

DJ11

DJ11 LGC TEST
CZDJAFO

AH-8504F-MC

COPYRIGHT '72-79

FICHE 1 OF 1

SEP 1979

digital

MADE IN USA

The microfiche card contains a grid of 150 frames (10 columns by 15 rows). Each frame is a tiny, high-contrast image of a document page, likely containing text and possibly some diagrams or tables. The text is too small to be legible. The frames are arranged in a regular grid pattern across the left side of the card.

.REM %

IDENTIFICATION

PRODUCT CODE: AC-8502F-MC
PRODUCT NAME: CZDJAFO DJ11 LGC TEST
PROGRAM DATE: FEB 1979
MAINTAINER: DIAGNOSTIC ENGINEERING

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48

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

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

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

COPYRIGHT (C) 1972, 1979 BY DIGITAL EQUIPMENT CORPORATION

ZZ- (ZDJA-F-D DJ11 LOGIC TESTS
TABLE OF CONTENTS

CONTENTS

49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81

1.	ABSTRACT
2.	REQUIREMENTS
2.1	EQUIPMENT
2.2	STORAGE
2.3	PRELIMINARY PROGRAMS
3.	LOADING PROCEDURE
4.	STARTING PROCEDURE
4.1	CONTROL SWITCH SETTINGS
4.2	STARTING ADDRESS
4.3	PROGRAM AND OPERATOR ACTION
5.	OPERATING PROCEDURE
5.1	OPERATIONAL SWITCH SETTINGS
5.2	SUBROUTINE ABSTRACTS
5.3	PROGRAM AND OPERATOR ACTION
6.	ERRORS
6.1	ERROR PRINTOUT
6.2	ERROR RECOVERY
6.3	ERROR COUNTER
7.	RESTRICTIONS
8.	MISCELLANEOUS
8.1	EXECUTION TIME
8.2	STACK POINTER
8.3	PASS COUNTER
8.4	POWER FAIL
9.	PROGRAM DESCRIPTION

ZZ- CZDJA-F-D DJ11 LOGIC TESTS
DESCRIPTION

82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137

1. ABSTRACT

THIS PROGRAM TESTS THE LOGIC OF THE DJ11 ASYNCHRONOUS MULTIPLEXER IN MAINTENANCE MODE. IT CHECKS THAT ALL THE CONTROL REGISTERS FUNCTION PROPERLY, THAT INTERRUPTS OCCURE AT THE RIGHT LEVEL, AND THAT DATA CAN BE TRANSMITTED AND RECEIVED CORRECTLY. THIS PROGRAM DOES NOT TEST THAT THE INPUT AND OUTPUT LEAD CONNECTIONS ARE FUNCTIONAL. (SEE MAINDEC-11-DZDJB, PROGRAMS 2 AND 3 FOR ON-LINE TESTING). THE PROGRAM SHOU'D BE RUN FOR AT LEAST 2 PASSES WITH ALL SWITCHES DOWN.

NOTE: THE PROGRAM WILL AUTOSIZE THE SILO ALARM LEVEL OF THE DJ11 AND WILL RUN AT ANY ALARM LEVEL SET. AND WILL RUN AT ANY ALARM LEVEL SET.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 STANDARD COMPUTER WITH CONSOLE TELETYPE UP TO 16 DJ11 ASYNCHRONOUS MULTIPLEXERS.

2.2 STORAGE

THIS PROGRAM USES ALL OF 8K, EXCEPT ABS LOADER.

2.3 PRELIMINARY PROGRAMS

NONE

3. LOADING PROCEDURE

USE STANDARD PROCEDURE FOR ABS TAPES.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SFE 5.1 (ALL DOWN FOR WORST CASE TESTING)

4.2 STARTING ADDRESS

THE PROGRAM SHOULD ALWAYS BE STARTED AT 200. IT MAY BE

138

RESTARTED AT 1000 AFTER ALL PARAMETERS HAVE BEEN SELECTED.

139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194

ZZ- CZDJA-F-D DJ11 LOGIC TESTS
DESCRIPTION

PAGE 4

4.3 PROGRAM AND OPERATOR ACTION

- 1) LOAD PROGRAM INTO MEMORY USING ABS LOADER.
- 2) LOAD ADDRESS 200.
- 3) IF HARDWARE SWITCH REGISTER IS AVAILABLE, SET SWITCHES (SEE SEC. 5.1), ALL DOWN FOR WORST CASE, PRESS START.
- 4) IF SWITCH-LESS PROCESSOR SIMPLY PRESS START.
- 5) ENTER PARAMETERS (SEE SEC. 5.3) AS THEY ARE REQUESTED.
- 6) THE PROGRAM WILL LOOP AND BELL WILL RING ONCE EVERY PASS. EOP' IS ALSO PRINTED ON EACH PASS.
- 7) A MINIMUM OF TWO PASSES SHOULD ALWAYS BE RUN.

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

AT SA 200, ALL SWITCHES DOWN IS WORST CASE TESTING. EACH SUBTEST WILL BE LOOPED UPON UNTIL COMPLETION OF 16 PASSES OF THAT SUBTEST. THE BELL WILL RING UPON COMPLETION OF A PASS OF THE ENTIRE PROGRAM.

THE SWITCH SETTINGS ARE:

- SW<15> = 1 HALT ON ERROR
- SW<14> = 1 SCOPE LOOP
- SW<13> = 1 INHIBIT PRINTOUT
- SW<12> = 1 PRINT SILO ALARM LEVEL (IN OCTAL)
- SW<11> = 1 INHIBIT ITERATIONS OF SUBTEST
- SW<10> = 1 BELL ON ERROR
- 0 BELL ON PASS COMPLETE
- SW<09> = 1 LOOP ON ERROR
- SW<08> = 1 LOOP ON TEST IN SW<7:0>

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

(I.E.) 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

195

COMMITTED KEYING IN SWREG VALUE.

196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251

ZZ- CZDJA-F-D DJ11 LOGIC TESTS
DESCRIPTION

PAGE 5

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.

5.2 SUBROUTINE ABSTRACTS

5.2.1 SCOPE

THIS SUBROUTINE CALL (VIA A TRAP INSTRUCTION) IS PLACED BETWEEN EACH SUBTEST IN THE INSTRUCTION SECTION. IT RECORDS THE STARTING ADDRESS OF EACH SUBTEST AS IT IS BEING ENTERED IN LOCATION 'LAD'. IF A SCOPE LOOP IS REQUESTED, THE CURRENT SUBTEST WILL BE LOOPED UPON. SW<11> ON A 1 INHIBITS ITERATION OF SUBTESTS. THE CONTENTS OF 'LAD' MAY BE USED TO DETERMINE THE LAST SUBTEST SUCCESSFULLY COMPLETED.

5.2.2 HLT

THIS ROUTINE (CALLED BY AN EMT INSTRUCTION) PRINTS OUT AN ERROR MESSAGE (SEE 6.1). IF SW<9> IS ON A 1 AND A HLT IS EXECUTED, THE SUBTEST WILL BE LOOPED UPON UNTIL 16 CONSECUTIVE GOOD PASSES ARE COMPLETED. TO INHIBIT TYPEOUTS, PUT SW<13> ON A 1. TO RING THE BELL ON AN ERROR, PUT SW<10> ON A 1.

5.2.4 ALMCK

IN THE NORMAL OPERATION THE 'DONE' BIT IS SET AS EACH CHARACTER IS READ INTO THE FI/FO BUFFER (SILO). BUT THIS 'DONE' CONDITION CAN BE DELAYED TO CAUSE DONE ON THE 5, 9, OR 17 CHARACTER. THIS IS DONE BY CUTTING ONE OF THE JUMPERS (W1,W2,W3) ON THE M7285 CONTROL BOARD. THE PROGRAM TESTS FOR THIS 'SILO ALARM LEVEL' AND IF SW12 IS SET (1) IT WILL PRINT OUT THE LEVEL AT WHICH EACH DJ11 WILL SET 'DONE.' THE SUBROUTINE ALSO ADJUSTS THE CHARACTER COUNTERS TO ENSURE THAT THE MAXIMUM NUMBER OF CHARACTERS TO BE TRANSFERED IS A MULTIPLE OF THE SILO ALARM LEVEL. THIS ENSURES THAT ALL DATA WILL BE READ OUT OF THE SILO. DONE WILL NOT SET IF THE NUMBER OF CHARACTERS IN THE FI/FO BUFFER IS LESS THAN THE SILO ALARM LEVEL. (NOTE CHARACTER PRESENT IS SET ON EACH CHARACTER IN THE BUFFER, REGARDLESS OF THE SILO ALARM LEVEL.)

5.2.5 TRAPCATCHER

252
253

A ".+2" - "HALT" SEQUENCE IS REPEATED FROM 0 - 56 TO DETECT

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
301
302
303
304
305
306
307
308
309

ZZ- CZDJA-F-D DJ11 LOGIC TESTS
DESCRIPTION

PAGE 6

ANY UNEXPECTED TRAPS AND A ".+2" - "IOT" SEQUENCE IS REPEATED FROM 60 - 776 TO DETECT ANY UNEXPECTED INTERRUPTS. THUS ANY UNEXPECTED TRAPS WILL HALT AT THE VECTOR + 2. ANY UNEXPECTED INTERRUPTS WILL RESULT IN AN ERROR "HLT" IN "IOTRAP".

5.3 PROGRAM AND OPERATOR ACTION

THE FOLLOWING REQUESTS ARE MADE TO THE OPERATOR AT THE BEGINNING OF THE PROGRAM. A DETAILED DESCRIPTION OF WHAT IS REQUIRED FOR EACH PARAMETER IS GIVEN BELLOW.

- 1) 'FIRST DJ11 ADDRESS: ''
THE CSR ADDRESS OF THE FIRST DJ11 YOU WISH TO TEST. MUST BE BETWEEN 160000(8) AND 177777(8). THE DEFAULT (CARRIAGE RETURN) IS TO 160010(8). 'P(REVIOUS)'' SELECTS THE ADDRESS PREVIOUSLY SELECTED.
- 2) 'VECTOR ADDRESS: ''
THE RECEIVER INTERRUPT VECTOR ADDRESS OF THE FIRST DJ11 YOU WISH TO TEST. MUST BE BETWEEN 300(8) AND 1000(8). THE DEFAULT (CARRIAGE RETURN) IS TO 300(8). 'P(REVIOUS)'' SELECTS THE ADDRESS PREVIOUSLY SELECTED.
- 3) 'NO. OF DJ11'S: ''
THE NUMBER OF DJ11 UNITS YOU WISH TO TEST AT ONE TIME. MUST BE BETWEEN 1 AND 16. THE DEFAULT (CARRIAGE RETURN) IS TO 1. 'P(REVIOUS)'' SELECTS THE NUMBER OF UNITS PREVIOUSLY SELECTED.
- 4) 'STANDARD CONFIGURATION? ''
'Y(ES)'' OR DEFAULT (CARRIAGE RETURN) SELECTS 8 LEVEL CODE, NO PARITY. 'N(O)'' CAUSES REQUESTS FOR CODE LEVEL AND PARITY ON ALL REQUESTED LINES IN GROUPS OF FOUR. 'P(REVIOUS)'' SELECTS THE CODE LEVELS AND PARITIES PREVIOUSLY SELECTED.
- 5) 'CHAR LENGTH: ''
THE CODE LEVEL FOR THE LINE GROUP SPECIFIED. MUST BE 5, 6, 7, OR 8. THE DEFAULT (CARRIAGE RETURN) IS TO 8 LEVEL CODE.
- 6) 'PARITY (NO, ODD, EVEN): ''
THE TYPE OF PARITY SELECTED FOR THE LINE GROUP SPECIFIED. THE DEFAULT (CARRIAGE RETURN) IS TO NO PARITY.

6. ERRORS

6.1 ERROR PRINTOUT

THE FORMAT IS AS FOLLOWS:

310
311

ADR DJ4DR (R1) (R2) (R3) (R4)

ZZ- CZDJA-F-D DJ11 LOGIC TESTS
DESCRIPTION

PAGE 7

312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367

WHERE:

ADR = ADDRESS OF ERROR HLT
DJADR = CSR ADDRESS OF DJ11 UNDER TEST
(RN) - CONTENTS OF GENERAL REGISTER 'N'. FROM NONE TO FOUR OF THESE MAY BE TYPED DEPENDING ON THE NUMBER FOLLOWING THE HLT; E.G., HLT+3 WOULD TYPE (R1) THRU (R3); HLT (BY ITSELF) WOULD STOP AFTER TYPING ADR AND DJADR.

TO FIND THE FAILING TEST, LOOK AT THE LISTING ABOVE THE ADDRESS TYPED. IN MOST CASES THE COMMENT BESIDE THE HLT TELLS WHAT WAS BEING CHECKED AND WHAT WAS EXPECTED. ALSO, A LIST OF THE PROBABLE FAILING LOGIC IS GIVEN IN THE COMMENTS AT THE BEGINNING OF THE TEST.

6.2 ERROR RECOVERY

RESTART AT 200 OR 1000.

6.3 ERROR COUNTER

AN ERROR COUNT IS KEPT IN 'ERRORS'. IT CAN BE CLEARED FROM THE CONSOLE, BY RESTARTING AT 200, OR BY RELOADING THE PROGRAM.

7. RESTRICTIONS

IF MORE THAN ONE DJ11 IS TESTED AT A TIME, THE DEVICE ADDRESSES AND THE VECTOR ADDRESSES MUST ALL BE CONTIGUOUS.

IF THIS PROGRAM IS RUN WITH A MONITOR, I.E. ACT11 OR DDP, THE DEVICE ADDRESSES MUST FOLLOW THE FLOATING ADDRESS CONVENTION. DJ11'S WILL BE FIRST, STARTING AT 160010..

8. MISCELLANEOUS

8.1 EXECUTION TIME

DUE TO THE VARIOUS BAUD RATES AVAILABLE AND THE ABILITY TO
7) "WHAT ARE THE BR PRIORITIES? FOR:"
"DEVICE 1=" ETC.
"DEVICE 2=" ETC.
THIS WILL GO TO THE NUMBER OF DEVICES SPECIFIED
IN QUESTION 3 ABOVE. PRIORITIES ALLOWED ARE 0 THROUGH 7.
CHECK UP TO 16 DJ11'S AT ONCE, THE EXECUTION TIME CAN BE
ANYWHERE FROM 15 SECONDS TO THREE AND A HALF HOURS. THE

368
369
370
371
372
373
374

FOLLOWING TYPICAL TIMES ARE FOR ONE DJ11 WITH ALL LINES AT THE SAME SPEED, 8 LEVEL CODE, 2 STOP BITS, AND NO PARITY ON A PDP-11/20 WITHOUT TRACE TRAPPING. FOR MULTIPLE DJ11'S, MULTIPLY THESE TIMES BY THE NUMBER OF UNITS SELECTED FOR TEST.

BAUD RUN TIME

ZZ- CZDJA-F-D DJ11 LOGIC TESTS
DESCRIPTION

PAGE 8

375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430

75 00:26:00
110 00:18:00
134.5 00:15:00
150 00:13:00
300 00:06:30
600 00:03:15
1200 00:01:45
1800 00:01:20
2400 00:00:55
4800 00:00:30
9600 00:00:15

8.2 STACK POINTER
STACK IS INITALLY SET TO 1200

8.3 PASS COUNT
A 32 BIT (2 WORDS) PASS COUNT IS KEPT IN PCNT. IT CAN BE
CLEARED FROM THE CONSOLE, BY RESTARTING AT 200, OR BY
RELOADING THE PROGRAM.

8.4 POWER FAIL
EACH TEST CAN BE POWER FAILED WITH NO ERRORS. TO USE, START
THE TEST AS USUAL AND POWER DOWN THEN UP AT ANY TIME. THE
ROUTINE SHOULD TYPE 'POWER' AND RESTART THE PROGRAM WITH NO
OTHER ERROR TYPEOUTS.

9. PROGRAM DESCRIPTION
THIS PROGRAM TESTS THE LOGIC OF UP TO 16 DJ11 ASYNCHRONOUS
DATA MULTIPLEXERS IN MAINTENANCE MODE. IT CHECKS THAT ALL
THE CONTROL REGISTERS FUNCTION PROPERLY, THAT INTERRUPTS
OCCURE AT THE RIGHT PRIORITY LEVEL, AND THAT DATA CAN BE
TRANSMITTED AND RECEIVED CORRECTLY. THE PROGRAM HAS MANY
SUBTESTS (THE CODE BETWEEN 2 SCOPE STATEMENTS) WHICH ARE RUN
16 TIMES BEFORE CONTINUING TO THE NEXT. SW<11> ON A 1
CAUSES EACH SUBTEST TO BE RUN ONLY ONCE. SW<9> ON A 1
ENABLES LOOP ON ERROR. THE ADDRESS 'ICNT' CONTAINS THE
ITERATION COUNT IN THE LEFT BYTE AND THE TEST NUMBER IN THE
RIGHT BYTE. ALL THE SUBTESTS SHOULD BE RUN SEQUENTIALLY BY
STARTING AT 200 NOT BY STARTING AT THE BEGINNING OF THE
SUBTEST. TO LOOP ON A PARTICULAR SUBTEST, PUT THE TEST
NUMBER (SEE LISTING) IN THE RIGHT BYTE OF THE SWITCH
REGISTER AND SW<8> ON A 1. THIS TEST WILL BE LOOPED UPON
UNTIL SW<8> IS PUT ON A 0 OR THE RIGHT BYTE IS CHANGED. IF

45
43

THE TEST IS NON-EXISTANT, THE PROGRAM WILL BE RUN AS USUAL,
IF MORE THAN ONE DJ11 IS SELECTED, ALL THE SUBTFSTS ARE

433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488

ZZ- CZDJA-F-D DJ11 LOGIC TESTS
 DESCRIPTION

PAGE 9

PERFORMED ON ONE UNIT AT A TIME. THE BELL WILL NOT RING AND
 EOP PRINT, UNTIL A PASS HAS BEEN MADE ON EACH OF THE DJ11'S
 SELECTED FOR TEST.

%

.TITLE ZZ-CZDJA-F-0 DJ11 LOGIC TESTS
 .ENABLE ABS
 .ENABLE AMA
 :COPYRIGHT 1972,1979 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.

	SWITCH	USE
	SW15= 100000	:HALT ON ERROR
	SW14= 40000	:LOOP ON TEST
	SW13= 20000	:INHIBIT ERROR TYPEOUTS
	SW12= 10000	:PRINT SILO ALARM LEVEL
	SW11= 4000	:INHIBIT ITERATIONS
	SW10= 2000	:0 - BELL ON PASS COMPLETE
		:1 - BELL ON ERROR
	SW9= 1000	:LOOP ON ERROR
	SW8= 400	:LOOP ON TEST IN SW<7:0>
	.REM!	

DJ11 REGISTER BIT ASSIGNMENTS:

CONTROL STATUS REGISTER (CSR) XXXXX0

BIT0	RECEIVER ENABLE (READ/WRITE)
BIT1	HALF DUPLEX SELECT (READ/WRITE)
BIT2	MAINTENANCE (READ/WRITE)
BIT3	CLEAR MOS (WRITE ONLY)
BIT4	CLEAR MOS FLAG (READ ONLY)
BIT5	NOT USED
BIT6	RECEIVER INTERRUPT ENABLE (READ/WRITE)
BIT7	DONE (READ ONLY)
BIT8	MASTER TRANSMITTER SCAN ENABLE (READ/WRITE)
BIT9	NOT USED
BIT10	READ/WRITE BREAK REGISTER (READ/WRITE)
BIT11	NOT USED
BIT12	STATUS ENABLE (READ/WRITE)
BIT13	FI/FO OVERRUN (READ ONLY)
BIT14	MASTER TRANSMITTER INTERRUPT ENABLE (READ/WRITE)
BIT15	TRANSMITTER READY (READ ONLY)

RECEIVER BUFFER REGISTER (RBUF) XXXXX2 (READ ONLY)

BIT0-7	RECEIVED CHARACTER
BIT8-11	LINE NUMBER
BIT12	PARITY ERROR
BIT13	FRAMING ERROR
BIT14	UART OVERRUN ERROR
BIT15	CHARACTER PRESENT

489
 490
 491
 492
 493
 494
 495
 496
 497
 498
 499
 500
 501
 502
 503
 504
 505
 506
 507
 508
 509
 510
 511
 512
 513
 514
 515
 516
 517
 518
 519
 520
 521
 522
 523
 524
 525
 526
 527
 528
 529
 530
 531
 532
 533
 534
 535
 536

104400
 104000
 000004
 177776
 000000
 000001
 000002
 000003
 000004
 000005
 000005
 000006
 000007
 000007
 000001
 000002
 000004
 000010
 000020
 000040
 000100
 000200
 000400
 001000
 002000
 004000
 010000
 020000
 040000
 100000
 000340
 000000
 000020
 001200

TRANSMITTER CONTROL REGISTER (TCR) XXXXX4 (READ/WRITE)
 BIT0-15 STOP THE SCANNER ON CORRESPONDING LINE
 TRANSMITTER BUFFER (TBUF) XXXXX6
 BIT0-7 TRANSMITTED CHARACTER (WRITE ONLY)
 BIT8-11 LINE NUMBER (READ ONLY)
 BREAK CONTROL STATUS REGISTER (BCSR) XXXXX4 (BIT10 OF CSR SET) (READ/WRITE)
 BIT0-15 TRANSMIT A BREAK ON CORRESPONDING LINE!

SCOPE= TRAP
 HLT= EMT
 TYPE= IOT
 PS= 177776
 R0= %0
 R1= %1
 R2= %2
 R3= %3
 R4= %4
 R5= %5
 TTY= %5
 SP= %6
 PC= %7
 BELL= 7
 BIT0= 1
 BIT1= 2
 BIT2= 4
 BIT3= 10
 BIT4= 20
 BIT5= 40
 BIT6= 100
 BIT7= 200
 BIT8= 400
 BIT9= 1000
 BIT10= 2000
 BIT11= 4000
 BIT12= 10000
 BIT13= 20000
 BIT14= 40000
 BIT15= 100000
 LEVEL7 = 340
 OPEN = 0
 DJMXNO=16.
 STACK =1200

;MAX ALLOWED NUMBER OF DJ11'S

537		000000		. = 0					
538	000000	000000	000000		0,0				:TRAP CATCHER IN LOCATIONS 0 THRU 776
539									:LOCATIONS 0 AND 2 CONTAIN 'HALT' INSTRUCTIONS
540									:LOCATIONS 4 THRU 56 CONTAIN ''+2'' AND 'HALT' IN EVERY VECTOR
541									:LOCATIONS 60 THRU 776 CONTAIN ''+2'' AND 'IOT' IN EVERY VECTOR
542									
543		000046		. -	46				
544	000046	014450			\$ENDAD				
545									
546		000174		. -	174				
547	000174	000000			DISPREG:0				
548	000176	000000			SWREG: 0				
549									
550		000200		. =	200				
551	000200	000137	001402		JMP	BEGIN			:200 ALWAYS IS THE STARTING ADDRESS
552		001000		. =	1000				:RESTART ADDRESS
553	001000	000137	002344		JMP	QABR			
554									
555		001200		. -	1200				
556					ADDRESS TABLES				
557									
558									
559									: 'LENGTH' TABLE IS SET SO THE FIRST FOUR BYTES (2 WORDS) CONTAIN
560									THE CHARACTER LENGTH FOR THE 4 LINE GROUPS OF THE FIRST DJ11, ETC.
561									
562									: 'PARITY' TABLE HAS 4 WORDS, ONE FOR EACH LINE GROUP. THE FIRST WORD CONTAINS
563									THE ODD PARITY FLAGS FOR LINES 0-3 OF ALL DJ11'S. THE NEXT WORD
564									HAS PARITY FLAGS FOR LINES 4-7, ETC. EACH BIT IN THE WORD REPRESENTS
565									EACH DJ11. BIT 0 IN EACH OF THE FOUR WORDS IS FOR DJ/0 TO BIT
566									15 FOR DJ/15.
567	001200	000100		LENGTH:	.BLKB 100				:TABLE OF CHAR. LENGTHS (MASK)
568	001300	000000		PARITY:	0				:ODD PARITY FLAGS FOR LINES 0-3
569	001302	000000		PARIT2:	0				:ODD PARITY FLAGS FOR LINES 4-7
570	001304	000000		PARIT3:	0				:ODD PARITY FLAGS FOR LINES 10-13
571	001306	000000		PARIT4:	0				:ODD PARITY FLAGS FOR LINES 14-17
572	001310	000000		ICNT:	0				:ITERATION COUNT-LH, TEST NO.-RH
573	001312	000000		ERRORS:	0				:ERROR COUNT REGISTER
574	001314	000000	000000	PCNT:	0,0				:PASS COUNT REGISTER
575									
576	001320	160010		CSR:	160010				:CONTROL STATUS REGISTER(DJ UNDER TEST)
577	001322	160012		RBUF:	160012				:RECEIVER BUFFER REGISTER(DJ UNDER TEST)
578	001324			TCR:					:TRANSMITTER CONTROL REGISTER(DJ UNDER TEST)
579	001324	160014		BCSR:	160014				:BREAK STATUS REGISTER(DJ UNDER TEST)
580	001326	160016		TBUF:	160016				:TRANSMITTER BUFFER REGISTER(DJ UNDER TEST)
581									
582	001330	000300		RCVVEC:	300				:RECEIVER INTERRUPT VECTOR ADDRESS(DJ UNDER TEST)
583	001332	000302		RCVLVL:	302				
584	001334	000304		XMTVEC:	304				:TRANSMITTER INTERRUPT VECTOR ADDRESS(DJ UNDER TEST)
585	001336	000306		XMTLVL:	306				
586									
587	001340	160010		DEVADR:	160010				:FIRST DEVICE ADDRESS(SELECTED)
588	001342	000300		VECADR:	300				:FIRST VECTOR ADDRESS(SELECTED)
589	001344	000001		UNITS:	1				:NUMBER OF UNITS TO BE TESTED
590	001346	000000		TIMER:	0				:USED TO SAVE RELATIVE TIMES
591	001350	000000		ALMFLG:	0				:TEST FLAG FOR SILO ALARM LEVEL
592	001352	000000		TIMERA:	0				:TIME COUNTERS

593	001354	000000	TIMERS:	0	
594	001356	000000	COUNT:	0	:CHAR COUNTER
595	001360	000000	SUM:	0	:SUMATION OF ALARM LEVEL CHAR'S
596					
597	001362	000000	DJUUT:	0	:UNIT NUMBER OF DJ11 UNDER TEST
598	001364	000000	DJLEN:	0	:CHAR MASK TABLE POINTER
599	001366	000000	DJPAR:	0	:UNIT FLAG (FOR ODD PARITY FLAG)
600	001370	160010	DEFADR:	160010	:DEFAULT FIRST DEVICE ADDRESS
601	001372	000300	DEFVEC:	300	:DEFAULT FIRST VECTOR ADDRESS
602					
603	001374	177570	SWR:	177570	
604	001376	177570	DISPLAY:	177570	
605					
606					
607	001400	177777	FTIME:	-1	
608					
609					

```

610
611 001402 012706 001200      BEGIN:  MOV    #STACK, SP      ;SET UP STACK POINTER
612 001406 004737 017524      JSR    PC, SUSWRR          ;CHECK FOR SWITCH REGISTER
613 001412 012700 000020      MOV    #20,R0
614 001416 012720 016364      MOV    #IOTRAP,(R0)+      ;IOT VECTOR (20)
615 001422 012720 000340      MOV    #340,(R0)+
616 001426 012720 016224      MOV    #PDOWNS,(R0)+     ;POWER FAIL VECTOR (24)
617 001432 012720 000340      MOV    #340,(R0)+
618 001436 012720 015254      MOV    #EMTS,(R0)+       ;EMT VECTOR (30)
619 001442 012720 000340      MOV    #340,(R0)+
620 001446 012720 015112      MOV    #TRAPS,(R0)+      ;TRAP VECTOR (34)
621 001452 012720 000340      MOV    #340,(R0)+
622 001456 005037 001312      CLR    ERRORS             ;CLEAR ERROR COUNTER
623 001462 005037 001314      CLR    PCNT              ;CLEAR PASS COUNTER
624 001466 005037 001316      CLR    PCNT+2
625 001472 005037 001350      CLR    ALMFLG
626 001476 005737 000042      TST    @#42              ;CHECK FOR ACT11 OR DDP PRESENT
627 001502 001404              BEQ    GETADR            ;BRANCH IF NONE
628 001504 004737 017304      JSR    PC, AUTO          ;GO TO SUBROUTINE TO 'MAP' DJ11 ON THE SYS
629 001510 000137 002344      JMP    QABR              ;SKIP OPERATOR ACTION
630
631 001514 000004 017013      GETADR: TYPE, MTITLE
632 001520 000004 016524      TYPE, RETURN
633 001524 000004 016532      TYPE, MSGADR
634 001530 004537 015476      JSR    R5, READIN        ;TYPE 'FIRST DJ11 ADDRESS'
635 001534 001340              .WORD DEVADR             ;READ INPUT FROM TTY AND SAVE
636 001536 001366              BNE   GETADR            ; IN DEVADR
637 001540 005737 001340      TST    DEVADR           ;BRANCH IF BAD INPUT
638 001544 001003              BNE   1$
639 001546 013737 00370 001340      MOV    DEFADR,DEVADR
640 001554 042737 000007 001340 1$:  BIC    #7,DEVADR
641 001562 022737 160000 001340      CMP    #160000,DEVADR
642 001570 101351              BHI   GETADR
643
644 001572 000004 016562      GETVEC: TYPE, MSGVEC
645 001576 004537 015476      JSR    R5, READIN        ;TYPE 'VECTOR ADDRESS:'
646 001602 001342              .WORD VECADR            ;READ INPUT FROM TTY AND SAVE
647 001604 001372              BNE   GETVEC           ; IN VECADR
648 001606 005737 001342      TST    VECADR           ;BRANCH IF BAD INPUT
649 001612 001003              BNE   1$               ;CHECK FOR CR
650 001614 012737 000300 001342      MOV    #300, VECADR     ;SET TO FIRST FLOATING VECTOR
651 001622 042737 000007 001342 1$:  BIC    #7, VECADR      ;CLEAR TO MODULO 10
652 001630 022737 000300 001342      CMP    #300, VECADR    ;CHECK FOR LOW LIMIT
653 001636 003355              BGT   GETVEC
654 001640 022737 001000 001342      CMP    #1000, VECADR   ;CHECK FOR UPPER LIMIT
655 001646 003751              BLE   GETVEC
656 001650 000004 016606      GETNUM: TYPE, MSGNUM
657 001654 004737 015644      JSR    PC,READS         ;TYPE 'NUMBER OR UNITS:'
658      ; CHECK STRING FOR VALID DIGITS, TERMINATOR, AND 'P'
659 001660 012702 015772      MOV    #INPUT,R2       ;READ INPUT FROM THE TTY
660 001664 112201              MOVB  (R2)+,R1         ;POINTS TO INPUT STRING
661 001666 001434              BEQ   1$              ;LOAD 1ST CHAR
662 001670 122701 000120      CMPB  #'P',R1         ;BRANCH IF IMMEDIATE TERMINATOR
663 001674 001443              BEQ   CONFIG          ;CHECK FOR A P
664 001676 120127 000071      CMPB  R1,#71          ;CONFIGURE MODE IF SO
665 001702 101362              BHI   GETNUM          ;MUST BE A VALID ASCII NUMBER
                        ;ELSE RETRY

```

666	001704	162701	000060		SUB #60,R1		:STRIP ASCII CODE
667	001710	003757			BLE GETNUM		:SHOULDN'T BE ZERO OR NEG
668						:1ST CHAR IS A VALID DIGIT, 1 THRU 9 - CHECK REST OF STRING	
669	001712	105722			TSTB (R2)+		:2ND CHAR A TERMINATOR?
670	001714	001423			BEO 3\$:YES - R1 HAS THE FINAL # OF UNITS
671	001716	105712			TSTB (R2)		:NO - NXT CHAR MUST BE THE TERMINATOR
672	001720	001353			BNE GETNUM		:ELSE RETRY.
673	001722	124227	000071		CMPB -(R2),#71		:CHECK 2ND CHAR FOR A VALID NUMBER
674	001726	101350			BHI GETNUM		
675	001730	111203			MOVB (R2),R3		:MAKE IT MORE ACCESSIBLE
676	001732	162703	000060		SUB #60,R3		:STRIP ASCII CODE
677	001736	100744			BMI GETNUM		:SHOULDN'T BE NEGATIVE
678						:BOTH LOW AND HIGH ORDER DIGITS ARE OK - CONVERT DECIMAL VALUE.	
679	001740	010146			MOV R1,-(SP)		:SAVE IT FOR LATER
680	001742	006301			ASL R1		:MULT HI ORDER BY 8
681	001744	006301			ASL R1		
682	001746	006301			ASL R1		
683	001750	006316			ASL (SP)		:MULTIPLY IT BY 2
684	001752	062601			ADD (SP)+,R1		:RESULT IS (HI ORD)*10.
685	001754	060301			ADD R3,R1		:NOW ADD IN THE LOW ORDER
686	001756	000402			BR 3\$		
687	001760	012701	000001		1\$: MOV #1,R1		:LOAD DEFAULT VALUE HERE
688					:R1 HAS THE CONVERTED VALUE - COMPARE AGAINST MAX ALLOWED.		
689	001764	020127	000020		3\$: CMP R1,#D.MXNO		:TOO BIG?
690	001770	101327			BHI GETNUM		:YES - RETRY.
691	001772	010137	001344		MOV R1,UNIS		:VALUE OK - STORE RESULT
692	001776	012737	000100	001362	MOV #100,DJUUT		:PRIME UNIT UNDER TEST NUMBER
693							
694	002004	000004	016631		CONFIG: TYPE, MSGCON		:TYPE "STANDARD CONFIG?"
695	002010	004737	015644		JSR PC, READS		:READ INPUT FROM ITY
696	002014	122737	000120	015772	CMPB #P, INPUT		:CHECK FOR 'P'
697	002022	001002			BNE AAA		:GO THRU IF NOT PREVIOUS
698	002024	000137	002344		JMP QABR		:BRANCH IF PREVIOUS
699	002030	012700	000044		AAA: MOV #44, R0		:SET UP COUNTER
700	002034	012701	001200		MOV #LENGTH,R1		:POINT TO CHAR LEN TABLE
701	002040	005021			1\$: CLR (R1)+		:PUT CHAR MASK FOR 8 IN CHAR TABLE
702							:AND CLR PARITY TABLE
703	002042	005300			DEC R0		:COUNT DOWN
704	002044	001375			BNE 1\$:BRANCH IF NOT DONE
705	002046	105737	015772		TSTB INPUT		:CHECK FOR CR
706	002052	001002			BNE SSS		:GO THRU IF NOT DEFAULT
707	002054	000137	002344		JMP QABR		:BRANCH IF DEFAULT
708	002060	122737	000131	015772	SSS: CMPB #131, INPUT		:CHECK FOR 'Y'
709	002066	001002			BNE DDD		:GO THRU IF NOT DEFAULT
710	002070	000137	002344		JMP QABR		:BRANCH IF DEFAULT
711	002074	122737	000116	015772	DDD: CMPB #116, INPUT		:CHECK FOR 'N'
712	002102	001340			BNE CONFIG		:BRANCH IF ILLEGAL ENTRY
713	002104	005000			CLR R0		:CLR UNIT COUNTER
714	002106	005001			CLR R1		:CLR LINE COUNTER
715	002110	012702	001200		MOV #LENGTH,R2		:SET UP POINTER TO CHAR MASK TABLE
716	002114	012703	001300		MOV #PARITY,R3		:SET UP POINTER TO PARITY TABLE
717	002120	012704	000001		MOV #1,R4		:SET UP MARKER
718	002124	000004	016665		TYPLIN: TYPE, MSGLIN		:TYPE 'LINES'
719	002130	010105			MOV R1,TTY		:TYPE R1 IN OCTAL
720	002132	004737	016052		JSR PC,PRINTS		:AND SUPPRESS LEADING ZERO'S
721	002136	000004	016676		TYPE, MSGDAS		:TYPE A DASH (-)

778	002430	022737	000067	015772		CMP	#67, INPUT	:	CHECK IF A VALID ENTRY
779	002436	100411				BMI	NOGOOD	:	REPORT ERROR
780	002440	013724	015772			MOV	INPUT, (R4)+	:	STORE VALUE OF Prio
781	002444	005237	017210			INC	DEVNUM	:	TO NEXT DEVICE # (IN ASCII)
782	002450	023737	017212	017210		CMP	UNITS1, DEVNUM	:	ARE YOU TESTING LAST DEVICE???
783	002456	001350				BNE	DEVNEW	:	BACK IF MORE DEVICES
784	002460	000403				BR	AROUND	:	THRU IF ALL DEVICES FINISHED
785	002462	000004	017117			NOGOOD:	TYPE, MSBAD	:	TYPE ERROR IN INPUT VALUE
786	002466	000744				BR	DEVNEW	:	BACK TO ASK AGAIN
787	002470	012604				AROUND:	MOV (SP)+, R4	:	RESTORE R4

```

788
789 002472 000005          RESTAR: RESET          ;ISSUE RE FT
790 002474 012706 001200      MOV      #STACK, SP      ;SET UP STACK POINTER
791 002500 005737 001400      TST     FTIME
792 002504 001410          BEQ     CLRVEC
793 002506 022737 000176 001374  CMP     #SWREG, SWR
794 002514 001004          BNE     CLRVEC
795 002516 004737 017604      JSR     PC,      CNTLU
796 002522 005037 001400      CLR     FTIME
797 002526 012700 000300      CLRVEC: MOV    #300,R0    ;BEGINNING OF FLOATING VECTORS
798 002532 005720          1$:  TST     (R0)+
799 002534 010060 177776      MOV     R0,-2(R0)      ;'+2'
800 002540 012720 000004      MOV     #!OT, (R0)+   ;'!OT'
801 002544 022700 001000      CMP     #1000,R0
802 002550 001370          BNE     1$
803 002552 062737 000010 001320  ADD     #10,  CSR      ;UPDATE DEVICE ADDRESS TO NEXT UNIT
804 002560 062737 000010 001330  ADD     #10,  RCVVEC   ;UPDATE DEVICE VECTOR ADDRESS
805 002566 062737 000004 001364  ADD     #4,   DJLEN    ;MOVE CHAR TABLE POINTER
806 002574 006337 001366          ASL     DJPAR        ;UPDATE PARITY FLAG MARKER
807 002600 023737 001362 001344  CMP     DJUUT,UNITS
808 002606 002416          BLT     2$
809 002610 005037 001362          CLR     DJUUT
810 002614 013737 001340 001320  MOV     DEVADR,CSR
811 002622 013737 001342 001330  MOV     VECADR,RCVVEC
812 002630 012737 001200 001364  MOV     #LENGTH,DJLEN ;INIT CHAR TABLE POINTER
813 002636 012737 000001 001366  MOV     #1,DJPAR      ;INIT PARITY FLAG MARKER
814 002644 005237 001362          2$:  INC     DJUUT
815 002650 013701 001320          MOV     CSR,  R1      ;SET UP ALL THE REGISTER ADDRESSES
816 002654 062701 000002          ADD     #2,  R1      ;ADD 2
817 002660 010137 001322          MOV     R1,  RBUF    ;SET UP RECEIVER BUFFER
818 002664 062701 000002          ADD     #2,  R1      ;ADD 2
819 002670 010137 001324          MOV     R1,  TCP     ;SET UP TRANSMITTER CONTROL REG
820                                     ; AND BREAK STATUS REG
821 002674 062701 000002          ADD     #2,  R1      ;ADD 2
822 002700 010137 001326          MOV     R1,  TBUF    ;SET UP TRANSMITTER BUFFER
823 002704 013701 001330          MOV     RCVVEC, R1   ;POINTER FOR VECTOR SETUP
824 002710 005721          TST     (R1)+        ;INC R1
825 002712 010137 001332          MOV     R1,  RCVLVL  ;SET INT LVL
826 002716 005721          TST     (R1)+        ;INC R1
827 002720 010137 001334          MOV     R1,  XMTVEC  ;TRANSMITTER VECTOR
828 002724 005721          TST     (R1)+        ;INC R1
829 002726 010137 001336          MOV     R1,  XMTLVL  ;XMT INT LVL ADR
830 002732 005037 001310          CLR     ICNT
831 002736 005037 015250          CLR     LAD
832 002742 104400          SCOPE
833
834

```



```

835
836
837
838
839
840
841
842 002744 012737 003002 000004 TST1: MOV #1$, @#4 ;SET UP TIME-OUT TRAP VECTOR
843 002752 012737 000340 000006 MOV #LEVEL7,@#6
844 002760 005777 176334 TST @CSR ;REFERENCE CSR (READ)
845 002764 005077 176330 CLR @CSR ;CLEAR CSR (WRITE)
846 002770 005577 176324 ADC @CSR ;CHECK CSR (READ AND WRITE)
847 002774 001405 BEQ 2$ ;BRANCH IF OK
848 002776 104000 HLT ;CSR NOT CLEARED
849
850 003000 000403 BR 2$ ;SKIP ISR
851
852 003002 012601 1$: MOV (SP)+,R1 ;SAVE RTI ADR FOR TYPING
853 003004 104001 HLT+1 ;CAN'T REFERENCE CSR!
854
855 003006 005726 TST (SP)+ ;FINISH CLEARING STACK
856 003010 012737 000006 000004 2$: MOV #6,@#4
857 003016 005037 000006 CLR @#6
858 003022 104400 SCOPE
859
860
861
862
863
864
865 003024 012777 000002 176266 TST2: MOV #BIT1,@CSR ;SET BIT1
866 003032 032777 000002 176260 BIT #BIT1,@CSR ;CHECK THAT BIT1 IS SET
867 003040 001001 BNE .+4 ;BRANCH IF OK
868 003042 104000 HLT ;CSR BIT1 FAILED TO SET
869
870 003044 032777 177775 176246 BIT #177775,@CSR ;CHECK THAT NO OTHER BIT SET
871 003052 001401 BEQ .+4 ;BRANCH IF OK
872 003054 104000 HLT ;EXTRA BIT SET IN CSR
873
874 003056 005077 176236 CLR @CSR ;CLEAR BIT1
875 003062 032777 000002 176230 BIT #BIT1,@CSR ;CHECK THAT BIT1 IS CLEARED
876 003070 001401 BEQ .+4 ;BRANCH IF OK
877 003072 104000 HLT ;CSR BIT1 FAILED TO CLEAR
878
879 003074 104400 SCOPE
880
881
882
883
884
885
886 003076 012777 000004 176214 TST3: MOV #BIT2,@CSR ;SET BIT2
887 003104 032777 000004 176206 BIT #BIT2,@CSR ;CHECK THAT BIT2 IS SET
888 003112 001001 BNE .+4 ;BRANCH IF OK
889 003114 104000 HLT ;CSR BIT2 FAILED TO SET
890

```

```

891 003116 032777 177773 176174 BIT #177773,@CSR ;CHECK THAT NO OTHER BIT SET
892 003124 001401 BEQ .+4 ;BRANCH IF OK
893 003126 104000 HLT ;EXTRA BIT SET IN CSR
894
895 003130 005077 176164 CLR @CSR ;CLEAR BIT2
896 003134 032777 000004 176156 BIT #BIT2,@CSR ;CHECK THAT BIT2 IS CLEARED
897 003142 001401 BEQ .+4 ;BRANCH IF OK
898 003144 104000 HLT ;CSR BIT2 FAILED TO CLEAR
899
900 003146 104400 SCOPE
  
```

```

:*****
:TEST 4: TEST THAT CSR BIT6 CAN BE SET AND CLEARED
:PROBABLE FAULTY LOGIC: M7285 (D2-2) E4,E18, (D2-4) E47,E64
:*****
  
```

```

907 003150 012777 000100 176142 TST4: MOV #BIT6,@CSR ;SET BIT6
908 003156 032777 000100 176134 BIT #BIT6,@CSR ;CHECK THAT BIT6 IS SET
909 003164 001001 BNE .+4 ;BRANCH IF OK
910 003166 104000 HLT ;CSR BIT6 FAILED TO SET
911
912 003170 032777 177677 176122 BIT #177677,@CSR ;CHECK THAT NO OTHER BIT SET
913 003176 001401 BEQ .+4 ;BRANCH IF OK
914 003200 104000 HLT ;EXTRA BIT SET IN CSR
915
916 003202 005077 176112 CLR @CSR ;CLEAR BIT6
917 003206 032777 000100 176104 BIT #BIT6,@CSR ;CHECK THAT BIT6 IS CLEARED
918 003214 001401 BEQ .+4 ;BRANCH IF OK
919 003216 104000 HLT ;CSR BIT6 FAILED TO CLEAR
920
921 003220 104400 SCOPE
  
```

```

:*****
:TEST 5: TEST THAT CSR BIT8 CAN BE SET AND CLEARED
:PROBABLE FAULTY LOGIC: M7285 (D2-2) E6,E18, (D2-4) E47,E31
:*****
  
```

```

928 003222 012777 000400 176070 TST5: MOV #BIT8,@CSR ;SET BIT8
929 003230 032777 000400 176062 BIT #BIT8,@CSR ;CHECK THAT BIT8 IS SET
930 003236 001001 BNE .+4 ;BRANCH IF OK
931 003240 104000 HLT ;CSR BIT8 FAILED TO SET
932
933 003242 032777 177377 176050 BIT #177377,@CSR ;CHECK THAT NO OTHER BIT SET
934 003250 001401 BEQ .+4 ;BRANCH IF OK
935 003252 104000 HLT ;EXTRA BIT SET IN CSR
936
937 003254 005077 176040 CLR @CSR ;CLEAR BIT8
938 003260 032777 000400 176032 BIT #BIT8,@CSR ;CHECK THAT BIT8 IS CLEARED
939 003266 001401 BEQ .+4 ;BRANCH IF OK
940 003270 104000 HLT ;CSR BIT8 FAILED TO CLEAR
941
942 003272 104400 SCOPE
  
```

```

:*****
:TEST 6: TEST THAT CSR BIT10 CAN BE SET AND CLEARED
:PROBABLE FAULTY LOGIC: M7285 (D2-2) E30,E24, (D2-4) E47,E31
:*****
  
```

943
944
945
946

```

947
948
949 003274 012777 002000 176016 TST6: MOV #BIT10,@CSR ;SET BIT10
950 003302 032777 002000 176010 BIT #BIT10,@CSR ;CHECK THAT BIT10 IS SET
951 003310 001001 BNE .+4 ;BRANCH IF OK
952 003312 104000 HLT ;CSR BIT10 FAILED TO SET
953
954 003314 032777 175777 175776 BIT #175777,@CSR ;CHECK THAT NO OTHER BIT SET
955 003322 001401 BEQ .+4 ;BRANCH IF OK
956 003324 104000 HLT ;EXTRA BIT SET IN CSR
957
958 003326 005077 175766 CLR @CSR ;CLEAR BIT10
959 003332 032777 002000 175760 BIT #BIT10,@CSR ;CHECK THAT BIT10 IS CLEARED
960 003340 001401 BEQ .+4 ;BRANCH IF OK
961 003342 104000 HLT ;CSR BIT10 FAILED TO CLEAR
962
963 003344 104400 SCOPE
964
965
966 ;*****
967 ;TEST 7: TEST THAT CSR BIT12 CAN BE SET AND CLEARED
968 ;PROBABLE FAULTY LOGIC: M7285 (D2-2) E5,E1, (D2-4) E47,E31
969 ;*****
970 003346 012777 010000 175744 TST7: MOV #BIT12,@CSR ;SET BIT12
971 003354 032777 010000 175736 BIT #BIT12,@CSR ;CHECK THAT BIT12 IS SET
972 003362 001001 BNE .+4 ;BRANCH IF OK
973 003364 104000 HLT ;CSR BIT12 FAILED TO SET
974
975 003366 032777 167777 175724 BIT #167777,@CSR ;CHECK THAT NO OTHER BIT SET
976 003374 001401 BEQ .+4 ;BRANCH IF OK
977 003376 104000 HLT ;EXTRA BIT SET IN CSR
978
979 003400 005077 175714 CLR @CSR ;CLEAR BIT12
980 003404 032777 010000 175706 BIT #BIT12,@CSR ;CHECK THAT BIT12 IS CLEARED
981 003412 001401 BEQ .+4 ;BRANCH IF OK
982 003414 104000 HLT ;CSR BIT12 FAILED TO CLEAR
983
984 003416 104400 SCOPE
985
986
987 ;*****
988 ;TEST 10: TEST THAT CSR BIT14 CAN BE SET AND CLEARED
989 ;PROBABLE FAULTY LOGIC: M7285 (D2-2) E3,E1, (D2-4) E47,E31
990 ;*****
991 003420 012777 040000 175672 TST10: MOV #BIT14,@CSR ;SET BIT14
992 003426 032777 040000 175664 BIT #BIT14,@CSR ;CHECK THAT BIT14 IS SET
993 003434 001001 BNE .+4 ;BRANCH IF OK
994 003436 104000 HLT ;CSR BIT14 FAILED TO SET
995
996 003440 032777 137777 175652 BIT #137777,@CSR ;CHECK THAT NO OTHER BIT SET
997 003446 001401 BEQ .+4 ;BRANCH IF OK
998 003450 104000 HLT ;EXTRA BIT SET IN CSR
999
1000 003452 005077 175642 CLR @CSR ;CLEAR BIT14
1001 003456 032777 040000 175634 BIT #BIT14,@CSR ;CHECK THAT BIT14 IS CLEARED
1002 003464 001401 BEQ .+4 ;BRANCH IF OK

```

```

1003 003466 104000          HLT          ;CSR BIT14 FAILED TO CLEAR
1004
1005 003470 104400          SCOPE
1006
1007
1008 ;*****
1009 ;TEST 11:      TEST THAT RECEIVER ENABLE (BIT0 OF THE CSR) CAN BE SET
1010 ;              AND CLEARED, AND THAT CLEAR MOS (BIT3) IS WRITE ONLY.
1011 ;PROBABLE FAULTY LOGIC: M7285 (D2-2) E26,E36,E7,E24, (D2-8) E17,E14,E15
1012 ;*****
1013 003472 012777 000004 175620 TST11: MOV    #BIT2, @CSR    ;SET MAINTENANCE MODE (BIT2)
1014 003500 005005          CLR    R5                ;SET UP COUNTER
1015 003502 052777 000010 175610 1$:  BIS    #BIT3, @CSR    ;SET CLEAR MOS (BIT3)
1016 003510 017701 175604          MOV    @CSR, R1         ;SAVE CSR
1017 003514 032701 000010          BIT    #BIT3, R1       ;CHECK CLEAR MOS (BIT3)
1018 003520 001401          BEQ    .+4              ;BRANCH IF OK
1019 003522 104001          HLT+1                 ;CLEAR MOS (BIT3) SET (WRITE-ONLY)
1020 ;R1 = CONTENTS OF CSR
1021 003524 032701 000020          BIT    #BIT4, R1       ;CHECK CLEAR MOS FLAG
1022 003530 001403          BEQ    2$              ;BRANCH IF CLEARED
1023 003532 105305          DECB  R5                ;WAIT FOR MOS TO CLEAR
1024 003534 001365          BNE    1$              ;BRANCH IF MORE TIME
1025 003536 104001          HLT+1                 ;CLEAR MOS FLAG (BIT4) FAILED TO CLEAR
1026 ;R1 = CONTENTS OF CSR
1027 003540 022701 000004 2$:  CMP    #BIT2, R1       ;CHECK THAT ONLY MAINTENANCE BIT SET
1028 003544 001401          BEQ    .+4              ;BRANCH IF OK
1029 003546 104001          HLT+1                 ;CLEAR MOS CLEARED MAINTENANCE
1030 ;OR SET OTHER CSR BITS
1031 ;R1 = CONTENTS OF CSR
1032 003550 052777 000001 175542  BIS    #BIT0, @CSR    ;SET RECEIVER ENABLE
1033 003556 017701 175536          MOV    @CSR, R1         ;SAVE CSR
1034 003562 032777 000001 175530  BIT    #BIT0, @CSR    ;CHECK THAT RECEIVER ENABLE SET
1035 003570 001001          BNE    .+4              ;BRANCH IF OK
1036 003572 104001          HLT+1                 ;RECEIVER ENABLE FAILED TO SET
1037 ;R1 = CONTENTS OF CSR
1038 003574 022777 000005 175516  CMP    #5, @CSR        ;CHECK REST OF CSR
1039 003602 001401          BEQ    .+4              ;BRANCH IF OK
1040 003604 104001          HLT+1                 ;CSR ERROR
1041 ;R1 = CONTENTS OF CSR
1042 ;NOTE: IF THE TTY MODULE IS BEING USED AND DONE (BIT7) IS SET,
1043 ;THE ERROR COULD BE DUE TO MAINTENANCE OR CLEAR MOS NOT WORKING.
1044 003606 042777 000001 175504  BIC    #BIT0, @CSR    ;CLEAR RECEIVER ENABLE
1045 003614 017701 175500          MOV    @CSR, R1         ;SAVE CSR
1046 003620 022777 000004 175472  CMP    #BIT2, @CSR    ;CHECK CSR
1047 003626 001401          BEQ    .+4              ;BRANCH IF OK
1048 003630 104001          HLT+1                 ;RECEIVER ENABLE DIDN'T CLEAR
1049 ;OR OTHER CSR BIT SET
1050 ;R1 = CONTENTS OF CSR
1051
1052 003632 104400          SCOPE
1053
1054 ;*****
1055 ;TEST 12:      TEST THAT CSR RESPONDS PROPERLY TO BYTE COMMANDS
1056 ;PROBABLE FAULTY LOGIC: M7285 (D2-4) E47
1057 ;*****
1058

```

```

1059 003634 012777 052506 175456 TST12: MOV #052506,@CSR ;SET TEST NUMBER IN CSR
1060 003642 105077 175452 CLR @CSR ;CLR EVEN BYTE
1061 003646 017701 175446 MOV @CSR,R1 ;SAVE CSR
1062 003652 022701 052400 CMP #052400,R1 ;CHECK CSR
1063 003656 001401 BEQ .+4 ;BRANCH IF OK
1064 003660 104001 HLT+1 ;EVEN BYTE CLR FAILED ON CSR
1065 ;R1 = CONTENTS OF CSR
1066 003662 012777 052506 175430 MOV #052506,@CSR ;SET TEST NUMBER IN CSR
1067 003670 005237 001320 INC CSR ;INC TO ODD BYTE
1068 003674 105077 175420 CLR @CSR ;CLR ODD BYTE
1069 003700 005337 001320 DEC CSR ;RESTORE TO EVEN
1070 003704 017701 175410 MOV @CSR,R1 ;SAVE CSR
1071 003710 022701 000106 CMP #000106,R1 ;CHECK CSR
1072 003714 001401 BEQ .+4 ;BRANCH IF OK
1073 003716 104001 HLT+1 ;ODD BYTE CLR FAILED ON CSR
1074 ;R1 = CONTENTS OF CSR
1075

```

SCOPE

```

:*****
:TEST 13: TEST THAT THE BIS AND BIC INSTRUCTIONS SET AND CLEAR R/W
:BITS OF CSR
:PROBABLE FAULTY LOGIC: M7285 (D2-4) E47
:*****

```

```

1084 003722 005077 175372 TST13: CLR @CSR ;CLEAR THE CSR
1085 003726 017701 175366 MOV @CSR,R1 ;CHECK AND SAVE CSR
1086 003732 001401 BEQ .+4 ;BRANCH IF CLEARED OK
1087 003734 104001 HLT+1 ;RESET FAILED TO CLR CSR
1088
1089 003736 052777 052506 175354 BIS #052506,@CSR ;SET ALL R/W BITS OF CSR
1090 003744 022777 052506 175346 CMP #052506,@CSR ;CHECK THAT THEY GOT SET
1091 003752 001401 BEQ .+4 ;BRANCH IF OK
1092 003754 104000 HLT ;REG FAILED CMP
1093
1094 003756 042777 052506 175334 BIC #052506,@CSR ;CLEAR CSR
1095 003764 017701 175330 MOV @CSR,R1 ;CHECK AND SAVE CSR
1096 003770 001401 BEQ .+4 ;BRANCH IF CLEARED OK
1097 003772 104001 HLT+1 ;CLR FAILED TO CLR CSR
1098

```

SCOPE

```

:*****
:TEST 14: TEST BITS OF TCR FOR READ/WRITE CAPABILITY
:PROBABLE FAULTY LOGIC: M7285 (D2-2) ALL, (D2-3) E8,E20,E21,E43,E47
:*****

```

```

1106 003776 012777 177777 175320 TST14: MOV #177777,@TCR ;SET ALL BITS OF TCR
1107 004004 017701 175314 MOV @TCR,R1 ;CHECK AND SAVE TCR
1108 004010 022701 177777 CMP #177777,R1 ;CHECK THAT ALL THE BITS ARE SET
1109 004014 001401 BEQ .+4 ;BRANCH IF OK
1110 004016 104001 HLT+1 ;BIT(S) OF TCR FAILED TO SET
1111
1112 004020 005077 175300 CLR @TCR ;CLEAR TCR
1113 004024 017701 175274 MOV @TCR,R1 ;CHECK THAT IT CLEARED AND SAVE
1114 004030 001401 BEQ .+4 ;BRANCH IF CLR

```

1115 004032 104001 HLT+1 ;BIT(S) OF TCR FAILED TO CLEAR

1116
1117 004034 104400 SCOPE

1118
1119
1120 :*****
1121 : TEST 15: TEST THAT TCR RESPONDS PROPERLY TO BYTE COMMANDS
1122 : PROBABLE FAULTY LOGIC: M7285 (D2-3) E41
1123 :*****

1124 004036 012777 177777 175260 TST15: MOV #177777,@TCR ;SET TEST NUMBER IN TCR
1125 004044 105077 175254 CLR @TCR ;CLR EVEN BYTE
1126 004050 017701 175250 MOV @TCR,R1 ;SAVE TCR
1127 004054 022701 177400 CMP #177400,R1 ;CHECK TCR
1128 004060 001401 BEQ .+4 ;BRANCH IF OK
1129 004062 104001 HLT+1 ;EVEN BYTE CLR FAILED ON TCR

1130 ;R1 = CONTENTS OF TCR
1131 004064 012777 177777 175232 MOV #177777,@TCR ;SET TEST NUMBER IN TCR
1132 004072 005237 001324 INC TCR ;INC TO ODD BYTE
1133 004076 105077 175222 CLR @TCR ;CLR ODD BYTE
1134 004102 005337 001324 DEC TCR ;RESTORE TO EVEN
1135 004106 017701 175212 MOV @TCR,R1 ;SAVE TCR
1136 004112 022701 000377 CMP #000377,R1 ;CHECK TCR
1137 004116 001401 BEQ .+4 ;BRANCH IF OK
1138 004120 104001 HLT+1 ;ODD BYTE CLR FAILED ON TCR
1139 ;R1 = CONTENTS OF TCR

1140
1141 004122 104400 SCOPE

1142
1143 :*****
1144 : TEST 16: TEST THAT THE BIS AND BIC INSTRUCTIONS SET AND CLEAR R/W
1145 : BITS OF TCR
1146 : PROBABLE FAULTY LOGIC: M7285 (D2-3) E41
1147 :*****

1148
1149 004124 005077 175174 TST16: CLR @TCR ;CLEAR THE TCR
1150 004130 017701 175170 MOV @TCR,R1 ;CHECK AND SAVE TCR
1151 004134 001401 BEQ .+4 ;BRANCH IF CLEARED OK
1152 004136 104001 HLT+1 ;RESET FAILED TO CLR TCR

1153
1154 004140 052777 177777 175156 BIS #177777,@TCR ;SET ALL R/W BITS OF TCR
1155 004146 022777 177777 175150 CMP #177777,@TCR ;CHECK THAT THEY GOT SET
1156 004154 001401 BEQ .+4 ;BRANCH IF OK
1157 004156 104000 HLT ;REG FAILED CMP

1158
1159 004160 042777 177777 175136 BIC #177777,@TCR ;CLEAR TCR
1160 004166 017701 175132 MOV @TCR,R1 ;CHECK AND SAVE TCR
1161 004172 001401 BEQ .+4 ;BRANCH IF CLEARED OK
1162 004174 104001 HLT+1 ;CLR FAILED TO CLR TCR

1163
1164 004176 104400 SCOPE
1165 004200 012737 000001 001346 MOV #1,TIMER ;INITIALIZE TIMER
1166 004206 012737 004214 015250 MOV #.+6,LAD ;RESET LOOP ADDRESS

1167
1168 :*****
1169 : TEST 17: TEST THAT TRANSMIT READY (BIT15 OF CSR) SETS AND CLEARS
1170 : WHEN TCR0 IS SET AND CLEARED.

```

1171
1172
1173
1174
1175 004214 012777 000400 175076 TST17: MOV #400, @CSR ;SET XMTR SCAN ENABLE
1176 004222 052777 000001 175074 BIS #BIT0, @TCR ;SET XMTR CONTROL BIT, LINE 0
1177 004230 005000 CLR R0 ;SET UP WAIT COUNTER
1178 004232 017701 175062 1$: MOV @CSR, R1 ;CHECK AND SAVE XMTR READY
1179 004236 100404 BMI 2$ ;BRANCH IF SET OK
1180 004240 005200 INC R0 ;WAIT A WHILE
1181 004242 001373 BNE 1$
1182 004244 104001 HLT+1 ;XMTR READY FAILED TO SET
1183
1184 004246 000416 BR 3$ ;SKIP LINE # CHECK ON ERROR
1185
1186 004250 050037 001346 2$: BIS R0, TIMEP ;SET UP TIMER
1187 004254 017701 175046 MOV @TBUF, R1 ;SAVE LINE NUMBER
1188 004260 105001 CLRB R1
1189 004262 000301 SWAB R1
1190 004264 022701 000000 CMP #0, R1 ;CHECK THAT THE XMTR STOPPED ON LINE 0
1191 004270 001401 BEQ .+4 ;BRANCH IF OK
1192 004272 104001 HLT+1 ;WRONG LINE NUMBER APPEARED IN TBUF
1193
1194 004274 017702 175020 MOV @CSR, R2 ;CHECK AND SAVE XMTR READY
1195 004300 100401 BMI .+4 ;BRANCH IF OK
1196 004302 104002 HLT+2 ;READING THE TBUF CLEARED XMTR READY
1197
1198 004304 042777 000001 175012 3$: BIC #BIT0, @TCR ;CLR XMTR CONTROL BIT, LINE 0
1199 004312 017701 175002 MOV @CSR, R1 ;CHECK AND SAVE XMTR READY
1200 004316 100001 BPL .+4 ;BRANCH IF OK
1201 004320 104001 HLT+1 ;XMTR READY FAILED TO CLEAR WHEN
1202 ; XMTR CONTROL FOR LINE 0 WAS CLEARED
1203
1204 004322 104400 SCOPE
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226

```

ALSO CHECK THAT THE RIGHT LINE NO. (0) APPEARS IN TBUF.
 ;PROBABLE FAULTY LOGIC: M7285 (D2-5) ALL, (D2-6) E23, E32, E33, E49, (D2-2) E3, E1
 ;*****

 ;TEST 20: TEST THAT TRANSMIT READY (BIT15 OF CSR) SETS AND CLEARS
 ; WHEN EACH TCR BIT IS SET AND CLEARED.
 ; ALSO CHECK THAT THE RIGHT LINE NO. APPEARS IN TBUF.
 ;PROBABLE FAULTY LOGIC: M7285 (D2-5) ALL, (D2-6) E23, E32, E33, E49, (D2-2) E3, E1
 ;*****

```

1214 004324 012777 000400 174766 TST20: MOV #400, @CSR ;SET XMTR SCAN ENABLE
1215 004332 005001 CLR R1
1216 004334 012704 000001 MOV #1, R4 ;SET UP MARKER
1217 004340 050477 174760 4$: BIS R4, @TCR ;SET XMTR CONTROL BIT, EACH LINE
1218 004344 005000 CLR R0 ;SET UP WAIT COUNTER
1219 004346 017702 174746 1$: MOV @CSR, R2 ;CHECK AND SAVE XMTR READY
1220 004352 100404 BMI 2$ ;BRANCH IF SET OK
1221 004354 005300 DEC R0 ;WAIT A WHILE
1222 004356 001373 BNE 1$
1223 004360 104002 HLT+2 ;XMTR READY FAILED TO SET
1224 ;R1=LINE #
1225 ;R2=CSR
1226

```

```

1227 004362 000414 BR 3$ ;SKIP LINE # CHECK ON ERROR
1228
1229 004364 050037 001346 2$: BIS R0, TIMER ;SET UP TIMER
1230 004370 017702 174732 MOV @TBUF, R2 ;SAVE LINE NUMBER
1231 004374 000302 SWAB R2
1232 004376 020201 CMP R2, R1 ;CHECK THAT THE XMTR STOPPED ON RIGHT LINE
1233 004400 001401 BEQ .+4 ;BRANCH IF OK
1234 004402 104002 HLT+2 ;WRONG LINE NUMBER APPEARED IN TBUF
1235 ;R1=LINE # (SHOULD BE)
1236 ;R2=LINE # (TBUF)
1237
1238
1239 004404 017703 174710 MOV @CSR, R3 ;CHECK AND SAVE XMTR READY
1240 004410 100401 BMI .+4 ;BRANCH IF OK
1241 004412 104003 HLT+3 ;READING THE TBUF CLEARED XMTR READY
1242
1243 004414 040477 174704 3$: BIC R4, @TCR ;CLR XMTR CONTROL BIT, EACH LINE
1244 004420 017702 174674 MOV @CSR, R2 ;CHECK AND SAVE XMTR READY
1245 004424 100001 BPL .+4 ;BRANCH IF OK
1246 004426 104002 HLT+2 ;XMTR READY FAILED TO CLEAR WHEN
1247 ; XMTR CONTROL FOR LINE .LINE WAS CLEARED
1248 ;R1 = LINE #
1249 ;R2 = CONTENTS OF CSR
1250
1251 004430 005201 INC R1 ;INC LINE COUNTER TO NEXT LINE
1252 004432 006304 ASL R4 ;SHIFT MARKER TO NEXT LINE
1253 004434 103341 BCC 4$ ;BRANCH IF NOT LAST LINE
1254
1255 004436 104400 SCOPE
1256
1257
1258
1259
1260
1261
1262
1263
1264

```

```

:*****
:TEST 21: TEST THAT MASTER TRANSMIT SCAN ENABLE (CSR BIT 8) ON A 0
:          DISABLES TRANSMITTER READY (CSR BIT 15)
:          WHEN TCR BIT, LINE 0 IS SET.
:PROBABLE FAULTY LOGIC: M7285 (D2-6) E49,E23,E32,E33
:*****

```

```

1265 004440 005077 174654 TST21: CLR @CSR ;CLEAR CONTROL STATUS
1266 004444 052777 000001 174652 BIS #BIT0,@TCR ;SET XMTR CONTROL BIT, LINE 0
1267 004452 013700 001346 MOV TIMER,R0 ;GET TIMER FROM PREVIOUS TEST
1268 004456 017701 174636 1$: MOV @CSR,R1 ;CHECK AND SAVE XMTR READY
1269 004462 100002 BPL 2$ ;BRANCH IF NOT SET
1270 004464 104001 HLT+1 ;XMTR READY SET WITHOUT
1271 004466 000402 BR 3$ ;MASTER TRAN SCAN ENABLE
1272 004470 005300 2$: DEC R0 ;WAIT A WHILE
1273 004472 001371 BNE 1$ ;AND CHECK AGAIN
1274
1275 004474 052777 000400 174616 3$: BIS #400,@CSR ;SET MASTER TRAN SCAN ENABLE
1276 004502 005000 CLR R0
1277 004504 017701 174610 4$: MOV @CSR, R1 ;CHECK AND SAVE XMTR READY
1278 004510 100403 BMI 5$
1279 004512 005200 INC R0
1280 004514 001373 BNE 4$
1281 004516 104001 HLT+1 ;TRAN READY NEVER CAME UP
1282 ;TCR BIT WAS SET FIRST

```



```

1283
1284 004520 042777 000001 174576 5S: BIC #BIT0, @TCR ; THEN MASTER TRAN SCAN ENABLE
1285 004526 017701 174566 MOV @CSR, R1 ; CLEAR IT OUT
1286 004532 100001 BPL .+4 ; CHECK AND SAVE TRAN READY
1287 004534 104001 HLT+1 ; BRANCH IF OK
1288 ; TRAN READY DIDN'T CLEAR.
1289 004536 104400 SCOPE
1290
1291
1292
1293 ;*****
1294 ;TEST 22: TEST TRANSMIT INTERRUPT LEVEL
1295 ;
1296 ; THIS TESTS ALL INTERRUPT LEVELS. THIS TEST WILL
1297 ; TEST ALL DEVICES WHICH ARE SPECIFIED DURING QUESTIONING.
1298 ;
1299 ; ERROR PRINT OUT IS AS FOLLOWS:::
1300 ; ADDR DJADR RO R1
1301 ;
1302 ; ADDR= ADDRESS OF ERROR HLT
1303 ; DJADR= CSR ADDRESS OF DJ11 UNDER TEST
1304 ; R1= BR LEVEL FOR DJ11 TO ALLOW INTR
1305 ; R2= BR LEVEL WHERE ERROR OCCURED
1306 ;
1307 ;PROBABLE FAULTY LOGIC: M7821 WIRING, PROPER PRIORITY CHIP
1308 ;*****
1309 004540 012777 004656 174566 TST22: MOV #INTR1,@XMTVEC ; SET UP XMTR INTERRUPT VECTOR
1310 004546 012777 000340 174562 MOV #340,@XMTLVL ; AT LEVEL 7
1311 004554 012737 000010 017214 MOV #10,PRIOLO ; LOAD BR7+1 INTO BR TEST LEVEL
1312 004562 012737 000005 017226 AGAIN1: MOV #5,CT ; LOADS COUNTER 5 ROL -BR PRIOT
1313 004570 005337 017214 DEC PRIOLO ; START TEST BR7, THEN BR6,BR5...ETC.
1314 004574 013737 017214 017216 MOV PRIOLO,NOW ; LOAD PRE-ROL-ED DECIMAL PRIO
1315 004602 004737 014464 JSR PC, BRSET ; GO TO SUBROUTINE TO :::::
1316 ;
1317 ; LOAD BR INTR LEVEL OF DJ11
1318 ; (LEVEL)= DECIMAL #(0 TO 7)
1319 ; (MASK)= OCTAL #(0 TO 340)
1320 ; (0,40,100,140,200,240,300,340)
1321 ; (NOW)= OCTAL #(0 TO 340) AS
1322 004606 042737 000340 177776 BIC #340,@PS ; MASK, BUT CURRENTLY TESTING
1323 004614 053737 017216 177776 BIS NOW,@PS ; CLEAR PS LEVEL
1324 004622 012777 040400 174470 MOV #040400,@CSR ; SET PS TO CURRENT TEST LEVEL
1325 004630 012777 000001 174466 MOV #BIT0, @TCR ; SET TRAN MASTER INT. ENABLE
1326 004636 017701 174456 WAIT1: MOV @CSR, R1 ; SET TRAN CONTROL BIT. LINE G
1327 004642 100375 BPL WAIT1 ; WAIT
1328
1329 004644
1330
1331 ;
1332 ;IF YOU ARE HERE NO INTR OCCURED
1333
1332 004644 023737 017216 017220 CMP NOW,MASK ; IS THIS AN ERROR?????
1333 004652 002407 BLT ERROR1 ; BRANCH IF YES
1334 004654 000417 BR THRU1 ; IF NO,GO ON TO NEXT BR
1335
1336 004656
1337
1338 INTR1:;
;IF YOU ARE HERE AN INTR OCCURED

```



```

1395          ;WAIT FOR MOS TO CLEAR
1396          ;
1397          ;
1398          ;
1399 004774 052777 000001 171322          ;SET XMTR CONTROL BIT, LINE 0
1400 005002 017701 174312          ;WAIT FOR XMTR READY
1401 005006 100375          ;
1402 005010 012777 000377 174310          ;SEND A RUBOUT
1403 005016 005000          ;CLEAR COUNTER
1404 005020 105305          ;SHORT WAIT LOOP
1405 005022 001376          ;
1406 005024 017701 174272          ;WAIT FOR CHAR. PRES.
1407 005030 100405          ;BRANCH WHEN FOUND
1408 005032 005200          ;TIME COUNTER
1409 005034 001371          ;BRANCH IF NOT TIME-OUT
1410 005036 017702 174256          ;SAVE CSR
1411 005042 104002          ;CHARACTER READY DIDN'T SET
1412          ;R1 = CONTENTS OF RBUF
1413          ;R2 = CONTENTS OF CSR
1414 005044 050037 001346          ;SAVE THE TIMER
1415 005050 032701 070000          ;CHECK FOR ERROR BITS
1416 005054 001401          ;BRANCH IF NONE
1417 005056 104001          ;ERROR IN RECEIVED CHAR
1418          ;R1 = CONTENTS OF RBUF
1419          ;BIT14=UART OVERRUN
1420          ;BIT13=FRAMING ERROR
1421          ;BIT12=PARITY ERROR
1422 005060 010102          ;DUPLICATE DATA WORD
1423 005062 042702 170377          ;MASK LINE#
1424 005066 000302          ;LINE # IN LOW BYTE
1425 005070 122702 000000          ;CHECK LINE #
1426 005074 001401          ;BRANCH IF OK
1427 005076 104001          ;WRONG LINE # RECEIVED
1428          ;R1 = CONTENTS OF RBUF
1429          ;BITS8-11 = LINE #
1430 005100 117702 174260          ;GET MASK OF CHARACTER
1431 005104 130201          ;CHECK CHAR LENGTH.
1432 005106 001401          ;BRANCH IF OK
1433 005110 104002          ;WRONG CHARACTER LENGTH
1434          ;R1=DATA FROM FI/FO
1435          ;R2=MASK (BITS SET NOT EXPECTED)
1436 005112 105102          ;REVERSE THE MASK
1437 005114 120102          ;CHECK THE ACTURAL DATA
1438 005116 001401          ;BRANCH IF OK
1439 005120 104002          ;WRONG CHAR LEN OR DATA ERROR
1440          ;R1=DATA FROM FI/FO (COMPLETE WORD)
1441          ;R2=DATA (LOW BYTE) EXPECTED
1442 005122 017701 174174          ;READ FI/FO
1443 005126 100001          ;BRANCH IF CHAR PRESENT NOT SET
1444 005130 104001          ;CHARACTER PRESENT STAYED SET
1445          ;R1 = CONTENTS OF RBUF
1446 005132 017701 174162          ;SAVE THE CSR
1447 005136 022701 100405          ;CHECK THE CSR
1448 005142 001401          ;BRANCH IF OK
1449 005144 104001          ;DONE DIDN'T CLEAR OR OTHER CSR ERROR
1450          ;R1 = CONTENTS OF CSR
    
```

```

1451 005146 005077 174152          CLR    @TCR          ;CLEAR TCR
1452 005152 005077 174142          CLR    @CSR          ;CLEAR CSR
1453 005156 104400                   SCOPE
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471 005160 004737 014600          TST24: JSR    PC,    @#INITD  SET:
1472
1473
1474
1475
1476
1477
1478 005164 052777 000002 174132          LOP24: BIS    #BIT1, @TCR    ;SET XMTR CONTROL BIT, LINE 1
1479 005172 017701 174122          MOV    @CSR, R1      ;WAIT FOR XMTR READY
1480 005176 100375
1481 005200 012777 000377 174120          BPL   LOP24
1482 005206 005000          MOV    #377, @RBUF   ;SEND A RUBOUT
1483 005210 105305          CLR    R0            ;CLEAR COUNTER
1484 005212 001376          1$:   DECB  R5        ;SHORT WAIT LOOP
1485 005214 017701 174102          BNE   1$
1486 005220 100405          MOV    @RBUF, R1     ;WAIT FOR CHAR. PRES.
1487 005222 005200          BMI   3$            ;BRANCH WHEN FOUND
1488 005224 001371          INC   R0            ;TIME COUNTER
1489 005226 017702 174066          BNE   1$            ;BRANCH IF NOT TIME-OUT
1490 005232 104002          MOV    @CSR, R2     ;SAVE CSR
1491
1492
1493 005234 050037 001346          3$:   BIS    R0, TIMER  ;CHARACTER READY DIDN'T SET
1494 005240 032701 070000          BIT   #70000, R1    ;R1 = CONTENTS OF RBUF
1495 005244 001401          BEQ   .+4           ;R2 = CONTENTS OF CSR
1496 005246 104001          HLT+1             ;SAVE THE TIMER
1497
1498
1499
1500
1501 005250 010102          4$:   MOV    R1, R2    ;CHECK FOR ERROR BITS
1502 005252 042702 170377          BIC   #170377, R2   ;BRANCH IF NONE
1503 005256 000302          SWAB  R2            ;ERROR IN RECEIVED CHAR
1504 005260 122702 000001          CMPB  #1, R2       ;R1 = CONTENTS OF RBUF
1505 005264 001401          BEQ   .+4           ;BIT14=UART OVERRUN
1506 005266 104001          HLT+1             ;BIT13=FRAMING ERROR
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000

```

 TEST 24: TEST THAT LINE 1 CAN TRANSMIT AND
 RECEIVE A CHARACTER. (377)
 1\$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
 3\$: CHECKS THAT NO ERRORS IN FI/FO
 4\$: CHECKS THAT RIGHT LINE # (1) IN FI/FO
 5\$: CHECKS FOR RIGHT CHARACTER LENGTH
 6\$: CHECKS THAT CORRECT DATA WAS RECEIVED
 7\$: CHECKS THAT CHARACTER PRESENT CLEARS
 8\$: CHECKS THAT DONE CLEARS
 PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES

INITIALIZE
 DEVICE 'CSR' REGISTER
 TST24: JSR PC, @#INITD SET:
 ;BIT2 = MAINTENANCE
 ;BIT3 = CLEAR MOS
 ;BIT8 = MASTER XMTR SCAN ENB
 ;WAIT FOR MOS TO CLEAR
 SET:
 ;BIT0 = RECEIVER ENABLE
 LOP24: BIS #BIT1, @TCR ;SET XMTR CONTROL BIT, LINE 1
 MOV @CSR, R1 ;WAIT FOR XMTR READY
 BPL LOP24
 MOV #377, @RBUF ;SEND A RUBOUT
 CLR R0 ;CLEAR COUNTER
 1\$: DECB R5 ;SHORT WAIT LOOP
 BNE 1\$
 MOV @RBUF, R1 ;WAIT FOR CHAR. PRES.
 BMI 3\$;BRANCH WHEN FOUND
 INC R0 ;TIME COUNTER
 BNE 1\$;BRANCH IF NOT TIME-OUT
 MOV @CSR, R2 ;SAVE CSR
 HLT+2 ;CHARACTER READY DIDN'T SET
 ;R1 = CONTENTS OF RBUF
 ;R2 = CONTENTS OF CSR
 3\$: BIS R0, TIMER ;SAVE THE TIMER
 BIT #70000, R1 ;CHECK FOR ERROR BITS
 BEQ .+4 ;BRANCH IF NONE
 HLT+1 ;ERROR IN RECEIVED CHAR
 ;R1 = CONTENTS OF RBUF
 ;BIT14=UART OVERRUN
 ;BIT13=FRAMING ERROR
 ;BIT12=PARITY ERROR
 4\$: MOV R1, R2 ;DUPLICATE DATA WORD
 BIC #170377, R2 ;MASK LINE#
 SWAB R2 ;LINE # IN LOW BYTE
 CMPB #1, R2 ;CHECK LINE #
 BEQ .+4 ;BRANCH IF OK
 HLT+1 ;WRONG LINE # RECEIVED

```

1507
1508
1509 005270 117702 174070 5$: MOV @DJLEN, R2
1510 005274 130207 BITB R2, R1
1511 005276 001401 BEQ .+4
1512 005300 104002 HLT+2
1513
1514
1515 005302 105102 6$: COMB R2
1516 005304 120102 CMPB R1, R2
1517 005306 001401 BEQ .+4
1518 005310 104002 HLT+2
1519
1520
1521 005312 017701 174004 7$: MOV @RBUF, R1
1522 005316 100001 BPL .+4
1523 005320 104001 HLT+1
1524
1525 005322 017701 173772 8$: MOV @CSR, R1
1526 005326 022701 100405 CMP #100405, R1
1527 005332 001401 BEQ .+4
1528 005334 104001 HLT+1
1529
1530 005336 005077 173762 CLR @TCR
1531 005342 005077 173752 CLR @CSR
1532 005346 104400 SCOPE
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550 005350 004737 014600
1551
1552
1553
1554
1555
1556
1557 005354 052777 000004 173742
1558 005362 017701 173732
1559 005366 100375
1560 005370 012777 000377 173730
1561 005376 005000
1562 005400 105305

```

```

;R1 = CONTENTS OF RBUF
;BITS8-11 - LINE #
;GET MASK OF CHARACTER
;CHECK CHAR LENGTH.
;BRANCH IF OK
;WRONG CHARACTER LENGTH
;R1=DATA FROM FI/FO
;R2=MASK (BITS SET NOT EXPECTED)
;REVERSE THE MASK
;CHECK THE ACTUAL DATA
;BRANCH IF OK
;WRONG CHAR LEN OR DATA ERROR
;R1=DATA FROM FI/FO (COMPLETE WORD)
;R2=DATA (LOW BYTE) EXPECTED
;READ FI/FO
;BRANCH IF CHAR PRESENT NOT SET
;CHARACTER PRESENT STAYED SET
;R1 = CONTENTS OF RBUF
;SAVE THE CSR
;CHECK THE CSR
;BRANCH IF OK
;DONE DIDN'T CLEAR OR OTHER CSR ERROR
;R1 = CONTENTS OF CSR
;CLEAR TCR
;CLEAR CSR

```

```

*****
:TEST 25: TEST THAT LINE 2 CAN TRANSMIT AND
RECEIVE A CHARACTER. (377)
:
: 1$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
: 3$: CHECKS THAT NO ERRORS IN FI/FO
: 4$: CHECKS THAT RIGHT LINE # (2) IN FI/FO
: 5$: CHECKS FOR RIGHT CHARACTER LENGTH
: 6$: CHECKS THAT CORRECT DATA WAS RECEIVED
: 7$: CHECKS THAT CHARACTER PRESENT CLEARS
: 8$: CHECKS THAT DONE CLEARS
:PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD DG3 SERIES
*****

```

```

:
: INITIALIZE
: DEVICE 'CSR' REGISTER
:
: TST25: JSR PC, @#INITD SET:
: BIT2 = MAINTENANCE
: BIT3 = CLEAR MOS
: BIT8 = MASTER XMTR SCAN ENB
:
: WAIT FOR MOS TO CLEAR
:
: SET:
: BIT0 = RECEIVER ENABLE
:
: LOP25: BIS #BIT2, @TCR ;SET XMTR CONTROL BIT, LINE 2
MOV @CSR, R1 ;WAIT FOR XMTR READY
BPL LOP25
MOV #377, @RBUF ;SEND A RUFOUT
CLR R0 ;CLEAR COUNTER
1$: DECB R5 ;SHORT WAIT LOOP

```

```

1563 005402 001376      BNE      1$
1564 005404 017701 173712  MOV      @RBUF,R1      ;WAIT FOR CHAR. PRES.
1565 005410 100405      BMI      3$            ;BRANCH WHEN FOUND
1566 005412 005200      INC      R0            ;TIME COUNTER
1567 005414 001371      BNE      1$            ;BRANCH IF NOT TIME-OUT
1568 005416 017702 173676  MOV      @CSR, R2      ;SAVE CSR
1569 005422 104002      HLT+2      ;CHARACTER READY DIDN'T SET
1570                                     ;R1 = CONTENTS OF RBUF
1571                                     ;R2 = CONTENTS OF CSR
1572 005424 050037 001346  3$:     BIS      R0,TIMER    ;SAVE THE TIMER
1573 005430 032701 0700C0  BIT      #70000,R1     ;CHECK FOR ERROR BITS
1574 005434 001401      BEQ      .+4           ;BRANCH IF NONE
1575 005436 104001      HLT+1      ;ERROR IN RECEIVED CHAR
1576                                     ;R1 = CONTENTS OF RBUF
1577                                     ;BIT14=UART OVERRUN
1578                                     ;BIT13=FRAMING ERROR
1579                                     ;BIT12=PARITY ERROR
1580 005440 010102      4$:     MOV      R1, R2      ;DUPLICATE DATA WORD
1581 005442 042702 170377  BIC      #170377,R2    ;MASK LINE#
1582 005446 000302      SWAB    R2            ;LINE # IN LOW BYTE
1583 005450 122702 000002  CMPB    #2., R2       ;CHECK LINE #
1584 005454 001401      BEQ      .+4           ;BRANCH IF OK
1585 005456 104001      HLT+1      ;WRONG LINE # RECEIVED
1586                                     ;R1 = CONTENTS OF RBUF
1587                                     ;BITS8-11 = LINE #
1588 005460 117702 173700  5$:     MOVB    @DJLEN,R2 ;GET MASK OF CHARACTER
1589 005464 130201      BITB    R2, R1        ;CHECK CHAR LENGTH.
1590 005466 001401      BEQ      .+4           ;BRANCH IF OK
1591 005470 104002      HLT+2      ;WRONG CHARACTER LENGTH
1592                                     ;R1=DATA FROM FI/FO
1593                                     ;R2=MASK (BITS SET NOT EXPECTED)
1594 005472 105102      6$:     COMB    R2            ;REVERSE THE MASK
1595 005474 120102      CMPB    R1, R2        ;CHECK THE ACTUAL DATA
1596 005476 001401      BEQ      .+4           ;BRANCH IF OK
1597 005500 104002      HLT+2      ;WRONG CHAR LEN OR DATA ERROR
1598                                     ;R1=DATA FROM FI/FO (COMPLETE WORD)
1599                                     ;R2=DATA (LOW BYTE) EXPECTED
1600 005502 017701 173614  7$:     MOV      @RBUF, R1 ;READ FI/FO
1601 005506 100001      BPL      .+4           ;BRANCH IF CHAR PRESENT NOT SET
1602 005510 104001      HLT+1      ;CHARACTER PRESENT STAYED SET
1603                                     ;R1 = CONTENTS OF RBUF
1604 005512 017701 173602  8$:     MOV      @CSR, R1 ;SAVE THE CSR
1605 005516 022701 100405  CMP      #100405,R1    ;CHECK THE CSR
1606 005522 001401      BEQ      .+4           ;BRANCH IF OK
1607 005524 104001      HLT+1      ;DONE DIDN'T CLEAR OR OTHER CSR ERROR
1608                                     ;R1 = CONTENTS OF CSR
1609 005526 005077 173572  CLR      @TCR          ;CLEAR TCR
1610 005532 005077 173562  CLR      @CSR          ;CLEAR CSR
1611 005536 104400      SCOPE

```

```

1612
1613 ;*****
1614 ;TEST 26: TEST THAT LINE 3 CAN TRANSMIT AND
1615 ; RECEIVE A CHARACTER. (377)
1616 ;
1617 ; 1$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
1618 ; 3$: CHECKS THAT NO ERRORS IN FI/FO
; 4$: CHECKS THAT RIGHT LINE # (3) IN FI/FO

```

```

1619          :          5$: CHECKS FOR RIGHT CHARACTER LENGTH
1620          :          6$: CHECKS THAT CORRECT DATA WAS RECEIVED
1621          :          7$: CHECKS THAT CHARACTER PRESENT CLEARS
1622          :          8$: CHECKS THAT DONE CLEARS
1623          :PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES
1624          :*****
1625          :
1626          :          INITIALIZE
1627          :          DEVICE 'CSR' REGISTER
1628          :
1629 005540 004737 014600 TST26: JSR    PC,    @#INIT'D SET:
1630          :          ;BIT2 = MAINTENANCE
1631          :          ;BIT3 = CLEAR MOS
1632          :          ;BIT8 = MASTER XMTR SCAN ENB
1633          :
1634          :          ;WAIT FOR MOS TO CLEAR
1635          :
1636 005544 052777 000010 173552 LOP26: BIS    #BIT3, @TCR ;SET XMTR CONTROL BIT, LINE 3
1637 005552 017701 173542          MOV    @CSR,  R1 ;WAIT FOR XMTR READY
1638 005556 100375          BPL    LOP26
1639 005560 012777 000377 173540          MOV    #377, @TBUF ;SEND A RUBCUT
1640 005566 005000          CLR    R0 ;CLEAR COUNTER
1641 005570 105305          1$: DEC  R5 ;SHORT WAIT LOOP
1642 005572 001376          BNE    1$
1643 005574 017701 173522          MOV    @RBUF, R1 ;WAIT FOR CHAR. PRES.
1644 005600 100405          BMI    3$ ;BRANCH WHEN FOUND
1645 005602 005200          INC    R0 ;TIME COUNTER
1646 005604 001371          BNE    1$ ;BRANCH IF NOT TIME-OUT
1647 005606 017702 173506          MOV    @CSR,  R2 ;SAVE CSR
1648 005612 104002          HLT+2 ;CHARACTER READY DIDN'T SET
1649          :          ;R1 = CONTENTS OF RBUF
1650          :          ;R2 = CONTENTS OF CSR
1651 005614 050037 001346          3$: BIS    R0, TIMER ;SAVE THE TIMER
1652 005620 032701 070000          BIT    #70000, R1 ;CHECK FOR ERROR BITS
1653 005624 001401          BEQ    .+4 ;BRANCH IF NONE
1654 005626 104001          HLT+1 ;ERROR IN RECEIVED CHAR
1655          :          ;R1 = CONTENTS OF RBUF
1656          :          ;BIT14=UART OVERRUN
1657          :          ;BIT13=FRAMING ERROR
1658          :          ;BIT12=PARITY ERROR
1659 005630 010102          4$: MOV    R1,    R2 ;DUPLICATE DATA WORD
1660 005632 042702 170377          BIC    #170377, R2 ;MASK LINE#
1661 005636 000302          SWAB  R2 ;LINE # IN LOW BYTE
1662 005640 122702 000003          CMPB  #3,    R2 ;CHECK LINE #
1663 005644 001401          BEQ    .+4 ;BRANCH IF OK
1664 005646 104001          HLT+1 ;WRONG LINE # RECEIVED
1665          :          ;R1 = CONTENTS OF RBUF
1666          :          ;BITS8-11 = LINE #
1667 005650 117702 173510          5$: MOVB  @DJLEN, R2 ;GET MASK OF CHARACTER
1668 005654 130201          BITB  R2,    R1 ;CHECK CHAR LENGTH.
1669 005656 001401          BEQ    .+4 ;BRANCH IF OK
1670 005660 104002          HLT+2 ;WRONG CHARACTER LENGTH
1671          :          ;R1=DATA FROM FI/FO
1672          :          ;R2=MASK (BITS SET NOT EXPECTED)
1673 005662 105102          6$: COMB  R2 ;REVERSE THE MASK
1674 005664 120102          CMPB  R1,    R2 ;CHECK THE ACTUAL DATA
  
```

```

1675 005666 001401 BEQ .+4 ;BRANCH IF OK
1676 005670 104002 HLT+2 ;WRONG CHAR LEN OR DATA ERROR
1677 ;R1=DATA FROM FI/FO (COMPLETE WORD)
1678 ;R2=DATA (LOW BYTE) EXPECTED
1679 005672 017701 173424 7$: MOV @RBUF, R1 ;READ FI/FO
1680 005676 100001 BPL .+4 ;BRANCH IF CHAR PRESENT NOT SET
1681 005700 104001 HLT+1 ;CHARACTER PRESENT STAYED SET
1682 ;R1 = CONTENTS OF RBUF
1683 005702 017701 173412 8$: MOV @CSR, R1 ;SAVE THE CSR
1684 005706 022701 100405 CMP #100405,R1 ;CHECK THE CSR
1685 005712 001401 BEQ .+4 ;BRANCH IF OK
1686 005714 104001 HLT+1 ;DONE DIDN'T CLEAR OR OTHER CSR ERROR
1687 ;R1 = CONTENTS OF CSR
1688 005716 005077 173402 CLR @TCR ;CLEAR TCR
1689 005722 005077 173372 CLR @CSR ;CLEAR CSR
1690 005726 104400 SCOPE
1691 005730 005237 001364 INC DJLEN
1692 005734 012737 005742 015250 MOV #.+6, LAD
  
```

```

1693
1694
1695 :*****
1696 :TEST 27: TEST THAT LINE 4 CAN TRANSMIT AND
1697 : RECEIVE A CHARACTER. (377)
1698 : 1$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
1699 : 3$: CHECKS THAT NO ERRORS IN FI/FO
1700 : 4$: CHECKS THAT RIGHT LINE # (4) IN FI/FO
1701 : 5$: CHECKS FOR RIGHT CHARACTER LENGTH
1702 : 6$: CHECKS THAT CORRECT DATA WAS RECEIVED
1703 : 7$: CHECKS THAT CHARACTER PRESENT CLEARS
1704 : 8$: CHECKS THAT DONE CLEARS
1705 :PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES
1706 :*****
  
```

```

1707 :
1708 : INITIALIZE
1709 : DEVICE 'CSR' REGISTER
1710 005742 004737 014600 TST27: JSR PC, @#INITD SET:
1711 ;BIT2 = MAINTENANCE
1712 ;BIT3 = CLEAR MOS
1713 ;BIT8 = MASTER XMTR SCAN ENB
1714 :WAIT FOR MOS TO CLEAR
1715 :
1716 : SET:
1717 :BIT0 = RECEIVER ENABLE
1718 005746 052777 000020 173350 LOP27: BIS #BIT4, @TCR ;SET XMTR CONTROL BIT, LINE 4
1719 005754 017701 173340 MOV @CSR, R1 ;WAIT FOR XMTR READY
1720 005760 100375 BPL LOP27
1721 005762 012777 000377 173336 MOV #377, @TBUF ;SEND A RUBOUT
1722 005770 005000 CLR R0 ;CLEAR COUNTER
1723 005772 105305 1$: DECB R5 ;SHORT WAIT LOOP
1724 005774 001376 BNE 1$
1725 005776 017701 173320 MOV @RBUF,R1 ;WAIT FOR CHAR. PRES.
1726 006002 100405 BMI 3$ ;BRANCH WHEN FOUND
1727 006004 005200 INC R0 ;TIME COUNTER
1728 006006 001371 BNE 1$ ;BRANCH IF NOT TIME-OUT
1729 006010 017702 173304 MOV @CSR, R2 ;SAVE CSR
1730 006014 104002 HLT+2 ;CHARACTER READY DIDN'T SET
;R1 = CONTENTS OF RBUF
  
```



```

1731
1732 006016 050037 001346      3$:   BIS      RO,TIMER      ;R2 = CONTENTS OF CSR
1733 006022 032701 070000      BIT      #70000,R1      ;SAVE THE TIMER
1734 006026 001401              BEQ      .+4             ;CHECK FOR ERROR BITS
1735 006030 104001              HLT+1          ;BRANCH IF NONE
1736
1737
1738
1739
1740 006032 010102      4$:   MOV      R1, R2      ;R1 = CONTENTS OF RBUF
1741 006034 042702 170377      BIC      #170377,R2     ;BIT14=UART OVERRUN
1742 006040 000302              SWAB      R2           ;BIT13=FRAMING ERROR
1743 006042 122702 000004      CMPB     #4., R2       ;BIT12=PARITY ERROR
1744 006046 001401              BEQ      .+4             ;DUPLICATE DATA WORD
1745 006050 104001              HLT+1          ;MASK LINE#
1746
1747
1748 006052 117702 173306      5$:   MOVB     @DJLEN,R2  ;LINE # IN LOW BYTE
1749 006056 130201              BITB     R2, R1        ;CHECK LINE #
1750 006060 001401              BEQ      .+4             ;CHECK LINE #
1751 006062 104002              HLT+2          ;BRANCH IF OK
1752
1753
1754 006064 105102      6$:   COMB     R2           ;WRONG LINE # RECEIVED
1755 006066 120102              CMPB     R1, R2       ;R1 = CONTENTS OF RBUF
1756 006070 001401              BEQ      .+4             ;BITS8-11 = LINE #
1757 006072 104002              HLT+2          ;GET MASK OF CHARACTER
1758
1759
1760 006074 017701 173222      7$:   MOV      @RBUF, R1   ;CHECK CHAR LENGTH.
1761 006100 100001              BPL      .+4             ;BRANCH IF OK
1762 006102 104001              HLT+1          ;WRONG CHARACTER LENGTH
1763
1764 006104 017701 173210      8$:   MOV      @CSR, R1    ;R1=DATA FROM FI/FO
1765 006110 022701 100405      CMP      #100405,R1    ;R2=MASK (BITS SET NOT EXPECTED)
1766 006114 001401              BEQ      .+4             ;REVERSE THE MASK
1767 006116 104001              HLT+1          ;CHECK THE ACTUAL DATA
1768
1769 006120 005077 173200      CLR      @TCR          ;BRANCH IF OK
1770 006124 005077 173170      CLR      @CSR          ;WRONG CHAR LEN OR DATA ERROR
1771 006130 104400      SCOPE              ;R1=DATA FROM FI/FO (COMPLETE WORD)
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786

```

```

:*****
:TEST 30:      TEST THAT LINE 5 CAN TRANSMIT AND
:              RECEIVE A CHARACTER. (377)
:
:      1$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
:      3$: CHECKS THAT NO ERRORS IN FI/FO
:      4$: CHECKS THAT RIGHT LINE # (5) IN FI/FO
:      5$: CHECKS FOR RIGHT CHARACTER LENGTH
:      6$: CHECKS THAT CORRECT DATA WAS RECEIVED
:      7$: CHECKS THAT CHARACTER PRESENT CLEARS
:      8$: CHECKS THAT DONE CLEARS
:PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES
:*****
:
:              INITIALIZE

```


1843 006274 017701 173020
 1844 006300 022701 100405
 1845 006304 001401
 1846 006306 104001
 1847
 1848 006310 005077 173010
 1849 006314 005077 173000
 1850 006320 104400
 1851
 1852
 1853
 1854
 1855
 1856
 1857
 1858
 1859
 1860
 1861
 1862
 1863
 1864
 1865
 1866
 1867
 1868 006322 004737 014600
 1869
 1870
 1871
 1872
 1873
 1874
 1875 006326 052777 000100 172770
 1876 006334 017701 172760
 1877 006340 100375
 1878 006342 012777 000377 172756
 1879 006350 005000
 1880 006352 105305
 1881 006354 001376
 1882 006356 017701 172740
 1883 006362 100405
 1884 006364 005200
 1885 006366 001371
 1886 006370 017702 172724
 1887 006374 104002
 1888
 1889
 1890 006376 050037 001346
 1891 006402 032701 070000
 1892 006406 001401
 1893 006410 104001
 1894
 1895
 1896
 1897
 1898 006412 010102

```

8$:  MOV @CSR, R1 ;SAVE THE CSR
      CMP #100405,R1 ;CHECK THE CSR
      BEQ .+4 ;BRANCH IF OK
      HLT+1 ;DONE DIDN'T CLEAR OR OTHER CSR ERROR
                                ;R1 = CONTENTS OF CSR
      CLR @TCR ;CLEAR TCR
      CLR @CSR ;CLEAR CSR
      SCOPE

:*****
:TEST 31: TEST THAT LINE 6 CAN TRANSMIT AND
          RECEIVE A CHARACTER. (377)
:      1$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
:      3$: CHECKS THAT NO ERRORS IN FI/FO
:      4$: CHECKS THAT RIGHT LINE # (6) IN FI/FO
:      5$: CHECKS FOR RIGHT CHARACTER LENGTH
:      6$: CHECKS THAT CORRECT DATA WAS RECEIVED
:      7$: CHECKS THAT CHARACTER PRESENT CLEARS
:      8$: CHECKS THAT DONE CLEARS
:PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES
:*****

:      INITIALIZE
:      DEVICE 'CSR' REGISTER
TST31: JSR PC, @WINITD SET:
                                ;BIT2 = MAINTENANCE
                                ;BIT3 = CLEAR MOS
                                ;BIT8 = MASTER XMTR SCAN ENB
:WAIT FOR MOS TO CLEAR
:
:      SET:
:      ;BIT0 = RECEIVER ENABLE
:
LOP31: BIS #BIT6, @TCR ;SET XMTR CONTROL BIT, LINE 6
      MOV @CSR, R1 ;WAIT FOR XMTR READY
      BPL LOP31
      MOV #377, @TBUF ;SEND A RUBOUT
      CLR R0 ;CLEAR COUNTER
      1$: DECB R5 ;SHORT WAIT LOOP
      BNE 1$
      MOV @RBUF,R1 ;WAIT FOR CHAR. PRES.
      BMI 3$ ;BRANCH WHEN FOUND
      INC R0 ;TIME COUNTER
      BNE 1$ ;BRANCH IF NOT TIME-OUT
      MOV @CSR, R2 ;SAVE CSR
      HLT+2 ;CHARACTER READY DIDN'T SET
                                ;R1 = CONTENTS OF RBUF
                                ;R2 = CONTENTS OF CSR
      3$: BIS R0,TIMER ;SAVE THE TIMER
      BIT #70000, R1 ;CHECK FOR ERROR BITS
      BFO .+4 ;BRANCH IF NONE
      HLT+1 ;ERROR IN RECEIVED CHAR
                                ;R1 = CONTENTS OF RBUF
                                ;BIT14=UART OVERRUN
                                ;BIT13=FRAMING ERROR
                                ;BIT12=PARITY ERROR
      4$: MOV R1, R2 ;DUPLICATE DATA WORD
  
```

```

1899 006414 042702 170377      BIC      #170377,R2      ;MASK LINE#
1900 006420 000302              SWAB     R2              ;LINE # IN LOW BYTE
1901 006422 122702 000006      CMPB    #6., R2        ;CHECK LINE #
1902 006426 001401              BEQ     .+4              ;BRANCH IF OK
1903 006430 104001              HLT+1    ;WRONG LINE # RECEIVED
1904                                ;R1 = CONTENTS OF RBUF
1905                                ;BITS8-11 = LINE #
1906 006432 117702 172726      5$:     MOVB    @DJLEN, R2      ;GET MASK OF CHARACTER
1907 006436 130201              BITB    R2, R1          ;CHECK CHAR LENGTH.
1908 006440 001401              BEQ     .+4              ;BRANCH IF OK
1909 006442 104002              HLT+2    ;WRONG CHARACTER LENGTH
1910                                ;R1=DATA FROM FI/FO
1911                                ;R2=MASK (BITS SET NOT EXPECTED)
1912 006444 105102      6$:     COMB    R2              ;REVERSE THE MASK
1913 006446 120102      CMPB    R1, R2          ;CHECK THE ACTURAL DATA
1914 006450 001401              BEQ     .+4              ;BRANCH IF OK
1915 006452 104002              HLT+2    ;WRONG CHAR LEN OR DATA ERROR
1916                                ;R1=DATA FROM FI/FO (COMPLETE WORD)
1917                                ;R2=DATA (LOW BYTE) EXPECTED
1918 006454 017701 172642      7$:     MOV     @RBUF, R1      ;READ FI/FO
1919 006460 100001              BPL     .+4              ;BRANCH IF CHAR PRESENT NOT SET
1920 006462 104001              HLT+1    ;CHARACTER PRESENT STAYED SET
1921                                ;R1 = CONTENTS OF RBUF
1922 006464 017701 172630      8$:     MOV     @CSR, R1      ;SAVE THE CSR
1923 006470 022701 100405      CMP     #100405,R1      ;CHECK THE CSR
1924 006474 001401              BEQ     .+4              ;BRANCH IF OK
1925 006476 104001              HLT+1    ;DONE DIDN'T CLEAR OR OTHER CSR ERROR
1926                                ;R1 = CONTENTS OF CSR
1927 006500 005077 172620      CLR     @TCR            ;CLEAR TCR
1928 006504 005077 172610      CLR     @CSR            ;CLEAR CSR
1929 006510 104400              SCOPE

```

```

1930
1931
1932 :*****
1933 :TEST 32:      TEST THAT LINE 7 CAN TRANSMIT AND
1934 :              RECEIVE A CHARACTER. (377)
1935 :              1$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
1936 :              3$: CHECKS THAT NO ERRORS IN FI/FO
1937 :              4$: CHECKS THAT RIGHT LINE # (7) IN FI/FO
1938 :              5$: CHECKS FOR RIGHT CHARACTER LENGTH
1939 :              6$: CHECKS THAT CORRECT DATA WAS RECEIVED
1940 :              7$: CHECKS THAT CHARACTER PRESENT CLEARS
1941 :              8$: CHECKS THAT DONE CLEARS
1942 :PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL: M7279 ALL: UART CARD D03 SERIES
1943 :*****

```

```

1944 :
1945 :              INITIALIZE
1946 :              DEVICE'CSR'REGISTER
1947 006512 004737 014600      TST32: JSR     PC, @#INITD SET:
1948                                ;BIT2 = MAINTENANCE
1949                                ;BIT3 = CLEAR MOS
1950                                ;BIT8 = MASTER XMTR SCAN ENB
1951 :WAIT FOR MOS TO CLEAR
1952 :
1953 :
1954 006516 052777 000200 172600      BIS     #BIT7, @TCR      ;SET XMTR CONTROL BIT, LINE 7

```

F 4

7Z-CZDJA-F-0 DJ11 LOGIC TESTS MACY11 30A(1052) 26-FEB-79 15:47 PAGE 45
 CZDJA.F.P11 26-FEB-79 15:41 TST32: TEST ALL OF LINE 7 TRANSMIT AND RECEIVE LOGIC SEQ 0044

1955	006524	017701	172570		LOP32:	MOV	@CSR,	R1		:WAIT FOR XMTR READY
1956	006530	100375				BPL	LOP32			
1957	006532	012777	000377	172566		MOV	#377,	@RBUF		:SEND A RUBOUT
1958	006540	005000				CLR	R0			:CLEAR COUNTER
1959	006542	105305			1\$:	DECB	R5			:SHORT WAIT LOOP
1960	006544	001376				BNE	1\$			
1961	006546	017701	172550			MOV	@RBUF,R1			:WAIT FOR CHAR. PRES.
1962	006552	100405				BMI	3\$:BRANCH WHEN FOUND
1963	006554	005200				INC	R0			:TIME COUNTER
1964	006556	001371				BNE	1\$:BRANCH IF NOT TIME-OUT
1965	006560	017702	172534			MOV	@CSR,	R2		:SAVE CSR
1966	006564	104002				HLT+2				:CHARACTER READY DIDN'T SET
1967										:R1 = CONTENTS OF RBUF
1968										:R2 = CONTENTS OF CSR
1969	006566	050037	001346		3\$:	BIS	R0,TIMER			:SAVE THE TIMER
1970	006572	032701	070000			BIT	#70000,R1			:CHECK FOR ERROR BITS
1971	006576	001401				BEQ	+.4			:BRANCH IF NONE
1972	006600	104001				HLT+1				:ERROR IN RECEIVED CHAR
1973										:R1 = CONTENTS OF RBUF
1974										:BIT14=UART OVERRUN
1975										:BIT13=FRAMING ERROR
1976										:BIT12=PARITY ERROR
1977	006602	010102			4\$:	MOV	R1,	R2		:DUPLICATE DATA WORD
1978	006604	042702	170377			BIC	#170377,R2			:MASK LINE#
1979	006610	000302				SWAB	R2			:LINE # IN LOW BYTE
1980	006612	122702	000007			CMPB	#7.,	R2		:CHECK LINE #
1981	006616	001401				BEQ	+.4			:BRANCH IF OK
1982	006620	104001				HLT+1				:WRONG LINE # RECEIVED
1983										:R1 = CONTENTS OF RBUF
1984										:BITS8-11 = LINE #
1985	006622	117702	172536		5\$:	MOV8	@DJLEN,	R2		:GET MASK OF CHARACTER
1986	006626	130201				BITB	R2,	R1		:CHECK CHAR LENGTH.
1987	006630	001401				BEQ	+.4			:BRANCH IF OK
1988	006632	104002				HLT+2				:WRONG CHARACTER LENGTH
1989										:R1=DATA FROM FI/FO
1990										:R2=MASK (BITS SET NOT EXPECTED)
1991	006634	105102			6\$:	COMB	R2			:REVERSE THE MASK
1992	006636	120102				CMPB	R1,	R2		:CHECK THE ACTUAL DATA
1993	006640	001401				BEQ	+.4			:BRANCH IF OK
1994	006642	104002				HLT+2				:WRONG CHAR LEN OR DATA ERROR
1995										:R1=DATA FROM FI/FO (COMPLETE WORD)
1996										:R2=DATA (LOW BYTE) EXPECTED
1997	006644	017701	172452		7\$:	MOV	@RBUF,	R1		:READ FI/FO
1998	006650	100001				BPL	+.4			:BRANCH IF CHAR PRESENT NOT SET
1999	006652	104001				HLT+1				:CHARACTER PRESENT STAYED SET
2000										:R1 = CONTENTS OF RBUF
2001	006654	017701	172440		8\$:	MOV	@CSR,	R1		:SAVE THE CSR
2002	006660	022701	100405			CMP	#100405,R1			:CHECK THE CSR
2003	006664	001401				BEQ	+.4			:BRANCH IF OK
2004	006666	104001				HLT+1				:DONE DIDN'T CLEAR OR OTHER CSR ERROR
2005										:R1 = CONTENTS OF CSR
2006	006670	005077	172430			CLR	@TCR			:CLEAR TCR
2007	006672	005077	172427			CLR	@CSR			:CLEAR CSR
2008	006674	104400				SCOPE				
2009	006702	005237	001364			INC	DJLEN			
2010	006706	012737	006714	C15250		MOV	#.+6,	LAD		

2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066

006714 004737 014600

006720 052777 000400 172376
006726 017701 172366
006732 100375
006734 012777 000377 172364
006742 005000
006744 105305
006746 001376
006750 017701 172346
006754 100405
006756 005200
006760 001371
006762 017702 172332
006766 104002

006770 050037 001346
006774 032701 070000
007000 001401
007002 104001

007004 010102
007006 042702 170377
007012 000302
007014 122702 000010
007020 001401
007022 104001

007024 117702 172334

:TEST 33: TEST THAT LINE 8 CAN TRANSMIT AND
RECEIVE A CHARACTER. (377)
: 1\$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
: 3\$: CHECKS THAT NO ERRORS IN FI/FO
: 4\$: CHECKS THAT RIGHT LINE # (8) IN FI/FO
: 5\$: CHECKS FOR RIGHT CHARACTER LENGTH
: 6\$: CHECKS THAT CORRECT DATA WAS RECEIVED
: 7\$: CHECKS THAT CHARACTER PRESENT CLEARS
: 8\$: CHECKS THAT DONE CLEARS
:PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES

: INITIALIZE
: DEVICE 'CSR' REGISTER
TST33: JSR PC, @#INITD SET:
:BIT2 = MAINTENANCE
:BIT3 = CLEAR MOS
:BIT8 = MASTER XMTR SCAN ENB
:WAIT FOR MOS TO CLEAR
: SET:
:BIT0 = RECEIVER ENABLE
: LOP33: BIS #BIT8, @TCR ;SET XMTR CONTROL BIT, LINE 8
: MOV @CSR, R1 ;WAIT FOR XMTR READY
: BPL LOP33
: MOV #377, @RBUF ;SEND A RUBOUT
: CLR R0 ;CLEAR COUNTER
: 1\$: DECB R5 ;SHORT WAIT LOOP
: BNE 1\$
: MOV @RBUF, R1 ;WAIT FOR CHAR. PRES.
: BMI 3\$;BRANCH WHEN FOUND
: INC R0 ;TIME COUNTER
: BNE 1\$;BRANCH IF NOT TIME-OUT
: MOV @CSR, R2 ;SAVE CSR
: HLT+2 ;CHARACTER READY DIDN'T SET
: R1 = CONTENTS OF RBUF
: R2 = CONTENTS OF CSR
: 3\$: BIS R0, TIMER ;SAVE THE TIMER
: BIT #70000, R1 ;CHECK FOR ERROR BITS
: BEQ .+4 ;BRANCH IF NONE
: HLT+1 ;ERROR IN RECEIVED CHAR
: R1 = CONTENTS OF RBUF
: BIT14=UART OVERRUN
: BIT13=FRAMING ERROR
: BIT12=PARITY ERROR
: 4\$: MOV R1, R2 ;DUPLICATE DATA WORD
: BIC #170377, R2 ;MASK LINE#
: SWAB R2 ;LINE # IN LOW BYTE
: CMPB #8, R2 ;CHECK LINE #
: BEQ .+4 ;BRANCH IF OK
: HLT+1 ;WRONG LINE # RECEIVED
: R1 = CONTENTS OF RBUF
: BIT8-11 = LINE #
: 5\$: MOVB @DJLEN, R2 ;GET MASK OF CHARACTER

```

2067 007030 130201          BITB  R2,   R1      ;CHECK CHAR LENGTH.
2068 007032 001401          BEQ   .+4          ;BRANCH IF OK
2069 007034 104002          HLT+2          ;WRONG CHARACTER LENGTH
2070                                ;R1=DATA FROM FI/FO
2071                                ;R2=MASK (BITS SET NOT EXPECTED)
2072 007036 105102          6$:  COMB  R2           ;REVERSE THE MASK
2073 007040 120102          CMPB  R1,   R2      ;CHECK THE ACTUAL DATA
2074 007042 001401          BEQ   .+4          ;BRANCH IF OK
2075 007044 104002          HLT+2          ;WRONG CHAR LEN OR DATA ERROR
2076                                ;R1=DATA FROM FI/FO (COMPLETE WORD)
2077                                ;R2=DATA (LOW BYTE) EXPECTED
2078 007046 017701 172250    7$:  MOV   @RBUF, R1      ;READ FI/FO
2079 007052 100001          BPL   .+4          ;BRANCH IF CHAR PRESENT NOT SET
2080 007054 104001          HLT+1          ;CHARACTER PRESENT STAYED SET
2081                                ;R1 = CONTENTS OF RBUF
2082 007056 017701 172236    8$:  MOV   @CSR,  R1      ;SAVE THE CSR
2083 007062 022701 100405    CMP   #100405,R1    ;CHECK THE CSR
2084 007066 001401          BEQ   .+4          ;BRANCH IF OK
2085 007070 104001          HLT+1          ;DONE DIDN'T CLEAR OR OTHER CSR ERROR
2086                                ;R1 = CONTENTS OF CSR
2087 007072 005077 172226    CLR   @TCR          ;CLEAR TCR
2088 007076 005077 172216    CLR   @CSR          ;CLEAR CSR
2089 007102 104400          SCOPE
  
```

```

*****
:TEST 34:          TEST THAT LINE 9 CAN TRANSMIT AND
:                  RECEIVE A CHARACTER. (377)
:
: 1$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
: 3$: CHECKS THAT NO ERRORS IN FI/FO
: 4$: CHECKS THAT RIGHT LINE # (9) IN FI/FO
: 5$: CHECKS FOR RIGHT CHARACTER LENGTH
: 6$: CHECKS THAT CORRECT DATA WAS RECEIVED
: 7$: CHECKS THAT CHARACTER PRESENT CLEARS
: 8$: CHECKS THAT DONE CLEARS
:PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES
*****
  
```

```

:
:          INITIALIZE
:          DEVICE 'CSR' REGISTER
:
2107 007104 004737 014600    TST34: JSR   PC,   @#INITD SET:
:                                          ;BIT2 = MAINTENANCE
:                                          ;BIT3 = CLEAR MCS
:                                          ;BIT8 = MASTER XMTR SCAN ENB
:
:;WAIT FOR MOS TO CLEAR
:
:                                          SET:
:                                          ;BIT0 = RECEIVER ENABLE
:
2114 007110 052777 001000 172206    LOP34: BIS   #BIT9, @TCR  ;SET XMTR CONTROL BIT, LINE 9
2115 007116 017701 172176          MOV   @CSR,  R1      ;WAIT FOR XMTR READY
2116 007122 100375          BPL   LOP34
2117 007124 012777 000377 172174    MOV   #377,   @TBUF  ;SEND A RUBOUT
2118 007132 005000          CLR   R0           ;CLEAR COUNTER
2119 007134 105305          1$:  DECB  R5           ;SHORT WAIT LOOP
2120 007136 001376          BNE   1$
2121 007140 017701 172156    MOV   @RBUF, R1      ;WAIT FOR CHAR. PRES.
2122 007144 100405          BMI   3$           ;BRANCH WHEN FOUND
  
```

2123	007146	005200		INC	R0			:TIME COUNTER
2124	007150	001371		BNF	1\$:BRANCH IF NOT TIME-OUT
2125	007152	017702	172142	MOV	@CSR,	R2		:SAVE CSR
2126	007156	104002		HLT+2				:CHARACTER READY DIDN'T SET
2127								:R1 = CONTENTS OF RBUF
2128								:R2 = CONTENTS OF CSR
2129	007160	050037	001346	3\$: BIS	R0,TIMER			:SAVE THE TIMER
2130	007164	032701	070000	BIT	#70000,	R1		:CHECK FOR ERROR BITS
2131	007170	001401		BEQ	+.4			:BRANCH IF NONE
2132	007172	104001		HLT+1				:ERROR IN RECEIVED CHAR
2133								:R1 = CONTENTS OF RBUF
2134								:BIT14=UART OVERRUN
2135								:BIT13=FRAMING ERROR
2136								:BIT12=PARITY ERROR
2137	007174	010102		4\$: MOV	R1,	R2		:DUPLICATE DATA WORD
2138	007176	042702	170377	BIC	#170377,	R2		:MASK LINE#
2139	007202	000302		SWAB	R2			:LINE # IN LOW BYTE
2140	007204	122702	000011	CMPS	#9.,	R2		:CHECK LINE #
2141	007210	001401		BEQ	+.4			:BRANCH IF OK
2142	007212	104001		HLT+1				:WRONG LINE # RECEIVED
2143								:R1 = CONTENTS OF RBUF
2144								:BITS8-11 = LINE #
2145	007214	117702	172144	5\$: MOV	@JLEN,	R2		:GET MASK OF CHARACTER
2146	007220	130201		BITB	R?,	R1		:CHECK CHAR LENGTH.
2147	007222	001401		BEQ	+.4			:BRANCH IF OK
2148	007224	104002		HLT+2				:WRONG CHARACTER LENGTH
2149								:R1=DATA FROM FI/FO
2150								:R2=MASK (BITS SET NOT EXPECTED)
2151	007226	105102		6\$: COMB	R2			:REVERSE THE MASK
2152	007230	120102		CMPS	R1,	R2		:CHECK THE ACTUAL DATA
2153	007232	001401		BEQ	+.4			:BRANCH IF OK
2154	007234	104002		HLT+2				:WRONG CHAR LEN OR DATA ERROR
2155								:R1=DATA FROM FI/FO (COMPLETE WORD)
2156								:R2=DATA (LOW BYTE) EXPECTED
2157	007236	017701	172060	7\$: MOV	@RBUF,	R1		:READ FI/FO
2158	007242	100001		BPL	+.4			:BRANCH IF CHAR PRESENT NOT SET
2159	007244	104001		HLT+1				:CHARACTER PRESENT STAYED SET
2160								:R1 = CONTENTS OF RBUF
2161	007246	017701	172046	8\$: MOV	@CSR,	R1		:SAVE THE CSR
2162	007252	022701	100405	CMP	#100405,	R1		:CHECK THE CSR
2163	007256	001401		BEQ	+.4			:BRANCH IF OK
2164	007260	104001		HLT+1				:DONE DIDN'T CLEAR OR OTHER CSR ERROR
2165								:R1 = CONTENTS OF CSR
2166	007262	005077	172036	CLR	@TCR			:CLEAR TCR
2167	007266	005077	172026	CLR	@CSR			:CLEAR CSR
2168	007272	104400		SCOPE				

:TEST 35: TEST THAT LINE 10 CAN TRANSMIT AND
RECEIVE A CHARACTER. (377)
: 1\$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
: 3\$: CHECKS THAT NO ERRORS IN FI/FO
: 4\$: CHECKS THAT RIGHT LINE # (10) IN FI/FO
: 5\$: CHECKS FOR RIGHT CHARACTER LENGTH
: 6\$: CHECKS THAT CORRECT DATA WAS RECEIVED
: 7\$: CHECKS THAT CHARACTER PRESENT CLEARS


```

2179 ; 8$: CHECKS THAT DONE CLEARS
2180 ;PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES
2181 ;*****
2182 ;
2183 ; INITIALIZE
2184 ; DEVICE 'CSR' REGISTER
2185 ;
2186 007274 004737 014600 TST35: JSR PC, @#INITD SET:
2187 ;BIT2 = MAINTENANCE
2188 ;BIT3 = CLEAR MOS
2189 ;BIT8 = MASTER XMTR SCAN ENB
2190 ;WAIT FOR MOS TO CLEAR
2191 ;
2192 ;
2193 007300 052777 002000 172016 LOP35: BIS #BIT10, @TCR ;SET XMTR CONTROL BIT, LINE 10
2194 007306 017701 172006 MOV @CSR, R1 ;WAIT FOR XMTR READY
2195 007312 100375 BPL LOP35
2196 007314 012777 000377 172004 MOV #377, @RBUF ;SEND A RUBOUT
2197 007322 005000 CLR R0 ;CLEAR COUNTER
2198 007324 105305 1$: DECR R5 ;SHORT WAIT LOOP
2199 007326 001376 BNE 1$
2200 007330 017701 171766 MOV @RBUF, R1 ;WAIT FOR CHAR. PRES.
2201 007334 100405 BMI 3$ ;BRANCH WHEN FOUND
2202 007336 005200 INC R0 ;TIME COUNTER
2203 007340 001371 BNE 1$ ;BRANCH IF NOT TIME-OUT
2204 007342 017702 171752 MOV @CSR, R2 ;SAVE CSR
2205 007346 104002 HLT+2 ;CHARACTER READY DIDN'T SET
2206 ;R1 = CONTENTS OF RBUF
2207 ;R2 = CONTENTS OF CSR
2208 007350 050037 001346 3$: BIS R0, TIMER ;SAVE THE TIMER
2209 007354 032701 070000 BIT #70000, R1 ;CHECK FOR ERROR BITS
2210 007360 001401 BEO .+4 ;BRANCH IF NONE
2211 007362 104001 HLT+1 ;ERROR IN RECEIVED CHAR
2212 ;R1 = CONTENTS OF RBUF
2213 ;BIT14=UART OVERRUN
2214 ;BIT13=FRAMING ERROR
2215 ;BIT12=PARITY ERROR
2216 007364 010102 4$: MOV R1, R2 ;DUPLICATE DATA WORD
2217 007366 042702 170377 BIC #170377, R2 ;MASK LINE#
2218 007372 000302 SWAB R2 ;LINE # IN LOW BYTE
2219 007374 122702 000012 CMPB #10, R2 ;CHECK LINE #
2220 007400 001401 BEQ .+4 ;BRANCH IF OK
2221 007402 104001 HLT+1 ;WRONG LINE # RECEIVED
2222 ;R1 = CONTENTS OF RBUF
2223 ;BITS8-11 = LINE #
2224 007404 117702 171754 5$: MOVB @DJLEN, R2 ;GET MASK OF CHARACTER
2225 007410 130201 BITB R2, R1 ;CHECK CHAR LENGTH.
2226 007412 001401 BFO .+4 ;BRANCH IF OK
2227 007414 104002 HLT+2 ;WRONG CHARACTER LENGTH
2228 ;R1=DATA FROM FI/FO
2229 ;R2=MASK (BITS SET NOT EXPECTED)
2230 007416 105102 6$: COMB R2, R2 ;REVERSE THE MASK
2231 007420 120102 CMPB R1, R2 ;CHECK THE ACTUAL DATA
2232 007422 001401 BEQ .+4 ;BRANCH IF OK
2233 007424 104002 HLT+2 ;WRONG CHAR LEN OR DATA ERROR
2234 ;R1=DATA FROM FI/FO (COMPLETE WORD)

```

```

2235
2236 007426 017701 171670 7$: MOV @RBUF, R1 ;R2-DATA (LOW BYTE) EXPECTED
2237 007432 100001 BPL .+4 ;READ FI/FO
2238 007434 104001 HLT+1 ;BRANCH IF CHAR F-PRESENT NOT SET
2239 ;CHARACTER PRESENT STAYED SET
2240 007436 017701 171656 8$: MOV @CSR, R1 ;R1 = CONTENTS OF RBUF
2241 007442 022701 100405 CMP #100405, R1 ;SAVE THE CSR
2242 007446 001401 BEQ .+4 ;CHECK THE CSR
2243 007450 104001 HLT+1 ;BRANCH IF OK
2244 ;DONE DIDN'T CLEAR OR OTHER CSR ERROR
2245 007452 005077 171646 CLR @TCR ;R1 = CONTENTS OF CSR
2246 007456 005077 171636 CLR @CSR ;CLEAR TCR
2247 007462 104400 SCOPE ;CLEAR CSR
2248
2249

```

```

*****
:TEST 36: TEST THAT LINE 11 CAN TRANSMIT AND
: RECEIVE A CHARACTER. (377)
: 1$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
: 3$: CHECKS THAT NO ERRORS IN FI/FO
: 4$: CHECKS THAT RIGHT LINE # (11) IN FI/FO
: 5$: CHECKS FOR RIGHT CHARACTER LENGTH
: 6$: CHECKS THAT CORRECT DATA WAS RECEIVED
: 7$: CHECKS THAT CHARACTER PRESENT CLEARS
: 8$: CHECKS THAT DONE CLEARS
:PROBABLE FAULTY LOGIC: M7285 (62-7) ALL; M7279 ALL; UART CARD D03 SERIES
*****

```

INITIALIZE
 DEVICE 'CSR' REGISTER

```

2262
2263
2264
2265 007464 004737 014600 TST36: JSR PC, @#INITD SET:
2266 ;BIT2 = MAINTENANCE
2267 ;BIT3 = CLEAR MOS
2268 ;BIT8 = MASTER XMTR SCAN ENB
2269
2270
2271
2272 007470 052777 004000 171626 :WAIT FOR MOS TO CLEAR SET:
2273 007476 017701 171616 LOP36: BIS #BIT11, @TCR ;SET XMTR CONTROL BIT, LINE 11
2274 007502 100375 MOV @CSR, R1 ;WAIT FOR XMTR READY
2275 007504 012777 000377 171614 BPL LOP36
2276 007512 005000 MOV #377, @RBUF ;SEND A RIBOUT
2277 007514 105305 CLR R0 ;CLEAR COUNTER
2278 007516 001376 1$ DEC R5 ;SHORT WAIT LOOP
2279 007520 017701 171576 BNE 1$
2280 007524 100405 MOV @RBUF, R1 ;WAIT FOR CHAR. PRES.
2281 007526 005200 BMI 3$ ;BRANCH WHEN FOUND
2282 007530 001371 INC R0 ;TIME COUNTER
2283 007532 017702 171562 BNE 1$ ;BRANCH IF NOT TIME-OUT
2284 007536 104002 MOV @CSR, R2 ;SAVE CSR
2285 HLT+2 ;CHARACTER READY DIDN'T SET
2286 ;R1 = CONTENTS OF RBUF
2287 007540 050037 001346 3$: BIS R0, TIMER ;R2 = CONTENTS OF CSR
2288 007544 032701 070000 BIT #70000, R1 ;SAVE THE TIMER
2289 007550 001401 BEQ .+4 ;CHECK FOR ERROR BITS
2290 007552 104001 HLT+1 ;BRANCH IF NONE
;ERROR IN RECEIVED CHAR

```

```

2291
2292
2293
2294
2295 007554 010102
2296 007556 042702 170377
2297 007562 000302
2298 007564 122702 000013
2299 007570 001401
2300 007572 104001
2301
2302
2303 007574 117702 171564
2304 007600 130201
2305 007602 001401
2306 007604 104002
2307
2308
2309 007606 105102
2310 007610 120102
2311 007612 001401
2312 007614 104002
2313
2314
2315 007616 017701 171500
2316 007622 100001
2317 007624 104001
2318
2319 007626 017701 171466
2320 007632 022701 100405
2321 007636 001401
2322 007640 104001
2323
2324 007642 005077 171456
2325 007646 005077 171446
2326 007652 104400
2327 007654 005237 001364
2328 007660 012737 007666 015250
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346 007666 004737 014600
    
```

```

;R1 - CONTENTS OF RBUF
;BIT14=UART OVER RUN
;BIT13=FRAMING ERROR
;BIT12=PARITY ERROR
;DUPLICATE DATA WORD
;MASK LINE#
;LINE # IN LOW BYTE
;CHECK LINE #
;BRANCH IF OK
;WRONG LINE # RECEIVED
;R1 = CONTENTS OF RBUF
;BITS8-11 = LINE #
;GET MASK OF CHARACTER
;CHECK CHAR LENGTH.
;BRANCH IF OK
;WRONG CHARACTER LENGTH
;R1=DATA FROM FI/FO
;R2=MASK (BITS SET NOT EXPECTED)
;REVERSE THE MASK
;CHECK THE ACTURAL DATA
;BRANCH IF OK
;WRONG CHAR LEN OR DATA ERROR
;R1=DATA FROM FI/FO (COMPLETE WORD)
;R2=DATA (LOW BYTE) EXPECTED
;READ FI/FO
;BRANCH IF CHAR PRESENT NOT SET
;CHARACTER PRESENT STAYED SET
;R1 = CONTENTS OF RBUF
;SAVE THE CSR
;CHECK THE CSR
;BRANCH IF OK
;DONE DIDN'T CLEAR OR OTHER CSR ERROR
;R1 = CONTENTS OF CSR
;CLEAR TCR
;CLEAR CSR
    
```

```

*****
:TEST 37: TEST THAT LINE 12 CAN TRANSMIT AND
: RECEIVE A CHARACTER. (377)
:
: 1$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
: 3$: CHECKS THAT NO ERRORS IN FI/FO
: 4$: CHECKS THAT RIGHT LINE # (12) IN FI/FO
: 5$: CHECKS FOR RIGHT CHARACTER LENGTH
: 6$: CHECKS THAT CORRECT DATA WAS RECEIVED
: 7$: CHECKS THAT CHARACTER PRESENT CLEARS
: 8$: CHECKS THAT DONE CLEARS
:
:PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES
*****
:
: INITIALIZE
: DEVICE 'CSR' REGISTER
:
: TST37: JSR PC, @#INITD SET:
: :BIT2 - MAINTENANCE
    
```

```

2347
2348
2349
2350
2351
2352
2353 007672 052777 010000 171424
2354 007700 017701 171414
2355 007704 100375
2356 007706 012777 000377 171412
2357 007714 005000
2358 007716 105305
2359 007720 001376
2360 007722 017701 171374
2361 007726 100405
2362 007730 005200
2363 007732 001371
2364 007734 017702 171360
2365 007740 104002
2366
2367
2368 007742 050037 001346
2369 007746 032701 070000
2370 007752 001401
2371 007754 104001
2372
2373
2374
2375
2376 007756 010102
2377 007760 042702 170377
2378 007764 000302
2379 007766 122702 000014
2380 007772 001401
2381 007774 104001
2382
2383
2384 007776 117702 171362
2385 010002 130201
2386 010004 001401
2387 010006 104002
2388
2389
2390 010010 105102
2391 010012 120102
2392 010014 001401
2393 010016 104002
2394
2395
2396 010020 017701 171276
2397 010024 100001
2398 010026 104001
2399
2400 010030 017701 171264
2401 010034 022701 100405
2402 010040 001401

```

:WAIT FOR MOS TO CLEAR

...

LOP37: BIS #BIT12, @TCR ;SET XMTR CONTROL BIT, LINE 12
 MOV @CSR, R1 ;WAIT FOR XMTR READY
 BPL LOP37
 MOV #377, @TBUF ;SEND A RUBOUT
 CLR R0 ;CLEAR COUNTER
 1\$: DECB R5 ;SHORT WAIT LOOP
 BNE 1\$
 MOV @RBUF, P1 ;WAIT FOR CHAR. PRES.
 BMI 3\$;BRANCH WHEN FOUND
 INC R0 ;TIME COUNTER
 BNE 1\$;BRANCH IF NOT TIME-OUT
 MOV @CSR, R2 ;SAVE CSR
 HLT+2 ;CHARACTER READY DIDN'T SET
 ;R1 = CONTENTS OF RBUF
 ;R2 = CONTENTS OF CSR
 3\$: BIS R0, TIMER ;SAVE THE TIMER
 BIT #70000, R1 ;CHECK FOR ERROR BITS
 BEQ .+4 ;BRANCH IF NONE
 HLT+1 ;ERROR IN RECEIVED CHAR
 ;R1 = CONTENTS OF RBUF
 ;BIT14=UART OVERRUN
 ;BIT13=FRAMING ERROR
 ;BIT12=PARITY ERROR
 4\$: MOV R1, R2 ;DUPLICATE DATA WORD
 BIC #170377, R2 ;MASK LINE#
 SWAB R2 ;LINE # IN LOW BYTE
 CMPB #12, R2 ;CHECK LINE #
 BEQ .+4 ;BRANCH IF OK
 HLT+1 ;WRONG LINE # RECEIVED
 ;R1 = CONTENTS OF RBUF
 ;BITS8-11 = LINE #
 5\$: MOVB @DJLEN, R2 ;GET MASK OF CHARACTER
 BITB R2, R1 ;CHECK CHAR LENGTH.
 BEQ .+4 ;BRANCH IF OK
 HLT+2 ;WRONG CHARACTER LENGTH
 ;R1=DATA FROM FI/FO
 ;R2=MASK (BITS SET NOT EXPECTED)
 6\$: COMB R2 ;REVERSE THE MASK
 CMPB R1, R2 ;CHECK THE ACTURAL DATA
 BEQ .+4 ;BRANCH IF OK
 HLT+2 ;WRONG CHAR LEN OR DATA ERROR
 ;R1=DATA FROM FI/FO (COMPLETE WORD)
 ;R2=DATA (LOW BYTE) EXPECTED
 7\$: MOV @RBUF, R1 ;READ FI/FO
 BPL .+4 ;BRANCH IF CHAR PRESENT NOT SET
 HLT+1 ;CHARACTER PRESENT STAYED SET
 ;R1 = CONTENTS OF RBUF
 8\$: MOV @CSR, R1 ;SAVE THE CSR
 CMP #100405, R1 ;CHECK THE CSR
 BEQ .+4 ;BRANCH IF OK

```

2403 010042 104001          HLT+1          ;DONE DIDN'T CLEAR OR OTHER CSR ERROR
2404                          ;R1 = CONTENTS OF CSR
2405 010044 005077 171254    CLR @TCR      ;CLEAR TCR
2406 010050 005077 171244    CLR @CSR      ;CLEAR CSR
2407 010054 104400          SCOPE
2408
2409
2410 :*****
2411 :TEST 40:          TEST THAT LINE 13 CAN TRANSMIT AND
2412 :                  RECEIVE A CHARACTER. (377)
2413 :                  1$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
2414 :                  3$: CHECKS THAT NO ERRORS IN FI/FO
2415 :                  4$: CHECKS THAT RIGHT LINE # (13) IN FI/FO
2416 :                  5$: CHECKS FOR RIGHT CHARACTER LENGTH
2417 :                  6$: CHECKS THAT CORRECT DATA WAS RECEIVED
2418 :                  7$: CHECKS THAT CHARACTER PRESENT CLEARS
2419 :                  8$: CHECKS THAT DONE CLEARS
2420 :PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES
2421 :*****
2422 :
2423 :              INITIALIZE
2424 :              DEVICE 'CSR' REGISTER
2425 010056 004737 014600    TST40: JSR PC, @#INITD SET:
2426                          ;BIT2 = MAINTENANCE
2427                          ;BIT3 - CLEAR MOS
2428                          ;BIT8 - MASTER XMTR SCAN ENB
2429 :WAIT FOR MOS TO CLEAR
2430 :
2431 :              SET:
2432 :              ;BIT0 = RECEIVER ENABLE
2433 010062 052777 020000 171234 LOP40: BIS #BIT13, @TCR ;SET XMTR CONTROL BIT, LINE 13
2434 010070 017701 171224      MOV @CSR, R1 ;WAIT FOR XMTR READY
2435 010074 100375          BPL LOP40
2436 010076 012777 000377 171222 MOV #377, @TBUF ;SEND A RUBOUT
2437 010104 005000          CLR R0 ;CLEAR COUNTER
2438 010106 105305          1$: DEC R5 ;SHORT WAIT LOOP
2439 010110 001376          BNE 1$
2440 010112 017701 171204      MOV @RBUF, R1 ;WAIT FOR CHAR. PRES.
2441 010116 100405          BMI 3$ ;BRANCH WHEN FOUND
2442 010120 005200          INC R0 ;TIME COUNTER
2443 010122 001371          BNE 1$ ;BRANCH IF NOT TIME-OUT
2444 010124 017702 171170      MOV @CSR, R2 ;SAVE CSR
2445 010130 104002          HLT+2 ;CHARACTER READY DIDN'T SET
2446                          ;R1 = CONTENTS OF RBUF
2447 010132 050037 001346      3$: BIS R0, TIMER ;SAVE THE TIMER
2448 010136 032701 070000      BIT #70000, R1 ;CHECK FOR ERROR BITS
2449 010142 001401          BEQ .+4 ;BRANCH IF NONE
2450 010144 104001          HLT+1 ;ERROR IN RECEIVED CHAR
2451                          ;R1 = CONTENTS OF RBUF
2452                          ;BIT14=UART OVERRUN
2453                          ;BIT13=FRAMING ERROR
2454                          ;BIT12=PARITY ERROR
2455 010146 010102          4$: MOV R1, R2 ;DUPLICATE DATA WORD
2456 010150 042702 170377      BIC #170377, R2 ;MASK LINE#
2457 010154 000302          SWAB R2 ;LINE # IN LOW BYTE
2458 010156 122702 000015      CMPB #13., R2 ;CHECK LINE #
  
```

```

2459 010152 001401          BEQ      .+4          ;BRANCH IF OK
2460 010164 104001          HLT+1                    ;WRONG LINE # RECEIVED
2461                                     ;R1 = CONTENTS OF RBUF
2462                                     ;BITS8-11 = LINE #
2463 010166 117702 171172 5$:  MOVB    @DJLEN, R2      ;GET MASK OF CHARACTER
2464 010172 130201          BITB    R2, R1          ;CHECK CHAR LENGTH.
2465 010174 001401          BEQ      .+4          ;BRANCH IF OK
2466 010176 104002          HLT+2                    ;WRONG CHARACTER LENGTH
2467                                     ;R1=DATA FROM FI/FO
2468                                     ;R2=MASK (BITS SET NOT EXPECTED)
2469 010200 105102 6$:  COMB    R2                    ;REVERSE THE MASK
2470 010202 120102          CMPB    R1, R2          ;CHECK THE ACTURAL DATA
2471 010204 001401          BEQ      .+4          ;BRANCH IF OK
2472 010206 104002          HLT+2                    ;WRONG CHAR LEN OR DATA ERPR
2473                                     ;R1=DATA FROM FI/FO (COMPLETE WORD)
2474                                     ;R2=DATA (LOW BYTE) EXPECTED
2475 010210 017701 171106 7$:  MOV     @RBUF, R1        ;READ FI/FO
2476 010214 100001          BPL     .+4          ;BRANCH IF CHAR PRESENT NOT SET
2477 010216 104001          HLT+1                    ;CHARACTER PRESENT STAYED SET
2478                                     ;R1 = CONTENTS OF RBUF
2479 010220 017701 171074 8$:  MOV     @CSR, R1         ;SAVE THE CSR
2480 010224 022701 100405  CMP     #100405,R1      ;CHECK THE CSR
2481 010230 001401          BEQ      .+4          ;BRANCH IF OK
2482 010232 104001          HLT+1                    ;DONE DIDN'T CLEAR OR OTHER CSR ERROR
2483                                     ;R1 = CONTENTS OF CSR
2484 010234 005077 171064  CLR     @TCR            ;CLEAR TCR
2485 010240 005077 171054  CLR     @CSR            ;CLEAR CSR
2486 010244 104400          SCOPE

```

```

2487
2488
2489 *****
2490 :TEST 41:          TEST THAT LINE 14 CAN TRANSMIT AND
2491                   RECEIVE A CHARACTER. (377)
2492                   1$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
2493                   3$: CHECKS THAT NO ERRORS IN FI/FO
2494                   4$: CHECKS THAT RIGHT LINE # (14) IN FI/FO
2495                   5$: CHECKS FOR RIGHT CHARACTER LENGTH
2496                   6$: CHECKS THAT CORRECT DATA WAS RECEIVED
2497                   7$: CHECKS THAT CHARACTER PRESENT CLEARS
2498                   8$: CHECKS THAT DONE CLEARS
2499 :PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES
2500 *****

```

```

2501 :
2502 :           INITIALIZE
2503 :           DEVICE'CSR'REGISTER
2504 010246 004737 014600 TST41: JSR    PC, @#INITD SET:
2505                                     ;BIT2 = MAINTENANCE
2506                                     ;BIT3 = CLEAR MOS
2507                                     ;BIT8 = MASTER XMTR SCAN ENB
2508 :WAIT FOR MOS TO CLEAR
2509 :
2510 :           SET:
2511 :           ;BIT0 = RECEIVER ENABLE
2512 010252 052777 040000 171044 LOP41: BIS    #BIT14, @TCR ;SET XMTR CONTROL BIT, LINE 14
2513 010260 017701 171034          MOV     @CSR, R1      ;WAIT FOR XMTR READY
2514 010264 100375          BPL     LOP41
2514 010266 012777 000377 171032  MOV     #377, @TBUF ;SEND A RUBOUT

```

```

2515 010274 005000          CLR      R0          ;CLEAR COUNTER
2516 010276 105305          1$:     DECB     R5          ;SHORT WAIT LOOP
2517 010300 001376          BNE     1$
2518 010302 017701 171014  MOV     @RBUF,R1      ;WAIT FOR CHAR. PRES.
2519 010306 100405          BMI     3$          ;BRANCH WHEN FOUND
2520 010310 005200          INC     R0          ;TIME COUNTER
2521 010312 001371          BNE     1$          ;BRANCH IF NOT TIME-OUT
2522 010314 017702 171000  MOV     @CSR, R2      ;SAVE CSR
2523 010320 104002          HLT+2          ;CHARACTER READY DIDN'T SET
2524
2525
2526 010322 050037 001346  3$:     BIS     R0,TIMER  ;SAVE THE TIMER
2527 010326 032701 070000  BIT     #70000,R1     ;CHECK FOR ERROR BITS
2528 010332 001401          BEQ     .+4         ;BRANCH IF NONE
2529 010334 104001          HLT+1         ;ERROR IN RECEIVED CHAR
2530
2531
2532
2533
2534 010336 010102          4$:     MOV     R1, R2      ;DUPLICATE DATA WORD
2535 010340 042702 170377  BIC     #170377,R2    ;MASK LINE#
2536 010344 000302          SWAB   R2          ;LINE # IN LOW BYTE
2537 010346 122702 000016  CMPB   #14., R2      ;CHECK LINE #
2538 010352 001401          BEQ     .+4         ;BRANCH IF OK
2539 010354 104001          HLT+1         ;WRONG LINE # RECEIVED
2540
2541
2542 010356 117702 171002  5$:     MOVB   @DJLEN, R2 ;GET MASK OF CHARACTER
2543 010362 130201          BITB   R2, R1      ;CHECK CHAR LENGTH.
2544 010364 001401          BEQ     .+4         ;BRANCH IF OK
2545 010366 104002          HLT+2         ;WRONG CHARACTER LENGTH
2546
2547
2548 010370 105102          6$:     COMB   R2          ;R2=MASK (BITS SET NOT EXPECTED)
2549 010372 120102          CMPB   R1, R2      ;REVERSE THE MASK
2550 010374 001401          BEQ     .+4         ;CHECK THE ACTUAL DATA
2551 010376 104002          HLT+2         ;BRANCH IF OK
2552
2553
2554 010400 017701 170716  7$:     MOV     @RBUF, R1   ;READ FI/FO
2555 010404 100001          BPL     .+4         ;BRANCH IF CHAR PRESENT NOT SET
2556 010406 104001          HLT+1         ;CHARACTER PRESENT STAYED SET
2557
2558 010410 017701 170704  8$:     MOV     @CSR, R1   ;R1 = CONTENTS OF RBUF
2559 010414 022701 100405  CMP     #100405,R1   ;SAVE THE CSR
2560 010420 001401          BEQ     .+4         ;CHECK THE CSR
2561 010422 104001          HLT+1         ;BRANCH IF OK
2562
2563 010424 005077 170674  CLR     @TCR        ;DONE DIDN'T CLEAR OR OTHER CSR ERROR
2564 010430 005077 170664  CLR     @CSR        ;R1 = CONTENTS OF CSR
2565 010434 104400          SCOPE          ;CLEAR TCR
2566
2567
2568
2569
2570

```

```

*****
;TEST 42: TEST THAT LINE 15 CAN TRANSMIT AND
; RECEIVE A CHARACTER. (377)
;
; 1$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.

```

```

2571 : 3$: CHECKS THAT NO ERRORS IN FI/FO
2572 : 4$: CHECKS THAT RIGHT LINE # (15) IN FI/FO
2573 : 5$: CHECKS FOR RIGHT CHARACTER LENGTH
2574 : 6$: CHECKS THAT CORRECT DATA WAS RECEIVED
2575 : 7$: CHECKS THAT CHARACTER PRESENT CLEARS
2576 : 8$: CHECKS THAT DONE CLEARS
2577 : PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES
2578 : *****
2579 :
2580 : INITIALIZE
2581 : DEVICE 'CSR' REGISTER
2582 :
2583 010436 004737 014600 TST42: JSR PC, @#INITD SET:
2584 : ;BIT2 = MAINTENANCE
2585 : ;BIT3 = CLEAR MOS
2586 : ;BIT8 = MASTER XMTR SCAN ENB
2587 :
2588 : ;WAIT FOR MOS TO CLEAR
2589 :
2590 010442 052777 100000 170654 LOP42: BIS #BIT15, @TCR ;SET XMTR CONTROL BIT, LINE 15
2591 010450 017701 170644 MOV @CSR, R1 ;WAIT FOR XMTR READY
2592 010454 100375 BPL LOP42
2593 010456 012777 000377 170642 MOV #377, @TBUF ;SEND A RUBOUT
2594 010464 005000 CLR R0 ;CLEAR COUNTER
2595 010466 105305 1$: DECB R5 ;SHORT WAIT LOOP
2596 010470 001376 BNE 1$
2597 010472 017701 170624 MOV @RBUF, R1 ;WAIT FOR CHAR. PRES.
2598 010476 100405 BMI 3$ ;BRANCH WHEN FOUND
2599 010500 005200 INC R0 ;TIME COUNTER
2600 010502 001371 BNE 1$ ;BRANCH IF NOT TIME-OUT
2601 010504 017702 170610 MOV @CSR, R2 ;SAVE CSR
2602 010510 104002 HLT+2 ;CHARACTER READY DIDN'T SET
2603 : ;R1 = CONTENTS OF RBUF
2604 : ;R2 = CONTENTS OF CSR
2605 010512 050037 001346 3$: BIS R0, TIMER ;SAVE THE TIMER
2606 010516 032701 070000 BIT #70000, R1 ;CHECK FOR ERROR BITS
2607 010522 001401 BEQ .+4 ;BRANCH IF NONE
2608 010524 104001 HLT+1 ;ERROR IN RECEIVED CHAR
2609 : ;R1 = CONTENTS OF RBUF
2610 : ;BIT14=UART OVERRUN
2611 : ;BIT13=FRAMING ERROR
2612 : ;BIT12=PARITY ERROR
2613 010526 010102 4$: MOV R1, R2 ;DUPLICATE DATA WORD
2614 010530 042702 170377 BIC #170377, R2 ;MASK LINE#
2615 010534 000302 SWAB R2 ;LINE # IN LOW BYTE
2616 010536 122702 000017 CMPB #15., R2 ;CHECK LINE #
2617 010542 001401 BEQ .+4 ;BRANCH IF OK
2618 010544 104001 HLT+1 ;WRONG LINE # RECEIVED
2619 : ;R1 = CONTENTS OF RBUF
2620 : ;BITS8-11 = LINE #
2621 010546 117702 170612 5$: MOVB @DJLEN, R2 ;GET MASK OF CHARACTER
2622 010552 130201 BITB R2, R1 ;CHECK CHAR LENGTH.
2623 010554 001401 BEQ .+4 ;BRANCH IF OK
2624 010556 104002 HLT+2 ;WRONG CHARACTER LENGTH
2625 : ;R1=DATA FROM FI/FO
2626 : ;R2=MASK (BITS SET NOT EXPECTED)

```



```
2627 010560 105102      6$:  COMB      R2      :REVERSE THE MASK
2628 010562 120102      CMPB      R1      R2      :CHECK THE ACTURAL DATA
2629 010564 001401      BEQ      .+4      :BRANCH IF OK
2630 010566 104002      HLT-2      :WRONG CHAR LEN OR DATA ERROR
2631      :R1=DATA FROM FI/FO (COMPLETE WORD)
2632      :R2=DATA (LOW BYTE) EXPECTED
2633 010570 017701 170526 7$:  MOV      @RBUF, R1      :READ FI/FO
2634 010574 100001      BPL      .+4      :BRANCH IF CHAR PRESENT NOT SET
2635 010576 104001      HLT+1      :CHARACTER PRESENT STAYED SET
2636      :R1 = CONTENTS OF RBUF
2637 010600 017701 170514 8$:  MOV      @CSR, R1      :SAVE THE CSR
2638 010604 022701 100405  CMP      #100405,R1      :CHECK THE CSR
2639 010610 001401      BEQ      .+4      :BRANCH IF OK
2640 010612 104001      HLT+1      :DONE DIDN'T CLEAR OR OTHER CSR ERROR
2641      :R1 = CONTENTS OF CSR
2642 010614 005077 170504      CLR      @TCR      :CLEAR TCR
2643 010620 005077 170474      CLR      @CSP      :CLEAR CSR
2644 010624 10440C      SCOPE
2645 010626 162737 000003 001364  SUB      #3, DJLEN
2646 010634 005237 001346      INC      TIMER      :WORSE CASE TIME, ONE CHARACTER
2647 010640 013700 001346      MOV      TIMER, R0      :DUP TIMER
2648 010644 006200      ASR      R0      :/2
2649 010646 006200      ASR      R0      :/4
2650 010650 005200      INC      R0      :+1
2651 010652 006137 001346      ROL      TIMER      :TIMER * 2
2652 010656 060037 001346      ADD      R0, TIMER      :2.25 TIMES ONE CHARACTER TIME
2653 010662 012737 010670 015250  MOV      #.+6, LAD      :RESET LOOP ADDRESS
```

```
*****
:TEST 43: TEST THAT CHARACTER PRESENT (BIT15) OF RBUF
: IS CLEARED (BY CLEAR MOS).
:PROBABLE FAULTY LOGIC: M7285 (D2-8) E17,E14,E15; M7279; M7280
*****
```

```
: INITIALIZE
:
: TST43: JSR PC,@#INITD
:SET:
: BIT2 = MAINTENANCE
: BIT3 = CLEAR MOS
: BIT8 = TRANS SCAN ENABLE
: WAIT FOR BIT4 = MOS CLEAR
```

```
2661
2662
2663 010670 004737 014600 170422 :
2664 :
2665 : BIT0 = RECEIVER ENABLE
2666 :
2667 : MOV #BIT0, @TCR :TRANS CONTROL, LINE0
2668 : MOV #40,R4 :SET UP COUNTER TO OUTPUT 32 CHAR'S
2669 :SET:
2670 :
2671 010674 012777 000001 170406 1$:  MOV      @CSR,R1      :WAIT FOR TRANS READY
2672 010702 012704 000040      MOV      #40,R4
2673 010706 017701 170406      MOV      @CSR,R1
2674 010712 100375      BPL      1$
2675 010714 010477 170406      MOV      R4,@TBUF      :TRANSMIT COUNT
2676 010720 005304      DEC      R4      :COUNT DOWN
2677 010722 001371      BNE      1$
2678
2679 010724 105777 170370 2$:  TSTB      @CSR      :MAKE SURE DONE IS SET
2680 010730 100375      BPL      2$
2681
2682 010732 052777 000010 170360  BIS      #BIT3,@CSR      :CLEAR MOS
```

```

2683 010740 032777 000020 170352 3$: BIT #BIT4,@CSR ;WAIT FOR CLRMOS TO FINISH
2684 010746 001374 BNE 3$
2685
2686 010750 017701 170346 MOV @RBUF,R1 ;CHECK RBUF AND SAVE
2687 010754 100001 BPL .+4
2688 010756 104001 HLT+1
2689 010760 017701 170334 MOV @CSR,R1 ;SAVE CSR
2690 010764 022701 100405 CMP #100405,R1 ;CHECK CSR
2691 010770 001401 BEQ .+4 ;BRANCH IF OK
2692 010772 104001 HLT+1 ;CSR ERROR. POSSIBILITIES:
2693 ;(1) DONE DIDN'T CLEAR
2694
2695 ;(2) TRANSMITTER UART DIDN'T CLR.
2696 ;R1=CONTENTS OF CSR
2697
2698 010774 013700 001346 MOV TIMER, R0 ;SET UP TIMFR
2699 011000 105305 4$: DECB R5 ;SHORT WAIT LOOP
2700 011002 001376 BNE 4$
2701
2702 011004 017701 170310 MOV @CSR,R1 ;SAVE CSR
2703 011010 105701 TSTB R1 ;CHECK FOR DONE
2704 011012 100001 BPL .+4 ;BRANCH IF OK
2705 011014 104001 HLT+1 ;DONE CAME UP!
2706 ;MOS MUST NOT HAVE CLEARED
2707 011016 005300 DEC R0 ;TIMER COUNT
2708 011020 001367 BNE 4$
2709
2710 011022 022701 100405 CMP #100405,R1 ;CHECK CSR
2711 011026 001401 BEQ .+4 ;BRANCH IF OK
2712 011030 104001 HLT+1 ;CSR ERROR
2713
2714 011032 017701 170264 MOV @RBUF,R1 ;CHECK RBUF
2715 011036 100001 BPL .+4 ;BRANCH IF OK
2716 011040 104001 HLT+1 ;RBUF NOT EMPTY!
2717
2718 011042 005077 170256 CLR @TCR
2719 011046 005077 170246 CLR @CSR
2720
2721 011052 104400 SCOPE
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734 011054 004737 014600
2735
2736
2737
2738
  
```

```

*****
:TEST 44: TEST THAT TRANSMITTER READY CLEARS WHEN TBUF IS LOADED
: NOTE: DUE TO THE DOUBLE BUFFERING BY THE UART, TWO CHARACTERS
: MUST BE LOADED TO INSURE SEEING TRANSMITTER READY CLEAR
:PROBABLE FAULTY LOGIC: M7285 (D2-6) E39, E23, E49
*****
  
```

```

: INITIALIZE
: DEVICE 'CSR' REGISTER
:
: TST44: JSR PC,@#INITD SET:
: ;BIT2 = MAINTENANCE
: ;BIT3 = CLEAR MOS
: ;BIT8 = MASTER XMITR SCAN ENB
:
: ;WAIT FOR MOS TO CLEAR
: SET:
  
```

```

2739
2740
2741 011060 052777 000001 170236
2742 011066 017701 170226 1$:
2743 011072 100375
2744 011074 012777 000001 170224
2745 011102 017701 170212 2$:
2746 011106 100375
2747 011110 012777 000002 170210
2748 011116 017701 170176
2749 011122 100001
2750 011124 104001
2751
2752 011126 013700 001346
2753 011132 105305 3$:
2754 011134 001376
2755 011136 017701 170156
2756 011142 000400
2757 011144 005300
2758 011146 001371
2759
2760 011150 042701 000200
2761 011154 022701 100405
2762 011160 001401
2763 011162 104001
2764
2765
2766 011164 017701 170132
2767 011170 100401
2768 011172 104001
2769
2770 011174 022701 100001
2771 011200 001401
2772 011202 104001
2773
2774 011204 017701 170112
2775 011210 100401
2776 011212 104001
2777
2778 011214 022701 100002
2779 011220 001401
2780 011222 104001
2781
2782 011224 017701 170072
2783 011230 100001
2784 011232 104001
2785
2786 011234 017701 170060
2787 011240 022701 100405
2788 011244 001401
2789 011246 104001
2790
2791 011250 005077 170050
2792 011254 005077 170040
2793 011260 104400
2794
  
```

```

;BIT0 - RECEIVER ENABLE
;TRANS CONTROL, LINE 0
;WAIT FOR XMTR READY
;TRANSMIT A 1
;WAIT FOR XMTR READY
;TRANSMIT A 2
;CHECK FOR XMTR READY
;BRANCH IF XMTR READY CLEARED
;TRANSMITTER READY FAILED TO CLEAR
;R1 = CONTENTS OF CSR
;SET UP TIMER
;SHORT WAIT LOOP
;SAVE CSR FOR THE RECORD
;NOP FOR TIMING
;TIMER COUNT
;BRANCH IF MORE TIME
;DONE MAY OR MAY NOT BE SET SO DONT TEST FOR IT
;TEST REST OF THE CSR
;A CSR ERROR
;R1 = CONTENTS OF CSR
;CHECK RBUF FOR CHAR PRESENT
;BRANCH IF CHAR PRESENT
;CHAR PRESENT MISSING
;R1 = CONTENTS OF RBUF
;CHECK THE DATA
;BRANCH IF OK
;RECEIVER ERROR
;R1 = CNTENTS OF RBUF
;CHECK RBUF FOR SECOND CHAR
;BRANCH IF CHAR PRESENT
;CHAR PRESENT MISSING
;R1 = CONTENTS OF RBUF
;CHECK THE DATA
;BRANCH IF OK
;RECEIVER ERROR
;R1 = CNTENTS OF RBUF
;CHECK FOR NO MORE CHARACTERS
;BRANCH IF CHAR PRESENT CLEARED
;CHAR PRESENT NOT CLEAR!
;R1 = CONTENTS OF RBUF
;SAVE CSR
;CHECK CSR
;BRANCH IF OK
;CSR ERROR
;R1 = CONTENTS OF CSR
;CLEAR TCR
;CLEAR CSR
  
```

2795
 2796
 2797
 2798
 2799
 2800
 2801
 2802
 2803
 2804
 2805 011262 004737 014600
 2806
 2807
 2808
 2809
 2810
 2811 011266 052777 000001 170030
 2812 011274 005737 001350
 2813 011300 001005
 2814 011302 004737 014644
 2815 011306 012737 000001 001350
 2816 011314 012777 000414 167776
 2817
 2818
 2819 011322 032777 000020 167770
 2820 011330 001374
 2821
 2822
 2823 011332 013702 001356
 2824 011336 017701 167756
 2825 011342 100375
 2826 011344 012777 000252 167754
 2827 011352 005302
 2828 011354 001370
 2829 011356 013700 001346
 2830 011362 105305
 2831 011364 001376
 2832 011366 017701 167726
 2833 011372 105701
 2834 011374 100002
 2835 011376 104001
 2836
 2837 011400 000402
 2838 011402 005300
 2839 011404 001366
 2840
 2841 011406 017701 167710
 2842 011412 100001
 2843 011414 104001
 2844
 2845 011416 005277 167676
 2846 011422 017701 167674
 2847 011426 100403
 2848 011430 105300
 2849 011432 001373
 2850 011434 104001

```

*****
:TEST 45: TEST THE SILO ALARM LEVEL AT WHICH DONE WILL SET.
          TEST THAT RECEIVER ENABLE ON A 0 INHIBITS
          CHARACTER PRESENT.
:PROBABLE FAULTY LOGIC: M7285 (D2-7) E32, E15; M7279 (D11-3) E19,E23
*****
          INITIALIZE
          DEVICE 'CSR' REGISTER
TST45: JSR PC,@#INITD SET:
          :BIT2 = MAINTENANCE
          :BIT3 = CLEAR MOS
          :BIT8 = MASTER XMTR SCAN ENB
:WAIT FOR MOS TO CLEAR
:
          SET:
          :BITC = RECEIVER ENABLE
          :TRANS CONTROL, LINE 0
          :HAS THE ALARM LEVEL BEEN TESTED?
          :YES
          :NO, GO DO IT
          :SET THE FLAG
          :BIT2 = MAINTENANCE
          :BIT3 = CLEAR MOS
          :BIT8 = MASTER TRAN SCAN ENB
          :WAIT FOR MOS TO CLEAR
          :
          :OUTPUT THE # OF CHARACTERS NECESSARY FOR THE SILO ALARM
          :LEVEL TO ALLOW DONE TO SET.
          MOV COUNT,R2 :SET CHAR COUNT
          1$: MOV @CSR, R1 :WAIT FOR XMTR READY
          BPL 1$
          MOV #252, @TBUF :SEND AN '*'
          DEC R2
          BNE 1$
          MOV TIMER, R0 :SET UP TIMER
          2$: DECB R5 :SHORT WAIT LOOP
          BNE 2$
          MOV @CSR, R1 :SAVE CSR FOR TYPING
          TSTB R1 :CHECK FOR DONE
          BPL 3$ :BRANCH IF NOT SET
          HLT+1 :DONE SET WHEN RCV ENB CLR
          :R1=CONTENTS OF CSR
          3$: BR 4$
          DEC R0 :TIMER COUNT
          BNE 2$ :BRANCH IF MORE TIMER
          4$: MOV @RBUF, R1 :CHECK AND SAVE FI/FO
          BPL .+4 :BRANCH IF OK
          HLT+1 :CHARACTER PRESENT IN FI/FO
          :R1=DATA FROM FI/FO
          5$: INC @CSR :SET RECEIVER ENABLE
          MOV @RBUF, R1 :CHECK FOR CHARACTER PRESENT
          BML 6$ :BRANCH IF OK
          DECB R0 :SHORT TIMER
          BNE 5$
          HLT+1 :CHARACTER PRESENT MISSING
  
```

```

2851
2852 011436 005077 167662 6$: CLR @TCR ;R1 = CONTENTS OF RBUF
2853 ;CLR TRANS CONTROL REG
2854 011442 104400 SCOPE
2855
2856 ;*****
2857 :TEST 46: TEST THAT HALF DUPLEX (BIT1) DISABLES THE RECEIVER UARTS.
2858 :PROBABLE FAULTY LOGIC: M7285 (D2-4) E32, E17, E22, (D2-2) E5, E1
2859 :*****
2860
2861 011444 004737 014570 TST46: JSR PC, @#INITC ;INITIALIZE
2862 ;BIT1 = HALF DUPLEX
2863 ;BIT2 = MAINTENANCE
2864 ;BIT3 = CLEAR MOS
2865 ;BIT8 = MASTER TRAN SCAN ENB
2866 ;WAIT FOR MOS TO CLEAR
2867 ;BIT0 = RECEIVER ENABLE
2868 011450 012777 000001 167646 MOV #BIT0, @TCR ;SET XMTR CONTROL BIT, LINE0
2869 ;OUTPUT THE # OF CHARACTERS NECESSARY FOR THE SILO ALARM
2870 ;LEVEL TO ALLOW DONE TO SET.
2871 011456 013702 001356 MOV COUNT,R2 ;SET CHAR COUNT
2872 011462 017701 167632 1$: MOV @CSR, R1 ;WAIT FOR XMTR READY
2873 011466 100375 BPL 1$
2874 011470 012777 000252 167630 MOV #252, @TBUF ;SEND AN '*'
2875 011476 005302 DEC R2
2876 011500 001370 BNE 1$
2877 011502 013700 001346 MOV TIMER, R0 ;SET UP TIMER
2878 011506 105305 2$: DECB R5 ;SHORT WAIT LOOP
2879 011510 001376 BNE 2$
2880 011512 017701 167602 MOV @CSR, R1 ;SAVE CSR
2881 011516 105701 TSTB R1 ;CHECK FOR DONE
2882 011520 100002 BPL 3$ ;BRANCH IF NOT SET
2883 011522 104001 HLT+1 ;DONE SET WHEN HALF DUPLEX (BIT1) SET
2884 ;R1=CONTENTS OF CSR
2885 011524 000402 BR 4$
2886 011526 005300 3$: DEC R0 ;TIMER COUNT
2887 011530 001366 BNE 2$ ;BRANCH IF MORE TIMER
2888
2889 011532 017701 167564 4$: MOV @RBUF, R1 ;CHECK AND SAVE FI/FO
2890 011536 100001 BPL .+4 ;BRANCH IF OK
2891 011540 104001 HLT+1 ;CHARACTER PRESENT IN FI/FO
2892 ;R1=DATA FROM FI/FO
2893 011542 042777 000002 167550 BIC #BIT1, @CSR ;CLEAR HALF DUPLEX BIT
2894 011550 000240 NOP
2895 011552 000240 NOP
2896 011554 000240 NOP
2897 011556 000240 NOP
2898 011560 017701 167536 5$: MOV @RBUF, R1 ;CHECK FOR CHAR PRESENT
2899 011564 100001 RPL .+4 ;BRANCH IF CHAR NOT PRESENT
2900 011566 104001 HLT+1 ;CHAR PRESENT AFTER H/D CLEARED
2901 ;R1 = CONTENTS OF RBUF
2902 011570 005077 167530 CLR @TCR ;CLR TRANS CONTROL REG
2903
2904 011574 104400 SCOPE
2905
2906 ;*****
  
```

2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962

011576 012777 011736 167524
 011604 012777 000340 167520
 011612 012737 000010 017214
 011620 012737 000005 017226
 011626 005337 017214
 011632 013737 017214 017216
 011540 004737 014464

 011644 042737 000340 177776
 011652 053737 017216 177776
 011660 004737 014560

 011664 012777 000001 167432

 011672 013702 001356
 011676 017701 167416
 100375
 011704 012777 000025 167414
 011712 005302
 011714 001370
 011716 105717 167376
 011722 100375
 011724

 011724 023737 017216 017220
 011732 002407
 011734 000417

 011736

 011736 022626
 011740 023737 017216 017220

```

:TEST 47: TEST RECEIVER INTERRUPT LEVEL
:ERROR PRINT OUT IS AS FOLLOWS:::
:ADDR DJADR R0 R1
:
:ADDR= ADDRESS OF ERROR HLT
:DJADR= CSR ADDRESS OF DJ11 UNDER TEST
:R1= BR LEVEL FOR DJ11 TO ALLOW INTR
:R2= BR LEVEL WHERE ERROR OCCURED
:
:PROBABLE FAULTY LOGIC: M7821 WIRING, PROPLR PRIORITY CHIP
:*****
TST47: MOV #INTR2,@RCVVEC ; SET UP RCVR INTERRUPT VECTOR
MOV #340,@RCVLVL ; AT LEVEL 7
MOV #10,PRIOLO ; LOAD BR7+1 INTO BR TEST LEVEL
AGAIN2: MOV #5,CT ; LOADS COUNTER FOR 5 ROL -BR
DEC PRIOLO ; START TEST BR7, THEN BR6,BR5...ETC.
MOV PRIOLO,NOW ; LOAD PRE-ROL-ED DECIMAL PRIO
JSR PC, BRSET ; GO TO SUBROUTINE TO :::::
; LOAD BR INTR LEVEL OF DJ11
; (LEVEL)= DECIMAL #(0 TO 7)
; (MASK)= OCTAL #(0 TO 340)
; (0,40,100,140,200,240,300,340)
; (NOW)= OCTAL #(0 TO 340) AS
; MASK, BUT CURRENTLY TESTING
BIC #340,@MPS ; CLEAR PS LEVEL
BIS NOW,@MPS ; SET PS TO CURRENT TEST LEVEL
JSR PC, @#INITA ; SET:
; BIT2 = MAINTENANCE
; BIT3 = CLEAR MOS
; BIT6 = RECEIVER INTERRUPT ENABLE
; BIT8 = MASTER TRANS SCAN ENABLE
; WAIT FOR MOS TO CLEAR
; BIT0 = RECEIVER ENABLE
; SET TRAN CONTROL BIT, LINE 0
MOV #BIT0, @TCR
:OUTPUT THE # OF CHARACTERS NECESSARY FOR THE SILO ALARM
:LEVEL TO ALLOW DONE TO SET.
1$: MOV COUNT,R2 ; SET CHAR COUNT
MOV @CSR, R1 ; WAIT FOR TRAN RDY
BPL 1$
MOV #25, @TBUF ;SEND #25
DEC R2
BNE 1$
2$: TSTB @CSR ;WAIT FOR DONE
BPL 2$
NINTR2:
; IF YOU ARE HERE NO INTR OCCURED
:
CMP NOW,MASK ; IS THIS AN ERROR????
BLT ERROR2 ; BRANCH IF YES
BR THRU2 ; IF NO,GO ON NEXT BR
INTR2:
; IF YOU ARE HERE AN INTR OCCURED
CMP (SP)+,(SP)+ ; CLEAN UP THE STACK
CMP NOW,MASK ; IS THIS AN ERROR????
  
```

```

2963 011746 002001      BGE      ERROR2      ; BRANCH IF ERROR
2964 011750 000411      BR       THRU2       ; IF NO,SEE IF ALL BR LEVELS TESTED
2965
2966 011752      ERROR2: ; IF YOU ARE HERE , THEN SOMETHING IS WRONG
2967                ;
2968                ; ERROR!.....!
2969 011752 010146      MOV      R1,-(6)     ;PUSH R1 ON STACK
2970 011754 010246      MOV      R2,-(6)     ;PUSH R2 ON STACK
2971
2972 011756 013701 017220  MOV MASK,      R1    ; R1=BR LEVEL ALLOWED
2973                ; (BR BITS OF PSW)
2974 011762 013702 017216  MOV NOW,      R2    ; R2=BR LEVEL OF ERROR
2975                ; (BR BITS OF PSW)
2976
2977 011766 104002      HLT+2          ; REPORT ERROR+ REG. 0,1
2978
2979
2980 011770 012602      MOV      (6)+,R2    ;POP STACK INTO R2
2981 011772 012601      MOV      (6)+,R1    ;POP STACK INTO R1
2982
2983 011774      THRU2:
2984
2985 011774 017701 167322  MOV      @RBUF, R1   ; READ THE CHARACTER
2986 012000 100401      BMI     .+4        ; BRANCH IF CHAR PRESENT
2987 012002 104001      HLT+1          ; CHAR PRESENT MISSING
2988                ; R1= CONTENTS OF RBUF
2989 012004 022701 100025  CMP      #100025,R1 ; CHECK THE DATA
2990 012010 001401      BEQ     .+4        ; BRANCH IF OK
2991 012012 104001      HLT+1          ; RECEIVED DATA ERROR
2992                ; R1= CONTENTS OF RBUF
2993
2994 012014 005077 167304  CLR     @TCR
2995 012020 005077 167274  CLR     @CSR
2996 012024 005737 017216  TST     NOW
2997 012030 001273      BNE     AGAIN2     ; TEST IF LAST BR WAS CHECKED
2998                ; BACK TO TEST NEXT INTR LOWER
2999 012032 013777 001332 167270  MOV     RCVLVL,@RCVVEC ; (NOW) = 0
3000 012040 012777 000004 167270  MOV     #IOT, @XMTLVL ; RELOAD AND CLEAN UP
3001 012046 042737 000340 177776  BIC     #340,@MPS
3002
3003
3004
3005 012054 104400      SCOPE
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017 012056 004737 014600  TST50: JSR     PC, @#INITD SET:
3018                ;BIT2 = MAINTENANCE
                ;BIT3 = CLEAR MOS
  
```

```

*****
:TEST 50:      TEST FI/FO OVERRUN
:              THE FI/FO BUFFER SHOULD HOLD 64 CHARACTERS.
:PROBABLE FAULTY LOGIC: M7285 (D1-7) E32, E17, E22 (D2-2) E5, E1
*****
  
```

```

:
:              INITIALIZE
:              DEVICE 'CSR' REGISTER
:
:              SET:
TST50: JSR     PC, @#INITD ;BIT2 = MAINTENANCE
                ;BIT3 = CLEAR MOS
  
```

```

3019                                     ;BIT8 = MASTER XMTR SCAN ENB
3020                                     ;WAIT FOR MOS TO CLEAR
3021                                     ;SET:
3022                                     ;BIT0 = RECEIVER ENABLE
3023                                     ;
3024 012062 012777 177777 167234      MOV      #177777,@TCR      ;TRANS CONTROL BIT, ALL LINES
3025 012070 012700 000100              MOV      #100, R0        ;SET UP COUNTER = 64, CHAR FI/FO BUFF
3026 012074 017701 167220      1$:     MOV      @CSR, R1      ;SAVE AND WAIT FOR TRANS READY
3027 012100 100375                  BPL      1$
3028 012102 000377 167220      SWAB    @TBUF          ;TRANSMIT LINE # ON LINE
3029 012106 032701 020000      BIT      #BIT13, R1    ;CHECK FI/FO OVERRUN
3030 012112 001401                  BEQ      .+4            ;BRANCH IF OK
3031 012114 104001                  HLT+1                ;FI/FO OVERRUN TOO SOON
3032                                     ;R1=CONTENTS OF CSR
3033 012116 005300                  DEC      R0            ;COUNT DOWN
3034 012120 001365                  BNE      1$
3035                                     ;
3036 012122 013700 001346      2$:     MOV      TIMER, R0    ;SET UP TIMER
3037 012126 017701 167166      MOV      @CCR, R1      ;WAIT FOR XMTR READY
3038 012132 100375                  BPL      2$
3039 012134 105305      3$:     DECB    R5            ;SHORT WAIT LOOP
3040 012136 001376                  BNE      3$
3041 012140 017701 167154      MOV      @CSR, R1      ;SAVE CSR FOR THE RECORD
3042 012144 100401                  BMI      .+4            ;BRANCH IF TRANS READY
3043 012146 104001                  HLT+1                ;TRANS READY MISTERIOUSLY DISAPPEARED
3044                                     ;R1=CONTENTS OF CSR
3045 012150 005300                  DEC      R0            ;TIME 3 CHARACTER LENGTHS
3046 012152 001370                  BNE      3$            ;BRANCH IF MORE TIME
3047                                     ;
3048                                     ;FI/FO SHOULD NOW BE FULL
3049                                     ;
3050 012154 105701                  TSTB    R1            ;CHECK THAT DONE IS SET
3051 012156 100401                  BMI      .+4            ;BRANCH IF OK
3052 012160 104001                  HLT+1                ;DONE DIDN'T COME UP!
3053                                     ;R1 = CONTENTS OF CSR
3054 012162 022701 100605      CMP      #100605,R1    ;CHECK THAT FI/FO NOT OVERRUN
3055 012166 001401                  BEQ      .+4            ;BRANCH IF OK
3056 012170 104001                  HLT+1                ;FI/FO OVERRUN SET
3057                                     ;OR SOME OTHER CSR ERROR
3058                                     ;R1=CONTENTS OF CSR
3059                                     ;
3060                                     ;*****
3061                                     ;TEST 50A: TEST THAT FI/FO OVERRUN COMES UP WHEN 65TH CHARACTER
3062                                     ;IS RECEIVED WITHOUT READING FI/FO
3063                                     ;*****
3064                                     ;
3065 012172 000377 167130      T50A:  SWAB    @TBUF          ;SEND 65TH CHARACTER
3066 012176 013700 001346      MOV      TIMER, R0    ;SET UP TIMER
3067 012202 105305      11$:   DECB    R5            ;SHORT WAIT LOOP
3068 012204 001376                  BNE      11$
3069 012206 017701 167106      MOV      @CSR,R1      ;SAVE CSR
3070 012212 032701 020000      BIT      #BIT13,R1    ;CHECK FI/FO OVERRUN
3071 012216 001003                  BNE      12$            ;BRANCH WHEN SET
3072 012220 005300                  DEC      R0            ;TIMER
3073 012222 001367                  BNE      11$            ;BRANCH IF MORE TIME
3074 012224 104001                  HLT+1                ;FI/FO OVERRUN DIDN'T COME UP

```



```

3075
3076 012226 022701 120605      12$:  CMP      #120605,R1      ;R1 = CONTENTS OF CSR
3077 012232 001401              BEQ      .+4                ;CHECK TOTAL CSR
3078 012234 104001              HLT+1                ;BRANCH IF OK
3079                                     ;SOMETHING IN CSR FOULED UP
3080                                     ;R1=CONTENTS OF CSR
3081
3082 :*****
3083 :TEST 50B:  TEST THAT READING THE RECEIVER BUFFER CAUSES FI/FO
3084 :          NOTE:  BECAUSE OF TIMING OF THE FI/FO, FI/FO OVERRUN CAN COME
3085 :                  BACK UP AFTER READING ONE CHARACTER, SO A SECOND MUST
3086 :                  BE READ TO INSURE THAT FI/FO OVERRUN IS CLEAR.
3087 :*****
3088
3089 012236 012700 000002      T50B:  MOV      #2,      R0      ;SET UP COUNTER - 2 CHARACTERS
3090 012242 017701 167054      21$:  MOV      @RBUF, R1      ;CHECK AND SAVE FIRST CHAR IN FI/FO
3091 012246 100401              BMI      .+4                ;BRANCH IF CHAR PRESENT
3092 012250 104001              HLT+1                ;CHARACTER PRESENT GONE!
3093
3094 012252 032701 070000              BIT      #070000,R1      ;CHECK RECEIVER ERRORS
3095 012256 001401              BEQ      .+4                ;BRANCH IF OK
3096 012260 104001              HLT+1                ;RECEIVER ERROR
3097                                     ;R1=CONTENTS OF RBUF
3098                                     ;BIT14=UART OVERRUN
3099                                     ;BIT13=FRAMMING ERROR
3100                                     ;BIT12=PARITY ERROR
3101
3102 012262 010102              MOV      R1,      R2      ;PUT LINE # IN R2
3103 012264 000302              SWAB    R2
3104 012266 042702 177760      BIC      #177760,R2
3105 012272 120102              CMPB    R1,      R2      ;CHECK DATA (=LINE#)
3106 012274 001401              BEQ      .+4                ;BRANCH IF OK
3107 012276 104001              HLT+1                ;WRONG DATA RECEIVED
3108                                     ;R1=FI/FO DATA
3109                                     ;(DATA SHOULD=LINE#)
3110 012300 005300      22$:  DEC      R0      ;COUNT CHARACTERS
3111 012302 001403              BEQ      24$            ;BRANCH WHEN DONE
3112 012304 105305      23$:  DECB   R5      ;SHORT WAIT LOOP - GIVE FI/FO TIME
3113 012306 001376              BNE      23$
3114 012310 000754              BR       21$            ;GO READ ANOTHER
3115
3116 012312 017701 167002      24$:  MOV      @CSR,   R1      ;SAVE CSR
3117 012316 022701 100605      CMP      #100605,R1      ;CHECK THAT FI/FO OVERRUN CLEARED
3118 012322 001401              BEQ      .+4                ;BRANCH IF OK
3119 012324 104001              HLT+1                ;FI/FO OVERRUN DIDN'T CLR
3120                                     ;OR SOMEOTHER CSR PROBLEM
3121                                     ;R1=CONTENTS OF CSR
3122
3123
3124 :*****
3125 :TEST 50C:  TEST THAT FI/FO OVERRUN INTERRUPT
3126 :          DOESN'T OCLUR WHEN THE PROCESSOR IS AT LEVEL 5
3127 :*****
3128
3129 012326 042737 000340 177776  T50C:  BIC      #340,  @#PS    ;CLEAR PSW
3130 012334 052737 000240 177776  BIS      #240,  @#PS    ;SET PROCESSOR TO LEVEL 5

```

```

3131 012342 012777 012432 166760      MOV    #32$, @RCVVEC ;SET UP RECEIVER INTERRUPT VEC
3132 012350 012777 000340 166754      MOV    #340, @RCVLVL
3133 012356 052777 010000 166734      BIS    #BIT12, @CSR ;SET STATUS ENABLE
3134 012364 017701 166730      MOV    @CSR, R1 ;SAVE CSR
3135 012370 022701 110605      CMP    #110605,R1 ;CHECK CSR
3136 012374 001401      BEQ    .+4
3137 012376 104001      HLT+1 ;CSR ERROR
3138 ;R1=CONTENTS OF CSR
3139 012400 000377 166722      SWAB  @TBUF ;SEND LINE #
3140 012404 005777 166710      30$: TST  @CSR ;WAIT FOR TRANSMITTER READY
3141 012410 100375      BPL    30$
3142 012412 000377 156710      SWAB  @TBUF ;SEND LINE #
3143 012416 017701 166676      31$: MOV  @CSR, R1 ;SAVE CSR
3144 012422 032701 020000      BIT   #BIT13, R1 ;WAIT FOR FI/FO OVERRUN
3145 012426 001773      BEQ   31$
3146 012430 000410      BR    33$ ;SKIP ISR
3147
3148 012432 017701 166662      32$: MOV  @CSR, R1 ;SAVE CSR
3149 012436 104001      HLT+1 ;INTERRUPT OCCURRED AT LEVEL 5
3150 012440 005777 166656      TST  @RBUF ;'POP' ONE CHARACTER
3151 012444 012716 012456      MOV  #34$, (SP) ;RESET RETURN ADDRESS
3152 012450 000002      RTI   ;RETURN
3153
3154 012452 012700 000002      33$: MOV  #2, R0 ;SET UP COUNTER - 2 CHARACTERS
3155 012456 017701 166640      34$: MOV  @RBUF, R1 ;READ ONE CHARACTER
3156 012462 100401      BMI  .+4 ;BRANCH IF CHARACTER PRESENT
3157 012464 104001      HLT+1
3158
3159
3160 012466 032701 070000      BIT   #70000, R1 ;CHECK ERRORS
3161 012472 001401      BEQ   .+4
3162 012474 104001      HLT+1
3163
3164 012476 010102      MOV   R1, R2 ;DUP DATA
3165 012500 000302      SWAB  R2
3166 012502 042702 177760      BIC   #177760,R2 ;CLR ALL BUT LINE #
3167 012506 120102      CMPB  R1, R2 ;CHECK DATA
3168 012510 001401      BEQ   .+4
3169 012512 104001      HLT+1
3170
3171 012514 005300      DEC   R0 ;COUNT CHARACTERS
3172 012516 003403      BLE  36$ ;BRANCH IF DONE
3173 012520 105305      35$: DECB R5 ;SHORT WAIT LOOP
3174 012522 001376      BNE  35$
3175 012524 000754      BR   34$ ;GO READ ANOTHER CHARACTER
3176
3177 012526 017701 166566      36$: MOV  @CSR, R1 ;SAVE CSR
3178 012532 022701 110605      CMP  #110605,R1 ;CHECK THAT FI/FO OVERRUN CLEARED
3179 012536 001401      BEQ  .+4 ;BRANCH IF OK
3180 012540 104001      HLT+1 ;FI/FO OVERRUN DIDN'T CLR
3181 ;OR SOMEOTHER CSR PROBLEM
3182 ;R1=CONTENTS OF CSR
3183

```

```

;*****
;TEST 50D: TEST THAT FI/FO OVERRUN INTERRUPT
; OCCURS WHEN PROCESSOR IS AT LEVEL 4
;

```

```

3187
3188
3189 012542 042737 000340 177776 T50D: BIC #340, @#PS ;CLEAR PROCESSOR LEVEL
3190 012550 052737 000200 177776 BIS #200, @#PS ;SET PROCESSOR TO LEVEL 4
3191 012556 012777 012642 166544 MOV #42$, @RCVVEC ;SET UP RECEIVER INTERRUPT VEC
3192 012564 017701 166530 MOV @CSR, R1 ;SAVE CSR
3193 012570 022701 110605 CMP #110605, R1 ;CHECK CSR
3194 012574 001401 BEQ .+4
3195 012576 104001 HLT+1
3196
3197 012600 000377 166522 SWAB @TBUF ;SEND LINE #
3198 012604 005777 166510 40$: TST @CSR ;WAIT FOR TRANSMITTER READY
3199 012610 100375 BPL 40$
3200 012612 000377 166510 SWAB @TBUF ;SEND LINE #
3201 012616 017701 166476 41$: MOV @CSR, R1 ;SAVE CSR
3202 012622 032701 020000 BIT #BIT13, R1 ;WAIT FOR FI/FO OVERRUN
3203 012626 001773 BEQ 41$
3204 012630 104001 HLT+1 ;INTERUPT DIDN'T OCCURE WHEN OVERRUN SET
3205 ;R1 = CONTENTS OF CSR
3206 012632 052777 000010 166460 BIS #BIT3, @CSR ;CLEAR MOS
3207 012640 000470 BR 45$ ;SKIP TO THE END
3208
3209 012642 017701 166452 42$: MOV @CSR, R1 ;SAVE CSR
3210 012646 022701 130605 CMP #130605, R1 ;CHECK CSR
3211 012652 001401 BEQ .+4 ;BRANCH IF OK
3212 012654 104001 HLT+1 ;CSR ERROR
3213 ;R1 = CONTENTS OF CSR
3214 012656 012700 000101 43$: MOV #101, RC ;SET UP COUNTER - 65 CHARACTERS
3215 012662 017701 166434 43$: MOV @RBUF, R1 ;READ ONE CHARACTER
3216 012666 100401 BMI .+4 ;BRANCH IF CHARACTER PRESENT
3217 012670 104001 HLT+1
3218
3219 012672 032701 070000 BIT #70000, R1 ;CHECK ERRORS
3220 012676 001401 BEQ .+4
3221 012700 104001 HLT+1
3222
3223 012702 010102 MOV R1, R2 ;DUP DATA
3224 012704 000302 SWAB R2
3225 012706 042702 177760 BIC #177760, R2 ;CLR ALL BUT LINE #
3226 012712 120102 CMPB R1, R2 ;CHECK DATA
3227 012714 001401 BEQ .+4
3228 012716 104001 HLT+1
3229
3230 012720 017701 166374 MOV @CSR, R1 ;SAVE CSR
3231 012724 005300 DEC R0
3232 012726 001420 BEQ 46$ ;GET OUT IF ALL CHAR'S READ
3233 012730 022700 000100 CMP #100, R0 ;CHECK FOR FIRST CHAR READ
3234 012734 001752 BEQ 43$ ;SKIP CSR CHECK ON FIRST CHAR
3235 012736 020037 001356 CMP R0, COUNT ;IF CHAR'S LEFT IN SILO IS LESS THEN
3236 ;ALARM LEVEL, DONE WILL GO AWAY.
3237 012742 002405 BLT 44$
3238 012744 022701 110605 CMP #110605, R1 ;CHECK CSR
3239 012750 001401 BEQ .+4
3240 012752 104001 HLT+1
3241
3242 012754 000742 BR 43$
  
```

```

3243
3244 012756 022701 110405 44$: CMP #110405,R1 ;CHECK CSR
3245 012762 001401 BEQ .+4 ;BRANCH IF OK
3246 012764 104001 HLT+1 ;DONE DIDN'T GO AWAY
3247 ;OR OTHER CSR ERROR
3248 012766 000735 BR 43$
3249
3250 012770 017701 166326 46$: MOV @RBUF, R1 ;READ A CHARACTER
3251 012774 100001 BPL .+4 ;BRANCH IF NO CHAR. PRES
3252 012776 104001 HLT+1 ;CHAR. PRESENT.
3253
3254 013000 013777 001332 166322 MOV RCVLVL, @RCVVEC
3255 013006 012777 000004 166316 MOV #IOT, @RCVLVL
3256 013014 012716 013022 MOV #45$, (SP) ;RESET RETURN ADDRESS ON STACK
3257 013020 000002 RTI ;RESTORE PSW
3258 013022 005077 166276 45$: CLR @TCR
3259 013026 005077 166266 CLR @CSR
3260 013032 042737 000340 177776 BIC #340, @MPS ;LOWER PROCESSOR STATUS
3261
3262 013040 104400 SCOPE
3263
3264
3265
3266 *****
3267 :TEST 51: TEST THAT UART OVERRUN IS DETECTED ON ALL LINES
3268 :PROBABLE FAULTY LOGIC: M7285 (D2-2) E3, E1; M7279; M7280
3269 *****
3270 013042 012777 000414 166250 TST51: MOV #414, @CSR ;BIT2 = MAINTENANCE
3271 ;BIT3 = CLEAR MOS
3272 ;BIT8 = TRANS SCAN ENABLE
3273 013050 032777 000020 166242 10$: BIT #BIT4, @CSR ;WAIT FOR MOS TO CLEAR
3274 013056 001374 BNE 10$
3275 013060 005001 CLR R1 ;SET UP LINE COUNTER
3276 013062 012777 000001 166234 MOV #1, @TCR ;TRANS CONTROL, LINE 0
3277 013070 017702 166274 LOP51: MOV @CSR, R2 ;WAIT FOR TRANS READY
3278 013074 100375 BPL LOP51
3279 013076 012777 000001 166222 MOV #1, @TBUF ;SEND #1
3280 013104 017702 166210 2$: MOV @CSR, R2 ;WAIT FOR TRANS READY
3281 013110 100375 BPL 2$
3282 013112 012777 000002 166206 MOV #2, @TBUF ;SEND #2
3283 013120 013700 001346 3$: MOV TIMER, R0 ;SET UP TIMER
3284 013124 105305 DECB R5 ;SHORT TIME LOOP
3285 013126 001376 BNE 3$
3286 013130 017702 166164 MOV @CSR, R2 ;SAVE CSR FOR THE RECORD
3287 013134 000400 BR .+2 ;NOP FOR TIMING
3288 013136 005300 DEC R0 ;TIMER COUNT
3289 013140 001371 BNE 3$
3290 013142 005277 166152 INC @CSR ;SET RECEIVER ENABLE
3291 013146 017702 166146 MOV @CSR, R2 ;SAVE CSR
3292 013152 032702 000001 BIT #BIT0, R2 ;CHECK RECEIVER ENABLE
3293 013156 001001 BNE .+4 ;BRANCH IF OK
3294 013160 104002 HLT+2 ;RECEIVER ENABLE FAILED TO SET
3295 ;R1 = LINE #
3296 ;R2 = CONTENTS OF CSR
3297 013162 013700 001346 4$: MOV TIMER,R0 ;SET UP TIMER
3298 013166 105305 DECB R5 ;WAIT SHORT TIME LOOP
  
```

```

3299 013170 001376 BNE 4$
3300 013172 017702 166124 MOV @RBUF,R2 ;CHECK FOR CHAR PRES
3301 013176 100403 BMI 5$ ;OK
3302 013200 005300 DEC R0 ;TIMER COUNT
3303 013202 001371 BNE 4$
3304 013204 104002 HLT+2 ;NO CHARACTER PRESENT
3305 ;R1 = LINE #
3306 ;R2 = CONTENTS OF RBUF
3307 013206 032702 040000 5$: BIT #BIT14,R2 ;CHECK FOR UART OVERRUN
3308 013212 001001 BNE .+4 ;BRANCH IF OK
3309 013214 104002 HLT+2 ;UART OVERRUN MISSING
3310 ;R1 = LINE #
3311 ;R2 = CONTENTS OF RBUF
3312 013216 122702 000002 CMPB #2,R2 ;CHECK THE DATA
3313 013222 001401 BEQ .+4 ;BRANCH IF OK
3314 013224 104002 HLT+2 ;DATA ERROR - 3RD CHAR OVERRUNS 2ND
3315 ;R1 = LINE #
3316 ;R2 = CONTENTS OF RBUF
3317 013226 010203 MOV R2,R3 ;DUP DATA
3318 013230 000303 SWAB R3
3319 013232 042703 177700 BIC #177700,R3 ;MASK ALL BIT LINE #, ERROR BITS
3320 013236 020103 CMP R1,R3 ;CHECK LINE #, ERRORS
3321 013240 001401 BEQ .+4 ;BRANCH IF OK
3322 013242 104002 HLT+2 ;LINE # OR OTHER RBUF ERROR
3323 ;R1 = LINE #
3324 ;R2 = CONTENTS OF RBUF
3325 013244 017702 166052 7$: MOV @RBUF,R2 ;CHECK FOR MORE DATA
3326 013250 100002 BPL 8$ ;BRANCH IF OK
3327 013252 104002 HLT+2 ;EXTRA DATA IN FI/FO!
3328 ;R1 = LINE #
3329 ;R2 = CONTENTS OF RBUF
3330 013254 000773 BR 7$
3331
3332 013256 005377 166036 8$: DEC @CSR ;CLEAR RECEIVER ENABLE
3333 013262 017702 166032 MOV @CSR,R2 ;SAVE CSR
3334 013266 022702 100404 CMP #100404,R2 ;CHECK CSR
3335 013272 001401 BEQ .+4 ;BRANCH IF OK
3336 013274 104002 HLT+2 ;CSR ERROR
3337 ;R1 = LINE NUMBER
3338 ;R2 = CONTENTS OF CSR
3339 013276 005201 INC R1 ;COUNT LINES
3340 013300 006377 166020 ASL @TCR ;GO TO NEXT LINE
3341 013304 103271 BCC LOP51 ;BRANCH BACK IF MORE LINES
3342
3343 013306 005077 166006 CLR @CSR
3344
3345 013312 104400 SCOPE
3346
3347
3348
3349 *****
3349 ;TEST 52: TEST BITS OF BCSR FOR READ/WRITE CAPABILITY
3350 ;PROBABLE FAULTY LOGIC: M7285 (D2-2) E5, E1, (D2-3) E16, E2, E19, E35
3351 *****
3352
3353 013314 012777 002010 165776 TST52: MOV #002010,@CSR ;SET CSR
3354 013322 012777 177777 165774 MOV #177777,@BCSR ;SET ALL BITS OF BCSR
  
```

```

3355 013330 017701 165770      MOV    @BCSR, R1      ;CHECK AND SAVE BCSR
3356 013334 022701 177777      CMP    #177777,R1    ;CHECK THAT ALL THE BITS ARE SET
3357 013340 001401              BEQ    .+4           ;BRANCH IF OK
3358 013342 104001              HLT+1              ;BIT(S) OF BCSR FAILED TO SET
3359
3360 013344 005077 165754      CLR    @BCSR        ;CLEAR BCSR
3361 013350 017701 165750      MOV    @BCSR, R1    ;CHECK THAT IT CLEARED AND SAVE
3362 013354 001401              BEQ    .+4           ;BRANCH IF CLR
3363 013356 104001              HLT+1              ;BIT(S) OF BCSR FAILED TO CLEAR
3364
3365 013360 104400              SCOPE
3366
3367
3368
3369
3370
3371
3372
3373
3374
3375 013362 004737 014550      TST53: JSR    PC,    @#INITA ;INITIALIZE
3376
3377
3378
3379
3380
3381 013366 012777 000001 165730      MOV    #1,    @BCSR ;BIT2 = MAINTENANCE
3382 013374 013700 001346      MOV    TIMER, R0    ;BIT3 = CLEAR MOS
3383 013400 105305      1$:   DECB    R5      ;BIT10= R/W BCSR
3384 013402 001376              BNE    1$           ;WAIT FOR MOS TO CLEAR
3385 013404 017701 165712      MOV    @RBUF, R1    ;BIT0 = RECEIVER ENABLE
3386 013410 100403              BMI    2$           ;SEND BREAKS, LINE 0
3387 013412 005200              INC    R0           ;SET UP TIMER
3388 013414 001371              BNE    1$           ;SHORT WAIT LOOP
3389 013416 104001              HLT+1              ;SAVE CHAR PRES
3390
3391
3392 013420 032701 020000      2$:   BIT    #020000,R1 ;CHECK FOR FRAMING ERROR
3393 013424 001001              BNE    .+4           ;BRANCH IF OK
3394 013426 104001              HLT+1              ;FRAMING ERROR NOT UP
3395
3396 013430 133737 001366 001300      BITB   DJPAR,  PARITY ;R1=CONTENTS OF RBUF
3397 013436 001404              BEQ    3$           ;CHECK ODD PARITY FLAG
3398 013440 032701 010000      BIT    #010000,R1    ;BRANCH IF NOT
3399 013444 001005              BNE    4$           ;CHECK PARITY ERROR
3400 013446 104001              HLT+1              ;
3401
3402
3403 013450 032701 010000      3$:   BIT    #010000,R1 ;CHECK PARITY ERROR
3404 013454 001401              BEQ    .+4           ;BRANCH IF OK
3405 013456 104001              HLT+1              ;EVEN PARITY OR NO PARITY
3406
3407
3408 013460 032701 040000      4$:   BIT    #040000,R1 ;SHOULDN'T CAUSE PARITY ERROR
3409 013464 001401              BEQ    .+4           ;R1=CONTENTS OF RBUF
3410 013466 104001              HLT+1              ;CHECK UART OVERRUN
3411
3412
3413
3414
3415
3416
3417
3418
3419
3420
3421
3422
3423
3424
3425
3426
3427
3428
3429
3430
3431
3432
3433
3434
3435
3436
3437
3438
3439
3440
3441
3442
3443
3444
3445
3446
3447
3448
3449
3450
3451
3452
3453
3454
3455
3456
3457
3458
3459
3460
3461
3462
3463
3464
3465
3466
3467
3468
3469
3470
3471
3472
3473
3474
3475
3476
3477
3478
3479
3480
3481
3482
3483
3484
3485
3486
3487
3488
3489
3490
3491
3492
3493
3494
3495
3496
3497
3498
3499
3500
  
```

```

3411
3412 013470 032701 007400      BIT      #007400,R1      ;R1=CONTENTS OF RBUF
3413 013474 001401              BEQ      .+4            ;CHECK LINE #
3414 013476 104001              HLT+1          ;BRANCH IF OK
3415                                ;WRONG LINE# IN FI/FO
3416 013500 105701              TSTB     R1          ;R1=CONTENTS OF RBUF
3417 013502 001401              BEQ      .+4            ;CHECK DATA
3418 013504 104001              HLT+1          ;BRANCH IF OK
3419                                ;WRONG DATA RECEIVED
3420                                ;R1=CONTENTS OF RBUF
3421 013506 017701 165610      MOV      @RBUF, R1    ;READ FI/FO AGAIN
3422 013512 100001              BPL      .+4            ;BRANCH IF OK
3423 013514 104001              HLT+1          ;EXTRA CHAR IN FI/FO
3424                                ;R1 = CONTENTS OF RBUF
3425 013516 005077 165602      CLR      @BCSR        ;CLEAR BREAK CONTROL REG
3426 013522 105305              5$:      DECB     R5          ;SHORT WAIT LOOP-REGISTER A MARK
3427 013524 001376              BNE      5$
3428
3429 013526 104400              SCOPE
3430
3431
3432
3433
3434
3435
3436
3437
3438
3439
3440 013530 004737 014550      TST54:  JSR      PC,      @#INITA ;INITIALIZE
3441                                ;BIT2 = MAINTENANCE
3442                                ;BIT3 = CLEAR MOS
3443                                ;BIT10= R/W BCSR
3444                                ;WAIT FOR MOS TO CLEAR
3445                                ;BIT0 = RECEIVER ENABLE
3446 013534 005001              CLR      R1          ;SET UP LINE COUNTER
3447 013536 012704 000001      MOV      #1,      R4          ;SET UP LINE MARKER
3448 013542 013700 001346      LOP54:  MOV      TIMER, R0      ;SET UP TIMER
3449 013546 050477 165552      BIS      R4,      @BCSR      ;SET BREAK CONTROL BIT, LINE # IN R1
3450 013552 105305              1$:      DECB     R5          ;SHORT WAIT LOOP
3451 013554 001376              BNE      1$
3452 013556 017702 165540      MOV      @RBUF, R2      ;READ RBUF FOR CHAR PRES
3453 013562 100403              BMI      7$            ;BRANCH WHEN FOUND
3454 013564 005200              INC      R0            ;WAIT A WHILE
3455 013566 001371              BNE      1$
3456 013570 104002              HLT+2          ;CHAR PRES NEVER CAME UP
3457                                ;R1 = LINE #
3458                                ;R2 = CONTENTS OF RBUF
3459
3460 013572 032702 020000      2$:      BIT      #020000,R2      ;CHECK FOR FRAMING ERROR
3461 013576 001001              BNE      .+4            ;BRANCH IF OK
3462 013600 104002              HLT+2          ;FRAMING ERROR NOT UP
3463                                ;R1 = LINE #
3464                                ;R2 = CONTENTS OF RBUF
3465
3466 013602 010103              MOV      R1,      R3      ;GET LINE #
    
```

```

*****
:TEST 54:      TEST THAT EACH LINE CAN TRANSMIT AND RECEIVE A BREAK
:              ALSO CHECKS FRAMING ERROR(RBUF BIT13)
:              ALSO CHECKS PARITY ERROR(RBUF BIT12)
:              IF ODD PARITY IS SELECTED
:PROBABLE FAULTY LOGIC: M7285 (D2-2) E5, E1, (D2-3) E16, E2, E19 E35
*****
    
```

```

3467 013604 006203          ASR      R3          ;/4 TO GET RID OF BITS 1 & 0
3468 013606 006203          ASR      R3
3469 013610 006303          ASL      R3          ;SHIFT BACK FOR EVEN ADDR OF PARITY TABLE
3470 013612 133763 001366 001300 BITB     DJPAR,PARITY(3) ;CHECK ODD PARITY FLAG
3471 013620 001404          BEQ      3$          ;BRANCH IF NOT
3472 013622 032702 010000   BIT      #010000,R2   ;CHECK PARITY ERROR
3473 013626 001005          BNE      4$          ;
3474 013630 104002          HLT+2          ;ODD PARITY SHOULD CAUSE PARITY ERROR
3475
3476
3477
3478 013632 032702 010000   3$:     BIT      #010000,R2   ;CHECK PARITY ERROR
3479 013636 001401          BEQ      4$          ;BRANCH IF OK
3480 013640 104002          HLT+2          ;EVEN PARITY OR NO PARITY
3481
3482
3483
3484 013642 032702 040000   4$:     BIT      #040000,R2   ;CHECK UART OVERRUN
3485 013646 001401          BEQ      .+4        ;BRANCH IF OK
3486 013650 104002          HLT+2          ;UART OVERRUN SET!.
3487
3488
3489 013652 010203          MOV      R2, R3     ;R2 = CONTENTS OF RBUF
3490 013654 000303          SWAB    R3          ;DUP DATA
3491 013656 042703 177760   BIC     #177760,R3   ;LINE # IN LOW BYTE
3492 013662 020103          CMP     R1, R3     ;MASK ALL BUT LINE #
3493 013664 001401          BEQ     .+4        ;CHECK FOR RIGHT LINE #
3494 013666 104002          HLT+2          ;BRANCH IF OK
3495
3496
3497
3498
3499 013670 105702          TSTB   R2          ;WRONG LINE# IN FI/FO
3500 013672 001401          BEQ     .+4        ;R1 = LINE #
3501 013674 104002          HLT+2          ;R2 = CONTENTS OF RBUF
3502
3503
3504 013676 017702 165420   MOV     @RBUF, R2   ;READ FI/FO AGAIN
3505 013702 100001          BPL     .+4        ;BRANCH IF OK
3506 013704 104002          HLT+2          ;EXTRA CHAR IN FI/FO
3507
3508
3509 013706 005077 165412   CLR     @BCSR       ;R1 = LINE #
3510 013712 005201          INC     R1          ;R2 = CONTENTS OF RBUF
3511 013714 006304          ASL     R4          ;CLEAR BREAK CONTROL REG
3512 013716 103311          BCC     LOP54      ;COUNT LINES
3513
3514 013720 005077 165374   CLR     @CSR        ;UPDATE LINE MARKER
3515 013724 104400          SCOPE          ;BRANCH IF MORE LINES
3516
3517 013726 012737 000002 015252 MOV     #2, TIMES   ;CLEAR CSR
3518
3519
3520
3521
3522
*****
:TEST 55: TEST THAT RESET CLEARS ALL BUFFERS
:NOTE: THE FI/FO BUFFER IS NOT COMPLETELY CLEARED
: BY RESET; ONLY CHARACTER PRESENT IS CLEARED.

```



```

3523 ;PROBABLE FAULTY LOGIC. M7285 (D2-8) E13
3524 ;*****
3525
3526 013734 052737 000340 177776 TST55: BIS #340,@#PS ;SET PROCESSOR TO LEVEL 7
3527 013742 012777 177777 165354 MOV #177777,@TCR ;SET ALL TCR BITS
3528 013750 012777 052507 165342 MOV #052507,@CSR ;SET ALL R/W BITS OF CSR
3529 013756 012777 177777 165340 MOV #177777,@BCSR ;SET ALL BREAK CONTROL BITS (SEND BREAKS)
3530 ;NOTE: ALL LINES SHOULD BE SENDING BREAKS, BUT NONE SHOULD BE RECEIVING
3531 ; BECAUSE THE HALF DUPLEX BIT IS SET
3532 013764 012777 177777 165334 MOV #177777,@TBUF ;LOADING TRANS BUFF WHEN BREAK BIT SET
3533 ; SHOULD DO NOTHING
3534 013772 000005 RESET ;CLEAR THE WORLD
3535 013774 013737 001320 014004 MOV CSR,1$ ;CHECK CSR AND SAVE
3536 014002 013701 MOV @PC)+,R1
3537 014004 000000 1$: 000000
3538 014006 001401 BEQ .+4
3539 014010 104001 HLT+1
3540
3541 014012 017701 165306 MOV @TCR,R1 ;CHECK TCR AND SAVE
3542 014016 001401 BEQ .+4
3543 014020 104001 HLT+1
3544
3545 014022 017701 165276 MOV @BCSR,R1 ;CHECK BCSR AND SAVE
3546 014026 001401 BEQ .+4
3547 014030 104001 HLT+1
3548
3549 014032 017701 165264 MOV @RBUF,R1 ;CHECK RBUF AND SAVE
3550 014036 100001 BPL .+4
3551 014040 104001 HLT+1
3552
3553 014042 017701 165260 MOV @TBUF,R1 ;CHECK TBUF AND SAVE
3554 014046 001401 BEQ .+4
3555 014050 104001 HLT+1
3556
3557 014052 104400 SCOPE
3558
3559
3560 ;*****
3561 ;TEST 56: SEND A BINARY COUNT PATTERN ON EACH LINE
3562 ;PROBABLE FAULTY LOGIC: COULD BE ALMOST ANYWHERE!
3563 ;*****
3564
3565 014054 005001 TST56: CLR R1 ;SET UP LINE COUNTER
3566 014056 012703 100000 MOV #100000,R3 ;SET UP RCV DATA
3567 014062 012777 014260 165240 MOV #ISR56,@RCVVEC ;SET UP RECEIVER INTERRUPT VECTOR
3568 014070 012777 000240 165234 MOV #240,@RCVLVL
3569 014076 042737 000340 177776 BIC #340,@#PS ;CLEAR PROCESSOR PRIORITY
3570 014104 052737 000200 177776 BIS #200,@#PS ;SET PRIORITY TO 4
3571 014112 012700 000001 MOV #1,R0 ;SET UP LINE MARKER
3572 014116 012777 010514 165174 MOV #010514,@CSR ;BIT2 = MAINTENANCE
3573 ;BIT3 = CLR MOS
3574 ;BIT6 = RECEIVER INTERRUPT ENB
3575 ;BIT8 = TRANS SCAN ENABLE
3576 ;BIT12= STATUS ENABLE
3577 014124 032777 000020 165166 10$: BIT #BIT4,@CSR ;WAIT FOR MOS TO CLEAR
3578 014132 001374 BNE 10$
  
```

3579	014134	052777	000001	165156		BIS	#1,@CSR		:SET THE RCV EN BIT
3580	014142	005004			1\$:	CLR	R4		:SET FOR OUTPUT DATA
3581	014144	010077	165154			MOV	R0,@TCR		:TRANS CONTROL, ONE LINE AT A TIME
3582	014150	005777	165144		2\$:	TST	@CSR		:WAIT FOR TRANS READY
3583	014154	100375				BPL	2\$		
3584	014156	010477	165144			MOV	R4,@TBUF		:SEND DATA
3585	014162	105204				INCB	R4		:BINARY COUNT
3586	014164	123704	001360			CMPB	SUM,R4		:MAKE SURE THE CORECT NUMBER IS OUTPUTED
3587	014170	001367				BNE	2\$		
3588	014172	147704	165166		3\$:	BICB	@DJLEN,R4		:SET FOR MAX. RECIEVER COUNT
3589	014176	120403				CMPB	R4,R3		:WAIT FOR RECEIVER DONE
3590	014200	001374				BNE	3\$		
3591	014202	017702	165112			MOV	@CSR,R2		:SAVE CSR
3592	014206	022702	110505			CMP	#1'0505,R2		:CHECK CSR
3593	014212	001401				BFO	.+4		:BRANCH IF CK
3594	014214	104002				HLT+2			:CSR ERROR
3595									:R1=LINE #
3596									:R2=CONTENTS OF CSR
3597	014216	017702	165100			MOV	@RBUF,R2		:CHECK CHARACTER PRESENT
3598	014222	100001				BPL	.+4		:BRANCH IF OK
3599	014224	104002				HLT+2			:CHARACTER PRESENT SET!!
3600									:R1=LINE #
3601									:R2=CONTENTS OF CSR
3602	014226	105003				CLRB	R3		:RESET EXPECTED DATA
3603	014230	062703	000400			ADD	#400,R3		:UPDATE LINE # IN EXPECTED DATA
3604	014234	005201				INC	R1		:UPDATE LINE #
3605	014236	032701	000003			BIT	#3,R1		:CHECK FOR FOURTH LINE
3606	014242	001002				BNE	4\$:BRANCH IF NOT
3607	014244	005237	001364			INC	DJLEN		:MOVE CHARACTER LENGTH POINTER
3608	014250	000241			4\$:	CLC			
3609	014252	006300				ASL	R0		:UPDATE LINE MARKER
3610	014254	103332				BCC	1\$:BRANCH IF MORE
3611	014256	000416				BR	END56		:SKIP ISR
3612									
3613	014260	017702	165036		ISR56:	MOV	@RBUF,R2		:READ FIRST DATA
3614	014264	100401				BMI	11\$:BRANCH IF CHARACTER PRESENT
3615	014266	104003				HLT+3			:INTERRUPT BUT NO CHAR PRESENT
3616									:R1=LINE #
3617									:R2=CONTENTS OF RBUF
3618									:R3=EXPECTED DATA
3619	014270	147703	165070		11\$:	BICB	@DJLEN,R3		:MASK CHAR. LENGTH
3620	014274	020203				CMP	R2,R3		:CHECK THE DATA
3621	014276	001401				BEQ	.+4		:BRANCH IF OK
3622	014300	104003				HLT+3			:DATA ERROR
3623									:R1=LINE #
3624									:R2=CONTENTS OF RBUF
3625									:R3=EXPECTED DATA
3626	014302	105203				INCB	R3		:UPDATE EXPECTED DATA
3627	014304	017702	165012			MOV	@RBUF,R2		:READ MORE DATA
3628	014310	100767				BMI	11\$:BRANCH IF MORE
3629	014312	000002			12\$:	RTI			:RETURN
3630									
3631	014314	162737	000004	001364	END56:	SUB	#4,DJLEN		:RESET CHAR LENGTH POINTER
3632	014322	013777	001332	165000		MOV	RCV LVL,@RCVVEC		:RESTORE RECEIVER INT. VEC
3633	014330	012777	000004	164774		MOV	#IOT,@RCV LVL		
3634	014336	005077	164762			CLR	@TCR		:CLEAR TCR

```

3635 014342 005077 164752 CLR @CSR ;CLEAR CSR
3636 014346 104400 SCOPE
3637
3638 014350 012737 000020 01:252 MOV #20, TIMES
3639 014356 023737 001362 00:344 CMP DJUUT,UNITS ;CHECK FOR LAST UNIT
3640 014364 002004 BGE DONE ;BRANCH IF LAST UNIT
3641 014366 005037 001350 CLR ALMFLG ;CLEAR FLAG FOR NEXT UNIT
3642 014372 000137 002472 JMP RESTAR ;JUMP IF NOT
3643
3644 014376 DONE:
3645 014376 004737 017454 JSR PC, KBDINT
3646 014402 062737 000001 001316 ADD #1,PCNT+2 ;ADD 1 TO THE PASS COUNT
3647 014410 005537 001314 ADC PCNT ;MAKE IT DOUBLE PREC.
3648 014414 000004 016516 TYPE ,MEOP ;END OF PASS INDICATOR
3649 014420 032777 002000 164746 BIT #SW10,@SWR ;RING THE BELL?
3650 014426 001004 BNE 4$ ;NO!
3651 014430 000004 000007 TYPE ,BELL ;RING THE BELL
3652 014434 000004 000177 TYPE ,177 ;TYPE A FILLER FOR 11/05
3653 014440 013700 000042 4$: MOV @42,R0 ;GET MONITOR ADDRESS
3654 014444 001405 BEQ 3$ ;IF NONE
3655 014446 000005 RESET ;RESET AND
3656 014450 $ENDAD -
3657 014450 004710 JSR 7,(0) ;GO TO MONITOR
3658 014452 000240 NOP ;SAVE ROOM
3659 014454 000240 NOP ;FOR
3660 014456 000240 NOP ;ACT11
3661 014460 000137 002472 3$: JMP RESTAR ;RETURN
3662
3663
3664
3665
3666
3667 014464 BRSET:
3668 014464 010046 MOV R0, -(SP) ; SAVE R0
3669 014466 010146 MOV R1, -(SP) ; SAVE R1
3670 014470 013700 001362 MOV DJUUT, R0 ; LOAD UP UUT DEVICE #
3671 014474 006100 ROL R0 ; MULT X 2 TO SAVE OFFSET
3672 014476 012701 017230 MOV #PRTBLE, R1 ; LOAD IN BASE OF TABLE
3673 014502 060100 ADD R1, R0 ; STORE LOCATION OF BR IN R0
3674 014504 011037 017220 MOV (R0), MASK ; SAVE BR FROM QUESTION
3675 014510 162737 000060 017220 SUB #60, MASK ; MASK= DECIMAL #(0 TO 7)
3676
3677 014516 013737 017220 017222 1$: MOV MASK, LEVEL ; CREATED FROM ASCII
3678 014524 006137 017220 ROL MASK ; STORE FOR ERROR PRINTOUT
3679 014530 006137 017216 ROL NGW ; ROTATE 5 TIMES TO
3680 014534 005337 017226 DEC CT ; CREATE BR LEVELS AS BELOW
3681 014540 001371 BNE 1$ ; COUNT TO 5
3682
3683 ; NOW= 340, THEN 300, THEN 240, THEN 200 ETC.
3684 ; (340,300,240,200,140,100,40,0)
3685 014542 012601 MOV (SP)+, R1 ; RELOAD R1
3686 014544 012600 MOV (SP)+, R0 ; RELOAD R0
3687
3688 014546 000207 RTS PC ; BACK TO MAIN CODE/ QUESTIONING
3689
3690 ; MASK AND NOW ARE LOADED.

```

3691
 3692
 3693
 3694
 3695
 3696
 3697
 3698
 3699
 3700
 3701
 3702
 3703
 3704
 3705
 3706
 3707
 3708
 3709
 3710
 3711
 3712
 3713
 3714
 3715
 3716
 3717
 3718
 3719
 3720
 3721
 3722
 3723
 3724
 3725
 3726
 3727
 3728
 3729
 3730
 3731
 3732
 3733
 3734
 3735
 3736
 3737
 3738

```

:      INITIALIZATION ROUTINE
:      DEVICE CSR REGISTER
:ON FAILURE: REGISTER 0 CONTAINS ERROR ADDRESS
:SET:
:BIT01 = HALF DUPLEX
:BIT02 = MAINTENANCE
:BIT03 = CLEAR MOS
:BIT06 = RECEIVER INTERRUPT ENABLE
:BIT08 = MASTER XMTR SCAN ENB
:BIT10 = R/W BCSR
:WAIT FOR MOS TO CLEAR
:SET:
:BIT00 = RECEIVER ENABLE
  
```

```

INITA: MOV #2014,@CSR SET
        BR INTR :BIT(S)2,3,10 THEN 0
INITB: MOV #514,@CSR :BIT(S)2,3,6,8 THEN 0
        BR INTR
INITC: MOV #416,@CSR :BIT(S)1,2,3,8 THEN 0
        BR INTR
INITD: MOV #414,@CSR :BIT(S)2,3,8 THEN 0
INIR: CLR R0
1$: INC R0 :ANTIHANG
   BNE 2$ :ROUTINE
MOV (SP),R0 :RECORD SUBROUTINE CALL RETURN
SUB #2,R0 :FORM CALL ADDRESS FOR DISPLAY
HLT :BIT#4 OF DEVICE CSR FAILED TO CLEAR
2$: BIT #BIT4,@CSR :TEST HAS MOS CLEARED
   BNE 1$ :NO BRANCHES
   :SET:
   BIS #1,@CSR :BIT00 RECEIVE ENABLE
   RTS PC :CONTINUE
  
```

```

3739
3740 014644 013701 001320 ALMCK: MOV CSR,R1 ;GET CSR ADDR INTO R1
3741 014650 012761 000001 000004 MOV #1,(R1) ;SET LINE 0 IN THE TCR
3742 014656 052711 000004 BIS #BIT2,(R1) ;SET THE MAINT BIT
3743 014662 005037 001356 CLR COUNT
3744 014666 005037 001360 CLR SUM
3745 014672 005037 001352 CLR TIMERA
3746 014676 012737 000200 001354 2$: MOV #200,TIMERB ;SET UP TIME CONSTANTS
3747 014704 005711 TST (R1) ;WAIT FOR TRANSFER READY BIT
3748 014706 100373 BPL 2$
3749 014710 112761 000377 000006 MOVB #377,6(R1) ;OUTPUT A CHAR TO TBUF
3750 014716 005237 001356 INC COUNT ;COUNT EACH CHAR
3751 014722 105711 1$: TSTB (R1) ;CHECK FOR DONE IN THE CSR
3752 014724 100405 BMI 3$ ;IF SET GET OUT OF THE LOOP
3753 014726 004537 015056 JSR R5,TIME ;GIVE DONE TIME TO SET
3754 014732 000773 BR 1$ ;RETURN TO TEST FOR DONE AGAIN
3755 014734 000760 BR 2$ ;RETURN TO OUTPUT ANOTHER CHAR
3756 014736 000742 BR ALMCK ;ERROR RETURN TRY AGAIN
3757 014740 042711 000001 3$: BIC #BIT0,(R1) ;TURN OFF RCV ENABLE
3758 014744 022737 000001 001356 CMP #1,COUNT ;IF SILO LEVEL SET FOR 1 THEN GET OUT
3759 014752 001414 BEQ 4$
3760 014754 063737 001356 001360 7$: ADD COUNT,SUM ;CONTINUE TO A COUNT OF 256
3761 014762 023727 001360 000377 CMP SUM,#377
3762 014770 002771 BLT 7$
3763 014772 001403 BEQ 10$
3764 014774 163737 001356 001360 SUB COUNT,SUM ;IF EQUAL USE IT
3765 015002 000403 10$: BR 11$ ;IF GREATER SUBTRACT 1 COUNT SO IT'S LESS
3766 015004 012737 000377 001360 4$: MOV #377,SUM ;ALL SET GET OUT
3767 015012 004737 017454 11$: JSR PC,KBDINT ;PUT SUM TO MAX VALUE
3768 015016 032777 010000 164350 BIT #SW12,@SWR ;GET THE SWITCH REGISTER
3769 015024 001413 BEQ 13$ ;PRINT ALARM LEVEL?
3770 015026 000004 016756 TYPE, MALARM ;NO
3771 015032 010105 MOV R1,TTY ;YES, PRINT CSR FIRST
3772 015034 004737 016042 JSR PC,PRINTR
3773 015040 000004 016676 TYPE, MSGDAS
3774 015044 013705 001356 MOV COUNT,TTY ;PRINT ALARM LEVEL
3775 015050 004737 016042 JSR PC,PRINTR
3776 015054 000207 13$: RTS PC
3777
3778
3779
3780 015056 105237 001352 TIME: INCB TIMERA ;INCREMENT THROUGH ONE WORD
3781 015062 001012 BNE 1$ ;GO TEST FOR DONE AGAIN
3782 015064 005337 001354 DEC TIMERB ;MAKE TIMERB LARGER IF FAST PROCESSOR
3783 015070 001007 BNE 1$
3784 015072 023727 001356 000022 CMP COUNT,#22 ;HAVE OUTPUTTED 18 TIMES
3785 015100 001002 BNE 2$ ;NO, GO OUTPUT ANOTHER CHAR
3786 015102 104001 HLT+1 ;YES, DONE DID NOT SET AFTER 18 OUTPUTS
3787 ;R1 = CSR
3788 015104 005725 TST (R5)+ ;SET R5 FOR ERROR RETURN
3789 015106 005725 2$: TST (R5)+ ;SET R5 FOR NEXT OUTPUT RETURN
3790 015110 000205 1$: RTS R5 ;RETURN FROM ABOVE OR RETEST DONE
  
```

```

3791          ;          $SCOPE          SCOPE LOOP HANDLER
3792
3793          ;THIS ROUTINE HANDLES THE ITERATIONS, LOOPING, ERROR
3794          ;LOOPING, AND THE DISPLAYING OF THE TEST NUMBER.
3795
3796          ;"SCOPE" IS PLACED BETWEEN EACH SUBTEST IN THE TEST AND
3797          ;RECORDS THE STARTING ADDRESS OF THE SUBTEST IN "LAD:"
3798
3799 015112 004737 017454 TRAP$: JSR      PC,      KBDINT
3800 015116 032777 000400 164250 BIT      #SW8,@SWR      ;LOOP ON SPEC. TEST?
3801 015124 001404 BEQ      1$           ;NO LOOP ON SPEC. TEST
3802 015126 127737 164242 001310 CMPB    @SWR,ICNT     ;ON RIGHT TEST? *SW7-0*
3803 015134 001434 BEQ      OVER$       ;NOT RIGHT TEST
3804 015136 032777 040000 164230 1$: BIT    #SW14,@SWR    ;LOOP ON TEST?
3805 015144 001026 BNE     KITS$       ;LOOP ON TEST IS SET
3806 015146 032777 004000 164220 BIT    #SW11,@SWR    ;KILL ITERATIONS
3807 015154 001012 BNE     SVLAD$     ;YES - KILL ITERATIONS
3808 015156 105737 001311 TSTB   ICNT+1      ;FIRST ONE?
3809 015162 001404 BEQ      2$           ;BRANCH IF FIRST
3810 015164 123737 015252 001311 CMPB    TIMES,ICNT+1 ;DONE?
3811 015172 001013 BNE     KITS$       ;BRANCH IF NOT
3812 015174 112737 000001 001311 2$: MOVB   #1,ICNT+1   ;FIRST ITERATION
3813 015202 105237 001310 SVLAD$: INCB   ICNT     ;COUNT TEST NUMBERS
3814 015206 011637 015250 MOV     (6),LAD     ;SAVE LOOP ADDRESS
3815 015212 013737 001310 001376 MOV     ICNT,@WDISPLAY ;DISPLAY TEST NO. AND ITERATION COUNT
3816 015220 000002 RTI                    ;RETURN
3817
3818 015222 105237 001311 KITS$: INCB   ICNT+1   ;INC THE ITERATION COUNT
3819 015226 013737 001310 001376 OVER$: MOV    ICNT,@WDISPLAY ;SET UP DISPLAY
3820 015234 005737 015250 TST     LAD         ;FIRST ONE?
3821 015240 001760 BEQ     SVLAD$     ;YES
3822 015242 013716 015250 MOV     LAD,(6)    ;FUDGE RETURN ADDRESS
3823 015246 000002 RTI                    ;FIXES PS
3824
3825 015250 000000 LAD:    0           ;LOOP ADDRESS
3826 015252 000020 TIMES:  20         ;RUN 20 TIMES
  
```

```

3827
3828           :           $HLT           ERROR TIMEOUT HANDLER
3829
3830           : THIS ROUTINE PRINTS OUT ERROR MESSAGES STARTING WITH THE
3831           : ADDRESS OF THE 'HLT'. IT ALSO COUNTS THE NUMBER OF ERRORS
3832           : AND HAS THE CAPABILITY OF LOOPING ON ERROR, BELL ON ERROR,
3833           : 'HALT' ON ERROR, AND INHIBIT TIMEOUTS. AN OPTIONAL ARGUMENT
3834           : (HLT+3) WILL BE PLACED IN 'HLTCTS:' FOR ADITIONAL TIMEOUTS.
3835
3836 015254 004737 017454           EMT$: JSR PC, KBDINT
3837 015260 032777 002000 164106     BIT #SW10,@SWR           :BELL ON ERROR?
3838 015266 001402           BEQ 1$           :NO - SKIP
3839 015270 000004 000007           TYPE BELL           :RING BELL
3840 015274 005237 001312           1$: INC ERRORS           :COUNT THE NUMBER OF ERRORS
3841 015300 032777 020000 164066     BIT #SW13,@SWR           :SKIP TIMEOUT IF SET
3842 015306 001026           BNE 2$           :SKIP TIMEOUTS
3843 015310 000004 015314           TYPE ..+2           :.ASCIZ <15><12>
3844 015320 011637 015424           MOV (6),HLTADR           :PUT ADDRESS OF INSTRUCTION ON STACK
3845 015324 162737 000002 015424     SUB #2,HLTADR           :FUDGE ADDRESS
3846 015332 117737 000066 015422     MOV# @HLTADR,HLTCTS      :GET HLT ARGUMENT
3847 015340 013705 015424           MOV HLTADR,TTY           :TYPE HLTADR IN OCTAL
3848 015344 004737 016042           JSR PC,PRINTR           :TYPE LEADING ZERO'S
3849 015350 000004 015354           TYPE ..+2           :.ASCIZ " "
3850 015360 004737 015426           JSR PC,ERRORS           :GO TO USER ERROR ROUTINE
3851 015364 005777 164004           2$: TST @SWR           :HALT ON ERROR
3852 015370 100001           BPL .+4           :SKIP IF CONTINUE
3853 015372 000000           HALT           :HALT ON ERROR!
3854 015374 004737 017454           JSR PC,KBDINT
3855 015400 032777 001000 163766     BIT #SW9,@SWR           :CHECK FOR INHIBIT LOOP ON ERROR
3856 015406 001001           BNE .+4           :SKIP IF LOOP ON ERROR
3857 015410 000002           RTI           :RETURN
3858 015412 105037 001311           CLRE ICNT+1           :CLEAR ITERATION COUNT
3859 015416 000137 015222           JMP KITS           :LOOP ON TEST UNTIL NO ERRORS
3860
3861 015422 000000           HLTCTS: 0           :HLT ARGUMENT
3862 015424 000000           HLTADR: 0           :LAST HLT INSTRUCTION EXECUTED
3863
3864 015426           ERRORS:
3865 015426 013705 001320           MOV CSR,TTY           :TYPE CSR IN OCTAL
3866 015432 004737 016042           JSR PC,PRINTR           :TYPE LEADING ZERO'S
3867 015436 042737 007700 015464     BIC #7700,2$
3868 015444 105337 015422           1$: DECB HLTCTS           :
3869 015450 100411           BMI 3$           :
3870 015452 062737 000100 015464     ADD #100,2$
3871 015460 000004 016527           TYPE SPACE
3872 015464 010005           2$: MOV %0,TTY           :TYPE REGISTER X IN OCTAL
3873 015466 004737 016042           JSR %7,PRINTR
3874 015472 000164           BR 1$
3875 015474 000207           3$: RTS PC

```

```

3876
3877
3878 ;SUBROUTINE TO SAVE INPUT AS OCTAL NUMBER
3879 015476 012737 000001 015770 READIN: MOV #1,INHRE
3880 015504 004737 015644 JSR PC, READ$ ;GO READ TTY UNTIL CR
3881 015510 005037 015770 CLR INHRE
3882 015514 010146 MOV R1,-(6) ;PUSH R1 ON STACK
3883 015516 010246 MOV R2,-(6) ;PUSH R2 ON STACK
3884 015520 010346 MOV R3,-(6) ;PUSH R3 ON STACK
3885 015522 012501 MOV (R5)+, R1
3886 015524 012737 000020 017664 MOV #20,CNT
3887 015532 012702 015772 MOV #INPUT,R2
3888 015536 122712 000120 CMPB #120,(R2) ;CHECK FOR 'P'
3889 015542 001425 BEQ 3$
3890 015544 005011 CLR (R1)
3891 015546 112203 1$: MOVB (R2)+,R3
3892 015550 120327 000015 CMPB R3,#15
3893 015554 001420 BEQ 3$
3894 015556 162703 000060 SUB #60,R3
3895 015562 032703 177770 BIT #177770,R3
3896 015566 001013 BNE 3$ ;BRANCH IF BAD DATA
3897 015570 006311 ASL (R1)
3898 015572 103410 BCS 2$
3899 015574 006311 ASL (R1)
3900 015576 103406 BCS 2$
3901 015600 006311 ASL (R1)
3902 015602 103404 BCS 2$
3903 015604 050311 BIS R3,(R1)
3904 015606 005337 017664 DEC CNT
3905 015612 000755 BR 1$
3906 015614 000244 2$: CLZ
3907 015616 013737 177776 015642 3$: MOV @#PS, PSTEMP ;MAKE SURE Z-BIT IS CLR
3908 015624 012603 MOV (6)+,R3 ;SAVE CONDITION CODES
3909 015626 012602 MOV (6)+,R2 ;POP STACK INTO R3
3910 015630 012601 MOV (6)+,R1 ;POP STACK INTO R2
3911 015632 013737 015642 177776 MOV PSTEMP,@#PS ;POP STACK INTO R1
3912 015640 000205 RTS R5 ;RESTORE CONDITION CODES
3913
3914 015642 000000 PSTEMP: 0 ;TEMPORARY STORAGE FOR PS
3915
3916 015644 010346 READ$: MOV R3,-(6) ;SAVE R3
3917 015646 012703 015772 1$: MOV #INPUT,R3 ;GET ADDRESS
3918 015652 022703 016012 2$: CMP #INPUT+20,R3 ;BUFFER FULL?
3919 015656 001415 BEQ 4$ ;YES - TYPE '?'
3920 015660 105737 177560 TSTB @#177560 ;WAIT FOR
3921 015664 100375 BPL -4 ;A CHARACTER
3922 015666 113713 177562 MOVB @#177562,(3) ;GET CHARACTER
3923 015672 142713 000200 BICB #200,(3) ;GET RID OF JUNK
3924 015676 122713 000177 CMPB #177,(3) ;IS IT A RUBOUT
3925 015702 001403 BEQ 4$ ;SKIP IF NOT
3926 015704 122713 000025 CMPB #25,(3)
3927 015710 001006 BNE 3$
3928 015712 4$:
3929 015712 000004 015716 TYPE +2 ;ASCIZ '?'<15><12>'= ''
3930 015724 000750 BR 1$ ;ZAP THE BUFFER AND LOOP
3931 015726 111337 016514 3$: MOVB (3),TYPE ;SET UP FOR TYPING
    
```


3932	015732	000004	016514		TYPE	..TYPE		
3933	015736	122723	000015		CMPB	#15.(3)+		:ECHO IT
3934	015742	001343			BNE	2\$:CHECK FOR RETURN
3935	015744	005737	015770		TST	INHRE		:LOOP IF NOT RETURN
3936	015750	001401			BEQ	5\$		
3937	015752	000402			BR	6\$		
3938	015754	105063	177777	5\$:	CLRB	-1(3)		:ZAP RETURN (THE 15)
3939	015760	000004	000012	6\$:	TYPE	.12		:TYPE A LINE FEED
3940	015764	012603			MOV	(6)+,R3		:RESTORE R3
3941	015766	000207			RTS	PC		:RETURN
3942								
3943	015770	000000		INHRE:	0			
3944	015772	000020		INPUT:	.BLKW 20			:TTY INPUT AREA

```

3945          :          $OCTAL          OCTAL TYPEOUT ROUTINE
3946
3947          :THIS ROUTINE IS USED TO TYPE AN OCTAL NUMBER ON THE TTY. IT WILL TYPE
3948          :ALL 6 CHARACTERS, SUPPRESS LEADING ZEROES, TYPE AN 18 BIT ADDRESS, OR TYPE
3949          :THE 16 BITS. IT IS CALLED VIA THE DUMP, SDUMP, DUMP18, OR BITYPE MACRO'S.
3950
3951 016032 012737 170101 016200 BITYPS: MOV #170101,.PR ;SET BIT FLAG ANS 16. CHARACTER COUNT
3952 016040 000411          BR .PTIT ;NOW TYPE IT IN BIT FORM
3953 016042 112737 000001 016200 PRINTR: MOVB #1,.PR ;SET ZERO FILL SWITCH
3954 016050 000402          BR .+6 ;SKIP
3955 016052 005037 016200 PRINTS: CLR .PR ;SUPPRESS LEADING ZERO'S
3956 016056 112737 177772 016201          MOVB #-6,.PR+1 ;SET COUNT
3957 016064 010446          .PTIT: MOV R4,-(6) ;SAVE R4
3958 016066 012704 016202          MOV #.PR+2,R4 ;SET POINTER TO FIRST ASCII CHAR.
3959 016072 105014          CLR B (4) ;CLEAR FIRST BYTE
3960 016074 000411          BR .PRF ;ROTATE FIRST BIT
3961 016076 105014          .PRL: CLRB (4) ;CLEAR BYTE OF CHARACTER
3962 016100 032737 000100 016200          BIT #100,.PR ;BIT TYPING MODE?
3963 016106 001004          BNE .PRF ;YES - SKIP 2 ROTATES
3964 016110 006105          ROL TTY ;ROTATE BIT INTO C
3965 016112 106114          ROL B (4) ;PACK IT
3966 016114 006105          ROL TTY ;ROTATE BIT INTO C
3967 016116 106114          ROL B (4) ;PACK IT
3968 016120 006105          .PRF: ROL TTY ;ROTATE BIT INTO C
3969 016122 106114          ROL B (4) ;PACK IT
3970 016124 105714          TST B (4) ;IS IT ZERO?
3971 016126 001402          BEQ .+6 ;SKIP INC
3972 016130 105237 016200          INCB .PR ;SET FILL SWITCH
3973 016134 105737 016200          TST B .PR ;CHECK FILL SWITCH
3974 016140 001402          BEQ .+6 ;SKIP BITSET
3975 016142 152724 000060          BISB #'0,(4)+ ;MAKE INTO ASCII CHAR
3976 016146 105237 016201          INCB .PR+1 ;INC COUNT
3977 016152 001351          BNE .PRL ;REPEAT
3978 016154 022704 016202          CMP #.PR+2,R4 ;EMPTY BUFFER?
3979 016160 001002          BNE .+6 ;SKIP IF NOT
3980 016162 112724 000060          MOVB #'0,(4)+ ;LOAD 1 ZERO
3981 016166 105014          CLRB (4) ;NULL TERMINATOR
3982 016170 000004 016202          TYPE .PR+2 ;TYPE IT
3983 016174 012604          MOV (6)+,R4 ;RESTORE R4
3984 016176 000207          RTS PC ;RETURN
3985 016200 000012          .PR: .BLKW 12 ;COUNT, SWITCH, AND OUTPUT BUFFER

```

```

3986 016224 012777 016352 000126 PDOWN$: MOV #ILLUP,@PUVECS ;SET FOR FAST UP
3987 016232 012777 000340 000122 MOV #340,@PUVECS+2 ;PRIO:7
3988 016240 010046 MOV R0,-(6) ;PUSH R0 ON STACK
3989 016242 010146 MOV R1,-(6) ;PUSH R1 ON STACK
3990 016244 010246 MOV R2,-(6) ;PUSH R2 ON STACK
3991 016246 010346 MOV R3,-(6) ;PUSH R3 ON STACK
3992 016250 010446 MOV R4,-(6) ;PUSH R4 ON STACK
3993 016252 010546 MOV R5,-(6) ;PUSH R5 ON STACK
3994 016254 010637 016356 MOV SP,.SAVR6 ;SAVE SP
3995 016260 012777 016270 000072 MOV #PUP$,@PUVECS ;SET UP VECTOR
3996 016266 000000 HALT ;WAIT FOR PF
3997
3998 016270 013706 016356 PUP$: MOV .SAVR6,SP ;GET SP
3999 016274 005001 CLR R1 ;WAIT LOOP FOR THE TTY
4000 016276 005201 1$: INC R1 ;WAIT FOR THE INC
4001 016300 001376 BNE 1$ ;OF WORD
4002 016302 012605 MOV (6)+,R5 ;POP STACK INTO R5
4003 016304 012604 MOV (6)+,R4 ;POP STACK INTO R4
4004 016306 012603 MOV (6)+,R3 ;POP STACK INTO R3
4005 016310 012602 MOV (6)+,R2 ;POP STACK INTO R2
4006 016312 012601 MOV (6)+,R1 ;POP STACK INTO R1
4007 016314 012600 MOV (6)+,R0 ;POP STACK INTO R0
4008 016316 012737 016224 000024 MOV #PDOWN$,@#24 ;SET UP THE POWER DOWN VECTOR
4009 016324 012737 000340 000026 MOV #340,@#26 ;PRIO:7
4010 016332 000004 016336 TYPE .,+2 ;.ASCIZ <15><12>'POWER'
4011 016346 000137 002472 JMP RESTAR ;JMP TO USER ADDRESS
4012
4013 016352 000000 ILLUP: HALT ;THE POWER UP SEQUENCE WAS STARTED
4014 016354 000776 BR .-2 ; BEFORE THE POWER DOWN WAS COMPLETE
4015
4016 016356 000000 .SAVR6: 0 ;PUT THE SP HERE
4017 016360 000024 000026 PUVECS: 24.26 ;POWER UP VECTOR
  
```

```

4018
4019
4020
4021
4022
4023 016364 022716 001000
4024 016370 002407
4025 016372 162716 000004
4026 016376 012601
4027 016400 005726
4028 016402 011602
4029 016404 104002
4030
4031
4032 016406 000002
4033
4034
4035
4036
4037
4038
4039
4040
4041
4042
4043 016410 010546
4044 016412 017605 000002
4045 016416 032705 177400
4046 016422 001004
4047 016424 010537 016514
4048 016430 012705 016514
4049 016434 105715
4050 016436 001406
4051 016440 112537 177566
4052 016444 105737 177564
4053 016450 100375
4054 016452 000770
4055 016454 017646 000002
4056 016460 062766 000002 000004
4057 016466 022666 000002
4058 016472 001006
4059 016474 062705 000002
4060 016500 042705 000001
4061 016504 010566 000002
4062 016510 012605
4063 016512 000002
4064 016514 000000
4065
4066 016516 005015 047505 000120
4067 016524 005015 000
4068 016527 040 000040
4069 016532 005015 044506 051522
4070 016540 020124 045104 030461
4071 016546 040440 042104 042522
4072 016554 051523 020072 000040
4073 016562 005015 042526 052103

```

```

:*****
:IOT HANDLER. REENTERENT ROUTINE TO EITHER TYPE MESSAGES OR
:INDICATE A FAULSE INTERUPT OR TRAP.
:*****
IOTRAP: CMP #1000, (SP) ;CHECK RETURN ADDRESS FOR FAULSE TRAP
BLT IOT$ ;BRANCH IF 'TYPE' COMMAND INTENDED
SUB #4, (SP) ;GET VECTOR ADDRESS FROM RETURN ADDRESS
MOV (SP)+, R1 ;PUT IN R1 FOR TYPING
TST (SP)+ ;POP STACK
MOV (SP), R2 ;SAVE RETURN ADDRESS FOR TYPING
HLT+2 ;UNEXPECTED INTERUPT OR TRAP
;R1 = VECTOR ADDRESS
;R2 = RETURN PC
RTI ;CONTINUE THE PROGRAM

:MCALL $TYPE
: $TYPE MESSAGE TIMEOUT ROUTINE

:THIS ROUTINE IS USE TO TYPE ASCII MESSAGES ON THE TTY. THE
:CALL CAN BE IN ONE OF 3 FORMS: 1) 'TYPE ,ADR' - TYPES THE
:MESSAGE STARTING IN LOCATION 'ADR:' 2) 'TYPE ,CHAR' - TYPES
:THE ASCII 'CHAR', AND 3) 'PRINT <<15><12>'MESSAGE'> - TYPES
:THE MESSAGE WHICH IS INLINE ASCII.
IOT$: MOV TTY,-(6) ;SAVE TTY
MOV @2(6),TTY ;GET ADDRESS TO BE TYPED
BIT #177400,TTY ;IS IT A TYPEN?
BNE 1$ ;NO
MOV TTY,,TYPE ;GET THE CHARACTER
MOV #,TYPE,TTY ;FUDGE THE ADDRESS
1$: TSTB (TTY) ;TERMINATOR?
BEQ 2$ ;GET OUT IF SO
MOVB (TTY)+,@#177566 ;LOAD AND TYPE THE CHARACTER
TSTB @#177566 ;IS THE PRINTER READY
BPL -4 ;WAIT UNTIL IT IS
BR 1$ ;GET THE NEXT CHARACTER
2$: MOV @2(6),-(6) ;GET ADDRESS TO BE TYPED
ADD #2,4(6) ;ADD 2 TO THE ADDRESS
CMP (6)+,2(6) ;IS IT .+2?
BNE 3$ ;NO
ADD #2,TTY ;ADD 2 TO THE ADDRESS
BIC #1,TTY ;BACK UP TO AN EVEN BYTE
MOV TTY,2(6) ;RESTORE ADDRESS
3$: MOV (6)+,TTY ;RESTORE TTY
RTI ;RETURN
;CHARACTER TYPE LOCATION
.TYPE: 0

MEOP: .ASCIZ <15><12> 'EOP'
RETURN: .ASCIZ <15><12>
SPACE: .ASCIZ ""
MSGADR: .ASCIZ <15><12>'FIRST DJ11 ADDRESS: ''

MSGVEC: .ASCIZ <15><12>'VECTOR ADDRESS: ''

```

4074	016570	051117	040440	042104	
4075	016576	042522	051523	020072	
4076	016604	000040			
4077	016606	005015	047516	020056	MSGNUM: .ASCIZ <15><12>'NO. OF DJ11'S: ''
4078	016614	043117	042040	030512	
4079	016622	023461	035123	020040	
4080	016630	000			
4081	016631	015	051412	040524	MSGCON: .ASCIZ <15><12>'STANDARD CONFIGURATION? ''
4082	016636	042116	051101	020104	
4083	016644	047503	043116	043511	
4084	016652	051125	052101	017511	
4085	016660	037516	020040	000	
4086	016665	015	046012	047111	MSGLEN: .ASCIZ <15><12>'LINES ''
4087	016672	051505	000040		
4088	016676	026440	000040		MSGDAS: .ASCIZ '' - ''
4089	016702	005015	044103	051101	MSGLEN: .ASCIZ <15><12>'CHAR LENGTH: ''
4090	016710	046040	047105	052107	
4091	016716	035110	020040	000	
4092	016723	015	050012	051101	MSGPAR: .ASCIZ <15><12>'PARITY(NO, ODD, EVEN): ''
4093	016730	052111	024131	047516	
4094	016736	020054	042117	026104	
4095	016744	042440	042526	024516	
4096	016752	020072	000040		
4097	016756	005015	044523	047514	MALARM: .ASCIZ <15><12>'SILO ALARM LEVEL FOR CSR'<15><12>
4098	016764	040440	040514	046522	
4099	016772	046040	053105	046105	
4100	017000	043040	051117	041440	
4101	017006	051123	005015	000	
4102	017013	015	041412	042132	MTITLE: .ASCIZ <15><12>'CZDJA-F DJ11 LOGIC TESTS'<15><12>
4103	017020	040512	043055	020040	
4104	017026	045114	030461	046040	
4105	017034	043517	041511	052040	
4106	017042	051505	051524	005015	
4107	017050	000			
4108					
4109					
4110					
4111					
4112	017051	015	053412	040510	MSBR. .ASCIZ <15><12>'WHAT ARE THE BR PRIORITIES ?? FOR: /
4113	017056	020124	051101	020105	
4114	017064	044124	020105	051102	
4115	017072	050040	044522	051117	
4116	017100	052111	042511	020123	
4117	017106	037477	043040	051117	
4118	017114	035072	000		
4119	017117	015	020012	044124	MSBAD: .ASCIZ <15><12>' THAT BR IS A GOOD FOR NOTHING. TRY AGAIN? /
4120	017124	052101	041040	020122	
4121	017132	051511	040440	043440	
4122	017140	047517	020104	047506	
4123	017146	020122	047516	044124	
4124	017154	047111	027107	052040	
4125	017162	054522	040440	040507	
4126	017170	047111	000077		
4127	017174	005015	042504	044526	VICE: .ASCIZ<15><12>'DEVICE /
4128	017202	042503	020040	000040	
4129					

```

4130          :          STORAGE FOR INTERRUPT TESTING
4131
4132          .EVEN
4133 017210 000061  DEVNUM: 000061          : DEFAULT FIRST DEVICE # 1
4134 017212 000061  UNITS1: 000061          : STORE FOR UNITS+1, DFAULT 1
4135 017214 000001  PRI0LO: .WORD 1          : FOR DECIMAL PRIORITIES #
4136 017216 000001  NOW: .WORD 1          : OCTAL # OF BR
4137          :          : OF CURRENT TESTING
4138 017220 000001  MASK: .WORD 1          : OCTAL # OF ALLOWED BR
4139 017222 000001  LEVEL: .WORD 1        : DECIMAL BR PRIO
4140 017224 000001  UUTD1: .WORD 1        : DEVICE # FOR ROTATE LEFT OF BR
4141 017226 000001  CT: .WORD 1          : STORE COUNTER FOR ROLS
4142 017230 000026  PRTBLE: .BLKW 26      : TABLE OF INPUT BR
4143          .EVEN
4144
4145          :SUBROUTINE TO AUTOMATICLY DETERMINE THE NUMBER OF DJ11'S ON THE SYSTEM
4146          : AND WHERE THEIR INTERUPT VECTORS ARE.
4147          : THIS ROUTINE IS ONLY USED IF LOC 42 IS NOT ZERO, AS WHEN THE PROGRAM IS
4148          : BEING RUN UNDER ACT11 OR DDP MONITOR CONTROL.
4149          : NOTE: SOME OF THE LOGIC MUST BE FUNCTIONAL OR THIS ROUTINE WILL BOMB
4150
4151 017304 012700 160000  AUTO:  MOV #160000,R0          : START AT NO RESPONSE BASE DJ11: -10'8)
4152 017310 012702 000001  MOV #1, R2          : COUNTER OF NON RESPONSE OR DJ11 OR DONE
4153 017314 012737 000002 000006  MOV #RTI, @#6          : RTI WHEN TIME-OUT
4154 017322 005001 5$: CLR R1          : SET UP COUNTER
4155 017324 000261 1$: SEC          : SET CARRY
4156 017326 005710  IST (R0)          : CHECK FOR ANY DJ11'S
4157 017330 103404  BCS 2$          : BRANCH IF IT TIMED OUT
4158 017332 062700 000010  ADD #10, R0          : POINT TO NEXT DJ11 ADDRESS
4159 017336 005201  INC R1          : COUNT DJ11'S
4160 017340 000771  BR 1$          : LOOK FOR MORE
4161
4162 017342 005302 2$: DEC R2          : COUNT DOWN DEVICES
4163 017344 100405  BMI 7$          : BRANCH IF DONE
4164 017346 062700 000010  ADD #10, R0          : POINT TO FIRST DJ11
4165 017352 010037 001340  MOV R0, DEVADR      : SAVE FIRST DJ11 ADDRESS
4166 017356 000761  BR 5$          : GO COUNT DJ11'S
4167
4168 017360 005037 000006 7$: CLR @#6          : RESTORE TIME-OUT CATCHER
4169 017364 010137 001344  MOV R1, UNITS       : SAVE COUNT
4170 017370 001003  BNE 3$          : BRANCH IF NOT ZERO
4171 017372 104000  HLT          : REPORT THAT NO DJ11'S WERE FOUND
4172 017374 000137 014376  JMP DONE          : EXIT THIS PROGRAM
4173
4174          :ROUTINE TO DETERMINE VECTOR ADDRESSES
4175
4176 017400 013746 000020 3$: MOV @#20, -(SP)    : SAVE IOT VECTOR ON THE STACK
4177 017404 012737 017440 000020  MOV #4$, @#20      : RESET IOT VECTOR
4178 017412 013701 001340  MOV DEVADR, R1     : GET FIRST DJ ADR
4179 017416 012721 040400  MOV #40400, (R1)+  : SET CSR
4180          :BIT8 = TRANS SCAN ENABLE
4181          :BIT14 = TRANS INTERUPT ENABLE
4182 017422 005721  IST (R1)+          : INC POINTER
4183 017424 012721 000001  MOV #1, (R1)+     : SET ICR, LINE 0
4184 017430 000001  WAIT          : WAIT FOR AN INTERRUPT
4185 017432 012637 000020  MOV (SP)+, @#20   : RESTORE IOT VECTOR
  
```

```

4186 017436 000207          RTS      PC
4187
4188 017440 162716 000010      4$:    SUB      #10, (SP)      ;REPOSITION ADDRESS TO RVC VEC
4189 017444 011637 001342      MOV      (SP), VECADR    ;SAVE FIRST VECTOR
4190 017450 022626      CMP      (SP)+, (SP)+    ;RESET STACK FROM IOT
4191 017452 000002      RTI      ;RETURN FROM INTERRUPT - RESTORE STATUS
4192
4193
4194 017454 022737 000176 001374 KBDINT: CMP      #SWREG,SWR
4195 017462 001016      BNE      1$
4196 017464 005037 017522      CLR      TMP1
4197 017470 113737 177562 017522      MOV      177562,TMP1
4198 017476 142737 000200 017522      BIC      #200,TMP1
4199 017504 122737 000007 017522      CMP      #7,TMP1
4200 017512 001002      BNE      1$
4201 017514 004737 017604      JSR      PC,CNTLU
4202 017520 000207      1$:    RTS      PC
4203
4204 017522 000000      TMP1:  0
4205
4206
4207 017524 013746 000006      SUSWRR: MOV      6,-(SP)
4208 017530 013746 000004      MOV      4,-(SP)
4209 017534 012737 017554 000004      MOV      #1$,4
4210 017542 022777 177777 161624      CMP      #-1,@SWR
4211 017550 001402      BEQ      2$
4212 017552 000407      BR       3$
4213 017554 022626      1$:    CMP      (SP)+,(SP)+
4214 017556 012737 000176 001374 2$:    MOV      #SWREG,SWR
4215 017564 012737 000174 001376      MOV      #DISPREG,DISPLAY
4216 017572 012637 000004 3$:    MOV      (SP)+,4
4217 017576 012637 000006      MOV      (SP)+,6
4218 017602 000207      RTS      PC
4219
4220
4221 017604 022737 000176 001374 CNTLU:  CMP      #SWREG,SWR
4222 017612 001023      BNE      1$
4223 017614 000004 017676      TYPE      ,SWREQ
4224 017620 013705 000176      MOV      SWREG,TTY
4225 017624 004737 016042      JSR      PC,PRINTR      ;TYPE SWREG IN OCTAL
4226 017630 000004 017666      TYPE      ,NEWIS      ;TYPE LEADING ZERO'S
4227 017634 004537 015476      JSR      R5,READIN
4228 017640 017522      .WORD    TMP1
4229 017642 001360      BNE      CNTLU
4230 017644 022737 000020 017664      CMP      #20,CNT
4231 017652 001403      BEQ      1$
4232 017654 013777 017522 161512      MOV      TMP1,@SWR
4233 017662 000207      1$:    RTS      PC
4234
4235 017664 000000      CNT:   0
4236
4237 017666 020040 042516 036527 NEWIS:  .ASCIZ  '' NEW= ''
4238 017674 000040
4239 017676 005015 053523 036522 SWREQ:  .ASCIZ  <15><12>'SWR= ''
4240 017704 000040
4241

```

77-CZDJA-F-0 DJ11 LOGIC TESTS MACY11 30A(1052) 26-FEB-79 15:47 J 7 PAGE 88
CZDJAF.P11 26-FEB-79 15:41 AUTOMATIC SYSTEM SIZER

SFO 0087

4242

000001

.END

DEVNEW	002400	772#	783	786										
DEVNUM	017210	766*	773	781*	782	4133#								
DISPLA	001376	604#	3815*	3819*	4215*									
DISPRE	000174	547#	4215											
DJLEN	001364	598#	805*	812*	1430	1509	1588	1667	1691*	1748	1827	1906	1985	2009*
		2066	2145	2224	2303	2327*	2384	2463	2542	2621	2645*	3588	3607*	3619
		3631*												
DJMXNO-	000020	535#	689											
DJPAR	001366	599#	806*	813*	3396	3470								
DJUUT	001362	597#	692*	807	809*	814*	3639	3670						
DONE	014376	3640	3644#	4172										
EMTS	015254	618	3836#											
END22	004732	1367#												
END56	014314	3611	3631#											
ERRORS	001312	573#	622*	3840*	3861									
ERRORS	015426	3850	3864#											
ERROR1	004672	1333	1341	1344#										
ERROR2	011752	2956	2963	2966#										
FTIME	001400	607#	791	796*										
GETADR	001514	627	631#	636	642									
GETLEN	002156	726#	732	736	740									
GETNUM	001650	656#	665	667	672	674	677	690						
GETPAR	002242	730	742#	752										
GETVER	001572	644#	647	653	655									
HLT	104000	504#	848	853	868	872	877	889	893	898	910	914	919	931
		935	940	952	956	961	973	977	982	994	998	1003	1019	1025
		1029	1036	1040	1048	1064	1073	1087	1092	1097	1110	1115	1129	1138
		1152	1157	1162	1182	1192	1196	1201	1223	1234	1241	1246	1270	1281
		1287	1355	1411	1417	1427	1433	1439	1444	1449	1490	1496	1506	1512
		1518	1523	1528	1569	1575	1585	1591	1597	1602	1607	1648	1654	1664
		1670	1676	1681	1686	1729	1735	1745	1751	1757	1762	1767	1808	1814
		1824	1830	1836	1841	1846	1887	1893	1903	1909	1915	1920	1925	1966
		1972	1982	1988	1994	1999	2004	2047	2053	2063	2069	2075	2080	2085
		2126	2132	2142	2148	2154	2159	2164	2205	2211	2221	2227	2233	2238
		2243	2284	2290	2300	2306	2312	2317	2322	2365	2371	2381	2387	2397
		2398	2403	2444	2450	2460	2466	2472	2477	2482	2523	2529	2539	2545
		2551	2556	2561	2602	2608	2618	2624	2630	2635	2640	2688	2692	2705
		2712	2716	2750	2763	2768	2772	2776	2780	2784	2789	2835	2843	2850
		2883	2891	2900	2977	2987	2991	3031	3043	3052	3056	3074	3078	3092
		3096	3107	3119	3137	3149	3157	3162	3169	3180	3195	3204	3212	3217
		3221	3228	3240	3246	3252	3294	3304	3309	3314	3322	3327	3336	3358
		3363	3389	3394	3400	3405	3410	3414	3418	3423	3456	3462	3474	3480
		3486	3494	3501	3506	3539	3543	3547	3551	3555	3594	3599	3615	3622
		3730	3786	4029	4171									
		3844*	3845*	3846	3847	3862#								
HLTADR	015424	3846*	3861#	3868*										
HLTCTS	015422	572#	830*	802	3808	3810	3812*	3813*	3815	3818*	3819	3825	3858*	
ICNT	001310	3986	4013#											
ILLUP	016352	727*	743*	3879*	3881*	3935	3943#							
INHRE	015770	3375	3440	3711#										
INITA	014550	2934	3714#											
INITB	014560	2861	3717#											
INITC	014570	1392	1471	1550	1629	1710	1789	1868	1947	2028	2107	2186	2265	2346
INITD	014600	2425	2504	2583	2663	2734	2805	3017	3720#					
INTR	014606	3712	3715	3718	3722#									
INPUT	015772	659	696	705	708	711	729	731	733*	737*	745	747	749	751

OVERS	015226	3803	3819#															
PARITY	001300	568#	716	757	3396	3470												
PARIT2	001302	569#																
PARIT3	001304	570#																
PARIT4	001306	571#																
PCNT	001314	574#	623*	624*	3646*	3647*	3603											
PDOWN\$	016224	616	3986#	4008														
PRINTR	016042	3772	3775	3848	3866	3873	3953#	4225										
PRINTS	016052	720	724	3955#														
PRIOL0	017214	1311*	1313*	1314	2921*	2923*	2924	4135#										
PRIBLE	017230	769	3672	4142#														
PS =	177776	506#	1322*	1323*	1369*	2932*	2933*	3001*	3129*	3130*	3189*	3190*	3260*	3526*				
		3569*	3570*	3907	3911*													
PSTEM0	015642	3907*	3911	3914#														
PUPS	016270	3995	3998#															
PUVEC\$	016360	3986*	3987*	3995*	4017#													
QABR	002344	553	629	698	707	710	765#											
RBUF	001322	577#	817*	1406	1442	1485	1521	1564	1600	1643	1679	1724	1760	1803				
		1839	1882	1918	1961	1997	2042	2078	2121	2157	2200	2236	2279	2315				
		2360	2396	2439	2475	2518	2554	2597	2633	2686	2714	2766	2774	2782				
		2841	2846	2889	2898	2985	3090	3150	3155	3215	3250	3300	3325	3385				
		3421	3452	3504	3549	3597	3613	3627										
RCV LVL	001332	583#	825*	2920*	2999	3132*	3254	3255*	3568*	3632	3633*							
RCVVEC	001330	582#	804*	811*	823	2919*	2999*	3131*	3191*	3254*	3567*	3632*						
READIN	015476	634	645	3879#	4227													
READS	015644	657	695	728	744	775	3880	3916#										
RESTAR	002472	789#	3642	3661	4011													
RETURN	016524	632	4067#															
SCOPE -	104400	503#	832	858	879	900	921	942	963	984	1005	1052	1076	1099				
		1117	1141	1164	1204	1255	1289	1371	1453	1532	1611	1690	1771	1850				
		1929	2008	2089	2168	2247	2326	2407	2486	2565	2644	2721	2793	2854				
		2904	3005	3262	3345	3365	3429	3515	3557	3636								
SPACE	U16527	3871	4068#															
SSS	002060	706	708#															
STACK -	001200	536#	611	790														
SUM	001360	595#	3586	3744*	3760*	3761	3764*	3766*										
SUSWRR	017524	612	4207#															
SVLADS	015202	3807	3813#	3821														
SWR	001374	603#	793	3649	3768	3800	3802	3804	3806	3837	3841	3851	3855	4194				
		4210	4214*	4221	4232*													
SWREG	000176	548#	793	4194	4214	4221	4224											
SWREQ	017676	4223	4239#															
SW10 -	002000	455#	3649	3836	3837													
SW11	004000	454#	3806															
SW12 =	010000	453#	3768															
SW13	020000	452#	3841															
SW14 =	040000	451#	3804															
SW15 -	100000	450#																
SW8 =	000400	458#	3799	3800														
SW9 -	001000	457#	3855															
TBUF	001326	580#	822*	1187	1230	1402*	1481*	1560*	1639*	1720*	1799*	1878*	1957*	2038*				
		2117*	2196*	2275*	2356*	2435*	2514*	2593*	2675*	2744*	2747*	2826*	2874*	2947*				
		3028*	3065*	3139*	3142*	3197*	3200*	3279*	3282*	3532*	3553	3584*						
TCR	001324	578#	819*	1106*	1107	1112*	1113	1124*	1125*	1126	1131*	1132*	1133*	1134*				
		1135	1149*	1150	1154*	1155	1159*	1160	1176*	1198*	1217*	1243*	1266*	1284*				
		1325*	1361*	1399*	1451*	1478*	1530*	1557*	1609*	1636*	1688*	1717*	1769*	1796*				

TSI55	013734	3526#												
TST56	014054	3565#												
TST6	003274	949#												
TST7	003346	970#												
TYPE =	000004	505#	631	632	633	644	656	694	718	721	726	742	765	772
		773	774	785	3648	3651	3652	3770	3775	3839	3843	3849	3871	3929
		3932	3939	3982	4010	4223	4226							
TYPLIN	002124	718#	756	761										
T50A	012172	3065#												
T50B	012236	3089#												
T50C	012326	3129#												
T50D	012542	3189#												
UNITS	001344	589#	691*	760	771	807	3639	4169*						
UNITS1	017212	767*	771*	782	4134#									
UUTDJ	017224	4140#												
VECADR	001342	588#	646	648	650*	651*	652	654	811	4189*				
VICE	017174	772	4127#											
WAIT1	004636	1326#	1327											
XMTLVL	001336	585#	829*	1310*	1367	1368*	3000*							
XMTVEC	001334	584#	827*	1309*	1367*									
\$ENDAD=	014450	544	3656#											
.	017706	537#	541	543#	546#	550#	552#	555#	567#	867	871	876	888	892
		897	909	913	918	930	934	939	951	955	960	972	888	892
		993	997	1002	1018	1028	1035	1039	1047	1063	1072	1086	1091	1096
		1109	1114	1128	1137	1151	1156	1161	1166	1191	1195	1200	1233	1240
		1245	1286	1374	1416	1426	1432	1438	1443	1448	1495	1505	1511	1517
		1522	1527	1574	1584	1590	1596	1601	1606	1653	1663	1669	1675	1680
		1685	1692	1734	1744	1750	1756	1761	1766	1813	1823	1829	1835	1840
		1845	1892	1902	1908	1914	1919	1924	1971	1981	1987	1993	1998	2003
		2010	2052	2062	2068	2074	2079	2084	2131	2141	2147	2153	2158	2163
		2210	2220	2226	2232	2237	2242	2289	2299	2305	2311	2316	2321	2328
		2370	2380	2386	2392	2397	2402	2449	2459	2465	2471	2476	2481	2528
		2538	2544	2550	2555	2560	2607	2617	2623	2629	2634	2639	2653	2687
		2691	2704	2711	2715	2749	2756	2762	2767	2771	2775	2779	2783	2788
		2842	2890	2899	2986	2990	3030	3042	3051	3055	3077	3091	3095	3106
		3118	3136	3156	3161	3168	3179	3194	3211	3216	3220	3227	3239	3245
		3251	3287	3293	3308	3313	3321	3335	3357	3362	3393	3404	3409	3413
		3417	3422	3461	3485	3493	3500	3505	3538	3542	3546	3550	3554	3593
		3598	3621	3656	3663	3825	3843	3844#	3849	3850#	3852	3856	3861	3862
		3921	3929	3944#	3954	3971	3974	3979	3985#	4010	4014	4053	4142#	
.BIT	177777	441#	450	451	452	453	454	455	456	457	458	459	3649	3799
		3836	3841	3855										
.PR	016200	3951*	3953*	3955*	3956*	3958	3962	3972*	3973	3976*	3978	3982	3985#	
.PRF	016120	3960	3963	3968#										
.PRL	016076	3961#	3977											
.PTIT	016064	3952	3957#											
.SAVR6	016356	3994*	3998	4016#										
.TYPE	016514	3931*	3932	4047*	4048	4064#								

\$SCOPE	1#	3791
\$SETUP	1#	
\$SRAT	1#	
\$SWDOC	1#	447
\$SWRDF	1#	602
\$SWRRR	1#	4206
\$TRAP	1#	
\$TYPE	1#	4035
\$URAT	1#	
\$XRDY	835#	1206
\$XRDYC	835#	2723
\$XRDYO	835#	1167
\$XUOR	835#	3264
.SCOP	1#	
.SCOPE	1#	

. ABS. 017706 000

ERRORS DETECTED: 0

CZDJAF.BIN,CZDJAF.SEQ/CRF/SOL/NL:TOC=CZDJAF.MAC,CZDJAF.P11
RUN-TIME: 11 16 .9 SECONDS
RUN-TIME RATIO: 159/29-5.4
CORE USED: 32K (63 PAGES)