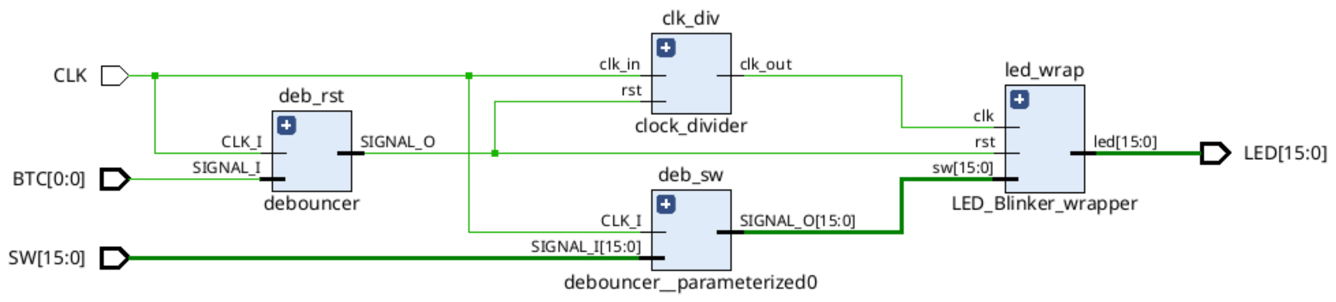
Lab 17: Timing Constraints

Goal of this lab is to learn using timing constraints in Vivado by:

- Using the Timing Constraint wizard to a simple HDL design
- Observe the effect of constraints
- Observe the effect of over constraining

The design

The example design we are using is schematized in the following screenshot.



The system enables the blinking of the LEDs, when the corresponding switches are connected. The LED blinking is controlled by a counter module `LED_Blinker`, which uses clock derived by the system clock.

The derived clock frequency is 20 MHz, one fifth of the system input clock of 100 MHz.

Exercise 1. Timing Constraints

Go to `~/labs/lab17/` and open the `lab17.xpr` Vivado project.

```
vivado lab17.xpr &
```

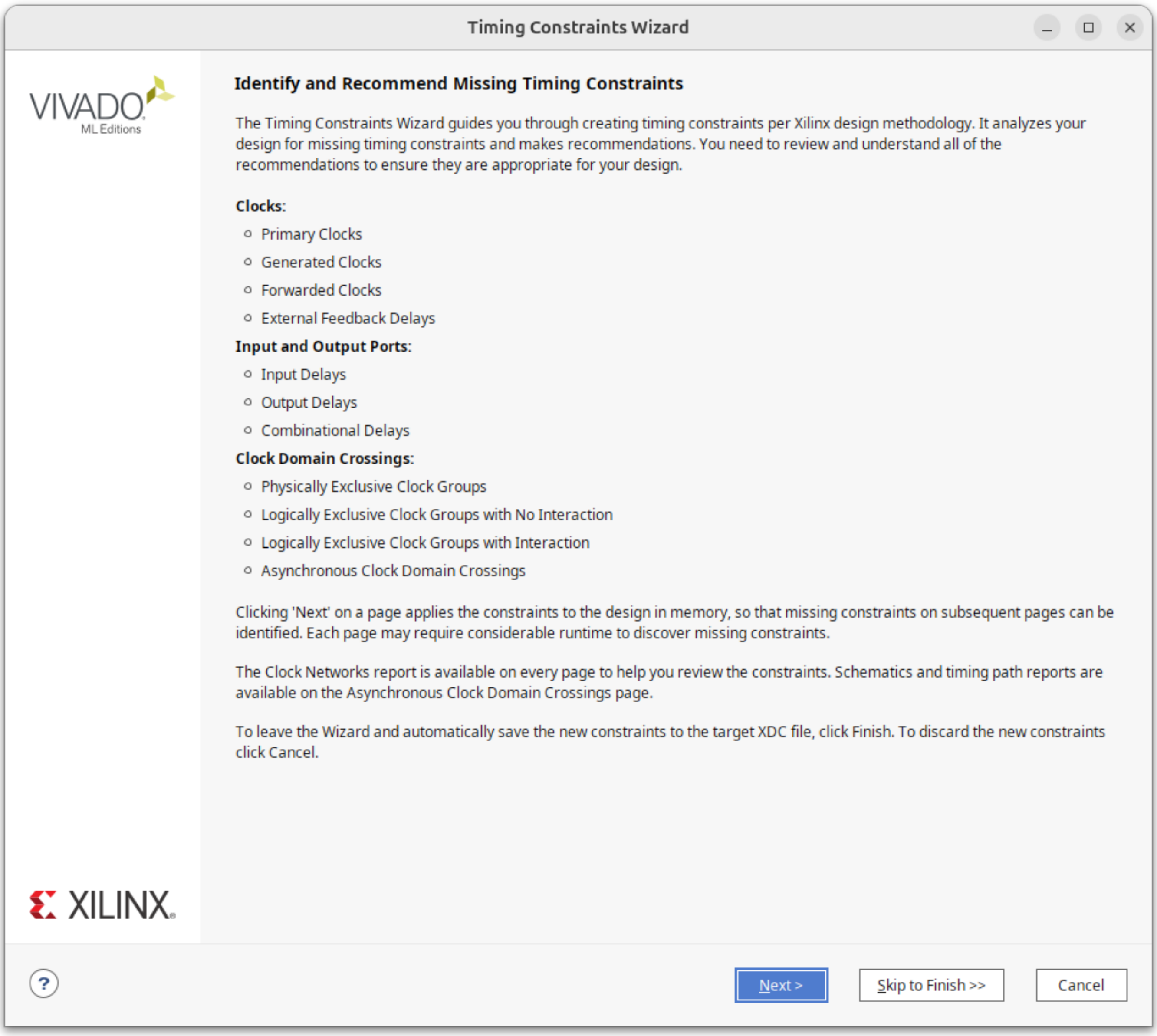
The current constraints applies only the IOs and to the primary clock, as we did in previous labs. You can have a look at the constraint file `Basys3_Master.xdc`.

Run the synthesis, and open the synthesized design once finished.

Now click on Constraints Wizard on the left sidebar.

You should be prompted with a Warning, telling you that we don't have a target constraint file. Click on *Define Target* and select the `Basys3_Master.xdc` as the target and click OK.

Click now on *Constraint Wizard* again.



Identify and Recommend Missing Timing Constraints

The Timing Constraints Wizard guides you through creating timing constraints per Xilinx design methodology. It analyzes your design for missing timing constraints and makes recommendations. You need to review and understand all of the recommendations to ensure they are appropriate for your design.

Clocks:
- Primary Clocks
- Generated Clocks
- Forwarded Clocks
- External Feedback Delays

Input and Output Ports:
- Input Delays
- Output Delays
- Combinational Delays

Clock Domain Crossings:
- Physically Exclusive Clock Groups
- Logically Exclusive Clock Groups with No Interaction
- Logically Exclusive Clock Groups with Interaction
- Asynchronous Clock Domain Crossings

Clicking 'Next' on a page applies the constraints to the design in memory, so that missing constraints on subsequent pages can be identified. Each page may require considerable runtime to discover missing constraints.

The Clock Networks report is available on every page to help you review the constraints. Schematics and timing path reports are available on the Asynchronous Clock Domain Crossings page.

To leave the Wizard and automatically save the new constraints to the target XDC file, click Finish. To discard the new constraints click Cancel.

The wizard will guide you through a series of checks and suggest specific constraints. Click Next.

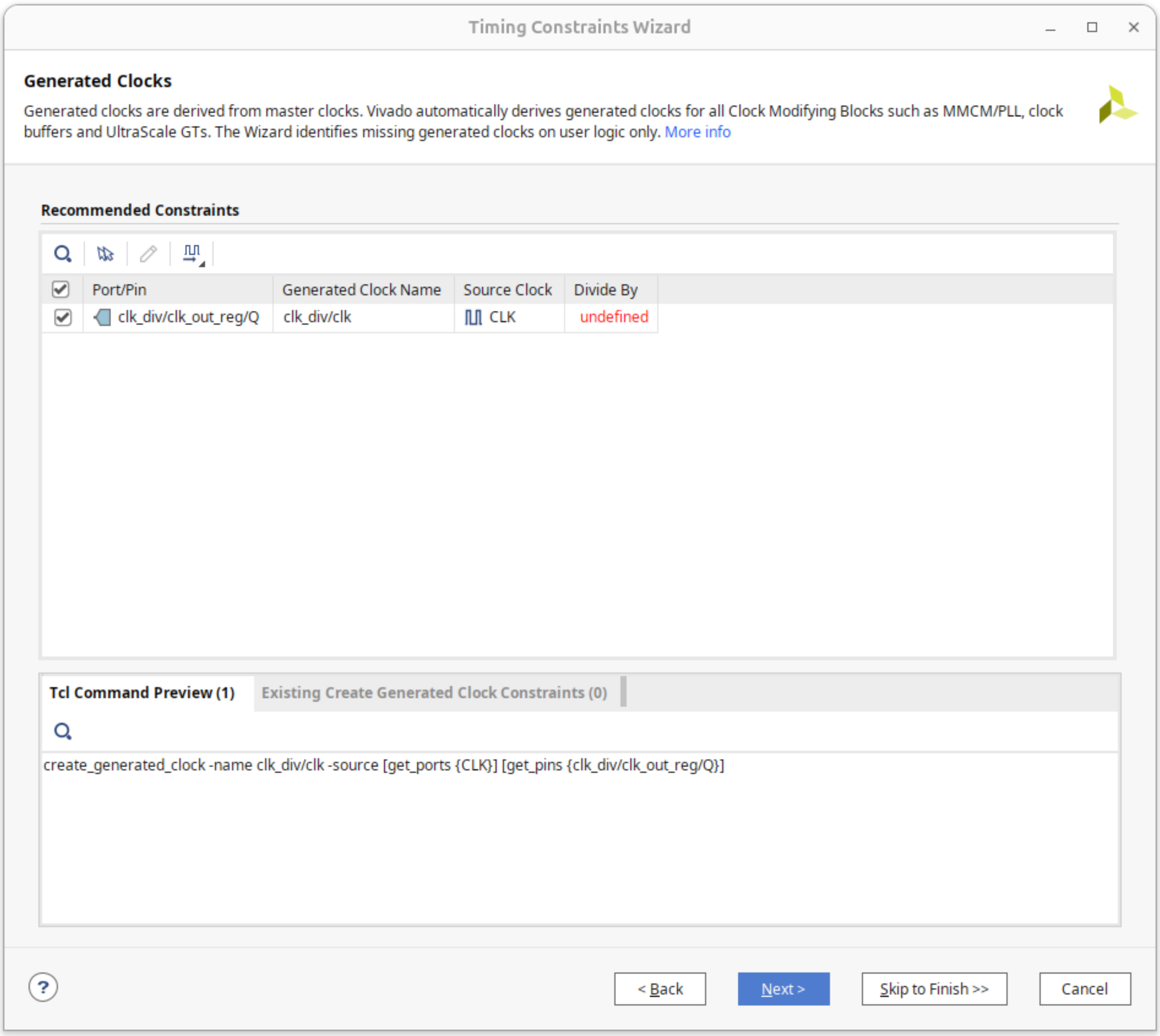
In the *Primary Clock* page, there are no recommendations, meaning that our primary clock has been already constrained. You can double check the existing constraint, by opening the *Existing Create Clock Constraint* tab, in the bottom box. It should show this:

```
create_clock -period 10.000 -name sys_clk_pin -waveform {0.000 5.000} -add
[get_ports CLK]
```

Click Next.

We are now in *Generated Clock* page. Vivado recognised that the derived clock is not constrained in our file. In particular, we are missing the divide factor. Click on the red text, and type 5.

You can also see the command, that will be written in the constraint file in the bottom box. Click Next.



Generated Clocks

Generated clocks are derived from master clocks. Vivado automatically derives generated clocks for all Clock Modifying Blocks such as MMCM/PLL, clock buffers and UltraScale GTs. The Wizard identifies missing generated clocks on user logic only. More info

Recommended Constraints

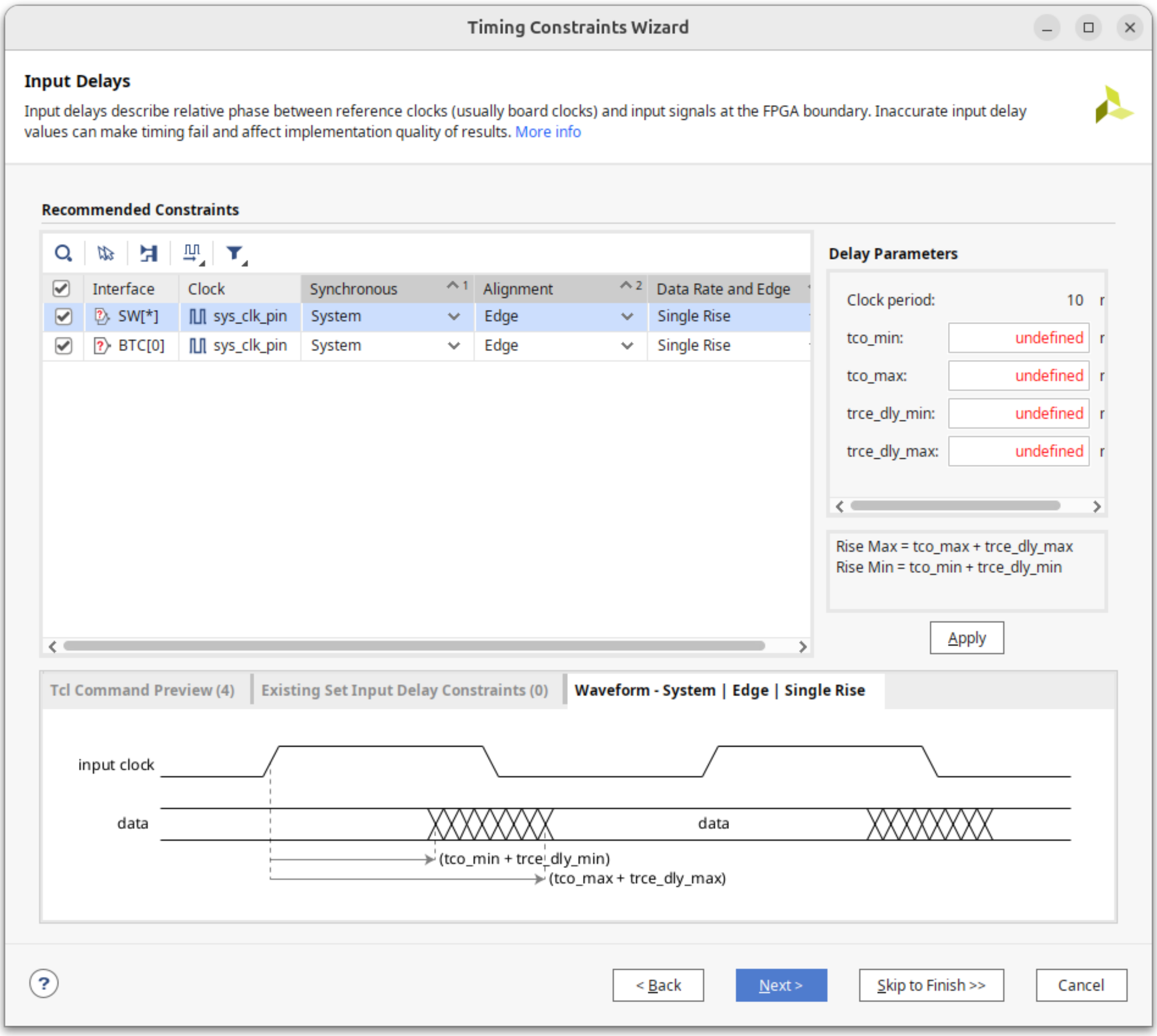
✓	Port/Pin	Generated Clock Name	Source Clock	Divide By
✓	clk_div/clk_out_reg/Q	clk_div/clk	CLK	undefined

Tcl Command Preview (1) | Existing Create Generated Clock Constraints (0)

```
create_generated_clock -name clk_div/clk -source [get_ports {CLK}] [get_pins {clk_div/clk_out_reg/Q}]
```

Continue until you reach the *Input Delays* page. This is interesting, but we don't need to specify any input constraints for our input data, since they are asynchronous to our clock.

Untick the box, at the top of the list and click Next.


Timing Constraints Wizard
Input Delays
Input delays describe relative phase between reference clocks (usually board clocks) and input signals at the FPGA boundary. Inaccurate input delay values can make timing fail and affect implementation quality of results. More info
Recommended Constraints
Delay Parameters
Tcl Command Preview (4)
Existing Set Input Delay Constraints (0)
Waveform - System | Edge | Single Rise
input clock
data
data
(tco_min + trce_dly_min)
(tco_max + trce_dly_max)
< Back
Next >
Skip to Finish >>
Cancel
Apply

✓	Interface	Clock	Synchronous	Alignment	Data Rate and Edge
✓	SW[*]	sys_clk_pin	System	Edge	Single Rise
✓	BTC[0]	sys_clk_pin	System	Edge	Single Rise

Clock period:	10
tco_min:	undefined
tco_max:	undefined
trce_dly_min:	undefined
trce_dly_max:	undefined

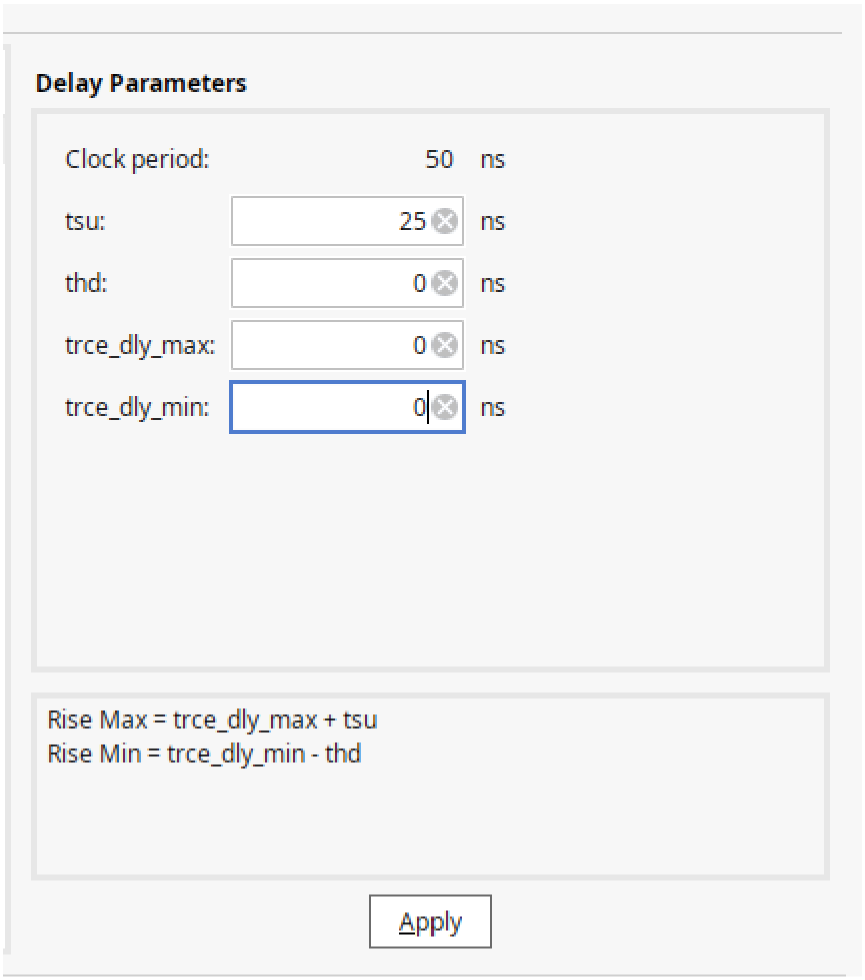
Rise Max = tco_max + trce_dly_max
Rise Min = tco_min + trce_dly_min

We are now in the *Output Delay* page. Here we want to specify the constraints. All our LEDs are synchronous to the same derived clock. A constraint on a 50ns period clock should be reached easily.

Vivado splits the the minimum and maximum delays here, destination setup (on max), destination hold (on min) and propagation(trace) delays on both. Even if you can measure these values separately, Vivado will not. So you can put all the max delay as setup time or max trace delay, and the result will be the same.

For this exercise, we'll set the maximum delay to half-period. Set the values as shown below, and click Apply. This is not really needed for our output, since the LEDs also work asynchronously.

Remember, the input/output delay constraints are used by Vivado only to estimate the timing of your design. Here, we are requesting that the output data should arrive with a maximum delay w.r.t. the clock of half a period.



Delay Parameters

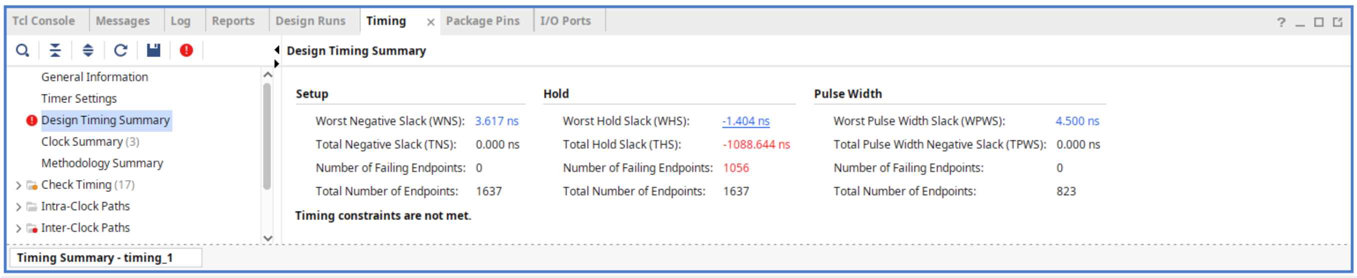
Clock period:	50	ns
tsu:	25	ns
thd:	0	ns
trce_dly_max:	0	ns
trce_dly_min:	0	ns

Rise Max = trce_dly_max + tsu
Rise Min = trce_dly_min - thd

Click now on *Skip to Finish*. You will see a summary page. Click on Finish.

Open again `Basys3_Master.xdc` and have a look at the changes.

Click on *Report Timing Summary*, to perform the timing analysis. Keep the default in the pop-up window, and click OK.

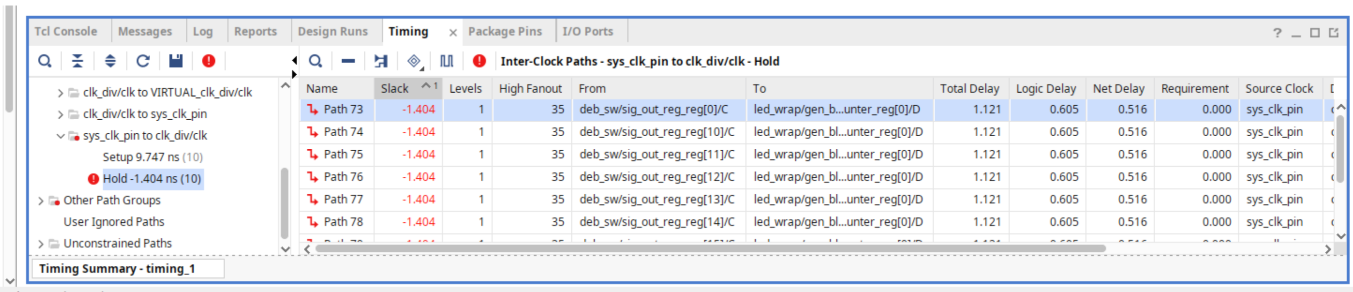


Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	3.617 ns	Worst Hold Slack (WHS):	-1.404 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	-1088.644 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	1056	Number of Failing Endpoints:	0
Total Number of Endpoints:	1637	Total Number of Endpoints:	1637	Total Number of Endpoints:	823

Timing constraints are not met.

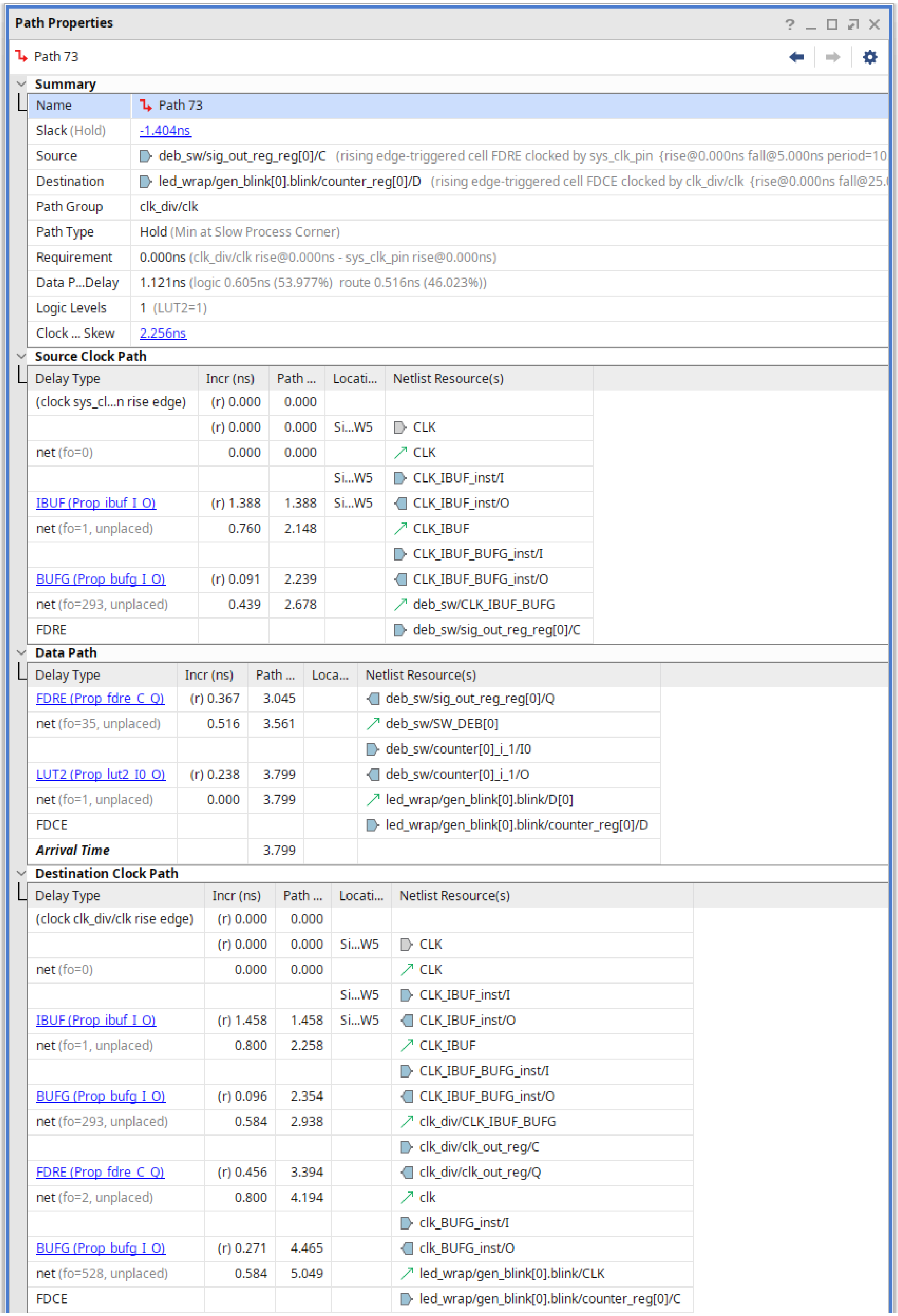
It seems that we have some interclock path violations. If you click on blue link, next to *Worst Hold Slack (WHS)*, you should see a list of path violating the timing.



Inter-Clock Paths - sys_clk_pin to clk_div/clk - Hold

Name	Slack	Levels	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock
Path 73	-1.404	1	35	deb_sw/sig_out_reg_reg[0]/C	led_wrap/gen_b...unter_reg[0]/D	1.121	0.605	0.516	0.000	sys_clk_pin
Path 74	-1.404	1	35	deb_sw/sig_out_reg_reg[10]/C	led_wrap/gen_bl...unter_reg[0]/D	1.121	0.605	0.516	0.000	sys_clk_pin
Path 75	-1.404	1	35	deb_sw/sig_out_reg_reg[11]/C	led_wrap/gen_bl...unter_reg[0]/D	1.121	0.605	0.516	0.000	sys_clk_pin
Path 76	-1.404	1	35	deb_sw/sig_out_reg_reg[12]/C	led_wrap/gen_bl...unter_reg[0]/D	1.121	0.605	0.516	0.000	sys_clk_pin
Path 77	-1.404	1	35	deb_sw/sig_out_reg_reg[13]/C	led_wrap/gen_bl...unter_reg[0]/D	1.121	0.605	0.516	0.000	sys_clk_pin
Path 78	-1.404	1	35	deb_sw/sig_out_reg_reg[14]/C	led_wrap/gen_bl...unter_reg[0]/D	1.121	0.605	0.516	0.000	sys_clk_pin

You can double click on one of the path to see the full timing report.



Path Properties

Path 73

Summary

Name	Path 73
Slack (Hold)	-1.404ns
Source	deb_sw/sig_out_reg_reg[0]/C (rising edge-triggered cell FDRE clocked by sys_clk_pin {rise@0.000ns fall@5.000ns period=10
Destination	led_wrap/gen_blink[0].blink/counter_reg[0]/D (rising edge-triggered cell FDCE clocked by clk_div/clk {rise@0.000ns fall@25.
Path Group	clk_div/clk
Path Type	Hold (Min at Slow Process Corner)
Requirement	0.000ns (clk_div/clk rise@0.000ns - sys_clk_pin rise@0.000ns)
Data P...Delay	1.121ns (logic 0.605ns (53.977%) route 0.516ns (46.023%))
Logic Levels	1 (LUT2=1)
Clock ... Skew	2.256ns

Source Clock Path

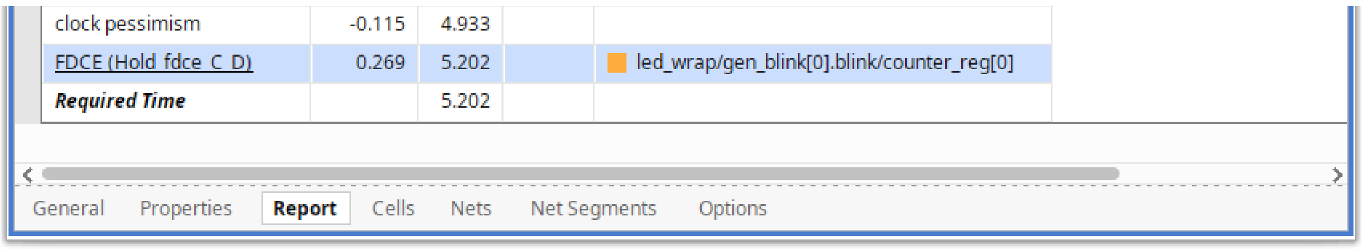
Delay Type	Incr (ns)	Path ...	Locati...	Netlist Resource(s)
(clock sys_cl...n rise edge)	(r) 0.000	0.000		
	(r) 0.000	0.000	Si...W5	CLK
net (fo=0)	0.000	0.000		CLK
			Si...W5	CLK_IBUF_inst/I
IBUF (Prop ibuf I O)	(r) 1.388	1.388	Si...W5	CLK_IBUF_inst/O
net (fo=1, unplaced)	0.760	2.148		CLK_IBUF
				CLK_IBUF_BUFG_inst/I
BUFG (Prop bufg I O)	(r) 0.091	2.239		CLK_IBUF_BUFG_inst/O
net (fo=293, unplaced)	0.439	2.678		deb_sw/CLK_IBUF_BUFG
FDRE				deb_sw/sig_out_reg_reg[0]/C

Data Path

Delay Type	Incr (ns)	Path ...	Loca...	Netlist Resource(s)
FDRE (Prop fdre C Q)	(r) 0.367	3.045		deb_sw/sig_out_reg_reg[0]/Q
net (fo=35, unplaced)	0.516	3.561		deb_sw/SW_DEB[0]
				deb_sw/counter[0]_i_1/I0
LUT2 (Prop lut2 I0 O)	(r) 0.238	3.799		deb_sw/counter[0]_i_1/O
net (fo=1, unplaced)	0.000	3.799		led_wrap/gen_blink[0].blink/D[0]
FDCE				led_wrap/gen_blink[0].blink/counter_reg[0]/D
Arrival Time		3.799		

Destination Clock Path

Delay Type	Incr (ns)	Path ...	Locati...	Netlist Resource(s)
(clock clk_div/clk rise edge)	(r) 0.000	0.000		
	(r) 0.000	0.000	Si...W5	CLK
net (fo=0)	0.000	0.000		CLK
			Si...W5	CLK_IBUF_inst/I
IBUF (Prop ibuf I O)	(r) 1.458	1.458	Si...W5	CLK_IBUF_inst/O
net (fo=1, unplaced)	0.800	2.258		CLK_IBUF
				CLK_IBUF_BUFG_inst/I
BUFG (Prop bufg I O)	(r) 0.096	2.354		CLK_IBUF_BUFG_inst/O
net (fo=293, unplaced)	0.584	2.938		clk_div/CLK_IBUF_BUFG
				clk_div/clk_out_reg/C
FDRE (Prop fdre C Q)	(r) 0.456	3.394		clk_div/clk_out_reg/Q
net (fo=2, unplaced)	0.800	4.194		clk
				clk_BUFG_inst/I
BUFG (Prop bufg I O)	(r) 0.271	4.465		clk_BUFG_inst/O
net (fo=528, unplaced)	0.584	5.049		led_wrap/gen_blink[0].blink/CLK
FDCE				led_wrap/gen_blink[0].blink/counter_reg[0]/C



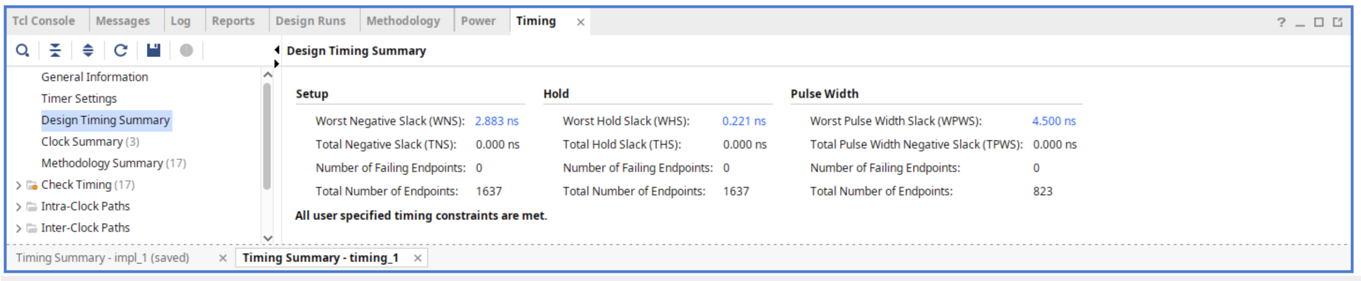
clock pessimism	-0.115	4.933		
FDCE (Hold_fdce_C_D)	0.269	5.202		🟧 led_wrap/gen_blink[0].blink/counter_reg[0]
Required Time		5.202		

Now let's remember that we are looking at the timing report of the synthesized design. This means that the blocks have not yet being placed on the actual logic on the FPGA, and the timing is just an estimation.

Since we modified our constraints, we have to re-run the synthesis. Click on *Run Synthesis* again.

Re-open the timing report, once it's finished. It should still fail the timing.

Run now the implementation, and open the implemented design at the end. Click on *Report Timing Summary* for the implemented design. The report should now tell you that the design meets timing.

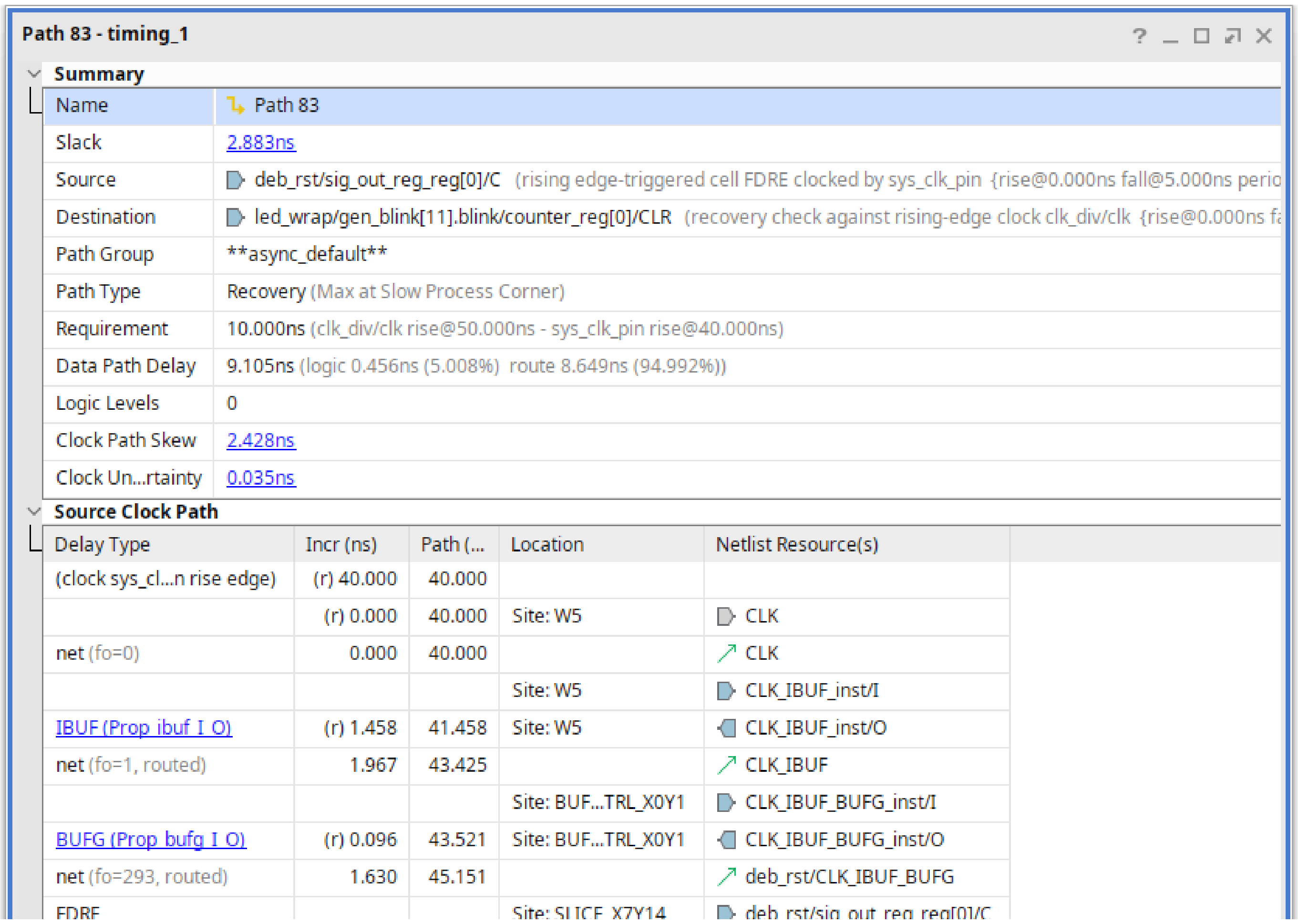


Design Timing Summary

Setup		**Hold**		**Pulse Width**	
Worst Negative Slack (WNS):	2.883 ns	Worst Hold Slack (WHS):	0.221 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	1637	Total Number of Endpoints:	1637	Total Number of Endpoints:	823

All user specified timing constraints are met.

You can also open a report for one of the implemented paths. This is now different from the previous one in synthesis. Namely, we can see here the actual location of the implemented blocks and nets.



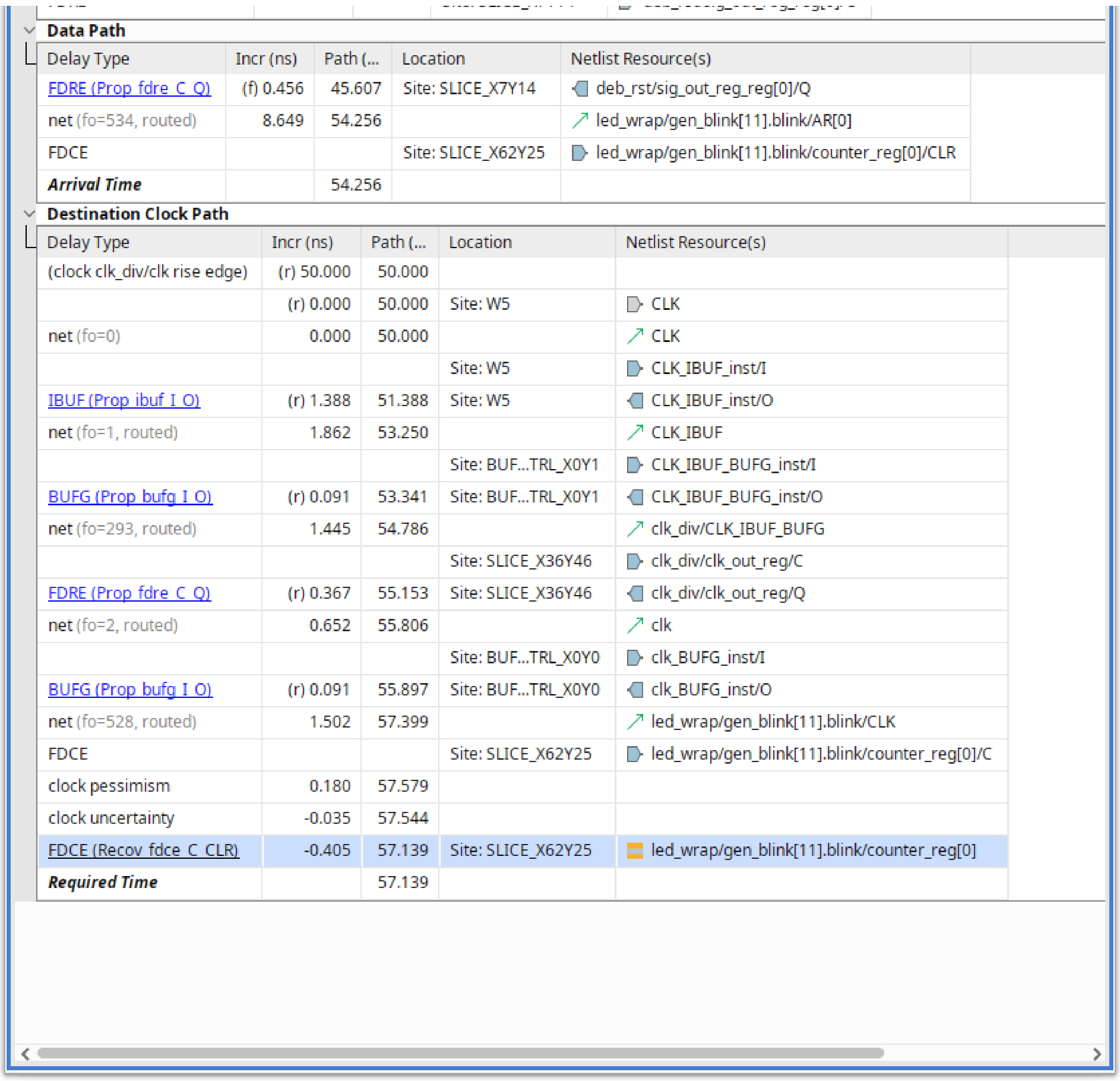
Path 83 - timing_1

Summary

Name	Path 83
Slack	2.883ns
Source	deb_rst/sig_out_reg_reg[0]/C (rising edge-triggered cell FDRE clocked by sys_clk_pin {rise@0.000ns fall@5.000ns perio
Destination	led_wrap/gen_blink[11].blink/counter_reg[0]/CLR (recovery check against rising-edge clock clk_div/clk {rise@0.000ns fa
Path Group	**async_default**
Path Type	Recovery (Max at Slow Process Corner)
Requirement	10.000ns (clk_div/clk rise@50.000ns - sys_clk_pin rise@40.000ns)
Data Path Delay	9.105ns (logic 0.456ns (5.008%) route 8.649ns (94.992%))
Logic Levels	0
Clock Path Skew	2.428ns
Clock Un...rtainty	0.035ns

Source Clock Path

Delay Type	Incr (ns)	Path (...	Location	Netlist Resource(s)
(clock sys_cl...n rise edge)	(r) 40.000	40.000		
	(r) 0.000	40.000	Site: W5	CLK
net (fo=0)	0.000	40.000		CLK
			Site: W5	CLK_IBUF_inst/I
IBUF (Prop_ibuf_I_O)	(r) 1.458	41.458	Site: W5	CLK_IBUF_inst/O
net (fo=1, routed)	1.967	43.425		CLK_IBUF
			Site: BUF...TRL_X0Y1	CLK_IBUF_BUFG_inst/I
BUFG (Prop_bufg_I_O)	(r) 0.096	43.521	Site: BUF...TRL_X0Y1	CLK_IBUF_BUFG_inst/O
net (fo=293, routed)	1.630	45.151		deb_rst/CLK_IBUF_BUFG
FDRE			Site: SLICE_X7Y14	deb_rst/sig_out_reg_reg[0]/C



Data Path

Delay Type	Incr (ns)	Path (...	Location	Netlist Resource(s)
FDRE (Prop fdre C Q)	(f) 0.456	45.607	Site: SLICE_X7Y14	deb_rst/sig_out_reg_reg[0]/Q
net (fo=534, routed)	8.649	54.256		led_wrap/gen_blink[11].blink/AR[0]
FDCE			Site: SLICE_X62Y25	led_wrap/gen_blink[11].blink/counter_reg[0]/CLR
Arrival Time		54.256		

Destination Clock Path

Delay Type	Incr (ns)	Path (...	Location	Netlist Resource(s)
(clock clk_div/clk rise edge)	(r) 50.000	50.000		
	(r) 0.000	50.000	Site: W5	CLK
net (fo=0)	0.000	50.000		CLK
			Site: W5	CLK_IBUF_inst/I
IBUF (Prop ibuf I O)	(r) 1.388	51.388	Site: W5	CLK_IBUF_inst/O
net (fo=1, routed)	1.862	53.250		CLK_IBUF
			Site: BUF...TRL_X0Y1	CLK_IBUF_BUFG_inst/I
BUFG (Prop bufg I O)	(r) 0.091	53.341	Site: BUF...TRL_X0Y1	CLK_IBUF_BUFG_inst/O
net (fo=293, routed)	1.445	54.786		clk_div/CLK_IBUF_BUFG
			Site: SLICE_X36Y46	clk_div/clk_out_reg/C
FDRE (Prop fdre C Q)	(r) 0.367	55.153	Site: SLICE_X36Y46	clk_div/clk_out_reg/Q
net (fo=2, routed)	0.652	55.806		clk
			Site: BUF...TRL_X0Y0	clk_BUFG_inst/I
BUFG (Prop bufg I O)	(r) 0.091	55.897	Site: BUF...TRL_X0Y0	clk_BUFG_inst/O
net (fo=528, routed)	1.502	57.399		led_wrap/gen_blink[11].blink/CLK
FDCE			Site: SLICE_X62Y25	led_wrap/gen_blink[11].blink/counter_reg[0]/C
clock pessimism	0.180	57.579		
clock uncertainty	-0.035	57.544		
FDCE (Recov fdce C CLR)	-0.405	57.139	Site: SLICE_X62Y25	led_wrap/gen_blink[11].blink/counter_reg[0]
Required Time		57.139		

Since our design meets timing, you can generate the bitstream and load it to the board. You should see the LEDs blinking at different rates, if their corresponding switch is high.

Exercise 2. Making timing fail

Increase the output delay

Now we over constrain our design, to artificially fail the timing.

Open again the *Synthesized Design* and click on *Edit Timing Constraint*. Click on *Set Output Delay* on the left menu. Double click on the *max* delay path and change the max delay to 40, in the opened window. Click OK to close it, and then click *Apply* at the bottom of the box. Click now on the save icon on the top left of the window.

Run again the synthesis and implementation. You can see that now our timing failed. However, the design didn't change. To double-check, create the bitstream and load it to the board.

It should still work.

Increase the clock frequency

We want to generate now a not working design. The Basys3 has a system clock of 100 MHz, but the Artix-7 FPGA can run much faster than that.

Close the implemented design and open the `Basys3_Master.xdc` file. Select the output delays at the bottom and comment them out (`CTRL-/`).

Let's now double the clock speed. Go at the top of the file. And change the `sys_clk`, to run at `400 MHz`. You need to calculate the new period. The clock should still be half-duty.

Now find the line corresponding to the generated clock, it should be at the bottom of the file. We want our generated clock still to run at 20 MHz. Adjust the divide factor accordingly.

Save and run the implementation. Open the timing report. The timing should now fail.

However, since the actual system clock on the board is 100 MHz, if you try to load this design on the board it should still work. With the difference that the generated clock will be now much slower than before, having increased the division factor.