

DH11

DH11 MULTILINE DATA
CZDHGC0

AH-FG26C-MC
1 OF 1 OCT 1985
COPYRIGHT © 1972-85

digital
MADE IN USA

The image displays a grid of 180 small, illegible data plots or charts, arranged in 12 columns and 15 rows. Each plot appears to contain technical information, possibly related to digital signal processing or system performance, but the text is too small to read. The plots are organized in a structured, repeating pattern across the left side of the page.



.REM !

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

IDENTIFICATION

PRODUCT CODE: AC-8468C-MC
PRODUCT NAME: CZDHGCO DH11 MULTI-LINE DATA TEST
DATE: JUNE 1985
MAINTAINER: NAC SOFTWARE ENGINEERING
AUTHOR: G. BAISLEY

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

NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES

COPYRIGHT (C) 1985 BY DIGITAL EQUIPMENT CORPORATION

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51

1.0 ABSTRACT

THE SINGLE LINE PARITY TEST CHECKS THAT THE PARITY ERROR FLAG ASSERTS FOR ALL 16 LINES. . .ONE LINE AT A TIME.

THE DH11 MULTI-LINE DATA TEST TRANSMITS BINARY COUNT PATTERNS ON ALL 16 LINES SIMULTANEOUSLY. ALSO, PARITY GENERATION LOGIC AND PARITY RECEPTION LOGIC IS VERIFIED. (ODD & EVEN PARITY FOR 5, 6, 7, 8 LEVEL CODES)

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 FAMILY STANDARD COMPUTER WITH 4KW OF MEMORY
ASR-33 TELETYPE OR EQUIVALENT
DH11 ASYNCHRONOUS MULTIPLEXER
DM11 MAINTENANCE CARD INSTALLED

2.2 STORAGE

THE PROGRAM LOADS INTO 4KW OF MEMORY

3.0 LOADING PROCEDURE

THE STANDARD PROCEDURE FOR LOADING ABSOLUTE BINARY TAPES IS TO BE USED

4.0 STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

4.1.1 AFTER PROGRAM LOAD (INITIAL PROGRAM START)

ALL CONSOLE SWITCHES DOWN

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51

4.1.2 TO MODIFY DEVICE VECTOR AND CONTROL REGISTER ADDRESSES AFTER PROGRAM RESTART

SW00=1

4.1.3 TO START PROGRAM AT SELECTED TEST AFTER PROGRAM RESTART

SW01=1

4.2 STARTING ADDRESS

THE STARTING ADDRESS FOR ALL TESTS IS 000200
THE RESTART ADDRESS FOR ALL TESTS IS 0002000
THE STARTING ADDRESS TO ENTER A SELECTED TEST IS 000200

4.3 PROGRAM AND/OR OPERATOR ACTION

4.3.1 INITIAL PROGRAM START

4.3.1.1 LOAD PROGRAM INTO MEMORY

4.3.1.2 LOAD ADDRESS 000200

4.3.1.3 CLEAR CONSOLE SWITCHES

4.3.1.4 PRESS START

4.3.1.5 THE PROGRAM WILL TYPE "DH11 MULTI-LINE DATATEST" AND WILL THEN TYPE "VECTOR ADDRESS-" AND WAIT FOR AN INPUT FROM THE TELETYPE KEYBOARD.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51

4.3.1.6 TYPE IN THE ADDRESS OF THE RECEIVER INTERRUPT VECTOR FOR THE
DH11 TO BE TESTED FOLLOWED BY <CARRIAGE RETURN>

NOTE

WORDS IN ANGLE BRACKETS, I.E. <CARRIAGE
RETURN> MEAN THAT THE TELETYPE KEY WITH
THE NAMED FUNCTION SHOULD BE STRUCK

IF AN INCORRECT ADDRESS IS ENTERED, THE PROGRAM WILL TYPE "?" AND WILL
REPEAT THE SECOND MESSAGE OF 4.3.1.5

4.3.1.7 THE PROGRAM WILL TYPE "CONTROL REGISTER ADDRESS-" AND WAIT
FOR AN INPUT FROM THE TELETYPE KEYBOARD

4.3.1.8 TYPE IN THE ADDRESS OF THE SYSTEM CONTROL REGISTER OF THE
DH11 TO BE TESTED FOLLOWED BY <CARRIAGE RETURN>

IF AN INCORRECT ADDRESS IS TYPED, THE PROGRAM WILL TYPE "?" AND WILL
THEN REPEAT THE OF 4.3.1.7

4.3.1.9 THE PROGRAM WILL TYPE "R" TO INDICATE THAT IT IS ABOUT TO
START TESTING, AND THEN TESTING WILL BEGIN

4.3.2 PROGRAM RESTART WITH ALL SWITCHES DOWN

4.3.2.1 PERFORM 4.3.1.2 TO 4.3.1.5

4.3.2.2 THE PROGRAM WILL TYPE "DH11 MULTI-LINE DATATEST" AND WILL
THEN CONTINUE AS DESCRIBED IN 4.3.1.9

4.3.3 PROGRAM RESTART WITH SW00=1

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56

4.3.3.1 LOAD ADDRESS 000200

4.3.3.2 SET SW01=1

4.3.3.3 PRESS START

4.3.3.4 THE PROGRAM WILL PERFORM AS DESCRIBED IN 4.3.1.5 TO 4.3.1.9

4.3.4 PROGRAM RESTART WITH SW01=1

4.3.4.1 LOAD ADDRESS 000200

4.3.4.2 SET SW01=1

4.3.4.3 PRESS START

4.3.4.4 THE PROGRAM WILL TYPE "DH11 MULTI-LINE DATATEST" AND WILL THEN TYPE "TEST PC-" AND WILL WAIT FOR AN INPUT FROM THE TELETYPE KEYBOARD

4.3.4.5 TYPE IN THE ADDRESS OF THE TEST AT WHICH THE PROGRAM IS TO BE STARTED FOLLOWED BY <CARRIAGE RETURN>

4.3.4.6 THE PROGRAM WILL TYPE R TO INDICATE THAT IT HAS STARTED AND WILL START TESTING AT THE SELECTED TEST.

NOTE

CARE MUST BE TAKEN WHEN THIS FEATURE IS USED. SINCE THERE IS NO PROTECTION AGAINST SELECTING AN ADDRESS THAT IS IN THE MIDDLE OF A TEST

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53

NOTE

IF IT IS DESIRED TO LOOP ON THE TEST
THAT IS SELECTED SET SW14=1 BEFORE
ENTERING THE TEST ADDRESS

5.0 OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

SW15=1, HALT ON ERROR
SW14=1, LOOP ON CURRENT TEST
SW13=1, SUPPRESS ERROR TYPEOUT
SW11=1, INHIBIT ITERATIONS
SW10=1, ESCAPE TO NEXT TEST ON ERROR
SW09=1, FREEZE VARIABLE PARAMETER IN CURRENT TEST
SW01=1, START PROGRAM AT SELECTED TEST
SW00=1, CHANGE PARAMETERS AT PROGRAM RESTART

5.2 SUBROUTINE ABSTRACTS

5.2.1 TRAPCATCHER (LOCATIONS 000000-000776)

THIS ROUTINE IS USED TO INTERCEPT UNEXPECTED INTERRUPTS AND TRAPS.
THE AREA FROM 000000-000776 IS LOADED WITH THE FOLLOWING SEQUENCE

2
0
4
0
...
772
0
776
0

IF AN UNEXPECTED INTERRUPT OR TRAP OCCURS, THE PROGRAM WILL HALT WITH
THE PC 2 GREATER THAN THE ADDRESS TO WHICH THE PROGRAM TRAPPED. THE
PROCESSOR STACK MAY BE EXAMINED TO DETERMINE WHERE THE PROGRAM WAS
WHEN THE TRAP OR INTERRUPT OCCURED.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52

5.2.2 START (PROGRAM INITIALIZATION)

THIS ROUTINE INITIALIZES ALL PROGRAM FLAGS AND COUNTERS, TYPES THE PROGRAM TITLE MESSAGE, AND INPUTS THE VECTOR AND CONTROL REGISTER ADDRESSES OF THE DH11 TO BE TESTED.

5.2.3 BEGIN (PROGRAM START AND RESTART)

THIS ROUTINE IS ENTERED IMMEDIATELY AFTER "START" AND EACH TIME A PROGRAM PASS HAS BEEN COMPLETED. THE ROUTINE SETS UP THE PROCESSOR STACK AND STATUS WORD AND THEN TRANSFERS CONTROL TO THE TEST AT WHICH TESTING WILL BEGIN. IF SW01=0 WHEN THIS ROUTINE IS ENTERED TESTING WILL START AT T1 (TEST 1). IF SW01=1 WHEN THIS ROUTINE IS ENTERED, TESTING WILL START AT THE PC ENTERED FROM THE TELETYPE KEYBOARD.

5.2.4 EOP (END OF PASS)

THIS ROUTINE IS ENTERED ONCE PER PASS AFTER ALL TESTS HAVE BEEN COMPLETED. THIS ROUTINE TYPES THE MAINDEC IDENTIFICATION CODE OF THE PROGRAM, CLEARS ERROR FLAGS AND UPDATES THE PASS COUNT. IF THE PROGRAM WAS LOADED UNDER ACT11 OR DDP, THE ROUTINE CHECKS FOR RETURN TO THE ACT11 OR DDP MONITOR. IF THE PROGRAM IS NOT UNDER MONITOR CONTROL, THE ROUTINE TRANSFERS TO BEGIN.

5.2.5 SCOPER (SCOPE LOOP AND ITERATION HANDLER)

THIS ROUTINE IS ENTERED EACH TIME A TEST IS COMPLETED. THE ROUTINE CHECKS FOR THE FOLLOWING UPON ENTRY

1. IF SW10=1, THE ROUTINE WILL TRANSFER TO THE NEXT TEST IN SEQUENCE, AFTER CLEARING ERROR FLAGS.
2. IF SW11=1, THE ROUTINE WILL TRANSFER TO THE NEXT TEST SEQUENCE, AFTER CLEARING ERROR FLAGS.
3. IF SW14=1, THE ROUTINE WILL LOOP ON THE CURRENT TEST REGARDLESS OF THE ITERATION COUNT.

IF NONE OF THE ABOVE IS TRUE, THE ROUTINE WILL ADD 1 TO THE COUNT OF TEST ITERATIONS, AND COMPARE THIS VALUE TO THE NUMBER OF ITERATIONS THAT SHOULD BE PERFORMED. IF THESE NUMBERS ARE EQUAL, THE ROUTINE WILL TRANSFER TO THE NEXT TEST IN SEQUENCE. IF THE NUMBERS ARE NOT EQUAL, THE TEST CURRENTLY IN PROGRESS WILL BE REPEATED.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52

5.2.6 SCOP1R (FREEZE ON CURRENT DATA)

THE CALL TO THIS ROUTINE FOLLOWS IMMEDIATELY AFTER THE CALL TO THE ERROR HANDLER IN THOSE TESTS THAT HAVE VARIABLE PARAMETERS. THIS ROUTINE IS ALWAYS ENTERED IN THOSE TESTS, WHETHER OR NOT AN ERROR OCCURS. IF SW09=1, THE ROUTINE WILL TRANSFER CONTROL BACK TO THE TEST AT A POINT WHICH WILL ALLOW REPEATING THE FUNCTION UNDER TEST CONTINUOUSLY WITH THE SAME DATA. IF THIS OPTION IS SELECTED, THE ROUTINE "SCOPER" IS NEVER ENTERED AND ITERATION COUNTS WILL NOT BE UPDATED.

5.2.7 ERRORS (ERROR HANDLER)

THIS ROUTINE IS ENTERED UPON ERROR DETECTION ONLY. WITH ALL CONSOLE SWITCHES DOWN, THE ROUTINE PROCEEDS AS FOLLOWS:

1. THE PC OF THE INSTRUCTION THAT CALLED THE ERROR HANDLER IS ACCESSED THRU THE STACK, AND THEN THE EMT INSTRUCTION ITSELF IS FETCHED. THE 8 LSB OF THE EMT INSTRUCTION ARE THE ERROR CODE. THIS CODE IS USED TO ACCESS A TABLE OF ERROR MESSAGES AND ERROR DATA STORAGE LOCATIONS.
2. IF THE TEST THAT FAILED DID NOT FAIL PREVIOUSLY DURING THIS PASS, A COMPLETE ERROR REPORT IS MADE IF THE TEST THAT FAILED FAILED MORE THAN ONCE DURING THE CURRENT PASS, ONLY THE DATA RELATING TO THE FAILURE IS TYPED. IF SW13=1, NO ERROR TYPEOUT IS MADE.
3. THE ROUTINE NOW CHECKS FOR HALT ON ERROR. IF SW15=1 THE PROGRAM WILL HALT WITH THE PC OF THE CALL TO THE ERROR ROUTINE IN RO. IF SW15=0, THE PROGRAM WILL NOT HALT, BUT WILL CHECK FOR ESCAPE TO NEXT TEST.
4. IF SW10=0, THE ROUTINE WILL RETURN TO THE TEST IN PROGRESS. IF SW10=1, THE ROUTINE WILL ABORT THE CURRENT TEST, AND TRANSFER TO THE NEXT TEST IN SEQUENCE, THRU THE ROUTINE "SCOPER".

5.2.8 TRPSRV (TRAP DECODE AND DISPATCH)

THIS ROUTINE DECODES THE 8 LSB OF THE TRAP INSTRUCTION THAT CAUSED THE PROGRAM INTERRUPT, AND TRANSFERS CONTROL TO THE ROUTINE THRU THE TABLE "TRPTAB" USING THE 8 LSB OF THE TRAP INSTRUCTION AS AN OFFSET TO THE POINTER TO THE ROUTINE TO BE ENTERED.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50

5.3 PROGRAM AND OR OPERATOR ACTION

5.3.1 PROGRAM START WITH ALL SWITCHES DOWN

5.3.1.1 REFER TO SECTIONS 4.3.1 AND 4.3.2 FOR INITIAL PROGRAM BEHAVIOR.

5.3.1.2 AFTER "R" HAS BEEN TYPED BY THE PROGRAM, TEST EXECUTION WILL BEGIN. EACH TEST WILL BE REPEATED A SELECTED NUMBER OF ITERATIONS (SEE LISTING FOR EXACT NUMBER FOR EACH TEST) AND THEN THE PROGRAM WILL PROCEED TO THE NEXT TEST.

5.3.1.3 WHEN ALL ITERATIONS HAVE BEEN COMPLETED, THE PROGRAM WILL TYPE "CZDHG-C" AND THEN RESTART TESTING AT TEST 1 (LOCATION T1 IN THE PROGRAM).

5.3.1.4 IF AN ERROR OCCURS, THE PROGRAM WILL TYPE AN APPROPRIATE ERROR MESSAGE, AND THEN CONTINUE THE TEST IN PROGRESS.

5.3.2 PROGRAM START WITH SW00=1

THE PROGRAM WILL PERFORM AS DESCRIBED IN 4.3.1 AND 5.3.1

5.3.3 PROGRAM START WITH SW01=1

5.3.3.1 REFER TO SECTION 4.3.4 FOR INITIAL PROGRAM BEHAVIOR

5.3.3.2 TEST EXECUTION WILL START AT THE ADDRESS SPECIFIED AND WILL CONTINUE AS DESCRIBED IN 5.3.1.2

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

5.3.3.3 AFTER "CZDHG-C" HAS BEEN TYPED, THE PROGRAM WILL RESUME TESTING AT TEST 1

5.3.4 PROGRAM OPERATION WITH SW15-1

SAME AS 5.3.1, EXCEPT THAT IN THE CASE OF AN ERROR, THE PROGRAM WILL HALT AFTER THE ERROR TYPEOUT, AND THE PC+2 OF THE CALL TO THE ERROR ROUTINE WILL BE DISPLAYED IN RO.

5.3.5 PROGRAM OPERATION WITH SW13-1

SAME AS 5.3.1 EXCEPT THAT NO ERROR TYPEOUTS WILL OCCUR

5.3.6 PROGRAM OPERATION WITH SW11-1

SAME AS 5.3.1 EXCEPT THAT EACH TEST WILL BE REPEATED ONCE ONLY

5.3.7 PROGRAM OPERATION WITH SW10-1

SAME AS 5.3.1, EXCEPT THAT IN THE CASE OF AN ERROR THE CURRENT TEST WILL BE ABORTED, AND THE PROGRAM WILL PROCEED TO THE NEXT TEST IN SEQUENCE.

5.3.8 PROGRAM OPERATION WITH SW14-1, OR SW09-1

THESE FUNCTIONS ARE NORMALLY USED FOR TROUBLE SHOOTING. SEE SECTION 6.3 FOR THEIR USE.

44
45
46
47
48
49
50
51
52
53
54
55
56
57
58

6.0 ERRORS

6.1 ERROR HALTS

THE ERROR MESSAGE FORMAT FOR ALL ERROR TYPEOUTS IS AS FOLLOWS

PC+2	MESSAGE
	HEADER (IF APPLICABLE)
	DATA (IF APPLICABLE)

WHERE

PC+2 IS THE ADDRESS OF THE CALL TO THE ERROR HANDLER + 2;
MESSAGE IS AN ASCII MESSAGE DESCRIBING (BRIEFLY) THE FAILURE;
HEADER IS A DESCRIPTION OF THE DATA TO FOLLOW;
DATA IS OCTAL INFORMATION RELATING TO THE CAUSE OF THE FAILURE. IF

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50

THE SAME ERROR OCCURS IN A GIVEN TEST ON THE SAME PASS, AND IF DATA IS ASSOCIATED WITH THAT ERROR, ONLY DATA IS TYPE ON SUCCEEDING ERROR TYPEOUTS.

IF NO DATA IS ASSOCIATED WITH THE ERROR THE COMPLETE ERROR MESSAGE IS TYPED.

6.1.1 ERROR DESCRIPTIONS

SEE LISTING FOR DETAILS OF ERRORS

6.2 ERROR RECOVERY

6.2.1 SW15=0

IF THE PROGRAM IS RUN WITH SW15=0, NO OPERATOR ACTION IS REQUIRED TO CONTINUE TESTING

6.2.2 SW15=1

IF THE PROGRAM IS RUN WITH SW15=1, TO CONTINUE TESTING AFTER THE PROGRAM HAS HALTED, PRESS THE PROCESSOR CONSOLE CONTINUE SWITCH

6.2.3 ILLEGAL INTERRUPTS

IF AN INTERRUPT OCCURS TO A VECTOR ADDRESS NOT SELECTED DURING PROGRAM INITIALIZATION, THE PROGRAM WILL HALT IN THE TRAPCATCHER. THE ADDRESS AT WHICH THE PROGRAM HALTS IS 2 GREATER THAN THE ADDRESS TO WHICH THE INTERRUPT OCCURED. THE PROGRAM MUST BE RESTARTED AT 200 TO RECOVER FROM THIS ERROR.

6.3 SCOPE LOOPING

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

6.3.1 TO SCOPE ON A SPECIFIC TEST, SET SW14=1 AND SW13=1. THIS WILL CAUSE THE PROGRAM TO CONTINUOUSLY LOOP ON THE SAME TEST, AND WILL CAUSE ALL ERROR TYPEOUTS TO BE INHIBITED

6.3.2 TO SCOPE ON A SPECIFIC VALUE OF A PARAMETER WITHIN A TEST, SET SW09=1 TO FREEZE THE DATA. (SEE LISTING FOR THOSE TESTS THAT INCORPORATE THIS FEATURE)

6.3.3 PROGRAM START TO SCOPE LOOP ON SELECTED TEST
PERFORM SECTION 4.3.4 WITH SW14=1

7.0 RESTRICTIONS

7.1 STARTING

THE DH11 TEST CARD MUST BE INSTALLED

7.2 RUNNING

NONE

38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58

8.0 MISCELLANEOUS

8.1 EXECUTION TIME

THE TIME FOR ONE PASS OF THE PROGRAM (END OF TYPEOUT OF CZDHG-C TO END OF TYPEOUT OF CZDHG-C) IS GIVEN FOR VARIOUS PROCESSORS IN THE TABLE BELOW

	TIME
PROCESSOR	
PDP-11/05,10	
PDP-11/20	
PDP-11/40	
PDP-11/45	

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

9.0 PROGRAM DESCRIPTION

THIS PROGRAM CONSISTS OF 2 TESTS. THE FIRST TEST RUNS ALL 16 LINES AT 4800 BAUD WITH THE RECEIVER OPERATING IN INTERRUPT MODE. THE TRANSMITTER INTERRUPT IS NOT ENABLED, BUT A BACKGROUND ROUTINE IS EXECUTED THAT FIRST WAITS FOR ALL TRANSMITTER BAR BITS TO CLEAR, AND THEN WAITS FOR THE SILO TO BE EMPTIED BY THE RECEIVER INTERRUPT SERVICE ROUTINE. WHEN BOTH OF THESE HAVE OCCURED, THE TEST HAS BEEN COMPLETED, AND IS RESTARTED IF ITERATIONS HAVE NOT BEEN INHIBITED.

THE RECEIVER INTERRUPT SERVICE ROUTINE FOR THIS TEST FIRST CHECKS TO SEE THAT CHARACTER AVAILABLE WAS SET AND THAT SILO OVERRUN WAS NOT SET. IT THEN READS THE RECEIVED CHARACTER FROM THE SILO, EXTRACTS THE LINE NUMBER FROM THE DATA READ, AND COMPARES THE RECEIVED DATA WORD (16 BITS) TO AN EXPECTED RECEIVED CHARACTER WORD FOR THE LINE DECODED. THE EXPECTED RECEIVED CHARACTER FOR THAT LINE IS THEN UPDATED, AND THE ROUTINE THEN RETURNS BACK TO THE TRANSMITTER BACKGROUND ROUTINE

THE SECOND TEST OPERATES WITH THE TRANSMITTER AND RECEIVER BOTH IN INTERRUPT MODE, BUT THE SILO ALARM LEVEL IS SET TO 63 (DECIMAL) CHARACTERS, AND A CHARACTER AVAILABLE INTERRUPT SHOULD NOT OCCUR.

THE RECEIVER IS SERVICED ON A PER CHARACTER BASIS BY CONTINUOUSLY TRYING TO READ CHARACTERS FROM THE NEXT RECEIVED CHARACTER REGISTER. WHEN THE VALID DATA BIT OF THE NEXT RECEIVED CHARACTER REGISTER IS SET, THE ROUTINE PROCEEDS TO CHECK THE CHARACTER AS IN THE PREVIOUS TEST. WHEN THE EXPECTED RECEIVED CHARACTER INCREMENTS TO 0 10 (DECIMAL) TIMES, AN END OF TRANSMIT FLAG IS SET. IF THIS FLAG IS NOT SET, THE TRANSMITTER IS RESTARTED FOR THE LINE WHOSE NEXT EXPECTED RECEIVED CHARACTER INCREMENTED TO 0. WHEN THE END OF TRANSMIT FLAG IS SET, NO LINES ARE RESTARTED, THE SILO IS EMPTIED OF ALL REMAINING CHARACTERS, AND THE TEST IS RESTARTED IF ITERATIONS ARE NOT INHIBITED.

10.0 LISTING

!

1 ; DHMAC-A - DH11 MACRO LIBRARY
 2 ; COPYRIGHT 1985, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
 3
 4

5 .LIST ME
 6 .NLIST MC,MD,CND
 7

104
 119
 131
 148
 158
 167
 303
 339
 373
 520
 563
 595
 607
 652
 664
 691
 712
 743
 744
 745
 746
 747
 748

; 3

; CMS REPLACEMENT HISTORY

; *9 SKONETSKI 26-APR-1985 16:23:08 "FIXED TYPO CAUSING ASSEMBLY ERRORS"
 ; *8 SKONETSKI 22-APR-1985 16:48:03 "TYPO ERROR IN VECTOR CHANGE CODE SOURCE FIXED"
 ; *7 SKONETSKI 22-APR-1985 16:26:04 "ADDED CODE TO SET VECTORS FOR PWR FAIL, ERRORS, AND EMT
 ; *6 SKONETSKI 22-APR-1985 14:22:35 "FIXED BRANCH ERROR IN END OF PASS ROUTINE"
 ; *5 SKONETSKI 22-APR-1985 08:28:54 "FIXED BUG (AN OCTASC MACRO CALL WAS WRONG) AND ADDED A
 ; *4 SKONETSKI 18-APR-1985 14:20:15 "ADDED SOFTWARE SWITCH REG SUPPORT, BUT UNTESTED"
 ; *3 SKONETSKI 12-APR-1985 10:34:52 "FIXED PROBLEMS WITH SPURIOUS CR/LFS"
 ; *2 SKONETSKI 11-APR-1985 16:00:24 "ADDED MACRO FROM SYSMAC.SML THAT SIZES FOR SOFTWARE SWI
 ; *1 SKONETSKI 11-APR-1985 15:49:05 "LIBRARY FOR DH11 DIAGNOSTICS"

TRAPS."
 749
 750
 CLEAN END OF PASS
 MESSAGE.
 751
 752
 753
 TCH REGISTER"
 754

2
3
5 000000

```
.LIST ME
.NLIST MC,MD,CND
.HEADER †/1972,1985/,†/DH11 SINGLE LINE PARITY CHECK & MULTI-LINE DATA TEST/,†/CZDHG-CO/
```

```
;STARTING PROCEDURE
;LOAD PROGRAM
;LOAD ADDRESS 000200
;PRESS START
;PROGRAM WILL TYPE DH11 SINGLE LINE PARITY CHECK & MULTI-LINE DATA TEST
;PROGRAM WILL TYPE "VECTOR ADDRESS-"
;TYPE IN THE ADDRESS OF THE RECEIVER INTERRUPT VECTOR
;FOR THE DH11 TO BE TESTED, FOLLOWED BY <CARRIAGE RETURN>
;PROGRAM WILL TYPE "CONTROL REGISTER ADDRESS-"
;TYPE IN THE ADDRESS OF THE SYSTEM CONTROL REGISTER
;FOR THE DH11 TO BE TESTED, FOLLOWED BY <CARRIAGE RETURN> ; 3
;PROGRAM WILL TYPE "R" TO INDICATE THAT TESTING HAS STARTED
;AT THE END OF A PASS, PROGRAM WILL TYPE " CZDHG-CO "
;AND THEN RESUM TESTING
```

000000
6 0C0000

```
.TITLE CZDHG-CO
.ENABLE ABS
.NLIST MC,MD,CND
.LIST ME
.SYMBOLS
```

```
;SWITCH REGISTER OPTIONS
```

```
100000 SW15=100000 ;=1,HALT ON ERROR
040000 SW14=40000 ;=1,LOOP ON CURRENT TEST
020000 SW13=20000 ;=1,INHIBIT ERROR TYPEOUT
010000 SW12=10000
004000 SW11=4000 ;=1,INHIBIT ITERATIONS
002000 SW10=2000 ;=1,ESCAPE TO NEXT TEST ON ERROR ; 3
001000 SW09=1000 ;=1,LOOP WITH CURRENT DATA
000400 SW08=400
000100 SW06=100
000040 SW05=40
000020 SW04=20
000010 SW03=10
000004 SW02=4
000002 SW01=2
000001 SW00=1

;RESTART PROGRAM AT SELECTED TEST
;RESELECT VECTOR AND CONTROL REGISTER
;ADDRESS AFTER PROGRAM RESTART
```

0

;REGISTER DEFINITIONS

```

000000      R0=#0      ;GENERAL REGISTER
000001      R1=#1      ;GENERAL REGISTER
000002      R2=#2      ;GENERAL REGISTER
000003      R3=#3      ;GENERAL REGISTER
000004      R4=#4      ;GENERAL REGISTER
000005      R5=#5      ;GENERAL REGISTER
000006      SP=#6      ;PROCESSOR STACK POINTER
000007      PC=#7      ;PROGRAM COUNTER

```

;LOCATION EQUIVALENCIES

```

;SWR=177570 ;CONSOLE SWITCH REGISTER ; 3
;LIGHTS=177570 ;PDP-11/45 DISPLAY REGISTER ; 4
177776      PS=177776 ;PROCESSOR STATUS WORD ; 4
013752      STACK=ENDCOD+200 ;START OF PROCESSOR STACK ; 3

```

;INSTRUCTION DEFINITIONS

```

005746      PUSH1SP=5746 ;DECREMENT PROCESSOR STACK 1 WORD
005726      POP1SP=5726 ;INCREMENT PROCESSOR STACK 1 WORD
010046      PUSHRO=10046 ;SAVE R0 ON STACK
012600      POPRO=12600 ;RESTORE R0 FROM STACK
024646      PUSH2SP=24646 ;DECREMENT STACK TWICE
022626      POP2SP=22626 ;INCREMENT STACK TWICE

```

```

;
.MACRO HLT $A
          EMT $A
.ENDM HLT
;
;

```

```

100000      BIT15=100000
040000      BIT14=40000
020000      BIT13=20000
010000      BIT12=10000
004000      BIT11=4000
002000      BIT10=2000
001000      BIT09=1000
000400      BIT08=400
000200      BIT07=200
000100      BIT06=100
000040      BIT05=40
000020      BIT04=20
000010      BIT03=10
000004      BIT02=4
000002      BIT01=2
000001      BIT00=1
1 000000    .CATCH

```

0

000000
000200

;TRAPCATCAER FOR ILLEGAL INTERRUPTS

.=0
.REPT 200

.ENDR

000000	000002	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000002	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000004	000006	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000006	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000010	000012	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000012	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000014	000016	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000016	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000020	000022	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000022	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000024	000026	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000026	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000030	000032	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000032	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000034	000036	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000036	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000040	000042	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000042	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000044	000046	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000046	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000050	000052	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000052	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000054	000056	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000056	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000060	000062	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000062	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000064	000066	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000066	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000070	000072	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000072	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000074	000076	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000076	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000100	000102	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000102	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000104	000106	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000106	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000110	000112	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000112	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000114	000116	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000116	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000120	000122	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000122	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000124	000126	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000126	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000130	000132	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000132	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000134	000136	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000136	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000140	000142	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000142	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000144	000146	.+2	;UNEXPECTED TRAP TO THIS LOCATION

000146	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000150	000152	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000152	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000154	000156	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000156	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000160	000162	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000162	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000164	000166	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000166	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000170	000172	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000172	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000174	000176	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000176	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000200	000202	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000202	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000204	000206	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000206	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000210	000212	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000212	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000214	000216	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000216	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000220	000222	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000222	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000224	000226	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000226	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000230	000232	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000232	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000234	000236	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000236	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000240	000242	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000242	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000244	000246	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000246	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000250	000252	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000252	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000254	000256	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000256	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000260	000262	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000262	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000264	000266	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000266	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000270	000272	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000272	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000274	000276	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000276	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000300	000302	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000302	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000304	000306	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000306	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000310	000312	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000312	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000314	000316	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000316	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000320	000322	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000322	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000324	000326	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000326	000000	HALT	;EXAMINE STACK TO FIND CAUSE

000330	000332	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000332	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000334	000336	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000336	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000340	000342	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000342	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000344	000346	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000346	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000350	000352	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000352	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000354	000356	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000356	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000360	000362	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000362	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000364	000366	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000366	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000370	000372	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000372	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000374	000376	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000376	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000400	000402	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000402	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000404	000406	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000406	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000410	000412	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000412	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000414	000416	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000416	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000420	000422	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000422	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000424	000426	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000426	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000430	000432	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000432	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000434	000436	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000436	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000440	000442	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000442	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000444	000446	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000446	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000450	000452	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000452	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000454	000456	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000456	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000460	000462	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000462	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000464	000466	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000466	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000470	000472	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000472	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000474	000476	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000476	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000500	000502	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000502	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000504	000506	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000506	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000510	000512	.+2	;UNEXPECTED TRAP TO THIS LOCATION

000512	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000514	000516	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000516	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000520	000522	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000522	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000524	000526	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000526	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000530	000532	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000532	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000534	000536	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000536	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000540	000542	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000542	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000544	000546	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000546	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000550	000552	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000552	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000554	000556	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000556	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000560	000562	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000562	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000564	000566	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000566	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000570	000572	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000572	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000574	000576	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000576	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000600	000602	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000602	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000604	000606	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000606	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000610	000612	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000612	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000614	000616	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000616	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000620	000622	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000622	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000624	000626	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000626	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000630	000632	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000632	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000634	000636	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000636	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000640	000642	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000642	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000644	000646	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000646	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000650	000652	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000652	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000654	000656	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000656	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000660	000662	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000662	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000664	000666	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000666	000000	HALT	;EXAMINE STACK TO FIND CAUSE
000670	000672	.+2	;UNEXPECTED TRAP TO THIS LOCATION
000672	000000	HALT	;EXAMINE STACK TO FIND CAUSE

```
000674 000676      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000676 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
000700 000702      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000702 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
000704 000706      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000706 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
000710 000712      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000712 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
000714 000716      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000716 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
000720 000722      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000722 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
000724 000726      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000726 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
000730 000732      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000732 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
000734 000736      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000736 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
000740 000742      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000742 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
000744 000746      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000746 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
000750 000752      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000752 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
000754 000756      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000756 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
000760 000762      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000762 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
000764 000766      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000766 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
000770 000772      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000772 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
000774 000776      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
000776 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
1 001000      .SETVEC
```



```

0          ;STANDARD INTERRUPT VECTORS
000200 000200 000167 000600  .-200  JMP  START          ;GO TO START OF PROGRAM

1 000204  .TRPDEF

          ;DEFINITIONS FOR TRAP SUBROUTINE CALLS
          ;POINTERS TO SUBROUTINES CAN BE FOUND STARTING
          ;AT LOCATION "TRPTAB"

000204 104400 TRPDEF SCOPE,+/SCOPE LOOP AND ITERATION HANDLER/
000001 000001 SCOPE=TRAP+Y          ;SCOPE LOOP AND ITERATION HANDLER
          Y=Y+1

000204 104401 TRPDEF TYPE,+/TELETYPE OUTPUT ROUTINE/
000002 000002 TYPE=TRAP+Y          ;TELETYPE OUTPUT ROUTINE
          Y=Y+1

000204 104402 TRPDEF OCTASC,+/OCTAL TO ASCII CONVERSION/
000003 000003 OCTASC=TRAP+Y          ;OCTAL TO ASCII CONVERSION
          Y=Y+1

000204 104403 TRPDEF INSTR,+/INPUT ASCII STRING/
000004 000004 INSTR=TRAP+Y          ;INPUT ASCII STRING
          Y=Y+1

000204 104404 TRPDEF INSTER,+/STRING INPUT ERROR/
000005 000005 INSTER=TRAP+Y          ;STRING INPUT ERROR
          Y=Y+1

000204 104405 TRPDEF PARAM,+/CONVERT STRING TO OCTAL, CHECK LIMITS/
000006 000006 PARAM=TRAP+Y          ;CONVERT STRING TO OCTAL, CHECK LIMITS
          Y=Y+1

000204 104406 TRPDEF SAV05P,+/SAVE R0-R5, PC/
000007 000007 SAV05P=TRAP+Y          ;SAVE R0-R5, PC
          Y=Y+1

000204 104407 TRPDEF RES05,+/RESTORE R0-R5/
000010 000010 RES05=TRAP+Y          ;RESTORE R0-R5
          Y=Y+1

000204 104410 TRPDEF SCOPE1,+/CHECK FOR FREEZE ON CURRENT DATA/
000011 000011 SCOPE1=TRAP+Y          ;CHECK FOR FREEZE ON CURRENT DATA
          Y=Y+1

        .MACRO CODEM1
        MOV  DHSSR,DHSLR          ;SET UP ADDRESS OF SILO
        INC  DHSLR              ;STATUS REGISTER HIGH BYTE

        .ENDM CODEM1
6 000204 .START DHRVEC,3,4,DHSCR,0,177776,7,10,...1

```

2
 4
 5
 6

0

001000

.-1000

```

;PROGRAM INITIALIZATION
;LOCK OUT INTERRUPTS
;SET UP PROCESSOR STACK
;SET UP POWER FAIL VECTOR
;CLEAR PROGRAM FLAGS AND COUNTS
;TYPE TITLE MESSAGE
.IIF NB <>, ;DETERMINE MEMORY SIZE
.IIF NB <>, ;SET UP TRACE TRAP RETURN

001000 177570 SWR: .WORD 177570 ; SWITCH DHSCR ADDRESS ; 4
001002 177570 LIGHTS: .WORD 177570 ; LIGHTS ; 4
; 4

001004 012767 000340 176764 START: MOV #340,PS ;LOCK OUT INTERRUPTS
001012 012706 013752 MOV #STACK,SP ;SET UP PROCESSOR STACK
001016 012702 000024 MOV #24,R2 ; POINT TO VECTOR AREA ; 7
001022 012722 012512 MOV #PFail,(R2)+ ;SET UP POWER FAIL TRAP ; 7
001026 012722 000340 MOV #340,(R2)+ ;SERVICE AT LEVEL 7 ; 7
001032 012722 010704 MOV #ERRORS,(R2)+ ;ERROR HANDLER ; 7
001036 012722 000340 MOV #340,(R2)+ ;SERVICE AT LEVEL 7 ; 7
001042 012722 011116 MOV #TRPSRV,(R2)+ ;GENERAL HANDLER DISPATCH SERVICE ; 7
001046 012712 000340 MOV #340,(R2) ;SERVICE AT LEVEL 7 ; 8
001052 005067 010762 CLR STFLG ;CLEAR TEST START FLAG
001056 005067 010716 CLR PASCNT ;CLEAR PASS COUNT
001062 005067 010714 CLR ERRCNT ;CLEAR ERROR COUNT
001066 005067 010704 CLR ERRFLG ;CLEAR ERROR FLAG
001072 005067 010700 CLR ERRFLG ;CLEAR LAST ERROR PC
001076 016746 176702 MOV 4,-(SP) ; PUSH TRAP VECTOR ; 4
001102 016746 176700 MOV 6,-(SP) ; 4
001106 012767 001122 176670 MOV #1#,4 ; SET UP TRAP VECTOR ; 4
001114 005777 177660 TST #SMR ; TEST SWITCH REGISTER ADDRESS ; 4
001120 000405 BR 2# ; IF SUCCESSFUL, LEAVE IT ALONE ; 4
001122 1# : ; 4
001122 012767 000176 177650 MOV #176, SWR ; POINT TO SOFT SWITCH DHSCR ; 4
001130 005067 177646 CLR LIGHTS ; 0 MEANS WE ARE NOT GOING TO USE LIGHTS ; 4
001134 2# : ; 5
001134 005726 TST (SP)+ ; CLEAN UP STACK ; 4
001136 005726 TST (SP)+ ; 4
001140 012667 176642 MOV (SP)+, 6 ; 4
001144 012667 176634 MOV (SP)+, 4 ; 4
001150 104401 012662 TYPE ,MTITLE ;TYPE TITLE MESSAGE ; 4
001154 005767 010656 TST INIFLG ;CHECK INITIALIZATION FLAG

.IF NB <DHRVEC>
BNE VEC1 ;IF NOT 0, CHECK SWITCHES
;FOR REINITIALIZATION

.IFF
BNE BEGIN ;IF NOT 0, START TEST

.ENDC
.IF NB <>
SIZE: CLR R0
MOV #2#,R04 ;SET UP TIME OUT RETURN
1# : TST (R0)+ ;WILL TRAP WHEN NO MEMORY ; 9
BR 1# ;LOCATION RESPONDED, CONTINUE
2# : MOV R0,HCORE ;R0 CONTAINS ADDRESS OF
SUB #2,HCORE ;NON EXISTANT MEMORY ; 9
MOV #6,R04 ;RESTORE TRAPCATCHER

```

```

.ENDC
.IF NB <>
TRACER: MOV #14,0#10 ;SET UP ILLEGAL INSTRUCTION TRAP RETURN
SXT R0 ;DO 11/40, 11/45 INSTRUCTION
MOV #RTT,TRTRET ;11/40,45 RTT RETURN FROM TRACE TRAP
BR 2#
1# : MOV #RTI,TRTRET ;1105,10,20 RTI RETURN FROM TRACE TRAP
MOV #12,0#10 ;RESTORE TRAPCATCHER
MOV #TRTRET,0#16 ;SET UP TRACE TRAP VECTOR

.ENDC
.IF NB <DHRVEC>
.IF B <>
001162 000404 BR VEC2 ; 3
.IFF
TST INIFLG ;IF INITIALIZE FLAG=0
BEQ VEC2 ;GET VECTOR AND CSR ADDRESS

.ENDC
VEC1: BIT #SW00,0SMR ;IF SW00=1, GET NEW VECTOR ; 4
BEQ BEGIN ;AND CSR ; 4
VEC2: MOV #300,R1 ; 4
MOV #302,R2 ; 4
MOV #4,R3
1# : MOV R2,(R1) ;RESTORE TRAPCATCHER
CLR (R2) ;IN FLOATING VECTOR AREA
ADD R3,R1
ADD R3,R2
001220 020127 001000 CMP R1,#1000
001224 001371 BNE 1#
001226 104403 INSTR ;INPUT ADDRESS OF DEVICE VECTOR
001230 012755 MVECTOR ;MESSAGE "VECTOR ADDRESS-"
001232 104405 PARAM ;CONVERT STRING TO OCTAL
001234 000300 300 ;LOW LIMIT
001236 000770 770 ;HIGH LIMIT ; 3
001240 011766 DHRVEC ;LOCATIONS TO BE FILLED
001242 003 .BYTE 3 ;NUMBER OF LOCATIONS
001243 004 .BYTE 4 ;LSB MASK
001244 104403 INSTR ;INPUT ADDRESS OF DEVICE CSR
001246 012777 MREGAD ;MESSAGE "CONTROL REGISTER ADDRESS-"
001250 104405 PARAM ;CONVERT STRING TO OCTAL
001252 000000 0 ;LOW LIMIT
001254 177776 177776 ;HIGH LIMIT
001256 011744 DHSCR ;LOCATIONS TO BE FILLED
001260 007 .BYTE 7 ;NUMBER OF LOCATIONS
001261 010 .BYTE 10 ;LSB MASK

.ENDC
.IF NB <1>
001262 016767 010474 010474 MOV DHSSR,DHSLR ;SET UP ADDRESS OF SILO
001270 005267 010470 INC DHSLR ;STATUS REGISTER HIGH BYTE

.ENDC
TST INIFLG ;IF INITIALIZATION FLAG
BNE BEGIN ;IS CLEARED
001302 005167 010530 COM INIFLG ;SET IT

;PROGRAM START
;CHECK FOR PROGRAM START AT SELECTED ADDRESS ; 3

```

```

001306 012767 000340 176462 BEGIN: MOV    #340,PS           ;LOCK OUT INTERRUPTS
001314 012706 013752          MOV    #STACK,SP        ;SET UP PROCESSOR STACK
001320 032777 000002 177452  BIT    #SW01,@SWR      ;IF SW01=1                ; 4
001326 001410          BEQ    1#              ;GET PC FOR PROGRAM START
001330 104403          INSTR                     ;GET PC
001332 013166          MTSTPC                    ;MESSAGE "TEST PC"
001334 104405          PARAM                     ;CONVERT STRING TO OCTAL
001336 000000          0
001340 017500          17500
001342 012004          RETRN
001344    001          .BYTE 1
001345    001          .BYTE 1
001346 000410          BR    2#
001350 012767 001400 010426 1#:  MOV    #T1,RETRN      ;NORMAL START, TEST 1
001356 005767 010456          TST    STFLG          ;IF LOOPING, BYPASS TYPEOUT
001362 001004          BNE    3#
001364 005167 010450          COM    STFLG
001370 104401 013162          2#:  TYPE  ,MR
001374 000177 010404          3#:  JMP    @RETRN        ;TYPE "R" TO INDICATE START ; 3
;START TESTING

```

```

2      000000      LINE=0
3      000000      XLINE=LINE
4      000001      BITX=1
5      000001      XBIT=BITX
7      000020      .REPT 20
8      POPOVER \LINE,\BITX
9      .NLIST
10     LINE=LINE+1
11     BITX=BITX+BITX
12     .LIST
13     .ENDR
001400 POPOVER \LINE,\BITX

```

```

;RECEIVER PARITY LOGIC TEST ON LINE 0.
;TRANSMIT ONE CHARACTER TO THE RECEIVER
;WITH ODD PARITY SET,BREAK SET,HDX SET.
;THE RECEIVER WILL RECEIVE AN ALL 0'S CHARACTER
;WITH EVEN PARITY AND THEREBY
;CAUSE A PARITY ERROR AND OVERRUN
;CHARACTER LENGTH: 8 BITS
;LINE SPEED: 9600 BAUD

```

```

001400 TS      \XN,10,4$,1$
001400 012767 000340 176370 T1:  MOV      #340,PS      ;DISABLE ALL INTERRUPTS
001406 012767 000010 010376      MOV      #10,ICOUNT  ;SET UP FOR 10 ITERATIONS
001414 012767 001540 010364      MOV      #4$,ESCAPE  ;SET UP TO ESCAPE TO NEXT TEST
      .IF NB <1$>
001422 012767 001470 010360      MOV      #1$,FREEZ1  ;SET UP TO LOOP WITH DATA      ; 3
      .ENDC
      XN=XN+1
001430 012777 004000 010306      MOV      #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
001436 012703 000000      MOV      #0,R3      ;SET UP LINE #
001442 012705 130000      MOV      #0+400+130000,R5 ;SET EXPECTED LINE NUMBER
      ;VALID DATA FLAG,
      ;PARITY ERROR,DATA OVERRUN,
      ;AND EXPECTED DATA
001446 012767 000377 010372      MOV      #377,TDATA  ;ACTUAL DATA TO BE TRANSMITTED
001454 012777 000000 010262      MOV      #0,@DHSCR  ;SELECT LINE 0
001462 012777 073563 010260      MOV      #73563,@DHLPR ;SELECT 8 BITS/CHAR,ODD PARITY HDX,
      ;9600 BAUD SPEED FOR LINE 0
001470 012777 177777 010256 1$:  MOV      #-1,@DHBC  ;TRANSMIT 1 CHARACTER
001476 012777 012046 010246      MOV      #TDATA,@DHBA ;ADDRESS OF TRANSMITTED DATA
001504 012777 000001 010246      MOV      #1,@DHBCR  ;SET BREAK FOR LINE 0.
001512 012777 000001 010236      MOV      #1,@DHBAR  ;START TRANSMITTER
001520 105777 010220      2$:  TSTB   @DHSCR ;WAIT FOR CHARACTER TO BE RECEIVED
001524 100375      BPL     2$      ;
001526 017701 010214      MOV      @DHNR,R1  ;GET RECEIVED CHARACTER
001532 020501      CMP     R5,R1  ;COMPARE EXPECTED AND RECEIVED DATA
001534 001401      BEQ     .+4
001536      HLT     3      ;DATA ERROR
001536      EMT     3
001540 104003      4$:  SCOPE  ;CHECK FOR ITERATIONS,LOOP
      104400
      000001
      000002
001542 POPOVER \LINE,\BITX

```

```

;RECEIVER PARITY LOGIC TEST ON LINE 1.

```

```

;TRANSMIT ONE CHARACTER TO THE RECEIVER
;WITH ODD PARITY SET,BREAK SET,HDX SET.
;THE RECEIVER WILL RECEIVE AN ALL 0'S CHARACTER
;WITH EVEN PARITY AND THEREBY
;CAUSE A PARITY ERROR AND OVERRUN
;CHARACTER LENGTH: 8 BITS
;LINE SPEED: 9600 BAUD

001542      TS      \XN,10,4$,1$
001542 012767 000340 176226 T2:  MOV      #340,PS      ;DISABLE ALL INTERRUPTS
001550 012767 000010 010234      MOV      #10,ICOUNT   ;SET UP FOR 10 ITERATIONS
001556 012767 001702 010222      MOV      #4$,ESCAPE   ;SET UP TO ESCAPE TO NEXT TEST

      .IF NB <1$>
001564 012767 001632 010216      MOV      #1$,FREEZ1   ;SET UP TO LOOP WITH DATA      ; 3
      .ENDC
      XN=XN+1
001572 012777 004000 010144      MOV      #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
001600 012703 000001          MOV      #1,R3      ;SET UP LINE #
001604 012705 130400          MOV      #1*400+130000,R5 ;SET EXPECTED LINE NUMBER
;VALID DATA FLAG,
;PARITY ERROR,DATA OVERRUN,
;AND EXPECTED DATA
001610 012767 000377 010230      MOV      #377,TDATA   ;ACTUAL DATA TO BE TRANSMITTED
001616 012777 000001 010120      MOV      #1,@DHSCR    ;SELECT LINE 1
001624 012777 073563 010116      MOV      #73563,@DHLPR ;SELECT 8 BITS/CHAR,ODD PARITY HDX,
;9600 BAUD SPEED FOR LINE 1
001632 012777 177777 010114 1$:  MOV      #-1,@DHBC     ;TRANSMIT 1 CHARACTER
001640 012777 012046 010104      MOV      @TDATA,@DHBA ;ADDRESS OF TRANSMITTED DATA
001646 012777 000002 010104      MOV      #2,@DHBCR    ;SET BREAK FOR LINE 1.
001654 012777 000002 010074      MOV      #2,@DHBAR    ;START TRANSMITTER
001662 105777 010056          2$:  TSTB     @DHSCR      ;WAIT FOR CHARACTER TO BE RECEIVED
001666 100375          BPL      2$          ;
001670 017701 010052          MOV      @DHNR,R1     ;GET RECEIVED CHARACTER
001674 020501          CMP      R5,R1      ;COMPARE EXPECTED AND RECEIVED DATA
001676 001401          BEQ      .+4
001700          HLT      3          ;DATA ERROR
001700 104003          EMT      3
001702 104400          4$:  SCOPE     ;CHECK FOR ITERATIONS,LOOP
      LINE=LINE+1
      BITX=BITX+BITX
      POPOVER \LINE,\BITX

;RECEIVER PARITY LOGIC TEST ON LINE 2.
;TRANSMIT ONE CHARACTER TO THE RECEIVER
;WITH ODD PARITY SET,BREAK SET,HDX SET.
;THE RECEIVER WILL RECEIVE AN ALL 0'S CHARACTER
;WITH EVEN PARITY AND THEREBY
;CAUSE A PARITY ERROR AND OVERRUN
;CHARACTER LENGTH: 8 BITS
;LINE SPEED: 9600 BAUD

001704      TS      \XN,10,4$,1$
001704 012767 000340 176064 T3:  MOV      #340,PS      ;DISABLE ALL INTERRUPTS
001712 012767 000010 010072      MOV      #10,ICOUNT   ;SET UP FOR 10 ITERATIONS
001720 012767 002044 010060      MOV      #4$,ESCAPE   ;SET UP TO ESCAPE TO NEXT TEST

      .IF NB <1$>
001726 012767 001774 010054      MOV      #1$,FREEZ1   ;SET UP TO LOOP WITH DATA      ; 3

```

```

                                .ENDC
                                XN=XN+1
001734 000004 004000 010002      MOV    #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
001742 012777 000002 010002      MOV    #2,R3 ;SET UP LINE #
001746 012705 131000 010002      MOV    #2*400+130000,R5 ;SET EXPECTED LINE NUMBER
                                ;VALID DATA FLAG,
                                ;PARITY ERROR,DATA OVERRUN,
                                ;AND EXPECTED DATA
001752 012767 000377 010066      MOV    #377,TDATA ;ACTUAL DATA TO BE TRANSMITTED
001760 012777 000002 007756      MOV    #2,@DHSCR ;SELECT LINE 2
001766 012777 073563 007754      MOV    #73563,@DHLPR ;SELECT 8 BITS/CHAR,ODD PARITY HDX,
                                ;9600 BAUD SPEED FOR LINE 2
001774 012777 177777 007752 1$:  MOV    #-1,@DHBC ;TRANSMIT 1 CHARACTER
002002 012777 012046 007742      MOV    #TDATA,@DHBA ;ADDRESS OF TRANSMITTED DATA
002010 012777 000004 007742      MOV    #4,@DHBCR ;SET BREAK FOR LINE 2.
002016 012777 000004 007732      MOV    #4,@DHBAR ;START TRANSMITTER
002024 105777 007714 007714      2$:  TSTB  @DHSCR ;WAIT FOR CHARACTER TO BE RECEIVED
002030 100375 007714 007714      BPL   @DHSCR ;
002032 017701 007710 007710      MOV    @DHNRC,R1 ;GET RECEIVED CHARACTER
002036 020501 007710 007710      CMP    R5,R1 ;COMPARE EXPECTED AND RECEIVED DATA
002040 001401 007710 007710      BEQ   .+4
002042 001401 007710 007710      HLT   3 ;DATA ERROR
002042 104003 007710 007710      EMT   3
002044 104400 000003 000010      4$:  SCOPE ;CHECK FOR ITERATIONS,LOOP
                                LINE=LINE+1
                                BITX=BITX+BITX
002046 000010 000010 000010      POPOVER \LINE,\BITX

                                ;RECEIVER PARITY LOGIC TEST ON LINE 3.
                                ;TRANSMIT ONE CHARACTER TO THE RECEIVER
                                ;WITH ODD PARITY SET,BREAK SET,HDx SET.
                                ;THE RECEIVER WILL RECEIVE AN ALL 0'S CHARACTER
                                ;WITH EVEN PARITY AND THEREBY
                                ;CAUSE A PARITY ERROR AND OVERRUN
                                ;CHARACTER LENGTH: 8 BITS
                                ;LINE SPEED: 9600 BAUD

002046 000005 000340 175722 TS \XN,10,4$,1$
002046 012767 000340 175722 T4:  MOV    #340,PS ;DISABLE ALL INTERRUPTS
002054 012767 000010 007730      MOV    #10,ICOUNT ;SET UP FOR 10 ITERATIONS
002062 012767 002206 007716      MOV    #4$,ESCAPE ;SET UP TO ESCAPE TO NEXT TEST
                                .IF NB <1$>
002070 012767 002136 007712      MOV    #1$,FREEZ1 ;SET UP TO LOOP WITH DATA ; 3
                                .ENDC
                                XN=XN+1
002076 000005 004000 007640      MOV    #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
002104 012703 000003 007640      MOV    #3,R3 ;SET UP LINE #
002110 012705 131400 007640      MOV    #3*400+130000,R5 ;SET EXPECTED LINE NUMBER
                                ;VALID DATA FLAG,
                                ;PARITY ERROR,DATA OVERRUN,
                                ;AND EXPECTED DATA
002114 012767 000377 007724      MOV    #377,TDATA ;ACTUAL DATA TO BE TRANSMITTED
002122 012777 000003 007614      MOV    #3,@DHSCR ;SELECT LINE 3
002130 012777 073563 007612      MOV    #73563,@DHLPR ;SELECT 8 BITS/CHAR,ODD PARITY HDX,
                                ;9600 BAUD SPEED FOR LINE 3
002136 012777 177777 007610 1$:  MOV    #-1,@DHBC ;TRANSMIT 1 CHARACTER
002144 012777 012046 007600      MOV    #TDATA,@DHBA ;ADDRESS OF TRANSMITTED DATA

```

```

002152 012777 000010 007600      MOV      #10, @DHBCR      ;SET BREAK FOR LINE 3.
002160 012777 000010 007570      MOV      #10, @DHBAR      ;START TRANSMITTER
002166 105777 007552              2$:      TSTB      @DHSCR      ;WAIT FOR CHARACTER TO BE RECEIVED
002172 100375                      BPL      2$              ;
002174 017701 007546              MOV      @DHNRC,R1      ;GET RECEIVED CHARACTER
002200 020501                      CMP      R5,R1      ;COMPARE EXPECTED AND RECEIVED DATA
002202 001401                      BEQ      .+4              ;
002204                      HLT      3              ;DATA ERROR
002204 104003                      EMT      3              ;
002206 104400                      4$:      SCOPE      ;CHECK FOR ITERATIONS,LOOP
002206 000004                      LINE=LINE+1
002210 000020                      BITX=BITX+BITX
                                POPOVER \LINE,\BITX

                                ;RECEIVER PARITY LOGIC TEST ON LINE 4.
                                ;TRANSMIT ONE CHARACTER TO THE RECEIVER
                                ;WITH ODD PARITY SET,BREAK SET,HDX SET.
                                ;THE RECEIVER WILL RECEIVE AN ALL 0'S CHARACTER
                                ;WITH EVEN PARITY AND THEREBY
                                ;CAUSE A PARITY ERROR AND OVERRUN
                                ;CHARACTER LENGTH: 8 BITS
                                ;LINE SPEED: 9600 BAUD

002210                      TS      \XN,10,4$,1$
002210 012767 000340 175560      T5$:      MOV      #340,PS      ;DISABLE ALL INTERRUPTS
002216 012767 000010 007566      MOV      #10,ICOUNT      ;SET UP FOR 10 ITERATIONS
002224 012767 002350 007554      MOV      #4$,ESCAPE      ;SET UP TO ESCAPE TO NEXT TEST
                                .IF NB <1$>
002232 012767 002300 007550      MOV      #1$,FREEZ1      ;SET UP TO LOOP WITH DATA ; 3
                                .ENDC
                                XN=XN+1
002240 012777 004000 007476      MOV      @BIT11,@DHSCR      ;MASTER CLEAR INTERFACE
002246 012703 000004              MOV      #4,R3      ;SET UP LINE #
002252 012705 132000              MOV      #4*400+130000,R5      ;SET EXPECTED LINE NUMBER
                                ;VALID DATA FLAG,
                                ;PARITY ERROR,DATA OVERRUN,
                                ;AND EXPECTED DATA
002256 012767 000377 007562      MOV      #377,TDATA      ;ACTUAL DATA TO BE TRANSMITTED
002264 012777 000004 007452      MOV      #4,@DHSCR      ;SELECT LINE 4
002272 012777 073563 007450      MOV      #73563,@DHLPR      ;SELECT 8 BITS/CHAR,ODD PARITY HDX,
                                ;9600 BAUD SPEED FOR LINE 4
002300 012777 177777 007446      1$:      MOV      #-1,@DHBC      ;TRANSMIT 1 CHARACTER
002306 012777 012046 007436      MOV      @TDATA,@DHBA      ;ADDRESS OF TRANSMITTED DATA
002314 012777 000020 007436      MOV      #20,@DHBCR      ;SET BREAK FOR LINE 4.
002322 012777 000020 007426      MOV      #20,@DHBAR      ;START TRANSMITTER
002330 105777 007410              2$:      TSTB      @DHSCR      ;WAIT FOR CHARACTER TO BE RECEIVED
002334 100375                      BPL      2$              ;
002336 017701 007404              MOV      @DHNRC,R1      ;GET RECEIVED CHARACTER
002342 020501                      CMP      R5,R1      ;COMPARE EXPECTED AND RECEIVED DATA
002344 001401                      BEQ      .+4              ;
002346                      HLT      3              ;DATA ERROR
002346 104003                      EMT      3              ;
002350 104400                      4$:      SCOPE      ;CHECK FOR ITERATIONS,LOOP
002352 000005                      LINE=LINE+1
                                BITX=BITX+BITX
                                POPOVER \LINE,\BITX

```



```

;RECEIVER PARITY LOGIC TEST ON LINE 5.
;TRANSMIT ONE CHARACTER TO THE RECEIVER
;WITH ODD PARITY SET,BREAK SET,HDX SET.
;THE RECEIVER WILL RECEIVE AN ALL 0'S CHARACTER
;WITH EVEN PARITY AND THEREBY
;CAUSE A PARITY ERROR AND OVERRUN
;CHARACTER LENGTH: 8 BITS
;LINE SPEED: 9600 BAUD

002352      TS      \XN,10,4$,1$
002352 012767 000340 175416 T6:  MOV      #340,PS      ;DISABLE ALL INTERRUPTS
002360 012767 000010 007424      MOV      #10,ICOUNT   ;SET UP FOR 10 ITERATIONS
002366 012767 002512 007412      MOV      #4$,ESCAPE  ;SET UP TO ESCAPE TO NEXT TEST

      .IF NB <1$>
002374 012767 002442 007406      MOV      #1$,FREEZ1  ;SET UP TO LOOP WITH DATA      ; 3
      .ENDC
      XN=XN+1
002402      000007
002402 012777 004000 007334      MOV      #BIT11,0DHSCR ;MASTER CLEAR INTERFACE
002410 012703 000005      MOV      #5,R3      ;SET UP LINE #
002414 012705 132400      MOV      #5*400+130000,R5 ;SET EXPECTED LINE NUMBER
                                       ;VALID DATA FLAG,
                                       ;PARITY ERROR,DATA OVERRUN,
                                       ;AND EXPECTED DATA
002420 012767 000377 007420      MOV      #377,TDATA  ;ACTUAL DATA TO BE TRANSMITTED
002426 012777 000005 007310      MOV      #5,0DHSCR   ;SELECT LINE 5
002434 012777 073563 007306      MOV      #73563,0DHLPR ;SELECT 8 BITS/CHAR,ODD PARITY HDX,
                                       ;9600 BAUD SPEED FOR LINE 5
002442 012777 177777 007304 1$:  MOV      #-1,0DHBC   ;TRANSMIT 1 CHARACTER
002450 012777 012046 007274      MOV      #TDATA,0DHBA ;ADDRESS OF TRANSMITTED DATA
002456 012777 000040 007274      MOV      #40,0DHBCR  ;SET BREAK FOR LINE 5.
002464 012777 000040 007264      MOV      #40,0DHBAR  ;START TRANSMITTER
002472 105777 007246      2$:  TSTB     0DHSCR    ;WAIT FOR CHARACTER TO BE RECEIVED
002476 100375      BPL      2$        ;
002500 017701 007242      MOV      0DHNR,R1    ;GET RECEIVED CHARACTER
002504 020501      CMP      R5,R1      ;COMPARE EXPECTED AND RECEIVED DATA
002506 001401      BEQ     .+4
002510      HLT     3      ;DATA ERROR
002510 104003      EMT     3
002512 104400      4$:  SCOPE   ;CHECK FOR ITERATIONS,LOOP
      000006
      000100
      LINE=LINE+1
      BITX=BITX+BITX
      POPOVER \LINE,\BITX

;RECEIVER PARITY LOGIC TEST ON LINE 6.
;TRANSMIT ONE CHARACTER TO THE RECEIVER
;WITH ODD PARITY SET,BREAK SET,HDX SET.
;THE RECEIVER WILL RECEIVE AN ALL 0'S CHARACTER
;WITH EVEN PARITY AND THEREBY
;CAUSE A PARITY ERROR AND OVERRUN
;CHARACTER LENGTH: 8 BITS
;LINE SPEED: 9600 BAUD

002514      TS      \XN,10,4$,1$
002514 012767 000340 175254 T7:  MOV      #340,PS      ;DISABLE ALL INTERRUPTS
002522 012767 000010 007262      MOV      #10,ICOUNT   ;SET UP FOR 10 ITERATIONS
002530 012767 002654 007250      MOV      #4$,ESCAPE  ;SET UP TO ESCAPE TO NEXT TEST

      .IF NB <1$>

```

```

002536 012767 002604 007244      MOV      #1$,FREEZ1      ;SET UP TO LOOP WITH DATA      ; 3
                                .ENDC
                                XN=XN+1
002544 012777 004000 007172      MOV      #BIT11,@DHSCR  ;MASTER CLEAR INTERFACE
002552 012703 000006 000006      MOV      #6,R3          ;SET UP LINE #
002556 012705 133000 000000      MOV      #6*400+130000,R5 ;SET EXPECTED LINE NUMBER
                                ;VALID DATA FLAG,
                                ;PARITY ERROR,DATA OVERRUN,
                                ;AND EXPECTED DATA
002562 012767 000377 007256      MOV      #377,TDATA    ;ACTUAL DATA TO BE TRANSMITTED
002570 012777 000006 007146      MOV      #6,@DHSCR     ;SELECT LINE 6
002576 012777 073563 007144      MOV      #73563,@DHLPR ;SELECT 8 BITS/CHAR,ODD PARITY HDX,
                                ;9600 BAUD SPEED FOR LINE 6
002604 012777 177777 007142 1$:  MOV      #-1,@DHBC     ;TRANSMIT 1 CHARACTER
002612 012777 012046 007132      MOV      #TDATA,@DHBA  ;ADDRESS OF TRANSMITTED DATA
002620 012777 000100 007132      MOV      #100,@DHBCR   ;SET BREAK FOR LINE 6.
002626 012777 000100 007122      MOV      #100,@DHBAR   ;START TRANSMITTER
002634 105777 007104 000000 2$:  TSTB     @DHSCR        ;WAIT FOR CHARACTER TO BE RECEIVED
002640 100375 000000 000000      BPL      2$            ;
002642 017701 007100 000000      MOV      @DHNRC,R1     ;GET RECEIVED CHARACTER
002646 020501 000000 000000      CMP      R5,R1        ;COMPARE EXPECTED AND RECEIVED DATA
002650 001401 000000 000000      BEQ      .+4           ;
002652 000000 000000 000000      HLT      3             ;DATA ERROR
002652 104003 000000 000000      EMT      3             ;
002654 104400 000007 000000 4$:  SCOPE    ;CHECK FOR ITERATIONS,LOOP
                                LINE=LINE+1
                                BITX=BITX+BITX
                                POPOVER \LINE,\BITX
                                ;RECEIVER PARITY LOGIC TEST ON LINE 7.
                                ;TRANSMIT ONE CHARACTER TO THE RECEIVER
                                ;WITH ODD PARITY SET,BREAK SET,HDX SET.
                                ;THE RECEIVER WILL RECEIVE AN ALL 0'S CHARACTER
                                ;WITH EVEN PARITY AND THEREBY
                                ;CAUSE A PARITY ERROR AND OVERRUN
                                ;CHARACTER LENGTH: 8 BITS
                                ;LINE SPEED: 9600 BAUD
002656 012767 000340 175112 TS      \XN,10,4$,1$
002656 012767 000010 007120 T10:   MOV      #340,PS      ;DISABLE ALL INTERRUPTS
002664 012767 000010 007120      MOV      #10,ICOUNT   ;SET UP FOR 10 ITERATIONS
002672 012767 003016 007106      MOV      #4$,ESCAPE   ;SET UP TO ESCAPE TO NEXT TEST
                                .IF NB <1$>
002700 012767 002746 007102      MOV      #1$,FREEZ1   ;SET UP TO LOOP WITH DATA      ; 3
                                .ENDC
                                XN=XN+1
002706 012777 004000 007030      MOV      #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
002714 012703 000007 000007      MOV      #7,R3        ;SET UP LINE #
002720 012705 133400 000000      MOV      #7*400+130000,R5 ;SET EXPECTED LINE NUMBER
                                ;VALID DATA FLAG,
                                ;PARITY ERROR,DATA OVERRUN,
                                ;AND EXPECTED DATA
002724 012767 000377 007114      MOV      #377,TDATA    ;ACTUAL DATA TO BE TRANSMITTED
002732 012777 000007 007004      MOV      #7,@DHSCR     ;SELECT LINE 7
002740 012777 073563 007002      MOV      #73563,@DHLPR ;SELECT 8 BITS/CHAR,ODD PARITY HDX,
                                ;9600 BAUD SPEED FOR LINE 7
002746 012777 177777 007000 1$:  MOV      #-1,@DHBC     ;TRANSMIT 1 CHARACTER

```



```

;RECEIVER PARITY LOGIC TEST ON LINE 11.
;TRANSMIT ONE CHARACTER TO THE RECEIVER
;WITH ODD PARITY SET,BREAK SET,HDX SET.
;THE RECEIVER WILL RECEIVE AN ALL 0'S CHARACTER
;WITH EVEN PARITY AND THEREBY
;CAUSE A PARITY ERROR AND OVERRUN
;CHARACTER LENGTH: 8 BITS
;LINE SPEED: 9600 BAUD

003162      TS      \XN,10,4$,1$
003162 012767 000340 174606 T12:  MOV      #340,PS      ;DISABLE ALL INTERRUPTS
003170 012767 000010 006614      MOV      #10,ICOUNT   ;SET UP FOR 10 ITERATIONS
003176 012767 003322 006602      MOV      #4$,ESCAPE  ;SET UP TO ESCAPE TO NEXT TEST

003204 012767 003252 006576      .IF NB  <1$>
                                MOV      #1$,FREEZ1      ;SET UP TO LOOP WITH DATA      ; 3
                                .ENDC
                                XN=XN+1

003212      000013
003212 012777 004000 006524      MOV      #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
003220 012703 000011      MOV      #11,R3      ;SET UP LINE #
003224 012705 134400      MOV      #11*400+130000,R5 ;SET EXPECTED LINE NUMBER
                                ;VALID DATA FLAG,
                                ;PARITY ERROR,DATA OVERRUN,
                                ;AND EXPECTED DATA

003230 012767 000377 006610      MOV      #377,TDATA   ;ACTUAL DATA TO BE TRANSMITTED
003236 012777 000011 006500      MOV      #11,@DHSCR  ;SELECT LINE 11
003244 012777 073563 006476      MOV      #73563,@DHLPR ;SELECT 8 BITS/CHAR,ODD PARITY HDX,
                                ;9600 BAUD SPEED FOR LINE 11

003252 012777 177777 006474 1$:  MOV      #-1,@DHBC    ;TRANSMIT 1 CHARACTER
003260 012777 012046 006464      MOV      #TDATA,@DHBA ;ADDRESS OF TRANSMITTED DATA
003266 012777 001000 006464      MOV      #1000,@DHBC  ;SET BREAK FOR LINE 11.
003274 012777 001000 006454      MOV      #1000,@DHBAR ;START TRANSMITTER
003302 105777 006436      2$:  TSTB     @DHSCR     ;WAIT FOR CHARACTER TO BE RECEIVED
003306 100375      BPL      2$          ;
003310 017701 006432      MOV      @DHNR,R1    ;GET RECEIVED CHARACTER
003314 020501      CMP      R5,R1      ;COMPARE EXPECTED AND RECEIVED DATA
003316 001401      BEQ     .+4
003320      HLT     3      ;DATA ERROR
003320 104003      EMT     3
003322 104400      4$:  SCOPE   ;CHECK FOR ITERATIONS,LOOP
                                LINE=LINE+1
                                BITX=BITX+BITX
003324      POPOVER \LINE,\BITX

;RECEIVER PARITY LOGIC TEST ON LINE 12.
;TRANSMIT ONE CHARACTER TO THE RECEIVER
;WITH ODD PARITY SET,BREAK SET,HDX SET.
;THE RECEIVER WILL RECEIVE AN ALL 0'S CHARACTER
;WITH EVEN PARITY AND THEREBY
;CAUSE A PARITY ERROR AND OVERRUN
;CHARACTER LENGTH: 8 BITS
;LINE SPEED: 9600 BAUD

003324      TS      \XN,10,4$,1$
003324 012767 000340 174444 T13:  MOV      #340,PS      ;DISABLE ALL INTERRUPTS
003332 012767 000010 006452      MOV      #10,ICOUNT   ;SET UP FOR 10 ITERATIONS
003340 012767 003464 006440      MOV      #4$,ESCAPE  ;SET UP TO ESCAPE TO NEXT TEST

```

```

003346 012767 003414 006434 .IF NB <1>
MOV #1,FREEZ1 ;SET UP TO LOOP WITH DATA ; 3
.ENDC
XN=XN+1
003354 012777 004000 006362 MOV #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
003362 012703 000012 MOV #12,R3 ;SET UP LINE #
003366 012705 135000 MOV #12+400+130000,R5 ;SET EXPECTED LINE NUMBER
;VALID DATA FLAG,
;PARITY ERROR,DATA OVERRUN,
;AND EXPECTED DATA
003372 012767 000377 006446 MOV #377,TDATA ;ACTUAL DATA TO BE TRANSMITTED
003400 012777 000012 006336 MOV #12,@DHSCR ;SELECT LINE 12
003406 012777 073563 006334 MOV #73563,@DHLPR ;SELECT 8 BITS/CHAR,ODD PARITY HDX,
;9600 BAUD SPEED FOR LINE 12
003414 012777 177777 006332 1$: MOV #-1,@DHBC ;TRANSMIT 1 CHARACTER
003422 012777 012046 006322 MOV #TDATA,@DHBA ;ADDRESS OF TRANSMITTED DATA
003430 012777 002000 006322 MOV #2000,@DHBCR ;SET BREAK FOR LINE 12.
003436 012777 002000 006312 MOV #2000,@DHBAR ;START TRANSMITTER
003444 105777 006274 2$: TSTB @DHSCR ;WAIT FOR CHARACTER TO BE RECEIVED
003450 100375 BPL 2$ ;
003452 017701 006270 MOV @DHNRC,R1 ;GET RECEIVED CHARACTER
003456 020501 CMP R5,R1 ;COMPARE EXPECTED AND RECEIVED DATA
003460 001401 BEQ .+4
003462 HLT 3 ;DATA ERROR
003462 104003 EMT 3
003464 104400 4$: SCOPE ;CHECK FOR ITERATIONS,LOOP
000013 LINE=LINE+1
004000 BITX=BITX+BITX
POPOVER \LINE,\BITX

;RECEIVER PARITY LOGIC TEST ON LINE 13.
;TRANSMIT ONE CHARACTER TO THE RECEIVER
;WITH ODD PARITY SET,BREAK SET,HDX SET.
;THE RECEIVER WILL RECEIVE AN ALL 0'S CHARACTER
;WITH EVEN PARITY AND THEREBY
;CAUSE A PARITY ERROR AND OVERRUN
;CHARACTER LENGTH: 8 BITS
;LINE SPEED: 9600 BAUD

003466 TS \XN,10,4$,1$
003466 012767 000340 174302 T14: MOV #340,PS ;DISABLE ALL INTERRUPTS
003474 012767 000010 006310 MOV #10,ICOUNT ;SET UP FOR 10 ITERATIONS
003502 012767 003626 006276 MOV #4$,ESCAPE ;SET UP TO ESCAPE TO NEXT TEST
.ENDC
003510 012767 003556 006272 .IF NB <1>
MOV #1,FREEZ1 ;SET UP TO LOOP WITH DATA ; 3
.ENDC
XN=XN+1
003516 012777 004000 006220 MOV #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
003524 012703 000013 MOV #13,R3 ;SET UP LINE #
003530 012705 135400 MOV #13+400+130000,R5 ;SET EXPECTED LINE NUMBER
;VALID DATA FLAG,
;PARITY ERROR,DATA OVERRUN,
;AND EXPECTED DATA
003534 012767 000377 006304 MOV #377,TDATA ;ACTUAL DATA TO BE TRANSMITTED
003542 012777 000013 006174 MOV #13,@DHSCR ;SELECT LINE 13
003550 012777 073563 006172 MOV #73563,@DHLPR ;SELECT 8 BITS/CHAR,ODD PARITY HDX,
;9600 BAUD SPEED FOR LINE 13

```

```

003556 012777 177777 006170 1$: MOV    # -1, @DHBC      ; TRANSMIT 1 CHARACTER
003564 012777 012046 006160      MOV    @TDATA, @DHBA    ; ADDRESS OF TRANSMITTED DATA
003572 012777 004000 006160      MOV    #4000, @DHBCR    ; SET BREAK FOR LINE 13.
003600 012777 004000 006150      MOV    #4000, @DHBAR    ; START TRANSMITTER
003606 105777 006132 2$: TSTB   @DHSCR    ; WAIT FOR CHARACTER TO BE RECEIVED
003612 100375      BPL    2$              ;
003614 017701 006126      MOV    @DHNRC, R1      ; GET RECEIVED CHARACTER
003620 020501      CMP    R5, R1         ; COMPARE EXPECTED AND RECEIVED DATA
003622 001401      BEQ   .+4            ;
003624      HLT    3          ; DATA ERROR
003626 104003      EMT    3              ;
104400 4$: SCOPE    ; CHECK FOR ITERATIONS, LOOP
000014 LINE=LINE+1
010000 BITX=BITX+BITX
003630 POPOVER \LINE, \BITX

; RECEIVER PARITY LOGIC TEST ON LINE 14.
; TRANSMIT ONE CHARACTER TO THE RECEIVER
; WITH ODD PARITY SET, BREAK SET, HDX SET.
; THE RECEIVER WILL RECEIVE AN ALL 0'S CHARACTER
; WITH EVEN PARITY AND THEREBY
; CAUSE A PARITY ERROR AND OVERRUN
; CHARACTER LENGTH: 8 BITS
; LINE SPEED: 9600 BAUD

003630 TS      \XN, 10, 4$, 1$
003630 012767 000340 174140 T15: MOV    #340, PS      ; DISABLE ALL INTERRUPTS
003636 012767 000010 006146      MOV    #10, ICOUNT    ; SET UP FOR 10 ITERATIONS
003644 012767 003770 006134      MOV    #4$, ESCAPE    ; SET UP TO ESCAPE TO NEXT TEST
      .IF NB <1$>
003652 012767 003720 006130      MOV    #1$, FREEZ1    ; SET UP TO LOOP WITH DATA      ; 3
      .ENDC
      XN=XN+1
003660 012777 004000 006056      MOV    @BIT11, @DHSCR ; MASTER CLEAR INTERFACE
003666 012703 000014      MOV    #14, R3        ; SET UP LINE #
003672 012705 136000      MOV    #14*400+130000, R5 ; SET EXPECTED LINE NUMBER
      ; VALID DATA FLAG,
      ; PARITY ERROR, DATA OVERRUN,
      ; AND EXPECTED DATA
003676 012767 000377 006142      MOV    #377, TDATA    ; ACTUAL DATA TO BE TRANSMITTED
003704 012777 000014 006032      MOV    #14, @DHSCR    ; SELECT LINE 14
003712 012777 073563 006030      MOV    #73563, @DHLPR ; SELECT 8 BITS/CHAR, ODD PARITY HDX,
      ; 9600 BAUD SPEED FOR LINE 14
003720 012777 177777 006026 1$: MOV    # -1, @DHBC      ; TRANSMIT 1 CHARACTER
003726 012777 012046 006016      MOV    @TDATA, @DHBA  ; ADDRESS OF TRANSMITTED DATA
003734 012777 010000 006016      MOV    #10000, @DHBCR ; SET BREAK FOR LINE 14.
003742 012777 010000 006006      MOV    #10000, @DHBAR ; START TRANSMITTER
003750 105777 005770 2$: TSTB   @DHSCR    ; WAIT FOR CHARACTER TO BE RECEIVED
003754 100375      BPL    2$              ;
003756 017701 005764      MOV    @DHNRC, R1      ; GET RECEIVED CHARACTER
003762 020501      CMP    R5, R1         ; COMPARE EXPECTED AND RECEIVED DATA
003764 001401      BEQ   .+4            ;
003766      HLT    3          ; DATA ERROR
003766 104003      EMT    3              ;
003770 104400 4$: SCOPE    ; CHECK FOR ITERATIONS, LOOP
000015 LINE=LINE+1
020000 BITX=BITX+BITX

```

003772

POPOVER \LINE,\BITX

```

;RECEIVER PARITY LOGIC TEST ON LINE 15.
;TRANSMIT ONE CHARACTER TO THE RECEIVER
;WITH ODD PARITY SET,BREAK SET,HDX SET.
;THE RECEIVER WILL RECEIVE AN ALL 0'S CHARACTER
;WITH EVEN PARITY AND THEREBY
;CAUSE A PARITY ERROR AND OVERRUN
;CHARACTER LENGTH: 8 BITS
;LINE SPEED: 9600 BAUD

```

```

003772 012767 000340 173776 TS \XN,10,4#,1#
003772 012767 000010 006004 T16: MOV #340,PS ;DISABLE ALL INTERRUPTS
004000 012767 000010 006004 MOV #10,ICOUNT ;SET UP FOR 10 ITERATIONS
004006 012767 004132 005772 MOV #4#,ESCAPE ;SET UP TO ESCAPE TO NEXT TEST

004014 012767 004062 005766 .IF NB <1#>
MOV #1#,FREEZ1 ;SET UP TO LOOP WITH DATA ; 3
.ENDC
XN=XN+1

004022 012777 004000 005714 MOV #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
004030 012703 000015 MOV #15,R3 ;SET UP LINE #
004034 012705 136400 MOV #15*400+130000,R5 ;SET EXPECTED LINE NUMBER
;VALID DATA FLAG,
;PARITY ERROR,DATA OVERRUN,
;AND EXPECTED DATA

004040 012767 000377 006000 MOV #377,TDATA ;ACTUAL DATA TO BE TRANSMITTED
004046 012777 000015 005670 MOV #15,@DHSCR ;SELECT LINE 15
004054 012777 073563 005666 MOV #73563,@DHLPR ;SELECT 8 BITS/CHAR,ODD PARITY HDX,
;9600 BAUD SPEED FOR LINE 15

004062 012777 177777 005664 1#: MOV #-1,@DHBC ;TRANSMIT 1 CHARACTER
004070 012777 012046 005654 MOV #TDATA,@DHBA ;ADDRESS OF TRANSMITTED DATA
004076 012777 020000 005654 MOV #20000,@DHBCR ;SET BREAK FOR LINE 15.
004104 012777 020000 005644 MOV #20000,@DHBAR ;START TRANSMITTER
004112 105777 005626 2#: TSTB @DHSCR ;WAIT FOR CHARACTER TO BE RECEIVED
004116 100375 BPL 2# ;
004120 017701 005622 MOV @DHNR,R1 ;GET RECEIVED CHARACTER
004124 020501 CMP R5,R1 ;COMPARE EXPECTED AND RECEIVED DATA
004126 001401 BEQ .+4 ;
004130 HLT 3 ;DATA ERROR
004130 104003 EMT 3 ;
004132 104400 4#: SCOPE ;CHECK FOR ITERATIONS,LOOP
000016 LINE=LINE+1
040000 BITX=BITX+BITX
004134 POPOVER \LINE,\BITX

```

```

;RECEIVER PARITY LOGIC TEST ON LINE 16.
;TRANSMIT ONE CHARACTER TO THE RECEIVER
;WITH ODD PARITY SET,BREAK SET,HDX SET.
;THE RECEIVER WILL RECEIVE AN ALL 0'S CHARACTER
;WITH EVEN PARITY AND THEREBY
;CAUSE A PARITY ERROR AND OVERRUN
;CHARACTER LENGTH: 8 BITS
;LINE SPEED: 9600 BAUD

```

```

004134 012767 000340 173634 TS \XN,10,4#,1#
004134 012767 000010 005642 T17: MOV #340,PS ;DISABLE ALL INTERRUPTS
004142 012767 000010 005642 MOV #10,ICOUNT ;SET UP FOR 10 ITERATIONS

```

```

004150 012767 004274 005630      MOV      #44,ESCAPE      ;SET UP TO ESCAPE TO NEXT TEST
                                .IF NB <14>
004156 012767 004224 005624      MOV      #14,FREEZ1     ;SET UP TO LOOP WITH DATA      ; 3
                                .ENDC
                                XN=XN+1
004164 012777 004000 005552      MOV      #BIT11,0DHSCR  ;MASTER CLEAR INTERFACE
004172 012703 000016 000016      MOV      #16,R3        ;SET UP LINE #
004176 012705 137000 000000      MOV      #16*400+130000,R5 ;SET EXPECTED LINE NUMBER
                                ;VALID DATA FLAG,
                                ;PARITY ERROR,DATA OVERRUN,
                                ;AND EXPECTED DATA
004202 012767 000377 005636      MOV      #377,TDATA    ;ACTUAL DATA TO BE TRANSMITTED
004210 012777 000016 005526      MOV      #16,0DHSCR    ;SELECT LINE 16
004216 012777 073563 005524      MOV      #73563,0DHLPR ;SELECT 8 BITS/CHAR,ODD PARITY HDX,
                                ;9600 BAUD SPEED FOR LINE 16
004224 012777 177777 005522 14:  MOV      #-1,0DHBC     ;TRANSMIT 1 CHARACTER
004232 012777 012046 005512      MOV      #TDATA,0DHBA  ;ADDRESS OF TRANSMITTED DATA
004240 012777 040000 005512      MOV      #40000,0DHBCR ;SET BREAK FOR LINE 16.
004246 012777 040000 005502      MOV      #40000,0DHBR  ;START TRANSMITTER
004254 105777 005464 000000 24:  TSTB    0DHSCR        ;WAIT FOR CHARACTER TO BE RECEIVED
004260 100375 000000 000000      BPL      24            ;
004262 017701 005460 000000      MOV      #0DHNR,R1     ;GET RECEIVED CHARACTER
004266 020501 000000 000000      CMP      R5,R1        ;COMPARE EXPECTED AND RECEIVED DATA
004270 001401 000000 000000      BEQ      .+4          ;
004272 000000 000000 000000      HLT      3            ;DATA ERROR
004272 104003 000000 000000      ENT      3            ;
004274 104400 000017 000000 44:  SCOPE   ;CHECK FOR ITERATIONS,LOOP
                                LINE=LINE+1
                                BITX=BITX+BITX
004276 100000 000000 000000      POPOVER \LINE,\BITX

                                ;RECEIVER PARITY LOGIC TEST ON LINE 17.
                                ;TRANSMIT ONE CHARACTER TO THE RECEIVER
                                ;WITH ODD PARITY SET,BREAK SET,HDX SET.
                                ;THE RECEIVER WILL RECEIVE AN ALL 0'S CHARACTER
                                ;WITH EVEN PARITY AND THEREBY
                                ;CAUSE A PARITY ERROR AND OVERRUN
                                ;CHARACTER LENGTH: 8 BITS
                                ;LINE SPEED: 9600 BAUD

004276 012767 000340 173472  TS      \XN,10,44,14
004276 012767 000010 005500  T20:   MOV      #340,PS      ;DISABLE ALL INTERRUPTS
004304 012767 000010 005500      MOV      #10,ICOUNT   ;SET UP FOR 10 ITERATIONS
004312 012767 004436 005466      MOV      #44,ESCAPE  ;SET UP TO ESCAPE TO NEXT TEST
                                .IF NB <14>
004320 012767 004366 005462      MOV      #14,FREEZ1   ;SET UP TO LOOP WITH DATA      ; 3
                                .ENDC
                                XN=XN+1
004326 012777 004000 005410      MOV      #BIT11,0DHSCR ;MASTER CLEAR INTERFACE
004334 012703 000017 000017      MOV      #17,R3       ;SET UP LINE #
004340 012705 137400 000000      MOV      #17*400+130000,R5 ;SET EXPECTED LINE NUMBER
                                ;VALID DATA FLAG,
                                ;PARITY ERROR,DATA OVERRUN,
                                ;AND EXPECTED DATA
004344 012767 000377 005474      MOV      #377,TDATA   ;ACTUAL DATA TO BE TRANSMITTED
004352 012777 000017 005364      MOV      #17,0DHSCR   ;SELECT LINE 17
004360 012777 073563 005362      MOV      #73563,0DHLPR ;SELECT 8 BITS/CHAR,ODD PARITY HDX.

```



```

004366 012777 177777 005360 1#: MOV @-1,@DHBC ;9600 BAUD SPEED FOR LINE 17
004374 012777 012046 005350 MOV @TDATA,@DHBA ;TRANSMIT 1 CHARACTER
004402 012777 100000 005350 MOV @100000,@DHBCR ;ADDRESS OF TRANSMITTED DATA
004410 012777 100000 005340 MOV @100000,@DHBCR ;SET BREAK FOR LINE 17.
004416 105777 005322 2#: MOV @100000,@DHBCR ;START TRANSMITTER
004422 100375 BPL 2# ;WAIT FOR CHARACTER TO BE RECEIVED
004424 017701 005316 MOV @DHNR,R1 ;GET RECEIVED CHARACTER
004430 020501 CMP R5,R1 ;COMPARE EXPECTED AND RECEIVED DATA
004432 001401 BEQ .+4
004434 HLT 3 ;DATA ERROR
004434 104003 CMT 3
004436 104400 4#: SCOPE ;CHECK FOR ITERATIONS,LOOP
14 004440 000020 LINE=LINE+1
000000 BITX=BITX+BITX
SUPERCALIFRIGILISTICEXPEHALIDOSIS 27363.0DD,8

;MULTI-LINE PARITY DATA TEST
;TRANSMIT A BINARY COUNT WITH ODD PARITY
;PATTERN ON ALL LINES.
;SILO ALARM LEVEL SET TO 0
;RECEIVER WILL BE SERVICED ON A PER CHARACTER BASIS
;IN INTERRUPT MODE.
;TRANSMITTER INTERRUPT WILL BE DISABLED
;LINE SPEED: 2400 BAUD
;CHARACTER LENGTH: 8 * PARITY

004440 TS \XN,10,11#
004440 012767 000340 173330 T21: MOV @340,PS ;DISABLE ALL INTERRUPTS
004446 012767 000010 005336 MOV @10,ICOUNT ;SET UP FOR 10 ITERATIONS
004454 012767 004706 005324 MOV @11#,ESCAPE ;SET UP TO ESCAPE TO NEXT TEST
;IF NB <>
MOV @,FREEZ1 ;SET UP TO LOOP WITH DATA ; 3
.ENDC
XN=XN+1
004462 000022 012777 004000 005254 MOV @BIT11,@DHSCR ;MASTER CLEAR INTERFACE
004470 012700 000020 MOV @16.,R0 ;SET UP TO START 16. LINES
004474 005001 CLR R1 ;COUNT AND BUS ADDRESS MEMORIES
;AND RECEIVED DATA STORAGE
004476 012702 000200 MOV @200,R2 ;SET UP TO LOAD HIGH BYTE
004502 012703 000001 MOV @1,R3 ;OF EXPECTED DATA
004506 012777 012050 005236 1#: MOV @TBUF,@DHBA ;LOAD BUSS ADDRESS
;IF IDN 8,8
MOV @-400,@DHBC ;LOAD BYTE COUNT
.ENDC
;IF IDN 7,8
MOV @-200,@DHBC ;LOAD BYTE COUNT
.ENDC
;IF IDN 6,8
MOV @-100,@DHBC ;LOAD BYTE COUNT
.ENDC
;IF IDN 5,8
MOV @-40,@DHBC ;LOAD BYTE COUNT
.ENDC
004522 012777 027363 005220 MOV @27363,@DHLPR ;SET LINE SPEED AND
;CHARACTER LENGTH 8 * ODD PARITY
004530 105061 012450 CLR RBUF(R1) ;CLEAR NEXT RECEIVED CHARACTER

```

```

004534 110263 012450      MOVB   R2,RBUF(R3)      ;LOAD HIGH BYTE
004540 005277 005200      INC    @DHSCR          ;ADVANCE TO NEXT LINE
004544 005202              INC    R2              ;UPDATE POINTER TO
004546 062701 000002      ADD    @2,R1          ;LOW AND HIGH BYTE OF
004552 062703 000002      ADD    @2,R3          ;NEXT EXPECTED CHARACTER
004556 005300              DEC    R0              ;CONTINUE IF ALL LINES
004560 001352              BNE    1$             ;NOT SET UP
004562 012777 004622 005176  MOV    @6$,@DHRVEC     ;SET UP POINTER FOR
004570 012777 000240 005172  MOV    @240,@DHRLVL    ;RECEIVER INTERRUPT
004576 012777 000100 005140  MOV    @100,@DHSCR     ;ENABLE RECEIVER INTERRUPT
004604 012777 177777 005144  MOV    @-1,@DHBAR      ;SET ALL BAR BITS
004612 005067 173160      CLR    PS              ;ENABLE INTERRUPTS
004616 000167 000076      JMP    12$            ;GO TO TRANSMITTER WAITING ROUTINE

;RECEIVER INTERRUPT SERVICE ROUTINE
;CHECK FOR RECEIVER DONE AND NO ERRORS
;CHECK RECEIVED DATA FOR "VALID DATA" FLAG AND NO ERRORS
;VERIFY THAT RECEIVED DATA IS CORRECT
;CHECK FOR END OF PASS

004622 105777 005116      6$:   TSTB   @DHSCR          ;IF CHARACTER AVAILABLE NOT SET, ERROR
004626 100026              BPL    10$            ;IF SILO OVERRUN SET, ERROR
004630 032777 040000 005106  BIT    @40000,@DHSCR   ;IF SILO OVERRUN SET, ERROR
004636 001401              BEQ    .+4            ;SILO OVERRUN, ERROR
004640              HLT    2              ;SILO OVERRUN, ERROR
004640              EMT    2
004642 017701 005100      7$:   MOV    @DHNRC,R1      ;READ CHARACTER FROM SILO
004646 010102              MOV    R1,R2          ;EXTRACT LINE NUMBER
004650 000302              SWAB   R2
004652 042702 177760      BIC    @177760,R2
004656 010203              MOV    R2,R3          ;SAVE LINE NUMBER
004660 006302              ASL    R2              ;USE LINE NUMBER AS OFFSET
004662 026201 012450      CMP    RBUF(R2),R1    ;COMPARE EXPECTED AND RECEIVED DATA
004666 001403              BEQ    .+10           ;GET EXPECTED DATA
004670 016205 012450      MOV    RBUF(R2),R5    ;DATA ERROR
004674              HLT    3
004674              EMT    3
004676 105262 012450      INCB   RBUF(R2)       ;UPDATE EXPECTED CHARACTER
004702 000002              RTI
004704              HLT    1              ;CONTINUE
004704 104001              10$:  EMT    1              ;CHARACTER AVAILABLE NOT SET, ERROR
004706 022626              11$:  POP2SP
004710 012777 004000 005026  MOV    @BIT11,@DHSCR   ;RESTORE STACK
004716 000411              BR     13$            ;MASTER CLEAR INTERFACE
                                ;RESTART TEST

;WAIT FOR ALL BAR BITS TO CLEAR
;WHEN ALL BAR BITS HAVE CLEARED,
;WAIT FOR SILO TO EMPTY
;WHEN SILO IS EMPTY, SCOPE ON TEST

004720 005777 005032      12$:  TST    @DHBAR          ;WAIT FOR ALL BAR BITS TO CLEAR
004724 001375              BNE    .-4            ;WAIT FOR SILO TO EMPTY
004726 105777 005032      TSTB   @DHSLR
004732 001375              BNE    .-4
004734 012767 000340 173034  MOV    @340,PS        ;PREVENT INTERRUPTS
004742 104400              13$:  SCOPE                ;CHECK FOR ITERATIONS, LOOP

```

15 004744

SUPERCALIFRIGILISTICEXPEHALIDOSIS

27323,EVEN,8

```

;MULTI-LINE PARITY DATA TEST
;TRANSMIT A BINARY COUNT WITH EVEN PARITY
;PATTERN ON ALL LINES.
;SILO ALARM LEVEL SET TO 0
;RECEIVER WILL BE SERVICED ON A PER CHARACTER BASIS
;IN INTERRUPT MODE.
;TRANSMITTER INTERRUPT WILL BE DISABLED
;LINE SPEED: 2400 BAUD
;CHARACTER LENGTH: 8 + PARITY

```

```

004744 000023 TS \XN,10,11$
004744 012767 000340 173024 T22: MOV #340,PS ;DISABLE ALL INTERRUPTS
004752 012767 000010 005032 MOV #10,ICOUNT ;SET UP FOR 10 ITERATIONS
004760 012767 005212 005020 MOV #11$,ESCAPE ;SET UP TO ESCAPE TO NEXT TEST
;IF NB <>
MOV #,FREEZ1 ;SET UP TO LOOP WITH DATA ; 3
.ENDC
XN=XN+1
004766 012777 004000 004750 MOV #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
004774 012700 000020 MOV #16.,R0 ;SET UP TO START 16. LINES
005000 005001 CLR R1 ;COUNT AND BUS ADDRESS MEMORIES
;AND RECEIVED DATA STORAGE
005002 012702 000200 MOV #200,R2 ;SET UP TO LOAD HIGH BYTE
005006 012703 000001 MOV #1,R3 ;OF EXPECTED DATA
005012 012777 012050 004732 1$: MOV #TBUF,@DHBA ;LOAD BUSS ADDRESS
;IF IDN 8,8
MOV #-400,@DHBC ;LOAD BYTE COUNT
.ENDC
;IF IDN 7,8
MOV #-200,@DHBC ;LOAD BYTE COUNT
.ENDC
;IF IDN 6,8
MOV #-100,@DHBC ;LOAD BYTE COUNT
.ENDC
;IF IDN 5,8
MOV #-40,@DHBC ;LOAD BYTE COUNT
.ENDC
005026 012777 027323 004714 MOV #27323,@DHLPR ;SET LINE SPEED AND
;CHARACTER LENGTH 8 + EVEN PARITY
005034 105061 012450 CLRB RBUF(R1) ;CLEAR NEXT RECEIVED CHARACTER
005040 110263 012450 MOVB R2,RBUF(R3) ;LOAD HIGH BYTE
005044 005277 004674 INC @DHSCR ;ADVANCE TO NEXT LINE
005050 005202 INC R2 ;UPDATE POINTER TO
005052 062701 000002 ADD #2,R1 ;LOW AND HIGH BYTE OF
005056 062703 000002 ADD #2,R3 ;NEXT EXPECTED CHARACTER
005062 005300 DEC R0 ;CONTINUE IF ALL LINES
005064 001352 BNE 1$ ;NOT SET UP
005066 012777 005126 004672 MOV #6,@DHRVEC ;SET UP POINTER FOR
005074 012777 000240 004666 MOV #240,@DHRLVL ;RECEIVER INTERRUPT
005102 012777 000100 004634 MOV #100,@DHSCR ;ENABLE RECEIVER INTERRUPT
005110 012777 177777 004640 MOV #-1,@DHBAR ;SET ALL BAR BITS
005116 005067 172654 CLR PS ;ENABLE INTERRUPTS
005122 000167 000076 JMP 12$ ;GO TO TRANSMITTER WAITING ROUTINE

```

;RECEIVER INTERRUPT SERVICE ROUTINE

```

;CHECK FOR RECEIVER DONE AND NO ERRORS
;CHECK RECEIVED DATA FOR "VALID DATA" FLAG AND NO ERRORS
;VERIFY THAT RECEIVED DATA IS CORRECT
;CHECK FOR END OF PASS

```

```

005126 105777 004612      6$:  TSTB   @DHSCR           ;IF CHARACTER AVAILABLE NOT SET, ERROR
005132 100026                BPL    10$
005134 032777 040000 004602  BIT    @40000,@DHSCR       ;IF SILO OVERRUN SET, ERROR
005142 001401                BEQ    .+4
005144                HLT    2           ;SILO OVERRUN, ERROR
005144 104002                EMT    2
005146 017701 004574      7$:  MOV    @DHNRC,R1           ;READ CHARACTER FROM SILO
005152 010102                MOV    R1,R2              ;EXTRACT LINE NUMBER
005154 000302                SWAB   R2
005156 042702 177760      BIC    @177760,R2
005162 010203                MOV    R2,R3           ;SAVE LINE NUMBER
005164 006302                ASL    R2              ;USE LINE NUMBER AS OFFSET
005166 026201 012450      CMP    RBUF(R2),R1       ;COMPARE EXPECTED AND RECEIVED DATA
005172 001403                BEQ    .+10
005174 016205 012450      MOV    RBUF(R2),R5      ;GET EXPECTED DATA
005200                HLT    3           ;DATA ERROR
005200 104003                EMT    3
005202 105262 012450      INCB   RBUF(R2)         ;UPDATE EXPECTED CHARACTER
005206 000002                RTI
005210                10$:  HLT    1           ;CONTINUE
005210 104001                EMT    1              ;CHARACTER AVAILABLE NOT SET, ERROR
005212 022626                11$:  POP2SP          ;RESTORE STACK
005214 012777 004000 004522  MOV    @BIT11,@DHSCR     ;MASTER CLEAR INTERFACE
005222 000411                BR     13$            ;RESTART TEST

```

```

;WAIT FOR ALL BAR BITS TO CLEAR
;WHEN ALL BAR BITS HAVE CLEARED,
;WAIT FOR SILO TO EMPTY
;WHEN SILO IS EMPTY, SCOPE ON TEST

```

```

005224 005777 004526      12$:  TST    @DHBAR           ;WAIT FOR ALL BAR BITS TO CLEAR
005230 001375                BNE    .-4
005232 105777 004526      TSTB   @DHSLR           ;WAIT FOR SILO TO EMPTY
005236 001375                BNE    .-4
005240 012767 000340 172530  MOV    @340,PS ;PREVENT INTERRUPTS
005246 104400                13$:  SCOPE              ;CHECK FOR ITERATIONS, LOOP
16 005250 SUPERCALIFRIGILISTICEXPEHALIDOSIS 27362,000,7

```

```

;MULTI-LINE PARITY DATA TEST
;TRANSMIT A BINARY COUNT WITH ODD PARITY
;PATTERN ON ALL LINES.
;SILO ALARM LEVEL SET TO 0
;RECEIVER WILL BE SERVICED ON A PER CHARACTER BASIS
;IN INTERRUPT MODE.
;TRANSMITTER INTERRUPT WILL BE DISABLED
;LINE SPEED: 2400 BAUD
;CHARACTER LENGTH: 7 + PARITY

```

```

005250                TS \XN,10,11$
005250 012767 000340 172520 T23:  MOV    @340,PS           ;DISABLE ALL INTERRUPTS
005256 012767 000010 004526  MOV    @10,ICOUNT       ;SET UP FOR 10 ITERATIONS

```

```

005264 012767 005516 004514      .IF NB  MOV    #11$,ESCAPE      ;SET UP TO ESCAPE TO NEXT TEST
                                       <>
                                       MOV    #,FREEZ1          ;SET UP TO LOOP WITH DATA      ; 3
                                       .ENDC
                                       XN=XN+1
005272 012777 004000 004444      MOV    #BIT11,@DHSCR      ;MASTER CLEAR INTERFACE
005300 012700 000020                MOV    #16.,R0           ;SET UP TO START 16. LINES
005304 005001                CLR    R1                ;COUNT AND BUS ADDRESS MEMORIES
                                       ;AND RECEIVED DATA STORAGE
005306 012702 000200                MOV    #200,R2          ;SET UP TO LOAD HIGH BYTE
005312 012703 000001                MOV    #1,R3           ;OF EXPECTED DATA
005316 012777 012050 004426      1$:  MOV    #TBUF,@DHBA     ;LOAD BUSS ADDRESS
                                       .IF  IDN    8,7
                                       MOV    #-400,@DHBC        ;LOAD BYTE COUNT
                                       .ENDC
005324 012777 177600 004422      .IF  IDN    7,7
                                       MOV    #-200,@DHBC        ;LOAD BYTE COUNT
                                       .ENDC
                                       .IF  IDN    6,7
                                       MOV    #-100,@DHBC       ;LOAD BYTE COUNT
                                       .ENDC
                                       .IF  IDN    5,7
                                       MOV    #-40,@DHBC        ;LOAD BYTE COUNT
                                       .ENDC
005332 012777 027362 004410      MOV    #27362,@DHLPR     ;SET LINE SPEED AND
                                       ;CHARACTER LENGTH 7 + ODD PARITY
005340 105061 012450                CLRB   RBUF(R1)         ;CLEAR NEXT RECEIVED CHARACTER
005344 110263 012450                MOV    R2,RBUF(R3)      ;LOAD HIGH BYTE
005350 005277 004370                INC    @DHSCR           ;ADVANCE TO NEXT LINE
005354 005202                INC    R2              ;UPDATE POINTER TO
005356 062701 000002                ADD    #2,R1           ;LOW AND HIGH BYTE OF
005362 062703 000002                ADD    #2,R3           ;NEXT EXPECTED CHARACTER
005366 005300                DEC    R0              ;CONTINUE IF ALL LINES
005370 001352                BNE   1$              ;NOT SET UP
005372 012777 005432 004366      MOV    #6$,@DHRVEC      ;SET UP POINTER FOR
005400 012777 000240 004362      MOV    #240,@DHRLVL     ;RECEIVER INTERRUPT
005406 012777 000100 004330      MOV    #100,@DHSCR      ;ENABLE RECEIVER INTERRUPT
005414 012777 177777 004334      MOV    #-1,@DHBAR       ;SET ALL BAR BITS
005422 005067 172350                CLR    PS              ;ENABLE INTERRUPTS
005426 000167 000076                JMP    12$             ;GO TO TRANSMITTER WAITING ROUTINE

;RECEIVER INTERRUPT SERVICE ROUTINE
;CHECK FOR RECEIVER DONE AND NO ERRORS
;CHECK RECEIVED DATA FOR "VALID DATA" FLAG AND NO ERRORS
;VERIFY THAT RECEIVED DATA IS CORRECT
;CHECK FOR END OF PASS

005432 105777 004306                6$:  TSTB   @DHSCR          ;IF CHARACTER AVAILABLE NOT SET, ERROR
005436 100026                BPL   10$
005440 032777 040000 004276      BIT    #40000,@DHSCR     ;IF SILO OVERRUN SET, ERROR
005446 001401                BEQ   .+4
005450                HLT   2                ;SILO OVERRUN, ERROR
005450                EMT   2
005452 017701 004270                7$:  MOV    @DHNRC,R1        ;READ CHARACTER FROM SILO
005456 010102                MOV    R1,R2           ;EXTRACT LINE NUMBER
005460 000302                SWAB  R2
005462 042702 177760                BIC   #177760,R2

```

```

005466 010203      MOV      R2,R3      ;SAVE LINE NUMBER
005470 006302      ASL      R2          ;USE LINE NUMBER AS OFFSET
005472 026201 012450  CMP      RBUF(R2),R1 ;COMPARE EXPECTED AND RECEIVED DATA
005476 001403      BEQ      .+10
005500 016205 012450  MOV      RBUF(R2),R5 ;GET EXPECTED DATA
005504      HLT      3          ;DATA ERROR
005504 104003      EMT      3
005506 105262 012450  INCB     RBUF(R2)    ;UPDATE EXPECTED CHARACTER
005512 000002      RTI
005514      HLT      1          ;CONTINUE
005514 104001 10#      EMT      1          ;CHARACTER AVAILABLE NOT SET, ERROR
005516 022626 11#      POP2SP
005520 012777 004000 004216  MOV      @BIT11,@DHSCR ;RESTORE STACK
005526 000411      BR       13#        ;MASTER CLEAR INTERFACE
                                ;RESTART TEST

```

```

;WAIT FOR ALL BAR BITS TO CLEAR
;WHEN ALL BAR BITS HAVE CLEARED,
;WAIT FOR SILO TO EMPTY
;WHEN SILO IS EMPTY, SCOPE ON TEST

```

```

005530 005777 004222 12#      TST      @DHBAR      ;WAIT FOR ALL BAR BITS TO CLEAR
005534 001375      BNE     .-4
005536 105777 004222  TSTB    @DHSLR      ;WAIT FOR SILO TO EMPTY
005542 001375      BNE     .-4
005544 012767 000340 172224  MOV      @340,PS ;PREVENT INTERRUPTS
17 005552 104400 13#      SCOPE          ;CHECK FOR ITERATIONS, LOOP
                                27322,EVEN,7
005554      SUPERCALIFRIGILISTICEXPEHALIDOSIS

```

```

;MULTI-LINE PARITY DATA TEST
;TRANSMIT A BINARY COUNT WITH EVEN PARITY
;PATTERN ON ALL LINES.
;SILO ALARM LEVEL SET TO 0
;RECEIVER WILL BE SERVICED ON A PER CHARACTER BASIS
;IN INTERRUPT MODE.
;TRANSMITTER INTERRUPT WILL BE DISABLED
;LINE SPEED: 2400 BAUD
;CHARACTER LENGTH: 7 + PARITY

```

```

005554      TS \XN,10,11#
005554 012767 000340 172214 T24:    MOV      @340,PS      ;DISABLE ALL INTERRUPTS
005562 012767 000010 004222      MOV      @10,ICOUNT  ;SET UP FOR 10 ITERATIONS
005570 012767 006022 004210      MOV      @11#,ESCAPE ;SET UP TO ESCAPE TO NEXT TEST
                                .IF NB <>
                                MOV      @,FREEZ1      ;SET UP TO LOOP WITH DATA ; 3
                                .ENDC
                                XN=XN+1
005576 012777 004000 004140      MOV      @BIT11,@DHSCR ;MASTER CLEAR INTERFACE
005604 012700 000020      MOV      @16.,R0     ;SET UP TO START 16. LINES
005610 005001      CLR      R1          ;COUNT AND BUS ADDRESS MEMORIES
                                ;AND RECEIVED DATA STORAGE
005612 012702 000200      MOV      @200,R2     ;SET UP TO LOAD HIGH BYTE
005616 012703 000001      MOV      @1,R3       ;OF EXPECTED DATA
005622 012777 012050 004122 1#      MOV      @TBUF,@DHBA ;LOAD BUSS ADDRESS
                                .IF
                                IDN      8,7
                                MOV      @-400,@DHBC    ;LOAD BYTE COUNT
                                .ENDC

```

```

005630 012777 177600 004116 .IF IDN 7,7
MOV #200, @DHBC ;LOAD BYTE COUNT
.ENDC
.IF IDN 6,7
MOV #100, @DHBC ;LOAD BYTE COUNT
.ENDC
.IF IDN 5,7
MOV #40, @DHBC ;LOAD BYTE COUNT
.ENDC
005636 012777 027322 004104 MOV #27322, @DHLPR ;SET LINE SPEED AND
;CHARACTER LENGTH 7 + EVEN PARITY
005644 105061 012450 CLR R1 ;CLEAR NEXT RECEIVED CHARACTER
005650 110263 012450 MOV R2, R1 ;LOAD HIGH BYTE
005654 005277 004064 INC @DHSCR ;ADVANCE TO NEXT LINE
005660 005202 INC R2 ;UPDATE POINTER TO
005662 062701 000002 ADD #2, R1 ;LOW AND HIGH BYTE OF
005666 062703 000002 ADD #2, R3 ;NEXT EXPECTED CHARACTER
005672 005300 DEC R0 ;CONTINUE IF ALL LINES
005674 001352 BNE 1$ ;NOT SET UP
005676 012777 005736 004062 MOV #6$, @DHRVEC ;SET UP POINTER FOR
005704 012777 000240 004056 MOV #240, @DHRLVL ;RECEIVER INTERRUPT
005712 012777 000100 004024 MOV #100, @DHSCR ;ENABLE RECEIVER INTERRUPT
005720 012777 177777 004030 MOV #1, @DHBAR ;SET ALL BAR BITS
005726 005067 172044 CLR PS ;ENABLE INTERRUPTS
005732 000167 000076 JMP 12$ ;GO TO TRANSMITTER WAITING ROUTINE

;RECEIVER INTERRUPT SERVICE ROUTINE
;CHECK FOR RECEIVER DONE AND NO ERRORS
;CHECK RECEIVED DATA FOR "VALID DATA" FLAG AND NO ERRORS
;VERIFY THAT RECEIVED DATA IS CORRECT
;CHECK FOR END OF PASS

005736 105777 004002 6$: TSTB @DHSCR ;IF CHARACTER AVAILABLE NOT SET, ERROR
005742 100026 BPL 10$
005744 032777 040000 003772 BIT #40000, @DHSCR ;IF SILO OVERRUN SET, ERROR
005752 001401 BEQ .+4
005754 HLT 2 ;SILO OVERRUN, ERROR
005754 104002 EMT 2
005756 017701 003764 7$: MOV @DHNRC, R1 ;READ CHARACTER FROM SILO
005762 010102 MOV R1, R2 ;EXTRACT LINE NUMBER
005764 000302 SWAB R2
005766 042702 177760 BIC #177760, R2
005772 010203 MOV R2, R3 ;SAVE LINE NUMBER
005774 006302 ASL R2 ;USE LINE NUMBER AS OFFSET
005776 026201 012450 CMP RBUF(R2), R1 ;COMPARE EXPECTED AND RECEIVED DATA
006002 001403 BEQ .+10
006004 016205 012450 MOV RBUF(R2), R5 ;GET EXPECTED DATA
006010 HLT 3 ;DATA ERROR
006010 104003 EMT 3
006012 105262 012450 INCB RBUF(R2) ;UPDATE EXPECTED CHARACTER
006016 000002 RTI ;CONTINUE
006020 104001 10$: HLT 1 ;CHARACTER AVAILABLE NOT SET, ERROR
006020 EMT 1
006022 022626 11$: POP2SP ;RESTORE STACK
006024 012777 004000 003712 MOV #BIT11, @DHSCR ;MASTER CLEAR INTERFACE
006032 000411 BR 13$ ;RESTART TEST

```

```

;WAIT FOR ALL BAR BITS TO CLEAR
;WHEN ALL BAR BITS HAVE CLEARED,
;WAIT FOR SILO TO EMPTY
;WHEN SILO IS EMPTY, SCOPE ON TEST

006034 005777 003716      12$: TST    @DHBAR      ;WAIT FOR ALL BAR BITS TO CLEAR
006040 001375              BNE    .-4
006042 105777 003716      TSTB   @DHSLR      ;WAIT FOR SILO TO EMPTY
006046 001375              BNE    .-4
006050 012767 000340 171720  MOV    #340,PS ;PREVENT INTERRUPTS
006056 104400              13$: SCOPE          ;CHECK FOR ITERATIONS, LOOP
18 006060 SUPERCALIFRIGILISTICEXPEHALIDOSIS 27361,ODD,6

;MULTI-LINE PARITY DATA TEST
;TRANSMIT A BINARY COUNT WITH ODD PARITY
;PATTERN ON ALL LINES.
;SILO ALARM LEVEL SET TO 0
;RECEIVER WILL BE SERVICED ON A PER CHARACTER BASIS
;IN INTERRUPT MODE.
;TRANSMITTER INTERRUPT WILL BE DISABLED
;LINE SPEED: 2400 BAUD
;CHARACTER LENGTH: 6 + PARITY

006060 012767 000340 171710  TS \XN,10,11$
006060 012767 000010 003716  T25:  MOV    #340,PS      ;DISABLE ALL INTERRUPTS
006066 012767 000010 003716  MOV    #10,ICOUNT      ;SET UP FOR 10 ITERATIONS
006074 012767 006326 003704  MOV    #11$,ESCAPE     ;SET UP TO ESCAPE TO NEXT TEST
                          .IF NB <>
                          MOV    @,FREEZ1      ;SET UP TO LOOP WITH DATA      ; 3
                          .ENDC
                          XN=XN+1

006102 000026 004000 003634  MOV    @BIT11,@DHSCR   ;MASTER CLEAR INTERFACE
006110 012700 000020          MOV    #16.,R0        ;SET UP TO START 16. LINES
006114 005001          CLR    R1             ;COUNT AND BUS ADDRESS MEMORIES
                          MOV    #200,R2      ;AND RECEIVED DATA STORAGE
006116 012702 000200          MOV    #1,R3         ;SET UP TO LOAD HIGH BYTE
006122 012703 000001          MOV    #TBUF,@DHBA    ;OF EXPECTED DATA
006126 012777 012050 003616  1$:  MOV    #-400,@DHBC ;LOAD BUSS ADDRESS
                          .IF
                          IDN    8,6          ;LOAD BYTE COUNT
                          MOV    #-200,@DHBC ;LOAD BYTE COUNT
                          .ENDC
                          .IF
                          IDN    7,6          ;LOAD BYTE COUNT
                          MOV    #-100,@DHBC ;LOAD BYTE COUNT
                          .ENDC
                          .IF
                          IDN    6,6          ;LOAD BYTE COUNT
                          MOV    #-40,@DHBC  ;LOAD BYTE COUNT
                          .IF
                          IDN    5,6          ;LOAD BYTE COUNT
                          MOV    #-40,@DHBC
                          .ENDC

006134 012777 177700 003612  MOV    #27361,@DHLPR  ;SET LINE SPEED AND
                          ;CHARACTER LENGTH 6 + ODD PARITY
006142 012777 027361 003600  CLRB   RBUF(R1)       ;CLEAR NEXT RECEIVED CHARACTER
006150 105061 012450          MOVB   R2,RBUF(R3)    ;LOAD HIGH BYTE
006154 110263 012450          INC    @DHSCR        ;ADVANCE TO NEXT LINE
006160 005277 003560          INC    R2            ;UPDATE POINTER TO
006164 005202

```



```

006166 062701 000002      ADD     #2,R1      ;LOW AND HIGH BYTE OF
006172 062703 000002      ADD     #2,R3      ;NEXT EXPECTED CHARACTER
006176 005300              DEC     R0         ;CONTINUE IF ALL LINES
006200 001352              BNE     1$        ;NOT SET UP
006202 012777 006242 003556  MOV     #6$,@DHRVEC ;SET UP POINTER FOR
006210 012777 000240 003552  MOV     #240,@DHRLVL ;RECEIVER INTERRUPT
006216 012777 000100 003520  MOV     #100,@DHSCR  ;ENABLE RECEIVER INTERRUPT
006224 012777 177777 003524  MOV     #-1,@DHBAR   ;SET ALL BAR BITS
006232 005067 171540              CLR     PS         ;ENABLE INTERRUPTS
006236 000167 000076              JMP     12$       ;GO TO TRANSMITTER WAITING ROUTINE

```

```

;RECEIVER INTERRUPT SERVICE ROUTINE
;CHECK FOR RECEIVER DONE AND NO ERRORS
;CHECK RECEIVED DATA FOR "VALID DATA" FLAG AND NO ERRORS
;VERIFY THAT RECEIVED DATA IS CORRECT
;CHECK FOR END OF PASS

```

```

006242 105777 003476      6$:   TSTB   @DHSCR      ;IF CHARACTER AVAILABLE NOT SET, ERROR
006246 100026              BPL     10$       ;IF SILO OVERRUN SET, ERROR
006250 032777 040000 003466  BIT     #40000,@DHSCR
006256 001401              BEQ     .+4       ;SILO OVERRUN, ERROR
006260              HLT     2
006260 104002              EMT     2
006262 017701 003460      7$:   MOV     @DHNRC,R1   ;READ CHARACTER FROM SILO
006266 010102              MOV     R1,R2     ;EXTRACT LINE NUMBER
006270 000302              SWAB   R2
006272 042702 177760      BIC     #177760,R2
006276 010203              MOV     R2,R3     ;SAVE LINE NUMBER
006300 006302              ASL     R2         ;USE LINE NUMBER AS OFFSET
006302 026201 012450      CMP     RBUF(R2),R1 ;COMPARE EXPECTED AND RECEIVED DATA
006306 001403              BEQ     .+10
006310 016205 012450      MOV     RBUF(R2),R5 ;GET EXPECTED DATA
006314              HLT     3
006314              EMT     3
006316 104003              INCB   RBUF(R2)   ;UPDATE EXPECTED CHARACTER
006322 000002              RTI
006324              10$:  HLT     1
006324 104001              EMT     1
006326 022626      11$:  POP2SP ;RESTORE STACK
006330 012777 004000 003406  MOV     #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
006336 000411              BR     13$       ;RESTART TEST

```

```

;WAIT FOR ALL BAR BITS TO CLEAR
;WHEN ALL BAR BITS HAVE CLEARED,
;WAIT FOR SILO TO EMPTY
;WHEN SILO IS EMPTY, SCOPE ON TEST

```

```

006340 005777 003412      12$:  TST     @DHBAR     ;WAIT FOR ALL BAR BITS TO CLEAR
006344 001375              BNE     .-4
006346 105777 003412      TSTB   @DHSLR     ;WAIT FOR SILO TO EMPTY
006352 001375              BNE     .-4
006354 012767 000340 171414  MOV     #340,PS ;PREVENT INTERRUPTS
006362 104400      13$:  SCOPE ;CHECK FOR ITERATIONS, LOOP
19 006364 SUPERCALIFRIGILISTICEXPEHALIDOSIS 27321,EVEN,6

```

```

;MULTI-LINE PARITY DATA TEST

```

```

;TRANSMIT A BINARY COUNT WITH EVEN PARITY
;PATTERN ON ALL LINES.
;SILO ALARM LEVEL SET TO 0
;RECEIVER WILL BE SERVICED ON A PER CHARACTER BASIS
;IN INTERRUPT MODE.
;TRANSMITTER INTERRUPT WILL BE DISABLED
;LINE SPEED: 2400 BAUD
;CHARACTER LENGTH: 6 + PARITY

```

```

006364      006364      006364      006364      TS \XN,10,11#
006364 012767 000340 171404 T26:  MOV    #340,PS           ;DISABLE ALL INTERRUPTS
006372 012767 000010 003412      MOV    #10,ICOUNT        ;SET UP FOR 10 ITERATIONS
006400 012767 006632 003400      MOV    #11$,ESCAPE      ;SET UP TO ESCAPE TO NEXT TEST
                                .IF NB  <>
                                MOV    #,FREEZ1          ;SET UP TO LOOP WITH DATA      ; 3
                                .ENDC
                                XN=XN+1
006406 000027      006406 012777 004000 003330      MOV    #BIT11,&DHSCR     ;MASTER CLEAR INTERFACE
006414 012700 000020      MOV    #16.,R0          ;SET UP TO START 16. LINES
006420 005001      CLR    R1               ;COUNT AND BUS ADDRESS MEMORIES
                                ;AND RECEIVED DATA STORAGE
006422 012702 000200      MOV    #200,R2         ;SET UP TO LOAD HIGH BYTE
006426 012703 000001      MOV    #1,R3           ;OF EXPECTED DATA
006432 012777 012050 003312 1$:  MOV    #TBUF,&DHBA      ;LOAD BUSS ADDRESS
                                .IF
                                MOV    #8,6
                                MOV    #-400,&DHBC        ;LOAD BYTE COUNT
                                .ENDC
                                .IF
                                IDN    7,6
                                MOV    #-200,&DHBC        ;LOAD BYTE COUNT
                                .ENDC
                                .IF
                                IDN    6,6
                                MOV    #-100,&DHBC        ;LOAD BYTE COUNT
                                .ENDC
                                .IF
                                IDN    5,6
                                MOV    #-40,&DHBC         ;LOAD BYTE COUNT
                                .ENDC
006440 012777 177700 003306      MOV    #27321,&DHLP     ;SET LINE SPEED AND
                                ;CHARACTER LENGTH 6 + EVEN PARITY
006446 012777 027321 003274      CLR    RBUF(R1)        ;CLEAR NEXT RECEIVED CHARACTER
006454 105061 012450      MOV    R2,RBUF(R3)     ;LOAD HIGH BYTE
006460 110263 012450      INC    &DHSCR          ;ADVANCE TO NEXT LINE
006464 005277 003254      INC    R2             ;UPDATE POINTER TO
006470 005202      ADD    #2,R1          ;LOW AND HIGH BYTE OF
006472 062701 000002      ADD    #2,R3          ;NEXT EXPECTED CHARACTER
006476 062703 000002      DEC    R0             ;CONTINUE IF ALL LINES
006502 005300      BNE    1$            ;NOT SET UP
006504 001352      MOV    #6$,&DHRVEC     ;SET UP POINTER FOR
006506 012777 006546 003252      MOV    #240,&DHRLVL    ;RECEIVER INTERRUPT
006514 012777 000240 003246      MOV    #100,&DHSCR     ;ENABLE RECEIVER INTERRUPT
006522 012777 000100 003214      MOV    #-1,&DHBAR      ;SET ALL BAR BITS
006530 012777 177777 003220      CLR    PS             ;ENABLE INTERRUPTS
006536 005067 171234      JMP    12$           ;GO TO TRANSMITTER WAITING ROUTINE
006542 000167 000076

```

```

;RECEIVER INTERRUPT SERVICE ROUTINE
;CHECK FOR RECEIVER DONE AND NO ERRORS
;CHECK RECEIVED DATA FOR "VALID DATA" FLAG AND NO ERRORS
;VERIFY THAT RECEIVED DATA IS CORRECT

```

```

;CHECK FOR END OF PASS
006546 105777 003172      6#:  TSTB   @DHSCR           ;IF CHARACTER AVAILABLE NOT SET, ERROR
006552 100026                BPL    10#
006554 032777 040000 003162  BIT    @40000,@DHSCR      ;IF SILO OVERRUN SET, ERROR
006562 001401                BEQ    .+4
006564                HLT    2           ;SILO OVERRUN, ERROR
006564 104002                EMT    2
006566 017701 003154      7#:  MOV    @DHNRC,R1         ;READ CHARACTER FROM SILO
006572 010102                MOV    R1,R2             ;EXTRACT LINE NUMBER
006574 000302                SWAB   R2
006576 042702 177760      BIC    @177760,R2
006602 010203                MOV    R2,R3
006604 006302                ASL    R2
006606 026201 012450      CMP    RBUF(R2),R1       ;SAVE LINE NUMBER
006612 001403                BEQ    .+10             ;USE LINE NUMBER AS OFFSET
006614 016205 012450      MOV    RBUF(R2),R5      ;COMPARE EXPECTED AND RECEIVED DATA
006620                HLT    3
006620 104003                EMT    3
006622 105262 012450      INCB   RBUF(R2)         ;GET EXPECTED DATA
006626 000002                RTI
006630                HLT    1           ;DATA ERROR
006630 104001                EMT    1
006632 022626                POP2SP          ;UPDATE EXPECTED CHARACTER
006634 012777 004000 003102 11#:  MOV    @BIT11,@DHSCR     ;CONTINUE
006642 000411                BR     13#          ;CHARACTER AVAILABLE NOT SET, ERROR

;WAIT FOR ALL BAR BITS TO CLEAR
;WHEN ALL BAR BITS HAVE CLEARED,
;WAIT FOR SILO TO EMPTY
;WHEN SILO IS EMPTY, SCOPE ON TEST
006644 005777 003106      12#:  TST    @DHBAR           ;WAIT FOR ALL BAR BITS TO CLEAR
006650 001375                BNE    .-4
006652 105777 003106      TSTB   @DHSLR          ;WAIT FOR SILO TO EMPTY
006656 001375                BNE    .-4
006660 012767 000340 171110  MOV    @340,PS ;PREVENT INTERRUPTS
006666 104400                SCOPE
20 006670 SUPERCALIFRIGILISTICEXPEHALIDOSIS ;CHECK FOR ITERATIONS, LOOP
;MULTI-LINE PARITY DATA TEST
;TRANSMIT A BINARY COUNT WITH ODD PARITY
;PATTERN ON ALL LINES.
;SILO ALARM LEVEL SET TO 0
;RECEIVER WILL BE SERVICED ON A PER CHARACTER BASIS
;IN INTERRUPT MODE.
;TRANSMITTER INTERRUPT WILL BE DISABLED
;LINE SPEED: 2400 BAUD
;CHARACTER LENGTH: 5 + PARITY
006670                TS \XN,10,11#
006670 012767 000340 171100  T27:  MOV    @340,PS         ;DISABLE ALL INTERRUPTS
006676 012767 000010 003106  MOV    @10,ICOUNT      ;SET UP FOR 10 ITERATIONS
006704 012767 007136 003074  MOV    @11#,ESCAPE     ;SET UP TO ESCAPE TO NEXT TEST
;IF NB <>
MOV    @,FREEZ1         ;SET UP TO LOOP WITH DATA
; 3

```

```

      .ENDC
      XN=XN+1
006712 000030      MOV    #BIT11, @DHSCR      ;MASTER CLEAR INTERFACE
006720 012777 004000 003024      MOV    #16., R0      ;SET UP TO START 16. LINES
006724 005001      CLR    R1      ;COUNT AND BUS ADDRESS MEMORIES
                                ;AND RECEIVED DATA STORAGE
006726 012702 000200      MOV    #200, R2      ;SET UP TO LOAD HIGH BYTE
006732 012703 000001      MOV    #1, R3      ;OF EXPECTED DATA
006736 012777 012050 003006 1$:  MOV    #TBUF, @DHBA      ;LOAD BUSS ADDRESS
                                .IF
                                IDN    8,5
                                MOV    #-400, @DHBC      ;LOAD BYTE COUNT
                                .ENDC
                                .IF
                                IDN    7,5
                                MOV    #-200, @DHBC      ;LOAD BYTE COUNT
                                .ENDC
                                .IF
                                IDN    6,5
                                MOV    #-100, @DHBC      ;LOAD BYTE COUNT
                                .ENDC
                                .IF
                                IDN    5,5
                                MOV    #-40, @DHBC      ;LOAD BYTE COUNT
                                .ENDC
006744 012777 177740 003002      MOV    #27360, @DHLPR      ;SET LINE SPEED AND
                                ;CHARACTER LENGTH 5 + ODD PARITY
006752 012777 027360 002770      CLRB  RBUF(R1)      ;CLEAR NEXT RECEIVED CHARACTER
006760 105061 012450      MOV    R2, RBUF(R3)      ;LOAD HIGH BYTE
006764 110263 012450      INC   @DHSCR      ;ADVANCE TO NEXT LINE
006770 005277 002750      INC   R2      ;UPDATE POINTER TO
006774 005202      ADD   #2, R1      ;LOW AND HIGH BYTE OF
006776 062701 000002      ADD   #2, R3      ;NEXT EXPECTED CHARACTER
007002 062703 000002      DEC   R0      ;CONTINUE IF ALL LINES
007006 005300      BNE   1$      ;NOT SET UP
007010 001352      MOV    #6$, @DHRVEC      ;SET UP POINTER FOR
007012 012777 007052 002746      MOV    #240, @DHRLVL      ;RECEIVER INTERRUPT
007020 012777 000240 002742      MOV    #100, @DHSCR      ;ENABLE RECEIVER INTERRUPT
007026 012777 000100 002710      MOV    #-1, @DHBAR      ;SET ALL BAR BITS
007034 012777 177777 002714      CLR   PS      ;ENABLE INTERRUPTS
007042 005067 170730      JMP   12$      ;GO TO TRANSMITTER WAITING ROUTINE
007046 000167 000076

;RECEIVER INTERRUPT SERVICE ROUTINE
;CHECK FOR RECEIVER DONE AND NO ERRORS
;CHECK RECEIVED DATA FOR "VALID DATA" FLAG AND NO ERRORS
;VERIFY THAT RECEIVED DATA IS CORRECT
;CHECK FOR END OF PASS

007052 105777 002666      6$:  TSTB  @DHSCR      ;IF CHARACTER AVAILABLE NOT SET, ERROR
007056 100026      BPL   10$
007060 032777 040000 002656      BIT   #40000, @DHSCR      ;IF SILO OVERRUN SET, ERROR
007066 001401      BEQ   .+4
007070      HLT   2      ;SILO OVERRUN, ERROR
007070      EMT   2
007072 017701 002650      7$:  MOV    @DHNR, R1      ;READ CHARACTER FROM SILO
007076 010102      MOV    R1, R2      ;EXTRACT LINE NUMBER
007100 000302      SWAB R2
007102 042702 177760      BIC   #177760, R2
007106 010203      MOV    R2, R3      ;SAVE LINE NUMBER
007110 006302      ASL   R2      ;USE LINE NUMBER AS OFFSET
007112 026201 012450      CMP   RBUF(R2), R1      ;COMPARE EXPECTED AND RECEIVED DATA

```

```

007116 001403      BEQ      .+10
007120 016205 012450  MOV      RBUF(R2),R5      ;GET EXPECTED DATA
007124      HLT      3      ;DATA ERROR
007124 104003      EMT      3
007126 105262 012450  INCB     RBUF(R2)      ;UPDATE EXPECTED CHARACTER
007132 000002      RTI
007134      HLT      1      ;CONTINUE
007134 104001 10# :      EMT      1      ;CHARACTER AVAILABLE NOT SET, ERROR
007136 022626 11# :      POP2SP
007140 012777 004000 002576  MOV      @BIT11,@DHSCR    ;RESTORE STACK
007146 000411      BR       13#             ;MASTER CLEAR INTERFACE
                                           ;RESTART TEST

                                           ;WAIT FOR ALL BAR BITS TO CLEAR
                                           ;WHEN ALL BAR BITS HAVE CLEARED,
                                           ;WAIT FOR SILO TO EMPTY
                                           ;WHEN SILO IS EMPTY, SCOPE ON TEST

007150 005777 002602 12# :      TST      @DHBAR      ;WAIT FOR ALL BAR BITS TO CLEAR
007154 001375      BNE     .-4
007156 105777 002602      TSTB     @DHSLR          ;WAIT FOR SILO TO EMPTY
007162 001375      BNE     .-4
007164 012767 000340 170604  MOV      @340,PS ;PREVENT INTERRUPTS
21 007174 104400 13# :      SCOPE          ;CHECK FOR ITERATIONS, LOOP
SUPERCALIFRIGILISTICEXPEHALIDOSIS 27320,EVEN,5

                                           ;MULTI-LINE PARITY DATA TEST
                                           ;TRANSMIT A BINARY COUNT WITH EVEN PARITY
                                           ;PATTERN ON ALL LINES.
                                           ;SILO ALARM LEVEL SET TO 0
                                           ;RECEIVER WILL BE SERVICED ON A PER CHARACTER BASIS
                                           ;IN INTERRUPT MODE.
                                           ;TRANSMITTER INTERRUPT WILL BE DISABLED
                                           ;LINE SPEED: 2400 BAUD
                                           ;CHARACTER LENGTH: 5 + PARITY

007174      TS \XN,10,11#
007174 012767 000340 170574 T30:      MOV      @340,PS      ;DISABLE ALL INTERRUPTS
007202 012767 000010 002602      MOV      @10,ICOUNT    ;SET UP FOR 10 ITERATIONS
007210 012767 007442 002570      MOV      @11#,ESCAPE   ;SET UP TO ESCAPE TO NEXT TEST
                                           .IF NB <>
                                           MOV      @,FREEZ1      ;SET UP TO LOOP WITH DATA      ; 3
                                           .ENDC
XN=XN+1
007216 000031      MOV      @BIT11,@DHSCR    ;MASTER CLEAR INTERFACE
007224 012700 000020      MOV      @16.,R0      ;SET UP TO START 16. LINES
007230 00:001      CLR      R1           ;COUNT AND BUS ADDRESS MEMORIES
                                           ;AND RECEIVED DATA STORAGE
007232 012702 000200      MOV      @200,R2      ;SET UP TO LOAD HIGH BYTE
007236 012703 000001      MOV      @1,R3        ;OF EXPECTED DATA
007242 012777 012050 002502 1# :      MOV      @TBUF,@DHBA   ;LOAD BUSS ADDRESS
                                           .IF
                                           IDN     8,5
                                           MOV     @-400,@DHBC    ;LOAD BYTE COUNT
                                           .ENDC
                                           .IF
                                           IDN     7,5
                                           MOV     @-200,@DHBC   ;LOAD BYTE COUNT
                                           .ENDC

```

```

        .IF      IDN      6,5
        MOV      @-100,@DHBC          ;LOAD BYTE COUNT
        .ENDC
        .IF      IDN      5,5
        MOV      @-40,@DHBC          ;LOAD BYTE COUNT
        .ENDC
007250 012777 177740 002476
007256 012777 027320 002464      MOV      @27320,@DHLPR          ;SET LINE SPEED AND
                                ;CHARACTER LENGTH 5 + EVEN PARITY
007264 105061 012450      CLR      RBUF(R1)          ;CLEAR NEXT RECEIVED CHARACTER
007270 110263 012450      MOV      R2,RBUF(R3)      ;LOAD HIGH BYTE
007274 005277 002444      INC      @DHSCR          ;ADVANCE TO NEXT LINE
007300 005202      INC      R2          ;UPDATE POINTER TO
007302 062701 000002      ADD      @2,R1          ;LOW AND HIGH BYTE OF
007306 062703 000002      ADD      @2,R3          ;NEXT EXPECTED CHARACTER
007312 005300      DEC      R0          ;CONTINUE IF ALL LINES
007314 001352      BNE     1$          ;NOT SET UP
007316 012777 007356 002442      MOV      @6$,@DHRVEC      ;SET UP POINTER FOR
007324 012777 000240 002436      MOV      @240,@DHRLVL     ;RECEIVER INTERRUPT
007332 012777 000100 002404      MOV      @100,@DHSCR      ;ENABLE RECEIVER INTERRUPT
007340 012777 177777 002410      MOV      @-1,@DHBAR      ;SET ALL BAR BITS
007346 005067 170424      CLR      PS          ;ENABLE INTERRUPTS
007352 000167 000076      JMP     12$          ;GO TO TRANSMITTER WAITING ROUTINE

;RECEIVER INTERRUPT SERVICE ROUTINE
;CHECK FOR RECEIVER DONE AND NO ERRORS
;CHECK RECEIVED DATA FOR "VALID DATA" FLAG AND NO ERRORS
;VERIFY THAT RECEIVED DATA IS CORRECT
;CHECK FOR END OF PASS

007356 105777 002362      6$:   TSTB     @DHSCR          ;IF CHARACTER AVAILABLE NOT SET, ERROR
007362 100026      BPL     10$          ;IF SILO OVERRUN SET, ERROR
007364 032777 040000 002352      BIT     @40000,@DHSCR
007372 001401      BEQ     .+4
007374      HLT     2          ;SILO OVERRUN, ERROR
007374 104002      EMT     2
007376 017701 002344      7$:   MOV      @DHNR,R1          ;READ CHARACTER FROM SILO
007402 010102      MOV      R1,R2          ;EXTRACT LINE NUMBER
007404 000302      SWAB   R2
007406 042702 177760      BIC     @177760,R2
007412 010203      MOV      R2,R3          ;SAVE LINE NUMBER
007414 006302      ASL     R2          ;USE LINE NUMBER AS OFFSET
007416 026201 012450      CMP     RBUF(R2),R1      ;COMPARE EXPECTED AND RECEIVED DATA
007422 001403      BEQ     .+10
007424 016205 012450      MOV      RBUF(R2),R5      ;GET EXPECTED DATA
007430      HLT     3          ;DATA ERROR
007430 104003      EMT     3
007432 105262 012450      INCB   RBUF(R2)          ;UPDATE EXPECTED CHARACTER
007436 000002      RTI
007440      HLT     1          ;CHARACTER AVAILABLE NOT SET, ERROR
007440 104001      EMT     1
007442 022626      11$:  POP2SP          ;RESTORE STACK
007444 012777 004000 002272      MOV      @BIT11,@DHSCR    ;MASTER CLEAR INTERFACE
007452 000411      BR     13$          ;RESTART TEST

;WAIT FOR ALL BAR BITS TO CLEAR
;WHEN ALL BAR BITS HAVE CLEARED,

```

```
                                ;WAIT FOR SILO TO EMPTY
                                ;WHEN SILO IS EMPTY, SCOPE ON TEST
007454 005777 002276           12$: TST    @DMBAR                ;WAIT FOR ALL BAR BITS TO CLEAR
007460 001375                BNE    .-4
007462 105777 002276           TSTB   @DHSLR                ;WAIT FOR SILO TO EMPTY
007466 001375                BNE    .-4
007470 012767 000340 170300    MOV    @340,PS ;PREVENT INTERRUPTS
007476 104400           13$:  SCOPE                ;CHECK FOR ITERATIONS, LOOP
```

```

1
2
3
4
5
6
7
8
9
10
11 007500
007500 012767 000340 170270
007506 012767 000010 002276
007514 012767 007746 002264
TS \XN,10,11$
T31: MOV #340,PS ;DISABLE ALL INTERRUPTS
MOV #10,ICOUNT ;SET UP FOR 10 ITERATIONS
MOV #11$,ESCAPE ;SET UP TO ESCAPE TO NEXT TEST
.IF NB <>
MOV #,FREEZ1 ;SET UP TO LOOP WITH DATA ; 3
.ENDC
XN=XN+1
12 007522 000032
007522 012777 004000 002214
13 007530 012700 000020
14 007534 005001
15
16 007536 012702 000200
17 007542 012703 000001
18 007546 012777 012050 002176 1$:
19 007554 012777 177400 002172
20 007562 012777 031403 002160
21
22 007570 105061 012450
23 007574 110263 012450
24 007600 005277 002140
25 007604 005202
26 007606 062701 000002
27 007612 062703 000002
28 007616 005300
29 007620 001352
30 007622 012777 007662 002136
31 007630 012777 000240 002132
32 007636 012777 000100 002100
33 007644 012777 177777 002104
34 007652 005067 170120
35 007656 000167 000076
CLR RBUF(R1) ;CLEAR NEXT RECEIVED CHARACTER
MOV R2,RBUF(R3) ;LOAD HIGH BYTE
INC #DHSCR ;ADVANCE TO NEXT LINE
INC R2 ;UPDATE POINTER TO
ADD #2,R1 ;LOW AND HIGH BYTE OF
ADD #2,R3 ;NEXT EXPECTED CHARACTER
DEC R0 ;CONTINUE IF ALL LINES
BNE 1$ ;NOT SET UP
MOV #6$,#DHRVEC ;SET UP POINTER FOR
MOV #240,#DHLVL ;RECEIVER INTERRUPT
MOV #100,#DHSCR ;ENABLE INTERFACE
MOV #-1,#DHBAR ;SET ALL BAR BITS
CLR PS ;ENABLE INTERRUPTS
JMP 12$ ;GO TO TRANSMITTER WAITING ROUTINE
36
37
38
39
40
41
42
43 007662 105777 002056 6$:
44 007666 100026
45 007670 032777 040000 002046
46 007676 001401
47 007700
007700 104002
48 007702 017701 002040 7$:
49 007706 010102
TSTB #DHSCR ;IF CHARACTER AVAILABLE NOT SET, ERROR
BPL 10$
BIT #40000,#DHSCR ;IF SILO OVERRUN SET, ERROR
BEQ .+4
HLT 2 ;SILO OVERRUN, ERROR
EMT 2
MOV #DHNRC,R1 ;READ CHARACTER FROM SILO
MOV R1,R2 ;EXTRACT LINE NUMBER

```


50	007710	000302			SWAB	R2		
51	007712	042702	177760		BIC	#177760,R2		
52	007716	010203			MOV	R2,R3		;SAVE LINE NUMBER
53	007720	006302			ASL	R2		;USE LINE NUMBER AS OFFSET
54	007722	026201	012450		CMP	RBUF(R2),R1		;COMPARE EXPECTED AND RECEIVED DATA
55	007726	001403			BEQ	.-10		
56	007730	016205	012450		MOV	RBUF(R2),R5		;GET EXPECTED DATA
57	007734				HLT	3		;DATA ERROR
	007734	104003			EMT	3		
58	007736	105262	012450		INCB	RBUF(R2)		;UPDATE EXPECTED CHARACTER
59	007742	000002			RTI			;CONTINUE
60	007744			10\$:	HLT	1		;CHARACTER AVAILABLE NOT SET, ERROR
	007744	104001			EMT	1		
61	007746	022626		11\$:	POP2SP			;RESTORE STACK
62	007750	012777	004000	001766	MOV	#BIT11,#DHSCR		;MASTER CLEAR INTERFACE
63	007756	000406			BR	13\$;RESTART TEST
64								
65								
66								
67								
68								
69								
70								
71	007760	005777	001772	12\$:	TST	#DHBAR		;WAIT FOR ALL BAR BITS TO CLEAR
72	007764	001375			BNE	.-4		
73	007766	105777	001772		TSTB	#DHSLR		;WAIT FOR SILO TO EMPTY
74	007772	001375			BNE	.-4		
75	007774	104400		13\$:	SCOPE			;CHECK FOR ITERATIONS, LOOP

```

1
2
3
4
5
6
7
8
9
10 007776
    007776 012767 000340 167772
    010004 012767 000012 002000
    010012 012767 010344 001766
    TS \XN,12,8#
    T32:  MOV    #340,PS          ;DISABLE ALL INTERRUPTS
          MOV    #12,ICOUNT      ;SET UP FOR 12 ITERATIONS
          MOV    #8#,ESCAPE      ;SET UP TO ESCAPE TO NEXT TEST
    .IF NB <>
          MOV    #,FREEZ1        ;SET UP TO LOOP WITH DATA          ; 3
    .ENDC
    XN=XN+1
11 010020 012777 004000 001716
12 010026 012700 000020
13 010032 005001
14
15 010034 012702 000200
16 010040 012703 000001
17 010044 012777 012050 001700
18 010052 012777 177400 001674
19 010060 012777 033503 001662
20
21 010066 105061 012450
22 010072 110263 012450
23 010076 005277 001642
24 010102 005202
25 010104 062701 000002
26 010110 062703 000002
27 010114 005300
28 010116 001352
29 010120 012777 010210 001644
30 010126 012777 000240 001640
31 010134 012777 010222 001624
32 010142 012777 000240 001620
33 010150 012700 000012
34 010154 005067 001664
35
36 010160 012777 000077 001574
37 010166 012777 030000 001550
38 010174 012777 177777 001554
39 010202 005067 167570
40 010206 000421
41
42
43
44
45 010210 005777 001530
46 010214 100401
47 010216
    010216 104004
48 010220 000002
49
    ;MULTI-LINE DATA TEST
    ;TRANSMIT A BINARY COUNT COUNT PATTERN ON ALL LINES
    ;LINE SPEED IS 9600 BAUD
    ;SILO ARM LEVEL IS SET TO 63 (NO RECEIVER DONE
    ;INTERRUPTS WILL OCCUR)
    ;CONTINUOUSLY ATTEMPT TO READ DATA FROM SILO
    ;IF "VALID DATA" FLAG IS DETECTED,
    ;THE DATA WILL BE CHECKED
    ;MASTER CLEAR INTERFACE
    ;SET UP TO LOAD BYTE
    ;COUNT AND BUS ADDRESS MEMORIES
    ;AND RECEIVED DATA STORAGE
    ;SET UP TO LOAD HIGH BYTE
    ;OF EXPECTED DATA
    ;LOAD BUSS ADDRESS
    ;LOAD BYTE COUNT
    ;SET LINE SPEED AND
    ;CHARACTER LENGTH
    ;CLEAR NEXT RECEIVED CHARACTER
    ;LOAD HIGH BYTE
    ;ADVANCE TO NEXT LINE
    ;UPDATE POINTER TO
    ;LOW AND HIGH BYTE OF
    ;NEXT EXPECTED CHARACTER
    ;CONTINUE IF ALL LINES
    ;NOT SET UP
    ;SET UP POINTER FOR
    ;TRANSMITTER INTERRUPT
    ;SET UP POINTER FOR
    ;RECEIVER INTERRUPT
    ;SET UP TO TRANSMIT
    ;10 (DEDIMAL) BLOCKS OF DATA
    ;ON EACH LINE
    ;ENABLE INTERFACE
    ;SET ALL BAR BITS
    ;ENABLE INTERRUPTS
    ;TRANSFER TO RECEIVER SERVICE
    ;TRANSMITTER INTERRUPT SERVICE ROUTINE
    ;CHECK FOR TRANSMIT DONE AND NO ERRORS
    ;IS TRANSMITTER DONE SET
    ;TRANSMIT DONE NOT SET, ERROR
    ;RETURN

```

```

50                                     ;RECEIVER INTERRUPT SERVICE ROUTINE
51
52 010222 105777 001516      6$:  TSTB   @DHSCR           ;IS CHARACTER AVAILABLE SET
53 010226 100001                                     BPL   .+4
54 010230                                     HLT   5           ;CHARACTER AVAILABLE SET, ERROR
   010230 104005                                     EMT   5
55 010232 032777 040000 001504  BIT   @40000,@DHSCR ;IS SILO OVERRUN SET
56 010240 001402                                     BEQ   .+6
57 010242                                     HLT   2           ;SILO OVERRUN, ERROR
   010242 104002                                     EMT   2
58 010244 000401                                     BR    3$
59 010246                                     HLT   0           ;SPURIOUS INTERRUPT
   010246 104000                                     EMT   0
60                                     ;TEST WILL RESTART
61 010250 000002      3$:  RTI
62
63                                     ;RECEIVER DATA CHECK
64                                     ;GET A CHARACTER FROM THE SILO
65                                     ;IF IT IS VALID DATA, CHECK IT
66                                     ;OTHERWISE, TRY AGAIN
67
68 010252 017701 001470      7$:  MOV    @DHNRC,R1       ;READ NEXT RECEIVED CHARACTER REGISTER
69 010256 005701                                     TST   R1         ;IF VALID DATA BIT NOT SET
70 010260 100374                                     BPL   7$        ;TRY AGAIN
71 010262 010102                                     MOV   R1,R2     ;EXTRACT LINE NUMBER
72 010264 000302                                     SWAB  R2
73 010266 042702 177760      BIC   @177760,R2
74 010272 010203                                     MOV   R2,R3
75 010274 006302                                     ASL   R2
76 010276 126201 012450      CMPB  RBUF(R2),R1 ;DOUBLE LINE NUMBER FOR OFFSET
77 010302 001403                                     BEQ   .+10      ;COMPARE RECEIVED AND EXPECTED RESULTS
78 010304 016205 012450      MOV   RBUF(R2),R5 ;GET EXPECTED DATA
79 010310                                     HLT   3         ;DATA ERROR
   010310 104003                                     EMT   3
80 010312 105262 012450      INCB  RBUF(R2)   ;UPDATE EXPECTED DATA
81 010316 001355                                     BNE   7$        ;CONTINUE IF NOT 0
82 010320 005767 001520      TST   ENDFLG    ;IF ALL LINES HAVE FINISHED
83 010324 001007                                     BNE   8$        ;10 BLOCKS OF DATA, CLEAN UP
84 010326 005703                                     TST   R3
85 010330 001014                                     BNE   9$        ;IF THE LINE IS 0
86 010332 005300                                     DEC   R0
87 010334 001012                                     BNE   9$        ;UPDATE BLOCK COUNT
88 010336 012767 000001 001500  MOV   @1,ENDFLG ;IF LINE 0 HAS TRANSMITTED 10 BLOCKS
89 010344 005777 001406      8$:  TST   @DHBAR    ;SET END FLAG
90 010350 001340                                     BNE   7$        ;IF ALL LINES NOT DONE
91 010352 105777 001406      TSTB  @DHSLR    ;GET MORE CHARACTERS
92 010356 001335                                     BNE   7$        ;IF SILO IS NOT EMPTY
93 010360 000417                                     BR    10$       ;GET REST OF DATA
94 010362 042777 000017 001354  9$:  BIC   @17,@DHSCR ;WHEN EMPTY, SCOPE ON TEST
95 010370 050377 001350      BIS   R3,@DHSCR ;ADDRESS LINE THAT JUST FINISHED
96 010374 012777 012050 001350  MOV   @TBUF,@DHBA ;SET UP BUS ADDRESS AND BYTE COUNT
97 010402 012777 177400 001344  MOV   @-400,@DHBC ;FOR THAT LINE
98 010410 056277 010426 001340  BIS   BARBIT(R2),@DHBAR ;SET BAR BIT FOR THAT LINE
99 010416 000715                                     BR    7$        ;CONTINUE
100 010420 104400      10$:  SCOPE ;CHECK FOR ITERATIONS, LOOP
101 010422 000167 000040      JMP   EOP      ;GO TO END OF PASS ROUTINE
102 010426 000001      BARBIT: 1

```

103	010430	000002	2
104	010432	000004	4
105	010434	000010	10
106	010436	000020	20
107	010440	000040	40
108	010442	000100	100
109	010444	000200	200
110	010446	000400	400
111	010450	001000	1000
112	010452	002000	2000
113	010454	004000	4000
114	010456	010000	10000
115	010460	020000	20000
116	010462	040000	40000
117	010464	100000	100000

1
2 010466

.EOP †/BEGIN/
;END OF PASS
;TYPE NAME OF TEST
;UPDATE PASS COUNT
;CHECK FOR EXIT TO ACT-11
;RESTART TEST

010466	104401			EOP:	TYPE		;TYPE NAME OF TEST	
010470	013127				MEPASS			
010472	005067	001344			CLR	LAST	;CLEAR LAST ERROR PC	
010476	005067	001274			CLR	ERRFLG	;CLEAR ERROR FLAG	
010502	005267	001272			INC	PASCNT	;UPDATE PASS COUNT	
010506	005767	170270			TST	LIGHTS	; ARE WE USING LIGHTS?	: 4
010512	001005				BNE	2‡	; BRANCH IF WE ARE	: 6
010514	104401				TYPE		; TYPE PASCOUNT MESSAGE	: 5
010516	013142				PASTXT			: 5
010520	104402				OCTASC		; PRINT PASSCOUNT	: 4
010522	010560				PASARG			: 4
010524	000403				BR	3‡	; CONTINUE	: 4 ; 6
010526				2‡:				: 4
010526	016767	001246	1702‡C		MOV	PASCNT,LIGHTS	;DISPLAY PASS COUNT	: 4
010534				3‡:				: 4
010534	013701	000042			MOV	@42,R1	;CHECK FOR ACT-11 OR DDP	
010540	001405				BEQ	RESTR	;IF NOT, CONTINUE TESTING	
010542	000005				RESET			
010544	004711			LOGICAL:		JSR PC,(R1)		
010546	000240				NOP			
010550	000240				NOP			
010552	000240				NOP			
010554	000167	170526		RESTR:	JMP	BEGIN		
010560	000001			PASARG:	.WORD	1	; PARAMETERS TO PRINT PASSCOUNT	: 5
010562	006	002			.BYTE	6,2		: 5
010564	012000				.WORD	PASCNT		: 5
3 010566				.SCOPE				
							;CHECK FOR LOOP ON CURRENT TEST	: 3
							;CHECK FOR ITERATION SUPPRESSION	
010566	032777	002000	170204	SCOPER:	BIT	@SW10,@SWR		: 4
010574	001030				BNE	4‡		: 4
010576	032777	040000	170174	1‡:	BIT	@SW14,@SWR		: 4
010604	001021				BNE	3‡		: 4
010606	032777	004000	170164		BIT	@SW11,@SWR		: 4
010614	001006				BNE	2‡		
010616	005267	001172			INC	LPCNT		
010622	026767	001166	001162		CMP	LPCNT,ICOUNT		
010630	001007				BNE	3‡		
010632	005067	001156		2‡:	CLR	LPCNT		
010636	005067	001134			CLR	ERRFLG		
010642	011667	001136			MOV	(SP),RETRN		
010646	000002				RTI			
010650	016716	001130		3‡:	MOV	RETRN,(SP)		
010654	000002				RTI			
010656	005767	001114		4‡:	TST	ERRFLG		
010662	001745				BEQ	1‡		

```

010664 000762          BR      2#
4 010666          .SCOP1

          ;CHECK FOR FREEZE ON CURRENT DATA

010666 032777 001000 170104 SCOP1R: BIT      #SW09,#SWR          ; 4
010674 001402          BEQ      1#
010676 016716 001106          MOV      FREEZ1,(SP)
010702 000002          1#:     RTI
5 010704          .ERROR

          ;ERROR HANDLER

010704 032777 020000 170066 ERRORS: BIT      #SW13,#SWR          ; 4
010712 001055          BNE      HALTS
010714 021667 001122          CMP      (SP),LAST
010720 001404          BEQ      1#
010722 011667 001114          MOV      (SP),LAST
010726 005067 001044          CLR      ERRFLG
010732 104406          1#:     SAV05P
010734 011605          MOV      (SP),R5
010736 162705 000002          SUB      #2,R5
010742 011504          MOV      (R5),R4
010744 006304          ASL      R4
010746 006304          ASL      R4
010750 042704 177001          BIC      #177001,R4
010754 062704 013262          ADD      #ERRTAB,R4
010760 012467 000040          MOV      (R4)+,ERRMSG
010764 011467 000052          MOV      (R4),DATABP
010770 005767 001002          TST      ERRFLG
010774 001403          BEQ      TYPMSG
010776 005767 000040          TST      DATABP
011002 001011          BNE      TYPDAT
011004 104401          TYPMSG: TYPE
011006 013037          MCRLF
011010 104402          OCTASC
011012 011110          ERTABO
011014 012767 000001 000754 MOV      #1,ERRFLG
011022 104401          TYPE
011024 000000          ERRMSG: 0
011026 005767 000010          TYPDAT: TST      DATABP
011032 001404          BEQ      RESREG
011034 104401          TYPE
011036 013037          MCRLF
011040 104402          OCTASC
011042 000000          DATABP: 0
011044 104407          RESREG: RES05
011046 005777 167726          HALTS: TST      #SWR          ; 4
011052 100005          BPL      EXITER
011054 010046          PUSHRO
011056 016600 000002          MOV      2(SP),R0
011062 000000          HALT
011064 012600          POPRO
011066 005267 000710          EXITER: INC      ERRCNT
011072 032777 002000 167700 BIT      #SW10,#SWR          ; 4
011100 001402          BEQ      1#
011102 016716 000700          MOV      ESCAPE,(SP)

```

011106 000002
 011110 000001
 011112 006 002
 011114 012034
 6 011116

1\$: RTI
 ERTAB0: 1
 .BYTE 6,2
 SAVPC
 .TRPSRV
 ;TRAP DISPATCH SERVICE
 ;ARGUMENT OF TRAP IS EXTRACTED
 ;AND USED AS OFFSET TO OBTAIN POINTER
 ;TO SELECTED SUBROUTINE

: 3

011116 011646
 011120 162716 000002
 011124 017616 000000
 011130 006316
 011132 042716 177001
 011136 062716 013202
 011142 017616 000000
 011146 000136
 7 011150

TRPSRV: MOV (SP),-(SP) ;GET PC OF RETURN
 SUB #2,(SP) ;=PC OF TRAP
 MOV #0(SP),(SP) ;GET TRP
 TRPOK: ASL (SP) ;MULTIPLY TRAP ARG BY 2
 BIC #177001,(SP) ;CLEAR UNWANTED BITS
 ADD #TRPTAB,(SP) ;POINTER TO SUBROUTINE ADDRESS
 MOV #0(SP),(SP) ;SUBROUTINE ADDRESS
 JMP #0(SP)+ ;GO TO SUBROUTINE

.TYPER

;TELETYPE OUTPUT ROUTINE

011150 017605 000000
 011154 062716 000002
 011160 105777 000554
 011164 100375
 011166 105715
 011170 001001
 011172 000002
 011174 112577 000542
 011200 000767
 8 011202

TYPER: MOV #0(SP),R5
 ADD #2,(SP)
 1\$: TSTB #TPCSR
 BPL 1\$
 TSTB (R5)
 BNE 2\$
 RTI
 2\$: MOVB (R5)+,#TPDDBR
 BR 1\$

: 3

.INSTRG

;ASCII STRING INPUT ROUTINE

011202 017667 000000 000006
 011210 062716 000002
 011214 104401
 011216 000000
 011220 012704 013224
 011224 012703 000007
 011230 105777 000500
 011234 100375
 011236 117714 000474
 011242 142714 000200
 011246 122427 000015
 011252 001413
 011254 117777 000456 000460
 011262 105777 000452
 011266 100375
 011270 005303
 011272 001356
 011274 104401
 011276 013033
 011300 000745
 011302 000002
 9 011304

INSTRG: MOV #0(SP),MSG
 ADD #2,(SP)
 INSTR1: TYPE
 MSG: 0
 MOV #INBUF,R4
 MOV #7,R3
 1\$: TSTB #TKCSR
 BPL 1\$
 MOVB #TKDDBR,(R4)
 BICB #200,(R4)
 CMPB (R4)+,#15
 BEQ INSTR2
 MOVB #TKDDBR,#TPDDBR
 2\$: TSTB #TPCSR
 BPL 2\$
 DEC R3
 BNE 1\$
 INSTR2: TYPE
 MQM
 BR INSTR1
 INSTR2: RTI
 .PARAMS

;CONVERT ASCII STRING TO OCTAL

; 3

011304 011605
 011306 012567 000146
 011312 012567 000144
 011316 012567 000142
 011322 112567 000140
 011326 112567 000135
 011332 010516
 011334 005005
 011336 012704 013224
 011342 122714 000015
 011346 001420
 011350 121427 000060
 011354 002415
 011356 121427 000067
 011362 003012
 011364 142714 000060
 011370 152405
 011372 122714 000015
 011376 001406
 011400 006305
 011402 006305
 011404 006305
 011406 000760
 011410 104404
 011412 000750

PARAMS: MOV (SP),R5
 MOV (R5)+,LOLIM
 MOV (R5)+,HILIM
 MOV (R5)+,DEVADR
 MOV (R5)+,LOBITS
 MOV (R5)+,ADRCNT
 MOV R5,(SP)
 PARAM1: CLR R5
 MOV #INBUF,R4
 CMPB #15,(R4)
 BEQ PARERR
 1\$: CMPB (R4),#60
 BLT PARERR
 CMPB (R4),#67
 BGT PARERR
 BICB #60,(R4)
 BISB (R4)+,R5
 CMPB #15,(R4)
 BEQ LIMITS
 ASL R5
 ASL R5
 ASL R5
 BR 1\$
 PARERR: INSTER
 BR PARAM1

;TEST TO SEE IF NUMBER IS WITHIN LIMITS

011414 020567 000042
 011420 101373
 011422 020567 000032
 011426 103770
 011430 136705 000032
 011434 001365

LIMITS: CMP R5,HILIM
 BHI PARERR
 CMP R5,LOLIM
 BLO PARERR
 BITB LOBITS,R5
 BNE PARERR

; 3

;STORE NUMBER AT SPECIFIED ADDRESS

011436 016704 000022
 011442 010524
 011444 062705 000002
 011450 105367 000013
 011454 001372
 011456 000002
 011460 000000
 011462 000000
 011464 000000
 011466 000000
 011467
 10 011470

1\$: MOV DEVADR,R4
 MOV R5,(R4)+
 ADD #2,R5
 DECB ADRCNT
 BNE 1\$
 RTI
 LOLIM: 0
 HILIM: 0
 DEVADR: 0
 LOBITS: 0
 ADRCNT=LOBITS+1
 .OCTASC

;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER

011470 017601 000000
 011474 062716 000002

OCTASN: MOV #2,(SP),R1
 ADD #2,(SP)

; 5


```

011500 012167 000130      MOV      (R1)+,WRDCNT
011504 112167 000126      1#:     MOVB   (R1)+,CHRCNT
011510 112167 000123      MOVB   (R1)+,SPACNT
011514 013167 000120      MOV     @R1+ ,BINWRD
011520 016704 000114      2#:     MOV     BINWRD,R4
011524 116705 000106      MOVB   CHRCNT,R5
011530 012700 013236      MOV     @TEMP,R0
011534 010403      3#:     MOV     R4,R3
011536 042703 177770      BIC     @177770,R3
011542 062703 000260      ADD     @260,R3
011546 110320      MOVB   R3,(R0)+
011550 006204      ASR     R4
011552 006204      ASR     R4
011554 006204      ASR     R4
011556 005305      DEC     R5
011560 001365      BNE     3#
011562 012703 013250      MOV     @MDATA,R3
011566 114023      4#:     MOVB   -(R0),(R3)+
011570 105367 000042      DECB   CHRCNT
011574 001374      BNE     4#
011576 105767 000035      TSTB   SPACNT
011602 001405      BEQ     6#
011604 112723 000240      5#:     MOVB   @240,(R3)+
011610 105367 000023      DECB   SPACNT
011614 001373      BNE     5#
011616 105013      6#:     CLRB   (R3)
011620 104401      TYPE
011622 013250      MDATA
011624 005367 000004      DEC     WRDCNT
011630 001325      BNE     1#
011632 000002      RTI
011634 000000      WRDCNT: 0
011636 000000      CHRCNT: 0
011637 011637      SPACNT=CHRCNT+1
011640 000000      BINWRD: 0
11 011642      .SAVREG
                                ;SAVE PC OF TEST THAT FAILED AND R0-R5
011642 016667 000004 000164 SV05P: MOV     4(SP),SAVPC
                                ;SAVE R0-R5
011650 010567 000154      SV05:  MOV     R5,SAVR5
011654 010467 000146      MOV     R4,SAVR4
011660 010367 000140      MOV     R3,SAVR3
011664 010267 000132      MOV     R2,SAVR2
011670 010167 000124      MOV     R1,SAVR1
011674 010067 000116      MOV     R0,SAVR0
011700 000002      RTI
12 011702      .RESREG
                                ;RESTORE R0-R5
011702 016700 000110      RS05:  MOV     SAVR0,R0
011706 015701 000106      MOV     SAVR1,R1
011712 016702 000104      MOV     SAVR2,R2
011716 016703 000102      MOV     SAVR3,R3

```

: 3

: 3

011722 016704 000100
 011726 016705 000076
 011732 000002
 13 011734

MOV SAVR4,R4
 MOV SAVR5,R5
 RTI
 .POINT †/DHSCR,DHNRC,DHLPR,DHBA,DHBC,DHBAR,DHBCR,DHSSR,DHSLR,DHRVEC,DHRLVL,DHTVEC,DHTLVL/
 ;INDIRECT POINTERS ; 3

011734 177560
 011736 177562
 011740 177564
 011742 177566

TKCSR: 177560
 TKDBR: 177562
 TPCSR: 177564
 TPDBR: 177566
 .IRP A <DHSCR,DHNRC,DHLPR,DHBA,DHBC,DHBAR,DHBCR,DHSSR,DHSLR,DHRVEC,DHRLVL,DHTVEC,DH

TLVL>

011744 000000
 011746 000000
 011750 000000
 011752 000000
 011754 000000
 011756 000000
 011760 000000
 011762 000000
 011764 000000
 011766 000000
 011770 000000
 011772 000000
 011774 000000
 14 011776

A: 0
 .ENDM
 DHSCR: 0
 DHNRC: 0
 DHLPR: 0
 DHBA: 0
 DHBC: 0
 DHBAR: 0
 DHBCR: 0
 DHSSR: 0
 DHSLR: 0
 DHRVEC: 0
 DHRLVL: 0
 DHTVEC: 0
 DHTLVL: 0
 .VARIA †/ENDFLG,TDATA/
 ;PROGRAM VARIABLES

011776 000000
 012000 000000
 012002 000000
 012004 000000
 012006 000000
 012010 000000
 012012 000000
 012014 000000
 012016 000000
 012020 000000
 012022 000000
 012024 000000
 012026 000000
 012030 000000
 012032 000000
 012034 000000
 012036 000000
 012040 000000
 012042 000000

ERRFLG: 0 ;ERROR FLAG
 PASCNT: 0 ;PASS COUNT
 ERRCNT: 0 ;ERROR COUNT
 RETRN: 0 ;SCOPE RETURN ADDRESS FOR TEST LOOPING
 ESCAPE: 0 ;ADDRESS FOR ERROR ESCAPE
 FREEZ1: 0 ;DATA LOOPING RETURN ADDRESS
 ICOUNT: 0 ;ITERATION COUNT FOR TEST IN PROGRESS
 LPCNT: 0 ;NUMBER OF ITERATIONS THIS TEST
 SAVR0: 0 ;R0 SAVE AREA
 SAVR1: 0 ;R1 SAVE AREA
 SAVR2: 0 ;R2 SAVE AREA
 SAVR3: 0 ;R3 SAVE ARE
 SAVR4: 0 ;R4 SAVE AREA
 SAVR5: 0 ;R5 SAVE AREA
 SAVSP: 0 ;STACK POINTER SAVE AREA
 SAVPC: 0 ;CALLING ROUTINE SAVE AREA
 INIFLG: 0 ;PROGRAM INITIALIZATION FLAG
 STFLG: 0 ;PROGRAM START FLAG
 LAST: 0 ;LAST ERROR PC
 .IRP A <ENDFLG,TDATA>

; 3

012044 000000
 012046 000000
 16 012050 000001
 18 012050 000
 19 000377
 20

.ENDM
 ENDFLG: 0
 TDATA: 0
 TDAT-1
 TBUF: .BYTE 0
 .REPT 377
 .BYTE TDAT

21
22
23
24

012051	001	.NLIST
	000002	TDAT=TDAT+1
012052	002	.LIST
	000003	.ENDR
012053	003	.BYTE TDAT
	000004	TDAT=TDAT+1
012054	004	.BYTE TDAT
	000005	TDAT=TDAT+1
012055	005	.BYTE TDAT
	000006	TDAT=TDAT+1
012056	006	.BYTE TDAT
	000007	TDAT=TDAT+1
012057	007	.BYTE TDAT
	000010	TDAT=TDAT+1
012060	010	.BYTE TDAT
	000011	TDAT=TDAT+1
012061	011	.BYTE TDAT
	000012	TDAT=TDAT+1
012062	012	.BYTE TDAT
	000013	TDAT=TDAT+1
012063	013	.BYTE TDAT
	000014	TDAT=TDAT+1
012064	014	.BYTE TDAT
	000015	TDAT=TDAT+1
012065	015	.BYTE TDAT
	000016	TDAT=TDAT+1
012066	016	.BYTE TDAT
	000017	TDAT=TDAT+1
012067	017	.BYTE TDAT
	000020	TDAT=TDAT+1
012070	020	.BYTE TDAT
	000021	TDAT=TDAT+1
012071	021	.BYTE TDAT
	000022	TDAT=TDAT+1
012072	022	.BYTE TDAT
	000023	TDAT=TDAT+1
012073	023	.BYTE TDAT
	000024	TDAT=TDAT+1
012074	024	.BYTE TDAT
	000025	TDAT=TDAT+1
012075	025	.BYTE TDAT
	000026	TDAT=TDAT+1
012076	026	.BYTE TDAT
	000027	TDAT=TDAT+1
012077	027	.BYTE TDAT
	000030	TDAT=TDAT+1
012100	030	.BYTE TDAT
	000031	TDAT=TDAT+1
012101	031	.BYTE TDAT
	000032	TDAT=TDAT+1
012102	032	.BYTE TDAT
	000033	TDAT=TDAT+1
012103	033	.BYTE TDAT

012104	000034	TDAT=TDAT+1
	034	.BYTE TDAT
012105	000035	TDAT=TDAT+1
	035	.BYTE TDAT
012106	000036	TDAT=TDAT+1
	036	.BYTE TDAT
012107	000037	TDAT=TDAT+1
	037	.BYTE TDAT
012110	000040	TDAT=TDAT+1
	040	.BYTE TDAT
012111	000041	TDAT=TDAT+1
	041	.BYTE TDAT
012112	000042	TDAT=TDAT+1
	042	.BYTE TDAT
012113	000043	TDAT=TDAT+1
	043	.BYTE TDAT
012114	000044	TDAT=TDAT+1
	044	.BYTE TDAT
012115	000045	TDAT=TDAT+1
	045	.BYTE TDAT
012116	000046	TDAT=TDAT+1
	046	.BYTE TDAT
012117	000047	TDAT=TDAT+1
	047	.BYTE TDAT
012120	000050	TDAT=TDAT+1
	050	.BYTE TDAT
012121	000051	TDAT=TDAT+1
	051	.BYTE TDAT
012122	000052	TDAT=TDAT+1
	052	.BYTE TDAT
012123	000053	TDAT=TDAT+1
	053	.BYTE TDAT
012124	000054	TDAT=TDAT+1
	054	.BYTE TDAT
012125	000055	TDAT=TDAT+1
	055	.BYTE TDAT
012126	000056	TDAT=TDAT+1
	056	.BYTE TDAT
012127	000057	TDAT=TDAT+1
	057	.BYTE TDAT
012130	000060	TDAT=TDAT+1
	060	.BYTE TDAT
012131	000061	TDAT=TDAT+1
	061	.BYTE TDAT
012132	000062	TDAT=TDAT+1
	062	.BYTE TDAT
012133	000063	TDAT=TDAT+1
	063	.BYTE TDAT
012134	000064	TDAT=TDAT+1
	064	.BYTE TDAT
012135	000065	TDAT=TDAT+1
	065	.BYTE TDAT
012136	000066	TDAT=TDAT+1
	066	.BYTE TDAT
012137	000067	TDAT=TDAT+1
	067	.BYTE TDAT
	000070	TDAT=TDAT+1

012140	070	.BYTE TDAT
	000071	TDAT=TDAT+1
012141	071	.BYTE TDAT
	000072	TDAT=TDAT+1
012142	072	.BYTE TDAT
	000073	TDAT=TDAT+1
012143	073	.BYTE TDAT
	000074	TDAT=TDAT+1
012144	074	.BYTE TDAT
	000075	TDAT=TDAT+1
012145	075	.BYTE TDAT
	000076	TDAT=TDAT+1
012146	076	.BYTE TDAT
	000077	TDAT=TDAT+1
012147	077	.BYTE TDAT
	000100	TDAT=TDAT+1
012150	100	.BYTE TDAT
	000101	TDAT=TDAT+1
012151	101	.BYTE TDAT
	000102	TDAT=TDAT+1
012152	102	.BYTE TDAT
	000103	TDAT=TDAT+1
012153	103	.BYTE TDAT
	000104	TDAT=TDAT+1
012154	104	.BYTE TDAT
	000105	TDAT=TDAT+1
012155	105	.BYTE TDAT
	000106	TDAT=TDAT+1
012156	106	.BYTE TDAT
	000107	TDAT=TDAT+1
012157	107	.BYTE TDAT
	000110	TDAT=TDAT+1
012160	110	.BYTE TDAT
	000111	TDAT=TDAT+1
012161	111	.BYTE TDAT
	000112	TDAT=TDAT+1
012162	112	.BYTE TDAT
	000113	TDAT=TDAT+1
012163	113	.BYTE TDAT
	000114	TDAT=TDAT+1
012164	114	.BYTE TDAT
	000115	TDAT=TDAT+1
012165	115	.BYTE TDAT
	000116	TDAT=TDAT+1
012166	116	.BYTE TDAT
	000117	TDAT=TDAT+1
012167	117	.BYTE TDAT
	000120	TDAT=TDAT+1
012170	120	.BYTE TDAT
	000121	TDAT=TDAT+1
012171	121	.BYTE TDAT
	000122	TDAT=TDAT+1
012172	122	.BYTE TDAT
	000123	TDAT=TDAT+1
012173	123	.BYTE TDAT
	000124	TDAT=TDAT+1
012174	124	.BYTE TDAT

012175	000125	TDAT=TDAT+1
	125	.BYTE TDAT
	000126	TDAT=TDAT+1
012176	126	.BYTE TDAT
	000127	TDAT=TDAT+1
012177	127	.BYTE TDAT
	000130	TDAT=TDAT+1
012200	130	.BYTE TDAT
	000131	TDAT=TDAT+1
012201	131	.BYTE TDAT
	000132	TDAT=TDAT+1
012202	132	.BYTE TDAT
	000133	TDAT=TDAT+1
012203	133	.BYTE TDAT
	000134	TDAT=TDAT+1
012204	134	.BYTE TDAT
	000135	TDAT=TDAT+1
012205	135	.BYTE TDAT
	000136	TDAT=TDAT+1
012206	136	.BYTE TDAT
	000137	TDAT=TDAT+1
012207	137	.BYTE TDAT
	000140	TDAT=TDAT+1
012210	140	.BYTE TDAT
	000141	TDAT=TDAT+1
012211	141	.BYTE TDAT
	000142	TDAT=TDAT+1
012212	142	.BYTE TDAT
	000143	TDAT=TDAT+1
012213	143	.BYTE TDAT
	000144	TDAT=TDAT+1
012214	144	.BYTE TDAT
	000145	TDAT=TDAT+1
012215	145	.BYTE TDAT
	000146	TDAT=TDAT+1
012216	146	.BYTE TDAT
	000147	TDAT=TDAT+1
012217	147	.BYTE TDAT
	000150	TDAT=TDAT+1
012220	150	.BYTE TDAT
	000151	TDAT=TDAT+1
012221	151	.BYTE TDAT
	000152	TDAT=TDAT+1
012222	152	.BYTE TDAT
	000153	TDAT=TDAT+1
012223	153	.BYTE TDAT
	000154	TDAT=TDAT+1
012224	154	.BYTE TDAT
	000155	TDAT=TDAT+1
012225	155	.BYTE TDAT
	000156	TDAT=TDAT+1
012226	156	.BYTE TDAT
	000157	TDAT=TDAT+1
012227	157	.BYTE TDAT
	000160	TDAT=TDAT+1
012230	160	.BYTE TDAT
	000161	TDAT=TDAT+1

012231	161	.BYTE TDAT
	000162	TDAT=TDAT+1
012232	162	.BYTE TDAT
	000163	TDAT=TDAT+1
012233	163	.BYTE TDAT
	000164	TDAT=TDAT+1
012234	164	.BYTE TDAT
	000165	TDAT=TDAT+1
012235	165	.BYTE TDAT
	000166	TDAT=TDAT+1
012236	166	.BYTE TDAT
	000167	TDAT=TDAT+1
012237	167	.BYTE TDAT
	000170	TDAT=TDAT+1
012240	170	.BYTE TDAT
	000171	TDAT=TDAT+1
012241	171	.BYTE TDAT
	000172	TDAT=TDAT+1
012242	172	.BYTE TDAT
	000173	TDAT=TDAT+1
012243	173	.BYTE TDAT
	000174	TDAT=TDAT+1
012244	174	.BYTE TDAT
	000175	TDAT=TDAT+1
012245	175	.BYTE TDAT
	000176	TDAT=TDAT+1
012246	176	.BYTE TDAT
	000177	TDAT=TDAT+1
012247	177	.BYTE TDAT
	000200	TDAT=TDAT+1
012250	200	.BYTE TDAT
	000201	TDAT=TDAT+1
012251	201	.BYTE TDAT
	000202	TDAT=TDAT+1
012252	202	.BYTE TDAT
	000203	TDAT=TDAT+1
012253	203	.BYTE TDAT
	000204	TDAT=TDAT+1
012254	204	.BYTE TDAT
	000205	TDAT=TDAT+1
012255	205	.BYTE TDAT
	000206	TDAT=TDAT+1
012256	206	.BYTE TDAT
	000207	TDAT=TDAT+1
012257	207	.BYTE TDAT
	000210	TDAT=TDAT+1
012260	210	.BYTE TDAT
	000211	TDAT=TDAT+1
012261	211	.BYTE TDAT
	000212	TDAT=TDAT+1
012262	212	.BYTE TDAT
	000213	TDAT=TDAT+1
012263	213	.BYTE TDAT
	000214	TDAT=TDAT+1
012264	214	.BYTE TDAT
	000215	TDAT=TDAT+1
012265	215	.BYTE TDAT

	000216	TDAT=TDAT+1
012266	216	.BYTE TDAT
	000217	TDAT=TDAT+1
012267	217	.BYTE TDAT
	000220	TDAT=TDAT+1
012270	220	.BYTE TDAT
	000221	TDAT=TDAT+1
012271	221	.BYTE TDAT
	000222	TDAT=TDAT+1
012272	222	.BYTE TDAT
	000223	TDAT=TDAT+1
012273	223	.BYTE TDAT
	000224	TDAT=TDAT+1
012274	224	.BYTE TDAT
	000225	TDAT=TDAT+1
012275	225	.BYTE TDAT
	000226	TDAT=TDAT+1
012276	226	.BYTE TDAT
	000227	TDAT=TDAT+1
012277	227	.BYTE TDAT
	000230	TDAT=TDAT+1
012300	230	.BYTE TDAT
	000231	TDAT=TDAT+1
012301	231	.BYTE TDAT
	000232	TDAT=TDAT+1
012302	232	.BYTE TDAT
	000233	TDAT=TDAT+1
012303	233	.BYTE TDAT
	000234	TDAT=TDAT+1
012304	234	.BYTE TDAT
	000235	TDAT=TDAT+1
012305	235	.BYTE TDAT
	000236	TDAT=TDAT+1
012306	236	.BYTE TDAT
	000237	TDAT=TDAT+1
012307	237	.BYTE TDAT
	000240	TDAT=TDAT+1
012310	240	.BYTE TDAT
	000241	TDAT=TDAT+1
012311	241	.BYTE TDAT
	000242	TDAT=TDAT+1
012312	242	.BYTE TDAT
	000243	TDAT=TDAT+1
012313	243	.BYTE TDAT
	000244	TDAT=TDAT+1
012314	244	.BYTE TDAT
	000245	TDAT=TDAT+1
012315	245	.BYTE TDAT
	000246	TDAT=TDAT+1
012316	246	.BYTE TDAT
	000247	TDAT=TDAT+1
012317	247	.BYTE TDAT
	000250	TDAT=TDAT+1
012320	250	.BYTE TDAT
	000251	TDAT=TDAT+1
012321	251	.BYTE TDAT
	000252	TDAT=TDAT+1

012322	252	.BYTE TDAT
	000253	TDAT=TDAT+1
012323	253	.BYTE TDAT
	000254	TDAT=TDAT+1
012324	254	.BYTE TDAT
	000255	TDAT=TDAT+1
012325	255	.BYTE TDAT
	000256	TDAT=TDAT+1
012326	256	.BYTE TDAT
	000257	TDAT=TDAT+1
012327	257	.BYTE TDAT
	000260	TDAT=TDAT+1
012330	260	.BYTE TDAT
	000261	TDAT=TDAT+1
012331	261	.BYTE TDAT
	000262	TDAT=TDAT+1
012332	262	.BYTE TDAT
	000263	TDAT=TDAT+1
012333	263	.BYTE TDAT
	000264	TDAT=TDAT+1
012334	264	.BYTE TDAT
	000265	TDAT=TDAT+1
012335	265	.BYTE TDAT
	000266	TDAT=TDAT+1
012336	266	.BYTE TDAT
	000267	TDAT=TDAT+1
012337	267	.BYTE TDAT
	000270	TDAT=TDAT+1
012340	270	.BYTE TDAT
	000271	TDAT=TDAT+1
012341	271	.BYTE TDAT
	000272	TDAT=TDAT+1
012342	272	.BYTE TDAT
	000273	TDAT=TDAT+1
012343	273	.BYTE TDAT
	000274	TDAT=TDAT+1
012344	274	.BYTE TDAT
	000275	TDAT=TDAT+1
012345	275	.BYTE TDAT
	000276	TDAT=TDAT+1
012346	276	.BYTE TDAT
	000277	TDAT=TDAT+1
012347	277	.BYTE TDAT
	000300	TDAT=TDAT+1
012350	300	.BYTE TDAT
	000301	TDAT=TDAT+1
012351	301	.BYTE TDAT
	000302	TDAT=TDAT+1
012352	302	.BYTE TDAT
	000303	TDAT=TDAT+1
012353	303	.BYTE TDAT
	000304	TDAT=TDAT+1
012354	304	.BYTE TDAT
	000305	TDAT=TDAT+1
012355	305	.BYTE TDAT
	000306	TDAT=TDAT+1
012356	306	.BYTE TDAT

012357	000307	TDAT=TDAT+1
	307	.BYTE TDAT
	000310	TDAT=TDAT+1
012360	310	.BYTE TDAT
	000311	TDAT=TDAT+1
012361	311	.BYTE TDAT
	000312	TDAT=TDAT+1
012362	312	.BYTE TDAT
	000313	TDAT=TDAT+1
012363	313	.BYTE TDAT
	000314	TDAT=TDAT+1
012364	314	.BYTE TDAT
	000315	TDAT=TDAT+1
012365	315	.BYTE TDAT
	000316	TDAT=TDAT+1
012366	316	.BYTE TDAT
	000317	TDAT=TDAT+1
012367	317	.BYTE TDAT
	000320	TDAT=TDAT+1
012370	320	.BYTE TDAT
	000321	TDAT=TDAT+1
012371	321	.BYTE TDAT
	000322	TDAT=TDAT+1
012372	322	.BYTE TDAT
	000323	TDAT=TDAT+1
012373	323	.BYTE TDAT
	000324	TDAT=TDAT+1
012374	324	.BYTE TDAT
	000325	TDAT=TDAT+1
012375	325	.BYTE TDAT
	000326	TDAT=TDAT+1
012376	326	.BYTE TDAT
	000327	TDAT=TDAT+1
012377	327	.BYTE TDAT
	000330	TDAT=TDAT+1
012400	330	.BYTE TDAT
	000331	TDAT=TDAT+1
012401	331	.BYTE TDAT
	000332	TDAT=TDAT+1
012402	332	.BYTE TDAT
	000333	TDAT=TDAT+1
012403	333	.BYTE TDAT
	000334	TDAT=TDAT+1
012404	334	.BYTE TDAT
	000335	TDAT=TDAT+1
012405	335	.BYTE TDAT
	000336	TDAT=TDAT+1
012406	336	.BYTE TDAT
	000337	TDAT=TDAT+1
012407	337	.BYTE TDAT
	000340	TDAT=TDAT+1
012410	340	.BYTE TDAT
	000341	TDAT=TDAT+1
012411	341	.BYTE TDAT
	000342	TDAT=TDAT+1
012412	342	.BYTE TDAT
	000343	TDAT=TDAT+1

012413	343	.BYTE TDAT
	000344	TDAT=TDAT+1
012414	344	.BYTE TDAT
	000345	TDAT=TDAT+1
012415	345	.BYTE TDAT
	000346	TDAT=TDAT+1
012416	346	.BYTE TDAT
	000347	TDAT=TDAT+1
012417	347	.BYTE TDAT
	000350	TDAT=TDAT+1
012420	350	.BYTE TDAT
	000351	TDAT=TDAT+1
012421	351	.BYTE TDAT
	000352	TDAT=TDAT+1
012422	352	.BYTE TDAT
	000353	TDAT=TDAT+1
012423	353	.BYTE TDAT
	000354	TDAT=TDAT+1
012424	354	.BYTE TDAT
	000355	TDAT=TDAT+1
012425	355	.BYTE TDAT
	000356	TDAT=TDAT+1
012426	356	.BYTE TDAT
	000357	TDAT=TDAT+1
012427	357	.BYTE TDAT
	000360	TDAT=TDAT+1
012430	360	.BYTE TDAT
	000361	TDAT=TDAT+1
012431	361	.BYTE TDAT
	000362	TDAT=TDAT+1
012432	362	.BYTE TDAT
	000363	TDAT=TDAT+1
012433	363	.BYTE TDAT
	000364	TDAT=TDAT+1
012434	364	.BYTE TDAT
	000365	TDAT=TDAT+1
012435	365	.BYTE TDAT
	000366	TDAT=TDAT+1
012436	366	.BYTE TDAT
	000367	TDAT=TDAT+1
012437	367	.BYTE TDAT
	000370	TDAT=TDAT+1
012440	370	.BYTE TDAT
	000371	TDAT=TDAT+1
012441	371	.BYTE TDAT
	000372	TDAT=TDAT+1
012442	372	.BYTE TDAT
	000373	TDAT=TDAT+1
012443	373	.BYTE TDAT
	000374	TDAT=TDAT+1
012444	374	.BYTE TDAT
	000375	TDAT=TDAT+1
012445	375	.BYTE TDAT
	000376	TDAT=TDAT+1
012446	376	.BYTE TDAT
	000377	TDAT=TDAT+1
012447	377	.BYTE TDAT

25
26 012450
27
28 012512

000400
000000
012512

TDAT=TDAT+1
.EVEN
RBUF: 0
. = .+40
.PFAIL

;ENTER HERE ON POWER FAILURE

012512 010046
012514 010146
012516 010246
012520 010346
012522 010446
012524 010546
012526 016746 165272
012532 010667 177274
012536 012767 012550 165260
012544 000000
012546 000777

PFAIL: MOV R0,-(SP)
MOV R1,-(SP)
MOV R2,-(SP)
MOV R3,-(SP)
MOV R4,-(SP)
MOV R5,-(SP)
MOV 24,-(SP)
MOV SP,SAVSP
MOV #RESTART,24
HALT
BR

;SAVE R0-R5 ON PROCESSOR STACK

;SAVE STACK POINTER
;SET UP FOR POWER UP TRAP : 3
;HALT ON POWER DOWN NORMAL

;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED

012550 016706 177256
012554 012605
012556 012604
012560 012603
012562 012602
012564 012601
012566 012600
012570 012767 012512 165226
012576 012767 000340 165172
012604 012706 013752
012610 005067 000422
012614 005267 000416
012620 001375
012622 104401
012624 013037
012626 104402
012630 012652
012632 104401
012634 013042
012636 005067 177134
012642 005067 177174
012646 000177 177132
012652 000001
012654 000006 000002
012660 012004

RESTAR: MOV SAVSP,SP
MOV (SP)+,R5
MOV (SP)+,R4
MOV (SP)+,R3
MOV (SP)+,R2
MOV (SP)+,R1
MOV (SP)+,R0
MOV #PFAIL,24
MOV #340,PS
MOV #STACK,SP
CLR TEMP
INC TEMP
BNE .-4
TYPE
MCRLF
OCTASC
PFTAB
TYPE
MPFAIL
CLR ERRFLG
CLR LAST
JMP #RETRN

;RESTORE STACK POINTER
;RESTORE R0-R5

;SET UP FOR POWER FAILURE

: 5
: 5

PFTAB: 1
6,2
RETRN

29 012662 .MSG †/DH11 SINGLE LINE PARITY CHECK & MULTI-LINE DATA TEST/,†/CZDHG-CO/
012662 015 012 012 MTITLE: .ASCIZ <15><12><12>/DH11 SINGLE LINE PARITY CHECK & MULTI-LINE DATA TEST /<15><12>
012665 104 110 061
012670 061 040 123
012673 111 116 107
012676 114 105 040
012701 114 111 116
012704 105 040 120
012707 101 122 111
012712 124 131 040

012715	103	110	105	
012720	103	113	040	
012723	046	040	115	
012726	125	114	124	
012731	111	055	114	
012734	111	116	105	
012737	040	104	101	
012742	124	101	040	
012745	124	105	123	
012750	124	040	015	
012753	012	000		
012755	015	012	126	MVECTOR: .ASCIZ <15><12>/VECTOR ADDRESS- /
012760	105	103	124	
012763	117	122	040	
012766	101	104	104	
012771	122	105	123	
012774	123	055	000	
012777	015	012	103	MREGAD: .ASCIZ <15><12>/CONTROL REGISTER ADDRESS- /
013002	117	116	124	
013005	122	117	114	
013010	040	122	105	
013013	107	111	123	
013016	124	105	122	
013021	040	101	104	
013024	104	122	105	
013027	123	123	055	
013032	000			
013033	040	040	077	MQM: .ASCIZ / ? /
013036	000			
013037	015	012	000	MCRLF: .ASCIZ <15><12>
013042	040	040	120	MPFAIL: .ASCIZ / POWER FAILURE, PROGRAM RESTART AT TEST IN PROGRESS /
013045	117	127	105	
013050	122	040	106	
013053	101	111	114	
013056	125	122	105	
013061	054	040	120	
013064	122	117	107	
013067	122	101	115	
013072	040	122	105	
013075	123	124	101	
013100	122	124	040	
013103	101	124	040	
013106	124	105	123	
013111	124	040	111	
013114	116	040	120	
013117	122	117	107	
013122	122	105	123	
013125	123	000		
013127	015	012	103	MEPASS: .ASCIZ <15><12>/CZDMG-CO /
013132	132	104	110	
013135	107	055	103	
013140	060	000		
013142	015	012	120	PASTXT: .ASCIZ <15><12>/PASS COUNT = /
013145	101	123	123	
013150	040	103	117	
013153	125	116	124	

013156	040	075	040		
013161	000				
013162	015	012	122	MR:	.ASCIZ <15><12>/R/
013165	000				
013166	015	012	124	MTSTPC:	.ASCIZ <15><12>/TEST PC-/
013171	105	123	124		
013174	040	120	103		
013177	055	000			

30
31 013202

.EVEN
.EVEN
.TRPTAB

;TABLE OF POINTERS FOR TRAP DECODING

013202	010566	TRPTAB:	SCOPER
013204	011150		TYPER
013206	011470		OCTASN
013210	011202		INSTRG
013212	011274		INSTRE
013214	011304		PARAMS
013216	011642		SV05P
013220	011702		RS05
013222	010666		SCOP1R
32 013224		.BUFFER	

;BUFFERS FOR INPUT-OUTPUT

013224	000000	INBUF:	0
	013236	.+.10	
013236	000000	TEMP:	0
	013250	.+.10	
013250	000000	MDATA:	0
	013262	.+.10	
33 013262		.ERRTAB	

;TABLE OF POINTERS TO ERROR MESSAGES AND DATA

013262		ERRTAB:	EM0		
34 013262	013312		0		
35 013264	000000		EM1		
36 013266	013337		0		
37 013270	000000		EM2		
38 013272	013373		0		
39 013274	000000		EM3		
40 013276	013414		0		
41 013300	013534		EM4		
42 013302	013455		DT1		
43 013304	000000		EM4		
44 013306	013503		0		
45 013310	000000		EM5		
			0		
46 013312	125	116	105	EMO:	.ASCIZ /UNEXPECTED INTERRUPT/
	013315	130	120		
	013320	103	124		
	013323	104	040		
	013326	116	124		
	013331	122	122		

	013334	120	124	000				
47	013337	103	110	101	EM1:	.ASCIZ	/CHARACTER AVAILABLE NOT SET/	
	013342	122	101	103				
	013345	124	105	122				
	013350	040	101	126				
	013353	101	111	114				
	013356	101	102	114				
	013361	105	040	116				
	013364	117	124	040				
	013367	123	105	124				
	013372	000						
48	013373	123	111	114	EM2:	.ASCIZ	/SILO OVERRUN SET/	
	013376	117	040	117				
	013401	126	105	122				
	013404	122	125	116				
	013407	040	123	105				
	013412	124	000					
49	013414	104	101	124	EM3:	.ASCII	/DATA ERROR/	
	013417	101	040	105				
	013422	122	122	117				
	013425	122						
50	013426	015	012	105		.ASCIZ	<15><12>/EXP	REC LINE/
	013431	130	120	040				
	013434	040	040	040				
	013437	040	122	105				
	013442	103	040	040				
	013445	040	040	040				
	013450	114	111	116				
	013453	105	000					
51	013455	124	122	101	EM4:	.ASCIZ	/TRANSMIT DONE NOT SET/	
	013460	116	123	115				
	013463	111	124	040				
	013466	104	117	116				
	013471	105	040	116				
	013474	117	124	040				
	013477	123	105	124				
	013502	000						
52	013503	103	110	101	EM5:	.ASCIZ	/CHARACTER AVAILABLE SET/	
	013506	122	101	103				
	013511	124	105	122				
	013514	040	101	126				
	013517	101	111	114				
	013522	101	102	114				
	013525	105	040	123				
	013530	105	124	000				
53						.EVEN		
54								
55								
56								
57	013534	000003			DT1:	3		
58	013536	006	002		.BYTE	6.2		
59	013540	012030				SAVR5		
60	013542	006	002		.BYTE	6.2		
61	013544	012020				SAVR1		
62	013546	002	000		.BYTE	2.0		
63	013550	012024				SAVR3		
64	013552					.ENDCOD		

65

013552

000000
000001

ENDCOD: 0
.END

ADRCNT = 011467	EM2 013373	MQM 013033	SAVR5 012030	TRPTAB 013202
BARBIT 010426	EM3 013414	MR 013162	SAVSP 012032	TYPDAT 011026
BEGIN 001306	EM4 013455	MREGAD 012777	SAV05P = 104406	TYPE = 104401
BINWRD 011640	EM5 013503	MSG 011216	SCOPE = 104400	TYPER 011150
BITX = 000000	ENDCOD 013552	MTITLE 012662	SCOPE1 = 104410	TYPMSG 011004
BIT00 = 000001	ENDFLG 012044	MTSTPC 013166	SCOPE1R 010666	T1 001400
BIT01 = 000002	EOP 010466	MVECTO 012755	SPACNT = 011637	T10 002656
BIT02 = 000004	ERRCNT 012002	N = 000001	STACK = 013752	T11 003020
BIT03 = 000010	ERRFLG 011776	OCTASC = 104402	START 001004	T12 003162
BIT04 = 000020	ERRMSG 011024	OCTASN 011470	STFLG 012040	T13 003324
BIT05 = 000040	ERRORS 010704	PARAM = 104405	SV05 011650	T14 003466
BIT06 = 000100	ERRTAB 013262	PARAMS 011304	SV05P 011642	T15 003630
BIT07 = 000200	ERTAB0 011110	PARAM1 011334	SWR 001000	T16 003772
BIT08 = 000400	ESCAPE 012006	PARERR 011410	SW00 = 000001	T17 004134
BIT09 = 001000	EXITER 011066	PASARG 010560	SW01 = 000002	T2 001542
BIT10 = 002000	FREEZ1 012010	PASCNT 012000	SW02 = 000004	T20 004276
BIT11 = 004000	HALTS 011046	PASTXT 013142	SW03 = 000010	T21 004440
BIT12 = 010000	HILIM 011462	PFAIL 012512	SW04 = 000020	T22 004744
BIT13 = 020000	ICOUNT 012012	PFTAB 012652	SW05 = 000040	T23 005250
BIT14 = 040000	INBUF 013224	POP0 = 012600	SW06 = 000100	T24 005554
BIT15 = 100000	INIFLG 012036	POP1SP = 005726	SW08 = 000400	T25 006060
CHRCNT 011636	INSTER = 104404	POP2SP = 022626	SW09 = 001000	T26 006364
DATABP 011042	INSTR = 104403	PS = 177776	SW10 = 002000	T27 006670
DEVADR 011464	INSTRE 011274	PUSHRO = 010046	SW11 = 004000	T3 001704
DHBA 011752	INSTRG 011202	PUSH1S = 005746	SW12 = 010000	T30 007174
DHBAR 011756	INSTR1 011214	PUSH2S = 024646	SW13 = 020000	T31 007500
DHBC 011754	INSTR2 011302	RBUF 012450	SW14 = 040000	T32 007776
DHBCR 011760	LAST 012042	RESREG 011044	SW15 = 100000	T4 002046
DHLPR 011750	LIGHTS 001002	RESTAR 012550	TBUF 012050	T5 002210
DHNRC 011746	LIMITS 011414	RESTR1 010554	TDAT = 000400	T6 002352
DHRLVL 011770	LINE = 000020	RES05 = 104407	TDATA 012046	T7 002514
DHRVEC 011766	LOBITS 011466	RETRN 012004	TEMP 013236	VEC1 001164
DHSCR 011744	LOGICA 010544	RS05 011702	TKCSR 011734	VEC2 001174
DHSLR 011764	LOLIM 011460	SAVPC 012034	TKDBR 011736	WRDCNT 011634
DHSSR 011762	LPCNT 012014	SAVR0 012016	TPCSR 011740	X = 000000
DHTLVL 011774	MCRLF 013037	SAVR1 012020	TPDBR 011742	XBIT = 000001
DHTVEC 011772	MDATA 013250	SAVR2 012022	TRPOK 011130	XLIN = 000000
DT1 013534	MEPASS 013127	SAVR3 012024	TRPSRV 011116	XN = 000033
EMO 013312	MPFAIL 013042	SAVR4 012026		Y = 000011
EM1 013337				

. ABS. 013554 000
000000 001
ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 18944 WORDS (74 PAGES)
DYNAMIC MEMORY AVAILABLE FOR 71 PAGES
CZDHGC.BIN,CZDHGC.SEQ=CZDHGC.DOC,DHMACA.MAC,CZDHGC.P11