

Performance Baseline Demo Script

Introduction

This demonstration will use the performance baselining procedure recommended in the UltraFast™ design methodology. The performance baselining procedure enables designers to follow a consistent approach for gaining timing closure.

Preparation:

- Required files: `wave_gen.xpr`, `stage1_impl_design.xpr`, `stage2_impl_design.xpr`, and `stage3_impl_design.xpr`
- Required hardware: None
- Supporting materials: *Performance Baseline Top Level Diagram.pdf* and *wave_gen Design Overview.pdf*

Performance Baselining

Action with Description	Point of Emphasis and Key Takeaway
<ul style="list-style-type: none"> • Launch the Vivado™ Design Suite. 	<ul style="list-style-type: none"> • The Getting Started page provides links to various tasks, including creating a new project and opening an existing project.
<ul style="list-style-type: none"> • Unzip the project using the Tcl Console: <pre>exec unzip \$::env(TRAINING_PATH)/Baselining /demo/wave_gen.zip -d \$::env(TRAINING_PATH)/Baselining /demo/wave_gen</pre> • Open the provided <code>wave_gen.xpr</code> design from the following directory: <code>\$TRAINING_PATH/Baselining/demo/wave_gen</code> 	<ul style="list-style-type: none"> • The <code>wave_gen</code> design is best suited for demonstrating the application of timing constraints and static timing analysis since it needs timing exception constraints.
<ul style="list-style-type: none"> • Open the synthesized design. 	<ul style="list-style-type: none"> • It is recommended that designers use timing constraints as a primary means for gaining timing closure. • The recommendation is that designers apply timing constraints during synthesis so that the Vivado Design Suite can make significant progress toward gaining timing closure.

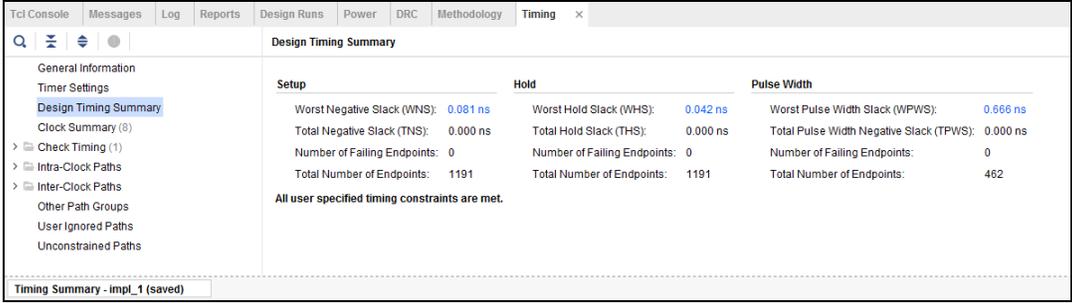
Action with Description	Point of Emphasis and Key Takeaway
<ul style="list-style-type: none"> Open the <i>Performance Baseline Top-Level Diagram.pdf</i> file in the project directory and review the diagrams. The performance baselining process is the recommended approach for gaining timing closure. In the process, designers apply basic timing constraints and add path-specific constraints incrementally while also checking to make sure that the design is constrained properly using the <code>check_timing</code> report. 	<ul style="list-style-type: none"> The block diagram illustrates the entire baselining procedure and highlights the constraints and reports that are frequently used. Except for the possibility of using area constraints to floorplan logic and the possible use of advanced implementation options, this diagram encompasses timing closure completely.
<ul style="list-style-type: none"> Switch to the wave_gen.xpr project that has been opened in the Vivado Design Suite. Open the <i>wave_gen_timing.xdc</i> file and review the timing constraints. 	<p>The timing constraints that have been commented out of the XDC file are:</p> <ul style="list-style-type: none"> <code>create_clock</code> <code>create_generated_clock</code> <code>set_clock_groups -async</code> <code>set_input_delay</code> <code>set_output_delay</code> <code>set_multicycle_path</code> <code>set_false_path</code> constraints
<ul style="list-style-type: none"> Generate the Clock Networks report to see if there are any unconstrained clocks in the design. 	<ul style="list-style-type: none"> The Clock Networks report indicates what FPGA resources have been used to generate and distribute the clocks in the design.

Action with Description	Point of Emphasis and Key Takeaway
	
<ul style="list-style-type: none"> • After noticing that there are unconstrained clocks in the design, uncomment the clock constraints in lines 1 and 2 in the <i>wave_gen_timing.xdc</i> file. • Save the constraints file and reload the design with the new constraints. 	<ul style="list-style-type: none"> • In the Constraints Development stage of the performance baselining procedure, the design needs to be constrained with primary clocks and generated, derived clocks (if any), and finally asynchronous clock groups (if any). • The <i>wave_gen</i> design has three clocks. <i>clk_pin_p</i> is the primary clock. <i>clk_samp</i> and <i>spi_clk</i> are generated clocks. (Do not worry about finding these clocks in the Clock Networks report.)
<ul style="list-style-type: none"> • Rerun the Clock Networks report to verify that the <i>clk_pin_p</i> clock has now been constrained (<i>clk_pin_p</i> should not show up as unconstrained now). 	<ul style="list-style-type: none"> • What resources were used to bring the input clock into the FPGA and generate the desired clocks? • Draw the corresponding circuit or use the Schematic viewer to confirm the answer.
<ul style="list-style-type: none"> • There are two generated clocks in the design that also need to be constrained. • Uncomment lines 3 and 4 in the XDC file. • Save the constraints file and reload the design with the new constraints. 	<ul style="list-style-type: none"> • The clocks are <i>clk_samp</i> and the ODDR output clocks. • The XDC constraints in lines 3 and 4 create generated clocks for <i>clk_samp</i> and ODDR output clocks.

Action with Description	Point of Emphasis and Key Takeaway
<ul style="list-style-type: none"> The performance baselining process also asks you to define each clock domain as asynchronous (this prevents paths between the clock domains from being constrained). Uncomment line 5 in the XDC file, which makes the <code>clk_out1_clk_core</code> and <code>clk_out2_clk_core</code> clocks asynchronous to each other. Save the constraints file and reload the design with the new constraints. 	<ul style="list-style-type: none"> Draw a conceptual representation of each of the clocks on the board and use this to explain how the <code>-async</code> constraints prevent paths between the clocks from being constrained.
<ul style="list-style-type: none"> Generate the Timing Summary report to observe whether the synthesized design has much of a chance of meeting timing before implementing the design. Record the WNS, TNS, and the number of failing endpoints for this timing report. Observe how the timing report can produce such different timing numbers for the same netlist. Optional: Observe an ignored path and how it is denoted in the data path. 	<ul style="list-style-type: none"> Notice that the timing paths that have been analyzed have been ignored and listed in the User Ignored Paths section. Why do we recommend that designers evaluate the WNS before implementing the design (ideally WNS < 300 ps)? What could happen if the WNS is not evaluated?
<ul style="list-style-type: none"> Use the Check Timing section of the Timing Summary report to see the missing timing constraints on I/O ports (i.e., <code>no_input_delay</code>, <code>no_output_delay</code>). 	<ul style="list-style-type: none"> Review the contents of the <code>check_timing</code> report. Go through each of the <code>check_timing</code> report sections.
<ul style="list-style-type: none"> Click the WNS and WHS values in the Setup and Hold sections respectively. The link opens the worst-case timing path. 	<ul style="list-style-type: none"> The <code>wave_gen</code> design contains multicycle paths. These paths are between the <code>cmd_parse</code> and <code>resp_gen</code> blocks in the design. As per the performance baselining process, you will apply the timing exceptions constraints, such as the multicycle, false path, and max delay constraints, at the very last stage. For now, you can ignore these multicycle paths, which will be fixed later.

Action with Description	Point of Emphasis and Key Takeaway
<ul style="list-style-type: none"> Unzip the project using the Tcl Console: <pre>exec unzip \$::env(TRAINING_PATH)/Baselining /demo/Stage1_impl_design.zip -d \$::env(TRAINING_PATH)/Baselining /demo/Stage1_impl_design</pre> Open the provided stage1_impl_design.xpr project located in the <code>\$TRAINING_PATH/Baselining/demo/Stage1_impl_design</code> directory. Close the wave_gen.xpr design if prompted. 	<ul style="list-style-type: none"> This is the implemented design with complete clock constraints and asynchronous constraints. These are the complete baseline clock constraints.
<ul style="list-style-type: none"> Open the implemented design. Click OK in the dialog box if any critical warnings appear. 	
<ul style="list-style-type: none"> Review the Timing Summary report to see whether the design met the timing objectives. 	<ul style="list-style-type: none"> How many ways can you confirm that the timing constraints were or were not met?
<ul style="list-style-type: none"> Click the WNS value in the Setup section. The link opens the worst-case timing path. 	<ul style="list-style-type: none"> What logic was used to make this data path?
<ul style="list-style-type: none"> Uncomment lines 7 to 18 in the <i>wave_gen_timing.xdc</i> file, which applies I/O timing constraints to the design. Save the constraints file and reload the design with the new constraints. <ul style="list-style-type: none"> Right-click Implementation in the Flow Navigator and select Reload to reload the implemented design. Next, you would normally need to implement the design with the new constraints added to the existing constraints. However, to save time the implemented design with clock and I/O constraints has been provided. 	<ul style="list-style-type: none"> This forms the stage 2 timing constraints of the performance baseline procedure.

Action with Description	Point of Emphasis and Key Takeaway
<ul style="list-style-type: none"> Unzip the project using the Tcl Console: <pre>exec unzip \$::env(TRAINING_PATH)/Baselining /demo/Stage2_impl_design.zip -d \$::env(TRAINING_PATH)/Baselining /demo/Stage2_impl_design</pre> Open the provided stage2_impl_design.xpr project located in the <code>Stage2_impl_design</code> directory under the project directory. Close the stage1_impl_design.xpr design if prompted. 	<ul style="list-style-type: none"> This is the implemented design with baseline constraints plus I/O timing constraints.
<ul style="list-style-type: none"> Open the implemented design. 	<ul style="list-style-type: none"> Ignore any warnings and critical messages
<ul style="list-style-type: none"> Generate the Timing Summary report to see if the design met the timing objectives. Record the WNS, TNS, and the number of failing endpoints for this timing report. 	<ul style="list-style-type: none"> What do you think was the cause of the design failing to meet timing?
<ul style="list-style-type: none"> Uncomment lines 20 to 22 in the <i>wave_gen_timing.xdc</i> file, which applies timing exceptions, such as multicycle, false paths, and max_delay constraints, to the design. Save the constraints file and reload the design with the new constraints. Next, you need to implement the design with the new constraints. In order to save time the implemented design has already been provided. 	<ul style="list-style-type: none"> This forms the stage 3 timing constraints of the performance baselining procedure. The timing exception constraints are dependent on the design. Some designs do not need timing exception constraints. If the clocks are NOT asynchronous in the actual design, then comment out the <code>set_clock_groups -async</code> constraint. However, in the wave_gen design clocks are asynchronous so the <code>set_clock_groups</code> constraint should be applied.
<ul style="list-style-type: none"> Unzip the project using the Tcl Console: <pre>exec unzip \$::env(TRAINING_PATH)/Baselining /demo/Stage3_impl_design.zip -d \$::env(TRAINING_PATH)/Baselining /demo/Stage3_impl_design</pre> Open the provided stage3_impl_design.xpr project located in the <code>Stage3_impl_design</code> directory under the project directory. Close the stage2_impl_design.xpr design if prompted. 	<p>This is the implemented design with baseline constraints, plus I/O timing constraints, plus timing exceptions.</p>

Action with Description	Point of Emphasis and Key Takeaway
<ul style="list-style-type: none"> Open the implemented design. Review the Timing Summary report that is automatically generated after opening the implemented design to see if the design met the timing objectives. 	<p>The Timing Summary report after implementation is often considered as a sign-off criterion for meeting the timing objectives of the design.</p> <p>Were the implementation tools able to meet the design's timing constraints?</p>
<div style="border: 1px solid black; padding: 5px;">  <p>Note: The timing numbers may vary depending on the version of the Vivado Design Suite and the OS.</p> </div>	
<ul style="list-style-type: none"> Review the Check Timing section of the Timing Summary report to see the missing constraints on I/O ports, clock objects, etc. 	<p>It is recommended that there are no unconstrained internal endpoints in the design.</p> <p>In this case, there is one output that is not constrained with an output constraint but has a clock definition. This is okay because the output port will be used as a clock for downstream devices.</p>
<ul style="list-style-type: none"> Close the Vivado Design Suite. 	

Summary

In this demonstration, you followed the performance baselining process to make certain that the *wave_gen* design was constrained properly.

References:

- Supporting materials
 - UltraFast Design Methodology Guide for FPGAs and SoCs (UG949)*