

DZV11

DZV-11
CVDZABO

AH-A878B-MC
FICHE 1 OF 1

MAY 1981
COPYRIGHT © 77 B
MADE IN USA



Microfiche grid containing multiple frames of data, likely a technical drawing or document. The content is too small to read clearly but appears to be organized in a structured layout.

.REM 8

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

IDENTIFICATION

PRODUCT CODE:	AC-A877B-PC
PRODUCT NAME:	CVDZAB0 DZV11 4 LINE ASYNC MUX TESTS PART 1 OF 2
DATE RELEASED:	JANUARY 1981
MAINTAINER:	DIAGNOSTIC ENGINEERING

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) 1977,1981 DIGITAL EQUIPMENT CORPORATION

41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79

1. ABSTRACT

THE FUNCTION OF THE DZV11 DIAGNOSTICS IS TO VERIFY THE OPTION OPERATES ACCORDING TO SPECIFICATIONS. THE DIAGNOSTICS ALSO VERIFY THAT THE DZV11 OPERATES IN ITS ENVIRONMENT SUCH AS THE SYSTEM IN WHICH IT IS INSTALLED.

PARAMETERS MAY BE SUPPLIED TO THE PROGRAM BY EITHER 'AUTO SIZING' OR INPUT FROM THE USER ON THE CONSOLE BY HAVING SW00=1 AT START TIME. AUTO SIZING WILL BE DONE ONLY THE FIRST TIME THE PROGRAM IS STARTED AND SW07=0 AND SW00=0 AND SW03=0. THE AUTOSIZER IS DESIGNED TO DETECT DZV11 DEVICE ADDRESSES AND VECTORS ONLY. ALL REMAINING PARAMETERS WILL DEFAULT TO CERTAIN VALUES (SEE SEC.8.5). CONSOLE INPUT MAY BE CONTROLLED AT ANY START TIME THROUGH THE USE OF SW00, SW03, SW04, AND SW06 (SEE SEC. 4.1.1 FOR A DETAILED DESCRIPTION OF THESE SWITCHES).

CURRENTLY THERE ARE THREE STANDALONE DIAGNOSTICS (CVDZA, CVDZB, AND CVDZC) ONE SYSTEM MODULE FOR DEC X/11 (CXDZBA), AND AN OVERLAY FOR ITEP (CVDZD).

CVDZA TOGETHER WITH CVDZB WILL TEST ALL LOGICAL FUNCTIONS OF THE DZV11 INTERFACE MODULE.

CVDZC IS DESIGNED AS A NON-CHAINABLE STANDALONE DIAGNOSTIC PROVIDING THE OPERATOR WITH DIRECT CONTROL OVER THE TESTING OF ALL DZV11 EIA CABLES.

2. REQUIREMENTS

2.1 EQUIPMENT

AN LSI11 CPU WITH MINIMUM 4K OF MEMORY.

ASR 33 (OR EQUIVALENT FOR CONSOLE)

DZV11 INTERFACE MODULE

H329 STAGGERED TURNAROUND CONNECTOR.

H325 CABLE TURNAROUND CONNECTOR.

NOTE: A STAGGERED TURNAROUND CONNECTOR IS NEEDED IN ORDER TO TEST THE PARITY LOGIC.

80
81
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

2.2 STORAGE

PROGRAM WILL USE ALL 4K OF MEMORY EXCEPT WHERE ABL AND BOOTSTRAP LOADER RESIDE. LOCATION 1500 THRU 1740 ARE ESPECIALLY TO BE NOTED AND TO BE UNTOUCHED BY OPERATOR AFTER PARAMETERS HAVE BEEN INPUT FROM CONSOLE (SW00=1); OR AFTER THE 'AUTO SIZING' HAS BEEN DONE. THESE LOCATIONS MAY BE CHANGED IF THE USER UNDERSTANDS THEIR MEANING AND DIFFERENT PARAMETERS ARE REQUIRED.

3. LOADING PROCEEDURE

3.1 METHOD

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

ABSOLUTE LOADER STARTING ADDRESS *500

MEMORY * SIZE

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

3.1.1 STARTING THE PROCESSOR AT THE ABSOLUTE LOADER STARTING ADDRESS WILL LOAD THE DIAGNOSTIC INTO MEMORY.

117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172

4. STARTING PROCEEDURE

- A. SET SWR TO ZERO FOR 'AUTO SIZING' OR SET SW00=1 FOR USER PARAMETER INPUT FROM CONSOLE TERMINAL. NOTE: LOC. 000176 IS USED AS A SOFTWARE SWITCH REGISTER IN ALL OF THE DZV11 DIAGNOSTICS. (SEE SEC. 4.1) ON THE FIRST STARTUP OF THE DIAGNOSTIC IF SW07=1 AND SW00=0 THE PROGRAM WILL ASSUME THAT THE STATUS TABLE HAS BEEN ALREADY BUILT FROM A PREVIOUS DZV11 DIAGNOSTIC RUN. NOTE: ANY DZV11 DIAGNOSTIC WILL OVERLAY THE STATUS TABLE WHEN LOADED TO PRESERVE ITS CONTENTS AND THUS WILL NOT ALTER A PREVIOUSLY BUILT TABLE.
- B. START THE DIAGNOSTIC AT LOC. 200(8). THE PROGRAM WILL TYPE MAINDEC AND PROGRAM NAMES (IF THIS WAS THE FIRST START UP OF THE PROGRAM) AND ALSO THE FOLLOWING: (ON THE FIRST PROGRAM RUN OR IF PARAMETERS WERE CHANGED)

```
'MAP OF DZV11 STATUS'
1500 160100
1502 000300
1504 000017
1506 017470
1510 000000
```

THE ABOVE IS ONLY AN EXAMPLE! THIS WOULD INDICATE THE STATUS TABLE STARTING AT ADD. 1500 IN THE PROGRAM. THE STATUS TABLE MUST BE VERIFIED BY THE USER IF AUTO SIZING IS DONE. FOR INFORMATION OF STATUS TABLE SEE SECTION 8.4 FOR HELP.

THE PROGRAM WILL TYPE 'RUNNING' AND PROCEED TO RUN THE DIAGNOSTIC.

4.1 CONTROL SWITCH SETTINGS

NOTE: THIS PROGRAM UTILIZES A SOFTWARE SWITCH REGISTER WHICH MAY BE MODIFIED BY CHANGING LOC. 176 OR BY TYPING CONTROL 'G' (^G) ON THE CONSOLE TERMINAL WHILE THE PROGRAM IS RUNNING.

```
SW 15 SET: HALT ON ERROR
SW 14 SET: LOOP ON CURRENT TEST
SW 13 SET: INHIBIT ERROR PRINT OUT
SW 12 SET: INHIBIT **ALL** TYPE OUT/BELL ON ERROR.
SW 11 SET: INHIBIT ITERATIONS. (QUICK PASS)
SW 10 SET: ESCAPE TO NEXT TEST
SW 09 SET: LOOP WITH CURRENT DATA
SW 08 SET: CATCH ERROR AND LOOP ON IT
SW 07 SET: NO AUTO SIZE. IF 1ST START OF PROGRAM AFTER LOADING AND
IF SW00=0 THEN THE PROGRAM WILL ASSUME THAT THE STATUS MAP
HAS BEEN BUILT FROM A PREVIOUS DZV11 DIAGNOSTIC RUN.
SW 06 SET: RESELECT DZV11'S DESIRED ACTIVE
SW 05 SET: RESERVED
SW 04 SET: SELECT DELAY PARAMETER (SEE SEC. 4.1.1)
SW 03 SET: EXTRA PARAMETER INPUT (SEE SEC. 4.1.1)
SW 02 SET: LOCK ON SELECTED TEST
SW 01 SET: RESTART PROGRAM AT SELECTED TEST
SW 00 SET: GET USERS PARAMETERS FROM CONSOLE
```

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

4.1.1 SWITCH REGISTER CONTROL OF PARAMETER INPUT FROM CONSOLE

SW 00 GET USERS PARAMETERS FROM CONSOLE. SETTING THIS SWITCH AT START UP TIME ALLOWS THE USER TO INPUT AT THE CONSOLE TERMINAL THE FOLLOWING PARAMETERS: BASE DEVICE ADDRESS, BASE VECTOR ADDRESS, MODE OF OPERATION (EXTERNAL, INTERNAL, OR STAGGERED), AND THE NUMBER OF DZV11'S THAT ARE RUNNING. USING THIS SWITCH ALONE WILL DEFAULT THE FOLLOWING PARAMETERS: ALL 4 LINES ARE SET TO BE TESTED ON EACH DZV11, THE DEFAULT BAUD RATE IS SET AT 19.2 KBAUD AND THE CHARACTER LENGTH FOR THE MAJORITY OF TESTING IS SET AT EIGHT BITS PER CHARACTER WITH TWO STOP BITS.

SW 03 EXTRA PARAMETER INPUT. SETTING THIS SWITCH AT START UP TIME PROVIDES THE USER WITH THE ABILITY TO SET THE LINES ACTIVE FOR TESTING AND TO SET THE DEFAULT BAUD RATE USED FOR THE MAJORITY OF THE DIAGNOSTIC TESTS. THE DELAY PARAMETER IS AUTOMATICALLY ADJUSTED TO THE BAUD RATE GIVEN BY THE USER.

SW 04 SELECT DELAY PARAMETER. THE DELAY PARAMETER THIS SWITCH CONTROLS DETERMINES THE LENGTH OF TIME THE PROGRAM STALLS WAITING FOR A CHARACTER TO BE COMPLETELY TRANSMITTED OR RECEIVED. THIS DELAY COUNT IS AUTOMATICALLY SET TO PROVIDE ENOUGH DELAY TIME FOR THE DEFAULT BAUD RATE SPECIFIED WHEN RUNNING THE PROGRAM ON AN LS111 WITH MOS MEMORY. WHEN RUNNING THIS PROGRAM ON A PROCESSOR WITH A FASTER MEMORY SPEED THIS DELAY COUNT SHOULD BE ADJUSTED PROPORTIONATELY HIGHER THAN THE FOLLOWING DEFAULTED VALUES:

2450	:TIME FOR	50 BAUD
1560	:TIME FOR	75 BAUD
1120	:TIME FOR	110 BAUD
0750	:TIME FOR	134 BAUD
0660	:TIME FOR	150 BAUD
0330	:TIME FOR	300 BAUD
0150	:TIME FOR	600 BAUD
0060	:TIME FOR	1200 BAUD
0040	:TIME FOR	1800 BAUD
0030	:TIME FOR	2000 BAUD
0020	:TIME FOR	2400 BAUD
0010	:TIME FOR	3600 BAUD
0001	:TIME FOR	4800 BAUD
0001	:TIME FOR	7200 BAUD
0001	:TIME FOR	9600 BAUD
0001	:TIME FOR	19.2 KBAUD

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

4.1.2 SWITCH REGISTER RESTRICTIONS

- SW 06 RESELECT DZV11'S DESIRED ACTIVE. A MESSAGE IS TYPED OUT ON THE CONSOLE TERMINAL ASKING THE OPERATOR TO TYPE A BIT MAP OF THE DZV'S DESIRED ACTIVE. USING THIS SWITCH ALLOWS LOCATION DZVACTV TO BE ALTERED (SEE SEC. 8.3 FOR A DESCRIPTION OF THIS LOCATION).
EXAMPLE:
IF THE DEVICES CORRESPONDING TO THE DZV11'S NUMBERED ZERO, TWO, AND FOUR IN THE DZV11 STATUS MAP (LOC. 1500 THROUGH 1740) ARE TO BE TESTED, TYPE IN: 25
THIS WILL SET BITS ZERO, TWO, AND FOUR IN LOCATION DZVACTV. ALL REMAINING DEVICES IN THE STATUS MAP WILL THEN NOT BE TESTED.
- SW 01 RESTART PROGRAM AT SELECTED TEST IT IS STRONGLY SUGGESTED THAT AT LEAST ONE PASS HAS BEEN MADE BEFORE TRYING TO SELECT A THAT THE PROGRAM HAS TO CLEAR AREAS AND SET UP PARAMETERS.
NOTE: IF RUNNING MULTIPLE DZV11'S; THE DZV11 YOU DESIRE TO BE UNDER TEST MUST BE SELECTED BY THE USE OF SW06 BEFORE LOCKING ON THE TEST. IN OTHER WORDS; EACH TIME THE PROGRAM IS STARTED; THE FIRST DZV11 WILL BE SELECTED TO BE UNDER TEST UNLESS SW06 IS USED TO SELECT ONLY ONE.
- SW 09 LOOP ON CURRENT DATA: THIS SWITCH WILL ONLY WORK IF CALL 'SCOP1' IS IN THAT TEST. THE REASON BEING THAT MOST TESTS DEAL WITH BLOCKS OF DIFFERENT DATA TO BE SENT OR RECEIVED ALL AT ONCE THUS IN BLOCK DATA, ONE PATTERN CAN'T BE SINGLED OUT.
THIS SWITCH IS DESIGNED TO PROVIDE AN AID FOR A TRAINED TROUBLE-SHOOTER TO SAMPLE VARIOUS SIGNALS ON THE MODULE AND IS NOT MEANT TO BE USED AS A GENERAL USER CONTROL SWITCH.
- SW 04 SELECT DELAY PARAMETER: THIS SWITCH SHOULD BE USED WITH CARE AS TOO SHORT A DELAY WILL CAUSE VALID TESTS TO FAIL.
(SEE SEC. 4.1.1)

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

4.1.3 SWITCH REGISTER PRIORITIES

ERROR SWITCHES

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

SCOPE SWITCHES

1. SW 09 (IF ENABLED BY 'SCOP1'). IF AN '*' IS PRINTED IN FRONT OF THE TEST NO. ON AN ERROR REPORT (EX. *TEST NO. 10) SW09 IS INCORPORATED IN THAT TEST AND THEREFORE SW09 IS *USUALLY* THE BEST SWITCH FOR THE SCOPE LOOP (SW14=0, SW10=0, SW09=1, SW08=0) IF THE PROGRAM USER IS TECHNICALLY TRAINED TO ELECTRONICALLY ISOLATE SIGNAL PROBLEMS ON THE DZV11 MODULE. IF SW09 IS NOT ENABELED; AND THERE IS A *HARD* ERROR (CONSTANT); SW08 IS BEST.
2. FOR INTERMITTENT ERRORS EITHER START THE PROGRAM WITH SW01 AND SW02 SET WHICH WILL ALLOW THE USER TO LOCK ON A SELECTED TEST, OR ELSE SET SW14 AS AN ERROR IS BEING TYPED OUT ON THE TERMINAL. SW14 WILL CONTINUE TO LOOP ON THAT TEST REGARDLESS OF WHETHER AN ERROR OCCURS.
3. SW 14 LOOP ON CURRENT TEST.

4.2 STARTING ADDRESS

SA 200 - THE STARTING ADDRESS FOR ANY DZV11 DIAGNOSTIC IS LOC. 200

NOTE: IF ADDRESS 000042 IS NON-ZERO THE PROGRAM ASSUMES IT IS UNDER ACT11 OR XXDP CONTROL AND WILL ACT ACCORDINGLY. AFTER *ALL* AVAILABLE DZV11S ARE TESTED THE PROGRAM WILL RETURN TO 'XXDP' OR 'ACT-11'.

5. OPERATING PROCEEDURE

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

5.1 NORMAL START OF DIAGNOSTIC

ON THE FIRST START OF THE DIAGNOSTIC AT ADDRESS 200, IF SW00=1 THEN THE FOLLOWING QUESTIONS ARE ASKED AND MUST BE ANSWERED:

'1ST CSR ADDRESS (160000:163770): ''
YOU MUST TYPE IN THE FIRST DZV11 CSR IN THE SYSTEM YOU WISH TESTING TO BEGIN AT. RANGE: 160000:163770

'1ST VECTOR ADDRESS (300:770): ''
YOU MUST TYPE IN THE VECTOR OF THE FIRST DZV11 IN THE SYSTEM UNDER TEST. RANGE 300:770

'MAINTENANCE MODE
[EXTERNAL <H325> (E)]
[INTERNAL <DZCSR03=1>(I)]
[STAGGERED <H329> (S)] :
TYPE 'E' OR 'I' OR 'S' DEPENDING ON WHICH MODE YOU WISH TO RUN IN. IF RUNNING 'EXTERNAL'; ALL SELECTED LINES MUST BE TERMINATED BY AN H325 TEST CONNECTOR.

'# OF DZV11'S <IN OCTAL> (1:20): ''
TYPE TOTAL NUMBER OF DZV11'S TO BE TESTED IN THE SYSTEM. RANGE IS 1 THRU 20 IN OCTAL.

***** IF SW03=1 THEN THE FOLLOWING WILL BE PRINTED *****

'LINES ACTIVE BY BIT <IN OCTAL> (001:017):''
EACH BIT REPRESENTS A LINE AND ANY COMBINATION OF LINES MAY BE SELECTED (HOWEVER IN STAGGERED MODE TWO ADJACENT LINES MUST BE SELECTED (0-1, 2-3).

'DEFAULT BAUD RATE <IN OCTAL> (00:17): ''
THIS GIVES THE USER A CHANCE TO CHANGE THE DEFAULT BAUD RATE USED IN APP. 90% OF THE TEST. BAUD RATE CHOICES ARE:
'00'(50 BAUD), '01'(75 BAUD), '02'(110 BAUD), '03'(134 BAUD),
'04'(150 BAUD), '05'(300 BAUD), '06'(600 BAUD), '07'(1200 BAUD),
'10'(1800 BAUD), '11'(2000 BAUD), '12'(2400 BAUD), '13'(3600 BAUD),
'14'(4800 BAUD), '15'(7200 BAUD), '16'(9600 BAUD), '17'(19.2 KBAUD)
LOW DEFAULT BAUD RATES ARE NOT SUGGESTED SINCE THEY LENGTHEN THE TIME TO COMPLETE A PROGRAM PASS DRAMATICALLY.

IT IS IMPORTANT TO NOTE THAT ALL DZV11'S IN THE SYSTEM MUST BE CONTIGIOUS FOR BOTH ADDRESS AND VECTORS. ALSO ALL THE EXTRA PARAMETERS OTHER THAN CSR AND VECTORS ARE GIVEN TO THE EXISTING DZV11'S IN THE SYSTEM.

IF THE MODE OF OPERATION IS DIFFERENT FOR EACH DZV11 THIS MUST BE PATCHED INTO THE CORRECT STATUS MAP ENTRY WHICH IS PRINTED AT START TIME. AN ALTERNATIVE IS TO PUT SW00=1 AT START TIME; ANSWER QUESTIONS ABOUT DZV11 UNDER TEST AND INDICATE ONE DZV11 IN THE SYSTEM. IF THE STATUS MAP IS TO BE 'PATCHED' IT MUST BE DONE AFTER THE QUESTIONS ARE ANSWERED OR AFTER THE AUTO SIZE.

300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354

355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398

5.2 PROGRAM AND/OR OPERATOR ACTION

THE VARIETY OF PROGRAM CONTROL SWITCHES PROVIDED IN THIS DIAGNOSTIC PACKAGE IS DESIGNED TO PROVIDE THE USER WITH A WIDE RANGE OF TROUBLE-SHOOTING TECHNIQUES. BEFORE THE USER ATTEMPTS TO RUN THIS DIAGNOSTIC HE SHOULD BECOME FAMILIAR WITH THE USE OF THESE CONTROL SWITCHES AND THEIR RESTRICTIONS. (SEE SEC. 4.1, 4.1.1, 4.1.2, 4.1.3)

WHEN THE PROGRAM DETECTS AN ERROR THE TEST NUMBER AND PC WILL BE TYPED OUT AND POSSIBLY AN ERROR MESSAGE (DEPENDING ON THE PARTICULAR ERROR). IF IT IS NECESSARY TO KNOW MORE INFORMATION CONCERNING THE ERROR REPORT THEN LOOK IN THE PROGRAM LISTING FOR THAT TEST NUMBER AND THEN NOTE THE PC OF THE ERROR REPORT. THE REASON FOR THE ERROR REPORT WILL BECOME CLEARER WHEN READING THE COMMENTS IN THE PROGRAM LISTING.

6. ERRORS

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

6.1 ERROR RECOVERY

IF FOR SOME REASON THE DZV11 SHOULD 'HANG THE BUS' (GAIN CONTROL OF BUS SO THAT CONSOLE MANUAL FUNCTIONS ARE INHIBITED) AN INIT OR POWER DOWN/UP IS NECESSARY FOR OPERATOR TO REGAIN CONTROL OF CPU. IF THIS SHOULD HAPPEN, LOOK IN LOCATION '\$TSTNM' (ADDRESS 1246) FOR THE NUMBER OF THE TEST THAT WAS RUNNING AT THE TIME OF THE CATASTROPHIC ERROR. IN THIS WAY THE OPERATOR WILL HAVE AN IDEA AS TO WHAT THE DZV11 WAS DOING AT THE TIME OF THE ERROR.

7. RESTRICTIONS

7.1 STARTING RESTRICTIONS

SEE SECTION 4.1.2

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

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

7.2 OPERATING RESTRICTIONS

PARAMETER MUST BE INPUT FROM USER OR APT IF 'AUTO SIZING' IS NOT USED.

8. MISCELLANEOUS

8.1 EXECUTION TIME

ALL DZV11 DEVICE DIAGNOSTICS WILL GIVE AN 'END PASS' MESSAGE (PROVIDING NO ERRORS AND SW12=0) WITHIN 2 MIN. THIS IS ASSUMING SW11=1 (INHIBIT ITERATIONS) IS SET TO GIVE THE FASTEST POSSIBLE EXECUTION.

8.2 PASS COMPLETE

NOTE: *EVERY* TIME THE PROGRAM IS STARTED; THE TESTS WILL RUN AS IF SW11 (DELETE ITERATIONS) WAS UP (=1). THIS IS TO 'VERIFY NO *HARD* ERRORS' AS SOON AS POSSIBLE. THEREFORE THE FIRST PASS -EACH TIME PROGRAM IS STARTED- WILL BE A 'QUICK PASS' UNTIL ALL DZV11'S IN SYSTEM ARE TESTED. WHEN THE DIAGNOSTIC HAS COMPLETED A PASS THE FOLLOWING IS AN EXAMPLE OF THE PRINT OUT TO BE EXPECTED.

END PASS CVDZA-B CSR: 160100 VEC: 300 PASSES: 000001 ERRORS: 000000

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

428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
446
447
448
449
450
451
452
453
454
455
456

8.3 KEY LOCATIONS

SLPADR (1252)

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

NEXT (1362)

CONTAINS THE ADDRESS OF THE NEXT TEST TO BE PERFORMED.

STSTNM (1246)

CONTAINS THE NUMBER OF THE TEST NOW BEING PERFORMED.

RUN (1412)

THE BIT IN 'RUN' ALWAYS POINTS ONE PAST THE DZV11 CURRENTLY BEING TESTED. EXAMPLE: (RUN) 1412/0000000001000000 MEANS THAT DZV11 NO.5 IS THE DZV11 NOW RUNNING.

STATUS MAP
(1500)-(1740)

THESE LOCATIONS CONTAIN THE INFORMATION NEEDED TO TEST CSR, VECTOR AND STATUS CONCERNING THE CONFIGURATION OF EACH DZV11.

DZVACTV(1406)

EACH BIT SET IN THIS LOCATION INDICATES THAT THE ASSOCIATED DZV11 WILL BE TESTED IN TURN. EXAMPLE: (DZVACTV) 1406/0000000000011111 MEANS THAT DZV11 NO. 00,01,02,03,04 WILL BE TESTED. EXAMPLE: (DZVACTV) 1406/0000000000010001 MEANS THAT DZV11 NO. 00,04 WILL BE TESTED.

SBASE (1174)

CONTAINS THE RECEIVER CSR OF THE CURRENT DZV11 UNDER TEST.

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
489
490
491
492
493
494
495
496
497

8.4 MORE ON THAT 'STATUS TABLE' (1500-1740)

'MAP OF DZV11 STATUS'

1500	160100
1502	000300
1504	000017
1506	017470
1510	000000

THE ABOVE INFORMATION WILL BE REPEATED FOR EACH OF UP TO 16 DZV11'S IN THE SYSTEM (THESE WILL FOLLOW UNDER THIS TABLE). EXPLANATION:

1500	160100	THIS IS THE SYSTEM CONTROL REGISTER FOR THE 1ST DZV11 IN THE SYSTEM.
1502	000300	THIS IS VECTOR 'A' FOR THE FIRST DZV11 IN THE SYSTEM.
1504	000017	THIS IS THE BINARY REPRESENTATION OF WHAT LINES ARE TO BE TESTED.
1506	017470	THIS IS THE PARAMETER LOCATION USED IN MOST OF THE TESTS. IT INDICATES PARAMETERS OF: RX ON, SPEED SELECT 17 (19.2K BAUD) EIGHT BITS PER CHAR, AND TWO STOP BITS. THE USER MAY ALTER THE STOP BITS AND THE SPEED, BUT THE REMAINING PARAMETERS SHOULD BE LEFT ALONE.
		THIS LOCATION IS USED TO LOAD THE DZV11 LINE PARAMETER REGISTER FOR EACH LINE. THE MEANING OF THE BITS SET IN THIS LOCATION IS THE SAME AS THE FUNCTION OF THE RELATED BITS IN THE DEVICE LINE PARAMETER REGISTER.
1510	000000	THIS LOCATION WILL CONTAIN EITHER ALL ZEROS INDICATING THAT INTERNAL LOOP WAS SELECTED AS MODE OF OPERATION OR IT WILL CONTAIN 10000 INDICATING THAT 'STAGGERED MODE' WAS SELECTED OR IT WILL CONTAIN 000200 INDICATING THAT 'EXTERNAL' WAS THE MODE SELECTED.

THE ABOVE IS REPEATED FOR EACH DZV11 IN THE SYSTEM. THE TABLE IS FILLED BY AUTO SIZING OR BY THE MANUAL PARAMETER INPUT PROGRAM AS DESCRIBED PREVIOUSLY. ALSO IF DESIRED BY USER; THE LOCATIONS MAY BE ALTERED BY HAND TO SUIT THE SPECIFIC CONFIGURATION.

498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544

8.5 *** METHOD OF AUTO SIZING ***

8.5.1 FINDING THE CONTROL STATUS REGISTER.

THE PROGRAM WILL START AT ADDRESS 160000 AND START 'REFERENCING' THE ADDRESS IN THE POINTER. IF A NON-EX MEMORY TRAP OCCURS, THE POINTER (HOLDING 160000) IS UPDATED BY 10 AND THE ABOVE IS REPEATED UNTIL ADDRESS 163770 IS REACHED. IF A 'BUS REPLY' RESPONSE WAS ISSUED BY THE DZV11 (OR ANY OTHER DEVICE) (NO NDM TRAP), 'MASTER SCAN ENABLE' IS ATTEMPTED TO BE SET AND THE TCR BITS FOR ALL FOUR LINES ARE SET. 'TRDY' IS THEN TESTED TO BE SET AND 'MASTER SCAN ENABLE' IS TESTED TO BE STILL SET. THE DIAGNOSTIC WILL THEN CHECK THAT AT LEAST ONE TCR BIT IS STILL SET. IF ALL OF THE ABOVE WORKED, THIS DEVICE IS ASSUMED TO BE A DZV11. IF ANY OF THE ABOVE FAILED, UPDATING OF THE POINTER IS DONE AND THE NOTE: IF THE PROGRAM DOES NOT FIND YOUR DZV11, SOMETHING IS WRONG AND AUTO SIZING SHOULD NOT BE DONE.

8.5.2 FINDING THE VECTOR

THE VECTOR AREA (ADDRESS 300-776) IS FILLED WITH THE INSTRUCTION IOT AND '+2' (NEXT ADDRESS). BIT14 AND BIT5 (TX INTERRUPT ENABLE AND MSTSCAN ENABLE) ARE SET INTO THE DZVCSR. ALL TCR BITS ARE SET, A DELAY OCCURS, AND IF NO INTERRUPT OCCURS (BECAUSE OF A BAD DZV11) THE PROGRAM ASSUMES VECTOR ADDRESS 300 AND THE PROBLEM SHOULD BE FIXED IN THE DIAGNOSTIC. ONCE THE PROBLEM IS FIXED, THE PROGRAM SHOULD BE SETUP AGAIN TO SET THE CORRECT VECTOR. IF AN INTERRUPT OCCURRED, THE ADDRESS TO WHICH THE DZV11 INTERRUPTED TO IS PICKED UP AND REPORTED AS THE VECTOR. NOTE: IF THE VECTOR REPORTED IS NOT THE VECTOR SET UP BY YOU, THERE IS A PROBLEM AND AUTO SIZING SHOULD NOT BE DONE.

8.5.3 PARAMETER ASSUMPTIONS.

SINCE TOO MUCH HARDWARE WOULD NEED TO BE TURNED ON TO SIZE THE REST OF THE PARAMETERS; THE PROGRAM MUST ASSUME THE REMAINING VARIATIONS. THE RESULT IF NOT TO YOUR SPECIFIC CONFIGURATION MAY BE ALTERED BY HAND. IN THIS WAY 95% OF THE PARAMETER SETUP WAS DONE BY THE PROGRAM AND 5% BY YOU.

THEREFORE:

- 1) ALL FOUR LINES ARE ASSUMED TO BE TESTED.
- 2) DEFAULT BAUD RATE IS SET TO 17 (19.2 KBAUD).
- 3) MODE OF OPERATION IS "INTERNAL MODE".

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

545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593

9.0 RUNNING THE DZV11 DIAGNOSTIC UNDER APT

9.1.1 THE APT INTERFACE

THE DZV DIAGNOSTICS HAVE BEEN DESIGNED TO BE COMPATIBLE WITH THE APT (AUTOMATED PRODUCT TEST) SYSTEM. THE DZV LOGIC TEST DIAGNOSTICS (CVDZA, AND CVDZB) CAN BE RUN AS STANDALONE DIAGNOSTICS OR IN EITHER OF THE APT MODES. CVDZC, HOWEVER IS DESIGNED AS A STANDALONE DIAGNOSTIC ONLY AND REQUIRES DIRECT OPERATOR PARTICIPATION.

9.1.2 SETTING UP THE DIAGNOSTIC USING APT

THE DIAGNOSTIC USES SEVERAL VARIABLES IN THE REGION SUBTITLED "APT MAILBOX-ETABLE". THESE VARIABLES ARE:

\$SWREG -(1142)	USED AS THE SOFTWARE SWITCH REGISTER WHILE RUNNING UNDER APT.
\$VECT1 -(1170)	USED TO SPECIFY THE FIRST VECTOR ADDRESS
\$BASE -(1174)	USED TO INDICATE BOTTOM ADDRESS OF DZV11 UNDER TEST
\$DEVN -(1176)	A BIT MAP REPRESENTING WHICH DZV11'S WILL BE TESTED
\$CDW1 -(1200)	USED TO INDICATE WHICH LINES TO RUN ON ALL DZV11'S
\$CDW2 -(1202)	USED TO INDICATE THE DEFAULT TEST MODE. SET TO 0 FOR INTERNAL TESTING, 200 FOR EXTERNAL LOOP BACK (M325 INSTALLED), OR SET TO 10000 FOR STAGGERED LOOP BACK TESTING (M329 INSTALLED).
\$DDW0 -(1204)	EACH OF THE \$DDW WORDS DESCRIBES THE PARAMETERS (LPR) FOR A PARTICULAR DZV11, GOING UP TO 16 DZV11'S

9.1.3 RUNNING UNDER APT

ALL OF THE VARIABLES MENTIONED IN SECTION 9.1.2 SHOULD BE SET UP PRIOR TO RUNNING THE DIAGNOSTIC UNDER APT.

NOTE

BE SURE \$BASE POINTS TO THE FIRST DZV11 BEFORE RUNNING

BASED ON THESE VALUES, THE DIAGNOSTIC WILL SET UP THE STATUS TABLE. THE USER IS THEN FREE TO MONITOR UNDER APT AS NORMAL.

594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646

10.0 PROGRAM DESCRIPTION

THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC PACKAGE (MAINDEC-11-DZQAC-C3).

INITIAL ADDRESS OF THE STACK POINTER *** 1120 ***

MISCELLANEOUS DEFINITIONS

GENERAL PURPOSE REGISTER DEFINITIONS

PRIORITY LEVEL DEFINITIONS

'SWITCH REGISTER' SWITCH DEFINITIONS

DATA BIT DEFINITIONS (BIT00 TO BIT15)

BASIC 'CPU' TRAP VECTOR ADDRESSES

BITS 15-11=CPU TYPE
 11/04=01,11/05=02,11/20=03,11/40
 11/70=06,PDQ=07,Q=10
 BIT 10=REAL TIME CLOCK
 BIT 9=FLOATING POINT PROCESSOR
 BIT 8=MEMORY MANAGEMENT

MEM.TYPE BYTE — (HIGH BYTE)
 900 NSEC CORE=001
 300 NSEC BIPOLAR=002
 500 NSEC MOS=003

MEM.LAST ADDR.=3 BYTES,THIS WORD AND L

THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS USED IN THE PROGRAM.

THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE I
 NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
 NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS

EM	::POINTS TO THE ERROR MESSAGE
DH	::POINTS TO THE DATA HEADER
DT	::POINTS TO THE DATA
DF	::POINTS TO THE DATA FORMAT

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

INCREMENT THE PASS NUMBER (SPASS)
IF THERES A MONITOR GO TO IT
IF THERE ISN'T JUMP TO CYCLE

THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
AND LOAD THE TEST NUMBER(\$TSTNM) INTO THE DISPLAY REG.(DISPLAY<7
AND LOAD THE ERROR FLAG (\$ERFLG) INTO DISPLAY<15:08>
THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
SW14=1 LOOP ON TEST
SW11=1 INHIBIT ITERATIONS
CALL

SCOPE ;;SCOPE=IOT

ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 B
THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE
NOTE1: \$NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CH
NOTE2: \$FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED
NOTE3: \$FILLC CONTAINS THE CHARACTER TO FILL AFTER.

CALL:

1) USING A TRAP INSTRUCTION
 TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN
OR
 TYPE
 MESADR

ROUTINE USED TO SET UP THE DIAGNOSTIC VIA APT.
IF BIT7 IN THE ENVIRONMENT MODE (\$ENVMD) BYTE IS SET,
THE PROGRAM WILL LOAD ITS PARAMETERS FROM THE ETABLE.

ROUTINE USED TO "AUTO SIZE" THE DZV11
CSR AND VECTOR.

NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
ADDRESS RANGE (160000:163770)
AND THE VECTOR MAY BE ANY WHERE IN THE
FLOATING VECTOR RANGE (300:770)

***** TEST 1 *****
THIS TEST PROVES THE BUS REPLY RESPONSE
DURING A READ OR WRITE TO THE FOLLOWING ADDRESS:
DZVCSR, DZVRBUF, DZVTCR, DZVMSR

***** TEST 2 *****
THIS TEST PROVES THAT BIT "DCLR"
CAN BE SET AND THAT IT WILL CLEAR
BY ITSELF

698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753

***** TEST 3 *****
TEST TO VERIFY THAT THE R/W BITS OF THE
DZVCSR REGISTER CAN BE SET. THEN VERIFY THAT
THESE BITS CAN BE CLEARED. AND FINALLY, VERIFY
THAT AFTER BEING SET AGAIN THEY CAN BE
CLEARED BY A 'DEVICE CLEAR'.
THE BITS TESTED ARE: MAINT, MSENAB, SILOEN,
RIE, AND TIE.

***** TEST 4 *****
THIS TESTS THAT ALL OF THE TCR BITS
CAN BE: SET, CLEARED, AND CLEARED BY A DEVICE CLEAR.
THIS TEST ALSO DETERMINES IF THE DTR BITS CAN
BE SET, CLEARED, AND CLEARED BY A RESET.

***** TEST 5 *****
THIS TEST VERIFIES THAT
BITS 'RDONE, TRDY, BIT9, BIT8,
AND SILOAL' ARE READ ONLY AND THAT TRDY IS
ZERO UNTIL A LINE IS SELECTED AND MSENAB IS SET.

***** TEST 6 *****
THIS TEST VERIFIES THAT:
TIE, SILOEN, RIE, MSENAB, AND MAINT ARE THE
ONLY R/W BITS IN THE DZVCSR AND THAT
SETTING 'DCLR' IN THE CSR WILL CLEAR THESE BITS.

***** TEST 7 *****
THIS TEST PERFORMS RESET TESTING AND
TESTING OF READ ONLY REGISTER DZVRBUF
AND TESTING OF WRITE ONLY REGISTER DZVLPR

***** TEST 10 *****
THIS TEST PERFORMS RESET TESTING AND
TESTING OF READ ONLY REGISTER DZVMSR
AND TESTING OF WRITE ONLY REGISTER DZVTDR

***** TEST 11 *****
VERIFY THAT SETTING 'DTR' FOR A LINE WILL
BRING UP 'CO' AND 'RING' FOR:
THE SAME LINE IF IN EXTERNAL MODE
THE STAGGERED LINE IF IN STAGGERED MODE.
LINES ARE STAGGERED AS FOLLOWS:
LINE0 WITH LINE1; LINE2 WITH LINE3.
THIS TEST IS ONLY RUN IF AN H325, OR H329
IS CONNECTED ON THE DZV UNDER TEST.

***** TEST 12 *****
THIS TEST VERIFIES THAT TRDY IS SET WHEN A LINE
IS READY TO BE LOADED, AND THAT THE LINE SPECI-
FIED IN BITS 8-9 OF DZVCSR CORRESPOND
TO THE LINE SELECTED IN DZVTCR

754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803

***** TEST 13 *****
TEST TO TRANSMIT ONE CHAR AND
RECEIVE ONE CHAR ON ONE LINE
AT A TIME. THE CHAR IS '252' AND
ALL SELECTED LINES WILL BE TURNED ON .

THIS IS THE FIRST TIME ANY
DATA IS CHECKED IN THE RECEIVER.
USING SWITCH NINE WITH THIS TEST CREATES A TIGHT SCOPE LOOP
WHICH TRANSMITS A STEADY STREAM OF CHARACTERS.

***** TEST 14 *****
THIS TEST VERIFIES THAT EACH RECEIVING LINE CAN BE
DISABLED BY SETTING RCVON (BIT12 IN THE LPR REGISTER)
TO ZERO FOR EACH LINE.
THIS TEST ALSO VERIFIES THAT THE SILO CAN BE
EMPTIED BY ISSUING A DEVICE MASTER CLEAR.

***** TEST 15 *****
THIS TEST PROVES THAT THE TRANSMITTER TRANSMITS
CHARACTERS (FLAG MODE) AND THE RECEIVER RECEIVES (FLAG MODE)
(ONE LINE AT A TIME BASED UPON VALID LINES)
THIS IS THE FIRST TIME THAT ALL DATA IS CHECKED

***** TEST 16 *****
THIS TEST WILL PROVE THAT:
1) THE TRANSMITTER 'BREAK BIT' WORKS
2) THE RECEIVER CAN FLAG 'FRAMING ERRORS'
3) THE RECEIVER CAN FLAG 'PARITY ERRORS'
ONLY ONE LINE AT A TIME WILL BE EXERCISED.

***** TEST 17 *****
THIS TEST VERIFIES THAT THE DEVICE DOES NOT INTERRUPT
WHILE THE PROCESSOR STATUS DOES NOT ALLOW INTERRUPTS
BUT WILL INTERRUPT IF THE PROCESSOR STATUS
ALLOWS INTERRUPTS.

***** TEST 20 *****
THIS TEST VERIFIES THAT THE RECEIVER WILL
INTERRUPT BEFORE THE TRANSMITTER EVEN
THOUGH THE TRANSMITTER WAS ENABLED
FIRST. SET PS TO HIGH (MASK INTERRUPTS);
GET RDONE AND TRDY TO SET;
SET TX IE AND RX IE;
CLEAR PS AND EXPECT RX TO INTERRUPT FIRST

```

804      .TITLE CVDZA-B
805      :*COPYRIGHT (C) 1977,1981
806      :*DIGITAL EQUIPMENT CORP.
807      :*MAYNARD, MASS. 01754
808      :*
809      :*
810      :*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
811      :*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
812
813      000001      $TN=1
814                  ;STARTING PROCEDURE
815                  ;LOAD PROGRAM
816                  ;LOAD ADDRESS 000200
817                  ;PRESS START
818                  ;PROGRAM WILL TYPE
819                  ;'CVDZAB/<200>/FOUR LINE ASYNC MUX TESTS, PART 1 OF 2''
820                  ;PROGRAM WILL TYPE 'RUNNING' TO INDICATE THAT TESTING HAS STARTED
821                  ;AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE
822                  ;AND THEN RESUME TESTING
823
824      .REM      !
825      ;SWITCH REGISTER OPTIONS
826      ;-----
827
828
829      SW15=100000      ;=1,HALT ON ERROR
830      SW14=40000      ;=1,LOOP ON CURRENT TEST
831      SW13=20000      ;=1,INHIBIT ERROR TYPEOUT
832      SW12=10000      ;=1,DELETE TYPEOUT/BELL ON ERROR.
833      SW11=4000      ;=1,INHIBIT ITERATIONS
834      SW10=2000      ;=1,ESCAPE TO NEXT TEST ON ERROR
835      SW09=1000      ;=1,LOOP WITH CURRENT DATA
836      SW08=400      ;=1,LOOP ON ERROR
837      SW07=200      ;=1, DO 'AUTO SIZING' ON INITIAL START UP.
838      SW06=100      ;=1, DESELECT SPECIFIC DEVICES
839                  ;NOTE: THIS MUST NOT EXCEED ORIGINAL COUNT
840
841      SW05=40
842      SW04=20      ;=1, SELECT DELAY PARAMETER
843      SW03=10      ;=1, SELECT SPECIFIC PARAMETERS
844      SW02=4      ;=1, LOCK ON TEST SELECT
845      SW01=2      ;=1, RESTART PROGRAM AT SELECTED TEST
846      SW00=1      ;=1, SELECT DEVICE ADDRESS, VECTOR, ETC.
847      !
848      .SBTTL BASIC DEFINITIONS
849
850      ;*INITIAL ADDRESS OF THE STACK POINTER *** 1120 ***
851      001120      STACK= 1120
852      .EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
853      .EQUIV IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
854
855      ;*MISCELLANEOUS DEFINITIONS
856      000011      HT= 11      ;;CODE FOR HORIZONTAL TAB
857      000012      LF= 12      ;;CODE FOR LINE FEED
858      000015      CR= 15      ;;CODE FOR CARRIAGE RETURN
859      000200      CRLF= 200      ;;CODE FOR CARRIAGE RETURN-LINE FEED
860      177776      PS= 177776      ;;PROCESSOR STATUS WORD

```

```

860      .EQUIV PS,PSW
861      177774      STKLMT= 177774      ::STACK LIMIT REGISTER
862      177772      PIRQ= 177772      ::PROGRAM INTERRUPT REQUEST REGISTER
863      177570      DSWR= 177570      ::HARDWARE SWITCH REGISTER
864      177570      DDISP= 177570      ::HARDWARE DISPLAY REGISTER
865
866      ;*GENERAL PURPOSE REGISTER DEFINITIONS
867      000000      R0= %0      ::GENERAL REGISTER
868      000001      R1= %1      ::GENERAL REGISTER
869      000002      R2= %2      ::GENERAL REGISTER
870      000003      R3= %3      ::GENERAL REGISTER
871      000004      R4= %4      ::GENERAL REGISTER
872      000005      R5= %5      ::GENERAL REGISTER
873      000006      R6= %6      ::GENERAL REGISTER
874      000007      R7= %7      ::GENERAL REGISTER
875      000006      SP= %6      ::STACK POINTER
876      000007      PC= %7      ::PROGRAM COUNTER
877
878      ;*PRIORITY LEVEL DEFINITIONS
879      000000      PR0= 0      ::PRIORITY LEVEL 0
880      000040      PR1= 40      ::PRIORITY LEVEL 1
881      000100      PR2= 100      ::PRIORITY LEVEL 2
882      000140      PR3= 140      ::PRIORITY LEVEL 3
883      000200      PR4= 200      ::PRIORITY LEVEL 4
884      000240      PR5= 240      ::PRIORITY LEVEL 5
885      000300      PR6= 300      ::PRIORITY LEVEL 6
886      000340      PR7= 340      ::PRIORITY LEVEL 7
887
888      ;*'SWITCH REGISTER' SWITCH DEFINITIONS
889      100000      SW15= 100000
890      040000      SW14= 40000
891      020000      SW13= 20000
892      010000      SW12= 10000
893      004000      SW11= 4000
894      002000      SW10= 2000
895      001000      SW09= 1000
896      000400      SW08= 400
897      000200      SW07= 200
898      000100      SW06= 100
899      000040      SW05= 40
900      000020      SW04= 20
901      000010      SW03= 10
902      000004      SW02= 4
903      000002      SW01= 2
904      000001      SW00= 1
905      .EQUIV SW09,SW9
906      .EQUIV SW08,SW8
907      .EQUIV SW07,SW7
908      .EQUIV SW06,SW6
909      .EQUIV SW05,SW5
910      .EQUIV SW04,SW4
911      .EQUIV SW03,SW3
912      .EQUIV SW02,SW2
913      .EQUIV SW01,SW1
914      .EQUIV SW00,SW0
915

```

```

916          ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
917          100000 BIT15= 100000
918          040000 BIT14= 40000
919          020000 BIT13= 20000
920          010000 BIT12= 10000
921          004000 BIT11= 4000
922          002000 BIT10= 2000
923          001000 BIT09= 1000
924          000400 BIT08= 400
925          000200 BIT07= 200
926          000100 BIT06= 100
927          000040 BIT05= 40
928          000020 BIT04= 20
929          000010 BIT03= 10
930          000004 BIT02= 4
931          000002 BIT01= 2
932          000001 BIT00= 1
933          .EQUIV BIT09,BIT9
934          .EQUIV BIT08,BIT8
935          .EQUIV BIT07,BIT7
936          .EQUIV BIT06,BIT6
937          .EQUIV BIT05,BIT5
938          .EQUIV BIT04,BIT4
939          .EQUIV BIT03,BIT3
940          .EQUIV BIT02,BIT2
941          .EQUIV BIT01,BIT1
942          .EQUIV BIT00,BIT0
943
944          ;*BASIC "CPU" TRAP VECTOR ADDRESSES
945          000004 ERRVEC= 4          ;;TIME OUT AND OTHER ERRORS
946          000010 RESVEC= 10        ;;RESERVED AND ILLEGAL INSTRUCTIONS
947          000014 TBITVEC=14        ;;'T' BIT
948          000014 TRTVEC= 14        ;;TRACE TRAP
949          000014 BPTVEC= 14        ;;BREAKPOINT TRAP (BPT)
950          000020 IOTVEC= 20        ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
951          000024 PWRVEC= 24        ;;POWER FAIL
952          000030 EMTVEC= 30        ;;EMULATOR TRAP (EMT) **ERROR**
953          000034 TRAPVEC=34        ;;'TRAP' TRAP
954          000060 TKVEC= 60         ;;TTY KEYBOARD VECTOR
955          000064 TPVEC= 64         ;;TTY PRINTER VECTOR
956          000240 PIRQVEC=240       ;;PROGRAM INTERRUPT REQUEST VECTOR
957
958          ;INSTRUCTION DEFINITIONS
959          -----
960
961
962          005746 PUSH1SP=5746       ;;DECREMENT PROCESSOR STACK 1 WORD
963          005726 POP1SP=5726       ;;INCREMENT PROCESSOR STACK 1 WORD
964          010046 PUSHRO=10046      ;;SAVE R0 ON STACK
965          012600 POPRO=12600       ;;RESTORE R0 FROM STACK
966          024646 PUSH2SP=24646    ;;DECREMENT STACK TWICE
967          022626 POP2SP=22626     ;;INCREMENT STACK TWICE
968          000200 MASK=BIT7        ;;SET INTERRUPT MASK (INHIBIT FURTHER INTERRUPTS)
969          000000 CLEAR=0           ;;ALLOW INTERRUPTS (CLEAR PROCESSOR STATUS)
970
971

```

```

972                                     :DZV11 CONTROL AND STATUS REGISTER DEFINITIONS
973                                     :(DZVCSR)      BIT DEFINITIONS
974                                     -----
975
976      000010      MAINT = BIT3      :MAINTENANCE MODE ENABLE
977      000020      DCLR=BIT4      :DEVICE CLEAR
978      000040      MSENAB=BIT5     :MASTER SCAN ENABLE
979      000100      RIE=BIT6       :RECEIVER INTERRUPT ENABLE
980      000200      RDONE=BIT7     :RECEIVER DONE
981      010000      SILOEN= BIT12   :SILO ALARM ENABLE
982      020000      SILOAL = BIT13  :SILO ALARM
983      040000      TIE=BIT14      :TRANSMITTER INTERRUPT ENABLE
984      100000      TRDY=BIT15     :TRANSMITTER READY
985
986                                     :DZVCSR WORD DEFINITIONS
987                                     -----
988      000000      TL0=0           :TRANSMIT LINE 0
989      000400      TL1=BIT8       :TRANSMIT LINE 1
990      001000      TL2=BIT9       :TRANSMIT LINE 2
991      001400      TL3=BIT9!BIT8  :TRANSMIT LINE 3
992
993                                     :DZVRBUF BIT DEFINITIONS
994                                     -----
995
996      010000      PARER=BIT12     :PARITY ERROR
997      020000      FRMERR=BIT13   :FRAME ERROR
998      040000      OVRRUN=BIT14   :OVERRUN ERROR
999      100000      DVALID=BIT15   :DATA VALID
1000
1001                                     :DZVRBUF WORD DEFINITIONS
1002                                     -----
1003
1004      000000      RL0=0           :RECEIVER LINE 0
1005      000400      RL1=BIT8       :RECEIVER LINE 1
1006      001000      RL2=BIT9       :RECEIVER LINE 2
1007      001400      RL3=BIT9!BIT8  :RECEIVER LINE 3
1008
1009                                     :DZVLPR WORD DEFINITIONS
1010                                     -----
1011
1012      000000      LP0=0           :LINE PARAMETER 0
1013      000001      LP1=BIT0       :LINE PARAMETER 1
1014      000002      LP2=BIT1       :LINE PARAMETER 2
1015      000003      LP3=BIT1!BIT0  :LINE PARAMETER 3
1016
1017      000000      FIVE=0          :FIVE BITS/CHAR,1 STOP BIT
1018      000010      SIX=BIT3       :SIX BITS/CHAR,1 STOP BIT
1019      000020      SEVEN=BIT4     :SEVEN BITS/CHAR,1 STOP BIT
1020      000030      EIGHT=BIT4!BIT3 :EIGHT BITS/CHAR,1 STOP BIT
1021      000040      FIVES=BIT5     :FIVE BITS/CHAR,2 STOP BITS
1022      000050      SIXS=BIT5!BIT3 :SIX BITS/CHAR,2 STOP BITS
1023      000060      SEVENS=BIT5!BIT4 :SEVEN BITS/CHAR, 2 STOP BITS
1024      000070      EIGHTS=BIT5!BIT4!BIT3 :EIGHT BITS/CHAR, 2 STOP BITS
1025
1026      000100      PARITY=BIT6     :PARITY ENABLED
1027

```

```

1028      000200      ODDPAR=BIT7           :ODD PARITY ENABLED
1029      000000      ONESTOP=0             :ONE STOP BIT ENABLED
1030      000040      TWOSTOP=BIT5        :TWO STOP BITS ENABLED
1031      000000      EVEPAR=0            :EVEN PARITY ENABLED
1032      010000      RCVON=BIT12         :ENABLE RECEIVER (RECEIVER ON)
1033
1034      000000      S50=0              :SPEED 50 BAUD
1035      000400      S75=BIT8           :SPEED 75 BAUD
1036      001000      S110=BIT9          :SPEED 110 BAUD
1037      001400      S134=BIT9!BIT8     :SPEED 134.5 BAUD
1038      002000      S150=BIT10         :SPEED 150 BAUD
1039      002400      S300=BIT10!BIT8    :SPEED 300 BAUD
1040      003000      S600=BIT10!BIT9    :SPEED 600 BAUD
1041      003400      S1200=BIT10!BIT9!BIT8 :SPEED 1200 BAUD
1042      004000      S1800=BIT11        :SPEED 1800 BAUD
1043      004400      S2000=BIT11!BIT8   :SPEED 2000 BAUD
1044      005000      S2400=BIT11!BIT9   :SPEED 2400 BAUD
1045      005400      S3600=BIT11!BIT9!BIT8 :SPEED 3600 BAUD
1046      006000      S4800=BIT11!BIT10  :SPEED 4800 BAUD
1047      006400      S7200=BIT11!BIT10!BIT8 :SPEED 7200 BAUD
1048      007000      S9600=BIT11!BIT10!BIT9 :SPEED 9600 BAUD
1049      007400      S19200=BIT11!BIT10!BIT9!BIT8 :SPEED 19200 BAUD
1050
1051      :DZVTCR BIT DEFINITIONS
1052      :-----
1053      000001      TCR0=BIT0           :ENABLE TRANSMISSION ON LINE 0
1054      000002      TCR1=BIT1           :ENABLE TRANSMISSION ON LINE 1
1055      000004      TCR2=BIT2           :ENABLE TRANSMISSION ON LINE 2
1056      000010      TCR3=BIT3           :ENABLE TRANSMISSION ON LINE 3
1057      000400      DTR0=BIT8           :DATA TERMINAL READY FOR LINE 0
1058      001000      DTR1=BIT9           :DATA TERMINAL READY FOR LINE 1
1059      002000      DTR2=BIT10          :DATA TERMINAL READY FOR LINE 2
1060      004000      DTR3=BIT11          :DATA TERMINAL READY FOR LINE 3
1061
1062      :DZVMSR BIT DEFINITIONS
1063      :-----
1064      000001      RING0=BIT0          :RING INDICATED ON LINE 0
1065      000002      RING1=BIT1          :RING INDICATED ON LINE 1
1066      000004      RING2=BIT2          :RING INDICATED ON LINE 2
1067      000010      RING3=BIT3          :RING INDICATED ON LINE 3
1068      000400      CO0=BIT8            :CARRIER PRESENT ON LINE 0
1069      001000      CO1=BIT9            :CARRIER PRESENT ON LINE 1
1070      002000      CO2=BIT10           :CARRIER PRESENT ON LINE 2
1071      004000      CO3=BIT11           :CARRIER PRESENT ON LINE 3
1072
1073      :DZVTDR BIT DEFINITIONS
1074      :-----
1075
1076      000400      BRK0=BIT8            :BREAK FOR LINE 0
1077      001000      BRK1=BIT9            :BREAK FOR LINE 1
1078      002000      BRK2=BIT10           :BREAK FOR LINE 2
1079      004000      BRK3=BIT11           :BREAK FOR LINE 3

```


1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097

:TABLE OF LOOP AROUND FUNCTIONS (H325)

I	^
V	^
REC DATA	TRANS DATA

I	^
V	^
CO	RTS

I	^
V	^
RING	DTR

```

1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
(2)

```

```

:*****
:-----
:TRAPCATCHER FOR ILLEGAL INTERRUPTS
:THE STANDARD 'TRAP CATCHER' IS PLACED
:BEWEEN ADDRESS 0 TO ADDRESS 776.
:IT LOOKS LIKE 'PC+2 HALT'.
:-----
:*****
.=0
:STANDARD INTERRUPT VECTORS
:-----
.=20
.SCOPE
MASK
$PWRDN
340
$ERROR
340
.TRPSRV
340
:SCOPE LOOP HANDLER
:HANDLE AT PRIORITY 7
:POWER FAIL HANDLER
:SERVICE AT PRIORITY LEVEL 7
:ERROR HANDLER
:SERVICE AT PRIORITY LEVEL 7
:GENERAL HANDLER DISPATCH SERVICE
:SERVICE AT PRIORITY LEVEL 7
.SBTTL ACT11 HOOKS
:*****
:HOOKS REQUIRED BY ACT11
$SVPC=.
.=46
$ENDAD
.=52
.WORD 0
.= $SVPC
;SAVE PC
;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
;;2)SET LOC.52 TO ZERO
;; RESTORE PC
.=174
DISPREG:0
SWREG: 0
.=200
JMP .START
;SOFTWARE DISPLAY REGISTER FOR SWITCHLESS 11S
;SOFTWARE SWITCH REGISTER FOR SWITCHLESS 11S
;GO TO START OF PROGRAM
.=1000
MTITLE: .ASCIZ <200><12>/CVDZAB/<200>/FOUR LINE ASYNC MUX TESTS, PART 1 OF 2/<200>

```


1196 001220 017470
1197 001222 017470
1198 001224 017470
1199 001226 017470
1200 001230 017470
1201 001232 017470
1202 001234 017470
1203 001236 017470
1204 001240 017470
1205 001242 017470
1206
1207
1208 001244
1209

SDDW6: .WORD ADDW6 ::DEVICE DESCRIPTOR WORD#6
SDDW7: .WORD ADDW7 ::DEVICE DESCRIPTOR WORD#7
SDDW8: .WORD ADDW8 ::DEVICE DESCRIPTOR WORD#8
SDDW9: .WORD ADDW9 ::DEVICE DESCRIPTOR WORD#9
SDDW10: .WORD ADDW10 ::DEVICE DESCRIPTOR WORD#10
SDDW11: .WORD ADDW11 ::DEVICE DESCRIPTOR WORD#11
SDDW12: .WORD ADDW12 ::DEVICE DESCRIPTOR WORD#12
SDDW13: .WORD ADDW13 ::DEVICE DESCRIPTOR WORD#13
SDDW14: .WORD ADDW14 ::DEVICE DESCRIPTOR WORD#14
SDDW15: .WORD ADDW15 ::DEVICE DESCRIPTOR WORD#15

BETEND:

Address	Label	Value	Format	Value	Description
1210	.SBTTL COMMON TAGS				
1211	:*****				
1212	: *THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS				
1213	: *USED IN THE PROGRAM.				
1214					
1215					
1216	001244	000000	SCMTAG:	0	:: START OF COMMON TAGS
1217	001244	000	\$STNM:	.BYTE 0	:: CONTAINS THE TEST NUMBER
1218	001246	000	\$ERFLG:	.BYTE 0	:: CONTAINS ERROR FLAG
1219	001247	000	\$ICNT:	.WORD 0	:: CONTAINS SUBTEST ITERATION COUNT
1220	001250	000000	\$LPADR:	.WORD 0	:: CONTAINS SCOPE LOOP ADDRESS
1221	001252	000000	\$LPERR:	.WORD 0	:: CONTAINS SCOPE RETURN FOR ERRORS
1222	001254	000000	\$ERTTL:	.WORD 0	:: CONTAINS TOTAL ERRORS DETECTED
1223	001256	000000	\$ITEMB:	.BYTE 0	:: CONTAINS ITEM CONTROL BYTE
1224	001260	000	\$ERMAX:	.BYTE 1	:: CONTAINS MAX. ERRORS PER TEST
1225	001261	001	\$ERRPC:	.WORD 0	:: CONTAINS PC OF LAST ERROR INSTRUCTION
1226	001262	000000	\$GDADR:	.WORD 0	:: CONTAINS ADDRESS OF 'GOOD' DATA
1227	001264	000000	\$BDADR:	.WORD 0	:: CONTAINS ADDRESS OF 'BAD' DATA
1228	001266	000000	\$GDDAT:	.WORD 0	:: CONTAINS 'GOOD' DATA
1229	001270	000000	\$BDDAT:	.WORD 0	:: CONTAINS 'BAD' DATA
1230	001272	000000		.WORD 0	:: RESERVED—NOT TO BE USED
1231	001274	000000		.WORD 0	
1232	001276	000000		.WORD 0	
1233	001300	000	\$AUTOB:	.BYTE 0	:: AUTOMATIC MODE INDICATOR
1234	001301	000	\$INTAG:	.BYTE 0	:: INTERRUPT MODE INDICATOR
1235	001302	000000		.WORD 0	
1236	001304	177570	\$SWR:	.WORD DSWR	:: ADDRESS OF SWITCH REGISTER
1237	001306	177570	\$DISPLAY:	.WORD DDISP	:: ADDRESS OF DISPLAY REGISTER
1238	001310	177560	\$TKS:	177560	:: TTY KBD STATUS
1239	001312	177562	\$TKB:	177562	:: TTY KBD BUFFER
1240	001314	177564	\$TPS:	177564	:: TTY PRINTER STATUS REG. ADDRESS
1241	001316	177566	\$TPB:	177566	:: TTY PRINTER BUFFER REG. ADDRESS
1242	001320	000	\$NULL:	.BYTE 0	:: CONTAINS NULL CHARACTER FOR FILLS
1243	001321	002	\$FILLS:	.BYTE 2	:: CONTAINS # OF FILLER CHARACTERS REQUIRED
1244	001322	012	\$FILLC:	.BYTE 12	:: INSERT FILL CHARS. AFTER A 'LINE FEED'
1245	001323	000	\$TPFLG:	.BYTE 0	:: 'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
1246	001324	000000	\$REGAD:	.WORD 0	:: CONTAINS THE ADDRESS FROM WHICH (\$REGO) WAS OBTAINED
1247					
1248	001326	000000	\$REG0:	.WORD 0	:: CONTAINS ((\$REGAD)+0)
1249	001330	000000	\$REG1:	.WORD 0	:: CONTAINS ((\$REGAD)+2)
1250	001332	000000	\$REG2:	.WORD 0	:: CONTAINS ((\$REGAD)+4)
1251	001334	000000	\$REG3:	.WORD 0	:: CONTAINS ((\$REGAD)+6)
1252	001336	000000	\$REG4:	.WORD 0	:: CONTAINS ((\$REGAD)+10)
1253	001340	000000	\$REG5:	.WORD 0	:: CONTAINS ((\$REGAD)+12)
1254	001342	000000	\$TMP0:	.WORD 0	:: USER DEFINED
1255	001344	000000	\$TMP1:	.WORD 0	:: USER DEFINED
1256	001346	000000	\$TMP2:	.WORD 0	:: USER DEFINED
1257	001350	000000	\$TMP3:	.WORD 0	:: USER DEFINED
1258	001352	000000	\$TMP4:	.WORD 0	:: USER DEFINED
1259	001354	000000	\$TIMES:	0	:: MAX. NUMBER OF ITERATIONS
1260	001356	077	\$QUES:	.ASCII /?/	:: QUESTION MARK
1261	001357	015	\$CRLF:	.ASCII <15>	:: CARRIAGE RETURN
1262	001360	000012	\$LF:	.ASCII <12>	:: LINE FEED

```

1263 .SBTTL ERROR POINTER TABLE
1264
1265 : *THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
1266 : *THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
1267 : *LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
1268 : *NOTE1: IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
1269 : *NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
1270
1271 : * EM ;;POINTS TO THE ERROR MESSAGE
1272 : * DH ;;POINTS TO THE DATA HEADER
1273 : * DT ;;POINTS TO THE DATA
1274 : * DF ;;POINTS TO THE DATA FORMAT
1275
1276
1277 001362 $ERRTB:
1278
1279 :PROGRAM CONTROL PARAMETERS
1280 :-----
1281
1282 001362 000000 NEXT: 0 ;ADDRESS OF NEXT TEST TO BE EXECUTED
1283 001364 000000 LOCK: 0 ;ADDRESS FOR LOCK ON CURRENT TEST,TIGHT LOOP
1284
1285 :PROGRAM VARIABLES
1286 :-----
1287
1288 001366 000017 LINE: 17 ;DEFAULT ALL FOUR LINES RUNNING
1289 001370 017470 PAR: 17470 ;PARAMETERS: 8 BITS/CHAR,2 STOP BITS,19200 BAUD,NO PARIT
1290 001372 000000 MODE: 0 ;DEFAULT MAINTENANCE MODE
1291 001374 000000 SAVLIN: 0 ;LINE NUMBER
1292 001376 000000 XMITLIN: 0 ;TRANSMISSION LINE NUMBER
1293 001400 000000 XMITCNT: 0 ;COUNT OF WORDS IN A TRANSMISSION PATTERN
1294 001402 000000 REGIST: 0 ;DEVICE ADDRESS STORAGE LOCATION
1295 001404 000000 SAVPC: 0 ;PROGRAM COUNTER STORAGE
1296 001406 000001 DZVACTV: .BLKW 1 ;*DZV11'S SELECTED ACTIVE.
1297 001410 000001 SAVACTV: .BLKW 1 ;*A BIT MAP OF DZV11'S IN THE SYSTEM
1298 001412 000001 RUN: 1 ;*POINTER ONE PAST RUNNING DEVICE.
1299 001414 000001 DZVNUM: .BLKB 1 ;*OCTAL NUMBER OF DZV11'S IN THE SYSTEM
1300 001415 001 SAVNUM: .BYTE 1 ;*WORKABLE NUMBER.
1301 001416 000001 SAVNO: .BLKB 1 ;*OCTAL NO. OF DZV11'S BEING TESTED
1302 001420 .EVEN
1303 001420 001500 ACTIVE: DZV.MAP ;TABLE POINTER.

```

```

1304
1305
1306          :PROGRAM CONTROL FLAGS
1307          :-----
1308 001422    000      INIFLG: .BYTE 0          ;PROGRAM INITIALIZATION FLAG
1309 001423    000      HDRFLG: .BYTE 0          ;PROGRAM INITIALIZATION FLAG FOR HEADER MAP
1310 001424    000      MNTFLG: .BYTE 0          ;MAINTENANCE BIT SET FLAG
1311 001425    000      DONFLG: .BYTE 0          ;TRANSMISSION COMPLETION FLAG
1312
1313          .EVEN
1313          :DATA VARIABLES
1314 001426    000000    TD0:      .WORD 0
1315 001430    000000    TD1:      .WORD 0
1316 001432    000000    TD2:      .WORD 0
1317 001434    000000    TD3:      .WORD 0
1318 001436    000000    TR0:      .WORD 0
1319 001440    000000    TR1:      .WORD 0
1320 001442    000000    TR2:      .WORD 0
1321 001444    000000    TR3:      .WORD 0
1322 001446
1323          STOP:
1324          .SBTTL  APT PARAMETER BLOCK
1325
1326          :*****
1326          :SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1327          :*****
1328          001446      .SX=.      ;;SAVE CURRENT LOCATION
1329          000024      =24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1330 000024    000200    200      ;;FOR APT START UP
1331          000044      =44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
1332 000044    001446    $APTHDR  ;;POINT TO APT HEADER BLOCK
1333          001446      =.SX      ;;RESET LOCATION COUNTER
1334          :*****
1335          :SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1336          :INTERFACE SPEC.
1337
1338          $APTHD:
1339 001446    000000    $HIBTS: .WORD 0          ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1340 001450    001120    $MBADR: .WORD $MAIL      ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1341 001452    000120    $STMT:  .WORD 80.       ;;RUN TIM OF LONGEST TEST
1342 001454    000024    $PASTM: .WORD 20.      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1343 001456    000000    $UNITM: .WORD 0.       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1344 001460    000052    .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
1345
1346          :DZV11 STATUS TABLE AND ADDRESS ASSIGNMENTS
1347          :-----
1348          =1500
1349 001500    001500    DZV.MAP:
1350
1351 001500    000001    DZCR0: .BLKW 1          ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 0
1352 001502    000001    DZVC0: .BLKW 1          ;RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 0
1353 001504    000001    LINE0: .BLKW 1          ;ALL LINES SELECTED
1354 001506    000001    PAR0:  .BLKW 1          ;PARAMETERS
1355 001510    000001    MANT0: .BLKW 1          ;MAINTENANCE MODE FOR THIS DEVICE
1356
1357 001512    000001    DZCR1: .BLKW 1          ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 1
1358 001514    000001    DZVC1: .BLKW 1          ;RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 1
1359 001516    000001    LINE1: .BLKW 1          ;ALL LINES SELECTED

```

1360	001520	000001	PAR1:	.BLKW	1	:PARAMETERS
1361	001522	000001	MANT1:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
1362						
1363	001524	000001	DZCR2:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 2
1364	001526	000001	DZVC2:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 2
1365	001530	000001	LINE2:	.BLKW	1	:ALL LINES SELECTED
1366	001532	000001	PAR2:	.BLKW	1	:PARAMETERS
1367	001534	000001	MANT2:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
1368						
1369	001536	000001	DZCR3:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 3
1370	001540	000001	DZVC3:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 3
1371	001542	000001	LINE3:	.BLKW	1	:ALL LINES SELECTED
1372	001544	000001	PAR3:	.BLKW	1	:PARAMETERS
1373	001546	000001	MANT3:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
1374						
1375	001550	000001	DZCR4:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 4
1376	001552	000001	DZVC4:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 4
1377	001554	000001	LINE4:	.BLKW	1	:ALL LINES SELECTED
1378	001556	000001	PAR4:	.BLKW	1	:PARAMETERS
1379	001560	000001	MANT4:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
1380						
1381	001562	000001	DZCR5:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 5
1382	001564	000001	DZVC5:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 5
1383	001566	000001	LINE5:	.BLKW	1	:ALL LINES SELECTED
1384	001570	000001	PAR5:	.BLKW	1	:PARAMETERS
1385	001572	000001	MANT5:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
1386						
1387	001574	000001	DZCR6:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 6
1388	001576	000001	DZVC6:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 6
1389	001600	000001	LINE6:	.BLKW	1	:ALL LINES SELECTED
1390	001602	000001	PAR6:	.BLKW	1	:PARAMETERS
1391	001604	000001	MANT6:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
1392						
1393	001606	000001	DZCR7:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 7
1394	001610	000001	DZVC7:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 7
1395	001612	000001	LINE7:	.BLKW	1	:ALL LINES SELECTED
1396	001614	000001	PAR7:	.BLKW	1	:PARAMETERS
1397	001616	000001	MANT7:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
1398						
1399	001620	000001	DZCR10:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 10
1400	001622	000001	DZVC10:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 10
1401	001624	000001	LINE10:	.BLKW	1	:ALL LINES SELECTED
1402	001626	000001	PAR10:	.BLKW	1	:PARAMETERS
1403	001630	000001	MANT10:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
1404						
1405	001632	000001	DZCR11:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 11
1406	001634	000001	DZVC11:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 11
1407	001636	000001	LINE11:	.BLKW	1	:ALL LINES SELECTED
1408	001640	000001	PAR11:	.BLKW	1	:PARAMETERS
1409	001642	000001	MANT11:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
1410						
1411	001644	000001	DZCR12:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 12
1412	001646	000001	DZVC12:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 12
1413	001650	000001	LINE12:	.BLKW	1	:ALL LINES SELECTED
1414	001652	000001	PAR12:	.BLKW	1	:PARAMETERS
1415	001654	000001	MANT12:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE

1416					
1417	001656	000001	DZCR13:	.BLKW	1
1418	001660	000001	DZVC13:	.BLKW	1
1419	001662	000001	LINE13:	.BLKW	1
1420	001664	000001	PAR13:	.BLKW	1
1421	001666	000001	MANT13:	.BLKW	1
1422					
1423	001670	000001	DZCR14:	.BLKW	1
1424	001672	000001	DZVC14:	.BLKW	1
1425	001674	000001	LINE14:	.BLKW	1
1426	001676	000001	PAR14:	.BLKW	1
1427	001700	000001	MANT14:	.BLKW	1
1428					
1429	001702	000001	DZCR15:	.BLKW	1
1430	001704	000001	DZVC15:	.BLKW	1
1431	001706	000001	LINE15:	.BLKW	1
1432	001710	000001	PAR15:	.BLKW	1
1433	001712	000001	MANT15:	.BLKW	1
1434					
1435	001714	000001	DZCR16:	.BLKW	1
1436	001716	000001	DZVC16:	.BLKW	1
1437	001720	000001	LINE16:	.BLKW	1
1438	001722	000001	PAR16:	.BLKW	1
1439	001724	000001	MANT16:	.BLKW	1
1440					
1441	001726	000001	DZCR17:	.BLKW	1
1442	001730	000001	DZVC17:	.BLKW	1
1443	001732	000001	LINE17:	.BLKW	1
1444	001734	000001	PAR17:	.BLKW	1
1445	001736	000001	MANT17:	.BLKW	1
1446					
1447	001740	177777	DZV.END:		177777

1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495

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

```

:*****
:-----
:TRPTAB:
ADVANCE=TRAP+0           ;CALL TO ADVANCE TO NEXT TEST( OR SCOPE THIS ONE)
      .ADVANCE
SCOPI=TRAP+1            ;CALL TO LOOP ON CURRENT DATA HANDLER
      .SCOPI
TYPE=TRAP+2            ;CALL TO TELETYPE OUTPUT ROUTINE
      .TYPE
INSTR=TRAP+3           ;CALL TO ASCII STRING INPUT ROUTINE
      .INSTR
INSTER=TRAP+4          ;CALL TO INPUT ERROR HANDLER
      .INSTER
PARAM=TRAP+5           ;CALL TO NUMERICAL DATA INPUT ROUTINE
      .PARAM
SETFLG=TRAP+6          ;CALL TO SET FLAG ROUTINE
      .SETFLG
SAVOS=TRAP+7           ;CALL TO REGISTER SAVE ROUTINE
      .SAVOS
RESOS=TRAP+10          ;CALL TO REGISTER RESTORE ROUTINE
      .RESOS
CONVRT=TRAP+11         ;CALL TO DATA OUTPUT ROUTINE
      .CONVRT
CNVRT=TRAP+12          ;CALL TO DATA OUTPUT ROUTINE WITHOUT CR/LF.
      .CNVRT
DEVICE.CLR=TRAP+13     ;CALL TO ISSUE A DEVICE CLEAR
      .DEVICE.CLR
DELAY=TRAP+14          ;CALL TO DELAY FOR FAST CPU'S
      .DELAY
PARMD=TRAP+15          ;CONVERT DECIMAL STRING TO OCTAL
      .PARMD
PAWCH=TRAP+16          ;SET FLAG ECHO OR CABLE
      .PAWCH
DCLASM=TRAP+17         ;CLEAR DEVICE, SET MAINT. BIT IF I MODE
      .DCLASM
SHIFT=TRAP+20          ;CALL TO ROTATE LINE POINTER
      .SHIFT
LPRSET=TRAP+21         ;CALL TO SET UP LPR DEVICE REGISTER
      .LPRSET
BUFSET=TRAP+22         ;CALL TO ZERO BUFFER AREA
      .BUFSET
:-----
:*****

```

```

1496                                     :DZV11 VECTOR AND REGISTER INDIRECT POINTERS
1497                                     :WORKING AREA
1498
1499 002010 160040 DZVCSR: 160040 :R/W
1500 002012 160041 HDZVCSR:160041 :R/W
1501 002014 160042 DZVRBUF:160042 :READ ONLY
1502 002016 160043 HDZVRBUF:160043 :READ ONLY
1503 002020 160042 DZVLPR: 160042 :WRITE ONLY
1504 002022 160043 HDZVLPR:160043 :WRITE ONLY
1505 002024 160044 DZVTCR: 160044 :R/W
1506 002026 160045 HDZVTCR:160045 :R/W
1507 002030 160046 DZVMSR: 160046 :READ ONLY
1508 002032 160047 HDZVMSR:160047 :READ ONLY
1509 002034 160046 DZVTDR: 160046 :WRITE ONLY
1510 002036 160047 HDZVTDR:160047 :WRITE ONLY
1511
1512                                     :DEFAULT DZV VECTORS
1513
1514 002040 000300 DZVRIV: 300 :REC INTR VECTOR
1515 002042 000302 DZVRIS: 302 :REC INTR STATUS
1516 002044 000304 DZVTIV: 304 :XMIT INTR VECTOR
1517 002046 000306 DZVTIS: 306 :XMIT INTR STATUS
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

002050
002050 000000
002052 000000
002054 000000
002056 000000
002060 000000
002062 000000
002064 000000
002066 000000
002070 000000
002072 000000
002074 000000
002076 000000
002100 000000
002102 000000
002104 000000
002106 000000
002110 000000
002112 000000
002114 000000

:TIME TABLE FOR RELATIVE TIMING TESTS

TMTBL:
T50: 0
T75: 0
T110: 0
T134: 0
T150: 0
T300: 0
T600: 0
T1200: 0
T1800: 0
T2000: 0
T2400: 0
T3600: 0
T4800: 0
T7200: 0
T9600: 0
TEIGHT:0
TSEVEN: 0
TSIX: 0
TFIVE: 0

```

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

```

```

:PROGRAM INITIALIZATION
:LOCK OUT INTERRUPTS
:SET UP PROCESSOR STACK
:SET UP POWER FAIL VECTOR
:CLEAR PROGRAM CONTROL FLAGS AND COUNTS
:TYPE TITLE MESSAGE

.START:
RESET
MOV #STACK,SP
MTPS #MASK
MOV #SPWRDN,@#24
MOV #ERROR,EMTVEC
MOV #340,EMTVEC+2
CLR $PASS
CLRB $ERFLG
MOV #DZV.MAP,ACTIVE
MOV #1,RUN
CLR $ERTTL
CLR $ERRPC
CLR $STNM
MOV #.START,$LPADR

: CLEAR THE WORLD. START NEW ENVIRONMENT
: SET UP STACK
: LOCK OUT INTERRUPTS
: SET UP POWER FAIL VECTOR
: SET UP ERROR VECTOR
: CLEAR PASS COUNT
: CLEAR ERROR FLAG
: GET MAP POINTER.
: POINT POINTER TO FIRST DEVICE.
: CLEAR ERROR COUNT
: CLEAR LAST ERROR POINTER
: SET UP FOR TEST 1
: SET UP FOR POWER FAIL BEFORE
: TESTING STARTS

: SET UP FOR SMALL 11 SWITCH REGISTER COMPATIBILITY
MOV #SWREG,SWR
MOV #DISPREG,DISPLAY
TSTB INIFLG
BNE 10$
CMP @#42,#SENDAD
BEQ 1$
TYPE ,MTITLE
DECB INIFLG
TSTB $ENVM
BPL 15$
JSR PC,SETAPT
JMP 105$
BIT #SW00,@SWR
BNE 20$
JMP 55$
MOV #DZV.MAP,RO
CLRB HDRFLG
CLR (RO)+
CMP RO,#DZV.END
BNE 25$
DECB INIFLG

: POINT TO SOFTWARE SWR
: POINT TO SOFTWARE DISPLAY REGISTER
: HAVE WE ALREADY BEEN HERE TODAY?
: IF SO, SKIP PRINTING THE TITLE
: IF RUNNING UNDER ACT
: DON'T PRINT TITLE
: PRINT THE DIAGNOSTIC'S TITLE
: SET THE ONCE ONLY FLAG
: DETERMINE WHETHER APT SIZING SHOULD BE DONE
: IF NOT, GO CHECK FOR AUTO-SIZING
: OTHERWISE, GO DO APT SIZING FROM ETABLE
: GO PRINT DZV STATUS TABLE
: RESELECT ?
: IF YES, GO SET UP THE INFORMATION
: IF NO, SKIP THE INTERROGATION
: POINT TO THE BEGINNING OF THE MAP TABLE
: MAKE SURE A MAP GETS PRINTED
: CLEAR A TABLE LOCATION
: HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
: IF NOT, CLEAR THE NEXT LOCATION IN THE TABLE
: INSURE NO AUTO SIZING IF QUESTIONS ANSWERED!

: THE FOLLOWING ARE PARAMETERS USED TO FILL IN THE MAP
: TABLE AND SET UP THE DIAGNOSTIC.

: GET THE BASE ADDRESS OF THE DZV11'S

INSTR 91$
PARAM 160000

: CALL THE STRING INPUT ROUTINE
: POINTER TO MESSAGE TO BE PRINTED
: CALL THE OCTAL TO ASCII CONVERT ROUTINE
: LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE

```

1600	002350	163770				163770		:HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1601	002352	001500				DZCRO		:POINTER TO MAP LOCATION TO BE FILLED
1602	002354	007				.BYTE	7	:MASK OF INVALID BITS FOR THIS PARAMETER
1603	002355	001				.BYTE	1	:NUMBER OF PARAMETERS TO STORE
1604	002356	013737	001500	001174		MOV	DZCRO,\$BASE	:COPY BASE ADDRESS TO ETABLE
1605								
1606								:GET THE BASE VECTOR ADDRESS
1607								
1608	002364	104403				INSTR		:CALL THE STRING INPUT ROUTINE
1609	002366	003076				92\$:POINTER TO MESSAGE TO BE PRINTED
1610	002370	104405				PARAM		:CALL THE OCTAL TO ASCII CONVERT ROUTINE
1611	002372	000300				300		:LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1612	002374	000776				776		:HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1613	002376	001502				DZVCO		:POINTER TO MAP LOCATION TO BE FILLED
1614	002400	003				.BYTE	3	:MASK OF INVALID BITS FOR THIS PARAMETER
1615	002401	001				.BYTE	1	:NUMBER OF PARAMETERS TO STORE
1616	002402	013737	001502	001170		MOV	DZVCO,\$VECT1	:COPY VECTOR TO ETABLE
1617								:GET THE MODE OF OPERATION (E,I,S)
1618								
1619	002410	104403				INSTR		:CALL THE STRING INPUT ROUTINE
1620	002412	003325				96\$:POINTER TO THE MESSAGE TO BE PRINTED
1621	002414	104406				SETFLG		:CALL THE MAINTENANCE FLAG SETUP ROUTINE
1622	002416	001510				MANTO		:THIS IS THE FLAG BEING SETUP
1623								
1624								:GET THE NUMBER OF DZV11'S RUNNING
1625								
1626	002420	104403				INSTR		:CALL THE STRING INPUT ROUTINE
1627	002422	003262				95\$:POINTER TO MESSAGE TO BE PRINTED
1628	002424	104405				PARAM		:CALL THE OCTAL TO ASCII CONVERT ROUTINE
1629	002426	000001				1		:LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1630	002430	000020				16.		:HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1631	002432	001344				\$TMP1		:POINTER TO MAP LOCATION TO BE FILLED
1632	002434	000				.BYTE	0	:MASK OF INVALID BITS FOR THIS PARAMETER
1633	002435	001				.BYTE	1	:NUMBER OF PARAMETERS TO STORE
1634								
1635	002436	012737	000017	001504		MOV	#17,LINE0	:SET UP DEFAULT LINES
1636	002444	012737	017470	001506		MOV	#17470,PAR0	:SET UP DEFAULT LPR PARAMETER
1637								:RECEIVER ON; 19.2 KBAUD; 2STOP BITS; 8 BIT/CHAR
1638	002452	032777	000010	176624		BIT	#SW03,@SWR	:DO YOU WANT PARAMETERS?
1639	002460	001402				BEQ	30\$:IF NO, SKIP THE PARAMETER CALL
1640	002462	004737	002642			JSR	PC,65\$:GET PARAMETERS
1641	002466	012737	000001	001410	30\$:	MOV	#1,SAVACTV	:INITIALIZE ACTIVE DEVICE SELECTION PARAMETER
1642	002474	113737	001344	001414		MOV	\$TMP1,DZVNUM	:COPY THE NUMBER OF DEVICES
1643	002502	005337	001344		35\$:	DEC	\$TMP1	: \$TMP1 CONTAINS THE COUNT OF UNINITIALIZED
1644	002506	001404				BEQ	40\$: SELECTED DEVICES
1645	002510	000261				SEC		:SET A BIT FLAG TO INDICATE AN ACTIVE DEVICE
1646	002512	006137	001410			ROL	SAVACTV	:POINT TO THE NEXT DEVICE
1647	002516	000771				BR	35\$:GO DO THIS PROCEDURE AGAIN
1648	002520	013737	001410	001346	40\$:	MOV	SAVACTV,\$TMP2	:# OF TIMES
1649	002526	012700	001500			MOV	#DZCRO,R0	:SET A POINTER TO THE SPECIFIED INFORMATION
1650	002532	012701	001512			MOV	#DZCR1,R1	:POINT R1 TO THE REST OF THE MAP TABLE
1651	002536	012702	001204			MOV	#SDDWO,R2	:POINT TO ETABLE'S DEVICE DESCRIPTOR WORDS
1652	002542	000241				CLC		:INITIALIZE THE 'C' BIT FOR A ROTATION
1653	002544	006037	001346			ROR	\$TMP2	:SKIP MAPPING SETUP FOR DEVICE 0- IT'S DONE
1654	002550	006237	001346		45\$:	ASR	\$TMP2	:ISOLATE A SELECTION FLAG IN THE 'C' BIT
1655	002554	103404				BCS	50\$:IS THIS DEVICE SELECTED? IF YES, GO LOAD TABLE

```

1656 002556 012711 177777      MOV      #-1,(R1)      ;TERMINATE THE LIST
1657 002562 000137 003530      JMP      100$          ;GO TO THE NEXT BLOCK
1658 002566 012011                50$:     MOV      (R0)+,(R1)    ;ADDRESS
1659 002570 062721 000010      ADD      #10,(R1)+    ;POINT TO THE NEXT DZV11 ADDRESS VALUE
1660 002574 012011                MOV      (R0)+,(R1)    ;VECTOR
1661 002576 062721 000010      ADD      #10,(R1)+    ;POINT TO THE NEXT VECTOR VALUE
1662 002602 012021                MOV      (R0)+,(R1)+   ;LINES
1663 002604 012021                MOV      (R0)+,(R1)+   ;PARAMETERS
1664 002606 012021                MOV      (R0)+,(R1)+   ;MAINTENANCE MODE
1665 002610 000757                BR       45$           ;
1666 002612 032777 000010 176464 55$:     BIT      #SW03,@SWR    ;ASK PARAMETERS ?
1667 002620 001002                BNE     60$           ;IF NO, GO DO AUTO SIZING
1668 002622 000137 003530      JMP      100$          ;GO SET UP FOR AUTO SIZING
1669 002626 004737 002642      60$:     JSR      PC,65$        ;GO ASK PARAMETERS
1670 002632 105337 001422      DECB    INIFLG        ;INSURE NO AUTO SIZE IF QUESTIONS ANSWERED
1671 002636 000137 003554      JMP      105$          ;GO TO THE NEXT BLOCK
1672
1673                                ;GET THE ACTIVE LINES PARAMETER
1674
1675 002642                65$:     INSTR     ;CALL THE STRING INPUT ROUTINE
1676 002642 104403          93$      ;POINTER TO MESSAGE TO BE PRINTED
1677 002644 003137          PARAM    ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
1678 002646 104405          1        ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1679 002650 000001          17       ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1680 002652 000017          LINE0   ;POINTER TO MAP LOCATION TO BE FILLED
1681 002654 001504          .BYTE   360        ;MASK OF INVALID BITS FOR THIS PARAMETER
1682 002656                .BYTE   1          ;NUMBER OF PARAMETERS TO STORE
1683 002657                CLRB    HDRFLG     ;MAKE SURE THE CHANGES ARE PRINTED
1684 002660 105037 001423
1685
1686                                ;THIS SEGMENT CHECKS TO MAKE SURE THE LINE PARAMETER JUST ENTERED
1687                                ;IS LEGITIMATE IN STAGGERED MODE OPERATION IF THAT MODE WAS SELECTED
1688
1689 002664 005737 001510      TST     MANT0        ;IS STAGGERED THE MODE OF OPERATION?
1690 002670 100021          BPL     85$          ;IF NOT, SKIP THIS SEGMENT
1691 002672 013703 001504      MOV     LINE0,R3     ;GET A SCRATCH COPY OF THE ACTIVE LINES
1692 002676 006003          ROR     R3          ;GET A LINE SELECTION BIT(EVEN NUMBER LINE)
1693 002700 103410          BCS     80$          ;IF IT IS SELECTED, CHECK TO SEE IF THE NEXT IS TOO
1694 002702 001414          BEQ     85$          ;IF ALL HAVE BEEN CHECKED, CONTINUE PROCESSING
1695 002704 006203          ASR     R3          ;IF IT IS 0,CHECK TO SEE IF THE NEXT IS TOO
1696 002706 103373          BCC     70$          ;IF THIS ONE'S 0 TOO, GO CHECK THE NEXT PAIR
1697 002710 104402 001356      TYPE   ,SQUES        ;THIS IS AN INCORRECT PARAMETER
1698 002714 104402 010034      TYPE   ,MBADLN      ;LET THE USER KNOW ABOUT IT
1699 002720 000750          BR     65$          ;GO GET THE CORRECT PARAMETER
1700 002722 001772          BEQ     75$          ;IF ANOTHER FLAG ISN'T SET, THERE'S AN ERROR
1701 002724 006203          ASR     R3          ;GET THE NEXT FLAG
1702 002726 103370          BCC     75$          ;IF IT ISN'T SET, THERE'S AN ERROR
1703 002730 000241          CLC
1704 002732 000761          BR     70$          ;INITIALIZE THE 'C' BIT FOR TESTING OF THE NEXT PAIR
1705                                ;GO TEST THE NEXT PAIR OF FLAGS
1706
1707                                ;GET THE LINE PARAMETER REGISTER ARGUMENT
1708
1708 002734                85$:     INSTR     ;CALL THE STRING INPUT ROUTINE
1709 002734 104403          94$      ;POINTER TO MESSAGE TO BE PRINTED
1710 002736 003212          PARAM    ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
1711 002740 104405

```

1712	002742	000000				0		:LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1713	002744	000017				17		:HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1714	002746	001506				PAR0		:POINTER TO MAP LOCATION TO BE FILLED
1715	002750	000				.BYTE 0		:MASK OF INVALID BITS FOR THIS PARAMETER
1716	002751	001				.BYTE 1		:NUMBER OF PARAMETERS TO STORE
1717	002752	012702	001504			MOV #LINE0,R2		:POINT TO THE LINE SELECTION PARAMETER
1718	002756	012703	001506			MOV #PAR0,R3		:POINT TO THE CHOSEN PARAMETERS
1719	002762	011304				MOV (R3),R4		:USE BAUD RATE AS AN INDEX IN DELAY TABLE
1720	002764	006304				ASL R4		:ALIGN INDEX ON WORD BOUNDARY
1721	002766	016437	017374	006244		MOV DLYTBL(R4),DLYCNT		:SET THE DELAY COUNT FOR THIS BAUD RATE
1722	002774	000313				SWAB (R3)		:PLACE IN HIGH BYTE
1723	002776	052713	010070			BIS #10070,(R3)		:PLACE EXTRA PARAMETERS INTO LOC
1724	003002	011262	000012		90\$:	MOV (R2),12(R2)		:LOAD THE LINES
1725	003006	011363	000012			MOV (R3),12(R3)		:LOAD THE PARAMETERS
1726	003012	062702	000012			ADD #12,R2		:POINT TO THE NEXT SET
1727	003016	062703	000012			ADD #12,R3		:... OF BOTH PARAMETERS
1728	003022	020327	001734			CMP R3,#PAR17		:HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
1729	003026	001365				BNE 90\$:IF NOT, GO LOAD SOME MORE PARAMETERS
1730	003030	000207				RTS PC		:RETURN TO CALLING BLOCK
1731	003032	030600	052123	041440	91\$:	.ASCIZ <200>/1ST CSR ADDRESS (160000:163770): /		
(1)	003076	030600	052123	053040	92\$:	.ASCIZ <200>/1ST VECTOR ADDRESS (300:770): /		
(1)	003137	200	044514	042516	93\$:	.ASCIZ <200>/LINES ACTIVE BY BIT <IN OCTAL>(001:17): /		
(1)	003212	042200	043105	052501	94\$:	.ASCIZ <200>/DEFAULT BAUD RATE <IN OCTAL>(00:17): /		
(1)	003262	021600	047440	020106	95\$:	.ASCIZ <200>/# OF DZV11'S <IN OCTAL> (1:20): /		
(1)	003325	200	040515	047111	96\$:	.ASCII <200>/MAINTENANCE MODE/		
(1)	003346	020200	042533	052130		.ASCII <200>/ [EXTERNAL <H325> (E)]/		
(1)	003402	020200	044533	052116		.ASCII <200>/ [INTERNAL <DZVCSR03=1>(I)]/		
(1)	003437	200	055440	052123		.ASCIZ <200>/ [STAGGERED <H329> (S)]: /		
(1)	003476	042600	052116	051105	97\$:	.ASCIZ <200>/ENTER DELAY PARAMETER: /		
(1)	003530	003530			.EVEN			
(1)	003530				100\$:			
1732	003530	122737	000377	001422		CMPB #377,INIFLG		:ONLY DO AUTO SIZE ON 1ST START
1733	003536	001006				BNE 105\$		
1734	003540	032777	000200	175536		BIT #BIT7,@SWR		:BIT7=1??
1735	003546	001002				BNE 105\$:BR IF NO AUTO SIZE
1736	003550	004737	011500			JSR PC,AUTO.SIZE		:GO DO THE AUTO SIZE
1737	003554	105737	001423		105\$:	TSTB HDRFLG		:HAS THE TABLE BEEN TYPED YET?
1738	003560	001021				BNE 120\$:IF SO, DON'T TYPE IT AGAIN
1739	003562	105337	001423			DECB HDRFLG		:INDICATE THAT THE TABLE WILL BE TYPED
1740	003566	104402	010006			TYPE ,XHEAD		:TYPE MAP HEADER
1741	003572	012700	001500			MOV #DZV.MAP,R0		:SET POINTER
1742	003576	010037	001344		110\$:	MOV R0,\$TMP1		:POINT TO THE MAP LOCATION
1743	003602	012037	001346			MOV (R0)+,\$TMP2		:SET DATA
1744	003606	022737	177777	001346		CMP #-1,\$TMP2		:END OF LIST?
1745	003614	001403				BEQ 120\$:BR IF YES
1746	003616	104411			115\$:	CONVRT		:CALL THE OCTAL TO ASCII CONVERSION ROUTINE
1747	003620	010076				XSTATQ		:CONVERT THE DATA AT THIS ADDRESS
1748	003622	000765				BR 110\$:GO PRINT THE NEXT PARAMETER
1749	003624	013737	001410	001406	120\$:	MOV SAVACTV,DZVACTV		:COPY BIT MAP OF SYSTEM DEVICES ACTIVE
1750	003632	113737	001414	001416		MOV DZVNUM,SAVNO		:COPY NO. OF SYSTEM DEVICES ACTIVE
1751	003640	032777	000100	175436		BIT #SW06,@SWR		:DESELECT SPECIFIC DEVICES??
1752	003646	001431				BEQ 135\$:BR IF NO.
1753	003650				121\$:			
1754	003650	104403				INSTR		:CALL THE STRING INPUT ROUTINE
1755	003652	007724				MNEW		:POINTER TO MESSAGE TO BE PRINTED
1756	003654	104405				PARAM		:CALL THE OCTAL TO ASCII CONVERT ROUTINE

1757	003656	000001			1			:LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1758	003660	177777			177777			:HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1759	003662	001406			DZVACTV			:POINTER TO MAP LOCATION TO BE FILLED
1760	003664	000			.BYTE	0		:MASK OF INVALID BITS FOR THIS PARAMETER
1761	003665	001			.BYTE	1		:NUMBER OF PARAMETERS TO STORE
1762	003666	023737	001406	001410	COMP	DZVACTV, SAVACTV		:IS THE VALUE VALID?
1763	003674	101403			BLOS	1228		:BRANCH IF YES
1764	003676	104402	007576		TYPE	,MERR3		:IF NOT THEN TYPE ERROR
1765	003702	000762			BR	1218		:GO REASK QUESTION
1766	003704	105037	001416		CLRB	SAVND	1228:	:CLEAR NO. OF DEVICES BEING TESTED
1767	003710	013737	001406	001344	MOV	DZVACTV, STMP1		:COPY BIT MAP OF ACTIVE DEVICES BEING TESTED
1768	003716	006237	001344		ASR	STMP1	1268:	:SHIFT OUT AN ACTIVE BIT
1769	003722	103002			BCC	1278		:IF NOT ACTIVE SKIP INCREMENT
1770	003724	105237	001416		INCB	SAVND		:IF ACTIVE RECORD IT
1771	003730	001372			BNE	1268	1278:	:IF ALL ACTIVE BITS RECORDED DON'T BRANCH
1772	003732	032777	000020	175344	BIT	#SW04, @SWR	1358:	:CHECK TO SEE IF DELAY COUNT CHANGES
1773	003740	001407			BEQ	1408		:IF NOT, GO CLEAR VECTOR AREA
1774	003742	104403			INSTR			:CALL THE STRING INPUT ROUTINE
1775	003744	003476			978			:POINTER TO MESSAGE TO BE PRINTED
1776	003746	104405			PARAM			:CALL THE OCTAL TO ASCII CONVERT ROUTINE
1777	003750	000001			1			:LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1778	003752	177777			177777			:HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1779	003754	006244			DLYCNT			:POINTER TO MAP LOCATION TO BE FILLED
1780	003756	000			.BYTE	0		:MASK OF INVALID BITS FOR THIS PARAMETER
1781	003757	001			.BYTE	1		:NUMBER OF PARAMETERS TO STORE
1782	003760	012700	000300		MOV	#300, R0	1408:	:PREPARE TO CLEAR THE FLOATING
1783	003764	012701	000302		MOV	#302, R1		:VECTOR AREA. 300-776
1784	003770	010120			MOV	R1, (R0)+	1458:	:START PUTTING 'PC+2 - HALT'
1785	003772	005021			CLR	(R1)+		:IN VECTOR AREA.
1786	003774	022021			COMP	(R0)+, (R1)+		:POP POINTERS
1787	003776	022700	001000		COMP	#1000, R0		:ALL DONE??
1788	004002	001372			BNE	1458		:BR IF NO.
1789								
1790								
1791								
1792								
1793	004004	012706	001120		.BEGIN: MOV	#STACK, SP		:SET UP STACK
1794	004010	106427	000200		MTPS	#MASK		:LOCK OUT INTERRUPTS
1795	004014	005737	000042		TST	@#42		:IS PROGRAM UNDER MONITOR CONTROL
1796	004020	001015			BNE	28		:BR IF YES
1797	004022	032777	000004	175254	BIT	#BIT2, @SWR		:CHECK FOR LOCK ON TEST
1798	004030	001406			BEQ	18		:BR IF NO LOCK DESIRED.
1799	004032	104402	007622		TYPE	,MLOCK		:TYPE LOCK SELECTED.
1800	004036	012737	000240	004326	MOV	#NOP, TTST		:ADJUST SCOPE ROUTINE.
1801	004044	000403			BR	28		:CONTINUE ALONG.
1802	004046	013737	004554	004326	MOV	BRW, TTST	18:	:PREPARE NORMAL SCOPE ROUTINE
1803	004054	012737	010452	001252	MOV	#CYCLE, SLPADR	28:	:START AT 'CYCLE' FIND WHICH DEVICE TO TEST
1804	004062	113737	001416	001415	MOV	SAVND, SAVNUM		:COPY ACTIVE DEVICES BEING TESTED
1805	004070	104402	007513		TYPE	,MR		:TYPE 'RUNNING'
1806	004074	000177	175152		JMP	@SLPADR		:START TESTING

:TEST START AND RESTART

:-----

```

1807                                     :END OF PASS
1808                                     :TYPE NAME OF TEST
1809                                     :UPDATE PASS COUNT
1810                                     :CHECK FOR EXIT TO ACT-11
1811                                     :RESTART TEST
1812 .SBTTL END OF PASS ROUTINE
1813
1814 ::*****
1815 :*INCREMENT THE PASS NUMBER ($PASS)
1816 :*IF THERES A MONITOR GO TO IT
1817 :*IF THERE ISN'T JUMP TO CYCLE
1818
1819 004100 SEOP:
1820 004100 000004 SCOPE
1821 004102 005037 001262 CLR SERRPC ;CLEAR LAST ERROR PC
1822 004106 105037 001247 CLRB SERFLG ;CLEAR ERROR FLAG
1823 004112 104402 007467 TYPE ,MEPASS ;TYPE END PASS
1824 004116 104402 007651 TYPE ,MCSRX ;TYPE CSR
1825 004122 104412 004264 CNVRT ,XCSR ;SHOW IT
1826 004126 104402 007657 TYPE ,MVECX ;TYPE VECTOR
1827 004132 104412 004272 CNVRT ,XVEC ;SHOW IT
1828 004136 005237 001126 INC $PASS ;RAISE PASS COUNT
1829 004142 104402 007665 TYPE ,MPASSX ;TYPE PASSES
1830 004146 104412 004300 CNVRT ,XPASS ;SHOW IT
1831 004152 005337 001126 DEC $PASS ;RESTORE PASS COUNT
1832 004156 104402 007676 TYPE ,MERRX ;TYPE ERRORS
1833 004162 104412 004306 CNVRT ,XERR ;SHOW IT
1834 004166 005237 001130 INC $DEVCT ;INC DEVCNT FOR APT
1835 004172 105337 001415 DECB SAVNUM ;ARE ALL DEVICES TESTED?
1836 004176 001030 BNE SDOAGN ;BR IF NO.
1837 004200 113737 001416 001415 MOVB SAVNO,SAVNUM ;RESTORE THE COUNT
1838 004206 005037 001354 CLR $TIMES ;ZERO THE NUMBER OF ITERATIONS
1839 004212 005237 001126 INC $PASS ;INCREMENT THE PASS NUMBER
1840 004216 042737 100000 001126 BIC #100000,$PASS ;DON'T ALLOW A NEG. NUMBER
1841 004224 005327 DEC (PC)+ ;LOOP?
1842 004226 000001 SEOPCT: .WORD 1
1843 004230 003013 BGT SDOAGN ;:YES
1844 004232 012737 MOV (PC)+,@(PC)+ ;:RESTORE COUNTER
1845 004234 000001 SENDCT: .WORD 1
1846 004236 004226 SEOPCT
1847 004240 013700 000042 $GET42: MOV @#42,R0 ;:GET MONITOR ADDRESS
1848 004244 001405 BEQ SDOAGN ;:BRANCH IF NO MONITOR
1849 004246 000005 RESET ;:CLEAR THE WORLD
1850 004250 004710 SENDAD: JSR PC,(R0) ;:GO TO MONITOR
1851 004252 000240 NOP ;:SAVE ROOM
1852 004254 000240 NOP ;:FOR
1853 004256 000240 NOP ;:ACT11
1854 004260 SDOAGN:
1855 004260 000137 JMP @(PC)+ ;:RETURN
1856 004262 010452 SRTNAD: .WORD CYCLE
1857
1858 004264 000001 XCSR: 1
1859 004266 006 002 .BYTE 6,2
1860 004270 002010 DZVCSR
1861 004272 000001 XVEC: 1
1862 004274 003 002 .BYTE 3,2

```

```

1863 004276 002040
1864 004300 000001
1865 004302 006 002
1866 004304 001126
1867 004306 000001
1868 004310 006 002
1869 00-312 001256
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886 004314
1887 004314 005037 001262
1888 004320 022716 012206
1889 004324 001413
1890 004326 000406
1891 004330 105777 174754
1892 004334 100067
1893 004336 017766 174750 177776
1894 004344 032777 040000 174732
1895 004352 001060
1896
1897 004354 000416
1898
1899 004356 013746 000004
1900 004362 012737 004402 000004
1901 004370 005737 177060
1902 004374 012637 000004
1903 004400 000436
1904 004402 022626
1905 004404 012637 000004
1906 004410 000441
1907 004412
1908 004412 105737 001247
1909 004416 001404
1910 004420 105037 001247
1911 004424 005037 001354
1912 004430 032777 004000 174646
1913 004436 001011
1914 004440 005737 001126
1915 004444 001406
1916 004446 005237 001250
1917 004452 023737 001354 001250
1918 004460 002015

DZVRIV
XPASS: 1
      .BYTE 6.2
      $PASS
XEERR: 1
      .BYTE 6.2
      $ERTTL

:SCOPE LOOP AND ITERATION HANDLER
:-----

.SBTTL SCOPE HANDLER ROUTINE

:*****
:THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
:AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG. (DISPLAY<7:0>)
:AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
:THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
:SW14=1 LOOP ON TEST
:SW11=1 INHIBIT ITERATIONS
:CALL
:* SCOPE ;;SCOPE=IOT

$SCOPE:
.SCOPE: CLR $ERRPC ;CLEAR LAST ERROR PC.
      CMP #TST1+2,(SP) ;IS THIS THE SCOPE AT THE BEGINNING OF TST1?
      BEQ $XTSTR ;IF SO, DON'T LOOP ON IT
TTST: BR 1$ ;GOTO 1$ (IF LOCK SW02=1; THIS LOC =240)
      TSTB @STKS ;KEYBOARD DONE?
      BPL $OVER ;BR IF NO. (LOCK: HIT KEY TO GOTO NEXT TEST)
      MOV @STKB,-2(SP) ;CLEAR DONE BIT
1$: BIT #BIT14,@SWR ;LOOP ON PRESENT TEST?
      BNE $OVER ;YES IF SW14=1
:*****START OF CODE FOR THE XOR TESTER*****
$XTSTR: BR 6$ ;IF RUNNING ON THE 'XOR' TESTER CHANGE
      MOV @ERRVEC,-(SP) ;THIS INSTRUCTION TO A 'NOP' (NOP=240)
      MOV #5$,@ERRVEC ;SAVE THE CONTENTS OF THE ERROR VECTOR
      TST @177060 ;SET FOR TIMEOUT
      MOV (SP)+,@ERRVEC ;TIME OUT ON XOR?
      BR $SVLAD ;RESTORE THE ERROR VECTOR
      CMP (SP)+,(SP)+ ;GO TO THE NEXT TEST
      MOV (SP)+,@ERRVEC ;CLEAR THE STACK AFTER A TIME OUT
      BR $OVER ;RESTORE THE ERROR VECTOR
      ;LOOP ON THE PRESENT TEST
6$: ;*****END OF CODE FOR THE XOR TESTER*****
2$: TSTB $ERFLG ;HAS AN ERROR OCCURRED?
      BEQ 3$ ;BR IF NO
4$: CLRB $ERFLG ;ZERO THE ERROR FLAG
      CLR $TIMES ;CLEAR THE NUMBER OF ITERATIONS TO MAKE
3$: BIT #BIT11,@SWR ;INHIBIT ITERATIONS?
      BNE 1$ ;BR IF YES
      TST $PASS ;IF FIRST PASS OF PROGRAM
      BEQ 1$ ; INHIBIT ITERATIONS
      INC $ICNT ;INCREMENT ITERATION COUNT
      CMP $TIMES,$ICNT ;CHECK THE NUMBER OF ITERATIONS MADE
      BGE $OVER ;BR IF MORE ITERATION REQUIRED

```

```

1919 004462 012737 000001 001250 1$: MOV #1,$ICNT ;;REINITIALIZE THE ITERATION COUNTER
1920 004470 013737 004556 001354 MOV $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
1921 004476 105237 001246 $SVLAD: INCB $STNM ;;COUNT TEST NUMBERS
1922 004502 113737 001246 001124 MOV $STNM,$TESTN ;;SET TEST NUMBER IN APT MAILBOX
1923 004510 011637 001252 MOV (SP),$LPADR ;;SAVE SCOPE LOOP ADDRESS
1924 004514 013777 001246 174564 $OVER: MOV $STNM,@DISPLAY ;;DISPLAY TEST NUMBER
1925 004522 013716 001252 MOV $LPADR,(SP) ;;FUDGE RETURN ADDRESS
1926 004526 004737 007006 JSR PC,SERV.G ;;FIND OUT IF ^G WAS TYPED
1927 004532 105037 001424 CLRB MNTFLG ;;CLEAR THE MAINTENANCE BIT SETTER AFTER EACH TEST
1928 004536 005737 001372 TST MODE ;;HAS THE MODE BEEN CHANGED?
1929 004542 001003 BNE 4$ ;;IF NOT INTERNAL, GO DO A TEST
1930 004544 112737 000010 001424 MOVB #MAINT,MNTFLG ;;IF INTERNAL MODE NOW, SET THE MAINTENANCE BIT
1931 004552 000002 4$: RTI ;;GO DO THE TEST
1932 004554 000406 BRW: 406
1933 004556 000005 $MXCNT: 5 ;;MAX. NUMBER OF ITERATIONS
1934
1935 ;;CHECK FOR FREEZE ON CURRENT DATA
1936 -----
1937
1938 004560 032777 001000 174516 .SCOP1: BIT #SW09,@SWR ;;IS SW09=1(SET)?
1939 004566 001405 BEQ 1$ ;;BR IF NOT SET.
1940 004570 005737 001364 TST LOCK ;;IS THERE A TIGHT LOOP SPECIFIED?
1941 004574 001402 BEQ 1$ ;;IF NO, RETURN
1942 004576 013716 001364 MOV LOCK,(SP) ;;IF YES, GOTO THE ADDRESS IN LOCK.
1943 004602 000002 1$: RTI ;;GO BACK.
1944
1945 004604 032777 010000 174472 .TYPE: BIT #SW12,@SWR ;;INHIBIT ALL PRINTOUT??
1946 004612 001403 BEQ $TYPE ;;IF NOT, GO TYPE
1947 004614 062716 000002 ADD #2,(SP) ;;SKIP OVER MESSAGE POINTER
1948 004620 000002 RTI ;;RETURN TO WHERE PROCEDURE WAS INVOKED
1949
1950 .SBTTL TYPE ROUTINE
1951
1952 ;;*****
1953 ;;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1954 ;;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1955 ;;*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
1956 ;;*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
1957 ;;*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
1958
1959 ;;*CALL:
1960 ;;*1) USING A TRAP INSTRUCTION
1961 ;;* TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
1962 ;;*OR
1963 ;;* TYPE
1964 ;;* MESADR
1965 ;;*
1966 004622 105737 001323 $TYPE: TSTB $TPFLG ;;IS THERE A TERMINAL?
1967 004626 100002 BPL 1$ ;;BR IF YES
1968 004630 000000 HALT ;;HALT HERE IF NO TERMINAL
1969 004632 000430 BR 3$ ;;LEAVE
1970 004634 010046 1$: MOV R0,-(SP) ;;SAVE R0
1971 004636 017600 000002 MOV @2(SP),R0 ;;GET ADDRESS OF ASCIZ STRING
1972 004642 122737 000001 001140 CMPB #APTENV,$ENV ;;RUNNING IN APT MODE
1973 004650 001011 BNE 62$ ;;NO,GO CHECK FOR APT CONSOLE
1974 004652 132737 000100 001141 BITB #APTPOOL,$ENVM ;;SPOOL MESSAGE TO APT

```

```

1975 004660 001405          BEQ      62$          ;;NO,GO CHECK FOR CONSOLE
1976 004662 010037 004672    MOV      RO,61$      ;;SETUP MESSAGE ADDRESS FOR APT
1977 004666 004737 005112    JSR      PC,$ATY3    ;;SPOOL MESSAGE TO APT
1978 004672 000000          .WORD    0           ;;MESSAGE ADDRESS
1979 004674 132737 000040 001141 61$:     BITB     #APTCSUP,$ENVM ;;APT CONSOLE SUPPRESSED
1980 004702 001003          BNE      60$          ;;YES,SKIP TYPE OUT
1981 004704 112046          2$:     MOVB     (RO)+,-(SP) ;;PUSH CHARACTER TO BE TYPED ONTO STACK
1982 004706 001005          BNE      4$           ;;BR IF IT ISN'T THE TERMINATOR
1983 004710 005726          TST      (SP)+        ;;IF TERMINATOR POP IT OFF THE STACK
1984 004712 012600          60$:     MOV      (SP)+,RO  ;;RESTORE RO
1985 004714 062716 000002    3$:     ADD      #2,(SP)   ;;ADJUST RETURN PC
1986 004720 000002          RTI                    ;;RETURN
1987 004722 122716 000011    4$:     CMPB     #HT,(SP)   ;;BRANCH IF <HT>
1988 004726 001430          BEQ      8$           ;;
1989 004730 122716 000200    CMPB     #CRLF,(SP)   ;;BRANCH IF NOT <CRLF>
1990 004734 001006          BNE      5$           ;;
1991 004736 005726          TST      (SP)+        ;;POP <CR><LF> EQUIV
1992 004740 104402          TYPE                    ;;TYPE A CR AND LF
1993 004742 001357          $CRLF
1994 004744 105037 005100    CLRB     $CHARCNT     ;;CLEAR CHARACTER COUNT
1995 004750 000755          BR      2$           ;;GET NEXT CHARACTER
1996 004752 004737 005034    5$:     JSR      PC,$TYPEC  ;;GO TYPE THIS CHARACTER
1997 004756 123726 001322    6$:     CMPB     $+ILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
1998 004762 001350          BNE      2$           ;;IF NO GO GET NEXT CHAR.
1999 004764 013746 001320    MOV      $NULL,-(SP)  ;;GET # OF FILLER CHARS. NEEDED
2000                                     ;;AND THE NULL CHAR.
2001 004770 105366 000001    7$:     DECB     1(SP)     ;;DOES A NULL NEED TO BE TYPED?
2002 004774 002770          BLT      6$           ;;BR IF NO--GO POP THE NULL OFF OF STACK
2003 004776 004737 005034    JSR      PC,$TYPEC  ;;GO TYPE A NULL
2004 005002 105337 005100    DECB     $CHARCNT     ;;DO NOT COUNT AS A COUNT
2005 005006 000770          BR      7$           ;;LOOP
2006
2007                                     ;HORIZONTAL TAB PROCESSOR
2008
2009 005010 112716 000040    8$:     MOVB     #' ,(SP)  ;;REPLACE TAB WITH SPACE
2010 005014 004737 005034    9$:     JSR      PC,$TYPEC  ;;TYPE A SPACE
2011 005020 132737 000007 005100    BITB     #7,$CHARCNT  ;;BRANCH IF NOT AT
2012 005026 001372          BNE      9$           ;;TAB STOP
2013 005030 005726          TST      (SP)+        ;;POP SPACE OFF STACK
2014 005032 000724          BR      2$           ;;GET NEXT CHARACTER
2015 005034 105777 174254    $TYPEC: TSTB     @STPS    ;;WAIT UNTIL PRINTER IS READY
2016 005040 100375          BPL      $TYPEC
2017 005042 116677 000002 174246    MOVB     2(SP),@STPB  ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2018 005050 122766 000015 000002    CMPB     #CR,2(SP)   ;;IS CHARACTER A CARRIAGE RETURN?
2019 005056 001003          BNE      1$           ;;BRANCH IF NO
2020 005060 105037 005100    CLRB     $CHARCNT     ;;YES--CLEAR CHARACTER COUNT
2021 005064 000406          BR      $TYPEX        ;;EXIT
2022 005066 122766 000012 000002    1$:     CMPB     #LF,2(SP) ;;IS CHARACTER A LINE FEED?
2023 005074 001402          BEQ      $TYPEX        ;;BRANCH IF YES
2024 005076 105227          INCB     (PC)+        ;;COUNT THE CHARACTER
2025 005100 000000          $CHARCNT: .WORD    0  ;;CHARACTER COUNT STORAGE
2026 005102 000207          $TYPEX:  RTS      PC
2027
2028                                     .SBTTL  APT COMMUNICATIONS ROUTINE
2029
2030                                     ;:*****

```

```

2031 005104 112737 000001 005350 SATY1: MOVB #1,$FFLG ;;TO REPORT FATAL ERROR
2032 005112 112737 000001 005346 SATY3: MOVB #1,$MFLG ;;TO TYPE A MESSAGE
2033 005120 000403
2034 005122 112737 000001 005350 SATY4: MOVB #1,$FFLG ;;TO ONLY REPORT FATAL ERROR
2035 005130 SATYC:
2036 005130 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
2037 005132 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
2038 005134 105737 005346 TSTB $MFLG ;;SHOULD TYPE A MESSAGE?
2039 005140 001450 BEQ 5$ ;;IF NOT: BR
2040 005142 122737 000001 001140 CMPB #APTENV,$ENV ;;OPERATING UNDER APT?
2041 005150 001031 BNE 3$ ;;IF NOT: BR
2042 005152 132737 000100 001141 BITB #APTPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
2043 005160 001425 BEQ 3$ ;;IF NOT: BR
2044 005162 017600 000004 MOV @4(SP),R0 ;;GET MESSAGE ADDR.
2045 005166 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
2046 005174 005737 001120 1$: TST $MSGTYPE ;;SEE IF DONE W/ LAST XMISSION?
2047 005200 001375 BNE 1$ ;;IF NOT: WAIT
2048 005202 010037 001134 MOV R0,$MSGAD ;;PUT ADDR IN MAILBOX
2049 005206 105720 2$: TSTB (R0)+ ;;FIND END OF MESSAGE
2050 005210 001376 BNE 2$
2051 005212 163700 001134 SUB $MSGAD,R0 ;;SUB START OF MESSAGE
2052 005216 006200 ASR R0 ;;GET MESSAGE LGTH IN WORDS
2053 005220 010037 001136 MOV R0,$MSGLGT ;;PUT LENGTH IN MAILBOX
2054 005224 012737 000004 001120 MOV #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
2055 005232 000413 BR 5$
2056 005234 017637 000004 005260 3$: MOV @4(SP),4$ ;;PUT MSG ADDR IN JSR LINKAGE
2057 005242 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDRESS
2058 005250 013746 177776 MOV 177776,-(SP) ;;PUSH 177776 ON STACK
2059 005254 004737 004622 JSR PC,$TYPE ;;CALL TYPE MACRO
2060 005260 000000 4$: .WORD 0
2061 005262 5$:
2062 005262 105737 005350 10$: TSTB $FFLG ;;SHOULD REPORT FATAL ERROR?
2063 005266 001416 BEQ 12$ ;;IF NOT: BR
2064 005270 005737 001140 TST $ENV ;;RUNNING UNDER APT?
2065 005274 001413 BEQ 12$ ;;IF NOT: BR
2066 005276 005737 001120 11$: TST $MSGTYPE ;;FINISHED LAST MESSAGE?
2067 005302 001375 BNE 11$ ;;IF NOT: WAIT
2068 005304 017637 000004 001122 MOV @4(SP),$FATAL ;;GET ERROR #
2069 005312 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
2070 005320 005237 001120 INC $MSGTYPE ;;TELL APT TO TAKE ERROR
2071 005324 105037 005350 12$: CLRB $FFLG ;;CLEAR FATAL FLAG
2072 005330 105037 005347 CLRB $LFLG ;;CLEAR LOG FLAG
2073 005334 105037 005346 CLRB $MFLG ;;CLEAR MESSAGE FLAG
2074 005340 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
2075 005342 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
2076 005344 000207 RTS PC ;;RETURN
2077 005346 000 SMFLG: .BYTE 0 ;;MESSG. FLAG
2078 005347 000 $LFLG: .BYTE 0 ;;LOG FLAG
2079 005350 000 $FFLG: .BYTE 0 ;;FATAL FLAG
2080 005352 .EVEN
2081 000200 APTSIZE=200
2082 000001 APTENV=001
2083 000100 APTPOOL=100
2084 000040 APTCSUP=040
2085
2086 ;STRING INPUT ROUTINE

```

```

2087
2088
2089 005352 010346      .INSTR: MOV    R3,-(SP)      :SAVE R3 ON STACK
2090 005354 010446      MOV    R4,-(SP)      :SAVE R4 ON STACK
2091 005356 017637 000004 005374      MOV    @4(SP),MSG     :GET THE ADDRESS OF THE MESSAGE TO BE PRINTED
2092 005364 062766 000002 000004      ADD    #2,4(SP)      :POINT TO INSTRUCTION AFTER ADDRESS POINTER
2093 005372 104402      .INST1: TYPE           :PRINT THE MESSAGE
2094 005374 000000      .MSG:  0             :MESSAGE IS POINTED TO FROM HERE
2095 005376 012704 010304      MOV    #INBUF,R4     :POINT R4 TO THE INPUT BUFFER
2096 005402 012703 000007      MOV    #7,R3         :SET THE MAXIMUM NUMBER OF CHARACTERS ALLOWED
2097 005406 105777 173676      1$:  TSTB @STKS      :HAS A CHARACTER BEEN RECEIVED?
2098 005412 100375      BPL    1$            :IF NO, KEEP WAITING FOR IT
2099 005414 117714 173672      MOVB @STKB,(R4)     :IF YES, SAVE IT IN THE INPUT BUFFER
2100 005420 142714 000200      BICB  #200,(R4)     :KEEP ONLY THE 7-BIT ASCII INFORMATION
2101 005424 122427 000015      CMPB  (R4)+,#15     :IS THIS CHARACTER A LINE FEED?
2103 005432 105777 173656      2$:  TSTB @STPS      :IF NOT, CHECK TO SEE IF THE CHARACTER CAN PRINT
2104 005436 100375      BPL    2$            :IF WE CAN'T, WAIT UNTIL WE CAN
2105 005440 017777 173646 173650      MOV    @STKB,@STPB   :ECHO THE CHARACTER BACK
2106 005446 005303      DEC    R3            :REDUCE THE NUMBER OF CHARACTERS RECEIVED
2107 005450 001356      BNE    1$            :IF WE DON'T HAVE 7, GO GET SOME MORE
2108 005452 012604      MOV    (SP)+,R4     :IF WE HAVE 7, RESTORE R4
2109 005454 012603      MOV    (SP)+,R3     :RESTORE R3
2110 005456 010346      .INSTE: MOV    R3,-(SP) :SAVE R3 ON THE STACK
2111 005460 010446      MOV    R4,-(SP)     :SAVE R4 ON THE STACK
2112 005462 104402 001356      TYPE   ,SQUES       :PRINT A QUESTION MARK... WHAT'S GOING ON?
2113 005466 000741      BR     .INST1       :GO PRINT THE MESSAGE AGAIN
2114 005470 012604      INSTR2: MOV    (SP)+,R4 :RESTORE R4
2115 005472 012603      MOV    (SP)+,R3     :RESTORE R3
2116 005474 000002      RTI                    :RETURN TO THE MAIN PROCEDURE
2117
2118      :CONVERT ASCII STRING TO OCTAL
2119
2120
2121 005476 010546      .PARAM: MOV    R5,-(SP) :SAVE R5 ON THE STACK
2122 005500 010446      MOV    R4,-(SP)     :SAVE R4 ON THE STACK
2123 005502 016605 000004      MOV    4(SP),R5     :GET THE SETUP INFORMATION POINTER
2124 005506 012537 005666      MOV    (R5)+,LOLIM  :SET THE LOW LIMIT FOR THE INPUT
2125 005512 012537 005670      MOV    (R5)+,HILIM  :SET THE HIGH LIMIT FOR THE INPUT
2126 005516 012537 005672      MOV    (R5)+,DEVADR  :SAVE THE ADDRESS WHERE THE RESULT WILL BE STORED
2127 005522 112537 005674      MOVB  (R5)+,LOBITS   :GET THE MASK OF THE INCORRECT BITS
2128 005526 112537 005675      MOVB  (R5)+,ADRCNT  :GET THE COUNT OF ITEMS TO BE STORED
2129 005532 010566 000004      MOV    R5,4(SP)     :POINT TO WHERE MAIN LINE PROGRAM WILL RESUME
2130 005536 005005      PARAM1: CLR    R5     :INITIALIZE THE ASCII TO OCTAL RESULT WORD
2131 005540 012704 010304      MOV    #INBUF,R4     :POINT TO THE INPUT BUFFER
2132 005544 122714 000015      CMPB  #15,(R4)       :IS THIS CHARACTER A CARRIAGE RETURN?
2133 005550 001420      BEQ   PARERR        :IF SO, PRINT THE MESSAGE AGAIN
2134 005552 121427 000060      1$:  CMPB  (R4),#60    :IS THIS CHARACTER BELOW THE NUMERIC RANGE?
2135 005556 002415      BLT   PARERR        :IF SO, GO PRINT THE MESSAGE AGAIN
2136 005560 121427 000067      CMPB  (R4),#67      :IS THIS CHARACTER ABOVE THE NUMERIC RANGE?
2137 005564 003012      BGT   PARERR        :IF SO, GO PRINT THE MESSAGE AGAIN
2138 005566 142714 000060      BICB  #60,(R4)       :ISOLATE THE NUMBER THE CHARACTER REPRESENTS
2139 005572 152405      BISB  (R4)+,R5       :CONCATENATE THESE BITS TO THE ALREADY EXISTING STRING
2140 005574 122714 000015      CMPB  #15,(R4)       :IS THE NEXT CHARACTER A CARRIAGE RETURN?
2141 005600 001406      BEQ   LIMITS        :IF SO, GO SEE IF NUMBER IS WITHIN LIMITS
2142 005602 006305      ASL   R5             :CLEAR BIT POSITION 0, MOVE EXISTING STRING TO LEFT

```

```

2143 005604 006305          ASL    R5          ;CLEAR POSITION 1, MOVE STRING TO LEFT AGAIN
2144 005606 006305          ASL    R5          ;MOVE THE STRING ONE MORE TIME TO MAKE ROOM FOR
2145                          ;NEXT THREE BITS
2146 005610 000760          BR     1$          ;GO GET THE NEXT CHARACTER
2147 005612 104404          PARERR: INSTER    ;THERE WAS AN ERROR... GO PRINT MESSAGE AGAIN
2148 005614 000750          BR     PARAM1     ;TRY GETTING THE PARAMETERS AGAIN
2149
2150                          ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
2151                          ;-----
2152
2153 005616 020537 005670          LIMITS: CMP     R5,HILIM      ;DOES RESULT EXCEED ITS MAXIMUM CORRECT VALUE?
2154 005622 101373          BHI    PARERR      ;IF YES, GO PRINT THE MESSAGE AGAIN
2155 005624 020537 005666          CMP     R5,LOLIM     ;IS THE RESULT LOWER THAN ALLOWED?
2156 005630 103770          BLO    PARERR      ;IF YES, GO PRINT THE MESSAGE AGAIN
2157 005632 133705 005674          BITB   LOBITS,R5    ;ARE ANY INCORRECT BITS SET IN THE RESULT?
2158 005636 001365          BNE    PARERR      ;IF SO, GO PRINT THE MESSAGE AGAIN
2159
2160                          ;STORE NUMBER AT SPECIFIED ADDRESS
2161
2162 005640 013704 005672          1$:   MOV     DEVADR,R4     ;POINT TO THE LOCATION WHERE THE RESULT WILL BE STORED
2163 005644 010524          MOV     R5,(R4)+      ;STORE THE RESULT
2164 005646 062705 000002          ADD     #2,R5         ;CALCULATE THE NEXT DATUM
2165 005652 105337 005675          DECB   ADRCNT        ;REDUCE COUNT OF STORED RESULTS. IS IT EXCEEDED?
2166 005656 001372          BNE    1$           ;IF NOT, GO STORE THE NEXT DATUM
2167 005660 012604          MOV     (SP)+,R4      ;RESTORE R4
2168 005662 012605          MOV     (SP)+,R5      ;RESTORE R5
2169 005664 000002          RTI                    ;RETURN TO THE MAIN PROGRAM
2170
2171 005666 000000          LOLIM: 0              ;LOWEST ACCEPTABLE VALUE
2172 005670 000000          HILIM: 0              ;HIGHEST ACCEPTABLE
2173 005672 000000          DEVADR: 0            ;LOCATION WHERE RESULT WILL BE STORED
2174 005674 000          LOBITS: .BYTE 0      ;INCORRECT BITS MASK
2175 005675 000          ADRCNT: .BYTE 0     ;COUNT OF ITEMS TO BE STORED
2176
2177                          ;SAVE PC OF TEST THAT FAILED AND R0-R5
2178                          ;-----
2179
2180 005676 016637 000004 001404 .SAV05: MOV     4(SP),SAVPC    ;SAVE R7 (PC)
2181
2182                          ;SAVE R0-R5
2183
2184 005704 010537 001340          SV05:  MOV     R5,$REG5     ;SAVE R5
2185 005710 010437 001336          MOV     R4,$REG4     ;SAVE R4
2186 005714 010337 001334          MOV     R3,$REG3     ;SAVE R3
2187 005720 010237 001332          MOV     R2,$REG2     ;SAVE R2
2188 005724 010137 001330          MOV     R1,$REG1     ;SAVE R1
2189 005730 010037 001326          MOV     R0,$REG0     ;SAVE R0
2190 005734 000002          RTI                    ;LEAVE.
2191
2192                          ;RESTORE R0-R5
2193
2194 005736 013700 001326          .RES05: MOV     $REG0,R0    ;RESTORE R0
2195 005742 013701 001330          MOV     $REG1,R1     ;RESTORE R1
2196 005746 013702 001332          MOV     $REG2,R2     ;RESTORE R2
2197 005752 013703 001334          MOV     $REG3,R3     ;RESTORE R3
2198 005756 013704 001336          MOV     $REG4,R4     ;RESTORE R4

```



```

2199 005762 013705 001340      MOV      $REG5,R5      ;RESTORE R5
2200 005766 000002      RTI                    ;LEAVE
2201
2202      ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
2203      -----
2204
2205 005770 104402 001357      .CONVR: TYPE          ,SCLRF          ;PRINT A CARRIAGE RETURN
2206 005774 010046      .CNVRT: MOV           R0,-(SP)        ;SAVE R0
2207 005776 010146      MOV           R1,-(SP)        ;SAVE R1
2208 006000 010346      MOV           R3,-(SP)        ;SAVE R3
2209 006002 010446      MOV           R4,-(SP)        ;SAVE R4
2210 006004 010546      MOV           R5,-(SP)        ;SAVE R5
2211 006006 017601 000012      MOV           @12(SP),R1      ;PLACE THE ADDRESS OF THE ARGUMENTS IN R1
2212 006012 062766 000002 000012      ADD           #2,12(SP)      ;POINT TO WHERE MAIN PROGRAM WILL RESUME
2213 006020 012137 006144      MOV           (R1)+,WRDCNT    ;GET NUMBER OF WORDS TO BE PRINTED
2214 006024 112105      1$: MOVB        (R1)+,R5      ;GET THE NUMBER OF CHARACTERS TO BE PRINTED
2215 006026 112100      MOVB        (R1)+,R0      ;GET THE NUMBER OF SPACES TO PRINT
2216 006030 013104      MOV           @1(R1)+,R4      ;COPY THE WORD TO BE CONVERTED
2217 006032 110537 006146      MOVB        R5,CHRCNT      ;COPY THE CHARACTER COUNT
2218 006036 010403      3$: MOV           R4,R3      ;COPY THE ARGUMENT WORD AGAIN
2219 006040 042703 177770      BIC           #^C<7>,R3      ;ISOLATE THREE BITS TO BE TREATED AS A CHARACTER
2220 006044 062703 000060      ADD           #060,R3      ;MAKE AN ASCII CHARACTER OUT OF THEM
2221 006050 110346      MOVB        R3,-(SP)        ;SAVE THAT CHARACTER
2222 006052 006004      ROR           R4            ;MOVE THE NEXT THREE BITS INTO PLACE
2223 006054 006204      ASR           R4            ;MOVE THEM AGAIN
2224 006056 006204      ASR           R4            ;AND FINALLY A THIRD TIME
2225 006060 005305      DEC           R5            ;REDUCE CHARACTER COUNT. ARE ALL CHARACTERS
2226      ;BUILT?
2227 006062 001365      BNE           3$           ;IF NO, GO BUILD THE NEXT ONE.
2228 006064 012703 010410      MOV           #MDATA,R3      ;NOW POINT TO WHERE NUMBER WILL BE PRINTED FROM
2229 006070 112623      4$: MOVB        (SP)+,(R3)+    ;STORE THE CHARACTER, STARTING WITH THE MOST
2230 006072 105337 006146      DECB        CHRCNT        ;REDUCE COUNT. ARE ALL CHARACTERS TRANSFERRED?
2231 006076 001374      BNE           4$           ;IF NO, GO TRANSFER ANOTHER
2232 006100 105700      TSTB        R0            ;ARE ANY SPACES TO BE PRINTED?
2233 006102 001404      BEQ           6$           ;IF NO, DON'T SET UP ANY
2234 006104 112723 000040      5$: MOVB        #040,(R3)+    ;ADD A SPACE TO THE OUTPUT BUFFER
2235 006110 105300      DECB        R0            ;REDUCE THE COUNT. SHOULD WE PRINT MORE?
2236 006112 001374      BNE           5$           ;IF YES, GO ADD ANOTHER SPACE
2237 006114 105013      6$: CLRB        (R3)        ;TERMINATE THE OUTPUT BUFFER WITH A ZERO
2238 006116 104402 010410      TYPE        ,MDATA        ;PRINT THE STRING WE JUST BUILT
2239 006122 005337 006144      DEC         WRDCNT        ;REDUCE THE WORD COUNT. ARE ANY MORE WORDS LEFT?
2240 006126 001336      BNE           1$           ;IF YES, GO CONVERT THEM
2241 006130 012605      MOV         (SP)+,R5      ;RESTORE R5
2242 006132 012604      MOV         (SP)+,R4      ;RESTORE R4
2243 006134 012603      MOV         (SP)+,R3      ;RESTORE R3
2244 006136 012601      MOV         (SP)+,R1      ;RESTORE R1
2245 006140 012600      MOV         (SP)+,R0      ;RESTORE R0
2246 006142 000002      RTI                    ;RETURN TO THE MAIN PROGRAM
2247 006144 000000      WRDCNT: 0
2248 006146      000      CHRCNT: .BYTE          ;NUMBER OF CHARACTERS TO PRINT
2249 006147      000      SPACNT: .BYTE          ;NUMBER OF SPACES TO PRINT
2250
2251 006150 000000      BINWRD: 0
2252
2253
2254      ;TRAP DISPATCH SERVICE

```

```

2255                                     ;ARGUMENT OF TRAP IS EXTRACTED
2256                                     ;AND USED AS OFFSET TO OBTAIN POINTER
2257                                     ;TO SELECTED SUBROUTINE
2258
2259 006152 010046                                .TRPSR: MOV    R0,-(SP)          ;SAVE R0. USE R0 TO FIND TRAP ROUTINE
2260 006154 016600 000002                       MOV    2(SP),R0          ;GET TRAP ADDRESS
2261 006160 005740                                TST    -(R0)            ;GET TRAP
2262 006162 111000                                MOVSB  (R0),R0          ;GET RIGHT BYTE OF TRAP(TRAP OFFSET)
2263 006164 006300                                ASL    R0               ;POSITION OFFSET FOR TABLE INDEXING
2264 006166 016000 001742                       MOV    ,TRPTAB(R0),R0   ;PLACE INDEXED ADDRESS OF TABLE IN R0
2265 006172 000200                                RTS    R0               ;TRANSFER TO THAT ADDRESS AND RESTORE OLD R0
2266
2267                                     ;DEVICE CLEAR ROUTINE
2268                                     ;ISSUE A DEVICE CLEAR
2269                                     ;-----
2270 006174                                .DEVICE.CLR:
2271 006174 052777 000020 173606                 BIS    #DCLR,@DZVCSR   ;SET DCLR
2272 006202 032777 000020 173600 1$:          BIT    #DCLR,@DZVCSR   ;DID IT CLEAR?
2273 006210 001374                                BNE    1$              ;BR IF NO
2274 006212 000002                                RTI                    ;EXIT ROUTINE
2275
2276                                     ;ROUTINE TO HANDLE MAINTENANCE BIT SETTING WITH DEVICE CLEAR
2277                                     ;-----
2278 006214 104413                                .DCLASM:DEVICE.CLR     ;ISSUE A DEVICE CLEAR
2279 006216 153777 001424 173564                 BISB   MNTFLG,@DZVCSR ;LOAD THE MAINTENANCE BIT IF IT IS I MODE
2280 006224 000002                                RTI                    ;RETURN TO CALLING ROUTINE
2281
2282 006226                                .DELAY:
2283 006226 010046                                MOV    R0,-(SP)        ;SAVE R0
2284 006230 013700 006244                       MOV    DLYCNT,R0      ;SET COUNT
2285 006234 005300 1$:                          DEC    R0              ;DELAY
2286 006236 001376                                BNE    1$              ;
2287 006240 012600                                MOV    (SP)+,R0       ;RESTORE R0
2288 006242 000002                                RTI                    ;LEAVE ROUTINE
2289 006244 000001  DLYCNT: .WORD    1          ;PATCHABLE LOC FOR MORE TIME
2290
2291                                     ;ADVANCE TO NEXT TEST HANDLER
2292                                     ;-----
2293
2294 006246 013716 001362                                .ADVANCE:MOV    NEXT,(SP) ;CRUNCH STACK WITH ADDRESS OF SCOPE CALL
2295 006252 005037 001364                       CLR    LOCK           ;RESET TIGHT LOOP ADDRESS
2296 006256 000002                                RTI                    ;CHECK TO SEE IF OLD TEST GETS REPEATED
2297
2298                                     ;ROUTINE TO SHIFT LINE POINTER
2299                                     ;AND SWITCH TESTS IF NECESSARY
2300                                     ;-----
2301 006260 106302                                .SHIFT:ASLB   R2        ;POINT TO THE NEXT LINE
2302 006262 032702 000020                       BIT    #BIT4,R2       ;HAVE WE PASSED ALL LINE POINTERS?
2303 006266 001402                                BEQ    1$             ;IF NOT, RETURN TO THE TEST
2304 006270 022626                                POP2SP                ;REMOVE THE TRAP CALL FROM THE STACK
2305 006272 104400                                ADVANCE                ;GO TO THE NEXT TEST
2306 006274 000002 1$:                          RTI                    ;RETURN TO THE PRESENT TEST
2307

```

```

2308                                     ;LINE PARAMETER REGISTER SETUP ROUTINE
2309
2310 006276 010146                       .LPRSET:MOV R1,-(SP)           ;SAVE CONTENTS OF R1
2311 006300 010246                       MOV R2,-(SP)           ;SAVE CONTENTS OF R2
2312 006302 013701 001370               MOV PAR,R1            ;MOVE DEFAULT PARAM. INTO R1
2313 006306 012702 000001               MOV #1,R2             ;INIT. FOR LINE 1
2314 006312 010177 173502               1$: MOV R1,@DZVLPR    ;LOAD PARAM. REGISTER
2315 006316 005201                       INC R1                ;SET R1 FOR NEXT LINE
2316 006320 106302                       ASLB R2               ;SET R2 FOR NEXT LINE
2317 006322 032702 000020               BIT #BIT4,R2         ;ALL LINES DONE?
2318 006326 001771                       BEQ 1$               ;IF NO LOAD NEXT LINE
2319 006330 012602                       MOV (SP)+,R2         ;RELOAD R2
2320 006332 012601                       MOV (SP)+,R1         ;RELOAD R1
2321 006334 000002                       RTI                  ;RETURN
2322
2323                                     ;ROUTINE TO ZERO DATA BUFFER
2324
2325 006336 010046                       .BUFSET:MOV R0,-(SP)  ;SAVE CONTENTS OF R0
2326 006340 012700 001426               MOV #TDO,R0          ;SET R0 TO TOP OF BUFFER
2327 006344 005020                       1$: CLR (R0)+         ;CLEAR BUFFER LOCATION
2328 006346 022700 001446               CMP #STOP,R0        ;IS BUFFER ALL CLEARED
2329 006352 001374                       BNE 1$              ;IF NOT CLEAR NEXT LOCATION
2330 006354 012600                       MOV (SP)+,R0        ;RELOAD R0
2331 006356 000002                       RTI                  ;RETURN
2332
2333                                     ;ERROR HANDLER
2334 -----
2335
2336 006360 004737 007006 172712          $ERROR: JSR PC,SERV.G   ;FIND OUT IF <^G> WAS HIT
2337 006364 032777 010000 172712          BIT #SW12,@SWR      ;BELL ON ERROR?
2338 006372 001406                       BEQ XBX             ;BR IF NO BELL
2339 006374 105777 172714               TSTB @STPS         ;TTY READY.
2340 006400 100003                       BPL XBX            ;DON'T WAIT IF TTY NOT READY.
2341 006402 112777 000207 172706          MOVB #207,@STPB    ;PUSH A BELL AT THE TTY.
2342 006410 032777 020000 172666          XBX: BIT #SW13,@SWR ;DELETE ERROR PRINT OUT?
2343 006416 001113                       BNE HALTS          ;BR IF NO PRINT OUT WANTED.
2344 006420 021637 001262               CMP (SP),$ERRPC    ;WAS THIS ERROR FOUND LAST TIME?
2345 006424 001404                       BEQ 1$             ;BR IF YES
2346 006426 011637 001262               MOV (SP),$ERRPC    ;RECORD BEING HERE
2347 006432 105037 001247               CLRB $ERFLG       ;PREPARE HEADER
2348 006436 104407                       1$: SAVO5          ;SAVE ALL PROC REGISTERS
2349 006440 011605                       MOV (SP),R5        ;GET THE PC OF ERROR
2350 006442 162705 000002               SUB #2,R5          ;GET ADDRESS OF TRAP CALL
2351 006446 011504                       MOV (R5),R4        ;GET ERROR INSTRUCTION
2352 006450 110437 001260               MOVB R4,$ITEMB    ;COPY TEST NUMBER FOR APT HANDLING
2353 006454 006304                       ASL R4             ;MULT BY TWO
2354 006456 061504                       ADD (R5),R4        ;DOUBLE IT
2355 006460 006304                       ASL R4             ;MULT AGAIN
2356 006462 042704 177001               BIC #177001,R4    ;CLEAR JUNK
2357 006466 062704 016214               ADD #.ERRTAB,R4   ;GET POINTER
2358 006472 012437 006616               MOV (R4)+,ERRMSG  ;GET ERROR MESSAGE
2359 006476 012437 006630               MOV (R4)+,DATAHD  ;GET DATA HEADRER
2360 006502 011437 006642               MOV (R4),DATABP   ;GET DATA TABLE
2361 006506 105737 001247               TSTB $ERFLG       ;TYPE HEADER
2362 006512 001403                       BEQ TYPMSG        ;BR IF YES
2363 006514 005737 006642               TST DATABP        ;DOES DATA TABLE EXIST?

```

```

2364 006520 001044 .
2365 006522 104402 001357 TYPMSG: BNE TYPDAT :BR IF YES.
2366 006526 104402 001357 TYPE ,SCRLF :TYPE A CARRIAGE RETURN
2367 006532 005737 001364 TYPE ,SCRLF :AND TYPE ANOTHER
2368 006536 001402 TST LOCK
2369 006540 104402 007721 BEQ 1$
2370 006544 104402 007707 1$: TYPE ,MASTEK
2371 006550 104412 007000 TYPE ,MTSTN
2372 006554 104402 010001 CNVRT ,XTSTN :SHOW IT
2373 006560 104412 006772 TYPE ,MERRPC :TYPE PC.
2374 006564 104402 007651 CNVRT ,ERTABO :SHOW IT
2375 006570 104412 004264 TYPE ,MCSRX
2376 006574 104402 001357 CNVRT ,XCSR
2377 006600 112737 177777 001247 TYPE ,SCRLF :GIVE A CR/LF
2378 006606 005737 006616 MOV#B #-1, SERFLG :NO MORE HEADER UNLESS NO DATA TABLE.
2379 006612 001402 TST ERRMSG :IS THERE AN ERROR MESSAGE?
2380 006614 104402 BEQ WTBS.FM :BR IF NO.
2381 006616 000000 ERRMSG: 0 :TYPE
2382 006620 WTBS.FM: : ERROR MESSAGE
2383 006620 005737 006630 TST DATAHD :DATA HEADER?
2384 006624 001402 BEQ TYPDAT :BR IF NO
2385 006626 104402 TYPE :TYPE
2386 006630 000000 DATAHD: 0 : DATA HEADER
2387 006632 005737 006642 TYPDAT: TST DATABP :DATA TABLE?
2388 006636 001402 BEQ RESREG :BR IF NO.
2389 006640 104411 CNVRT :SHOW
2390 006642 000000 DATABP: 0 : DATA TABLE
2391 006644 104410 RESREG: RES05 :RESTORE PROC REGISTERS
2392 006646 122737 000001 001140 HALTS: CMPB #APTENV, $ENV :IS APT RUNNING?
2393 006654 001007 BNE 15$ :SKIP APT CALL IF NOT
2394 006656 113737 001260 006670 MOV#B $ITEMB, 5$ :COPY ERROR NUMBER
2395 006664 004737 005122 JSR PC, $ATY4 :CALL APT SERVICE
2396 006670 000000 5$: .WORD 0 :ERROR NUMBER STUCK HERE
2397 006672 000777 10$: BR 10$ :LOCK UP HERE
2398 006674 022737 004250 000042 15$: CMP #SENDAD, @#42 :CHECK TO SEE IF IN ACT-11 MODE
2399 006702 001403 BEQ 20$ :IF SO, HANDLE ACCORDINGLY
2400 006704 005777 172374 TST @SWR :HALT ON ERROR?
2401 006710 100004 BPL EXITER :BR IF NO HALT ON ERROR
2402 006712 016677 000002 172366 20$: MOV 2(SP), @DISPLAY :SHOW ERROR PC IN DATA DISPLAY
2403 006720 000000 HALT :HALT
2404 006722 005237 001256 EXITER: INC $ERTTL :UPDATE ERROR COUNT
2405 006726 004737 007006 JSR PC, $SERV.G :FIND OUT IF ^G WAS TYPED
2406 006732 032777 000400 172344 BIT #SW08, @SWR :GOTO TOP OF TEST?
2407 006740 001007 BNE 1$ :BR IF YES
2408 006742 032777 002000 172334 BIT #SW10, @SWR :GOTO NEXT TEST?
2409 006750 001407 BEQ 2$ :BR IF NO
2410 006752 013737 001362 001252 MOV NEXT, $LPADR :SET FOR NEXT TEST
2411 006760 012706 001120 1$: MOV #STACK, SP :RESET SP
2412 006764 000177 172262 JMP @SLPADR :GOTO SPECIFIED TEST
2413 006770 000002 2$: RTI :RETURN
2414 006772 000001 ERTABO: 1
2415 006774 006 002 .BYTE 6, 2
2416 006776 001404 SAVPC
2417 007000 000001 XTSTN: 1
2418 007002 002 002 .BYTE 2, 2
2419 007004 001246 $TSTNM

```

```

2420 007006 017746 172300      SERV.G: MOV    @STKB,-(SP)      ;OTHERWISE, GET THE LAST CHARACTER TYPED
2421 007012 042716 000200      BIC    #BIT7,(SP)           ;STRIP PARITY(EIGHTH) BIT
2422 007016 122726 000007      CMPB   #7,(SP)+            ;IS IT ^G?
2423 007022 001076          BNE    6$                  ;IF NOT, IGNORE INPUT
2424 007024 032777 004000 172256      BIT    #4000,@STKS         ;RX BUSY?
2425 007032 001365          BNE    SERV.G             ;BR IF YES
2426 007034 017737 172244 007242      MOV    @SWR,90$           ;SAVE (SWR).
2427 007042 104402 007222      1$:    TYPE   ,89$          ;TYPE HEADER FOR OLD SWITCH REGISTER
2428 007046 104412 007234      CNVRT  ,88$              ;TYPE THE NUMBER ITSELF
2429 007052 104402 007244      TYPE   ,91$              ;AFTER HAVING CONVERTED IT TO ASCII
2430 007056 105037 007250      CLRB   92$               ;CLEAR SWR CHANGE FLAG
2431 007062 005077 172216      CLR    @SWR              ;CLEAR THE SOFTWARE SWITCH REGISTER
2432 007066 105777 172216      3$:    TSTB   @STKS        ;WAIT FOR DONE.
2433 007072 100375          BPL    3$                 ;CONTINUE WAITING FOR IT
2434 007074 017746 172212      MOV    @STKB,-(SP)        ;PUT THE CHARACTER ON THE STACK
2435 007100 042716 000200      BIC    #BIT7,(SP)         ;STRIP PARITY BIT
2436 007104 122726 000015      CMPB   #15,(SP)+         ;IS IT THE CARRIAGE RETURN CHAR?
2437 007110 001433          BEQ    4$                 ;IF SO, GO PRINT CRLF
2438 007112 105777 172176      2$:    TSTB   @STPS        ;IS THE OUTPUT BUFFER AVAILABLE
2439 007116 100375          BPL    2$                 ;IF NOT, WAIT FOR IT TO BE READY
2440 007120 105237 007250      INCB   92$               ;INDICATE THAT THE SWR WAS CHANGED
2441 007124 014677 172166      MOV    -(SP),@STPB       ;PLACE THE CHARACTER THERE(ECHO BACK)
2442 007130 000241          CLC                      ;GET READY TO ROTATE
2443 007132 006177 172146      ROL    @SWR              ;MOVE THE EXISTING BITS OVER
2444 007136 006177 172142      ROL    @SWR              ;TO MAKE ROOM FOR THE INCOMING
2445 007142 006177 172136      ROL    @SWR              ;THREE BITS FROM THIS CHARACTER
2446 007146 103735          BCS    1$                 ;ERROR
2447 007150 022627 000060      CMP    (SP)+,#60         ;IS IT LOWER THAN 0?
2448 007154 002732          BLT    1$                 ;IF SO, GO ASK AGAIN
2449 007156 026627 177776 000067      CMP    -2(SP),#67       ;IS IT HIGHER THAN 7?
2450 007164 003326          BGT    1$                 ;IF SO, GO ASK AGAIN
2451 007166 042746 77770       BIC    #^C<7>,-(SP)      ;ISOLATE INFORMATION BITS
2452 007172 052677 172106      BIS    (SP)+,@SWR        ;ADD THEM TO THE SWITCH REGISTER
2453 007176 000733          BR     3$                 ;GO CHECK FOR THE NEXT CHARACTER
2454 007200 105737 007250      4$:    TSTB   92$          ;HAS THE SWR BEEN CHANGED?
2455 007204 001003          BNE    5$                 ;IF YES GO TYPE CRLF
2456 007206 013777 007242 172070      MOV    90$,@SWR          ;IF NOT RESTORE SWR
2457 007214 104402 001357      5$:    TYPE   ,$CRLF       ;TYPE A CARRIAGE RETURN AND LINE FEED
2458 007220 000207          6$:    RTS    PC          ;RETURN TO CALLING PROCEDURE
2459
2460 007222 020200 051450 051127 89$:    .ASCIZ  <200>? (SWR)=/?
2461 007230 036451 000057
2462
2463 007234 000001          .EVEN
2464 007236 006 000       88$:    1
2465 007240 007242          .BYTE  6,0
2466 007242 000000          90$:    .WORD  0
2467 007244 036457 000057      91$:    .ASCIZ  ?/=/?
2468 007250 000          92$:    .BYTE  0
2469
2470          .EVEN
2471          .SBTTL  POWER DOWN AND UP ROUTINES
2472
2473          ;:*****
2474          ;:POWER DOWN ROUTINE
2475          $PWRDN: MOV    #SILLUP,@PWRVEC ;:SET FOR FAST UP
2476          MOV    #340,@PWRVEC+2 ;:PRIO:7

```

```

2476 007266 010046      MOV      R0,-(SP)      ::PUSH R0 ON STACK
2477 007270 010146      MOV      R1,-(SP)      ::PUSH R1 ON STACK
2478 007272 010246      MOV      R2,-(SP)      ::PUSH R2 ON STACK
2479 007274 010346      MOV      R3,-(SP)      ::PUSH R3 ON STACK
2480 007276 010446      MOV      R4,-(SP)      ::PUSH R4 ON STACK
2481 007300 010546      MOV      R5,-(SP)      ::PUSH R5 ON STACK
2482 007302 017746 171776      MOV      @SWR,-(SP)    ::PUSH @SWR ON STACK
2483 007306 010637 007422      MOV      SP,$$AVR6    ::SAVE SP
2484 007312 012737 007324 000024      MOV      #SPWRUP,@PWRVEC ::SET UP VECTOR
2485 007320 000000      HALT
2486 007322 000776      BR      -2            ::HANG UP
2487
2488
2489
2490 007324 012737 007416 000024      ::POWER UP ROUTINE
2491 007332 013706 007422      SPWRUP: MOV      #SILLUP,@PWRVEC ::SET FOR FAST DOWN
2492 007336 005037 007422      MOV      $$AVR6,SP    ::GET SP
2493 007342 005237 007422      CLR      $$AVR6      ::WAIT LOOP FOR THE TTY
2494 007346 001375      1$: INC      $$AVR6    ::WAIT FOR THE INC
2495 007350 012677 171730      BNE      1$          ::OF WORD
2496 007354 012605      MOV      (SP)+,@SWR   ::POP STACK INTO @SWR
2497 007356 012604      MOV      (SP)+,R5    ::POP STACK INTO R5
2498 007360 012603      MOV      (SP)+,R4    ::POP STACK INTO R4
2499 007362 012602      MOV      (SP)+,R3    ::POP STACK INTO R3
2500 007364 012601      MOV      (SP)+,R2    ::POP STACK INTO R2
2501 007366 012600      MOV      (SP)+,R1    ::POP STACK INTO R1
2502 007370 012737 007252 000024      MOV      #SPWRDN,@PWRVEC ::SET UP THE POWER DOWN VECTOR
2503 007376 012737 000340 000026      MOV      #340,@PWRVEC+2 ::PRIO:7
2504 007404 104402      TYPE
2505 007406 007424      SPWRMG: .WORD  MPFAIL    ::REPORT THE POWER FAILURE
2506 007410 012716      MOV      (PC)+,(SP)   ::POWER FAIL MESSAGE POINTER
2507 007412 011012      SPWRAD: .WORD  RESTART  ::RESTART AT RESTART
2508 007414 000002      RTI
2510 007420 000776      BR      -2            ::BEFORE THE POWER DOWN WAS COMPLETE
2511 007422 000000      $$AVR6: 0            ::PUT THE SP HERE
2512 007424 050200 051127 043040      MPFAIL: .ASCIZ <200>/PWR FAILED. RESTART AT LAST TEST /
(2) 007467 200 047105 020104      MEPASS: .ASCIZ <200>/END PASS CVDZA-B /
(2) 007513 200 052522 047116      MR: .ASCIZ <200>/RUNNING /
(2) 007527 200 051120 043517      MERR2: .ASCIZ <200>/PROGRAM INDICATES NO DEVICES PRESENT./
(2) 007576 044600 051516 043125      MERR3: .ASCIZ <200>/INSUFFICIENT DATA!/
(2) 007622 046200 041517 020113      MLOCK: .ASCIZ <200>/LOCK ON SELECTED TEST/
(2) 007651 103 051123 020072      MCSR: .ASCIZ /CSR: /
(2) 007657 126 041505 020072      MVEC: .ASCIZ /VEC: /
(2) 007665 120 051501 042523      MPASSX: .ASCIZ /PASSES: /
(2) 007676 051105 047522 051522      MERRX: .ASCIZ /ERRORS: /
(2) 007707 124 051505 020124      MTSTN: .ASCIZ /TEST NO: /
(2) 007721 052 000040      MASTEK: .ASCIZ /* /
(2) 007724 052200 050131 020105      MNEW: .ASCIZ <200>/TYPE A BIT MAP OF DZV11'S DESIRED ACTIVE: /
(2) 010001 120 035103 000040      MERRPC: .ASCIZ /PC: /
(2) 010006 046600 050101 047440      XHEAD: .ASCIZ <200>/MAP OF DZV11 STATUS/<200>
(2) 010034 046600 046114 043505      MBADLN: .ASCIZ <200>/ILLEGAL ENTRY IN STAGGERED MODE/<200>
(2)
(2) 010076 000002
(2) 010100 006 003
2514 010102 001344
XSTATQ: 2
.BYTE 6.3
$TMP1

```

```

2515 010104 006 002 .BYTE 6.2
2516 010106 001346 $TMP2
2517 .EVEN
2518 :THIS ROUTINE ESTABLISHES WHICH MAINTENANCE MODE THE DEVICE IS IN
2519 :-----
2520 :E=EXTERNAL LOOP BACK
2521 :I=INTERNAL LOOP BACK
2522 :S=STAGGERED LOOP BACK
2523 010110 017605 000000 .SETFLG:MOV @ (SP),R5 :PICK UP ADDRESS OF TAG
2524 010114 042737 000040 010304 BIC #40,INBUF :STRIP LOWER CASE
2525 010122 122737 000105 010304 CMPB #'E,INBUF :IS IT EXTERNAL LOOP BACK ?
2526 010130 001005 BNE 4$ :NO
2527 010132 013715 010222 MOV 1$, (R5) :YES STORE INFO
2528 010136 105037 001424 CLRB MNTFLG :SET MAINT BIT =0
2529 010142 000422 BR 7$ :GET OUT
2530 010144 122737 000111 010304 4$: CMPB #'I,INBUF :IS IT INTERNAL LOOP BACK ?
2531 010152 001006 BNE 5$ :NO
2532 010154 013715 010224 MOV 2$, (R5) :YES STORE INFO
2533 010160 112737 000010 001424 MOVB #MAINT,MNTFLG :SET UP THE MAINTENANCE FLAG LOADER
2534 010166 000410 BR 7$ :GET OUT
2535 010170 122737 000123 010304 5$: CMPB #'S,INBUF :IS IT STAGGERED LOOP BACK ?
2536 010176 001007 BNE 6$ :WHAT ?
2537 010200 013715 010226 MOV 3$, (R5) :YES STORE INFO
2538 010204 105037 001424 CLRB MNTFLG :ZERO BITS
2539 010210 062716 000002 7$: ADD #2, (SP) :POP AROUND
2540 010214 000002 RTI
2541 010216 104404 6$: INSTER :RETRY
2542 010220 000733 BR .SETFLG :DITTO
2543 010222 000200 1$: .WORD 200 :EXTERNAL = E
2544 010224 000000 2$: .WORD 0 :INTERNAL = I
2545 010226 100000 3$: .WORD 100000 :STAGGERED = S
2546

```

```

2547                                     ;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
2548                                     ;BUFFER TO THE CHARACTERS 'E' AND 'C'.
2549                                     ;IF THE CHARACTER IS 'E' CLEAR THE FLAG
2550                                     ;IF THE CHARACTER IS 'C' SET THE FLAG
2551
2552 010230 017605 000000 .PAWCH:MOV @ (SP),R5
2553 010234 142737 000040 010304 BICB #40,INBUF ;SET FOR LOWER CASE INPUT
2554 010242 122737 000105 010304 CMPB #'E',INBUF ;IS IT 'E' ?
2555 010250 001002 BNE 1$
2556 010252 105015 CLRB (R5) ;000
2557 010254 000406 BR 2$
2558 010256 122737 000103 010304 1$: CMPB #'C',INBUF ;IS IT 'C' ?
2559 010264 001005 BNE 3$
2560 010266 112715 MOVB #-1,(R5) ;3177
2561 010272 062716 000002 2$: ADD #2,(SP)
2562 010276 000002 RTI
2563 010300 104404 3$: INSTER ;RETRY
2564 010302 000752 BR .PAWCH
2565
2566                                     ;BUFFERS FOR INPUT-OUTPUT
2567
2568 010304 000000 INBUF: 0
2569 010346 010346 .=. +40
2570 010346 000000 TEMP: 0
2571 010410 010410 .=. +40
2572 010410 000000 MDATA: 0
2573 010452 010452 .=. +40
2574

```



```

2575
2576
2577
2578
2579
2580
2581
2582
2583
2584 010452 005737 001406          CYCLE: 1ST      DZVACTV      :ARE ANY DZV11'S TO BE TESTED?
2585 010456 001004                    BNE          1$      :BR IF OK.
2586 010460 104402 007527          TYPE        ,MERR2  :NO DZV11'S SELECTED!!
2587 010464 000000                    HALT         :STOP THE SHOW.
2588 010466 000776                    BR          -2      :DISQUALIFY CONT. SW.
2589 010470 013737 004556 001354 1$:  MOV        $MIXCNT,$TIMES :RESTORE THE NUMBER OF ITERATIONS TO MAKE
2590 010476 033737 001412 001406  BIT        RUN,DZVACTV :IS THIS ONE 'ACTIVE'
2591 010504 001017                    BNE          2$      :BR IF GOOD ONE FOUND.
2592 010506 006137 001412          ROL        RUN      :UPDATE POINTER
2593 010512 005537 001412          ADC        RUN      :CATCH CARRY FROM RUN
2594 010516 062737 000012 001420  ADD        #12,ACTIVE :UPDATE ADDRESS POINTER.
2595 010524 022737 001740 001420  CMP        #DZV.END,ACTIVE :HAVE WE PASSED THE END OF THE MAP?
2596 010532 001356                    BNE          1$      :IF NO, KEEP GOING; NOT ALL TESTED FOR.
2597 010534 012737 001500 001420  MOV        #DZV.MAP,ACTIVE :RESET ADDRESS POINTER.
2598 010542 000752                    BR          1$      :KEEP LOOKING FOR ACTIVE DZV11
2599 010544 006137 001412          2$:  ROL        RUN      :UPDATE POINTER.
2600 010550 005537 001412          ADC        RUN      :CATCH CARRY.
2601 010554 013700 001420          MOV        ACTIVE,R0  :GET ADDRESS POINTER.
2602 010560 062737 000012 001420  ADD        #12,ACTIVE :UPDATE.
2603 010566 022737 001740 001420  CMP        #DZV.END,ACTIVE
2604
2605 010574 001003                    BNE          3$      :ALL DONE?
2606 010576 012737 001500 001420  MOV        #DZV.MAP,ACTIVE :BR IF NO.
2607 010604 012037 001174          3$:  MOV        (R0)+,$BASE  :RESTORE POINTER.
2608 010610 012037 002040          MOV        (R0)+,DZVRIV :LOAD SYSTEM CTRL. REG
2609 010614 012037 001366          MOV        (R0)+,LINE  :LOAD VECTOR
2610 010620 012037 001370          MOV        (R0)+,PAR   :SET UP DZV LINES ACTIVE
2611 010624 012037 001372          MOV        (R0)+,MODE  :SET UP PARAMETERIZATION
2612 010630 105037 001424          CLRB      MNTFLG ;RESET MAINT. FLAG IF
2613 010634 005737 001372          TST      MODE      :RUNNING TESTS
2614 010640 001003                    BNE          9$      :IN
2615 010642 112737 000010 001424  MOVB     #MAINT,MNTFLG :INTERNAL MAINT. MODE
2616 010650 004737 011016          9$:  JSR      PC,DZVLEV  :SET UP
2617 010654 005737 000042          TST      @#42      :ARE WE UNDER MONITOR CONTROL?
2618 010660 001051                    BNE          7$      :IF YES, SKIP THIS SETUP
2619 010662 032777 000002 170414  BIT      #SW01,@SWR  :IF SW01=1, GET STARTING TEST #
2620 010670 001445                    BEQ          7$      :BR IF NO TEST IS TO BE INPUTTED
2621 010672 104402 001357          4$:  TYPE        ,SCLRF
2622 010676 104403                    INSTR
2623 010700 007707                    MTSTN
2624 010702 104405                    PARAM
2625 010704 000001                    1
2626 010706 001000                    1000
2627 010710 001246                    $TSTNM
2628 010712 000                    .BYTE 0
2629 010713 001                    .BYTE 1
2630 010714 012700 012204          MOV        #TST1,R0

```

```

2631 010720 022710 000004      5$:    CMP      #4,(R0)
2632 010724 001020                BNE      6$
2633 010726 022760 012737 000002    CMP      #12737,2(R0)
2634 010734 001014                BNE      6$
2635 010736 023760 001246 000004    CMP      $STNM,4(R0)      ;IS THIS THE TEST ?
2636 010744 001010                BNE      6$              ;IF NOT, DON'T PROCESS NUMBER
2637 010746 010037 001252                MOV      R0,$LPADR        ;SAVE PC
2638 010752 062737 000002 001252    ADD      #2,$LPADR        ;POP OVER PREVIOUS SCOPE
2639 010760 104402 001357                TYPE     $SRLF
2640 010764 000412                BR       8$
2641 010766 005720                6$:    TST      (R0)+
2642 010770 020027 015662                CMP      R0,#TLAST+10
2643 010774 001351                BNE      5$
2644 010776 104402 001356                TYPE     $QUES
2645 011002 000733                BR       4$
2646 011004 012737 012204 001252    7$:    MOV      #TST1,$LPADR ;PREPARE TEST ADDRESS
2647 011012                8$:
2648 011012 000177 170234    RESTART:JMP  @$LPADR      ;GO START TESTING.***WARNING!***
2649                                     ;THIS JUMP IS USED BY POWER UP ROUTINE!!!
2650
2651                                     ;THIS UTILITY SETS UP CSR'S,SETS UP VECTORS.
2652 011016 013700 002040    DZVLEV:MOV  DZVRIV,R0      ;PLACE THE BASE VECTOR ADDRESS IN R0
2653 011022 062700 000002                ADD      #2,R0           ;CALCULATE THE RECEIVER INTERRUPT STATUS ADDR.
2654 011026 010037 002042                MOV      R0,DZVRIS      ;STORE IT HERE
2655 011032 062700 000002                ADD      #2,R0           ;CALCULATE THE TRANSMITTER INTERRUPT VECTOR
2656 011036 010037 002044                MOV      R0,DZVTIV      ;STORE IT HERE
2657 011042 062700 000002                ADD      #2,R0           ;CALCULATE THE TRANSMITTER VECTOR STATUS ADDRESS
2658 011046 010037 002046                MOV      R0,DZVTIS      ;STORE IT HERE
2659
2660                                     ;THIS SEGMENT SETS UP POINTERS FOR THE GIVEN DZV11. $BASE IS THE BASE ADDRESS
2661                                     ;OF THE DEVICE
2662 011052 013700 001174                MOV      $BASE,R0       ;COPY THE ADDRESS BEING LOADED
2663 011056 010037 002010                MOV      R0,DZVCSR      ;XXX0
2664 011062 005200                INC      R0
2665 011064 010037 002012                MOV      R0,HDZVCSR     ;XXX1
2666 011070 005200                INC      R0
2667 011072 010037 002014                MOV      R0,DZVRBUF     ;XXX2
2668 011076 010037 002020                MOV      R0,DZVLPR      ;XXX2
2669 011102 005200                INC      R0
2670 011104 010037 002016                MOV      R0,HDZVRBUF    ;XXX3
2671 011110 010037 002022                MOV      R0,HDZVLPR     ;XXX3
2672 011114 005200                INC      R0
2673 011116 010037 002024                MOV      R0,DZVTCR      ;XXX4
2674 011122 005200                INC      R0
2675 011124 010037 002026                MOV      R0,HDZVTCR    ;XXX5
2676 011130 005200                INC      R0
2677 011132 010037 002030                MOV      R0,DZVMSR      ;XXX6
2678 011136 010037 002034                MOV      R0,DZVTDR      ;XXX6
2679 011142 005200                INC      R0
2680 011144 010037 002032                MOV      R0,HDZVMSR     ;XXX7
2681 011150 010037 002036                MOV      R0,HDZVTDR     ;XXX7
2682 011154 000207                RTS      PC

```

2683					
2684	011156	011605			
2685	011160	012537	011342		
2686	011164	012537	011344		
2687	011170	012537	011346		
2688	011174	112537	011350		
2689	011200	112537	011351		
2690	011204	010516			
2691	011206	005005			
2692	011210	012704	010304		
2693	011214	122714	000015		
2694	011220	001424			
2695	011222	121427	000060		
2696	011226	002421			
2697	011230	121427	000071		
2698	011234	003016			
2699	011236	142714	000060		
2700	011242	005002			
2701	011244	152402			
2702	011246	060205			
2703	011250	122714	000015		
2704	011254	001410			
2705	011256	006305			
2706	011260	010502			
2707	011262	006305			
2708	011264	006305			
2709	011266	060205			
2710	011270	000754			
2711	011272	104404			
2712	011274	000744			
2713					
2714					
2715					
2716	011276	020537	011344		
2717	011302	101373			
2718	011304	020537	011342		
2719	011310	103770			
2720	011312	133705	011350		
2721	011316	001365			
2722					
2723					
2724					
2725	011320	013704	011346		
2726	011324	010524			
2727	011326	062705	000002		
2728	011332	105337	011351		
2729	011336	001372			
2730	011340	000002			
2731	011342	000000			
2732	011344	000000			
2733	011346	000000			
2734	011350	000			
2735	011351	000			

```

: CONVERT DECIMAL ASCII STRING TO OCTAL
.PARMD: MOV (SP),R5
MOV (R5)+,6$
MOV (R5)+,7$
MOV (R5)+,8$
MOV (R5)+,9$
MOV (R5)+,10$
MOV R5,(SP)
2$: CLR R5
MOV #INBUF,R4
CMPB #15,(R4)
BEQ 3$
1$: CMPB (R4),#0
BLT 3$
CMPB (R4),#9
BGT 3$
BICB #0,(R4)
CLR R2
BISB (R4)+,R2
ADD R2,R5
CMPB #15,(R4)
BEQ 4$
ASL R5 ;X2
MOV R5,R2 ;SAVE X2
ASL R5 ;X4
ASL R5 ;X8
ADD R2,R5 ;TIMES 10
BR 1$
3$: INSTER
BR 2$

: TEST TO SEE IF NUMBER IS WITHIN LIMITS
4$: CMP R5,7$
BHI 3$
CMP R5,6$
BLO 3$
BITB 9$,R5
BNE 3$

: STORE NUMBER AT SPECIFIED ADDRESS
5$: MOV 8$,R4
MOV R5,(R4)+
ADD #2,R5
DECB 10$
BNE 5$
RTI
6$: 0
7$: 0
8$: 0
9$: .BYTE 0
10$: .BYTE 0

```

```

2736                                     : *ROUTINE USED TO SET UP THE DIAGNOSTIC VIA APT.
2737                                     : *IF BIT7 IN THE ENVIRONMENT MODE (SENVN) BYTE IS SET,
2738                                     : *THE PROGRAM WILL LOAD ITS PARAMETERS FROM THE ETABLÉ.
2739
2740 011352 012700 001500      SETAPT: MOV      #DZV.MAP,R0      : POINT TO THE DEVICE MAP TABLE
2741 011356 013701 001174      MOV      $BASE,R1      : BUILD DEVICE ADDRESSES IN R1
2742 011362 013702 001170      MOV      $VECT1,R2     : BUILD DEVICE VECTORS IN R2
2743 011366 042702 177007      BIC      #^C<770>,R2   : STRIP AWAY OTHER INFORMATION
2744
2745 011372 012704 001204      MOV      #SDDW0,R4     : POINT TO THE BEGINNING OF DEVICE PARAMETERS
2746 011376 013705 001176      MOV      $DEVN,R5     : GET THE MAP OF ACTIVE DEVICES
2747 011402 105037 001414      CLRB    DZVNUM        : INITIALIZE NO. OF DEVICES IN SYSTEM
2748 011406 005037 001410      CLR     SAVACTV       : CLEAR THE ACTIVE BIT MAP
2749 011412 006005 1$:        ROR      R5           : GET A DEVICE SELECTION BIT
2750 011414 103407           BCS     3$            : IF IT IS SELECTED, GO SET UP A MAP
2751 011416 001422           BEQ     5$            : IF NO MORE ARE SELECTED, GET OUT OF SETUP
2752 011420 005724           TST     (R4)+         : POINT TO NEXT DEVICE DESCRIPTOR
2753 011422 062701 000010      2$:      ADD     #10,R1     : SET UP THE NEXT ADDRESS
2754 011426 062702 000010      ADD     #10,R2     : SET UP THE NEXT VECTOR GROUP
2755 011432 000767           BR      1$           : GO SEE IF MORE DEVICES REMAIN
2756 011434 006137 001410      3$:      ROL     SAVACTV     : SET BIT IN ACTIVE DEVICE MAP
2757 011440 105237 001414      INCB   DZVNUM       : INCREMENT NO. OF ACTIVE DEVICES IN SYSTEM
2758 011444 010120           MOV     R1,(R0)+    : LOAD DEVICE ADDRESS
2759 011446 010220           MOV     R2,(R0)+    : LOAD THE VECTOR ADDRESS
2760 011450 013720 001200      MOV     $CDW1,(R0)+  : GET THE NUMBER OF LINES IN OPERATION
2761 011454 012420           MOV     (R4)+,(R0)+ : LOAD DEVICE PARAMETERS
2762 011456 013720 001202      MOV     $CDW2,(R0)+  : LOAD DEFAULT TESTING MODE
2763 011462 000757           BR      2$           : GO BUILD THE NEXT ADDRESS
2764 011464 012710 177777      5$:      MOV     #-1,(R0)    : TERMINATE THE DEVICE MAP
2765 011470 012737 001142 001304  MOV     #$$SWREG,SWR : SET TO SOFTWARE APT SWITCH REGISTER
2766 011476 000207           RTS     PC           : RETURN TO PRINT STATUS TABLE
2767
2768
2769                                     : *ROUTINE USED TO 'AUTO SIZE' THE DZV11
2770                                     : *CSR AND VECTOR.
2771                                     : *NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
2772                                     : *      ADDRESS RANGE (160000:163770)
2773                                     : *      AND THE VECTOR MAY BE ANY WHERE IN THE
2774                                     : *      FLOATING VECTOR RANGE (300:770)
2775                                     : *
2776
2777 011500      AUTO.SIZE:
2778 011500 000005      RESET
2779 011502 105337 001422      DECB   INIFLG        : INSURE A BUS INIT.
2780 011506 012702 001500      CSRMAP: MOV     #DZV.MAP,R2 : SHOW THAT I WAS HERE
2781 011512 012703 001204      MOV     #SDDW0,R3     : LOAD MAP POINTER.
2782 011516 005022      1$:      CLR     (R2)+         : POINT TO ETABLE DEVICE DESCRIPTOR WORDS
2783 011520 022702 001740      CMP     #DZV.END,R2   : ZERO ENTIRE MAP
2784 011524 001374           BNE     1$           : ALL DONE?
2785 011526 105037 001414      CLRB   DZVNUM        : BR IF NO
2786 011532 012702 001500      MOV     #DZV.MAP,R2   : SET OCTAL NUMBER OF DZV11'S TO 0
2787 011536 012701 160000      MOV     #160000,R1    : SET FOR FIRST ADDRESS TO BE TESTED
2788 011542 012737 012006 000004  MOV     #6$,$#4       : SET FOR NON-EXISTENT DEVICE TIME OUT
2789 011550 052711 000040           BIS     #BIT5,(R1)    : TRY TO SET MASTER SCAN ENABLE
2790 011554 052761 000017 000004  BIS     #17,4(R1)    : TRY TO TRANSMIT ON ANY LINE
2791 011562 005000           CLR     R0           : USE R0 AS A COUNTER

```

2792	011564	005711			7\$:	TST	(R1)	:HAS TRANSMITTER READY COME UP?
2793	011566	100403				BMI	8\$:IF SO, GO GET A FINAL CHECK
2794	011570	005300				DEC	R0	:REDUCE COUNT. TIME UP?
2795	011572	001374				BNE	7\$:IF NOT, KEEP WAITING
2796	011574	000437				BR	3\$:ASSUME IT'S NOT A DZV11
2797	011576	032761	000017	000004	8\$:	BIT	#17,4(R1)	:ARE ANY TCR BITS STILL SET? THEY SHOULD BE
2798	011604	001433				BEQ	3\$:IF IT'S NOT, ASSUME IT'S NOT A DZV11
2799	011606	032711	000040			BIT	#BITS,(R1)	:IS MASTER SCAN ENABLE STILL SET?
2800	011612	001430				BEQ	3\$:IF NOT, ASSUME IT'S NOT A DZV11
2801	011614	052711	000020			BIS	#20,(R1)	:SET DEVICE CLEAR
2802	011620	000240				NOP		
2803	011622	032711	000040			BIT	#40,(R1)	:DID SCANNER CLEAR
2804	011626	001022				BNE	3\$:IF NOT ASSUME IT IS NOT DZV
2805	011630	005061	000004			CLR	4(R1)	:GET RID OF TCR BITS
2806								:AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A DZV11 CSR ADDRESS.
2807	011634	010122				MOV	R1,(R2)+	:STORE CSR IN CORE TABLE.
2808	011636	005722				TST	(R2)+	:POP OVER VECTOR STORE AREA
2809	011640	012722	000017			MOV	#17,(R2)+	:SET THE DEFAULT LINE SELECTION PARAMETER
2810	011644	012712	017470			MOV	#17470,(R2)	:SET THE DEFAULT PARAMETERS
2811	011650	012223				MOV	(R2)+,(R3)+	:COPY PARAMETERS INTO ETABLE DESCRIPTOR
2812	011652	005022				CLR	(R2)+	:SET THE DEFAULT MODE OF OPERATION
2813	011654	012712	177777			MOV	#-1,(R2)	:TERMINATE LIST
2814	011660	105237	001414			INCB	DZVNUM	:UPDATE DEVICE COUNTER
2815	011664	122737	000020	001414		CMPB	#20,DZVNUM	:ARE MAX. NO. OF DEV FOUND?
2816	011672	001405				BEQ	100\$:YES DON'T LOOK FOR ANY MORE.
2817	011674	062701	000010		3\$:	ADD	#10,R1	:UPDATE CSR POINTER ADDRESS
2818	011700	022701	164000			CMP	#164000,R1	
2819	011704	001321				BNE	2\$:BR IF MORE ADDRESS TO CHECK.
2820	011706				100\$:			
2821	011706	105737	001414			TSTB	DZVNUM	:WERE ANY DZV11'S FOUND AT ALL?
2822	011712	001430				BEQ	5\$:ERROR AUTO SIZER FOUND NO DZV11'S IN THIS SYS.
2823	011714	113701	001414			MOVB	DZVNUM,R1	
2824	011720	012737	000001	001410		MOV	#1,SAVACTV	:CREATE A BIT MAP OF THE ACTIVE
2825	011726	005301			4\$:	DEC	R1	:DEVICES IN THE SYSTEM
2826	011730	001404				BEQ	98\$	
2827	011732	000261				SEC		
2828	011734	006137	001410			ROL	SAVACTV	
2829	011740	000772				BR	4\$	
2830	011742	013737	001500	001174	98\$:	MOV	DZCRO,\$BASE	:POINT TO THE ADDRESS OF FIRST DEVICE
2831	011750	013737	001510	001202		MOV	MANTO,\$CDW2	:INDICATE TO ETABLE WHAT MODE IS BEING USED
2832	011756	012737	000006	000004	99\$:	MOV	#6,@4	:RESTORE TRAP VECTOR
2833	011764	013737	001410	001176		MOV	SAVACTV,\$DEVN	:SAVE ACTIVE REGISTER
2834	011772	000410				BR	VECMAP	:GO FIND THE VECTOR NOW.
2835	011774	104402	007527		5\$:	TYPE	,MERR2	:NOTIFY OPR THAT NO DZV11'S FOUND.
2836	012000	005000				CLR	R0	:MAKE DATA DISPLAY ZERO
2837	012002	000000				HALT		:STOP THE SHOW
2838	012004	000776				BR	.-2	:DISABLE CONT. SW.
2839	012006	012716	011674		6\$:	MOV	#3\$,(SP)	:ENTERED BY NON-EXISTENT TIME-OUT
2840	012012	000002				RTI		:RETURN TO MAINSTREAM
2841								
2842	012014	012737	000200	000022	VECMAP:	MOV	#MASK,@#22	:SET IOT TRAP PRIORITY
2843	012022	012737	012136	000020		MOV	#4\$,@#20	:SET IOT TRAP VECTOR
2844	012030	012702	001500			MOV	#DZV.MAP,R2	:SET SOFTWARE POINTER
2845	012034	012700	000300			MOV	#300,R0	:FLOATING VECTORS START HERE.
2846	012040	012701	000302			MOV	#302,R1	:PC OF IOT INSTR.
2847	012044	010120			1\$:	MOV	R1,(R0)+	:START FILLING VECTOR AREA

2848	012046	012721	000004		MOV	#4,(R1)+	:WITH .+2: IOT
2849	012052	022021			CMP	(R0)+,(R1)+	:ADD 2 TO R0 +R1
2850	012054	020127	001000		CMP	R1,#1000	:HAS THE VECTOR AREA BEEN EXCEEDED?
2851	012060	101771			BLOS	1\$:BR IF MORE TO FILL
2852	012062	013704	001410		MOV	SAVACTV,R4	:STORE TEMPORARILY
2853	012066	006004		2\$:	ROR	R4	:BRING OUT A BIT
2854	012070	103036			BCC	5\$:BR IF ALL DONE
2855	012072	106427	000000		MTPS	#0	:ZERO CPU PRIO
2856	012076	012772	040040	000000	MOV	#BIT14+BIT5,@(R2)	:SET TIE AND MAS SCAN
2857	012104	011201			MOV	(R2),R1	:GET CSR
2858	012106	112761	000017	000004	MOVB	#17,4(R1)	:SET THE TCR BITS FOR ALL LINES
2859							:ATTEMPT TO FORCE AN INTERRUPT
2860	012114	005200			INC	R0	:STALL
2861	012116	001376			BNE	.-?	: FOR TIME TO INTERRUPT
2862	012120	012762	000300	000002	MOV	#300,2(R2)	:NO INTERRUPT ASSUME 300 AND FIX DZV1 LATER
2863	012126	000005			RESET		:INIT
2864	012130	062702	000012		ADD	#12,R2	:POP SOFTWARE POINTER
2865	012134	000754		3\$:	BR	2\$:KEEP GOING
2866	012136	011662	000002		MOV	(SP),2(R2)	:GET VECTOR ADDRESS
2867	012142	162762	000010	000002	SUB	#10,2(R2)	:POINT BACK TO THE CORRECT VECTOR
2868	012150	042762	000007	000002	BIC	#7,2(R2)	:CLEAR JUNK
2869	012156	022626			POP2SP		:POP IOT JUNK OFF STACK
2870	012160	012716	012130		MOV	#3\$,(SP)	:SET FOR RETURN
2871	012164	000002			RTI		
2872	012166	013737	001502	001170	MOV	DZVCO,\$VECT1	:COPY VECTOR OF FIRST DEVICE INTO ETABLE
2873	012174	012737	004314	000020	MOV	#.SCOPE,IOTVEC	:RESTORE THE SCOPE TRAP
2874	012202	000207			RTS	PC	:ALL DONE WITH 'AUTO SIZING'
2875							

```

2876
2877
2878
2879
2880
2881
2882
2883 012204 000004
2884 012206 012737 000001 001246
2885 012214 012737 012374 001362
2886 012222 012737 012362 000004
2887 012230 012737 000200 000006
2888 012236 012737 012244 001364
2889 012244 013700 002010
2890 012250 011001
2891 012252 000240
2892 012254 005010
2893 012256 000240
2894 012260 012737 012266 001364
2895 012266 013700 002014
2896 012272 011001
2897 012274 000240
2898 012276 005010
2899 012300 000240
2900 012302 012737 012310 001364
2901 012310 013700 002024
2902 012314 011001
2903 012316 000240
2904 012320 005010
2905 012322 000240
2906 012324 012737 012332 001364
2907 012332 013700 002030
2908 012336 011001
2909 012340 000240
2910 012342 005010
2911 012344 000240
2912 012346 012737 000006 000004
2913 012354 005037 000006
2914 012360 104400
2915 012362 011601
2916 012364 022626
2917 012366 104001
2918 012370 104401
2919 012372 000111
2920
2921
2922
2923
2924
2925
2926 012374 000004
2927 012376 012737 000002 001246
2928 012404 012737 012440 001362
2929 012412 013700 002010
2930 012416 012710 000020
2931 012422 005005

```

```

***** TEST 1 *****
*THIS TEST PROVES THE BUS REPLY RESPONSE
*DURING A READ OR WRITE TO THE FOLLOWING ADDRESS:
* DZVCSR, DZVRBUF, DZVTCR, DZVMSR
::* TEST 1
*****
TST1: SCOPE
MOV #1,STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST2,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV #5$,4 ;SET TRAP VECTOR
MOV #MASK,6 ;SET PRIORITY TO HIGH(MASK INTERRUPTS)
MOV #1$,LOCK ;SET RETURN IF SW09=11
1$: MOV DZVCSR,RO ;SET ADDRESS TO TEST
MOV (RO),R1 ;READ THE ADDRESS
NOP ;WASTE TIME
CLR (RO) ;WRITE THE ADDRESS
NOP ;WASTE TIME
MOV #2$,LOCK ;SET RETURN ADDRESS FOR SW09
2$: MOV DZVRBUF,RO ;SET ADDRESS TO TEST
MOV (RO),R1 ;READ THE ADDRESS
NOP ;
CLR (RO) ;WRITE THE ADDRESS
NOP ;WASTE TIME
MOV #3$,LOCK ;SET RETURN ADDRESS FOR SW09
3$: MOV DZVTCR,RO ;SET ADDRESS TO TEST
MOV (RO),R1 ;READ THE ADDRESS
NOP ;
CLR (RO) ;WRITE THE ADDRESS
NOP ;
MOV #4$,LOCK ;SET RETURN ADDRESS
4$: MOV DZVMSR,RO ;SET ADDRESS TO TEST
MOV (RO),R1 ;READ FROM ADDRESS
NOP ;
CLR (RO) ;WRITE THE ADDRESS
NOP ;
MOV #6,4 ;SET TRAP CATCHER BACK TO NORMAL
5$: ADVANCE 6 ;SCOPE THIS TEST
MOV (SP),R1 ;SAVE PC OF TRAP
POP2SP ;POP TRAP OFF STACK
ERROR 1 ;*NO BUS REPLY RESPONSE.
SCOP1 ;SW09=1?
JMP (R1) ;RTI
***** TEST 2 *****
*THIS TEST PROVES THAT BIT 'DCLR'
*CAN BE SET AND THAT IT WILL CLEAR
*BY ITSELF
::* TEST 2
*****
TST2: SCOPE
MOV #2,STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST3,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV DZVCSR,RO ;SET POINTER
MOV #DCLR,(RO) ;SET DCLR
CLR R5 ;SET EXPECTED TO 0

```

```

2932 012424 005003
2933 012426 011004
2934 012430 001403
2935 012432 105203
2936 012434 001374
2937 012436 104002
2938 012440
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949 012440 000004
2950 012442 012737 000003 001246
2951 012450 012737 012616 001362
2952 012456 013700 002010
2953 012462 012703 012576
2954 012466 011305
2955 012470 012737 012476 001364
2956 012476 010510
2957 012500 011004
2958 012502 020504
2959 012504 001401
2960 012506 104002
2961 012510 104401
2962 012512 012737 012520 001364
2963 012520 040510
2964 012522 011004
2965 012524 001403
2966 012526 005005
2967 012530 104002
2968 012532 011305
2969 012534 104401
2970 012536 012737 012544 001364
2971 012544 010510
2972 012546 104413
2973 012550 011004
2974 012552 001403
2975 012554 005005
2976 012556 104002
2977 012560 011305
2978 012562 104401
2979 012564 062703 000002
2980 012570 005713
2981 012572 001407
2982 012574 000734
2983 012576 000010
2984 012600 000040
2985 012602 010000
2986 012604 000100
2987 012606 040000

2$: CLR R3 ;DUAL LOOP COUNTER
MOV (R0),R4 ;IS DCLR CLEAR?
BEQ 3$ ;IF YES, GO TO THE NEXT TEST
INCB R3 ;IF NO, COUNT 1 OF 256 TICKS
BNE 2$ ;HAS THE TIME EXPIRED? IF NO, GO TEST BIT AGAIN
ERROR 2 ;*DCLR FAILED TO CLEAR

3$: ;***** TEST 3 *****
; *TEST TO VERIFY THAT THE R/W BITS OF THE
; *DZVCSR REGISTER CAN BE SET. THEN VERIFY THAT
; *THESE BITS CAN BE CLEARED. AND FINALLY, VERIFY
; *THAT AFTER BEING SET AGAIN THEY CAN BE
; *CLEARED BY A 'DEVICE CLEAR'.
; *THE BITS TESTED ARE: MAINT, MSENAB, SILOEN,
; *RIE, AND TIE.

;:* TEST 3
;*****
TST3: SCOPE
MOV #3,STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST4,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV DZVCSR,R0 ;GET BASE ADDRESS
MOV #5$,R3 ;SET R3 TO TOP OF TABLE
1$: MOV (R3),R5 ;SET BIT
MOV #11$,LOCK ;SETUP FOR TIGHT SCOPE LOOP
11$: MOV R5,(R0) ;SET BIT IN DEVICE
MOV (R0),R4 ;READ THE BIT FROM DEVICE
CMP R5,R4 ;WAS BIT SET?
BEQ 2$ ;BR IF YES
ERROR 2 ;*BIT R/W FAILURE
2$: SCOP1 ;IS SWITCH 9 SET?
MOV #12$,LOCK ;SET FOR NEXT TIGHT SCOPE LOOP
12$: BIC R5,(R0) ;CLEAR THE BIT.
MOV (R0),R4 ;READ DEVICE
BEQ 3$ ;BR IF BITS WERE CLEARED.
CLR R5 ;CLEAR FOR ERROR PRINTOUT
ERROR 2 ;*BIT FAILED TO CLEAR
MOV (R3),R5 ;RESTORE THE BIT.
3$: SCOP1 ;SW09 SET?
MOV #13$,LOCK ;SET UP FOR NEXT TIGHT SCOPE
13$: MOV R5,(R0) ;SET THE BIT AGAIN
DEVICE.CLR ;ISSUE DEVICE CLEAR
MOV (R0),R4 ;READ THE BIT.
BEQ 4$ ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
CLR R5 ;SET EXPECTED TO ZERO
ERROR 2 ;*BIT NOT CLEARED BY DEVICE CLEAR
MOV (R3),R5 ;RESTORE BIT AGAIN
4$: SCOP1 ;SW09 SET?
ADD #2,R3 ;POP R3
TST (R3) ;IS THIS THE END OF TABLE?
BEQ 6$ ;IF YES GET OUT
BR 1$ ;OTHERWISE TEST NEXT BIT
5$: #MAINT ;CSR BIT: INTERNAL MAINTENANCE
#MSENAB ;CSR BIT: MASTER SCAN ENABLE
#SILOEN ;CSR BIT: SILO ENABLE
#RIE ;CSR BIT: RECEIVER INTER. ENABLE
#TIE ;CSR BIT: TRANS. INTER. ENABLE

```



```

2988 012610 000000
2989 012612 005037 001364
2990
2991
2992
2993
2994
2995
2996
2997 012616 000004
2998 012620 012737 000004 001246
2999 012626 012737 013022 001362
3000 012634 013700 002024
3001 012640 012703 012726
3002 012644 012737 012654 001364
3003 012652 011305
3004 012654 010510
3005 012656 011004
3006 012660 020504
3007 012662 001401
3008 012664 104002
3009 012666 104401
3010 012670 012737 012676 001364
3011 012676 040510
3012 012700 011004
3013 012702 001403
3014 012704 005005
3015 012706 104002
3016 012710 011305
3017 012712 104401
3018 012714 062703 000002
3019 012720 005713
3020 012722 001412
3021 012724 000747
3022 012726 000001
3023 012730 000002
3024 012732 000004
3025 012734 000010
3026 012736 000400
3027 012740 001000
3028 012742 002000
3029 012744 004000
3030 012746 000000
3031 012750 005037 001364
3032 012754 012710 177777
3033 012760 012705 007400
3034 012764 104413
3035 012766 011004
3036 012770 020504
3037 012772 001401
3038 012774 104002
3039 012776 005005
3040 013000 005227 000000
3041 013004 001375
3042 013006 012710 177777
3043 013012 000005

#0 :END OF TABLE
6$: CLR LOCK :ZERO LOCK INDICATOR
:***** TEST 4 *****
:*THIS TESTS THAT ALL OF THE TCR BITS
:*CAN BE: SET, CLEARED, AND CLEARED BY A DEVICE CLEAR.
:*THIS TEST ALSO DETERMINES IF THE DTR BITS CAN
:*BE SET, CLEARED, AND CLEARED BY A RESET.
::* TEST 4
:*****
TST4: SCOPE
MOV #4,STSTNM :LOAD THE NUMBER OF THIS TEST
MOV #TST5,NEXT :POINT TO THE START OF THE NEXT TEST
MOV DZVTCR,R0 :SET DEVICE ADDRESS
MOV #5,R3 :SET R3 POINTER TO TOP OF TABLE
1$: MOV #1$,LOCK :SET LOCK FOR SW09 SCOPE LOOP
MOV (R3),R5 :SET EXPECTED RESULTS
11$: MOV R5,(R0) :SET THE BIT
MOV (R0),R4 :READ THE BIT FROM THE DEVICE
CMP R5,R4 :DID THE BIT SET?
BEQ 2$ :BR IF YES
ERROR 2 :*BIT FAILED TO SET.
2$: SCOP1 :SW09 SET?
MOV #3$,LOCK :SET UP FOR NEXT TIGHT SCOPE LOOP
3$: BIC R5,(R0) :CLEAR THE BIT
MOV (R0),R4 :READ THE REGISTER
BEQ 4$ :BR IF YES
CLR R5 :SET EXPECTED TO 0
ERROR 2 :*REPORT BIT NOT CLEAR
MOV (R3),R5 :RESTORE R5
4$: SCOP1 :SW09 SET?
ADD #2,R3 :POP POINTER TO NEXT TABLE ENTRY
TST (R3) :END OF TABLE?
BEQ 6$ :IF YES JUMP OVER TABLE
BR 1$ :START TESTING NEXT BIT
5$: #TCR0 :TCR BIT FOR LINE 0
#TCR1 :TCR BIT FOR LINE 1
#TCR2 :TCR BIT FOR LINE 2
#TCR3 :TCR BIT FOR LINE 3
#DTR0 :DTR BIT FOR LINE 0
#DTR1 :DTR BIT FOR LINE 1
#DTR2 :DTR BIT FOR LINE 2
#DTR3 :DTR BIT FOR LINE 3
#0 :END OF TABLE
6$: CLR LOCK :CLEAR TIGHT SCOPE LOOP INDIC.
MOV #-1,(R0) :SET ALL BITS IN TCR REGISTER
MOV #007400,R5 :SET EXPECTED
DEVICE.CLR :SET DCLR BIT IN CSR
MOV (R0),R4 :READ REGISTER
CMP R5,R4 :TCR BITS CLEARED?
BEQ 7$ :IF YES BRANCH
ERROR 2 :TCR BITS NOT CLEARED!
7$: CLR R5 :SET EXPECTED TO ZERO
8$: INC #0 :DELAY FOR ACT
BNE 8$
MOV #-1,(R0) :SET ALL POSSIBLE BITS
RESET :DO BUS INIT

```

```

3044 013014 011004      MOV      (R0),R4      ;DID REGISTER CLEAR?
3045 013016 001401      BEQ      9$           ;IF YES GET OUT
3046 013020 104002      ERROR   2            ;REGISTER DID NOT CLEAR!
3047 013022
3048
3049
3050
3051
3052
3053
3054
3055
3056 013022 000004      TST5:  SCOPE
3057 013024 012737 000005 001246      MOV      #5,STSTNM   ;LOAD THE NUMBER OF THIS TEST
3058 013032 012737 013124 001362      MOV      #TST6,NEXT ;POINT TO THE START OF THE NEXT TEST
3059 013040 013700 002010      MOV      DZVCSR,R0  ;SET ADDRESS TO R0
3060 013044 104413      DEVICE.CLR          ;DO A DEVICE CLEAR
3061 013046 005005      CLR      R5         ;SET EXPECTED TO 0
3062 013050 012710 121600      MOV      #RDONE+TRDY+BIT9+BIT8+SILOAL,(R0)
3063
3064 013054 011004      MOV      (R0),R4    ;WRITE THE BITS
3065 013056 001401      BEQ      2$         ;READ BACK THE BITS
3066 013060 104002      ERROR   2            ;BR IF NONE ARE SET.
3067 013062 012705 100040 166730 2$:      MOV      #TRDY+MSENAB,R5 ;*BITS WERE SET.
3068 013066 052777 000017      BIS      #17,@DZVTCR ;SET EXPECTED BIT
3069 013074 052710 000040      BIS      #MSENAB,(R0) ;SET TCR BITS FOR ALL LINES
3070 013100 005002      CLR      R2         ;SET SCAN ENABLE
3071 013102 011004      MOV      (R0),R4    ;SET COUNTER TO ZERO
3072 013104 042704 001400 3$:      BIC      #BIT9!BIT8,R4 ;READ THE REGISTER
3073 013110 020504      CMP      R5,R4     ;MASK OUT LINE NO.
3074 013112 001404      BEQ      4$         ;BIT SET?
3075 013114 104414      DELAY                    ;BR IF YES
3076 013116 005202      INC      R2         ;STALL TIME
3077 013120 001370      BNE     3$         ;UPDATE COUNTER
3078 013122 104002      ERROR   2            ;BR IF COUNTER NOT DONE.
3079 013124
3080
3081
3082
3083
3084
3085
3086
3087 013124 000004      TST6:  SCOPE
3088 013126 012737 000006 001246      MOV      #6,STSTNM   ;LOAD THE NUMBER OF THIS TEST
3089 013134 012737 013254 001362      MOV      #TST7,NEXT ;POINT TO THE START OF THE NEXT TEST
3090 013142 104413      DEVICE.CLR          ;SET DCLR IN CSR
3091 013144 013700 002010      MOV      DZVCSR,R0  ;SET UP FOR ERROR MESSAGE
3092 013150 012710 177757      MOV      #^C<DCLR>,(R0) ;TRY TO SET ALL BITS EXCEPT DCLR
3093 013154 012705 050150      MOV      #TIE!SILOEN!RIE!MSENAB!MAINT,R5 ;MAKE EXPECTED
3094 013160 011004      MOV      (R0),R4    ;ACTUAL
3095 013162 020405      CMP      R4,R5     ;CMP EXPECTED VS ACTUAL
3096 013164 001401      BEQ      1$         ;YES
3097 013166 104002      ERROR   2            ;*NO
3098 013170 105010 1$:      CLRB    (R0)        ;CLEAR LOW BYTE OF CSR
3099 013172 105005      CLRB    R5         ;CLEAR LOW BYTE OF EXPECTED DATA

```

```

3100 013174 011004      MOV      (R0),R4      ;READ CSR
3101 013176 020405      CMP      R4,R5       ;DOES CSR COMPARE WITH EXPECTED?
3102 013200 001401      BEQ      3$          ;BRANCH IF YES
3103 013202 104002      ERROR   2           ;IF NOT PRINT ERROR
3104 013204 012710 177757 3$:  MOV      #^C<DCLR>,(R0) ;SET ALL CSR BITS POSSIBLE
3105 013210 105077 166576  CLRB    @DZVCSR      ;CLEAR HIGH BYTE OF CSR
3106 013214 012705 000150  MOV      #RIE!MSENAB!MAINT,R5 ;SET EXPECTED IN R5
3107 013220 011004      MOV      (R0),R4      ;READ CSR REGISTER
3108 013222 020405      CMP      R4,R5       ;DOES ACTUAL=EXPECTED
3109 013224 001401      BEQ      4$          ;IF YES CONTINUE
3110 013226 104002      ERROR   2           ;IF NO PRINT ERROR
3111 013230 012710 177757 4$:  MOV      #^C<DCLR>,(R0) ;SET ALL POSSIBLE CSR BITS
3112 013234 005005      CLR      R5          ;SET R5 TO EXPECTED RESULTS
3113 013236 052710 000020  BIS     @DCLR,(R0)    ;DEVICE MASTER RESET
3114 013242 000240      NOP
3115 013244 011004      MOV      (R0),R4      ;ACTUAL
3116 013246 020405      CMP      R4,R5       ;CMP ACTUAL VS EXPECTED
3117 013250 001401      BEQ      2$          ;YES
3118 013252 104002      ERROR   2           ;NO
3119 013254
3120
3121
3122
3123
3124
3125
3126 013254 000004      ;***** TEST 7 *****
3127 013256 012737 000007 001246  ;*THIS TEST PERFORMS RESET TESTING AND
3128 013264 012737 013340 001362  ;*TESTING OF READ ONLY REGISTER DZVRBUF
3129 013272 104413      ;*AND TESTING OF WRITE ONLY REGISTER DZVLPR
3130 013274 013700 002014      ;:* TEST 7
3131 013300 011005      ;*****
3132 013302 042705 106000      TST7:  SCOPE
3133 013306 012777 177777 166504  MOV      #7,$TSTNM    ;LOAD THE NUMBER OF THIS TEST
3134 013314 011004      MOV      #TST10,NEXT ;POINT TO THE START OF THE NEXT TEST
3135 013316 020405      DEVICE.CLR          ;CLEAR DZV11
3136 013320 001401      MOV      DZVRBUF,R0  ;SET UP FOR ERROR MESSAGE
3137 013322 104002      MOV      (R0),R5     ;COPY PRESENT CONTENTS
3138 013324 005077 166470 1$:  BIC     #DVALID!BIT11!BIT10,R5 ;CLEAR ILLEGAL BITS
3139 013330 011004      MOV      #-1,@DZVLPR ;TRY TO WRITE ALL 1'S
3140 013332 020405      MOV      (R0),R4     ;ACTUAL
3141 013334 001401      CMP      R4,R5       ;CMP ACTUAL VS EXPECTED
3142 013336 104002      BEQ      1$          ;IF YES,GO CONTINUE PROCESSING
3143 013340      ERROR   2           ;*ERROR- BIT PATTERN NOT CORRECT
3144
3145
3146
3147
3148
3149
3150 013340 000004      ;***** TEST 10 *****
3151 013342 012737 000010 001246  ;*THIS TEST PERFORMS RESET TESTING AND
3152 013350 012737 013424 001362  ;*TESTING OF READ ONLY REGISTER DZVMSR
3153 013356 104413      ;*AND TESTING OF WRITE ONLY REGISTER DZVTDR
3154 013360 013700 002030      ;:* TEST 10
3155 013364 011005      ;*****
3156
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3220
3221
3222
3223
3224
3225
3226
3227
3228
3229
3230
3231
3232
3233
3234
3235
3236
3237
3238
3239
3240
3241
3242
3243
3244
3245
3246
3247
3248
3249
3250
3251
3252
3253
3254
3255
3256
3257
3258
3259
3260
3261
3262
3263
3264
3265
3266
3267
3268
3269
3270
3271
3272
3273
3274
3275
3276
3277
3278
3279
3280
3281
3282
3283
3284
3285
3286
3287
3288
3289
3290
3291
3292
3293
3294
3295
3296
3297
3298
3299
3300
3301
3302
3303
3304
3305
3306
3307
3308
3309
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3340
3341
3342
3343
3344
3345
3346
3347
3348
3349
3350
3351
3352
3353
3354
3355
3356
3357
3358
3359
3360
3361
3362
3363
3364
3365
3366
3367
3368
3369
3370
3371
3372
3373
3374
3375
3376
3377
3378
3379
3380
3381
3382
3383
3384
3385
3386
3387
3388
3389
3390
3391
3392
3393
3394
3395
3396
3397
3398
3399
3400
3401
3402
3403
3404
3405
3406
3407
3408
3409
3410
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
3501
3502
3503
3504
3505
3506
3507
3508
3509
3510
3511
3512
3513
3514
3515
3516
3517
3518
3519
3520
3521
3522
3523
3524
3525
3526
3527
3528
3529
3530
3531
3532
3533
3534
3535
3536
3537
3538
3539
3540
3541
3542
3543
3544
3545
3546
3547
3548
3549
3550
3551
3552
3553
3554
3555
3556
3557
3558
3559
3560
3561
3562
3563
3564
3565
3566
3567
3568
3569
3570
3571
3572
3573
3574
3575
3576
3577
3578
3579
3580
3581
3582
3583
3584
3585
3586
3587
3588
3589
3590
3591
3592
3593
3594
3595
3596
3597
3598
3599
3600
3601
3602
3603
3604
3605
3606
3607
3608
3609
3610
3611
3612
3613
3614
3615
3616
3617
3618
3619
3620
3621
3622
3623
3624
3625
3626
3627
3628
3629
3630
3631
3632
3633
3634
3635
3636
3637
3638
3639
3640
3641
3642
3643
3644
3645
3646
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673
3674
3675
3676
3677
3678
3679
3680
3681
3682
3683
3684
3685
3686
3687
3688
3689
3690
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
3739
3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759
3760
3761
3762
3763
3764
3765
3766
3767
3768
3769
3770
3771
3772
3773
3774
3775
3776
3777
3778
3779
3780
3781
3782
3783
3784
3785
3786
3787
3788
3789
3790
3791
3792
3793
3794
3795
3796
3797
3798
3799
3800
3801
3802
3803
3804
3805
3806
3807
3808
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
3825
3826
3827
3828
3829
3830
3831
3832
3833
3834
3835
3836
3837
3838
3839
3840
3841
3842
3843
3844
3845
3846
3847
3848
3849
3850
3851
3852
3853
3854
3855
3856
3857
3858
3859
3860
3861
3862
3863
3864
3865
3866
3867
3868
3869
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879
3880
3881
3882
3883
3884
3885
3886
3887
3888
3889
3890
3891
3892
3893
3894
3895
3896
3897
3898
3899
3900
3901
3902
3903
3904
3905
3906
3907
3908
3909
3910
3911
3912
3913
3914
3915
3916
3917
3918
3919
3920
3921
3922
3923
3924
3925
3926
3927
3928
3929
3930
3931
3932
3933
3934
3935
3936
3937
3938
3939
3940
3941
3942
3943
3944
3945
3946
3947
3948
3949
3950
3951
3952
3953
3954
3955
3956
3957
3958
3959
3960
3961
3962
3963
3964
3965
3966
3967
3968
3969
3970
3971
3972
3973
3974
3975
3976
3977
3978
3979
3980
3981
3982
3983
3984
3985
3986
3987
3988
3989
3990
3991
3992
3993
3994
3995
3996
3997
3998
3999
4000

```

```

3156 013366 042705 170360          BIC    #170360,R5      :CLEAR ILLEGAL BITS
3157 013372 112777 177777 166434    MOVB   #-1,@DZVDR     :TRY TO WRITE ALL 1'S
3158 013400 011004          MOV    (R0),R4        :ACTUAL
3159 013402 020405          CMP    R4,R5          :CMP ACTUAL VS EXPECTED
3160 013404 001401          BEQ    1$             :IF YES,GO CONTINUE PROCESSING
3161 013406 104002          ERROR  2             :*ERROR- BIT PATTERN NOT CORRECT
3162 013410 005077 166420 1$:    CLR   @DZVTDR        :TRY TO WRITE ALL ZEROES
3163 013414 011004          MOV    (R0),R4        :READ REGISTER
3164 013416 020405          CMP    R4,R5          :CMP ACTUAL VS. EXPECTED
3165 013420 001401          BEQ    2$             :BRANCH IF EQUAL
3166 013422 104002          ERROR  2             :VALUES DID NOT COMPARE
3167 013424
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181 013424 000004          :***** TEST 11 *****
3182 013426 012737 000011 001246    :*VERIFY THAT SETTING 'DTR' FOR A LINE WILL
3183 013434 012737 013620 001362    :*BRING UP 'CO' AND 'RING' FOR:
3184 013442 005737 001372          :*THE SAME LINE IF IN EXTERNAL MODE
3185 013446 001001          :*THE STAGGERED LINE IF IN STAGGERED MODE.
3186 013450 104400          :*LINES ARE STAGGERED AS FOLLOWS:
3187 013452 012737 013542 001364 8$:  :*LINE0 WITH LINE1; LINE2 WITH LINE3.
3188 013460 104413          :*THIS TEST IS ONLY RUN IF AN H325,OR H329
3189 013462 013700 002030          :*IS CONNECTED ON THE DZV UNDER TEST.
3190 013466 005003          :::* TEST 11
3191 013470 012702 000001          :*****
3192 013474 130237 001366    TST11: SCOPE
3193 013500 001003          MOV    #11,$STSTM     :LOAD THE NUMBER OF THIS TEST
3194 013502 005203          MOV    #TST12,NEXT   :POINT TO THE START OF THE NEXT TEST
3195 013504 104420          TST    MODE           :TEST TO SEE IF TESTING WITH
3196 013506 000772          BNE    8$             :CONNECTOR
3197 013510 010204 3$:    ADVANCE              :IF NO, GO TO NEXT TEST
3198 013512 105737 001372 8$:    MOV    #10$,LOCK     :SET FOR TIGHT SCOPE LOOP
3199 013516 100406          DEVICE.CLR           :SET DCLR IN CSR TO ZERO DEVICE
3200 013520 032703 000001          MOV    DZVMSR,R0     :SET REGISTER
3201 013524 001402          CLR    R3             :ZERO LINE NUMBER
3202 013526 006204          MOV    #1,R2         :SET POINTER
3203 013530 000401          BITB   R2,LINE       :TEST THIS LINE?
3204 013532 006304 1$:    BNE    3$            :YES
3205 013534 010405 2$:    INC    R3            :LINE #
3206 013536 000305          SHIFT              :GET NEXT LINE
3207 013540 150405 3$:    BR    1$            :TEST NEXT LINE
3208 013542 150277 166260 10$:  MOV    R2,R4         :SAVE BINARY BIT FOR LINE #
3209 013546 104414          TSTB   MODE          :RUNNING IN EXTERNAL MODE?
3210 013550 011004          BMI    5$            :IF YES SKIP STAGGERED SETUP
3211 013552 020504          BIT    #BIT0,R3     :IF EVEN LINE
                        BEQ    4$            :GO GET ODD PARTNER
                        ASR    R4            :OTHERWISE GET EVEN COMPANION
                        BR    5$            :GO SETUP EXPECTED RESULTS
3212 013554 006304 4$:    ASL    R4            :FIND ODD PARTNER
3213 013556 010405 5$:    MOV    R4,R5         :LOAD R5 FOR EXPECTED
3214 013558 000305          SWAB   R5            :PLACE IN UPPER BYTE
3215 013560 150405          BISB   R4,R5         :SET FOR RING BITS
3216 013562 150277 166260 10$:  BISB   R2,@HDZVTCR   :SET DTR BIT
3217 013564 104414          DELAY              :DELAY FOR CABLE LAG
3218 013566 011004          MOV    (R0),R4       :MOVE RESULTS OF MSR REGISTER TO R4
3219 013568 020504          CMP    R5,R4         :RESULTS=EXPECTED?

```

```

3212 013554 001401          BEQ      6$          :IF YES CONTINUE
3213 013556 104002          ERROR    2          :IF NOT PRINT ERROR RESULTS
3214 013560 104401          6$: SCOP1          :IS SW09 SET?
3215 013562 012737 013570 001364  MOV      #11$,LOCK  :SET UP FOR NEXT TIGHT SCOPE
3216 013570 140277 166232          11$: BICB      R2,@HDZVTCR :CLEAR DTR BIT FOR LINE UNDER TEST
3217 013574 104414          DELAY                    :DELAY FOR CABLE LAG
3218 013576 011004          MOV      (R0),R4      :LOAD MSR REGISTER INTO R4
3219 013600 001402          BEQ      7$          :IF CO AND RING CLEARED CONTINUE
3220 013602 005005          CLR      R5          :OTHERWISE SET EXPECTED FOR ERROR
3221 013604 104002          ERROR    2          :PRINTOUT
3222 013606 104401          7$: SCOP1          :IS SW09 SET?
3223 013610 012737 013542 001364  MOV      #10$,LOCK  :RESET TIGHT SCOPE LOOP
3224 013616 000731          BR       2$          :GET NEXT LINE
3225
3226          :***** TEST 12 *****
3227          :* THIS TEST VERIFIES THAT TRDY IS SET WHEN A LINE
3228          :* IS READY TO BE LOADED, AND THAT THE LINE SPECI-
3229          :* FIED IN BITS 8-9 OF DZVCSR CORRESPOND
3230          :* TO THE LINE SELECTED IN DZVTCR
3231          ::* TEST 12
3232          :*****
3233 013620 000004          TST12: SCOPE
3234 013622 012737 000012 001246  MOV      #12,$STNM   :LOAD THE NUMBER OF THIS TEST
3235 013630 012737 013752 001362  MOV      #TST13,NEXT :POINT TO THE START OF THE NEXT TEST
3236 013636 104413          DEVICE.CLR          :ISSUE A 'DEVICE CLEAR' (RESET)
3237 013640 012737 013674 001364  MOV      #2$,LOCK    :SET UP FOR TIGHT SCOPE LOOP
3238 013646 005037 001374          CLR      SAVLIN     :INITIALIZE FOR ERROR PRINTOUT
3239 013652 013700 002010          MOV      DZVCSR,R0  :SET POINTER
3240 013656 012705 100040          MOV      #MSENAB!TRDY,R5 :START THE EXPECTED LINE NUMBER AT 0
3241 013662 012702 000001          MOV      #1,R2      :USING R2 AS A BIT POINTER, POINT TO LINE 0
3242 013666 130237 001366          1$: BITB      R2,LINE :IS THIS LINE SELECTED?
3243 013672 001421          BEQ      6$          :IF NO, SKIP THE STARTUP
3244 013674 050277 166124          2$: BIS       R2,@DZVTCR :SET THE GO BIT FOR THIS LINE
3245 013700 052710 000040          BIS      #MSENAB,(R0) :START THE SCANNER
3246 013704 005004          CLR      R4          :SET FOR DELAY
3247 013706 005710          3$: TST      (R0)     :TX READY?
3248 013710 100404          BMI      4$          :BR IF YES
3249 013712 104414          DELAY                    :DELAY
3250 013714 005204          INC      R4          :COUNTER
3251 013716 001373          BNE      3$          :BR IF <0!
3252 013720 104003          ERROR    3          :*TX NOT READY!
3253 013722 011004          4$: MOV      (R0),R4   :GET THE LINE POINTED TO BY THE SCANNER
3254 013724 020405          CMP      R4,R5      :IS THE LINE NUMBER WHAT IT SHOULD BE?
3255 013726 001401          BEQ      5$          :IF YES,GO WORK ON THE NEXT LINE
3256 013730 104002          ERROR    2          :*LINE NUMBER DID NOT MATCH TCR BIT
3257 013732 104401          5$: SCOP1          :IS SW09 SET?
3258 013734 104413          DEVICE.CLR          :SET DCLR IN CSR;SETUP FOR NEXT LINE
3259 013736 062705 000400          6$: ADD      #400,R5  :POINT TO THE NEXT EXPECTED LINE
3260 013742 104420          SHIFT                    :POINT TO THE NEXT LINE,ARE ALL LINES TESTED?
3261 013744 005237 001374          INC      SAVLIN     :ADJUST FOR ERROR PRINTOUT
3262 013750 000746          BR       1$          :IF NOT, GO DO THE NEXT LINE
3263          :***** TEST 13 *****
3264          :*TEST TO TRANSMIT ONE CHAR AND
3265          :*RECEIVE ONE CHAR ON ONE LINE
3266          :*AT A TIME, THE CHAR IS '252' AND
3267          :*ALL SELECTED LINES WILL BE TURNED ON .

```

```

3268
3269
3270
3271
3272
3273
3274 013752 000004
3275 013754 012737 000013 001246
3276 013762 012737 014242 001362
3277 013770 012737 014224 001364
3278 013776 104417
3279 014000 104421
3280 014002 005037 001374
3281 014006 105037 001425
3282 014012 012702 000001
3283 014016 012701 000252
3284 014022 057777 000040 165760
3285 014030 030237 001366 38:
3286 014034 001467
3287 014036 010277 165762
3288 014042 005005 58:
3289 014044 105777 165740
3290 014050 100001
3291 014052 104020
3292 014054 005777 165730 68:
3293 014060 100404
3294 014062 104414
3295 014064 005205
3296 014066 001372
3297 014070 104003
3298 014072 105737 001425 78:
3299 014076 001041
3300 014100 105237 001425
3301 014104 110177 165724
3302 014110 013705 001374
3303 014114 005737 001372
3304 014120 100006
3305
3306
3307
3308 014122 006205
3309 014124 103402
3310 014126 000261
3311 014130 000401
3312 014132 000241 88:
3313 014134 006105 98:
3314 014136 000305 108:
3315 014140 150105
3316 014142 052705 100000
3317 014146 005003
3318 014150 105777 165634 118:
3319 014154 100404
3320 014156 104414
3321 014160 005203
3322 014162 001372
3323 014164 104004

```

```

:*THIS IS THE FIRST TIME ANY
:*DATA IS CHECKED IN THE RECEIVER.
:*USING SWITCH NINE WITH THIS TEST CREATES A TIGHT SCOPE LOOP
:*WHICH TRANSMITS A STEADY STREAM OF CHARACTERS.

```

```

::* TEST 13

```

```

*****

```

```

TST13: SCOPE
MOV #13,STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST14,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV #16$,LOCK ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
DCLASM ;SET DCLR IN CSR AND SET MAINT MODE
LPRSET ;LOAD LPR REGISTER FOR ALL LINES
CLR SAVLIN ;INIT. FOR ERROR PRINTOUT
CLRB DONFLG ;INIT FOR TCR BIT HANDLER
MOV #1,R2 ;LINE POINTER
MOV #252,R1 ;SAVE CHARACTER TO BE TRANSMITTED
BIS #MSENAB,@DZVCSR ;START SCANNER
38: BIT R2,LINE ;VALID LINE ?
BEQ 15$ ;NO SET UP NEXT LINE
MOV R2,@DZVTCCR ;SET TCR BIT
58: CLR R5 ;SET R5 FOR A DELAY LOOP
TSTB @DZVCSR ;IS REC DONE = 0 ?
BPL 68 ;IF YES, ALLOW TIME FOR TRDY TO SET
ERROR 20 ;*REC DONE SHOULD = 0
68: TST @DZVCSR ;TRDY SET?
BMI 78 ;IF YES BRANCH
DELAY ;IF NO THEN WAIT FOR IT
INC R5 ;DELAY LOOP
BNE 68 ;BRANCH BACK AND TEST AGAIN
ERROR 3 ;*TRDY FAILED TO SET!
78: TSTB DONFLG ;HAVE WE ALREADY SENT CHARAC.
BNE 13$ ;IF YES GO CLEAR TCR BIT
INCB DONFLG ;IF NOT INDICATE HAVING BEEN HERE
MOVB R1,@DZVTDR ;LOAD CHARACTER
MOV SAVLIN,R5 ;MAKE EXPECTED LINE #
TST MODE ;IS THIS TEST IN STAGGERED MODE?
BPL 108 ;IF NOT, SKIP STAGGERED SETUP

```

```

;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER

```

```

ASR R5 ;GET THE LAST BIT INTO THE CARRY BIT
BCS 88 ;IF IT IS SET, GO CLEAR IT
SEC ;IF IT IS CLEAR SET IT HERE
BR 98 ;SKIP THE CLEARING
88: CLC ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
98: ROL R5 ;GET THE NEW BIT BACK INTO R5
108: SWAB R5 ;MOVE THE LINE NUMBER TO THE UPPER BYTE
BISB R1,R5 ;ADD CHARACTER
BIS #DVALID,R5 ;ADD DATA VALID
CLR R3
118: TSTB @DZVCSR ;IS RDONE SET?
BMI 128 ;IF YES GO GET CHAR.
DELAY ;IF NOT THEN WAIT
INC R3 ;DELAY LOOP
BNE 118 ;DELAY DONE?
ERROR 4 ;*RDONE FAILED TO SET!

```

```

3324 014166 017704 165622      12$:  MOV    @DZVRBUF,R4      ;LOAD THE VALUE ACTUALLY RECEIVED
3325 014172 020405                CMP    R4,R5              ;COMPARE ACTUAL VS EXPECTED. ARE THEY THE SAME?
3326 014174 001722                BEQ    5$                 ;IF YES, GO DO THE NEXT LINE
3327 014176 104006                ERROR  6                 ;*NO DATA/CONTENTS DID NOT COMPARE
3328 014200 000720                BR     5$                 ;GO BACK AND WAIT TO CLEAR TCR BIT
3329 014202 104401                13$:  SCOP1                ;CHECK TO SEE IF SWITCH NINE IS SET
3330 014204 105037 001425        CLR   DONFLG             ;SET UP FOR NEXT LINE
3331 014210 005077 165610        CLR   @DZVTCR           ;CLEAR PREVIOUS TCR BIT
3332 014214 005237 001374        15$:  INC    SAVLIN          ;SET LINE INDICATOR FOR NEXT LINE
3333 014220 104420                SHIFT                ;CALCULATE NEXT LINE
3334 014222 000702                BR     3$                 ;GET GET STARTED
3335
3336 ;TIGHT SCOPE LOOP FOR THIS TEST. LOOP TRANSMITS CHARACTERS ONLY
3337
3338 014224 005777 165560        16$:  TST    @DZVCSR          ;IS TRANSMITTER READY?
3339 014230 100375                BPL    16$               ;IF NOT, WAIT FOR IT
3340 014232 110177 165576        MOV   R1,@DZVTDR        ;LOAD THE CHARACTER
3341 014236 104401                SCOP1                ;LOOP AGIN IF SW09=1
3342 014240 000760                BR     13$              ;OTHERWISE, GO PICK UP THE TEST NORMALLY
3343
3344 ;***** TEST 14 *****
3345 ;*THIS TEST VERIFIES THAT EACH RECEIVING LINE CAN BE
3346 ;*DISABLED BY SETTING RCVON (BIT12 IN THE LPR REGISTER)
3347 ;*TO ZERO FOR EACH LINE.
3348 ;*THIS TEST ALSO VERIFIES THAT THE SILO CAN BE
3349 ;*EMPTIED BY ISSUING A DEVICE MASTER CLEAR.
3350
3351 ;:* TEST 14
3352 ;*****
3353 014242 000004                TST14: SCOPE
3354 014244 012737 000014 001246  MOV    #14,$STSTNM      ;LOAD THE NUMBER OF THIS TEST
3355 014252 012737 014564 001362  MOV    #TST15,NEXT     ;POINT TO THE START OF THE NEXT TEST
3356 014260 105037 001425                CLR   DONFLG           ;CLEAR TEST CONTROL FLAG
3357 014264 005037 001374                CLR   SAVLIN           ;CLEAR LINE INDICATOR
3358 014270 104417                DCLASM                ;ISSUE A DEVICE MASTER CLEAR
3359 014272 013701 001370                MOV    PAR,R1          ;AND SET MAINT BIT IF NECESSARY
3360 014276 042737 010000 001370        BIC   #RCVON,PAR       ;SAVE DEFAULT PARAMETERS
3361 014304 104421                LPRSET                ;DISABLE RECEIVER IN DEFAULT PAR.
3362 014306 010137 001370                MOV    R1,PAR          ;LOAD PARAMETERS IN LPR REGISTER
3363 014312 012701 000252                MOV    #252,R1         ;RESTORE DEFAULT PARAMETERS
3364 014316 013702 001366                MOV    LINE,R2         ;LOAD A CHARAC. INTO R1
3365 014322 010277 165476                MOV    R2,@DZVTCR     ;COPY AN IMAGE OF THE ACTIVE LINES
3366 014326 052777 000040 165454        BIS   #MSENAB,@DZVCSR ;SET TCR BITS FOR ALL ACTIVE LINES
3367 014334 005005                CLR   R5               ;SET MASTER SCAN ENABLE
3368 014336 005777 165446        1$:  CLR   R5               ;INIT DELAY COUNTER
3369 014342 100404                2$:  TST    @DZVCSR       ;IS TRANS READY SET?
3370 014344 104414                BMI   3$               ;BRANCH IF YES
3371 014346 005205                DELAY                ;WAIT FOR TRDY TO SET
3372 014350 001372                INC   R5               ;INCREMENT DELAY COUNTER
3373 014352 104003                BNE   2$              ;RETURN TO CHECK TRDY
3374 014354 117705 165432        ERROR  3               ;TRDY FAILED TO SET!
3375 014360 012703 000001        3$:  MOV   @HDZVCSR,R5     ;MOVE LINE NO. TO R5
3376 014364 042705 177774        MOV   #1,R3           ;INIT TCR POINTER
3377 014370 001403                BIC   #^C<3>,R5       ;ISOLATE LINE NO.
3378 014372 106303                BEQ   31$             ;IF LINE 0 BRANCH
3379 014374 005305                30$: ASLB  R3           ;SHIFT R3 POINTER TO NEXT LINE
                                DEC   R5               ;DECREMENT LINE NO.

```

```

3380 014376 001375
3381 014400 030302
3382 014402 001007
3383 014404 140377 165414
3384 014410 001351
3385 014412 105737 001425
3386
3387 014416 001037
3388 014420 000404
3389 014422 110177 165406
3390 014426 040302
3391 014430 000741
3392 014432 005077 165366
3393 014436 005005
3394 014440 105777 165344
3395 014444 100002
3396 014446 104020
3397 014450 000403
3398 014452 104414
3399 014454 005205
3400 014456 001370
3401 014460 017704 165330
3402 014464 100007
3403 014466 000304
3404 014470 042704 177774
3405 014474 010437 001374
3406 014500 104017
3407 014502 000766
3408 014504 105237 001425
3409 014510 013701 001370
3410 014514 000673
3411
3412 014516 005005
3413 014520 104414
3414 014522 005205
3415 014524 001375
3416 014526 104413
3417 014530 000240
3418 014532 000240
3419 014534 105777 165250
3420 014540 100003
3421 014542 005037 001374
3422 014546 104020
3423 014550 017704 165240
3424 014554 100003
3425 014556 005037 001374
3426 014562 104017
3427 014564
3428
3429
3430
3431
3432
3433
3434
3435

```

31\$: BNE 30\$; WHEN R5=0, R3 POINTS TO LINE TCR
BIT R3,R2 ; HAS CHARACTER BEEN SENT?
BNE 4\$; BRANCH IF NO
BICB R3,@ZVTCR ; IF YES THEN CLEAR TCR BIT
BNE 1\$; IF ALL CHARAC. SENT DROP THROUGH
TSTB DONFLG ; IF NO MORE ACTIVE IS THIS SECOND
; TIME HERE?
BNE 10\$; IF YES SKIP TO SECOND PART OF TEST
BR 5\$; IF FIRST TIME HERE GO ZERO TCR BITS
4\$: MOVB R1,@ZVTDR ; LOAD CHAR. INTO BUFFER
BIC R3,R2 ; INDICATE CHARAC. SENT ON THIS LINE
BR 1\$; GO BACK AND WAIT FOR TRDY TO SET
5\$: CLR @ZVTCR ; CLEAR OUT TCR BITS
CLR R5 ; INIT DELAY COUNTER
6\$: TSTB @ZVCSR ; IS RECEIV. DONE SET?
BPL 7\$; IF NOT THEN WAIT TO SEE IF IT WILL
ERROR 20 ; REC DONE SHOULD NOT SET!
BR 8\$; GO FIND WHICH LINE RECEIVED
7\$: DELAY ; STALL FOR RECEIVER
INC R5 ; INCREMENT DELAY COUNTER
BNE 6\$; IF NOT DONE GO RETEST REC DONE
8\$: MOV @ZVRBUF,R4 ; READ REC. BUFFER
BPL 9\$; IS DVALID SET?
SWAB R4 ; IF YES GET LINE NO.
BIC #C<3>,R4 ; ISOLATE LINE NO.
MOV R4,SAVLIN ; SET UP LINE NO. FOR ERROR REPORT
ERROR 17 ; DVALID SHOULD NOT BE SET
BR 8\$; GO CHECK FOR ANY OTHER CHAR. IN SILO
9\$: INCB DONFLG ; INDICATE THAT FIRST PART OF TEST IS DONE
MOV PAR,R1 ; SAVE DEFAULT LINE PARAM.
BR 100\$; NOW GO RELOAD LPR REGISTER TO
; TURN RECEIVERS ON
10\$: CLR R5 ; ZERO DELAY COUNTER
11\$: DELAY ; WAIT FOR ALL CHARAC. TO BE RECEIVED
INC R5 ; INCREASE DELAY COUNT
BNE 11\$; CONT. DELAY IF NOT FINISHED
DEVICE.CLR ; ISSUE A MASTER CLEAR
12\$: MOV @ZVRBUF,R4 ; NOW IS RECEIV. DONE SET?
BPL 12\$; BRANCH IF NO
CLR SAVLIN ; CLEAR LINE NO FOR ERROR REPORT
ERROR 20 ; REC. DONE SHOULD NOT BE SET!
12\$: MOV @ZVRBUF,R4 ; READ REC. BUFFER
BPL 13\$; IS DVALID SET? IT SHOULDN'T BE
CLR SAVLIN ; DEVICE. CLR DID NOT ZERO SILO
ERROR 17 ; PRINT OUT THE ERROR.(LINE NO. IS IRRELEVANT)
13\$:

:***** TEST 15 *****
:* THIS TEST PROVES THAT THE TRANSMITTER TRANSMITS
:*CHARACTERS (FLAG MODE)AND THE RECEIVER RECEIVES (FLAG MODE)
:*(ONE LINE AT A TIME BASED UPON VALID LINES)
:*THIS IS THE FIRST TIME THAT ALL DATA IS CHECKED
:;* TEST 15
:*****

3436	014564	000004			TST15:	SCOPE		
3437	014566	012737	000015	G01246		MOV	#15,STSTNM	:LOAD THE NUMBER OF THIS TEST
3438	014574	012737	015054	001362		MOV	#TST16,NEXT	:POINT TO THE START OF THE NEXT TEST
3439	014602	012737	014670	001364		MOV	#5\$,LOCK	:USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
3440	014610	104417				DCLASM		:SET DCLR AND SET MNTFLG
3441	014612	104421				LPRSET		:LOAD LPR REGISTER FOR ALL LINES
3442	014614	005037	001374			CLR	SAVLIN	:INIT FOR FIRST LINE
3443	014620	104422				BUFSET		:ZERO BUFFER AREA
3444	014622	105037	001425			CLRB	DONFLG	:ZERO TCR BIT HANDLER FLAG
3445	014626	012702	000001			MOV	#1,R2	:LINE POINTER
3446	014632	052777	000040	165150		BIS	#MSENAB,@DZVCSR	:START SCANNER
3447	014640	030237	001366		3\$:	BIT	R2,LINE	:VALID LINE ?
3448	014644	001477				BEQ	15\$:NO SET UP NEXT LINE
3449	014646	010277	165152			MOV	R2,@DZVTCR	:SET TCR BIT
3450	014652	013700	001374			MOV	SAVLIN,R0	:ADJUST BUFFER POINTER
3451	014656	006300				ASL	R0	:OFFSET
3452	014660	105777	165124		4\$:	TSTB	@DZVCSR	:IS REC DONE = 0 ?
3453	014664	100001				BPL	5\$:IF YES, ALLOW TIME FOR TRDY TO SET
3454	014666	104020				ERROR	20	:*REC DONE SHOULD = 0
3455	014670	005005			5\$:	CLR	R5	:USE R5 AS TIMER WAITING FOR TRDY TO SET
3456	014672	005777	165112		6\$:	TST	@DZVCSR	:IS THE TRANSMITTER READY?
3457	014676	100404				BMI	7\$:IF SO, GO TRANSMIT A CHARACTER
3458	014700	104414				DELAY		:WAIT A LITTLE BIT
3459	014702	005205				INC	R5	:UP THE LOCAL COUNTER.TIME EXCEEDED?
3460	014704	001372				BNE	6\$:IF NOT, GO TRY AGAIN
3461	014706	104003				ERROR	3	:*TRDY FAILED TO SET!
3462	014710	105737	001425		7\$:	TSTB	DONFLG	:ALL CHARAC. TRANS.?
3463	014714	001047				BNE	14\$:IF YES GO ZERO TCR BIT
3464	014716	116077	001426	165110		MOVB	TDO(R0),@DZVTDR	:LOAD CHARACTER
3465	014724	013705	C01374			MOV	SAVLIN,R5	:MAKE EXPECTED LINE #
3466	014730	005737	001372			TST	MODE	:IS THIS TEST IN STAGGERED MODE?
3467	014734	100006				BPL	10\$:IF NOT, SKIP STAGGERED SETUP
3468								
3469								
3470								:WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
3471	014736	006205				ASR	R5	:GET THE LAST BIT INTO THE CARRY BIT
3472	014740	103402				BCS	8\$:IF IT IS SET, GO CLEAR IT
3473	014742	000261				SEC		:IF IT IS CLEAR SET IT HERE
3474	014744	000401				BR	9\$:SKIP THE CLEARING
3475	014746	000241			8\$:	CLC		:CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
3476	014750	006105			9\$:	ROL	R5	:GET THE NEW BIT BACK INTO R5
3477	014752	000305			10\$:	SWAB	R5	:MOVE THE LINE NUMBER TO THE UPPER BYTE
3478	014754	156005	001426			BISB	TDO(R0),R5	:ADD CHARACTER
3479	014760	052705	100000			BIS	#DVALID,R5	:ADD DATA VALID
3480	014764	005003				CLR	R3	
3481	014766	105777	165016		11\$:	TSTB	@DZVCSR	:REC DONE?
3482	014772	100404				BMI	12\$:IF YES GO CHECK CHAR.
3483	014774	104414				DELAY		:IF NOT WAIT FOR REC.
3484	014776	005203				INC	R3	:DELAY LOOP TIMER
3485	015000	001372				BNE	11\$:DELAY FINISHED?
3486	015002	104004				ERROR	4	:*RDONE FAILED TO SET!
3487	015004	017704	165004		12\$:	MOV	@DZVRBUF,R4	:LOAD THE VALUE ACTUALLY RECEIVED
3488	015010	020405				CMP	R4,R5	:COMPARE ACTUAL VS EXPECTED. ARE THEY THE SAME?
3489	015012	001401				BEQ	13\$:IF YES, GO DO THE NEXT LINE
3490	015014	104006				ERROR	6	:*NO DATA/CONTENTS DID NOT COMPARE
3491	015016	104401			13\$:	SCOP1		:CHECK TO SEE IF SWITCH NINE IS SET

```

3492 015020 105260 001426      INCB   TD0(R0)      ;INCREMENT BINARY PATTERN FOR THIS LINE
3493 015024 001315      BNE    4$         ;GO 'ROUND AGAIN FOR NEXT CHARACTER
3494 015026 105237 001425      INCB   DONFLG     ;INDICATE ALL CHAR. SENT
3495 015032 000712      BR     4$         ;BRANCH TO CLEAR TCR BIT
3496 015034 005077 164764      14$:  CLR    @DZVTCR ;CLEAR TCR REGISTER
3497 015040 105037 001425      CLR   DONFLG     ;INIT FOR NEXT LINE
3498 015044 005237 001374      15$:  INC    SAVLIN   ;INC EXPECTED LINE
3499 015050 104420      SHIFT ;SHIFT THE LINE POINTER. ARE WE ALL DONE?
3500 015052 000672      BR     3$         ;IF NO, GO AROUND AGAIN FOR NEXT LINE
3501
3502
3503      ;***** TEST 16 *****
3504      ;*THIS TEST WILL PROVE THAT:
3505      ;* 1) THE TRANSMITTER 'BREAK BIT' WORKS
3506      ;* 2) THE RECEIVER CAN FLAG 'FRAMING ERRORS'
3507      ;* 3) THE RECEIVER CAN FLAG 'PARITY ERRORS'
3508      ;*ONLY ONE LINE AT A TIME WILL BE EXERCISED.
3509      ;:* TEST 16
3510      ;*****
3511 015054 000004      TST16: SCOPE
3512 015056 012737 000016 001246      MOV    #16,$STSTM ;LOAD THE NUMBER OF THIS TEST
3513 015064 012737 015256 001362      MOV    #TST17,NEXT ;POINT TO THE START OF THE NEXT TEST
3514 015072 012737 015202 001364      MOV    #5$,LOCK   ;SET FOR LOOP
3515 015100 005037 001374      CLR    SAVLIN     ;INIT LINE INDIC. FOR ERROR PRINTOUT
3516 015104 012702 000001      MOV    #1,R2      ;LINE POINTER
3517 015110 030237 001366      1$:  BIT    R2,LINE    ;VALID LINE?
3518 015114 001454      BEQ    9$         ;IF NOT SET FOR NEXT LINE
3519 015116 104417      DCLASM ;SET DCLR IN CSR AND SET MNTFLG
3520 015120 013701 001370      MOV    PAR,R1     ;PICK UP PARAMETERS
3521 015124 052737 000300 001370      BIS    #ODDPAR!PARITY,PAR ;FORCE ODD PARITY
3522 015132 104421      LPRSET ;LOAD LPR REGISTER
3523 015134 010137 001370      MOV    R1,PAR     ;RESET PAR TO ORIGINAL VALUE
3524 015140 052777 000040 164642      BIS    #MSENAB,@DZVCSR ;START SCANNER
3525 015146 013705 001374      MOV    SAVLIN,R5  ;MAKE EXPECTED DATA
3526 015152 005737 001372      TST   MODE        ;IS THIS TEST IN STAGGERED MODE?
3527 015156 100006      BPL   4$         ;IF NOT, SKIP STAGGERED SETUP
3528
3529      ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
3530
3531 015160 006205      ASR    R5         ;GET THE LAST BIT INTO THE CARRY BIT
3532 015162 103402      BCS   2$         ;IF IT IS SET, GO CLEAR IT
3533 015164 000261      SEC   ;IF IT IS CLEAR SET IT HERE
3534 015166 000401      BR    3$         ;SKIP THE CLEARING
3535 015170 000241      2$:  CLC   ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
3536 015172 006105      3$:  ROL   R5        ;GET THE NEW BIT BACK INTO R5
3537 015174 000305      4$:  SWAB  R5        ;PUT LINE NUMBER IN UPPER BYTE
3538 015176 052705 130000      BIS    #DVALID!PARER!FRMERR,R5 ;ADD EXPECTED
3539 015202 005003      5$:  CLR    R3        ;INIT DELAY ACCUMULATOR
3540 015204 110277 164626      MOV   R2,@HDZVTDR ;SET BREAK BIT
3541 015210 105777 164574      6$:  TST   @DZVCSR   ;RECEIVER DONE?
3542 015214 100404      BMI   7$         ;BRANCH IF YES
3543 015216 104414      DELAY ;WAIT FOR REC DONE TO SET
3544 015220 005203      INC   R3        ;INC DELAY LOOP
3545 015222 001372      BNE   6$        ;DELAY FINISHED?
3546 015224 104004      ERROR 4          ;*RDONE FAILED TO SET!
3547 015226 017704 164562      7$:  MOV   @DZVRBUF,R4 ;ACTUAL

```

```

3548 015232 020405          CMP      R4,R5          :CMP ACTUAL VS EXPECTED. DO THEY MATCH?
3549 015234 001401          BEQ      8$            :IF YES, GO CLEAN UP
3550 015236 104006          ERROR   6             :*DATA/CONTENTS FAILED TO COMPARE
3551 015240 105077 164572      8$:    CLRB    @HDZVTDR    :CLEAR BREAK BITS
3552 015244 104401          SCOP1   :LOOP?
3553 015246 005237 001374      9$:    INC     SAVLIN     :INC LINE #
3554 015252 104420          SHIFT  :SET R2 TO NEXT LINE
3555 015254 000715          BR      1$           :GO BACK AND TEST NEXT LINE
3556                                     :***** TEST 17 *****
3557                                     :* THIS TEST VERIFIES THAT THE DEVICE DOES NOT INTERRUPT
3558                                     :*WHILE THE PROCESSOR STATUS DOES NOT ALLOW INTERRUPTS
3559                                     :*BUT WILL INTERRUPT IF THE PROCESSOR STATUS
3560                                     :*ALLOWS INTERRUPTS.
3561                                     :::* TEST 17
3562                                     :*****
3563 015256 000004          TST17: SCOPE
3564 015260 012737 000017 001246  MOV     #17,$STNM     :LOAD THE NUMBER OF THIS TEST
3565 015266 012737 015652 001362  MOV     #TST20,NEXT  :POINT TO THE START OF THE NEXT TEST
3566 015274 104417          DCLASM :SET DCLR IN CSR AND SET MAINT BIT
3567                                     :IF NECESSARY (INTERNAL MODE)
3568 015276 104421          LPRSET :SET UP LPR REGISTER
3569 015300 005037 001374          CLR     SAVLIN       :INIT LINE INDIC. FOR ERROR
3570 015304 105037 001425          CLRB   DONFLG       :INIT TCR BIT HANDLER FLAG
3571 015310 113777 001366 164506  MOVB   LINE,@DZVTCR  :SET ALL VALID TCR BITS
3572 015316 106427 000200          MTPS   #MASK        :SET CPU STATUS TO DZV11 PRIO,
3573 015322 012777 000200 164512  MOV     #MASK,@DZVRIS :SET RECEIVER STATUS
3574 015330 012777 000200 164510  MOV     #MASK,@DZVTIS :SET TRANSMITTER STATUS
3575 015336                                     1$:
3576 015336 012777 015424 164500  MOV     #6$,@DZVTIV  :SET UP THE TRANSMITTER INTERRUPT VECTOR
3577 015344 012777 015446 164466  MOV     #7$,@DZVRIV  :SET UP THE RECEIVER INTERRUPT VECTOR
3578 015352 012777 000200 164462  MOV     #MASK,@DZVRIS :SET THE INTERRUPT VECTOR STATUS
3579 015360 012777 000200 164460  MOV     #MASK,@DZVTIS :SET TRANSMITTER INTERRUPT PRIORITY
3580 015366 052777 040040 164414  BIS    #TIE!MSENAB,@DZVCSR :ENABLE THE DEVICE
3581 015374 005005          CLR     R5           :INIT DELAY COUNTER
3582 015376 005777 164406      4$:    TST     @DZVCSR     :TRDY SET?
3583 015402 100003          BPL     5$           :IF NOT GO DO DELAY
3584 015404 000240          NOP
3585 015406 000240          NOP
3586 015410 000420          BR      8$           :GO CLEAR TIE BIT
3587 015412 104414      5$:    DELAY  :DELAY ROUTINE CALL
3588 015414 005205          INC     R5           :INC DELAY COUNTER
3589 015416 001367          BNE    4$           :DELAY FINISHED?
3590 015420 104003          ERROR  3            :*TRDY NOT SET!
3591 015422 000413          BR      8$           :GO CLEAR TIE
3592 015424 022626      6$:    POP2SP :REMOVE THE INTERRUPT FROM THE STACK
3593 015426 042777 040000 164354  BIC    #TIE,@DZVCSR  :DON'T LET ANY MORE INTERRUPTS OCCUR
3594 015434 105737 001425          TSTB   DONFLG       :PROCESSOR ALLOWING INTER?
3595 015440 001013          BNE    10$          :IF YES NO ERROR
3596 015442 104010          ERROR  10           :IF NOT PRINT ERROR
3597 015444 000413          BR      9$           :RETURN TO THE NORMAL FLOW
3598 015446 104012      7$:    ERROR  12           :*RECEIVER SHOULD NOT INTERRUPT
3599 015450 022626          POP2SP :POP FOR FAKE RTI
3600 015452 042777 040000 164330  8$:    BIC    #TIE,@DZVCSR :RESET TRANSMITTER INTERRUPT ENABLE
3601 015460 105737 001425          TSTB   DONFLG       :INTERRUPTS ENABLED?
3602 015464 001403          BEQ    9$           :IF NOT GET OUT
3603 015466 104007          ERROR  7            :IF YES TRANS FAILED TO INTER.

```

```

3604 015470 106427 000000      10$:  MTPS      #CLEAR      :ALLOW INTERRUPTS
3605 015474                                     9$:
3606 015474 012777 015600 164342      MOV      #11$,@DZVTIV  :SET UP THE TRANSMITTER INTERRUPT VECTOR
3607 015502 012777 015604 164330      MOV      #12$,@DZVRIV  :SET UP THE RECEIVER INTERRUPT VECTOR
3608 015510 012777 000200 164324      MOV      #MASK,@DZVRIS :SET THE INTERRUPT VECTOR STATUS
3609 015516 012777 000200 164322      MOV      #MASK,@DZVTIS :SET TRANSMITTER INTERRUPT PRIORITY
3610 015524 052777 000140 164256      BIS      #RIE!MSENA,@DZVCSR :ENABLE THE DEVICE
3611 015532 113777 001426 164274      MOV      TD0,@DZVTDR   :LOAD BUFFER WITH ANY CHAR.
3612 015540 005005                                     CLR      R5             :INIT DELAY ACCUMULATOR
3613 015542 105777 164242      13$:  TSTB      @DZVCSR      :REC. DONE?
3614 015546 100003                                     BPL      14$           :IF NOT DELAY
3615 015550 000240                                     NOP                                     :WAIT FOR INTERRUPT
3616 015552 000240                                     NOP
3617 015554 000404                                     BR       18$
3618 015556 104414      14$:  DELAY                                     :DELAY FOR INTERRUPT
3619 015560 005205                                     INC      R5             :INCREMENT DELAY COUNTER
3620 015562 001367                                     BNE     13$            :DELAY FINISHED?
3621 015564 104004                                     ERROR   4              :*NO RX DONE! (NOT SET)
3622 015566 105737 001425      18$:  TSTB      DONFLG      :PROCESSOR ALLOWING INTERRUPTS?
3623 015572 001411                                     BEQ     15$            :IF NOT DON'T PRINT ERROR
3624 015574 104011                                     ERROR   11             :RECEIVER FAILED TO INTERRUPT
3625 015576 000407                                     BR      15$            :CONTINUE TEST
3626 015600 104010      11$:  ERROR   10             :TRANSMITTER SHOULD NOT INTER.
3627 015602 000404                                     BR      16$            :CONT TEST
3628 015604 105737 001425      12$:  TSTB      DONFLG      :PROCESSOR ALLOWING INTERRUPTS?
3629 015610 001001                                     BNE     16$            :IF YES DON'T PRINT ERROR
3630 015612 104012                                     ERROR   12             :*RECEIVER SHOULD NOT INTERRUPT
3631 015614 022626      16$:  POP2SP                                     :POP FOR FAKE RTI
3632 015616 042777 040100 164164      15$:  BIC      #RIE!TIE,@DZVCSR :CLEAR INTERRUPTS
3633 015624 105737 001425                                     TSTB    DONFLG        :SECOND TIME THROUGH?
3634 015630 001005                                     BNE     17$            :IF YES LEAVE TEST
3635 015632 105237 001425                                     INCB    DONFLG        :IF NO INDICATE SECOND TEST PASS
3636 015636 106427 000000      MTPS      #CLEAR      :ALLOW INTERRUPTS
3637 015642 000635                                     BR      1$             :RESTART TEST
3638 015644 106427 000200      17$:  MTPS      #MASK      :DON'T ALLOW INTERRUPTS
3639 015650 104413      DEVICE.CLR :CLEAR DEVICE, LEAVE TEST
3640
3641 :***** TEST 20 *****
3642 :*THIS TEST VERIFIES THAT THE RECEIVER WILL
3643 :*INTERRUPT BEFORE THE TRANSMITTER EVEN
3644 :*THOUGH THE TRANSMITTER WAS ENABLED
3645 :*FIRST. SET PS TO HIGH (MASK INTERRUPTS);
3646 :*GET RDONE AND TRDY TO SET;
3647 :*SET TX IE AND RX IE;
3648 :*CLEAR PS AND EXPECT RX TO INTERRUPT FIRST
3649 ::* TEST 20
3650 :*****
3651 015652 000004      TST20:  SCOPE
3652 015654 012737 000020 001246      MOV      #20,$STNM    :LOAD THE NUMBER OF THIS TEST
3653 015662 012737 004100 001362      MOV      #SEOP,NEXT   :POINT TO THE END-OF-PASS HANDLER
3654 015670 104417      DCLASM                                     :SET DCLR IN CSR AND MNTFLG
3655 015672 104421      LPRSET                                     :LOAD PAR REGISTER FOR ALL LINES
3656 015674 005037 001374      CLR      SAVLIN       :INIT. ERROR LINE INDIC.
3657 015700 012777 016110 164132      MOV      #8$,@DZVRIV  :SETUP INTERRUPT STUFF
3658 015706 012777 000200 164126      MOV      #MASK,@DZVRIS
3659 015714 012777 016176 164122      MOV      #12$,@DZVTIV

```

```

3660 015722 012777 000200 164116      MOV    #MASK,@DZVTIS      ;
3661 015730 052777 000040 164052      BIS    #MSENAB,@DZVCSR   ;
3662 015736 012702 000001                MOV    #1,R2              ;LINE POINTER
3663 015742 030237 001366      3$:   BIT    R2,LINE        ;VALID LINE ?
3664 015746 001515                BEQ    14$                ;IF NOT GO TO NEXT LINE
3665 015750 106427 000200      4$:   MTPS   #MASK
3666 015754 110277 164044      MOVB   R2,@DZVTCR        ;SET TCR BIT
3667 015760 005777 164030      TST    @DZVRBUF          ;VALID DATA?
3668 015764 100001                BPL    +4                 ;IT BETTER NOT BE SET
3669 015766 104017                ERROR  17                 ;DATA VALID SHOULD NOT BE SET
3670 015770 105777 164014      5$:   TSTB   @DZVCSR        ;RECEIVER DONE ?
3671 015774 100001                BPL    +4
3672 015776 104020                ERROR  20                 ;RECEIVER DONE BIT SHOULD NOT BE SET
3673 016000 005005                CLR    R5
3674 016002 005004                CLR    R4
3675 016004 005777 164000      99$:  TST    @DZVCSR          ;WAIT FOR TRDY
3676 016010 100404                BMI    100$              ;BR IF READY
3677 016012 104414                DELAY
3678 016014 005204                INC    R4
3679 016016 001372                BNE    99$
3680 016020 104003                ERROR  3
3681 016022 105077 164006      100$: CLRB   @DZVTDR          ;TRDY FAILED TO SET
3682 016026 005004                CLR    R4                ;SEND A ZERO CHARACTER
3683 016030 105777 163754      6$:   TSTB   @DZVCSR        ;IS RDONE SET?
3684 016034 100404                BMI    7$
3685 016036 104414                DELAY
3686 016040 005204                INC    R4
3687 016042 001372                BNE    6$
3688 016044 104004                ERROR  4
3689 016046 005777 163736      7$:   TST    @DZVCSR        ;*RDONE FAILED TO SET!
3690 016052 100401                BMI    +4                 ;TRANS DONE BIT = 1 ?
3691 016054 104003                ERROR  3
3692                                ;*NO TRANS DONE FAILED TO SET
3693                                ;NOW THAT BOTH TRANSMITTER AND RECEIVER DONE BIT =1
3694 016056 052777 040000 163724      ;SET INTERRUPT ENABLES
3695 016064 052777 000100 163716      BIS    #TIE,@DZVCSR
3696 016072 106427 000000                BIS    #RIE,@DZVCSR
3697 016076 000240                MTPS   #CLEAR            ;ALLOW THE INTERRUPTS
3698 016100 000240                NOP
3699 016102 104007                ERROR  7
3700 016104 104011                ERROR  11                 ;*TRANSMITTER FAILED TO INTERRUPT
3701 016106 000435                BR     14$                ;*RECEIVER FAILED TO INTERRUPT
3702                                ;GET OUT
3703                                ;RECEIVER INTERRUPT ROUTINE
3704 016110 017704 163700      8$:   MOV    @DZVRBUF,R4      ;ACTUAL
3705 016114 010403                MOV    R4,R3
3706 016116 000303                SWAB   R3
3707 016120 042703 177770      BIC    #^C<7>,R3         ;STRIP JUNK
3708 016124 005737 001372      TST    MODE              ;IS THIS TEST IN STAGGERED MODE?
3709 016130 100006                BPL    11$                ;IF NOT, SKIP STAGGERED SETUP
3710                                ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
3711
3712                                ;GET THE LAST BIT INTO THE CARRY BIT
3713 016132 006203                ASR    R3
3714 016134 103402                BCS   9$
3715 016136 000261                SEC

```

3716	016140	000401			BR	10\$:SKIP THE CLEARING
3717	016142	000241		9\$:	CLC			:CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
3718	016144	006103		10\$:	ROL	R3		:GET THE NEW BIT BACK INTO R3
3719	016146	020337	001374	11\$:	CMP	R3,SAVLIN		:IS THIS A VALID LINE
3720	016152	001401			BEQ	+4		:YES
3721	016154	104015			ERROR	15		:*INVALID LINE
3722	016156	042704	177400		BIC	#*C<377>,R4		:STRIP JUNK
3723	016162	120504			CMPB	R5,R4		:DATA COMPARE ?
3724	016164	001401			BEQ	+4		:YES
3725	016166	104005			ERROR	5		:*DATA DOES NOT COMPARE
3726	016170	040277	163630		BIC	R2,@DZVTCR		:CLEAR TCR BIT
3727	016174	000401			BR	13\$:GO GET OUT OF INTERRUPT MODE
3728								:TRANSMITTER INTERRUPT SVC ROUTINE
3729	016176	104011		12\$:	ERROR	11		:THE RECEIVER INTERRUPT FAILED
3730								:TO OVERRIDE THE TRANSMITTER
3731	016200	022626		13\$:	POP2SP			:REMOVE THE INTERRUPT VECTOR FROM THE STACK
3732	016202	005237	001374	14\$:	INC	SAVLIN		:ADJUST FOR NEXT LINE
3733	016206	104420			SHIFT			:GET THE NEXT POINTER. IF DONE, ADVANCE
3734	016210	000137	015742		JMP	3\$:OTHERWISE GO DO THE NEXT LINE

			:ERROR TABLE	
3735				
3736	016214	000000	.ERRTAB: 0	:ERROR 0
3737	016216	000000	0	
3738	016220	000000	0	
3739				
3740	016222	016362	EM1	:ERROR
3741	016224	017200	DH1	
3742	016226	017320	DT1	
3743				
3744	016230	016435	EM2	:ERROR 2
3745	016232	017224	DH2	
3746	016234	017332	DT2	
3747				
3748	016236	016463	EM3	:ERROR 3
3749	016240	017257	DH3	
3750	016242	017350	DT3	
3751				
3752	016244	016522	EM4	:ERROR 4
3753	016246	017257	DH3	
3754	016250	017350	DT3	
3755				
3756	016252	016551	EM5	:ERROR 5
3757	016254	017271	DH4	
3758	016256	017356	DT4	
3759				
3760	016260	016600	EM6	:ERROR 6
3761	016262	017271	DH4	
3762	016264	017356	DT4	
3763				
3764	016266	016637	EM7	:ERROR 7
3765	016270	017257	DH3	
3766	016272	017350	DT3	
3767				
3768	016274	016700	EM10	:ERROR 10
3769	016276	017257	DH3	
3770	016300	017350	DT3	
3771				
3772	016302	016742	EM11	:ERROR 11
3773	016304	017257	DH3	
3774	016306	017350	DT3	
3775				
3776	016310	017000	EM12	:ERROR 12
3777	016312	017257	DH3	
3778	016314	017350	DT3	
3779				
3780	016316	000000	0	
3781	016320	000000	0	
3782	016322	000000	0	
3783				
3784	016324	000000	0	
3785	016326	000000	0	
3786	016330	000000	0	
3787				
3788	016332	017037	EM15	:ERROR 15
3789	016334	000000	0	
3790	016336	000000	0	

3791				
3792	016340	000000	0	
3793	016342	000000	0	
3794	016344	000000	0	
3795				
3796	016346	017101	EM17	;ERROR 17
3797	016350	017257	DH3	
3798	016352	017350	DT3	
3799				
3800	016354	017137	EM20	
3801	016356	017257	DH3	
3802	016360	017350	DT3	


```

3803
3804 016362 047200 020117 052502 EM1: .ASCIZ <200>/NO BUS REPLY RESPONSE FROM DZV11 REGISTER/
      016435      200 042522 044507 EM2: .ASCIZ <200>/REGISTER R/W FAILURE?
      016463      200 051124 047101 EM3: .ASCIZ <200>/TRANSMIT READY (TRDY) NOT SET/
      016522 051200 041505 044505 EM4: .ASCIZ <200>/RECEIVER DONE NOT SET/
      016551      200 040504 040524 EM5: .ASCIZ <200>/DATA COMPARISON ERROR/
      016600 042200 053132 030461 EM6: .ASCIZ <200>/DZV11 *RECEIVER BUFFER* ERROR/
      016637      200 051124 047101 EM7: .ASCIZ <200>/TRANSMITTER FAILED TO INTERRUPT/
      016700 052600 042516 050130 EM10: .ASCIZ <200>/UNEXPECTED TRANSMITTER INTERRUPT/
      016742 051200 041505 044505 EM11: .ASCIZ <200>/RECEIVER FAILED TO INTERRUPT/
      017000 052600 042516 050130 EM12: .ASCIZ <200>/UNEXPECTED RECEIVER INTERRUPT/
      017037      200 041501 044524 EM15: .ASCIZ <200>/ACTION DETECTED ON INVALID LINE./
      017101      200 040504 040524 EM17: .ASCIZ <200>/DATA VALID SHOULD NOT BE SET/
      017137      200 042522 042503 EM20: .ASCIZ <200>/RECEIVER DONE SHOULD NOT BE SET/

      017200 052200 040522 020120 DH1: .ASCIZ <200>/TRAP PC DZV11 REG/
      017224 042600 050130 041505 DH2: .ASCIZ <200>/EXPECTED FOUND REGISTER/
      017257      200 044514 042516 DH3: .ASCIZ <200>/LINE NO./
      017271      200 054105 042520 DH4: .ASCIZ <200>/EXPECTED FOUND LINE/
  
```

.EVEN

```

3805 017320 000002 DT1: .DATA TABLES FOR ERROR MESSAGES
3806 017322      006      003      2
3807 017324 001330      .BYTE 6,3
3808 017326      006      001      $REG1
3809 017330 001326      .BYTE 6,1
3810      $REG0
3811 017332 000003 DT2: 3
3812 017334      006      004      .BYTE 6,4
3813 017336 001340      $REG5
3814 017340      006      001      .BYTE 6,1
3815 017342 001336      $REG4
3816 017344      006      001      .BYTE 6,1
3817 017346 001326      $REG0
3818
3819 017350 000001 DT3: 1
3820 017352      003      001      .BYTE 3,1
3821 017354 001374      SAVLIN
3822
3823 017356 000003 DT4: 3
3824 017360      006      004      .BYTE 6,4
3825 017362 001340      $REG5
3826 017364      006      001      .BYTE 6,1
3827 017366 001336      $REG4
3828 017370      003      001      .BYTE 3,1
3829 017372 001374      SAVLIN
3830
3831
3832
3833
  
```

:TABLE OF DELAY TIMES FOR INDIVIDUAL BAUD RATES

```

3834 017374 002450 DLYTBL: 2450 :TIME FOR 50 BAUD
3835 017376 001560      1560 :TIME FOR 75 BAUD
3836 017400 001120      1120 :TIME FOR 110 BAUD
3837 017402 000750      750 :TIME FOR 134 BAUD
3838 017404 000660      660 :TIME FOR 150 BAUD
  
```

```

3839 017406 000330
3840 017410 000150
3841 017412 000060
3842 017414 000040
3843 017416 000030
3844 017420 000020
3845 017422 000010
3846 017424 000001
3847 017426 000001
3848 017430 000001
3849 017432 000001
3850
3851
3852
3853
3854 017434 000001
3855

```

```

330
150
60
40
30
20
10
1
1
1
1

```

```

:TIME FOR 300 BAUD
:TIME FOR 600 BAUD
:TIME FOR 1200 BAUD
:TIME FOR 1800 BAUD
:TIME FOR 2000 BAUD
:TIME FOR 2400 BAUD
:TIME FOR 3600 BAUD
:TIME FOR 4800 BAUD
:TIME FOR 7200 BAUD
:TIME FOR 9600 BAUD
:TIME OF DELAY FOR 19200 BAUD

```

```

:DELAYS WERE COMPUTED TO ALLOW MAXIMUM TIME AT EACH BAUD RATE
:FOR ALL TESTS TO FUNCTION CORRECTLY ON A LSI11.

```

```

CORMAX:
.END

```

ABASE = 160010	804#	1145	1186						
ACDW1 = 000017	804#	1145	1188						
ACDW2 = 000000	1145	1189							
ACPLUP= 000000	1145	1160							
ACTIVE 001420	1303#	1561*	2594*	2595	2597*	2601	2602*	2603	2606*
ADDW0 = 017470	804#	1145	1190						
ADDW1 = 017470	804#	1145	1191						
ADDW10= 017470	804#	1145	1200						
ADDW11= 017470	804#	1145	1201						
ADDW12= 017470	804#	1145	1202						
ADDW13= 017470	804#	1145	1203						
ADDW14= 017470	804#	1145	1204						
ADDW15= 017470	804#	1145	1205						
ADDW2 = 017470	804#	1145	1192						
ADDW3 = 017470	804#	1145	1193						
ADDW4 = 017470	804#	1145	1194						
ADDW5 = 017470	804#	1145	1195						
ADDW6 = 017470	804#	1145	1196						
ADDW7 = 017470	804#	1145	1197						
ADDW8 = 017470	804#	1145	1198						
ADDW9 = 017470	804#	1145	1199						
ADEVCT= 000000	1145	1151							
ADEVN = 000001	804#	1145	1187						
ADRCNT 005675	2128*	2165*	2175#						
ADVANC= 104400	1455#	2305	2914	3186					
AENV = 000000	1145	1156							
AENVN = 000000	1145	1157							
AFATAL= 000000	1145	1148							
AMADR1= 000000	1145	1173							
AMADR2= 000000	1145	1177							
AMADR3= 000000	1145	1180							
AMADR4= 000000	1145	1183							
AMAMS1= 000000	1145	1167							
AMAMS2= 000000	1145	1175							
AMAMS3= 000000	1145	1178							
AMAMS4= 000000	1145	1181							
AMSGAD= 000000	1145	1153							
AMSGLG= 000000	1145	1154							
AMSGTY= 000000	1145	1147							
AMTYP1= 000000	1145	1168							
AMTYP2= 000000	1145	1176							
AMTYP3= 000000	1145	1179							
AMTYP4= 000000	1145	1182							
APASS = 000000	1145	1150							
APRIOR= 000000	1145								
APTCSU= 000040	1979	2084#							
APTENV= 000001	1972	2040	2082#	2392					
APTSIZ= 000200	2081#								
APTSPO= 000100	1974	2042	2083#						
ASWREG= 000000	1145	1158							
AESTN= 000000	1145	1149							
ALNIT = 000000	1145	1152							
AUSWR = 000000	1145	1159							
AUTO.S 011500	1736	2777#							
AVECT1= 000300	804#	1145	1184						
AVECT2= 000000	1145	1185							

SW03 = 000010	901#	911	1638	1666				
SW04 = 000020	900#	910	1772					
SW05 = 000040	899#	909						
SW06 = 000100	898#	908	1751					
SW07 = 000200	897#	907						
SW08 = 000400	896#	906	2406					
SW09 = 001000	895#	905	1938					
SW1 = 000002	913#							
SW10 = 002000	894#	2408						
SW11 = 004000	893#							
SW12 = 010000	892#	1945	2337					
SW13 = 020000	891#	2342						
SW14 = 040000	890#							
SW15 = 100000	889#							
SW2 = 000004	912#							
SW3 = 000010	911#							
SW4 = 000020	910#							
SW5 = 000040	909#							
SW6 = 000100	908#							
SW7 = 000200	907#							
SW8 = 000400	906#							
SW9 = 001000	905#							
S110 = 001000	1036#							
S1200 = 003400	1041#							
S134 = 001400	1037#							
S150 = 002000	1038#							
S1800 = 004000	1042#							
S19200 = 007400	1049#							
S2000 = 004400	1043#							
S2400 = 005000	1044#							
S300 = 002400	1039#							
S3600 = 005400	1045#							
S4800 = 006000	1046#							
S50 = 000000	1034#							
S600 = 003000	1040#							
S7200 = 006400	1047#							
S75 = 000400	1035#							
S9600 = 007000	1048#							
TBITVE = 000014	947#							
TCR0 = 000001	1053#	3022						
TCR1 = 000002	1054#	3023						
TCR2 = 000004	1055#	3024						
TCR3 = 000010	1056#	3025						
TD0 = 001426	1314#	2326	3464	3478	3492*	3611		
TD1 = 001430	1315#							
TD2 = 001432	1316#							
TD3 = 001434	1317#							
TEIGHT = 002106	1540#							
TEMP = 010346	2570#							
TFIVE = 002114	1543#							
TIE = 040000	983#	2987	3093	3580	3593	3600	3632	3694
TKVEC = 000060	954#							
TLAST = 015652	2642	3831#						
TLO = 000000	988#							
TL1 = 000400	989#							
TL2 = 001000	990#							

XERR	004306	1833	1867#											
XHEAD	010006	1740	2512#											
XPTCNT	001400	1293#												
XPTLIN	001376	1292#												
XPASS	004300	1830	1864#											
XSTATQ	010076	1747	2512#											
XTSTN	007000	2371	2417#											
XVEC	004272	1827	1861#											
XX =	160210	1350#	1356#	1362#	1368#	1374#	1380#	1386#	1392#	1398#	1404#	1410#	1416#	1422#
		1428#	1434#	1440#	1446#									
YY =	000500	1350#	1356#	1362#	1368#	1374#	1380#	1386#	1392#	1398#	1404#	1410#	1416#	1422#
		1428#	1434#	1440#	1446#									
ZZ =	000020	1350#	1356#	1362#	1368#	1374#	1380#	1386#	1392#	1398#	1404#	1410#	1416#	1422#
		1428#	1434#	1440#	1446#									
SAPTHD	001446	1332	1338#											
SASTAT=	***** U	2062	2077											
SATYC	005130	2033	2035#											
SATY1	005104	2031#												
SATY3	005112	1977	2032#											
SATY4	005122	2034#	2395											
SAUTOB	001300	1233#												
SBASE	001174	1186#	1604*	2607*	2662	2741	2830*							
SBDADR	001266	1228#												
SDDAT	001272	1230#												
SCDW1	001200	1188#	2760											
SCDW2	001202	1189#	2762	2831*										
SCHARC	005100	1994*	2004*	2011	2020*	2025#								
SCMTAG	001244	1216#												
SCM1 =	000006	1248#	1249#	1250#	1251#	1252#	1253#	1254#						
SCM2 =	000014	1248#	1249#	1250#	1251#	1252#	1253#	1254#						
SCM3 =	000006	1246#	1248											
SCM4 =	000005	1254#	1255#	1256#	1257#	1258#	1259#							
SCPUOP	001146	1160#												
SCRLF	001357	1261#	1993	2028	2205	2365	2366	2376	2457	2621	2639			
SDDW0	001204	1190#	1651	2745	2781									
SDDW1	001206	1191#												
SDDW10	001230	1200#												
SDDW11	001232	1201#												
SDDW12	001234	1202#												
SDDW13	001236	1203#												
SDDW14	001240	1204#												
SDDW15	001242	1205#												
SDDW2	001210	1192#												
SDDW3	001212	1193#												
SDDW4	001214	1194#												
SDDW5	001216	1195#												
SDDW6	001220	1196#												
SDDW7	001222	1197#												
SDDW8	001224	1198#												
SDDW9	001226	1199#												
SDEVCT	001130	1151#	1834*											
SDEVN	001176	1187#	2746	2833*										
SDOAGN	004260	1836	1843	1848	1854#									
SE =	000022	804#	2885	2886#	2928	2929#	2951	2952#	2999	3000#	3058	3059#	3089	3090#
		3128	3129#	3152	3153#	3183	3184#	3235	3236#	3276	3278#	3354	3355#	3438
		3440#	3513	3514#	3565	3566#	3653	3654#						

SENDAD	004250	1126	1573	1850#	2398									
SENDCT	004234	1845#												
SENV	001140	1156#	1972	2040	2064	2392								
SENVH	001141	1157#	1577	1974	1979	2042								
SEOP	004100	1819#	3653											
SEOPCT	004226	1842#	1846											
SERFLG	001247	1219#	1560*	1822*	1879	1908	1910*	1934	2347*	2361	2377*			
SERMAX	001261	1225#	1934											
SERROR	006360	1116	1557	2336#										
SERRPC	001262	1226#	1564*	1821*	1887*	2344	2346*							
SERRTB	001362	1277#												
SERTTL	001256	1223#	1563*	1869	2404*									
SETABL	001140	1155#												
SETEND	001244	1208#	1344											
SFATAL	001122	1148#	2068*											
SFFLG	005350	2031*	2034*	2062	2071*	2079#								
SFILLC	001322	1244#	1997	2028										
SFILLS	001321	1243#	2028											
SFLIP =	177777	1#	2877#	2881#	2920#	2924#	2939#	2947#	2990#	2995#	3048#	3054#	3080#	3085#
		3120#	3124#	3144#	3148#	3169#	3178#	3226#	3231#	3263#	3272#	3344#	3350#	3429#
		3434#	3503#	3509#	3556#	3561#	3641#	3649#						
SGDADR	001264	1227#												
SGDDAT	001270	1229#												
SGET42	004240	1847#												
SHD =	000001	813	814											
SHIBTS	001446	1339#												
SICNT	001250	1220#	1916*	1917	1919*	1933								
SILLUP	007416	2474	2490	2509#										
SINTAG	001301	1234#												
SITEMB	001260	1224#	2352*	2394										
SLF	001360	1262#	2028											
SLFLG	005347	2072*	2078#											
SLPADR	001252	1221#	1566*	1803*	1806	1923*	1925	1933	2410*	2412	2637*	2638*	2646*	2648
SLPERR	001254	1222#												
SPADR1	001152	1173#												
SPADR2	001156	1177#												
SPADR3	001162	1180#												
SPADR4	001166	1183#												
SPAIL	001120	1146#	1340	1344	1922	1972								
SPAPIS1	001150	1167#												
SPAPIS2	001154	1175#												
SPAPIS3	001160	1178#												
SPAPIS4	001164	1181#												
SPBADR	001450	1340#												
SPFLG	005346	2032*	2038	2073*	2077#									
SPSGAD	001134	1153#	2048*	2051										
SPSGLG	001136	1154#	2053*											
SPSGTY	001120	1147#	2046	2054*	2066	2070*								
SMTYP1	001151	1168#												
SMTYP2	001155	1176#												
SMTYP3	001161	1179#												
SMTYP4	001165	1182#												
SPXCNT	004556	1920	1933#	2589										
SN =	000020	304#	2877	2881	2886#	2920	2924	2929#	2939	2947	2952#	2990	2995	3000#
		3048	3054	3059#	3080	3085	3090#	3120	3124	3129#	3144	3148	3153#	3169
		3179	3184#	3226	3231	3236#	3263	3272	3278#	3344	3350	3355#	3429	3434

.1170 1#

. ABS. 017434 000

ERRORS DETECTED: 0

RUN-TIME: 14 18 1 SECONDS
RUN-TIME RATIO: $48/34=1.4$
CORE USED: 50K (99 PAGES)