Using The TLA700 Series Logic Analyzers Integrated Software Tools

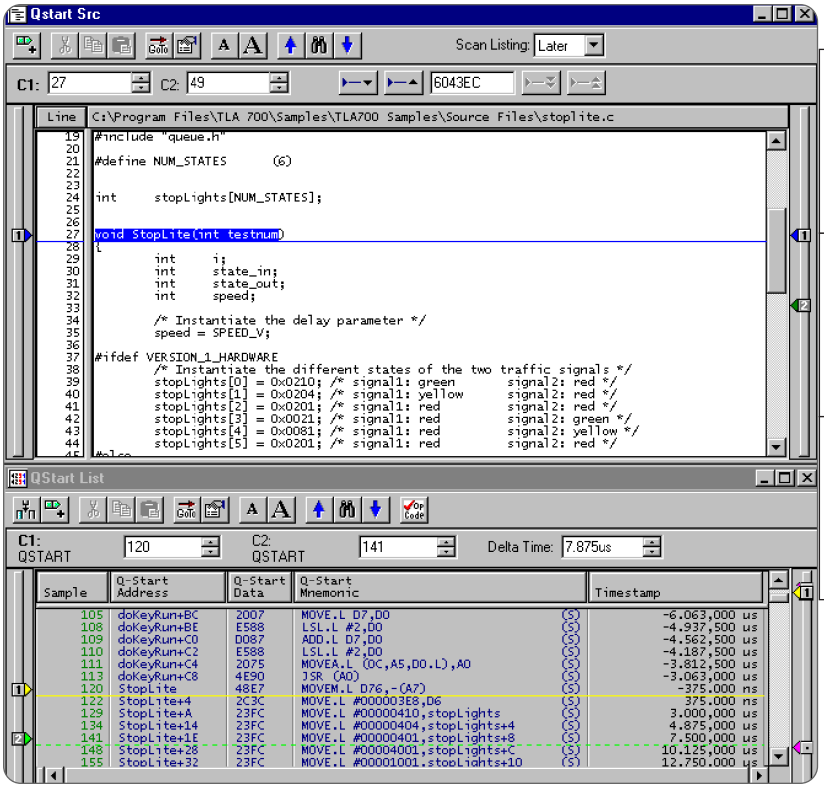


Figure 1. *Source Window where user's source code is displayed.*

Embedded software developers are responsible for the flawless execution of their software in a real-time embedded system. They must ensure that both the input and output data-flow processing and system time constraints are validated to design specifications. Valid measurements, however, can only be performed on the actual target system; this requires very close collaboration with the hardware developers on the team.

When debugging embedded target systems, the digital design team often encounters four types of problems:

- **Logical Problems.** These are simple mistakes in logic design or coding, or an incorrect design assumption. These problems are often caught by the hardware simulator or by the software debugger. This type of problem often comprises 80% of the debug problems encountered, but often only consume 20% of the debug time. While tedious and time-consuming to find, they are relatively straightforward to fix. The remaining 20% of the problems fall into the remaining three categories and may take up to 80% of the debug time to find and fix.
- **Hardware/Software Interaction Problems.** These problems are more difficult to debug and often require some form of physical trace tool. The logic analyzer has been the traditional physical trace tool of choice for such problems. Logic analyzers provide both the hardware and software analysis capability to examine the interaction of the hardware and the software in a target system. With the ability to correlate specific signals with the processor's instruction execution flow, the hardware and software developers can debug complex problems such as why certain instructions cause memory errors or why a specific data pattern causes crosstalk-induced glitches on the processor's data bus.
- **Real-time Software Problems.** These are the most difficult problems to debug – problems that occur only when the target system is running at full speed. Logic analyzers excel in solving these problems because they run at the full speed of the target system and provide powerful triggering and deep trace memory to capture the most elusive real-time faults.

Tektronix
Enabling Innovation

- **Crash Problems.** Embedded systems differ from non-embedded systems (e.g., PCs or workstations) in that they generally do not have protection from an errant program crashing the target system. Robust operating systems, such as those found on PCs or workstations, have a variety of mechanisms to isolate the system from a misbehaving application – embedded systems often do not. Thus when the software on an embedded system crashes, it often takes the whole target system down, thereby losing any information that might be useful in determining the root cause. Logic analyzers, especially those with deep acquisition memory, can provide the real-time history of the target system, up to and including the crash, to aid in quickly determining the cause of the crash.

Solving these problems requires a tool that both the embedded software and hardware developer can use to resolve the difficult cross-domain problems that span the range from high-level language source code down to the underlying signal behavior.

The TLA700 Series logic analyzer provides a set of tools for the embedded software developer that acquire data from a target system at the analog, timing, and state levels, thereby providing a "big picture" overview of target system behavior from signals to source code. All of these tools are standard on every TLA700 – no optional tools to buy or configure. Four such tools are discussed here:

1) High-level language source code support

2) Performance analysis

3) Symbol support

4) Magnitude mode support

High-Level-Language Source Code Support

The TLA700's High Level Language (HLL) source code support, which is integrated and standard on every TLA700, consists of a data window – the Source Window – in the TLA700 application where the user's source code is displayed (see Figure 1). Three things are required to use this support:

1) Which column in the TLA700's Listing Window (usually the processor's Address column) that the TLA700's Source Window should link to.

2) Where the object file (which must be compiled and linked for debug) which contains the source symbols is located.

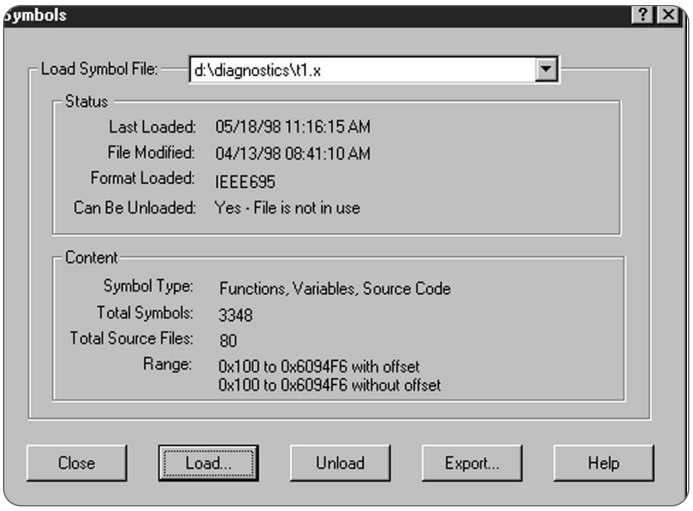
3) Where the source files are located.

The key to how the Source Window operates is the symbol file. In addition to the name and value of the symbol, it contains the line number and address of each source line. With this information, the Source Window can correlate the source code that the developer wrote with the acquired data in the Listing Window.

There are a variety of mechanisms for navigating the data. Within the Source Window, you can move to the next or previous executed source statement using the toolbar buttons, scroll or page through the source file using the window sliders, move either of the two cursors, or switch to a different source file. From the Listing Window, you can move either of the cursors. When you move a cursor in either the Source or Listing Window, the corresponding cursor in the other window moves as well.

You can set a trigger condition based on the source line in the Source Window. You can also control all of the various window properties (e.g., variable font size, colors, show or hide line numbers, and tab spacing).

You can have multiple source windows for either different source files or for multi-processor support.



▶ ***Figure 2.*** *Loading symbols.*

Symbolic Support

The symbolic information contained within the object file that is loaded on the target system is the key to debugging at a symbolic and HLL source code level.

There are two types of symbol files:

1) **Pattern Symbol Files.** Pattern symbols have one value (the pattern) which can contain don't care (X) values. Pattern symbols are often used for displaying variable names and in the control group of a disassembler display.

2) **Range Symbol Files**. Range symbols have two values – an upper and lower bound – but cannot contain don't care (X) values. Range symbols are often used in the address group of a disassembler display. Range symbols are found in the software executable file or object file that is downloaded to the target system. Three types of symbols can be found within this file:

   ▶ **Function –** Range symbols describing the beginning and ending addresses of software functions.

   ▶ **Variable –** Range symbols describing the beginning and ending addresses of software variables.

   ▶ **Source Code –** Range symbols describing the beginning and ending addresses of source statements. The TLA700 Source Window uses these range symbols to perform the linkage with the acquired data in the Listing Window. Any missing line numbers contained either comments or non-executable source lines. The "Beg" and "End" columns are generated by software tools to provide statement-level granularity within a line which the TLA700 Source Window can use to highlight individual statements in a single source line.

The TLA700's symbol file capability is integrated and standard on every TLA700. Symbols are loaded directly from the object module or executable file (see Figure 2). Symbol files must contain absolute addresses. There are no limits to the number of symbols that can be loaded (limited only by the amount of virtual memory in the TLA700's Windows® 95 PC). The following file formats are supported:

– IEEE695

– OMFx86

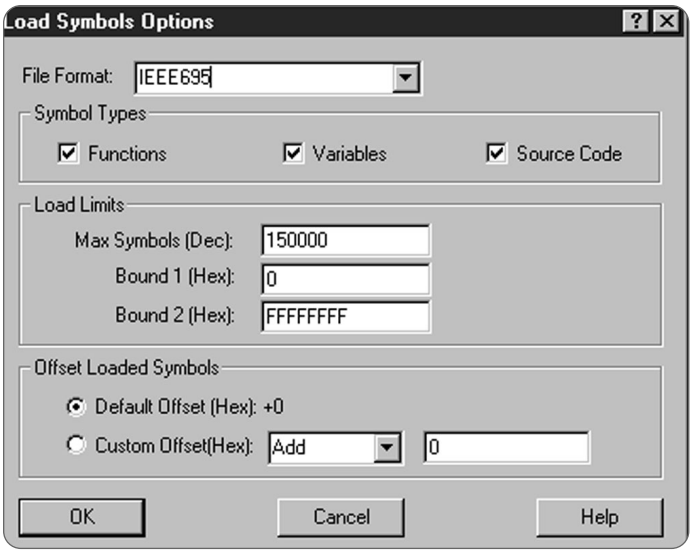
– COFF

– ELF/Dwarf

– ELF/Stabs

– ASCII

Once loaded, the same symbols are shared between the Source Window, Listing Window, Histogram Window (Performance Analysis), and LA Module trigger.



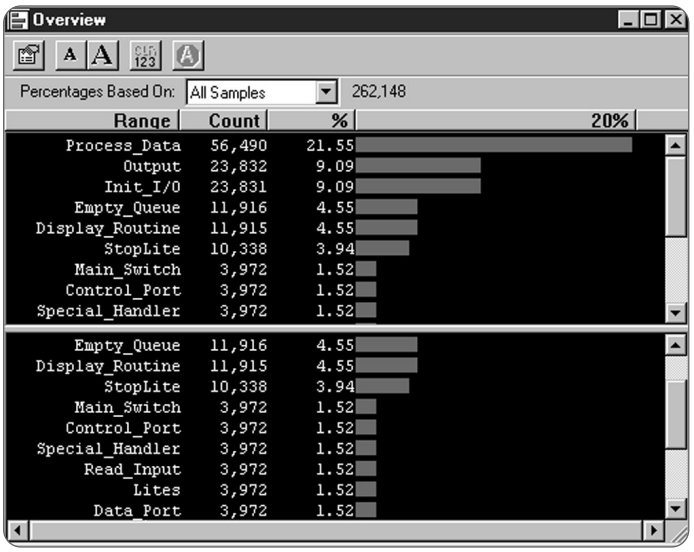
▶ ***Figure 3.*** *Symbol options.*

In the Load Symbol operation, you can access the Load Symbol Options which allows you to choose which symbols to load, limit the number of symbols to load, limit the address range of symbols to load, and add/subtract an offset to all symbols (see Figure 3).

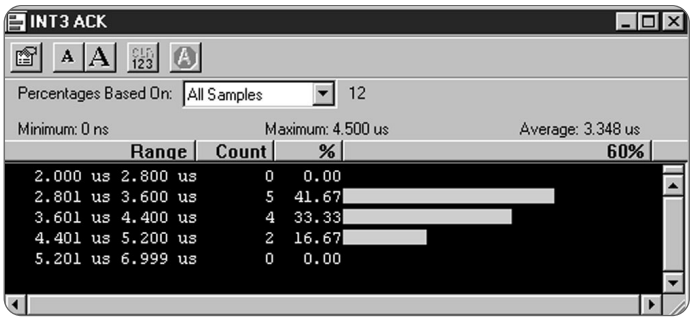
You can also export a loaded symbol file and edit the symbols using any word processor such as WordPad and then reload the edited symbol file. By exporting a loaded symbol file, you can view what the actual values are for each symbol. Understanding the contents of a loaded symbol file can be very useful in understanding how the various TLA700 software tools operate. For additional information, please refer to Appendix B of the TLA700 User's Manual.

Performance Analysis Support

Today's embedded software applications are getting increasingly larger, which makes it difficult to see the "big picture" of the overall flow and execution time of the software. Often times, the embedded software developer gets the code functioning correctly, but performance tuning is usually done near the end of the project. An often quoted rule of thumb is that "20% of the code executes 80% of the time;" however, the big question is which 20% of the code! What's needed is some form of overview tool to show which of the hundreds of software modules are consuming the majority of the processor's execution time. The TLA 700's Performance Analysis (PA) capability shows where the software is spending its time. With this information, the embedded software developer can quickly zero in on the routines that, if optimized, have the greatest payback in improved performance. The TLA700's PA support, which is integrated and standard on every TLA700, provides two measurement types:



▶ ***Figure 4.*** *Range overview.*



▶ ***Figure 5.*** *Single event (counter/timer).*

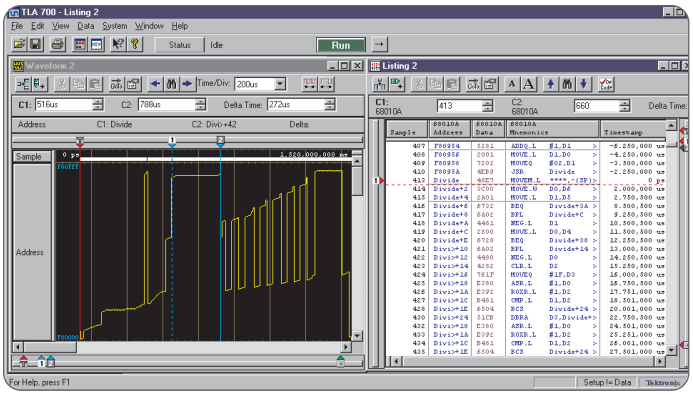
▶ **Range Overview (Channel Groups)**. With range overview, the TLA700 analyzes the data acquired for a user-selected logic analyzer module group and displays the hits in each range in a histogram format. It can use the same loaded symbol table (with absolute addresses) with no limit on the number of ranges. Ranges can be generated and displayed using either numbers (linear or logarithmic) or from a loaded symbol file. Data can easily be sorted (ascending or descending) by either range, count, or percent. The Histogram window can be split to view both the beginning and end of the ranges. Various properties can be selected including colors, variable fonts, and channel polarity. You can also choose by percent based upon all samples or only matched samples (see Figure 4).

Range overview is especially useful for providing information on which software modules consume the most execution time. The developer must be careful, however, to ensure that recorded hits are actually measuring processor execution time, not prefetch or other non-execution activity.

▶ **Single Event (Counter/Timer)**. Often times, however, developers have a routine that has a specified execution time for which they need to validate that it never exceeds the specification. Single event is a measurement mode that uses the logic analyzer module's 4 ns counter/timers to display the range of execution time/counts for a single routine. By using the TLA700's logic analyzer trigger machine to define an event, you can measure that event with 4 ns resolution. The resulting display shows the minimum, maximum, and average time a single event takes to execute over multiple runs (see Figure 5).

This capability is useful in monitoring time-critical routines such as interrupt/exception handlers to ensure that the system specification for maximum service time is never violated.

You can have multiple histogram windows for each logic analyzer module. This is useful for multi-processor support.



▶ ***Figure 6.*** *Magnitude Mode.*

Magnitude Mode Support

An often overlooked, but useful, feature of the TLA700 for debugging embedded software is the TLA700's Magnitude Mode which is available within the Waveform Window. Magnitude Mode displays the value of each group versus time. This capability is valuable as a "digital-to-analog" display, e.g., displaying the value of digitized video data.

Magnitude mode can also be used to speed up embedded software debug by displaying the value of the processor's address group over time. This is particularly useful as an overview tool. The information provided shows where in the memory map the processor is spending it's time. If it appears that the processor is spending an inordinate amount of time in a particular memory region (usually indicated by a relatively flat line), you can move either of the cursors anywhere on the line and open a new Listing Window or Source Window. You can then see what particular code is executing and whether this is a symptom of a deeper problem. If needed, you can always rely upon the TLA700's 500 ps timing resolution or 200 ps analog capability, to quickly zero in on the underlying root cause (see Figure 6).

Conclusion

The TLA700's integrated software tools, such as HLL source code support, symbolic capabilities, performance analysis, and magnitude mode support, allow embedded software developers to effectively work with hardware developers in solving difficult cross-domain problems. Third-party HLL source-level debuggers with run-control installed on the TLA700 provides all members of the digital design team with a powerful set of hardware and software tools to resolve those difficult cross-domain problems.

Contact Tektronix:

ASEAN Countries & Pakistan (65) 6356 3900

Australia & New Zealand (65) 6356 3900

Austria +43 2236 8092 262

Belgium +32 (2) 715 89 70

Brazil & South America 55 (11) 3741-8360

Canada 1 (800) 661-5625

Central Europe & Greece +43 2236 8092 301

Denmark +45 44 850 700

Finland +358 (9) 4783 400

France & North Africa +33 (0) 1 69 86 80 34

Germany +49 (221) 94 77 400

Hong Kong (852) 2585-6688

India (91) 80-2275577

Italy +39 (02) 25086 1

Japan (Sony/Tektronix Corporation) 81 (3) 3448-3111

Mexico, Central America & Caribbean 52 (55) 56666-333

The Netherlands +31 (0) 23 569 5555

Norway +47 22 07 07 00

People's Republic of China 86 (10) 6235 1230

Poland +48 (0) 22 521 53 40

Republic of Korea 82 (2) 528-5299

Russia, CIS & The Baltics +358 (9) 4783 400

South Africa +27 11 254 8360

Spain +34 (91) 372 6055

Sweden +46 8 477 6503/4

Taiwan 886 (2) 2722-9622

United Kingdom & Eire +44 (0) 1344 392400

USA 1 (800) 426-2200

For other areas contact Tektronix, Inc. at: 1 (503) 627-7111

For Further Information

Tektronix maintains a comprehensive, constantly expanding collection of application notes, technical briefs and other resources to help engineers working on the cutting edge of technology. Please visit **www.tektronix.com**

03/02 OA/XBS 52W-12480-1

Tektronix
Enabling Innovation