

1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
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
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
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
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300

- 4. USE PROCEDURE
- 4.1 STARTING PROCEDURE

BEFORE STARTING MAKE SURE THAT THE TTY IS IN REMOTE MODE, AND THE JUMPERS ARE INSTALLED. THREE STARTING ADDRESSES ARE PROVIDED.

0200 - THIS STARTING ADDRESS REQUESTS DM11 PARAMETERS, AND MUST BE USED TO INITIALLY START THE PROGRAM, AND WHENEVER ANY OF THE PARAMETERS LISTED BELOW IS CHANGED.

A. VECTOR ADDRESS ?
RESPONSE: TYPE IN THE VECTOR ADDRESS OF THE DM11 RECEIVER UNDER TEST. CARRIAGE RETURN SELECTS 0300

B. UNIT #(8)?
RESPONSE: THE DM11 UNIT NUMBER CORRESPONDS TO THE ADDRESS TO WHICH THE CLOCK STATUS REGISTER (CSR) RESPONDS.

CSR ADDRESS	DM11 UNIT #	CSR ADDRESS	DM11 UNIT #
175000	0	175100	10
175010	1	175110	11
175020	2	175120	12
175030	3	175130	13
175040	4	175140	14
175050	5	175150	15
175060	6	175160	16
175070	7	175170	17

CARRIAGE RETURN SELECTS UNIT # 0.

C. WHAT IS THE CHARACTER LENGTH?
RESPONSE: CHARACTER LENGTH REFERS TO THE NUMBER OF DATA BITS PER CHARACTER (5-8). CARRIAGE RETURN SELECTS 8 DATA BITS PER CHARACTER.

D. PRG #
RESPONSE: TYPE PROGRAM NUMBER OF PROGRAM YOU WISH TO RUN. CARRIAGE RETURN SELECTS PROGRAM # 0.

NOTES:
CARRIAGE RETURN TERMINATES ALL RESPONSES.
ANY UNACCEPTABLE RESPONSE WILL RESULT IN A ? TYPEOUT AND THE PARAMETER WILL AGAIN BE REQUESTED.

0204 - THIS STARTING ADDRESS USES PREVIOUSLY DEFINED DM11 PARAMETERS AND REQUESTS THE PROGRAM NUMBER OF THE PROGRAM YOU WISH TO RUN.

0210 - THIS STARTING ADDRESS STARTS THE PREVIOUSLY SELECTED PROGRAM USING PREVIOUSLY SELECTED PARAMETERS.

132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173

4.2 SWITCH SETTINGS

THE FOLLOWING SWITCH SETTINGS APPLY TO PROGRAM #0.

SR 0-6	ROUTINE TO BE RUN (IF ENABLED BY SR-9)
SR 8	RING BELL ON ERROR
SR 9	LOOP SELECTED ROUTINE
SR 11	INHIBIT ITERATION (DO EACH ROUTINE ONCE)
SR 13	INHIBIT PRINTOUT
SR 14	SCOPE (LOOP ROUTINE)
SR 15	HALT ON ERROR

THIS PROGRAM HAS BEEN MODIFIED TO RUN ON A PROCESSOR WITH OR WITHOUT A HARDWARE SWITCH REGISTER. WHEN FIRST EXECUTED THE PROGRAM TESTS THE EXISTENCE OF A HARDWARE SWITCH REGISTER. IF NOT FOUND A SOFTWARE SWITCH REGISTER LOCATION (SWREG=LOC. 176) IS DEFAULTED TO. IF THIS IS THE CASE, UPON EXECUTION THE CONTENTS OF THE SWREG ARE DUMPED IN OCTAL ON THE CONSOLE TTY AND ANY CHANGES ARE REQUESTED

(IE) SWR=XXXXXX NEW=

POSSIBLE RESPONSES ARE:

1. <CR> IF NO CHANGES ARE TO BE MADE
2. 6 DIGITS 0-7 TO REPRESENT IN OCTAL THE NEW SWITCH REGISTER VALUE ;LAST DIGIT FOLLOWED BY <CR>.
3. ^U TO ALLOW REENTERING VALUE IF ERROR IS COMMITTED KEYING IN SWREG VALUE.

BUILT INTO THE PROGRAM IS THE ABILITY TO DYNAMICALLY CHANGE THE CONTENTS OF SWREG DURING PROGRAM EXECUTION. BY STRIKING ^G (CNTRL G) ON CONSOLE TTY THE OPERATOR SETS A REQUEST FLAG TO CHANGE THE CONTENTS OF SWREG, WHICH IS PROCESSED IN KEY AREAS OF THE PROGRAM CODE (IE) ERROR ROUTINES, AFTER HALTS END OF PASS, AND OTHER APPLICABLE AREAS.

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

5.0 PROGRAM DESCRIPTION

5.1 PRGO - LOGIC TESTS

PRGO CONSISTS OF 152(8) INDEPENDENT ROUTINES WHICH TEST VARIOUS FUNCTIONS OF THE DM11 HARDWARE. ANY OF THESE ROUTINES MAY BE INDIVIDUALLY SELECTED AND RUN (SEE SEC. 4.2 FOR SWITCH SETTING)

5.1.1 ROUTINE DESCRIPTION

ROUTINE TESTS

RT0 TESTS THE ABILITY TO REFERENCE THE FOUR DM11 REGISTERS (CONTROL STATUS REGISTER (CSR), BUFFER ACTIVE REGISTER (BAR), BREAK STATUS REGISTER (BKCSR), AND THE BASE REGISTER (BASREG)). IF AN ILLEGAL REFERENCE OCCURS WHEN THE CSR IS REFERENCED THE PROGRAM WILL INDICATE AN ERROR, AND AUTOMATICALLY LOOP THE ERROR AS LONG AS THE ERROR CONDITION EXISTS.
RT0 PC=XXXXXX

RT1-RT10 BIT 'BANGS' THE CSR (BITS 0,1,2,4,5,6,12,13), TESTING THAT EACH BIT IN THE CSR CAN BE INDIVIDUALLY SET AND CLEARED. TWO ERROR TYPES ARE DETECTED IN THESE TESTS, A BIT FAILED TO SET, AND/OR A BIT FAILED TO CLEAR. THE ERROR PRINTOUT SHOWS THE ROUTINE THAT FAILED AND THE PC WHERE THE ERROR WAS DETECTED.

RT11- TESTS THAT RESET AND CLEAR CLEAR ALL R/W BITS IN THE CSR. TWO ERROR TYPES ARE DETECTED IN THIS ROUTINE SHOWING THE CONTENTS OF THE CSR AFTER THE RESET & CLEAR INSTRUCTION. THE PROGRAM AUTOMATICALLY LOOPS IF AN ERROR OCCURS. SHOWN BELOW IS THE ERROR TYPEOUT.
RT11 PC=XXXXXX ERR S/B:000000 WAS:XXXXXX

209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300

RT12 LOADS A BINARY COUNT PATTERN INTO THE BKCSR AND READS BACK THE RESULTS. IF THE DATA READ BACK IS INCORRECT AN ERROR IS INDICATED. THE SCOPE SWITCH WILL CAUSE THE PROGRAM TO RELOAD THE BINARY NUMBER AND REPEAT THE TEST. THE ERROR TYPEOUT SHOWS CORRECT AND ACTUAL RESULTS. THE SECOND PORTION OF THE TEST CLEARS THE PREVIOUSLY LOADED NUMBER IF THE SCOPE SWITCH IS SET THE PROGRAM LOOPS BACK AND REPEATS THE CLEAR INSTRUCTION.

RT13 THIS ROUTINE LOADS RANDOM NUMBERS INTO THE BKCSR. IF A RANDOM NUMBER IS LOADED INCORRECTLY AN ERROR IS INDICATED SHOWING THE CORRECT AND ACTUAL RESULTS.

RT14 THIS ROUTINE TESTS THAT RESET WILL CLEAR ALL BREAK STATUS REGISTER (BKCSR) BITS. IF ALL BITS DO NOT CLEAR WHEN THE RESET IS GIVEN AN ERROR IS INDICATED. THE ERROR TYPEOUT SHOWS THE CORRECT RESULT (ALL 0'S) AND THE ACTUAL RESULT.

RT15-RT16 THESE ROUTINES ARE THE SAME AS RT12 & RT13 EXCEPT THAT THE BASE REGISTER IS TESTED.

RT17 THIS ROUTINE TESTS THAT ALL BAR BITS CAN BE INDIVIDUALLY SET AND CLEARED. THE ROUTINE SHIFTS A '1' THROUGH THE BAR THEREBY SETTING EACH BAR BIT AND THEN THE BAR BIT IS CLEARED. THE ERROR TYPEOUTS SHOW CORRECT AND ACTUAL RESULTS.

RT20 THIS ROUTINE TESTS THAT RESET AND CLEAR CLEAR ALL BAR BITS THE ERROR TYPEOUT SHOWS CORRECT AND ACTUAL RESULTS.

RT21-RT23 THESE ROUTINES TEST THAT THE CSR, BAR, AND BKCSR RESPOND PROPERLY TO BYTE COMMANDS. BOTH BYTES ARE REFERENCED IN THESE ROUTINES USING CLRB INSTRUCTIONS. THE ERROR TYPEOUT SHOWS CORRECT AND ACTUAL RESULTS.

RT24 THIS ROUTINE TESTS THAT THE DM11 CAN INTERRUPT THE PROCESSOR VIA THE OVER RUN BIT (CSR BIT 13). THE ERROR TYPEOUT SHOWS THE ROUTINE NUMBER AND THE PC WHERE THE ERROR WAS DETECTED.

RT25 THIS ROUTINE TESTS THAT THE DM11 INTERRUPTS THE PROCESSOR AT THE PROPER LEVEL.

RT26-RT45 THESE ROUTINES TEST THE BASIC TRANSMITTER FUNCTIONS ON EACH LINE

RT46-RT65 THESE ROUTINES TEST THE BASIC RECEIVER FUNCTIONS ON EACH LINE

RT66 THIS ROUTINE TESTS THAT THE DM11 WILL SET THE NEX BIT (CSR BIT 14). WHEN THE DM11 TRIES TO TRANSMIT FROM NON-EXISTANT MEMORY. ALL LINES ARE INDIVIDUALLY TRANSMITTED ON. THE ERROR TYPEOUT SHOWS THE FAILING LINE. ALSO TESTED IS THAT THE NEX BIT WHEN SET CAUSES AN INTERRUPT.

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

RT67 THIS ROUTINE TESTS THAT THE NEX BIT (CSR BIT 14) SETS WHEN THE DM11 TRIES TO REFERENCE THE TUMBLE TABLE THAT IS IN NON-EXISTANT MEMORY.

RT70 THIS ROUTINE TESTS THAT WHEN THE GO BIT (CSR BIT 0) IS CLEAR THAT NO DATA IS RECEIVED ON ANY LINE. ALL LINES ARE TRANSMITTED ON AND AFTER THE TRANSMISSION IS COMPLETE THE RECEIVER DONE FLAG IS TESTED. THE ERROR TYPEOUT SHOWS THE LINE ON WHICH DATA WAS RECEIVED.

THE TYPEOUT SHOWN BELOW SHOWS THAT DATA WAS RECEIVED ON LINE 0

RT70 PC=XXXXXX ERRS/B:000001 WAS: 000001

RT71 THIS ROUTINE TESTS THAT THE CURRENT ADDRESS IS INCREMENTED PROPERLY BY THE DM11. THE TABLE BELOW SHOWS THE ADDRESS LOADED INTO IN THE CURRENT ADDRESS TABLE BEFORE 2 CHARACTERS ARE TRANSMITTED THE RESULTANT ADDRESS AFTER THE CHARACTER IS TRANSMITTED

BEFORE	AFTER	BEFORE	AFTER
000000	000001	000777	001000
000001	000002	001777	002000
000003	000004	003777	004000
000007	000010	007777	010000
000017	000020	017777	020000
000037	000040	037777	040000
000077	000100	077777	100000
000177	000200	177777	000000

000377 000400

THE ERROR TYPEOUT SHOWS CORRECT AND ACTUAL CURRENT ADDRESS.

RT72 THIS ROUTINE TESTS THAT DATA CAN BE TRANSMITTED FROM ALL AVAILABLE CORE AND RECEIVED CORRECTLY. THIS IS DONE BY TRANSMITTING 1 CHARACTER FROM SEVERAL ADDRESSES IN EACH 4K BLOCK OF CORE ON LINE 0. THE ERROR TYPEOUT WILL SHOW TRANSMITTED AND ACTUAL RECEIVED DATA. IF A DATA ERROR RESULTED WHEN TRANSMITTING FROM THE FIRST 4K OF CORE EXAMINE THE CURRENT ADDRESS OF LINE 0 TO DETERMINE WHERE IN THE FIRST 4K OF CORE THE DM11 WAS TRANSMITTING FROM WHEN ERROR OCCURRED. FOR ERRORS IN OTHER 4K BLOCKS THE CORRECT RESULT CORRELATES TO THE ADDRESS WHERE THE ERROR OCCURRED. FOR EXAMPLE

RT72 PC=XXXXXX ERR S/B:000001 WAS XXXXXX.
INDICATES THAT THE DM11 FAILED TO TRANSMIT AND RECEIVE CORRECT DATA WHEN TRANSMITTING FROM LOCATION 2000.
THE TEST IS ABORTED BEFORE TRANSMITTING IF THE CORE LOCATION IS NON-EXISTANT.

309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351

RT73 THIS ROUTINE TESTS THAT THE TRANSMITTER CAN TRANSMIT 100 CHARACTERS ON EACH LINE. THE ROUTINE TESTS THAT EXACTLY 100 CHARACTERS HAVE BEEN TRANSMITTED BEFORE READY (CSR BIT15) SETS AND THE BAR BIT CLEARS. THE ERROR TIMEOUT GIVES THE NUMBER OF CHARACTERS RECEIVED AT THE TIME OF AN ERROR, AND THE FAILING LINE NUMBER (X2).

RT74 THIS ROUTINE TESTS THAT THE DM11 WILL STORE DATA SEQUENTIALLY IN THE TUMBLE TABLE AND ALSO THAT THE POINTER RETURNS TO THE TOP OF THE TABLE WHEN 64 CHARACTERS HAVE BEEN RECEIVED.

RT75-114 THESE ROUTINES CHECK THAT A BREAK CAN BE TRANSMITTED AND RECEIVED ON ALL ALINES.

R115-R134 THESE ROUTINES INDIVIDUALLY TRANSMIT, RECEIVE AND CHECK DATA PLUS PARITY ON EACH OF THE 16 DM11 LINES. ONLY DATA AND PARITY ERRORS ARE REPORTED.

RT131 THIS ROUTINE SIMULTANEOUSLY TRANSMITS AND RECEIVES A CHARACTER (ALL 1'S) ON THE 16 DM11 LINES. THE FOLLOWING TESTS ARE PERFORMED:

- A: THERE ARE 16 DATA ENTRIES (1 PER LINE)
- B: THERE ISN'T A 17TH ENTRY
- C: DATA IS CORRECT
- D: ONE ENTRY FOR EACH LINE

RT136 THIS ROUTINE TRANSMITS A BREAK ON EACH LINE. TESTS PERFORMED ARE THE SAME AS IN RT135.

RT137-RT144 THESE ROUTINES TRANSMIT 64 CHARACTERS ON EACH LINE WITH A DELAY BEFORE BEGINNING TRANSMISSION ON THE NEXT SUCCESSIVE LINE. THE DELAY BEFORE TRANSMITTING ON THE NEXT LINE IS HALVED BY SUCCESSIVE TESTS. NO DATA CHECKING IS PERFORMED BY THESE TESTS. TESTED ARE THAT OVER RUN (CSR BIT13) AND NEX (CSR BIT14) ARE NOT SET DURING TRANSMISSION/RECEPTION.

RT145 THIS ROUTINE TESTS PROPER OPERATION OF THE HALF DUPLEX BIT (CSR BIT1)

RT146 THIS ROUTINE TESTS THAT THE DM11 COMES TO AN 'ORDERLY HALT' WHEN THE RESET INSTRUCTION IS GIVEN. 'ORDERLY HALT' IS DEFINED AS CSR, BAR, AND BKCS CLEAR IMMEDIATLY AFTER THE RESET INSTRUCTION AND STAY CLEARED.

5.2 PRG1- TRANSMITTER SCOPE LOOP
PROGRAM 1 ALLOWS THE USER TO SCOPE THE DM11 TRANSMITTER FUNCTIONS WITH THE DM11 CONTINUOUSLY RUNNING UNDER USER SUPPLIED PARAMETERS.

5.3 PRG2- TRANSMITTER/RECEIVER SCOPE LOOP
PROGRAM 2 ALLOWS THE USER TO SCOPE THE DM11 RECEIVER FUNCTIONS WITH THE DM11 CONTINUOUSLY RUNNING UNDER USER SUPPLIED PARAMETERS.

403
404
405
406
407
408
409
410
411
412
413
414
415
416

6.0 PROGRAM 1 AND PROGRAM 2 PARAMETERS
WHEN PROGRAM 1 OR PROGRAM 2 ARE SELECTED ADDITIONAL PARAMETERS WILL
BE REQUESTED BY EACH PROGRAM AS SHOWN BELOW.

A: TYPE LINES TO BE TESTED

EXAMPLES:

TYPE TO SELECT LINE(S)

1	0
3	1,0
10	3
17	3,2,1,0
50	5,3
3101	10,7,6,0
17770	14,13,12,11,10,7,6,5,4,3
177777	ALL

NOTE, LINE NUMBERS ARE GIVEN IN OCTAL.

B: HOW MANY CHARACTERS

TYPE THE NUMBER OF CHARACTERS YOU WISH TO TRANSMIT. NOTE,
THE NUMBER OF CHARACTERS MUST BE LESS THAN 200, AND IS TAKEN
IN OCTAL.

C: PUT CHARACTER IN SR (0-7); DELAY IN SR (8-15)

SELF-EXPLANATORY. NOTE, THE DELAY REFERS TO A DELAY AFTER
ALL THE CHARACTERS HAVE BEEN TRANSMITTED AND BEFORE A NEW
TRANSMISSION PERIOD BEGINS.

7.0 PROGRAM LIMITATIONS
BECAUSE THE DM11 DIAGNOSTICS ARE INSENSITIVE TO 'REAL' ELAPSED TIME
THE DIAGNOSTIC DOES NOT 'KNOW' IF THE DM11 IS OPERATING AT THE COR-
RECT FREQUENCY OR THAT THE STOP CODE SELECTION LOGIC IS CORRECT,
THESE SHOULD BE CHECKED WITH A SCOPE.

8.0 PROGRAM NOTES
IF THE POWER FAILS THE PROGRAM TYPES AN ERROR MESSAGE INDICATING THE
ROUTINE THAT WAS RUNNING (PROG #0 ONLY) AND RESTARTS THE PROGRAM.

***** IMPORTANT NOTE *****

POWER FAIL TEST

A TEST OF THE POWER FAIL LOGIC SHOULD BE PERFORMED ON EACH UNIT.
SELECT & RUN ROUTINE 144 (L.A. = 210 SR =5144 PRESS START). TURN
THE POWER OFF THEN ON. THE PROGRAM WILL TYPE OUT THE POWER FAIL
ERROR

R144 PC=003622

AND CONTINUE RUNNING ROUTINE 144. LOWER SR 9 AND WAIT FOR END OF
TEST MESSAGE. 'TEST DZDMA COMPLETE'

NOTE: IF THE POWER IS TURNED OFF DURING A RESET INSTRUCTION THE
PROGRAM WILL HALT. PRESS CONTINUE AND REPEAT THE TEST

IF THE PROGRAM HANGS THE BUS EXAMINE THE CONTENTS OF RTNNO. THE
CONTENTS OF RTNNO IS THE ROUTINE NUMBER THAT WAS RUNNING AT THE TIME
OF THE FAILURE.

417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472

%
.TITLE MAINDEC-11-DZDMA-B DM11 LOGIC TESTS
.NLIST MC,MD,CND
.LIST ME
.ENABLE ABS,AMA

;DM11 LOGIC TESTS DIAGNOSTIC (MAINDEC-11-DZDMA)
;COPYRIGHT 1972,1976 DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
;PRG0- INPUT-OUTPUT LOGIC TESTS
;PRG1- TRANSMITTER SCOPE LOOP
;PRG2- TRANSMIT/RECEIVE SCOPE LOOP
;STANDARD SR SWITCH OPTIONS (SWITCH SET TO A 1)
;SR15- HALT ON ERROR
;SR14- SCOPE.
;SR13- INHIBIT PRINTOUT
;SR12- INHIBIT TRACE (NOT USED)
;SR11- INHIBIT ITERATION.
;SR10- LOOP PROGRAM. (NOT USED)
;SR9- LOOP ROUTINE.
;SR8- RING BELL ON AN ERROR
;SR6 THROUGH SR0 - NUMBER OF ROUTINE TO BE LOOPED.

;EQUATE STATEMENTS
CC=177776
PSW=177776
ERRVEC=4
NOP=240
OPEN=0
MANUAL=BIT15
LBIT17=100000
LBIT16=40000
LBIT15=20000
LBIT14=10000
LBIT13=4000
LBIT12=2000
LBIT11=1000
LBIT10=400
LBIT7=200
LBIT6=100
LBIT5=40
LBIT4=20
LBIT3=10
LBIT2=4
LBIT1=2
LBIT0=1
BIT15=100000
BIT14=40000
BIT13=20000
BIT12=10000
BIT11=4000
BIT10=2000
BIT9=1000
BIT8=400
BIT7=200
BIT6=100

;ADDRESS OF ERROR TRAP VECTOR

177776
177776
000004
000240
000000
100000
100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001
100000
040000
020000
010000
004000
002000
001000
000400
000200
000100

473 000040
474 000020
475 000010
476 000004
477 000002
478 000001
479 005726
480 022626
481 000340
482 000300
483 000240
484 000200
485 000140
486 000100
487 000040
488 000000
489
490 000000
491 000002
492 000004
493 000006
494 000010
495 000012
496 000014
497 000016
498 000020
499 000022
500 000024
501 000026
502 000030
503 000032
504 000034
505 000036
506 000000
507 000001
508 000002
509 000003
510 000004
511 000005
512 000006
513 000007
514
515 104000
516 104001
517 104002
518 104003
519 104004
520 104005
521 104006
522 104007
523 104010
524 104011
525 104012
526 104013
527 104014
528 104015

BIT5=40
BIT4=20
BIT3=10
BIT2=4
BIT1=2
BIT0=1
POPS1=5726
POPS2=022626
PRTY7=340
PRTY6=300
PRTY5=240
PRTY4=200
PRTY3=140
PRTY2=100
PRTY1=40
PRTY0=0
;LINE NUMBERS
LINED=0
LINE1=2
LINE2=4
LINE3=6
LINE4=10
LINE5=12
LINE6=14
LINE7=16
LINE10=20
LINE11=22
LINE12=24
LINE13=26
LINE14=30
LINE15=32
LINE16=34
LINE17=36
R0=%0
R1=%1
R2=%2
R3=%3
R4=%4
R5=%5
SP=%6
PC=%7
;ENT CALLS
TYPE=ENT+0
ERROR=ENT+1
DATCHK=ENT+2
CHALT=ENT+3
EHALT=ENT+4
SRESET=ENT+5
SCOPE=ENT+6
SAVREG=ENT+7
RSTREG=ENT+10
ERROR1=ENT+11
INITIALIZE=ENT+12
SUSWR=ENT+13
KBDIN=ENT+14
CNTLU=ENT+15

;POP THE STACK. SAME AS TST (6)+
;POP STACK TWICE. SAME AS CMP (6)+,(6)+
;PRIORITY LEVEL DEFINITIONS

529	104400	DELAY=TRAP+0	
530	000007	BELL=007	
531	177777	RTLAST=-1	
532			
533	000000	Y=0	
534	177777	X=-1	
535	000000	A=0	
536	000000	. = 0	
537	000000	.+2	; UNASSIGNED TRAP
538	000002	HALT	
539	000004	.+2	; SP OVERFLOW, BUS ERROR TRAP
540	000006	HALT	
541	000010	.+2	; RESERVED INSTRUCTION TRAP
542	000012	HALT	
543	000014	.+2	; TRACE TRAP
544	000016	HALT	
545	000020	.+2	; TRAP TO CALL IOX
546	000022	HALT	
547	000024	.+2	; POWER FAIL TRAP
548	000026	HALT	
549	000030	EMTINT	; EMT TRAP
550	000032	PRTY7	
551	000034	DLY	; TRAP TRAP. SIMILAR TO EMT
552	000036	PRTY7	
553	000040	.+2	
554	000042	HALT	; TRAPPED TO PREVIOUS ADDRESS.
555	000044	.+2	
556	000046	HALT	; TRAPPED TO PREVIOUS ADDRESS.
557	000050	.+2	
558	000052	HALT	; TRAPPED TO PREVIOUS ADDRESS.
559	000054	.+2	
560	000056	HALT	; TRAPPED TO PREVIOUS ADDRESS.
561	000060	.+2	
562	000062	HALT	; TRAPPED TO PREVIOUS ADDRESS.
563	000064	.+2	
564	000066	HALT	; TRAPPED TO PREVIOUS ADDRESS.
565	000070	.+2	
566	000072	HALT	; TRAPPED TO PREVIOUS ADDRESS.
567	000074	.+2	
568	000076	HALT	; TRAPPED TO PREVIOUS ADDRESS.
569	000100	.+2	
570	000102	HALT	; TRAPPED TO PREVIOUS ADDRESS.
571			
572			
573	000104	.+2	
574	000106	HALT	; TRAPPED TO PREVIOUS ADDRESS.
575	000110	.+2	
576	000112	HALT	; TRAPPED TO PREVIOUS ADDRESS.
577	000114	.+2	
578	000116	HALT	; TRAPPED TO PREVIOUS ADDRESS.
579	000120	.+2	
580	000122	HALT	; TRAPPED TO PREVIOUS ADDRESS.
581	000124	.+2	
582	000126	HALT	; TRAPPED TO PREVIOUS ADDRESS.
583	000130	.+2	
584	000132	HALT	; TRAPPED TO PREVIOUS ADDRESS.

MACHER:

585	000134	000136	.+2	
586	000136	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
587	000140	000142	.+2	
588	000142	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
589	000144	000146	.+2	
590	000146	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
591	000150	000152	.+2	
592	000152	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
593	000154	000156	.+2	
594	000156	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
595	000160	000162	.+2	
596	000162	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
597	000164	000166	.+2	
598	000166	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
599	000170	000172	.+2	
600	000172	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
601	000174	000176	.+2	
602	000176	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
603	000200	000202	.+2	
604	000202	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
605	000204	000206	.+2	
606	000206	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
607	000210	000212	.+2	
608	000212	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
609	000214	000216	.+2	
610	000216	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
611	000220	000222	.+2	
612	000222	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
613	000224	000226	.+2	
614	000226	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
615	000230	000232	.+2	
616	000232	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
617	000234	000236	.+2	
618	000236	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
619	000240	000242	.+2	
620	000242	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
621	000244	000246	.+2	
622	000246	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
623	000250	000252	.+2	
624	000252	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
625	000254	000256	.+2	
626	000256	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
627	000260	000262	.+2	
628	000262	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
629	000264	000266	.+2	
630	000266	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
631	000270	000272	.+2	
632	000272	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
633	000274	000276	.+2	
634	000276	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
635	000300	000302	.+2	
636	000302	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
637	000304	000306	.+2	
638	000306	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
639	000310	000312	.+2	
640	000312	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.

641	000314	000316	.+2	
642	000316	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
643	000320	000322	.+2	
644	000322	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
645	000324	000326	.+2	
646	000326	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
647	000330	000332	.+2	
648	000332	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
649	000334	000336	.+2	
650	000336	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
651	000340	000342	.+2	
652	000342	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
653	000344	000346	.+2	
654	000346	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
655	000350	000352	.+2	
656	000352	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
657	000354	000356	.+2	
658	000356	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
659	000360	000362	.+2	
660	000362	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
661	000364	000366	.+2	
662	000366	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
663	000370	000372	.+2	
664	000372	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
665	000374	000376	.+2	
666	000376	000000	HALT	; TRAPPED TO PREVIOUS ADDRESS.
667				

```

668
669
670 000046 000046
671 000052 000052
672 000052 060000
673
674
675 000174 000174
676 000174 000000
677 000176 000000
678
679 000200 000200
680 000200 000137 002062
681 000204 000137 002130
682
683 000210 000137 002206
684
685
686 001100 001100
687 001100 000000
688
689 001102 177570
690 001104 177570
691 001106 000000
692 001146 001146
693 001146 000000
694 001206 001206
695 001206 000000
696 001246 001246
697 001246 000000
698 001250 175000
699 001252 175002
700 001254 175004
701 001256 175006
702 001260 000000
703 001262 000240
704 001264 000000
705 001266 000240
706 001270 000000
707 001272 000000
708 001274 000000
709 001276 000000
710 001300 000000
711 001302 000000
712 001304 000000
713 001306 001306
714 001306 000000
715 001506 001506
716 001510 000000
717 001512 000000
718 001514 000000
719 001516 000000
720 001520 000000
721 001522 000000
722 001524 000000
723 001524 006362

```

```

.=46
SENDAD
.=52
60000
.=174
DISPREG: 0
SMREG: 0

```

```

.=200
JMP @#START
JMP @#RSTAT1
JMP @#RSTAT2

```

```

.=1100
SPBOT: 0
SMR: 177570
DISPLAY: 177570
CAT: OPEN
WCT: OPEN
BAT: OPEN
VAC: OPEN
CSR: 175000
BAR: 175002
BKCSR: 175004
BASREG: 175006
CLKINT: OPEN
CLKLVL: PRTYS
XMTINT: OPEN
XMTLVL: PRTYS
TTDAT: OPEN
LIMBIT: OPEN
BARDAT: OPEN
TTPTR: OPEN
RCVDAT: OPEN
XMTDAT: OPEN
CARMSK: OPEN
TUMTAB: OPEN
KSTART: OPEN
CURTST: OPEN
RTNNO: OPEN
NXTST: OPEN
ICTR: OPEN
SCOPTR: OPEN
PRGLIN: OPEN
PRGTAB: PRGO

```

```

;GO TO START OF DIAGNOSTIC.
;GO GET PROGRAM # & RESTART PROGRAM
;USING PREVIOUS DM11 PARAMETERS
;RESTART PREVIOUS PROGRAM USING
;PREVIOUS DM11 PARAMETERS

```

```

;STARTING ADDRESS OF
;CURRENT ADDRESS TABLE
;STARTING ADDRESS OF
;WORD COUNT TABLE
;STARTING ADDRESS OF
;BIT ASSEMBLY TABLE
;32. SPARE WORDS
;ADDRESS OF CLOCK STATUS REGISTER
;ADDRESS OF BUFFER ACTIVE REGISTER
;ADDRESS OF BREAK STATUS REGISTER
;ADDRESS OF BASE REGISTER
;DM11 VECTOR ADDRESS (RECEIVER)
;PRIORITY LEVEL
;DM11 VECTOR ADDRESS (TRANSMITTER)
;TRANSMITTER PRIORITY LEVEL
;TUMBLE TABLE DATA
;LINE BIT (FOR BAR)
;BAR DATA
;PROGRAM TUMBLE TABLE POINTER

```

```

;STARTING ADDRESS OF
;TUMBLE TABLE
;CURRENT PROGRAM START ADDRESS.
;CONTAINS ADDR OF CURRENT TEST.
;CONTAINS CURRENT TEST #.
;CONTAINS ADDR OF NEXT TEST.
;CONTAINS CURRENT ITERATION COUNT
;CONTAINS CURRENT SCOPE POINTER.
;PRGO START ADDRESS

```

724	001526	016002
725	001530	016052
726	001532	006404
727	001534	016014
728	001536	016064
729	001540	002764
730	001542	001660
731	001544	001640
732	001546	000000
733	001550	000000
734	001552	002674
735	001554	002364
736	001556	002574
737	001560	002634
738	001562	001676
739	001564	003076
740	001566	016660
741	001570	016740
742	001572	017004
743	001574	177560
744	001576	177562
745	001600	177564
746	001602	177566
747	001604	000000
748	001606	000000
749	001610	000000
750	001612	000000
751	001614	000000
752	001616	000000
753	001620	000000
754	001622	175001
755	001624	175003
756	001626	175005
757	001630	175007

```

PRG1
PRG2
RSTART: PRGOR
          PRG1R
          PRG2R
EMTTAB: TYP
          ERR
          DTCHK
          0
          0
          SRSETT
          ESCOPE
          SAVRG
          RSTRG
          ERR1
          INIT
          SUSMRR
          KBDINTT
          CNTLUU
TKCSR: 177560
TKOBR: 177562
TPCSR: 177564
TPOBR: 177566
COUNT: OPEN
BFRPTR: OPEN
PARBIT: OPEN
PCADD: OPEN
APCADD: OPEN
PRVCNT: OPEN
TIME: OPEN
.CSR: 175001
.BAR: 175003
.BKCSR: 175005
.BASREG: 175007

```

```

:PRG1 START ADDRESS
:PRG2 START ADDRESS
:PRGO RESTART ADDRESS
:PRG1 " "
:PRG2 " "
:POINTER TO TYPEOUT ROUTINE
:POINTER TO ERROR ROUTINE
:POINTER TO DATA COMPARISON ROUTINE

:POINTER TO RESET ROUTINE
:POINTER TO SCOPE ROUTINE
:POINTER TO SAVE REGISTERS ROUTINE
:POINTER TO RESTORE REGISTERS ROUTINE
:POINTER TO ERROR1 ROUTINE
:POINTER TO INITIALIZE ROUTINE

```

```

758
759
760      :ROUTINE TO TYPE OUT INCORRECT ROUTINE SELECTED
761 001632 104000      INCRTN: TYPE
762 001634 017241      MI
763 001636 000207      RTS      x7      ;TYPE INCORRECT ROUTINE SELECTED.
764                                     ;EXIT.
765
766 001640 123737 001300 001302 :DATA COMPARISON ROUTINE.
767 001646 001403      DTCHK: CMPB      RCVDAT,XMTDAT ;COMPARE RECEIVED & TRANSMITTED DATA
768 001650 004737 002034      BEQ      1$      ;CHARS. BRANCH IF SAME.
769 001654 104011      JSR      7,CNVDAT ;CONVERT RCVDAT & XMTDAT TO ASCII
770 001656 000002      1$:      RTI      ; EXIT.
771
772      :ERROR ROUTINE WHENEVER THE PROGRAM DETECTS AN ERROR THE ERROR
773      :AND ERROR1 EMT INSTRUCTIONS ENTER HERE. ERROR AT ERR:,AND
774      :ERROR1 AT ERR1:
775 001660 012737 000402 001760 ERR:  MOV      $402,ERRB ;MOV BR .+6 TO ERRB
776 001666 013737 001612 001614      MOV      PCADD,APCADD ;GET PC WHERE ERROR OCCURRED
777 001674 000410      BR      ERRA
778 001676 012737 000240 001760 ERR1: MOV      $240,ERRB ;MOVE NOP TO ERRB
779 001704 013737 001612 001614      MOV      PCADD,APCADD ;GET PC WHERE ERROR OCCURRED
780 001712 004737 002034      JSR      7,CNVDAT ;CONVERT RCVDAT & XMT DAT TO ASCII
781 001716 104014      ERRB:  KBDIN ;GO CHECK FOR ↑G
782 001720 032777 020000 177154      BIT      $BIT13,$SMR ;ERROR PRINTOUT DESIRED
783 001726 001017      BNE      ERRC ;BRANCH IF NO PRINTOUT
784 001730 004537 005440      JSR      5,0ACNV ;CONVERT
785 001734 001614      APCADD ;DATA
786 001736 017623      APC ;TO
787 001740 000006      6 ;ASCII
788 001742 004537 005440      JSR      5,0ACNV ;FOR
789 001746 001512      RTMNO ;PRINTOUT
790 001750 017614      ATNUMB
791 001752 000003      3
792 001754 104000      TYPE ;TYPE ERROR
793 001756 017611      EMO ;MESSAGE
794 001760 000000      ERRB:  OPEN ;NOP IF ERROR!, BR .+6 IF ERROR
795 001762 104000      TYPE ;TYPE ANOTHER MESSAGE
796 001764 017173      ERDAT ;IF ERROR 1
797 001766 032777 000400 177106 ERRC: BIT      $BIT8,$SMR ;RING BELL ON ERROR?
798 001774 001411      BEQ      ERRD ;BRANCH IF NO BELL ON ERROR
799 001776 105777 177576      TSTB   $TPCSR ;TELEPRINTER
800 002002 100375      BPL      .-4 ;READY?
801 002004 012777 000007 177570      MOV      $BELL,$TPDBR ;RING THE BELL
802 002012 105777 177562      TSTB   $TPCSR ;WAIT FOR THE BELL TO RING
803 002016 100375      BPL      .-4
804 002020 005777 177056      ERRD:  TST      $SMR ;HALT ON ERROR
805 002024 100001      BPL      ERREX ;GO TO EXIT IF NO HALT ON ERROR
806 002026 000000      HALT
807 002030 104014      ERREX: KBDIN ;CHECK FOR ↑G
808 002032 000002      RTI ;RETURN
809
810
811      :SUBROUTINE TO CONVERT RCVDAT AND XMTDAT TO ASCII AND PLACE
812      :IN MESSAGE.
813 002034 004537 005440      CNVDAT: JSR      5,0ACNV

```

814	002040	001302		XMTDAT		
815	002042	017205		AASB		
816	002044	000006		6		
817	002046	004537	005440	JSR	5,0ACNV	
818	002052	001300		RCVDAT		
819	002054	017221		AMAS		
820	002056	000006		6		
821	002060	000207		RTS	7	;EXIT
822						
823						

```

824                                     ;THE FIRST PART OF THE START ROUTINE CONTAINS A SHORT
825                                     ;ROUTINE TO CHECK FOR MEMORY MANAGEMENT, ALTHOUGH THIS
826                                     ;DIAGNOSTIC DOES NOT USE MEMORY MANAGEMENT, ITS PRESENCE
827                                     ;INDICATES THAT OVER 28K OF MEMORY MAY BE PRESENT IN WHICH
828                                     ;CASE TESTS RT66 AND RT67 MAY FAIL. IF MEM. MAN. IS
829                                     ;PRESENT THESE TESTS ARE SKIPPED BY THE PROGRAM.
830
831 002062 012706 001100          START:  MOV    #SPBOT,%6
832 002066 104013                SUSMR
833 002070 012737 002112 000004      MOV    #15,%ERRVEC          ;SEE IF SWITCH-LESS PROCESSOR
834 002076 005737 172300          TST    %172300             ;SET UP FOR ERROR TRAP
835 002102 012737 012202 011614      MOV    #RT70,%RT65+2      ;TEST FOR KT11
836                                     ;KT11 PRESENT, SET UP TO
837 002110 000402                BR     +6                  ;SKIP RT66 AND RT67
838 002112 012706 001100          1$:   MOV    #SPBOT,%6          ;TRAP OCCURRED, NO KT11 PRESENT,
839                                     ;RESET STACK
840 002116 012737 000006 000004      MOV    #ERRVEC+2,%ERRVEC  ;RESET ERROR TRAP
841
842
843                                     ;OUT INTERRUPTS (SET PRIORITY LEVEL 7)
844 002124 004737 003126          RSTAT1: JSR    7,%DMPAR        ;GET DM11 PARAMETERS
845 002130 012706 001100          MOV    #SPBOT,%6
846 002134 104012                INITIALIZE
847 002136 104000                TYPE
848 002140 017230                NO
849 002142 004537 003702          PRGNUM: JSR    5,RECD          ;GET PRGNUM AND PUT IT
850 002146 000000                PRGNUM: 0                ;HERE
851 002150 043737 001522 002146      BIC    PRGLIM,PRGNUM      ;MASK OFF UNUSED BITS
852 002156 006337 002146          ASL    PRGNUM            ;SHIFT PROGRAM #
853 002162 012737 004062 000024      MOV    #PFAIL,24
854 002170 012737 000340 000026      MOV    #PRTY7,26
855 002176 013700 002146          MOV    PRGNUM,%0         ;GET PROGRAM #
856 002202 000170 001524          JMP    %PRGTAB(0)        ;GO START PROGRAM
857 002206 012737 004062 000024      RSTAT2: MOV    #PFAIL,24
858 002214 012737 000340 000026      MOV    #PRTY7,26
859 002222 012706 001100          MOV    #SPBOT,%6
860 002226 104012                INITIALIZE
861 002230 013700 002146          MOV    PRGNUM,%0         ;GET PROGRAM #
862 002234 000170 001532          JMP    %RSTART(0)       ;GO RESTART PROGRAM
863 002240 022737 000176 001102      SRSET: CMP    #SWREG,SWR 1$
864 002246 001404                BEQ
865 002250 104000                TYPE
866 002252 017271                M3
867 002254 000000                HALT
868 002256 000401                BR     GETRDY
869 002260 104015                1$:   CNTLU
870 002262 013737 001506 001514      GETRDY: MOV    KSTART,NXTST ;GO GET SWREG SETTINGS
871 002270 012737 000006 000004      GTRDYX: MOV    #6,%ERRVEC ;ADDR OF 1ST ROUTINE TO NXTST
872 002276 000240                NOP
873 002300 104012                INITIALIZE
874 002302 004737 002510          GTRDYA: JSR    %7,FORMD      ;ROLL FORWARD TO "NEXT" ROUTINE.
875 002306 032777 001000 176566      BIT    #19,%SMR          ;CHECK SELECT ROUTINE SWITCH
876 002314 001003                BNE    GTRDYC            ;BRANCH IF SELECT ROUTINE SWITCH IS SET.
877 002316 000177 177166          JMP    %CURTST          ;GO RUN CURRENT ROUTINE.
878 002322 000450                BR     SCOPED            ;NO GO, MANUAL RTN BYPASSED.
879 002324 017700 176552          GTRDYC: MOV    %SMR,%0    ;(SR) TO RD

```

880	002330	042700	177600		BIC	#177600,%0	:MASK UNDESIRED BITS
881	002334	123700	001512		CMPB	RTNNO,%0	:COMPARE RTNNO TO (R0)
882	002340	001002			BNE	GTRDYD	:BRANCH IF ROUTINE NOT FOUND YET.
883	002342	000177	177142		JMP	%CURTST	:GO RUN ROUTINE.
884	002346	022737	177777	001514	GTRDYD: CMP	#-1,NXTST	:NO. CHECK FOR LAST ROUTINE.
885	002354	001352			BNE	GTRDYA	:BRANCH IF NOT LAST ROUTINE.
886	002356	004737	001632		JSR	%7,INCRTN	:YES, INCORRECT ROUTINE SELECTED.
887	002362	000737			BR	GETROY	:START OVER.
888							
889					:SCOPE SERVICE ROUTINE		
890	002364	000240			ESCOPE: NOP		
891	002366	104014			KBDIN		:CHECK FOR IG
892	002370	005077	176656		CLR	%BAR	:CLEAR ALL DM11 REGISTERS
893	002374	005077	176650		CLR	%CSR	:AND SET BASE REGISTER
894	002400	005077	176650		CLR	%BKCSR	:AT THE STARTING ADDRESS
895	002404	104012			INITIALIZE		
896	002406	013716	001520		MOV	SCOPTR,(SP)	
897	002412	032777	040000	176462	BIT	#BIT14,%SMR	:CHECK FOR SCOPE OPTION.
898	002420	001402			BEQ	SCOPEB	:BRANCH IF SCOPE SW NOT SET.
899	002422	000176	000000		SCOPEA: JMP	%SP	:RETURN TO ROUTINE
900	002426	032777	004000	176446	SCOPEB: BIT	#BIT11,%SMR	:TEST INHIBIT ITERATION SWITCH
901	002434	001003			BNE	SCOPEB	:BRANCH IF INHIBIT ITERATION SW SET.
902	002436	005337	001516		DEC	ICTR	:DECREMENT ITERATION COUNT.
903	002442	001367			BNE	SCOPEA	:BRANCH IF COUNT NOT 0.
904	002444	032777	001000	176430	SCOPEB: BIT	#BIT9,%SMR	:CHECK SELECT ROUTINE SWITCH
905	002452	001303			BNE	GETROY	:BRANCH IF SELECT RTN SW SET
906	002454	022737	177777	001514	CMP	#-1,NXTST	:LAST TEST?
907	002462	001302			BNE	GTRDYX	:BRANCH IF NOT LAST TEST.
908	002464	104000			TYPE		:TYPE
909	002466	017244			M2		: 'PRGEND'
910	002470	013700	000042		MOV	%42,%0	:CHECK XXDP/ACT11 MONITOR HOOK
911	002474	001672			BEQ	GETROY	
912	002476	000005			RESET		
913	002500	004710			SENDAD: JSR	7,(0)	:RETURN TO XXDP/ACT11 MONITOR
914	002502	000240			NOP		
915	002504	000240			NOP		
916	002506	000240			NOP		
917	002510	013705	001514		FORMD: MOV	NXTST,%5	:ADDR OF NEXT ROUTINE TO R5.
918	002514	012537	001512		MOV	(5)+,RTNNO	:GET NEXT ROUTINE NUMBER.
919	002520	012537	001514		MOV	(5)+,NXTST	:GET ADDR OF NEXT "NEXT" ROUTINE.
920	002524	012537	001516		MOV	(5)+,ICTR	:GET ITERATION COUNT.
921	002530	012537	001520		MOV	(5)+,SCOPTR	:GET SCOPE LOOP ENTRY POINTER.
922	002534	010537	001510		FORMDA: MOV	%5,CURTST	:ADDR OF NOW CURRENT TEST TO CURTST.
923	002540	000207			RTS	%7	:EXIT FORMD SUBROUTINE.
924							
925							
926					:EMT TRAP INTERPRETER		
927	002542	011646	000002		EMTINT: MOV	(6)-,(6)	:GET PC OF NEXT INSTRUCTION
928	002544	162716	000002		SUB	#2,(6)	:FORM PC OF EMT INSTRUCTION
929	002550	011637	001612		MOV	(6),PCADD	:GET PC OF EMT INSTRUCTION
930	002554	017616	000000		MOV	%6,%6	:GET EMT INSTRUCTION
931	002560	105066	000001		CLRB	1(6)	:CLEAR MSH OF EMT INSTRUCTION
932	002564	006316			ASL	(6)	:SHIFT EMT IDENTIFIER
933	002566	062716	001540		ADD	#EMTTAB,(6)	
934	002572	013607			MOV	%6)+,%7	:GO TO PROPER EMT
935							

```

936
937 002574 012637 002630
938 002600 012637 002632
939 002604 010446
940 002606 010346
941 002610 010246
942 002612 010146
943 002614 010046
944 002516 013746 002632
945 002622 013746 002630
946 002626 000002
947 002630 000000
948 002632 000000
949
950
951 002634 012637 002670
952 002640 012637 002672
953 002644 012600
954 002646 012601
955 002650 012602
956 002652 012603
957 002654 012604
958
959 002656 013746 002672
960 002662 013746 002670
961 002666 000002
962 002670 000000
963 002672 000000
964
965
966 002674 012700 052525
967 002700 005100
968 002702 010037 002676
969 002706 000005
970 002710 000002
971
972
973 002712 013700 002760
974 002716 006100
975 002720 006100
976 002722 013700 002762
977 002726 010037 002760
978 002732 006100
979 002734 006100
980 002736 063700 002762
981 002742 006100
982 002744 006100
983 002746 010037 002762
984 002752 013700 002760
985 002756 000207
986 002760 001233
987 002762 007622
988
989 002764 011600
990 002766 062716 000002
991 002772 011000
    
```

```

:SAVE REGS 0 TO 4 SUBROUTINE.
SAVRG:  MOV      (6)+,SVRPC      ;SAVE PC AND PSW.
        MOV      (6)+,SVRPSW
        MOV      %4,-(6)        ;SAVE REGS 0 - 4
        MOV      %3,-(6)        ;IN STACK.
        MOV      %2,-(6)
        MOV      %1,-(6)
        MOV      %0,-(6)
        MOV      SVRPSW,-(6)    ;RESTORE PC AND PSW.
        MOV      SVRPC,-(6)
        RTI                    ;EXIT.
SVRPC:  OPEN
SVRPSW: OPEN

:RESTORE REGS 0 TO 4 SUBROUTINE.
RSTRG:  MOV      (6)+,RSTPC      ;SAVE PC AND PSW.
        MOV      (6)+,RSTPSW
        MOV      (6)+,%0        ;RESTORE REGS 0 - 4
        MOV      (6)+,%1        ;FROM STACK.
        MOV      (6)+,%2
        MOV      (6)+,%3
        MOV      (6)+,%4
        MOV      RSTPSW,-(6)    ;RESTORE PC AND PSW.
        MOV      RSTPC,-(6)
        RTI                    ;EXIT
RSTPC:  OPEN
RSTPSW: OPEN

:ROUTINE TO ISSUE RESET.
SRSETT: MOV      #52525,%0      ;DATA TO RO.
        COM      %0            ;COMPLEMENT (RO).
        MOV      %0,SRSETT+2    ; (RO) TO SRSETT+2.
        RESET                    ;ISSUE RESET. (RO) IS
        RTI                    ;DISPLAYED. EXIT.

:RANDOM NUMBER GENERATOR. ROUTINE EXITS WITH NUMBER IN REGISTER 0.
RNGEN:  MOV      RP1,%0
        ROL      %0
        ROL      %0
        ADD     RP2,%0
        MOV     %0,RP1
        ROL      %0
        ROL      %0
        ADD     RP2,%0
        ROL      %0
        ROL      %0
        MOV     %0,RP2
        MOV     RP1,%0
        RTS     %7              ;EXIT. NUMBER IN RO
RP1:    1233
RP2:    7622

:SUBROUTINE TO OUTPUT ASCII MESSAGE ON TELETYPE PRINTER.
TYP:   MOV      @%6,%0        ;GET ADDRESS THAT CONTAINS MESSAGE ADDRESS.
        ADD     @2,%6         ;SET UP EXIT.
        MOV     @%0,%0        ;ADDRESS OF MESSAGE TO RO.
    
```

```

992 002774 112037 003074          TYPB:  MOV  (0),TYPDAT      ;GET CHARACTER
993 003000 122737 000100 003074  TYPB:  CMPB  #100,TYPDAT    ;CHECK FOR "a" CHARACTER
994 003006 001001          TYPB:  BNE   TYPD          ;BRANCH IF NOT "a".
995 003010 000002          TYPB:  RTI                    ;TERMINATOR CHAR. DONE. EXIT.
996 003012 122737 000045 003074  TYPD:  CMPB  #45,TYPDAT    ;CHECK FOR "x".
997 003020 001412          TYPD:  BEQ   TYPF          ;BRANCH IF "x".
998 003022 004737 003030          TYPD:  JSR   %7,TYPD      ;TYPE CHAR IN TYPDAT
999 003026 000762          TYPD:  BR    TYPB          ;
1000 003030 113777 003074 176544  TYPD:  MOV  TYPDAT,@TPDBR  ;OUTPUT CHARACTER TO PRINTER
1001 003036 105777 176536          TYPD:  TSTB @TPCSR      ;WAIT FOR DONE FLAG.
1002 003042 100375          TYPD:  BPL  .-4            ;
1003 003044 000207          TYPD:  RTS  %7            ;EXIT
1004 003046 112737 000015 003074  TYPF:  MOV  #15,TYPDAT    ;MOVE CARRIAGE RETURN CODE TO TYPDAT
1005 003054 004737 003030          TYPF:  JSR   %7,TYPD      ;GO TYPE CHAR.
1006 003060 112737 000012 003074  TYPG:  MOV  #12,TYPDAT    ;MOVE LF CODE TO TYPDAT.
1007 003066 004737 003030          TYPG:  JSR   %7,TYPD      ;GO TYPE CHAR.
1008 003072 000740          TYPG:  BR    TYPB          ;
1009 003074 000000          TYPD:  OPEN
1010
1011
1012          ;SUBROUTINE TO INITIALIZE STACK POINTER AND SET PROCESSOR PRIORITY
1013          ;LEVEL 7
1014 003076 012777 001106 176152  INIT:  MOV  #CAT,@BASREG  ;INITIALIZE THE BASE REGISTER
1015 003104 012737 000340 177776  INIT:  MOV  #PTY7,PSW    ;SET PRIORITY LEVEL 7
1016 003112 011637 001100          INIT:  MOV  (SP),SPBOT   ;GET RETURN ADDRESS
1017 003116 012706 001100          INIT:  MOV  #SPBOT,SP    ;SET BOTTOM OF THE STACK
1018 003122 000176 000000          INIT:  JMP  @(SP)        ;RETURN
1019
1020
1021          ;SUBROUTINE TO GET DM11 PARAMETERS
1022
1023          ;VECTOR ADDRESS
1024 003126 000240          DMPAR:  NOP                    ;BEGIN
1025 003130 004737 003646          DMPAR:  JSR   7,OVRLAY      ;PUT HALT,..+2 IN VECTOR AREA
1026 003134 104000          DMPAR:  TYPE                    ;ASK USER FOR RECEIVER INT. VECTOR
1027 003136 017064          DMPAR:  WHERE                    ;OF UNIT UNDER TEST
1028 003140 004537 003702          DMPAR:  JSR   5,RECD        ;GET VECTOR AND PUT IT
1029 003144 000000          DMPAR:  VECTOR:  0            ;HERE
1030 003146 005737 003144          DMPAR:  TST  VECTOR        ;
1031 003152 001003          DMPAR:  BNE  VECOK         ;
1032 003154 012737 000300 003144  DMPAR:  MOV  #300,VECTOR    ;SET VECTOR = TO 0300
1033 003162 023727 003144 000300  DMPAR:  CMP  VECTOR,#300   ;IS VECTOR HIGHER OR
1034 003170 103003          DMPAR:  BHS  VECOKB       ;EQUAL TO 0300
1035 003172 104000          DMPAR:  TYPE  '?'         ;TYPE '?'
1036 003174 017241          DMPAR:  M1                    ;
1037 003176 000753          DMPAR:  BR    DMPAR        ;ASK FOR ANOTHER VECTOR
1038 003200 023727 003144 000770  DMPAR:  CMP  VECTOR,#770   ;IS VECTOR = TO OR
1039 003206 101371          DMPAR:  BHI  VECOKA       ;LESS THAN 770
1040 003210 032737 000007 003144  DMPAR:  BIT  #7,VECTOR     ;LSB OF VECTOR MUST BE ALL 0'S
1041 003216 001365          DMPAR:  BNE  VECOKA       ;
1042 003220 013737 003144 001260  DMPAR:  MOV  VECTOR,CLKINT ;
1043 003226 062737 000004 003144  DMPAR:  ADD  #4,VECTOR     ;
1044 003234 013737 003144 001264  DMPAR:  MOV  VECTOR,XMTINT ;
1045
1046          ;UNIT NUMBER
1047 003242 104000          DMPAR:  TYPE

```

```

1048 003244 017140          WHICH
1049 003246 004537 003702    JSR      5,RECD          ;GET UNIT AND PUT IT
1050 003252 000000          UNIT:    0              ;HERE
1051 003254 023727 003252 000017  CMP      UNIT,#17      ;UNIT SELECTED MUST BE
1052 003262 101403          BLOS    UNTOKA        ;BETWEEN 0 & 17
1053 003264 104000          TYPE
1054 003266 017241          M1
1055 003270 000764          BR      DMPARB
1056 003272 006337 003252    UNTOKA: ASL      UNIT
1057 003276 006337 003252    ASL      UNIT
1058 003302 006337 003252    ASL      UNIT
1059 003306 012702 000004    MOV      #4,X2
1060 003312 012701 001250    MOV      #CSR,X1
1061 003316 042711 000370    UNTOKB: BIC      #370,(1) ;FORM ADDRESSES OF
1062 003322 063721 003252    ADD      UNIT,(1)+    ;REGISTERS OF UNIT SELECTED
1063 003326 005302          DEC      X2
1064 003330 001372          BNE     UNTOKB
1065
1066 003332 012702 000004    MOV      #4,X2
1067 003336 012703 001250    MOV      #CSR,X3
1068 003342 012701 001622    MOV      #.CSR,X1
1069 003346 012311          UNTOKC: MOV      (3)+,(1) ;FORM ODD BYTE ADDRESSES
1070 003350 005221          INC      (1)+
1071 003352 005302          DEC      X2
1072 003354 001374          BNE     UNTOKC
1073
1074 003356 104000          ;CHARACTER LENGTH
1075 003360 017155          DMPARC: TYPE
1076 003362 004537 003702    LEVEL
1077 003366 000000          JSR      5,RECD          ;GET LENGTH AND PUT IT
1078 003370 005737 003366    LENGTH: 0              ;HERE
1079 003374 001003          TST     LENGTH
1080 003376 012737 000010 003366    BNE     LENOKA
1081 003404 023727 003366 000005  MOV      #8,LENGTH
1082 003412 103003          LENOKA: CMP      LENGTH,#5 ;CHARACTER LENGTH SELECTED MUST
1083 003414 104000          LENOKB: BHS     LENGTH    ;BE BETWEEN 5-8
1084 003416 017241          TYPE    ;CARRIAGE RETURN SELECTS 8
1085 003420 000756          M1
1086 003422 023727 003366 000010  BR      DMPARC
1087 003430 101371          LENOKC: CMP      LENGTH,#8.
1088 003432 162737 000005 003366    BHI     LENOKB
1089 003440 006337 003366    SUB      #5,LENGTH
1090 003444 013701 003366    ASL     LENGTH
1091 003450 016137 003462 001304  MOV      LENGTH,X1
1092 003456 000240          MOV     LENOKD(1),#CARMSK ;SET CHARACTER LENGTH MASK
1093 003460 000207          NOP
1094
1095          RTS      7          ;EXIT PARAMETERS ROUTINE
1096
1097          ;THE BELOW TABLE REPRESENTS THE CHARACTER LENGTH MASK FOR 5,6,7, AND 8
1098          ;BITS PER CHARACTER RESPECTIVELY.
1099          LENOKD: 177740
1100          177700
1101          177600
1102          177400
1103          ;CALCULATE MACHINE TIME TO TRANSMIT ONE CHARACTER
1104          TIMER: CLR      TIME1

```

```

1104 003476 012737 177777 001146      MOV      #-1,UCT          ;SET UP TO TRANSMIT
1105 003504 012737 016350 001106      MOV      #OUTBUF,CAT     ;1 CHARACTER ON LINE 1
1106 003512 012777 003612 175540      MOV      #TIMEC,ACLKINT ;LOAD RECEIVER INTERRUPT
1107 003520 012777 000340 175534      MOV      #PRTY7,ACLKLVL ;AND PRIORITY
1108 003526 012777 000001 175516      MOV      #LBIT0,ABAR     ;START TRANSMITTING
1109 003534 012777 000105 175506      MOV      #BIT6+BIT2+BIT0,ACSR ;SET IE BIT
1110 003542 005037 177776          CLR      @PSW            ;SET PROCESSER PRIORITY LEVEL = 0
1111 003546 012737 000044 001604  TIMEA:  MOV      #44,COUNT
1112 003554 062737 000001 003642      ADD      #1,TIME1       ;INCREMENT MACH. TIME TO TRANSMIT
1113 003562 001007          BNE     TIMEB
1114 003564 005077 175460          CLR      @CSR
1115 003570 012737 000340 177776      MOV      #PRTY7,PSW     ;SET PROCESSER PRIORITY LEVEL = 7
1116 003576 104001          ERROR
1117 003600 000734          BR      TIMER
1118 003602 005337 001604  TIMEB:  DEC      COUNT
1119 003606 001375          BNE     -4
1120 003610 000756          BR      TIMEA
1121 003612 005077 175432          CLR      @CSR
1122 003616 013737 003642 003644  TIMEC:  MOV      TIME1,TIME14
1123 003624 006037 003644          ROR     TIME14
1124 003630 000241          CLC
1125 003632 006037 003644          ROR     TIME14
1126 003636 022626          POPSP2
1127 003640 000207          RTS      7              ;RESTORE STACK POINTER
1128                                     ;EXIT TIME CALCULATION ROUTINE
1129 003642 000000  TIME1:  OPEN
1130 003644 000000  TIME14: OPEN           ;CONTAINS MACHINE TIME TO XMIT 1 CHAR
1131                                     ;CONTAIN TIME TO XMIT 14 CHAR
1132                                     ;SUBROUTINE TO PUT HALT,..+2 IN VECTOR AREA (0300-1000)
1133 003646 012701 000300  OVRLAY: MOV      #300,%1
1134 003652 012702 000302      MOV      #302,%2
1135 003656 010221 1S:      MOV      %2,(1)+
1136 003660 005021      CLR      (1)+
1137 003662 020227 000776      CMP      %2,#776
1138 003666 001403      BEQ     2S
1139 003670 062702 000004      ADD      #4,%2
1140 003674 000770      BR      1S
1141 003676 000240 2S:      NOP
1142 003700 000207      RTS      7              ;EXIT
1143
1144
1145                                     ;SUBROUTINE TO RECEIVE DATA
1146                                     ;THIS SUBROUTINE RECEIVES DATA FROM THE KEYBOARD (UP TO SIX OCTAL
1147                                     ;DIGITS AND PLACES THEM INTO THE ADDRESS FOLLOWING THE SUBROUTINE
1148                                     ;CALL (JSR 5,RECD). NO REGISTER CONTENTS ARE DISTURBED.
1149
1150                                     ;SUBROUTINE TO INPUT DATA FROM TTY
1151
1152
1153 003702 010046  RECD:  MOV      R0,-(SP)
1154 003704 005015 1S:      CLR      (5)           ;CLEAR OLD DATA
1155 003706 012737 000007 004060      MOV      #7,CNT        ;SET CHAR COUNT
1156 003714 105777 175654 2S:      TSTB    @TKCSR         ;WAIT FOR CHAR
1157 003720 100375      BPL     2S
1158 003722 117700 175650      MOVB    @TKDBR,R0
1159 003726 142700 000200      BICB    #200,R0        ;STRIP OFF PARITY

```

```

1160 003732 110077 175644      MOVB   R0, JTPDBR      ;ECHO CHARACTER
1161 003736 122700 000025      CMPB   #25, R0        ;IS IT A ^U
1162 003742 001443              BEQ    5$             ;BRANCH IF YES
1163 003744 122700 000015      CMPB   #15, R0       ;IS IT A <CR>
1164 003750 001415              BEQ    6$             ;BRANCH IF YESS
1165 003752 142700 000060      BICB   #60, R0       ;CHECK FOR 0-7 (8)
1166 003756 132700 000110      BITB   #110, R0      ;BRANCH IF NOT
1167 003762 001031              BNE    7$
1168 003764 006315              ASL    (5)
1169 003766 006315              ASL    (5)
1170 003770 006315              ASL    (5)          ;SHIFT DATA
1171 003772 150015              BISB   R0, (5)      ;INSET NEW CHAR
1172 003774 005337 004060      DEC    CNT
1173 004000 001422              BEQ    7$            ;ONLY 6 CHAR'S PLEASE
1174 004002 000744              BR     2$            ;NEXT CHARACTER
1175 004004 105777 175570      6$:  TSTB  JTPCSR
1176 004010 100375              BPL    6$            ;WAIT FOR READY
1177 004012 012777 000012 175562  MOV    #12, JTPDBR  ;TYPE <LF>
1178 004020 105777 175554      8$:  TSTB  JTPCSR
1179 004024 100375              BPL    1$            ;WAIT FOR READY
1180 004026 005777 175550      CLR    JTPDBR       ;LOAD CHAR
1181 004032 105777 175542      9$:  TSTB  JTPCSR
1182 004036 100375              BPL    9$
1183 004040 005725              TST   (R5)+          ;ADJUST R5
1184 004042 012600              MOV   (SP)+, R0     ;RESTORE R0
1185 004044 000205              RTS    R5
1186 004046 104000              7$:  TYPE
1187 004050 017241              M1
1188 004052 104000              5$:  TYPE
1189 004054 017135              SCTLU
1190 004056 000712              BR     1$            ;START OVER
1191 004060 000000      CNT:  0
1192
1193      ;POWER FAIL ROUTINE
1194 004062 012737 004072 000024  PFAIL: MOV    #PWRUP, 24
1195 004070 000000              HALT
1196
1197      ;POWER UP SUBROUTINE
1198 004072 000005      PWRUP: RESET          ;GIVE TELEPRINTER TIME TO START
1199 004074 012706 001100      MOV    #SPBOT, %6
1200 004100 104001              ERROR          ;TYPE POWER FAIL ERROR
1201 004102 000137 002206      JMP    J#RSTAT2   ;GO RESTART PROGRAM
1202
1203      ;LINE TEST SUBROUTINE: THIS LINE TEST PROVIDES SEVERAL TESTS ON A DM11 LINE.
1204      ;THE SUBROUTINE IS CALLED BY JSR 5, LNTST. THIS INSTRUCTION PROVIDES THE
1205      ;LINE BIT AND LINE NUMBER. THE FOLLOWING LINE TESTS ARE PERFORMED:
1206      ;WAITS UNTIL CHARACTER SHOULD HAVE BEEN TRANSMITTED, THEN TESTS
1207
1208      ;THAT BAR BIT CLEARED          ;DO NEXT TEST IF ERROR
1209      ;READY SET                      ;DO NEXT TEST IF ERROR
1210      ;WORD COUNT WENT TO 0          ;DO NEXT TEST IF ERROR
1211      ;CURRENT ADDRESS DID NOT INCREMENT ;DO NEXT TEST IF ERROR
1212      ;INTERRUPTS TO CORRECT VECTOR    ;DO NEXT TEST IF ERROR (NO INTERRUPT)
1213      ;READY BIT CAN BE CLEARED        ;END OF TEST
1214 004106 012537 016176      XMTTST: MOV    (5)+, J#LINE ;GET LINE NUMBER
1215 004112 004737 006330      JSR    7, GTLINE    ;GO FORM LINE BIT (FOR BAR)

```

1216	004116	005037	001302		CLR	XMTDAT	
1217	004122	004537	005532	1S:	JSR	5, @XMITD	: GO TO TRANSMIT SUBROUTINE
1218	004126	177777			-1		: TRANSMIT ONE CHARACTER
1219	004130	012703	000010		MOV	#10, %3	: WAIT IN
1220	004134	005002			CLR	%2	: THIS
1221	004136	005302		2S:	DEC	%2	: LOOP
1222	004140	001376			BNE	.-2	: UNTIL THE
1223	004142	005303			DEC	%3	: TRANSMITTER
1224	004144	001374			BNE	2S	: IS FINISHED
1225	004146	017737	175100 001300		MOV	@BAR, RCVDAT	: BAR SHOULD NOW BE CLEAR
1226	004154	001401			BEQ	3S	: BRANCH IF IT IS
1227	004156	104011			ERROR1		: ERROR! BAR BIT FAILED TO CLEAR
1228	004160	005777	175064	3S:	TST	@CSR	: TEST READY BIT SHOULD BE SET
1229	004164	100401			BMI	4S	: BRANCH IF SET
1230	004166	104001			ERROR		: ERROR! READY NOT SET
1231	004170	013701	016176	4S:	MOV	@LINE, %1	: GET LINE NUMBER
1232	004174	016137	001146 001300		MOV	WCT(1), RCVDAT	: WORD COUNT SHOULD BE 0
1233	004202	001401			BEQ	5S	
1234	004204	104011			ERROR1		: ERROR! WORD COUNT NOT EQUAL TO 0
1235	004206	012737	016350 001302	5S:	MOV	@OUTBUF, XMTDAT	
1236	004214	016137	001106 001300		MOV	CAT(1), RCVDAT	: CURRENT ADDRESS SHOULD NOT HAVE INCREMENTED
1237	004222	023737	001300 001302		CMP	RCVDAT, XMTDAT	
1238	004230	001401			BEQ	6S	
1239	004232	104011			ERROR1		: ERROR! CURRENT ADDR. DID NOT INCREMENT
1240	004234	012777	004266 175022	6S:	MOV	@7S, @XMTINT	: LOAD TRANSMITTER INTERRUPT VECTOR
1241	004242	052777	010000 175000		BIS	@BIT12, @CSR	: ENABLE TRANSMITTER INTERRUPT
1242	004250	005037	177776		CLR	@PSW	: SET PROCESSOR PRIORITY =0
1243	004254	000240			NOP		
1244	004256	012737	000340 177776		MOV	@PRTY7, @PSW	: LOCK OUT INTERRUPTS
1245	004264	104001			ERROR		: TRANSMITTER FAILED TO INTERRUPT OR
1246					ERROR		: INTERRUPTED TO WRONG LOCATION AND HALTED WITH ADDRESS +2 DISPLAYED.
1247	004266	022626		7S:	CMP	(6)+, (6)+	: RESET STACK PTR
1248	004270	012737	000340 177776		MOV	@PRTY7, @PSW	: LOCK OUT INTERRUPTS
1249	004276	042777	110000 174744		BIC	@BIT12+BIT15, @CSR	: CLEAR XMIT IE & READY BITS
1250	004304	005777	174740		TST	@CSR	: TEST THAT READY CLEARED
1251	004310	100001			BPL	8S	: GO TO EXIT
1252	004312	104001			ERROR		: ERROR! READY FAILED TO CLEAR
1253	004314	005726		8S:	TST	(6)+	: RESET STACK PTR
1254	004316	104006			SCOPE		: SCOPE

1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310

: RECEIVER LINE TESTS
: THE RECEIVER LINE TEST SUBROUTINE IS ENTERED WITH
: A JSR 5, RCVTST INSTRUCTION FOLLOWED BY THE
: LINE BIT AND LINE NUMBER OF THE LINE TO BE
: TESTED. THE SUBROUTINE PERFORMS THE FOLLOWING
: TEST AS SHOWN BELOW. IN THE EVENT OF AN ERROR
: THE REMAINING TESTS ARE ABORTED
: TEST SEQUENCE AND ADDRESS TAG

CHARACTER DONE SETS RTSTA
CHARACTER DONE CAUSES INTERRUPT RTSTB
CHARACTER DONE CAN BE CLEARED RTSTC
TUMBLE TABLE ENTRY IS CORRECT RTSTD
NO ENTRY IN NEXT TABLE ADDRESS RTSTE
HARDWARE TABLE POINTER INCREMENTED RTSTF
NEXT ENTRY WAS CORRECT RTSTG

: NOTES: IF THE HARDWARE PROVIDES AN INCORRECT VECTOR
: ADDRESS THE PROGRAM WILL HALT AND DISPLAY
: THE INCORRECT VECTOR+2 IN THE ADDRESS LIGHTS.

1278	004320	012737	177777	016350	RCVTST: MOV	8-1,OUTBUF	:LOAD ALL 1'S INTO OUTPUT BUFFER
1279	004326	005037	001306		CLR	TUMTAB	:CLEAR THE FIRST
1280	004332	005037	001310		CLR	TUMTAB+2	:TWO TUMBLE TABLE ADDRESSES
1281	004336	012737	000340	177776	MOV	8PTY7,8PSW	:LOCK OUT INTERRUPTS
1282	004344	012537	016176		MOV	(5)+,LINE	:GET LINE NUMBER
1283	004350	004537	005532		JSR	5,8XMITD	:TRANSMIT 1 CHARACTER (0'S)
1284	004354	177777			-1		:ON LINE SPECIFIED BY JSR
1285	004356	052777	000001	174664	BIS	8BIT0,8CSR	:SET GO BIT
1286	004364	005777	174660		TST	8CSR	:WAIT FOR TRANSMITTER
1287	004370	100375			BPL	-4	:TO TRANSMIT 1 CHAR.
1288	004372	042777	100000	174650	BIC	8BIT15,8CSR	:CLEAR TRANSMITTER READY FLAG
1289	004400	005046			CLR	-(SP)	:SET WATCH DOG TIMER
1290	004402	105777	174642		1S:	TSTB 8CSR	:TEST CHAR. DONE FLAG
1291	004406	100404			BMI	2S	:BRANCH IF SET
1292	004410	005216			INC	(SP)	:WAIT FOR THE FLAG
1293	004412	001373			BNE	1S	
1294	004414	104001			ERROR		:ERROR! CHAR. DONE FLAG FAILED TO SET
1295	004416	000550			BR	8S	:GO TO EXIT
1296	004420	005726			2S:	TST (SP)+	:RESTORE STACK PTR
1297	004422	012777	004456	174630	MOV	83S,8CLKINT	:LOAD RECEIVER INTERRUPT VEC. ADRS.
1298	004430	052777	000100	174612	BIS	8BIT6,8CSR	:SET RECEIVER IE BIT
1299	004436	005037	177776		CLR	8PSW	:ENABLE INTERRUPTS
1300	004442	000240			NOP		
1301	004444	012737	000340	177776	MOV	8PTY7,PSW	:LOCK OUT INTERRUPTS
1302	004452	104001			ERROR		:RECEIVER FAILED TO INTERRUPT
1303	004454	000531			BR	8S	:GO TO EXIT
1304	004456	012737	000340	177776	3S:	MOV 8PTY7,8PSW	:LOCK OUT INTERRUPTS
1305	004464	022626			CMP	(6)+,(6)+	
1306	004466	042777	000300	174554	BIC	8BIT7+BIT6,8CSR	:CLEAR CHAR. DONE FLAG
1307	004474	105777	174550		TSTB	8CSR	:TEST THAT CHAR DONE FLAG CLEARED
1308	004500	100002			BPL	4S	:BRANCH IF CHAR. DONE FLAG CLEARED
1309	004502	104001			ERROR		:ERROR! CHAR. DONE FAILED TO CLEAR
1310	004504	000515			BR	8S	:GO TO EXIT

```

1311 004506 013737 001306 001300 4S:  MOV  TUMTAB,RCV DAT      ;GET TUMBLE TABLE ENTRY
1312 004514 042737 020000 001300      BIC  #BIT13,RCV DAT      ;CLEAR PARITY INDICATOR
1313 004522 012737 000377 001302      MOV  #377,XMT DAT      ;LOAD XMT DAT WITH TRANSMITTED DATA
1314 004530 043737 001304 001302      BIC  CARM SK,XMT DAT      ;CLEAR NON TRANSMITTED BITS
1315 004536 153737 016176 001303      BISB LINE,XMT DAT+1      ;LOAD XMT DAT WITH PROPER LINE #
1316 004544 052737 100000 001302      BIS  #BIT15,XMT DAT      ;SET VALID DATA ENTRY BIT IN XMT DAT
1317 004552 023737 001300 001302      CMP  RCV DAT,XMT DAT      ;COMPARE TUMBLE TABLE ENTRY (RCV DAT) &
1318                                     ;CORRECT RESULT (XMT DAT)
1319
1320 004560 001402      BEQ  5S
1321 004562 104011      ERROR1
1322 004564 000465      BR   5S      ;ERROR! INCORRECT TUMBLE TABLE
1323 004566 005037 001306      CLR  TUMTAB      ;ENTRY; GO TO EXIT
1324 004572 013737 001310 001300 5S:  MOV  TUMTAB+2,RCV DAT      ;GET NEXT ENTRY
1325 004600 001404      BEQ  6S      ;BRANCH IF ALL 0'S
1326 004602 005037 001302      CLR  XMT DAT
1327 004606 104011      ERROR1
1328 004610 000453      BR   6S      ;ERROR! FALSE ENTRY IN NEXT
1329 004612 004537 005532 6S:  JSR  5,#XMTD      ;TUMBLE TABLE ADDRESS
1330 004616 177777      -1      ;TRANSMIT 1 CHARACTER (ALL 1'S)
1331 004620 005777 174424      TST  @CSR      ;ON LINE SPECIFIED BY JSR
1332 004624 100375      BPL  -4      ;WAIT FOR TRANSMITTER
1333 004626 105777 174416      TSTB @CSR      ;READY FLAG
1334 004632 100375      BPL  -4      ;TEST FOR THE DONE FLAG
1335 004634 042777 000200 174406      BIC  #BIT7,@CSR      ;CLEAR CHAR. DONE FLAG
1336 004642 013737 001306 001300      MOV  TUMTAB,RCV DAT      ;TEST THAT HARDWARE TUMBLE
1337 004650 001404      BEQ  7S      ;TABLE POINTER INCREMENTED (+2)
1338 004652 005037 001302      CLR  XMT DAT
1339 004656 104011      ERROR1
1340 004660 000427      BR   7S      ;ERROR! TUMBLE TABLE POINTER DID
1341 004662 013737 001310 001300 7S:  MOV  TUMTAB+2,RCV DAT      ;NOT INCREMENT; GO TO EXIT
1342 004670 042737 020000 001300      BIC  #BIT13,RCV DAT      ;GET TUMBLE TABLE ENTRY
1343 004676 012737 000377 001302      MOV  #377,XMT DAT      ;CLEAR PARITY INDICATOR
1344 004704 043737 001304 001302      BIC  CARM SK,XMT DAT      ;LOAD XMT DAT WITH TRANSMITTED DATA
1345 004712 153737 016176 001303      BISB LINE,XMT DAT+1      ;CLEAR NON-TRANSMITTED BITS
1346 004720 052737 100000 001302      BIS  #BIT15,XMT DAT      ;LOAD LINE # INTO XMT DAT
1347 004726 023737 001300 001302      CMP  RCV DAT,XMT DAT      ;SET VALID DATA ENTRY BIT INTO XMT DAT
1348                                     ;COMPARE TUMBLE TABLE ENTRY (RCV DAT) &
1349                                     ;CORRECT RESULT (XMT DAT)
1349 004734 001401      BEQ  8S
1350 004736 104011      ERROR1
1351 004740 104006 8S:  SCOPE      ;ERROR! 2ND TUMBLE TABLE ENTRY
1352                                     ;WAS INCORRECT; SCOPE
1353
1354 :SUBROUTINE TO TEST BREAK OPERATION
1355 :THE TRANSMITTER WILL TRANSMIT THE BREAK FOR TWO CHARACTER
1356 :TIMES AND THEN THE FOLLOWING TESTS WILL BE PERFORMED
1357 :
1358 :A VALID DATA ENTRY WAS MADE      BKTSTB
1359 :BREAK BIT SET                      BKTSTC
1360 :DATA WAS ALL 0'S                   BKTSTD
1360 004742 012777 000001 174300 BKTST: MOV  #1,@CSR      ;SET THE GO BIT
1361 004750 011577 174300      MOV  (5),@BKCSR      ;SET THE BREAK BIT
1362 004754 105777 174270      TSTB @CSR      ;WAIT FOR THE RECEIVER TO
1363 004760 100375      BPL  -4      ;RECEIVE BREAK
1364 004762 042777 000200 174260      BIC  #BIT7,@CSR      ;CLEAR FLAG
1365 004770 105777 174254      TSTB @CSR      ;WAIT FOR THE RECEIVER TO
1366 004774 100375      BPL  -4      ;TO RECEIVE BREAK

```

```

1367 004776 042777 000200 174244 BIC @BIT7,@CSR ;CLEAR FLAG
1368 005004 005077 174244 CLR @BKCSR ;CLEAR BREAK BIT
1369 005010 005737 001306 15: TST TUMTAB ;TEST FOR VALID DATA ENTRY
1370 005014 100402 BMI 25
1371 005016 104001 ERROR ;ERROR! NO VALID DATA ENTRY
1372 005020 000421 BR 45 ;GO TO EXIT
1373 005022 032737 040000 001306 25: BIT @BIT14,TUMTAB ;TEST THAT BREAK BIT IS SET
1374 005030 001002 BNE 35 ;IN TUMBLE TABLE
1375 005032 104001 ERROR ;ERROR! BREAK BIT FAILED TO SET
1376 005034 000413 BR 45 ;GO TO EXIT
1377 005036 105737 001306 35: TSTB TUMTAB ;TEST THAT DATA IS ALL 0'S
1378 005042 001410 BEQ 45
1379 005044 005037 001300 CLR RCVDAT
1380 005050 113737 001306 001300 MOVB TUMTAB,RCVDAT ;GET RECEIVED DATA
1381 005056 005037 001302 CLR XMTDAT
1382 005062 104011 ERROR1 ;ERROR! DATA WAS NOT ALL 0'S
1383 005064 104006 45: SCOPE ;SCOPE
1384
1385
1386 ;SUBROUTINE TO TRANSMIT & RECEIVE ON ALL LINES THE DELAY BETWEEN
1387 ;TRANSMITTING ON A LINE IS SUPPLIED BY THE CALLING JSR INSTRUCTION.
1388 ;NOTE NO DATA CHECKING IS PERFORMED BY THIS TEST
1389 005066 012537 001604 DLYXMT: MOV (5)+,@COUNT ;GET CHARACTER DELAY COUNT
1390 005072 005037 001302 CLR XMTDAT
1391 005076 004537 005510 JSR 5,BMOVE ;LOAD OUTPUT BUFFER WITH DATA
1392 005102 017632 MSG1 ;TO BE TRANSMITTED
1393 005104 016350 OUTBUF
1394 005106 000100 64.
1395 005110 012737 000001 001272 MOV @LBIT0,@LINBIT
1396 005116 005037 016176 CLR @LINE
1397 005122 012777 000001 174120 15: MOV @BIT0,@CSR ;SET THE GO BIT
1398 005130 004537 005532 25: JSR 5,@XMTD ;TRANSMIT 64. CHAR.
1399 005134 177700 -64. ;ON A LINE
1400 005136 013737 003642 005152 MOV @TIME1,45
1401 005144 013704 001604 MOV @COUNT,%4 ;GET CHARACTER DELAY COUNT
1402 005150 104400 35: DELAY
1403 005152 000000 45: 0
1404 005154 005304 DEC %4
1405 005156 001374 BNE 35
1406 005160 062737 000002 016176 ADD @2,LINE ;FORM NEXT LINE NUMBER
1407 005166 006337 001272 ASL LINBIT ;SHIFT LINE BIT
1408 005172 103356 BCC 25 ;BRANCH IF ALL LINES NOT DONE
1409 005174 012704 000100 MOV @64,%4
1410 005200 013737 003642 005210 MOV TIME1,65
1411 005206 104400 55: DELAY
1412 005210 000000 65: 0
1413 005212 005304 DEC %4
1414 005214 001374 BNE 55
1415 005216 017737 174030 001300 MOV @BAR,RCVDAT ;GET & TEST BAR DATA
1416 005224 001402 BEQ 75 ;EXIT IF DONE
1417 005226 104011 ERROR1 ;ERROR! BAR SHOULD'VE BEEN CLEAR
1418 005230 000413 BR 85
1419 005232 022777 100201 174010 75: CMP @100201,@CSR ;TEST THAT ONLY DONE,GO,& READY BITS ARE SET
1420 005240 001407 BEQ 85
1421 005242 012737 100201 001302 MOV @100201,XMTDAT
1422 005250 017737 173774 001300 MOV @CSR,RCVDAT ;GET CSR CONTENTS

```

```

1423 005256 104011          ERROR1          ; INCORRECT CSR CONTENTS
1424 005260 005726          POPSP          ; RESET THE STACK
1425 005262 104006          SCOPE          ; SCOPE
1426
1427          ; SUBROUTINE TO DELAY A SPECIFIED NUMBER OF MILLISECONDS
1428 005264 011637 005334  DLY:  MOV      (6),38      ; GET DELAY COUNT ADDRESS.
1429 005270 062716 000002      ADD      #2,(6)        ; SET UP EXIT ADDRESS
1430 005274 017737 000034 005334  MOV      #38,38        ; GET DELAY COUNT
1431 005302 001413          BEQ      28            ; EXIT IF NO DELAY
1432 005304 012737 000050 005336  18:  MOV      #50,48        ;
1433 005312 162737 000001 005334  SUB      #1,38
1434 005320 001404          BEQ      28
1435 005322 005337 005336  DEC      48
1436 005326 001375          BNE     #-4
1437 005330 000765          BR      18
1438 005332 000002          28:  RTI              ; EXIT
1439 005334 000000          38:  OPEN            ; CONTAINS DELAY COUNT
1440 005336 000000          48:  OPEN            ; CONTAINS DELAY ROUTINE CONSTANT
1441
1442          ; SUBROUTINE TO INITIALIZE BINARY COUNT PATTERNS
1443 005340 012737 177777 005362  INBIN: MOV      #-1,RIND ; SET ALL VARIABLES
1444 005346 004537 005510          JSR      %S,BMOVE     ; TO MINUS 1.
1445 005352 005362          RIND
1446 005354 005363          RIND+1
1447 005356 000005          S
1448 005360 000207          RTS      %7          ; EXIT
1449 005362 000000          RIND: OPEN
1450 005364 000000          PTO:  OPEN
1451 005366 000000          PT1:  OPEN
1452
1453          ; SPECIAL BINARY COUNT PATTEFN SUBROUTINE. EXITS WITH BIN CHAR IN R1
1454 005370 013737 005364 005366  GTBIN: MOV      PTO,PT1 ; PREVIOUS BIN CHAR TO PT1
1455 005376 005137 005366          COM      PT1
1456 005402 005137 005362          COM      RIND
1457 005406 001002          BNE      +6
1458 005410 005237 005366          INC      PT1
1459 005414 042737 177400 005366  BIC      #177400,PT1 ; MASK TO 8 BITS
1460 005422 013737 005366 005364  MOV      PT1,PTO      ; SAVE BIN CHAR IN PTO
1461 005430 013701 005366          MOV      PT1,%1      ; BIN CHAR TO R1.
1462 005434 000240          NOP
1463 005436 000207          RTS      %7          ; EXIT.
1464
1465          ; OCTAL TO ASCII CONVERT ROUTINE
1466 005440 104007          OACNV: SAVREG        ; SAVE REGISTERS ON THE STACK
1467 005442 013504          MOV      @5(%R4),%4   ; GET OCTAL VALUE.
1468 005444 012501          MOV      5(%R1),%1    ; GET DESTINATION ADDR.
1469 005446 012502          MOV      5(%R2),%2    ; GET CONVERT COUNT.
1470 005450 060201          ADD      %2,%1        ; DEVELOP ADDR TO STORE 1ST CHAR.
1471 005452 010403          OACNVA: MOV      %4,%3
1472 005454 042703 177770          BIC      #177770,%3   ; ISOLATE LEAST SIGNIFICANT DIGIT.
1473 005460 062703 000060          ADD      #0,%3        ; CONVERT DIGIT TO ASCII.
1474 005464 110341          MOV      %3,-(1)      ; STORE ASCII CHARACTER.
1475 005466 042704 000007          BIC      #7,%4
1476 005472 006004          ROR      %4
1477 005474 006004          ROR      %4
1478 005476 006004          ROR      %4

```

```

1479 005500 005302          DEC    %2          :DONE ALL DIGITS?
1480 005502 001363          BNE    OACNVA       :BRANCH IF NOT DONE.
1481 005504 104010          RSTREG              :RESTORE THE REGISTERS
1482 005506 000205          RTS     %5          :DONE. EXIT.
1483
1484          :SUBROUTINE TO MOVE A VARIABLE NUMBER OF BYTES.
1485 005510 104007          BMOVE: SAVREG       :SAVE REGS.
1486 005512 012501          MOV     (5)+,%1     :GET FROM ADDRESS
1487 005514 012502          MOV     (5)+,%2     :GET TO ADDRESS
1488 005516 012503          MOV     (5)+,%3     :GET COUNT
1489 005520 112122          1S:  MOVB  (1)+,(2)+ :MOVE BYTE
1490 005522 005303          DEC     %3          :DECREMENT COUNT
1491 005524 001375          BNE    1S           :BRANCH IF NOT DONE.
1492 005526 104010          RSTREG              :RESTORE REGS.
1493 005530 000205          RTS     %5          :DONE EXIT
1494
1495          :SUBROUTINE TO TRANSMIT DATA. SUBROUTINE CALLED BY
1496          :JSR 5,XMITD
1497 005532 010046          XMITD: MOV     %0,-(SP) :SAVE RD ON THE STACK
1498 005534 013700 016176    MOV     %0LINE,%0    :GET LINE
1499 005540 004737 006330    JSR    7,%GTLINE    :FORM LINE BIT (FOR BAR)
1500 005544 012777 001106 173504  MOV     %CAT,%BASEG  :INITIALIZE BASE REGISTER
1501 005552 012760 016350 001106  MOV     %OUTBUF,CAT(0) :LOAD FIRST CHAR ADDRESS IN CAT
1502 005560 012560 001146    MOV     (5)+,%CT(0)  :GET WORD COUNT
1503 005564 053777 001272 173460  BIS    %LINEBIT,%BAR :LOAD LINE POSITION INTO BAR
1504 005572 012600          MOV     (SP)+,%0    :RESTORE RD
1505 005574 000205          RTS     5           :EXIT
1506
1507          :ROUTINE TO TEST A LINE.
1508          :THE LINE TO BE TESTED IS PROVIDED BY THE JSR CALL TO THE ROUTINE.
1509          :100. CHARACTERS ARE TRANSMITTED, RECEIVED AND CHECKED BY THIS ROUTINE.
1510 005576 012537 016176    DAT1ST: MOV     (5)+,%LINE :GET LINE NUMBER
1511 005602 012737 000144 001604  DAT1AA: MOV     %100,COUNT :GET CHARACTER COUNT
1512 005610 012702 016350    MOV     %OUTBUF,%2   :GET ADDRESS OF OUTPUT BUFFER
1513 005614 004737 005370    1S:  JSR    7,%GTBIN   :GET DATA
1514 005620 110122          MOVB   %1,(2)+      :LOAD OUTPUT BUFFER WITH DATA
1515 005622 005337 001604    DEC     COUNT        :GOT ALL DATA?
1516 005626 001372          BNE    1S           :
1517 005630 012701 001306    MOV     %TUMTAB,%1  :LOAD TUMBLE TABLE POINTER
1518 005634 005037 001306    CLR    TUMTAB
1519 005640 004537 005510    JSR    5,BMOVE
1520 005644 001306          TUMTAB
1521 005646 001307          TUMTAB+1
1522 005650 000177          177
1523 005652 012702 016514    MOV     %INBUF,%2   :SETUP INPUT BUFFER POINTER
1524 005656 052777 000001 173364  BIS    %BIT0,%CSR   :SET THE GO BIT
1525 005664 004537 005532    JSR    5,%XMITD
1526 005670 177634          -100.
1527 005672 032777 160000 173350  2S:  BIT    %BIT15+BIT14+BIT13,%CSR :TEST IF READY OR ANY ERROR
1528 005700 001004          BNE    3S           :FLAGS ARE SET
1529 005702 105777 173342          TSTB  %CSR         :WAIT FOR THE RECEIVER
1530 005706 100371          BPL    2S          :TO RECEIVE A CHARACTER
1531 005710 000415          BR     5S
1532 005712 032777 060000 173330  3S:  BIT    %BIT14+BIT13,%CSR :TEST FOR ERROR FLAGS
1533 005720 001403          BEQ   4S           :BRANCH NO ERROR
1534 005722 104001          ERROR

```

```

1535 005724 000137 006244          JMP      15$           ;GO EXIT
1536 005730 042777 100000 173312 4$: BIC      #BIT15,RCSR ;CLEAR TRANSMITTER READY FLAG
1537 005736 105777 173306          TSTB    RCSR         ;TEST FOR CHARACTER READY
1538 005742 100375          BPL     .-4
1539 005744 042777 000200 173276 5$: BIC      #BIT7,RCSR ;CLEAR CHAR DONE BIT
1540 005752 005711          TST     (1)
1541 005754 100401          BMI     .+4         ;TEST FOR VALID ENTRY
1542 005756 104001          ERROR  ;REPORT INVALID ENTRY
1543 005760 111122          MOVB   (1),(2)+     ;MOVE CHAR FROM TUM. TAB. TO INPUT BUFFER
1544
1545          ;ROUTINE TO STORE RECEIVED PARITY BIT IN PARITY BIT BUFFER
1546 005762 012705 000001          MOV     #1,%S      ;GET ROTATE COUNT
1547 005766 000261          SEC     ;SET THE CARRY BIT
1548 005770 032711 020000          BIT     #BIT13,(1) ;TEST RECEIVED PARITY BIT
1549 005774 001001          BNE     6$         ;BRANCH IF RECEIVED PARITY WAS ODD
1550 005776 000241          CLC     ;CLEAR CARRY BIT
1551 006000 004537 006250 6$: JSR     5,RORPARBUF ;ROTATE RECEIVED PARITY INTO PARITY BUFFER
1552
1553          ;ROUTINE TO TEST THAT ENTRY IS FOR THE CORRECT LINE.
1554 006004 011137 001270          MOV     (1),@TTDAT ;GET TABLE ENTRY
1555 006010 042737 160777 001270 BIC     #160777,TTDAT ;CLEAR ALL BUT LINE NUMBER
1556 006016 123737 016176 001271 CMPB    LINE,TTDAT+1 ;COMPARE LINE NUMBERS
1557 006024 001410          BEQ    7$
1558 006026 013737 016176 001302 MOV     LINE,XMTDAT ;GET CORRECT LINE # (X2)
1559 006034 013737 001270 001300 MOV     TTDAT,RCVDAT ;GET LINE # (X2) THAT FALSE DATA CAME IN ON
1560 006042 104011          ERROR1 ;ERROR! DATA CAME IN ON A LINE THAT
1561          ;PROGRAM WAS NOT TRANSMITTING ON.
1562 006044 000477          BR     15$         ;EXIT TEST
1563 006046 020127 001504 7$: CMP     %1,@TUMTAB+176 ;IS POINTER AT THE END
1564 006052 001002          BNE    8$         ;OF THE TABLE
1565 006054 012701 001304          MOV     @TUMTAB-2,%1
1566 006060 005721 8$: TST     (1)+      ;INCREMENT POINTER
1567 006062 010046          MOV     %0,-(6)    ;SAVE REGISTER ZERO
1568 006064 013700 016176          MOV     LINE,%0    ;FETCH LINE NUMBER
1569 006070 005760 001146          TST     WCT(0)     ;HAS THE LAST CHARACTER BEEN TRANSMITTED
1570 006074 001402          BEQ    .+6        ;LAST CHARACTER HAS BEEN TRANSMITTED
1571 006076 012600          MOV     (6)+,%0    ;RESTORE REGISTER ZERO
1572 006100 000674          BR     2$         ;GO WAIT FOR NEXT CHARACTER
1573 006102 012600          MOV     (6)+,%0    ;RESTORE REGISTER ZERO
1574 006104 012701 016350 9$: MOV     @OUTBUF,%1
1575 006110 012702 016514          MOV     @INBUF,%2
1576 006114 012705 000014          MOV     #12,%S    ;ROTATE PARITY BUFFER
1577 006120 004537 006250          JSR     5,RORPARBUF ;12. PLACES RIGHT
1578 006124 005037 001300 10$: CLR     RCVDAT
1579 006130 005037 001302          CLR     XMTDAT
1580 006134 020127 016513          CMP     %1,@OUTBUF+99. ;HAVE ALL CHARS. BEEN COMPARED
1581 006140 001441          BEQ    15$
1582 006142 112137 001302          MOVB   (1)+,XMTDAT ;GET TRANSMITTED CHARACTER
1583 006146 043737 001304 001302 BIC     CARMASK,XMTDAT ;CLEAR NON-TRANSMITTED BITS
1584 006154 111237 001300          MOVB   (2),RCVDAT ;GET RECEIVED CHARACTER
1585 006160 104002          DATCHK ;COMPARE TRANS. & RCVD. CHARS.
1586
1587          ;ROUTINE TO COMPUTE AND CHECK PARITY ON RECEIVED DATA
1588 006162 012703 000010 11$: MOV     #8,%3    ;GET BIT COUNTER
1589 006166 005000          CLR     %0        ;CLEAR COMPUTED PARITY INDICATOR
1590 006170 106037 001300 12$: RORB   RCVDAT    ;LOOK AT RECEIVED BIT

```

```

1591 006174 103001          BCC      13$      ;BRANCH IF A 0
1592 006176 005100          COM      %0      ;COMPLEMENT RO IF A 1
1593 006200 005303      13$: DEC      %3      ;DECREMENT BIT COUNTER
1594 005202 001372          BNE      12$      ;LOOK AT NEXT BIT IF NOT DONE
1595 006204 000240          NOP              ;IF COMPUTED PARITY WAS ODD RO WILL
1596                          ;CONTAIN ALL 1'S, IF EVEN RO = 0
1597 006206 112237 001300      MOVB     (2)+,RCVDAT ;GET RECEIVED CHARACTER
1598 006212 012705 000001      MOV      #1,%5     ;ROTATE PARITY BUFFER 1 PLACE
1599 006216 004537 006250      JSR     5,RORPARBUF ;RIGHT LEAVING RECEIVED PARITY BIT IN CARRY
1600 006222 103004          BCC      14$      ;BRANCH IF RECEIVED PARITY WAS EVEN
1601 006224 005700          TST      %0      ;TEST FOR COMPUTED ODD PARITY
1602 006226 001336          BNE      10$      ;BRANCH IF COMPUTED & RECEIVED WAS ODD
1603 006230 104001          ERROR     ;ERROR! COMPUTED =EVEN,RECEIVED = ODD
1604 006232 000734          BR       10$      ;CONTINUE TEST
1605 006234 005700      14$: TST      %0      ;TEST FOR EVEN COMPUTED PARITY
1606 006236 001732          BEQ      10$      ;BRANCH IF COMPUTED PARITY WAS EVEN
1607 006240 104001          ERROR     ;ERROR! COMPUTED =ODD,RECEIVED = EVEN
1608 006242 000730          BR       10$      ;CONTINUE TEST
1609 006244 005726      15$: POPSP     ;REPOSITION STACK POINTER
1610 006246 104006          SCOPE
1611
1612                          ;ROUTINE TO ROTATE PARITY BUFFER.
1613 006250 006037 006312      RORPARBUF:ROR     PAR0
1614 006254 006037 006314          ROR     PAR1
1615 006260 006037 006316          ROR     PAR2
1616 006264 006037 006320          ROR     PAR3
1617 006270 006037 006322          ROR     PAR4
1618 006274 006037 006324          ROR     PAR5
1619 006300 006037 006326          ROR     PAR6
1620 006304 005316          DEC     (SP)      ;DECREMENT ROTATE COUNT
1621 006306 001360          BNE     RORPARBUF
1622 006310 000205          RTS      5
1623
1624                          ;PARITY BUFFER
1624 006312 000000      PAR0: OPEN
1625 006314 000000      PAR1: OPEN
1626 006316 000000      PAR2: OPEN
1627 006320 000000      PAR3: OPEN
1628 006322 000000      PAR4: OPEN
1629 006324 000000      PAR5: OPEN
1630 006326 000000      PAR6: OPEN
1631
1632                          ;SUBROUTINE TO FORM LINE BIT POSITION WITH THE LINE # IN LINE
1633 006330 010046      GTLINE: MOV     %0,-(SP) ;SAVE RO ON THE STACK
1634 006332 005037 001272      CLR     2(LINEBIT) ;CLEAR LINE BIT
1635 006336 013700 016176      MOV     2(LINE,%0) ;GET LINE
1636 006342 000261          SEC     ;SET CARRY
1637 006344 006137 001272      1$: ROL     LINEBIT ;SHIFT LINE BIT
1638 006350 162700 000002      SUB     #2,%0      ;SUBTRACT 2 FROM LINE NUMBER
1639 006354 100373          BPL     1$         ;BRANCH IF GREATER THAN 0
1640 006356 012600          MOV     (SP)+,%0   ;RESTORE RO
1641 006360 000207          RTS      7         ;EXIT SUBROUTINE
1642

```

```

1643
1644 006362 104000 PRGO: TYPE
1645 006364 017427 PRGOM
1646 006366 012737 006422 001506 MOV #RTO,KSTART ;GET ADDRESS OF FIRST TEST
1647 006374 005037 001512 CLR RTNNO ;CLEAR ROUTINE #
1648 006400 000137 002240 JMP SRSET
1649 006404 012737 006422 001506 PRGOR: MOV #RTO,KSTART ;GET ADDRESS OF FIRST TEST
1650 006412 005037 001512 CLR RTNNO ;CLEAR ROUTINE NUMBER
1651 006416 000137 002262 JMP GETRDY ;GO AND START PROGRAM
1652
1653 006422 000000 RT0: 0 ;ROUTINE # 0 *
1654 006424 006464 RT1 ;ADDR OF NEXT ROUTINE. *
1655 006426 000144 100. ;ITERATION COUNT *
1656 006430 006432 RT0A ;SCOPE ENTRY POINT. *
1657 000000 X=X+1
1658 ;*****
1659
1660 ;TEST ABILITY TO REFERENCE CSR WITHOUT TRAPPING
1661 006432 012737 006454 000004 RT0A: MOV #15,2#ERRVEC ;SET UP ERROR TRAP.
1662 006440 005777 172604 TST 2CSR ;REFERENCE CSR
1663 006444 012737 000006 000004 MOV #ERRVEC+2,2#ERRVEC ;RESET TIME OUT TRAP
1664 006452 104006 SCOPE
1665 006454 162716 000004 15: SUB #4,(6) ;RESTORE PC TO WHERE THE ILLEGAL
1666 ;REFERENCE OCCURED
1667 006460 104001 ERROR ;ERROR! ILLEGAL REFERENCE OCCURED
1668 006462 000002 RTI ;LOOP ILLEGAL REFERENCE INSTRUCTION
1669 ;*****
1670 006464 000001 RT1: 1 ;ROUTINE # 1 *
1671 006466 006536 RT2 ;ADDR OF NEXT ROUTINE. *
1672 006470 000144 100. ;ITERATION COUNT *
1673 006472 006474 RT1A ;SCOPE ENTRY POINT. *
1674 000001 X=X+1
1675 ;*****
1676
1677 ;TEST THAT CSR BIT0 CAN BE SET AND CLEARED
1678 006474 012777 000001 172546 RT1A: MOV #BIT0,2CSR ;SET BIT0.
1679 006502 022777 000001 172540 CMP #BIT0,2CSR ;TEST THAT BIT0 IS SET
1680 006510 001402 BEQ 15 ;BRANCH IF SET
1681 006512 104001 ERROR ;CSR BIT0 FAILED TO SET
1682 006514 000407 BR 25 ;OR AN ADDITIONAL BIT ALSO SET
1683 006516 042777 000001 172524 15: BIC #BIT0,2CSR ;CLEAR BIT0
1684 006524 005777 172520 TST 2CSR ;TEST THAT BIT0 IS CLEAR
1685 006530 001401 BEQ 25
1686 006532 104001 ERROR ;CSR BIT0 FAILED TO CLEAR
1687 006534 104006 25: SCOPE
1688 ;*****
1689 006536 000002 RT2: 2 ;ROUTINE # 2 *
1690 006540 006610 RT3 ;ADDR OF NEXT ROUTINE. *
1691 006542 000144 100. ;ITERATION COUNT *
1692 006544 006546 RT2A ;SCOPE ENTRY POINT. *
1693 000002 X=X+1
1694 ;*****
1695
1696 ;TEST THAT CSR BIT1 CAN BE SET AND CLEARED
1697 006546 012777 000002 172474 RT2A: MOV #BIT1,2CSR ;SET BIT1.
1698 006554 022777 000002 172466 CMP #BIT1,2CSR ;TEST THAT BIT1 IS SET

```

```

1699 006562 001402          BEQ      1S          ;BRANCH IF SET
1700 006564 104001          ERROR                    ;CSR BIT1 FAILED TO SET
1701 006566 000407          BR       2S          ;OR AN ADDITIONAL BIT ALSO SET
1702 006570 042777 000002 172452 1S:    BIC     #BIT1,@CSR    ;CLEAR BIT1
1703 006576 005777 172446          TST     @CSR          ;TEST THAT BIT1 IS CLEAR
1704 006602 001401          BEQ      2S          ;CSR BIT1 FAILED TO CLEAR
1705 006604 104001          ERROR                    ;CSR BIT1 FAILED TO CLEAR
1706 006606 104006          2S:    SCOPE
1707          ;*****
1708 006610 000003          RT3:    3             ;ROUTINE # 3
1709 006612 006662          RT4     ;ADDR OF NEXT ROUTINE.
1710 006614 000144          100.    ;ITERATION COUNT
1711 006616 006620          RT3A   ;SCOPE ENTRY POINT.
1712          X=X+1
1713          ;*****
1714          ;TEST THAT CSR BIT2 CAN BE SET AND CLEARED
1715          RT3A:    MOV     #BIT2,@CSR    ;SET BIT2.
1716 006620 012777 000004 172422          CMP     #BIT2,@CSR    ;TEST THAT BIT2 IS SET
1717 006626 022777 000004 172414          BEQ     1S          ;BRANCH IF SET
1718 006634 001402          ERROR                    ;CSR BIT2 FAILED TO SET
1719 006636 104001          BR       2S          ;OR AN ADDITIONAL BIT ALSO SET
1720 006640 000407          BIC     #BIT2,@CSR    ;CLEAR BIT2
1721 006642 042777 000004 172400 1S:    TST     @CSR          ;TEST THAT BIT2 IS CLEAR
1722 006650 005777 172374          BEQ     2S          ;CSR BIT2 FAILED TO CLEAR
1723 006654 001401          ERROR                    ;CSR BIT2 FAILED TO CLEAR
1724 006656 104001          2S:    SCOPE
1725 006660 104006          ;*****
1726          ;*****
1727 006662 000004          RT4:    4             ;ROUTINE # 4
1728 006664 006734          RT5     ;ADDR OF NEXT ROUTINE.
1729 006666 000144          100.    ;ITERATION COUNT
1730 006670 006672          RT4A   ;SCOPE ENTRY POINT.
1731          X=X+1
1732          ;*****
1733          ;TEST THAT CSR BIT4 CAN BE SET AND CLEARED
1734          RT4A:    MOV     #BIT4,@CSR    ;SET BIT4.
1735 006672 012777 000020 172350          CMP     #BIT4,@CSR    ;TEST THAT BIT4 IS SET
1736 006700 022777 000020 172342          BEQ     1S          ;BRANCH IF SET
1737 006706 001402          ERROR                    ;CSR BIT4 FAILED TO SET
1738 006710 104001          BR       2S          ;OR AN ADDITIONAL BIT ALSO SET
1739 006712 000407          BIC     #BIT4,@CSR    ;CLEAR BIT4
1740 006714 042777 000020 172326 1S:    TST     @CSR          ;TEST THAT BIT4 IS CLEAR
1741 006722 005777 172322          BEQ     2S          ;CSR BIT4 FAILED TO CLEAR
1742 006726 001401          ERROR                    ;CSR BIT4 FAILED TO CLEAR
1743 006730 104001          2S:    SCOPE
1744 006732 104006          ;*****
1745          ;*****
1746 006734 000005          RT5:    5             ;ROUTINE # 5
1747 006736 007006          RT6     ;ADDR OF NEXT ROUTINE.
1748 006740 000144          100.    ;ITERATION COUNT
1749 006742 006744          RT5A   ;SCOPE ENTRY POINT.
1750          X=X+1
1751          ;*****
1752          ;TEST THAT CSR BITS CAN BE SET AND CLEARED
1753          RT5A:    MOV     #BITS,@CSR    ;SET BITS.
1754 006744 012777 000040 172276

```

```

1755 006752 022777 000040 172270      CMP      #BIT5,@CSR      ;TEST THAT BIT5 IS SET
1756 006760 001402                      BEQ      15              ;BRANCH IF SET
1757 006762 104001                      ERROR                      ;CSR BIT5 FAILED TO SET
1758 006764 000407                      BR       25              ;OR AN ADDITIONAL BIT ALSO SET
1759 006766 042777 000040 172254 15:   BIC      #BIT5,@CSR      ;CLEAR BIT5
1760 006774 005777 172250          TST      @CSR            ;TEST THAT BIT5 IS CLEAR
1761 007000 001401                      BEQ      25              ;CSR BIT5 FAILED TO CLEAR
1762 007002 104001                      ERROR                      ;CSR BIT5 FAILED TO CLEAR
1763 007004 104006                      SCOPE
1764                                     ;*****
1765 007006 000006          RT6:     6              ;ROUTINE # 6 *
1766 007010 007060          RT7      ;ADDR OF NEXT ROUTINE. *
1767 007012 000144          100.     ;ITERATION COUNT *
1768 007014 007016          RT6A     ;SCOPE ENTRY POINT. *
1769 000006          X=X+1
1770                                     ;*****
1771                                     ;TEST THAT CSR BIT6 CAN BE SET AND CLEARED
1772                                     RT6A:    MOV      #BIT6,@CSR      ;SET BIT6.
1773 007016 012777 000100 172224          CMP      #BIT6,@CSR      ;TEST THAT BIT6 IS SET
1774 007024 022777 000100 172216          BEQ      15              ;BRANCH IF SET
1775 007032 001402                      ERROR                      ;CSR BIT6 FAILED TO SET
1776 007034 104001                      BR       25              ;OR AN ADDITIONAL BIT ALSO SET
1777 007036 000407                      BR       25              ;OR AN ADDITIONAL BIT ALSO SET
1778 007040 042777 000100 172202 15:   BIC      #BIT6,@CSR      ;CLEAR BIT6
1779 007046 005777 172176          TST      @CSR            ;TEST THAT BIT6 IS CLEAR
1780 007052 001401                      BEQ      25              ;CSR BIT6 FAILED TO CLEAR
1781 007054 104001                      ERROR                      ;CSR BIT6 FAILED TO CLEAR
1782 007056 104006                      SCOPE
1783                                     ;*****
1784 007060 000007          RT7:     7              ;ROUTINE # 7 *
1785 007062 007132          RT10    ;ADDR OF NEXT ROUTINE. *
1786 007064 000144          100.     ;ITERATION COUNT *
1787 007066 007070          RT7A     ;SCOPE ENTRY POINT. *
1788 000007          X=X+1
1789                                     ;*****
1790                                     ;TEST THAT CSR BIT12 CAN BE SET AND CLEARED
1791                                     RT7A:    MOV      #BIT12,@CSR     ;SET BIT12.
1792 007070 012777 010000 172152          CMP      #BIT12,@CSR     ;TEST THAT BIT12 IS SET
1793 007076 022777 010000 172144          BEQ      15              ;BRANCH IF SET
1794 007104 001402                      ERROR                      ;CSR BIT12 FAILED TO SET
1795 007106 104001                      BR       25              ;OR AN ADDITIONAL BIT ALSO SET
1796 007110 000407                      BR       25              ;OR AN ADDITIONAL BIT ALSO SET
1797 007112 042777 010000 172130 15:   BIC      #BIT12,@CSR     ;CLEAR BIT12
1798 007120 005777 172124          TST      @CSR            ;TEST THAT BIT12 IS CLEAR
1799 007124 001401                      BEQ      25              ;CSR BIT12 FAILED TO CLEAR
1800 007126 104001                      ERROR                      ;CSR BIT12 FAILED TO CLEAR
1801 007130 104006                      SCOPE
1802                                     ;*****
1803 007132 000010          RT10:    10             ;ROUTINE # 10 *
1804 007134 007204          RT11    ;ADDR OF NEXT ROUTINE. *
1805 007136 000144          100.     ;ITERATION COUNT *
1806 007140 007142          RT10A   ;SCOPE ENTRY POINT. *
1807 000010          X=X+1
1808                                     ;*****
1809                                     ;TEST THAT CSR BIT13 CAN BE SET AND CLEARED
1810

```

```

1811 007142 012777 020000 172100 RT10A: MOV      #BIT13, @CSR      ;SET BIT13.
1812 007150 022777 020000 172072      CMP      #BIT13, @CSR      ;TEST THAT BIT13 IS SET
1813 007156 001402                BEQ      1$                ;BRANCH IF SET
1814 007160 104001                ERROR1   ;CSR BIT13 FAILED TO SET
1815 007162 000407                BR       2$                ;OR AN ADDITIONAL BIT ALSO SET
1816 007164 042777 020000 172056 1$: BIC      #BIT13, @CSR      ;CLEAR BIT13
1817 007172 005777 172052      TST      @CSR              ;TEST THAT BIT13 IS CLEAR
1818 007176 001401                BEQ      2$                ;CSR BIT13 FAILED TO CLEAR
1819 007200 104001                ERROR1   ;CSR BIT13 FAILED TO CLEAR
1820 007202 104006                SCOPE
1821                                     ;*****
1822 007204 000011 RT11: 11                ;ROUTINE # 11
1823 007206 007274      RT12                ;ADDR OF NEXT ROUTINE.
1824 007210 000144      100.                ;ITERATION COUNT
1825 007212 007214      RT11A                ;SCOPE ENTRY POINT.
1826 000011      X=X+1
1827                                     ;*****
1828                                     ;TEST THAT RESET & CLEAR INSTRUCTION CLEAR ALL R/W BITS IN THE CONTROL
1829                                     ;STATUS REG. (CSR)
1830 RT11A: MOV      #30167, @CSR      ;SET ALL R/W BITS IN THE CSR
1831 007214 012777 030167 172026      CLR      XMTDAT
1832 007222 005037 001302                SRESET   ;ISSUE RESET
1833 007226 104005                MOV      @CSR, RCVDAT      ;GET CSR CONTENTS
1834 007230 017737 172014 001300      BEQ      1$                ;BRANCH IF RESET CLEARED ALL BITS
1835 007236 001402                ERROR1   ;ERROR! RESET DID NOT CLEAR ALL BITS
1836 007240 104011                BR       RT11A            ;LOOP ON ERROR
1837 007242 000764                MOV      #30167, @CSR      ;SET ALL R/W BITS IN CSR
1838 007244 012777 030167 171776 1$: CLR      @CSR              ;CLEAR THE CSR
1839 007252 005077 171772                MOV      @CSR, RCVDAT      ;GET & TEST CSR
1840 007256 017737 171766 001300      BEQ      2$                ;GO TO EXIT IF RESULT = 0
1841 007264 001402                ERROR1   ;ERROR! CLEAR INST. DID NOT CLEAR ALL BITS
1842 007266 104011                BR       1$                ;LOOP ERROR
1843 007270 000765                SCOPE
1844 007272 104006                SCOPE
1845                                     ;*****
1846 007274 000012 RT12: 12                ;ROUTINE # 12
1847 007276 007430      RT13                ;ADDR OF NEXT ROUTINE.
1848 007300 000012      10.                ;ITERATION COUNT
1849 007302 007304      RT12A                ;SCOPE ENTRY POINT.
1850 000012      X=X+1
1851                                     ;*****
1852                                     ;TEST THAT A BINARY COUNT CAN BE LOADED INTO A CLEAR BKCSR AND THAT
1853                                     ;A BINARY COUNT CAN BE CLEARED.
1854 RT12A: CLR      XMTDAT
1855 007304 005037 001302                MOV      XMTDAT, @BKCSR    ;LOAD BINARY COUNT INTO BKCSR
1856 007310 013777 001302 171736 1$: MOV      @BKCSR, RCVDAT    ;GET BKCSR DATA
1857 007316 017737 171732 001300      MOV      @BKCSR, RCVDAT    ;GET BKCSR DATA
1858 007324 023737 001302 001300      CMP      XMTDAT, RCVDAT    ;COMPARE DATA LOADED & DATA READ BACK
1859 007332 001405                BEQ      2$                ;BRANCH IF DATA COMPARES
1860 007334 104011                ERROR1   ;ERROR! DATA DOD NIT COMPARE
1861 007336 032777 040000 171536      BIT      #BIT14, @SWR      ;SCOPE LOOP?
1862 007344 001361                BNE      1$                ;BRANCH IF SCOPE LOOP
1863 007346 013701 001302      2$: MOV      XMTDAT, %1      ;SAVE BINARY COUNT
1864 007352 005037 001302                CLR      XMTDAT
1865 007356 005077 171672      3$: CLR      @BKCSR          ;CLEAR BKCSR AND TEST
1866 007362 017737 171666 001300      MOV      @BKCSR, RCVDAT    ;BKCSR CAN BE CLEARED

```

```

1867 007370 001405          BEQ      45          ;BRANCH IF BKCSR CLEARED
1868 007372 104011          ERROR1          ;ERROR! BKCSR DID NOT CLEAR
1869 007374 032777 040000 171500 BIT      #BIT14,JSWR ;SCOPE LOOP?
1870 007402 001365          BNE      35          ;BRANCH IF SCOPE LOOP
1871 007404 010137 001302 45:  MOV      %1,XMTDAT ;GET BINARY COUNT
1872 007410 023727 001302 177777 CMP      XMTDAT,#-1 ;ALL NUMBERS BEEN LOADED
1873 007416 001403          BEQ      55          ;GO TO EXIT
1874 007420 005237 001302 INC      XMTDAT ;INCREMENT BINARY COUNT
1875 007424 000731          BR       15          ;REPEAT TEST
1876 007426 104006          55:  SCOPE ;SCOPE
1877 ;*****
1878 007430 000013          RT13: 13 ;ROUTINE # 13 *
1879 007432 007542          RT14 ;ADDR OF NEXT ROUTINE. *
1880 007434 000144          100. ;ITERATION COUNT *
1881 007436 007440          RT13A ;SCOPE ENTRY POINT. *
1882 000013          X=X+1
1883 ;*****
1884
1885 ;TEST THAT RANDOM NUMBERS CAN BE LOADED INTO THE BKCSR
1886 007440 012702 010000          RT13A: MOV      #10000,%2 ;GET RANDOM #COUNTER
1887 007444 017737 171604 001616 15:  MOV      @BKCSR,PRVCNT ;GET PREVIOUS CONTENTS
1888 007452 004737 002712 JSR      7,@RNGEN ;GO GET A RANDOM NUMBER
1889 007456 010037 001302 MOV      %0,XMTDAT ;GET RANDOM NUMBER
1890 007462 013777 001302 171564 25:  MOV      XMTDAT,@BKCSR ;LOAD RANDOM NUMBER INTO BKCSR
1891 007470 017737 171560 001300 MOV      @BKCSR,RCVDAT ;GET BKCSR DATA
1892 007476 023737 001302 001300 CMP      XMTDAT,RCVDAT ;COMPARE DATA
1893 007504 001401          BEQ      35          ;BRANCH IF SAME
1894 007506 104011          ERROR1          ;ERROR! DATA NOT THE SAME
1895 007510 032777 040000 171364 35:  BIT      #BIT14,JSWR ;SCOPE LOOP?
1896 007516 001406          BEQ      45          ;BRANCH IF NO LOOP ON ERROR
1897 007520 005077 171530 CLR      @BKCSR ;
1898 007524 013777 001615 171522 MOV      PRVCNT,@BKCSR ;LOAD PREVIOUS CONTENTS
1899 007532 000753          BR       25          ;REPEAT TEST
1900 007534 005302          45:  DEC      %2
1901 007536 001342          BNE      15          ;BRANCH IF NOT
1902 007540 104006          55:  SCOPE ;SCOPE
1903 ;*****
1904 007542 000014          RT14: 14 ;ROUTINE # 14 *
1905 007544 007602          RT15 ;ADDR OF NEXT ROUTINE. *
1906 007546 000012          10. ;ITERATION COUNT *
1907 007550 007552          RT14A ;SCOPE ENTRY POINT. *
1908 000014          X=X+1
1909 ;*****
1910
1911 ;TEST THAT RESET CLEARS ALL BREAK STATUS REGISTER BITS
1912
1913 007552 012777 177777 171474          RT14A: MOV      #-1,@BKCSR
1914 007560 005037 001302 CLR      XMTDAT
1915 007564 104005          SRESET
1916 007566 017737 171462 001300 MOV      @BKCSR,RCVDAT
1917 007574 001401          BEQ      15
1918 007576 104011          ERROR1
1919 007600 104006          15:  SCOPE
1920 ;*****
1921 007602 000015          RT15: 15 ;ROUTINE # 15 *
1922 007604 007736          RT16 ;ADDR OF NEXT ROUTINE. *

```

```

1923 007606 000012          10.          ; ITERATION COUNT          *
1924 007610 007612          RT15A        ; SCOPE ENTRY POINT.      *
1925          000015          X=X+1
1926          ;*****
1927
1928          ;TEST THAT A BINARY COUNT CAN BE LOADED INTO A CLEAR BASREG AND THAT
1929          ;A BINARY COUNT CAN BE CLEARED.
1930 007612 005037 001302          RT15A: CLR      XMTDAT
1931 007616 013777 001302 171432 1S:  MOV      XMTDAT,2BASREG ;LOAD BINARY COUNT INTO BASREG
1932 007624 017737 171426 001300  MOV      2BASREG,RCVDAT ;GET BASREG DATA
1933 007632 023737 001302 001300  CMP      XMTDAT,RCVDAT ;COMPARE DATA
1934 007640 001405          BEQ      2S ;BRANCH IF DATA COMPARES
1935 007642 104011          ERROR1     ;ERROR! DATA DID NOT COMPARE
1936 007644 032777 040000 171230  BIT      #BIT14,2SWR ;SCOPE LOOP?
1937 007652 001361          BNE      1S ;BRANCH IF SCOPE LOOP
1938 007654 013701 001302          2S:  MOV      XMTDAT,%1 ;SAVE BINARY COUNT
1939 007660 005037 001302          CLR      XMTDAT
1940 007664 005077 171366          3S:  CLR      2BASREG
1941 007670 017737 171362 001300  MOV      2BASREG,RCVDAT
1942 007676 001405          BEQ      4S ;BRANCH IF BKCSR CLEARED
1943 007700 104011          ERROR1     ;ERROR! BKCSR DID NOT CLEAR
1944 007702 032777 040000 171172  BIT      #BIT14,2SWR ;SCOPE LOOP?
1945 007710 001365          BNE      3S ;BRANCH IF SCOPE LOOP
1946 007712 010137 001302          4S:  MOV      %1,XMTDAT ;GET BINARY COUNT
1947 007716 023727 001302 177000  CMP      XMTDAT,#177000 ;ALL NUMBERS BEEN LOADED
1948 007724 001403          BEQ      5S ;GO TO EXIT
1949 007726 105237 001303          INCB     XMTDAT+1 ;INCREMENT BINARY COUNT
1950 007732 000731          BR       1S ;REPEAT TEST
1951 007734 104006          5S:  SCOPE ;SCOPE
1952          ;*****
1953 007736 000016          RT16:  16 ;ROUTINE # 16
1954 007740 010054          RT17     ;ADDR OF NEXT ROUTINE.
1955 007742 000144          100.    ;ITERATION COUNT
1956 007744 007746          RT16A   ;SCOPE ENTRY POINT.
1957          X=X+1
1958          ;*****
1959
1960          ;TEST THAT RANDOM NUMBERS CAN BE LOADED INTO THE BASE REGISTER
1961 007746 012702 030000          RT16A: MOV      #10000,%2 ;GET RANDOM #COUNTER
1962 007752 017737 171300 001616 1S:  MOV      2BASREG,PRVCNT ;GET PREVIOUS CONTENTS
1963 007760 004737 002712          JSR      7,2RNGEN ;GO GET A RANDOM NUMBER
1964 007764 042700 000377          BIC      #000377,%0 ;CLEAR UNUSED BITS
1965 007770 010037 001302          MOV      %0,XMTDAT ;GET RANDOM NUMBER
1966 007774 013777 001302 171254 2S:  MOV      XMTDAT,2BASREG ;LOAD RANDOM NUMBER INTO BASREG
1967 010002 017737 171250 001300  MOV      2BASREG,RCVDAT ;GET BASREG DATA
1968 010010 023737 001302 001300  CMP      XMTDAT,RCVDAT ;COMPARE DATA
1969 010016 001401          BEQ      3S ;BRANCH IF SAME
1970 010020 104011          ERROR1     ;ERROR! DATA NOT THE SAME
1971 010022 032777 040000 171052 3S:  BIT      #BIT14,2SWR ;SCOPE LOOP?
1972 010030 001406          BEQ      4S ;BRANCH IF NO LOOP ON ERROR
1973 010032 005077 171220          CLR      2BASREG
1974 010036 013777 001616 171212  MOV      PRVCNT,2BASREG ;LOAD PREVIOUS CONTENTS
1975 010044 000753          BR       2S ;REPEAT TEST
1976 010046 005302          4S:  DEC      %2 ;10000 NUMBERS BEEN TESTED
1977 010050 001340          BNE      1S ;BRANCH IF NOT
1978 010052 104006          5S:  SCOPE ;SCOPE

```

1979
1980 010054 000017
1981 010056 010146
1982 010060 000144
1983 010062 010064
1984 000017
1985
1986
1987
1988 010064 013701 001252
1989 010070 012777 001106 171160
1990 010076 012700 000001
1991 010102 050011
1992 010104 020011
1993 010106 001006
1994 010110 040011
1995 010112 005711
1996 010114 005711
1997 010116 005711
1998 010120 103370
1999 010122 104006
2000 010124 010037 001302
2001 010130 011137 001300
2002 010134 104011
2003 010136 000771
2004 010140 005037 001302
2005 010144 000771
2006
2007 010146 000020
2008 010150 010234
2009 010152 000012
2010 010154 010156
2011 000020
2012
2013
2014
2015 010156 005037 001302
2016 010162 052777 177777 171062
2017 010170 104005
2018 010172 017737 171054 001300
2019 010200 001402
2020 010202 104011
2021 010204 000412
2022 010206 052777 177777 171036
2023 010214 005077 171032
2024 010220 017737 171026 001300
2025 010226 001401
2026 010230 104011
2027 010232 104006
2028
2029 010234 000021
2030 010236 010342
2031 010240 000144
2032 010242 010244
2033 000021
2034

```
*****  
RT17: 17 ;ROUTINE # 17 *  
RT20 ;ADDR OF NEXT ROUTINE. *  
100. ;ITERATION COUNT *  
RT17A ;SCOPE ENTRY POINT. *  
X=X+1  
*****  
:TEST THAT ALL BAR BITS CAN BE INDIVIDUALLY SET AND CLEARED  
RT17A: MOV BAR,%1 ;GET BAR ADDRESS *  
MOV #CAT,2BASREG ;INITIALIZE BASE REGISTER *  
MOV #1,%0 ;GET BIT TESTER *  
1S: BIS %0,(1) ;SET BAR BIT *  
CMP %0,(1) ;TEST THAT ONLY THE PROPER BAR BIT SET *  
BNE 3S ;BRANCH IF ERROR *  
BIC %0,(1) ;CLEAR BAR BIT *  
TST (1) ;TEST THAT BAR BIT CLEARED *  
BNE 5S ;BRANCH IF BAR BIT FAILED TO CLEAR *  
ASL %0 ;SHIFT BIT TESTER *  
BCC 1S  
2S: SCOPE ;SCOPE *  
3S: MOV %0,XMTDAT ;GET WHAT DATA WAS SUPPOSED TO BE *  
4S: MOV (1),RCVDAT ;GET WHAT DATA WAS *  
ERROR1 ;ERROR! IMPROPER BIT OPERATION *  
BR 2S ;GO TO SCOPE *  
5S: CLR XMTDAT ;GET WHAT DATA WAS SUPPOSED TO BE *  
BR 4S  
*****  
RT20: 20 ;ROUTINE # 20 *  
RT21 ;ADDR OF NEXT ROUTINE. *  
10. ;ITERATION COUNT *  
RT20A ;SCOPE ENTRY POINT. *  
X=X+1  
*****  
:TEST THAT RESET CLEARS ALL BAR BITS  
RT20A: CLR XMTDAT *  
BIS #-1,2BAR ;SET ALL BAR BITS *  
SRESET ;RESET *  
MOV 2BAR,RCVDAT ;GET BAR DATA *  
BEQ 1S ;BRANCH IF ALL 0'S *  
ERROR1 ;ERROR! RESET DID NOT CLEAR ALL BAR BITS *  
BR 2S ;GO TO EXIT *  
1S: BIS #-1,2BAR ;SET ALL BIT IN THE BAR *  
CLR 2BAR ;CLEAR ALL BITS IN THE BAR *  
MOV 2BAR,RCVDAT ;GET & TEST RESULT OF CLEAR OPERATION *  
BEQ 2S ;EXIT IF ALL BITS CLEARED *  
ERROR1 ;ERROR! ALL BITS DID NOT CLEAR *  
2S: SCOPE ;SCOPE *  
*****  
RT21: 21 ;ROUTINE # 21 *  
RT22 ;ADDR OF NEXT ROUTINE. *  
100. ;ITERATION COUNT *  
RT21A ;SCOPE ENTRY POINT. *  
X=X+1  
*****
```

```

2035
2036
2037 010244 012777 010100 170776 RT21A: MOV #10100, @CSR ;LOAD TEST NUMBER IN CSR
2038 010252 105077 170772 CLR B @CSR ;CLEAR EVEN BYTE
2039 010256 022777 010000 170764 CMP #10000, @CSR ;TEST THAT ONLY EVEN BYTE CLEARED
2040 010264 001410 BEQ 15
2041 010266 012737 010100 001302 MOV #10100, XMTDAT ;LOAD CORRECT RESULT
2042 010274 017737 170772 001300 MOV @CSR, RCVDAT ;GET ACTUAL RESULT
2043 010302 104011 ERROR1 ;ERROR! EVEN BYTE INSTRUCTION FAILED
2044 010304 000415 BR 25 ;GO TO SCOPE
2045 010306 012777 010100 170734 15: MOV #10100, @CSR ;LOAD TEST NUMBER IN CSR
2046 010314 105077 171302 CLR B @CSR ;TEST THAT ONLY ODD BYTE CLEARED
2047 010320 001407 BEQ 25
2048 010322 012737 000100 001302 MOV #00100, XMTDAT ;LOAD CORRECT RESULT
2049 010330 017737 170714 001300 MOV @CSR, RCVDAT ;LOAD ACTUAL RESULT
2050 010336 104011 ERROR1 ;ERROR! ODD BYTE INSTRUCTION FAILED
2051 010340 104006 25: SCOPE ;SCOPE
2052
2053 *****
2054 RT22: 22 ;ROUTINE # 22 *
2055 010344 010450 RT23 ;ADDR OF NEXT ROUTINE. *
2056 010346 000144 100. ;ITERATION COUNT *
2057 010350 010352 RT22A ;SCOPE ENTRY POINT. *
2058 000022 X=X+1
2059 *****
2060
2061 :TEST THAT BAR RESPONDS PROPERLY TO BYTE COMMANDS
2062 RT22A: MOV #10100, @BAR ;LOAD TEST NUMBER IN BAR
2063 010352 012777 010100 170672 CLR B @BAR ;CLEAR EVEN BYTE
2064 010360 105077 170666 CMP #10000, @BAR ;TEST THAT ONLY EVEN BYTE CLEARED
2065 010364 022777 010000 170660 BEQ 15
2066 010372 001410 MOV #10100, XMTDAT ;LOAD CORRECT RESULT
2067 010374 012737 010100 001302 MOV @BAR, RCVDAT ;GET ACTUAL RESULT
2068 010402 017737 170644 001300 ERROR1 ;ERROR! EVEN BYTE INSTRUCTION FAILED
2069 010410 104011 BR 25 ;GO TO SCOPE
2070 010412 000415 15: MOV #10100, @BAR ;LOAD TEST NUMBER IN BAR
2071 010414 012777 010100 170630 CLR B @BAR ;TEST THAT ONLY ODD BYTE CLEARED
2072 010422 105077 171176 BEQ 25
2073 010426 001407 MOV #00100, XMTDAT ;LOAD CORRECT RESULT
2074 010430 012737 000100 001302 MOV @CSR, RCVDAT ;LOAD ACTUAL RESULT
2075 010436 017737 170606 001300 ERROR1 ;ERROR! ODD BYTE INSTRUCTION FAILED
2076 010444 104011 25: SCOPE ;SCOPE
2077 010446 104006 *****
2078 RT23: 23 ;ROUTINE # 23 *
2079 010450 000023 RT24 ;ADDR OF NEXT ROUTINE. *
2080 010452 010556 100. ;ITERATION COUNT *
2081 010454 000144 RT23A ;SCOPE ENTRY POINT. *
2082 010456 010460 X=X+1
2083 *****
2084
2085 :TEST THAT BKCSR RESPONDS PROPERLY TO BYTE COMMANDS
2086 RT23A: MOV #10100, @BKCSR ;LOAD TEST NUMBER IN BKCSR
2087 010460 012777 010100 170566 CLR B @BKCSR ;CLEAR EVEN BYTE
2088 010466 105077 170562 CMP #10000, @BKCSR ;TEST THAT ONLY EVEN BYTE CLEARED
2089 010472 022777 010000 170554 BEQ 15
2090 010500 001410 MOV #10100, XMTDAT ;LOAD CORRECT RESULT
2091 010502 012737 010100 001302 MOV @BKCSR, RCVDAT ;GET ACTUAL RESULT
2092 010510 017737 170540 001300

```

```

2091 010516 104011          ERROR1          ;ERROR! EVEN BYTE INSTRUCTION FAILED
2092 010520 000415          BR              25          ;GO TO SCOPE
2093 010522 012777 010100 170524 15:  MOV            #010100,2BKCSR ;LOAD TEST NUMBER IN BKCSR
2094 010530 105077 171072          CLR            2,BKCSR      ;TEST THAT ONLY ODD BYTE CLEARED
2095 010534 001407          BEQ            25
2096 010536 012737 000100 001302  MOV            #000100,XMTDAT ;LOAD CORRECT RESULT
2097 010544 017737 170500 001300  MOV            2CSR,RCVDAT    ;LOAD ACTUAL RESULT
2098 010552 104011          ERROR1          ;ERROR! ODD BYTE INSTRUCTION FAILED
2099 010554 104006          25:           SCOPE        ;SCOPE
2100          ;*****
2101 010556 000024          RT24:         24          ;ROUTINE # 24
2102 010560 010622          RT25          ;ADDR OF NEXT ROUTINE.
2103 010562 000144          100          ;ITERATION COUNT
2104 010564 010566          RT24A        ;SCOPE ENTRY POINT.
2105          X=X+1
2106          ;*****
2107          ;TEST THAT OVER RUN BIT (CSR BIT13) CAUSES AN INTERRUPT WHEN SET
2108          RT24A:  MOV            #15,2XMTINT ;LOAD TRANSMITTER INTERRUPT VECTOR
2109 010566 012777 010620 170470  MOV            #BIT12,2CSR    ;SET TRANSMITTER IE BIT
2110 010574 012777 010000 170446  BIS            #BIT13,2CSR    ;SET OVER RUN BIT
2111 010632 052777 020000 170440  CLR            2,PSW         ;ENABLE INTERRUPTS
2112 010610 005037 177776          NOP
2113 010614 000240          ERROR
2114 010616 104001          ;ERROR! OVERRUN FAILED TO CAUSE AN
2115          ;INTERRUPT, OR INTERRUPTED TO INCOR-
2116          ;RECT ADDRESS
2117 010620 104006          15:           SCOPE        ;SCOPE
2118          ;*****
2119 010622 000025          RT25:         25          ;ROUTINE # 25
2120 010624 010730          RT26          ;ADDR OF NEXT ROUTINE.
2121 010626 000100          100          ;ITERATION COUNT
2122 010630 010632          RT25A        ;SCOPE ENTRY POINT.
2123          X=X+1
2124          ;*****
2125          ;TEST THAT THE DM11 INTERRUPTS AT THE CORRECT LEVEL
2126          RT25A:  MOV            #PRTY7,2,PSW
2127 010632 012737 000340 177776  MOV            #15,2XMTINT    ;LOAD TRANSMITTER INTERRUPT VECTOR
2128 010640 012777 010670 170416  MOV            #30000,2CSR    ;SET OVER RUN & IE BITS
2129 010646 012777 030000 170374  MOV            #PRTY4,2,PSW   ;ALLOW INTERRUPTS ON LEVEL 5 & ABOVE
2130 010654 012737 000200 177776  NOP
2131 010662 000240          ERROR
2132 010664 104001          ;ERROR!DM11 FAILED TO INTERRUPT
2133 010666 000417          BR              35          ;GO TO EXIT
2134 010670 022626          15:           CMP            (6)+,(6)+    ;RESET STACK POINTER
2135 010672 013737 001266 177776  MOV            XMTLVL,2,PSW   ;LOAD DM11 INTERRUPT LEVEL
2136 010700 012777 010724 170356  MOV            #25,2XMTINT    ;LOAD TRANSMITTER INTERRUPT VECTOR
2137 010706 005077 170336          CLR            2CSR
2138 010712 012777 030000 170330  MOV            #30000,2CSR
2139 010720 000240          NOP
2140 010722 000401          BR              35          ;GO TO EXIT
2141 010724 104001          25:           ERROR        ;ERROR! DM11 INTERRUPTED ON HIGHER
2142          ;PRIORITY LEVEL THAN SET FOR
2143 010726 104006          35:           SCOPE
2144          ;*****
2145 010730 000026          RT26:         26          ;ROUTINE # 26
2146 010732 010746          RT27          ;ADDRESS OF NEXT TEST.

```

```

2147 010734 000144          100.          ; ITERATION COUNT          *
2148 010736 010740          LTST0         ; SCOPE ENTRY POINT       *
2149          000026          X=X+1
2150          ; *****
2151          ; TRANSMITTER LINE TEST LINE 0
2152 010740 004537 004106  LTST0:JSR     5,XMTTST ; GO TEST TRANSMITTER LINE 0
2153 010744 000000          LINED
2154          000001          Y=Y+1
2155          ; *****
2156 010746 000027          RT27: 27      ; ROUTINE # 27            *
2157 010750 010764          RT30         ; ADDRESS OF NEXT TEST.  *
2158 010752 000144          100.         ; ITERATION COUNT        *
2159 010754 010756          LTST1        ; SCOPE ENTRY POINT      *
2160          000027          X=X+1
2161          ; *****
2162          ; TRANSMITTER LINE TEST LINE 1
2163 010756 004537 004106  LTST1:JSR     5,XMTTST ; GO TEST TRANSMITTER LINE 1
2164 010762 000002          LINE1
2165          000002          Y=Y+1
2166          ; *****
2167 010764 000030          RT30: 30      ; ROUTINE # 30            *
2168 010766 011002          RT31         ; ADDRESS OF NEXT TEST.  *
2169 010770 000144          100.         ; ITERATION COUNT        *
2170 010772 010774          LTST2        ; SCOPE ENTRY POINT      *
2171          000030          X=X+1
2172          ; *****
2173          ; TRANSMITTER LINE TEST LINE 2
2174 010774 004537 004106  LTST2:JSR     5,XMTTST ; GO TEST TRANSMITTER LINE 2
2175 011000 000004          LINE2
2176          000003          Y=Y+1
2177          ; *****
2178 011002 000031          RT31: 31      ; ROUTINE # 31            *
2179 011004 011020          RT32         ; ADDRESS OF NEXT TEST.  *
2180 011006 000144          100.         ; ITERATION COUNT        *
2181 011010 011012          LTST3        ; SCOPE ENTRY POINT      *
2182          000031          X=X+1
2183          ; *****
2184          ; TRANSMITTER LINE TEST LINE 3
2185 011012 004537 004106  LTST3:JSR     5,XMTTST ; GO TEST TRANSMITTER LINE 3
2186 011016 000006          LINE3
2187          000004          Y=Y+1
2188          ; *****
2189 011020 000032          RT32: 32      ; ROUTINE # 32            *
2190 011022 011036          RT33         ; ADDRESS OF NEXT TEST.  *
2191 011024 000144          100.         ; ITERATION COUNT        *
2192 011026 011030          LTST4        ; SCOPE ENTRY POINT      *
2193          000032          X=X+1
2194          ; *****
2195          ; TRANSMITTER LINE TEST LINE 4
2196 011030 004537 004106  LTST4:JSR     5,XMTTST ; GO TEST TRANSMITTER LINE 4
2197 011034 000010          LINE4
2198          000005          Y=Y+1
2199          ; *****
2200 011036 000033          RT33: 33      ; ROUTINE # 33            *
2201 011040 011054          RT34         ; ADDRESS OF NEXT TEST.  *
2202 011042 000144          100.         ; ITERATION COUNT        *

```

```

2203 011044 011046      LTST5           ;SCOPE ENTRY POINT      *
2204      000033      X=X+1
2205      ;*****
2206      ;TRANSMITTER LINE TEST LINE 5
2207 011046 004537 004106 LTST5:JSR      5,XMTTST      ;GO TEST TRANSMITTER LINE 5
2208 011052 000012      LINE5
2209      000006      Y=Y+1
2210      ;*****
2211 011054 000034      RT34: 34           ;ROUTINE # 34          *
2212 011056 011072      RT35           ;ADDRESS OF NEXT TEST. *
2213 011060 000144      100.          ;ITERATION COUNT      *
2214 011062 011064      LTST6         ;SCOPE ENTRY POINT    *
2215      000034      X=X+1
2216      ;*****
2217      ;TRANSMITTER LINE TEST LINE 6
2218 011064 004537 004106 LTST6:JSR      5,XMTTST      ;GO TEST TRANSMITTER LINE 6
2219 011070 000014      LINE6
2220      000007      Y=Y+1
2221      ;*****
2222 011072 000035      RT35: 35           ;ROUTINE # 35          *
2223 011074 011110      RT36           ;ADDRESS OF NEXT TEST. *
2224 011076 000144      100.          ;ITERATION COUNT      *
2225 011100 011102      LTST7         ;SCOPE ENTRY POINT    *
2226      000035      X=X+1
2227      ;*****
2228      ;TRANSMITTER LINE TEST LINE 7
2229 011102 004537 004106 LTST7:JSR      5,XMTTST      ;GO TEST TRANSMITTER LINE 7
2230 011106 000016      LINE7
2231      000010      Y=Y+1
2232      ;*****
2233 011110 000036      RT36: 36           ;ROUTINE # 36          *
2234 011112 011126      RT37           ;ADDRESS OF NEXT TEST. *
2235 011114 000144      100.          ;ITERATION COUNT      *
2236 011116 011120      LTST10        ;SCOPE ENTRY POINT    *
2237      000036      X=X+1
2238      ;*****
2239      ;TRANSMITTER LINE TEST LINE 10
2240 011120 004537 004106 LTST10:JSR     5,XMTTST      ;GO TEST TRANSMITTER LINE 10
2241 011124 000020      LINE10
2242      000011      Y=Y+1
2243      ;*****
2244 011126 000037      RT37: 37           ;ROUTINE # 37          *
2245 011130 011144      RT40           ;ADDRESS OF NEXT TEST. *
2246 011132 000144      100.          ;ITERATION COUNT      *
2247 011134 011136      LTST11        ;SCOPE ENTRY POINT    *
2248      000037      X=X+1
2249      ;*****
2250      ;TRANSMITTER LINE TEST LINE 11
2251 011136 004537 004106 LTST11:JSR     5,XMTTST      ;GO TEST TRANSMITTER LINE 11
2252 011142 000022      LINE11
2253      000012      Y=Y+1
2254      ;*****
2255 011144 000040      RT40: 40           ;ROUTINE # 40          *
2256 011146 011162      RT41           ;ADDRESS OF NEXT TEST. *
2257 011150 000144      100.          ;ITERATION COUNT      *
2258 011152 011154      LTST12        ;SCOPE ENTRY POINT    *

```

```
2259          000040          X=X+1
2260          ;*****
2261          ;TRANSMITTER LINE TEST LINE 12
2262 011154 004537 004106 LTST12:JSR      5,XMTTST      ;GO TEST TRANSMITTER LINE 12
2263 011160 000024          LINE12
2264          000013          Y=Y+1
2265          ;*****
2266 011162 000041          RT41: 41          ;ROUTINE # 41          *
2267 011164 011200          RT42          ;ADDRESS OF NEXT TEST.  *
2268 011166 000144          100.          ;ITERATION COUNT        *
2269 011170 011172          LTST13         ;SCOPE ENTRY POINT     *
2270          000041          X=X+1
2271          ;*****
2272          ;TRANSMITTER LINE TEST LINE 13
2273 011172 004537 004106 LTST13:JSR      5,XMTTST      ;GO TEST TRANSMITTER LINE 13
2274 011176 000026          LINE13
2275          000014          Y=Y+1
2276          ;*****
2277 011200 000042          RT42: 42          ;ROUTINE # 42          *
2278 011202 011216          RT43          ;ADDRESS OF NEXT TEST.  *
2279 011204 000144          100.          ;ITERATION COUNT        *
2280 011206 011210          LTST14         ;SCOPE ENTRY POINT     *
2281          000042          X=X+1
2282          ;*****
2283          ;TRANSMITTER LINE TEST LINE 14
2284 011210 004537 004106 LTST14:JSR      5,XMTTST      ;GO TEST TRANSMITTER LINE 14
2285 011214 000030          LINE14
2286          000015          Y=Y+1
2287          ;*****
2288 011216 000043          RT43: 43          ;ROUTINE # 43          *
2289 011220 011234          RT44          ;ADDRESS OF NEXT TEST.  *
2290 011222 000144          100.          ;ITERATION COUNT        *
2291 011224 011226          LTST15         ;SCOPE ENTRY POINT     *
2292          000043          X=X+1
2293          ;*****
2294          ;TRANSMITTER LINE TEST LINE 15
2295 011226 004537 004106 LTST15:JSR      5,XMTTST      ;GO TEST TRANSMITTER LINE 15
2296 011232 000032          LINE15
2297          000016          Y=Y+1
2298          ;*****
2299 011234 000044          RT44: 44          ;ROUTINE # 44          *
2300 011236 011252          RT45          ;ADDRESS OF NEXT TEST.  *
2301 011240 000144          100.          ;ITERATION COUNT        *
2302 011242 011244          LTST16         ;SCOPE ENTRY POINT     *
2303          000044          X=X+1
2304          ;*****
2305          ;TRANSMITTER LINE TEST LINE 16
2306 011244 004537 004106 LTST16:JSR      5,XMTTST      ;GO TEST TRANSMITTER LINE 16
2307 011250 000034          LINE16
2308          000017          Y=Y+1
2309          ;*****
2310 011252 000045          RT45: 45          ;ROUTINE # 45          *
2311 011254 011270          RT46          ;ADDRESS OF NEXT TEST.  *
2312 011256 000144          100.          ;ITERATION COUNT        *
2313 011260 011262          LTST17         ;SCOPE ENTRY POINT     *
2314          000045          X=X+1
```

```

2315 ;*****
2316 ;TRANSMITTER LINE TEST LINE 17
2317 011262 004537 004106 LTST17:JSR 5,XMTTST ;GO TEST TRANSMITTER LINE 17
2318 011266 000036 LINE17
2319 000020 Y=Y+1
2320 000000 Y=0
2321 000000 A=0
2322 ;*****
2323 011270 000046 RT46: 46 ;ROUTINE # 46 *
2324 011272 011306 RT47 ;ADDRESS OF NEXT TEST *
2325 011274 000144 100. ;ITERATION COUNT *
2326 011276 011300 RCV0 ;SCOPE ENTRY POINT *
2327 000046 X=X+1
2328 ;*****
2329 ;RECEIVER LINE TEST LINE 0
2330 011300 004537 004320 RCV0: JSR 5,RCVTST ;GO TEST RECEIVER LINE 0
2331 011304 000000 LINE0
2332 000001 Y=Y+1
2333 ;*****
2334 011306 000047 RT47: 47 ;ROUTINE # 47 *
2335 011310 011324 RT50 ;ADDRESS OF NEXT TEST *
2336 011312 000144 100. ;ITERATION COUNT *
2337 011314 011316 RCV1 ;SCOPE ENTRY POINT *
2338 000047 X=X+1
2339 ;*****
2340 ;RECEIVER LINE TEST LINE 1
2341 011316 004537 004320 RCV1: JSR 5,RCVTST ;GO TEST RECEIVER LINE 1
2342 011322 000002 LINE1
2343 000002 Y=Y+1
2344 ;*****
2345 011324 000050 RT50: 50 ;ROUTINE # 50 *
2346 011326 011342 RT51 ;ADDRESS OF NEXT TEST *
2347 011330 000144 100. ;ITERATION COUNT *
2348 011332 011334 RCV2 ;SCOPE ENTRY POINT *
2349 000050 X=X+1
2350 ;*****
2351 ;RECEIVER LINE TEST LINE 2
2352 011334 004537 004320 RCV2: JSR 5,RCVTST ;GO TEST RECEIVER LINE 2
2353 011340 000004 LINE2
2354 000003 Y=Y+1
2355 ;*****
2356 011342 000051 RT51: 51 ;ROUTINE # 51 *
2357 011344 011360 RT52 ;ADDRESS OF NEXT TEST *
2358 011346 000144 100. ;ITERATION COUNT *
2359 011350 011352 RCV3 ;SCOPE ENTRY POINT *
2360 000051 X=X+1
2361 ;*****
2362 ;RECEIVER LINE TEST LINE 3
2363 011352 004537 004320 RCV3: JSR 5,RCVTST ;GO TEST RECEIVER LINE 3
2364 011356 000006 LINE3
2365 000004 Y=Y+1
2366 ;*****
2367 011360 000052 RT52: 52 ;ROUTINE # 52 *
2368 011362 011376 RT53 ;ADDRESS OF NEXT TEST *
2369 011364 000144 100. ;ITERATION COUNT *
2370 011366 011370 RCV4 ;SCOPE ENTRY POINT *

```

```

2371      000052
2372      X=X+1
2373      ;*****
2374      011370 004537 004320      :RECEIVER LINE TEST LINE 4
2375      011374 000010      RCV4: JSR 5,RCVTST ;GO TEST RECEIVER LINE 4
2376      000005      LINE4
2377      Y=Y+1
2378      011376 000053      ;*****
2379      011400 011414      RT53: 53 ;ROUTINE # 53 *
2380      011402 000144      RT54 ;ADDRESS OF NEXT TEST *
2381      011404 011406      100. ;ITERATION COUNT *
2382      000053      RCV5 ;SCOPE ENTRY POINT *
2383      X=X+1
2384      ;*****
2385      011406 004537 004320      :RECEIVER LINE TEST LINE 5
2386      011412 000012      RCV5: JSR 5,RCVTST ;GO TEST RECEIVER LINE 5
2387      000006      LINE5
2388      Y=Y+1
2389      011414 000054      ;*****
2390      011416 011432      RT54: 54 ;ROUTINE # 54 *
2391      011420 000144      RT55 ;ADDRESS OF NEXT TEST *
2392      011422 011424      100. ;ITERATION COUNT *
2393      000054      RCV6 ;SCOPE ENTRY POINT *
2394      X=X+1
2395      ;*****
2396      011424 004537 004320      :RECEIVER LINE TEST LINE 6
2397      011430 000014      RCV6: JSR 5,RCVTST ;GO TEST RECEIVER LINE 6
2398      000007      LINE6
2399      Y=Y+1
2400      011432 000055      ;*****
2401      011434 011450      RT55: 55 ;ROUTINE # 55 *
2402      011436 000144      RT56 ;ADDRESS OF NEXT TEST *
2403      011440 011442      100. ;ITERATION COUNT *
2404      000055      RCV7 ;SCOPE ENTRY POINT *
2405      X=X+1
2406      ;*****
2407      011442 004537 004320      :RECEIVER LINE TEST LINE 7
2408      011446 000016      RCV7: JSR 5,RCVTST ;GO TEST RECEIVER LINE 7
2409      000010      LINE7
2410      Y=Y+1
2411      011450 000056      ;*****
2412      011452 011466      RT56: 56 ;ROUTINE # 56 *
2413      011454 000144      RT57 ;ADDRESS OF NEXT TEST *
2414      011456 011460      100. ;ITERATION COUNT *
2415      000056      RCV10 ;SCOPE ENTRY POINT *
2416      X=X+1
2417      ;*****
2418      011460 004537 004320      :RECEIVER LINE TEST LINE 10
2419      011464 000020      RCV10: JSR 5,RCVTST ;GO TEST RECEIVER LINE 10
2420      000011      LINE10
2421      Y=Y+1
2422      011466 000057      ;*****
2423      011470 011504      RT57: 57 ;ROUTINE # 57 *
2424      011472 000144      RT60 ;ADDRESS OF NEXT TEST *
2425      011474 011476      100. ;ITERATION COUNT *
2426      000057      RCV11 ;SCOPE ENTRY POINT *
2427      X=X+1

```

2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482

011476 004537 004320
011502 000022
000012
011504 000060
011506 011522
011510 000144
011512 011514
000060
011514 004537 004320
011520 000024
000013
011522 000061
011524 011540
011526 000144
011530 011532
000061
011532 004537 004320
011536 000026
000014
011540 000062
011542 011556
011544 000144
011546 011550
000062
011550 004537 004320
011554 000030
000015
011556 000063
011560 011574
011562 000144
011564 011566
000063
011566 004537 004320
011572 000032
000016
011574 000064
011576 011612
011600 000144
011602 011604
000064

```
*****  
:RECEIVER LINE TEST LINE 11  
RCV11: JSR 5,RCVTST ;GO TEST RECEIVER LINE 11  
LINE11  
Y=Y+1  
*****  
RT60: 60 ;ROUTINE # 60 *  
RT61 ;ADDRESS OF NEXT TEST *  
100. ;ITERATION COUNT *  
RCV12 ;SCOPE ENTRY POINT *  
X=X+1  
*****  
:RECEIVER LINE TEST LINE 12  
RCV12: JSR 5,RCVTST ;GO TEST RECEIVER LINE 12  
LINE12  
Y=Y+1  
*****  
RT61: 61 ;ROUTINE # 61 *  
RT62 ;ADDRESS OF NEXT TEST *  
100. ;ITERATION COUNT *  
RCV13 ;SCOPE ENTRY POINT *  
X=X+1  
*****  
:RECEIVER LINE TEST LINE 13  
RCV13: JSR 5,RCVTST ;GO TEST RECEIVER LINE 13  
LINE13  
Y=Y+1  
*****  
RT62: 62 ;ROUTINE # 62 *  
RT63 ;ADDRESS OF NEXT TEST *  
100. ;ITERATION COUNT *  
RCV14 ;SCOPE ENTRY POINT *  
X=X+1  
*****  
:RECEIVER LINE TEST LINE 14  
RCV14: JSR 5,RCVTST ;GO TEST RECEIVER LINE 14  
LINE14  
Y=Y+1  
*****  
RT63: 63 ;ROUTINE # 63 *  
RT64 ;ADDRESS OF NEXT TEST *  
100. ;ITERATION COUNT *  
RCV15 ;SCOPE ENTRY POINT *  
X=X+1  
*****  
:RECEIVER LINE TEST LINE 15  
RCV15: JSR 5,RCVTST ;GO TEST RECEIVER LINE 15  
LINE15  
Y=Y+1  
*****  
RT64: 64 ;ROUTINE # 64 *  
RT65 ;ADDRESS OF NEXT TEST *  
100. ;ITERATION COUNT *  
RCV16 ;SCOPE ENTRY POINT *  
X=X+1  
*****
```

2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
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

011604 004537 004320
011610 000034
011610 000017
011612 000065
011614 011632
011616 000144
011620 011622
011620 000065
011622 004537 004320
011626 000036
011626 000020
011630 000240
011632 000066
011634 012072
011636 000012
011640 011646
011640 000066
011642 000240
011644 000240
011646 004737 003472
011652 012701 001106
011656 012702 160000
011662 012703 000020
011666 010221
011670 005303
011672 001375
011674 012701 000001
011700 012777 012036 167356
011706 052777 000060 167334
011714 050177 167332
011720 013737 003644 011730
011726 104400
011730 000000
011732 017737 167314 001300
011740 001406
011742 005037 001302
011746 104011
011750 005077 167276
011754 000440
011756 032777 040000 167264
011764 001002
011766 104001
011770 000432
011772 042777 100000 167250
012000 052777 010000 167242

```
:RECEIVER LINE TEST LINE 16
RCV16: JSR 5,RCVTST ;GO TEST RECEIVER LINE 16
        LINE16
        Y=Y+1
:*****
RT65: 65 ;ROUTINE # 65 *
      RT66 ;ADDRESS OF NEXT TEST *
      100. ;ITERATION COUNT *
      RCV17 ;SCOPE ENTRY POINT *
      X=X+1
:*****
:RECEIVER LINE TEST LINE 17
RCV17: JSR 5,RCVTST ;GO TEST RECEIVER LINE 17
        LINE17
        Y=Y+1
        NOP
:*****
RT66: 66 ;ROUTINE # 66 *
      RT67 ;ADDR OF NEXT ROUTINE. *
      10. ;ITERATION COUNT *
      RT66A ;SCOPE ENTRY POINT. *
      X=X+1
:*****
        NOP
        NOP
;TEST THAT NEX BIT (CSR BIT 14) SETS WHEN THE TRANSMITTER REFERENCES
;NON-EXISTANT MEMORY, THE CORRESPONDING BAR BIT CLEARS
;AND THAT AN INTERRUPT OCCURS. ALL LINES ARE USED FOR THE TEST.
RT66A: JSR 7,TIMER ;GO CALCULATE MACHINE TIME TO TRANSMIT
        ;ONE CHARACTER
        MOV #CAT,%1 ;GET CAT ADDRESS
        MOV #160000,%2 ;GET A NON-EXISTANT ADDRESS
        MOV #16,%3 ;GET COUNTER
1S: MOV %2,(1)+ ;LOAD THE CURRENT ADDRESS
     DEC %3 ;TABLE WITH NON-EXISTANT
     BNE 1S ;ADDRESSES
     MOV #LBIT0,%1 ;GET LINE BIT
     MOV #65,%XMTINT ;LOAD TRANSMITTER INT. VECTOR
2S: BIS #60,%CSR ;SET EXTENDED ADDRESS BITS
     BIS %1,%BAR ;START TRANSMITTER
     MOV TIME14,%3S ;LOAD DELAY TIME TO
     DELAY ;DELAY FOR 1/4TH OF A CHARACTER
3S: OPEN ;TO RESPOND TO NEX
     MOV %BAR,RCV DAT ;GET BAR DATA & TEST
     BEQ 4S ;THAT IT IS CLEAR
     CLR XMT DAT
     ERROR1 ;ERROR!BAR BIT DID NOT CLEAR
     CLR %BAR
     BR 7S
4S: BIT #BIT14,%CSR ;GO TO SCOPE
     BNE 5S ;TEST THAT NEX BIT IS SET
     ERROR ;BRANCH IF SET
     ERROR ;ERROR! NEX BIT FAILED TO SET
     BR 7S ;GO TO SCOPE
5S: BIC #BIT15,%CSR ;CLEAR TRANSMITTER READY FLAG
     BIS #BIT12,%CSR ;SET TRANSMITTER IE BIT
```

```

2539 012006 005037 177776          CLR      2#PSW          ;ALLOW INTERRUPTS
2540 012012 000240          NOP
2541 012014 012737 000340 177776  MOV      #PRTY7,2#PSW ;LOCK OUT INTERRUPTS
2542 012022 010137 001302          MOV      %1,XMTDAT    ;LOAD LINE THAT FAILED
2543 012026 005037 001300          CLR      RCVDAT
2544 012032 104011          ERROR1                ;ERROR! NEX FAILED TO CAUSE INTERRUPT
2545                                ;TYPEOUT SHOWS LINE # THAT FAILED
2546                                ;GO TO SCOPE
2547 012034 000410          BR       7$
2548 012036 005077 167206 6$:      CLR      2CSR
2549 012042 012737 000340 177776  MOV      #PRTY7,2#PSW ;LOCK OUT INTERRUPTS
2550 012050 022626          CMP      (6)+,(6)+    ;ADJUST STACK PTR
2551 012054 103314          ASL     %1            ;SHIFT LINE BIT
2552 012056 013737 003642 012066 7$:  MOV      TIME1,8$     ;WAIT FOR TRANSMITTER TO RUN
2553 012064 104400          DELAY                                ;TO COMPLETION BEFORE
2554 012066 000000 8$:      OPEN                                ;EXITING TEST
2555 012070 104006          SCOPE                  ;SCOPE
2556                                ;*****
2557 012072 000067          RT67:  67                ;ROUTINE # 67 *
2558 012074 012202          RT70                ;ADDR OF NEXT ROUTINE. *
2559 012076 000012          10.                ;ITERATION COUNT *
2560 012100 012102          RT67A                ;SCOPE ENTRY POINT. *
2561 000067          X=X+1
2562                                ;*****
2563                                ;TEST THAT NEX BIT SETS IF THE DM11 TABLES ARE IN NON-EXISTANT CORE
2564                                ;*****
2565 012102 012777 160000 167146  RT67A:  MOV      #160000,2BASREG ;SET BASE REGISTER TO NON-EXISTANT ADRS.
2566 012110 012737 012172 000004  MOV      #4$,2ERRVEC ;SET TIME OUT TRAP VECTOR
2567 012116 005737 160000          TST     2#160000    ;CHECK THAT ADDRESS TIMES OUT
2568 012122 013737 003644 012140  MOV      TIME14,1$   ;GET TIME TO TRANSMIT 1/4 CHAR.
2569 012130 052777 000001 167114  BIS     #LBITO,2BAR  ;START TO TRANSMIT ON LINE 0
2570 012136 104400          DELAY                                ;DELAY 1/4TH OF A CHARACTER
2571 012140 000000 1$:      OPEN                                ;TIME
2572 012142 005077 167104          CLR     2BAR         ;STOP TRANSMITTER
2573 012146 022777 040060 167074  CMP     #BIT14+60,2CSR ;TEST THAT ONLY NEX IS SET
2574 012154 001401          BEQ    2$
2575 012156 104001          ERROR                ;ERROR! EITHER NEX FALED TO SET
2576                                ;OR OTHER BITS SET
2577 012160 013737 003642 012170 2$:  MOV      TIME1,3$   ;DELAY 1 CHARACTER TIME TO ALLOW
2578 012166 104400          DELAY                                ;TRANSMITTER TO RUN TO
2579 012170 000000 3$:      OPEN                                ;COMPLETION
2580 012172 012737 000006 000004 4$:  MOV      #ERRVEC+2,2#ERRVEC ;RESTORE TIME OUT TRAP
2581 012200 104006          SCOPE
2582                                ;*****
2583 012202 000070          RT70:  70                ;ROUTINE # 70 *
2584 012204 012316          RT71                ;ADDR OF NEXT ROUTINE. *
2585 012206 000144          100.                ;ITERATION COUNT *
2586 012210 012212          RT70A                ;SCOPE ENTRY POINT. *
2587 000070          X=X+1
2588                                ;*****
2589                                ;TEST THAT WHEN THE GO BIT IS CLEAR THAT THE RECEIVERS DO NOT RECEIVE
2590                                ;DATA.EACH LINE IN TURN IS TRANSMITTED ON,AND WHEN TEN CHARACTERS
2591                                ;HAVE BEEN TRANSMITTED THE RECEIVER DONE FLAG IS TESTED. IF IT IS SET
2592                                ;AN ERROR IS INDICATED ON THE LINE DATA WAS RECEIVED ON.
2593                                ;*****
2594 012212 005037 016176  RT70A:  CLR      LINE          ;SET UP TO TRANSMIT

```

```

2595 012216 004537 005532 1S: JSR 5,2#XMITD ;10. CHARACTERS
2596 012222 177766 -10. ;ON EACH LINE
2597 012224 005777 167020 TST 2CSR ;WAIT FOR 10. CHARACTERS
2598 012230 100375 BPL .-4 ;TO BE TRANSMITTED
2599 012232 042777 100000 167010 BIC #100000,2CSR
2600 012240 105777 167004 TSTB 2CSR ;TEST RECEIVER DONE FLAG
2601 012244 100010 BPL 2S
2602 012246 013737 001272 001300 MOV LINBIT,RCV DAT ;GET LINE BIT OF ACTIVE LINE
2603 012254 013737 001272 001302 MOV LINBIT,XMT DAT ;THAT ERROR OCCURED ON
2604 012262 104011 ERROR1 ;ERROR! DATA WAS RECEIVED ON LINE INDICATED
2605 012264 000413 BR 4S ;GO TO SCOPE
2606 012266 062737 000002 016176 2S: ADD #2,LINE ;SET UP NEXT LINE NUMBER
2607 012274 006337 001272 ASL LINBIT ;GET READY TO TRANSMIT ON NEXT LINE
2608 012300 103346 BCC 1S ;GO TRANSMIT ON NEXT LINE
2609 012302 013737 003642 012312 MOV TIME1,3S
2610 012310 104400 DELAY ;DELAY 1 CHARACTER
2611 012312 000000 3S: 0 ;TIME BEFORE ENTERING NEXT TEST
2612 012314 104006 4S: SCOPE ;SCOPE
2613 *****
2614 012316 000071 RT71: 71 ;ROUTINE # 71 *
2615 012320 012512 RT72 ;ADDR OF NEXT ROUTINE. *
2616 012322 000024 20. ;ITERATION COUNT *
2617 012324 012326 RT71A ;SCOPE ENTRY POINT. *
2618 000071 X=X+1
2619 *****
2620
2621 ;TEST THAT CURRENT ADDRESS INCREMENTS PROPERLY WHEN A CHAR-
2622 ;ACTER IS TRANSMITTED. LINE 0 IS USED FOR THE TEST.
2623 012326 005000 RT71A: CLR %0 ;R0=CURRENT ADRS AFTER TRANSMISSION
2624 012330 010001 1S: MOV %0,%1 ;R0=CURRENT ADDRESS BEFORE TRANSMISSION
2625 012332 005201 INC %1 ;AND R1=CURRENT ADDRESS AFTER TRANSMISSION
2626 012334 012737 012476 000004 MOV #3S,2#ERRVEC ;SET UP PROCESSOR
2627 012342 012737 000340 000006 MOV #PRTY7,2#ERRVEC+2 ;TIME OUT TRAP
2628 012350 012777 001106 166700 MOV #CAT,2BASREG ;SET UP BASE REGISTER
2629 012356 010037 001106 MOV %0,CAT ;LOAD CURRENT ADDRESS TABLE (LINE 0)
2630 012362 105710 TSTB (0) ;DOES MEMORY EXIST?
2631 012364 012737 177776 001146 MOV #-2,WCT ;SET CHAR. COUNT TO TRANSMIT 1 CHAR.
2632 012372 012777 000005 166650 MOV #5,2CSR ;SET MAINT & GO BITS
2633 012400 012777 000001 166644 MOV #LBITO,2BAR ;TRANSMIT ON LINE 0
2634 012406 105777 166636 TSTB 2CSR ;WAIT FOR THE RECEIVER
2635 012412 100375 BPL .-4 ;TO RECEIVE FIRST CHARACTER
2636 012414 042777 000200 166626 BIC #200,2CSR ;CLEAR RECEIVER DONE FLAG
2637 012422 105777 166622 TSTB 2CSR ;WAIT FOR RECEIVER TO RECEIVE
2638 012426 100375 BPL .-4 ;THE SECOND CHARACTER
2639 012430 023701 001106 CMP CAT,%1 ;TEST THAT CURRENT ADRS
2640 ;INCREMENTED PROPERLY
2641 012434 001413 BEQ 2S
2642 012436 010137 001302 MOV %1,XMT DAT ;GET COMPUTED RESULT
2643 012442 013737 001106 001300 MOV CAT,RCV DAT ;GET ACTUAL RESULT
2644 012450 104011 ERROR1 ;ERROR! CURRENT ADDRESS DID NOT
2645 012452 032777 040000 166422 BIT #BIT14,2SWR ;INCREMENT PROPERLY
2646 012460 001323 BNE 1S ;BRANCH IF SCOPE SWITCH IS SET
2647 012462 000405 BR 3S ;GO TO EXIT
2648 012464 005701 2S: TST %1
2649 012466 001403 BEQ 3S
2650 012470 000261 SEC

```

```

2651 012472 006100          ROL    %0
2652 012474 100715          BMI    1$
2653 012476 012737 000006 000004 3$:  MOV    #ERRVEC+2, @#ERRVEC    ;RESTORE TIME OUT TRAP
2654 012504 005037 000006          CLR    @#ERRVEC+2
2655 012510 104006          SCOPE          ;SCOPE
2656          ;*****
2657 012512 000072 RT72:  72          ;ROUTINE # 72          *
2658 012514 013060          RT73          ;ADDR OF NEXT ROUTINE.  *
2659 012516 000024          20.          ;ITERATION COUNT        *
2660 012520 012522          RT72A        ;SCOPE ENTRY POINT.    *
2661          000072          X=X+1
2662          ;*****
2663
2664          ;TEST THAT DATA CAN BE TRANSMITTED FROM ALL AVAILABLE CORE.
2665          ;LINE 0 IS USED FOR THE TEST AND ONLY ONE WORD IS TRANSMITTED
2666          ;AT A TIME.
2667 012522 005000 RT72A:  CLR    %0          ;CLEAR INDEX REGISTER
2668 012524 000005          RESET
2669 012526 016037 013014 001106 1$:  MOV    AREA(0), CAT    ;LOAD CURRENT ADDRESS
2670 012534 012777 000005 166506  MOV    #5, @CSR        ;SET MAINT & GO BITS
2671 012542 012737 177777 001146  MOV    #-1, WCT        ;SET UP CHAR COUNT TO TRANSMIT 1 CHAR.
2672 012550 012777 000001 166474  MOV    @LBIT0, @BAR    ;TRANSMIT CHAR ON LINE 0
2673 012556 005777 166466          TST    @CSR            ;WAIT FOR THE TRANSMITTER
2674 012562 100375          BPL    -4             ;TO TRANSMIT THE CHARACTER
2675 012564 105777 166460          TSTB   @CSR            ;TEST FOR DONE
2676 012570 100375          BPL    -4
2677 012572 005077 166452          CLR    @CSR            ;CLEAR ALL FLAGS
2678 012576 005037 001300          CLR    RCVDAT
2679 012602 113737 001306 001300  MOVB   TUMTAB, RCVDAT  ;GET RECEIVED CHARACTER
2680 012610 117037 013014 001302  MOVB   @AREA(0), XMTDAT ;GET TRANSMITTED CHARACTER
2681 012616 043737 001304 001302  BIC    CARMSK, XMTDAT  ;CLEAR NON-TRANSMITTED BITS
2682 012624 123737 001300 001302  CMPB   RC/DAT, XMTDAT  ;COMPARE CHARACTERS
2683 012632 001402          BEQ    2$             ;BRANCH IF VALID COMPARISON
2684 012634 104011          ERROR1            ;ERROR! DATA COMPARISON ERROR
2685          ;(CAT)-1 IS THE MEMORY LOCATION WHERE THE DATA WAS TRANSMITTED FROM
2686 012636 000465          BR     6$
2687 012640 020027 000006 2$:  CMP    %0, #6         ;GO TO EXIT
2688 012644 001402          BEQ    3$             ;HAS FIRST 4K BEEN TESTED
2689 012646 005720          TST    (0)+          ;BRANCH IF IT HAS
2690 012650 000726          BR     1$             ;INCREMENT INDEX
2691
2692
2693 012652 000240 3$:  NOP
2694 012654 012737 013002 000004  MOV    #5$, @#ERRVEC  ;BEGIN TESTING ABOVE 4K
2695
2696 012662 005001          CLR    %1
2697 012664 005201 4$:  INC    %1            ;SET TIME OUT TRAP TO EXIT
2698 012666 005720          TST    (0)+          ;TEST IF MEMORY TIMES OUT
2699 012670 110170 013014 001106  MOVB   %1, @AREA(0)    ;SET UP DATA IDENTIFIER
2700 012674 016037 013014 001106  MOV    AREA(0), CAT    ;INCREMENT DATA IDENTIFIER
2701 012702 012777 000005 166340  MOV    #5, @CSR        ;INCREMENT INDEX
2702 012710 012737 177777 001146  MOV    #-1, WCT        ;LOAD IDENTIFIER INTO MEMORY
2703 012716 012777 000001 166326  MOV    @LBIT0, @BAR    ;LOAD CURRENT ADDRESS
2704 012724 005777 166320          TST    @CSR            ;SET MAINT & GO BITS
2705 012730 100375          BPL    -4             ;SET UP CHAR COUNT TO TRANSMIT 1 CHAR.
2706 012732 105777 166312          TSTB   @CSR            ;TRANSMIT ON LINE 0
          ;WAIT FOR THE TRANSMITTER TO
          ;TRANSMIT THE CHARACTER
          ;TEST FOR CHARACTER DONE

```

```

2707 012736 100375          BPL      -4
2708 012740 005077 166304  CLR      @CSR
2709 012744 113737 001306 001300  MOVB   TUMTAB,RCV DAT ;GET THE RECEIVED CHARACTER
2710 012752 117037 013014 001302  MOVB   @AREA(0),XMT DAT ;GET THE TRANSMITTED CHARACTER
2711 012760 043737 001304 001302  BIC    CARMSK,XMT DAT ;CLEAR NON-TRANSMITTED BITS
2712 012766 123737 001300 001302  CMPB   RCV DAT,XMT DAT ;COMPARE CHARACTERS
2713 012774 001733          BEQ     45 ;BRANCH IF VALID COMPARISON
2714 012776 104011          ERROR1  ;ERROR! DATA COMPARISON ERROR NUMBER
2715 013000 000404          BR      65 ;IN S/B GIVES MEMORY LOCATION (SEE TABLE)
2716 013002 022626          POPSP2 ;RESET THE STACK
2717 013004 012737 000006 000004 5S:    MOV    @6,@ERRVEC ;RESTORE TIME OUT TRAP
2718 013012 104006          SCOPE   ;EXIT TEST
2719                                     ;MEMORY LOCATIONS TRANSMITTED FROM TABLE
2720 013014 000000          AREA:  0 ;FOR DATA IN FIRST
2721 013016 005252          5252   ;4K SEE THE LISTING
2722 013020 012525          12525  ;CONTENTS OF THESE LOCATIONS (BYTE)
2723 013022 017777          17777  ;IS THE DATA TRANSMITTED
2724 013024 020000          20000  ;CONTENTS =1 (IF AVAILABLE)
2725 013026 026314          26314  ;
2726 013030 031463          31463  ;
2727                                     ;
2728 013032 037477          37477  ;
2729 013034 040000          A12K:  40000 ;
2730 013036 057477          57477  ;
2731 013040 060000          A16K:  60000 ;
2732 013042 077477          77477  ;
2733 013044 100000          A20K: 100000 ;
2734 013046 117477          117477 ;
2735 013050 120000          A24K: 120000 ;
2736 013052 137477          137477 ;
2737 013054 140000          A28K: 140000 ;
2738 013056 173000          173000 ;
2739                                     ;
2740 013060 000073          RT73:  73 ;ROUTINE # 73
2741 013062 013274          RT74   ;ADDR OF NEXT ROUTINE.
2742 013064 000012          10.   ;ITERATION COUNT
2743 013066 013070          RT73A ;SCOPE ENTRY POINT.
2744 000073          X=X+1
2745                                     ;
2746                                     ;
2747 ;TEST THAT THE TRANSMITTER CAN TRANSMIT 100. CHARACTERS BEFORE SETTING
2748 ;THE READY BIT (CSR 15),AND CLEARING THE BAR BIT.
2749 013070 005037 016176          RT73A: CLR    LINE
2750 013074 012777 000001 166146 1S:    MOV    @1,@CSR ;SET THE GO BIT
2751 013102 005037 001300          CLR    RCV DAT
2752 013106 013737 016176 001302  MOV    @LINE,XMT DAT ;GET LINE NUMBER (X2)
2753 013114 004537 005532          JSR    5,@XMITD ;TRANSMIT 100. CHARACTERS
2754 013120 177634          -100. ;ON LINE AS SPECIFIED BY LINE
2755 013122 105777 166122          2S:    TSTB  @CSR ;WAIT FOR THE RECEIVER
2756 013126 100375          BPL    25 ;TO RECEIVE ONE CHARACTER
2757 013130 042777 000200 166112  BIC    @BIT7,@CSR ;CLEAR CHAR. DONE FLAG
2758 013136 005237 001300          INC    RCV DAT ;INCREMENT CHAR. RCVD COUNT
2759 013142 023727 001300 000144  CMP    RCV DAT,@100. ;HAVE 100. CHARS. BEEN RCVD
2760 013150 001416          BEQ    45
2761 013152 005777 166072          TST   @CSR ;TEST READY FLAG
2762 013156 100002          BPL    35 ;GO TEST BAR

```

```

2763 013160 104011          ERROR1          ;ERROR!READY BIT SET TOO SOON
2764          ;TYPEOUT SHOWS HOW MANY CHARS WERE RECEIVED WHEN READY SET AND THE LINE # (X2)
2765 013162 000443          BR          BS          ;GO TO EXIT
2766
2767 013164 023777 001272 166060 3$:  CMP          2#LINBIT,2BAR ;TEST THAT BAR BIT IS SET
2768 013172 001753          BEQ          2$          ;BRANCH IF SET
2769 013174 017737 166052 001302          MOV          2BAR,XMTDAT ;GET BAR CONTENTS
2770 013202 104011          ERROR1          ;ERROR! BAR BIT CLEARED TOO SOON
2771          ;TYPEOUT SHOWS THE BAR CONTENTS AND HOW MANY CHARS WERE RECEIVED WHEN BAR FAILED
2772          ;LOCATION LINBIT HAS THE CORRECT BAR CONTENTS.
2773 013204 000432          BR          BS          ;EXIT TEST
2774 013206 013737 003644 013216 4$:  MOV          TIME14,5$ ;DELAY 1/4 CHARACTER TIME
2775 013214 104400          DELAY          ;TO ALLOW TRANSMITTER TO FINISH
2776 013216 000000          5$:  OPEN
2777 013220 005777 166024          TST          2CSR          ;TEST READY FLAG (SHOULD BE SET)
2778 013224 100402          BMI          6$          ;GO TEST BAR
2779 013226 104001          ERROR          ;ERROR! READY FLAG FAILED TO SET
2780 013230 000420          BR          BS          ;GO TO EXIT
2781 013232 005777 166014          6$:  TST          2BAR          ;TEST THAT BAR BIT IS CLEAR
2782 013236 001407          BEQ          7$          ;GO TO 7$ IF CLEAR
2783 013240 017737 166006 001300          MOV          2BAR,RCVDAT
2784 013246 005037 001302          CLR          XMTDAT
2785 013252 104011          ERROR1          ;ERROR! BAR BIT FAILED TO CLEAR
2786 013254 000406          BR          BS
2787 013256 062737 000002 016176 7$:  ADD          2,2#LINE
2788 013264 006337 001272          ASL          2#LINBIT
2789 013270 103301          BCC          1$
2790 013272 104006          8$:  SCOPE
2791          ;*****
2792 013274 000074          RT74:  74          ;ROUTINE # 74
2793 013276 013462          RT75          ;ADDR OF NEXT ROUTINE.
2794 013300 000012          10.          ;ITERATION COUNT
2795 013302 013304          RT74A          ;SCOPE ENTRY POINT.
2796 000074          X=X+1
2797          ;*****
2798
2799          ;TEST THAT THE TUMBLE TABLE POINTER INCREMENTS PROPERLY AND
2800          ;RETURNS TO THE BEGINNING AFTER 64. CHARACTERS HAVE BEEN RECEIVED
2801          ;LINE 0 IS USED FOR THE TEST.
2802 013304 012701 001306          RT74A:  MOV          2TUMTAB,%1 ;CLEAR THE
2803 013310 012702 000100          MOV          2#64,%2 ;TUMBLE TABLE
2804 013314 005021          1$:  CLR          (1)+
2805 013316 005302          DEC          %2
2806 013320 001375          BNE          1$
2807 013322 012701 001306          MOV          2TUMTAB,%1
2808 013326 012777 000004 165714          MOV          2BIT2,2CSR ;SET MAINT BIT & CLEAR GO BIT
2809 013334 005037 001302          CLR          XMTDAT
2810 013340 005037 001300          CLR          RCVDAT
2811 013344 012737 177677 001146          MOV          2-65,%CT ;SET UP TO TRANSMIT 65. CHARACTERS
2812 013352 052777 000001 165670          BIS          2BIT0,2CSR ;SET THE GO BIT
2813 013360 012777 000001 165664          MOV          2LBIT0,2BAR ;TRANSMIT ON LINE 0
2814 013366 105777 165656          2$:  TSTB          2CSR ;WAIT FOR CHAR. DONE FLAG
2815 013372 100375          BPL          2$
2816 013374 042777 000200 165646          BIC          2BIT7,2CSR ;CLEAR CHAR. DONE FLAG
2817 013402 005237 001300          INC          RCVDAT ;INCREMENT CHARACTERS
2818 013406 005237 001302          INC          XMTDAT ;RECEIVED COUNT

```

```

2819 013412 005711          TST      (1)          ;TEST TT ENTRY FOR VALID
2820 013414 100402          BMI      35          ;DATA ENTRY
2821 013416 104011          ERROR1          ;ERROR! NO VALID DATA ENTRY
2822                                     ;TYPEOUT SHOWS # OF CHARS. RCVD WHEN ERROR OCCURED
2823 013420 000417          BR       45          ;GO TO SCOPE
2824 013422 005021          CLR      (1)+       ;CLEAR TT ENTRY
2825 013424 023727 001302 000100 3S:  CMP      XMTDAT,#64. ;HAVE 64. CHARACTERS BEEN RECEIVED
2826 013432 001355          BNE      25          ;
2827 013434 005777 165610          TST      @CSR       ;WAIT FOR THE LAST CHARACTER
2828 013440 100375          BPL      -4         ;TO BE TRANSMITTED
2829 013442 105777 165602          TSTB    @CSR       ;TEST FOR DONE
2830 013446 100375          BPL      -4         ;
2831 013450 005737 001306          TST      TUMTAB     ;TEST FIRST TT ENTRY
2832 013454 100401          BMI      45          ;FOR VALID DATA
2833 013456 104001          ERROR          ;ERROR! POINTER DID NOT RETURN
2834 013460 104006          4S:  SCOPE          ;SCOPE
2835                                     A=0
2836                                     Y=0
2837                                     ;*****
2838 013462 000075          RT75:  75          ;ROUTINE # 75
2839 013464 013500          RT76          ;ADDRESS OF NEXT TEST
2840 013466 000144          100.         ;ITERATION COUNT
2841 013470 013472          BRK0          ;SCOPE ENTRY POINT
2842                                     X=X+1
2843                                     ;*****
2844                                     ;BREAK TEST ON LINE 0.
2845 013472 004537 004742          BRK0:  JSR      5,BRKTST ;GO DO BREAK TEST
2846 013476 000001          LBIT0          ;ON LINE 0
2847                                     Y=Y+1
2848                                     ;*****
2849 013500 000076          RT76:  76          ;ROUTINE # 76
2850 013502 013516          RT77          ;ADDRESS OF NEXT TEST
2851 013504 000144          100.         ;ITERATION COUNT
2852 013506 013510          BRK1          ;SCOPE ENTRY POINT
2853                                     X=X+1
2854                                     ;*****
2855                                     ;BREAK TEST ON LINE 1.
2856 013510 004537 004742          BRK1:  JSR      5,BRKTST ;GO DO BREAK TEST
2857 013514 000002          LBIT1          ;ON LINE 1
2858                                     Y=Y+1
2859                                     ;*****
2860 013516 000077          RT77:  77          ;ROUTINE # 77
2861 013520 013534          RT100         ;ADDRESS OF NEXT TEST
2862 013522 000144          100.         ;ITERATION COUNT
2863 013524 013526          BRK2          ;SCOPE ENTRY POINT
2864                                     X=X+1
2865                                     ;*****
2866                                     ;BREAK TEST ON LINE 2.
2867 013526 004537 004742          BRK2:  JSR      5,BRKTST ;GO DO BREAK TEST
2868 013532 000004          LBIT2          ;ON LINE 2
2869                                     Y=Y+1
2870                                     ;*****
2871 013534 000100          RT100: 100        ;ROUTINE # 100
2872 013536 013552          RT101         ;ADDRESS OF NEXT TEST
2873 013540 000144          100.         ;ITERATION COUNT
2874 013542 013544          BRK3          ;SCOPE ENTRY POINT

```

```

2875          000100          X=X+1
2876          ;*****
2877          ;BREAK TEST ON LINE 3.
2878 013544 004537 004742 BRK3: JSR      5,BRKTST      ;GO DO BREAK TEST
2879 013550 000010          LBIT3      ;ON LINE 3
2880          Y=Y+1
2881          ;*****
2882 013552 000101 RT101: 101      ;ROUTINE # 101      *
2883 013554 013570          RT102      ;ADDRESS OF NEXT TEST *
2884 013556 000144          100.      ;ITERATION COUNT    *
2885 013560 013562          BRK4      ;SCOPE ENTRY POINT  *
2886          X=X+1
2887          ;*****
2888          ;BREAK TEST ON LINE 4.
2889 013562 004537 004742 BRK4: JSR      5,BRKTST      ;GO DO BREAK TEST
2890 013566 000020          LBIT4      ;ON LINE 4
2891          Y=Y+1
2892          ;*****
2893 013570 000102 RT102: 102      ;ROUTINE # 102      *
2894 013572 013606          RT103      ;ADDRESS OF NEXT TEST *
2895 013574 000144          100.      ;ITERATION COUNT    *
2896 013576 013600          BRK5      ;SCOPE ENTRY POINT  *
2897          X=X+1
2898          ;*****
2899          ;BREAK TEST ON LINE 5.
2900 013600 004537 004742 BRK5: JSR      5,BRKTST      ;GO DO BREAK TEST
2901 013604 000040          LBIT5      ;ON LINE 5
2902          Y=Y+1
2903          ;*****
2904 013606 000103 RT103: 103      ;ROUTINE # 103      *
2905 013610 013624          RT104      ;ADDRESS OF NEXT TEST *
2906 013612 000144          100.      ;ITERATION COUNT    *
2907 013614 013616          BRK6      ;SCOPE ENTRY POINT  *
2908          X=X+1
2909          ;*****
2910          ;BREAK TEST ON LINE 6.
2911 013616 004537 004742 BRK6: JSR      5,BRKTST      ;GO DO BREAK TEST
2912 013622 000100          LBIT6      ;ON LINE 6
2913          Y=Y+1
2914          ;*****
2915 013624 000104 RT104: 104      ;ROUTINE # 104      *
2916 013626 013642          RT105      ;ADDRESS OF NEXT TEST *
2917 013630 000144          100.      ;ITERATION COUNT    *
2918 013632 013634          BRK7      ;SCOPE ENTRY POINT  *
2919          X=X+1
2920          ;*****
2921          ;BREAK TEST ON LINE 7.
2922 013634 004537 004742 BRK7: JSR      5,BRKTST      ;GO DO BREAK TEST
2923 013640 000200          LBIT7      ;ON LINE 7
2924          Y=Y+1
2925          ;*****
2926 013642 000105 RT105: 105      ;ROUTINE # 105      *
2927 013644 013660          RT106      ;ADDRESS OF NEXT TEST *
2928 013646 000144          100.      ;ITERATION COUNT    *
2929 013650 013652          BRK10     ;SCOPE ENTRY POINT  *
2930          X=X+1

```

2931
2932
2933 013652 004537 004742
2934 013656 000400
2935 000011
2936
2937 013660 000106
2938 013662 013676
2939 013664 000144
2940 013666 013670
2941 000106
2942
2943
2944 013670 004537 004742
2945 013674 001000
2946 000012
2947
2948 013676 000107
2949 013700 013714
2950 013702 000144
2951 013704 013706
2952 000107
2953
2954
2955 013706 004537 004742
2956 013712 002000
2957 000013
2958
2959 013714 000110
2960 013716 013732
2961 013720 000144
2962 013732 013724
2963 000110
2964
2965
2966 013724 004537 004742
2967 013730 004000
2968 000014
2969
2970 013732 000111
2971 013734 013750
2972 013736 000144
2973 013740 013742
2974 000111
2975
2976
2977 013742 004537 004742
2978 013746 010000
2979 000015
2980
2981 013750 000112
2982 013752 013766
2983 013754 000144
2984 013756 013760
2985 000112
2986

```
*****  
:BREAK TEST ON LINE 10.  
BRK10: JSR      5,BRKTST      ;GO DO BREAK TEST  
        LBIT10                ;ON LINE 10  
        Y=Y+1  
*****  
RT106: 106                    ;ROUTINE # 106 *  
        RT107                  ;ADDRESS OF NEXT TEST *  
        100.                   ;ITERATION COUNT *  
        BRK11                  ;SCOPE ENTRY POINT *  
        X=X+1  
*****  
:BREAK TEST ON LINE 11.  
BRK11: JSR      5,BRKTST      ;GO DO BREAK TEST  
        LBIT11                ;ON LINE 11  
        Y=Y+1  
*****  
RT107: 107                    ;ROUTINE # 107 *  
        RT110                  ;ADDRESS OF NEXT TEST *  
        100.                   ;ITERATION COUNT *  
        BRK12                  ;SCOPE ENTRY POINT *  
        X=X+1  
*****  
:BREAK TEST ON LINE 12.  
BRK12: JSR      5,BRKTST      ;GO DO BREAK TEST  
        LBIT12                ;ON LINE 12  
        Y=Y+1  
*****  
RT110: 110                    ;ROUTINE # 110 *  
        RT111                  ;ADDRESS OF NEXT TEST *  
        100.                   ;ITERATION COUNT *  
        BRK13                  ;SCOPE ENTRY POINT *  
        X=X+1  
*****  
:BREAK TEST ON LINE 13.  
BRK13: JSR      5,BRKTST      ;GO DO BREAK TEST  
        LBIT13                ;ON LINE 13  
        Y=Y+1  
*****  
RT111: 111                    ;ROUTINE # 111 *  
        RT112                  ;ADDRESS OF NEXT TEST *  
        100.                   ;ITERATION COUNT *  
        BRK14                  ;SCOPE ENTRY POINT *  
        X=X+1  
*****  
:BREAK TEST ON LINE 14.  
BRK14: JSR      5,BRKTST      ;GO DO BREAK TEST  
        LBIT14                ;ON LINE 14  
        Y=Y+1  
*****  
RT112: 112                    ;ROUTINE # 112 *  
        RT113                  ;ADDRESS OF NEXT TEST *  
        100.                   ;ITERATION COUNT *  
        BRK15                  ;SCOPE ENTRY POINT *  
        X=X+1  
*****
```

```
2987  
2988 013760 004537 004742 :BREAK TEST ON LINE 15.  
2989 013764 020000 BRK15: JSR 5,BRKTST ;GO DO BREAK TEST  
2990 000016 LBIT15 ;ON LINE 15  
2991 Y=Y+1  
2992 013766 000113 :*****  
2993 013770 014004 RT113: 113 ;ROUTINE # 113 *  
2994 013772 000144 RT114 ;ADDRESS OF NEXT TEST *  
2995 013774 013776 100. ;ITERATION COUNT *  
2996 000113 BRK16 ;SCOPE ENTRY POINT *  
2997 X=X+1  
2998 :*****  
2999 013776 004537 004742 :BREAK TEST ON LINE 16.  
3000 014002 040000 BRK16: JSR 5,BRKTST ;GO DO BREAK TEST  
3001 000017 LBIT16 ;ON LINE 16  
3002 Y=Y+1  
3003 014004 000114 :*****  
3004 014006 014022 RT114: 114 ;ROUTINE # 114 *  
3005 014010 000144 RT115 ;ADDRESS OF NEXT TEST *  
3006 014012 014014 100. ;ITERATION COUNT *  
3007 000114 BRK17 ;SCOPE ENTRY POINT *  
3008 X=X+1  
3009 :*****  
3010 014014 004537 004742 :BREAK TEST ON LINE 17.  
3011 014020 100000 BRK17: JSR 5,BRKTST ;GO DO BREAK TEST  
3012 000020 LBIT17 ;ON LINE 17  
3013 000000 Y=Y+1  
3014 000000 A=0  
3015 000000 Y=0  
3016 014022 000115 :*****  
3017 014024 014040 RT115: 115 ;ROUTINE #115 *  
3018 014026 000144 RT116 ;ADDRESS OF NEXT TEST *  
3019 014030 014032 100. ;ITERATION COUNT *  
3020 000115 DAT0 ;SCOPE ENTRY POINT *  
3021 X=X+1  
3022 :*****  
3023 014032 004537 005576 :DATA TEST 100 CHARACTERS LINED  
3024 014036 000000 DAT0: JSR 5,DATTST ;GO RUN DATA TEST  
3025 000001 LINED ;ON LINED  
3026 Y=Y+1  
3027 014040 000116 :*****  
3028 014042 014056 RT116: 116 ;ROUTINE #116 *  
3029 014044 000144 RT117 ;ADDRESS OF NEXT TEST *  
3030 014046 014050 100. ;ITERATION COUNT *  
3031 000116 DAT1 ;SCOPE ENTRY POINT *  
3032 X=X+1  
3033 :*****  
3034 014050 004537 005576 :DATA TEST 100 CHARACTERS LINE1  
3035 014054 000002 DAT1: JSR 5,DATTST ;GO RUN DATA TEST  
3036 000002 LINE1 ;ON LINE1  
3037 Y=Y+1  
3038 014056 000117 :*****  
3039 014060 014074 RT117: 117 ;ROUTINE #117 *  
3040 014062 000144 RT120 ;ADDRESS OF NEXT TEST *  
3041 014064 014066 100. ;ITERATION COUNT *  
3042 000117 DAT2 ;SCOPE ENTRY POINT *  
X=X+1
```

3043
3044
3045 014066 004537 005576
3046 014072 000004
3047 000003
3048
3049 014074 000120
3050 014076 014112
3051 014100 000144
3052 014102 014104
3053 000120
3054
3055
3056 014104 004537 005576
3057 014110 000006
3058 000004
3059
3060 014112 000121
3061 014114 014130
3062 014116 000144
3063 014120 014122
3064 000121
3065
3066
3067 014122 004537 005576
3068 014126 000010
3069 000005
3070
3071 014130 000122
3072 014132 014146
3073 014134 000144
3074 014136 014140
3075 000122
3076
3077
3078 014140 004537 005576
3079 014144 000012
3080 000006
3081
3082 014146 000123
3083 014150 014164
3084 014152 000144
3085 014154 014156
3086 000123
3087
3088
3089 014156 004537 005576
3090 014162 000014
3091 000007
3092
3093 014164 000124
3094 014166 014202
3095 014170 000144
3096 014172 014174
3097 000124
3098

```
*****  
:DATA TEST 100 CHARACTERS LINE2  
DAT2: JSR 5,DATTST ;GO RUN DATA TEST  
LINE2 ;ON LINE2  
Y=Y+1  
*****  
RT120: 120 ;ROUTINE #120 *  
RT121 ;ADDRESS OF NEXT TEST *  
100. ;ITERATION COUNT *  
DAT3 ;SCOPE ENTRY POINT *  
X=X+1  
*****  
:DATA TEST 100 CHARACTERS LINE3  
DAT3: JSR 5,DATTST ;GO RUN DATA TEST  
LINE3 ;ON LINE3  
Y=Y+1  
*****  
RT121: 121 ;ROUTINE #121 *  
RT122 ;ADDRESS OF NEXT TEST *  
100. ;ITERATION COUNT *  
DAT4 ;SCOPE ENTRY POINT *  
X=X+1  
*****  
:DATA TEST 100 CHARACTERS LINE4  
DAT4: JSR 5,DATTST ;GO RUN DATA TEST  
LINE4 ;ON LINE4  
Y=Y+1  
*****  
RT122: 122 ;ROUTINE #122 *  
RT123 ;ADDRESS OF NEXT TEST *  
100. ;ITERATION COUNT *  
DAT5 ;SCOPE ENTRY POINT *  
X=X+1  
*****  
:DATA TEST 100 CHARACTERS LINES  
DAT5: JSR 5,DATTST ;GO RUN DATA TEST  
LINES ;ON LINES  
Y=Y+1  
*****  
RT123: 123 ;ROUTINE #123 *  
RT124 ;ADDRESS OF NEXT TEST *  
100. ;ITERATION COUNT *  
DAT6 ;SCOPE ENTRY POINT *  
X=X+1  
*****  
:DATA TEST 100 CHARACTERS LINE6  
DAT6: JSR 5,DATTST ;GO RUN DATA TEST  
LINE6 ;ON LINE6  
Y=Y+1  
*****  
RT124: 124 ;ROUTINE #124 *  
RT125 ;ADDRESS OF NEXT TEST *  
100. ;ITERATION COUNT *  
DAT7 ;SCOPE ENTRY POINT *  
X=X+1  
*****
```

```

3099
3100 014174 004537 005576
3101 014200 000016
3102 000010
3103
3104 014202 000125
3105 014204 014220
3106 014206 000144
3107 014210 014212
3108 000125
3109
3110
3111 014212 004537 005576
3112 014216 000020
3113 000011
3114
3115 014220 000126
3116 014222 014236
3117 014224 000144
3118 014226 014230
3119 000126
3120
3121
3122 014230 004537 005576
3123 014234 000022
3124 000012
3125
3126 014236 000127
3127 014240 014254
3128 014242 000144
3129 014244 014246
3130 000127
3131
3132
3133 014246 004537 005576
3134 014252 000024
3135 000013
3136
3137 014254 000130
3138 014256 014272
3139 014260 000144
3140 014262 014264
3141 000130
3142
3143
3144 014264 004537 005576
3145 014270 000026
3146 000014
3147
3148 014272 000131
3149 014274 014310
3150 014276 000144
3151 014300 014302
3152 000131
3153
3154

```

```

:DATA TEST 100 CHARACTERS LINE7
DAT7: JSR 5,DATTST ;GO RUN DATA TEST
      LINE7 ;ON LINE7
      Y=Y+1
:*****
RT125: 125 ;ROUTINE #125 *
      RT126 ;ADDRESS OF NEXT TEST *
      100. ;ITERATION COUNT *
      DAT10 ;SCOPE ENTRY POINT *
      X=X+1
:*****
:DATA TEST 100 CHARACTERS LINE10
DAT10: JSR 5,DATTST ;GO RUN DATA TEST
      LINE10 ;ON LINE10
      Y=Y+1
:*****
RT126: 126 ;ROUTINE #126 *
      RT127 ;ADDRESS OF NEXT TEST *
      100. ;ITERATION COUNT *
      DAT11 ;SCOPE ENTRY POINT *
      X=X+1
:*****
:DATA TEST 100 CHARACTERS LINE11
DAT11: JSR 5,DATTST ;GO RUN DATA TEST
      LINE11 ;ON LINE11
      Y=Y+1
:*****
RT127: 127 ;ROUTINE #127 *
      RT130 ;ADDRESS OF NEXT TEST *
      100. ;ITERATION COUNT *
      DAT12 ;SCOPE ENTRY POINT *
      X=X+1
:*****
:DATA TEST 100 CHARACTERS LINE12
DAT12: JSR 5,DATTST ;GO RUN DATA TEST
      LINE12 ;ON LINE12
      Y=Y+1
:*****
RT130: 130 ;ROUTINE #130 *
      RT131 ;ADDRESS OF NEXT TEST *
      100. ;ITERATION COUNT *
      DAT13 ;SCOPE ENTRY POINT *
      X=X+1
:*****
:DATA TEST 100 CHARACTERS LINE13
DAT13: JSR 5,DATTST ;GO RUN DATA TEST
      LINE13 ;ON LINE13
      Y=Y+1
:*****
RT131: 131 ;ROUTINE #131 *
      RT132 ;ADDRESS OF NEXT TEST *
      100. ;ITERATION COUNT *
      DAT14 ;SCOPE ENTRY POINT *
      X=X+1
:*****
:DATA TEST 100 CHARACTERS LINE14

```

```

3155 014302 004537 005576
3156 014306 000030
3157 000015
3158
3159 014310 000132
3160 014312 014326
3161 014314 000144
3162 014316 014320
3163 000132
3164
3165
3166 014320 004537 005576
3167 014324 000032
3168 000016
3169
3170 014326 000133
3171 014330 014344
3172 014332 000144
3173 014334 014336
3174 000133
3175
3176
3177 014336 004537 005576
3178 014342 000034
3179 000017
3180
3181 014344 000134
3182 014346 014362
3183 014350 000144
3184 014352 014354
3185 000134
3186
3187
3188 014354 004537 005576
3189 014360 000036
3190 000020
3191
3192 014362 000135
3193 014364 014762
3194 014366 000144
3195 014370 014372
3196 000135
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206 014372 005037 001306
3207 014376 004537 005510
3208 014402 001306
3209 014404 001307
3210 014406 000177

```

```

DAT14: JSR      5,DATTST      ;GO RUN DATA TEST
        LINE14                ;ON LINE14
        Y=Y+1
;*****
RT132: 132                    ;ROUTINE #132
        RT133                  ;ADDRESS OF NEXT TEST
        100.                   ;ITERATION COUNT
        DAT15                  ;SCOPE ENTRY POINT
        X=X+1
;*****
;DATA TEST 100 CHARACTERS LINE15
DAT15: JSR      5,DATTST      ;GO RUN DATA TEST
        LINE15                ;ON LINE15
        Y=Y+1
;*****
RT133: 133                    ;ROUTINE #133
        RT134                  ;ADDRESS OF NEXT TEST
        100.                   ;ITERATION COUNT
        DAT16                  ;SCOPE ENTRY POINT
        X=X+1
;*****
;DATA TEST 100 CHARACTERS LINE16
DAT16: JSR      5,DATTST      ;GO RUN DATA TEST
        LINE16                ;ON LINE16
        Y=Y+1
;*****
RT134: 134                    ;ROUTINE #134
        RT135                  ;ADDRESS OF NEXT TEST
        100.                   ;ITERATION COUNT
        DAT17                  ;SCOPE ENTRY POINT
        X=X+1
;*****
;DATA TEST 100 CHARACTERS LINE17
DAT17: JSR      5,DATTST      ;GO RUN DATA TEST
        LINE17                ;ON LINE17
        Y=Y+1
;*****
RT135: 135                    ;ROUTINE # 135
        RT136                  ;ADDR OF NEXT ROUTINE.
        100.                   ;ITERATION COUNT
        RT135A                 ;SCOPE ENTRY POINT.
        X=X+1
;*****
;TEST THAT DATA (ALL 1'S) CAN BE TRANSMITTED ON LINES SIMULTANEOUSLY.
;THE FOLLOWING TESTS ARE PERFORMED:
;THERE ARE 16. DATA ENTRIES
;THERE ISN'T A 17TH ENTRY
;DATA RECEIVED IS CORRECT
;ONE DATA ENTRY PER LINE
RT135A: CLR      TUMTAB        ;CLEAR THE
        JSR      5,BMOVE      ;TUMBLE
        TUMTAB                 ;TABLE
        TUMTAB+1              ;(200
        177                   ;ENTRIES)

```

```

3211 014410 012737 177777 016350      MOV      #-1,OUTBUF      ;LOAD CHAR INTO OUTPUT BUFFER
3212 014416 005000                    CLR      %0              ;SET RO = LINE 0
3213 014420 012737 000001 001272      MOV      @LBIT0,LINBIT   ;GET LINE BIT
3214 014426 012777 000001 164614      MOV      @BIT0,@CSR      ;SET THE GO BIT
3215 014434 010037 016176                    MOV      %0,LINE        ;GET LINE NUMBER
3216 014440 004537 005532      1S:     JSR      5,XMITD      ;TRANSMIT 1 CHAR.
3217 014444 177777                    -1              ;ON EACH LINE
3218 014446 005720                    TST      (0)+           ;INCREMENT LINE NUMBER (+2)
3219 014450 006337 001272      ASL      LINBIT         ;SHIFT LINE BIT TO NEXT LINE
3220 014454 103367                    BCC      1S             ;BRANCH IF ALL LINES NOT DONE
3221 014456 013737 003642 014466      MOV      TIME1,2S       ;PUT TIME TO TRANSMIT 1 CHAR
3222 014464 104400                    DELAY         ;DELAY 1
3223 014466 000000      2S:     OPEN          ;CHARACTER TIME
3224 014470 017737 164556 001300      MOV      @BAR,RCVDAT    ;GET & TEST BAR CONTENTS
3225 014476 011410                    BEQ      3S             ;BRANCH IF 0
3226 014500 005037 001302      CLR      XMTDAT         ;
3227 014504 104011                    ERROR1        ;ERROR! BAR NOT CLEAR AFTER ALL
3228 014506 005077 164540                    CLR      @BAR          ;LINES FINISHED
3229 014512 005077 164532                    CLR      @CSR          ;
3230 014516 000520                    BR       16S           ;GO TO EXIT
3231 014520 032777 020000 164522      3S:     BIT      @BIT13,@CSR ;TEST THAT OVER RUN DID NOT SET
3232 014526 001404                    BEQ      4S             ;
3233 014530 104001                    ERROR        ;ERROR! OVER RUN BIT SET
3234 014532 005077 164512                    CLR      @CSR          ;
3235 014536 000510                    BR       16S           ;GO TO EXIT
3236
3237      ;TEST THAT THERE ARE 16. VALID DATA ENTRIES
3238 014540 005077 164504      4S:     CLR      @CSR      ;CLEAR THE CSR
3239 014544 012702 000020                    MOV      @16.,%2       ;GET TT SCAN COUNT
3240 014550 012701 001306                    MOV      @TUMTAB,%1    ;GET FIRST TT ADDRESS
3241 014554 005302      5S:     DEC      %2        ;DECREMENT SCAN COUNTER
3242 014556 100404                    BMI      6S            ;BRANCH IF 16. ENTRIES SCANNED
3243 014560 005721                    TST      (1)+         ;TEST FOR VALID DATA ENTRY
3244 014562 100774                    BMI      5S            ;BRANCH IF FOUND
3245 014564 104001                    ERROR      ;ERROR! MISSING DATA ENTRY
3246 014566 000474                    BR       16S           ;GO TO EXIT
3247 014570 005721      6S:     TST      (1)+         ;TEST 17TH ENTRY (SHOULD BE = TO 0)
3248 014572 001402                    BEQ      7S            ;BRANCH IF 0
3249 014574 104001                    ERROR      ;ERROR! EXTRA DATA ENTRY
3250 014576 000470                    BR       16S           ;GO TO EXIT
3251
3252      ;TEST THAT THE DATA IS CORRECT IN ALL 16. ENTRIES
3253 014600 012701 001306      7S:     MOV      @TUMTAB,%1 ;GET FIST TT ADDRESS
3254 014604 012702 000020                    MOV      @16.,%2       ;GET SCAN COUNT
3255 014610 005302      8S:     DEC      %2        ;DECREMENT SCAN COUNT
3256 014612 100421                    BMI      10S           ;BRANCH IF 16. ENTRIES SCANNED
3257 014614 013737 016350 001302      MOV      OUTBUF,XMTDAT ;GET TRANSMITTED DATA
3258 014622 043737 001304 001302      BIC      CARMSK,XMTDAT ;CLEAR NON-TRANSMITTED BITS
3259 014630 113737 001306 001300      MOV      TUMTAB,RCVDAT ;GET RECEIVED DATA
3260 014636 123737 001302 001300      CMPB    XMTDAT,RCVDAT ;COMPARE DATA
3261 014644 001402                    BEQ      9S            ;
3262 014646 104011                    ERROR1        ;ERROR INCORRECT DATA
3263 014650 000443                    BR       16S           ;GO TO EXIT
3264 014652 005721      9S:     TST      (1)+         ;INCREMENT TT ADDRESS
3265 014654 000755                    BR       8S            ;TEST NEXT ENTRY
3266

```

```

3267
3268 014656 012701 001306
3269 014662 012702 000020
3270 014666 005302
3271 014670 100403
3272 014672 042721 160777
3273 014676 000773
3274
3275
3276 014700 005037 001302
3277 014704 012701 000020
3278 014710 012702 000020
3279 014714 012700 001306
3280 014720 023720 001302
3281 014724 001406
3282 014726 005302
3283 014730 001373
3284 014732 005037 001300
3285 014736 104011
3286 014740 000407
3287 014742 005301
3288 014744 005701
3289 014746 001404
3290 014750 062737 001000 001302
3291 014756 000754
3292 014760 104006
3293
3294 014762 000136
3295 014764 015276
3296 014766 000144
3297 014770 014772
3298 000136
3299
3300
3301
3302 014772 013737 003642 015046
3303 015000 005037 001306
3304 015004 004537 005510
3305 015010 001306
3306 015012 001307
3307 015014 000177
3308 015016 012777 000001 164224
3309 015024 012777 177777 164222
3310 015032 105777 164212
3311 015036 100375
3312 015040 005077 164210
3313 015044 104400
3314 015046 000000
3315 015050 022777 000201 164172
3316 015056 001410
3317 015060 017737 164164 001300
3318 015066 012737 000201 001302
3319 015074 104011
3320 015076 000476
3321
3322

```

```

: CLEAR ALL BUT LINE NUMBER IN TUMBLE TABLE ENTRY
10S:  MOV      #TUMTAB,%1      : GET FIRST TT ADDRESS
      MOV      #16.,%2        : GET SCAN COUNT
11S:  DEC      %2              : DECREMENT SCAN COUNT
      BMI      12S             : BRANCH IF ALL LINES TESTED
      BIC      #160777,(1)+    : CLEAR ALL BUT LINE NUMBER IN TT
      BR      11S             : DO NEXT TT ADDRESS

: TEST THAT THERE IS AN ENTRY FOR EACH OF THE 16. LINES
12S:  CLR      XMTDAT
      MOV      #16.,%1
13S:  MOV      #16.,%2
      MOV      #TUMTAB,%0
14S:  CMP      XMTDAT,(0)+     : TEST FOR LINE ENTRY
      BEQ      15S             : BRANCH IF FOUND
      DEC      %2              : DECREMENT SCAN COUNT
      BNE      14S             : LOOK AT NEXT ENTRY
      CLR      RCVDAT
      ERROR1
      BR      16S
15S:  DEC      %1              : DECREMENT LINES FOUND COUNT
      TST      %1
      BEQ      16S             : BRANCH IF ALL LINE TESTED
      ADD      #1000,XMTDAT    : INCREMENT LINE NUMBER
      BR      13S             : GO DO NEXT LINE
16S:  SCOPE
: *****
RT136: 136                    : ROUTINE # 136 *
      RT137                    : ADDR OF NEXT ROUTINE. *
      100.                      : ITERATION COUNT *
      RT136A                    : SCOPE ENTRY POINT. *
      X=X+1
: *****

: TEST THAT THE DM11 CAN TRANSMIT A BREAK ON ALL LINES SIMULTANEOUSLY
RT136A: MOV      @TIME1,1S     : GET TIME TO TRANSMIT ONE CHARACTER
      CLR      TUMTAB         : CLEAR
      JSR      5,BMOVE        : THE
      TUMTAB
      TUMTAB+1                : TUMBLE
      177                      : TABLE
      MOV      @BIT0,@CSR     : SET GO
      MOV      @-1,@BKCSR     : SET BREAK BIT FOR ALL LINES
      TSTB    @CSR           : WAIT FOR THE RECEIVER
      BPL     -4              : TO RECEIVE A BREAK
      CLR     @BKCSR         : CLEAR ALL BREAK BITS
      DELAY
      OPEN
1S:   CMP      #201,@CSR      : TEST THAT ONLY GO AND DONE ARE SET
      BEQ     2S
      MOV     @CSR,RCVDAT    : GET CSR ENTRY
      MOV     #201,XMTDAT    : GET CORRECT RESULT
      ERROR1
      BR     13S            : ERROR! INCORRECT CSR DATA
                          : EXIT

: TEST THAT THERE IS 16. VALID DATA ENTRIES

```

```

3323 015100 012701 001306
3324 015104 012702 000020
3325 015110 005721
3326 015112 100402
3327 015114 104001
3328 015116 000466
3329 015120 005302
3330 015122 001372
3331
3332
3333 015124 012701 001306
3334 015130 012702 000020
3335 015134 032721 040000
3336 015140 001002
3337 015142 104001
3338 015144 000453
3339 015146 005302
3340 015150 001371
3341
3342
3343 015152 012701 001306
3344 015156 012702 000020
3345 015162 105721
3346 015164 001402
3347 015166 104001
3348 015170 000441
3349 015172 105721
3350 015174 005302
3351 015176 001371
3352
3353
3354 015200 012701 001306
3355 015204 012702 000020
3356 015210 042721 160777
3357 015214 005302
3358 015216 001374
3359
3360
3361 015220 005004
3362 015222 012703 000020
3363 015226 012702 000020
3364 015232 012701 001306
3365 015236 020421
3366 015240 001410
3367 015242 005302
3368 015244 001374
3369 015246 010437 001302
3370 015252 010437 001300
3371 015256 104011
3372 015260 000405
3373 015262 005303
3374 015264 001403
3375 015266 062704 001000
3376 015272 000755
3377 015274 104006
3378
    
```

```

25:  MOV      #TUMTAB,%1      ;GET TUMBLE TABLE BASE ADDRESS
      MOV      #16.,%2        ;GET SCAN COUNT
35:  TST      (1)+             ;TEST FOR VALID DATA ENTRY
      BMI      45             ;BRANCH IF VALID DATA ENTRY FOUND
      ERROR    45             ;ERROR! MISSING VALID DATA ENTRY
      BR       13$           ;EXIT
45:  DEC      %2              ;DECREMENT SCAN COUNT
      BNE     35             ;BRANCH IF 16. ENTRIES NOT SCANNED

;TEST THAT THE BREAK BIT IS SET IN 16. TUMBLE TABLE ENTRIES
      MOV      #TUMTAB,%1
      MOV      #16.,%2
55:  BIT      #BIT14,(1)+     ;BREAK BIT SET?
      BNE     65             ;BRANCH IF SET
      ERROR    65             ;ERROR! MISSING BREAK BIT
      BR       13$           ;EXIT
65:  DEC      %2              ;DECREMENT SCAN COUNT
      BNE     55

;TEST THAT THE TUMBLE TABLE DATA BYTE IS ALL 0'S
      MOV      #TUMTAB,%1
      MOV      #16.,%2
75:  TSTB    (1)+             ;TEST DATA BYTE
      BEQ     85             ;BRANCH IF 0'S
      ERROR    85             ;ERROR! INCORRECT DATA
      BR       13$           ;EXIT
85:  TSTB    (1)+             ;STEP TABLE POINTER TO NEXT DATA BYTE
      DEC     %2
      BNE     75

;CLEAR ALL BUT LINE NUMBER IN TUMBLE TABLE ENTRY
      MOV      #TUMTAB,%1
      MOV      #16.,%2
95:  BIC     #160777,(1)+    ;CLEAR ALL BUT LINE NUMBER
      DEC     %2
      BNE     95

;TEST THAT THERE IS A TUMBLE TABLE ENTRY FOR EACH LINE
      CLR      %4             ;CLEAR LINE NUMBER
      MOV      #16.,%3
105: MOV      #16.,%2
      MOV      #TUMTAB,%1
115: CMP     %4,(1)+         ;TEST FOR LINR ENTRY FOR THIS LINE
      BEQ     12$           ;BRANCH IF FOUND
      DEC     %2
      BNE     115
      MOV     %4,XMTDAT
      MOV     %4,RCVDAT
      ERROR1  115           ;ERROR! NO LINE ENTRY FOUND FOR THIS LINE
      BR       13$           ;EXIT
125: DEC     %3              ;ALL LINES BEEN FOUND
      BEQ     13$           ;EXIT IF YES
      ADD     #1000,%4       ;SEARCH FOR
      BR       105          ;NEXT LINE
135: SCOPE
;*****
    
```

3379 015276 000137
 3380 015300 015314
 3381 015302 000002
 3382 015304 015306
 3383 000137
 3384
 3385
 3386
 3387
 3388 015306 004537 005066
 3389 015312 000040
 3390
 3391 015314 000140
 3392 015316 015332
 3393 015320 000002
 3394 015322 015324
 3395 000140
 3396
 3397
 3398
 3399
 3400 015324 004537 005066
 3401 015330 000020
 3402
 3403 015332 000141
 3404 015334 015350
 3405 015336 000002
 3406 015340 015342
 3407 000141
 3408
 3409
 3410
 3411
 3412 015342 004537 005066
 3413 015346 000010
 3414
 3415 015350 000142
 3416 015352 015366
 3417 015354 000002
 3418 015356 015360
 3419 000142
 3420
 3421
 3422
 3423
 3424 015360 004537 005066
 3425 015364 000004
 3426
 3427 015366 000143
 3428 015370 015404
 3429 015372 000002
 3430 015374 015376
 3431 000143
 3432
 3433
 3434

```

RT137: 137          ;ROUTINE # 137          *
        RT140       ;ADDR OF NEXT ROUTINE. *
        2           ;ITERATION COUNT      *
        RT137A     ;SCOPE ENTRY POINT.   *
        X=X+1
;*****
;TEST TO TRANSMIT ON EACH LINE WITH A DELAY BEFORE STATING THE
;NEXT LINE.
RT137A: JSR      5,2#DLYXMT ;GO DO TEST. DELAY
        32.        ;THIS MUCH BETWEEN LINES
;*****
RT140: 140        ;ROUTINE # 140          *
        RT141       ;ADDR OF NEXT ROUTINE. *
        2           ;ITERATION COUNT      *
        RT140A     ;SCOPE ENTRY POINT.   *
        X=X+1
;*****
;TEST TO TRANSMIT ON EACH LINE WITH A DELAY BEFORE STATING THE
;NEXT LINE.
RT140A: JSR      5,2#DLYXMT ;GO DO TEST. DELAY
        16.        ;THIS MUCH BETWEEN LINES
;*****
RT141: 141        ;ROUTINE # 141          *
        RT142       ;ADDR OF NEXT ROUTINE. *
        2           ;ITERATION COUNT      *
        RT141A     ;SCOPE ENTRY POINT.   *
        X=X+1
;*****
;TEST TO TRANSMIT ON EACH LINE WITH A DELAY BEFORE STATING THE
;NEXT LINE.
RT141A: JSR      5,2#DLYXMT ;GO DO TEST. DELAY
        8.         ;THIS MUCH BETWEEN LINES
;*****
RT142: 142        ;ROUTINE # 142          *
        RT143       ;ADDR OF NEXT ROUTINE. *
        2           ;ITERATION COUNT      *
        RT142A     ;SCOPE ENTRY POINT.   *
        X=X+1
;*****
;TEST TO TRANSMIT ON EACH LINE WITH A DELAY BEFORE STATING THE
;NEXT LINE.
RT142A: JSR      5,2#DLYXMT ;GO DO TEST. DELAY
        4          ;THIS MUCH BETWEEN LINES
;*****
RT143: 143        ;ROUTINE # 143          *
        RT144       ;ADDR OF NEXT ROUTINE. *
        2           ;ITERATION COUNT      *
        RT143A     ;SCOPE ENTRY POINT.   *
        X=X+1
;*****
;TEST TO TRANSMIT ON EACH LINE WITH A DELAY BEFORE STATING THE

```

```

3435
3436 015376 004537 005064
3437 015402 000002
3438
3439 015404 000144
3440 015406 015422
3441 015410 000002
3442 015412 015414
3443 000144
3444
3445
3446
3447
3448 015414 004537 005066
3449 015420 000001
3450
3451 015422 000145
3452 015424 015570
3453 015426 000144
3454 015430 015432
3455 000145
3456
3457
3458
3459
3460
3461 015432 005037 001306
3462 015436 005037 001310
3463 015442 012737 016350 001106
3464 015450 012737 177777 001146
3465 015456 012777 000007 163564
3466 015464 012777 000001 163560
3467 015472 012777 000002 163554
3468 015500 105777 163544
3469 015504 100375
3470 015506 005077 163542
3471
3472
3473 015512 022737 141000 001306
3474 015520 001410
3475 015522 013737 001306 001300
3476 015530 012737 141000 001302
3477 015536 104011
3478 015540 000407
3479 015542 013737 001310 001300
3480 015550 001403
3481 015552 005037 001302
3482 015556 104011
3483 015560 005777 163464
3484 015564 100375
3485 015566 104006
3486
3487 015570 000146
3488 015572 177777
3489 015574 000144
3490 015576 015500

```

```

: NEXT LINE.
RT143A: JSR      5, @DLYXMT      ; GO DO TEST. DELAY
2                          ; THIS MUCH BETWEEN LINES
: *****
RT144: 144      ; ROUTINE # 144
RT145      ; ADDR OF NEXT ROUTINE.
2          ; ITERATION COUNT
RT144A     ; SCOPE ENTRY POINT.
X=X+1
: *****

: TEST TO TRANSMIT ON EACH LINE WITH A DELAY BEFORE STATING THE
: NEXT LINE.
RT144A: JSR      5, @DLYXMT      ; GO DO TEST. DELAY
1                          ; THIS MUCH BETWEEN LINES
: *****
RT145: 145      ; ROUTINE # 145
RT146      ; ADDR OF NEXT ROUTINE.
100.       ; ITERATION COUNT
RT145A     ; SCOPE ENTRY POINT.
X=X+1
: *****

: TEST THAT THE DM11 WORKS PROPERLY WHEN THE HALF-DUPLEX BIT (CSR BIT 1)
: IS SET. THE TEST TRANSMITS DATA ON LINE 0, AND 'BREAKS' ON LINE 1. ONLY
: THE BREAK SHOULD BE RECEIVED ON LINE 0 IN THE TUMBLE TABLE.
RT145A: CLR      TUMTAB          ; CLEAR THE FIRST TWO
CLR      TUMTAB+2              ; TUMBLE TABLE ENTRIES
MOV      @OUTBUF, CAT          ; SET UP TO
MOV      @-1, WCT              ; TRANSMIT 1 CHARACTER
MOV      @7, @CSR              ; SET GO, HALF DUPLEX & MAINT BITS
MOV      @LBIT0, @BAR          ; TRANSMIT 1 CHAR. ON LINE 0
MOV      @LBIT1, @BKCSR        ; SET BREAK ON LINE 1
TSTB    @CSR                   ; WAIT FOR THE CHARACTER
BPL     -4                     ; TO BE RECEIVED
CLR      @BKCSR                ; CLEAR THE BREAK BIT ON LINE 1

: TEST THAT ONLY THE BREAK WAS RECEIVED
CMP      @141000, TUMTAB       ; TOST FOR BREAK ENTRY (LINE 1)
BEQ     15
MOV      TUMTAB, RCVDAT        ; GET ACTUAL ENTRY
MOV      @141000, XMTDAT        ; GET CORRECT ENTRY
ERROR1  ; ERROR! INCORRECT BREAK ENTRY
BR      25                      ; GO TO EXIT
15:     MOV      TUMTAB+2, RCVDAT ; TEST THAT NEXT ENTRY IS CLEAR
BEQ     25                      ; EXIT IF CORRECT
CLR      XMTDAT
ERROR1  ; ERROR! SECOND ENTRY WAS NOT CLEAR
25:     TST      @CSR            ; WAIT FOR THE TRANSMITTER
BPL     -4                     ; TO FINISH
SCOPE   ; SCOPE
: *****
RT146: 146      ; ROUTINE # 146
RTLAST  ; ADDR OF NEXT ROUTINE.
100.    ; ITERATION COUNT
RT146A  ; SCOPE ENTRY POINT.

```

```

3491          000146          X=X+1
3492          ;*****
3493          :TEST THAT THE DM11 RESPONDS CORRECTLY TO A RESET
3494          RT146A: MOV      @OUTBUF,CAT      ;SET UP TO TRANSMIT 10
3495          MOV      @-10,WCT      ;CHARACTERS ON LINE 0
3496          MOV      TIME1,2$      ;GET TIME TO TRANSMIT 2 CHARACTERS
3497          MOV      TIME1,6$
3498          CLR      XMTDAT
3499          MOV      @7,@CSR      ;SET MAINT., HALF DUPLEX & GO BITS
3500          MOV      @LBIT0,@BAR      ;START TO TRANSMIT ON LINE 0
3501          MOV      @LBIT1,@BKCSR      ;BREAK ON LINE 1
3502          MOV      @2,@X4
3503          15: DELAY
3504          25: OPEN
3505          DEC      @X4
3506          BNE     @15
3507          SRESET
3508          MOV      @CSR,RCVDAT      ;RESET
3509          BEQ     @35
3510          ERROR1 ;GET CSR CONTENTS
3511          MOV      @BAR,RCVDAT      ;BRANCH IF 0
3512          BEQ     @35
3513          ERROR1 ;ERROR! CSR DID NOT CLEAR
3514          BR      @95
3515          MOV      @88,@X4
3516          45: DELAY
3517          55: OPEN
3518          65: DEC      @X4
3519          BNE     @55
3520          MOV      @CSR,RCVDAT      ;TEST THAT CSR IS CLEAR
3521          BEQ     @75
3522          ERROR1 ;ERROR! CSR WAS NOT CLEAR
3523          BR      @95
3524          MOV      @BAR,RCVDAT      ;GO TO EXIT
3525          BEQ     @75
3526          ERROR1 ;TES THAT BAR IS CLEAR
3527          BR      @95
3528          MOV      @BKCSR,RCVDAT    ;ERROR! BAR DID NOT CLEAR
3529          BEQ     @85
3530          ERROR1 ;TEST THAT BKCSR IS CLEAR
3531          BR      @95
3532          ERROR1 ;ERROR! BKCSR DID NOT CLEAR
          SCOPE

```

3533
3534
3535
3536
3537
3538
3539
3540
3541
3542
3543
3544
3545
3546
3547
3548

016002 000240
016004 104000
016006 017444
016010 004737 016166
016014 004737 016240
016020 005777 163226
016024 001375
016026 005077 163216
016032 005037 016046
016036 117737 163041 016046
016044 104400
016046 000000
016050 000761

:PRG1- TRANSMITTER SCOPE LOOP
PRG1: NOP
TYPE
PRG1M
JSR 7,PARAM
PRG1R: JSR 7,LOOP
PRG1B: TST @BAR
BNE PRG1B
PRG1C: CLR @CSR
CLR PRG1D
MOVB @SWR+1,PRG1D
DELAY
PRG1D: OPEN
BR PRG1R

:BEGIN
:TYPE PROGRAM TITLE
:GO GET USER PARAMETERS
:GO LOOP TRANSMITTER
:WAIT FOR ALL LINES TO FINISH
:BRANCH IF NOT DONE
:CLEAR THE CSR
:CLEAR DELAY TIME
:GET DELAY
:DELAY AS SPECIFIED
:BY USER
:LOOP BACK

```

3549
3550
3551 016052 000240
3552 016054 104000
3553 016056 017474
3554 016060 004737 016166
3555 016064 004737 016240
3556 016070 012777 000001 163152
3557 016076 005777 163150
3558 016102 001415
3559 016104 105777 163140
3560 016110 100372
3561 016112 042777 000200 163130
3562 016120 020127 001504
3563 016124 001002
3564 016126 012701 001304
3565 016132 005721
3566 016134 000755
3567 016136 005077 163106
3568 016142 005077 163104
3569 016146 005037 016162
3570 016152 117737 162725 016162
3571 016160 104400
3572 016162 000000
3573 016164 000737
3574
3575
3576
3577 016166 104000
3578 016170 017531
3579 016172 004537 003702
3580 016176 000000
3581 016200 104000
3582 016202 017563
3583 016204 004537 003702
3584 016210 000000
3585 016212 023727 016210 000310
3586 016220 101403
3587 016222 104000
3588 016224 017241
3589 016226 000764
3590 016230 104000
3591 016232 017356
3592 016234 104015
3593 016236 000207
3594
3595
3596
3597 016240 117737 162636 016350
3598 016246 004537 005510
3599 016252 016350
3600 016254 016351
3601 016256 000307
3602 016260 012777 001106 162770
3603 016266 012737 016350 001106
3604 016274 004537 005510

:PRG2- RECEIVER SCOPE LOOP
PRG2: NOP
TYPE
PRG2M
JSR 7,PARAM
PRG2R: JSR 7,LOOP
PRG2A: MOV #BIT0,@CSR
PRG2AA: TST @BAR
BEQ PRG2B
TSTB @CSR
BPL PRG2AA
BIC #BIT7,@CSR
CMP %1,@TUMTAB+176
BNE .+6
MOV @TUMTAB-2,%1
TST (1)+
BR PRG2A
PRG2B: CLR @CSR
CLR @BAR
CLR PRG2C
MOVB @SWR+1,PRG2C
DELAY
PRG2C: OPEN
BR PRG2R

;BEGIN
;TYPE PROGRAM
;TITLE
;GO GET USER PARAMETERS
;GO START TRANSMITTER
;SET GO,CLEAR THE OTHERS
;HAVE ALL LINES SELECTED FINISHED
;BRANCH IF FINISHED TRANSMITTING
;WAIT FOR THE RECEIVER TO
;RECEIVE A CHARACTER
;CLEAR RECEIVER FLAG
;IS THE POINTER AT THE END OF THE TT
;BRANCH IF NOT
;RESET POINTER
;INCREMENT POINTER
;GO BACK & TEST TRANSMITTER FLAG
;CLEAR THE CSR
;CLEAR THE BAR
;CLEAR USER DELAY
;LOAD USER DELAY
;DELAY AS SPECIFIED
;BY USER
;REPEAT LOOP

;SUBROUTINE TO GET USER PARAMETERS (FOR PRG1 & 2)
PARAM: TYPE
LINPAR
JSR 5,RECD
LINE: 0
PARAMA: TYPE
HOWMAN
JSR 5,RECD
CHARS: 0
CMP CHARS,#200.
BLOS PARAMB
TYPE
M1
BR PARAMA
PARAMB: TYPE
M4
CNTRLU
RTS 7

;ASK USER WHICH LINE
;TO TEST
;GET LINE AND PUT IT
;HERE
;ASK USER HOW MANY
;CHARACTERS TO TRANSMIT
;GET CHARS AND PUT IT
;HERE
;LIMIT RESPONSE TO 200.
;(CORE LIMITATION)

;RE-REQUEST PARAMETER
;TYPE INSTRUCTIONS

;GO GET VALUE
;EXIT

;SUBROUTINE TO TRANSMIT DATA FROM THE SR
LOOP: MOVB @SWR,OUTBUF
JSR 5,BMOVE
OUTBUF
OUTBUF+1
199.
MOV @CAT,@BASEREG
MOV @OUTBUF,CAT
JSR 5,BMOVE

;FILL OUTPUT
;BUFFER
;WITH
;DATA TO BE
;TRANSMITTED
;INITIALIZE BASE REGISTER
;LOAD CURRENT
;ADDRESS TABLE

```

```

3605 016300 001106 CAT ;WITH ADDRESS
3606 016302 001110 CAT+2 ;OF OUTPUT BUFFER
3607 016304 000040 32.
3608 016306 013737 016210 001146 MOV CHARS,WCT ;LOAD WORD COUNT
3609 016314 005437 001146 NEG WCT ;FORM TWO'S COMPLEMENT
3610 016320 004537 005510 JSR 5,BMOVE ;TABLE WITH
3611 016324 001146 WCT ;NUMBER OF
3612 016326 001150 WCT+2 ;CHARACTERS TO BE
3613 016330 000040 32. ;TRANSMITTED
3614 016332 013737 016176 001272 MOV LINE,LINBIT ;SAVE LINES TO BE TRANSMITTED ON
3615 016340 013777 016176 162704 MOV LINE,ABAR ;START TRANSMITTING ON SELECTED LINES
3616 016346 000207 RTS 7 ;EXIT
3617
3618
3619 016350 000000 OUTBUF: 0 ;FIRST ADDRESS OF 100.
3620 016514 016514 ;.=OUTBUF+100. ;CHARACTER OUTPUT BUFFER
3621 016514 000000 INBUF: 0 ;FIRST ADDRESS OF 100.
3622 016660 016660 ;.=INBUF+100. ;CHARACTER INPUT BUFFER (WHERE RECEIVED
3623 ;DATA IS STORED)
3624
3625 016660 013746 000006 SUSMRR: MOV @#6,-(SP) ;SAVE VECTORS
3626 016664 013746 000004 MOV @#4,-(SP)
3627 016670 012737 016710 000004 MOV @15,@#4 ;SET UP FOR TIMEOUT
3628 016676 022777 177777 162176 CMP @-1,@SMR ;REFERENCE HARDWARE SWITCH REGISTER
3629 016704 001402 BEQ 25
3630 016706 000407 BR 35
3631 016710 022626 15: CMP (SP)+,(SP)+ ;ADJUST STACK
3632 016712 012737 000176 001102 25: MOV @SMREG,SMR ;POINT TO SOFTWARE SWITCH REG
3633 016720 012737 000174 001104 MOV @DISPREG,DISPLAY ;POINT TO SOFT DISPLAY REG
3634 016726 012637 000004 35: MOV (SP)+,@#4 ;RESTORE VECTORS
3635 016732 012637 000006 MOV (SP)+,@#6
3636 016736 000002 RTI
3637
3638
3639 ;ROUTINE TO CHECK FOR FG BEING TYPED
3640
3641 016740 022737 000176 001102 KBDINTT: CMP @SMREG,SMR
3642 016746 001015 BNE 15
3643 016750 005037 017042 CLR TMP1 ;CLEAR TEMP AREA
3644 016754 117737 162616 017042 MOVB @TKOBR,TMP1 ;FETCH THE BUFFER
3645 016762 142737 000200 017042 BICB @200,TMP1 ;STRIP OFF PARITY
3646 016770 122737 000007 017042 CMPB @7,TMP1 ;WAS IT FG
3647 016776 001001 BNE 15 ;NOP
3648 017000 104015 CNTLU ;GO CHANGE IT
3649 017002 000002 15: RTI ;EXIT
3650
3651
3652 ;ROUTINE TO CHANGE CONTENTS OF SMREG(LOC 176)
3653
3654 017004 022737 000176 001102 CNTLU: CMP @SMREG,SMR
3655 017012 001023 BNE FJX
3656 017014 104000 TYPE
3657 017016 017106 $SMREG
3658 017020 004537 005440 JSR RS,0ACNV ;CONVERT TO ASCII
3659 017024 000176 SMREG
3660 017026 017115 $VALUE

```

3661	017030	000006			6			
3662	017032	104000			TYPE			
3663	017034	017115			SVALUE			
3664	017036	004537	003702		JSR	5,RECD		:GET TMP1 AND PUT IT
3665	017042	000000		TMP1:	0			:HERE
3666	017044	022737	000007 004060		CMP	#7,CNT		
3667	017052	001403			BEG	FJX		
3668	017054	013777	017042 162020		MOV	TMP1,@SWR		:CHANGE CONTENTS OF SWREG
3669	017062	000002		FJX:	RTI			
3670								

3671					:MESSAGES	
3672	017064	053045	041505	047524	WHERE: .ASCII	'%VECTOR ADDRESS? @'
3673	017072	020122	042101	051104		
3674	017100	051505	037523	040040		
3675	017106	051445	051127	020075	SSWREG: .ASCII	'%SWR= @'
3676	017114	100				
3677	017115	040	020040	020040	SVALUE: .ASCII	' NEW= @'
3678	017122	020040	020040	047040		
3679	017130	053505	020075	100		
3680	017135	045	020075		SCTLU: .ASCII	'%= '
3681	017140	052445	044516	020124	WHICH: .ASCII	'%UNIT #(8)? @'
3682	017146	024043	024470	020077		
3683	017154	100				
3684	017155	045	044103	051101	LEVEL: .ASCII	'%CHAR LENGTH @'
3685	017162	046040	047105	052107		
3686	017170	020110	100			
3687	017173	045	051105	020122	ERDAT: .ASCII	'%ERR S/B: '
3688	017200	027523	035102	040		
3689	017205	040	020040	020040	AASB: .ASCII	' WAS: '
3690	017212	020040	040527	035123		
3691	017220	040				
3692	017221	040	020040	020040	AAS: .ASCII	' @'
3693	017226	040040				
3694	017230	050045	043522	020056	MD: .ASCII	'%PRG. # @'
3695	017236	020043	100			
3696	017241	045	040077		M1: .ASCII	'%?@'
3697	017244	052045	051505	020124	M2: .ASCII	'%TEST DZDMA COMPLETE@'
3698	017252	055104	046504	020101		
3699	017260	047503	050115	042514		
3700	017266	042524	100			
3701	017271	045	042523	020124	M3: .ASCII	'%SET SR OPTIONS. NORMAL OPERATION'
3702	017276	051123	047440	052120		
3703	017304	047511	051516	020056		
3704	017312	047516	046522	046101		
3705	017320	047440	042520	040522		
3706	017326	044524	047117			
3707	017332	051511	051440	036522	.ASCII	'%IS SR=0.PRESS CONT.@'
3708	017340	027060	051120	051505		
3709	017346	020123	047503	052116		
3710	017354	040056				
3711	017356	050045	052125	041440	M4: .ASCII	'%PUT CHAR IN SR (0-7);DELAY IN SR (8-15)@'
3712	017364	040510	020122	047111		
3713	017372	051440	020122	030050		
3714	017400	033455	035451	042504		
3715	017406	040514	020131	047111		
3716	017414	051440	020122	034050		
3717	017422	030455	024465	100		
3718	017427	045	047514	044507	PRGOM: .ASCII	'%LOGIC TESTS@'
3719	017434	020103	042524	052123		
3720	017442	040123				
3721	017444	052045	040522	051516	PRG1M: .ASCII	'%TRANSMITTER SCOPE LOOP@'
3722	017452	044515	052124	051105		
3723	017460	051440	047503	042520		
3724	017466	046040	047517	040120		
3725	017474	052045	040522	051516	PRG2M: .ASCII	'%TRANSMIT/RECEIVE SCOPE LOOP@'
3726	017502	044515	027524	042522		

3727	017510	042503	053111	020105
3728	017516	041523	050117	020105
3729	017524	047514	050117	100
3730	017531	045	054524	042520
3731	017536	046040	047111	051505
3732	017544	052040	020117	042502
3733	017552	052040	051505	042524
3734	017560	020104	100	
3735	017563	045	047510	020127
3736	017570	040515	054516	041440
3737	017576	040510	040522	052103
3738	017604	051105	020123	100
3739	017611	045	020122	
3740	017614	020040	050040	036503
3741	017622	040		
3742	017623	040	020040	020040
3743	017630	040040		
3744	017632	015	012	
3745	017634	044124	020105	052521
3746	017642	041511	020113	051102
3747	017650	053517	020116	047506
3748	017656	020130	052512	050115
3749	017664	042105	047440	042526
3750	017672	020122	044124	020105
3751	017700	040514	054532	042040
3752	017706	043517	020123	040502
3753	017714	045503	030440	031462
3754	017722	032464	033466	034470
3755	017730	060		
3756		000001		

LINPAR: .ASCII '%TYPE LINES TO BE TESTED @'

HOWMAN: .ASCII '%HOW MANY CHARACTERS @'

EMO: .ASCII '%R '
ATNUMB: .ASCII ' PC= '

APC: .ASCII ' @'

MSG1: .BYTE 15,12
.ASCII 'THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 1234567890'

.END

RT101	013552	2872	2882#
RT102	013570	2883	2893#
RT103	013606	2894	2904#
RT104	013624	2905	2915#
RT105	013642	2916	2926#
RT106	013660	2927	2937#
RT107	013676	2938	2948#
RT11	007204	1804	1822#
RT11A	007214	1825	1831#
RT110	013714	2949	2959#
RT111	013732	2960	2970#
RT112	013750	2971	2981#
RT113	013766	2982	2992#
RT114	014004	2993	3003#
RT115	014022	3004	3016#
RT116	014040	3017	3027#
RT117	014056	3028	3038#
RT12	007274	1823	1846#
RT12A	007304	1849	1855#
RT120	014074	3039	3049#
RT121	014112	3050	3060#
RT122	014130	3061	3071#
RT123	014146	3072	3082#
RT124	014164	3083	3093#
RT125	014202	3094	3104#
RT126	014220	3105	3115#
RT127	014236	3116	3126#
RT13	007430	1847	1878#
RT13A	007440	1881	1886#
RT130	014254	3127	3137#
RT131	014272	3138	3148#
RT132	014310	3149	3159#
RT133	014326	3160	3170#
RT134	014344	3171	3181#
RT135	014362	3182	3192#
RT135A	014372	3195	3206#
RT136	014762	3193	3294#
RT136A	014772	3297	3302#
RT137	015276	3295	3379#
RT137A	015306	3382	3388#
RT14	007542	1879	1904#
RT14A	007552	1907	1913#
RT140	015314	3380	3391#
RT140A	015324	3394	3400#
RT141	015332	3392	3403#
RT141A	015342	3406	3412#
RT142	015350	3404	3415#
RT142A	015360	3418	3424#
RT143	015366	3416	3427#
RT143A	015376	3430	3436#
RT144	015404	3428	3439#
RT144A	015414	3442	3448#
RT145	015422	3440	3451#
RT145A	015432	3454	3461#
RT146	015570	3452	3487#
RT146A	015600	3490	3495#

1837

RT15	007602	1905	19218
RT15A	007612	1924	19308
RT16	007736	1922	19538
RT16A	007746	1956	19618
RT17	010054	1954	19808
RT17A	010064	1983	19888
RT2	006536	1671	16898
RT2A	006546	1692	16978
RT20	010146	1981	20078
RT20A	010156	2010	20158
RT21	010234	2008	20298
RT21A	010244	2032	20378
RT22	010342	2030	20538
RT22A	010352	2056	20618
RT23	010450	2054	20778
RT23A	010460	2080	20858
RT24	010556	2078	21018
RT24A	010566	2104	21098
RT25	010622	2102	21198
RT25A	010632	2122	21278
RT26	010730	2120	21458
RT27	010746	2146	21568
RT3	006610	1690	17088
RT3A	006620	1711	17168
RT30	010764	2157	21678
RT31	011002	2168	21788
RT32	011020	2179	21898
RT33	011036	2190	22008
RT34	011054	2201	22118
RT35	011072	2212	22228
RT36	011110	2223	22338
RT37	011126	2234	22448
RT4	006662	1709	17278
RT4A	006672	1730	17358
RT40	011144	2245	22558
RT41	011162	2256	22668
RT42	011200	2267	22778
RT43	011216	2278	22888
RT44	011234	2289	22998
RT45	011252	2300	23108
RT46	011270	2311	23238
RT47	011306	2324	23348
RT5	006734	1728	17468
RT5A	006744	1749	17548
RT50	011324	2335	23458
RT51	011342	2346	23568
RT52	011360	2357	23678
RT52A	011376	2368	23788
RT54	011414	2379	23898
RT55	011432	2390	24008
RT56	011450	2401	24118
RT57	011466	2412	24228
RT6	007006	1747	17658
RT6A	007016	1768	17738
RT60	011504	2423	24338
RT61	011522	2434	24448

TIMER	003472	1103#	1117	2512										
TIME1	003642	1103#	1112#	1122	1129#	1400	1410	2552	2577	2609	3221	3302	3497	3498
TIME14	003644	1122#	1123#	1125#	1130#	2524	2568	2774						
TKCSR	001574	743#	1156											
TKDBR	001576	744#	1158	3644										
TNP1	017042	3643#	3644#	3645#	3646	3665#	3668							
TPCSR	001600	745#	799	802	1001	1175	1178	1181						
TPDBR	001602	746#	801#	1000#	1160#	1177#	1180#							
TTDAT	001270	706#	1554#	1555#	1556	1559								
TTPTR	001276	709#												
TUNTAB	001306	714#	715	1279#	1280#	1311	1323#	1324	1336	1341	1369	1373	1377	1380
		1517	1518#	1520	1521	1563	1565	2679	2709	2802	2807	2831	3206#	3208
		3209	3240	3253	3259	3268	3279	3303#	3305	3306	3323	3333	3343	3354
		3364	3461#	3462#	3473	3475	3479	3562	3564					
TYP	002764	729	989#											
TYPA	002774	992#	999	1008										
TYPC	003012	994	996#											
TYPD	003030	998	1000#	1005	1007									
TYPDAT	003074	992#	993	996	1000	1004#	1006#	1009#						
TYPE =	104000	515#	761	792	795	847	865	908	1026	1035	1047	1053	1074	1083
		1186	1188	1644	3535	3552	3577	3581	3587	3590	3656	3662		
TYPF	003046	997	1004#											
TYPG	003060	1006#												
UNIT	003252	1050#	1051	1056#	1057#	1058#	1062							
UNTOKA	003272	1052	1056#											
UNTOKB	003316	1061#	1064											
UNTOKC	003346	1069#	1072											
VAC	001246	697#	713											
VECOK	003162	1031	1033#											
VECOKA	003172	1035#	1039	1041										
VECOKB	003200	1034	1038#											
VECTOR	003144	1029#	1030	1032#	1033	1038	1040	1042	1043#	1044				
MCT	001146	693#	694	1104#	1232	1502#	1569	2631#	2671#	2702#	2811#	3464#	3496#	3608#
		3609#	3611	3612										
WHERE	017064	1027	3672#											
WHICH	017140	1048	3681#											
X =	000146	534#	1652	1657#	1669	1674#	1688	1693#	1707	1712#	1726	1731#	1745	1750#
		1764	1769#	1783	1788#	1802	1807#	1821	1826#	1845	1850#	1877	1882#	1903
		1908#	1920	1925#	1952	1957#	1979	1984#	2006	2011#	2028	2033#	2052	2057#
		2076	2081#	2100	2105#	2118	2123#	2144	2149#	2155	2160#	2166	2171#	2177
		2182#	2188	2193#	2199	2204#	2210	2215#	2221	2226#	2232	2237#	2243	2248#
		2254	2259#	2265	2270#	2276	2281#	2287	2292#	2298	2303#	2309	2314#	2322
		2327#	2333	2338#	2344	2349#	2355	2360#	2366	2371#	2377	2382#	2388	2393#
		2399	2404#	2410	2415#	2421	2426#	2432	2437#	2443	2448#	2454	2459#	2465
		2470#	2476	2481#	2487	2492#	2498	2499	2504#	2556	2561#	2582	2587#	2613
		2618#	2656	2661#	2739	2744#	2791	2796#	2837	2842#	2848	2853#	2859	2864#
		2870	2875#	2881	2886#	2892	2897#	2903	2908#	2914	2919#	2925	2930#	2936
		2941#	2947	2952#	2958	2963#	2969	2974#	2980	2985#	2991	2996#	3002	3007#
		3015	3020#	3026	3031#	3037	3042#	3048	3053#	3059	3064#	3070	3075#	3081
		3086#	3092	3097#	3103	3108#	3114	3119#	3125	3130#	3136	3141#	3147	3152#
		3158	3163#	3169	3174#	3180	3185#	3191	3196#	3203	3208#	3214	3219#	3225
		3395#	3402	3407#	3414	3419#	3426	3431#	3438	3443#	3450	3455#	3462	3468#
XMITD	005532	1217	1283	1329	1398	1497#	1525	2595	2753	3216				
XMITDAT	001302	711#	766	814	1216#	1235#	1237	1313#	1314#	1315#	1316#	1317	1326#	1338#
		1343#	1344#	1345#	1346#	1347	1381#	1390#	1421#	1558#	1579#	1582#	1583#	1832#
		1855#	1856	1858	1863	1864#	1871#	1872	1874#	1889#	1890	1892	1914#	1930#

DZDMAB.SRC CROSS REFERENCE TABLE -- MACRO NAMES

.STRAP	18
.STYPB	18
.STYPD	18
.STYPE	18
.STYPO	18
.S4OCA	18
.1170	18

ADD	933	976	980	990	1043	1062	1112	1139	1406	1429	1470	1473	2606	2787	3290
ASL	3375														
BCC	852	932	1056	1057	1058	1089	1168	1169	1170	1407	1997	2550	2607	2788	3219
BEG	1408	1591	1600	1998	2551	2608	2789	3220							
	767	798	864	898	911	997	1138	1162	1164	1173	1226	1233	1238	1320	1325
	1337	1349	1378	1416	1420	1431	1434	1533	1557	1570	1581	1606	1680	1685	1699
	1704	1718	1723	1737	1742	1756	1761	1775	1780	1794	1799	1813	1818	1835	1841
	1859	1867	1873	1893	1896	1917	1934	1942	1948	1969	1972	2019	2025	2040	2047
	2064	2071	2088	2095	2528	2574	2641	2649	2683	2688	2713	2760	2768	2782	3225
	3232	3248	3261	3281	3289	3316	3346	3366	3374	3474	3480	3510	3513	3522	3526
	3530	3558	3629	3667											
BHI	1039	1087													
BHIS	1034	1082													
BIC	851	880	1061	1249	1288	1306	1312	1314	1335	1342	1344	1364	1367	1459	1472
	1475	1536	1539	1555	1583	1683	1702	1721	1740	1759	1778	1797	1816	1964	1994
	2537	2599	2636	2681	2711	2757	2816	3258	3272	3356	3561				
BICB	1159	1165	3645												
BIS	1241	1285	1298	1316	1346	1503	1524	1991	2016	2022	2111	2522	2523	2538	2569
	2812														
BISB	1171	1315	1345												
BIT	782	797	875	897	900	904	1040	1373	1527	1532	1548	1861	1869	1895	1936
	1944	1971	2533	2645	3231	3335									
BITB	1166														
BLOS	1052	3586													
BMI	1229	1291	1370	1541	2652	2778	2820	2832	3242	3244	3256	3271	3326		
BNE	783	876	882	885	901	903	905	907	994	1031	1041	1064	1072	1079	1113
	1119	1167	1222	1224	1293	1374	1405	1414	1436	1457	1480	1491	1516	1528	1549
	1564	1594	1602	1621	1862	1870	1901	1937	1945	1977	1993	1996	2519	2534	2646
	2806	2826	3283	3330	3336	3340	3351	3358	3368	3507	3520	3540	3563	3642	3647
	3655														
BPL	800	803	805	1002	1157	1176	1179	1182	1251	1287	1308	1332	1334	1363	1366
	1530	1538	1639	2598	2601	2635	2638	2674	2676	2705	2707	2756	2762	2815	2828
	2830	3311	3469	3484	3560										
BR	777	837	868	878	887	999	1008	1037	1055	1085	1117	1120	1140	1174	1190
	1295	1303	1310	1322	1328	1340	1372	1376	1418	1437	1531	1562	1572	1604	1608
	1682	1701	1720	1739	1758	1777	1796	1815	1837	1843	1875	1899	1950	1975	2003
	2005	2021	2044	2068	2092	2133	2140	2532	2536	2546	2605	2647	2686	2690	2715
	2765	2773	2780	2786	2823	3230	3235	3246	3250	3263	3265	3273	3286	3291	3320
	3328	3338	3348	3372	3376	3478	3515	3524	3528	3546	3566	3573	3589	3630	
CLC	1124	1550													
CLR	892	893	894	1103	1110	1114	1121	1136	1154	1180	1216	1220	1242	1279	1280
	1289	1299	1323	1326	1338	1368	1379	1381	1390	1396	1518	1578	1579	1589	1634
	1647	1650	1832	1839	1855	1864	1865	1897	1914	1930	1939	1940	1973	2004	2015
	2023	2112	2137	2529	2531	2539	2543	2547	2572	2594	2623	2654	2667	2677	2678
	2696	2708	2749	2751	2784	2804	2809	2810	2824	3206	3212	3226	3228	3229	3234
	3238	3276	3284	3303	3312	3361	3461	3462	3470	3481	3499	3541	3542	3567	3568
	3569	3643													
CLRB	931	2038	2046	2062	2070	2086	2094								
CMP	863	884	906	1033	1038	1051	1081	1086	1137	1237	1247	1305	1317	1347	1419
	1563	1580	1679	1698	1717	1736	1755	1774	1793	1812	1858	1872	1892	1933	1947
	1968	1992	2039	2063	2087	2134	2549	2573	2639	2687	2759	2767	2825	3280	3315
	3365	3473	3562	3585	3628	3631	3641	3654	3666						
CMPB	766	881	993	996	1161	1163	1556	2682	2712	3260	3646				
COM	967	1455	1456	1592											
DEC	902	1063	1071	1118	1172	1221	1223	1404	1413	1435	1479	1490	1515	1593	1620
	1900	1976	2518	2805	3241	3255	3270	3282	3287	3329	3339	3350	3357	3367	3373

SEC	1547	1636	2650												
SUB	928	1088	1433	1638	1665										
TRAP	529														
TST	804	834	1030	1078	1183	1228	1250	1253	1286	1296	1331	1369	1540	1566	1569
	1601	1605	1662	1684	1703	1722	1741	1760	1779	1798	1817	1995	2567	2597	2648
	2673	2689	2698	2704	2761	2777	2781	2819	2827	2831	3218	3243	3247	3264	3288
	3325	3483	3539	3557	3565										
TSTB	799	802	1001	1156	1175	1178	1181	1290	1307	1333	1362	1365	1377	1529	1537
	2600	2630	2634	2637	2675	2706	2755	2814	2829	3310	3345	3349	3468	3559	
.ASCII	3672	3675	3677	3680	3681	3684	3687	3689	3692	3694	3696	3697	3701	3707	3711
	3718	3721	3725	3730	3735	3739	3740	3742	3745						
.BYTE	3744														
.ENABL	1	422													
.END	3756														
.LIST	1	421	424												
.MACR	424														
.MACRO	1														
.NLIST	1	420	424												
.REM	4														
.REPT	2144	2322	2837	3015											
.TITLE	419														

ERRORS DETECTED: 0
 DEFAULT GLOBALS GENERATED: 0

*.NOW.SEQ/SOL/CRF/PAGNUM/NL:TOC=SYSMAC.CO,DZDMAB.SRC
 RUN-TIME: 36 49 5 SECONDS
 RUN-TIME RATIO: 232/92=2.5
 CORE USED: 41K (81 PAGES)

