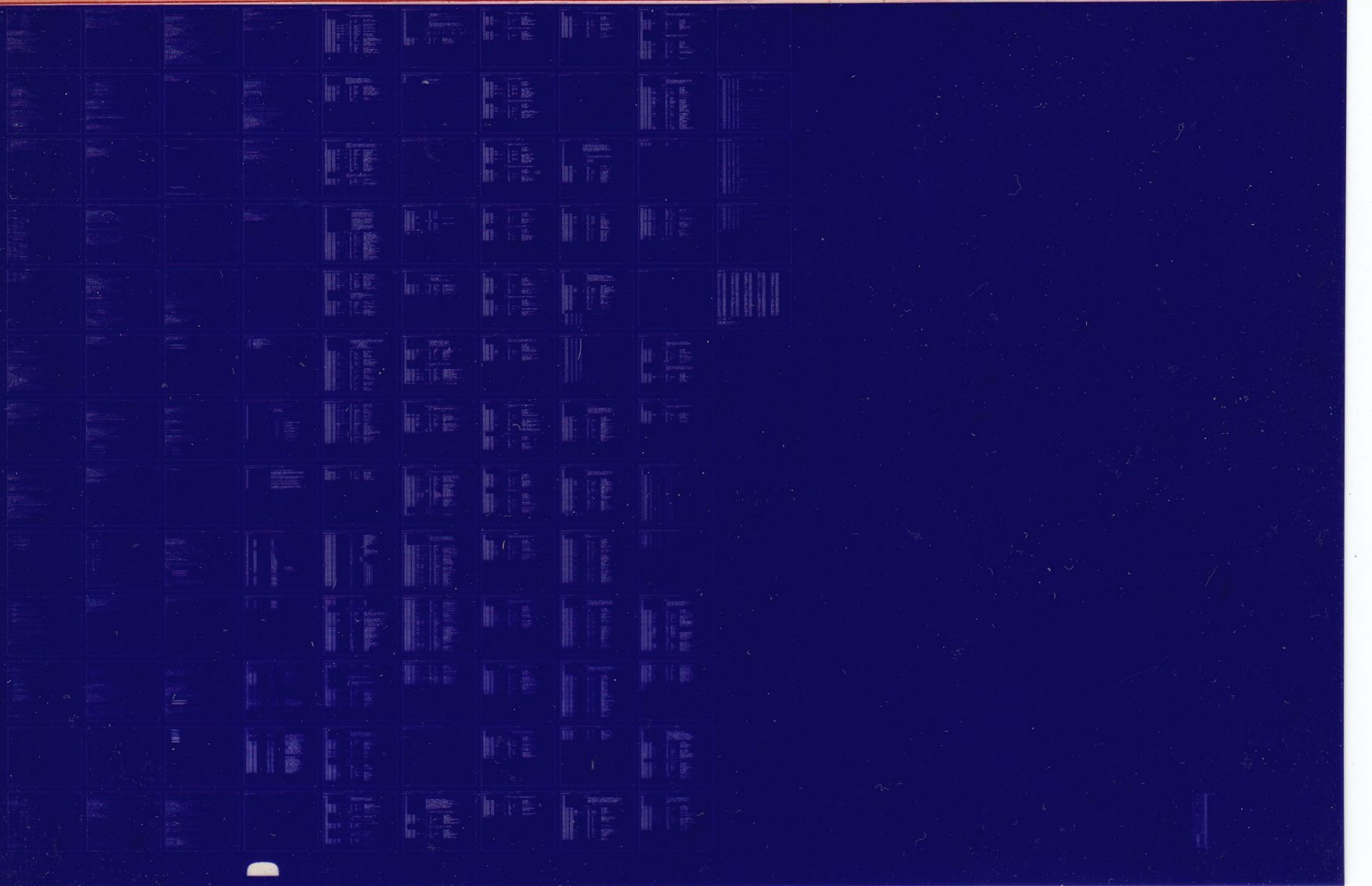


LA36

LA36 TERMINATOR (DL11&KL11)
CZLACE0

AH-8899E-MC
COPYRIGHT © 74-78
FICHE 1 OF 1

DEC 1978
digital
MADE IN USA



IDENTIFICATION

B 1

SEQ 0001

Product Code: AC-8898E-MC
Product Name: CZLACEO LA36 TERM (DL11 & KL11)
Date Created: August 1978
Maintainer: DIAGNOSTIC GROUP
Authors: Robert W. Baker
R. Quenneville
Ralph A. Schaubert
John V. Chatalian

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 manual.

The software described in this document is furnished to the purchaser under a license for use on a single computer system and can be copied (with inclusion of Digital's copyright notice) only for use in such system, except as may otherwise be provided in writing by Digital.

Digital Equipment Corporation assumes no responsibility for the use or reliability of its software on equipment that is not supplied by Digital.

Copyright (C) 1974,1977,1978 by Digital Equipment Corporation

PAGE 2

HISTORY

1.0 DECO CZL
AC-E-0

1.1.0 Closed Problem Report AA3318

1.1.1 It was reported that a DL11-A operated at 110 baud caused failure in the AREAD routine because the 200 msec. delay is not of sufficient duration to allow setting of the Receiver Register Status 'DONE' bit through the Maintenance bit facility.
The time delay was increased from 200 to 600 msec.

1.2.0 Closed Problem Report AA3643

1.2.1 Tests 56,57,60,61,62,63,64,65, and 66 do not run properly when run on an LSI-11. This problem was resolved by changing the branch after the CHAIN command to go back to test for the LSI-11 switch in order to effect the appropriate action during each test.

1.2.2 Second time-out in Test 64 allows excessive wait for operator response. The time delay was reduced from '177777' to '600'.

1.2.3 Common routine TYPE does not save the contents of R0 resulting in the loss of this information and consequent failure.

Instructions were included to save the contents of R0 on entry into the routine and to restore them upon exit.

1.2.4 Loss of stack contents for non-LSI-11 computers due to incorrect sequence of instructions in Test 65 was also reported in Problem Report AA3803. Refer to 1.3.1.

1.3.0 Closed Problem Report AA3803

1.3.1 Testing of non-LSI-11 computer
s results in the program hanging
up because the stack gets popped awa
y in Test 65. The branch
after the test for the LSI-11 switch in Test 6
5 should go to
the CHAIN command for proper exit from the test for n
on-LSI-11
machines. This change supersedes the change released in DEP
0
MD-11-DZLAC-D-1.

TABLE OF CONTENTS

1.0	ABSTRACT
2.0	REQUIREMENTS
2.1	Equipment and Assignments
2.2	Storage
2.3	Preliminary Programs
2.4	Additional Programs
3.0	LOADING PROCEDURE AND INITIALIZATION
4.0	STARTING PROCEDURE
4.1	Startin g Addresses
4.2	Switch Register Control With I/O Tests
4.3	Switch Register Control Without I/O Tests
4.4	Keyboard Control With I/O Tests
4.5	Keyboard Control Without I/O Tests
5.0	OPERATING PROCEDURE
5.1	Switch Register Control
5.2	Keyboard Control
6.0	TEST DE SCRIPTIONS
6.1	Printing Tests
6.1.1	Test0 - Data Path Test
6.1.2	Test1 - Printable Character Test
6.1.3	Test2 - Non-printable Character Test
6.1.4	Test3 - Carriage Return Test
6.1.5	Test4 - Multiple Line Feed Test
6.1.6	Test5 - Single Line Feed Test
6.1.7	Test6 - Backspace Test
6.1.8	Test7 - Overprint Test
6.1.9	Test8 - Printing Frequency Sweep Test
6.1.10	Test11 - Ribbon Feed Test
6.1.11	Test12 - Printer Bell Test
6.1.12	Test17 - Life Test

- 6.2 Echo Tests
- 6.2.1 Test20 - Character Echo Test
- 6.2.2 Test21
- Line Echo Test, Fast Rate
- 6.2.3 Test22 - Line Echo Test, Slow Rate
- 6.
- 6.2.4 Test23 - Character/Code Echo Test
- 6.2.5 Test24 - Selected Pat
- tern Echo Test
- 6.2.6 Test25 - Bell Echo Test
- 6.4 Standard I/O Tes
- ts

1.0 ABSTRACT

This diagnostic is divided into three basic sections:

1. A check of the console terminal interface logic.
2. A check of the printing characteristics and control logic.
3. An echo portion designed to check the keyboard and to aid in the diagnosis of terminal problems.

Patterns used by the printing tests were chosen for ease of verification. The echo tests were designed for maximum flexibility, with Test 24 allowing any desired pattern to be used.

2.0 REQUIREMENTS

2.1 EQUIPMENT AND ASSIGNMENTS

The diagnostic is written to run on all models of the PDP-11 computer with either a KL11 or DL11 console terminal interface. The diagnostic is preset to test up to 16 additional terminals (on DL11's) assigned between addresses 776500 and 776676. This preset quantity (16) and pre-set address (776500) can be changed by depositing the quantity in DLNR and the starting address in DLADR. For example, to allow for up to 31 additional terminals, the address 775610 could be placed into DLADR and the octal equivalent of 31, i.e., (37) would be placed into DLNR. The number of additional DL11's actually tested will be adjusted automatically downward based upon the first DL11 address (within the implied range) found to be unresponsive. Thus if there is no DL11 present to match the address in DLADR only the console terminal will be tested. Therefore, all DL11's in excess of the console terminal must have contiguous address assignments with the lowest address corresponding

ponding to the value in DLADR.

The console terminal (assigned standard) can be reassigned by placing the address of its receiver status register into CONNADD and its receiver interrupt vector into CONVEC. This reassignment can be made to a terminal within the set of terminals implied by DLNR and DLADR without adverse effect. Note that a terminal with a slower speed (if any) will determine the speed at which all of the terminals are tested. Such a terminal should generally be excluded from the test, or tested separately. (Refer to the symbol definitions in the listing for the above mentioned locations.)

2.2 STORAGE

The diagnostic program uses all of 4K of memory with exception of the area used by the absolute loader.

2.3 PRELIMINARY PROGRAMS

Any applicable PDP-11 diagnostics should be run on the processor. If any errors are encountered during the interface check, refer to the appropriate interface diagnostic for further help in locating the problem if needed.

2.4 ADDITIONAL PROGRAMS

This diagnostic is for verification of basic terminal functions only. If the terminals under test have hardware options installed run diagnostic MAINDEC-11-DZLAF-A, the LA36 TERMINAL OPTIONS TEST.

3.0 LOADING PROCEDURE AND INITIALIZATION

Load the LA36 diagnostic program tape following normal procedures. Before starting the program, refer to the description of the routine 'DLY'. Time delays used by the program are a function of the CPU model and memory type and should be set-up before running the diagnostic. The routine is preset for a PDP-11/05 with core memory. Refer to Section 2.1 for non-standard terminal addresses and for testing multiple DL11 interfaces.

If a hardware switch register does not exist, the program will use the contents of location 176 as the value of the switches. Therefore, be sure to load location 176 with the switch value before starting the program when not using hardware switches.

If the CPU is an LSI-11, 11/03 be sure to set switch register bit 9 to a 1. Special tests are run on the DLV11 interface.

4.0 STARTING PROCEDURE

4.1 STARTING ADDRESSES

200(8) = Run with Switch Register Control

- perform Console Terminal I/O tests.

204(8) = Run with Switch Register Control

- skip Console Terminal I/O tests.

210(8) = Run with Keyboard Control

- perform Console Terminal I/O tests.

214(8) = Run with Keyboard Control

- skip Console Terminal I/O tests.

ests.

ests.

ests.

E
P
B
E
H
4
7
:
:
E
E
T
L

4.2 Switch Register Control With I/O Tests

A. Set the switch register to 200(8) and press the load address switch.

B. Set switch register bit 9 to a 1 if the processor is an LSI-11, 11/0.
3. Refer to Section 5.1.5.

C. Set the switch register bits 7-0 equal to the paper width in terms of the number of columns (octal). Refer to Section 5.1.8.

D. Set the switch register bit 8 equal to 1 or 0 and press the start switch. A message will be printed indicating the number of DL11's being tested. Refer to Section 5.1.
6.

E. If bit 8 were zero when starting, the Printer tests are executed sequentially, after the entire series of I/O tests are executed.

F. If bit 8 was set when the start switch was pressed, the entire series of I/O tests will be executed and the CPU will halt at location SELHLT. The program will then be waiting for control via the switch register.

4.3 Switch Register Control - Without I/O Tests

Same as Section 4.2 except in step A, set the switch register to 204(8).

8
7
6
5
4
3
2
1
0

PAGE 7

4.4 Keyboard Control - With I/O Tests

- A. Set the switch register to 210(8) and press the load address switch.
- B. Set the switch register bits 7-0 equal to the paper width in terms of the number of columns (octal). Refer to Section 5.1.8.
- C. Set switch register bit 9 to a 1 if the processor is an LSI-11, 11/03. Refer to Section 5.1.5.
- D. Set switch 8 and press the start switch. A message will be printed indicating the number of DL11's being tested. Refer to Section 5.1.6.
- E. If bit 8 was zero when starting, the printer tests are executed sequentially after the entire series of I/O tests are executed.
- F. If bit 8 were set when the start switch was pressed, the entire series of I/O tests will be executed followed by the select test message. The program will then be waiting for a test selection via any terminal keyboard. Refer to Section 5.2.

4.5 Keyboard Control - Without I/O Tests

Same as Section 4.4 except in step A, set the switch register to 214 (8).

5.0 OPERATING PROCEDURE

The program can be controlled in either of two methods: by the con-

sole switch register or from the keybo
ard of the terminal(s) under
test.

M 1

SEQ 0012

E
O

5.1 SWITCH REGISTER CONTROL

The various switches and their functions are listed below. Switches may be changed and set as desired except as noted in the specific switch descriptions. Refer to the detailed switch descriptions for further, more complete information.

SWITCH NUMBER	DESCRIPTION
15 TEST AT END OF TEST	1(up) = HALT 0(down) = CONTINUE TEST SEQUENCE
14 ERROR	1(up) = CONTINUE ON ERROR 0(down) = HALT ON ERROR
13	1(up) = DRIVE ONLY CONSOLE TERMINAL 0(down) = DRIVE ALL TERMINALS
11 INDIVIDUAL TEST	1(up) = LOOP ON INDIVIDUAL TEST 0(down) = NORMAL TEST SEQUENCE
9 ALL OTHER PDP-11'S	1(up) = CPU TYPE IS AN LSI-11, 11/03 0(down) = ALL OTHER PDP-11'S
8	1(up) = RUN TEST ONCE AND HALT 0(down) = LOOP ON TEST SEQUENCE
5-0 TEST NUMBER SELECTION	
7-0 ART-UP	NUMBER OF COLUMNS AT START

5.1.1 Switch 15

With switch 15 in the up position, the program will halt at the end of the current test. Replacing switch 15 to the down position and press-

ing CONTINUE will continue the normal test operation. Dur
ing the
halt, any of the control switches may be changed or set as desired.

SEQ 0014

5.1.2 Switch 14

Placing switch 14 in the up position will cause the pro
gram to contin-
ue on errors during any of the I/O tests only. With switch 1
4 down,
the program will halt (at ERRHLT) on any errors during the I-O tests
with the location of the error in R0. Pressing CONTINUE will cause
the pro
gram to continue if switch 14 is down. With switch 14 up,
pressing contin
ue will cause the program to loop on the error.

PAGE 9

NOTE

Error halts can occur only during the I/O tests. The terminal is connected to a serial line and there is no error information returned to the program from the terminal. Therefore the program cannot report errors occurring in the terminal. Errors detected during the interface tests will result in halts as described above.

5.1.3 Switch 13

Placing switch 13 in the down position will cause the driving of all multiple terminals during the Printer tests only. If switch 13 is up, only the console terminal is driven.

** Note: Switch 13 should only be changed when the program is waiting for a test selection.

5.1.4 Switch 11

Placing switch 11 up at any time will cause the program to loop on the current test as long as switch 11 remains up. Replacing switch 11 down will cause the program to resume normal operation at the completion of the test.

5.1.5 Switch 9

Placing switch 9 up at the start of the test will cause an automatic change in the DELAY timing, and the execution of special DLV11 I/O tests. The DLV11 has no maintenance mode and will cause th

e program
to hang if tested as a DL11.

5.1.6 Switch 8

With switch 8
in the down position the program will continue to loop
through the present t
est sequence. Placing switch 8 up will cause the
program to halt (at SELHLT) a
t the completion of the current test.
After the halt, set the control switc
hes as desired and set switches 5
to 0 to the next desired test number, and the
n press CONTINUE to start
the test.

PAGE 10

When starting the diagnostic the operator can select a specific test rather than automatically starting the printing test sequence by setting switch 8 up before starting the diagnostic. Upon completion of the I/O test sequence (if being run) the program will either halt at SELHLT waiting for a test selection via the switch register or print the select test message and wait for a test selection from any keyboard. Refer to Section 4 for further information.

5.1.7 Switches 5 to 0

Switches 5 to 0 are used to select specific tests when under switch register control. Test numbers are always in octal.

5.1.8 Switches 7 to 0 (at start-up only)

At start-up only, switches 7 to 0 are used to set the desired maximum number of columns the diagnostic is to test. If the number set is greater than 132(10) or less than 30(10), the program will default to 132(10). The value set must be in octal form. Thus, for normal operation leave switches 7 to 0 down to test the full 132(10) columns.

5.2 KEYBOARD CONTROL

The program will be under keyboard control whenever the diagnostic is started at location 210 or 214. Switches on the console switch register will have no effect when under terminal control except for switch 15. The I/O tests cannot be selected when under keyboard control.

To stop a test at any time, type the 'RUBOUT' or 'DELETE' key on any keyboard. Any terminal may stop the test and select the next test if switch 13 is down. When a test is stopped by typing a 'RUBOUT' or 'DELETE', the test will terminate and the following message will be typed:

SELECT TEST NUMBER

At this time, type the desired test number followed by any one of the following control characters:

. (period) = Run the selected test once and return for another test selection.

L = Loop on the selected test until a 'RUBOUT' is typed.

S = Start the test sequence with the selected test. Continue to loop on the printing test sequence until a 'RUBOUT' is typed.

The 'L' or 'S' may be either upper or lower case, but the test number must always be a 2 digit octal number. The test number and terminator are echoed by the program, thus each character will be printed twice if the terminal is in half duplex. For all echo tests, the 'L' and 'S' will only run the test once (the same as if typing a period). For

all option tests, the 'S' will only run the test once (the same as if typing a period), however, typing an 'L' will cause the program to loop on the selected test. If an error is detected in the test selection (illegal test number or control character), a question mark is printed and the message will be repeated.

6.0 TEST DESCRIPTION

6.1 PRINTING TESTS

These tests are designed as a test of the printing mechanism and the associated control logic. At the beginning of each test, the test number will be printed indicating which test is being executed and, if the test is a function of the number of columns, the number of columns being tested will be indicated. A detailed description and sample patterns for each printing test follows:

6.1.1 Test 0 - Data Path Test

This test is used to test the data lines to and through the interface and to the terminal. An alternating bit pattern is sent which will print alternating *'s and U's in a checkerboard pattern to the maximum column width. The starting character for each line is alternated and a total of four lines are printed.

With the Auto Line Feed Option set to produce an automatic line feed after every received carriage return, there will be a blank line between each printed line.

EXAMPLE:

```

    *U*U*U*U*U*U*U*U*U*U*U*U*U*U*U
U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*
    *U*U*U*U*U*U*U*U*U*U*U*U*U*U*U
U*U*U*U*U*U*U*U*U*U*U*U*U*U*U
  
```

6.1.2 Test 1 - Printable Character Test

This test produces a check of all 94(10) printable characters. The

6
T
a
c
a
o
h
a

T
n
p
b
a

E

E

&
W
a
n
e
a
b
c
b

E

a

characters are printed in groups of three with three groups per line, separated by three spaces between groups. The first column will contain all ASCII codes from 040 to 077. Column two will contain all ASCII codes from 100 to 137 - primarily the capital letter set. The last column will contain all ASCII codes from 140 to 176 - primarily the small letter set.

With the Auto Line Feed Option set to produce an automatic line feed after every carriage return, there will be a blank line between each printed line.

EXAM
PLE:

```

      !!!   aaa   ...
      :...  AAA   aaa
B   bbb   BB
      ###   CCC   ccc
      $$$   DDD   ddd
      %%%   E
EE   eee   FFF   fff
      &&&   GGG   ggg
      (((
HHH   hhh   )))   iii
      )))   III   iii
      ***   JJJ   jjj
      +++
KKK   kkk   ...   lll
      ---   MMM   mmm
      ...
NNN   nnn   ///   ooo
      000   PPP   ppp
      111
QQQ   qqq   222   rrr
      333   SSS   sss
      44
4   TTT   ttt   UUU   uuU
      555   VVV   vvv
      666
      7
77   www   WWW   xxx
      888   YYY   yyy
      999
:::   ZZZ   zzz
      :::   [[[
      <<<  \\\
      ===  ]]
]
      >>>
      ???
```

6.1.3 Test 2 - Non-printable Character Test

This test checks all non-printable characters that have no control function in the LA36 terminal or the LA36 options (such as CR, LF, BS,

& BEL).
First the ASCII code will be printed followed by the mnemonic
after a few separating spaces. Following the mnemonic, the actual
control character will be sent three times and nothing should happen
at the printer. This pattern is repeated, three times on a line,
until all of the non-printing characters have been tested.

With the Auto Line Feed Option set to produce an automatic
line feed after every received carriage return, there will be a blank
line between each printed line.

W
a
c
b
e
E
X
X
X
X
X
X
X

EXAMPLE:

```

006 ACK 001 SOH 002 STX
        020 DLE 021 DC1 022 DC2
        023 DC3 024 DC
4 025 NAK
        026 SYN 027 ETB 030 CAN
        031 EM 032
SUB 034 FS
        035 GS 036 RS 037 US
        177 DEL

```

6.1.4 Test 3 - Carriage Return Test

This test checks the carriage return from all even numbered columns and the spacing of the solenoid head from the left margin. It is also a good check for proper operation of the position decoder.

The test prints a full line of alternating 0's and spaces, starting with a 0. At the end of the line the print head is returned to the left margin in with a carriage return. The spaces are then filled in by spacing the print head out from the left margin to the first space, printing an 'X', and executing a carriage return. This pattern is repeated until the line is completed. Check to see that all X's are in the middle of the space between the zeroes on either side of it.

EXAMPLE:

```

XOXOX      OXOXOXOXOXOXOXOXOXOXOXOXO

```

With the Auto Line Feed Option set to produce an automatic line feed after every received carriage return, this test will print a line of 0's and spaces, then print a diagonal line of X's. To correctly check the encoder, the Auto Line Feed Option should be disabled.

EXAMPLE:

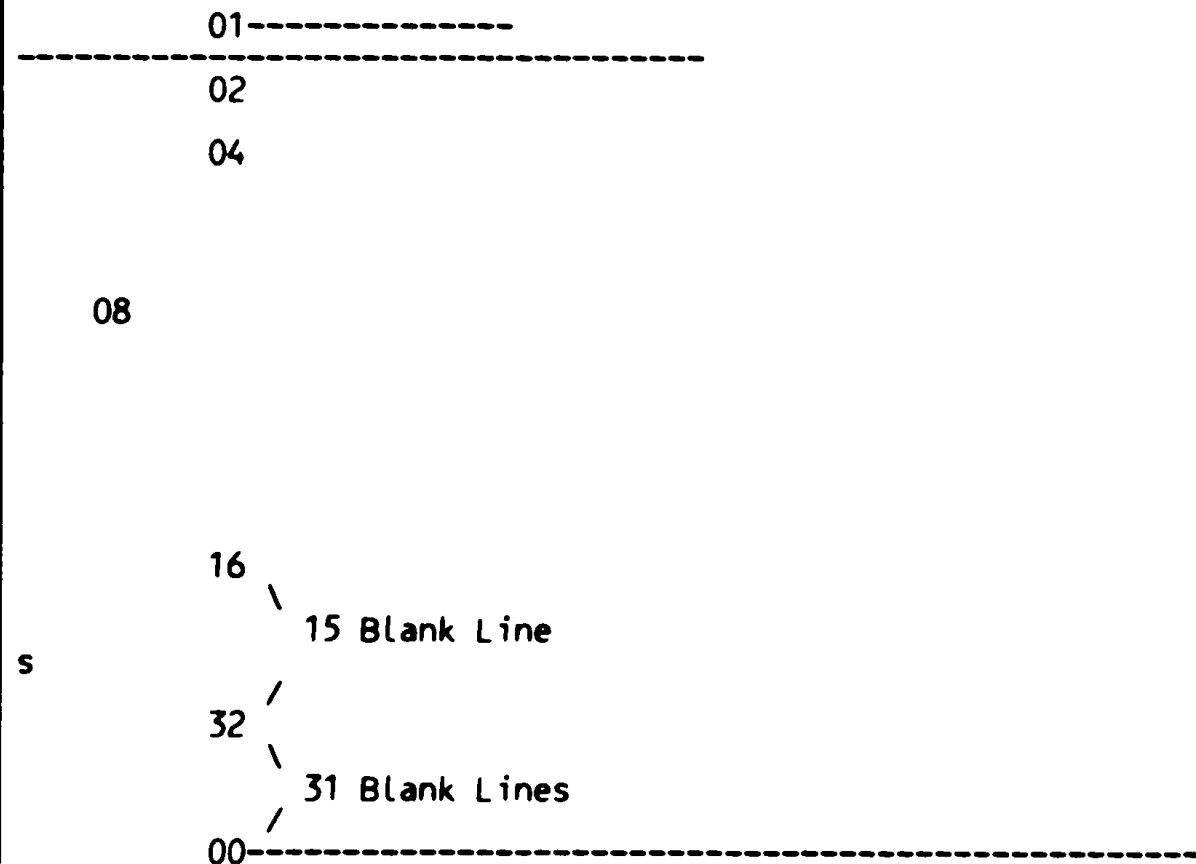
```
0 0 0 0 0 0 0 0 0 0
  X
   X
    X
     X
      X
       X
        X
         X
          X
           X
            X
```

6.1.5 Test 4 - Multiple Line Feed Test

This test checks the line feed capability of the printer by sending various groups of line feeds interspaced with reference lines. The number printed as the reference line indicates the number of line feeds that follow. The first and last lines also contain a string of dashes as reference points for measuring the total distance between the two dashed lines, i.e., 63(10) lines.

With the Auto Line Feed Option set to produce an automatic line feed after every carriage return, the number printed will indicate one less than the number of line feeds (the number of blank lines) that follow. The total distance between the two dashed lines will then be 69 lines.

EXAMPLE:



6. Th
t
la
Le
th
be
ec
'T
cc
E
mi
wi
ec
Ir
)
fc
h
te
6.
Th
si
mc
pe
ec
Il
e
av
yp
th
h1
6.
Th
ua
in
te
a
1
ch
yp
by
it
ra

6.1.6 Test 5 - Single Line Feed Test

This test is designed to check the timing of single line feeds and the capability of doing line feeds in all columns. Two reference lines are used by this test (and Test 6) which also can be used to easily check the number of columns the printer is printing.

The first reference line contains 130(10) zeroes followed by two 2's if testing 132(10) columns. If less than 132 columns, the line will contain 0's for two less than the maximum number of columns followed by the two 2's. This reference line is a quick check for 132(10) columns if testing the full 132(10) columns. The second reference line prints a string of numbers (1 to 9 & 0) repeated to the maximum column. This line, again, can be used as a quick check of the number of columns.

The line feed test is accomplished by: printing the first reference line of 0's and two 2's; then either sending 60(10) 3's, if testing 132(10) columns, or waiting 1.8 seconds for an LCV, if testing less than 132(10) columns. If testing 132(10) columns, nothing should happen, except for an LCV, at the end of the line. The 3's should be lost and never printed. After the LCV, with the print head at the extreme right, a carriage return - line feed will be sent followed by repeated backslashes '\ ' and linefeeds to print a diagonal line down the paper. When a backslash is printed in the maximum column, a carriage return will be sent immediately after the line feed and the second reference line of sequential numbers will be printed. After completing the line, a carriage return - line feed will be sent and the program will wait one second for the carriage return function to complete. After the delay, the reference line will be repeated, the last line being guaranteed to be correct. Any timing problems

during the line feeds will show as misprints or missing characters during the first 16(10) characters of the middle reference line. Also, any paper feed problems will cause misalignment of the slashes forming the diagonal line.

EXAMP
LE:

000000000000000000000000000022

123456789012345678901234567890
123456789012345678901234567890

With the Auto Line Feed Option set to produce an automatic line feed after every received carriage return, there will be a blank line every

W
a
a
6
T
C
A
A
E
k
E
n
E
t

place a
carriage return is executed.

E 3

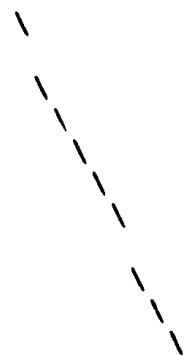
SEQ 0030

6
T
t
a
t
s
i
I
R

PAGE 18

EXAMPLE:

000000022



1234567890

1234567890

6.1.7 Test 6 - Backspace Test

This test is designed to test the print timing as in Test 5 as well as the backward and forward movement of the print solenoid head.

The test consists of the same first reference line as in Test 5 then a carriage return-line feed. A full line is then printed using the following pattern:

```
Forward Slash  '/'
Backspace
Back Slash    '\'
```

This pattern produces a line of all X's. The two slashes should cross exactly at the middle, producing the X character. When the line is completed a carriage return-line feed is sent and the last two reference lines are printed as in Test 5. Any timing problems will show in the first 16(10) characters of the middle reference line; again as in Test 5.

With the Auto Line Feed Option set
to produce an automatic line feed
after every received carriage return,
there will be a blank line
between each printed line.

EXAMPLE:

```
0000000000000000000000000000022  
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX  
123456789012345678901234567890  
123456789012345678901234567890
```

6.1.8 Test 7 - Overprint Test

This test is designed to check the spacing and repeatable printing characteristics of the printer. Three rows of characters are each overprinted two times. The rows consist of the following characters alternated across the line:

Row 1	M-SP
Row 2	SP-a
Row 3	&-SP

The resulting pattern will be a checkerboard pattern and the overprinted characters should be aligned properly with the initial characters.

EXAMPLE:

```

      M
M M M M M M M M M M M M M M
      a a a a a a a a a a a a a a
& & & & & & & & & & & & & & &

```

With the Auto Line Feed Option set to produce an automatic line feed after every received carriage return, the lines will not be overprinted. There will be three lines of each character with a blank line between each group of characters. The characters in each group should be in the same columns.

EXAMPLE:

```

      M M M M M M M M M M
M M M M M M M M M M
      M M M M M M M M M M
a a a a      a a a a a
a a a a      a a a a a a a a a
              a a a a a a a a a

```

⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘
⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘
⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘
⌘

C
SI

6.1.9 Test 10 - Printing Frequency Sweep Test

This test prints the character 'H' repeatedly, 30(10) characters per line for four lines. During the first two lines, the time interval between characters is increased from 30(10) milliseconds to 1.8 seconds using the following formula to create a logarithmic increase:

$$\text{New Delay} = \text{Old Delay} + \text{Old Delay}/16 + \text{Old Delay}/128$$

The last two lines do just the reverse. The time interval between characters is decreased from 1.8 seconds to 30(10) milliseconds using the following formula to again create a logarithmic decrease:

$$\text{New Delay} = \text{Old Delay} - \text{Old Delay}/16 - \text{Old Delay}/128$$

Look for possible misalignment of the characters or spaces between characters as an indication of timing problems.

With the Auto Line Feed Option set to produce an automatic line feed after every received carriage return, there will be a blank line between each printed line.

EXAMPLE:

```

              HHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHH
H           HHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHH
              HHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHH

```

6.1.10 Test 11 - Ribbon Feed Test

This test checks the ribbon feed mechanism by printing a single column of 24 lines of X's down the left hand margin of the page. Visually check for proper operation of the ribbon feed mechanism during this test.

With the
Auto Line Feed Option set to produce an automatic line feed
after every received carriage return, there will be a blank line
between each printed line.

EXAMPLE:

X
X
X
X
X
X
X

6.1.11 Test 12 - Printer Bell Test

This test checks the printer bell buffer to insure that eight bells are distinctly heard, even when sent at the maximum transfer rate. The program sends 8 bell codes at the maximum rate to the printer then waits 2.5 seconds to allow the operator to hear the bells.

6.1.12 Test 17 - Life Test

This test runs continuously and is run as an individual, special test. It is not part of the standard printing test sequence.

This test prints 2 lines of each printable character and then repeats continuously. The second line of each character is overprinted 4 times to conserve paper. At the end of each complete pass through the character set, a message is printed indicating the number of passes executed. If any character (except 'Rubout') is typed on the keyboard during this test, the pattern will change and restart with the typed character. This will only happen if keyboard control is in use.

EXAMPLE:

AAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AA
BBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBB

If the Auto Line Feed Option is set to produce an automatic line feed after every received carriage return, the test will print six lines of each character with a blank line between the first and second lines as well as between each group of characters.

EXAMPLE:

AAAAAAAAAAAAA

AAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAA

BBBBBBBBBBBBBBBB

BBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBB

BBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBB

CZ
SY

6.2 ECHO TESTS

These

tests are designed as a test of the keyboard and an aid in isolating troubles within the terminal. At the beginning of each test, the test number will

be printed indicating which test is being executed. Typing a 'RUBOUT' or

'DELETE' at any time, whether in keyboard control or not, will exit the current

Echo test and print a test termination message. If in keyboard control, the select test message

will be printed and the program will await a test selection as usual.

In switch register control, the program will halt (at SELLHLT) waiting

for control via the switch register. A detailed description of each

test follows:

6.2.1 Test 20 - Character Echo Test

This test is designed to operate the terminal in a simulated local mode. Any character typed on the keyboard (except a 'rubout') will be echoed to the printer.

If the LA36 terminal is in half duplex with the Auto Line Feed Option available, typing a carriage return may cause a garbled response on the terminal during this test.

6.2.2 Test 21 - Line Echo Test, Fast Rate

This test continually sends full lines of any character up to the maximum column width. The test prints a '0' character when started until a key is typed on the keyboard.

The program will then send the typed character until another character is typed or the test is terminated by typing a 'rubout'. The characters are transmitted at the maximum rate with a carriage return-line feed inserted after

every 132(10)
printable characters.

If the LA36 is in half duplex when running this test, characters may be lost or garbled whenever a character is typed on the keyboard.

With the Auto Line Feed Option set to produce an automatic line feed after every carriage return, there will be a blank line between each printed line.

6.2.3 Test 22 - Line Echo Test, Slow Rate

This test is identical to Test 21 except a delay of 1.8 seconds is inserted between each character to allow the print head to perform an LCV between characters.

6.2.4 Test 23 - Character/Code Echo Test

This test will print the octal code received by the processor followed by the character or the mnemonic of the character every time a key is pressed on the keyboard. The parity of the received code will be indicated as either odd or even. Allow sufficient time between characters for the line to be printed.

With the Auto Line Feed Option set to produce an automatic line feed after every received carriage return, there will be a blank line between each printed line.

EXAMPLE:

	301	A	ODD
	263	3	ODD
	215	CR	E
VEN	240	SP	EVEN

6.2.5 Test 24 - Selected Pattern Echo Test

This test is designed to give maintenance the flexibility to choose their own patterns for isolating any specific problems which may arise in the field.

Type any characters (except control-C and rubout) and each character will be echoed as typed. A maximum of 256(10) characters may be inputted. No carriage returns or line feeds are inserted by the program, all characters must be inputted by the operator. To terminate the input string type a control-C, the program will then continually echo the inputted pattern. To stop the printing, type control-C. The program will stop printing the pattern and will wait for either another pattern input terminated by a control-C, or the same pattern may be used again by typing control-C. To exit the test at any time, type a 'rubout'.

When any options are available, be careful what characters or character sequences are selected.

6.2.6 Test 25 - Bell Echo Test

This test is designed to test the bell on column 64 if typing has occurred on the line. The test prints a message:

```
TYPE ANY PRINTABLE CHARACTER AND LISTEN FOR BELL .....
```

After the test message is printed, type any printable character on the keyboard. The character will be echoed and the bell should ring. The message will then be typed again. Type the 'rubout' key to terminate the test at any time.

6.4 STANDARD I/O TESTS

These tests are designed as a brief check of the console terminal interface logic. Each check is structured as an independent test and the switch register controls may be used. A description of each test is given in the program listing.

Any errors encountered during the I/O tests will cause a halt at location 'ER RHLT' if switch 14 is down.

1-	1900	SWITCH REGISTER OPTIONS
2-	4500	SPECIAL OPERATIONAL INFORMATION
3-	6700	SYSTEM EQUATES
4-	13500	TRAP CATCHER & STARTING ADDRESSES
5-	19100	SYMBOL DEFINITIONS
7-	100	PROGRAM INITIALIZATION & CONTROL
13-	46300	COMMON ROUTINES USED BY LA36 TESTS
26-	100	I/O LOGIC TESTS
42-	100	LA36 PRINTER TESTS
55-	100	LA36 ECHO TESTS
62-	100	MISC. DIAGNOSTIC MESSAGES

400
500
600
700
800
900
1000
1100
1200
1300
1400
1500
1600
1700
1800
1900
2000
2100
2200
2300
2400
2500
2600
2700
2800
2900
3000
3100
3200
3300
3400
3500
3600
3700
3800
3900
4000
4100
4200
4300

.TITLE CZLACEO LA36 TERM (DL11 & KL11)

:LA36 DIAGNOSTIC (DL11 & KL11 INTERFACE)

:AUTHORS: ROBERT W. BAKER
R. QUENNEVILLE
RALPH A. SCHAUER
JOHN V. CHATALIAN

:COPYRIGHT 1974,1977,1978 DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754

.SBTTL SWITCH REGISTER OPTIONS

SWITCH	POSITION	FUNCTION
15	UP (1)	HALT AT COMPLETION OF CURRENT TEST
	DOWN (0)	CONTINUE NORMAL TEST SEQUENCE
14	UP (1)	CONTINUE ON ERROR
	DOWN (0)	HALT ON ERROR
13	UP (1)	DRIVE ONLY CONSOLE TERMINAL
	DOWN (0)	DRIVE ALL TERMINALS
11	UP (1)	LOOP ON INDIVIDUAL TEST
	DOWN (0)	NORMAL TEST SEQUENCE
09	UP (1)	CPU TYPE IS AN LSI-11, PDP-11/03
	DOWN (0)	ALL OTHER PDP-11 CPU'S
08	UP (1)	HALT TO SELECT TEST AT END OF CURRENT TEST
	DOWN (0)	LOOP ON TEST SEQUENCE
05-00		TEST # SELECTION
07-00		# OF COLUMNS AT START-UP

4500
4600
4700
4800
4900
5000
5100
5200
5300
5400
5500
5600
5700
5800
5900
6000
6100
6200
6300
6400
6500

.SBTTL SPECIAL OPERATIONAL INFORMATION

- :1.-- THE STANDARD CONSOLE TERMINAL INTERRUPT VECTOR AND REGISTER ADDRESSES ARE USED. TO REDEFINE THE LOCATION OF THE CONSOLE TERMINAL THE SYMBOLIC LOCATIONS 'CONADD' AND 'CONVEC' SHOULD BE CHANGED BEFORE START UP.
- :2.-- BEFORE START UP REFER TO THE DESCRIPTION OF THE ROUTINE 'DLY'. TIMING IS A FUNCTION OF THE PDP-11 MODEL AND MEMORY TYPE AND SHOULD BE SET UP BEFORE RUNNING THE DIAGNOSTIC.
- :3.-- IF CPU IS A PDP-11/03 , LSI-11 SET SWITCH REGISTER BIT 09 TO A 1. SPECIAL TESTS ARE RUN ON THE DLV11.
- :4.-- SYSTEMS WITHOUT A HARDWARE SWITCH REGISTER SHOULD USE MEMORY LOCATION 176 AS A SOFTWARE SWITCH REGISTER.
- :5.-- THIS DIAGNOSTIC IS FOR VERIFICATION OF BASIC TERMINAL FUNCTIONS ONLY. IF THE TERMINAL UNDER TEST HAS HARDWARE OPTIONS INSTALLED RUN DIAGNOSTIC MAINDEC-11-DZLAF-A, THE LA36 TERMINAL OPTIONS TEST.


```

6700          .SBTTL  SYSTEM EQUATES
6800
6900          :
7000          :
7100          :
7200          000000      R0=%0
7300          000001      R1=%1
7400          000002      R2=%2
7500          000003      R3=%3
7600          000004      R4=%4
7700          000005      R5=%5
7800          000006      SP=%6
7900          000007      PC=%7
8000          177776      PSW=177776
8100          :
8200          :
8300          :
8400          000001      BIT0=1
8500          000002      BIT1=2
8600          000004      BIT2=4
8700          000010      BIT3=10
8800          000020      BIT4=20
8900          000040      BIT5=40
9000          000100      BIT6=100
9100          000200      BIT7=200
9200          000400      BIT8=400
9300          001000      BIT9=1000
9400          002000      BIT10=2000
9500          004000      BIT11=4000
9600          010000      BIT12=10000
9700          020000      BIT13=20000
9800          040000      BIT14=40000
9900          100000      BIT15=100000
10000         000000      OPEN=0
10100         040000      SCOPSW=BIT14      ;SCOPE SWITCH
10200         004000      NITRSW=BIT11     ;TEST LOOP SWITCH
10300         005726      POPSP=5726       ;POP STACK ONCE
10400         022626      POPSP2=22626     ;POP STACK TWICE
10500         000340      PRTY7=340        ;PRIORITY LEVEL DEFINITIONS
10600         000200      PRTY4=200
10700         000200      ACRLF=200
10800         001000      LSI11=BIT9       ;FLAG FOR LSI-11,11/03
10900         :
11000         :
11100         :
11200         104000      TYPE=EMT+0
11300         104001      ERROR=EMT+1
11400         104002      EHALT=EMT+2
11500         104003      STRDRV=EMT+3
11600         104004      STPCHV=EMT+4
11700         104005      CHAIN=EMT+5
11800         104006      CHALT=EMT+6
11900         104007      TYPEM=EMT+7
12000         104010      DELAY=EMT+10
12100         104011      TTYCTL=EMT+11
12200         104012      CRLF=EMT+12
12300         104013      SCRLF=EMT+13
    
```

12400	104014	LF=EMT+14
12500	104015	PRINTC=EMT+15
12600	104016	PRTHDR=EMT+16
12700	104017	PRNT=EMT+17
12800	104020	READ=EMT+20
12900	104021	AREAD=EMT+21
13000	104022	CR=EMT+22
13100	104023	BTOASC=EMT+23
13200	104024	FORWD=EMT+24
13300	104025	READC=EMT+25

```

13500      .SBTTL  TRAP CATCHER & STARTING ADDRESSES
13600      ;
13700 000000      .ENABL  ABS,AMA
13800 000000      .ASECT
13900
14000
14100      000000      .=0
14200
14300 000000 000002      .+2      ;UNASSIGNED TRAP
14400 000002 000000      HALT
14500 000004 000006      MACHER: .+2      ;SP OVERFLOW, BUS ERROR TRAP
14600 000006 000000      HALT
14700 000010 000012      .+2      ;RESERVED INSTRUCTION TRAP
14800 000012 000000      HALT
14900 000014 000016      .+2      ;TRACE TRAP
15000 000016 000000      HALT
15100 000022 000022      .+2      ;TRAP TO CALL IOX
15200 000022 000000      HALT
15300 000024 000026      .+2      ;POWER FAIL TRAP
15400 000026 000000      HALT
15500 000030 002722      EMTINT      ;EMT TRAP
15600 000032 000340      PRTY7
16300
16400      000042      .=42
16500
16600 000042 000000      0
16700
16800      000046      .=46
16900
17000 000046 011522      LOGICAL
17100
17200      000052      .=52
17300
17400 000052 010000      010000
17500
17600      000174      .=174
17700
17800 000174 000000      DISPREG: .WORD 0      ;SOFTWARE DISPLAY
17900 000176 000000      SWREG:   .WORD 0      ;SOFTWARE SWITCH REGISTER
18000
18100 000200 000167 000604      JMP      START      ;START UP WITH I/O TESTS RUNNING
18200 000204 000167 000526      JMP      START1     ;START UP, SKIP ALL I/O TESTS
18300 000210 000167 000540      JMP      START2     ;START UP TERMINAL CONTROL WITH I/O TESTS
18400 000214 000167 000552      JMP      START3     ;START UP TERMINAL CONTROL WITHOUT I/O TESTS
18500
18600
18700      000600      .=600
18800
18900 000600 000000      SPBOT: 0      ;BOTTOM OF STACK
    
```

SYMBOL DEFINITIONS

.SBTTL SYMBOL DEFINITIONS

19100				
19200			:	
19300			:	
19400			:	
19500	000602	177560	CONADD: 177560	:ADDR OF CONSOLE RECEIVER STATUS REG
19600	000604	000060	CONVEC: 60	:CONSOLE TERMINAL INTERRUPT VECTOR
19700	000606	176500	DLADR: 176500	:ADDRESS OF FIRST DL11, DEFAULT TO DL11-A,B
19800				:IF DL11-C,D,E,, THEN
19900				:SET TO 175610 FOR FIRST 16 (OF 31) OR
20000				:SET TO 176000 FOR LAST 16 (OF 31)
20100				:OR SET OTHER DESIRED START ADDRESS
20200	000610	000020	DLNR: 16	:# OF DL11'S TO BE INITIALLY ASSUMED
20300	000612	177560	TKS: 177560	:CONSOLE RECEIVER STATUS REG
20400	000614	177562	TKB: 177562	:CONSOLE RECEIVER BUFFER
20500	000616	177564	TPS: 177564	:CONSOLE TRANSMITTER STATUS REG
20600	000620	177566	TPB: 177566	:CONSOLE TRANSMITTER BUFFER
20700	000622	000060	TKVTR: 60	:C.T. RECEIVER INTERRUPT VECTOR
20800	000624	000200	TKLVL: PRTY4	:C.T. RECEIVER PRIORITY LEVEL
20900	000626	000064	TPVTR: 64	:C.T. TRANSMITTER INTERRUPT VECTOR
21000	000630	000200	TPLVL: PRTY4	:C.T. TRANSMITTER PRIORITY LEVEL
21100	000632	000000	FSTDL: OPEN	:ADDRESS OF FIRST ACTIVE DL11
21200	000634	000000	CNTLSW: OPEN	:CONSOLE TERMINAL CONTROL SWITCH
21300	000636	000000	RTNNO: OPEN	:CONTAINS CURRENT TEST NUMBER
21400	000640	000000	NXTST: OPEN	:CONTAINS ADDRESS OF NEXT TEST
21500	000642	000000	SCOPTR: OPEN	:CONTAINS ADDRESS OF TEST SCOPE ENTRY
21600	000644	000000	PRGID: OPEN	:CONTAINS TEST PROGRAM INDICATORS
21700	000646	000000	CRBUF: OPEN	
21800	000650	000000	CTRA: OPEN	
21900	000652	000000	WIDTH: OPEN	:CURRENT PAPER WIDTH, BINARY
22000	000654	000000	LEVEL: OPEN	:LEVEL OF EXECUTION
22100	000656	000000	DLCNT: OPEN	:# OF MULTIPLE DL11S
22200	000660	000000	ICTR: OPEN	:I/O TEST ITERATION COUNT
22300	000662	000000	REPT: OPEN	:TEMP STORAGE FOR TESTS E021 & E022
22400	000664	000000	BRCTR: OPEN	:COUNTER FOR ROUTINE 'AREAD'
22500	000666	000000	COUNT3: OPEN	:COUNTER FOR ROUTINE 'PRINTC'
22600	000670	000000	XCSR: OPEN	:ADDRESS OF MULTIPLE DL11 STATUS
22700	000672	000251	TIMER: 251	:1 MSEC COUNTER FOR ROUTINE 'DELAY'
22800	000674	000000	SPCNT: OPEN	:COUNTER FOR TEST ROUTINE 'PT3'
22900	000676	000000	CURTST: OPEN	:ADDRESS OF CURRENT TEST
23000	000700	000000	TEMPCH: OPEN	:TEMP STOR FOR ECHO TESTS
23100	000702	000000	PARITY: OPEN	:PARITY FLAG FOR RECEIVED CHAR
23200	000704	000000	PCHAR: OPEN	:CHAR CODE WITH PARITY BIT
23300	000706	000000	LCFNT: OPEN	:COUNTER FOR TEST ROUTINE 'PT4'
23400	000710	000000	INCHK: OPEN	:CHECK FOR INPUT FLAG
23500	000712	000000	TEMP: OPEN	:TEMPORARY WORKING STORAGE
23600	000714	177570	SR: 177570	:SW REG ADDRESS
23700	000716	000000	CNTR: OPEN	:TIME COUNTER FOR LSI-11 TESTS

23900

CZ
CC

```

100                                     .SBTTL PROGRAM INITIALIZATION & CONTROL
200
300                                     :*****
400                                     :COMMON HALT---WHEN IN SWITCH REGISTER CONTROL THE CPU
500                                     :WILL BE ADVANCED TO THIS COMMON HALT WHERE
600                                     :A NEW TEST WILL BE EXPECTED TO BE STARTED
700                                     :*****
800
900 000720 005767 177730  CHLT:  TST      LEVEL      :TEST CURRENT LEVEL
1000 000724 001403        BEQ      SELHLT     :BRANCH IF 0, DO NOT HALT
1100 000726 011600        MOV      @SP,R0    :PUT ADDRESS OF CALLER INTO R0
1200 000730 005740        TST      -(R0)
1300 000732 000000        HALT
1400 000734 000002        SELHLT: RTI      :RETURN FROM INTERRUPT
1500 000736 012767 177777 004324  START1: MOV      #177777,ATOX  :FORCE END OF I/O TESTS
1600 000744 012767 104006 000726        MOV      #CHALT,WAITF  :FORCE SR CONTROL
1700 000752 000424        BR        STARTX
1800 000754 012767 104011 000716  START2: MOV      #TTYCTL,WAITF  :FORCE TERMINAL CONTROL
1900 000762 012767 005320 004300        MOV      #AT1,ATOX    :FORCE ALL I/O TESTS
2000 000770 000415        BR        STARTX
2100 000772 012767 104011 000700  START3: MOV      #TTYCTL,WAITF  :FORCE TERMINAL CONTROL
2200 001000 012767 177777 004262        MOV      #177777,ATOX  :FORCE END OF I/O TESTS
2300 001006 000406        BR        STARTX
2400 001010 012767 005320 004252  START:  MOV      #AT1,ATOX    :FORCE ALL I/O TESTS
2500 001016 012767 104006 000654        MOV      #CHALT,WAITF  :FORCE SR CONTROL
2600 001024 012706 000600        STARTX: MOV      #SPBOT,SP    :SET STACK POINTER
2700 001030 016746 176752        MOV      6,-(SP)      :SAVE CURRENT VECTOR
2800 001034 016746 176744        MOV      4,-(SP)
2900 001040 012767 001054 176736        MOV      #10$,4      :SET UP TIMEOUT VECTOR
3000 001046 005777 177642        TST      @SR          :TRY TO REFERENCE HARDWARE SW REG
3100 001052 000411        BR        11$        :BRANCH IF NO TIMEOUT TRAP OCCURS
3200 001054 012767 000176 177632  10$:  MOV      #SWREG,SR    :POINT TO SOFTWARE SWITCH REGISTER
3300 001062 022626        CMP      (SP)+,(SP)+  :RESTORE STACK
3400 001064 104000        TYPE
3500 001066 014424        NOSWR
3600 001070 012767 000202 177574        MOV      #202,TIMER   :SWITCH REG AT LOC 176
3700 001076 012667 176702  11$:  MOV      (SP)+,4      :ADJUST TIMER FOR LSI-11
3800 001102 012667 176700        MOV      (SP)+,6      :RESTORE TIMEOUT VECTOR
3900 001106 005067 177576        CLR      INCHK       :ALLOW INPUT CHECKING
4000 001112 012767 000006 176664        MOV      #6,MACHER   :CLEAN UP
4100 001120 005067 177520        CLR      PRGID       :INITIALIZE PROGRAM FLAGS
4200 001124 005067 177504        CLR      CNTLSW      :INITIALIZE TERMINAL CONTROL SWITCH
4300 001130 005067 177520        CLR      LEVEL       :INITIALIZE LEVEL
4400 001134 012767 003460 176662        MOV      #PFAIL,24   :SET ADDR POWER FAIL ROUTINE
4500 001142 004767 002542        JSR      PC,CONIT    :SET UP CONSOLE TERMINAL ADDRESS

```

4700
 4800
 4900
 5000
 5100
 5200
 5300
 5400
 5500
 5600 001146 017701 177542
 5700 001152 042701 177400
 5800 001156 020127 000204
 5900 001162 003003
 6000 001164 020127 000035
 6100 001170 101002
 6200 001172 012701 000204
 6300 001176 010167 177450
 6400 001202 012700 014127
 6500 001206 012702 000003
 6600 001212 104023
 6700 001214 000401
 6800 001216 000410
 6900 001220 012700 000000
 7000 001224 104015
 7100 001226 104007
 7200 001230 013676
 7300 001232 012767 000240 177754

```

*****
:READ THE PAPER WIDTH, NUMBER OF COLUMNS,
:FROM SWITCH REGISTER POSITIONS 0-7. SAVE AND
:CONVERT TO 3 ASCII CHARACTERS. A WIDTH GT132
:OR LT30 COLUMNS (DECIMAL) WILL BE ABORTED TO 132.
:THE SWITCHES MAY BE CHANGED ONCE THE PROGRAM TITLE OR THE DL11 COUNT
:MESSAGE HAS STARTED TO PRINT.
*****
MOV @SR,R1 ;PUT (SR) INTO R1
BIC #177400,R1 ;SAVE ONLY BITS 0-7
CMP R1,#204 ;TEST NO. COLUMN GT132
BGT 2$ ;COLUMNS GT132, DEFAULT TO 132
1$: CMP R1,#35 ;CHECK IF NO. COLUMNS LT 30
BHI 3$ ;NOT LT 30 NOR GT 132
2$: MOV #204,R1 ;COLUMNS LT 30 OR GT 132, DEFAULT
3$: MOV R1,WIDTH ;SAVE NO. COLUMNS IN WIDTH
MOV #HDRO,R0 ;ADDR TO STORE ASCII COLUMN VALUE
MOV #3,R2 ;DO A 3 CHAR. CONVERSION
BTOASC ;CONVERT NO. COLUMNS TO ASCII
4$: BR 5$
BR 6$
5$: MOV #0,R0 ;TRANSMIT A
PRINTC ;NULL CODE
TYPEM ;TYPE PROGRAM TITLE FIRST TIME RUN
STARTM
MOV #NOP,4$

```

```

7500
7600
7700
7800
7900
8000
8100
8200 001240 012767 001320 176536 6$:  MOV #END2A,MACHER ;INITIALIZE TIME OUT TRAP
8300 001246 016700 177334      MOV DLADR,R0 ;ADDRESS OF FIRST DL11 TO R0
8400 001252 016701 177332      MOV DLNR,R1 ;SET DL CHECK COUNT
8500 001256 005067 177374      CLR DLCNT ;INITIALIZE DLCNT
8600 001262 005710      END3: TST (R0) ;IS DL PRESENT?
8700 001264 012767 001332 176512      MOV #END2,MACHER ;YES, RESET TIME OUT TRAP
8800 001272 010067 177334      MOV R0,FSTDL ;STORE ADDRESS OF FIRST DL11
8900 001276 000401      BR 2$ ;CONTINUE
9000 001300 005710      1$: TST (R0) ;IS DL11 PRESENT
9100 001302 062700 000010      2$: ADD #10,R0 ;POINTER AND DL11 ADDRESS
9200 001306 005267 177344      INC DLCNT ;INCREMENT COUNT OF DL11'S
9300 001312 005301      DEC R1 ;DECREMENT DL CHECK COUNT, DONE?
9400 001314 001407      BEQ END4 ;BRANCH IF DONE
9500 001316 000770      BR 1$ ;CHECK PRESENCE OF NEXT DL11
9600 001320 005301      END2A: DEC R1 ;DONE DL CHECK?
9700 001322 001404      BEQ END4 ;YES, EXIT
9800 001324 0062700 000010      ADD #10,R0 ;NO, CHECK NEXT DL
9900 001330 000754      BR END3 ;CONTINUE
10000 001332 022626      END2: POPSP2 ;DL11 NOT PRESENT
10100 001334 016701 177316      END4: MOV DLCNT,R1 ;GET # DL11'S
10200 001340 012700 014064      MOV #DL11S1,R0 ;ADR OF ASCII CHAR STORAGE
10300 001344 012702 000002      MOV #2,R2 ;# OF ASCII CHARS
10400 001350 104023      BTOASC ;CONVERT NUMBER
10500 001352 104007      TYPEN ;TYPE MESSAGE
10600 001354 014051      DL11S
10700
10800
10900
11000
11100
11200
11300
11400 001356 005067 177254      CLR RTNNO ;SET ROUTINE NO = 0
11500 001362 005067 177266      CLR LEVEL ;SET LEVEL = 0
11600 001366 026727 003676 177777      CMP ATOX,#177777 ;SEE IF I/O IS TO BE SKIPPED
11700 001374 001515      BEQ SKIP
11800 001376 012767 005266 177234      MOV #ATO,NXTST ;ADDRESS OF FIRST I/O TEST
11900 001404 104024      FORWD ;SET UP TEST PARAMETERS
12000 001406 000177 177264      JMP @CURTST ;GO TO I/O TEST ROUTINE

```

:*****
:THIS NEXT PART CHECKS THE PRESENCE OF DL11-A OR DL11-C
:STARTING AT 776500. A MESSAGE WILL BE PRINTED INDICATING THE NUMBER
:PRESENT. THE PRINTER DIAGNOSTIC WILL ADDRESS EACH OF
:THE MULTIPLE DL11S IN THE SYSTEM IF SWITCH 13 IS DOWN (0).
:*****

END3:

1\$:

2\$:

END2A:

END2:

END4:

:*****
:EXECUTE THE STRING OF CONSOLE TERMINAL I/O TESTS
:THEN EITHER HALT AT LOCATION SELHLT OR CONTINUE WITH
:PRINTER TESTS AS A FUNCTION OF SR BIT 8.
:*****

12200
12300
12400
12500
12600
12700
12800
12900
13000
13100
13200
13300
13400
13500
13600
13700
13800
13900
14000
14100
14200
14300
14400
14500
14600
14700
14800
14900
15000
15100
15200
15300
15400
15500
15600
15700
15800
15900
16000
16100
16200
16300
16400
16500
16600
16700
16800
16900
17000
17100
17200
17300
17400
17500
17600
17700
17800

001412 032767 000001 177214
001420 001401
001422 104011
001424 005767 177214
001430 100016
001432 032777 040000 177254
001440 001407
001442 022767 177777 177172
001450 001403
001452 017716 177164
001456 000002
001460 042767 100000 177156
001466 005767 177162
001472 001405
001474 032777 004000 177212
001502 001405
001504 000002
001506 005367 177146
001512 001407
001514 000002
001516 032777 000400 177170
001524 001402
001526 000167 000146
001532 022626
001534 000240
001536 005777 177152
001542 100003
001544 116700 177066
001550 000000
001552 005767 177076
001556 001420

CHAINN-- THIS PORTION IS THE COMMON RETURN
FOR ALL THREE CLASSES OF TESTS.

1--IF AN ERROR OCCURRED DURING AN I/O TEST THE
OPERATOR CAN CAUSE THAT TEST TO BE LOOPED
WITHOUT ANY FURTHER ERROR HALTS BY
SETTING THE 'SCOPE' BIT (#14) ON THE SR=1.
RESETTING SR BIT 14 TO 0 WILL ALLOW THE
ERROR HALT TO OCCUR AGAIN IF IT STILL EXISTS.

2--IF THE OPERATOR IS IN THE MAINTENANCE
MODE (BIT 8 SET = 1 AT START UP TIME), THE
SELECTED PROGRAM WILL LOOP CONTINUOUSLY
IF SR BIT 11 IS SET=1. IF BIT 11 IS = 0
THEN THE PROGRAM WILL BE ADVANCED TO
THE NEXT TEST IN IT'S CLASS IF BIT 8=0.
AS LONG AS BIT 11 AND
BIT 8 ARE 0, THE CLASS OF TESTS SELECTED
WILL BE CONTINUOUSLY SEQUENCED THROUGH.
IF BIT 11 IS 0 AND BIT 8=1, THEN THE CPU
WILL HALT AT LOCATION SELHLT AND WAIT FOR THE
NEXT TEST NUMBER TO BE SET IN THE
SWITCH REGISTER.

CHAINN: BIT #1,CNTLSW ;CHECK IF TERMINAL CONTROL
BEQ 1\$;BRANCH IF NOT
TTYCTL ;GO TO TERMINAL CONTROL
1\$: TST PRGID ;TEST ERROR BIT IN PRGID
BPL 3\$;BRANCH IF ERROR BIT NOT SET
BIT #SCOPSW,@SR ;ERROR, CHECK IF SCOPE OPTION ON
BEQ 2\$;BRANCH IF NO SCOPING
CMP #-1,SCOPTR ;YES, CHECK IF OK TO SCOPE THIS TEST
BEQ 2\$;BRANCH IF NOT OK
MOV @SCOPTR,@SP ;PUT ADDR OF SCOPE ENTRY INTO STACK
RTI ;GO TO SCOPE ENTRY IN TEST
2\$: BIC #BIT15,PRGID ;CLEAR ERROR IND. IN PRGID
3\$: TST LEVEL ;CHECK LEVEL
BEQ 4\$;BRANCH IF LEVEL=0
BIT #NITRSW,@SR ;TEST LOOP SWITCH ON (=1)
BEQ 5\$;BRANCH IF NO LOOP TEST
RTI ;GO BACK TO TEST
4\$: DEC ICTR ;DECREMENT TEST ITERATION COUNT
BEQ 6\$;BRANCH IF COUNT=0
RTI ;NOT ZERO, REPEAT TEST
5\$: BIT #BIT8,@SR ;TEST IF SEQUENCE TEST (BIT8)
BEQ 6\$;BRANCH TO NEXT TEST IF BIT8=0
JMP WAITF ;GO WAIT FOR MORE INPUT
6\$: POPSP2 ;POP 2 OFF STACK
CHAINY: NOP ;THIS FORMERLY WAS RESET
TST @SR ;CHECK SR
BPL 1\$;BRANCH IF NO HALT WANTED
MOVB RTNNO,R0 ;CURRENT TEST NUMBER TO R0
HALT ;HALT (NOT FOR TEST SELECTION)
1\$: TST LEVEL ;TEST THE CURRENT LEVEL
BEQ 3\$;BRANCH IF 0

```

17900 001560 012767 000006 176216      MOV      #6,MACHER      ;CLEAN UP
18000 001566 012706 000600              MOV      #SPBOT,SP      ;SET UP STACK POINTER
18100 001572 104024              FORWD                    ;SET UP VALUES FOR NEXT TEST
18200 001574 022767 177777 177036      CMP      #-1,NXTST      ;END OF I/O TESTS (=-1)
18300 001602 001004              BNE      2$              ;BRANCH IF NOT END
18400 001604 012767 005266 177026      MOV      #A0,NXTST      ;RESET NXTST TO FIRST I/O TEST
18500 001612 104024              FORWD                    ;SET UP VALUES FOR NEXT TEST
18600 001614 000177 177056      2$:      JMP      @CURTST         ;GO TO TEST
18700 001620 022767 177777 177012      3$:      CMP      #-1,NXTST      ;END OF I/O TESTS (=-1)
18800 001626 001012              BNE      NEXT           ;BRANCH IF NOT
18900 001630 032777 000400 177056      SKIP:   BIT      #BIT8,@SR ;TEST IF WANT TEST SELECTION RIGHT AWAY
19000 001636 001016              BNE      NEXT1          ;BRANCH IF NOT
19100 001640 052767 000200 176776      BIS      #BIT7,PRGID    ;BYPASS SCOPING
19200 001646 012767 007372 176764      MOV      #PT0,NXTST     ;PROD TESTING, GO TO PRINTER TESTS
19300 001654 012767 000006 176122      NEXT:   MOV      #6,MACHER ;CLEAN UP
19400 001662 012706 000600              MOV      #SPBOT,SP      ;SET UP STACK POINTER
19500 001666 104024              FORWD                    ;SET UP NEXT TEST PARAMETERS
19600 001670 000177 177002              JMP      @CURTST         ;GO TO ROUTINE
19700 001674 005267 176754      NEXT1:  INC      LEVEL

19800
19900
20000      ;*****
20100      ;WAIT FOR FURTHER INSTRUCTIONS:
20200      ;      -LOAD PROGRAM NUMBER INTO BITS 0-5 OF THE SR
20300      ;      -SET SR BIT 11=1 TO LOOP ON SELECTED TEST
20400      ;      -SET SR BIT 11=0 AND BIT 8=0 TO LOOP THROUGH
20500      ;      SEQUENCE OF SELECTED TESTS.
20600      ;      -SET SR BIT 11=0 AND BIT 8=1 TO HALT AGAIN AFTER
20700      ;      EXECUTING TEST ONCE
20800      ;*****
20900 001700 104006      WAITF:  CHALT          ;OR TTYCTL IF START WAS AT 210
21000 001702 012767 000006 176074      MOV      #6,MACHER      ;CLEAN UP
21100 001710 012706 000600              MOV      #SPBOT,SP      ;SET UP STACK POINTER
21200 001714 017700 176774              MOV      @SR,R0         ;GET CURRENT SW REG
21300 001720 042700 177700              BIL     #177700,R0
21400 001724 020027 000037              CMP      R0,#37         ;TEST IF PROG NO. IS I/O TEST
21500 001730 101403              BLOS    1$              ;BRANCH IF EQ OR LT 37. AN ECHO OR PRINTER
21600 001732 005067 176706              CLR     PRGID           ;I/O TEST, CLEAR PRGID
21700 001736 000403              BR      2$
21800 001740 052767 000200 176676      1$:     BIS      #BIT7,PRGID ;BYPASS SCOPING
21900 001746 000241      2$:     CLC
22000 001750 006100              ROL     R0              ;GET PROGRAM ADDRESS OUT OF
22100 001752 016067 002522 176660      MOV     PRGTAB(R0),NXTST ;PROGRAM ADDRESS TABLE
22200 001760 026727 176654 001700      CMP     NXTST,#WAITF    ;TEST IF LEGAL TEST NO.
22300 001766 001744              BEQ     WAITF           ;BRANCH IF ILLEGAL
22400 001770 104024              FORWD
22500 001772 000177 176700              JMP     @CURTST         ;SET UP TEST PARAMETERS
                                ;GO TO TEST

```

```

22700
22800
22900
23000
23100
23200
23300
23400
23500
23600
23700
23800
23900
24000 001776 022626
24100 002000 105777 176606
24200 002004 100013
24300 002006 017705 176602
24400 002012 042705 177600
24500 002016 020527 000177
24600 002022 001004
24700 002024 042767 004400 176602
24800 002032 000413
24900 002034 032767 004000 176572 1$:
25000 002042 001401
25100 002044 000002
25200 002046 032767 000400 176560 2$:
25300 002054 001402
25400 002056 000167 177452
25500 002062 012767 177777 176620 TTY1B:
25600 002070 012700 000036
25700 002074 104010
25800 002076 104007
25900 002100 014341
26000 002102 005067 176602
26100 002106 104020
26200 002110 026727 176564 000040 1$:
26300 002116 001773
26400 002120 012700 000036
26500 002124 104010
26600 002126 104017
26700 002130 117777 176460 176462
26800 002136 004767 000316
26900 002142 000541
27000 002144 010005
27100 002146 006305
27200 002150 006305
27300 002152 006305
27400 002154 104020
27500 002156 026727 176516 000040 2$:
27600 002164 001773
27700 002166 012700 000036
27800 002172 104010
27900 002174 104017
28000 002176 117777 176412 176414
28100 002204 004767 000250
28200 002210 000516
28300 002212 060005

```

```

:*****
:TTY1-- THIS SECTION IS USED WHEN THE DIAGNOSTIC IS BEING CONTROLLED BY
:THE CONSOLE TERMINAL. IT IS EFFECTIVE ONLY WHEN THE DIAGNOSTIC
:STARTING ADDRESS IS 210 AND SR BIT 8 WAS SET AT START TIME.
:THE RESPONSE TO THE MESSAGE 'SELECT TEST NO.' MUST BE THE 2
:DIGIT OCTAL TEST NUMBER FOLLOWED BY :
: 'L' TO LOOP ON TEST
: 'S' TO LOOP ON SEQUENCE
: '.' TO EXECUTE TEST ONCE
:
: ALL SPACES WILL BE IGNORED. AN ILLEGAL INPUT WILL BE FLAGGED BY A '?'
: AND THE RETYPING OF THE ABOVE MESSAGE.
:*****

```

```

TTY1: POPSP2 ;POP 2 FROM STACK
TSTB @TKS ;TEST IF ANY INPUT
BPL 1$ ;BRANCH IF NOT
MOV @TKB,R5 ;GET CHAR
BIC #177600,R5 ;MASK BITS
CMP R5,#177 ;CHECK IF RUBOUT
BNE 1$ ;BRANCH IF NOT
BIC #4400,CNTLSW ;CLEAR LOOP BITS
BR TTY1B
BIT #NITRSW,CNTLSW ;CHECK IF LOOP ON TEST
BEQ 2$ ;BRANCH IF NO LOOP ON TEST
RTI ;LOOP ON TEST
BIT #BIT8,CNTLSW ;TEST IF LOOP ON SEQUENCE
BEQ TTY1B ;BRANCH IF NO LOOP ON SEQUENCE
JMP CHAINY ;CHAIN TO NEXT TEST
TTY1B: MOV #-1,INCHK ;STOP INPUT CHECKING
MOV #30.,R0 ;DELAY FOR HALF DUPLEX
DELAY
TYPEM ;TYPE MESSAGE
MESG3 ;ALLOW INPUT CHECKING AGAIN
CLR INCHK ;WAIT FOR INPUT
1$: READ ;TEST IF CHAR IS A SPACE
CMP TEMPCH,#40 ;BRANCH IF YES
BEQ 1$ ;DELAY FOR HALF DUPLEX
MOV #30.,R0
DELAY
PRNT ;READY?
MOVB @TKB,@TPB ;ECHO CHAR
JSR PC,TESTC ;CHECK IF CHAR IS OK
BR 8$ ;NO, ERROR
MOV R0,R5 ;OK, PUT CHAR INTO R5
ASL R5 ;SHIFT INTO POSITION 5-3
ASL R5
2$: READ ;WAIT FOR NEXT CHAR
CMP TEMPCH,#40 ;CHECK IF A SPACE
BEQ 2$ ;BRANCH IF SPACE
MOV #30.,R0 ;DELAY FOR HALF DUPLEX
DELAY
PRNT ;READY?
MOVB @TKB,@TPB ;ECHO CHAR
JSR PC,TESTC ;CHECK IF CHAR IS OK
BR 8$ ;ERROR IN CHAR
ADD R0,R5 ;OK,R5 NOW = OCTAL TEST NO.

```

```

28400 002214 104020          3$:  READ          ;WAIT FOR TERMINATION CHARACTER
28500 002216 026727 176456 000040  CMP          TEMPCH,#40      ;CHECK IF SPACE
28600 002224 001773          BEQ          3$           ;BRANCH IF SPACE
28700 002226 012700 000036  MOV          #30.,R0      ;DELAY FOR HALF DUPLEX
28800 002232 104010          DELAY
28900 002234 104017          PRNT
29000 002236 117777 176352 176354  MOVB        @TKB,@TPB      ;READY?
29100 002244 012767 004001 176362  MOV          #4001,CNTLSW  ;ECHO CHAR
29200 002252 026727 176422 000114  CMP          TEMPCH,#114   ;SET BITS 11 & 0
29300 002260 001427          BEQ          5$           ;NO, IS IT AN 'L' ?
29400 002262 026727 176412 000154  CMP          TEMPCH,#154   ;BRANCH IF YES
29500 002270 001423          BEQ          5$           ;CHECK LOWER CASE
29600 002272 026727 176402 000123  CMP          TEMPCH,#123   ;NO, IS IT AN 'S'
29700 002300 001414          BEQ          4$           ;BRANCH IF YES
29800 002302 026727 176372 000163  CMP          TEMPCH,#163   ;CHECK LOWER CASE
29900 002310 001410          BEQ          4$
30000 002312 026727 176362 000056  CMP          TEMPCH,#56    ;NO, IS IT A '.' ?
30100 002320 001052          BNE          8$           ;NO, ERROR
30200 002322 012767 000001 176304  MOV          #1,CNTLSW     ;YES SET ONLY BIT 0 IN CONTROL WD
30300 002330 000403          BR           5$
30400 002332 012767 000401 176274  4$:  MOV          #401,CNTLSW  ;SET BITS 8 & 0
30500 002340 012767 000006 175436  5$:  MOV          #6,MACHER    ;CLEAN UP
30600 002346 012706 000600  MOV          #SPBOT,SP     ;INIT SP
30700 002352 020527 000040  CMP          R5,#40        ;IS THIS AN I/O TEST
30800 002356 103033          BHS          8$           ;BRANCH IF YES
30900 002360 020527 000030  CMP          R5,#30        ;IS THIS AN OPTION TEST?
31000 002364 103007          BHS          6$           ;SKIP IF YES
31100 002366 020527 000020  CMP          R5,#20        ;IS THIS AN ECHO TEST
31200 002372 103404          BLO          6$           ;BRANCH IF NOT
31300 002374 012767 000001 176232  MOV          #1,CNTLSW     ;FORCE ECHO TEST TO A SINGLE RUN
31400 002402 000402          BR           7$           ;LEAVE THIS TERMINAL AS CONSOLE
31500 002404 004767 001300  6$:  JSR          PC,CONIT     ;RESET CONSOLE TERMINAL ADDRESS
31600 002410 052767 000200 176226  7$:  BIS          #BIT7,PRGID  ;BYPASS SCOPING
31700 002416 000241          CLC
31800 002420 006105          ROL          R5           ;CLEAR C BIT
31900 002422 016567 002522 176210  MOV          PRGTAB(R5),NXTST;ADDR OF TEST TO NXTST
32000 002430 026727 176204 001700  CMP          NXTST,#WAITF  ;CHECK IF TEST EXISTS
32100 002436 001403          BEQ          8$           ;BRANCH IF NOT
32200 002440 104024          FORWD
32300 002442 000177 176230  JMP          @CURTST       ;SET UP TEST PARAMETERS
32400 002446 104017          PRNT          ;GO TO TEST
32500 002450 112777 000077 176142  8$:  MOVB        #77,@TPB     ;CHECK IF PRINTER IS READY
32600 002456 000601          BR           TTY1B       ;SEND A '?'
                                ;TRY AGAIN

```

```
32800                                     ;TESTC--CHECKS THAT THE INPUTTED CHARACTER IS BETWEEN 0 AND 7 INCLUSIVE
32900
33000 002460 026727 176214 000060 TESTC:  CMP      TEMPCH,#60      ;CHECK IF NUMERIC AND EQ OR GT 0
33100 002466 103001                    BHIS     1$              ;BRANCH IF OK
33200 002470 000207                    RTS      PC              ;ERROR RETURN
33300 002472 026727 176202 000067 1$:  CMP      TEMPCH,#67      ;CHECK IF EQ OR LT 7
33400 002500 101401                    BLOS     2$              ;BRANCH IF OK
33500 002502 000207                    RTS      PC              ;ERROR RETURN
33600 002504 062716 000002 2$:  ADD      #2,@SP        ;SET UP RETURN ADDRESS
33700 002510 016700 176164                    MOV      TEMPCH,R0      ;GET CHAR
33800 002514 042700 177770                    BIC      #177770,R0     ;SAVE ONLY THE DIGIT
33900 002520 000207                    RTS      PC              ;NORMAL RETURN
```

34100	002522	007372	PRGTAB: PT0	:DATA PATH TEST
34200	002524	007446	PT1	:PRINTER CHARACTER TEST
34300	002526	007570	PT2	:NON-PRINTING CHARACTER TEST
34400	002530	010164	PT3	:CARRIAGE RETURN TEST
34500	002532	010304	PT4	:MULTIPLE LINE FEED TEST
34600	002534	010462	PT5	:SINGLE LINE FEED TEST
34700	002536	010666	PT6	:BACKSPACE TEST
34800	002540	011054	PT7	:OVERPRINT TEST
34900	002542	011266	PT10	:PRINTING FREQUENCY SWEEP TEST
35000	002544	011424	PT11	:RIBBON FEED TEST
35100	002546	011456	PT12	:PRINTER BELL TEST
35200	002550	001700	WAITF	:SPARE
35300	002552	001700	WAITF	:SPARE
35400	002554	001700	WAITF	:SPARE
35500	002556	001700	WAITF	:SPARE
35600	002560	011546	PT17	:LIFE TEST
35700	002562	012116	E020	:CHARACTER ECHO TEST
35800	002564	012166	E021	:LINE ECHO TEST, FAST RATE
35900	002566	012224	E022	:LINE ECHO TEST, SLOW RATE
36000	002570	012476	E023	:CHARACTER/CODE ECHO TEST
36100	002572	013020	E024	:SELECTIVE PATTERN ECHO TEST
36200	002574	013566	E025	:BELL ECHO TEST
36300	002576	001700	WAITF	:SPARE
36400	002600	001700	WAITF	:SPARE
36500	002602	001700	WAITF	:SPARE
36600	002604	001700	WAITF	:SPARE
36700	002606	001700	WAITF	:PRARE
36800	002610	001700	WAITF	:SPARE
36900	002612	001700	WAITF	:SPARE
37000	002614	001700	WAITF	:SPARE
37100	002616	001700	WAITF	:SPARE
37200	002620	001700	WAITF	:SPARE
37300	002622	005266	AT0	:I/O TEST NO. 40
37400	002624	005320	AT1	:I/O TEST NO. 41
37500	002626	005352	AT2	:I/O TEST NO. 42
37600	002630	005404	AT3	:I/O TEST NO. 43
37700	002632	005436	AT4	:I/O TEST NO. 44
37800	002634	005526	AT5	:I/O TEST NO. 45
37900	002636	005604	AT6	:I/O TEST NO. 46
38000	002640	005674	AT7	:I/O TEST NO. 47
38100	002642	005744	AT10	:I/O TEST NO. 50
38200	002644	006002	AT11	:I/O TEST NO. 51
38300	002646	006042	AT12	:I/O TEST NO. 52
38400	002650	006116	AT13	:I/O TEST NO. 53
38500	002652	006176	AT14	:I/O TEST NO. 54
38600	002654	006262	AT15	:I/O TEST NO. 55
38700	002656	006362	AT16	:I/O TEST NO. 56
38800	002660	006430	AT17	:I/O TEST NO. 57
38900	002662	006500	AT20	:I/O TEST NO. 60
39000	002664	006572	AT21	:I/O TEST NO. 61
39100	002666	006672	AT22	:I/O TEST NO. 62
39200	002670	007000	AT23	:I/O TEST NO. 63
39300	002672	007112	AT24	:LSI TEST NO. 64
39400	002674	007212	AT25	:LSI TEST NO. 65
39500	002676	007270	AT26	:LSI TEST NO. 66
39600	002700	001700	WAITF	:SPARE
39700	002702	001700	WAITF	:SPARE

```

39800 002704 001700      WAITF      : SPARE
39900 002706 001700      WAITF      : SPARE
40000 002710 001700      WAITF      : SPARE
40100 002712 001700      WAITF      : SPARE
40200 002714 001700      WAITF      : SPARE
40300 002716 001700      WAITF      : SPARE
40400 002720 001700      WAITF      : SPARE
40500
40600
40700
40800
40900
41000

```

```

:*****
:EMTINT -----SERVICE ROUTINE FOR TRAPS THROUGH
:                               LOCATION 30.
:*****

```

```

41100 002722 011646      EMTINT: MOV    @SP,-(SP)      :PUSH STACKED PC TO GET A WORK COPY. (Q)
41200 002724 162716 000002      SUB    #2,@SP          :SUB 2 TO POINT TO CALLING TRAP INSTR.
41300 002730 017616 000000      MOV    @ (SP),@SP      :PLACE TRAP INSTR INTO THIS STACK WORK AREA.
41400 002734 121627 000035      CMPB  @SP,#35         :EXAMINE ITS RIGHT SIDE. (Q)
41500 002740 101402      BLOS  2$              :BRANCH IF WITHIN RANGE OF ESTABLISHED TABLE.
41600 002742 000000      1$:  HALT              :ELSE HALT.
41700 002744 000776      BR    1$
41800 002746 006116      2$:  ROL    @SP          :MULT INSTR BY 2 TO GET WORD DISPLACEMENT.
41900 002750 042716 177001      BIC   #177001,@SP     :STRIP OFF OP CODE AND LS BIT.
42000 002754 062716 002776      ADD   #EMTTAB,@SP     :ADD IN STARTING ADDRESS OF TABLE.
42100 002760 017616 000000      MOV   @ (SP),@SP      :FROM TABLE GET OUT DESIRED POINTER.
42200 002764 005046      CLR   -(SP)           :PUSH A ZERO PSW.
42300 002766 012746 002774      MOV   #3$,-(SP)      :PUSH A PC = TO #3$ OF THIS ROUTINE.
42400 002772 000002      RTI                      :DO RTI (POP-POP) TO ESTABLISH THE ZERO PSW.
42500 002774 000136      3$:  JMP   @ (SP)+       :JMP TO ROUTINE LEAVING STACK AS FOUND.
42600

```

```

EMTTAB: TYP      :MESSAGE OUTPUT ROUTINE
ERR      :I/O TEST ERROR ROUTINE
EHLT     :UNCONDITIONAL HALT
STLSRV   :KEYBOARD VECTOR/PRIORITY SETUP
STLSPV   :PRINTER VECTOR/PRIORITY SETUP
CHAINN   :COMMON TEST EXIT
CHLT     :SR BIT 15 HALT
TYPM     :MESSAGE OUTPUT ROUTINE, MULTI DEVICES
DLY      :DELAY ROUTINE
TTY1     :CONSOLE TERMINAL CONTROL
$CRLF    :CARRIAGE RETURN-LINE FEED TO ALL DL11'S
$$CRLF   :CARRIAGE RETURN-LINE FEED TO CONSOLE
$LF      :LINE FEED ONLY (TO ALL)
$PRTC    :PRINT CHAR
$PRHDR   :PRINT TEST HEADER
$PRNT    :PRINTER READY
$READ    :READ CHAR
$AREAD   :I/O TEST READ ROUTINE
$CR      :CARRIAGE RETURN ONLY (TO ALL)
$BTASC   :BINARY TO ASCII CONVERSION
$FORWD   :FORWARD ROUTINE ( BETWEEN TESTS )
$READC   :READ CONSOLE KYBD ONLY
SPARET   :SPARE EMT
SPARET   :SPARE EMT
SPARET   :SPARE EMT
SPARET   :SPARE EMT
SPARET   :SPARE EMT
SPARET   :SPARE EMT
SPARET   :SPARE EMT

```

```

42700 002776 003076
42800 003000 003320
42900 003002 003346
43000 003004 003356
43100 003006 003406
43200 003010 001412
43300 003012 000720
43400 003014 003164
43500 003016 003436
43600 003020 001776
43700 003022 003214
43800 003024 003142
43900 003026 003216
44000 003030 004324
44100 003032 003236
44200 003034 004314
44300 003036 004112
44400 003040 003640
44500 003042 003226
44600 003044 004006
44700 003046 003562
44800 003050 004204
44900 003052 003072
45000 003054 003072
45100 003056 003072
45200 003060 003072
45300 003062 003072
45400 003064 003072

```

```

45500 003066 003072          SPARET          ;SPARE EMT
45600 003070 003072          SPARET          ;SPARE EMT
45700 003072 000000          SPARET: HALT      ;HALT IF TRAP TO UNDEFINED
45800 003074 000776          BR          SPARET      ;EMT IS ATTEMPTED.
45900
46000
46100
46200
46300
46400
46500
46600
46700
46800
46900
47000
47100
47200
47300
47400
47500
47600
47700
47800
47900
48000
  
```

.SBTTL COMMON ROUTINES USED BY LA36 TESTS

```

:*****
:THIS SECTION CONTAINS MOST ROUTINES CALLED BY
:THE VARIOUS TESTS EITHER BY TRAPPING THROUGH LOCATION
:30 OR BY SUBROUTINE CALLS (JSR PC,***)
```

```

:*****
:TYPE-- A COMMON ROUTINE USED TO TYPE MESSAGES ON THE
:CONSOLE TERMINAL ONLY. THE NULL CHARACTER TERMINATES
:THE MESSAGE. CALLED THROUGH AN EMT TRAP.
:CALLING SEQUENCE
:TYPE
:MSG          ;ADDRESS OF MESSAGE
```

```

48100 003076 010046          TYP:  MOV      R0,-(SP)      ;SAVE R0
48200 003100 016601 000002    MOV      2(SP),R1      ;GET POINTER TO ADDR. OF MSG.
48300 003104 062766 000002 000002  ADD      #2,2(SP)
48400 003112 011101          MOV      (R1),R1      ;ADDR. OF MSG TO R1
48500 003114 112100          1$:  MOVVB   (R1)+,R0      ;GET CHAR
48600 003116 100403          BMI      2$          ;BRANCH IF WANT AUTO CR-LF
48700 003120 001004          BNE      3$          ;PRINT CHAR IF NOT NULL
48800 003122 012600          MOV      (SP)+,R0    ;RESTORE R0
48900 003124 000002          RTI          ;EXIT IF NULL CHAR
49000 003126 104013          2$:  SCRLF          ;YES, SEND CR-LF
49100 003130 000771          BR      1$          ;GET NEXT CHAR
49200 003132 104017          3$:  PRNT          ;PRINTER READY?
49300 003134 110077 175460    MOVVB   R0,@TPB      ;LOAD PRINTER BUFFER WITH CHAR
49400 003140 000765          BR      1$          ;GO GET NEXT CHAR
49500
49600 003142 104017          $$SCRLF: PRNT        ;PRINTER READY?
49700 003144 112777 000015 175446  MOVVB   #15,@TPB     ;SEND CR
49800 003152 104017          PRNT        ;PRINTER READY?
49900 003154 112777 000012 175436  MOVVB   #12,@TPB     ;SEND LF
50000 003162 000002          RTI          ;RETURN TO CALLER
  
```



```

50200      ;XXXXXXXXXX
50300      ;
50400      ;TYPM---MULTI TYPE-A COMMON ROUTINE TO OUTPUT
50500      ;      A MESSAGE ON ALL DL11S IF THE MULTI TEST
50600      ;      SWITCH (BIT 13) IS RESET. THIS ROUTINE IS USED BY
50700      ;      THE PRINTER TESTS TO TYPE HEADINGS. IF A UNIT
50800      ;      IS NOT READY, THE CHARACTER WILL NOT BE TYPED.
50900      ;
51000      ;XXXXXXXXXX
51100
51200 003164 011601      TYPM:  MOV      (SP),R1      ;GET POINTER TO ADDR OF MESG
51300 003166 062716 000002      ADD      #2,@SP
51400 003172 011101      MOV      (R1),R1      ;ADDR OF MESG TO R1
51500 003174 112100      1$:  MOVB   (R1)+,R0      ;GET CHAR
51600 003176 100402      BMI     2$           ;BRANCH IF WANT AUTO CR-LF
51700 003200 001003      BNE     3$           ;CONTINUE IF NOT NULL
51800 003202 000002      RTI                    ;RETURN
51900 003204 104012      2$:  CRLF                    ;YES, SEND CR-LF
52000 003206 000772      BR      1$           ;NEXT CHAR
52100 003210 104015      3$:  PRINTC  ;PRINT CHAR
52200 003212 000770      BR      1$           ;GO GET NEXT CHAR.
52300
52400 003214 104022      $CRLF: CR              ;SEND CR
52500 003216 012700 000012      $LF:  MOV      #12,R0      ;SET LF CHAR
52600 003222 104015      PRINTC  ;SEND IT
52700 003224 000002      RTI                    ;RETURN TO CALLER
52800
52900 003226 012700 000015      $CR:  MOV      #15,R0      ;SET CR CHAR
53000 003232 104015      PRINTC  ;SEND IT
53100 003234 000002      RTI                    ;RETURN
53200
53300      ;*****
53400      ;
53500      ;ROUTINE TO PRINT TEST HEADER
53600      ;
53700      ;*****
53800
53900 003236 012700 000000      $PRHDR: MOV      #0,R0      ;TRANSMIT
54000 003242 104015      PRINTC  ;NUL CODE.
54100 003244 104007      TYPEM   ;PRINT MESSAGE
54200 003246 014113      HDRMSG
54300 003250 016700 175362      MOV      RTNNO,R0      ;GET TEST NUMBER
54400 003254 006200      ASR     R0            ;GET FIRST DIGIT
54500 003256 006200      ASR     R0
54600 003260 006200      ASR     R0
54700 003262 042700 177770      BIC     #177770,R0      ;MASK FIRST DIGIT
54800 003266 062700 000060      ADD     #60,R0        ;MAKE ASCII
54900 003272 104015      PRINTC  ;PRINT DIGIT
55000 003274 016700 175336      MOV      RTNNO,R0      ;GET TEST NUMBER AGAIN
55100 003300 042700 177770      BIC     #177770,R0      ;MASK LAST DIGIT
55200 003304 062700 000060      ADD     #60,R0        ;MAKE ASCII
55300 003310 104015      PRINTC  ;PRINT DIGIT
55400 003312 104012      CRLF                    ;CR-LF
55500 003314 104014      LF                    ;BLANK LINE
55600 003316 000002      RTI                    ;RETURN

```

```

55800
55900
56000
56100
56200
56300
56400
56500
56600
56700 003320 032777 040000 175366 ERR: BIT #SCOPSW,@SR ;CHECK SCOPE SWITCH
56800 003326 001404 BEQ 1$ ;BRANCH IF NO SCOPE
56900 003330 005767 175310 TST PRGID ;SCOPING WANTED, FIRST ERROR?
57000 003334 100001 BPL 1$ ;BRANCH AND HALT ON FIRST ERROR
57100 003336 000002 RTI ;SCOPE EXIT
57200 003340 052767 100000 175276 1$: BIS #BIT15,PRGID ;SET ERROR INDICATOR
57300 003346 011600 EHLT: MOV @SP,R0
57400 003350 005740 TST -(R0) ;ADDRESS OF CALL INTO R0
57500 003352 000000 HALT
57600 003354 000002 ERRHLT: RTI ;RETURN TO TEST FOLLOWING CALL
57700
57800
57900
58000
58100
58200
58300
58400
58500
58600
58700
58800 003356 017667 000000 000012 STLSRV: MOV @(SP),STPRA+2 ;SET RETURN ADR AND VECTOR
58900 003364 062716 000002 ADD #2,@SP
59000 003370 016701 175226 MOV TKVTR,R1
59100 003374 012721 000000 STPRA: MOV #0,(R1)+
59200 003400 016721 175220 MOV TKLVL,(R1)+
59300 003404 000002 RTI
59400
59500
59600
59700
59800
59900
60000
60100
60200
60300
60400
60500 003406 017667 000000 000012 STLSPV: MOV @(SP),STPPA+2 ;SET RETURN ADR AND VECTOR
60600 003414 062716 000002 ADD #2,@SP
60700 003420 016701 175202 MOV TPVTR,R1
60800 003424 012721 000000 STPPA: MOV #0,(R1)+
60900 003430 016721 175174 MOV TPLVL,(R1)+
61000 003434 000002 RTI ;RETURN TO CALLER
    
```

61200
 61300
 61400
 61500
 61600
 61700
 61800
 61900
 62000
 62100
 62200
 62300
 62400
 62500
 62600
 62700
 62800
 62900
 63000
 63100
 63200
 63300
 63400
 63500
 63600
 63700
 63800
 63900
 64000 003436 010146
 64100 003440 016701 175226
 64200 003444 005301
 64300 003446 001376
 64400 003450 005300
 64500 003452 001372
 64600 003454 012601
 64700 003456 000002

```

*****
:DELAY--A COMMON ROUTINE TO DELAY PROCESSING
:      A GIVEN NUMBER OF MSEC.
:      CALLING SEQUENCE:
:      MOV #5,R0 ;R0 CONTAINS THE NUMBER OF MSEC DELAY DESIRED
:      DELAY

:      THE DELAY IS EFFECTED BY THE EXECUTION OF THE LOOP;
:      1$: DEC R1
:         BNE 1$

:      SINCE THE EXECUTION TIMES OF THE PDP11 LINE DOES VARY FROM
:      MACHINE TO MACHINE, THE VALUE AT SYMBOLIC LOCATION
:      'TIMER' MUST BE CHANGED TO THE APPROPRIATE VALUE AS SHOWN BELOW
:      BEFORE STARTING THE DIAGNOSTIC. 'TIMER' IS INITIALIZED
:      FOR AN 11/05,11/10(=251).
  
```

MACHINE	05&10	35&40	15&20	LSI&03	BIPOLAR	11/45 & 11/70	MOS	CORE	
LOOP: DEC R1	3.4	.99	2.3		.30		.51	.90	
BNE LOOP	2.5	1.76	2.6		.60		.98	1.13	
TIME=	5.9USEC	2.75	4.9	7.7	.90USEC		1.49USEC		2.03USEC
SET TIMER	251	554	314	202	2127		1237	755	

```

:XXXXXXXXXX
DLY:  MOV R1,-(SP) ;SAVE R1
1$:   MOV TIMER,R1 ;MOV 1 MSEC LOOP CNT TO R1
2$:   DEC R1 ;DECREMENT COUNT
      BNE 2$ ;BRANCH IF NOT ZERO
      DEC R0 ;DEC NO. OF MSEC DELAY
      BNE 1$ ;DELAY AGAIN IF NOT ZERO
      MOV (SP)+,R1 ;ALL DONE RESTORE R1
      RTI
  
```

64900
65000
65100
65200
65300
65400
65500

```
:*****  
:PFAIL--POWER FAIL ROUTINE  
:      SAVE ALL REGISTERS AND SET RESTART ADDRESS  
:      INTO LOCATION 24  
:RESTART--POWER FAIL RECOVERY
```

100

; RESTORE ALL REGISTERS AND GO TO START

```

200          ;*****
300
400 003460 010046          PFAIL:  MOV  R0,-(SP)
500 003462 010146          MOV  R1,-(SP)
600 003464 010246          MOV  R2,-(SP)
700 003466 010346          MOV  R3,-(SP)
800 003470 010446          MOV  R4,-(SP)
900 003472 010546          MOV  R5,-(SP)
1000 003474 016746 174324  MOV  24,-(SP)
1100 003500 010667 000010  MOV  SP,SAVR6          ;SAVE STACK POSITION
1200 003504 012767 003516 174312  MOV  #RESTRT,24      ;STORE RESTART ADDRESS
1300 003512 000000          HALT
1400 003514 000000          SAVR6: .WORD 0
1500 003516 104007          RESTRT: TYPEM
1600 003520 003552          1$
1700 003522 016706 177766  MOV  SAVR6,SP          ;RESTORE STACK POINTER
1800 003526 012667 174272  MOV  (SP)+,24          ;RESTORE PFAIL ADDRESS
1900 003532 012605          MOV  (SP)+,R5
2000 003534 012604          MOV  (SP)+,R4
2100 003536 012603          MOV  (SP)+,R3
2200 003540 012602          MOV  (SP)+,R2
2300 003542 012601          MOV  (SP)+,R1
2400 003544 012600          MOV  (SP)+,R0
2500 003546 000167 175236  JMP  START
2600
2700 003552 200 120 117 1$: .ASCIZ <ACRLF>/POWER/<ACRLF>
    003555 127 105 122
    003560 200 000
2800          .EVEN
    
```

3000
 3100
 3200
 3300
 3400
 3500
 3600
 3700
 3800
 3900
 4000
 4100
 4200
 4300
 4400
 4500
 4600
 4700
 4800
 4900
 5000
 5100
 5200
 5300

```

:*****
:FORWARD--THIS ROUTINE TRANSFERS THE 2 OR 4 ARGUMENTS
:          FROM THE TEST ROUTINE. THEY ARE;
:
:          1- ROUTINE NUMBER
:          2- ADDRESS OF NEXT TEST
:          3- ITERATION COUNT (I/O TESTS ONLY)
:          4- SCOPE ENTRY ADDRESS (I/O TESTS ONLY)
:*****
  
```

003562	016705	175052	\$FORWD:	MOV	NXTST,R5	:ADDR OF NEXT TEST TO R5
003566	012567	175044		MOV	(R5)+,RTNNO	:GET NUMBER OF NEXT TEST
003572	012567	175042		MOV	(R5)+,NXTST	:GET ADDR OF FOLLOWING TEST
003576	105767	175042		TSTB	PRGID	:CHECK IF I/O TEST
003602	100407			BMI	FORWDB	:SKIP THE FETCH OF ITER CNT AND SCOPE
003604	012567	175050		MOV	(R5)+,ICTR	:GET ITERATION COUNT
003610	012567	175026		MOV	(R5)+,SCOPTR	:GET SCOPE ENTRY POINT
003614	010567	175056	FORWDA:	MOV	R5,CURTST	:ENTRY POINT TO TEST IN CUR TST
003620	000002			RTI		:EXIT
003622	012767	177777 175012	FORWDB:	MOV	#-1,SCOPTR	:FORCE NO SCOPE
003630	012767	000001 175022		MOV	#1,ICTR	:FORCE INTERATION COUNT OF 1
003636	000766			BR	FORWDA	

5500
5600
5700
5800
5900
6000
6100
6200
6300
6400
6500
6600
6700
6800
6900
7000
7100
7200
7300
7400
7500
7600
7700
7800
7900
8000
8100
8200
8300
8400
8500
8600
8700
8800
8900
9000
9100
9200
9300
9400
9500
9600
9700
9800
9900
10000
10100
10200
10300
10400

003640 012767 000600 175016
 003646 052777 000004 174742
 003654 005077 174740
 003660 105777 174726
 003664 100410
 003666 012700 000001
 003672 104010
 003674 005367 174764
 003700 001367
 003702 104002
 003704 000755
 003706 000002

 003710 016700 174666
 003714 010067 174672
 003720 005720
 003722 010067 174666
 003726 005720
 003730 016767 174662 000044
 003736 010067 174654
 003742 005720
 003744 016767 174650 000032
 003752 010067 174642
 003756 016767 174622 174636
 003764 016767 174614 174634
 003772 062767 000004 174626
 004000 000207

 004002 000000
 004004 000000

```

:*****
:AREAD--A ROUTINE WHICH, THROUGH THE FACILITY OF
:THE MAINTENANCE BIT, OUTPUTS TO THE
:PRINTER BUFFER AND READS THE KEYBOARD
:STATUS DONE. IF THE DONE IS NOT SET
:WITHIN 600 MSEC, THE CPU WILL HALT WITH
:THE LOCATION OF THE ERROR IN RO. PRESS
:CONTINUE TO CONTINUE WITH TESTS.
:*****
    
```

```

$AREAD: MOV #600, BRCTR ;SET UP 600 MSEC DELAY
        BIS #4, @TPS ;SET MAINTENANCE BIT
        CLR @TPB ;LOAD PRINTER BUFFER
1$:     TST @TKS ;CHECK DONE BIT
        BMI 2$ ;BRANCH IF DONE
        MOV #1, R0 ;ONE TO R0
        DELAY ;DELAY 1 MSEC.
        DEC BRCTR ;600 MSEC OVER
        BNE 1$ ;BRANCH IF NO
2$:     BR $AREAD ;TRY AGAIN
        RTI ;RETURN TO TEST
    
```

```

:*****
:CONIT--THIS ROUTINE SETS UP THE DEVICE ADDRESSES
:AND INTERRUPT VECTORS FOR THE CONSOLE
:TERMINAL.
:*****
    
```

```

CONIT: MOV CONADD, R0 ;CONSOLE KEYBOARD STATUS ADDR TO R0
CONSET: MOV R0, TKS ;KEYBOARD STATUS ADDRESS (777560) TO TKS
        TST (R0)+ ;INCREMENT R0 BY TWO
        MOV R0, TKB ;KEYBOARD DATA ADDR (777562) TO TKB
        TST (R0)+ ;INCREMENT R0 BY TWO
        MOV TPS, TPSS ;SAVE TPS OF LAST TERMINAL
        MOV R0, TPS ;PRINTER STATUS ADDR(777564) TO TPS
        TST (R0)+ ;INCREMENT R0 BY TWO
        MOV TPB, TPBS ;SAVE TPB OF LAST TERMINAL
        MOV R0, TPB ;PRINTER DATA ADDR (777566) TO TPB
        MOV CONVEC, TKVTR ;KEYBOARD INTERRUPT VECTOR (60) TO TKVTR
        MOV CONVEC, TPVTR ;KEYBOARD INTERRUPT VECTOR (60) TO TPVTR
        ADD #4, TPVTR ;PRINTER INTERRUPT VECTOR (64) TO TPVTR
        RTS PC
  

TPSS: .WORD 0 ;LAST TERM STATUS REG ADR
TPBS: .WORD 0 ;LAST TERM BUFFER REG ADR
    
```



```

10600
10700
10800
10900
11000
11100
11200
11300
11400
11500
11600
11700 004006 010267 000060
11800 004012 006302
11900 004014 062702 004100
12000
12100 004020 014267 000052
12200 004024 005067 000044
12300 004030 166701 000042
12400 004034 103403
12500 004036 005267 000032
12600 004042 000772
12700 004044 066701 000026
12800 004050 062767 000060 000016
12900 004056 116720 000012
13000 004062 005367 000004
13100 004066 001354
13200 004070 000002
13300 004072 000000
13400 004074 000000
13500 004076 000000
13600 004100 000001 000012 000144
      004106 001750 023420

:*****
:
: BINARY TO ASCII CONVERSION (1 TO 5 ASCII CHARACTERS)
: CALLING SEQUENCE
:
:   MOV   ADDRESS OF LOC. TO STORE FIRST ASCII CHAR. INTO R0
:   MOV   BINARY NUMBER TO BE CONVERTED INTO R1
:   MOV   NUMBER TO BE CONVERTED AS A POWER OF TEN INTO R2
:   BTOASC
:
:*****
$BTASC: MOV   R2,CNVCTR   ;SAVE TEN POWER
        ASL   R2         ;R2*2
        ADD   #ADTENP,R2 ;CALCULATE ADDRESS OF
                        ;STARTING TEN POWER
1$:    MOV   -(R2),TENPWR ;POWER OF TEN VALUE TO TEN PWR
        CLR   DIGIT     ;CLEAR CURRENT DIGIT
2$:    SUB   TENPWR,R1   ;SUBTRACT TEN POWER FROM BINARY VALUE
        BCS  3$         ;BRANCH IF END
        INC  DIGIT     ;
        BR   2$         ;
3$:    ADD   TENPWR,R1   ;RESTORE SUBTRACTED VALUE
        ADD   #60,DIGIT ;CONVERT (DIGIT) TO ASCII
        MOVB DIGIT,(R0)+ ;PUT ASCII CHAR INTO USER BUFFER
        DEC  CNVCTR     ;FINISHED ALL CHARS. CALLED FOR
        BNE  1$         ;BRANCH IF NOT FINISHED
        RTI             ;YES, EXIT
CNVCTR: .WORD 0         ;CONVERSION CHARACTER COUNT
DIGIT:  .WORD 0         ;CONVERTED CHARACTER
TENPWR: .WORD 0         ;CURRENT TEN POWER
ADTENP: .WORD 1.,10.,100.,1000.,10000.
    
```

```

13800      ;XXXXXXXXXX
13900      :
14000      :READ-- A COMMON ROUTINE WHICH CHECKS THE KEYBOARD
14100      :          DONE FLAG & SETS A FLAG INDICATING CHAR PARITY
14200      :
14300      :XXXXXXXXXX
14400
14500 004112 004767 177572      $READ: JSR      PC,CONIT      ;RESET CONSOLE ADR AND VECTORS
14600 004116 005767 174534      TST      DLCNT          ;CHECK IF MULTI DL11'S AVAILABLE
14700 004122 001430      BEQ      $READC        ;NONE, WAIT FOR CONSOLE INPUT
14800 004124 016767 174526 174534 1$: MOV      DLCNT,COUNT3    ;SET DL11 COUNT
14900 004132 016767 174474 174530 2$: MOV      FSTDL,XCSR     ;ADDRESS OF FIRST DL11 INTO XCSR
15000 004140 105777 174524      TSTB     @XCSR          ;TEST IF ANY INPUT
15100 004144 100005      BPL      3$            ;CONTINUE IF NO INPUT
15200 004146 016700 174516      MOV      XCSR,R0       ;SET THIS DL11 AS CONSOLE
15300 004152 004767 177536      JSR      PC,CONSET
15400 004156 000415      BR       READ1        ;READ CHAR AND RETURN
15500 004160 005367 174502      3$: DEL      COUNT3      ;DECREMENT DL11 COUNT
15600 004164 001404      BEQ      4$            ;TEST CONSOLE WHEN DONE DL11'S
15700 004166 062767 000010 174474 4$: ADD      #10,XCSR     ;NEXT DL11 ADDRESS
15800 004174 000761      BR       2$            ;CONTINUE
15900 004176 105777 174410      4$: TSTB     @TKS          ;CHECK CONSOLE
16000 004202 100350      BPL      1$            ;WAIT, NO INPUT
16100 004204 105777 174402      $READC: TSTB     @TKS          ;CHECK KEYBOARD DONE FLAG
16200 004210 100375      BPL      $READC        ;BRANCH IF NOT SET
16300 004212 117767 174376 174460 READ1: MOVB     @TKB,TEMPCH    ;SAVE CHARACTER
16400 004220 116767 174454 174456  MOVB     TEMPCH,PCHAR   ;SAVE CODE WITH PARITY BIT
16500 004226 042767 177400 174450  BIC      #177400,PCHAR   ;MASK UNWANTED BITS
16600 004234 116767 174440 174441  MOVB     TEMPCH,PARITY+1 ;SAVE CHAR WITH PARITY BIT
16700 004242 042767 177600 174430  BIC      #177600,TEMPCH  ;MAKE IT 7 BIT ASCII
16800 004250 026727 174424 000004  CMP      TEMPCH,#4      ;DISREGARD EOT
16900 004256 001715      BEQ      $READ
17000 004260 012700 000011      MOV      #11,R0        ;SET SHIFT COUNT
17100 004264 042767 000377 174410  BIC      #377,PARITY    ;CLEAR PARITY FLAG
17200 004272 005300      1$: DEC      R0          ;DECREMENT SHIFT COUNT
17300 004274 001406      BEQ      2$            ;EXIT IF DONE
17400 004276 106367 174401      ASLB     PARITY+1      ;SHIFT CODE
17500 004302 103373      BCC      1$            ;CONTINUE IF BIT WAS ZERO
17600 004304 105167 174372      COMB     PARITY        ;CHANGE PARITY FLAG IF BIT WAS ONE
17700 004310 000770      BR       1$            ;CONTINUE
17800 004312 000002      2$: RTI              ;SET, RET. TO CALLER
17900
18000      ;XXXXXXXXXX
18100      :
18200      :PRINT-- A COMMON ROUTINE TO CHECK THE PRINTER READY FLAG
18300      :
18400      :XXXXXXXXXX
18500
18600 004314 105777 174276      $PRNT: TSTB     @TPS          ;CHECK PRINTER READY FLAG
18700 004320 100375      BPL      $PRNT        ;BRANCH IF NOT SET
18800 004322 000002      RTI              ;SET, RETURN
  
```

```

19000
19100
19200
19300
19400
19500
19600
19700
19800
19900
20000
20100 004324 016767 174252 174360 $PRTC: MOV CONADD,TEMP ;SET CONSOLE ADR
20200 004332 062767 000004 174352 ADD #4,TEMP
20300 004340 105777 174346 1$: TSTB @TEMP
20400 004344 100375 BPL 1$ ;WAIT FOR CONSOLE READY
20500 004346 062767 000002 174336 ADD #2,TEMP ;SET ADR
20600 004354 010077 174332 MOV RO,@TEMP ;LOAD CONSOLE PRINTER BUFFER
20700 004360 032777 020000 174326 BIT #BIT13,@SR ;CHECK SW 13
20800 004366 001003 BNE 2$ ;SEND ALL TERMS IF SW13 DOWN
20900 004370 005767 174262 TST DLCNT ;CHECK IF MULTIPLE DL11'S
21000 004374 001002 BNE 3$ ;CHECK FOR INPUT IF THERE
21100 004376 000167 000432 JMP 18$
21200 004402 016767 174250 174256 2$: MOV DLCNT,COUNT3 ;PUT NO. DL11'S INTO COUNT3
21300 004410 016767 174216 174252 3$: MOV FSTDL,XCSR ;ADDR OF FIRST DL INTO XCSR
21400 004416 005767 174266 4$: TST INCHK ;CHECK FOR INPUT?
21500 004422 001140 BNE 13$
21600 004424 026727 174206 000020 CMP RTNNO,#20 ;PRINTING TEST?
21700 004432 002004 BGE 5$ ;BRANCH IF NOT
21800 004434 022767 104011 175236 CMP #TTYCTL,WAITF ;KEYBOARD CONTROL?
21900 004442 001130 BNE 13$ ;SKIP INPUT CHECK IF NOT
22000 004444 105777 174220 5$: TSTB @XCSR ;TEST IF ANY INPUT
22100 004450 100125 BPL 13$ ;CONTINUE IF NO INPUT
22200 004452 062767 000002 174210 ADD #2,XCSR ;SET BUFFER ADDRESS
22300 004460 017767 174204 174212 MOV @XCSR,TEMPCH
22400 004466 042767 177600 174204 BIC #177600,TEMPCH
22500 004474 026727 174200 000003 CMP TEMPCH,#3 ;CHECK IF CONTROL-C
22600 004502 001006 BNE 6$ ;CONTINUE IF NOT
22700 004504 026727 174126 000024 CMP RTNNO,#24 ;CHECK IF TEST 24
22800 004512 001002 BNE 6$ ;CONTINUE IF NOT CONTROL-C
22900 004514 000167 000420 JMP 20$
23000 004520 026727 174154 000177 6$: CMP TEMPCH,#177 ;CHECK IF RUBOUT
23100 004526 001427 BEQ 9$ ;YES, CHECK TEST NUMBER
23200 004530 026727 174102 000017 CMP RTNNO,#17 ;TEST 17?
23300 004536 001003 BNE 7$ ;BRANCH IF NOT
23400 004540 016703 174134 MOV TEMPCH,R3 ;SAVE CHARACTER
23500 004544 000461 BR 12$ ;CONTINUE
23600 004546 026727 174064 000021 7$: CMP RTNNO,#21 ;TEST 21?
23700 004554 001004 BNE 8$ ;BRANCH IF NOT
23800 004556 016767 174116 174076 MOV TEMPCH,REPT ;SAVE CHARACTER
23900 004564 000451 BR 12$ ;CONTINUE
24000 004566 026727 174044 000022 8$: CMP RTNNO,#22 ;TEST 22?
24100 004574 001056 BNE 14$ ;CONTINUE IF NOT
24200 004576 016767 174076 174056 MOV TEMPCH,REPT ;SAVE CHARACTER
24300 004604 000441 BR 12$ ;CONTINUE
24400 004606 026727 174024 000021 9$: CMP RTNNO,#21 ;CHECK IF TEST 21
24500 004614 001011 BNE 10$ ;NO, CHECK IF TEST 22
24600 004616 022626 POPSP2 ;ASJUST STACK
    
```

:PRINTC--SENDS A CHARACTER AT A TIME FIRST TO THE
 CONSOLE DL11 THEN TO ALL MULTIPLE DL11S IF
 SR BIT 13 IS = 0. IF THE REFERENCED PRINTER
 READY BIT IS NOT SET, THE CHARACTER WILL NOT BE
 SENT TO THAT PRINTER. ENTER WITH CHARACTER IN RO.
 CALL: PRINTC

24700	004620	012700	000036			MOV	#30.,R0		;DELAY FOR HALF DUPLEX
24800	004624	104010				DELAY			
24900	004626	104007				TYPEN			;YES, TEST 21
25000	004630	014272				ECOEND			;PRINT TERMINATION MESSAGE
25100	004632	104005				CHAIN			;CHAIN TO NEXT TEST
25200	004634	000167	005334			JMP	E021A		;REPEAT TEST IF LOOP ON TEST SW SET
25300	004640	026727	173772	000022	10\$:	CMP	RTNNO,#22		;CHECK IF TEST 22
25400	004646	001011				BNE	11\$;NO, CHECK IF TEST 24
25500	004650	022626				POPSP2			;ADJUST STACK
25600	004652	012700	000036			MOV	#30.,R0		;DELAY FOR HALF DUPLEX
25700	004656	104010				DELAY			
25800	004660	104007				TYPEN			;YES, PRINT TERMINATION MESSAGE
25900	004662	014272				ECOEND			
26000	004664	104005				CHAIN			;CHAIN TO NEXT TEST
26100	004666	000167	005340			JMP	E022A		;REPEAT TEST IF LOOP ON TEST SW SET
26200	004672	026727	173740	000024	11\$:	CMP	RTNNO,#24		;TEST 24?
26300	004700	001133				BNE	22\$;WAIT FOR NEXT TEST IF NOT TEST 24
26400	004702	022626				POPSP2			;RESET STACK
26500	004704	000167	006244			JMP	TERM		;TERMINATE TEST
26600	004710	012700	000036		12\$:	MOV	#30.,R0		;DELAY FOR HALF DUPLEX
26700	004714	104010				DELAY			
26800	004716	016700	173756			MOV	TEMPCH,R0		;SET NEW CHARACTER
26900	004722	000403				BR	14\$;CONTINUE
27000	004724	062767	000002	173736	13\$:	ADD	#2,XCSR		;SET STATUS ADDRESS IN XCSR
27100	004732	062767	000002	173730	14\$:	ADD	#2,XCSR		
27200	004740	016767	173636	173744		MOV	CONADD,TEMP		;CHECK IF CONSOLE TERMINAL
27300	004746	062767	000004	173736		ADD	#4,TEMP		;IS THIS DL
27400	004754	026767	173732	173706		CMP	TEMP,XCSR		
27500	004762	001420				BEQ	17\$		
27600	004764	105777	173700		15\$:	TSTB	@XCSR		;TEST PRINTER READY
27700	004770	100375				BPL	15\$;WAIT FOR READY
27800	004772	062767	000002	173670		ADD	#2,XCSR		;SET XCSR TO PRINTER BUFFER
27900	005000	010077	173664			MOV	R0,@XCSR		;LOAD CHARACTER INTO BUFFER
28000	005004	005367	173656		16\$:	DEC	COUNT3		;DECREASE COUNT OF DL11'S
28100	005010	001411				BEQ	18\$;ALL DONE,EXIT
28200	005012	062767	000002	173650		ADD	#2,XCSR		;SET XCSR TO NEXT DL11 PRINTER STATUS
28300	005020	000167	177372			JMP	4\$;GO TEST NEXT DL11 READY FLAG
28400	005024	062767	000002	173636	17\$:	ADD	#2,XCSR		;SET XCSR TO PRINTER BUFFER
28500	005032	000764				BR	16\$;DO NOT LOAD BUFFER
28600	005034	005767	173650		18\$:	TST	INCHK		;WANT INPUT CHECK?
28700	005040	001111				BNE	26\$;NO, BRANCH
28800	005042	026727	173570	000020		CMP	RTNNO,#20		;PRINTING TEST?
28900	005050	002004				BGE	19\$;BRANCH IF NOT
29000	005052	022767	104011	174620		CMP	#TTYCTL,WAITF		;KEYBOARD CONTROL?
29100	005060	001101				BNE	26\$;SKIP INPUT CHECK IF NOT
29200	005062	105777	173514		19\$:	TSTB	@CONADD		;TEST IF ANY INPUT
29300	005066	100076				BPL	26\$;BRANCH IF NONE
29400	005070	016767	173506	173614		MOV	CONADD,TEMP		;SET ADR
29500	005076	062767	000002	173606		ADD	#2,TEMP		
29600	005104	117767	173602	173566		MOVB	@TEMP,TEMPCH		
29700	005112	042767	177600	173560		BIC	#177600,TEMPCH		;MASK UNWANTED BITS
29800	005120	026727	173554	000003		CMP	TEMPCH,#3		;CHARACTER = CONTROL-C?
29900	005126	001013				BNE	21\$;CONTINUE IF NOT
30000	005130	026727	173502	000024		CMP	RTNNO,#24		;TEST 24?
30100	005136	001007				BNE	21\$;CONTINUE IF NOT
30200	005140	012700	000036		20\$:	MOV	#30.,R0		;DELAY FOR HALF DUPLEX
30300	005144	104010				DELAY			

32800

100
 200
 300
 400
 500
 600
 700
 800
 900
 1000
 1100
 1200
 1300
 1400
 1500
 1600
 1700
 1800
 1900
 2000
 2100
 2200
 2300
 2400
 2500
 2600
 2700
 2800
 2900
 3000
 3100
 3200
 3300
 3400
 3500
 3600
 3700
 3800
 3900
 4000
 4100
 4200
 4300
 4400
 4500
 4600
 4700

005266 000040
 005270 005320
 005272 000012
 005274 005304
 005276 012767 005314 172500
 005304 005777 173302
 005310 104005
 005312 000774
 005314 104001
 005316 000774
 005320 000041
 005322 005352
 005324 000012
 005326 005336
 005330 012767 005346 172446
 005336 005777 173252
 005342 104005
 005344 000774
 005346 104001
 005350 000774

```

.SBTTL I/O LOGIC TESTS
:
:*****
:ONLY THE CONSOLE TERMINAL IS TESTED.
:UPON COMPLETION, THE CPU WILL EITHER HALT IF SR
:BIT8 IS = 1 AND AWAIT FUTHER INSTRUCTIONS OR CONTINUE
:AND EXECUTE THE PRINTER TESTS CONTINUOUSLY
:IF AN I/O TEST FAILS, THE CPU WILL HALT AT ERRHLT
:WITH THE ADDRESS OF THE ERROR IN RO (LOC 777700). PRESSING
:THE CONTINUE SWITCH WILL CAUSE THE I/O TEST TO
:CONTINUE WITH THE NEXT TEST. HOWEVER IF SWITCH 14
:WERE SET, OR IS SET BEFORE THE CONTINUE SWITCH IS
:PRESSED, THE FAILED TEST WILL LOOP ON ITSELF
:WITHOUT FURTHFR HALTS
:
:*****
:ATO-- TEST #40--TESTS THE ABILITY TO REFERENCE THE
:RECEIVER STATUS WORD (TKS) WITHOUT TRAPPING.
:*****
ATO: 40 ;TEST NUMBER
ATOX: AT1 ;NEXT TEST
10. ;ITERATION COUNT
1$ ;SCOPE ENTRY
MOV #3$,MACHER ;SET UP MACHINE ERROR TRAP
1$: TST @TKS ;REFERENCE RECEIVER STATUS WORD
2$: CHAIN ;CHAIN TO NEXT TEST
BR 1$ ;REPEAT TEST
3$: ERROR ;ERROR TRAPPED WHEN REFERENCING
BR 2$ ;RECEIVER STATUS WORD (TKS)
:
:*****
:AT1--TEST #41--TESTS THE ABILITY TO REFERENCE THE
:RECEIVER BUFFER (TKB) WITHOUT TRAPPING.
:*****
AT1: 41 ;TEST NUMBER
AT2 ;NEXT TEST
10. ;ITERATION COUNT
1$ ;SCOPE ENTRY
MOV #3$,MACHER ;SET UP MACHINE ERROR TRAP
1$: TST @TKB ;REFERENCE RECEIVER BUFFER
2$: CHAIN ;CHAIN TO NEXT TEST
BR 1$ ;REPEAT TEST
3$: ERROR ;TRAPPED WHEN REFERENCING
BR 2$ ;RECEIVER BUFFER (TKB)

```

```

4900
5000
5100
5200
5300
5400 005352 000042
5500 005354 005404
5600 005356 000012
5700 005360 005370
5800 005362 012767 005400 172414
5900 005370 005777 173222
6000 005374 104005
6100 005376 000774
6200 005400 104001
6300 005402 000774
6400
6500
6600
6700
6800
6900
7000 005404 000043
7100 005406 005436
7200 005410 000012
7300 005412 005422
7400 005414 012767 005432 172362
7500 005422 005777 173172
7600 005426 104005
7700 005430 000774
7800 005432 104001
7900 005434 000774

:*****
:AT2--TEST #42--TESTS THE ABILITY TO REFERENCE THE
: TRANSMITTER STATUS WORD (TPS) WITHOUT TRAPPING.
:*****
AT2: 42 :TEST NUMBER
      AT3 :NEXT TEST
      10. :ITERATION COUNT
      1$ :SCOPE ENTRY
      MOV #3$,MACHER :SET UP MACHINE ERROR TRAP
      TST @TPS :REFERENCE TRANSMITTER STATUS
1$: :CHAIN TO NEXT TEST
2$: CHAIN :CHAIN TO NEXT TEST
      BR 1$ :REPEAT TEST
3$: ERROR 1$ :TRAPPED WHEN REFERENCING
      BR 2$ :TRANSMITTER STATUS WORD

:*****
:AT3-- TEST #43--TESTS THE ABILITY TO REFERENCE THE
: TRANSMITTER BUFFER (TPB) WITHOUT TRAPPING.
:*****
AT3: 43 :TEST NUMBER
      AT4 :NEXT TEST
      10. :ITERATION COUNT
      1$ :SCOPE ENTRY
      MOV #3$,MACHER :SET UP ERROR TRAP
      TST @TPB :REFERENCE TRANSMITTER BUFFER
1$: :CHAIN TO NEXT TEST
2$: CHAIN :CHAIN TO NEXT TEST
      BR 1$ :REPEAT TEST
3$: ERROR 1$ :TRAPPED WHEN REFERENCING
      BR 2$ :TRANSMITTER BUFFER.
    
```



```

8100
8200
8300
8400
8500
8600 005436 000044
8700 005440 005526
8800 005442 000012
8900 005444 005460
9000 005446 012746 000340
9100 005452 012746 005460
9200 005456 000002
9300 005460 052777 000100 173124 1$:
9400 005466 032777 000100 173116
9500 005474 001002
9600 005476 104001 2$:
9700 005500 000410
9800 005502 042777 000100 173102 3$:
9900 005510 032777 000100 173074
10000 005516 001401
10100 005520 104001 4$:
10200 005522 104005 5$:
10300 005524 000755
10400
10500
10600
10700
10800
10900
11000 005526 000045
11100 005530 005604
11200 005532 000012
11300 005534 005550
11400 005536 012746 000340
11500 005542 012746 005550
11600 005546 000002
11700 005550 052777 000100 173034 1$:
11800 005556 105777 173034 3$:
11900 005562 001775
12000 005564 000005
12100 005566 032777 000100 173016
12200 005574 001401
12300 005576 104001
12400 005600 104005 2$:
12500 005602 000762

:*****
:AT4-- TEST #44--TESTS THE ABILITY TO SET AND CLEAR THE
: RECEIVER INTERRUPT ENABLE BIT.
:*****
AT4: 44 ;TEST NUMBER
AT5 ;NEXT TEST
10. ;ITERATION COUNT
1$ ;SCOPE ENTRY
MOV #PRTY7,-(SP) ;SET PRIORITY 7
MOV #1$,-(SP)
RTI
BIS #BIT6,@TKS ;SET INTERRUPT ENABLE BIT
BIT #BIT6,@TKS ;CHECK IF BIT IS SET
BNE 3$ ;BRANCH IF SET
2$: ERROR ;NOT SET, ERROR
BR 5$ ;CHAIN TO NEXT TEST
BIC #BIT6,@TKS ;CLEAR INTERRUPT ENABLE BIT
BIT #BIT6,@TKS ;CHECK IF BIT IS CLEARED
BEQ 5$ ;BRANCH IF CLEARED
4$: ERROR ;NOT CLEARED, ERROR
5$: CHAIN ;CHAIN TO NEXT TEST
BR 1$ ;DO TEST AGAIN

:*****
:AT5-- TEST #45--CHECKS THAT THE RECEIVER INTERRUPT
: ENABLE BIT CAN BE CLEARED WITH RESET INSTRUCTION.
:*****
AT5: 45 ;TEST NUMBER
AT6 ;NEXT TEST
10. ;ITERATION COUNT
1$ ;SCOPE ENTRY
MOV #PRTY7,-(SP) ;SET PRIORITY TO 7
MOV #1$,-(SP)
RTI
BIS #BIT6,@TKS ;SET INTERRUPT ENABLE BIT
TSTB @TPS ;BE SURE PRINTER IS DONE WITH DL11S1 MESSAGE
BEQ 3$ ;BEFORE ALLOWING FOLLOWING RESET.
RESET ;RESET
BIT #BIT6,@TKS ;TEST INTERRUPT ENABLE BIT
BEQ 2$ ;BRANCH IF CLEARED
2$: ERROR ;STILL SET,ERROR
CHAIN ;CHAIN TO NEXT ROUTINE
BR 1$ ;REPEAT TEST
    
```

```

12700
12800
12900
13000
13100
13200 005604 000046
13300 005606 005674
13400 005610 000012
13500 005612 005626
13600 005614 012746 000340
13700 005620 012746 005626
13800 005624 000002
13900 005626 052777 000100 172762
14000 005634 032777 000100 172754
14100 005642 001002
14200 005644 104001
14300 005646 000410
14400 005650 042777 000100 172740
14500 005656 032777 000100 172732
14600 005664 001401
14700 005666 104001
14800 005670 104005
14900 005672 000755
15000
15100
15200
15300
15400
15500
15600 005674 000047
15700 005676 005744
15800 005700 000012
15900 005702 005716
16000 005704 012746 000340
16100 005710 012746 005716
16200 005714 000002
16300 005716 052777 000100 172672
16400 005724 000005
16500 005726 032777 000100 172662
16600 005734 001401
16700 005736 104001
16800 005740 104005
16900 005742 000765

:*****
:AT6-- TEST#46--TESTS THE ABILITY TO SET AND CLEAR
: TRANSMITTER INTERRUPT ENABLE BIT.
:*****
AT6: 46 ;TEST NUMBER
AT7 ;NEXT TEST
10. ;ITERATION COUNT
1$ ;SCOPE ENTRY
MOV #PRTY7,-(SP) ;SET PRIORITY TO 7
MOV #1$,-(SP)
RTI
1$: BIS #BIT6,@TPS ;SET INTERRUPT ENABLE BIT
BIT #BIT6,@TPS ;CHECK THAT BIT IS SET
BNE 2$ ;BRANCH IF SET
ERROR ;NOT SET, ERROR
BR 3$ ;CHAIN TO NEXT TEST
2$: BIC #BIT6,@TPS ;CLEAR INTERRUPT ENABLE BIT
BIT #BIT6,@TPS ;CHECK IF BIT IS CLEARED
BEQ 3$ ;BRANCH IF CLEARED
ERROR ;NOT CLEARED, ERROR
3$: CHAIN ;CHAIN TO NEXT TEST
BR 1$ ;DO AGAIN

:*****
:AT7-- TEST #47--TESTS THE ABILITY TO CLEAR TRANSMITTER
: INTERRUPT ENABLE BIT WITH RESET INSTRUCTION.
:*****
AT7: 47 ;TEST NUMBER
AT10 ;NEXT TEST
10. ;ITERATION COUNT
1$ ;SCOPE ENTRY
MOV #PRTY7,-(SP) ;SET PRIORITY TO 7
MOV #1$,-(SP)
RTI
1$: BIS #BIT6,@TPS ;SET INTERRUPT BIT
RESET ;RESET
BIT #BIT6,@TPS ;CHECK IF BIT IS CLEARED
BEQ 2$ ;BRANCH IF CLEARED
ERROR ;ERROR, RESET DID NOT CLEAR BIT
2$: CHAIN ;CHAIN TO NEXT ROUTINE
BR 1$ ;REPEAT TEST

```

```
17100
17200
17300
17400
17500
17600 005744 000050
17700 005746 006002
17800 005750 000012
17900 005752 005754
18000 005754 032777 001000 172732 1$:
18100 005762 001005
18200 005764 000005
18300 005766 105777 172624
18400 005772 100401
18500 005774 104001
18600 005776 104005
18700 006000 000765
18800
18900
19000
19100
19200
19300
19400 006002 000051
19500 006004 006042
19600 006006 000012
19700 006010 006012
19800 006012 012700 000226
19900 006016 104010
20000 006020 000005
20100 006022 005077 172572
20200 006026 105777 172564
20300 006032 100001
20400 006034 104001
20500 006036 104005
20600 006040 000764

:*****
:AT10-- TEST #50--CHECKS THAT RESET SETS THE TRANSMITTER
: READY BIT AND THAT THE READY BIT CAN BE READ RELIABLY.
:*****
AT10: 50 ;TEST NUMBER
      AT11 ;NEXT TEST
      10. ;ITERATION COUNT
      1$ ;SCOPE ENTRY
1$: BIT #LSI11,@SR ;SKIP TEST IF AN LSI-11
   BNE 2$
   RESET ;RESET
   TSTB @TPS ;CHECK TRANSMIT READY BIT
   BMI 2$ ;BRANCH IF SET
   ERROR ;ERROR, RESET DID NOT SET READY BIT
2$: CHAIN ;CHAIN TO NEXT TEST
   BR 1$ ;DO AGAIN

:*****
:AT11-- TEST #51--TESTS THAT THE TRANSMITTER READY RESETS
: BY LOADING THE TRANSMITTER BUFFER.
:*****
AT11: 51 ;TEST NUMBER
      AT12 ;NEXT TEST
      10. ;ITERATION COUNT
      1$ ;SCOPE ENTRY
1$: MOV #226,R0 ;DELAY 150 MSEC.
   DELAY 150 MSEC.
   RESET ;RESET
   CLR @TPB ;LOAD TRANSMITTER BUFFER
   TSTB @TPS ;CHECK TRANSMIT READY BIT
   BPL 2$ ;BRANCH IF CLEARED
   ERROR ;NOT CLEARED, ERROR
2$: CHAIN ;CHAIN TO NEXT TEST
   BR 1$ ;REPEAT TEST
```

```

20800
20900
21000
21100
21200
21300 006042 000052
21400 006044 006116
21500 006046 000012
21600 006050 006056
21700 006052 104004
21800 006054 006112
21900 006056 000005
22000 006060 005077 172532
22100 006064 005046
22200 006066 012746 006074
22300 006072 000002
22400 006074 052777 000100 172514
22500 006102 000240
22600 006104 104001
22700 006106 104005
22800 006110 000762
22900 006112 022626
23000 006114 000774
23100
23200
23300
23400
23500
23600
23700 006116 000035
23800 006120 006176
23900 006122 000012
24000 006124 006132
24100 006126 104004
24200 006130 006170
24300 006132 016746 172472
24400 006136 012746 006144
24500 006142 000002
24600 006144 005077 172446
24700 006150 052777 000100 172440
24800 006156 000240
24900 006160 005077 172432
25000 006164 104005
25100 006166 000761
25200 006170 022626
25300 006172 104001
25400 006174 000771

:*****
:AT12-- TEST #52--CHECKS THAT THE TRANSMIT READY BIT CAN
: CAUSE AN INTERRUPT
:*****
AT12: 52 ;TEST NUMBER
      AT13 ;NEXT TEST
      10. ;ITERATION COUNT
      1$ ;SCOPE ENTRY
      STPCHV ;SET UP TRANSMITTER INTERRUPT VECTOR
      4$ ;TO 4$
1$: RESET ;SEE CHAINY COMMENT
   CLR @TPS ;DISABLE TRANSMIT INTERRUPT
   CLR -(SP) ;SET PRIORITY TO ZERO
   MOV #2$,-(SP)
   RTI
2$: BIS #BIT6,@TPS ;ENABLE TRANSMIT INTERRUPT
   NOP
   ERROR ;TRANSMIT READY DID NOT CAUSE INTERRUPT
3$: CHAIN ;CHAIN TO NEXT TEST
   BR 1$ ;REPEAT TEST
4$: POPSP2 ;INTERRUPT OCCURRED, CLEAN STACK
   BR 3$ ;CHAIN TO NEXT TEST

:*****
:AT13-- TEST#53--TESTS THAT THE TRANSMIT READY DOES NOT CAUSE AN
: INTERRUPT WHEN THE PROCESSOR IS AT THE SAME LEVEL
:*****
AT13: 35 ;TEST NUMBER
      AT14 ;NEXT TEST
      10. ;ITERATION COUNT
      1$ ;SCOPE ENTRY
      STPCHV ;SET UP TRANSMIT INTERRUPT
      4$ ;VECTOR TO 4$
1$: MOV TPLVL,-(SP) ;SET PROCESSOR TO SAME LEVEL AS XMITTER
   MOV #2$,-(SP)
   RTI
2$: CLR @TPS ;DISABLE TRANSMITTER INTERRUPTS
   BIS #BIT6,@TPS ;ENABLE TRANSMITTER INTERRUPTS
   NOP
3$: CLR @TPS ;OK, NO INTERRUPT OCCURRED
   CHAIN ;CHAIN TO NEXT TEST
   BR 1$ ;REPEAT TEST
4$: POPSP2 ;INTERRUPT OCCURRED,ERROR,CLEAN
   ERROR ;UP STACK
   BR 3$ ;CHAIN TO NEXT TEST
  
```

```

25600
25700
25800
25900
26000
26100
26200 006176 000054
26300 006200 006262
26400 006202 000012
26500 006204 006212
26600 006206 104004
26700 006210 006250
26800 006212 005077 172400
26900 006216 016746 172406
27000 006222 162716 000040
27100 006226 012746 006234
27200 006232 000002
27300 006234 052777 000100 172354
27400 006242 000240
27500 006244 104001
27600 006246 000401
27700 006250 022626
27800 006252 005077 172340
27900 006256 104005
28000 006260 000754

:*****
:AT14-- TEST#54--TESTS THAT THE TRANSMIT READY DOES CAUSE AN
:      INTERRUPT WHEN THE PROCESSOR IS AT A PRIORITY LEVEL
:      ONE LOWER THAN THE TRANSMIT INTERRUPT REQUEST LEVEL
:*****
AT14:  54          ;TEST NUMBER
        AT15       ;NEXT TEST
        10.        ;ITERATION COUNT
        1$        ;SCOPE ENTRY
        STPCHV     ;SET UP TRANSMIT INTERRUPT
        3$        ;VECTOR TO 3$
1$:    CLR        @TPS ;DISABLE TRANSMIT INTERRUPTS
        MOV        TPLVL,-(SP) ;SET PROCESSOR PRIORITY ONE
        SUB        #40,(SP) ;LEVEL LOWER THAN TRANSMITTER
        MOV        #2$,-(SP)
        RTI
2$:    BIS        #BIT6,@TPS ;ENABLE TRANSMITTER INTERRUPTS
        NOP
        ERROR     ;NO INTERRUPT, ERROR
        BR        4$ ;CHAIN TO NEXT TEST
3$:    POPSP2
4$:    CLR        @TPS ;INTERRUPT OCCURED, OK, CLEAN STACK
        CHAIN     ;DISABLE TRANSMITTER INTERRUPTS
        BR        1$ ;CHAIN TO NEXT TEST
        ;REPEAT TEST
    
```

```

28200
28300
28400
28500
28600
28700
28800 006262 000055
28900 006264 006362
29000 006266 000012
29100 006270 006272
29200 006272 104004
29300 006274 006334
29400 006276 005077 172314
29500 006302 005046
29600 006304 012746 006312
29700 006310 000002
29800 006312 052777 000100 172276
29900 006320 000240
30000 006322 104001
30100 006324 005077 172266
30200 006330 104005
30300 006332 000757
30400 006334 012777 006354 172264
30500 006342 012716 006350
30600 006346 000002
30700 006350 000240
30800 006352 000764
30900 006354 022626
31000 006356 104001
31100 006360 000761
31200
31300
31400
31500
31600
31700 006362 000056
31800 006364 006430
31900 006366 000012
32000 006370 006372
32100 006372 032777 001000 172314
32200 006400 001011
32300 006402 012700 000226
32400 006406 104010
32500 006410 104021
32600 006412 000005
32700 006414 105777 172172
32800 006420 100001
32900 006422 104001
33000 006424 104005
33100 006426 000761

;*****
;AT15-- TEST#55--TESTS THAT THE TRANSMIT READY DOES NOT
;          REINTERRUPT AFTER AN RTI WHEN THE READY BIT HAS
;          NOT BEEN RESET.
;*****
AT15: 55 ;TEST NUMBER
      AT16 ;NEXT TEST
      10. ;ITERATION COUNT
      1$ ;SCOPE ENTRY
1$: STPCHV ;SET TRANSMIT INTERRUPT VECTOR
   4$ ;TO 4$
   CLR @TPS ;DISABLE TRANSMITTER INTERRUPTS
   CLR -(SP) ;SET PROCESSOR PRIORITY TO ZERO
   MOV #2$,-(SP)
   RTI
2$: BIS #BIT6,@TPS ;ENABLE TRANSMITTER INTERRUPTS
   NOP
   ERROR ;ERROR1, TRANSMITTER FAILED TO INTERRUPT
3$: CLR @TPS ;DISABLE TRANSMITTER INTERRUPTS
   CHAIN ;CHAIN TO NEXT TEST
   BR 1$ ;REPEAT TEST
4$: MOV #6$,@TPVTR ;INTERRUPT OCCURRED, CHANGE INTERRUPT
   MOV #5$,@SP ;VECTOR TO 6$ AND RETURN TO 5$
   RTI ;RETURN FROM INTERRUPT
5$: NOP
   BR 3$ ;CHAIN TO NEXT TEST
6$: POPSP2 ;ERROR2, TRANSMITTER REINTERRUPTED
   ERROR ;AFTER RTI WITH READY BIT LEFT ON.
   BR 3$ ;CLEAN STACK, CHAIN TO NEXT TEST.

```

```

;*****
;AT16--TEST#56--CHECKS THAT RESET CLEARS THE RECEIVER DONE BIT
;*****
AT16: 56 ;TEST NUMBER
      AT17 ;NEXT TEST
      10. ;ITERATION COUNT
      1$ ;SCOPE ENTRY
1$: BIT #LSI11,@SR ;SKIP TEST IF LSI-11
   BNE 3$
   MOV #226,R0
2$: DELAY 150 MSEC.
   AREAD ;ENABLE RECEIVER
   RESET ;RESET
   TSTB @TKS ;TEST DONE BIT
   BPL 3$ ;BRANCH IF DONE IS CLEARED
3$: ERROR ;NOT CLEARED, ERROR
   CHAIN ;CHAIN TO NEXT TEST
   BR 1$ ;REPEAT TEST

```

```

33300
33400
33500
33600
33700
33800 006430 000057
33900 006432 006500
34000 006434 000012
34100 006436 006440
34200 006440 032777 001000 172246 1$
34300 006446 001012
34400 006450 012700 000226
34500 006454 104010
34600 006456 104021
34700 006460 105777 172130
34800 006464 105777 172122
34900 006470 100001
35000 006472 104001
35100 006474 104005
35200 006476 000760
35300
35400
35500
35600
35700
35800
35900 006500 000060
36000 006502 006572
36100 006504 000012
36200 006506 006514
36300 006510 104003
36400 006512 006564
36500 006514 032777 001000 172172 1$
36600 006522 001021
36700 006524 012700 000226
36800 006530 104010
36900 006532 104021
37000 006534 005077 172052
37100 006540 005046
37200 006542 012746 006550
37300 006546 000002
37400 006550 052777 000100 172034 3$
37500 006556 000240
37600 006560 104001
37700 006562 000401
37800 006564 022626
37900 006566 104005
38000 006570 000751

:*****
:AT17-- TEST#57--CHECKS THAT REFERENCING THE RECEIVER BUFFER
:          CLEARS THE DONE BIT.
:*****
AT17:  57          ;TEST NUMBER
        AT20       ;NEXT TEST
        10.        ;ITERATION COUNT
        1$        ;SCOPE ENTRY
        BIT        #LSI11,@SR ;CHECK FOR LSI-11
        BNE       3$ ;SKIP TEST IF SET
        MOV       #226,R0
        DELAY     ;DELAY 150 MSEC.
        AREAD     ;ENABLE RECEIVER
        TSTB     @TKB ;REFERENCE RECEIVER BUFFER
        TSTB     @TKS ;TEST DONE BIT
        BPL      3$ ;BRANCH IF NOT SET
        ERROR    ;DONE BIT IS SET, ERROR
        CHAIN    ;CHAIN TO NEXT TEST
        BR       1$ ;REPEAT TEST

:*****
:AT20-- TEST#60--CHECK THAT THE RECEIVER DONE BIT IS ABLE TO
:          CAUSE AN INTERRUPT.
:*****
AT20:  60          ;TEST NUMBER
        AT21       ;NEXT TEST
        10.        ;ITERATION COUNT
        1$        ;SCOPE ENTRY
        STRDRV    ;SET UP RECEIVER INTERRUPT
        4$        ;VECTOR TO 4$
        BIT       #LSI11,@SR ;CHECK FOR LSI-11
        BNE      5$ ;SKIP TEST IF SET
        MOV      #226,R0
        DELAY    ;DELAY 150 MSEC
        AREAD   ;ENABLE RECEIVER
        CLR     @TKS ;DISABLE RECEIVER INTERRUPTS
        CLR    -(SP) ;SET PROCESS STATUS TO ZERO
        MOV    #3$,-(SP)
        RTI
        BIS   #BIT6,@TKS ;ENABLE RECEIVER INTERRUPT
        NOP
        ERROR ;ERROR,RECEIVER FAILED TO INTERRUPT
        BR   5$ ;CHAIN TO NEXT TEST
        4$  POPSP2 ;OK, CLEAN STACK
        5$  CHAIN  ;CHAIN TO NEXT TEST
        BR   1$ ;REPEAT TEST
    
```

J 7

```
38200
38300
38400
38500
38600
38700
38800 006572 000061
38900 006574 006672
39000 006576 000012
39100 006600 006606
39200 006602 104003
39300 006604 006664
39400 006606 032777 001000 172100 1$:
39500 006614 001017
39600 006616 012700 000226
39700 006622 104010
39800 006624 104021
39900 006626 005077 171760
40000 006632 016746 171766
40100 006636 012746 006644
40200 006642 000002
40300 006644 052777 000100 171740 3$:
40400 006652 000240
40500 006654 005077 171732 4$:
40600 006660 104005
40700 006662 000751
40800 006664 022626 5$:
40900 006666 104001
41000 006670 000771
```

```
*****
:AT21-- TEST#61--TESTS THAT THE RECEIVER DONE DOES NOT CAUSE AN
:          INTERRUPT WHEN THE PROCESSOR IS AT THE SAME LEVEL AS
:          THE RECEIVER'S INTERRUPT REQUEST LEVEL.
*****
AT21: 61          ;TEST NUMBER
      AT22       ;NEXT TEST
      10.       ;ITERATION COUNT
      1$        ;SCOPE ENTRY
      STRDRV    ;SET RECEIVER VECTOR TO 5$
      5$
      BIT #LSI11,@SR ;CHECK FOR LSI-11
      BNE 4$    ;SKIP TEST IF SET
      MOV #226,R0
      DELAY    ;DELAY 150 MSEC
      AREAD   ;ENABLE RECEIVER
      CLR @TKS ;DISABLE RECEIVER INTERRUPTS
      MOV TKLVL,-(SP) ;SET PROCESSOR PRIORITY TO SAME LEVEL AS RECEIVER
      MOV #3$,-(SP)
      RTI
      BIS #BIT6,@TKS ;ENABLE RECEIVER INTERRUPTS
      NOP
      CLR @TKS ;OK, NO INTERRUPT OCCURRED
      CHAIN   ;CHAIN TO NEXT TEST
      BR 1$   ;REPEAT TEST
      POPSP2 ;ERROR, RECEIVER INTERRUPTED, CLEAN STACK
      ERROR
      BR 4$   ;BRANCH 4$
```



```

41200
41300
41400
41500
41600
41700
41800
41900 006672 000062
42000 006674 007000
42100 006676 000012
42200 006700 006706
42300 006702 104003
42400 006704 006766
42500 006706 032777 001000 172000 1$:
42600 006714 001025
42700 006716 012700 000226
42800 006722 104010
42900 006724 104021
43000 006726 005077 171660
43100 006732 016746 171666
43200 006736 012746 006744
43300 006742 000002
43400 006744 162767 000040 171024 3$:
43500 006752 052777 000100 171632
43600 006760 000240
43700 006762 104001
43800 006764 000401
43900 006766 022626
44000 006770 005077 171616
44100 006774 104005
44200 006776 000743

:*****
:AT22-- TEST#62--TESTS THAT THE RECEIVER DONE DOES CAUSE AN
:          INTERRUPT WHEN THE PROCESSOR IS AT A PRIORITY ONE
:          LEVEL LOWER THAN THE RECEIVER'S INTERRUPT
:          REQUEST LEVEL
:*****
AT22: 62 ;TEST NUMBER
      AT23 ;NEXT TEST
      10. ;ITERATION COUNT
      1$ ;SCOPE ENTRY
      STRDRV ;SET RECEIVER INTERRUPT
      4$ ;VECTOR TO 4$
      BIT #LSI11,@SR ;CHECK FOR LSI11
      BNE 5$ ;SKIP TEST IF SET
      MOV #226,R0
      DELAY ;DELAY 150 MSEC
      AREAD ;ENABLE RECEIVER
      CLR @TKS ;DISABLE READER INTERRUPTS
      MOV TKLVL,-(SP) ;SET PROCESSOR PRIORITY ONE LEVEL
      MOV #3$,-(SP)
      RTI
      SUB #40,PSW ;LOWER THAN READER
      BIS #BIT6,@TKS ;ENABLE INTERRUPTS
      NOP
      ERROR ;FAILED TO INTERRUPT
      BR 5$ ;CHAIN TO NEXT TEST
      POPSP2 ;OK, CLEAN STACK
      CLR @TKS ;DISABLE RECEIVER INTERRUPTS
      CHAIN ;CHAIN TO NEXT TEST
      BR 1$ ;REPEAT TEST
    
```

```

44400
44500
44600
44700
44800
44900
45000 007000 000063
45100 007002 007112
45200 007004 000012
45300 007006 007010
45400 007010 032777 001000 171676 1$: BIT #LSI11,@SR
45500 007016 001015 BNE 3$
45600 007020 012700 000226 2$: MOV #226,R0
45700 007024 104010 DELAY
45800 007026 104021 AREAD
45900 007030 104003 STRDRV
46000 007032 007064 4$
46100 007034 005077 171552 CLR @TKS
46200 007040 052777 000100 171544 BIS #BIT6,@TKS
46300 007046 000240 NOP
46400 007050 104001 ERROR
46500 007052 005077 171534 3$: CLR @TKS
46600 007056 000005 RESET
46700 007060 104005 CHAIN
46800 007062 000752 BR 1$
46900 007064 012777 007104 171530 4$: MOV #6$,@TKVTR
47000 007072 012716 007100 MOV #5$,@SP
47100 007076 000002 RTI
47200 007100 000240 5$: NOP
47300 007102 000763 BR 3$
47400 007104 022626 6$: POPSP2
47500 007106 104001 ERROR
47600 007110 000760 BR 3$

```

```

:*****
:AT23-- TEST#63--CHECKS THAT THE RECEIVER DONE DOES NOT
: REINTERRUPT AFTER RTI INSTRUCTION WHEN DONE
: BIT IS LEFT SET.
:*****
:TEST NUMBER
:NEXT TEST
:ITERATION COUNT
:SCOPE ENTRY
:CHECK FOR LSI-11
:SKIP TEST IF SET
:DELAY 150 MSEC
:ENABLE RECEIVER
:SET RECEIVER INTERRUPT
:VECTOR TO 4$
:DISABLE RECEIVER INTERRUPTS
:ENABLE RECEIVER INTERRUPT
:
:NO INTERRUPT, ERROR
:DISABLE RECEIVER INTERRUPTS
:RESET AFTER LAST INTERRUPT
:CHAIN TO NEXT TEST
:REPEAT TEST
:INTERRUPT, OK, CHANGE VECTOR TO 6$
:CHANGE RET ADDR TO 5$
:RETURN
:OK, NO ADDITIONAL INTERRUPT
:ERROR, ADDITIONAL INTERRUPT
:CHAIN TO NEXT TEST

```

```

47800
47900
48000
48100
48200
48300
48400
48500 007112 000064
48600 007114 007212
48700 007116 000001
48800 007120 007122
48900 007122 032777 001000 171564 1$:
49000 007130 001426
49100 007132 005777 171454
49200 007136 001401
49300 007140 104001
49400 007142 012700 000600
49500 007146 012767 000030 171542 2$:
49600 007154 104000
49700 007156 014401
49800 007160 104010
49900 007162 105777 171424
50000 007166 100407
50100 007170 005367 171522
50200 007174 001403
50300 007176 012700 000600
50400 007202 000766
50500 007204 104001
50600
50700 007206 104005
50800 007210 000744

:*****
:AT24--TEST#64--HAVE OPERATOR TYPE A CHARACTER ON THE
:      KEYBOARD, THEN CHECK FOR RECEIVER DONE.
:
:      ALLOW 12 SECONDS FOR OPERATOR RESPONSE.
:*****
AT24:  64          ;TEST NUMBER
      AT25        ;NEXT TEST
      1           ;ITERATION COUNT
      1$         ;SCOPE ENTRY
      BIT        #LSI11,@SR ;SKIP TEST IF NOT AN LSI-11
      BEQ        5$
      TST        @TKS      ;SHOULD BE CLEAR
      BEQ        2$
      ERROR      ;RECEIVER STATUS NOT =0
      MOV        #600,R0   ;1/2 SEC DELAY
      MOV        #30,CNTR ;SET UP FOR 12 SEC WAIT
      TYPE
      OPMSG      ;MESSAGE TO TYPE A CHARACTER
      DELAY      ;1/2 SECOND
      TSTB       @TKS     ;CHECK DONE BIT
      BMI        5$      ;SET - EXIT LOOP
      DEC        CNTR
      BEQ        4$
      MOV        #600,R0   ;TIME HAS RUN OUT...
      BR         3$      ;ANOTHER 1/2 SEC
      ERROR      ;CONTINUE WAIT
      ERROR      ;NO RECEIVER DONE, OR
      CHAIN      ;OPERATOR DID NOT RESPOND
      BR         1$      ;CHAIN TO NEXT TEST
  
```

```
51000
51100
51200
51300
51400
51500 007212 000065
51600 007214 007270
51700 007216 000001
51800 007220 007222
51900 007222 032777 001000 171464 1$: BIT #LSI11,@SR
52000 007230 001415
52100 007232 105777 171354 2$: TSTB @TKS
52200 007236 001001
52300 007240 104001
52400 007242 104003
52500 007244 007262
52600 007246 052777 000100 171336
52700 007254 000240
52800 007256 000240
52900 007260 104001
53000 007262 022626
53100 007264 104005
53200
53300 007266 000755

:*****
:AT25--TEST#65--CHECK THAT RECEIVER DONE CAUSES AN INTERRUPT
:                               WHEN BIT 6 (INTERRUPT ENABLE) IS SET.
:*****
AT25: 65 ;TEST NUMBER
      AT26 ;NEXT TEST
      1 ;ITERATION COUNT
      1$ ;SCOPE ENTRY
      BEQ #LSI11,@SR ;SKIP TEST IF NOT AN LSI-11
      6$
      3$ ;DONE SHOULD BE SET
      BNE
      ERROR ;RECEIVER DONE NOT SET
      STRDRV ;SET RECEIVER INTERRUPT
      5$ ;VECTOR TO 5$
      BIS #BIT6,@TKS ;ENABLE INTERRUPT
      NOP
      NOP
      4$: ERROR ;RECEIVER DID NOT INTERRUPT
      5$: POPSP2 ;CLEAN UP THE STACK
      6$: CHAIN ;CHAIN TO NEXT TEST
      BR 1$
```

```

53500
53600
53700
53800
53900
54000 007270 000066
54100 007272 177777
54200 007274 000001
54300 007276 007300
54400 007300 032777 001000 171406 1$: BIT #LSI11,@SR
54500 007306 001422 BEQ 5$
54600 007310 105777 171276 2$: TSTB @TKS
54700 007314 001001 BNE 3$
54800 007316 104001 ERROR
54900 007320 017767 171270 171370 3$: MOV @TKB,CNTR
55000 007326 105777 171260 TSTB @TKS
55100 007332 100001 BPL 4$
55200 007334 104001 ERROR
55300 007336 104003 4$: STRDRV
55400 007340 007364 6$
55500 007342 052777 000100 171242 BIS #BIT6,@TKS
55600 007350 000240 NOP
55700 007352 000240 NOP
55800 007354 005077 171232 5$: CLR @TKS
55900 007360 104005 CHAIN
56000 007362 000746 BR 1$
56100 007364 104001 6$: ERROR
56200 007366 022626 POPSP2
56300 007370 000771 BR 5$

```

```

:*****
:AT26--TEST#66--CHECK THAT READING TKB CLEARS DONE BIT
: AND THAT DONE CLEARED DOES NOT CAUSE AN INTERRUPT
:*****

```

```

;TEST NUMBER
;LAST TEST
;ITERATION COUNT
;SCOPE ENTRY
;SKIP TEST IF NOT AN LSI-11
;MAKE SURE DONE IS STILL SET
;RECEIVER DONE NOT SET
;READ DATA BUFFER
;CHECK THE DONE BIT
;OK
;READING DATA BUFFER DID NOT CLEAR DONE
;SET RECEIVER INTERRUPT
;VECTOR TO 6$
;ENABLE INTERRUPT
;OK- CLEAN UP
;EXIT TESTS
;DLV INTERRUPTED WITH DONE CLEAR
;CLEAN UP THE STACK
;EXIT TESTS

```

56500

100
 200
 300
 400
 500
 600
 700
 800
 900
 1000
 1100
 1200
 1300
 1400
 1500
 1600
 1700
 1800
 1900
 2000
 2100
 2200
 2300
 2400
 2500
 2600 007372 000000
 2700 007374 007446
 2800 007376 104016
 2900 007400 104007
 3000 007402 014127
 3100 007404 012703 025125
 3200 007410 012702 000004
 3300 007414 010300
 3400 007416 016701 171230
 3500 007422 104015
 3600 007424 000300
 3700 007426 005301
 3800 007430 001374
 3900 007432 000303
 4000 007434 104012
 4100 007436 005302
 4200 007440 001365
 4300 007442 104005
 4400 007444 000757

.SBTTL LA36 PRINTER TESTS

:THE LA36 PRINTER TESTS WILL BE EXECUTED IN A
 :CONTINOUS LOOP OUTPUTTING TO ALL MULTIPLE DL11'S
 :IF SR BIT 8 IS SET TO ZERO AT START UP TIME. IF
 :BIT 8 IS SET TO 1 AT START UP THEY MAY BE EXECUTED
 :INDIVIDUALLY ONCE OR CONTINUALLY LOOPED, OR
 :BECOME THE FIRST OF THE ENTIRE SEQUENCE OF PRINTER
 :TESTS. REFERENCE INTRUCTIONS IN THE INTRODUCTION
 :FOR PROPER MODE OF OPERATION.

:XXXXXXXXXX

:PT0 -- DATA PATH TEST---FOUR LINES OF ALTERNATING
 : '*' AND 'U' ARE PRINTED, OUT TO THE GIVEN PAPER
 : WIDTH. THE PATTERN WILL APPEAR AS FOLLOWS.

```
*U*U*U*U*U*U
U*U*U*U*U*U*
*U*U*U*U*U*U
U*U*U*U*U*U*
```

:XXXXXXXXXX

```
PT0: 0 ;TEST NUMBER
      PT1 ;NEXT TEST
      PRTHDR
      TYPEM ;PRINT COLUMN # MMSG
      HDRO
1$: MOV #'U',R3 ;SET FIRST CHAR PAIR
    MOV #4,R2 ;SET LINE COUNT
2$: MOV R3,R0 ;SET CHAR PAIR
    MOV WIDTH,R1 ;SET COLUMN COUNT
3$: PRINTC ;PRINT CHAR
    SWAB R0 ;SET NEXT CHAR
    DEC R1 ;DEC COLUMN COUNT
    BNE 3$ ;FINISH LINE
    SWAB R3 ;SET NEXT LINE START CHAR
    CRLF ;SEND CR-LF
    DEC R2 ;DEC LINE COUNT
    BNE 2$ ;FINISH TEST
    CHAIN ;ALL DONE, EXIT
    BR 1$ ;REPEAT TEST
```

```

4600          :XXXXXXXXXX
4700          :
4800          :PT1 -- PRINTER CHARACTER TEST --- PRINTS ALL PRINTABLE CHARACTERS
4900          :
5000          :XXXXXXXXXX
5100
5200 007446   000001   PT1:      1           ;TEST NUMBER
5300 007450   007570           PT2           ;NEXT TEST
5400 007452   104016           PRTHDR
5500 007454   012701   000040   1$:      MOV      #40,R1   ;SPACE TO R1
5600 007460   012702   000100           MOV      #100,R2      ;@ TO R2
5700 007464   012703   000140           MOV      #140,R3     ;\ TO R3
5800 007470   110100           2$:      MOVB     R1,R0   ;CHAR TO R0
5900 007472   004767   000042           JSR      PC,SPSP     ;SEND TWO SPACES
6000 007476   110200           MOVB     R2,R0       ;NEXT CHAR TO R0
6100 007500   004767   000034           JSR      PC,SPSP     ;SEND TWO SPACES
6200 007504   012704   000003           MOV      #3,R4      ;PRINT COUNT TO R4
6300 007510   110300           MOVB     R3,R0       ;THIRD CHAR TO R0
6400 007512   104015           3$:      PRINTC    ;PRINT THE CHAR
6500 007514   005304           DEC      R4          ;THREE TIMES ?
6600 007516   001375           BNE      3$          ;BRANCH IF NOT
6700 007520   104012           CRLF                    ;CARRIAGE RETURN LINE FEED
6800 007522   122122           CMPB     (R1)+,(R2)+  ;NEXT CHARACTERS
6900 007524   105723           TSTB     (R3)+
7000 007526   020327   000200           CMP      R3,#200    ;CHECK IF ALL DONE
7100 007532   103756           BLO      2$          ;BRANCH IF NOT
7200 007534   104005           CHAIN                    ;EXIT TO NEXT TEST
7300 007536   000746           BR       1$          ;REPEAT TEST
7400 007540   012704   000003   SPSP:    MOV      #3,R4   ;PRINT COUNT TO R4
7500 007544   104015           1$:      PRINTC    ;PRINT CHAR
7600 007546   005304           DEC      R4          ;THREE TIMES?
7700 007550   001375           BNE      1$          ;BRANCH IF NOT
7800 007552   012700   000040   SP2:    MOV      #40,R0   ;SPACE TO R0
7900 007556   104015           PRINTC    ;SEND A SPACE
8000 007560   012700   000040   SPC:    MOV      #40,R0   ;SPACE TO R0
8100 007564   104015           PRINTC    ;SEND ANOTHER
8200 007566   000207           RTS      PC          ;RETURN
    
```



```

8400 :XXXXXXXXXX
8500 :
8600 :PT2 -- NON-PRINTING CHARACTER TEST. THIS TEST
8700 :PRINTS THE OCTAL CODE FOLLOWED BY THE MNEMONIC
8800 :OF ALL NON-PRINTING CHARACTERS. FOLLOWING EACH
8900 :MNEMONIC, THE PRINTER IS DRIVEN BY THE NON-PRINTING
9000 :CODE (000 THROUGH 037 PLUS 177)
9100 :ALL CONTROL CHARACTERS (INCLUDING THOSE FOR OPTIONS
9200 :WILL BE SKIPPED, REFER TO THE DOCUMENT FOR A LIST OF THOSE
9300 :TESTED.
9400 :
9500 :XXXXXXXXXX
9600 :

```

```

9700 007570 000002 PT2: 2 ;TEST NUMBER
9800 007572 010164 PT3 ;NEXT TEST
9900 007574 104016 PRTHDR ;PRINT TEST HEADER
10000 007576 012701 007676 1$: MOV #IDEZ,R1 ;ADDR OF IDENT TO R1
10100 007602 012703 010137 MOV #NPCODE,R3 ;ADDR OF NON-PRINT-CODES TO R3
10200 007606 012702 000003 2$: MOV #3,R2 ;NO. OF ID'S PER LINE TO R2
10300 007612 012704 000010 3$: MOV #10,R4 ;NO. OF CHARS PER ID TO R4
10400 007616 121327 000055 4$: CMPB (R3),#55 ;ZERO TERMINATOR IN NP TABLE?
10500 007622 001422 BEQ 7$ ;BRANCH IF YES
10600 007624 112100 MOVB (R1)+,R0 ;GET ID CHARACTERS
10700 007626 104015 PRINTC ;AND PRINT A
10800 007630 005304 DEC R4 ;GROUP OF
10900 007632 001371 BNE 4$ ;8 CHARACTERS
11000 007634 112300 MOVB (R3)+,R0 ;GET NP CODE FROM TABLE
11100 007636 012704 000003 MOV #3,R4 ;AND
11200 007642 104015 5$: PRINTC ;TRY TO PRINT IT
11300 007644 005304 DEC R4 ;THREE
11400 007646 001375 BNE 5$ ;TIMES
11500 007650 005302 DEC R2 ;MORE TO GO ON THIS LINE ?
11600 007652 001404 BEQ 6$ ;BRANCH IF NO
11700 007654 004767 177672 JSR PC,SP2 ;SEND 3 SPACES
11800 007660 104015 PRINTC
11900 007662 000753 BR 3$ ;BRANCH TO CONTINUE LINE
12000 007664 104012 6$: CRLF
12100 007666 000747 BR 2$ ;GO DO NEXT LINE
12200 007670 104012 7$: CRLF
12300 007672 104005 CHAIN ;CHAIN TO NEXT TEST
12400 007674 000740 BR 1$
12500
12600
12700

```

12800	007676	060	060	060	IDEZ: .ASCII /000 NUL001 SOH002 STX/
	007701	040	040	116	
	007704	125	114	060	
	007707	060	061	040	
	007712	040	123	117	
	007715	110	060	060	
	007720	062	040	040	
	007723	123	124	130	
12900	007726	060	060	066	.ASCII /006 ACK020 DLE021 DC1/
	007731	040	040	101	
	007734	103	113	060	
	007737	062	060	040	
	007742	040	104	114	

	007745	105	060	062	
	007750	061	040	040	
	007753	104	103	061	
13000	007756	060	062	062	.ASCII /022 DC2023 DC3024 DC4/
	007761	040	040	104	
	007764	103	062	060	
	007767	062	063	040	
	007772	040	104	103	
	007775	063	060	062	
	010000	064	040	040	
	010003	104	103	064	
13100	010006	060	062	065	.ASCII /025 NAK026 SYN027 ETB/
	010011	040	040	116	
	010014	101	113	060	
	010017	062	066	040	
	010022	040	123	131	
	010025	116	060	062	
	010030	067	040	040	
13200	010033	105	124	102	
	010036	060	063	060	.ASCII /030 CAN031 EM 032 SUB/
	010041	040	040	103	
	010044	101	116	060	
	010047	063	061	040	
	010052	040	105	115	
	010055	040	060	063	
	010060	062	040	040	
	010063	123	125	102	
13300	010066	060	063	064	.ASCII /034 FS 035 GS 036 RS /
	010071	040	040	106	
	010074	123	040	060	
	010077	063	065	040	
	010102	040	107	123	
	010105	040	060	063	
	010110	066	040	040	
	010113	122	123	040	
13400	010116	060	063	067	.ASCII /037 US 177 DEL /
	010121	040	040	125	
	010124	123	040	061	
	010127	067	067	040	
	010132	040	104	105	
	010135	114	040		
13500	010137	000	002	006	NPCODE: .BYTE 0,2,6,20,21,22,23,24
	010142	020	021	022	
	010145	023	024		
13600	010147	025	025	027	.BYTE 25,25,27,30,31,32,34,35
	010152	030	031	032	
	010155	034	035		
13700	010157	036	037	177	.BYTE 36,37,177,55
13800	010162	055			.EVEN

```

14000          :XXXXXXXXXX
14100          :
14200          :PT3 -- CARRIAGE RETURN TEST
14300          :
14400          :   THE LINE CONSISTS OF A STRING OF O'S AND
14500          :   X'S.  FIRST, THE O'S ARE PRINTED OUT TO THE LAST
14600          :   COLUMN WITH A SPACE SEPARATING EACH.  THEN THE
14700          :   CARRIAGE IS SPACED TO THE FIRST BLANK SPACE, AN X
14800          :   IS PRINTED AND THEN RETURNED TO THE MARGIN.  THIS
14900          :   PROCESS IS CONTINUE UNTIL ALL SPACES BETWEEN
15000          :   THE ZEROES HAVE BEEN FILLED.
15100          :
15200          :XXXXXXXXXX
15300          :
15400 010164 000003 PT3:      3          ;TEST NUMBER
15500 010166 010304 PT4          ;NEXT TEST
15600 010170 104016 PRTHDR      ;TYPE HEADER
15700 010172 005067 170476 1$:      CLR      SPCNT      ;CLEAR SPACE COUNTER
15800 010176 016701 170450          ;POSITION COUNTER TO R1
15900 010202 012700 000117 2$:      MOV      #117,R0      ;'O' TO R0
16000 010206 104015          ;PRINT THE 'O'
16100 010210 005301          ;DECREMENT POSITION COUNTER
16200 010212 001404          ;BRANCH IF 0
16300 010214 004767 177340 JSR      PC,SPC      ;SEND SPACE
16400 010220 005301          ;DECREMENT POSITION COUNTER
16500 010222 001367          ;BRANCH IF NOT ZERO
16600 010224 104022 3$:      CR          ;SEND A CR
16700 010226 012767 000001 170440 MOV      #1,SPCNT      ;SPACE, COUNTER SET TO 1
16800 010234 016701 170434 4$:      MOV      SPCNT,R1      ;NO. OF SPACES TO R1
16900 010240 004767 177314 5$:      JSR      PC,SPC      ;SEND SPACE
17000 010244 005301          ;DECREMENT SPACE COUNTER
17100 010246 001374          ;BRANCH IF NOT ZERO
17200 010250 012700 000130 MOV      #130,R0      ;'X' INTO R0
17300 010254 104015          ;PRINT 'X'
17400 010256 104022          ;PRINT CR
17500 010260 062767 000002 170406 ADD      #2,SPCNT      ;INCREMENT SPACE COUNT BY 2
17600 010266 026767 170402 170356 CMP      SPCNT,WIDTH  ;COMPARE POSITION COUNTER WITH COLM. COUNT
17700 010274 103757          ;BRANCH IF LOWER
17800 010276 104014          ;SEND LF
17900 010300 104005          ;CHAIN TO NEXT TEST
18000 010302 000733 BR       1$          ;REPEAT TEST
    
```

```

18200          :XXXXXXXXXX
18300
18400          :PT4 -- MULTIPLE LINE FEED TEST -- 63 LINE FEEDS ARE
18500          :SENT WITH A REFERENCE LINE AT THE START AND END.
18600          :A NUMBER IS PRINTED WHICH INDICATES THE NUMBER OF LINE
18700          :FEEDS THAT WILL BE ISSUED BEFORE THE NEXT
18800          :NUMBER OR REFERENCE LINE IS PRINTED.
18900          :XXXXXXXXXX
19000
19100
19200 010304 000004 PT4: 4 ;TEST NUMBER
19300 010306 010462 PT5 ;NEXT TEST
19400 010310 104016 PRTHDR ;TYPE HEADER
19500 010312 012767 000001 170366 1$: MOV #1,LFCNT ;LINE FEED COUNT TO 1
19600 010320 016701 170326 MOV WIDTH,R1 ;COLUMN COUNT TO R1
19700 010324 012702 010444 MOV #LINE3,R2 ;ADDR OF NUMBER FIELD TO R2
19800 010330 004767 000060 JSR PC,REF ;PRINT REFERENCE LINE
19900 010334 016701 170346 2$: MOV LFCNT,R1 ;LINE FEED COUNT TO R1
20000 010340 104014 3$: LF ;SEND LF
20100 010342 005301 DEC R1 ;DECREMENT COUNTER
20200 010344 001375 BNE 3$ ;BRANCH IF NOT YET 0
20300 010346 006367 170334 ASL LFCNT ;DOUBLE LINE FEED COUNT
20400 010352 022767 000100 170326 CMP #BIT6,LFCNT ;TEST IF COUNT IS 32
20500 010360 001406 BEQ 4$ ;BRANCH IF =32, END
20600 010362 112200 MOVB (R2)+,R0 ;NUMBER TO R0
20700 010364 104015 PRINTC ;PRINT IT
20800 010366 112200 MOVB (R2)+,R0 ;NUMBER TO R0
20900 010370 104015 PRINTC ;PRINT IT
21000 010372 104022 CR ;PRINT CR
21100 010374 000757 BR 2$ ;DRIVE THE LINEFEEDS
21200 010376 016701 170250 4$: MOV WIDTH,R1 ;COLUMN COUNT TO R1
21300 010402 004767 000006 JSR PC,REF ;SEND END REFERENCE LINE
21400 010406 104014 LF ;ADVANCE PAPER
21500 010410 104005 CHAIN
21600 010412 000737 BR 1$ ;REPEAT TEST
21700 010414 112200 REF: MOVB (R2)+,R0 ;NUMBER TO R0
21800 010416 104015 PRINTC ;PRINT IT
21900 010420 112200 MOVB (R2)+,R0 ;NUMBER TO R0
22000 010422 104015 PRINTC ;PRINT IT
22100 010424 005741 TST -(R1) ;DECREASE COUNTER BY 2
22200 010426 012700 000137 1$: MOV #137,R0 ;DASH (-) TO R0
22300 010432 104015 PRINTC ;PRINT IT
22400 010434 005301 DEC R1 ;DECREMENT COLUMN COUNTER
22500 010436 001375 BNE 1$ ;BRANCH IF NO ZERO
22600 010440 104022 CR ;PRINT CR
22700 010442 000207 RTS PC ;RETURN
22800
22900 010444 060 061 060 LINE3: .ASCII /01020408163200/
      010447 062 060 064
      010452 060 070 061
      010455 066 063 062
      010460 060 060
    
```

```

23100          ;XXXXXXXXXX
23200          ;PT5-- SINGLE LINE FEED TEST -- TESTS THE LINE FEED
23300          ;          CAPABILITY FROM ALL COLUMNS.
23400          ;XXXXXXXXXX
23500
23600 010462 000005      PT5:      5          ;TEST NUMBER
23700 010464 010666      PT6          ;NEXT TEST
23800 010466 104016      PRTHDR      ;TYPE HEADER
23900 010470 016701 170156 1$:      MOV      WIDTH,R1      ;COLUMN COUNT TO R1
24000 010474 005741      TST      -(R1)      ;DECREASE BY 2
24100 010476 012700 000060      MOV      #60,R0      ;'0' TO R0
24200 010502 104015      2$:      PRINTC      ;SEND 0
24300 010504 005301      DEC      R1          ;DECREMENT COLUMN COUNTER
24400 010506 001375      BNE      2$          ;BRANCH IF NOT ZERO
24500 010510 012700 000062      MOV      #62,R0      ;SEND A 2
24600 010514 104015      PRINTC
24700 010516 104015      PRINTC
24800 010520 026727 170126 000204  CMP      WIDTH,#132.  ;SEND A SECOND TWO
24900 010526 001404      BEQ      3$          ;COMPARE COLUMN COUNT
25000 010530 012700 003410      MOV      #3410,R0    ;BRANCH IF EQ 132
25100 010534 104010      DELAY
25200 010536 000407      BR      5$          ;DELAY 1.8 SEC
25300 010540 012700 000063      3$:      MOV      #63,R0      ;3'S TO R0
25400 010544 012701 000100      MOV      #100,R1     ;64 TO COUNTER
25500 010550 104015      4$:      PRINTC      ;SEND CHARACTER
25600 010552 005301      DEC      R1          ;DECREMENT COUNT
25700 010554 001375      BNE      4$          ;BRANCH IF NOT ZERO
25800 010556 104012      5$:      CRLF          ;SEND A CR,LF
25900 010560 016701 170066      MOV      WIDTH,R1     ;NO. COLUMNS TO R1
26000 010564 012700 000134      6$:      MOV      #134,R0    ;BACKSLASH TO R0
26100 010570 104015      PRINTC      ;SEND IT
26200 010572 104014      LF          ;PRINT LF
26300 010574 005301      DEC      R1          ;DECREMENT COUNTER
26400 010576 001372      BNE      6$          ;BRANCH IF NOT ZERO.
26500 010600 104022      CR          ;SEND CR
26600 010602 004767 000022      JSR      PC,PT5AL    ;SEND REF LINE #1
26700 010606 104012      CRLF          ;SEND A CR,LF
26800 010610 012700 001750      MOV      #1750,R0    ;DELAY 1 SEC
26900 010614 104010      DELAY
27000 010616 004767 000006      JSR      PC,PT5AL    ;SEND A SECOND REF. LIA
27100 010622 104012      CRLF          ;SEND A CR,LF
27200 010624 104005      CHAIN      ;CHAIN TO NEXT TEST
27300 010626 000720      BR      1$          ;REPEAT TEST
27400 010630 016701 170016      PT5AL:  MOV      WIDTH,R1      ;COLUMN COUNT TO R1
27500 010634 012700 000061      MOV      #61,R0      ;'1' TO R0
27600 010640 104015      1$:      PRINTC      ;PRINT R0
27700 010642 005301      DEC      R1          ;DECREMENT COUNTER
27800 010644 001407      BEQ      2$          ;BRANCH IF=0
27900 010646 005200      INC      R0          ;INCREMENT CHARACTER
28000 010650 020027 000071      CMP      R0,#71      ;COMP CHAR TO '9'
28100 010654 101771      BLOS     1$          ;BRANCH IF LOWER OR SAME
28200 010656 012700 000060      MOV      #60,R0      ;RESET CHAR TO '0'
28300 010662 000766      BR      1$          ;CONTINUE
28400 010664 000207      2$:      RTS      PC      ;FINISHED, RETURN TO CALLER
    
```

```

28600          ;XXXXXXXXXX
28700          ;PT6-- BACKSPACE TEST -- A REFERENCE LINE SUCH AS IN
28800          ;          TEST PT5 IS PRINTED. THE SECOND LINE CONSISTS
28900          ;          OF PRINTING A BACKSLASH, BACKSPACE AND FORWARD
29000          ;          SLASH COMBINATION OUT TO THE GIVEN COLUMN WIDTH.
29100          ;          THIS LINE IS THEN FOLLOWED BY THE SAME TWO REFERENCE
29200          ;          LINES AS PRINTED IN TEST PT5.
29300          ;XXXXXXXXXX
29400
29500 010666 000006 PT6: 6          ;TEST NUMBER
29600 010670 011054 PT7          ;NEXT TEST
29700 010672 104016 PRTHDR        ;PRINT HEADER
29800 010674 104007 TYPEM          ;PRINT COLUMN # MMSG
29900 010676 014127 HDRO
30000 010700 016701 167746 1$: MOV WIDTH,R1      ;COLUMN COUNT TO R1
30100 010704 005741 TST -(R1)          ;DECREMENT BY 2
30200 010706 012700 000060 MOV #60,R0      ;'0' TO R0
30300 010712 104015 2$: PRINTC          ;SEND 0
30400 010714 005301 DEC R1          ;DECREMENT COLUMN COUNTER
30500 010716 001375 BNE 2$          ;BRANCH IF NOT ZERO
30600 010720 012700 000062 MOV #62,R0      ;'2' TO R0
30700 010724 104015 PRINTC          ;SEND A '2'
30800 010726 104015 PRINTC          ;SEND A SECOND '2'
30900 010730 026727 167716 000204 CMP WIDTH,#132. ;COMPARE COLUMN COUNT
31000 010736 001404 BEQ 3$          ;
31100 010740 012700 003410 MOV #3410,R0    ;DELAY 1.8 SEC
31200 010744 104010 DELAY
31300 010746 000407 BR 5$
31400 010750 012700 000063 3$: MOV #63,R0      ;3'S TO R0
31500 010754 012701 000100 MOV #100,R1    ;64 TO CHAR COUNT
31600 010760 104015 4$: PRINTC          ;SEND CHAR
31700 010762 005301 DEC R1          ;DECREMENT CHAR COUNT
31800 010764 001375 BNE 4$          ;CONTINUE IF NOT DONE
31900 010766 104012 5$: CRLF          ;SEND A CR,LF
32000 010770 016701 167656 MOV WIDTH,R1    ;COLUMN COUNT TO R1
32100 010774 012700 000134 6$: MOV #134,R0    ;BACKSLASH TO R0
32200 011000 104015 PRINTC          ;SEND IT
32300 011002 012700 000010 MOV #10,R0      ;BACKSPACE TO R0
32400 011006 104015 PRINTC          ;SEND IT
32500 011010 012700 000057 MOV #57,R0      ;FORWARD SLASH TO R0
32600 011014 104015 PRINTC          ;SEND IT
32700 011016 005301 DEC R1          ;END OF PAPER
32800 011020 001365 BNE 6$          ;BRANCH IF NO
32900 011022 104014 LF          ;SEND LF
33000 011024 104022 CR          ;SEND CR
33100 011026 004767 177576 JSR PC,PT5AL    ;SEND REF LINE #1
33200 011032 104012 CRLF          ;SEND A CR,LF
33300 011034 012700 001750 MOV #1750,R0    ;DELAY 1 SEC
33400 011040 104010 DELAY
33500 011042 004767 177562 JSR PC,PT5AL    ;SEND SECOND REF LINE
33600 011046 104012 CRLF          ;SEND A CR,LF
33700 011050 104005 CHAIN          ;CHAIN TO NEXT TEST
33800 011052 000712 BR 1$          ;REPEAT TEST
    
```

```

34000          ;XXXXXXXXXX
34100          ;
34200          ;PT7-- OVERPRINT TEST-- A ROW OF ALTERNATING M'S AND
34300          ;          SPACES ARE PRINTED, OUT TO THE LAST COLUMN AND OVERPRINTED TWICE.
34400          ;          A SECOND LINE OF ALTERNATING SPACES AND 'a'S' IS THEN
34500          ;          SENT 3 TIMES AS THE FIRST LINE. THIS IS FOLLOWED
34600          ;          BY A THIRD AND FINAL LINE OF ALTERNATING '8'
34700          ;          AND SPACES.
34800          ;
34900          ;XXXXXXXXXX
35000          ;
35100 011054 000007 PT7: 7          ;TEST NUMBER
35200 011056 011266 PT10        ;NEXT TEST
35300 011060 104016 PRTHDR      ;PRINT MESSAGE
35400 011062 012703 000002 1$:  MOV #2,R3    ;2 COUNT TO R3
35500 011066 016701 167560 2$:  MOV WIDTH,R1  ;NO. OF COLUMNS TO R1
35600 011072 012700 000115 3$:  MOV #115,R0 ;'M' TO R0
35700 011076 104015 PRINTC      ;SEND IT
35800 011100 005301 DEC R1       ;END OF LINE
35900 011102 001404 BEQ 4$      ;BRANCH IF YES
36000 011104 004767 176450 JSR PC,SPC ;SEND SPACE
36100 011110 005301 DEC R1       ;END OF LINE?
36200 011112 001367 BNE 3$      ;BRANCH IF NO
36300 011114 022703 000002 4$:  CMP #2,R3    ;TEST R3
36400 011120 001003 BNE 6$      ;BRANCH IF NOT FIRST TIME
36500 011122 104022 5$:  CR          ;SEND CR
36600 011124 005303 DEC R3       ;DECREASE LINE COUNTER
36700 011126 000757 BR 2$       ;REPEAT LINE
36800 011130 005703 6$:  TST R3       ;THIRD TIME?
36900 011132 001373 BNE 5$      ;BRANCH IF NOT
37000 011134 104012 CRLF      ;NEXT LINE
37100 011136 005723 TST (R3)+  ;REPEAT COUNTER TO R3
37200 011140 016701 167506 7$:  MOV WIDTH,R1  ;COLUMN COUNT TO R1
37300 011144 004767 176410 8$:  JSR PC,SPC ;SEND SPACE
37400 011150 005301 DEC R1       ;DECREASE COLUMN COUNT
37500 011152 001405 BEQ 9$      ;BRANCH IF 0, END OF LINE
37600 011154 012700 000100 MOV #100,R0 ;'a' TO R0
37700 011160 104015 PRINTC      ;SEND IT
37800 011162 005301 DEC R1       ;DECREASE COLUMN COUNT
37900 011164 001367 BNE 8$      ;BRANCH IF NOT 0 (NOT END)
38000 011166 022703 000002 9$:  CMP #2,R3    ;END OF LINE, FIRST TIME?
38100 011172 001003 BNE 11$     ;BRANCH IF NOT
38200 011174 104022 10$: CR       ;SEND CR
38300 011176 005303 DEC R3       ;DECREASE LINE COUNTER
38400 011200 000757 BR 7$       ;REPEAT LINE
38500 011202 005703 11$: TST R3     ;TEST IF THIRD REPEAT
38600 011204 001373 BNE 10$     ;BRANCH IF NOT
38700 011206 104012 CRLF      ;DO NEXT LINE
38800 011210 005723 TST (R3)+  ;LINE REPEAT COUNTER TO R3
38900 011212 016701 167434 12$: MOV WIDTH,R1  ;COLUMN COUNT TO R1
39000 011216 012700 000046 13$: MOV #46,R0  ;'8' TO R0
39100 011222 104015 PRINTC      ;SEND IT
39200 011224 005301 DEC R1       ;DECREASE COLUMN COUNT
39300          ;
39400 011226 001404 BEQ 14$     ;BRANCH IF END
39500 011230 004767 176324 JSR PC,SPC ;SEND SPACE
39600 011234 005301 DEC R1       ;DECREASE COLUMN COUNT
    
```

39700	011236	001367			BNE	13\$:BRANCH IF NOT END
39800	011240	022703	000002	14\$:	CMP	#2,R3	:TEST IF FIRST TIME
39900	011244	001003			BNE	16\$:BRANCH IF =2, FIRST TIME
40000	011246	104022		15\$:	CR		:SENT CR
40100	011250	005303			DEC	R3	:DECREASE REPEAT COUNTER
40200	011252	000757			BR	12\$:PRINT LINE AGAIN
40300	011254	005703		16\$:	TST	R3	:TEST IF END, R3=0
40400	011256	001373			BNE	15\$:BRANCH IF NOT END
40500	011260	104012			CRLF		:SEND CR,LF
40600	011262	104005			CHAIN		:CHAIN TO NEXT TEST
40700	011264	000676			BR	1\$:REPEAT TEST


```

40900          :XXXXXXXXXX
41000          :
41100          :PT10-- PRINTING FREQUENCY TEST-- 120 H'S ARE PRINTED ON 4 LINES
41200          :30 PER LINE. THE TEST IS SUCH THAT BETWEEN THE FIRST AND SECOND
41300          :'H' A 30 MSEC DELAY IS INTRODUCED. THIS DELAY IS THEN INCREASED
41400          :BETWEEN CHARACTERS OUT TO 60 CHARACTERS IN AN EXPONENTIAL
41500          :MANNER. THE DELAY IS THEN DECREASED IN THE SAME MANNER OUT TO THE
41600          :120TH CHARACTER. THIS DELAY IS CALCULATED AS FOLLOWS:
41700          :
41800          :NEW DELAY = OLD DELAY [+ OR -] (OLD DELAY/16 + OLD DELAY/128 )
41900          :
42000          :XXXXXXXXXX
42100          :
42200 011266 000010 PT10: 10          ;TEST NUMBER
42300 011270 011424          PT11          ;NEXT TEST
42400 011272 104016          PRTHDR          ;TYPE MESSAGE
42500 011274 012701 000036 1$:  MOV #36,R1          ;SET R1=30
42600 011300 012702 000170          MOV #120.,R2          ;SET CHAR COUNT = 120
42700 011304 012767 000036 000010  MOV #30.,3$+2          ;SET UP DELAY VALUE
42800 011312 012700 000110 2$:  MOV #110,R0          ;'H' TO R0
42900 011316 104015          PRINTC          ;SEND IT
43000 011320 012700 000036 3$:  MOV #30.,R0
43100 011324 104010          DELAY          ;DELAY
43200 011326 005301          DEC R1          ;DEC. COUNT OF CHARS PER LINE
43300 011330 001426          BEQ 6$          ;BRANCH IF 0, END OF LINE
43400 011332 005302 4$:  DEC R2          ;DECREMENT CHAR COUNTER
43500 011334 001430          BEQ 7$          ;BRANCH IF END
43600 011336 016704 177760  MOV 3$+2,R4          ;GET OLD DELAY
43700 011342 006204          ASR R4          ;CAL 1/16 OF OLD DELAY
43800 011344 006204          ASR R4
43900 011346 006204          ASR R4
44000 011350 006204          ASR R4
44100 011352 010405          MOV R4,R5          ;SAVE 1/16 IN R5
44200 011354 006204          ASR R4          ;CAL 1/128 OF OLD DELAY
44300 011356 006204          ASR R4
44400 011360 006204          ASR R4
44500 011362 060405          ADD R4,R5          ;1/16 +1/128 TO R5
44600 011364 022702 000074          CMP #60.,R2          ;TEST WHICH HALF OF THE 120 CHARS.
44700 011370 003403          BLE 5$          ;BRANCH IF LT OR EQ 60
44800 011372 160567 177724          SUB R5,3$+2          ;GT 51, DECREASE DELAY BY 34 MEC.
44900 011376 000745          BR 2$          ;GO PRINT AGAIN
45000 011400 060567 177716 5$:  ADD R5,3$+2          ;LT HALF WAY, ADD DELAY OF 34 MEC.
45100 011404 000742          BR 2$          ;GO PRINT AGAIN
45200 011406 104012 6$:  CRLF          ;SEND CRLF
45300 011410 012701 000036          MOV #36,R1          ;SET R1=30
45400 011414 000746          BR 4$
45500 011416 104012 7$:  CRLF          ;SEND CR,LF
45600 011420 104005          CHAIN          ;CHAIN TO NEXT TEST
45700 011422 000724          BR 1$          ;REPEAT TEST
    
```

```

45900          :XXXXXXXXXX
46000
46100          :PT11-- RIBBON FEED TEST-- THIS TEST PRINTS A SINGLE COLUMN OF X'S
46200          :          (24 LINES) DOWN THE LEFT MARGIN OF THE PAGE.
46300          :          VISUALLY CHECK THE RIBBON FEED MECHANISM FOR PROPER OPERATION.
46400
46500          :XXXXXXXXXX
    
```

```

46800 011424 000011 PT11: 11          :TEST NUMBER
46900 011426 011456          :NEXT TEST
47000 011430 104016          :TYPE MESSAGE
47100 011432 012701 000030 1$:  MOV #30,R1      :SET R1=24(10), LINE COUNT
47200 011436 012700 000130 2$:  MOV #130,R0     :SET CHAR = X
47300 011442 104015          :PRINT X
47400 011444 104012          :SEND CR-LF
47500 011446 005301          :DECREMENT LINE COUNT
47600 011450 001372          :CONTINUE IF NOT DONE TEST
47700 011452 104005          :CHAIN TO NEXT TEST
47800 011454 000766          :REPEAT TEST
47900
48000
48100
48200          :XXXXXXXXXX
    
```

```

48300
48400          :PT12-- PRINTER BELL TEST-- THE LAST TEST IN THE
48500          :          PRINTER TEST SEQUENCE. THIS TEST OUTPUTS
48600          :          EIGHT BELL SIGNALS TO THE PRINTER
48700
48800
48900          :XXXXXXXXXX
    
```

```

49000
49100 011456 000012 PT12: 12          :THIS TEST
49200 011460 007372          :NEXT TEST
49300 011462 104016          :TYPE HEADER
49400 011464 012701 000010 PT12A: MOV #10,R1   :COUNTER TO R1
49500 011470 012700 000007  MOV #7,R0     :BELL TO R0
49600 011474 104015          :SEND IT
49700 011476 005301          :DECREMENT COUNT
49800 011500 001375          :BRANCH IF NOT ZERO
49900 011502 104014          :
50000 011504 012700 003720 1$:  MOV #3720,R0   :DELAY 2 SEC BEFORE RESTARTING
50100 011510 104010          :
50200 011512 013700 000042  MOV @#42,R0   :CHECK IF UNDER ACT11 OR XXDP
50300 011516 001405          :CONTINUE TEST SEQUENCE
50400 011520 000240          :A RESET WAS FORMERLY HERE
50500 011522 004710          LOGICAL:JSR PC,(R0)
50600 011524 000240          :
50700 011526 000240          :
50800 011530 000240          :
50900 011532 104005          HERE:  CHAIN    :CHAIN TO NEXT TEST
51000 011534 000753          BR      PT12A   :REPEAT TEST
    
```

```

51200          :XXXXXXXXXX
51300          :
51400          :PT17-- LIFE TEST
51500          :      THIS TEST PRINTS 2 FULL LINES OF EACH PRINTABLE
51600          :      CHARACTER AND OVERPRINTS THE SECOND LINE 4 TIMES.
51700          :      THIS TEST IS CONTINUOUS RUNNING ONCE INITIATED,
51800          :      LOOPING AUTOMATICALLY ON ITSELF.
51900          :      END OF PASS COUNT IS CLEARED WHENEVER TEST IS RESTARTED
52000          :
52100          :XXXXXXXXXX
52200          :
52300 011536 000017 PT17B: 17          ;TEST NUMBER
52400 011540 011536 PT17B          ;NEXT TEST
52500 011542 000167 000030 JMP          PT17D          ;CONTINUE
52600 011546 000017 PT17: 17          ;TEST NUMBER
52700 011550 011536 PT17B          ;NEXT TEST
52800 011552 005067 000336 CLR          PASCNT          ;CLEAR PASS COUNT
52900 011556 016704 167070 MOV          WIDTH,R4        ;INITIALIZE R4
53000 011562 012767 000001 000322 MOV          #1,DIRTN        ;AND DIRECTION OF PRECESS
53100 011570 104016 PRTHDR          ;PRINT COLUMN # MESG
53200 011572 104007 TYPEN          ;
53300 011574 014127 HDRO          ;
53400 011576 012703 000041 PT17D: MOV          #41,R3          ;SET START CHAR
53500 011602 005267 000306 INC          PASCNT          ;
53600 011606 026727 000302 000031 CMP          PASCNT,#31        ;DO 31 TIMES
53700 011614 001003 BNE          20$          ;BRANCH IF NOT DONE
53800 011616 012767 000001 000270 MOV          #1,PASCNT        ;START OVER
53900 011624 012700 014042 20$: MOV          #PASMES,R0        ;SET MMSG ADDR
54000 011630 016701 000260 MOV          PASCNT,R1        ;# TO CONVERT
54100 011634 012702 000002 MOV          #2,R2          ;# DIGITS
54200 011640 104023 BTOASC          ;CONVERT PASCNT TO ASCII
54300 011642 016701 167004 1$: MOV          WIDTH,R1          ;SET COLUMN COUNT
54400 011646 010300 2$: MOV          R3,R0          ;GET CHARACTER
54500 011650 004767 000110 JSR          PC,CKPOS        ;TIME TO INSERT PASS # ?
54600 011654 104015 PRINTC          ;SEND CHAR
54700 011656 005301 DEC          R1          ;DECREMENT COUNT
54800 011660 003372 BGT          2$          ;BRANCH IF NOT DONE
54900 011662 004767 000144 JSR          PC,ADJR4        ;ADJUST R4 POINTER
55000 011666 104012 CRLF          ;
55100 011670 012702 000005 3$: MOV          #5,R2          ;SET OVERPRINT COUNT
55200 011674 016701 166752 4$: MOV          WIDTH,R1          ;SET COLUMN COUNT
55300 011700 010300 MOV          R3,R0          ;GET CHARACTER
55400 011702 004767 000056 JSR          PC,CKPOS        ;TIME TO INSERT PASS # ?
55500 011706 104015 PRINTC          ;SEND CHAR
55600 011710 005301 DEC          R1          ;DECREMENT COUNT
55700 011712 003372 BGT          4$          ;BRANCH IF NOT DONE
55800 011714 104022 CR          ;SEND CR
55900 011716 005302 DEC          R2          ;DONE OVERPRINTS ?
56000 011720 001365 BNE          3$          ;NO. CONTINUE
56100 011722 004767 000104 JSR          PC,ADJR4        ;ADJUST R4 POINTER
56200 011726 104014 LF          ;SEND LF
56300 011730 005203 INC          R3          ;SET NEXT CHAR
56400 011732 022703 000177 CMP          #177,R3        ;DONE CHAR SET ?
56500 011736 001341 BNE          1$          ;NO. CONTINUE
56600 011740 004767 000066 JSR          PC,ADJR4        ;OFFSET POINTER 3 PLACES
56700 011744 004767 000062 JSR          PC,ADJR4        ;TO RETAIN VISUAL ALIGNMENT
56800 011750 004767 000056 JSR          PC,ADJR4        ;THROUGH END OF PASS
    
```

56900 011754 104007
57000 011756 014023
57100 011760 104005
57200 011762 000705
57300

TYPEM
ENDPAS
CHAIN
BR PT17D

;TYPE END OF PASS MESG
;REPEAT TEST

```

57500
57600 011764 020401          CKPOS:  CMP    R4,R1          ;IS IT TIME TO INSERT PASS # ?
57700 011766 001020          BNE    1$          ;BRANCH IF NO
57800 011770 012700 000040    MOV    #40,R0      ;PRINT A SPACE
57900 011774 104015          PRINTC
58000 011776 116700 002040    MOVB   PASMES,R0   ;PRINT MSG OF PASS COUNT
58100 012002 104015          PRINTC
58200 012004 116700 002033    MOVB   PASMES+1,R0
58300 012010 104015          PRINTC
58400 012012 012700 000040    MOV    #40,R0      ;PRINT A SPACE
58500 012016 104015          PRINTC
58600 012020 162701 000003    SUB    #3,R1        ;ADJUST R1 3 POSITIONS
58700 012024 062716 000002    ADD    #2,(SP)     ;ADJUST RETURN PC OVER PRINTC
58800 012030 000207          1$:    RTS    PC
58900
59000 012032 005767 000054    ADJR4: TST   DIRTN          ;TEST DIRECTION OF PRECESS
59100 012036 001013          BNE    1$          ;BR IF LEFT
59200 012040 005204          INC    R4           ;INCREASE POSITION CNTR
59300 012042 020467 166604    CMP    R4,WIDTH     ;IS R4 > WIDTH ?
59400 012046 101420          BLOS   3$          ;BR IF NOT GREATER
59500 012050 016704 166576    MOV    WIDTH,R4     ;CHANGE DIRECTION
59600 012054 005304          DEC    R4           ;      TO
59700 012056 012767 000001 000026  MOV    #1,DIRTN     ;      LEFT.
59800 012064 000411          BR    3$
59900 012066 005304          1$:    DEC    R4           ;DECREASE POSITION CNTR
60000 012070 020427 000004    CMP    R4,#4        ;LESS THAN 4 ?
60100 012074 002401          BLT    2$          ;BR IF YES
60200 012076 000404          BR    3$          ;ELSE EXIT
60300 012100 012704 000005    2$:    MOV    #5,R4        ;SET R4 TO POS 5
60400 012104 005067 000002    CLR    DIRTN       ;CHANGE DIRECTION TO RIGHT
60500 012110 000207          3$:    RTS    PC        ;EXIT
60600
60700 012112 000000          DIRTN: .WORD 0      ;DIRECTION OF PRECESS (0=LEFT)
60800
60900 012114 000000          PASCNT: .WORD 0
    
```

61100

100
 200
 300
 400
 500
 600
 700
 800
 900
 1000
 1100
 1200
 1300 012116 000020
 1400 012120 012166
 1500 012122 104016
 1600 012124 104020
 1700 012126 012700 000036
 1800 012132 104010
 1900 012134 022767 000177 166536
 2000 012142 001405
 2100 012144 104017
 2200 012146 117777 166442 166444
 2300 012154 000763
 2400 012156 104007
 2500 012160 014272
 2600 012162 104005
 2700 012164 000757
 2800
 2900
 3000
 3100
 3200
 3300
 3400
 3500
 3600
 3700
 3800
 3900 012166 000021
 4000 012170 012224
 4100 012172 104016
 4200 012174 012767 000060 166460
 4300 012202 016702 166444
 4400 012206 016700 166450
 4500 012212 104015
 4600 012214 005302
 4700 012216 003373
 4800 012220 104012
 4900 012222 000767

```

.SBTTL LA36 ECHO TESTS
:XXXXXXXXXX
:EO20-- CHARACTER ECHO TEST-- ALL PRINTABLE AND
:NON-PRINTING CHARACTERS TYPED ON THE KEYBOARD
:ARE USED TO DRIVE THE PRINTER, ONE CHARACTER AT
:A TIME. A 'RUBOUT' WILL CAUSE THE TEST TO BE
:TERMINATED.
:XXXXXXXXXX
EO20: 20 ;TEST NUMBER
      EO21 ;NEXT TEST
      PRTHDR ;TYPE HEADER
1$:  READ ;GO WAIT FOR KEYBOARD INPUT
     MOV #30.,R0 ;DELAY FOR HALF DUPLEX
     DELAY
     CMP #177,TEMPCH ;CHECK IF RUBOUT
     BEQ 2$ ;BRANCH IF YES
     PRNT ;NO, CHECK PRINTER READY
     MOVB @TKB,@TPB ;READY, ECHO CHARACTER
     BR 1$
2$:  TYPEM ;PRINT TERMINATION MESSAGE
     ECOEND
     CHAIN ;CHAIN TO NEXT TEST
     BR 1$ ;REPEAT TEST
:XXXXXXXXXX
:EO21-- LINE ECHO TEST, FAST RATE-- THIS TEST WILL
:CAUSE THE CONTINUAL PRINTING OF 'O' AT THE MAXIMUM
:RATE UNTIL EITHER ANOTHER CHARACTER IS SELECTED
:BY PRESSING A KEY ON THE KEYBOARD OR TERMINATION BY THE
:RUBOUT.
:XXXXXXXXXX
EO21: 21 ;TEST NUMBER
      EO22 ;NEXT TEST
      PRTHDR ;TYPE HEADER
EO21A: MOV #60,REPT ;CHARACTER TO BE REPEATED (O)
1$:  MOV WIDTH,R2 ;SET COLUMN COUNT
2$:  MOV REPT,R0 ;GET CHAR
     PRINTC ;PRINT CHAR
     DEC R2 ;DEC COLUMN COUNT
     BGT 2$ ;FINISH LINE
     CRLF ;SEND A CR AND LF
     BR 1$

```

```
5100  
5200  
5300  
5400  
5500  
5600  
5700  
5800  
5900 012224 000022  
6000 012226 012476  
6100 012230 104016  
6200 012232 012767 000060 166422 E022A: MOV #60,REPT  
6300 012240 016702 166406 1$: MOV WIDTH,R2  
6400 012244 016700 166412 2$: MOV REPT,R0  
6500 012250 104015 PRINTC  
6600 012252 005302 DEC R2  
6700 012254 001404 BEQ 3$  
6800 012256 012700 003410 MOV #3410,R0  
6900 012262 104010 DELAY  
7000 012264 000767 BR 2$  
7100 012266 104012 3$: CRLF  
7200 012270 000763 BR 1$
```

```
:XXXXXXXXXX  
:E022-- LINE ECHO TEST, SLOW RATE-- SAME AS E021 EXCEPT  
: THAT A DELAY IS INTRODUCED BETWEEN CHARACTERS  
: TO PRODUCE A LCV ACTION  
:XXXXXXXXXX  
E022: 22  
E023  
PRTHDR  
:TYPE HEADER  
:LOAD 0 AS INITIAL CHARACTER  
E022A: MOV #60,REPT  
1$: MOV WIDTH,R2  
2$: MOV REPT,R0  
:SET COLUMN COUNT  
:GET CHAR  
:PRINT CHAR  
DEC R2  
:DEC COLUMN COUNT  
BEQ 3$  
MOV #3410,R0  
:BRANCH IF DONE LINE  
DELAY  
:DELAY 1.8 SEC.  
BR 2$  
:OUTPUT NEW CHAR.  
3$: CRLF  
BR 1$  
:SEND A CR AND LF
```


7400					
7500					
7600					
7700					
7800					
7900					
8000	012272	116	125	114	MONIC: .ASCII /NUL /
	012275	040			
8100	012276	123	117	110	.ASCII /SOH /
	012301	040			
8200	012302	123	124	130	.ASCII /STX /
	012305	040			
8300	012306	105	124	130	.ASCII /ETX /
	012311	040			
8400	012312	105	117	124	.ASCII /EOT /
	012315	040			
8500	012316	105	116	121	.ASCII /ENQ /
	012321	040			
8600	012322	101	103	113	.ASCII /ACK /
	012325	040			
8700	012326	102	105	114	.ASCII /BEL /
	012331	040			
8800	012332	102	123	040	.ASCII /BS /
	012335	040			
8900	012336	110	124	040	.ASCII /HT /
	012341	040			
9000	012342	114	106	040	.ASCII /LF /
	012345	040			
9100	012346	126	124	040	.ASCII /VT /
	012351	040			
9200	012352	106	106	040	.ASCII /FF /
	012355	040			
9300	012356	103	122	040	.ASCII /CR /
	012361	040			
9400	012362	123	117	040	.ASCII /SO /
	012365	040			
9500	012366	123	111	040	.ASCII /SI /
	012371	040			
9600	012372	104	114	105	.ASCII /DLE /
	012375	040			
9700	012376	104	103	061	.ASCII /DC1 /
	012401	040			
9800	012402	104	103	062	.ASCII /DC2 /
	012405	040			
9900	012406	104	103	063	.ASCII /DC3 /
	012411	040			
10000	012412	104	103	064	.ASCII /DC4 /
	012415	040			
10100	012416	116	101	113	.ASCII /NAK /
	012421	040			
10200	012422	123	131	116	.ASCII /SYN /
	012425	040			
10300	012426	105	124	102	.ASCII /ETB /
	012431	040			
10400	012432	103	101	116	.ASCII /CAN /
	012435	040			
10500	012436	105	115	040	.ASCII /EM /

 :
 : THIS FOLLOWING TABLE IS USED BY TEST E023
 :
 :*****

10600	012441	040				
	012442	123	125	102	.ASCII	/SUB /
	012445	040				
10700	012446	105	123	103	.ASCII	/ESC /
	012451	040				
10800	012452	106	123	040	.ASCII	/FS /
	012455	040				
10900	012456	107	123	040	.ASCII	/GS /
	012461	040				
11000	012462	122	123	040	.ASCII	/RS /
	012465	040				
11100	012466	125	123	040	.ASCII	/US /
	012471	040				
11200	012472	123	120	040	.ASCII	/SP /
	012475	040				
11300						
11400					.EVEN	

```

11600 :XXXXXXXXXX
11700 :
11800 :E023-- CHARACTER CODE TEST-- ANY CHARACTER SELECTED
11900 :          WILL BE ECHOED ALONG WITH ITS OCTAL CODE.
12000 :          A MNEMONIC WILL BE PRINTED INSTEAD OF THE CHARACTER
12100 :          IF IT IS A NON-PRINTING CHARACTER.
12200 :          THE PARITY OF THE RECEIVED CODE WILL ALSO BE
12300 :          INDICATED AS EITHER EVEN OR ODD.
12400 :
12500 :XXXXXXXXXX
12600 :
12700 012476 000023 E023: 23 :TEST NUMBER
12800 012500 013020 E024 :NEXT TEST
12900 012502 104016 PRTHDR :TYPE HEADER
13000 012504 104020 1$: READ :GO WAIT FOR CHARACTER
13100 012506 012700 000036 MOV #30.,R0 :DELAY FOR HALF DUPLEX
13200 012512 104010 DELAY
13300 012514 026727 166160 000041 CMP TEMPCH,#41 :TEST IF CHAR IS PRINTABLE
13400 012522 103015 BHIS 3$ :BRANCH IF IT IS
13500 012524 004767 000130 JSR PC,STRLN :STORE CODE INTO MESSAGE
13600 012530 116700 166144 MOVB TEMPCH,R0 :GET CODE AGAIN
13700 012534 006300 ASL R0 :MULT BY 2
13800 012536 006300 ASL R0 :MULT BY 4
13900 012540 062700 012272 ADD #MONIC,R0 :ADD ADDR OF MNEMONIC TABLE
14000 012544 004767 000166 JSR PC,MOVNUM :MOV MNEMONIC TO MESSAGE
14100 012550 104000 2$: TYPE :TYPE CODE AND MNEMONIC
14200 012552 014321 E023M :ADDRESS OF MESSAGE
14300 012554 000753 BR 1$ :GO WAIT FOR NEXT CHARACTER
14400 012556 026727 166116 000177 3$: CMP TEMPCH,#177 :TEST IF CHAR IS A RUBOUT
14500 012564 001421 BEQ 4$ :BRANCH IF RUBOUT
14600 012566 012701 013010 MOV #MG24,R1
14700 012572 116721 166102 MOVB TEMPCH,(R1)+
14800 012576 112721 000040 MOVB #40,(R1)+
14900 012602 112721 000040 MOVB #40,(R1)+
15000 012606 112721 000040 MOVB #40,(R1)+
15100 012612 004767 000042 JSR PC,STRLN :STORE CODE INTO MESSAGE
15200 012616 012700 013010 MOV #MG24,R0 :ADDR OF CHAR INTO R0
15300 012622 004767 000110 JSR PC,MOVNUM :MOVE CHAR INTO MESSAGE
15400 012626 000750 BR 2$ :TYPE MESSAGE
15500 012630 004767 000024 4$: JSR PC,STRLN :RUBOUT, CONVERT AND STOR CODE
15600 012634 012700 013014 MOV #MG25,R0 :ADDR. OF DEL INTO R0
15700 012640 004767 000072 JSR PC,MOVNUM :MOVE DEL INTO MESSAGE
15800 012644 104000 TYPE :TYPE MESSAGE
15900 012646 014321 E023M :ADDR OF MESSAGE
16000 012650 104007 TYPEM
16100 012652 014272 ECOEND
16200 012654 104005 CHAIN
16300 012656 000712 BR 1$ :CHAIN TO NEXT TEST
16400 012660 012702 000003 STRLN: MOV #3,R2 :REPEAT TEST
16500 012664 012701 014323 MOV #LINE5,R1 :COUNT OF 3 TO R2
16600 012670 062701 000003 ADD #3,R1 :ADDR OF MESSG TO R1
16700 012674 016700 166004 1$: MOV PCHAR,R0 :POINT TO LAST SPACE IN MESSG
16800 012700 042700 177770 BIC #177770,R0 :MOVE OCTAL CODE TO R0
16900 012704 062700 000060 ADD #60,R0 :SAVE LS OCTAL CHAR
17000 012710 110041 MOVB R0,-(R1) :MAKE ASCII
17100 012712 005302 DEC R2 :MOVE INTO MESSG
17200 012714 001407 BEQ 2$ :DECREMENT CHAR COUNTER
:BRANCH IF 3 MOVED
    
```

17300	012716	006267	165762			ASR	PCHAR		:NOT THREE, SHIFT NEXT OCTAL
17400	012722	006267	165756			ASR	PCHAR		:CHARACTER TO THE RIGHT
17500	012726	006267	165752			ASR	PCHAR		:
17600	012732	000760				BR	1\$:CONVERT AND STORE NEXT CHAR
17700	012734	000207				RTS	PC		:RETURN TO CALLER
17800	012736	012701	014327			MOVNUM: MOV	#LINE5A,R1		:ADDR OF LINES IN R1
17900	012742	012702	000004			MOV	#4,R2		:COUNT OF 4 TO R2
18000	012746	112021				1\$: MOV	(R0)+,(R1)+		:MOV 4 CHARS TO MMSG AREA
18100	012750	005302				DEC	R2		:DECREMENT COUNTER
18200	012752	001375				BNE	1\$:BRANCH IF NOT ALL DONE
18300	012754	105767	165722			TSTB	PARITY		:TEST PARITY FLAG
18400	012760	001003				BNE	2\$:BRANCH IF ODD PARITY
18500	012762	012700	014371			MOV	#EVEN,R0		:SET ADDRESS FOR EVEN PARITY MMSG
18600	012766	000402				BR	3\$:CONTINUE
18700	012770	012700	014375			2\$: MOV	#ODD,R0		:SET ADDRESS FOR ODD PARITY MMSG
18800	012774	012702	000004			3\$: MOV	#4,R2		:COUNT OF 4 TO R2
18900	013000	112021				4\$: MOV	(R0)+,(R1)+		:MOVE 4 CHARS TO MMSG AREA
19000	013002	005302				DEC	R2		:DECREMENT COUNTER
19100	013004	001375				BNE	4\$:BRANCH IF NOT DONE
19200	013006	000207				RTS	PC		:RETURN
19300									
19400	013010	040	040	040	MG24:	.ASCII	/ /		:SAVE CHARACTER CODE
	013013	040							
19500									
19600						.EVEN			
19700									
19800	013014	104	105	114	MG25:	.ASCII	/DEL /		:MNEMONIC FOR RUBOUT
	013017	040							
19900									
20000						.EVEN			

```

20200      ;XXXXXXXXXX
20300      ;E024-- SELECTED PATTERN ECHO TEST-- SELECT 1 TO 256
20400      ;          CHARACTERS. EACH WILL BE ECHOED
20500      ;          AND STORED UNTIL THE CNTL/C IS SELECTED.
20600      ;          AT THAT TIME ALL CHARACTERS WILL BE PRINTED AS
20700      ;          A CONTINUOUS STRING UNTIL EITHER THE RUBOUT IS
20800      ;          SELECTED TO TERMINATE OR THE CNTL/C IS SELECTED
20900      ;          AGAIN. A TERMINATING CNTL/C FOLLOWED BY ANOTHER
21000      ;          CNTL/C WILL ALWAYS CAUSE THE LAST INPUTTED STRING TO
21100      ;          BE PRINTED. A TERMINATING CNTL/C FOLLOWED BY A CHARACTER OTHER THAN A
21200      ;          RUBOUT WILL CAUSE A NEW STRING TO BE INPUTTED.
21300      ;XXXXXXXXXX
21400
21500 013020 000024      E024: 24          ;TEST NUMBER
21600 013022 013566      E025          ;NEXT TEST
21700 013024 104016      PRTHDR      ;TYPE TEST HEADER
21800 013026 005001      E024B: CLR R1          ;CLEAR CHARACTER COUNT
21900 013030 012702 013164  MOV #BUFR,R2      ;ADDRESS OF BUFFER TO R2
22000 013034 104020      1$: READ          ;WAIT FOR INPUT
22100 013036 012700 000036  MOV #30.,R0       ;DELAY FOR HALF DUPLEX
22200 013042 104010      DELAY
22300 013044 022767 000177 165626  CMP #177,TEMPCH   ;TEST IF RUBOUT
22400 013052 001440      BEQ TERM          ;BRANCH IF RUBOUT
22500 013054 022767 000003 165616  CMP #3,TEMPCH     ;TEST IF CNTL-C
22600 013062 001413      BEQ OUTPUT        ;BRANCH IF CNTL-C
22700 013064 020127 000400      CMP R1,#256.      ;YES, CHECK IF CHAR CNT IS EQ, GT 256
22800 013070 103361      BHIS 1$          ;BRANCH IF YES, IGNORE CHAR
22900 013072 116722 165602      MOVB TEMPCH,(R2)+  ;STORE CHAR INTO BUFFER
23000 013076 005201      INC R1          ;INCREMENT CHARACTER COUNT
23100 013100 104017      PRNT          ;CHECK IF PRINTER READY
23200 013102 116777 165572 165510  MOVB TEMPCH,@TPB  ;ECHO CHAR
23300 013110 000751      BR 1$          ;GO WAIT FOR NEXT CHAR
23400
23500      ;SECTION TO OUTPUT CONTINUOUS STRING
23600
23700 013112 020227 013164      OUTPUT: CMP R2,#BUFR   ;CHECK IF POINTER IS AT START OF TABLE
23800 013116 001403      BEQ 1$          ;YES, BRANCH
23900 013120 116722 165554      MOVB TEMPCH,(R2)+ ;NO, STORE ^C IN TABLE
24000 013124 104013      SCRLF          ;SEND A CR LF
24100 013126 012702 013164      1$: MOV #BUFR,R2     ;BUFFER ADDRESS TO R2
24200 013132 021227 000003      CMP (R2),#3      ;CHECK IF FIRST CHAR IS ^C
24300 013136 001733      BEQ E024B       ;YES, LOOK FOR INPUT AGAIN
24400 013140 112200      2$: MOVB (R2)+,R0   ;GET CHARACTER
24500 013142 020027 000003      CMP R0,#3       ;DONE STRING?
24600 013146 001767      BEQ 1$          ;YES, RESTART STRING
24700 013150 104015      PRINTC        ;PRINT CHAR
24800 013152 000772      BR 2$          ;CONTINUE
24900 013154 104007      TERM: TYPEM     ;OUTPUT TERMINATION MESSAGE
25000 013156 014272      ECOEND
25100 013160 104005      CHAIN          ;CHAIN TO NEXT TEST
25200 013162 000721      BR E024B       ;REPEAT TEST
25300 013164 000003      BUFR: 3          ;INITIALIZE FIRST CHAR AS CNTL-C IN TABLE
25400 013166      .BLKB 256. ;256 CHARACTER BUFFER
    
```

```

25600      :XXXXXXXXXX
25700      :
25800      :E025-- BELL ECHO TEST-- A MESSAGE IS PRINTED AND
25900      :          THE TEST WAITS FOR SOME PRINTABLE CHARACTER
26000      :          TO BE SELECTED ON THE KEYBOARD (GT040). THIS
26100      :          TEST IS VALID ONLY IF THE PAPER WIDTH IS GT 64
26200      :          COLUMNS. IF LT64 COLUMNS AN ILLEGAL BELL TEST
26300      :          MESSAGE IS PRINTED.
26400      :
26500      :XXXXXXXXXX
26600      :
26700 013566 000025      E025: 25          ;TEST NUMBER
26800 013570 012116      E020          ;NEXT TEST HEADER
26900 013572 104016      PRTHDR        ;PRINT HEADER
27000 013574 026727 165052 000101 1$: CMP WIDTH,#101 ;TEST IF COLUMN COUNT IS EQ,GT 64
27100 013602 103427      BLO 4$        ;BRANCH IF NOT
27200 013604 104007      TYPEM          ;TYPE TEST MESSG
27300 013606 014145      E025MA        ;
27400 013610 000402      BR 3$          ;WAIT FOR CHAR
27500 013612 104000      2$: TYPE        ;TYPE TEST MESSG ON TERM CHAR RCVD ON
27600 013614 014145      E025MA        ;
27700 013616 104020      3$: READ        ;WAIT FOR OPERATOR RESPONSE
27800 013620 012700 000036      MOV #30.,R0    ;DELAY FOR HALF DUPLEX
27900 013624 104010      DELAY        ;
28000 013626 026727 165046 000040      CMP TEMPCH,#40 ;TEST IF PRINTABLE
28100 013634 103770      BLO 3$          ;BRANCH IF NON-PRINTABLE
28200 013636 022767 000177 165034      CMP #177,TEMPCH ;CHECK IF CHAR IS RUBOUT
28300 013644 001410      BEQ 5$          ;BRANCH IF YES
28400 013646 104017      PRNT          ;CHECK IF PRINTER IS READY
28500 013650 116777 165024 164742      MOVB TEMPCH,@TPB ;PRINT CHAR. (BELL SHOULD SOUND)
28600 013656 104013      SCRLF        ;SEND A CRLF
28700 013660 000754      BR 2$          ;REPEAT
28800 013662 104007      4$: TYPEM        ;TYPE ERROR MESSAGE
28900 013664 014245      E025MB        ;
29000 013666 104007      5$: TYPEM        ;PRINT TERMINATION
29100 013670 014272      ECOEND        ;
29200 013672 104005      CHAIN        ;EXIT TO NEXT TEST
29300 013674 000737      BR 1$          ;REPEAT TEST
    
```

29500

```

100          .SBTTL  MISC. DIAGNOSTIC MESSAGES
200
300 013676    007    002    200  STARTM: .ASCII  <7><2><ACRLF><17>/CZLACEO LA36 TERM (DL11 & KL11)/<ACRLF>
    013701    017    103    132
    013704    114    101    103
    013707    105    060    040
    013712    114    101    063
    013715    066    040    124
    013720    105    122    115
    013723    040    050    104
    013726    114    061    061
    013731    040    046    040
    013734    113    114    061
    013737    061    051    200
400 013742    114    101    063    .ASCII  /LA36 TERMINAL DIAGNOSTIC/<ACRLF>
    013745    066    040    124
    013750    105    122    115
    013753    111    116    101
    013756    114    040    104
    013761    111    101    107
    013764    116    117    123
    013767    124    111    103
    013772    200
500 013773    104    114    061    .ASCIZ  /DL11 & KL11 INTERFACE/<ACRLF><12>
    013776    061    040    046
    014001    040    113    114
    014004    061    061    040
    014007    111    116    124
    014012    105    122    106
    014015    101    103    105
    014020    200    012    000
600 014023    200    012    105  ENDPAS: .ASCII  <ACRLF><12>/END OF PASS  /
    014026    116    104    040
    014031    117    106    040
    014034    120    101    123
    014037    123    040    040
700 014042    060    060    060  PASMES: .ASCIZ  /0000/<ACRLF><12>
    014045    060    200    012
    014050    000
800 014051    200    103    117  DL11S: .ASCII  <ACRLF>/CONSOLE & /
    014054    116    123    117
    014057    114    105    040
    014062    046    040
900 014064    060    060    040  DL11S1: .ASCIZ  /00 DL11'S UNDER TEST/<ACRLF><12>
    014067    104    114    061
    014072    061    047    123
    014075    040    125    116
    014100    104    105    122
    014103    040    124    105
    014106    123    124    200
    014111    012    000
1000 014113    007    002    200  HDRMSG: .ASCIZ  <7><2><ACRLF><17><12>/TEST #/
    014116    017    012    124
    014121    105    123    124
    014124    040    043    000
1100 014127    060    060    060  HDRO: .ASCIZ  /000 COLUMNS/<ACRLF><12>
    014132    040    103    117

```


	014135	114	125	115	
	014140	116	123	200	
	014143	012	000		
1200	014145	124	131	120	E025MA: .ASCII /TYPE ANY PRINTABLE CHARACTER /
	014150	105	040	101	
	014153	116	131	040	
	014156	120	122	111	
	014161	116	124	101	
	014164	102	114	105	
	014167	040	103	110	
	014172	101	122	101	
	014175	103	124	105	
	014200	122	040		
1300	014202	101	116	104	.ASCIZ /AND LISTEN FOR BELL...../
	014205	040	114	111	
	014210	123	124	105	
	014213	116	040	106	
	014216	117	122	040	
	014221	102	105	114	
	014224	114	056	056	
	014227	056	056	056	
	014232	056	056	056	
	014235	056	056	056	
	014240	056	056	056	
	014243	056	000		
1400	014245	200	116	117	E025MB: .ASCIZ <ACRLF>/NOT ENOUGH COLUMNS/<ACRLF>
	014250	124	040	105	
	014253	116	117	125	
	014256	107	110	040	
	014261	103	117	114	
	014264	125	115	116	
	014267	123	200	000	
1500	014272	200	105	103	ECOEND: .ASCIZ <ACRLF>/ECHO TEST TERMINATED/<ACRLF>
	014275	110	117	040	
	014300	124	105	123	
	014303	124	040	124	
	014306	105	122	115	
	014311	111	116	101	
	014314	124	105	104	
	014317	200	000		
1600	014321	040	040		E023M: .ASCII / /
1700	014323	040	040	040	LINE5: .ASCII / / ;MSG FOR TEST E024
	014326	040			
1800	014327	040	040	040	LINE5A: .ASCIZ / /<ACRLF>
	014332	040	040	040	
	014335	040	040	200	
	014340	000			
1900	014341	200	017	012	MESG3: .ASCIZ <ACRLF><17><12>/SELECT TEST NUMBER /
	014344	123	105	114	
	014347	105	103	124	
	014352	040	124	105	
	014355	123	124	040	
	014360	116	125	115	
	014363	102	105	122	
	014366	040	040	000	
2000	014371	105	126	105	EVEN: .ASCII /EVEN/
	014374	116			

2100	014375	117	104	104	ODD: .ASCII /ODD /
	014400	040			
2200	014401	124	131	120	OPMSG: .ASCIZ /TYPE ANY CHARACTER/
	014404	105	040	101	
	014407	116	131	040	
	014412	103	110	101	
	014415	122	101	103	
	014420	124	105	122	
	014423	000			
2300	014424	125	123	105	NOSWR: .ASCIZ /USE SOFTWARE SWITCH REG AT MEMORY ADDR 176/<7>
	014427	040	123	117	
	014432	106	124	127	
	014435	101	122	105	
	014440	040	123	127	
	014443	111	124	103	
	014446	110	040	122	
	014451	105	107	040	
	014454	101	124	040	
	014457	115	105	115	
	014462	117	122	131	
	014465	040	101	104	
	014470	104	122	040	
	014473	061	067	066	
	014476	007	000		
2400					
2500					
2600	000001				.END

ACRLF = 000200	CHAINN 001412	E025MA 014145	PRGTAB 002522	START3 000772
ADJR4 012032	CHAINY 001534	E025MB 014245	PRINTC= 104015	STLSPV 003406
ADTEMP 004100	CHALT = 104006	ERR 003320	PRNT = 104017	STLSRV 003356
AREAD = 104021	CHLT 000720	ERRHLT 003354	PRTHDR= 104016	STPCHV= 104004
ATO 005266	CKPOS 011764	ERROR = 104001	PRTY4 = 000200	STPPA 003424
ATOX 005270	CNTLSW 000634	EVEN 014371	PRTY7 = 000340	STPRA 003374
AT1 005320	CNTR 000716	FORWD = 104024	PSW = 177776	STRDRV= 104003
AT10 005744	CNVCTR 004072	FORWDA 003614	PT0 007372	STRLN 012660
AT11 006002	CONADD 000602	FORWDB 003622	PT1 007446	SWREG 000176
AT12 006042	CONIT 003710	FSTDL 000632	PT10 011266	TEMP 000712
AT13 006116	CONSET 003714	HDRMSG 014113	PT11 011424	TEMPCH 000700
AT14 006176	CONVEC 000604	HDRO 014127	PT12 011456	TENPWR 004076
AT15 006262	COUNT3 000666	HERE 011532	PT12A 011464	TERM 013154
AT16 006362	CR = 104022	ICTR 000660	PT17 011546	TESTC 002460
AT17 006430	CRBUF 000646	IDEZ 007676	PT17B 011536	TIMER 000672
AT2 005352	CRLF = 104012	INCHK 000710	PT17D 011576	TKB 000614
AT20 006500	CTRA 000650	LEVEL 000654	PT2 007570	TKLVL 000624
AT21 006572	CURTST 000676	LF = 104014	PT3 010164	TKS 000612
AT22 006672	DELAY = 104010	LCFNT 000706	PT4 010304	TKVTR 000622
AT23 007000	DIGIT 004074	LINE3 010444	PT5 010462	TPB 000620
AT24 007112	DIRTN 012112	LINE5 014323	PT5AL 010630	TPBS 004004
AT25 007212	DISPRE 000174	LINE5A 014327	PT6 010666	TPLVL 000630
AT26 007270	DLADR 000606	LOGICA 011522	PT7 011054	TPS 000616
AT3 005404	DLCNT 000656	LSI11 = 001000	READ = 104020	TPSS 004002
AT4 005436	DLNR 000610	MACHER 000004	READC = 104025	TPVTR 000626
AT5 005526	DLY 003436	MESG3 014341	READ1 004212	TTYCTL= 104011
AT6 005604	DL11S 014051	MG24 013010	REF 010414	TTY1 001776
AT7 005674	DL11S1 014064	MG25 013014	REPT 000662	TTY1B 002062
BIT0 = 000001	ECOEND 014272	MONIC 012272	RESTRT 003516	TYP 003076
BIT1 = 000002	EHALT = 104002	MOVNUM 012736	RTNNO 000636	TYPE = 104000
BIT10 = 002000	EHLT 003346	NEXT 001654	SAVR6 003514	TYPEM = 104007
BIT11 = 004000	EMTINT 002722	NEXT1 001674	SCOPSW= 040000	TYPM 003164
BIT12 = 010000	EMTTAB 002776	NITRSW= 004000	SCOPTR 000642	WAITF 001700
BIT13 = 020000	ENDPAS 014023	NOSWR 014424	SCRLF = 104013	WIDTH 000652
BIT14 = 040000	END2 001332	NPCODE 010137	SELHLT 000734	XCSR 000670
BIT15 = 100000	END2A 001320	NXTST 000640	SKIP 001630	\$AREAD 003640
BIT2 = 000004	END3 001262	ODD 014375	SPARET 003072	\$BTASC 004006
BIT3 = 000010	END4 001334	OPEN = 000000	SPBOT 000600	\$CR 003226
BIT4 = 000020	E020 012116	OPMSG 014401	SPC 007560	\$CRLF 003214
BIT5 = 000040	E021 012166	OUTPUT 013112	SPCNT 000674	\$FORWD 003562
BIT6 = 000100	E021A 012174	PARITY 000702	SPSP 007540	\$LF 003216
BIT7 = 000200	E022 012224	PASCNT 012114	SP2 007552	\$PRHDR 003236
BIT8 = 000400	E022A 012232	PASMES 014042	SR 000714	\$PRNT 004314
BIT9 = 001000	E023 012476	PCHAR 000704	START 001010	\$PRTC 004324
BRCTR 000664	E023M 014321	PFAIL 003460	STARTM 013676	\$READ 004112
BTOASC= 104023	E024 013020	POPSP = 005726	STARTX 001024	\$READC 004204
BUFR 013164	E024B 013026	POPSP2= 022626	START1 000736	\$SCRLF 003142
CHAIN = 104005	E025 013566	PRGID 000644	START2 000754	

. ABS. 014500 000
000000 001
ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 2846 WORDS (12 PAGES)
DYNAMIC MEMORY: 3844 WORDS (14 PAGES)
ELAPSED TIME: 00:00:52
CZLACE.BIN,CZLACE.LST/-SP=CZLACE.MAC