1

Introduction to PrimeTime and the Tutorial

PrimeTime is the Synopsys stand-alone full chip, gate-level static timing analyzer. It analyzes the timing of large, synchronous, digital ASICs. PrimeTime works with designs at the gate level, providing a tight integration with other Synopsys tools.

PrimeTime differs from Design Compiler and DesignTime. Because PrimeTime is stand-alone, it does not require all the data constructs required for logic synthesis, so its performance is greater than Design Compiler and DesignTime and its memory requirements are lower. PrimeTime has features that make it suitable for analyzing large, system-on-a-chip (SOC) designs. PrimeTime is synthesis-compatible, and you can use it throughout your design process.

This chapter introduces PrimeTime and the tutorial. It tells you about

- Basic features and capabilities
- PrimeTime timing analysis flow and methodology

- Getting started using PrimeTime
- Preparing to run the tutorial
- Starting and exiting PrimeTime and getting help
- Paging output from PrimeTime
- The tutorial design
- Setting up the timing analysis

Basic Features and Capabilities

PrimeTime can perform the following timing checks on a design:

- Setup and hold checks
- Recovery and removal checks
- Clock pulse width checks
- Clock gating checks

PrimeTime can perform the following design checks:

- Unclocked registers
- Unconstrained timing endpoints
- Master-slave clock separation
- Multiple clocked registers
- Level-sensitive clocking

- Combinational feedback loops
- Design rule checks (maximum capacitance, maximum transition time, and maximum fanout)

Other PrimeTime advanced analysis features, which are explored in this tutorial.

PrimeTime Timing Analysis Flow and Methodology

PrimeTime follows the same timing analysis flow and methodology you use for DesignTime.

1. Set up the design environment.
   - Set the search path and the link path.
   - Read the design and the libraries.
   - Link the top design.
   - Set up the operating conditions, wireload models, port load, drive, and transition time.
2. Specify the timing assertions (constraints).
   - Define the clock period, waveform, uncertainty, and latency.
   - Specify the input and output port delays.

3. Specify the timing exceptions.
   - Set multicycle paths.
   - Set false paths.
   - Specify minimum and maximum delays, path segmentation, and disabled arcs.
4. Perform the analysis and create reports.
   - Check timing.
   - Generate constraint reports.
   - Generate path timing reports.

Getting Started

This tutorial uses Synopsys tools, including PrimeTime and Design Compiler. You can use PrimeTime with other vendors' tools, but the instructions in this tutorial apply to Synopsys products.

The following describes the minimum configuration required to run PrimeTime:

- Sun SPARCstation 5 running SunOS4 or Solaris, or HP workstation running HP-UX
- 64 MB RAM
- 500 MB swap space
- 300 MB disk space
- Color or monochrome display

Preparing to Run the Tutorial

PrimeTime and Design Compiler must be installed before you can use the tutorial. The tutorial files are installed during the standard installation in the Synopsys root path. Copy the tutorial files from the root path into a working directory, as follows:

1. Create a working PrimeTime directory containing a tutorial directory, then change to the tutorial directory by entering the following commands at the UNIX prompt:

```
mkdir primetime
cd primetime
```

2. Copy the tutorial directory and files from the source tree using the following command:

```
cp -r $SYNOPSYS/doc/pt/tutorial .
```

3. Check the tutorial directory for the following files:

AM2910.db	The design .db for the top-level of the design
CONTROL.db	The design .db for the CONTROL block
REGCNT.db	The design .db for the REGCNT block
UPC.db	The design .db for the UPC block
Y.data	The Stamp data file for the Y block
Y.mod	The Stamp model file for the Y block
Y_lib.db	The library .db for the Y block
STACK_lib.db	The library .db for the STACK block
pt_lib.db	The technology library .db
stack.qtm.pt	The quick timing model script for the stack block

optimize.dcsh	The dc_shell optimization script
timing.dcsh	An example DC shell timing script for translation
tutorial.pt	The complete PrimeTime tutorial script for your reference.

Starting PrimeTime

To start the PrimeTime command-line shell, enter the following command at the UNIX command prompt:

```
% pt_shell
```

The PrimeTime initial display and the PrimeTime prompt appear.

```
pt_shell>
```

If the PrimeTime banner does not appear or you see an error message, check the following:

- PrimeTime was correctly installed.
- The $SYNOPSYS variable was correctly defined in UNIX.
- The Synopsys license server is running and you have a PrimeTime license key available.
- The pt_shell executable is accessible by UNIX. The $SYNOPSYS/sparc/syn/bin directory must be in the UNIX $PATH variable.

Exiting PrimeTime

You can exit PrimeTime any time. To exit PrimeTime, enter quit or exit, or press (Control-d) at the pt_shell> prompt.

The PrimeTime exit text appears, similar to the following:

```
Maximum memory usage for this session: 0.72 MB
CPU usage for this session: 0 seconds

Thank you for using pt_shell!
%
```

Getting Help

To get a basic summary of all the PrimeTime commands, enter

```
pt_shell> help
```

```
Procedures:
 get_unix_variable, ls, printenv, set_unix_variable, sh
Builtins:
 alias, append, array, break, catch, cd, close, concat, continue, echo, eof,
 error, error_info, eval, exec, exit, expr, file, flush, for, foreach, format,
 gets, glob, global, help, history, if, incr, info
```

Use a wildcard to find a command. Enter

```
pt_shell> help *clock
```

```
clock                    # Builtin
create_clock             # Create a clock object
create_generated_clock   # Create a generated clock object
remove_clock             # Remove a clock object
remove_generated_clock   # Remove a generated_clock_object
remove_propagated_clock  # Remove a propagated clock specification
report_clock             # Report clock info
select_clock             # Synonym for 'get_clocks'
```

```
select_generated_clock   # Synonym for 'get_generated_clocks'
set_propagated_clock     # Specify propagated clock latency
```

Use the help -verbose command to get command syntax information. Enter

```
pt_shell> help -verbose set_input_delay
```

```
set_input_delay # Set input delay on ports or pins
        [-clock clock_name]   (Relative clock)
        [-clock_fall]         (Delay is relative to falling edge of clock)
        [-level-sensitive]    (Delay is from level-sensitive latch)
        [-rise]               (Specifies rising delay)
        [-fall]               (Specifies falling delay)
        [-max]                (Specifies maximum delay)
        [-min                 (Specifies minimum delay)
        [-add_delay]          (Don't remove existing input delay)
        delay_value           (Path delay)
        port_pin_list         (List of ports and/or pins)
```

Use the man pages for detailed help for each command and variable. To access the man pages, enter man followed by the name of a command or variable. For example, enter

```
pt_shell> man read_db
```

Paging Output from PrimeTime

By default, PrimeTime output (reports, man pages, and so forth) scrolls continuously from PrimeTime. You can cause PrimeTime to page the output, requiring the spacebar to output each page. To page the PrimeTime output, set this PrimeTime variable:

```
pt_shell> set sh_enable_page_mode true
```

If you want the output to always be paged each time you use PrimeTime, set the sh_enable_page_mode variable in your .synopsys_pt.setup file.

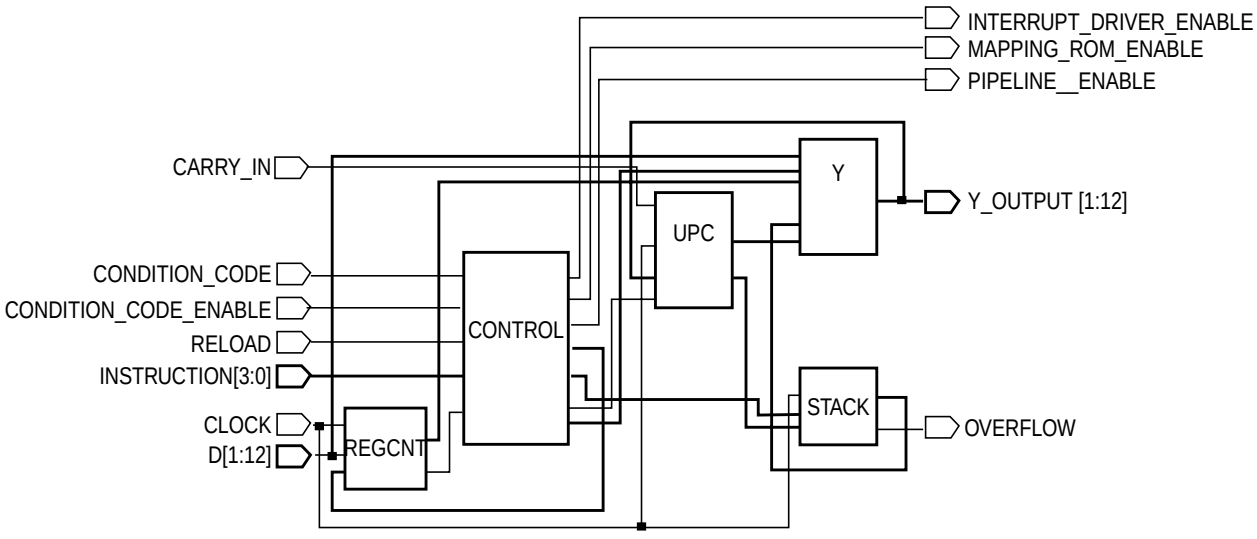
To view all the PrimeTime variables, enter

```
pt_shell> printvar
```

Looking at the Tutorial Design

The tutorial uses a 12-bit sliced AMD 2910 microprocessor. Figure 1-1 shows the block diagram for the design.

Figure 1-1 AMD 2910 Microprocessor



Setting Up the Timing Analysis

Setting up the timing analysis requires that you perform these tasks:

- Define the search path and the link path.
- Read in the design or designs.
- Link the design.

Defining the Search Path and the Link Path

The first step in setting up a timing analysis is to define the search path and the link path. This notifies PrimeTime the files to use and where to use them to perform the link process. The search path and the link path are defined through the PrimeTime variables search_path and link_path.

Set the search_path and the link_path variables. Enter

```
pt_shell> set search_path "."
.
pt_shell> set link_path "* pt_lib.db STACK_lib.db \
                  Y_lib.db"
*pt_lib.db STACK_lib.db Y_lib.db
```

The asterisk (*) in the link_path setting causes PrimeTime to use designs and libraries in memory when it links the design. Almost always, use * as the first item in your link_path variable.

Reading the Design

After defining the search path and the link path, the next step usually is to read in the design or designs. PrimeTime accepts the following design formats:

- Synopsys database files (.db) (Use the read_db command.)
- Verilog netlist files (Use the read_verilog command.)
- Electronic Data Interchange Format (EDIF) netlist files (Use the read_edif command.)
- VHDL netlist files (Use the read_vhdl command.)

These designs must be fully mapped.

Read in the top-level design db for the AM2910 design. Enter

```
pt_shell> read_db AM2910.db
Loading db file'/u/joe/primetime/tutorial/AM2910.db'
1
```

Linking the Design

The next step, linking, creates a fully instantiated design that is ready to be analyzed. The link process resolves the design references and connects the designs and components from the library.

The link process first loads all libraries and designs specified in the link_path variable (one reason to put all technology and model library files in the link_path). Putting the technology and model library files in the link_path also documents the design better.

If a design is referenced but was not explicitly loaded, the linker attempts to load (read in) design.db, where design is the referenced design. This feature is called autoload.

If the linker cannot resolve a reference and the PrimeTime variable link_create_black_boxes is set to “true” (the default), the linker creates a black box for that design.

If the variable link_create_black_boxes is set to false, the design will not link successfully.

The design must be successfully linked for timing analysis to be possible.

Link the AM2910 design. Enter

```
pt_shell> link_design AM2910

Loading db file '/u/joe/primetime/tutorial/pt_lib.db'
Loading db file '/u/joe/primetime/tutorial/STACK_lib.db'
Loading db file '/u/joe/primetime/tutorial/Y_lib.db'
Linking design AM2010 ...
Loading db file '/u/joe/primetime/tutorial/STACK.db'
Loading db file '/u/joe/primetime/tutorial/UPC.db'
Loading db file '/u/joe/primetime/tutorial/REGCNT.db'
Loading db file '/u/joe/primetime/tutorial/Y.db'
Loading db file '/u/joe/primetime/tutorial/CONTROL.db'

Designs used to link AM2910:
  CONTROL, REGCNT, STACK, UPC, Y

Libraries used to link AM2910:
  pt_lib, STACK_lib, Y_lib

Design 'AM2910' was successfully linked
1
```

The technology library pt_lib.db was loaded first because it was specified in the PrimeTime link_path variable.

If you have not read all the subdesign or library files into memory, PrimeTime attempts to load them automatically during linking. This process is known as an autoload, which depends on the values of the search_path and the link_path. The following designs were loaded as a result of the autoload feature:

STACK.db

UPC.db

REGCNT.db

Y.db

CONTROL.db

- If you do not specify the design in the link_design command, PrimeTime links the current design.
- If you do not specify the current design, PrimeTime sets the last design loaded as the current design and links that design.