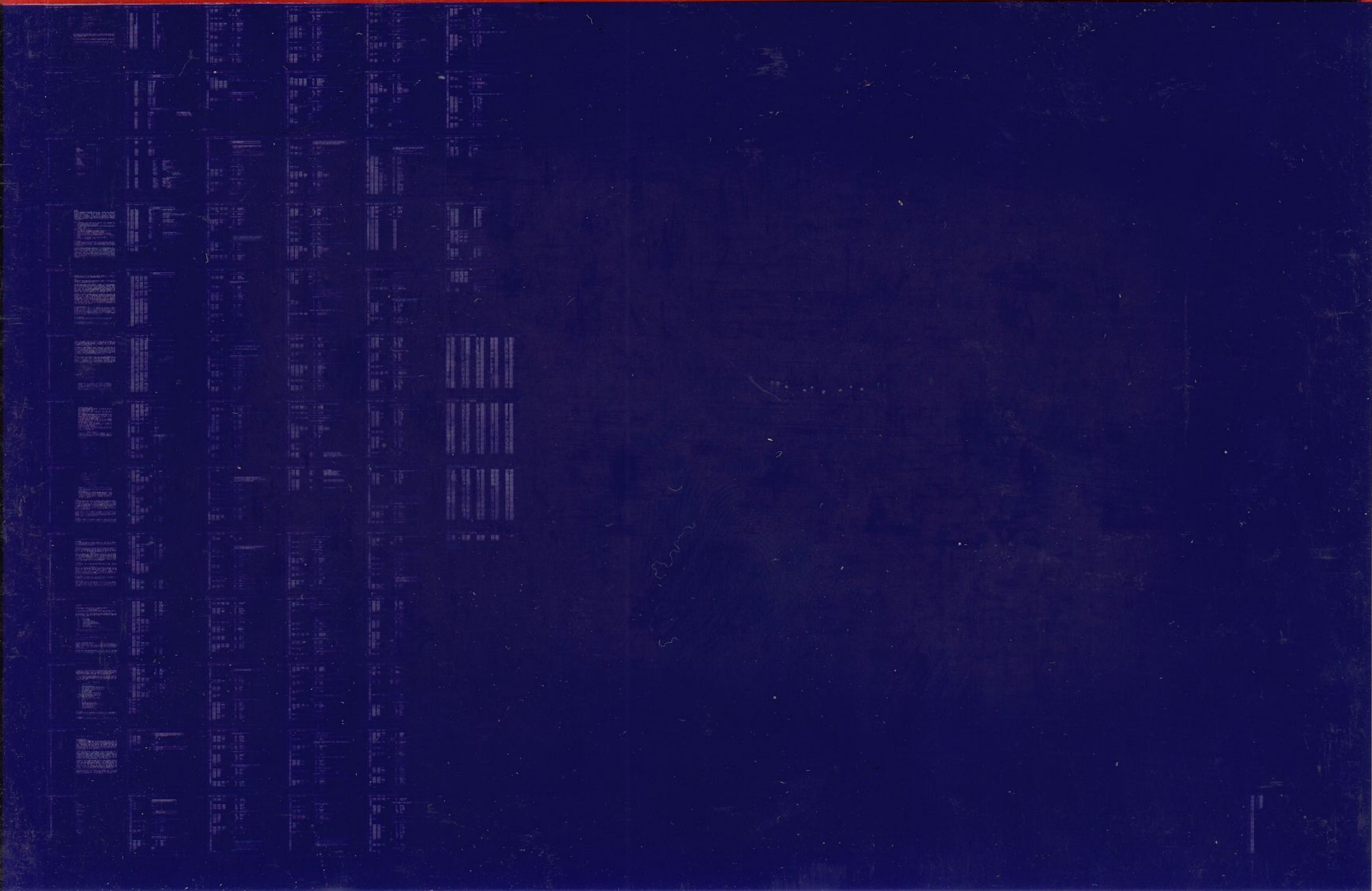


VT62

VT62/DUP11 SELF-TEST/LOOPBACK EP-DZDPG-A-DL
MD-11-DZDPG-A

COPYRIGHT © 1977
FICHE 1 OF 1

JAN 1978
digital
MADE IN USA



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

000000 .REPT 0

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZDPG-A-D
PRODUCT NAME: VT62/DUP11 SELFTTEST/LOOPBACK
DATE: 1-SEP-77
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHOR: MICHAEL DENSMORE

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 WITHIN.

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 DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

39
40
41
42
43
44
45

REVISION HISTORY

REVISION
A

BY:
M. DENSMORE ORIGINAL RELEASE

47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84

TABLE OF CONTENTS

1.	GENERAL	PG.	4
1.1	ABSTRACT	PG.	4
1.2	DEFINITIONS	PG.	4
1.3	DESIGN NOTES	PG.	4
1.4	DDCMP	PG.	5
2.	EQUIPMENT REQUIREMENTS	PG.	5
3.	LOAD PROCEDURE	PG.	5
4.	STARTING AND RUNNING THE PROGRAM	PG.	5
4.1	START	PG.	6
4.2	RESTART	PG.	8
4.3	CONTINUE	PG.	8
4.4	ZERO FLAGS	PG.	9
4.5	ADD UNITS	PG.	9
4.6	DROP UNITS	PG.	9
4.7	DISPLAY PTABLE	PG.	9
4.8	SWITCH OPTIONS	PG.	9
5.	ERROR AND INFORMATIONAL MESSAGES	PG.	10
6.	RESTRICTIONS	PG.	11
7.	MISC.	PG.	11
8.	TEST DESCRIPTIONS	PG.	12
8.1	MICRODIAGNOSTIC	PG.	12
8.2	LOOPBACK	PG.	12
9.	INDEX	PG.	13

86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138

- 1.0 GENERAL
- 1.1 ABSTRACT
MAINDEC-11-DZDPG RUNS THE SELFTTEST FOR THE VT62 TERMINAL AND EXERCISES THE LINK BETWEEN THE TERMINAL AND THE HOST. THIS PARTICULAR VERSION SUPPORTS MULTIDROP, FULL DUPLEX COMMUNICATION VIA THE DUP-11 COMMUNICATION INTERFACE. OTHER VERSIONS (MAINDEC-11-DZDZC AND MAINDEC-11-DZDVP) SUPPORT THE DZ-11 AND DV-11 COMMUNICATION INTERFACES RESPECTIVELY. THE PROGRAM IS XXDP COMPATIBLE AND SUPPORTS SEQUENTIAL TESTING OF UP TO 8 VT62'S. PLEASE READ ALL SECTIONS OF THIS DOCUMENT CAREFULLY BEFORE USING THE PROGRAM. PAY PARTICULAR ATTENTION TO SECTIONS 3.0 THROUGH 6.0
- 1.2 DEFINITIONS
MODEM - EQUIPMENT, ALSO KNOWN AS A "DATASET", WHICH PERFORMS THE MODULATION/DEMULATION OF DATA SIGNALS. THIS UNIT PROVIDES TIMING FOR SYNCHRONOUS COMMUNICATION DEVICES.
MODEM ELIMINATOR - EQUIPMENT WHICH CONNECTS TWO SYNCHRONOUS DEVICES USING A CABLE RATHER THAN TELEPHONE VOICE LINES. THE ELIMINATOR HAS A CLOCK FOR TIMING.
DDCMP PROTOCOL - DEC'S COMMUNICATION LINE PROTOCOL.
UUT - UNIT UNDER TEST.
PTABLE - A TABLE OF INFORMATION, WITH ENTRIES FOR EACH UNIT UNDER TEST, THAT DESCRIBES THE HARDWARE (BUS ADDRESS, VECTOR, ETC.)
DUP - SINGLE LINE, SYNCHRONOUS COMMUNICATION INTERFACE.
DZ - EIGHT LINE, MULTIPLEXED, SYNCHRONOUS COMMUNICATION INTERFACE.
DV - SIXTEEN LINE, MULTIPLEXED, SYNCHRONOUS COMMUNICATION INTERFACE.
CRC CHECKSUM - CHECKSUM USED IN DEC NETWORK PROTOCOL (CYCLIC REDUNDANCY CHECK).
MICRODIAGNOSTIC - THE MICROCODE IN THE VT62 THAT PERFORMS THE DIAGNOSTIC CHECKOUT OF THE HARDWARE. (ALSO KNOWN AS "SELF TEST")
SELF TEST - SEE "MICRODIAGNOSTIC".
- 1.3 DESIGN NOTES
THIS PROGRAM WAS DESIGNED AND IMPLEMENTED USING STRUCTURED TECHNIQUES. THE PROCEDURE CAUSED THE PROGRAM TO HAVE SOME OPERATIONAL CHARACTERISTICS WHICH ARE OF INTEREST TO THE USER. THESE CHARACTERISTICS ARE DISCUSSED IN THIS SECTION. THIS DISCUSSION IS NOT VITAL TO LEARNING HOW TO OPERATE THE PROGRAM HOWEVER.

THE MOST IMPORTANT OPERATIONAL CHARACTERISTIC OF STRUCTURED DESIGN, AS IT AFFECTS THE USER, IS HOW ERRORS ARE REPORTED. THE TOP-DOWN DESIGN OF THE PROGRAM RESULTS IN A BOTTOM-UP REPORT OF ERROR INFORMATION. SEVERAL LINES OF ERROR INFORMATION ARE PRINTED AS EACH PROGRAM SECTION REPORTS ITS LEVEL OF INFORMATION. AN EXAMPLE OF THIS TYPE OF ERROR LOG WOULD BE A DUP ERROR. THE FIRST ERROR MESSAGE WOULD GIVE SPECIFIC DEVICE INFORMATION SUCH AS REGISTER CONTENTS AND THEN RETURN CONTROL TO THE NEXT LEVEL. AT THE NEXT LEVEL, THE LOGICAL DEVICE NUMBER OR DEVICE SERIAL NUMBER WOULD BE REPORTED. AT THE LEVEL ABOVE THAT, STILL MORE GENERAL INFORMATION WOULD BE GIVEN. THE REPORTS CONTINUE UNTIL THE HIGHEST LEVEL IS REACHED.

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

COMMENT FIELDS IN THE LISTING WITH SQUARE BRACKETS () REFER TO FUNCTIONAL BLOCKS IN THE DESIGN AND HAVE NO SIGNIFICANCE TO THE USER. THEY ARE RETAINED FOR USE BY PROGRAM MAINTAINERS.

1.4

DDCMP

THIS SECTION GIVES A VERY BRIEF DESCRIPTION OF DDCMP AS USED BY THIS DIAGNOSTIC. THE DESCRIPTION IS BY NO MEANS COMPLETE; IT IS INTENDED TO HELP THE USER UNDERSTAND ERROR REPORTS.

THERE ARE THREE TYPES OF DDCMP MESSAGES: CONTROL, DATA AND BOOTSTRAP. THE BOOTSTRAP MESSAGE IS NOT USED BY THIS DIAGNOSTIC. THE DATA MESSAGES CONSIST OF A 6-BYTE HEADER, A 2-BYTE CRC FOR THE HEADER, THE DATA ITSELF AND A 2-BYTE CRC FOR THE DATA. CONTROL MESSAGES CONSIST OF THE HEADER (AND ITS CRC) ONLY. ALL CONTROL MESSAGES HAVE AN "ENQ" CHARACTER AS THE FIRST BYTE. ALL DATA MESSAGES ARE NUMBERED SO THAT EACH STATION ON THE LINK CAN KEEP TRACK OF WHICH MESSAGES ARE TRANSMITTED PROPERLY. ALL MESSAGES HAVE A STATION ADDRESS IN THE HEADER SO THAT MESSAGES ARE DIRECTED TO AND FROM THE PROPER STATIONS ON A MULTIDROP LINE.

THERE ARE FOUR CONTROL MESSAGES USED BY THIS DIAGNOSTIC: ACK, NAK, STRT AND STACK. ACK IS AN ACKNOWLEDGEMENT THAT THE LAST MESSAGE SENT WAS PROPERLY RECEIVED. NORMALLY, MESSAGES ARE ACKNOWLEDGED BY SENDING A DATA MESSAGE WITH THE NUMBER OF THE LAST MESSAGE IN THE HEADER. ACK'S ARE USED WHEN THERE IS NO DATA MESSAGE TO SEND BACK. NAK IS A NEGATIVE ACKNOWLEDGEMENT FOR THE LAST MESSAGE SENT AND INDICATES THE REASON FOR THE ERROR. STRT IS USED TO INITIATE COMMUNICATION BETWEEN TWO STATIONS. THE RECEIVING STATION SETS ITSELF UP AND ACKNOWLEDGES THE STRT WITH A STACK, OR START ACKNOWLEDGE. THE STRT-STACK SEQUENCE IS USED BEFORE AND AFTER EACH TEST. ALL MESSAGES ARE REQUIRED TO BE ACKNOWLEDGED WITH EITHER A DATA MESSAGE OR AN ACK.

2.0

EQUIPMENT REQUIREMENTS

THIS PROGRAM REQUIRES A PDP-11 WITH A MINIMUM OF 16K CORE AND A CONSOLE TERMINAL. A CLOCK OPTION IS NOT REQUIRED, BUT ONLY ROUGH TIMING IS AVAILABLE WITHOUT ONE. THE PDP-11 CONSOLE SWITCH REGISTER IS NOT USED. ONE OR MORE DUP-11'S MUST BE CONNECTED TO THE TERMINALS UNDER TEST VIA COMMUNICATION LINES THROUGH EITHER MODEM'S OR MODEM ELIMINATORS. THE VT62'S MUST BE IN SYNCHRONOUS MODE AND ON-LINE. THE TERMINAL ADDRESS SWITCHES ON THE CONTROL BOARD MUST BE SET TO MATCH THE TERMINAL ADDRESS ASSIGNED TO THE LOGICAL UNIT (SEE SECTION 4.0).

3.0

LOADING PROCEDURE

THIS PROGRAM MAY BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER. IT MAY ALSO BE LOADED FROM PAPER TAPE OR DISK USING XXDP PROCEDURES.

4.0

STARTING AND RUNNING THE PROGRAM

THE STARTING ADDRESS FOR THE PROGRAM IS 200. THE PROGRAM WILL FIRST ASK IF A CLOCK (KW11-L OR KW11-P) IS AVAILABLE. IT WILL THEN ASK IF A LINE

195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248

PRINTER IS AVAILABLE. IF A LINE PRINTER IS SPECIFIED, THE PROGRAM WILL ASK FOR ITS ADDRESS (DEFAULT = 177514). THE PROGRAM ALSO ASKS FOR MEMORY SIZE, Q-BUS OPTION AND LINE FREQUENCY. IF NEITHER CLOCK IS SPECIFIED, THE PROGRAM WILL ASK THE USER TO TYPE TWO CHARACTERS, FIVE SECONDS APART, IN ORDER TO PROVIDE ROUGH TIMING.

THE PROGRAM WILL THEN PROMPT THE USER BY TYPING A ">". NOW THE USER MAY TYPE ONE OF SEVEN COMMANDS: STA(RT), RES(TART), CON(TINUE), ZFL(AG), ADD, DRO(P) OR DIS(PLAY), WHERE THE LETTERS IN PARENTHESIS NEED NOT BE TYPED. IF A TYPING MISTAKE IS MADE, THE RUBOUT KEY IS USED TO DELETE THE LAST CHARACTER(S) TYPED. THE PROGRAM WILL TYPE A MESSAGE IF THE COMMAND IS INVALID. EACH OF THESE COMMANDS IS EXPLAINED IN THE SECTIONS BELOW. SWITCHES ARE USED TO SET SOFTWARE FLAGS SUCH AS "LOOP ON ERROR". THE FLAGS USED BY THIS PROGRAM ARE DESCRIBED IN SECTION 4.8.

4.1

START
THE START COMMAND (STA) ZEROS ALL FLAGS (PRINT ALL ERROR MESSAGES AND DO NOT LOOP WITHIN TESTS) AND INITIATES THE OPERATOR DIALOGUE. AT THIS POINT, THE DEVICE TABLE (PTABLE) IS EMPTY AND THE OPERATOR MUST USE THE DIALOGUE TO DESCRIBE ALL UUT'S AS SHOWN BELOW. THE NUMBERS IN SQUARE BRACKETS, , RELATE THE TYPEOUT TO THE EXPLANATION BELOW AND ARE NOT TYPED IN ACTUAL OPERATION. THE PROCEDURES FOR HANDLING TYPING ERRORS ARE ALSO DESCRIBED BELOW. CR REFERS TO THE RETURN KEY (CARRIAGE RETURN).

DIALOGUE

UNITS ? (DECIMAL) 1
DUP ADDRESS: ? (OCTAL) NNNNNN 2
DUP VECTOR: ? (OCTAL) NNN 3
DUP BUS PRIORITY: ? (OCTAL) N 4
TERMINAL ADDRESS: ? (OCTAL) N 5
CHANGE S.W. ? (Y OR N) 6
HOW MANY INITIATE TRIES BEFORE ABORT ? (OCTAL) 10 7
USE ALL ZERO'S FOR LOOPBACK DATA ? (Y OR N) Y 8

EXPLANATION

- 1 TYPE THE TOTAL NO. OF UUT'S IN DECIMAL. QUESTIONS 2 THROUGH 5 WILL BE REPEATED UNTIL ALL UUT'S HAVE BEEN DESCRIBED. THE USER DOES NOT HAVE TO ANSWER THESE QUESTIONS FOR EACH DUP HOWEVER, IF THERE ARE MULTIPLE UUT'S PER DUP. THE EXPLANATION FOR QUESTION 5 SHOWS HOW TO ENTER MULTIPLE UUT'S.
- 2 TYPE THE BUS ADDRESS OF THE DUP. THIS MUST BE AN EVEN OCTAL NUMBER NOT GREATER THAN 177776. THE USER MAY TYPE CR TO USE

250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303

- THE DEFAULT VALUE: NNNNNN.
- 3 TYPE THE VECTOR ADDRESS FOR THE DUP. THIS MUST BE AN EVEN OCTAL NUMBER NOT GREATER THAN 376. THE USER MAY TYPE CR TO USE THE DEFAULT VALUE: NNN.
- 4 TYPE THE BUS PRIORITY LEVEL OF THE DUP. THIS MUST BE A DIGIT FROM 0 TO 7. AGAIN, THE USER MAY TYPE CR TO GET THE DEFAULT VALUE: N.
- 5 TYPE THE TERMINAL ADDRESS FOR EACH TERMINAL ATTACHED TO THE DUP. SEPARATE EACH ADDRESS WITH A COMMA. THIS IS THE ADDRESS SELECTED BY THE SWITCHES ON THE I/O BOARD IN THE TERMINAL AND IS AN OCTAL NUMBER BETWEEN 0 AND 377. THE USER MAY TYPE CR TO GET THE DEFAULT VALUE: N.
- 6 IF THE USER WISHES TO CHANGE EITHER OF THE TWO SOFTWARE PARAMETERS DESCRIBED FOR QUESTIONS 7 OR 8, HE MUST ANSWER THIS QUESTION BY TYPING "Y". IF THE USER WISHES TO BYPASS QUESTIONS 7 AND 8, HE TYPES "N".
- 7 TYPE THE NUMBER OF TIMES THE PROGRAM IS TO ATTEMPT TO INITIALIZE A DUP BEFORE ABORTING THE UNIT. THE USER MAY TYPE ANY OCTAL NUMBER BETWEEN 0 AND 177777. TYPE CR TO DEFAULT TO 10 (OCTAL) ATTEMPTS.
- 8 IF THE USER TYPES "Y", THE LOOPBACK DATA WILL BE 255 ZERO BYTES. IF THE USER TYPES "N", THE LOOPBACK DATA WILL BE A PATTERN OF ALTERNATING 1 AND 0 BITS. SEE SECTION 8.2 FOR DETAILS ON THE LOOPBACK TEST.

EXAMPLE OF DIALOGUE

ASSUME THE FOLLOWING HARDWARE CONFIGURATION: A DUP (160000,300,3) WITH TWO TERMINALS (ADDRESSES 0 AND 1), A SECOND DUP (160010,320,3) WITH A SINGLE TERMINAL (ADDRESS 5) AND A THIRD DUP (170000,340,4) WITH TWO TERMINALS (ADDRESSES 1 AND 2). THE USER WOULD USE THE FOLLOWING DIALOGUE TO SET UP THE PROGRAM:

```
>STA  
  
# UNITS ? (DECIMAL) 5  
  
DUP ADDRESS: ? (OCTAL) 160000  
  
DUP VECTOR: ? (OCTAL) 310 300  
  
DUP BUS PRIORITY: ? (OCTAL) 4 3  
  
TERMINAL ADDRESS: ? (OCTAL) 0 0,1  
  
DUP ADDRESS: ? (OCTAL) 160000 160010  
  
DUP VECTOR: ? (OCTAL) 300 320  
  
DUP BUS PRIORITY: ? (OCTAL) 3  
  
TERMINAL ADDRESS: ? (OCTAL) 1 5
```


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

DUP ADDRESS: ? (OCTAL) 160010 170000

DUP VECTOR: ? (OCTAL) 320 340

DUP BUS PRIORITY: ? (OCTAL) 3 5

TERMINAL ADDRESS: ? (OCTAL) 5 1,2

CHANGE S. W. ? (Y OR N) N

(NOTE WHERE CR WAS USED TO ACCEPT THE DEFAULT)

TYPING ERRORS

- (1) THE RUBOUT KEY IS USED TO DELETE ANY CHARACTERS WHICH ARE IN ERROR.
- (2) IF THE RESPONSE IS NOT WITHIN LIMITS OR IS IN THE WRONG FORMAT, THE PROGRAM WILL NOTIFY THE USER AND ASK THE QUESTION AGAIN.
- (3) WHEN ENTERING MULTIPLE TERMINAL ADDRESSES FOR A SINGLE DUP, THE USER MAY NEGLECT TO TYPE ONE OR MORE ADDRESSES. IN SUCH A CASE, THE USER MERELY ENTERS THE SAME DUP ADDRESS, VECTOR AND PRIORITY IN THE NEXT SERIES OF QUESTIONS AND THEN ENTERS THE REMAINING TERMINAL ADDRESSES. REMEMBER; THE SERIES OF QUESTIONS (2 THROUGH 5) WILL BE RE-ASKED UNTIL ALL UUT'S HAVE BEEN ACCOUNTED FOR.

4.2

RESTART

THE RESTART COMMAND (RES) STARTS THE PROGRAM WITHOUT CLEARING THE PTABLE. ALL FLAGS REVERT TO THEIR INITIAL STATE UNLESS THE FLAG SWITCH IS USED TO CHANGE THEM (SECTION 4.8). THE OPERATOR DIALOGUE DESCRIBED IN 4.1 DOES NOT TAKE PLACE UNLESS REQUESTED. WHEN THIS COMMAND IS USED, THE PROGRAM WILL ASK IF THE USER WISHES TO CHANGE THE HARDWARE TABLE AND IF HE WISHES TO CHANGE THE SOFTWARE TABLE. IF THE USER ANSWERS YES (BY TYPING "Y") TO THE HARDWARE TABLE QUESTION, THAT PORTION OF THE DIALOGUE DESCRIBED IN SECTION 4.1 BEGINNING WITH QUESTION 1 ENDING WITH QUESTION 5 WILL TAKE PLACE. IF THE USER ANSWERS YES TO THE SOFTWARE TABLE QUESTION, HE WILL BE ASKED QUESTIONS 7 AND 8.

THE MOST COMMON USE OF THIS COMMAND IS TO CHANGE RUN-TIME FLAGS OR TO ADD/DROP UNITS. THESE FLAGS ARE DESCRIBED IN SECTION 4.8 - SWITCH OPTIONS. TYPING A "C" WILL FORCE THE SUPERVISOR TO ISSUE A PROMPT CHARACTER (">") SO THAT THE USER MAY USE THE RESTART COMMAND FOR THIS PURPOSE. NOTE: IF THE USER USES THE START COMMAND AFTER A C, HE WILL HAVE TO REENTER ALL DEVICE INFORMATION!

4.3

CONTINUE

THE CONTINUE COMMAND WILL RESTART THE PROGRAM AT THE TEST WHICH WAS INTERRUPTED. UNLIKE THE RESTART COMMAND, THIS COMMAND DOES NOT FORCE FLAGS BACK TO THE INITIAL STATE. THE FLAGS WILL REMAIN AS SET UP DURING THE LAST START OR RESTART. THE USER HAS THE SAME DIALOGUE AND FLAG

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

OPTIONS AS DESCRIBED FOR THE RESTART COMMAND.

- 4.4 ZERO FLAGS
THIS COMMAND WILL ZERO ALL FLAGS (SECTION 4.8) AND TYPE A PROMPT. SECTION 4.8 DESCRIBES THE EFFECTS OF CLEARING ALL FLAGS, BUT ESSENTIALLY, ALL LOOPING AND HALTS WILL BE DISABLED AND ALL ERROR INFORMATION WILL BE PRINTED IN THE EVENT OF AN ERROR.
- 4.5 ADD UNITS
WHILE TESTING MULTIPLE UNITS, THE USER MAY WISH TO DROP A PARTICULAR UNIT FROM THE TEST PROCEDURE AND, LATER ON, BEGIN TESTING THAT UNIT AGAIN. IN THIS CASE, THE USER NEED NOT STOP THE DIAGNOSTIC AND REBUILD THE PTABLE. AFTER A UNIT HAS BEEN DROPPED (SEE SECTION 4.6), THE USER MAY ADD THE UNIT BY TYPING "C" AND USING THE ADD COMMAND AS FOLLOWS:
ADD/UNI: N
WHERE N IS A DECIMAL UNIT NUMBER THAT SPECIFIES THE UNIT TO BE ADDED. UNITS ARE NUMBERED FROM 1 TO 32 ACCORDING TO THE ORDER IN WHICH THEY WERE SPECIFIED FOR THE HARDWARE TABLE. MULTIPLE UNITS MAY BE ADDED BY TYPING EACH UNIT NUMBER SEPARATED BY COLONS (I.E., ADD/UNI: 2: 4: 22) OR BY TYPING THE FIRST AND LAST UNIT NUMBERS OF A CONTIGUOUS GROUP OF UNITS SEPARATED BY A DASH (I.E., ADD/UNI: 1-10 WILL ADD THE FIRST TEN UNITS). NOTE: AFTER A START COMMAND, ALL UNITS ARE CONSIDERED ADDED. THERE IS NO NEED TO USE THE ADD COMMAND UNLESS THE UNIT WAS DROPPED.
- 4.6 DROP UNITS
TO REMOVE A UNIT FROM TESTING, THE DROP COMMAND (DRO) IS USED. TO USE THIS COMMAND, TYPE "C" AND DROP THE UNIT(S) USING THE DROP COMMAND AS FOLLOWS:
DRO/UNI: N
WHERE "N" IS A DECIMAL UNIT NUMBER WHICH SPECIFIES THE UNIT TO BE DROPPED. UNITS ARE NUMBERED FROM 1 TO 32 ACCORDING TO THE ORDER IN WHICH THEY WERE SPECIFIED FOR THE HARDWARE TABLE. MULTIPLE UNITS MAY BE DROPPED BY TYPING EACH UNIT NUMBER SEPARATED BY COLONS (I.E., DRO/UNI: 1: 3: 7) OR BY TYPING THE FIRST AND LAST UNIT NUMBERS OF A CONTIGUOUS GROUP SEPARATED BY A DASH (I.E., DRO/UNI: 1-9 CAUSES THE FIRST NINE UNITS TO BE DROPPED). AFTER THE PROGRAM IS RESTARTED, THE SPECIFIED UNIT(S) WILL NOT BE INCLUDED FOR TESTING.
- 4.7 DISPLAY PTABLE
WHEN THE DISPLAY COMMAND IS TYPED, THE ENTIRE CONTENTS OF THE PTABLE WILL BE PRINTED. UNITS THAT HAVE BEEN DROPPED WILL BE SO MARKED. THIS COMMAND IS USEFUL FOR VERIFYING THE PTABLE CONTENTS.
- 4.8 SWITCH OPTIONS
THERE ARE THREE SWITCH OPTIONS AVAILABLE FOR USE WITH THE COMMANDS: /TES:, /PAS: AND /FLA:. THE TEST SWITCH, /TES:, IS USED ONLY WITH THE START COMMAND AND THE RESTART COMMAND. IT IS USED TO SPECIFY THE INDIVIDUAL TEST(S) TO BE RUN IN CASES WHERE ALL TESTS ARE NOT TO BE RUN. FOR EXAMPLE, IF THE USER WISHED TO RUN THE LOOPBACK TEST ONLY, HE WOULD TYPE:

>STA/TES: 2

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

IF THIS SWITCH IS NOT USED, ALL TESTS WILL BE RUN AS THE DEFAULT CONDITION.

THE PASS SWITCH IS USED TO SPECIFY THE NUMBER OF PASSES TO BE RUN. TO RUN THE LOOPBACK TEST TWENTY TIMES, THE USER WOULD TYPE:

>STA/TES: 2/PAS: 20

THIS SWITCH MAY BE USED WITH THE START, RESTART AND CONTINUE COMMANDS. THE DEFAULT CONDITION IS ESSENTIALLY AN INFINITE NUMBER OF PASSES.

THE FLAG SWITCH IS USED TO SET FLAGS WHICH CONTROL PROGRAM OPERATION AND ERROR PRINTOUTS. THIS SWITCH MAY BE USED WITH THE CONTINUE COMMAND AS WELL AS THE START COMMAND AND RESTART COMMAND. THE AVAILABLE FLAGS ARE SHOWN IN THE TABLE BELOW.

FLAG	EFFECT IF SET
LOE	LOOP ON ERROR
HLT	HALT ON ERROR
IER	INHIBIT ERROR REPORTS
IBE	INHIBIT BASIC ERROR REPORT*
IXE	INHIBIT EXTENDED ERROR REPORT*
PRI	SEND ALL TYPEOUTS TO THE PRINTER
BOE	BELL ON ERROR
UAM	UNATTENDED MODE (NO MANUAL INTERVENTION)
PNT	PRINT TEST NUMBER

*SEE SECTION 5.0

EXAMPLE:

IF THE USER WISHES TO RUN THE PROGRAM WITHOUT ERROR MESSAGES AND HAVE IT LOOP ON ERROR, HE WOULD TYPE:

>RES/FLA: LOE: IER

EXAMPLE:

IF THE USER LATER WANTED TO STOP THE LOOP ON ERROR FEATURE, HE WOULD TYPE:

>CON/FLA: LOE=0

NOTE: THE ERROR REPORTS WOULD STILL BE INHIBITED.

ALL FLAGS ARE DEFAULTED TO ZERO (NOT SET).

5.0

ERROR AND INFORMATIONAL MESSAGES

AS NOTED IN THE DESIGN NOTES (SECTION 1.3), A SINGLE ERROR MAY RESULT IN SEVERAL MESSAGES BEING TYPED. THIS IS DUE TO THE STRUCTURE OF THE PROGRAM. EACH SECTION, IN THE EVENT OF AN ERROR, REPORTS THE INFORMATION AVAILABLE TO IT. THE FINAL MESSAGE FOR EACH ERROR CONTAINS THE UNIT NUMBER (DECIMAL - 1 TO 32 - SEE SECTIONS 4.5 AND 4.6) AND THE PHRASE: "END OF ERROR REPORT".

THERE ARE THREE LEVELS OF ERROR REPORTS. BY USING THE FLAG SWITCHES (SECTION 4.8), ONE OR MORE LEVELS OF REPORT MAY BE INHIBITED. THE FIRST LEVEL CONSISTS OF AN ERROR NUMBER, PC CONTENTS AND ERROR TYPE (HARD, SOFT OR FATAL). THE ERROR NUMBER CAN BE RELATED TO THE TABLE AT THE END OF

470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523

THIS SECTION. THE PC CONTENTS GIVE THE ADDRESS WHERE THE ERROR REPORT MECHANISM WAS INVOKED. THE NEXT LEVEL (BASIC ERROR REPORT) GIVES AN ENGLISH DESCRIPTION OF THE ERROR. AN EXAMPLE OF THIS TYPE OF MESSAGE IS "HEADER HAS CRC ERROR". THE FINAL LEVEL (EXTENDED ERROR REPORT) GIVES ADDITIONAL DATA CONCERNING THE ERROR, SUCH AS REGISTER CONTENTS.

TO INHIBIT SOME OR ALL ERROR MESSAGES, USE THE FLAGS DESCRIBED IN SECTION 4.8. THE INHIBIT-EXTENDED-ERROR-REPORT FLAG (IXE) INHIBITS THAT LEVEL ONLY. THE INHIBIT-BASIC-ERROR-REPORT FLAG (IBE) INHIBITS BOTH BASIC AND EXTENDED ERROR REPORTS. FINALLY, THE INHIBIT-ERROR-REPORT FLAG (IER) INHIBITS ALL TYPEOUTS.

TABLE OF ERROR NUMBERS

NO.	ERROR DESCRIPTION
00	CANNOT INITIALIZE DUP
01	REPLY MESSAGE FROM TERM. WAS NOT AN ENQ
02	REPLY FROM TERM. WAS NOT EXPECTED TYPE OF CONTROL MESSAGE
03	TERM. ADDRESS IN RESPONSE WAS NOT CORRECT
04	RESPONSE HAD CRC ERROR IN THE HEADER
20	DUP TRANSMIT ERROR
21	DUP RECEIVE ERROR
22	MESSAGE TO TERMINAL WAS NOT ACK'ED
23	ERROR OCCURED WHILE INITIATING DUP
24	ERROR OCCURED DURING TRANSMISSION
25	DDCMP STRT MESSAGE DID NOT GET A STACK
26	ERROR DURING RECEPTION
30	CANNOT GET CLEAR TO SEND SIGNAL
31	TIMEOUT WHILE WAITING FOR RESPONSE
40	SEE 21
41	SEE 04
42	SEE 03
43	MESSAGE DATA HAD CRC ERROR
50	ERROR OCCURED DURING SELFTEST (TEST #1)
51	ERROR OCCURED DURING LOOPBACK (TEST #2)
52	ERROR IN RECEIVED DATA
70	SELFTEST RESPONSE IN ERROR (NOT RECOGNIZED)
71	TERMINAL MEMORY ERROR
72	TERMINAL CONTROL-ROM ERROR

6.0 RESTRICTIONS
THIS DIAGNOSTIC DOES NOT ATTEMPT TO TEST THE DUP11. IF A LARGE NUMBER OF COMM ERRORS OCCUR, OR IF ATTEMPTS TO SEND OR RECIEVE DATA TO AND FROM THE TERMINAL FAIL; USE THE SPECIFIC DUP11 DIAGNOSTICS TO VERIFY THE DUP11.

7.0 MISCELLANEOUS
PLEASE REPORT ALL PROGRAM PROBLEMS VIA THE AIDS REPORTING SYSTEM. REFER ALL COMMENTS REGARDING DOCUMENTATION TO THE DIAGNOSTIC ENGINEERING

525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567

TERMINAL GROUP IN MAYNARD, MASS.

8.0 TEST DESCRIPTIONS

8.1 MICRODIAGNOSTIC (TEST #1)

THE MICRODIAGNOSTIC, OR SELFTEST, IS ROM RESIDENT CODE IN THE VT62 THAT IS EXECUTED UPON COMMAND FROM THE HOST COMPUTER OR DURING POWER UP. A FULL DESCRIPTION OF THE TEST IS AVAILABLE IN THE HARDWARE MANUAL FOR THE VT62. ESSENTIALLY, THE TEST VERIFIES THE MICROPROCESSOR, CHECKS ROM MEMORY, EXERCISES RAM MEMORY AND EXERCISES THE SCREEN. THE HOST MAY REQUEST THE RESULTS OF THE TEST. IN ANY CASE, THE RESULTS ARE ALWAYS DISPLAYED ON THE LAST LINE OF THE SCREEN. THE FIRST CHARACTERS ARE A CODED ERROR REPORT FOR THE HOST COMPUTER AND THE REMAINDER OF THE LINE IS AN ENGLISH DESCRIPTION OF THE ERROR STATUS.

8.2 LOOPBACK (TEST #2)

IN THIS TEST, THE TERMINAL IS COMMANDED TO ENTER LOOPBACK MODE AND A SINGLE, 256(DECIMAL) BYTE DATA MESSAGE IS SENT TO THE TERMINAL. THIS DATA MESSAGE WILL HAVE A DDCMP PROTOCOL ENVELOPE. LOOPBACK DATA IS DISPLAYED ON THE VT62'S SCREEN. WHEN THE DATA IS RETURNED, THE TERMINAL IS TAKEN OUT OF LOOPBACK MODE USING A DDCMP "STRT" SEQUENCE. THE CRC CHECKSUMS ARE VALIDATED FOR BOTH THE MESSAGE HEADER AND THE MESSAGE DATA. THE DATA IS THEN CHECKED, BYTE BY BYTE. THUS THERE ARE THREE POSSIBLE ERRORS DETECTED: NO TERMINAL RESPONSE, BAD CRC OR BAD DATA.

INTERPRETATION OF ANY ERROR MESSAGES REQUIRES SOME EDUCATED GUESSWORK BECAUSE OF THE NUMBER OF COMPONENTS IN THE COMMUNICATION LINK. INTERMITTENT ERRORS WOULD GENERALLY INDICATE LINE PROBLEMS UNLESS THE LINE IS PHYSICALLY SHORT OR THE ENVIRONMENT IS RELATIVELY NOISE FREE. IF EITHER OF THESE CONDITIONS IS THE CASE OR IF THE ERROR IS NOT INTERMITTENT, THEN THE TERMINAL I/O BOARD, THE MODEM AND THE DUP ARE SUSPECT. IF THERE IS NO TERMINAL RESPONSE, RUN THE TERMINAL SELF TEST OFF-LINE. THIS WILL SHOW WHETHER THE TERMINAL IS COMPLETELY "DEAD" OR THE PORTIONS OF THE COMMUNICATION LINK MENTIONED ABOVE ARE BROKEN.

THERE ARE TWO DATA PATTERNS AVAILABLE FOR LOOPBACK: ALL ZERO'S AND ALTERNATING 1/0 BITS (I.E., BYTES OF 252,125,252,125,...). THE ZERO PATTERN IS NOISE-SUSCEPTIBLE, BUT APPEARS ON THE SCREEN AS NULLS. HOWEVER THE ALTERNATING BITS PATTERN EXERCISES THE DATA PATHS BETTER. SECTION 4.0 SPECIFIES HOW TO SELECT THE PATTERN.

569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609

9.0

INDEX

ACK	5, 11
ADD COMMAND	9
CONTINUE COMMAND	8, 10
CRC	4-5, 11
DDCMP	4-5
DISPLAY COMMAND	9
DROP COMMAND	9
ENQ	5, 11
ERROR NUMBER	10-11
ERROR REPORT	10-11
FLAGS	6, 8-10
LOOPBACK	7, 11-12
MULTIPLE UNITS	9
MULTIPLE UUT'S	6
NAK	5
PTABLE	4, 6, 9
RESTART COMMAND	8-10
STACK	5, 11
START COMMAND	6, 8-10
STRT	5, 11
SWITCH OPTIONS	8-9
SWITCHES	6
UNIT NUMBER	9-10

ENDR

5712	002000	. =2000
5713		
5714		HEADER - VT62 ACCEPTANCE TEST
5715		
5725	002000	. ASCII @D@
(5)	002001	. ASCII @Z@
(5)	002002	. ASCII @D@
(5)	002003	. ASCII @P@
(5)	002004	. ASCII @G@
(6)	002005	. BYTE 0
(6)	002006	. BYTE 0
(5)	002007	. BYTE 0
(4)	002010	. ASCII @A@
(4)	002011	. ASCII @O@
(4)	002012	. BYTE C\$REVISION
(3)	002013	. BYTE C\$EDIT
(4)	002014	. WORD 0
(4)	002016	. WORD 0
(4)	002020	. WORD 0
(4)	002022	. WORD 0
(4)	002024	. WORD 0
(5)	002026	. WORD 0
(4)	002030	. WORD 0
(4)	002032	. WORD 0
(4)	002034	. WORD 0
(4)	002036	. WORD 0
(4)	002040	. WORD L\$DISPATCH
(4)	002042	. WORD L\$INIT
(4)	002044	. WORD L\$CLEAN
(4)	002046	. WORD L\$HARD
(4)	002050	. WORD L\$SOFT
(4)	002052	. WORD L\$DVTYP
(4)	002054	. WORD L\$RPT
(4)	002056	. WORD L\$HW
(4)	002060	. WORD L\$SW
(4)	002062	. WORD L\$DR
(4)	002064	. WORD L\$DRST
(4)	002066	. WORD 0
(4)	002070	. WORD L\$AU
(4)	002072	. WORD L\$DU
(4)	002074	. WORD 0
(4)	002076	. WORD L\$LAST

```
5745 ;*****
5746 ; GLOBAL EQUATES *
5747 ;*****
5748 ;
(1) ; BIT DIFINITIONS
(1) ;
(1) 100000 BIT15== 100000
(1) 040000 BIT14== 40000
(1) 020000 BIT13== 20000
(1) 010000 BIT12== 10000
(1) 004000 BIT11== 4000
(1) 002000 BIT10== 2000
(1) 001000 BIT09== 1000
(1) 000400 BIT08== 400
(1) 000200 BIT07== 200
(1) 000100 BIT06== 100
(1) 000040 BIT05== 40
(1) 000020 BIT04== 20
(1) 000010 BIT03== 10
(1) 000004 BIT02== 4
(1) 000002 BIT01== 2
(1) 000001 BIT00== 1
(1) ;
(1) 001000 BIT9== BIT09
(1) 000400 BIT8== BIT08
(1) 000200 BIT7== BIT07
(1) 000100 BIT6== BIT06
(1) 000040 BIT5== BIT05
(1) 000020 BIT4== BIT04
(1) 000010 BIT3== BIT03
(1) 000004 BIT2== BIT02
(1) 000002 BIT1== BIT01
(1) 000001 BIT0== BIT00
(1) ;
(1) ; EVENT FLAG DEFINITIONS
(1) ; EF32: EF17 RESERVED FOR SUPERVISOR TO PROGRAM COMMUNICATION
(1) ; EF16: EF01 AVAILABLE FOR PROGRAM USE
(1) ;
(1) 000040 EF. START== 32. ; START COMMAND WAS ISSUED
(1) 000037 EF. RESTART== 31. ; RESTART COMMAND WAS ISSUED
(1) 000036 EF. CONTINUE== 30. ; CONTINUE COMMAND WAS ISSUED
(1) 000035 EF. NEW== 29. ; A NEW PASS HAS BEEN STARTED
(1) 000034 EF. PWR== 28. ; A POWER-FAIL/POWER-UP OCCURRED
(1) ;
(1) 000020 EF16== 16.
(1) 000017 EF15== 15.
(1) 000016 EF14== 14.
(1) 000015 EF13== 13.
(1) 000014 EF12== 12.
(1) 000013 EF11== 11.
(1) 000012 EF10== 10.
(1) 000011 EF09== 9.
(1) 000010 EF08== 8.
(1) 000007 EF07== 7.
(1) 000006 EF06== 6.
(1) 000005 EF05== 5.
```



```
(1) 000004 EFO4== 4
(1) 000003 EFO3== 3
(1) 000002 EFO2== 2
(1) 000001 EFO1== 1
(1) ;
(1) ; PRIORITY LEVEL DEFINITIONS
(1) ;
(1) 000340 PRI07== 340
(1) 000300 PRI06== 300
(1) 000240 PRI05== 240
(1) 000200 PRI04== 200
(1) 000140 PRI03== 140
(1) 000100 PRI02== 100
(1) 000040 PRI01== 40
(1) 000000 PRI00== 0
5749 ;
5750 ;
5751 ; VT62 DIAGNOSTIC EQUATES
5752 ;
5753 000201 SOH=201 ; START OF HEADER (DDCMP)
5754 000001 ACK=1 ; ACKNOWLEDGE "
5755 000002 NAK=2 ; NEGATIVE ACKNOWLEDGE "
5756 000003 REP=3 ; REPLY "
5757 000005 ENQ=5 ; ENQUIRY "
5758 000006 STRT=6 ; START "
5759 000007 STACK=7 ; START ACKNOWLEDGE "
5761 000226 SYNC=226 ; SYNCHRONIZATION CHARACTER
5766 000220 DLE=220 ; DATA LINK ESCAPE.
5767 000024 OSOP=24 ; LOOPBACK.
5768 000200 PARITY=200 ; PARITY BIT.
5776 000002 TSTS=2 ; NO. OF TESTS
5782 000000 RCVCSR=0 ; OFFSETS: RECEIVER CSR
5783 000002 RCVBUF=2 ; RECEIVER BUFFER REG.
5784 000004 XMTCSR=4 ; TRANSMITTER CSR
5785 000006 XMTBUF=6 ; TRANSMITTER BUFFER REG.
5787 ;
5789 ; HARDWARE REGISTER BIT DEFINITIONS
5790 000400 TSOM=400 ; TRANSMIT, START OF MESSAGE
5791 000002 DTR=2 ; DATA TERMINAL READY
5792 000400 MASCLR=400 ; MASTER CLEAR
5793 101226 PARAM=101226 ; PARAM. REG. - DEC MODE, AUTO-CRC INHIB. AND SYNC
5794 000004 RTS=4 ; REQUEST TO SEND
5795 020000 CTS=20000 ; CLEAR TO SEND
5796 000100 TXINTE=100 ; TRANSMIT INTERRUPT ENABLE
5797 000020 SENDFD=20 ; SEND -- FULL DUX
5798 000120 RXINTE=120 ; RECEIVE INTERRUPT ENABLE
5799 000020 RCENBL=20 ; RECEIVER ENABLE
5865 177400 HYBYTE=177400 ; MASK FOR HIGH BYTE OF A WORD
5866 000377 LOBYTE=377 ; MASK FOR LOW BYTE OF A WORD
```

```
5887 ;*****  
5888 ; GLOBAL DATA *  
5889 ;*****  
5890 002100 000000 R5STCK: 0 ; FLAG TO INDICATE IF R5 STACK IS SET UP  
5891 002102 000000 SCRTCH: 0 ; SCRATCH LOCATIONS  
5892 002104 000000 TEMP: 0  
5893 002106 000000 ABORT: 0 ; TEST ABORT FLAG  
5894 002110 020000 TIMLIM: 20000 ; TIMEOUT LIMIT (IN 100 MICROSEC. INCREMENTS)  
5895 002112 000000 STIME: 0 ; SHORT TIMEOUT LIMIT (APPROX. 20 MICROSEC. INCREM.)  
5896 002114 000010 ILIMIT: 10 ; LIMIT ON DEVICE INITIALIZATION ATTEMPTS  
5897 002116 000000 INERR: 0 ; INPUT ERROR  
5898 002120 000000 OUTERR: 0 ; OUTPUT ERROR  
5899 002122 000001 PASCNT: 1  
5900 002124 000000 EOMSG: 0 ; END_OF_MESSAGES FLAG  
5901 002126 000000 LOGDEV: 0 ; LOGICAL DEVICE NUMBER  
5905 002130 000000 EXPINP: 0 ; EXPECT_INPUT FLAG  
5906 002132 000000 OUTMSG: 0 ; OUTPUT MESSAGE POINTER AND SIZE  
5907 002134 000000 0  
5908 002136 000000 INMSG: 0 ; INPUT MESSAGE POINTER AND SIZE  
5909 002140 000000 0  
5910 002142 000000 OUTBUF: 0 ; OUTPUT BUFFER POINTER AND SIZE  
5911 002144 000000 OUTSIZ: 0  
5912 002146 000000 INBUF: 0 ; INPUT BUFFER POINTER AND SIZE  
5913 002150 000000 INSIZ: 0  
5914  
5915 002152 MSGAP:  
5916 002152 000000 OUT: 0 ; TEST MESSAGE AREA POINTER  
5917 002154 000000 INITMA: 0 ; TEMPORARY MESSAGE POINTER AND SIZE  
5918 002156 000000 INITMS: 0 ; USED FOR INITIALIZATION OF OUTPUT ROUTINES  
5919 002160 000000 BAPNT: 0 ; GENERAL BUFFER AREA POINTER AND SIZE  
5920 002162 000000 BASIZE: 0  
5921 002164 000000 MSGNO: 0 ; MESSAGE NUMBER (DDCMP)  
5922 002166 000000 RMSGNO: 0  
5929 002170 000000 PTABLE: 0 ; PRESENT UNIT PTABLE DATA  
5930 002172 000000 0  
5931 002174 000000 0  
5932 002176 000000 0  
5936 002200 PTEND:  
5937 002200 000000 DEVPRI: 0 ; DEVICE PRIORITY  
5949  
5951 -002202 052126 031066 042054 .ASCIZ @VT62,DUP11@  
(3) 002210 050125 030461 000  
(2) 002216  
5952 002216 000001 .EVEN  
(6) 002220 000017 .WORD 1  
(4) 002222 000004 .WORD 17  
(2) 002224 000004 .WORD 4  
.BLKW 4
```

```
5963 ;*****  
5964 ; GLOBAL TEXT *  
5965 ;*****  
5966  
5967 002234 040503 020116 047516 MSG00: .ASCIZ /CAN NOT INITIALIZE COMM DEVICE/  
002242 020124 047111 052111  
002250 040511 044514 042532  
002256 041440 046517 020115  
002264 042504 044526 042503  
002272 000  
5968 002273 122 050105 054514 MSG01: .ASCIZ /REPLY WAS NOT AN ENQ/  
002300 053440 051501 047040  
002306 052117 040440 020116  
002314 047105 000121  
5969 002320 047111 047503 051122 MSG02: .ASCIZ /INCORRECT REPLY TYPE/  
002326 041505 020124 042522  
002334 046120 020131 054524  
002342 042520 000  
5970 002345 MSG42:  
5971 002345 122 050105 054514 MSG03: .ASCIZ /REPLY HAS INCORRECT STATION ADDRESS/  
002352 044040 051501 044440  
002360 041516 051117 042522  
002366 052103 051440 040524  
002374 044524 047117 040440  
002402 042104 042522 051523  
002410 000  
5972 002411 MSG41:  
5973 002411 110 040505 042504 MSG04: .ASCIZ /HEADER HAS CRC ERROR/  
002416 020122 040510 020123  
002424 051103 020103 051105  
002432 047522 000122  
5974 002436 051105 047522 020122 MSG10: .ASCIZ /ERROR DURING KEYBOARD TEST ** END OF REPORT **/  
002444 052504 044522 043516  
002452 045440 054505 047502  
002460 051101 020104 042524  
002466 052123 020040 025052  
002474 042440 042116 047440  
002502 020106 042522 047520  
002510 052122 025040 000052  
5975 002516 040502 020104 042522 MSG11: .ASCIZ /BAD RESPONSE FROM KEYBOARD/  
002524 050123 047117 042523  
002532 043040 047522 020115  
002540 042513 041131 040517  
002546 042122 000  
5976 002551 124 047517 046440 MSG12: .ASCIZ /TOO MANY CHARACTERS SENT FROM KEYBOARD/  
002556 047101 020131 044103  
002564 051101 041501 042524  
002572 051522 051440 047105  
002600 020124 051106 046517  
002606 045440 054505 047502  
002614 051101 000104  
5977 002620 051124 047101 046523 MSG20: .ASCIZ /TRANSMIT ERROR/  
002626 052111 042440 051122  
002634 051117 000  
5978 002637 MSG40:  
5979 002637 122 041505 050105 MSG21: .ASCIZ /RECEPTION ERROR/
```

	002644	044524	047117	042440	
	002652	051122	051117	000	
5980	002657	116	020117	041501	MSG22: .ASCIZ /NO ACK RECEIVED AFTER TRANSMISSION/
	002664	020113	042522	042503	
	002672	053111	042105	040440	
	002700	052106	051105	052040	
	002706	040522	051516	044515	
	002714	051523	047511	000116	
5981	002722	051105	047522	020122	MSG23: .ASCIZ /ERROR WHILE INITIALIZING DEVICE/
	002730	044127	046111	020105	
	002736	047111	052111	040511	
	002744	044514	044532	043516	
	002752	042040	053105	041511	
	002760	000105			
5982	002762	051105	047522	020122	MSG24: .ASCIZ /ERROR OCCURRED DURING TRANSMISSION/
	002770	041517	052503	051122	
	002776	042105	042040	051125	
	003004	047111	020107	051124	
	003012	047101	046523	051511	
	003020	044523	047117	000	
5983	003025	116	020117	052123	MSG25: .ASCIZ /NO STACK IN RESPONSE TO STRT/
	003032	041501	020113	047111	
	003040	051040	051505	047520	
	003046	051516	020105	047524	
	003054	051440	051124	000124	
5984	003062	051105	047522	020122	MSG26: .ASCIZ /ERROR WHILE RECEIVING BUFFER/
	003070	044127	046111	020105	
	003076	042522	042503	053111	
	003104	047111	020107	052502	
	003112	043106	051105	000	
5985	003117	103	047101	052047	MSG30: .ASCIZ /CAN'T GET CLEAR TO SEND/
	003124	043440	052105	041440	
	003132	042514	051101	052040	
	003140	020117	042523	042116	
	003146	000			
5986	003147	124	046511	020105	MSG31: .ASCIZ /TIME OUT WHILE WAITING FOR TERMINAL RESPONSE/
	003154	052517	020124	044127	
	003162	046111	020105	040527	
	003170	052111	047111	020107	
	003176	047506	020122	042524	
	003204	046522	047111	046101	
	003212	051040	051505	047520	
	003220	051516	000105		
5987	003224	040504	040524	041440	MSG43: .ASCIZ /DATA CRC ERROR/
	003232	041522	042440	051122	
	003240	051117	000		
5988	003243	116	044517	042523	MSG45: .ASCIZ /NOISE PATTERN SUBTEST/
	003250	050040	052101	042524	
	003256	047122	051440	041125	
	003264	042524	052123	000	
5989	003271	105	051122	051117	MSG50: .ASCIZ /ERROR DURING MICRODIAGNOSTIC/
	003276	042040	051125	047111	
	003304	020107	044515	051103	
	003312	042117	040511	047107	
	003320	051517	044524	000103	
5990	003326	051105	047522	020122	MSG51: .ASCIZ /ERROR DURING LOOPBACK/

003334	052504	044522	043516		
003342	046040	047517	041120		
003350	041501	000113			
5991	003354	040504	040524	042440	MSG52: .ASCIZ /DATA ERROR/
	003362	051122	051117	000	
5992	003367	111	046114	043505	MSG70: .ASCIZ /ILLEGAL SELF TEST RESPONSE/
	003374	046101	051440	046105	
	003402	020106	042524	052123	
	003410	051040	051505	047520	
	003416	051516	000105		
5993	003422	042515	047515	054522	MSG71: .ASCII /MEMORY ERROR TYPE /
	003430	042440	051122	051117	
	003436	020040	052040	050131	
	003444	020105			
5994	003446	000	000		MTY: .BYTE 0,0
5995	003450	020115	020055	040502	.ASCII /M - BANK #/
	003456	045516	021440		
5996	003462	000	000		BNO: .BYTE 0,0
5997					.EVEN
5998	003464	051103	046517	042440	MSG72: .ASCIZ /CROM ERROR/
	003472	051122	051117	000	
5999		003500			.EVEN
6000					
6001	003500	005			STRTB: .BYTE ENQ
6002	003501	006			.BYTE STRT
6003	003502	300			.BYTE 300
6004	003503	000			.BYTE 0
6005	003504	000			.BYTE 0
6006	003505	000			STRTSA: .BYTE 0
6007	003506	000000			STRTC: .WORD 0
6008					
6009					:*****
6010					: GLOBAL ERROR REPORT *
6011					:*****
6013	003510	016746	176400		MOV !LIMIT, -(SP)
(7)	003514	012746	003536		MOV #FORM00, -(SP)
(6)	003520	012746	000002		MOV #2, -(SP)
(3)	003524	010600			MOV SP, R0
(4)	003526	104015			EMT C\$PNTX
(4)	003530	062706	000006		ADD #6, SP
6014	003534				L10000: EMT C\$MSG
(3)	003534	104023			FORM00: .ASCIZ /%AFTER %06%A ATTEMPTS%N/
6015	003536	040445	043101	042524	
	003544	020122	047445	022466	
	003552	020101	052101	042524	
	003560	050115	051524	047045	
	003566	000			
6016		003570			.EVEN
6017					
6019	003570				L10001: EMT C\$MSG
(3)	003570	104023			
6020					
6022	003572	016746	000030		MOV A02B, -(SP)
(8)	003576	016746	000022		MOV A02A, -(SP)
(7)	003602	012746	003630		MOV #FORM02, -(SP)
(6)	003606	012746	000003		MOV #3, -(SP)

(3)	003612	010600			MOV	SP,RO
(4)	003614	104015			EMT	C\$PNTX
(4)	003616	062706	000010		ADD	#10,SP
6023	003622			L10002:		
(3)	003622	104023			EMT	C\$MSG
6024	003624	000000		AO2A:	0	
6025	003626	000000		AO2B:	0	
6026	003630	040445	042522	042503	FORM02:	ASCIZ /%RECEIVED %03%A -- EXPECTED %03%N/
	003636	053111	042105	022440		
	003644	031517	040445	026440		
	003652	020055	054105	042520		
	003660	052103	042105	022440		
	003666	031517	047045	000		
6027		003674				EVEN
6028						
6030	003674			AREA42:		
6031	003674	016746	176202		MOV	SCRATCH, -(SP)
(8)	003700	016746	176200		MOV	TEMP, -(SP)
(7)	003704	012746	003630		MOV	#FORM02, -(SP)
(6)	003710	012746	000003		MOV	#3, -(SP)
(3)	003714	010600			MOV	SP,RO
(4)	003716	104015			EMT	C\$PNTX
(4)	003720	062706	000010		ADD	#10,SP
6032	003724			L10003:		
(3)	003724	104023			EMT	C\$MSG
6033						
6035	003726			AREA41:		
6036	003726			AREA43:		
6037	003726	016746	000030		MOV	AO4EX, -(SP)
(8)	003732	016746	000022		MOV	AO4RC, -(SP)
(7)	003736	012746	003764		MOV	#FORM04, -(SP)
(6)	003742	012746	000003		MOV	#3, -(SP)
(3)	003746	010600			MOV	SP,RO
(4)	003750	104015			EMT	C\$PNTX
(4)	003752	062706	000010		ADD	#10,SP
6038	003756			L10004:		
(3)	003756	104023			EMT	C\$MSG
6039	003760	000000		AO4RC:	0	
6040	003762	000000		AO4EX:	0	
6041	003764	040445	042522	042503	FORM04:	ASCIZ /%RECEIVED %06%A -- EXPECTED %06%N/
	003772	053111	042105	022440		
	004000	033117	040445	026440		
	004006	020055	054105	042520		
	004014	052103	042105	022440		
	004022	033117	047045	000		
6042		004030				EVEN
6043						
6045	004030	016746	000030		MOV	KEX, -(SP)
(8)	004034	016746	000022		MOV	KRC, -(SP)
(7)	004040	012746	003764		MOV	#FORM04, -(SP)
(6)	004044	012746	000003		MOV	#3, -(SP)
(3)	004050	010600			MOV	SP,RO
(4)	004052	104015			EMT	C\$PNTX
(4)	004054	062706	000010		ADD	#10,SP
6046	004060			L10005:		
(3)	004060	104023			EMT	C\$MSG

6047	004062	000000			KRC:	0			
6048	004064	000000			KEX:	0			
6050	004066				L10006:				
(3)	004066	104023				EMT	C\$MSG		
6052	004070				L10007:				
(3)	004070	104023				EMT	C\$MSG		
6054	004072				AREA24:				
6055	004072				AREA40:				
6056	004072	016767	176030	000064	MOV		LOGDEV, TEMP2		
6057	004100	005767	000060		TST		TEMP2		
(9)	004104	002002			BGE		50000\$		
6058	004106	005067	000052		CLR		TEMP2		
6059	004112				50000\$:				
6060	004112	005267	000046		INC		TEMP2		
6061	004116	016746	000042		MOV		TEMP2, -(SP)		
(7)	004122	012746	004166		MOV		#FORM23, -(SP)		
(6)	004126	012746	000002		MOV		#2, -(SP)		
(3)	004132	010600			MOV		SP, RO		
(4)	004134	104015			EMT		C\$PNTX		
(4)	004136	062706	000006		ADD		#6, SP		
6062	004142	012746	004206		MOV		#FORM40, -(SP)		
(6)	004146	012746	000001		MOV		#1, -(SP)		
(3)	004152	010600			MOV		SP, RO		
(4)	004154	104015			EMT		C\$PNTX		
(4)	004156	062706	000004		ADD		#4, SP		
6063	004162				L10010:				
(3)	004162	104023				EMT	C\$MSG		
6064	004164	000000			TEMP2:	0			
6065	004166	040445	042504	044526	FORM23:	ASCIZ	/%ADEVICE: %D3%N/		
	004174	042503	020072	042045					
	004202	022463	000116						
6066	004206	040445	020040	020040	FORM40:	ASCIZ	/%A ***** END OF ERROR REPORT *****%N/		
	004214	020040	020040	025040					
	004222	025052	025052	042440					
	004230	042116	047440	020106					
	004236	051105	047522	020122					
	004244	042522	047520	052122					
	004252	025040	025052	025052					
	004260	047045	000						
6067		004264							
6068									
6070	004264				L10011:				
(3)	004264	104023				EMT	C\$MSG		
6071									
6074	004266				AREA31:				
6075	004266	010146			MOV		R1, -(SP)		
6076	004270	012701	002224		MOV		#DUPREG, R1		
6077	004274	016146	000000		MOV		RCVCSR(R1), -(SP)		
(7)	004300	012746	004420		MOV		#FORM30, -(SP)		
(6)	004304	012746	000002		MOV		#2, -(SP)		
(3)	004310	010600			MOV		SP, RO		
(4)	004312	104015			EMT		C\$PNTX		
(4)	004314	062706	000006		ADD		#6, SP		
6078	004320	016146	000002		MOV		RCVBUF(R1), -(SP)		
(7)	004324	012746	004442		MOV		#FRM30A, -(SP)		
(6)	004330	012746	000002		MOV		#2, -(SP)		

(3)	004334	010600				MOV	SP, R0
(4)	004336	104015				EMT	CSPNTX
(4)	004340	062706	000006			ADD	#6, SP
6079	004344	016146	000004			MOV	XMTCSR(R1), -(SP)
(7)	004350	012746	004465			MOV	#FRM30B, -(SP)
(6)	004354	012746	000002			MOV	#2, -(SP)
(3)	004360	010600				MOV	SP, R0
(4)	004362	104015				EMT	CSPNTX
(4)	004364	062706	000006			ADD	#6, SP
6080	004370	016146	000006			MOV	XMTBUF(R1), -(SP)
(7)	004374	012746	004507			MOV	#FRM30C, -(SP)
(6)	004400	012746	000002			MOV	#2, -(SP)
(3)	004404	010600				MOV	SP, R0
(4)	004406	104015				EMT	CSPNTX
(4)	004410	062706	000006			ADD	#6, SP
6081	004414	012601				MOV	(SP)+, R1
6082	004416				L10012:		
(3)	004416	104023				EMT	CMSG
6083	004420	040445	052504	020120	FORM30:	.ASCIZ	/%ADUP RCSR: %06%N/
	004426	041522	051123	020072			
	004434	047445	022466	000116			
6084	004442	040445	052504	020120	FRM30A:	.ASCIZ	/%ADUP RBUFF: %06%N/
	004450	041122	043125	035106			
	004456	022440	033117	047045			
	004464	000					
6085	004465	045	042101	050125	FRM30B:	.ASCIZ	/%ADUP TCSR: %06%N/
	004472	052040	051503	035122			
	004500	022440	033117	047045			
	004506	000					
6086	004507	045	042101	050125	FRM30C:	.ASCIZ	/%ADUP TBUFF: %06%N/
	004514	052040	052502	043106			
	004522	020072	047445	022466			
	004530	000116					
6087						.EVEN	
6127							
6129	004532	012746	004554			MOV	#FORM60, -(SP)
(6)	004536	012746	000001			MOV	#1, -(SP)
(3)	004542	010600				MOV	SP, R0
(4)	004544	104015				EMT	CSPNTX
(4)	004546	062706	000004			ADD	#4, SP
6130	004552				L10013:		
(3)	004552	104023				EMT	CMSG
6131	004554	040445	040503	023516	FORM60:	.ASCIZ	/%ACAN'T GET INTO LOOPBACK%N/
	004562	020124	042507	020124			
	004570	047111	047524	046040			
	004576	047517	041120	041501			
	004604	022513	000116				
6132						.EVEN	
6134	004610	012746	004702			MOV	#FORM61, -(SP)
(6)	004614	012746	000001			MOV	#1, -(SP)
(3)	004620	010600				MOV	SP, R0
(4)	004622	104015				EMT	CSPNTX
(4)	004624	062706	000004			ADD	#4, SP
6135	004630	016767	175272	177326		MOV	LOGDEV, TEMP2
6136	004636	005767	177322			TST	TEMP2
(9)	004642	002002				BGE	500015

6137	004644	005067	177314		CLR	TEMP2
6138	004650			500015:		
6139	004650	005267	177310		INC	TEMP2
6140	004654	016746	177304		MOV	TEMP2, -(SP)
(7)	004660	012746	004166		MOV	#FORM23, -(SP)
(6)	004664	012746	000002		MOV	#2, -(SP)
(3)	004670	010600			MOV	SP, RO
(4)	004672	104015			EMT	CSPNTX
(4)	004674	062706	000006		ADD	#6, SP
6141	004700			L10014:		
(3)	004700	104023			EMT	C\$MSG
6142	004702	040445	040503	023516	FORM61:	.ASCIZ /%ACAN'T GET OUT OF LOOPBACK%N/
	004710	020124	042507	020124		
	004716	052517	020124	043117		
	004724	046040	047517	041120		
	004732	041501	022513	000116		
6143					.EVEN	
6145	004740	016746	000074		MOV	A52B, -(SP)
(8)	004744	016746	000066		MOV	A52A, -(SP)
(7)	004750	012746	004772		MOV	#FORM52, -(SP)
(6)	004754	012746	000003		MOV	#3, -(SP)
(3)	004760	010600			MOV	SP, RO
(4)	004762	104015			EMT	CSPNTX
(4)	004764	062706	000010		ADD	#10, SP
6146	004770			L10015:		
(3)	004770	104023			EMT	C\$MSG
6147	004772	040445	054105	042520	FORM52:	.ASCIZ /%AEXPECTED: %03%A -RECEIVED: %03%N/
	005000	052103	042105	020072		
	005006	047445	022463	020101		
	005014	026440	042522	042503		
	005022	053111	042105	020072		
	005030	047445	022463	000116		
6148					.EVEN	
6149	005036	000000		A52A:	0	
6150	005040	000000		A52B:	0	
6151						
6153	005042			L10016:		
(3)	005042	104023			EMT	C\$MSG
6154						
6156	005044			L10017:		
(3)	005044	104023			EMT	C\$MSG
6157						
6158	005046			TEMBUF:		
6159	005046	001452			.BLKW 810.	
6160	010172			TMPBUF:		
6161	010172	000416			.BLKW 270.	
6162	011226	000100			.BLKW 100	
6163	011426			SSTACK:		.STACK FOR SUBROUTINE LINKAGE.
6164						

```
6173 ;*****  
6174 ;      DEFAULT HARDWARE P-TABLE AND SOFTWARE P-TABLE      *  
6175 ;*****  
6176 011426 000004      .WORD  L10020-L5HW/2  
6177 011430 160010      160010  
6178 011432 000320      320  
6179 011434 000140      140  
6180 011436 000000      0  
6184 011440 L10020:  
6185 011440 000002      .WORD  L10021-L5SW/2  
6186 011442 000010      000010  
6187 011444 000001      PATTERN: 000001  
6191 011446 L10021:  
6192 ;*****  
6193 ;      DISPATCH TABLE      *  
6194 ;*****  
6195      .WORD  TSTS  
6196 011446 000002      .WORD  T1  
(6) 011450 017174      .WORD  T2  
(6) 011452 020200
```

```
6387
6389 ;*****
6390 ;STUB AREAS FOR SUPERVISOR COMPATIBILITY *
6391 ;*****
6393 011454 L10022: EMT CSRPT
(3) 011454 104025
6394
6396 011456 L10023: EMT CSAU
(3) 011456 104054
6397
6399 011460 L10024: EMT CSDU
(3) 011460 104055
6401
6402 ;*****
6403 ; INITIALIZE CODING *
6404 ;*****
6406
6407 ;SAVE R1
6408
6409 011462 010146 MOV R1, -(SP)
6410
6411 ;CHECK START EVENT FLAG TO SEE IF WE GOT HERE BECAUSE OF A START
6412
6413 011464 012700 000040 MOV #EF.START, R0
(3) 011470 104050 EMT CSREFG
6414
6415 ; IF A START, THEN...
6416
6417 011472 103005 BCC 50002$
6445
6446 ; INITIALIZE THE SUBROUTINE LINKAGE STACK
6447
6448 011474 012705 011426 MOV #SSTACK, R5
6449 011500 005267 170374 INC R5STCK
6450 011504 000414 BR 50003$
(3) 011506 50002$:
6451
6452 ; IF NOT START THEN...
6453
6454 ;CHECK FOR RESTART AND IF IT IS:
6455 ;UPDATE LOGICAL DEVICE AND STAT. TABLE ENTRY POINTER,
6456 ;BUT DON'T EXCEED THE NUMBER OF UNITS SPECIFIED
6457
6458 011506 012700 000037 MOV #EF.RESTART, R0
(3) 011512 104050 EMT CSREFG
6459 011514 103410 BCS 50004$
6460 011516 005267 170404 INC LOGDEV
6464 011522 026767 170400 170264 CMP LOGDEV, L$UNIT
(9) 011530 001002 BNE 50005$
6465 011532 005067 170370 CLR LOGDEV
6469 011536 50005$:
6470 011536 50004$:
6471 011536 50003$:
6472
6473 ;CHECK FOR NEW PASS... IF IT IS, FORCE LOGICAL UNIT TO ZERO.
6474
```

6475	011536	012700	000035		MOV	#EF, NEW, RD
(3)	011542	104050			EMT	CSREFG
6476	011544	103002			BCC	50006\$
6477	011546	005067	170354		CLR	LOGDEV
6481	011552				50006\$:	
6482					;	
6483					;	END OF START CODE
6484					;	FORCE THE COMM DEVICE TO BE INITIALIZED
6485					;	
6486	011552	005067	003640		CLR	INIT
6487	011556	005067	170324		CLR	ABORT
6488					;	
6489					;	SET UP THE INPUT BUFFERS QUEUE
6490					;	
6491	011562	012767	015442	003650	MOV	#RQUET, RQUE
6492	011570	016767	003644	003640	MOV	RQUE, RQUEUE
6493	011576	012767	015460	003652	MOV	#RBQUET, RBQUE
6494	011604	016701	003646		MOV	RBQUE, R1
6495	011610	012767	005046	170264	MOV	#TEMBUF, SCRTCH
6496	011616				50007\$:	
6497	011616	016721	170260		MOV	SCRTCH, (R1)+
6498	011622	062767	000416	170252	ADD	#270, SCRTCH
6499	011630	020127	015464		CMP	R1, #RBQUEB
(6)	011634	003770			BLE	50007\$
6516					;	
6517					;	GET THE PTABLE ENTRY FOR THIS LOGICAL DEVICE
6518					;	
6519	011636				50010\$:	
6520	011636	005201			INC	R1
6521	011640	016700	170262		MOV	LOGDEV, RD
(3)	011644	104042			EMT	CSGPHRD
(3)	011646	010067	170230		MOV	RD, SCRTCH
6522					;	
6523					;	IF UNIT HAS BEEN DROPPED, UPDATE POINTERS ETC. AND TRY AGAIN
6524					;	
6525	011652	103411			BCS	50011\$
6526	011654	005267	170246		INC	LOGDEV
6530	011660	026767	170242	170126	CMP	LOGDEV, L\$UNIT
(9)	011666	001002			BNE	50012\$
6531	011670	005067	170232		CLR	LOGDEV
6535	011674				50012\$:	
6536	011674	005001			CLR	R1
6537	011676				50011\$:	
6538	011676	005701			TST	R1
(6)	011700	001756			BEQ	50010\$
6551					;	
6552					;	NOW GET THE PTABLE DATA FROM THE SUPERVISOR AND STORE WITHIN THE
6553					;	DIAGNOSTIC PROGRAM
6554					;	
6555	011702	012701	002170		MOV	#PTABLE, R1
6556	011706				50013\$:	
6557	011706	017711	170170		MOV	@SCRTCH, (R1)
6558	011712	062701	000002		ADD	#2, R1
6559	011716	062767	000002	170156	ADD	#2, SCRTCH
6560	011724	020127	002200		CMP	R1, #PTEND
(6)	011730	001366			BNE	50013\$

```
6579 ;  
6580 ;RESTORE R1 AND EXIT INIT CODE  
6581 ;  
6582 011732 012601          MOV      (SP)+,R1  
6583 011734          L10025:  
   (3) 011734 104011          EMT      C$INIT  
6588 011736 047045 040445 052126 VMSG: .ASCIZ /%N%AVT62 ACCEPTANCE TEST  --  VERSION 10 TERMINALS%N/  
   011744 031066 040440 041503  
   011752 050105 040524 041516  
   011760 020105 042524 052123  
   011766 020040 026440 020055  
   011774 020040 042526 051522  
   012002 047511 020116 030061  
   012010 052040 051105 044515  
   012016 040516 051514 047045  
   012024      000  
6589      012026          .EVEN  
6590  
6591 ;*****  
6592 ;      CLEAN-UP CODING      *  
6593 ;*****  
6595 ;  
6596 ;MAKE SURE DEVICE IS SHUT DOWN  
6597 ;  
6599 012026 010146          MOV      R1, -(SP)  
6600 012030 016701 170134          MOV      PTABLE, R1  
6601 012034 052761 000400 000004          BIS      #MASCLR, XMTCSR(R1)  
6602 012042 012601          MOV      (SP)+, R1  
6611 ;  
6612 ;UPDATE PASS COUNT  
6613 ;  
6614 012044 005267 170052          INC      PASCNT  
6629 ;  
6630 ;EXIT CLEAN-UP CODE  
6631 ;  
6632 012050          L10026:  
   (3) 012050 104012          EMT      C$CLEAN
```

```

6636 ;*****
6637 ; GLOBAL SUBROUTINES *
6638 ;*****
6639 ;
6640 ;+++++
6641 ; ROUTINE TO PROCESS TERMINAL MESSAGES FOR ALL TESTS +
6642 ; RETURNS POINTER TO MESSAGE ("OUTMSG") AND ITS SIZE +
6643 ; IN BYTES ("OUTSIZ") +
6644 ; +
6645 ; SIZE MAY BE ANY NUMBER OF BYTES +
6646 ; +
6647 ; SETS A FLAG ("EXPINP") IF RESPONSE OTHER THAN "ACK" IS +
6648 ; TO BE EXPECTED +
6649 ; +
6650 ; SETS ANOTHER FLAG ("EOMSG") IF THIS IS THE LAST MESSAGE +
6651 ;+++++
6652 012052 MSG:
6653
6656 012052 010446 MOV R4, -(SP)
(2) 012054 010604 MOV SP, R4
(3) 012056 162706 000002 SUB #2, SP
6657 ;
6658 ; GET MESSAGE POINTER
6659 ;
6660 012062 017764 170064 177776 MOV @MSGAP, A(R4)
6661 ;
6662 ; IF MESSAGE POINTER IS ODD.
6663 ; SET END_OF_MESSAGE INDICATOR
6664 ;
6665 012070 016467 177776 170004 MOV A(R4), SCRTCH
(6) 012076 016746 170000 MOV SCRTCH, -(SP)
(6) 012102 042716 000001 BIC #1, (SP)
(6) 012106 042667 167770 BIC (SP)+, SCRTCH
6666 012112 005767 167764 TST SCRTCH
(9) 012116 001405 BEQ 50002$
6667 012120 005267 170000 INC EOMSG
6668 012124 042764 000001 177776 BIC #1, A(R4)
6669 012132 50002$:
6670 ;
6671 ; IF MESSAGE_POINTER IS NEGATIVE.
6672 ; SET EXPECT_INPUT INDICATOR
6673 ;
6674 012132 005764 177776 TST A(R4)
(9) 012136 002006 BGE 50003$
6675 012140 005267 167764 INC EXPINP
6676 012144 042764 100000 177776 BIC #100000, A(R4)
6677 ;
6678 ; OTHERWISE, CLEAR EXPECT_INPUT
6679 ;
6680 012152 000402 BR 50004$
(3) 012154 50003$:
6681 012154 005067 167750 CLR EXPINP
6682 012160 50004$:
6683 ;
6684 ; SET MESSAGE_ADDRESS AND
6685 ; MESSAGE_SIZE.

```

```

6686
6687 012160 016467 177776 167744      MOV      A(R4), OUTMSG
6688 012166 062767 000002 167756      ADD      #2, MSGAP
6689 012174 017767 167752 167732      MOV      @MSGAP, OUTMSG+2
6690
6691      ; UPDATE MESSAGE_AREA_POINTER
6692
6693 012202 067767 167744 167742      ADD      @MSGAP, MSGAP
6694 012210 062767 000002 167734      ADD      #2, MSGAP
6695
6696      ; IF MESSAGE_AREA_POINTER IS ODD
6697      ; ADD 1 TO MAKE IT EVEN
6698
6699 012216 016767 167730 167656      MOV      MSGAP, SCRTCH
    (6) 012224 016746 167652      MOV      SCRTCH, -(SP)
    (6) 012230 042716 000001      BIC      #1, (SP)
    (6) 012234 042667 167642      BIC      (SP)+, SCRTCH
6700 012240 005767 167636      TST      SCRTCH
    (9) 012244 001402      BEQ      50005$
6701 012246 005267 167700      INC      MSGAP
6702 012252      50005$:
6703 012252      50000$:
    (3) 012252      50001$:
    (3) 012252 010406      MOV      R4, SP
    (3) 012254 012604      MOV      (SP)+, R4
    (2) 012256 000207      RTS      PC
6704
6705
6706
6707
6708      ; ++++++
6709      ; ROUTINE TO HANDLE OUTPUT OF MESSAGES TO THE TERMINAL      +
6710      ; UNDER TEST. MESSAGES MAY BE OF ANY LENGTH SINCE      +
6711      ; ROUTINES USED BY THIS ROUTINE SPLIT THE MESSAGES INTO      +
6712      ; BUFFERS OF APPROPRIATE LENGTH FOR DDCMP.      +
6713      ; ++++++
6714 012260      TEROUT:
6715 012260 005767 003132      TST      INIT
    (9) 012264 001011      BNE      50002$
6716 012266 004767 000222      JSR      PC, XSTRT
6717 012272 103002      BCC      50003$
6718 012274 000261      SEC
    (4) 012276 000504      BR       50001$
6719 012300      50003$:
6720 012300 005067 167660      CLR      MSGNO
6721 012304 005067 167656      CLR      RMSGNO
6722 012310      50002$:
6723
6724      ; INITIALIZE THE BUFFER ROUTINES 3.2.20.4
6725
6726 012310 016767 167616 167636      MOV      OUTMSG, INITMA
6727 012316 012767 002132 167556      MOV      #OUTMSG, SCRTCH
    (6) 012324 062767 000002 167550      ADD      #2, SCRTCH
6728 012332 017767 167544 167616      MOV      @SCRTCH, INITMS
6729
6730 012340 012767 010172 167612      BUFFER #1, BAPNT
    MOV      #TMPBUF, BAPNT
  
```

```

6731 012346 012767 000412 167606      MOV      #266.,BASIZE
6732                                     ;
6733                                     ; GET A BUFFER OF THE MESSAGE
6734                                     ;
6735 012354 162705 000004      SUB      #2*2,R5
(3) 012360 004767 000410      JSR      PC,GETOB
(4) 012364 012567 167552      MOV      (R5)+,OUTBUF
(4) 012370 012567 167550      MOV      (R5)+,OUTSIZ
6736                                     ;
6737                                     ; PERFORM THE FOLLOWING LOOP WHILE
6738                                     ; A AND ABORT ARE CLEAR AND OUTSIZ
6739                                     ; IS GREATER THAN 0
6740                                     ;
6741 012374                                     50004$:
(6) 012374 005767 167544      TST      OUTSIZ
(9) 012400 003442      BLE      50005$
(6) 012402 005767 167500      TST      ABORT
(9) 012406 001037      BNE      50005$
6742                                     ;
6743                                     ; ADD PROTOCOL ENVELOPE AND TRANSMIT
6744                                     ;
6745 012410 162705 000004      SUB      #2*2,R5
(3) 012414 010546      MOV      R5,-(SP)
(5) 012416 016745 167522      MOV      OUTSIZ,-(R5)
(4) 012422 016745 167514      MOV      OUTBUF,-(R5)
(3) 012426 004767 000470      JSR      PC,PROBUF
(3) 012432 012605      MOV      (SP)+,R5
(4) 012434 012567 002762      MOV      (R5)+,XBUFFER
(4) 012440 012567 002760      MOV      (R5)+,XSIZE
6746 012444 004767 001756      JSR      PC,XMIT
6747                                     ;
6748                                     ; IF TRANSMIT ERROR, REPORT IT AND SET ERRXMT
6749                                     ;
6750 012450 103005      BCC      50006$
6751 012452 104423      TRAP    T$ERCODE
(5) 012454 000024      .WORD   20
(5) 012456 002620      .WORD   MSG20
6752 012460 000261      SEC
(4) 012462 000412      BR      50001$
6753 012464                                     50006$:
6754                                     ;
6755                                     ; GET THE NEXT BUFFER
6756                                     ;
6757 012464 162705 000004      SUB      #2*2,R5
(3) 012470 004767 000300      JSR      PC,GETOB
(4) 012474 012567 167442      MOV      (R5)+,OUTBUF
(4) 012500 012567 167440      MOV      (R5)+,OUTSIZ
6758                                     ;
6759                                     ; END OF LOOP
6760                                     ;
6761 012504 000733      BR      50004$
(3) 012506                                     50005$:
6762                                     ;
6763                                     ; IF ERROR OCCURRED, DO AN ERROR RETURN
6764                                     ;
6765 012506                                     50000$:

```


(2) 012506 000241
 (3) 012510
 (2) 012510 000207
 6766 012512 000000
 6767
 6768
 6769
 6770
 6771
 6772
 6773
 6774
 6775
 6776 012514
 6779 012514 010446
 (2) 012516 010604
 (3) 012520 162706 000002
 6780
 6781
 6782
 6783
 6784
 6785 012524 012767 000010 002672
 6786 012532 012767 003500 002662
 6787
 6788
 6789
 6790 012540 012764 002170 177776
 (6) 012546 062764 000005 177776
 6791 012554 117467 177776 170723
 6792
 6793
 6794
 6795
 6796 012562 162705 000002
 (3) 012566 010546
 (5) 012570 012745 000006
 (4) 012574 016745 002622
 (3) 012600 004767 004014
 (3) 012604 012605
 (4) 012606 012567 170674
 6797
 6798
 6799
 6800 012612 005067 000154
 6801 012616
 6802 012616 004767 001604
 6803
 6804
 6805
 6806 012622 103005
 6807 012624 104423
 (5) 012626 000024
 (5) 012630 002620
 6808 012632 000261
 (4) 012634 000453

```

                    CLC
500015:             RTS      PC
TYPE:              0      ;TYPE OF MESSAGE EXPECTED

;*****
;      ROUTINE TO SEND A "STRT" MESSAGE TO THE TERMINAL      +
;      A "STACK" MUST BE RETURNED BY THE TERMINAL            +
;*****
XSTRT:             MOV      R4, -(SP)
                   MOV      SP, R4
                   SUB      #2, SP
;
; THE DDCMP STRT MESSAGE IS 10 BYTES LONG (INCLUDING CRC) AND IS IN
; A BUFFER STARTING AT "STRTB" -- PUT THIS INFORMATION
; IN THE TRANSMIT ROUTINE'S POINTERS
;
                   MOV      #10, XSIZE
                   MOV      #STRTB, XBUFFER
;
; GET THE TERMINAL ADDRESS AND PUT INTO THE STRT MESSAGE
;
                   MOV      #PTABLE, A(R4)
                   ADD      #5, A(R4)
                   MOVB     @A(R4), STRTSA
;
; COMPUTE THE CRC FOR THE STRT MESSAGE AND PUT IN THE LAST TWO BYTES
; OF THE BUFFER ("STRTC")
;
                   SUB      #1*2, R5
                   MOV      R5, -(SP)
                   MOV      #6, -(R5)
                   MOV      XBUFFER, -(R5)
                   JSR      PC, CRC
                   MOV      (SP)+, R5
                   MOV      (R5)+, STRTC
;
; TRANSMIT THE MESSAGE
;
                   CLR      DDSTRY
500025:             JSR      PC, XMIT
;
; IF AN ERROR OCCURS, REPORT IT AND EXIT WITH THE ERROR FLAG SET
;
                   BCC      500035
                   TRAP     T$ERCODE
                   WORD     20
                   WORD     MSG20
                   SEC
                   BR       500015

```

```

6809 012636          50003$:
6810
6811                ; OTHERWISE, GET THE TERMINAL'S RESPONSE
6812                ;
6813 012636 010546          MOV     R5, -(SP)
   (4) 012640 012745 000001  MOV     #1, -(R5)
   (3) 012644 004767 003456  JSR    PC, RCVE
   (3) 012650 012605          MOV     (SP)+, R5
6814                ;
6815                ; IF RECEPTION ERROR, REPORT IT AND EXIT WITH THE ERROR FLAG SET
6816                ;
6817 012652 103003          BCC    50004$
6818 012654 005267 000112  INC    DDSTRY
6819 012660 000402          BR     50005$
   (3) 012662          50004$:
6820 012662 005067 000104  CLR    DDSTRY
6821 012666          50005$:
6822 012666 005767 000100  TST    DDSTRY
   (6) 012672 001404          BEQ    50006$
   (4) 012674 026727 000072 000003  CMP    DDSTRY, #3
   (7) 012702 001345          BNE    50002$
   (4) 012704          50006$:
6823 012704 005767 000062  TST    DDSTRY
   (9) 012710 001405          BEQ    50007$
6824 012712 104423          TRAP   TSEPCODE
   (5) 012714 000025          .WORD  21
   (5) 012716 002637          .WORD  MSG21
6825 012720 000261          SEC
   (4) 012722 000420          BR     50001$
6826 012724          50007$:
6827
6828                ; OTHERWISE, SET TYPE TO "STACK" AND CHECK TO SEE THAT IT WAS THE REPLY
6829                ;
6830 012724 012767 000007 177560  MOV    #STACK, TYPE
6831 012732 010546          MOV    R5, -(SP)
   (4) 012734 016745 177552  MOV    TYPE, -(R5)
   (3) 012740 004767 001072  JSR    PC, CHKREP
   (3) 012744 012605          MOV    (SP)+, R5
6832                ;
6833                ; IF REPLY WASN'T A "STACK", REPORT THE ERROR AND EXIT WITH THE ERROR FLAG SET
6834                ;
6835 012746 103005          BCC    50010$
6836 012750 104423          TRAP   TSEPCODE
   (5) 012752 000031          .WORD  25
   (5) 012754 003025          .WORD  MSG25
6837 012756 000261          SEC
   (4) 012760 000401          BR     50001$
6838 012762          50010$:
6839
6840                ; END OF XSTRT ROUTINE
6841                ;
6842 012762          50000$:
   (2) 012762 000241          CLC
   (3) 012764          50001$:
   (3) 012764 010406          MOV    R4, SP
   (3) 012766 012604          MOV    (SP)+, R4

```

(2)	012770	000207			RTS	PC
6843	012772	000000			DDSTRY:	0
6844						
6845						
6846						
6847						
6848						
6849						
6850						
6851						
6852						
6853						
6854						
6855	012774					
6858	012774	010446			MOV	R4, -(SP)
(2)	012776	010604			MOV	SP, R4
(3)	013000	162706	000002		SUB	#2, SP
6859						
6860						
6861						
6862	013004	016765	167150	000000	MOV	BAPNT, A(R5)
(6)	013012	062765	000010	000000	ADD	#10, A(R5)
6863						
6864						
6865						
6866	013020	005065	000002		CLR	B(R5)
6867						
6868						
6869						
6870	013024	016767	167132	167050	MOV	BASIZE, SCRTCH
(6)	013032	162767	000012	167042	SUB	#12, SCRTCH
6871						
6872						
6873						
6874	013040	016564	000000	177776	MOV	A(R5), C(R4)
6875						
6876						
6877						
6878	013046					
(6)	013046	026567	000002	167026	CMP	B(R5), SCRTCH
(9)	013054	002017			BGE	50003%
(6)	013056	005767	167074		TST	INITMS
(9)	013062	003414			BLE	50003%
6879						
6880						
6881						
6882	013064	117774	167064	177776	MOVB	@INITMA, @C(R4)
6883	013072	005367	167060		DEC	INITMS
6884	013076	005267	167052		INC	INITMA
6885	013102	005265	000002		INC	B(R5)
6886	013106	005264	177776		INC	C(R4)
6887	013112	000755			BR	50002%
(3)	013114					
6888						
6889						
6890						

```

;*****
; THIS ROUTINE; SPLITS A MESSAGE OF ANY LENGTH INTO
; SEGMENTS OF A SIZE SUITABLE FOR TRANSMISSION OVER A DATA
; LINK -- RETURNS BUFFER ADDRESS, BUFFER SIZE AND END_OF_MESSAGE+
; (BUFFER SIZE = 0 IS END_OF_MESSAGE)
;*****

```

```

GETOB:
MOV R4, -(SP)
MOV SP, R4
SUB #2, SP

```

```

; LEAVE ENOUGH ROOM FOR A DDCMP HEADER
MOV BAPNT, A(R5)
ADD #10, A(R5)

```

```

; INIT SIZE TO ZERO
CLR B(R5)

```

```

; SCRTCH HAS MAXIMUM DDCMP MESSAGE SIZE
MOV BASIZE, SCRTCH
SUB #12, SCRTCH

```

```

; GET A COPY OF THE BUFFER POINTER
MOV A(R5), C(R4)

```

```

; REPEAT LOOP UNTIL MAX SIZE OR END OF DATA
50002%:
CMP B(R5), SCRTCH
BGE 50003%
TST INITMS
BLE 50003%

```

```

; MOVE DATA INTO MESSAGE BUFFER
MOVB @INITMA, @C(R4)
DEC INITMS
INC INITMA
INC B(R5)
INC C(R4)
BR 50002%

```

```

50003%:
; END OF LOOP

```

6891 013114
 (3) 013114
 (3) 013114 010406
 (3) 013116 012604
 (2) 013120 000207
 6892
 6893
 6894
 6895
 6896
 6897
 6898
 6899
 6900
 6901
 6902 013122
 6903 013122 010146
 6904 013124 010246
 6905
 6906
 6907
 6908 013126 016502 000000
 (6) 013132 162702 000010
 6909
 6910
 6911
 6912 013136 112722 000201
 6913
 6914
 6915
 6916 013142 016501 000002
 6917 013146 110122
 6918 013150 000301
 6919 013152 110112
 6920
 6921
 6922
 6923 013154 152712 000200
 6924 013160 005202
 6925
 6926
 6927
 6928 013162 116722 167000
 6929
 6930
 6931
 6932 013166 005267 166772
 6933 013172 116722 166766
 6934
 6935
 6936
 6937 013176 012701 002170
 (6) 013202 062701 000005
 6938 013206 111122
 6939
 6940

```

50000$:
50001$:
      MOV      R4, SP
      MOV      (SP)+, R4
      RTS      PC

;+++++
;      ROUTINE TO PROVIDE LINE PROTOCOL HEADER AND CHECKSUMS      +
;      CALL WITH DATA AREA ADDRESS AND SIZE (IN BYTES)          +
;      RETURN WITH MODIFIED ADDRESS AND SIZE                      +
;+++++
PROBUF:
      MOV      R1, -(SP)
      MOV      R2, -(SP)
;
; CHANGE POINTER TO INCLUDE DDCMP HEADER
;
      MOV      A(R5), R2
      SUB      #10, R2
;
; FIRST BYTE IS SOH (START OF HEADER)
;
      MOVB    #SOH, (R2)+
;
; NEXT TWO BYTES GET MESSAGE SIZE
;
      MOV      B(R5), R1
      MOVB    R1, (R2)+
      SWAB    R1
      MOVB    R1, (R2)
;
; SET SELECT BIT
;
      BISB    #200, (R2)
      INC     R2
;
; NEXT BYTE IS LAST GOOD RESPONSE
;
      MOVB    RMSGNO, (R2)+
;
; NEXT BYTE GETS MESSAGE NUMBER --- POP NUMBER FOR NEXT MESSAGE (IF ANY)
;
      INC     MSGNO
      MOVB    MSGNO, (R2)+
;
; GET THE TERMINAL ADDRESS FOR THE NEXT HEADER BYTE
;
      MOV      #PTABLE, R1
      ADD     #5, R1
      MOVB    (R1), (R2)+
;
; GET HEADER START ADDRESS AND SIZE IN BYTES

```

```
6941
6942 013210 016567 000000 166664      MOV    A(R5), SCRTCH
(6) 013216 162767 000010 166656      SUB    #10, SCRTCH
6943 013224 012767 000006 166652      MOV    #6, TEMP
6944
6945      ; COMPUTE HEADER CRC
6946
6947 013232 162705 000002      SUB    #1*2, R5
(3) 013236 010546      MOV    R5, -(SP)
(5) 013240 016745 166640      MOV    TEMP, -(R5)
(4) 013244 016745 166632      MOV    SCRTCH, -(R5)
(3) 013250 004767 003344      JSR    PC, CRC
(3) 013254 012605      MOV    (SP)+, R5
(4) 013256 012567 166622      MOV    (R5)+, TEMP
6948
6949      ; PUT CRC INTO BUFFER
6950
6951 013262 016712 166616      MOV    TEMP, (R2)
6952
6953      ; GET ADDRESS OF MESSAGE AND MESSAGE SIZE IN BYTES
6954
6955 013266 016567 000000 166606      MOV    A(R5), SCRTCH
6956 013274 016567 000002 166602      MOV    B(R5), TEMP
6957
6958      ; COMPUTE DATA CRC
6959
6960 013302 162705 000002      SUB    #1*2, R5
(3) 013306 010546      MOV    R5, -(SP)
(5) 013310 016745 166570      MOV    TEMP, -(R5)
(4) 013314 016745 166562      MOV    SCRTCH, -(R5)
(3) 013320 004767 003274      JSR    PC, CRC
(3) 013324 012605      MOV    (SP)+, R5
(4) 013326 012567 166552      MOV    (R5)+, TEMP
6961
6962      ; PUT DATA CRC IN LAST TWO BYTES OF BUFFER
6963
6964 013332 016502 000000      MOV    A(R5), R2
(6) 013336 066502 000002      ADD    B(R5), R2
6965 013342 116722 166536      MOVB  TEMP, (R2)+
6966 013346 000367 166532      SWAB  TEMP
6967 013352 116712 166526      MOVB  TEMP, (R2)
6968
6969      ; RETURN WITH MODIFIED ADDRESS AND SIZE
6970
6971 013356 016565 000002 000006      MOV    B(R5), D(R5)
(6) 013364 062765 000012 000006      ADD    #12, D(R5)
6972 013372 016565 000000 000004      MOV    A(R5), C(R5)
(6) 013400 162765 000010 000004      SUB    #10, C(R5)
6973 013406 012602      MOV    (SP)+, R2
6974 013410 012601      MOV    (SP)+, R1
6975 013412      500005:
(3) 013412      500015:
(2) 013412 000207      RTS    PC
6976
6977
6978
```

```

6979
6980
6981 ;+++++
6982 ; ROUTINE TO GET TERMINAL INPUT +
6983 ;+++++
6984 013414 TERIN:
6985 ;
6986 ;SAVE R1
6987 ;
6988 013414 010146 MOV R1,-(SP)
6989 ;
6990 ;GET TERMINAL RESPONSE
6991 ;
6992 013416 010546 MOV R5,-(SP)
(4) 013420 012745 000000 MOV #0,-(R5)
(3) 013424 004767 002676 JSR PC,RCVE
(3) 013430 012605 MOV (SP)+,R5
6993 ;
6994 ; IF RECEIVE ERROR, REPORT IT AND EXIT WITH THE ERROR FLAG SET
6995 ;
6996 013432 103007 BCC 50002$
6997 013434 104463 TRAP T$ERCODE
(5) 013436 000050 .WORD 40
(5) 013440 002637 .WORD MSG40
(5) 013442 004072 .WORD AREA40
6998 ;
6999 ;.... RESTORE R1 FIRST
7000 ;
7001 013444 012601 MOV (SP)+,R1
7002 013446 000261 SEC
(4) 013450 000571 BR 50001$
7003 013452 50002$:
7004 ;
7005 ; IF MESSAGE IS ONLY 10 BYTES, SEE IF IT IS AN ACK.
7006 ; IT'S OK TO BE AN ACK, BUT RECEPTION IS INCOMPLETE (CLEAR FLAG).
7007 ;
7008 013452 026727 166472 000010 CMP INSIZ,#10
(9) 013460 001027 BNE 50003$
7009 013462 005065 000000 CLR A(R5)
7010 013466 012767 000001 177016 MOV #ACK,TYPE
7011 013474 010546 MOV R5,-(SP)
(4) 013476 016745 177010 MOV TYPE,-(R5)
(3) 013502 004767 000330 JSR PC,CHKREP
(3) 013506 012605 MOV (SP)+,R5
7012 013510 103003 BCC 50004$
7013 013512 012601 MOV (SP)+,R1
7014 013514 000261 SEC
(4) 013516 000546 BR 50001$
7015 013520 50004$:
7016 013520 016701 166422 MOV INBUF,R1
(6) 013524 062701 000004 ADD #4,R1
7017 013530 111167 166432 MOVB (R1),RMSGNO
7018 013534 012601 MOV (SP)+,R1
7019 013536 000535 BR 50000$
7020 013540 50003$:
7021 ;
  
```

```

7022 ; IF RECEIVE OK, CHECK THE HEADER CRC
7023 ; COMPUTE EXPECTED CRC
7024 ;
7025 013540 162705 000002          SUB    #1*2,R5
      (3) 013544 010546          MOV    R5,-(SP)
      (5) 013546 012745 000006          MOV    #6,-(R5)
      (4) 013552 016745 166370          MOV    INBUF,-(R5)
      (3) 013556 004767 003036          JSR    PC,CRC
      (3) 013562 012605          MOV    (SP)+,R5
      (4) 013564 012567 170172          MOV    (R5)+,A04EX
7026 ;
7027 ; GET THE CRC RECEIVED
7028 ;
7029 013570 016701 166352          MOV    INBUF,R1
      (6) 013574 062701 000010          ADD    #10,R1
7030 013600 114167 170154          MOVB  -(R1),A04RC
7031 013604 000367 170150          SWAB  A04RC
7032 013610 114167 170144          MOVB  -(R1),A04RC
7033 ;
7034 ; IF THEY AREN'T THE SAME, REPORT THE ERROR, RESTORE R1 AND EXIT WITH
7035 ; THE ERROR FLAG SET
7036 ;
7037 013614 026767 170142 170136    CMP    A04EX,A04RC
      (9) 013622 001407          BEQ    50005$
7038 013624 104463          TRAP  T$EPCODE
      (5) 013626 000051          .WORD 41
      (5) 013630 002411          .WORD MSG41
      (5) 013632 003726          .WORD AREA41
7039 013634 012601          MOV    (SP)+,R1
7040 013636 000261          SEC
      (4) 013640 000475          BR     50001$
7041 013642
7042 ;
7043 ; COMPARE TERMINAL ADDRESS RECEIVED WITH THAT EXPECTED
7044 ; AND IF THEY AREN'T THE SAME, REPORT THE ERROR, RESTORE R1 AND
7045 ; EXIT WITH THE ERROR FLAG SET
7046 ;
7047 013642 124167 167637          CMPB  -(R1),STRTSA
      (9) 013646 001407          BEQ    50006$
7048 013650 104463          TRAP  T$EPCODE
      (5) 013652 000052          .WORD 42
      (5) 013654 002345          .WORD MSG42
      (5) 013656 003674          .WORD AREA42
7049 013660 012601          MOV    (SP)+,R1
7050 013662 000261          SEC
      (4) 013664 000463          BR     50001$
7051 013666
7052 ;
7053 ; MOVE THE POINTERS PAST THE HEADER AND COMPUTE THE DATA CRC
7054 ;
7055 013666 062767 000010 166252    ADD    #10,INBUF
7056 013674 162767 000012 166246    SUB    #12,INSIZ
7057 013702 162705 000002          SUB    #1*2,R5
      (3) 013706 010546          MOV    R5,-(SP)
      (5) 013710 016745 166234          MOV    INSIZ,-(R5)
      (4) 013714 016745 166226          MOV    INBUF,-(R5)
  
```

```
(3) 013720 004767 002674      JSR    PC,CRC
(3) 013724 012605              MOV    (SP)+,R5
(4) 013726 012567 170030      MOV    (R5)+,A04EX
7058
7059      ; GET THE DATA CRC RECEIVED
7060      ;
7061 013732 016701 166210      MOV    INBUF,R1
(6) 013736 066701 166206      ADD    INSIZ,R1
7062 013742 062701 000002      ADD    #2,R1
7063 013746 114167 170006      MOVB  -(R1),A04RC
7064 013752 000367 170002      SWAB  A04RC
7065 013756 114167 167776      MOVB  -(R1),A04RC
7066
7067      ; IF THEY AREN'T THE SAME; REPORT THE ERROR, RESTORE R1 AND
7068      ; EXIT WITH THE ERROR FLAG SET
7069      ;
7070 013762 026767 167772 167772  CMP    A04RC,A04EX
(9) 013770 001407              BEQ    50007$
7071 013772 104463              TRAP  T$ERCODE
(5) 013774 000053              .WORD 43
(5) 013776 003224              .WORD MSG43
(5) 014000 003726              .WORD AREA43
7072 014002 012601              MOV    (SP)+,R1
7073 014004 000261              SEC
(4) 014006 000412              BR     50001$
7074 014010      50007$:
7075      ;
7076      ; OTHERWISE; SET COMPLETE FLAG, RESTORE R1 AND EXIT NORMALLY
7077      ;
7078 014010 005265 000000      INC    A(R5)
7079 014014 016701 166126      MOV    INBUF,R1
(6) 014020 162701 000004      SUB    #4,R1
7080 014024 111167 166136      MOVB  (R1),RMSGNO
7081 014030 012601              MOV    (SP)+,R1
7082 014032      50000$:
(2) 014032 000241              CLC
(3) 014034      50001$:
(2) 014034 000207              RTS    PC
7083
7084
7085
7086
7087
7088
7089      ; ++++++
7090      ; ROUTINE TO CHECK AN INPUT BUFFER FOR A STANDARD +
7091      ; RESPONSE ("ACK" OR "STACK") +
7092      ; +
7093      ; LOCATION "TYPE" CONTAINS THE RESPONSE CODE EXPECTED +
7094      ; +
7095      ; INPUT BUFFER POINTER IS IN "INBUF" +
7096      ; +
7097      ; INPUT BUFFER SIZE (IN BYTES) IS IN "INSIZ" +
7098      ; ++++++
7099      ;
7102 014036 010446      CHKREP: MOV    R4,-(SP)
```


(2)	014040	010604			MOV	SP, R4
(3)	014042	162706	000002		SUB	#2, SP
7103						
7104					; IF MESSAGE IS NOT 10 BYTES LONG, IT CAN'T BE A CONTROL MESSAGE	
7105					; TYPE AN ERROR MESSAGE AND EXIT WITH ERROR SET	
7106						
7107	014046	026727	166076	000010	CMP	INSIZ, #10
(9)	014054	001406			BEQ	500025
7108	014056	104463			TRAP	T\$ERCODE
(5)	014060	000001			. WORD	1
(5)	014062	002273			. WORD	MSG01
(5)	014064	003570			. WORD	AREA01
7109	014066	000261			SEC	
(4)	014070	000553			BR	500015
7110	014072				500025:	
7111						
7112					; OTHERWISE, COMPUTE THE CRC AND CHECK IT AGAINST THAT SENT	
7113						
7114	014072	162705	000002		SUB	#1*2, R5
(3)	014076	010546			MOV	R5, -(SP)
(5)	014100	012745	000006		MOV	#6, -(R5)
(4)	014104	016745	166036		MOV	INBUF, -(R5)
(3)	014110	004767	002504		JSR	PC, CRC
(3)	014114	012605			MOV	(SP)+, R5
(4)	014116	012567	165762		MOV	(R5)+, TEMP
7115	014122	016764	166020	177776	MOV	INBUF, A(R4)
(6)	014130	062764	000007	177776	ADD	#7, A(R4)
7116	014136	117467	177776	165736	MOVB	@A(R4), SCRTCH
7117	014144	000367	165732		SWAB	SCRTCH
7118	014150	005364	177776		DEC	A(R4)
7119	014154	117467	177776	165720	MOVB	@A(R4), SCRTCH
7120						
7121					; IF CRC'S MATCH GO ON... OTHERWISE SKIP FOLLOWING CODE	
7122						
7123	014162	026767	165716	165712	CMP	TEMP, SCRTCH
(9)	014170	001076			BNE	500045
7124						
7125					; COMPARE CONTROL MESSAGE TYPE WITH THAT PASSED AS AN ARGUMENT	
7126						
7127	014172	016764	165750	177776	MOV	INBUF, A(R4)
(6)	014200	005264	177776		INC	A(R4)
7128	014204	117464	177776	177776	MOVB	@A(R4), A(R4)
7129	014212	042764	177400	177776	BIC	#HYBYTE, A(R4)
7130						
7131					; IF NOT THE EXPECTED TYPE, PRINT ERROR MESSAGE AND EXIT WITH ERROR SET	
7132						
7133	014220	026465	177776	000000	CMP	A(R4), B(R5)
(9)	014226	001414			BEQ	500055
7134	014230	016467	177776	167366	MOV	A(R4), A02A
7135	014236	016567	000000	167362	MOV	B(R5), A02B
7136	014244	104463			TRAP	T\$ERCODE
(5)	014246	000002			. WORD	2
(5)	014250	002320			. WORD	MSG02
(5)	014252	003572			. WORD	AREA02
7137	014254	000261			SEC	
(4)	014256	000460			BR	500015

```

7138
7139 ; OTHERWISE, CHECK TERMINAL ADDRESS
7140 ;
7141 014260 500055:
7142 014260 012767 002170 165614 MOV #PTABLE, SCRTCH
(6) 014266 062767 000005 165606 ADD #5, SCRTCH
7143 014274 117767 165602 165600 MOVB @SCRTCH, SCRTCH
7144 014302 042767 177400 165572 BIC #HYBYTE, SCRTCH
7145 014310 016767 165632 165566 MOV INBUF, TEMP
(6) 014316 062767 000005 165560 ADD #5, TEMP
7146 014324 117767 165554 165552 MOVB @TEMP, TEMP
7147 014332 042767 177400 165544 BIC #HYBYTE, TEMP
7148
7149 ; IF NOT THE EXPECTED TERMINAL ADDRESS, THEN REPORT THE ERROR AND
7150 ; EXIT WITH ERROR SET
7151 ;
7152 014340 026767 165540 165534 CMP TEMP, SCRTCH
(9) 014346 001406 BEQ 500075
7153 014350 104463 TRAP TSERCODE
(5) 014352 000003 .WORD 3
(5) 014354 002345 .WORD MSG03
(5) 014356 003674 .WORD AREA03
7154 014360 000261 SEC
(4) 014362 000416 BR 500015
7155 014364 500075:
7156 014364 500065:
7157 ;
7158 ; IF CRC WAS IN ERROR, SKIP ABOVE CODE, COME TO HERE, REPORT ERROR AND
7159 ; EXIT WITH ERROR SET
7160 ;
7161 014364 000414 BR 500105
(3) 014366 500045:
7162 014366 016767 165510 167364 MOV SCRTCH, A04RC
7163 014374 016767 165504 167360 MOV TEMP, A04EX
7164 014402 104463 TRAP TSERCODE
(5) 014404 000004 .WORD 4
(5) 014406 002411 .WORD MSG04
(5) 014410 003726 .WORD AREA04
7165 014412 000261 SEC
(4) 014414 000401 BR 500015
7166 014416 500105:
7167 014416 500035:
7168 ;
7169 ; NORMAL RETURN
7170 ;
7171 014416 500005:
(2) 014416 000241 CLC
(3) 014420 500015:
(3) 014420 010406 MOV R4, SP
(3) 014422 012604 MOV (SP)+, R4
(2) 014424 000207 RTS PC
7172
7173
7174
7175
7176

```

```

7177 ;+++++
7178 ; THIS ROUTINE SETS UP AND CHECKS FOR COMPLETE TRANSMISSION +
7179 ; RETURNS AN ERROR FLAG IF ANY ERROR OCCURS DURING TRANSMISSION +
7180 ; SET "ABORT" IF INITIALIZATION OF THE DEVICE FAILS +
7181 ;+++++
7182 014426 XMIT:
7185 014426 010446 MOV R4, -(SP)
(2) 014430 010604 MOV SP, R4
(3) 014432 162706 000002 SUB #2, SP
7186 ;
7187 ; CLEAR ERROR COUNT, TRANSMIT DONE FLAG AND RECEIVE DONE FLAG
7188 ;
7189 014436 005064 177776 CLR B(R4)
7190 014442 005067 000270 CLR XDONE
7191 014446 005067 000266 CLR RDONE
7192 ;
7193 ; START OF LOOP
7194 ;
7195 014452 50002$:
7196 ;
7197 ; TRY TO INIT DEVICE
7198 ;
7199 014452 004767 000264 JSR PC, DEVINIT
7200 ;
7201 ; IF ERROR OCCURRED...
7202 ;
7203 014456 103033 BCC 50003$
7204 014460 010146 MOV R1, -(SP)
7205 014462 010246 MOV R2, -(SP)
7206 ;
7207 ; GET A COPY OF THE DEVICE REGISTERS
7208 ;
7209 014464 016701 165500 MOV PTAB1, R1
7211 014470 012702 002224 MOV #DUPREG, R2
7217 014474 016162 000000 000000 MOV RCVCSR(R1), RCVCSR(R2)
7218 014502 016152 000002 000002 MOV RCVBUF(R1), RCVBUF(R2)
7219 014510 016162 000004 000004 MOV XMTCSR(R1), XMTCSR(R2)
7220 014516 016162 000006 000006 MOV XMTBUF(R1), XMTBUF(R2)
7231 014524 012602 MOV (SP)+, R2
7232 014526 012601 MOV (SP)+, R1
7233 ;
7234 ; REPORT THE ERROR AND UPDATE THE ERROR COUNT
7235 ;
7236 014530 104463 TRAP TSERCODE
(5) 014532 000027 .WORD 23
(5) 014534 002722 .WORD MSG23
(5) 014536 004072 .WORD AREA23
7237 014540 005264 177776 INC B(R4)
7238 ;
7239 ; OTHERWISE, CLEAR THE ERROR COUNT
7240 ;
7241 014544 000402 BR 50004$
(3) 014546 50003$:
7242 014546 005064 177776 CLR B(R4)
7243 014552 50004$:
7244 ;
  
```

```

7245 ; IF ERROR COUNT HAS EXCEEDED LIMITS, ABORT AND NEGATE THE COUNT
7246 ;
7247 014552 026467 177776 165334      CMP      B(R4), ILIMIT
(9) 014560 001010                      BNE      50005$
7248 014562 104462                      TRAP     TSERCODE
(5) 014564 000000                      .WORD   0
(5) 014566 002234                      .WORD   MSG00
(5) 014570 003510                      .WORD   AREA00
7249 014572 005267 165310              INC      ABORT
7250 014576 005464 177776              NEG      B(R4)
7251 014602                               50005$:
7252 ;
7253 ;                                     END OF LOOP
7254 ; EXIT LOOP WHEN ERROR COUNT IS ZERO OR NEGATIVE (ABORT)
7255 ;
7256 014602 005764 177776              TST      B(R4)
(6) 014606 003321                      BGT      50002$
7257 ;
7258 ; IF NOT ABORT THEN EXECUTE THE FOLLOWING CODE
7259 ; OTHERWISE SKIP AND RETURN WITH ERROR SET
7260 ;
7261 014610 005764 177776              TST      B(R4)
(9) 014614 001042                      BNE      50006$
7262 ;
7263 ; WAIT FOR TRANSMIT DONE FLAG TO SET
7264 ;
7265 014616                               50007$:
7266 014616 104022                      EMT      C$BRK
7267 014620 005767 000112              TST      XDONE
(6) 014624 001774                      BEQ      50007$
7268 ;
7269 ; IF TRANSMIT FLAG IS NEGATIVE, AN ERROR OCCURRED.
7270 ;
7271 014626 005767 000104              TST      XDONE
(9) 014632 002032                      BGE      50010$
7272 014634 010146                      MOV      R1, -(SP)
7273 014636 010246                      MOV      R2, -(SP)
7274 ;
7275 ; IF ERROR, GET COPY OF DEVICE REGISTERS, PRINT MESSAGE AND EXIT WITH ERROR SET
7276 ; OTHERWISE DO A NORMAL EXIT
7277 ;
7278 014640 016701 165324              MOV      PTABLE, R1
7280 014644 012702 002224              MOV      #DUPREG, R2
7286 014650 016162 000000 000000      MOV      RCVCSR(R1), RCVCSR(R2)
7287 014656 016162 000002 000002      MOV      RCVBUF(R1), RCVBUF(R2)
7288 014664 016162 000004 000004      MOV      XMTCSR(R1), XMTCSR(R2)
7289 014672 016162 000006 000006      MOV      XMTBUF(R1), XMTBUF(R2)
7300 014700 012602                      MOV      (SP)+, R2
7301 014702 012601                      MOV      (SP)+, R1
7302 014704 104463                      TRAP     TSERCODE
(5) 014706 000030                      .WORD   24
(5) 014710 002762                      .WORD   MSG24
(5) 014712 004072                      .WORD   AREA24
7303 014714 000261                      SEC
(4) 014716 000404                      BR       50001$
7304 014720                               50010$:

```

7305 014720 000402
 (3) 014722
 7306 014722 000261
 (4) 014724 000401
 7307 014726
 7308 014726
 (2) 014726 000241
 (3) 014730
 (3) 014730 010406
 (3) 014732 012604
 (2) 014734 000207
 7309 014736 000000
 7310 014740 000000
 7311
 7312
 7313
 7314
 7315
 7316
 7317
 7318
 7319
 7320
 7321
 7322
 7323
 7324
 7325
 7327
 7328 014742
 7331 014742 010446
 (2) 014744 010604
 (3) 014746 162706 000002
 7332 014752 010146
 7333
 7334
 7335
 7336 014754 016701 165210
 7337
 7338
 7339
 7340
 7341 014760 052761 000400 000004
 7342 014766 052761 000002 000000
 7343 014774 052761 101226 000002
 7344
 7345
 7346
 7347 015002 005067 000402
 7348
 7349
 7350
 7351 015006
 7352
 7353
 7354

50006\$: BR 50011\$
 SEC
 BR 50001\$
 50011\$:
 50000\$:
 CLC
 50001\$:
 MOV R4, SP
 MOV (SP)+, R4
 RTS PC
 XDONE: 0
 RDONE: 0

```

;+++++
; ROUTINE TO INITIALIZE A DEVICE +
; IF "INIT" IS NON-ZERO, THE ROUTINE WILL EXECUTE AND SET IT +
; RETURNS WITH AN ERROR IF CTS CANNOT BE SET +
; RETURNS WITH ERROR AND "ABORT" SET IF THE DEVICE DOES +
; NOT EXIST +
;+++++

```

```

DEVINIT:
MOV R4, -(SP)
MOV SP, R4
SUB #2, SP
MOV R1, -(SP)
; GET THE DEVICE ADDRESS
MOV PTABLE, R1
; DO MASTER CLEAR, SET DATA TERMINAL READY, AND SET
; PARAMETER REGISTER (DEC MODE, AUTO-CRC INHIBIT, SYNC CHARACTER.
BIS #MASCLR, XMTCSR(R1)
BIS #DTR, RCVCSR(R1)
BIS #PARAM, RCVBUF(R1)
; SET TIMER TO ZERO
CLR TIMER
; LOOP
50002$:
; SET REQUEST TO SEND AND WAIT 10000 MICROSECS

```

```

7355 015006 052711 000004      BIS      #RTS, (R1)
7356 015012 012700 000100      MOV      #100, R0
      (3) 015016 104027      EMT      C$WTU
7357
7358      GET THE CLEAR TO SEND BIT AND UPDATE THE TIMER
7359
7360 015020 011164 177776      MOV      (R1), A(R4)
      (6) 015024 016446 177776      MOV      A(R4), -(SP)
      (6) 015030 042716 020000      BIC      #CTS, (SP)
      (6) 015034 042664 177776      BIC      (SP)+, A(R4)
7361 015040 005267 000344      INC      TIMER
7362
7363      END LOOP
7364      LOOP ENDS WHEN TIMER EXCEEDS LIMIT OR CLEAR TO SEND IS UP
7365
7366 015044 026767 000340 165036      CMP      TIMER, TIM LIM
      (6) 015052 001403      BEQ      50003$
      (4) 015054 005764 177776      TST      A(R4)
      (7) 015060 001752      BEQ      50002$
      (4) 015062
50003$:
7367      IF TIME LIMIT NOT EXCEEDED...
7368
7369
7370 015062 026767 000322 165020      CMP      TIMER, TIM LIM
      (9) 015070 001507      BEQ      50004$
7371
7372      GET THE DEVICE PRIORITY AND SET UP THE RECEIVE INTERRUPT HANDLER
7373
7374 015072 012767 002170 165100      MOV      #PTABLE, DEVPRI
      (6) 015100 062767 000004 165072      ADD      #4, DEVPRI
7375 015106 117767 165066 165064      MOVB    @DEVPRI, DEVPRI
7376 015114 042767 177400 165056      BIC      #HYBYTE, DEVPRI
7377 015122 012764 002170 177776      MOV      #PTABLE, A(R4)
      (6) 015130 062764 000002 177776      ADD      #2, A(R4)
7378 015136 016746 165036      MOV      DEVPRI, -(SP)
      (6) 015142 012746 015634      MOV      #RINT, -(SP)
      (5) 015146 017446 177776      MOV      @A(R4), -(SP)
      (4) 015152 012746 000003      MOV      #3, -(SP)
      (3) 015156 104037      EMT      C$SVEC
      (2) 015160 062706 000010      ADD      #10, SP
7379
7380      CLEAR THE NEW MESSAGE FLAG
7381
7382 015164 005067 000236      CLR      NMSG
7383 015170 017464 177776 177776      MOV      @A(R4), A(R4)
      (6) 015176 062764 000004 177776      ADD      #4, A(R4)
7384
7385      SET UP THE TRANSMIT INTERRUPT HANDLER
7386
7387 015204 016746 164770      MOV      DEVPRI, -(SP)
      (6) 015210 012746 015466      MOV      #XINT, -(SP)
      (5) 015214 016446 177776      MOV      A(R4), -(SP)
      (4) 015220 012746 000003      MOV      #3, -(SP)
      (3) 015224 104037      EMT      C$SVEC
      (2) 015226 062706 000010      ADD      #10, SP
7388
  
```

```

7389 ; SET UP COPIES OF THE DEVICE BUFFER REGISTER ADDRESSES
7390 ;
7391 015232 010167 000146      MOV    R1,XMTADD
(6) 015236 062767 000006 000140  ADD    #6,XMTADD
7392 015244 010167 000136      MOV    R1,RCVADD
(6) 015250 062767 000002 000130  ADD    #2,RCVADD
7393 ;
7394 ; SET COUNTERS FOR FOUR SYNC'S AND FOUR FILL CHARACTERS
7395 ;
7396 015256 012767 000004 000126  MOV    #4,SYNCNT
7397 015264 012767 000004 000122  MOV    #4,FLLCNT
7398 ;
7399 ; ENABLE TRANSMIT AND RECEIVE INTERRUPTS... SET TO FULL DUPLEX
7400 ;
7401 015272 052761 000120 000004  BIS    #TXINTE!SENDFD,XMTCSR(R1)
7402 015300 052761 000120 000000  BIS    #RXINTE,RCVCSR(R1)
7403 ;
7404 ; IF TIME LIMIT WAS EXCEEDED, GET REGISTER CONTENTS, PRINT MESSAGE,
7405 ; AND RETURN WITH ERROR SET
7406 ;
7407 015306 000427      BR     50005$
(3) 015310      50004$:
7408 015310 010246      MOV    R2,-(SP)
7409 015312 012702 002224      MOV    #DUPREG,R2
7410 015316 016162 000000 000000  MOV    RCVCSR(R1),RCVCSR(R2)
7411 015324 016162 000002 000002  MOV    RCVBUF(R1),RCVBUF(R2)
7412 015332 016162 000004 000004  MOV    XMTCSR(R1),XMTCSR(R2)
7413 015340 016162 000006 000006  MOV    XMTBUF(R1),XMTBUF(R2)
7414 015346 012602      MOV    (SP)+,R2
7415 015350 012601      MOV    (SP)+,R1
7416 015352 104463      TRAP   T$ERCODE
(5) 015354 000036      .WORD 30
(5) 015356 003117      .WORD MSG30
(5) 015360 004266      .WORD AREA30
7417 015362 000261      SEC
(4) 015364 000404      BR     50001$
7418 015366      50005$:
7419 ;
7420 ; IF NO ERROR, DO NORMAL EXIT
7421 ;
7422 015366 012601      MOV    (SP)+,R1
7423 015370 005267 000022      INC    INIT
7424 015374      50000$:
(2) 015374 000241      CLC
(3) 015376      50001$:
(3) 015376 010406      MOV    R4,SP
(3) 015400 012604      MOV    (SP)+,R4
(2) 015402 000207      RTS    PC
7571 ;
7572 ;
7573 015404 000000  XMTADD: 0 ; TRANSMIT BUFFER REGISTER ADDRESS
7574 015406 000000  RCVADD: 0 ; RECEIVE BUFFER REGISTER ADDRESS
7575 015410 000000  TIMER: 0 ; TIMER FOR CLEAR TO SEND OPERATION
7576 015412 000000  SYNCNT: 0 ; NUMBER OF SYNC CHARACTERS INSERTED BEFORE DDCMP MESSAGE
7577 015414 000000  FLLCNT: 0 ; NUMBER OF FILL CHARACTERS AFTER MESSAGE
7578 015416 000000  INIT: 0 ; INITIALIZE COMPLETE FLAG

```

7579	015420	000377	FIL:	377	; FILL CHARACTER
7580	015422	000000	XBUFFER:	0	; CURRENT TRANSMIT BUFFER POINTER
7581	015424	000000	XSIZE:	0	; CURRENT TRANSMIT BUFFER SIZE
7582	015426	000000	NMSG:	0	; NEW MESSAGE EXPECTED FLAG
7583	015430	000000	RTEMP1:	0	; CURRENT RECEIVE BUFFER POINTER
7584	015432	000000	RTEMP2:	0	; CURRENT RECEIVE BUFFER SIZE
7585	015434	000000	CNTFLG:	0	; CONTROL FLAG FOR INPUTTING DDCMP MESSAGES
7586			;		
7587			;	RECEIVER QUEUES	
7588			;		
7589	015436	000000	RQUEUE:	0	; QUEUE POINTER FOR BUFFER PROCESSING
7590	015440	000000	RQUE:	0	; QUEUE POINTER FOR NEXT AVAILABLE BUFFER
7591	015442	000000	RQUET:	0	; QUEUE OF BUFFER ADDRESSES AND SIZES
7592	015444	000000		0	
7593	015446	000000		0	
7594	015450	000000		0	
7595	015452	000000		0	
7596	015454	000000	RQUEB:	0	
7597	015456	000000	RBQUE:	0	; QUEUE POINTER FOR NEXT AVAILABLE BUFFER AREA
7598	015460	000000	RBQUET:	0	; QUEUE OF BUFFER AREAS
7599	015462	000000		0	
7600	015464	000000	RBQUEB:	0	


```

7602
7603          ;          TRANSMISSION INTERRUPT HANDLER
7604          ;
7607          ;
7608          ; SET PROCESSOR PRIORITY LEVEL
7609          ;
7610 015466 016700 164506          MOV    DEVPRI,RO
      (3) 015472 104041          EMT    C$SPRI
7611          ;
7612          ; IF SYNC COUNT EQUAL TO ZERO...
7613          ;
7614 015474 005767 177712          TST    SYNCNT
      (9) 015500 001047          BNE    50006$
7615          ;
7616          ; CHECK BUFFER SIZE.
7617          ; IF GREATER THAN ZERO, THEN...
7618          ;
7619 015502 010146          MOV    R1,-(SP)
7620 015504 005767 177714          TST    XSIZE
      (9) 015510 003416          BLE    50007$
7621          ;
7622          ; SEND THE NEXT CHARACTER AND UPDATE POINTER AND COUNT
7623          ;
7624 015512 117701 177704          MOVB  @XBUFFER,R1
7625 015516 042701 177400          BIC   #HYBYTE,R1
7628 015522 042777 000400 177654          BIC   #TSOM,@XMTADD
7631 015530 110177 177650          MOVB  R1,@XMTADD
7632 015534 005267 177662          INC   XBUFFER
7633 015540 005367 177660          DEC   XSIZE
7634          ;
7635          ; IF ALL CHARACTERS SENT, CHECK FILL COUNT.
7636          ;
7637 015544 000423          BR    50010$
      (3) 015546          50007$:
7638 015546 005767 177642          TST   FLLCNT
      (9) 015552 001406          BEQ   50011$
7639          ;
7640          ; IF FILL COUNT ISN'T ZERO SEND A FILL CHAR. AND DECREMENT COUNT
7641          ;
7642 015554 056777 177640 177622          BIS   FIL,@XMTADD
7643 015562 005367 177626          DEC   FLLCNT
7644 015566 000412          BR    50012$
      (3) 015570          50011$:
7645          ;
7646          ; IF ALL TRANSMIT DONE, DISABLE TRANSMISSION AND SET DONE FLAG
7647          ;
7648 015570 016701 177610          MOV   XMTADD,R1
      (6) 015574 162701 000006          SUB   #6,R1
7649 015600 005267 177132          INC   XDONE
7650 015604 104021          EMT   C$ABRT
7657 015606 042761 000120 000004          BIC   #TXINTE!SENDFD,XMTCSR(R1)
7660 015614          50012$:
7661 015614          50010$:
7662 015614 012601          MOV   (SP)+,R1
7663          ;
7664          ; ... OTHERWISE, SEND A SYNC AND DECREMENT THE SYNC COUNT

```

```

7665
7666 015616 000405          BR      50013$
      (3) 015620          50006$:
7669 015620 052777 000626 177556  BIS    #SYNC!TSOM,@XMTADD
7675 015626 005367 177560          DEC    SYNCNT
7676 015632          50013$:
7677
7678          ; DO A SUPERVISOR-TYPE RTI
7679
7680 015632          L10027:
      (2) 015632 000002          RTI
7682
7683
7684
7685
7686
7687
7688          ;
7689          ; RECEPTION INTERRUPT ROUTINE
7690          ;
7691          ; SET PROCESSOR PRIORITY
7692          ;
7693          ;
7694 015634 016700 164340      MOV    DEVPRI,R0
      (3) 015640 104041      EMT    C$SPRI
7695 015642 010146          MOV    R1,-(SP)
7696
7697          ; IF NEW MESSAGE FLAG IS RESET...
7698          ;
7699 015644 005767 177556      TST    NMSG
      (9) 015650 001111      BNE    50014$
7700
7701          ; OBTAIN A BUFFER FROM THE QUEUE AND GET THE RECEIVED BYTE.
7702          ;
7703 015652 017767 177600 177550  MOV    @RBQUE,RTEMP1
7704 015660 016777 177544 177552  MOV    RTEMP1,@RQUE
7705 015666 117777 177514 177534  MOVB  @RCVADD,@RTEMP1
7706 015674 117701 177530          MOVB  @RTEMP1,R1
7707
7708          ; CHECK FOR SYNC/FILL CHARACTER
7709          ; (LOOKING FOR FIRST NON-SYNC/NON-FILL)
7710          ;
7711 015700 120127 000226      CMPB  R1,#SYNC
      (9) 015704 001472      BEQ    50015$
      (6) 015706 120167 177506  CMPB  R1,FIL
      (9) 015712 001467      BEQ    50015$
7712
7713          ; NOT FILL OR SYNC. CHECK FOR ENQ (DDCMP CONTROL MESSAGE).
7714          ;
7715 015714 120127 000005      CMPB  R1,#ENQ
      (9) 015720 001027      BNE    50016$
7716
7717          ; IF IT IS AN ENQ, SET EXPECTED BYTE COUNT TO 7, SET MESSAGE SIZE
7718          ; TO 10 (OCTAL) AND UPDATE THE BUFFER QUEUES
7719          ;
7720 015722 012767 000007 177502  MOV    #7,RTEMP2
7721 015730 062767 000002 177502  ADD    #2,RQUE
  
```

```

7722 015736 012777 000010 177474      MOV      #10, RQUE
7723 015744 062767 000002 177466      ADD      #2, RQUE
7724 015752 026727 177462 015454      CMP      RQUE, #RQUEB
(9) 015760 003403                      BLE      50017$
7725 015762 012767 015442 177450      MOV      #RQUET, RQUE
7726 015770                      50017$:
7727                      ;
7728                      ; SET COUNTER FLAG TO -1 (MESSAGE SIZE COUNT COMPLETE)
7729                      ;
7730 015770 012767 177777 177436      MOV      #-1, CNTFLG
7731 015776 000417                      BR       50020$
(3) 016000                      50016$:
7732                      ;
7733                      ; NOT FILL, SYNC OR ENQ. CHECK FOR SOH OR DLE (DDCMP DATA MESSAGE)
7734                      ;
7735 016000 120127 000201      CMPB     R1, #SOH
(8) 016004 001403                      BEQ      50021$
(6) 016006 120127 000220      CMPB     R1, #DLE
(9) 016012 001007                      BNE      50022$
(6) 016014                      50021$:
7736                      ;
7737                      ; MOVE QUEUE POINTER TO SIZE ENTRY AND SET COUNTER FLAG TO 1 (EXPECT
7738                      ; FIRST COUNT BYTE)
7739                      ;
7740 016014 062767 000002 177416      ADD      #2, PQUE
7741 016022 012767 000001 177404      MOV      #1, CNTFLG
7742 016030 000402                      BR       50023$
(3) 016032                      50022$:
7743                      ;
7744                      ; BYTE WASN'T ANY OF THE LEGAL CHARACTERS... FORCE
7745                      ; A RESTART BY FORCING THE NEW MESSAGE FLAG TO RESET ON EXIT.
7746                      ;
7747 016032 005367 177370      DEC      NMSG
7748 016036                      50023$:
7749 016036                      50020$:
7750                      ;
7751                      ; COUNT THE CHARACTER AND UPDATE THE FREE BUFFER QUEUE
7752                      ;
7753 016036 005267 177366      INC      RTEMP1
7754 016042 062767 000002 177406      ADD      #2, RBQUE
7755 016050 026727 177402 015464      CMP      RBQUE, #RBQUEB
(9) 016056 003403                      BLE      50024$
7756 016060 012767 015460 177370      MOV      #RBQUET, RBQUE
7757 016066                      50024$:
7758                      ;
7759                      ; SET THE NEW MESSAGE FLAG
7760                      ; EXIT ROUTINE
7761                      ;
7762 016066 005267 177334      INC      NMSG
7763 016072                      50015$:
7764                      ;
7765                      ; NEW MESSAGE FLAG IS SET... CONTINUE BUILDING MESSAGE
7766                      ;
7767 016072 000513                      BR       50025$
(3) 016074                      50014$:
7768                      ;
  
```

```

7769 ;NOT A NEW MESSAGE...GET BYTE AND STORE IT
7770 ;
7771 016074 117777 177306 177326      MOVB   @RCVADD,@RTEMP1
7772 ;
7773 ;      IF COUNTER FLAG IS GREATER THAN ZERO, BYTE IS ALSO THE FIRST COUNT BYTE
7774 ;
7775 016102 005767 177326      TST    CNTFLG
(9) 016106 003413      BLE    500265
7776 ;
7777 ;      PUT THE BYTE IN RTEMP2 (EXPECTED COUNT), UPDATE THE BUFFER POINTER
7778 ;      DECREMENT THE COUNTER FLAG (SET TO GET SECOND COUNT BYTE) AND
7779 ;      GO DO AN RTI
7780 ;
7781 016110 117767 177314 177314      MOVB   @RTEMP1,RTEMP2
7782 016116 042767 177400 177306      BIC    #HYBYTE,RTEMP2
7783 016124 005267 177300      INC    RTEMP1
7784 016130 005367 177300      DEC    CNTFLG
7785 016134 000472      BR     500275
(3) 016136      500265:
7786 ;
7787 ;      COUNTER FLAG NOT GREATER THAN ZERO, IS IT ZERO?
7788 ;
7789 016136 005767 177272      TST    CNTFLG
(9) 016142 001041      BNE    500305
7790 ;
7791 ;      IF ZERO, BYTE IS ALSO SECOND BYTE OF COUNT (6 HIGH ORDER BITS)
7792 ;      GET IT, MASK IT AND PUT IT INTO RTEMP2 (EXPECTED COUNT)
7793 ;
7794 016144 010146      MOV    R1,-(SP)
7795 016146 117701 177256      MOVB   @RTEMP1,R1
7796 016152 005267 177252      INC    RTEMP1
7797 016156 000301      SWAB  R1
7798 016160 042701 140377      BIC    #140377,R1
7799 016164 060167 177242      ADD    R1,RTEMP2
7800 016170 012601      MOV    (SP)+,R1
7801 ;
7802 ;      PUT THE COMPLETE SIZE IN THE QUEUE (EXPECTED COUNT +
7803 ;      12 (OCTAL) BYTES OF HEADER AND CRC)
7804 ;
7805 016172 016777 177234 177240      MOV    RTEMP2,@RQUE
7806 016200 062777 000012 177232      ADD    #12,@RQUE
7807 ;
7808 ;      ADJUST EXPECTED COUNT TO INCLUDE THE REMAINDER OF THE HEADER AND
7809 ;      THE DATA CRC
7810 ;
7811 016206 062767 000007 177216      ADD    #7,RTEMP2
7812 ;
7813 ;      UPDATE THE QUEUE AND DECREMENT THE COUNTER FLAG (COUNT COMPLETE)
7814 ;
7815 016214 062767 000002 177216      ADD    #2,RQUE
7816 016222 026727 177212 015454      CMP    RQUE,#RQUEB
(9) 016230 003403      BLE    500315
7817 016232 012767 015442 177200      MOV    #RQUET,RQUE
7818 016240      500315:
7819 016240 005367 177170      DEC    CNTFLG
7820 ;

```

```
7821      ; GO DO AN RTI
7822      ;
7823 016244 000426      ; BR      500325
(3) 016246      500305:
7824      ;
7825      ; COUNTER FLAG IS LESS THAN ZERO... THIS IS A DATA BYTE
7826      ; UPDATE POINTER AND COUNT
7827      ;
7828 016246 005267 177156      ; INC      RTEMP1
7829 016252 005367 177154      ; DEC      RTEMP2
7830      ;
7831      ; IF COUNT HAS REACHED ZERO, END OF MESSAGE. SET DONE FLAG.
7832 016256 005767 177150      ; TST      RTEMP2
(9) 016262 001017      ; BNE      500335
7833 016264 005267 176450      ; INC      RDONE
7835      ;
7836      ; FORCE DUP TO RESTART RECEPTION
7837      ;
7838 016270 016701 177112      ; MOV      RCVADD,R1
(6) 016274 162701 000002      ; SUB      #2,R1
7839 016300 042761 000020 000000      ; BIC      #RCENBL,RCVCSR(R1)
7840 016306 052761 000020 000000      ; BIS      #RCENBL,RCVCSR(R1)
7842      ;
7843      ; ABORT ANY WAITS AND RESET NEW MESSAGE FLAG
7844      ;
7845 016314 104021      ; EMT      CSABRT
7846 016316 005067 177104      ; CLR      NMSG
7847 016322      500335:
7848 016322      500325:
7849 016322      500275:
7850 016322      500255:
7851      ;
7852      ; DO RTI
7853      ;
7854 016322 012601      ; MOV      (SP)+,R1
7855 016324      L10030:
(2) 016324 000002      ; RTI
7856      ;
7857      ;
7858      ;
7859      ;
7860      ; ++++++
7861      ; ROUTINE TO GET AN INPUT BUFFER FROM THE INPUT QUEUES      +
7862      ; WILL RETURN AN ERROR IF TERMINAL DOES NOT RESPOND      +
7863      ; ARGUMENT (IF SET) INHIBITS ERROR PRINTOUT. THIS ALLOWS      +
7864      ; THE PROGRAM TO RETRY THE STRT-STACK SEQUENCE WITHOUT HAVING      +
7865      ; AN ERROR MESSAGE EACH TIME      +
7866      ; ++++++
7867      ;
7868      ;
7869      ;
7870      ;
7871 016326      RCVE:
7872      ;
7873      ; RESET TIMER
7874      ;
```

```

7875 016326 016767 161446 177054      MOV      O,TIMER
7876                                     ;
7877                                     ; GO TO SUPERVISOR FOR AWHILE AND INCREMENT TIMER
7878                                     ;
7879 016334                               50002$:
7880 016334 012700 000100                MOV      #100,R0
(3) 016340 104027                          EMT      C$WTU
7881 016342 005267 177042                INC      TIMER
7882                                     ;
7883                                     ; LOOP UNTIL TIME LIMIT EXCEEDED OR RECEIVER DONE FLAG IS SET
7884                                     ;
7885 016346 026767 177036 163534          CMP      TIMER,TIMLIM
(6) 016354 003003                          BGT      50003$
(4) 016356 005767 176356                TST      RDONE
(7) 016362 001764                          BEQ      50002$
(4) 016364                               50003$:
7886                                     ;
7887                                     ; RESET THE NEW MESSAGE FLAG
7888                                     ;
7889 016364 005067 177036                CLR      NMSG
7890                                     ;
7891                                     ; IF THE TIME LIMIT WASN'T EXCEEDED...
7892                                     ;
7893 016370 026767 177014 163512          CMP      TIMER,TIMLIM
(9) 016376 002044                          BGE      50004$
7894                                     ;
7895                                     ; RESET THE TIMER
7896                                     ; CHECK FOR ERROR (RDONE LESS THAN ZERO)
7897                                     ;
7898 016400 005067 177004                CLR      TIMER
7899 016404 005767 176330                TST      RDONE
(9) 016410 003426                          BLE      50005$
7900                                     ;
7901                                     ; NOT ERROR, GET BUFFER ADDRESS AND SIZE FROM QUEUE AND UPDATE QUEUE
7902                                     ;
7903 016412 017767 177020 163526          MOV      @RQUEUE,INBUF
7904 016420 062767 000002 177010          ADD      #2,RQUEUE
7905 016426 017767 177004 163514          MOV      @RQUEUE,INSIZ
7906 016434 062767 000002 176774          ADD      #2,RQUEUE
7907 016442 026727 176770 015454          CMP      RQUEUE,#RQUEB
(9) 016450 003403                          BLE      50006$
7908 016452 012767 015442 176756          MOV      #RQUET,RQUEUE
7909 016460                               50006$:
7910                                     ;
7911                                     ; RESET DONE FLAG AND LEAVE
7912                                     ;
7913 016460 005067 176254                CLR      RDONE
7914 016464 000410                          BR       50007$
(3) 016466                               50005$:
7915                                     ;
7916                                     ; ERROR DID OCCUR... RESET THE DONE FLAG, REPORT ERROR, RETURN WITH ERROR
7917                                     ;
7918 016466 005067 176246                CLR      RDONE
7919 016472 104463                          TRAP    T$ERCODE
(5) 016474 000032                          .WORD  26
(5) 016476 003062                          .WORD  MSG26

```

```

(5) 016500 004264
7920 016502 000261
(4) 016504 000443
7921 016506
7922 016506 000441
(3) 016510
7923
7924
7925
7926 016510 005067 176674
7927 016514 010146
7928 016516 016701 176664
(6) 016522 162701 000002
7929 016526 010246
7931 016530 012702 002224
7936 016534 016162 000000 000000
7937 016542 016162 000002 000002
7938 016550 016162 000004 000004
7939 016556 016162 000006 000006
7940 016564 012602
7941 016566 012601
7942
7943
7944
7945 016570 005765 000000
(9) 016574 001004
7946 016576 104463
(5) 016600 000037
(5) 016602 003147
(5) 016604 004266
7947 016606
7948
7949
7950
7951 016606 000261
(4) 016610 000401
7952 016612
7953 016612
(2) 016612 000241
(3) 016614
(2) 016614 000207
7954
7955
7956
7957
7958
7960
7961
7962
7963
7964
7965 016616
7966 016616
(3) 016616
(2) 016616 000207
8029

```

```

      .WORD AREA26
      SEC
      BR 50001$
50007$:
      BR 50010$
50004$:
;
; TIME LIMIT EXCEEDED... RESET TIMER, GET REGISTER CONTENTS
;

```

```

      CLR TIMER
      MOV R1, -(SP)
      MOV RCVADD, R1
      SUB #2, R1
      MOV R2, -(SP)
      MOV #DUPREG, R2
      MOV RCVCSR(R1), RCVCSR(R2)
      MOV RCVBUF(R1), RCVBUF(R2)
      MOV XMTCSR(R1), XMTCSR(R2)
      MOV XMTBUF(R1), XMTBUF(R2)
      MOV (SP)+, R2
      MOV (SP)+, R1

```

```

; IF NOT INHIBITED, PRINT ERROR MESSAGE
;

```

```

      TST DDSTRT(R5)
      BNE 50011$
      TRAP T$ERCODE
      .WORD 31
      .WORD MSG31
      .WORD AREA31

```

```

50011$:
;
; RETURN WITH ERROR IN ANY CASE
;

```

```

      SEC
      BR 50001$
50010$:
50000$:
      CLC
50001$:
      RTS PC

```

```

; ++++++
;
; DUMMY ROUTINE FOR COMPATIBILITY
;
; ++++++

```

```

INSERR:
50000$:
50001$:
      RTS PC

```

```

8030
8031
8032
8033
8034
8035
8036
8037
8038
8039
8040
8041
8042
8043
8044
8045
8046 016620
8049 016620 010446
      (2) 016622 010604
      (3) 016624 162706 000C04
8050 016630 010146
8051 016632 010246
8052 016634 010346
8053 016636 016501 000000
8054 016642 016502 000002
8055 016646 005003
8056 016650
      (6) 016650 005702
      (9) 016652 003500
8057 016654 112164 177776
8058 016660 042764 177400 177776
8059 016666 005302
8060 016670 010346
      (6) 016672 046416 177776
      (6) 016676 040364 177776
      (6) 016702 052664 177776
8061 016706 016464 177776 177774
8062 016714 042764 177760 177776
8063 016722 006364 177776
8064 016726 062764 017074 177776
8065 016734 017464 177776 177776
8066 016742 042764 177417 177774
8067 016750 006264 177774
      (7) 016754 006264 177774
      (7) 016760 006264 177774
8068 016764 062764 017074 177774
8069 016772 062764 000040 177774
8070 017000 017464 177774 177774
8071 017006 016446 177774
      (6) 017012 046416 177776
      (6) 017016 046464 177774 177776
      (6) 017024 052664 177776
8072 017030 042703 000377
8073 017034 00G303
8074 017036 016446 177776
      (6) 017042 040316
  
```

```

;+++++
;
; THIS ROUTINE COMPUTES CRC-16 AND REQUIRES THREE ARGUMENTS:
; DATA ADDRESS, DATA SIZE (IN BYTES) AND A WORD LOCATION
; FOR THE COMPUTED CRC
; 3. 2. 1. 2. 1, 3. 2. 1. 2. 4, 3. 2. 20. 1. 2, 3. 2. 20. 3. 1,
; 3. 2. 20. 6. 2, 3. 2. 20. 6. 4 AND 3. 2. 20. 10. 1
;
; THIS IS A STANDARD ROUTINE FROM DEC'S NETWORK GROUP
;
;+++++
  
```

```

CRC:
MOV R4, -(SP)
MOV SP, R4
SUB #4, SP
MOV R1, -(SP)
MOV R2, -(SP)
MOV R3, -(SP)
MOV A(R5), R1
MOV B(R5), R2
CLR R3
500025:
TST R2
BLE 500035
MOVB (R1)+, D(R4)
BIC #HYBYTE, D(R4)
DEC R2
MOV R3, -(SP)
BIC D(R4), (SP)
BIC R3, D(R4)
BIS (SP)+, D(R4)
MOV D(R4), E(R4)
BIC #177760, D(R4)
ASL D(R4)
ADD #CRCTAB, D(R4)
MOV @D(R4), D(R4)
BIC #177417, E(R4)
ASR E(R4)
ASR E(R4)
ASR E(R4)
ADD #CRCTAB, E(R4)
ADD #32, E(R4)
MOV @E(R4), E(R4)
MOV E(R4), -(SP)
BIC D(R4), (SP)
BIC E(R4), D(R4)
BIS (SP)+, D(R4)
BIC #LOBYTE, R3
SWAB R3
MOV D(R4), -(SP)
BIC R3, (SP)
  
```


(6)	017044	046403	177776	BIC	D(R4),R3
(6)	017050	052603		BIS	(SP)+,R3
8075	017052	000676		BR	50002\$
(3)	017054		50003\$:		
8076	017054	010365	000004	MOV	R3,C(R5)
8077	017060	012603		MOV	(SP)+,R3
8078	017062	012602		MOV	(SP)+,R2
8079	017064	012601		MOV	(SP)+,R1
8080	017066		50000\$:		
(3)	017066		50001\$:		
(3)	017066	010406		MOV	R4,SP
(3)	017070	012604		MOV	(SP)+,R4
(2)	017072	000207		RTS	PC
8081					
8082	017074	000000	CRCTAB:	000000	
8083	017076	140301		140301	
8084	017100	140601		140601	
8085	017102	000500		000500	
8086	017104	141401		141401	
8087	017106	001700		001700	
8088	017110	001200		001200	
8089	017112	141101		141101	
8090	017114	143001		143001	
8091	017116	003300		003300	
8092	017120	003600		003600	
8093	017122	143501		143501	
8094	017124	002400		002400	
8095	017126	142701		142701	
8096	017130	142201		142201	
8097	017132	002100		002100	
8098	017134	000000		000000	
8099	017136	146001		146001	
8100	017140	154001		154001	
8101	017142	012000		012000	
8102	017144	170001		170001	
8103	017146	036000		036000	
8104	017150	024000		024000	
8105	017152	162001		162001	
8106	017154	120001		120001	
8107	017156	066000		066000	
8108	017160	074000		074000	
8109	017162	132001		132001	
8110	017164	050000		050000	
8111	017166	116001		116001	
8112	017170	104001		104001	
8113	017172	042000		042000	

```

8117 ;
8118 ;   HARDWARE TESTS 3.2
8119 ;
8120 ;
8121 ; <><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>
8122 ;
8123 ;   SELF TEST 3.2C
8124 ;
8125 ;
8126 ; <><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>
8128 ;
8129 ; FORCE INIT OF COMM DEVICE AND LINK
8130 ;
8131 017174 005067 176216   CLR   INIT
8132 ;
8133 ; GET MESSAGE ADDRESS AND SIZE
8134 ;
8135 017200 016767 000744,162724   MOV   MDMSG,OUTMSG
      (6) 017206 042767 100001,162716   BIC   #100001,OUTMSG
8136 017214 016767 000732 162712   MOV   MDMSG+2,OUTMSG+2
8137 ;
8138 ; SEND THE MESSAGE
8139 ;
8140 017222 004767 173032   JSR   PC,TEROUT
8141 ;
8142 ; IF ERROR...
8143 ;
8144 017226 103015          BCC   50004$
8145 ;
8146 ;   INSERT ERROR INTO STATISTICS TABLE (MANUFACTURING VERSIONS ONLY),
8147 ;   REPORT THE ERROR, REQUEST SUPERVISOR TO LOOP
8148 ;   IF LOOPING ENABLED (EXIT TEST IF NO LOOPING)
8149 ;
8150 017230 010546          MOV   R5,-(SP)
      (4) 017232 012745 000000          MOV   #0,-(R5)
      (3) 017236 004767 177354          JSR   PC,INSERR
      (3) 017242 012605          MOV   (SP)+,R5
8151 017244 104463          TRAP  T$ERRCODE
      (5) 017246 000062          .WORD 50
      (5) 017250 003271          .WORD MSG50
      (5) 017252 004072          .WORD AREA40
8152 017254 104006          EMT   C$CLP1
8153 017256 104032          EMT   C$EXIT
      (3) 017260 000364          .WORD L10031-
8154 017262          50004$:
8155 ;
8156 ; LOOK FOR ACK
8157 ;
8158 017262 162705 000002   SUB   #1*2,R5
      (3) 017266 004767 174122   JSR   PC,TERIN
      (4) 017272 012567 000650   MOV   (R5)+,MCOMP
8159 ;
8160 ; IF ERROR... INSERT ERROR INTO TABLE (MANUFACTURING TEST ONLY)
8161 ; REPORT THE ERROR, REQUEST LOOP IF ENABLED (EXIT TEST
8162 ; IF NO LOOPING)
8163 ;
8164 017276 103015          BCC   50005$

```

```

8165 017300 010546      MOV    R5,-(SP)
      (4) 017302 012745 000000      MOV    #0,-(R5)
      (3) 017306 004767 177304      JSR    PC,INSERR
      (3) 017312 012605      MOV    (SP)+,R5
8166 017314 104463      TRAP   T$ERCODE
      (5) 017316 000062      .WORD 50
      (5) 017320 003271      .WORD MSG50
      (5) 017322 004072      .WORD AREA40
8167 017324 104006      EMT    C$CLP1
8168 017326 104032      EMT    C$EXIT
      (3) 017330 000314      .WORD L10031-
8169 017332      500055:
8170      ;
8171      ; IF MESSAGE WAS NOT AN ACK... ERROR
8172      ; REPORT ERROR, LOOP IF ENABLED, EXIT TEST IF NOT
8173      ;
8174 017332 005767 000610      TST    MCOMP
      (9) 017336 001407      BEQ    500065
8175 017340 104463      TRAP   T$ERCODE
      (5) 017342 000062      .WORD 50
      (5) 017344 003271      .WORD MSG50
      (5) 017346 004072      .WORD AREA40
8176 017350 104006      EMT    C$CLP1
8177 017352 104032      EMT    C$EXIT
      (3) 017354 000270      .WORD L10031-
8178 017356      500065:
8179      ;
8180      ; FORCE A RESTART ON THE LINK AND WAIT FOR SELFTTEST TO COMPLETE
8181      ;
8182 017356 005067 176034      CLR    INIT
8183 017362 012700 000400      MOV    #400,R0
      (3) 017366 104026      EMT    C$WTM
8184      ;
8185      ; SET UP MESSAGE ASKING FOR RESULTS AND SEND IT
8186      ;
8187 017370 016767 000564 162534      MOV    MDMSG1,OUTMSG
      (6) 017376 042767 100001 162526      BIC    #100001,OUTMSG
8188 017404 016767 000552 162522      MOV    MDMSG1+2,OUTMSG+2
8189 017412 004767 172642      JSR    PC,TEROUT
8190      ;
8191      ; IF ERROR... INSERT ERROR IN TABLE (MANUF. TESTS ONLY), REPORT ERROR,
8192      ; LOOP IF ENABLED, EXIT TEST IF NO LOOPING
8193      ;
8194 017416 103015      BCC    500075
8195 017420 010546      MOV    R5,-(SP)
      (4) 017422 012745 000000      MOV    #0,-(R5)
      (3) 017426 004767 177164      JSR    PC,INSERR
      (3) 017432 012605      MOV    (SP)+,R5
8196 017434 104463      TRAP   T$ERCODE
      (5) 017436 000062      .WORD 50
      (5) 017440 003271      .WORD MSG50
      (5) 017442 004072      .WORD AREA40
8197 017444 104006      EMT    C$CLP1
8198 017446 104032      EMT    C$EXIT
      (3) 017450 000174      .WORD L10031-
8199 017452      500075:
    
```

8200				;	
8201				;	ACK TERMINAL AND GET ITS RESPONSE
8202				;	
8203	017452			50010\$:	
8204	017452	004767	000170		JSR PC, OUTREP
8205				;	
8206				;	HANDLE ERROR IF ONE OCCURED WHILE ACK'ING
8207				;	
8208	017456	103023			BCC 50011\$
8209	017460	010546			MOV R5, -(SP)
(4)	017462	012745	000000		MOV #0, -(R5)
(3)	017466	004767	177124		JSR PC, INSERR
(3)	017472	012605			MOV (SP)+, R5
8210	017474	010546			MOV R5, -(SP)
(4)	017476	012745	000001		MOV #1, -(R5)
(3)	017502	004767	177110		JSR PC, INSERR
(3)	017506	012605			MOV (SP)+, R5
8211	017510	104463			TRAP T\$ERCODE
(5)	017512	000062			. WORD 50
(5)	017514	003271			. WORD MSG50
(5)	017516	004072			. WORD AREA40
8212	017520	104006			EMT C\$CLP1
8213	017522	104032			EMT C\$EXIT
(3)	017524	000120			. WORD L19031-
8214	017526			50011\$:	
8215				;	
8216				;	GET THE RESPONSE
8217				;	
8218	017526	162705	000002		SUB #1*2, R5
(3)	017532	004767	173656		JSR PC, TERIN
(4)	017536	012567	000404		MOV (R5)+, MCOMP
8219				;	
8220				;	HANDLE ERROR IF ONE OCCURED.
8221				;	
8222	017542	103015			BCC 50012\$
8223	017544	010546			MOV R5, -(SP)
(4)	017546	012745	000000		MOV #0, -(R5)
(3)	017552	004767	177040		JSR PC, INSERR
(3)	017556	012605			MOV (SP)+, R5
8224	017560	104463			TRAP T\$ERCODE
(5)	017562	000062			. WORD 50
(5)	017564	003271			. WORD MSG50
(5)	017566	004072			. WORD AREA40
8225	017570	104006			EMT C\$CLP1
8226	017572	104032			EMT C\$EXIT
(3)	017574	000050			. WORD L10031-
8227	017576			50012\$:	
8228				;	
8229				;	REPEAT THE LOOP UNTIL A MESSAGE IS RECEIVED
8230				;	
8231	017576	005767	000344		TST MCOMP
(6)	017602	001723			BEQ 50010\$
8232				;	
8233				;	PROCESS THE MESSAGE RECEIVED
8234				;	
8235	017604	004767	000146		JSR PC, STINPT

8236										
8237										; IF IN ERROR... INSERT ERROR INTO TABLE (MANUF. TESTS ONLY, REPORT
8238										; ERROR, LOOP IF ENABLED, EXIT TEST IF NOT
8239										
8240	017610	103015						BCC	50013\$	
8241	017612	010546						MOV	R5, -(SP)	
(4)	017614	012745	000001					MOV	#1, -(R5)	
(3)	017620	004767	176772					JSR	PC, INSERR	
(3)	017624	012605						MOV	(SP)+, R5	
8242	017626	104463						TRAP	T\$ERCODE	
(5)	017630	000062						.WORD	50	
(5)	017632	003271						.WORD	MSG50	
(5)	017634	004072						.WORD	AREA40	
8243	017636	104006						EMT	C\$CLP1	
8244	017640	104032						EMT	C\$EXIT	
(3)	017642	000002						.WORD	L10031-	
8245	017644									50013\$:
8246										;
8247										; END OF TEST
8248										;
8249	017644									L10031:
(3)	017644	104001						EMT	C\$ETST	
8250										
8251										
8252										
8253										
8254										
8255										*****
8256										ROUTINE TO SEND AND ACK IN ORDER TO POLL A TERMINAL
8257										*****
8258										
8259										OUTREP:
8260	017646									;
8261										; INSERT MESSAGE NUMBER (LAST MESSAGE SENT) AND RESPONSE NUMBER
8262										; (LAST MESSAGE RECEIVED)... INSERT STATION ADDRESS
8263										
8264										
8265	017646	116767	162312	000320				MOVB	MSGNO, MSGN	
8266	017654	116767	162306	000311				MOVB	RMSGNO, REPN	
8267	017662	116767	163617	000305				MOVB	STRTSA, REPA	
8268										
8269										; COMPUTE AND INSERT CRC
8270										
8271	017670	162705	000002					SUB	#1*2, R5	
(3)	017674	010546						MOV	R5, -(SP)	
(5)	017676	012745	000006					MOV	#6, -(R5)	
(4)	017702	012745	020170					MOV	#REPM, -(R5)	
(3)	017706	004767	176706					JSR	PC, CRC	
(3)	017712	012605						MOV	(SP)+, R5	
(4)	017714	012567	000256					MOV	(R5)+, REPCRC	
8272										
8273										; SET UP BUFFER POINTER AND SIZE AND SEND THE MESSAGE
8274										
8275	017720	012767	020170	175474				MOV	#REPM, XBUFFER	
8276	017726	012767	000010	175470				MOV	#10, XSIZE	
8277	017734	004767	174466					JSR	PC, XMIT	

```
8278 ;
8279 ; IF AN ERROR OCCURRED, RETURN WITH ERROR. OTHERWISE RETURN NORMALLY
8280 ;
8281 017740 103004          BCC    50002$
8282 017742 005267 162152  INC    OUTERR
8283 017746 000261          SEC
      (4) 017750 000401          BR    50001$
8284 017752          50002$:
8285 017752          50000$:
      (2) 017752 000241          CLC
      (3) 017754          50001$:
      (2) 017754 000207          RTS    PC
8286
8287
8288
8289
8290
8291 ; ++++++
8292 ; ROUTINE TO PROCESS TERMINAL SELFTEST RESULTS +
8293 ; ++++++
8294 ;
8295 ; ++++++
8296 017756          STINPT:
8297 017756 010146          MOV    R1, -(SP)
8298 017760 010246          MOV    R2, -(SP)
8299 017762 010346          MOV    R3, -(SP)
8300
8301 ; SKIP RESPONSE HEADER AND CHECK FIRST DATA BYTE
8302 ;
8303 017764 016701 162156  MOV    INBUF, R1
      (6) 017770 062701 000004  ADD    #4, R1
8304
8305 ; IF IT IS "A", GOOD RESPONSE. OTHERWISE .
8306 ;
8307 017774 121127 000101  CMPB   (R1), #101
      (9) 020000 001455          BEQ    50002$
8308
8309 ; GET THE BYTE AND CHECK TO SEE IF IT IS "M" (MEMORY ERROR)
8310 ;
8311 020002 112103          MOVB   (R1)+, R3
8312 020004 120327 000115  CMPB   R3, #115
      (9) 020010 001026          BNE    50003$
8313
8314 ; MEMORY ERROR... GET THE TYPE AND BANK NUMBER
8315 ; INSERT INTO ERROR MESSAGE, REPORT THE ERROR AND
8316 ; RETURN WITH ERROR
8317 ; (NOTE: ERROR REPORTING PROTOCOL DELIMITERS ARE
8318 ; IGNORED. THIS PROGRAM WILL NOT ACCEPT INPUT FROM
8319 ; ANY OTHER INTELLIGENT TERMINAL PROPERLY.)
8320 ;
8321 020012 062701 000002  ADD    #2, R1
8322 020016 012702 003446  MOV    #MTY, R2
8323 020022 112122          MOVB   (R1)+, (R2)+
8324 020024 112122          MOVB   (R1)+, (R2)+
8325 020026 012702 003462  MOV    #BNO, R2
8326 020032 062701 000003  ADD    #3, R1
```

8327	020036	111103		MOVB	(R1),R3
8328	020040	062703	000060	ADD	#60,R3
8329	020044	110312		MOVB	R3,(R2)
8330	020046	104423		TRAP	T\$ERCODE
(5)	020050	000107		.WORD	71
(5)	020052	003422		.WORD	MSG71
8331	020054	012603		MOV	(SP)+,R3
8332	020056	012602		MOV	(SP)+,R2
8333	020060	012601		MOV	(SP)+,R1
8334	020062	000261		SEC	
(4)	020064	000427		BR	50001\$
8335	020066				
8336				50003\$:	
8337				:	NOT A MEMORY ERROR OR PASS... CHECK FOR "P" (PROCESSOR ERROR)
8338				:	
8339	020066	120327	000120	CMPB	R3,#120
(9)	020072	001010		BNE	50005\$
8340				:	
8341				:	PROCESSOR ERROR... REPORT IT AND EXIT WITH ERROR
8342				:	
8343	020074	104423		TRAP	T\$ERCODE
(5)	020076	000110		.WORD	72
(5)	020100	003464		.WORD	MSG72
8344	020102	012603		MOV	(SP)+,R3
8345	020104	012602		MOV	(SP)+,R2
8346	020106	012601		MOV	(SP)+,R1
8347	020110	000261		SEC	
(4)	020112	000414		BR	50001\$
8348	020114			50005\$:	
8349				:	RESPONSE WAS NOT LEGAL... REPORT IT AND RETURN WITH ERROP
8350				:	
8351				:	
8352	020114	104423		TRAP	T\$ERCODE
(5)	020116	000106		.WORD	70
(5)	020120	003367		.WORD	MSG70
8353	020122	012603		MOV	(SP)+,R3
8354	020124	012602		MOV	(SP)+,R2
8355	020126	012601		MOV	(SP)+,R1
8356	020130	000261		SEC	
(4)	020132	000404		BR	50001\$
8357	020134			50006\$:	
8358	020134			50004\$:	
8359	020134			50002\$:	
8360	020134	012603		MOV	(SP)+,R3
8361	020136	012602		MOV	(SP)+,R2
8362	020140	012601		MOV	(SP)+,R1
8363	020142			50000\$:	
(2)	020142	000241		CLC	
(3)	020144			50001\$:	
(2)	020144	000207		RTS	PC
8364				:	
8365				:	MESSAGE AREA FOR SELFTEST
8366				:	
8367				:	
8368	020146	000000		MCOMP:	0
8369	020150	020154		MDMSG:	+4

8445	020264	012701	021066		MOV	#LOOPB,R1
8446	020270	005767	171150		TST	PATTERN
(9)	020274	001403			BEQ	500105
8447	020276	005067	161600		CLR	SCRATCH
8448	020302	000403			BR	500115
(3)	020304			500105:		
8449	020304	012767	125125	161570	MOV	#125125,SCRATCH
8450	020312			500115:		
8451	020312			500125:		
8452	020312	016721	161564		MOV	SCRATCH,(R1)+
8453	020316	020127	021466		CMP	R1,#ELOOPB
(6)	020322	001373			BNE	500125
8454						
8455					:	PUT A 1 AT THE END TO ASSURE THAT THE FINAL BYTE IS NOT NULL (0)
8456					:	
8457	020324	012701	021466		MOV	#ELOOPB,R1
(6)	020330	005301			DEC	R1
8458	020332	112711	000001		MOVB	#1,(R1)
8459					:	
8460					:	GET THE STATION ADDRESS AND INSERT IT IN THE HEADER
8461					:	
8462	020336	012701	002170		MOV	#PTABLE,R1
(6)	020342	062701	000005		ADD	#5,R1
8463	020346	111167	000511		MOVB	(R1),LOOSTA
8464	020352	012601			MOV	(SP)+,R1
8465					:	
8466					:	INSERT LOOP DATA TYPE
8467					:	
8468	020354	112767	000024	000504	MOVB	#OSOP,LOOPB
8469					:	
8470					:	SET MESSAGE NUMBER
8471					:	
8472	020362	112767	000001	000472	MOVB	#1,LOOPN
8473					:	
8474					:	COMPUTE AND INSERT HEADER CRC
8475					:	
8476	020370	162705	000002		SUB	#1*2,R5
(3)	020374	010546			MOV	R5,-(SP)
(5)	020376	012745	000006		MOV	#6,-(R5)
(4)	020402	012745	021056		MOV	#LOOPH,-(R5)
(3)	020406	004767	176206		JSR	PC,CRC
(3)	020412	012605			MOV	(SP)+,R5
(4)	020414	012567	000444		MOV	(R5)+,LCRCH
8477					:	
8478					:	COMPUTE AND INSERT DATA CRC
8479					:	
8480	020420	162705	000002		SUB	#1*2,R5
(3)	020424	010546			MOV	R5,-(SP)
(5)	020426	012745	000400		MOV	#256,-(R5)
(4)	020432	012745	021066		MOV	#LOOPB,-(R5)
(3)	020436	004767	176156		JSR	PC,CRC
(3)	020442	012605			MOV	(SP)+,R5
(4)	020444	012567	001016		MOV	(R5)+,LCRCD
8481					:	
8482					:	SET UP BUFFER POINTER AND SIZE... SEND MESSAGE
8483					:	

```

8484 020450 012767 021056 174744      MOV  #LOOPH,XBUFFER
8485 020456 012767 000412 174740      MOV  #266.,XSIZE
8486 020464 004767 173736      JSR  PC,XMIT
8487                                     ;
8488                                     ; IF ERROR OCCURRED... INSERT THE ERROR INTO TABLE (MANUF. TESTS ONLY),
8489                                     ; REPORT THE ERROR, LOOP IF ENABLED
8490                                     ;
8491 020470 103012      BCC  50013$
8492 020472 010546      MOV  R5,-(SP)
   (4) 020474 012745 000000      MOV  #0,-(R5)
   (3) 020500 004767 176112      JSR  PC,INSERR
   (3) 020504 012605      MOV  (SP)+,R5
8493 020506 104463      TRAP T$ERCODE
   (5) 020510 000063      .WORD 51
   (5) 020512 003326      .WORD MSG51
   (5) 020514 004072      .WORD AREA40
8494 020516      50013$:
8495 020516 104006      EMT  C$CLP1
8496                                     ;
8497                                     ; SEND ACK'S TO TERMINAL UNTIL IT RESPONDS
8498                                     ;
8499 020520      50014$:
8500 020520 004767 177122      JSR  PC,OUTREP
8501                                     ;
8502                                     ; IF AN ERROR OCCURS... INSERT IN TABLE (MAN. ONLY), REPORT IT, LOOP
8503                                     ; IF ENABLED - EXIT TEST IF NOT
8504                                     ;
8505 020524 103023      BCC  50015$
8506 020526 010546      MOV  R5,-(SP)
   (4) 020530 012745 000000      MOV  #0,-(R5)
   (3) 020534 004767 176056      JSR  PC,INSERR
   (3) 020540 012605      MOV  (SP)+,R5
8507 020542 010546      MOV  R5,-(SP)
   (4) 020544 012745 000002      MOV  #2,-(R5)
   (3) 020550 004767 176042      JSR  PC,INSERR
   (3) 020554 012605      MOV  (SP)+,R5
8508 020556 104463      TRAP T$ERCODE
   (5) 020560 000063      .WORD 51
   (5) 020562 003326      .WORD MSG51
   (5) 020564 004072      .WORD AREA40
8509 020566 104006      EMT  C$CLP1
8510 020570 104032      EMT  C$EXIT
   (3) 020572 000134      .WORD L10032-
8511 020574      50015$:
8512                                     ;
8513                                     ; GET RESPONSE
8514                                     ;
8515 020574 162705 000002      SUB  #1*2,R5
   (3) 020600 004767 172610      JSR  PC,TERIN
   (4) 020604 012567 000240      MOV  (R5)+,LCOMP
8516                                     ;
8517                                     ; IF ERROR OCCURS... INSERT IN TABLE (MAN. ONLY), REPORT IT, LOOP IF
8518                                     ; ENABLED, EXIT TEST IF NOT
8519                                     ;
8520 020610 103012      BCC  50016$
8521 020612 010546      MOV  R5,-(SP)
  
```

(4) 020614 012745 000000
(3) 020620 004767 175772
(3) 020624 012605
8522 020626 104463
(5) 020630 000063
(5) 020632 003326
(5) 020634 004072
8523 020636
8524 020636 104006
8525
8526
8527
8528 020640 005767 000204
(6) 020644 001725
8529
8530
8531
8532 020646 004767 000056
8533 020652 104004
8534
8535
8536
8537 020654 004767 171634
8538
8539
8540
8541
8542 020660 103020
8543 020662 010546
(4) 020664 012745 000000
(3) 020670 004767 175722
(3) 020674 012605
8544 020676 010546
(4) 020700 012745 000002
(3) 020704 004767 175706
(3) 020710 012605
8545 020712 104463
(5) 020714 000063
(5) 020716 003326
(5) 020720 004610
8546 020722
8547 020722 104006
8548 020724
(3) 020724 104005
8549
8550
8551
8552 020726
(3) 020726 104001
8553
8554
8555
8556
8557
8558
8559

```
MOV #0, -(R5)
JSR PC, INSERR
MOV (SP)+, R5
TRAP T$ERCODE
  WORD 51
  WORD MSG51
  WORD AREA40
500165: EMT C$CLP1
; REPEAT LOOP UNTIL TERMINAL SENDS LOOPBACK DATA
;
TST LCOMP
BEQ 500145
; PROCESS THE LOOPBACK DATA
;
JSR PC, PROLOP
EMT C$BSEG
; REINIT THE LINK
;
JSR PC, XSTRT
; IF ERROR OCCURS, INSERT ERROR INTO TABLE (MANUF. ONLY), REPORT ERROR,
; LOOP ON REINIT IF ERROR LOOPING ENABLED... EXIT TEST IF NO LOOPING
;
BCC 500175
MOV R5, -(SP)
MOV #0, -(R5)
JSR PC, INSERR
MOV (SP)+, R5
MOV R5, -(SP)
MOV #2, -(R5)
JSR PC, INSERR
MOV (SP)+, R5
TRAP T$ERCODE
  WORD 51
  WORD MSG51
  WORD AREA61
500175: EMT C$CLP1
100005: EMT C$ESEG
; END OF TEST
;
L10032: EMT C$ETST
```

*

```

8560      ;          ROUTINE TO PROCESS LOOPBACK DATA          +
8561      ;          +
8562      ;          +
8563      ;          +
8564      ;          +
8565      ;          +
8566      ;          +
8567      ;          +
8568      ;          +
8569      ;          +
8570      ;          +
8571      ;          +
8572      ;          +
8573      ;          +
8574      ;          +
8575      ;          +
8576      ;          +
8577      ;          +
8578      ;          +
8579      ;          +
8580      ;          +
8581      ;          +
8582      ;          +
8583      ;          +
8584      ;          +
8585      ;          +
8586      ;          +
8587      ;          +
8588      ;          +
8589      ;          +
8590      ;          +
8591      ;          +
8592      ;          +
8593      ;          +
8594      ;          +
8595      ;          +
8596      ;          +
8597      ;          +
8598      ;          +
8599      ;          +
8600      ;          +
8601      ;          +
8602      ;          +
8603      ;          +
8604      ;          +
8605      ;          +
8606      ;          +
    
```

```

ROUTINE TO PROCESS LOOPBACK DATA
+++++
PROLOP:
    MOV     R1, -(SP)
    MOV     R2, -(SP)
    MOV     R3, -(SP)
; R1 = POINTER TO EXPECTED DATA
; R2 = COUNT OF DATA EXPECTED
; R3 = POINTER TO DATA FROM TERMINAL
;
    MOV     #LOOPB, R1
    MOV     #128, R2
    MOV     INBUF, R3
; CHECK DATA BYTE-BY-BYTE...
50002$:
; IF ERROR... INSERT ERROR INTO TABLE (MANUF. ONLY), REPORT ERROR
; (INCLUDING RECEIVED/EXPECTED BYTES), RETURN WITH ERROR
;
    CMPB    (R1), (R3)
    BEQ     50003$
    MOV     (R1), A52A
    MOV     (R3), A52B
    MOV     R5, -(SP)
    MOV     #2, -(R5)
    JSR     PC, INSERR
    MOV     (SP)+, R5
    TRAP    T$ERRCODE
    .WORD   52
    .WORD   MSG52
    .WORD   AREA52
    MOV     (SP)+, R3
    MOV     (SP)+, R2
    MOV     (SP)+, R1
    SEC
    BR      50001$
50003$:
; UPDATE COUNT AND POINTERS
;
    INC     R1
    DEC     R2
    INC     R3
; REPEAT UNTIL ALL DATA HAS BEEN CHECKED
;
    TST     R2
    BNE     50002$
    MOV     (SP)+, R3
    MOV     (SP)+, R2
    MOV     (SP)+, R1
50000$:
    
```

```
(2) 021044 000241
(3) 021046
(2) 021046 000207
8607
8608
8609
8610
8611 021050 000000
8612 021052 121057
8613 021054 000410
8614 021056 220 000 201 LOOPH: . BYTE DLE, 0, 201, 0
      021061 000
8615 021062 000
8616 021063 000
8617 021064 000000
8618 021066 000200
8619 021466
8620 021466 000000
8622
```

CLC
50001\$:
RTS PC
;
; LOOPBACK MESSAGE AREA
;
LCOMP: 0
LOOPMG: . +100005
ELOOPB-2-
LOOPN: . BYTE 0
LOOSTA: . BYTE 0
LCRCH: 0
LOOPB: . BLKW 128.
ELOOPE:
LCRCD: 0

```

8877 ;
8878 ;
8879 ;
8880 021470 000022 . WORD L10033-L$HARD/2
8881 021472 000031 . WORD T$CODE
(4) 021474 021536 . WORD DEVICE
(4) 021476 000000 . WORD T$LOLIM
(4) 021500 177776 . WORD T$HILIM
8882 021502 001031 . WORD T$CODE
(4) 021504 021554 . WORD VECTOR
(4) 021506 000000 . WORD T$LOLIM
(4) 021510 000400 . WORD T$HILIM
8883 021512 002032 . WORD T$CODE
(4) 021514 021571 . WORD PRIOR
(4) 021516 000340 . WORD 340
(4) 021520 000000 . WORD T$LOLIM
(4) 021522 000007 . WORD T$HILIM
8884 021524 002032 . WORD T$CODE
(4) 021526 021614 . WORD STATION
(4) 021530 177400 . WORD 177400
(4) 021532 000000 . WORD T$LOLIM
(4) 021534 000377 . WORD T$HILIM
8895 . EVEN

```

```

(3) 021536 L10033:
8896
8898 021536 052504 020120 042101 DEVICE: .ASCIZ /DUP ADDRESS: /
      021544 051104 051505 035123
      021552 000040
8899 021554 052504 020120 042526 VECTOR: .ASCIZ /DUP VECTOR: /
      021562 052103 051117 020072
      021570 000
8900 021571 104 050125 041040 PRIOR: .ASCIZ /DUP BUS PRIORITY: /
      021576 051525 050040 044522
      021604 051117 052111 035131
      021612 000040
8913 021614 042524 046522 047111 STATION: .ASCIZ /TERMINAL ADDRESS: /
      021622 046101 040440 042104
      021630 042522 051523 020072
      021636 000
8955 021640 . EVEN
8956
8957 ;
8958 ;
8959 ;

```

```

8960 021640 003010 . WORD L10034-L$SOFT/2
8961 021642 000032 . WORD T$CODE
(4) 021644 021662 . WORD ALIMIT
(4) 021646 177777 . WORD 177777
(4) 021650 000000 . WORD T$LOLIM
(4) 021652 077777 . WORD T$HILIM
8962 021654 001130 . WORD T$CODE
(4) 021656 021731 . WORD PATTRN
(4) 021660 177777 . WORD 177777
8966 . EVEN
(3) 021662 L10034:
8967 021662 047510 020127 040515 ALIMIT: .ASCIZ /HOW MANY INITIATE TRIES BEFORE ABORT? /

```

	021670	054516	044440	044516
	021676	044524	052101	020105
	021704	051124	042511	020123
	021712	042502	047506	042522
	021720	040440	047502	052122
	021726	020077	000	
8968	021731	125	042523	040440
	021736	046114	055040	051105
	021744	023517	020123	047506
	021752	020122	047514	050117
	021760	040502	045503	042040
	021766	052101	037501	000040
8972				
8974				
(3)	021774			
8975				
8976				
20272		052674		
20274		071776		
20275	071776	000000		
20277		072000		
20278		000200		

PATRN: .ASCIZ /USE ALL ZERO'S FOR LOOPBACK DATA? /

.EVEN
.EVEN

LSLAST: :

.REQ DOCTOR. P11 400,1066
.TITLE PDP-11 DIAGNOSTIC SUPERVISOR
END. SUPV=. +2
.=71776
.WORD 0
X1X1=
.END 200

A = 000000	BIT06 = 000100 G	CODE 011454 G	CSQ10 = 000377	EF10 = 000012 G
ABOFLA 022246 G	BIT07 = 000200 G	COMMTA 045326	CSRDBU= 000007	EF11 = 000013 G
ABOPAS 022160 G	BIT08 = 000400 G	CONTCL 051110 G	CSREFG= 000050	EF12 = 000014 G
ABORT 002106	BIT09 = 001000 G	CRC 016620	CSREQT= 000045	EF13 = 000015 G
ABO.FM 025122	BIT1 = 000002 G	CRCTAB 017074	CSRESE= 000033	EF14 = 000016 G
ACK = 000001	BIT10 = 002000 G	CRLF 041424	CSREVI= 000001	EF15 = 000017 G
ALIMIT 021662	BIT11 = 004000 G	CTS = 020000	CSRPT = 000025	EF16 = 000020 G
ALLOC 042712	BIT12 = 010000 G	CURR. T 022174 G	CSSEFG= 000047	ELOOP8 021466
AREA00 003510 G	BIT13 = 020000 G	CSAAD 034530	CSSPRI= 000041	EMDMG1 020160
AREA01 003570 G	BIT14 = 040000 G	CSAAE 034542	CSSVEC= 000037	EMDMG2 020170
AREA02 003572 G	BIT15 = 100000 G	CSAAK 035360	CSTPRI= 000013	EMT. TR 022252 G
AREA03 003674 G	BIT2 = 000004 G	CSAAL 035470	CSUNBU= 000031	END. OF 030600
AREA04 003726 G	BIT3 = 000010 G	CSABRT= 000021	CSWTM = 000026	END. SU= 052674
AREA11 004030 G	BIT4 = 000020 G	CSADR = 000020	CSWTU = 000027	ENQ = 000005
AREA21 004066 G	BIT5 = 000040 G	CSAU = 000054	D = 177776	EOMSG 002124
AREA22 004070 G	BIT6 = 000100 G	CSBRK = 000022	DDSTRT= 000000	EOP. CH 051352 G
AREA23 004072 G	BIT7 = 000200 G	CSBSEG= 000004	DDSTRY 012772	EOP. FM 025136
AREA24 004072	BIT8 = 000400 G	CSBSUB= 000002	DECMG 041236	EOP. IN 027210
AREA26 004264 G	BIT9 = 001000 G	CSBUFF= 000030	DEVICE 021536	ERRFOR 035546
AREA30 004266 G	BLD. HW 027712	CSCEFG= 000046	DEVINI 014742	ERRHAN 034562
AREA31 004266	BLOCK 045046	CSCLEA= 000012	DEVPRI 002200	ERR. HR 035340
AREA40 004072	BNC 003462	CSCLP1= 000006	DIAG. T 022254 G	ERR. SF 035344
AREA41 003726	BNSAB 031314	CSCVEC= 000036	DLE = 000220	ERR1FO 035632
AREA42 003674	BNSAF 031226	CSDCLN= 000044	DPDVD 052040 G	ESC. PC 034554
AREA43 003726	C = 000004	CSDODU= 000053	DPMUL 051726 G	EXPINP 002130
AREA52 004740 G	CALLPC= 000022	CSDRPT= 000024	DTR = 000002	FIL 015420
AREA60 004532 G	CALLPS= 000024	CSDU = 000055	DUNIT. 022164 G	FILL 042072
AREA61 004610 G	CALLSP= 000026	CSEDIT= 000006	DUPMSK 002220 G	FILL. C 000204 G
AREA70 005042 G	CALLTC= 000030	CSERDF= 000002	DUPREG 002224 G	FLAGS 022216 G
AREA71 005044 G	CAL. CL 047430	CSERHR= 000003	DUPS = 000001	FLAGTA 045244
ASAAW 027032	CAL. TI 047466 G	CSERSF= 000001	DVC. FT 035330	FLAG. I 027256
ASAAZ 027046	CHKLUP 031330	CSERSO= 000004	DSAG 036200	FLA. SE 045212
ASAAZ 027054	CHKREP 014036	CSERSCA= 000010	DSAAH 036216	FLG. MA 027216
ASAAZ 027070	CHKSTR 043254	CSSESEG= 000005	DSAAI 040770	FLLCNT 015414
ASABA 027100	CHKTTY 041342	CSesub= 000003	DSAAJ 040774	FORM. T 035642
A02A 003624	CHK. FO 023540	CSETST= 000001	DSAAK 041012	FORM00 003536
A02B 003626	CHK. MA 027470	CSEXIT= 000032	DSAAL 041030	FORM02 003630
A04EX 003762	CHK. PC 034556	CSGMAN= 000043	DSAAAM 041040	FORM04 003764
A04RC 003760	CHK. SW 023262	CSGPHR= 000042	E = 177774	FORM23 004166
A52A 005036	CHRCNT 042574	CSGPRI= 000040	EF. CON= 000036 G	FORM30 004420
A52B 005040	CH. FLA 027172	CSGTIM= 000052	EF. NEW= 000035 G	FORM40 004206
B = 000002	CH. PAS 027214	CSINIT= 000011	EF. PWR= 000034 G	FORM52 004772
BAPNT 002160	CLEAR 030612	CSINLP= 000020	EF. RES= 000037 G	FORM60 004554
BASIZE 002162	CLKACC 022156 G	CSKWF= 000035	EF. STA= 000040 G	FORM61 004702
BGN. SU= 021774	CLKBFR 047432	CSKWON= 000034	EF01 = 000001 G	FREE 043150
BINMSG 041222	CLKCNT 022154 G	CSLOOP= 000100	EF02 = 000002 G	FRM30A 004442
BIT0 = 000001 G	CLKRES 051030 G	CSMANI= 000051	EF03 = 000003 G	FRM30B 004465
BIT00 = 000001 G	CLKSER 051330 G	CSMSG = 000023	EF04 = 000004 G	FRM30C 004507
BIT01 = 000002 G	CLKSON 022220 G	CSPNTB= 000014	EF05 = 000005 G	FSS = 000001
BIT02 = 000004 G	CLK. SE 027272	CSPNTF= 000017	EF06 = 000006 G	FSAU = 000015
BIT03 = 000010 G	CLR. MA 027546	CSPNTS= 000016	EF07 = 000007 G	F SBGN = 000040
BIT04 = 000020 G	CNTFLG 015434	CSPNTX= 000015	EF08 = 000010 G	FSCLEA= 000007
BIT05 = 000040 G	CNVT 045506	CSPOIN= 000040	EF09 = 000011 G	FSDU = 000016

FSEND = 000041	ININIT 022176 G	LUP 047334	L10007 004070	MSG24 002762
FSHARD= 000004	INIT 015416	LUP. AD 034560	L10010 004162	MSG25 003025
FSHW = 000013	INITIA 041252	LSAU 011456 G	L10011 004264	MSG26 003062
FSINIT= 000006	INITMA 002154	LSAUT 002070 G G	L10012 004416	MSG30 003117
FSJMP = 000050	INITMS 002156	LSCCP 002044 G	L10013 004552	MSG31 003147
FSMOD = 030000	INIT.M 027614	LSCLEA 012026 G	L10014 004700	MSG40 002637
FSMSG = 000011	INIT. R 022010 G	LSEDEP 002011 G	L10015 004770	MSG41 002411
FSPWR = 000017	INMSG 002136	LSEDEVP 002052 G	L10016 005042	MSG42 002345
FSRPT = 000012	INPUTA 042200	LSDISP 011450 G	L10017 005044	MSG43 003224
FSSEG = 000003	INSERR 016616	LSDR 002220 G	L10020 011440	MSG45 003243
FSSOFT= 000005	INSIZ 002150	LSDRCT 002062 G	L10021 011446	MSG50 003271
FSSRV = 000010	INTFOR 035476	LSDRS 002064 G	L10022 011454	MSG51 003326
FSSUB = 000002	INVAL. 026722	LSDRST 002224 G	L10023 011456	MSG52 003354
FSSW = 000014	INVINT 035370	LSDTP 002040 G	L10024 011460	MSG70 003367
FSTEST= 000001	INV. SW 023216	LSDU 011460 G	L10025 011734	MSG71 003422
GARBAG 042576	IN. SUF 030564	LSDUT 002072 G	L10026 012050	MSG72 003464
GETCHR 041302	LSAU = 000041	LSDVTY 002202 G	L10027 015632	MTY 003446
GETCMN 044666	LSCLN = 000041	LSEF 002024 G	L10030 016324	MUL 051574 G
GETOB 012774	LSDU = 000041	LSEXP1 002032 G	L10031 017644	NAK = 000002
GETPAR 036360	LSHRD = 000041	LSEXP2 002034 G	L10032 020726	NEWPRI 051320 G
GETSWI 043662	LSINIT= 000041	LSEXP3 002036 G	L10033 021536	NEXTAR 045430
GET. TH 043432	LSMOD = 000041	LSHARD 021472 G	L10034 021662	NMSG 015426
GLOB 002100 G	LSMSG = 000041	LSHPCP 002046 G	MAJ. IN 022000 G	NO. CLK 024332
GSUB 012052 G	LSPWR = 000041	LSHPTP 002056 G	MAJ. LO 047434	NO. FLA 045224
GSEXCP= 000400	LSRPT = 000041	LSHW 011430 G	MAJ. US 022002 G	NO. LPT 042542
GSHILI= 000002	LSSEG = 000041	LSICP 002042 G	MAN. TI 024302	NO. PTA 027022
GSOLOI= 000001	LSST = 000041	LSINIT 011462 G	MAP16 052276 G	NR = 000001
GSNO = 000000	LSRV = 000041	LSLADP 002076 G	MASCLR= 000400	NUMBIN 035666
GSOFFS= 000400	LSUB = 000041	LSLAST 021774 G	MASK. B 031326	NUM. LA 036034
GSOFSI= 000376	LSST = 000041	LSMREV 002012 G	MASK. W 031324	NUM. UN 022372
GSPRMA= 000001	JSJMP = 000167	LSNAME 002000 G	MCOMP 020146	NUNITS 031302
GSPRMD= 000002	KBPTR 022026 G	LSREPP 002054 G	MDMSG 020150	NXTFOR 045500
GSPRML= 000000	KBUF 022030 G	LSREV 002010 G	MDMSG1 020160	OCTMSG 041230
GSRADA= 000140	KEX 004064	LSRPT 011454 G	MEM. SI 026622	OSOP = 000024
GSRADB= 000000	KRC 004062	LSOFT 021642 G	MIN. IN 021774 G	OUT 002152
GSRADD= 000040	LCOMP 021050	LSPC 002030 G	MIN. US 021776 G	OUTBUF 002142
GSRADF= 000200	LCRCD 021466	LSPCP 002050 G	MODR 051640 G	OUTERR 002120
GSRADL= 000120	LCRCH 021064	LSPTP 002060 G	MSG 012052	OUTMSG 002132
GSRADO= 000020	LINE. F 022250 G	LSSTA 002066 G	MSGAP 002152	OUTREP 017646
GSRADT= 000100	LOAD. F 027212	LSW 011442 G	MSGN 020174	OUTSIZ 002144
GSXFER= 000004	LOBYTE= 000377	LSTIML 002022 G	MSGNO 002164	O\$APTS= 000001
GSYES = 000010	LOGDEV 002126	LSTIMU 002020 G	MSG00 002234	O\$AU = 000001
HCORED 026754	LOGMSG 041244	LSTIM1 002016 G	MSG01 002273	O\$BGNR= 000001
HCOREQ 026634	LOOPB 021066	LSTST1 002074 G	MSG02 002320	O\$BGN= 000001
HCORET 022206 G	LOOPH 021056	LSUNIT 002014 G	MSG03 002345	O\$DU = 000001
HEAD 002000 G	LOOPMG 021052	L. CLK. 026560	MSG04 002411	O\$GNSW= 000001
HERTZ 026574	LOOPN 021062	L10003 003534	MSG10 002436	O\$POIN= 000001
HOLDSP= 000020	LOOSTA 021063	L10001 003570	MSG11 002516	O\$PWR = 000001
HYBYTE= 177400	LPBFR 022024 G	L10002 003622	MSG12 002551	PARAM = 101226
H\$AAB 046034	LPCNTR 022022 G	L10003 003724	MSG20 002620	PARITY= 000200
ILIMIT 002114	LPT AD 026612	L10004 003756	MSG21 002637	PARSES 044740
INBUF 002146	LPT. RE 026606	L10005 004060	MSG22 002657	PAR. LA 040732
INERR 002116	LSI. RE 026602	L10006 004066	MSG23 002722	PASCNT 002122

\$TEMP = 000300	\$\$BYTE= 000402	\$\$FROM= 000000	\$\$RTN2= 050001	\$\$\$TAG= 050000
\$TSKO = 050002	\$\$CASE= 000000	\$\$LOC = 021034	\$\$SRC = 000064	= 072000
\$TSK1 = 050003	\$\$DST = 000003	\$\$LOCN= 000000	\$\$TGSV= 000000	
\$TSK2 = 050006	\$\$ELOC= 000402	\$\$REG = 177777	\$\$TGS1= 000000	
\$TSK3 = 050033	\$\$ERFL= 000000	\$\$RETU= 000001	\$\$TGS2= 000000	
\$\$ARGC= 000000	\$\$FLAG= 000001	\$\$RTN1= 050000	\$\$TO = 000001	

ABS. 072000 000

ERRORS DETECTED: 0

DZDPGA=DZDPGA
RUN-TIME: 93 95 .9 SECONDS
RUN-TIME RATIO: 1402/190=7.3
CORE USED: 35K (69 PAGES)