# TracList manual

This document has been compiled on 2024-04-19.

# Contents

# Section 1: Purpose

TracList is a pager that is meant to display NASM listing files. It connects to the output of lDebug. Then the pager will trace through the listing file as the debugger steps through the program. The pager updates its trace automatically while in the background, allowing for convenient use if multiple terminal windows are available. The input focus can stay on the debugger terminal window then.

# Section 2: Building

TracList is an interpreted Perl program, so doesn't need to be build itself. However, you need a recent Perl, and the following modules from CPAN:

- Curses
- Audio::Beep
- Syntax::Keyword::Try
- Getopt::Long
- File::Spec
- Switch
- File::HomeDir
- File::Path

traclist.pl also uses signals; SIGUSR1 is essential for automatic update. If the `--gofile` option is used it expects the environment variable `$PWD` to hold the pathname of the current working directory.

tractest.c is a standard C program, except that it uses `usleep()` and `kill()` with the signal `SIGUSR1`. Compile it with any C compiler.

# Section 3: Overview

The debugger connects to a serial port within its environment. Its output goes to that port and it receives its input from there too.

The other end of the serial I/O connection is hooked up to a picocom process. This allows controlling the debugger from another terminal. The output of picocom is piped into tee, which appends to the trace log file.

The tractest C program keeps re-reading the trace log file. It first checks the file size and assumes an update has occurred if it changed. Then it scans the file from the back to read up to 16 line references. A line reference is detected from the debugger's disassembly output format. The line references are normalised and written to the line file. After finding 16 line references or encountering the start of the file, the program goes to sleep for 0.1 seconds before checking the trace log again. If the content written to the line file changed then an alert is initiated. After finishing the scan (either scanned up to start of file or wrote 16 line references) the alert is carried out, sending a SIGUSR1 signal to the processes indicated in the PID files.

The traclist.pl program accepts one or multiple listing files as parameters. It reads the line file when instructed to do an update, either from a user command or when signalled with SIGUSR1. It then tries to match the selected line reference with the opcode listings. If found, the program jumps within the listing to view the line match.

# Section 4: Example Setup

Set up debugger I/O terminal like this:

```
$ socat pty,link=/tmp/vptty-dos,rawer \
  pty,link=/tmp/vptty-linux,rawer & disown
$ picocom /tmp/vptty-linux | tee -a /tmp/trace.log
```

In another terminal enter the lDebug bin directory and then set up the DOS machine like so:

```
$ dosemu -I "serial { com 2 device /tmp/vptty-dos }" \
  -K "$PWD" -E "ldebugu.com lddebugu.com" -dumb
```

To use qemu instead of dosemu2, you can set up a machine like this:

```
$ qemu-system-i386 -chardev serial,id=serial2,path=/tmp/vptty-dos \
 -serial null -serial chardev:serial2 [further options here]
```

Enter the following command to the debugger:

```
-install serial
```

The DCO variable change will pop up a line like this on the I/O terminal:

```
lDebug connected to serial port. Enter KEEP to confirm.
```

Enter "KEEP" to that prompt to confirm the connection.

In yet another terminal, enter the tractest directory and start the tractest C program:

```
$ ./tractest /tmp/trace.log /tmp/trace.lin /tmp/trace.pid
```

Note that the .LOG file should be the same as the one specified to tee. The third filename parameter defaults to `/tmp/trace.pid` if not specified. Subsequent filename parameters are allowed to specify additional PID files.

In a fourth terminal, start traclist like this:

```
$ ./traclist.pl --listing path/to/lst/ddebug.lst \
  --line /tmp/trace.lin --pid /tmp/trace.pid
```

The `--line` and `--pid` options and their filename parameters are optional. The line and PID filenames default to those given here. If to use a goto line file such as for Pluma, append the `--goline` and `--gofile` options:

```
$ ./traclist.pl --listing path/to/lst/ddebug.lst \
  --line /tmp/trace.lin --pid /tmp/trace.pid \
  --goline /tmp/pluma-goto-line --gofile /tmp/pluma-goto-file
```

Note that the .LIN file and the .PID file should be the same as those specified to the tractest C program.

Now you can use TracList.

# Section 5: Usage

Most commands are accepted either capitalised or uncapitalised, but a few commands behave differently depending on capitalisation.

## 5.1 Command Q - Quit

The Q command quits the TracList application. The PID file is deleted by this.

## 5.2 Command U - Update

The U command makes TracList reread the .LIN file and selects the most recent entry (first line) from the file. If the U is entered capitalised and trace mode is off then the drawing line is not changed. That is, the referenced line is highlighted (if visible) but display will not jump to make sure to make it visible.

## 5.3 Command R - Reload

This is similar to the U command, except that it first closes the listing file and re-opens it. This may be needed to discard buffered content in case the listing file is rewritten. The R can be entered capitalised, similar to the U command, in order to (if trace mode is off) avoid jumping to the highlighted line.

## 5.4 Command T - Trace or Untrace

This command toggles trace mode. If trace mode is enabled, then any update (from commands U, R, or upon receiving the USR1 signal) will jump within the listing file to display the newly-highlighted lines. Further, movement within the listing file is limited to the area around the highlighted lines. If trace mode is disabled then movement is free throughout the entire listing file.

## 5.5 Command M - Multi or Unmulti

This command toggles multi file mode. If multi mode and trace mode are enabled, then updates will search through all open files in order. Additionally, if both modes are enabled then the C commands are not available.

## 5.6 Command A - Auto-Multi, reload, then unmulti

This command enables multi file mode, updates the tracing, then disables multi file mode. The motive is to speed up tracing, which can be achieved by disabling multi file mode. However, when tracing through different programs it is desirable to briefly enable multi file mode to find the correct file whenever the control flow jumps to another listing. Instead of enabling multi file mode then disabling it again with two consecutive M commands, a single A command can be used.

## 5.7 Command C - Cycle through listing files

This command allows one to cycle manually through multiple listing files. All listing files have to be specified on the TracList script's command line when starting the pager. Depending on whether the ‘`C`’ is capitalised or not, cycles backwards or forwards through the specified files.

## 5.8 Command S - Switch priority order of listing files

This command swaps the currently active listing file with the one before it in the internal list of files. This is used to change the order of the files. It is only valid if the currently active file is not the first one yet.

## 5.9 Command O - Open new listing file

This command brings up a filename prompt. Entering a filename allows opening a new listing file. The prompt can be aborted by entering Control-C or using the Backspace key on an empty input. The new listing file is appended to the end of the internal list of files. It is opened as the active listing file.

## 5.10 Command E - End an open listing file

This command closes the currently active listing file. It removes that file from the internal list of files. This is only valid if more than one file is currently opened. The index of the active listing file is preserved unless the closed file was last. That is, the next file following the file to be closed is opened as active.

## 5.11 Command H - Reload history files

The search pattern input, filename open input, and offset input all have their own histories. Usually, TracList will initialise the histories during the application startup. The histories are read from the following files:

~/.local/share/.traclist/search_history

~/.local/share/.traclist/file_history

~/.local/share/.traclist/offset_history

Whenever a line is entered into one of these inputs, TracList will append that line to the corresponding history file. The only exception is if the last line in the file matches the newly entered line exactly then it is not entered again.

The H command causes TracList to reload the histories from the files. Usually, if someone else modifies the history files then TracList will not change its in-memory histories to pick up the new content. This command allows forcing TracList to do just that.

## 5.12 Command I - Enter new offsets

This command brings up a prompt to enter new offsets. Offsets are accepted the same as for the --offset application option. In particular, an offset input entry consists of three fields.

The first field is the filename to which to restrict an offset. It is terminated by a colon. It must match the regexp ‘`[-_.\$A-Za-z0-9]+`’. Offsets restricted to a name are only used when a listing file with a matching basename is searched. The basename is matched capitalisationsensitively. The filename is optional. If not specified, or specified as the string '`-`', then the offset is entered into the internal offset list that is not restricted to filenames.

The second field is the priority to assign. It is also terminated by a colon. The priority is optional. A priority of zero (most important) is assumed if the field is not present or the priority is the empty string.

The third field is the offset. It can be given numerically in decimal, or in hexadecimal. Hexadecimal input requires a '`0x`' prefix or a '`h`' suffix. The offset given is assigned the given priority. If it did not exist yet in the internal list of offsets, the offset is added to that list.

As a special case, if the priority is given as '`-`' (a dash) then the corresponding offset is deleted. As another special case, if the offset is given as '`-`' and no filename is specified then all existing offsets in the internal list of offsets not restricted to filenames are given the specified priority, or (if both special cases apply) deleted. The special case for offset '`-`' when combined with a filename field will set the priority of or delete all entries in the internal list of offsets associated with that particular filename. Yet another special case is if the offset is given as '`--`' and no filename is specified. In this case, all offset list entries are affected, including all entries of all the filename-scoped offset lists.

Examples:

```
-:--
```

Delete all entries.

```
-:-
```

Delete all entries that are not specific to a filename.

```
256
```

Enter priority = 0 and offset = 256 into the list not specific to a filename.

```
boot12.lst::7C00h
```

Enter priority = 0 and offset = 7C00h into the list specific to the filename '`boot12.lst`'.

```
ddebug.lst:-:256
```

Delete entry for offset 256 specific to the filename '`ddebug.lst`'.

```
1:0
```

Enter priority = 1 and offset = 0 into the list not specific to a filename.

## 5.13 Command Left and Right - Decrement or Increment stop position

The stop position specifies how many columns at the line start are hidden from display. There are a number of hardcoded stop positions and then high stops. High stops are at every 8th column, matching the tabstops used in our files. The hardcoded stops are as follows:

0

Display at the start of the line.

7

Hide the source line numbers in the listing file.

11

Hide the source line numbers and the high words of the offsets.

16

Hide the source line numbers and the offsets.

40

Hide the line numbers, offsets, and opcode listings. (This is the same as displaying only the original source.)

## 5.14 Command Tab - Cycle stop position

This command cycles through the hardcoded stop positions. It never chooses any of the high stops. If the current stop position is the highest hardcoded stop position then this command cycles back to 0.

## 5.15 Command F and B - Forwards and Backwards line reference

These commands cycle through the lines referenced in the .LIN file. Backwards means to switch to the next older entry. Forwards means to switch to the next newer entry. The tractest C program currently emits up to 16 entries. If the command letter is entered capitalised, and trace mode is off, then the display does not jump to make sure the highlighted lines are made visible.

## 5.16 Command N and P - Next and Previous line match

These commands cycle through matches for the same selected line referenced from the .LIN file. They are only available if a line is matched to multiple candidate listing lines. The pager will attempt to select the most likely line by itself. Using N and P allows disabling the automatic selection and cycling through the candidate matches.

## 5.17 Movement commands

### 5.17.1 Up command

Moves the listing a line up. Does not move the listing if it already is at the very first line. Also does not move the listing file if trace mode is on and the highlighted line is already at the bottom-most line considered visible. Beeps if movement is not possible.

### 5.17.2 Down command

Moves the listing a line down. Does not move the listing if it already is at the very last line. Also does not move the listing file if trace mode is on and the highlighted line is already at the top-most line considered visible. Beeps if movement is not possible.

### 5.17.3 Page Up command

Moves the listing file a page up. Similarly restricted as the Up command.

### 5.17.4 Page Down command

Moves the listing file a page down. Similarly restricted as the Down command.

### 5.17.5 g command - Move to first line

(This command depends on capitalisation.) Moves the listing file to the first valid line. This is the very first line of the file if not in trace mode. In trace mode, the display is moved up as much as possible while keeping the highlighted line at the bottom-most line considered visible. Beeps if in trace mode and further movement is not possible.

### 5.17.6 G command - Move to last line

(This command depends on capitalisation.) Moves the listing file to the last valid line. This is the very last line of the file if not in trace mode. In trace mode, the display is moved down as much as possible while keeping the highlighted line at the top-most line considered visible. Beeps if in trace mode and further movement is not possible.

## 5.18 Search commands - '`/`' and '`?`'

In tracing mode, the search commands can be used to highlight matches within the viewed area. If not in tracing mode, upon a successful search the view additionally jumps so as to put the next match into the first line of the view. Search patterns can match an arbitrary regular expression. The slash '`/`' command searchs forwards starting from the line currently at the top of the view. The question mark '`?`' command instead searchs backwards. Both of the commands do not wrap around upon reaching the start or end of file.

The search pattern input prompt that comes up upon entering one of the search commands allows simple line editing:

A regular text codepoint

> Insert codepoint at cursor position.

Left and Right

> Move cursor position.

Backspace

> Delete codepoint before the cursor position.

Backspace (when current input is empty)

> Abort search pattern input.

Del

> Delete codepoint at cursor position.

Home

> Move cursor position to start of line.

End

> Move cursor position to end of current input.

Up and Down

Recall search pattern history. (Only valid if input not yet edited.)

Enter

Submit current input as search pattern.

Control-C

Abort search pattern input.

Submitting empty input as a search pattern will automatically choose the most-recent search pattern from the history. Line matching (when not in trace mode) is slightly different depending on whether a search pattern was entered from an empty input or not.

The search pattern can start with the sequence ‘`(?i)`’ to search ignoring capitalisation. By default, capitalisation is observed.

# Section 6: Errors

The second-to-last line is displayed in ‘standout’ mode. It contains error messages if a number of error conditions occur. If a new non-empty error message is displayed, a beep is produced. The following error messages can occur:

Invalid entry in line file

> An update occurred, and there was a line found in the .LIN file that does not match the expected format. (The format is `SEGSEL:OFS OPCODES`, where `SEGSEL` is a 16-bit hexadecimal word, `OFS` is a 32-bit or 16-bit hexadecimal word, and `OPCODES` are a list of 8-bit hexadecimal bytes without delimiters.)

Empty line file

> An update occurred, and there were no lines found in the .LIN file.

Wrong selection

> An update occurred, and there are less instructions listed in the .LIN file than needed to fulfil the current selection.

Invalid double bracket opening

> An update occurred, and one of the opcode listings has two opening brackets in a row.

Invalid bracket closing

> An update occurred, and one of the opcode listings has a closing bracket without a corresponding opening bracket.

Line not found

> An update has occurred, and no line was found for the selected reference.

Cannot open input file: (Error condition)

> The R or C or O command was used and the listing file could not be opened.

Missing additional listing file for switching

> The S command was used while only one listing file had been specified.

Listing file is already first in list

> The S command was used but the active listing file already is first.

Pattern not found

> A search command entered while not in tracing mode did not match any lines.

Invalid search pattern

A search command received an invalid regular expression as input.

No search pattern in history

A search command received an empty input while no search history patterns were available.

File already opened

An open command received an input matching an existing opened file.

No filename in history

An open command received an empty input while no filename history entries were available.

Must leave open at least one file

An end command was entered while only one file is open.

# Section 7: Display elements

The following display elements make up the display of the pager.

Listing file content

All lines except for the last three. The selected referenced lines are highlighted (standout), except for the matched opcodes in the opcodes listing (not standout). Search matches are highlighted. (Standout if not in referenced line, no standout if in referenced line.) If a given line continues past the end of the display, the last display column contains a ‘`>`’ that is highlighted (standout).

Opcode line

The third-to-last line. This displays the literal contents of the selected line from the .LIN file. After that, it displays how many bytes are given in the opcode list. Near the end of this line there is a display of the current listing filename. It is given without any of the pathname's directory components. At the end of this line, there is a display of how many lines are available from the .LIN file. This is given in a format like ‘`  1/16`’. Omitted if only one line is available in the .LIN file.

Error Line

The second-to-last line. This displays error conditions as specified in section 6. It is also used as the area for the search pattern input prompt when one of the search commands was used. Likewise it is used for the filename input prompt when the O command was used.

Command Line

The last line. This lists some of the currently available commands. The Q, U, R, and T commands are always displayed. The C and S commands are displayed if available. The F, B, N, and P commands are displayed if available. The Increment and Decrement stops commands are displayed if available. The vertical movement commands are not displayed. The search commands are not displayed.

# Section 8: Options reference

The following options may be specified when starting the TracList application.

## 8.1 `--listing` - Specify list files

This option can appear one or more times. All files specified to this option are opened as listing files. Use the ‘`C`’, ‘`c`’, and ‘`S`’ commands to cycle between open files.

## 8.2 `--offset` - Specify offsets and offset priorities

This option can appear any amount of times. Each specified offset is taken as one offset to use to match line references to line matches. For example, when the offset ‘`256`’ is specified then a line reference of ‘`0126`’ will match a listing line that starts with the address ‘`00000026`’. An offset is specified in decimal if only digits are given, or in hexadecimal with a ‘`0x`’ prefix or a ‘`h`’ suffix.

Each offset may be prefixed by a priority, separated from the offset with a colon ‘`:`’. The priority specifies which offsets to prefer if there are multiple line matches. The lowest priority value is given the highest precedence. If two offsets share a priority and both of them match to the same line reference then the one that appears first in the listing file is preferred. If no priority is specified then a priority value of zero is assumed. If the same offset is specified multiple times, then the last occurrence will be treated as authoritative as to the priority.

If this option does not appear at all, then a default list of offsets is used. This list is equivalent to specifying the following options:

```
--offset 0:0 --offset 1:256 --offset 2:7C00h
```

For more details on what offsets can be specified, refer to section 5.12.

## 8.3 `--local-offset` - Specify offsets for last file

This option can occur any amount of times. This option depends on the last listing file specified prior to it on the application's command line. The offset can be specified as for the `--offset` option, or as for command I (refer to section 5.12) without a filename. The filename is automatically set to the basename of whatever listing file was last specified. Therefore, this option sets a file-local offset. If this option occurs before any `--listing` options then the offset is entered into the non-file-scoped internal list.

Note that if this option occurs at all, it will also prevent the three default offsets effectively used as listed for if the `--offset` is not used.

## 8.4 `--multi` - Set multi file mode default

This option can occur any amount of times, but only the last setting is applied. This option can be prefixed as in `--no-multi` to disable multi file mode. Otherwise, it enables multi file mode. This is the same mode as toggled by the M command, refer to section 5.5. The default is off.

## 8.5 `--trace` - Set trace mode default

This option can occur any amount of times, but only the last setting is applied. This option can be prefixed as in `--no-trace` to disable trace mode. Otherwise, it enables trace mode. This is the same mode as toggled by the T command, refer to section 5.4. The default is on.

## 8.6 `--line` - Specify line file

This option can occur any amount of times. However, only the last occurrence is effectively used. This option takes a parameter that specifies the pathname of the line file, usually named with a `.LIN` filename extension. If this option does not appear, then a default name is used. The default name is equivalent to specifying the following option:

```
--line /tmp/trace.lin
```

## 8.7 `--pid` - Specify PID file

This option can occur any amount of times. However, only the last occurrence is effectively used. This option takes a parameter that specifies the pathname of the PID file, usually named with a `.PID` filename extension. The PID file is overwritten by the TracList application and is deleted upon exitting the application. If this option does not appear, then a default name is used. The default name is equivalent to specifying the following option:

```
--pid /tmp/trace.pid
```

(Note: To use multiple instances of TracList with the same tractest process, you can specify a unique PID file to each instance. All files to use must be specified to tractest.)

## 8.8 `--goline` - Specify go-to-line file

This option can occur any amount of times. However, only the last occurrence is effectively used. This option takes a parameter that specifies the pathname of the Go To Line file. Upon finding a line match, the application truncates this file then writes the line number of the match to it. This can be used by other scripting to open an editor at the specified line. If this option does not occur, or the specified name is the empty string, then go-to-line file writing is disabled.

## 8.9 `--gofile` - Specify go-to-file file

This option can occur any amount of times. However, only the last occurrence is effectively used. This option takes a parameter that specifies the pathname of the Go To File file. Upon finding a line match, the application truncates this file then writes the filename of the match to it. This can be used by other scripting to open an editor with the specified file. If this option does not occur, or the specified name is the empty string, then go-to-file file writing is disabled.

# Section 9: Scripts for use

## 9.1 Pluma snippet for go to line

To create this snippet, enter Pluma's Snippet Manager and create a new global snippet. Enter the following content for the snippet:

```
$<
import os
cmd='pluma "'
cmd += '$(cat /tmp/pluma-goto-file)'
cmd += '" +$(cat /tmp/pluma-goto-line)'
os.system(cmd)
# return cmd
>
```

Enter a Shortcut key for the snippet, such as `Shift+Alt+E`. Then, start `traclist.pl` with the following options:

```
--goline /tmp/pluma-goto-line --gofile /tmp/pluma-goto-file
```

Now whenever a line match is found by the TracList application, you can press `Shift+Alt+E` in Pluma to open the match's position in the corresponding listing file.

## 9.2 Scriptlet to truncate trace log file

This truncates the trace log file, to lessen the space it uses up and speed up the tractest C program's task. Because tee is used with its `-a` switch, it will correctly append to the start of the file after this.

```
$ truncate --size=0 /tmp/trace.log
```

Note that the tractest C program and TracList do not have to be restarted to use this scriptlet.

# Source Control Revision ID

hg f34bc863449f, from commit on at 2024-04-19 22:44:43 +0200

If this is in ecm's repository, you can find it at https://hg.pushbx.org/ecm/tractest/rev/f34bc863449f