

DMC11

DDCMP MD LN UNIT TST
CZDMEDO

AH-8553D-MC
FICHE 1 OF 1

NOV 1980
COPYRIGHT © 76-80
MADE IN USA



.REM 8

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

IDENTIFICATION

PRODUCT CODE: AC-8552D-MC
PRODUCT NAME: CZDMEDO DDCMP MD LN UNIT TST
DATE: AUGUST 1980
MAINTAINER: DIAGNOSTICS-MERRIMACK

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) 1976, 1980 BY DIGITAL EQUIPMENT CORPORATION

44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93

1. ABSTRACT

THE FUNCTION OF THE DMC11 DIAGNOSTICS IS TO VERIFY THAT THE OPTION OPERATES ACCORDING TO SPECIFICATIONS. THE DIAGNOSTICS VERIFY THAT THERE ARE NO MALFUNCTIONS AND THE ALL OPERATIONS OF THE DMC11 ARE CORRECT IN ITS ENVIRONMENT.

PARAMETERS MUST BE SET UP TO ALERT THE DIAGNOSTICS TO THE DMC11 CONFIGURATION. THESE PARAMETERS ARE CONTAINED IN THE STATUS TABLE AND ARE GENERATED IN TWO WAYS: 1) MANUAL INPUT - THE OPERATOR ANSWERS QUESTIONS. 2) AUTOSIZING - THE PROGRAM DETERMINES THE PARAMETERS AUTOMATICALLY.

CZDME TESTS THE DMC-11 LINE UNIT (M8201 OR M8202). IT PERFORMS WRITE/READ TESTS ON THE DMC LINE UNIT REGISTERS. IT CHECKS FOR PROPER TRANSMITTER, RECEIVER, AND BCC OPERATION IN DDCMP MODE. THE MODEM SIGNALS ARE ALSO CHECKED. CZDME REQUIRES A DMC MICRO-PROCESSOR (M8200 OR M8204) TO RUN. FOR BEST DIAGNOSIS A TURN-AROUND CONNECTOR SHOULD BE INSTALLED, HOWEVER THE DIAGNOSTIC WILL RUN WITHOUT IT (SOME TESTS ARE SKIPPED).

CURRENTLY THERE ARE FIVE OFF LINE DIAGNOSTICS THAT ARE TO BE RUN IN SEQUENCE TO INSURE THAT IF AN ERROR SHOULD OCCUR IT WILL BE DETECTED AT AN EARLY STAGE.

NOTE: ADDITIONAL DIAGNOSTICS MAY BE ADDED IN THE FUTURE.

THE FIVE DIAGNOSTICS ARE:

1. CZDMC [REV] BASIC W/R AND MICRO-PROCESSOR TESTS
2. CZDME [REV] DDCMP MODE LINE UNIT TESTS
3. DZDMF [REV] BITSTUFF LINE UNIT TESTS
4. DZDMG [REV] JUMP AND CROM TESTS
5. DZDMH [REV] FREE-RUNNING TESTS (HEAT TEST TAPE)

NOTE: THE NAMES OF THESE DIAGNOSTICS MAY VAYIE AS THEY ARE UPDATED.

2. REQUIREMENTS

2.1 EQUIPMENT

ANY PDP11 FAMILY CPU (EXCEPT AN .SI-11) WITH MINIMUM 8K MEMORY
ASR 33 (OR EQUILIVALENT)
DMC11-AR WITH DMC11-DA OR DMC11-FA OR
DMC11-AL WITH DMC11-MA OR DMC11-4D

94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136

2.2 STORAGE

PROGRAM WILL USE ALL 8K OF MEMORY EXCEPT WHERE ABL AND BOOTSTRAP LOADER RESIDE. LOCATIONS 1500 THRU 1640; CONTAIN THE "STATUS TABLE" INFORMATION WHICH IS GENERATED AT START OF DIAGNOSTICS BY MANUAL INPUT (QUESTIONS) OR AUTOMATICALLY (AUTO-SIZING). THIS AREA IS AN OVERLAY AREA AND SHOULD NOT BE ALTERED BY THE OPERATOR.

? LOADING PROCEDURE

3.1 METHOD

ALL PROGRAMS ARE IN ABSOLUTE FORMAT AND ARE LOADED USING THE ABSOLUTE LOADER. NOTE: IF THE DIAGNOSTICS ARE ON A MEDIA SUCH AS DISK ,MAGTAPE,DECTAPE, OR CASSETTE; FOLLOW INSTRUCTIONS FOR THE MONITOR WHICH HAS BEEN PROVIDED ON THAT SPECIFIC MEDIA.

ABSOLUTE LOADER STARTING ADDRESS *500

MEMORY * SIZE

4K	17
8K	37
12K	57
16K	77
20K	117
24K	137
28K	157

3.1.1 PLACE ADDRESS OF ABS LOADER INTO SWITCH REGISTER.
(ALSO PLACE 'HALT' SW UP)

3.1.2 DEPRESS 'LOAD ADDRESS' KEY ON CONSOLE AND RELEASE.

3.1.3 DEPRESS 'START KEY' ON CONSOLE AND RELEASE (PPROGRAM SHOULD NOW BE LOADING INTO CPU)

137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192

4. STARTING PROCEDURE

- A. SET SWITCH REGISTER TO 000200
- B. DEPRESS 'LOAD ADDRESS' KEY AND RELEASE
- C. SET SWR TO ZERO FOR 'AUTO SIZING' OR SWR BIT0=1 FOR MANUAL INPUT (QUESTIONS) OR SWR BIT7=1 TO USE EXISTING PARAMETERS SET UP BY A PREVIOUS START OR A PREVIOUSLY RUN DMC11 DIAGNOSTIC.
- D. DEPRESS 'START KEY' AND RELEASE. THE PROGRAM WILL TYPE MAINDEC NAME AND PROGRAM NAME (IF THIS WAS THE FIRST START UP OF THE PROGRAM) AND ALSO THE FOLLOWING:

MAP OF DMC11 STATUS

PC	CSR	STAT1	STAT2	STAT3
--	---	-----	-----	-----
001500	160010	145310	177777	000000
001510	160020	145320	177777	000000

THE PROGRAM WILL TYPE 'R' AND PROCEED TO RUN THE DIAGNOSTIC. THE ABOVE IS ONLY AN EXAMPLE. THIS WOULD INDICATE THE STATUS TABLE STARTING AT ADD. 1500 IN THE PROGRAM. IN THIS EXAMPLE THE TABLE CONTAINS THE INFORMATION AND STATUS OF TWO DMC11'S. THE STATUS TABLE MUST BE VERIFIED BY THE USER IF AUTO SIZING IS DONE. FOR INFORMATION OF STATUS TABLE SEE SECTION 8.4 FOR HELP.

IF THE DIAGNOSTIC WAS STARTED WITH SW00=1 INDICATING MANUAL PARAMETER INPUT THEN THE FOLLOWING SHOWS AN EXAMPLE OF THE QUESTIONS ASKED AND SOME EXAMPLE ANSWERS:

HOW MANY DMC11'S TO BE TESTED?1

01
CSR ADDRESS?160010
VECTOR ADDRESS?310
BR PRIORITY LEVEL? (4,5,6,7)?5
DOES MICRO-PROCESSOR HAVE CRAM? (Y OR N)N
WHICH LINE UNIT? IF NONE TYPE 'N', IF M8201 TYPE '1', IF M8202 TYPE '2'?1
IS THE LOOP BACK CONNECTOR ON?Y

(THE NEXT QUESTION WILL BE ASKED IF M8201 AND LOOP BACK CONNECTOR)

WHICH MODEM TYPE, TYPE 'D' FOR DMC11-DA (RS232C), OR TYPE 'F' FOR DMC11-FA (V.35) ? D
SWITCH PAC#1 (DDCMP LINE#)?377
SWITCH PAC#2 (BM873 BOOT ADD)?377

FOLLOWING THE QUESTIONS THE STATUS MAP IS PRINTED OUT AS DESCRIBED ABOVE, THE INFORMATION IN THE MAP REFLECTS THE ANSWERS TO THE QUESTIONS. IF THE DIAGNOSTIC WAS STARTED WITH

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

SW00=0 AND SW07=0 (AUTO-SIZING) THEN NO QUESTIONS ARE ASKED AND ONLY THE STATUS-MAP IS PRINTED OUT. IF AUTO-SIZING IS USED THE STATUS INFORMATION MUST BE VERIFIED TO BE CORRECT (MATCH THE HARDWARE). IF IT DOES NOT MATCH THE HARDWARE THE DIAGNOSTIC MUST BE RESTARTED WITH SW00=1 AND THE QUESTIONS ANSWERED.

4.1 CONTROL SWITCH SETTINGS

SW 15 SET: HALT ON ERROR
SW 14 SET: LOOP ON CURRENT TEST
SW 13 SET: INHIBIT ERROR PRINT OUT
SW 12 SET: INHIBIT TYPE OUT/ABELL ON ERROR.
SW 11 SET: INHIBIT ITERATIONS. (QUICK PASS)
SW 10 SET: ESCAPE TO NEXT TEST ON ERROR
SW 09 SET: LOOP WITH CURRENT DATA
SW 08 SET: CATCH ERROR AND LOOP ON IT
SW 07 SET: USE PREVIOUS STATUS TABLE.
SW 06 SET: HALT IN ROMCLK ROUTINE BEFORE CLOCKING
MICRO-PROCESSOR
SW 05 SET: RESERVED
SW 04 SET: RESERVED
SW 03 SET: RESELECT DMC11'S DESIRED ACTIVE
SW 02 SET: LOCK ON SELECTED TEST
SW 01 SET: RESTART PROGRAM AT SELECTED TEST
SW 00 SET: BUILD NEW STATUS TABLE FROM QUESTIONS. (IF SW07=0
AND SW00=0 A NEW STATUS TABLE IS BUILT BY
AUTO-SIZING)

SWITCH 06 AND 08-15 ARE DYNAMIC AND CAN BE CHANGED AS NEEDED WHILE THE DIAGNOSTIC IS RUNNING. SWITCHES 00-03 AND SWITCH 07 ARE STATIC, AND ARE USED ONLY ON STARTING OR RESTARTING THE DIAGNOSTIC.

227
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

4.1.2 SWITCH REGISTER OPTIONS (AT START UP)

SW 01 RESTART PROGRAM AT SELECTED TEST. IT IS STRONGLY SUGGESTED THAT AT LEAST ONE PASS HAS BEEN MADE BEFORE TRYING TO SELECT A TEST, THE REASON BEING IS THAT THE PROGRAM HAS TO CLEAR AREAS AND SET UP PARAMETERS. WHEN THIS SWITCH IS USED THE DIAGNOSTIC WILL ASK TEST NO.? ANSWER BY TYPING THE NUMBER OF THE TEST DESIRED AND CARRIGE RETURN TO BEGIN EXECUTION AT THE SELECTED TEST.

SW 02 LOCK ON SELECTED TEST. THIS SWITCH WHEN USED WITH SW01 WILL CAUSE THE PROGRAM TO CONSTANTLY LOOP ON THE SELECTED TEST. HITTING ANY KEY ON THE CONSOLE WILL LET IT ADVANCE TO THE NEXT TEST AND LOOP UNTIL A KEY IS HIT AGAIN. IF SW02=0 WHEN SW01 IS USED. THE PROGRAM WILL BEGIN AT THE SELECTED TEST AND CONTINUE NORMAL OPERATIONS.

SW 03 RESELECT DMC11'S DESIRED ACTIVE. PLEASE NOTE THAT A MESSAGE IS TYPED OUT FOR SETTING THE SWITCH REGISTER EQUAL TO DMC11'S ACTIVE. THIS MEANS IF THE SYSTEM HAS FOUR DMC11S; BITS 00,01,02,03 WILL BE SET IN LOC 'DMACTV' FROM THE SWITCH REGISTER. USING THIS SWITCH(SW00) ALTERS THAT LOCATION; THEREFORE IF FOUR DMC11S ARE IN THE SYSTEM ***DO NOT*** SET SWITCHS GREATER THAN SW 03 IN THE UP POSITION. THIS WOULD BE A FATAL ERROR. DO NOT SELECT MORE ACTIVE DMC11S THAN THERE IS INFORMATION ON IN THE STATUS TABLE.

METHOD: A: LOAD ADDRESS 200
B: START WITH SW 00=1
C: PROGRAM WILL TYPE MESSAGE
D: SET A SWITCH FOR EACH DMC DESIRED ACTIVE.
EXAMPLE: IF YOU HAVE 4 DMC'S BUT ONLY WANT TO RUN THE FIRST AND THE LAST SET SWR BITS 0 AND 3 = 1. PRESS CONTINUE
E: NUMBER (IF VALID) WILL BE IN DATA LIGHTS (EXCLUDING 11/05)
F: SET WITH ANY OTHER SWITCH SETTINGS DESIRED. PRESS CONTINUE.

270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317

4.1.3 DYNAMIC SWITCHES

ERROR SWITCHES

1. SW 12 DELETE PRINT OUT/BELL ON ERROR.
2. SW 13 DELETE ERROR PRINTOUT.
3. SW 15 HALT ON THE ERROR.
4. SW 08 GOTO BEGINNING OF THE TEST(ON ERROR).
5. SW 10 GOTO NEXT TEST(ON ERROR).

SCOPE SWITCHES

1. SW06 HALT IN ROMCLK ROUTINE BEFORE CLOCKING MICRO-PROCESSOR INSTRUCTION. THIS ALLOWS THE OPERATOR TO SCOPE A MICRO-PROCESSOR INSTRUCTION IN THE STATIC STATE BEFORE IT IS CLOCKED. HIT CONTINUE TO RESUME RUNNING.
2. SW09 (IF ENABLED BY 'SCOPI') ON AN ERROR; IF AN '*' IS PRINTED IN FRONT OF THE TEST NO. (EX. *TEST NO. 10) SW09 IS INCORPORATED IN THAT TEST AND THEREFORE SW09 IS USUALLY THE BEST SWITCH FOR THE SCOPE LOOP (SW14=0, SW10=0, SW09=1, SW08=0). IF SW09 IS NOT ENABLED; AND THERE IS A HARD ERROR (CONSTANT); SW08 IS BEST. (SW14=1,0, SW10=0, SW09=0, SW08=1). FOR INTERMITTENT ERRORS; SW14=1 WILL LOOP ON TEST REGARDLESS OF ERROR OR NOT ERROR. (SW14=1, SW10=0, SW09=0, SW08=1,0)
3. SW11 INHIBIT INTERATIONS.
4. SW14 LOOP ON CURRENT TEST.

4.2 STARTING ADDRESS

STARTING ADDRESS IS AT 000200 THERE ARE NO OTHER STARTING ADDRESSES FOR THE DMC11 DIAGNOSTICS. (SEE SECTION 4.0)

NOTE: IF ADDRESS 000042 IS NON-ZERO THE PROGRAM ASSUMES IT IS UNDER ACT11 OR XXDP CONTROL AND WILL ACT ACCORDINGLY AFTER ALL AVAILABLE DMC11'S ARE TESTED THE PROGRAM WILL RETURN TO 'XXDP' OR 'ACT-11'.

5. OPERATING PROCEDURE

WHEN PROGRAM IS INITIALLY STARTED MESSAGES AS DESCRIBED IN SECTION 4.0 WILL BE PRINTED, AND PROGRAM WILL BEGIN RUNNING THE DIAGNOSTIC

318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365

5.2 PROGRAM AND/OR OPERATOR ACTION

THE TYPICAL APPROACH SHOULD BE

1. HALT ON ERROR (VIA SW 15=1) WHEN EVER AN ERROR OCCURS.
2. CLEAR SW 15.
3. SET SW 14: (LOOP ON THIS TEST)
4. SET SW 13: (INHIBIT ERROR PRINT OUT)

THE TEST NUMBER AND PC WILL BE TYPED OUT AND POSSIBLY AN ERROR MESSAGE (THIS DEPENDS ON THE TEST) TO GIVE THE OPERATOR AN IDEA AS TO THE SOURCE OF THE PROBLEM. IF IT IS NECESSARY TO KNOW MORE INFORMATION CONCERNING THE ERROR REPORT; LOOK IN THE LISTING FOR THAT TEST NUMBER WHICH WAS TYPED OUT AND THEN NOTE THE PC OF THE ERROR REPORT THIS WAY THE EXACT FUNCTION OF THE TEST CAN BE DETERMINED.

6. ERRORS

AS DESCRIBED PREVIOUSLY THERE WILL ALWAYS BE A TEST NUMBER AND PC TYPED OUT AT THE TIME OF AN ERROR (PROVIDING SW 13=0 AND SW 12=0). IN MOST CASES ADDITIONAL INFORMATION WILL BE SUPPLIED IN THE THE ERROR MESSAGE TO GIVE THE OPERATOR AN INDICATION OF THE ERROR.

6.2 ERROR RECOVERY

IF FOR SOME REASON THE DMC11 SHOULD 'HANG THE BUS' (GAIN CONTROL OF BUS SO THAT CONSOLE MANUAL FUNCTIONS ARE INHIBITED) AN INIT OR POWER DOWN/UP IS NECESSARY FOR OPERATOR TO REGAIN CONTROL OF CPU. IF THIS SHOULD HAPPEN; LOOK IN LOCATION 'TSTNO' (ADDRESS 1226)FOR THE NUMBER OF THE TEST THAT WAS RUNNING AT THE TIME OF THE CATASTROPHIC ERROR. IN THIS WAY THE OPERATOR WILL HAVE AN IDEA AS TO WHAT THE DMC11 WAS DOING AT THE TIME OF THE ERROR.

7. RESTRICTIONS

7.1 STARTING RESTRICTIONS

SEE SECTION 4. (PLEASE)
STATUS TABLE SHOULD BE VERIFIED REGARDLESS OF HOW PROGRAM WAS STARTED. ALSO IT IS IMPORTANT TO USE THIS LISTING ALONG WITH THE INFORMATION PRINTED ON THE TTY TO COMPLETELY ISOLATE PROBLEMS.

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
418
419
420
421

7.2 OPERATING RESTRICTIONS

THE FIRST TIME A DMC11 DIAGNOSTIC IS LOADED INTO CORE AND RUN THE STATUS TABLE MUST BE SET UP. THIS IS DONE BY MANUAL INPUT (SW00=1) OR BY AUTOSIZING (SW00=0 AND SW07=0). THEREAFTER HOWEVER THE STATUS TABLE NEED NOT BE SETUP BY SUBSEQUENT RESTARTS OR EVEN LOADING THE NEXT DMC DIAGNOSTIC BECAUSE THE STATUS TABLE IS OVERLAYED. THE CURRENT PARAMETERS IN THE STATUS TABLE ARE USED WHEN SW07=1 ON START UP.

7.3 HARDWARE CONFIGURATION RESTRICTIONS

DMC11(M8200)- JUMPER W1 MUST BE IN, AND SWITCH 7 OF E76 MUST BE IN THE OFF POSITION.

KMC(M8204)- JUMPER W1 MUST BE IN.

LINE UNIT(M8201)- JUMPERS W1, W2, AND W4 MUST BE IN. JUMPERS W3, AND W5 MUST BE OUT. SW8 OF E26 MUST BE IN THE ON POSITION.

LINE UNIT (M8202)- JUMPER W1 MUST BE IN. SW8 OF E26 MUST BE IN THE OFF POSITION.

8. MISCELLANEOUS

8.1 EXECUTION TIME

ALL DMC11 DEVICE DIAGNOSTICS WILL GIVE AN 'END PASS' MESSAGE (PROVIDING NO ERRORS AND SW12=0) WITHIN 4 MINS. THIS IS ASSUMING SW11=1 (DELETE ITERATIONS) IS SET TO GIVE THE FASTEST POSSIBLE EXECUTION. THE ACTUAL EXECUTION TIME DEPENDS GREATLY ON THE PDP11 CPU CONFIGURATION AND THE AMOUNT OF MEMORY IN THE SYSTEM.

8.2 PASS COMPLETE

NOTE: EVERY TIME THE PROGRAM IS STARTED; THE TESTS WILL RUN AS IF SW11 (DELETE ITERATIONS) WAS UP (=1). THIS IS TO 'VERIFY NO HARD ERRORS' AS SOON AS POSSIBLE. THEREFORE THE FIRST PASS -EACH TIME PROGRAM IS STARTED- WILL BE A 'QUICK PASS' UNTIL ALL DMC11'S IN SYSTEM ARE TESTED. WHEN THE DIAGNOSTIC HAS COMPLETED A PASS THE FOLLOWING IS AN EXAMPLE OF THE PRINT OUT TO BE EXPECTED.

END PASS CZDME CSR: 175000 VEC: 0300 PASSES: 000001
ERRORS: 000000

NOTE: THE PASS COUNT AND ERROR COUNTS ARE CUMMULITIVE FOR EACH DMC11 THAT IS RUNNING, AND ARE SET TO ZERO ONLY WHEN THE DIAGNOSTIC IS STARTED. THEREFORE AFTER AN OVERNIGHT RUN FOR EXAMPLE, THE TOTAL PASSES AND ERRORS FOR EACH DMC11 SINCE THE DIAGNOSTIC WAS STARTED ARE REFLECTED IN PASSES: AND ERRORS:.

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
475
476
477

8.4 KEY LOCATIONS

RETURN (1214) CONTAINS THE ADDRESS WHERE PROGRAM WILL RETURN WHEN ITERATION COUNT IS REACHED OR IF LOOP ON TEST IS ASSERTED.

NEXT (1216) CONTAINS THE ADDRESS OF THE NEXT TEST TO BE PERFORMED.

TSTNO (1226) CONTAINS THE NUMBER OF THE TEST NOW BEING PERFORMED.

RUN (1316) THE BIT IN 'RUN' ALWAYS POINTS TO THE DMC11 CURRENTLY BEING TESTED. EXAMPLE: (RUN) 1302/00000000100000 MEANS THAT DMC11 NO.06 IS THE DMC11 NOW RUNNING.

DMC00-DMC17
DMST00-DMST17
(1500)-(1640)

THESE LOCATIONS CONTAIN THE INFORMATION NEEDED TO TEST UP TO 16 (DECIMAL) DMC11S SEQUENTIALLY. THEY CONTAIN THE CSR, VECTOR AND STATUS CONCERNING THE CONFIGURATION OF EACH DMC11.

DMACTV (1306) EACH BIT SET IN THIS LOCATION INDICATES THAT THE ASSOCIATED DMC11 WILL BE TESTED IN TURN. EXAMPLE: (DMACTV) 1276/000000000011111 MEANS THAT DMC11 NO. 00,01,02,03,04 WILL BE TESTED. EXAMPLE: (DMACTV) 1276/000000000010001 MEANS THAT DMC11 NO. 00,04 WILL BE TESTED.

DMCSR (1402) CONTAINS THE CSR OF THE CURRENT DMC11 UNDER TEST.

8.4A 'STATUS TABLE' (1500-1640)

THE TABLE IS FILLED BY AUTO SIZING OR BY THE MANUAL PARAMETER INPUT (QUESTIONS) AS DESCRIBED PREVIOUSLY. ALSO IF DESIRED BY USER; THE LOCATIONS MAY BE ALTERED BY HAND (TOGGLED IN) TO SUIT THE SPECIFIC CONFIGURATION.

THE EXAMPLE STATUS MAP SHOWN BELOW CONTAINS INFORMATION FOR TWO DMC11'S. THE TABLE CAN CONTAIN UP TO 16 DMC11'S. FOLLOWING THE MAP IS A DESCRIPTION OF THE BITS FOR EACH MAP ENTRY

MAP OF DMC11 STATUS

PC	CSR	STAT1	STAT2	STAT3
001500	160010	145310	177777	000000
001510	160020	016320	000000	000000

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

EACH MAP ENTRY CONTAINS 4 WORDS WHICH CONTAIN THE STATUS INFORMATION FOR 1 DMC11. THE PC SHOWS WHERE IN CORE MEMORY THE FIRST OF THE 4 WORDS IS. IN THE EXAMPLE ABOVE THE FIRST DMC'S STATUS IS IN LOCATIONS, 1500, 1502, 1504, AND 1506. THE SECOND DMC STATUS IS LOCATED AT 1510, 1512, 1514, AND 1516. THE INFORMATION CONTAINED IN EACH 4 WORD ENTRY IS DEFINED AS FOLLOWS:

CSR: CONTAINS DMC11 CSR ADDRESS

STAT1: BITS 00-08 IS DMC11 VECTOR ADDRESS
BIT15=1 MICRO-PROCESSOR HAS CRAM
BIT15=0 MICRO-PROCESSOR HAS CROM
BIT14=1 TURNAROUND CONNECTOR IS ON
BIT14=0 NO TURNAROUND CONNECTOR
BIT13=0 LINE UNIT IS AN M8201
BIT13=1 LINE UNIT IS AN M8202
BIT12=1 NO LINE UNIT
BITS 09-11 IS DMC11 BR PRIORITY LEVEL

STAT2: LOW BYTE IS SWITCH PAC#1 (DDCMP LINE NUMBER)
HIGH BYTE IS SWITCH PAC#2 (BM873 BOOT ADD)

STAT3: BIT0=1 RUN FREE RUNNING TESTS ON KMC11
BIT1=0 DMC11-AR (LOW SPEED)
BIT1=1 DMC11-AL (HIGH SPEED)
BIT2=0 DMC11-DA (RS232)
BIT2=1 DMC11-FA (V.35)

510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
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

8.5 METHOD OF AUTO SIZING

8.5.1 FINDING THE CONTROL STATUS REGISTER.

THE AUTO-SIZING ROUTINE FINDS A DMC11 AS FOLLOWS: IT STARTS AT ADDRESS 160000 AND TESTS ALL ADDRESS IN INCREMENTS OF 10 UP TO AND INCLUDING ADDRESS 167760. IF THE ADDRESS DOES NOT TIME OUT, THE FOLLOWING IS DONE, THE FIRST CROM ADDRESS IS WRITTEN TO A 125252 THEN IT IS READ BACK. IF IT CONTAINS A -1 OR 125252 OR 626 OR 16520 A DMC11 HAS BEEN FOUND, IF NOT, THE ADDRESS IS UPDATED BY 10 AND THE SEARCH CONTINUES. A -1 INDICATES A DMC11 WITH NO CROM OR CROM, A 125252 INDICATES A KMC11 WITH CROM, A 626 INDICATES A DMC11-AL AND A 16520 INDICATES A DMC11-AR. FURTHER TESTS ARE PERFORMED AT THIS POINT TO DETERMINE WHICH LINE UNIT, IF ANY, IS INSTALLED, IF A LOOP-BACK CONNECTOR IS INSTALLED AND VARIOUS SWITCH SETTINGS ON THE LINE UNIT. THIS IS WHY THE STATUS TABLE MUST BE VERIFIED BY THE USER AND IF ANY OF THE INFORMATION DOES NOT AGREE WITH THE HARDWARE THE DIAGNOSTIC MUST BE RESTARTED AND THE QUESTIONS MUST BE ANSWERED. ALL DMC11'S IN THE SYSTEM WILL BE FOUND BY THE AUTO-SIZER. IF IT DOES NOT FIND A DMC11 THE DIAGNOSTIC MUST BE RESTARTED AND THE QUESTIONS ANSWERED.

8.5.2 FINDING THE VECTOR AND BR LEVEL

THE VECTOR AREA (ADDRESS 300-776) IS FILLED WITH THE INSTRUCTION IOT AND '+2' (NEXT ADDRESS). THE PROCESSOR STATUS IS STARTED AT 7 AND THE DMC IS PROGRAMMED TO INTERRUPT. THE PS IS LOWERED BY 1 UNTIL THE DMC INTERRUPTS, A DELAY IS MADE AND IF NO INTERRUPT OCCURS AT PS LEVEL 3 (BECAUSE OF A BAD DMC11) THE PROGRAM ASSUMES VECTOR ADDRESS 300 AT BR LEVEL 5 AND THE PROBLEM SHOULD BE FIXED IN THE DIAGNOSTIC. ONCE THE PROBLEM IS FIXED; THE PROGRAM SHOULD BE RE-SETUP AGAIN TO GET CORRECT VECTOR. IF AN INTERRUPT OCCURED; THE ADDRESS TO WHICH THE DMC11 INTERRUPTED TO IS PICKED UP AND REPORTED AS THE VECTOR. NOTE: IF THE VECTOR REPORTED IS NOT THE VECTOR SET UP BY YOU; THERE IS A PROBLEM AND AUTO SIZING SHOULD NOT BE DONE.

8.6 SOFTWARE SWITCH REGISTER

IF THE DIAGNOSTIC IS RUN ON AN 11/04 OR OTHER CPU WITHOUT A SWITCH REGISTER THEN A SOFTWARE SWITCH REGISTER IS USED TO ALLOW USER THE SAME SWITCH OPTIONS AS DESCRIBED PREVIOUSLY. IF THE HARDWARE SWITCH REGISTER DOES NOT EXIST OR IF ONE DOFS AND IT CONTAINS ALL ONES (177777) THIS SOFTWARE SWITCH REGISTER IS USED.

CONTROL:

TO OBTAIN CONTROL AT ANY ALLOWABLE TIME DURING EXECUTION OF THE DIAGNOSTIC THE OPERATOR TYPES A CTRL G ON THE CONSOLE TERMINAL KEYBOARD. AS SOON AS THE CTRL G IS RECOGNIZED, BY THE DIAGNOSTIC, THE FOLLOWING MESSAGE WILL BE DISPLAYED:

566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610

SWR=XXXXXX NEW?

WHERE XXXXXX IS THE CURRENT CONTENTS OF THE SOFTWARE SWITCH REGISTER IN OCTAL. THE SOFTWARE CONTROL ROUTINE WILL THEN AWAIT OPERATOR ACTION. AT WHICH TIME THE OPERATOR IS REQUIRED TO TYPE ONE OR MORE OF THE LEGAL CHARACTERS: 1) 0 - 7, 2) LINE FEED(<LF>), 3) CARRIAGE RETURN(<CR>), OR 4) CONTROL-U (CTRL U). NO CHECK IS MADE FOR LEGALITY. IF THE INPUT CHARACTER IS NOT A <LF>, <CR>, OR CTRL U IT IS ASSUMED TO BE AN OCTAL DIGIT.

TO CHANGE THE CONTENTS OF THE SSR THE OPERATOR SIMPLY TYPES THE NEW DESIRED VALUE IN OCTAL - LEADING ZEROS NEED NOT BE TYPED. AND TERMINATES THE INPUT STRING WITH A <CR> OR <LF> DEPENDING ON THE PROGRAM ACTION DESIRED AS DESCRIBED BELOW. THE INPUT VALUE WILL BE TRUNCATED TO THE LAST 6 DIGITS TYPED. AT LEAST ONE DIGIT MUST BE TYPED ON ANY GIVEN INPUT STRING PRIOR TO THE TERMINATOR BEFORE A CHANGE TO THE SSR WILL OCCUR.

WHEN THE INPUT STRING IS TERMINATED WITH A <CR> THE DIAGNOSTIC WILL CONTINUE EXECUTION FROM THE POINT AT WHICH IT WAS INTERRUPTED. IF A <CR> IS THE ONLY THING TYPED THE PROGRAM WILL CONTINUE WITHOUT CHANGING THE SSR. THE <LF> DIFFERS FROM THE <CR> BY RESTARTING THE PROGRAM AS IF IT WERE RESTARTED AT ADDRESS 200.

IF A CTRL U IS TYPED AT ANY POINT IN THE INPUT STRING PRIOR TO THE TERMINATOR THE INPUT VALUE WILL BE DISREGARDED AND THE PROMPT DISPLAYED (SWR = XXXXXX NEW?).

TO SET THE SSP FOR THE STARTING SWITCHES, FIRST LOAD THE DIAGNOSTIC, THEN HIT CTRL G, THEN STAR THE DIAGNOSTIC.

9.0 HISTORY

THIS DIAGNOSTIC WAS UPDATED TO DETECT FOR THE CONDITION OF V.35 AND M8201. IN THIS CONIFURATION, RING WILL NOT BE LOOPED BACK AND SHOULD NOT BE TESTED FOR.

8

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
646
647
648
649
650
651
652
653
654
655

:*AC-8552D-MC DMC11 DDCMP MODE LINE UNIT TESTS
:*COPYRIGHT 1976,1978, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
:-----

:STARTING PROCEDURE
:LOAD PROGRAM
: OAD ADDRESS 000200
:SWR=0 AUTOSIZE DMC11
:SW07=1 USE CURRENT DMC11 PARAMETERS
:SW00=1 INPUT NEW DMC11 PARAMETERS
:PRESS START
:PROGRAM WILL TYPE "AC-8552D-MC DMC11 DDCMP MODE LINE UNIT TESTS"
:PROGRAM WILL TYPE STATUS MAP
:PROGRAM WILL TYPE "R" TO INDICATE THAT TESTING HAS STARTED
:AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE
:AND THEN RESUME TESTING
:SUBSEQUENT RESTARTS WILL NOT TYPE PROGRAM TITLE

:SWITCH REGISTER OPTIONS
:-----

100000	SW15=100000	=1,HALT ON ERROR
040000	SW14=40000	=1,LOOP ON CURRENT TEST
020000	SW13=20000	=1,INHIBIT ERROR TIMEOUT
010000	SW12=10000	=1,DELETE TIMEOUT/BELL ON ERROR.
004000	SW11=4000	=1,INHIBIT ITERATIONS
002000	SW10=2000	=1,ESCAPE TO NEXT TEST ON ERROR
001000	SW09=1000	=1,LOOP WITH CURRENT DATA
000400	SW08=400	=1,LOOP ON ERROR
000200	SW07=200	=1,USE CURRENT DMC11 PARAMETERS, =0,AUTOSIZE DMC11
000100	SW06=100	=1, HALT BEFORE CLOCKING MICRO-PROCESSOR INSTRUCTION
000040	SW05=40	
000020	SW04=20	
000010	SW03=10	:RESELECT DMC11'S TO BE TESTED (ACTIVE)
000004	SW02=4	:LOCK ON TEST SELECT
000002	SW01=2	:RESTART PROGRAM AT SELECTED TEST
000001	SW00=1	:INPUT DMC11 PARAMETERS

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
703
704
705
706
707

000000
000001
000002
000003
000004
000005
000006
000007

177776
001200

005746
005726
010046
012600
024646
022626

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

;REGISTER DEFINITIONS

R0=%0 ;GENERAL REGISTER
R1=%1 ;GENERAL REGISTER
R2=%2 ;GENERAL REGISTER
R3=%3 ;GENERAL REGISTER
R4=%4 ;GENERAL REGISTER
R5=%5 ;GENERAL REGISTER
SP=%6 ;PROCESSOR STACK POINTER
PC=%7 ;PROGRAM COUNTER

;LOCATION EQUIVALENCIES

PS=177776 ;PROCESSOR STATUS WORD
STACK=1200 ;START OF PROCESSOR STACK

;INSTRUCTION DEFINITIONS

PUSH1SP=5746 ;DECREMENT PROCESSOR STACK 1 WORD
POP1SP=5726 ;INCREMENT PROCESSOR STACK 1 WORD
PUSHRO=10046 ;SAVE R0 ON STACK
POPPO=12600 ;RESTORE R0 FROM STACK
PUSH2SP=24646 ;DECREMENT STACK TWICE
POP2SP=22626 ;INCREMENT STACK TWICE
.EQUIV EMT,HLT ;BASIC DEFINITION OF ERROR CALL

;BIT DEFINITIONS

BIT15=100000
BIT14=40000
BIT13=20000
BIT12=10000
BIT11=4000
BIT10=2000
BIT9=1000
BIT8=400
BIT7=200
BIT6=100
BIT5=40
BIT4=20
BIT3=10
BIT2=4
BIT1=2
BIT0=1

TRAPCATCHER FOR UNEXPECTED INTERUPTS

SEQ 0016

```

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

```

```

:*****
:-----
:TRAPCATCHER FOR ILLEGAL INTERRUPTS
:THE STANDARD "TRAP CATCHER" IS PLACED
:BEWEEN ADDRESS 0 TO ADDRESS 776.
:IT LOOKS LIKE "PC+2 HALT".
:-----
:*****

.=0
:STANDARD INTERRUPT VECTORS
:-----

.=24
.PFAIL          ;POWER FAIL HANDLER
340             ;SERVICE AT LEVEL 7
.HLT            ;ERROR HANDLER
340             ;SERVICE AT LEVEL 7
.TRPSRV        ;GENERAL HANDLER DISPATCH SERVICE
340             ;SERVICE AT LEVEL 7

.-40
0               ;SAVE FOR ACT-11 OR XXDP
0               ;RETURN ADDRESS IF UNDER ACT-11 OR XXDP
0               ;SAVE FOR ACT-11 OR XXDP
SENDAD         ;FOR USE WITH ACT-11 OR XXDP

.=52
0               ;ACT-11 PROGRAM CHARACTERISTICS

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

.=200
JMP .START    ;GO TO START OF PROGRAM

.=1000
MTITLE: .ASCII <377><12>/AC-8552D-MC/<377>
        .ASCIIZ /DMC11 DDCMP MODE LINE UNIT TESTS/<377>

.=1200

:INDIRECT POINTERS TO SWITCH REGISTER AND LIGHT DISPLAY
:-----

DISPLAY:177570
SWR: 177570

```



```
754  
755 ;INDIRECT POINTERS TO TELETYPE VECTORS AND REGISTERS  
756 ;-----  
757  
758 001204 177560 TKCSR: 177560 ;TELETYPE KEYBOARD CONTROL REGISTER  
759 001206 177562 TKDBR: 177562 ;TELETYPE KEYBOARD DATA BUFFER  
760 001210 177564 TPCSR: 177564 ;TELEPRINTER CONTROL REGISTER  
761 001212 177566 TPDBR: 177566 ;TELEPRINTER DATA BUFFER  
762  
763 ;PROGRAM CONTROL PARAMETERS  
764 ;-----  
765  
766 001214 000000 RETURN: 0 ;SCOPE ADDRESS FOR LOOP ON TEST  
767 001216 000000 NEXT: 0 ;ADDRESS OF NEXT TEST TO BE EXECUTED  
768 001220 000000 LOCK: 0 ;ADDRESS FOR LOCK ON CURRENT DATA  
769 001222 000003 ICOUNT: 3 ;NUMBER OF ITERATIONS THAT CURRENT TEST WILL BE EXECUTED  
770 001224 000000 LPCNT: 0 ;NUMBER OF ITERATIONS COMPLETED  
771 001226 000000 TSTNO: 0 ;NUMBER OF TEST IN PROGRESS  
772 001230 000000 PASCNT: 0 ;NUMBER OF PASSES COMPLETED  
773 001232 000000 ERRCNT: 0 ;TOTAL NUMBER OF ERRORS  
774 001234 000000 LSTERR: 0 ;PC OF LAST ERROR CALL  
775  
776 ;PROGRAM VARIABLES  
777 ;-----  
778  
779 001236 000000 STRTSW: 0 ;SWITCHES AT START OF PROGRAM  
780 001240 000000 STAT: 0 ;DM STATUS WORD STORAGE  
781 001242 000000 CLXX: 0  
782 001244 000000 MASKX: 0  
783 001246 000000 TEMP1: 0 ;TEMPORARY STORAGE  
784 001250 000000 TEMP2: 0 ;TEMPORARY STORAGE  
785 001252 000000 TEMP3: 0 ;TEMPORARY STORAGE  
786 001254 000000 TEMP4: 0 ;TEMPORARY STORAGE  
787 001256 000000 TEMP5: 0 ;TEMPORARY STORAGE  
788 001260 000000 SAVR0: 0 ;R0 STORAGE  
789 001262 000000 SAVR1: 0 ;R1 STORAGE  
790 001264 000000 SAVR2: 0 ;R2 STORAGE  
791 001266 000000 SAVR3: 0 ;R3 STORAGE  
792 001270 000000 SAVR4: 0 ;R4 STORAGE  
793 001272 000000 SAVR5: 0 ;R5 STORAGE  
794 001274 000000 SAVSP: 0 ;STACK POINTER STORAGE  
795 001276 000000 SAVPC: 0 ;PROGRAM COUNTER STORAGE  
796 001300 000000 ZERO: 0  
797 001302 000001 ONE: 1  
798 001304 000000 MEMLIM: 0 ;HIGHEST LOCATION FOR NPR'S  
799 001306 000001 DMACTV: .BLKW 1 ;DMC11'S SELECTED ACTIVE.  
800 001310 000001 DMNUM: .BLKW 1 ;OCTAL NUMBER OF DMC11'S.  
801 001312 000001 SAVACT: .BLKW 1 ;ORIGINAL ACTV DEVICES  
802 001314 000001 SAVNUM: .BLKW 1 ;WORKABLE NUMBER  
803 001316 000000 RUN: 0 ;POINTER TO RUNNING DEVICE.  
804 .EVEN  
805 001320 001472 CREAM: DM.MAP-6 ;TABLE POINTER.  
806 001322 001676 MILK: CNT.MAP-4 ;TABLE POINTER
```

```

807
808 ;PROGRAM CONTROL FLAGS
809 ;-----
810
811 001324 000 INIFLG: .BYTE 0 ;PROGRAM INITIALIZATION FLAG
812 001325 000 ERRFLG: .BYTE 0 ;ERROR OCCURED FLAG
813 001326 000 LOKFLG: .BYTE 0 ;LOCK ON CURRENT TEST FLAG
814 001327 000 QV.FLG: .BYTE 0 ;QUICK VERIFY FLAG.
815 ;ON FIRST PASS OF EACH DMC11 ITERATIONS WILL BE SUPPRESS
816
817
818 ;DEFINITIONS FOR TRAP SUBROUTINE CALLS
819 ;POINTERS TO SUBROUTINES CAN BE FOUND
820 ;IN THE TABLE IMMEDIATLY FOLLOWING THE DEFINITIONS
821
822 ;*****
823 ;-----
824 001330
825 104400 .TRPTAB:
826 001330 003576 SCOPE=TRAP+0 ;CALL TO SCOPE LOOP AND ITERATION HANDLER
827 104401 .SCOPE
828 001332 003736 SCOPI=TRAP+1 ;CALL TO LOOP ON CURRENT DATA HANDLER
829 104402 .SCOPI
830 001334 003766 TYPE=TRAP+2 ;CALL TO TELETYPE OUTPUT ROUTINE
831 104403 .TYPE
832 001336 004050 INSTR=TRAP+3 ;CALL TO ASCII STRING INPUT ROUTINE
833 104404 .INSTR
834 001340 004154 INSTER=TRAP+4 ;CALL TO INPUT ERROR HANDLER
835 104405 .INSTER
836 001342 004174 PARAM=TRAP+5 ;CALL TO NUMERICAL DATA INPUT ROUTINE
837 104406 .PARAM
838 001344 004374 SAV05=TRAP+6 ;CALL TO REGISTER SAVE ROUTINE
839 104407 .SAV05
840 001346 004434 RES05=TRAP+7 ;CALL TO REGISTER RESTORE ROUTINE
841 104410 .RES05
842 001350 004466 CONVRT=TRAP+10 ;CALL TO DATA OUTPUT ROUTINE
843 104411 .CONVRT
844 001352 004472 CNVRT=TRAP+11 ;CALL TO DATA OUTPUT ROUTINE WITHOUT CR/LF.
845 104412 .CNVRT
846 001354 005466 MSTCLR=TRAP+12 ;CALL TO ISUE A MASTER CLEAR
847 104413 .MSTCLR
848 001356 005436 DELAY=TRAP+13 ;CALL TO DELAY
849 104414 .DELAY
850 001360 005504 ROMCLK=TRAP+14 ;CALL TO CLOCK ROM ONCE
851 104415 .ROMCLK
852 001362 005552 DATACLK=TRAP+15 ;CALL TO CLK DATA
853 104416 .DATACLK
854 001364 005616 TIMER=TRAP+16 ;CALL TO DELAY A CLOCK TICK
855 .TIMER
856 ;-----
857 ;*****

```

```

858 ;DMC11 CONTROL INDICATORS FOR CURRENT DMC11 UNDER TEST
859 ;-----
860
861 001366 000000 STAT1: 0
862 001370 000000 STAT2: 0
863 001372 000000 STAT3: 0
864
865 ;DMC11 VECTOR AND REGISTER INDIRECT POINTERS
866 ;-----
867
868 001374 000000 DMRVEC: 0 ; POINTER TO DMC11 RECEIVER INTERRUPT VECTOR
869 001376 000000 DMRLVL: 0 ; POINTER TO DMC11 RECEIVER INTERRUPT SERVICE PS
870 001400 000000 DMTVEC: 0 ; POINTER TO DMC11 TRANSMITTER INTERRUPT VECTOR
871 001402 000000 DMTLVL: 0 ; POINTER TO DMC11 TRANSMITTER INTERRUPT SERVICE PS
872 001404 000000 DMCSR: 0 ; POINTER TO DMC11 CONTROL STATUS REGISTER
873 001406 000000 DMCSRH: 0 ; POINTER TO DMC11 CONTROL STATUS REGISTER HIGH BYTE.
874 001410 000000 DMCTL: 0 ; POINTER TO DMC11 CONTROL OUT REGISTER
875 001412 000000 DMP04: 0 ; POINTER TO DMC11 PORT REGISTER(SEL 4)
876 001414 000000 DMP06: 0 ; POINTER TO DMC11 PORT REGISTER(SEL 6)
877
878 ;TEMP STORAGE
879 ;-----
880
881 001416 000000 TEMP: 0
882 001460 .=. +40
883
884 ;DMC11 STATUS TABLE AND ADDRESS ASSIGNMENTS
885 ;-----
886
887 . =1500
888 001500 DM.MAP:
889 001500 000001 DMC00: .BLKW 1 ; CONTROL STATUS REGISTER FOR DMC11 NUMBER 00
890 001502 000001 DMS100: .BLKW 1 ; VECTOR FOR DMC11 NUMBER 00
891 001504 000001 DMS200: .BLKW 1 ; DDCMP LINE# FOR DMC11 NUMBER 00
892 001506 000001 DMS300: .BLKW 1 ; 3RD STATUS WORD
893
894 001510 000001 DMC01: .BLKW 1 ; CONTROL STATUS REGISTER FOR DMC11 NUMBER 01
895 001512 000001 DMS101: .BLKW 1 ; VECTOR FOR DMC11 NUMBER 01
896 001514 000001 DMS201: .BLKW 1 ; DDCMP LINE# FOR DMC11 NUMBER 01
897 001516 000001 DMS301: .BLKW 1 ; 3RD STATUS WORD
898
899 001520 000001 DMC02: .BLKW 1 ; CONTROL STATUS REGISTER FOR DMC11 NUMBER 02
900 001522 000001 DMS102: .BLKW 1 ; VECTOR FOR DMC11 NUMBER 02
901 001524 000001 DMS202: .BLKW 1 ; DDCMP LINE# FOR DMC11 NUMBER 02
902 001526 000001 DMS302: .BLKW 1 ; 3RD STATUS WORD
903
904 001530 000001 DMC03: .BLKW 1 ; CONTROL STATUS REGISTER FOR DMC11 NUMBER 03
905 001532 000001 DMS103: .BLKW 1 ; VECTOR FOR DMC11 NUMBER 03
906 001534 000001 DMS203: .BLKW 1 ; DDCMP LINE# FOR DMC11 NUMBER 03
907 001536 000001 DMS303: .BLKW 1 ; 3RD STATUS WORD
908
909 001540 000001 DMC04: .BLKW 1 ; CONTROL STATUS REGISTER FOR DMC11 NUMBER 04
910 001542 000001 DMS104: .BLKW 1 ; VECTOR FOR DMC11 NUMBER 04
911 001544 000001 DMS204: .BLKW 1 ; DDCMP LINE# FOR DMC11 NUMBER 04
912 001546 000001 DMS304: .BLKW 1 ; 3RD STATUS WORD
913

```


CZDME MACY11 30A(1052) 07-JUL-80 13:53 PAGE 20

CZDME.P11 01-JUL-80 15:29

PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

SEQ 0020

914	001550	000001	DMCR05: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 05
915	001552	000001	DMS105: .BLKW	1	;VECTOR FOR DMC11 NUMBER 05
916	001554	000001	DMS205: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 05
917	001556	000001	DMS305: .BLKW	1	;3RD STATUS WORD
918					
919	001560	000001	DMCR06: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 06
920	001562	000001	DMS106: .BLKW	1	;VECTOR FOR DMC11 NUMBER 06
921	001564	000001	DMS206: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 06
922	001566	000001	DMS306: .BLKW	1	;3RD STATUS WORD
923					
924	001570	000001	DMCR07: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 07
925	001572	000001	DMS107: .BLKW	1	;VECTOR FOR DMC11 NUMBER 07
926	001574	000001	DMS207: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 07
927	001576	000001	DMS307: .BLKW	1	;3RD STATUS WORD
928					
929	001600	000001	DMCR10: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 10
930	001602	000001	DMS110: .BLKW	1	;VECTOR FOR DMC11 NUMBER 10
931	001604	000001	DMS210: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 10
932	001606	000001	DMS310: .BLKW	1	;3RD STATUS WORD
933					
934	001610	000001	DMCR11: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 11
935	001612	000001	DMS111: .BLKW	1	;VECTOR FOR DMC11 NUMBER 11
936	001614	000001	DMS211: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 11
937	001616	000001	DMS311: .BLKW	1	;3RD STATUS WORD
938					
939	001620	000001	DMCR12: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 12
940	001622	000001	DMS112: .BLKW	1	;VECTOR FOR DMC11 NUMBER 12
941	001624	000001	DMS212: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 12
942	001626	000001	DMS312: .BLKW	1	;3RD STATUS WORD
943					
944	001630	000001	DMCR13: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 13
945	001632	000001	DMS113: .BLKW	1	;VECTOR FOR DMC11 NUMBER 13
946	001634	000001	DMS213: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 13
947	001636	000001	DMS313: .BLKW	1	;3RD STATUS WORD
948					
949	001640	000001	DMCR14: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 14
950	001642	000001	DMS114: .BLKW	1	;VECTOR FOR DMC11 NUMBER 14
951	001644	000001	DMS214: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 14
952	001646	000001	DMS314: .BLKW	1	;3RD STATUS WORD
953					
954	001650	000001	DMCR15: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 15
955	001652	000001	DMS115: .BLKW	1	;VECTOR FOR DMC11 NUMBER 15
956	001654	000001	DMS215: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 15
957	001656	000001	DMS315: .BLKW	1	;3RD STATUS WORD
958					
959	001660	000001	DMCR16: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 16
960	001662	000001	DMS116: .BLKW	1	;VECTOR FOR DMC11 NUMBER 16
961	001664	000001	DMS216: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 16
962	001666	000001	DMS316: .BLKW	1	;3RD STATUS WORD
963					
964	001670	000001	DMCR17: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 17
965	001672	000001	DMS117: .BLKW	1	;VECTOR FOR DMC11 NUMBER 17
966	001674	000001	DMS217: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 17
967	001676	000001	DMS317: .BLKW	1	;3RD STATUS WORD
968					
969	001700	000000	DM.END: 000000		

```
970
971 ;DMC11 PASS COUNT AND ERROR COUNT TABLE
972 ;-----
973
974 CNT.MAP:
975 001702 000000 PACT00: 0 ;PASS COUNT FOR DMC11 NUMBER 00
976 001704 000000 ERCT00: 0 ;ERROR COUNT FOR DMC11 NUMBER 00
977
978 001706 000000 PACT01: 0 ;PASS COUNT FOR DMC11 NUMBER 01
979 001710 000000 ERCT01: 0 ;ERROR COUNT FOR DMC11 NUMBER 01
980
981 001712 000000 PACT02: 0 ;PASS COUNT FOR DMC11 NUMBER 02
982 001714 000000 ERCT02: 0 ;ERROR COUNT FOR DMC11 NUMBER 02
983
984 001716 000000 PACT03: 0 ;PASS COUNT FOR DMC11 NUMBER 03
985 001720 000000 ERCT03: 0 ;ERROR COUNT FOR DMC11 NUMBER 03
986
987 001722 000000 PACT04: 0 ;PASS COUNT FOR DMC11 NUMBER 04
988 001724 000000 ERCT04: 0 ;ERROR COUNT FOR DMC11 NUMBER 04
989
990 001726 000000 PACT05: 0 ;PASS COUNT FOR DMC11 NUMBER 05
991 001730 000000 ERCT05: 0 ;ERROR COUNT FOR DMC11 NUMBER 05
992
993 001732 000000 PACT06: 0 ;PASS COUNT FOR DMC11 NUMBER 06
994 001734 000000 ERCT06: 0 ;ERROR COUNT FOR DMC11 NUMBER 06
995
996 001736 000000 PACT07: 0 ;PASS COUNT FOR DMC11 NUMBER 07
997 001740 000000 ERCT07: 0 ;ERROR COUNT FOR DMC11 NUMBER 07
998
999 001742 000000 PACT10: 0 ;PASS COUNT FOR DMC11 NUMBER 10
1000 001744 000000 ERCT10: 0 ;ERROR COUNT FOR DMC11 NUMBER 10
1001
1002 001746 000000 PACT11: 0 ;PASS COUNT FOR DMC11 NUMBER 11
1003 001750 000000 ERCT11: 0 ;ERROR COUNT FOR DMC11 NUMBER 11
1004
1005 001752 000000 PACT12: 0 ;PASS COUNT FOR DMC11 NUMBER 12
1006 001754 000000 ERCT12: 0 ;ERROR COUNT FOR DMC11 NUMBER 12
1007
1008 001756 000000 PACT13: 0 ;PASS COUNT FOR DMC11 NUMBER 13
1009 001760 000000 ERCT13: 0 ;ERROR COUNT FOR DMC11 NUMBER 13
1010
1011 001762 000000 PACT14: 0 ;PASS COUNT FOR DMC11 NUMBER 14
1012 001764 000000 ERCT14: 0 ;ERROR COUNT FOR DMC11 NUMBER 14
1013
1014 001766 000000 PACT15: 0 ;PASS COUNT FOR DMC11 NUMBER 15
1015 001770 000000 ERCT15: 0 ;ERROR COUNT FOR DMC11 NUMBER 15
1016
1017 001772 000000 PACT16: 0 ;PASS COUNT FOR DMC11 NUMBER 16
1018 001774 000000 ERCT16: 0 ;ERROR COUNT FOR DMC11 NUMBER 16
1019
1020 001776 000000 PACT17: 0 ;PASS COUNT FOR DMC11 NUMBER 17
1021 002000 000000 ERCT17: 0 ;ERROR COUNT FOR DMC11 NUMBER 17
1022
```

1023

FORMAT OF STATUS TABLE

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00					
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	CSR				
I	C	O	N	T	R	O	L	R	E	G	I	S	T	E	R					
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I					
I	*	I	*	I	*	I	*	I	*	I	*	I	*	I	*	STAT1				
I	I	I	I	I	I	I	I	I	V	E	C	T	O	R	*					
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I					
I	*	I	B	M	I	A	D	D	*	I	*	I	L	I	N	E	#	*	I	STAT2
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I					
I	I	I	I	I	I	I	I	I	I	I	I	I	I	*	I	*	I	*	I	STAT3
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I					

DEFINITION OF FORMAT

- CSR: CONTAINS DMC11 CSR ADDRESS
- STAT1: BITS 00-08 IS DMC11 VECTOR ADDRESS
 BIT15=1 MICRO-PROCESSOR HAS CROM
 BIT15=0 MICRO-PROCESSOR HAS CROM
 BIT14=1 ??? TURNAROUND CONNECTOR IS ON
 BIT14=0 NO TURNAROUND CONNECTOR
 BIT13=0 LINE UNIT IS AN M8201
 BIT13=1 LINE UNIT IS AN M8202
 BIT12=1 NO LINE UNIT
 BITS 09-11 IS DMC11 BR PRIORITY LEVEL
- STAT2: LOW BYTE IS SWITCH PAC#1 (DDCMP LINE NUMBER)
 HIGH BYTE IS SWITCH PAC#2 (BM873 BOOT ADD)
- STAT3: BIT0=1 DO FREE RUNNING TESTS ON KMC
 (MUST BE SET TO A ONE MANUALLY [PROGRAM DZDMI ONLY])
 KMC MUST HAVE MICRO-CODE WRITTEN FROM RUNNING
 DZDMG TEST 2 FIRST
 BIT1=1 DMC11-AL LOCAL HIGH SPEED MICRO-CODE
 BIT1=0 DMC11-AR REMOTE LOW SPEED MICRO-CODE
 BIT2=0 DMC11-DA (RS232C)
 BIT2=1 DMC11-FA (V.35)

CZDME MACY11 30A(1052) 07-JUL-80 13:53 PAGE 23

CZDME.P11 01-JUL-80 15:29

PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

SEQ 0023

S

```

1081
1082
1083
1084
1085
1086
1087
1088
1089 002002 012737 000340 177776 .START: MOV #340,PS ;LOCK OUT INTERRUPTS
1090 002010 012706 001200 MOV #STACK,SP ;SET UP STACK
1091 002014 012737 005336 000024 MOV #.PFAIL,@#24 ;SET UP POWER FAIL VECTOR
1092 002022 013737 001310 001314 MOV DMNUM,SAVNUM ;SAVE NUMBER OF DEVICES IN SYSTEM.
1093 002030 005037 010144 CLR SWFLG ;CLEAR SOFT TIMEOUT FLAG
1094 002034 105037 001325 CLRB ERRFLG ;CLEAR ERROR FLAG
1095 002040 105037 001327 CLRB QV.FLG ;ZERO QUICK VERIFY FLAG
1096 002044 012737 001470 001320 MOV #DM.MAP-10,CREAM ;GET MAP POINTER.
1097 002052 012737 001676 001322 MOV #CNT.MAP-4,MILK ;GET PASS COUNT MAP POINTER
1098 002060 012737 100000 001316 MOV #BIT15,RUN ;POINT POINTER TO FIRST DEVICE.
1099 002066 012700 001702 MOV #CNT.MAP,RO ;PASS COUNT POINTER TO RO
1100 002072 005020 23$: CLR (RO)+ ;CLEAR TABLE
1101 002074 022700 002002 CMP #CNT.MAP+100,RO ;DONE YET?
1102 002100 001374 BNE 23$ ;KEEP GOING
1103 002102 005037 001234 CLR LSTERR ;CLEAR LAST ERROR POINTER
1104 002106 012737 000001 001226 MOV #1,TSTNO ;SET UP FOR TEST 1
1105 002114 012737 002002 001214 MOV #.START,RETURN ;SET UP FOR POWER FAIL BEFORE
1106 ;TESTING STARTS
1107 002122 013746 000006 MOV @#6,-(SP) ;SAVE CURRENT VECTORS
1108 002126 013746 000004 MOV @#4,-(SP) ;
1109 002132 012737 002166 000004 MOV #6$,@#4 ;SET UP FOR TIMEOUT
1110 002140 012737 177570 001202 MOV #177570,SWR ;SET SWR TO HARD SWR ADDRESS
1111 002146 012737 177570 001200 MOV #177570,DISPLAY ;SET DISPLAY TO HARD SWR ADDRESS
1112 002154 022777 177777 177020 CMP #-1,@SWR ;REFERENCE HARDWARE SWITCH REGISTER
1113 002162 001402 BEQ 6$+2 ;IF = -1 USE SOFT SWR ANYWAY
1114 002164 000407 BR 7$ ;IF IT EXISTS AND NOT = -1 USE HARD SWR
1115 002166 022626 6$: CMP (SP)+,(SP)+ ;ADJUST STACK
1116 002170 012737 000176 001202 MOV #SWREG,SWR ;POINTER TO SOFT SWR
1117 002176 012737 000174 001200 MOV #DISPREG,DISPLAY ;POINTER TO SOFT DISPLAY REG
1118 002204 012637 000004 7$: MOV (SP)+,@#4 ;RESTORE VECTORS
1119 002210 012637 000006 MOV (SP)+,@#6 ;
1120 002214 105737 001324 TSTB INIFLG ;HAS INITIALIZATION BEEN PERFORMED
1121 002220 001006 BNE 20$ ;BR IF YES
1122 002222 022737 003522 000042 CMP #SENDAD,@#42 ;IF ACT-11 AUTOMATIC MODE, DON'T TYPE ID
1123 002230 001402 BEQ 20$ ;
1124 002232 104402 001000 TYPE ,MTITLE ;TYPE TITLE MESSAGE
1125 002236 004737 007734 20$: JSR PC,CKSWR ;CHECK FOR SOFT SWR
1126 002242 017737 176734 001236 MOV @SWR,STRTSW ;STORE STARTING SWITCHES
1127 002250 005737 000042 TST @#42 ;IS IT RUNNING IN AUTO MODE?
1128 002254 001402 BEQ .+6 ;BR IF NO
1129 002256 005037 001236 CLR STRTSW ;IF YES, CLEAR SWITCHES
1130 002262 032737 000001 001236 BIT #SW00,STRTSW ;IF SW00=1, QUESTIONS ARE ASKED.
1131 002270 001012 BNE 17$ ;BR IF SW00=1
1132 002272 105737 001236 TSTB STRTSW ;BIT7=1??
1133 002276 100007 BPL 17$ ;BR IF SW07=0
1134 002300 005737 001306 TST DMACTV ;ARE ANY DEVICES SELECTED?
1135 002304 001006 BNE 16$ ;BR IF YES
1136 002306 104402 007154 TYPE, NOACT ;NO DEVICES SELECTED

```



```

1137 002312 000000          HALT          ;STOP THE SHOW
1138 002314 000776          BR          ;DISQUALIFY CONTINUE SWITCH
1139 002316 004737 010640    17$: JSR      PC,AUTO.SIZE ;GO DO THE AUTO SIZE
1140 002322 105737 001324    16$: TSTB   INIFLG ;FIRST TIME?
1141 002326 001410          BEQ      21$ ;BR IF YES
1142 002330 105737 001236    TSTB   STRTSW ;IF USING SAME PARAMETERS DONT TYPE MAP
1143 002334 100431          BMI      1$
1144 002336 032737 000006 001236 BIT     #BIT1,BIT2,STRTSW;IS TEST NO. OR LOCK SELECTED
1145 002344 001403          BEQ      24$ ;IF NO THEN TYPE STATUS
1146 002346 000424          BR          ;IF YES DO NOT TYPE STATUS
1147 002350 005137 001324    21$: COM    INIFLG ;SET FLAG
1148 002354 104402 006224    24$: TYPE   ,XHEAD ;TYPE HEADER
1149 002360 012704 001500    MOV     #DM.MAP,R4 ;SET POINTER
1150 002364 010437 001246    5$: MOV     R4,TEMP1 ;SET ADDRESS
1151 002370 012437 001250    MOV     (R4)+,TEMP2 ;SET CSR
1152 002374 001411          BEQ      1$ ;ALL DONE IF ZEPO
1153 002376 012437 001252    MOV     (R4)+,TEMP3 ;SET STAT1
1154 002402 012437 001254    MOV     (R4)+,TEMP4 ;SET STAT2
1155 002406 012437 001256    MOV     (R4)+,TEMP5 ;SET STAT3
1156 002412 104410          CONVRT   ;TYPE OUT STATUS MAP
1157 002414 007602          XSTATQ
1158 002416 000762          BR          5$
1159 002420 012700 001500    1$: MOV     #DM.MAP,R0 ;R0 POINTS TO STATUS TABLE

```

```

1160
1161 *****
1162 : *AUTO SIZE TEST
1163 : *THIS TEST VERIFYS THAT THE DMC11S AND/OR KMC11S ARE AT THE CORRECT FLOATING
1164 : *ADDRESSES FOR YOUR SYSTEM. IF THIS TEST FAILS, IT IS NOT A HARDWARE ERROR.
1165 : *CHECK THE ADDRESSES OF ALL FLOATING DEVICES (DJ,DM,DQ,DU,DUP,LK,DMC,DZ,KMC).
1166 : *IF THERE ARE NO OTHER FLOATING DEVICES BEFORE THE DMC11, THE FIRST
1167 : *DMC11 ADDRESS IS 760070, KMC11 IS 760110. NO DEVICE SHOULD EVER BE AT
1168 : *ADDRESS 760000. THIS TEST MAY REQUIRE 2 OR MORE ATTEMPTS TO GET THE
1169 : *RIGHT ADDFESSES. AFTER YOU HAVE CHANGED THE ADDRESS TO WHAT IT TOLD
1170 : *YOU THE FIRST TIME, IT MAY COME BACK AND TELL YOU A DIFFERENT ADDRESS
1171 : *THE NEXT TIME YOU RUN IT. PLEASE HAVE PATIENCE, THE FINAL ADDRESS
1172 : *WILL BE CORRECT (AS LONG AS ALL DEVICES IN FRONT OF THE DMC'S ARE
1173 : *CORRECT).
1174 : *****
1175

```

```

1176 002424 013746 000004          MOV     @#4,-(SP) ;SAVE LOC 4
1177 002430 013746 000006          MOV     @#6,-(SP) ;SAVE LOC 6
1178 002434 005037 000006          CLR     @#6 ;CLEAR VEC+2
1179 002440 005037 001252          CLR     TEMP3 ;CLEAR FLAG
1180 002444 005005          CLR     R5 ;R5=0=DMC, R5=-1=KMC
1181 002446 011337 001404    AUSTRT: MOV     (R0),DMCSR ;GET NEXT DMC CSR
1182 002452 001564          BEQ     AUDONE ;BR IF DONE
1183 002454 005705          TST    R5 ;DMC OR KMC?
1184 002456 001005          BNE    1$ ;BR IF KMC
1185 002460 032760 100000 000002 BIT     #BIT15,2(R0) ;CHECK FOR DMC CSR
1186 002466 001061          BNE    SKIP ;SKIP IF NOT DMC
1187 002470 000404          BR     2$ ;ITS A DMC SO CONTINUE
1188 002472 032760 100000 000002 1$: BIT     #BIT15,2(R0) ;CHECK FOR KMC CSR
1189 002500 001454          BEQ     SKIP ;SKIP IF NOT KMC
1190 002502 012737 002674 000004 2$: MOV     #NODEV,@#4 ;SET JP FOR TIMEOUT
1191 002510 005705          TST    R5 ;DMC OR KMC?
1192 002512 001003          BNE    3$ ;BR IF KMC

```

PROGRAM INITIALIZATION AND START UP.

```

1193 002514 012703 0000C6      MOV      #6,R3      ;R3 IS COUNT OF DEVICES BEFORE DMC
1194 002520 000402      BR       4$        ;GO ON
1195 002522 012703 000010      3$: MOV     #10,R3   ;R3 IS COUNT OF DEVICES BEFORE KMC
1196 002526 012702 003010      4$: MOV     #DEVTAB,R2 ;R2 IS DEVICE TABLE PONTER
1197 002532 012701 160010      MOV     #160010,R1 ;START WITH ADDRESS 160010
1198 002536 005711      FLOAT: TST      (R1) ;CHECK ADDRESS IN R1
1199 002540 111204      MOV     (R2),R4    ;IF NO TIMEOUT, GET NEXT ADDRESS
1200 002542 060401      ADD     R4,R1     ;IN R1
1201 002544 005201      INC     R1
1202 002546 040401      BIC     R4,R1
1203 002550 005703      TST     R3        ;ANY MORE DEVICES TO CHECK FOR?
1204 002552 001371      BNE     FLOAT     ;BR IF YES
1205 002554 012737 002700 000004      MOV     #ERR,@#4   ;OK ONLY DMC'S ARE LEFT, SET UP FOR TIMEOUT
1206 002562 010137 003022      MOV     R1,XLOC   ;SAVE FIRST DMC/KMC ADDRESS
1207 002566 005705      FY:    TST      R5   ;DMC OR KMC?
1208 002570 001005      BNE     1$       ;BR IF KMC
1209 002572 032760 100000 000002      BIT     #BIT15,2(R0) ;CHECK FOR DMC CSR
1210 002600 001014      BNE     SKIP     ;SKIP IF NOT DMC
1211 002602 000404      BR      2$       ;ITS A DMC SO CONTINUE
1212 002604 032760 100000 000002 1$: BIT     #BIT15,2(R0) ;CHECK FOR KMC CSR
1213 002612 001407      BEQ     SKIP     ;SKIP IF NOT KMC
1214 002614 005711      2$:    TST      (R1) ;CHECK DMC ADDRESS
1215 002616 020137 001404      CMP     R1,DMCSR ;DOES Ii MATCH
1216 002622 001411      BEQ     OK       ;BR IF YES
1217 002624 062701 000010      ADD     #10,R1   ;GET NEXT DMC ADDRESS
1218 002630 000756      BR      FY       ;DO IT AGAIN
1219 002632 062700 000010      SKIP: ADD     #10,R0 ;SKIP TO NEXT CSR IN TABLE
1220 002636 011037 001404      MOV     (R0),DMCSR ;GET NEXT CSR
1221 002642 001470      BEQ     AUDONE   ;BR IF DONE
1222 002644 000750      BR      FY       ;ELSE CONTINUE
1223 002646 062700 000010      OK:    ADD     #10,R0 ;SKIP TO NEXT DMC CSR
1224 002652 062737 000010 003022      ADD     #10,XLOC  ;UPDATE EXPECTED DMC/KMC ADDRESS
1225 002660 011037 001404      MOV     (R0),DMCSR ;GET NEX DMC/KMC CSR
1226 002664 001457      BEQ     AUDONE   ;BR IF DGNE
1227 002666 013701 003022      MOV     XLOC,R1  ;GET EXPECTED DMC/KMC ADDRESS
1228 002672 000735      BR      FY       ;CONTINUE
1229 002674 122243      NODEV: CMPB   (R2)+,-(R3) ;ON TIMEOUT, INC R2, DEC R3
1230 002676 000002      RTI
1231 002700 005737 001252      ERR:   TST      TEMP3 ;CHECK FLAG IF = 0 TYPE HEADER
1232 002704 001014      BNE     1$       ;SKIP HEADER
1233 002706 104402      TYPE
1234 002710 007223      CONERR ;TYPEOUT HEADER MESSAGE
1235 002712 012737 002700 001276      MOV     #ERR,SAVPC ;CONFIGURATION ERROR! ..
1236 002720 104411      CNVRT  ;SAVE PC FOR TYPEOUT
1237 002722 002770      ERRPC  ;TYPE OUT ERROR PC
1238 002724 104402      TYPE
1239 002726 007277      CNERR  ;TYPE REST OF HEADER
1240 002730 012737 177777 001252      MOV     #-1,TEMP3 ;SET FLAG SO IT ONLY GETS TYPED ONCE
1241 002736 010137 001262 1$: MOV     R1,SAVR1 ;SAVE R1 FOR TYPEOUT
1242 002742 104410      CONVRT
1243 002744 002776      CONTAB ;TYPE CSR VALUES
1244 002746 005705      TST     R5      ;DMC OR KMC ?
1245 002750 001003      BNE     3$      ;BR IF KMC
1246 002752 104402      TYPE
1247 002754 007320      DMCN
1248 002756 000402      BR      4$      ;CONTINUE

```

PROGRAM INITIALIZATION AND START UP.

```

1249 002760 104402          3$:      TYPE
1250 002762 007330          KCMC
1251 002764 022626          4$:      CMP      (SP)+,(SP)+      ;ADJUST STACK
1252 002766 000727          BR      OK      ;BR TO GET OUT
1253 002770 000001          ERRPC:  1
1254 002772      006      002      .BYTE  6,2
1255 002774 001276          SAVPC
1256 002776 000002          CONTAB: 2
1257 003000      006      004      .BYTE  6,4
1258 003002 003022          XLOC
1259 003004      006      002      .BYTE  6,2
1260 003006 001404          DMCSR
1261 003010      007          DEVTAB: .BYTE  7      ;DJ
1262 003011      017          .BYTE  17     ;DH
1263 003012      007          .BYTE  7      ;DQ
1264 003013      007          .BYTE  7      ;DU
1265 003014      007          .BYTE  7      ;DUP
1266 003015      007          .BYTE  7      ;LK
1267 003016      007          .BYTE  7      ;DMC
1268 003017      007          .BYTE  7      ;DZ
1269 003020      007          .BYTE  7      ;KMC
1270      003022 003022          .EVEN
1271 003022 000000          XLOC:  0
1272 003024 005705          AUDONE: TST      R5      ;DMC?
1273 003026 001005          BNE     1$      ;BR IF KMC AND ALL DONE
1274 003030 012705 177777          MOV     #-1,R5      ;SET R5 TO -1 (KMC)
1275 003034 012700 001500          MOV     #DM.MAP,RO  ;RESET RO TO START OF TABLE
1276 003040 000602          BR      AUSTRT    ;GO DO KMC'S
1277 003042 012637 000006          1$:    MOV     (SP)+,@#6    ;RESTORE LOC 6
1278 003046 012637 000004          MOV     (SP)+,@#4    ;RESTORE LOC 4
1279 003052 032737 000010 001236          BIT     #SW03,STRTSW ;SELECT SPECIFIC DEVICES??
1280 003060 001422          BEQ     3$      ;BR IF NO.
1281 003062 104402 006144          TYPE   ,MNEW     ;TYPE THE MESSAGE.
1282 003066 005000          CLR     RO      ;ZERO DATA LIGHTS
1283 003070 000000          HALT                    ;WAIT FOR USER TO TELL WHAT DEVICES TO RUN
1284 003072 027737 176104 001312          CMP     @SWR,SAVACT  ;IS THE NUMBER VALID?
1285 003100 101404          BLOS   2$      ;BR IF NUMBER IS OK.
1286 003102 104402 006005          TYPE   ,MERR3    ;TELL USER OF INVALID NUMBER.
1287 003106 000000          HALT                    ;STOP EVERY THING.
1288 003110 000776          BR      -2      ;RESTART THE PROGRAM AGAIN.
1289 003112 017737 176064 001306          2$:    MOV     @SWR,DMACTV ;GET NEW DEVICE PATTERN
1290 003120 013700 001306          MOV     DMACTV,RO   ;SHOW THE USER WHAT HE SELECTED.
1291 003124 000000          HALT                    ;CONTINUE DYNAMIC SWITCHES.
1292 003126 012700 000300          3$:    MOV     #300,RO  ;PREPARE TO CLEAR THE FLOATING
1293 003132 012701 000302          MOV     #302,R1    ;VECTOR AREA. 300-776
1294 003136 010120          4$:    MOV     R1,(RO)+ ;START PUTTING 'PC+2 - HALT'
1295 003140 005021          CLR     (R1)+      ;IN VECTOR AREA.
1296 003142 022021          CMP     (RO)+,(R1)+ ;POP POINTERS
1297 003144 022700 001000          CMP     #1000,RO   ;ALL DONE??
1298 003150 001372          BNE     4$      ;BR IF NO.
1299
1300          ;TEST START AND RESTART
1301          ;-----
1302
1303 003152 012706 001200          .BEGIN: MOV     #STACK,SP ;SET UP STACK
1304 003156 013746 000006          MOV     @#6,-(SP) ;SAVE LOC 6

```

1305	003162	013746	000004			MOV	@#4,-(SP)	;SAVE LOC 4
1306	003166	005000				CLR	RO	;START AT 0
1307	003170	012737	003234	000004		MOV	#2\$,@#4	;SET UP FOR TIME OUT
1308	003176	005037	000006			CLR	@#6	;TO AUTOSIZE MEMORY
1309	003202	005720			6\$:	TST	(RO)+	;CHECK ADDRESS IN RO
1310	003204	022700	157776			CMP	#157776,RO	;IS IT AT LEAST 28K
1311	003210	001374				BNE	6\$;BR IF NO
1312	003212	162700	007776			SUB	#7776,RO	;SAVE 2K FOR MONITORS
1313	003216	010037	001304		7\$:	MOV	RO,MEMLIM	;STORE MEMORY LIMIT
1314	003222	012637	000004			MOV	(SP)+,@#4	;RESTORE LOC 4
1315	003226	012637	000006			MOV	(SP)+,@#6	;RESTORE LOC 6
1316	003232	000413				BR	10\$;CONTINUE
1317	003234	022626			2\$:	CMP	(SP)+,(SP)+	;ADJUST STACK
1318	003236	162700	000004			SUB	#4,RO	;GET LAST GOOD ADDRESS
1319	003242	162700	007776			SUB	#7776,RO	;SAVE 2K FOR MONITORS
1320	003246	022700	030000			CMP	#30000,RO	;IS IT 8K?
1321	003252	001361				BNE	7\$;BR IF NO
1322	003254	012700	037400			MOV	#37400,RO	;IF 8K DON'T SAVE 2K
1323	003260	000756				BR	7\$	
1324	003262	012737	000340	177776	10\$:	MOV	#340,PS	;LOCK OUT INTERRUPTS
1325	003270	032737	000004	001236		BIT	#BIT2,STRTSW	;CHECK FOR LOCK ON TEST
1326	003276	001411				BEQ	1\$;BR IF NO LOCK DESIRED.
1327	003300	104402	006043			TYPE	,MLOCK	;TYPE LOCK SELECTED.
1328	003304	012737	000240	003612		MOV	#NOP,TTST	;ADJUST SCOPE ROUTINE.
1329	003312	012737	000240	003614		MOV	#NOP,TTST+2	;SET UP TO LOCK
1330	003320	000406				BR	3\$;CONTINUE ALONG.
1331	003322	013737	003730	003612	1\$:	MOV	BRW,TTST	;PREPARE NORMAL SCOPE ROUTINE
1332	003330	013737	003732	003614		MOV	BRX,TTST+2	;LOCK NOT SELECTED, SET UP FOR NORMAL SCOPE LOOP
1333	003336	012737	010206	001214	3\$:	MOV	#CYCLE,RETURN	;START AT "CYCLE" FIND WHICH DEVICE TO TEST
1334	003344	032737	000002	001236	4\$:	BIT	#SW01,STRTSW	;IS TEST NO. SELECTED?
1335	003352	001002				BNE	5\$;BR IF YES
1336	003354	104402	005755			TYPE	,MR	;TYPE R
1337	003360	000177	175630		5\$:	JMP	@RETURN	;START TESTING

```

1338                                     :END OF PASS
1339                                     :TYPE NAME OF TEST
1340                                     :UPDATE PASS COUNT
1341                                     :CHECK FOR EXIT TO ACT-11
1342                                     :RESTART TEST
1343
1344 003364 000005                                     .EOP: RESET                                     ;MAKE THE WORLD CLEAN AGAIN.
1345 003366 005037 001234                             CLR          LSTERR                                     ;CLEAR LAST ERROR PC
1346 003372 105037 001325                             CLR          ERRFLG                                     ;CLEAR ERROR FLAG
1347 003376 005237 001230                             INC          PASCNT                                     ;UPDATE PASS COUNT
1348 003402 013777 001230 175570                     MOV          PASCNT,@DISPLAY ;DISPLAY PASS COUNT
1349 003410 104402 005733                             TYPE        ,MEPASS                                     ;TYPE END PASS
1350 003414 104402 006072                             TYPE        ,MCSRX                                      ;TYPE CSR
1351 003420 104411 003546                             CNVRT       ,XCSR                                      ;SHOW IT
1352 003424 104402 006100                             TYPE        ,MVECX                                      ;TYPE VECTOR
1353 003430 104411 003554                             CNVRT       ,XVEC                                      ;SHOW IT
1354 003434 104402 006106                             TYPE        ,MPASSX                                    ;TYPE PASSES
1355 003440 104411 003562                             CNVRT       ,XPASS                                    ;SHOW IT
1356 003444 104402 006117                             TYPE        ,MERRX                                    ;TYPE ERRORS
1357 003450 104411 003570                             CNVRT       ,XERR                                      ;SHOW IT
1358 003454 013700 001322                             MOV          MILK,RO                                     ;GET POINTER TO PASS COUNT
1359 003460 013720 001230                             MOV          PASCNT,(RO)+ ;STORE PASS COUNT FOR THIS DMC11
1360 003464 013720 001232                             MOV          ERRCNT,(RO)+ ;STORE ERROR COUNT FOR THIS DMC11
1361 003470 005337 001314                             DEC          SAVNUM                                     ;ARE ALL DEVICES TESTED?
1362 003474 001017                                     BNE          RESTRT                                    ;BR IF NO.
1363 003476 112737 000377 001327                     MOV          #377,QV.FLG ;SET THE QUICK VERIFY FLAG.
1364 003504 013737 001310 001314                     MOV          DMNUM,SAVNUM ;RESTORE THE COUNT
1365 003512 013701 000042                             MOV          @#42,R1  ;CHECK FOR ACT-11 OR DDP
1366 003516 001406                                     BEQ          RESTRT                                    ;IF NOT, CONTINUE TESTING
1367 003520 000005                                     RESET                                                ;STOP THE SHOW--CLEAR THE WORLD
1368 003522
1369 003522 004711                                     $ENDAD: JSR          PC,(R1)
1370 003524 000240                                     NOP
1371 003526 000240                                     NOP
1372 003530 000240                                     NOP
1373 003532 000240                                     NOP
1374 003534 012737 010206 001214 RESTRT: MOV          #CYCLE,RETURN
1375 003542 000137 010206                             JMP          CYCLE
1376 003546 000001                                     XCSR:      1
1377 003550 006 002                                     .BYTE      6,2
1378 003552 001404                                     DMCSR
1379 003554 000001                                     XVEC:      1
1380 003556 004 002                                     .BYTE      4,2
1381 003560 001374                                     DMRVEC
1382 003562 000001                                     XPASS:     1
1383 003564 006 002                                     .BYTE      6,2
1384 003566 001230                                     PASCNT
1385 003570 000001                                     XERR:      1
1386 003572 006 002                                     .BYTE      6,2
1387 003574 001232                                     ERRCNT
1388
1389                                     ;SCOPE LOOP AND INTERATION HANDLER
1390                                     ;-----
1391
1392 003576 004737 007734                                     .SCOPE: JSR          PC,CKSWR ;CHECK FOR SOFT SWR
1393 003602 010016                                     MOV          RO,(SP)  ;SAVE RO ON THE STACK

```



```

1394 003604 032777 040000 175370          BIT      #BIT14,@SWR      ;'LOOP ON THIS TEST'?
1395 003612 001407          TTST:    BEQ      1$          ;BR IF NO. (IF LOCK SW01=1; THIS LOC =240)
1396 003614 000437          BR      3$          ;GOTO 3$ (IF LOCK SW01=1; THIS LOC =240)
1397 003616 005737 003734          TST     DONE         ;WAS TKCSR DONE SET?
1398 003622 001434          BEQ     3$          ;BR IF NO (LOCKED ON TEST)
1399 003624 005037 003734          CLR     DONE         ;YES, CLEAR FLAG
1400 003630 000415          BR     2$          ;GO TO NEXT TEST
1401 003632 032777 004000 175342 1$:    BIT     #SW11,@SWR     ;DELETE ITERATION? (QUICK PASS)
1402 003640 001011          BNE     2$          ;BR IF YES
1403 003642 105737 001327          TSTB   QV.FLG        ;HAVE PASSES BEECOMPLETED?
1404 003646 001406          BEQ     2$          ;BR IF QUICK PASS.
1405 003650 005237 001224          INC     LPCNT         ;UPDATE ITERATION COUNTER
1406 003654 023737 001224 001222          CMP     LPCNT,ICOUNT ;ARE ALL ITERATIONS DONE??
1407 003662 101414          BLOS   3$          ;BR IF NOT YET
1408 003664 105037 001325          CLRB   ERRFLG        ;PREPARE FOR NEW TEST
1409 003670 005037 001224          CLR     LPCNT         ;START ICOUNTER AT 0
1410 003674 005037 001220          CLR     LOCK          ;
1411 003700 012737 000020 001222          MOV     #20,ICOUNT    ;RESET ITERATIONS
1412 003706 013737 001216 001214          MOV     NEXT,RETURN   ;GET NEXT TEST
1413 003714 011600          3$:    MOV     (SP),R0      ;POP R0 OFF OF THE STACK
1414 003716 022626          POP2SP                ;FAKE AN 'RTI'
1415 003720 013701 001404          MOV     DMCSR,R1      ;R1 CONTAINS BASE DML ADDRESS
1416 003724 000177 175264          JMP     @RETURN        ;GO DO THE TEST
1417 003730 001407          BRW:   1407           ;
1418 003732 000437          BRX:   437            ;
1419 003734 000000          DONE:  0              ;

```

```

1420
1421          ;CHECK FOR FREEZE ON CURRENT DATA
1422          ;-----

```

```

1423
1424 003736 004737 007734          .SCOPI: JSR     PC,CKSWR    ;CHECK FOR SOFT SWR
1425 003742 032777 001000 175232          BIT     #SW09,@SWR    ;IS SW09=1(SFT)?
1426 003750 001405          BEQ     1$          ;BR IF NOT SET.
1427 003752 005737 001220          TST     LOCK          ;
1428 003756 001402          BEQ     1$          ;
1429 003760 013716 001220          MOV     LOCK,(SP)     ;GOTO THE ADDRESS IN LOCK.
1430 003764 000002          1$:    RTI             ;GO BACK.

```

```

1431
1432          ;TELETYPE OUTPUT ROUTINE
1433          ;-----

```

```

1434
1435 003766 010546          .TYPE:  MOV     R5,-(SP)    ;SAVE R5 ON THE STACK.
1436 003770 017605          MOV     @2(SP),R5     ;GET ADDRESS OF MESSAGE.
1437 003774 062766 000002 000002          ADD     #2,2(SP)      ;POP OVER ADDRESS.
1438 004002 005737 010144          4$:    TST     SWFLG      ;SOFT SWR MESSAGE?
1439 004006 001004          BNE     1$          ;IF YES TYPE IT OUT REGARDLESS OF SW12
1440 004010 032777 010000 175164          BIT     #SW12,@SWR    ;INHIBIT ALL PRINT OUT??
1441 004016 001012          BNE     3$          ;BR IF NO PRINT OUT WANTED (SW12=1)
1442 004020 105715          1$:    TSTB   (R5)        ;IS NUMBER MINUS? (MSB=1(BIT7))
1443 004022 100002          BPL     2$          ;BR IF NUMBER IS PLUS
1444 004024 104402 005672          TYPE   ,MCRLF         ;TYPE A CR/LF!
1445 004030 105777 175154          2$:    TSTB   @TPCSR     ;TTY READY?
1446 004034 100375          BPL     2$          ;BR IF NO.
1447 004036 112577 175150          MOVB   (R5)+,@TPDBR  ;PRINT CURRENT CHAR.
1448 004042 001357          BNE     4$          ;IF NOT ZERO KEEP PRINTING.
1449 004044 012605          3$:    MOV     (SP)+,R5   ;END OF OUTPUT. RESTORE R5

```

```

1450 004046 000002          RTI          ;GO HOME
1451
1452
1453 004050 010346          .INSTR: MOV      R3,-(SP)          ;SAVE R3 ON STACK
1454 004052 010446          MOV      R4,-(SP)          ;SAVE R4 ON STACK
1455 004054 017637 000004 004072  MOV      @4(SP),.MSG
1456 004062 062766 000002 000004  ADD      #2,4(SP)
1457 004070 104402          .INST1: TYPE
1458 004072 000000          .MSG: 0
1459 004074 012704 007630  MOV      #INBUF,R4
1460 004100 012703 000007  MOV      #7,R3
1461 004104 105777 175074  1$:  TSTB   @TKCSR
1462 004110 100375          BPL     1$
1463 004112 117714 175070  MOVB   @TKDBR,(R4)
1464 004116 142714 000290  BICB   #200,(R4)
1465 004122 122427 000015  CMPB   (R4)+,#15
1466 004126 001417          BEQ    INSTR2
1467 004130 105777 175054  2$:  TSTB   @TPCSR
1468 004134 100375          BPL     2$
1469 004136 017777 175044 175046  MCV    @TKDBR,@TPDBR
1470 004144 005303          DEC    R3
1471 004146 001356          BNE    1$
1472 004150 012604          MOV    (SP)+,R4
1473 004152 012603          MOV    (SP)+,R3
1474 004154 104402 005666  .INSTE: TYPE ,MQM
1475 004160 010346          MOV    R3,-(SP)
1476 004162 010446          MOV    R4,-(SP)
1477 004164 000741          BR     .INST1
1478 004166 012604  INSTR2: MOV    (SP)+,R4          ;RESTORE R4
1479 004170 012603          MOV    (SP)+,R3          ;RESTORE R3
1480 004172 000002          RTI
1481
1482          ;CONVERT ASCII STRING TO OCTAL
1483
1484
1485 004174 010546          .PARAM: MOV    R5,-(SP)
1486 004176 010446          MOV    R4,-(SP)
1487 004200 016605 000004  MOV    4(SP),R5
1488 004204 012537 004364  MOV    (R5)+,LOLIM
1489 004210 012537 004366  MOV    (R5)+,HILIM
1490 004214 012537 004370  MOV    (R5)+,DEVADR
1491 004220 112537 004372  MOVB   (R5)+,LOBITS
1492 004224 112537 004373  MOVB   (R5)+,ADRCNT
1493 004230 010566 000004  MOV    R5,4(SP)
1494 004234 005005  PARAM*: CLR    R5
1495 004236 012704 007630  MOV    #INBUF,R4
1496 004242 122714 000015  CMPB   #15,(R4)
1497 004246 001420          BEQ    PARERR
1498 004250 121427 000060  1$:  CMPB   (R4),#60
1499 004254 002415          BLT    PARERR
1500 004256 121427 000067  CMPB   (R4),#67
1501 004262 003012          BGT    PARERR
1502 004264 142714 000060  BICB   #60,(R4)
1503 004270 152405          BICB   (R4)+,R5
1504 004272 122714 000015  CMPB   #15,(R4)
1505 004276 001406          BEQ    LIMITS

```

CZDME MACY11 30A(1052) 07-JUL-80 13:53 PAGE 32

CZDME.P11 01-JUL-80 15:29

GENERAL UTILITIES (TYPEOUT, ERROR, SCOPE, ETC)

SEQ 0032

```

1506 004300 006305          ASL    R5
1507 004302 006305          ASL    R5
1508 004304 006305          ASL    R5
1509 004306 000760          BR     1$
1510 004310 104404          PARERR: INSTER
1511 004312 000750          BR     PARAM1
1512
1513                          ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
1514                          ;-----
1515
1516 004314 020537 004366    LIMITS: CMP    R5,HILIM
1517 004320 101373          BHI    PARERR
1518 004322 020537 004364    CMP    R5,LOLIM
1519 004326 103770          BLO    PARERR
1520 004330 133705 004372    BITB   LOBITS,R5
1521 004334 001365          BNE    PARERR
1522
1523                          ;STORE NUMBER AT SPECIFIED ADDRESS
1524
1525 004336 013704 004370    1$:   MOV    DEVADR,R4
1526 004342 010524          MOV    R5,(R4)+
1527 004344 062705 000002    ADD    #2,R5
1528 004350 105337 004373    DECB   ADRCNT
1529 004354 001372          BNE    1$
1530 004356 012604          MOV    (SP)+,R4
1531 004360 012605          MOV    (SP)+,R5
1532 004362 000002          RTI
1533 004364 000000    LOLIM: 0
1534 004366 000000    HILIM: 0
1535 004370 000000    DEVADR: 0
1536 004372 000000    LOBITS: 0
1537          ADRCNT=LOBITS+1
1538
1539                          ;SAVE PC OF TEST THAT FAILED AND R0-R5
1540                          ;-----
1541
1542 004374 016637 000004 001276 .SAV05: MOV    4(SP),SAVPC    ;SAVE R7 (PC)
1543
1544                          ;SAVE R0-R5
1545
1546 004402 010537 001272    SV05: MOV    R5,SAVR5    ;SAVE R5
1547 004406 010437 001270    MOV    R4,SAVR4    ;SAVE R4
1548 004412 010337 001266    MOV    R3,SAVR3    ;SAVE R3
1549 004416 010237 001264    MOV    R2,SAVR2    ;SAVE R2
1550 004422 010137 001262    MOV    R1,SAVR1    ;SAVE R1
1551 004426 010037 001260    MOV    R0,SAVR0    ;SAVE R0
1552 004432 000002          RTI    ;LEAVE.
1553
1554                          ;RESTORE R0-R5
1555
1556 004434 013700 001260    .RES05: MOV    SAVR0,R0    ;RESTORE R0
1557 004440 013701 001262    MOV    SAVR1,R1    ;RESTORE R1
1558 004444 013702 001264    MOV    SAVR2,R2    ;RESTORE R2
1559 004450 013703 001266    MOV    SAVR3,R3    ;RESTORE R3
1560 004454 013704 001270    MOV    SAVR4,R4    ;RESTORE R4
1561 004460 013705 001272    MOV    SAVR5,R5    ;RESTORE R5

```

```

1562 004464 000002          RTI          ;LEAVE
1563
1564          ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
1565          ;-----
1566
1567 004466 104402 005672    .CONVR: TYPE    ,MCRLF
1568 004472 010046          .CNVRT: MOV     R0,-(SP)
1569 004474 010146          MOV     R1,-(SP)
1570 004476 010346          MOV     R3,-(SP)
1571 004500 010446          MOV     R4,-(SP)
1572 004502 010546          MOV     R5,-(SP)
1573 004504 017601 000012    MOV     @12(SP),R1
1574 004510 062766 000002 000012    ADD     #2,12(SP)
1575 004516 012137 004710          MOV     (R1)+,WRDCNT
1576 004522 112137 004712    1$:    MOVB   (R1)+,CHRCNT
1577 004526 112137 004713          MOVB   (R1)+,SPACNT
1578 004532 013137 004714          MOV     @ (R1)+,BINWRD
1579 004536 122737 000003 004712    CMPB   #3,CHRCNT
1580 004544 001003          BNE     2$
1581 004546 042737 177400 004714    BIC     #177400,BINWRD
1582 004554 013704 004714    2$:    MOV     BINWRD,R4
1583 004560 113705 004712          MOVB   CHRCNT,R5
1584 004564 012700 001416          MOV     #TEMP,R0
1585 004570 010403          3$:    MOV     R4,R3
1586 004572 042703 177770          BIC     #177770,k3
1587 004576 062703 000060          ADD     #060,R3
1588 004602 110320          MOVB   R3,(R0)+
1589 004604 000241          CLC
1590 004606 006004          ROR     R4
1591 004610 000241          CLC
1592 004612 006004          ROR     R4
1593 004614 000241          CLC
1594 004616 006004          ROR     R4
1595 004620 005305          DEC     R5
1596 004622 001362          BNE     3$
1597 004624 012703 007672    MOV     #MDATA,R3
1598 004630 114023          4$:    MOVB   -(R0),(R3)+
1599 004632 105337 004712          DECB   CHRCNT
1600 004636 001374          BNE     4$
1601 004640 105737 004713          TSTB   SPACNT
1602 004644 001405          BEQ     6$
1603 004646 112723 000040          5$:    MOVB   #040,(R3)+
1604 004652 105337 004713          DECB   SPACNT
1605 004656 001373          BNE     5$
1606 004660 105013          6$:    CLRB   (R3)
1607 004662 104402 007672    TYPE   ,MDATA
1608 004666 005337 004710          DEC     WRDCNT
1609 004672 001313          BNE     1$
1610 004674 012605          MOV     (SP)+,R5
1611 004676 012604          MOV     (SP)+,R4
1612 004700 012603          MOV     (SP)+,R3
1613 004702 012601          MOV     (SP)+,R1
1614 004704 012600          MOV     (SP)+,R0
1615 004706 000002          RTI
1616 004710 000000          WRDCNT: 0
1617 004712 000000          CHRCNT: 0

```

```

1618          004713          SPACNT=CHRCNT+1
1619 004714 000000          BINWRD: 0
1620
1621
1622          ;TRAP DISPATCH SERVICE
1623          ;ARGUMENT OF TRAP IS EXTRACTED
1624          ;AND USED AS OFFSET TO OBTAIN POINTER
1625          ;TO SELECTED SUBROUTINE
1626
1627 004716 011646          .TRPSR: MOV      (SP),-(SP)          ;GET PC OF RETURN
1628 004720 162716 000002          SUB      #2,(SP)          ;=PC OF TRAP
1629 004724 017616 000000          MOV      @(SP),(SP)          ;GET TRP
1630 004730 006316          TRPOK: ASL      (SP)          ;MULTIPLY TRAP ARG BY 2
1631 004732 042716 177001          BIC      #177001,(SP)          ;CLEAR UNWANTED BITS
1632 004736 062716 001330          ADD      #.TRPTAB,(SP)          ;POINTER TO SUBROUTINE ADDRESS
1633 004742 017616 000000          MOV      @(SP),(SP)          ;SUBROUTINE ADDRESS
1634 004746 000136          JMP      @(SP)+          ;GO TO SUBROUTINE
1635
1636          ;ERROR HANDLER
1637          ;-----
1638
1639 004750 004737 007734          .HLT: JSR      PC,CKSWR          ;CHECK FOR SOFT SWR
1640 004754 032777 010000 174220          BIT      #SW12,@SWR          ;BELL ON ERROR?
1641 004762 001406          BEQ      XBX          ;BR IF NO BELL
1642 004764 105777 174220          TSTB     @TPCSR          ;TTY READY.
1643 004770 100003          BPL      XBX          ;DON'T WAIT IF TTY NOT READY.
1644 004772 112777 000207 174212          MOVB     #207,@TPDBR          ;PUSH A BELL AT THE TTY.
1645 005000 032777 020000 174174          XBX: BIT      #SW13,@SWR          ;DELETE ERROR PRINT OUT?
1646 005006 001105          BNE      HALTS          ;BR IF NO PRINT OUT WANTED.
1647 005010 021637 001234          CMP      (SP),LSTERR          ;WAS THIS ERROR FOUND LAST TIME?
1648 005014 001404          BEQ      1$          ;BR IF YES
1649 005016 011637 001234          MOV      (SP),LSTERR          ;RECORD BEING HERE
1650 005022 105037 001325          CLRB     ERRFLG          ;PREPARE HEADER
1651 005026 104406          1$: SAVO5          ;SAVE ALL PROC REGISTERS
1652 005030 011605          MOV      (SP),R5          ;GET THE PC OF ERROR
1653 005032 162705 000002          SUB      #2,R5          ;GET ADDRESS OF TRAP CALL
1654 005036 011504          MOV      (R5),R4          ;GET HLT INSTRUCTION
1655 005040 006304          ASL      R4          ;MULT BY TWO
1656 005042 061504          ADD      (R5),R4          ;DOUBLE IT
1657 005044 006304          ASL      R4          ;MULT AGAIN
1658 005046 042704 177001          BIC      #177001,R4          ;CLEAR JUNK
1659 005052 062704 032406          ADD      #.ERRTAB,R4          ;GET POINTER
1660 005056 012437 005172          MOV      (R4)+,ERRMSG          ;GET ERROR MESSAGE
1661 005062 012437 005204          MOV      (R4)+,DATAHD          ;GET DATA HEADRER
1662 005066 011437 005216          MOV      (R4),DATABP          ;GET DATA TABLE
1663 005072 105737 001325          TSTB     ERRFLG          ;TYPE HEADREER
1664 005076 001403          BEQ      TYPMSG          ;BR IF YES
1665 005100 005737 005216          TST      DATABP          ;DOES DATA TABLE EXIST?
1666 005104 001040          BNE      TYPDAT          ;BR IF YES.
1667 005106 104402 005672          TYPMSG: TYPE     ,MCRLF
1668 005112 104402 005672          TYPE     ,MCRLF
1669 005116 005737 001220          TST      LOCK
1670 005122 001402          BEQ      1$
1671 005124 104402 006142          TYPE     ,MASTEK
1672 005130 104402 006130          1$: TYPE     ,MTSTN
1673 005134 104411 005330          CNVRT    ,XTSTN          .SHOW IT

```



```

1674 005140 104402 006217          TYPE      ,MERRPC      ;TYPE PC.
1675 005144 104411 005322          CNVRT     ,ERTABO      ;SHOW IT
1676 005150 104402 005672          TYPE      ,MCRLF       ;GIVE A CR/LF
1677 005154 112737 177777 001325    MOV      #-1,ERRFLG ;NO MORE HEADER UNLESS NO DATA TABLE.
1678 005162 005737 005172          TST      ERRMSG     ;IS THERE AN ERROR MESSAGE?
1679 005166 001402          BEQ      WRKO.FM    ;BR IF NO.
1680 005170 104402          TYPE      ;TYPE
1681 005172 000000          ERRMSG: 0          ;ERROR MESSAGE
1682 005174          WRKO.FM:          ;
1683 005174 005737 005204          TST      DATAHD   ;DATA HEADER?
1684 005200 001402          BEQ      TYPDAT    ;BR IF NO
1685 005202 104402          TYPE      ;TYPE
1686 005204 000000          DATAHD: 0        ;DATA HEADER
1687 005206 005737 005216          TYPDAT: TST      DATABP ;DATA TABLE?
1688 005212 001402          BEQ      RESREG   ;BR IF NO.
1689 005214 104410          CONVRT          ;SHOW
1690 005216 000000          DATABP: 0        ;DATA TABLE
1691 005220 104407          RESREG: RESOS    ;RESTORE PROC REGISTERS
1692 005222 022737 003522 000042    HALTS:  CMP      #SENDAD,@#42 ;IF ACT-11 AUTOMATIC MODE, HALT..
1693 005230 001403          BEQ      1$
1694 005232 005777 173744          TST      @SWR      ;HALT ON ERROR?
1695 005236 100005          BPL      EXITER   ;BR IF NO HALT ON ERROR
1696 005240 010046          1$:  PUSHRO     ;SAVE RO
1697 005242 016600 000002          MOV      2(SP),RO ;SHOW ERROR PC IN DATA LIGHTS
1698 005246 000000          HALT
1699 005250 012600          POPRO
1700 005252 005237 001232          EXITER: INC      ERRCNT ;UPDATE ERROR COUNT
1701 005256 032777 000400 173716    BIT      #SW08,@SWR ;GOTO TOP OF TEST?
1702 005264 001007          BNE      1$
1703 005266 032777 002000 173706    BIT      #SW10,@SWR ;GOTO NEXT TEST?
1704 005274 001411          BEQ      2$
1705 005276 013737 001216 001214    MOV      NEXT,RETURN ;SET FOR NEXT TEST
1706 005304 012706 001200          1$:  MOV      #STACK,SP ;RESET SP
1707 005310 013701 001404          MOV      DMCSR,R1 ;SET UP R1
1708 005314 000177 173674          JMP      @RETURN   ;GOTO SPECIFIED TEST
1709 005320 000002          2$:  RTI
1710 005322 000001          ERTABO: 1          ;RETURN
1711 005324          006          002          .BYTE      6,2
1712 005326 001276          SAVPC
1713 005330 000001          XTSTN: 1
1714 005332          003          002          .BYTE      3,2
1715 005334 001226          TSTNO
1716          ;ENTER HERE ON POWER FAILURE
1717          ;-----
1718
1719
1720 005336          .PFAIL:
1721 005336 012737 005350 000024    MOV      #RESTART,24 ;SET UP FOR POWER UP TRAP
1722 005344 000000          HALT      ;HALT ON POWER DOWN NORMAL
1723 005346 000777          BR
1724
1725          ;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
1726
1727 005350          RESTAR:
1728 005350 012737 005336 000024    MOV      #.PFAIL,24 ;SET UP FOR POWER FAILURE
1729 005356 012706 001200          MOV      #STACK,SP ;RESET THE STACK POINTER

```

```

1730 005362 013701 001404      MOV      DMCSR,R1      ;RESTORE R1
1731 005366 005037 001416      CLR      TEMP          ;READY FOR TIMER
1732 005372 005237 001416      INC      TEMP          ;PLUS ONE TO THE TIMER!
1733 005376 001375                BNE      .-4           ;BR IF MORE TO GO
1734 005400 104402 005675      TYPE    ,MPFAIL       ;TYPE THE MESSAGE
1735 005404 104411 005430      CNVRT   ,PFTAB        ;TELL WHAT TEST TO RETURN TO.
1736 005410 105037 001325      CLR     ERRFLG        ;START CLEAN
1737 005414 005037 001234      CLR     LSTERR       ;*****
1738 005420 005011                CLR     (R1)          ;CLEAR MAINT BITS
1739 005422 104412                MSTCLR ;START CLEAN UP OF DEVICE
1740 005424 000177 173564      JMP     @RETURN       ;START DOING THAT TEST AGAIN.
1741 005430 000001                F%TAB: 1
1742 005432 003      002      .BYTE  3,2
1743 005434 001226                TSTNO
1744
1745 005436                .DELAY:
1746 005436 012777 000020 173746      MOV     #20,@DMP04
1747 005444 104414                ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1748 005446 121111                121111 ;POKE CLOCK DELAY BIT
1749 005450                1$:
1750 005450 104414                ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1751 005452 121224                121224 ;PORT4_IBUS*11
1752 005454 032777 000020 173730      BIT     #BIT4,@DMP04 ;IS CLOCK BIT SET?
1753 005462 001772                BEQ     1$           ;BR IF NO
1754 005464 000002                RTI
1755
1756 005466                .MSTCLR:
1757 005466 152777 000100 173712      BISB   #BIT6,@DMCSRH ;SET MASTER CLEAR
1758 005474 142777 000300 173704      BICB   #BIT6 BIT7,@DMCSRH ;CLEAR MASTER CLEAR AND RUN
1759 005502 000002                RTI           ;RETURN
1760
1761 005504                .ROMCLK:
1762 005504 152777 000002 173674      BISB   #BIT1,@DMCSRH ;SET ROMI
1763 005512 013677 173676      MOV     @(SP)+,@DMP06 ;LOAD INSTRUCTION IN SEL6
1764 005516 062746 000002                ADD     #2,-(SP)      ;ADJUST STACK
1765 005522 032777 000100 173452      BIT     #SW06,@SWR    ;HALT IF SW06 =1
1766 005530 001401                BEQ     1$           ;BR IF SW06 =0
1767 005532 000000                HALT    ;HALT BEFORE CLOCKING INSTRUCTION
1768 005534 152777 000003 173644      1$:  BISB   #BIT1!BIT0,@DMCSRH ;CLOCK INSTRUCTION
1769 005542 142777 000007 173636      BICB   #BIT2!BIT1!BIT0,@DMCSRH ;CLEAR ROMO, ROMI, STEP
1770 005550 000002                RTI
1771
1772 005552                .DATACLK:
1773 005552 013637 001416      MOV     @(SP)+,TEMP    ;PUT TICK COUNT IN TEMP
1774 005556 062746 000002                ADD     #2,-(SP)      ;ADJUST STACK
1775 005562 152777 000020 173616      1$:  BISB   #BIT4,@DMCSRH ;SET STEP LU
1776 005570 027777 173610 173606      CMP     @DMCSR,@DMCSR ;WASTE TIME
1777 005576 142777 000020 173602      BICB   #BIT4,@DMCSRH ;CLEAR STEP LU
1778 005604 005337 001416      DEC     TEMP          ;DEC TICK COUNT
1779 005610 001364                BNE     1$           ;BR IF NOT DONE
1780 005612 000002                RTI           ;RETURN
1781 005614 000001                3$:  .BLKW 1
1782
1783 005616                .TIMER:
1784 005616 013637 001416      MOV     @(SP)+,TEMP    ;MOVE COUNT TO TEMP
1785 005622 062746 000002                ADD     #2,-(SP)      ;ADJUST STACK

```

```

1786 005626          1$:
1787 005626 104414      ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1788 005630 021364      021364          ;PORT4 IBUS* REG11
1789 005632 032777 000002 173552      BIT #2,@DMP04    ;IS PGM CLOCK BIT CLEAR?
1790 005640 001772      REQ 1$          ;BR IF YES
1791 005642          2$:
1792 005642 104414      ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1793 005644 021364      021364          ;PORT4 IBUS* REG11
1794 005646 032777 000002 173536      BIT #2,@DMP04    ;IS PGM CLOCK BIT SET?
1795 005654 001372      BNE 2$          ;BR IF YES
1796 005656 005337 001416      DEC TEMP         ;DEC COUNT
1797 005662 001361      BNE 1$          ;BR IF NOT DONE
1798 005664 000002      RTI             ;RETURN
1799
1800 005666 020040 000077      MQM: .ASCIZ / ?/
(2) 005672 005015 000      MCRLF: .ASCIZ <15><12>
(2) 005675 377 053520 020122      MPFAIL: .ASCIZ <377>/PWR FAILED. RESTART AT TEST /
(2) 005733 377 047105 020104      MEPASS: .ASCIZ <377>/END PASS CZDMF /
(2) 005755 377 000122      MR: .ASCIZ <377>/R/
(2) 005760 047377 020117 042504      MERR2: .ASCIZ <377>/NO DEVICES PRESENT./
(2) 006005 377 047111 052523      MERR3: .ASCIZ <377>/INSUFFICIENT DATA!/
(2) 006031 377 042524 052123      MTSTPC: .ASCIZ <377>/TEST PC-/
(2) 006043 377 047514 045503      MLOCK: .ASCIZ <377>/LOCK ON SELECTED TEST/
(2) 006072 051503 035122 000040      MCSRX: .ASCIZ /CSR: /
(2) 006100 042526 035103 000040      MVECX: .ASCIZ /VEC: /
(2) 006106 040520 051523 051505      MPASSX: .ASCIZ /PASSES: /
(2) 006117 105 051122 051117      MERRX: .ASCIZ /ERRORS: /
(2) 006130 042524 052123 047040      MTSTN: .ASCIZ /TEST NO: /
(2) 006142 000052      MASTEK: .ASCIZ /*/
(2) 006144 051777 052105 051440      MNEW: .ASCIZ <377>/SET SWITCH REG TO DMC11'S DESIRED ACTIVE./
(2) 006217 120 035103 000040      MERRPC: .ASCIZ /PC: /
(2) 006224 020212 020040 020040      XHEAD: .ASCIZ <212>/
(2) 006263 377 020040 020040      .ASCIZ <377>/
(2) 006322 020212 050040 020103      .ASCIZ <212>/ PC CSR STAT1 STAT2 STAT3/
(2) 006374 026777 026455 026455      .ASCIZ <377>/-----/
(2) 006450 044377 053517 046440      NUM: .ASCIZ <377>/HOW MANY DMC11'S TO BE TESTED?/
(2) 006510 041777 051123 040440      CSR: .ASCIZ <377>/CSR ADDRESS?/
(2) 006526 053377 041505 047524      VEC: .ASCIZ <377>/VECTOR ADDRESS?/
(2) 006547 377 051102 050040      PRIO: .ASCIZ <377>/BR PRIORITY LEVEL? (4,5,6,7)?/
(2) 006606 044777 020106 046504      CRAM: .ASCIZ <377>/IF DMC HAS CRAM (M8204) TYPE 'Y', IF CROM (M8200) TYPE 'N' ?/
(2) 006704 053777 044510 044103      MODU: .ASCIZ <377>/WHICH LINE UNIT? IF NONE TYPE 'N', IF M8201 TYPE '1', IF M8202 TYP
(2) 007016 051777 044527 041524      LINE: .ASCIZ <377>/SWITCH PAC#1 (DDCMP LINE #)?/
(2) 007054 051777 044527 041524      BM: .ASCIZ <377>/SWITCH PAC#2 (BM873 BOOT ADD)?/
(2) 007114 044777 020123 044124      CONN: .ASCIZ <377>/IS THE LOOP BACK CONNECTOR ON?/
(2) 007154 047377 020117 042504      NOACT: .ASCIZ <377>/NO DEVICES ARE SELECTED/
(2) 007205 377 051412 051127      SWMES: .ASCIZ <377><12>/SWR= /
(2) 007215 116 053505 020077      SWMES1: .ASCIZ /NEW? /
(2) 007223 377 042377 041515      CONERR: .ASCIZ <377><377>/DMC11 FOUND AT NON-STANDARD ADDRESS PC: /
(2) 007277 377 054105 042520      CNERR: .ASCIZ <377>/EXPECTED FOUND/
(2) 007320 024040 046504 024503      DMC1: .ASCIZ / (DMC) /
(2) 007330 024040 046513 024503      KMC1: .ASCIZ / (KMC) /
(2) 007340 042377 041515 030461      SPEED: .ASCIZ <377>/DMC11-AR(REMOTE,LOW SPEED) OR DMC11-AL(LOCAL,HIGH SPEED) TYPE 'R'
(2) 007454 053600 044510 044103      MV35: .ASCIZ <200>/WHICH MODEM TYPE, TYPE 'D' FOR DMC11-DA (RS232C),OR/
(2) 007540 052200 050131 020105      .ASCIZ <200>/TYPE 'F' FOR DMC11-FA (V.35) ? /
(2)
(2) 007602 .EVEN

```

```

(2) 007602 000005
1801 007604 006 003
1802 007606 001246
1803 007610 006 003
1804 007612 001250
1805 007614 006 003
1806 007616 001252
1807 007620 006 003
1808 007622 001254
1809 007624 006 002
1810 007626 001256
1811
1812
1813
1814
1815 007630 000000
1816 007672 007672
1817 007672 000000
1818 007734 007734
1819
1820
1821
1822
1823
1824
1825 007734 022737 000176 001202 CKSWR: CMP #SWREG,SWR ;IS THE SOFT SWR BEING USED?
1826 007742 001077 BNE CKSWR5 ;BR IF NO
1827 007744 105777 171234 TSTB @TKCSR ;IS DONE SET?
1828 007750 100003 BPL 2$ ;GO ON IF NOT SET
1829 007752 012737 177777 003734 MOV #-1,DONE ;IF DONE SET, SET FLAG
1830 007760 022777 000007 171220 2$: CMP #7,@TKDDBR ;WAS CTRL G TYPED? (7 BIT ASCII)
1831 007766 001404 BEQ 1$ ;BR IF YES
1832 007770 022777 000207 171210 CMP #207,@TKDDBR ;WAS CTRL G TYPED? (8 BIT ASCII)
1833 007776 001061 BNE CKSWR5 ;BR IF NO
1834 010000 010246 1$: MOV R2,-(SP) ;STORE R2
1835 010002 010346 MOV R3,-(SP) ;STORE R3
1836 010004 010446 MOV R4,-(SP) ;STORE R4
1837 010006 012737 177777 010144 MOV #-1,SWFLG ;SET SOFT TYPE OUT FLAG
1838 010014 005002 CKSWR1: CLR R2 ;CLEAR NEW SWR CONTENTS
1839 010016 012704 177777 MOV #-1,R4 ;SET FLAG TO ALL ONES
1840 010022 104402 007205 TYPE ,SWMES ;TYPE "SWR= "
1841 010026 104411 CKSWR2: CNVRT ;TYPE OUT PRESENT CONTENTS
1842 010030 010200 SOFTSW ;OF SOFT SWITCH REGISTER
1843 010032 104402 007215 CKSWR3: TYPE ,SWMES1 ;TYPE "NEW? "
1844 010036 004737 010146 CKSWR4: JSR PC,INCHAR ;GET RESPONSE
1845 010042 022707 000015 CMP #15,R3 ;WAS IT A CR?
1846 010046 001424 BEQ 5$ ;BR IF YES
1847 010050 022703 000012 CMP #12,R3 ;WAS IT A LF?
1848 010054 001416 BEQ 4$ ;BR IF YES
1849 010056 022703 000025 CMP #25,R3 ;WAS IT CTRL U?
1850 010062 001754 BEQ CKSWR1 ;BR IF YES(START OVER)
1851 010064 022703 000007 CMP #7,R3 ;IF CNTL G GET NEXT CHAR
1852 010070 001762 BEQ CKSWR4
1853 010072 005004 CLR R4 ;IT MUST BE A DIGIT SO CLR FLAG
1854 010074 042703 177770 BIC #177770,R3 ;ONLY 0-7 ARE LEGAL SO MASK OFF BITS
1855 010100 006302 ASL R2 ;SHIFT R2 3 TIMES

```

```

1856 010102 006302          ASL      R2
1857 010104 006302          ASL      R2
1858 010106 050302          BIS      R3,R2          :ADD LAST DIGIT
1859 010110 000752          BR       CKSWR4         :GET NEXT CHARACTER
1860 010112 012766 002002 000006 4$:  MOV     #.START,6(SP)   :LF WAS TYPED SO GO TO START
1861 010120 005704          5$:  TST      R4          :IS FLAG CLEAR?
1862 010122 001002          BNE     6$             :IF NOT DON'T CHANGE SOFT SWR
1863 010124 010277 171052    MOV     R2,@SWR         :IF YES THEN WRITE NEW CONTENTS TO SOFT SWR
1864 010130 005037 010144    6$:  CLR     SWFLG         :CLEAR TYPEOUT FLAG
1865 010134 012604          MOV     (SP)+,R4       :RESTORE R4
1866 010136 012603          MOV     (SP)+,R3       :RESTORE R3
1867 010140 012602          MOV     (SP)+,R2       :RESTORE R2
1868 010142 000207          CKSWR5: RTS          PC          :RETURN
1869
1870 010144 000000          SWFLG: 0
1871
1872 010146 105777 171032    INCHAR: TSTB          @TKCSR
1873 010152 100375          BPL     .-4
1874 010154 017703 171026    MOV     @TKDDBR,R3
1875 010160 105777 171024    TSTB   @TPCSR
1876 010164 100375          BPL     .-4
1877 010166 010377 171020    MOV     R3,@TPDDBR
1878 010172 042703 000200    BIC     #BIT7,R3
1879 010176 000207          RTS     PC
1880
1881 010200 000001          SCFTSW: 1
1882 010202 006          .BYTE  6,2
1883 010204 000176          SWREG

```

```

1884
1885
1886          ; ROUTINE USED TO "CYCLE" THROUGH UP TO 16 DMC11'S
1887          ; THIS ROUTINE SETS UP THE CONTROL ADDRESS FOR THE DIAGNOSTIC
1888          ; AND RUNS THE SPECIFIED DMC11'S.  THIS ROUTINE *MUST*
1889          ; BE RUN FIRST BEFORE ENTERING THE DIAGNOSTIC FOR THE
1890          ; SETUP NECESSARY.
1891          ;
1892
1893 010206 005737 001306      CYCLE:  TST      DMACTV      ; ARE ANY DMC11'S TO BE TESTED?
1894 010212 001004          BNE      1$      ; BR IF OK.
1895 010214 104402 007154    TYPE     ,NOACT    ; NO DMC11'S SELECTED!!
1896 010220 000000          HALT     ; STOP THE SHOW.
1897 010222 000776          BR       .-2      ; DISQUALIFY CONT. SW.
1898 010224 000241          1$:      CLC      ; CLEAR PROC. CARRY BIT.
1899 010226 006137 001316    ROL      RUN      ; UPDATE POINTER
1900 010232 005537 001316    ADC      RUN      ; CATCH CARRY FROM RUN
1901 010236 062737 000004 001322  ADD     #4,MILK    ; UPDATE POINTER
1902 010244 062737 000010 001320  ADD     #10,CREAM  ; UPDATE ADDRESS POINTER.
1903 010252 022737 001700 001320  CMP     #DM.MAP+200,CREAM
1904 010260 001006          BNE     2$      ; KEEP GOING; NOT ALL TESTED FOR.
1905 010262 012737 001500 001320  MOV     #DM.MAP,CREAM ; RESET ADDRESS POINTER.
1906 010270 012737 001702 001322  MOV     #CNT.MAP,MILK ; RESET PASS COUNT POINTER
1907 010276 033737 001316 001306  2$:    BIT     RUN,DMACTV ; IS THIS ONE ACTIVE?
1908 010304 001747          BEQ     1$      ; BR IF NO
1909 010306 013700 001320      MOV     CREAM,R0    ; GET ADDRESS POINTER
1910 010312 013702 001322      MOV     MILK,R2     ; GET PASS COUNT POINTER
1911 010316 012037 001404      MOV     (R0)+,DMCSR ; LOAD SYSTEM CTRL. REG
1912 010322 011037 001374      MOV     (R0),DMRVEC ; LOAD VECTOR
1913 010326 042737 177000 001374  BIC     #177000,DMRVEC ; CLEAR UNWANTED BITS
1914 010334 012037 001366      MOV     (R0)+,STAT1 ; LOAD STAT1
1915 010340 012037 001370      MOV     (R0)+,STAT2 ; LOAD STAT2
1916 010344 012037 001372      MOV     (R0)+,STAT3 ; LOAD STAT3
1917 010350 012237 001230      MOV     (R2)+,PASCNT ; LOAD PASS COUNT
1918 010354 012237 001232      MOV     (R2)+,ERRCNT ; LOAD ERRJR COUNT
1919 010360 012700 000002      MOV     #2,R0      ; SAVE CORE THIS WAY
1920 010364 013737 001404 001406  MOV     DMCSR,DMCSRH
1921 010372 005237 001406      INC     DMCSRH
1922 010376 013737 001406 001410  MOV     DMCSRH,DMCTL
1923 010404 005237 001410      INC     DMCTL
1924 010410 013737 001410 001412  MOV     DMCTL,DMP04
1925 010416 060037 001412      ADD     R0,DMP04
1926 010422 013737 001412 001414  MOV     DMP04,DMP06
1927 010430 060037 001414      ADD     R0,DMP06
1928
1929 010434 013737 001374 001376  MOV     DMRVEC,DMRLVL ; PTY LVL
1930 010442 060037 001376      ADD     R0,DMRLVL
1931 010446 013737 001376 001400  MOV     DMRLVL,DMTVEC ; TX VEC
1932 010454 060037 001400      ADD     R0,DMTVEC
1933 010460 013737 001400 001402  MOV     DMTVEC,DMTLVL ; TX LVL
1934 010466 060037 001402      ADD     R0,DMTLVL
1935
1936 010472 032737 000002 001236  BIT     #SW01,STRTSW ; IS TEST NO. SELECTED
1937 010500 001450          BEQ     7$      ; BR IF NO
1938 010502
1939 010502 005737 000042      4$:    TST     @#42    ; RUNNING IN AUTO MODE?
  
```


1940	010506	001045			BNE	7\$:BR IF YES
1941	010510	104402	005672		TYPE	,MCRLF		
1942	010514	104403			INSTR			:GET TEST NO.
1943	010516	006130			MTSTN			
1944	010520	104405			PARAM			
1945	010522	000001			1			
1946	010524	001000			1000			
1947	010526	001226			TSTNO			
1948	010530	000			.BYTE	0		
1949	010531	001			.BYTE	1		
1950	010532	012700	012526		MOV	#TST1,R0		
1951	010536	022710			5\$:	CMP	(PC)+,(R0)	:CMP FIRST WORD TO 12737
1952	010540	012737				MOV	(PC)+,@(PC)+	
1953	010542	001020			BNE	6\$:BR IF NOT SAME
1954	010544	023760	001226	000002	CMP	TSTNO,2(R0)		:DOES TSTNO MATCH?
1955	010552	001014			BNE	6\$:BR IF NO
1956	010554	022760	001226	000004	CMP	#TSTNO,4(R0)		:IS LAST WORD OK?
1957	010562	001010			BNE	6\$:BR IF NO
1958	010564	010037	001214		MOV	R0,RETURN		:IT IS A LEGAL TEST SO DO IT
1959	010570	104402	005755		TYPE	,MR		
1960	010574	042737	000002	001236	BIC	#SW01,STRISW		
1961	010602	000412			BR	8\$		
1962	010604	005720			6\$:	TST	(R0)+	:POP R0
1963	010606	020027	026342			CMP	R0,#TLAST+10	:AT END YET?
1964	010612	001351			BNE	5\$:BR IF NO
1965	010614	104402	005666		TYPE	,MQM		:YES ILLEGAL TEST NO.
1966	010620	000730			BR	4\$:TRY AGAIN
1967								
1968	010622	012737	012526	001214	7\$:	MOV	#TST1,RETURN	:PREPARE RETURN ADDRESS
1969	010630	013701	001404		8\$:	MOV	DMCSR,R1	:R1 = BASE DMC11 ADDRESS
1970	010634	000177	170354			JMP	@RETURN	:GO START TESTING.
1971								
1972								
1973								:ROUTINE USED TO "AUTO SIZE" THE DMC11
1974								:CSR AND VECTOR.
1975								:NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
1976								ADDRESS RANGE (160000:164000)
1977								AND THE VECTOR MAY BE ANY WHERE IN THE
1978								FLOATING VECTOR RANGE (300:770)
1979								
1980								
1981	010640							AUTO.SIZE:
1982	010640	000005				RESET		:INSURE A BUS INIT.
1983	010642	012702	001500		CSRMAP:	MOV	#DM.MAP,R2	:LOAD MAP POINTER.
1984	010646	005022			1\$:	CLR	(R2)+	:ZERO ENTIRE MAP
1985	010650	022702	001700			CMP	#DM.END,R2	:ALL DONE?
1986	010654	001374				BNE	1\$:BR IF NO
1987	010656	005037	001310			CLR	DMNUM	:SET OCTAL NUMBER OF DMC11'S TO 0
1988	010662	012702	001500			MOV	#DM.MAP,R2	:R2 POINTS TO DMC MAP
1989	010666	005037	001306			CLR	DMACTV	:CLEAR ACTIVE
1990	010672	032737	000001	001236		BIT	#SW00,STRISW	:QUESTIONS?
1991	010700	001002				BNE	+6	:BR IF YES
1992	010702	000137	011460			JMP	7\$:IF NO SKIP QUESTIONS
1993	010706	0:2737	000001	001256		MOV	#1,TEMP5	:START WITH 1
1994	010714	104403				INSTR		
1995	010716	006450				NUM		

1996	010720	104405				PARAM			
1997	010722	000001				1			
1998	010724	000020				16.			
1999	010726	001252				TEMP3			
2000	010730	000				.BYTE	0		
2001	010731	001				.BYTE	1		
2002	010732	013737	001252	001310		MOV	TEMP3,DMNUM	;DMNUM = HOW MANY	
2003	010740	104402	005672		12\$:	TYPE	,MCR LF		
2004	010744	104410				CONVRT		;TYPE WHICH DMC IS BEING DONE	
2005	010746	012210				WHICH		;TEMPS IS WHICH DMC	
2006	010750	005237	001256			INC	TEMPS		
2007	010754	104403				INSTR			
2008	010756	006510				CSR			
2009	010760	104405				PARAM			
2010	010762	160000				160000			
2011	010764	164000				164000			
2012	010766	001254				TEMP4			
2013	010770	000				.BYTE	0		
2014	010771	001				.BYTE	1		
2015	010772	013722	001254			MOV	TEMP4,(R2)+	;STORE CSR IN MAP	
2016	010776	104403				INSTR			
2017	011000	006526				VEC			
2018	011002	104405				PARAM			
2019	011004	000000				0			
2020	011006	000776				776			
2021	011010	001254				TEMP4			
2022	011012	000				.BYTE	0		
2023	011013	001				.BYTE	1		
2024	011014	013712	001254			MOV	TEMP4,(R2)	;STORE VECTOR IN MAP	
2025	011020	104402			10\$:	TYPE			
2026	011022	006547				PRIO		;ASK WHAT BR LEVEL	
2027	011024	004737	012474			JSR	PC,INTTY	;GET RESPONSE	
2028	011030	022703	000024			CMP	#24,R3		
2029	011034	101014				BHI	50\$;BR IF LESS THAN 4	
2030	011036	022703	000027			CMP	#27,R3		
2031	011042	103411				BLO	50\$;BR IF GREATER THAN 7	
2032	011044	012704	000011			MOV	#11,R4	;R4 = NUMBER OF SHIFTS	
2033	011050	006303				ASL	R3	;SHIFT R3 LEFT	
2034	011052	005304				DEC	R4	;DEC SHIFT COUNT	
2035	011054	001375				BNE	.-4	;BR IF NOT DONE	
2036	011056	042703	170777			BIC	#170777,R3	;BIC UNWANTED BITS	
2037	011062	050312				BIS	R3,(R2)	;PUT BR LEVEL IN STATUS MAP	
2038	011064	000403				BR	8\$;CONTINUE	
2039	011066	104402			50\$:	TYPE			
2040	011070	005666				MQM		;RESPONSE IS OUT OF LIMITS	
2041	011072	000752				BR	10\$;TRY AGAIN	
2042	011074	104402			8\$:	TYPE			
2043	011076	006606				CRAM		;DOES DMC HAVE CRAM?	
2044	011100	004737	012474			JSR	PC,INTTY	;GET REPLY	
2045	011104	022703	000131			CMP	#131,R3		
2046	011110	001427				BEQ	9\$;YES	
2047	011112	022703	000116			CMP	#116,R3	;NO	
2048	011116	001403				BEQ	40\$;NOT A Y OR N	
2049	011120	104402				TYPE			
2050	011122	005666				MQM		;TYPE '?'	
2051	011124	000763				BR	8\$;ASK AGAIN	

```

2052 011126 104402          40$:  TYPE
2053 011130 007340          SPEED
2054 011132 004737 012474  JSR    PC,INTTY      ;DMC11-AR OR DMC11-AL?
2055 011136 022703 000122  CMP    #122,R3      ;GET RESPONSE
2056 011142 001414          BEQ    16$          ;IS IT R
2057 011144 022703 000114  CMP    #114,R3      ;BR IF REMOTE
2058 011150 001403          BEQ    41$          ;IS IT L
2059 011152 104402          TYPE      ;BR IF LOCAL
2060 011154 005666          MQM
2061 011156 000763          BR     40$          ;TRY AGAIN
2062 011160 052762 000002 000004 41$:  BIS    #BIT1,4(R2)  ;SET BIT1 IN STAT3
2063 011166 000402          BR     16$          ;CONTINUE
2064 011170 052712 100000          9$:  BIS    #BIT15,(R2) ;SET BIT 15 IF CRAM
2065 011174          15$:
2066
2067 011174 104402          TYPE
2068 011176 006704          MODU
2069 011200 004737 012474  JSR    PC,INTTY      ;ASK WHICH LINE UNIT
2070 011204 022703 000021  CMP    #21,R3      ;GET REPLY
2071 011210 001417          BEQ    30$          ;'1'
2072 011212 022703 000022  CMP    #22,R3      ;'2'
2073 011216 001412          BEQ    31$          ;'N'
2074 011220 022703 000116  CMP    #116,R3
2075 011224 001403          BEQ    32$
2076 011226 104402          TYPE
2077 011230 005666          MQM
2078 011232 000760          BR     16$          ;IF NOT A 1,2 OR N TYPE '?'
2079 011234 052722 010000          32$:  BIS    #BIT12,(R2)+ ;TRY AGIAN
2080 011240 022222          CMP    (R2)+,(R2)+ ;SET BIT 12 IN STAT2 IF NO LU
2081 011242 000477          BR     33$          ;POP OVER STAT2 AND STAT3
2082 011244 052712 020000          31$:  BIS    #BIT13,(R2) ;SET BIT 13 IN STAT2 IF M8202
2083 011250 104402          30$:  TYPE
2084 011252 007114          CONN
2085 011254 004737 012474  JSR    PC,INTTY      ;ASK IF LOOP-BACK IS ON
2086 011260 022703 000131  CMP    #131,R3      ;GET REPLY
2087 011264 001406          BEQ    17$          ;Y
2088 011266 022703 000116  CMP    #116,R3      ;N
2089 011272 001436          BEQ    18$
2090 011274 104402          TYPE
2091 011276 005666          MQM
2092 011300 000763          BR     30$          ;IF NOT Y OR N TYPE '?'
2093 011302 052722 040000          17$:  BIS    #BIT14,(R2)+ ;TRY AGAIN
2094 011306 032762 020000 177776  BIT    #BIT13,-2(R2) ;TURNAROUND IS CONNECTED
2095 011314 001027          BNE   19$          ; IS IT A M8202?
2096 011316          440$:  ; YES, IGNOR.
2097
2098 011316 104402          TYPE      ; ASK QUESTION
2099 011320 007454          MV35     ; ABOUT MODEM TTYPE
2100 011322 004737 012474  JSR    PC,INTTY      ; GET ANSWER.
2101 011326 122703 000104  CMPB   #'D',R3      ; IS IT DMC11-DA?
2102 011332 001004          BNE   442$         ; NO.
2103
2104 011334 042762 000004 000002  BIC    #BIT2,2(R2)  ; YES INDICATE IT IN STAT3
2105 011342 000411          BR     441$
2106
2107 011344 122703 000106          442$:  CMPB   #'F',R3      ; IS IT A DMC11-FA (V.35)?

```

```

2108 011350 001403          BEQ      443$      ; YES-TAKE CARE OF IT.
2109
2110 011352 104402          TYPE
2111 011354 005666          MQM
2112 011356 000757          BR      440$      ; REASK QUESTION
2113
2114 011360 052762 000004 000002 443$:  BIS      #BIT2,2(R2)  ; YES V.35, RECORD IN STAT3
2115
2116 011366          441$:
2117 011366 000402          BR      19$
2118 011370 042722 040000    18$:  BIC      #BIT14,(R2)+ ;NO TURNAROUND
2119 011374          19$:
2120 011374 104403          INSTR
2121 011376 007016          LINE
2122 011400 104405          PARAM
2123 011402 000000          0
2124 011404 000377          377
2125 011406 001254          TEMP4
2126 011410          000      .BYTE  0
2127 011411          001      .BYTE  1
2128 011412 113722 001254    MOVB    TEMP4,(R2)+ ;STORE SWITCH PAC IN MAP
2129 011416 104403          INSTR
2130 011420 007054          BM
2131 011422 104405          PARAM
2132 011424 000000          0
2133 011426 000377          377
2134 011430 001254          TEMP4
2135 011432          000      .BYTE  0
2136 011433          001      .BYTE  1
2137 011434 113722 001254    MOVB    TEMP4,(R2)+ ;STORE SWITCH PAC IN MAP
2138 011440 005722          TST    (R2)+ ;POP OVER STAT3
2139 011442 005337 001252    33$:  DEC    TEMP3 ;DEC DMC COUNT
2140 011446 001402          BEQ    34$      ;BR IF DONE
2141 011450 000137 010740    JMP    12$      ;JUMP IF NOT
2142 011454 000137 012110    34$:  JMP    13$      ;CONTINUE
2143 011460 012701 160000    7$:  MOV    #160000,R1 ;SET FOR FIRST ADDRESS TO BE TESTED
2144 011464 012737 012202 000004  MOV    #6$,R4 ;SET FOR NON-EXISTANT DEVICE TIME OUT
2145 011472 005011          2$:  CLR    (R1) ;CLEAR SEL0
2146 011474 005711          TST    (R1) ;IF DMC11 DMC SR S/B 0
2147 011476 001172          BNE    3$      ;IF NO DEV ; TRAP TO 4. IF NO BIT 8 THEN NO DMC11
2148 011500 005061 000006    CLR    6(R1) ;CLEAR SEL6
2149 011504 005761 000006    TST    6(R1) ;IF DMC11 THEN DMRIC S/B =0.
2150 011510 001165          BNE    3$      ;BR IF NOT DMC11
2151 011512 012711 002000    MOV    #BIT10,(R1) ;SET ROM0
2152 011516 005061 000004    CLR    4(R1) ;CLEAR SEL4
2153 011522 012761 125252 000006  MOV    #125252,6(R1) ;WRITE THIS TO SEL6
2154 011530 052711 020000    BIS    #BIT13,(R1) ;WRITE IT!
2155 011534 022761 125252 000004  CMP    #125252,4(R1) ;WAS IT WRITTEN?
2156 011542 001004          BNE    21$     ;IF NO IT IS NOT C RAM
2157 011544 052762 100000 000002  BIS    #BIT15,2(R2) ;SET BIT15 IF C RAM
2158 011552 000431          BR     22$
2159 011554 012711 001000    21$:  MOV    #BIT9,(R1) ;SET ROM1
2160 011560 012761 100430 000006  MOV    #100430,6(R1) ;PUT INSTRUCTION IN SEL6
2161 011566 012711 001400    MOV    #BIT9!BIT8,(R1) ;CLOCK INSTRUCTION (MICRO PROC PC TO 0)
2162 011572 012711 002000    MOV    #BIT10,(R1) ;SET ROM0
2163 011576 022761 016472 000006  CMP    #016472,6(R1) ;IS IT LOCAL C RAM?

```

```

2164 011604 001411      BEQ      23$      ;BR IF YES
2165 011606 022761 016461 000006  CMP      #016461,6(R1) ;IS IT REMOTE CROM?
2166 011614 001410      BEQ      22$      ;BR IF YES
2167 011616 022761 177777 000006  CMP      #-1,6(R1)  ;NO CROM?
2168 011624 001404      BEQ      22$      ;BR IF YES
2169 011626 000516      BR       3$       ;NOT A DMC
2170 011630 052762 000002 000006 23$: BIS      #BIT1,6(R2) ;SET BIT 1 IN STAT3
2171      ;AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A DMC11 CSR ADDRESS.
2172 011636 010122      22$: MOV      R1,(R2)+ ;STORE CSR IN CORE TABLE.
2173 011640 012711 001000      15$: MOV      #BIT9,(R1) ;CLEAR LINE UNIT LOOP
2174 011644 005061 000004      CLR      4(R1) ;CLEAR PORT4
2175 011650 012761 122113 000006  MOV      #122113,6(R1) ;LOAD INSTRUCTION (CLR DTR)
2176 011656 052711 000400      BIS      #BIT8,(R1) ;CLOCK INSTRUCTION
2177 011662 012761 021264 000006  MOV      #021264,6(R1) ;LOAD INSTRUCTION
2178 011670 052711 000400      BIS      #BIT8,(R1) ;CLOCK INSTRUCTION
2179 011674 122761 000377 000004  CMPB     #377,4(R1) ;IS IT ALL ONES?
2180 011702 001003      BNE     .+10 ;BR IF NO
2181 011704 052712 010000      BIS      #BIT12,(R2) ;IF YES, NO LINE UNIT, SET STATUS BIT
2182 011710 000436      BR       20$
2183 011712 032761 000002 000004  BIT      #BIT1,4(R1) ;IS SWITCH A ONE?
2184 011720 001403      BEQ     .+10 ;BR IF M8201
2185 011722 052712 060000      BIS      #BIT13.BIT14,(R2) ;M8202 ASSUME CONNECTOR
2186 011726 000427      BR       20$ ;CONNECTOR ON)
2187 011730 032761 000010 000004  BIT      #BIT3,4(R1) ;IS MRDY SET
2188 011736 001023      BNE     20$ ;BR IF M8201 NO CONNECTOR (ON LINE)
2189 011740 012761 000100 000004  MOV      #BIT6,4(R1) ;LOAD PORT4
2190 011746 012761 122113 000006  MOV      #122113,6(R1) ;LOAD INSTRUCTION
2191 011754 052711 000400      BIS      #BIT8,(R1) ;CLOCK INSTRUCTION(SET DTR)
2192 011760 012761 021264 000006  MOV      #021264,6(R1) ;LOAD INSTRUCTION
2193 011766 052711 000400      BIS      #BIT8,(R1) ;CLOCK INSTRUCTION(READ MODEM REG)
2194 011772 032761 000010 000004  BIT      #BIT3,4(R1) ;IS MRDY SET NOW?
2195 012000 001402      BEQ     20$ ;BR IF NO CONNECTOR
2196 012002 052712 040000      BIS      #BIT14,(R2) ;SET STATUS BIT FOR CONNECTOR
2197 012006 005722      20$: TST     (R2)+ ;POP POINTER
2198 012010 012761 021324 000006  MOV      #021324,6(R1) ;PUT INSTRUCTION IN PORT6
2199 012016 012711 001400      MOV      #BIT9!BIT8,(R1) ;PORT4_LU 15
2200 012022 156122 000004      BISB     4(R1),(R2)+ ;STORE DDCMP LINE # IN TABLE
2201 012026 012761 021344 000006  MOV      #021344,6(R1) ;PORT6_INSTRUCTION
2202 012034 012711 001400      MOV      #BIT8!BIT9,(R1) ;CLOCK INSTR.
2203 012040 156122 000004      BISB     4(R1),(R2)+ ;STORE BM873 ADD IN TABLE
2204 012044 005722      TST     (R2)+ ;POP OVER STAT3
2205 012046 005011      CLR     (R1) ;CLEAR ROMI
2206 012050 005237 001310      INC     DMNUM ;UPDATE DEVICE COUNTER
2207 012054 022737 000020 001310  CMP      #20,DMNUM ;ARE MAX. NO. OF DEV FOUND?
2208 012062 001412      BEQ     13$ ;YES DON'T LOOK FOR ANY MORE.
2209 012064 005011      3$: CLR     (R1) ;CLEAR BIT 10
2210 012066 005061 000006      CLR     6(R1) ;CLEAR SEL 6
2211 012072 062701 000010      14$: ADD     #10,R1 ;UPDATE CSR POINTER ADDRESS
2212 012076 022701 164000      CMP      #164000,R1
2213 012102 001402      BEQ     13$ ;BR IF DONE
2214 012104 000137 011472      JMP     2$ ;JUMP IF NOT
2215 012110 005037 001306      13$: CLR     DMACTV
2216 012114 005737 001310      TST     DMNUM ;WERE ANY DMC11'S FOUND AT ALL?
2217 012120 001423      BEQ     5$ ;ERROR AUTO SIZER FOUND NO DMC11'S IN THIS SYS.
2218 012122 013701 001310      MOV     DMNUM,R1
2219 012126 010137 001314      MOV     R1,SAVNUM ;SAVE NUMBER OF DEVICES

```

```

2220 012132 000241          4$: CLC
2221 012134 006137 001306   ROL      DMACTV      ;GENERATE ACTIVE REGISTER OF DEVICES.
2222 012140 005237 001306   INC      DMACTV      ;SET THE BIT
2223 012144 005301          DEC      R1
2224 012146 001371          BNE     4$           ;BR IF MORE TO GENERATE
2225 012150 012737 000006 000004 MOV     #6,@#4       ;RESTORE TRAP VECTOR
2226 012156 013737 001306 001312 MOV     DMACTV,SAVACT ;SAVE ACTIVE REGISTER
2227 012164 000137 012216   JMP     VECMAP      ;GO FIND THE VECTOR NOW.
2228 012170 104402 005760   5$: TYPE  ,MERR2     ;NOTIFY OPR THAT NO DMC11'S FOUND.
2229 012174 005000          CLR     R0           ;MAKE DATA LIGHTS ZERO
2230 012176 000000          HALT
2231 012200 000776          BR     .-2           ;STOP THE SHOW
2232 012202 012716 012072   6$: MOV     #14$, (SP) ;DISABLE CONT. SW.
2233 012206 000002          RTI                    ;ENTERED BY NON-EXISTANT TIME-OUT.
2234                                     ;RETURN TO MAINSTREAM
2235 012210 000001          WHICH: 1
2236 012212      002      002  .BYTE  2,2
2237 012214 001256          TEMP5
2238
2239 012216 032737 000001 001236 VECMAP: BIT  #SW00,STRTSW
2240 012224 001114          BNE     5$
2241 012226 012737 000340 000022 MOV     #340,@#22    ;SET IOT TRAP PRIO TO 7
2242 012234 012737 012410 000020 MOV     #4$,@#20     ;SET IOT TRAP VECTOR
2243 012242 012702 001500          MOV     #DM.MAP,R2  ;SET SOFTWARE POINTER
2244 012246 012700 000300          MOV     #300,R0     ;FLOATING VECTORS START HERE.
2245 012252 012701 000302          MOV     #302,R1     ;PC OF IOT INSTR.
2246 012256 010120   1$: MOV     R1,(R0)+  ;START FILLING VECTOR AREA
2247 012260 012721 000004          MOV     #4,(R1)+   ;WITH .+2; IOT
2248 012264 022021          CMP     (R0)+,(R1)+ ;ADD 2 TO R0 +R1
2249 012266 020127 001000          CMP     R1,#1000
2250 012272 101771          BLOS   1$           ;BR IF MORE TO FILL
2251 012274 013737 001306 001246 MOV     DMACTV,TEMP1 ;STORE TEMPORALLY
2252 012302 006037 001246   2$: ROR     TEMP1     ;BRING OUT A BIT
2253 012306 103063          BCC     5$           ;BR IF ALL DONE
2254 012310 012704 000012          MOV     #12,R4     ;R4 IS INDEX REGISTER
2255 012314 016437 012460 177776 MOV     BRLVL(R4),PS ;SET PS TO 7
2256 012322 011201          MOV     (R2),R1
2257 012324 012761 000200 000004 MOV     #200,4(R1)
2258 012332 012711 001000          MOV     #BIT9,(R1) ;SET ROMI
2259 012336 012761 121111 000006 MOV     #121111,6(R1);PUT INSTRUCTION IN PORT6
2260 012344 012711 001400          MOV     #BIT9!BIT8,(R1);FORCE AN INTERRUPT
2261 012350 105200   7$: INCB   R0           ;STALL
2262 012352 001376          BNE     .-2         ;FOR TIME TO INTERUPT
2263 012354 162704 000002          SUB     #2,R4       ;GET NEXT LOWEST PS LEVEL
2264 012360 001404          BEQ     6$          ;BR IF R4 = 0
2265 012362 016437 012460 177776 MOV     BRLVL(R4),PS ;MOVE NEXT LOWER LEVEL IN PS
2266 012370 000767          BR     7$          ;BR TO DELAY
2267 012372 052762 005300 000002 6$: BIS     #5300,2(R2) ;NO INTERUPT ASSUME 300 AT LEVEL 5 AND FIX DMC11 LATER
2268 012400 005011   3$: CLR     (R1)       ;CLEAR ROMI
2269 012402 062702 000010          ADD     #10,R2      ;POP SOFTWARE POINTER
2270 012406 000735          BR     2$          ;KEEP GOING
2271 012410 051662 000002   4$: BIS     (SP),2(R2)  ;GET VECTOR ADDRESS
2272 012414 042762 000007 000002 BIC     #7,2(R2)    ;CLEAR JUNK
2273 012422 016405 012462          MOV     BRLVL+2(R4),R5 ;GET BR LEVEL OF DMC11
2274 012426 006305          ASL     R5           ;SHIFT LEVEL 4 PLACES
2275 012430 006305          ASL     R5           ;TO THE LEFT FOR THE

```

CZDME MACY11 30A(1052) 07-JUL-80 13:53 PAGE 47
CZDME.P11 01-JUL-80 15:29

GENERAL UTILITIES (TYPEOUT, ERROR, SCOPE, ETC)

SEQ 0047

```

2276 012432 006305      ASL      R5      ;STATUS TABLE
2277 012434 006305      ASL      R5
2278 012436 042705 170777 BIC      #170777,R5 ;CLEAR UNWANTED BITS
2279 012442 050562 000002 BIS      R5,2(R2) ;PUT BR LEVEL IN STATUS TABLE
2280 012446 022626      (MP      (SP)+,(SP)+ ;POP IOT JUNK OFF STACK
2281 012450 012716 012400 MOV      #3$, (SP) ;SET FOR RETURN
2282 012454 000002      RTI
2283 012456 000207      RTS      PC      ;ALL DONE WITH 'AUTO SIZING'
2284
2285 012460 000000      BPLVL: 0      ;LEVEL 0
2286 012462 000000      0      ;LEVEL 0
2287 012464 000700      200     ;LEVEL 4
2288 012466 000240      240     ;LEVEL 5
2289 012470 000300      300     ;LEVEL 6
2290 012472 000340      340     ;LEVEL 7
2291
2292
2293 012474 105777 166504      INITY: TSTB   @TKCSR ;WAIT FOR DONE
2294 012500 100375      BPL      -4
2295 012502 017703 166500      MOV      @TKDBR,R3 ;PUT CHAR IN R3
2296 012506 105777 166476      TSTB   @TPCSR ;WAIT UNTIL PRINTER IS READY
2297 012512 100375      BPL      -4
2298 012514 010377 166472      MOV      R3,@TPDBR ;ECHO CHAR
2299 012520 042703 000240      BIC      #BIT7.BITS,R3 ;MASK OFF LOWER CASE
2300 012524 000207      RTS      PC      ;RETURN
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313 012526 012737 000001 001226 TST1: MOV      #1,TSTNO
2314 012534 012737 012602 001216 MOV      #TST2,NEXT
2315
2316 012542 005077 166636      CLR      @DMCSR ;R1 CONTAINS BASE DMC11 ADDRESS
2317 012546 012702 000011      MOV      #11,R2 ;CLEAR SELO
2318 012552 104414      ROMCLK ;SAVE R2 FOR TYPEOUT
2319 012554 021224      021004!<20*11> ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
2320 012556 016104 000004      MOV      4(R1),R4 ;PORT4 LINE UNIT REG 11
2321 012562 042704 000054      BIC      #54,R4 ;PUT 'FOUND' IN R4
2322 012566 012705 000020      MOV      #20,R5 ;CLEAR UNKNOWN BITS
2323 012572 120504      CMPB   R5,R4 ;PUT 'EXPECTED' IN R5
2324 012574 001401      BEQ     1$ ;IS OUT READY SET?
2325 012576 104412      HLT     2 ;BR IF YES
2326 012600 104400      SCOPE  ;ERROR IN LU 11
2327
2328
2329
2330
2331

```

***** TEST 1 *****
*OUT CONTROL REGISTER READ/ONLY TEST
*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
*BITS ARE IN THE CORRECT STATE

TEST 1

:R1 CONTAINS BASE DMC11 ADDRESS
:CLEAR SELO
:SAVE R2 FOR TYPEOUT
:NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
:PORT4 LINE UNIT REG 11
:PUT 'FOUND' IN R4
:CLEAR UNKNOWN BITS
:PUT 'EXPECTED' IN R5
:IS OUT READY SET?
:BR IF YES
:ERROR IN LU 11
:SCOPE THIS TEST

***** TEST 2 *****
*IN CONTROL REGISTER READ/ONLY TEST
*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY


```

2332                                     ;*BITS ARE IN THE CORRECT STATE
2333                                     ;:*****
2334                                     ;
2335                                     ; TEST 2
2336                                     ;-----
2337 012602 012737 0000G2 001226 TST2: MOV #2,TSTNO
2338 012610 012737 012650 001216 MOV #TST3,NEXT
2339                                     ;R1 CONTAINS BASE DMC11 ADDRESS
2340 012616 012702 000012 MOV #12,R2 ;SAVE R2 FOR TYPEOUT
2341 012622 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
2342 012624 021244 021004!<20*12> ;PORT4 LINE UNIT REG 12
2343 012626 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
2344 012632 042704 000017 BIC #17,R4 ;CLEAR UNKNOWN BITS
2345 012636 005005 CLR R5 ;PUT 'EXPECTED' IN R5
2346 012640 120504 CMPB R5,R4 ;ARE ALL BITS CLEARED?
2347 012642 001401 BEQ 1$ ;BR IF YES
2348 012644 104002 HLT 2 ;ERROR IN LU 12
2349 012646 104400 1$: SCOPE ;SCOPE THIS TEST
2350
2351
2352                                     ;***** TEST 3 *****
2353                                     ;*MODEM CONTROL REGISTER READ/ONLY TEST
2354                                     ;*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
2355                                     ;*BITS ARE IN THE CORRECT STATE
2356                                     ;:*****
2357
2358                                     ; TEST 3
2359                                     ;-----
2360 012650 012737 000003 001226 TST3: MOV #3,TSTNO
2361 012656 012737 012722 001216 MOV #TST4,NEXT
2362                                     ;R1 CONTAINS BASE DMC11 ADDRESS
2363 012664 104412 MSTCLR ;MASTER CLEAR DMC11
2364 012666 012702 000013 MOV #13,R2 ;SAVE R2 FOR TYPEOUT
2365 012672 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2366 012674 021264 021004!<20*13> ;PORT4 LINE UNIT REG 13
2367 012676 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
2368 012702 042704 000213 BIC #213,R4 ;CLEAR UNKNOWN BITS
2369 012706 012705 000100 MOV #100,R5 ;PUT 'EXPECTED' IN R5
2370 012712 120504 CMPB R5,R4 ;ARE RING, DTR, AND MODEM READY SET?
2371 012714 001401 BEQ 1$ ;BR IF YES
2372 012716 104002 HLT 2 ;ERROR IN LU 13
2373 012720 104400 1$: SCOPE ;SCOPE THIS TEST
2374
2375
2376                                     ;***** TEST 4 *****
2377                                     ;*MAINTENANCE REGISTER READ/ONLY TEST
2378                                     ;*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
2379                                     ;*BITS ARE IN THE CORRECT STATE
2380                                     ;:*****
2381
2382                                     ; TEST 4
2383                                     ;-----
2384 012722 012737 000004 001226 TST4: MOV #4,TSTNO
2385 012730 012737 013024 001216 MOV #TST5,NEXT
2386                                     ;R1 CONTAINS BASE DMC11 ADDRESS
2387 012736 104412 MSTCLR ;MASTER CLEAR DMC11

```

```

2388 012740 012702 000017      MOV      #17,R2      ;SAVE R2 FOR TYPEOUT
2389 012744 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2390 012746 021364      021004!<20*17>    ;PORT4 LINE UNIT REG 17
2391 012750 016104 000004      MOV      4(R1),R4    ;PUT 'FOUND' IN R4
2392 012754 042704 000206      BIC      #206,R4     ;CLEAR UNKNOWN BITS
2393 012760 012705 000051      MOV      #51,R5     ;PUT 'EXPECTED' IN R5
2394 012764 032737 020000 001366      BIT      #BIT13,STAT1 ;IS LU AN M8202 OR M8201?
2395 012772 001004      BNE      2$         ;BR IF M8202
2396 012774 032737 040000 001366      BIT      #BIT14,STAT1 ;CONNECTOR???
2397 013002 001004      BNE      3$         ;BR IF M8201 WITH CONNECTOR
2398 013004 042704 000040      2$:      BIC      #40,R4     ;MASK OFF SI BIT IF M8202 OR M8201, NO CONNECTOR
2399 013010 042705 000040      BIC      #BIT5,R5   ;SI BIT IS UNKNOWN
2400 013014      3$:
2401 013014 120504      CMPB     R5,R4      ;ARE SI AND ICIR SET?
2402 013016 001401      BEQ      1$         ;BR IF YES
2403 013020 104002      HLT      2          ;ERROR IN LU 17
2404 013022 104400      1$:      SCOPE     ;SCOPE THIS TEST
2405
2406
2407      ;***** TEST 5 *****
2408      ;*LINE UNIT REGISTER WRITE/READ TEST
2409      ;*SET BITS IN LU REGISTER 12, VERIFY IT IS SET
2410      ;*CLEAR BITS IN LU REGISTER 12, VERIFY IT IS CLEAR
2411      ;:*****
2412
2413      ; TEST 5
2414      ;-----
2415 013024 012737 000005 001226 TST5:  MOV      #5,TSTNO
2416 013032 012737 013164 001216      MOV      #TST6,NEXT
2417 013040 012737 013054 001220      MOV      #1$,LOCK
2418
2419 013046 104412      MSTCLR    ;R1 CONTAINS BASE DMC11 ADDRESS
2420 013050 012702 000012      MOV      #12,R2     ;MASTER CLEAR DMC11
2421 013054 012761 000040 000004 1$:      MOV      #40,4(R1)  ;SAVE REGISTER ADDRESS FOR TYPEOUT
2422 013062 104414      ROMCLK    ;LOAD PORT4
2423 013064 122112      122112    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2424 013066 104414      ROMCLK    ;SET BITS IN LU-12
2425 013070 021245      021245    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2426 013072 012705 000040      MOV      #40,R5     ;READ LU-12
2427 013076 116104 000005      MOV      5(R1),R4   ;PUT 'EXPECTED' IN R5
2428 013102 042704 000337      MOV      #337,R4    ;PUT 'FOUND' IN R4
2429 013106 120504      BIC      #337,R4    ;CLEAR UNWANTED BITS
2430 013110 001401      CMPB     R5,R4     ;IS BIT5 SFT?
2431 013112 104003      BEQ      2$         ;BR IF YES
2432 013114 104401      HLT      3          ;ERROR, BIT 5 IS NOT SET
2433 013116 012737 013124 001220 2$:      SCOPE     ;SCOPE SUBTEST (SW09=1)
2434 013124 005061 007004      3$:      MOV      #3$,LOCK  ;NEW SCOPE
2435 013130 104414      CLR      4(R1)     ;LOAD PORT4
2436 013132 122112      ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2437 013134 104414      122112    ;CLEAR BIT 5 IN LU-12
2438 013136 021245      ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2439 013140 005005      021245    ;READ LU-12
2440 013142 116104 000005      CLR      R5        ;PUT 'EXPECTED' IN R5
2441 013146 042704 000337      MOV      5(R1),R4   ;PUT 'FOUND' IN R4
2442 013152 120504      BIC      #337,R4    ;CLEAR UNWANTED BITS
2443 013154 001401      CMPB     R5,R4     ;IS BIT5 CLEAR?
                BEQ      4$         ;BR IF YES

```

```

2444 013156 104003          HLT      3          ;ERROR, BIT5 IS NOT CLEAR
2445 013160 104401          4$: SCOPE1        ;SCOPE SUBTEST (SW09=1)
2446 013162 104400          SCOPE          ;SCOPE THIS TEST
2447
2448
2449
2450          ;***** TEST 6 *****
2451          ;*LINE UNIT REGISTER WRITE/READ TEST
2452          ;*SET BIT1 IN LU REGISTER 17, VERIFY IT IS SET
2453          ;*CLEAR BIT1 IN LU REGISTER 17, VERIFY IT IS CLEAR
2454          ;*****
2455          ; TEST 6
2456          ;-----
2457 013164 012737 000006 001226 TST6: MOV      #6,TSTNO
2458 013172 012737 013324 001216      MOV      #TST7,NEXT
2459 013200 012737 013214 001220      MOV      #18,LOCK
2460          ;R1 CONTAINS BASE DMC11 ADDRESS
2461 013206 104412          MSTCLR      ;MASTER CLEAR DMC11
2462 013210 012702 000017          MOV      #17,R2      ;SAVE REGISTER ADDRESS FOR TYPEOUT
2463 013214 012761 000001 000004 1$: MOV      #1,4(R1)    ;LOAD PORT4
2464 013222 104414          ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2465 013224 122117          122117      ;SET BIT1 IN LU-17
2466 013226 104414          ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2467 013230 021365          021365      ;READ LU-17
2468 013232 012705 000001          MOV      #1,R5      ;PUT 'EXPECTED' IN R5
2469 013236 116104 000005          MOV      5(R1),R4    ;PUT 'FOUND' IN R4
2470 013242 042704 000376          BIC      #376,R4     ;CLEAR UNWANTED BITS
2471 013246 120504          CMPB      R5,R4     ;IS BIT1 SET?
2472 013250 001401          BEQ      2$,        ;BR IF YES
2473 013252 104003          HLT      3          ;ERROR, BIT 1 IS NOT SET
2474 013254 104401          2$: SCOPE1        ;SCOPE SUBTEST (SW09=1)
2475 013256 012737 013264 001220 3$: MOV      #3$,LOCK
2476 013264 005061 000004          CLR      4(R1)      ;NEW SCOPE1
2477 013270 104414          ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2478 013272 122117          122117      ;CLEAR BIT 1 IN LU-17
2479 013274 104414          ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2480 013276 021365          021365      ;READ LU-17
2481 013300 005005          CLR      R5        ;PUT 'EXPECTED' IN R5
2482 013302 116104 000005          MOV      5(R1),R4    ;PUT 'FOUND' IN R4
2483 013306 042704 000376          BIC      #376,R4     ;CLEAR UNWANTED BITS
2484 013312 120504          CMPB      R5,R4     ;IS BIT1 CLEAR?
2485 013314 001401          BEQ      4$,        ;BR IF YES
2486 013316 104003          HLT      3          ;ERROR, BIT1 IS NOT CLEAR
2487 013320 104401          4$: SCOPE1        ;SCOPE SUBTEST (SW09=1)
2488 013322 104400          SCOPE          ;SCOPE THIS TEST
2489
2490
2491          ;***** TEST 7 *****
2492          ;*LINE UNIT REGISTER WRITE/READ TEST
2493          ;*FLOAT A 1 THROUGH LINE UNIT REGISTER 13
2494          ;*FLOAT A 0 THROUGH LINE UNIT REGISTER 13
2495          ;*****
2496          ; TEST 7
2497          ;-----
2498
2499 013324 012737 000007 001226 TST7: MOV      #7,TSTNO

```

```

2500 013332 012737 013534 001216      MOV    #TST10,NEXT
2501 013340 012737 013360 001220      MOV    #64$,LOCK
2502                                     ;R1 CONTAINS BASE DMC11 ADDRESS
2503 013346 104412                                     ;MASTER CLEAR DMC11
2504 013350 012702 000013      MOV    #13,R2      ;SAVE REGISTER ADDRESS FOR TYPEOUT
2505 013354 012700 000001      MOV    #1,R0      ;START WITH BIT 0
2506 013360                                     64$:
2507 013360 010061 000004      MOV    R0,4(R1)    ;PUT PATTERN INTO PORT4
2508 013364 042761 000257 000004      BIC    #257,4(R1) ;CLEAR UNWANTED BITS
2509 013372 104414                                     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2510 013374 122113      ROMCLK 122100.13 ;MOV DATA TO IBUS REGISTER 13
2511 013376 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2512 013400 021265      21005!<13*20> ;READ FROM IBUS REGISTER 13
2513 013402 010005      MOV    R0,R5      ;PUT EXPECTED IN R5
2514 013404 042705 000257      BIC    #257,R5    ;CLEAR UNWANTED BITS
2515 013410 116104 000005      MOV    5(R1),R4   ;PUT "FOUND" INTO R4
2516 013414 042704 000257      BIC    #257,R4    ;CLEAR UNWANTED BITS
2517 013420 120504      CMPB  R5,R4      ;DATA CORRECT?
2518 013422 001401      BEQ   65$        ;BR IF YES
2519 013424 104003      HLT   3          ;ERROR
2520 013426 104401                                     65$:
2521 013430 000241      SCOP1 ;SW09=1?
2522 013432 106100      CLC                                     ;CLEAR CARRY
2523 013434 001351      ROLB  R0        ;SHIFT BIT IN R0
2524 013436 012737 013452 001220      BNE   64$       ;IF R0=0 THEN DONE
2525 013444 012700 000001      MOV    #67$,LOCK ;NEW SCOP1
2526 013450 005100                                     69$:
2527 013452 005100                                     67$:
2528 013452 010061 000004      MOV    R0,4(R1)    ;PUT PATTERN INTO PORT4
2529 013456 042761 000257 000004      BIC    #257,4(R1) ;CLEAR UNWANTED BITS
2530 013464 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2531 013466 122113      ROMCLK 122100.13 ;MOV DATA TO IBUS REGISTER 13
2532 013470 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2533 013472 021265      21005!<13*20> ;READ FROM IBUS REGISTER 13
2534 013474 010005      MOV    R0,R5      ;PUT EXPECTED IN R5
2535 013476 042705 000257      BIC    #257,R5    ;CLEAR UNWANTED BITS
2536 013502 116104 000005      MOV    5(R1),R4   ;PUT "FOUND" INTO R4
2537 013506 042704 000257      BIC    #257,R4    ;CLEAR UNWANTED BITS
2538 013512 120504      CMPB  R5,R4      ;DATA CORRECT?
2539 013514 001401      BEQ   68$        ;BR IF YES
2540 013516 104003      HLT   3          ;ERROR
2541 013520 104401                                     68$:
2542 013522 005100      SCOP1 ;SW09=1?
2543 013524 000241      COM   R0        ;CHANGE TO FLOATING 1
2544 013526 106100      CLC                                     ;CLEAR CARRY
2545 013530 001347      ROLB  R0        ;SHIFT BIT IN R0
2546 013532 104400      BNE   69$       ;IF R0=0 THEN DONE
2547                                     SCOPE
2548                                     ;SCOPE THIS TEST

```

```

2549
2550 ***** TEST 10 *****
2551 ;*LINE UNIT REGISTER WRITE/READ TEST
2552 ;*FLOAT A 1 THROUGH LINE UNIT REGISTER 14
2553 ;*FLOAT A 0 THROUGH LINE UNIT REGISTER 14
2554 ;*****
2555 ; TEST 10

```



```

2612 013724 104412          MSTCLR          ;MASTER CLEAR DMC11
2613 013726 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2614 013730 021324          021324         ;PORT4 LU15
2615 013732 016104 000004  MOV      4(R1),R4 ;PUT 'FOUND' IN R4
2616 013736 113705 001370  MOVB    STAT2,R5 ;PUT 'EXPECTED' IN R5
2617 013742 120504          CMPB     R5,R4   ;SW OK?
2618 013744 001401          BEQ     1$      ;BR IF YES
2619 013746 104031          HLT     31      ;ERROR, SWITCH PAC READ ERROR
2620 013750 104400          1$: SCOPE     ;SCOPE THIS TEST
2621
2622
2623          ;***** TEST 12 *****
2624          ;*SWITCH PAC TEST
2625          ;*THIS TEST READS SWITCH PAC#2
2626          ;*THIS SWITCH PAC CONTAINS THE BM873 BOOT ADD
2627          ;*****
2628
2629          ; TEST 12
2630          ;-----
2631 013752 012737 000012 001226 TST12: MOV     #12,TSTNO
2632 013760 012737 014014 001216  MOV     #TST13,NEXT
2633
2634 013766 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
2635 013770 104414          ROMCLK          ;MASTER CLEAR DMC11
2636 013772 021344          021344         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2637 013774 016104 000004  MOV     4(R1),R4 ;PORT4 LU16
2638 014000 113705 001371  MOVB    STAT2+1,R5 ;PUT 'FOUND' IN R4
2639 014004 120504          CMPB     R5,R4   ;PUT 'EXPECTED' IN R5
2640 014006 001401          BEQ     1$      ;SW OK?
2641 014010 104031          HLT     31      ;BR IF YES
2642 014012 104400          1$: SCOPE     ;ERROR, SWITCH PAC READ ERROR
2643          ;SCOPE THIS TEST
2644
2645          ;***** TEST 13 *****
2646          ;*LINE UNIT CLOCK TEST
2647          ;*THIS TEST VERIFYS THAT THE LU INTERNAL CLOCK
2648          ;*(BIT 1 IN LU-17) IS WORKING
2649          ;*****
2650
2651          ; TEST 13
2652          ;-----
2653 014014 012737 000013 001226 TST13: MOV     #13,TSTNO
2654 014022 012737 014114 001216  MOV     #TST14,NEXT
2655
2656 014030 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
2657 014032 005037 001416          CLR     TEMP    ;MASTER CLEAR DMC11
2658 014036          1$:          ;PREPARE FOR DELAY
2659 014036 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2660 014040 021364          021364         ;PORT4 LU-17
2661 014042 032761 000002 000004  BIT     #2,4(R1) ;IS CLOCK BIT SET?
2662 014050 001004          BNE     2$      ;BR IF YES
2663 014052 005237 001416          INC     TEMP    ;DELAY
2664 014056 001367          BNE     1$      ;DELAY FINISHED?
2665 014060 104004          HLT     4       ;ERROR BIT IS STUCK CLEAR
2666 014062 005037 001416          2$: CLR     TEMP ;PREPARE FOR DELAY
2667 014066          3$:

```

```

2668 014066 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2669 014070 021364 021364 ;PORT4 LU-17
2670 014072 032761 000002 00U004 BIT #2,4(R1) ;IS CLOCK BIT CLEAR?
2671 014100 001404 BEQ 4$ ;BR IF YES
2672 014102 005237 001416 INC TEMP ;DELAY
2673 014106 001367 BNE 3$ ;BR IF DELAY NOT DONE
2674 014110 104004 HLT 4 ;ERROR BIT IS STUCK SET
2675 014112 104400 4$: SCOPE

```

```

2676
2677
2678 ;***** TEST 14 *****
2679 ;*OUT DATA SILO TEST
2680 ;*SET SOM AND LOAD OUT DATA SILO
2681 ;*VERIFY THAT OCOR SET, INDICATING THAT THE
2682 ;*CHARACTER IS AT THE BOTTOM OF THE OUT SILO
2683 ;*****

```

```

2684 ; TEST 14
2685 -----
2686
2687 014114 012737 000014 001226 TST14: MOV #14,TSTNO
2688 014122 012737 014214 001216 MOV #TST15,NEXT
2689 ;R1 CONTAINS BASE DMC11 ADDRESS
2690 014130 104412 MSTCLR ;MASTER CLEAR DMC11
2691 014132 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
2692 014136 012761 000001 000004 MOV #1,4(R1) ;LOAD PORT4 WITH BIT0
2693 014144 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2694 014146 122111 122111 ;SET SOM
2695 014150 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2696 014152 122110 122110 ;LOAD OUT DATA SILO
2697 014154 104416 000002 TIMER, 2 ;WAIT FOR OCOR
2698 014160 012702 000017 MOV #17,R2 ;SAVE ADDRESS FOR TYPEOUT
2699 014164 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2700 014166 021364 021364 ;PORT4 LU 17
2701 014170 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
2702 014174 042704 000357 BIC #357,R4 ;CLEAR UNWANTED BITS
2703 014200 012705 000020 MOV #20,R5 ;PUT 'EXPECTED' IN R5
2704 014204 120504 CMPB R5,R4 ;IS OCOR SET?
2705 014206 001401 BEQ 1$ ;BR IF YES
2706 014210 104005 HLT 5
2707 014212 1$:
2708 014212 104400 SCOPE ;SCOPE THIS TEST

```

```

2709
2710 ;***** TEST 15 *****
2711 ;*DDCMP TEST OF RTS AND OUT ACTIVE
2712 ;*SET SOM AND LOAD OUT DATA SILO
2713 ;*SINGLE STEP 2 DATA CLOCKS, VERIFY
2714 ;*THAT RTS AND ACTIVE ARE SET
2715 ;*****

```

```

2716 ; TEST 15
2717 -----
2718
2719
2720 014214 012737 000015 001226 TST15: MOV #15,TSTNO
2721 014222 012737 014352 001216 MOV #TST16,NEXT
2722 ;R1 CONTAINS BASE DMC11 ADDRESS
2723 014230 104412 MSTCLR ;MASTER CLEAR DMC11

```



```

2780 014440 012702 000017      MOV      #17,R2      ;SAVE ADDRESS FOR TYPEOUT
2781 014444 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2782 014446 021364      021364      ;PORT4 LU 17
2783 014450 016104 000004      MOV      4(R1),R4    ;PUT 'FOUND' IN R4
2784 014454 042704 000357      BIC      #357,R4     ;CLEAR UNWANTED BITS
2785 014460 005005      CLR      R5         ;PUT 'EXPECTED' IN R5
2786 014462 120504      CMPB     R5,R4      ;IS OCOR CLEARED?
2787 014464 001401      BEQ      1$         ;BR IF YES
2788 014466 104005      HLT      5
2789 014470
2790 014470 012702 000013      1$: MOV      #13,R2      ;SAVE ADDRESS FOR TYPEOUT
2791 014474 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2792 014476 021264      021264      ;PORT4 LU 13
2793 014500 016104 000004      MOV      4(R1),R4    ;PUT EXPECTED IN R4
2794 014504 042704 000337      BIC      #337,R4     ;CLEAR UNWANTED BITS
2795 014510 005005      CLR      R5         ;PUT 'EXPECTED' IN R5, RTS SHOULD BE CLEARED
2796 014512 120504      CMPB     R5,R4      ;IS RTS OK?
2797 014514 001401      BEQ      2$         ;BR IF YES
2798 014516 104005      HLT      5         ;RTS ERROR
2799 014520
2800 014520 012702 000011      2$: MOV      #11,R2      ;SAVE ADDRESS FOR TYPEOUT
2801 014524 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2802 014526 021224      021224      ;PORT4 LU11
2803 014530 016104 000004      MOV      4(R1),R4    ;PUT 'FOUND' IN R4
2804 014534 012705 000020      MOV      #BIT4,R5    ;ONLY OUT READY SHOULD BE SET
2805 014540 120504      CMPB     R5,R4      ;IS ACTIVE CLEAR?
2806 014542 001401      BEQ      3$         ;BR IF YES
2807 014544 104005      HLT      5         ;ERROR ACTIVE NOT CLEARED
2808 014546
2809 014546 104400      3$: SCOPE          ;SCOPE THIS TEST
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822 014550 012737 000017 001226 TST17: MOV      #17,TSTNO
2823 014556 012737 014732 001216      MOV      #TST20,NEXT
2824
2825 014564 104412      MSTCLR     ;R1 CONTAINS BASE DMC11 ADDRESS
2826 014566 012711 004000      MOV      #BIT11,(R1) ;MASTER CLEAR DMC11
2827 014572 004737 027034      JSR      PC,OUTRDY   ;SET LINE UNIT LOOP
2828 014576 012761 000001 000004      MOV      #1,4(R1)    ;WAIT FOR OUT-READY
2829 014604 104414      ROMCLK     ;SET BIT0 IN PORT4
2830 014606 122111      122111     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2831 014610 012705 000000      MOV      #0,R5      ;SET SOM!
2832 014614 004737 027034      JSR      PC,OUTRDY   ;LOAD CHARACTER IN R5 FOR TYPEOUT
2833 014620 010561 000004      MOV      R5,4(R1)   ;WAIT FOR OUT-READY
2834 014624 104414      ROMCLK     ;LOAD PORT4 WITH CHARACTER
2835 014626 122110      122110     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
                ;LOAD OUT DATA

```

```

:***** TEST 17 *****
:*DDCMP TRANSMITTER TEST
:*SINGLE CLOCK THE CHARACTER 0
:*VERIFY EACH BIT POSITION AS IT
:*PASSES THE BIT WINDOW (SI BIT)
:*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
:*****

```

```

: TEST 17
:-----

```

```

2836 014630 004737 026702 JSR PC,OCOR ;WAIT FOR OCOR TO SET
2837 014634 005003 CLR R3 ;CLEAR BIT COUNTER
2838 014636 010502 MOV R5,R2 ;LOAD CHARACTER IN R2
2839 014640 104415 000002 DATACLK, 2 ;2 TICKS TO SET UP TRANSMITTER
2840 014644 104415 000001 1$: DATACLK, 1 ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2841 014650 106002 RORB R2 ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
2842 014652 103005 BCC 2$ ;BR IF CARRY CLEAR
2843 014654 004737 026650 JSR PC,GETSI ;GET THE WINDOW
2844 014660 103406 BCS 3$ ;BR IF BIT IS A MARK
2845 014662 104006 HLT 6 ;ERROR BIT WAS A SPACE
2846 014664 000404 BR 3$ ;CONTINE WITH TEST
2847 014666 004737 026650 2$: JSR PC,GETSI ;GET THE WINDOW
2848 014672 103001 BCC 3$ ;BR IF BIT IS A SPACE
2849 014674 104006 HLT 6 ;ERROR BIT WAS A MARK
2850 014676 3$:
2851 014676 005203 INC R3 ;NEXT BIT
2852 014700 022703 000010 CMP #10,R3 ;DONE YET?
2853 014704 001357 BNE 1$ ;BR IF NO
2854 014706 104415 000014 DATACLK, 14 ;CLOCK TRANSMITTER 14 MORE TICKS
2855 014712 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2856 014714 021264 021264 ;PORT4 LU-13
2857 014716 032761 000040 000004 BIT #BIT5,4(R1) ;RTS SHOULD BE CLEAR NOW
2858 014724 001401 BEQ 4$ ;BR IF YES
2859 014726 104034 HLT 34 ;ERROR, RTS NOT CLEAR
2860 014730 104400 4$: SCOPE ;SCOPE THIS TEST

```

```

2861
2862
2863 ;***** TEST 20 *****
2864 ;*DDCMP TRANSMITTER TEST
2865 ;*SINGLE CLOCK THE CHARACTER 125
2866 ;*VERIFY EACH BIT POSITION AS IT
2867 ;*PASSES THE BIT WINDOW (SI BIT)
2868 ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2869 ;*****

```

```

2870
2871 ; TEST 20
2872 ;-----
2873 014732 012737 000020 001226 TST20: MOV #20,TSTNO
2874 014740 012737 015114 001216 MOV #TST21,NEXT
2875 ;R1 CONTAINS BASE DMC11 ADDRESS
2876 014746 104412 MSTCLR ;MASTER CLEAR DMC11
2877 014750 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
2878 014754 004737 027034 JSR PC,OUTRDY ;WAIT FOR OUT-READY
2879 014760 012761 000001 000004 MOV #1,4(R1) ;SET BIT0 IN PORT4
2880 014766 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
2881 014770 122111 122111 ;SET SOM!
2882 014772 012705 000125 MOV #125,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
2883 014776 004737 027034 JSR PC,OUTRDY ;WAIT FOR OUT-READY
2884 015002 010561 000004 MOV R5,4(R1) ;LOAD PORT4 WITH CHARACTER
2885 015006 104414 ROMCLK ;NEXT WRD IS INSTRUCTION, ROMCLK PC-5304
2886 015010 122110 122110 ;LOAD OUT DATA
2887 015012 004737 026702 JSR PC,OCOR ;WAIT FOR OCOR TO SET
2888 015016 005003 CLR R3 ;CLEAR BIT COUNTER
2889 015020 010502 MOV R5,R2 ;LOAD CHARACTER IN R2
2890 015022 104415 000002 DATACLK, 2 ;2 TICKS TO SET UP TRANSMITTER
2891 015026 104415 000001 1$: DATACLK, 1 ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)

```

```

2892 015032 106002          RORB    R2          ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
2893 015034 103005          BCC     2$         ;BR IF CARRY CLEAR
2894 015036 004737 026650   JSR     PC,GETSI   ;GET THE WINDOW
2895 015042 103406          BCS     3$         ;BR IF BIT IS A MARK
2896 015044 104006          HLT     6          ;ERROR BIT WAS A SPACE
2897 015046 000404          BR      3$         ;CONTINE WITH TEST
2898 015050 004737 026650   2$: JSR     PC,GETSI   ;GET THE WINDOW
2899 015054 103001          BCC     3$         ;BR IF BIT IS A SPACE
2900 015056 104006          HLT     6          ;ERROR BIT WAS A MARK
2901 015060                    3$:
2902 015060 005203          INC     R3         ;NEXT BIT
2903 015062 022703 000010   CMP     #10,R3    ;DONE YET?
2904 015066 001357          BNE     1$         ;BR IF NO
2905 015070 104415 000014   DATACLK,          14 ;CLOCK TRANSMITTER 14 MORE TICKS
2906 015074 104414          ROMCLK           ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2907 015076 021264          021264          ;PORT4 LU-13
2908 015100 032761 000040 000004   BIT     #BIT5,4(R1) ;RTS SHOULD BE CLEAR NOW
2909 015106 001401          BEQ     4$         ;BR IF YES
2910 015110 104034          HLT     34        ;ERROR, RTS NOT CLEAR
2911 015112 104400          4$: SCOPE        ;SCOPE THIS TEST
2912
2913
2914          :***** TEST 21 *****
2915          :*DDCMP TRANSMITTER TEST
2916          :*SINGL CLOCK THE CHARACTER 252
2917          :*VERIFY EACH BIT POSITION AS IT
2918          :*PASSES THE BIT WINDOW (SI BIT)
2919          :*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2920          :*****
2921
2922          : TEST 21
2923          :-----
2924 015114 012737 000021 001226 TST21: MOV     #21,TSTNO
2925 015122 012737 015276 001216   MOV     #TST22,NEXT
2926
2927 015130 104412          MSTCLR           ;R1 CONTAINS BASE DMC11 ADDRESS
2928 015132 012711 004000   MOV     #BIT11,(R1) ;MASTER CLEAR DMC11
2929 015136 004737 027034   JSR     PC,OUTRDY  ;SET LINE UNIT LTOP
2930 015142 012761 000001 000004   MOV     #1,4(R1)   ;WAIT FOR OUT-READY
2931 015150 104414          ROMCLK           ;SET BIT0 IN PORT4
2932 015152 122111          122111          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2933 015154 012705 000252   MOV     #252,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
2934 015160 004737 027034   JSR     PC,OUTRDY  ;SET SOM!
2935 015164 010561 000004   MOV     R5,4(R1)   ;LOAD CHARACTER IN R5 FOR TYPEOUT
2936 015170 104414          ROMCLK           ;WAIT FOR OUT-READY
2937 015172 122110          122110          ;LOAD PORT4 WITH CHARACTER
2938 015174 004737 026702   JSR     PC,OCOR    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2939 015200 005003          CLR     R3         ;LOAD OUT DATA
2940 015202 010502          MOV     R5,R2     ;WAIT FOR OCOR TO SET
2941 015204 104415 000002   DATACLK,          2 ;CLEAR BIT COUNTER
2942 015210 104415 000001   1$: DATACLK,          1 ;LOAD CHARACTER IN R2
2943 015214 106002          RORB    R2         ;2 TICKS TO SET UP TRANSMITTER
2944 015216 103005          BCC     2$         ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2945 015220 004737 026650   JSR     PC,GETSI   ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
2946 015224 103406          BCS     3$         ;BR IF CARRY CLEAR
2947 015226 104006          HLT     6          ;GET THE WINDOW
                ;BR IF BIT IS A MARK
                ;ERROR BIT WAS A SPACE

```

```

2948 015230 000404          BR      3$          ;CONTINE WITH TEST
2949 015232 004737 026650 2$: JSR     PC,GETSI    ;GET THE WINDOW
2950 015236 103001          BCC    3$          ;BR IF BIT IS A SPACE
2951 015240 104006          HLT    6           ;ERROR BIT WAS A MARK
2952 015242          3$:
2953 015242 005203          INC     R3         ;NEXT BIT
2954 015244 022703 000010  CMP    #10,R3     ;DONE YET?
2955 015250 001357          BNE    1$         ;BR IF NO
2956 015252 104415 000014  DATACLK, 14      ;CLOCK TRANSMITTER 14 MORE TICKS
2957 015256 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2958 015260 021264          021264          ;PORT4 LU-13
2959 015262 032761 000040 000004 BIT    #BIT5,4(R1) ;RTS SHOULD BE CLEAR NOW
2960 015270 001401          BEQ    4$         ;BR IF YES
2961 015272 104034          HLT    34        ;ERROR, RTS NOT CLEAR
2962 015274 104400          4$: SCOPE       ;SCOPE THIS TEST
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974

```

```

:***** TEST 22 *****
:*DDCMP TRANSMITTER TEST
:*SINGLE CLOCK THE CHARACTER 377
:*VERIFY EACH BIT POSITION AS IT
:*PASSES THE BIT WINDOW (SI BIT)
:*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
:*****

```

```

2975 015276 012737 000022 001226 TST22: MOV    #22,TSTNO
2976 015304 012737 015460 001216  MOV    #TST23,NEXT
2977
2978 015312 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
2979 015314 012711 004000  MOV    #BIT11,(R1) ;MASTER CLEAR DMC11
2980 015320 004737 027034  JSR    PC,OUTRDY  ;SET LINE UNIT LOOP
2981 015324 012761 000001 000004  MOV    #1,4(R1)   ;WAIT FOR OUT-READY
2982 015332 104414          ROMCLK          ;SET BIT0 IN PORT4
2983 015334 122111          122111          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2984 015336 012705 000377  MOV    #377,R5   ;LOAD CHARACTER IN R5 FOR TYPEOUT
2985 015342 004737 027034  JSR    PC,OUTRDY ;SET SOM!
2986 015346 010561 000004  MOV    R5,4(R1)  ;LOAD CHARACTER IN R5 FOR TYPEOUT
2987 015352 104414          ROMCLK          ;WAIT FOR OUT-READY
2988 015354 122110          122110          ;LOAD PORT4 WITH CHARACTER
2989 015356 004737 026702  JSR    PC,OCOR   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2990 015362 005003          CLR    R3        ;LOAD OUT DATA
2991 015364 010502          MOV    R5,R2     ;WAIT FOR OCOR TO SET
2992 015366 104415 000002  DATACLK, 2      ;CLEAR BIT COUNTER
2993 015372 104415 000001  1$: DATACLK, 1   ;LOAD CHARACTER IN R2
2994 015376 106002          RORB   R2        ;2 TICKS TO SET UP TRANSMITTER
2995 015400 103005          BCC    2$        ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2996 015402 004737 026650  JSR    PC,GETSI  ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
2997 015406 103406          BCS    3$        ;BR IF CARRY CLEAR
2998 015410 104006          HLT    6         ;GET THE WINDOW
2999 015412 000404          BR     3$        ;BR IF BIT IS A MARK
3000 015414 004737 026650  2$: JSR    PC,GETSI ;ERROR BIT WAS A SPACE
3001 015420 103001          BCC    3$        ;CONTINE WITH TEST
3002 015422 104006          HLT    6         ;GET THE WINDOW
3003          3$:      ;BR IF BIT IS A SPACE
                   ;ERROR BIT WAS A MARK

```

```

3004 015424 005203          INC      R3          ;NEXT BIT
3005 015426 022703 000010  CMP      #10,R3     ;DONE YET?
3006 015432 001357          BNE      1$         ;BR IF NO
3007 015434 104415 000014  DATACLK,          14 ;CLOCK TRANSMITTER 14 MORE TICKS
3008 015440 104414          ROMCLK             ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3009 015442 021264          021264            ;PORT4 LU-13
3010 015444 032761 000040 000004  BIT      #BIT5,4(R1) ;RTS SHOULD BE CLEAR NOW
3011 015452 001401          BEQ      4$         ;BR IF YES
3012 015454 104034          HLT      34         ;ERROR, RTS NOT CLEAR
3013 015456 104400          4$: SCOPE          ;SCOPE THIS TEST
3014
3015

```

```

:***** TEST 23 *****
:*DDCMP TRANSMITTER TEST
:*SINGLE CLOCK A BINARY COUNT PATTERN
:*VERIFY EACH BIT POSITION AS IT
:*PASSES THE BIT WINDOW (SI BIT)
:*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
:*AND R5 CONTAINS THE CHARACTER THAT FAILED
:*****

```

: TEST 23

```

3026
3027 015460 012737 000023 001226 TST23: MOV      #23,TSTNO
3028 015466 012737 015666 001216  MOV      #TST24,NEXT
3029
3030 015474 104412          MSTCLR             ;R1 CONTAINS BASE DMC11 ADDRESS
3031 015476 012711 004000  MOV      #BIT11,(R1) ;MASTER CLEAR DMC11
3032 015502 005003          CLR      R3         ;SET LINE UNIT LOOP
3033 015504 005004          CLR      R4         ;R3 CONTAINS BIT COUNT
3034 015506 005005          CLR      R5         ;R4 CONTAINS CHAR TO BE LOADED IN SILO
3035 015510 004737 027034  JSR      PC,OUTRDY  ;R5 CONTAINS CHARACTER CURRENTLY BEING SHIFTED OUT
3036 015514 012761 000001 000004  MOV      #1,4(R1)   ;WAIT FOR OUT-READY
3037 015522 104414          ROMCLK             ;SET BIT0 IN PORT4
3038 015524 122111          122111            ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3039 015526 004737 027034  JSR      PC,OUTRDY  ;SET SOM!
3040 015532 010461 000004  MOV      R4,4(R1)   ;WAIT FOR OUT-READY
3041 015536 104414          ROMCLK             ;LOAD PORT4 WITH CHARACTER
3042 015540 122110          122110            ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3043 015542 005204          INC      R4         ;LOAD OUT DATA
3044 015544 004737 027034  JSR      PC,OUTRDY  ;INCREMENT TO NEXT CHARACTER
3045 015550 010461 000004  MOV      R4,4(R1)   ;WAIT FOR OUT-READY
3046 015554 104414          ROMCLK             ;LOAD PORT4 WITH CHARACTER
3047 015556 122110          122110            ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3048 015560 004737 026702  JSR      PC,OCOR    ;LOAD OUT DATA
3049 015564 104415 000002          DATACLK,          2 ;WAIT FOR OCOR TO SET
3050 015570 005003          4$: CLR      R3         ;2 TICKS TO SET UP TRANSMITTER
3051 015572 010502          1$: MOV      R5,R2    ;CLEAR BIT COUNTER
3052 015574 104415 000001          DATACLK,          1 ;LOAD CHARACTER IN R2
3053 015600 106002          RORB      R2        ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
3054 015602 103005          BCC      2$         ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
3055 015604 004737 026650  JSR      PC,GETSI   ;BR IF CARRY CLEAR
3056 015610 103406          BCS      3$         ;GET THE WINDOW
3057 015612 104006          HLT      6         ;BR IF BIT IS A MARK
3058 015614 000404          BR       3$         ;ERROR BIT WAS A SPACE
3059 015616 004737 026650          2$: JSR      PC,GETSI ;CONTINUE WITH TEST
3059 015616 004737 026650          2$: JSR      PC,GETSI ;GET THE WINDOW

```

```

3060 015622 103001          BCC 3$      ;BR IF BIT IS A SPACE
3061 015624 104006          HLT 6       ;ERROR BIT WAS A MARK
3062 015626                3$:
3063 015626 005203          INC R3      ;NEXT BIT
3064 015630 022703 000010  CMP #10,R3 ;DONE YET?
3065 015634 001357          BNE 1$     ;BR IF NO
3066 015636 005204          INC R4     ;NEXT CHARACTER
3067 015640 004737 027034  JSR PC,OUTRDY ;WAIT FOR OUT-READY
3068 015644 010461 000004  MOV R4,4(R1) ;LOAD PORT4 WITH CHARACTER
3069 015650 104414          ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3070 015652 122110          122110    ;LOAD OUT DATA
3071 015654 005205          INC R5     ;NEXT CHARACTER
3072 015656 022705 000400  CMP #400,R5 ;DONE YET?
3073 015662 001342          BNE 4$     ;BR IF NO
3074 015664 104400          5$: SCOPE ;SCOPE THIS TEST
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084

```

```

:***** TEST 24 *****
:*DDCMP STRIP SYNC TEST
:*SET LU LOOP, SINGLE STEP 5 SYNCs.
:*VERIFY THAT IN ACTIVE DOES NOT SET
:*****

```

: TEST 24

```

3085 015666 012737 000024 001226 TST24: MOV #24,TSTNO
3086 015674 012737 015754 001216  MOV #TST25,NEXT
3087
3088 015702 104412          MSTCLR    ;R1 CONTAINS BASE DMC11 ADDRESS
3089 015704 012711 004000  MOV #BIT11,(R1) ;MASTER CLEAR DMC11
3090 015710 012702 000012  MOV #12,R2    ;SET LU LOOP
3091 015714 004737 026720  JSR PC,SYNC   ;SAVE LU REG FOR TYPEOUT
3092 015720 000005          5          ;SINGLE CLOCK 5 SYNC CHARACTERS
3093 015722 104415 000054  DATACLK, 54
3094 015726 104414          ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3095 015730 021244          021244    ;PORT4,LU12
3096 015732 016104 000004  MOV 4(R1),R4 ;PUT 'FOUND' IN R4
3097 015736 042704 000277  BIC #277,R4  ;CLEAR UNWANTED BITS
3098 015742 005005          CLR R5     ;PUT 'EXPECTED' IN R5
3099 015744 120504          CMPB R5,R4 ;IS ACTIVE CLEAR?
3100 015746 001401          BEQ 1$    ;BR IF YES
3101 015750 104040          HLT 40    ;ERROR ACTIVE IS NOT CLEAR
3102 015752 104400          1$: SCOPE ;SCOPE THIS TEST
3103
3104
3105
3106
3107
3108
3109
3110
3111
3112

```

```

:***** TEST 25 *****
:*DDCMP IN ACTIVE TEST
:*SET LU LOOP, SINGLE STEP 5 SYNCs AND A NON-SYNC (301)
:*VERIFY THAT IN ACTIVE IS SET
:*****

```

: TEST 25

```

3113 015754 012737 000025 001226 TST25: MOV #25,TSTNO
3114 015762 012737 016044 001216  MOV #TST26,NEXT
3115

```

;R1 CONTAINS BASE DMC11 ADDRESS


```
3116 015770 104412 MSTCLR ;MASTER CLEAR DMC11
3117 015772 012711 004000 MOV #BIT11,(R1) ;SET LU LOOP
3118 015776 012702 000012 MOV #12,R2 ;SAVE LU REG FOR TYPEOUT
3119 016002 004737 026720 JSR PC,SYNC ;SINGLE CLOCK 5 SYNC CHARACTERS
3120 016006 000005 5
3121 016010 104415 000064 DATACLK, 64
3122 016014 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3123 016016 021244 021244 ;PORT4 LU12
3124 016020 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
3125 016024 042704 000277 BIC #277,R4 ;CLEAR UNWANTED BITS
3126 016030 012705 000100 MOV #BIT6,R5 ;PUT 'EXPECTED' IN R5
3127 016034 120504 CMPB R5,R4 ;IS ACTIVE SET?
3128 016036 001401 BEQ 1$ ;BR IF YES
3129 016040 104040 HLT 40 ;ERROR ACTIVE IS NOT SET
3130 016042 104400 1$: SCOPE ;SCOPE THIS TEST
```

```
3131
3132
3133 :***** TEST 26 *****
3134 :*DDCMP IN ACTIVE TEST
3135 :*SET LU LOOP, SINGLE STEP 1 SYNC AND A NON-SYNC (301)
3136 :*VERIFY THAT IN ACTIVE DOES NOT SET
3137 :*****
```

```
3138
3139 : TEST 26
3140 :-----
3141 016044 012737 000026 001226 TST26: MOV #26,TSTNO
3142 016052 012737 016132 001216 MOV #TST27,NEXT
3143 ;R1 CONTAINS BASE DMC11 ADDRESS
3144 016060 104412 MSTCLR ;MASTER CLEAR DMC11
3145 016062 012711 004000 MOV #BIT11,(R1) ;SET LU LOOP
3146 016066 012702 000012 MOV #12,R2 ;SAVE LU REG FOR TYPEOUT
3147 016072 004737 026720 JSR PC,SYNC ;SINGLE CLOCK 1 SYNC CHARACTERS
3148 016076 000001 1
3149 016100 104415 000024 DATACLK, 24
3150 016104 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3151 016106 021244 021244 ;PORT4 LU12
3152 016110 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
3153 016114 042704 000277 BIC #277,R4 ;CLEAR UNWANTED BITS
3154 016120 005005 CLR R5 ;PUT 'EXPECTED' IN R5
3155 016122 120504 CMPB R5,R4 ;IS ACTIVE CLEAR?
3156 016124 001401 BEQ 1$ ;BR IF YES
3157 016126 104040 HLT 40 ;ERROR ACTIVE IS NOT CLEAR
3158 016130 104400 1$: SCOPE ;SCOPE THIS TEST
```

```
3159
3160
3161 :***** TEST 27 *****
3162 :*DDCMP IN ACTIVE TEST
3163 :*SET LU LOOP, SINGLE STEP 2 SYNC AND A NON-SYNC (301)
3164 :*VERIFY THAT IN ACTIVE IS SET
3165 :*****
```

```
3166
3167 : TEST 27
3168 :-----
3169 016132 012737 000027 001226 TST27: MOV #27,TSTNO
3170 016140 012737 016222 001216 MOV #TST30,NEXT
3171 ;R1 CONTAINS BASE DMC11 ADDRESS
```

```

3172 016146 104412 MSTCLR ;MASTER CLEAR DMC11
3173 016150 012711 004000 MOV #BIT11,(R1) ;SET LU LOOP
3174 016154 012702 000012 MOV #12,R2 ;SAVE LU REG FOR TYPEOUT
3175 016160 004737 026720 JSR PC,SYNC ;SINGLE CLOCK 2 SYNC CHARACTERS
3176 016164 000002 2
3177 016166 104415 000034 DATACLK, 34
3178 016172 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3179 016174 021244 021244 ;PORT4 LU12
3180 016176 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
3181 016202 042704 000277 BIC #277,R4 ;CLEAR UNWANTED BITS
3182 016206 012705 000100 MOV #BIT6,R5 ;PUT 'EXPECTED' IN R5
3183 016212 120504 CMPB R5,R4 ;IS ACTIVE SET?
3184 016214 001401 BEQ 1$ ;BR IF YES
3185 016216 104040 HLT 40 ;ERROR ACTIVE IS NOT SET
3186 016220 104400 1$: SCOPE ;SCOPE THIS TEST
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197

```

```

:***** TEST 30 *****
:*IN CLEAR TEST
:*SYNC UP RECEIVER AND TRANSMIT A CHARACTER
:*WAIT FOR IN RDY, THEN SET IN CLEAR
:*VERIFY THAT IN ACTIVE AND IN RDY ARE CLEARED
:*****

```

```

: TEST 30
:-----
3198 016222 012737 000030 001226 TST30: MOV #30,TSTNO
3199 016230 012737 016374 001216 MOV #TST31,NEXT
3200
3201 016236 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3202 016240 012702 000012 MOV #12,R2 ;MASTER CLEAR DMC11
3203 016244 012711 004000 MOV #BIT11,(R1) ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3204 016250 004737 027066 JSR PC,CHAR ;SET LINE UNIT LOOP
3205 016254 000301 301 ;LOAD SILO WITH 3 SYNC
3206 016256 104415 000053 DATACLK, 53 ;AND A NON-SYNC (301)
3207 016262 104416 000002 TIMER, 2 ;SINGLE CLOCK THE DATA
3208 016266 104414 ROMCLK ;WAIT FOR INRDY
3209 016270 021244 021244 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3210 016272 016104 000004 MOV 4(R1),R4 ;PORT4 LU 12
3211 016276 042704 000357 BIC #357,R4 ;PUT 'FOUND' IN R4
3212 016302 012705 000020 MOV #BIT4,R5 ;CLEAR UNWANTED BITS
3213 016306 120504 CMPB R5,R4 ;PUT 'EXPECTED' IN R5
3214 016310 001401 BEQ 1$ ;IS INRDY SET?
3215 016312 104040 HLT 40 ;ERROR, INRDY IS NOT SET
3216 016314
3217 016314 012761 000200 000004 1$: MOV #BIT7,4(R1) ;LOAD PORT4
3218 016322 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3219 016324 122112 122112 ;SET IN CLEAR
3220 016326 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3221 016330 021244 021244 ;PORT4 LU 12
3222 016332 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
3223 016336 042704 000277 BIC #277,R4 ;CLEAR UNWANTED BITS
3224 016342 005005 CLR R5 ;PUT 'EXPECTED' IN R5
3225 016344 120504 CMPB R5,R4 ;IS IN ACTIVE CLEAR?
3226 016346 001401 BEQ 2$
3227 016350 104040 HLT 40 ;ERROR, IN ACTIVE IS NOT CLEAR

```

BASIC RECEIVER TESTS

SEQ 0064

```
3228 016352          2$:
3229 016352 016104 000004      MOV    4(R1),R4      ;PUT 'FOUND' IN R4
3230 016356 042704 000357      BIC    #357,R4      ;CLEAR UNWANTED BITS
3231 016362 005005          CLR    R5           ;PUT 'EXPECTED' IN R5
3232 016364 120504          CMPB   R5,R4        ;IS INRDY CLEARED?
3233 016366 001401          BEQ    3$           ;
3234 016370 104040          HLT    40           ;ERROR, INRDY IS NOT CLEARED
3235 016372 104400          3$: SCOPE         ;SCOPE THIS TEST
3236
3237
3238          ;***** TEST 31 *****
3239          ;*DDCMP BASIC RECEICER TEST
3240          ;*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 0
3241          ;*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
3242          ;*****
3243
3244          ; TEST 31
3245          ;-----
3246 016374 012737 000031 001226 TST31: MOV    #31,TSTNO
3247 016402 012737 016510 001216      MOV    #TST32,NEXT
3248          ;R1 CONTAINS BASE DMC11 ADDRESS
3249 016410 104412          MSTCLR ;MASTER CLEAR DMC11
3250 016412 012702 000012      MOV    #12,R2      ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3251 016416 012711 004000      MOV    #BIT11,(R1) ;SET LINE UNIT LOOP
3252 016422 004737 027066      JSR    PC,CHAR     ;LOAD SILO WITH 3 SYNCs
3253 016426 000000          0              ;AND THE CHARACTER 0
3254 016430 104415 000053      DATACLK, 53    ;SINGLE CLOCK THE DATA
3255 016434 104416 000002      TIMER, 2        ;WAIT FOR INRDY
3256 016440 104414          ROMCLK         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3257 016442 021244          021244        ;PORT4 LU 12
3258 016444 016104 000004      MOV    4(R1),R4    ;PUT 'FOUND' IN R4
3259 016450 042704 000357      BIC    #357,R4    ;CLEAR UNWANTED BITS
3260 016454 012705 000020      MOV    #BIT4,R5   ;PUT 'EXPECTED' IN R5
3261 016460 120504          CMPB   R5,R4      ;IS INRDY SET?
3262 016462 001401          BEQ    1$         ;
3263 016464 104040          HLT    40         ;ERROR, INRDY IS NOT SET
3264 016466          1$:
3265 016466 104414          ROMCLK         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3266 016470 021204          021204        ;PORT4 IN DATA
3267 016472 016104 000004      MOV    4(R1),R4    ;PUT 'FOUND' IN R4
3268 016476 005005          CLR    R5         ;PUT 'EXPECTED' IN R5
3269 016500 120504          CMPB   R5,R4      ;WAS A 0 RECEIVED?
3270 016502 001401          BEQ    2$         ;
3271 016504 104010          HLT    10        ;ERROR, RECEIVED DATA IS WRONG
3272 016506 104400          2$: SCOPE         ;SCOPE THIS TEST
3273
3274
3275          ;***** TEST 32 *****
3276          ;*DDCMP BASIC RECEICER TEST
3277          ;*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 125
3278          ;*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
3279          ;*****
3280
3281          ; TEST 32
3282          ;-----
3283 016510 012737 000032 001226 TST32: MOV    #32,TSTNO
```

```

3284 016516 012737 016626 001216      MOV      #TST33,NEXT
3285                                     ;R1 CONTAINS BASE DMC11 ADDRESS
3286 016524 104412      MSTCLR   ;MASTER CLEAR DMC11
3287 016526 012702 000012      MOV      #12,R2      ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3288 016532 012711 004000      MOV      #BIT11,(R1) ;SET LINE UNIT LOOP
3289 016536 004737 027066      JSR      PC,CHAR     ;LOAD SILO WITH 3 SYNCs
3290 016542 000125      125          ;AND THE CHARACTER 125
3291 016544 104415 000053      DATACLK,    53     ;SINGLE CLOCK THE DATA
3292 016550 104416 000002      TIMER,      2      ;WAIT FOR INRDY
3293 016554 104414      ROMCLK     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3294 016556 021244      021244      ;PORT4 LU 12
3295 016560 016104 000004      MOV      4(R1),R4   ;PUT 'FOUND' IN R4
3296 016564 042704 000357      BIC      #357,R4    ;CLEAR UNWANTED BITS
3297 016570 012705 000020      MOV      #BIT4,R5   ;PUT 'EXPECTED' IN R5
3298 016574 120504      CMPB     R5,R4      ;IS INRDY SET?
3299 016576 001401      BEQ      1$
3300 016600 104040      HLT      40         ;ERROR, INRDY IS NOT SET
3301 016602                                     ;
3302 016602 104414      ROMCLK     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3303 016604 021204      021204      ;PORT4 IN DATA
3304 016606 016104 000004      MOV      4(R1),R4   ;PUT 'FOUND' IN R4
3305 016612 012705 000125      MOV      #125,R5    ;PUT 'EXPECTED' IN R5
3306 016616 120504      CMPB     R5,R4      ;WAS A 125 RECEIVED?
3307 016620 001401      BEQ      2$
3308 016622 104010      HLT      10         ;ERROR, RECEIVED DATA IS WRONG
3309 016624 104400      SCOPE     ;SCOPE THIS TEST
3310
3311
3312                                     ;***** TEST 33 *****
3313                                     ;*DDCMP BASIC RECEICER TEST
3314                                     ;*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 252
3315                                     ;*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
3316                                     ;*****
3317
3318                                     ; TEST 33
3319                                     ;-----
3320 016626 012737 000033 001226 TST33: MOV      #33,TSTNO
3321 016634 012737 016744 001216      MOV      #TST34,NEXT
3322                                     ;R1 CONTAINS BASE DMC11 ADDRESS
3323 016642 104412      MSTCLR   ;MASTER CLEAR DMC11
3324 016644 012702 000012      MOV      #12,R2      ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3325 016650 012711 004000      MOV      #BIT11,(R1) ;SET LINE UNIT LOOP
3326 016654 004737 027066      JSR      PC,CHAR     ;LOAD SILO WITH 3 SYNCs
3327 016660 000252      252          ;AND THE CHARACTER 252
3328 016662 104415 000053      DATACLK,    53     ;SINGLE CLOCK THE DATA
3329 016666 104416 000002      TIMER,      2      ;WAIT FOR INRDY
3330 016672 104414      ROMCLK     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3331 016674 021244      021244      ;PORT4 LU 12
3332 016676 016104 000004      MOV      4(R1),R4   ;PUT 'FOUND' IN R4
3333 016702 042704 000357      BIC      #357,R4    ;CLEAR UNWANTED BITS
3334 016706 012705 000020      MOV      #BIT4,R5   ;PUT 'EXPECTED' IN R5
3335 016712 120504      CMPB     R5,R4      ;IS INRDY SET?
3336 016714 001401      BEQ      1$
3337 016716 104040      HLT      40         ;ERROR, INRDY IS NOT SET
3338 016720                                     ;
3339 016720 104414      ROMCLK     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
  
```

```
3340 016722 021204          021204          :PORT4 IN DATA
3341 016724 016104 000004    MOV      4(R1),R4      :PUT 'FOUND' IN R4
3342 016730 012705 000252    MOV      #252,R5      :PUT 'EXPECTED' IN R5
3343 016734 120504          CMPB     R5,R4        :WAS A 252 RECEIVED?
3344 016736 001401          BEQ      2$           :
3345 016740 104010          HLT      10           :ERROR, RECEIVED DATA IS WRONG
3346 016742 104400          2$: SCOPE           :SCOPE THIS TEST
3347
3348
3349
```

```
:***** TEST 34 *****
:DDCMP BASIC RECEICER TEST
:SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 377
:VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
:*****
```

```
: TEST 34
```

```
3356
3357 016744 012737 000034 001226 TST34: MOV      #34,TSTNO
3358 016752 012737 017062 001216    MOV      #TST35,NEXT
3359
3360 016760 104412          MSTCLR          :R1 CONTAINS BASE DMC11 ADDRESS
3361 016762 012702 000012    MOV      #12,R2      :MASTER CLEAR DMC11
3362 016766 012711 004000    MOV      #BIT11,(R1) :SAVE REG ADDRESS IN R2 FOR TYPEOUT
3363 016772 004737 027066    JSR      PC,CHAR     :SET LINE UNIT LOOP
3364 016776 000377          377            :LOAD SILO WITH 3 SYNCs
3365 017000 104415 000053    DATACLK,      53  :AND THE CHARACTER 377
3366 017004 104416 000002    TIMER, 2        :SINGLE CLOCK THE DATA
3367 017010 104414          ROMCLK         :WAIT FOR INRDY
3368 017012 021244          021244        :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3369 017014 016104 000004    MOV      4(R1),R4      :PORT4 LU 12
3370 017020 042704 000357    BIC      #357,R4      :PUT 'FOUND' IN R4
3371 017024 012705 000020    MOV      #BIT4,R5     :CLEAR UNWANTED BITS
3372 017030 120504          CMPB     R5,R4        :PUT 'EXPECTED' IN R5
3373 017032 001401          BEQ      1$           :IS INRDY SET?
3374 017034 104040          HLT      40           :ERROR, INRDY IS NOT SET
3375 017036          1$:
3376 017036 104414          ROMCLK         :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3377 017040 021204          021204        :PORT4 IN DATA
3378 017042 016104 000004    MOV      4(R1),R4      :PUT 'FOUND' IN R4
3379 017046 012705 000377    MOV      #377,R5      :PUT 'EXPECTED' IN R5
3380 017052 120504          CMPB     R5,R4        :WAS A 377 RECEIVED?
3381 017054 001401          BEQ      2$           :
3382 017056 104010          HLT      10           :ERROR, RECEIVED DATA IS WRONG
3383 017060 104400          2$: SCOPE           :SCOPE THIS TEST
3384
```

```
:***** TEST 35 *****
:DDCMP DATA TEST
:THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
:CHECKING EACH CHARACTER AS IT IS RECEIVED
:*****
```

```
: TEST 35
```

```
3393
3394 017062 012737 000035 001226 TST35: MOV      #35,TSTNO
3395 017070 012737 017212 001216    MOV      #TST36,NEXT
```

```
3396                                     ;R1 CONTAINS BASE DMC11 ADDRESS
3397 017076 104412                       MSTCLR      ;MASTER CLEAR DMC11
3398 017100 005037 027504                CLR        SCHAR  ;START BINARY COUNT AT ZERO
3399 017104 005037 027506                CLR        STUFLG ;CLEAR BITSTUFF FLAG
3400 017110 005002                       CLR        R2      ;R2 IS 'EXPECTED' DATA
3401 017112 012703 000073                MOV        #73,R3  ;R3 IS CHARACTER COUNT
3402 017116 012711 004000                MOV        #BIT11,(R1) ;SET LINE UNIT LOOP
3403 017122 004737 027244                JSR        PC,SILOLD ;LOAD SILO WITH COUNT PATTERN
3404 017126 104415 000043                DATACLK, 43      ;SYNC RECEIVER AND GET IT ACTIVE
3405 017132 104415 000730                1$: DATACLK, 730 ;CLOCK IN 73 CHARACTERS
3406 017136 004737 027510                4$: JSR        PC,INRDY ;WAIT FOR INRDY
3407 017142 104414                       ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
3408 017144 021204 021204                021204     ;PORT4 IN DATA
3409 017146 016104 000004                MOV        4(R1),R4 ;PUT 'FOUND' IN R4
3410 017152 010205                       MOV        R2,R5   ;PUT 'EXPECTED' IN R5
3411 017154 120504                       CMPB       R5,R4   ;IS DATA CORRECT?
3412 017156 001401                       BEQ        2$     ;BR IF YES
3413 017160 104010                       HLT        10     ;DATA ERROR
3414 017162 005202 2$: INC        R2      ;NEXT CHARACTER
3415 017164 022702 000400                CMP        #400,R2 ;ALL DONE?
3416 017170 001407                       BEQ        3$     ;BR IF YES
3417 017172 005303                       DEC        R3     ;DECREMENT CHARACTER COUNT
3418 017174 001360                       BNE        4$     ;BR IF SILO NOT EMPTY
3419 017176 004737 027244                JSR        PC,SILOLD ;LOAD SILO WITH MORE OF COUNT PATTERN
3420 017202 012703 000073                MOV        #73,R3  ;RELOAD CHARACTER COUNT
3421 017206 000751                       BR         1$     ;CONTINUE
3422 017210 104400                3$: SCOPE      ;SCOPE THIS TEST
3423
3424
3425                                     ;***** TEST 36 *****
3426                                     ;*DDCMP DATA TEST
3427                                     ;*THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
3428                                     ;*CHECKING EACH CHARACTER AS IT IS RECEIVED
3429                                     ;*THIS TEST IS EXACTLY THE SAME AS THE LAST TEST,
3430                                     ;*EXCEPT LINE UNIT LOOP IS SET IN LU REGISTER 12
3431                                     ;*****
3432
3433                                     ; TEST 36
3434                                     ;-----
3435 017212 012737 000036 001226 TST36: MOV        #36,TSTNO
3436 017220 012737 017352 001216        MOV        #TST37,NEXT
3437
3438 017226 104412                       MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
3439 017230 005037 027504                CLR        SCHAR  ;MASTER CLEAR DMC11
3440 017234 005037 027506                CLR        STUFLG ;START BINARY COUNT AT ZERO
3441 017240 005002                       CLR        R2      ;CLEAR BITSTUFF FLAG
3442 017242 012703 000073                MOV        #73,R3  ;R2 IS 'EXPECTED' DATA
3443 017246 005011                       CLR        (R1)    ;R3 IS CHARACTER COUNT
3444 017250 012761 000040 000004        MOV        #BITS,4(R1) ;CLEAR LU LOOP IN MAINT REG
3445 017256 104414                       ROMCLK      ;LOAD PORT4
3446 017260 122112 122112                122112     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3447 017262 004737 027244                JSR        PC,SILOLD ;SET LU LOOP IN LU REG 12
3448 017266 104415 000043                DATACLK, 43      ;LOAD SILO WITH COUNT PATTERN
3449 017272 104415 000730                1$: DATACLK, 730 ;SYNC RECEIVER AND GET IT ACTIVE
3450 017276 004737 027510                4$: JSR        PC,INRDY ;CLOCK IN 73 CHARACTERS
3451 017302 104414                       ROMCLK      ;WAIT FOR INRDY
                                     ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
```

```

3452 017304 021204          021204          ;PORT4 IN DATA
3453 017306 016104 000004  MOV      4(R1),R4      ;PUT 'FOUND' IN R4
3454 017312 010205          MOV      R2,R5        ;PUT 'EXPECTED' IN R5
3455 017314 120504          CMPB    R5,R4        ;IS DATA CORRECT?
3456 017316 001401          BEQ     2$           ;BR IF YES
3457 017320 104010          HLT     10          ;DATA ERROR
3458 017322 005202          2$: INC     R2        ;NEXT CHARACTER
3459 017324 022702 000400  CMP     #400,R2     ;ALL DONE?
3460 017330 001407          BEQ     3$           ;BR IF YES
3461 017332 005303          DEC     R3          ;DECREMENT CHARACTER COUNT
3462 017334 001360          BNE     4$           ;BR IF SILO NOT EMPTY
3463 017336 004737 027244  JSR     PC,SILOLD   ;LOAD SILO WITH MORE OF COUNT PATTERN
3464 017342 012703 000073  MOV     #73,R3      ;RELOAD CHARACTER COUNT
3465 017346 000751          BR      1$           ;CONTINUE
3466 017350 104400          3$: SCOPE          ;SCOPE THIS TEST
3467
3468
3469
3470
3471
3472
3473
3474
3475
3476
3477
3478
3479 017352 012737 000037 001226 TST37: MOV     #37,TSTNO
3480 017360 012737 017512 001216  MOV     #TST40,NEXT
3481
3482 017366 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
3483 017370 012711 004000  MOV     #BIT11,(R1) ;MASTER CLEAR DMC11
3484 017374 004737 027066  JSR     PC,CHAR    ;SET LINE UNIT LOOP
3485 017400 000301          301           ;LOAD SILO WITH 3 SYNCs
3486 017402 104415 000073  DATACLK, 73    ;AND A 301
3487 017406 004737 027510  JSR     PC,INRDY   ;CLOCK THE 301 IN AND 20 EXTRA TICKS
3488 017412 104414          ROMCLK        ;WAIT FOR INRDY
3489 017414 021204          021204       ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3490 017416 016104 000004  MOV     4(R1),R4    ;PORT4 IN DATA
3491 017422 012705 000301  MOV     #301,R5     ;PUT 'FOUND' IN R4
3492 017426 120504          CMPB    R5,R4      ;PUT 'EXPECTED' IN R5
3493 017430 001401          BEQ     1$           ;WAS A 301 RECEIVED?
3494 017432 104010          HLT     10          ;ERROR FIRST CHARACTER INCORRECT
3495 017434 004737 027510  1$: JSR     PC,INRDY   ;WAIT FOR INRDY
3496 017440 104414          ROMCLK        ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3497 017442 021204          021204       ;PORT4 IN DATA
3498 017444 016104 000004  MOV     4(R1),R4    ;PUT 'FOUND' IN R4
3499 017450 012705 000377  MOV     #377,R5     ;PUT 'EXPECTED' IN R5
3500 017454 120504          CMPB    R5,R4      ;WAS A 377 RECEIVED?
3501 017456 001401          BEQ     2$           ;ERROR, 377 WAS NOT RECEIVED
3502 017460 104010          HLT     10          ;WAIT FOR INRDY
3503 017462 004737 027510  2$: JSR     PC,INRDY   ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
3504 017466 104414          ROMCLK        ;PORT4 IN DATA
3505 017470 021204          021204       ;PUT 'FOUND' IN R4
3506 017472 016104 000004  MOV     4(R1),R4    ;PUT 'EXPECTED' IN R5
3507 017476 012705 000377  MOV     #377,R5
  
```

```

3508 017502 120504          CMPB   R5,R4          ;WAS A 377 RECEIVED?
3509 017504 001401          BEQ    3$             ;
3510 017506 104010          HLT    10             ;ERROR, 377 WAS NOT RECEIVED
3511 017510 104400          3$:  SCOPE           ;SCOPE THIS TEST
3512
3513
3514          :***** TEST 40 *****
3515          :*CABLE TURNAROUND TEST
3516          :*CLEAR LINE UNIT LOOP, SET DTR
3517          :*VERIFY THAT RING ANDODEM READY ARE SET
3518          :*CLEAR DTR, VERIFY THAT MRDY IS CLEARED
3519          :*****
3520
3521          : TEST 40
3522          :-----
3523 017512 012737 000040 001226 TST40: MOV    #40,TSTNO
3524 017520 012737 017744 001216      MOV    #TST41,NEXT
3525          ;R1 CONTAINS BASE DMC11 ADDRESS
3526 017526 104412          MSTCLR ;MASTER CLEAR DMC11
3527 017530 032737 020000 001366      BIT    #BIT13,STAT1 ;IS LINE UNIT M8202?
3528 017536 001004          BNE    .+12          ;BR IF YES (DO TEST EVEN IF NO LOOP-BACK CONN)
3529 017540 032737 040000 001366      BIT    #BIT14,STAT1 ;IS TURNAROUND CONNECTOR ON?
3530 017546 001475          BEQ    2$             ;SKIP TEST IF NO
3531 017550 005011          CLR    (R1)          ;CLEAR LINE UNIT LOOP
3532 017552 012761 000100 000004      MOV    #100,4(R1)   ;LOAD PORT4
3533 017560 104414          ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3534 017562 122113          122113 ;SET DTR
3535 017564 104416 000002          TIMER, 2 ;WAIT
3536 017570 104414          ROMCLK ;NEXT WORD IS INSTRUCTION, FOMCLK PC=5304
3537 017572 021264          021264 ;PORT4 LU13
3538 017574 016104 000004          MOV    4(R1),R4     ; PUT FOUND IN R4      +---+NEW
3539 017600 042704 000023          BIC    #23,R4       ; CLEAR JUNK
3540 017604 012705 000310          MOV    #310,R5      ; GET EXPECTED.
3541 017610 032737 020000 001366      BIT    #BIT13,STAT1 ; IS LINE UNIT M8202?
3542 017616 001402          BEQ    .+6           ;
3543 017620 042705 000200          BIC    #BIT7,R5     ; NO RING ON M8202
3544
3545 017624 032737 000004 001372      BIT    #BIT2,STAT3  ; IS THIS V.35 MODEM?
3546 017632 001402          BEQ    3$             ; NO THEN GO AHEAD.
3547
3548 017634 042705 000200          3$:  BIC    #BIT7,R5     ; YES-NO RING ON V.35 MODEM
3549 017640
3550 017640 020504          CMP    R5,R4         ; ARE RING AND MODEM READY SET?
3551 017642 001403          BEQ    1$             ; WARNING! IF V.35 AND AUTO STARTED,
3552          ; YOU WILL GET THIS ERROR. YOU MUST
3553          ; MANUALL NASWER THE QUESTIONS FOR V.35.
3554 017644 120405          CMPB   R4,R5         ; ARE RING AND MRDY CLEAR?
3555 017646 001435          BEQ    2$             ;
3556          ; WARNING! YOU MAY GET THIS ERROR IF V.35
3557          ; AND AUTOSTART. YOU MUST MANNJALLY ANSWER
3558          ; ALL QUESTIONS IF V.35.
3559 017650 104011          HLT    11             ;ERROR, MRDY NOT SET
3560 017652 005061 000004          1$:  CLR    4(R1)      ;CLEAR PORT4
3561 017656 104414          ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
3562 017660 122113          122113 ;CLEAR DTR
3563 017662 104416 000002          TIMER, 2

```



```

3564 017666 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3565 017670 021264 021264 ;PORT4 LU13
3566 017672 016104 000004 MOV 4(R1),R4 ;PUR FOUND IN R4 ***NEW
3567 017676 042704 000023 BIC #23,R4 ;STRIP JUNK
3568 017702 005005 CLR R5 ;SET EXPECTED.
3569 017704 032737 020000 001366 BIT #BIT13,STAT1 ;IS THIS A M8202?
3570 017712 001402 BEQ .+6
3571 017714 052705 000010 BIS #BIT3,R5 ;YES THEN EXPECT MRDY SET.
3572 017720 032737 000004 001372 BIT #BIT2,STAT3 ;IS THIS V.35?
3573 017726 001402 BEQ 4$
3574 017730 042704 000200 BIC #BIT7,R4
3575 017734 4$:
3576 017734 120405 CMPB R4,R5 ;ARE RING AND MRDY CLEAR?
3577 017736 001401 BEQ 2$
3578 ;WARNING! YOU MAY GET THIS ERROR IF V.35
3579 ;AND AUTOSTART. YOU MUST MANNUALLY ANSWER
3580 ;ALL QUESTIONS IF V.35.
3581 017740 104011 HLT 11 ;ERROR, MRDY NOT CLEAR
3582 017742 104400 2$: SCOPE ;SCOPE THIS TEST
3583
3584
3585 :***** TEST 41 *****
3586 :*CABLE TURNAROUND TEST
3587 :*CLEAR LINE UNIT LOOP, LOAD OUT DATA SILO
3588 :*VERIFY THAT ALL MODEM SIGNALS ARE SET
3589 :*****
3590
3591 : TEST 41
3592 :-----
3593 017744 012737 000041 001226 TST41: MOV #41,TSTNO
3594 017752 012737 020140 001216 MOV #TST42,NEXT
3595 ;R1 CONTAINS BASE DMC11 ADDRESS
3596 017760 104412 MSTCLR ;MASTER CLEAR DMC11
3597 017762 032737 020000 001366 BIT #BIT13,STAT1 ;IS LINE UNIT M8202?
3598 017770 001004 BNE .+12 ;BR IF YES (DO TEST EVEN IF NO LOOP-BACK CONN)
3599 017772 032737 040000 001366 BIT #BIT14,STAT1 ;IS TURNAROUND CONNECTOR ON?
3600 020000 001456 BEQ 1$ ;SKIP TEST IF NO
3601 020002 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
3602 020006 012761 000100 000004 MOV #100, 4(R1) ;LOAD PORT4
3603 020014 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3604 020016 122113 122113 ;CLEAR ALL MODEM SIGNALS,EXCEPT DTR
3605 020020 104416 000002 TIMER, 2 ;WAIT
3606 020024 012761 000001 000004 MOV #1,4(R1) ;LOAD PORT4
3607 020032 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3608 020034 122111 122111 ;SET SOM
3609 020036 004537 030152 JSR R5,MESLD ;FILL OUT DATA SILO
3610 020042 030434 MESDAT ;WITH 64 CHARACTERS
3611 020044 000107 64.
3612 020046 012700 000050 MOV #50,R0 ;PREPARE FOR DELAY
3613 020052 005011 CLR (R1) ;CLEAR LINE UNIT LOOP
3614 020054 2$:
3615 020054 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3616 020056 021264 021264 ;PORT4 LU13
3617 020060 016104 000004 MOV 4(R1),R4 ;PUT "FOUND" IN R4
3618 020064 042704 000023 BIC #23,R4 ;CLEAR UNWANTED BITS
3619 020070 012705 000354 MOV #354,R5 ;PUT "EXPECTED" IN R5

```

```

3620 020074 032737 000004 001372 BIT #BIT2,STAT3 ; IS THIS V.35?
3621 020102 001402 BEQ 40$ ;
3622 020104 042705 000200 BIC #BIT7,R5 ; YES THEN GET RID OF RING
3623 020110 032737 020000 001366 40$: BIT #BIT13,STAT1 ; IS THIS MB202?
3624 020116 001402 BEQ 4$ ;
3625 020120 042705 000200 BIC #BIT7,R5 ;
3626 020124 4$: CMPB R5,R4 ;COMPARE EXPECTED AND FOUND
3627 020124 120504 BFO 1$ ;BR IF OK
3628 020126 001403 DEC R0 ;DEC DELAY COUNT
3629 020130 005300 BNE 2$ ;BR IF NOT ZERO
3630 020132 001350 HLT 11 ;ERROR, ALL SIGNALS ARE NOT SET
3631 020134 104011 ; WARNING THIS TEST WILL FAIL IF YOU
3632 ; DON'T MANUALLY ANSWER QUESTIONS IF
3633 ; YES HAVE V.35.
3634 ; SCOPE THIS TEST
3635 020136 104400 1$: SCOPE
3636
3637
3638 ;***** TEST 42 *****
3639 ;*TEST OF CRC OPERATION
3640 ;*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
3641 ;*0, VERIFY THE LSB OF THE BCC ON EACH SHIFT
3642 ;*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
3643 ;*****
3644
3645 ; TEST 42
3646 ;-----
3647 020140 012737 000042 001226 TST42: MOV #42,TSTNO
3648 020146 012737 020454 001216 MOV #TST43,NEXT
3649 020154 012737 020170 001220 MOV #64$,LOCK
3650
3651 020162 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3652 020164 012711 004000 MOV #BIT11,(R1) ;MASTER CLEAR DMC11
3653 020170 004737 030214 64$: JSR PC,CLRIO ;SET LU LOOP
3654 020174 005000 CLR R0 ;CLEAR BCC REGISTERS
3655 020176 012737 120001 027650 MOV #CRC16,XPOLY ;START SHIFT COUNTER AT ZERO
3656 020204 012737 000000 020244 MOV #0,66$ ;LOAD POLYNOMIAL FOR SOFTWARE BCC
3657 020212 005037 020246 CLR 67$ ;LOAD CHAR FOR SOFTWARE BCC
3658 020216 004737 027654 JSR PC,BCCLD ;CLEAR OLD SOFTWARE BCC
3659 020222 000000 0 ;LOAD OUT SILO WITH 2 SYNCs
3660 020224 104415 000021 DATACLK, 21 ;AND THE CHARACTER 0
3661 020230 104415 000001 65$: DATACLK, 1 ;GET TRANSMITTER ACTIVE
3662 020234 005200 INC R0 ;SHIFT BCC ONCE
3663 020236 004537 027544 JSR R5,SIMBCC ;BUMP SHIFT COUNT
3664 020242 000001 1 ;CALCULATE SOFTWARE BCC LSB
3665 020244 000000 66$: 0 ;ONE SHIFT
3666 020246 000000 67$: 0 ;DATA CHARACTER
3667 020250 103405 BCS 68$ ;OLD BCC
3668 020252 004737 027766 JSR PC,GETQ0 ;BR IF SOFT BCC LSB IS SET
3669 020256 103006 BCC 69$ ;GET HARDWARE TRANSMITTER BCC LSB
3670 020260 104012 HLT 12 ;BR IF HARD BCC LSB IS CLEAR
3671 020262 000404 BR 69$ ;ERROR, BCC LSB IS SET
3672 020264 004737 027766 68$: JSR PC,GETQ0 ;CONTINUE
3673 020270 103401 BCS 69$ ;GET HARDWARE TRANSMITTER BCC LSB
3674 020272 104016 HLT 16 ;BR IF HARD BCC LSB IS SET
3675 020274 69$: ;ERROR, HARD BCC LSB IS CLEAR

```

3676	020274	006037	020244			ROR	66\$;SHIFT SOFT DATA
3677	020300	013737	027652	020246		MOV	CALBCC,67\$;LOAD OLD SOFT BCC
3678	020306	022700	000010			CMP	#10,RO		;DONE YET?
3679	020312	001346				BNE	65\$;BR IF NOT DONE
3680	020314	104401				SCOPI			;SCOPE SUBTEST (SW09=1)
3681	020316	012737	020324	001220		MOV	#71\$,LOCK		;NEW SCOPE1
3682	020324	004737	030214		71\$:	JSR	PC,CLRIO		;CLEAR BCC REGISTERS
3683	020330	005000				CLR	RO		;START SHIFT COUNTER AT ZERO
3684	020332	012737	120001	027650		MOV	#CRC16,XPOLY		;LOAD POLYNOMIAL FOR SOFTWARE BCC
3685	020340	012737	000000	020400		MOV	#0,73\$;LOAD CHAR FOR SOFTWARE BCC
3686	020346	005037	020402			CLR	74\$;CLEAR OLD SOFTWARE BCC
3687	020352	004737	027654			JSR	PC,BCCLD		;LOAD OUT SILO WITH 2 SYNCs
3688	020356	000000				0			;AND THE CHARACTER 0
3689	020360	104415	000032			DATACLK,	32		;GET RECEIVER ACTIVE
3690	020364	104415	000001		72\$:	DATACLK,	1		;SHIFT BCC ONCE
3691	020370	005200				INC	RO		;BUMP SHIFT COUNT
3692	020372	004537	027544			JSR	R5,SIMBCC		;CALCULATE SOFTWARE BCC LSB
3693	020376	000001				1			;ONE SHIFT
3694	020400	000000			73\$:	0			;DATA CHARACTER
3695	020402	000000			74\$:	0			;OLD BCC
3696	020404	103405				BCS	75\$;BR IF SOFT BCC LSB IS SET
3697	020406	004737	030000			JSR	PC,GETQI		;GET HARDWARE RECEIVER BCC LSB
3698	020412	103006				BCC	76\$;BR IF HARD BCC LSB IS CLEAR
3699	020414	104013				HLT	13		;ERROR, BCC LSB IS SET
3700	020416	000404				BR	76\$;CONTINUE
3701	020420	004737	030000		75\$:	JSR	PC,GETQI		;GET HARDWARE RECEIVER BCC LSB
3702	020424	103401				BCS	76\$;BR IF HARD BCC LSB IS SET
3703	020426	104017				HLT	17		;ERROR, BCC LSB IS CLEAR
3704	020430				76\$:				
3705	020430	006037	020400			ROR	73\$;SHIFT SOFT DATA
3706	020434	013737	027652	020402		MOV	CALBCC,74\$;LOAD OLD SOFT BCC
3707	020442	022700	000010			CMP	#10,RO		;DONE YET?
3708	020446	001346				BNE	72\$;BR IF NOT DONE
3709	020450	104401				SCOPI			;SCOPE SUBTEST (SW09=1)
3710	020452	104400			77\$:	SCOPE			;SCOPE THIS TEST

```

:***** TEST 43 *****
:*TEST OF CRC OPERATION
:*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
:*377, VERIFY THE LSB OF THE BCC ON EACH SHIFT
:*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
:*****

```

: TEST 43

3721									
3722	020454	012737	000043	001226	TST43:	MOV	#43,TSTNO		
3723	020462	012737	020770	001216		MOV	#TST44,NEXT		
3724	020470	012737	020504	001220		MOV	#64\$,LOCK		
3725									
3726	020476	104412				MSTCLR			;R1 CONTAINS BASE DMC11 ADDRESS
3727	020500	012711	004000			MOV	#BIT11,(R1)		;MASTER CLEAR DMC11
3728	020504	004737	030214		64\$:	JSR	PC,CLRIO		;SET LU LOOP
3729	020510	005000				CLR	RO		;CLEAR BCC REGISTERS
3730	020512	012737	120001	027650		MOV	#CRC16,XPOLY		;START SHIFT COUNTER AT ZERO
3731	020520	012737	000377	020560		MOV	#377,66\$;		;LOAD POLYNOMIAL FOR SOFTWARE BCC


```

3788 :***** TEST 44 *****
3789 :*TEST OF CRC OPERATION
3790 :*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
3791 :*125, VERIFY THE LSB OF THE BCC ON EACH SHIFT
3792 :*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
3793 :*****
3794
3795 : TEST 44
3796 :-----
3797 020770 012737 000044 001226 TST44: MOV #44,TSTNO ;R1 CONTAINS BASE DMC11 ADDRESS
3798 020776 012737 021304 001216 MOV #TST45,NEXT ;MASTER CLEAR DMC11
3799 021004 012737 021020 001220 MOV #64$,LOCK ;SET LU LOOP
3800
3801 021012 104412 MSTCLR ;CLEAR BCC REGISTERS
3802 021014 012711 004000 MOV #BIT11,(R1) ;START SHIFT COUNTER AT ZERO
3803 021020 004737 030214 64$: JSR PC,CLRIO ;LOAD POLYNOMIAL FOR SOFTWARE BCC
3804 021024 005000 CLR R0 ;LOAD CHAR FOR SOFTWARE BCC
3805 021026 012737 120001 027650 MOV #CRC16,XPOLY ;CLEAR OLD SOFTWARE BCC
3806 021034 012737 000125 021074 MOV #125,66$; ;LOAD OUT SILO WITH 2 SYNCs
3807 021042 005037 021076 CLR 67$ ;AND THE CHARACTER 125
3808 021046 004737 027654 JSR PC,BCCLD ;GET TRANSMITTER ACTIVE
3809 021052 000125 125 ;SHIFT BCC ONCE
3810 021054 104415 000021 DATACLK, 21 ;BUMP SHIFT COUNT
3811 021060 104415 000001 65$: DATACLK, 1 ;CALCULATE SOFTWARE BCC LSB
3812 021064 005200 INC R0 ;ONE SHIFT
3813 021066 004537 027544 JSR R5,SIMBCC ;DATA CHARACTER
3814 021072 000001 1 ;OLD BCC
3815 021074 000000 66$: 0 ;BR IF SOFT BCC LSB IS SET
3816 021076 000000 67$: 0 ;GET HARDWARE TRANSMITTER BCC LSB
3817 021100 103405 BCS 68$ ;BR IF HARD BCC LSB IS CLEAR
3818 021102 004737 027766 JSR PC,GETQ0 ;CONTINUE
3819 021106 103006 BCC 69$ ;GET HARDWARE TRANSMITTER BCC LSB
3820 021110 104012 HLT 12 ;BR IF HARD BCC LSB IS SET
3821 021112 000404 BR 69$ ;ERROR, BCC LSB IS SET
3822 021114 004737 027766 68$: JSR PC,GETQ0 ;CONTINUE
3823 021120 103401 BCS 69$ ;GET HARDWARE TRANSMITTER BCC LSB
3824 021122 104016 HLT 16 ;BR IF HARD BCC LSB IS SET
3825 021124 69$: ;ERROR, HARD BCC LSB IS CLEAR
3826 021124 006037 021074 ROR 66$ ;SHIFT SOFT DATA
3827 021130 013737 027652 021076 MOV CALBCC,67$ ;LOAD OLD SOFT BCC
3828 021136 022700 000010 CMP #10,R0 ;DONE YET?
3829 021142 001346 BNE 65$ ;BR IF NOT DONE
3830 021144 104401 SCOP1 ;SCOPE SUBTEST (SW09=1)
3831 021146 012737 021154 001220 MOV #71$,LOCK ;NEW SCOPE1
3832 021154 004737 030214 71$: JSR PC,CLRIO ;CLEAR BCC REGISTERS
3833 021160 005000 CLR R0 ;START SHIFT COUNTER AT ZERO
3834 021162 012737 120001 027650 MOV #CRC16,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE BCC
3835 021170 012737 000125 021230 MOV #125,73$; ;LOAD CHAR FOR SOFTWARE BCC
3836 021176 005037 021232 CLR 74$ ;CLEAR OLD SOFTWARE BCC
3837 021202 004737 027654 JSR PC,BCCLD ;LOAD OUT SILO WITH 2 SYNCs
3838 021206 000125 125 ;AND THE CHARACTER 125
3839 021210 104415 000032 DATACLK, 32 ;GET RECEIVER ACTIVE
3840 021214 104415 000001 72$: DATACLK, 1 ;SHIFT BCC ONCE
3841 021220 005200 INC R0 ;BUMP SHIFT COUNT
3842 021222 004537 027544 JSR R5,SIMBCC ;CALCULATE SOFTWARE BCC LSB
3843 021226 000001 1 ;ONE SHIFT

```

```

3844 021230 000000          73$: 0          ;DATA CHARACTER
3845 021232 000000          74$: 0          ;OLD BCC
3846 021234 103405          ;BCS 75$       ;BR IF SOFT BCC LSB IS SET
3847 021236 004737 030000  ;JSR PC,GETQI  ;GET HARDWARE RECEIVER BCC LSB
3848 021242 103006          ;BCC 76$       ;BR IF HARD BCC LSB IS CLEAR
3849 021244 104013          ;HLT 13        ;ERROR, BCC LSB IS SET
3850 021246 000404          ;BR 76$        ;CONTINUE
3851 021250 004737 030000  75$: JSR PC,GETQI  ;GET HARDWARE RECEIVER BCC LSB
3852 021254 103401          ;BCS 76$       ;BR IF HARD BCC LSB IS SET
3853 021256 104017          ;HLT 17        ;ERROR, BCC LSB IS CLEAR
3854 021260
3855 021260 006037 021230  76$: ROR 73$      ;SHIFT SOFT DATA
3856 021264 013737 027652 021232 ;MOV CALBCC,74$ ;LOAD OLD SOFT BCC
3857 021272 022700 000010 ;CMP #10,RO    ;DONE YET?
3858 021276 001346          ;BNE 72$       ;BR IF NOT DONE
3859 021300 104401          ;SCOPE1        ;SCOPE SUBTEST (SW09=1)
3860 021302 104400          77$: SCOPE      ;SCOPE THIS TEST
3861
3862
3863 ;***** TEST 45 *****
3864 ;*TEST OF CRC OPERATION
3865 ;*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
3866 ;*252, VERIFY THE LSB OF THE BCC ON EACH SHIFT
3867 ;*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
3868 ;*****
3869
3870 ; TEST 45
3871 ;-----
3872 021304 012737 000045 001226 TST45: MOV #45,TSTNO
3873 021312 012737 021620 001216 MOV #TST46,NEXT
3874 021320 012737 021334 001220 MOV #64$,LOCK
3875
3876 021326 104412          ;R1 CONTAINS BASE DMC11 ADDRESS
3877 021330 012711 004000          ;MSTCLR        ;MASTER CLEAR DMC11
3878 021334 004737 030214          ;MOV #BIT11,(R1) ;SET LU LOOP
3879 021340 005000          64$: JSR PC,CLRIO  ;CLEAR BCC REGISTERS
3880 021342 012737 120001 027650 ;CLR RO        ;START SHIFT COUNTER AT ZERO
3881 021350 012737 000252 021410 ;MOV #CRC16,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE BCC
3882 021356 005037 021412          ;MOV #252,66$  ;LOAD CHAR FOR SOFTWARE BCC
3883 021362 004737 027654          ;CLR 67$       ;CLEAR OLD SOFTWARE BCC
3884 021366 000252          ;JSR PC,BCCLD  ;LOAD OUT SILO WITH 2 SYNCs
3885 021370 104415 000021          252          ;AND THE CHARACTER 252
3886 021374 104415 000001          65$: DATACLK, 21  ;GET TRANSMITTER ACTIVE
3887 021400 005200          ;DATACLK, 1    ;SHIFT BCC ONCE
3888 021402 004537 027544          ;INC RO        ;BUMP SHIFT COUNT
3889 021406 000001          ;JSR R5,SIMBCC ;CALCULATE SOFTWARE BCC LSB
3890 021410 000000          66$: 1          ;ONE SHIFT
3891 021412 000000          67$: 0          ;DATA CHARACTER
3892 021414 103405          ;BCS 68$       ;BR IF SOFT BCC LSB IS SET
3893 021416 004737 027766          ;JSR PC,GETQO  ;GET HARDWARE TRANSMITTER BCC LSB
3894 021422 103006          ;BCC 69$       ;BR IF HARD BCC LSB IS CLEAR
3895 021424 104012          ;HLT 12        ;EPROR, BCC LSB IS SET
3896 021426 000404          ;BR 69$        ;CONTINUE
3897 021430 004737 027766          68$: JSR PC,GETQO  ;GET HARDWARE TRANSMITTER BCC LSB
3898 021434 103401          ;BCS 69$       ;BR IF HARD BCC LSB IS SET
3899 021436 104016          ;HLT 16        ;ERROR, HARD BCC LSB IS CLEAR

```

3900	021440				69\$:					
3901	021440	006037	021410			ROR	66\$:SHIFT SOFT DATA	
3902	021444	013737	027652	021412		MOV	CALBCC,67\$:LOAD OLD SOFT BCC	
3903	021452	022700	000010			CMP	#10,R0		:DONE YET?	
3904	021456	001346				BNE	65\$:BR IF NOT DONE	
3905	021460	104401				SCOPE1			:SCOPE SUBTEST (SW09=1)	
3906	021462	012737	021470	001220		MOV	#71\$,LOCK		:NEW SCOPE1	
3907	021470	004737	030214		71\$:	JSR	PC,CLRIO		:CLEAR BCC REGISTERS	
3908	021474	005000				CLR	R0		:START SHIFT COUNTER AT ZERO	
3909	021476	012737	120001	027650		MOV	#CRC16,XPOLY		:LOAD POLYNOMIAL FOR SOFTWARE BCC	
3910	021504	012737	000252	021544		MOV	#252,73\$;		:LOAD CHAR FOR SOFTWARE BCC	
3911	021512	005037	021546			CLR	74\$:CLEAR OLD SOFTWARE BCC	
3912	021516	004737	027654			JSR	PC,BCCLD		:LOAD OUT SILO WITH 2 SYNCs	
3913	021522	000252				252			:AND THE CHARACTER 252	
3914	021524	104415	000032			DATACLK,	32		:GET RECEIVER ACTIVE	
3915	021530	104415	000001		72\$:	DATACLK,	1		:SHIFT BCC ONCE	
3916	021534	005200				INC	R0		:BUMP SHIFT COUNT	
3917	021536	004537	027544			JSR	R5,SIMBCC		:CALCULATE SOFTWARE BCC LSB	
3918	021542	000001				1			:ONE SHIFT	
3919	021544	000000			73\$:	0			:DATA CHARACTER	
3920	021546	000000			74\$:	0			:OLD BCC	
3921	021550	103405				BCS	75\$:BR IF SOFT BCC LSB IS SET	
3922	021552	004737	030000			JSR	PC,GETQ1		:GET HARDWARE RECEIVER BCC LSB	
3923	021556	103006				BCC	76\$:BR IF HARD BCC LSB IS CLEAR	
3924	021560	104013				HLT	13		:ERROR, BCC LSB IS SET	
3925	021562	000404				BR	76\$:CONTINUE	
3926	021564	004737	030000		75\$:	JSR	PC,GETQ1		:GET HARDWARE RECEIVER BCC LSB	
3927	021570	103401				BCS	76\$:BR IF HARD BCC LSB IS SET	
3928	021572	104017				HLT	17		:ERROR, BCC LSB IS CLEAR	
3929	021574				76\$:					
3930	021574	006037	021544			ROR	73\$:SHIFT SOFT DATA	
3931	021600	013737	027652	021546		MOV	CALBCC,74\$:LOAD OLD SOFT BCC	
3932	021606	022700	000010			CMP	#10,R0		:DONE YET?	
3933	021612	001346				BNE	72\$:BR IF NOT DONE	
3934	021614	104401				SCOPE1			:SCOPE SUBTEST (SW09=1)	
3935	021616	104400			77\$:	SCOPE			:SCOPE THIS TEST	
3936										
3937										
3938										
3939										
3940										
3941										
3942										
3943										
3944										
3945										
3946	021620	012737	000046	001226	TST46:	MOV	#46,TSTNO			
3947	021626	012737	022056	001216		MOV	#TST47,NEXT			
3948										
3949	021634	104412				MSTCLR			:R1 CONTAINS BASE DMC11 ADDRESS	
3950	021636	012711	004000			MOV	#BIT11,(R1)		:MASTER CLEAR DMC11	
3951	021642	005003				CLR	R3		:SET LINE UNIT LOOP	
3952	021644	005004				CLR	R4		:ZERO BIT COUNT	
3953	021646	005005				CLR	R5		:R4 CONTAINS CHAR TO BE LOADED IN SILO	
3954	021650	005037	021752			CLR	4\$:R5 CONTAINS CHAR CURRENTLY BEING SHIFTED OUT	
3955	021654	012737	120001	027650		MOV	#CRC16,XPOLY		:CLEAR SOFT BCC	
									:LOAD POLYNOMIAL	

***** TEST 46 *****
 ;*TRANSMITTER CRC TEST
 ;*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK A BINARY
 ;*COUNT PATTERN, VERIFY THE LSB OF THE TRANSMITTER BCC ON EACH SHIFT
 ;*****

; TEST 46

 ;R1 CONTAINS BASE DMC11 ADDRESS
 ;MASTER CLEAR DMC11
 ;SET LINE UNIT LOOP
 ;ZERO BIT COUNT
 ;R4 CONTAINS CHAR TO BE LOADED IN SILO
 ;R5 CONTAINS CHAR CURRENTLY BEING SHIFTED OUT
 ;CLEAR SOFT BCC
 ;LOAD POLYNOMIAL

3956	021662	004737	030016		JSR	PC,SYNLD		:LOAD SILO WITH 2 SYNCs, SOM SET
3957	021666	010461	000004		MOV	R4,4(R1)		:PORT4 CHAR
3958	021672	104414			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3959	021674	122110			122110			:LOAD OUT DATA
3960	021676	005204			INC	R4		:INCREMENT TO NEXT CHARACTER
3961	021700	010461	000004		MOV	R4,4(R1)		:PORT4 CHAR
3962	021704	104414			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3963	021706	122110			122110			:LOAD OUT DATA
3964	021710	005204			INC	R4		:INCREMENT TO NEXT CHARACTER
3965	021712	010461	000004		MOV	R4,4(R1)		:PORT4 CHAR
3966	021716	104414			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3967	021720	122110			122110			:LOAD OUT DATA
3968	021722	004737	026702		JSR	PC,GCOR		:WAIT FOR OCOR
3969	021726	104415	000021		DATACLK,21			:CLOCK DATA
3970	021732	010537	021750	1\$:	MOV	R5,3\$:LOAD CHAR FOR SOFT CRC
3971	021736	104415	000001	2\$:	DATACLK,1			:SHIFT BCC ONCE
3972	021742	004537	027544		JSR	R5,SIMBCC		:CALCULATE SOFT BCC
3973	021746	000001			1			:SOFT SHIFT COUNT
3974	021750	000000		3\$:	0			:SOFT CHARACTER
3975	021752	000000		4\$:	0			:OLD SOFT BCC
3976	021754	103405			BCS	5\$:BR IF SOFT BCC LSB IS SET
3977	021756	004737	027766		JSR	PC,GETQO		:GET HARDWARE TRANSMITTER BCC LSB
3978	021762	103006			BCC	6\$:BR IF OK (CLEARED)
3979	021764	104020			HLT	20		:ERROR, BCC LSB WAS SET
3980	021766	000404			BR	6\$:CONTINUE WITH TEST
3981	021770	004737	027766	5\$:	JSR	PC,GETQO		:GET HARDWARE TRANSMITTER BCC LSB
3982	021774	103401			BCS	6\$:BR IF OK (SET)
3983	021776	104021			HLT	2'		:ERROR, BCC LSB WAS CLEAR
3984								
3985	022000			6\$:				
3986	022000	006037	021750		ROR	3\$:SHIFT SOFT DATA
3987	022004	013737	027652	021752	MOV	CALBCC,4\$:LOAD OLD SOFT BCC
3988	022012	005203			INC	R3		:INCREMENT BIT COUNTER
3989	022014	022703	000010		CMP	#10,R3		:DONE A FULL CHARACTER YET?
3990	022020	001346			BNE	2\$:BR IF NO
3991	022022	005003			CLR	R3		:RESTART BIT COUNTER
3992	022024	005204			INC	R4		:INCREMENT DATA FOR SILO
3993	022026	022704	000400		CMP	#400,R4		:DONE BINARY COUNT YET?
3994	022032	003404			BLE	9\$:BR IF YES
3995	022034	010461	000004		MOV	R4,4(R1)		:PORT4 DATA
3996	022040	104414			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
3997	022042	122110			122110			:LOAD OUT DATA
3998	022044	005205		9\$:	INC	R5		:INCREMENT DATA
3999	022046	022705	000400		CMP	#400,R5		:DONE BINARY PATTERN YET?
4000	022052	001327			BNE	1\$:BR IF NO
4001	022054	104400		7\$:	SCOPE			:SCOPE THIS TEST

4002
 4003
 4004
 4005
 4006
 4007
 4008
 4009
 4010
 4011

```

:***** TEST 47 *****
:*RECEIVER CRC TEST
:*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK A BINARY
:*COUNT PATTERN, VERIFY THE LSB OF THE RECEIVER BCC ON EACH SHIFT
:*****
:
: TEST 47
:-----

```


4012	022056	012737	000047	001226	TST47:	MOV	#47,TSTNO		
4013	022064	012737	022314	001216		MOV	#TST50,NEXT		
4014									:R1 CONTAINS BASE DMC11 ADDRESS
4015	022072	104412				MSTCLR			:MASTER CLEAR DMC11
4016	022074	012711	004000			MOV	#BIT11,(R1)		:SET LINE UNIT LOOP
4017	022100	005003				CLR	R3		:ZERO BIT COUNT
4018	022102	005004				CLR	R4		:R4 CONTAINS CHAR TO BE LOADED IN SILO
4019	022104	005005				CLR	R5		:R5 CONTAINS CHAR CURRENTLY BEING SHIFTED OUT
4020	022106	005037	022210			CLR	4\$:CLEAR SOFT BCC
4021	022112	012737	120001	027650		MOV	#CRC16,XPOLY		:LOAD POLYNOMIAL
4022	022120	004737	030016			JSR	PC,SYNLD		:LOAD SILO WITH 2 SYNCs, SOM SET
4023	022124	010461	000004			MOV	R4,4(R1)		:PORT4 CHAR
4024	022130	104414				ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4025	022132	122110				122110			:LOAD OUT DATA
4026	022134	005204				INC	R4		:INCREMENT TO NEXT CHARACTER
4027	022136	010461	000004			MOV	R4,4(R1)		:PORT4 CHAR
4028	022142	104414				ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4029	022144	122110				122110			:LOAD OUT DATA
4030	022146	005204				INC	R4		:INCREMENT TO NEXT CHARACTER
4031	022150	010461	000004			MOV	R4,4(R1)		:PORT4 CHAR
4032	022154	104414				ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4033	022156	122110				122110			:LOAD OUT DATA
4034	022160	004737	026702			JSR	PC,OCOR		:WAIT FOR OCOR
4035	022164	104415	000032			DATACLK,	32		:CLOCK DATA
4036	022170	010537	022206	1\$:		MOV	R5,3\$:LOAD CHAR FOR SOFT CRC
4037	022174	104415	000001	2\$:		DATACLK,	1		:SHIFT BCC ONCE
4038	022200	004537	027544			JSR	R5,SIMBCC		:CALCULATE SOFT BCC
4039	022204	000001				1			:SOFT SHIFT COUNT
4040	022206	000000		3\$:		0			:SOFT CHARACTER
4041	022210	000000		4\$:		0			:OLD SOFT BCC
4042	022212	103405				BCS	5\$:BR IF SOFT BCC LSB IS SET
4043	022214	004737	030000			JSR	PC,GETQI		:GET HARDWARE RECEIVER BCC LSB
4044	022220	103006				BCC	6\$:BR IF OK (CLEARED)
4045	022222	104022				HLT	22		:ERROR, BCC LSB WAS SET
4046	022224	000404				BR	6\$:CONTINUE WITH TEST
4047	022226	004737	030000	5\$:		JSR	PC,GETQI		:GET HARDWARE RECEIVER BCC LSB
4048	022232	103401				BCS	6\$:BR IF OK (SET)
4049	022234	104023				HLT	23		:ERROR, BCC LSB WAS CLEAR
4050									
4051	022236			6\$:					
4052	022236	006037	022206			ROR	3\$:SHIFT SOFT DATA
4053	022242	013737	027652	022210		MO	CALBCC,4\$:LOAD OLD SOFT BCC
4054	022250	005203				INC	R3		:INCREMENT BIT COUNTER
4055	022252	022703	000010			CMP	#10,R3		:DONE A FULL CHARACTER YET?
4056	022256	001346				BNE	2\$:BR IF NO
4057	022260	005003				CLR	R3		:RESTART BIT COUNTER
4058	022262	005204				INC	R4		:INCREMENT DATA FOR SILO
4059	022264	022704	000400			CMP	#400,R4		:DONE BINARY COUNT YET?
4060	022270	003404				BLE	9\$:BR IF YES
4061	022272	010461	000004			MOV	R4,4(R1)		:PORT4 DATA
4062	022276	104414				ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4063	022300	122110				122110			:LOAD OUT DATA
4064	022302	005205			9\$:	INC	R5		:INCREMENT DATA
4065	022304	022705	000400			CMP	#400,R5		:DONE BINARY PATTERN YET?
4066	022310	001327				BNE	1\$:BR IF NO
4067	022312	104400			7\$:	SCOPE			:SCOPE THIS TEST

4068
 4069
 4070
 4071
 4072
 4073
 4074
 4075
 4076
 4077
 4078
 4079
 4080
 4081
 4082
 4083
 4084
 4085
 4086
 4087
 4088
 4089
 4090
 4091
 4092
 4093
 4094
 4095
 4096
 4097
 4098
 4099
 4100
 4101
 4102
 4103
 4104
 4105
 4106
 4107
 4108
 4109
 4110
 4111
 4112
 4113
 4114
 4115
 4116
 4117
 4118
 4119
 4120
 4121
 4122
 4123

022314 012737 000050 001226 TST50:
 022322 012737 022646 001216
 022330 104412
 022332 012711 004000
 022336 012704 030434
 022342 005037 022436
 022346 012700 000004
 022352 004737 030016
 022356 004737 027034
 022362 004537 030152
 022366 030434
 022370 000004
 022372 004737 030126
 022376 004737 026702
 022402 005003
 022404 104415 000022
 022410 112405 12\$:
 022412 010502
 022414 012737 120001 027650
 022422 010537 022434
 022426 004537 027544
 022432 000010
 022434 000000 67\$:
 022436 000000 10\$:
 022440 013737 027652 022436
 022446 104415 000001 64\$:
 022452 106002
 022454 103005
 022456 004737 026650
 022462 103406
 022464 104006
 022466 000404
 022470 004737 026650 65\$:
 022474 103001
 022476 104006
 022500 66\$:

```

***** TEST 50 *****
*TRANSMITTER DDCMP CRC TEST
*THIS TEST TRANSMITS A FOUR CHARACTER MESSAGE WITH CRC
*BOTH DATA AND THE BCC ARE VERIFIED IN THE BIT
*WINDOW. THE FOUR CHARACTERS ARE 0,125,252,377
*THE TRANSMITTER IS CHECKED FOR GOING TO A MARK STATE AFTER THE BCC
*****

: TEST 50
-----
MOV #50,TSTNO
MOV #TST51,NEXT
MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11

;LOAD OUT DATA SILO

MOV #BIT11,(R1) ;SET LINE UNIT LOOP
MOV #MESDAT,R4 ;LOAD POINTER TO DATA
CLR 10$ ;CLEAR SOFT BCC
MOV #4,R0 ;LOAD CHARACTER COUNT
JSR PC,SYNLD ;LOAD 2 SYNCs IN OUT SILO
JSR PC,OUTRDY ;WAIT FOR OUTRDY
JSR R5,MESLD ;LOAD SILO WITH 4 CHAR MESS
MESDAT ;ADDRESS OF MESSAGE
4 ;NUMBER OF CHARACTERS
JSR PC,EOM ;LOAD GARBAGE CHARACTER, WITH EOM SET
JSR PC,OCOR ;WAIT FOR OCOR
CLR R3 ;CLEAR BIT COUNTER
DATACLK,22 ;CLOCK DATA
MOV (R4)+,R5 ;LOAD R5 WITH CHAR
MOV R5,R2 ;LOAD R2 WITH CHAR

;CHECK FIRST FOUR CHARACTER MESSAGE
;IN THE BIT WINDOW (0,125,252,377)

MOV #CRC16,XPOLY ;LOAD POLYNOMIAL
MOV R5,67$ ;LOAD SOFT CHAR FOR BCC
JSR R5,SIMBCC ;CALCULATE SOFT BCC
10 ;SHIFT COUNT
0 ;CHARACTER
0 ;OLD BCC
MOV CALBCC,10$ ;LOAD SOFT BCC FOR NEXT SHIFT
DATACLK, 1 ;SHIFT DATA IN TO BIT WINDOW
RORB R2 ;SHIFT SOFT DATA
BCC 65$ ;BR IF A SPACE
JSR PC,GETSI ;LOOK AT BIT WINDOW
BCS 66$ ;BR IF OK (MARK)
HLT 6 ;ERROR, BIT WINDOW WAS A SPACE
BR 66$ ;CONTINUE
JSR PC,GETSI ;LOOK AT BIT WINDOW
BCC 66$ ;BR IF OK (SPACE)
HLT 6 ;ERROR, BIT WINDOW WAS A MARK
  
```

```
4124 022500 005203          INC      R3          ;BUMP BIT COUNTER
4125 022502 022703 000010  CMP      #10,R3      ;DONE FULL 8 BITS YET
4126 022506 001357          BNE      64$         ;BR IF NO
4127 022510 005003          CLR      R3          ;CLEAR BIT COUNTER
4128 022512 005300          DEC      R0          ;DEC CHARACTER COUNT
4129 022514 001335          BNE      12$         ;BR IF NOT DONE YET
4130
4131          ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4132
4133 022516 013700 027652  MOV      CALBCC,R0   ;PUT BCC IN R0
4134 022522 104415 000001 68$:    DATACLK,1        ;SHIFT HARDWARE BCC
4135 022526 006000          ROR      R0          ;SHIFT SOFT BCC
4136 022530 103005          BCC      69$         ;BR IF CARRY CLEAR
4137 022532 004737 026650  JSR      PC,GETSI    ;LOOK AT BIT WINDOW
4138 022536 103406          BCS      70$         ;BR IF OK (MARK)
4139 022540 104014          HLT      14          ;ERROR, CRC WRONG (SPACE)
4140 022542 000404          BR       70$         ;CONTINUE
4141 022544 004737 026650 69$:    JSR      PC,GETSI    ;LOOK AT BIT WINDOW
4142 022550 103001          BCC      70$         ;BR IF OK (SPACE)
4143 022552 104014          HLT      14          ;ERROR, CRC WRONG (MARK)
4144 022554          70$:
4145 022554 005203          INC      R3          ;BUMP BIT COUNTER
4146 022556 022703 000020  CMP      #20,R3      ;FINISHED BCC YET?
4147 022562 001357          BNE      68$         ;BR IF NO
4148 022564 005003          CLR      R3          ;CLEAR BIT COUNTER
4149
4150          ;CHECK TO SEE IF TRANSMITTER IS MARKING
4151
4152 022566 104415 000001 2$:    DATACLK, 1          ;CLOCK TRANSMITTER
4153 022572 004737 026650  JSR      PC,GETSI    ;LOOK AT WINDOW
4154 022576 103401          BCS      3$          ;IT SHOULD BE MARKING
4155 022600 104024          HLT      24          ;ERROR, BIT WAS A SPACE
4156 022602 005203          3$:    INC      R3          ;BUMP BIT COUNTER
4157 022604 022703 000007  CMP      #7,R3      ;DONE YET
4158 022610 001366          BNE      2$          ;BR IF NO
4159 022612 104415 000010  DATACLK, 10        ;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
4160 022616 005003          CLR      R3          ;CLEAR BIT COUNTER
4161 022620 104415 000001 4$:    DATACLK, 1          ;SHIFT OUT NEXT BIT
4162 022624 004737 026650  JSR      PC,GETSI    ;LOOK AT BIT WINDOW
4163 022630 103401          BCS      +4          ;BR IF IT IS A MARK
4164 022632 104024          HLT      24          ;ERROR, TRANSMITTER IS NOT MARKING
4165 022634 005203          INC      R3          ;INC BIT COUNT
4166 022636 022703 000020  CMP      #20,R3      ;DONE YET?
4167 022642 001366          BNE      4$          ;BR IF NO
4168 022644 104400          5$:    SCOPE          ;SCOPE THIS TEST
4169
4170
4171          ;***** TEST 51 *****
4172          ;*RECEIVER DDCMP CRC TEST
4173          ;*THIS TEST CLOCKS A FOUR CHARACTER MESSAGE WITH BCC
4174          ;*AND VERIFYS CORRECT DATA RECEPTION AND BCC MATCH.
4175          ;*IN BCC MATCH IS CHECKED TO SEE THAT IT IS NOT ASSERTED
4176          ;*UNTIL THE LAST HALF OF THE CRC IS RECEIVED
4177          ;*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
4178          ;*****
4179
```

```

4180 ; TEST 51
4181 ;-----
4182 022646 012737 000051 001226 TST51: MOV #51,TSTNO
4183 022654 012737 023074 001216 MOV #TST52,NEXT
4184 ;R1 CONTAINS BASE DMC11 ADDRESS
4185 022662 104412 MSTCLR ;MASTER CLEAR DMC11
4186
4187 022664 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
4188 022670 012702 030434 MOV #MESDAT,R2 ;LOAD POINTER TO DATA
4189 022674 012700 000004 MOV #4,R0 ;LOAD CHARACTER COUNT
4190 022700 004737 030016 JSR PC,SYNLD ;LOAD 2 SYNCs IN OUT SILO
4191 022704 004737 027034 JSR PC,OUTRDY ;WAIT FOR OUTRDY
4192 022710 004537 030152 JSR R5,MESLD ;LOAD SILO WITH 4 CHAR MESS
4193 022714 030434 MESDAT ;ADDRESS OF MESSAGE
4194 022716 000004 4 ;NUMBER OF CHARACTERS
4195 022720 004737 030126 JSR PC,EOM ;LOAD GARBAGE CHARACTER, WITH EOM SET
4196 022724 004737 026702 JSR PC,OCOR ;WAIT FOR OCOR
4197 022730 104415 000114 DATACLK,114 ;CLOCK DATA
4198 022734 004737 027510 3$: JSR PC,INRDY ;WAIT FOR INRDY
4199 022740 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4200 022742 021204 021204 ;GET IN DATA
4201 022744 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
4202 022750 112205 MOVB (R2)+,R5 ;PUT 'EXPECTED' IN R5
4203
4204 022752 120504 CMPB R5,R4 ;COMPARE RECEIVED DATA
4205 022754 001401 BEQ 1$ ;BR IF OK
4206 022756 104010 HLT 10 ;DATA ERROR
4207 022760 1$: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4208 022760 104414 021244
4209 022762 021244 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
4210 022764 016104 000004 BIC #376,R4 ;CLEAR UNWANTED BITS
4211 022770 042704 000376 CLR R5 ;PUT 'EXPECTED' IN R5
4212 022774 005005 CMPB R5,R4 ;IS IN BCC MATCH STILL CLEAR?
4213 022776 120504 BEQ 4$ ;IN BCC MATCH ERROR
4214 023000 001401 HLT 15
4215 023002 104015 4$: DEC R0 ;DEC CHARACTER COUNT
4216 023004 005300 BNE 3$ ;BR IF NOT DONE YET
4217 023006 001352
4218
4219 ;CHECK TO SEE THAT IN BCC MATCH IS SET
4220
4221
4222 023010 004737 027510 JSR PC,INRDY ;WAIT FOR INRDY
4223 023014 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4224 023016 021204 021204 ;GET FIRST HALF OF CRC
4225 023020 116137 000004 001252 MOVB 4(R1),TEMP3 ;PUT IN TEMP3
4226 023026 042737 177400 001252 BIC #177400,TEMP3 ;CLEAR HI BYTE
4227 023034 004737 027510 JSR PC,INRDY ;WAIT FOR INRDY
4228 023040 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4229 023042 021244 021244
4230 023044 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
4231 023050 042704 000376 BIC #376,R4 ;CLEAR UNWANTED BITS
4232 023054 012705 000001 MOV #1,R5 ;PUT 'EXPECTED' IN R5
4233 023060 120504 CMPB R5,R4 ;IS IN BCC MATCH SET?
4234 023062 001401 BEQ 25$
4235 023064 104015 HLT 15 ;IN BCC MATCH ERROR

```

```

4236 023066          25$:
4237 023066 104414   ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4238 023070 021204   021204          ;GET LAST HALF
4239 023072 104400   SCOPE           ;SCOPE THIS TEST
4240
4241
4242
4243   ;***** TEST 52 *****
4244   ;*DDCMP EOM FUNCTION TEST
4245   ;*THIS TEST LOADS OUT SILO WITH: 2 SYNCs,4 CHAR MESSAGE,EOM
4246   ;*4 CHARACTER MESS,EOM. THE DATA STREAM IS CHECKED TO BE
4247   ;*4 CHAR,BCC,4 CHAR,BCC,MARKS. THIS TEST VERIFYS THAT
4248   ;*THE CHARCTERS LOADED WITH EOM SET ARE LOST
4249   ;*ALL DATA AND BCC'S ARE CHECKED IN THE BIT WINDOW
4250   ;*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
4251   ;*RECEIVED DATA IS VERIFIED, AND IN BCC MATCH IS CHECKED
4252   ;*****
4253   ; TEST 52
4254   ;-----
4255 023074 012737 000052 001226 TST52: MOV #52,TSTNO
4256 023102 012737 024174 001216 MOV #TST53,NEXT
4257
4258 023110 104412   MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
4259
4260
4261   ;LOAD OUT DATA SILO
4262 023112 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
4263 023116 012704 030434 MOV #MESDAT,R4  ;LOAD POINTER TO DATA
4264 023122 005037 023232 CLR 10$         ;CLEAR SOFT BCC
4265 023126 012700 000004 MOV #4,R0       ;LOAD CHARACTER COUNT
4266 023132 004737 030016 JSR PC,SYNLD   ;LOAD 2 SYNCs IN OUT SILO
4267 023136 004737 027034 JSR PC,OUTRDY  ;WAIT FOR OUTRDY
4268 023142 004537 030152 JSR R5,MESLD   ;LOAD SILO WITH 4 CHAR MESS
4269 023146 030434 MESDAT          ;ADDRESS OF MESSAGE
4270 023150 000004 4             ;NUMBER OF CHARACTERS
4271 023152 004737 030126 JSR PC,EOM     ;LOAD GARBAGE CHARACTER, WITH EOM SET
4272 023156 004537 030152 JSR R5,MESLD   ;LOAD FOUR MORE CHARACTERS
4273 023162 030434 MESDAT          ;ADDRESS OF MESSAGE
4274 023164 000004 4             ;NUMBER OF CHACTERS
4275 023166 004737 030126 JSR PC,EOM     ;SET EOM
4276 023172 004737 026702 JSR PC,OCOR    ;WAIT FOR OCOR
4277 023176 005003 CLR R3         ;CLEAR BIT COUNTER
4278 023200 104415 000022 DATACLK,22    ;CLOCK DATA
4279 023204 112405 12$: MOVB (R4)+,R5 ;LOAD R5 WITH CHAR
4280 023206 010502 MOV R5,R2     ;LOAD R2 WITH CHAR
4281
4282   ;CHECK FIRST FOUR CHARACTER MESSAGE
4283   ;IN THE BIT WINDOW (0,125,252,377)
4284
4285 023210 012737 120001 027650 MOV #CRC16,XPOLY ;LOAD POLYNOMIAL
4286 023216 010537 023230 MOV R5,67$     ;LOAD SOFT CHAR FOR BCC
4287 023222 004537 027544 JSR R5,SIMBCC  ;CALCULATE SOFT BCC
4288 023226 000010 10          ;SHIFT COUNT
4289 023230 000000 67$: 0          ;CHARACTER
4290 023232 000000 10$: 0          ;OLD BCC
4291 023234 013737 027652 023232 MOV CALBCC,10$ ;LOAD SOFT BCC FOR NEXT SHIFT

```

```

4292 023242 104415 000001 64$: DATACLK, 1 ;SHIFT DATA IN TO BIT WINDOW
4293 023246 106002 RORB R2 ;SHIFT SOFT DATA
4294 023250 103005 BCC 65$ ;BR IF A SPACE
4295 023252 004737 026650 JSR PC,GETSI ;LOOK AT BIT WINDOW
4296 023256 103406 BCS 66$ ;BR IF OK (MARK)
4297 023260 104006 HLT 6 ;ERROR, BIT WINDOW WAS A SPACE
4298 023262 000404 BR 66$ ;CONTINUE
4299 023264 004737 026650 65$: JSR PC,GETSI ;LOOK AT BIT WINDOW
4300 023270 103001 BCC 66$ ;BR IF OK (SPACE)
4301 023272 104006 HLT 6 ;ERROR, BIT WINDOW WAS A MARK
4302 023274 66$.
4303 023274 005203 INC R3 ;BUMP BIT COUNTER
4304 023276 022703 000010 CMP #10,R3 ;DONE FULL 8 BITS YET
4305 023302 001357 BNE 64$ ;BR IF NO
4306 023304 005003 CLR R3 ;CLEAR BIT COUNTER
4307 023306 005300 DEC R0 ;DEC CHARACTER COUNT
4308 023310 001335 BNE 12$ ;BR IF NOT DONE YET
4309
4310 ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4311
4312 023312 013700 027652 MOV CALBCC,R0 ;PUT BCC IN R0
4313 023316 104415 000001 68$: DATACLK,1 ;SHIFT HARDWARE BCC
4314 023322 006000 ROR R0 ;SHIFT SOFT BCC
4315 023324 103005 BCC 69$ ;BR IF CARRY CLEAR
4316 023326 004737 026650 JSR PC,GETSI ;LOOK AT BIT WINDOW
4317 023332 103406 BCS 70$ ;BR IF OK (MARK)
4318 023334 104014 HLT 14 ;ERROR, CRC WRONG (SPACE)
4319 023336 000404 BR 70$ ;CONTINUE
4320 023340 004737 026650 69$: JSR PC,GETSI ;LOOK AT BIT WINDOW
4321 023344 103001 BCC 70$ ;BR IF OK (SPACE)
4322 023346 104014 HLT 14 ;ERROR, CRC WRONG (MARK)
4323 023350 70$:
4324 023350 005203 INC R3 ;BUMP BIT COUNTER
4325 023352 022703 000020 CMP #20,R3 ;FINISHED BCC YET?
4326 023356 001357 BNE 68$ ;BR IF NO
4327 023360 005003 CLR R3 ;CLEAR BIT COUNTER
4328 023362 012700 000004 MOV #4,R0 ;RESET CHARACTER COUNTER
4329 023366 012704 030434 MOV #MESDAT,R4 ;LOAD MESSAGE POINTER
4330 023372 005037 023424 CLR 11$ ;CLR SOFT BCC
4331 023376 112405 13$: MOVB (R4)+,R5 ;LOAD CHAR IN R5
4332 023400 010502 MOV R5,R2 ;LOAD CHAR IN R2
4333
4334 ;CHECK SECOND MESSAGE IN THE BIT WINDOW (0,125,252,377)
4335
4336 023402 012737 120001 027650 MOV #CRC16,XPOLY ;LOAD POLYNOMIAL
4337 023410 010537 023422 MOV R5,76$ ;LOAD SOFT CHAR FOR BCC
4338 023414 004537 027544 JSR R5,SIMBCC ;CALCULATE SOFT BCC
4339 023420 000010 10 ;SHIFT COUNT
4340 023422 000000 76$ 0 ;CHARACTER
4341 023424 000000 11$ 0 ;OLD BCC
4342 023426 013737 027652 023424 MOV CALBCC,11$ ;LOAD SOFT BCC FOR NEXT SHIFT
4343 023434 104415 000001 73$: DATACLK, 1 ;SHIFT DATA IN TO BIT WINDOW
4344 023440 106002 RORB R2 ;SHIFT SOFT DATA
4345 023442 103005 BCC 74$ ;BR IF A SPACE
4346 023444 004737 026650 JSR PC,GETSI ;LOOK AT BIT WINDOW
4347 023450 103406 BCS 75$ ;BR IF OK (MARK)

```

BASIC RECEIVER TESTS

4348	023452	104006		HLT	6		:ERROR, BIT WINDOW WAS A SPACE
4349	023454	000404		BR	758		:CONTINUE
4350	023456	004737	026650	748:	JSR	PC,GETSI	:LOOK AT BIT WINDOW
4351	023462	103001		BCC	758		:BR IF OK (SPACE)
4352	023464	104006		HLT	6		:ERROR, BIT WINDOW WAS A MARK
4353	023466			758:			
4354	023466	005203		INC	R3		:BUMP BIT COUNTER
4355	023470	022703	000010	CMP	#10,R3		:DONE FULL 8 BITS YET
4356	023474	001357		BNE	738		:BR IF NO
4357	023476	005003		CLR	R3		:CLEAR BIT COUNTER
4358	023500	005300		DEC	R0		:DEC CHARACTER COUNT
4359	023502	001335		BNE	138		:BR IF NOT DONE YET
4360							
4361							:CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4362							
4363	023504	013700	027652	778:	MOV	CALBCC,R0	:PUT BCC IN R0
4364	023510	104415	000001	DATACLK,	1		:SHIFT HARDWARE BCC
4365	023514	006000		ROR	R0		:SHIFT SOFT BCC
4366	023516	103005		BCC	788		:BR IF CARRY CLEAR
4367	023520	004737	026650	JSR	PC,GETSI		:LOOK AT BIT WINDOW
4368	023524	103406		BCS	798		:BR IF OK (MARK)
4369	023526	104014		HLT	14		:ERROR, CRC WRONG (SPACE)
4370	023530	000404		BR	798		:CONTINUE
4371	023532	004737	026650	788:	JSR	PC,GETSI	:LOOK AT BIT WINDOW
4372	023536	103001		BCC	798		:BR IF OK (SPACE)
4373	023540	104014		HLT	14		:ERROR, CRC WRONG (MARK)
4374	023542			798:			
4375	023542	005203		INC	R3		:BUMP BIT COUNTER
4376	023544	022703	000020	CMP	#20,R3		:FINISHED BCC YET?
4377	023550	001357		BNE	778		:BR IF NO
4378	023552	005003		CLR	R3		:CLEAR BIT COUNTER
4379							
4380							:CHECK TO SEE IF TRANSMITTER IS MARKING
4381							
4382	023554	104415	000001	28:	DATACLK,	1	:CLOCK TRANSMITTER
4383	023560	004737	026650	JSR	PC,GETSI		:LOOK AT WINDOW
4384	023564	103401		BCS	38		:IT SHOULD BE MARKING
4385	023566	104024		HLT	24		:ERROR, BIT WAS A SPACE
4386	023570	005203		38:	INC	R3	:BUMP BIT COUNTER
4387	023572	022703	000007	CMP	#7,R3		:DONE YET
4388	023576	001366		BNE	28		:BR IF NO
4389	023600	104415	000010	DATACLK,	10		:GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
4390	023604	005003		CLR	R3		:CLEAR BIT COUNTER
4391	023606	104415	000001	48:	DATACLK,	1	:SHIFT OUT NEXT BIT
4392	023612	004737	026650	JSR	PC,GETSI		:LOOK AT BIT WINDOW
4393	023616	103401		BCS	+4		:BR IF IT IS A MARK
4394	023620	104024		HLT	24		:ERROR, TRANSMITTER IS NOT MARKING
4395	023622	005203		INC	R3		:INC BIT COUNT
4396	023624	022703	000020	CMP	#20,R3		:DONE YET?
4397	023630	001366		BNE	48		:BR IF NO
4398							
4399							:CHECK TO SEE THAT FIRST FOUR CHARACTER MESSAGE
4400							:WAS RECEIVED CORRECTLY (0,125,252,377)
4401							
4402	023632	104415	000001	DATACLK,	1		:GET LAST BIT IN RECEIVER
4403	023636	012703	000004	MOV	#4,R3		:R3=CHARACTER COUNT

```

4404 023642 012702 030434      MOV      #MESDAT,R2      ;LOAD MESSAGE POINTER IN R2
4405 023646 004737 027510      JSR      PC,INRDY       ;WAIT FOR INRDY
4406 023652 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4407 023654 021204 021204
4408 023656 016104 000004      MOV      4(R1),R4       ;PUT 'FOUND' IN R4
4409 023662 112205      MOV      (R2)+,R5       ;PUT 'EXPECTED' IN R5
4410 023664 120504      CMPB    R5,R4          ;IS RECEIVED DATA CORRECT?
4411 023666 001401      BEQ     41$            ;BR IF YES
4412 023670 104010      HLT     10             ;RECEIVE DATA ERROR
4413 023672 005303      41$:  DEC     R3         ;DEC CHARACTER COUNT
4414 023674 001364      BNE     40$            ;BR IF NOT DONE YET
4415
4416
4417
4418
4419 023676 004737 027510      JSR      PC,INRDY       ;WAIT FOR INRDY
4420 023702 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4421 023704 021204 021204                ;GET FIRST HALF OF CRC
4422 023706 116137 000004 001252      MOV      4(R1),TEMP3    ;PUT IN TEMP3
4423 023714 042737 177400 001252      BIC     #177400,TEMP3   ;CLEAR HI BYTE
4424 023722 004737 027510      JSR      PC,INRDY       ;WAIT FOR INRDY
4425 023726 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4426 023730 021244 021244
4427 023732 016104 000004      MOV      4(R1),R4       ;PUT 'FOUND' IN R4
4428 023736 042704 000376      BIC     #376,R4         ;CLEAR UNWANTED BITS
4429 023742 012705 000001      MOV      #1,R5          ;PUT 'EXPECTED' IN R5
4430 023746 120504      CMPB    R5,R4          ;IS IN BCC MATCH SET?
4431 023750 001401      BEQ     50$            ;IN BCC MATCH ERROR
4432 023752 104015
4433 023754
4434 023754 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4435 023756 021204 021204                ;GET LAST HALF
4436 023760 116137 000004 001251      MOV      4(R1),TEMP2+1  ;PUT IN TEMP2
4437 023766 042737 000377 001250      BIC     #377,TEMP2     ;CLEAR LO BYTE
4438 023774 053737 001250 001252      BIS     TEMP2,TEMP3     ;16 BIT BCC NOW IN TEMP3
4439 024002 023737 027652 001252      CMP     CALBCC,TEMP3    ;IS IT CORRECT?
4440 024010 001401      BEQ     42$            ;BR IF OK
4441 024012 104027
4442
4443
4444
4445
4446 024014 012703 000004      42$:  MOV     #4,R3        ;R3=CHARACTER COUNT
4447 024020 012702 030434      MOV     #MESDAT,R2     ;LOAD MESSAGE POINTER IN R2
4448 024024 004737 027510      43$:  JSR     PC,INRDY     ;WAIT FOR INRDY
4449 024030 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4450 024032 021204 021204
4451 024034 016104 000004      MOV     4(R1),R4       ;PUT 'FOUND' IN R4
4452 024040 112205      MOV     (R2)+,R5       ;PUT 'EXPECTED' IN R5
4453 024042 120504      CMPB   R5,R4          ;IS RECEIVED DATA CORRECT?
4454 024044 001401      BEQ    44$            ;BR IF YES
4455 024046 104010      HLT    10             ;RECEIVE DATA ERROR
4456 024050 005303      44$:  DEC     R3         ;DEC CHARACTER COUNT
4457 024052 001364      BNE    43$            ;BR IF NOT DONE YET
4458
4459
;CHECK TO SEE THAT IN BCC MATCH IS SET

```



```

4460 ;AND THAT THE BCC WAS RECEIVED CORRECTLY
4461
4462 024054 004737 027510 JSR PC,INRDY ;WAIT FOR INRDY
4463 024060 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4464 024062 021204 021204 ;GET FIRST HALF OF CRC
4465 024064 116137 000004 001252 MOV 4(R1),TEMP3 ;PUT IN TEMP3
4466 024072 042737 177400 001252 BIC #177400,TEMP3 ;CLEAR HI BYTE
4467 024100 004737 027510 JSR PC,INRDY ;WAIT FOR INRDY
4468 024104 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4469 024106 021244 021244
4470 024110 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
4471 024114 042704 000376 BIC #376,R4 ;CLEAR UNWANTED BITS
4472 024120 012705 000001 MOV #1,R5 ;PUT 'EXPECTED' IN R5
4473 024124 120504 CMPB R5,R4 ;IS IN BCC MATCH SET?
4474 024126 001401 BEQ 51$
4475 024130 104015 HLT 15 ;IN BCC MATCH ERROR
4476 024132 51$:
4477 024132 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4478 024134 021204 021204 ;GET LAST HALF
4479 024136 116137 000004 001251 MOV 4(R1),TEMP2+1 ;PUT IN TEMP2
4480 024144 042737 000377 001250 BIC #377,TEMP2 ;CLEAR LO BYTE
4481 024152 053737 001250 001252 BIS TEMP2,TEMP3 ;16 BIT BCC NOW IN TEMP3
4482 024160 023737 027652 001252 CMP CALBCC,TEMP3 ;IS IT CORRECT?
4483 024166 001401 BEQ 5$
4484 024170 104027 HLT 27 ;BR IF OK
4485 024172 104400 5$: SCOPE ;SCOPE THIS TEST
4486
4487
4488 :***** TEST 53 *****
4489 :*DDCMP EOM FUNCTION TEST
4490 :*THIS TEST LOADS OUT SILO WITH: 2 SYNCs,4 CHAR MESSAGE,EOM
4491 :*SOM,4 CHAR MESS,EOM. THE DATA STREAM IS CHECKED TO BE
4492 :*4 CHAR,BCC,4 CHAR,BCC,MARKS. THIS TEST VERIFYS THAT
4493 :*THE CHARCTERS LOADED WITH EOM SET ARE LOST
4494 :*ALSO THAT THE CHAR LOADED WITH SOM IS NOT IN THE BCC
4495 :*ALL DATA AND BCC'S ARE CHECKED IN THE BIT WINDOW
4496 :*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
4497 :*RECEIVED DATA IS VERIFIED, AND IN BCC MATCH IS CHECKED
4498 :*****
4499
4500 : TEST 53
4501 :-----
4502 024174 012737 000053 001226 TST53: MOV #53,TSTNO
4503 024202 012737 025374 001216 MOV #TST54,NEXT
4504 ;R1 CONTAINS BASE DMC11 ADDRESS
4505 024210 104412 MSTCLR ;MASTER CLEAR DMC11
4506
4507 ;LOAD OUT DATA SILO
4508
4509 024212 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
4510 024216 012704 030434 MOV #MESDAT,R4 ;LOAD POINTER TO DATA
4511 024222 005037 024336 CLR 10$ ;CLEAR SOFT BCC
4512 024226 012700 000004 MOV #4,R0 ;LOAD CHARACTER COUNT
4513 024232 004737 030016 JSR PC,SYNLD ;LOAD 2 SYNCs IN OUT SILO
4514 024236 004737 027034 JSR PC,OUTRDY ;WAIT FOR OUTRDY
4515 024242 004537 030152 JSR R5,MESLD ;LOAD SILO WITH 4 CHAR MESS
    
```

4516	024246	030434			MESDAT		: ADDRESS OF MESSAGE
4517	024250	000004			4		: NUMBER OF CHARACTERS
4518	024252	004737	030126		JSR	PC,EOM	: LOAD GARBAGE CHARACTER, WITH EOM SET
4519	024256	004737	030076		JSR	PC,SOM	: LOAD GARBAGE CHAR WITH SOM SET
4520	024262	004537	030152		JSR	R5,MESLD	: LOAD FOUR MORE CHARACTERS
4521	024266	030434			MESDAT		: ADDRESS OF MESSAGE
4522	024270	000004			4		: NUMBER OF CHACTERS
4523	024272	004737	030126		JSR	PC,EOM	: SET EOM
4524	024276	004737	026702		JSR	PC,OCOR	: WAIT FOR OCOR
4525	024302	005003			CLR	R3	: CLEAR BIT COUNTER
4526	024304	104415	000022		DATACLK,22		: CLOCK DATA
4527	024310	112405		17\$:	MOVB	(R4)+,R5	: LOAD R5 WITH CHAR
4528	024312	010502			MOV	R5,R2	: LOAD R2 WITH CHAR
4529							
4530							: CHECK FIRST FOUR CHARACTER MESSAGE
4531							: IN THE BIT WINDOW (0,125,252,377)
4532							
4533	024314	012737	120001	027650	MOV	#CRC16,XPOLY	: LOAD POLYNOMIAL
4534	024322	010537	024334		MOV	R5,67\$: LOAD SOFT CHAR FOR BCC
4535	024326	004537	027544		JSR	R5,SIMBCC	: CALCULATE SOFT BCC
4536	024332	000010			10		: SHIFT COUNT
4537	024334	000000		67\$:	0		: CHARACTER
4538	024336	000000		10\$:	0		: OLD BCC
4539	024340	013737	027652	024336	MOV	CALBCC,10\$: LOAD SOFT BCC FOR NEXT SHIFT
4540	024346	104415	000001	64\$:	DATACLK,1		: SHIFT DATA IN TO BIT WINDOW
4541	024352	106002			RORB	R2	: SHIFT SOFT DATA
4542	024354	103005			BCC	65\$: BR IF A SPACE
4543	024356	004737	026650		JSR	PC,GETSI	: LOOK AT BIT WINDOW
4544	024362	103406			BCS	66\$: BR IF OK (MARK)
4545	024364	104006			HLT	6	: ERROR, BIT WINDOW WAS A SPACE
4546	024366	000404			BR	66\$: CONTINUE
4547	024370	004737	026650	65\$:	JSR	PC,GETSI	: LOOK AT BIT WINDOW
4548	024374	103001			BCC	66\$: BR IF OK (SPACE)
4549	024376	104006			HLT	6	: ERROR, BIT WINDOW WAS A MARK
4550	024400			66\$:			
4551	024400	005203			INC	R3	: BUMP BIT COUNTER
4552	024402	022703	000010		CMP	#10,R3	: DONE FULL 8 BITS YET
4553	024406	001357			BNE	64\$: BR IF NO
4554	024410	005003			CLR	R3	: CLEAR BIT COUNTER
4555	024412	005300			DEC	R0	: DEC CHARACTER COUNT
4556	024414	001335			BNE	12\$: BR IF NOT DONE YET
4557							
4558							: CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4559							
4560	024416	013700	027652		MOV	CALBCC,R0	: PUT BCC IN R0
4561	024422	104415	000001	68\$:	DATACLK,1		: SHIFT HARDWARE BCC
4562	024426	006000			ROR	R0	: SHIFT SOFT BCC
4563	024430	103005			BCC	69\$: BR IF CARRY CLEAR
4564	024432	004737	026650		JSR	PC,GETSI	: LOOK AT BIT WINDOW
4565	024436	103406			BCS	70\$: BR IF OK (MARK)
4566	024440	104014			HLT	14	: ERROR, CRC WRONG (SPACE)
4567	024442	000404			BR	70\$: CONTINUE
4568	024444	004737	026650	69\$:	JSR	PC,GETSI	: LOOK AT BIT WINDOW
4569	024450	103001			BCC	70\$: BR IF OK (SPACE)
4570	024452	104014			HLT	14	: ERROR, CRC WRONG (MARK)
4571	024454			70\$:			


```

4628                                     ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4629
4630 024660 013700 027652
4631 024664 104415 000001      77$: MOV CALBCC,R0      ;PUT BCC IN R0
4632 024670 006000              DATACLK,1      ;SHIFT HARDWARE BCC
4633 024672 103005              ROR RO         ;SHIFT SOFT BCC
4634 024674 004737 026650      BCC 78$       ;BR IF CARRY CLEAR
4635 024700 103406              JSR PC,GETSI   ;LOOK AT BIT WINDOW
4636 024702 104014              BCS 79$       ;BR IF OK (MARK)
4637 024704 000404              HLT 14        ;ERROR, CRC WRONG (SPACE)
4638 024706 004737 026650      BR 79$        ;CONTINUE
4639 024712 103001              JSR PC,GETSI   ;LOOK AT BIT WINDOW
4640 024714 104014              BCC 79$       ;BR IF OK (SPACE)
4641 024716                      HLT 14        ;ERROR, CRC WRONG (MARK)
4642 024716 005203              79$: INC R3     ;BUMP BIT COUNTER
4643 024720 022703 000020      CMP #20,R3    ;FINISHED BCC YET?
4644 024724 001357              BNE 77$       ;BR IF NO
4645 024726 005003              CLR R3        ;CLEAR BIT COUNTER
4646
4647                                     ;CHECK TO SEE IF TRANSMITTER IS MARKING
4648
4649 024730 104415 000001      2$: DATACLK, 1 ;CLOCK TRANSMITTER
4650 024734 004737 026650      JSR PC,GETSI  ;LOOK AT WINDOW
4651 024740 103401              BCS 3$        ;IT SHOULD BE MARKING
4652 024742 104024              HLT 24        ;ERROR, BIT WAS A SPACE
4653 024744 005203              3$: INC R3     ;BUMP BIT COUNTER
4654 024746 022703 000007      CMP #7,R3     ;DONE YET
4655 024752 001366              BNE 2$        ;BR IF NO
4656 024754 104415 000010      DATACLK, 10 ;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
4657 024760 005003              CLR R3        ;CLEAR BIT COUNTER
4658 024762 104415 000001      4$: DATACLK, 1 ;SHIFT OUT NEXT BIT
4659 024766 004737 026650      JSR PC,GETSI  ;LOOK AT BIT WINDOW
4660 024772 103401              BCS +4        ;BR IF IT IS A MARK
4661 024774 104024              HLT 24        ;ERROR, TRANSMITTER IS NOT MARKING
4662 024776 005203              INC R3        ;INC BIT COUNT
4663 025000 022703 000020      CMP #20,R3    ;DONE YET?
4664 025004 001366              BNE 4$        ;BR IF NO
4665
4666                                     ;CHECK TO SEE THAT FIRST FOUR CHARACTER MESSAGE
4667                                     ;WAS RECEIVED CORRECTLY (0,125,252,377)
4668
4669 025006 104415 000001      DATACLK, 1   ;GET LAST BIT IN RECEIVER
4670 025012 012703 000004      MOV #4,R3     ;R3=CHARACTER COUNT
4671 025016 012702 030434      MOV #MESDAT,R2 ;LOAD MESSAGE POINTER IN R2
4672 025022 004737 027510      40$: JSR PC,INRDY ;WAIT FOR INRDY
4673 025026 104414              ROMCLK       ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4674 025030 021204              021204
4675 025032 016104 000004      MOV 4(R1),R4 ;PUT 'FOUND' IN R4
4676 025036 112205              MOVB (R2)+,R5 ;PUT 'EXPECTED' IN R5
4677 025040 120504              CMPB R5,R4   ;IS RECEIVED DATA CORRECT?
4678 025042 001401              BEQ 41$      ;BR IF YES
4679 025044 104010              HLT 10       ;RECEIVE DATA ERROR
4680 025046 005303              41$: DEC R3   ;DEC CHARACTER COUNT
4681 025050 001364              BNE 40$      ;BR IF NOT DONE YET
4682
4683                                     ;CHECK TO SEE THAT IN BCC MATCH IS SET

```

```

4684                                     ;AND THAT THE BCC WAS RECEIVED CORRECTLY
4685
4686 025052 004737 027510                JSR     PC,INRDY      ;WAIT FOR INRDY
4687 025056 104414                        ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4688 025060 021204                        021204                ;GET FIRST HALF OF CRC
4689 025062 116137 000004 001252        MOV     4(R1),TEMP3   ;PUT IN TEMP3
4690 025070 042737 177400 001252        BIC     #177400,TEMP3 ;CLEAR HI BYTE
4691 025076 004737 027510                JSR     PC,INRDY      ;WAIT FOR INRDY
4692 025102 104414                        ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4693 025104 021244                        021244                ;
4694 025106 016104 000004                MOV     4(R1),R4      ;PUT 'FOUND' IN R4
4695 025112 042704 000376                BIC     #376,R4       ;CLEAR UNWANTED BITS
4696 025116 012705 000001                MOV     #1,R5         ;PUT 'EXPECTED' IN R5
4697 025122 120504                        CMP     R5,R4         ;IS IN BCC MATCH SET?
4698 025124 001401                        BEQ     50$           ;
4699 025126 104015                        HLT     15            ;IN BCC MATCH ERROR
4700 025130                                50$:
4701 025130 104414                        ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4702 025132 021204                        021204                ;GET LAST HALF
4703 025134 116137 000004 001251        MOV     4(R1),TEMP2+1 ;PUT IN TEMP2
4704 025142 042737 000377 001250        BIC     #377,TEMP2   ;CLEAR LO BYTE
4705 025150 053737 001250 001252        BIS     TEMP2,TEMP3  ;16 BIT BCC NOW IN TEMP3
4706 025156 023737 027652 001252        CMP     CALBCC,TEMP3 ;IS IT CORRECT?
4707 025164 001401                        BEQ     45$           ;BR IF OK
4708 025166 104027                        HLT     27            ;
4709
4710                                     ;CHECK THAT CHARACTER LOADED WITH SOM WAS RECEIVED (000)
4711
4712 025170 004737 027510                45$: JSR     PC,INRDY      ;WAIT FOR INRDY
4713 025174 104414                        ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4714 025176 021204                        021204                ;GET RECEIVE DATA
4715 025200 016104 000004                MOV     4(R1),R4      ;PUT 'FOUND' IN R4
4716 025204 005005                        CLR     R5            ;PUT 'EXPECTED' IN R5
4717 025206 120504                        CMP     R5,R4         ;IS RECEIVED DATA CORRECT?
4718 025210 001401                        BEQ     42$           ;BR IF YES
4719 025212 104010                        HLT     10            ;RECEIVE DATA ERROR
4720
4721                                     ;CHECK TO SEE THAT SECOND FOUR CHARACTER MESSAGE
4722                                     ;WAS RECEIVED CORRECTLY (0,125,252,377)
4723
4724 025214 012703 000004                42$: MOV     #4,R3      ;R3=CHARACTER COUNT
4725 025220 012702 030434                MOV     #MESDAT,R2   ;LOAD MESSAGE POINTER IN R2
4726 025224 004737 027510                43$: JSR     PC,INRDY      ;WAIT FOR INRDY
4727 025230 104414                        ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4728 025232 021204                        021204                ;
4729 025234 016104 000004                MOV     4(R1),R4      ;PUT 'FOUND' IN R4
4730 025240 112205                        MOV     (R2)+,R5     ;PUT 'EXPECTED' IN R5
4731 025242 120504                        CMP     R5,R4         ;IS RECEIVED DATA CORRECT?
4732 025244 001401                        BEQ     44$           ;BR IF YES
4733 025246 104010                        HLT     10            ;RECEIVE DATA ERROR
4734 025250 005303                44$: DEC     R3      ;DEC CHARACTER COUNT
4735 025252 001364                BNE     43$          ;BR IF NOT DONE YET
4736
4737                                     ;CHECK TO SEE THAT IN BCC MATCH IS SET
4738                                     ;AND THAT THE BCC WAS RECEIVED CORRECTLY
4739

```

```

4740 025254 004737 027510 JSR PC,INRDY ;WAIT FOR INRDY
4741 025260 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4742 025262 021204 021204 ;GET FIRST HALF OF CRC
4743 025264 116137 000004 001252 MOVB 4(R1),TEMP3 ;PUT IN TEMP3
4744 025272 042137 177400 001252 BIC #177400,TEMP3 ;CLEAR HI BYTE
4745 025300 004737 027510 JSR PC,INRDY ;WAIT FOR INRDY
4746 025304 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4747 025306 021244 021244
4748 025310 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
4749 025314 042704 000376 BIC #376,R4 ;CLEAR UNWANTED BITS
4750 025320 012705 000001 MOV #1,R5 ;PUT 'EXPECTED' IN R5
4751 025324 120504 CMPB R5,R4 ;IS IN BCC MATCH SET?
4752 025326 001401 BEQ 51$
4753 025330 104015 HLT 15 ;IN BCC MATCH ERROR
4754 025332 51$:
4755 025332 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4756 025334 021204 021204 ;GET LAST HALF
4757 025336 116137 000004 001251 MOVB 4(R1),TEMP2+1 ;PUT IN TEMP2
4758 025344 042737 000377 001250 BIC #377,TEMP2 ;CLEAR LO BYTE
4759 025352 053737 001250 001252 BIS TEMP2,TEMP3 ;16 BIT BCC NOW IN TEMP3
4760 025360 023737 027652 001252 CMP CALBCC,TEMP3 ;IS IT CORRECT?
4761 025366 001401 BEQ 5$ ;BR IF OK
4762 025370 104027 HLT 27
4763 025372 104400 5$: SCOPE ;SCOPE THIS TEST
4764
4765
4766 :***** TEST 54 *****
4767 :*EMPTY SILO TEST
4768 :*LOAD SILO WITH 2 SYNCs, 4 CHAR MESSAGE, SINGLE CLOCK
4769 :*UNTIL THE SILO IS EMPTY, LOAD 4 MORE CHARACTERS IN THE
4770 :*SILO. GIVE MORE TICKS, AND VERIFY THAT ONLY THE FIRST
4771 :*4 CHARACTER MESSAGE WAS RECEIVED AND THAT RTS IS CLEAR
4772 :*****
4773
4774 : TEST 54
4775 :-----
4776 025374 012737 000054 001226 TST54: MOV #54,TSTNO
4777 025402 012737 025626 001216 MOV #TST55,NEXT
4778 ;R1 CONTAINS BASE DMC11 ADDRESS
4779 025410 104412 MSTCLR ;MASTER CLEAR DMC11
4780 025412 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
4781 025416 012702 030434 MOV #MESDAT,R2 ;R2 POINTS TO MESSAGE
4782 025422 012700 000004 MOV #4,R0 ;R0 = CHAR COUNT
4783 025426 004737 030016 JSR PC,SYNLD ;LOAD SILO WITH TWO SYNCs
4784 025432 004737 027034 JSR PC,OUTRDY ;WAIT FOR OUTRDY
4785 025436 004537 030152 JSR R5,MESLD ;LOAD MESSAGE IN SILO
4786 025442 030434 MESDAT ;START OF MESSAGE
4787 025444 000004 4 ;CHARACTER COUNT
4788 025446 004737 026702 JSR PC,OCOR ;WAIT FOR OCOR
4789 025452 104415 000065 DATACLK, 65 ;CLOCK DATA (EMPTY SILO)
4790 025456 004537 030152 JSR R5,MESLD ;PUT MORE CHARACTERS IN SILO
4791 025462 030434 MESDAT
4792 025464 000004 4
4793 025466 004737 026702 JSR PC,OCOR
4794 025472 104415 000005 DATACLK, 5 ;CLOCK UNTIL RTS IS CLEARED
4795 025476 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

```

```

4796 025500 021264          021264          ;GET RTS
4797 025502 032761 000040 000004          BIT      #BIT5,4(R1) ;IS IT CLEAR?
4798 025510 001401          BEQ      5$          ;BR IF YES
4799 025512 104034          HLT      34          ;ERROR, RTS NOT CLEAR
4800 025514 104415 000041          5$: DATACLK, 41 ;CLOCK XMITTER SOME MORE
4801 025520 004737 027510          $: JSR      PC,INRDY ;OK LETS CHECK WHAT WAS RECEIVED
4802 025524 104414          ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4803 025526 021204          021204 ;GET RECEIVE DATA
4804 025530 016104 000004          MOV      4(R1),R4 ;PUT IT IN R4
4805 025534 112205          MOVB    (R2)+,P5 ;R5 = 'EXPECTED'
4806 025536 120504          CMPB   R5,R4 ;IS DATA CORRECT?
4807 025540 001401          BEQ     2$          ;BR IF OK
4808 025542 104010          HLT     10          ;DATA ERROR
4809 025544 005300          2$: DEC     R0 ;DEC CHAR COUNT
4810 025546 001364          BNE     1$          ;BR IF NOT DONE YET
4811 025550 004737 027510          3$: JSR     PC,INPDY ;WAIT FOR INPDY
4812 025554 104414          ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4813 025556 021204          021204 ;GET RECEIVE DATA
4814 025560 016104 000004          MOV      4(R1),R4 ;PUT IT IN 'FOUND'
4815 025564 012705 000377          MOV      #377,R5 ;R5 = 'EXPECTED'
4816 025570 120504          CMPB   R5,R4 ;SHOULD SEE 377
4817 025572 001401          BEQ     4$          ;BR IF OK
4818 025574 104010          HLT     10          ;ERROR, TRANSMITTER DID NOT ABORT
4819 025576 004737 027510          4$: JSR     PC,INRDY ;WAIT FOR INRDY
4820 025602 104414          ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4821 025604 021204          021204 ;GET RECEIVE DATA
4822 025606 016104 000004          MOV      4(R1),R4 ;PUT IT IN 'FOUND'
4823 025612 012705 000377          MOV      #377,R5 ;R5 = 'EXPECTED'
4824 025616 120504          CMPB   R5,R4 ;SHOULD SEE 377
4825 025620 001401          BEQ     10$         ;BR IF OK
4826 025622 104010          HLT     10          ;ERROR, TRANSMITTER DID NOT ABORT
4827 025624          10$:
4828 025624 104400          SCOPE ;SCOPE THIS TEST
4829
4830
4831
4832
4833
4834
4835
4836
4837
4838

```

```

:***** TEST 55 *****
:*HALF DUPLEX TEST
:*SET LINE UNIT LOOP AND HALF DUPLEX, SEND SYNCs AND A
:*MESSAGE. VERIFY THAT IN-ACTIVE AND IN-READY ARE CLEAR
:*****

```

```

: TEST 55
:-----

```

```

4839 025626 012737 000055 001226 TST55: MOV      #55,TSTNO
4840 025634 012737 025744 001216          MOV      #TST56,NEXT
4841
4842 025642 104412          MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
4843 025644 012702 000012          MOV      #12,R2 ;MASTER CLEAR DMC11
4844 025650 012711 004000          MOV      #BIT11,(R1) ;SAVE R2 FOR TYPEOUT
4845 025654 012761 000020 000004          MOV      #BIT4,4(R1) ;SET LINE UNIT LOOP
4846 025662 104414          ROMCLK ;LOAD PORT4
4847 025664 122113          122113 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4848 025666 004737 030016          JSR      PC,SYNLD ;SET H/D BIT
4849 025672 004737 027034          JSR      PC,OUTRDY ;LOAD 2 SYNCs
4850 025676 004537 030152          JSR      R5,MESLD ;WAIT FOR OUTRDY
4851 025702 030434          MESDAT ;LOAD 4 CHAR MESSAGE
;ADDRESS OF MESSAGE

```

```

4852 025704 000004          4          ; CHARACTER COUNT
4853 025706 004737 026702 JSR      PC,OCOR      ; WAIT FOR OCOR
4854 025712 104415 000073 DATACLK, 73        ; SEND MESSAGE
4855 025716 104414          ROMCLK      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4856 025720 021244          021244      ; READ LU-12
4857 025722 016104 000004 MOV      4(R1),R4     ; PUT 'FOUND' IN R4
4858 025726 042704 000257 BIC      #257,R4     ; CLEAR UNWANTED BITS
4859 025732 005005          CLR      R5          ; R5 = 'EXPECTED'
4860 025734 120504          CMPB     R5,R4       ; IN-ACTIVE AND IN-RDY SHOULD BE CLEAR
4861 025736 001401          BEQ      1$         ; BR IF OK
4862 025740 104035          HLT      3$         ; ERROR BOTH ARE NOT CLEAR
4863 025742 104400          1$: SCOPE
4864
4865
4866          ;***** TEST 56 *****
4867          ;*DDCMP CABLE DATA TFST
4868          ;*THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
4869          ;*4 SYNCs,16 CHAR,EOM,16 CHAR,EOM,16 CHAR,EOM
4870          ;*THE 16 CHARACTERS INCLUDE A FLOATING ONE AND ZERO
4871          ;*THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
4872          ;*RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
4873          ;*LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST
4874          ;:*****
4875
4876          ; TEST 56
4877          ;-----
4878 025744 012737 000056 001226 TST56: MOV      #56,TSTNO
4879 025752 012737 026332 001216 MOV      #TST57,NEXT
4880
4881 025760 104412          MSTCLR      ; R1 CONTAINS BASE DMC11 ADDRESS
4882          ; MASTER CLEAR DMC11
4883 025762 032737 040000 001366 BIT      #BIT14,STAT1 ; SKIP TEST IF NO
4884 025770 001557          BEQ      3$         ; LOOPBACK CONNECTOR ON
4885 025772 012711 004000          MOV      #BIT11,(R1) ; SET LINE UNIT LOOP
4886 025776 004737 030016          JSR      PC,SYNLD    ; LOAD 2 SYNCs
4887 026002 004737 030016          JSR      PC,SYNLD    ; LOAD 2 MORE SYNCs
4888 026006 012737 120001 027650 MOV      #CRC16,XPOLY ; LOAD POLYNOMIAL FOR SOFT CRC CALC
4889 026014 005037 026044          CLR      6$         ; CLEAR OLD BCC
4890 026020 012703 000020          MOV      #16.,R3    ; CHARACTER COUNT
4891 026024 012702 030440          MOV      #FLTDAT,R2 ; R2= POINTER
4892 026030 112237 026042          7$: MOVB   (R2)+,5$    ; LOAD CHAR FOR SOFT BCC CALC.
4893 026034 004537 027544          JSR      R5,SIMBCC  ; CALC SOFT BCC
4894 026040 000010          10        ; SHIFT COUNT
4895 026042 000000          5$: 0          ; CHARACTER
4896 026044 000000          6$: 0          ; OLD BCC
4897 026046 013737 027652 026044 MOV      CALBCC,6$   ; LOAD OLD BCC
4898 026054 005303          DEC      R3         ; DEC COUNT
4899 026056 001364          BNE     7$         ; BR IF NOT DONE YET
4900 026060 004537 030152          JSR      R5,MESLD   ; LOAD SILO
4901 026064 030440          FLTDAT      ; MESSAGE ADDRESS
4902 026066 000020          16.        ; CHARACTER COUNT
4903 026070 004737 030126          JSR      PC,EOM     ; LOAD AN EOM
4904 026074 004537 030152          JSR      R5,MESLD   ; LOAD SILO
4905 026100 030440          FLTDAT      ; MESSAGE ADDRESS
4906 026102 000020          16.        ; CHARACTER COUNT
4907 026104 004737 030126          JSR      PC,EOM     ; LOAD AN EOM

```



```

4908 026110 004537 030152 JSR R5,MESLD ;LOAD SILO
4909 026114 030440 FLTDAT ;MESSAGE ADDRESS
4910 026116 000020 16. ;CHARACTER COUNT
4911 026120 004737 030126 JSR PC,EOM ;LOAD AN EOM
4912 026124 004737 026702 JSR PC,OCOR ;WAIT FOR OCOR
4913 026130 005011 CLR (R1) ;CLEAR LINE UNIT LOOP
4914 026132 012700 000003 MOV #3,R0 ;R0 = MESSAGE COUNT
4915 026136 012703 000020 MOV #16.,R3 ;R3= CHARACTER COUNT
4916 026142 012702 030440 MOV #FLTDAT,R2 ;LOAD MESSAGE POINTER IN R2
4917 026146 004737 027510 1$: JSR PC,INRDY ;WAIT FOR INRDY
4918 026152 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4919 026154 021204 021204 ;GET DATA FROM IN SILO
4920 026156 016104 000004 MOV 4(R1),R4 ;PUT CHARACTER IN 'FOUND'
4921 026162 112205 MOVB (R2)+,R5 ;PUT 'EXPECTED' IN R5
4922 026164 120504 CMPB R5,R4 ;IS RECEIVED DATA CORRECT
4923 026166 001401 BEQ 2$ ;BR IF OK
4924 026170 104025 HLT 2$ ;DATA ERROR
4925 026172 2$:
4926 026172 005303 DEC R3 ;DEC CHARACTER COUNT
4927 026174 001364 BNE 1$ ;BR IF NOT DONE THIS MESSAGE
4928 026176 012703 000020 MOV #16.,R3 ;RESET CHARACTER COUNT
4929
4930 ;CHECK TO SEE THAT IN BCC MATCH IS SET
4931 ;AND THAT THE BCC WAS RECEIVED CORRECTLY
4932
4933 026202 004737 027510 JSR PC,INRDY ;WAIT FOR INRDY
4934 026206 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4935 026210 021204 021204 ;GET FIRST HALF OF CRC
4936 026212 116137 000004 001252 MOVB 4(R1),TEMP3 ;PUT IN TEMP3
4937 026220 042737 177400 001252 BIC #177400,TEMP3 ;CLEAR HI BYTE
4938 026226 004737 027510 JSR PC,INRDY ;WAIT FOR INRDY
4939 026232 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4940 026234 021244 021244
4941 026236 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
4942 026242 042704 000376 BIC #376,R4 ;CLEAR UNWANTED BITS
4943 026246 012705 000001 MOV #1,R5 ;PUT 'EXPECTED' IN R5
4944 026252 120504 CMPB R5,R4 ;IS IN BCC MATCH SET?
4945 026254 001401 BEQ 25$
4946 026256 104015 HLT 15 ;IN BCC MATCH ERROR
4947 026260 25$:
4948 026260 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4949 026262 021204 021204 ;GET LAST HALF
4950 026264 116137 000004 001251 MOVB 4(R1),TEMP2+1 ;PUT IN TEMP2
4951 026272 042737 000377 001250 BIC #377,TEMP2 ;CLEAR LO BYTE
4952 026300 053737 001250 001252 BIS TEMP2,TEMP3 ;16 BIT BCC NOW IN TEMP3
4953 026306 023737 027652 001252 CMP CALBCC,TEMP3 ;IS IT CORRECT?
4954 026314 001401 BEQ 4$ ;BR IF OK
4955 026316 104027 HLT 27
4956 026320 012702 030440 4$: MOV #FLTDAT,R2 ;RESET MESSAGE POINTER
4957 026324 005300 DEC R0 ;DECREMENT COUNTER
4958 026326 001307 BNE 1$ ;BR IF NOT DONE
4959 026330 104400 3$: SCOPE ;SCOPE THIS TEST
4960
4961
4962
4963

```

***** TEST 57 *****
*DDCMP CABLE DATA TEST

```

4964      ;*THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
4965      ;*4 SYNCs, 59 DATA CHARACTERS, EOM WITH GARBAGE CHARACTER
4966      ;*THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
4967      ;*RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
4968      ;*LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST
4969      ;*****
4970
4971      ; TEST 57
4972      ;-----
4973 026332 012737 000057 001226 TST57: MOV #57,TSTNO
4974 026340 012737 003364 001216 MOV #.EOP,NEXT
4975
4976 026346 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
4977 ;MASTER CLEAR DMC11
4978 026350 032737 040000 001366 BIT #BIT14,STAT1 ;SKIP TEST IF NO
4979 026356 001533 BEQ 3$ ;LOOPBACK CONNECTOR ON
4980 026360 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
4981 026364 004737 030016 JSR PC,SYNLD ;LOAD 2 SYNCs
4982 026370 004737 030016 JSR PC,SYNLD ;LOAD 2 MORE SYNCs
4983 026374 012737 120001 027650 MOV #CRC16,XPOLY ;LOAD POLYNOMIAL FOR SOFT CRC CALC
4984 026402 005037 026432 CLR 6$ ;CLEAR OLD BCC
4985 026406 012703 000073 MOV #59.,R3 ;CHARACTER COUNT
4986 026412 012702 030434 MOV #MESDAT,R2 ;R2= POINTER
4987 026416 112237 026430 7$: MOVB (R2)+,5$ ;LOAD CHAR FOR SOFT BCC CALC.
4988 026422 004537 027544 JSR R5,SIMBCC ;CALC SOFT BCC
4989 026426 000010 10 ;SHIFT COUNT
4990 026430 000000 0 ;CHARACTER
4991 026432 000000 0 6$: ;OLD BCC
4992 026434 013737 027652 026432 MOV CALBCC,6$ ;LOAD OLD BCC
4993 026442 005303 DEC R3 ;DEC COUNT
4994 026444 001364 BNE 7$ ;BR IF NOT DONE YET
4995 026446 004537 030152 JSR R5,MESLD ;LOAD SILO
4996 026452 030434 MESDAT ;MESSAGE ADDRESS
4997 026454 000073 59. ;CHARACTER COUNT
4998 026456 004737 030126 JSR PC,EOM ;LOAD AN EOM
4999 026462 004737 026702 JSR PC,OCOR ;WAIT FOR OCOR
5000 026466 005011 CLR (R1) ;CLEAR LINE UNIT LOOP
5001 026470 012700 000073 MOV #59.,R0 ;R0= CHARACTER COUNT
5002 026474 012702 030434 MOV #MESDAT,R2 ;LOAD MESSAGE POINTER IN R2
5003 026500 004737 027510 1$: JSR PC,INRDY ;WAIT FOR INRDY
5004 026504 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
5005 026506 021204 021204 ;GET DATA FROM IN SILO
5006 026510 016104 000004 MOV 4(R1),R4 ;PUT CHARACTER IN "FOUND"
5007 026514 112205 MOVB (R2)+,R5 ;PUT "EXPECTED" IN R5
5008 026516 120504 CMPB R5,R4 ;IS RECEIVED DATA CORRECT
5009 026520 001401 BEQ 2$ ;BR IF OK
5010 026522 104025 HLT 25 ;DATA ERROR
5011 026524 2$:
5012 026524 005300 DEC R0 ;DECREMENT COUNTER
5013 026526 001364 BNE 1$ ;BR IF NOT DONE
5014
5015 ;CHECK TO SEE THAT IN BCC MATCH IS SET
5016 ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5017
5018 026530 004737 027510 JSR PC,INRDY ;WAIT FOR INRDY
5019 026534 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304

```

```

5020 026536 021204          021204          ;GET FIRST HALF OF CRC
5021 026540 116137 000004 001252      MOVB      4(R1),TEMP3      ;PUT IN TEMP3
5022 026546 042737 177400 0C1252      BIC      #177400,TEMP3    ;CLEAR HI BYTE
5023 026554 004737 027510          JSR      PC,INRDY         ;WAIT FOR INRDY
5024 026560 104414          ROMCLK           ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5025 026562 021244          021244
5026 026564 016104 000004          MOV      4(R1),R4         ;PUT 'FOUND' IN R4
5027 026570 042704 000376          BIC      #376,R4         ;CLEAR UNWANTED BITS
5028 026574 012705 000001          MOV      #1,R5          ;PUT 'EXPECTED' IN R5
5029 026600 120504          CMPB     R5,R4          ;IS IN BCC MATCH SET?
5030 026602 001401          BEQ      25$            ;
5031 026604 104015          MLT      15             ;IN BCC MATCH ERROR
5032 026606          25$:
5033 026606 104414          ROMCLK           ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5034 026610 021204          021204
5035 026612 116137 000004 001251      MOVB     4(R1),TEMP2+1    ;PUT IN TEMP2
5036 026620 042737 000377 001250      BIC     #377,TEMP2       ;CLEAR LO BYTE
5037 026626 053737 001250 001252      BIS     TEMP2,TEMP3      ;16 BIT BCC NOW IN TEMP3
5038 026634 023737 027652 001252      CMP     CALBCC,TEMP3     ;IS IT CORRECT?
5039 026642 001401          BEQ      3$            ;BR IF OK
5040 026644 104027          MLT      27
5041 026646 104400          3$: SCOPE             ;SCOPE THIS TEST
5042
5043
5044          ;SUBROUTINES
5045          ;-----
5046
5047 026650          GETSI:
5048          ;THIS SUBROUTINE READS LU 17, AND PUTS IT INTO NITC.
5049          ;NITC IS ROTATED LEFT UNTILL THE SI BIT IS IN CARRY
5050
5051 026650 104414          ROMCLK           ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5052 026652 021364          021364
5053 026654 017737 152532 026700      MOV     @DMP04,NITC      ;STORE LU 17
5054 026662 106137 026700          ROLB     NITC
5055 026666 106137 026700          ROLB     NITC
5056 026672 106137 026700          ROLB     NITC           ;PUT SI IN THE CARRY BIT
5057 026676 000207          RTS      PC
5058 026700 000000          NITC: 0
5059
5060
5061 026702          OCOR:
5062          ;THIS SUBROUTINE SPINS ON OCOR
5063
5064 026702 104414          ROMCLK           ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5065 026704 021364          021364
5066 026706 032777 000020 152476      BIT     #BIT4,@DMP04     ;IS OCOR SET?
5067 026714 001772          BEQ     OCOR           ;BR IF NO
5068 026716 000207          RTS      PC           ;OK OCOR IS SET, GO BACK
5069
5070
5071 026720          SYNC:
5072          ;THIS SUBROUTINE LOADS THE SILO WITH THE NUMBER OF SYNC
5073          ;CHARACTERS PASSED TO IT IN THE WORD AFTER THE JSR CALL
5074          ;AND A NON-SYNC CHARACTER (301)
5075

```

```

5076 026720 013637 001246      MOV      @ (SP)+,TEMP1      ;GET COUNT
5077 026724 062746 000002      ADD      #2,-(SP)          ;ADJUST STACK
5078 026730 012761 000026 000004  MOV      #26,4(R1)         ;LOAD PORT4
5079 026736 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5080 026740 122114                122114              ;LOAD SYNC REGISTER
5081 026742 004737 027034 1$:      JSR      PC,OUTRDY        ;WAIT FOR OUTRDY
5082 026746 012761 000001 000004  MOV      #1,4(R1)         ;LOAD PORT4
5083 026754 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5084 026756 122111                122111              ;SET SOM
5085 026760 012761 000026 000004  MOV      #26,4(R1)         ;LOAD PORT4
5086 026766 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5087 026770 122110                122110              ;LOAD OUT DATA
5088 026772 005337 001246      DEC      TEMP1            ;ALL DONE?
5089 026776 001361                BNE      1$              ;BR IF NOT
5090 027000 004737 027034      JSR      PC,OUTRDY        ;WAIT FOR OUTRDY
5091 027004 005061 000004      CLR      4(R1)            ;LOAD PORT4
5092 027010 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5093 027012 122111                122111              ;SET SOM
5094 027014 012761 000301 000004  MOV      #301,4(R1)        ;LOAD PORT4
5095 027022 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5096 027024 122110                122110              ;LOAD OUT DATA
5097 027026 004737 026702      JSR      PC,OCOR          ;WAIT FOR OCOR
5098 027032 000207                RTS      PC
5099
5100
5101 027034      OUTRDY:
5102                ;THIS SUBROUTINE SPINS ON OUT READY
5103
5104 027034 005037 001256      CLR      TEMP5            ;CLEAR TIMER
5105 027040 1$:
5106 027040 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5107 027042 021224                021224              ;PORT4 LUT1
5108 027044 032777 000020 152340  BIT      #BIT4,@DMP04      ;IS OUT RDY SET?
5109 027052 001004                BNE      2$              ;BR IF YES
5110 027054 005237 001256      INC      TEMP5            ;INC TIMER
5111 027060 001367                BNE      1$              ;KEEP CHECKING IF NOT DONE
5112 027062 104036                HLT      36              ;ERROR, OUT READY NOT SET
5113 027064 000207                2$:      RTS      PC
5114
5115
5116 027066      CHAR:
5117                ;THIS SUBROUTINE LOADS THE SILO WITH 3 SYNCs
5118                ;AND THE CHARACTER PASSED TO IT.
5119
5120 027066 013637 001250      MOV      @ (SP)+,TEMP2      ;GET CHARACTER
5121 027072 062746 000002      ADD      #2,-(SP)          ;ADJUST STACK
5122 027076 012737 000003 001246  MOV      #3,TEMP1          ;SET FOR 3 SYNCs
5123 027104 012761 000026 000004  MOV      #26,4(R1)         ;LOAD PORT4
5124 027112 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5125 027114 122114                122114              ;LOAD SYNC REGISTER
5126 027116 004737 027034 1$:      JSR      PC,OUTRDY        ;WAIT FOR OUTRDY
5127 027122 012761 000001 000004  MOV      #1,4(R1)         ;LOAD PORT4
5128 027130 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5129 027132 122111                122111              ;SET SOM
5130 027134 012761 000026 000004  MOV      #26,4(R1)         ;LOAD PORT4
5131 027142 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
  
```

```

5132 027144 122110          122110          ;LOAD OUT DATA
5133 027146 005337 001246    DEC      TEMP1      ;ALL DONE?
5134 027152 001361          BNF      1$         ;BR IF NOT
5135 027154 004737 027034    JSR      PC,OUTRDY  ;WAIT FOR OUTRDY
5136 027160 013761 001250 000004  MOV      TEMP2,4(R1) ;LOAD PORT4
5137 027166 104414          ROMCLK           ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5138 027170 122110          122110          ;LOAD OUT DATA
5139 027172 004737 026702    JSR      PC,OCOR    ;WAIT FOR OCOR
5140 027176 000207          RTS      PC
5141
5142

```

```

5143 027200          CCHARSD:
5144          ;THIS SUBROUTINE LOADS THE SILO WITH THE CHARACTER PASSED TO IT.
5145
5146 027200 013637 001250    MOV      @(SP)+,TEMP2 ;GET CHARACTER
5147 027204 062746 000002    ADD      #2,-(SP)     ;ADJUST STACK
5148 027210 004737 027034    JSR      PC,OUTRDY  ;WAIT FOR OUTRDY
5149 027214 013761 001250 000004  MOV      TEMP2,4(R1) ;LOAD PORT4
5150 027222 104414          ROMCLK           ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5151 027224 122110          122110          ;LOAD OUT DATA
5152 027226 004737 027034    JSR      PC,OUTRDY  ;WAIT FOR OUTRDY
5153 027232 104414          ROMCLK           ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5154 027234 122110          122110          ;LOAD GARBAGE CHAR
5155 027236 004737 026702    JSR      PC,OCOR    ;WAIT FOR OCOR
5156 027242 000207          RTS      PC
5157
5158

```

```

5159 027244          SILOLD:
5160          ;THIS SUBROUTINE FILLS THE OUT SILO
5161          ; WITH A BINARY COUNT PATTERN
5162
5163 027244 012737 000073 001250    MOV      #73,TEMP2   ;LOAD COUNT
5164 027252 005737 027504          TST      SCHAR       ;FIRST TIME HERE?
5165 027256 100470          BMI      4$         ;BR IF BITSTUFF
5166 027260 001032          BNE      2$         ;BR IF NO
5167 027262 062737 000002 001250    ADD      #2,TEMP2   ;ADD 2 TO CHARACTER COUNT
5168 027270 012737 000003 001246    MOV      #3,TEMP1   ;SET FOR 3 SYNC
5169 027276 012761 000026 000004    MOV      #26,4(R1)  ;LOAD PORT4
5170 027304 104414          ROMCLK           ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5171 027306 122114          122114          ;LOAD SYNC REGISTER
5172 027310 004737 027034          1$: JSR      PC,OUTRDY  ;WAIT FOR OUTRDY
5173 027314 012761 000001 000004    MOV      #1,4(R1)   ;LOAD PORT4
5174 027322 104414          ROMCLK           ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5175 027324 122111          122111          ;SET SOM
5176 027326 012761 000026 000004    MOV      #26,4(R1)  ;LOAD PORT4
5177 027334 104414          ROMCLK           ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5178 027336 122110          122110          ;LOAD OUT DATA
5179 027340 005337 001246    DEC      TEMP1      ;ALL DONE?
5180 027344 001361          BNE      1$         ;BR IF NOT
5181 027346 004737 027034          2$: JSR      PC,OUTRDY  ;WAIT FOR OUTRDY
5182 027352 013761 027504 000004    MOV      SCHAR,4(R1) ;LOAD PORT4
5183 027360 104414          ROMCLK           ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5184 027362 122110          122110          ;LOAD OUT DATA
5185 027364 005737 027506          TST      STUFLG     ;BITSTUFF???
5186 027370 001407          BEQ      6$         ;BR IF NO
5187 027372 013737 027504 027404    MOV      SCHAR,5$   ;IT IS SLD SO CHECK BITSTUFFING

```

```

5188 027400 004537 030234          JSR      R5,STFFCL      ;ADD ANY BIT STUFF CLOCK TICKS
5189 027404 000000          5$:      0              ;CHARACTER
5190 027406 000010                   10             ;SHIFT COUNT
5191 027410 005237 027504          6$:      INC      SCHAR      ;NEXT CHARACTER
5192 027414 022737 000400 027504    CMP      #400,SCHAR      ;ALL DONE?
5193 027422 001403          BEQ      3$
5194 027424 005337 001250          DEC      TEMP2          ;DECREMENT COUNT
5195 027430 001346          BNE      2$            ;BR IF NOT DONE
5196 027432 004737 026702          3$:      JSR      PC,OCOR      ;WAIT FOR OCOR
5197 027436 000207          RTS      PC
5198 027440 005037 027504          4$:      CLR      SCHAR      ;START PATTERN AT ZERO
5199 027444 012737 177777 027506    MOV      #-1,STUFLG      ;SET BITSTUFF FLAG
5200 027452 005037 030432          CLR      BITCON         ;CLEAR STUFF COUNT
5201 027456 062737 000002 001250    ADD      #2,TEMP2        ;ADD 2 TO CHARACTER COUNT
5202 027464 012761 000001 000004    MOV      #1,4(R1)       ;SET BIT0 IN PORT4
5203 027472 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5204 027474 122111          122111         ;SET SOM!
5205 027476 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5206 027500 122110          122110         ;LOAD GARBAGE CHAR
5207 027502 000721          BR       2$          ;GO LOAD SILO
5208 027504 000000          SCHAR: 0
5209 027506 000000          STUFLG: 0
5210
5211
5212 027510          INRDY:
5213          ;THIS SUBROUTINE SPINS ON INRDY
5214          ;IF INRDY FAILS TO SET THE DELAY TIMES OUT AND AN
5215          ;ERROR IS REPORTED. FOR BETTER SCOPE LOOPS THIS
5216          ;DELAY CAN BE MADE SHORTER BY ALTERING THE NUMBER
5217          ;INITIALLY LOADED INTO TEMP1, THE SMALLER THE NUMBER
5218          ;THE SHORTER THE DELAY. 0 IS THE LONGEST DELAY.
5219
5220 027510 012737 000000 001246          1$:      MOV      #0,TEMP1      ;SET UP DELAY COUNTER
5221 027516          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5222 027516 104414          021244         ;PORT4 LUI2
5223 027520 021244          BIT      #BIT4,@DMP04    ;IS INRDY SET?
5224 027522 032777 000020 151662    BNE      2$          ;BR IF YES
5225 027530 001004          INC      TEMP1         ;INC DELAY
5226 027532 005237 001246          BNE      1$          ;TRY AGAIN
5227 027536 001367          HLT      37          ;ERROR,NO INRDY
5228 027540 104037          2$:      RTS      PC          ;RETURN
5229 027542 000207
5230
5231
5232 027544          SIMBCC:
5233          ;THIS SUBROUTINE CALCULATES THE CRC USING POLYNOMIAL GIVEN
5234          ;IN XPOLY. THE CORRECT CRC IS RETURNED IN CALBCC, AND THE
5235          ;STATE OF THE LSB OF THE BCC IS RETURNED IN THE C BIT.
5236
5237 027544 010046          MOV      RO,-(SP)      ;SAVE RO ON STACK
5238 027546 012537 001246          MOV      (R5)+,TEMP1   ;TEMP1 = SHIFT COUNT
5239 027552 012537 001250          MOV      (R5)+,TEMP2   ;TEMP2 = CHARACTER
5240 027556 012537 027652          MOV      (R5)+,CALBCC  ;CALBCC = OLD BCC
5241 027562 013700 027652          1$:      MOV      CALBCC,RO   ;PUT OLD BCC IN RO
5242 027566 000241          CLC
5243 027570 006037 027652          ROR      CALBCC        ;SHIFT OLD BCC

```

```

5244 027574 006037 001250          ROR    TEMP2          ;SHIFT CHARACTER
5245 027600 005500                ADC    R0              ;ADD CHAR CARRY TO OLD BCC
5246 027602 006000                ROR    R0              ;PUT BITO TO CARRY BIT
5247 027604 103011                BCC    2$              ;CARRY IS FEEDBACK BIT
5248 027606 013700 027650          MOV    XPOLY,R0        ;IF FEEDBACK = 1
5249 027612 043700 027652          BIC    CALBCC,R0      ;EXCLUSIVLY OR XPOLY TO CALBCC
5250 027616 043737 027650 027652  BIC    XPOLY,CALBCC
5251 027624 050037 027652          BIS    R0,CALBCC
5252 027630 005337 001246          2$: DEC    TEMP1          ;DEC SHIFT COUNT
5253 027634 001352                BNE    1$              ;BR IF NOT DONE
5254 027636 013700 027652          MOV    CALBCC,R0     ;PUT RESULT IN R0
5255 027642 006000                ROR    R0              ;SHIFT BITO TO CARRY
5256 027644 012600                MOV    (SP)+,R0       ;RESTORE R0
5257 027646 000205                RTS    R5              ;RETURN
5258 027650 000000          XPOLY: 0
5259 027652 000000          CALBCC: 0
5260                LRC8=200
5261                CRC16=120001
5262                CRC.CCITT=102010
5263
5264
5265 027654          BCCLD:
5266                ;THIS SUBROUTINE LOADS THE OUT SILO WITH 2 SYNCs
5267                ;WITH SOM SET, AND ONE CHARACTER PASSED TO IT
5268                ;WITH THE SOM BIT CLEAR (ENABLE CRC)
5269
5270 027654 013637 001250          MOV    @ (SP)+,TEMP2  ;GET CHARACTER
5271 027660 062746 000002          ADD    #2,-(SP)       ;ADJUST STACK
5272 027664 012737 000002 001246  MOV    #2,TEMP1        ;SET FOR 2 SYNCs
5273 027672 012761 000026 000004  MOV    #26,4(R1)      ;LOAD PORT4
5274 027700 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5275 027702 122114 122114          ;LOAD SYNC REGISTER
5276 027704 004737 027034          1$: JSR    PC,OUTRDY      ;WAIT FOR OUTRDY
5277 027710 012761 000001 000004  MOV    #1,4(R1)       ;LOAD PORT4
5278 027716 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5279 027720 122111 122111          ;SET SOM
5280 027722 012761 000026 000004  MOV    #26,4(R1)      ;LOAD PORT4
5281 027730 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5282 027732 122110 122110          ;LOAD OUT DATA
5283 027734 005337 001246          DEC    TEMP1          ;ALL DONE?
5284 027740 001361                BNE    1$              ;BR IF NOT
5285 027742 004737 027034          JSR    PC,OUTRDY      ;WAIT FOR OUTRDY
5286 027746 013761 001250 000004  MOV    TEMP2,4(R1)    ;LOAD PORT4
5287 027754 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5288 027756 122110 122110          ;LOAD OUT DATA
5289 027760 004737 026702          JSR    PC,OCOR        ;WAIT FOR OCOR
5290 027764 000207                RTS    PC
5291
5292
5293 027766          GETQO:
5294                ;THIS SUBROUTINE READS THE STATE OF THE TRANSMIT
5295                ;BCC LSB AND PUTS IT IN THE CARRY BIT
5296
5297 027766 104414                ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5298 027770 021364 021364          ;PORT4 LU-17
5299 027772 106177 151414          ROLB    @DMP04        ;PUT Q0 IN CARRY

```

```

5300 027776 000207          RTS      PC          ;RETURN
5301
5302
5303 030000          GETQI:
5304          ;THIS SUBROUTINE READS THE STATE OF THE RELEIVE
5305          ;BCC LSB AND PUTS IT IN THE CARRY BIT
5306
5307 030000 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5308 030002 021364          021364          ;PORT4 LU-17
5309 030004 106177 151402  ROLB      @DMPO4      ;PUT Q0 IN CARRY
5310 030010 106177 151376  ROLB      @DMPO4      ;PUT Q1 IN CARRY
5311 030014 000207          RTS      PC          ;RETURN
5312
5313
5314 030016          SYNLD:
5315          ;THIS SUBROUTINE LOADS OUT SILO WITH
5316          ;2 SYNC CHARACTERS WITH SOM SET
5317
5318 030016 012737 000002 001246  MOV      #2,TEMP1      ;LOAD COUNTER FOR 2 SYNCs
5319 030024 012761 000026 000004  MOV      #26,4(R1)     ;PORT4 26
5320 030032 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5321 030034 122114          122114          ;LOAD SYNC REG
5322 030036 004737 027034 1$:      JSR      PC,OUTRDY     ;WAIT FOR OUTRDY
5323 030042 012761 000001 000004  MOV      #1,4(R1)     ;LOAD PORT4
5324 030050 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5325 030052 122111          122111          ;SET SOM
5326 030054 012761 000026 000004  MOV      #26,4(R1)     ;PORT 26
5327 030062 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5328 030064 122110          122110          ;LOAD OUT DATA WITH SYNC
5329 030066 005337 001246          DEC      TEMP1        ;DECREMENT COUNTER
5330 030072 001361          BNE     1$           ;BR IF NOT DONE
5331 030074 000207          RTS      PC          ;RETURN
5332
5333
5334 030076          SOM:
5335          ;THIS SUBROUTINE LOADS SOM AND OUT DATA WITH A
5336          ;GARBAGE CHARACTER (0)
5337
5338 030076 004737 027034          JSR      PC,OUTRDY     ;WAIT FOR OUTRDY
5339 030102 012761 000001 000004  MOV      #1,4(R1)     ;PORT4 1
5340 030110 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5341 030112 122111          122111          ;SET SOM
5342 030114 005061 000004          CLR      4(R1)        ;CLEAR DATA CHAR
5343 030120 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5344 030122 122110          122110          ;LOAD GARBAGE CHARACTER
5345 030124 000207          RTS      PC          ;RETURN
5346
5347
5348 030126          EOM:
5349          ;THIS SUBROUTINE LOADS EOM AND OUT DATA WITH A
5350          ;GARBAGE CHARACTER (2) TO ENABLE TRANSMISSION OF BCC
5351
5352 030126 004737 027034          JSR      PC,OUTRDY     ;WAIT FOR OUTRDY
5353 030132 012761 000002 000004  MOV      #2,4(R1)     ;PORT4 2
5354 030140 104414          ROMC_K          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5355 030142 122111          122111          ;SET EOM

```



```

5356 030144 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5357 030146 122110      122110      ;LOAD GARBAGE CHARACTER
5358 030150 000207      RTS         PC      ;RETURN
5359
5360
5361 030152      MESLD:
5362      ;THIS SUBROUTINE LOADS SILO WITH MESSAGE
5363      ;THE FIRST ARGUMENT IS THE ADDRESS OF THE MESSAGE
5364      ;THE SECOND ARGUMENT IS THE NUMBER OF CHARACTERS IN THE MESSAGE
5365
5366 030152 010046      MOV         RO,-(SP)      ;SAVE RO
5367 030154 012500      MOV         (R5)+,RO      ;RO=MESSAGE POINTER
5368 030156 012537 001246      MOV         (R5)+,TEMP1    ;TEMP1=CHARACTER COUNT
5369 030162 004737 027034      1$:      JSR         PC,OUTRDY      ;WAIT FOR OUT RDY
5370 030166 112061 000004      MOVVB      (R0)+,4(R1)    ;LOAD PORT4 WITH CHARACTER
5371 030172 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5372 030174 122110      122110      ;LOAD OUT DATA SILO
5373 030176 005337 001246      DEC         TEMP1        ;DEC CHAR COUNT
5374 030202 001367      BNE        1$           ;BR IF NOT DONE
5375 030204 004737 026702      JSR         PC,OCOR      ;WAIT FOR OCOR
5376 030210 012600      MOV         (SP)+,RO      ;RESTORE RO
5377 030212 000205      RTS         R5          ;RETURN
5378
5379
5380 030214      CLRIO:
5381      ;THIS SUBROUTINE SETS IN CLR AND OUT CLR TO
5382      ;CLEAR THE TRANSMIT AND RECEIVE BCC REGISTERS
5383
5384 030214 012761 000200 000004      MOV         #BIT7,4(R1)    ;LOAD PORT4
5385 030222 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5386 030224 122112      122112      ;SET IN CLR!
5387 030226 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5388 030230 122111      122111      ;SET OUT CLR!
5389 030232 000207      RTS         PC          ;RETURN
5390
5391
5392 030234      STFFCL:
5393      ;THIS SUBROUTINE ADDS ANY NECESSARY BIT STUFF CLOCK TICKS
5394      ;FIRST ARGUMENT IS CHAR, SECOND ARGUMENT IS SHIFT COUNT.
5395
5396 030234 010046      MOV         RO,-(SP)      ;SAVE RO
5397 030236 012500      MOV         (R5)+,RO      ;PUT CHAR IN RO
5398 030240 012537 001252      MOV         (R5)+,TEMP3    ;PUT SHIFT COUNT IN TEMP3
5399 030244 106000      1$:      RORB        RO           ;LOOK AT NEXT BIT
5400 030246 103403      BCS        2$           ;BR IF A MARK
5401 030250 005037 030432      CLR         BITCON       ;IT WAS A SPACE, CLEAR 1'S COUNTER
5402 030254 000412      BR         3$           ;CONTINUE
5403 030256 005237 030432      2$:      INC         BITCON       ;INC CONSECUTIVE 1'S COUNTER
5404 030262 022737 000005 030432      CMP         #5,BITCON     ;IS IT 5 YET?
5405 030270 001004      BNE        3$           ;BR IF NO
5406 030272 005037 030432      CLR         BITCON       ;YES! SO START AGAIN
5407 030276 104415 000001      DATACLK, 1             ;GIVE EXTRA TICK TO STUFF ZERO
5408 030302 005337 001252      3$:      DEC         TEMP3        ;DEC SHIFT COUNT
5409 030306 001356      BNE        1$           ;BR IF NOT DONE
5410 030310 012600      MOV         (SP)+,RO      ;RESTORE RO
5411 030312 000205      RTS         R5          ;RETURN

```

```

5412
5413
5414 030314          STFFCK:
5415                ;THIS SUBROUTINE CHECKS TO SEE IF TRANSMITTER
5416                ;IS STUFFING ZEROS WHEN IT SHOULD. FIRST ARGUMENT
5417                ;IS THE CHARACTER, SECOND ARGUMENT IS SHIFT COUNT.
5418
5419 030314 010046     MOV     RO,-(SP)      ;SAVE RO
5420 030316 012500     MOV     (R5)+,RO      ;PUT CHAR IN RO
5421 030320 012537 001252  MOV     (R5)+,TEMP3    ;PUT SHIFT COUNT IN TEMP3
5422 030324 106000     1$:  RORB    RO          ;SHIFT OUT NEXT BIT
5423 030326 103403     BCS     2$          ;BR IF IT IS A MARK
5424 030330 005037 030432  CLR     BITCON      ;IT WAS A SPACE, CLEAR 1'S COUNTER
5425 030334 000416     BR      3$          ;CONTINUE
5426 030336 005237 030432  2$:  INC     BITCON      ;INC CONSECUTIVE 1'S COUNTER
5427 030342 022737 000005 030432  CMP     #5,BITCON    ;5 IN A ROW YET?
5428 030350 001010     BNE     3$          ;BR IF NO
5429 030352 005037 030432  CLR     BITCON      ;YES, SO START OVER
5430 030356 104415 000001  DATACLK, 1 ;EXTRA TICK TO STUFF ZERO
5431 030362 004737 026650  JSR     PC,GETSI    ;LOOK AT WINDOW
5432 030366 103001     BCC     3$          ;IS IT A ZERO, BR IF YES
5433 030370 104030     HLT     30          ;NO, ERROR ZERO WAS NOT STUFFED
5434 030372 005337 001252  3$:  DEC     TEMP3       ;DEC SHIFT COUNT
5435 030376 001352     BNE     1$          ;BR IF NOT DONE
5436 030400 012600     MOV     (SP)+,RO    ;RESTORE RO
5437 030402 000205     RTS     R5          ;RETURN
5438
5439
5440 030404          CTSDLY:
5441                ;THIS SUBROUTINE WASTES TIME UNTIL CTS SETS,
5442                ;BUT HOPEFULLY NOT SO LONG THAT THE SILO RUNS O.T
5443
5444 030404 010046     MOV     RO,-(SP)      ;SAVE RO
5445 030406 012700 000032 150564 1$:  MOV     #32,RO      ;LOAD RO WITH COUNT
5446 030412 027777 150566  CMP     @TKCSR,@TKCSR ;WASTE TIME
5447 030420 005300     DEC     RO          ;DECREMENT COUNTER
5448 030422 001373     BNE     1$          ;DO IT AGAIN IF NOT = 0
5449 030424 012600     MOV     (SP)+,RO    ;RESTORE RO
5450 030426 000207     RTS     PC          ;RETURN
5451
5452
5453 030430 000176     FLAG:  ^B<01111110> ;FLAG CHARACTER
5454 030432 000000     BITCON: 0
5455 030434 000 125 252  MESDAT: .BYTE 0,125,252,377
5456 030437 377
5457 030440 001 002 004  FLTDAT: .BYTE 1,2,4,10,20,40,100,200,376,375,373,367,357,337,277,177
5458 030443 010 020 040
5459 030446 100 200 376
5460 030451 375 373 367
5461 030454 357 337 277
5462 030457 177
5463 030460 100 140 160  STUFDT: .BYTE 100,140,160,170,3,300,174,176,177,1
5464 030463 170 003 300
5465 030466 174 176 177
5466 030471 001
5467 030472 363 347 317 .BYTE 363,347,317,200,0,377,377,377,200,37

```

5468	030475	200	000	377			
5469	030500	377	377	200			
5470	030503	037					
5471							
5472	030504	046377	047111	020105	.EVEN		
	030542	046377	047111	020105	EM1:	.ASCIZ	<377>/LINE UNIT INITIALIZATION TEST/
	030605	377	044514	042516	EM2:	.ASCIZ	<377>^LINE UNIT REGISTER READ/ONLY TEST^
	030651	377	044514	042516	EM3:	.ASCIZ	<377>^LINE UNIT REGISTER WRITE/READ TEST^
	030713	377	051124	047101	EM4:	.ASCIZ	<377>/LINE UNIT INTERNAL CLOCK FAILURE/
	030743	377	042522	042503	EM5:	.ASCIZ	<377>/TRANSMITTER DATA ERROR/
	030762	051377	041505	044505	EM6:	.ASCIZ	<377>/RECEIVER TEST/
	031007	377	047515	042504	EM7:	.ASCIZ	<377>/RECEIVER DATA ERROR/
	031032	052377	044510	020123	EM10:	.ASCII	<377>/MODEM SIGNAL ERROR/
	031112	040777	052125	051517		.ASCII	<377>/THIS ERROR COULD BE CAUSED IF YOU HAVE V.35 AND/
	031174	054777	052517	044040		.ASCII	<377>/AUTOSIZED. YOU MUST MANUALLY ANSWER QUESTIONS IF/
	031226	052377	040522	051516		.ASCII	<377>/YOU HAVE V.35 (DMC11-FA)/
	031255	377	042522	042503	EM11:	.ASCIZ	<377>/TRANSMITTER CRC ERROR/
	031301	377	047111	041040	EM12:	.ASCIZ	<377>/RECEIVER CRC ERROR/
	031341	377	051124	047101	EM13:	.ASCIZ	<377>/IN BCC MATCH ERROR (LU REG 12)/
	031411	377	040503	046102	EM14:	.ASCIZ	<377>/TRANSMITTER FAILED TO GO TO MARK STATE/
	031432	043377	040514	020107	EM15:	.ASCIZ	<377>/CABLE DATA TEST/
	031446	052377	040522	051516	EM16:	.ASCIZ	<377>/FLAG ERROR/
	031512	051777	044527	041524	EM17:	.ASCIZ	<377>/TRANSMITTER FAILED TO STUFF A ZERO/
	031533	377	041101	051117	EM20:	.ASCIZ	<377>/SWITCH PAC TEST/
	031550	052377	040522	051516	EM21:	.ASCIZ	<377>/ABORT ERROR/
	031573	377	040510	043114	EM22:	.ASCIZ	<377>/TRANSMITTER ERROR/
	031615	377	052517	020124	EM23:	.ASCIZ	<377>/HALF DUPLEX TEST/
	031640	044777	020116	042522	EM24:	.ASCIZ	<377>/OUT READY NOT SET/
					EM25:	.ASCIZ	<377>/IN READY NOT SET/
	031662	042777	050130	041505	DH1:	.ASCIZ	<377>/EXPECTED FOUND/
	031703	377	054105	042520	DH2:	.ASCIZ	<377>/EXPECTED FOUND LU-REGISTER/
	031741	377	044103	051101	DH3:	.ASCIZ	<377>/CHARACTER BIT THAT FAILED/
	031777	377	047503	051122	DH4:	.ASCIZ	<377>/CORRECT CRC BIT THAT FAILED/
	032037	377	054105	042520	DH5:	.ASCIZ	<377>/EXPECTED FOUND SHIFT/
	032071	377	054105	042520	DH6:	.ASCIZ	<377>/EXPECTED FOUND CHARACTER SHIFT/
	032137	377	046102	041517	DH7:	.ASCIZ	<377>/BLOCK END NOT SET/
	032162	051377	051524	042040	DH10:	.ASCIZ	<377>/RTS DID NOT CLEAR/
		032206			.EVEN		
	032206	000002			DT1:	2	
	032210	003	007			.BYTE	3,7
	032212	001272				SAVR5	
	032214	003	002			.BYTE	3,2
	032216	001270				SAVR4	
	032220	000003			DT2:	3	
	032222	003	007			.BYTE	3,7
	032224	001272				SAVR5	
	032226	003	010			.BYTE	3,10
	032230	001270				SAVR4	
	032232	003	002			.BYTE	3,2
	032234	001264				SAVR2	
	032236	000002			DT3:	2	
	032240	003	017			.BYTE	3,17
	032242	001272				SAVR5	
	032244	002	002			.BYTE	2,2
	032246	001266				SAVR3	

032250	000002		DT4:	2	
032252	006	021		.BYTE	6,21
032254	027652			CALBCC	
032256	002	002		.BYTE	2,2
032260	001266			SAVR3	
032262	000003		DT5:	3	
032264	001	011		.BYTE	1,11
032266	001300			ZERO	
032270	001	011		.BYTE	1,11
032272	001302			ONE	
032274	002	002		.BYTE	2,2
032276	001260			SAVR0	
032300	000003		DT6:	3	
032302	001	011		.BYTE	1,11
032304	001302			ONE	
032306	001	011		.BYTE	1,11
032310	001300			ZERO	
032312	002	002		.BYTE	2,2
032314	001260			SAVR0	
032316	000004		DT7:	4	
032320	001	011		.BYTE	1,11
032322	001300			ZERO	
032324	001	011		.BYTE	1,11
032326	001302			ONE	
032330	003	007		.BYTE	3,7
032332	001272			SAVR5	
032334	002	001		.BYTE	2,1
032336	001266			SAVR3	
032340	000004		DT10:	4	
032342	001	011		.BYTE	1,11
032344	001302			ONE	
032346	001	011		.BYTE	1,11
032350	001300			ZERO	
032352	003	007		.BYTE	3,7
032354	001272			SAVR5	
032356	002	001		.BYTE	2,1
032360	001266			SAVR3	
032362	000002		DT11:	2	
032364	003	007		.BYTE	3,7
032366	030430			FLAG	
032370	002	002		.BYTE	2,2
032372	001266			SAVR3	
032374	000002		DT12:	2	
032376	006	004		.BYTE	6,4
032400	027652			CALBCC	
032402	006	002		.BYTE	6,2
032404	001252			TEMP3	
032406			.ERRTAB:		
032406	000000			0	
032410	000000			0	
032412	000000			0	
032414	030504		EM1		
032416	031703		DH2	:HLT	1
032420	032220		DT2		
032422	030542		EM2		

CZDME MACY11 30A(1052) 07-JUL-80 13:53 PAGE 106
CZDME.P11 01-JUL-80 15:29 SUBROUTINES

C 0

SEQ 0106

032424 031703 DH2 ;HLT 2

CZDME MACY11 30A(1052) 07-JUL-80 13:53 PAGE 107
CZDME.P11 01-JUL-80 15:29 SUBROUTINES

D 9

SEQ 0107

032426 032220
032430 030605
032432 031703

DT2
EM3
DH2 :MLT 3

032434	032220	DT2		
032436	030651	EM4		
032440	000000	0	:HLT	4
032442	000000	0		
032444	030713	EM5		
032446	031703	DH2	:HLT	5
032450	032220	DT2		
032452	030713	EM5		
032454	031741	DH3	:HLT	6
032456	032236	DT3		
032460	030743	EM6		
032462	031662	DT4	:HLT	7
032464	032206	DT1		
032466	030762	EM7		
032470	031662	DH1	:HLT	10
032472	032206	DT1		
032474	031007	EM10		
032476	031662	DH1	:HLT	11
032500	032206	DT1		
032502	031226	EM11		
032504	032037	DH5	:HLT	12
032506	032262	DT5		
032510	031255	EM12		
032512	032037	DH5	:HLT	13
032514	032262	DT5		
032516	031226	EM11		
032520	031777	DH4	:HLT	14
032522	032250	DT4		
032524	031301	EM13		
032526	031662	DH1	:HLT	15
032530	032206	DT1		
032532	031226	EM11		
032534	032037	DH5	:HLT	16
032536	032300	DT6		
032540	031255	EM12		
032542	032037	DH5	:HLT	17
032544	032300	DT6		
032546	031226	EM11		
032550	032071	DH6	:HLT	20
032552	032316	DT7		
032554	031226	EM11		
032556	032071	DH6	:HLT	21
032560	032340	DT10		
032562	031255	EM12		
032564	032071	DH6	:HLT	22
032566	032316	DT7		
032570	031255	EM12		
032572	032071	DH6	:HLT	23
032574	032340	DT10		
032576	031341	EM14		
032600	000000	0	:HLT	24
032602	000000	0		
032604	031411	EM15		
032606	031662	DH1	:HLT	25
032610	032206	DT1		
032612	031432	EM16		

CZDME MACY11 30A(1052) 07-JUL-80 13:53 PAGE 109
CZDME.P11 01-JUL-80 15:29 SUBROUTINES

SEQ 0109

032614	031741	DH3	:HLT	26
032616	032362	DT11		
032620	031255	EM12		
032622	031662	DH1	:HLT	27
032624	032374	DT12		
032626	031446	EM17		
032630	000000	0	:HLT	30
032632	000000	0		
032634	031512	EM20		
032636	031662	DH1	:HLT	31
032640	032206	DT1		
032642	031533	EM21		
032644	032137	DH7	:HLT	32
032646	000000	0		
032650	031533	EM21		
032652	031741	DH3	:HLT	33
032654	032236	DT3		
032656	031550	EM22		
032660	032162	DH10	:HLT	34
032662	000000	0		
032664	031573	EM23		
032666	031703	DH2	:HLT	35
032670	032220	DT2		
032672	031615	EM24		
032674	000000	0	:HLT	36
032676	000000	0		
032700	031640	EM25		
032702	000000	0	:HLT	37
032704	000000	0		
032706	030743	EM6		
032710	031703	DH2	:HLT	40
032712	032220	DT2		
032714	030713	EM5		
032716	032037	DH5	:HLT	41
032720	032262	DT5		
032722	031301	EM13		
032724	031662	DH1	:HLT	42
032726	032206	DT1		

032730 000001

CORMAX:
.END

CROSS REFERENCE TABLE -- USER SYMBOLS

GETQ1	030000	3697	3701	3772	3776	3847	3851	3922	3926	4043	4047	5303#		
GETQ0	027766	3668	3672	3743	3747	3818	3822	3893	3897	3977	3981	5293#		
GETSI	026650	2843	2847	2894	2898	2945	2949	2996	3000	3055	3059	4116	4120	4137
		4141	4153	4162	4295	4299	4316	4320	4346	4350	4367	4371	4383	4392
		4543	4547	4564	4568	4584	4588	4613	4617	4634	4638	4650	4659	5047#
		5431												
HALTS	005222	1646	1692#											
HILIM	004366	1489*	1516	1534#										
ICOUNT	001222	769#	1406	1411*										
INBUF	007630	1459	1495	1815#										
INCI'AR	010146	1844	1872#											
INIFLG	001324	811#	1120	1140	1147*									
INRDY	027510	3406	3450	3487	3495	3503	4198	4222	4227	4405	4419	4424	4448	4462
		4467	4672	4686	4691	4712	4726	4740	4745	4801	4811	4819	4917	4933
		4938	5003	5018	5023	5212#								
INSTER=	104404	833#	1510											
INSTR =	104403	831#	1942	1994	2007	2016	2120	2129						
INSTR2	004166	1466	1478#											
INTTY	012474	2027	2044	2054	2069	2085	2100	2293#						
KMCM	007330	1250	1800#											
LIMITS	004314	1505	1516#											
LINE	007016	1800#	2121											
LOBITS	004372	1491*	1520	1536#	1537									
LOCK	001220	768#	1410*	1427	1429	1669	2417*	2433*	2459*	2475*	2501*	2524*	2559*	2579*
		3649*	3681*	3724*	3756*	3799*	3831*	3874*	3906*					
LOKFLG	001326	813#												
LOLIM	004364	1488*	1518	1533#										
LPCNT	001224	770#	1405*	1406	1409*									
LRCB =	000200	5260#												
LSTERR	001234	774#	1103*	1345*	1647	1649*	1737*							
LUTYPE=	***** U	611	5232											
MASKX	001244	782#												
MASTEK	006142	1671	1800#											
MCRLF	005672	1444	1567	1667	1668	1676	1800#	1941	2003					
MCSRX	006072	1350	1800#											
MDATA	007672	1597	1607	1817#										
MEMLIM	001304	798#	1313*											
MEPASS	005733	1349	1800#											
MERRPC	006217	1674	1800#											
MERRX	006117	1356	1800#											
MERR2	005760	1800#	2228											
MERR3	006005	1286	1800#											
MESDAT	030434	3610	4088	4094	4188	4193	4263	4269	4273	4329	4404	4447	4510	4516
		4521	4596	4671	4725	4781	4786	4791	4851	4986	4996	5002	5455#	
MESLD	030152	3609	4093	4192	4268	4272	4515	4520	4785	4790	4850	4900	4904	4908
		4995	5361#											
MILK	001322	806#	1097*	1358	1901*	1906*	1910							
MLOCK	006043	1327	1800#											
MNEW	006144	1281	1800#											
MODU	006704	1800#	2068											
MPASSX	006106	1354	1800#											
MPFAIL	005675	1734	1800#											
MQM	005666	1474	1800#	1965	2040	2050	2060	2077	2091	2111				
MR	005755	1336	1800#	1959										
MRESET=	004000	706#												
MSTCLR=	104412	845#	1739	2363	2387	2419	2461	2503	2561	2612	2634	2656	2690	2723

XHEAD	006224	1148	1900#																	
XLOC	003022	1206*	1224*	1227	1258	1271#														
XPASS	003562	1355	1382#																	
XPOLY	027650	3655*	3684*	3730*	3759*	3805*	3834*	3880*	3909*	3955*	4021*	4106*	4285*	4336*						
		4533*	4603*	4888*	4983*	5248	5250	5258#												
XSTATQ	007602	1157	1800#																	
XTSTM	005330	1673	1713#																	
XVEC	003554	1353	1379#																	
ZERO	001300	796#	5472																	
SCOD =	***** U	611																		
SCRAP =	177777	611#	2303#	2306	2309#	2327#	2330	2333#	2350#	2353	2356#	2374#	2377	2380#						
		2405#	2408	2411#	2447#	2450	2453#	2489#	2492	2495#	2547#	2550	2553#	2599#						
		2602	2605#	2621#	2624	2627#	2643#	2646	2649#	2676#	2679	2683#	2709#	2712						
		2716#	2753#	2756	2760#	2810#	2813	2818#	2861#	2864	2869#	2912#	2915	2920#						
		2963#	2966	2971#	3014#	3017	3023#	3075#	3078	3081#	3103#	3106	3109#	3131#						
		3134	3137#	3159#	3162	3165#	3187#	3190	3194#	3236#	3239	3242#	3273#	3276						
		3279#	3310#	3313	3316#	3347#	3350	3353#	3384#	3387	3390#	3423#	3426	3431#						
		3467#	3470	3475#	3512#	3515	3519#	3583#	3586	3589#	3636#	3639	3643#	3711#						
		3714	3718#	3786#	3789	3793#	3861#	3864	3868#	3936#	3939	3942#	4002#	4005						
		4008#	4068#	4071	4076#	4169#	4172	4178#	4240#	4243	4251#	4486#	4489	4498#						
		4764#	4767	4772#	4829#	4832	4835#	4864#	4867	4874#	4960#	4963	4969#							
		733	1122	1368#	1692															
SENDAD	003522	611#	2303	2309	2311	2316#	2327	2333	2335	2340#	2350	2356	2358	2363						
SN =	000057	2364#	2374	2380	2382	2387	2388#	2405	2411	2413	2419	2420#	2447	2453						
		2455	2461	2462#	2489	2495	2497	2503	2504#	2547	2553	2555	2561	2562#						
		2599	2605	2607	2612	2613#	2621	2627	2629	2634	2635#	2643	2649	2651						
		2656	2657#	2676	2683	2685	2690	2691#	2709	2716	2718	2723	2724#	2753						
		2760	2762	2767	2768#	2810	2818	2820	2825	2826#	2861	2869	2871	2876						
		2877#	2912	2920	2922	2927	2928#	2963	2971	2973	2978	2979#	3014	3023						
		3025	3030	3031#	3075	3081	3083	3088	3089#	3103	3109	3111	3116	3117#						
		3131	3137	3139	3144	3145#	3159	3165	3167	3172	3173#	3187	3194	3196						
		3201	3202#	3236	3242	3244	3249	3250#	3273	3279	3281	3286	3287#	3310						
		3316	3318	3323	3324#	3347	3353	3355	3360	3361#	3384	3390	3392	3397						
		3398#	3423	3431	3433	3438	3439#	3467	3475	3477	3482	3483#	3512	3519						
		3521	3526	3527#	3583	3589	3591	3596	3597#	3636	3643	3645	3651	3652#						
		3711	3718	3720	3726	3727#	3786	3793	3795	3801	3802#	3861	3868	3870						
		3876	3877#	3936	3942	3944	3949	3950#	4002	4008	4010	4015	4016#	4068						
		4076	4078	4083	4084#	4169	4178	4180	4185	4186#	4240	4251	4253	4258						
		4259#	4486	4498	4500	4505	4506#	4764	4772	4774	4779	4780#	4829	4835						
		4837	4842	4843#	4864	4874	4876	4881	4882#	4960	4969	4971	4976	4977#						
		5042#																		
SS	- 000061	611#	2314	2316#	2338	2340#	2361	2364#	2385	2388#	2416	2420#	2458	2462#						
		2500	2504#	2558	2562#	2610	2613#	2632	2635#	2654	2657#	2688	2691#	2721						
		2724#	2765	2768#	2823	2826#	2874	2877#	2925	2928#	2976	2979#	3028	3031#						
		3086	3089#	3114	3117#	3142	3145#	3170	3173#	3199	3202#	3247	3250#	3284						
		3287#	3321	3324#	3358	3361#	3395	3398#	3436	3439#	3480	3483#	3524	3527#						
		3594	3597#	3648	3652#	3723	3727#	3798	3802#	3873	3877#	3947	3950#	4013						
		4016#	4081	4084#	4183	4186#	4256	4259#	4503	4506#	4777	4780#	4840	4843#						
		4879	4882#	4974	4977#															
SY	= 000017	611#	817#	825	827#	829#	831#	833#	835#	837#	839#	841#	843#	845#						
		847#	849#	851#	853#	855#														
	= 032730	718#	719	722#	729#	734#	737#	741#	745#	747#	799#	800#	801#	802#						
		882#	887#	889#	890#	891#	892#	894#	895#	896#	897#	899#	900#	901#						
		902#	904#	905#	906#	907#	909#	910#	911#	912#	914#	915#	916#	917#						
		919#	920#	921#	922#	924#	925#	926#	927#	929#	930#	931#	932#	934#						
		935#	936#	937#	939#	940#	941#	942#	944#	945#	946#	947#	949#	950#						

DMEND	611#	1338													
DMFRNT	611#														
HLT	685#	2325	2348	2372	2403	2431	2444	2473	2486	2519	2540	2574	2592	2619	2641
	2665	2674	2706	2740	2750	2788	2798	2807	2845	2849	2859	2896	2900	2910	2947
	2951	2961	2998	3002	3012	3057	3061	3101	3129	3157	3185	3215	3227	3234	3263
	3271	3300	3308	3337	3345	3374	3382	3413	3457	3494	3502	3510	3559	3581	3631
	3670	3674	3699	3703	3745	3749	3774	3778	3820	3824	3849	3853	3895	3899	3924
	3928	3979	3983	4045	4049	4118	4122	4139	4143	4155	4164	4206	4215	4235	4297
	4301	4318	4322	4348	4352	4369	4373	4385	4394	4412	4432	4441	4455	4475	4484
	4545	4549	4566	4570	4586	4590	4615	4619	4636	4640	4652	4661	4679	4699	4708
	4719	4733	4753	4762	4799	4808	4818	4826	4862	4924	4946	4955	5010	5031	5040
	5112	5278	5433												
\$ABORT	611#														
\$AUTO	611#	1160													
\$BCC	611#	3636	3711	3786	3861										
\$BINCR	611#	3936	4002												
\$BINWI	611#	3014													
\$BUFE	611#	1812													
\$CDATA	611#	4864	4960												
\$CLOCK	611#	2643													
\$COMP	611#	3210	3222	3229	3258	3267	3295	3304	3332	3341	3369	3378	3490	3498	3506
	4210	4230	4427	4470	4694	4748	4941	5026							
\$CRC	611#	3653	3682	3728	3757	3803	3832	3878	3907						
\$CRCSH	611#	3950	4016												
\$CYCLE	611#	1884													
\$EMPTY	611#	4764													
\$EOP	611#	1338													
\$FINI	611#	5042													
\$FLAG	611#														
\$FLOAT	611#	2505	2525	2563	2580										
\$GETPA	611#														
\$HALF	611#	4829													
\$HEADE	611#														
\$INACT	611#	3075	3103	3131	3159										
\$INIT	611#														
\$LINE1	611#	2489	2547												
\$LU1	611#	2303	2327	2350	2374										
\$LU12	611#	2405													
\$LU17	611#	2447													
\$MARHI	611#														
\$MARK	611#	3467													
\$MATCH	611#	4219	4415	4458	4682	4736	4929	5014							
\$MOCK	611#														
\$MODEM	611#	3512													
\$MSG	611#	1800													
\$MULT	611#														
\$PATTE	611#	3384	3423												
\$PFAIL	611#	1716													
\$QOOI	611#	5293	5303												
\$QUEST	611#	1994	2007	2016	2120	2129									
\$RAMCL	611#	1744													
\$RCLK	611#	1747	1750	1787	1792	2318	2341	2365	2389	2422	2424	2435	2437	2464	2466
	2477	2479	2509	2511	2530	2532	2566	2568	2584	2586	2613	2635	2659	2668	2693
	2695	2699	2726	2728	2733	2743	2770	2772	2777	2781	2791	2801	2829	2834	2855
	2880	2885	2906	2931	2936	2957	2982	2987	3008	3037	3041	3046	3069	3094	3122
	3150	3178	3208	3218	3220	3256	3265	3293	3302	3330	3339	3367	3376	3407	3445

(CZDME MACY11 3GA(1052) 07-JUL-80 13:53 PAGE 124
CZDME.P11 01-JUL-80 15:29

CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0122

	3451	3488	3496	3504	3533	3536	3561	3564	3603	3607	3614	3958	3962	3966	3996
	4024	4028	4032	4062	4199	4208	4223	4228	4237	4406	4420	4425	4434	4449	4463
	4468	4477	4673	4687	4692	4701	4713	4727	4741	4746	4755	4795	4802	4812	4820
	4846	4855	4918	4934	4939	4948	5004	5019	5024	5033	5051	5064	5079	5083	5086
	5092	5095	5106	5124	5128	5131	5137	5150	5153	5170	5174	5177	5183	5203	5205
	5222	5274	5278	5281	5287	5297	5307	5320	5324	5327	5340	5343	5354	5356	5371
	5385	5387													
\$RCRC	611#	4169													
\$REC	611#	3187	3236	3273	3310	3347									
\$SCOPE	611#	1388													
\$SIMBC	611#	5232													
\$SINAC	611#														
\$SOFTC	611#	1820													
\$STUFF	611#														
\$SUPAC	611#	2599	2621												
\$TCHAR	611#	4087	4187	4262	4509										
\$TCRC	611#	4068	4240	4486											
\$TRANW	611#	4106	4285	4336	4533	4603									
\$TRAN1	611#	2676	2709	2753											
\$TRPDE	611#	825	827	829	831	833	835	837	839	841	843	845	847	849	851
	853														
\$TSTN	611#	2311	2335	2358	2382	2413	2455	2497	2555	2607	2629	2651	2685	2718	2762
	2820	2871	2922	2973	3025	3083	3111	3139	3167	3196	3244	3281	3318	3355	3392
	3433	3477	3521	3591	3645	3720	3795	3870	3944	4010	4078	4180	4253	4500	4774
	4837	4876	4971												
\$VARIA	611#	744													
\$WINDO	611#	2810	2861	2912	2963										
\$XZ	611#	2303	2309	2327	2333	2350	2356	2374	2380	2405	2411	2447	2453	2489	2495
	2547	2553	2599	2605	2621	2627	2643	2649	2676	2683	2709	2716	2753	2760	2810
	2818	2861	2869	2912	2920	2963	2971	3014	3023	3075	3081	3103	3109	3131	3137
	3159	3165	3187	3194	3236	3242	3273	3279	3310	3316	3347	3353	3384	3390	3423
	3431	3467	3475	3512	3519	3583	3589	3636	3643	3711	3718	3786	3793	3861	3868
	3936	3942	4002	4008	4068	4076	4169	4178	4240	4251	4486	4498	4764	4772	4829
	4835	4864	4874	4960	4969										
\$ZEROS	611#														

. ABS. 032730 000

ERRORS DETECTED: 0

CZDME,CZDME.SEQ/SOL/CRF/NL:TOC=(CZDME.P11
RUN-TIME: 15 21 1 SECONDS
RUN-TIME RATIO: 64/39=1.6
CORE USED: 32K (63 PAGES)