

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

.REM !

IDENTIFICATION  
-----

PRODUCT CODE: AC-8476D-MC  
PRODUCT NAME: CZDH100 DH11 BREAK AND HALF DUPLEX TEST  
DATE: JUNE 1985  
MAINTAINER: MK NAC SOFTWARE ENGINEERING

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

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

COPYRIGHT (C) 1985 BY DIGITAL EQUIPMENT CORPORATION

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11

1. ABSTRACT

THE DH11 BREAK AND HALF DUPLEX TEST CHECKS THE BREAK CONTROL LOGIC OF THE DH11 AND VERIFIES THAT THE UARTS RECEIVE ONLY ONE BREAK CHARACTER ON A GIVEN LINE NO MATTER HOW LONG BREAK IS ASSERTED. THE TEST ALSO VERIFIES THAT NO CHARACTERS ARE RECEIVED ON A A LINE IF THE HALF DUPLEX FUNCTION FOR THAT LINE IS SELECTED.

```
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
```

2. REQUIREMENTS

2.1 EQUIPMENT

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

2.2 STORAGE

THE PROGRAM LOADS INTO 4KW OF MEMORY

3. LOADING PROCEDURE

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

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

4.1.1 AFTER PROGRAM LOAD (INITIAL PROGRAM START)

ALL CONSOLE SWITCHES DOWN

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 I 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 BREAK AND HALF-DUPLEX TEST"  
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  
52  
53  
54

## 4.3 (CONT'D)

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 MESSAGE 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 BREAK AND HALF-DUPLEX TEST"  
AND WILL THEN CONTINUE AS DESCRIBED IN 4.3.1.9

4.3.3 PROGRAM RESTART WITH SW00=1

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 BREAK AND HALF-DUPLEX TEST"  
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

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

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

5. 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.

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 IMMEDIATLY 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

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

#### 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  
A) IF SW10=1, THE ROUTINE WILL TRANSFER TO THE NEXT TEST IN SEQUENCE, AFTER CLEARING ERROR FLAGS.  
B) IF SW11=1, THE ROUTINE WILL TRANSFER TO THE NEXT TEST SEQUENCE, AFTER CLEARING ERROR FLAGS.  
C) 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.

#### 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.

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

### 5.2.7 ERRORS (ERROR HANDLER)

THIS ROUTINE IS ENTERED UPON ERROR DETECTION ONLY.  
WITH ALL CONSOLE SWITCHES DOWN, THE ROUTINE PROCEEDS AS FOLLOWS:  
A) 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.  
B) IF THE TEST THAT FAILED DID NOT FAIL PREVIOUSLY  
DURING THIS PASS, A COMPLETE ERROR REPORT IS MADE  
IF THE TEST THAT FAILED FAILED MOR THAT ONCE DURING  
THE CURRENT PASS, ONLY THE DATA RELATING TO THE FAILUER  
IS TYPED. IF SW13=1, NO ERROR TYPEOUT IS MADE.  
C) 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 R0. IF SW15=0, THE PROGRAM WILL  
NOT HALT, BUT WILL CHECK FOR ESCAPE TO NEXT TEST.  
D) 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 TH 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  
51  
52  
53  
54  
55

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 "CZDHI" 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

5.3.3.3 AFTER "CZDHI" 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.



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

5. (CONT'D)

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.

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

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

6.3 SCOPE LOOPING

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. (CONT'D)

6.3.3 PROGRAM START TO SCOPE LOOP ON SELECTED TEST

PERFORM SECTION 4.3.4 WITH SW14=1

7. RESTRICTIONS

7.1 STARTING

THE DH11 TEST CARD MUST BE INSTALLED

7.2 RUNNING

NONE

8. MISCELLANEOUS

8.1 EXECUTION TIME

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

PROCESSOR	TIME
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  
29  
30  
32 000000

## 9. PROGRAM DESCRIPTION

THE FIRST GROUP OF TESTS VERIFIES THAT ONLY ONE BREAK CHARACTER IS RECEIVED ON A GIVEN LINE, EVEN THOUGH BREAK IS ASSRETED FOR THAT LINE FOR 400 (OCTAL) CHARACTER TIMES. THERE IS AN INDIVIDUAL TEST LOOP FOR EACH LINE. THE TEST BEGINS BY FLUSING EACH UART BY TRANSMITTING TO NULL (0) CHARACTERS. THE BREAK BIT IS THEN SET FOR THE LINE TO BE TESTED, TRANSMISSION OF THE BINARY COUNT PATTERN IS STARTED. THE SILO IS THEN CHECKED TO VERIFY THAT ONLY ONE CHARACTER WAS RECEIVED AND THAT IT WAS A BREAK CHARACTER.

THE SECOND GROUP OF TESTS VERIFIES THAT NO CHARACTERS ARE RECEIVED ON A SELECTED LINE IF THE HALF DUPLEX BIT IS SET FOR THE LINE TO BE TESTED. THERE IS AN INDIVIDUAL TEST LOOP FOR EACH LINE TO BE TESTED. A BINARY COUNT PATTERN IS THEN TRANSMITTED ON THE LINE TO BE TESTED. WHEN ALL CHARACTERS HAVE BEEN TRANSMITTED, THE CHARACTER AVAILABLE FLAG IS TESTED TO DETERMINE IF ANY CHARACTERS HAVE BEEN RECEIVED.

## 10 LISTING

```

;
.LIST ME
.NLIST MC,MD,CND
.HEADER †/1972, 1976, 1985/,†/DH11 BREAK AND HALF DUPLEX TEST/,†/CZDHI-DO/

```

```

;STARTING PROCEDURE
;LOAD PROGRAM
;LOAD ADDRESS 000200
;PRESS START
;PROGRAM WILL TYPE DH11 BREAK AND HALF DUPLEX 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>
;PROGRAM WILL TYPE "R" TO INDICATE THAT TESTING HAS STARTED
;AT THE END OF A PASS, PROGRAM WILL TYPE " CZDHI-DO "
;AND THEN RESUM TESTING

```

; 3

000000

```

.TITLE CZDHI-DO
.ENABLE ABS
.NLIST MC,MD,CND
.LIST ME
.SYMBOLS

```

33 000000

## ;SWITCH REGISTER OPTIONS

100000  
040000  
020000

```

SW15=100000      ;=1,HALT ON ERROR
SW14=40000       ;=1,LOOP ON CURRENT TEST
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
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
```

; 3

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
013724 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 ; 3
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 000000      ;TRAPCATCAER FOR ILLEGAL INTERRUPTS
000200      .-0
          .REPT 200
          .+2      ;UNEXPECTED TRAP TO THIS LOCATION
          HALT     ;EXAMINE STACK TO FIND CAUSE
          .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          000200          .-200      ;STANDARD INTERRUPT VECTORS
000200    000167  000600      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          TRPDEF SCOPE,+/SCOPE LOOP AND ITERATION HANDLER/
          104400          SCOPE=TRAP+Y          ;SCOPE LOOP AND ITERATION HANDLER
          000001          Y=Y+1

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

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

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

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

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

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

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

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

2          000046          .-46
3          010556          LOGICAL
4          000052          .-52
5          040000          40000
6          .MACRO CODEM1
7          MOV      DHSSR,DHSLR          ;SET UP ADDRESS OF SILO
8          INC      DHSLR              ;STATUS REGISTER HIGH BYTE
9          .ENDM CODEM1
10 000054          .START DHRVEC,3,4,DHSCR,0,177776,7,10,...1

```

```

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 013724      MOV #STACK,SP ;SET UP PROCESSOR STACK
001016 012702 000024      MOV #24,R2 ; POINT TO VECTOR AREA      ; 7
001022 012722 012614      MOV #PFAIL,(R2)+ ;SET UP POWER FAIL TRAP    ; 7
001026 012722 000340      MOV #340,(R2)+ ;SERVICE AT LEVEL 7       ; 7
001032 012722 010716      MOV #ERRORS,(R2)+ ;ERROR HANDLER             ; 7
001036 012722 000340      MOV #340,(R2)+ ;SERVICE AT LEVEL 7       ; 7
001042 012722 011130      MOV #TRPSRV,(R2)+ ;GENERAL HANDLER DISPATCH SERVICE ; 7
001046 012712 000340      MOV #340,(R2) ;SERVICE AT LEVEL 7       ; 8
0C1052 005067 010774      CLR STFLG ;CLEAR TEST START FLAG
001056 005067 010730      CLR PASCNT ;CLEAR PASS COUNT
001062 005067 010726      CLR ERRCNT ;CLEAR ERROR COUNT
001066 005067 010716      CLR ERRFLG ;CLEAR ERROR FLAG
001072 005067 010712      CLR ERRFLG ;CLEAR LAST ERROR PC
001076 016746 176702      MOV 4,-(SP) ; PUSH TRAP VECTOR          ; 4
001102 016746 176700      MOV 6,-(SP) ;                               ; 4
0C1106 012767 001122 176670 MOV #1$,4 ; SET UP TRAP VECTOR        ; 4
001114 005777 177660      TST #SWR ; TEST SWITCH REGISTER ADDRESS ; 4
001120 000405          BR 2$ ; IF SUCCESSFUL, LEAVE IT ALONE ; 4
001122          ;                               ; 4
001122 012767 000176 177650 1$: MOV #176, SWR ; POINT TO SOFT SWITCH DHSCR ; 4
001130 005067 177646      CLR LIGHTS ; 0 MEANS WE ARE NOT GOING TO USE LIGHTS ; 4
001134          2$: ;                               ; 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 012764      TYPE ,MTITLE ;TYPE TITLE MESSAGE
001154 005767 010670      TST INIFLG ;CHECK INITIALIZATION FLAG

.IF NB <DHRVEC>
001160 001001      BNE VEC1 ;IF NOT 0, CHECK SWITCHES ;FOR REINITIALIZATION

.IFF
BNE BEGIN ;IF NOT 0, START TEST

.ENDC
.IF NB <>
SIZE: CLR RO
MOV #2$,0#4 ;SET UP TIME OUT RETURN
1$: TST (RO)+ ;WILL TRAP WHEN NO MEMORY ; 9
BR 1$ ;LOCATION RESPONDED, CONTINUE
2$: MOV RO,HCORE ;RO CONTAINS ADDRESS OF
SUB #2,HCORE ;NON EXISTANT MEMORY ; 9
MOV #6,0#4 ;RESTORE TRAPCATCHER

```

```

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

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

.ENDC
VEC1: BIT #SW00,@SWR ;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$: MOV R2,(R1) ;RESTORE TRAPCATCHER
    CLR (R2) ;IN FLOATING VECTOR AREA
    ADD R3,R1
    ADD R3,R2
    CMP R1,#1000
    BNE 1$
    INSTR ;INPUT ADDRESS OF DEVICE VECTOR
    MVECTOR ;MESSAGE "VECTOR ADDRESS-"
    PARAM ;CONVERT STRING TO OCTAL
    300 ;LOW LIMIT
    770 ;HIGH LIMIT ; 3
    DHRVEC ;LOCATIONS TO BE FILLED
    3 ;NUMBER OF LOCATIONS
    .BYTE 4 ;LSB MASK
    .BYTE 4 INSTR ;INPUT ADDRESS OF DEVICE CSR
    MREGAD ;MESSAGE "CONTROL REGISTER ADDRESS-"
    PARAM ;CONVERT STRING TO OCTAL
    0 ;LOW LIMIT
    177776 ;HIGH LIMIT
    DHSCR ;LOCATIONS TO BE FILLED
    7 ;NUMBER OF LOCATIONS
    .BYTE 10 ;LSB MASK
    .BYTE
.ENDC
.IF NB <1>
001262 CODEM1
001262 016767 010506 010506 MOV DHSSR,DHSLR ;SET UP ADDRESS OF SILO
001270 005267 010502 INC DHSLR ;STATUS REGISTER HIGH BYTE

.ENDC
001274 005767 010550 TST INIFLG ;IF INITIALIZATION FLAG
001300 001002 BNE BEGIN ;IS CLEARED
001302 005167 010542 COM INIFLG ;SET IT

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

```

```

001306 012767 000340 176462 BEGIN: MOV #340,PS ;LOCK OUT INTERRUPTS
001314 012706 013724 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 013243 MTSTPC ;MESSAGE "TEST PC"
001334 104405 PARAM ;CONVERT STRING TO OCTAL
001336 000000 0
001340 017500 17500
001342 012016 RETRN
001344 001 .BYTE 1
001345 001 .BYTE 1
001346 000410 BR 2#
001350 012767 001400 010440 1#: MOV #T1,RETRN ;NORMAL START, TEST 1
001356 005767 010470 TST STFLG ;IF LOOPING, BYPASS TYPEOUT
001362 001004 BNE 3#
001364 005167 010462 COM STFLG
001370 104401 013237 2#: TYPE ,MR ;TYPE "R" TO INDICATE START
001374 000177 010416 3#: JMP @RETRN ;START TESTING ; 3

```

```

1      .MACRO BREAK1 XLINE,XBIT,K
2
3      ;FLUSH UART BY TRANSMITTING 2 NULL CHARACTERS
4      ;ON LINE 'XLINE'
5      ;SET BREAK BIT FOR LINE 'XLINE'
6      ;TRANSMIT BINARY COUNT PATTERN ON LINE 'XLINE'
7      ;VERIFY HTAT ONLY 1 CHARACTER IS RECEIVED
8      ;AND THAT IT IS A BREAK
9
10     TS \XN,20,4$
11     MOV     #BIT11,@DHSCR           ;MASTER CLEAR INTERFACE
12     JSR     PC,CLRALL              ;CLEAR ALL BUS ADDRESS AND
13                                         ;BUS ADDRESS MEMORY LOCATIONS
14     MOV     #XLINE,@DHSCR          ;SELECT LINE XLINE
15     MOV     #NULL,@DMBA            ;SET UPT TO TRANSMIT 0 CHARACTER
16     MOV     #-2,@DMBC              ;TWO 0S WILL BE TRANSMITTED
17     MOV     #33503,@DHLPR         ;SET LINE SPEED=9600 BAUD
18                                         ;CHARACTER LENGTH =8 BITS
19     MOV     #'XBIT',@DMBAR         ;SET BAR BIT FOR LINE XLINE
20     1$:    CMPB  #2,@DHSLR          ;WAIT FOR 2 CHARACTERS TO BE RECEIVED
21     BNE     1$
22     MOV     #BIT11,@DHSCR          ;MASTER CLEAR INTERFACE
23     MOV     #XLINE,@DHSCR          ;SELECT LINE XLINE
24     MOV     #TBUF,@DMBA            ;SET UP TO TRANSMIT 400
25     MOV     #-400,@DMBC            ;(OCTAL) CHARACTERs
26     MOV     #33503,@DHLPR         ;LINE SPEED = 9600 BAUD
27     MOV     #XBIT,@DMBCR          ;SET BREAK BIT FOR LINE XLINE
28     MOV     #XBIT,@DMBAR          ;SET BAR BIT FOR LINE XLINE
29     2$:    TST  @DMBAR              ;WAIT FOR ALL CHARACTERS
30     BNE     2$                    ;TO BE TRANSMITTED
31     CMPB   #1,@DHSLR              ;CHECK TO SEE THAT ONLY
32     BEQ     3$                    ;1 CHARACTER WAS RECEIVED
33     MOV     @DHSSR,R4              ;(R4)=ACTUAL RECEIVED DATA
34     BIC     #300,R4                ;CLEAR UNWANTED BITS
35     MOV     #400,R5                ;(R5)=EXPECTED SILO FILL LEVEL, 1
36     HLT     0                      ;MORE THAN ONE CHARACTER RECEIVED, ERROR
37     3$:    MOV  @DHNRC,R4           ;READ NEXT RECEIVED CHARACTER REGISTER
38     CMP     RWRD'K',R4             ;IS RECEIVED CHARACTER A BREAK
39     BEQ     4$
40     MOV     RWRD'K',R5              ;(R5)=EXPECTED RECEIVED CHARACTER
41     HLT     1                      ;RECEIVED DATA ERROR
42     4$:    SCOPE                     ;CHECK FOR ITERATIONS, LOOP
43     .ENDM    BREAK1

```

```
1      .MACRO BLIND1,XLINE,XBIT,K
2
3      ;SET HALF DUPLEX ON LINE 'K'
4      ;TRANSMIT A BINARY COUNT PATTERN
5      ;VERIFY THAT NO CHARACTERS ARE RECEIVED
6
7      TS \XN,20,2‡
8          MOV    ‡BIT11,‡DHSCR      ;MASTER CLEAR INTERFACE
9          JSR    PC,CLRALL          ;CLEAR ALL BYTE COUNT AND
10                                     ;AND BUS ADDRESS MEMORY LOCATIONS
11          MOV    ‡XLINE,‡DHSCR     ;SELECT LINE XLINE
12          MOV    ‡TBUF,‡DHBA       ;SET UP TO TRANSMIT
13          MOV    ‡-400,‡DHBC       ;400 (OCTAL) CHARACTERS
14          MOV    ‡735C3,‡DHLP      ;SET RECEIVER BLIND
15                                     ;LINE SPEED =9600 BAUD
16                                     ;CHARACTER LENGTH = 8 BITS
17          MOV    ‡XBIT,‡DHBAR       ;SET BAR BIT FOR LINE XLINE
18      1‡:  TST    ‡DHBAR            ;WAIT FOR ALL CHARACTERS TO BE TRANSMITTED
19          BNE    1‡
20          TSTB   ‡DHSCR            ;WERE ANY CHARACTERS RECEIVED
21          BPL    2‡
22          HLT    2
23      2‡:  SCOPE
24      .ENDM BLIND1
```



```
1          .MACRO SSETALL
2
3          ;SET BYTE COUNT FOR ALL LINES TO 400
4          ;SET BUS ADDRESS FOR ALL LINES TO TBUF
5          ;CLEAR EXPECTED CHARACTER BUFFERS
6          ;SET LINE ACTIVE BITS FOR ALL LINES
7
8          SETALL: MOV    #20,R0          ;SET UP TO LOAD 20 MEMORY LOCATIONS
9                  CLR    R1            ;START WITH LINE 0
10                 MOV    #200,R2
11                 MOV    #1,R3
12          1$:     MOV    #TBUF,@DHBA
13                 MOV    #-400,@DHBC
14                 MOV    #31403,@DHLPR
15                 CLRB   RBUF(R1)
16                 MOVB  R2,RBUF(R3)
17                 INC    @DHSCR
18                 INC    R2
19                 ADD    #2,R1
20                 ADD    #2,R3
21                 DEC    R0
22                 BNE    1$
23                 MOV    #-1,LINACT
24                 RTS    PC
25          .ENDM SSETALL
26
27          .MACRO CCLRALL
28
29          ;CLEAR ALL BYTE COUNT AND BUS ADDRESS REGISTERS
30
31          CLRALL: MOV    #20,R0
32          1$:     CLR    @DHBA
33                 CLR    @DHBC
34                 INC    @DHSCR
35                 DEC    R0
36                 BNE    1$
37                 RTS    PC
38          .ENDM CCLRALL
```

```

2      000020      XLINE=LINE
3      000000      XBIT=BITX
4      000020      K=KX
5      000000      LINE=0
6      000001      BITX=1
7      000000      KX=0
9      000020      .REPT 20
10     BREAK1 \LINE,\BITX,\KX
11     .NLIST
12     LINE=LINE+1
13     BITX=BITX+BITX
14     KX=KX+1
15     .LIST
16     .ENDR
001400 BREAK1 \LINE,\BITX,\KX

;FLUSH UART BY TRANSMITTING 2 NULL CHARACTERS
;ON LINE 0
;SET BREAK BIT FOR LINE 0
;TRANSMIT BINARY COUNT PATTERN ON LINE 0
;VERIFY THAT ONLY 1 CHARACTER IS RECEIVED
;AND THAT IT IS A BREAK

001400 TS \XN,20,4#
001400 012767 000340 176370 T1: MOV #340,PS ;DISABLE ALL INTERRUPTS
001406 012767 000020 010410 MOV #20,ICOUNT ;SET UP FOR 20 ITERATIONS
001414 012767 001630 010376 MOV #4#,ESCAPE ;SET UP TO ESCAPE TO NEXT TEST

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

001422 000002
001430 004767 010430 MOV #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
JSR PC,CLRALL ;CLEAR ALL BUS ADDRESS AND
;BUS ADDRESS MEMORY LOCATIONS

001434 012777 000000 010314 MOV #0,@DHSCR ;SELECT LINE 0
001442 012777 012060 010314 MOV #NULL,@DHBA ;SET UP TO TRANSMIT 0 CHARACTER
001450 012777 177776 010310 MOV #-2,@DHBC ;TWO 0S WILL BE TRANSMITTED
001456 012777 033503 010276 MOV #33503,@DHLPR ;SET LINE SPEED=9600 BAUD
;CHARACTER LENGTH =8 BITS

001464 012777 000001 010276 MOV #1,@DHBAR ;SET BAR BIT FOR LINE 0
001472 122777 000002 010276 1#: CMPB #2,@DHSLR ;WAIT FOR 2 CHARACTERS TO BE RECEIVED
BNE 1#

001500 001374
001502 012777 004000 010246 MOV #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
001510 012777 000000 010240 MOV #0,@DHSCR ;SELECT LINE 0
001516 012777 012112 010240 MOV #TBUF,@DHBA ;SET UP TO TRANSMIT 400
001524 012777 177400 010234 MOV #-400,@DHBC ;(OCTAL) CHARACTERS
001532 012777 033503 010222 MOV #33503,@DHLPR ;LINE SPEED = 9600 BAUD
001540 012777 000001 010224 MOV #1,@DHBCR ;SET BREAK BIT FOR LINE 0
001546 012777 000001 010214 MOV #1,@DHBAR ;SET BAR BIT FOR LINE 0
001554 005777 010210 2#: TST @DHBAR ;WAIT FOR ALL CHARACTERS
BNE 2# ;TO BE TRANSMITTED
001562 122777 000001 010206 CMPB #1,@DHSLR ;CHECK TO SEE THAT ONLY
001570 001407 BEQ 3# ;1 CHARACTER WAS RECEIVED
001572 017704 010176 MOV @DHSSR,R4 ;(R4)=ACTUAL RECEIVED DATA
001576 042704 000300 BIC #300,R4 ;CLEAR UNWANTED BITS
001602 012705 000400 MOV #400,R5 ;(R5)=EXPECTED SILO FILL LEVEL, 1
001606 HLT 0 ;MORE THAN ONE CHARACTER RECEIVED, ERROR

```

```

001606 104000          EMT      0
001610 017704 010144  3$:  MOV    @DHNRC,R4      ;READ NEXT RECEIVED CHARACTER REGISTER
001614 026704 010734    CMP    RWRDO,R4          ;IS RECEIVED CHARACTER A BREAK
001620 001403          BEQ    #1
001622 016705 010726    MOV    RWRDO,R5          ;(R5)=EXPECTED RECEIVED CHARACTER
001626          HLT    1          ;RECEIVED DATA ERROR
001626 104001          EMT      1
001630 104400          4$:  SCOPE          ;CHECK FOR ITERATIONS, LOOP
      LINE=LINE+1
      BITX=BITX+BITX
      KX=KX+1
001632          BREAK1 \LINE,\BITX,\KX

      ;FLUSH UART BY TRANSMITTING 2 NULL CHARACTERS
      ;ON LINE 1
      ;SET BREAK BIT FOR LINE 1
      ;TRANSMIT BINARY COUNT PATTERN ON LINE 1
      ;VERIFY HTAT ONLY 1 CHARACTER IS RECEIVED
      ;AND THAT IT IS A BREAK

001632          TS \XN,20,4$
001632 012767 000340 176136 T2:  MOV    #340,PS          ;DISABLE ALL INTERRUPTS
001640 012767 000020 010156    MOV    #20,ICOUNT       ;SET UP FOR 20 ITERATIONS
0C1646 012767 002062 010144    MOV    #4$,ESCAPE       ;SET UP TO ESCAPE TO NEXT TEST
      .IF NB
      MOV    #,FREEZ1          ;SET UP TO LOOP WITH DATA      ; 3
      .ENDC
      XN=XN+1
001654 012777 004000 010074    MOV    #BIT11,@DHSCR     ;MASTER CLEAR INTERFACE
001662 004767 010176    JSR    PC,CLRALL        ;CLEAR ALL BUS ADDRESS AND
      ;BUS ADDRESS MEMORY LOCATIONS
001666 012777 000001 010062    MOV    #1,@DHSCR        ;SELECT LINE 1
001674 012777 012060 010062    MOV    #NULL,@DHBA      ;SET UP TO TRANSMIT 0 CHARACTER
001702 012777 177776 010056    MOV    #-2,@DHBC        ;TWO 0S WILL BE TRANSMITTED
001710 012777 033503 010044    MOV    #33503,@DHLPR    ;SET LINE SPEED=9600 BAUD
      ;CHARACTER LENGTH =8 BITS
001716 012777 000002 010044    MOV    #2,@DHBAR        ;SET BAR BIT FOR LINE 1
001724 122777 000002 010044 1$:  CMPB   #2,@DHSLR        ;WAIT FOR 2 CHARACTERS TO BE RECEIVED
001732 001374          BNE    1$
001734 012777 004000 010014    MOV    #BIT11,@DHSCR     ;MASTER CLEAR INTERFACE
001742 012777 000001 010006    MOV    #1,@DHSCR        ;SELECT LINE 1
001750 012777 012112 010006    MOV    #TBUF,@DHBA      ;SET UP TO TRANSMIT 400
001756 012777 177400 010002    MOV    #-400,@DHBC      ;(OCTAL) CHARACTERS
001764 012777 033503 007770    MOV    #33503,@DHLPR    ;LINE SPEED = 9600 BAUD
001772 012777 000002 007772    MOV    #2,@DHBCR        ;SET BREAK BIT FOR LINE 1
002000 012777 000002 007762    MOV    #2,@DHBAR        ;SET BAR BIT FOR LINE 1
002006 005777 007756 2$:  TST    @DHBAR          ;WAIT FOR ALL CHARACTERS
002012 001375          BNE    2$              ;TO BE TRANSMITTED
002014 122777 000001 007754    CMPB   #1,@DHSLR        ;CHECK TO SEE THAT ONLY
002022 001407          BEQ    3$              ;1 CHARACTER WAS RECEIVED
002024 017704 007744          MOV    @DHSSR,R4        ;(R4)=ACTUAL RECEIVED DATA
002030 042704 000300          BIC    #300,R4          ;CLEAR UNWANTED BITS
002034 012705 000400          MOV    #400,R5         ;(R5)=EXPECTED SILO FILL LEVEL, 1
002040          HLT    0          ;MORE THAN ONE CHARACTER RECEIVED, ERROR
002040 104000          EMT      0
002042 017704 007712  3$:  MOV    @DHNRC,R4      ;READ NEXT RECEIVED CHARACTER REGISTER
002046 026704 010504    CMP    RWRD1,R4        ;IS RECEIVED CHARACTER A BREAK

```

```

002052 001403          BEQ      4$
002054 016705 010476    MOV      RWRD1,R5      ;(R5)=EXPECTED RECEIVED CHARACTER
002060          HLT      1      ;RECEIVED DATA ERROR
002060 104001          EMT      1
002062 104400          4$: SCOPE          ;CHECK FOR ITERATIONS, LOOP
000002          LINE=LINE+1
000004          BITX=BITX+BITX
000002          KX=KX+1
002064          BREAK1 \LINE,\BITX,\KX

          ;FLUSH UART BY TRANSMITTING 2 NULL CHARACTERS
          ;ON LINE 2
          ;SET BREAK BIT FOR LINE 2
          ;TRANSMIT BINARY COUNT PATTERN ON LINE 2
          ;VERIFY THAT ONLY 1 CHARACTER IS RECEIVED
          ;AND THAT IT IS A BREAK

002064          TS \XN,20,4$
002064 012767 000340 175704 T3: MOV      #340,PS      ;DISABLE ALL INTERRUPTS
002072 012767 000020 007724 MOV      #20,ICOUNT    ;SET UP FOR 20 ITERATIONS
002100 012767 002314 007712 MOV      #4$,ESCAPE    ;SET UP TO ESCAPE TO NEXT TEST

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

002106 012777 004000 007642 MOV      #BIT11,@DHSCR   ;MASTER CLEAR INTERFACE
002114 004767 007744 JSR      PC,CLRALL      ;CLEAR ALL BUS ADDRESS AND
          ;BUS ADDRESS MEMORY LOCATIONS
          ;SELECT LINE 2
002120 012777 000002 007630 MOV      #2,@DHSCR      ;SET UP TO TRANSMIT 0 CHARACTER
002126 012777 012060 007630 MOV      #NULL,@DHBA    ;TWO 0S WILL BE TRANSMITTED
002134 012777 177776 007624 MOV      #-2,@DHBC      ;SET LINE SPEED=9600 BAUD
002142 012777 033503 007612 MOV      #33503,@DHLPR  ;CHARACTER LENGTH =8 BITS
          ;SET BAR BIT FOR LINE 2
002150 012777 000004 007612 MOV      #4,@DHBAR      ;WAIT FOR 2 CHARACTERS TO BE RECEIVED
002156 122777 000002 007612 1$: CMPB     #2,@DHSLR
002164 001374 BNE     1$
002166 012777 004000 007562 MOV      #BIT11,@DHSCR   ;MASTER CLEAR INTERFACE
002174 012777 000002 007554 MOV      #2,@DHSCR      ;SELECT LINE 2
002202 012777 012112 007554 MOV      #TBUF,@DHBA    ;SET UP TO TRANSMIT 400
002210 012777 177400 007550 MOV      #-400,@DHBC    ;(OCTAL) CHARACTERS
002216 012777 033503 007536 MOV      #33503,@DHLPR  ;LINE SPEED = 9600 BAUD
002224 012777 000004 007540 MOV      #4,@DHBCR      ;SET BREAK BIT FOR LINE 2
002232 012777 000004 007530 MOV      #4,@DHBAR      ;SET BAR BIT FOR LINE 2
002240 005777 007524 2$: TST      @DHBAR
002244 001375 BNE     2$
002246 122777 000001 007522 CMPB     #1,@DHSLR
002254 001407 BEQ     3$
002256 017704 007512 MOV      @DHSSR,R4      ;CHECK TO SEE THAT ONLY
          ;1 CHARACTER WAS RECEIVED
002262 042704 000300 BIC     #300,R4        ;(R4)=ACTUAL RECEIVED DATA
002266 012705 000400 MOV      #400,R5        ;CLEAR UNWANTED BITS
002272          HLT      0      ;(R5)=EXPECTED SILO FILL LEVEL, 1
          ;MORE THAN ONE CHARACTER RECEIVED, ERROR
002272 104000          EMT      0
002274 017704 007460 3$: MOV      @DHNR, R4      ;READ NEXT RECEIVED CHARACTER REGISTER
002300 026704 010254 CMP      RWRD2,R4      ;IS RECEIVED CHARACTER A BREAK
002304 001403 BEQ     4$
002306 016705 010246 MOV      RWRD2,R5      ;(R5)=EXPECTED RECEIVED CHARACTER
002312          HLT      1      ;RECEIVED DATA ERROR

```

```

002312 104001          EMT      1
002314 104400          4$: SCOPE                                ;CHECK FOR ITERATIONS, LOOP
000003          LINE=LINE+1
000010          BITX=BITX+8,1
000003          KX=KX+1
002316          BREAK1 \LINE,\BITX,\KX

;FLUSH UART BY TRANSMITTING 2 NULL CHARACTERS
;ON LINE 3
;SET BREAK BIT FOR LINE 3
;TRANSMIT BINARY COUNT PATTERN ON LINE 3
;VERIFY THAT ONLY 1 CHARACTER IS RECEIVED
;AND THAT IT IS A BREAK

002316          TS \XN,20,4$
002316 012767 000340 175452 T4:  MOV      #340,PS                                ;DISABLE ALL INTERRUPTS
002324 012767 000020 007472      MOV      #20,ICOUNT                            ;SET UP FOR 20 ITERATIONS
002332 012767 002546 007460      MOV      #4$,ESCAPE                                ;SET UP TO ESCAPE TO NEXT TEST

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

002340 012777 004000 007410      MOV      #BIT11,@DHSCR                            ;MASTER CLEAR INTERFACE
002346 004767 007512              JSR      PC,CLRALL                                ;CLEAR ALL BUS ADDRESS AND
;BUS ADDRESS MEMORY LOCATIONS
002352 012777 000003 007376      MOV      #3,@DHSCR                                ;SELECT LINE 3
002360 012777 012060 007376      MOV      #NULL,@DHBA                            ;SET UP TO TRANSMIT 0 CHARACTER
002366 012777 177776 007372      MOV      #-2,@DHBC                              ;TWO 0S WILL BE TRANSMITTED
002374 012777 033503 007360      MOV      #33503,@DHLPR                          ;SET LINE SPEED=9600 BAUD
;CHARACTER LENGTH =8 BITS
;SET BAR BIT FOR LINE 3
002402 012777 000010 007360      MOV      #10,@DHBAR                             ;WAIT FOR 2 CHARACTERS TO BE RECEIVED
002410 122777 000002 007360 1$:  CMPB     #2,@DHSLR
002416 001374          BNE     1$
002420 012777 004000 007330      MOV      #BIT11,@DHSCR                            ;MASTER CLEAR INTERFACE
002426 012777 000003 007322      MOV      #3,@DHSCR                                ;SELECT LINE 3
002434 012777 012112 007322      MOV      #TBUF,@DHBA                            ;SET UP TO TRANSMIT 400
002442 012777 177400 007316      MOV      #-400,@DHBC                             ;(OCTAL) CHARACTERS
002450 012777 033503 007304      MOV      #33503,@DHLPR                          ;LINE SPEED = 9600 BAUD
002456 012777 000010 007306      MOV      #10,@DHBC                              ;SET BREAK BIT FOR LINE 3
002464 012777 000010 007276      MOV      #10,@DHBAR                             ;SET BAR BIT FOR LINE 3
002472 005777 007272          2$:  TST     @DHBAR
002476 001375          BNE     2$
002500 122777 000001 007270      CMPB     #1,@DHSLR
002506 001407          BEQ     3$
002510 017704 007260          MOV      @DHSSR,R4                                ;(R4)=ACTUAL RECEIVED DATA
002514 042704 000300          BIC     #300,R4
002520 012705 000400          MOV      #400,R5                                ;(R5)=EXPECTED SILO FILL LEVEL, 1
002524          HLT     0                                ;MORE THAN ONE CHARACTER RECEIVED. ERROR
002524 104000          EMT     0
002526 017704 007226          3$:  MOV      @DHNR, R4                                ;READ NEXT RECEIVED CHARACTER REGISTER
002532 026704 010024          CMP     RWRD3,R4                                ;IS RECEIVED CHARACTER A BREAK
002536 001403          BEQ     4$
002540 016705 010016          MOV     RWRD3,R5                                ;(R5)=EXPECTED RECEIVED CHARACTER
002544          HLT     1                                ;RECEIVED DATA ERROR
002544 104001          EMT     1
002546 104400          4$:  SCOPE                                ;CHECK FOR ITERATIONS, LOOP
000004          LINE=LINE+1

```

```

000020
000004
002550          BITX=BITX+BITX
                KX=KX+1
                BREAK1 \LINE,\BITX,\KX

                ;FLUSH UART BY TRANSMITTING 2 NULL CHARACTERS
                ;ON LINE 4
                ;SET BREAK BIT FOR LINE 4
                ;TRANSMIT BINARY COUNT PATTERN ON LINE 4
                ;VERIFY THAT ONLY 1 CHARACTER IS RECEIVED
                ;AND THAT IT IS A BREAK

002550          TS \XN,20,4#
002550 012767 000340 175220 TS:   MOV    #340,PS          ;DISABLE ALL INTERRUPTS
002556 012767 000020 007240      MOV    #20,ICOUNT      ;SET UP FOR 20 ITERATIONS
002564 012767 003000 007226      MOV    #4#,ESCAPE     ;SET UP TO ESCAPE TO NEXT TEST

                .IF NB <>
                MOV    #,FREEZ1          ;SET UP TO LOOP WITH DATA          ; 3
                .ENDC
                XN=XN+1
002572 012777 004000 007156      MOV    #BIT11,@DHSCR   ;MASTER CLEAR INTERFACE
002600 004767 007260                JSR    PC,CLRALL       ;CLEAR ALL BUS ADDRESS AND
                                ;BUS ADDRESS MEMORY LOCATIONS
002604 012777 000004 007144      MOV    #4,@DHSCR      ;SELECT LINE 4
002612 012777 012060 007144      MOV    #NULL,@DHBA    ;SET UP TO TRANSMIT 0 CHARACTER
002620 012777 177776 007140      MOV    #-2,@DHBC      ;TWO 0S WILL BE TRANSMITTED
002626 012777 033503 007126      MOV    #33503,@DHLPR  ;SET LINE SPEED=9600 BAUD
                                ;CHARACTER LENGTH =8 BITS
                                ;SET BREAK BIT FOR LINE 4
002634 012777 000020 007126      MOV    #20,@DHBAR     ;WAIT FOR 2 CHARACTERS TO BE RECEIVED
002642 122777 000002 007126 1#:   CMPB   #2,@DHSLR
002650 001374                BNE    1#
002652 012777 004000 007076      MOV    #BIT11,@DHSCR   ;MASTER CLEAR INTERFACE
002660 012777 000004 007070      MOV    #4,@DHSCR      ;SELECT LINE 4
002666 012777 012112 007070      MOV    #TBUF,@DHBA    ;SET UP TO TRANSMIT 400
002674 012777 177400 007064      MOV    #-400,@DHBC    ;(OCTAL) CHARACTERS
002702 012777 033503 007052      MOV    #33503,@DHLPR  ;LINE SPEED = 9600 BAUD
002710 012777 000020 007054      MOV    #20,@DHBCR     ;SET BREAK BIT FOR LINE 4
002716 012777 000020 007044      MOV    #20,@DHBAR     ;SET BREAK BIT FOR LINE 4
002724 005777 007040 2#:   TST    @DHBAR
002730 001375                BNE    2#
                                ;WAIT FOR ALL CHARACTERS
                                ;TO BE TRANSMITTED
002732 122777 000001 007036      CMPB   #1,@DHSLR
002740 001407                BEQ    3#
                                ;CHECK TO SEE THAT ONLY
                                ;1 CHARACTER WAS RECEIVED
002742 017704 007026      MOV    @DHSSR,R4      ;(R4)=ACTUAL RECEIVED DATA
002746 042704 000300      BIC    #300,R4        ;CLEAR UNWANTED BITS
002752 012705 000400      MOV    #400,R5        ;(R5)=EXPECTED SILO FILL LEVEL, 1
002756                HLT    0
                                ;MORE THAN ONE CHARACTER RECEIVED, ERROR
002756 104000                EMT    0
002760 017704 006774 3#:   MOV    @DHNRC,R4      ;READ NEXT RECEIVED CHARACTER REGISTER
002764 026704 007574      CMP    RWRD4,R4       ;IS RECEIVED CHARACTER A BREAK
002770 001403                BEQ    4#
002772 016705 007566      MOV    RWRD4,R5       ;(R5)=EXPECTED RECEIVED CHARACTER
002776                HLT    1
                                ;RECEIVED DATA ERROR
002776 104001                EMT    1
003000 104400 4#:   SCOPE
                LINE=LINE+1
                BITX=BITX+BITX
                KX=KX+1
                BREAK1 \LINE,\BITX,\KY
003002          000005
                000040
                000005

```

```

;FLUSH UART BY TRANSMITTING 2 NULL CHARACTERS
;ON LINE 5
;SET BREAK BIT FOR LINE 5
;TRANSMIT BINARY COUNT PATTERN ON LINE 5
;VERIFY THAT ONLY 1 CHARACTER IS RECEIVED
;AND THAT IT IS A BREAK

003002
003002 012767 000340 174766 TS \XN,20,4#
003010 012767 000020 007006 T6: MOV #340,PS ;DENABLE ALL INTERRUPTS
003016 012767 003232 006774 MOV #20,ICOUNT ;SET UP FOR 20 ITERATIONS
MOV #4#,ESCAPE ;SET UP TO ESCAPE TO NEXT TEST
.JF N6 <>
MOV #,FREEZ1 ;SET UP TO LOOP WITH DATA ; 3
.ENDC
XN=XN+1

003024 000007
003032 012777 004000 006724 MOV #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
JSR PC,CLRALL ;CLEAR ALL BUS ADDRESS AND
;BUS ADDRESS MEMORY LOCATIONS

003036 012777 000005 006712 MOV #5,@DHSCR ;SELECT LINE 5
003044 012777 012060 006712 MOV #NULL,@DHBA ;SET UP TO TRANSMIT 0 CHARACTER
003052 012777 177776 006706 MOV #-2,@DHBC ;TWO 0S WILL BE TRANSMITTED
003060 012777 033503 006674 MOV #33503,@DHLPR ;SET LINE SPEED=9600 BAUD
;CHARACTER LENGTH =8 BITS

003066 012777 000040 006674 MOV #40,@DHBAR ;SET BAR BIT FOR LINE 5
003074 122777 000002 006674 1#: CMPB #2,@DHSLR ;WAIT FOR 2 CHARACTERS TO BE RECEIVED
003102 001374 BNE 1#
003104 012777 004000 006644 MOV #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
003112 012777 000005 006636 MOV #5,@DHSCR ;SELECT LINE 5
003120 012777 012112 006636 MOV #TBUF,@DHBA ;SET UP TO TRANSMIT 400
003126 012777 177400 006632 MOV #-400,@DHBC ;(OCTAL) CHARACTERS
003134 012777 033503 006620 MOV #33503,@DHLPR ;LINE SPEED = 9600 BAUD
003142 012777 000040 006622 MOV #40,@DHBCR ;SET BREAK BIT FOR LINE 5
003150 012777 000040 006612 MOV #40,@DHBAR ;SET BAR BIT FOR LINE 5
003156 005777 006606 2#: TST @DHBAR ;WAIT FOR ALL CHARACTERS
003162 001375 BNE 2# ;TO BE TRANSMITTED
003164 122777 000001 006604 CMPB #1,@DHSLR ;CHECK TO SEE THAT ONLY
003172 001407 BEQ 3# ;1 CHARACTER WAS RECEIVED
003174 017704 006574 MOV @DHSSR,R4 ;(R4)=ACTUAL RECEIVED DATA
003200 042704 000300 BIC #300,R4 ;CLEAR UNWANTED BITS
003204 012705 000400 MOV #400,R5 ;(R5)=EXPECTED SILO FILL LEVEL, 1
003210 HLT 0 ;MORE THAN ONE CHARACTER RECEIVED, ERROR
003210 104000 EMT 0
003212 017704 006542 3#: MOV @DHNRX,R4 ;READ NEXT RECEIVED CHARACTER REGISTER
003216 026704 007344 CMP RWRD5,R4 ;IS RECEIVED CHARACTER A BREAK
003222 001403 BEQ 4#
003224 016705 007336 MOV RWRD5,R5 ;(R5)=EXPECTED RECEIVED CHARACTER
003230 HLT 1 ;RECEIVED DATA ERROR
003230 104001 EMT 1
003232 104400 4#: SCOPE ;CHECK FOR ITERATIONS. LOOP
000006 LINE=LINE+1
000100 BITX=BITX+BITX
000006 KX=KX+1
003234 BREAK1 \LINE,\BITX,\KX

;FLUSH UART BY TRANSMITTING 2 NULL CHARACTERS
;ON LINE 6

```

```

;SET BREAK BIT FOR LINE 6
;TRANSMIT BINARY COUNT PATTERN ON LINE 6
;VERIFY HTAT ONLY 1 CHARACTER IS RECEIVED
;AND THAT IT IS A BREAK

003234 012767 000340 174534 TS \XN,20,4#
003242 012767 000020 006554 T7: MOV #340,PS ;DISABLE ALL INTERRUPTS
003250 012767 003464 006542 MOV #20,ICOUNT ;SET UP FOR 20 ITERATIONS
MOV #4,ESCAPE ;SET UP TO ESCAPE TO NEXT TEST
;IF NB <>
MOV #,FREEZ1 ;SET UP TO LOOP WITH DATA ; 3
.ENDC
XN=XN+1
003256 000010 012777 004000 006472 MOV #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
003264 004767 006574 JSR PC,CLRALL ;CLEAR ALL BUS ADDRESS AND
;BUS ADDRESS MEMORY LOCATIONS
003270 012777 000006 006460 MOV #6,@DHSCR ;SELECT LINE 6
003276 012777 012060 006460 MOV #NULL,@DHBA ;SET UPT TO TRANSMIT 0 CHARACTER
003304 012777 177776 006454 MOV #-2,@DHBC ;TWO OS WILL BE TRANSMITTED
003312 012777 033503 006442 MOV #33503,@DHLPR ;SET LINE SPEED=9600 BAUD
;CHARACTER LENGTH =8 BITS
003320 012777 000100 006442 MOV #100,@DHBAR ;SET BAR BIT FOR LINE 6
003326 122777 000002 006442 1#: CMPB #2,@DHSLR ;WAIT FOR 2 CHARACTERS TO BE RECEIVED
003334 001374 BNE 1#
003336 012777 004000 006412 MOV #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
003344 012777 000006 006404 MOV #6,@DHSCR ;SELECT LINE 6
003352 012777 012112 006404 MOV #TBUF,@DHBA ;SET UP TO TRANSMIT 400
003360 012777 177400 006400 MOV #-400,@DHBC ;(OCTAL) CHARACTERS
003366 012777 033503 006366 MOV #33503,@DHLPR ;LINE SPEED = 9600 BAUD
003374 012777 000100 006370 MOV #100,@DHBCR ;SET BREAK BIT FOR LINE 6
003402 012777 000100 006360 MOV #100,@DHBAR ;SET BAR BIT FOR LINE 6
003410 005777 006354 2#: TST @DHBAR ;WAIT FOR ALL CHARACTERS
003414 001375 BNE 2# ;TO BE TRANSMITTED
003416 122777 000001 006352 CMPB #1,@DHSLR ;CHECK TO SEE THAT ONLY
003424 001407 BEQ 3# ;1 CHARACTER WAS RECEIVED
003426 017704 006342 MOV @DHSSR,R4 ;(R4)=ACTUAL RECEIVED DATA
003432 042704 000300 BIC #300,R4 ;CLEAR UNWANTED BITS
003436 012705 000400 MOV #400,R5 ;(R5)=EXPECTED SILO FILL LEVEL, 1
003442 HLT 0 ;MORE THAN ONE CHARACTER RECEIVED, ERROR
003442 104030 EMT 0
003444 017704 006310 3#: MOV @DHRC,R4 ;READ NEXT RECEIVED CHARACTER REGISTER
003450 026704 007114 CMP RWRD6,R4 ;IS RECEIVED CHARACTER A BREAK
003454 001403 BEQ 4#
003456 016705 007106 MOV RWRD6,R5 ;(R5)=EXPECTED RECEIVED CHARACTER
003462 HLT 1 ;RECEIVED DATA ERROR
003462 104001 EMT 1
003464 104400 4#: SCOPE ;CHECK FOR ITERATIONS, LOOP
LINE=LINE+1
000007 BITX=BITX+BITX
000200 KX=KX+1
000007 BREAK1 \LINE,\BITX,\KX

;FLUSH UART BY TRANSMITTING 2 NULL CHARACTERS
;ON LINE 7
;SET BREAK BIT FOR LINE 7
;TRANSMIT BINARY COUNT PATTERN ON LINE 7
;VERIFY HTAT ONLY 1 CHARACTER IS RECEIVED

```



```

                                ;AND THAT IT IS A BREAK
003466                          TS \XN,20,4#
003466 012767 000340 174302 T10:  MOV    #340,PS          ;DISABLE ALL INTERRUPTS
003474 012767 000020 006322      MOV    #20,ICOUNT      ;SET UP FOR 20 ITERATIONS
003502 012767 003716 006310      MOV    #4#,ESCAPE     ;SET UP TO ESCAPE TO NEXT TEST
                                .IF NB <>
                                MOV    #,FREEZ1          ;SET UP TO LOOP WITH DATA      ; 3
                                .ENDC
                                XN=XN+1
003510 012777 004000 006240      MOV    #BIT11,@DHSCR   ;MASTER CLEAR INTERFACE
003516 004767 006342              JSR    PC,CLRALL      ;CLEAR ALL BUS ADDRESS AND
                                ;BUS ADDRESS MEMORY LOCATIONS
003522 012777 000007 006226      MOV    #7,@DHSCR      ;SELECT LINE 7
003530 012777 012060 006226      MOV    #NULL,@DHBA    ;SET UP TO TRANSMIT 0 CHARACTER
003536 012777 177776 006222      MOV    #-2,@DHBC      ;TWO OS WILL BE TRANSMITTED
003544 012777 033503 006210      MOV    #33503,@DHLPR  ;SET LINE SPEED=9600 BAUD
                                ;CHARACTER LENGTH =8 BITS
003552 012777 000200 006210      MOV    #200,@DHBAR    ;SET BAR BIT FOR LINE 7
003560 122777 000002 006210 1#:  CMPB   #2,@DHSLR      ;WAIT FOR 2 CHARACTERS TO BE RECEIVED
003566 001374                    BNE    1#
003570 012777 004000 006160      MOV    #BIT11,@DHSCR   ;MASTER CLEAR INTERFACE
003576 012777 000007 006152      MOV    #7,@DHSCR      ;SELECT LINE 7
003604 012777 012112 006152      MOV    #TBUF,@DHBA    ;SET UP TO TRANSMIT 400
003612 012777 177400 006146      MOV    #-400,@DHBC    ;(OCTAL) CHARACTERS
003620 012777 033503 006134      MOV    #33503,@DHLPR  ;LINE SPEED = 9600 BAUD
003626 012777 000200 006136      MOV    #200,@DHBCR    ;SET BREAK BIT FOR LINE 7
003634 012777 000200 006126      MOV    #200,@DHBAR    ;SET BAR BIT FOR LINE 7
003642 005777 006122 2#:  TST    @DHBAR        ;WAIT FOR ALL CHARACTERS
003646 001375                    BNE    2#            ;TO BE TRANSMITTED
003650 122777 000001 006120      CMPB   #1,@DHSLR      ;CHECK TO SEE THAT ONLY
003656 001407                    BEQ    3#            ;1 CHARACTER WAS RECEIVED
003660 017704 006110              MOV    @DHSSR,R4      ;(R4)=ACTUAL RECEIVED DATA
003664 042704 000300              BIC    #300,R4        ;CLEAR UNWANTED BITS
003670 012705 000400              MOV    #400,R5        ;(R5)=EXPECTED SILO FILL LEVEL, 1
003674                          HLT    0                ;MORE THAN ONE CHARACTER RECEIVED, ERROR
003674 104000                    EMT    0
003676 017704 006056 3#:  MOV    @DHNRC,R4      ;READ NEXT RECEIVED CHARACTER REGISTER
003702 026704 006664              CMP    RWRD7,R4      ;IS RECEIVED CHARACTER A BREAK
003706 001403                    BEQ    4#
003710 016705 006656              MOV    RWRD7,R5      ;(R5)=EXPECTED RECEIVED CHARACTER
003714                          HLT    1                ;RECEIVED DATA ERROR
003716 104001                    EMT    1
                                4#:  SCOPE
                                LINE=LINE+1
                                BITX=BITX+BITX
                                KX=KX+1
003720                          BREAK1 \LINE,\BITX,\KX

                                ;FLUSH UART BY TRANSMITTING 2 NULL CHARACTERS
                                ;ON LINE 10
                                ;SET BREAK BIT FOR LINE 10
                                ;TRANSMIT BINARY COUNT PATTERN ON LINE 10
                                ;VERIFY THAT ONLY 1 CHARACTER IS RECEIVED
                                ;AND THAT IT IS A BREAK

003720                          TS \XN,20,4#

```

```

003720 012767 000340 174050 T11:  MOV    #340,PS          ;DISABLE ALL INTERRUPTS
003726 012767 000020 006070      MOV    #20,ICOUNT       ;SET UP FOR 20 ITERATIONS
003734 012767 004150 006056      MOV    #4$,ESCAPE      ;SET UP TO ESCAPE TO NEXT TEST
                                .IF NB  <>
                                MOV    #,FREEZ1          ;SET UP TO LOOP WITH DATA      ; 3
                                .ENDC
                                XN=XN+1
003742 012777 004000 006006      MOV    #BIT11,@DHSCR   ;MASTER CLEAR INTERFACE
003750 004767 006110                JSR    PC,CLRALL       ;CLEAR ALL BUS ADDRESS AND
                                ;BUS ADDRESS MEMORY LOCATIONS
003754 012777 000010 005774      MOV    #10,@DHSCR     ;SELECT LINE 10
003762 012777 012060 005774      MOV    #NULL,@DHBA    ;SET UP TO TRANSMIT 0 CHARACTER
003770 012777 177776 005770      MOV    #-2,@DHBC      ;TWO 0S WILL BE TRANSMITTED
003776 012777 033503 005756      MOV    #33503,@DHLPR  ;SET LINE SPEED=9600 BAUD
                                ;CHARACTER LENGTH =8 BITS
                                ;SET BAR BIT FOR LINE 10
004004 012777 000400 005756      MOV    #400,@DHBAR    ;WAIT FOR 2 CHARACTERS TO BE RECEIVED
004012 122777 000002 005756 1$:  CMPB   #2,@DHSLR
004020 001374                BNE    1$
004022 012777 004000 005726      MOV    #BIT11,@DHSCR   ;MASTER CLEAR INTERFACE
004030 012777 000010 005720      MOV    #10,@DHSCR     ;SELECT LINE 10
004036 012777 012112 005720      MOV    #TBUF,@DHBA    ;SET UP TO TRANSMIT 400
004044 012777 177400 005714      MOV    #-400,@DHBC    ;(OCTAL) CHARACTERS
004052 012777 033503 005702      MOV    #33503,@DHLPR  ;LINE SPEED = 9600 BAUD
004060 012777 000400 005704      MOV    #400,@DHBCR    ;SET BREAK BIT FOR LINE 10
004066 012777 000400 005674      MOV    #400,@DHBAR    ;SET BAR BIT FOR LINE 10
004074 005777 005670 2$:  TST    @DHBAR          ;WAIT FOR ALL CHARACTERS
004100 001375                BNE    2$             ;TO BE TRANSMITTED
004102 122777 000001 005666      CMPB   #1,@DHSI R     ;CHECK TO SEE THAT ONLY
004110 001407                BEQ    3$             ;1 CHARACTER WAS RECEIVED
004112 017704 005656      MOV    @DHSSR,R4      ;(R4)=ACTUAL RECEIVED DATA
004116 042704 000300      BIC    #300,R4        ;CLEAR UNWANTED BITS
004122 012705 000400      MOV    #400,R5        ;(R5)=EXPECTED SILO FILL LEVEL, 1
004126                HLT    0              ;MORE THAN ONE CHARACTER RECEIVED, ERROR
004126 104000                EMT    0
004130 017704 005624 3$:  MOV    @DHNR, R4      ;READ NEXT RECEIVED CHARACTER REGISTER
004134 026704 006434      CMP    RWRD10,R4     ;IS RECEIVED CHARACTER A BREAK
004140 001403                BEQ    4$
004142 016705 006426      MOV    RWRD10,R5     ;(R5)=EXPECTED RECEIVED CHARACTER
004146                HLT    1              ;RECEIVED DATA ERROR
004146 104001                EMT    1
004150 104400 4$:  SCOPE          ;CHECK FOR ITERATIONS, LOOP
                                LINE=LINE+1
                                BITX=BITX+BITX
                                KX=KX+1
004152                BREAK1 \LINE,\BITX,\KX

                                ;FLUSH UART BY TRANSMITTING 2 NULL CHARACTERS
                                ;ON LINE 11
                                ;SET BREAK BIT FOR LINE 11
                                ;TRANSMIT BINARY COUNT PATTERN ON LINE 11
                                ;VERIFY THAT ONLY 1 CHARACTER IS RECEIVED
                                ;AND THAT IT IS A BREAK

004152                TS \XN,20,4$
004152 012767 000340 173616 T12:  MOV    #340,PS          ;DISABLE ALL INTERRUPTS
004160 012767 000020 005636      MOV    #20,ICOUNT       ;SET UP FOR 20 ITERATIONS
004166 012767 004402 005624      MOV    #4$,ESCAPE      ;SET UP TO ESCAPE TO NEXT TEST

```

```

      .IF NB <>
      MOV     @,FREEZ1                ;SET UP TO LOOP WITH DATA      ; 3
      .ENDC
      XN=XN+1
004174 000013 004000 005554      MOV     @BIT11,@DHSCR                ;MASTER CLEAR INTERFACE
004202 004767 005656                JSR     PC,CLRALL                    ;CLEAR ALL BUS ADDRESS AND
                                          ;BUS ADDRESS MEMORY LOCATIONS
004206 012777 000011 005542      MOV     @11,@DHSCR                   ;SELECT LINE 11
004214 012777 012060 005542      MOV     @NULL,@DHBA                  ;SET UP TO TRANSMIT 0 CHARACTER
004222 012777 177776 005536      MOV     @-2,@DHBC                     ;TWO OS WILL BE TRANSMITTED
004230 012777 033503 005524      MOV     @33503,@DHLPR                 ;SET LINE SPEED=9600 BAUD
                                          ;CHARACTER LENGTH =8 BITS
004236 012777 001000 005524      MOV     @1000,@DHBCR                 ;SET BAR BIT FOR LINE 11
004244 122777 000002 005524 1$:    CMPB   @2,@DHSLR                      ;WAIT FOR 2 CHARACTERS TO BE RECEIVED
004252 001374                BNE     1$
004254 012777 004000 005474      MOV     @BIT11,@DHSCR                ;MASTER CLEAR INTERFACE
004262 012777 000011 005466      MOV     @11,@DHSCR                   ;SELECT LINE 11
004270 012777 012112 005466      MOV     @TBUF,@DHBA                  ;SET UP TO TRANSMIT 400
004276 012777 177400 005462      MOV     @-400,@DHBC                  ;(OCTAL) CHARACTERS
004304 012777 033503 005450      MOV     @33503,@DHLPR                 ;LINE SPEED = 9600 BAUD
004312 012777 001000 005452      MOV     @1000,@DHBCR                 ;SET BREAK BIT FOR LINE 11
004320 012777 001000 005442      MOV     @1000,@DHBAR                 ;SET BAR BIT FOR LINE 11
004326 005777 005436 2$:    TST    @DHBAR                        ;WAIT FOR ALL CHARACTERS
004332 001375                BNE     2$                            ;TO BE TRANSMITTED
004334 122777 000001 005434      CMPB   @1,@DHSLR                      ;CHECK TO SEE THAT ONLY
004342 001407                BEQ     3$                            ;1 CHARACTER WAS RECEIVED
004344 017704 005424                MOV     @DHSSR,R4                     ;(R4)=ACTUAL RECEIVED DATA
004350 042704 000300                BIC     @300,R4                        ;CLEAR UNWANTED BITS
004354 012705 000400                MOV     @400,R5                        ;(R5)=EXPECTED SILO FILL LEVEL, 1
004360                HLT     0                              ;MORE THAN ONE CHARACTER RECEIVED, ERROR
004362 104000                EMT     0
004366 017704 005372 3$:    MOV     @DHNRC,R4                     ;READ NEXT RECEIVED CHARACTER REGISTER
004372 001403                CMP     RWRD11,R4                      ;IS RECEIVED CHARACTER A BREAK
004374 016705 006176                BEQ     4$
004400                MOV     RWRD11,R5                      ;(R5)=EXPECTED RECEIVED CHARACTER
004400 104001                HLT     1                              ;RECEIVED DATA ERROR
004402 104400                EMT     1
4$:    SCOPE                            ;CHECK FOR ITERATIONS, LOOP
      LINE=LINE+1
      BITX=BITX+BITX
      KX=KX+1
004404      BREAK1 \LINE,\BITX,\KX

      ;FLUSH UART BY TRANSMITTING 2 NULL CHARACTERS
      ;ON LINE 12
      ;SET BREAK BIT FOR LINE 12
      ;TRANSMIT BINARY COUNT PATTERN ON LINE 12
      ;VERIFY HTAT ONLY 1 CHARACTER IS RECEIVED
      ;AND THAT IT IS A BREAK

004404      TS \XN,20,4$
004404 012767 000340 173364      T13:   MOV     @340,PS                    ;DISABLE ALL INTERRUPTS
004412 012767 000020 005404      MOV     @20,ICOUNT                    ;SET UP FOR 20 ITERATIONS
004420 012767 004634 005372      MOV     @4$,ESCAPE                    ;SET UP TO ESCAPE TO NEXT TEST

      .IF NB <>
      MOV     @,FREEZ1                ;SET UP TO LOOP WITH DATA      ; 3
      .ENDC

```

```

000014      XN=XN+1
004426 012777 004000 005322      MOV    #BIT11,@DHSCP      ;MASTER CLEAR INTERFACE
004434 004767 005424                JSR    PC,CLRALL          ;CLEAR ALL BUS ADDRESS AND
                                ;BUS ADDRESS MEMORY LOCATIONS
004440 012777 000012 005310      MOV    #12,@DHSCR        ;SELECT LINE 12
004446 012777 012060 005310      MOV    #NULL,@DHBA       ;SET UPT TO TRANSMIT 0 CHARACTER
004454 012777 177776 005304      MOV    #-2,@DHBC         ;TWO 0S WILL BE TRANSMITTED
004462 012777 033503 005272      MOV    #33503,@DHLPR     ;SET LINE SPEED=9600 BAUD
                                ;CHARACTER LENGTH =8 BITS
004470 012777 002000 005272      MOV    #2000,@DHBAR      ;SET BAR BIT FOR LINE 12
004476 122777 000002 005272 1$:   CMPB   #2,@DHSLR          ;WAIT FOR 2 CHARACTERS TO BE RECEIVED
004504 001374                BNE    1$
004506 012777 004000 005242      MOV    #BIT11,@DHSCP     ;MASTER CLEAR INTERFACE
004514 012777 000012 005234      MOV    #12,@DHSCR        ;SELECT LINE 12
004522 012777 012112 005234      MOV    #TBUF,@DHBA       ;SET UP TO TRANSMIT 400
004530 012777 177400 005230      MOV    #-400,@DHBC       ;(OCTAL) CHARACTERS
004536 012777 033503 005216      MOV    #33503,@DHLPR     ;LINE SPEED = 9600 BAUD
004544 012777 002000 005220      MOV    #2000,@DHBCR      ;SET BREAK BIT FOR LINE 12
004552 012777 002000 005210      MOV    #2000,@DHBAR      ;SET BAR BIT FOR LINE 12
004560 005777 005204 2$:   TST    @DHBAR            ;WAIT FOR ALL CHARACTERS
004564 001375                BNE    2$                ;TO BE TRANSMITTED
004566 122777 000001 005202      CMPB   #1,@DHSLR         ;CHECK TO SEE THAT ONLY
004574 001407                BEQ    3$                ;1 CHARACTER WAS RECEIVED
004576 017704 005172      MOV    @DHSSR,R4         ;(R4)=ACTUAL RECEIVED DATA
004602 042704 000300      BIC    #300,R4           ;CLEAR UNWANTED BITS
004606 012705 000400      MOV    #400,R5           ;(R5)=EXPECTED SILO FILL LEVEL, 1
004612                HLT    0                 ;MORE THAN ONE CHARACTER RECEIVED, ERROR
004612 104000                EMT    0
004614 017704 005140 3$:   MOV    @DHNRC,R4         ;READ NEXT RECEIVED CHARACTER REGISTER
004620 026704 005754      CMP    RWRD12,R4         ;IS RECEIVED CHARACTER A BREAK
004624 001403                BEQ    4$
004626 016705 005746      MOV    RWRD12,R5         ;(R5)=EXPECTED RECEIVED CHARACTER
004632                HLT    1                 ;RECEIVED DATA ERROR
004634 104001                EMT    1
004634 104400 4$:   SCOPE                ;CHECK FOR ITERATIONS, LOOP
                                LINE=LINE+1
                                BITX=BITX+BITX
                                KX=KX+1
004636      BREAK1 \LINE,\BITX,\KX

                                ;FLUSH UART BY TRANSMITTING 2 NULL CHARACTERS
                                ;ON LINE 13
                                ;SET BREAK BIT FOR LINE 13
                                ;TRANSMIT BINARY COUNT PATTERN ON LINE 13
                                ;VERIFY HTAT ONLY 1 CHARACTER IS RECEIVED
                                ;AND THAT IT IS A BREAK

004636      TS \XN,20,4$
004636 012767 000340 173132 T14:  MOV    #340,PS          ;DISABLE ALL INTERRUPTS
004644 012767 000020 005152      MOV    #20,ICOUNT        ;SET UP FOR 20 ITERATIONS
004652 012767 005066 005140      MOV    #4$,ESCAPE        ;SET UP TO ESCAPE TO NEXT TEST

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

004660 000015
004666 012777 004000 005070      MOV    #BIT11,@DHSCP     ;MASTER CLEAR INTERFACE
004666 004767 005172      JSR    PC,CLRALL          ;CLEAR ALL BUS ADDRESS AND

```

```

004672 012777 000013 005056      MOV      #13, @DHSCR      ;BUS ADDRESS MEMORY LOCATIONS
004700 012777 012060 005056      MOV      @NULL, @DHBA    ;SELECT LINE 13
004706 012777 177776 005052      MOV      # 2, @DHBC      ;SET UPT TO TRANSMIT 0 CHARACTER
004714 012777 033503 005040      MOV      @33503, @DHLPR  ;TWO OS WILL BE TRANSMITTED
                                ;SET LINE SPEED=9600 BAUD
                                ;CHARACTER LENGTH =8 BITS
004722 012777 004000 005040      MOV      #4000, @DHBAR    ;SET BAR BIT FOR LINE 13
004730 122777 000002 005040  1$:  CMPB     #2, @DHSLR      ;WAIT FOR 2 CHARACTERS TO BE RECEIVED
004736 001374                                BNE      1$
004740 012777 004000 005010      MOV      @BIT11, @DHSCR   ;MASTER CLEAR INTERFACE
004746 012777 000013 005002      MOV      #13, @DHSCR      ;SELECT LINE 13
004754 012777 012112 005002      MOV      @TBUF, @DHBA     ;SET UP TO TRANSMIT 400
004762 012777 177400 004776      MOV      #-400, @DHBC     ;(OCTAL) CHARACTERS
004770 012777 033503 004764      MOV      @33503, @DHLPR   ;LINE SPEED = 9600 BAUD
004776 012777 004000 004766      MOV      #4000, @DHBCR    ;SET BREAK BIT FOR LINE 13
005004 012777 004000 004756      MOV      #4000, @DHBAR    ;SET BAR BIT FOR LINE 13
005012 005777 004752 2$:  TST      @DHBAR          ;WAIT FOR ALL CHARACTERS
005016 001375                                BNE      2$              ;TO BE TRANSMITTED
005020 122777 000001 004750      CMPB     #1, @DHSLR      ;CHECK TO SEE THAT ONLY
005026 001407                                BEQ      3$              ;1 CHARACTER WAS RECEIVED
005030 017704 004740      MOV      @DHSSR, R4       ;(R4)=ACTUAL RECEIVED DATA
005034 042704 000300      BIC      #300, R4         ;CLEAR UNWANTED BITS
005040 012705 000400      MOV      #400, R5        ;(R5)=EXPECTED SILO FILL LEVEL, 1
005044                                HLT      0               ;MORE THAN ONE CHARACTER RECEIVED, ERROR
005044 104000                                EMT      0
005046 017704 004706 3$:  MOV      @DHNRC, R4       ;READ NEXT RECEIVED CHARACTER REGISTER
005052 026704 005524      CMP      RWRD13, R4       ;IS RECEIVED CHARACTER A BREAK
005056 001403                                BEQ      4$
005060 016705 005516      MOV      RWRD13, R5       ;(R5)=EXPECTED RECEIVED CHARACTER
005064                                HLT      1               ;RECEIVED DATA ERROR
005064 104001                                EMT      1
005066 104400 4$:  SCOPE          ;CHECK FOR ITERATIONS, LOOP
                                LINE=LINE+1
                                BITX=BITX+BITX
                                KX=KX+1
005070 000014      BREAK1  \LINE, \BITX, \KX
                                010000
                                000014

                                ;FLUSH UART BY TRANSMITTING 2 NULL CHARACTERS
                                ;ON LINE 14
                                ;SET BREAK BIT FOR LINE 14
                                ;TRANSMIT BINARY COUNT PATTERN ON LINE 14
                                ;VERIFY THAT ONLY 1 CHARACTER IS RECEIVED
                                ;AND THAT IT IS A BREAK

005070      TS \XN, 20, 4$
005070 012767 000340 172700  T15:  MOV      #340, PS        ;DISABLE ALL INTERRUPTS
005076 012767 000020 004720      MOV      #20, ICOUNT     ;SET UP FOR 20 ITERATIONS
005104 012767 005320 004706      MOV      #4$, ESCAPE     ;SET UP TO ESCAPE TO NEXT TEST
                                .IF NE <>
                                MOV      #, FREEZ1          ;SET UP TO LOOP WITH DATA      ; 3
                                .ENDC
                                XN=XN+1

005112 012777 004000 004636      MOV      @BIT11, @DHSCR   ;MASTER CLEAR INTERFACE
005120 004767 004740      JSR      PC, CLRALL       ;CLEAR ALL BUS ADDRESS AND
                                ;BUS ADDRESS MEMORY LOCATIONS
005124 012777 000014 004624      MOV      #14, @DHSCR      ;SELECT LINE 14
005132 012777 012060 004624      MOV      @NULL, @DHBA     ;SET UPT TO TRANSMIT 0 CHARACTER

```



```

005406 012777 020000 004354      MOV      #20000, @DHBAR      ;SET BAR BIT FOR LINE 15
005414 122777 000002 004354 1#:  CMPB     #2, @DHSLR      ;WAIT FOR 2 CHARACTERS TO BE RECEIVED
005422 001374                BNE      1#
005424 012777 004000 004324      MOV      @BIT11, @DHSCR     ;MASTER CLEAR INTERFACE
005432 012777 000015 004316      MOV      #15, @DHSCR     ;SELECT LINE 15
005440 012777 012112 004316      MOV      @TBUF, @DHBA     ;SET UP TO TRANSMIT 400
005446 012777 177400 004312      MOV      #-400, @DHBC     ;(OCTAL) CHARACTERS
005454 012777 033503 004300      MOV      #33503, @DHLPR   ;LINE SPEED = 9600 BAUD
005462 012777 020000 004302      MOV      #20000, @DHBCR   ;SET BREAK BIT FOR LINE 15
005470 012777 020000 004272      MOV      #20000, @DHBAR   ;SET BAR BIT FOR LINE 15
005476 005777 004266                2#:  TST      @DHBAR          ;WAIT FOR ALL CHARACTERS
005502 001375                BNE      2#              ;TO BE TRANSMITTED
005504 122777 000001 004264      CMPB     #1, @DHSLR      ;CHECK TO SEE THAT ONLY
005512 001407                BEQ      3#              ;1 CHARACTER WAS RECEIVED
005514 017704 004254                MOV      @DHSSR, R4      ;(R4)=ACTUAL RECEIVED DATA
005520 042704 000300                BIC      #300, R4        ;CLEAR UNWANTED BITS
005524 012705 000400                MOV      #400, R5        ;(R5)=EXPECTED SILO FILL LEVEL, 1
005530                HLT      0              ;MORE THAN ONE CHARACTER RECEIVED, ERROR
005530 104000                EMT      0
005532 017704 004222                3#:  MOV      @DHNRC, R4      ;READ NEXT RECEIVED CHARACTER REGISTER
005536 026704 005044                CMP      RWRD15, R4      ;IS RECEIVED CHARACTER A BREAK
005542 001403                BEQ      4#
005544 016705 005036                MOV      RWRD15, R5      ;(R5)=EXPECTED RECEIVED CHARACTER
005550                HLT      1              ;RECEIVED DATA ERROR
005550 104001                EMT      1
005552 104400                4#:  SCOPE
005552 104400                LINE=LINE+1
005554 000016                BITX=BITX+BITX
005554 040000                KX=KX+1
005554 000016                BREAK1 \LINE, \BITX, \KX

;FLUSH UART BY TRANSMITTING 2 NULL CHARACTERS
;ON LINE 16
;SET BREAK BIT FOR LINE 16
;TRANSMIT BINARY COUNT PATTERN ON LINE 16
;VERIFY THAT ONLY 1 CHARACTER IS RECEIVED
;AND THAT IT IS A BREAK

005554                TS \XN, 20, 4#
005554 012767 000340 172214 T17:  MOV      #340, PS        ;DISABLE ALL INTERRUPTS
005562 012767 000020 004234      MOV      #20, ICOUNT     ;SET UP FOR 20 ITERATIONS
005570 012767 006004 004222      MOV      #4#, ESCAPE     ;SET UP TO ESCAPE TO NEXT TEST

;IF NB <>
005570                MOV      #, FREEZ1      ;SET UP TO LOOP WITH DATA      ; 3

.ENDC
XN=XN+1

005576 012777 004000 004152      MOV      @BIT11, @DHSCR   ;MASTER CLEAR INTERFACE
005604 004767 004254      JSR      PC, CLRALL      ;CLEAR ALL BUS ADDRESS AND
;BUS ADDRESS MEMORY LOCATIONS
005610 012777 000016 004140      MOV      #16, @DHSCR     ;SELECT LINE 16
005616 012777 012060 004140      MOV      @NULL, @DHBA    ;SET UP TO TRANSMIT 0 CHARACTER
005624 012777 177776 004134      MOV      #-2, @DHBC     ;TWO OS WILL BE TRANSMITTED
005632 012777 033503 004122      MOV      #33503, @DHLPR  ;SET LINE SPEED=9600 BAUD
;CHARACTER LENGTH =8 BITS
005640 012777 040000 004122      MOV      #40000, @DHBAR  ;SET BAR BIT FOR LINE 16
005646 122777 000002 004122 1#:  CMPB     #2, @DHSLR      ;WAIT FOR 2 CHARACTERS TO BE RECEIVED
005654 001374                BNE      1#

```

```

005656 012777 004000 004072      MOV      @BIT11,@DHSCR      ;MASTER CLEAR INTERFACE
005664 012777 000016 004064      MOV      @16,@DHSCR        ;SELECT LINE 16
005672 012777 012112 004064      MOV      @TBUF,@DHBA       ;SET UP TO TRANSMIT 400
005700 012777 177400 004060      MOV      @-400,@DHBC       ;(OCTAL) CHARACTERS
005706 012777 033503 004046      MOV      @33503,@DHLP      ;LINE SPEED = 9600 BAUD
005714 012777 040000 004050      MOV      @40000,@DHBC      ;SET BREAK BIT FOR LINE 16
005722 012777 040000 004040      MOV      @40000,@DHBAR     ;SET BAR BIT FOR LINE 16
005730 005777 004034      2$: TST      @DHBAR          ;WAIT FOR ALL CHARACTERS
005734 001375      BNE      2$                ;TO BE TRANSMITTED
005736 122777 000001 004032      CMPB     @1,@DHSLR         ;CHECK TO SEE THAT ONLY
005744 001407      BEQ      3$                ;1 CHARACTER WAS RECEIVED
005746 017704 004022      MOV      @DHSSR,R4        ;(R4)=ACTUAL RECEIVED DATA
005752 042704 000300      BIC      @300,R4          ;CLEAR UNWANTED BITS
005756 012705 000400      MOV      @400,R5         ;(R5)=EXPECTED SILO FILL LEVEL, 1
005762      HLT      0                ;MORE THAN ONE CHARACTER RECEIVED, ERROR
005762 104000      EMT      0
005764 017704 003770      3$: MOV      @DHNRC,R4      ;READ NEXT RECEIVED CHARACTER REGISTER
005770 026704 004614      CMP      RWRD16,R4        ;IS RECEIVED CHARACTER A BREAK
005774 001403      BEQ      4$                ;
005776 016705 004606      MOV      RWRD16,R5        ;(R5)=EXPECTED RECEIVED CHARACTER
006002      HLT      1                ;RECEIVED DATA ERROR
006002 104001      EMT      1
006004 104400      4$: SCOPE                  ;CHECK FOR ITERATIONS, LOOP
      000017      LINE=LINE+1
      100000      BITX=BITX+BITX
      000017      KX=KX+1
006006      BREAK1 \LINE,\BITX,\KX

      ;FLUSH UART BY TRANSMITTING 2 NULL CHARACTERS
      ;ON LINE 17
      ;SET BREAK BIT FOR LINE 17
      ;TRANSMIT BINARY COUNT PATTERN ON LINE 17
      ;VERIFY THAT ONLY 1 CHARACTER IS RECEIVED
      ;AND THAT IT IS A BREAK

006006      TS \XN,20,4$
006006 012767 000340 171762      T20: MOV      @340,PS        ;DISABLE ALL INTERRUPTS
006014 012767 000020 004002      MOV      @20,ICOUNT       ;SET UP FOR 20 ITERATIONS
006022 012767 006236 003770      MOV      @4$,ESCAPE       ;SET UP TO ESCAPE TO NEXT TEST
      .IF NB <>
006022      MOV      @,FREEZ1      ;SET UP LOOP WITH DATA ; 3
      .ENDC
      XN=XN+1
006030 000021      MOV      @BIT11,@DHSCR    ;MASTER CLEAR INTERFACE
006036 004767 004022 003720      JSR      PC,CLRALL        ;CLEAR ALL BUS ADDRESS AND
      ;BUS ADDRESS MEMORY LOCATIONS
006042 012777 000017 003706      MOV      @17,@DHSCR       ;SELECT LINE 17
006050 012777 012060 003706      MOV      @NULL,@DHBA      ;SET UP TO TRANSMIT 0 CHARACTER
006056 012777 177776 003702      MOV      @-2,@DHBC        ;TWO 0S WILL BE TRANSMITTED
006064 012777 033503 003670      MOV      @33503,@DHLP     ;SET LINE SPEED=9600 BAUD
      ;CHARACTER LENGTH =8 BITS
006072 012777 100000 003670      MOV      @100000,@DHBAR   ;SET BAR BIT FOR LINE 17
006100 122777 000002 003670      1$: CMPB     @2,@DHSLR     ;WAIT FOR 2 CHARACTERS TO BE RECEIVED
006106 001374      BNE      1$
006110 012777 004000 003640      MOV      @BIT11,@DHSCR    ;MASTER CLEAR INTERFACE
006116 012777 000017 003632      MOV      @17,@DHSCR       ;SELECT LINE 17
006124 012777 012112 003632      MOV      @TBUF,@DHBA      ;SET UP TO TRANSMIT 400

```



```

006132 012777 177400 003626      MOV      #-400, @DHBC      ;(OCTAL) CHARACTERS
006140 012777 033503 003614      MOV      #33503, @DHLPR   ;LINE SPEED = 9600 BAUD
006146 012777 100C00 003616      MOV      #100000, @DHBCR  ;SET BREAK BIT FOR LINE 17
006154 012777 100000 003606      MOV      #100000, @DHBAR  ;SET BAR BIT FOR LINE 17
006162 005777 003602      2$:    TST      @DHBAR      ;WAIT FOR ALL CHARACTERS
006166 001375      BNE      2$              ;TO BE TRANSMITTED
006170 122777 030001 003600      CMPB     #1, @DHSLR      ;CHECK TO SEE THAT ONLY
006176 001407      BEQ      3$              ;1 CHARACTER WAS RECEIVED
006200 017704 003570      MOV      @DHSSR, R4      ;(R4)=ACTUAL RECEIVED DATA
006204 042704 000300      BIC      #300, R4        ;CLEAR UNWANTED BITS
006210 012705 000400      MOV      #400, R5        ;(R5)=EXPECTED SILO FILL LEVEL, 1
006214      HLT      0              ;MORE THAN ONE CHARACTER RECEIVED, ERROR
006214 104000      EMT      0
006216 017704 003536      3$:    MOV      @DHNR, R4    ;READ NEXT RECEIVED CHARACTER REGISTER
006222 026704 004364      CMP      RWORD17, R4     ;IS RECEIVED CHARACTER A BREAK
006226 001403      BEQ      4$              ;
006230 016705 004356      MOV      RWORD17, R5     ;(R5)=EXPECTED RECEIVED CHARACTER
006234      HLT      1              ;RECEIVED DATA ERROR
006234 104001      EMT      1
006236 1C4400      4$:    SCOPE
000020      LINE=LINE+1
000000      BITX=BITX+BITX
000020      KX=KX+1
18      000020      XLINE=LINE
19      000000      XBIT=BITX
20      000020      K=KX
21      000000      LINE=0
22      000001      BITX=1
23      000000      KX=0
25      000020      .REPT 20
26      BLIND1 \LINE, \BITX, \KX
27      .NLIST
28      LINE=LINE+1
29      BITX=BITX+BITX
30      KX=KX+1
31      .LIST
32      .ENDR
006240      BLIND1 \LINE, \BITX, \KX

;SET HALF DUPLEX ON LINE 0
;TRANSMIT A BINARY COUNT PATTERN
;VERIFY THAT NO CHARACTERS ARE RECEIVED

006240      TS \XN, 20, 2$
006240 012767 000340 171530      T21:    MOV      #340, PS    ;DISABLE ALL INTERRUPTS
006246 012767 000020 003550      MOV      #20, ICOUNT   ;SET UP FOR 20 ITERATIONS
006254 012767 006350 003536      MOV      #2$, ESCAPE    ;SET UP TO ESCAPE TO NEXT TEST
;IF NB
;MOV      #, FREEZ1      ;SET UP TO LOOP WITH DATA ; 3
.ENDC
XN=XN+1
006262 000022      MOV      #BIT11, @DHSCR  ;MASTER CLEAR INTERFACE
006270 004767 003570      JSR      PC, CLRALL     ;CLEAR ALL BYTE COUNT AND
;AND BUS ADDRESS MEMORY LOCATIONS
006274 012777 000000 003454      MOV      #0, @DHSCR     ;SELECT LINE 0
006302 012777 012112 003454      MOV      #TBUF, @DHBA   ;SET UP TO TRANSMIT
006310 012777 177400 003450      MOV      #-400, @DHBC   ;400 (OCTAL) CHARACTERS

```

```

006316 012777 073503 003436      MOV      #73503, @DHLPR      ;SET RECEIVER BLIND
                                           ;LINE SPEED =9600 BAUD
006324 012777 000001 003436      MOV      #1, @DHBAR        ;CHARACTER LENGTH = 8 BITS
006332 005777 003432      1$: TST      @DHBAR        ;SET BAR BIT FOR LINE 0
006336 001375                        BNE      1$                ;WAIT FOR ALL CHARACTERS TO BE TRANSMITTED
006340 105777 003412      TSTB     @DHSCR           ;WERE ANY CHARACTERS RECEIVED
006344 100001                        BPL      2$                ;RECEIVER NOT BLINDED, ERROR
006346                        HLT      2
006346 104002                        EMT      2
006350 104400      2$: SCOPE                ;CHECK FOR ITERATIONS, LOOP
      000001      LINE=LINE+1
      000002      BITX=BITX+BITX
      000001      KX=KX+1
006352                        BLIND1 \LINE, \BITX, \KX

                                           ;SET HALF DUPLEX ON LINE 1
                                           ;TRANSMIT A BINARY COUNT PATTERN
                                           ;VERIFY THAT NO CHARACTERS ARE RECEIVED

006352      TS \XN, 20, 2$
006352 012767 000340 171416      T22: MOV      #340, PS      ;DISABLE ALL INTERRUPTS
006360 012767 000020 003436      MOV      #20, ICOUNT      ;SET UP FOR 20 ITERATIONS
006366 012767 006462 003424      MOV      #2$, ESCAPE      ;SET UP TO ESCAPE TO NEXT TEST
                                           .IF NB <>
      MOV      #, FREEZ1      ;SET UP TO LOOP WITH DATA      ; 3
                                           .ENDC
      XN=XN+1
006374 012777 004000 003354      MOV      @BIT11, @DHSCR    ;MASTER CLEAR INTERFACE
006402 004767 003456      JSR      PC, CLRALL        ;CLEAR ALL BYTE COUNT AND
                                           ;AND BUS ADDRESS MEMORY LOCATIONS
006406 012777 000001 003342      MOV      #1, @DHSCR        ;SELECT LINE 1
006414 012777 012112 003342      MOV      @TBUF, @DHBA      ;SET UP TO TRANSMIT
006422 012777 177400 003336      MOV      #-400, @DHBC      ;400 (OCTAL) CHARACTERS
006430 012777 073503 003324      MOV      #73503, @DHLPR    ;SET RECEIVER BLIND
                                           ;LINE SPEED =9600 BAUD
                                           ;CHARACTER LENGTH = 8 BITS
006436 012777 000002 003324      1$: MOV      #2, @DHBAR        ;SET BAR BIT FOR LINE 1
006444 005777 003320      TST      @DHBAR        ;WAIT FOR ALL CHARACTERS TO BE TRANSMITTED
006450 001375                        BNE      1$                ;WERE ANY CHARACTERS RECEIVED
006452 105777 003300      TSTB     @DHSCR           ;RECEIVER NOT BLINDED, ERROR
006456 100001                        BPL      2$                ;CHECK FOR ITERATIONS, LOOP
006460                        HLT      2
006462 104002                        EMT      2
      000002      2$: SCOPE
      000004      LINE=LINE+1
      000002      BITX=BITX+BITX
      000001      KX=KX+1
006464                        BLIND1 \LINE, \BITX, \KX

                                           ;SET HALF DUPLEX ON LINE 2
                                           ;TRANSMIT A BINARY COUNT PATTERN
                                           ;VERIFY THAT NO CHARACTERS ARE RECEIVED

006464      TS \XN, 20, 2$
006464 012767 000340 171304      T23: MOV      #340, PS      ;DISABLE ALL INTERRUPTS
006472 012767 000020 003324      MOV      #20, ICOUNT      ;SET UP FOR 20 ITERATIONS

```

```

006500 012767 006574 003312      MOV      #2$,ESCAPE      ;SET UP TO ESCAPE TO NEXT TEST
      .IF NB <>
      MOV      #,FREEZ1      ;SE` UP TO LOOP WITH DATA      ; 3
      .ENDC
      XN=XN+1
006506 000024 012777 004000 003242      MOV      #BIT11,@DHSCR      ;MASTER CLEAR INTERFACE
006514 004767 003344      JSR      PC,CLRALL      ;CLEAR ALL BYTE COUNT AND
      ;AND BUS ADDRESS MEMORY LOCATIONS
006520 012777 000002 003230      MOV      #2,@DHSCR      ;SELECT LINE 2
006526 012777 012112 003230      MOV      #TBUF,@DHBA      ;SET UP TO TRANSMIT
006534 012777 177400 003224      MOV      #-400,@DHBC      ;400 (OCTAL) CHARACTERS
006542 012777 073503 003212      MOV      #73503,@DHLPR      ;SET RECEIVER BLIND
      ;LINE SPEED =9600 BAUD
      ;CHARACTER LENGTH = 8 BITS
006550 012777 000004 003212      MOV      #4,@DHBAR      ;SET BAR BIT FOR LINE 2
006556 005777 003206      1$: TST      @DHBAR      ;WAIT FOR ALL CHARACTERS TO BE TRANSMITTED
006562 001375      BNE      1$
006564 105777 003166      TSTB     @DHSCR      ;WERE ANY CHARACTERS RECEIVED
006570 100001      BPL      2$
006572      HLT      2
006572 104002      EMT      2
006574 104400      2$: SCOPE      ;CHECK FOR ITERATIONS, LOOP
      LINE=LINE+1
      BITX=BITX+BITX
      KX=KX+1
006576      BLIND1 \LINE,\BITX,\KX
      ;SET HALF DUPLEX ON LINE 3
      ;TRANSMIT A BINARY COUNT PATTERN
      ;VERIFY THAT NO CHARACTERS ARE RECEIVED

006576      TS \XN,20,2$
006576 012767 000340 171172      T24: MOV      #340,PS      ;DISABLE ALL INTERRUPTS
006604 012767 000020 003212      MOV      #20,ICOUNT      ;SET UP FOR 20 ITERATIONS
006612 012767 006706 003206      MOV      #2$,ESCAPE      ;SET UP TO ESCAPE TO NEXT TEST
      .IF NB <>
      MOV      #,FREEZ1      ;SET UP TO LOOP WITH DATA      . 3
      .ENDC
      XN=XN+1
006620 000025 012777 004000 003130      MOV      #BIT11,@DHSCR      ;MASTER CLEAR INTERFACE
006626 004767 003232      JSR      PC,CLRALL      ;CLEAR ALL BYTE COUNT AND
      ;AND BUS ADDRESS MEMORY LOCATIONS
006632 012777 000003 003116      MOV      #3,@DHSCR      ;SELECT LINE 3
006640 012777 012112 003116      MOV      #TBUF,@DHBA      ;SET UP TO TRANSMIT
006646 012777 177400 003112      MOV      #-400,@DHBC      ;400 (OCTAL) CHARACTERS
006654 012777 073503 003100      MOV      #73503,@DHLPR      ;SET RECEIVER BLIND
      ;LINE SPEED =9600 BAUD
      ;CHARACTER LENGTH = 8 BITS
006662 012777 000010 003100      MOV      #10,@DHBAR      ;SET BAR BIT FOR LINE 3
006670 005777 003074      1$: TST      @DHBAR      ;WAIT FOR ALL CHARACTERS TO BE TRANSMITTED
006674 001375      BNE      1$
006676 105777 003054      TSTB     @DHSCR      ;WERE ANY CHARACTERS RECEIVED
006702 1C0001      BPL      2$
006704      HLT      2
006704 104C02      EMT      2
006706 104400      2$: SCOPE      ;CHECK FOR ITERATIONS, LOOP
      LINE=LINE+1
      000004

```

```

000020          BITX=BITX+BITX
000004          KX=KX+1
006710          BLIND1 \LINE,\BITX,\KX

                  ;SET HALF DUPLEX ON LINE 4
                  ;TRANSMIT A BINARY COUNT PATTERN
                  ;VERIFY THAT NO CHARACTERS ARE RECEIVED

006710          TS \XN,20,2#
006710 012767 000340 171060 T25:  MOV  #340,PS          ;DISABLE ALL INTERRUPTS
006716 012767 000020 003100      MOV  #20,ICOUNT      ;SET UP FOR 20 ITERATIONS
006724 012767 007020 003066      MOV  #2#,ESCAPE     ;SET UP TO ESCAPE TO NEXT TEST

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

006732 000026 012777 004000 003016      MOV  #BIT11,@DHSCR   ;MASTER CLEAR INTERFACE
006740 004767 003120              JSR  PC,CLRALL       ;CLEAR ALL BYTE COUNT AND
                  ;AND BUS ADDRESS MEMORY LOCATIONS
006744 012777 000004 003004      MOV  #4,@DHSCR      ;SELECT LINE 4
006752 012777 012112 003004      MOV  #TBUF,@DHBA    ;SET UP TO TRANSMIT
006760 012777 177400 003000      MOV  #-400,@DHBC    ;400 (OCTAL) CHARACTERS
006766 012777 073503 002766      MOV  #73503,@DHLPR ;SET RECEIVER BLIND
                  ;LINE SPEED =9600 BAUD
                  ;CHARACTER LENGTH = 8 BITS
006774 012777 000020 002766      MOV  #20,@DHBAR     ;SET BAR BIT FOR LINE 4
007002 005777 002762      1#:  TST  @DHBAR          ;WAIT FOR ALL CHARACTERS TO BE TRANSMITTED
007006 001375              BNE  1#
007010 105777 002742              TSTB @DHSCR          ;WERE ANY CHARACTERS RECEIVED
007014 100001              BPL  2#
007016              MLT  2#          ;RECEIVER NOT BLINDED, ERROR
007016 104002              EMT  2#
007020 104400      2#:  SCOPE          ;CHECK FOR ITERATIONS, LOOP
000005          LINE=LINE+1
000040          BITX=BITX+BITX
000005          KX=KX+1
007022          BLIND1 \LINE,\BITX,\KX

                  ;SET HALF DUPLEX ON LINE 5
                  ;TRANSMIT A BINARY COUNT PATTERN
                  ;VERIFY THAT NO CHARACTERS ARE RECEIVED

007022          TS \XN,20,2#
007022 012767 000340 170746 T26:  MOV  #340,PS          ;DISABLE ALL INTERRUPTS
007030 012767 000020 002766      MOV  #20,ICOUNT      ;SET UP FOR 20 ITERATIONS
007036 012767 007132 002754      MOV  #2#,ESCAPE     ;SET UP TO ESCAPE TO NEXT TEST

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

007044 000027 012777 004000 002704      MOV  #BIT11,@DHSCR   ;MASTER CLEAR INTERFACE
007052 004767 003006              JSR  PC,CLRALL       ;CLEAR ALL BYTE COUNT AND
                  ;AND BUS ADDRESS MEMORY LOCATIONS
007056 012777 000005 002672      MOV  #5,@DHSCR      ;SELECT LINE 5
007064 012777 012112 002672      MOV  #TBUF,@DHBA    ;SET UP TO TRANSMIT
007072 012777 177400 002666      MOV  #-400,@DHBC    ;400 (OCTAL) CHARACTERS
007100 012777 073503 002654      MOV  #73503,@DHLPR ;SET RECEIVER BLIND

```

```

007106 012777 000040 002654      MOV    #40, @DHBAR
007114 005777 002650      1$:   TST    @DHBAR
007120 001375                BNE    1$
007122 105777 002630      TSTB   @DHSCR
007126 100001                BPL    2$
007130                HLT    2
007130 104002                EMT    2
007132 104400      2$:   SCOPE
000006      LINE=LINE+1
000100      BITX=BITX+BITX
000006      KX=KX+1
007134      BLIND1 \LINE, \BITX, \KX

;LINE SPEED =9600 BAUD
;CHARACTER LENGTH = 8 BITS
;SET BAR BIT FOR LINE 5
;WAIT FOR ALL CHARACTERS TO BE TRANSMITTED
;WERE ANY CHARACTERS RECEIVED
;RECEIVER NOT BLINDED, ERROR
;CHECK FOR ITERATIONS, LOOP

;SET HALF DUPLEX ON LTNE 6
;TRANSMITI A BINARY COUNT PATTERN
;VERIFY THAT NO CHARACTERS ARE RECEIVED

007134      TS \XN,20,2$
007134 012767 000340 170634 T27:  MOV    #340,PS
007142 012767 000020 002654      MOV    #20,ICOUNT
007150 012767 007244 002642      MOV    #2$,ESCAPE
;DISABLE ALL INTERRUPTS
;SET UP FOR 20 ITERATIONS
;SET UP TO ESCAPE TO NEXT TEST

;IF NB <>
MOV    #,FREEZ1
;SET UP TO LOOP WITH DATA ; 3

.ENDC
XN=XN+1

007156 000030
007164 012777 004000 002572      MOV    #BIT11,@DHSCR
004767 002674      JSR    PC,CLRALL
;MASTER CLEAR INTERFACE
;CLEAR ALL BYTE COUNT AND
;AND BUS ADDRESS MEMORY LOCATIONS

007170 012777 000006 002560      MOV    #6,@DHSCR
007176 012777 012112 002560      MOV    #TBUF,@DHBA
007204 012777 177400 002554      MOV    #-400,@DHBC
007212 012777 073503 002542      MOV    #73503,@DHLPR
;SELECT LINE 6
;SET UP TO TRANSMIT
;400 (OCTAL) CHARACTERS
;SET RECEIVER BLIND
;LINE SPEED =9600 BAUD
;CHARACTER LENGTH = 8 BITS

007220 012777 000100 002542      1$:   MOV    #100,@DHBAR
007226 005777 002536      TST    @DHBAR
007232 001375                BNC    1$
007234 105777 002516      TSTB   @DHSCR
007240 100001                BPL    2$
007242                HLT    2
007242 104002                EMT    2
007244 104400      2$:   SCOPE
000007      LINE=LINE+1
000200      BITX=BITX+BITX
000007      KX=KX+1
007246      BLIND1 \LINE, \BITX, \KX

;SET HALF DUPLEX ON LINE 7
;TRANSMITI A BINARY COUNT PATTERN
;VERIFY THAT NO CHARACTERS ARE RECEIVED

007246      TS \XN,20,2$
007246 012767 000340 170522 T30:  MOV    #340,PS
007254 012767 000020 002542      MOV    #20,ICOUNT
007262 012767 007356 002530      MOV    #2$,ESCAPE
;DISABLE ALL INTERRUPTS
;SET UP FOR 20 ITERATIONS
;SET UP TO ESCAPE TO NEXT TEST

```

```

      .IF NB <>
      MOV     #,FREEZ1           ;SET UP TO LOOP WITH DATA           ; 3
      .ENDC
      XN=XN+1
007270 000031 012777 004000 002460  MOV     #BIT11,@DHSCR      ;MASTER CLEAR INTERFACE
007276 004767 002562                JSR     PC,CLRALL          ;CLEAR ALL BYTE COUNT AND
                                ;AND BUS ADDRESS MEMORY LOCATIONS
007302 012777 000007 002446  MOV     #7,@DHSCR         ;SELECT LINE 7
007310 012777 012112 002446  MOV     #TBUF,@DHBA       ;SET UP TO TRANSMIT
007316 012777 177400 002442  MOV     #-400,@DHBC       ;400 (OCTAL) CHARACTERS
007324 012777 073503 002430  MOV     #73503,@DHLPR     ;SET RECEIVER BLIND
                                ;LINE SPEED =9600 BAUD
                                ;CHARACTER LENGTH = 8 BITS
007332 012777 000200 002430  MOV     #200,@DHBAR       ;SET BAR BIT FOR LINE 7
007340 005777 002424 1$:      TST     @DHBAR           ;WAIT FOR ALL CHARACTERS TO BE TRANSMITTED
007344 001375                BNE     1$
007346 105777 002404                TSTB   @DHSCR           ;WERE ANY CHARACTERS RECEIVED
007352 100001                BPL     2$
007354                HLT     2
007356 104002                EMT     2
                                ;RECEIVER NOT BLINDED, ERROR
                                ;CHECK FOR ITERATIONS, LOOP
                                2$:      SCOPE
                                LINE=LINE+1
                                BITX=BITX+BITX
                                KX=KX+1
007360 000010                BLIND1 \LINE,\BITX,\KX
                                ;SET HALF DUPLEX ON LINE 10
                                ;TRANSMIT A BINARY COUNT PATTERN
                                ;VERIFY THAT NO CHARACTERS ARE RECEIVED

007360                TS \XN,20,2$
007360 012767 000340 170410  T31:   MOV     #340,PS      ;DISABLE ALL INTERRUPTS
007366 012767 000020 002430  MOV     #20,ICOUNT       ;SET UP FOR 20 ITERATIONS
007374 012767 007470 002416  MOV     #2$,ESCAPE       ;SET UP TO ESCAPE TO NEXT TEST

      .IF NB <>
      MOV     #,FREEZ1           ;SET UP TO LOOP WITH DATA           ; 3
      .ENDC
      XN=XN+1
007402 000032 012777 004000 002346  MOV     #BIT11,@DHSCR      ;MASTER CLEAR INTERFACE
007410 004767 002450                JSR     PC,CLRALL          ;CLEAR ALL BYTE COUNT AND
                                ;AND BUS ADDRESS MEMORY LOCATIONS
007414 012777 000010 002334  MOV     #10,@DHSCR        ;SELECT LINE 10
007422 012777 012112 002334  MOV     #TBUF,@DHBA       ;SET UP TO TRANSMIT
007430 012777 177400 002330  MOV     #-400,@DHBC       ;400 (OCTAL) CHARACTERS
007436 012777 073503 002316  MOV     #73503,@DHLPR     ;SET RECEIVER BLIND
                                ;LINE SPEED =9600 BAUD
                                ;CHARACTER LENGTH = 8 BITS
007444 012777 000400 002316  MOV     #400,@DHBAR       ;SET BAR BIT FOR LINE 10
007452 005777 002312 1$:      TST     @DHBAR           ;WAIT FOR ALL CHARACTERS TO BE TRANSMITTED
007456 001375                BNE     1$
007460 105777 002272                TSTB   @DHSCR           ;WERE ANY CHARACTERS RECEIVED
007464 100001                BPL     2$
007466                HLT     2
007466 104002                EMT     2
                                ;RECEIVER NOT BLINDED, ERROR
                                ;CHECK FOR ITERATIONS, LOOP
                                2$:      SCOPE
                                LINE=LINE+1
                                BITX=BITX+BITX
007470 104400                BITX=BITX+BITX
000011
001000

```

```

007472 000011 KX=KX+1
BLIND1 \LINE,\BITX,\KX

;SET HALF DUPLEX ON LINE 11
;TRANSMIT A BINARY COUNT PATTERN
;VERIFY THAT NO CHARACTERS ARE RECEIVED

007472 TS \XN,20,2#
007472 012767 000340 170276 T32: MOV #340,PS ;DISABLE ALL INTERRUPTS
007500 012767 000020 002316 MOV #20,ICOUNT ;SET UP FOR 20 ITERATIONS
007506 012767 007602 002304 MOV #2#,ESCAPE ;SET UP TO ESCAPE TO NEXT TEST

;IF NB <>
MOV #,FREEZ1 ;SET UP TO LOOP WITH DATA ; 3

.ENDC
XN=XN+1

007514 000033
007522 012777 004000 002234 MOV #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
JSR PC,CLRALL ;CLEAR ALL BYTE COUNT AND
;AND BUS ADDRESS MEMORY LOCATIONS
007526 012777 000011 002222 MOV #11,@DHSCR ;SELECT LINE 11
007534 012777 012112 002222 MOV #TBUF,@DHBA ;SET UP TO TRANSMIT
007542 012777 177400 002216 MOV #-400,@DHBC ;400 (OCTAL) CHARACTERS
007550 012777 073503 002204 MOV #73503,@DHLPR ;SET RECEIVER BLIND
;LINE SPEED =9600 BAUD
;CHARACTER LENGTH = 8 BITS
;SET BAR BIT FOR LINE 11
007556 012777 001000 002204 MOV #1000,@DHBAR ;WAIT FOR ALL CHARACTERS TO BE TRANSMITTED
007564 005777 002200 1#: TST @DHBAR
007570 001375 BNE 1#
007572 105777 002160 TSTB @DHSCR ;WFRE ANY CHARACTERS RECEIVED
007576 100001 BPL 2#
007600 HLT 2 ;RECEIVER NOT BLINDED, ERROR
007600 104002 EMT 2
007602 104400 2#: SCOPE ;CHECK FOR ITERATIONS, LOOP
000012 LINE=LINE+1
002000 BITX=BITX+BITX
000012 KX=KX+1
007604 BLIND1 \LINE,\BITX,\KX

;SET HALF DUPLEX ON LINE 12
;TRANSMIT A BINARY COUNT PATTERN
;VERIFY THAT NO CHARACTERS ARE RECEIVED

007604 TS \XN,20,2#
007604 012767 000340 170164 T33: MOV #340,PS ;DISABLE ALL INTERRUPTS
007612 012767 000020 002204 MOV #20,ICOUNT ;SET UP FOR 20 ITERATIONS
007620 012767 007714 002172 MOV #2#,ESCAPE ;SET UP TO ESCAPE TO NEXT TEST

;IF NB <>
MOV #,FREEZ1 ;SET UP TO LOOP WITH DATA ; 3

.ENDC
XN=XN+1

007626 000034
007634 012777 004000 002122 MOV #BIT11,@DHSCR ;MASTER CLEAR INTERFACE
JSR PC,CLRALL ;CLEAR ALL BYTE COUNT AND
;AND BUS ADDRESS MEMORY LOCATIONS
007640 012777 000012 002110 MOV #12,@DHSCR ;SELECT LINE 12
007646 012777 012112 002110 MOV #TBUF,@DHBA ;SET UP TO TRANSMIT
007654 012777 177400 002104 MOV #-400,@DHBC ;400 (OCTAL) CHARACTERS
007662 012777 073503 002072 MOV #73503,@DHLPR ;SET RECEIVER BLIND
;LINE SPEED =9600 BAUD

```

```

007670 012777 002000 002072      MOV      #2000, @DHBAR      ;CHARACTER LENGTH = 8 BITS
007676 005777 002066      1$: TST      @DHBAR      ;SET BAR BIT FOR LINE 12
007702 001375      BNE      1$              ;WAIT FOR ALL CHARACTERS TO BE TRANSMITTED
007704 105777 002046      TSTB     @DHSCR          ;WERE ANY CHARACTERS RECEIVED
007710 100001      BPL      2$              ;RECEIVER NOT BLINDED, ERROR
007712      HLT      2              ;RECEIVER NOT BLINDED, ERROR
007712 104002      EMT      2              ;RECEIVER NOT BLINDED, ERROR
007714 104400      2$: SCOPE              ;CHECK FOR ITERATIONS, LOOP
      000013      LINE=LINE+1
      004000      BITX=BITX+BITX
      000013      KX=KX+1
007716      BLIND1 \LINE, \BITX, \KX

      ;SET HALF DUPLEX ON LINE 13
      ;TRANSMIT A BINARY COUNT PATTERN
      ;VERIFY THAT NO CHARACTERS ARE RECEIVED

007716      TS \XN, 20, 2$
007716 012767 000340 170052 T34: MOV      #340, PS      ;DISABLE ALL INTERRUPTS
007724 012767 000020 062072      MOV      #20, ICOUNT     ;SET UP FOR 20 ITERATIONS
007732 012767 010026 002060      MOV      #2$, ESCAPE    ;SET UP TO ESCAPE TO NEXT TEST

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

007740 012777 004000 002010      MOV      #BIT11, @DHSCR  ;MASTER CLEAR INTERFACE
007746 004767 002112      JSR      PC, CLRALL     ;CLEAR ALL BYTE COUNT AND
      ;AND BUS ADDRESS MEMORY LOCATIONS
007752 012777 000013 001776      MOV      #13, @DHSCR    ;SELECT LINE 13
007760 012777 012112 001776      MOV      @TBUF, @DHBA   ;SET UP TO TRANSMIT
007766 012777 177400 001772      MOV      #-400, @DHBC   ;400 (OCTAL) CHARACTERS
007774 012777 073503 001760      MOV      #73503, @DHLPR ;SET RECEIVER BLIND
      ;LINE SPEED =9600 BAUD
      ;CHARACTER LENGTH = 8 BITS
010002 012777 004000 001760      MOV      #4000, @DHBAR  ;SET BAR BIT FOR LINE 13
010010 005777 001754      1$: TST      @DHBAR      ;WAIT FOR ALL CHARACTERS TO BE TRANSMITTED
010014 001375      BNE      1$              ;WERE ANY CHARACTERS RECEIVED
010016 105777 001734      TSTB     @DHSCR          ;WERE ANY CHARACTERS RECEIVED
010022 100001      BPL      2$              ;RECEIVER NOT BLINDED, ERROR
010024      HLT      2              ;RECEIVER NOT BLINDED, ERROR
010024 104002      EMT      2              ;RECEIVER NOT BLINDED, ERROR
010026 104400      2$: SCOPE              ;CHECK FOR ITERATIONS, LOOP
      000014      LINE=LINE+1
      010000      BITX=BITX+BITX
      000014      KX=KX+1
010030      BLIND1 \LINE, \BITX, \KX

      ;SET HALF DUPLEX ON LINE 14
      ;TRANSMIT A BINARY COUNT PATTERN
      ;VERIFY THAT NO CHARACTERS ARE RECEIVED

010030      TS \XN, 20, 2$
010030 012767 000340 167740 T35: MOV      #340, PS      ;DISABLE ALL INTERRUPTS
010036 012767 000020 001760      MOV      #20, ICOUNT     ;SET UP FOR 20 ITERATIONS
010044 012767 010140 001746      MOV      #2$, ESCAPE    ;SET UP TO ESCAPE TO NEXT TEST

      .IF NB <>

```



```

                                MOV     #,FREEZ1                ;SET UP TO LOOP WITH DATA      ; 3
                                .ENDC
                                XN=XN+1
010052 012777 004000 001676      MOV     #BIT11,@DHSCR      ;MASTER CLEAR INTERFACE
010060 004767 002000                JSR     PC,CLRALL          ;CLEAR ALL BYTE COUNT AND
                                ;AND BUS ADDRESS MEMORY LOCATIONS
010064 012777 000014 001664      MOV     #14,@DHSCR        ;SELECT LINE 14
010072 012777 012112 001664      MOV     #TBUF,@DHBA       ;SET UP TO TRANSMIT
010100 012777 177400 001660      MOV     #-400,@DHBC       ;400 (OCTAL) CHARACTERS
010106 012777 073503 001646      MOV     #73503,@DHLPR     ;SET RECEIVER BLIND
                                ;LINE SPEED =9600 BAUD
                                ;CHARACTER LENGTH = 8 BITS
010114 012777 010000 001646      MOV     #10000,@DHBAR     ;SET BAR BIT FOR LINE 14
010122 005777 001642      1$:   TST     @DHBAR         ;WAIT FOR ALL CHARACTERS TO BE TRANSMITTED
010126 001375                BNE     1$
010130 105777 001622      TSTB   @DHSCR             ;WERE ANY CHARACTERS RECEIVED
010134 100001                BPL     2$
010136                HLT     2
010136 104002                EMT     2
010140 104400      2$:   SCOPE          ;CHECK FOR ITERATIONS, LOOP
                                LINE=LINE+1
                                BITX=BITX+BITX
                                KX=KX+1
010142      BLIND1 \LINE,\BITX,\KX

                                ;SET HALF DUPLEX ON LINE 15
                                ;TRANSMIT A BINARY COUNT PATTERN
                                ;VERIFY THAT NO CHARACTERS ARE RECEIVED

010142      TS \XN,20,2$
010142 012767 000340 167626      T36:   MOV     #340,PS      ;DISABLE ALL INTERRUPTS
010150 012767 000020 001646      MOV     #20,ICOUNT       ;SET UP FOR 20 ITERATIONS
010156 012767 010252 001634      MOV     #2$,ESCAPE       ;SET UP TO ESCAPE TO NEXT TEST
                                .IF NB <>
                                MOV     #,FREEZ1                ;SET UP TO LOOP WITH DATA      ; 3
                                .ENDC
                                XN=XN+1
010164 012777 004000 001564      MOV     #BIT11,@DHSCR      ;MASTER CLEAR INTERFACE
010172 004767 001666      JSR     PC,CLRALL          ;CLEAR ALL BYTE COUNT AND
                                ;AND BUS ADDRESS MEMORY LOCATIONS
010176 012777 000015 001552      MOV     #15,@DHSCR        ;SELECT LINE 15
010204 012777 012112 001552      MOV     #TBUF,@DHBA       ;SET UP TO TRANSMIT
010212 012777 177400 001546      MOV     #-400,@DHBC       ;400 (OCTAL) CHARACTERS
010220 012777 073503 001534      MOV     #73503,@DHLPR     ;SET RECEIVER BLIND
                                ;LINE SPEED =9600 BAUD
                                ;CHARACTER LENGTH = 8 BITS
010226 012777 020000 001534      MOV     #20000,@DHBAR     ;SET BAR BIT FOR LINE 15
010234 005777 001530      1$:   TST     @DHBAR         ;WAIT FOR ALL CHARACTERS TO BE TRANSMITTED
010240 001375                BNE     1$
010242 105777 001510      TSTB   @DHSCR             ;WERE ANY CHARACTERS RECEIVED
010246 100001                BPL     2$
010250                HLT     2
010250 104002                EMT     2
010252 104400      2$:   SCOPE          ;CHECK FOR ITERATIONS, LOOP
                                LINE=LINE+1
                                BITX=BITX+BITX
                                KX=KX+1
010252 000016
                                040000
                                000016

```



```
010452 012777 100000 001310      MOV      #100000,@DHBAR      ;SET BAR BIT FOR LINE 17
010460 005777 001304      1$:      TST      @DHBAR      ;WAIT FOR ALL CHARACTERS TO BE TRANSMITTED
010464 001375      BNE      1$
010466 105777 001264      TSTB     @DHSCR      ;WERE ANY CHARACTERS RECEIVED
010472 100001      BPL      2$
010474      HLT      2      ;RECEIVER NOT BLINDED, ERROR
010474 104002      EMT      2
010476 104400      2$:      SCOPE
000020      LINE=LINE+1
000000      BITX=BITX+BITX
000020      KX=KX+1
;CHECK FOR ITERATIONS, LOOP
```

1  
2 010500

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

010500	104401			EOP:	TYPE		;TYPE NAME OF TEST	
010502	013204				MEPASS			
010504	005067	001344			CLR	LAST	;CLEAR LAST ERROR PC	
010510	005067	001274			CLR	ERRFLG	;CLEAR ERROR FLAG	
010514	005267	001272			INC	PASCNT	;UPDATE PASS COUNT	
010520	005767	170256			TST	LIGHTS	; ARE WE USING LIGHTS?	: 4
010524	001005				BNE	2#	; BRANCH IF WE ARE	: 6
010526	104401				TYPE		; TYPE PASCOUNT MESSAGE	: 5
010530	013217				PASTXT			: 5
010532	104402				OCTASC		; PRINT PASSCOUNT	: 4
010534	010572				PASARG			: 4
010536	000403				BR	3#	; CONTINUE	: 4
010540				2#:				: 4
010540	016767	001246	170234		MOV	PASCNT,LIGHTS	;DISPLAY PASS COUNT	: 4
010546				3#:				: 4
010546	013701	000042			MOV	B#42,R1	;CHECK FOR ACT-11 OR DDP	
010552	001405				BEQ	RESTR	;IF NOT, CONTINUE TESTING	
010554	000005				RESET			
010556	004711			LOGICAL:	JSR	PC,(R1)		
010560	000240				NOP			
010562	000240				NOP			
010564	000240				NOP			
010566	000167	170514		RESTR:	JMP	BEGIN		
010572	000001			PASARG:	.WORD	1	; PARAMETERS TO PRINT PASSCOUNT	: 5
010574	006	002			.BYTE	6,2		: 5
010576	012012				.WORD	PASCNT		: 5

3 010600

.SCOPE  
;CHECK FOR LOOP ON CURRENT TEST  
;CHECK FOR ITERATION SUPPRESSION

010600	032777	002000	170172	SCOPER:	BIT	#SW10,BSMR		: 4
010606	001030				BNE	4#		: 4
010610	032777	040000	170162	1#:	BIT	#SW14,BSMR		: 4
010616	001021				BNE	3#		: 4
010620	032777	004000	170152		BIT	#SW11,BSMR		: 4
010626	001006				BNE	2#		
010630	005267	001172			INC	LPCNT		
010634	026767	001166	001162		CMP	LPCNT,ICOUNT		
010642	001007				BNE	3#		
010644	005067	001156		2#:	CLR	LPCNT		
010650	005067	001134			CLR	ERRFLG		
010654	011667	001136			MOV	(SP),RETRN		
010660	000002				RTI			
010662	016716	001130		3#:	MOV	RETRN,(SP)		
010666	000002				RTI			
010670	005767	001114		4#:	TST	ERRFLG		
010674	001745				BEQ	1#		

010676 000762  
4 010700

BR      2#  
.SCOP1

;CHECK FOR FREEZE ON CURRENT DATA

010700 032777 001000 170072  
010706 001402  
010710 016716 001106  
010714 000002

SCOP1R: BIT      #SW09,BSWR  
          BEQ      1#  
          MOV      FREEZ1,(SP)  
1# :      RTI

; 4

1 010716

.ERROR

;ERROR HANDLER

```

010716 032777 020000 170054 ERRORS: BIT #SW13,@SWR ; 4
010724 001055 BNE HALTS
010726 021667 001122 CMP (SP),LAST
010732 001404 BEQ 1#
010734 011667 001114 MOV (SP),LAST
010740 005067 001044 CLR ERRFLG
010744 104406 1# : SAVO5P
010746 011605 MOV (SP),R5
010750 162705 000002 SUB #2,R5
010754 011504 MOV (R5),R4
010756 006304 ASL R4
010760 006304 ASL R4
010762 042704 177001 BIC #177001,R4
010766 062704 013336 ADD #ERRTAB,R4
010772 012467 000040 MOV (R4)+,ERRMSG
010776 011467 000052 MOV (R4),DATABP
011002 005767 001002 TST ERRFLG
011006 001403 BEQ TYPMSG
011010 005767 000040 TST DATABP
011014 001011 BNE TYPDAT ; 3
011016 104401 TYPMSG: TYPE ; 5
011020 013114 MCRLF ; 5
011022 104402 OCTASC ; 5
011024 011122 ERTABO
011026 012767 000001 000754 MOV #1,ERRFLG
011034 104401 TYPE
011036 000000 ERRMSG: 0
011040 005767 000010 TYPDAT: TST DATABP
011044 001404 BEQ RESREG
011046 104401 TYPE ; 5
011050 013114 MCRLF ; 5
011052 104402 OCTASC
011054 000000 DATABP: 0
011056 104407 RESREG: RES05
011060 005777 167714 HALTS: TST @SWR ; 4
011064 100005 BPL EXITER
011066 010046 PUSHRO
011070 016600 000002 MOV 2(SP),R0
011074 000000 HALT
011076 012600 POPRO
011100 005267 000710 EXITER: INC ERRCNT
011104 032777 002000 167666 BIT #SW10,@SWR ; 4
011112 001402 BEQ 1#
011114 016716 000700 MOV ESCAPE,(SP)
011120 000002 1# : RTI
011122 000001 ERTABO: 1
011124 006 002 .BYTE 6.2
011126 012046 SAVPC
    
```

```

011130          .TRPSRV
                ;TRAP DISPATCH SERVICE
                ;ARGUMENT OF TRAP IS EXTRACTED
                ;AND USED AS OFFSET TO OBTAIN POINTER
                ;TO SELECTED SUBROUTINE
                ; 3

011130 011646          TRPSRV: MOV      (SP),-(SP)          ;GET PC OF RETURN
011132 162716 000002    SUB      #2,(SP)              ;=-PC OF TRAP
011136 017616 000000    MOV      @((SP)),(SP)          ;GET TRP
011142 006316          TRPOK: ASL      (SP)              ;MULTIPLY TRAP ARG BY 2
011144 042716 177001    BIC      #177001,(SP)          ;CLEAR UNWANTED BITS
011150 062716 013256    ADD      #TRPTAB,(SP)          ;POINTER TO SUBROUTINE ADDRESS
011154 017616 000000    MOV      @((SP)),(SP)          ;SUBROUTINE ADDRESS
011160 000136          JMP      @((SP)+)          ;GO TO SUBROUTINE
2 011162          .SAVREG
                ;SAVE PC OF TEST THAT FAILED AND R0-R5

011162 016667 000004 000656 SV05P: MOV      4(SP),SAVPC
                ;SAVE R0-R5

011170 010567 000646    SV05:  MOV      R5,SAVR5
011174 010467 000640    MOV      R4,SAVR4
011200 010367 000632    MOV      R3,SAVR3
011204 010267 000624    MOV      R2,SAVR2
011210 010167 000616    MOV      R1,SAVR1
011214 010067 000610    MOV      R0,SAVR0
011220 000002          RTI
3 011222          .RESREG
                ;RESTORE R0-R5

011222 016700 000602    RS05:  MOV      SAVR0,R0
011226 016701 000600    MOV      SAVR1,R1
011232 016702 000576    MOV      SAVR2,R2
011236 016703 000574    MOV      SAVR3,R3
011242 016704 000572    MOV      SAVR4,R4
011246 016705 000570    MOV      SAVR5,R5
011252 000002          RTI

```

1 011254

.TYPER

;TELETYPE OUTPUT ROUTINE

011254 017605 000000  
 011260 062716 000002  
 011264 105777 000462  
 011270 100375  
 011272 105715  
 011274 001001  
 011276 000002  
 011300 112577 000450  
 011304 000767  
 2 011306

TYPER: MOV @ (SP),P5  
 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

011306 017667 000000 000006  
 011314 062716 000002  
 011320 104401  
 011322 000000  
 011324 012704 013300  
 011330 012703 000007  
 011334 105777 000406  
 011340 100375  
 011342 117714 000402  
 011346 142714 000200  
 011352 122427 000015  
 011356 001413  
 011360 117777 000364 000366  
 011366 105777 000360  
 011372 100375  
 011374 005303  
 011376 001356  
 011400 104401  
 011402 013110  
 011404 000745  
 011406 000002

INSTRG: MOV @ (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



1 011410

.PARAMS

;CONVERT ASCII STRING TO OCTAL

; 3

011410 011605  
 011412 012567 000146  
 011416 012567 000144  
 011422 012567 000142  
 011426 112567 000140  
 011432 112567 000135  
 011436 010516  
 011440 005005  
 011442 012704 013300  
 011446 122714 000015  
 011452 001420  
 011454 121427 000060  
 011460 002415  
 011462 121427 000067  
 011466 003012  
 011470 142714 000060  
 011474 152405  
 011476 122714 000015  
 011502 001406  
 011504 006305  
 011506 006305  
 011510 006305  
 011512 000760  
 011514 104404  
 011516 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 #INGUF,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

011520 020567 000042  
 011524 101373  
 011526 020567 000032  
 011532 103770  
 011534 136705 000032  
 011540 001365

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

; 3

;STORE NUMBER AT SPECIFIED ADDRESS

011542 016704 000022  
 011546 010524  
 011550 062705 000002  
 011554 105367 000013  
 011560 001372  
 011562 000002  
 011564 000000  
 011566 000000  
 011570 000000  
 011572 000000  
 011573

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

011574

.OCTASC

;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER

```

011574 017601 000000
011600 062716 000002
011604 012167 000130
011610 112167 000126
011614 112167 000123
011620 013167 000120
011624 016704 000114
011630 116705 000106
011634 012700 013312
011640 010403
011642 042703 177770
011645 062703 000260
011652 110320
011654 006204
011656 006204
011660 006204
011662 005305
011664 001365
011666 012703 013324
011672 114023
011674 105367 000042
011700 001374
011702 105767 000035
011706 001405
011710 112723 000240
011714 105367 000023
011720 001373
011722 105013
011724 104401
011726 013324
011730 005367 000004
011734 001325
011736 000002
011740 000000
011742 000000
011743 011743
011744 000000

```

```

OCTASN: MOV      6(SP),R1
          ADD      @2,(SP)
          MOV      (R1)+,WRDCNT
1$:      MOV      (R1)+,CHRCNT
          MOV      (R1),SPACNT
          MOV      @8(R1)+,BINWRD
2$:      MOV      BINWRD,R4
          MOV      CHRCNT,R5
          MOV      @TEMP,R0
3$:      MOV      R4,R3
          BIC      @177770,R3
          ADD      @260,R3
          MOV      R3,(R0)+
          ASR      R4
          ASR      R4
          ASR      R4
          DEC      R5
          BNE      3$
          MOV      @MDATA,R3
4$:      MOV      -(R0),(R3)+
          DECB    CHRCNT
          BNE      4$
          TSTB    SPACNT
          BEQ      6$
5$:      MOV      @240,(R3)+
          DECB    SPACNT
          BNE      5$
6$:      CLRB    (R3)
          TYPE    MDATA
          DEC      WRDCNT
          BNE      1$
          RTI
WRDCNT: 0
CHRCNT: 0
SPACNT=CHRCNT+1
BINWRD: 0

```

; 5

; 3

```

011746          .POINT  +/DHSCR,DHNRC,DHLPR,DHBA,DHBC,DHBAR,DHBCR,DHSSR,DHSLR,DHRVEC,DHRLVL,DHTVEC,DHTLVL/
                  ;INDIRECT POINTERS                               ; 3

011746 177560    TKCSR:  177560
011750 177562    TKOBR:  177562
011752 177564    TPCSR:  177564
011754 177566    TPDBR:  177566
TLVL>          .IRP   A      <DHSCR,DHNRC,DHLPR,DHBA,DHBC,DHBAR,DHBCR,DHSSR,DHSLR,DHRVEC,DHRLVL,DHTVEC,DH
                  A:      0
                  .ENDM
011756 000000    DHSCR:  0
011760 000000    DHNRC:  0
011762 000000    DHLPR:  0
011764 000000    DHBA:  0
011766 000000    DHBC:  0
011770 000000    DHBAR:  0
011772 000000    DHBCR:  0
011774 000000    DHSSR:  0
011776 000000    DHSLR:  0
012000 000000    DHRVEC:  0
012002 000000    DHRLVL:  0
012004 000000    DHTVEC:  0
012006 000000    DHTLVL:  0
2 012010        .VARI  +/ENDFLG,NULL,LINACT/
                  ;PROGRAM VARIABLES

012010 000000    ERRFLG:  0          ;ERROR FLAG
012012 000000    PASCNT:  0          ;PASS COUNT
012014 000000    ERRCNT:  0          ;ERROR COUNT
012016 000000    RETRN:  0          ;SCOPE RETURN ADDRESS FOR TEST LOOPING
012020 000000    ESCAPE:  0          ;ADDRESS FOR ERROR ESCAPE
012022 000000    FREEZ1:  0          ;DATA LOOPING RETURN ADDRESS
012024 000000    ICOUNT:  0          ;ITERATION COUNT FOR TEST IN PROGRESS
012026 000000    LPCNT:  0          ;NUMBER OF ITERATIONS THIS TEST
012030 000000    SAVRO:  0          ;R0 SAVE AREA
012032 000000    SAVR1:  0          ;R1 SAVE AREA
012034 000000    SAVR2:  0          ;R2 SAVE AREA
012036 000000    SAVR3:  0          ;R3 SAVE ARE
012040 000000    SAVR4:  0          ;R4 SAVE AREA
012042 000000    SAVR5:  0          ;R5 SAVE AREA
012044 000000    SAVSP:  0          ;STACK POINTER SAVE AREA
012046 000000    SAVPC:  0          ;CALLING ROUTINE SAVE AREA
012050 000000    INIFLG:  0          ;PROGRAM INITIALIZATION FLAG
012052 000000    STFLG:  0          ;PROGRAM START FLAG
012054 000000    LAST:  0          ;LAST ERROR PC
                  .IRP   A      <ENDFLG,NULL,LINACT>
                  A:      0
                  .ENDM
012056 000000    ENDFLG:  0
012060 000000    NULL:  0
012062 000000    LINACT:  0
    
```

: 3

1 012064

CCLRALL

;CLEAR ALL BYTE COUNT AND BUS ADDRESS REGISTERS

012064	012700	000020	CLRALL: MOV	#20,RO
012070	005077	177670	1\$: CLR	@DHBA
012074	005077	177666	CLR	@DHBC
012100	005277	177652	INC	@DHSCR
012104	005300		DEC	RO
012106	001370		BNE	1\$
012110	000207		RTS	PC
3	000001		TDAT=1	
5	012112	000	TBIF: .BYTE	0
6	000377		.REPT	377
7			.BYTE	TDAT
8			.NLIST	
9			TDAT=TDAT+1	
10			.LIST	
11			.ENDR	
012113	001		.BYTE	TDAT
	000002		TDAT=TDAT+1	
012114	002		.BYTE	TDAT
	000003		TDAT=TDAT+1	
012115	003		.BYTE	TDAT
	000004		TDAT=TDAT+1	
012116	004		.BYTE	TDAT
	000005		TDAT=TDAT+1	
012117	005		.BYTE	TDAT
	000006		TDAT=TDAT+1	
012120	006		.BYTE	TDAT
	000007		TDAT=TDAT+1	
012121	007		.BYTE	TDAT
	000010		TDAT=TDAT+1	
012122	010		.BYTE	TDAT
	000011		TDAT=TDAT+1	
012123	011		.BYTE	TDAT
	000012		TDAT=TDAT+1	
012124	012		.BYTE	TDAT
	000013		TDAT=TDAT+1	
012125	013		.BYTE	TDAT
	000014		TDAT=TDAT+1	
012126	014		.BYTE	TDAT
	000015		TDAT=TDAT+1	
012127	015		.BYTE	TDAT
	000016		TDAT=TDAT+1	
012130	016		.BYTE	TDAT
	000017		TDAT=TDAT+1	
012131	017		.BYTE	TDAT
	000020		TDAT=TDAT+1	
012132	020		.BYTE	TDAT
	000021		TDAT=TDAT+1	
012133	021		.BYTE	TDAT
	000022		TDAT=TDAT+1	
012134	022		.BYTE	TDAT
	000023		TDAT=TDAT+1	
012135	023		.BYTE	TDAT
	000024		TDAT=TDAT+1	

012136	024	.BYTE TDAT
	000025	TDAT=TDAT+1
012137	025	.BYTE TDAT
	000026	TDAT=TDAT+1
012140	026	.BYTE TDAT
	000027	TDAT=TDAT+1
012141	027	.BYTE TDAT
	000030	TDAT=TDAT+1
012142	030	.BYTE TDAT
	000031	TDAT=TDAT+1
012143	031	.BYTE TDAT
	000032	TDAT=TDAT+1
012144	032	.BYTE TDAT
	000033	TDAT=TDAT+1
012145	033	.BYTE TDAT
	000034	TDAT=TDAT+1
012146	034	.BYTE TDAT
	000035	TDAT=TDAT+1
012147	035	.BYTE TDAT
	000036	TDAT=TDAT+1
012150	036	.BYTE TDAT
	000037	TDAT=TDAT+1
012151	037	.BYTE TDAT
	000040	TDAT=TDAT+1
012152	040	.BYTE TDAT
	000041	TDAT=TDAT+1
012153	041	.BYTE TDAT
	000042	TDAT=TDAT+1
012154	042	.BYTE TDAT
	000043	TDAT=TDAT+1
012155	043	.BYTE TDAT
	000044	TDAT=TDAT+1
012156	044	.BYTE TDAT
	000045	TDAT=TDAT+1
012157	045	.BYTE TDAT
	000045	TDAT=TDAT+1
012160	046	.BYTE TDAT
	000047	TDAT=TDAT+1
012161	047	.BYTE TDAT
	000050	TDAT=TDAT+1
012162	050	.BYTE TDAT
	000051	TDAT=TDAT+1
012163	051	.BYTE TDAT
	000052	TDAT=TDAT+1
012164	052	.BYTE TDAT
	000053	TDAT=TDAT+1
012165	053	.BYTE TDAT
	000054	TDAT=TDAT+1
012166	054	.BYTE TDAT
	000055	TDAT=TDAT+1
012167	055	.BYTE TDAT
	000056	TDAT=TDAT+1
012170	056	.BYTE TDAT
	000057	TDAT=TDAT+1
012171	057	.BYTE TDAT
	000060	TDAT=TDAT+1
012172	060	.BYTE TDAT

012173	000061	061	TDAT=TDAT+1
			.BYTE TDAT
012174	000062	062	TDAT=TDAT+1
			.BYTE TDAT
012175	000063	063	TDAT=TDAT+1
			.BYTE TDAT
012176	000064	064	TDAT=TDAT+1
			.BYTE TDAT
012177	000065	065	TDAT=TDAT+1
			.BYTE TDAT
012200	000066	066	TDAT=TDAT+1
			.BYTE TDAT
012201	000067	067	TDAT=TDAT+1
			.BYTE TDAT
012202	000070	070	TDAT=TDAT+1
			.BYTE TDAT
012203	000071	071	TDAT=TDAT+1
			.BYTE TDAT
012204	000072	072	TDAT=TDAT+1
			.BYTE TDAT
012205	000073	073	TDAT=TDAT+1
			.BYTE TDAT
012206	000074	074	TDAT=TDAT+1
			.BYTE TDAT
012207	000075	075	TDAT=TDAT+1
			.BYTE TDAT
012210	000076	076	TDAT=TDAT+1
			.BYTE TDAT
012211	000077	077	TDAT=TDAT+1
			.BYTE TDAT
012212	000100	100	TDAT=TDAT+1
			.BYTE TDAT
012213	000101	101	TDAT=TDAT+1
			.BYTE TDAT
012214	000102	102	TDAT=TDAT+1
			.BYTE TDAT
012215	000103	103	TDAT=TDAT+1
			.BYTE TDAT
012216	000104	104	TDAT=TDAT+1
			.BYTE TDAT
012217	000105	105	TDAT=TDAT+1
			.BYTE TDAT
012220	000106	106	TDAT=TDAT+1
			.BYTE TDAT
012221	000107	107	TDAT=TDAT+1
			.BYTE TDAT
012222	000110	110	TDAT=TDAT+1
			.BYTE TDAT
012223	000111	111	TDAT=TDAT+1
			.BYTE TDAT
012224	000112	112	TDAT=TDAT+1
			.BYTE TDAT
012225	000113	113	TDAT=TDAT+1
			.BYTE TDAT
012226	000114	114	TDAT=TDAT+1
			.BYTE TDAT
	000115	115	TDAT=TDAT+1

012227	115	.BYTE TDAT
	000116	TDAT=TDAT+1
012230	116	.BYTE TDAT
	000117	TDAT=TDAT+1
012231	117	.BYTE TDAT
	000120	TDAT=TDAT+1
012232	120	.BYTE TDAT
	000121	TDAT=TDAT+1
012233	121	.BYTE TDAT
	000122	TDAT=TDAT+1
012234	122	.BYTE TDAT
	000123	TDAT=TDAT+1
012235	123	.BYTE TDAT
	000124	TDAT=TDAT+1
012236	124	.BYTE TDAT
	000125	TDAT=TDAT+1
012237	125	.BYTE TDAT
	000126	TDAT=TDAT+1
012240	126	.BYTE TDAT
	000127	TDAT=TDAT+1
012241	127	.BYTE TDAT
	000130	TDAT=TDAT+1
012242	130	.BYTE TDAT
	000131	TDAT=TDAT+1
012243	131	.BYTE TDAT
	000132	TDAT=TDAT+1
012244	132	.BYTE TDAT
	000133	TDAT=TDAT+1
012245	133	.BYTE TDAT
	000134	TDAT=TDAT+1
012246	134	.BYTE TDAT
	000135	TDAT=TDAT+1
012247	135	.BYTE TDAT
	000136	TDAT=TDAT+1
012250	136	.BYTE TDAT
	000137	TDAT=TDAT+1
012251	137	.BYTE TDAT
	000140	TDAT=TDAT+1
012252	140	.BYTE TDAT
	000141	TDAT=TDAT+1
012253	141	.BYTE TDAT
	000142	TDAT=TDAT+1
012254	142	.BYTE TDAT
	000143	TDAT=TDAT+1
012255	143	.BYTE TDAT
	000144	TDAT=TDAT+1
012256	144	.BYTE TDAT
	000145	TDAT=TDAT+1
012257	145	.BYTE TDAT
	000146	TDAT=TDAT+1
012260	146	.BYTE TDAT
	000147	TDAT=TDAT+1
012261	147	.BYTE TDAT
	000150	TDAT=TDAT+1
012262	150	.BYTE TDAT
	000151	TDAT=TDAT+1
012263	151	.BYTE TDAT

	000152	TDAT-TDAT+1
012264	152	.BYTE TDAT
	000153	TDAT-TDAT+1
012265	153	.BYTE TDAT
	000154	TDAT-TDAT+1
012266	154	.BYTE TDAT
	000155	TDAT-TDAT+1
012267	155	.BYTE TDAT
	000156	TDAT-TDAT+1
012270	156	.BYTE TDAT
	000157	TDAT-TDAT+1
012271	157	.BYTE TDAT
	000160	TDAT-TDAT+1
012272	160	.BYTE TDAT
	000161	TDAT-TDAT+1
012273	161	.BYTE TDAT
	000162	TDAT-TDAT+1
012274	162	.BYTE TDAT
	000163	TDAT-TDAT+1
012275	163	.BYTE TDAT
	000164	TDAT-TDAT+1
012276	164	.BYTE TDAT
	000165	TDAT-TDAT+1
012277	165	.BYTE TDAT
	000166	TDAT-TDAT+1
012300	166	.BYTE TDAT
	000167	TDAT-TDAT+1
012301	167	.BYTE TDAT
	000170	TDAT-TDAT+1
012302	170	.BYTE TDAT
	000171	TDAT-TDAT+1
012303	171	.BYTE TDAT
	000172	TDAT-TDAT+1
012304	172	.BYTE TDAT
	000173	TDAT-TDAT+1
012305	173	.BYTE TDAT
	000174	TDAT-TDAT+1
012306	174	.BYTE TDAT
	000175	TDAT-TDAT+1
012307	175	.BYTE TDAT
	000176	TDAT-TDAT+1
012310	176	.BYTE TDAT
	000177	TDAT-TDAT+1
012311	177	.BYTE TDAT
	000200	TDAT-TDAT+1
012312	200	.BYTE TDAT
	000201	TDAT-TDAT+1
012313	201	.BYTE TDAT
	000202	TDAT-TDAT+1
012314	202	.BYTE TDAT
	000203	TDAT-TDAT+1
012315	203	.BYTE TDAT
	000204	TDAT-TDAT+1
012316	204	.BYTE TDAT
	000205	TDAT-TDAT+1
012317	205	.BYTE TDAT
	000206	TDAT-TDAT+1



012320	206	.BYTE TDAT
	000207	TDAT-TDAT+1
012321	207	.BYTE TDAT
	000210	TDAT-TDAT+1
012322	210	.BYTE TDAT
	000211	TDAT-TDAT+1
012323	211	.BYTE TDAT
	000212	TDAT-TDAT+1
012324	212	.BYTE TDAT
	000213	TDAT-TDAT+1
012325	213	.BYTE TDAT
	000214	TDAT-TDAT+1
012326	214	.BYTE TDAT
	000215	TDAT-TDAT+1
012327	215	.BYTE TDAT
	000216	TDAT-TDAT+1
012330	216	.BYTE TDAT
	000217	TDAT-TDAT+1
012331	217	.BYTE TDAT
	000220	TDAT-TDAT+1
012332	220	.BYTE TDAT
	000221	TDAT-TDAT+1
012333	221	.BYTE TDAT
	000222	TDAT-TDAT+1
012334	222	.BYTE TDAT
	000223	TDAT-TDAT+1
012335	223	.BYTE TDAT
	000224	TDAT-TDAT+1
012336	224	.BYTE TDAT
	000225	TDAT-TDAT+1
012337	225	.BYTE TDAT
	000226	TDAT-TDAT+1
012340	226	.BYTE TDAT
	000227	TDAT-TDAT+1
012341	227	.BYTE TDAT
	000230	TDAT-TDAT+1
012342	230	.BYTE TDAT
	000231	TDAT-TDAT+1
012343	231	.BYTE TDAT
	000232	TDAT-TDAT+1
012344	232	.BYTE TDAT
	000233	TDAT-TDAT+1
012345	233	.BYTE TDAT
	000234	TDAT-TDAT+1
012346	234	.BYTE TDAT
	000235	TDAT-TDAT+1
012347	235	.BYTE TDAT
	000236	TDAT-TDAT+1
012350	236	.BYTE TDAT
	000237	TDAT-TDAT+1
012351	237	.BYTE TDAT
	000240	TDAT-TDAT+1
012352	240	.BYTE TDAT
	000241	TDAT-TDAT+1
012353	241	.BYTE TDAT
	000242	TDAT-TDAT+1
012354	242	.BYTE TDAT

	000243	TDAT=TDAT+1
012355	243	.BYTE TDAT
	000244	TDAT=TDAT+1
012356	244	.BYTE TDAT
	000245	TDAT=TDAT+1
012357	245	.BYTE TDAT
	000246	TDAT=TDAT+1
012360	246	.BYTE TDAT
	000247	TDAT=TDAT+1
012361	247	.BYTE TDAT
	000250	TDAT=TDAT+1
012362	250	.BYTE TDAT
	000251	TDAT=TDAT+1
012363	251	.BYTE TDAT
	000252	TDAT=TDAT+1
012364	252	.BYTE TDAT
	000253	TDAT=TDAT+1
012365	253	.BYTE TDAT
	000254	TDAT=TDAT+1
012366	254	.BYTE TDAT
	000255	TDAT=TDAT+1
012367	255	.BYTE TDAT
	000256	TDAT=TDAT+1
012370	256	.BYTE TDAT
	000257	TDAT=TDAT+1
012371	257	.BYTE TDAT
	000260	TDAT=TDAT+1
012372	260	.BYTE TDAT
	000261	TDAT=TDAT+1
012373	261	.BYTE TDAT
	000262	TDAT=TDAT+1
012374	262	.BYTE TDAT
	000263	TDAT=TDAT+1
012375	263	.BYTE TDAT
	000264	TDAT=TDAT+1
012376	264	.BYTE TDAT
	000265	TDAT=TDAT+1
012377	265	.BYTE TDAT
	000266	TDAT=TDAT+1
012400	266	.BYTE TDAT
	000267	TDAT=TDAT+1
012401	267	.BYTE TDAT
	000270	TDAT=TDAT+1
012402	270	.BYTE TDAT
	000271	TDAT=TDAT+1
012403	271	.BYTE TDAT
	000272	TDAT=TDAT+1
012404	272	.BYTE TDAT
	000273	TDAT=TDAT+1
012405	273	.BYTE TDAT
	000274	TDAT=TDAT+1
012406	274	.BYTE TDAT
	000275	TDAT=TDAT+1
012407	275	.BYTE TDAT
	000276	TDAT=TDAT+1
012410	276	.BYTE TDAT
	000277	TDAT=TDAT+1

012411	277	.BYTE TDAT
	000300	TDAT=TDAT+1
012412	300	.BYTE TDAT
	000301	TDAT=TDAT+1
012413	301	.BYTE TDAT
	000302	TDAT=TDAT+1
012414	302	.BYTE TDAT
	000303	TDAT=TDAT+1
012415	303	.BYTE TDAT
	000304	TDAT=TDAT+1
012416	304	.BYTE TDAT
	000305	TDAT=TDAT+1
012417	305	.BYTE TDAT
	000306	TDAT=TDAT+1
012420	306	.BYTE TDAT
	000307	TDAT=TDAT+1
012421	307	.BYTE TDAT
	000310	TDAT=TDAT+1
012422	310	.BYTE TDAT
	000311	TDAT=TDAT+1
012423	311	.BYTE TDAT
	000312	TDAT=TDAT+1
012424	312	.BYTE TDAT
	000313	TDAT=TDAT+1
012425	313	.BYTE TDAT
	000314	TDAT=TDAT+1
012426	314	.BYTE TDAT
	000315	TDAT=TDAT+1
012427	315	.BYTE TDAT
	000316	TDAT=TDAT+1
012430	316	.BYTE TDAT
	000317	TDAT=TDAT+1
012431	317	.BYTE TDAT
	000320	TDAT=TDAT+1
012432	320	.BYTE TDAT
	000321	TDAT=TDAT+1
012433	321	.BYTE TDAT
	000322	TDAT=TDAT+1
012434	322	.BYTE TDAT
	000323	TDAT=TDAT+1
012435	323	.BYTE TDAT
	000324	TDAT=TDAT+1
012436	324	.BYTE TDAT
	000325	TDAT=TDAT+1
012437	325	.BYTE TDAT
	000326	TDAT=TDAT+1
012440	326	.BYTE TDAT
	000327	TDAT=TDAT+1
012441	327	.BYTE TDAT
	000330	TDAT=TDAT+1
012442	330	.BYTE TDAT
	000331	TDAT=TDAT+1
012443	331	.BYTE TDAT
	000332	TDAT=TDAT+1
012444	332	.BYTE TDAT
	000333	TDAT=TDAT+1
012445	333	.BYTE TDAT

	000334	TDAT=TDAT+1
012446	334	.BYTE TDAT
	000335	TDAT=TDAT+1
012447	335	.BYTE TDAT
	000336	TDAT=TDAT+1
012450	336	.BYTE TDAT
	000337	TDAT=TDAT+1
012451	337	.BYTE TDAT
	000340	TDAT=TDAT+1
012452	340	.BYTE TDAT
	000341	TDAT=TDAT+1
012453	341	.BYTE TDAT
	000342	TDAT=TDAT+1
012454	342	.BYTE TDAT
	000343	TDAT=TDAT+1
012455	343	.BYTE TDAT
	000344	TDAT=TDAT+1
012456	344	.BYTE TDAT
	000345	TDAT=TDAT+1
012457	345	.BYTE TDAT
	000346	TDAT=TDAT+1
012460	346	.BYTE TDAT
	000347	TDAT=TDAT+1
012461	347	.BYTE TDAT
	000350	TDAT=TDAT+1
012462	350	.BYTE TDAT
	000351	TDAT=TDAT+1
012463	351	.BYTE TDAT
	000352	TDAT=TDAT+1
012464	352	.BYTE TDAT
	000353	TDAT=TDAT+1
012465	353	.BYTE TDAT
	000354	TDAT=TDAT+1
012466	354	.BYTE TDAT
	000355	TDAT=TDAT+1
012467	355	.BYTE TDAT
	000356	TDAT=TDAT+1
012470	356	.BYTE TDAT
	000357	TDAT=TDAT+1
012471	357	.BYTE TDAT
	000360	TDAT=TDAT+1
012472	360	.BYTE TDAT
	000361	TDAT=TDAT+1
012473	361	.BYTE TDAT
	000362	TDAT=TDAT+1
012474	362	.BYTE TDAT
	000363	TDAT=TDAT+1
012475	363	.BYTE TDAT
	000364	TDAT=TDAT+1
012476	364	.BYTE TDAT
	000365	TDAT=TDAT+1
012477	365	.BYTE TDAT
	000366	TDAT=TDAT+1
012500	366	.BYTE TDAT
	000367	TDAT=TDAT+1
012501	367	.BYTE TDAT
	000370	TDAT=TDAT+1

012502	370	.BYTE TDAT
	000371	TDAT=TDAT+1
012503	371	.BYTE TDAT
	000372	TDAT=TDAT+1
012504	372	.BYTE TDAT
	000373	TDAT=TDAT+1
012505	373	.BYTE TDAT
	000374	TDAT=TDAT+1
012506	374	.BYTE TDAT
	000375	TDAT=TDAT+1
012507	375	.BYTE TDAT
	000376	TDAT=TDAT+1
012510	376	.BYTE TDAT
	000377	TDAT=TDAT+1
012511	377	.BYTE TDAT
	000400	TDAT=TDAT+1
12		.EVEN
13	012512	000000
14	012554	RBUF: 0
		. = .+40

1		.MACRO WORDS WDNAM,K,DATA
2		'WDNAME' K' : DATA
3		.ENDM WORDS
5	000020	K=KX
6	127400	DATA=DATA
7	000000	KX=0
9	000020	.REPT 20
10		.NLIST
11		DATA=KX*400+120000
12		.LIST
13		WORDS +/RWRD/, \KX, \DATA
14		.NLIST
15		KX=KX+1
16		.LIST
17		.ENDR
	120000	DATA=KX*400+120000
012554		WORDS +/RWRD/, \KX, \DATA
012554	120000	RWRD0: 120000
	000001	KX=KX+1
	120400	DATA=KX*400+120000
012556		WORDS +/RWRD/, \KX, \DATA
012556	120400	RWRD1: 120400
	000002	KX=KX+1
	121000	DATA=KX*400+120000
012560		WORDS +/RWRD/, \KX, \DATA
012560	121000	RWRD2: 121000
	000003	KX=KX+1
	121400	DATA=KX*400+120000
012562		WORDS +/RWRD/, \KX, \DATA
012562	121400	RWRD3: 121400
	000004	KX=KX+1
	122000	DATA=KX*400+120000
012564		WORDS +/RWRD/, \KX, \DATA
012564	122000	RWRD4: 122000
	000005	KX=KX+1
	122400	DATA=KX*400+120000
012566		WORDS +/RWRD/, \KX, \DATA
012566	122400	RWRD5: 122400
	000006	KX=KX+1
	123000	DATA=KX*400+120000
012570		WORDS +/RWRD/, \KX, \DATA
012570	123000	RWRD6: 123000
	000007	KX=KX+1
	123400	DATA=KX*400+120000
012572		WORDS +/RWRD/, \KX, \DATA
012572	123400	RWRD7: 123400
	000010	KX=KX+1
	124000	DATA=KX*400+120000
012574		WORDS +/RWRD/, \KX, \DATA
012574	124000	RWRD10: 124000
	000011	KX=KX+1
	124400	DATA=KX*400+120000
012576		WORDS +/RWRD/, \KX, \DATA
012576	124400	RWRD11: 124400
	000012	KX=KX+1
	125000	DATA=KX*400+120000
012600		WORDS +/RWRD/, \KX, \DATA

012600	125000	RWRD12: 125000
	000013	KX=KX+1
	125400	DATAX=KX*400+120000
012602		WORDS +/RWRD/, \KX, \DATAX
012602	125400	RWRD13: 125400
	000014	KX=KX+1
	126000	DATAX=KX*400+120000
012604		WORDS +/RWRD/, \KX, \DATAX
012604	126000	RWRD14: 126000
	000015	KX=KX+1
	126400	DATAX=KX*400+120000
012606		WORDS +/RWRD/, \KX, \DATAX
012606	126400	RWRD15: 126400
	000016	KX=KX+1
	127000	DATAX=KX*400+120000
012610		WORDS +/RWRD/, \KX, \DATAX
012610	127000	RWRD16: 127000
	000017	KX=KX+1
	127400	DATAX=KX*400+120000
012612		WORDS +/RWRD/, \KX, \DATAX
012612	127400	RWRD17: 127400
	000020	KX=KX+1

; 012614

.PFAIL

;ENTER HERE ON POWER FAILURE

```

012614 010046          PFAIL:  MOV    R0,-(SP)          ;SAVE R0-R5 ON PROCESSOR STACK
012616 010146          MOV    R1,-(SP)
012620 010246          MOV    R2,-(SP)
012622 010346          MOV    R3,-(SP)
012624 010446          MOV    R4,-(SP)
012626 010546          MOV    R5,-(SP)
012630 016746 165170   MOV    24,-(SP)
012634 010667 177204   MOV    SP,SAVSP          ;SAVE STACK POINTER
012640 012767 012652 165156  MOV    @RESTART,24      ;SET UP FOR POWER UP TRAP ; 3
012646 000000          HALT                    ;HALT ON POWER DOWN NORMAL
012650 000777          BR      .

```

;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED

```

012652 016706 177166   RESTAR: MOV    SAVSP,SP          ;RESTORE STACK POINTER
012656 012605          MOV    (SP)+,R5          ;RESTORE R0-R5
012660 012604          MOV    (SP)+,R4
012662 012603          MOV    (SP)+,R3
012664 012602          MOV    (SP)+,R2
012666 012601          MOV    (SP)+,R1
012670 012600          MOV    (SP)+,R0
012672 012767 012614 165124  MOV    @PFAIL,24      ;SET UP FOR POWER FAILURE
012700 012767 000340 165070  MOV    @340,PS
012706 012706 013724   MOV    @STACK,SP
012712 005067 000374   CLR    TEMP
012716 005267 000370   INC    TEMP
012722 001375          BNE    -4
012724 104401          TYPE                                ; 5
012726 013114          MCRLF                                ; 5
012730 104402          OCTASC
012732 012754          PFTAB
012734 104401          TYPE
012736 013117          MPFAIL
012740 005067 177044   CLR    ERRFLG
012744 005067 177104   CLR    LAST
012750 000177 177042   JMP    @RETRN
012754 000001          PFTAB: 1
012756 000006 000002   6,2
012762 012016          RETRN

```



012764				.MSG	↑/DH11 HALF-DUPLEX AND BREAK TEST/,↑/CZDHI-D0/
012764	015	012	012	MTITLE:	.ASCIZ <15><12><12>/DH11 HALF-DUPLEX AND BREAK TEST /<15><12>
012767	104	110	061		
012772	061	040	110		
012775	101	114	106		
013000	055	104	125		
013003	120	114	105		
013006	130	040	101		
013011	116	104	040		
013014	102	122	105		
013017	101	113	040		
013022	124	105	123		
013025	124	040	015		
013030	012	000			
013032	015	012	126	MVECTOR:	.ASCIZ <15><12>/VECTOR ADDRESS-/
013035	105	103	124		
013040	117	122	040		
013043	101	104	104		
013046	122	105	123		
013051	123	055	000		
013054	015	012	103	MREGAD:	.ASCIZ <15><12>/CONTROL REGISTER ADDRESS-/
013057	117	116	124		
013062	122	117	114		
013065	040	122	105		
013070	107	111	123		
013073	124	105	122		
013076	040	101	104		
013101	104	122	105		
013104	123	123	055		
013107	000				
013110	040	040	077	MQM:	.ASCIZ / ?/
013113	000				
013114	015	012	000	MCRLF:	.ASCIZ <15><12>
013117	040	040	120	MPFAIL:	.ASCIZ / POWER FAILURE, PROGRAM RESTART AT TEST IN PROGRESS/
013122	117	127	105		
013125	122	040	106		
013130	101	111	114		
013133	125	122	105		
013136	054	040	120		
013141	122	117	107		
013144	122	101	115		
013147	040	122	105		
013152	123	124	101		
013155	122	124	040		
013160	101	124	040		
013163	124	105	123		
013166	124	040	111		
013171	116	040	120		
013174	122	117	107		
013177	122	105	123		
013202	123	000			
013204	015	012	103	MEPASS:	.ASCIZ <15><12>/CZDHI-D0/
013207	132	104	110		
013212	111	055	104		
013215	060	000			
013217	015	012	120	PASTXT:	.ASCIZ <15><12>/PASS COUNT = /

013222	101	123	123	
013225	040	103	117	
013230	125	116	124	
013233	040	075	040	
013236	000			
013237	015	012	122	MR: .ASCIZ <15><12>/R/
013242	000			
013243	015	012	124	MTSTPC: .ASCIZ <15><12>/TEST PC-/
013246	105	123	124	
013251	040	120	103	
013254	055	000		

2  
3 013256 .EVEN  
.EVEN  
.TRPTAB

;TABLE OF POINTERS FOR TRAP DECODING

013256	010600	TRPTAB:	SCOPER
013260	011254		TYPER
013262	011574		OCTASN
013264	011306		INSTRG
013266	011400		INSTRE
013270	011410		PARAMS
013272	011162		SV05P
013274	011222		RS05
013276	010700		SCOP1R
4 013300		.BUFFER	

;BUFFERS FOR INPUT-OUTPUT

013300	000000	INBUF:	0
013312	000000	. = . +10	
013312	000000	TEMP:	0
	013324	. = . +10	
013324	000000	MDATA:	0
	013336	. = . +10	
5 013336		.ERRTAB	

;TABLE OF POINTERS TO ERROR MESSAGES AND DATA

013336		ERRTAB:	EM1	
6 013336	013352		DT1	
7 013340	013512		EM2	
8 013342	013426		DT1	
9 013344	013512		EM3	
10 013346	013464		0	
11 013350	000000			
12 013352	115	117	122	EM1: .ASCIZ /MORE THAN 1 CHARACTER RECEIVED/<15><12>/EXP REC/
	013355	105	040	124
	013360	110	101	116
	013363	040	061	040
	013366	103	110	101
	013371	122	101	103
	013374	124	105	122
	013377	040	122	105
	013402	103	105	111

	013405	126	105	104			
	013410	015	012	105			
	013413	130	120	040			
	C13416	040	040	040			
	013421	040	122	105			
	013424	103	000				
13	013426	102	122	105	EM2:	.ASCIZ	/BREAK DATA ERROR/<15><12>/EXP REC/
	013431	101	113	040			
	013434	104	101	124			
	013437	101	040	105			
	013442	122	122	117			
	013445	122	015	012			
	013450	105	130	120			
	013453	040	040	040			
	013456	040	040	122			
	013461	105	103	000			
14	013464	122	105	103	EM3:	.ASCIZ	/RECEIVFR NOT BLINDED/
	013467	105	111	125			
	013472	105	122	040			
	013475	116	117	124			
	013500	040	102	114			
	013503	111	116	104			
	013506	105	104	000			
15					.EVEN		
16	013512	000002			DT1:	2	
17	013514	006	002		.BYTE	6.2	
18	013516	012042				SAVR5	
19	013520	006	002		.BYTE	6.2	
20	013522	012040				SAVR4	
21	013524				.ENDCOD		
	013524	000000			ENDCOD:	0	
22		000001			.END		

ADRCNT = 011573	ENDFLG 012056	N = 000001	SAVR0 012030	TYPDAT 011040
BEGIN 001306	EOP 010500	NULL 012060	SAVR1 012032	TYPE = 104401
BINWRD 011744	ERRCNT 012014	OCTASC = 104402	SAVR2 012034	TYPER 011254
BITX = 000000	ERRFLG 012010	OCTASN 011574	SAVR3 012036	TYPMSG 011016
BIT00 = 000001	ERRMSG 011036	PARAM = 104405	SAVR4 012040	T1 001400
BIT01 = 000002	ERRORS 010716	PARAMS 011410	SAVR5 012042	T10 003466
BIT02 = 000004	ERRTAB 013336	PARAM1 011440	SAVSP 012044	T11 003720
BIT03 = 000010	ERTAB0 011122	PARERR 011514	SAV05P = 104406	T12 004152
BIT04 = 000020	ESCAPE 012020	PASARG 010572	SCOPE = 104400	T13 004404
BIT05 = 000040	EX1TER 011100	PASCNT 012012	SCOPE1 = 104410	T14 004636
BIT06 = 000100	FREEZ1 012022	PASTXT 013217	SCOPE1R = 104410	T15 005070
BIT07 = 000200	HALTS 011060	PFAIL 012614	SCOPE1R 010700	T16 005322
BIT08 = 000400	HILIM 011566	PFTAB 012754	SPACNT = 011743	T17 005554
BIT09 = 001000	ICOUNT 012024	POPRO = 012600	STACK = 013724	T2 001632
BIT10 = 002000	INBUF 013300	POP1SP = 005726	START 001004	T20 006006
BIT11 = 004000	INIFLG 012050	POP2SP = 022626	STFLG 012052	T21 006240
BIT12 = 010000	INSTER = 104404	PS = 177776	SV05 011170	T22 006352
BIT13 = 020000	INSTR = 104403	PUSHRO = 010046	SV05P 011162	T23 006464
BIT14 = 040000	INSTKE 011400	PUSH1S = 005746	SWR 001000	T24 006576
BIT15 = 100000	INSTRG 011306	PUSH2S = 024646	SW00 = 000001	T25 006710
CHRCNT 011742	INSTR1 011320	RBUF 012512	SW01 = 000002	T26 007022
CLRAL. 012064	INSTR2 011406	RESREG 011056	SW02 = 000004	T27 007134
DATA = 127400	K = 000020	RESTAR 012652	SW03 = 000010	T3 002064
DATABP 011054	KX = 000020	RESTRT 010566	SW04 = 000020	T30 007246
DATAX = 127400	LAST 012054	RES05 = 104407	SW05 = 000040	T31 007360
DEVADR 011570	LIGHTS 001002	RETRN 012016	SW06 = 000100	T32 007472
DHBA 011764	LIMITS 011520	RS05 011222	SW08 = 000400	T33 007604
DHBAR 011770	LINACT 012062	RWRD0 012554	SW09 = 001000	T34 007716
DHBC 011766	LINE = 000020	RWRD1 012556	SW10 = 002000	T35 010030
DHBCR 011772	LOBITS 011572	RWRD10 012574	SW11 = 004000	T36 010142
DHLPR 011762	LOGICA 010556	RWRD11 012576	SW12 = 010000	T37 010254
DHNR 011760	LOLIM 011564	RWRD12 012600	SW13 = 020000	T4 002316
DHR.VL 012002	LPCNT 012026	RWRD13 012602	SW14 = 040000	T40 010366
DHRVEC 012000	MCRLF 013114	RWRD14 012604	SW15 = 100000	T5 002550
DHSCR 011756	MDATA 013324	RWRD15 012606	TBUF 012112	T6 003002
DHSLR 011776	MEPASS 013204	RWRD16 012610	TDAT = 000400	T7 003234
DHSSR 011774	MPFAIL 013117	RWRD17 012612	TEMP 013312	VEC1 001164
DHTLVL 012006	MQM 013110	RWRD2 012560	TKCSR 011746	VEC2 001174
DHTVEC 012004	MR 013237	RWRD3 012562	TKDBR 011750	WRDCNT 011740
DT1 013512	MREGAD 013054	RWRD4 012564	TPCSR 011752	X = 000000
EM1 013352	MSG 011322	RWRD5 012566	TPDBR 011754	XBIT = 000000
EM2 013426	MTITLE 012764	RWRD6 012570	TRPOK 011142	XLIN = 000020
EM3 013464	MTSTPC 013243	RWRD7 012572	TRPSRV 011130	XN = 000041
ENDCOD 013524	MVECTO 013032	SAVPC 012046	TRPTAB 013256	Y = 000011

. ABS. 013526 000  
000000 001  
ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 18176 WORDS ( 71 PAGES)  
DYNAMIC MEMORY AVAILABLE FOR 71 PAGES  
CZDHID.BIN,CZDHID.SEQ=DHMACA.MAC,CZDHID.P11