

DUP11

BASIC DUP11 OFFLINE SDLC
MD-11-DZDPB-B

EP-DZDPB-B-DL-A

AUG 1977

COPYRIGHT © 75-77

digital

FICHE 1 OF 1

MADE IN USA

This image shows a microfiche card with a grid of frames. Each frame contains a small, high-contrast image of a document page, likely containing technical specifications or data. The frames are arranged in approximately 12 columns and 15 rows. The overall appearance is that of a standard microfiche card used for data storage and retrieval.

B01

EOF1DZDMHBSEQ

00010000

770712

PDP10 411

IHDR1DZDPBBSEQ

00010000

770712

45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96

1.0 ABSTRACT

THE FUNCTION OF THE DU11 DIAGNOSTICS IS TO VERIFY THAT THE OPTION OPERATES ACCORDING TO SPECIFICATIONS. THE DIAGNOSTICS VERIFY THAT THERE ARE NO MALFUNCTIONS AND THAT ALL OPERATIONS OF THE DU11 ARE CORRECT IN ITS ENVIRONMENT. PARAMETERS MAY BE SET TO ALERT DIAGNOSTICS AS TO THE DU11 CONFIGURATION BY ANSWERING THE PARAMETER DIALOG (LOAD ADDRESS=200, START ADDRESS=1). ALL QUESTIONS SHOULD BE ANSWERED AND THEN EACH DIAGNOSTIC WILL "OVERLAY" THESE PARAMETERS WHICH ARE STORED IN THE "STATUS TABLE" (SEE SECTION 8.4). THE ALTERNATIVE TO THE PARAMETER DIALOG IS DEFAULT PARAMETERS (SEE SECTION 8.5).

THE DIAGNOSTICS WILL RUN UP TO EIGHT CONSECUTIVELY ADDRESSED AND CONSECUTIVELY VECTORED DU11'S IN A CHAIN MODE, I.E., RUNNING THE DIAGNOSTIC COMPLETELY FOR ONE DEVICE BEFORE STARTING THE NEXT.

DZDPB DOES READ/WRITE CHECKING, BUS ADDRESS CHECKING, DEVICE AND BUS RESET TESTS ON THE CONTROL AND STATUS REGISTERS. TESTS ARE MADE TO PROVE THERE IS NO INTERACTION WITHIN AND BETWEEN REGISTERS. THE REGISTERS ARE CHECKED BOTH A BIT AT A TIME AND ALL AT ONCE.

IN ADDITION, THE TRANSMITTER SDLC FUNCTIONS ARE CHECKED IN MAINTENANCE INTERNAL MODE, I.E., CLOCKING OF THE DEVICE IS DONE BY THE PROGRAM. THE DEVICE IS SET UP, A SPECIFIC NUMBER OF HALF-CLOCKS ARE DONE, AND A TEST IS MADE FOR A SIGNIFICANT EVENT.

IN CHECKING DATA, THE SOFTWARE EMULATES THE HARDWARE AND USES THE PROCESSOR CARRY BIT AFTER A ROTATE TO DETERMINE WHAT THE OUTPUT SHOULD BE AT THE TRANSMITTER BIT WINDOW A BIT AT A TIME. THE PROGRAM THEN COMPARES THE EMULATED SOFTWARE BIT TO THE HARDWARE BIT OUTPUT. THE PROCESS IS REPEATED UNTIL ALL DATA IS CHECKED.

THE TRANSMITTER BCC IS CHECKED USING THE CRC.CCITT POLYNOMIAL IN THE SAME WAY AS DATA, WITH ONE EXCEPTION--THE BCC IS CALCULATED FIRST BY THE PROGRAM AND THEN COMPARED TO THE OUTPUT OF THE TRANSMITER.

CURRENTLY THERE ARE THREE OFF-LINE DIAGNOSTICS THAT ARE TO BE RUN IN SEQUENCE TO ENSURE THAT IF AN ERROR SHOULD OCCUR IT WILL BE DETECTED AT AN EARLY STAGE AND ESTABLISH THAT DIAGNOSIS OF THE ERROR WILL BE IMMEDIATE TO DISCOVERING THE

97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116

PROBLEM.

NOTE: ADDITIONAL DIAGNOSTICS MAY BE ADDED IN THE FUTURE.

THE THREE DIAGNOSTICS ARE:

1. DZDPB [REV] BASIC AND OFFLINE TRANSMITTER TESTS
2. DZDPC [REV] OFFLINE RECEIVER AND MODEM CONTROL AND INTERRUPT TESTS
3. DZDPD [REV] OFFLINE SDLC AND DECMODE DATA AND FUNCTION TESTS

NOTE: THERE IS A FOURTH MAINDEC, TAPE DZDPE [REV] WHICH IS A QUICK-VERIFY TAPE THAT REQUIRES ANSWERING A DIALCG. ITS FUNCTION IS TO ENABLE THE OPERATOR TO QUICKLY DETERMINE IF THERE IS A PROBLEM WITH THE DEVICE. SEE THE DOCUMENTATION IN THAT LISTING FOR MORE INFORMATION.

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

2.0 REQUIREMENTS

2.1 EQUIPMENT

ANY PDP11 FAMILY CPU (WITH MINIMUM 8K MEMORY)
ASR 33 (OR EQUIVALENT)
DUP11

2.2 STORAGE

PROGRAM WILL USE ALL 8K OF MEMORY EXCEPT WHERE ABS AND
BOOTSTRAP LOADER RESIDE. LOCATION 1500 THRU 1560 ARE
ESPECIALLY TO BE NOTED AND LEFT UNTOUCHED BY THE OPERATOR
AFTER THE DUP11 PARAMETER DIALOG HAS BEEN EXECUTED OR AFTER
THE DEFAULT SETUP HAS BEEN DONE.

3.0 LOADING PROCEDURE

3.1 METHOD

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

ABSOLUTE LOADER STARTING ADDRESS = **500

MEMORY	SIZE
	(*)=
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)

166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221

4.0 STARTING PROCEEDURE

- A. SET SWITCH REGISTER TO 000200
- B. DEPRESS 'LOAD ADDRESS' KEY AND RELEASE
- C. SET SWR TO ZERO FOR DEFAULT PARAMETERS ESTABLISHED IN THE TAPE (SEE SECTION 8.5.3 FOR FULL EXPLANATION OF DEFAULT PARAMETERS) OR LEAVE SWR BIT 7=1 TO USE EXISTING PARAMETERS PREVIOUSLY SET UP BY THE DUP11 PARAMETER DIALOG OR A PREVIOUSLY RUN DUP11 DIAGNOSTIC. SET SWR=1 TO GO THROUGH THE PARAMETER DIALOG. (IT IS NOT NECESSARY TO INPUT NEW PARAMETERS FOR EACH TAPE.) (SECTION 7.2, 8.4 AND 8.5 MAY BE HELPFUL)
- 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:

'EXAMPLE'

'MAP OF DUP11 STATUS'

1500	160050	CSR OF FIRST DUP11
1502	000300	VECTOR OF FIRST DUP11
1504	140026	STATUS AND SYNC FOR FIRST DUP11
1506	160060	CSR OF SECOND DUP11
1510	000310	VECTOR OF SECOND DUP11
1512	140026	STATUS AND SYNC FOR SECOND DUP11

THE ABOVE IS ONLY AN EXAMPLE! THIS WOULD INDICATE THE STATUS TABLE STARTING AT ADDRESS 1500 IN THE PROGRAM. THE STATUS TABLE MUST BE VERIFIED BY THE USER. FOR INFORMATION ON THE STATUS TABLE SEE SECTION 8.4 FOR HELP.

IT IS POSSIBLE FOR THE OPERATOR TO MANUALLY CHANGE (TOGGLE IN) THE INFORMATION IN THE MAP TO SUIT A SPECIFIC CONFIGURATION OF DEVICES, BUT THE RESPONSIBILITY FOR VERIFYING THAT INFORMATION RESTS WITH THE OPERATOR.

THE PROGRAM WILL TYPE 'R' AND PROCEED TO RUN THE DIAGNOSTIC

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/BELL 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.

H01

DZDPBB MACY11 27(1006) 18-MAY-77 00:00 PAGE 7
DZDPBB.P11 13-MAY-77 15:27

SEQ 0006

222
223

SW 06 SET: RESERVED
SW 05 SET: RESERVED

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
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279

SW 04 SET: RESERVED
 SW 03 SET: SELECT DUP11'S DESIRED ACTIVE
 SW 02 SET: LOCK ON SELECTED TEST
 SW 01 SET: RESTART PROGRAM AT SELECTED TEST
 SW 00 SET: ENTER PARAMETERS USING MANUAL DIALOG

SWITCHES 8 THROUGH 15 ARE DYNAMIC AND SHOULD BE USED AS NEEDED IN THE DIAGNOSTIC. SWITCHES 0 THROUGH 3 ARE STATIC (ONLY ARE OPERABLE WHEN THE MONITOR PORTION OF THE TAPE IS RUNNING) AND SHOULD BE SET UP PRIOR TO STARTING OR RESTARTING THE DIAGNOSTIC.

4.1.2 SWITCH REGISTER RESTRICTIONS

SW 03 RESELECT DUP11'S DESIRED ACTIVE. PLEASE NOTE THAT A MESSAGE IS TYPED OUT FOR SETTING THE SWITCH REGISTER EQUAL TO DUP11'S ACTIVE. THIS MEANS IF THE SYSTEM HAS THREE DUP11S BITS 00, 01, 02 WILL BE SET IN LOC 'DUPACTV' FROM THE SWITCH REGISTER. USING THIS SWITCH(SW03) ALTERS THAT LOCATION. THEREFORE, IF THREE DUP11S ARE IN THE SYSTEM ***DO NOT*** SET SWITCHES GREATER THAN SW 02 IN THE UP POSITION. THIS WOULD BE A FATAL ERROR. DO NOT SELECT MORE ACTIVE DUP11S THAN HAS BEEN GIVEN INFORMATION ABOUT IN THE PARAMETER PROGRAM.

AS EXPLAINED IN SECTION 1.0, DEVICES SHOULD BE CONSECUTIVELY ADDRESSED, AND CAN BE SELECTED OR DESELECTED USING THIS SWITCH.

- METHOD:
- A. LOAD ADDRESS 200
 - B. START WITH SW 03=1
 - C. PROGRAM WILL TYPE MESSAGE
 - D. SET THE BINARY NUMBER OF DUP11S DESIRED ACTIVE. EXAMPLE: 1=1 DUP11; 3=2 DUP11; 7=3 DUP11; 17=4 DUP11 37=5 DUP11 ETC. PRESS CONTINUE.
 - E. NUMBER (IF VALID) WILL BE IN DATA LIGHTS (EXCLUDING 11/05)
 - F. SET WITH ANY OTHER SWITCH SETTINGS DESIRED. PRESS CONTINUE.

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 THAT IS NOT IN THE ORDER OF SEQUENCE. THE REASON FOR THIS IS THAT THE PROGRAM HAS TO CLEAR AREAS AND SET UP PARAMETERS IN THE MONITOR PORTION OF THE PROGRAM. IT IS POSSIBLE TO LD200, AND RAISE SW01, THEN START, PROVIDED PARAMETERS HAVE BEEN PREVIOUSLY SET UP AS DESCRIBED IN SECTION 4.0. ALSO, WHEN A TEST IS SELECTED, ALWAYS START AT THE VERY BEGINNING OF THAT TEST.

J01

DZDPBB MACY11 27(1006) 18-MAY-77 00:00 PAGE 9
DZDPBB.P11 13-MAY-77 15:27

SEQ 0008

280
281

CW 09 LOOP ON CURRENT DATA. THIS SWITCH WILL ONLY WORK IF
CALL 'SCOPI' IS IN THAT TEST. THE REASON IS THAT MOST

282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328

TESTS DEAL WITH BLOCKS OF DIFFERENT DATA TO BE SENT OR RECEIVED ALL AT ONCE, THUS KNOWN AS BLOCK DATA--ONE PATTERN CAN'T BE SINGLED OUT. (SEE SECTION 4.1.3.B.1)

4.1.3 SWITCH REGISTER PRIORITIES

A) 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).

B) SCOPE SWITCHES

1. SW 09 - (IF ENABLED BY 'SCOPI') ON AN ERROR. IF AN ASTERISK '*' IS PRINTED IN FRONT OF THE TEST NUMBER (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 ERROR) SW08 IS BEST. (SW14=0, SW10=0, SW09=0, SW08=1).

FOR INTERMITTENT ERRORS, SW14=1 WILL LOOP ON TEST REGARDLESS OF ERROR OR NO ERROR. (SW14=1, SW10=0, SW09=0, SW08=1,0)

2. SW 14 - LOOP ON TEST. WILL LOOP ON TEST UNTIL SWITCH IS LOWERED.
3. SW 11 - INHIBIT ITERATIONS (QUICK PASS). ALLOWS ONLY ONE PASS THROUGH A TEST.

4.2 STARTING ADDRESS

STARTING ADDRESS IS AT 000200. THERE ARE NO OTHER STARTING ADDRESSES FOR THE DUP11 DIAGNOSTICS.

NOTE: IF ADDRESS 000042 IS NON-ZERO THE PROGRAM ASSUMES IT IS UNDER ACT11 OR XXDP CONTROL AND WILL ACT ACCORDINGLY. AFTER *ALL* AVAILABLE DUP11'S ARE TESTED THE PROGRAM WILL RETURN TO 'XXDP' OR 'ACT-11'.

329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381

5.0 OPERATING PROCEDURE

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

5.1 PROGRAM AND/OR OPERATOR ACTION

THE TYPICAL APPROACH SHOULD BE

1. HALT ON ERROR (VIA SW 15=1) WHENEVER 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. IN THIS WAY THE EXACT FUNCTIONING OF THE TEST CAN BE INTERPRETED SINCE THE ERROR PC IS THE HLT+2 LOCATION.

IN SOME TESTS, THERE IS A SUBROUTINE CALL THROUGH A REGISTER (E.G., JSR R1, FLAG). THE SUBROUTINE DOES THE DATA CHECKING FOR THE TEST AND WILL REPORT AN ERROR IF ONE OCCURS. THIS MEANS THAT THE FAILING TEST COULD BE IN ONE PART OF THE LISTING WHILE THE SUBROUTINE THAT FOUND THE ERROR IS IN ANOTHER PART. TO DETERMINE THE PC OF THE FAILING TEST, CHECK THE REGISTER USED BY THE SUBROUTINE. IT WILL CONTAIN THE RETURN ADDRESS OF THE FAILING TEST.

6.0 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 TO THE ERROR MESSAGE WHICH IS TO GIVE THE OPERATOR AN INDICATION OF THE ERROR.

6.1 ERROR RECOVERY

IF FOR SOME REASON THE DUP11 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' FOR THE NUMBER OF THE TEST THAT WAS RUNNING AT THE TIME OF THE CATASTROPHIC ERROR. THIS GIVES THE OPERATOR SOME IDEA AS TO WHAT THE DUP11 WAS DOING AT THE TIME OF THE ERROR.

382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421

7.0 RESTRICTIONS

7.1 STARTING RESTRICTIONS

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

7.2 OPERATING RESTRICTIONS

DUP11 "PARAMETER DIALOG" MUST BE RUN ONLY ONCE PRIOR TO THE FIRST RUNNING OF ANY DUP11 DIAGNOSTIC IF "DEFAULT PARAMETERS" ARE NOT USED. IF ONLY DUP11 DIAGNOSTICS WERE LOADED AFTER DUP11 PARAMETER SETUP, AND IF CORE MEMORY HAS NOT BEEN CHANGED, I.E. USE OF DIAGNOSTICS OTHER THAN DUP11 DIAGNOSTICS, AND IF THERE WERE NO DUP11 CONFIGURATION CHANGES, THE DUP11 PARAMETER SETUP NEED NEVER BE RUN AGAIN. HOWEVER, IF ANY OF THE ABOVE HAVE BEEN VIOLATED THE DUP11 PARAMETER SETUP MUST BE RUN AGAIN BEFORE RUNNING THE DIAGNOSTICS. UNDER NORMAL OPERATING CONDITIONS IT SHOULD NOT BE NECESSARY TO INPUT NEW PARAMETERS TO SUBSEQUENT DIAGNOSTICS, UNLESS A CHANGE IS REQUIRED.

NOTE: AN ALTERNATIVE TO THE ABOVE IS ATTEMPTING THE DEFAULT PARAMETERS WHEN THE PROGRAM IS INITIALLY STARTED WITH SWR=0.

7.3 HARDWARE CONFIGURATION RESTRICTIONS FOR THE PURPOSE OF RUNNING MULTIPLE DUP11'S IN CHAIN MODE.

1. CSR ADDRESSES MUST BE CONSECUTIVE.
2. VECTORS ARE CONSECUTIVE IF PARAMETER PROGRAM IS USED.
3. ALL JUMPERS ARE ASSUMED TO BE AS SETUP IN PARAMETER DIALOG.
4. PRIORITY LEVEL MUST BE THE SAME FOR ALL DEVICES.

422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477

8.0 MISCELLANEOUS

8.1 EXECUTION TIME

ALL DUP11 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.

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 DUP11'S IN SYSTEM ARE TESTED. WHEN THE DIAGNOSTIC HAS COMPLETED A PASS WITH THE NORMAL ITERATION COUNT (ICOUNT=50), THE FOLLOWING IS AN EXAMPLE OF THE PRINT OUT TO BE EXPECTED.

END PASS DZDPBB CSR:160050 VEC:300 PASSES:000001 ERRORS:000000

NOTE: THE NUMBERS FOR CSR AND VEC ARE NOT NECESSARILY THE VALUES FOR THE DEVICE. THEY ARE ONLY FOR THIS EXAMPLE.

8.3 KEY LOCATIONS

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

NEXT CONTAINS THE ADDRESS OF THE NEXT TEST TO BE PERFORMED.

TSTNO CONTAINS THE NUMBER OF THE TEST NOW BEING PERFORMED.

RUN THE BIT IN 'RUN' ALWAYS POINTS ONE PAST THE DUP11 CURRENTLY BEING TESTED. EXAMPLE: (RUN) /0000000001000000 MEANS THAT DUP11 NO.05 IS THE DUP11 NOW RUNNING.

DUPCR00-DUPCR07 (1500)-(1560) THESE LOCATIONS CONTAIN THE INFORMATION NEEDED TO TEST UP TO 8 (DECIMAL) DUP11S SEQUENTIALY. THEY CONTAIN THE CSR, VECTOR AND STATUS CONCERNING THE CONFIGURATION OF EACH DUP11.

DUPACTV EACH BIT SET IN THIS LOCATION INDICATES THAT THE ASSOCIATED DUP11 WILL BE TESTED IN TURN. EXAMPLE: (DUPACTV) /0000000000011111 MEANS

B02

DZDPBB MACY11 27(1006) 18-MAY-77 00:00 PAGE 14
DZDPBB.P11 13-MAY-77 15:27

SEQ 0013

478
479

THAT DUP11 NO. 00,01,02,03,04 WILL BE TESTED.
EXAMPLE: (DUPACTV) /0000000000010001 MEANS

480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535

THAT DUP11 NO. 00,04 WILL BE TESTED.

RXCSR CONTAINS THE RECEIVER CSR OF THE CURRENT DUP11 UNDER TEST.

8.4 MORE ON THAT 'STATUS TABLE' (1500-1560)

'MAP OF DUP11 STATUS'

1500	160050
1502	000300
1504	140000

THE ABOVE INFORMATION WILL BE REPEATED FOR EACH OF UP TO 8 DUP11'S IN THE SYSTEM (THESE WILL FOLLOW UNDER THIS TABLE).
EXPLANATION:

1500 160050 THIS IS THE SYSTEM CONTROL REGISTER FOR THE 1ST DUP11 IN THE SYSTEM.

1502 000300 THIS IS VECTOR 'A' FOR THE FIRST DUP11 IN THE SYSTEM.

1504 140026 THIS REPRESENTS SYNC AND SOFTWARE STATUS FOR THE FIRST DUP.

THE BITS ARE AS FOLLOWS:

- BIT 15 SET: OPTIONAL CLEAR JUMPER IN
- BIT 14 SET: TURNAROUND CONNECTOR ON
- BIT 13 SET:
- BIT 12 SET:
- BIT 11 SET:
- BIT 10 SET:
- BIT 09 SET:
- BIT 08 SET:
- BIT 07-00 SYNC CHARACTER FOR DECMODE TESTS.

THE ABOVE IS REPEATED FOR EACH DUP11 IN THE SYSTEM. THE TABLE IS FILLED BY DEFAULT PARAMETERS OR BY THE MANUAL PARAMETER INPUT AS DESCRIBED PREVIOUSLY. ALSO, IF DESIRED BY THE USER - THE LOCATIONS MAY BE ALTERED BY HAND (TOGGLED IN) TO SUIT THE SPECIFIC CONFIGURATION, THUS MAKING EACH DEVICE MAP DIFFERENT. IT IS THE RESPONSIBILITY OF THE OPERATOR TO VERIFY THE DATA IN THE MAP.

8.5 METHOD OF DEVELOPING DEFAULT PARAMETERS

8.5.1 DEFAULT PARAMETER ASSUMPTIONS

TOO MUCH HARDWARE WOULD HAVE TO BE ANALYZED TO SIZE THE THE PARAMETERS. THE PROGRAM MUST ASSUME THE VARIATIONS. THE

002

DZDP88 MACY11 27(1006) 18-MAY-77 00:00 PAGE 16
DZDP88.P11 13-MAY-77 15:27

SEQ 0015

536
537

RESULT, IF NOT TO YOUR SPECIFIC CONFIGURATION, MAY BE ALTERED
BY HAND (TOGGLE IN) AS DESIRED. IN THIS WAY 95% OF THE

538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564

PARAMETER SETUP WAS DONE BY THE PROGRAM AND 5% BY YOU.
THEREFORE:

1) ALL JUMPERS ARE ASSUMED TO BE IN THE FOLLOWING
CONFIGURATION.

	IN	OUT
W1=SECONDARY REC ENABLE	X	
W2=SEC REC DISABLE		X
W3=CLEAR OPTION	X	
W4=SEC TX ENABLE	X	
W5=DSC A CONTROL		X
W6=A+B DS CONTROL	X	
W7=BUS GRANT CONTROL	X	

2) THE H325 TURN AROUND CONNECTOR IS ASSUMED TO BE ON.

3) THE MANUFACTURING OPTION CSR OF 160050 AND VECTOR OF 770
ARE USED.

4) THE BR LEVEL IS ASSUMED TO BE 5.

IN ALL ADJUSTMENTS PLEASE REFER TO SECTION 8.4 FOR GREATER
DETAIL.

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
599
600
601
602
603
604
605
606
607
608
609

DOCUMENT

COPYRIGHT 1977
DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASS. 01754

610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659

1 MAINDEC-11-DZDPB-B /(<377>)/BASIC DUP11 AND OFFLINE SDLC TRANSMITT
COPYRIGHT(C) 1975,1977, DIGITAL EQUIPMENT CORP., MAYNARD, MASS.

1197 ***** TEST 1 *****
VERIFY THAT ADDRESSING DEVICE DOES *NOT* CAUSE
A TIME-OUT TRAP.
1227 ***** TEST 2 *****
PRIMARY REGISTER ADDRESSING TEST
LOAD EACH PRIMARY REGISTER WITH A
DIFFERENT NUMBER AND VERIFY EACH
WAS INDIVIDUALLY ADDRESSED
1282 ***** TEST 3 *****
RECEIVER CONTROL REGISTER RESET TEST. TEST THAT AFTER
RECEIVER CONTROL REGISTER IS WRITTEN AND A BUS RESET IS
DONE THAT RECEIVER CONTROL REGISTER IS CLEARED.
1307 ***** TEST 4 *****
RECEIVER BUFFER REGISTER RESET TEST. TEST THAT AFTER A BUS
1309 RESET IS DONE THAT RECEIVER BUFFER REGISTER IS CLEARED.
1330 ***** TEST 5 *****
PARAMETER STATUS REGISTER RESET TEST. TEST THAT AFTER
PARAMETER STATUS REGISTER IS WRITTEN AND A BUS RESET IS
DONE THAT PARAMETER STATUS REGISTER IS CLEARED.
1355 ***** TEST 6 *****
TRANSMITTER CONTROL REGISTER RESET TEST. TEST THAT AFTER
TRANSMITTER CONTROL REGISTER IS WRITTEN AND A BUS RESET IS
DONE THAT TRANSMITTER CONTROL REGISTER IS CLEARED.
1381 ***** TEST 7 *****
TRANSMITTER BUFFER REGISTER RESET TEST. TEST THAT AFTER
TRANSMITTER BUFFER REGISTER IS WRITTEN AND A BUS RESET IS
DONE THAT TRANSMITTER BUFFER REGISTER IS CLEARED.
1406 ***** TEST 10 *****
TRANSMITTER CONTROL REGISTER READ/WRITE BIT TEST.
SET BIT3, VERIFY BIT3 WAS SET.
CLEAR BIT3, VERIFY BIT3 WAS CLEARED.
1438 ***** TEST 11 *****
TRANSMITTER CONTROL REGISTER READ/WRITE BIT TEST.
SET BIT4, VERIFY BIT4 WAS SET.
CLEAR BIT4, VERIFY BIT4 WAS CLEARED.

660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710

1470 ***** TEST 12 *****
TRANSMITTER CONTROL REGISTER READ/WRITE BIT TEST.
SET BIT10, VERIFY BIT10 WAS SET.
CLEAR BIT10, VERIFY BIT10 WAS CLEARED.

1502 ***** TEST 13 *****
TRANSMITTER CONTROL REGISTER READ/WRITE BIT TEST.
SET BIT11, VERIFY BIT11 WAS SET.
CLEAR BIT11, VERIFY BIT11 WAS CLEARED.

1534 ***** TEST 14 *****
TRANSMITTER CONTROL REGISTER READ/WRITE BIT TEST.
SET BIT12, VERIFY BIT12 WAS SET.
CLEAR BIT12, VERIFY BIT12 WAS CLEARED.

1566 ***** TEST 15 *****
TRANSMITTER CONTROL REGISTER READ/WRITE BIT TEST.
SET BIT13, VERIFY BIT13 WAS SET.
CLEAR BIT13, VERIFY BIT13 WAS CLEARED.

1598 ***** TEST 16 *****
TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
SET BIT0, VERIFY BIT0 WAS SET.
CLEAR BIT0, VERIFY BIT0 WAS CLEARED.

1628 ***** TEST 17 *****
TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
SET BIT1, VERIFY BIT1 WAS SET.
CLEAR BIT1, VERIFY BIT1 WAS CLEARED.

1658 ***** TEST 20 *****
TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
SET BIT2, VERIFY BIT2 WAS SET.
CLEAR BIT2, VERIFY BIT2 WAS CLEARED.

1688 ***** TEST 21 *****
TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
SET BIT3, VERIFY BIT3 WAS SET.
CLEAR BIT3, VERIFY BIT3 WAS CLEARED.

1718 ***** TEST 22 *****
TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
SET BIT4, VERIFY BIT4 WAS SET.
CLEAR BIT4, VERIFY BIT4 WAS CLEARED.

1748 ***** TEST 23 *****
TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
SET BITS, VERIFY BITS WAS SET.
CLEAR BITS, VERIFY BITS WAS CLEARED.

711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762

1778 ***** TEST 24 *****
TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
SET BIT6, VERIFY BIT6 WAS SET.
CLEAR BIT6, VERIFY BIT6 WAS CLEARED.

1808 ***** TEST 25 *****
TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
SET BIT7, VERIFY BIT7 WAS SET.
CLEAR BIT7, VERIFY BIT7 WAS CLEARED.

1838 ***** TEST 26 *****
TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
SET BIT8, VERIFY BIT8 WAS SET.
CLEAR BIT8, VERIFY BIT8 WAS CLEARED.

1868 ***** TEST 27 *****

1869 TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
SET BIT9, VERIFY BIT9 WAS SET.
CLEAR BIT9, VERIFY BIT9 WAS CLEARED.

1898 ***** TEST 30 *****
TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
SET BIT10, VERIFY BIT10 WAS SET.
CLEAR BIT10, VERIFY BIT10 WAS CLEARED.

1928 ***** TEST 31 *****
RECEIVER CONTROL REGISTER READ/WRITE BIT 1 RESET AND CLEAR TEST
WRITE BIT 1, AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND RECEIVER CONTROL REGISTER CLR INSTRUCTION

1960 ***** TEST 32 *****
RECEIVER CONTROL REGISTER READ/WRITE BIT 2 RESET AND CLEAR TEST
WRITE BIT 2, AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND RECEIVER CONTROL REGISTER CLR INSTRUCTION

1992 ***** TEST 33 *****
RECEIVER CONTROL REGISTER READ/WRITE BIT 3 RESET AND CLEAR TEST
WRITE BIT 3, AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND RECEIVER CONTROL REGISTER CLR INSTRUCTION

2024 ***** TEST 34 *****
RECEIVER CONTROL REGISTER READ/WRITE BIT 4 RESET AND CLEAR TEST
WRITE BIT 4, AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND RECEIVER CONTROL REGISTER CLR INSTRUCTION

2056 ***** TEST 35 *****
RECEIVER CONTROL REGISTER READ/WRITE BIT 8 RESET AND CLEAR TEST
WRITE BIT 8, AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND RECEIVER CONTROL REGISTER CLR INSTRUCTION

763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813

2088 ***** TEST 36 *****
TRANSMITTER CONTROL REGISTER READ/WRITE BIT 3 RESET AND CLEAR TE
WRITE BIT 3 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER CONTROL REGISTER CLR INSTRUCTION

2120 ***** TEST 37 *****
TRANSMITTER CONTROL REGISTER READ/WRITE BIT 4 RESET AND CLEAR TE
WRITE BIT 4 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER CONTROL REGISTER CLR INSTRUCTION

2152 ***** TEST 40 *****
TRANSMITTER CONTROL REGISTER READ/WRITE BIT 10 RESET AND CLEAR T
WRITE BIT 10 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER CONTROL REGISTER CLR INSTRUCTION

2184 ***** TEST 41 *****
TRANSMITTER CONTROL REGISTER READ/WRITE BIT 11 RESET AND CLEAR T
WRITE BIT 11 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER CONTROL REGISTER CLR INSTRUCTION

2216 ***** TEST 42 *****
TRANSMITTER CONTROL REGISTER READ/WRITE BIT 12 RESET AND CLEAR T
WRITE BIT 12 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER CONTROL REGISTER CLR INSTRUCTION

2248 ***** TEST 43 *****
TRANSMITTER CONTROL REGISTER READ/WRITE BIT 13 RESET AND CLEAR T
WRITE BIT 13 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER CONTROL REGISTER CLR INSTRUCTION

2280 ***** TEST 44 *****
TRANSMITTER BUFFER REGISTER READ/WRITE BIT 0 RESET AND CLEAR TES
WRITE BIT 0 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION

2312 ***** TEST 45 *****
TRANSMITTER BUFFER REGISTER READ/WRITE BIT 1 RESET AND CLEAR TES
WRITE BIT 1 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION

2344 ***** TEST 46 *****
TRANSMITTER BUFFER REGISTER READ/WRITE BIT 2 RESET AND CLEAR TES
WRITE BIT 2 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION

2376 ***** TEST 47 *****
TRANSMITTER BUFFER REGISTER READ/WRITE BIT 3 RESET AND CLEAR TES
WRITE BIT 3 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION

814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863

2408 ***** TEST 50 *****
TRANSMITTER BUFFER REGISTER READ/WRITE BIT 4 RESET AND CLEAR TES
WRITE BIT 4 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION

2440 ***** TEST 51 *****
TRANSMITTER BUFFER REGISTER READ/WRITE BIT 5 RESET AND CLEAR TES
WRITE BIT 5 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION

2472 ***** TEST 52 *****
TRANSMITTER BUFFER REGISTER READ/WRITE BIT 6 RESET AND CLEAR TES
WRITE BIT 6 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION

2504 ***** TEST 53 *****
TRANSMITTER BUFFER REGISTER READ/WRITE BIT 7 RESET AND CLEAR TES
WRITE BIT 7 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION

2536 ***** TEST 54 *****
TRANSMITTER BUFFER REGISTER READ/WRITE BIT 8 RESET AND CLEAR TES
WRITE BIT 8 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION

2568 ***** TEST 55 *****
TRANSMITTER BUFFER REGISTER READ/WRITE BIT 9 RESET AND CLEAR TES
WRITE BIT 9 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION

2600 ***** TEST 56 *****
TRANSMITTER BUFFER REGISTER READ/WRITE BIT 10 RESET AND CLEAR TE
WRITE BIT 10 AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION

2632 ***** TEST 57 *****
RECEIVER BUFFER REGISTER TEST
TEST THAT RECEIVER BUFFER REGISTER CANNOT BE WRITTEN.
WRITE RECEIVER BUFFER REGISTER WITH ALL 1'S
AND VERIFY THAT ALL 0'S ARE READ BACK.

2658 ***** TEST 60 *****

2660 PARAMETER STATUS REGISTER TEST
TEST THAT PARAMETER STATUS REGISTER CANNOT BE WRITTEN
READ THE PARAMETER STATUS REGISTER AND STORE THE DATA;
COMPLEMENT THE DATA AND WRITE THE PARAMETER STATUS REGISTER
VERIFYING THAT THE PARAMETER STATUS REGISTER DID NOT CHANGE.

864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914

2688 ***** TEST 61 *****
RECEIVER CONTROL REGISTER BIT 0 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 0 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

2728 ***** TEST 62 *****
RECEIVER CONTROL REGISTER BIT 7 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 7 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

2768 ***** TEST 63 *****
RECEIVER CONTROL REGISTER BIT 9 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 9 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

2808 ***** TEST 64 *****
RECEIVER CONTROL REGISTER BIT 10 READ ONLY DEVICE RESET AND CLEA
WRITE BIT 10 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

2848 ***** TEST 65 *****
RECEIVER CONTROL REGISTER BIT 11 READ ONLY DEVICE RESET AND CLEA
WRITE BIT 11 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

2888 ***** TEST 66 *****
RECEIVER CONTROL REGISTER BIT 12 READ ONLY DEVICE RESET AND CLEA
WRITE BIT 12 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

2928 ***** TEST 67 *****
RECEIVER CONTROL REGISTER BIT 13 READ ONLY DEVICE RESET AND CLEA
WRITE BIT 13 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

2968 ***** TEST 70 *****
RECEIVER CONTROL REGISTER BIT 14 READ ONLY DEVICE RESET AND CLEA
WRITE BIT 14 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3008 ***** TEST 71 *****
RECEIVER CONTROL REGISTER BIT 15 READ ONLY DEVICE RESET AND CLEA
WRITE BIT 15 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3048 ***** TEST 72 *****
RECEIVER BUFFER REGISTER BIT 0 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 0 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965

3088 ***** TEST 73 *****
RECEIVER BUFFER REGISTER BIT 1 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 1 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3128 ***** TEST 74 *****
RECEIVER BUFFER REGISTER BIT 2 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 2 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3168 ***** TEST 75 *****
RECEIVER BUFFER REGISTER BIT 3 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 3 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3208 ***** TEST 76 *****
RECEIVER BUFFER REGISTER BIT 4 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 4 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3248 ***** TEST 77 *****
RECEIVER BUFFER REGISTER BIT 5 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 5 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3288 ***** TEST 100 *****
RECEIVER BUFFER REGISTER BIT 6 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 6 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3328 ***** TEST 101 *****
RECEIVER BUFFER REGISTER BIT 7 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 7 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3368 ***** TEST 102 *****
RECEIVER BUFFER REGISTER BIT 8 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 8 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3408 ***** TEST 103 *****
RECEIVER BUFFER REGISTER BIT 9 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 9 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3448 ***** TEST 104 *****
RECEIVER BUFFER REGISTER BIT 10 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 10 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017

3488 ***** TEST 105 *****
RECEIVER BUFFER REGISTER BIT 12 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 12 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3528 ***** TEST 106 *****
RECEIVER BUFFER REGISTER BIT 14 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 14 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3568 ***** TEST 107 *****
RECEIVER BUFFER REGISTER BIT 15 READ ONLY DEVICE RESET AND CLEAR
WRITE BIT 15 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3607 ***** TEST 110 *****
THIS TEST VERIFIES BIT7 OF THE TRANSMITTER CONTROL REGISTER
TEST THAT BIT 7 SETS AFTER A RESET AND MRESET
VERIFY THAT WRITING THE LOW BYTE OF
THE TRANSMITTER BUFFER REGISTER CLEARS BIT 7

3647 ***** TEST 111 *****
TRANSMITTER CONTROL REGISTER BIT 9 READ ONLY DEVICE RESET AND CL
WRITE BIT 9 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3689 ***** TEST 112 *****
TRANSMITTER CONTROL REGISTER BIT 14 READ ONLY DEVICE RESET AND C
WRITE BIT 14 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3731 ***** TEST 113 *****
TRANSMITTER CONTROL REGISTER BIT 15 READ ONLY DEVICE RESET AND C
WRITE BIT 15 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3773 ***** TEST 114 *****
TRANSMITTER BUFFER REGISTER BIT 12 READ ONLY DEVICE RESET AND CL
WRITE BIT 12 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3813 ***** TEST 115 *****
TRANSMITTER BUFFER REGISTER BIT 13 READ ONLY DEVICE RESET AND CL
WRITE BIT 13 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3853 ***** TEST 116 *****
TRANSMITTER BUFFER REGISTER BIT 14 READ ONLY DEVICE RESET AND CL
WRITE BIT 14 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070

3893 ***** TEST 117 *****
TRANSMITTER BUFFER REGISTER BIT 15 READ ONLY DEVICE RESET AND CL
WRITE BIT 15 A ONE AND VERIFY A ZERO IS READ BACK
REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

3933 ***** TEST 120 *****
PARAMETER STATUS REGISTER BIT 9 WRITE ONLY DEVICE RESET AND CLEA
TEST THAT BIT 9 CANNOT BE WRITTEN AND READ
BACK THE SAME. READ BIT 9, COMPLEMENT IT AND WRITE IT
VERIFYING THAT IT DID NOT CHANGE. WRITE BIT 9,
DO A DEVICE RESET AND VERIFY THE BIT WAS CLEARED.
REPEAT FOR A CLR INSTRUCTION.

3981 ***** TEST 121 *****
PARAMETER STATUS REGISTER BIT 12 WRITE ONLY DEVICE RESET AND CLE
TEST THAT BIT 12 CANNOT BE WRITTEN AND READ
BACK THE SAME. READ BIT 12, COMPLEMENT IT AND WRITE IT
VERIFYING THAT IT DID NOT CHANGE. WRITE BIT 12,
DO A DEVICE RESET AND VERIFY THE BIT WAS CLEARED.
REPEAT FOR A CLR INSTRUCTION.

4029 ***** TEST 122 *****
PARAMETER STATUS REGISTER BIT 15 WRITE ONLY DEVICE RESET AND CLE
TEST THAT BIT 15 CANNOT BE WRITTEN AND READ
BACK THE SAME. READ BIT 15, COMPLEMENT IT AND WRITE IT
VERIFYING THAT IT DID NOT CHANGE. WRITE BIT 15,
DO A DEVICE RESET AND VERIFY THE BIT WAS CLEARED.
REPEAT FOR A CLR INSTRUCTION.

4077 ***** TEST 123 *****
TRANSMITTER CONTROL REGISTER BIT 8 WRITE ONLY DEVICE RESET AND C
TEST THAT BIT 8 CANNOT BE WRITTEN AND READ
BACK THE SAME. READ BIT 8, COMPLEMENT IT AND WRITE IT
VERIFYING THAT IT DID NOT CHANGE. WRITE BIT 8,
DO A DEVICE RESET AND VERIFY THE BIT WAS CLEARED.
REPEAT FOR A CLR INSTRUCTION.

4125 ***** TEST 124 *****
RECEIVER CONTROL REGISTER READ/WRITE BIT TEST.
SET BIT8, VERIFY BIT8 WAS SET.
CLEAR BIT8, VERIFY BIT8 WAS CLEARED.

4155 ***** TEST 125 *****
RECEIVER CONTROL REGISTER READ/WRITE BIT TEST.
SET BIT6, VERIFY BIT6 WAS SET.
CLEAR BIT6, VERIFY BIT6 WAS CLEARED.

4185 ***** TEST 126 *****
RECEIVER CONTROL REGISTER READ/WRITE BIT TEST.
SET BITS, VERIFY BITS WAS SET.
CLEAR BITS, VERIFY BITS WAS CLEARED.

1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120

4215 ***** TEST 127 *****
RECEIVER CONTROL REGISTER READ/WRITE BIT TEST.
SET BIT4, VERIFY BIT4 WAS SET.
CLEAR BIT4, VERIFY BIT4 WAS CLEARED.

4245 ***** TEST 130 *****
TRANSMITTER CONTROL REGISTER READ/WRITE BIT TEST.
SET BIT6, VERIFY BIT6 WAS SET.
CLEAR BIT6, VERIFY BIT6 WAS CLEARED.

4277 ***** TEST 131 *****
RECEIVER CONTROL REGISTER READ/WRITE BIT TEST.
SET BIT1, VERIFY BIT1 WAS SET.
CLEAR BIT1, VERIFY BIT1 WAS CLEARED.

4309 ***** TEST 132 *****
RECEIVER CONTROL REGISTER READ/WRITE BIT TEST.
SET BIT2, VERIFY BIT2 WAS SET.
CLEAR BIT2, VERIFY BIT2 WAS CLEARED.

4341 ***** TEST 133 *****
RECEIVER CONTROL REGISTER READ/WRITE BIT TEST.
SET BIT3, VERIFY BIT3 WAS SET.
CLEAR BIT3, VERIFY BIT3 WAS CLEARED.

4373 ***** TEST 134 *****
RECEIVER CONTROL REGISTER READ/WRITE BIT 6 RESET AND CLEAR TEST
WRITE BIT 6, AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND RECEIVER CONTROL REGISTER CLR INSTRUCTION

4405 ***** TEST 135 *****
RECEIVER CONTROL REGISTER READ/WRITE BIT 5 RESET AND CLEAR TEST
WRITE BIT 5, AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND RECEIVER CONTROL REGISTER CLR INSTRUCTION

4437 ***** TEST 136 *****
TRANSMITTER CONTROL REGISTER READ/WRITE BIT 6 RESET AND CLEAR TE
WRITE BIT 6, AND TEST THAT IT WILL BE CLEARED AFTER A
DEVICE RESET AND TRANSMITTER CONTROL REGISTER CLR INSTRUCTION

4469 ***** TEST 137 *****
THIS TEST CHECKS THE MAINTENANCE CLOCK
USED THROUGHOUT THE REMAINING DIAGNOSTICS

4506 ***** TEST 140 *****
THIS TEST WILL PERFORM STATIC TRANSMITTER FUNCTIONS
IN MAINTENANCE MODE. IT WILL PROVE THE INTERACTION
OF SEND, DONE AND TSOM.

1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173

- 4553 ***** TEST 141 *****
THIS TEST CHECKS THE STATIC FUNCTIONS OF THE TRANSMITTER
IN MAINTENANCE MODE. THIS TEST PROVES THE INTERACTION OF
TXDBUF, TXACT, TSOM, TRANSMITTED DATA AND DONE.
- 4628 ***** TEST 142 *****
THIS TEST VERIFIES THAT THE DEVICE IDLES FLAGS
IDLE A MINIMUM OF 64. FLAGS.
- 4654 ***** TEST 143 *****
THIS TEST PUSHES DATA THRU THE TRANSMITTER
IN MAINTENANCE MODE. THE TEST SENDS A FLAG,
AND TWO ALTERNATING ONES AND ZEROES CHARACTERS,
AN ALL ZEROES CHARACTER AND AN ALL ONES
CHARACTER TO VERIFY THE BIT STUFF CAPABILITY OF
THE DUP WITHOUT A CRC CHECK. THE TEST ROTATES
THE BITS THRU, SAMPLING THE DATA ON A BIT-BY-BIT
BASIS, LSB FIRST. IT STORES THE BIT IN THE MSB OF
THE SAVE LOCATION, COMPARES AND ROTATES RIGHT UNTIL
THE CHARACTER IS ASSEMBLED.
- 4763 ***** TEST 144 *****
THIS TEST VERIFIES THE ABORT SEQUENCE AND
NORMAL DATA SEQUENCE OF FLAG, DATA, FLAG
FOLLOWED BY IDLE LINE. THIS TEST ALSO PROVES
THE FUNCTIONING OF ACTIVE, TSOM, TEOM, SEND AND DONE.
WITHOUT USING A CRC CHECK.
- 4831 ***** TEST 145 *****
THIS TEST VERIFIES THAT A DATA
UNDER RUN CONDITION WILL CAUSE
THE TRANSMITTER DATA LATE BIT TO SET
AND THAT THE DEVICE WILL ABORT
UNTIL SEND IS CLEARED WHEN THE OUTPUT GOES TO A SPACE
- 4879 ***** TEST 146 *****
THIS TEST VERIFIES THAT DROPPING OF
SEND BEFORE SETTING TEOM CAUSES
A SPACE TO BE OUTPUT AFTER COMPLETION OF
A CHARACTER WITH AND WITHOUT BIT STUFF.
- 4952 ***** TEST 147 *****
THIS TEST VERIFIES THAT A DATA UNDERRUN
CONDITION WILL CAUSE THE TRANSMITTER DATA
LATE BIT TO SET AND THAT THE DUP WILL GO TO
A MARK OUTPUT
- 4998 ***** TEST 150 *****
THIS TEST VERIFIES THAT SETTING TEOM
AND TSOM AT THE SAME TIME WILL HOLD
ACTIVE UP AND SEND A FLAG

1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189

5033 ***** TEST 151 *****
TEST OF THE BCC OPERATION USING
CRC.CCITT FOR THE POLYNOMIAL. SPECIFIC
DATA PATTERNS ARE USED TO ISOLATE FAULTS.

5213 ***** TEST 152 *****
THIS TEST IS AN AID FOR DEBUGGING CRC
ERRORS. A CHARACTER IS LOADED INTO THE
DUP AND PUSHED OUT BIT BY BIT WHILE
ALLOWING THE OPERATOR TO MONITOR THE CRC
CHARACTER AS IT IS GENERATED. THE DATA CHARACTER
CAN ALSO BE CHANGED BY THE OPERATOR.
PUT SW09=1 TO LOCK ON BITS. TO CONTINUE HIT
ANY KEY ON THE TTY. AFTER 16 TIMES PUT DOWN SW09 TO LEAVE

3

1190
 1191
 1192
 1193
 1194
 1195
 1196
 1197
 1198
 1199
 1200
 1201
 1202
 1203
 1204
 1205
 1206
 1207
 1208
 1209
 1210
 1211
 1212
 1213
 1214
 1215
 1216
 1217
 1218
 1219
 1220
 1221
 1222
 1223

:*MAINDEC-11-DZDPB-B /<377>/BASIC DUP11 AND OFFLINE SDLC TRANSMITTER TESTS
 :*COPYRIGHT(C) 1975,1977, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
 :*

```

:STARTING PROCEDURE
:LOAD PROGRAM
:LOAD ADDRESS 000200
:PRESS START
:PROGRAM WILL TYPE "MAINDEC-11-DZDPB-B /<377>/BASIC DUP11 AND OFFLINE SDLC TRANS
:PROGRAM WILL TYPE "R" TO INDICATE THAT TESTING HAS STARTED
:AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE
:AND THEN RESUME TESTING
    
```

:SWITCH REGISTER OPTIONS

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

SW15=100000
 SW14=40000
 SW13=20000
 SW12=10000
 SW11=4000
 SW10=2000
 SW09=1000
 SW08=400
 SW07=200
 SW06=100
 SW05=40
 SW04=20
 SW03=10
 SW02=4
 SW01=2
 SW00=1

```

:=1, HALT ON ERROR
:=1, LOOP ON CURRENT TEST
:=1, INHIBIT ERROR TYPEOUT
:=1, DELETE TYPEOUT/BELL ON ERROR.
:=1, INHIBIT ITERATIONS
:=1, ESCAPE TO NEXT TEST ON ERROR
:=1, LOOP WITH CURRENT DATA
:=1, LOOP ON ERROR

:SELECT DUP'S DESIRED ACTIVE
:NOTE: THIS MUST NOT EXCEED ORIGINAL COUNT
:LOCK ON TEST SELECT
:RESTART PROGRAM AT SELECTED TEST
:ENTER PARAMETERS
    
```



```

1224
1225
1226
1227
1228
1229      000000      RD=%0      :GENERAL REGISTER
1230      000001      R1=%1      :GENERAL REGISTER
1231      000002      R2=%2      :GENERAL REGISTER
1232      000003      R3=%3      :GENERAL REGISTER
1233      000004      R4=%4      :GENERAL REGISTER
1234      000005      R5=%5      :GENERAL REGISTER
1235      000006      SP=%6      :PROCESSOR STACK POINTER
1236      000007      PC=%7      :PROGRAM COUNTER
1237
1238      :LOCATION EQUIVALENCIES
1239      :-----
1240
1241      177776      PS=177776      :PROCESSOR STATUS WORD
1242      001150      STACK=1150      :START OF PROCESSOR STACK
1243
1244      :INSTRUCTION DEFINITIONS
1245      :-----
1246
1247      005746      PUSH1SP=5746      :DECREMENT PROCESSOR STACK 1 WORD
1248      005726      POP1SP=5726      :INCREMENT PROCESSOR STACK 1 WORD
1249      010046      PUSHRO=10046      :SAVE R0 ON STACK
1250      012600      POPRO=12600      :RESTORE R0 FROM STACK
1251      024646      PUSH2SP=24646      :DECREMENT STACK TWICE
1252      022626      POP2SP=22626      :INCREMENT STACK TWICE
1253      .EQUIV EMT,HLT :BASIC DEFINITION OF ERROR CALL
1254
1255
1256      100000      BIT15=100000
1257      040000      BIT14=40000
1258      020000      BIT13=20000
1259      010000      BIT12=10000
1260      004000      BIT11=4000
1261      002000      BIT10=2000
1262      001000      BIT9=1000
1263      000400      BIT8=400
1264      000200      BIT7=200
1265      000100      BIT6=100
1266      000040      BIT5=40
1267      000020      BIT4=20
1268      000010      BIT3=10
1269      000004      BIT2=4
1270      000002      BIT1=2
1271      000001      BIT0=1
1272
1273

```

TRAPCATCHER FOR UNEXPECTED INTERRUPTS

```

1274 :*****
1275 :-----
1276 : TRAPCATCHER FOR ILLEGAL INTERRUPTS
1277 : THE STANDARD "TRAP CATCHER" IS PLACED
1278 : BETWEEN ADDRESS 0 TO ADDRESS 776.
1279 : IT LOOKS LIKE "PC+2 HALT".
1280 :-----
1281 :*****
1282
1283      000000      .=0
1284      : STANDARD INTERRUPT VECTORS
1285      :-----
1286
1287      000024      .=24
1288      000024      005050      .PFAIL      : POWER FAIL HANDLER
1289      000026      000340      340          : SERVICE AT LEVEL 7
1290      000030      004350      .HLT       : ERROR HANDLER
1291      000032      000340      340          : SERVICE AT LEVEL 7
1292      000034      004316      .TRPSRV    : GENERAL HANDLER DISPATCH SERVICE
1293      000036      000340      340          : SERVICE AT LEVEL 7
1294
1295      000040      000040      .=40
1296      000042      000000      0          : SAVE FOR ACT-11 OR DDP2
1297      000044      000000      0          : RETURN ADDRESS IF UNDER ACT-11 OR DDP2
1298      000046      003104      0          : SAVE FOR ACT-11 OR DDP2
1299      000052      000052      $ENDAD     : FOR USE WITH ACT-11 OR DDP2
1300      000052      000000      .=52
1301      0          : ACT-11 PROGRAM CHARACTERISTICS
1302
1303      000174      000000      .=174
1304      000176      000000      DISPREG:0  : SOFTWARE DISPLAY REGISTER
1305      000200      000200      SWREG: 0   : SOFTWARE SWITCH REGISTER
1306      000200      000137      001562     .=200
1307      JMP      .START      : GO TO START OF PROGRAM
1308
1309
1310      001000      001000      040515  047111  .=1000
1311      005377      MTITLE: .ASCIZ <377><12>/MAINDEC-11-DZDPB-B /<377>/BASIC DUP11 AND OFFLINE SDLC TRANSMI
1312      (2)
1313      001200      .=1200
1314      : SWR AND LIGHTS
1315      :-----
1316      001200      177570      DISPLAY: 177570      : 11/45 CONSOLE LIGHTS
1317      001202      177570      SWR: 177570         : INDIRECT POINTER TO SWITCH REGISTER
1318      : INDIRECT POINTERS TO TELETYPE VECTORS AND REGISTERS
1319      :-----
1320
1321      001204      177560      TKCSR: 177560      : TELETYPE KEYBOARD CONTROL REGISTER
1322      001206      177562      TKDBR: 177562      : TELETYPE KEYBOARD DATA BUFFER
1323      001210      177564      TPCSR: 177564      : TELEPRINTER CONTROL REGISTER
1324      001212      177566      TPDBR: 177566      : TELEPRINTER DATA BUFFER
1325
1326      : PROGRAM CONTROL PARAMETERS
1327      :-----
1328

```

PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

1329	001214	000000	RETURN:	0	: SCOPE ADDRESS FOR LOOP ON TEST
1330	001216	000000	NEXT:	0	: ADDRESS OF NEXT TEST TO BE EXECUTED
1331	001220	000000	LOCK:	0	: ADDRESS FOR LOCK ON CURRENT DATA
1332	001222	000001	ICOUNT:	1	: NUMBER OF ITERATIONS THAT CURRENT TEST WILL BE EXECUTED
1333	001224	000000	LPCNT:	0	: NUMBER OF ITERATIONS COMPLETED
1334	001226	000000	TSTNO:	0	: NUMBER OF TEST IN PROGRESS
1335	001230	000000	PASCNT:	0	: NUMBER OF PASSES COMPLETED
1336	001232	000000	ERRCNT:	0	: TOTAL NUMBER OF ERRORS
1337	001234	000000	LSTERR:	0	: PC OF LAST ERROR CALL
1338					
1339					
1340			: PROGRAM VARIABLES		
1341			: -----		
1342	001236	000000	TEMP1:	0	: TEMPORARY STORAGE
1343	001240	000000	TEMP2:	0	: TEMPORARY STORAGE
1344	001242	000000	TEMP3:	0	: TEMPORARY STORAGE
1345	001244	000000	TEMP4:	0	: TEMPORARY STORAGE
1346	001246	000000	TEMP5:	0	: TEMPORARY STORAGE
1347	001250	000000	SAVR0:	0	: R0 STORAGE
1348	001252	000000	SAVR1:	0	: R1 STORAGE
1349	001254	000000	SAVR2:	0	: R2 STORAGE
1350	001256	000000	SAVR3:	0	: R3 STORAGE
1351	001260	000000	SAVR4:	0	: R4 STORAGE
1352	001262	000000	SAVR5:	0	: R5 STORAGE
1353	001264	000000	SAVSP:	0	: STACK POINTER STORAGE
1354	001266	000000	SAVPC:	0	: PROGRAM COUNTER STORAGE
1355					
1356	001270	000000	SAVR0A:	0	: R0 STORAGE
1357	001272	000000	SAVR1A:	0	: R1 STORAGE
1358	001274	000000	SAVR2A:	0	: R2 STORAGE
1359	001276	000000	SAVR3A:	0	: R3 STORAGE
1360	001300	000000	SAVR4A:	0	: R4 STORAGE
1361	001302	000000	SAVR5A:	0	: R5 STORAGE
1362	001304	000000	SAVSPA:	0	: STACK POINTER STORAGE
1363	001306	000000	SAVPCA:	0	: PROGRAM COUNTER STORAGE
1364					
1365	001310	000001	DUPACTV:	.BLKB 1	: DUP11'S SELECTED ACTIVE.
1366	001311	000001	DUPNUM:	.BLKB 1	: OCTAL NUMBER OF DUP11'S.
1367	001312	000001	SAVACT:	.BLKB 1	: ORIGINAL ACTV. DEVICES.
1368	001313	000001	SAVNUM:	.BLKB 1	: WORKABLE NUMBER.
1369	001314	000001	RUN:	.BLKB 1	: POINTER ONE PAST RUNNING DEVICE.
1370		001316	.EVEN		
1371	001316	001500	CREAM:	DUP.MAP	: TABLE POINTER.

```

1372
1373
1374
1375
1376      100000
1377      040000
1378      020000
1379      010000
1380      004000
1381      002000
1382      001000
1383      000400
1384      000200
1385      000100
1386      000040
1387      000020
1388      000010
1389      000004
1390      000002
1391      000001
1392
1393      100000
1394      040000
1395      010000
1396      002000
1397      001000
1398      000400
1399
1400      100000
1401      001000
1402      010000
1403
1404      100000
1405      040000
1406      020000
1407      010000
1408      004000
1409      002000
1410      001000
1411      000400
1412      000200
1413      000100
1414      000020
1415      000010
1416
1417      000000
1418      014000
1419      010000
1420      004000
1421
1422
1423
1424      100000
1425      040000
1426      020000
1427      010000

;CONTROL REGISTER DEFINITIONS
-----
;RXCSR BIT DEFINITIONS
DSCA=BIT15      ;DATA SET CHANGE A
RING=BIT14      ;RING
CTS=BIT13       ;CLR TO SEND
CARDET=BIT12    ;CARRIER DETECT
RECACT=BIT11    ;REC ACTIVE
SRD=BIT10       ;SEC REC DATA
DSR=BIT9        ;DATA SET RDY
STPSYN=BIT8     ;STRIP SYNC
RXDONE=BIT7     ;REC DONE
RINTEN=BIT6     ;REC INTR ENABLE
DSINTE=BIT5     ;DSC INTR ENABLE
RCVEN=BIT4      ;REC ENABLE
STD=BIT3        ;SEC XMIT DATA
RTS=BIT2        ;REQ TO SEND
DTR=BIT1        ;DATA TERM RDY
DSCB=BIT0       ;DATA SET CHANGE B

;RXDBUF BIT DEFINITIONS
RXDERR=BIT15    ;REC DATA ERROR
OVRUN=BIT14     ;OVERRUN ERROR
CRCERR=BIT12    ;CRC ERROR
RABORT=BIT10    ;REC ABORT
REOM=BIT9       ;REC END OF MESSAGE
RSOM=BIT8       ;REC START OF MESSAGE

;PARCSR BIT DEFINITIONS
DECMOD=BIT15    ;DEC MODE (DDCMP)
CRCEN=BIT9      ;CRC ENABLE
PRISEC=BIT12    ;PRI/SEC SELECT

;TXCSR BIT DEFINITIONS
TXDLAT=BIT15    ;TX DATA LATE
MTDATA=BIT14    ;MAINT DATA OUT
CLK=BIT13       ;CLK
MMODEB=BIT12    ;MAINT MODE B
MMODEA=BIT11    ;MAINT MODE A
BITW=BIT10      ;BIT WINDOW INPUT
TXACT=BIT9      ;TX ACTIVE
MRESET=BIT8     ;MASTER RESET
TXDONE=BIT7     ;XMIT DONE
TXINTE=BIT6     ;XMIT DONE INTR ENABLE
SEND=BIT4       ;SEND
HDXEN=BIT3      ;HDX/FDX

;TXCSR WRD DEFINITIONS
USER=0          ;USER MODE
MMODE=14000     ;MAINT INT MODE
MEXT=10000      ;MAINT EXT MODE
SYSTST=4000     ;SYSTEM TEST MODE

;TXDBUF BIT DEFINITIONS
-----
RCRC7T=BIT15
RCRCIN=BIT14
TCRC7T=BIT13
TCRCIN=BIT12
    
```

DZDPBB MACY11 27(1006) 18-MAY-77 00:00 PAGE 36
 DZDPBB.P11 13-MAY-77 15:27

PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

SEQ 0035

1428 004000
 1429 002000
 1430 001000
 1431 000400
 1432
 1433
 1434
 1435 001320 000000
 1436 001322 000001
 1437 001323 000001
 1438 001324 000000
 1439 001326 000000
 1440 001330 000000
 1441 001332 000000
 1442 001334 000001
 1443 001336 000001
 1444
 1445

TIMER=BIT11 ; MAINTENANCE TIMER
 TABORT=BIT10 ; TRANSMIT ABORT
 TEOM=BIT9 ; TRANSMIT END OF MESSAGE
 TSOM=BIT8 ; TRANSMIT START OF MESSAGE

; MISC. PROGRAM DEFINITIONS

 PIRTY: .WORD 0
 TCNFLG: .BLKB 1
 OPCLRJ: .BLKB 1
 DATA: .WORD 0
 SHIFTS: .WORD 00
 MIND: .WORD 00
 FLAG: .WORD 0
 STJMFL: .BLKW 1
 SRJMFL: .BLKW 1

1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492

PROGRAM CONTROL FLAGS

001340 000
001341 000
001342 000
001343 000

INIFLG: .BYTE 0
ERRFLG: .BYTE 0
LOKFLG: .BYTE 0
QV.FLG: .BYTE 0

PROGRAM INITIALIZATION FLAG
ERROR OCCURED FLAG
LOCK ON CURRENT TEST FLAG
QUICK VERIFY FLAG.
ON FIRST PASS OF EACH DUP11 ITERATIONS
WILL BE SUPPRESSED

000000

.EVEN
SY=0

DEFINITIONS FOR TRAP SUBROUTINE CALLS
POINTERS TO SUBROUTINES CAN BE FOUND
IN THE TABLE IMMEDIATLY FOLLOWING THE DEFINITIONS

001344
001344 104400
001344 003160
104401
001346 003312
104402
001350 003336
104403
001352 003412
104404
001354 003516
104405
001356 003536
104406
001360 003736
104407
001362 003776
104410
001364 004030
104411
001366 004034
104412
001370 005006
104413
001372 004242

TRPTAB:
SCOPE=TRAP+0 ;CALL TO SCOPE LOOP AND ITERATION HANDLER
.SCOPE
SCOPI=TRAP+1 ;CALL TO LOOP ON CURRENT DATA HANDLER
.SCOPI
TYPE=TRAP+2 ;CALL TO TELETYPE OUTPUT ROUTINE
.TYPE
INSTR=TRAP+3 ;CALL TO ASCII STRING INPUT ROUTINE
.INSTR
INSTER=TRAP+4 ;CALL TO INPUT ERROR HANDLER
.INSTER
PARAM=TRAP+5 ;CALL TO NUMERICAL DATA INPUT ROUTINE
.PARAM
SAVOS=TRAP+6 ;CALL TO REGISTER SAVE ROUTINE
.SAVOS
RESOS=TRAP+7 ;CALL TO REGISTER RESTORE ROUTINE
.RESOS
CONVRT=TRAP+10 ;CALL TO DATA OUTPUT ROUTINE
.CONVRT
CNVRT=TRAP+11 ;CALL TO DATA OUTPUT ROUTINE WITHOUT CR/LF.
.CNVRT
PKCLK=TRAP+12 ;CALL TO CLOCK ROUTINE
.PKCLK
SETFLG=TRAP+13 ;CALL TO TELETYPE INPUT ROUTINE
.SETFLG

PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

```

1493                                     ;DUP11 VECTOR AND REGISTER INDIRECT POINTERS
1494
1495 001374 000000 DUPRVC: 0 ; POINTER TO DUP11 RECEIVER INTERRUPT VECTOR
1496 001376 000000 DUPRPS: 0 ; POINTER TO DUP11 RECEIVER INTERRUPT SERVICE PS
1497 001400 000000 DUPTVC: 0 ; POINTER TO DUP11 TRANSMITTER INTERRUPT VECTOR
1498 001402 000000 DUPTPS: 0 ; POINTER TO DUP11 TRANSMITTER INTERRUPT SERVICE PS
1499 001404 000000 RXCSR: 0 ; POINTER TO DUP11 RECEIVER STATUS REGISTER
1500 001406 000000 RXDBUF: 0 ; POINTER TO DUP11 RECEIVER DATA BUFFER
1501 001410 000000 PARCSR: 0 ; POINTER TO DUP11 PARAMETER STATUS REGISTER
1502 001412 000000 TXCSR: 0 ; POINTER TO DUP11 TRANSMITTER STATUS REGISTER
1503 001414 000000 TXDBUF: 0 ; POINTER TO DUP11 TRANSMITTER DATA BUFFER
1504 001416 000000 DUPSEC: 0 ; POINTER TO DUP11 SECONDARY REGISTER SELECT REGISTER
1505 001420 000000 HUPPSR: 0 ; POINTER TO PARAMETER STATUS HIGH BYTE
1506 001422 000000 HUPRBF: 0 ; POINTER TO RECEIVER BUFFER HIGH BYTE
1507 001424 000000 HUPRCR: 0 ; POINTER TO RECEIVER CONTROL REG HIGH BYTE
1508 001426 000000 HUPTBF: 0 ; POINTER TO TRANSMITTER BUFFER HIGH BYTE
1509 001430 000000 HUPTCR: 0 ; POINTER TO TRANSMITTER CONTROL REG HIGH BYTE
1510
1511
1512                                     ;DUP11 CONTROL INDICATORS FOR CURRENT DUP11 UNDER TEST
1513                                     ;-----
1514
1515 001432 000 MASK.A: .BYTE 000 ; LAST CHAR TO TEST AND PARITY MASK
1516
1517 001433 010 CLK.A: .BYTE 8. ; NUMBER OF CLOCKS NEEDED FOR ONE CHAR
1518
1519 001434 000000 L00.00: 000000 ; PARAMETERS
1520

```

PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

; DUP11 STATUS TABLE AND ADDRESS ASSIGNMENTS

1521				
1522				
1523				
1524		001500	.=1500	
1525	001500	001500	DUP.MAP:	
1526	001500	000001	DUPCR0: .BLKW 1	; CONTROL STATUS REGISTER FOR DUP11 NUMBER 0
1527	001502	000001	DUPTR0: .BLKW 1	; VECTOR "A" FOR DUP11 NUMBER 0
1528	001504	000001	DUP0.A: .BLKW 1	; PARAMETER FOR DUP11 NUMBER 0
1529				
1530	001506	000001	DUPCR1: .BLKW 1	; CONTROL STATUS REGISTER FOR DUP11 NUMBER 1
1531	00151C	000001	DUPTR1: .BLKW 1	; VECTOR "A" FOR DUP11 NUMBER 1
1532	001512	000001	DUP1.A: .BLKW 1	; PARAMETER FOR DUP11 NUMBER 1
1533				
1534	001514	000001	DUPCR2: .BLKW 1	; CONTROL STATUS REGISTER FOR DUP11 NUMBER 2
1535	001516	000001	DUPTR2: .BLKW 1	; VECTOR "A" FOR DUP11 NUMBER 2
1536	001520	000001	DUP2.A: .BLKW 1	; PARAMETER FOR DUP11 NUMBER 2
1537				
1538	001522	000001	DUPCR3: .BLKW 1	; CONTROL STATUS REGISTER FOR DUP11 NUMBER 3
1539	001524	000001	DUPTR3: .BLKW 1	; VECTOR "A" FOR DUP11 NUMBER 3
1540	001526	000001	DUP3.A: .BLKW 1	; PARAMETER FOR DUP11 NUMBER 3
1541				
1542	001530	000001	DUPCR4: .BLKW 1	; CONTROL STATUS REGISTER FOR DUP11 NUMBER 4
1543	001532	000001	DUPTR4: .BLKW 1	; VECTOR "A" FOR DUP11 NUMBER 4
1544	001534	000001	DUP4.A: .BLKW 1	; PARAMETER FOR DUP11 NUMBER 4
1545				
1546	001536	000001	DUPCR5: .BLKW 1	; CONTROL STATUS REGISTER FOR DUP11 NUMBER 5
1547	001540	000001	DUPTR5: .BLKW 1	; VECTOR "A" FOR DUP11 NUMBER 5
1548	001542	000001	DUP5.A: .BLKW 1	; PARAMETER FOR DUP11 NUMBER 5
1549				
1550	001544	000001	DUPCR6: .BLKW 1	; CONTROL STATUS REGISTER FOR DUP11 NUMBER 6
1551	001546	000001	DUPTR6: .BLKW 1	; VECTOR "A" FOR DUP11 NUMBER 6
1552	001550	000001	DUP6.A: .BLKW 1	; PARAMETER FOR DUP11 NUMBER 6
1553				
1554	001552	000001	DUPCR7: .BLKW 1	; CONTROL STATUS REGISTER FOR DUP11 NUMBER 7
1555	001554	000001	DUPTR7: .BLKW 1	; VECTOR "A" FOR DUP11 NUMBER 7
1556	001556	000001	DUP7.A: .BLKW 1	; PARAMETER FOR DUP11 NUMBER 7
1557				
1558	001560	000000	DUP.END:	000000
1559				
1560				
1561				
1562				
1563				


```

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 I
I C I O I N I T I R I O I L I I R I E I G I I S I T I E I R I
I I I I I I I I I I I I I I I I I I I I I I
-----
I I I I I I I I I I I * I V I E I C I T I O I R I * I
I I I I I I I I I I I I I I I I I I I I I I
-----
I I I I I I I I I I I I I * I * I S I Y I N I C 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
-----

```

DEFINITIONS

- A- OPTIONAL CLEAR JUMPER IN=1
- B- TURNAROUND CONNECTOR ON=1
- C-
- D-

```

1583
1584
1585
1586
1587
1588
1589
1590
1591 001562 012737 000340 177776 .START: MOV #340,PS ;LOCK OUT INTERRUPTS
1592 001570 012706 001150 MOV #STACK,SP ;SET UP STACK
1593 001574 012737 005050 000024 MOV #.PFAIL,#24 ;SET UP POWER FAIL VECTOR
1594 001602 113737 001311 001313 MOV# DUPNUM,SAVNUM ;SAVE NUMBER OF DEVICES IN SYSTEM
1595 001610 005037 001230 CLR PASCNT ;CLEAR PASS COUNT
1596 001614 105037 001341 CLR# ERRFLG ;CLEAR ERROR FLAG
1597 001620 105037 001343 CLR# QV.FLG ;ZERO QUICK VERIFY FLAG
1598 001624 012737 001500 001316 MOV #DUP.MAP,CREAM ;GET MAP POINTER.
1599 001632 112737 000001 001314 MOV# #1,RUN ;POINT POINTER TO FIRST DEVICE.
1600 001640 005037 001232 CLR ERRCNT ;CLEAR ERROR COUNT
1601 001644 005037 001234 CLR LSTERR ;CLEAR LAST ERROR POINTER
1602 001650 012737 000001 001226 MOV #1,TSTNO ;SET UP FOR TEST 1
1603 001656 012737 001562 001214 MOV #.START,RETURN ;SET UP FOR POWER FAIL BEFORE
1604 ;TESTING STARTS
1605 001664 013746 000006 MOV #6,-(SP) ;SAVE CURRENT VECTORS
1606 001670 013746 000004 MOV #4,-(SP)
1607 001674 012737 001710 000004 MOV #12,#24 ;SETUP FOR TIMEOUT
1608 001702 005777 177274 TST #SWR ;REFERENCE HARDWARE SWITCH REG
1609 001706 000407 BR 13$ ;BR IF IT EXISTS
1610 001710 012737 000176 001202 12$: MOV #SWREG,SWR ;POINT TO SOFT SWR
1611 001716 012737 000174 001200 MOV #DISPREG,DISPLAY ;POINT TO SOFT DISPLAY REG
1612 001724 022626 CMP (SP)+,(SP)+ ;ADJUST STACK
1613 001726 012637 000004 13$: MOV (SP)+,#4 ;RESTORE VECTORS
1614 001732 012637 000006 MOV (SP)+,#6
1615 001736 105737 001340 TSTB INIFLG ;HAS INITIALIZATION BEEN PERFORMED
1616 001742 001401 BEQ 11$
1617 001744 000410 BR 6$
1618 001746 022737 003104 000042 11$: CMP #SENDAD,#42 ;IF ACT-11 AUTO MODE,
1619 001754 001404 BEQ 6$ ;DON'T TYPE ID
1620 001756 104402 001000 TYPE #MTITLE ;TYPE TITLE MESSAGE
1621 001762 105137 001340 COMB INIFLG ;IF NOT SET FLAG AND DO
1622 001766 105777 177210 6$: TSTB #SWR ;BIT7=1??
1623 001772 100002 BPL 10$
1624 001774 000137 002520 JMP 1$
1625 002000
1626 002000 032777 000001 177174 10$: BIT #SW00,#SWR ;ENTER PARAMETERS
1627 002006 001002 BNE .+6 ;YES
1628 002010 000137 002360 JMP 21$ ;NO
1629 002014 105137 001332 COMB FLAG
1630 002020 112737 000001 001340 MOV# #1,INIFLG ;SET TO MANUAL ENTRY
1631 002026 012700 001500 MOV #DUP.MAP,RO ;CLR MAP
1632 002032 005020 CLR (RO)+
1633 002034 020027 001560 68$: CMP RO,#DUP.END ;DONE WITH MAP?
1634 002040 001374 BNE 68$ ;BR IF NO
1635 002042 104403 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1636 002044 005474 MCSR ;MESSAGE
1637 002046 104405 PARAM ;CONVERT STRING
1638 002050 160000 ;LOW LIMIT

```

PROGRAM INITIALIZATION AND START UP.

1639	002052	175500			175500	:HIGH LIMIT
1640	002054	001500			DUPCRO	:STORE AT THIS LOCATION
1641	002056	001			.BYTE 1	:MASK
1642	002057	001			.BYTE 1	:HOW MANY TIMES + 2
1643	002060	104403			INSTR	:OUTPUT MESSAGE & GET INPUT STRING
1644	002062	005513			MVEC	:MESSAGE
1645	002064	104405			PARAM	:CONVERT STRING
1646	002066	000300			300	:LOW LIMIT
1647	002070	000770			770	:HIGH LIMIT
1648	002072	001502			DUPTRO	:STORE AT THIS LOCATION
1649	002074	001			.BYTE 1	:MASK
1650	002075	001			.BYTE 1	:HOW MANY TIMES + 2
1651	002076	104403			INSTR	:OUTPUT MESSAGE & GET INPUT STRING
1652	002100	005703			MPAR	:MESSAGE
1653	002102	104405			PARAM	:CONVERT STRING
1654	002104	000004			4	:LOW LIMIT
1655	002106	000007			7	:HIGH LIMIT
1656	002110	001240			TEMP2	:STORE AT THIS LOCATION
1657	002112	000			.BYTE 0	:MASK
1658	002113	001			.BYTE 1	:HOW MANY TIMES + 2
1659	002114	013737	001240	001320	MOV	TEMP2,PRTY :SAVE PRIORITY
1660	002122	104403			INSTR	:OUTPUT MESSAGE & GET INPUT STRING
1661	002124	005650			MTOTAL	:MESSAGE
1662	002126	104405			PARAM	:CONVERT STRING
1663	002130	000001			1	:LOW LIMIT
1664	002132	000010			8.	:HIGH LIMIT
1665	002134	001236			TEMP1	:STORE AT THIS LOCATION
1666	002136	000			.BYTE 0	:MASK
1667	002137	001			.BYTE 1	:HOW MANY TIMES + 2
1668	002140	104403			INSTR	:OUTPUT MESSAGE & GET INPUT STRING
1669	002142	005526			MJMPR	:MESSAGE
1670	002144	104413			SETFLG	:SET FLAG BASED UPON INPUT STRING
1671	002146	001323			OPCLRJ	:THIS FLAG
1672	002150	104403			INSTR	:OUTPUT MESSAGE & GET INPUT STRING
1673	002152	005601			MTCN	:MESSAGE
1674	002154	104413			SETFLG	:SET FLAG BASED UPON INPUT STRING
1675	002156	001322			TCNFLG	:THIS FLAG
1676	002167	105737	001322		TSTB	TCNFLG
1677	002164	001410			BEQ	71\$
1678	002166	104403			INSTR	:OUTPUT MESSAGE & GET INPUT STRING
1679	002170	005727			MSTJM	:MESSAGE
1680	002172	104413			SETFLG	:SET FLAG BASED UPON INPUT STRING
1681	002174	001334			STJMFL	:THIS FLAG
1682	002176	104403			INSTR	:OUTPUT MESSAGE & GET INPUT STRING
1683	002200	005762			MSRJM	:MESSAGE
1684	002202	104413			SETFLG	:SET FLAG BASED UPON INPUT STRING
1685	002204	001336			SRJMFL	:THIS FLAG
1686	002206	105737	001323		TSTB	OPCLRJ
1687	002212	001403			BEQ	69\$
1688	002214	052737	100000	001504	BIS	#BIT15,DUPO.A
1689	002222	105737	001322		TSTB	TCNFLG
1690	002226	001403			BEQ	70\$
1691	002230	052737	040000	001504	BIS	#BIT14,DUPO.A
1692	002236	112737	000001	001312	MOV B	#1,SAVACT
1693	002244	113737	001236	001311	MOV B	TEMP1,DUPNUM
1694	002252	113737	001236	001313	MOV B	TEMP1,SAVNUM

PROGRAM INITIALIZATION AND START UP.

1695	002260	005337	001236	65\$:	DEC	TEMP1			
1696	002264	001404			BEQ	64\$			
1697	002266	000261			SEC				
1698	002270	106137	001312		ROLB	SAVACT			
1699	002274	000771			BR	65\$			
1700	002276	113737	001312	001240	64\$:	MOVB	SAVACT,TEMP2	;# OF TIMES	
1701	002304	113737	001312	001310		MOVB	SAVACT,DUPACTV		
1702	002312	000241			CLC				
1703	002314	106037	001240		RORB	TEMP2			
1704	002320	012700	001500		MOV	#DUPCRO,RO			
1705	002324	012701	001506		MOV	#DUPCRI,R1			
1706	002330	000241			67\$:	CLC			
1707	002332	106037	001240		RORB	TEMP2			
1708	002336	103051			BCC	66\$			
1709	002340	012011			MOV	(RO)+,(R1)			
1710	002342	062721	000010		ADD	#10,(R1)+		;CSR	
1711	002346	012011			MOV	(RO)+,(R1)			
1712	002350	062721	000010		ADD	#10,(R1)+		;VECTOR	
1713	002354	012021			MOV	(RO)+,(R1)+		;PARAMETERS	
1714	002356	000764			BR	67\$			
1715	002360	012700	001500	21\$:	MOV	#DUP.MAP,RO		;SETUP TO CLEAR MAP	
1716	002364	005020		20\$:	CLR	(RO)+		;CLEAR	
1717	002366	020027	001560		CMP	RO,#DUP.END		;CHECK FOR FINISH	
1718	002372	001374			BNE	20\$;BR IF MORE TO GO	
1719	002374	012700	001500		MOV	#DUP.MAP,RO		;SETUP TO DEFAULT	
1720	002400	012710	160050		MOV	#160050,(RO)		;LOAD CSR	
1721	002404	012760	000770	000002	MOV	#770,2(RO)		;LOAD VECTOR	
1722	002412	012760	140026	000004	MOV	#140026,4(RO)		;LOAD PARAMETERS AND SYNC	
1723	002420	112737	000005	001320	MOVB	#5,PRIORITY		;LOAD PRIORITY	
1724	002426	012700	000001		MOV	#1,RO		;SAVE CORE THIS WAY	
1725	002432	110037	001310		MOVB	RO,DUPACTV		;PRESET PROGRAM CONTROLS	
1726	002436	110037	001311		MOVB	RO,DUPNUM		;DITTO	
1727	002442	110037	001312		MOVB	RO,SAVACT		;DITTO	
1728	002446	110037	001313		MOVB	RO,SAVNUM		;DITTO	
1729	002452	110037	001322		MOVB	RO,TCNFLAG		;DITTO	
1730	002456	110037	001323		MOVB	RO,OPCLRJ		;DITTO	
1731	002462				66\$:				
1732	002462	104402	006015	16\$:	TYPE	,XHEAD		;TYPE HEADER	
1733	002466	012737	001500	001236	MOV	#DUP.MAP,TEMP1		;SET POINTER	
1734	002474	017737	176536	001240	5\$:	MOV	TEMP1,TEMP2	;SET DATA	
1735	002502	001406			BEQ	1\$;ALL DONE WITH DATA	
1736	002504	104410			CONVRT				
1737	002506	006044			XSTATQ				
1738	002510	062737	000002	001236	ADD	#2,TEMP1		;UPDATE POINTER	
1739	002516	000766			BR	5\$			
1740	002520	032777	000001	176454	1\$:	BIT	#SW00,ASWR		
1741	002526	001405			BEQ	7\$			
1742	002530	005737	001332		TST	FLAG			
1743	002534	001002			BNE	7\$			
1744	002536	000137	002000		JMP	10\$			
1745	002542	005037	001332		7\$:	CLR	FLAG		
1746	002546	005737	000042		TST	#42		;IS PROGRAM RUNNING UNDER MONITOR	
1747	002552	001030			BNE	3\$;BR IF YES	
1748	002554	032777	000010	176420	BIT	#SW03,ASWR		;SELECT SPECIFIC DEVICES??	
1749	002562	001424			BEQ	3\$;BR IF NO.	
1750	002564	104402	005414		TYPE	,MNEW		;TYPE THE MESSAGE.	

1751	002570	005000		CLR	RO							:ZERO DATA LIGHTS
1752	002572	000000		HALT								:WAIT FOR USER TO TELL WHAT DEVICES TO RUN
1753	002574	127737	176402	001312	CMPB	JSWR, SAVACT						:IS THE NUMBER VALID?
1754	002602	101404			BLOS	2\$:BR IF NUMBER IS OK.
1755	002604	104402	005255		TYPE	,MERR3						:TELL USER OF INVALID NUMBER.
1756	002610	000000			HALT							:STOP EVERY THING.
1757	002612	000776			BR	.-2						:RESTART THE PROGRAM AGAIN.
1758	002614	117737	176362	001310	2\$:	MOV8	JSWR, DUPACTV					:GET NEW DEVICE PATTERN
1759	002622	113700	001310		MOV8	DUPACTV, RO						:SHOW THE USER WHAT HE SELECTED.
1760	002626	042700	177400		BIC	#1C<377>, RO						:USE ONLY LOW BYTE.
1761	002632	000000			HALT							:CONTINUE DYNAMIC SWITCHES.
1762	002634	012700	000300		3\$:	MOV	#300, RO					:PREPARE TO CLEAR THE FLOATING
1763	002640	012701	000302		MOV	#302, R1						:VECTOR AREA. 300-776
1764	002644	010120			4\$:	MOV	R1 (RO)+					:START PUTTING "PC+2 - HALT"
1765	002646	005021			CLR	(R1)+						:IN VECTOR AREA.
1766	002650	022021			CMP	(RO)+, (R1)+						:POP POINTERS
1767	002652	022700	001000		CMP	#1000, RO						:ALL DONE??
1768	002656	001372			BNE	4\$:BR IF NO.
1769												
1770												
1771												
1772												
1773	002660	012737	000340	177776	.BEGIN:	MOV	#340, PS					:LOCK OUT INTERRUPTS
1774	002666	012706	001150		MOV	#STACK, SP						:SET UP STACK
1775	002672	005737	000042		TST	JS42						:IS PROGRAM UNDER MONITOR CONTROL
1776	002676	001023			BNE	2\$:BR IF YES
1777	002700	032777	000004	176274	BIT	#BIT2, JSWR						:CHECK FOR LOCK ON TEST
1778	002706	001411			BEQ	1\$:BR IF NO LOCK DESIRED.
1779	002710	104402	005313		TYPE	,MLOCK						:TYPE LOCK SELECTED.
1780	002714	012737	000240	003174	MOV	#NOP, TTST						:ADJUST SCOPE ROUTINE.
1781	002722	012737	000240	003176	MOV	#NOP, TTST+2						:SET UP TO LOCK
1782	002730	000406			BR	2\$:CONTINUE ALONG.
1783	002732	013737	003306	003174	1\$:	MOV	BRW, TTST					:PREPARE NORMAL SCOPE ROUTINE
1784	002740	013737	003310	003176	MOV	BRX, TTST+2						:LOCK NOT SELECTED, SET UP FOR NORMAL SCOPE LOOP
1785	002746	012737	006226	001214	2\$:	MOV	#CYCLE, RETURN					:START AT "CYCLE" FIND WHICH DEVICE TO TEST
1786	002754	104402	005203		TYPE	MR						:TYPE R
1787	002760	000177	176230		JMP	RETURN						:START TESTING

:TEST START AND RESTART
 :-----

```

1788 ;END OF PASS
1789 ;TYPE NAME OF TEST
1790 ;UPDATE PASS COUNT
1791 ;CHECK FOR EXIT TO ACT-11
1792 ;RESTART TEST
1793
1794 002764 005037 001234 .EOP: CLR LSTERR ;CLEAR LAST ERROR PC
1795 002770 105037 001341 CLRFB ERRFLG ;CLEAR ERROR FLAG
1796 002774 005237 001230 INC PASCNT ;UPDATE PASS COUNT
1797 003000 013777 001230 176172 MOV PASCNT, @DISPLAY ;DISPLAY PASS COUNT
1798 003006 104402 005157 TYPE ,MEPASS ;TYPE END PASS
1799 003012 104402 005342 TYPE ,MCSRX ;TYPE CSR
1800 003016 104411 003130 CNVRT ,XCSR ;SHOW IT
1801 003022 104402 005350 TYPE ,MVECX ;TYPE VECTOR
1802 003026 104411 003136 CNVRT ,XVEC ;SHOW IT
1803 003032 104402 005356 TYPE ,MPASSX ;TYPE PASSES
1804 003036 104411 003144 CNVRT ,XPASS ;SHOW IT
1805 003042 104402 005367 TYPE ,MERRX ;TYPE ERRORS
1806 003046 104411 003152 CNVRT ,XERR ;SHOW IT
1807 003052 105337 001313 DECB $AVNUM ;ARE ALL DEVICES TESTED?
1808 003056 001017 BNE RESTRT ;BR IF NO.
1809 003060 112737 000377 001343 MOVB #377, QV, FLG ;SET THE QUICK VERIFY FLAG.
1810 003066 113737 001311 001313 MOVB DUPNUM, $AVNUM ;RESTORE THE COUNT
1811 003074 013701 000042 MOV @#42, R1 ;CHECK FOR ACT-11 OR DDP
1812 003100 001406 BEQ RESTRt ;IF NOT, CONTINUE TESTING
1813 003102 000005 RESET ;STOP THE SHOW--CLEAR THE WORLD
1814 003104
1815 003104 004711 $ENDAD: JSR PC, (R1)
1816 003106 000240 NOP
1817 003110 000240 NOP
1818 003112 000240 NOP
1819 003114 000240 NOP
1820 003116 012737 006226 001214 RESTRt: MOV #CYCLE, RETURN
1821 003124 000137 006226 JMP CYCLE
1822 003130 000001 XCSR: 1
1823 003132 006 002 .BYTE 6,2
1824 003134 001404 RXCSR
1825 003136 000001 XVEC: 1
1826 003140 003 002 .BYTE 3,2
1827 003142 001374 DUPRVC
1828 003144 000001 XPASS: 1
1829 003146 006 002 .BYTE 6,2
1830 003150 001230 PASCNT
1831 003152 000001 XERR: 1
1832 003154 006 002 .BYTE 6,2
1833 003156 001232 ERRCNT
1834
1835 ;SCOPE LOOP AND ITERATION HANDLER
1836
1837 003160 005037 001234 .SCOPE: CLR LSTERR ;CLEAR LAST ERROR PC
1838 003164 010016 MOV RO, (SP) ;SAVE RO ON STACK
1839 003166 032777 040000 176006 BIT #BIT14, @SWR ;LOOP ON TEST?
1840 003174 001407 TTST: BEQ 1$ ;BR IF NO (IF LOCK SW01 = 1; THIS LOCATION = 240)
1841 003176 000437 BR 3$ ;GO TO 3$ (DITTO)
1842 003200 105777 176000 TSTB @TKCSR ;KYBD DONE?
1843 003204 100034 BPL 3$ ;BR IF NO (LOCK: HIT A KEY ON TTY TO GO TO NEXT TEST)

```

DZDPBB MACY11 27(1006) 18-MAY-77 00:00 PAGE 46
 DZDPBB.P11 13-MAY-77 15:27 END OF PASS ROUTINE

```

1844 003206 017700 175774      MOV      @TKDBR,R0      ;CLR DONE BIT
1845 003212 000415              BR      2$            ;CONTINUE
1846 003214 032777 004000 175760 1$:  BIT      #SW11,@SWR    ;DELETE ITERATION (QUICK PASS)?
1847 003222 001011              BNE     2$            ;BR IF YES
1848 003224 105737 001343      TSTB   QV.FLG        ;HAS FIRST PASS BEEN COMPLETED?
1849 003230 001406              BEQ     2$            ;BR IF QUICK VERIFY
1850 003232 005237 001224      INC     LPCNT         ;UPDATE ITERATION COUNTER
1851 003236 023737 001224 001222  CMP     LPCNT,ICOUNT  ;ALL ITERATIONS DONE?
1852 003244 001014              BNE     3$            ;BR IF NOT YET
1853 003246 105037 001341      CLRB   ERRFLG        ;PREPARE FOR NEW TEST
1854 003252 005037 001224      CLR     LPCNT         ;START ICOUNT AT ZERO
1855 003256 005037 001220      CLR     LOCK
1856 003262 012737 000050 001222  MOV     #50,ICOUNT    ;RESET ITERATIONS
1857 003270 013737 001216 001214  MOV     NEXT,RETURN   ;GET NEXT TEST
1858 003276 011600      3$:  MOV     (SP),R0      ;POP R0 OFF STACK
1859 003300 022626      POP2SP ;FAKE AN RTI
1860 003302 000177 175706      JMP     @RETURN      ;GO DO THE TEST
1861 003306 001407      BRW:   1407
1862 003310 000437      BRX:   437
1863
1864      ;CHECK FOR FREEZE ON CURRENT DATA
1865      ;-----
1866
1867 003312 032777 001000 175662 .SCOP1: BIT      #SW09,@SWR    ;IS SW09=1(SET)?
1868 003320 001405              BEQ     1$            ;BR IF NOT SET.
1869 003322 005737 001220      TST    LOCK
1870 003326 001402              BEQ     1$
1871 003330 013716 001220      MOV     LOCK,(SP)    ;GOTO THE ADDRESS IN LOCK.
1872 003334 000002      1$:  RTI              ;GO BACK.
1873
1874      ;TELETYPE OUTPUT ROUTINE
1875      ;-----
1876
1877 003336 010546      .TYPE: MOV     R5,-(SP)    ;SAVE R5 ON THE STACK.
1878 003340 017605 000002      MOV     @2(SP),R5    ;GET ADDRESS OF MESSAGE.
1879 003344 062766 000002 000002  ADD     #2,2(SP)     ;POP OVER ADDRESS.
1880 003352 032777 010000 175622 1$:  BIT      #SW12,@SWR  ;INHIBIT ALL PRINT OUT??
1881 003360 001012              BNE     3$            ;BR IF NO PRINT OUT WANTED (SW12=1)
1882 003362 105715              TSTB   (R5)          ;IS NUMBER MINUS? (MSB=1(BIT7))
1883 003364 100002              BPL     2$            ;BR IF NUMBER IS PLUS
1884 003366 104402 005136      TYPE   MCRLF         ;TYPE A CR/LF!
1885 003372 105777 175612      2$:  TSTB   @TPCSR     ;TTY READY?
1886 003376 100375              BPL     2$            ;BR IF NO.
1887 003400 112577 175606      MOVB   (R5)+,@TPDBR ;PRINT CURRENT CHAR.
1888 003404 001362              BNE     1$            ;IF NOT ZERO KEEP PRINTING!
1889 003406 012605      3$:  MOV     (SP)+,R5    ;END OF OUTPUT. RESTORE R5
1890 003410 000002      RTI              ;GO HOME
1891
1892
1893 003412 010346      .INSTR: MOV     R3,-(SP) ;SAVE R3 ON STACK
1894 003414 010446      MOV     R4,-(SP)    ;SAVE R4 ON STACK
1895 003416 017637 000004 003434      MOV     @4(SP),.MSG
1896 003424 062766 000002 000004      ADD     #2,4(SP)
1897 003432 104402      .INST1: TYPE
1898 003434 000000      .MSG:  0
1899 003436 012704 006162      MOV     #INBUF,R4
    
```

1900	003442	012703	000007		MOV	#7,R3	
1901	003446	105777	175532	1\$:	TSTB	@TKCSR	
1902	003452	100375			BPL	1\$	
1903	003454	117714	175526		MOVB	@TKDBR,(R4)	
1904	003460	142714	000200		BICB	#200,(R4)	
1905	003464	122427	000015		CMPB	(R4)+,#15	
1906	003470	001417			BEQ	INSTR2	
1907	003472	105777	175512	2\$:	TSTB	@TPCSR	
1908	003476	100375			BPL	2\$	
1909	003500	017777	175502	175504	MOV	@TKDBR,@TPDBR	
1910	003506	005303			DEC	R3	
1911	003510	001356			BNE	1\$	
1912	003512	012604			MOV	(SP)+,R4	
1913	003514	012603			MOV	(SP)+,R3	
1914	003516	010346		.INSTE:	MOV	R3,-(SP)	
1915	003520	010446			MOV	R4,-(SP)	
1916	003522	104402	005132		TYPE	,MQM	
1917	003526	000741			BR	:INST1	
1918	003530	012604		INSTR2:	MOV	(SP)+,R4	:RESTORE R4
1919	003532	012603			MOV	(SP)+,R3	:RESTORE R3
1920	003534	000002			RTI		
1921							
1922							
1923							
1924							
1925	003536	010546		.PARAM:	MOV	R5,-(SP)	
1926	003540	010446			MOV	R4,-(SP)	
1927	003542	016605	000004		MOV	4(SP),R5	
1928	003546	012537	003726		MOV	(R5)+,LOLIM	
1929	003552	012537	003730		MOV	(R5)+,HILIM	
1930	003556	012537	003732		MOV	(R5)+,DEVADR	
1931	003562	112537	003734		MOVB	(R5)+,LOBITS	
1932	003566	112537	003735		MOVB	(R5)+,ADRCNT	
1933	003572	010566	000004		MOV	R5,4(SP)	
1934	003576	005005		PARAM1:	CLR	R5	
1935	003600	012704	006162		MOV	#INBUF,R4	
1936	003604	122714	000015		CMPB	#15,(R4)	
1937	003610	001420			BEQ	PARERR	
1938	003612	121427	000060	1\$:	CMPB	(R4),#60	
1939	003616	002415			BLT	PARERR	
1940	003620	121427	000067		CMPB	(R4),#67	
1941	003624	003012			BGT	PARERR	
1942	003626	142714	000060		BICB	#60,(R4)	
1943	003632	152405			BISB	(R4)+,R5	
1944	003634	122714	000015		CMPB	#15,(R4)	
1945	003640	001406			BEQ	LIMITS	
1946	003642	006305			ASL	R5	
1947	003644	006305			ASL	R5	
1948	003646	006305			ASL	R5	
1949	003650	000760			BR	1\$	
1950	003652	104404		PARERR:	INSTR		
1951	003654	000750			BR	PARAM1	
1952							
1953							
1954							
1955							

; CONVERT ASCII STRING TO OCTAL

 ; TEST TO SEE IF NUMBER IS WITHIN LIMITS

```

1956 003656 020537 003730 LIMITS: CMP R5,HILIM
1957 003662 101373 BHI PARERR
1958 003664 020537 003726 CMP R5,LOLIM
1959 003670 103770 BLO PARERR
1960 003672 133705 003734 BITB LOBITS,R5
1961 003676 001365 BNE PARERR
1962
1963 ;STORE NUMBER AT SPECIFIED ADDRESS
1964
1965 003700 013704 003732 1$: MOV DEVADR,R4
1966 003704 010524 MOV R5,(R4)+
1967 003706 062705 000002 ADD #2,R5
1968 003712 105337 003735 DECB ADRCNT
1969 003716 001372 BNE 1$
1970 003720 012604 MOV (SP)+,R4
1971 003722 012605 MOV (SP)+,R5
1972 003724 000002 RTI
1973 003726 000000 LOLIM: 0
1974 003730 000000 HILIM: 0
1975 003732 000000 DEVADR: 0
1976 003734 000000 LOBITS: 0
1977 003735 003735 ADRCNT=LOBITS+1
1978
1979 ;SAVE PC OF TEST THAT FAILED AND R0-R5
1980 -----
1981
1982 003736 016637 000004 001266 .SAV05: MOV 4(SP),SAVPC ;SAVE R7 (PC)
1983
1984 ;SAVE R0-R5
1985
1986 003744 010537 001262 SV05: MOV R5,SAVR5 ;SAVE R5
1987 003750 010437 001260 MOV R4,SAVR4 ;SAVE R4
1988 003754 010337 001256 MOV R3,SAVR3 ;SAVE R3
1989 003760 010237 001254 MOV R2,SAVR2 ;SAVE R2
1990 003764 010137 001252 MOV R1,SAVR1 ;SAVE R1
1991 003770 010037 001250 MOV R0,SAVR0 ;SAVE R0
1992 003774 000002 RTI ;LEAVE.
1993
1994 ;RESTORE R0-R5
1995
1996 003776 013700 001250 .RES05: MOV SAVR0,R0 ;RESTORE R0
1997 004002 013701 001252 MOV SAVR1,R1 ;RESTORE R1
1998 004006 013702 001254 MOV SAVR2,R2 ;RESTORE R2
1999 004012 013703 001256 MOV SAVR3,R3 ;RESTORE R3
2000 004016 013704 001260 MOV SAVR4,R4 ;RESTORE R4
2001 004022 013705 001262 MOV SAVR5,R5 ;RESTORE R5
2002 004026 000002 RTI ;LEAVE
2003
2004
2005 ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
2006 -----
2007
2008 004030 104402 005136 .CONVR: TYPE MCRLF
2009 004034 010046 .CNVRT: MOV R0,-(SP)
2010 004036 010146 MOV R1,-(SP)
2011 004040 010346 MOV R3,-(SP)

```

2012	004042	010446			MOV	R4,-(SP)
2013	004044	010546			MOV	R5,-(SP)
2014	004046	017601	000012		MOV	2(2(SP),R1
2015	004052	062766	000002	000012	ADD	#2,12(SP)
2016	004060	012137	004234		MOV	(R1)+,WRDCNT
2017	004064	112137	004236	1\$:	MOVB	(R1)+,CHRCNT
2018	004070	112137	004237		MOVB	(R1)+,SPACNT
2019	004074	013137	004240		MOV	2(R1)+,BINWRD
2020	004100	013704	004240	2\$:	MOV	BINWRD,R4
2021	004104	113705	004236		MOVB	CHRCNT,R5
2022	004110	012700	006056		MOV	#TEMP,R0
2023	004114	010403		3\$:	MOV	R4,R3
2024	004116	042703	177770		BIC	#177770,R3
2025	004122	062703	000060		ADD	#060,R3
2026	004126	110320			MOVB	R3,(R0)+
2027	004130	000241			CLC	
2028	004132	006004			ROR	R4
2029	004134	000241			CLC	
2030	004136	006004			ROR	R4
2031	004140	000241			CLC	
2032	004142	006004			ROR	R4
2033	004144	005305			DEC	R5
2034	004146	001362			BNE	3\$
2035	004150	012703	006120		MOV	#MDATA,R3
2036	004154	114023		4\$:	MOVB	-(R0),(R3)+
2037	004156	105337	004236		DECB	CHRCNT
2038	004162	001374			BNE	4\$
2039	004164	105737	004237		TSTB	SPACNT
2040	004170	001405			BEG	6\$
2041	004172	112723	000040	5\$:	MOVB	#040,(R3)+
2042	004176	105337	004237		DECB	SPACNT
2043	004202	001373			BNE	5\$
2044	004204	105013		6\$:	CLRB	(R3)
2045	004206	104402	006120		TYPE	,MDATA
2046	004212	005337	004234		DEC	WRDCNT
2047	004216	001322			BNE	1\$
2048	004220	012605			MOV	(SP)+,R5
2049	004222	012604			MOV	(SP)+,R4
2050	004224	012603			MOV	(SP)+,R3
2051	004226	012601			MOV	(SP)+,R1
2052	004230	012600			MOV	(SP)+,R0
2053	004232	000002			RTI	
2054	004234	000000			WRDCNT:	0
2055	004236	000000			CHRCNT:	0
2056		004237			SPACNT=	CHRCNT+1
2057	004240	000000			BINWRD:	0
2058						
2059						
2060						
2061						
2062						
2063						
2064						
2065	004242	017605	000000		.SETFLG:	MOV 2(SP),R5
2066	004246	042737	000040	006162	BIC	#40,INBUF
2067	004254	122737	000116	006162	CMPB	#'N',INBUF ; IS IT "N" ?

```

;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
;BUFFER TO THE CHARACTERS "N" AND "Y"
;IF THE CHARACTER IS "N" CLEAR THE FLAG
;IF THE CHARACTER IS "Y" SET THE FLAG

```

```

2068 004262 001002      BNE      1$
2069 004264 105015      CLRB    (R5) ;000
2070 004266 000406      BR      2$
2071 004270 122737 000131 006162 1$:  CMPB   #'Y,INBUF ;IS IT "Y" ?
2072 004276 001005      BNE     3$
2073 004300 112715 177777      MOVB   #-1,(R5) ;377
2074 004304 062716 000002      2$:  ADD    #2,(SP)
2075 004310 000002      RTI
2076 004312 104404      3$:  INSTER ;RETRY
2077 004314 000752      BR     .SETFLG
2078
2079
2080
2081      ;TRAP DISPATCH SERVICE
2082      ;ARGUMENT OF TRAP IS EXTRACTED
2083      ;AND USED AS OFFSET TO OBTAIN POINTER
2084      ;TO SELECTED SUBROUTINE
2085 004316 011646      .TRPSR: MOV   (SP),-(SP) ;GET PC OF RETURN
2086 004320 162716 000002      SUB   #2,(SP) ;=PC OF TRAP
2087 004324 017616 000000      MOV   @2(SP),(SP) ;GET TRP
2088 004330 006316      TRPOK: ASL   (SP) ;MULTIPLY TRAP ARG BY 2
2089 004332 042716 177001      BIC   #177001,(SP) ;CLEAR UNWANTED BITS
2090 004336 062716 001344      ADD   #.TRPTAB,(SP) ;POINTER TO SUBROUTINE ADDRESS
2091 004342 017616 000000      MOV   @2(SP),(SP) ;SUBROUTINE ADDRESS
2092 004346 000136      JMP   @2(SP)+ ;GO TO SUBROUTINE
2093
2094      ;ERROR HANDLER
2095      ;-----
2096
2097 004350 032777 010000 174624 .HLT:  BIT   #SW12,@SWR ;BELL ON ERROR?
2098 004356 001406      BEQ   XBX ;BR IF NO BELL
2099 004360 105777 174624      TSTB @TPCSR ;TTY READY
2100 004364 100003      BPL   XBX ;DON'T WAIT IF TTY NOT READY.
2101 004366 112777 000207 174616      MOVB  #207,@TPDBR ;PUSH A BELL AT THE TTY.
2102 004374 032777 020000 174600 XBX:  BIT   #SW13,@SWR ;DELETE ERROR PRINT OUT?
2103 004402 001105      BNE   HALTS ;BR IF NO PRINT OUT WANTED.
2104 004404 021637 001234      CMP   (SP),LSTERR ;WAS THIS ERROR FOUND LAST TIME?
2105 004410 001404      BEQ   1$ ;BR IF YES
2106 004412 011637 001234      MOV   (SP),LSTERR ;RECORD BEING HERE
2107 004416 105037 001341      CLRB  ERRFLG ;PREPARE HEADER
2108 004422 104406      1$:  SAVOS ;SAVE ALL PROC REGISTERS
2109 004424 011605      MOV   (SP),R5 ;GET THE PC OF ERROR
2110 004426 162705 000002      SUB   #2,R5 ;GET ADDRESS OF TRAP CALL
2111 004432 011504      MOV   (R5),R4 ;GET HLT INSTRUCTION
2112 004434 006304      ASL   R4 ;MULT BY TWO
2113 004436 061504      ADD   (R5),R4 ;DOUBLE IT
2114 004440 006304      ASL   R4 ;MULT AGAIN
2115 004442 042704 177001      BIC   #177001,R4 ;CLEAR JUNK
2116 004446 062704 027732      ADD   #.ERRTAB,R4 ;GET POINTER
2117 004452 012437 004566      MOV   (R4)+,ERRMSG ;GET ERROR MESSAGE
2118 004456 012437 004600      MOV   (R4)+,DATAHD ;GET DATA HEADREER
2119 004462 011437 004612      MOV   (R4),DATABP ;GET DATA TABLE
2120 004466 105737 001341      TSTB  ERRFLG ;TYPE HEADREER
2121 004472 001403      BEQ   TYPMSG ;BR IF YES
2122 004474 005737 004612      TST   DATABP ;DOES DATA TABLE EXIST?
2123 004500 001040      BNE   TYPDAT ;BR IF YES.
  
```

2124	004502	104402	005136		TYPMSG:	TYPE	,MCRLF	
2125	004506	104402	005136			TYPE	,MCRLF	
2126	004512	005737	001220			TST	LOCK	
2127	004516	001402				BEQ	1\$	
2128	004520	104402	005412			TYPE	,MASTEK	
2129	004524	104402	005400		1\$:	TYPE	,MTSTN	
2130	004530	104411	005000			CONVRT	,XTSTN	;SHOW IT
2131	004534	104402	005467			TYPE	,MERRPC	;TYPE PC.
2132	004540	104411	004772			CONVRT	,ERTABO	;SHOW IT
2133	004544	104402	005136			TYPE	,MCRLF	;GIVE A CR/LF
2134	004550	112737	177777	001341		MOVB	#-1,ERRFLG	;NO MORE HEADER UNLESS NO DATA TABLE.
2135	004556	005737	004566			TST	ERRMSG	;IS THERE AN ERROR MESSAGE?
2136	004562	001402				BEQ	WRKO.FM	;BR IF NO.
2137	004564	104402				TYPE		;TYPE
2138	004566	000000			ERRMSG:	0		;ERROR MESSAGE
2139	004570				WRKO.FM:			
2140	004570	005737	004600			TST	DATAHD	;DATA HEADER?
2141	004574	001402				BEQ	TYPDAT	;BR IF NO
2142	004576	104402				TYPE		;TYPE
2143	004600	000000			DATAHD:	0		;DATA HEADER
2144	004602	005737	004612		TYPDAT:	TST	DATABP	;DATA TABLE?
2145	004606	001402				BEQ	RESREG	;BR IF NO.
2146	004610	104410				CONVRT		;SHOW
2147	004612	000000			DATABP:	0		;DATA TABLE
2148	004614	104407			RESREG:	RESOS		;RESTORE PROC REGISTERS
2149	004616	022737	003104	000042	HALTS:	CMP	#SENDAD,@#42	;IF ACT-11 AUTO MODE--HALT!!
2150	004624	001403				BEQ	1\$	
2151	004626	005777	174350			TST	@SWR	;HALT ON ERROR?
2152	004632	100035				BPL	EXITER	;BR IF NO HALT ON ERROR
2153	004634	010046			1\$:	PUSHRO		;SAVE RO
2154	004636	016600	000002			MOV	2(SP),RO	;SHOW ERROR PC IN DATA LIGHTS
2155	004642	013746	000004			MOV	4,-(SP)	;SAVE OLD TRAP
2156	004646	013746	000006			MOV	6,-(SP)	
2157	004652	012737	004710	000004		MOV	#22\$,4	;FORCE HALT IF TIME-OUT
2158	004660	012737	000340	000006		MOV	#340,6	;WHEN REFERENCING TXCSR
2159	004666	042777	014000	174516		BIC	#SYSTST!MEXT,@TXCSR	
2160	004674	000000				HALT		;HALT
2161	004676	012637	000006			MOV	(SP)+,6	;RESTORE TRAP
2162	004702	012637	000004			MOV	(SP)+,4	
2163	004706	000406				BR	33\$	
2164	004710	000000			22\$:	HALT		;HALT
2165	004712	022626				CMP	(SP)+,(SP)+	;POP STACK
2166	004714	012637	000006			MOV	(SP)+,6	;RESTORE TRAP
2167	004720	012637	000004			MOV	(SP)+,4	
2168	004724	012600			33\$:	POPPO		;GET RO
2169	004726	005237	001232		EXITER:	INC	ERRCNT	;UPDATE ERROR COUNT
2170	004732	032777	000400	174242		BIT	#SW08,@SWR	;GOTO TOP OF TEST?
2171	004740	001007				BNE	1\$;BR IF YES
2172	004742	032777	002000	174232		BIT	#SW10,@SWR	;GOTO NEXT TEST?
2173	004750	001407				BEQ	2\$;BR IF NO
2174	004752	013737	001216	001214		MOV	NEXT,RETURN	;SET FOR NEXT TEST
2175	004760	012706	001150		1\$:	MOV	#STACK,SP	;RESET SP
2176	004764	000177	174224			JMP	@RETURN	;GOTO SPECIFIED TEST
2177	004770	000002			2\$:	RTI		;RETURN
2178	004772	000001			ERTABO:	1		
2179	004774	006	002			.BYTE	6,2	

```

2180 004776 001266 SAVPC
2181 005000 000001 XTSTN: 1
2182 005002 003 002 .BYTE 3,2
2183 005004 001226 TSTNO
2184 005006 017600 000000 .PKCLK: MOV @ (SP), R0 ; GET THE # OF TICKS TO POKE
2185 005012 062716 000002 ADD #2, (SP) ; POP OVER THE #
2186 005016
2187 005016 052777 020000 174366 1$: BIS #CLK, @TXCSR ; POKE CLOCK UP
2188 005024 005300 DEC R0 ; ARE WE DONE?
2189 005026 001405 BEQ 2$ ; YES-GO TO 2$
2190 005030 042777 020000 174354 BIC #CLK, @TXCSR ; POKE CLOCK DOWN
2191 005036 005300 DEC R0 ; ARE WE DONE?
2192 005040 001366 BNE 1$ ; NO-REPEAT
2193 005042 000002 2$: RTI ; RETURN
2194
2195
2196 ; WAIT ROUTINE
2197 005044 000240 SMALL: NOP ; STALL
2198 005046 000207 RTS PC ; RETURN
2199
2200 ; POWER FAIL ROUTINE
2201
2202 005050 012737 005060 000024 .PFAIL: MOV #PWRUP, 24 ; LOAD PFAIL VECTOR FOR POWER UP
2203 005056 000000 HALT ;
2204 005060 000005 PWRUP: RESET ; WAIT TTY TO COME UP
2205 005062 012706 001150 MOV #STACK, SP ; REINIT STACK POINTER
2206 005066 012737 005050 000024 MOV #.PFAIL, 24 ; LOAD PFAIL VECTOR FOR POWER DOWN
2207 005074 104402 TYPE
2208 005076 005141 MPOWER
2209 005100 000177 174110 JMP @RETURN
2210 ; CLRVEC, ROUTINE TO FILL COMMUNICATION VECTOR AREA WITH .+2, HALT
2211
2212 005104 012702 000300 CLRVEC: MOV #300, R2 ; R2 COMM VECTOR AREA ADRS
2213 005110 012701 000302 MOV #302, R1 ; INIT R1 WITH ADRS OF HALT
2214 005114 010122 1$: MOV R1, (R2)+ ; MOV .+2 TO PC
2215 005116 005022 CLR (R2)+ ; MOV HALT TO PC
2216 005120 022121 CMP (R1)+, (R1)+ ; INC TO NEXT VECTOR AREA
2217 005122 022701 000776 CMP #776, R1 ; END OF VECTOR AREA
2218 005126 001372 BNE 1$ ; NO
2219 005130 000207 RTS PC ; RETURN
2220
2221
2222
2223 005132 020040 000077 MQM: .ASCIZ / ?/
(2) 005136 005015 000 MCRLF: .ASCIZ <15><12>
(2) 005141 377 053520 020122 MPOWER: .ASCIZ <377>/PWR FAILED. /
(2) 005157 015 042777 042116 MEPASS: .ASCIZ <15><377>/END PASS DZDPBB /
(2) 005203 377 000122 MR: .ASCIZ <377>/R/
(2) 005206 050377 047522 051107 MERR2: .ASCIZ <377>/PROGRAM INDICATES NO DEVICES PRESENT. /
(2) 005255 377 047111 052523 MERR3: .ASCIZ <377>/INSUFFICIENT DATA! /
(2) 005301 377 042524 052123 MTSTPC: .ASCIZ <377>/TEST PC- /
(2) 005313 377 047514 045503 MLOCK: .ASCIZ <377>/LOCK ON SELECTED TEST /
(2) 005342 051503 035122 000040 MCSRX: .ASCIZ /CSR: /
(2) 005350 042526 035103 000040 MVECX: .ASCIZ /VEC: /
(2) 005356 040520 051523 051505 MPASSX: .ASCIZ /PASSES: /
(2) 005367 105 051122 051117 MERRX: .ASCIZ /ERRORS: /
    
```

(2)	005400	042524	052123	047040	MTSTN: .ASCIZ	/TEST NO: /
(2)	005412	000052			MASTEK: .ASCIZ	/#/
(2)	005414	051777	052105	051440	MNEW: .ASCIZ	<377>/SET SWITCH REG TO DUP11'S DESIRED ACTIVE./
(2)	005467	120	035103	000040	MERRPC: .ASCIZ	/PC: /
(2)	005474	051377	041505	041440	MCSR: .ASCIZ	<377>/REC CSR ADRS /
(2)	005513	377	042526	020103	MVEC: .ASCIZ	<377>/VEC ADRS /
(2)	005526	044777	020123	044124	MJMPR: .ASCIZ	<377>/IS THE OPTIONAL CLR JMPR IN? (Y OR N) /
(2)	005601	377	051511	052040	MTCN: .ASCIZ	<377>/IS THE M325 CONNECTOR ON? (Y OR N) /
(2)	005650	021777	047440	020106	MTOTAL: .ASCIZ	<377>/# OF DUP'S (IN OCTAL) /
(2)	005703	377	051120	047511	MPAR: .ASCIZ	<377>/PRIORITY (4 TO 7) /
(2)	005727	377	042523	020103	MSTJM: .ASCIZ	<377>/SEC TX JMPR IN? (Y OR N) /
(2)	005762	051777	041505	051040	MSRJM: .ASCIZ	<377>/SEC RX JMPR IN? (Y OR N) /
(2)	006015	377	040515	020120	XHEAD: .ASCIZ	<377>/MAP OF DUP11 STATUS/<377>
(2)		006044			.EVEN	
(2)	006044	000002			XSTATQ: 2	
2224	006046	006	003		.BYTE	6,3
2225	006050	001236			TEMP1	
2226	006052	006	002		.BYTE	6,2
2227	006054	001240			TEMP2	
2228					.EVEN	
2229						
2230	006056	000000			TEMP: 0	
2231		006120			.=. +40	
2232	006120	000000			MDATA: 0	
2233		006162			.=. +40	
2234	006162	000000			INBUF: 0	
2235		006224			.=. +40	
2236	006224	000001			TRP.PC: .BLKW 1	
2237						

```

2238
2239
2240 ;ROUTINE USED TO "CYCLE" THROUGH UP TO EIGHT DUP11'S
2241 ;THIS ROUTINE SETS UP THE CONTROL ADDRESS FOR THE DIAGNOSTIC
2242 ;AND RUNS THE SPECIFIED DUP11'S. THIS ROUTINE *MUST*
2243 ;BE RUN FIRST BEFORE ENTERING THE DIAGNOSTIC FOR THE
2244 ;SETUP NECESSARY.
2245
2246
2247 006226 105737 001310 CYCLE: TSTB DUPACTV ;ARE ANY DUP11'S TO BE TESTED?
2248 006232 001004 BNE 1$ ;BR IF OK.
2249 006234 104402 005206 TYPE ,MERR2 ;NO DUP11'S SELECTED!!
2250 006240 000000 HALT ;STOP THE SHOW.
2251 006242 000776 BR -2 ;DISQUALIFY CONT. SW.
2252 006244 133737 001314 001310 1$: BITB RUN,DUPACTV ;IS THIS ONE "ACTIVE"
2253 006252 001020 BNE 2$ ;BR IF GOOD ONE FOUND.
2254 006254 000241 CLC ;CLEAR PROC. CARRY BIT.
2255 006256 106137 001314 ROLB RUN ;UPDATE POINTER
2256 006262 105537 001314 ADCB RUN ;CATCH CARRY FROM RUN
2257 006266 062737 000006 001316 ADD #6,CREAM ;UPDATE ADDRESS POINTER.
2258 006274 022737 001560 001316 CMP #DUP.END,CREAM
2259 006302 001360 BNE 1$ ;KEEP GOING; NOT ALL TESTED FOR.
2260 006304 012737 001500 001316 MOV #DUP.MAP,CREAM ;RESET ADDRESS POINTER.
2261 006312 000754 BR 1$ ;KEEP LOOKING FOR ACTIVE DUP11
2262 006314 000241 2$: CLC ;CLEAR PROC. CARRY.
2263 006316 106137 001314 ROLB RUN ;UPDATE POINTER.
2264 006322 105537 001314 ADCB RUN ;CATCH CARRY.
2265 006326 013700 001316 MOV CREAM,RO ;GET ADDRESS POINTER.
2266 006332 062737 000006 001316 ADD #6,CREAM ;UPDATE.
2267 006340 022737 001560 001316 CMP #DUP.END,CREAM
2268
2269 006346 001003 BNE 3$ ;ALL DONE?
2270 006350 012737 001500 001316 MOV #DUP.MAP,CREAM ;BR IF NO.
2271 006356 012037 001404 3$: MOV (RO)+,RXCSR ;RESTORE POINTER.
2272 006362 012037 001374 MOV (RO)+,DUPRVC ;LOAD SYSTEM CTRL. REG
2273 006366 012037 001434 MOV (RO)+,LOO.OO ;LOAD VECTOR
2274 006372 012700 000002 MOV #2,RO ;GET PARAMETERS
2275 006376 013737 001404 001424 MOV RXCSR,HUPRCR ;SAVE CORE THIS WAY!
2276 006404 005237 001424 INC HUPRCR ;GET CONTROL REG HIGH BYTE
2277 006410 013737 001424 001406 MOV HUPRCR,RXDBUF ;GOT IT
2278 006416 005237 001406 INC RXDBUF ;GET RX CONTROL REG BUFFER
2279 006422 013737 001406 001416 MOV RXDBUF,DUPSEC ;GOT IT
2280 006430 013737 001406 001410 MOV RXDBUF,PARCSR ;GOT SECONDARY REG SELECT REG
2281 006436 013737 001406 001422 MOV RXDBUF,HUPRBF ;GOT PARAMETER STATUS REGISTER
2282 006444 005237 001422 INC HUPRBF ;GET RX BUFFER HIGH BYTE
2283 006450 013737 001422 001420 MOV HUPRBF,HUPPSR ;GOT IT
2284 006456 013737 001420 001412 MOV HUPPSR,TXCSR ;GOT PAR STATUS REG HIGH BYTE
2285 006464 005237 001412 INC TXCSR ;GET TX CONTROL REGISTER
2286 006470 013737 001412 001430 MOV TXCSR,HUPTCR ;GOT IT
2287 006476 005237 001430 INC HUPTCR ;GET TX CONTROL REG HIGH BYTE
2288 006502 013737 001430 001414 MOV HUPTCR,TXDBUF ;GOT IT
2289 006510 005237 001414 INC TXDBUF ;BET TX BUFFER
2290 006514 013737 001414 001426 MOV TXDBUF,HUPTBF ;GOT IT
2291 006522 005237 001426 INC HUPTBF ;GET TX BUFFER HIGH BYTE
2292
2293 006526 013737 001374 001376 MOV DUPRVC,DUPRPS ;GOT IT
;RX VECTOR
    
```

2294	006534	060037	001376			ADD	RO,DUPRPS	;RX PRIORITY LEVEL
2295	006540	013737	001376	001400		MOV	DUPRPS,DUPTVC	
2296	006546	060037	001400			ADD	RO,DUPTVC	;TX VECTOR
2297	006552	013737	001400	001402		MOV	DUPTVC,DUPTPS	
2298	006560	060037	001402			ADD	RO,DUPTPS	;TX PRIORITY LEVEL
2299								
2300								
2301	006564	012700	001434			MOV	#L00.00,RO	;LOAD STAU8 00-00
2302	006570	012701	001432			MOV	#MASK.A,R1	;PREPARE MASK.
2303	006574	012702	001433			MOV	#CLK.A,R2	;PREPARE CLOCKS
2304	006600	004737	006744			JSR	PC,FIX.00	;GO AND CALCULATE CONFIGURATION.
2305	006604	005737	000042			TST	#42	
2306	006610	001050				BNE	4\$	
2307	006612	032777	000002	172362		BIT	#SW01,#SWR	;IF SW01=1,GET STARTING TEST #
2308	006620	001444				BEQ	4\$	
2309	006622	104402	005136		7\$:	TYPE	,MCRLF	
2310	006626	104403				INSTR	;OUTPUT MESSAGE & GET INPUT STRING	
2311	006630	005400				MTSTN	;MESSAGE	
2312	006632	104405				PARAM	;CONVERT STRING	
2313	006634	000001				1	;LOW LIMIT	
2314	006636	001000				1000	;HIGH LIMIT	
2315	006640	001226				TSTNO	;STORE AT THIS LOCATION	
2316	006642	000			.BYTE	0	;MASK	
2317	006643	001			.BYTE	1	;HOW MANY TIMES + 2	
2318	006644	012700	007160			MOV	#TST1,RO	
2319	006650	022710	012737		5\$:	CMP	#12737,(RO)	
2320	006654	001017				BNE	6\$	
2321	006656	023760	001226	000002		CMP	TSTNO,2(RO)	
2322	006664	001013				BNE	6\$	
2323	006666	022760	001226	000004		CMP	#TSTNO,4(RO)	
2324	006674	001007				BNE	6\$	
2325	006676	010037	001214			MOV	RO,RETURN	;SAVE PC
2326	006702	104402	005136			TYPE	,MCRLF	
2327	006706	104402	005203			TYPE	,MR	
2328	006712	000412				BR	8\$	
2329	006714	005720			6\$:	TST	(RO)+	
2330	006716	020027	026316			CMP	RO,#TLAST+10	
2331	006722	001352				BNE	5\$	
2332	006724	104402	005132			TYPE	,MQM	
2333	006730	000734				BR	7\$	
2334								
2335	006732	012737	007160	001214	4\$:	MOV	#TST1,RETURN	;PREPARE RETURN ADDRESS
2336	006740	000177	172250		8\$:	JMP	JRETURN	;GO START TESTING.
2337								
2338	006744	011003			FIX.00:	MOV	(RO),R3	;GET PARAMETERS.
2339	006746	000207			5\$:	RTS	PC	


```

2340
2341
2342
2343 006750 012104 ACC: MOV (R1)+,R4 ;GET THE FLAG FOR # OF CLOCK TICKS
2344 006752 104412 000002 1$: PKCLK ,2
2345 006756 000241 CLC ;PUT CARRY IN A KNOWN STATE
2346 006760 032777 040000 172424 BIT #MTDATA,@TXCSR ;FIND OUT IF BIT IS A 1 OR 0
2347 006766 001401 BEQ .+4 ;BR IF 0
2348 006770 000261 SEC ;SET THE BIT
2349 006772 006004 ROR R4 ;PICK UP CARRY AND PUSH INTO R4
2350 006774 103366 BCC 1$ ;BRANCH IF MORE TO GO
2351 006776 000201 RTS R1
2352 007000 005037 001246 ABRT: CLR TEMPS
2353 007004 012137 001244 1$: MOV (R1)+,TEMP4 ;GET THE # OF ABORTS TO DO
2354 007010 104412 000002 PKCLK ,2 ;POKE OUT A BIT
2355 007014 032777 040000 172370 BIT #MTDATA,@TXCSR ;CHECK MAINT DATA OUT
2356 007022 001001 BNE 2$ ;BRANCH IF IT IS A ONE
2357 007024 104013 HLT 13 ;OUTPUT WAS A ZERO--NG
2358 007026 005237 001246 2$: INC TEMPS ;INC THE # OF BITS OUTPUT
2359 007032 022737 000010 001246 CMP #8.,TEMP5 ;IS THE CHARACTER DONE?
2360 007040 001363 BNE 1$ ;BRANCH IF NOT DONE
2361 007042 005337 001244 DEC TEMP4 ;LOWER THE #TO DO
2362 007046 001360 BNE 1$ ;BRANCH IF NOT DONE
2363 007050 000201 RTS R1
2364
2365 007052 012137 001244 FLG: MOV (R1)+,TEMP4 ;GET THE # OF FLAGS
2366 007056 104412 000002 64$: PKCLK ,2 ;POKE OUT THE FIRST BIT OF THE FLAG
2367 007062 032777 040000 172322 BIT #MTDATA,@TXCSR ;CHECK MAINT DATA OUT
2368 007070 001401 BEQ 65$ ;BRANCH IF 0
2369 007072 104012 HLT 12 ;DUP FAILED TO CLOCK OUT FIRST BIT
2370 007074 005037 001246 65$: CLR TEMPS ;SETUP FOR 1'S OUTPUT
2371 007100 104412 000002 1$: PKCLK ,2 ;CONTINUE TO POKE OUT BITS
2372 007104 032777 040000 172300 BIT #MTDATA,@TXCSR ;TEST MAINT DATA OUT
2373 007112 001001 BNE 2$ ;BRANCH IF A 1
2374 007114 104013 HLT 13 ;DUP FAILED TO CLOCK A ONE
2375 007116 005237 001246 2$: INC TEMPS ;KEEP UP WITH THE # OF 1'S OUTPUT
2376 007122 022737 000006 001246 CMP #6,TEMP5 ;ARE WE DONE WITH SIX ONES?
2377 007130 001363 BNE 1$ ;NO-BRANCH BACK
2378 007132 104412 000002 PKCLK ,2 ;YES,OUTPUT THE LAST 0
2379 007136 032777 040000 172246 BIT #MTDATA,@TXCSR ;CHECK IT
2380 007144 001401 BEQ 3$ ;BRANCH IF A 0
2381 007146 104012 HLT 12 ;LAST BIT OF FLAG WAS NOT CORECT
2382 007150 005337 001244 3$: DEC TEMP4 ;ARE WE DONE WITH FLAGS?
2383 007154 001340 BNE 64$ ;BR IF NO
2384 007156 000201 RTS R1
2385

```

CONTROL REGISTERS ADDRESS TIMEOUT TEST

2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441

007160 012737 000001 001226
007166 012737 007264 001216
007174 012737 007226 001220
007202 012700 000004
007206 013701 001404
007212 012737 007256 000004
007220 012737 000340 000006
007226 005711
007230 104401
007232 062701 000002
007236 005300
007240 001372
007242 012737 000006 000004
007250 005037 000006
007254 104400
007256 011602
007260 104001
007262 000002

007264 012737 000002 001226
007272 012737 007442 001216
007300 012737 007324 001220
007306 012700 007432
007312 013703 001404
007316 005013
007320 012702 000004
007324 011005
007326 010513
007330 011304
007332 042704 000200
007336 042704 000200
007342 020504

***** TEST 1 *****
*VERIFY THAT ADDRESSING DEVICE DOES *NOT* CAUSE
*A TIME-OUT TRAP.

*
TEST 1
*

```
TST1:  MOV #1, @TSTNO
        MOV #TST2, NEXT
        MOV #1$, LOCK
        MOV #4, R0 ; SET FOR MAX. 4 PRI. REGISTERS
        MOV RXC5R, R1 ; GET FIRST PRI. ADDRESS
        MOV #2$, 4 ; SET FOR TIME-OUT TRAP.
        MOV #340, 6 ; SAFE GUARD.
1$:     TST (R1) ; REFERENCE THE ADDRESS.
        SCOP1 ; IF SW09=1; GOTO 1$
        ADD #2, R1 ; UPDATE TO NEXT ADDRESS.
        DEC R0 ; ARE ALL ADDRESS CHECKED?
        BNE 1$ ; BR IF NO.
        MOV #6, 4 ; RESET TRAP ZONE.
        CLR @#6
2$:     SCOPE ; SCOPE THIS TEST
        MOV (SP), R2 ; SAVE THE TRAP PC
        HLT 1 ; REPORT TIME-OUT TRAP
        RTI ; RETURN TO MAIN PROGRAM
```

***** TEST 2 *****
*PRIMARY REGISTER ADDRESSING TEST
*LOAD EACH PRIMARY REGISTER WITH A
*DIFFERENT NUMBER AND VERIFY EACH
*WAS INDIVIDUALLY ADDRESSED

*
TEST 2
*

```
TST2:  MOV #2, @TSTNO
        MOV #TST3, NEXT
        MOV #1$, LOCK
        MOV #3$, R0 ; SET THE TABLE POINTER
        MOV RXC5R, R3 ; SET THE DUP HARDWARE POINTER
        CLR (R3) ; CLR THE REGISTER BEFORE STARTING
        MOV #4, R2 ; SET FOR 4 PRIMARY REGISTERS
1$:     MOV (R0), R5 ; SET "EXPECTED"
        MOV R5, (R3) ; WRITE "EXPECTED" TO THE REGISTER
        MOV (R3), R4 ; READ THE REGISTER BACK
        BIC #BIT7, R4 ; CLR UNWANTED BIT
        BIC #BIT7, R4 ; CLR UNWANTED BITS
        CMP R5, R4 ; DOES EXPECTED=RECEIVED?
```

CONTROL REGISTERS DUAL ADDRESSING TEST

```

2442 007344 001401      BEQ      2$      ;BR IF YES
2443 007346 104003      HLT      3      ;THIS IS A DATA ERROR. IT IS **NOT**
2444                                     ;A DUAL ADDRESSING ERROR!!!!!!
2445 007350 104401      2$: SCOPE1      ;SW09=1?
2446 007352 022023      CMP      (R0)+,(R3)+ ;POP DATA AND HARDWARE POINTERS
2447 007354 005302      DEC      R2      ;UPDATE THE REGISTER COUNTER
2448 007356 001362      BNE      1$      ;BRANCH IF MORE TO GO
2449                                     ;:NOW CHECK FOR DUAL ADDRESSING
2450 007360 012700 007432  MOV      #3$,R0      ;SET THE TABLE POINTER
2451 007364 013703 001404  MOV      RXCSR,R3    ;SET THE DUP HARDWARE POINTER
2452 007370 012737 007402 001220  MOV      #4$,LOCK    ;SET FOR SW09=1
2453 007376 012702 000004  MOV      #4,R2      ;SET FOR 4 PRIMARY REGISTERS
2454 007402 011005      4$: MOV      (R0),R5    ;SET "EXPECTED"
2455 007404 011304      MOV      (R3),R4    ;READ THE REGISTER BACK
2456 007406 042704 000200  BIC      #BIT7,R4   ;CLR UNWANTED BITS
2457 007412 020504      CMP      R5,R4
2458 007414 001401      BEQ      5$
2459 007416 104003      HLT      3
2460 007420 104401      5$: SCOPE1      ;BRANCH IF OK
2461 007422 022023      CMP      (R0)+,(R3)+ ;THIS IS A DUAL ADDRESSING ERROR
2462 007424 005302      DEC      R2      ;SW09=1?
2463 007426 001365      BNE      4$      ;POP POINTERS
2464 007430 104400      SCOPE      ;UPDATE THE REGISTER COUNTER
2465 007432 000020      3$: .WORD 00020    ;BRANCH IF MORE TO GO
2466 007434 000000      .WORD 00000    ;SCOPE THIS TEST
2467 007436 000010      .WORD 0C010    ;RXCSR
2468 007440 000100      .WORD 00100    ;RXDBUF AND PARCSR
2469                                     ;TXCSR
2470                                     ;TXDBUF

```

```

:***** TEST 3 *****
:*RECEIVER CONTROL REGISTER RESET TEST. TEST THAT AFTER
:*RECEIVER CONTROL REGISTER IS WRITTEN AND A BUS RESET IS
:*DONE THAT RECEIVER CONTROL REGISTER IS CLEARED.
:*****

```

```

:*****
:TEST 3
:*****

```

```

2483 007442 012737 000003 001226  TST3: MOV      #3,#TSTNO
2484 007450 012737 007504 001216  MOV      #TST4,NEXT
2485 007456 005005      CLR      R5      ;SET "EXPECTED"
2486 007460 013703 001404  MOV      RXCSR,R3  ;GET THE RECEIVER CONTROL REGISTER
2487 007464 012713 177777  MOV      #-1,(R3) ;LOAD RECEIVER CONTROL REGISTER WITH ALL ONES
2488 007470 000005      RESET
2489 007472 011304      MOV      (R3),R4  ;READ THE RECEIVER CONTROL REGISTER
2490 007474 020504      CMP      R5,R4   ;R5=GOOD,R4=?
2491 007476 001401      BEQ      1$      ;BR IF OK
2492 007500 104002      HLT      2      ;COMPARISON ERROR
2493 007502 104400      1$: SCOPE      ;SCOPE THIS TEST

```

```

:***** TEST 4 *****
:*RECEIVER BUFFER REGISTER RESET TEST. TEST THAT AFTER A BUS

```

```

2494
2495
2496
2497

```

2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553

007504 012737 000004 001226
007512 012737 007542 001216
007520 005005
007522 013703 001406
007526 000005
007530 011304
007532 020504
007534 001401
007536 104002
007540 104400

007542 012737 000005 001226
007550 012737 007604 001216
007556 005005
007560 013703 001410
007564 012713 177777
007570 000005
007572 011304
007574 020504
007576 001401
007600 104002
007602 104400

; *RESET IS DONE THAT RECEIVER BUFFER REGISTER IS CLEARED.
; *****

; *****
; *
; TEST 4
; *
; *****

```
TST4:  MOV    #4, @TSTNO
      MOV    @TST5, NEXT
      CLR    R5                ; SET "EXPECTED"
      MOV    RXDBUF, R3        ; GET THE RECEIVER BUFFER REGISTER
      RESET
      MOV    (R3), R4          ; READ THE RECEIVER BUFFER REGISTER
      CMP    R5, R4            ; R5=GOOD, R4= ?
      BEQ    1$                ; BR IF OK
      HLT    2                  ; COMPARISON ERROR
1$:    SCOPE                    ; SCOPE THIS TEST
```

; ***** TEST 5 *****
; *PARAMETER STATUS REGISTER RESET TEST. TEST THAT AFTER
; *PARAMETER STATUS REGISTER IS WRITTEN AND A BUS RESET IS
; *DONE THAT PARAMETER STATUS REGISTER IS CLEARED.
; *****

; *****
; *
; TEST 5
; *
; *****

```
TST5:  MOV    #5, @TSTNO
      MOV    @TST6, NEXT
      CLR    R5                ; SET "EXPECTED"
      MOV    PARCSR, R3        ; GET THE PARAMETER STATUS REGISTER
      MOV    #-1, (R3)         ; LOAD PARAMETER STATUS REGISTER WITH ALL ONES
      RESET
      MOV    (R3), R4          ; READ THE PARAMETER STATUS REGISTER
      CMP    R5, R4            ; R5=GOOD, R4= ?
      BEQ    1$                ; BR IF OK
      HLT    2                  ; COMPARISON ERROR
1$:    SCOPE                    ; SCOPE THIS TEST
```

; ***** TEST 6 *****
; *TRANSMITTER CONTROL REGISTER RESET TEST. TEST THAT AFTER
; *TRANSMITTER CONTROL REGISTER IS WRITTEN AND A BUS RESET IS
; *DONE THAT TRANSMITTER CONTROL REGISTER IS CLEARED.
; *****

; *****
; *
; TEST 6
; *

```

2554
2555
2556 007604 012737 000006 001226
2557 007612 012737 007652 001216
2558 007620 005005
2559 007622 013703 001412
2560 007626 012713 177777
2561 007632 000005
2562 007634 011304
2563 007636 042704 000200
2564 007642 020504
2565 007644 001401
2566 007646 104002
2567 007650 104400
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582 007652 012737 000007 001226
2583 007660 012737 007714 001216
2584 007666 005005
2585 007670 013703 001414
2586 007674 012713 177777
2587 007700 000005
2588 007702 011304
2589 007704 020504
2590 007706 001401
2591 007710 104002
2592 007712 104400
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607 007714 012737 000010 001226
2608 007722 012737 010000 001216
2609 007730 013703 001412

```

```

:*****
:*****

```

```

TST6:  MOV    #6,#TSTNO
        MOV    #TST7,NEXT
        CLR    R5                ;SET "EXPECTED"
        MOV    TXCSR,R3         ;GET THE TRANSMITTER CONTROL REGISTER
        MOV    #-1,(R3)        ;LOAD TRANSMITTER CONTROL REGISTER WITH ALL ONES
        RESET
        MOV    (R3),R4          ;READ THE TRANSMITTER CONTROL REGISTER
        BIC    #BIT7,R4        ;CLR UNWANTED BITS
        CMP    R5,R4            ;R5=GOOD,R4= ?
        BEQ    1$              ;BR IF OK
        HLT    2                ;COMPARISON ERROR
1$:     SCOPE                    ;SCOPE THIS TEST

```

```

:***** TEST 7 *****
:*TRANSMITTER BUFFER REGISTER RESET TEST. TEST THAT AFTER
:*TRANSMITTER BUFFER REGISTER IS WRITTEN AND A BUS RESET IS
:*DONE THAT TRANSMITTER BUFFER REGISTER IS CLEARED.
:*****

```

```

:*****
:
: TEST 7
:
:*****

```

```

:*****
:*****

```

```

TST7:  MOV    #7,#TSTNO
        MOV    #TST10,NEXT
        CLR    R5                ;SET "EXPECTED"
        MOV    TXDBUF,R3        ;GET THE TRANSMITTER BUFFER REGISTER
        MOV    #-1,(R3)        ;LOAD TRANSMITTER BUFFER REGISTER WITH ALL ONES
        RESET
        MOV    (R3),R4          ;READ THE TRANSMITTER BUFFER REGISTER
        CMP    R5,R4            ;R5=GOOD,R4= ?
        BEQ    1$              ;BR IF OK
        HLT    2                ;COMPARISON ERROR
1$:     SCOPE                    ;SCOPE THIS TEST

```

```

:***** TEST 10 *****
:*TRANSMITTER CONTROL REGISTER READ/WRITE BIT TEST.
:*SET BIT3, VERIFY BIT3 WAS SET.
:*CLEAR BIT3, VERIFY BIT3 WAS CLEARED.
:*****

```

```

:*****
:
: TEST 10
:
:*****

```

```

:*****
:*****

```

```

TST10: MOV    #10,#TSTNO
        MOV    #TST11,NEXT
        MOV    TXCSR,R3         ;SET REGISTER TO BE TESTED.

```

```

2610 007734 012705 000010      MOV      #BIT3,R5      ;SET "EXPECTED"
2611 007740 010513      MOV      R5,(R3)      ;WRITE THE REGISTER.
2612 007742 011304      MOV      (R3),R4      ;READ THE REGISTER.
2613 007744 042704 000200      BIC      #BIT7,R4     ;CLEAR UNWANTED BITS
2614 007750 020504      CMP      R5,R4        ;R5=GOOD; R4=UNKNOWN.
2615 007752 001401      BEQ      1$          ;ARE THEY THE SAME?
2616 007754 104003      HLT      3           ;COMPARISON ERROR.
2617 007756 040513 1$:      BIC      R5,(R3)     ;CLEAR BIT3
2618 007760 011304      MOV      (R3),R4      ;READ THE REGISTER.
2619 007762 042704 000200      BIC      #BIT7,R4     ;CLEAR UNWANTED BITS
2620 007766 005005      CLR      R5          ;SET "EXPECTED"
2621 007770 020504      CMP      R5,R4        ;R5=GOOD; R4=?
2622 007772 001401      BEQ      2$          ;BR IF OK
2623 007774 104003      HLT      3           ;COMPARISON ERROR
2624 007776 104400 2$:      SCOPE              ;SCOPE THIS TEST

```

```

;***** TEST 11 *****
;TRANSMITTER CONTROL REGISTER READ/WRITE BIT TEST.
;SET BIT4, VERIFY BIT4 WAS SET.
;CLEAR BIT4, VERIFY BIT4 WAS CLEARED.
;*****

```

```

;*****
;TEST 11
;*****

```

```

2638 010000 012737 000011 001226 1ST11: MOV      #11,0#TSTNO
2639 010006 012737 010064 001216  MOV      #TST12,NEXT
2640 010014 013703 001412  MOV      TXCSR,R3     ;SET REGISTER TO BE TESTED.
2641 010020 012705 000020  MOV      #BIT4,R5     ;SET "EXPECTED"
2642 010024 010513  MOV      R5,(R3)      ;WRITE THE REGISTER.
2643 010026 011304  MOV      (R3),R4      ;READ THE REGISTER.
2644 010030 042704 000200  BIC      #BIT7,R4     ;CLEAR UNWANTED BITS
2645 010034 020504  CMP      R5,R4        ;R5=GOOD; R4=UNKNOWN.
2646 010036 001401  BEQ      1$          ;ARE THEY THE SAME?
2647 010040 104003  HLT      3           ;COMPARISON ERROR.
2648 010042 040513 1$:      BIC      R5,(R3)     ;CLEAR BIT4
2649 010044 011304  MOV      (R3),R4      ;READ THE REGISTER.
2650 010046 042704 000200  BIC      #BIT7,R4     ;CLEAR UNWANTED BITS
2651 010052 005005  CLR      R5          ;SET "EXPECTED"
2652 010054 020504  CMP      R5,R4        ;R5=GOOD; R4=?
2653 010056 001401  BEQ      2$          ;BR IF OK
2654 010060 104003  HLT      3           ;COMPARISON ERROR
2655 010062 104400 2$:      SCOPE              ;SCOPE THIS TEST

```

```

;***** TEST 12 *****
;TRANSMITTER CONTROL REGISTER READ/WRITE BIT TEST.
;SET BIT10, VERIFY BIT10 WAS SET.
;CLEAR BIT10, VERIFY BIT10 WAS CLEARED.
;*****

```

```

;*****

```

```

2657
2658
2659
2660
2661
2662
2663
2664
2665

```

```

2666
2667
2668
2669
2670
2671 010064 012737 000012 001226
2672 010072 012737 010150 001216
2673 010100 013703 001412
2674 010104 012705 002000
2675 010110 010513
2676 010112 011304
2677 010114 042704 000200
2678 010120 020504
2679 010122 001401
2680 010124 104003
2681 010126 040513
2682 010130 011304
2683 010132 042704 000200
2684 010136 005005
2685 010140 020504
2686 010142 001401
2687 010144 104003
2688 010146 104400
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703 010150 012737 000013 001226
2704 010156 012737 010234 001216
2705 010164 013703 001412
2706 010170 012705 004000
2707 010174 010513
2708 010176 011304
2709 010200 042704 000200
2710 010204 020504
2711 010206 001401
2712 010210 104003
2713 010212 040513
2714 010214 011304
2715 010216 042704 000200
2716 010222 005005
2717 010224 020504
2718 010226 001401
2719 010230 104003
2720 010232 104400
2721

```

```

; TEST 12 *
;*****
;*****
TST12: MOV #12, #TSTNO
MOV #TST13, NEXT
MOV TXCSR, R3 ; SET REGISTER TO BE TESTED.
MOV #BIT10, R5 ; SET "EXPECTED"
MOV R5, (R3) ; WRITE THE REGISTER.
MOV (R3), R4 ; READ THE REGISTER.
BIC #BIT7, R4 ; CLEAR UNWANTED BITS
CMP R5, R4 ; R5=GOOD; R4=UNKNOWN.
BEQ 1$ ; ARE THEY THE SAME?
HLT 3 ; COMPARISON ERROR.
1$: BIC R5, (R3) ; CLEAR BIT10
MOV (R3), R4 ; READ THE REGISTER.
BIC #BIT7, R4 ; CLEAR UNWANTED BITS
CLR R5 ; SET "EXPECTED"
CMP R5, R4 ; R5=GOOD; R4=?
BEQ 2$ ; BR IF OK
HLT 3 ; COMPARISON ERROR
2$: SCOPE ; SCOPE THIS TEST

```

```

;***** TEST 13 *****
;TRANSMITTER CONTROL REGISTER READ/WRITE BIT TEST.
;SET BIT11, VERIFY BIT11 WAS SET.
;CLEAR BIT11, VERIFY BIT11 WAS CLEARED.
;*****

```

```

;*****
; TEST 13 *
;*****
;*****
TST13: MOV #13, #TSTNO
MOV #TST14, NEXT
MOV TXCSR, R3 ; SET REGISTER TO BE TESTED.
MOV #BIT11, R5 ; SET "EXPECTED"
MOV R5, (R3) ; WRITE THE REGISTER.
MOV (R3), R4 ; READ THE REGISTER.
BIC #BIT7, R4 ; CLEAR UNWANTED BITS
CMP R5, R4 ; R5=GOOD; R4=UNKNOWN.
BEQ 1$ ; ARE THEY THE SAME?
HLT 3 ; COMPARISON ERROR.
1$: BIC R5, (R3) ; CLEAR BIT11
MOV (R3), R4 ; READ THE REGISTER.
BIC #BIT7, R4 ; CLEAR UNWANTED BITS
CLR R5 ; SET "EXPECTED"
CMP R5, R4 ; R5=GOOD; R4=?
BEQ 2$ ; BR IF OK
HLT 3 ; COMPARISON ERROR
2$: SCOPE ; SCOPE THIS TEST

```

2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777

010234 012737 000014 001226
010242 012737 010320 001216
010250 013703 001412
010254 012705 010000
010260 010513
010262 011304
010264 042704 000200
010270 020504
010272 001401
010274 104003
010276 040513
010300 011304
010302 042704 000200
010306 005005
010310 020504
010312 001401
010314 104003
010316 104400

010320 012737 000015 001226
010326 012737 010404 001216
010334 013703 001412
010340 012705 020000
010344 010513
010346 011304
010350 042704 000200
010354 020504
010356 001401
010360 104003
010362 040513

```
***** TEST 14 *****  
*TRANSMITTER CONTROL REGISTER READ/WRITE BIT TEST.  
*SET BIT12, VERIFY BIT12 WAS SET.  
*CLEAR BIT12, VERIFY BIT12 WAS CLEARED.  
*****  
*****  
TEST 14  
*****  
*****  
TST14: MOV #14, #TSTNO  
MOV #TST15, NEXT  
MOV TXCSR, R3 ; SET REGISTER TO BE TESTED.  
MOV #BIT12, R5 ; SET "EXPECTED"  
MOV R5, (R3) ; WRITE THE REGISTER.  
MOV (R3), R4 ; READ THE REGISTER.  
BIC #BIT7, R4 ; CLEAR UNWANTED BITS  
CMP R5, R4 ; R5=GOOD; R4=UNKNOWN.  
BEQ 1$ ; ARE THEY THE SAME?  
HLT 3 ; COMPARISON ERROR.  
1$: BIC R5, (R3) ; CLEAR BIT12  
MOV (R3), R4 ; READ THE REGISTER.  
BIC #BIT7, R4 ; CLEAR UNWANTED BITS  
CLR R5 ; SET "EXPECTED"  
CMP R5, R4 ; R5=GOOD; R4=?  
BEQ 2$ ; BR IF OK  
HLT 3 ; COMPARISON ERROR  
2$: SCOPE ; SCOPE THIS TEST
```

```
***** TEST 15 *****  
*TRANSMITTER CONTROL REGISTER READ/WRITE BIT TEST.  
*SET BIT13, VERIFY BIT13 WAS SET.  
*CLEAR BIT13, VERIFY BIT13 WAS CLEARED.  
*****  
*****  
TEST 15  
*****  
*****  
TST15: MOV #15, #TSTNO  
MOV #TST16, NEXT  
MOV TXCSR, R3 ; SET REGISTER TO BE TESTED.  
MOV #BIT13, R5 ; SET "EXPECTED"  
MOV R5, (R3) ; WRITE THE REGISTER.  
MOV (R3), R4 ; READ THE REGISTER.  
BIC #BIT7, R4 ; CLEAR UNWANTED BITS  
CMP R5, R4 ; R5=GOOD; R4=UNKNOWN.  
BEQ 1$ ; ARE THEY THE SAME?  
HLT 3 ; COMPARISON ERROR.  
1$: BIC R5, (R3) ; CLEAR BIT13
```


TRANSMITTER CONTROL REGISTER READ/WRITE TEST BIT 13

```

2778 010364 011304          MOV      (R3),R4      ;READ THE REGISTER.
2779 010366 042704 000200  BIC      #BIT7,R4    ;CLEAR UNWANTED BITS
2780 010372 005005          CLR      R5          ;SET "EXPECTED"
2781 010374 020504          CMP      R5,R4      ;R5=GOOD; R4=?
2782 010376 001401          BEQ     2$          ;BR IF OK
2783 010400 104003          HLT     3          ;COMPARISON ERROR
2784 010402 104400          2$:     SCOPE      ;SCOPE THIS TEST
2785
2786
2787

```

```

***** TEST 16 *****
*TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
*SET BIT0, VERIFY BIT0 WAS SET.
*CLEAR BIT0, VERIFY BIT0 WAS CLEARED.
*****

```

```

*****
*
* TEST 16
*
*****

```

```

2798
2799 010404 012737 000016 001226 TST16: MOV      #16,#TSTNO
2800 010412 012737 010460 001216  MOV      #TST17,NEXT
2801 010420 013703 001414          MOV      TXDBUF,R3   ;SET REGISTER TO BE TESTED.
2802 010424 012705 000001          MOV      #BIT0,R5   ;SET "EXPECTED"
2803 010430 010513          MOV      R5,(R3)    ;WRITE THE REGISTER.
2804 010432 011304          MOV      (R3),R4    ;READ THE REGISTER.
2805 010434 020504          CMP      R5,R4      ;R5=GOOD; R4=UNKNOWN.
2806 010436 001401          BEQ     1$          ;ARE THEY THE SAME?
2807 010440 104003          HLT     3          ;COMPARISON ERROR.
2808 010442 040513          1$:     BIC      R5,(R3) ;CLEAR BIT0
2809 010444 011304          MOV      (R3),R4    ;READ THE REGISTER.
2810 010446 005005          CLR      R5          ;SET "EXPECTED"
2811 010450 020504          CMP      R5,R4      ;R5=GOOD; R4=?
2812 010452 001401          BEQ     2$          ;BR IF OK
2813 010454 104003          HLT     3          ;COMPARISON ERROR
2814 010456 104400          2$:     SCOPE      ;SCOPE THIS TEST
2815
2816
2817

```

```

***** TEST 17 *****
*TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
*SET BIT1, VERIFY BIT1 WAS SET.
*CLEAR BIT1, VERIFY BIT1 WAS CLEARED.
*****

```

```

*****
*
* TEST 17
*
*****

```

```

2828
2829 010460 012737 000017 001226 TST17: MOV      #17,#TSTNO
2830 010466 012737 010534 001216  MOV      #TST20,NEXT
2831 010474 013703 001414          MOV      TXDBUF,R3   ;SET REGISTER TO BE TESTED.
2832 010500 012705 000002          MOV      #BIT1,R5   ;SET "EXPECTED"
2833 010504 010513          MOV      R5,(R3)    ;WRITE THE REGISTER.

```

TRANSMITTER BUFFER REGISTER READ/WRITE TEST BIT 1

```

2834 010506 011304      MOV      (R3),R4      ;READ THE REGISTER.
2835 010510 020504      CMP      R5,R4       ;R5=GOOD; R4=UNKNOWN.
2836 010512 001401      BEQ     1$           ;ARE THEY THE SAME?
2837 010514 104003      HLT     3            ;COMPARISON ERROR.
2838 010516 040513      1$:    BIC     R5,(R3) ;CLEAR BIT1
2839 010520 011304      MOV     (R3),R4     ;READ THE REGISTER.
2840 010522 005005      CLR     R5          ;SET "EXPECTED"
2841 010524 020504      CMP     R5,R4       ;R5=GOOD; R4=?
2842 010526 001401      BEQ     2$           ;BR IF OK
2843 010530 104003      HLT     3            ;COMPARISON ERROR
2844 010532 104400      2$:    SCOPE        ;SCOPE THIS TEST

```

```

;***** TEST 20 *****
;TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
;SET BIT2, VERIFY BIT2 WAS SET.
;CLEAR BIT2, VERIFY BIT2 WAS CLEARED.
;*****

```

```

;*****
;TEST 20
;*****

```

```

2858
2859 010534 012737 000020 001226 1ST20: MOV     #20,#TSTNO
2860 010542 012737 010610 001216  MOV     #TST21,NEXT
2861 010550 013703 001414  MOV     TXDBUF,R3    ;SET REGISTER TO BE TESTED.
2862 010554 012705 000004  MOV     #BIT2,R5     ;SET "EXPECTED"
2863 010560 010513  MOV     R5,(R3)     ;WRITE THE REGISTER.
2864 010562 011304  MOV     (R3),R4     ;READ THE REGISTER.
2865 010564 020504  CMP     R5,R4       ;R5=GOOD; R4=UNKNOWN.
2866 010566 001401  BEQ     1$           ;ARE THEY THE SAME?
2867 010570 104003  HLT     3            ;COMPARISON ERROR.
2868 010572 040513  1$:    BIC     R5,(R3) ;CLEAR BIT2
2869 010574 011304  MOV     (R3),R4     ;READ THE REGISTER.
2870 010576 005005  CLR     R5          ;SET "EXPECTED"
2871 010600 020504  CMP     R5,R4       ;R5=GOOD; R4=?
2872 010602 001401  BEQ     2$           ;BR IF OK
2873 010604 104003  HLT     3            ;COMPARISON ERROR
2874 010606 104400  2$:    SCOPE        ;SCOPE THIS TEST

```

```

;***** TEST 21 *****
;TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
;SET BIT3, VERIFY BIT3 WAS SET.
;CLEAR BIT3, VERIFY BIT3 WAS CLEARED.
;*****

```

```

;*****
;TEST 21
;*****

```

```

2888 010610 012737 000021 001226 1ST21: MOV     #21,#TSTNO

```

TRANSMITTER BUFFER REGISTER READ/WRITE TEST BIT 3

```

2890 010616 012737 010664 001216      MOV      #TST22,NEXT
2891 010624 013703 001414      MOV      TXDBUF,R3          ;SET REGISTER TO BE TESTED.
2892 010630 012705 000010      MOV      #BIT3,R5          ;SET "EXPECTED"
2893 010634 010513      MOV      R5,(R3)           ;WRITE THE REGISTER.
2894 010636 011304      MOV      (R3),R4           ;READ THE REGISTER.
2895 010640 020504      CMP      R5,R4             ;R5=GOOD; R4=UNKNOWN.
2896 010642 001401      BEQ      1$                ;ARE THEY THE SAME?
2897 010644 104003      HLT      3                  ;COMPARISON ERROR.
2898 010646 040513      1$:    BIC      R5,(R3)       ;CLEAR BIT3
2899 010650 011304      MOV      (R3),R4           ;READ THE REGISTER.
2900 010652 005005      CLR      R5                ;SET "EXPECTED"
2901 010654 020504      CMP      R5,R4             ;R5=GOOD; R4=?
2902 010656 001401      BEQ      2$                ;BR IF OK
2903 010660 104003      HLT      3                  ;COMPARISON ERROR
2904 010662 104400      2$:    SCOPE                ;SCOPE THIS TEST

```

```

***** TEST 22 *****
*TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
*SET BIT4, VERIFY BIT4 WAS SET.
*CLEAR BIT4, VERIFY BIT4 WAS CLEARED.
*****

```

```

*****
: TEST 22
*****

```

```

2919 010664 012737 000022 001226      TST22: MOV      #22,#TSTNO
2920 010672 012737 010740 001216      MOV      #TST23,NEXT
2921 010700 013703 001414      MOV      TXDBUF,R3          ;SET REGISTER TO BE TESTED.
2922 010704 012705 000020      MOV      #BIT4,R5          ;SET "EXPECTED"
2923 010710 010513      MOV      R5,(R3)           ;WRITE THE REGISTER.
2924 010712 011304      MOV      (R3),R4           ;READ THE REGISTER.
2925 010714 020504      CMP      R5,R4             ;R5=GOOD; R4=UNKNOWN.
2926 010716 001401      BEQ      1$                ;ARE THEY THE SAME?
2927 010720 104003      HLT      3                  ;COMPARISON ERROR.
2928 010722 040513      1$:    BIC      R5,(R3)       ;CLEAR BIT4
2929 010724 011304      MOV      (R3),R4           ;READ THE REGISTER.
2930 010726 005005      CLR      R5                ;SET "EXPECTED"
2931 010730 020504      CMP      R5,R4             ;R5=GOOD; R4=?
2932 010732 001401      BEQ      2$                ;BR IF OK
2933 010734 104003      HLT      3                  ;COMPARISON ERROR
2934 010736 104400      2$:    SCOPE                ;SCOPE THIS TEST

```

```

***** TEST 23 *****
*TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
*SET BITS, VERIFY BITS WAS SET.
*CLEAR BITS, VERIFY BITS WAS CLEARED.
*****

```

```

*****
: TEST 23

```

```

2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945

```

```

2946
2947
2948
2949 010740 012737 000023 001226
2950 010746 012737 011014 001216
2951 010754 013703 001414
2952 010760 012705 000040
2953 010764 010513
2954 010766 011304
2955 010770 020504
2956 010772 001401
2957 010774 104003
2958 010776 040513
2959 011000 011304
2960 011002 005005
2961 011004 020504
2962 011006 001401
2963 011010 104003
2964 011012 104400

```

```

:*****
:*****
:*****
TST23: MOV #23, @TSTNO
MOV @TST24, NEXT
MOV TXDBUF, R3 ; SET REGISTER TO BE TESTED.
MOV #BIT5, R5 ; SET "EXPECTED"
MOV R5, (R3) ; WRITE THE REGISTER.
MOV (R3), R4 ; READ THE REGISTER.
CMP R5, R4 ; R5=GOOD; R4=UNKNOWN.
BEQ 1$ ; ARE THEY THE SAME?
HLT 3 ; COMPARISON ERROR.
1$: BIC R5, (R3) ; CLEAR BIT5
MOV (R3), R4 ; READ THE REGISTER.
CLR R5 ; SET "EXPECTED"
CMP R5, R4 ; R5=GOOD; R4=?
BEQ 2$ ; BR IF OK
HLT 3 ; COMPARISON ERROR
2$: SCOPE ; SCOPE THIS TEST

```

```

:***** TEST 24 *****
:TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
:SET BIT6, VERIFY BIT6 WAS SET.
:CLEAR BIT6, VERIFY BIT6 WAS CLEARED.
:*****

```

```

2972
2973
2974
2975
2976
2977
2978
2979 011014 012737 000024 001226
2980 011022 012737 011070 001216
2981 011030 013703 001414
2982 011034 012705 000100
2983 011040 010513
2984 011042 011304
2985 011044 020504
2986 011046 001401
2987 011050 104003
2988 011052 040513
2989 011054 011304
2990 011056 005005
2991 011060 020504
2992 011062 001401
2993 011064 104003
2994 011066 104400
2995
2996
2997
2998
2999
3000
3001

```

```

:*****
:*****
:*****
TEST 24
:*****
:*****
TST24: MOV #24, @TSTNO
MOV @TST25, NEXT
MOV TXDBUF, R3 ; SET REGISTER TO BE TESTED.
MOV #BIT6, R5 ; SET "EXPECTED"
MOV R5, (R3) ; WRITE THE REGISTER.
MOV (R3), R4 ; READ THE REGISTER.
CMP R5, R4 ; R5=GOOD; R4=UNKNOWN.
BEQ 1$ ; ARE THEY THE SAME?
HLT 3 ; COMPARISON ERROR.
1$: BIC R5, (R3) ; CLEAR BIT6
MOV (R3), R4 ; READ THE REGISTER.
CLR R5 ; SET "EXPECTED"
CMP R5, R4 ; R5=GOOD; R4=?
BEQ 2$ ; BR IF OK
HLT 3 ; COMPARISON ERROR
2$: SCOPE ; SCOPE THIS TEST

```

```

:***** TEST 25 *****
:TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
:SET BIT7, VERIFY BIT7 WAS SET.
:CLEAR BIT7, VERIFY BIT7 WAS CLEARED.
:*****

```

3002
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057

011070 012737 000025 001226
011076 012737 011144 001216
011104 013703 001414
011110 012705 000200
011114 010513
011116 011304
011120 020504
011122 001401
011124 104003
011126 040513
011130 011304
011132 005005
011134 020504
011136 001401
011140 104003
011142 104400

```
*****  
*  
: TEST 25  
*  
*****  
*****  
TST25: MOV #25, J#TSTNO  
MOV #TST26, NEXT  
MOV TXDBUF, R3 ; SET REGISTER TO BE TESTED.  
MOV #BIT7, R5 ; SET "EXPECTED"  
MOV R5, (R3) ; WRITE THE REGISTER.  
MOV (R3), R4 ; READ THE REGISTER.  
CMP R5, R4 ; R5=GOOD; R4=UNKNOWN.  
BEQ 1$ ; ARE THEY THE SAME?  
HLT 3 ; COMPARISON ERROR.  
1$: BIC R5, (R3) ; CLEAR BIT7  
MOV (R3), R4 ; READ THE REGISTER.  
CLR R5 ; SET "EXPECTED"  
CMP R5, R4 ; R5=GOOD; R4=?  
BEQ 2$ ; BR IF OK  
HLT 3 ; COMPARISON ERROR  
2$: SCOPE ; SCOPE THIS TEST
```

***** TEST 26 *****
*TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
*SET BIT8, VERIFY BIT8 WAS SET.
*CLEAR BIT8, VERIFY BIT8 WAS CLEARED.

011144 012737 000026 001226
011152 012737 011220 001216
011160 013703 001414
011164 012705 000400
011170 010513
011172 011304
011174 020504
011176 001401
011200 104003
011202 040513
011204 011304
011206 005005
011210 020504
011212 001401
011214 104003
011216 104400

```
*****  
*  
: TEST 26  
*  
*****  
*****  
TST26: MOV #26, J#TSTNO  
MOV #TST27, NEXT  
MOV TXDBUF, R3 ; SET REGISTER TO BE TESTED.  
MOV #BIT8, R5 ; SET "EXPECTED"  
MOV R5, (R3) ; WRITE THE REGISTER.  
MOV (R3), R4 ; READ THE REGISTER.  
CMP R5, R4 ; R5=GOOD; R4=UNKNOWN.  
BEQ 1$ ; ARE THEY THE SAME?  
HLT 3 ; COMPARISON ERROR.  
1$: BIC R5, (R3) ; CLEAR BIT8  
MOV (R3), R4 ; READ THE REGISTER.  
CLR R5 ; SET "EXPECTED"  
CMP R5, R4 ; R5=GOOD; R4=?  
BEQ 2$ ; BR IF OK  
HLT 3 ; COMPARISON ERROR  
2$: SCOPE ; SCOPE THIS TEST
```

***** TEST 27 *****

```

3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069 011220 012737 000027 001226
3070 011226 012737 011274 001216
3071 011234 013703 001414
3072 011240 012705 001000
3073 011244 010513
3074 011246 011304
3075 011250 020504
3076 011252 001401
3077 011254 104003
3078 011256 040513
3079 011260 011304
3080 011262 005005
3081 011264 020504
3082 011266 001401
3083 011270 104003
3084 011272 104400
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099 011274 012737 000030 001226
3100 011302 012737 011350 001216
3101 011310 013703 001414
3102 011314 012705 002000
3103 011320 010513
3104 011322 011304
3105 011324 020504
3106 011326 001401
3107 011330 104003
3108 011332 040513
3109 011334 011304
3110 011336 005005
3111 011340 020504
3112 011342 001401
3113 011344 104003

```

```

; *TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
; *SET BIT9, VERIFY BIT9 WAS SET.
; *CLEAR BIT9, VERIFY BIT9 WAS CLEARED.
; *****

; *****
; TEST 27
; *****
; *****
TST27:  MOV    #27, #TSTNO
        MOV    #TST30, NEXT
        MOV    TXDBUF, R3
        MOV    #BIT9, R5
        MOV    R5, (R3)
        MOV    (R3), R4
        CMP    R5, R4
        BEQ    1$
        HLT    3
1$:     BIC    R5, (R3)
        MOV    (R3), R4
        CLR    R5
        CMP    R5, R4
        BEQ    2$
        HLT    3
2$:     SCOPE

; SET REGISTER TO BE TESTED.
; SET "EXPECTED"
; WRITE THE REGISTER.
; READ THE REGISTER.
; R5=GOOD; R4=UNKNOWN.
; ARE THEY THE SAME?
; COMPARISON ERROR.
; CLEAR BIT9
; READ THE REGISTER.
; SET "EXPECTED"
; R5=GOOD; R4=?
; BR IF OK
; COMPARISON ERROR
; SCOPE THIS TEST

```

```

; ***** TEST 30 *****
; *TRANSMITTER DATA BUFFER REGISTER READ/WRITE BIT TEST.
; *SET BIT10, VERIFY BIT10 WAS SET.
; *CLEAR BIT10, VERIFY BIT10 WAS CLEARED.
; *****

```

```

; *****
; TEST 30
; *****
; *****
TST30:  MOV    #30, #TSTNO
        MOV    #TST31, NEXT
        MOV    TXDBUF, R3
        MOV    #BIT10, R5
        MOV    R5, (R3)
        MOV    (R3), R4
        CMP    R5, R4
        BEQ    1$
        HLT    3
1$:     BIC    R5, (R3)
        MOV    (R3), R4
        CLR    R5
        CMP    R5, R4
        BEQ    2$
        HLT    3
2$:     SCOPE

; SET REGISTER TO BE TESTED.
; SET "EXPECTED"
; WRITE THE REGISTER.
; READ THE REGISTER.
; R5=GOOD; R4=UNKNOWN.
; ARE THEY THE SAME?
; COMPARISON ERROR.
; CLEAR BIT10
; READ THE REGISTER.
; SET "EXPECTED"
; R5=GOOD; R4=?
; BR IF OK
; COMPARISON ERROR

```

3114 011346 104400

2\$: SCOPE ;SCOPE THIS TEST

3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169

011350 012737 000031 001226
011356 012737 011444 001216
011364 013703 001404
011370 052713 000002
011374 005005
011376 052777 000400 170006
011404 004737 005044
011410 032713 000002
011414 001402
011416 011304
011420 104003
011422 052713 000002
011426 005013
011430 032713 000002
011434 001402
011436 011304
011440 104003
011442 104400

***** TEST 31 *****
:RECEIVER CONTROL REGISTER READ/WRITE BIT 1 RESET AND CLEAR TEST
:WRITE BIT 1 AND TEST THAT IT WILL BE CLEARED AFTER A
:DEVICE RESET AND RECEIVER CONTROL REGISTER CLR INSTRUCTION

: TEST 31

TST31: MOV #31, @TSTNO
MOV #TST32, NEXT
MOV RXCSR, R3 ;GET THE RECEIVER CONTROL REGISTER
BIS #BIT1, (R3) ;SET BIT 1 AT RECEIVER CONTROL REGISTER
CLR R5 ;SET "EXPECTED"
BIS #MRESET, @TXCSR ;RESET THE DEVICE
JSR PC, SMALL ;WAIT FOR RESET TO FINISH
BIT #BIT1, (R3) ;TEST BIT 1
BEQ 1\$;BIT 1 IS CLEARED
MOV (R3), R4 ;LOAD "FOUND"
HLT 3 ;BIT 1 IS SET AND SHOULDN'T BE
1\$: BIS #BIT1, (R3) ;SET BIT 1 AGAIN
CLR (R3) ;CLEAR THE RECEIVER CONTROL REGISTER
BIT #BIT1, (R3) ;TEST TO SEE IF BIT 1 CLEARED
BEQ 2\$;BIT 1 IS OK
MOV (R3), R4 ;LOAD "FOUND"
HLT 3 ;BIT 1 FAILED TO CLEAR
2\$: SCOPE ;SCOPE THIS TEST

***** TEST 32 *****
:RECEIVER CONTROL REGISTER READ/WRITE BIT 2 RESET AND CLEAR TEST
:WRITE BIT 2 AND TEST THAT IT WILL BE CLEARED AFTER A
:DEVICE RESET AND RECEIVER CONTROL REGISTER CLR INSTRUCTION

: TEST 32

011444 012737 000032 001226
011452 012737 011540 001216
011460 013703 001404
011464 052713 000004
011470 005005
011472 052777 000400 167712
011500 004737 005044
011504 032713 000004
011510 001402

TST32: MOV #32, @TSTNO
MOV #TST33, NEXT
MOV RXCSR, R3 ;GET THE RECEIVER CONTROL REGISTER
BIS #BIT2, (R3) ;SET BIT 2 AT RECEIVER CONTROL REGISTER
CLR R5 ;SET "EXPECTED"
BIS #MRESET, @TXCSR ;RESET THE DEVICE
JSR PC, SMALL ;WAIT FOR RESET TO FINISH
BIT #BIT2, (R3) ;TEST BIT 2
BEQ 1\$;BIT 2 IS CLEARED

RECEIVER CONTROL REGISTER RESET AND CLEAR TEST BIT 2

```

3170 011512 011304      MOV      (R3),R4      ;LOAD "FOUND"
3171 011514 104003      HLT      3            ;BIT 2 IS SET AND SHOULDN'T BE
3172 011516 052713 000004 1$:      BIS      #BIT2,(R3)  ;SET BIT 2 AGAIN
3173 011522 005013      CLR      (R3)        ;CLEAR THE RECEIVER CONTROL REGISTER
3174 011524 032713 000004      BIT      #BIT2,(R3)  ;TEST TO SEE IF BIT 2 CLEARED
3175 011530 001402      BEQ      2$          ;BIT 2 IS OK
3176 011532 011304      MOV      (R3),R4      ;LOAD "FOUND"
3177 011534 104003      HLT      3            ;BIT 2 FAILED TO CLEAR
3178 011536 104400      2$:      SCOPE        ;SCOPE THIS TEST
3179
3180

```

```

:***** TEST 33 *****
:*RECEIVER CONTROL REGISTER READ/WRITE BIT 3 RESET AND CLEAR TEST
:*WRITE BIT 3, AND TEST THAT IT WILL BE CLEARED AFTER A
:*DEVICE RESET AND RECEIVER CONTROL REGISTER CLR INSTRUCTION
:*****

```

```

:*****
: TEST 33
:*****

```

```

3192
3193 011540 012737 000033 001226 TST33: MOV      #33,2#TSTNO
3194 011546 012737 011634 001216      MOV      #TST34,NEXT
3195 011554 013703 001404      MOV      RXCSR,R3    ;GET THE RECEIVER CONTROL REGISTER
3196 011560 052713 000010      BIS      #BIT3,(R3)  ;SET BIT 3 AT RECEIVER CONTROL REGISTER
3197 011564 005005      CLR      R5          ;SET "EXPECTED"
3198 011566 052777 000400 167616      BIS      #MRESET,2#TXCSR ;RESET THE DEVICE
3199 011574 004737 005044      JSR      PC_SMALL    ;WAIT FOR RESET TO FINISH
3200 011600 032713 000010      BIT      #BIT3,(R3)  ;TEST BIT 3
3201 011604 001402      BEQ      1$          ;BIT 3 IS CLEARED
3202 011606 011304      MOV      (R3),R4      ;LOAD "FOUND"
3203 011610 104003      HLT      3            ;BIT 3 IS SET AND SHOULDN'T BE
3204 011612 052713 000010 1$:      BIS      #BIT3,(R3)  ;SET BIT 3 AGAIN
3205 011616 005013      CLR      (R3)        ;CLEAR THE RECEIVER CONTROL REGISTER
3206 011620 032713 000010      BIT      #BIT3,(R3)  ;TEST TO SEE IF BIT 3 CLEARED
3207 011624 001402      BEQ      2$          ;BIT 3 IS OK
3208 011626 011304      MOV      (R3),R4      ;LOAD "FOUND"
3209 011630 104003      HLT      3            ;BIT 3 FAILED TO CLEAR
3210 011632 104400      2$:      SCOPE        ;SCOPE THIS TEST
3211
3212

```

```

:***** TEST 34 *****
:*RECEIVER CONTROL REGISTER READ/WRITE BIT 4 RESET AND CLEAR TEST
:*WRITE BIT 4, AND TEST THAT IT WILL BE CLEARED AFTER A
:*DEVICE RESET AND RECEIVER CONTROL REGISTER CLR INSTRUCTION
:*****

```

```

:*****
: TEST 34
:*****

```

```

3225 011634 012737 000034 001226 TST34: MOV      #34,2#TSTNO

```


RECEIVER CONTROL REGISTER RESET AND CLEAR TEST BIT 4

```

3226 011642 012737 011730 001216      MOV      #TST35, NEXT
3227 011650 013703 001404          MOV      RXCSR, R3          ;GET THE RECEIVER CONTROL REGISTER
3228 011654 052713 000020      BIS      #BIT4, (R3)      ;SET BIT 4 AT RECEIVER CONTROL REGISTER
3229 011660 005005          CLR      R5              ;SET "EXPECTED"
3230 011662 052777 000400 167522      BIS      #MRESET, @TXCSR ;RESET THE DEVICE
3231 011670 004737 005044          JSR      PC, SMALL       ;WAIT FOR RESET TO FINISH
3232 011674 032713 000020      BIT      #BIT4, (R3)     ;TEST BIT 4
3233 011700 001402          BEQ      1$             ;BIT 4 IS CLEARED
3234 011702 011304          MOV      (R3), R4       ;LOAD "FOUND"
3235 011704 104003          HLT      3              ;BIT 4 IS SET AND SHOULDN'T BE
3236 011706 052713 000020 1$:      BIS      #BIT4, (R3)     ;SET BIT 4 AGAIN
3237 011712 005013          CLR      (R3)          ;CLEAR THE RECEIVER CONTROL REGISTER
3238 011714 032713 000020      BIT      #BIT4, (R3)     ;TEST TO SEE IF BIT 4 CLEARED
3239 011720 001402          BEQ      2$             ;BIT 4 IS OK
3240 011722 011304          MOV      (R3), R4       ;LOAD "FOUND"
3241 011724 104003          HLT      3              ;BIT 4 FAILED TO CLEAR
3242 011726 104400 2$:      SCOPE                ;SCOPE THIS TEST

```

```

***** TEST 35 *****
*RECEIVER CONTROL REGISTER READ/WRITE BIT 8 RESET AND CLEAR TEST
*WRITE BIT 8 AND TEST THAT IT WILL BE CLEARED AFTER A
*DEVICE RESET AND RECEIVER CONTROL REGISTER CLR INSTRUCTION
*****

```

```

*****
*
* TEST 35
*
*****

```

```

3251 *****
3252 *
3253 * TEST 35
3254 *
3255 *****
3256 *****
3257 011730 012737 000035 001226 1ST35: MOV      #35, @TSTNO
3258 011736 012737 012024 001216      MOV      #TST36, NEXT
3259 011744 013703 001404          MOV      RXCSR, R3          ;GET THE RECEIVER CONTROL REGISTER
3260 011750 052713 000400      BIS      #BIT8, (R3)      ;SET BIT 8 AT RECEIVER CONTROL REGISTER
3261 011754 005005          CLR      R5              ;SET "EXPECTED"
3262 011756 052777 000400 167426      BIS      #MRESET, @TXCSR ;RESET THE DEVICE
3263 011764 004737 005044          JSR      PC, SMALL       ;WAIT FOR RESET TO FINISH
3264 011770 032713 000400      BIT      #BIT8, (R3)     ;TEST BIT 8
3265 011774 001402          BEQ      1$             ;BIT 8 IS CLEARED
3266 011776 011304          MOV      (R3), R4       ;LOAD "FOUND"
3267 012000 104003          HLT      3              ;BIT 8 IS SET AND SHOULDN'T BE
3268 012002 052713 000400 1$:      BIS      #BIT8, (R3)     ;SET BIT 8 AGAIN
3269 012006 005013          CLR      (R3)          ;CLEAR THE RECEIVER CONTROL REGISTER
3270 012010 032713 000400      BIT      #BIT8, (R3)     ;TEST TO SEE IF BIT 8 CLEARED
3271 012014 001402          BEQ      2$             ;BIT 8 IS OK
3272 012016 011304          MOV      (R3), R4       ;LOAD "FOUND"
3273 012020 104003          HLT      3              ;BIT 8 FAILED TO CLEAR
3274 012022 104400 2$:      SCOPE                ;SCOPE THIS TEST

```

```

***** TEST 36 *****
*TRANSMITTER CONTROL REGISTER READ/WRITE BIT 3 RESET AND CLEAR TEST
*WRITE BIT 3 AND TEST THAT IT WILL BE CLEARED AFTER A
*DEVICE RESET AND TRANSMITTER CONTROL REGISTER CLR INSTRUCTION
*****

```

```

3275
3276
3277
3278
3279
3280
3281

```

TRANSMITTER CONTROL REGISTER RESET AND CLEAR TEST BIT 3

3282
3283
3284
3285
3286
3287
3288
3289
3290
3291
3292
3293
3294
3295
3296
3297
3298
3299
3300
3301
3302
3303
3304
3305
3306
3307
3308
3309
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337

012024 012737 000036 001226
012032 012737 012120 001216
012040 013703 001412
012044 052713 000010
012050 005005
012052 052777 000400 167332
012060 004737 005044
012064 032713 000010
012070 001402
012072 011304
012074 104003
012076 052713 000010
012102 005013
012104 032713 000010
012110 001402
012112 011304
012114 104003
012116 104400

```
*****  
*  
: TEST 36  
*  
*****  
*****  
↑ST36: MOV #36, @TSTNO  
MOV #TST37, NEXT  
MOV TXCSR, R3 ;GET THE TRANSMITTER CONTROL REGISTER  
BIS #BIT3, (R3) ;SET BIT 3 AT TRANSMITTER CONTROL REGISTER  
CLR R5 ;SET "EXPECTED"  
BIS #MRESET, @TXCSR ;RESET THE DEVICE  
JSR PC, SMALL ;WAIT FOR RESET TO FINISH  
BIT #BIT3, (R3) ;TEST BIT 3  
BEQ 1$ ;BIT 3 IS CLEARED  
MOV (R3), R4 ;LOAD "FOUND"  
HLT 3 ;BIT 3 IS SET AND SHOULDN'T BE  
1$: BIS #BIT3, (R3) ;SET BIT 3 AGAIN  
CLR (R3) ;CLEAR THE TRANSMITTER CONTROL REGISTER  
BIT #BIT3, (R3) ;TEST TO SEE IF BIT 3 CLEARED  
BEQ 2$ ;BIT 3 IS OK  
MOV (R3), R4 ;LOAD "FOUND"  
HLT 3 ;BIT 3 FAILED TO CLEAR  
2$: SCOPE ;SCOPE THIS TEST
```

***** TEST 37 *****
*TRANSMITTER CONTROL REGISTER READ/WRITE BIT 4 RESET AND CLEAR TEST
*WRITE BIT 4, AND TEST THAT IT WILL BE CLEARED AFTER A
*DEVICE RESET AND TRANSMITTER CONTROL REGISTER CLR INSTRUCTION

```
*****  
*  
: TEST 37  
*  
*****  
*****  
↑ST37: MOV #37, @TSTNO  
MOV #TST40, NEXT  
MOV TXCSR, R3 ;GET THE TRANSMITTER CONTROL REGISTER  
BIS #BIT4, (R3) ;SET BIT 4 AT TRANSMITTER CONTROL REGISTER  
CLR R5 ;SET "EXPECTED"  
BIS #MRESET, @TXCSR ;RESET THE DEVICE  
JSR PC, SMALL ;WAIT FOR RESET TO FINISH  
BIT #BIT4, (R3) ;TEST BIT 4  
BEQ 1$ ;BIT 4 IS CLEARED  
MOV (R3), R4 ;LOAD "FOUND"  
HLT 3 ;BIT 4 IS SET AND SHOULDN'T BE  
1$: BIS #BIT4, (R3) ;SET BIT 4 AGAIN  
CLR (R3) ;CLEAR THE TRANSMITTER CONTROL REGISTER  
BIT #BIT4, (R3) ;TEST TO SEE IF BIT 4 CLEARED  
BEQ 2$ ;BIT 4 IS OK  
MOV (R3), R4 ;LOAD "FOUND"  
HLT 3 ;BIT 4 FAILED TO CLEAR
```

TRANSMITTER CONTROL REGISTER RESET AND CLEAR TEST BIT 4

3338 012212 104400

2\$: SCOPE ;SCOPE THIS TEST

3339
3340
3341
3342
3343
3344
3345
3346
3347
3348
3349
3350
3351
3352
3353
3354
3355
3356
3357
3358
3359
3360
3361
3362
3363
3364
3365
3366
3367
3368
3369
3370
3371
3372
3373
3374
3375
3376
3377
3378
3379
3380
3381
3382
3383
3384
3385
3386
3387
3388
3389
3390
3391
3392
3393

012214 012737 000040 001226
012222 012737 012310 001216
012230 013703 001412
012234 052713 002000
012240 005005
012242 052777 000400 167142
012250 004737 005044
012254 032713 002000
012260 001402
012262 011304
012264 104003
012266 052713 002000
012272 005013
012274 032713 002000
012300 001402
012302 011304
012304 104003
012306 104400

***** TEST 40 *****
*TRANSMITTER CONTROL REGISTER READ/WRITE BIT 10 RESET AND CLEAR TEST
*WRITE BIT 10, AND TEST THAT IT WILL BE CLEARED AFTER A
*DEVICE RESET AND TRANSMITTER CONTROL REGISTER CLR INSTRUCTION

*
TEST 40
*

TST40: MOV #40, @TSTNO ;GET THE TRANSMITTER CONTROL REGISTER
MOV @TST41, NEXT ;SET BIT 10 AT TRANSMITTER CONTROL REGISTER
MOV TXCSR, R3 ;SET "EXPECTED"
BIS #BIT10, (R3) ;RESET THE DEVICE
CLR R5 ;WAIT FOR RESET TO FINISH
BIS #MRESET, @TXCSR ;TEST BIT 10
JSR PC, SMALL ;BIT 10 IS CLEARED
BIT #BIT10, (R3) ;LOAD "FOUND"
BEQ 1\$;BIT 10 IS SET AND SHOULDN'T BE
MOV (R3), R4 ;SET BIT 10 AGAIN
HLT 3 ;CLEAR THE TRANSMITTER CONTROL REGISTER
1\$: BIS #BIT10, (R3) ;TEST TO SEE IF BIT 10 CLEARED
CLR (R3) ;BIT 10 IS OK
BIT #BIT10, (R3) ;LOAD "FOUND"
BEQ 2\$;BIT 10 FAILED TO CLEAR
MOV (R3), R4 ;SCOPE THIS TEST
HLT 3
2\$: SCOPE

***** TEST 41 *****
*TRANSMITTER CONTROL REGISTER READ/WRITE BIT 11 RESET AND CLEAR TEST
*WRITE BIT 11, AND TEST THAT IT WILL BE CLEARED AFTER A
*DEVICE RESET AND TRANSMITTER CONTROL REGISTER CLR INSTRUCTION

*
TEST 41
*

012310 012737 000041 001226
012316 012737 012404 001216
012324 013703 001412
012330 052713 004000
012334 005005
012336 052777 000400 167046
012344 004737 005044
012350 032713 004000
012354 001402

TST41: MOV #41, @TSTNO ;GET THE TRANSMITTER CONTROL REGISTER
MOV @TST42, NEXT ;SET BIT 11 AT TRANSMITTER CONTROL REGISTER
MOV TXCSR, R3 ;SET "EXPECTED"
BIS #BIT11, (R3) ;RESET THE DEVICE
CLR R5 ;WAIT FOR RESET TO FINISH
BIS #MRESET, @TXCSR ;TEST BIT 11
JSR PC, SMALL ;BIT 11 IS CLEARED
BIT #BIT11, (R3)
BEQ 1\$

```

3394 012356 011304      MOV      (R3),R4      ;LOAD "FOUND"
3395 012360 104003      HLT      3            ;BIT 11 IS SET AND SHOULDN'T BE
3396 012362 052713 004000 1$:  BIS      #BIT11,(R3) ;SET BIT 11 AGAIN
3397 012366 005013      CLR      (R3)        ;CLEAR THE TRANSMITTER CONTROL REGISTER
3398 012370 032713 004000  BIT      #BIT11,(R3) ;TEST TO SEE IF BIT 11 CLEARED
3399 012374 001402      BEQ     2$          ;BIT 11 IS OK
3400 012376 011304      MOV      (R3),R4      ;LOAD "FOUND"
3401 012400 104003      HLT      3            ;BIT 11 FAILED TO CLEAR
3402 012402 104400      2$:  SCOPE          ;SCOPE THIS TEST

```

```

:***** TEST 42 *****
:*TRANSMITTER CONTROL REGISTER READ/WRITE BIT 12 RESET AND CLEAR TEST
:*WRITE BIT 12,AND TEST THAT IT WILL BE CLEARED AFTER A
:*DEVICE RESET AND TRANSMITTER CONTROL REGISTER CLR INSTRUCTION
:*****

```

```

:*****
: TEST 42
:*****

```

```

3417 012404 012737 000042 001226 TST42: MOV      #42,#TSTNO
3418 012412 012737 012500 001216  MOV      #TST43,NEXT
3419 012420 013703 001412  MOV      TXCSR,R3      ;GET THE TRANSMITTER CONTROL REGISTER
3420 012424 052713 010000  BIS      #BIT12,(R3)   ;SET BIT 12 AT TRANSMITTER CONTROL REGISTER
3421 012430 005005  CLR      R5            ;SET "EXPECTED"
3422 012432 052777 000400 166752  BIS      #MRESET,#TXCSR ;RESET THE DEVICE
3423 012440 004737 005044  JSR     PC,SMALL      ;WAIT FOR RESET TO FINISH
3424 012444 032713 010000  BIT      #BIT12,(R3)   ;TEST BIT 12
3425 012450 001402  BEQ     1$          ;BIT 12 IS CLEARED
3426 012452 011304  MOV      (R3),R4      ;LOAD "FOUND"
3427 012454 104003  HLT      3            ;BIT 12 IS SET AND SHOULDN'T BE
3428 012456 052713 010000 1$:  BIS      #BIT12,(R3) ;SET BIT 12 AGAIN
3429 012462 005013  CLR      (R3)        ;CLEAR THE TRANSMITTER CONTROL REGISTER
3430 012464 032713 010000  BIT      #BIT12,(R3)   ;TEST TO SEE IF BIT 12 CLEARED
3431 012470 001402  BEQ     2$          ;BIT 12 IS OK
3432 012472 011304  MOV      (R3),R4      ;LOAD "FOUND"
3433 012474 104003  HLT      3            ;BIT 12 FAILED TO CLEAR
3434 012476 104400  2$:  SCOPE          ;SCOPE THIS TEST

```

```

:***** TEST 43 *****
:*TRANSMITTER CONTROL REGISTER READ/WRITE BIT 13 RESET AND CLEAR TEST
:*WRITE BIT 13,AND TEST THAT IT WILL BE CLEARED AFTER A
:*DEVICE RESET AND TRANSMITTER CONTROL REGISTER CLR INSTRUCTION
:*****

```

```

:*****
: TEST 43
:*****

```

```

3449 012500 012737 000043 001226 TST43: MOV      #43,#TSTNO

```

TRANSMITTER CONTROL REGISTER RESET AND CLEAR TEST BIT 13

```

3450 012506 012737 012574 001216      MOV      #TST44,NEXT
3451 012514 013703 001412      MOV      TXCSR,R3          ;GET THE TRANSMITTER CONTROL REGISTER
3452 012520 052713 020000      BIS      #BIT13,(R3)      ;SET BIT 13 AT TRANSMITTER CONTROL REGISTER
3453 012524 005005                CLR      R5                ;SET "EXPECTED"
3454 012526 052777 000400 166656      BIS      #MRESET,@TXCSR  ;RESET THE DEVICE
3455 012534 004737 005044                JSR      PC,SMALL         ;WAIT FOR RESET TO FINISH
3456 012540 032713 020000      BIT      #BIT13,(R3)     ;TEST BIT 13
3457 012544 001402                BEQ      1$              ;BIT 13 IS CLEARED
3458 012546 011304                MOV      (R3),R4         ;LOAD "FOUND"
3459 012550 104003                HLT      3                ;BIT 13 IS SET AND SHOULDN'T BE
3460 012552 052713 020000 1$:      BIS      #BIT13,(R3)     ;SET BIT 13 AGAIN
3461 012556 005013                CLR      (R3)            ;CLEAR THE TRANSMITTER CONTROL REGISTER
3462 012560 032713 020000      BIT      #BIT13,(R3)     ;TEST TO SEE IF BIT 13 CLEARED
3463 012564 001402                BEQ      2$              ;BIT 13 IS OK
3464 012566 011304                MOV      (R3),R4         ;LOAD "FOUND"
3465 012570 104003                HLT      3                ;BIT 13 FAILED TO CLEAR
3466 012572 104400 2$:      SCOPE                    ;SCOPE THIS TEST

```

```

:***** TEST 44 *****
:*TRANSMITTER BUFFER REGISTER READ/WRITE BIT 0 RESET AND CLEAR TEST
:*WRITE BIT 0 AND TEST THAT IT WILL BE CLEARED AFTER A
:*DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION
:*****

```

```

:*****
:TEST 44
:*****

```

```

3481 012574 012737 000044 001226 TST44: MOV      #44,@TSTNO
3482 012602 012737 012670 001216      MOV      #TST45,NEXT
3483 012610 013703 001414      MOV      TXDBUF,R3       ;GET THE TRANSMITTER BUFFER REGISTER
3484 012614 052713 000001      BIS      #BIT0,(R3)      ;SET BIT 0 AT TRANSMITTER BUFFER REGISTER
3485 012620 005005                CLR      R5                ;SET "EXPECTED"
3486 012622 052777 000400 166562      BIS      #MRESET,@TXCSR  ;RESET THE DEVICE
3487 012630 004737 005044                JSR      PC,SMALL         ;WAIT FOR RESET TO FINISH
3488 012634 032713 000001      BIT      #BIT0,(R3)     ;TEST BIT 0
3489 012640 001402                BEQ      1$              ;BIT 0 IS CLEARED
3490 012642 011304                MOV      (R3),R4         ;LOAD "FOUND"
3491 012644 104003                HLT      3                ;BIT 0 IS SET AND SHOULDN'T BE
3492 012646 052713 000001 1$:      BIS      #BIT0,(R3)     ;SET BIT 0 AGAIN
3493 012652 005013                CLR      (R3)            ;CLEAR THE TRANSMITTER BUFFER REGISTER
3494 012654 032713 000001      BIT      #BIT0,(R3)     ;TEST TO SEE IF BIT 0 CLEARED
3495 012660 001402                BEQ      2$              ;BIT 0 IS OK
3496 012662 011304                MOV      (R3),R4         ;LOAD "FOUND"
3497 012664 104003                HLT      3                ;BIT 0 FAILED TO CLEAR
3498 012666 104400 2$:      SCOPE                    ;SCOPE THIS TEST

```

```

:***** TEST 45 *****
:*TRANSMITTER BUFFER REGISTER READ/WRITE BIT 1 RESET AND CLEAR TEST
:*WRITE BIT 1 AND TEST THAT IT WILL BE CLEARED AFTER A
:*DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION
:*****

```

```

3499
3500
3501
3502
3503
3504
3505

```

3506
3507
3508
3509
3510
3511
3512
3513
3514
3515
3516
3517
3518
3519
3520
3521
3522
3523
3524
3525
3526
3527
3528
3529
3530
3531
3532
3533
3534
3535
3536
3537
3538
3539
3540
3541
3542
3543
3544
3545
3546
3547
3548
3549
3550
3551
3552
3553
3554
3555
3556
3557
3558
3559
3560
3561

012670 012737 000045 001226
012676 012737 012764 001216
012704 013703 001414
012710 052713 000002
012714 005005
012716 052777 000400 166466
012724 004737 005044
012730 032713 000002
012734 001402
012736 011304
012740 104003
012742 052713 000002
012746 005013
012750 032713 000002
012754 001402
012756 011304
012760 104003
012762 104400

```
::*****  
: TEST 45  
:*****  
:*****  
TST45: MOV #45,@TSTNO  
MOV #TST46,NEXT  
MOV TXDBUF,R3 ;GET THE TRANSMITTER BUFFER REGISTER  
BIS #BIT1,(R3) ;SET BIT 1 AT TRANSMITTER BUFFER REGISTER  
CLR R5 ;SET "EXPECTED"  
BIS #MRESET,@TXCSR ;RESET THE DEVICE  
JSR PC,SMALL ;WAIT FOR RESET TO FINISH  
BIT #BIT1,(R3) ;TEST BIT 1  
BEQ 1$ ;BIT 1 IS CLEARED  
MOV (R3),R4 ;LOAD "FOUND"  
HLT 3 ;BIT 1 IS SET AND SHOULDN'T BE  
1$: BIS #BIT1,(R3) ;SET BIT 1 AGAIN  
CLR (R3) ;CLEAR THE TRANSMITTER BUFFER REGISTER  
BIT #BIT1,(R3) ;TEST TO SEE IF BIT 1 CLEARED  
BEQ 2$ ;BIT 1 IS OK  
MOV (R3),R4 ;LOAD "FOUND"  
HLT 3 ;BIT 1 FAILED TO CLEAR  
2$: SCOPE ;SCOPE THIS TEST
```

***** TEST 46 *****
*TRANSMITTER BUFFER REGISTER READ/WRITE BIT 2 RESET AND CLEAR TEST
*WRITE BIT 2 AND TEST THAT IT WILL BE CLEARED AFTER A
*DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION

```
::*****  
: TEST 46  
:*****  
:*****  
TST46: MOV #46,@TSTNO  
MOV #TST47,NEXT  
MOV TXDBUF,R3 ;GET THE TRANSMITTER BUFFER REGISTER  
BIS #BIT2,(R3) ;SET BIT 2 AT TRANSMITTER BUFFER REGISTER  
CLR R5 ;SET "EXPECTED"  
BIS #MRESET,@TXCSR ;RESET THE DEVICE  
JSR PC,SMALL ;WAIT FOR RESET TO FINISH  
BIT #BIT2,(R3) ;TEST BIT 2  
BEQ 1$ ;BIT 2 IS CLEARED  
MOV (R3),R4 ;LOAD "FOUND"  
HLT 3 ;BIT 2 IS SET AND SHOULDN'T BE  
1$: BIS #BIT2,(R3) ;SET BIT 2 AGAIN  
CLR (R3) ;CLEAR THE TRANSMITTER BUFFER REGISTER  
BIT #BIT2,(R3) ;TEST TO SEE IF BIT 2 CLEARED  
BEQ 2$ ;BIT 2 IS OK  
MOV (R3),R4 ;LOAD "FOUND"  
HLT 3 ;BIT 2 FAILED TO CLEAR
```

TRANSMITTER BUFFER REGISTER RESET AND CLEAR TEST BIT 2

3562 013056 104400

2\$: SCOPE ;SCOPE THIS TEST

3563
3564
3565
3566
3567
3568
3569
3570
3571
3572
3573
3574
3575
3576
3577
3578
3579
3580
3581
3582
3583
3584
3585
3586
3587
3588
3589
3590
3591
3592
3593
3594
3595
3596
3597
3598
3599
3600
3601
3602
3603
3604
3605
3606
3607
3608
3609
3610
3611
3612
3613
3614
3615
3616
3617

013060 012737 000047 001226
013066 012737 013154 001216
013074 013703 001414
013100 052713 000010
013104 005005
013106 052777 000400 166276
013114 004737 005044
013120 032713 000010
013124 001402
013126 011304
013130 104003
013132 052713 000010
013136 005013
013140 032713 000010
013144 001402
013146 011304
013150 104003
013152 104400

***** TEST 47 *****
*TRANSMITTER BUFFER REGISTER READ/WRITE BIT 3 RESET AND CLEAR TEST
*WRITE BIT 3 AND TEST THAT IT WILL BE CLEARED AFTER A
*DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION

*
TEST 47
*

TST47: MOV #47, @TSTNO
MOV #TST50, NEXT
MOV TXDBUF, R3 ;GET THE TRANSMITTER BUFFER REGISTER
BIS #BIT3, (R3) ;SET BIT 3 AT TRANSMITTER BUFFER REGISTER
CLR R5 ;SET "EXPECTED"
BIS #MRESET, @TXCSR ;RESET THE DEVICE
JSR PC, SMALL ;WAIT FOR RESET TO FINISH
BIT #BIT3, (R3) ;TEST BIT 3
BEQ 1\$;BIT 3 IS CLEARED
MOV (R3), R4 ;LOAD "FOUND"
HLT 3 ;BIT 3 IS SET AND SHOULDN'T BE
1\$: BIS #BIT3, (R3) ;SET BIT 3 AGAIN
CLR (R3) ;CLEAR THE TRANSMITTER BUFFER REGISTER
BIT #BIT3, (R3) ;TEST TO SEE IF BIT 3 CLEARED
BEQ 2\$;BIT 3 IS OK
MOV (R3), R4 ;LOAD "FOUND"
HLT 3 ;BIT 3 FAILED TO CLEAR
2\$: SCOPE ;SCOPE THIS TEST

***** TEST 50 *****
*TRANSMITTER BUFFER REGISTER READ/WRITE BIT 4 RESET AND CLEAR TEST
*WRITE BIT 4 AND TEST THAT IT WILL BE CLEARED AFTER A
*DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION

*
TEST 50
*

013154 012737 000050 001226
013162 012737 013250 001216
013170 013703 001414
013174 052713 000020
013200 005005
013202 052777 000400 166202
013210 004737 005044
013214 032713 000020
013220 001402

TST50: MOV #50, @TSTNO
MOV #TST51, NEXT
MOV TXDBUF, R3 ;GET THE TRANSMITTER BUFFER REGISTER
BIS #BIT4, (R3) ;SET BIT 4 AT TRANSMITTER BUFFER REGISTER
CLR R5 ;SET "EXPECTED"
BIS #MRESET, @TXCSR ;RESET THE DEVICE
JSR PC, SMALL ;WAIT FOR RESET TO FINISH
BIT #BIT4, (R3) ;TEST BIT 4
BEQ 1\$;BIT 4 IS CLEARED
1\$: SCOPE ;SCOPE THIS TEST

TRANSMITTER BUFFER REGISTER RESET AND CLEAR TEST BIT 4

```

3618 013222 011304      MOV      (R3),R4      ;LOAD "FOUND"
3619 013224 104003      HLT      3            ;BIT 4 IS SET AND SHOULDN'T BE
3620 013226 052713 000020 1$:      BIS      #BIT4,(R3)  ;SET BIT 4 AGAIN
3621 013232 005013      CLR      (R3)        ;CLEAR THE TRANSMITTER BUFFER REGISTER
3622 013234 032713 000020      BIT      #BIT4,(R3)  ;TEST TO SEE IF BIT 4 CLEARED
3623 013240 001402      BEQ      2$          ;BIT 4 IS OK
3624 013242 011304      MOV      (R3),R4      ;LOAD "FOUND"
3625 013244 104003      HLT      3            ;BIT 4 FAILED TO CLEAR
3626 013246 104400 2$:      SCOPE              ;SCOPE THIS TEST

```

```

3627
3628
3629
3630
3631
3632
3633
3634
3635
3636
3637
3638
3639
3640
3641
3642
3643
3644
3645
3646
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673

```

```

:***** TEST 51 *****
:*TRANSMITTER BUFFER REGISTER READ/WRITE BIT 5 RESET AND CLEAR TEST
:*WRITE BIT 5, AND TEST THAT IT WILL BE CLEARED AFTER A
:*DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION
:*****

```

```

:*****
:TEST 51
:*****
:*****

```

```

3641 013250 012737 000051 001226 TST51: MOV      #51,@TSTNO
3642 013256 012737 013344 001216      MOV      #TST52,NEXT
3643 013264 013703 001414      MOV      TXDBUF,R3    ;GET THE TRANSMITTER BUFFER REGISTER
3644 013270 052713 000040      BIS      #BITS,(R3)  ;SET BIT 5 AT TRANSMITTER BUFFER REGISTER
3645 013274 005005      CLR      R5          ;SET "EXPECTED"
3646 013276 052777 000400 166106      BIS      #MRESET,@TXCSR ;RESET THE DEVICE
3647 013304 004737 005044      JSR      PC,SMALL    ;WAIT FOR RESET TO FINISH
3648 013310 032713 000040      BIT      #BITS,(R3)  ;TEST BIT 5
3649 013314 001402      BEQ      1$          ;BIT 5 IS CLEARED
3650 013316 011304      MOV      (R3),R4      ;LOAD "FOUND"
3651 013320 104003      HLT      3            ;BIT 5 IS SET AND SHOULDN'T BE
3652 013322 052713 000040 1$:      BIS      #BITS,(R3)  ;SET BIT 5 AGAIN
3653 013326 005013      CLR      (R3)        ;CLEAR THE TRANSMITTER BUFFER REGISTER
3654 013330 032713 000040      BIT      #BITS,(R3)  ;TEST TO SEE IF BIT 5 CLEARED
3655 013334 001402      BEQ      2$          ;BIT 5 IS OK
3656 013336 011304      MOV      (R3),R4      ;LOAD "FOUND"
3657 013340 104003      HLT      3            ;BIT 5 FAILED TO CLEAR
3658 013342 104400 2$:      SCOPE              ;SCOPE THIS TEST

```

```

:***** TEST 52 *****
:*TRANSMITTER BUFFER REGISTER READ/WRITE BIT 6 RESET AND CLEAR TEST
:*WRITE BIT 6, AND TEST THAT IT WILL BE CLEARED AFTER A
:*DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION
:*****

```

```

:*****
:TEST 52
:*****
:*****

```

```

3673 013344 012737 000052 001226 TST52: MOV      #52,@TSTNO

```


TRANSMITTER BUFFER REGISTER RESET AND CLEAR TEST BIT 6

```

3674 013352 012737 013440 001216 MOV #TST53,NEXT
3675 013360 013703 001414 MOV TXDBUF,R3 ;GET THE TRANSMITTER BUFFER REGISTER
3676 013364 052713 000100 BIS #BIT6,(R3) ;SET BIT 6 AT TRANSMITTER BUFFER REGISTER
3677 013370 005005 CLR R5 ;SET "EXPECTED"
3678 013372 052777 000400 166012 BIS #MRESET,ATXCSR ;RESET THE DEVICE
3679 013400 004737 005044 JSR PC,SMALL ;WAIT FOR RESET TO FINISH
3680 013404 032713 000100 BIT #BIT6,(R3) ;TEST BIT 6
3681 013410 001402 BEQ 1$ ;BIT 6 IS CLEARED
3682 013412 011304 MOV (R3),R4 ;LOAD "FOUND"
3683 013414 104003 HLT 3 ;BIT 6 IS SET AND SHOULDN'T BE
3684 013416 052713 000100 1$: BIS #BIT6,(R3) ;SET BIT 6 AGAIN
3685 013422 005013 CLR (R3) ;CLEAR THE TRANSMITTER BUFFER REGISTER
3686 013424 032713 000100 BIT #BIT6,(R3) ;TEST TO SEE IF BIT 6 CLEARED
3687 013430 001402 BEQ 2$ ;BIT 6 IS OK
3688 013432 011304 MOV (R3),R4 ;LOAD "FOUND"
3689 013434 104003 HLT 3 ;BIT 6 FAILED TO CLEAR
3690 013436 104400 2$: SCOPE ;SCOPE THIS TEST

```

```

:***** TEST 53 *****
:*TRANSMITTER BUFFER REGISTER READ/WRITE BIT 7 RESET AND CLEAR TEST
:*WRITE BIT 7, AND TEST THAT IT WILL BE CLEARED AFTER A
:*DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION
:*****

```

```

:*****
:TEST 53
:*****

```

```

3705 013440 012737 000053 001226 TST53: MOV #53,ATSTNO
3706 013446 012737 013534 001216 MOV #TST54,NEXT
3707 013454 013703 001414 MOV TXDBUF,R3 ;GET THE TRANSMITTER BUFFER REGISTER
3708 013460 052713 000200 BIS #BIT7,(R3) ;SET BIT 7 AT TRANSMITTER BUFFER REGISTER
3709 013464 005005 CLR R5 ;SET "EXPECTED"
3710 013466 052777 000400 165716 BIS #MRESET,ATXCSR ;RESET THE DEVICE
3711 013474 004737 005044 JSR PC,SMALL ;WAIT FOR RESET TO FINISH
3712 013500 032713 000200 BIT #BIT7,(R3) ;TEST BIT 7
3713 013504 001402 BEQ 1$ ;BIT 7 IS CLEARED
3714 013506 011304 MOV (R3),R4 ;LOAD "FOUND"
3715 013510 104003 HLT 3 ;BIT 7 IS SET AND SHOULDN'T BE
3716 013512 052713 000200 1$: BIS #BIT7,(R3) ;SET BIT 7 AGAIN
3717 013516 005013 CLR (R3) ;CLEAR THE TRANSMITTER BUFFER REGISTER
3718 013520 032713 000200 BIT #BIT7,(R3) ;TEST TO SEE IF BIT 7 CLEARED
3719 013524 001402 BEQ 2$ ;BIT 7 IS OK
3720 013526 011304 MOV (R3),R4 ;LOAD "FOUND"
3721 013530 104003 HLT 3 ;BIT 7 FAILED TO CLEAR
3722 013532 104400 2$: SCOPE ;SCOPE THIS TEST

```

```

:***** TEST 54 *****
:*TRANSMITTER BUFFER REGISTER READ/WRITE BIT 8 RESET AND CLEAR TEST
:*WRITE BIT 8, AND TEST THAT IT WILL BE CLEARED AFTER A
:*DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION
:*****

```

```

3723
3724
3725
3726
3727
3728
3729

```

3730
3731
3732
3733
3734
3735
3736
3737
3738
3739
3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759
3760
3761
3762
3763
3764
3765
3766
3767
3768
3769
3770
3771
3772
3773
3774
3775
3776
3777
3778
3779
3780
3781
3782
3783
3784
3785

013534 012737 000054 001226
013542 012737 013630 001216
013550 013703 001414
013554 052713 000400
013560 005005
013562 052777 000400 165622
013570 004737 005044
013574 032713 000400
013600 001402
013602 011304
013604 104003
013606 052713 000400
013612 005013
013614 032713 000400
013620 001402
013622 011304
013624 104003
013626 104400

012737 000055 001226
012737 013724 001216
013703 001414
052713 001000
013654 005005
013656 052777 000400 165526
013664 004737 005044
013670 032713 001000
013674 001402
013676 011304
013700 104003
013702 052713 001000
013706 005013
013710 032713 001000
013714 001402
013716 011304
013720 104003

```
*****  
*  
: TEST 54  
*  
*****  
*****  
TST54: MOV #54, @TSTNO  
MOV #TST55, NEXT  
MOV TXDBUF, R3 ; GET THE TRANSMITTER BUFFER REGISTER  
BIS #BIT8, (R3) ; SET BIT 8 AT TRANSMITTER BUFFER REGISTER  
CLR R5 ; SET "EXPECTED"  
BIS #MRESET, @TXCSR ; RESET THE DEVICE  
JSR PC, SMALL ; WAIT FOR RESET TO FINISH  
BIT #BIT8, (R3) ; TEST BIT 8  
BEQ 1$ ; BIT 8 IS CLEARED  
MOV (R3), R4 ; LOAD "FOUND"  
HLT 3 ; BIT 8 IS SET AND SHOULDN'T BE  
1$: BIS #BIT8, (R3) ; SET BIT 8 AGAIN  
CLR (R3) ; CLEAR THE TRANSMITTER BUFFER REGISTER  
BIT #BIT8, (R3) ; TEST TO SEE IF BIT 8 CLEARED  
BEQ 2$ ; BIT 8 IS OK  
MOV (R3), R4 ; LOAD "FOUND"  
HLT 3 ; BIT 8 FAILED TO CLEAR  
2$: SCOPE ; SCOPE THIS TEST
```

***** TEST 55 *****
*TRANSMITTER BUFFER REGISTER READ/WRITE BIT 9 RESET AND CLEAR TEST
*WRITE BIT 9, AND TEST THAT IT WILL BE CLEARED AFTER A
*DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION

```
*****  
*  
: TEST 55  
*  
*****  
*****  
TST55: MOV #55, @TSTNO  
MOV #TST56, NEXT  
MOV TXDBUF, R3 ; GET THE TRANSMITTER BUFFER REGISTER  
BIS #BIT9, (R3) ; SET BIT 9 AT TRANSMITTER BUFFER REGISTER  
CLR R5 ; SET "EXPECTED"  
BIS #MRESET, @TXCSR ; RESET THE DEVICE  
JSR PC, SMALL ; WAIT FOR RESET TO FINISH  
BIT #BIT9, (R3) ; TEST BIT 9  
BEQ 1$ ; BIT 9 IS CLEARED  
MOV (R3), R4 ; LOAD "FOUND"  
HLT 3 ; BIT 9 IS SET AND SHOULDN'T BE  
1$: BIS #BIT9, (R3) ; SET BIT 9 AGAIN  
CLR (R3) ; CLEAR THE TRANSMITTER BUFFER REGISTER  
BIT #BIT9, (R3) ; TEST TO SEE IF BIT 9 CLEARED  
BEQ 2$ ; BIT 9 IS OK  
MOV (R3), R4 ; LOAD "FOUND"  
HLT 3 ; BIT 9 FAILED TO CLEAR
```

3786 013722 104400

2\$: SCOPE ;SCOPE THIS TEST

3787
3788
3789
3790
3791
3792
3793
3794
3795
3796
3797
3798
3799

***** TEST 56 *****
;TRANSMITTER BUFFER REGISTER READ/WRITE BIT 10 RESET AND CLEAR TEST
;WRITE BIT 10, AND TEST THAT IT WILL BE CLEARED AFTER A
;DEVICE RESET AND TRANSMITTER BUFFER REGISTER CLR INSTRUCTION

;TEST 56

3800
3801 013724 012737 000056 001226
3802 013732 012737 014020 001216
3803 013740 013703 001414
3804 013744 052713 002000
3805 013750 005005
3806 013752 052777 000400 165432
3807 013760 004737 005044
3808 013764 032713 002000
3809 013770 001402
3810 013772 011304
3811 013774 104003
3812 013776 052713 002000
3813 014002 005013
3814 014004 032713 002000
3815 014010 001402
3816 014012 011304
3817 014014 104003
3818 014016 104400

;ST56: MOV #56, @TSTNO
MOV #TST57, NEXT
MOV TXDBUF, R3 ;GET THE TRANSMITTER BUFFER REGISTER
BIS #BIT10, (R3) ;SET BIT 10 AT TRANSMITTER BUFFER REGISTER
CLR R5 ;SET "EXPECTED"
BIS #MRESET, @TXCSR ;RESET THE DEVICE
JSR PC, SMALL ;WAIT FOR RESET TO FINISH
BIT #BIT10, (R3) ;TEST BIT 10
BEQ 1\$;BIT 10 IS CLEARED
MOV (R3), R4 ;LOAD "FOUND"
HLT 3 ;BIT 10 IS SET AND SHOULDN'T BE
1\$: BIS #BIT10, (R3) ;SET BIT 10 AGAIN
CLR (R3) ;CLEAR THE TRANSMITTER BUFFER REGISTER
BIT #BIT10, (R3) ;TEST TO SEE IF BIT 10 CLEARED
BEQ 2\$;BIT 10 IS OK
MOV (R3), R4 ;LOAD "FOUND"
HLT 3 ;BIT 10 FAILED TO CLEAR
2\$: SCOPE ;SCOPE THIS TEST

3819
3820
3821
3822
3823
3824
3825
3826
3827
3828
3829
3830
3831
3832

***** TEST 57 *****
;RECEIVER BUFFER REGISTER TEST
;TEST THAT RECEIVER BUFFER REGISTER CANNOT BE WRITTEN.
;WRITE RECEIVER BUFFER REGISTER WITH ALL 1'S
;AND VERIFY THAT ALL 0'S ARE READ BACK.

;TEST 57

3833
3834 014020 012737 000057 001226
3835 014026 012737 014060 001216
3836 014034 013703 001406
3837
3838 014040 005005
3839 014042 012713 177777
3840 014046 011304
3841 014050 020504

;ST57: MOV #57, @TSTNO
MOV #TST60, NEXT
MOV RXDBUF, R3 ;GET THE REGISTER.
CLR R5 ;SET EXPECTED (ZERO)
MOV #-1, (R3) ;WRITE REGISTER WITH ALL 1'S
MOV (R3), R4 ;READ THE REGISTER.
CMP R5, R4 ;IS THE REGISTER EQUAL TO ZERO.

3842 014052 001401
3843 014054 104003
3844 014056 104400

1\$: BEQ 1\$;BR IF OK.
HLT 3 ;REGISTER NOT ZERO.
SCOPE ;SCOPE THIS TEST.

***** TEST 60 *****

*
: *PARAMETER STATUS REGISTER TEST
: *TEST THAT PARAMETER STATUS REGISTER CANNOT BE WRITTEN
: *READ THE PARAMETER STATUS REGISTER AND STORE THE DATA;
: *COMPLEMENT THE DATA AND WRITE THE PARAMETER STATUS REGISTER
: *VERIFING THAT THE PARAMETER STATUS REGISTER DID NOT CHANGE.
: *****

*
: TEST 60
*
: *****

3861
3862 014060 012737 000060 001226
3863 014066 012737 014122 001216
3864 014074 013703 001410
3865
3866 014100 011305
3867 014102 010504
3868 014104 005104
3869 014106 010413
3870 014110 011304
3871 014112 020504
3872 014114 001401
3873 014116 104003
3874 014120 104400

*
: *****
: *****
TST60: MOV #60, #TSTNO
MOV #TST61, NEXT
MOV PARCSR, R3
MOV (R3), R5 ;GET THE REGISTER.
MOV R5, R4 ;READ THE REGISTER INTO R5
COM R4 ;SAVE REG INTO R4
MOV R4, (R3) ;MAKE R4 OPPOSITE TO REGISTER
MOV (R3), R4 ;WRITE THE REGISTER WITH THE COMPLIMENT
CMP R5, R4 ;READ THE REGISTER.
BEQ 1\$;IS THE REGISTER EQUAL TO ZERO.
HLT 3 ;BR IF OK.
1\$: SCOPE ;REGISTER NOT ZERO.
;SCOPE THIS TEST.

***** TEST 61 *****
*RECEIVER CONTROL REGISTER BIT 0 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 0 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

*
: TEST 61
*
: *****

3888
3889 014122 012737 000061 001226
3890 014130 012737 014234 001216
3891 014136 013703 001404
3892 014142 012705 000001
3893 014146 010513
3894 014150 011304
3895 014152 005005
3896 014154 020504
3897 014156 001401

*
: *****
: *****
TST61: MOV #61, #TSTNO
MOV #TST62, NEXT
MOV RXCSR, R3 ;GET THE RECEIVER CONTROL REGISTER
MOV #BIT0, R5 ;GET BIT 0
MOV R5, (R3) ;WRITE BIT 0 TO RECEIVER CONTROL REGISTER
MOV (R3), R4 ;READ RECEIVER CONTROL REGISTER BACK
CLR R5 ;SET "EXPECTED"
CMP R5, R4 ;R5=GOOD, R4= ?
BEQ 5\$;BIT 0 IS OK

RECEIVER CONTROL REGISTER READ ONLY BIT 0 TEST

```

3898 014160 104003          HLT      3          ;BIT FAILED TO CLR
3899 014162 012705 000001 5$:  MOV     #BIT0,R5 ;RELOAD THE BIT
3900 014166 010513          MOV     R5,(R3)   ;WRITE BIT 0 TO THE REG
3901 014170 052777 000400 165214 BIS     #MRESET,@TXCSR ;RESET THE DEVICE
3902 014176 004737 005044 JSR     PC,SMALL  ;WAIT FOR RESET TO FINISH
3903 014202 011304          MOV     (R3),R4   ;GET BIT 0
3904 014204 032704 000001 BIT     #BIT0,R4   ;TEST BIT 0 FOR RESET CLR
3905 014210 001401          BEQ     7$        ;BIT 0 IS OK
3906 014212 104003          HLT     3          ;BIT FAILED TO CLEAR
3907 014214 010513          7$:  MOV     R5,(R3) ;SET BIT 0
3908 014216 005013          CLR     (R3)      ;CLR RECEIVER CONTROL REGISTER
3909 014220 011304          MOV     (R3),R4   ;READ THE RECEIVER CONTROL REGISTER
3910 014222 005005          CLR     R5        ;SET "EXPECTED"
3911 014224 020504          CMP     R5,R4     ;R5=GOOD,R4=?
3912 014226 001401          BEQ     10$       ;BIT 0 IS OK
3913 014230 104003          HLT     3          ;BIT FAILED TO CLEAR
3914 014232 104400          10$: SCOPE       ;SCOPE THIS TEST
3915
3916
3917
3918
3919
3920
3921
3922
3923
3924
3925
3926
3927
3928

```

```

:***** TEST 62 *****
:*RECEIVER CONTROL REGISTER BIT 7 READ ONLY DEVICE RESET AND CLEAR TEST
:*WRITE BIT 7 A ONE AND VERIFY A ZERO IS READ BACK
:*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.
:*****

```

```

:*****
:TEST 62
:*****
:*****

```

```

3929 014234 012737 000062 001226 TST62: MOV     #62,@TSTNO
3930 014242 012737 014346 001216 MOV     #TST63,NEXT
3931 014250 013703 001404 MOV     RXCSR,R3   ;GET THE RECEIVER CONTROL REGISTER
3932 014254 012705 000200 MOV     #BIT7,R5   ;GET BIT 7
3933 014260 010513          MOV     R5,(R3)   ;WRITE BIT 7 TO RECEIVER CONTROL REGISTER
3934 014262 011304          MOV     (R3),R4   ;READ RECEIVER CONTROL REGISTER BACK
3935 014264 005005          CLR     R5        ;SET "EXPECTED"
3936 014266 020504          CMP     R5,R4     ;R5=GOOD,R4=?
3937 014270 001401          BEQ     5$        ;BIT 7 IS OK
3938 014272 104003          HLT     3          ;BIT FAILED TO CLR
3939 014274 012705 000200 5$:  MOV     #BIT7,R5 ;RELOAD THE BIT
3940 014300 010513          MOV     R5,(R3)   ;WRITE BIT 7 TO THE REG
3941 014302 052777 000400 165102 BIS     #MRESET,@TXCSR ;RESET THE DEVICE
3942 014310 004737 005044 JSR     PC,SMALL  ;WAIT FOR RESET TO FINISH
3943 014314 011304          MOV     (R3),R4   ;GET BIT 7
3944 014316 032704 000200 BIT     #BIT7,R4   ;TEST BIT 7 FOR RESET CLR
3945 014322 001401          BEQ     7$        ;BIT 7 IS OK
3946 014324 104003          HLT     3          ;BIT FAILED TO CLEAR
3947 014326 010513          7$:  MOV     R5,(R3) ;SET BIT 7
3948 014330 005013          CLR     (R3)      ;CLR RECEIVER CONTROL REGISTER
3949 014332 011304          MOV     (R3),R4   ;READ THE RECEIVER CONTROL REGISTER
3950 014334 005005          CLR     R5        ;SET "EXPECTED"
3951 014336 020504          CMP     R5,R4     ;R5=GOOD,R4=?
3952 014340 001401          BEQ     10$       ;BIT 7 IS OK
3953 014342 104003          HLT     3          ;BIT FAILED TO CLEAR

```

RECEIVER CONTROL REGISTER READ ONLY BIT 7 TEST

3954 014344 104400 10\$: SCOPE ;SCOPE THIS TEST

***** TEST 63 *****
*RECEIVER CONTROL REGISTER BIT 9 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 9 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

*
: TEST 63
*

3969	014346	012737	000063	001226	TST63:	MOV	#63, R5	TSTNO	
3970	014354	012737	014460	001216		MOV	#TST64, R5	NEXT	
3971	014362	013703	001404			MOV	RXCSR, R3		;GET THE RECEIVER CONTROL REGISTER
3972	014366	012705	001000			MOV	#BIT9, R5		;GET BIT 9
3973	014372	010513				MOV	R5, (R3)		;WRITE BIT 9 TO RECEIVER CONTROL REGISTER
3974	014374	011304				MOV	(R3), R4		;READ RECEIVER CONTROL REGISTER BACK
3975	014376	005005				CLR	R5		;SET "EXPECTED"
3976	014400	020504				CMP	R5, R4		;R5=GOOD, R4= ?
3977	014402	001401				BEQ	5\$;BIT 9 IS OK
3978	014404	104003				HLT	3		;BIT FAILED TO CLR
3979	014406	012705	001000		5\$:	MOV	#BIT9, R5		;RELOAD THE BIT
3980	014412	010513				MOV	R5, (R3)		;WRITE BIT 9 TO THE REG
3981	014414	052777	000400	164770		BIS	#MRESET, R3	TXCSR	;RESET THE DEVICE
3982	014422	004737	005044			JSR	PC, SMALL		;WAIT FOR RESET TO FINISH
3983	014426	011304				MOV	(R3), R4		;GET BIT 9
3984	014430	032704	001000			BIT	#BIT9, R4		;TEST BIT 9 FOR RESET CLR
3985	014434	001401				BEQ	7\$;BIT 9 IS OK
3986	014436	104003				HLT	3		;BIT FAILED TO CLEAR
3987	014440	010513			7\$:	MOV	R5, (R3)		;SET BIT 9
3988	014442	005013				CLR	(R3)		;CLR RECEIVER CONTROL REGISTER
3989	014444	011304				MOV	(R3), R4		;READ THE RECEIVER CONTROL REGISTER
3990	014446	005005				CLR	R5		;SET "EXPECTED"
3991	014450	020504				CMP	R5, R4		;R5=GOOD, R4= ?
3992	014452	001401				BEQ	10\$;BIT 9 IS OK
3993	014454	104003				HLT	3		;BIT FAILED TO CLEAR
3994	014456	104400			10\$:	SCOPE			;SCOPE THIS TEST

***** TEST 64 *****
*RECEIVER CONTROL REGISTER BIT 10 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 10 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

*
: TEST 64
*

4009 014460 012737 000064 001226 TST64: MOV #64, R5 TSTNO

RECEIVER CONTROL REGISTER READ ONLY BIT 10 TEST

```

4010 014466 012737 014572 001216 MOV #TST65,NEXT
4011 014474 013703 001404 MOV RXCSR,R3 ;GET THE RECEIVER CONTROL REGISTER
4012 014500 012705 002000 MOV #BIT10,R5 ;GET BIT 10
4013 014504 010513 MOV R5,(R3) ;WRITE BIT 10 TO RECEIVER CONTROL REGISTER
4014 014506 011304 MOV (R3),R4 ;READ RECEIVER CONTROL REGISTER BACK
4015 014510 005005 CLR R5 ;SET "EXPECTED"
4016 014512 020504 CMP R5,R4 ;R5=GOOD,R4=?
4017 014514 001401 BEQ 5$ ;BIT 10 IS OK
4018 014516 104003 HLT 3 ;BIT FAILED TO CLR
4019 014520 012705 002000 5$: MOV #BIT10,R5 ;RELOAD THE BIT
4020 014524 010513 MOV R5,(R3) ;WRITE BIT 10 TO THE REG
4021 014526 052777 000400 164656 BIS #MRESET,@TXCSR ;RESET THE DEVICE
4022 014534 004737 005044 JSR PC,SMALL ;WAIT FOR RESET TO FINISH
4023 014540 011304 MOV (R3),R4 ;GET BIT 10
4024 014542 032704 002000 BIT #BIT10,R4 ;TEST BIT 10 FOR RESET CLR
4025 014546 001401 BEQ 7$ ;BIT 10 IS OK
4026 014550 104003 HLT 3 ;BIT FAILED TO CLEAR
4027 014552 010513 7$: MOV R5,(R3) ;SET BIT 10
4028 014554 005013 CLR (R3) ;CLR RECEIVER CONTROL REGISTER
4029 014556 011304 MOV (R3),R4 ;READ THE RECEIVER CONTROL REGISTER
4030 014560 005005 CLR R5 ;SET "EXPECTED"
4031 014562 020504 CMP R5,R4 ;R5=GOOD,R4=?
4032 014564 001401 BEQ 10$ ;BIT 10 IS OK
4033 014566 104003 HLT 3 ;BIT FAILED TO CLEAR
4034 014570 104400 10$: SCOPE ;SCOPE THIS TEST

```

```

:***** TEST 65 *****
:*RECEIVER CONTROL REGISTER BIT 11 READ ONLY DEVICE RESET AND CLEAR TEST
:*WRITE BIT 11 A ONE AND VERIFY A ZERO IS READ BACK
:*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.
:*****

```

```

:*****
:TEST 65
:*****

```

```

4049 014572 012737 000065 001226 TST65: MOV #65,@TSTNO
4050 014600 012737 014704 001216 MOV #TST66,NEXT
4051 014606 013703 001404 MOV RXCSR,R3 ;GET THE RECEIVER CONTROL REGISTER
4052 014612 012705 004000 MOV #BIT11,R5 ;GET BIT 11
4053 014616 010513 MOV R5,(R3) ;WRITE BIT 11 TO RECEIVER CONTROL REGISTER
4054 014620 011304 MOV (R3),R4 ;READ RECEIVER CONTROL REGISTER BACK
4055 014622 005005 CLR R5 ;SET "EXPECTED"
4056 014624 020504 CMP R5,R4 ;R5=GOOD,R4=?
4057 014626 001401 BEQ 5$ ;BIT 11 IS OK
4058 014630 104003 HLT 3 ;BIT FAILED TO CLR
4059 014632 012705 004000 5$: MOV #BIT11,R5 ;RELOAD THE BIT
4060 014636 010513 MOV R5,(R3) ;WRITE BIT 11 TO THE REG
4061 014640 052777 000400 164544 BIS #MRESET,@TXCSR ;RESET THE DEVICE
4062 014646 004737 005044 JSR PC,SMALL ;WAIT FOR RESET TO FINISH
4063 014652 011304 MOV (R3),R4 ;GET BIT 11
4064 014654 032704 004000 BIT #BIT11,R4 ;TEST BIT 11 FOR RESET CLR
4065 014660 001401 BEQ 7$ ;BIT 11 IS OK

```

RECEIVER CONTROL REGISTER READ ONLY BIT 11 TEST

4066 014662 104003
4067 014664 010513
4068 014666 005013
4069 014670 011304
4070 014672 005005
4071 014674 020504
4072 014676 001401
4073 014700 104003
4074 014702 104400
4075
4076
4077
4078
4079
4080
4081
4082
4083
4084
4085
4086
4087
4088

7\$: HLT 3 ;BIT FAILED TO CLEAR
MOV R5,(R3) ;SET BIT 11
CLR (R3) ;CLR RECEIVER CONTROL REGISTER
MOV (R3),R4 ;READ THE RECEIVER CONTROL REGISTER
CLR R5 ;SET "EXPECTED"
CMP R5,R4 ;R5=GOOD,R4=?
BEQ 10\$;BIT 11 IS OK
HLT 3 ;BIT FAILED TO CLEAR
10\$: SCOPE ;SCOPE THIS TEST

***** TEST 66 *****
*RECEIVER CONTROL REGISTER BIT 12 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 12 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

*
TEST 66
*

4089 014704 012737 000066 001226
4090 014712 012737 015016 001216
4091 014720 013703 001404
4092 014724 012705 010000
4093 014730 010513
4094 014732 011304
4095 014734 005005
4096 014736 020504
4097 014740 001401
4098 014742 104003
4099 014744 012705 010000
4100 014750 010513
4101 014752 052777 000400 164432
4102 014760 004737 005044
4103 014764 011304
4104 014766 032704 010000
4105 014772 001401
4106 014774 104003
4107 014776 010513
4108 015000 005013
4109 015002 011304
4110 015004 005005
4111 015006 020504
4112 015010 001401
4113 015012 104003
4114 015014 104400
4115
4116
4117
4118
4119
4120
4121

TST66: MOV #66,@TSTNO ;GET THE RECEIVER CONTROL REGISTER
MOV #TST67,NEXT ;GET BIT 12
MOV RXCSR,R3 ;WRITE BIT 12 TO RECEIVER CONTROL REGISTER
MOV #BIT12,R5 ;READ RECEIVER CONTROL REGISTER BACK
MOV (R3),R4 ;SET "EXPECTED"
CLR R5 ;R5=GOOD,R4=?
CMP R5,R4 ;BIT 12 IS OK
BEQ 5\$;BIT FAILED TO CLR
HLT 3 ;RELOAD THE BIT
5\$: MOV #BIT12,R5 ;WRITE BIT 12 TO THE REG
MOV R5,(R3) ;RESET THE DEVICE
BIS #MRESET,@TXCSR ;WAIT FOR RESET TO FINISH
JSR PC,SMALL ;GET BIT 12
MOV (R3),R4 ;TEST BIT 12 FOR RESET CLR
BIT #BIT12,R4 ;BIT 12 IS OK
BEQ 7\$;BIT FAILED TO CLEAR
HLT 3 ;SET BIT 12
7\$: MOV R5,(R3) ;CLR RECEIVER CONTROL REGISTER
CLR (R3) ;READ THE RECEIVER CONTROL REGISTER
MOV (R3),R4 ;SET "EXPECTED"
CLR R5 ;R5=GOOD,R4=?
CMP R5,R4 ;BIT 12 IS OK
BEQ 10\$;BIT FAILED TO CLEAR
HLT 3 ;SCOPE THIS TEST
10\$: SCOPE

***** TEST 67 *****
*RECEIVER CONTROL REGISTER BIT 13 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 13 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

4122
4123
4124
4125
4126
4127
4128
4129
4130
4131
4132
4133
4134
4135
4136
4137
4138
4139
4140
4141
4142
4143
4144
4145
4146
4147
4148
4149
4150
4151
4152
4153
4154
4155
4156
4157
4158
4159
4160
4161
4162
4163
4164
4165
4166
4167
4168
4169
4170
4171
4172
4173
4174
4175
4176
4177

015016 012737 000067 001226
015024 012737 015130 001216
015032 013703 001404
015036 012705 020000
015042 010513
015044 011304
015046 005005
015050 020504
015052 001401
015054 104003
015056 012705 020000
015062 010513
015064 052777 000400 164320
015072 004737 005044
015076 011304
015100 032704 020000
015104 001401
015106 104003
015110 010513
015112 005013
015114 011304
015116 005005
015120 020504
015122 001401
015124 104003
015126 104400

```

:*****
:
: TEST 67
:
:*****
:*****
†ST67:  MOV    #67, @#TSTNO
        MOV    #TST70, NEXT
        MOV    RXCSR, R3          ; GET THE RECEIVER CONTROL REGISTER
        MCV    #BIT13, R5        ; GET BIT 13
        MOV    R5, (R3)          ; WRITE BIT 13 TO RECEIVER CONTROL REGISTER
        MOV    (R3), R4          ; READ RECEIVER CONTROL REGISTER BACK
        CLR    R5                ; SET "EXPECTED"
        CMP    R5, R4            ; R5=GOOD, R4= ?
        BEQ    5$                ; BIT 13 IS OK
        HLT    3                  ; BIT FAILED TO CLR
5$:     MCV    #BIT13, R5        ; RELOAD THE BIT
        MOV    R5, (R3)          ; WRITE BIT 13 TO THE REG
        BIS    #MRESET, @TXCSR  ; RESET THE DEVICE
        JSR    PC, SMALL         ; WAIT FOR RESET TO FINISH
        MOV    (R3), R4          ; GET BIT 13
        BIT    #BIT13, R4        ; TEST BIT 13 FOR RESET CLR
        BEQ    7$                ; BIT 13 IS OK
        HLT    3                  ; BIT FAILED TO CLEAR
7$:     MOV    R5, (R3)          ; SET BIT 13
        CLR    (R3)              ; CLR RECEIVER CONTROL REGISTER
        MOV    (R3), R4          ; READ THE RECEIVER CONTROL REGISTER
        CLR    R5                ; SET "EXPECTED"
        CMP    R5, R4            ; R5=GOOD, R4= ?
        BEQ    10$               ; BIT 13 IS OK
        HLT    3                  ; BIT FAILED TO CLEAR
10$:    SCOPE                     ; SCOPE THIS TEST

:***** TEST 70 *****
: *RECEIVER CONTROL REGISTER BIT 14 READ ONLY DEVICE RESET AND CLEAR TEST
: *WRITE BIT 14 A ONE AND VERIFY A ZERO IS READ BACK
: *REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.
:*****
:*****
:
: TEST 70
:
:*****
:*****
†ST70:  MOV    #70, @#TSTNO
        MOV    #TST71, NEXT
        MOV    RXCSR, R3          ; GET THE RECEIVER CONTROL REGISTER
        MCV    #BIT14, R5        ; GET BIT 14
        MOV    R5, (R3)          ; WRITE BIT 14 TO RECEIVER CONTROL REGISTER
        MOV    (R3), R4          ; READ RECEIVER CONTROL REGISTER BACK
        CLR    R5                ; SET "EXPECTED"
        CMP    R5, R4            ; R5=GOOD, R4= ?
        BEQ    5$                ; BIT 14 IS OK

```

RECEIVER CONTROL REGISTER READ ONLY BIT 14 TEST

```

4178 015166 104003          HLT      3          ;BIT FAILED TO CLR
4179 015170 012705 040000 5$:  MOV      #BIT14,R5 ;RELOAD THE BIT
4180 015174 010513          MOV      R5,(R3)  ;WRITE BIT 14 TO THE REG
4181 015176 052777 000400 164206 BIS      #MRESET,@TXCSR ;RESET THE DEVICE
4182 015204 004737 005044 JSR      PC,SMALL  ;WAIT FOR RESET TO FINISH
4183 015210 011304          MOV      (R3),R4  ;GET BIT 14
4184 015212 032704 040000 BIT      #BIT14,R4 ;TEST BIT 14 FOR RESET CLR
4185 015216 001401          BEQ      7$       ;BIT 14 IS OK
4186 015220 104003          HLT      3          ;BIT FAILED TO CLEAR
4187 015222 010513          MOV      R5,(R3)  ;SET BIT 14
4188 015224 005013          CLR      (R3)     ;CLR RECEIVER CONTROL REGISTER
4189 015226 011304          MOV      (R3),R4  ;READ THE RECEIVER CONTROL REGISTER
4190 015230 005005          CLR      R5       ;SET "EXPECTED"
4191 015232 020504          CMP      R5,R4    ;R5=GOOD,R4=?
4192 015234 001401          BEQ      10$      ;BIT 14 IS OK
4193 015236 104003          HLT      3          ;BIT FAILED TO CLEAR
4194 015240 104400 10$:  SCOPE    ;SCOPE THIS TEST

```

```

:***** TEST 71 *****
:*RECEIVER CONTROL REGISTER BIT 15 READ ONLY DEVICE RESET AND CLEAR TEST
:*WRITE BIT 15 A ONE AND VERIFY A ZERO IS READ BACK
:*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.
:*****

```

```

:*****
:TEST 71
:*****

```

```

4209 015242 012737 000071 001226 TST71: MOV      #71,@TSTNO
4210 015250 012737 015354 001216 MOV      #TST72,NEXT
4211 015256 013703 001404          MOV      RXCSR,R3 ;GET THE RECEIVER CONTROL REGISTER
4212 015262 012705 100000          MOV      #BIT15,R5 ;GET BIT 15
4213 015266 010513          MOV      R5,(R3)  ;WRITE BIT 15 TO RECEIVER CONTROL REGISTER
4214 015270 011304          MOV      (R3),R4  ;READ RECEIVER CONTROL REGISTER BACK
4215 015272 005005          CLR      R5       ;SET "EXPECTED"
4216 015274 020504          CMP      R5,R4    ;R5=GOOD,R4=?
4217 015276 001401          BEQ      5$       ;BIT 15 IS OK
4218 015300 104003          HLT      3          ;BIT FAILED TO CLR
4219 015302 012705 100000 5$:  MOV      #BIT15,R5 ;RELOAD THE BIT
4220 015306 010513          MOV      R5,(R3)  ;WRITE BIT 15 TO THE REG
4221 015310 052777 000400 164074 BIS      #MRESET,@TXCSR ;RESET THE DEVICE
4222 015316 004737 005044 JSR      PC,SMALL  ;WAIT FOR RESET TO FINISH
4223 015322 011304          MOV      (R3),R4  ;GET BIT 15
4224 015324 032704 100000 BIT      #BIT15,R4 ;TEST BIT 15 FOR RESET CLR
4225 015330 001401          BEQ      7$       ;BIT 15 IS OK
4226 015332 104003          HLT      3          ;BIT FAILED TO CLEAR
4227 015334 010513          MOV      R5,(R3)  ;SET BIT 15
4228 015336 005013          CLR      (R3)     ;CLR RECEIVER CONTROL REGISTER
4229 015340 011304          MOV      (R3),R4  ;READ THE RECEIVER CONTROL REGISTER
4230 015342 005005          CLR      R5       ;SET "EXPECTED"
4231 015344 020504          CMP      R5,R4    ;R5=GOOD,R4=?
4232 015346 001401          BEQ      10$      ;BIT 15 IS OK
4233 015350 104003          HLT      3          ;BIT FAILED TO CLEAR

```

4234 015352 104400

10\$: SCOPE ;SCOPE THIS TEST

4235
4236
4237
4238
4239
4240
4241
4242
4243
4244
4245
4246
4247
4248
4249
4250
4251
4252
4253
4254
4255
4256
4257
4258
4259
4260
4261
4262
4263
4264
4265
4266
4267
4268
4269
4270
4271
4272
4273
4274
4275
4276
4277
4278
4279
4280
4281
4282
4283
4284
4285
4286
4287
4288
4289

015354 012737 000072 001226
015362 012737 015466 001216
015370 013703 001406
015374 012705 000001
015400 010513
015402 011304
015404 005005
015406 020504
015410 001401
015412 104003
015414 012705 000001 5\$:
015420 010513
015422 052777 000400 163762
015430 004737 005044
015434 011304
015436 032704 000001
015442 001401
015444 104003
015446 010513 7\$:
015450 005013
015452 011304
015454 005005
015456 020504
015460 001401
015462 104003
015464 104400 10\$:

***** TEST 72 *****
*RECEIVER BUFFER REGISTER BIT 0 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 0 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

*
TEST 72
*

TST72: MOV #72, @TSTNO
MOV @TST73, NEXT
MOV RXDBUF, R3 ;GET THE RECEIVER BUFFER REGISTER
MOV #BIT0, R5 ;GET BIT 0
MOV R5, (R3) ;WRITE BIT 0 TO RECEIVER BUFFER REGISTER
MOV (R3), R4 ;READ RECEIVER BUFFER REGISTER BACK
CLR R5 ;SET "EXPECTED"
CMP R5, R4 ;R5=GOOD, R4= ?
BEQ 5\$;BIT 0 IS OK
HLT 3 ;BIT FAILED TO CLR
5\$: MOV #BIT0, R5 ;RELOAD THE BIT
MOV R5, (R3) ;WRITE BIT 0 TO THE REG
BIS #MRESET, @TXCSR ;RESET THE DEVICE
JSR PC, SMALL ;WAIT FOR RESET TO FINISH
MOV (R3), R4 ;GET BIT 0
BIT #BIT0, R4 ;TEST BIT 0 FOR RESET CLR
BEQ 7\$;BIT 0 IS OK
HLT 3 ;BIT FAILED TO CLEAR
7\$: MOV R5, (R3) ;SET BIT 0
CLR (R3) ;CLR RECEIVER BUFFER REGISTER
MOV (R3), R4 ;READ THE RECEIVER BUFFER REGISTER
CLR R5 ;SET "EXPECTED"
CMP R5, R4 ;R5=GOOD, R4= ?
BEQ 10\$;BIT 0 IS OK
HLT 3 ;BIT FAILED TO CLEAR
10\$: SCOPE ;SCOPE THIS TEST

***** TEST 73 *****
*RECEIVER BUFFER REGISTER BIT 1 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 1 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

*
TEST 73
*

015466 012737 000073 001226

TST73: MOV #73, @TSTNO

RECEIVER BUFFER REGISTER READ ONLY BIT 1 TEST

```

4290 015474 012737 015600 001216      MOV      #TST74,NEXT
4291 015502 013703 001406      MOV      RXDBUF,R3          ;GET THE RECEIVER BUFFER REGISTER
4292 015506 012705 000002      MOV      #BIT1,R5          ;GET BIT 1
4293 015512 010513      MOV      R5,(R3)           ;WRITE BIT 1 TO RECEIVER BUFFER REGISTER
4294 015514 011304      MOV      (R3),R4           ;READ RECEIVER BUFFER REGISTER BACK
4295 015516 005005      CLR      R5                ;SET "EXPECTED"
4296 015520 020504      CMP      R5,R4             ;R5=GOOD,R4=?
4297 015522 001401      BEQ      5$                ;BIT 1 IS OK
4298 015524 104003      HLT      3                  ;BIT FAILED TO CLR
4299 015526 012705 000002 5$:      MOV      #BIT1,R5          ;RELOAD THE BIT
4300 015532 010513      MOV      R5,(R3)           ;WRITE BIT 1 TO THE REG
4301 015534 052777 000400 163650  BIS      #MRESET,@TXCSR    ;RESET THE DEVICE
4302 015542 004737 005044      JSR      PC,SMALL          ;WAIT FOR RESET TO FINISH
4303 015546 011304      MOV      (R3),R4           ;GET BIT 1
4304 015550 032704 000002      BIT      #BIT1,R4          ;TEST BIT 1 FOR RESET CLR
4305 015554 001401      BEQ      7$                ;BIT 1 IS OK
4306 015556 104003      HLT      3                  ;BIT FAILED TO CLEAR
4307 015560 010513 7$:      MOV      R5,(R3)           ;SET BIT 1
4308 015562 005013      CLR      (R3)              ;CLR RECEIVER BUFFER REGISTER
4309 015564 011304      MOV      (R3),R4           ;READ THE RECEIVER BUFFER REGISTER
4310 015566 005005      CLR      R5                ;SET "EXPECTED"
4311 015570 020504      CMP      R5,R4             ;R5=GOOD,R4=?
4312 015572 001401      BEQ      10$               ;BIT 1 IS OK
4313 015574 104003      HLT      3                  ;BIT FAILED TO CLEAR
4314 015576 104400 10$:     SCOPE                    ;SCOPE THIS TEST

```

```

***** TEST 74 *****
*RECEIVER BUFFER REGISTER BIT 2 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 2 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.
*****

```

```

*****
: TEST 74
*****

```

```

4329 015600 012737 000074 001226  TST74:  MOV      #74,@TSTNO
4330 015606 012737 015712 001216      MOV      #TST75,NEXT
4331 015614 013703 001406      MOV      RXDBUF,R3          ;GET THE RECEIVER BUFFER REGISTER
4332 015620 012705 000004      MOV      #BIT2,R5          ;GET BIT 2
4333 015624 010513      MOV      R5,(R3)           ;WRITE BIT 2 TO RECEIVER BUFFER REGISTER
4334 015626 011304      MOV      (R3),R4           ;READ RECEIVER BUFFER REGISTER BACK
4335 015630 005005      CLR      R5                ;SET "EXPECTED"
4336 015632 020504      CMP      R5,R4             ;R5=GOOD,R4=?
4337 015634 001401      BEQ      5$                ;BIT 2 IS OK
4338 015636 104003      HLT      3                  ;BIT FAILED TO CLR
4339 015640 012705 000004 5$:      MOV      #BIT2,R5          ;RELOAD THE BIT
4340 015644 010513      MOV      R5,(R3)           ;WRITE BIT 2 TO THE REG
4341 015646 052777 000400 163536  BIS      #MRESET,@TXCSR    ;RESET THE DEVICE
4342 015654 004737 005044      JSR      PC,SMALL          ;WAIT FOR RESET TO FINISH
4343 015660 011304      MOV      (R3),R4           ;GET BIT 2
4344 015662 032704 000004      BIT      #BIT2,R4          ;TEST BIT 2 FOR RESET CLR
4345 015666 001401      BEQ      7$                ;BIT 2 IS OK

```

RECEIVER BUFFER REGISTER READ ONLY BIT 2 TEST

```

4346 015670 104003          HLT      3          ;BIT FAILED TO CLEAR
4347 015672 010513      7$:  MOV      R5,(R3)    ;SET BIT 2
4348 015674 005013          CLR      (R3)        ;CLR RECEIVER BUFFER REGISTER
4349 015676 011304          MOV      (R3),R4     ;READ THE RECEIVER BUFFER REGISTER
4350 015700 005005          CLR      R5          ;SET "EXPECTED"
4351 015702 020504          CMP      R5,R4       ;R5=GOOD,R4=?
4352 015704 001401          BEQ      10$         ;BIT 2 IS OK
4353 015706 104003          HLT      3          ;BIT FAILED TO CLEAR
4354 015710 104400      10$:  SCOPE          ;SCOPE THIS TEST

```

```

:***** TEST 75 *****
:*RECEIVER BUFFER REGISTER BIT 3 READ ONLY DEVICE RESET AND CLEAR TEST
:*WRITE BIT 3 A ONE AND VERIFY A ZERO IS READ BACK
:*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.
:*****

```

```

:*****
:TEST 75
:*****
:*****

```

```

4369 015712 012737 000075 001226 75$:  MOV      #75,#TSTNO
4370 015720 012737 016024 001216 75$:  MOV      #TST76,NEXT
4371 015726 013703 001406 75$:  MOV      RXDBUF,R3    ;GET THE RECEIVER BUFFER REGISTER
4372 015732 012705 000010 75$:  MOV      #BIT3,R5    ;GET BIT 3
4373 015736 010513 75$:  MOV      R5,(R3)     ;WRITE BIT 3 TO RECEIVER BUFFER REGISTER
4374 015740 011304 75$:  MOV      (R3),R4     ;READ RECEIVER BUFFER REGISTER BACK
4375 015742 005005 75$:  CLR      R5          ;SET "EXPECTED"
4376 015744 020504 75$:  CMP      R5,R4       ;R5=GOOD,R4=?
4377 015746 001401 75$:  BEQ      5$          ;BIT 3 IS OK
4378 015750 104003 75$:  HLT      3          ;BIT FAILED TO CLR
4379 015752 012705 000010 5$:  MOV      #BIT3,R5    ;RELOAD THE BIT
4380 015756 010513 5$:  MOV      R5,(R3)     ;WRITE BIT 3 TO THE REG
4381 015760 052777 000400 163424 5$:  BIS      #MRESET,@TXCSR ;RESET THE DEVICE
4382 015766 004737 005044 5$:  JSR      PC,SMALL    ;WAIT FOR RESET TO FINISH
4383 015772 011304 5$:  MOV      (R3),R4     ;GET BIT 3
4384 015774 032704 000010 5$:  BIT      #BIT3,R4    ;TEST BIT 3 FOR RESET CLR
4385 016000 001401 5$:  BEQ      7$          ;BIT 3 IS OK
4386 016002 104003 5$:  HLT      3          ;BIT FAILED TO CLEAR
4387 016004 010513 7$:  MOV      R5,(R3)     ;SET BIT 3
4388 016006 005013 7$:  CLR      (R3)        ;CLR RECEIVER BUFFER REGISTER
4389 016010 011304 7$:  MOV      (R3),R4     ;READ THE RECEIVER BUFFER REGISTER
4390 016012 005005 7$:  CLR      R5          ;SET "EXPECTED"
4391 016014 020504 7$:  CMP      R5,R4       ;R5=GOOD,R4=?
4392 016016 001401 7$:  BEQ      10$        ;BIT 3 IS OK
4393 016020 104003 7$:  HLT      3          ;BIT FAILED TO CLEAR
4394 016022 104400 10$:  SCOPE          ;SCOPE THIS TEST

```

```

:***** TEST 76 *****
:*RECEIVER BUFFER REGISTER BIT 4 READ ONLY DEVICE RESET AND CLEAR TEST
:*WRITE BIT 4 A ONE AND VERIFY A ZERO IS READ BACK
:*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.
:*****

```

```

4395
4396
4397
4398
4399
4400
4401

```

4402
4403
4404
4405
4406
4407
4408
4409
4410
4411
4412
4413
4414
4415
4416
4417
4418
4419
4420
4421
4422
4423
4424
4425
4426
4427
4428
4429
4430
4431
4432
4433
4434
4435
4436
4437
4438
4439
4440
4441
4442
4443
4444
4445
4446
4447
4448
4449
4450
4451
4452
4453
4454
4455
4456
4457

016024 012737 000076 001226
016032 012737 016136 001216
016040 013703 001406
016044 012705 000020
016050 010513
016052 011304
016054 005005
016056 020504
016060 001401
016062 104003
016064 012705 000020
016070 010513
016072 052777 000400 163312
016100 004737 005044
016104 011304
016106 032704 000020
016112 001401
016114 104003
016116 010513
016120 005013
016122 011304
016124 005005
016126 020504
016130 001401
016132 104003
016134 104400

```
*****  
*  
: TEST 76  
*  
*****  
*****  
TST76: MOV #76, TSTNO  
MOV #TST77, NEXT  
MOV RXDBUF, R3 ; GET THE RECEIVER BUFFER REGISTER  
MOV #BIT4, R5 ; GET BIT 4  
MOV R5, (R3) ; WRITE BIT 4 TO RECEIVER BUFFER REGISTER  
MOV (R3), R4 ; READ RECEIVER BUFFER REGISTER BACK  
CLR R5 ; SET "EXPECTED"  
CMP R5, R4 ; R5=GOOD, R4= ?  
BEQ 5$ ; BIT 4 IS OK  
HLT 3 ; BIT FAILED TO CLR  
5$: MOV #BIT4, R5 ; RELOAD THE BIT  
MOV R5, (R3) ; WRITE BIT 4 TO THE REG  
BIS #MRESET, TXCSR ; RESET THE DEVICE  
JSR PC, SMALL ; WAIT FOR RESET TO FINISH  
MOV (R3), R4 ; GET BIT 4  
BIT #BIT4, R4 ; TEST BIT 4 FOR RESET CLR  
BEQ 7$ ; BIT 4 IS OK  
HLT 3 ; BIT FAILED TO CLEAR  
7$: MOV R5, (R3) ; SET BIT 4  
CLR (R3) ; CLR RECEIVER BUFFER REGISTER  
MOV (R3), R4 ; READ THE RECEIVER BUFFER REGISTER  
CLR R5 ; SET "EXPECTED"  
CMP R5, R4 ; R5=GOOD, R4= ?  
BEQ 10$ ; BIT 4 IS OK  
HLT 3 ; BIT FAILED TO CLEAR  
10$: SCOPE ; SCOPE THIS TEST
```

***** TEST 77 *****
*RECEIVER BUFFER REGISTER BIT 5 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 5 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

```
*****  
*  
: TEST 77  
*  
*****  
*****  
TST77: MOV #77, TSTNO  
MOV #TST100, NEXT  
MOV RXDBUF, R3 ; GET THE RECEIVER BUFFER REGISTER  
MOV #BIT5, R5 ; GET BIT 5  
MOV R5, (R3) ; WRITE BIT 5 TO RECEIVER BUFFER REGISTER  
MOV (R3), R4 ; READ RECEIVER BUFFER REGISTER BACK  
CLR R5 ; SET "EXPECTED"  
CMP R5, R4 ; R5=GOOD, R4= ?  
BEQ 5$ ; BIT 5 IS OK
```

```

4458 016174 104003          HLT      3          ;BIT FAILED TO CLR
4459 016176 012705 000040 5$:      MOV      #BIT5,R5    ;RELOAD THE BIT
4460 016202 010513          MOV      R5,(R3)    ;WRITE BIT 5 TO THE REG
4461 016204 052777 000400 163200  BIS      #MRESET,@TXCSR ;RESET THE DEVICE
4462 016212 004737 005044      JSR      PC,SMALL   ;WAIT FOR RESET TO FINISH
4463 016216 011304          MOV      (R3),R4    ;GET BIT 5
4464 016220 032704 000040      BIT      #BIT5,R4   ;TEST BIT 5 FOR RESET CLR
4465 016224 001401          BEQ      7$        ;BIT 5 IS OK
4466 016226 104003          HLT      3          ;BIT FAILED TO CLEAR
4467 016230 010513          7$:      MOV      R5,(R3)    ;SET BIT 5
4468 016232 005013          CLR      (R3)       ;CLR RECEIVER BUFFER REGISTER
4469 016234 011304          MOV      (R3),R4    ;READ THE RECEIVER BUFFER REGISTER
4470 016236 005005          CLR      R5         ;SET "EXPECTED"
4471 016240 020504          CMP      R5,R4      ;R5=GOOD,R4=?
4472 016242 001401          BEQ      10$       ;BIT 5 IS OK
4473 016244 104003          HLT      3          ;BIT FAILED TO CLEAR
4474 016246 104400 10$:      SCOPE    ;SCOPE THIS TEST

```

```

:***** TEST 100 *****
:RECEIVER BUFFER REGISTER BIT 6 READ ONLY DEVICE RESET AND CLEAR TEST
:WRITE BIT 6 A ONE AND VERIFY A ZERO IS READ BACK
:REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.
:*****

```

```

:*****
:TEST 100
:*****
:*****

```

```

4489 016250 012737 000100 001226 †TST100: MOV      #100,@TSTNO
4490 016256 012737 016362 001216      MOV      #TST101,NEXT
4491 016264 013703 001406      MOV      RXDBUF,R3  ;GET THE RECEIVER BUFFER REGISTER
4492 016270 012705 000100      MOV      #BIT6,R5   ;GET BIT 6
4493 016274 010513          MOV      R5,(R3)    ;WRITE BIT 6 TO RECEIVER BUFFER REGISTER
4494 016276 011304          MOV      (R3),R4    ;READ RECEIVER BUFFER REGISTER BACK
4495 016300 005005          CLR      R5         ;SET "EXPECTED"
4496 016302 020504          CMP      R5,R4      ;R5=GOOD,R4=?
4497 016304 001401          BEQ      5$        ;BIT 6 IS OK
4498 016306 104003          HLT      3          ;BIT FAILED TO CLR
4499 016310 012705 000100 5$:      MOV      #BIT6,R5   ;RELOAD THE BIT
4500 016314 010513          MOV      R5,(R3)    ;WRITE BIT 6 TO THE REG
4501 016316 052777 000400 163066  BIS      #MRESET,@TXCSR ;RESET THE DEVICE
4502 016324 004737 005044      JSR      PC,SMALL   ;WAIT FOR RESET TO FINISH
4503 016330 011304          MOV      (R3),R4    ;GET BIT 6
4504 016332 032704 000100      BIT      #BIT6,R4   ;TEST BIT 6 FOR RESET CLR
4505 016336 001401          BEQ      7$        ;BIT 6 IS OK
4506 016340 104003          HLT      3          ;BIT FAILED TO CLEAR
4507 016342 010513          7$:      MOV      R5,(R3)    ;SET BIT 6
4508 016344 005013          CLR      (R3)       ;CLR RECEIVER BUFFER REGISTER
4509 016346 011304          MOV      (R3),R4    ;READ THE RECEIVER BUFFER REGISTER
4510 016350 005005          CLR      R5         ;SET "EXPECTED"
4511 016352 020504          CMP      R5,R4      ;R5=GOOD,R4=?
4512 016354 001401          BEQ      10$       ;BIT 6 IS OK
4513 016356 104003          HLT      3          ;BIT FAILED TO CLEAR

```

4514 016360 104400

10\$: SCOPE ;SCOPE THIS TEST

4515
4516
4517
4518
4519
4520
4521
4522
4523
4524
4525
4526
4527
4528
4529
4530
4531
4532
4533
4534
4535
4536
4537
4538
4539
4540
4541
4542
4543
4544
4545
4546
4547
4548
4549
4550
4551
4552
4553
4554
4555
4556
4557
4558
4559
4560
4561
4562
4563
4564
4565
4566
4567
4568
4569

016362 012737 000101 001226
016370 012737 016474 001216
016376 013703 001406
016402 012705 000200
016406 010513
016410 011304
016412 005005
016414 020504
016416 001401
016420 104003
016422 012705 000200
016426 010513
016430 052777 000400 162754
016436 004737 005044
016442 011304
016444 032704 000200
016450 001401
016452 104003
016454 010513
016456 005013
016460 011304
016462 005005
016464 020504
016466 001401
016470 104003
016472 104400

***** TEST 101 *****
*RECEIVER BUFFER REGISTER BIT 7 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 7 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

*
TEST 101
*

TST101: MOV #101, @TSTNO
MOV #TST102, NEXT
MOV RXDBUF, R3 ;GET THE RECEIVER BUFFER REGISTER
MOV #BIT7, R5 ;GET BIT 7
MOV R5, (R3) ;WRITE BIT 7 TO RECEIVER BUFFER REGISTER
MOV (R3), R4 ;READ RECEIVER BUFFER REGISTER BACK
CLR R5 ;SET "EXPECTED"
CMP R5, R4 ;R5=GOOD, R4= ?
BEQ 5\$;BIT 7 IS OK
HLT 3 ;BIT FAILED TO CLR
5\$: MOV #BIT7, R5 ;RELOAD THE BIT
MOV R5, (R3) ;WRITE BIT 7 TO THE REG
BIS #MRESET, @TXCSR ;RESET THE DEVICE
JSR PC, SMALL ;WAIT FOR RESET TO FINISH
MOV (R3), R4 ;GET BIT 7
BIT #BIT7, R4 ;TEST BIT 7 FOR RESET CLR
BEQ 7\$;BIT 7 IS OK
HLT 3 ;BIT FAILED TO CLEAR
7\$: MOV R5, (R3) ;SET BIT 7
CLR (R3) ;CLR RECEIVER BUFFER REGISTER
MOV (R3), R4 ;READ THE RECEIVER BUFFER REGISTER
CLR R5 ;SET "EXPECTED"
CMP R5, R4 ;R5=GOOD, R4= ?
BEQ 10\$;BIT 7 IS OK
HLT 3 ;BIT FAILED TO CLEAR
10\$: SCOPE ;SCOPE THIS TEST

***** TEST 102 *****
*RECEIVER BUFFER REGISTER BIT 8 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 8 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

*
TEST 102
*

TST102: MOV #102, @TSTNO

RECEIVER BUFFER REGISTER READ ONLY BIT 8 TEST

```

4570 016502 012737 016606 001216      MOV      #TST103,NEXT
4571 016510 013703 001406      MOV      RXDBUF,R3          ;GET THE RECEIVER BUFFER REGISTER
4572 016514 012705 000400      MOV      #BIT8,R5          ;GET BIT 8
4573 016520 010513      MOV      R5,(R3)           ;WRITE BIT 8 TO RECEIVER BUFFER REGISTER
4574 016522 011304      MOV      (R3),R4           ;READ RECEIVER BUFFER REGISTER BACK
4575 016524 005005      CLR      R5                ;SET "EXPECTED"
4576 016526 020504      CMP      R5,R4             ;R5=GOOD,R4=?
4577 016530 001401      BEQ      $$                ;BIT 8 IS OK
4578 016532 104003      HLT      3                 ;BIT FAILED TO CLR
4579 016534 012705 000400      5$:     MOV      #BIT8,R5          ;RELOAD THE BIT
4580 016540 010513      MOV      R5,(R3)           ;WRITE BIT 8 TO THE REG
4581 016542 052777 000400 162642      BIS      #MRESET,@TXCSR    ;RESET THE DEVICE
4582 016550 004737 005044      JSR      PC,SMALL         ;WAIT FOR RESET TO FINISH
4583 016554 011304      MOV      (R3),R4           ;GET BIT 8
4584 016556 032704 000400      BIT      #BIT8,R4          ;TEST BIT 8 FOR RESET CLR
4585 016562 001401      BEQ      7$                ;BIT 8 IS OK
4586 016564 104003      HLT      3                 ;BIT FAILED TO CLEAR
4587 016566 010513      7$:     MOV      R5,(R3)           ;SET BIT 8
4588 016570 005013      CLR      (R3)              ;CLR RECEIVER BUFFER REGISTER
4589 016572 011304      MOV      (R3),R4           ;READ THE RECEIVER BUFFER REGISTER
4590 016574 005005      CLR      R5                ;SET "EXPECTED"
4591 016576 020504      CMP      R5,R4             ;R5=GOOD,R4=?
4592 016600 001401      BEQ      10$               ;BIT 8 IS OK
4593 016602 104003      HLT      3                 ;BIT FAILED TO CLEAR
4594 016604 104400      10$:    SCOPE                ;SCOPE THIS TEST

```

```

;***** TEST 103 *****
;*RECEIVER BUFFER REGISTER BIT 9 READ ONLY DEVICE RESET AND CLEAR TEST
;*WRITE BIT 9 A ONE AND VERIFY A ZERO IS READ BACK
;*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.
;*****

```

```

;*****
; TEST 103
;*****

```

```

4609 016606 012737 000103 001226      TST103: MOV      #103,@TSTNO
4610 016614 012737 016720 001216      MOV      #TST104,NEXT
4611 016622 013703 001406      MOV      RXDBUF,R3          ;GET THE RECEIVER BUFFER REGISTER
4612 016626 012705 001000      MOV      #BIT9,R5          ;GET BIT 9
4613 016632 010513      MOV      R5,(R3)           ;WRITE BIT 9 TO RECEIVER BUFFER REGISTER
4614 016634 011304      MOV      (R3),R4           ;READ RECEIVER BUFFER REGISTER BACK
4615 016636 005005      CLR      R5                ;SET "EXPECTED"
4616 016640 020504      CMP      R5,R4             ;R5=GOOD,R4=?
4617 016642 001401      BEQ      $$                ;BIT 9 IS OK
4618 016644 104003      HLT      3                 ;BIT FAILED TO CLR
4619 016646 012705 001000      5$:     MOV      #BIT9,R5          ;RELOAD THE BIT
4620 016652 010513      MOV      R5,(R3)           ;WRITE BIT 9 TO THE REG
4621 016654 052777 000400 162530      BIS      #MRESET,@TXCSR    ;RESET THE DEVICE
4622 016662 004737 005044      JSR      PC,SMALL         ;WAIT FOR RESET TO FINISH
4623 016666 011304      MOV      (R3),R4           ;GET BIT 9
4624 016670 032704 001000      BIT      #BIT9,R4          ;TEST BIT 9 FOR RESET CLR
4625 016674 001401      BEQ      7$                ;BIT 9 IS OK

```

RECEIVER BUFFER REGISTER READ ONLY BIT 9 TEST

4626 016676 104003
4627 016700 010513
4628 016702 005013
4629 016704 011304
4630 016706 005005
4631 016710 020504
4632 016712 001401
4633 016714 104003
4634 016716 104400
4635
4636
4637
4638
4639
4640
4641
4642
4643
4644
4645
4646
4647
4648
4649 016720 012737 000104 001226
4650 016726 012737 017032 001216
4651 016734 013703 001406
4652 016740 012705 002000
4653 016744 010513
4654 016746 011304
4655 016750 005005
4656 016752 020504
4657 016754 001401
4658 016756 104003
4659 016760 012705 002000
4660 016764 010513
4661 016766 052777 000400 162416
4662 016774 004737 005044
4663 017000 011304
4664 017002 032704 002000
4665 017006 001401
4666 017010 104003
4667 017012 010513
4668 017014 005013
4669 017016 011304
4670 017020 005005
4671 017022 020504
4672 017024 001401
4673 017026 104003
4674 017030 104400
4675
4676
4677
4678
4679
4680
4681

7\$: HLT 3 ;BIT FAILED TO CLEAR
MOV R5,(R3) ;SET BIT 9
CLR (R3) ;CLR RECEIVER BUFFER REGISTER
MOV (R3),R4 ;READ THE RECEIVER BUFFER REGISTER
CLR R5 ;SET "EXPECTED"
CMP R5,R4 ;R5=GOOD,R4=?
BEQ 10\$;BIT 9 IS OK
HLT 3 ;BIT FAILED TO CLEAR
10\$: SCOPE ;SCOPE THIS TEST

***** TEST 104 *****
*RECEIVER BUFFER REGISTER BIT 10 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 10 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

*
TEST 104
*

TST104: MOV #104,#TSTNO
MOV #TST105,NEXT
MOV RXDBUF,R3 ;GET THE RECEIVER BUFFER REGISTER
MOV #BIT10,R5 ;GET BIT 10
MOV R5,(R3) ;WRITE BIT 10 TO RECEIVER BUFFER REGISTER
MOV (R3),R4 ;READ RECEIVER BUFFER REGISTER BACK
CLR R5 ;SET "EXPECTED"
CMP R5,R4 ;R5=GOOD,R4=?
BEQ 5\$;BIT 10 IS OK
HLT 3 ;BIT FAILED TO CLR
5\$: MOV #BIT10,R5 ;RELOAD THE BIT
MOV R5,(R3) ;WRITE BIT 10 TO THE REG
BIS #MRESET,#TXCSR ;RESET THE DEVICE
JSR PC,SMALL ;WAIT FOR RESET TO FINISH
MOV (R3),R4 ;GET BIT 10
BIT #BIT10,R4 ;TEST BIT 10 FOR RESET CLR
BEQ 7\$;BIT 10 IS OK
HLT 3 ;BIT FAILED TO CLEAR
7\$: MOV R5,(R3) ;SET BIT 10
CLR (R3) ;CLR RECEIVER BUFFER REGISTER
MOV (R3),R4 ;READ THE RECEIVER BUFFER REGISTER
CLR R5 ;SET "EXPECTED"
CMP R5,R4 ;R5=GOOD,R4=?
BEQ 10\$;BIT 10 IS OK
HLT 3 ;BIT FAILED TO CLEAR
10\$: SCOPE ;SCOPE THIS TEST

***** TEST 105 *****
*RECEIVER BUFFER REGISTER BIT 12 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 12 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

4682
4683
4684
4685
4686
4687
4688
4689
4690
4691
4692
4693
4694
4695
4696
4697
4698
4699
4700
4701
4702
4703
4704
4705
4706
4707
4708
4709
4710
4711
4712
4713
4714
4715
4716
4717
4718
4719
4720
4721
4722
4723
4724
4725
4726
4727
4728
4729
4730
4731
4732
4733
4734
4735
4736
4737

017032 012737 000105 001226
017040 012737 017144 001216
017046 013703 001406
017052 012705 010000
017056 010513
017060 011304
017062 005005
017064 020504
017066 001401
017070 104003
017072 012705 010000
017076 010513
017100 052777 000400 162304
017106 004737 005044
017112 011304
017114 032704 010000
017120 001401
017122 104003
017124 010513
017126 005013
017130 011304
017132 005005
017134 020504
017136 001401
017140 104003
017142 104400

```
*****  
*  
: TEST 105 *  
*****  
*****  
TST105: MOV #105, #TSTNO  
MOV #TST106, NEXT  
MOV RXDBUF, R3 ; GET THE RECEIVER BUFFER REGISTER  
MOV #BIT12, R5 ; GET BIT 12  
MOV R5, (R3) ; WRITE BIT 12 TO RECEIVER BUFFER REGISTER  
MOV (R3), R4 ; READ RECEIVER BUFFER REGISTER BACK  
CLR R5 ; SET "EXPECTED"  
CMP R5, R4 ; R5=GOOD, R4= ?  
BEQ 5$ ; BIT 12 IS OK  
HLT 3 ; BIT FAILED TO CLR  
5$: MOV #BIT12, R5 ; RELOAD THE BIT  
MOV R5, (R3) ; WRITE BIT 12 TO THE REG  
BIS #MRESET, TXCSR ; RESET THE DEVICE  
JSR PC, SMALL ; WAIT FOR RESET TO FINISH  
MOV (R3), R4 ; GET BIT 12  
BIT #BIT12, R4 ; TEST BIT 12 FOR RESET CLR  
BEQ 7$ ; BIT 12 IS OK  
HLT 3 ; BIT FAILED TO CLEAR  
7$: MOV R5, (R3) ; SET BIT 12  
CLR (R3) ; CLR RECEIVER BUFFER REGISTER  
MOV (R3), R4 ; READ THE RECEIVER BUFFER REGISTER  
CLR R5 ; SET "EXPECTED"  
CMP R5, R4 ; R5=GOOD, R4= ?  
BEQ 10$ ; BIT 12 IS OK  
HLT 3 ; BIT FAILED TO CLEAR  
10$: SCOPE ; SCOPE THIS TEST
```

***** TEST 106 *****
*RECEIVER BUFFER REGISTER BIT 14 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 14 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

```
*****  
*  
: TEST 106 *  
*****  
*****  
TST106: MOV #106, #TSTNO  
MOV #TST107, NEXT  
MOV RXDBUF, R3 ; GET THE RECEIVER BUFFER REGISTER  
MOV #BIT14, R5 ; GET BIT 14  
MOV R5, (R3) ; WRITE BIT 14 TO RECEIVER BUFFER REGISTER  
MOV (R3), R4 ; READ RECEIVER BUFFER REGISTER BACK  
CLR R5 ; SET "EXPECTED"  
CMP R5, R4 ; R5=GOOD, R4= ?  
BEQ 5$ ; BIT 14 IS OK
```

RECEIVER BUFFER REGISTER READ ONLY BIT 14 TEST

```

4738 017202 104003          HLT      3          ;BIT FAILED TO CLR
4739 017204 012705 040000 5$:  MOV     #BIT14,R5 ;RELOAD THE BIT
4740 017210 010513          MOV     R5,(R3)   ;WRITE BIT 14 TO THE REG
4741 017212 052777 000400 162172 BIS     #MRESET,@TXCSR ;RESET THE DEVICE
4742 017220 004737 005044 JSR     PC,SMALL  ;WAIT FOR RESET TO FINISH
4743 017224 011304          MOV     (R3),R4   ;GET BIT 14
4744 017226 032704 040000 BIT     #BIT14,R4 ;TEST BIT 14 FOR RESET CLR
4745 017232 001401          BEQ     7$        ;BIT 14 IS OK
4746 017234 104003          HLT      3          ;BIT FAILED TO CLEAR
4747 017236 010513          MOV     R5,(R3)   ;SET BIT 14
4748 017240 005013          CLR     (R3)      ;CLR RECEIVER BUFFER REGISTER
4749 017242 011304          MOV     (R3),R4   ;READ THE RECEIVER BUFFER REGISTER
4750 017244 005005          CLR     R5        ;SET "EXPECTED"
4751 017246 020504          CMP     R5,R4     ;R5=GOOD,R4=?
4752 017250 001401          BEQ     10$       ;BIT 14 IS OK
4753 017252 104003          HLT      3          ;BIT FAILED TO CLEAR
4754 017254 104400          SCOPE          ;SCOPE THIS TEST
4755
4756
4757

```

```

:***** TEST 107 *****
:*RECEIVER BUFFER REGISTER BIT 15 READ ONLY DEVICE RESET AND CLEAR TEST
:*WRITE BIT 15 A ONE AND VERIFY A ZERO IS READ BACK
:*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.
:*****

```

```

:*****
:TEST 107
:*****

```

```

4768
4769 017256 012737 000107 001226 TST107: MOV     #107,@TSTNO
4770 017264 012737 017370 001216 MOV     @TST10,NEXT
4771 017272 013703 001406 MOV     RXDBUF,R3 ;GET THE RECEIVER BUFFER REGISTER
4772 017276 012705 100000 MOV     #BIT15,R5 ;GET BIT 15
4773 017302 010513          MOV     R5,(R3)   ;WRITE BIT 15 TO RECEIVER BUFFER REGISTER
4774 017304 011304          MOV     (R3),R4   ;READ RECEIVER BUFFER REGISTER BACK
4775 017306 005005          CLR     R5        ;SET "EXPECTED"
4776 017310 020504          CMP     R5,R4     ;R5=GOOD,R4=?
4777 017312 001401          BEQ     5$        ;BIT 15 IS OK
4778 017314 104003          HLT      3          ;BIT FAILED TO CLR
4779 017316 012705 100000 5$:  MOV     #BIT15,R5 ;RELOAD THE BIT
4780 017322 010513          MOV     R5,(R3)   ;WRITE BIT 15 TO THE REG
4781 017324 052777 000400 162060 BIS     #MRESET,@TXCSR ;RESET THE DEVICE
4782 017332 004737 005044 JSR     PC,SMALL  ;WAIT FOR RESET TO FINISH
4783 017336 011304          MOV     (R3),R4   ;GET BIT 15
4784 017340 032704 100000 BIT     #BIT15,R4 ;TEST BIT 15 FOR RESET CLR
4785 017344 001401          BEQ     7$        ;BIT 15 IS OK
4786 017346 104003          HLT      3          ;BIT FAILED TO CLEAR
4787 017350 010513          MOV     R5,(R3)   ;SET BIT 15
4788 017352 005013          CLR     (R3)      ;CLR RECEIVER BUFFER REGISTER
4789 017354 011304          MOV     (R3),R4   ;READ THE RECEIVER BUFFER REGISTER
4790 017356 005005          CLR     R5        ;SET "EXPECTED"
4791 017360 020504          CMP     R5,R4     ;R5=GOOD,R4=?
4792 017362 001401          BEQ     10$       ;BIT 15 IS OK
4793 017364 104003          HLT      3          ;BIT FAILED TO CLEAR

```

4794 017366 104400

10\$: SCOPE ;SCOPE THIS TEST

***** TEST 110 *****
*THIS TEST VERIFIES BIT7 OF THE TRANSMITTER CONTROL REGISTER
*TEST THAT BIT 7 SETS AFTER A RESET AND MRESET
*VERIFY THAT WRITING THE LOW BYTE OF
*THE TRANSMITTER BUFFER REGISTER CLEARS BIT 7

4795
4796
4797
4798
4799
4800
4801
4802
4803
4804
4805
4806
4807
4808
4809
4810
4811
4812
4813
4814
4815
4816
4817
4818
4819
4820
4821
4822
4823
4824
4825
4826
4827
4828
4829
4830
4831
4832
4833
4834
4835
4836
4837
4838
4839
4840
4841
4842
4843
4844
4845
4846
4847
4848
4849

017370 012737 000110 001226
017376 012737 017476 001216
017404 013702 001414
017410 013703 001412
017414 012705 000200
017420 000005
017422 011304
017424 020504
017426 001401
017430 104003
017432 005005
017434 005012
017436 011304
017440 020504
017442 001401
017444 104003
017446 012705 000200
017452 052777 000400 161732
017460 004737 005044
017464 011304
017466 020504
017470 001401
017472 104003
017474 104400

*
: TEST 110
*

TST110: MOV #110, J#TSTNO
MOV #TST111, NEXT
MOV TXDBUF, R2 ;LOAD SECOND REG
MOV TXCSR, R3 ;GET THE TRANSMITTER CONTROL REGISTER
MOV #BIT7, R5 ;SET "EXPECTED"
RESET
MOV (R3), R4 ;GET THE BIT
CMP R5, R4 ;R5=GOOD, R4=?
BEQ 1\$;ARE THEY THE SAME?
HLT 3 ;NO--REPORT THE ERROR
1\$: CLR R5 ;SET "EXPECTED"
CLR (R2) ;WRITE THE LOW BYTE OF THE TXDBUF
MOV (R3), R4 ;READ BACK BIT 7
CMP R5, R4 ;R5=GOOD, R4=?
BEQ 2\$;ARE THEY THE SAME?
HLT 3 ;NO-BIT 7 FAILED TO CLEAR
2\$: MOV #BIT7, R5 ;SET "EXPECTED"
BIS #MRESET, TXCSR ;RESET THE DEVICE
JSR PC, SMALL ;WAIT FOR RESET TO FINISH
MOV (R3), R4 ;READ BACK BIT 7
CMP R5, R4 ;R5=GOOD, R4=?
BEQ 3\$;BRANCH IF BIT 7 OK
HLT 3 ;BIT 7 FAILED TO SET AFTER A MRESET
3\$: SCOPE ;SCOPE THIS TEST

***** TEST 111 *****
*TRANSMITTER CONTROL REGISTER BIT 9 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 9 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

*
: TEST 111
*

017476 012737 000111 001226
017504 012737 017620 001216

TST111: MOV #111, J#TSTNO
MOV #TST112, NEXT

TRANSMITTER CONTROL REGISTER READ ONLY TEST BIT 9

```

4850 017512 013703 001412      MOV      TXCSR,R3      ;GET THE TRANSMITTER CONTROL REGISTER
4851 017516 012705 001000      MOV      #BIT9,R5     ;GET BIT 9
4852 017522 010513              MOV      R5,(R3)      ;WRITE BIT 9 TO TRANSMITTER CONTROL REGISTER
4853 017524 011304              MOV      (R3),R4      ;READ TRANSMITTER CONTROL REGISTER BACK
4854 017526 005005              CLR      R5           ;SET "EXPECTED"
4855 017530 042704 000200      BIC      #BIT7,R4     ;CLEAR UNWANTED BITS
4856 017534 020504              CMP      R5,R4       ;R5=GOOD,R4=?
4857 017536 001401              BEQ      5$          ;BIT 9 IS OK
4858 017540 104003              HLT      3           ;BIT FAILED TO CLR
4859 017542 012705 001000      5$:      MOV      #BIT9,R5     ;RELOAD THE BIT
4860 017546 010513              MOV      R5,(R3)      ;WRITE BIT 9 TO THE REG
4861 017550 052777 000400 161634  BIS      #MRESET,@TXCSR ;RESET THE DEVICE
4862 017556 004737 005044      JSR      PC,SMALL     ;WAIT FOR RESET TO FINISH
4863 017562 011304              MOV      (R3),R4      ;GET BIT 9
4864 017564 032704 001000      BIT      #BIT9,R4     ;TEST BIT 9 FOR RESET CLR
4865 017570 001401              BEQ      7$          ;BIT 9 IS OK
4866 017572 104003              HLT      3           ;BIT FAILED TO CLEAR
4867 017574 010513              7$:      MOV      R5,(R3)     ;SET BIT 9
4868 017576 005013              CLR      (R3)         ;CLR TRANSMITTER CONTROL REGISTER
4869 017600 011304              MOV      (R3),R4      ;READ THE TRANSMITTER CONTROL REGISTER
4870 017602 005005              CLR      R5           ;SET "EXPECTED"
4871 017604 042704 000200      BIC      #BIT7,R4     ;CLEAR UNWANTED BITS
4872 017610 020504              CMP      R5,R4       ;R5=GOOD,R4=?
4873 017612 001401              BEQ      10$         ;BIT 9 IS OK
4874 017614 104003              HLT      3           ;BIT FAILED TO CLEAR
4875 017616 104400              10$:     SCOPE        ;SCOPE THIS TEST

```

```

;***** TEST 112 *****
;*TRANSMITTER CONTROL REGISTER BIT 14 READ ONLY DEVICE RESET AND CLEAR TEST
;*WRITE BIT 14 A ONE AND VERIFY A ZERO IS READ BACK
;*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.
;*****

```

```

;*****
; TEST 112
;*****
;*****

```

```

4890 017620 012737 000112 001226 TST112: MOV      #112,@TSTNO
4891 017626 012737 017742 001216      MOV      #TST113,NEXT
4892 017634 013703 001412      MOV      TXCSR,R3     ;GET THE TRANSMITTER CONTROL REGISTER
4893 017640 012705 040000      MOV      #BIT14,R5    ;GET BIT 14
4894 017644 010513              MOV      R5,(R3)      ;WRITE BIT 14 TO TRANSMITTER CONTROL REGISTER
4895 017646 011304              MOV      (R3),R4      ;READ TRANSMITTER CONTROL REGISTER BACK
4896 017650 005005              CLR      R5           ;SET "EXPECTED"
4897 017652 042704 000200      BIC      #BIT7,R4     ;CLEAR UNWANTED BITS
4898 017656 020504              CMP      R5,R4       ;R5=GOOD,R4=?
4899 017660 001401              BEQ      5$          ;BIT 14 IS OK
4900 017662 104003              HLT      3           ;BIT FAILED TO CLR
4901 017664 012705 040000      5$:      MOV      #BIT14,R5    ;RELOAD THE BIT
4902 017670 010513              MOV      R5,(R3)      ;WRITE BIT 14 TO THE REG
4903 017672 052777 000400 161512  BIS      #MRESET,@TXCSR ;RESET THE DEVICE
4904 017700 004737 005044      JSR      PC,SMALL     ;WAIT FOR RESET TO FINISH
4905 017704 011304              MOV      (R3),R4      ;GET BIT 14

```

TRANSMITTER CONTROL REGISTER READ ONLY TEST BIT 14

```

4906 017706 032704 040000 BIT #BIT14,R4 ;TEST BIT 14 FOR RESET CLR
4907 017712 001401 BEQ 7$ ;BIT 14 IS OK
4908 017714 104003 HLT 3 ;BIT FAILED TO CLEAR
4909 017716 010513 7$: MOV R5,(R3) ;SET BIT 14
4910 017720 005013 CLR (R3) ;CLR TRANSMITTER CONTROL REGISTER
4911 017722 011304 MOV (R3),R4 ;READ THE TRANSMITTER CONTROL REGISTER
4912 017724 005005 CLR R5 ;SET "EXPECTED"
4913 017726 042704 000200 BIC #BIT7,R4 ;CLEAR UNWANTED BITS
4914 017732 020504 CMP R5,R4 ;R5=GOOD,R4=?
4915 017734 001401 BEQ 10$ ;BIT 14 IS OK
4916 017736 104003 HLT 3 ;BIT FAILED TO CLEAR
4917 017740 104400 10$: SCOPE ;SCOPE THIS TEST

```

```

:***** TEST 113 *****
:*TRANSMITTER CONTROL REGISTER BIT 15 READ ONLY DEVICE RESET AND CLEAR TEST
:*WRITE BIT 15 A ONE AND VERIFY A ZERO IS READ BACK
:*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.
:*****

```

```

:*****
:TEST 113
:*****
:*****

```

```

4932 017742 012737 000113 001226 TST113: MOV #113,@TSTNO
4933 017750 012737 020064 001216 MOV #TST114,NEXT
4934 017756 013703 001412 MOV TXCSR,R3 ;GET THE TRANSMITTER CONTROL REGISTER
4935 017762 012705 100000 MOV #BIT15,R5 ;GET BIT 15
4936 017766 010513 MOV R5,(R3) ;WRITE BIT 15 TO TRANSMITTER CONTROL REGISTER
4937 017770 011304 MOV (R3),R4 ;READ TRANSMITTER CONTROL REGISTER BACK
4938 017772 005005 CLR R5 ;SET "EXPECTED"
4939 017774 042704 000200 BIC #BIT7,R4 ;CLEAR UNWANTED BITS
4940 020000 020504 CMP R5,R4 ;R5=GOOD,R4=?
4941 020002 001401 BEQ 5$ ;BIT 15 IS OK
4942 020004 104003 HLT 3 ;BIT FAILED TO CLR
4943 020006 012705 100000 5$: MOV #BIT15,R5 ;RELOAD THE BIT
4944 020012 010513 MOV R5,(R3) ;WRITE BIT 15 TO THE REG
4945 020014 052777 000400 161370 BIS #MRESET,@TXCSR ;RESET THE DEVICE
4946 020022 004737 005044 JSR PC,SMALL ;WAIT FOR RESET TO FINISH
4947 020026 011304 MOV (R3),R4 ;GET BIT 15
4948 020030 032704 100000 BIT #BIT15,R4 ;TEST BIT 15 FOR RESET CLR
4949 020034 001401 BEQ 7$ ;BIT 15 IS OK
4950 020036 104003 HLT 3 ;BIT FAILED TO CLEAR
4951 020040 010513 7$: MOV R5,(R3) ;SET BIT 15
4952 020042 005013 CLR (R3) ;CLR TRANSMITTER CONTROL REGISTER
4953 020044 011304 MOV (R3),R4 ;READ THE TRANSMITTER CONTROL REGISTER
4954 020046 005005 CLR R5 ;SET "EXPECTED"
4955 020050 042704 000200 BIC #BIT7,R4 ;CLEAR UNWANTED BITS
4956 020054 020504 CMP R5,R4 ;R5=GOOD,R4=?
4957 020056 001401 BEQ 10$ ;BIT 15 IS OK
4958 020060 104003 HLT 3 ;BIT FAILED TO CLEAR
4959 020062 104400 10$: SCOPE ;SCOPE THIS TEST
4960
4961

```

4962
4963
4964
4965
4966
4967
4968
4969
4970
4971
4972
4973
4974
4975
4976
4977
4978
4979
4980
4981
4982
4983
4984
4985
4986
4987
4988
4989
4990
4991
4992
4993
4994
4995
4996
4997
4998
4999
5000
5001
5002
5003
5004
5005
5006
5007
5008
5009
5010
5011
5012
5013
5014
5015
5016
5017

020064 012737 000114 001226
020072 012737 020176 001216
020100 013703 001414
020104 012705 010000
020110 010513
020112 011304
020114 005005
020116 020504
020120 001401
020122 104003
020124 012705 010000
020130 010513
020132 052777 000400 161252
020140 004737 005044
020144 011304
020146 032704 010000
020152 001401
020154 104003
020156 010513
020160 005013
020162 011304
020164 005005
020166 020504
020170 001401
020172 104003
020174 104400

***** TEST 114 *****
*TRANSMITTER BUFFER REGISTER BIT 12 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 12 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

*
TEST 114
*

```
TST114: MOV #114, @TSTNO
MOV #TST115, NEXT
MOV TXDBUF, R3 ;GET THE TRANSMITTER BUFFER REGISTER
MOV #BIT12, R5 ;GET BIT 12
MOV R5, (R3) ;WRITE BIT 12 TO TRANSMITTER BUFFER REGISTER
MOV (R3), R4 ;READ TRANSMITTER BUFFER REGISTER BACK
CLR R5 ;SET "EXPECTED"
CMP R5, R4 ;R5=GOOD, R4= ?
BEQ 5$ ;BIT 12 IS OK
HLT 3 ;BIT FAILED TO CLR
5$: MOV #BIT12, R5 ;RELOAD THE BIT
MOV R5, (R3) ;WRITE BIT 12 TO THE REG
BIS #MRESET, @TXCSR ;RESET THE DEVICE
JSR PC, SMALL ;WAIT FOR RESET TO FINISH
MOV (R3), R4 ;GET BIT 12
BIT #BIT12, R4 ;TEST BIT 12 FOR RESET CLR
BEQ 7$ ;BIT 12 IS OK
HLT 3 ;BIT FAILED TO CLEAR
7$: MOV R5, (R3) ;SET BIT 12
CLR (R3) ;CLR TRANSMITTER BUFFER REGISTER
MOV (R3), R4 ;READ THE TRANSMITTER BUFFER REGISTER
CLR R5 ;SET "EXPECTED"
CMP R5, R4 ;R5=GOOD, R4= ?
BEQ 10$ ;BIT 12 IS OK
HLT 3 ;BIT FAILED TO CLEAR
10$: SCOPE ;SCOPE THIS TEST
```

***** TEST 115 *****
*TRANSMITTER BUFFER REGISTER BIT 13 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 13 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.

*
TEST 115
*

```
TST115: MOV #115, @TSTNO
MOV #TST116, NEXT
MOV TXDBUF, R3 ;GET THE TRANSMITTER BUFFER REGISTER
MOV #BIT13, R5 ;GET BIT 13
```


TRANSMITTER BUFFER REGISTER READ ONLY TEST BIT 13

```

5018 020222 010513      MOV      R5, (R3)      ;WRITE BIT 13 TO TRANSMITTER BUFFER REGISTER
5019 020224 011304      MOV      (R3),R4      ;READ TRANSMITTER BUFFER REGISTER BACK
5020 020226 005005      CLR      R5          ;SET "EXPECTED"
5021 020230 020504      CMP      R5,R4       ;R5=GOOD,R4=?
5022 020232 001401      BEQ      5$         ;BIT 13 IS OK
5023 020234 104003      HLT      3          ;BIT FAILED TO CLR
5024 020236 012705 020000 5$: MOV      #BIT13,R5    ;RELOAD THE BIT
5025 020242 010513      MOV      R5, (R3)    ;WRITE BIT 13 TO THE REG
5026 020244 052777 000400 161140 BIS      #MRESET,@TXCSR ;RESET THE DEVICE
5027 020252 004737 005044 JSR      PC,SMALL    ;WAIT FOR RESET TO FINISH
5028 020256 011304      MOV      (R3),R4    ;GET BIT 13
5029 020260 032704 020000 BIT      #BIT13,R4   ;TEST BIT 13 FOR RESET CLR
5030 020264 001401      BEQ      7$         ;BIT 13 IS OK
5031 020266 104003      HLT      3          ;BIT FAILED TO CLEAR
5032 020270 010513 7$: MOV      R5, (R3)    ;SET BIT 13
5033 020272 005013      CLR      (R3)       ;CLR TRANSMITTER BUFFER REGISTER
5034 020274 011304      MOV      (R3),R4    ;READ THE TRANSMITTER BUFFER REGISTER
5035 020276 005005      CLR      R5          ;SET "EXPECTED"
5036 020300 020504      CMP      R5,R4       ;R5=GOOD,R4=?
5037 020302 001401      BEQ      10$        ;BIT 13 IS OK
5038 020304 104003      HLT      3          ;BIT FAILED TO CLEAR
5039 020306 104400 10$: SCOPE      ;SCOPE THIS TEST

```

```

;***** TEST 116 *****
;*TRANSMITTER BUFFER REGISTER BIT 14 READ ONLY DEVICE RESET AND CLEAR TEST
;*WRITE BIT 14 A ONE AND VERIFY A ZERO IS READ BACK
;*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.
;*****

```

```

;*****
;TEST 116
;*****

```

```

5054 020310 012737 000116 001226 TST116: MOV      #116,@TSTNO
5055 020316 012737 020422 001216 MOV      #TST117,NEXT
5056 020324 013703 001414 MOV      TXDBUF,R3   ;GET THE TRANSMITTER BUFFER REGISTER
5057 020330 012705 040000 MOV      #BIT14,R5   ;GET BIT 14
5058 020334 010513      MOV      R5, (R3)    ;WRITE BIT 14 TO TRANSMITTER BUFFER REGISTER
5059 020336 011304      MOV      (R3),R4    ;READ TRANSMITTER BUFFER REGISTER BACK
5060 020340 005005      CLR      R5          ;SET "EXPECTED"
5061 020342 020504      CMP      R5,R4       ;R5=GOOD,R4=?
5062 020344 001401      BEQ      5$         ;BIT 14 IS OK
5063 020346 104003      HLT      3          ;BIT FAILED TO CLR
5064 020350 012705 040000 5$: MOV      #BIT14,R5    ;RELOAD THE BIT
5065 020354 010513      MOV      R5, (R3)    ;WRITE BIT 14 TO THE REG
5066 020356 052777 000400 161026 BIS      #MRESET,@TXCSR ;RESET THE DEVICE
5067 020364 004737 005044 JSR      PC,SMALL    ;WAIT FOR RESET TO FINISH
5068 020370 011304      MOV      (R3),R4    ;GET BIT 14
5069 020372 032704 040000 BIT      #BIT14,R4   ;TEST BIT 14 FOR RESET CLR
5070 020376 001401      BEQ      7$         ;BIT 14 IS OK
5071 020400 104003      HLT      3          ;BIT FAILED TO CLEAR
5072 020402 010513 7$: MOV      R5, (R3)    ;SET BIT 14
5073 020404 005013      CLR      (R3)       ;CLR TRANSMITTER BUFFER REGISTER

```

TRANSMITTER BUFFER REGISTER READ ONLY TEST BIT 14

```

5074 020406 011304      MOV      (R3),R4      ;READ THE TRANSMITTER BUFFER REGISTER
5075 020410 005005      CLR      R5          ;SET "EXPECTED"
5076 020412 020504      CMP      R5,R4       ;R5=GOOD,R4=?
5077 020414 001401      BEQ      10$         ;BIT 14 IS OK
5078 020416 104003      HLT      3           ;BIT FAILED TO CLEAR
5079 020420 104400      10$:     SCOPE      ;SCOPE THIS TEST
5080
5081
5082

```

```

***** TEST 117 *****
*TRANSMITTER BUFFER REGISTER BIT 15 READ ONLY DEVICE RESET AND CLEAR TEST
*WRITE BIT 15 A ONE AND VERIFY A ZERO IS READ BACK
*REPEAT FOR DEVICE RESET AND CLR INSTRUCTIONS.
*****

```

```

*****
*
TEST 117
*
*****

```

```

5094 020422 012737 000117 001226 1ST117: MOV      #117,#TSTNO
5095 020430 012737 020534 001216      MOV      #TST120,NEXT
5096 020436 013703 001414      MOV      TXDBUF,R3      ;GET THE TRANSMITTER BUFFER REGISTER
5097 020442 012705 100000      MOV      #BIT15,R5     ;GET BIT 15
5098 020446 010513      MOV      R5,(R3)       ;WRITE BIT 15 TO TRANSMITTER BUFFER REGISTER
5099 020450 011304      MOV      (R3),R4       ;READ TRANSMITTER BUFFER REGISTER BACK
5100 020452 005005      CLR      R5            ;SET "EXPECTED"
5101 020454 020504      CMP      R5,R4         ;R5=GOOD,R4=?
5102 020456 001401      BEQ      5$           ;BIT 15 IS OK
5103 020460 104003      HLT      3            ;BIT FAILED TO CLR
5104 020462 012705 100000      5$:     MOV      #BIT15,R5     ;RELOAD THE BIT
5105 020466 010513      MOV      R5,(R3)       ;WRITE BIT 15 TO THE REG
5106 020470 052777 000400 160714      BIS      #MRESET,#TXCSR ;RESET THE DEVICE
5107 020476 004737 005044      JSR      PC,SMALL     ;WAIT FOR RESET TO FINISH
5108 020502 011304      MOV      (R3),R4       ;GET BIT 15
5109 020504 032704 100000      BIT      #BIT15,R4     ;TEST BIT 15 FOR RESET CLR
5110 020510 001401      BEQ      7$           ;BIT 15 IS OK
5111 020512 104003      HLT      3            ;BIT FAILED TO CLEAR
5112 020514 010513      7$:     MOV      R5,(R3)       ;SET BIT 15
5113 020516 005013      CLR      (R3)          ;CLR TRANSMITTER BUFFER REGISTER
5114 020520 011304      MOV      (R3),R4       ;READ THE TRANSMITTER BUFFER REGISTER
5115 020522 005005      CLR      R5            ;SET "EXPECTED"
5116 020524 020504      CMP      R5,R4         ;R5=GOOD,R4=?
5117 020526 001401      BEQ      10$          ;BIT 15 IS OK
5118 020530 104003      HLT      3            ;BIT FAILED TO CLEAR
5119 020532 104400      10$:     SCOPE      ;SCOPE THIS TEST
5120
5121
5122

```

```

***** TEST 120 *****
*PARAMETER STATUS REGISTER BIT 9 WRITE ONLY DEVICE RESET AND CLEAR TEST
*TEST THAT BIT 9 CANNOT BE WRITTEN AND READ
*BACK THE SAME. READ BIT 9,COMPLEMENT IT AND WRITE IT
*VERIFYING THAT IT DID NOT CHANGE. WRITE BIT 9.
*DO A DEVICE RESET AND VERIFY THE BIT WAS CLEARED.
*REPEAT FOR A CLR INSTRUCTION.
*****

```

```

5123
5124
5125
5126
5127
5128
5129

```

5130
5131
5132
5133
5134
5135
5136
5137
5138
5139
5140
5141
5142
5143
5144
5145
5146
5147
5148
5149
5150
5151
5152
5153
5154
5155
5156
5157
5158
5159
5160
5161
5162
5163
5164
5165
5166
5167
5168
5169
5170
5171
5172
5173
5174
5175
5176
5177
5178
5179
5180
5181
5182
5183
5184
5185

020534 012737 000120 001226
020542 012737 020662 001216
020550 013703 001410
020554 012705 001000
020560 011305
020562 010504
020564 032704 001000
020570 001401
020572 104003
020574 005104
020576 010413
020600 011304
020602 020504
020604 001401
020606 104003
020610 012705 001000
020614 010513
020616 052777 000400 160566
020624 004737 005044
020630 011304
020632 032704 001000
020636 001401
020640 104003
020642 010513
020644 005013
020646 011304
020650 005005
020652 020504
020654 001401
020656 104003
020660 104400

```

:*****
: TEST 120
:*****
:*****
:*****
TST120: MOV #120, @TSTNO
MOV @TST121, NEXT
MOV PARCSR, R3 ; GET THE PARAMETER STATUS REGISTER
MOV #BIT9, R5 ; GET BIT 9
MOV (R3), R5 ; READ THE REGISTER
MOV R5, R4 ; SAVE THE BIT
BIT #BIT9, R4 ; CHECK BIT 9
BEQ 1$ ; BIT 9 IS OK
HLT 3 ; BIT FAILED TO CLEAR
1$: COM R4 ; REVERSE THE BIT
MOV R4, (R3) ; WRITE THE BIT TO THE REG
MOV (R3), R4 ; READ IT BACK
CMP R5, R4 ; R5=GOOD, R4= ?
BEQ 3$ ; BR IF OK
HLT 3 ; COMPARE ERROR
3$: MOV #BIT9, R5 ; LOAD THE BIT
MOV R5, (R3) ; WRITE THE BIT TO THE REG
BIS #MRESET, @TXCSR ; RESET THE DEVICE
JSR PC, SMALL ; WAIT FOR RESET TO FINISH
MOV (R3), R4 ; GET BIT 9
BIT #BIT9, R4 ; TEST BIT 9 FOR 0 AFTER RESET
BEQ 4$ ; BIT 9 IS OK
HLT 3 ; BIT IS NOT A 0
4$: MOV R5, (R3) ; WRITE THE BIT TO THE REG
CLR (R3) ; CLR THE PARAMETER STATUS REGISTER
MOV (R3), R4 ; GET THE PARAMETER STATUS REGISTER
CLR R5 ; SET "EXPECTED"
CMP R5, R4 ; R5=GOOD, R4= ?
BEQ 10$ ; BIT 9 IS OK
HLT 3 ; BIT FAILED TO CLEAR
10$: SCOPE ; SCOPE THIS TEST

```

```

:***** TEST 121 *****
: *PARAMETER STATUS REGISTER BIT 12 WRITE ONLY DEVICE RESET AND CLEAR TEST
: *TEST THAT BIT 12 CANNOT BE WRITTEN AND READ
: *BACK THE SAME. READ BIT 12, COMPLEMENT IT AND WRITE IT
: *VERIFYING THAT IT DID NOT CHANGE. WRITE BIT 12.
: *DO A DEVICE RESET AND VERIFY THE BIT WAS CLEARED.
: *REPEAT FOR A CLR INSTRUCTION.
:*****

```

```

:*****
: TEST 121
:*****
:*****
TST121: MOV #121, @TSTNO

```

PARAMETER STATUS REGISTER WRITE ONLY TEST BIT 12

```

5186 020670 012737 021010 001216      MOV      #TST122,NEXT
5187 020676 013703 001410              MOV      PARCSR,R3          ;GET THE PARAMETER STATUS REGISTER
5188 020702 012705 010000              MOV      #BIT12,R5        ;GET BIT 12
5189 020706 011305              MOV      (R3),R5          ;READ THE REGISTER
5190 020710 010504              MOV      R5,R4            ;SAVE THE BIT
5191 020712 032704 010000              BIT      #BIT12,R4        ;CHECK BIT 12
5192 020716 001401              BEQ      1$               ;BIT 12 IS OK
5193 020720 104003              HLT      3                ;BIT FAILED TO CLEAR
5194 020722 005104              1$:    COM      R4          ;REVERSE THE BIT
5195 020724 010413              MOV      R4,(R3)         ;WRITE THE BIT TO THE REG
5196 020726 011304              MOV      (R3),R4         ;READ IT BACK
5197 020730 020504              CMP      R5,R4           ;R5=GOOD, R4= ?
5198 020732 001401              BEQ      3$               ;BR IF OK
5199 020734 104003              HLT      3                ;COMPARE ERROR
5200 020736 012705 010000              3$:    MOV      #BIT12,R5  ;LOAD THE BIT
5201 020742 010513              MOV      R5,(R3)         ;WRITE THE BIT TO THE REG
5202 020744 052777 000400 160440          BIS      #MRESET,ATXCSR  ;RESET THE DEVICE
5203 020752 004737 005044              JSR      PC,SMALL        ;WAIT FOR RESET TO FINISH
5204 020756 011304              MOV      (R3),R4         ;GET BIT 12
5205 020760 032704 010000              BIT      #BIT12,R4        ;TEST BIT 12 FOR 0 AFTER RESET
5206 020764 001401              BEQ      4$               ;BIT 12 IS OK
5207 020766 104003              HLT      3                ;BIT IS NOT A 0
5208 020770 010513              4$:    MOV      R5,(R3)         ;WRITE THE BIT TO THE REG
5209 020772 005013              CLR      (R3)            ;CLR THE PARAMETER STATUS REGISTER
5210 020774 011304              MOV      (R3),R4         ;GET THE PARAMETER STATUS REGISTER
5211 020776 005005              CLR      R5              ;SET "EXPECTED"
5212 021000 020504              CMP      R5,R4           ;R5=GOOD, R4= ?
5213 021002 001401              BEQ      10$              ;BIT 12 IS OK
5214 021004 104003              HLT      3                ;BIT FAILED TO CLEAR
5215 021006 104400              10$:   SCOPE              ;SCOPE THIS TEST
5216
5217
5218
5219
5220
5221
5222
5223
5224
5225
5226
5227
5228
5229
5230
5231
5232

```

```

***** TEST 122 *****
*PARAMETER STATUS REGISTER BIT 15 WRITE ONLY DEVICE RESET AND CLEAR TEST
*TEST THAT BIT 15 CANNOT BE WRITTEN AND READ
*BACK THE SAME. READ BIT 15, COMPLEMENT IT AND WRITE IT
*VERIFYING THAT IT DID NOT CHANGE. WRITE BIT 15.
*DO A DEVICE RESET AND VERIFY THE BIT WAS CLEARED.
*REPEAT FOR A CLR INSTRUCTION.
*****

```

```

:*****
:
:TEST 122
:
:*****
:*****

```

```

5233 021010 012737 000122 001226      TST122: MOV      #122,@TSTNO
5234 021016 012737 021136 001216      MOV      #TST123,NEXT
5235 021024 013703 001410              MOV      PARCSR,R3          ;GET THE PARAMETER STATUS REGISTER
5236 021030 012705 100000              MOV      #BIT15,R5        ;GET BIT 15
5237 021034 011305              MOV      (R3),R5          ;READ THE REGISTER
5238 021036 010504              MOV      R5,R4            ;SAVE THE BIT
5239 021040 032704 100000              BIT      #BIT15,R4        ;CHECK BIT 15
5240 021044 001401              BEQ      1$               ;BIT 15 IS OK
5241 021046 104003              HLT      3                ;BIT FAILED TO CLEAR

```

PARAMETER STATUS REGISTER WRITE ONLY TEST BIT 15

```

5242 021050 005104      1$:  COM      R4      ;REVERSE THE BIT
5243 021052 010413      MOV      R4,(R3) ;WRITE THE BIT TO THE REG
5244 021054 011304      MOV      (R3),R4 ;READ IT BACK
5245 021056 020504      CMP      R5,R4   ;R5=GOOD, R4= ?
5246 021060 001401      BEQ      3$     ;BR IF OK
5247 021062 104003      HLT      3      ;COMPARE ERROR
5248 021064 012705 100000 3$:  MOV      #BIT15,R5 ;LOAD THE BIT
5249 021070 010513      MOV      R5,(R3) ;WRITE THE BIT TO THE REG
5250 021072 052777 000400 160312 BIS      #MRESET, @TXCSR ;RESET THE DEVICE
5251 021100 004737 005044 JSR      PC,SMALL ;WAIT FOR RESET TO FINISH
5252 021104 011304      MOV      (R3),R4 ;GET BIT 15
5253 021106 032704 100000 BIT      #BIT15,R4 ;TEST BIT 15 FOR 0 AFTER RESET
5254 021112 001401      BEQ      4$     ;BIT 15 IS OK
5255 021114 104003      HLT      3      ;BIT IS NOT A 0
5256 021116 010513      4$:  MOV      R5,(R3) ;WRITE THE BIT TO THE REG
5257 021120 005013      CLR      (R3)   ;CLR THE PARAMETER STATUS REGISTER
5258 021122 011304      MOV      (R3),R4 ;GET THE PARAMETER STATUS REGISTER
5259 021124 005005      CLR      R5    ;SET "EXPECTED"
5260 021126 020504      CMP      R5,R4 ;R5=GOOD, R4= ?
5261 021130 001401      BEQ      10$    ;BIT 15 IS OK
5262 021132 104003      HLT      3      ;BIT FAILED TO CLEAR
5263 021134 104400      10$: SCOPE     ;SCOPE THIS TEST
5264
5265
5266
5267
5268
5269
5270
5271
5272
5273
5274
5275
5276
5277
5278
5279
5280

```

```

;***** TEST 123 *****
;*TRANSMITTER CONTROL REGISTER BIT 8 WRITE ONLY DEVICE RESET AND CLEAR TEST
;*TEST THAT BIT 8 CANNOT BE WRITTEN AND READ
;*BACK THE SAME. READ BIT 8, COMPLEMENT IT AND WRITE IT
;*VERIFYING THAT IT DID NOT CHANGE. WRITE BIT 8.
;*DO A DEVICE RESET AND VERIFY THE BIT WAS CLEARED.
;*REPEAT FOR A CLR INSTRUCTION.
;*****

```

```

5281 021136 012737 000123 001226 1ST123: MOV      #123,@TSTNO
5282 021144 012737 021270 001216 MOV      #TST124,NEXT
5283 021152 013703 001412 MOV      TXCSR,R3 ;GET THE TRANSMITTER CONTROL REGISTER
5284 021156 012705 000200 MOV      #BIT7,R5 ;SET EXPECTED BIT
5285 021162 011304 MOV      (R3),R4 ;READ THE REGISTER
5286 021164 032704 000400 BIT      #BIT8,R4 ;CHECK BIT 8
5287 021170 001401 BEQ      1$     ;BIT 8 IS OK
5288 021172 104003 HLT      3      ;BIT FAILED TO CLEAR
5289 021174 005104      1$:  COM      R4   ;REVERSE THE BIT
5290 021176 010413      MOV      R4,(R3) ;WRITE THE BIT TO THE REG
5291 021200 004737 005044 JSR      PC,SMALL ;WAIT FOR DEVICE RESET TO FINISH
5292 021204 011304      MOV      (R3),R4 ;READ IT BACK
5293 021206 020504      CMP      R5,R4 ;R5=GOOD, R4= ?
5294 021210 001401      BEQ      3$     ;BR IF OK
5295 021212 104003      HLT      3      ;COMPARE ERROR
5296 021214 012704 000400 3$:  MOV      #BIT8,R4 ;LOAD THE BIT
5297 021220 010413      MOV      R4,(R3) ;WRITE THE BIT TO THE REG

```

TRANSMITTER CONTROL REGISTER WRITE ONLY TEST BIT 8

```

5298 021222 052777 000400 160162      BIS      #MRESET, BTXCSR  ;RESET THE DEVICE
5299 021230 004737 005044              JSR      PC, SMALL    ;WAIT FOR RESET TO FINISH
5300 021234 011304              MOV      (R3), R4     ;GET BIT 8
5301 021236 032704 000400      BIT      #BIT8, R4    ;TEST BIT 8 FOR 0 AFTER RESET
5302 021242 001401              BEQ      4$           ;BIT 8 IS OK
5303 021244 104003              HLT      3            ;BIT IS NOT A 0
5304 021246 012704 000400      4$:     MOV      #BIT8, R4 ;LOAD THE BIT
5305 021252 010413              MOV      R4, (R3)    ;WRITE THE BIT TO THE REG
5306 021254 005013              CLR      (R3)        ;CLR THE TRANSMITTER CONTROL REGISTER
5307 021256 011304              MOV      (R3), R4    ;GET THE TRANSMITTER CONTROL REGISTER
5308 021260 020504              CMP      R5, R4      ;R5=GOOD, R4=?
5309 021262 001401              BEQ      10$         ;BIT 8 IS OK
5310 021264 104003              HLT      3            ;BIT FAILED TO CLEAR
5311 021266 104400      10$:    SCOPE        ;SCOPE THIS TEST

```

5312
5313
5314
5315
5316
5317
5318
5319
5320
5321
5322
5323
5324
5325
5326
5327
5328
5329
5330
5331
5332
5333
5334
5335
5336
5337
5338
5339
5340
5341
5342
5343
5344
5345
5346
5347
5348
5349
5350
5351
5352
5353

```

;***** TEST 124 *****
;RECEIVER CONTROL REGISTER READ/WRITE BIT TEST.
;SET BIT8, VERIFY BIT8 WAS SET.
;CLEAR BIT8, VERIFY BIT8 WAS CLEARED.
;*****

```

```

;*****
; TEST 124
;*****
;*****
;*****
TST124: MOV      #124, #TSTNO ;SET REGISTER TO BE TESTED.
        MOV      #TST125, NEXT ;SET "EXPECTED"
        MOV      RXCSR, R3    ;WRITE THE REGISTER.
        MOV      #BIT8, R5    ;READ THE REGISTER.
        MOV      R5, (R3)     ;R5=GOOD; R4=UNKNOWN.
        MOV      (R3), R4     ;ARE THEY THE SAME?
        CMP      R5, R4      ;COMPARISON ERROR.
        BEQ      1$          ;CLEAR BIT8
        HLT      3            ;READ THE REGISTER.
        1$:     MOV      (R3), R4 ;SET "EXPECTED"
        CLR      R5          ;R5=GOOD; R4=?
        CMP      R5, R4      ;BR IF OK
        BEQ      2$          ;COMPARISON ERROR
        HLT      3            ;SCOPE THIS TEST
        2$:     SCOPE

```

```

;***** TEST 125 *****
;RECEIVER CONTROL REGISTER READ/WRITE BIT TEST.
;SET BIT6, VERIFY BIT6 WAS SET.
;CLEAR BIT6, VERIFY BIT6 WAS CLEARED.
;*****

```

```

;*****
; TEST 125
;*****

```

RECEIVER CONTROL REGISTER READ/WRITE TEST BIT 6

5354
5355
5356 021344 012737 000125 001226
5357 021352 012737 021420 001216
5358 021360 013703 001404
5359 021364 012705 000100
5360 021370 010513
5361 021372 011304
5362 021374 020504
5363 021376 001401
5364 021400 104003
5365 021402 040513
5366 021404 011304
5367 021406 005005
5368 021410 020504
5369 021412 001401
5370 021414 104003
5371 021416 104400
5372
5373
5374
5375
5376
5377
5378
5379
5380
5381
5382
5383
5384
5385
5386 021420 012737 000126 001226
5387 021426 012737 021474 001216
5388 021434 013703 001404
5389 021440 012705 000040
5390 021444 010513
5391 021446 011304
5392 021450 020504
5393 021452 001401
5394 021454 104003
5395 021456 040513
5396 021460 011304
5397 021462 005005
5398 021464 020504
5399 021466 001401
5400 021470 104003
5401 021472 104400
5402
5403
5404
5405
5406
5407
5408
5409

```

:*****
:*****
†ST125: MOV #125, #TSTNO
MOV #TST126, NEXT
MOV RXCSR, R3 ;SET REGISTER TO BE TESTED.
MOV #BIT6, R5 ;SET "EXPECTED"
MOV R5, (R3) ;WRITE THE REGISTER.
MOV (R3), R4 ;READ THE REGISTER.
CMP R5, R4 ;R5=GOOD; R4=UNKNOWN.
BEQ 1$ ;ARE THEY THE SAME?
HLT 3 ;COMPARISON ERROR.
1$: BIC R5, (R3) ;CLEAR BIT6
MOV (R3), R4 ;READ THE REGISTER.
CLR R5 ;SET "EXPECTED"
CMP R5, R4 ;R5=GOOD; R4=?
BEQ 2$ ;BR IF OK
HLT 3 ;COMPARISON ERROR
2$: SCOPE ;SCOPE THIS TEST

```

```

:***** TEST 126 *****
:RECEIVER CONTROL REGISTER READ/WRITE BIT TEST.
:SET BITS, VERIFY BITS WAS SET.
:CLEAR BITS, VERIFY BITS WAS CLEARED.
:*****

```

```

:*****
:*****
: TEST 126
:*****
:*****
†ST126: MOV #126, #TSTNO
MOV #TST127, NEXT
MOV RXCSR, R3 ;SET REGISTER TO BE TESTED.
MOV #BIT5, R5 ;SET "EXPECTED"
MOV R5, (R3) ;WRITE THE REGISTER.
MOV (R3), R4 ;READ THE REGISTER.
CMP R5, R4 ;R5=GOOD; R4=UNKNOWN.
BEQ 1$ ;ARE THEY THE SAME?
HLT 3 ;COMPARISON ERROR.
1$: BIC R5, (R3) ;CLEAR BITS
MOV (R3), R4 ;READ THE REGISTER.
CLR R5 ;SET "EXPECTED"
CMP R5, R4 ;R5=GOOD; R4=?
BEQ 2$ ;BR IF OK
HLT 3 ;COMPARISON ERROR
2$: SCOPE ;SCOPE THIS TEST

```

```

:***** TEST 127 *****
:RECEIVER CONTROL REGISTER READ/WRITE BIT TEST.
:SET BIT4, VERIFY BIT4 WAS SET.
:CLEAR BIT4, VERIFY BIT4 WAS CLEARED.
:*****

```

5410
5411
5412
5413
5414
5415
5416
5417
5418
5419
5420
5421
5422
5423
5424
5425
5426
5427
5428
5429
5430
5431
5432
5433
5434
5435
5436
5437
5438
5439
5440
5441
5442
5443
5444
5445
5446
5447
5448
5449
5450
5451
5452
5453
5454
5455
5456
5457
5458
5459
5460
5461
5462
5463
5464
5465

021474 012737 000127 001226
021502 012737 021550 001216
021510 013703 001404
021514 012705 000020
021520 010513
021522 011304
021524 020504
021526 001401
021530 104003
021532 040513
021534 011304
021536 005005
021540 020504
021542 001401
021544 104003
021546 104400

```
::*****  
: TEST 127 *  
:*****  
:*****  
TST127: MOV #127, R3 ;SET REGISTER TO BE TESTED.  
MOV #TST130, R4 ;SET "EXPECTED"  
MOV TXCSR, R3 ;WRITE THE REGISTER.  
MOV #BIT4, R5 ;READ THE REGISTER.  
MOV R5, (R3) ;R5=GOOD; R4=UNKNOWN.  
MOV (R3), R4 ;ARE THEY THE SAME?  
CMP R5, R4 ;COMPARISON ERROR.  
BEQ 1$ ;CLEAR BIT4  
HLT 3 ;READ THE REGISTER.  
1$: BIC R5, (R3) ;SET "EXPECTED"  
MOV (R3), R4 ;R5=GOOD; R4=?  
CLR R5 ;BR IF OK  
CMP R5, R4 ;COMPARISON ERROR  
BEQ 2$ ;SCOPE THIS TEST  
HLT 3  
2$: SCOPE
```

***** TEST 130 *****
*TRANSMITTER CONTROL REGISTER READ/WRITE BIT TEST.
*SET BIT6, VERIFY BIT6 WAS SET.
*CLEAR BIT6, VERIFY BIT6 WAS CLEARED.

021550 012737 000130 001226
021556 012737 021634 001216
021564 013703 001412
021570 012705 000100
021574 010513
021576 011304
021600 042704 000200
021604 020504
021606 001401
021610 104003
021612 040513
021614 011304
021616 042704 000200
021622 005005
021624 020504
021626 001401
021630 104003
021632 104400

```
::*****  
: TEST 130 *  
:*****  
:*****  
TST130: MOV #130, R3 ;SET REGISTER TO BE TESTED.  
MOV #TST131, R4 ;SET "EXPECTED"  
MOV TXCSR, R3 ;WRITE THE REGISTER.  
MOV #BIT6, R5 ;READ THE REGISTER.  
MOV R5, (R3) ;R5=GOOD; R4=UNKNOWN.  
MOV (R3), R4 ;ARE THEY THE SAME?  
BIC #BIT7, R4 ;CLEAR UNWANTED BITS  
CMP R5, R4 ;R5=GOOD; R4=UNKNOWN.  
BEQ 1$ ;ARE THEY THE SAME?  
HLT 3 ;COMPARISON ERROR.  
1$: BIC R5, (R3) ;CLEAR BIT6  
MOV (R3), R4 ;READ THE REGISTER.  
BIC #BIT7, R4 ;CLEAR UNWANTED BITS  
CLR R5 ;SET "EXPECTED"  
CMP R5, R4 ;R5=GOOD; R4=?  
BEQ 2$ ;BR IF OK  
HLT 3 ;COMPARISON ERROR  
2$: SCOPE ;SCOPE THIS TEST
```


RECEIVER CONTROL REGISTER READ/WRITE TEST BIT 1

***** TEST 131 *****
*RECEIVER CONTROL REGISTER READ/WRITE BIT TEST.
*SET BIT1, VERIFY BIT1 WAS SET.
*CLEAR BIT1, VERIFY BIT1 WAS CLEARED.

*
: TEST 131
*

5466
5467
5468
5469
5470
5471
5472
5473
5474
5475
5476
5477
5478 021634 012737 000131 001226
5479 021642 012737 021720 001216
5480 021650 013703 001404
5481 021654 012705 000002
5482 021660 010513
5483 021662 011304
5484 021664 042704 177001
5485 021670 020504
5486 021672 001401
5487 021674 104003
5488 021676 040513
5489 021700 011304
5490 021702 042704 177001
5491 021706 005005
5492 021710 020504
5493 021712 001401
5494 021714 104003
5495 021716 104400
5496
5497
5498
5499
5500
5501
5502
5503
5504
5505
5506
5507
5508
5509
5510 021720 012737 000132 001226
5511 021726 012737 022004 001216
5512 021734 013703 001404
5513 021740 012705 000004
5514 021744 010513
5515 021746 011304
5516 021750 042704 177001
5517 021754 020504
5518 021756 001401
5519 021760 104003
5520 021762 040513
5521 021764 011304

TST131: MOV #131, #TSTNO
MOV #TST132, NEXT
MOV RXCSR, R3 ; SET REGISTER TO BE TESTED.
MOV #BIT1, R5 ; SET "EXPECTED"
MOV R5, (R3) ; WRITE THE REGISTER.
MOV (R3), R4 ; READ THE REGISTER.
BIC #177001, R4 ; CLEAR UNWANTED BITS
CMP R5, R4 ; R5=GOOD; R4=UNKNOWN.
BEQ 1\$; ARE THEY THE SAME?
HLT 3 ; COMPARISON ERROR.
1\$: BIC R5, (R3) ; CLEAR BIT1
MOV (R3), R4 ; READ THE REGISTER.
BIC #177001, R4 ; CLEAR UNWANTED BITS
CLR R5 ; SET "EXPECTED"
CMP R5, R4 ; R5=GOOD; R4=?
BEQ 2\$; BR IF OK
HLT 3 ; COMPARISON ERROR
2\$: SCOPE ; SCOPE THIS TEST

***** TEST 132 *****
*RECEIVER CONTROL REGISTER READ/WRITE BIT TEST.
*SET BIT2, VERIFY BIT2 WAS SET.
*CLEAR BIT2, VERIFY BIT2 WAS CLEARED.

*
: TEST 132
*

TST132: MOV #132, #TSTNO
MOV #TST133, NEXT
MOV RXCSR, R3 ; SET REGISTER TO BE TESTED.
MOV #BIT2, R5 ; SET "EXPECTED"
MOV R5, (R3) ; WRITE THE REGISTER.
MOV (R3), R4 ; READ THE REGISTER.
BIC #177001, R4 ; CLEAR UNWANTED BITS
CMP R5, R4 ; R5=GOOD; R4=UNKNOWN.
BEQ 1\$; ARE THEY THE SAME?
HLT 3 ; COMPARISON ERROR.
1\$: BIC R5, (R3) ; CLEAR BIT2
MOV (R3), R4 ; READ THE REGISTER.

```

5522 021766 042704 177001      BIC    #177001,R4      ;CLEAR UNWANTED BITS
5523 021772 005005              CLR    R5              ;SET "EXPECTED"
5524 021774 020504              CMP    R5,R4           ;R5=GOOD; R4=?
5525 021776 001401              BEQ    2$              ;BR IF OK
5526 022000 104003              HLT    3               ;COMPARISON ERROR
5527 022002 104400      2$:   SCOPE            ;SCOPE THIS TEST

```

```

***** TEST 133 *****
*RECEIVER CONTROL REGISTER READ/WRITE BIT TEST.
*SET BIT3, VERIFY BIT3 WAS SET.
*CLEAR BIT3, VERIFY BIT3 WAS CLEARED.
*****

```

```

*****
*
* TEST 133
*
*****

```

```

5542 022004 012737 000133 001226 TST133: MOV    #133,2#TSTNO
5543 022012 012737 022070 001216      MOV    #TST134,NEXT
5544 022020 013703 001404              MOV    RXCSR,R3       ;SET REGISTER TO BE TESTED.
5545 022024 012705 000010              MOV    #BIT3,R5       ;SET "EXPECTED"
5546 022030 010513              MOV    R5,(R3)        ;WRITE THE REGISTER.
5547 022032 011304              MOV    (R3),R4        ;READ THE REGISTER.
5548 022034 042704 177001      BIC    #177001,R4     ;CLEAR UNWANTED BITS
5549 022040 020504              CMP    R5,R4           ;R5=GOOD; R4=UNKNOWN.
5550 022042 001401              BEQ    1$              ;ARE THEY THE SAME?
5551 022044 104003              HLT    3               ;COMPARISON ERROR.
5552 022046 040513      1$:   BIC    R5,(R3)     ;CLEAR BIT3
5553 022050 011304              MOV    (R3),R4        ;READ THE REGISTER.
5554 022052 042704 177001      BIC    #177001,R4     ;CLEAR UNWANTED BITS
5555 022056 005005              CLR    R5              ;SET "EXPECTED"
5556 022060 020504              CMP    R5,R4           ;R5=GOOD; R4=?
5557 022062 001401              BEQ    2$              ;BR IF OK
5558 022064 104003              HLT    3               ;COMPARISON ERROR
5559 022066 104400      2$:   SCOPE            ;SCOPE THIS TEST

```

```

***** TEST 134 *****
*RECEIVER CONTROL REGISTER READ/WRITE BIT 6 RESET AND CLEAR TEST
*WRITE BIT 6, AND TEST THAT IT WILL BE CLEARED AFTER A
*DEVICE RESET AND RECEIVER CONTROL REGISTER CLR INSTRUCTION
*****

```

```

*****
*
* TEST 134
*
*****

```

```

5574 022070 012737 000134 001226 TST134: MOV    #134,2#TSTNO
5575 022076 012737 022164 001216      MOV    #TST135,NEXT
5576 022104 013703 001404              MOV    RXCSR,R3       ;GET THE RECEIVER CONTROL REGISTER
5577 022110 052713 000100              BIS    #BIT6,(R3)     ;SET BIT 6 AT RECEIVER CONTROL REGISTER

```

RECEIVER CONTROL REGISTER RESET AND CLEAR TEST BIT 6

```

5578 022114 005005 CLR R5 ;SET "EXPECTED"
5579 022116 052777 000400 157266 BIS #MRESET, @TXCSR ;RESET THE DEVICE
5580 022124 004737 005044 JSR PC, SMALL ;WAIT FOR RESET TO FINISH
5581 022130 032713 000100 BIT #BIT6, (R3) ;TEST BIT 6
5582 022134 001402 BEQ 1$ ;BIT 6 IS CLEARED
5583 022136 011304 MOV (R3), R4 ;LOAD "FOUND"
5584 022140 104003 HLT 3 ;BIT 6 IS SET AND SHOULDN'T BE
5585 022142 052713 000100 1$: BIS #BIT6, (R3) ;SET BIT 6 AGAIN
5586 022146 005013 CLR (R3) ;CLEAR THE RECEIVER CONTROL REGISTER
5587 022150 032713 000100 BIT #BIT6, (R3) ;TEST TO SEE IF BIT 6 CLEARED
5588 022154 001402 BEQ 2$ ;BIT 6 IS OK
5589 022156 011304 MOV (R3), R4 ;LOAD "FOUND"
5590 022160 104003 HLT 3 ;BIT 6 FAILED TO CLEAR
5591 022162 104400 2$: SCOPE ;SCOPE THIS TEST

```

```

***** TEST 135 *****
*RECEIVER CONTROL REGISTER READ/WRITE BIT 5 RESET AND CLEAR TEST
*WRITE BIT 5, AND TEST THAT IT WILL BE CLEARED AFTER A
*DEVICE RESET AND RECEIVER CONTROL REGISTER CLR INSTRUCTION
*****

```

```

*****
*
* TEST 135
*
*****

```

```

5600
5601
5602
5603
5604
5605
5606 022164 012737 000135 001226 TST135: MOV #135, @TSTNO
5607 022172 012737 022260 001216 MOV #TST136, NEXT
5608 022200 013703 001404 MOV RXCSR, R3 ;GET THE RECEIVER CONTROL REGISTER
5609 022204 052713 000040 BIS #BIT5, (R3) ;SET BIT 5 AT RECEIVER CONTROL REGISTER
5610 022210 005005 CLR R5 ;SET "EXPECTED"
5611 022212 052777 000400 157172 BIS #MRESET, @TXCSR ;RESET THE DEVICE
5612 022220 004737 005044 JSR PC, SMALL ;WAIT FOR RESET TO FINISH
5613 022224 032713 000040 BIT #BIT5, (R3) ;TEST BIT 5
5614 022230 001402 BEQ 1$ ;BIT 5 IS CLEARED
5615 022232 011304 MOV (R3), R4 ;LOAD "FOUND"
5616 022234 104003 HLT 3 ;BIT 5 IS SET AND SHOULDN'T BE
5617 022236 052713 000040 1$: BIS #BIT5, (R3) ;SET BIT 5 AGAIN
5618 022242 005013 CLR (R3) ;CLEAR THE RECEIVER CONTROL REGISTER
5619 022244 032713 000040 BIT #BIT5, (R3) ;TEST TO SEE IF BIT 5 CLEARED
5620 022250 001402 BEQ 2$ ;BIT 5 IS OK
5621 022252 011304 MOV (R3), R4 ;LOAD "FOUND"
5622 022254 104003 HLT 3 ;BIT 5 FAILED TO CLEAR
5623 022256 104400 2$: SCOPE ;SCOPE THIS TEST

```

```

***** TEST 136 *****
*TRANSMITTER CONTROL REGISTER READ/WRITE BIT 6 RESET AND CLEAR TEST
*WRITE BIT 6, AND TEST THAT IT WILL BE CLEARED AFTER A
*DEVICE RESET AND TRANSMITTER CONTROL REGISTER CLR INSTRUCTION
*****

```

```

*****
*

```

5624
5625
5626
5627
5628
5629
5630
5631
5632
5633

TRANSMITTER CONTROL REGISTER RESET AND CLEAR TEST BIT 6

```

5634      : TEST 136
5635      : *****
5636      : *****
5637      : *****
5638 022260 012737 000136 001226 TST136: MOV #136, @TSTNO
5639 022266 012737 022354 001216      MOV #TST137, NEXT
5640 022274 013703 001412      MOV TXCSR, R3 ; GET THE TRANSMITTER CONTROL REGISTER
5641 022300 052713 000100      BIS #BIT6, (R3) ; SET BIT 6 AT TRANSMITTER CONTROL REGISTER
5642 022304 005005      CLR R5 ; SET "EXPECTED"
5643 022306 052777 000400 157076      BIS #MRESET, @TXCSR ; RESET THE DEVICE
5644 022314 004737 005044      JSR PC, SMALL ; WAIT FOR RESET TO FINISH
5645 022320 032713 000100      BIT #BIT6, (R3) ; TEST BIT 6
5646 022324 001402      BEQ 1$ ; BIT 6 IS CLEARED
5647 022326 011304      MOV (R3), R4 ; LOAD "FOUND"
5648 022330 104003      HLT 3 ; BIT 6 IS SET AND SHOULDN'T BE
5649 022332 052713 000100 1$: BIS #BIT6, (R3) ; SET BIT 6 AGAIN
5650 022336 005013      CLR (R3) ; CLEAR THE TRANSMITTER CONTROL REGISTER
5651 022340 032713 000100      BIT #BIT6, (R3) ; TEST TO SEE IF BIT 6 CLEARED
5652 022344 001402      BEQ 2$ ; BIT 6 IS OK
5653 022346 011304      MOV (R3), R4 ; LOAD "FOUND"
5654 022350 104003      HLT 3 ; BIT 6 FAILED TO CLEAR
5655 022352 104400 2$: SCOPE ; SCOPE THIS TEST
5656
5657
5658      : ***** TEST 137 *****
5659      : *THIS TEST CHECKS THE MAINTENANCE CLOCK
5660      : *USED THROUGHOUT THE REMAINING DIAGNOSTICS
5661      : *****
5662
5663      : *****
5664      : *****
5665      : TEST 137
5666      : *****
5667      : *****
5668      : *****
5669 022354 012737 000137 001226 TST137: MOV #137, @TSTNO
5670 022362 012737 022510 001216      MOV #TST140, NEXT
5671 022370 052777 000400 157014      BIS #MRESET, @TXCSR ; RESET THE DEVICE
5672 022376 004737 005044      JSR PC, SMALL ; WAIT FOR RESET TO FINISH
5673 022402 005037 001236      CLR TEMP1 ; TEST TIM SETUP
5674 022406 052777 004000 156776      BIS #SYSTST, @TXCSR ; ENTER SYSTEM TST MODE TO TURN ON CLOCK
5675 022414 032777 004000 156772 1$: BIT #TIMER, @TXDBUF ; CHECK THE CLOCK BIT
5676 022422 001407      BEQ 2$ ; BR IF OFF
5677 022424 005237 001236      INC TEMP1 ; INC WAIT LOOP
5678 022430 022737 177777 001236      CMP #-1, TEMP1 ; CHECK FOR LOOP TO BE DONE
5679 022436 001366      BNE 1$ ; BR IF MORE TIME TO WAIT
5680 022440 104000      HLT ; TIMER CLOCK BIT FAILED TO CLEAR
5681 022442 005037 001236 2$: CLR TEMP1 ; SECOND HALF SETUP
5682 022446 032777 004000 156740 3$: BIT #TIMER, @TXDBUF ; CHECK THE CLOCK BIT
5683 022454 001007      BNE 4$ ; BR IF ON
5684 022456 005237 001236      INC TEMP1 ; INC THE WAIT LOOP
5685 022462 022737 177777 001236      CMP #-1, TEMP1 ; CHECK FOR LOPP DONE
5686 022470 001366      BNE 3$ ; BR IF MORE TIME TO WAIT
5687 022472 104000      HLT ; TIMER BIT FAILED TO SET
5688 022474
5689 022474 052777 000400 156710 4$: BIS #MRESET, @TXCSR ; RESET THE DEVICE

```

5690 022502 004737 005044
5691 022506 104400

JSR PC, SMALL ;WAIT FOR RESET TO FINISH
SCOPE ;SCOPE THIS TEST

***** TEST 140 *****
*THIS TEST WILL PERFORM STATIC TRANSMITTER FUNCTIONS
*IN MAINTENANCE MODE. IT WILL PROVE THE INTERACTION
*OF SEND, DONE AND TSOM.

5692
5693
5694
5695
5696
5697
5698
5699

*
TEST 140
*

5700
5701
5702
5703
5704
5705
5706

5707 022510 012737 000140 001226
5708 022516 012737 022664 001216
5709 022524 013702 001414
5710 022530 013703 001412
5711 022534 005012
5712 022536 052777 000400 156646
5713 022544 004737 005044
5714 022550 052777 001000 156632
5715 022556 052713 014000
5716 022562 052713 000020
5717 022566 052705 014220
5718 022572 011304
5719 022574 020504
5720 022576 001401
5721 022600 104003
5722 022602 005005
5723 022604 011204
5724 022606 042704 170000
5725 022612 020504
5726 022614 001405
5727 022616 013703 001414
5728 022622 104003
5729 022624 013703 001412
5730 022630 010512
5731 022632 032713 000200
5732 022636 001401
5733 022640 104005
5734 022642 052712 000400
5735 022646 104412 000017
5736 022652 032713 000200
5737 022656 001001
5738 022660 104006
5739 022662 104400

TST140: MOV #140, @TSTNO
MOV #TST141, NEXT
MOV TXDBUF, R2 ;LOAD TX BUFFER
MOV TXCSR, R3 ;LOAD TX CONTROL REGISTER
CLR (R2) ;CLEAR BUFFER
BIS #MRESET, @TXCSR ;RESET THE DEVICE
JSR PC, SMALL ;WAIT FOR RESET TO FINISH
BIS #CRCEN, @PARCSR ;TURN OFF CRC
BIS #MMODE, (R3) ;ENTER M/M - PROGRAM NOW CLOCKING
BIS #SEND, (R3) ;ASSERT SEND
BIS #SEND!TXDONE!MMODE, R5 ;SET "EXPECTED"
MOV (R3), R4 ;READ TX CONTROL REGISTER
CMP R5, R4 ;ARE THEY EQUAL?
BEQ 1\$;BR IF YES
HLT 3 ;NO
1\$: CLR R5 ;"SET EXPECTED"
MOV (R2), R4 ;READ TX BUFFER
BIC #170000, R4 ;MASK
CMP R5, R4 ;R5=GOOD, R4=?
BEQ 2\$;ARE THEY EQUAL?
MOV TXDBUF, R3 ;ERROR MESSAGE SETUP
HLT 3 ;NO
2\$: MOV TXCSR, R3 ;RETURN TO NORMAL
MOV R5, (R2) ;LOAD BUFFER
BIT #TXDONE, (R3) ;TEST DONE
BEQ 3\$
HLT 5 ;BIT FAILED TO CLR
3\$: BIS #TSOM, (R2) ;SET TSOM
PKCLK #15.
BIT #TXDONE, (R3) ;TEST DONE
BNE 4\$
HLT 6 ;BIT FAILED TO SET
4\$: SCOPE ;SCOPE THIS TEST

5740
5741
5742
5743
5744
5745

***** TEST 141 *****
*THIS TEST CHECKS THE STATIC FUNCTIONS OF THE TRANSMITTER
*IN MAINTENANCE MODE. THIS TEST PROVES THE INTERACTION OF
*TXDBUF, TXACT, TSOM, TRANSMITTED DATA AND DONE.

```

;:*****
:*****
:*****
TEST 141
:*****
:*****
†ST141: MOV      #141, @TSTNO
MOV      @TST142, NEXT
MOV      TXCSR, R3      ;LOAD CONTROL REGISTER
MOV      TXDBUF, R2     ;LOAD BUFFER
BIS      #MRESET, @TXCSR ;RESET THE DEVICE
JSR      PC_SMALL      ;WAIT FOR RESET TO FINISH
CLR      (R2)          ;RESET TXDONE
BIS      #MMODE, (R3)   ;ENTER M/M--PROGRAM CLOCKING
BIS      #CRCEN, @PARCSR ;TURN OFF CRC
BIS      #SEND, (R3)   ;SET SEND
CLR      R5            ;SET "EXPECTED".
MOV      R5, (R2)      ;LOAD TX BUFFER
MOV      #TSOM, (R2)   ;TURN ON TSOM
PKCLK    ,6.           ;SYNC UP DUP
BIS      #CLK, @TXCSR  ;POKE CLOCK UP
BIT      #TXACT, (R3)  ;IS TXACT HIGH?
BNE      1$            ;BR IF SET
HLT      7             ;TXACT FAILED TO SET

1$: BIC      #CLK, @TXCSR      ;POKE CLOCK DOWN
PKCLK    ,13.               ;PUSH OUT DATA
BIT      #SEND, (R3)       ;CHECK SEND
BNE      2$                ;BR IF YES
HLT      17                ;BIT FAILED TO SET
2$: MOV      #2$, LOCK
MOV      #TSOM, R5
MOV      (R2), R4          ;SETUPFOR SW 09
BIC      #170000, R4       ;SET "EXPECTED".
CMP      R5, R4           ;GET THE BUFFER REG
BEQ      3$                ;MASK CRC BITS
MOV      TXDBUF, R3       ;R5=GOOD, R4=?
HLT      3                 ;BR IF A MATCH
SCOPI   ;ERROR MESSAGE SETUP
3$: MOV      TXCSR, R3     ;BIT FAILED TO SET
MOV      #3$, LOCK        ;SW09=1?
BIC      #TSOM, (R2)      ;LOAD TRANSMITTER CSR
BIT      #TXDONE, (R3)   ;SW09 SETUP
BEQ      4$                ;CLR TSOM
HLT      5                 ;TEST DONE
4$: MOV      #4$, LOCK     ;BR IF CLEAR
BIT      #TSOM, (R2)     ;DONE BIT IS SET AND SHOULD BE CLEARED
BEQ      5$                ;SW09=1?
HLT      10               ;SW09 SETUP
SCOPI   ;TEST TSOM
5$: BIC      #SEND, (R3)   ;BR IF CLEAR
PKCLK    ,16.             ;BIT FAILED TO CLR
BIT      #TXDONE, (R3)   ;SW09=1?

```

```

5746
5747
5748
5749
5750
5751
5752
5753
5754 027664 012737 000141 001226
5755 027672 012737 023200 001216
5756 022700 013703 001412
5757 022704 013702 001414
5758 022710 052777 000400 156474
5759 022716 004737 005044
5760 022722 005012
5761 022724 052713 014000
5762 022730 052777 001000 156452
5763 022736 052713 000020
5764 022742 005005
5765 022744 010512
5766 022746 012712 000400
5767 022752 104412 000006
5768 022756 052777 020000 156426
5769 022764 032713 001000
5770 022770 001001
5771 022772 104007
5772 022774
5773 022774 042777 020000 156410
5774 023002 104412 000015
5775 023006 032713 000020
5776 023012 001001
5777 023014 104017
5778 023016 012737 023016 001220
5779 023024 012705 000400
5780 023030 011204
5781 023032 042704 170000
5782 023036 020504
5783 023040 001406
5784 023042 013703 001414
5785 023046 104003
5786 023050 104401
5787 023052 012703 001412
5788 023056 012737 023056 001220
5789 023064 042712 000400
5790 023070 032713 000200
5791 023074 001402
5792 023076 104005
5793 023100 104401
5794 023102 012737 023102 001220
5795 023110 032712 000400
5796 023114 001402
5797 023116 104010
5798 023120 104401
5799 023122 042713 000020
5800 023126 104412 000020
5801 023132 032713 000200

```

5802	023136	001401			BEQ	+4		;BR IF OFF
5803	023140	104017			HLT	17		;DONE SET AND SHOULDN'T BE
5804	023142	104412	000002		PKCLK	2		;POKE ONE FULL CLOCK
5805	023146	032713	001000		BIT	#TXACT, (R3)		;CHECK ACTIVE
5806	023152	001401			BEQ	65		;BR IF OFF
5807	023154	104011			HLT	11		;ACTIVE SETS AND SHOULDN'T BE
5808	023156	032713	000200	65:	BIT	#TXDONE, (R3)		;IS DONE UP?
5809	023162	001001			BNE	75		;BR IF YES
5810	023164	104006			HLT	6		;NO-REPORT ERROR
5811	023166	032713	040000	75:	BIT	#MTDATA, (R3)		;CHECK DATA OUT
5812	023172	001401			BEQ	105		;BR IF OFF
5813	023174	104012			HLT	12		;DATA SET SHOULD BE CLEAR
5814	023176	104400		105:	SCOPE			;SCOPE THIS TEST

```

;***** TEST 142 *****
;THIS TEST VERIFIES THAT THE DEVICE IDLES FLAGS
;IDLE A MINIMUM OF 64. FLAGS.
;*****

```

```

;*****
;TEST 142
;*****

```

5827					ST142:	MOV	#142, #TSTNO	
5828	023200	012737	000142	001226		MOV	#TST143, NEXT	
5829	023206	012737	023270	001216		MOV	TXDBUF, R2	;LOAD TX BUFFER
5830	023214	013702	001414			MOV	TXCSR, R3	;LOAD TX CONTROL REGISTER
5831	023220	013703	001412			BIS	#MRESET, #TXCSR	;RESET THE DEVICE
5832	023224	052777	000400	156160		JSR	PC, SMALL	;WAIT FOR RESET TO FINISH
5833	023232	004737	005044			CLR	(R2)	;RESET TXDONE
5834	023236	005012				BIS	#MMODE, (R3)	;ENTER M/MODE
5835	023240	052713	014000			BIS	#SEND, (R3)	;SET SEND
5836	023244	052713	000020			BIS	#TSOM, (R2)	;TURN ON START OF MSG
5837	023250	052712	000400			PKCLK	4.	;SYNC UP DUP
5838	023254	104412	000004			JSR	R1, FLG	;SEND 64. FLAGS
5839	023260	004137	007052				64.	;64. FLAGS
5840	023264	000100				SCOPE		;SCOPE THIS TEST
5841	023266	104400						

```

;***** TEST 143 *****
;THIS TEST PUSHES DATA THRU THE TRANSMITTER
;IN MAINTENANCE MODE. THE TEST SENDS A FLAG,
;AND TWO ALTERNATING ONES AND ZEROES CHARACTERS,
;AN ALL ZEROES CHARACTER AND AN ALL ONES
;CHARACTER TO VERIFY THE BIT STUFF CAPABILITY OF
;THE DUP WITHOUT A CRC CHECK. THE TEST ROTATES
;THE BITS THRU, SAMPLING THE DATA ON A BIT-BY-BIT
;BASIS, LSB FIRST. IT STORES THE BIT IN THE MSB OF
;THE SAVE LOCATION, COMPARES AND ROTATES RIGHT UNTIL
;THE CHARACTER IS ASSEMBLED.
;*****

```

```

;*****
;

```

5842
5843
5844
5845
5846
5847
5848
5849
5850
5851
5852
5853
5854
5855
5856
5857

```

5858      : TEST 143
5859      :
5860      : *****
5861      : *****
5862      023270 012737 000143 001226 1ST143: MOV      #143, @TSTNO
5863      023276 012737 023644 001216      MOV      @TST144, NEXT
5864      023304 013703 001412      MOV      TXCSR, R3          ;LOAD TRANSMITTER CONTROL REGISTER
5865      023310 013702 001414      MOV      TXDBUF, R2        ;LOAD TRANSMITTER BUFFER
5866      023314 052777 000400 156070      BIS      @MRESET, @TXCSR  ;RESET THE DEVICE
5867      023322 004737 005044      JSR      PC, SMALL        ;WAIT FOR RESET TO FINISH
5868      023326 005012      CLR      (R2)             ;RESET TXDONE
5869      023330 052713 014000      BIS      @MMODE, (R3)     ;ENTER MINT MODE--PROGRAM CLOCKING
5870      023334 052777 001000 156046      BIS      @CRCEN, @PARCSR ;TURN OFF CRC
5871      023342 052713 000020      BIS      @SEND, (R3)     ;ASSERT SEND
5872      023346 052712 000400      BIS      @TSOM, (R2)
5873      023352 005037 001236      CLR      TEMP1           ;CLEAR
5874      023356 005037 001240      CLR      TEMP2           ;CLEAR
5875      023362 005037 001242      CLR      TEMP3           ;CLEAR
5876      023366 005037 001244      CLR      TEMP4           ;CLEAR
5877      023372 012704 000004      MOV      #4, R4          ;LOAD THE # OF CHARACTERS TO DO
5878      023376 012705 023632      MOV      @TBL1, R5        ;LOAD THE TABLE POINTER
5879      023402 012701 023632      MOV      @TBL1, R1        ;DITTO
5880      023406 011137 001246      MOV      (R1), TEMP5     ;DITTO
5881      023412 104412 000010      PKCLK   #,             ;START A FLAG
5882      023416 012512      MOV      (R5)+, (R2)     ;LOAD THE FIRST CHARACTER AND CLR TSOM
5883      023420 104412 000014      PKCLK   #,12.          ;FINISH THE FLAG
5884      023424 104412 000002 15:      PKCLK   #,2           ;PUSH OUT A BIT
5885      023430 000241      CLC                    ;PUT CARRY IN A KNOWN STATE
5886      023432 032713 040000      BIT      @MTDATA, (R3)  ;TEST THE BIT
5887      023436 001401      BEQ      2$            ;BR IF CLEAR
5888      023440 000261      SEC                    ;SET CARRY FOR SOFTWARE
5889      023442 106037 001236 25:      RORB    TEMP1          ;STORE THE BIT
5890      023446 000241      CLC                    ;PUT CARRY IN A KNOWN STATE
5891      023450 106037 001246      RORB    TEMP5          ;CHECK TO SEE WHAT THE BIT SHOULD BE
5892      023454 103006      BCC      3$            ;BR IF CLEAR
5893      023456 000261      SEC                    ;SET CARRY FOR SOFTWARE
5894      023460 106037 001242      RORB    TEMP3          ;SHIFT IN A ONE
5895      023464 005237 001240      INC     TEMP2          ;INC ONES COUNT
5896      023470 000404      BR      4$            ;JUMP OVER
5897      023472 106037 001242 35:      RORB    TEMP3          ;LOAD A ZERO
5898      023476 005037 001240      CLR     TEMP2          ;CLEAR ONES COUNT
5899      023502 023737 001236 45:      CMP     TEMP1, TEMP3   ;DOES HARDWARE = SOFTWARE
5900      023510 001401      BEQ     5$            ;BR IF YES
5901      023512 104004      HLT                    ;HARDWARE AND SOFTWARE DON'T MATCH
5902      :
5903      :
5904      :
5905      :
5906      :
5907      :
5908      023514 005704 55:      TST     R4             ;CHECK FOR LAST CHAR
5909      023516 001003      BNE     6$            ;BR IF NOT DONE
5910      023520 052712 001000      BIS     @TEOM, (R2)    ;LOAD END OF MESSAGE
5911      023524 000405      BR      7$            ;FINISH TEST
5912      023526 105777 155660 65:      TSTB   @TXCSR         ;CHECK TO SEE IF EREADY FOR NEXT CHAR
5913      023532 100002      BPL     7$            ;BR IF NO

```



```

5914 023534 012512          MOV      (R5)+, (R2)      ;LOAD NEXT CHAR
5915 023536 005304          DEC      R4              ;LOWER CHAR COUNT
5916 023540 022737 000005 001240 7$:  CMP      #5, TEMP2      ;CHECK FOR STUFFED ZERO
5917 023546 001006          BNE     10$             ;BR IF NO
5918 023550 104412 000002          PKCLK   2              ;PUSH OUT THE STUFFED ZERO
5919 023554 032713 040000          BIT     #MTDATA, (R3)  ;CHECK IT
5920 023560 001401          BEQ     10$             ;BR IF OK
5921 023562 104021          HLT     21             ;FAILED TO BIT-STUFF!!
5922 023564 005237 001244          INC     TEMP4           ;INC BIT COUNTER
5923 023570 022737 000010 001244 10$:  CMP      #8., TEMP4     ;ARE WE DONE WITH THIS CHAR?
5924 023576 001312          BNE     1$             ;BR IF MORE TO GO
5925 023600 005037 001236          CLR     TEMP1           ;CLEAR OUT HARDWARE SAVE
5926 023604 005037 001242          CLR     TEMP3           ;CLEAR OUT SOFTWARE SAVE
5927 023610 005037 001244          CLR     TEMP4           ;CLEAR OU BIT COUNTER
5928 023614 005721          TST     (R1)+          ;POP TBL POINTER
5929 023616 011137 001246          MOV     (R1), TEMP5    ;
5930 023622 032712 001000          BIT     #TEOM, (R2)   ;CHECK FOR END OF TEST
5931 023626 001676          BEQ     1$             ;BR IF MORE TO GO
5932 023630 104400          SCOPE                    ;SCOPE THIS TEST

```

```

TBL1: .WORD 252          ;THE FIRST FOUR CHARACTERS
      .WORD 000          ;OF THIS TABLE ARE OUTPUT.
      .WORD 125         ;THE LAST CHARACTER IS
      .WORD 377         ;A PAD.
      .WORD 000

```

```

;R1 = SOFTWARE TABLE POINTER
;R2 = TXDBUF
;R3 = TXCSR
;R4 = CHAR COUNTER
;R5 = HARDWARE TABLE POINTER
;TEMP1 = HARDWARE BIT
;TEMP2 = 1'S COUNT
;TEMP3 = SOFTWARE BIT
;TEMP4 = CHARACTER BIT COUNTER
;TEMP5 = SOFTWARE BYTE

```

```

;***** TEST 144 *****
;THIS TEST VERIFIES THE ABORT SEQUENCE AND
;NORMAL DATA SEQUENCE OF FLAG, DATA, FLAG
;FOLLOWED BY IDLE LINE. THIS TEST ALSO PROVES
;THE FUNCTIONING OF ACTIVE, TSOM, TEOM, SEND AND DONE.
;WITHOUT USING A CRC CHECK.
;*****

```

```

;*****
;TEST 144
;*****

```

```

5966 023644 012737 000144 001226 †ST144: MOV     #144, #TSTNO
5967 023652 012737 024132 001216      MOV     #TST145, NEXT
5968 023660 013703 001412          MOV     TXCSR, R3      ;LOAD CONTROL REGISTER
5969 023664 013702 001414          MOV     TXDBUF, R2     ;LOAD TRANSMITTER BUFFER

```

```

5970 023670 052777 000400 155514 BIS #MRESET, @TXCSR ; RESET THE DEVICE
5971 023676 004737 005044 JSR PC, SMALL ; WAIT FOR RESET TO FINISH
5972 023702 005012 CLR (R2) ; RESET TXDONE
5973 023704 052713 014000 BIS #MMODE, (R3) ; ENTER M/MODE
5974 023710 052777 001000 155472 BIS #CRCEN, @PARCSR ; SHUT OFF CRC
5975 023716 052713 000020 BIS #SEND, (R3) ; SET SEND
5976 023722 052712 000400 BIS #TSOM, (R2) ; TURN ON START OF MSG
5977 023726 104412 000010 PKCLK 8 ; SYNC UP DUP AND START PUSHING OUT A FLAG
5978 023732 052712 002000 BIS #TABORT, (R2) ; SET ABORT
5979 023736 104412 000014 PKCLK 12 ; PUSH OUT THE ABORT
5980 023742 005037 001236 CLR TEMP1 ; CLEAR HOLD
5981 023746 104412 000002 1$: PKCLK 2 ; PUSH A BIT
5982 023752 032713 040000 BIT #MTDATA, (R3) ; CHECK DATA
5983 023756 001001 BNE 2$ ; BR IF SET
5984 023760 104013 HLT 13 ; DATA FAILED TO SET
5985 023762 005237 001236 2$: INC TEMP1 ; INC HOLD
5986 023766 022737 000002 001236 CMP #2, TEMP1 ; CHECK FOR FINISH
5987 023774 001364 BNE 1$ ; BR IF NO
5988 023776 032713 001000 BIT #TXACT, (R3) ; TEST TRANSMITTER ACTIVE
5989 024002 001001 BNE 64$ ; BR IF ACTIVE SET
5990 024004 104007 HLT 7 ; ACTIVE IS RESET AND SHOULDN'T BE
5991 024006 64$:
5992 024006 042712 002000 BIC #TABORT, (R2) ; CLEAR ABORT
5993 024012 005037 001236 CLR TEMP1 ; CLEAR HOLD
5994 024016 104412 000002 3$: PKCLK, 2 ; PUSH A BIT
5995 024022 032713 040000 BIT #MTDATA, (R3) ; CHECK DATA
5996 024026 001001 BNE 4$ ; BR IF SET
5997 024030 104013 HLT 13 ; DATA OUT FAILED TO SET
5998 024032 005237 001236 4$: INC # TO DO
5999 024036 022737 000006 001236 CMP #6, TEMP1 ; AND CHECK IT
6000 024044 001364 BNE 3$ ; BR IF MORE TO GO
6001 024046 104412 000006 PKCLK 6 ; POKE CLOCK
6002 024052 012712 000252 MOV #252, (R2) ; CLEAR TSOM AND LOAD DATA
6003 024056 104412 000024 PKCLK 20 ; FINISH THE FLAG AND DATA
6004 024062 052712 001000 BIS #TEOM, (R2) ; SET TEOM
6005 024066 104412 000006 PKCLK 6 ; POKE CLOCK
6006 024072 004137 007052 JSR R1, FLG ; SEND THREE FLAGS
6007 024076 000003 3 ; DITTO
6008 024100 042713 000020 BIC #SEND, (R3) ; CLEAR SEND
6009 024104 104412 000004 PKCLK 4 ; POKE CLOCK
6010 024110 032713 040000 BIT #MTDATA, (R3) ; TEST TXDAT
6011 024114 001001 BNE 65$ ; MARK OUT
6012 024116 104013 HLT 13 ; TXDAT A SPACE - SHOULD BE 1
6013 024120 032713 000200 65$: BIT #TXDONE, (R3) ; IS DONE UP
6014 024124 001001 BNE 66$ ; YES
6015 024126 104006 HLT 6 ; NO - BUT IT SHOULD BE.
6016 024130 104400 66$: SCOPE ; SCOPE THIS TEST

```

6017
6018
6019
6020
6021
6022
6023
6024
6025

```

;***** TEST 145 *****
; *THIS TEST VERIFIES THAT A DATA
; *UNDER RUN CONDITION WILL CAUSE
; *THE TRANSMITTER DATA LATE BIT TO SET
; *AND THAT THE DEVICE WILL ABORT
; *UNTIL SEND IS CLEARED WHEN THE OUTPUT GOES TO A SPACE

```

```

6026
6027
6028
6029
6030
6031
6032
6033 024132 012737 000145 001226
6034 024140 012737 024334 001216
6035 024146 013703 001412
6036 024152 013702 001414
6037 024156 052777 000400 155226
6038 024164 004737 005044
6039 024170 005012
6040 024172 052713 014000
6041 024176 052777 001000 155204
6042 024204 052713 000020
6043 024210 052712 000400
6044 024214 104412 000006
6045 024220 012777 000000 155166
6046 024226 104412 000016
6047 024232 104412 000017
6048 024236 032777 100000 155146
6049 024244 001401
6050 024246 104014
6051 024250 104412 000002 155130
6052 024254 032777 100000
6053 024262 001001
6054 024264 104015
6055 024266
6056 024266 042777 000020 155116
6057 024274 104412 000020
6058 024300 012737 000010 001236
6059 024306 104412 000002 155072
6060 024312 032777 040000
6061 024320 001401
6062 024322 104012
6063 024324 005337 001236
6064 024330 001366
6065 024332 104400
6066
6067
6068
6069
6070
6071
6072
6073
6074
6075
6076
6077
6078
6079
6080
6081 024334 012737 000146 001226

```

```

:*****
:*****
: TEST 145
:*****
:*****
TST145: MOV #145,@TSTNO
MOV #TST146,NEXT
MOV TXCSR,R3 ;LOAD TRANSMITTER CONTROL REGISTER
MOV TXDBUF,R2 ;LOAD TRANSMITTER BUFFER
BIS #MRESET,@TXCSR ;RESET THE DEVICE
JSR PC,SMALL ;WAIT FOR RESET TO FINISH
CLR (R2) ;RESET TXDONE
BIS #MMODE,(R3) ;ENTER MINT MODE--PROGRAM CLOCKING
BIS #CRCEN,@PARCSR ;TURN OFF CRC
BIS #SEND,(R3) ;ASSERT SEND
BIS #TSOM,(R2)
PKCLK 6. ;START OUTPUTING FLAG
MOV #0,@TXDBUF ;CLEAR TSOM-LOAD BUFFER WITH ZEROES
PKCLK 14. ;FINISH FLAG
PKCLK 15. ;OUTPUT UP TO 7 1/2 CLOCK TICKS
BIT #TXDLAT,@TXCSR ;MAKE SURE DMA IS NOT SET
BEQ 15 ;BR IF CLEARED
HLT 14 ;BIT IS SET TOO SOON
PKCLK 2 ;FINISH LAST CLOCK
BIT #TXDLAT,@TXCSR ;NOW CHECK DMA
BNE 25 ;BRANCH IF SET
HLT 15 ;BIT IS CLEARED AND SHOULD BE SET
2$: BIC #SEND,@TXCSR ;TURN OFF SEND
PKCLK 16. ;PUSH 8 BITS
MOV #8.,TEMP1 ;SETUP FOR IDLE LINE
3$: PKCLK 2 ;OUTPUT BIT
BIT #MTDATA,@TXCSR ;CHECK IT
BEQ 45 ;BRANCH IF ZERO
HLT 12 ;BIT IS A 1
4$: DEC TEMP1 ;LOWER COUNT
BNE 35 ;NOT DONE? - GO BACK
SCOPE ;SCOPE THIS TEST

```

```

:***** TEST 146 *****
: *THIS TEST VERIFIES THAT DROPPING OF
: *SEND BEFORE SETTING TEOM CAUSES
: *A SPACE TO BE OUTPUT AFTER COMPLETION OF
: *A CHARACTER WITH AND WITHOUT BIT STUFF.
:*****

```

```

:*****
:*****
: TEST 146
:*****
:*****
TST146: MOV #146,@TSTNO

```

6082	024342	012737	024706	001216	MOV	#TST147, NEXT	
6083	024350	013703	001412		MOV	TXCSR, R3	:LOAD TRANSMITTER CONTROL REGISTER
6084	024354	013702	001414		MOV	TXDBUF, R2	:LOAD TRANSMITTER BUFFER
6085	024360	052777	000400	155024	BIS	#MRESET, @TXCSR	:RESET THE DEVICE
6086	024366	004737	005044		JSR	PC, SMALL	:WAIT FOR RESET TO FINISH
6087	024372	005012			CLR	(R2)	:RESET TXDONE
6088	024374	052713	014000		BIS	#MMODE, (R3)	:ENTER MINT MODE--PROGRAM CLOCKING
6089	024400	052777	001000	155002	BIS	#CRCEN, @PARCSR	:TURN OFF CRC
6090	024406	052713	000020		BIS	#SEND, (R3)	:ASSERT SEND
6091	024412	052712	000400		BIS	#TSOM, (R2)	
6092	024416	104412	000006		PKCLK	6.	:PUSH 2 BITS
6093	024422	012777	000252	154764	MOV	#252, @TXDBUF	:LOAD DATA
6094	024430	104412	000014		PKCLK	12.	:PUSH 6 BITS
6095	024434	042777	000020	154750	BIC	#SEND, @TXCSR	:TURN OFF TRANSMITTER
6096	024442	104412	000002		PKCLK	2	:POKE A FULL CLOCK
6097	024446	012737	000010	001236	MOV	#8., TEMP1	:LOAD TEMP1
6098	024454	104412	000012		PKCLK	10.	:PUSH 5 BITS
6099	024460	032777	040000	154724	BIT	#MTDATA, @TXCSR	:CHECK DATA
6100	024466	001401			BEQ	2\$:BR IF CLEAR
6101	024470	104012			HLT	12	:DATA IS SET - SHOULD BE CLEAR
6102	024472	005337	001236		DEC	TEMP1	:LOWER THE # OF TIMES TO REPEAT
6103	024476	001366			BNE	1\$:BR IF TO GOE MSG
6104	024500	032777	001000	154704	BIT	#TXACT, @TXCSR	:CHECK ACTIVE
6105	024506	001401			BEQ	7\$:BR IF OFF
6106	024510	104011			HLT	11	:ACTIVE FAILED TO CLEAR
6107	024512						
6108	024512	052777	000400	154672	BIS	#MRESET, @TXCSR	:RESET THE DEVICE
6109	024520	004737	005044		JSR	PC, SMALL	:WAIT FOR RESET TO FINISH
6110	024524	013703	001412		MOV	TXCSR, R3	:LOAD TRANSMITTER CONTROL REGISTER
6111	024530	013702	001414		MOV	TXDBUF, R2	:LOAD TRANSMITTER BUFFER
6112	024534	052777	000400	154650	BIS	#MRESET, @TXCSR	:RESET THE DEVICE
6113	024542	004737	005044		JSR	PC, SMALL	:WAIT FOR RESET TO FINISH
6114	024546	005012			CLR	(R2)	:RESET TXDONE
6115	024550	052713	014000		BIS	#MMODE, (R3)	:ENTER MINT MODE--PROGRAM CLOCKING
6116	024554	052777	001000	154626	BIS	#CRCEN, @PARCSR	:TURN OFF CRC
6117	024562	052713	000020		BIS	#SEND, (R3)	:ASSERT SEND
6118	024566	052712	000400		BIS	#TSOM, (R2)	
6119	024572	104412	000006		PKCLK	6.	:PUSH TWO BITS
6120	024576	012777	000177	154610	MOV	#177, @TXDBUF	:LOAD DATA
6121	024604	104412	000020		PKCLK	16.	:PUSH 8 BITS
6122	024610	042777	000020	154574	BIC	#SEND, @TXCSR	:TURN OFF TRANSMITTER
6123	024616	104412	000014		PKCLK	12.	:PUSH 6 BITS
6124	024622	032777	040000	154562	BIT	#MTDATA, @TXCSR	:CHECK DATA OUT FOR BIT STUFF FUNCTION
6125	024630	001001			BNE	3\$:BR IF SET
6126	024632	104013			HLT	13	:BIT IS A 0, SHOULD BE A 1 - DEVICE
6127							:FAILED TO BIT STUFF
6128	024634	104412	000004		PKCLK	4	:PUSH 2 BITS
6129	024640	012737	000010	001236	MOV	#8., TEMP1	:LOAD TEMP1
6130	024646	104412	000002		PKCLK	2	:PUSH A BIT
6131	024652	032777	040000	154532	BIT	#MTDATA, @TXCSR	:CHECK DATA
6132	024660	001401			BEQ	5\$:BR IF OFF
6133	024662	104012			HLT	12	:DATA WINDOW SET - SHOULD BE CLEAR
6134	024664	005337	001236		DEC	TEMP1	:LOWER THE # OF TIMES TO CHECK
6135	024670	001366			BNE	4\$:BR IF MORE TOGO
6136	024672	032777	001000	154512	BIT	#TXACT, @TXCSR	:CHECK ACTIVE
6137	024700	001401			BEQ	6\$:BR IF OFF

6138 024702 104011
6139 024704 104400
6140
6141
6142
6143
6144
6145
6146
6147
6148
6149
6150
6151
6152
6153
6154 024706 012737 000147 001226
6155 024714 012737 025102 001216
6156 024722 013703 001412
6157 024726 013702 001414
6158 024732 052777 000400 154452
6159 024740 004737 005044
6160 024744 005012
6161 024746 052713 014000
6162 024752 052777 001000 154430
6163 024760 052713 000020
6164 024764 052712 000400
6165 024770 104412 000006
6166 024774 112777 000176 154412
6167 025002 042777 000400 154404
6168 025010 104412 000016
6169 025014 104412 000017
6170 025020 032777 100000 154364
6171 025026 001401
6172 025030 104014
6173 025032 104412 000002 15:
6174 025036 032777 100000 154346
6175 025044 001401
6176 025046 104014
6177 025050 104412 000004 25:
6178 025054 032777 100000 154330
6179 025062 001001
6180 025064 104016
6181 025066 032777 040000 154316 35:
6182 025074 001001
6183 025076 104013
6184 025100 104400 45:
6185
6186
6187
6188
6189
6190
6191
6192
6193

```
65: HLT 11 ;ACTIVE FAILED TO CLEAR
SCOPE ;SCOPE THIS TEST

:***** TEST 147 *****
:*THIS TEST VERIFIES THAT A DATA UNDERRUN
:*CONDITION WILL CAUSE THE TRANSMITTER DATA
:*LATE BIT TO SET AND THAT THE DUP WILL GO TO
:*A MARK OUTPUT
:*****

:*****
: TEST 147
:*****
:*****
:*****
ST147: MOV #147,@TSTNO
MOV #TST150,NEXT
MOV TXCSR,R3 ;LOAD TRANSMITTER CONTROL REGISTER
MOV TXDBUF,R2 ;LOAD TRANSMITTER BUFFER
BIS #MRESET,@TXCSR ;RESET THE DEVICE
JSR PC,SMALL ;WAIT FOR RESET TO FINISH
CLR (R2) ;RESET TXDONE
BIS #MMODE,(R3) ;ENTER MINT MODE--PROGRAM CLOCKING
BIS #CRCEN,@PARCSR ;TURN OFF CRC
BIS #SEND,(R3) ;ASSERT SEND
BIS #TSOM,(R2)
PKCLK ,6. ;START FLAG
MOV#B #176,@TXDBUF ;LOAD CHARACTER TO BE BIT-STUFFED
BIC #TSOM,@TXDBUF ;SHUT OFF TSOM
PKCLK ,14. ;FINISH CLOCKING FLAG
PKCLK ,15. ;CLOCK TO WITHIN 1 1/2 CLOCKS
BIT #TXDLAT,@TXCSR ;CHECK DATA LATE
BEQ 15 ;BRANCH IF CLEAR
HLT 14 ;BIT IS SET TOO SOON
PKCLK ,2 ;CLOCK TO WITHIN A HALF-CLOCK OF DNA
BIT #TXDLAT,@TXCSR ;CHECK DNA
BEQ 25 ;BR IF OFF
HLT 14 ;BIT SET TOO SOON, DEVICE FAILED TO BIT-STUFF
PKCLK ,4 ;FINISH CHARACTER
BIT #TXDLAT,@TXCSR ;CHECK DNA
BNE 35 ;BRANCH IF SET
HLT 16 ;BIT SHOULD BE SET AND IS CLEARED
BIT #MTDATA,@TXCSR ;CHECK DATA
BNE 45 ;BR IF SET
HLT 13 ;DATA WAS CLEAR - SHOULD BE SET
45: SCOPE ;SCOPE THIS TEST

:***** TEST 150 *****
:*THIS TEST VERIFIES THAT SETTING TEOM
:*AND TSOM AT THE SAME TIME WILL HOLD
:*ACTIVE UP AND SEND A FLAG
:*****

:*****
```

```

6194
6195
6196
6197
6198
6199 025102 012737 000150 001226
6200 025110 012737 025236 001216
6201 025116 013703 001412
6202 025122 013702 001414
6203 025126 052777 000400 154256
6204 025134 004737 005044
6205 025140 005012
6206 025142 052713 014000
6207 025146 052777 001000 154234
6208 025154 052713 000020
6209 025160 052712 000400
6210 025164 104412 000006
6211 025170 012777 000252 154216
6212 025176 104412 000032
6213 025202 052777 001400 154204
6214 025210 104412 000004
6215 025214 032777 001000 154170
6216 025222 001001
6217 025224 104007
6218 025226 004137 007052
6219 025232 000001
6220 025234 104400
6221
6222
6223
6224
6225
6226
6227
6228
6229
6230
6231
6232
6233
6234 025236 012737 000151 001226
6235 025244 012737 026306 001216
6236 025252 012737 102010 026300
6237 025260 012737 026116 001244
6238 025266 012737 025274 001220
6239 025274
6240 025274 052777 000400 154110
6241 025302 004737 005044
6242 025306 005077 154102
6243 025312 052777 014000 154072
6244 025320 052777 000020 154064
6245 025326 052777 000400 154060
6246 025334 104412 000006
6247 025340 017777 153700 154046
6248 025346 104412 000020
6249 025352 005037 026304

```

```

: TEST 150 *
:*****
:*****
TST150: MOV #150,@TSTNO
MOV #TST151,NEXT
MOV TXCSR,R3 ;LOAD TRANSMITTER CONTROL REGISTER
MOV TXDBUF,R2 ;LOAD TRANSMITTER BUFFER
BIS #MRESET,@TXCSR ;RESET THE DEVICE
JSR PC,SMALL ;WAIT FOR RESET TO FINISH
CLR (R2) ;RESET TXDONE
BIS #MMODE,(R3) ;ENTER MINT MODE--PROGRAM CLOCKING
BIS #CRCEN,@PARCSR ;TURN OFF CRC
BIS #SEND,(R3) ;ASSERT SEND
BIS #TSOM,(R2)
PKCLK 6 ;SYNC UP DUP
MOV #252,@TXDBUF ;LOAD DATA
PKCLK 26 ;FINISH THE FLAG AND DATA
BIS #TEOM:@TSOM,@TXDBUF ;TURN ON START AND END OF MSG
PKCLK 4 ;PUSH 2 BITS
BIT #TXACT,@TXCSR ;TEST ACTIVE
BNE 15 ;BR IF SET
HLT 7 ;ACTIVE SHOULD BE SET AND IS CLEAR
15: JSR R1,FLG ;PUSH OUT A FLAG
1 ;ONE FLAG
SCOPE ;SCOPE THIS TEST

```

```

:***** TEST 151 *****
:*TEST OF THE BCC OPERATION USING
:*CRC.CCITT FOR THE POLYNOMIAL. SPECIFIC
:*DATA PATTERNS ARE USED TO ISOLATE FAULTS.
:*****

```

```

:*****
: TEST 151 *
:*****
:*****
TST151: MOV #151,@TSTNO
MOV #TST152,NEXT
MOV #CRC.CCITT,XPOLY ;SET THE POLYNOMIAL
MOV #55,TEMP4 ;GET THE TABLE POINTER
MOV #15,LOCK ;SETUP FOR SW09=1
15: BIS #MRESET,@TXCSR ;RESET THE DEVICE
JSR PC,SMALL ;WAIT FOR RESET TO FINISH
CLR @TXDBUF ;RESET TXDONE
BIS #MMODE,@TXCSR ;ENTER M/MODE
BIS #SEND,@TXCSR ;TURN ON DUP
BIS #TSOM,@TXDBUF ;TURN ON START OF MSG
PKCLK 6 ;SYNC UP DUP
MOV @TEMP4,@TXDBUF ;LOAD DATA
PKCLK 16 ;PUSH 8 BITS
CLR CALBCC ;CLEAR OUT OLD BCC

```

6250	025356	004537	026126		JSR	R5, SIMBCC	: CALCULATE A SOFTWARE BCC
6251	025362	000010			8.		: BASED
6252	025364	000000			0.		: ON THESE
6253	025366	177777			-1		: PARAMETERS
6254	025370	012777	001000	154016	MOV	#TEOM, @TXDBUF	: CLEAR TSOM, SET TEOM
6255	025376	104412	000016		PKCLK	14.	: PUSHOUT DATA
6256	025402	004137	006750		JSR	R1, ACC	: DO A BCC CALCULATION (HDWR)
6257	025406	100000			BIT15		: FOR 16 BITS
6258	025410	013705	026304		MOV	CALBCC, R5	: MOV SOFTWARE BCC TO EXPECTED
6259	025414	005105			COM	R5	: INVERT IT
6260	025416	020504			CMP	R5, R4	: DOES EXPECTED=FOUND?
6261	025420	001401			BEQ	64\$: BR IF OK
6262	025422	104020			HLT	20	: BCC CALCULATION ERROR
6263							: TO DEBUG CRC USE THE CRC
6264							: DEBUG AID TEST. SEE FRONT OF THE
6265							: LISTING FOR TEST LOCATION
6266	025424	104401			64\$:	SCOPI	: SW09=1?
6267	025426	062737	000002	001244	ADD	#2, TEMP4	: MOVE THE TABLE POINTER
6268	025434	012737	025442	001220	MOV	#2\$, LOCK	: SETUP FOR SW09=1
6269	025442				2\$:		
6270	025442	052777	000400	153742	BIS	#MRESET, @TXCSR	: RESET THE DEVICE
6271	025450	004737	005044		JSR	PC, SMALL	: WAIT FOR RESET TO FINISH
6272	025454	005077	153734		CLR	@TXDBUF	: RESET TXDONE
6273	025460	052777	014000	153724	BIS	#MMODE, @TXCSR	: ENTER M/MODE
6274	025466	052777	000020	153716	BIS	#SEND, @TXCSR	: TURN ON DUP
6275	025474	052777	000400	153712	BIS	#TSOM, @TXDBUF	: TURN ON START OF MSG
6276	025502	104412	000006		PKCLK	6.	: SYNC UP DUP
6277	025506	017777	153532	153700	MOV	@TEMP4, @TXDBUF	: LOAD DATA
6278	025514	104412	000020		PKCLK	16.	: PUSH 8 BITS
6279	025520	005037	026304		CLR	CALBCC	: CLEAR OUT OLD BCC
6280	025524	004537	026126		JSR	R5, SIMBCC	: CALCULATE A SOFTWARE BCC
6281	025530	000010			8.		: BASED
6282	025532	000252			↑B<10101010>		: ON THESE
6283	025534	177777			-1		: PARAMETERS
6284	025536	012777	001000	153650	MOV	#TEOM, @TXDBUF	: CLEAR TSOM, SET TEOM
6285	025544	104412	000016		PKCLK	14.	: PUSHOUT DATA
6286	025550	004137	006750		JSR	R1, ACC	: DO A BCC CALCULATION (HDWR)
6287	025554	100000			BIT15		: FOR 16 BITS
6288	025556	013705	026304		MOV	CALBCC, R5	: MOV SOFTWARE BCC TO EXPECTED
6289	025562	005105			COM	R5	: INVERT IT
6290	025564	020504			CMP	R5, R4	: DOES EXPECTED=FOUND?
6291	025566	001401			BEQ	65\$: BR IF OK
6292	025570	104020			HLT	20	: BCC CALCULATION ERROR
6293							: TO DEBUG CRC USE THE CRC
6294							: DEBUG AID TEST. SEE FRONT OF THE
6295							: LISTING FOR TEST LOCATION
6296	025572	104401			65\$:	SCOPI	: SW09=1?
6297	025574	062737	000002	001244	ADD	#2, TEMP4	: MOVE THE TABLE POINTER
6298	025602	012737	025610	001220	MOV	#3\$, LOCK	: SETUP FOR SW09=1
6299	025610				3\$:		
6300	025610	052777	000400	153574	BIS	#MRESET, @TXCSR	: RESET THE DEVICE
6301	025616	004737	005044		JSR	PC, SMALL	: WAIT FOR RESET TO FINISH
6302	025622	005077	153566		CLR	@TXDBUF	: RESET TXDONE
6303	025626	052777	014000	153556	BIS	#MMODE, @TXCSR	: ENTER M/MODE
6304	025634	052777	000020	153550	BIS	#SEND, @TXCSR	: TURN ON DUP
6305	025642	052777	000400	153544	BIS	#TSOM, @TXDBUF	: TURN ON START OF MSG

6306	025650	104412	000006		PKCLK	6.	: SYNC UP DUP
6307	025654	017777	153364	153532	MOV	#TEMP4, @TXDBUF	: LOAD DATA
6308	025662	104412	000020		PKCLK	16.	: PUSH 8 BITS
6309	025666	005037	026304		CLR	CALBCC	: CLEAR OUT OLD BCC
6310	025672	004537	026126		JSR	R5, SIMBCC	: CALCULATE A SOFTWARE BCC
6311	025676	000010			B.		: BASED
6312	025700	000125			↑B<01010101>		: ON THESE
6313	025702	177777			-1		: PARAMETERS
6314	025704	012777	001000	153502	MOV	#TEOM, @TXDBUF	: CLEAR TSOM, SET TEOM
6315	025712	104412	000016		PKCLK	14.	: PUSHOUT DATA
6316	025716	004137	006750		JSR	R1, ACC	: DO A BCC CALCULATION (HDWR)
6317	025722	100000			BIT15		: FOR 16 BITS
6318	025724	013705	026304		MOV	CALBCC, R5	: MOV SOFTWARE BCC TO EXPECTED
6319	025730	005105			COM	R5	: INVERT IT
6320	025732	020504			CMP	R5, R4	: DOES EXPECTED=FOUND?
6321	025734	001401			BEQ	66\$: BR IF OK
6322	025736	104020			HLT	20	: BCC CALCULATION ERROR
6323							: TO DEBUG CRC USE THE CRC
6324							: DEBUG AID TEST. SEE FRONT OF THE
6325							: LISTING FOR TEST LOCATION
6326	025740	104401			66\$: SCOP1		: SW09=1?
6327	025742	062737	000002	001244	ADD	#2, TEMP4	: MOVE THE TABLE POINTER
6328	025750	012737	025756	001220	MOV	#4\$, LOCK	: SETUP FOR SW09=1
6329	025756				4\$:		
6330	025756	052777	000400	153426	BIS	#MRESET, @TXCSR	: RESET THE DEVICE
6331	025764	004737	005044		JSR	PC, SMALL	: WAIT FOR RESET TO FINISH
6332	025770	005077	153420		CLR	@TXDBUF	: RESET TXDONE
6333	025774	052777	014000	153410	BIS	#MMODE, @TXCSR	: ENTER M/MODE
6334	026002	052777	000020	153402	BIS	#SEND, @TXCSR	: TURN ON DUP
6335	026010	052777	000400	153376	BIS	#TSOM, @TXDBUF	: TURN ON START OF MSG
6336	026016	104412	000006		PKCLK	6.	: SYNC UP DUP
6337	026022	017777	153216	153364	MOV	#TEMP4, @TXDBUF	: LOAD DATA
6338	026030	104412	000020		PKCLK	16.	: PUSH 8 BITS
6339	026034	005037	026304		CLR	CALBCC	: CLEAR OUT OLD BCC
6340	026040	004537	026126		JSR	R5, SIMBCC	: CALCULATE A SOFTWARE BCC
6341	026044	000010			B.		: BASED
6342	026046	000377			↑B<11111111>		: ON THESE
6343	026050	177777			-1		: PARAMETERS
6344	026052	012777	001000	153334	MOV	#TEOM, @TXDBUF	: CLEAR TSOM, SET TEOM
6345	026060	104412	000020		PKCLK	16.	: PUSH 8 BITS
6346	026064	004137	006750		JSR	R1, ACC	: DO A BCC CALCULATION (HDWR)
6347	026070	100000			BIT15		: FOR 16 BITS
6348	026072	012705	157400		MOV	#157400, R5	: LOAD THE BCC
6349	026076	020504			CMP	R5, R4	: DOES EXPECTED=FOUND?
6350	026100	001401			BEQ	67\$: BR IF OK
6351	026102	104020			HLT	20	: BCC CALCULATION ERROR
6352							: TO DEBUG CRC USE THE CRC
6353							: DEBUG AID TEST. SEE FRONT OF THE
6354							: LISTING FOR TEST LOCATION
6355	026104	104401			67\$: SCOP1		: SW09=1?
6356	026106	062737	000002	001244	ADD	#2, TEMP4	: MOVE THE TABLE POINTER
6357	026114	104400			SCOPE		: SCOPE THIS TEST
6358	026116	000000			5\$:		
6359	026120	000252			.WORD	0	
6360	026122	000125			.WORD	252	
6361	026124	000377			.WORD	125	
					.WORD	377	


```

6362 026126 010046
6363 026130 010146
6364 026132 010246
6365 026134 012537 001236
6366 026140 012537 001240
6367 026144 012537 001242
6368 026150 005037 026302
6369 026154 013700 001242
6370 026160 006037 001240
6371 026164 005500
6372 026166 032700 000001
6373 026172 001402
6374 026174 005137 026302
6375 026200 013700 026300
6376 026204 005100
6377 026206 040037 026302
6378 026212 000241
6379 026214 006037 001242
6380 026220 013700 026302
6381 026224 013701 001242
6382 026230 010102
6383 026232 040100
6384 026234 043702 026302
6385 026240 050200
6386 026242 043737 026300 001242
6387 026250 050037 001242
6388 026254 005337 001236
6389 026260 001333
6390 026262 013737 001242 026304
6391 026270 012602
6392 026272 012601
6393 026274 012600
6394 026276 000205
6395 026300 000000
6396 026302 000000
6397 026304 000000
6398 120001
6399 102010
6400
6401
6402
6403
6404
6405
6406
6407
6408
6409
6410
6411
6412
6413
6414
6415
6416
6417

```

```

SIMBCC: MOV R0,-(SP)
MOV R1,-(SP)
MOV R2,-(SP)
MOV (R5)+,TEMP1
MOV (R5)+,TEMP2
MOV (R5)+,TEMP3
1$: CLR BCCFBK
MOV TEMP3,R0
ROR TEMP2
ADC R0
BIT #BIT0,R0
BEQ 2$
COM BCCFBK
2$: MOV XPOLY,R0
COM R0
BIC R0,BCCFBK
CLC
ROR TEMP3
MOV BCCFBK,R0
MOV TEMP3,R1
MOV R1,R2
BIC R1,R0
BIC BCCFBK,R2
BIS R2,R0
BIC XPOLY,TEMP3
BIS R0,TEMP3
DEC TEMP1
1$: MOV TEMP3,CALBCC
MOV (SP)+,R2
MOV (SP)+,R1
MOV (SP)+,R0
RTS R5

```

```

XPOLY: 0
BCCFBK: 0
CALBCC: 0
CRC16=120001
CRC.CCITT=102010

```

```

:***** TEST 152 *****
:*THIS TEST IS AN AID FOR DEBUGGING CRC
:*ERRORS. A CHARACTER IS LOADED INTO THE
:*DUP AND PUSHED OUT BIT BY BIT WHILE
:*ALLOWING THE OPERATOR TO MONITOR THE CRC
:*CHARACTER AS IT IS GENERATED. THE DATA CHARACTER
:*CAN ALSO BE CHANGED BY THE OPERATOR.
:*PUT SW09=1 TO LOCK ON BITS. TO CONTINUE HIT
:*ANY KEY ON THE TTY. AFTER 16 TIMES PUT DOWN SW09 TO LEAVE
:*NOTE: REMEMBER--IN SDLC A ONE IS A LOGIC LOW IN
:*THE CRC GENERATOR.
:*****

```

```

:*****
:
: TEST 152
:

```

```

6418                                     ;:*****
6419                                     ;:*****
6420 026306 012737 000152 001226 †ST152: MOV #152,@TSTNO
6421 026314 012737 002764 001216 MOV #.EOP,NEXT
6422 026322 052777 000400 153062 BIS #MRESET,@TXCSR ;RESET THE DEVICE
6423 026330 004737 005044 JSR PC,SMALL ;WAIT FOR RESET TO FINISH
6424 026334 012737 102010 026300 MOV #CRC.CCITT,XPOLY ;LOAD THE POLYNOMIAL
6425 026342 012737 000125 026510 MOV #125,3$ ;LOAD DATA TO SOFTWARE BCC-CHANGE CHARACTER HERE
6426 026350 013737 026510 001252 MOV 3$,SAVR1
6427 026356 012737 177777 026304 MOV #-1,CALBCC ;CLEAR FOR SOFTWARE BCC
6428 026364 013737 026304 026512 MOV CALBCC,4$
6429 026372 005037 001242 CLR TEMP3 ;CLR SOFTWARE MEMORY
6430 026376 005037 001244 CLR TEMP4 ;CLEAR BIT COUNTER
6431 026402 005037 001246 CLR TEMP5
6432 026406 005077 153002 CLR @TXDBUF ;RESET TXDONE
6433 026412 052777 014000 152772 BIS #MMODE,@TXCSR ;ENTER MAINT MODE-PROGRAM CLOCKING
6434 026420 052777 000020 152756 BIS #RCVEN,@RXCSR ;TURN ON RECEIVER
6435 026426 052777 000020 152756 BIS #SEND,@TXCSR ;TURN ON TRANSMITTER
6436 026434 012777 000400 152752 MOV #TSOM,@TXDBUF ;
6437 026442 104412 000044 PKCLK 36. ;PUSH OUT 2
6438 026446 013777 026510 152740 MOV 3$,@TXDBUF ;LOAD DATA
6439 026454 104412 000020 PKCLK ,16. ;PUSH OUT ANOTHER
6440 026460 104412 000002 PKCLK 2 ;PUSH OUT A BIT
6441 026464 013737 001244 001254 1$: MOV TEMP4,SAVR2 ;SET UP TO TYPE
6442 026472 005237 001242 INC TEMP3 ;UPDATE THE COUNT
6443 026476 005237 001244 INC TEMP4 ;UPDATE BIT COUNTER
6444 026502 004537 026126 2$: JSR R5,SIMBCC ;CALCULATE SOFTWARE BCC BASED ON THESE PARAMETERS
6445 026506 000001 1 ;SHIFTS
6446 026510 000000 3$: .WORD 0 ;DATA
6447 026512 000000 4$: .WORD 0 ;PREVIOUS BCC
6448 026514 004737 026612 JSR PC,5$ ;CHECK TO SEE IF WE SHOULD WAIT FOR SCOPING
6449 026520 000241 CLC ;CLEAR FOR NEXT ROTATE
6450 026522 106037 026510 RORB 3$ ;SET UP THE NEXT BIT
6451 026526 013737 026304 026512 MOV CALBCC,4$ ;FOR THE SOFTWARE BCC
6452 026534 022737 000006 001244 CMP #6,TEMP4
6453 026542 001002 BNE .+6
6454 026544 005077 152644 CLR @TXDBUF
6455 026550 022737 000014 001242 CMP #12.,TEMP3
6456 026556 001003 BNE 12$
6457 026560 012777 001000 152626 MOV #TEOM,@TXDBUF
6458 026566 022737 000020 001244 12$: CMP #16.,TEMP4 ;ALL DONE WITH THE CHARACTER?
6459 ;INCREASE THE COMPARE NUMBER TO
6460 ;ALLOW CRC TO BE OUTPUT
6461 026574 001331 BNE 1$ ;BR IF MORE TO GO
6462 026576 052777 000400 152606 BIS #MRESET,@TXCSR ;RESET THE DEVICE
6463 026604 004737 005044 JSR PC,SMALL ;WAIT FOR RESET TO FINISH
6464 026610 104400 SCOPE ;SCOPE THIS TEST
6465
6466 026612 032777 001000 152362 5$: BIT #SW09,@SWR ;SW09=1?
6467 026620 001432 BEQ 6$ ;BR IF NO
6468 026622 013704 026304 MOV CALBCC,R4 ;THE DATA CHARACTER IS
6469 026626 012737 000001 001256 MOV #1,SAVR3 ;FOLLOWED BY A ZERO CHARACTER.THE
6470 026634 000241 CLC ;DATA BIT IN CRC SHOWS WHICH BIT
6471 026636 006004 11$: ROR R4 ;OF THE TWO CHARACTERS IS BEING
6472 026640 006137 001256 ROL SAVR3 ;GENERATED
6473 026644 103374 BCC 11$
    
```

DZDPBB MACY11 27(1006) 18-MAY-77 00:00 PAGE 130
DZDPBB.P11 13-MAY-77 15:27 CRC DEBUGGING AID TEST

6474	026646	105737	001246
6475	026652	001006	
6476	026654	104402	027407
6477	026660	104402	027436
6478	026664	105137	001246
6479	026670	104410	
6480	026672	030100	
6481	026674	105777	152304
6482	026700	100375	
6483	026702	017701	152300
6484	026706	000207	
6485			
6486			

	TSTB	TEMPS
	BNE	10\$
	TYPE	,EM1
	TYPE	MHI
	COMB	↑TEMPS
10\$:	CONVRT	
	DT1	
7\$:	TSTB	@TKCSR
	BPL	7\$
	MOV	@TKDBR,R1
6\$:	RTS	PC

;TYPE MSG
;TYPE HEADER

;CHECK TTY DONE--GO SCOPE THE CRC GENERATOR
;BR IF NOT YET
;READ THE BUFFER
;RETURN

6487

(1)	026710	005015	051124	047101	EM2:	.ASCIZ	<15><12>/TRANSMITTER /	
(1)	026730	005015	042522	042503	EM3:	.ASCIZ	<15><12>/RECEIVER /	
(1)	026745	015	052412	042516	EM4:	.ASCIZ	<15><12>/UNEXPECTED INTERRUPT/	
(1)	026774	005015	051120	046511	EM6:	.ASCIZ	<15><12>/PRIMARY REGISTER TEST †	
(1)	027025	377	051124	047101	EM10:	.ASCIZ	<377>/TRANSMITTER DATA ERROR. /	
(1)	027057	377	044103	041505	EM11:	.ASCIZ	<377>/CHECK CARRY AND OUTBIT. /	
(1)	027113	104	047117	020105	EM12:	.ASCIZ	/DONE BIT /	
(1)	027126	040506	046111	042105	EM13:	.ASCIZ	/FAILED TO CLEAR /	
(1)	027147	106	044501	042514	EM14:	.ASCIZ	/FAILED TO SET /	
(1)	027166	041501	044524	042526	EM15:	.ASCIZ	/ACTIVE BIT /	
(1)	027203	124	047523	020115	EM16:	.ASCIZ	/TSON BIT /	
(1)	027216	052517	050124	052125	EM17:	.ASCIZ	/OUTPUT DATA /	
(1)	027234	042523	020124	047524	EM22:	.ASCIZ	/SET TOO SOON /	
(1)	027252	052377	042130	040514	EM23:	.ASCIZ	<377>/TXDLAT BIT /	
(1)	027270	041777	041522	041440	EM24:	.ASCIZ	<377>/CRC CALCULATION ERROR /	
(1)	027320	047111	042524	052522	DH2:	.ASCIZ	/INTERRUPTED UNEXPECTEDLY./	
(1)	027351	377	054105	042520	DH6:	.ASCIZ	<377>/EXPECTED FOUND PRI REG /	
(1)	027407	377	051103	020103	EM1:	.ASCIZ	<377>/CRC GENERATOR STATUS /	
(1)	027436	042377	052101	020101	MH1:	.ASCIZ	<377>/DATA CHAR DATA BIT IN CRC GEN. CRC FOR THIS BIT /	
(1)	027531	106	044501	042514	DH7:	.ASCIZ	/FAILED TO BIT-STUFF. /	
(1)	027557	015	050012	044522	EM5:	.ASCIZ	<15><12>/PRIMARY REGISTER ADDRESSING TIME-OUT /	
(1)	027627	015	050012	044522	EM7:	.ASCIZ	<15><12>/PRIMARY REGISTER RESET TEST /	
(1)	027667	377	042522	044507	DH5:	.ASCIZ	<377>/REGISTER REFERENCED TRAPPED FROM /	
(1)					.EVEN			
(1)	027732				.ERRTAB:			
(1)	027732	000000				0		
(1)						0		
(1)	027734	000000				0		
(1)	027736	000000				0		
(1)	027740	027557				EM5		
(1)	027742	027667				DH5	;HALT 1	
(1)	027744	030116				DT5		
(1)								
(1)	027746	027627				EM7		
(1)	027750	027351				DH6	;HALT 2	
(1)	027752	030130				DT6		
(1)								
(1)	027754	026774				EM6		
(1)	027756	027351				DH6	;HALT 3	
(1)	027760	030130				DT6		
(1)								
(1)	027762	027025				EM10		
(1)	027764	000000				0	;HALT 4	
(1)	027766	000000				0		
(1)								
(1)	027770	027113				EM12		
(1)	027772	027126				EM13	;HALT 5	
(1)	027774	000000				0		
(1)								
(1)	027776	027113				EM12		
(1)	030000	027147				EM14	;HALT 6	
(1)	030002	000000				0		
(1)								

DZDPBB MACY11 27(1006) 18-MAY-77 00:00 PAGE 132
DZDPBB.P11 13-MAY-77 15:27 CRC DEBUGGING AID TEST

(1)	030004	027166		EM15	
(1)	030006	027147		EM14	;HALT 7
(1)	030010	000000		0	
(1)					
(1)	030012	027203		EM16	
(1)	030014	027126		EM13	;HALT 10
(1)	030016	000000		0	
(1)					
(1)	030020	027166		EM15	
(1)	030022	027126		EM13	;HALT 11
(1)	030024	000000		0	
(1)					
(1)	030026	027216		EM17	
(1)	030030	027126		EM13	;HALT 12
(1)	030032	000000		0	
(1)					
(1)	030034	027216		EM17	
(1)	030036	027147		EM14	;HALT 13
(1)	030040	000000		0	
(1)					
(1)	030042	027252		EM23	
(1)	030044	027234		EM22	;HALT 14
(1)	030046	000000		0	
(1)					
(1)	030050	027252		EM23	
(1)	030052	027147		EM14	;HALT 15
(1)	030054	000000		0	
(1)					
(1)	030056	027252		EM23	
(1)	030060	027126		EM13	;HALT 16
(1)	030062	000000		0	
(1)					
(1)	030064	027113		EM12	
(1)	030066	027234		EM22	;HALT 17
(1)	030070	000000		0	
(1)					
(1)	030072	027270		EM24	
(1)	030074	027351		DH6	;HALT 20
(1)	030076	030130		DT6	
(1)					
(1)	030100	000003			
(1)	030102	006	021	DT1: 3	.BYTE 6,17.
(1)	030104	001252			SAVR1
(1)	030106	006	017		.BYTE 6,15.
(1)	030110	001254			SAVR2
(1)	030112	006	002		.BYTE 6,2
(1)	030114	001256			SAVR3
(1)	030116	000002		DT5: 2	
(1)	030120	006	017		.BYTE 6,15.
(1)	030122	001252			SAVR1
(1)	030124	006	002		.BYTE 6,2
(1)	030126	001254			SAVR2
(1)	030130	000003		DT6: 3	
(1)	030132	006	004		.BYTE 6,4
(1)	030134	001262			SAVR5
(1)	030136	006	002		.BYTE 6,2

DZDPBB MACY11 27(1006) 18-MAY-77 00:00 PAGE 133
DZDPBB.P11 13-MAY-77 15:27 CRC DEBUGGING AID TEST

(1)	030140	001260			SAVR4	
(1)	030142	006	002		.BYTE	6,2
(1)	030144	001256			SAVR3	
(1)						
(1)						
(1)						
(1)						
(1)						
(1)						
(1)	030146					
(1)		000001				

CORMAX:
.END

ABRT	007000	2352#												
ACC	006750	2343#	6256	6286	6316	6346								
ADRCNT=	003735	1932*	1968*	1977#										
BCCFBK	026302	6368*	6374*	6377*	6380	6384	6396#							
BINWRD	004240	2019*	2020	2057#										
BITW =	002000	1409#												
BIT0 =	000001	1271#	1391	2802	3484	3488	3492	3494	3892	3899	3904	4252	4259	4264
		6372												
BIT1 =	000002	1270#	1390	2832	3132	3136	3140	3142	3516	3520	3524	3526	4292	4299
		4304	5481											
BIT10 =	002000	1261#	1381	1396	1409	1429	2674	3102	3356	3360	3364	3366	3804	3808
		3812	3814	4012	4019	4024	4652	4659	4664					
BIT11 =	004000	1260#	1380	1408	1428	2706	3388	3392	3396	3398	4052	4059	4064	
EIT12 =	010000	1259#	1379	1395	1402	1407	1427	2738	3420	3424	3428	3430	4092	4099
		4104	4692	4699	4704	4977	4984	4989	5188	5191	5200	5205		
BIT13 =	020000	1258#	1378	1406	1426	2770	3452	3456	3460	3462	4132	4139	4144	5017
		5024	5029											
BIT14 =	040000	1257#	1377	1394	1405	1425	1691	1839	4172	4179	4184	4732	4739	4744
		4893	4901	4906	5057	5064	5069							
BIT15 =	100000	1256#	1376	1393	1400	1404	1424	1688	4212	4219	4224	4772	4779	4784
		4935	4943	4948	5097	5104	5109	5236	5239	5248	5253	6257	6287	6317
		6347												
BIT2 =	000004	1269#	1389	1777	2862	3164	3168	3172	3174	3548	3552	3556	3558	4332
		4339	4344	5513										
BIT3 =	000010	1268#	1388	1415	2610	2892	3196	3200	3204	3206	3292	3296	3300	3302
		3580	3584	3588	3590	4372	4379	4384	5545					
BIT4 =	000020	1267#	1387	1414	2642	2922	3228	3232	3236	3238	3324	3328	3332	3334
		3612	3616	3620	3622	4412	4419	4424	5419					
BIT5 =	000040	1266#	1386	2952	3644	3648	3652	3654	4452	4459	4464	5389	5609	5613
		5617	5619											
BIT6 =	000100	1265#	1385	1413	2982	3676	3680	3684	3686	4492	4499	4504	5359	5449
		5577	5581	5585	5587	5641	5645	5649	5651					
BIT7 =	000200	1264#	1384	1412	2439	2440	2456	2563	2613	2619	2645	2651	2677	2683
		2709	2715	2741	2747	2773	2779	3012	3708	3712	3716	3718	3932	3939
		3944	4532	4539	4544	4813	4825	4855	4871	4897	4913	4939	4955	5284
		5452	5458											
BIT8 =	000400	1263#	1383	1398	1411	1431	3042	3260	3264	3268	3270	3740	3744	3748
		3750	4572	4579	4584	5286	5296	5301	5304	5329				
BIT9 =	001000	1262#	1382	1397	1401	1410	1430	3072	3772	3776	3780	3782	3972	3979
		3984	4612	4619	4624	4851	4859	4864	5140	5143	5152	5157		
BRW	003306	1783	1861#											
BRX	003310	1784	1862#											
CALBCC	026304	6249*	6258	6279*	6288	6309*	6318	6339*	6390*	6397#	6427*	6428	6451	6463
CARDET=	010000	1379#												
CHRCNT	004236	2017*	2021	2037*	2055#	2056								
CLK =	020000	1406#	2187	2190	5768	5773								
CLK.A	001433	1517#	2303											
CLRVEC	005104	2212#												
CNVRT =	104411	1484#	1800	1802	1804	1806	2130	2132						
CONVRT=	104410	1482#	1736	2146	6479									
CORMAX	030146	6487#												
CRCEN =	001000	1401#	5714	5762	5870	5974	6041	6089	6116	6162	6207			
CRCERR=	010000	1395#												
CRC.CC=	102010	6236	6399#	6424										
CRC16 =	120001	6398#												
CREAM	001316	1371#	1598*	2257*	2258	2260*	2265	2266*	2267	2270*				

MERRX	005367	1805	2223#													
MERR2	005206	2223#	2249													
MERR3	005255	1755	2223#													
MEXT =	010000	1419#	2159													
MHI	027436	6477	6487#													
MIND	001330	1440#														
MJMPR	005526	1669	2223#													
MLOCK	005313	1779	2223#													
MMODE =	014000	1418#	5715	5717	5761	5835	5869	5973	6040	6088	6115	6161	6206	6243		
		6273	6303	6333	6433											
MMODEA=	004000	1408#														
MMODEB=	010000	1407#														
MNEW	005414	1750	2223#													
MPAR	005703	1652	2223#													
MPASSX	005356	1803	2223#													
MPOWER	005141	2208	2223#													
MQM	005132	1916	2223#	2332												
MR	005203	1786	2223#	2327												
MRESET=	000400	1272#	1411#	3134	3166	3198	3230	3262	3294	3326	3358	3390	3422	3454		
		3486	3518	3550	3582	3614	3646	3678	3710	3742	3774	3806	3901	3941		
		3981	4021	4061	4101	4141	4181	4221	4261	4301	4341	4381	4421	4461		
		4501	4541	4581	4621	4661	4701	4741	4781	4826	4861	4903	4945	4986		
		5026	5066	5106	5154	5202	5250	5298	5579	5611	5643	5671	5689	5712		
		5758	5832	5866	5970	6037	6085	6108	6112	6158	6203	6240	6270	6300		
		6330	6422	6462												
MSRJM	005762	1683	2223#													
MSTJM	005727	1679	2223#													
MTCN	005601	1673	2223#													
MTDATA=	040000	1405#	2346	2355	2367	2372	2379	5811	5886	5919	5982	5995	6010	6060		
		6099	6124	6131	6181											
MTITLE	001000	1310#	1620													
MTOTAL	005650	1661	2223#													
MTSTN	005400	2129	2223#	2311												
MTSTPC	005301	2223#														
MVEC	005513	1644	2223#													
MVECX	005350	1801	2223#													
NEXT	001216	1330#	1857	2174	2398*	2430*	2484*	2508*	2532*	2557*	2583*	2608*	2640*	2672*		
		2704*	2736*	2768*	2800*	2830*	2860*	2890*	2920*	2950*	2980*	3010*	3040*	3070*		
		3100*	3130*	3162*	3194*	3226*	3258*	3290*	3322*	3354*	3386*	3418*	3450*	3482*		
		3514*	3546*	3578*	3610*	3642*	3674*	3706*	3738*	3770*	3802*	3835*	3863*	3890*		
		3930*	3970*	4010*	4050*	4090*	4130*	4170*	4210*	4250*	4290*	4330*	4370*	4410*		
		4450*	4490*	4530*	4570*	4610*	4650*	4690*	4730*	4770*	4810*	4849*	4891*	4933*		
		4975*	5015*	5055*	5095*	5138*	5186*	5234*	5282*	5327*	5357*	5387*	5417*	5447*		
		5479*	5511*	5543*	5575*	5607*	5639*	5670*	5708*	5755*	5829*	5863*	5967*	6034*		
		6082*	6155*	6200*	6235*	6421*										
OPCLRJ	001323	1437#	1671	1686	1730*											
OVRUN=	040000	1394#														
PARAM =	104405	1476#	1637	1645	1653	1662	2312									
PARAM1	003576	1934#	1951													
PARBIT=	000000	1272#														
PARCSR	001410	1501#	2280*	2534	3864	5139	5187	5235	5714*	5762*	5870*	5974*	6041*	6089*		
		6116#	6162*	6207*												
PARERR	003652	1937	1939	1941	1950#	1957	1959	1961								
PASCNT	001230	1335#	1595*	1796*	1797	1830										
PERFOR=	000000	1272#														
PKCLK =	104412	1486#	2344	2354	2366	2371	2378	5735	5767	5774	5800	5804	5838	5881		

TST126	021420	5357	5386#
TST127	021474	5387	5416#
TST13	010150	2672	2703#
TST130	021550	5417	5446#
TST131	021634	5447	5478#
TST132	021720	5479	5510#
TST133	022004	5511	5542#
TST134	022070	5543	5574#
TST135	022164	5575	5606#
TST136	022260	5607	5638#
TST137	022354	5639	5669#
TST14	010234	2704	2735#
TST140	022510	5670	5707#
TST141	022664	5708	5754#
TST142	023200	5755	5828#
TST143	023270	5829	5862#
TST144	023644	5863	5966#
TST145	024132	5967	6033#
TST146	024334	6034	6081#
TST147	024706	6082	6154#
TST15	010320	2736	2767#
TST150	025102	6155	6199#
TST151	025236	6200	6234#
TST152	026306	6235	6420#
TST153=	***** U	6421	
TST16	010404	2768	2799#
TST17	010460	2800	2829#
TST2	007264	2398	2429#
TST20	010534	2830	2859#
TST21	010610	2860	2889#
TST22	010664	2890	2919#
TST23	010740	2920	2949#
TST24	011014	2950	2979#
TST25	011070	2980	3009#
TST26	011144	3010	3039#
TST27	011220	3040	3069#
TST3	007442	2430	2483#
TST30	011274	3070	3099#
TST31	011350	3100	3129#
TST32	011444	3130	3161#
TST33	011540	3162	3193#
TST34	011634	3194	3225#
TST35	011730	3226	3257#
TST36	012024	3258	3289#
TST37	012120	3290	3321#
TST4	007504	2484	2507#
TST40	012214	3322	3353#
TST41	012310	3354	3385#
TST42	012404	3386	3417#
TST43	012500	3418	3449#
TST44	012574	3450	3481#
TST45	012670	3482	3513#
TST46	012764	3514	3545#
TST47	013060	3546	3577#
TST5	007542	2508	2531#
TST50	013154	3578	3609#

6487

SQUEST	1#	1626													
SRESET	1#	3134	3166	3198	3230	3262	3294	3326	3358	3390	3422	3454	3486	3518	3550
	3582	3614	3646	3678	3710	3742	3774	3806	3901	3941	3981	4021	4061	4101	4141
	4181	4221	4261	4301	4341	4381	4421	4461	4501	4541	4581	4621	4661	4701	4741
	4781	4826	4861	4903	4945	4986	5026	5066	5106	5154	5202	5250	5298	5579	5611
	5643	5671	5688	5712	5758	5832	5866	5970	6037	6085	6108	6112	6158	6203	6239
	6269	6299	6329	6422	6462										
SSCOPE	1#	1834													
SSCOP2	1#	1190#	6466												
SSETFL	1#	2059													
SSIMBC	1190#	6362													
SSMALL	1#	2195													
STCRCX	1190#	6238	6268	6298	6328										
STIME	1190#	5658													
STRPDE	1#	1466	1468	1470	1472	1474	1476	1478	1480	1482	1484	1486	1488		
STSETU	1190#	5864	6035	6083	6110	6156	6201								
STSTN	1#	2391	2423	2477	2501	2525	2550	2576	2601	2633	2665	2697	2729	2761	2793
	2823	2853	2883	2913	2943	2973	3003	3033	3063	3093	3123	3155	3187	3219	3251
	3283	3315	3347	3379	3411	3443	3475	3507	3539	3571	3603	3635	3667	3699	3731
	3763	3795	3828	3856	3883	3923	3963	4003	4043	4083	4123	4163	4203	4243	4283
	4323	4363	4403	4443	4483	4523	4563	4603	4643	4683	4723	4763	4803	4842	4884
	4926	4968	5008	5048	5088	5131	5179	5227	5275	5320	5350	5380	5410	5440	5472
	5504	5536	5568	5600	5632	5663	5701	5748	5822	5856	5960	6027	6075	6148	6193
	6228	6414													
STT1	1190#	5694													
STT10	1190#	6019													
STT11	1190#	6067													
STT12	1190#	6140													
STT13	1190#	6186													
STT2	1190#	5741													
STT3	1190#	5816													
STT4	1190#	5842													
STT5	1190#	5951													
STXXCR	1190#	6222													
SVARIA	1#	1308													
SWAIT	1#														
SXZ	1#	2386	2389	2416	2421	2471	2475	2496	2499	2519	2523	2544	2548	2570	2574
	2595	2599	2627	2631	2659	2663	2691	2695	2723	2727	2755	2759	2787	2791	2817
	2821	2847	2851	2877	2881	2907	2911	2937	2941	2967	2971	2997	3001	3027	3031
	3057	3061	3087	3091	3117	3121	3149	3153	3181	3185	3213	3217	3245	3249	3277
	3281	3309	3313	3341	3345	3373	3377	3405	3409	3437	3441	3469	3473	3501	3505
	3533	3537	3565	3569	3597	3601	3629	3633	3661	3665	3693	3697	3725	3729	3757
	3761	3789	3793	3821	3826	3847	3854	3877	3881	3917	3921	3957	3961	3997	4001
	4037	4041	4077	4081	4117	4121	4157	4161	4197	4201	4237	4241	4277	4281	4317
	4321	4357	4361	4397	4401	4437	4441	4477	4481	4517	4521	4557	4561	4597	4601
	4637	4641	4677	4681	4717	4721	4757	4761	4796	4801	4836	4840	4878	4882	4920
	4924	4962	4966	5002	5006	5042	5046	5082	5086	5122	5129	5170	5177	5218	5225
	5266	5273	5314	5318	5344	5348	5374	5378	5404	5408	5434	5438	5466	5470	5498
	5502	5530	5534	5562	5566	5594	5598	5626	5630	5658	5661	5695	5699	5742	5746
	5817	5820	5843	5854	5952	5958	6020	6026	6068	6073	6141	6146	6187	6191	6222
	6226	6402	6413												

DZDPBB MACY11 27(1006) 18-MAY-77 00:00 PAGE 149
DZDPBB.P11 13-MAY-77 15:27 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0146

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

DSKZ:DZDPBB.DSKZ:DZDPBB.SEQ/SOL/CRF/DOC/NL:TOC=DZDPBB.MAC,DZDPBB.P11
RUN-TIME: 18 25 1 SECONDS
RUN-TIME RATIO: 176/45=3.8
CORE USED: 25K (49 PAGES)

DOCUMENT PAGES: 146