TAU/BLAST Support in 2.1i

XAPP166 August 9, 1999 (Version 1.0) Application Note

Summary

The Xilinx 2.1I development system adds Stamp Model Generation. This feature supports the use of board level Static Timing Analysis tools, such as Mentor Graphics' Tau and Viewlogic's Blast. With these tools, users of Xilinx programmable logic products can accelerate board level design verification.

TAU/BLAST Support in 2.1i

For FPGA families, the Xilinx 2.1i program **trce** has the ability to produce STAMP files that will be used to pass the timing data about the Xilinx FPGA to Tau/Blast. By default **trce** will report back all timing paths covered by constraints (the .pcf file), but this may not be enough information in the STAMP file, since it is possible that not all inputs and outputs are reported, as they might not be covered by user constraints. To force **trce** to evaluate all the paths the following options should be used:

- `-a` for advanced design analysis in the absence of a constraint file (.pcf)
- `-u` will report additional paths that are not covered by the constraints within the PCF file. This option MUST be used when specifying the PCF file to ensure a complete STAMP file is created covering all paths.

By default **trce** will not use the constraint file (.pcf) when it is named differently than the routed .ncd file, and it has to be specified on the command line when running **trce**. The `–stamp <stamp_file>` option must be used to generate the `<stamp_file>.mod` and `<stamp_file>.dat` files. In the following example the speed grade as well as the part type was identified when the Xilinx implementation was run, so the `–s <speed>` option is not needed as the speed is embedded in the ncd file. Users must use the `–u` option when specifying the pcf file to ensure all paths are covered by the STAMP file.

Command line for **trce** to generate stamp models:

```
trce  –u  –stamp  <stamp_file>  <design>.ncd   <design>.pcf
```

In absence of <design>.pcf file invoke **trce** as follows:

```
trce  –a  –stamp <stamp_file>  <design>.ncd
```

NOTE: Xilinx 2.1i **trce** does not write STAMP data for both min and max timing values into the same (one) data file. Because of this we need to invoke **trce** twice to generate both STAMP data files. The first run is for max delay and the second run is for min delays.

1. First **trce** run

   ```
   trce  –u  –stamp  <design>  <design[.ncd]>  <constraint[.pcf]>
   ```

   This will generate two files

   - `design.mod` – stamp model file
   - `design.data` – stamp max delay data file

2. Second **trce** run

```
trce -u -s min -stamp <design_min> <design[.ncd]> <constraint[.pcf]>
```

This will generate two files

- design_min.mod – stamp model file (same as design.mod)
- design_min.data – stamp min delay data file

NOTE: The -s min option currently supports the XC4000XL/XLA families in 2.1i and will support the Virtex family shortly after the 2.1i initial release. The min delays reported are the absolute minimum delays (i.e. irrespective of any speed grade) for the part selected.

For CPLD families, the Xilinx 2.1i program **taengine** produces STAMP files that will be used to pass the timing data about the Xilinx CPLD to Tau/Blast. The **taengine** program is normally run as part of the CPLD fitter flow to generate the static timing report (.tim). The taengine program always produces STAMP model files that cover all timing paths in the CPLD device, regardless of which ones are covered by timing constraints.

Please refer to the Xilinx Development System Reference Guide for more information about **trce** and all the available options.

TAU

Tau is a Board-Level timing analysis tool from Mentor Graphics designed to do static timing analysis. Tau will check if timing constraints such as setup and hold requirements on component inputs are met. To determine if these requirements are satisfied it is necessary to take into account interconnect delay on the board, component delay, and the skew and phase shift between clocks. This board-level timing analysis is performed in the digital domain. This analysis can also be performed before physical design begins in order to identify the interconnect delay constraints that must be satisfied by a board.

Taulib is the tool used to import the cell timing for the Xilinx device cell. From the board level, the Xilinx FPGA or CPLD devices are considered cells. Within **taulib** the STAMP information can be imported by selecting the cell name that represents the chip, if it already exists, by clicking on the very left hand box next to the name of the cell. This will highlight the entire row. If the cell does not already exist taulib will automatically create one when the Stamp Model is imported.

The following three files are needed to import the stamp model into **taulib**:

- design.mod (or design_min.mod)
- design.data
- design_min.data

The stamp model (.mod) and then the data (.dat) are imported by using the File → Import pulldown menus. If the timing information is to be read and the data already exists, choose Override Existing Timing Model. If a new timing model is to be created choose the default option (Create New Timing Model). Choose the appropriate Timing Value to interpret the delay values in the data file as either minimum or maximum. After clicking OK you can select the Cell Timing sheet to examine the imported timing information.

Please refer to the Mentor Tau documentation for more information on running taulib, tauform, and tau.

BLAST

BLAST is a Viewlogic static timing analysis tool designed for board and system level timing analysis. BLAST identifies all setup and hold time violations in a PCB design, exhaustively tracing every signal delay path. The exhaustive timing analysis enables BLAST to report any violation that could cause the system to fail. To perform the timing analysis on a PCB design, BLAST requires the timing information of the ICs placed on the PCB. The timing information of the Xilinx FPGAs is supplied in the industry standard STAMP model files.

The `design.data` file in first **`trce`** run should be renamed as `design_max.data` in order to use the Viewlogic utility **`pt2blast`** v1.9. The following three files are needed by **`pt2blast`** to generate the BLAST model:

- `design.mod`
- `design_min.data`
- `design_max.data`

The last two files are used to generate one data file `design.data` and then `design.data` and `design.mod` are used to generate the BLAST model (`design.lib`).

`pt2blast` needs to be invoked in the directory where the STAMP model `files (design.mod, design_max.data, design_min.data)` are located. `Pt2blast` contains four file windows. The `design.mod` are read in the first window and `design_max.data` and `design_min.data` files are read in the third window . The files are read automatically when `pt2blast` is invoked in the STAMP files directory. The input files can be viewed by double clicking on them. Enter the `design_min.data` and `design_max.data` files in the Xilinx specific Min/Max data files window. Click the *`Create Xilinx data file`* button to generate the embedded min and max data file. Any error in the design file entry will be reported in the status field. **`pt2blast`** names the embedded data file as `design.data` by default. The generated `design.data` file is displayed in the second file window. Select the `design.mod/design.data` file and press the *`Translate`* button to create `design.lib`, the BLAST model.

Please refer to the Viewlogic BLAST documentation for more information on running **`pt2blast`** and BLAST.