(protocol) Protocol for TENEX <-> Line Processor interactions

Introduction

This document is a detailed description of the Line Processor
protocol. It is intended to serve as a guide to anyone wishing to
implement the Line Processor protocol, as well as, a piece of
documentation for the Line Processor.

It should be pointed out here that the Line Processor contains a
very small, slow microcomputer with little read/write memory. For
this reason the protocol is terse and error reports and/or
recovery almost non-existant. The Line Processor terminal is
treated more as a hardware device than an intelligent terminal.

There are two types of line processors - alpha and graphic. Alpha
line processors are used in configurations consisting of the line
processor alpha/numeric display, mouse, keyset, and possibly a
hard copy printer or a cassette drive. Graphics line processors
are used in the the minimum graphics configuration consisting of
a/n display, mouse, keyset, and either a Tektronix 4012 or 4014
storage tube display.

Conventions

Coordinates

Alpha

Coordinates designate character positions. For example
(1,1) is the second character on the second line up from the
bottom.

The origin is at the lower left corner of the screen.

As components of the protocol, coordinates are passed as one
byte of X and one of Y and always have 40B added to them to
get them in the printing character range. This limits the
max coordinate value to 137B which is 95 decimal.

Graphics

The mouse is used to track the cursor on either the a/n
display or the Storage tube. A switch acts as a toggle to
select which screen is to be tracked. Coordinate values are
identical to the alpha line processor when they originate
from the a/n display, although they are sent as two bytes
each of x and y. Graphics coordinates from the storage tube
are sent as 10 bit values in the range 1024 to 2047, with
1024 at the lower left of the screen.

TTY Simulation

In TTY simulation, scrolling always takes place on a line feed
(LF) not a carriage return (CR). Carriage return does the
obvious thing and no more.

Special and Control Characters

Protocol strings begin with 33B and are followed with an operation type character in the range 40B to 120B.

When outside a protocol string, all control characters (0 thru 37B) are ignored by the Line Processor, except:

When the cursor is being tracked:

^G which rings a bell if possible

CR and LF which do the right thing

Notice that backspace character (^H) is not implemented in TTY simulation (i.e. when the cursor is being tracked).

When the cursor has been positioned:

^G which rings a bell if possible

^H which does a backspace cursor

When inside a protocol string, RUBOUT is NOT ignored. When outside, it is ignored.

Conventions for this document

In this document, octal numbers are followed by "B".

"Unescorted" means that characters are sent as is without wrapping them in an protocol sequence.

Line Processor to Main Computer Protocol

Communication in this direction will adhear generally to the IMLAC protocol as outlined in (IJOURNAL,14345,).

In particular:

Keyboard characters 40B thru 177B are unescorted.

Keyboard characters 0 thru 37B are sent as:

34B, 43B, char+140B, coordinates

NOTE: An alternate (and preferred) way is to send these control characters as is (unescorted) except for 2B, 4B and 30B. Those are sent as above.

Mouse button changes are send as:

34B, 43B, buttons+100B, coordinates

where buttons is the binary image of button positons (000

thru 111 binary).

Keyset strokes 1 thru 32B are send as:

stroke+140B (e.g. 1 -> a)

keyset strokes 33B thru 37B are sent as:

33B -> 54B (,)

34B -> 56B (.)

35B -> 73B (;)

36B -> 77B (?)

37B -> 40B (space)

For alpha line processors coordinates are X + 40B, Y + 40B.

For graphics line processors coordinates are X(bits 10 - 6 (MSB's)) + 40B, x(bits 5 - 0 (LSB')) + 40B, Y(bits 10 - 6) + 40B, Y(bits 5 - 0).

When not in coordinate mode the mouse buttons are ignored and keyboard control characters (0 thru 37B) are sent in unescorted fashion.

At power-up and after the "system-reset" button is pushed, the Line Processor signals the Main computer by sending:

(176B, 177B)

The purpose of this is to indicate to the applications program that the Line Procesor is now in a "power-up" state (see below).

When the Line Processor detects an error that it cannot live with, it sends a string to the applications prorgam and dies with an error code flashing in the lights. The user is then forced to hit "system-reset". The string is as follows:

(176B, 41B, Ccount', Chars)

Where Ccount' is 40B more than the number of characters that follow. Currently 8 characters are sent, and the string looks like:

(176B, 41B, 50B, err', ctl', trk', rpt', sw', obuf', b1', b2')

where the ' indicates that 40B has been added.

err: The error code, one of

10B = output buffer to display overun (impropper

padding).

11B = some other buffer overun (e.g. printer buffer)

12B = strange error relating to display output buffer.

13B = protocol sequence error (e.g. bad comand)

14B = protocol value error (e.g. bad coordinate).

ctl: control state parameter (0 = not in a command)

trk: mouse tracking code:

0 = positioned

1 = tracking

2 = cursor in small TTY window

5: cursor at unknown position

12B = Cursor in full screen window

rpt: repeat code, normally zero

sw: sense switch immage in order 0-1-2-3 (sw3=LSB)

obuf: display output buffer character count

b1: possibly low order 4 bits of last input char

b2: possibly high order 4 bits of last input char

From Main Computer to the Line Processor

The following functions are sent by the applications program and
performed by the Line Processor. All codes, except the escape
(33B) should be printing characters. Padding characters should be
RUBOUTs (177B). The baud rate factor (f) and and display type are
obtained by the applications program by sending an interogate
command.

Note:

The cursor is generally used to track the mouse. Some
commands stop the tracking and allow the cursor to be used
for display manipulation. "Tracking mode" refers to whether
the mouse is being tracked by the cursor or not.

Display-terminal dependent parameters:

The following table yields the timing and other factors
required by the protocol that depend on the type of terminal
connected to the Line Procesor. That type, DItype, is
obtained from the interogate command (see below).

param		Ditype=		
1	2	3	4	
Del 80	7	1	17	
Ins 0	7	30	17	
Clr 5	6	3	17	
Xmark	No	Yes	Yes	Yes

Del is the time to delete a line.

Ins is the time taken to insert a new line.

Clr is the time taken to clear the screen.

Xmark indicates if a marked character needs to be re-written after the mark is removed.

See the interrogate response command for other display parameters.

Position cursor on alpha display and stop tracking mouse.

Send(33B, 40B, X', Y')

X' = X coord (0 thru Xmax) + 40B

Y' = Y coord (0 thru Ymax) + 40B

result:

Positions cursor to specified location. Tracking stops until a "resume tracking" or a reset is received. Any unescorted characters will be written on the screen and the cursor will be advanced once after each character. Writing beyond the end of the line is not advised as the result depends on the terminal manufacturer and model.

Specify (small) TTY simulation window on alpha display

Send(33B, 41B, top, bottom)

top = Y' for top line of window

bottom = Y' for bottom line of window

result:

Invokes a small TTY simulation window of specified size and location. This window will be used until a new one is specified or a reset is received. This does not change the tracking mode.

Reset

   Send(33B, 51B)

   result:

      screen cleared

      TTY simulation window set to full screen

      bug selection stack reset

      resume tracking (see)

   padding:

      Send pads as for clear screen.

Resume tracking mouse

   Send(33B, 42B)

   result:

      The cursor is used to track the mouse. Any unescorted
      characters will go into the TTY simulation window
      currently in use.

Write string of blanks

   Send(33B, 43B, N')

   N' = number of blanks to be written.

   result:

      The specified number of blanks are written starting at
      the current cursor position. The cursor is left at the
      character position following the last blank. Assumes the
      cursor has been positioned appropriately beforehand.

      This command is a no-op if N' is not >= 41B AND <= 177B.

   padding:

      This command must have N/f padding characters following
      it.

Push bug selection

   Send(33B, 46B, X', Y')

   result:

      The coordinates are pushed on a stack and the character

at that location is somehow brought to the user's
attention. The stack will hold a maximum of 8 selections.
This command includes a resume tracking.

padding:

This command must have 8/f padding characters following
it.

Pop bug selection

Send(33B, 47B)

result:

The top entry on the bug selection stack is popped. The
corresponding character on the screen is no longer marked
in a special way. If the stack is empty, this command is
a no-op. This command includes a resume tracking
operation.

For some DItypes, the applications program must restore
the character or the marked position will be replaced by
a space.

padding:

This command must have 8/f padding characters following
it.

Delete selected line

Send(33B, 44B)

result:

The cursor position selects a line to be removed from the
screen. All following lines are moved up one line. The
contents of the last line are undefined. The X
coordinate should be zero, otherwise the results are
undefined.

padding:

This command requires Del/f padding characters (Del is
obtained from the table).

Insert selected line

Send(33B, 45B)

result:

The line which the cursor is on, and all following lines,
are moved down one line. The cursor is not moved, and
hence is on a blank line. Lines above the cursor are not

altered. The last line (before the execution of this command) should be considered "lost." The X coordinate should be zero, otherwise the results are undefined.

padding:

This command requires Ins/f padding characters (Ins is obtained from the table).

Clear screen

Send(33B, 50B)

result:

The entire screen is cleared. The cursor position is not generally known. The TTY simulation window location and the bug selection stack are not altered. The tracking mode is not changed.

padding:

This command requires Clr/f pad characters;

Interrogate line processor

Send(33B, 55B)

result:

A response to the interrogate command is sent as a protocol string of this form:

34B, 46B, Xmax+40B, Ymax+40B, LPtype, Dtim Rate

Where

Xmax is the maximum x coordinate

Ymax is the maximum y coordinate

LPtype is in [40B-177B] and designates type

The least significant four bits of LPtype designate display terminal type (call it DItype)

Currently defined are:

(1) Delta Data 5200

(2) Hazeltine H2000

(3) Data Media Elite 2500

(4) Lear Siegler ADM-2

The most significant three bits designate Line
Processor type (call it Type)

Currently defined are:

(0) Complete alpha line processor with
copy printer receiver for cassette drive

(2) Line Processor with Mouse, Keyset,
Printer

(6) Graphics line processor with Tektronix
4014

(7) Graphics line processor with Tektronix
4012

Dtim is a characteristic delay time. For proper
scrolling, a line feed (LF) must be followed by
(Dtim+14)/f pad characters.

Rate indicates the Line Processor receive baud
rate:

300 buad: 100B, f=32 decimal

600 baud: 60B, f=16

1200 baud: 50B, f=8

2400 baud: 44B, f=4

4800 baud: 42B, f=2

9600 baud: 41B, f=1

The baud rate factor, f _ Rate-40B;

Note: Any additions to LPtype should be assigned by
ARC personel for best results. See DIA or CHI
@SRI-ARC.

This command does not change the tracking mode.

Turn off coordinate mode

Send(33B, 60B)

result:

Turns off the coordinate mode in the Line Processor. This
does not change the tracking mode.

Mouse buttons become inactive, keyboard control
characters sent to main computer without protocol
formating.

Turn on coordinate mode

    Send(33B, 61B)

    result:

        Turns on the coordinate mode in the Line Processor. This
        does not change the tracking mode.

        Mouse buttons become active, keyboard control characters
        are sent in input protocol format.

Begin standout mode

    Send(33B, 56B)

    result:

        All following text written on the screen will be altered
        is some way from "normal" text. This unfortunately
        includes characters which go into the TIY simulation
        window also, so don't leave the line processor in this
        state indefinitely. Does not change the tracking mode.

End standout mode

    Send(33B, 57B)

    result:

        Subsequent text written on the screen will be in "normal"
        mode. Does not change the tracking mode.

TENEX RESTARTING

    The Line Processor will detect a TENEX restart, by looking
    for the ten 33B's it sends out at startup time. At that
    time it will place itself in a state as though the hardware
    reset button had been pushed.

Open printer (alpha line processor only)

    Send(33B, 53B)

    Result:

        Opens the printer for output. Protocol to the printer
        must be observed: (1) open it. (2) wait for protocol
        string "request" (below). (3) send strings in response to
        requests. (4) close it.

        "Request" string, sent back to the main computer:

            OB NULL

Each request enables the application program to
send an additional 16 characters via the printer
string protocol below.

Note: The count indicates the Line Processor storage
allocated for the next printer string. Sending a longer
string will result in a "receive error" (error light on
panel).

Close printer (alpha line processor only)

Send(33B, 54B)

Result:

Closes the printer. Actual close will not take place
until all characters in the output buffer are printed.
That is, the close may follow the last string of
characters immediately. It is possible (but very
unlikely) that a "request" protocol string may be sent to
the main computer after the close is sent to the Line
Processor.

Printer string (alpha line processor only)

Send(33B, 52B, Dev, Count+40B, <characters>)

Result:

The Dev is normally 40B and is ignored by Line Processors
with one printer. The Count must not be greater than the
sum of the counts in all "request" protocol string not
already fulfilled. It may be less. The actual character
string may contain any characters. They will be sent to
the printer without translation or special handling.

Note:

Strings may be sent to the printer without opening it if
timing constraints are observed carefully. In this case
the applications program must know the baud rate of the
printing device a well as the Line Processor - Main
computer line. The program just issues printer strings
and no requests are sent back to the Main computer by the
Line processor. This was a deliberate implementation to
allow higher speed printing over networks without waiting
for the response. Observe that if strings are sent too
fast the printer buffer in the Line Processor will
overflow: data will be lost and the Line Processor will
die. The printer buffer normally holds 47 characters..

Open graphics display (graphics line processor only)

Send(33B, 53B)

Result:

Disables mouse tracking on the graphics display.

Close graphics display (graphics line processor only)

Send(33B, 54B)

Result:

Ensables mouse tracking on the graphics display.

Write graphics display (graphics line processor only)

Send(33B, 52B, Dev, Count+40B, <characters>)

Result:

The Dev is normally 40B and is ignored by Line
Processors. Characters from the application program are
written directly on the graphics display. Since the
characters are not buffered, the graphics display must be
connected at a higher baud rate than the external
processor.

Set graphics cursor resolution (graphics line processor only)

Send(33B, 62B, N')

Result:

N controls the mask applied to the cursor coordinates
before they are used to position the cursor on the
graphics display:

N = 0   Mask = 0

    = 1   = 1 LSB is cleared (etc)

    = 2   = 3

    = 3   = 7

    = 4   = 17B

    = 5   = 37B

Application notes:

Avoid writing text (or "string of blanks") beyond the end of a
line: the display may insert an unwanted line or drop the extra
characters.

Avoid positioning the cursor to any x>Xmax or y>Ymax.

Avoid doing an insert line on the last line: the display may
scroll the entire screen.

Delta Data (DItype=1) must be treated as a special case in the
following respect:

When writing text at (x,y) on a line which does not already
have text on it up to position x (e.g. after a clear screen
or insert line), the applications program must send x/f pad
characters after the first character written at position
(x,y). The display takes that long to move a CR symbol into
the proper display memory location. (Our thanks to Delta
Data).

We expect to stop supporting Delta Datas soon.

NOTE:

The Line Processor has a reset button on it (which will be used
only on rare occations). After power up or a hardware reset,
the following state prevails:

The screen is clear, the mouse tracking in operation.

The bug selection stack is empty.

The full screen TTY simulation is in effect.

Coordinate mode is NOT in effect.

Printer is closed

All TTY simulation windows currently work as follows: Text is
inserted in the last line and "scrolling" occurs on each line
feed (i.e. it does not start on the top line of the window as
you may prefer). A CR moves the cursor to left margin, a LF
effects a line break. Typing beyond the last character of the
line causes a line "wrap" - i.e. new text replaces the old
line, starting from the left margin. The only way to clear a
small TTY window is to send N line feeds into it, where N is
the number of lines in the window.

The usual sequence from the applications program will be to
position the cursor and perform some function, or write text,
or both. It must end such a sequence with a "resume tracking"
command. Any broadcast messages, links, etc. that come down
the line between the cursor position and the "resume tracking"
will go wherever the cursor happens to be.

Normally, broadcast messages and the like will go into the
TTY simulation window. The difference being that they are
not preceeded by a position cursor command.

REENTER code in NLS will clear and repaint the entire screen

Mouse tracking will be done by the Line Processor under the
following conditions:

IF the terminal has received a "resume tracking" command
since the last position cursor command, AND

IF there is no input from the TEN, AND

the mouse coords have changed since the last mouse tracking
operation, or the cursor has been moved since the last mouse
tracking operation.

Tracking stops under the following conditions:

A position cursor command comes from the TEN.

Summaries

Line processor to Exernal processor

CHAR SEQUENCE	MEANING
(all line processors)	
CHARACTER	Normal Character

(Ascii values 1B to 177B except 0 (String request), 2 (^B),
4 (^D), 34B (BCESC), and 176B (Reset))

BCESC 46 MX MY TP DT BD	Interrogate Response
176 177	System Reset
176 41 CCNT CCHRS	Error report

(alpha line processors)	
BCESC 43 CC X Y Control Chars	Optional Sequence For
BCESC 43 CC X Y ^B (CDOT), ^X (CD)	Sequence For ^D (CA),
BCESC 43 MB X Y Buttons	Sequence For Mouse
0 (NULL)	String request

(graphics line processors)	
BCESC 45 CC X1 X2 Y1 Y2	Optional Sequence For Control

Chars

BCESC 45 CC X1 X2 Y1 Y2   Sequence For ^D (CA), ^B (CDOT),
^X (CD)

BCESC 45 MB X1 X2 Y1 Y2   Sequence For Mouse Buttons

where:

  All numbers are in octal

  CCNT = number of CCHRS + 40

  CCHRS = CCNT-40 data bytes; each byte is offset by 40

  CC = control character + 140

  MB = current mouse button state + 100

  X = current x corrdinate + 40

  Y = current y corrdinate + 40

  X1 = top 6 significant bits of x coordinate + 40

  X2 = least significant 6 bits of x coordinate + 40

  Y1 = top 6 significant bits of y coordinate + 40

  Y2 = least significant 6 bits of y coordinate + 40

  MX = maximum x coordinate + 40

  MY = maximum y coordinate + 40

  TP = line processor type and version + 40

  DT = terminal delay time characteristic + 40

  BD = line processor receive baud rate + 40

Exernal processor to Line processor

COMMAND	CODE	PADDING
position	33B, 40B, X', Y'	none
TTY window	33B, 41B, Y TOP', Y BOTTOM'	none
resume tracking	33B, 42B	none
write blanks	33B, 43B, N'	N/F

delete line	33B, 44B	DEL/F
insert line	33B, 45B	INS/F
push bug	33B, 46B, X', Y'	8/F
pop bug	33B, 47B	8/F
clear screen	33B, 50B	CLR/F
reset	33B, 51B	CLR/F
printer string	33B, 52B, DEV, CNT', String	see text
open printer port	33B, 53B	see text
close printer port	33B, 54B	none
interrogate	33B, 55B	none
standout mode on	33B, 56B	none
standout mode off	33B, 57B	none
coordinate mode off	33B, 60B	none
coordinate mode on	33B, 61B	none
cursor resolution	33B, 62B, N'	none
remote resart	10 - 33B's	none