

DMC11

BTSTUF MD LN UNT TSTS AH-8557C-MC  
CZDMFCO FICHE 1 OF 1

NOV 1980  
COPYRIGHT © 76-80  
MADE IN USA





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

.REM \$

IDENTIFICATION

PRODUCT CODE: AC-8556C-MC  
PRODUCT NAME: CZDMFCO BTSTUF MD LN UNI TSTS  
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



43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92

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.

CZDMF 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 BITSTUFF MODE. THE MODEM SIGNALS ARE ALSO CHECKED. CZDMF 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. DZDMC [REV] BASIC W/R AND MICRO-PROCESSOR TESTS
2. DZDME [REV] DDCMP 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 VARIE AS EACH IS UPDATED.

2. REQUIREMENTS

2.1 EQUIPMENT

ANY PDP11 FAMILY CPU (EXCEPT AN LSI-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-MD



93  
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

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.

3. LOADING PROCEEDURE

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 (PROGRAM SHOULD NOW BE LOADING INTO CPU)



136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190

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 ONLY 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 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.



191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218

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.



219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261

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.



262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309

#### 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 ENABELED; AND THERE IS A HARD ERROR (CONSTANT); SW08 IS BEST. (SW14=1,0, SW10=0, SW09=0, SW08=1). FOR INTERMITTEMT 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



310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357

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.



358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413

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 CZDMF 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:.



414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469

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) 1302/0000000001000000 MEANS THAT DMC11 NO.06 IS THE DMC11 NOW RUNNING.

DMCROO-DMCR17  
DMST00-DMST17  
(1500)-(1640)

THESE LOCATIONS CONTAIN THE INFORMATION NEEDED TO TEST UP TO 16 (DECIMAL) DMC11S SEQUENTIALY. 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) 1276/0000000000011111 MEANS THAT DMC11 NO. 00,01,02,03,04 WILL BE TESTED. EXAMPLE: (DMACTV) 1276/0000000000010001 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



470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501

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 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 RUN FREE RUNNING TESTS ON; KMC11  
BIT1=0 DMC11-AR (LOW SPEED)  
BIT1=1 DMC11-AL (HIGH SPEED)  
BIT2=0 DMC11-DA  
BIT2=1 DMC11-FA



502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
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

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 A 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 CRAM, A 125252 INDICATES A KMC11 WITH CRAM, A 626 INDICATES A DMC11-AL AND A 16520 INDICATES 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 INTERUPT OCCURES 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 INTERUPT OCCURED; THE ADDRESS TO WHICH THE DMC11 INTERUPTED 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 DOES 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:



558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598

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 SSR FOR THE STARTING SWITCHES, FIRST LOAD THE DIAGNOSTIC, THEN HIT CTRL G, THEN START 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.

\$



0001

CZDMF MACY11 30A(1052) 08-JUL-80 08:26 PAGE 15  
CZDMF.P11 08-JUL-80 08:25

B 2

SEQ 0014

599  
600



```

601           ;          .NLIST
602
603
604
605
606
607           ;*CZDMFCO BTSTUF MD LN UNT TSTS   DMC11 DDCMP LINE UNIT TESTS
608           ;*COPYRIGHT 1976, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
609           ;-----
610
611           ;STARTING PROCEDURE
612           ;LOAD PROGRAM
613           ;LOAD ADDRESS 000200
614           ;SWR=0  AUTOSIZE DMC11
615           ;SW07=1  USE CURRENT DMC11 PARAMETERS
616           ;SW00=1  INPUT NEW DMC11 PARAMETERS
617           ;PRESS START
618           ;PROGRAM WILL TYPE 'CZDMFCO BTSTUF MD LN UNT TSTS   DMC11 DDCMP LINE UNIT TESTS'
619           ;PROGRAM WILL TYPE STATUS MAP
620           ;PROGRAM WILL TYPE 'R' TO INDICATE THAT TESTING HAS STARTED
621           ;AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE
622           ;AND THEN RESUME TESTING
623           ;SUBSEQUENT RESTARTS WILL NOT TYPE PROGRAM TITLE
624
625
626
627
628           ;SWITCH REGISTER OPTIONS
629           ;-----
630
631           100000      SW15=100000      :=1,HALT ON ERROR
632           040000      SW14=40000       :=1,LOOP ON CURRENT TEST
633           020000      SW13=20000       :=1,INHIBIT ERROR TYPEOUT
634           010000      SW12=10000       :=1,DELETE TYPEOUT/BELL ON ERROR.
635           004000      SW11=4000        :=1,INHIBIT ITERATIONS
636           002000      SW10=2000        :=1,ESCAPE TO NEXT TEST ON ERROR
637           001000      SW09=1000        :=1,LOOP WITH CURRENT DATA
638           000400      SW08=400         :=1,LOOP ON ERROR
639           000200      SW07=200         :=1,USE CURRENT DMC11 PARAMETERS, =0,AUTOSIZE DMC11
640           000100      SW06=100         :=1,HALT BEFORE CLOCKING MICRO-PROCESSOR INSTRUCTION
641           000040      SW05=40
642           000020      SW04=20
643           000010      SW03=10          ;RESELECT DMC11'S TO BE TESTED (ACTIVE)
644           000004      SW02=4          ;LOCK ON TEST SELECT
645           000002      SW01=2          ;RESTART PROGRAM AT SELECTED TEST
646           000001      SW00=1          ;INPUT DMC11 PARAMETERS

```



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

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



0004

TRAPCATCHER FOR UNEXPECTED INTERRUPTS

SEQ 0017

```

699
700
701
702
703
704
705
706
707
708
709          000000
710
711
712
713          000024
714 000024 005336
715 000026 000340
716 000030 004750
717 000032 000340
718 000034 004716
719 000036 000340
720
721 000040 000000
722 000042 000000
723 000044 000000
724 000046 003522
725
726 000052 000000
727
728          000174
729 000174 000000
730 000176 000000
731
732          000200
733 000200 000137 002002
734
735
736          001000
737 001000 005377 055103 046504
(2) 001040 046504 030503 020061
(2)
738          001200
739
740
741
742
743 001200 177570
744 001202 177570

```

```

:*****
:-----
:TRAPCATCHER FOR ILLEGAL INTERRUPTS
:THE STANDARD "TRAP CATCHER" IS PLACED
:BETWEEN 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
$ENDAD         ;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>/CZDMFCO BTSTUF MD LN UNT TSTS/<377>
        .ASCII /DMC11 DDCMP LINE UNIT TESTS/<377>

.=1200
:INDIRECT POINTERS TO SWITCH REGISTER AND LIGHT DISPLAY
:-----

DISPLAY:177570
SWR: 177570

```



```

745
746 ;INDIRECT POINTERS TO TELETYPE VECTORS AND REGISTERS
747 ;-----
748
749 001204 177560 TKCSR: 177560 ;TELETYPE KEYBOARD CONTROL REGISTER
750 001206 177562 TKDBR: 177562 ;TELETYPE KEYBOARD DATA BUFFER
751 001210 177564 TPCSR: 177564 ;TELEPRINTER CONTROL REGISTER
752 001212 177566 TPDBR: 177566 ;TELEPRINTER DATA BUFFER
753
754 ;PROGRAM CONTROL PARAMETERS
755 ;-----
756
757 001214 000000 RETURN: 0 ;SCOPE ADDRESS FOR LOOP ON TEST
758 001216 000000 NEXT: 0 ;ADDRESS OF NEXT TEST TO BE EXECUTED
759 001220 000000 LOCK: 0 ;ADDRESS FOR LOCK ON CURRENT DATA
760 001222 000003 ICOUNT: 3 ;NUMBER OF ITERATIONS THAT CURRENT TEST WILL BE EXECUTED
761 001224 000000 LPCNT: 0 ;NUMBER OF ITERATIONS COMPLETED
762 001226 000000 TSTNO: 0 ;NUMBER OF TEST IN PROGRESS
763 001230 000000 PASCNT: 0 ;NUMBER OF PASSES COMPLETED
764 001232 000000 ERRCNT: 0 ;TOTAL NUMBER OF ERRORS
765 001234 000000 LSTERR: 0 ;PC OF LAST ERROR CALL
766
767 ;PROGRAM VARIABLES
768 ;-----
769
770 001236 000000 STRTSW: 0 ;SWITCHES AT START OF PROGRAM
771 001240 000000 STAT: 0 ;DM STATUS WORD STORAGE
772 001242 000000 CLKX: 0
773 001244 000000 MASKX: 0
774 001246 000000 TEMP1: 0 ;TEMPORARY STORAGE
775 001250 000000 TEMP2: 0 ;TEMPORARY STORAGE
776 001252 000000 TEMP3: 0 ;TEMPORARY STORAGE
777 001254 000000 TEMP4: 0 ;TEMPORARY STORAGE
778 001256 000000 TEMP5: 0 ;TEMPORARY STORAGE
779 001260 000000 SAVR0: 0 ;R0 STORAGE
780 001262 000000 SAVR1: 0 ;R1 STORAGE
781 001264 000000 SAVR2: 0 ;R2 STORAGE
782 001266 000000 SAVR3: 0 ;R3 STORAGE
783 001270 000000 SAVR4: 0 ;R4 STORAGE
784 001272 000000 SAVR5: 0 ;R5 STORAGE
785 001274 000000 SAVSP: 0 ;STACK POINTER STORAGE
786 001276 000000 SAVPC: 0 ;PROGRAM COUNTER STORAGE
787 001300 000000 ZERO: 0
788 001302 000001 ONE: 1
789 001304 000000 MEMLIM: 0 ;HIGHEST LOCATION FOR NPR'S
790 001306 000001 DMACTV: .BLKW 1 ;DMC11'S SELECTED ACTIVE.
791 001310 000001 DMNUM: .BLKW 1 ;OCTAL NUMBER OF DMC11'S.
792 001312 000001 SAVACT: .BLKW 1 ;ORIGINAL ACTV DEVICES
793 001314 000001 SAVNUM: .BLKW 1 ;WORKABLE NUMBER
794 001316 000000 RUN: 0 ;POINTER TO RUNNING DEVICE.
795 .EVEN
796 001320 001472 CREAM: DM.MAP-6 ;TABLE POINTER.
797 001322 001676 MILK: CNT.MAP-4 ;TABLE POINTER

```







```

849 ;DMC11 CONTROL INDICATORS FOR CURRENT DMC11 UNDER TEST
850 ;-----
851
852 001366 000000 STAT1: 0
853 001370 000000 STAT2: 0
854 001372 000000 STAT3: 0
855
856 ;DMC11 VECTOR AND REGISTER INDIRECT POINTERS
857 ;-----
858
859 001374 000000 DMRVEC: 0 ; POINTER TO DMC11 RECEIVER INTERRUPT VECTOR
860 001376 000000 DMRLVL: 0 ; POINTER TO DMC11 RECEIVER INTERRUPT SERVICE PS
861 001400 000000 DMTRVEC: 0 ; POINTER TO DMC11 TRANSMITTER INTERRUPT VECTOR
862 001402 000000 DMTRLVL: 0 ; POINTER TO DMC11 TRANSMITTER INTERRUPT SERVICE PS
863 001404 000000 DMCSR: 0 ; POINTER TO DMC11 CONTROL STATUS REGISTER
864 001406 000000 DMCSRH: 0 ; POINTER TO DMC11 CONTROL STATUS REGISTER HIGH BYTE.
865 001410 000000 DMCTL: 0 ; POINTER TO DMC11 CONTROL OUT REGISTER
866 001412 000000 DMP04: 0 ; POINTER TO DMC11 PORT REGISTER(SEL 4)
867 001414 000000 DMP06: 0 ; POINTER TO DMC11 PORT REGISTER(SEL 6)
868
869 ;TEMP STORAGE
870 ;-----
871
872 001416 000000 TEMP: 0
873 001460 001460 . = +40
874
875 ;DMC11 STATUS TABLE AND ADDRESS ASSIGNMENTS
876 ;-----
877
878 . = 1500
879 001500 DM.MAP:
880 001500 000001 DMC00: .BLKW 1 ; CONTROL STATUS REGISTER FOR DMC11 NUMBER 00
881 001502 000001 DMS100: .BLKW 1 ; VECTOR FOR DMC11 NUMBER 00
882 001504 000001 DMS200: .BLKW 1 ; DDCMP LINE# FOR DMC11 NUMBER 00
883 001506 000001 DMS300: .BLKW 1 ; 3RD STATUS WORD
884
885 001510 000001 DMC01: .BLKW 1 ; CONTROL STATUS REGISTER FOR DMC11 NUMBER 01
886 001512 000001 DMS101: .BLKW 1 ; VECTOR FOR DMC11 NUMBER 01
887 001514 000001 DMS201: .BLKW 1 ; DDCMP LINE# FOR DMC11 NUMBER 01
888 001516 000001 DMS301: .BLKW 1 ; 3RD STATUS WORD
889
890 001520 000001 DMC02: .BLKW 1 ; CONTROL STATUS REGISTER FOR DMC11 NUMBER 02
891 001522 000001 DMS102: .BLKW 1 ; VECTOR FOR DMC11 NUMBER 02
892 001524 000001 DMS202: .BLKW 1 ; DDCMP LINE# FOR DMC11 NUMBER 02
893 001526 000001 DMS302: .BLKW 1 ; 3RD STATUS WORD
894
895 001530 000001 DMC03: .BLKW 1 ; CONTROL STATUS REGISTER FOR DMC11 NUMBER 03
896 001532 000001 DMS103: .BLKW 1 ; VECTOR FOR DMC11 NUMBER 03
897 001534 000001 DMS203: .BLKW 1 ; DDCMP LINE# FOR DMC11 NUMBER 03
898 001536 000001 DMS303: .BLKW 1 ; 3RD STATUS WORD
899
900 001540 000001 DMC04: .BLKW 1 ; CONTROL STATUS REGISTER FOR DMC11 NUMBER 04
901 001542 000001 DMS104: .BLKW 1 ; VECTOR FOR DMC11 NUMBER 04
902 001544 000001 DMS204: .BLKW 1 ; DDCMP LINE# FOR DMC11 NUMBER 04
903 001546 000001 DMS304: .BLKW 1 ; 3RD STATUS WORD
904

```



PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

905	001550	000001	DMCR05: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 05
906	001552	000001	DMS105: .BLKW	1	:VECTOR FOR DMC11 NUMBER 05
907	001554	000001	DMS205: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 05
908	001556	000001	DMS305: .BLKW	1	:3RD STATUS WORD
909					
910	001560	000001	DMCR06: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 06
911	001562	000001	DMS106: .BLKW	1	:VECTOR FOR DMC11 NUMBER 06
912	001564	000001	DMS206: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 06
913	001566	000001	DMS306: .BLKW	1	:3RD STATUS WORD
914					
915	001570	000001	DMCR07: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 07
916	001572	000001	DMS107: .BLKW	1	:VECTOR FOR DMC11 NUMBER 07
917	001574	000001	DMS207: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 07
918	001576	000001	DMS307: .BLKW	1	:3RD STATUS WORD
919					
920	001600	000001	DMCR10: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 10
921	001602	000001	DMS110: .BLKW	1	:VECTOR FOR DMC11 NUMBER 10
922	001604	000001	DMS210: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 10
923	001606	000001	DMS310: .BLKW	1	:3RD STATUS WORD
924					
925	001610	000001	DMCR11: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 11
926	001612	000001	DMS111: .BLKW	1	:VECTOR FOR DMC11 NUMBER 11
927	001614	000001	DMS211: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 11
928	001616	000001	DMS311: .BLKW	1	:3RD STATUS WORD
929					
930	001620	000001	DMCR12: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 12
931	001622	000001	DMS112: .BLKW	1	:VECTOR FOR DMC11 NUMBER 12
932	001624	000001	DMS212: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 12
933	001626	000001	DMS312: .BLKW	1	:3RD STATUS WORD
934					
935	001630	000001	DMCR13: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 13
936	001632	000001	DMS113: .BLKW	1	:VECTOR FOR DMC11 NUMBER 13
937	001634	000001	DMS213: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 13
938	001636	000001	DMS313: .BLKW	1	:3RD STATUS WORD
939					
940	001640	000001	DMCR14: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 14
941	001642	000001	DMS114: .BLKW	1	:VECTOR FOR DMC11 NUMBER 14
942	001644	000001	DMS214: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 14
943	001646	000001	DMS314: .BLKW	1	:3RD STATUS WORD
944					
945	001650	000001	DMCR15: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 15
946	001652	000001	DMS115: .BLKW	1	:VECTOR FOR DMC11 NUMBER 15
947	001654	000001	DMS215: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 15
948	001656	000001	DMS315: .BLKW	1	:3RD STATUS WORD
949					
950	001660	000001	DMCR16: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 16
951	001662	000001	DMS116: .BLKW	1	:VECTOR FOR DMC11 NUMBER 16
952	001664	000001	DMS216: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 16
953	001666	000001	DMS316: .BLKW	1	:3RD STATUS WORD
954					
955	001670	000001	DMCR17: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 17
956	001672	000001	DMS117: .BLKW	1	:VECTOR FOR DMC11 NUMBER 17
957	001674	000001	DMS217: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 17
958	001676	000001	DMS317: .BLKW	1	:3RD STATUS WORD
959					
960	001700	000000	DM.END: 000000		



```

961
962
963      ;DMC11 PASS COUNT AND ERROR COUNT TABLE
964      ;-----
965      CNT.MAP:
966      001702 000000      PACT00: 0      ;PASS COUNT FOR DMC11 NUMBER 00
967      001704 000000      ERCT00: 0      ;ERROR COUNT FOR DMC11 NUMBER 00
968
969      001706 000000      PACT01: 0      ;PASS COUNT FOR DMC11 NUMBER 01
970      001710 000000      ERCT01: 0      ;ERROR COUNT FOR DMC11 NUMBER 01
971
972      001712 000000      PACT02: 0      ;PASS COUNT FOR DMC11 NUMBER 02
973      001714 000000      ERCT02: 0      ;ERROR COUNT FOR DMC11 NUMBER 02
974
975      001716 000000      PACT03: 0      ;PASS COUNT FOR DMC11 NUMBER 03
976      001720 000000      ERCT03: 0      ;ERROR COUNT FOR DMC11 NUMBER 03
977
978      001722 000000      PACT04: 0      ;PASS COUNT FOR DMC11 NUMBER 04
979      001724 000000      ERCT04: 0      ;ERROR COUNT FOR DMC11 NUMBER 04
980
981      001726 000000      PACT05: 0      ;PASS COUNT FOR DMC11 NUMBER 05
982      001730 000000      ERCT05: 0      ;ERROR COUNT FOR DMC11 NUMBER 05
983
984      001732 000000      PACT06: 0      ;PASS COUNT FOR DMC11 NUMBER 06
985      001734 000000      ERCT06: 0      ;ERROR COUNT FOR DMC11 NUMBER 06
986
987      001736 000000      PACT07: 0      ;PASS COUNT FOR DMC11 NUMBER 07
988      001740 000000      ERCT07: 0      ;ERROR COUNT FOR DMC11 NUMBER 07
989
990      001742 000000      PACT10: 0      ;PASS COUNT FOR DMC11 NUMBER 10
991      001744 000000      ERCT10: 0      ;ERROR COUNT FOR DMC11 NUMBER 10
992
993      001746 000000      PACT11: 0      ;PASS COUNT FOR DMC11 NUMBER 11
994      001750 000000      ERCT11: 0      ;ERROR COUNT FOR DMC11 NUMBER 11
995
996      001752 000000      PACT12: 0      ;PASS COUNT FOR DMC11 NUMBER 12
997      001754 000000      ERCT12: 0      ;ERROR COUNT FOR DMC11 NUMBER 12
998
999      001756 000000      PACT13: 0      ;PASS COUNT FOR DMC11 NUMBER 13
1000     001760 000000      ERCT13: 0      ;ERROR COUNT FOR DMC11 NUMBER 13
1001
1002     001762 000000      PACT14: 0      ;PASS COUNT FOR DMC11 NUMBER 14
1003     001764 000000      ERCT14: 0      ;ERROR COUNT FOR DMC11 NUMBER 14
1004
1005     001766 000000      PACT15: 0      ;PASS COUNT FOR DMC11 NUMBER 15
1006     001770 000000      ERCT15: 0      ;ERROR COUNT FOR DMC11 NUMBER 15
1007
1008     001772 000000      PACT16: 0      ;PASS COUNT FOR DMC11 NUMBER 16
1009     001774 000000      ERCT16: 0      ;ERROR COUNT FOR DMC11 NUMBER 16
1010
1011     001776 000000      PACT17: 0      ;PASS COUNT FOR DMC11 NUMBER 17
1012     002000 000000      ERCT17: 0      ;ERROR COUNT FOR DMC11 NUMBER 17
1013

```



1014

-----  
 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	I	CSR
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	I	STAT1
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	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	I	STAT3
I	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 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 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)



0011

CZDMF MACY11 30A(1052) 08-JUL-80 08:26 PAGE 25  
CZDMF.P11 08-JUL-80 08:25 PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

L 2

SEQ 0024

S



```

1071
1072      ;PROGRAM INITIALIZATION
1073      ;LOCK OUT INTERRUPTS
1074      ;SET UP PROCESSOR STACK
1075      ;SET UP POWER FAIL VECTOR
1076      ;CLEAR PROGRAM CONTROL FLAGS AND COUNTS
1077      ;TYPE TITLE MESSAGE
1078
1079 002002 012737 000340 177776 .START: MOV #340,PS ;LOCK OUT INTERRUPTS
1080 002010 012706 001200 MOV #STACK,SP ;SET UP STACK
1081 002014 012737 005336 000024 MOV #.PFAIL,@#24 ;SET UP POWER FAIL VECTOR
1082 002022 013737 001310 001314 MOV DMNUM,SAVNUM ;SAVE NUMBER OF DEVICES IN SYSTEM.
1083 002030 005037 010144 CLR SWFLG ;CLEAR SOFT TYPEOUT FLAG
1084 002034 105037 001325 CLRB ERRFLG ;CLEAR ERROR FLAG
1085 002040 105037 001327 CLRB QV.FLG ;ZERO QUICK VERIFY FLAG
1086 002044 012737 001470 001320 MOV #DM.MAP-10,CREAM;GET MAP POINTER.
1087 002052 012737 001676 001322 MOV #CNT.MAP-4,MILK ;GET PASS COUNT MAP POINTER
1088 002060 012737 100000 001316 MOV #BIT15,RUN ;POINT POINTER TO FIRST DEVICE.
1089 002066 012700 001702 MOV #CNT.MAP,RO ;PASS COUNT POINTER TO RO
1090 002072 005020 23$: CLR (RO)+ ;CLEAR TABLE
1091 002074 022700 002002 CMP #CNT.MAP+100,RO ;DONE YET?
1092 002100 001374 BNE 23$ ;KEEP GOING
1093 002102 005037 001234 CLR LSTERR ;CLEAR LAST ERROR POINTER
1094 002106 012737 000001 001226 MOV #1,TSTNO ;SET UP FOR TEST 1
1095 002114 012737 002002 001214 MOV #.START,RETURN ;SET UP FOR POWER FAIL BEFORE
1096 ;TESTING STARTS
1097 002122 013746 000006 MOV @#6,-(SP) ;SAVE CURRENT VECTORS
1098 002126 013746 000004 MOV @#4,-(SP) ;
1099 002132 012737 002166 000004 MOV #6$,@#4 ;SET UP FOR TIMEOUT
1100 002140 012737 177570 001202 MOV #177570,SWR ;SET SWR TO HARD SWR ADDRESS
1101 002146 012737 177570 001200 MOV #177570,DISPLAY ;SET DISPLAY TO HARD SWR ADDRESS
1102 002154 022777 177777 177020 CMP #-1,@SWR ;REFERENCE HARDWARE SWITCH REGISTER
1103 002162 001402 BEQ 6$+2 ;IF = -1 USE SOFT SWR ANYWAY
1104 002164 000407 BR 7$ ;IF IT EXISTS AND NOT = -1 USE HARD SWR
1105 002166 022626 6$: CMP (SP)+,(SP)+ ;ADJUST STACK
1106 002170 012737 000176 001202 MOV #SWREG,SWR ;POINTER TO SOFT SWR
1107 002176 012737 000174 001200 MOV #DISPREG,DISPLAY;POINTER TO SOFT DISPLAY REG
1108 002204 012637 000004 7$: MOV (SP)+,@#4 ;RESTORE VECTORS
1109 002210 012637 000006 MOV (SP)+,@#6 ;
1110 002214 105737 001324 TSTB INIFLG ;HAS INITIALIZATION BEEN PERFORMED
1111 002220 001006 BNE 20$ ;BR IF YES
1112 002222 022737 003522 000042 CMP #SENDAD,@#42 ;IF ACT-11 AUTOMATIC MODE, DON'T TYPE ID
1113 002230 001402 BEQ 20$ ;
1114 002232 104402 001000 TYPE ,MTITLE ;TYPE TITLE MESSAGE
1115 002236 004737 007734 20$: JSR PC,CKSWR ;CHECK FOR SOFT SWR
1116 002242 017737 176734 001236 MOV @SWR,STRTSW ;STORE STARTING SWITCHES
1117 002250 005737 000042 TST @#42 ;IS IT RUNNING IN AUTO MODE?
1118 002254 001402 BEQ .+6 ;BR IF NO
1119 002256 005037 001236 CLR STRTSW ;IF YES, CLEAR SWITCHES
1120 002262 032737 000001 001236 BIT #SW00,STRTSW ;IF SW00=1, QUESTIONS ARE ASKED.
1121 002270 001012 BNE 17$ ;BR IF SW00=1
1122 002272 105737 001236 TSTB STRTSW ;BIT7=1??
1123 002276 100007 BPL 17$ ;BR IF SW07=0
1124 002300 005737 001306 TST DMACTV ;ARE ANY DEVICES SELECTED?
1125 002304 001006 BNE 16$ ;BR IF YES
1126 002306 104402 007155 TYPE, NOACT ;NO DEVICES SELECTED.

```



```

1127 002312 000000 HALT ;STOP THE SHOW
1128 002314 000776 BR .-2 ;DISQUALIFY CONTINUE SWITCH
1129 002316 004737 010640 17$: JSR PC,AUTO.SIZE ;GO DO THE AUTO SIZE
1130 002322 105737 001324 16$: TSTB INIFLG ;FIRST TIME?
1131 002324 001410 BEQ 21$ ;BR IF YES
1132 002330 105737 001236 TSTB SRTSW ;IF USING SAME PARAMETERS DONT TYPE MAP
1133 002334 100431 BMI 1$
1134 002336 032737 000006 001236 BIT #BIT1!BIT2,SRTSW ;IS TEST NO. OR LOCK SELECTED
1135 002344 001403 BEQ 24$ ;IF NO THEN TYPE STATUS
1136 002346 000424 BR 1$ ;IF YES DO NOT TYPE STATUS
1137 002350 005137 001324 21$: COM INIFLG ;SET FLAG
1138 002354 104402 006225 24$: TYPE ,XHEAD ;TYPE HEADER
1139 002360 012704 001500 MOV #DM.MAP,R4 ;SET POINTER
1140 002364 010437 001246 5$: MOV R4,TEMP1 ;SET ADDRESS
1141 002370 012437 001250 MOV (R4)+,TEMP2 ;SET CSR
1142 002374 001411 BEQ 1$ ;ALL DONE IF ZERO
1143 002376 012437 001252 MOV (R4)+,TEMP3 ;SET STAT1
1144 002402 012437 001254 MOV (R4)+,TEMP4 ;SET STAT2
1145 002406 012437 001256 MOV (R4)+,TEMP5 ;SET STAT3
1146 002412 104410 CONVRT ;TYPE OUT STATUS MAP
1147 002414 007602 XSTATQ ;
1148 002416 000762 BR 5$ ;
1149 002420 012700 001500 1$: MOV #DM.MAP,R0 ;R0 POINTS TO STATUS TABLE

```

```

1150
1151 ;*****
1152 ;*AUTO SIZE TEST
1153 ;*THIS TEST VERIFYS THAT THE DMC11S AND/OR KMC11S ARE AT THE CORRECT FLOATING
1154 ;*ADDRESSES FOR YOUR SYSTEM. IF THIS TEST FAILS, IT IS NOT A HARDWARE ERROR.
1155 ;*CHECK THE ADDRESSES OF ALL FLOATING DEVICES (DJ,DH,DQ,DU,DUP,LK,DMC,DZ,KMC).
1156 ;*IF THERE ARE NO OTHER FLOATING DEVICES BEFORE THE DMC11, THE FIRST
1157 ;*DMC11 ADDRESS IS 760070, KMC11 IS 760110. NO DEVICE SHOULD EVER BE AT
1158 ;*ADDRESS 760000. THIS TEST MAY REQUIRE 2 OR MORE ATTEMPTS TO GET THE
1159 ;*RIGHT ADDRESSES. AFTER YOU HAVE CHANGED THE ADDRESS TO WHAT IT TOLD
1160 ;*YOU THE FIRST TIME, IT MAY COME BACK AND TELL YOU A DIFFERENT ADDRESS
1161 ;*THE NEXT TIME YOU RUN IT. PLEASE HAVE PATIENCE, THE FINAL ADDRESS
1162 ;*WILL BE CORRECT (AS LONG AS ALL DEVICES IN FRONT OF THE DMC'S ARE
1163 ;*CORRECT).
1164 ;*****
1165

```

```

1166 002424 013746 000004 MOV @#4,-(SP) ;SAVE LOC 4
1167 002430 013746 000006 MOV @#6,-(SP) ;SAVE LOC 6
1168 002434 005037 000006 CLR @#6 ;CLEAR VEC+2
1169 002440 005037 001252 CLR TEMP3 ;CLEAR FLAG
1170 002444 005005 CLR R5 ;R5=0=DMC, R5=-1=KMC
1171 002446 011037 001404 AUSTRT: MOV (R0),DMCSR ;GET NEXT DMC CSR
1172 002452 001564 BEQ AUDONE ;BR IF DONE
1173 002454 005705 TST R5 ;DMC OR KMC?
1174 002456 001005 BNE 1$ ;BR IF KMC
1175 002460 032760 100000 000002 BIT #BIT15,2(R0) ;CHECK FOR DMC CSR
1176 002466 001061 BNE SKIP ;SKIP IF NOT DMC
1177 002470 000404 BR 2$ ;ITS A DMC SO CONTINUE
1178 002472 032760 100000 000002 1$: BIT #BIT15,2(R0) ;CHECK FOR KMC CSR
1179 002500 001454 BEQ SKIP ;SKIP IF NOT KMC
1180 002502 012737 002674 000004 2$: MOV #NODEV,@#4 ;SET UP FOR TIMEOUT
1181 002510 005705 TST R5 ;DMC OR KMC?
1182 002512 001003 BNE 3$ ;BR IF KMC

```



PROGRAM INITIALIZATION AND START UP.

1183	002514	012703	000006			MOV	#6,R3		:R3 IS COUNT OF DEVICES BEFORE DMC
1184	002520	000402				BR	4\$		:GO ON
1185	002522	012703	000010		3\$:	MOV	#10,R3		:R3 IS COUNT OF DEVICES BEFORE KMC
1186	002526	012702	003010		4\$:	MOV	#DEVTAB,R2		:R2 IS DEVICE TABLE PONTER
1187	002532	012701	160010			MOV	#160010,R1		:START WITH ADDRESS 160010
1188	002536	005711			FLOAT:	TST	(R1)		:CHECK ADDRESS IN R1
1189	002540	111204				MOV	(R2),R4		:IF NO TIMEOUT, GET NEXT ADDRESS
1190	002542	060401				ADD	R4,R1		:IN R1
1191	002544	005201				INC	R1		:
1192	002546	040401				BIC	R4,R1		:
1193	002550	005703				TST	R3		:ANY MORE DEVICES TO CHECK FOR?
1194	002552	001371				BNE	FLOAT		:BR IF YES
1195	002554	012737	002700	000004		MOV	#ERR,@#4		:OK ONLY DMC'S ARE LEFT, SET UP FOR TIMEOUT
1196	002562	010137	003022			MOV	R1,XLOC		:SAVE FIRST DMC/KMC ADDRESS
1197	002566	005705			FY:	TST	R5		:DMC OR KMC?
1198	002570	001005				BNE	1\$		:BR IF KMC
1199	002572	032760	100000	000002		BIT	#BIT15,2(R0)		:CHECK FOR DMC CSR
1200	002600	001014				BNE	SKIP		:SKIP IF NOT DMC
1201	002602	000404				BR	2\$		:ITS A DMC SO CONTINUE
1202	002604	032760	100000	000002	1\$:	BIT	#BIT15,2(R0)		:CHECK FOR KMC CSR
1203	002612	001407				BEQ	SKIP		:SKIP IF NOT KMC
1204	002614	005711			2\$:	TST	(R1)		:CHECK DMC ADDRESS
1205	002616	020137	001404			CMP	R1,DMCSR		:DOES IT MATCH
1206	002622	001411				BEQ	OK		:BR IF YES
1207	002624	062701	000010			ADD	#10,R1		:GET NEXT DMC ADDRESS
1208	002630	000756				BR	FY		:DO IT AGAIN
1209	002632	062700	000010		SKIP:	ADD	#10,R0		:SKIP TO NEXT CSR IN TABLE
1210	002636	011037	001404			MOV	(R0),DMCSR		:GET NEXT CSR
1211	002642	001470				BEQ	AUDONE		:BR IF DONE
1212	002644	000750				BR	FY		:ELSE CONTINUE
1213	002646	062700	000010		OK:	ADD	#10,R0		:SKIP TO NEXT DMC CSR
1214	002652	062737	000010	003022		ADD	#10,XLOC		:UPDATE EXPECTED DMC/KMC ADDRESS
1215	002660	011037	001404			MOV	(R0),DMCSR		:GET NEXT DMC/KMC CSR
1216	002664	001457				BEQ	AUDONE		:BR IF DONE
1217	002666	013701	003022			MOV	XLOC,R1		:GET EXPECTED DMC/KMC ADDRESS
1218	002672	000735				BR	FY		:CONTINUE
1219	002674	122243			NODEV:	CMP	(R2)+,-(R3)		:ON TIMEOUT, INC R2, DEC R3
1220	002676	000002				RTI			:RETURN
1221	002700	005737	001252		ERR:	TST	TEMP3		:CHECK FLAG IF = 0 TYPE HEADER
1222	002704	001014				BNE	1\$		:SKIP HEADER
1223	002706	104402				TYPE			:TYPEOUT HEADER MESSAGE
1224	002710	007224				CONERR			:CONFIGURATION ERROR!!!!
1225	002712	012737	002700	001276		MOV	#ERR,SAVPC		:SAVE PC FOR TYPEOUT
1226	002720	104411				CNVRT			:TYPE OUT ERROR PC
1227	002722	002770				ERRPC			:
1228	002724	104402				TYPE			:TYPE REST OF HEADER
1229	002726	007300				CNERR			:
1230	002730	012737	177777	001252		MOV	#-1,TEMP3		:SET FLAG SO IT ONLY GETS TYPED ONCE
1231	002736	010137	001262		1\$:	MOV	R1,SAVR1		:SAVE R1 FOR TYPEOUT
1232	002742	104410				CONVRT			:
1233	002744	002776				CONTAB			:TYPE CSR VALUES
1234	002746	005705				TST	R5		:DMC OR KMC ?
1235	002750	001003				BNE	3\$		:BR IF KMC
1236	002752	104402				TYPE			:
1237	002754	007321				DMCM			:
1238	002756	000402				BR	4\$		:CONTINUE



```

1239 002760 104402      3$:      TYPE
1240 002762 007331      KCMC
1241 002764 022626      4$:      CMP      (SP)+,(SP)+      ;ADJUST STACK
1242 002766 000727      BR      OK      ;BR TO GET OUT
1243 002770 000001      ERRPC:  1
1244 002772      006      002      .BYTE      6,2
1245 002774 001276      SAVPC
1246 002776 000002      CONTAB: 2
1247 003000      006      004      .BYTE      6,4
1248 003002 003022      XLOC
1249 003004      006      002      .BYTE      6,2
1250 003006 001404      DMCSR
1251 003010      007      DEVTAB: .BYTE      7      ;DJ
1252 003011      017      .BYTE      17      ;DH
1253 003012      007      .BYTE      7      ;DQ
1254 003013      007      .BYTE      7      ;DU
1255 003014      007      .BYTE      7      ;DUP
1256 003015      007      .BYTE      7      ;LK
1257 003016      007      .BYTE      7      ;DMC
1258 003017      007      .BYTE      7      ;DZ
1259 003020      007      .BYTE      7      ;KMC
1260      003022      .EVEN
1261 003022 000000      XLOC:  0
1262 003024 005705      AUDONE: TST      R5
1263 003026 001005      BNE      1$      ;DMC?
1264 003030 012705 177777      MOV      #-1,R5      ;BR IF KMC AND ALL DONE
1265 003034 012700 001500      MOV      #DM.MAP,RO      ;SET R5 TO -1 (KMC)
1266 003040 000602      BR      AUSTRT      ;RESET RO TO START OF TABLE
1267 003042 012637 000006      1$:      MOV      (SP)+,@#6      ;GO DO KMC'S
1268 003046 012637 000004      MOV      (SP)+,@#4      ;RESTORE LOC 6
1269 003052 032737 000010 001236      BIT      #SW03,STRTSW      ;RESTORE LOC 4
1270 003060 001422      BEQ      3$      ;SELECT SPECIFIC DEVICES??
1271 003062 104402 006145      TYPE      ,MNEW      ;BR IF
1272 003066 005000      CLR      RO      ;TYPE THE MESSAGE.
1273 003070 000000      HALT      ;ZERO DATA LIGHTS
1274 003072 027737 176104 001312      CMP      @SWR,SAVACT      ;WAIT FOR USER TO TELL WHAT DEVICES TO RUN
1275 003100 101404      BLOS     2$      ;IS THE NUMBER VALID?
1276 003102 104402 006006      TYPE      ,MERR3      ;BR IF NUMBER IS OK.
1277 003106 000000      HALT      ;TELL USER OF INVALID NUMBER.
1278 003110 000776      BR      -2      ;STOP EVERY THING.
1279 003112 017737 176064 001306 2$:      MOV      @SWR,DMACTV      ;RESTART THE PROGRAM AGAIN.
1280 003120 013700 001306      MOV      DMACTV,RO      ;GET NEW DEVICE PATTERN
1281 003124 000000      HALT      ;SHOW THE USER WHAT HE SELECTED.
1282 003126 0127C0 000300      3$:      MOV      #300,RO      ;CONTINUE DYNAMIC SWITCHES.
1283 003132 012701 000302      MOV      #302,R1      ;PREPARE TO CLEAR THE FLOATING
1284 003136 010120      4$:      MOV      R1,(RO)+      ;VECTOR AREA. 300-776
1285 003140 005021      CLR      (R1)+      ;START PUTTING 'PC+2 - HALT'
1286 003142 022021      CMP      (RO)+,(R1)+      ;IN VECTOR AREA.
1287 003144 022700 001000      CMP      #1000,RO      ;POP POINTERS
1288 003150 001372      BNE      4$      ;ALL DONE??
1289      ;BR IF NO.
1290      ;TEST START AND RESTART
1291      ;-----
1292
1293 003152 012706 001200      .BEGIN: MOV      #STACK,SP      ;SET UP STACK
1294 003156 013746 000006      MOV      @#6,-(SP)      ;SAVE LOC 6

```



1295	003162	013746	000004			MOV	@#4,-(SP)	:	SAVE LOC 4
1296	003166	005000				CLR	R0	:	START AT 0
1297	003170	012737	003234	000004		MOV	#2\$,@#4	:	SET UP FOR TIME OUT
1298	003176	005037	000006			CLR	@#6	:	TO AUTOSIZE MEMORY
1299	003202	005720			6\$:	TST	(R0)+	:	CHECK ADDRESS IN R0
1300	003204	022700	157776			CMP	#157776,R0	:	IS IT AT LEAST 28K
1301	003210	001374				BNE	6\$	:	BR IF NO
1302	003212	162700	007776			SUB	#7776,R0	:	SAVE 2K FOR MONITORS
1303	003216	010037	001304		7\$:	MOV	R0,MEMLIM	:	STORE MEMORY LIMIT
1304	003222	012637	000004			MOV	(SP)+,@#4	:	RESTORE LOC 4
1305	003226	012637	000006			MOV	(SP)+,@#6	:	RESTORE LOC 6
1306	003232	000413				BR	10\$	:	CONTINUE
1307	003234	022626			2\$:	CMP	(SP)+,(SP)+	:	ADJUST STACK
1308	003236	162700	000004			SUB	#4,R0	:	GET LAST GOOD ADDRESS
1309	003242	162700	007776			SUB	#7776,R0	:	SAVE 2K FOR MONITORS
1310	003246	022700	030000			CMP	#30000,R0	:	IS IT 8K?
1311	003252	001361				BNE	7\$	:	BR IF NO
1312	003254	012700	037400			MOV	#37400,R0	:	IF 8K DON'T SAVE 2K
1313	003260	000756				BR	7\$	:	
1314	003262	012737	000340	177776	10\$:	MOV	#340,PS	:	LOCK OUT INTERRUPTS
1315	003270	032737	000004	001236		BIT	#BIT2,STRTSW	:	CHECK FOR LOCK ON TEST
1316	003276	001411				BEQ	1\$	:	BR IF NO LOCK DESIRED.
1317	003300	104402	006044			TYPE	,MLOCK	:	TYPE LOCK SELECTED.
1318	003304	012737	000240	003612		MOV	#NOP,TTST	:	ADJUST SCOPE ROUTINE.
1319	003312	012737	000240	003614		MOV	#NOP,TTST+2	:	SET UP TO LOCK
1320	003320	000406				BR	3\$	:	CONTINUE ALONG.
1321	003322	013737	003730	003612	1\$:	MOV	BRW,TTST	:	PREPARE NORMAL SCOPE ROUTINE
1322	003330	013737	003732	003614		MOV	BRX,TTST+2	:	LOCK NOT SELECTED, SET UP FOR NORMAL SCOPE LOOP
1323	003336	012737	010206	001214	3\$:	MOV	#CYCLE,RETURN	:	START AT "CYCLE" FIND WHICH DEVICE TO TEST
1324	003344	032737	000002	001236	4\$:	BIT	#SW01,STRTSW	:	IS TEST NO. SELECTED?
1325	003352	001002				BNE	5\$	:	BR IF YES
1326	003354	104402	005756			TYPE	,MR	:	TYPE R
1327	003360	000177	175630		5\$:	JMP	@RETURN	:	START TESTING



```

1328                                     :END OF PASS
1329                                     :TYPE NAME OF TEST
1330                                     :UPDATE PASS COUNT
1331                                     :CHECK FOR EXIT TO ACT-11
1332                                     :RESTART TEST
1333
1334 003364 000005                       .EOP: RESET                               ;MAKE THE WORLD CLEAN AGAIN.
1335 003366 005037 001234                CLR      LSTERR                               ;CLEAR LAST ERROR PC
1336 003372 105037 001325                CLR      ERRFLG                              ;CLEAR ERROR FLAG
1337 003376 005237 001230                INC      PASCNT                              ;UPDATE PASS COUNT
1338 003402 013777 001230 175570        MOV      PASCNT,@DISPLAY                     ;DISPLAY PASS COUNT
1339 003410 104402 005733                TYPE    ,MEPASS                             ;TYPE END PASS
1340 003414 104402 006073                TYPE    ,MCSRX                              ;TYPE CSR
1341 003420 104411 003546                CNVRT   ,XCSR                               ;SHOW IT
1342 003424 104402 006101                TYPE    ,MVECX                              ;TYPE VECTOR
1343 003430 104411 003554                CNVRT   ,XVEC                               ;SHOW IT
1344 003434 104402 006107                TYPE    ,MPASSX                             ;TYPE PASSES
1345 003440 104411 003562                CNVRT   ,XPASS                              ;SHOW IT
1346 003444 104402 006120                TYPE    ,MERRX                              ;TYPE ERRORS
1347 003450 104411 003570                CNVRT   ,XERR                               ;SHOW IT
1348 003454 013700 001322                MOV      MILK,RO                            ;GET POINTER TO PASS COUNT
1349 003460 013720 001230                MOV      PASCNT,(RO)+                       ;STORE PASS COUNT FOR THIS DMC11
1350 003464 013720 001232                MOV      ERRCNT,(RO)+                       ;STORE ERROR COUNT FOR THIS DMC11
1351 003470 005337 001314                DEC      SAVNUM                              ;ARE ALL DEVICES TESTED?
1352 003474 001017                       BNE     RESTR                               ;BR IF NO.
1353 003476 112737 000377 001327        MOV     #377,QV.FLG                         ;SET THE QUICK VERIFY FLAG.
1354 003504 013737 001310 001314        MOV     DMNUM,SAVNUM                        ;RESTORE THE COUNT
1355 003512 013701 000042                MOV     @#42,R1                            ;CHECK FOR ACT-11 OR DDP
1356 003516 001406                       BEQ     RESTR                               ;IF NOT, CONTINUE TESTING
1357 003520 000005                       RESET                                       ;STOP THE SHOW--CLEAR THE WORLD
1358 003522
1359 003522 004711                       $ENDAD: JSR      PC,(R1)
1360 003524 000240                       NOP
1361 003526 000240                       NOP
1362 003530 000240                       NOP
1363 003532 000240                       NOP
1364 003534 012737 010206 001214        RESTR: MOV     #CYCLE,RETURN
1365 003542 000137 010206                JMP     CYCLE
1366 003546 000001                       XCSR:  1
1367 003550 006 002                       .BYTE  6,2
1368 003552 001404                       DMCSR
1369 003554 000001                       XVEC:  1
1370 003556 004 002                       .BYTE  4,2
1371 003560 001374                       DMRVEC
1372 003562 000001                       XPASS: 1
1373 003564 006 002                       .BYTE  6,2
1374 003566 001230                       PASCNT
1375 003570 000001                       XERR:  1
1376 003572 006 002                       .BYTE  6,2
1377 003574 001232                       ERRCNT
1378
1379                                     ;SCOPE LOOP AND ITERATION HANDLER
1380                                     ;-----
1381
1382 003576 004737 007734                       .SCOPE: JSR     PC,CKSWR                    ;CHECK FOR SOFT SWR
1383 003602 010016                       MOV     RO,(SP)                            ;SAVE RO ON THE STACK

```



```

1384 003604 032777 040000 175370          BIT      #BIT14,@SWR      ;'LOOP ON THIS TEST'?
1385 003612 001407          TTST:  BEQ      1$          ;BR IF NO. (IF LOCK SW01=1; THIS LOC =240)
1386 003614 000437          BR       3$          ;GOTO 3$ (IF LOCK SW01=1; THIS LOC =240)
1387 003616 005737 003734          TST     DONE        ;WAS TKCSR DONE SET?
1388 003622 001434          BEQ     3$          ;BR IF NO (LOCKED ON TEST)
1389 003624 005037 003734          CLR     DONE        ;YES, CLEAR FLAG
1390 003630 000415          BR     2$          ;GO TO NEXT TEST
1391 003632 032777 004000 175342 1$:  BIT     #SW11,@SWR    ;DELETE ITERATION? (QUICK PASS)
1392 003640 001011          BNE     2$          ;BR IF YES
1393 003642 105737 001327          TSTB   QV.FLG       ;HAVE PASSES BEECOMPLETED?
1394 003646 001406          BEQ     2$          ;BR IF QUICK PASS.
1395 003650 005237 001224          INC     LPCNT        ;UPDATE ITERATION COUNTER
1396 003654 023737 001224 001222          CMP     LPCNT,ICOUNT ;ARE ALL ITERATIONS DONE??
1397 003662 101414          BLOS   3$          ;BR IF NOT YET
1398 003664 105037 001325 2$:  CLRB   ERRFLG       ;PREPARE FOR NEW TEST
1399 003670 005037 001224          CLR     LPCNT        ;START ICOUNTER AT 0
1400 003674 005037 001220          CLR     LOCK
1401 003700 012737 000020 001222          MOV     #20,ICOUNT   ;RESET ITERATIONS
1402 003706 013737 001216 001214          MOV     NEXT,RETURN  ;GET NEXT TEST
1403 003714 011600 3$:  MOV     (SP),R0      ;POP R0 OFF OF THE STACK
1404 003716 022626          POP2SP              ;FAKE AN 'RTI'
1405 003720 013701 001404          MOV     DMCSR,R1    ;R1 CONTAINS BASE DMC ADDRESS
1406 003724 000177 175264          JMP     @RETURN      ;GO DO THE TEST
1407 003730 001407          BRW:   1407
1408 003732 000437          BRX:   437
1409 003734 000000          DONE:  0

;CHECK FOR FREEZE ON CURRENT DATA
-----
1414 003736 004737 007734 .SCOPE1: JSR     PC,CKSWR    ;CHECK FOR SOFT SWR
1415 003742 032777 001000 175232          BIT     #SW09,@SWR  ;IS SW09=1(SET)?
1416 003750 001405          BEQ     1$          ;BR IF NOT SET.
1417 003752 005737 001220          TST     LOCK
1418 003756 001402          BEQ     1$
1419 003760 013716 001220          MOV     LOCK,(SP)   ;GOTO THE ADDRESS IN LOCK.
1420 003764 000002 1$:  RTI          ;GO BACK.

;TELETYPE OUTPUT ROUTINE
-----
1425 003766 010546 .TYPE:  MOV     R5,-(SP)   ;SAVE R5 ON THE STACK.
1426 003770 017605          MOV     @2(SP),R5   ;GET ADDRESS OF MESSAGE.
1427 003774 062766 000002 000002          ADD     #2,2(SP)    ;POP OVER ADDRESS.
1428 004002 005737 010144 4$:  TST     SWFLG       ;SOFT SWR MESSAGE?
1429 004006 001004          BNE     1$          ;IF YES TYPE IT OUT REGARDLESS OF SW12
1430 004010 032777 010000 175164          BIT     #SW12,@SWR ;INHIBIT ALL PRINT OUT??
1431 004016 001012          BNE     3$          ;BR IF NO PRINT OUT WANTED (SW12=1)
1432 004020 105715 1$:  TSTB   (R5)         ;IS NUMBER MINUS? (MSB=1(BIT7))
1433 004022 100002          BPL     2$          ;BR IF NUMBER IS PLUS
1434 004024 104402 005672          TYPE   ,MCRLF       ;TYPE A CR/LF!
1435 004030 105777 175154 2$:  TSTB   @TPCSR       ;TTY READY?
1436 004034 100375          BPL     2$          ;BR IF NO.
1437 004036 112577 175150          MOVB   (R5)+,@TPDBR ;PRINT CURRENT CHAR.
1438 004042 001357          BNE     4$          ;IF NOT ZERO KEEP PRINTING!
1439 004044 012605 3$:  MOV     (SP)+,R5    ;END OF OUTPUT. RESTORE R5

```



0019

```

1440 004046 000002          RTI          ;GO HOME
1441                          ;-----
1442
1443 004050 010346          .INSTR: MOV      R3,-(SP)          ;SAVE R3 ON STACK
1444 004052 010446          MOV      R4,-(SP)          ;SAVE R4 ON STACK
1445 004054 017637 000004 004072  MOV      @4(SP),.MSG
1446 004062 062766 000002 000004  ADD      #2,4(SP)
1447 004070 104402          .INST1: TYPE
1448 004072 000000          .MSG: 0
1449 004074 012704 007630  MOV      #INBUF,R4
1450 004100 012703 000007  MOV      #7,R3
1451 004104 105777 175074  1$:  TSTB   @TKCSR
1452 004110 100375          BPL     1$
1453 004112 117714 175070  MOVB   @TKDBR,(R4)
1454 004116 142714 000200  BICB   #200,(R4)
1455 004122 122427 000015  CMPB   (R4)+,#15
1456 004126 001417          BEQ    INSTR2
1457 004130 105777 175054  2$:  TSTB   @TPCSR
1458 004134 100375          BPL     2$
1459 004136 017777 175044 175046  MOV      @TKDBR,@TPDBR
1460 004144 005303          DEC     R3
1461 004146 001356          BNE     1$
1462 004150 012604          MOV     (SP)+,R4
1463 004152 012603          MOV     (SP)+,R3
1464 004154 104402 005666          .INSTE: TYPE ,MQM
1465 004160 010346          MOV     R3,-(SP)
1466 004162 010446          MOV     R4,-(SP)
1467 004164 000741          BR     .INST1
1468 004166 012604          INSTR2: MOV    (SP)+,R4          ;RESTORE R4
1469 004170 012603          MOV    (SP)+,R3          ;RESTORE R3
1470 004172 000002          RTI
1471
1472                          ;CONVERT ASCII STRING TO OCTAL
1473                          ;-----
1474
1475 004174 010546          .PARAM: MOV     R5,-(SP)
1476 004176 010446          MOV     R4,-(SP)
1477 004200 016605 000004  MOV     4(SP),R5
1478 004204 012537 004364  MOV     (R5)+,LOLIM
1479 004210 012537 004366  MOV     (R5)+,HILIM
1480 004214 012537 004370  MOV     (R5)+,DEVADR
1481 004220 112537 004372  MOVB   (R5)+,LOBITS
1482 004224 112537 004373  MOVB   (R5)+,ADRCNT
1483 004230 010566 000004  MOV     R5,4(SP)
1484 004234 005005          PARAM1: CLR    R5
1485 004236 012704 007630  MOV     #INBUF,R4
1486 004242 122714 000015  CMPB   #15,(R4)
1487 004246 001420          BEQ    PARERR
1488 004250 121427 000060  1$:  CMPB   (R4),#60
1489 004254 002415          BLT    PARERR
1490 004256 121427 000067  CMPB   (R4),#67
1491 004262 003012          BGT    PARERR
1492 004264 142714 000060  B!CB   #60,(R4)
1493 004270 152405          BISB   (R4)+,R5
1494 004272 122714 000015  CMPB   #15,(R4)
1495 004276 001406          BEQ    LIMITS

```



1496 004300 006305  
 1497 004302 006305  
 1498 004304 006305  
 1499 004306 000760  
 1500 004310 104404  
 1501 004312 000750  
 1502  
 1503  
 1504  
 1505  
 1506 004314 020537 004366  
 1507 004320 101373  
 1508 004322 020537 004364  
 1509 004326 103770  
 1510 004330 133705 004372  
 1511 004334 001365  
 1512  
 1513  
 1514  
 1515 004336 013704 004370  
 1516 004342 010524  
 1517 004344 062705 000002  
 1518 004350 105337 004373  
 1519 004354 001372  
 1520 004356 012604  
 1521 004360 012605  
 1522 004362 000002  
 1523 004364 000000  
 1524 004366 000000  
 1525 004370 000000  
 1526 004372 000000  
 1527 004373  
 1528  
 1529  
 1530  
 1531  
 1532 004374 016637 000004 001276 .SAV05:  
 1533  
 1534  
 1535  
 1536 004402 010537 001272  
 1537 004406 010437 001270  
 1538 004412 010337 001266  
 1539 004416 010237 001264  
 1540 004422 010137 001262  
 1541 004426 010037 001260  
 1542 004432 000002  
 1543  
 1544  
 1545  
 1546 004434 013700 001260  
 1547 004440 013701 001262  
 1548 004444 013702 001264  
 1549 004450 013703 001266  
 1550 004454 013704 001270  
 1551 004460 013705 001272

```

ASL R5
ASL R5
ASL R5
BR 1$
PARERR: INSTER
BR PARAM1

;TEST TO SEE IF NUMBER IS WITHIN LIMITS
;-----
LIMITS: CMP R5,HILIM
BHI PARERR
CMP R5,LOLIM
BLO PARERR
BITB LOBITS,R5
BNE PARERR

;STORE NUMBER AT SPECIFIED ADDRESS
1$: MOV DEVADR,R4
MOV R5,(R4)+
ADD #2,R5
DECB ADRCNT
BNE 1$
MOV (SP)+,R4
MOV (SP)+,R5
RTI
LOLIM: 0
HILIM: 0
DEVADR: 0
LOBITS: 0
ADRCNT=LOBITS+1

;SAVE PC OF TEST THAT FAILED AND R0-R5
;-----
.SAV05: MOV 4(SP),SAVPC ;SAVE R7 (PC)

;SAVE R0-R5
SV05: MOV R5,SAVR5 ;SAVE R5
MOV R4,SAVR4 ;SAVE R4
MOV R3,SAVR3 ;SAVE R3
MOV R2,SAVR2 ;SAVE R2
MOV R1,SAVR1 ;SAVE R1
MOV R0,SAVR0 ;SAVE R0
RTI ;LEAVE.

;RESTORE R0-R5
.RES05: MOV SAVR0,R0 ;RESTORE R0
MOV SAVR1,R1 ;RESTORE R1
MOV SAVR2,R2 ;RESTORE R2
MOV SAVR3,R3 ;RESTORE R3
MOV SAVR4,R4 ;RESTORE R4
MOV SAVR5,R5 ;RESTORE R5

```



```

1552 004464 000002          RTI          ;LEAVE
1553
1554          ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
1555          ;-----
1556
1557 004466 104402 005672    .CONVR: TYPE      ,MCRLF
1558 004472 010046          .CNVRT: MOV       R0,-(SP)
1559 004474 010146          MOV       R1,-(SP)
1560 004476 010346          MOV       R3,-(SP)
1561 004500 010446          MOV       R4,-(SP)
1562 004502 010546          MOV       R5,-(SP)
1563 004504 017601 000012    MOV       @12(SP),R1
1564 004510 062766 000002 000012    ADD       #2,12(SP)
1565 004516 012137 004710    MOV       (R1)+,WRDCNT
1566 004522 112137 004712    1$: MOVB     (R1)+,CHRCNT
1567 004526 112137 004713    MOVB     (R1)+,SPACNT
1568 004532 013137 004714    MOV       @ (R1)+,BINWRD
1569 004536 122737 000003 004712    CMPB     #3,CHRCNT
1570 004544 001003          BNE      2$
1571 004546 042737 177400 004714    BIC      #177400,BINWRD
1572 004554 013704 004714    2$: MOV      BINWRD,R4
1573 004560 113705 004712    MOVB     CHRCNT,R5
1574 004564 012700 001416    MOV      #TEMP,R0
1575 004570 010403          3$: MOV      R4,R3
1576 004572 042703 177770    BIC      #177770,R3
1577 004576 062703 000060    ADD      #060,R3
1578 004602 110320          MCVB     R3,(R0)+
1579 004604 000241          CLC
1580 004606 006004          ROR      R4
1581 004610 000241          CLC
1582 004612 006004          ROR      R4
1583 004614 000241          CLC
1584 004616 006004          ROR      R4
1585 004620 005305          DEC      R5
1586 004622 001362          BNE      3$
1587 004624 012703 007672    MOV      #MDATA,R3
1588 004630 114023          4$: MOVB     -(R0),(R3)+
1589 004632 105337 004712    DECB    CHRCNT
1590 004636 001374          BNE      4$
1591 004640 105737 004713    TSTB    SPACNT
1592 004644 001405          BEQ      6$
1593 004646 112723 000040    5$: MOVB     #040,(R3)+
1594 004652 105337 004713    DECB    SPACNT
1595 004656 001373          BNE      5$
1596 004660 105013          6$: CLRB     (R3)
1597 004662 104402 007672    TYPE    ,MDATA
1598 004666 005337 004710    DEC     WRDCNT
1599 004672 001313          BNE     1$
1600 004674 012605          MOV     (SP)+,R5
1601 004676 012604          MOV     (SP)+,R4
1602 004700 012603          MOV     (SP)+,R3
1603 004702 012601          MOV     (SP)+,R1
1604 004704 012600          MOV     (SP)+,R0
1605 004706 000002          RTI
1606 004710 000000          WRDCNT: 0
1607 004712 000000          CHRCNT: 0

```



```

1608          004713          SPACNT=CHRCNT+1
1609 004714 000000          BINWRD: 0
1610
1611
1612          ;TRAP DISPATCH SERVICE
1613          ;ARGUMENT OF TRAP IS EXTRACTED
1614          ;AND USED AS OFFSET TO OBTAIN POINTER
1615          ;TO SELECTED SUBROUTINE
1616
1617 004716 011646          .TRPSR: MOV      (SP),-(SP)          ;GET PC OF RETURN
1618 004720 162716 000002          SUB      #2,(SP)          ;=PC OF TRAP
1619 004724 017616 000000          MOV      @ (SP),(SP)          ;GET TRP
1620 004730 006316          TRPOK: ASL      (SP)          ;MULTIPLY TRAP ARG BY 2
1621 004732 042716 177001          BIC      #177001,(SP)          ;CLEAR UNWANTED BITS
1622 004736 062716 001330          ADD      #.TRPTAB,(SP)          ;POINTER TO SUBROUTINE ADDRESS
1623 004742 017616 000000          MOV      @ (SP),(SP)          ;SUBROUTINE ADDRESS
1624 004746 000136          JMP      @ (SP)+          ;GO TO SUBROUTINE
1625
1626          ;ERROR HANDLER
1627          ;-----
1628
1629 004750 004737 007734          .HLT: JSR      PC,CKSWR          ;CHECK FOR SOFT SWR
1630 004754 032777 010000 174220          BIT      #SW12,@SWR          ;BELL ON ERROR?
1631 004762 001406          BEQ      XBX          ;BR IF NO BELL
1632 004764 105777 174220          TSTB    @TPCSR          ;TTY READY.
1633 004770 100003          BPL      XBX          ;DON'T WAIT IF TTY NOT READY.
1634 004772 112777 000207 174212          MOVB    #207,@TPDBR          ;PUSH A BELL AT THE TTY.
1635 005000 032777 020000 174174          XBX: BIT      #SW13,@SWR          ;DELETE ERROR PRINT OUT?
1636 005006 001105          BNE      HALTS          ;BR IF NO PRINT OUT WANTED.
1637 005010 021637 001234          CMP     (SP),LSTERR          ;WAS THIS ERROR FOUND LAST TIME?
1638 005014 001404          BEQ     1$          ;BR IF YES
1639 005016 011637 001234          MOV     (SP),LSTERR          ;RECORD BEING HERE
1640 005022 105037 001325          CLRB   ERRFLG          ;PREPARE HEADER
1641 005026 104406          1$: SAVO5          ;SAVE ALL PROC REGISTERS
1642 005030 011605          MOV     (SP),R5          ;GET THE PC OF ERROR
1643 005032 162705 000002          SUB     #2,R5          ;GET ADDRESS OF TRAP CALL
1644 005036 011504          MOV     (R5),R4          ;GET HLT INSTRUCTION
1645 005040 006304          ASL     R4          ;MULT BY TWO
1646 005042 061504          ADD     (R5),R4          ;DOUBLE IT
1647 005044 006304          ASL     R4          ;MULT AGAIN
1648 005046 042704 177001          BIC     #177001,R4          ;CLEAR JUNK
1649 005052 062704 036070          ADD     #.ERRTAB,R4          ;GET POINTER
1650 005056 012437 005172          MOV     (R4)+,ERRMSG          ;GET ERROR MESSAGE
1651 005062 012437 005204          MOV     (R4)+,DATAHD          ;GET DATA HEADRER
1652 005066 011437 005216          MOV     (R4),DATABP          ;GET DATA TABLE
1653 005072 105737 001325          TSTB   ERRFLG          ;TYPE HEADREER
1654 005076 001403          BEQ     TYPMSG          ;BR IF YES
1655 005100 005737 005216          TST    DATABP          ;DOES DATA TABLE EXIST?
1656 005104 001040          BNE     TYPDAT          ;BR IF YES.
1657 005106 104402 005672          TYPMSG: TYPE   ,MCRLF
1658 005112 104402 005672          TYPE   ,MCRLF
1659 005116 005737 001220          TST    LOCK
1660 005122 001402          BEQ     1$
1661 005124 104402 006143          TYPE   ,MASTEK
1662 005130 104402 006131          1$: TYPE   ,MTSTN
1663 005134 104411 005330          CNVRT  ,XTSTN          ;SHOW IT

```



```

1664 005140 104402 006220          TYPE      ,MERRPC      ;TYPE PC.
1665 005144 104411 005322          CNVRT     ,ERTABO       ;SHOW IT
1666 005150 104402 005672          TYPE      ,MCRLF        ;GIVE A CR/LF
1667 005154 112737 177777 001325    MOV      #~1,ERRFLG ;NO MORE HEADER UNLESS NO DATA TABLE.
1668 005162 005737 005172          TST      ERRMSG    ;IS THERE AN ERROR MESSAGE?
1669 005166 001402          BEQ      WRKO.FM   ;BR IF NO.
1670 005170 104402          TYPE      ;TYPE
1671 005172 000000          ERRMSG: 0        ;
1672 005174          WRKO.FM:        ;      ERROR MESSAGE
1673 005174 005737 005204          TST      DATAHD   ;DATA HEADER?
1674 005200 001402          BEQ      TYPDAT   ;BR IF NO
1675 005202 104402          TYPE      ;TYPE
1676 005204 000000          DATAHD: 0      ;      DATA HEADER
1677 005206 005737 005216          TYPDAT: TST      DATABP ;DATA TABLE?
1678 005212 001402          BEQ      RESREG   ;BR IF NO.
1679 005214 104410          CONVRT ;SHOW
1680 005216 000000          DATABP: 0      ;      DATA TABLE
1681 005220 104407          RESREG: RES05   ;RESTORE PROC REGISTERS
1682 005222 022737 003522 000042    HALTS:  CMP      #$ENDAD,@#42 ;IF ACT-11 AUTOMATIC MODE, HALT!!
1683 005230 001403          BEQ      1$
1684 005232 005777 173744          TST      @SWR
1685 005236 100005          BPL     EXITER   ;HALT ON ERROR?
1686 005240 010046          1$:  PUSHRO   ;BR IF NO HALT ON ERROR
1687 005242 016600 000002          MOV     2(SP),RO ;SAVE RO
1688 005246 000000          HALT    ;SHOW ERROR PC IN DATA LIGHTS
1689 005250 012600          POPRO   ;HALT
1690 005252 005237 001232          EXITER: INC     ERRCNT ;GET RO
1691 005256 032777 000400 173716    BIT     #SW08,@SWR ;UPDATE ERROR COUNT
1692 005264 001007          BNE     1$        ;GOTO TOP OF TEST?
1693 005266 032777 002000 173706    BIT     #SW10,@SWR ;BR IF YES
1694 005274 001411          BEQ     2$        ;GOTO NEXT TEST?
1695 005276 013737 001216 001214    MOV     NEXT,RETURN ;BR IF NO
1696 005304 012706 001200          1$:  MOV     #STACK,SP ;SET FOR NEXT TEST
1697 005310 013701 001404          MOV     DMCSR,R1  ;RESET SP
1698 005314 000177 173674          JMP     @RETURN   ;SET UP R1
1699 005320 000002          2$:  RTI         ;GOTO SPECIFIED TEST
1700 005322 000001          ERTABO: 1        ;RETURN
1701 005324          .BYTE  6,2
1702 005326 001276          SAVPC
1703 005330 000001          XTSTN: 1
1704 005332          .BYTE  3,2
1705 005334 001226          TSTNO
1706          ;ENTER HERE ON POWER FAILURE
1707          ;-----
1708
1709
1710 005336          .PFAIL:
1711 005336 012737 005350 000024    MOV     #RESTART,24 ;SET UP FOR POWER UP TRAP
1712 005344 000000          HALT
1713 005346 000777          BR     .          ;HALT ON POWER DOWN NORMAL
1714
1715          ;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
1716
1717 005350          RESTAR:
1718 005350 012737 005336 000024    MOV     #.PFAIL,24 ;SET UP FOR POWER FAILURE
1719 005356 012706 001200          MOV     #STACK,SP ;RESET THE STACK POINTER

```



```

1720 005362 013701 001404      MOV    DMCSR,R1      ;RESTORE R1
1721 005366 005037 001416      CLR    TEMP          ;READY FOR TIMMER
1722 005372 005237 001416      INC    TEMP          ;PLUS ONE TO THE TIMER!
1723 005376 001375                BNE    -.4           ;BR IF MORE TO GO
1724 005400 104402 005675      TYPE  ,MPFAIL       ;TYPE THE MESSAGE
1725 005404 104411 005430      CNVRT ,PFTAB        ;TELL WHAT TEST TO RETURN TO.
1726 005410 105037 001325      CLR   ERRFLG        ;START CLEAN
1727 005414 005037 001234      CLR   LSTERR        ;*****
1728 005420 005011                CLR   (R1)          ;CLEAR MAINT BITS
1729 005422 104412                MSTCLR ;START CLEAN UP OF DEVICE
1730 005424 000177 173564      JMP   @RETURN       ;START DOING THAT TEST AGAIN.
1731 005430 000001                FCTAB: 1
1732 005432 003 002              .BYTE 3,2
1733 005434 001226                TSTNO
1734
1735 005436                .DELAY:
1736 005436 012777 000020 173746  MOV   #20,@DMP04
1737 005444 104414                ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1738 005446 121111                121111 ;POKE CLOCK DELAY BIT
1739 005450
1740 005450 104414                ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1741 005452 121224                121224 ;PORT4 IBUS*11
1742 005454 032777 000020 173730  BIT   #BIT4,@DMP04 ;IS CLOCK BIT SET?
1743 005462 001772                BEQ   1$            ;BR IF NO
1744 005464 000002                RTI
1745
1746 005466                .MSTCLR:
1747 005466 152777 000100 173712  BISB  #BIT6,@DMCSRH ;SET MASTER CLEAR
1748 005474 142777 000300 173704  BICB  #BIT6!BIT7,@DMCSRH ;CLEAR MASTER CLEAR AND RUN
1749 005502 000002                RTI                ;RETURN
1750
1751 005504                .ROMCLK:
1752 005504 152777 000002 173674  BISB  #BIT1,@DMCSRH ;SET ROMI
1753 005512 013677 173676      MOV   @ (SP)+,@DMP06 ;LOAD INSTRUCTION IN SEL6
1754 005516 062746 000002      ADD  #2,-(SP)        ;ADJUST STACK
1755 005522 032777 000100 173452  BIT   #SW06,@SWR    ;HALT IF SW06 =1
1756 005530 001401                BEQ   1$            ;BR IF SW06 =0
1757 005532 000000                HALT ;HALT BEFORE CLOCKING INSTRUCTION
1758 005534 152777 000003 173644 1$: BISB  #BIT1!BIT0,@DMCSRH ;CLOCK INSTRUCTION
1759 005542 142777 000007 173636  BICB  #BIT2!BIT1!BIT0,@DMCSRH ;CLEAR ROMO, ROMI, STEP
1760 005550 000002                RTI
1761
1762 005552                .DATACLK:
1763 005552 013637 001416      MOV   @ (SP)+,TEMP  ;PUT TICK COUNT IN TEMP
1764 005556 062746 000002      ADD  #2,-(SP)        ;ADJUST STACK
1765 005562 152777 000020 173616 1$: BISB  #BIT4,@DMCSRH ;SET STEP LU
1766 005570 027777 173610 173606  CMP   @DMCSR,@DMCSR ;WASTE TIME
1767 005576 142777 000020 173602  BICB  #BIT4,@DMCSRH ;CLEAR STEP LU
1768 005604 005337 001416      DEC  TEMP           ;DEC TICK COUNT
1769 005610 001364                BNE  1$            ;BR IF NOT DONE
1770 005612 000002                RTI                ;RETURN
1771 005614 000001                3$: .BLKW 1
1772
1773 005616                .TIMER:
1774 005616 013637 001416      MOV   @ (SP)+,TEMP  ;MOVE COUNT TO TEMP
1775 005622 062746 000002      ADD  #2,-(SP)        ;ADJUST STACK

```



```

1776 005626          1$:
1777 005626 104414   ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1778 005630 021364   021364          ;PORT4 IBUS* REG11
1779 005632 032777 000002 173552  BIT #2,@DMP04    ;IS PGM CLOCK BIT CLEAR?
1780 005640 001772   BEQ 1$          ;BR IF YES
1781 005642          2$:
1782 005642 104414   ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1783 005644 021364   021364          ;PORT4 IBUS* REG11
1784 005646 032777 000002 173536  BIT #2,@DMP04    ;IS PGM CLOCK BIT SET?
1785 005654 001372   BNE 2$          ;BR IF YES
1786 005656 005337 001416   DEC TEMP        ;DEC COUNT
1787 005662 001361   BNE 1$          ;BR IF NOT DONE
1788 005664 000002   RTI             ;RETURN
1789
1790 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 CZDMFC /
(2) 005756 051377 000      MR: .ASCIZ <377>/R/
(2) 005761 377 047516 042040  MERR2: .ASCIZ <377>/NO DEVICES PRESENT./
(2) 006006 04777 051516 043125  MERR3: .ASCIZ <377>/INSUFFICIENT DATA!/
(2) 006032 052377 051505 020124  MTSTPC: .ASCIZ <377>/TEST PC-/
(2) 006044 J46377 041517 020113  MLOCK: .ASCIZ <377>/LOCK ON SELECTED TEST/
(2) 006073 103 051123 020072  MCSR: .ASCIZ /CSR: /
(2) 006101 126 041505 020072  MVEC: .ASCIZ /VEC: /
(2) 006107 120 051501 042523  MPASSX: .ASCIZ /PASSES: /
(2) 006120 051105 047522 051522  MERRX: .ASCIZ /ERRORS: /
(2) 006131 124 051505 020124  MTSTN: .ASCIZ /TEST NO: /
(2) 006143 052 000      MASTEK: .ASCIZ /*/
(2) 006145 377 042523 020124  MNEW: .ASCIZ <377>/SET SWITCH REG TO DMC11'S DESIRED ACTIVE./
(2) 006220 041520 020072 000      MERRPC: .ASCIZ /PC: /
(2) 006225 212 020040 020040  XHEAD: .ASCII <212>/
(2) 006264 020377 020040 020040  .ASCII <377>/
(2) 006323 212 020040 041520  .ASCII <212>/ PC CSR STAT1 STAT2 STAT3/
(2) 006375 377 026455 026455  .ASCII <377>/-----/
(2) 006451 377 047510 020127  NUM: .ASCIZ <377>/HOW MANY DMC11'S TO BE TESTED?/
(2) 006511 377 051503 020122  CSR: .ASCIZ <377>/CSR ADDRESS?/
(2) 006527 377 042526 052103  VEC: .ASCIZ <377>/VECTOR ADDRESS?/
(2) 006550 041377 020122 051120  PRIO: .ASCIZ <377>/BR PRIORITY LEVEL? (4,5,6,7)?/
(2) 006607 377 043111 042040  CRAM: .ASCIZ <377>/IF DMC HAS CRAM (M8204) TYPE 'Y', IF CROM (M8200) TYPE 'N' ?/
(2) 006705 377 044127 041511  MODU: .ASCIZ <377>/WHICH LINE UNIT? IF NONE TYPE 'N', IF M8201 TYPE '1', IF M8202 TYP
(2) 007017 377 053523 052111  LINE: .ASCIZ <377>/SWITCH PAC#1 (DDCMP LINE #)?/
(2) 007055 377 053523 052111  BM: .ASCIZ <377>/SWITCH PAC#2 (BM873 BOOT ADD)?/
(2) 007115 377 051511 052040  CONN: .ASCIZ <377>/IS THE LOOP BACK CONNECTOR ON?/
(2) 007155 377 047516 042040  NOACT: .ASCIZ <377>/NO DEVICES ARE SELECTED/
(2) 007206 005377 053523 036522  SWMES: .ASCIZ <377><12>/SWR= /
(2) 007216 042516 037527 000040  SWMES1: .ASCIZ /NEW? /
(2) 007224 177777 046504 030503  CONERR: .ASCIZ <377><377>/DMC11 FOUND AT NON-STANDARD ADDRESS PC: /
(2) 007300 042777 050130 041505  CNERR: .ASCIZ <377>/EXPECTED FOUND/
(2) 007321 040 042050 041515  DMCM: .ASCIZ / (DMC) /
(2) 007331 040 045450 041515  KMCM: .ASCIZ / (KMC) /
(2) 007341 377 046504 030503  SPEED: .ASCIZ <377>/DMC11-AR(REMOTE,LOW SPEED) OR DMC11-AL(LOCAL,HIGH SPEED) TYPE 'R'
(2) 007455 200 044127 041511  MV35: .ASCII <200>/WHICH MODEM TYPE, TYPE 'D' FOR DMC11-DA (RS232C),OR/
(2) 007541 200 054524 042520  .ASCII <200>/TYPE 'F' FOR DMC11-FA (V.35) ? /
(2)
(2)

```

.EVEN



```

(2) 007602 000005          XSTATQ: 5
1791 007604      006      003      .BYTE      6,3
1792 007606 001246          TEMP1
1793 007610      006      003      .BYTE      6,3
1794 007612 001250          TEMP2
1795 007614      006      003      .BYTE      6,3
1796 007616 001252          TEMP3
1797 007620      006      003      .BYTE      6,3
1798 007622 001254          TEMP4
1799 007624      006      002      .BYTE      6,2
1800 007626 001256          TEMP5
1801                                .EVEN
1802
1803                                ;BUFFERS FOR INPUT-OUTPUT
1804
1805 007630 000000          INBUF: 0
1806                                .+.40
1807 007672 000000          MDATA: 0
1808                                .+.40
1809
1810
1811                                ;ROUTINE USED TO CHANGE SOFTWARE SWITCH
1812                                ;REGISTER USING THE CONSOLE TERMINAL
1813                                ;-----
1814
1815 007734 022737 000176 001202 CKSWR: CMP      #SWREG,SWR      ;IS THE SOFT SWR BEING USED?
1816 007742 001077          BNE      CKSWR5      ;BR IF NO
1817 007744 105777 171234          TSTB     @TKCSR      ;IS DONE SET?
1818 007750 100003          BPL      2$          ;GO ON IF NOT SET
1819 007752 012737 177777 003734 MOV      #-1,DONE      ;IF DONE SET, SET FLAG
1820 007760 022777 000007 171220 2$:  CMP      #7,@TKDBR      ;WAS CTRL G TYPED? (7 BIT ASCII)
1821 007766 001404          BEQ      1$          ;BR IF YES
1822 007770 022777 000207 171210  CMP      #207,@TKDBR      ;WAS CTRL G TYPED? (8 BIT ASCII)
1823 007776 001061          BNE      CKSWR5      ;BR IF NO
1824 010000 010246          1$:  MOV      R2,-(SP)      ;STORE R2
1825 010002 010346          MOV      R3,-(SP)      ;STORE R3
1826 010004 010446          MOV      R4,-(SP)      ;STORE R4
1827 010006 012737 177777 010144 MOV      #-1,SWFLG      ;SET SOFT TYPE OUT FLAG
1828 010014 005002          CKSWR1: CLR      R2      ;CLEAR NEW SWR CONTENTS
1829 010016 012704 177777          MOV      #-1,R4      ;SET FLAG TO ALL ONES
1830 010022 104402 007206          TYPE     ,SWMES      ;TYPE "SWR= "
1831 010026 104411          CKSWR2: CNVRT          ;TYPE OUT PRESENT CONTENTS
1832 010030 010200          SOFTSW          ;OF SOFT SWITCH REGISTER
1833 010032 104402 007216          CKSWR3: TYPE     ,SWMES1 ;TYPE "NEW? "
1834 010036 004737 010146          CKSWR4: JSR      PC,INCHAR ;GET RESPONSE
1835 010042 022703 000015          CMP      #15,R3      ;WAS IT A CR?
1836 010046 001424          BEQ      5$          ;BR IF YES
1837 010050 022703 000012          CMP      #12,R3      ;WAS IT A LF?
1838 010054 001416          BEQ      4$          ;BR IF YES
1839 010056 022703 000025          CMP      #25,R3      ;WAS IT CTRL U?
1840 010062 001754          BEQ      CKSWR1      ;BR IF YES(START OVER)
1841 010064 022703 000007          CMP      #7,R3      ;IF CNTL G GET NEXT CHAR
1842 010070 001762          BEQ      CKSWR4
1843 010072 005004          CLR      R4      ;IT MUST BE A DIGIT SO CLR FLAG
1844 010074 042703 177770          BIC      #177770,R3 ;ONLY 0-7 ARE LEGAL SO MASK OFF BITS
1845 010100 006302          ASL      R2      ;SHIFT R2 3 TIMES

```



```

1846 010102 006302          ASL      R2
1847 010104 006302          ASL      R2
1848 010106 050302          BIS      R3,R2          ;ADD LAST DIGIT
1849 010110 000752          BR       CKSWR4         ;GET NEXT CHARACTER
1850 010112 012766 002002 000006 4$:  MOV     #.START,6(SP) ;LF WAS TYPED SO GO TO START
1851 010120 005704          5$:  TST     R4          ;IS FLAG CLEAR?
1852 010122 001002          BNE     6$            ;IF NOT DON'T CHANGE SOFT SWR
1853 010124 010277 171052    MOV     R2,@SWR        ;IF YES THEN WRITE NEW CONTENTS TO SOFT SWR
1854 010130 005037 010144    6$:  CLR     SWFLG        ;CLEAR TYPEOUT FLAG
1855 010134 012604          MOV     (SP)+,R4      ;RESTORE R4
1856 010136 012603          MOV     (SP)+,R3      ;RESTORE R3
1857 010140 012602          MOV     (SP)+,R2      ;RESTORE R2
1858 010142 000207          CKSWR5: RTS          PC          ;RETURN
1859
1860 010144 000000          SWFLG: 0
1861
1862 010146 105777 171032    INCHAR: TSTB @TKCSR
1863 010152 100375          BPL     .-4
1864 010154 017703 171026    MOV     @TKDDBR,R3
1865 010160 105777 171024    TSTB   @TPCSR
1866 010164 100375          BPL     .-4
1867 010166 010377 171020    MOV     R3,@TPDDBR
1868 010172 042703 000200    BIC     #BIT7,R3
1869 010176 000207          RTS     PC
1870
1871 010200 000001          SOFTSW: 1
1872 010202 006          .BYTE 6,2
1873 010204 000176          SWREG

```



```

1874
1875
1876
1877
1878
1879
1880
1881
1882
1883 010206 005737 001306          CYCLE: TST      DMACTV      ;ARE ANY DMC11'S TO BE TESTED?
1884 010212 001004                    BNE      1$      ;BR IF OK.
1885 010214 104402 007155          TYPE     ,NOACT  ;NO DMC11'S SELECTED!!
1886 010220 000000                    HALT                    ;STOP THE SHOW.
1887 010222 000776                    BR      -2        ;DISQUALIFY CONT. SW.
1888 010224 000241          1$: CLC                    ;CLEAR PROC. CARRY BIT.
1889 010226 006137 001316          ROL     RUN      ;UPDATE POINTER
1890 010232 005537 001316          ADC     RUN      ;CATCH CARRY FROM RUN
1891 010236 062737 000004 001322  ADD     #4,MILK   ;UPDATE POINTER
1892 010244 062737 000010 001320  ADD     #10,CREAM ;UPDATE ADDRESS POINTER.
1893 010252 022737 001700 001320  CMP     #DM.MAP+200,CREAM
1894 010260 001006                    BNE     2$      ;KEEP GOING; NOT ALL TESTED FOR.
1895 010262 012737 001500 001320  MOV     #DM.MAP,CREAM ;RESET ADDRESS POINTER.
1896 010270 012737 001702 001322  MOV     #CNT.MAP,MILK ;RESET PASS COUNT POINTER
1897 010276 033737 001316 001306  2$: BIT     RUN,DMACTV ;IS THIS ONE ACTIVE?
1898 010304 001747                    BEQ     1$      ;BR IF NO
1899 010306 013700 001320          MOV     CREAM,R0  ;GET ADDRESS POINTER
1900 010312 013702 001322          MOV     MILK,R2   ;GET PASS COUNT POINTER
1901 010316 012037 001404          MOV     (R0)+,DMCSR ;LOAD SYSTEM CTRL. REG
1902 010322 011037 001374          MOV     (R0),DMRVEC ;LOAD VECTOR
1903 010326 042737 177000 001374  BIC     #177000,DMRVEC ;CLEAR UNWANTED BITS
1904 010334 012037 001366          MOV     (R0)+,STAT1 ;LOAD STAT1
1905 010340 012037 001370          MOV     (R0)+,STAT2 ;LOAD STAT2
1906 010344 012037 001372          MOV     (R0)+,STAT3 ;LOAD STAT3
1907 010350 012237 001230          MOV     (R2)+,PASCNT ;LOAD PASS COUNT
1908 010354 012237 001232          MOV     (R2)+,ERRCNT ;LOAD ERROR COUNT
1909 010360 012700 000002          MOV     #2,R0    ;SAVE CORE THIS WAY!
1910 010364 013737 001404 001406  MOV     DMCSR,DMCSRH
1911 010372 005237 001406          INC     DMCSRH
1912 010376 013737 001406 001410  MOV     DMCSRH,DMCTL
1913 010404 005237 001410          INC     DMCTL
1914 010410 013737 001410 001412  MOV     DMCTL,DMP04
1915 010416 060037 001412          ADD     R0,DMP04
1916 010422 013737 001412 001414  MOV     DMP04,DMP06
1917 010430 060037 001414          ADD     R0,DMP06
1918
1919 010434 013737 001374 001376  MOV     DMRVEC,DMRLVL ;PTY LVL
1920 010442 060037 001376          ADD     R0,DMRLVL
1921 010446 013737 001376 001400  MOV     DMRLVL,DMTVEC ;TX VEC
1922 010454 060037 001400          ADD     R0,DMTVEC
1923 010460 013737 001400 001402  MOV     DMTVEC,DMTLVL ;TX LVL
1924 010466 060037 001402          ADD     R0,DMTLVL
1925
1926 010472 032737 000002 001236  BIT     #SW01,STRTSW ;IS TEST NO. SELECTED
1927 010500 001450                    BEQ                    ;BR IF NO
1928 010502
1929 010502 005737 000042          4$: TST     @#42    ;RUNNING IN AUTO MODE?

```



```

1930 010506 001045          BNE      7$          ;BR IF YES
1931 010510 104402 005672  TYPE      ,MCRLF
1932 010514 104403          INSTR
1933 010516 006131          MTSTN
1934 010520 104405          PARAM
1935 010522 000001          1
1936 010524 001000          1000
1937 010526 001226          TSTNO
1938 010530      000          .BYTE 0
1939 010531      001          .BYTE 1
1940 010532 012700 012526  MOV      #TST1,R0
1941 010536 022710 5$:    CMP      (PC)+,(R0)    ;CMP FIRST WORD TO 12737
1942 010540 012737          MOV      (PC)+,@(PC)+
1943 010542 001020          BNE      6$          ;BR IF NOT SAME
1944 010544 023760 001226 000002  CMP      TSTNO,2(R0)  ;DOES TSTNO MATCH?
1945 010552 001014          BNE      6$          ;BR IF NO
1946 010554 022760 001226 000004  CMP      #TSTNO,4(R0) ;IS LAST WORD OK?
1947 010562 001010          BNE      6$          ;BR IF NO
1948 010564 010037 001214  MOV      R0,RETURN    ;IT IS A LEGAL TEST SO DO IT
1949 010570 104402 005756  TYPE      ,MR
1950 010574 042737 000002 001236  BIC      #SW01,STRTSW
1951 010602 000412          BR       8$
1952 010604 005720          TST      (R0)+        ;POP R0
1953 010606 020027 031762  CMP      R0,#TLAST+10 ;AT END YET?
1954 010612 001351          BNE      5$          ;BR IF NO
1955 010614 104402 005666  TYPE      ,MQM        ;YES ILLEGAL TEST NO.
1956 010620 000730          BR       4$          ;TRY AGAIN
1957
1958 010622 012737 012526 001214 7$:    MOV      #TST1,RETURN ;PREPARE RETURN ADDRESS
1959 010630 013701 001404 8$:    MOV      DMCSR,R1    ;R1 = BASE DMC11 ADDRESS
1960 010634 000177 170354  JMP      @RETURN     ;GO START TESTING.
1961
1962
1963          ;ROUTINE USED TO "AUTO SIZE" THE DMC11
1964          ;CSR AND VECTOR.
1965          ;NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
1966          ;      ADDRESS RANGE (160000:164000)
1967          ;      AND THE VECTOR MAY BE ANY WHERE IN THE
1968          ;      FLOATING VECTOR RANGE (300:770)
1969          ;
1970          ;
1971          AUTO.SIZE:
1972 010640 000005          RESET
1973 010642 012702 001500  CSRMAP: MOV      #DM.MAP,R2    ;INSURE A BUS INIT.
1974 010646 005022          CLR      (R2)+        ;LOAD MAP POINTER.
1975 010650 022702 001700  1$:    CMP      #DM.END,R2    ;ZERO ENTIRE MAP
1976 010654 001374          BNE      1$          ;ALL DONE?
1977 010656 005037 001310  CLR      DMNUM        ;BR IF NO
1978 010662 012702 001500  MOV      #DM.MAP,R2    ;SET OCTAL NUMBER OF DMC11'S TO 0
1979 010666 005037 001306  CLR      DMACTV       ;R2 POINTS TO DMC MAP
1980 010672 032737 000001 001236  BIT      #SW00,STRTSW ;CLEAR ACTIVE
1981 010700 001002          BNE      ,+6         ;QUESTIONS?
1982 010702 000137 011460  JMP      7$          ;BR IF YES
1983 010706 012737 000001 001256  MOV      #1,TEMP5     ;IF NO SKIP QUESTIONS
1984 010714 104403          INSTR
1985 010716 006451          NUM

```



```

1986 010720 104405          PARAM
1987 010722 000001          1
1988 010724 000020          16.
1989 010726 001252          TEMP3
1990 010730 000          .BYTE 0
1991 010731 001          .BYTE 1
1992 010732 013737 001252 001310 12$: MOV TEMP3,DMNUM ;DMNUM = HOW MANY
1993 010740 104402 005672 TYPE ,MCRLF
1994 010744 104410          CONVRT ;TYPE WHICH DMC IS BEING DONE
1995 010746 012210          WHICH ;TEMP5 IS WHICH DMC
1996 010750 005237 001256 INC TEMP5
1997 010754 104403          INSTR
1998 010756 006511          CSR
1999 010760 104405          PARAM
2000 010762 160000          160000
2001 010764 164000          164000
2002 010766 001254          TEMP4
2003 010770 000          .BYTE 0
2004 010771 001          .BYTE 1
2005 010772 013722 001254 MOV TEMP4,(R2)+ ;STORE CSR IN MAP
2006 010776 104403          INSTR
2007 011000 006527          VEC
2008 011002 104405          PARAM
2009 011004 000000          0
2010 011006 000776          776
2011 011010 001254          TEMP4
2012 011012 000          .BYTE 0
2013 011013 001          .BYTE 1
2014 011014 013712 001254 MOV TEMP4,(R2) ;STORE VECTOR IN MAP
2015 011020 104402          TYPE 10$:
2016 011022 006550          PRIO ;ASK WHAT BR LEVEL
2017 011024 004737 012474 JSR PC,INTTY ;GET RESPONSE
2018 011030 022703 000024 CMP #24,R3 ;
2019 011034 101014          BHI 50$ ;BR IF LESS THAN 4
2020 011036 022703 000027 CMP #27,R3 ;
2021 011042 103411          BLO 50$ ;BR IF GREATER THAN 7
2022 011044 012704 000011 MOV #11,R4 ;R4 = NUMBER OF SHIFTS
2023 011050 006303          ASL R3 ;SHIFT R3 LEFT
2024 011052 005304          DEC R4 ;DEC SHIFT COUNT
2025 011054 001375          BNE .-4 ;BR IF NOT DONE
2026 011056 042703 170777 BIC #170777,R3 ;BIC UNWANTED BITS
2027 011062 050312          BIS R3,(R2) ;PUT BR LEVEL IN STATUS MAP
2028 011064 000403          BR 8$ ;CONTINUE
2029 011066 104402          TYPE 50$:
2030 011070 005666          MQM ;RESPONSE IS OUT OF LIMITS
2031 011072 000752          BR 10$ ;TRY AGAIN
2032 011074 104402          TYPE 8$:
2033 011076 006607          CRAM ;DOES DMC HAVE CRAM?
2034 011100 004737 012474 JSR PC,INTTY ;GET REPLY
2035 011104 022703 000131 CMP #131,R3
2036 011110 001427          BEQ 9$ ;YES
2037 011112 022703 000116 CMP #116,R3 ;NO
2038 011116 001403          BEQ 40$ ;NOT A Y OR N
2039 011120 104402          TYPE
2040 011122 005666          MQM ;TYPE "?"
2041 011124 000763          BR 8$ ;ASK AGAIN

```



```

2042 011126 104402          40$:  TYPE
2043 011130 007341          SPEED
2044 011132 004737 012474  JSR    PC,INTTY      ;DMC11-AR OR DMC11-AL?
2045 011136 022703 000122  CMP    #122,R3      ;GET RESPONSE
2046 011142 001414          BEQ    16$          ;IS IT R
2047 011144 022703 000114  CMP    #114,R3     ;BR IF REMOTE
2048 011150 001403          BEQ    41$          ;IS IT L
2049 011152 104402          TYPE
2050 011154 005666          MQM
2051 011156 000763          BR     40$          ;TRY AGAIN
2052 011160 052762 000002 000004 41$:  BIS    #BIT1,4(R2)  ;SET BIT1 IN STAT3
2053 011166 000402          BR     16$          ;CONTINUE
2054 011170 052712 100000  9$:   BIS    #BIT15,(R2) ;SET BIT 15 IF CRAM
2055 011174          16$:
2056
2057 011174 104402          TYPE
2058 011176 006705          MODU
2059 011200 004737 012474  JSR    PC,INTTY     ;ASK WHICH LINE UNIT
2060 011204 022703 000021  CMP    #21,R3      ;GET REPLY
2061 011210 001417          BEQ    30$          ;'1'
2062 011212 022703 000022  CMP    #22,R3      ;'2'
2063 011216 001412          BEQ    31$          ;'N'
2064 011220 022703 000116  CMP    #116,R3     ;'N'
2065 011224 001403          BEQ    32$
2066 011226 104402          TYPE
2067 011230 005666          MQM
2068 011232 000760          BR     16$          ;IF NOT A 1,2 OR N TYPE '?'
2069 011234 052722 010000  32$:  BIS    #BIT12,(R2)+ ;TRY AGIAN
2070 011240 022222          CMP    (R2)+,(R2)+ ;SET BIT 12 IN STAT2 IF NO LU
2071 011242 000477          BR     33$          ;POP OVER STAT2 AND STAT3
2072 011244 052712 020000  31$:  BIS    #BIT13,(R2)  ;SET BIT 13 IN STAT2 IF M8202
2073 011250 104402          30$:  TYPE
2074 011252 007115          CONN
2075 011254 004737 012474  JSR    PC,INTTY     ;ASK IF LOOP-BACK IS ON
2076 011260 022703 000131  CMP    #131,R3     ;GET REPLY
2077 011264 001406          BEQ    17$          ;Y
2078 011266 022703 000116  CMP    #116,R3     ;N
2079 011272 001436          BEQ    18$
2080 011274 104402          TYPE
2081 011276 005666          MQM
2082 011300 000763          BR     30$          ;IF NOT Y OR N TYPE '?'
2083 011302 052722 040000  17$:  BIS    #BIT14,(R2)+ ;TRY AGAIN
2084 011306 032762 020000 177776 BIT    #BIT13,-2(R2) ;TURNAROUND IS CONNECTED
2085 011314 001027          BNE    19$          ; M8202?
2086 011316          440$:
2087
2088 011316 104402          TYPE
2089 011320 007455          MV35
2090 011322 004737 012474  JSR    PC,INTTY     ; ASK QUESTION
2091 011326 122703 000104  CMPB  #'D,R3      ; ABOUT MODEM TTYPE
2092 011332 001004          BNE    442$        ; GET ANSWER.
2093
2094 011334 042762 000004 000002 BIC    #BIT2,2(R2)  ; IS IT DMC11-DA?
2095 011342 000411          BR     441$        ; NO.
2096
2097 011344 122703 000106  442$: CMPB  #'F,R3      ; YES INDICATE IT IN STAT3
                ; IS IT A DMC11-FA (V.35)?

```



```

2098 011350 001403          BEQ      443$      ; YES-TAKE CARE OF IT.
2099
2100 011352 104402          TYPE
2101 011354 005666          MQM           ; NO ASK OPERATER WHATS GOING ON.
2102 011356 000757          BR      440$      ; REASK QUESTION
2103
2104 011360 052762 000004 000002 443$:  BIS      #BIT2,2(R2) ; YES V.35, RECORD IN STAT3
2105
2106 011366                441$:          ; END DECISION POINT.
2107 011366 000402          BR      19$
2108 011370 042722 040000 18$:      BIC      #BIT14,(R2)+ ;NO TURNAROUND
2109 011374 19$:
2110 011374 104403          INSTR
2111 011376 007017          LINE
2112 011400 104405          PARAM
2113 011402 000000          0
2114 011404 000377          377
2115 011406 001254          TEMP4
2116 011410                000      .BYTE 0
2117 011411                001      .BYTE 1
2118 011412 113722 001254  MOVB    TEMP4,(R2)+ ;STORE SWITCH PAC IN MAP
2119 011416 104403          INSTR
2120 011420 007055          BM
2121 011422 104405          PARAM
2122 011424 000000          0
2123 011426 000377          377
2124 011430 001254          TEMP4
2125 011432                000      .BYTE 0
2126 011433                001      .BYTE 1
2127 011434 113722 001254  MOVB    TEMP4,(R2)+ ;STORE SWITCH PAC IN MAP
2128 011440 005722          TST    (R2)+      ;POP OVER STAT3
2129 011442 005337 001252 33$:    DEC    TEMP3      ;DEC DMC COUNT
2130 011446 001402          BEQ    34$        ;BR IF DONE
2131 011450 000137 010740  JMP    12$        ;JUMP IF NOT
2132 011454 000137 012110 34$:    JMP    13$        ;CONTINUE
2133 011460 012701 160000 7$:     MOV    #160000,R1 ;SET FOR FIRST ADDRESS TO BE TESTED
2134 011464 012737 012202 000004 MOV    #6$,@#4    ;SET FOR NON-EXISTANT DEVICE TIME OUT
2135 011472 005011          CLR    (R1)      ;CLEAR SEL0
2136 011474 005711          TST   (R1)      ;IF DMC11 DMCSR S/B 0
2137 011476 001172          BNE   3$         ;IF NO DEV ; TRAP TO 4. IF NO BIT 8 THEN NO DMC11
2138 011500 005061 000006  CLR    5(R1)     ;CLEAR SEL6
2139 011504 005761 000006  TST   6(R1)     ;IF DMC11 THEN DMRIC S/B =0!
2140 011510 001165          BNE   3$         ;BR IF NOT DMC11
2141 011512 012711 002000  MOV    #BIT10,(R1) ;SET ROM0
2142 011516 005061 000004  CLR    4(R1)     ;CLEAR SEL4
2143 011522 012761 125252 000006 MOV    #125252,6(R1) ;WRITE THIS TO SEL6
2144 011530 052711 020000  BIS    #BIT13,(R1) ;WRITE IT!
2145 011534 022761 125252 000004 CMP    #125252,4(R1) ;WAS IT WRITTEN?
2146 011542 001004          BNE   21$        ;IF NO IT IS NOT CRAM
2147 011544 052762 100000 000002 BIS    #BIT15,2(R2) ;SET BIT15 IF CRAM
2148 011552 000431          BR    22$
2149 011554 012711 001000 21$:    MOV    #BIT9,(R1) ;SET ROM1
2150 011560 012761 100430 000006 MOV    #100430,6(R1) ;PUT INSTRUCTION IN SEL6
2151 011566 012711 001400  MOV    #BIT9!BIT8,(R1) ;CLOCK INSTRUCTION (MICRO PROC PC TO 0)
2152 011572 012711 002000  MOV    #BIT10,(R1) ;SET ROM0
2153 011576 022761 016472 000006 CMP    #016472,6(R1) ;IS IT LOCAL CROM?

```



```

2154 011604 001411          BEQ      23$          ;BR IF YES
2155 011606 022761 016461 000006    CMP      #016461,6(R1) ;IS IT REMOTE CROM?
2156 011614 001410          BEQ      22$          ;BR IF YES
2157 011616 022761 177777 000006    CMP      #-1,6(R1)    ;NO CROM?
2158 011624 001404          BEQ      22$          ;BR IF YES
2159 011626 000516          BR       3$          ;NOT A DMC
2160 011630 052762 000002 000006    23$:  BIS      #BIT1,6(R2) ;SET BIT 1 IN STAT3
2161          ;AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A DMC11 CSR ADDRESS.
2162 011636 010122          22$:  MOV      R1,(R2)+ ;STORE CSR IN CORE TABLE.
2163 011640 012711 001000          15$:  MOV      #BIT9,(R1) ;CLEAR LINE UNIT LOOP
2164 011644 005061 000004          CLR      4(R1)        ;CLEAR PORT4
2165 011650 012761 122113 000006    MOV      #122113,6(R1) ;LOAD INSTRUCTION (CLR DTR)
2166 011656 052711 000400          BIS      #BIT8,(R1)   ;CLOCK INSTRUCTION
2167 011662 012761 021264 000006    MOV      #021264,6(R1) ;LOAD INSTRUCTION
2168 011670 052711 000400          BIS      #BIT8,(R1)   ;CLOCK INSTRUCTION
2169 011674 122761 000377 000004    CMPB    #377,4(R1)    ;IS IT ALL ONES?
2170 011702 001003          BNE     .+10         ;BR IF NO
2171 011704 052712 010000          BIS      #BIT12,(R2)  ;IF YES, NO LINE UNIT, SET STATUS BIT
2172 011710 000436          BR       20$
2173 011712 032761 000002 000004    BIT      #BIT1,4(R1)  ;IS SWITCH A ONE?
2174 011720 001403          BEQ     .+10         ;BR IF M8201
2175 011722 052712 060000          BIS      #BIT13!BIT14,(R2) ;M8202 ASSUME CONNECTOR
2176 011726 000427          BR       20$        ;CONNECTOR ON)
2177 011730 032761 000010 000004    BIT      #BIT3,4(R1)  ;IS MRDY SET
2178 011736 001023          BNE     20$         ;BR IF M8201 NO CONNECTOR (ON LINE)
2179 011740 012761 000100 000004    MOV      #BIT6,4(R1)  ;LOAD PORT4
2180 011746 012761 122113 000006    MOV      #122113,6(R1) ;LOAD INSTRUCTION
2181 011754 052711 000400          BIS      #BIT8,(R1)   ;CLOCK INSTRUCTION(SET DTR)
2182 011760 012761 021264 000006    MOV      #021264,6(R1) ;LOAD INSTRUCTION
2183 011766 052711 000400          BIS      #BIT8,(R1)   ;CLOCK INSTRUCTION(READ MODEM REG)
2184 011772 032761 000010 000004    BIT      #BIT3,4(R1)  ;IS MRDY SET NOW?
2185 012000 001402          BEQ     20$         ;BR IF NO CONNECTOR
2186 012002 052712 040000          BIS      #BIT14,(R2)  ;SET STATUS BIT FOR CONNECTOR
2187 012006 005722          20$:  TST      (R2)+      ;POP POINTER
2188 012010 012761 021324 000006    MOV      #021324,6(R1) ;PUT INSTRUCTION IN PORT6
2189 012016 012711 001400          MOV      #BIT9!BIT8,(R1) ;PORT4_LU 15
2190 012022 156122 000004          BISB    4(R1),(R2)+  ;STORE DDCMP LINE # IN TABLE
2191 012026 012761 021344 000006    MOV      #021344,6(R1) ;PORT6_INSTRUCTION
2192 012034 012711 001400          MOV      #BIT8!BIT9,(R1) ;CLOCK_INSTR.
2193 012040 156122 000004          BISB    4(R1),(R2)+  ;STORE BM873 ADD IN TABLE
2194 012044 005722          TST      (R2)+      ;POP OVER STAT3
2195 012046 005011          CLR      (R1)        ;CLEAR ROMI
2196 012050 005237 001310          INC     DMNUM        ;UPDATE DEVICE COUNTER
2197 012054 022737 000020 001310    CMP      #20,DMNUM    ;ARE MAX. NO. OF DEV FOUND?
2198 012062 001412          BEQ     13$         ;YES DON'T LOOK FOR ANY MORE.
2199 012064 005011          3$:  CLR      (R1)        ;CLEAR BIT 10
2200 012066 005061 000006          CLR     6(R1)       ;CLEAR SEL 6
2201 012072 062701 000010          14$:  ADD     #10,R1     ;UPDATE CSR POINTER ADDRESS
2202 012076 022701 164000          CMP     #164000,R1
2203 012102 001402          BEQ     13$         ;BR IF DONE
2204 012104 000137 011472          JMP     2$          ;JUMP IF NOT
2205 012110 005037 001306          13$:  CLR     DMACTV      ;WERE ANY DMC11'S FOUND AT ALL?
2206 012114 005737 001310          TST     DMNUM        ;ERROR AUTO SIZER FOUND NO DMC11'S IN THIS SYS.
2207 012120 001423          BEQ     5$
2208 012122 013701 001310          MOV     DMNUM,R1
2209 012126 010137 001314          MOV     R1,SAVNUM    ;SAVE NUMBER OF DEVICES

```



```

2210 012132 000241          4$: CLC
2211 012134 006137 001306  ROL    DMACTV      ;GENERATE ACTIVE REGISTER OF DEVICES.
2212 012140 005237 001306  INC    DMACTV      ;SET THE BIT
2213 012144 005301          DEC    R1
2214 012146 001371          BNE    4$         ;BR IF MORE TO GENERATE
2215 012150 012737 000006 000004  MOV    #6,@#4     ;RESTORE TRAP VECTOR
2216 012156 013737 001306 001312  MOV    DMACTV,SAVACT ;SAVE ACTIVE REGISTER
2217 012164 000137 012216  JMP    VECMAP     ;GO FIND THE VECTOR NOW.
2218 012170 104402 005761  5$:  TYPE    ,MERR2 ;NOTIFY OPR THAT NO DMC11'S FOUND.
2219 012174 005000          CLR    R0         ;MAKE DATA LIGHTS ZERO
2220 012176 000000          HALT
2221 012200 000776          BR     -2         ;STOP THE SHOW
2222 012202 012716 012072  6$:  MOV    #14$,(SP) ;DISABLE CONT. SW.
2223 012206 000002          RTI             ;ENTERED BY NON-EXISTANT TIME-OUT.
2224                                     ;RETURN TO MAINSTREAM
2225 012210 000001          WHICH: 1
2226 012212          002          .BYTE 2,2
2227 012214 001256          TEMPS
2228
2229 012216 032737 000001 001236  VECMAP: BIT    #SW00,STRTSW
2230 012224 001114          BNE    5$
2231 012226 012737 000340 000022  MOV    #340,@#22   ;SET IOT TRAP PRIO TO 7
2232 012234 012737 012410 000020  MOV    #4$,@#20   ;SET IOT TRAP VECTOR
2233 012242 012702 001500          MOV    #DM.MAP,R2 ;SET SOFTWARE POINTER
2234 012246 012700 000300          MOV    #300,R0    ;FLOATING VECTORS START HERE.
2235 012252 012701 000302          MOV    #302,R1    ;PC OF IOT INSTR.
2236 012256 010120  1$:  MOV    R1,(R0)+   ;START FILLING VECTOR AREA
2237 012260 012721 000004          MOV    #4,(R1)+  ;WITH .+2; IOT
2238 012264 022021          CMP    (R0)+,(R1)+ ;ADD 2 TO R0 +R1
2239 012266 020127 001000          CMP    R1,#1000
2240 012272 101771          BLOS  1$         ;BR IF MORE TO FILL
2241 012274 013737 001306 001246  MOV    DMACTV,TEMP1 ;STORE TEMPORALLY
2242 012302 006037 001246  2$:  ROR    TEMP1     ;BRING OUT A BIT
2243 012306 103063          BCC    5$         ;BR IF ALL DONE
2244 012310 012704 000012          MOV    #12,R4    ;R4 IS INDEX REGISTER
2245 012314 016437 012460 177776  MOV    BRLVL(R4),PS ;SET PS TO 7
2246 012322 011201          MOV    (R2),R1
2247 012324 012761 000200 000004  MOV    #200,4(R1)
2248 012332 012711 001000          MOV    #BIT9,(R1) ;SET ROMI
2249 012336 012761 121111 000006  MOV    #121111,6(R1) ;PUT INSTRUCTION IN PORT6
2250 012344 012711 001400          MOV    #BIT9!BIT8,(R1) ;FORCE AN INTERRUPT
2251 012350 105200  7$:  INCB  R0         ;STALL
2252 012352 001376          BNE    -2         ;FOR TIME TO INTERUPT
2253 012354 162704 000002          SUB    #2,R4     ;GET NEXT LOWEST PS LEVEL
2254 012360 001404          BEQ    6$         ;BR IF R4 = 0
2255 012362 016437 012460 177776  MOV    BRLVL(R4),PS ;MOVE NEXT LOWER LEVEL IN PS
2256 012370 000767          BR     7$         ;BR TO DELAY
2257 012372 052762 005300 000002  6$:  BIS    #5300,2(R2) ;NO INTERUPT ASSUME 300 AT LEVEL 5 AND FIX DMC11 LATER
2258 012400 005011  3$:  CLR    (R1)      ;CLEAR ROMI
2259 012402 062702 000010          ADD    #10,R2    ;POP SOFTWARE POINTER
2260 012406 000735          BR     2$         ;KEEP GOING
2261 012410 051662 000002  4$:  BIS    (SP),2(R2) ;GET VECTOR ADDRESS
2262 012414 042762 000007 000002  BIC    #7,2(R2)   ;CLEAR JUNK
2263 012422 016405 012462          MOV    BRLVL+2(R4),R5 ;GET BR LEVEL OF DMC11
2264 012426 006305          ASL    R5        ;SHIFT LEVEL 4 PLACES
2265 012430 006305          ASL    R5        ;TO THE LEFT FOR THE

```



```

2266 012432 006305      ASL      R5      ;STATUS TABLE
2267 012434 006305      ASL      R5
2268 012436 042705 170777 BIC      #170777,R5 ;CLEAR UNWANTED BITS
2269 012442 050562 000002 BIS      R5,2(R2) ;PUT BR LEVEL IN STATUS TABLE
2270 012446 022626      (MP      (SP)+,(SP)+ ;POP IOT JUNK OFF STACK
2271 012450 012716 012400 MOV      #3$, (SP) ;SET FOR RETURN
2272 012454 000002      RTI
2273 012456 000207      5$: RTS      PC      ;ALL DONE WITH "AUTO SIZING"
2274
2275 012460 000000      BPLVL: 0      ;LEVEL 0
2276 012462 000000      0      ;LEVEL 0
2277 012464 000200      200     ;LEVEL 4
2278 012466 000240      240     ;LEVEL 5
2279 012470 000300      300     ;LEVEL 6
2280 012472 000340      340     ;LEVEL 7
2281
2282
2283 012474 105777 166504 INTTY: TSTB @TKCSR ;WAIT FOR DONE
2284 012500 100375      BPL      -4
2285 012502 017703 166500 MOV      @TKDBR,R3 ;PUT CHAR IN R3
2286 012506 105777 166476 TSTB    @TPCSR ;WAIT UNTIL PRINTER IS READY
2287 012512 100375      BPL      -4
2288 012514 010377 166472 MOV      R3,@TPDBR ;ECHO CHAR
2289 012520 042703 000240 BIC      #BIT7!BIT5,R3 ;MASK OFF LOWER CASE
2290 012524 000207      RTS      PC      ;RETURN
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303 012526 012737 000001 001226 TST1: MOV      #1,TSTNO
2304 012534 012737 012602 001216 MOV      #TST2,NEXT
2305
2306 012542 005077 166636 CLR      @DMCSR ;R1 CONTAINS BASE DMC11 ADDRESS
2307 012546 012702 000011 MOV      #11,R2 ;CLEAR SELO
2308 012552 104414 ROMCLK ;SAVE R2 FOR TYPEOUT
2309 012554 021224 021004!<20*11> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2310 012556 016104 000004 MOV      4(R1),R4 ;PORT4 LINE UNIT REG 11
2311 012562 042704 000054 BIC      #54,R4 ;PUT "FOUND" IN R4
2312 012566 012705 000020 MOV      #20,R5 ;CLEAR UNKNOWN BITS
2313 012572 120504 CMPB    R5,R4 ;PUT "EXPECTED" IN R5
2314 012574 001401 BEQ     1$ ;IS OUT READY SET?
2315 012576 104002 HLT     2 ;BR IF YES
2316 012600 104400      1$: SCOPE ;ERROR IN LU 11
2317 ;SCOPE THIS TEST
2318
2319
2320
2321

```

```

:***** 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
:-----
MOV      #1,TSTNO
MOV      #TST2,NEXT
CLR      @DMCSR ;R1 CONTAINS BASE DMC11 ADDRESS
MOV      #11,R2 ;CLEAR SELO
ROMCLK ;SAVE R2 FOR TYPEOUT
021004!<20*11> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
MOV      4(R1),R4 ;PORT4 LINE UNIT REG 11
BIC      #54,R4 ;PUT "FOUND" IN R4
MOV      #20,R5 ;CLEAR UNKNOWN BITS
CMPB    R5,R4 ;PUT "EXPECTED" IN R5
BEQ     1$ ;IS OUT READY SET?
HLT     2 ;BR IF YES
1$: SCOPE ;ERROR IN LU 11
;SCOPE THIS TEST

```

```

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

```



```

2322                                     ;*BITS ARE IN THE CORRECT STATE
2323                                     ;:*****
2324
2325                                     ; TEST 2
2326                                     ;-----
2327 012602 012737 000002 001226 TST2: MOV #2,TSTNO
2328 012610 012737 012650 001216 MOV #TST3,NEXT
2329                                     ;R1 CONTAINS BASE DMC11 ADDRESS
2330 012616 012702 000012 MOV #12,R2 ;SAVE R2 FOR TYPEOUT
2331 012622 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2332 012624 021244 021004!<20*12> ;PORT4 LINE UNIT REG 12
2333 012626 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
2334 012632 042704 000017 BIC #17,R4 ;CLEAR UNKNOWN BITS
2335 012636 005005 CLR R5 ;PUT 'EXPECTED' IN R5
2336 012640 120504 CMPB R5,R4 ;ARE ALL BITS CLEARED?
2337 012642 001401 BEQ 1$ ;BR IF YES
2338 012644 104002 HLT 2 ;ERROR IN LU 12
2339 012646 104400 1$: SCOPE ;SCOPE THIS TEST
2340
2341
2342                                     ;***** TEST 3 *****
2343                                     ;*MODEM CONTROL REGISTER READ/ONLY TEST
2344                                     ;*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
2345                                     ;*BITS ARE IN THE CORRECT STATE
2346                                     ;:*****
2347
2348                                     ; TEST 3
2349                                     ;-----
2350 012650 012737 000003 001226 TST3: MOV #3,TSTNO
2351 012656 012737 012722 001216 MOV #TST4,NEXT
2352                                     ;R1 CONTAINS BASE DMC11 ADDRESS
2353 012664 104412 MSTCLR ;MASTER CLEAR DMC11
2354 012666 012702 000013 MOV #13,R2 ;SAVE R2 FOR TYPEOUT
2355 012672 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2356 012674 021264 021004!<20*13> ;PORT4 LINE UNIT REG 13
2357 012676 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
2358 012702 042704 000213 BIC #213,R4 ;CLEAR UNKNOWN BITS
2359 012706 012705 000100 MOV #100,R5 ;PUT 'EXPECTED' IN R5
2360 012712 120504 CMPB R5,R4 ;ARE RING, DTR, AND MODEM READY SET?
2361 012714 001401 BEQ 1$ ;BR IF YES
2362 012716 104002 HLT 2 ;ERROR IN LU 13
2363 012720 104400 1$: SCOPE ;SCOPE THIS TEST
2364
2365
2366                                     ;***** TEST 4 *****
2367                                     ;*MAINTENANCE REGISTER READ/ONLY TEST
2368                                     ;*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
2369                                     ;*BITS ARE IN THE CORRECT STATE
2370                                     ;:*****
2371
2372                                     ; TEST 4
2373                                     ;-----
2374 012722 012737 000004 001226 TST4: MOV #4,TSTNO
2375 012730 012737 013024 001216 MOV #TST5,NEXT
2376                                     ;R1 CONTAINS BASE DMC11 ADDRESS
2377 012736 104412 MSTCLR ;MASTER CLEAR DMC11

```



```

2378 012740 012702 000017      MOV      #17,R2      ;SAVE R2 FOR TYPEOUT
2379 012744 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2380 012746 021364      021004!<20*17>    ;PORT4 LINE UNIT REG 17
2381 012750 016104 000004      MOV      4(R1),R4    ;PUT 'FOUND' IN R4
2382 012754 042704 000206      BIC      #206,R4     ;CLEAR UNKNOWN BITS
2383 012760 012705 000051      MOV      #51,R5     ;PUT 'EXPECTED' IN R5
2384 012764 032737 020000 001366      BIT      #BIT13,STAT1 ;IS LU AN M8202 OR M8201?
2385 012772 001004      BNE      2$         ;BR IF M8202
2386 012774 032737 040000 001366      BIT      #BIT14,STAT1 ;CONNECTOR???
2387 013002 001004      BNE      3$         ;BR IF M8201 WITH CONNECTOR
2388 013004 042704 000040      2$:      BIC      #40,R4     ;MASK OFF SI BIT IF M8202 OR M8201, NO CONNECTOR
2389 013010 042705 000040      BIC      #BIT5,R5    ;SI BIT IS UNKNOWN
2390 013014      3$:
2391 013014 120504      CMPB     R5,R4      ;ARE SI AND ICIR SET?
2392 013016 001401      BEQ      1$         ;BR IF YES
2393 013020 104002      HLT      2          ;ERROR IN LU 17
2394 013022 104400      1$:      SCOPE     ;SCOPE THIS TEST
2395
2396
2397
2398      ;***** TEST 5 *****
2399      ;*LINE UNIT REGISTER WRITE/READ TEST
2400      ;*SET BIT5 IN LU REGISTER 12, VERIFY IT IS SET
2401      ;*CLEAR BIT5 IN LU REGISTER 12, VERIFY IT IS CLEAR
2402      ;*****
2403
2404      ; TEST 5
2405      ;-----
2405 013024 012737 000005 001226 TST5:  MOV      #5,TSTNO
2406 013032 012737 013164 001216      MOV      #TST6,NEXT
2407 013040 012737 013054 001220      MOV      #1$,LOCK
2408
2409 013046 104412      MSTCLR
2410 013050 012702 000012      MOV      #12,R2     ;R1 CONTAINS BASE DMC11 ADDRESS
2411 013054 012761 000040 000004 1$:  MOV      #40,4(R1)  ;MASTER CLEAR DMC11
2412 013062 104414      ROMCLK      ;SAVE REGISTER ADDRESS FOR TYPEOUT
2413 013064 122112      122112      ;LOAD PORT4
2414 013066 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2415 013070 021245      021245      ;SET BIT5 IN LU-12
2416 013072 012705 000040      MOV      #40,R5     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2417 013076 116104 000005      MOVB     5(R1),R4    ;READ LU-12
2418 013102 042704 000337      BIC      #337,R4     ;PUT 'EXPECTED' IN R5
2419 013106 120504      CMPB     R5,R4      ;PUT 'FOUND' IN R4
2420 013110 001401      BEQ      2$         ;CLEAR UNWANTED BITS
2421 013112 104003      HLT      3          ;IS BIT5 SET?
2422 013114 104401      2$:      SCOPE1      ;BR IF YES
2423 013116 012737 013124 001220      MOV      #3$,LOCK  ;ERROR, BIT 5 IS NOT SET
2424 013124 005061 000004      3$:      CLR      4(R1)     ;SCOPE SUBTEST (SW09=1)
2425 013130 104414      ROMCLK      ;NEW SCOPE1
2426 013132 122112      122112      ;LOAD PORT4
2427 013134 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2428 013136 021245      021245      ;CLEAR BIT 5 IN LU-12
2429 013140 005005      CLR      R5         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2430 013142 116104 000005      MOVB     5(R1),R4    ;READ LU-12
2431 013146 042704 000337      BIC      #337,R4     ;PUT 'EXPECTED' IN R5
2432 013152 120504      CMPB     R5,R4      ;PUT 'FOUND' IN R4
2433 013154 001401      BEQ      4$         ;CLEAR UNWANTED BITS
                                ;IS BIT5 CLEAR?
                                ;BR IF YES

```



2434 013156 104003  
2435 013160 104401  
2436 013162 104400

4\$: HLT 3 ;ERROR, BIT5 IS NOT CLEAR  
SCOPE1 ;SCOPE SUBTEST (SW09=1)  
SCOPE ;SCOPE THIS TEST

2437  
2438  
2439  
2440  
2441  
2442  
2443  
2444  
2445

\*\*\*\*\* TEST 6 \*\*\*\*\*  
\*LINE UNIT REGISTER WRITE/READ TEST  
\*SET BIT1 IN LU REGISTER 17, VERIFY IT IS SET  
\*CLEAR BIT1 IN LU REGISTER 17, VERIFY IT IS CLEAR  
\*\*\*\*\*

: TEST 6

2446  
2447 013164 012737 000006 001226 TST6:  
2448 013172 012737 013324 001216  
2449 013200 012737 013214 001220

MOV #6,TSTNO  
MOV #TST7,NEXT  
MOV #1\$,LOCK

;R1 CONTAINS BASE DMC11 ADDRESS

2450  
2451 013206 104412  
2452 013210 012702 000017  
2453 013214 012761 000001 000004 1\$:  
2454 013222 104414  
2455 013224 122117  
2456 013226 104414  
2457 013230 021365  
2458 013232 012705 000001  
2459 013236 116104 000005  
2460 013242 042704 000376  
2461 013246 120504  
2462 013250 001401  
2463 013252 104003

MSTCLR ;MASTER CLEAR DMC11  
MOV #17,R2 ;SAVE REGISTER ADDRESS FOR TYPEOUT  
MOV #1,4(R1) ;LOAD PORT4  
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304  
122117 ;SET BIT1 IN LU-17  
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304  
021365 ;READ LU-17  
MOV #1,R5 ;PUT "EXPECTED" IN R5  
MOVB 5(R1),R4 ;PUT "FOUND" IN R4  
BIC #376,R4 ;CLEAR UNWANTED BITS  
CMPB R5,R4 ;IS BIT1 SET?  
BEQ 2\$ ;BR IF YES  
HLT 3 ;ERROR, BIT 1 IS NOT SET

2464 013254 104401 2\$:  
2465 013256 012737 013264 001220  
2466 013264 005061 000004 3\$:

SCOPE1 ;SCOPE SUBTEST (SW09=1)  
MOV #3\$,LOCK ;NEW SCOPE  
CLR 4(R1) ;LOAD PORT4

2467 013270 104414  
2468 013272 122117  
2469 013274 104414  
2470 013276 021365  
2471 013300 005005  
2472 013302 116104 000005  
2473 013306 042704 000376  
2474 013312 120504  
2475 013314 001401  
2476 013316 104003  
2477 013320 104401 4\$:  
2478 013322 104400

ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304  
122117 ;CLEAR BIT 1 IN LU-17  
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304  
021365 ;READ LU-17  
CLR R5 ;PUT "EXPECTED" IN R5  
MOVB 5(R1),R4 ;PUT "FOUND" IN R4  
BIC #376,R4 ;CLEAR UNWANTED BITS  
CMPB R5,R4 ;IS BIT1 CLEAR?  
BEQ 4\$ ;BR IF YES  
HLT 3 ;ERROR, BIT1 IS NOT CLEAR  
SCOPE1 ;SCOPE SUBTEST (SW09=1)  
SCOPE ;SCOPE THIS TEST

2479  
2480  
2481  
2482  
2483  
2484  
2485

\*\*\*\*\* TEST 7 \*\*\*\*\*  
\*LINE UNIT REGISTER WRITE/READ TEST  
\*FLOAT A 1 THROUGH LINE UNIT REGISTER 13  
\*FLOAT A 0 THROUGH LINE UNIT REGISTER 13  
\*\*\*\*\*

: TEST 7

2486  
2487  
2488  
2489 013324 012737 000007 001226 TST7:

MOV #7,TSTNO



```

2490 013332 012737 013534 001216      MOV      #TST10,NEXT
2491 013340 012737 013360 001220      MOV      #64$,LOCK
2492                                     ;R1 CONTAINS BASE DMC11 ADDRESS
2493 013346 104412                       MSTCLR   ;MASTER CLEAR DMC11
2494 013350 012702 000013               MOV      #13,R2
2495 013354 012700 000001               MOV      #1,R0
2496 013360                                     ;START WITH BIT 0
2497 013360 010061 000004               MOV      R0,4(R1)
2498 013364 042761 000257 000004       BIC      #257,4(R1)
2499 013372 104414                       ;PUT PATTERN INTO PORT4
2500 013374 122113                       ;CLEAR UNWANTED BITS
2501 013376 104414                       ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2502 013400 021265                       ;MOV DATA TO IBUS REGISTER 13
2503 013402 010005                       ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2504 013404 042705 000257               ROMCLK  21005!<13*20>
2505 013410 116104 000005               ;READ FROM IBUS REGISTER 13
2506 013414 042704 000257               MOV      R0,R5
2507 013420 120504                       ;PUT EXPECTED IN R5
2508 013422 001401                       BIC      #257,R5
2509 013424 104003                       ;CLEAR UNWANTED BITS
2510 013426 104401                       ;PUT "FOUND" INTO R4
2511 013430 000241                       BIC      #257,R4
2512 013432 106100                       ;CLEAR UNWANTED BITS
2513 013434 001351                       ;DATA CORRECT?
2514 013436 012737 013452 001220       CMPB    R5,R4
2515 013444 012700 000001               BEQ     65$
2516 013450 005100                       ;BR IF YES
2517 013452                                     ;ERROR
2518 013452 010061 000004               HLT     3
2519 013456 042761 000257 000004       SCOPE1  ;SW09=1?
2520 013464 104414                       CLC     ;CLEAR CARRY
2521 013466 122113                       ROLB    R0
2522 013470 104414                       ;SHIFT BIT IN R0
2523 013472 021265                       BNE     64$
2524 013474 010005                       ;IF R0=0 THEN DONE
2525 013476 042705 000257               MOV     #67$,LOCK
2526 013502 116104 000005               ;NEW SCOPE1
2527 013506 042704 000257               MOV     #1,R0
2528 013512 120504                       ;START WITH BIT 0
2529 013514 001401                       ;CHANGE TO FLOATING ZERO
2530 013516 104003                       ;PUT PATTERN INTO PORT4
2531 013520 104401                       ;CLEAR UNWANTED BITS
2532 013522 005100                       ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2533 013524 000241                       ;MOV DATA TO IBUS REGISTER 13
2534 013526 106100                       ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2535 013530 001347                       ;READ FROM IBUS REGISTER 13
2536 013532 104400                       ;PUT EXPECTED IN R5
2537                                     ;CLEAR UNWANTED BITS
2538                                     ;PUT "FOUND" INTO R4
2539                                     ;CLEAR UNWANTED BITS
2540                                     ;DATA CORRECT?
2541                                     ;BR IF YES
2542                                     ;ERROR
2543                                     ;SW09=1?
2544                                     ;CHANGE TO FLOATING 1
2545                                     ;CLEAR CARRY
2546                                     ;SHIFT BIT IN R0
2547                                     ;IF R0=0 THEN DONE
2548                                     ;SCOPE THIS TEST
2549                                     ;***** TEST 10 *****
2550                                     ;*LINE UNIT REGISTER WRITE/READ TEST
2551                                     ;*FLOAT A 1 THROUGH LINE UNIT REGISTER 14
2552                                     ;*FLOAT A 0 THROUGH LINE UNIT REGISTER 14
2553                                     ;*****
2554                                     ; TEST 10

```



```

2546
2547 013534 012737 000010 001226 TST10:  MOV    #10,TSTNO
2548 013542 012737 013710 001216      MOV    #TST11,NEXT
2549 013550 012737 013570 001220      MOV    #64$,LOCK
2550
2551 013556 104412      MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
2552 013560 012702 000014      MOV    #14,R2 ;MASTER CLEAR DMC11
2553 013564 012700 000001      MOV    #1,R0  ;SAVE REGISTER ADDRESS FOR TYPEOUT
2554 013570      64$:      ;START WITH BIT 0
2555 013570 010061 000004      MOV    R0,4(R1) ;PUT PATTERN INTO PORT4
2556 013574 104414      ROMCLK     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2557 013576 122114      122100!14 ;MOV DATA TO IBUS REGISTER 14
2558 013600 104414      ROMCLK     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2559 013602 021305      21005!<14*20> ;READ FROM IBUS REGISTER 14
2560 013604 010005      MOV    R0,R5  ;PUT EXPECTED IN R5
2561 013606 116104 000005      MOV    5(R1),R4 ;PUT "FOUND" INTO R4
2562 013612 120504      CMPB    R5,R4 ;DATA CORRECT?
2563 013614 001401      BEQ    65$   ;BR IF YES
2564 013616 104003      HLT    3     ;ERROR
2565 013620 104401      65$:      SCOP1     ;SW09=1?
2566 013622 000241      CLC      ;CLEAR CARRY
2567 013624 106100      ROLB    R0   ;SHIFT BIT IN R0
2568 013626 001360      BNE    64$   ;IF R0=0 THEN DONE
2569 013630 012737 013644 001220      MOV    #67$,LOCK ;NEW SCOP1
2570 013636 012700 000001      MOV    #1,R0  ;START WITH BIT 0
2571 013642 005100      69$:      COM     R0   ;CHANGE TO FLOATING ZERO
2572 013644      67$:
2573 013644 010061 000004      MOV    R0,4(R1) ;PUT PATTERN INTO PORT4
2574 013650 104414      ROMCLK     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2575 013652 122114      122100!14 ;MOV DATA TO IBUS REGISTER 14
2576 013654 104414      ROMCLK     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2577 013656 021305      21005!<14*20> ;READ FROM IBUS REGISTER 14
2578 013660 010005      MOV    R0,R5  ;PUT EXPECTED IN R5
2579 013662 116104 000005      MOV    5(R1),R4 ;PUT "FOUND" INTO R4
2580 013666 120504      CMPB    R5,R4 ;DATA CORRECT?
2581 013670 001401      BEQ    68$   ;BR IF YES
2582 013672 104003      HLT    3     ;ERROR
2583 013674 104401      68$:      SCOP1     ;SW09=1?
2584 013676 005100      COM     R0   ;CHANGE TO FLOATING 1
2585 013700 000241      CLC      ;CLEAR CARRY
2586 013702 106100      ROLB    R0   ;SHIFT BIT IN R0
2587 013704 001356      BNE    69$   ;IF R0=0 THEN DONE
2588 013706 104400      SCOPE     ;SCOPE THIS TEST
2589
2590
2591
2592      ;***** TEST 11 *****
2593      ;*SWITCH PAC TEST
2594      ;*THIS TEST READS SWITCH PAC#1
2595      ;*THIS SWITCH PAC CONTAINS THE DDCMP LINE #
2596      ;*****
2597
2598      ; TEST 11
2599 013710 012737 000011 001226 TST11:  MOV    #11,TSTNO
2600 013716 012737 013752 001216      MOV    #TST12,NEXT
2601

```

;R1 CONTAINS BASE DMC11 ADDRESS



```

2602 013724 104412
2603 013726 104414
2604 013730 021324
2605 013732 016104 000004
2606 013736 113705 001370
2607 013742 120504
2608 013744 001401
2609 013746 104031
2610 013750 104400
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621 013752 012737 000012 001226 TST12:
2622 013760 012737 014014 001216
2623
2624 013766 104412
2625 013770 104414
2626 013772 021344
2627 013774 016104 000004
2628 014000 113705 001371
2629 014004 120504
2630 014006 001401
2631 014010 104031
2632 014012 104400
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643 014014 012737 000013 001226 TST13:
2644 014022 012737 014114 001216
2645
2646 014030 104412
2647 014032 005037 001416
2648 014036
2649 014036 104414
2650 014040 021364
2651 014042 032761 000002 000004
2652 014050 001004
2653 014052 005237 001416
2654 014056 001367
2655 014060 104004
2656 014062 005037 001416
2657 014066

```

```

MSTCLR ;MASTER CLEAR DMC11
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021324 ;PORT4 LU15
MOV 4(R1),R4 ;PUT 'FOUND' IN R4
MOVB STAT2,R5 ;PUT 'EXPECTED' IN R5
CMPB R5,R4 ;SW OK?
BEQ 1$ ;BR IF YES
HLT 31 ;ERROR, SWITCH PAC READ ERROR
1$: SCOPE ;SCOPE THIS TEST

```

```

;***** TEST 12 *****
; *SWITCH PAC TEST
; *THIS TEST READS SWITCH PAC#2
; *THIS SWITCH PAC CONTAINS THE BM873 BOOT ADD
;*****

```

; TEST 12

```

-----
MOV #12,TSTNO
MOV #TST13,NEXT
MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
ROMCLK ;MASTER CLEAR DMC11
021344 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
MOV 4(R1),R4 ;PORT4 LU16
MOVB STAT2+1,R5 ;PUT 'FOUND' IN R4
CMPB R5,R4 ;PUT 'EXPECTED' IN R5
BEQ 1$ ;SW OK?
HLT 31 ;BR IF YES
1$: SCOPE ;ERROR, SWITCH PAC READ ERROR
;SCOPE THIS TEST

```

```

;***** TEST 13 *****
; *LINE UNIT CLOCK TEST
; *THIS TEST VERIFYS THAT THE LU INTERNAL CLOCK
; *(BIT 1 IN LU-17) IS WORKING
;*****

```

; TEST 13

```

-----
MOV #13,TSTNO
MOV #TST14,NEXT
MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
ROMCLK ;MASTER CLEAR DMC11
1$: CLR TEMP ;PREPARE FOR DELAY
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021364 ;PORT4 LU-17
BIT #2,4(R1) ;IS CLOCK BIT SET?
BNE 2$ ;BR IF YES
INC TEMP ;DELAY
BNE 1$ ;DELAY FINISHED?
HLT 4 ;ERROR BIT IS STUCK CLEAR
2$: CLR TEMP ;PREPARE FOR DELAY
3$:

```



```

2658 014066 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2659 014070 021364 ;PORT4 LU-17
2660 014072 032761 000002 000004 BIT #2,4(R1) ;IS CLOCK BIT CLEAR?
2661 014100 001404 BEQ 4$ ;BR IF YES
2662 014102 005237 001416 INC TEMP ;DELAY
2663 014106 001367 BNE 3$ ;BR IF DELAY NOT DONE
2664 014110 104004 HLT 4 ;ERROR BIT IS STUCK SET
2665 014112 104400 4$: SCOPE

```

```

2666
2667
2668 :***** TEST 14 *****
2669 :*OUT DATA SILO TEST
2670 :*SET SOM AND LOAD OUT DATA SILO
2671 :*VERIFY THAT OCOR SET, INDICATING THAT THE
2672 :*CHARACTER IS AT THE BOTTOM OF THE OUT SILO
2673 :*****

```

```

2674
2675 : TEST 14
2676 :-----
2677 014114 012737 000014 001226 TST14: MOV #14,TSTNO
2678 014122 012737 014230 001216 MOV #TST15,NEXT
2679
2680 014130 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
2681 014132 005061 000004 CLR 4(R1) ;MASTER CLEAR DMC11
2682 014136 104414 ROMCLK ;CLEAR PORT4
2683 014140 122117 122117 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2684 014142 004737 033676 JSR PC,CLRIO ;PUT LINE UNIT IN BITSTUFF MODE
2685 014146 012711 004000 MOV #BIT11,(R1) ;DO THIS AFTER MODE IS SET
2686 014152 012761 000001 000004 MOV #1,4(R1) ;SET LINE UNIT LOOP
2687 014160 104414 ROMCLK ;LOAD PORT4 WITH BIT0
2688 014162 122111 122111 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2689 014164 104414 ROMCLK ;SET SOM
2690 014166 122110 122110 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2691 014170 104416 000002 TIMER, 2 ;LOAD OUT DATA SILO
2692 014174 012702 000017 MOV #17,R2 ;WAIT FOR OCOR
2693 014200 104414 ROMCLK ;SAVE ADDRESS FOR TYPEOUT
2694 014202 021364 021364 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2695 014204 016104 000004 MOV 4(R1),R4 ;PORT4 LU 17
2696 014210 042704 000357 BIC #357,R4 ;PUT 'FOUND' IN R4
2697 014214 012705 000020 MOV #20,R5 ;CLEAR UNWANTED BITS
2698 014220 120504 CMPB R5,R4 ;PUT 'EXPECTED' IN R5
2699 014222 001401 BEQ 1$ ;IS OCOR SET?
2700 014224 104005 HLT 5 ;BR IF YES
2701 014226 1$:
2702 014226 104400 SCOPE ;SCOPE THIS TEST

```

```

2703
2704
2705 :***** TEST 15 *****
2706 :*BITSTUFF TEST OF RTS AND OUT ACTIVE
2707 :*SET SOM AND LOAD OUT DATA SILO
2708 :*SINGLE STEP 2 DATA CLOCKS, VERIFY
2709 :*THAT RTS AND ACTIVE ARE SET
2710 :*****

```

```

2711 : TEST 15
2712 :-----
2713

```



```

2714 014230 012737 000015 001226 TST15: MOV #15,TSTNO
2715 014236 012737 014402 001216 MOV #TST16,NEXT
2716
2717 014244 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
2718 014246 005061 000004 CLR 4(R1) ;MASTER CLEAR DMC11
2719 014252 104414 ROMCLK ;CLEAR PORT4
2720 014254 122117 122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2721 014256 004737 033676 JSR PC,CLRIO ;PUT LINE UNIT IN BITSTUFF MODE
2722 014262 012711 004000 MOV #BIT11,(R1) ;DO THIS AFTER MODE IS SET
2723 014266 012761 000001 000004 MOV #1,4(R1) ;SET LINE UNIT LOOP
2724 014274 104414 ROMCLK ;LOAD PORT4 WITH BIT0
2725 014276 122111 122111 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2726 014300 104414 ROMCLK ;SET SOM
2727 014302 122110 122110 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2728 014304 004737 032346 JSR PC,OCOR ;LOAD OUT DATA SILO
2729 014310 104415 000002 DATACLK, 2 ;WAIT FOR OCOR
2730 014314 012702 000011 MOV #11,R2 ;CLOCK DATA FOUR TIMES
2731 014320 104414 ROMCLK ;SAVE ADDRESS FOR TYPEOUT
2732 014322 021224 021224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2733 014324 016104 000004 MOV 4(R1),R4 ;PORT4 LU 11
2734 014330 042704 000257 BIC #257,R4 ;PUT 'FOUND' IN R4
2735 014334 012705 000120 MOV #120,R5 ;CLEAR UNWANTED BITS
2736 014340 120504 120504 CMPB R5,R4 ;PUT 'EXPECTED' IN R5
2737 014342 001401 001401 BEQ 1$ ;IS ACTIVE SET?
2738 014344 104005 104005 HLT 5 ;BR IF YES
2739 014346 1$:
2740 014346 012702 000013 MOV #13,R2 ;SAVE ADDRESS FOR TYPEOUT
2741 014352 104414 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2742 014354 021264 021264 021264 ;PORT4 LU 13
2743 014356 016104 000004 MOV 4(R1),R4 ;PUT EXPECTED IN R4
2744 014362 042704 000337 BIC #337,R4 ;CLEAR UNWANTED BITS
2745 014366 012705 000040 MOV #BIT5,R5 ;PUT 'EXPECTED' IN R5, RTS SHOULD BE SET
2746 014372 120504 120504 CMPB R5,R4 ;IS RTS OK?
2747 014374 001401 001401 BEQ 2$ ;BR IF YES
2748 014376 104005 104005 HLT 5 ;RTS ERROR
2749 014400 2$:
2750 014400 104400 104400 SCOPE ;SCOPE THIS TEST
2751
2752
2753 ;***** TEST 16 *****
2754 ;*TEST OF OUT CLEAR
2755 ;*SET SOM AND LOAD OUT DATA SILO
2756 ;*SINGLE STEP DATA CLOCK, SET OUT CLEAR
2757 ;*VERIFY THAT OCOR,RTS, AND ACTIVE ARE CLEARED
2758 ;*****
2759
2760 ; TEST 16
2761 ;-----
2762 014402 012737 000016 001226 TST16: MOV #16,TSTNO
2763 014410 012737 014614 001216 MOV #TST17,NEXT
2764
2765 014416 104412 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
2766 014420 005061 000004 CLR 4(R1) ;MASTER CLEAR DMC11
2767 014424 104414 104414 ROMCLK ;CLEAR PORT4
2768 014426 122117 122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2769 014430 004737 033676 JSR PC,CLRIO ;PUT LINE UNIT IN BITSTUFF MODE
;DO THIS AFTER MODE IS SET

```



2770	014434	012711	004000		MOV	#BIT11,(R1)	:SET LINE UNIT LOOP
2771	014440	012761	000001	000004	MOV	#1,4(R1)	:LOAD PORT4 WITH BIT0
2772	014446	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2773	014450	122111			122111		:SET SOM
2774	014452	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2775	014454	122110			122110		:LOAD OUT DATA SILO
2776	014456	004737	032346		JSR	PC,OCOR	:WAIT FOR OCOR
2777	014462	104415	000002		DATACLK,	2	:CLOCK DATA FOUR TIMES
2778	014466	012761	000200	000004	MOV	#BIT7,4(R1)	:SET BIT7 IN PORT4
2779	014474	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2780	014476	122111			122111		:SET OUT CLEAR
2781	014500	104415	000001		DATACLK,	1	:GIVE A TICK TO CLEAR RTS
2782	014504	012702	000017		MOV	#17,R2	:SAVE ADDRESS FOR TYPEOUT
2783	014510	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2784	014512	021364			021364		:PORT4 LU 17
2785	014514	016104	000004		MOV	4(R1),R4	:PUT 'FOUND' IN R4
2786	014520	042704	000357		BIC	#357,R4	:CLEAR UNWANTED BITS
2787	014524	005005			CLR	R5	:PUT 'EXPECTED' IN R5
2788	014526	120504			CMPB	R5,R4	:IS OCOR CLEARED?
2789	014530	001401			BEQ	1\$	:BR IF YES
2790	014532	104005			HLT	5	
2791	014534						
2792	014534	012702	000013	1\$:	MOV	#13,R2	:SAVE ADDRESS FOR TYPEOUT
2793	014540	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2794	014542	021264			021264		:PORT4 LU 13
2795	014544	016104	000004		MOV	4(R1),R4	:PUT EXPECTED IN R4
2796	014550	042704	000337		BIC	#337,R4	:CLEAR UNWANTED BITS
2797	014554	005005			CLR	R5	:PUT 'EXPECTED' IN R5, RTS SHOULD BE CLEARED
2798	014556	120504			CMPB	R5,R4	:IS RTS OK?
2799	014560	001401			BEQ	2\$	:BR IF YES
2800	014562	104005			HLT	5	:RTS ERROR
2801	014564						
2802	014564	012702	000011	2\$:	MOV	#11,R2	:SAVE ADDRESS FOR TYPEOUT
2803	014570	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2804	014572	021224			021224		:PORT4 LU11
2805	014574	016104	000004		MOV	4(R1),R4	:PUT 'FOUND' IN R4
2806	014600	012705	000020		MOV	#BIT4,R5	:ONLY OUT READY SHOULD BE SET
2807	014604	120504			CMPB	R5,R4	:IS ACTIVE CLEAR?
2808	014606	001401			BEQ	3\$	:BR IF YES
2809	014610	104005			HLT	5	:ERROR ACTIVE NOT CLEARED
2810	014612						
2811	014612	104400		3\$:	SCOPE		:SCOPE THIS TEST



2812  
2813  
2814  
2815  
2816  
2817  
2818  
2819  
2820  
2821  
2822  
2823  
2824  
2825  
2826  
2827  
2828  
2829  
2830  
2831  
2832  
2833  
2834  
2835  
2836  
2837  
2838  
2839  
2840  
2841  
2842  
2843  
2844  
2845  
2846  
2847  
2848  
2849  
2850  
2851  
2852  
2853  
2854  
2855  
2856  
2857  
2858  
2859  
2860  
2861  
2862  
2863  
2864  
2865  
2866  
2867

014614 012737 000017 001226 TST17:  
014622 012737 015076 001216  
  
014630 104412  
014632 005061 000004  
014636 104414  
014640 122117  
014642 004737 033676  
014646 005037 034114  
014652 012711 004000  
014656 004737 032500  
014662 012761 000001 000004  
014670 104414  
014672 122111  
014674 104414  
014676 122110  
014700 012705 000000  
014704 004737 032500  
014710 010561 000004  
014714 104414  
014716 122110  
014720 004737 032346  
014724 005003  
014726 010502  
014730 104415 000002  
014734 012737 000176 001252  
014742 104415 000001  
014746 106037 001252  
014752 103405  
014754 004737 032314  
014760 103006  
014762 104026  
014764 000404  
014766 004737 032314  
014772 103401  
014774 104026  
014776 005203  
015000 022703 000010  
015004 001356  
015006 005003  
015010 104415 000001  
015014 106002

```
***** TEST 17 *****
*BITSTUFF TRANSMITTER TEST
*SINGLE CLOCK THE CHARACTER 0
*CHECK FLAG AND DATA IN THE BIT WINDOW
*VERIFY EACH BIT POSITION AS IT
*PASSES THE BIT WINDOW (SI BIT)
*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
*****
```

```
: TEST 17
-----
MOV #17,TSTNO
MOV #TST20,NEXT
MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
CLR 4(R1) ;MASTER CLEAR DMC11
ROMCLK ;CLEAR PORT4
122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
JSR PC,CLRIO ;PUT LINE UNIT IN BITSTUFF MODE
CLR BITCON ;DO THIS AFTER MODE IS SET
MOV #BIT11,(R1) ;CONSECUTIVE 1'S COUNTER INIT TO 0
JSR PC,OUTRDY ;SET LINE UNIT LOOP
MOV #1,4(R1) ;WAIT FOR OUT-READY
ROMCLK ;SET BIT0 IN PORT4
122111 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
ROMCLK ;SET SOM!
122110 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
MOV #0,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
JSR PC,OUTRDY ;WAIT FOR OUT-READY
MOV R5,4(R1) ;LOAD PORT4 WITH CHARACTER
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122110 ;LOAD OUT DATA
JSR PC,OCOR ;WAIT FOR OCOR TO SET
CLR R3 ;CLEAR BIT COUNTER
MOV R5,R2 ;LOAD CHARACTER IN R2
DATACLK, 2 ;2 TICKS TO SET UP TRANSMITTER
MOV #*B<01111110>,TEMP3 ;PUT FLAG CHARACTER IN TEMP3
64$: DATACLK, 1 ;CLOCK FLAG ONCE
RORB TEMP3 ;SHIFT SOFT FLAG
BCS 65$ ;BR IF BIT IS MARK
JSR PC,GETSI ;LOOK AT BIT WINDOW
BCC 66$ ;BR IF OK
HLT 26 ;ERROR IN FLAG CHAR
BR 66$
65$: JSR PC,GETSI ;LOOK AT BIT WINDOW
BCS 66$ ;BR IF OK
HLT 26 ;ERROR IN FLAG CHAR
66$: INC R3 ;INC BIT COUNT
CMP #10,R3 ;FLAG DONE YET?
BNE 64$ ;BR IF NO
CLR R3 ;CLEAR BIT COUNT
1$: DATACLK, 1 ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
RORB R2 ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
```



```

2868 015016 103005          BCC 2$          ;BR IF CARRY CLEAR
2869 015020 004737 032314  JSR PC,GETSI   ;GET THE WINDOW
2870 015024 103406          BCS 3$          ;BR IF BIT IS A MARK
2871 015026 104006          HLT 6           ;ERROR BIT WAS A SPACE
2872 015030 000404          BR 3$          ;CONTINE WITH TEST
2873 015032 004737 032314  2$: JSR PC,GETSI ;GET THE WINDOW
2874 015036 103001          BCC 3$          ;BR IF BIT IS A SPACE
2875 015040 104006          HLT 6           ;ERROR BIT WAS A MARK
2876 015042          3$:
2877 015042 005203          INC R3          ;NEXT BIT
2878 015044 022703 000010  CMP #10,R3     ;DONE YET?
2879 015050 001357          BNE 1$          ;BR IF NO
2880 015052 104415 000014  DATACLK, 14   ;CLOCK TRANSMITTER 14 MORE TICKS
2881 015056 104414          ROMCLK         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2882 015060 021264          021264        ;PORT4 LU-13
2883 015062 032761 000040 000004  BIT #BIT5,4(R1) ;RTS SHOULD BE CLEAR NOW
2884 015070 001401          BEQ 4$          ;BR IF YES
2885 015072 104034          HLT 34         ;ERROR, RTS NOT CLEAR
2886 015074 104400          4$: SCOPE      ;SCOPE THIS TEST

```

```

2887
2888
2889
2890          ;***** TEST 20 *****
2891          ;*BITSTUFF TRANSMITTER TEST
2892          ;*SINGLE CLOCK THE CHARACTER 125
2893          ;*CHECK FLAG AND DATA IN THE BIT WINDOW
2894          ;*VERIFY EACH BIT POSITION AS IT
2895          ;*PASSES THE BIT WINDOW (SI BIT)
2896          ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2897          ;*****

```

```

2898
2899          ; TEST 20
2900          ;-----
2901 015076 012737 000020 001226 TST20: MOV #20,TSTNO
2902 015104 012737 015360 001216  MOV #TST21,NEXT
2903
2904 015112 104412          MSTCLR         ;R1 CONTAINS BASE DMC11 ADDRESS
2905 015114 005061 000004  CLR 4(R1)      ;MASTER CLEAR DMC11
2906 015120 104414          ROMCLK         ;CLEAR PORT4
2907 015122 122117          122117        ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2908 015124 004737 033676  JSR PC,CLR10   ;PUT LINE UNIT IN BITSTUFF MODE
2909 015130 005037 034114  CLR BITCON     ;DO THIS AFTER MODE IS SET
2910 015134 012711 004000  MOV #BIT11,(R1) ;CONSECUTIVE 1'S COUNTER INIT TO 0
2911 015140 004737 032500  JSR PC,OUTRDY ;SET LINE UNIT LOOP
2912 015144 012761 000001 000004  MOV #1,4(R1)  ;WAIT FOR OUT-READY
2913 015152 104414          ROMCLK         ;SET BIT0 IN PORT4
2914 015154 122111          122111        ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2915 015156 104414          ROMCLK         ;SET SOM!
2916 015160 122110          122110        ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2917 015162 012705 000125  MOV #125,R5   ;LOAD GARBAGE CHAR
2918 015166 004737 032500  JSR PC,OUTRDY ;LOAD CHARACTER IN R5 FOR TYPEOUT
2919 015172 010561 000004  MOV R5,4(R1)  ;WAIT FOR OUT-READY
2920 015176 104414          ROMCLK         ;LOAD PORT4 WITH CHARACTER
2921 015200 122110          122110        ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2922 015202 004737 032346  JSR PC,OCOR   ;LOAD OUT DATA
2923 015206 005003          CLR R3        ;WAIT FOR OCOR TO SET
                ;CLEAR BIT COUNTER

```



```

2924 015210 010502          MOV     R5,R2          ;LOAD CHARACTER IN R2
2925 015212 104415 000002  DATACLK, 2          ;2 TICKS TO SET UP TRANSMITTER
2926 015216 012737 000176 001252  MOV     #*B<01111110>,TEMP3 ;PUT FLAG CHARACTER IN TEMP3
2927 015224 104415 000001 64$:  DATACLK, 1          ;CLOCK FLAG ONCE
2928 015230 106037 001252  RORB   TEMP3          ;SHIFT SOFT FLAG
2929 015234 103405          BCS    65$            ;BR IF BIT IS MARK
2930 015236 004737 032314  JSR    PC,GETSI       ;LOOK AT BIT WINDOW
2931 015242 103006          BCC    66$            ;BR IF OK
2932 015244 104026          HLT    26             ;ERROR IN FLAG CHAR
2933 015246 000404          BR     66$            ;
2934 015250 004737 032314 65$:  JSR    PC,GETSI       ;LOOK AT BIT WINDOW
2935 015254 103401          BCS    66$            ;BR IF OK
2936 015256 104026          HLT    26             ;ERROR IN FLAG CHAR
2937 015260 005203 66$:  INC    R3             ;INC BIT COUNT
2938 015262 022703 000010  CMP    #10,R3        ;FLAG DONE YET?
2939 015266 001356          BNE    64$            ;BR IF NO
2940 015270 005003          CLR    R3             ;CLEAR BIT COUNT
2941 015272 104415 000001 1$:  DATACLK, 1          ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2942 015276 106002          RORB   R2             ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
2943 015300 103005          BCC    2$             ;BR IF CARRY CLEAR
2944 015302 004737 032314  JSR    PC,GETSI       ;GET THE WINDOW
2945 015306 103406          BCS    3$             ;BR IF BIT IS A MARK
2946 015310 104006          HLT    6              ;ERROR BIT WAS A SPACE
2947 015312 000404          BR     3$             ;CONTINUE WITH TEST
2948 015314 004737 032314 2$:  JSR    PC,GETSI       ;GET THE WINDOW
2949 015320 103001          BCC    3$             ;BR IF BIT IS A SPACE
2950 015322 104006          HLT    6              ;ERROR BIT WAS A MARK
2951 015324 3$:
2952 015324 005203          INC    R3             ;NEXT BIT
2953 015326 022703 000010  CMP    #10,R3        ;DONE YET?
2954 015332 001357          BNE    1$             ;BR IF NO
2955 015334 104415 000014  DATACLK, 14          ;CLOCK TRANSMITTER 14 MORE TICKS
2956 015340 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2957 015342 021264          021264          ;PORT4 LU-13
2958 015344 032761 000040 000004  BIT    #BIT5,4(R1)    ;RTS SHOULD BE CLEAR NOW
2959 015352 001401          BEQ    4$             ;BR IF YES
2960 015354 104034          HLT    34             ;ERROR, RTS NOT CLEAR
2961 015356 104400 4$:  SCOPE          ;SCOPE THIS TEST

```

```

2962
2963
2964
2965 ***** TEST 21 *****
2966 *BITSTUFF TRANSMITTER TEST
2967 *SINGLE CLOCK THE CHARACTER 252
2968 *CHECK FLAG AND DATA IN THE BIT WINDOW
2969 *VERIFY EACH BIT POSITION AS IT
2970 *PASSES THE BIT WINDOW (SI BIT)
2971 *ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2972 *****

```

```

2973
2974 : TEST 21
2975 -----
2976 015360 012737 000021 001226 TST21: MOV    #21,TSTNO
2977 015366 012737 015642 001216  MOV    #TST22,NEXT
2978
2979 015374 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11

```



```

2980 015376 005061 000004      CLR      4(R1)      ;CLEAR PORT4
2981 015402 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2982 015404 122117      122117      ;PUT LINE UNIT IN BITSTUFF MODE
2983 015406 004737 033676      JSR      PC,CLRIO  ;DO THIS AFTER MODE IS SET
2984 015412 005037 034114      CLR      BITCON    ;CONSECUTIVE 1'S COUNTER INIT TO 0
2985 015416 012711 004000      MOV      #BIT11,(R1) ;SET LINE UNIT LOOP
2986 015422 004737 032500      JSR      PC,OUTRDY ;WAIT FOR OUT-READY
2987 015426 012761 000001 000004      MOV      #1,4(R1)  ;SET BIT0 IN PORT4
2988 015434 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2989 015436 122111      122111      ;SET SOM!
2990 015440 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2991 015442 122110      122110      ;LOAD GARBAGE CHAR
2992 015444 012705 000252      MOV      #252,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
2993 015450 004737 032500      JSR      PC,OUTRDY ;WAIT FOR OUT-READY
2994 015454 010561 000004      MOV      R5,4(R1)  ;LOAD PORT4 WITH CHARACTER
2995 015460 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2996 015462 122110      122110      ;LOAD OUT DATA
2997 015464 004737 032346      JSR      PC,OCOR   ;WAIT FOR OCOR TO SET
2998 015470 005003      CLR      R3        ;CLEAR BIT COUNTER
2999 015472 010502      MOV      R5,R2     ;LOAD CHARACTER IN R2
3000 015474 104415 000002      DATACLK, 2      ;2 TICKS TO SET UP TRANSMITTER
3001 015500 012737 000176 001252      MOV      #*B<01111110>,TEMP3 ;PUT FLAG CHARACTER IN TEMP3
3002 015506 104415 000001      DATACLK, 1      ;CLOCK FLAG ONCE
3003 015512 106037 001252      RORB    TEMP3     ;SHIFT SOFT FLAG
3004 015516 103405      BCS     65$      ;BR IF BIT IS MARK
3005 015520 004737 032314      JSR      PC,GETSI  ;LOOK AT BIT WINDOW
3006 015524 103006      BCC     66$      ;BR IF OK
3007 015526 104026      HLT     26       ;ERROR IN FLAG CHAR
3008 015530 000404      BR      66$
3009 015532 004737 032314      JSR      PC,GETSI  ;LOOK AT BIT WINDOW
3010 015536 103401      BCS     66$      ;BR IF OK
3011 015540 104026      HLT     26       ;ERROR IN FLAG CHAR
3012 015542 005203      66$: INC     R3        ;INC BIT COUNT
3013 015544 022703 000010      CMP     #10,R3    ;FLAG DONE YET?
3014 015550 001356      BNE     64$      ;BR IF NO
3015 015552 005003      CLR     R3        ;CLEAR BIT COUNT
3016 015554 104415 000001      1$: DATACLK, 1    ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
3017 015560 106002      RORB    R2        ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
3018 015562 103005      BCC     2$       ;BR IF CARRY CLEAR
3019 015564 004737 032314      JSR      PC,GETSI  ;GET THE WINDOW
3020 015570 103406      BCS     3$       ;BR IF BIT IS A MARK
3021 015572 104006      HLT     6        ;ERROR BIT WAS A SPACE
3022 015574 000404      BR      3$       ;CONTINUE WITH TEST
3023 015576 004737 032314      2$: JSR      PC,GETSI ;GET THE WINDOW
3024 015602 103001      BCC     3$       ;BR IF BIT IS A SPACE
3025 015604 104006      HLT     6        ;ERROR BIT WAS A MARK
3026 015606      3$:
3027 015606 005203      INC     R3        ;NEXT BIT
3028 015610 022703 000010      CMP     #10,R3    ;DONE YET?
3029 015614 001357      BNE     1$       ;BR IF NO
3030 015616 104415 000014      DATACLK, 14     ;CLOCK TRANSMITTER 14 MORE TICKS
3031 015622 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3032 015624 021264      021264      ;PORT4 LU-13
3033 015626 032761 000040 000004      BIT     #BIT5,4(R1) ;RTS SHOULD BE CLEAR NOW
3034 015634 001401      BEQ     4$       ;BR IF YES
3035 015636 104034      HLT     34       ;ERROR, RTS NOT CLEAR

```



```

3036 015640 104400 4$: SCOPE ;SCOPE THIS TEST
3037
3038
3039 ;***** TEST 22 *****
3040 ;*BIT STUFF TEST
3041 ;*THIS TEST CHECKS ZERO BIT STUFFING OF
3042 ;* THE TRANSMITTER IN THE BIT WINDOW
3043 ;:*****
3044
3045 ; TEST 22
3046 -----
3047 015642 012737 000022 001226 TST22: MOV #22,TSTNO
3048 015650 012737 016152 001216 MOV #TST23,NEXT
3049 ;R1 CONTAINS BASE DMC11 ADDRESS
3050 015656 104412 MSTCLR ;MASTER CLEAR DMC11
3051 015660 005061 000004 CLR 4(R1) ;CLEAR PORT4
3052 015664 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3053 015666 122117 122117 ;PUT LINE UNIT IN BITSTUFF MODE
3054 015670 004737 033676 JSR PC,CLRIO ;DO THIS AFTER MODE IS SET
3055 015674 012711 004000 MOV #BIT11,(R1) ;SET LU LOOP
3056 015700 004737 032500 JSR PC,OUTRDY ;WAIT FOR OUT-READY
3057 015704 012761 000001 000004 MOV #1,4(R1) ;SET BIT0 IN PORT4
3058 015712 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3059 015714 122111 122111 ;SET SOM!
3060 015716 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3061 015720 122110 122110 ;LOAD GARBAGE CHAR
3062 015722 004537 033634 JSR R5,MESLD ;LOAD OUT SILO DATA
3063 015726 034142 STUFDT ;MESSAGE ADDRESS
3064 015730 000024 20. ;NUMBER OF CHARACTERS
3065 015732 012704 034142 MOV #STUFDT,R4 ;R4=CHARACTER POINTER
3066 015736 005003 CLR R3 ;R3= BIT COUNTER
3067 015740 012700 000006 MOV #6,R0 ;BIT COUNTER FOR FLAG CHARACTER
3068 015744 104415 000002 DATACLK, 2 ;SET UP TRANSMITTER
3069 015750 012737 000176 001252 MOV #*B<01111110>,TEMP3 ;PUT FLAG CHARACTER IN TEMP3
3070 015756 104415 000001 64$: DATACLK, 1 ;CLOCK FLAG ONCE
3071 015762 106037 001252 RORB TEMP3 ;SHIFT SOFT FLAG
3072 015766 103405 BCS 65$ ;BR IF BIT IS MARK
3073 015770 004737 032314 JSR PC,GETSI ;LOOK AT BIT WINDOW
3074 015774 103006 BCC 66$ ;BR IF OK
3075 015776 104026 HLT 26 ;ERROR IN FLAG CHAR
3076 016000 000404 BR 66$
3077 016002 004737 032314 65$: JSR PC,GETSI ;LOOK AT BIT WINDOW
3078 016006 103401 BCS 66$ ;BR IF OK
3079 016010 104026 HLT 26 ;ERROR IN FLAG CHAR
3080 016012 005203 66$: INC R3 ;INC BIT COUNT
3081 016014 022703 000010 CMP #10,R3 ;FLAG DONE YET?
3082 016020 001356 BNE 64$ ;BR IF NO
3083 016022 005003 CLR R3 ;CLEAR BIT COUNT
3084 016024 012700 000024 MOV #20.,R0 ;R0=CHARACTER COUNTER
3085 016030 005037 034114 CLR BITCON ;CLEAR BIT STUFF COUNTER
3086 016034 112405 3$: MOVB (R4)+,R5 ;LOAD CHARACTER IN R5
3087 016036 010502 MOV R5,R2 ;LOAD CHARACTER IN R2
3088 016040 104415 000001 4$: DATACLK, 1 ;SHIFT DTAT ONCE
3089 016044 106002 RORB R2 ;SHIFT SOFT DATA
3090 016046 103407 BCS 5$ ;BR IF CARRY SET
3091 016050 005037 034114 CLR BITCON ;CLEAR BIT STUFF COUNTER

```

```

3092 016054 004737 032314 JSR PC,GETSI ;LOOK AT WINDOW
3093 016060 103010 BCC 6$ ;BR IF SPACE
3094 016062 104006 HLT 6 ;ERROR, WINDOW WAS A MARK
3095 016064 000406 BR 6$ ;CONTINUE
3096 016066 005237 034114 5$: INC BITCON ;ADD 1 TO BIT STUFF COUNTER
3097 016072 004737 032314 JSR PC,GETSI ;LOOK AT WINDOW
3098 016076 103401 BCS 6$ ;BR IF MARK
3099 016100 104006 HLT 6 ;ERROR, WINDOW WAS A SPACE
3100 016102 022737 000005 034114 6$: CMP #5,BITCON ;HAVE THERE BEEN 5 1'S IN A ROW
3101 016110 001010 BNE 7$ ;BR IF NO
3102 016112 005037 034114 CLR BITCON ;IF YES CLR BIT STUFF COUNTER
3103 016116 104415 000001 DATACLK, 1 ;AND CLOCK TRANSMITTER ONCE
3104 016122 004737 032314 JSR PC,GETSI ;CHECK WINDOW FOR A ZEOR STUFF!!
3105 016126 103001 BCC 7$ ;BR IF WINDOW IS A SPACE
3106 016130 104030 HLT 30 ;ERROR, TRANSMITTER DID NOT STUFF A ZERO
3107 016132 005203 7$: INC R3 ;BUMP BIT COUNTER
3108 016134 022703 000010 CMP #10,R3 ;DONE THIS CHARACTER YET?
3109 016140 001337 BNE 4$ ;BR IF NO
3110 016142 005003 CLR R3 ;RESTART BIT COUNTER AT ZERO
3111 016144 005300 DEC R0 ;DEC CHARACTER COUNTER
3112 016146 001332 BNE 3$ ;BR IF NOT DONE YET
3113 016150 104400 8$: SCOPE ;SCOPE THIS TEST

```

```

3114
3115
3116 :***** TEST 23 *****
3117 :*BITSTUFF TRANSMITTER TEST
3118 :*SINGLE CLOCK THE CHARACTER 377
3119 :*CHECK FLAG AND DATA IN THE BIT WINDOW
3120 :*VERIFY EACH BIT POSITION AS IT
3121 :*PASSES THE BIT WINDOW (SI BIT)
3122 :*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
3123 :*****

```

```

3124
3125 : TEST 23
3126 :-----
3127 016152 012737 000023 001226 TST23: MOV #23,TSTNO
3128 016160 012737 016460 001216 MOV #TST24,NEXT
3129
3130 016166 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3131 016170 005061 000004 CLR 4(R1) ;MASTER CLEAR DMC11
3132 016174 104414 ROMCLK ;CLEAR PORT4
122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3133 016176 122117 ;PUT LINE UNIT IN BITSTUFF MODE
3134 016200 004737 033676 JSR PC,CLRIO ;DO THIS AFTER MODE IS SET
3135 016204 005037 034114 CLR BITCON ;CONSECUTIVE 1'S COUNTER INIT TO 0
3136 016210 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
3137 016214 004737 032500 JSR PC,OUTRDY ;WAIT FOR OUT-READY
3138 016220 012761 000001 000004 MOV #1,4(R1) ;SET BIT0 IN PORT4
3139 016226 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122111 ;SET SOM!
3140 016230 122111 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3141 016232 104414 ROMCLK
122110 ;LOAD GARBAGE CHAR
3142 016234 122110
3143 016236 012705 000377 MOV #377,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
3144 016242 010537 016414 MOV R5,5$ ;LOAD CHAR FOR STUFF CHECK
3145 016246 004737 032500 JSR PC,OUTRDY ;WAIT FOR OUT-READY
3146 016252 010561 000004 MOV R5,4(R1) ;LOAD PORT4 WITH CHARACTER
3147 016256 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

```



```

3148 016260 122110          122110          ;LOAD OUT DATA
3149 016262 004737 032346  JSR    PC,OCOR      ;WAIT FOR OCOR TO SET
3150 016266 005003          CLR    R3           ;CLEAR BIT COUNTER
3151 016270 010502          MOV    R5,R2       ;LOAD CHARACTER IN R2
3152 016272 104415 000002  DATACLK,          ;2 TICKS TO SET UP TRANSMITTER
3153 016276 012737 000176 001252  MOV    #^B<01111110>,TEMP3 ;PUT FLAG CHARACTER IN TEMP3
3154 016304 104415 000001 64$:  DATACLK,          ;CLOCK FLAG ONCE
3155 016310 106037 001252  RORB  TEMP3        ;SHIFT SOFT FLAG
3156 016314 103405          BCS   65$          ;BR IF BIT IS MARK
3157 016316 004737 032314  JSR    PC,GETSI    ;LOOK AT BIT WINDOW
3158 016322 103006          BCC   66$          ;BR IF OK
3159 016324 104026          HLT   26           ;ERROR IN FLAG CHAR
3160 016326 000404          BR    66$
3161 016330 004737 032314 65$:  JSR    PC,GETSI    ;LOOK AT BIT WINDOW
3162 016334 103401          BCS   66$          ;BR IF OK
3163 016336 104026          HLT   26           ;ERROR IN FLAG CHAR
3164 016340 005203          INC   R3           ;INC BIT COUNT
3165 016342 022703 000010 66$:  CMP    #10,R3      ;FLAG DONE YET?
3166 016346 001356          BNE   64$          ;BR IF NO
3167 016350 005003          CLR   R3           ;CLEAR BIT COUNT
3168 016352 005037 034114  CLR   BITCON      ;CLEAR STUFF COUNT
3169 016356 104415 000001 1$:  DATACLK,          ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
3170 016362 106002          RORB  R2           ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
3171 016364 103005          BCC   2$           ;BR IF CARRY CLEAR
3172 016366 004737 032314  JSR    PC,GETSI    ;GET THE WINDOW
3173 016372 103406          BCS   3$           ;BR IF BIT IS A MARK
3174 016374 104006          HLT   6           ;ERROR BIT WAS A SPACE
3175 016376 000404          BR    3$           ;CONTINUE WITH TEST
3176 016400 004737 032314 2$:  JSR    PC,GETSI    ;GET THE WINDOW
3177 016404 103001          BCC   3$           ;BR IF BIT IS A SPACE
3178 016406 104006          HLT   6           ;ERROR BIT WAS A MARK
3179 016410          3$:
3180 016410 004537 033776  JSR    R5,STFFCK   ;CHECK FOR BIT STUFF
3181 016414 000377 5$:  377          ;DATA CHARACTER
3182 016416 000001          1           ;SHIFT COUNT
3183 016420 010237 016414  MOV    R2,5$       ;LOAD CHAR FOR STUFF CHECK
3184 016424 005203          INC   R3           ;NEXT BIT
3185 016426 022703 000010  CMP    #10,R3      ;DONE YET?
3186 016432 001351          BNE   1$           ;BR IF NO
3187 016434 104415 000014  DATACLK,          ;CLOCK TRANSMITTER 14 MORE TICKS
3188 016440 104414          ROMCLK           ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3189 016442 021264          021264          ;PORT4 LU-13
3190 016444 032761 000040 000004  BIT    #BIT5,4(R1) ;RTS SHOULD BE CLEAR NOW
3191 016452 001401          BEQ   4$           ;BR IF YES
3192 016454 104034          HLT   34          ;ERROR, RTS NOT CLEAR
3193 016456 104400 4$:  SCOPE          ;SCOPE THIS TEST

```

3194  
3195  
3196  
3197  
3198  
3199  
3200  
3201  
3202  
3203

```

:***** TEST 24 *****
:*BITSTUFF 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
:*****

```

```

3204
3205
3206 ; TEST 24
3207 016460 012737 000024 001226 TST24: MOV #24,TSTNO
3208 016466 012737 017012 001216 MOV #TST25,NEXT
3209
3210 016474 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3211 016476 005061 000004 CLR 4(R1) ;MASTER CLEAR DMC11
3212 016502 104414 ROMCLK ;CLEAR PORT4
3213 016504 122117 122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3214 016506 004737 033676 JSR PC,CLRIO ;PUT LINE UNIT IN BITSTUFF MODE
3215 016512 005037 034114 CLR BITCON ;DO THIS AFTER MODE IS SET
3216 016516 012711 004000 MOV #BIT11,(R1) ;CONSECUTIVE 1'S COUNTER INIT TO 0
3217 016522 005003 CLR R3 ;SET LINE UNIT LOOP
3218 016524 005004 CLR R4 ;R3 CONTAINS BIT COUNT
3219 016526 005005 CLR R5 ;R4 CONTAINS CHAR TO BE LOADED IN SILO
3220 016530 004737 032500 JSR PC,OUTRDY ;R5 CONTAINS CHARACTER CURRENTLY BEING SHIFTED OUT
3221 016534 012761 000001 000004 MOV #1,4(R1) ;WAIT FOR OUT-READY
3222 016542 104414 ROMCLK ;SET BIT0 IN PORT4
3223 016544 122111 122111 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3224 016546 104414 ROMCLK ;SET SOM!
3225 016550 122110 122110 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3226 016552 004737 032500 JSR PC,OUTRDY ;LOAD GARBAGE CHAR
3227 016556 010461 000004 MOV R4,4(R1) ;WAIT FOR OUT-READY
3228 016562 104414 ROMCLK ;LOAD PORT4 WITH CHARACTER
3229 016564 122110 122110 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3230 016566 005204 INC R4 ;LOAD OUT DATA
3231 016570 004737 032500 JSR PC,OUTRDY ;INCREMENT TO NEXT CHARACTER
3232 016574 010461 000004 MOV R4,4(R1) ;WAIT FOR OUT-READY
3233 016600 104414 ROMCLK ;LOAD PORT4 WITH CHARACTER
3234 016602 122110 122110 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3235 016604 004737 032346 JSR PC,OCOR ;LOAD OUT DATA
3236 016610 104415 000002 DATACLK, 2 ;WAIT FOR OCOR TO SET
3237 016614 012737 000176 001252 MOV #^B<01111110>,TEMP3 ;2 TICKS TO SET UP TRANSMITTER
3238 016622 104415 000001 64$: DATACLK, 1 ;PUT FLAG CHARACTER IN TEMP3
3239 016626 106037 001252 RORB TEMP3 ;CLOCK FLAG ONCE
3240 016632 103405 BCS 65$ ;SHIFT SOFT FLAG
3241 016634 004737 032314 JSR PC,GETSI ;BR IF BIT IS MARK
3242 016640 103006 BCC 66$ ;LOOK AT BIT WINDOW
3243 016642 104026 HLT 26 ;BR IF OK
3244 016644 000404 BR 66$ ;ERROR IN FLAG CHAR
3245 016646 004737 032314 65$: JSR PC,GETSI ;LOOK AT BIT WINDOW
3246 016652 103401 BCS 66$ ;BR IF OK
3247 016654 104026 HLT 26 ;ERROR IN FLAG CHAR
3248 016656 005203 66$: INC R3 ;INC BIT COUNT
3249 016660 022703 000010 CMP #10,R3 ;FLAG DONE YET?
3250 016664 001356 BNE 64$ ;BR IF NO
3251 016666 005003 CLR R3 ;CLEAR BIT COUNT
3252 016670 005037 034114 CLR BITCON ;CLEAR BIT STUFF COUNTER
3253 016674 005003 4$: CLR R3 ;CLEAR BIT COUNTER
3254 016676 010502 MOV R5,R2 ;LOAD CHARACTER IN R2
3255 016700 010237 016742 MOV R2,6$ ;LOAD CHAR FOR STUFF CHECK
3256 016704 104415 000001 1$: DATACLK, 1 ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
3257 016710 106002 RORB R2 ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
3258 016712 103005 BCC 2$ ;BR IF CARRY CLEAR
3259 016714 004737 032314 JSR PC,GETSI ;GET THE WINDOW

```



```

3260 016720 103406          BCS      3$          ;BR IF BIT IS A MARK
3261 016722 104006          HLT      6          ;ERROR BIT WAS A SPACE
3262 016724 000404          BR       3$          ;CONTINUE WITH TEST
3263 016726 004737 032314  2$:      JSR      PC,GETSI ;GET THE WINDOW
3264 016732 103001          BCC     3$          ;BR IF BIT IS A SPACE
3265 016734 104006          HLT      6          ;ERROR BIT WAS A MARK
3266 016736                3$:
3267 016736 004537 033776  JSR      R5,STFFCK ;CHECK FOR BIT STUFF
3268 016742 000000          6$:      0          ;DATA CHARACTER
3269 016744 000001          1          ;SHIFT COUNT
3270 016746 010237 016742  MOV      R2,6$      ;LOAD CHAR FOR STUFF CHECK
3271 016752 005203          INC      R3          ;NEXT BIT
3272 016754 022703 000010  CMP      #10,R3     ;DONE YET?
3273 016760 001351          BNE     1$          ;BR IF NO
3274 016762 005204          INC      R4          ;NEXT CHARACTER
3275 016764 004737 032500  JSR      PC,OUTRDY ;WAIT FOR OUT-READY
3276 016770 010461 000004  MOV      R4,4(R1)   ;LOAD PORT4 WITH CHARACTER
3277 016774 104414          ROMCLK   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3278 016776 122110          122110 ;LOAD OUT DATA
3279 017000 005205          INC      R5          ;NEXT CHARACTER
3280 017002 022705 000400  CMP      #400,R5    ;DONE YET?
3281 017006 001332          BNE     4$          ;BR IF NO
3282 017010 104400          5$:      SCOPE        ;SCOPE THIS TEST
3283
3284
3285
3286
3287
3288
3289
3290
3291
3292
3293
3294

```

```

:***** TEST 25 *****
: *MULTIPLE FLAG AND TRANSMITTER ABORT TEST
: *LOAD SILO WITH 5 FLAGS AND A CHAR (000)
: *VERIFY IN THE BIT WINDOW THAT THE FLAGS
: *AND DATA ARE CORRECT AND FOLLOWED BY AN ABORT
: *SEQUENCE (8 CONTIGUOUS 1'S)
:*****

```

```

3295 017012 012737 000025 001226 TST25: MOV      #25,TSTNO
3296 017020 012737 017300 001216 MOV      #TST26,NEXT
3297
3298 017026 104412          MSTCLR   ;R1 CONTAINS BASE DMC11 ADDRESS
3299 017030 005061 000004          CLR      4(R1)    ;MASTER CLEAR DMC11
3300 017034 104414          ROMCLK   ;CLEAR PORT4
3301 017036 122117          122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3302 017040 004737 033676  JSR      PC,CLRIO  ;PUT LINE UNIT IN BITSTUFF MODE
3303 017044 012711 004000  MOV      #BIT11,(R1) ;DO THIS AFTER MODE IS SET
3304 017050 012700 000005  MOV      #5,R0     ;SET LU LOOP
3305 017054 005003          CLR      R3        ;FLAG COUNT
3306 017056 004737 032500  1$:      JSR      PC,OUTRDY ;CLEAR BIT COUNTER
3307 017062 012761 000001 000004  MOV      #1,4(R1)  ;WAIT FOR OUT-READY
3308 017070 104414          ROMCLK   ;SET BIT0 IN PORT4
3309 017072 122111          122111 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3310 017074 104414          ROMCLK   ;SET SOM!
3311 017076 122110          122110 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3312 017100 005300          DEC      R0        ;LOAD GARBAGE CHAR
3313 017102 001365          BNE     1$        ;DEC COUNT
3314 017104 004737 032500  JSR      PC,OUTRDY ;LOAD ANOTHER
3315 017110 005061 000004          CLR      4(R1)    ;WAIT FOR OUTRDY
                                     ;CLEAR PORT4

```



```

3316 017114 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3317 017116 122110 ;LOAD A ZERO
3318 017120 004737 032346 JSR PC,OCOR ;WAIT
3319 017124 012700 000005 MOV #5,RO ;RO = FLAG COUNT
3320 017130 104415 000002 DATACLK, 2 ;SET UP TRANSMITTER
3321 017134
3322 017134 012737 000176 001252 2$: MOV #^B<01111110>,TEMP3 ;PUT FLAG CHARACTER IN TEMP3
3323 017142 104415 000001 64$: DATACLK, 1 ;CLOCK FLAG ONCE
3324 017146 106037 001252 RORB TEMP3 ;SHIFT SOFT FLAG
3325 017152 103405 BCS 65$ ;BR IF BIT IS MARK
3326 017154 004737 032314 JSR PC,GETSI ;LOOK AT BIT WINDOW
3327 017160 103006 BCC 66$ ;BR IF OK
3328 017162 104026 HLT 26 ;ERROR IN FLAG CHAR
3329 017164 000404 BR 66$
3330 017166 004737 032314 65$: JSR PC,GETSI ;LOOK AT BIT WINDOW
3331 017172 103401 BCS 66$ ;BR IF OK
3332 017174 104026 HLT 26 ;ERROR IN FLAG CHAR
3333 017176 005203 66$: INC R3 ;INC BIT COUNT
3334 017200 022703 000010 CMP #10,R3 ;FLAG DONE YET?
3335 017204 001356 BNE 64$ ;BR IF NO
3336 017206 005003 CLR R3 ;CLEAR BIT COUNT
3337 017210 005300 DEC RO ;DEC COUNT
3338 017212 001350 BNE 2$ ;BR IF NOT DONE
3339 017214 005003 CLR R3 ;R3 = BIT COUNT
3340 017216 005005 CLR R5 ;R5 = 'EXPECTED'
3341 017220 104415 000001 3$: DATACLK, 1 ;CLOCK ONCE
3342 017224 004737 032314 JSR PC,GETSI ;GO LOOK AT WINDOW
3343 017230 103001 BCC 4$ ;BR IF A SPACE
3344 017232 104006 HLT 6 ;ERROR, A MARK WAS SEEN
3345 017234 005203 4$: INC R3 ;INC BIT COUNT
3346 017236 022703 000010 CMP #10,R3 ;DONE YET?
3347 017242 001366 BNE 3$ ;BR IF NO
3348 017244 005003 CLR R3 ;CLEAR BIT COUNT
3349 017246 012705 000377 MOV #377,R5 ;R5 = 'EXPECTED'
3350 017252 104415 000001 5$: DATACLK, 1 ;CLOCK ONCE
3351 017256 004737 032314 JSR PC,GETSI ;LOOK AT WINDOW
3352 017262 103401 BCS 6$ ;BR IF A MARY
3353 017264 104033 HLT 33 ;ERROR, A SPACE WAS SEEN
3354 017266 005203 6$: INC R3 ;INC BIT COUNT
3355 017270 022703 000010 CMP #10,R3 ;DONE YET?
3356 017274 001366 BNE 5$ ;BR IF NO
3357 017276 104400 SCOPE ;SCOPE THIS TEST
3358
3359
3360 ;***** TEST 26 *****
3361 ;*LEADING ZEROS TEST
3362 ;*VERIFY THAT THE SETTING OF SOM AND EOM TOGETHER
3363 ;*AND THEN SOM ALONE WILL GENERATE 16 LEADING ZEROS
3364 ;*AND A FLAG,THE CHECK IS MADE USING THE BIT WINDOW
3365 ;:*****
3366
3367 ; TEST 26
3368 ;-----
3369 017300 012737 000026 001226 TST26: MOV #26,TSTNO
3370 017306 012737 017520 001216 MOV #TST27,NEXT
3371 ;R1 CONTAINS BASE DMC11 ADDRESS

```



```

3372 017314 104412          MSTCLR          ;MASTER CLEAR DMC11
3373 017316 005061 000004   CLR          4(R1) ;CLEAR PORT4
3374 017322 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3375 017324 122117          122117         ;SET TO BITSTUFF MODE
3376 017326 004737 033676   JSR          PC,CLRIO ;DO THIS AFTER MODE IS SET
3377 017332 012711 004000   MOV          #BIT11,(R1) ;SET LU LOOP
3378 017336 004737 032500   JSR          PC,OUTRDY ;WAIT FOR OUTRDY
3379 017342 012761 000003 000004   MOV          #3,4(R1) ;LOAD PORT4
3380 017350 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3381 017352 122111          122111         ;SET SOM & EOM
3382 017354 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3383 017356 122110          122110         ;GARBAGE CHARACTER
3384 017360 012761 000001 000004   MOV          #1,4(R1) ;LOAD PORT4
3385 017366 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3386 017370 122111          122111         ;SET SOM
3387 017372 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3388 017374 122110          122110         ;GARBAGE CHAR
3389 017376 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3390 017400 122110          122110         ;GARBAGE CHAR
3391 017402 004737 032346   JSR          PC,OCOR  ;WAIT FOR OCOR
3392 017406 005000          CLR          R0      ;R0 = BIT COUNT
3393 017410 104415 000002   DATACLK,2 ;SET UP TRANSMITTER
3394 017414 104415 000001   1$: DATACLK,1 ;SINGLE CLOCK TRANSMITTER
3395 017420 004737 032314   JSR          PC,GETSI ;LOOK AT BITWINDOW
3396 017424 103001          BCC          .+4
3397 017426 104041          HLT          41      ;ERROR WINDOW WAS A MARK
3398 017430 005200          INC          R0
3399 017432 022700 000020   CMP          #16.,R0 ;16 ZEROS YET?
3400 017436 001366          BNE          1$      ;BR IF NO
3401 017440 005003          CLR          R3      ;R3 = BIT COUNT
3402 017442 012737 000176 001252   MOV          #^B<01111110>,TEMP3 ;PUT FLAG CHARACTER IN TEMP3
3403 017450 104415 000001 64$: DATACLK, 1 ;CLOCK FLAG ONCE
3404 017454 106037 001252   RORB        TEMP3   ;SHIFT SOFT FLAG
3405 017460 103405          BCS          65$    ;BR IF BIT IS MARK
3406 017462 004737 032314   JSR          PC,GETSI ;LOOK AT BIT WINDOW
3407 017466 103006          BCC          66$    ;BR IF OK
3408 017470 104026          HLT          26     ;ERROR IN FLAG CHAR
3409 017472 000404          BR          66$
3410 017474 004737 032314 65$: JSR          PC,GETSI ;LOOK AT BIT WINDOW
3411 017500 103401          BCS          66$    ;BR IF OK
3412 017502 104026          HLT          26     ;ERROR IN FLAG CHAR
3413 017504 005203 66$: INC          R3      ;INC BIT COUNT
3414 017506 022703 000010   CMP          #10,R3 ;FLAG DONE YET?
3415 017512 001356          BNE          64$    ;BR IF NO
3416 017514 005003          CLR          R3      ;CLEAR BIT COUNT
3417 017516 104400          SCOPE          ;SCOPE THIS TEST

```

```

3418
3419
3420 ;***** TEST 27 *****
3421 ;*BITSTUFF STRIP FLAG TEST
3422 ;*SET LU LOOP, SINGLE STEP 5 FLAGS,
3423 ;*VERIFY THAT IN ACTIVE DOES NOT SET
3424 ;:*****
3425
3426 ; TEST 27
3427 ;-----

```



```

3428 017520 012737 000027 001226 TST27: MOV #27,TSTNO
3429 017526 012737 017622 001216 MOV #TST30,NEXT
3430
3431 017534 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3432 017536 005061 000004 CLR 4(R1) ;MASTER CLEAR DMC11
3433 017542 104414 ROMCLK ;CLEAR PORT4
122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3434 017544 122117 ;PUT LINE UNIT IN BITSTUFF MODE
3435 017546 004737 033676 JSR PC,CLRIO ;DO THIS AFTER MODE IS SET
3436 017552 012711 004000 MOV #BIT11,(R1) ;SET LU LOOP
3437 017556 012702 000012 MOV #12,R2 ;SAVE LU REG FOR TYPEOUT
3438 017562 004737 032364 JSR PC,SYNC ;SINGLE CLOCK 5 SYNC CHARACTERS
3439 017566 000005 5
3440 017570 104415 000054 DATACLK, 54
3441 017574 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021244 ;PORT4 LU12
3442 017576 021244 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
3443 017600 016104 000004 BIC #277,R4 ;CLEAR UNWANTED BITS
3444 017604 042704 000277 CLR R5 ;PUT 'EXPECTED' IN R5
3445 017610 005005 CMPB R5,R4 ;IS ACTIVE CLEAR?
3446 017612 120504 BEQ 1$ ;BR IF YES
3447 017614 001401 HLT 40 ;ERROR ACTIVE IS NOT CLEAR
3448 017616 104040
3449 017620 104400 1$: SCOPE ;SCOPE THIS TEST

```

```

3450
3451
3452 ;***** TEST 30 *****
3453 ;*BITSTUFF IN ACTIVE TEST
3454 ;*SET LU LOOP, SINGLE STEP 5 FLAGS AND A NON-FLAG (301)
3455 ;*VERIFY THAT IN ACTIVE IS SET
3456 ;*****
3457

```

```

3458 ; TEST 30
3459 ;-----
3460 017622 012737 000030 001226 TST30: MOV #30,TSTNO
3461 017630 012737 017726 001216 MOV #TST31,NEXT
3462
3463 017636 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3464 017640 005061 000004 CLR 4(R1) ;MASTER CLEAR DMC11
3465 017644 104414 ROMCLK ;CLEAR PORT4
122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3466 017646 122117 ;PUT LINE UNIT IN BITSTUFF MODE
3467 017650 004737 033676 JSR PC,CLRIO ;DO THIS AFTER MODE IS SET
3468 017654 012711 004000 MOV #BIT11,(R1) ;SET LU LOOP
3469 017660 012702 000012 MOV #12,R2 ;SAVE LU REG FOR TYPEOUT
3470 017664 004737 032364 JSR PC,SYNC ;SINGLE CLOCK 5 SYNC CHARACTERS
3471 017670 000005 5
3472 017672 104415 000064 DATACLK, 64
3473 017676 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021244 ;PORT4 LU12
3474 017700 021244 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
3475 017702 016104 000004 BIC #277,R4 ;CLEAR UNWANTED BITS
3476 017706 042704 000277 MOV #BIT6,R5 ;PUT 'EXPECTED' IN R5
3477 017712 012705 000100 CMPB R5,R4 ;IS ACTIVE SET?
3478 017716 120504 BEQ 1$ ;BR IF YES
3479 017720 001401 HLT 40 ;ERROR ACTIVE IS NOT SET
3480 017722 104040 1$: SCOPE ;SCOPE THIS TEST
3481 017724 104400
3482
3483

```



```

3484
3485
3486
3487
3488
3489
3490
3491
3492 017726 012737 000031 001226 TST31:
3493 017734 012737 020064 001216
3494
3495 017742 104412
3496 017744 005061 000004
3497 017750 104414
3498 017752 122117
3499 017754 004737 033676
3500 017760 012711 004000
3501 017764 012702 000012
3502 017770 004737 032500
3503 017774 012761 000001 000004
3504 020002 104414
3505 020004 122111
3506 020006 104414
3507 020010 122110
3508 020012 012761 000301 000004
3509 020020 104414
3510 020022 122110
3511 020024 004737 032346
3512 020030 104415 000023
3513 020034 104414
3514 020036 021244
3515 020040 016104 000004
3516 020044 042704 000277
3517 020050 012705 000100
3518 020054 120504
3519 020056 001401
3520 020060 104040
3521 020062 104400 1$:
3522
3523
3524
3525
3526
3527
3528
3529
3530
3531
3532 020064 012737 000032 001226 TST32:
3533 020072 012737 020170 001216
3534
3535 020100 104412
3536 020102 005061 000004
3537 020106 104414
3538 020110 122117
3539 020112 004737 033676

```

```

:***** TEST 31 *****
:*BITSTUFF IN ACTIVE TEST
:*SET LINE UNIT LOOP,SINGLE STEP ONE FLAG AND A CHAR (301)
:*VERIFY THAT IN ACTIVE IS SET
:*****

```

```

: TEST 31
:-----
MOV #31,TSTNO
MOV #TST32,NEXT
MSTCLR
CLR 4(R1)
ROMCLK
122117
JSR PC,CLRIO
MOV #BIT11,(R1)
MOV #12,R2
JSR PC,OUTRDY
MOV #1,4(R1)
ROMCLK
122111
ROMCLK
122110
MOV #301,4(R1)
ROMCLK
122110
JSR PC,OCOR
DATACLK, 23
ROMCLK
021244
MOV 4(R1),R4
BIC #277,R4
MOV #BIT6,R5
CMPB R5,R4
BEQ 1$
HLT 40
1$: SCOPE

```

```

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;CLEAR PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PUT LINE UNIT IN BITSTUFF MODE
;MUST DO THIS AFTER MODE IS SET
;SAVE REG ADDRESS FOR TYPEOUT
;WAIT FOR OUTRDY
;LOAD PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;SET SOM
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;LOAD GARBAGE CHAR
;LOAD PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;LOAD OUT DATA
;WAIT FOR OCOR
;SINGLE CLOCK THE DATA
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PORT4 LU-12
;PUT "FOUND" IN R4
;CLEAR UNWANTED BITS
;PUT "EXPECTED" IN R5
;IS IN ACTIVE SET?
;ERROR, IN ACTIVE NOT SET
;SCOPE THIS TEST

```

```

:***** TEST 32 *****
:*BITSTUFF IN ACTIVE TEST
:*SET LU LOOP, SINGLE STEP 2 FLAGS AND A NON-FLAG (301)
:*VERIFY THAT IN ACTIVE IS SET
:*****

```

```

: TEST 32
:-----
MOV #32,TSTNO
MOV #TST33,NEXT
MSTCLR
CLR 4(R1)
ROMCLK
122117
JSR PC,CLRIO

```

```

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;CLEAR PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PUT LINE UNIT IN BITSTUFF MODE
;DO THIS AFTER MODE IS SET

```



```

3540 020116 012711 004000      MOV    #BIT11,(R1)    ;SET LU LOOP
3541 020122 012702 000012      MOV    #12,R2        ;SAVE LU REG FOR TYPEOUT
3542 020126 004737 032364      JSR    PC,SYNC       ;SINGLE CLOCK 2 SYNC CHARACTERS
3543 020132 000002                2
3544 020134 104415 000033      DATACLK,           33
3545 020140 104414                ROMCLK
3546 020142 021244                021244      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3547 020144 016104 000004      MOV    4(R1),R4      ;PORT4 LU12
3548 020150 042704 000277      BIC    #277,R4       ;PUT 'FOUND' IN R4
3549 020154 012705 000100      MOV    #BIT6,P5      ;CLEAR UNWANTED BITS
3550 020160 120504                CMPB   R5,R4         ;PUT 'EXPECTED' IN R5
3551 020162 001401                BEQ    1$            ;IS ACTIVE SET?
3552 020164 104040                HLT    40            ;BR IF YES
3553 020166 104400                1$: SCOPE           ;ERROR ACTIVE IS NOT SET
3554
3555
3556
3557
3558
3559
3560
3561
3562
3563
3564
3565 020170 012737 000033 001226 TST33: MOV    #33,TSTNO
3566 020176 012737 020374 001216      MOV    #TST34,NEXT
3567
3568 020204 104412                MSTCLR
3569 020206 005061 000004      CLR    4(R1)         ;R1 CONTAINS BASE DMC11 ADDRESS
3570 020212 104414                ROMCLK              ;MASTER CLEAR DMC11
3571 020214 122117                122117             ;CLEAR PORT4
3572 020216 004737 033676      JSR    PC,CLRIO      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3573 020222 012702 000012      MOV    #12,R2        ;PUT LINE UNIT IN BITSTUFF MODE
3574 020226 012711 004000      MOV    #BIT11,(R1)   ;DO THIS AFTER MODE IS SET
3575 020232 012761 000001 000004      MOV    #1,4(R1)      ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3576 020240 104414                ROMCLK              ;SET LINE UNIT LOOP
3577 020242 122111                122111             ;SET BIT0 IN PORT4
3578 020244 104414                ROMCLK              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3579 020246 122110                122110             ;LOAD GARBAGE CHAR
3580 020250 004737 032644      JSR    PC,CHARSD     ;LOAD SILO WITH CHARACTER
3581 020254 000026                26                 ;CHARACTER
3582 020256 104415 000033      DATACLK,           33 ;SINGLE CLOCK THE DATA
3583 020262 104416 000002      TIMER, 2            ;WAIT FOR INRDY
3584 020266 104414                ROMCLK              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3585 020270 021244                021244             ;PORT4 LU 12
3586 020272 016104 000004      MOV    4(R1),R4      ;PUT 'FOUND' IN R4
3587 020276 042704 000357      BIC    #357,R4       ;CLEAR UNWANTED BITS
3588 020302 012705 000020      MOV    #BIT4,R5      ;PUT 'EXPECTED' IN R5
3589 020306 120504                CMPB   R5,R4         ;IS INRDY SET?
3590 020310 001401                BEQ    1$            ;ERROR, INRDY IS NOT SET
3591 020312 104040                HLT    40
3592 020314
3593 020314 012761 000200 000004      1$: MOV    #BIT7,4(R1) ;LOAD PORT4
3594 020322 104414                ROMCLK              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3595 020324 122112                122112             ;SET IN CLEAR

```



```

3596 020326 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3597 020330 021244 021244 ;PORT4_LU 12
3598 020332 016104 000004 MOV 4(R1),R4 ;PUT "FOUND" IN R4
3599 020336 042704 000277 BIC #277,R4 ;CLEAR UNWANTED BITS
3600 020342 005005 CLR R5 ;PUT "EXPECTED" IN R5
3601 020344 120504 CMPB R5,R4 ;IS IN ACTIVE CLEAR?
3602 020346 001401 BEQ 2$
3603 020350 104040 HLT 40 ;ERROR, IN ACTIVE IS NOT CLEAR
3604 020352
3605 020352 016104 000004 2$: MOV 4(R1),R4 ;PUT "FOUND" IN R4
3606 020356 042704 000357 BIC #357,R4 ;CLEAR UNWANTED BITS
3607 020362 005005 CLR R5 ;PUT "EXPECTED" IN R5
3608 020364 120504 CMPB R5,R4 ;IS INRDY CLEARED?
3609 020366 001401 BEQ 3$
3610 020370 104040 HLT 40 ;ERROR, INRDY IS NOT CLEARED
3611 020372 104400 3$: SCOPE ;SCOPE THIS TEST
3612
3613
3614 ;***** TEST 34 *****
3615 ;*BITSTUFF BASIC RECEICER TEST
3616 ;*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 0
3617 ;*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
3618 ;*****
3619
3620 ; TEST 34
3621 ;-----
3622 020374 012737 000034 001226 TST34: MOV #34,TSTNO
3623 020402 012737 020542 001216 MOV #TST35,NEXT
3624
3625 020410 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3626 020412 005061 000004 CLR 4(R1) ;MASTER CLEAR DMC11
3627 020416 104414 ROMCLK ;CLEAR PORT4
3628 020420 122117 122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3629 020422 004737 033676 JSR PC,CLRIO ;PUT LINE UNIT IN BITSTUFF MODE
3630 020426 012702 000012 MOV #12,R2 ;DO THIS AFTER MODE IS SET
3631 020432 012711 004000 MOV #BIT11,(R1) ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3632 020436 012761 000001 000004 MOV #1,4(R1) ;SET LINE UNIT LOOP
3633 020444 104414 ROMCLK ;SET BIT0 IN PORT4
3634 020446 122111 122111 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3635 020450 104414 ROMCLK ;SET SOM!
3636 020452 122110 122110 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3637 020454 004737 032644 JSR PC,CHARSD ;LOAD GARBAGE CHAR
3638 020460 000000 0 CHARACTER ;LOAD SILO WITH CHARACTER
3639 020462 104415 000033 DATACLK, 33 ;CHARACTER
3640 020466 104416 000002 TIMER, 2 ;SINGLE CLOCK THE DATA
3641 020472 104414 ROMCLK ;WAIT FOR INRDY
3642 020474 021244 021244 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3643 020476 016104 000004 MOV 4(R1),R4 ;PORT4_LU 12
3644 020502 042704 000357 BIC #357,R4 ;PUT "FOUND" IN R4
3645 020506 012705 000020 MOV #BIT4,R5 ;CLEAR UNWANTED BITS
3646 020512 120504 CMPB R5,R4 ;PUT "EXPECTED" IN R5
3647 020514 001401 BEQ 1$ ;IS INRDY SET?
3648 020516 104040 HLT 40 ;ERROR, INRDY IS NOT SET
3649 020520
3650 020520 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3651 020522 021204 021204 ;PORT4_IN DATA

```

```

3652 020524 016104 000004      MOV    4(R1),R4      ;PUT "FOUND" IN R4
3653 020530 005005              CLR    R5            ;PUT "EXPECTED" IN R5
3654 020532 120504              CMPB  R5,R4         ;WAS A 0 RECEIVED?
3655 020534 001401              BEQ   2$            ;
3656 020536 104010              HLT   10            ;ERROR, RECEIVED DATA IS WRONG
3657 020540 104400              2$: SCOPE          ;SCOPE THIS TEST
3658
3659
3660
3661      ;***** TEST 35 *****
3662      ;*BITSTUFF BASIC RECEICER TEST
3663      ;*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 125
3664      ;*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
3665      ;*****
3666
3667      ; TEST 35
3668 020542 012737 000035 001226 TST35: MOV    #35,TSTNO
3669 020550 012737 020712 001216      MOV    #TST36,NEXT
3670
3671 020556 104412              MSTCLR              ;R1 CONTAINS BASE DMC11 ADDRESS
3672 020560 005061 000004      CLR    4(R1)        ;MASTER CLEAR DMC11
3673 020564 104414              ROMCLK              ;CLEAR PORT4
3674 020566 122117              122117             ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3675 020570 004737 033676      JSR   PC,CLR10     ;PUT LINE UNIT IN BITSTUFF MODE
3676 020574 012702 000012      MOV    #12,R2       ;DO THIS AFTER MODE IS SET
3677 020600 012711 004000      MOV    #BIT11,(R1)  ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3678 020604 012761 000001 000004      MOV    #1,4(R1)    ;SET LINE UNIT LOOP
3679 020612 104414              ROMCLK              ;SET BIT0 IN PORT4
3680 020614 122111              122111             ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3681 020616 104414              ROMCLK              ;SET SOM!
3682 020620 122110              122110             ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3683 020622 004737 032644      JSR   PC,CHARSD    ;LOAD GARBAGE CHAR
3684 020626 000125              125                ;LOAD SILO WITH CHARACTER
3685 020630 104415 000033      DATACLK,          ;CHARACTER
3686 020634 104416 000002      TIMER, 2           ;SINGLE CLOCK THE DATA
3687 020640 104414              ROMCLK              ;WAIT FOR INRDY
3688 020642 021244              021244             ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3689 020644 016104 000004      MOV    4(R1),R4     ;PORT4 LU 12
3690 020650 042704 000357      BIC   #357,R4       ;PUT "FOUND" IN R4
3691 020654 012705 000020      MOV    #BIT4,R5     ;CLEAR UNWANTED BITS
3692 020660 120504              CMPB  R5,R4         ;PUT "EXPECTED" IN R5
3693 020662 001401              BEQ   1$            ;IS INRDY SET?
3694 020664 104040              HLT   40            ;ERROR, INRDY IS NOT SET
3695 020666
3696 020666 104414              1$: ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3697 020670 021204              021204             ;PORT4 IN DATA
3698 020672 016104 000004      MOV    4(R1),R4     ;PUT "FOUND" IN R4
3699 020676 012705 000125      MOV    #125,R5      ;PUT "EXPECTED" IN R5
3700 020702 120504              CMPB  R5,R4         ;WAS A 125 RECEIVED?
3701 020704 001401              BEQ   2$            ;
3702 020706 104010              HLT   10            ;ERROR, RECEIVED DATA IS WRONG
3703 020710 104400              2$: SCOPE          ;SCOPE THIS TEST
3704
3705
3706
3707      ;***** TEST 36 *****
3708      ;*BITSTUFF BASIC RECEICER TEST

```



```

3708 ;*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 252
3709 ;*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
3710 ;*****
3711
3712 ; TEST 36
3713 ;-----
3714 020712 012737 000036 001226 TST36: MOV #36,TSTNO
3715 020720 012737 021062 001216 MOV #TST37,NEXT
3716 ;R1 CONTAINS BASE DMC11 ADDRESS
3717 020726 104412 MSTCLR ;MASTER CLEAR DMC11
3718 020730 005061 000004 CLR 4(R1) ;CLEAR PORT4
3719 020734 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3720 020736 122117 122117 ;PUT LINE UNIT IN BITSTUFF MODE
3721 020740 004737 033676 JSR PC,CLR10 ;DO THIS AFTER MODE IS SET
3722 020744 012702 000012 MOV #12,R2 ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3723 020750 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
3724 020754 012761 000001 000004 MOV #1,4(R1) ;SET BIT0 IN PORT4
3725 020762 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3726 020764 122111 122111 ;SET SOM!
3727 020766 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3728 020770 122110 122110 ;LOAD GARBAGE CHAR
3729 020772 004737 032644 JSR PC,CHARSD ;LOAD SILO WITH CHARACTER
3730 020776 000252 252 ;CHARACTER
3731 021000 104415 000033 DATACLK, 33 ;SINGLE CLOCK THE DATA
3732 021004 104416 000002 TIMER, 2 ;WAIT FOR INRDY
3733 021010 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3734 021012 021244 021244 ;PORT4 LU 12
3735 021014 016104 000004 MOV 4(R1),R4 ;PUT "FOUND" IN R4
3736 021020 042704 000357 BIC #357,R4 ;CLEAR UNWANTED BITS
3737 021024 012705 000020 MOV #BIT4,R5 ;PUT "EXPECTED" IN R5
3738 021030 120504 CMPB R5,R4 ;IS INRDY SET?
3739 021032 001401 BEQ 1$
3740 021034 104040 HLT 40 ;ERROR, INRDY IS NOT SET
3741 021036 104414 1$: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3742 021036 104414 021204 ;PORT4 IN DATA
3743 021040 021204 021204 ;PUT "FOUND" IN R4
3744 021042 016104 000004 MOV 4(R1),R4 ;PUT "EXPECTED" IN R5
3745 021046 012705 000252 MOV #252,R5 ;WAS A 252 RECEIVED?
3746 021052 120504 CMPB R5,R4
3747 021054 001401 BEQ 2$
3748 021056 104010 HLT 10 ;ERROR, RECEIVED DATA IS WRONG
3749 021060 104400 2$: SCOPE ;SCOPE THIS TEST
3750
3751
3752 ;***** TEST 37 *****
3753 ;*BITSTUFF BASIC RECEICER TEST
3754 ;*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 377
3755 ;*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
3756 ;*****
3757
3758 ; TEST 37
3759 ;-----
3760 021062 012737 000037 001226 TST37: MOV #37,TSTNO
3761 021070 012737 021232 001216 MOV #TST40,NEXT
3762
3763 021076 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11

```

```

3764 021100 005061 000004 CLR 4(R1) ;CLEAR PORT4
3765 021104 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3766 021106 122117 122117 ;PUT LINE UNIT IN BITSTUFF MODE
3767 021110 004737 033676 JSR PC,CLRIO ;DO THIS AFTER MODE IS SET
3768 021114 012702 000012 MOV #12,R2 ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3769 021120 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
3770 021124 012761 000001 000004 MOV #1,4(R1) ;SET BIT0 IN PORT4
3771 021132 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3772 021134 122111 122111 ;SET SOM!
3773 021136 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3774 021140 122110 122110 ;LOAD GARBAGE CHAR
3775 021142 004737 032644 JSR PC,CHARSD ;LOAD SILO WITH CHARACTER
3776 021146 000377 377 ;CHARACTER
3777 021150 104415 000034 DATACLK, 34 ;SINGLE CLOCK THE DATA
3778 021154 104416 000002 TIMER, 2 ;WAIT FOR INRDY
3779 021160 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3780 021162 021244 021244 ;PORT4, LU 12
3781 021164 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
3782 021170 042704 000357 BIC #357,R4 ;CLEAR UNWANTED BITS
3783 021174 012705 000020 MOV #BIT4,R5 ;PUT 'EXPECTED' IN R5
3784 021200 120504 CMPB R5,R4 ;IS INRDY SET?
3785 021202 001401 BEQ 1$
3786 021204 104040 HLT 40 ;ERROR, INRDY IS NOT SET
3787 021206 1$:
3788 021206 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3789 021210 021204 021204 ;PORT4 IN DATA
3790 021212 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
3791 021216 012705 000377 MOV #377,R5 ;PUT 'EXPECTED' IN R5
3792 021222 120504 CMPB R5,R4 ;WAS A 377 RECEIVED?
3793 021224 001401 BEQ 2$
3794 021226 104010 HLT 10 ;ERROR, RECEIVED DATA IS WRONG
3795 021230 104400 2$: SCOPE ;SCOPE THIS TEST
3796
3797
3798 ;***** TEST 40 *****
3799 ;*BITSTUFF DATA TEST
3800 ;*THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
3801 ;*CHECKING EACH CHARACTER AS IT IS RECEIVED
3802 ;:*****
3803
3804 ; TEST 40
3805 ;-----
3806 021232 012737 000040 001226 TST40: MOV #40,TSTNO
3807 021240 012737 021406 001216 MOV #TST41,NEXT
3808
3809 021246 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3810 021250 005061 000004 CLR 4(R1) ;MASTER CLEAR DMC11
3811 021254 104414 ROMCLK ;CLEAR PORT4
3812 021256 122117 122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3813 021260 004737 033676 JSR PC,CLRIO ;PUT LINE UNIT IN BITSTUFF MODE
3814 021264 005037 033150 CLR SCHAR ;DO THIS AFTER MODE IS SET
3815 021270 005137 033150 COM SCHAR ;START BINARY COUNT AT ZERO
3816 021274 005037 034114 CLR BITCON ;IF BITSTUFF SCHAR IS MINUS NUMBER
3817 021300 005037 033152 CLR STUFLG ;START 1'S COUNT AT 0
3818 021304 005002 CLR R2 ;CLEAR BITSTUFF FLAG
3819 021306 012703 000073 MOV #73,R3 ;R2 IS 'EXPECTED' DATA
;R3 IS CHARACTER COUNT

```



```

3820 021312 012711 004000      MOV    #BIT11,(R1)      ;SET LINE UNIT LOOP
3821 021316 004737 032710      JSR    PC,SILOLD       ;LOAD SILO WITH COUNT PATTERN
3822 021322 104415 000023      DATACLK, 23          ;SYNC RECEIVER AND GET IT ACTIVE
3823 021326 104415 000730      1$:  DATACLK, 730     ;CLOCK IN 73 CHARACTERS
3824 021332 004737 033154      4$:  JSR    PC,INRDY    ;WAIT FOR INRDY
3825 021336 104414              ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3826 021340 021204              021204              ;PORT4 IN DATA
3827 021342 016104 000004      MOV    4(R1),R4        ;PUT 'FOUND' IN R4
3828 021346 010205              MOV    R2,R5           ;PUT 'EXPECTED' IN R5
3829 021350 120504              CMPB   R5,R4           ;IS DATA CORRECT?
3830 021352 001401              BEQ    2$              ;BR IF YES
3831 021354 104010              HLT    10              ;DATA ERROR
3832 021356 005202              2$:  INC    R2          ;NEXT CHARACTER
3833 021360 022702 000400      CMP    #400,R2        ;ALL DONE?
3834 021364 001407              BEQ    3$              ;BR IF YES
3835 021366 005303              DEC    R3              ;DECREMENT CHARACTER COUNT
3836 021370 001360              BNE    4$              ;BR IF SILO NOT EMPTY
3837 021372 004737 032710      JSR    PC,SILOLD       ;LOAD SILO WITH MORE OF COUNT PATTERN
3838 021376 012703 000073      MOV    #73,R3         ;RELOAD CHARACTER COUNT
3839 021402 000751              BR     1$              ;CONTINUE
3840 021404 104400              3$:  SCOPE            ;SCOPE THIS TEST
3841
3842
3843      ;***** TEST 41 *****
3844      ;*BITSTUFF DATA TEST
3845      ;*THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
3846      ;*CHECKING EACH CHARACTER AS IT IS RECEIVED
3847      ;*THIS TEST IS EXACTLY THE SAME AS THE LAST TEST,
3848      ;*EXCEPT LINE UNIT LOOP IS SET IN LU REGISTER 12
3849      ;*****
3850
3851      ; TEST 41
3852      ;-----
3853 021406 012737 000041 001226 TST41: MOV    #41,TSTNO
3854 021414 012737 021572 001216      MOV    #TST42,NEXT
3855
3856 021422 104412              MSTCLR                ;R1 CONTAINS BASE DMC11 ADDRESS
3857 021424 005061 000004      CLR    4(R1)          ;MASTER CLEAR DMC11
3858 021430 104414              ROMCLK                ;CLEAR PORT4
3859 021432 122117              122117              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3860 021434 004737 033676      JSR    PC,CLRIO       ;PUT LINE UNIT IN BITSTUFF MODE
3861 021440 005037 033150      CLR    SCHAR          ;DO THIS AFTER MODE IS SET
3862 021444 005137 033150      COM    SCHAR          ;START BINARY COUNT AT ZERO
3863 021450 005037 034114      CLR    BITCON         ;IF BITSTUFF SCHAR IS MINUS NUMBER
3864 021454 005037 033152      CLR    STUFLG        ;START 1'S COUNT AT 0
3865 021460 005002              CLR    R2              ;CLEAR BITSTUFF FLAG
3866 021462 012703 000073      MOV    #73,R3         ;R2 IS 'EXPECTED' DATA
3867 021466 005011              CLR    (R1)           ;R3 IS CHARACTER COUNT
3868 021470 012761 000040 000004      MOV    #BIT5,4(R1)   ;CLEAR LU LOOP IN MAINT REG
3869 021476 104414              ROMCLK                ;LOAD PORT4
3870 021500 122112              122112              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3871 021502 004737 032710      JSR    PC,SILOLD       ;SET LU LOOP IN LU REG 12
3872 021506 104415 000023      DATACLK, 23          ;LOAD SILO WITH COUNT PATTERN
3873 021512 104415 000730      1$:  DATACLK, 730     ;SYNC RECEIVER AND GET IT ACTIVE
3874 021516 004737 033154      4$:  JSR    PC,INRDY    ;CLOCK IN 73 CHARACTERS
3875 021522 104414              ROMCLK                ;WAIT FOR INRDY
                                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

```

```

3876 021524 021204 021204 :PORT4 IN DATA
3877 021526 016104 000004 MOV 4(R1),R4 :PUT 'FOUND' IN R4
3878 021532 010205 MOV R2,R5 :PUT 'EXPECTED' IN R5
3879 021534 120504 CMPB R5,R4 :IS DATA CORRECT?
3880 021536 001401 BEQ 2$ :BR IF YES
3881 021540 104010 HLT 10 :DATA ERROR
3882 021542 005202 2$: INC R2 :NEXT CHARACTER
3883 021544 022702 000400 CMP #400,R2 :ALL DONE?
3884 021550 001407 BEQ 3$ :BR IF YES
3885 021552 005303 DEC R3 :DECREMENT CHARACTER COUNT
3886 021554 001360 BNE 4$ :BR IF SILO NOT EMPTY
3887 021556 004737 032710 JSR PC,SILOLD :LOAD SILO WITH MORE OF COUNT PATTERN
3888 021562 012703 000073 MOV #73,R3 :RELOAD CHARACTER COUNT
3889 021566 000751 BR 1$ :CONTINUE
3890 021570 104400 3$: SCOPE :SCOPE THIS TEST
3891
3892
3893 :***** TEST 42 *****
3894 :*RECEIVER ABORT TEST
3895 :*SINGLE CLOCK 3 FLAGS, A 301, ANOTHER 301 AND 10 EXTRA
3896 :*CLOCK TICKS, VERIFY THAT A 301 AND A BLOCK END
3897 :*WERE RECEIVED INDICATING THAT THE RECEIVER RECOGNIZED
3898 :*THE ABORT SEQUENCE (8 CONTIGUIOUS 1'S)
3899 :*****
3900
3901 : TEST 42
3902 :-----
3903 021572 012737 000042 001226 TST42: MOV #42,TSTNO
3904 021600 012737 021734 001216 MOV #TST43,NEXT
3905 :R1 CONTAINS BASE DMC11 ADDRESS
3906 021606 104412 MSTCLR :MASTER CLEAR DMC11
3907 021610 005061 000004 CLR 4(R1)
3908 021614 104414 ROMCLK :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3909 021616 122117 122117 :PUT LINE UNIT IN BITSTUFF MODE
3910 021620 004737 033676 JSR PC,CLRIO :DO THIS AFTER MODE IS SET
3911 021624 012711 004000 MOV #BIT11,(R1) :SET LINE UNIT LOOP
3912 021630 004737 032532 JSR PC,CHAR :LOAD SILO WITH 3 FLAGS
3913 021634 000301 301 :AND A 301
3914 021636 004737 032500 JSR PC,OUTRDY :WAIT FOR OUTRDY
3915 021642 104414 ROMCLK :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3916 021644 122110 122110 :LOAD 2ND 301 CHARACTER
3917 021646 104415 000073 DATACLK, 73 :CLOCK THE 301 IN AND 10 EXTRA TICKS
3918 021652 004737 033154 JSR PC,INRDY :WAIT FOR INRDY
3919 021656 104414 ROMCLK :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3920 021660 021204 021204 :PORT4 IN DATA
3921 021662 016104 000004 MOV 4(R1),R4 :PUT 'FOUND' IN R4
3922 021666 012705 000301 MOV #301,R5 :PUT 'EXPECTED' IN R5
3923 021672 120504 CMPB R5,R4 :WAS A 301 RECEIVED?
3924 021674 001401 BEQ 1$
3925 021676 104010 HLT 10 :ERROR FIRST CHARACTER INCORRECT
3926 021700 004737 033154 1$: JSR PC,INRDY :WAIT FOR INRDY
3927 021704 104414 ROMCLK :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3928 021706 021244 021244 :READ LU-12
3929 021710 016104 000004 MOV 4(R1),R4 :PUT 'FOUND' IN R4
3930 021714 042704 000375 BIC #375,R4 :CLEAR UNWANTED BITS
3931 021720 012705 000002 MOV #2,R5 :PUT 'EXPECTED' IN R5

```



```

3932 021724 120504          CMPB   R5,R4          ; IS BLOCK END SET?
3933 021726 001401          BEQ    3$            ; BR IF YES
3934 021730 104032          HLT    32            ; ERROR, BLOCK END NOT SET
3935 021732 104400          3$:   SCOPE         ; SCOPE THIS TEST
3936
3937
3938                          ;***** TEST 43 *****
3939                          ;*CABLE TURNAROUND TEST
3940                          ;*CLEAR LINE UNIT LOOP, SET DTR
3941                          ;*VERIFY THAT RING AND MODEM READY ARE SET
3942                          ;*CLEAR DTR, VERIFY THAT MRDY IS CLEARED
3943                          ;*****
3944
3945                          ; TEST 43
3946                          ;-----
3947 021734 012737 000043 001226 TST43: MOV    #43,TSTNO
3948 021742 012737 022162 001216      MOV    #TST44,NEXT
3949
3950 021750 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
3951 021752 032737 020000 001366      BIT    #BIT13,STAT1 ; IS LINE UNIT M8202?
3952 021760 001004          BNE    .+12      ; BR IF YES (DO TEST EVEN IF NO LOOP-BACK CONN)
3953 021762 032737 040000 001366      BIT    #BIT14,STAT1 ; IS TURNAROUND CONNECTOR ON?
3954 021770 001473          BEQ    2$        ; SKIP TEST IF NO
3955 021772 005011          CLR    (R1)     ; CLEAR LINE UNIT LOOP
3956 021774 012761 000100 000004      MOV    #100,4(R1) ; LOAD PORT4
3957 022002 104414          ROMCLK        ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3958 022004 122113          122113        ; SET DTR
3959 022006 104416 000002          TIMER, 2     ; WAIT
3960 022012 104414          ROMCLK        ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3961 022014 021264          021264        ; PORT4_LU13
3962 022016 016104 000004      MOV    4(R1),R4  ; PUT FOUND IN R4      ****NEW
3963 022022 042704 000023      BIC    #23,R4   ; CLEAR JUNK
3964 022026 012705 000310      MOV    #310,R5  ; GET EXPECTED.
3965 022032 032737 020000 001366      BIT    #BIT13,STAT1 ; IS LINE UNIT M8202?
3966 022040 001402          BEQ    .+6      ;
3967 022042 042705 000200          BIC    #BIT7,R5  ; NO RING ON M8202
3968
3969 022046 032737 000004 001372      BIT    #BIT2,STAT3 ; IS THIS V.35 MODEM?
3970 022054 001402          BEQ    3$      ; NO THEN GO AHEAD.
3971
3972 022056 042705 000200          3$:   BIC    #BIT7,R5  ; YES-NO RING ON V.35 MODEM
3973 022062
3974 022062 020504          CMPB   R5,R4   ; ARE RING AND MODEM READY SET?
3975 022064 001401          BEQ    1$      ; WARNING! IF V.35 AND AUTO STARTED,
3976                          ; YOU WILL GET THIS ERROR. YOU MUST
3977                          ; MANUALL NASWER THE QUESTIONS FOR V.35.
3978 022066 104011          1$:   HLT    11   ; ERROR, MRDY NOT SET
3979 022070 005061 000004      CLR    4(R1)  ; CLEAR PORT4
3980 022074 104414          ROMCLK        ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3981 022076 122113          122113        ; CLEAR DTR
3982 022100 104416 000002          TIMER, 2     ;
3983 022104 104414          ROMCLK        ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3984 022106 021264          021264        ; PORT4_LU13
3985 022110 016104 000004      MOV    4(R1),R4  ; PUR FOUND IN R4      ****NEW
3986 022114 042704 000023      BIC    #23,R4   ; STRIP JUNK
3987 022120 005005          CLR    R5      ; SET EXPECTED.

```



```

3988 022122 032737 020000 001366      BIT    #BIT13,STAT1    ; IS THIS A M8202?
3989 022130 001402                      BEQ    .+6
3990 022132 052705 000010              BIS    #BIT3,R5       ; YES THEN EXPECT MRDY SET.
3991 022136 032737 000004 001372      BIT    #BIT2,STAT3    ; IS THIS V.35?
3992 022144 001402                      BEQ    4$
3993 022146 042704 000200              BIC    #BIT7,R4
3994 022152                                4$:
3995 022152 120405                      CMPB   R4,R5          ; ARE RING AND MRDY CLEAR?
3996 022154 001401                      BEQ    2$
3997
3998
3999
4000 022156 104011                      HLT    11             ; WARNING! YOU MAY GET THIS ERROR IF V.35
4001 022160 104400                      SCOPE                ; AND AUTOSTART. YOU MUST MANNUALLY ANSWER
4002
4003
4004
4005
4006
4007
4008
4009
4010
4011
4012 022162 012737 000044 001226      TST44: MOV    #44,TSTNO
4013 022170 012737 022356 001216      MOV    #TST45,NEXT
4014
4015 022176 104412                      MSTCLR                ;R1 CONTAINS BASE DMC11 ADDRESS
4016 022200 032737 020000 001366      BIT    #BIT13,STAT1    ;MASTER CLEAR DMC11
4017 022206 001004                      BNE    .+12           ;IS LINE UNIT M8202?
4018 022210 032737 040000 001366      BIT    #BIT14,STAT1    ;BR IF YES (DO TEST EVEN IF NO LOOP-BACK CONN)
4019 022216 001456                      BEQ    1$             ;IS TURNAROUND CONNECTOR ON?
4020 022220 012711 004000                      1$:
4021 022224 012761 000100 000004      MOV    #BIT11,(R1)    ;SKIP TEST IF NO
4022 022232 104414                      MOV    #100,4(R1)     ;SET LINE UNIT LOOP
4023 022234 122113                      ROMCLK                ;LOAD PORT4
4024 022236 104416 000002                      122113                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4025 022242 012761 000001 000004      TIMER, 2              ;CLEAR ALL MODEM SIGNALS,EXCEPT DTR
4026 022250 104414                      MOV    #1,4(R1)      ;WAIT
4027 022252 122111                      ROMCLK                ;LOAD PORT4
4028 022254 004537 033634                      122111                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4029 022260 034116                      JSR    R5,MESLD       ;SET SOM
4030 022262 000100                      MESDAT                ;FILL OUT DATA SILO
4031 022264 012700 000050                      64.                   ;WITH 64 CHARACTERS
4032 022270 005011                      MOV    #50,R0         ;PREPARE FOR DELAY
4033 022272                      CLR    (R1)           ;CLEAR LINE UNIT LOOP
4034 022272 104414                      ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4035 022274 021264                      021264                ;PORT4 LU13
4036 022276 016104 000004                      MOV    4(R1),R4       ;PUT 'FOUND' IN R4
4037 022302 042704 000023                      BIC    #23,R4         ;CLEAR UNWANTED BITS
4038 022306 012705 000354                      MOV    #354,R5        ;PUT 'EXPECTED' IN R5
4039
4040 022312 032737 000004 001372      BIT    #BIT2,STAT3    ; SEE IF V.35
4041 022320 001402                      BEQ    40$
4042 022322 042705 000200              BIC    #BIT7,R5       ; IF SO, NO RING.
4043 022326 032737 020000 001366      40$: BIT    #BIT13,STAT1    ; IS THIS M8202?

```



```

4044 022334 001402          BEQ      4$          ; NO,SKIP
4045 022336 042705 000200          BIC      #BIT7,R5
4046 022342          4$:          CMPB    R5,R4          ;COMPARE EXPECTED AND FOUND
4047 022342 120504          BEQ      1$          ;BR IF OK
4048 022344 001403          DEC      R0          ;DEC DELAY COUNT
4049 022346 005300          BNE     2$          ;BR IF NOT ZERO
4050 022350 001350          HLT     11          ;ERROR, ALL SIGNALS ARE NOT SET
4051 022352 104011          ; YOU MUST ANSWER ALL QUESTIONS
4052          ; MAUNALLY IF V.35 .
4053          ; SCOPE THIS TEST
4054 022354 104400          1$:     SCOPE
4055
4056
4057          ;***** TEST 45 *****
4058          ;*TEST OF CRC OPERATION
4059
4060          ;*USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK THE CHARACTER
4061          ;*0, VERIFY THE LSB OF THE BCC ON EACH SHIFT
4062
4063          ;*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
4064          ;:*****
4065
4066          ; TEST 45
4067          ;-----
4068 022356 012737 000045 001226 TST45: MOV     #45,TSTNO
4069 022364 012737 022722 001216 MOV     #TST46,NEXT
4070 022372 012737 022426 001220 MOV     #64$,LOCK
4071
4072 022400 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
4073 022402 005061 000004          CLR      4(R1)  ;MASTER CLEAR DMC11
4074 022406 104414          ROMCLK          ;CLEAR PORT4
4075 022410 122117          122117        ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4076          ;PUT LINE UNIT IN BITSTUFF MODE
4077 022412 004737 033676          JSR     PC,CLRIO  ;DO THIS AFTER MODE IS SET
4078 022416 005037 034114          CLR     BITCON   ;CONSECUTIVE 1'S COUNTER INIT TO 0
4079 022422 012711 004000          MOV     #BIT11,(R1) ;SET LU LOOP
4080 022426 004737 033676          64$:     JSR     PC,CLRIO  ;CLEAR BCC REGISTERS
4081 022432 005000          CLR     R0       ;START SHIFT COUNTER AT ZERO
4082 022434 012737 102010 033332 MOV     #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE BCC
4083 022442 012737 000000 022506 MOV     #0,66$   ;LOAD CHAR FOR SOFTWARE BCC
4084 022450 005037 022510          CLR     67$     ;CLEAR OLD SOFTWARE BCC
4085 022454 005137 022510          COM     67$     ;START AT -1
4086 022460 004737 033336          JSR     PC,BCCLD ;LOAD OUT SILO WITH 2 SYNCs
4087 022464 000000          0          ;AND THE CHARACTER 0
4088 022466 104415 000021          DATACLK, 21   ;GET TRANSMITTER ACTIVE
4089 022472 104415 000001          65$:     DATACLK, 1 ;SHIFT BCC ONCE
4090
4091 022476 005200          INC     R0       ;BUMP SHIFT COUNT
4092
4093 022500 004537 033210          JSR     R5,SIMBCC ;CALCULATE SOFTWARE BCC LSB
4094 022504 000001          1          ;ONE SHIFT
4095 022506 000000          66$:     0       ;DATA CHARACTER
4096 022510 000000          67$:     0       ;OLD BCC
4097 022512 103405          BCS     68$     ;BR IF SOFT BCC LSB IS SET
4098 022514 004737 033450          JSR     PC,GET00 ;GET HARDWARE TRANSMITTER BCC LSB
4099 022520 103006          BCC     69$     ;BR IF HARD BCC LSB IS CLEAR

```



```

4100 022522 104012
4101 022524 000404
4102 022526 004737 033450
4103 022532 103401
4104 022534 104016
4105 022536
4106 022536 006037 022506
4107 022542 013737 033334 022510
4108 022550 022700 000010
4109 022554 001346
4110 022556 104401
4111 022560 012737 022566 001220
4112 022566 004737 033676
4113 022572 005000
4114 022574 012737 102010 033332
4115 022602 012737 000000 022646
4116 022610 005037 022650
4117 022614 005137 022650
4118 022620 004737 033336
4119 022624 000000
4120 022626 104415 000032
4121 022632 104415 000001
4122
4123 022636 005200
4124
4125 022640 004537 033210
4126 022644 000001
4127 022646 000000
4128 022650 000000
4129 022652 103405
4130 022654 004737 033462
4131 022660 103006
4132 022662 104013
4133 022664 000404
4134 022666 004737 033462
4135 022672 103401
4136 022674 104017
4137 022676
4138 022676 006037 022646
4139 022702 013737 033334 022650
4140 022710 022700 000010
4141 022714 001346
4142 022716 104401
4143 022720 104400
4144
4145
4146
4147
4148
4149
4150
4151
4152
4153
4154
4155

```

```

          HLT 12 ;ERROR, BCC LSB IS SET
          BR 69$ ;CONTINUE
68$:     JSR PC,GETQO ;GET HARDWARE TRANSMITTER BCC LSB
          BCS 69$ ;BR IF HARD BCC LSB IS SET
          HLT 16 ;ERROR, HARD BCC LSB IS CLEAR
69$:     ROR 66$ ;SHIFT SOFT DATA
          MOV CALBCC,67$ ;LOAD OLD SOFT BCC
          CMP #10,RO ;DONE YET?
          BNE 65$ ;BR IF NOT DONE
          SCOP1 ;SCOPE SUBTEST (SW09=1)
71$:     MOV #71$,LOCK ;NEW SCOPE1
          JSR PC,CLRIO ;CLEAR BCC REGISTERS
          CLR RO ;START SHIFT COUNTER AT ZERO
          MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE BCC
          MOV #0,73$ ;LOAD CHAR FOR SOFTWARE BCC
          CLR 74$ ;CLEAR OLD SOFTWARE BCC
          COM 74$ ;START AT -1
          JSR PC,BCCLD ;LOAD OUT SILO WITH 2 SYNCs
          0 ;AND THE CHARACTER 0
72$:     DATACLK, 32 ;GET RECEIVER ACTIVE
          DATACLK, 1 ;SHIFT BCC ONCE
          INC RO ;BUMP SHIFT COUNT
          JSR R5,SIMBCC ;CALCULATE SOFTWARE BCC LSB
          1 ;ONE SHIFT
73$:     0 ;DATA CHARACTER
74$:     0 ;OLD BCC
          BCS 75$ ;BR IF SOFT BCC LSB IS SET
          JSR PC,GETQI ;GET HARDWARE RECEIVER BCC LSB
          BCC 76$ ;BR IF HARD BCC LSB IS CLEAR
          HLT 13 ;ERROR, BCC LSB IS SET
          BR 76$ ;CONTINUE
75$:     JSR PC,GETQI ;GET HARDWARE RECEIVER BCC LSB
          BCS 76$ ;BR IF HARD BCC LSB IS SET
          HLT 17 ;ERROR, BCC LSB IS CLEAR
76$:     ROR 73$ ;SHIFT SOFT DATA
          MOV CALBCC,74$ ;LOAD OLD SOFT BCC
          CMP #10,RO ;DONE YET?
          BNE 72$ ;BR IF NOT DONE
          SCOP1 ;SCOPE SUBTEST (SW09=1)
77$:     SCOPE ;SCOPE THIS TEST

```

```

;***** TEST 46 *****
; *TEST OF CRC OPERATION
; *USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK THE CHARACTER
; *377, VERIFY THE LSB OF THE BCC ON EACH SHIFT
; *TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
;*****
; TEST 46

```



```

4156
4157 022722 012737 000046 001226 TST46: MOV #46,TSTNO
4158 022730 012737 023314 0C1216 MOV #TST47,NEXT
4159 022736 012737 022772 001220 MOV #64$,LOCK
4160
4161 022744 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
4162 022746 005061 000004 CLR 4(R1) ;MASTER CLEAR DMC11
4163 022752 104414 ROMCLK ;CLEAR PORT4
4164 022754 122117 122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4165 ;PUT LINE UNIT IN BITSTUFF MODE
4166 022756 004737 033676 JSR PC,CLRIO ;DO THIS AFTER MODE IS SET
4167 022762 005037 034114 CLR BITCON ;CONSECUTIVE 1'S COUNTER INIT TO 0
4168 022766 012711 004000 MOV #BIT11,(R1) ;SET LU LOOP
4169 022772 004737 033676 64$: JSR PC,CLRIO ;CLEAR BCC REGISTERS
4170 022776 005000 CLR RO ;START SHIFT COUNTER AT ZERO
4171 023000 012737 102010 033332 MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE BCC
4172 023006 012737 000377 023072 MOV #377,66$ ;LOAD CHAR FOR SOFTWARE BCC
4173 023014 005037 023074 CLR 67$ ;CLEAR OLD SOFTWARE BCC
4174 023020 005137 023074 COM 67$ ;START AT -1
4175 023024 004737 033336 JSR PC,BCCLD ;LOAD OUT SILO WITH 2 SYNC'S
4176 023030 000377 377 ;AND THE CHARACTER 377
4177 023032 104415 000021 DATACLK, 21 ;GET TRANSMITTER ACTIVE
4178 023036 005037 034114 CLR BITCON ;CLEAR BIT COUNTER
4179 023042 005037 023056 CLR 60$
4180 023046 104415 000001 65$: DATACLK, 1 ;SHIFT BCC ONCE
4181 023052 004537 033776 JSR R5,STFFCK ;CHECK FOR STUFFING ZEROS
4182 023056 000000 60$: 0 ;CHARACTER
4183 023060 000001 1 ;SHIFT COUNT
4184
4185 023062 005200 INC RO ;BUMP SHIFT COUNT
4186
4187 023064 004537 033210 JSR R5,SIMBCC ;CALCULATE SOFTWARE BCC LSB
4188 023070 000001 1 ;ONE SHIFT
4189 023072 000000 66$: 0 ;DATA CHARACTER
4190 023074 000000 67$: 0 ;OLD BCC
4191 023076 103405 BCS 68$ ;BR IF SOFT BCC LSB IS SET
4192 023100 004737 033450 JSR PC,GETQO ;GET HARDWARE TRANSMITTER BCC LSB
4193 023104 103006 BCC 69$ ;BR IF HARD BCC LSB IS CLEAR
4194 023106 104012 HLT 12 ;ERROR, BCC LSB IS SET
4195 023110 000404 BR 69$ ;CONTINUE
4196 023112 004737 033450 68$: JSR PC,GETQO ;GET HARDWARE TRANSMITTER BCC LSB
4197 023116 103401 BCS 69$ ;BR IF HARD BCC LSB IS SET
4198 023120 104016 HLT 16 ;ERROR, HARD BCC LSB IS CLEAR
4199 69$:
4200 023122 013737 023072 023056 MOV 66$,60$
4201 023130 006037 023072 ROR 66$ ;SHIFT SOFT DATA
4202 023134 013737 033334 023074 MOV CALBCC,67$ ;LOAD OLD SOFT BCC
4203 023142 022700 000010 CMP #10,RO ;DONE YET?
4204 023146 001337 BNE 65$ ;BR IF NOT DONE
4205 023150 104401 SCOPE1 ;SCOPE SUBTEST (SW09=1)
4206 023152 012737 023160 001220 MOV #71$,LOCK ;NEW SCOPE1
4207 023160 004737 033676 71$: JSR PC,CLRIO ;CLEAR BCC REGISTERS
4208 023164 005000 CLR RO ;START SHIFT COUNTER AT ZERO
4209 023166 012737 102010 033332 MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE BCC
4210 023174 012737 000377 023240 MOV #377,73$ ;LOAD CHAR FOR SOFTWARE BCC
4211 023202 005037 023242 CLR 74$ ;CLEAR OLD SOFTWARE BCC

```



```

4212 023206 005137 023242          COM      74$          ;START AT -1
4213 023212 004737 033336          JSR      PC,BCCLD      ;LOAD OUT SILO WITH 2 SYNCs
4214 023216 000377                    377          ;AND THE CHARACTER 377
4215 023220 104415 000033          DATACLK, 33        ;GET RECEIVER ACTIVE
4216 023224 104415 000001          DATACLK, 1         ;SHIFT BCC ONCE
4217
4218 023230 005200                    INC      R0          ;BUMP SHIFT COUNT
4219
4220 023232 004537 033210          JSR      R5,SIMBCC    ;CALCULATE SOFTWARE BCC LSB
4221 023236 000001                    1           ;ONE SHIFT
4222 023240 000000          73$: 0           ;DATA CHARACTER
4223 023242 000000          74$: 0           ;OLD BCC
4224 023244 103405                    BCS      75$        ;BR IF SOFT BCC LSB IS SET
4225 023246 004737 033462          JSR      PC,GETQI    ;GET HARDWARE RECEIVER BCC LSB
4226 023252 103006                    BCC      76$        ;BR IF HARD BCC LSB IS CLEAR
4227 023254 104013                    HLT      13         ;ERROR, BCC LSB IS SET
4228 023256 000404                    BR       76$        ;CONTINUE
4229 023260 004737 033462          75$: JSR      PC,GETQI ;GET HARDWARE RECEIVER BCC LSB
4230 023264 103401                    BCS      76$        ;BR IF HARD BCC LSB IS SET
4231 023266 104017                    HLT      17         ;ERROR, BCC LSB IS CLEAR
4232
4233 023270 006037 023240          76$: ROR      73$    ;SHIFT SOFT DATA
4234 023274 013737 033334 023242    MOV      CALBCC,74$  ;LOAD OLD SOFT BCC
4235 023302 022700 000010          CMP      #10,R0     ;DONE YET?
4236 023306 001346                    BNE      72$        ;BR IF NOT DONE
4237 023310 104401                    SCOPE1          ;SCOPE SUBTEST (SW09=1)
4238 023312 104400          77$: SCOPE          ;SCOPE THIS TEST
4239
4240
4241 ;***** TEST 47 *****
4242 ;*TEST OF CRC OPERATION
4243
4244 ;*USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK THE CHARACTER
4245 ;*125, VERIFY THE LSB OF THE BCC ON EACH SHIFT
4246
4247 ;*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
4248 ;*****
4249
4250 ; TEST 47
4251 ;-----
4252 023314 012737 000047 001226 TST47: MOV      #47,TSTNO
4253 023322 012737 023660 001216    MOV      #TST50,NEXT
4254 023330 012737 023364 001220    MOV      #64$,LOCK
4255
4256 023336 104412                    MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
4257 023340 005061 000004          CLR      4(R1)    ;MASTER CLEAR DMC11
4258 023344 104414                    ROMCLK         ;CLEAR PORT4
4259 023346 122117                    122117        ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4260 ;PUT LINE UNIT IN BITSTUFF MODE
4261 023350 004737 033676          JSR      PC,CLRIO   ;DO THIS AFTER MODE IS SET
4262 023354 005037 034114          CLR      BITCON    ;CONSECUTIVE 1'S COUNTER INIT TO 0
4263 023360 012711 004000          MOV      #BIT11,(R1) ;SET LU LOOP
4264 023364 004737 033676          64$: JSR      PC,CLRIO ;CLEAR BCC REGISTERS
4265 023370 005000                    CLR      R0        ;START SHIFT COUNTER AT ZERO
4266 023372 012737 102010 033332    MOV      #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE BCC
4267 023400 012737 000125 023444    MOV      #125,66$ ;LOAD CHAR FOR SOFTWARE BCC

```



0071

4268	023406	005037	023446		CLR	67\$		;CLEAR OLD SOFTWARE BCC
4269	023412	005137	023446		COM	67\$		;START AT -1
4270	023416	004737	033336		JSR	PC,BCCLD		;LOAD OUT SILO WITH 2 SYNCs
4271	023422	000125			125			;AND THE CHARACTER 125
4272	023424	104415	000021		DATACLK,	21		;GET TRANSMITTER ACTIVE
4273	023430	104415	000001	65\$:	DATACLK,	1		;SHIFT BCC ONCE
4274								
4275	023434	005200			INC	RO		;BUMP SHIFT COUNT
4276								
4277	023436	004537	033210		JSR	R5,SIMBCC		;CALCULATE SOFTWARE BCC LSB
4278	023442	000001			1			;ONE SHIFT
4279	023444	000000		66\$:	0			;DATA CHARACTER
4280	023446	000000		67\$:	0			;OLD BCC
4281	023450	103405			BCS	68\$		;BR IF SOFT BCC LSB IS SET
4282	023452	004737	033450		JSR	PC,GETQO		;GET HARDWARE TRANSMITTER BCC LSB
4283	023456	103006			BCC	69\$		;BR IF HARD BCC LSB IS CLEAR
4284	023460	104012			HLT	12		;ERROR, BCC LSB IS SET
4285	023462	000404			BR	69\$		;CONTINUE
4286	023464	004737	033450	68\$:	JSR	PC,GETQO		;GET HARDWARE TRANSMITTER BCC LSB
4287	023470	103401			BCS	69\$		;BR IF HARD BCC LSB IS SET
4288	023472	104016			HLT	16		;ERROR, HARD BCC LSB IS CLEAR
4289	023474			69\$:				
4290	023474	006037	023444		ROR	66\$		;SHIFT SOFT DATA
4291	023500	013737	033334	023446	MOV	CALBCC,67\$		;LOAD OLD SOFT BCC
4292	023506	022700	000010		CMP	#10,RO		;DONE YET?
4293	023512	001346			BNE	65\$		;BR IF NOT DONE
4294	023514	104401			SCOPE1			;SCOPE SUBTEST (SW09=1)
4295	023516	012737	023524	001220	MOV	#71\$,LOCK		;NEW SCOPE1
4296	023524	004737	033676		JSR	PC,CLRIO		;CLEAR BCC REGISTERS
4297	023530	005000			CLR	RO		;START SHIFT COUNTER AT ZERO
4298	023532	012737	102010	033332	MOV	#CRC.CCITT,XPOLY		;LOAD POLYNOMIAL FOR SOFTWARE BCC
4299	023540	012737	000125	023604	MOV	#125,73\$;		;LOAD CHAR FOR SOFTWARE BCC
4300	023546	005037	023606		CLR	74\$		;CLEAR OLD SOFTWARE BCC
4301	023552	005137	023606		COM	74\$		;START AT -1
4302	023556	004737	033336		JSR	PC,BCCLD		;LOAD OUT SILO WITH 2 SYNCs
4303	023562	000125			125			;AND THE CHARACTER 125
4304	023564	104415	000032		DATACLK,	32		;GET RECEIVER ACTIVE
4305	023570	104415	000001	72\$:	DATACLK,	1		;SHIFT BCC ONCE
4306								
4307	023574	005200			INC	RO		;BUMP SHIFT COUNT
4308								
4309	023576	004537	033210		JSR	R5,SIMBCC		;CALCULATE SOFTWARE BCC LSB
4310	023602	000001			1			;ONE SHIFT
4311	023604	000000		73\$:	0			;DATA CHARACTER
4312	023606	000000		74\$:	0			;OLD BCC
4313	023610	103405			BCS	75\$		;BR IF SOFT BCC LSB IS SET
4314	023612	004737	033462		JSR	PC,GETQI		;GET HARDWARE RECEIVER BCC LSB
4315	023616	103006			BCC	76\$		;BR IF HARD BCC LSB IS CLEAR
4316	023620	104013			HLT	13		;ERROR, BCC LSB IS SET
4317	023622	000404			BR	76\$		;CONTINUE
4318	023624	004737	033462	75\$:	JSR	PC,GETQI		;GET HARDWARE RECEIVER BCC LSB
4319	023630	103401			BCS	76\$		;BR IF HARD BCC LSB IS SET
4320	023632	104017			HLT	17		;ERROR, BCC LSB IS CLEAR
4321	023634			76\$:				
4322	023634	006037	023604		ROR	73\$		;SHIFT SOFT DATA
4323	023640	013737	033334	023606	MOV	CALBCC,74\$		;LOAD OLD SOFT BCC



```

4324 023646 022700 000010          CMP    #10,R0          ;DONE YET?
4325 023652 001346          BNE    72$            ;BR IF NOT DONE
4326 023654 104401          SCOPE1                ;SCOPE SUBTEST (SW09=1)
4327 023656 104400          77$: SCOPE           ;SCOPE THIS TEST
4328
4329
4330
4331
4332
4333
4334
4335
4336
4337
4338
4339
4340
4341 023660 012737 000050 001226 TST50: MOV    #50,TSTNO
4342 023666 012737 024224 001216      MOV    #TST51,NEXT
4343 023674 012737 023730 001220      MOV    #64$,LOCK
4344
4345 023702 104412          MSTCLR                ;R1 CONTAINS BASE DMC11 ADDRESS
4346 023704 005061 000004          CLR    4(R1)         ;MASTER CLEAR DMC11
4347 023710 104414          ROMCLK                ;CLEAR PORT4
4348 023712 122117          122117              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4349
4350 023714 004737 033676          JSR    PC,CLRIO      ;DO THIS AFTER MODE IS SET
4351 023720 005037 034114          CLR    BITCON        ;CONSECUTIVE 1'S COUNTER INIT TO 0
4352 023724 012711 004000          MOV    #BIT11,(R1)   ;SET LU LOOP
4353 023730 004737 033676          64$: JSR    PC,CLRIO  ;CLEAR BCC REGISTERS
4354 023734 005000          CLR    R0            ;START SHIFT COUNTER AT ZERO
4355 023736 012737 102010 033332      MOV    #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE BCC
4356 023744 012737 000252 024010      MOV    #252,66$;    ;LOAD CHAR FOR SOFTWARE BCC
4357 023752 005037 024012          CLR    67$          ;CLEAR OLD SOFTWARE BCC
4358 023756 005137 024012          COM    67$          ;START AT -1
4359 023762 004737 033336          JSR    PC,BCCLD     ;LOAD OUT SILO WITH 2 SYNC'S
4360 023766 000252          252                ;AND THE CHARACTER 252
4361 023770 104415 000021          DATACLK, 21        ;GET TRANSMITTER ACTIVE
4362 023774 104415 000001          65$: DATACLK, 1    ;SHIFT BCC ONCE
4363
4364 024000 005200          INC    R0            ;BUMP SHIFT COUNT
4365
4366 024002 004537 033210          JSR    R5,SIMBCC    ;CALCULATE SOFTWARE BCC LSB
4367 024006 000001          1                ;ONE SHIFT
4368 024010 000000          66$: 0            ;DATA CHARACTER
4369 024012 000000          67$: 0            ;OLD BCC
4370 024014 103405          BCS    68$          ;BR IF SOFT BCC LSB IS SET
4371 024016 004737 033450          JSR    PC,GETQO     ;GET HARDWARE TRANSMITTER BCC LSB
4372 024022 103006          BCC    69$          ;BR IF HARD BCC LSB IS CLEAR
4373 024024 104012          HLT    12          ;ERROR, BCC LSB IS SET
4374 024026 000404          BR     69$          ;CONTINUE
4375 024030 004737 033450          68$: JSR    PC,GETQO  ;GET HARDWARE TRANSMITTER BCC LSB
4376 024034 103401          BCS    69$          ;BR IF HARD BCC LSB IS SET
4377 024036 104016          HLT    16          ;ERROR, HARD BCC LSB IS CLEAR
4378 024040
4379 024040 006037 024010          69$: ROR    66$     ;SHIFT SOFT DATA

```



```

4380 024044 013737 033334 024012      MOV      CALBCC,67$      ;LOAD OLD SOFT BCC
4381 024052 022700 000010                CMP      #10,RO         ;DONE YET?
4382 024056 001346                BNE      65$            ;BR IF NOT DONE
4383 024060 104401                SCOPE1                ;SCOPE SUBTEST (SW09=1)
4384 024062 012737 024070 001220      MOV      #71$,LOCK     ;NEW SCOPE1
4385 024070 004737 033676      71$:    JSR      PC,CLRIO   ;CLEAR BCC REGISTERS
4386 024074 005000                CLR      RO            ;START SHIFT COUNTER AT ZERO
4387 024076 012737 102010 033332      MOV      #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE BCC
4388 024104 012737 000252 024150      MOV      #252,73$     ;LOAD CHAR FOR SOFTWARE BCC
4389 024112 005037 024152                CLR      74$          ;CLEAR OLD SOFTWARE BCC
4390 024116 005137 024152                COM      74$          ;START AT -1
4391 024122 004737 033336      JSR      PC,BCCLD     ;LOAD OUT SILO WITH 2 SYNCs
4392 024126 000252                252                ;AND THE CHARACTER 252
4393 024130 104415 000032      DATACLK, 32        ;GET RECEIVER ACTIVE
4394 024134 104415 000001      72$:    DATACLK, 1    ;SHIFT BCC ONCE
4395
4396 024140 005200                INC      RO            ;BUMP SHIFT COUNT
4397
4398 024142 004537 033210      JSR      R5,SIMBCC    ;CALCULATE SOFTWARE BCC LSB
4399 024146 000001                1                ;ONE SHIFT
4400 024150 000000      73$:    0                ;DATA CHARACTER
4401 024152 000000      74$:    0                ;OLD BCC
4402 024154 103405                BCS      75$          ;BR IF SOFT BCC LSB IS SET
4403 024156 004737 033462      JSR      PC,GETQI     ;GET HARDWARE RECEIVER BCC LSB
4404 024162 103006                BCC      76$          ;BR IF HARD BCC LSB IS CLEAR
4405 024164 104013                HLT      13           ;ERROR, BCC LSB IS SET
4406 024166 000404                BR       76$          ;CONTINUE
4407 024170 004737 033462      75$:    JSR      PC,GETQI ;GET HARDWARE RECEIVER BCC LSB
4408 024174 103401                BCS      76$          ;BR IF HARD BCC LSB IS SET
4409 024176 104017                HLT      17           ;ERROR, BCC LSB IS CLEAR
4410 024200      76$:
4411 024200 006037 024150      ROR      73$          ;SHIFT SOFT DATA
4412 024204 013737 033334 024152      MOV      CALBCC,74$   ;LOAD OLD SOFT BCC
4413 024212 022700 000010                CMP      #10,RO         ;DONE YET?
4414 024216 001346                BNE      72$          ;BR IF NOT DONE
4415 024220 104401                SCOPE1                ;SCOPE SUBTEST (SW09=1)
4416 024222 104400      77$:    SCOPE                ;SCOPE THIS TEST
4417
4418
4419
4420
4421
4422
4423
4424
4425
4426

```

```

;***** TEST 51 *****
; *TRANSMITTER CRC TEST
; *USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK A BINARY
; *COUNT PATTERN, VERIFY THE LSB OF THE TRANSMITTER BCC ON EACH SHIFT
;*****

```

```

; TEST 51
;-----
4427 024224 012737 000051 001226 TST51: MOV      #51,TSTNO
4428 024232 012737 024546 001216      MOV      #TST52,NEXT
4429
4430 024240 104412                MSTCLR                ;R1 CONTAINS BASE DMC11 ADDRESS
4431 024242 005061 000004                CLR      4(R1)        ;MASTER CLEAR DMC11
4432 024246 104414                ROMCLK                ;CLEAR PORT4
4433 024250 122117                122117                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4434 024252 004737 033676      JSR      PC,CLRIO     ;PUT LINE UNIT IN BITSTUFF MODE
4435 024256 005037 034114                CLR      BITCON        ;DO THIS AFTER MODE IS SET
;CONSECUTIVE 1'S COUNTER INIT TO 0

```

4436	024262	012711	004000		MOV	#BIT11,(R1)	:SET LINE UNIT LOOP	
4437	024266	005003			CLR	R3	:ZERO BIT COUNT	
4438	024270	005004			CLR	R4	:R4 CONTAINS CHAR TO BE LOADED IN SILO	
4439	024272	005005			CLR	R5	:R5 CONTAINS CHAR CURRENTLY BEING SHIFTED OUT	
4440	024274	005037	024422		CLR	4\$	:CLEAR SOFT BCC	
4441	024300	005137	024422		COM	4\$	:START AT -1	
4442	024304	012737	102010	033332	MOV	#CRC.CCITT,XPOLY	:LOAD POLYNOMIAL	
4443	024312	004737	033500		JSR	PC,SYNLD	:LOAD SILO WITH 2 SYNCs, SOM SET	
4444	024316	010461	000004		MOV	R4,4(R1)	:PORT4 CHAR	
4445	024322	104414		ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304	
4446	024324	122110			122110		:LOAD OUT DATA	
4447	024326	005204			INC	R4	:INCREMENT TO NEXT CHARACTER	
4448	024330	010461	000004		MOV	R4,4(R1)	:PORT4 CHAR	
4449	024334	104414		ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304	
4450	024336	122110			122110		:LOAD OUT DATA	
4451	024340	005204			INC	R4	:INCREMENT TO NEXT CHARACTER	
4452	024342	010461	000004		MOV	R4,4(R1)	:PORT4 CHAR	
4453	024346	104414		ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304	
4454	024350	122110			122110		:LOAD OUT DATA	
4455	024352	004737	032346		JSR	PC,OCOR	:WAIT FOR OCOR	
4456	024356	104415	000021		DATACLK,	21	:CLOCK DATA	
4457	024362	010537	024406		MOV	R5,10\$	:START WITH ZERO	
4458	024366	012700	000001		MOV	#1,R0	:START COUNT AT 1	
4459	024372	010537	024420		1\$:	MOV	R5,3\$	:LOAD CHAR FOR SOFT CRC
4460	024376	104415	000001		2\$:	DATACLK,	1	:SHIFT BCC ONCE
4461	024402	004537	033776		JSR	R5,STFFCK	:CHECK BIT STUFFING	
4462	024406	000000		10\$:	0		:CHARACTER	
4463	024410	000001			1		:SHIFT COUNT	
4464	024412	004537	033210		JSR	R5,SIMBCC	:CALCULATE SOFT BCC	
4465	024416	000001			1		:SOFT SHIFT COUNT	
4466	024420	000000		3\$:	0		:SOFT CHARACTER	
4467	024422	000000		4\$:	0		:OLD SOFT BCC	
4468	024424	103405			BCS	5\$	:BR IF SOFT BCC LSB IS SET	
4469	024426	004737	033450		JSR	PC,GETQO	:GET HARDWARE TRANSMITTER BCC LSB	
4470	024432	103006			BCC	6\$	:BR IF OK (CLEARED)	
4471	024434	104020			HLT	20	:ERROR, BCC LSB WAS SET	
4472	024436	000404			BR	6\$	:CONTINUE WITH TEST	
4473	024440	004737	033450		5\$:	JSR	PC,GETQO	:GET HARDWARE TRANSMITTER BCC LSB
4474	024444	103401			BCS	6\$	:BR IF OK (SET)	
4475	024446	104021			HLT	21	:ERROR, BCC LSB WAS CLEAR	
4476								
4477	024450			6\$:				
4478	024450	006037	024406		ROR	10\$	:SHIFT CHAR FOR STUFF CHECK	
4479	024454	005300			DEC	R0	:DEC STUFF CHECK SHIFT COUNT	
4480	024456	001004			BNE	11\$	:BR IF NOT DONE THIS CHARACTER	
4481	024460	012700	000010		MOV	#10,R0	:RESET BIT COUNT TO 10	
4482	024464	010537	024406		MOV	R5,10\$	:LOAD NEXT CHAR FOR STUFF CHECK	
4483	024470			11\$:				
4484	024470	006037	024420		ROR	3\$	:SHIFT SOFT DATA	
4485	024474	013737	033334	024422	MOV	CALBCC,4\$	:LOAD OLD SOFT BCC	
4486	024502	005203			INC	R3	:INCREMENT BIT COUNTER	
4487	024504	022703	000010		CMP	#10,R3	:DONE A FULL CHARACTER YET?	
4488	024510	001332			BNE	2\$	:BR IF NO	
4489	024512	005003			CLR	R3	:RESTART BIT COUNTER	
4490	024514	005204			INC	R4	:INCREMENT DATA FOR SILO	
4491	024516	022704	000400		CMP	#400,R4	:DONE BINARY COUNT YET?	



```

4492 024522 003404
4493 024524 010461 000004
4494 024530 104414
4495 024532 122110
4496 024534 005205
4497 024536 022705 000400
4498 024542 001313
4499 024544 104400
4500
4501
4502
4503
4504
4505
4506
4507
4508
4509
4510 024546 012737 000052 001226 TST52:
4511 024554 012737 025104 001216
4512
4513 024562 104412
4514 024564 005061 000004
4515 024570 104414
4516 024572 122117
4517 024574 004737 033676
4518 024600 005037 034114
4519 024604 012711 004000
4520 024610 005003
4521 024612 005004
4522 024614 005005
4523 024616 005037 024750
4524 024622 005137 024750
4525 024626 012737 102010 033332
4526 024634 004737 033500
4527 024640 010461 000004
4528 024644 104414
4529 024646 122110
4530 024650 005204
4531 024652 010461 000004
4532 024656 104414
4533 024660 122110
4534 024662 005204
4535 024664 010461 000004
4536 024670 104414
4537 024672 122110
4538 024674 004737 032346
4539 024700 104415 000032
4540 024704 010537 024734
4541 024710 005237 024734
4542 024714 012700 000010
4543 024720 010537 024746
4544 024724 104415 000001
4545 024730 004537 033776
4546 024734 000000
4547 024736 000001

BLE 9$ ;BR IF YES
MOV R4,4(R1) ;PORT4 DATA
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122110 ;LOAD OUT DATA
9$: INC R5 ;INCREMENT DATA
CMP #400,R5 ;DONE BINARY PATTERN YET?
BNE 1$ ;BR IF NO
7$: SCOPE ;SCOPE THIS TEST

;***** TEST 52 *****
; *RECEIVER CRC TEST
; *USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK A BINARY
; *COUNT PATTERN, VERIFY THE LSB OF THE RECEIVER BCC ON EACH SHIFT
;*****

: TEST 52
-----
MOV #52,TSTNO
MOV #TST53,NEXT
MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
CLR 4(R1) ;MASTER CLEAR DMC11
ROMCLK ;CLEAR PORT4
122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
JSR PC,CLRIO ;PUT LINE UNIT IN BITSTUFF MODE
CLR BITCON ;DO THIS AFTER MODE IS SET
MOV #BIT11,(R1) ;CONSECUTIVE 1'S COUNTER INIT TO 0
CLR R3 ;SET LINE UNIT LOOP
CLR R4 ;ZERO BIT COUNT
CLR R5 ;R4 CONTAINS CHAR TO BE LOADED IN SILO
CLR 4$ ;R5 CONTAINS CHAR CURRENTLY BEING SHIFTED OUT
COM 4$ ;CLEAR SOFT BCC
MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL
JSR PC,SYNLD ;LOAD SILO WITH 2 SYNCs, SOM SET
MOV R4,4(R1) ;PORT4 CHAR
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122110 ;LOAD OUT DATA
INC R4 ;INCREMENT TO NEXT CHARACTER
MOV R4,4(R1) ;PORT4 CHAR
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122110 ;LOAD OUT DATA
INC R4 ;INCREMENT TO NEXT CHARACTER
MOV R4,4(R1) ;PORT4 CHAR
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122110 ;LOAD OUT DATA
JSR PC,OCOR ;WAIT FOR OCOR
DATACLK,32 ;CLOCK DATA
MOV R5,10$ ;START WITH ZERO
INC 10$ ;TRANSMITTER IS ONE CHAR AHEAD
MOV #10,R0 ;R0 = CHAR COUNT
1$: MOV R5,3$ ;LOAD CHAR FOR SOFT CRC
2$: DATACLK,1 ;SHIFT BCC ONCE
JSR R5,STFFCK ;CHECK BIT STUFFING
10$: 0 ;CHARACTER
1 ;SHIFT COUNT

```

```

4548 024740 004537 033210 JSR R5,SIMBCC ;CALCULATE SOFT BCC
4549 024744 000001 1 ;SOFT SHIFT COUNT
4550 024746 000000 3$: 0 ;SOFT CHARACTER
4551 024750 000000 4$: 0 ;OLD SOFT BCC
4552 024752 103405 BCS 5$ ;BR IF SOFT BCC LSB IS SET
4553 024754 004737 033462 JSR PC,GETQI ;GET HARDWARE RECEIVER BCC LSB
4554 024760 103006 BCC 6$ ;BR IF OK (CLEARED)
4555 024762 104022 HLT 22 ;ERROR, BCC LSB WAS SET
4556 024764 000404 BR 6$ ;CONTINUE WITH TEST
4557 024766 004737 033462 5$: JSR PC,GETQI ;GET HARDWARE RECEIVER BCC LSB
4558 024772 103401 BCS 6$ ;BR IF OK (SET)
4559 024774 104023 HLT 23 ;ERROR, BCC LSB WAS CLEAR
4560
4561 024776 6$:
4562 024776 006037 024734 ROR 10$ ;SHIFT CHAR FOR STUFF CHECK
4563 025002 005300 DEC R0 ;DEC STUFF CHECK SHIFT COUNT
4564 025004 001010 BNE 11$ ;BR IF NOT DONE THIS CHARACTER
4565 025006 012700 000010 MOV #10,R0 ;RESET BIT COUNT TO 10
4566 025012 010537 024734 MOV R5,10$ ;LOAD NEXT CHAR FOR STUFF CHECK
4567 025016 005237 024734 INC 10$ ;TRANSMITTER IS 2 CHAR AHEAD
4568 025022 005237 024734 INC 10$ ;
4569 025026 11$:
4570 025026 006037 024746 ROR 3$ ;SHIFT SOFT DATA
4571 025032 013737 033334 024750 MOV CALBCC,4$ ;LOAD OLD SOFT BCC
4572 025040 005203 INC R3 ;INCREMENT BIT COUNTER
4573 025042 022703 000010 CMP #10,R3 ;DONE A FULL CHARACTER YET?
4574 025046 001326 BNE 2$ ;BR IF NO
4575 025050 005003 CLR R3 ;RESTART BIT COUNTER
4576 025052 005204 INC R4 ;INCREMENT DATA FOR SILO
4577 025054 022704 000400 CMP #400,R4 ;DONE BINARY COUNT YET?
4578 025060 003404 BLE 9$ ;BR IF YES
4579 025062 010461 000004 MOV R4,4(R1) ;PORT4 DATA
4580 025066 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4581 025070 122110 122110 ;LOAD OUT DATA
4582 025072 005205 9$: INC R5 ;INCREMENT DATA
4583 025074 022705 000400 CMP #400,R5 ;DONE BINARY PATTERN YET?
4584 025100 001307 BNE 1$ ;BR IF NO
4585 025102 104400 7$: SCOPE ;SCOPE THIS TEST
4586
4587
4588 ;***** TEST 53 *****
4589 ;*TRANSMITTER BITSTUFF CRC TEST
4590 ;*THIS TEST TRANSMITS A FOUR CHARACTER MESSAGE WITH CRC
4591 ;*BOTH DATA AND THE BCC ARE VERIFIED IN THE BIT
4592 ;*WINDOW. THE FOUR CHARACTERS ARE 0,125,252,377
4593 ;*THE TRANSMITTER IS CHECKED FOR GOING TO A MARK STATE AFTER THE BCC
4594 ;*****
4595
4596 ; TEST 53
4597 ;-----
4598 025104 012737 000053 001226 TST53: MOV #53,TSTNO
4599 025112 012737 025606 001216 MOV #TST54,NEXT
4600 ;R1 CONTAINS BASE DMC11 ADDRESS
4601 025120 104412 MSTCLR ;MASTER CLEAR DMC11
4602 025122 005061 000004 CLR 4(R1) ;CLEAR PORT4
4603 025126 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

```



```

4604 025130 122117 122117 ;PUT LINE UNIT IN BITSTUFF MODE
4605 025132 004737 033676 JSR PC,CLRIO ;DO THIS AFTER MODE IS SET
4606 025136 005037 034114 CLR BITCON ;CONSECUTIVE 1'S COUNTER INIT TO 0
4607
4608 ;LOAD OUT DATA SILO
4609
4610 025142 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
4611 025146 012704 034116 MOV #MESDAT,R4 ;LOAD POINTER TO DATA
4612 025152 005037 025262 CLR 10$ ;CLEAR SOFT BCC
4613 025156 005137 025262 COM 10$ ;START AT -1
4614 025162 012700 000004 MOV #4,R0 ;LOAD CHARACTER COUNT
4615 025166 004737 033500 JSR PC,SYNLD ;LOAD 2 FLAG CHARACTERS IN OUT SILO
4616 025172 004737 032500 JSR PC,OUTRDY ;WAIT FOR OUTRDY
4617 025176 004537 033634 JSR R5,MESLD ;LOAD SILO WITH 4 CHAR MESS
4618 025202 034116 MESDAT ;ADDRESS OF MESSAGE
4619 025204 000004 4 ;NUMBER OF CHARACTERS
4620 025206 004737 033610 JSR PC,EOM ;LOAD GARBAGE CHARACTER, WITH EOM SET
4621 025212 004737 033610 JSR PC,EOM
4622 025216 004737 032346 JSR PC,OCOR ;WAIT FOR OCOR
4623 025222 005003 CLR R3 ;CLEAR BIT COUNTER
4624 025224 104415 000022 DATACLK,22 ;CLOCK DATA
4625 025230 112405 12$: MOVB (R4)+,R5 ;LOAD R5 WITH CHAR
4626 025232 010502 MOV R5,R2 ;LOAD R2 WITH CHAR
4627
4628 ;CHECK FIRST FOUR CHARACTER MESSAGE
4629 ;IN THE BIT WINDOW (0,125,252,377)
4630
4631 025234 010537 025330 MOV R5,71$ ;LOAD FOR STUFF CHECK
4632 025240 012737 102010 033332 MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL
4633 025246 010537 025260 MOV R5,67$ ;LOAD SOFT CHAR FOR BCC
4634 025252 004537 033210 JSR R5,SIMBCC ;CALCULATE SOFT BCC
4635 025256 000010 10 ;SHIFT COUNT
4636 025260 000000 67$: 0 ;CHARACTER
4637 025262 000000 10$: 0 ;OLD BCC
4638 025264 013737 033334 025262 MOV CALBCC,10$ ;LOAD SOFT BCC FOR NEXT SHIFT
4639 025272 104415 000001 64$: DATACLK, 1 ;SHIFT DATA IN TO BIT WINDOW
4640 025276 106002 RORB R2 ;SHIFT SOFT DATA
4641 025300 103005 BCC 65$ ;BR IF A SPACE
4642 025302 004737 032314 JSR PC,GETSI ;LOOK AT BIT WINDOW
4643 025306 103406 BCS 66$ ;BR IF OK (MARK)
4644 025310 104006 HLT 6 ;ERROR, BIT WINDOW WAS A SPACE
4645 025312 000404 BR 66$ ;CONTINUE
4646 025314 004737 032314 65$: JSR PC,GETSI ;LOOK AT BIT WINDOW
4647 025320 103001 BCC 66$ ;BR IF OK (SPACE)
4648 025322 104006 HLT 6 ;ERROR, BIT WINDOW WAS A MARK
4649 025324 66$:
4650 025324 004537 033776 71$: JSR R5,STFFCK
4651 025330 000000 0
4652 025332 000001 1
4653 025334 110237 025330 MOVB R2,71$ ;SHIFT FOR NEXT STUFF CHECK
4654 025340 005203 INC R3 ;BUMP BIT COUNTER
4655 025342 022703 000010 CMP #10,R3 ;DONE FULL 8 BITS YET
4656 025346 001351 BNE 64$ ;BR IF NO
4657 025350 005003 CLR R3 ;CLEAR BIT COUNTER
4658 025352 005300 DEC R0 ;DEC CHARACTER COUNT
4659 025354 001325 BNE 12$ ;BR IF NOT DONE YET

```

```

4660
4661
4662
4663 025356 005137 033334
4664 025362 013700 033334
4665 025366 010037 025430
4666 025372 104415 000001
4667 025376 006000
4668 025400 103005
4669 025402 004737 032314
4670 025406 103406
4671 025410 104014
4672 025412 000404
4673 025414 004737 032314
4674 025420 103001
4675 025422 104014
4676 025424
4677 025424 004537 033776
4678 025430 000000
4679 025432 000001
468^ 025434 010037 025430
4681 025440 005203
4682 025442 022703 000020
4683 025446 001351
4684 025450 005003
4685
4686
4687
4688 025452 012737 000176 001252
4689 025460 104415 000001
4690 025464 106037 001252
4691 025470 103405
4692 025472 004737 032314
4693 025476 103006
4694 025500 104026
4695 025502 000404
4696 025504 004737 032314
4697 025510 103401
4698 025512 104026
4699 025514 005203
4700 025516 022703 000010
4701 025522 001356
4702 025524 005003
4703
4704
4705
4706 025526 104415 000001
4707 025532 004737 032314
4708 025536 103401
4709 025540 104024
4710 025542 005203
4711 025544 022703 000007
4712 025550 001366
4713 025552 104415 000010
4714 025556 005003
4715 025560 104415 000001

;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
COM CALBCC ;ADJUST BCC FOR SDLC
MOV CALBCC,R0 ;PUT BCC IN R0
MOV R0,72$ ;LOAD BCC FOR STUFF CHECK
68$: DATACLK,1 ;SHIFT HARDWARE BCC
ROR R0 ;SHIFT SOFT BCC
BCC 69$ ;BR IF CARRY CLEAR
JSR PC,GETSI ;LOOK AT BIT WINDOW
BCS 70$ ;BR IF OK (MARK)
HLT 14 ;ERROR, CRC WRONG (SPACE)
BR 70$ ;CONTINUE
69$: JSR PC,GETSI ;LOOK AT BIT WINDOW
BCC 70$ ;BR IF OK (SPACE)
HLT 14 ;ERROR, CRC WRONG (MARK)
70$: JSR R5,STFFCK ;CHECK BCC CHAR FOR ZERO STUFFS
72$: 0 ;CHARACTER
1 ;SHIFT COUNT
MOV R0,72$ ;SHIFT SOFTBCC ONCE
INC R3 ;BUMP BIT COUNTER
CMP #20,R3 ;FINISHED BCC YET?
BNE 68$ ;BR IF NO
CLR R3 ;CLEAR BIT COUNTER

;CHECK FOR FLAG TO FOLLOW BCC
MOV #^B<01111110>,TEMP3 ;PUT FLAG CHARACTER IN TEMP3
73$: DATACLK,1 ;CLOCK FLAG ONCE
RORB TEMP3 ;SHIFT SOFT FLAG
BCS 74$ ;BR IF BIT IS MARK
JSR PC,GETSI ;LOOK AT BIT WINDOW
BCC 75$ ;BR IF OK
HLT 26 ;ERROR IN FLAG CHAR
BR 75$
74$: JSR PC,GETSI ;LOOK AT BIT WINDOW
BCS 75$ ;BR IF OK
HLT 26 ;ERROR IN FLAG CHAR
75$: INC R3 ;INC BIT COUNT
CMP #10,R3 ;FLAG DONE YET?
BNE 73$ ;BR IF NO
CLR R3 ;CLEAR BIT COUNT

;CHECK TO SEE IF TRANSMITTER IS MARKING
2$: DATACLK,1 ;CLOCK TRANSMITTER
JSR PC,GETSI ;LOOK AT WINDOW
BCS 3$ ;IT SHOULD BE MARKING
HLT 24 ;ERROR, BIT WAS A SPACE
3$: INC R3 ;BUMP BIT COUNTER
CMP #7,R3 ;DONE YET
BNE 2$ ;BR IF NO
DATACLK,10 ;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
CLR R3 ;CLEAR BIT COUNTER
4$: DATACLK,1 ;SHIFT OUT NEXT BIT

```



```

4716 025564 004737 032314          JSR    PC,GETSI      ;LOOK AT BIT WINDOW
4717 025570 103401          BCS    .+4           ;BR IF IT IS A MARK
4718 025572 104024          HLT    24            ;ERROR, TRANSMITTER IS NOT MARKING
4719 025574 005203          INC    R3            ;INC BIT COUNT
4720 025576 022703 000020        CMP    #20,R3        ;DONE YET?
4721 025602 001366          BNE    4$           ;BR IF NO
4722 025604 104400          5$:   SCOPE         ;SCOPE THIS TEST
4723
4724
4725          ;***** TEST 54 *****
4726          ;*RECEIVER BITSTUFF CRC TEST
4727          ;*THIS TEST CLOCKS A FOUR CHARACTER MESSAGE WITH BCC
4728          ;*AND VERIFYS CORRECT DATA RECEPTION AND BCC MATCH
4729          ;*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
4730          ;*****
4731
4732          ; TEST 54
4733          ;-----
4734 025606 012737 000054 001226 TST54: MOV    #54,TSTNO
4735 025614 012737 026030 001216        MOV    #TST55,NEXT
4736
4737 025622 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
4738 025624 005061 000004          CLR    4(R1)     ;MASTER CLEAR DMC11
4739 025630 104414          ROMCLK         ;CLEAR PORT4
4740 025632 122117          122117         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4741 025634 004737 033676          JSR    PC,CLRIO   ;PUT LINE UNIT IN BITSTUFF MODE
4742 025640 012711 004000          MOV    #BIT11,(R1);DO THIS AFTER MODE IS SET
4743 025644 012702 034116          MOV    #MESDAT,R2;SET LINE UNIT LOOP
4744 025650 012700 000004          MOV    #4,R0      ;LOAD POINTER TO DATA
4745 025654 004737 033500          JSR    PC,SYNLD   ;LOAD CHARACTER COUNT
4746 025660 004737 032500          JSR    PC,OUTRDY ;LOAD 2 FLAG CHARACTERS IN OUT SILO
4747 025664 004537 033634          JSR    R5,MESLD  ;WAIT FOR OUTRDY
4748 025670 034116          MESDAT         ;LOAD SILO WITH 4 CHAR MESS
4749 025672 000004          4              ;ADDRESS OF MESSAGE
4750 025674 004737 033610          JSR    PC,EOM     ;NUMBER OF CHARACTERS
4751 025700 004737 033610          JSR    PC,EOM     ;LOAD GARBAGE CHARACTER, WITH EOM SET
4752 025704 004737 032346          JSR    PC,OCOR    ;WAIT FOR OCOR
4753 025710 104415 000115          DATACLK,115    ;CLOCK DATA
4754 025714 004737 033154          3$:   JSR    PC,INRDY;WAIT FOR INRDY
4755 025720 104414          ROMCLK         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4756 025722 021204          021204         ;GET IN DATA
4757 025724 016104 000004          MOV    4(R1),R4   ;PUT 'FOUND' IN R4
4758 025730 112205          MOVB   (R2)+,R5   ;PUT 'EXPECTED' IN R5
4759 025732 120504          CMPB   R5,R4      ;COMPARE RECEIVED DATA
4760 025734 001401          BEQ    1$         ;BR IF OK
4761 025736 104010          HLT    10         ;DATA ERROR
4762 025740 005300          1$:   DEC    R0     ;DEC CHARACTER COUNT
4763 025742 001364          BNE    3$         ;BR IF NOT DONE YET
4764
4765
4766          ;CHECK TO SEE THAT IN BCC MATCH IS SET
4767
4768 025744 004737 033154          ROMCLK JSR    PC,INRDY  ;WAIT FOR INRDY
4769 025750 104414          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4770 025752 021204          021204         ;GET FIRST HALF OF CRC
4771 025754 116137 000004 001252        MOVB   4(R1),TEMP3;PUT IN TEMP3
  
```



4772 025762 042737 177400 001252  
 4773 025770 004737 033154  
 4774 025774 104414  
 4775 025776 021244  
 4776 026000 016104 000004  
 4777 026004 042704 000374  
 4778 026010 012705 000003  
 4779 026014 120504  
 4780 026016 001401  
 4781 026020 104042  
 4782 026022  
 4783 026022 104414  
 4784 026024 021204  
 4785 026026 104400  
 4786  
 4787  
 4788  
 4789  
 4790  
 4791  
 4792  
 4793  
 4794  
 4795  
 4796  
 4797  
 4798  
 4799  
 4800  
 4801 026030 012737 000055 001226 TST55:  
 4802 026036 012737 027430 001216  
 4803  
 4804 026044 104412  
 4805 026046 005061 000004  
 4806 026052 104414  
 4807 026054 122117  
 4808 026056 004737 033676  
 4809 026062 005037 034114  
 4810  
 4811  
 4812  
 4813 026066 012711 004000  
 4814 026072 012704 034116  
 4815 026076 005037 026226  
 4816 026102 005137 026226  
 4817 026106 012700 000004  
 4818 026112 004737 033500  
 4819 026116 004737 032500  
 4820 026122 004537 033634  
 4821 026126 034116  
 4822 026130 000004  
 4823 026132 004737 033610  
 4824 026136 004737 033610  
 4825 026142 004577 033634  
 4826 026146 034116  
 4827 026150 000004

```

BIC #177400,TEMP3 ;CLEAR HI BYTE
JSR PC,INRDY ;WAIT FOR INRDY
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021244
MOV 4(R1),R4 ;PUT 'FOUND' IN R4
BIC #374,R4 ;CLEAR UNWANTED BITS
MOV #3,R5 ;PUT 'EXPECTED' IN R5
CMPB R5,R4 ;ARE IN BCC MATCH AND BLOCK END SET?
BEQ 25$
HLT 42 ;IN BCC MATCH ERROR
25$:
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021204 ;GET LAST HALF
2$: SCOPE ;SCOPE THIS TEST

```

```

:***** TEST 55 *****
;*BITSTUFF EOM FUNCTION TEST
;*THIS TEST LOADS OUT SILO WITH: 2 FLAGS,4 CHAR MESSAGE,EOM
;*4 CHARACTER MESS,EOM. THE DATA STREAM IS CHECKED TO BE
;*4 CHAR,BCC,FLAG,4 CHAR,BCC,FLAG,MARKS. THIS TEST VERIFYS THAT
;*THE CHARCTERS LOADED WITH EOM SET ARE LOST
;*ALL DATA AND BCC'S ARE CHECKED IN THE BIT WINDOW
;*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
;*RECEIVED DATA IS VERIFIED, AND IN BCC MATCH IS CHECKED
:*****

```

```

: TEST 55
:-----
MOV #55,TSTNO
MOV #TST56,NEXT
MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
CLR 4(R1) ;CLEAR PORT4
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122117 ;PUT LINE UNIT IN BITSTUFF MODE
JSR PC,CLRIO ;DO THIS AFTER MODE IS SET
CLR BITCON ;CONSECUTIVE 1'S COUNTER INIT TO 0

;LOAD OUT DATA SILO

MOV #BIT11,(R1) ;SET LINE UNIT LOOP
MOV #MESDAT,R4 ;LOAD POINTER TO DATA
CLR 10$ ;CLEAR SOFT BCC
COM 10$ ;START AT -1
MOV #4,RO ;LOAD CHARACTER COUNT
JSR PC,SYNLD ;LOAD 2 FLAG CHARACTERS 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,EOM
JSR R5,MESLD ;LOAD FOUR MORE CHARACTERS
MESDAT ;ADDRESS OF MESSAGE
4 ;NUMBER OF CHACTERS

```



81

```

4828 026152 004737 033610 JSR PC,EOM ;SET EOM
4829 026156 004737 033610 JSR PC,EOM ;SET EOM
4830 026162 004737 032346 JSR PC,OCOR ;WAIT FOR OCOR
4831 026166 005003 CLR R3 ;CLEAR BIT COUNTER
4832 026170 104415 000022 DATACLK,22 ;CLOCK DATA
4833 026174 112405 12$: MOVB (R4)+,R5 ;LOAD R5 WITH CHAR
4834 026176 010502 MOV R5,R2 ;LOAD R2 WITH CHAR
4835
4836 ;CHECK FIRST FOUR CHARACTER MESSAGE
4837 ;IN THE BIT WINDOW (0,125,252,377)
4838
4839 026200 010537 026274 MOV R5,71$ ;LOAD FOR STUFF CHECK
4840 026204 012737 102010 033332 MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL
4841 026212 010537 026224 MOV R5,67$ ;LOAD SOFT CHAR FOR BCC
4842 026216 004537 033210 JSR R5,SIMBCC ;CALCULATE SOFT BCC
4843 026222 000010 10 ;SHIFT COUNT
4844 026224 000000 67$: 0 ;CHARACTER
4845 026226 000000 10$: 0 ;OLD BCC
4846 026230 013737 033334 026226 MOV CALBCC,10$ ;LOAD SOFT BCC FOR NEXT SHIFT
4847 026236 104415 000001 64$: DATACLK, 1 ;SHIFT DATA IN TO BIT WINDOW
4848 026242 106002 RORB R2 ;SHIFT SOFT DATA
4849 026244 103005 BCC 65$ ;BR IF A SPACE
4850 026246 004737 032314 JSR PC,GETSI ;LOOK AT BIT WINDOW
4851 026252 103406 BCS 66$ ;BR IF OK (MARK)
4852 026254 104006 HLT 6 ;ERROR, BIT WINDOW WAS A SPACE
4853 026256 000404 BR 66$ ;CONTINUE
4854 026260 004737 032314 65$: JSR PC,GETSI ;LOOK AT BIT WINDOW
4855 026264 103001 BCC 66$ ;BR IF OK (SPACE)
4856 026266 104006 HLT 6 ;ERROR, BIT WINDOW WAS A MARK
4857 026270 66$:
4858 026270 004537 033776 JSR R5,STFFCK
4859 026274 000000 71$: 0
4860 026276 000001 1
4861 026300 110237 026274 MOVB R2,71$ ;SHIFT FOR NEXT STUFF CHECK
4862 026304 005203 INC R3 ;BUMP BIT COUNTER
4863 026306 022703 000010 CMP #10,R3 ;DONE FULL 8 BITS YET
4864 026312 001351 BNE 64$ ;BR IF NO
4865 026314 005003 CLR R3 ;CLEAR BIT COUNTER
4866 026316 005300 DEC R0 ;DEC CHARACTER COUNT
4867 026320 001325 BNE 12$ ;BR IF NOT DONE YET
4868
4869 ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4870
4871 026322 005137 033334 COM CALBCC ;ADJUST BCC FOR SDLC
4872 026326 013700 033334 MOV CALBCC,R0 ;PUT BCC IN R0
4873 026332 010037 026374 MOV R0,72$ ;LOAD BCC FOR STUFF CHECK
4874 026336 104415 000001 68$: DATACLK,1 ;SHIFT HARDWARE BCC
4875 026342 006000 ROR R0 ;SHIFT SOFT BCC
4876 026344 103005 BCC 69$ ;BR IF CARRY CLEAR
4877 026346 004737 032314 JSR PC,GETSI ;LOOK AT BIT WINDOW
4878 026352 103406 BCS 70$ ;BR IF OK (MARK)
4879 026354 104014 HLT 14 ;ERROR, CRC WRONG (SPACE)
4880 026356 000404 BR 70$ ;CONTINUE
4881 026360 004737 032314 69$: JSR PC,GETSI ;LOOK AT BIT WINDOW
4882 026364 103001 BCC 70$ ;BR IF OK (SPACE)
4883 026366 104014 HLT 14 ;ERROR, CRC WRONG (MARK)

```



4884	026370				70\$:			
4885	026370	004537	033776			JSR	R5,STFFCK	;CHECK BCC CHAR FOR ZERO STUFFS
4886	026374	000000			72\$:	0		;CHARACTER
4887	026376	000001				1		;SHIFT COUNT
4888	026400	010037	026374			MOV	R0,72\$	;SHIFT SOFTBCC ONCE
4889	026404	005203				INC	R3	;BUMP BIT COUNTER
4890	026406	022703	000020			CMP	#20,R3	;FINISHED BCC YET?
4891	026412	001351				BNE	68\$	;BR IF NO
4892	026414	005003				CLR	R3	;CLEAR BIT COUNTER
4893								
4894								;CHECK FOR FLAG TO FOLLOW BCC
4895								
4896	026416	012737	000176	001252		MOV	#*B<01111110>,TEMP3	;PUT FLAG CHARACTER IN TEMP3
4897	026424	104415	000001		73\$:	DATACLK,	1	;CLOCK FLAG ONCE
4898	026430	106037	001252			RORB	TEMP3	;SHIFT SOFT FLAG
4899	026434	103405				BCS	74\$	;BR IF BIT IS MARK
4900	026436	004737	032314			JSR	PC,GETSI	;LOOK AT BIT WINDOW
4901	026442	103006				BCC	75\$	;BR IF OK
4902	026444	104026				HLT	26	;ERROR IN FLAG CHAR
4903	026446	000404				BR	75\$	
4904	026450	004737	032314		74\$:	JSR	PC,GETSI	;LOOK AT BIT WINDOW
4905	026454	103401				BCS	75\$	;BR IF OK
4906	026456	104026				HLT	26	;ERROR IN FLAG CHAR
4907	026460	005203			75\$:	INC	R3	;INC BIT COUNT
4908	026462	022703	000010			CMP	#10,R3	;FLAG DONE YET?
4909	026466	001356				BNE	73\$	;BR IF NO
4910	026470	005003				CLR	R3	;CLEAR BIT COUNT
4911	026472	012700	000004			MOV	#4,R0	;RESET CHARACTER COUNTER
4912	026476	012704	034116			MOV	#MESDAT,R4	;LOAD MESSAGE POINTER
4913	026502	005037	026544			CLR	11\$	;CLR SOFT BCC
4914	026506	005137	026544			COM	11\$	;ADJUST TO -1 FOR SDLC
4915	026512	112405			13\$:	MOVB	(R4)+,R5	;LOAD CHAR IN R5
4916	026514	010502				MOV	R5,R2	;LOAD CHAR IN R2
4917								
4918								;CHECK SECOND MESSAGE IN THE BIT WINDOW (0,125,252,377)
4919								
4920	026516	010537	026612			MOV	R5,83\$	;LOAD FOR STUFF CHECK
4921	026522	012737	102010	033332		MOV	#CRC.CCITT,XPOLY	;LOAD POLYNOMIAL
4922	026530	010537	026542			MOV	R5,79\$	;LOAD SOFT CHAR FOR BCC
4923	026534	004537	033210			JSR	R5,SIMBCC	;CALCULATE SOFT BCC
4924	026540	000010				10		;SHIFT COUNT
4925	026542	000000			79\$:	0		;CHARACTER
4926	026544	000000			11\$:	0		;OLD BCC
4927	026546	013737	033334	026544		MOV	CALBCC,11\$	;LOAD SOFT BCC FOR NEXT SHIFT
4928	026554	104415	000001		76\$:	DATACLK,	1	;SHIFT DATA IN TO BIT WINDOW
4929	026560	106002				RORB	R2	;SHIFT SOFT DATA
4930	026562	103005				BCC	77\$	;BR IF A SPACE
4931	026564	004737	032314			JSR	PC,GETSI	;LOOK AT BIT WINDOW
4932	026570	103406				BCS	78\$	;BR IF OK (MARK)
4933	026572	104006				HLT	6	;ERROR, BIT WINDOW WAS A SPACE
4934	026574	000404				BR	78\$	;CONTINUE
4935	026576	004737	032314		77\$:	JSR	PC,GETSI	;LOOK AT BIT WINDOW
4936	026602	103001				BCC	78\$	;BR IF OK (SPACE)
4937	026604	104006				HLT	6	;ERROR, BIT WINDOW WAS A MARK
4938	026606				78\$:			
4939	026606	004537	033776			JSR	R5,STFFCK	



```

4940 026612 000000          83$: 0
4941 026614 000001          1
4942 026616 110237 026612  MOVB  R2,83$      ;SHIFT FOR NEXT STUFF CHECK
4943 026622 005203          INC  R3           ;BUMP BIT COUNTER
4944 026624 022703 000010  CMP   #10,R3     ;DONE FULL 8 BITS YET
4945 026630 001351          BNE  76$         ;BR IF NO
4946 026632 005003          CLR  R3         ;CLEAR BIT COUNTER
4947 026634 005300          DEC  R0         ;DEC CHARACTER COUNT
4948 026636 001325          BNE  13$         ;BR IF NOT DONE YET
4949
4950                          ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4951
4952 026640 005137 033334  COM   CALBCC     ;ADJUST BCC FOR SDLC
4953 026644 013700 033334  MOV   CALBCC,R0 ;PUT BCC IN R0
4954 026650 010037 026712  MOV   R0,84$    ;LOAD BCC FOR STUFF CHECK
4955 026654 104415 000001  80$: DATACLK,1  ;SHIFT HARDWARE BCC
4956 026660 006000          ROR  R0         ;SHIFT SOFT BCC
4957 026662 103005          BCC  81$       ;BR IF CARRY CLEAR
4958 026664 004737 032314  JSR  PC,GETSI   ;LOOK AT BIT WINDOW
4959 026670 103406          BCS  82$       ;BR IF OK (MARK)
4960 026672 104014          HLT  14        ;ERROR, CRC WRONG (SPACE)
4961 026674 000404          BR   82$       ;CONTINUE
4962 026676 004737 032314  81$: JSR  PC,GETSI   ;LOOK AT BIT WINDOW
4963 026702 103001          BCC  82$       ;BR IF OK (SPACE)
4964 026704 104014          HLT  14        ;ERROR, CRC WRONG (MARK)
4965 026706
4966 026706 004537 033776  82$: JSR  R5,STFFCK ;CHECK BCC CHAR FOR ZERO STUFFS
4967 026712 000000          84$: 0           ;CHARACTER
4968 026714 000001          i           ;SHIFT COUNT
4969 026716 010037 026712  MOV   R0,84$    ;SHIFT SOFTBCC ONCE
4970 026722 005203          INC  R3         ;BUMP BIT COUNTER
4971 026724 022703 000020  CMP   #20,R3     ;FINISHED BCC YET?
4972 026730 001351          BNE  80$       ;BR IF NO
4973 026732 005003          CLR  R3         ;CLEAR BIT COUNTER
4974
4975                          ;CHECK FOR FLAG TO FOLLOW BCC
4976
4977 026734 012737 000176 001252  MOV   #^B<01111110>,TEMP3 ;PUT FLAG CHARACTER IN TEMP3
4978 026742 104415 000001  85$: DATACLK, 1  ;CLOCK FLAG ONCE
4979 026746 106037 001252  RORB  TEMP3     ;SHIFT SOFT FLAG
4980 026752 103405          BCS  86$       ;BR IF BIT IS MARK
4981 026754 004737 032314  JSR  PC,GETSI   ;LOOK AT BIT WINDOW
4982 026760 103006          BCC  87$       ;BR IF OK
4983 026762 104026          HLT  26        ;ERROR IN FLAG CHAR
4984 026764 000404          BR   87$
4985 026766 004737 032314  86$: JSR  PC,GETSI   ;LOOK AT BIT WINDOW
4986 026772 103401          BCS  87$       ;BR IF OK
4987 026774 104026          HLT  26        ;ERROR IN FLAG CHAR
4988 026776 005203          87$: INC  R3         ;INC BIT COUNT
4989 027000 022703 000010  CMP   #10,R3     ;FLAG DONE YET?
4990 027004 001356          BNE  85$       ;BR IF NO
4991 027006 005003          CLR  R3         ;CLEAR BIT COUNT
4992
4993                          ;CHECK TO SEE IF TRANSMITTER IS MARKING
4994
4995 027010 104415 000001  2$:  DATACLK, 1  ;CLOCK TRANSMITTER

```

```

4996 027014 004737 032314      JSR    PC,GETSI      ;LOOK AT WINDOW
4997 027020 103401              BCS    3$            ;IT SHOULD BE MARKING
4998 027022 104024              HLT    24            ;ERROR, BIT WAS A SPACE
4999 027024 005203              3$:   INC    R3      ;BUMP BIT COUNTER
5000 027026 022703 000007      CMP    #7,R3        ;DONE YET
5001 027032 001366              BNE    2$            ;BR IF NO
5002 027034 104415 000010      DATACLK,          10 ;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
5003 027040 005003              CLR    R3           ;CLEAR BIT COUNTER
5004 027042 104415 000001      4$:   DATACLK,          1 ;SHIFT OUT NEXT BIT
5005 027046 004737 032314      JSR    PC,GETSI      ;LOOK AT BIT WINDOW
5006 027052 103401              BCS    .+4          ;BR IF IT IS A MARK
5007 027054 104024              HLT    24            ;ERROR, TRANSMITTER IS NOT MARKING
5008 027056 005203              INC    R3           ;INC BIT COUNT
5009 027060 022703 000020      CMP    #20,R3       ;DONE YET?
5010 027064 001366              BNE    4$            ;BR IF NO
5011
5012                          ;CHECK TO SEE THAT FIRST FOUR CHARACTER MESSAGE
5013                          ;WAS RECEIVED CORRECTLY (0,125,252,377)
5014
5015 027066 104415 000001      DATACLK,          1 ;GET LAST BIT IN RECEIVER
5016 027072 012703 000004      MOV    #4,R3        ;R3=CHARACTER COUNT
5017 027076 012702 034116      MOV    #MESDAT,R2   ;LOAD MESSAGE POINTER IN R2
5018 027102 004737 033154      40$:  JSR    PC,INRDY    ;WAIT FOR INRDY
5019 027106 104414              ROMCLK              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5020 027110 021204
5021 027112 016104 000004      MOV    4(R1),R4     ;PUT "FOUND" IN R4
5022 027116 112205              MOVB   (R2)+,R5     ;PUT "EXPECTED" IN R5
5023 027120 120504              CMPB   R5,R4        ;IS RECEIVED DATA CORRECT?
5024 027122 001401              BEQ    41$          ;BR IF YES
5025 027124 104010              HLT    10            ;RECEIVE DATA ERROR
5026 027126 005303              41$:  DEC    R3          ;DEC CHARACTER COUNT
5027 027130 001364              BNE    40$          ;BR IF NOT DONE YET
5028
5029
5030                          ;CHECK TO SEE THAT IN BCC MATCH IS SET
5031                          ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5032
5033 027132 004737 033154      ROMCLK JSR    PC,INRDY    ;WAIT FOR INRDY
5034 027136 104414              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5035 027140 021204              021204              ;GET FIRST HALF OF CRC
5036 027142 116137 000004 001252 MOVB   4(R1),TEMP3   ;PUT IN TEMP3
5037 027150 042737 177400 001252 BIC    #177400,TEMP3 ;CLEAR HI BYTE
5038 027156 004737 033154      ROMCLK JSR    PC,INRDY    ;WAIT FOR INRDY
5039 027162 104414              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5040 027164 021244
5041 027166 016104 000004      MOV    4(R1),R4     ;PUT "FOUND" IN R4
5042 027172 042704 000374      BIC    #374,R4      ;CLEAR UNWANTED BITS
5043 027176 012705 000003      MOV    #3,R5        ;PUT "EXPECTED" IN R5
5044 027202 120504              CMPB   R5,R4        ;ARE IN BCC MATCH AND BLOCK END SET?
5045 027204 001401              BEQ    50$          ;IN BCC MATCH ERROR
5046 027206 104042              50$:  HLT    42
5047 027210
5048 027210 104414              ROMCLK              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5049 027212 021204              021204              ;GET LAST HALF
5050 027214 116137 000004 001251 MOVB   4(R1),TEMP2+1 ;PUT IN TEMP2
5051 027222 042737 000377 001250 BIC    #377,TEMP2   ;CLEAR LO BYTE

```



85

```

5052 027230 053737 001250 001252      BIS      TEMP2,TEMP3      ;16 BIT BCC NOW IN TEMP3
5053 027236 023737 033334 001252      CMP      CALBCC,TEMP3     ;IS IT CORRECT?
5054 027244 001401                      BEQ      42$              ;BR IF OK
5055 027246 104027                      HLT      27
5056
5057                                     ;CHECK TO SEE THAT SECOND FOUR CHARACTER MESSAGE
5058                                     ;WAS RECEIVED CORRECTLY (0,125,252,377)
5059
5060 027250 012703 000004      42$:    MOV      #4,R3          ;R3=CHARACTER COUNT
5061 027254 012702 034116      MOV      #MESDAT,R2       ;LOAD MESSAGE POINTER IN R2
5062 027260 004737 033154      43$:    JSR      PC,INRDY    ;WAIT FOR INRDY
5063 027264 104414      ROMCLK                      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5064 027266 021204
5065 027270 016104 000004      MOV      4(R1),R4         ;PUT 'FOUND' IN R4
5066 027274 112205      MOVB     (R2)+,R5        ;PUT 'EXPECTED' IN R5
5067 027276 120504      CMPB     R5,R4           ;IS RECEIVED DATA CORRECT?
5068 027300 001401      BEQ      44$              ;BR IF YES
5069 027302 104010      HLT      10              ;RECEIVE DATA ERROR
5070 027304 005303      44$:    DEC      R3          ;DEC CHARACTER COUNT
5071 027306 001364      BNE      43$              ;BR IF NOT DONE YET
5072
5073
5074                                     ;CHECK TO SEE THAT IN BCC MATCH IS SET
5075                                     ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5076
5077 027310 004737 033154      ROMCLK  JSR      PC,INRDY    ;WAIT FOR INRDY
5078 027314 104414      ROMCLK                      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5079 027316 021204
5080 027320 116137 000004 001252      MOVB     4(R1),TEMP3     ;GET FIRST HALF OF CRC
5081 027326 042737 177400 001252      BIC      #177400,TEMP3   ;PUT IN TEMP3
5082 027334 004737 033154      JSR      PC,INRDY       ;CLEAR HI BYTE
5083 027340 104414      ROMCLK                      ;WAIT FOR INRDY
5084 027342 021244
5085 027344 016104 000004      MOV      4(R1),R4         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5086 027350 042704 000374      MOV      #374,R4        ;PUT 'FOUND' IN R4
5087 027354 012705 000003      BIC      #374,R4         ;CLEAR UNWANTED BITS
5088 027360 120504      MOV      #3,R5           ;PUT 'EXPECTED' IN R5
5089 027362 001401      CMPB     R5,R4           ;ARE IN BCC MATCH AND BLOCK END SET?
5090 027364 104042      BEQ      51$              ;IN BCC MATCH ERROR
5091 027366
5092 027366 104414      51$:    HLT      42
5093 027370 021204      ROMCLK                      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5094 027372 116137 000004 001251      MOVB     4(R1),TEMP2+1   ;GET LAST HALF
5095 027400 042737 000377 001250      BIC      #377,TEMP2     ;PUT IN TEMP2
5096 027406 053737 001250 001252      BIS      TEMP2,TEMP3     ;CLEAR LO BYTE
5097 027414 023737 033334 001252      BIS      TEMP2,TEMP3     ;16 BIT BCC NOW IN TEMP3
5098 027422 001401      CMP      CALBCC,TEMP3    ;IS IT CORRECT?
5099 027424 104027      BEQ      5$              ;BR IF OK
5100 027426 104400      5$:    HLT      27
5101                                     ;SCOPE THIS TEST
5102
5103                                     ;***** TEST 56 *****
5104                                     ;*BITSTUFF EOM FUNCTION TEST
5105                                     ;*THIS TEST LOADS OUT SILO WITH: 2 FLAGS,4 CHAR MESSAGE,EOM
5106                                     ;*SOM,4 CHAR MESS,EOM. THE DATA STREAM IS CHECKED TO BE
5107                                     ;*4 CHAR,BCC,FLAG,4 CHAR,BCC,FLAG,MARKS. THIS TEST VERIFYS THAT

```

```

5108                                     ;*THE CHARCTERS LOADED WITH EOM SET ARE LOST
5109                                     ;*ALSO THAT THE CHAR LOADED WITH SOM IS NOT IN THE BCC
5110                                     ;*ALL DATA AND BCC'S ARE CHECKED IN THE BIT WINDOW
5111                                     ;*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
5112                                     ;*RECEIVED DATA IS VERIFIED, AND IN BCC MATCH IS CHECKED
5113                                     ;:*****
5114                                     ;:*****
5115                                     ;:*****
5116                                     ;:*****
5117                                     ;: TEST 56
5118 027430 012737 000056 001226 TST56: MOV #56,TSTNO
5119 027436 012737 031110 001216 MOV #TST57,NEXT
5120 027444 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
5121 027446 005061 000004 CLR 4(R1) ;MASTER CLEAR DMC11
5122 027452 104414 ROMCLK ;CLEAR PORT4
5123 027454 122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5124 027456 004737 033676 JSR PC,CLRIO ;PUT LINE UNIT IN BITSTUFF MODE
5125 027462 005037 034114 CLR BITCON ;DO THIS AFTER MODE IS SET
5126                                     ;CONSECUTIVE 1'S COUNTER INIT TO 0
5127                                     ;LOAD OUT DATA SILO
5128
5129 027466 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
5130 027472 012704 034116 MOV #MESDAT,R4 ;LOAD POINTER TO DATA
5131 027476 005037 027632 CLR 10$ ;CLEAR SOFT BCC
5132 027502 005137 027632 COM 10$ ;START AT -1
5133 027506 012700 000004 MOV #4,R0 ;LOAD CHARACTER COUNT
5134 027512 004737 033500 JSR PC,SYNLD ;LOAD 2 FLAG CHARACTERS IN OUT SILO
5135 027516 004737 032500 JSR PC,OUTRDY ;WAIT FOR OUTRDY
5136 027522 004537 033634 JSR R5,MESLD ;LOAD SILO WITH 4 CHAR MESS
5137 027526 034116 MESDAT ;ADDRESS OF MESSAGE
5138 027530 000004 4 ;NUMBER OF CHARACTERS
5139 027532 004737 033610 JSR PC,EOM ;LOAD GARBAGE CHARACTER, WITH EOM SET
5140 027536 004737 033610 JSR PC,EOM
5141 027542 004737 033560 JSR PC,SOM ;LOAD GARBAGE CHAR WITH SOM SET
5142 027546 004537 033634 JSR R5,MESLD ;LOAD FOUR MORE CHARACTERS
5143 027552 034116 MESDAT ;ADDRESS OF MESSAGE
5144 027554 000004 4 ;NUMBER OF CHACTERS
5145 027556 004737 033610 JSR PC,EOM ;SET EOM
5146 027562 004737 033610 JSR PC,EOM ;SET EOM
5147 027566 004737 032346 JSR PC,OCOR ;WAIT FOR OCOR
5148 027572 005003 CLR R3 ;CLEAR BIT COUNTER
5149 027574 104415 000022 DATACLK,22 ;CLOCK DATA
5150 027600 112405 12$: MOVB (R4)+,R5 ;LOAD R5 WITH CHAR
5151 027602 010502 MOV R5,R2 ;LOAD R2 WITH CHAR
5152
5153                                     ;CHECK FIRST FOUR CHARACTER MESSAGE
5154                                     ;IN THE BIT WINDOW (0,125,252,377)
5155
5156 027604 010537 027700 MOV R5,71$ ;LOAD FOR STUFF CHECK
5157 027610 012737 102010 033332 MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL
5158 027616 010537 027630 MOV R5,67$ ;LOAD SOFT CHAR FOR BCC
5159 027622 004537 033210 JSR R5,SIMBCC ;CALCULATE SOFT BCC
5160 027626 000010 10 ;SHIFT COUNT
5161 027630 000000 67$: 0 ;CHARACTER
5162 027632 000000 10$: 0 ;OLD BCC
5163 027634 013737 033334 027632 MOV CALBCC,10$ ;LOAD SOFT BCC FOR NEXT SHIFT

```



```

5164 027642 104415 000001      64$:  DATACLK,      1      ;SHIFT DATA IN TO BIT WINDOW
5165 027646 106002              RORB      R2      ;SHIFT SOFT DATA
5166 027650 103005              BCC      65$     ;BR IF A SPACE
5167 027652 004737 032314      JSR      PC,GETSI ;LOOK AT BIT WINDOW
5168 027656 103406              BCS      66$     ;BR IF OK (MARK)
5169 027660 104006              HLT      6       ;ERROR, BIT WINDOW WAS A SPACE
5170 027662 000404              BR       66$     ;CONTINUE
5171 027664 004737 032314      65$:  JSR      PC,GETSI ;LOOK AT BIT WINDOW
5172 027670 103001              BCC      66$     ;BR IF OK (SPACE)
5173 027672 104006              HLT      6       ;ERROR, BIT WINDOW WAS A MARK
5174 027674              66$:  JSR      R5,STFFCK
5175 027674 004537 033776      71$:  0
5176 027700 000000              1
5177 027702 000001              MOVB     R2,71$   ;SHIFT FOR NEXT STUFF CHECK
5178 027704 110237 027700              INC      R3       ;BUMP BIT COUNTER
5179 027710 005203              CMP      #10,R3  ;DONE FULL 8 BITS YET
5180 027712 022703 000010      BNE     64$     ;BR IF NO
5181 027716 001351              CLR      R3       ;CLEAR BIT COUNTER
5182 027720 005003              DEC      R0       ;DEC CHARACTER COUNT
5183 027722 005300              BNE     12$     ;BR IF NOT DONE YET
5184 027724 001325
5185
5186              ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
5187
5188 027726 005137 033334      COM      CALBCC   ;ADJUST BCC FOR SDLC
5189 027732 013700 033334      MOV      CALBCC,R0 ;PUT BCC IN R0
5190 027736 010037 030000      MOV      R0,72$  ;LOAD BCC FOR STUFF CHECK
5191 027742 104415 000001      68$:  DATACLK,1      ;SHIFT HARDWARE BCC
5192 027746 006000              ROR      R0       ;SHIFT SOFT BCC
5193 027750 103005              BCC      69$     ;BR IF CARRY CLEAR
5194 027752 004737 032314      JSR      PC,GETSI ;LOOK AT BIT WINDOW
5195 027756 103406              BCS      70$     ;BR IF OK (MARK)
5196 027760 104014              HLT      14      ;ERROR, CRC WRONG (SPACE)
5197 027762 000404              BR       70$     ;CONTINUE
5198 027764 004737 032314      69$:  JSR      PC,GETSI ;LOOK AT BIT WINDOW
5199 027770 103001              BCC      70$     ;BR IF OK (SPACE)
5200 027772 104014              HLT      14      ;ERROR, CRC WRONG (MARK)
5201 027774
5202 027774 004537 033776      70$:  JSR      R5,STFFCK ;CHECK BCC CHAR FOR ZERO STUFFS
5203 030000 000000      72$:  0
5204 030002 000001              1
5205 030004 010037 030000      MOV      R0,72$  ;SHIFT SOFTBCC ONCE
5206 030010 005203              INC      R3       ;BUMP BIT COUNTER
5207 030012 022703 000020      CMP      #20,R3  ;FINISHED BCC YET?
5208 030016 001351              BNE     68$     ;BR IF NO
5209 030020 005003              CLR      R3       ;CLEAR BIT COUNTER
5210
5211              ;CHECK FOR FLAG TO FOLLOW BCC
5212
5213 030022 012737 000176 001252      73$:  MOV      #^B<01111110>,TEMP3 ;PUT FLAG CHARACTER IN TEMP3
5214 030030 104415 000001              DATACLK,1      ;CLOCK FLAG ONCE
5215 030034 106037 001252      RORB     TEMP3    ;SHIFT SOFT FLAG
5216 030040 103405              BCS     74$     ;BR IF BIT IS MARK
5217 030042 004737 032314      JSR     PC,GETSI ;LOOK AT BIT WINDOW
5218 030046 103006              BCC     75$     ;BR IF OK
5219 030050 104026              HLT     26      ;ERROR IN FLAG CHAR

```

```

5220 030052 000404
5221 030054 004737 032314 74$: JSR PC,GETSI ;LOOK AT BIT WINDOW
5222 030060 103401 BCS 75$ ;BR IF OK
5223 030062 104026 HLT 26 ;ERROR IN FLAG CHAR
5224 030064 005203 75$: INC R3 ;INC BIT COUNT
5225 030066 022703 000010 CMP #10,R3 ;FLAG DONE YET?
5226 030072 001356 BNE 73$ ;BR IF NO
5227 030074 005003 CLR R3 ;CLEAR BIT COUNT
5228
5229 ;CHECK FOR ANOTHER FLAG CAUSED BY THE SOM
5230
5231 030076 012737 000176 001252 MOV #^B<0111!110>,TEMP3 ;PUT FLAG CHARACTER IN TEMP3
5232 030104 104415 000001 76$: DATACLK, 1 ;CLOCK FLAG ONCE
5233 030110 106037 001252 RORB TEMP3 ;SHIFT SOFT FLAG
5234 030114 103405 BCS 77$ ;BR IF BIT IS MARK
5235 030116 004737 032314 JSR PC,GETSI ;LOOK AT BIT WINDOW
5236 030122 103006 BCC 78$ ;BR IF OK
5237 030124 104026 HLT 26 ;ERROR IN FLAG CHAR
5238 030126 000404 BR 78$
5239 030130 004737 032314 77$: JSR PC,GETSI ;LOOK AT BIT WINDOW
5240 030134 103401 BCS 78$ ;BR IF OK
5241 030136 104026 HLT 26 ;ERROR IN FLAG CHAR
5242 030140 005203 78$: INC R3 ;INC BIT COUNT
5243 030142 022703 000010 CMP #10,R3 ;FLAG DONE YET?
5244 030146 001356 BNE 76$ ;BR IF NO
5245 030150 005003 CLR R3 ;CLEAR BIT COUNT
5246 030152 012700 000004 MOV #4,R0 ;RESET CHARACTER COUNTER
5247 030156 012704 034116 MOV #MESDAT,R4 ;LOAD MESSAGE POINTER
5248 030162 005037 030224 CLR 11$ ;CLR SOFT BCC
5249 030166 005137 030224 COM 11$ ;ADJUST TO -1 FOR SDLC
5250 030172 112405 13$: MOVB (R4)+,R5 ;LOAD CHAR IN R5
5251 030174 010502 MOV R5,R2 ;LOAD CHAR IN R2
5252
5253 ;CHECK SECOND MESSAGE IN THE BIT WINDOW (0,125,252,377)
5254
5255 030176 010537 030272 MOV R5,86$ ;LOAD FOR STUFF CHECK
5256 030202 012737 102010 033332 MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL
5257 030210 010537 030222 MOV R5,82$ ;LOAD SOFT CHAR FOR BCC
5258 030214 004537 033210 JSR R5,SIMBCC ;CALCULATE SOFT BCC
5259 030220 000010 10 ;SHIFT COUNT
5260 030222 000000 82$: 0 ;CHARACTER
5261 030224 000000 11$: 0 ;OLD BCC
5262 030226 013737 033334 030224 MOV CALBCC,11$ ;LOAD SOFT BCC FOR NEXT SHIFT
5263 030234 104415 000001 79$: DATACLK, 1 ;SHIFT DATA IN TO BIT WINDOW
5264 030240 106002 RORB R2 ;SHIFT SOFT DATA
5265 030242 103005 BCC 80$ ;BR IF A SPACE
5266 030244 004737 032314 JSR PC,GETSI ;LOOK AT BIT WINDOW
5267 030250 103406 BCS 81$ ;BR IF OK (MARK)
5268 030252 104006 HLT 6 ;ERROR, BIT WINDOW WAS A SPACE
5269 030254 000404 BR 81$ ;CONTINUE
5270 030256 004737 032314 80$: JSR PC,GETSI ;LOOK AT BIT WINDOW
5271 030262 103001 BCC 81$ ;BR IF OK (SPACE)
5272 030264 104006 HLT 6 ;ERROR, BIT WINDOW WAS A MARK
5273 030266
5274 030266 004537 033776 81$: JSR R5,STFFCK
5275 030272 000000 86$: 0

```



```

5276 030274 000001          1
5277 030276 110237 030272  MOVB   R2,86$      ;SHIFT FOR NEXT STUFF CHECK
5278 030302 005203          INC     R3          ;BUMP BIT COUNTER
5279 030304 022703 000010  CMP    #10,R3     ;DONE FULL 8 BITS YET
5280 030310 001351          BNE    79$        ;BR IF NO
5281 030312 005003          CLR    R3          ;CLEAR BIT COUNTER
5282 030314 005300          DEC    R0          ;DEC CHARACTER COUNT
5283 030316 001325          BNE    13$        ;BR IF NOT DONE YET
5284
5285                          ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
5286
5287 030320 005137 033334  COM    CALBCC      ;ADJUST BCC FOR SDLC
5288 030324 013700 033334  MOV    CALBCC,R0  ;PUT BCC IN R0
5289 030330 010037 030372  MOV    R0,87$    ;LOAD BCC FOR STUFF CHECK
5290 030334 104415 000001  83$:  DATACLK,1    ;SHIFT HARDWARE BCC
5291 030340 006000          ROR    R0          ;SHIFT SOFT BCC
5292 030342 103005          BCC    84$        ;BR IF CARRY CLEAR
5293 030344 004737 032314  JSR    PC,GETSI   ;LOOK AT BIT WINDOW
5294 030350 103406          BCS    85$        ;BR IF OK (MARK)
5295 030352 104014          HLT    14         ;ERROR, CRC WRONG (SPACE)
5296 030354 000404          BR     85$        ;CONTINUE
5297 030356 004737 032314  84$:  JSR    PC,GETSI   ;LOOK AT BIT WINDOW
5298 030362 103001          BCC    85$        ;BR IF OK (SPACE)
5299 030364 104014          HLT    14         ;ERROR, CRC WRONG (MARK)
5300
5301 030366 004537 033776  85$:  JSR    R5,STFFCK  ;CHECK BCC CHAR FOR ZERO STUFFS
5302 030372 000000  87$:  0            ;CHARACTER
5303 030374 000001          1            ;SHIFT COUNT
5304 030376 010037 030372  MOV    R0,87$    ;SHIFT SOFTBCC ONCE
5305 030402 005203          INC    R3          ;BUMP BIT COUNTER
5306 030404 022703 000020  CMP    #20,R3     ;FINISHED BCC YET?
5307 030410 001351          BNE    83$        ;BR IF NO
5308 030412 005003          CLR    R3          ;CLEAR BIT COUNTER
5309
5310                          ;CHECK FOR FLAG TO FOLLOW BCC
5311
5312 030414 012737 000176 001252  MOV    #^B<01111110>,TEMP3 ;PUT FLAG CHARACTER IN TEMP3
5313 030422 104415 000001  88$:  DATACLK, 1    ;CLOCK FLAG ONCE
5314 030426 106037 001252  RORB   TEMP3      ;SHIFT SOFT FLAG
5315 030432 103405          BCS    89$        ;BR IF BIT IS MARK
5316 030434 004737 032314  JSR    PC,GETSI   ;LOOK AT BIT WINDOW
5317 030440 103006          BCC    90$        ;BR IF OK
5318 030442 104026          HLT    26         ;ERROR IN FLAG CHAR
5319 030444 000404          BR     90$        ;CONTINUE
5320 030446 004737 032314  89$:  JSR    PC,GETSI   ;LOOK AT BIT WINDOW
5321 030452 103401          BCS    90$        ;BR IF OK
5322 030454 104026          HLT    26         ;ERROR IN FLAG CHAR
5323 030456 005203  90$:  INC    R3          ;INC BIT COUNT
5324 030460 022703 000010  CMP    #10,R3     ;FLAG DONE YET?
5325 030464 001356          BNE    88$        ;BR IF NO
5326 030466 005003          CLR    R3          ;CLEAR BIT COUNT
5327
5328                          ;CHECK TO SEE IF TRANSMITTER IS MARKING
5329
5330 030470 104415 000001  2$:  DATACLK, 1    ;CLOCK TRANSMITTER
5331 030474 004737 032314  JSR    PC,GETSI   ;LOOK AT WINDOW

```

```

5332 030500 103401          BCS 3$           ;IT SHOULD BE MARKING
5333 030502 104024          HLT 24           ;ERROR, BIT WAS A SPACE
5334 030504 005203          3$: INC R3        ;BUMP BIT COUNTER
5335 030506 022703 000007  CMP #7,R3       ;DONE YET
5336 030512 001366          BNE 2$           ;BR IF NO
5337 030514 104415 000010  DATACLK, 10    ;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
5338 030520 005003          CLR R3          ;CLEAR BIT COUNTER
5339 030522 104415 000001  4$: DATACLK, 1  ;SHIFT OUT NEXT BIT
5340 030526 004737 032314  JSR PC,GETSI   ;LOOK AT BIT WINDOW
5341 030532 103401          BCS .+4        ;BR IF IT IS A MARK
5342 030534 104024          HLT 24         ;ERROR, TRANSMITTER IS NOT MARKING
5343 030536 005203          INC R3         ;INC BIT COUNT
5344 030540 022703 000020  CMP #20,R3     ;DONE YET?
5345 030544 001366          BNE 4$         ;BR IF NO
5346
5347          ;CHECK TO SEE THAT FIRST FOUR CHARACTER MESSAGE
5348          ;WAS RECEIVED CORRECTLY (0,125,252,377)
5349
5350 030546 104415 000001  DATACLK, 1     ;GET LAST BIT IN RECEIVER
5351 030552 012703 000004  MOV #4,R3      ;R3=CHARACTER COUNT
5352 030556 012702 034116  MOV #MESDAT,R2 ;LOAD MESSAGE POINTER IN R2
5353 030562 004737 033154  40$: JSR PC,INRDY ;WAIT FOR INRDY
5354 030566 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5355 030570 021204          021204
5356 030572 016104 000004  MOV 4(R1),R4   ;PUT 'FOUND' IN R4
5357 030576 112205          MOVB (R2)+,R5  ;PUT 'EXPECTED' IN R5
5358 030600 120504          CMPB R5,R4     ;IS RECEIVED DATA CORRECT?
5359 030602 001401          BEQ 41$       ;BR IF YES
5360 030604 104010          HLT 10        ;RECEIVE DATA ERROR
5361 030606 005303          41$: DEC R3     ;DEC CHARACTER COUNT
5362 030610 001364          BNE 40$       ;BR IF NOT DONE YET
5363
5364
5365          ;CHECK TO SEE THAT IN BCC MATCH IS SET
5366          ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5367
5368 030612 004737 033154  ROMCLK JSR PC,INRDY ;WAIT FOR INRDY
5369 030616 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5370 030620 021204          021204        ;GET FIRST HALF OF CRC
5371 030622 116137 000004 001252  MOVB 4(R1),TEMP3 ;PUT IN TEMP3
5372 030630 042737 177400 001252  BIC #177400,TEMP3 ;CLEAR HI BYTE
5373 030636 004737 033154  ROMCLK JSR PC,INRDY ;WAIT FOR INRDY
5374 030642 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5375 030644 021244          021244
5376 030646 016104 000004  MOV 4(R1),R4   ;PUT 'FOUND' IN R4
5377 030652 042704 000374  BIC #374,R4    ;CLEAR UNWANTED BITS
5378 030656 012705 000003  MOV #3,R5      ;PUT 'EXPECTED' IN R5
5379 030662 120504          CMPB R5,R4     ;ARE IN BCC MATCH AND BLOCK END SET?
5380 030664 001401          BEQ 50$       ;IN BCC MATCH ERROR
5381 030666 104042          HLT 42
5382 030670          50$:
5383 030670 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5384 030672 021204          021204        ;GET LAST HALF
5385 030674 116137 000004 001251  MOVB 4(R1),TEMP2+1 ;PUT IN TEMP2
5386 030702 042737 000377 001250  BIC #377,TEMP2  ;CLEAR LO BYTE
5387 030710 053737 001250 001252  BIS TEMP2,TEMP3 ;16 BIT BCC NOW IN TEMP3

```



5388 030716 023737 033334 001252  
 5389 030724 001401  
 5390 030726 104027  
 5391  
 5392  
 5393  
 5394  
 5395 030730 012703 000004  
 5396 030734 012702 034116  
 5397 030740 004737 033154  
 5398 030744 104414  
 5399 030746 021204  
 5400 030750 016104 000004  
 5401 030754 112205  
 5402 030756 120504  
 5403 030760 001401  
 5404 030762 104010  
 5405 030764 005303  
 5406 030766 001364  
 5407  
 5408  
 5409  
 5410  
 5411  
 5412 030770 004737 033154  
 5413 030774 104414  
 5414 030776 021204  
 5415 031000 116137 000004 001252  
 5416 031006 042737 177400 001252  
 5417 031014 004737 033154  
 5418 031020 104414  
 5419 031022 021244  
 5420 031024 016104 000004  
 5421 031030 042704 000374  
 5422 031034 012705 000003  
 5423 031040 120504  
 5424 031042 001401  
 5425 031044 104042  
 5426 031046  
 5427 031046 104414  
 5428 031050 021204  
 5429 031052 116137 000004 001251  
 5430 031060 042737 000377 001250  
 5431 031066 053737 001250 001252  
 5432 031074 023737 033334 001252  
 5433 031102 001401  
 5434 031104 104027  
 5435 031106 104400  
 5436  
 5437  
 5438  
 5439  
 5440  
 5441  
 5442  
 5443

CMP CALBCC,TEMP3 ;IS IT CORRECT?  
 BEQ 42\$ ;BR IF OK  
 HLT 27  
  
 ;CHECK TO SEE THAT SECOND FOUR CHARACTER MESSAGE  
 ;WAS RECEIVED CORRECTLY (0,125,252,377)  
 42\$: MOV #4,R3 ;R3=CHARACTER COUNT  
 MOV #MESDAT,R2 ;LOAD MESSAGE POINTER IN R2  
 43\$: JSR PC,INRDY ;WAIT FOR INRDY  
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304  
 021204  
 MOV 4(R1),R4 ;PUT "FOUND" IN R4  
 MOVB (R2)+,R5 ;PUT "EXPECTED" IN R5  
 CMPB R5,R4 ;IS RECEIVED DATA CORRECT?  
 BEQ 44\$ ;BR IF YES  
 HLT 10 ;RECEIVE DATA ERROR  
 44\$: DEC R3 ;DEC CHARACTER COUNT  
 BNE 43\$ ;BR IF NOT DONE YET  
  
 ;CHECK TO SEE THAT IN BCC MATCH IS SET  
 ;AND THAT THE BCC WAS RECEIVED CORRECTLY  
 JSR PC,INRDY ;WAIT FOR INRDY  
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304  
 021204 ;GET FIRST HALF OF CRC  
 MOVB 4(R1),TEMP3 ;PUT IN TEMP3  
 BIC #177400,TEMP3 ;CLEAR HI BYTE  
 JSR PC,INRDY ;WAIT FOR INRDY  
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304  
 021244  
 MOV 4(R1),R4 ;PUT "FOUND" IN R4  
 BIC #374,R4 ;CLEAR UNWANTED BITS  
 MOV #3,R5 ;PUT "EXPECTED" IN R5  
 CMPB R5,R4 ;ARE IN BCC MATCH AND BLOCK END SET?  
 BEQ 51\$  
 HLT 42 ;IN BCC MATCH ERROR  
 51\$: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304  
 021204 ;GET LAST HALF  
 MOVB 4(R1),TEMP2+1 ;PUT IN TEMP2  
 BIC #377,TEMP2 ;CLEAR LO BYTE  
 BIS TEMP2,TEMP3 ;16 BIT BCC NOW IN TEMP3  
 CMP CALBCC,TEMP3 ;IS IT CORRECT?  
 BEQ 5\$ ;BR IF OK  
 HLT 27  
 5\$: SCOPE ;SCOPE THIS TEST  
  
 ;\*\*\*\*\* TEST 57 \*\*\*\*\*  
 ;\*EMPTY SILO TEST  
 ;\*LOAD SILO WITH 2 SYNCs, 4 CHAR MESSAGE, SINGLE CLOCK  
 ;\*UNTIL THE SILO IS EMPTY, LOAD 4 MORE CHARACTERS IN THE  
 ;\*SILO. GIVE MORE TICKS, AND VERIFY THAT ONLY THE FIRST  
 ;\*4 CHARACTERS AND A BLOCK END WERE RECEIVED, AND IN ACTIVE IS CLEAR



```

5444
5445
5446
5447
5448 031110 012737 000057 001226 TST57: MOV #57,TSTNO
5449 031116 012737 031330 001216 MOV #TST60,NEXT
5450
5451 031124 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
5452 031126 005061 000004 CLR 4(R1) ;MASTER CLEAR DMC11
5453 031132 104414 ROMCLK ;CLEAR PORT4
5454 031134 122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5455 031136 004737 033676 JSR PC,CLRIO ;PUT LU IN BITSTUFF MODE
5456 031142 012711 004000 MOV #BIT11,(R1) ;DO THIS AFTER MODE IS SET
5457 031146 012702 034116 MOV #MESDAT,R2 ;SET LINE UNIT LOOP
5458 031152 012700 000003 MOV #3,R0 ;R2 POINTS TO MESSAGE
5459 031156 004737 033500 JSR PC,SYNLD ;R0 = CHAR COUNT
5460 031162 004737 032500 JSR PC,OUTRDY ;LOAD SILO WITH TWO FLAGS
5461 031166 004537 033634 JSR R5,MESLD ;WAIT FOR OUTRDY
5462 031172 034116 MESDAT ;LOAD MESSAGE IN SILO
5463 031174 000004 4 ;START OF MESSAGE
5464 031176 004737 032346 JSR PC,OCOR ;CHARACTER COUNT
5465 031202 104415 000065 DATACLK, 65 ;WAIT FOR OCOR
5466 031206 004537 033634 JSR R5,MESLD ;CLOCK DATA (EMPTY SILO)
5467 031212 034116 MESDAT ;PUT MORE CHARACTERS IN SILO
5468 031214 000004 4
5469 031216 004737 032346 JSR PC,OCOR
5470 031222 104415 000006 DATACLK, 6
5471 031226 104414 ROMCLK ;CLOCK UNTIL RTS IS CLEARED
5472 031230 021264 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5473 031232 032761 000040 000004 BIT #BITS,4(R1) ;GET RTS
5474 031240 001401 BEQ 5$ ;IS IT CLEAR?
5475 031242 104034 HLT 34 ;BR IF YES
5476 031244 104415 000041 5$: DATACLK, 41 ;ERROR, RTS NOT CLEAR
5477 031250 004737 033154 JSR PC,INRDY ;CLOCK XMITTER SOME MORE
5478 031254 104414 ROMCLK ;OK LETS CHECK WHAT WAS RECEIVED
5479 031256 021204 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5480 031260 016104 000004 MOV 4(R1),R4 ;GET RECEIVE DATA
5481 031264 112205 MOVB (R2)+,R5 ;PUT IT IN R4
5482 031266 120504 CMPB R5,R4 ;R5 = 'EXPECTED'
5483 031270 001401 BEQ 2$ ;IS DATA CORRECT?
5484 031272 104010 HLT 10 ;BR IF OK
5485 031274 005300 2$: DEC R0 ;DATA ERROR
5486 031276 001364 BNE 1$ ;DEC CHAR COUNT
5487 031300 004737 033154 JSR PC,INRDY ;BR IF NOT DONE YET
5488 031304 104414 ROMCLK ;WAIT FOR INRDY
5489 031306 021244 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5490 031310 016104 000004 MCV 4(R1),R4 ;READ LU-12
5491 031314 012705 000022 MOV #22,R5 ;PUT 'FOUND' IN R4
5492 031320 120504 CMPB R5,R4 ;PUT 'EXPECTED' IN R5
5493
5494 031322 001401 BEQ 6$ ;ARE BLOCK END AND IN RDY SET?
5495 031324 104032 HLT 32 ;AND IN ACTIVE AND IN BCC MATCH CLEAR?
5496
5497
5498 031326 6$:
5499 031326 104400 SCOPE ;SCOPE THIS TEST
    
```



5500  
 5501  
 5502  
 5503  
 5504  
 5505  
 5506  
 5507  
 5508  
 5509  
 5510  
 5511  
 5512  
 5513  
 5514  
 5515  
 5516  
 5517  
 5518  
 5519  
 5520  
 5521  
 5522  
 5523  
 5524  
 5525  
 5526  
 5527  
 5528  
 5529  
 5530  
 5531  
 5532  
 5533  
 5534  
 5535  
 5536  
 5537  
 5538  
 5539  
 5540  
 5541  
 5542  
 5543  
 5544  
 5545  
 5546  
 5547  
 5548  
 5549  
 5550  
 5551  
 5552  
 5553  
 5554  
 5555

031330 012737 000060 001226  
 031336 012737 031752 001216  
 031344 104412  
 031346 032737 040000 001366  
 031354 001575  
 031356 005061 000004  
 031362 104414  
 031364 122117  
 031366 004737 033676  
 031372 012711 004000  
 031376 004737 033500  
 031402 012737 102010 033332  
 031410 005037 031444  
 031414 005137 031444  
 031420 012703 000020  
 031424 012702 034122  
 031430 112237 031442  
 031434 004537 033210  
 031440 000010  
 031442 000000  
 031444 000000  
 031446 013737 033334 031444  
 031454 005303  
 031456 001364  
 031460 005137 033334  
 031464 004537 033634  
 031470 034122  
 031472 000020  
 031474 004737 033610  
 031500 004737 033610  
 031504 004537 033634  
 031510 034122  
 031512 000020  
 031514 004737 033610  
 031520 004737 033610  
 031524 004537 033634  
 031530 034122  
 031532 000020  
 031534 004737 033610  
 031540 004737 033610  
 031544 004737 032346

```

;***** TEST 60 *****
;*BITSTUFF CABLE DATA TEST
;*THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
;*2 FLAGS,16 CHAR,EOM,16 CHAR,EOM,16 CHAR,EOM
;*THE 16 CHARACTERS INCLUDE A FLOATING ONE AND ZERO
;*THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
;*RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
;*LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST
;:*****

: TEST 60
-----
TST60: MOV #60,TSTNO
MOV #TST61,NEXT

MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
BIT #BIT14,STAT1 ;SKIP TEST IF NO
BEQ 3$ ;LOOPBACK CONNECTOR ON
CLR 4(R1) ;CLEAR PORT4

ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122117 ;PUT LINE UNIT IN BITSTUFF MODE
JSR PC,CLRIO ;DO THIS AFTER MODE IS SET
MOV #BIT11,(R1) ;SET LINE UNIT LOOP
JSR PC,SYNLD ;LOAD TWO FLAGS
MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL FOR SOFT CRC CALC
CLR 6$ ;CLEAR OLD BCC
COM 6$ ;ADJUST TO -1 FOR SDLC
MOV #16.,R3 ;CHARACTER COUNT
MOV #FLTDAT,R2 ;R2= POINTER
7$: MOVB (R2)+,5$ ;LOAD CHAR FOR SOFT BCC CALC.
JSR R5,SIMBCC ;CALC SOFT BCC
10 ;SHIFT COUNT
5$: 0 ;CHARACTER
6$: 0 ;OLD BCC
MOV CALBCC,6$ ;LOAD OLD BCC
DEC R3 ;DEC COUNT
BNE 7$ ;BR IF NOT DONE YET
COM CALBCC ;ADJUST CALBCC FOR SDLC
JSR R5,MESLD ;LOAD SILO
FLTDAT ;MESSAGE ADDRESS
16. ;CHARACTER COUNT
JSR PC,EOM ;LOAD AN EOM
JSR PC,EOM
JSR R5,MESLD ;LOAD SILO
FLTDAT ;MESSAGE ADDRESS
16. ;CHARACTER COUNT
JSR PC,EOM ;LOAD AN EOM
JSR PC,EOM
JSR R5,MESLD ;LOAD SILO
FLTDAT ;MESSAGE ADDRESS
16. ;CHARACTER COUNT
JSR PC,EOM ;LOAD AN EOM
JSR PC,EOM
JSR PC,OCOR ;WAIT FOR OCOR
  
```



```

5556 031550 005011          CLR      (R1)          ;CLEAR LINE UNIT LOOP
5557 031552 012700 000003  MOV      #3,R0        ;RO = MESSAGE COUNT
5558 031556 012703 000020  MOV      #16.,R3      ;R3= CHARACTER COUNT
5559 031562 012702 034122  MOV      #FLTDAT,R2   ;LOAD MESSAGE POINTER IN R2
5560 031566 004737 033154  JSR      PC,INRDY     ;WAIT FOR INRDY
1$: ROMCLK                    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5561 031572 104414          MOV      021204        ;GET DATA FROM IN SILO
5562 031574 021204          MOV      4(R1),R4     ;PUT CHARACTER IN 'FOUND'
5563 031576 016104 000004  MOVB    (R2)+,R5     ;PUT 'EXPECTED' IN R5
5564 031602 112205          CMPB    R5,R4        ;IS RECEIVED DATA CORRECT
5565 031604 120504          BEQ     2$           ;BR IF OK
5566 031606 001401          HLT     25           ;DATA ERROR
5567 031610 104025
5568 031612
2$: DEC      R3          ;DEC CHARACTER COUNT
5569 031612 005303          BNE     1$           ;BR IF NOT DONE THIS MESSAGE
5570 031614 001364          MOV     #16.,R3     ;RESET CHARACTER COUNT
5571 031616 012703 000020
5572
5573
5574
5575 ;CHECK TO SEE THAT IN BCC MATCH IS SET
5576 ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5577 031622 004737 033154  JSR      PC,INRDY     ;WAIT FOR INRDY
5578 031626 104414          ROMCLK                    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5579 031630 021204          MOV     021204        ;GET FIRST HALF OF CRC
5580 031632 116137 000004 001252  MOVB    4(R1),TEMP3   ;PUT IN TEMP3
5581 031640 042737 177400 001252  BIC     #177400,TEMP3 ;CLEAR HI BYTE
5582 031646 004737 033154  JSR      PC,INRDY     ;WAIT FOR INRDY
5583 031652 104414          ROMCLK                    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5584 031654 021244          MOV     021244        ;PUT 'FOUND' IN R4
5585 031656 016104 000004  MOV     4(R1),R4     ;CLEAR UNWANTED BITS
5586 031662 042704 000374  BIC     #374,R4      ;PUT 'EXPECTED' IN R5
5587 031666 012705 000003  MOV     #3,R5        ;ARE IN BCC MATCH AND BLOCK END SET?
5588 031672 120504          CMPB    R5,R4
5589 031674 001401          BEQ     25$
5590 031676 104042          HLT     42           ;IN BCC MATCH ERROR
5591 031700
25$: ROMCLK                    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5592 031700 104414          MOV     021204        ;GET LAST HALF
5593 031702 021204          MOVB    4(R1),TEMP2+1 ;PUT IN TEMP2
5594 031704 116137 000004 001251  BIC     #377,TEMP2   ;CLEAR LO BYTE
5595 031712 042737 000377 001250  BIS     TEMP2,TEMP3  ;16 BIT BCC NOW IN TEMP3
5596 031720 053737 001250 001252  CMP     CALBCC,TEMP3 ;IS IT CORRECT?
5597 031726 023737 033334 001252  BEQ     4$           ;BR IF OK
5598 031734 001401          HLT     27
5599 031736 104027
4$: MOV     #FLTDAT,R2   ;RESET MESSAGE POINTER
5600 031740 012702 034122  DEC     RO           ;DECREMENT COUNTER
5601 031744 005300          BNE     1$           ;BR IF NOT DONE
5602 031746 001307
3$: SCOPE                    ;SCOPE THIS TEST
5603 031750 104400
5604
5605
5606
5607
5608
5609
5610
5611
;***** TEST 61 *****
;*BITSTUFF CABLE DATA TEST
;*THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
;*2 FLAGS,59 DATA CHARACTERS,EOM WITH GARBAGE CHARACTER
;*THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
;*RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
    
```



```

5612                                     ;*LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST
5613                                     ;:*****
5614                                     ;
5615                                     ; TEST 61
5616                                     ;-----
5617 031752 012737 000061 001226 TST61: MOV #61,TSTNO
5618 031760 012737 003364 001216 MOV #.EOP,NEXT
5619                                     ;R1 CONTAINS BASE DMC11 ADDRESS
5620 031766 104412 MSTCLR ;MASTER CLEAR DMC11
5621 031770 032737 040000 001366 BIT #BIT14,STAT1 ;SKIP TEST IF NO
5622 031776 001545 BEQ 3$ ;LOOPBACK CONNECTOR ON
5623 032000 005061 000004 CLR 4(R1) ;CLEAR PORT4
5624 032004 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5625 032006 122117 122117 ;PUT LINE UNIT IN BITSTUFF MODE
5626 032010 004737 033676 JSR PC,CLRIO ;DO THIS AFTER MODE IS SET
5627 032014 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
5628 032020 004737 033500 JSR PC,SYNLD ;LOAD TWO FLAGS
5629 032024 012737 102010 033332 MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL FOR SOFT CRC CALC
5630 032032 005037 032066 CLR 6$ ;CLEAR OLD BCC
5631 032036 005137 032066 COM 6$ ;ADJUST TO -1 FOR SDLC
5632 032042 012703 000073 MOV #59.,R3 ;CHARACTER COUNT
5633 032046 012702 034116 MOV #MESDAT,R2 ;R2= POINTER
5634 032052 112237 032064 7$: MOVB (R2)+,5$ ;LOAD CHAR FOR SOFT BCC CALC.
5635 032056 004537 033210 JSR R5,SIMBCC ;CALC SOFT BCC
5636 032062 000010 10 ;SHIFT COUNT
5637 032064 000000 5$: 0 ;CHARACTER
5638 032066 000000 6$: 0 ;OLD BCC
5639 032070 013737 033334 032066 MOV CALBCC,6$ ;LOAD OLD BCC
5640 032076 005303 DEC R3 ;DEC COUNT
5641 032100 001364 BNE 7$ ;BR IF NOT DONE YET
5642 032102 005137 033334 COM CALBCC ;ADJUST CALBCC FOR SDLC
5643 032106 004537 033634 JSR R5,MESLD ;LOAD SILO
5644 032112 034116 MESDAT ;MESSAGE ADDRESS
5645 032114 000073 59. ;CHARACTER COUNT
5646 032116 004737 033610 JSR PC,EOM ;LOAD AN EOM
5647 032122 004737 033610 JSR PC,EOM
5648 032126 004737 032346 JSR PC,OCOR ;WAIT FOR OCOR
5649 032132 005011 CLR (R1) ;CLEAR LINE UNIT LOOP
5650 032134 012700 000073 MOV #59.,R0 ;R0= CHARACTER COUNT
5651 032140 012702 034116 MOV #MESDAT,R2 ;LOAD MESSAGE POINTER IN R2
5652 032144 004737 033154 1$: JSR PC,INRDY ;WAIT FOR INRDY
5653 032150 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5654 032152 021204 021204 ;GET DATA FROM IN SILO
5655 032154 016104 000004 MOV 4(R1),R4 ;PUT CHARACTER IN "FOUND"
5656 032160 112205 MOVB (R2)+,R5 ;PUT "EXPECTED" IN R5
5657 032162 120504 CMPB R5,R4 ;IS RECEIVED DATA CORRECT
5658 032164 001401 BEQ 2$ ;BR IF OK
5659 032166 104025 HLT 25 ;DATA ERROR
5660 032170 2$:
5661 032170 005300 DEC R0 ;DECREMENT COUNTER
5662 032172 001364 BNE 1$ ;BR IF NOT DONE
5663
5664
5665 ;CHECK TO SEE THAT IN BCC MATCH IS SET
5666 ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5667
    
```



5668	032174	004737	033154			JSR	PC,INRDY		;WAIT FOR INRDY
5669	032200	104414				ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5670	032202	021204					021204		;GET FIRST HALF OF CRC
5671	032204	116137	000004	001252		MOVB	4(R1),TEMP3		;PUT IN TEMP3
5672	032212	042737	177400	001252		BIC	#177400,TEMP3		;CLEAR HI BYTE
5673	032220	004737	033154			JSR	PC,INRDY		;WAIT FOR INRDY
5674	032224	104414				ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5675	032226	021244					021244		
5676	032230	016104	000004			MOV	4(R1),R4		;PUT "FOUND" IN R4
5677	032234	042704	000374			BIC	#374,R4		;CLEAR UNWANTED BITS
5678	032240	012705	000003			MOV	#3,R5		;PUT "EXPECTED" IN R5
5679	032244	120504				CMPB	R5,R4		;ARE IN BCC MATCH AND BLOCK END SET?
5680	032246	001401				BEQ	25\$		
5681	032250	104042				HLT	42		;IN BCC MATCH ERROR
5682	032252					25\$:			
5683	032252	104414				ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5684	032254	021204					021204		;GET LAST HALF
5685	032256	116137	000004	001251		MOVB	4(R1),TEMP2+1		;PUT IN TEMP2
5686	032264	042737	000377	001250		BIC	#377,TEMP2		;CLEAR LO BYTE
5687	032272	053737	001250	001252		BIS	TEMP2,TEMP3		;16 BIT BCC NOW IN TEMP3
5688	032300	023737	033334	001252		CMP	CALBCC,TEMP3		;IS IT CORRECT?
5689	032306	001401				BEQ	3\$		;BR IF OK
5690	032310	104027				HLT	27		
5691	032312	104400				3\$:	SCOPE		;SCOPE THIS TEST
5692									
5693									
5694									
5695									
5696									
5697	032314								
5698									
5699									
5700									
5701	032314	104414				ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5702	032316	021364					021364		;PORT4_LU 17
5703	032320	017737	147066	032344		MOV	@DMP04,NITCH		;STORE LU 17
5704	032326	106137	032344			ROLB	NITCH		
5705	032332	106137	032344			ROLB	NITCH		
5706	032336	106137	032344			ROLB	NITCH		;PUT SI IN THE CARRY BIT
5707	032342	000207				RTS	PC		
5708	032344	000000				NITCH:	0		
5709									
5710									
5711	032346								
5712									
5713									
5714	032346	104414				ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5715	032350	021364					021364		;PORT4_LU 17
5716	032352	032777	000020	147032		BIT	#BIT4,@DMP04		;IS OCOR SET?
5717	032360	001772				BEQ	OCOR		;BR IF NO
5718	032362	000207				RTS	PC		;OK OCOR IS SET, GO BACK
5719									
5720									
5721	032364								
5722									
5723									

;SUBROUTINES

-----

GETSI:

;THIS SUBROUTINE READS LU 17, AND PUTS IT INTO NITCH.  
 ;NITCH IS ROTATED LEFT UNTILL THE SI BIT IS IN CARRY

ROMCLK

;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

```

MOV @DMP04,NITCH ;STORE LU 17
ROLB NITCH
ROLB NITCH
ROLB NITCH ;PUT SI IN THE CARRY BIT
RTS PC
    
```

NITCH:

0

OCOR:

;THIS SUBROUTINE SPINS ON OCOR

ROMCLK

;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

```

MOV @DMP04,NITCH ;STORE LU 17
BIT #BIT4,@DMP04 ;IS OCOR SET?
BEQ OCOR ;BR IF NO
RTS PC ;OK OCOR IS SET, GO BACK
    
```

SYNC:

;THIS SUBROUTINE LOADS THE SILO WITH THE NUMBER OF SYNC  
 ;CHARACTERS PASSED TO IT IN THE WORD AFTER THE JSR CALL





5780	032600	012761	000026	000004		MOV	#26,4(R1)	;LOAD PORT4
5781	032606	104414			ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5782	032610	122110				122110		;LOAD OUT DATA
5783	032612	005337	001246			DEC	TEMP1	;ALL DONE?
5784	032616	001361				BNE	1\$	;BR IF NOT
5785	032620	004737	032500			JSR	PC,OUTRDY	;WAIT FOR OUTRDY
5786	032624	013761	001250	000004		MOV	TEMP2,4(R1)	;LOAD PORT4
5787	032632	104414			ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5788	032634	122110				122110		;LOAD OUT DATA
5789	032636	004737	032346			JSR	PC,OCOR	;WAIT FOR OCOR
5790	032642	000207				RTS	PC	
5791								
5792								
5793	032644							
5794								
5795								
5796	032644	013637	001250			MOV	@(SP)+,TEMP2	;GET CHARACTER
5797	032650	062746	000002			ADD	#2,-(SP)	;ADJUST STACK
5798	032654	004737	032500			JSR	PC,OUTRDY	;WAIT FOR OUTRDY
5799	032660	013761	001250	000004		MOV	TEMP2,4(R1)	;LOAD PORT4
5800	032666	104414			ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5801	032670	122110				122110		;LOAD OUT DATA
5802	032672	004737	032500			JSR	PC,OUTRDY	;WAIT FOR OUTRDY
5803	032676	104414			ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5804	032700	122110				122110		;LOAD GARBAGE CHAR
5805	032702	004737	032346			JSR	PC,OCOR	;WAIT FOR OCOR
5806	032706	000207				RTS	PC	
5807								
5808								
5809	032710							
5810								
5811								
5812								
5813	032710	012737	000073	001250		MOV	#73,TEMP2	;LOAD COUNT
5814	032716	005737	033150			TST	SCHAR	;FIRST TIME HERE?
5815	032722	100470				BMI	4\$	;BR IF BITSTUFF
5816	032724	001032				BNE	2\$	;BR IF NO
5817	032726	062737	000002	001250		ADD	#2,TEMP2	;ADD 2 TO CHARACTER COUNT
5818	032734	012737	000003	001246		MOV	#3,TEMP1	;SET FOR 3 SYNC
5819	032742	012761	000026	000004		MOV	#26,4(R1)	;LOAD PORT4
5820	032750	104414			ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5821	032752	122114				122114		;LOAD SYNC REGISTER
5822	032754	004737	032500		1\$:	JSR	PC,OUTRDY	;WAIT FOR OUTRDY
5823	032760	012761	000001	000004		MOV	#1,4(R1)	;LOAD PORT4
5824	032766	104414			ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5825	032770	122111				122111		;SET SOM
5826	032772	012761	000026	000004		MOV	#26,4(R1)	;LOAD PORT4
5827	033000	104414			ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5828	033002	122110				122110		;LOAD OUT DATA
5829	033004	005337	001246			DEC	TEMP1	;ALL DONE?
5830	033010	001361				BNE	1\$	;BR IF NOT
5831	033012	004737	032500		2\$:	JSR	PC,OUTRDY	;WAIT FOR OUTRDY
5832	033016	013761	033150	000004		MOV	SCHAR,4(R1)	;LOAD PORT4
5833	033024	104414			ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5834	033026	122110				122110		;LOAD OUT DATA
5835	033030	005737	033152			TST	STUFLG	;BITSTUFF???

CHARSD: ;THIS SUBROUTINE LOADS THE SILO WITH THE CHARACTER PASSED TO IT.

SILOLD: ;THIS SUBROUTINE FILLS THE OUT SILO  
 ; WITH A BINARY COUNT PATTERN



```

5836 033034 001407          BEQ      6$          ;BR IF NO
5837 033036 013737 033150 033050  MOV      SCHAR,5$   ;IT IS SLDL SO CHECK BITSTUFFING
5838 033044 004537 033716          JSR      R5,STFFCL  ;ADD ANY BIT STUFF CLOCK TICKS
5839 033050 000000          5$:      0          ;CHARACTER
5840 033052 000010          10       ;SHIFT COUNT
5841 033054 005237 033150          6$:      INC      SCHAR   ;NEXT CHARACTER
5842 033060 022737 000400 033150  CMP      #400,SCHAR ;ALL DONE?
5843 033066 001403          BEQ      3$
5844 033070 005337 001250          DEC      TEMP2      ;DECREMENT COUNT
5845 033074 001346          BNE      2$          ;BR IF NOT DONE
5846 033076 004737 032346          3$:      JSR      PC,OCOR   ;WAIT FOR OCOR
5847 033102 000207          RTS      PC
5848 033104 005037 033150          4$:      CLR      SCHAR   ;START PATTERN AT ZERO
5849 033110 012737 177777 033152  MOV      #-1,STUFLG ;SET BITSTUFF FLAG
5850 033116 005037 034114          CLR      BITCON     ;CLEAR STUFF COUNT
5851 033122 062737 000002 001250  ADD      #2,TEMP2   ;ADD 2 TO CHARACTER COUNT
5852 033130 012761 000001 000004  MOV      #1,4(R1)  ;SET BIT0 IN PORT4
5853 033136 104414          ROMCLK   122111     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5854 033140 122111          122111     ;SET SOM!
5855 033142 104414          ROMCLK   122110     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5856 033144 122110          122110     ;LOAD GARBAGE CHAR
5857 033146 000721          BR       2$          ;GO LOAD SILU
5858 033150 000000          SCHAR:   0
5859 033152 000000          STUFLG:  0

5860
5861
5862 033154          INRDY:
5863          ;THIS SUBROUTINE SPINS ON INRDY
5864          ;IF INRDY FAILS TO SET THE DELAY TIMES OUT AND AN
5865          ;ERROR IS REPORTED. FOR BETTER SCOPE LOOPS THIS
5866          ;DELAY CAN BE MADE SHORTER BY ALTERING THE NUMBER
5867          ;INITIALLY LOADED INTO TEMP1, THE SMALLER THE NUMBER
5868          ;THE SHORTER THE DELAY. 0 IS THE LONGEST DELAY.
5869
5870 033154 012737 000000 001246  MOV      #0,TEMP1   ;SET UP DELAY COUNTER
5871 033162
5872 033162 104414          1$:      ROMCLK   021244     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5873 033164 021244          021244     ;PORT4 LU12
5874 033166 032777 000020 146216  BIT      #BIT4,@DMP04 ;IS INRDY SET?
5875 033174 001004          BNE      2$          ;BR IF YES
5876 033176 005237 001246          INC      TEMP1     ;INC DELAY
5877 033202 001367          BNE      1$          ;TRY AGAIN
5878 033204 104037          HLT      37         ;ERROR,NO INRDY
5879 033206 000207          2$:      RTS      PC   ;RETURN

5880
5881
5882 033210          SIMBCC:
5883          ;THIS SUBROUTINE CALCULATES THE CRC USING POLYNOMIAL GIVEN
5884          ;IN XPOLY. THE CORRECT CRC IS RETURNED IN CALBCC, AND THE
5885          ;STATE OF THE LSB OF THE BCC IS RETURNED IN THE C BIT.
5886
5887 033210 010046          MOV      R0,-(SP)   ;SAVE R0 ON STACK
5888 033212 012537 001246          MOV      (R5)+,TEMP1 ;TEMP1 = SHIFT COUNT
5889 033216 012537 001250          MOV      (R5)+,TEMP2 ;TEMP2 = CHARACTER
5890 033222 012537 033334          MOV      (R5)+,CALBCC ;CALBCC = OLD BCC
5891 033226 013700 033334          1$:      MOV      CALBCC,R0 ;PUT OLD BCC IN R0

```

5892	033232	000241		CLC		
5893	033234	006037	033334	ROR	CALBCC	;SHIFT OLD BCC
5894	033240	006037	001250	ROR	TEMP2	;SHIFT CHARACTER
5895	033244	005500		ADC	RO	;ADD CHAR CARRY TO OLD BCC
5896	033246	006000		ROR	RO	;PUT BITO TO CARRY BIT
5897	033250	103011		BCC	2\$	;CARRY IS FEEDBACK BIT
5898	033252	013700	033332	MOV	XPOLY,RO	;IF FEEDBACK = 1
5899	033256	043700	033334	BIC	CALBCC,RO	;EXCLUSIVLY OR XPOLY TO CALBCC
5900	033262	043737	033332 033334	BIC	XPOLY,CALBCC	
5901	033270	050037	033334	BIS	RO,CALBCC	
5902	033274	005337	001246	DEC	TEMP1	;DEC SHIFT COUNT
5903	033300	001352		BNE	1\$	;BR IF NOT DONE
5904	033302	012737	000001 001246	MOV	#1,TEMP1	;GET SET TO INVERT BITO
5905	033310	013700	033334	MOV	CALBCC,RO	;PUT RESULT IN RO
5906	033314	006000		ROR	RO	;SHIFT BITO TO CARRY
5907	033316	005537	001246	ADC	TEMP1	;INVERT CARRY TO BITO OF TEMP1
5908	033322	006037	001246	ROR	TEMP1	;PUT INVERTED BIT IN CARRY
5909	033326	012600		MOV	(SP)+,RO	;RESTORE RO
5910	033330	000205		RTS	R5	;RETURN
5911	033332	000000				
5912	033334	000000				
5913		000200				
5914		120001				
5915		102010				
5916						
5917						
5918	033336			BCCLD:		
5919						;THIS SUBROUTINE LOADS THE OUT SILO WITH 2 SYNC
5920						;WITH SOM SET, AND ONE CHARACTER PASSED TO IT
5921						;WITH THE SOM BIT CLEAR (ENABLE CRC)
5922						
5923	033336	013637	001250	MOV	@(SP)+,TEMP2	;GET CHARACTER
5924	033342	062746	000002	ADD	#2,-(SP)	;ADJUST STACK
5925	033346	012737	000002 001246	MOV	#2,TEMP1	;SET FOR 2 SYNC
5926	033354	012761	000026 000004	MOV	#26,4(R1)	;LOAD PORT4
5927	033362	104414		ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5928	033364	122114			122114	;LOAD SYNC REGISTER
5929	033366	004737	032500	1\$: JSR	PC,OUTRDY	;WAIT FOR OUTRDY
5930	033372	012761	000001 000004	MOV	#1,4(R1)	;LOAD PORT4
5931	033400	104414		ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5932	033402	122111			122111	;SET SOM
5933	033404	012761	000026 000004	MOV	#26,4(R1)	;LOAD PORT4
5934	033412	104414		ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5935	033414	122110			122110	;LOAD OUT DATA
5936	033416	005337	001246	DEC	TEMP1	;ALL DONE?
5937	033422	001361		BNE	1\$	;BR IF NOT
5938	033424	004737	032500	JSR	PC,OUTRDY	;WAIT FOR OUTRDY
5939	033430	013761	001250 000004	MOV	TEMP2,4(R1)	;LOAD PORT4
5940	033436	104414		ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5941	033440	122110			122110	;LOAD OUT DATA
5942	033442	004737	032346	JSR	PC,OCOR	;WAIT FOR OCOR
5943	033446	000207		RTS	PC	
5944						
5945						
5946	033450			GETQO:		
5947						;THIS SUBROUTINE READS THE STATE OF THE TRANSMIT

XPOLY: 0  
 CALBCC: 0  
 LRC8=200  
 CRC16=120001  
 CRC.CCITT=102010



```

5948 ;BCC LSB AND PUTS IT IN THE CARRY BIT
5949
5950 033450 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5951 033452 021364 021364 ;PORT4 LU-17
5952 033454 106177 145732 ROLB @DMP04 ;PUT Q0 IN CARRY
5953 033460 000207 RTS PC ;RETURN
5954
5955
5956 033462 GETQI:
5957 ;THIS SUBROUTINE READS THE STATE OF THE RECEIVE
5958 ;BCC LSB AND PUTS IT IN THE CARRY BIT
5959
5960 033462 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5961 033464 021364 021364 ;PORT4 LU-17
5962 033466 106177 145720 ROLB @DMP04 ;PUT Q0 IN CARRY
5963 033472 106177 145714 ROLB @DMP04 ;PUT Q1 IN CARRY
5964 033476 000207 RTS PC ;RETURN
5965
5966
5967 033500 SYNLD:
5968 ;THIS SUBROUTINE LOADS OUT SILO WITH
5969 ;2 SYNC CHARACTERS WITH SOM SET
5970
5971 033500 012737 000002 001246 MOV #2,TEMP1 ;LOAD COUNTER FOR 2 SYNCs
5972 033506 012761 000026 000004 MOV #26,4(R1) ;PORT4_26
5973 033514 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5974 033516 122114 122114 ;LOAD SYNC REG
5975 033520 004737 032500 1$: JSR PC,OUTRDY ;WAIT FOR OUTRDY
5976 033524 012761 000001 000004 MOV #1,4(R1) ;LOAD PORT4
5977 033532 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5978 033534 122111 122111 ;SET SOM
5979 033536 012761 000026 000004 MOV #26,4(R1) ;PORT_26
5980 033544 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5981 033546 122110 122110 ;LOAD OUT DATA WITH SYNC
5982 033550 005337 001246 DEC TEMP1 ;DECREMENT COUNTER
5983 033554 001361 BNE 1$ ;BR IF NOT DONE
5984 033556 000207 RTS PC ;RETURN
5985
5986
5987 033560 SOM:
5988 ;THIS SUBROUTINE LOADS SOM AND OUT DATA WITH A
5989 ;GARBAGE CHARACTER (0)
5990
5991 033560 004737 032500 JSR PC,OUTRDY ;WAIT FOR OUTRDY
5992 033564 012761 000001 000004 MOV #1,4(R1) ;PORT4_1
5993 033572 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5994 033574 122111 122111 ;SET SOM
5995 033576 005061 000004 CLR 4(R1) ;CLEAR DATA CHAR
5996 033602 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5997 033604 122110 122110 ;LOAD GARBAGE CHARACTER
5998 033606 000207 RTS PC ;RETURN
5999
6000
6001 033610 EOM:
6002 ;THIS SUBROUTINE LOADS EOM AND OUT DATA WITH A
6003 ;GARBAGE CHARACTER (2) TO ENABLE TRANSMISSION OF BCC

```

6004									
6005	033610	004737	032500						
6006	033614	012761	000002	000004					
6007	033622	104414			ROMCLK				
6008	033624	122111							
6009	033626	104414			ROMCLK				
6010	033630	122110							
6011	033632	000207			RTS	PC			
6012									
6013									
6014	033634				MESLD:				
6015									
6016									
6017									
6018									
6019	033634	010046							
6020	033636	012500							
6021	033640	012537	001246						
6022	033644	004737	032500		1\$:				
6023	033650	112061	000004						
6024	033654	104414			ROMCLK				
6025	033656	122110							
6026	033660	005337	001246						
6027	033664	001367							
6028	033666	004737	032346						
6029	033672	012600							
6030	033674	000205							
6031									
6032									
6033	033676				CLR10:				
6034									
6035									
6036									
6037	033676	012761	000200	000004					
6038	033704	104414			ROMCLK				
6039	033706	122112							
6040	033710	104414			ROMCLK				
6041	033712	122111							
6042	033714	000207			RTS	PC			
6043									
6044									
6045	033716				STFFCL:				
6046									
6047									
6048									
6049	033716	010046							
6050	033720	012500							
6051	033722	012537	001252						
6052	033726	106000			1\$:				
6053	033730	103403							
6054	033732	005037	034114						
6055	033736	000412							
6056	033740	005237	034114		2\$:				
6057	033744	022737	000005	034114					
6058	033752	001004							
6059	033754	005037	034114						





6116 034142 100 140 160  
 6117 034145 170 003 300  
 6118 034150 174 176 177  
 6119 034153 001  
 6120 034154 363 347 317  
 6121 034157 200 000 377  
 6122 034162 377 377 200  
 6123 034165 037

STUFDI: .BYTE 100,140,160,170,3,300,174,176,177,1

.BYTE 363,347,317,200,0,377,377,377,200,37

6124 034166 046377 047111 020105  
 6125 034224 046377 047111 020105  
 034267 377 044514 042516  
 034333 377 044514 042516  
 034375 377 051124 047101  
 034425 377 042522 042503  
 034444 051377 041505 044505  
 034471 377 047515 042504  
 034514 052377 044510 020123  
 034574 040777 052125 051517  
 034656 054777 052517 044040  
 034710 052377 040522 051516  
 034737 377 042522 042503  
 034763 377 047111 041040  
 035023 377 051124 047101  
 035073 377 040503 046102  
 035114 043377 040514 020107  
 035130 052377 040522 051516  
 035174 051777 044527 041524  
 035215 377 041101 051117  
 035232 052377 040522 051516  
 035255 377 040510 043114  
 035277 377 052517 020124  
 035322 044777 020116 042522

.EVEN

EM1: .ASCIZ <377>/LINE UNIT INITIALIZATION TEST/  
 EM2: .ASCIZ <377>^LINE UNIT REGISTER READ/ONLY TEST^  
 EM3: .ASCIZ <377>^LINE UNIT REGISTER WRITE/READ TEST^  
 EM4: .ASCIZ <377>/LINE UNIT INTERNAL CLOCK FAILURE/  
 EM5: .ASCIZ <377>/TRANSMITTER DATA ERROR/  
 EM6: .ASCIZ <377>/RECEIVER TEST/  
 EM7: .ASCIZ <377>/RECEIVER DATA ERROR/  
 EM10: .ASCII <377>/MODEM SIGNAL ERROR/  
 .ASCII <377>/THIS ERROR COULD BE CAUSED IF YOU HAVE V.35 AND/  
 .ASCII <377>/AUTOSIZED. YOU MUST MANUALLY ANSWER QUESTIONS IF/  
 .ASCIZ <377>/YOU HAVE V.35 (DMC11-FA)/  
 EM11: .ASCIZ <377>/TRANSMITTER CRC ERROR/  
 EM12: .ASCIZ <377>/RECEIVER CRC ERROR/  
 EM13: .ASCIZ <377>/IN BCC MATCH ERROR (LU REG 12)/  
 EM14: .ASCIZ <377>/TRANSMITTER FAILED TO GO TO MARK STATE/  
 EM15: .ASCIZ <377>/CABLE DATA TEST/  
 EM16: .ASCIZ <377>/FLAG ERROR/  
 EM17: .ASCIZ <377>/TRANSMITTER FAILED TO STUFF A ZERO/  
 EM20: .ASCIZ <377>/SWITCH PAC TEST/  
 EM21: .ASCIZ <377>/ABORT ERROR/  
 EM22: .ASCIZ <377>/TRANSMITTER ERROR/  
 EM23: .ASCIZ <377>/HALF DUPLEX TEST/  
 EM24: .ASCIZ <377>/OUT READY NOT SET/  
 EM25: .ASCIZ <377>/IN READY NOT SET/

035344 042777 050130 041505  
 035365 377 054105 042520  
 035423 377 044103 051101  
 035461 377 047503 051122  
 035521 377 054105 042520  
 035553 377 054105 042520  
 035621 377 046102 041517  
 035644 051377 051524 042040  
 035670

DH1: .ASCIZ <377>/EXPECTED FOUND/  
 DH2: .ASCIZ <377>/EXPECTED FOUND LU-REGISTER/  
 DH3: .ASCIZ <377>/CHARACTER BIT THAT FAILED/  
 DH4: .ASCIZ <377>/CORRECT CRC BIT THAT FAILED/  
 DH5: .ASCIZ <377>/EXPECTED FOUND SHIFT/  
 DH6: .ASCIZ <377>/EXPECTED FOUND CHARACTER SHIFT/  
 DH7: .ASCIZ <377>/BLOCK END NOT SET/  
 DH10: .ASCIZ <377>/RTS DID NOT CLEAR/

.EVEN

035670 000002  
 035672 003 007  
 035674 001272  
 035676 003 002  
 035700 001270  
 035702 000003  
 035704 003 007  
 035706 001272  
 035710 003 010  
 035712 001270  
 035714 003 002  
 035716 001264

DT1: 2  
 .BYTE 3,7  
 SAVR5  
 .BYTE 3,2  
 SAVR4  
 DT2: 3  
 .BYTE 3,7  
 SAVR5  
 .BYTE 3,10  
 SAVR4  
 .BYTE 3,2  
 SAVR2



035720	000002		DT3:	2	
035722	003	017		.BYTE	3,17
035724	001272			SAVR5	
035726	002	002		.BYTE	2,2
035730	001266			SAVR3	
035732	000002		DT4:	2	
035734	006	021		.BYTE	6,21
035736	033334			CALBCC	
035740	002	002		.BYTE	2,2
035742	001266			SAVR3	
035744	000003		DT5:	3	
035746	001	011		.BYTE	1,11
035750	001300			ZERO	
035752	001	011		.BYTE	1,11
035754	001302			ONE	
035756	002	002		.BYTE	2,2
035760	001260			SAVR0	
035762	000003		DT6:	3	
035764	001	011		.BYTE	1,11
035766	001302			ONE	
035770	001	011		.BYTE	1,11
035772	001300			ZERO	
035774	002	002		.BYTE	2,2
035776	001260			SAVR0	
036000	000004		DT7:	4	
036002	001	011		.BYTE	1,11
036004	001300			ZERO	
036006	001	011		.BYTE	1,11
036010	001302			ONE	
036012	003	007		.BYTE	3,7
036014	001272			SAVR5	
036016	002	001		.BYTE	2,1
036020	001266			SAVR3	
036022	000004		DT10:	4	
036024	001	011		.BYTE	1,11
036026	001302			ONE	
036030	001	011		.BYTE	1,11
036032	001300			ZERO	
036034	003	007		.BYTE	3,7
036036	001272			SAVR5	
036040	002	001		.BYTE	2,1
036042	001266			SAVR3	
036044	000002		DT11:	2	
036046	003	007		.BYTE	3,7
036050	034112			FLAG	
036052	002	002		.BYTE	2,2
036054	001266			SAVR3	
036056	000002		DT12:	2	
036060	006	004		.BYTE	6,4
036062	033334			CALBCC	
036064	006	002		.BYTE	6,2
036066	001252			TEMP3	
036070			.ERRTAB:		
036070	000000			0	
036072	000000			0	



036074	000000		0	
036076	034166	EM1		
036100	035365	DH2	;HLT	1
036102	035702	DT2		
036104	034224	EM2		
036106	035365	DH2	;HLT	2
036110	035702	DT2		
036112	034267	EM3		
036114	035365	DH2	;HLT	3
036116	035702	DT2		
036120	034333	EM4		
036122	000000	0	;HLT	4
036124	000000	0		
036126	034375	EM5		
036130	035365	DH2	;HLT	5
036132	035702	DT2		
036134	034375	EM5		
036136	035423	DH3	;HLT	6
036140	035720	DT3		
036142	034425	EM6		
036144	035344	DH1	;HLT	7
036146	035670	DT1		
036150	034444	EM7		
036152	035344	DH1	;HLT	10
036154	035670	DT1		
036156	034471	EM10		
036160	035344	DH1	;HLT	11
036162	035670	DT1		
036164	034710	EM11		
036166	035521	DH5	;HLT	12
036170	035744	DT5		
036172	034737	EM12		
036174	035521	DH5	;HLT	13
036176	035744	DT5		
036200	034710	EM11		
036202	035461	DH4	;HLT	14
036204	035732	DT4		
036206	034763	EM13		
036210	000000	0	;HLT	15
036212	000000	0		
036214	034710	EM11		
036216	035521	DH5	;HLT	16
036220	035762	DT6		
036222	034737	EM12		
036224	035521	DH5	;HLT	17
036226	035762	DT6		
036230	034710	EM11		
036232	035553	DH6	;HLT	20
036234	036000	DT7		
036236	034710	EM11		
036240	035553	DH6	;HLT	21
036242	036022	DT10		
036244	034737	EM12		
036246	035553	DH6	;HLT	22
036250	036000	DT7		
036252	034737	EM12		



036254	035553	DH6	;HLT	23
036256	036022	DT10		
036260	035023	EM14		
036262	000000	0	;HLT	24
036264	000000	0		
036266	035073	EM15		
036270	035344	DH1	;HLT	25
036272	035670	DT1		
036274	035114	EM16		
036276	035423	DH3	;HLT	26
036300	036044	DT11		
036302	034737	EM12		
036304	035344	DH1	;HLT	27
036306	036056	DT12		
036310	035130	EM17		
036312	000000	0	;HLT	30
036314	000000	0		
036316	035174	EM20		
036320	035344	DH1	;HLT	31
036322	035670	DT1		
036324	035215	EM21		
036326	035621	DH7	;HLT	32
036330	000000	0		
036332	035215	EM21		
036334	035423	DH3	;HLT	33
036336	035720	DT3		
036340	035232	EM22		
036342	035644	DH10	;HLT	34
036344	000000	0		
036346	035255	EM23		
036350	035365	DH2	;HLT	35
036352	035702	DT2		
036354	035277	EM24		
036356	000000	0	;HLT	36
036360	000000	0		
036362	035322	EM25		
036364	000000	0	;HLT	37
036366	000000	0		
036370	034425	EM6		
036372	035365	DH2	;HLT	40
036374	035702	DT2		
036376	034375	EM5		
036400	035521	DH5	;HLT	41
036402	035744	DT5		
036404	034763	EM13		
036406	035344	DH1	;HLT	42
036410	035670	DT1		

036412 000001

CORMAX:  
.END

































TST60	031330	5449	5514#											
TST61	031752	5515	5617#	5692										
TST62	= ***** U	5618												
TST7	013324	2448	2489#											
TTST	003612	1318*	1319*	1321*	1322*	1385#								
TWOSYN=	010000	697#												
TYPDAT	005206	1656	1674	1677#										
TYPE =	104402	820#	1114	1126	1138	1223	1228	1236	1239	1271	1276	1317	1326	1339
		1340	1342	1344	1346	1434	1447	1464	1557	1597	1657	1658	1661	1662
		1664	1666	1670	1675	1724	1830	1833	1885	1931	1949	1955	1993	2015
		2029	2032	2039	2042	2049	2057	2066	2073	2080	2088	2100	2218	
TYPMSG	005106	1654	1657#											
VEC	006527	1790#	2007											
VECMAP	012216	2217	2229#											
WHICH	012210	1995	2225#											
WRDCNT	004710	1565*	1598*	1606#										
WRKO.F	005174	1669	1672#											
XBX	005000	1631	1633	1635#										
XCSR	003546	1341	1366#											
XERR	003570	1347	1375#											
XHEAD	006225	1138	1790#											
XLOC	003022	1196*	1214*	1217	1248	1261#								
XPASS	003562	1345	1372#											
XPOLY	033332	4082*	4114*	4171*	4209*	4266*	4298*	4355*	4387*	4442*	4525*	4632*	4840*	4921*
		5157*	5256*	5526*	5629*	5898	5900	5911#						
XSTATQ	007602	1147	1790#											
XTSTN	005330	1663	1703#											
XVEC	003554	1343	1369#											
ZERO	001300	787#	6125											
\$COD =	***** U	602												
\$CRAP =	177777	1#	2293#	2296	2299#	2317#	2320	2323#	2340#	2343	2346#	2364#	2367	2370#
		2395#	2398	2401#	2437#	2440	2443#	2479#	2340#	2343	2346#	2364#	2367	2370#
		2592	2595#	2611#	2614	2617#	2633#	2636	2639#	2666#	2669	2673#	2703#	2706
		2710#	2751#	2754	2758#	2813#	2816	2822#	2888#	2891	2897#	2963#	2966	2972#
		3037#	3040	3043#	3114#	3117	3123#	3194#	3197	3203#	3283#	3286	3291#	3358#
		3361	3365#	3418#	3421	3424#	3450#	3453	3456#	3482#	3485	3488#	3522#	3525
		3528#	3554#	3557	3561#	3612#	3615	3618#	3658#	3661	3664#	3704#	3707	3710#
		3750#	3753	3756#	3796#	3799	3802#	3841#	3844	3849#	3891#	3894	3899#	3936#
		3939	3943#	4002#	4005	4008#	4055#	4058	4064#	4144#	4147	4153#	4239#	4242
		4248#	4328#	4331	4337#	4417#	4420	4423#	4500#	4503	4506#	4586#	4589	4594#
		4723#	4726	4730#	4786#	4789	4797#	5101#	5104	5113#	5436#	5439	5444#	5500#
		5503	5510#	5604#	5607	5613#								
		724	1112	1358#	1682									
SENDAD	003522	1#	2293	2299	2301	2306#	2317	2323	2325	2330#	2340	2346	2348	2353
\$N =	000061	2354#	2364	2370	2372	2377	2378#	2395	2401	2403	2409	2410#	2437	2443
		2445	2451	2452#	2479	2485	2487	2493	2494#	2537	2543	2545	2551	2552#
		2589	2595	2597	2602	2603#	2611	2617	2619	2624	2625#	2633	2639	2641
		2646	2647#	2666	2673	2675	2680	2681#	2703	2710	2712	2717	2718#	2751
		2758	2760	2765	2766#	2813	2822	2824	2829	2830#	2888	2897	2899	2904
		2905#	2963	2972	2974	2979	2980#	3037	3043	3045	3050	3051#	3114	3123
		3125	3130	3131#	3194	3203	3205	3210	3211#	3283	3291	3293	3298	3299#
		3358	3365	3367	3372	3373#	3418	3424	3426	3431	3432#	3450	3456	3458
		3463	3464#	3482	3488	3490	3495	3496#	3522	3528	3530	3535	3536#	3554
		3561	3563	3568	3569#	3612	3618	3620	3625	3626#	3658	3664	3666	3671
		3672#	3704	3710	3712	3717	3718#	3750	3756	3758	3763	3764#	3796	3802
		3804	3809	3810#	3841	3849	3851	3856	3857#	3891	3899	3901	3906	3907#







DMEND	1#	1328													
DMFRNT	1#	602													
HLT	676#	2315	2338	2362	2393	2421	2434	2463	2476	2509	2530	2564	2582	2609	2631
	2655	2664	2700	2738	2748	2790	2800	2809	2857	2861	2871	2875	2885	2932	2936
	2946	2950	2960	3007	3011	3021	3025	3035	3075	3079	3094	3099	3106	3159	3163
	3174	3178	3192	3243	3247	3261	3265	3328	3332	3344	3353	3397	3408	3412	3448
	3480	3520	3552	3591	3603	3610	3648	3656	3694	3702	3740	3748	3786	3794	3831
	3881	3925	3934	3978	4000	4051	4100	4104	4132	4136	4174	4198	4227	4231	4284
	4288	4316	4320	4373	4377	4405	4409	4471	4475	4555	4559	4644	4648	4671	4675
	4694	4698	4709	4718	4761	4781	4852	4856	4879	4883	4902	4906	4933	4937	4960
	4964	4983	4987	4998	5007	5025	5046	5055	5069	5090	5099	5169	5173	5196	5200
	5219	5223	5237	5241	5268	5272	5295	5299	5318	5322	5333	5342	5360	5381	5390
	5404	5425	5434	5475	5484	5495	5567	5590	5599	5659	5681	5690	5762	5878	6086
\$ABORT	599#	3891													
\$AUTO	1#	1150													
\$BCC	601#	4055	4144	4239	4328										
\$BINCR	601#	4417	4500												
\$BINWI	599#	3194													
\$BUFFE	1#	1802													
\$CDATA	602#	5500	5604												
\$CLOCK	601#	2633													
\$COMP	1#	599#	3515	3586	3598	3605	3643	3652	3689	3698	3735	3744	3781	3790	3921
	4776	5041	5085	5376	5420	5585	5676								
\$CRC	601#	4080	4112	4169	4207	4264	4296	4353	4385						
\$CRCSH	601#	4436	4519												
\$CYCLE	1#	1874													
\$EMPTY	602#	5436													
\$EOP	1#	1328													
\$FINI	1#	5692													
\$FLAG	599#	2851	2926	3001	3069	3153	3237	3322	3402	4688	4896	4977	5213	5231	5312
\$FLOAT	599#	2495	2515	2553	2570										
\$GETPA	1#														
\$HALF	602#														
\$HEADE	1#	602													
\$INACT	599#	3418	3450	3522											
\$INIT	599#														
\$LINE1	599#	2479	2537												
\$LU1	599#	2293	2317	2340	2364										
\$LU12	599#	2395													
\$LU17	599#	2437													
\$MARHI	1#														
\$MARK	599#														
\$MATCH	601#	4764	5028	5072	5363	5407	5572	5663							
\$MOCK	1#														
\$MODEM	601#	3936													
\$MSG	1#	1790													
\$MULT	602#	3283													
\$PATTE	599#	3796	3841												
\$PFAIL	1#	1706													
\$QOOI	601#	5946	5956												
\$QUEST	1#	1984	1997	2006	2110	2119									
\$RAMCL	1#	1734													
\$RCLK	1#	1737	1740	1777	1782	2308	2331	2355	2379	2412	2414	2425	2427	2454	2456
	2467	2469	2499	2501	2520	2522	2556	2558	2574	2576	2603	2625	2649	2658	2682
	2687	2689	2693	2719	2724	2726	2731	2741	2767	2772	2774	2779	2783	2793	2803
	2831	2838	2840	2845	2881	2906	2913	2915	2920	2956	2981	2988	2990	2995	3031



	3052	3058	3060	3132	3139	3141	3147	3188	3212	3222	3224	3228	3233	3277	3300
	3308	3310	3316	3374	3380	3382	3385	3387	3389	3433	3441	3465	3473	3497	3504
	3506	3509	3513	3537	3545	3570	3576	3578	3584	3594	3596	3627	3633	3635	3641
	3650	3673	3679	3681	3687	3696	3719	3725	3727	3733	3742	3765	3771	3773	3779
	3788	3811	3825	3858	3869	3875	3908	3915	3919	3927	3957	3960	3980	3983	4022
	4026	4033	4074	4163	4258	4347	4432	4445	4449	4453	4494	4515	4528	4532	4536
	4580	4603	4739	4755	4769	4774	4783	4806	5019	5034	5039	5048	5063	5078	5083
	5092	5122	5354	5369	5374	5383	5398	5413	5418	5427	5453	5471	5478	5488	5521
	5561	5578	5583	5592	5624	5653	5669	5674	5683	5701	5714	5729	5733	5736	5742
	5745	5756	5774	5778	5781	5787	5800	5803	5820	5824	5827	5833	5853	5855	5872
	5927	5931	5934	5940	5950	5960	5973	5977	5980	5993	5996	6007	6009	6024	6038
	6040														
\$RCRC	602#	4723													
\$REC	599#	3554	3612	3658	3704	3750									
\$SCOPE	1#	1378													
\$SIMBC	1#	5882													
\$SINAC	599#	3482													
\$SOFTC	1#	1810													
\$STUFF	602#	3037													
\$SWPAC	602#	2589	2611												
\$TCHAR	601#	4610	4742	4813	5129										
\$TCRC	601#	4586	4786	5101											
\$TRANW	601#	4631	4839	4920	5156	5255									
\$TRAN1	599#	2666	2703	2751											
\$TRPDE	1#	816	818	820	822	824	826	828	830	832	834	836	838	840	842
	844														
\$TSTN	1#	2301	2325	2348	2372	2403	2445	2487	2545	2597	2619	2641	2675	2712	2760
	2824	2899	2974	3045	3125	3205	3293	3367	3426	3458	3490	3530	3563	3620	3666
	3712	3758	3804	3851	3901	3945	4010	4066	4155	4250	4339	4425	4508	4596	4732
	4799	5115	5446	5512	5615										
\$VARIA	1#	735													
\$WINDO	599#	2813	2888	2963	3114										
\$XZ	1#	2293	2299	2317	2323	2340	2346	2364	2370	2395	2401	2437	2443	2479	2485
	2537	2543	2589	2595	2611	2617	2633	2639	2666	2673	2703	2710	2751	2758	2813
	2822	2888	2897	2963	2972	3037	3043	3114	3123	3194	3203	3283	3291	3358	3365
	3418	3424	3450	3456	3482	3488	3522	3528	3554	3561	3612	3618	3658	3664	3704
	3710	3750	3756	3796	3802	3841	3849	3891	3899	3936	3943	4002	4008	4055	4064
	4144	4153	4239	4248	4328	4337	4417	4423	4500	4506	4586	4594	4723	4730	4786
	4797	5101	5113	5436	5444	5500	5510	5604	5613						
\$ZEROS	602#	3358													

. ABS. 036412 000

ERRORS DETECTED: 0

CZDMF,CZDMF/NL:TOC/SOL/CRF\_CZDMF.MAC,CZDMF.P11/EQ:LUTYPE  
 RUN-TIME: 15 22 1 SECONDS  
 RUN-TIME RATIO: 48/39=1.2  
 CORE USED: 33K (65 PAGES)