Timing Analysis Overview

CSULB
CECS 460

https://www.xilinx.com/support/documentation/sw_manuals/xilinx11/ise_c_timing_analysis_overview.htm

Introduction

- After utilizing the tools to design and verify your digital circuit you can use the Timing Analyzer to perform a detailed analysis of your FPGA design

- This will confirm that the specified timing constraints were properly passed to the implementation tools and the following requirements have been met

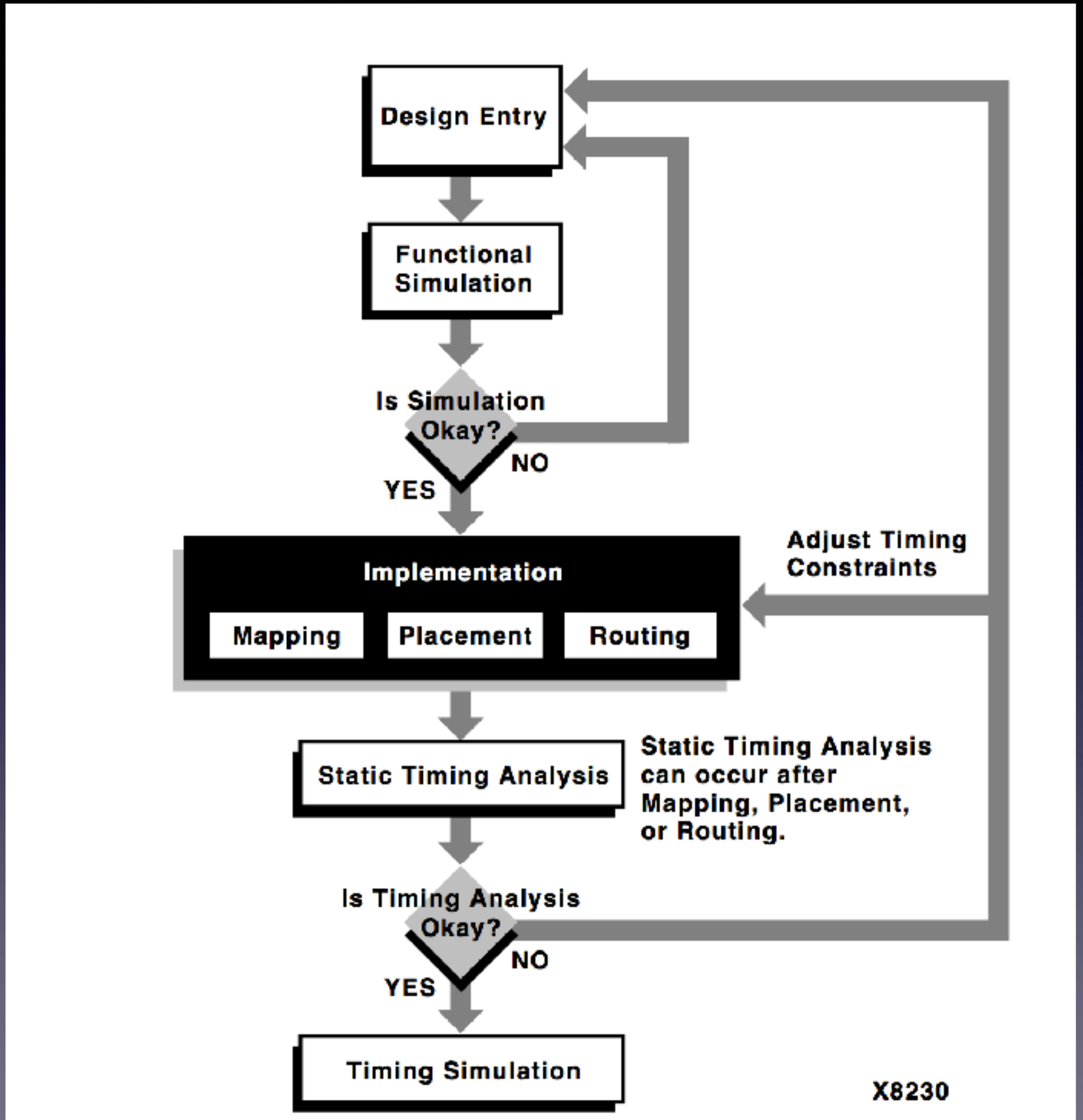
- Verify that timing requirements were met for all paths in your design

- Analyze setup and hold performance for all constrained paths in the design

- Verify that operational frequencies are within component performance limits

- Analyze unconstrained paths to determine if any critical timing paths have been left unconstrained

Design Flow


Design Entry
Functional Simulation
Is Simulation Okay?
NO
YES
Implementation
Mapping
Placement
Routing
Adjust Timing Constraints
Static Timing Analysis
Static Timing Analysis can occur after Mapping, Placement, or Routing.
Is Timing Analysis Okay?
NO
YES
Timing Simulation
X8230

Methodology

- To efficiently analyze timing, a top-down method is recommended, which begins with an inspection of the overall performance of the design, followed by an inspection of the different categories of constraints, a single constraint, and finally a specific path in the design

- After analysis is complete, a detailed report of that analysis is created, which can be customized to include only the information you need.

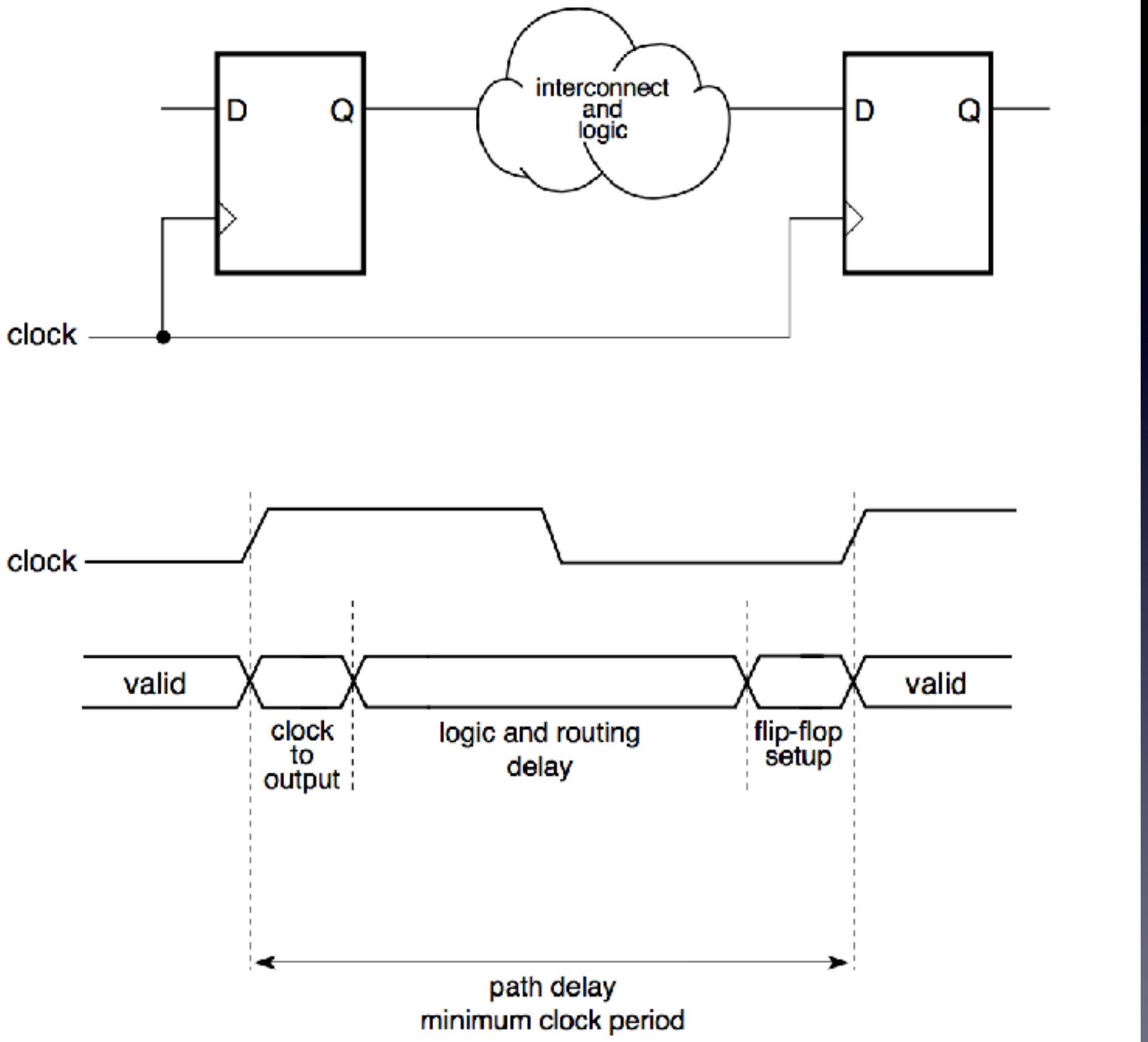
Basic Path Types

- After implementing the design use the Timing Analyzer to calculate the design's system performance which is limited by four basic types of timing paths
  - Clock to Setup
  - Clock to Pad
  - Pad to Pad
  - Pad to Setup

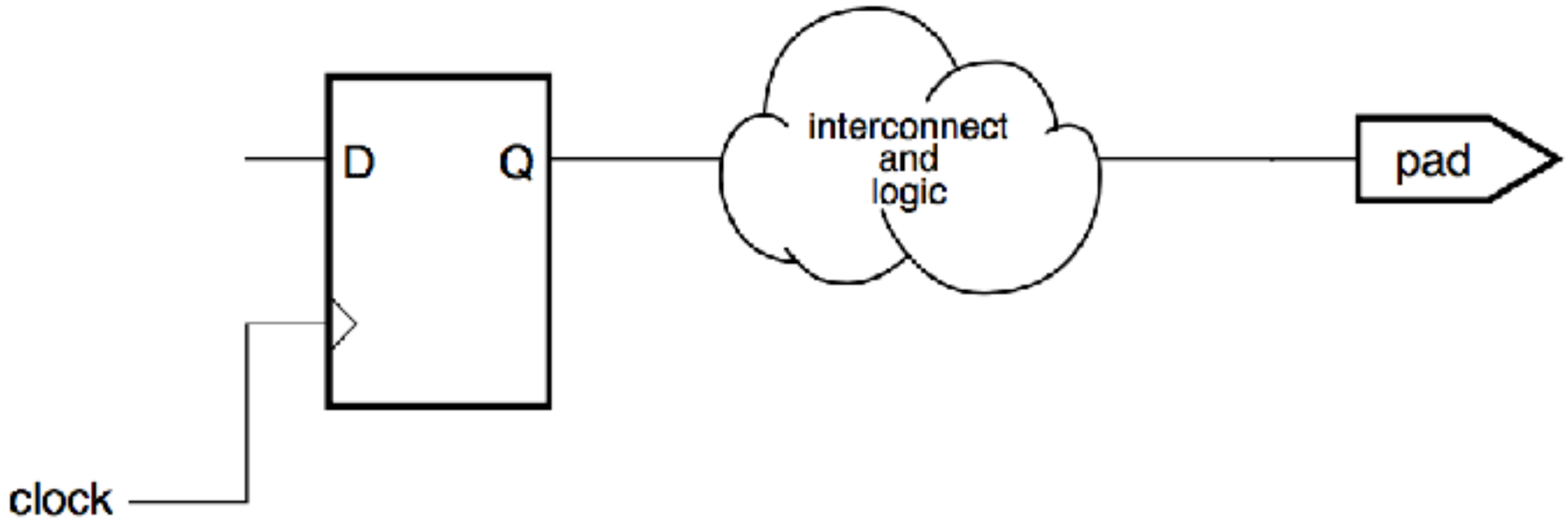
Timing Analyzer Steps

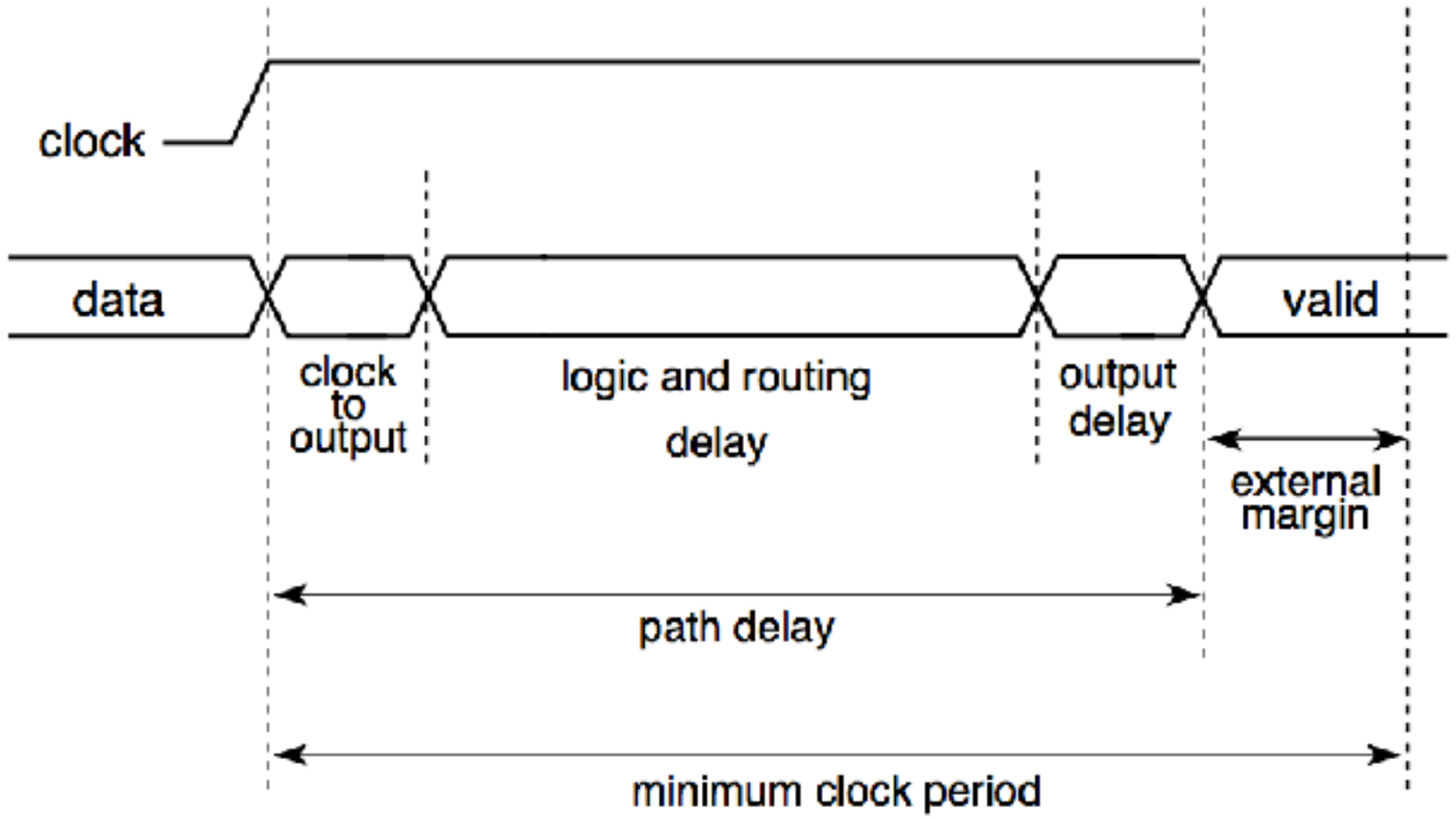
- Analyzing Input Timing
- Analyzing Synchronous Element Timing
- Analyzing Output Timing
- Analyzing Timing Exceptions
- Analyzing Unconstrained Paths
- Cross Probing from Timing Analyzer

Clock to Setup

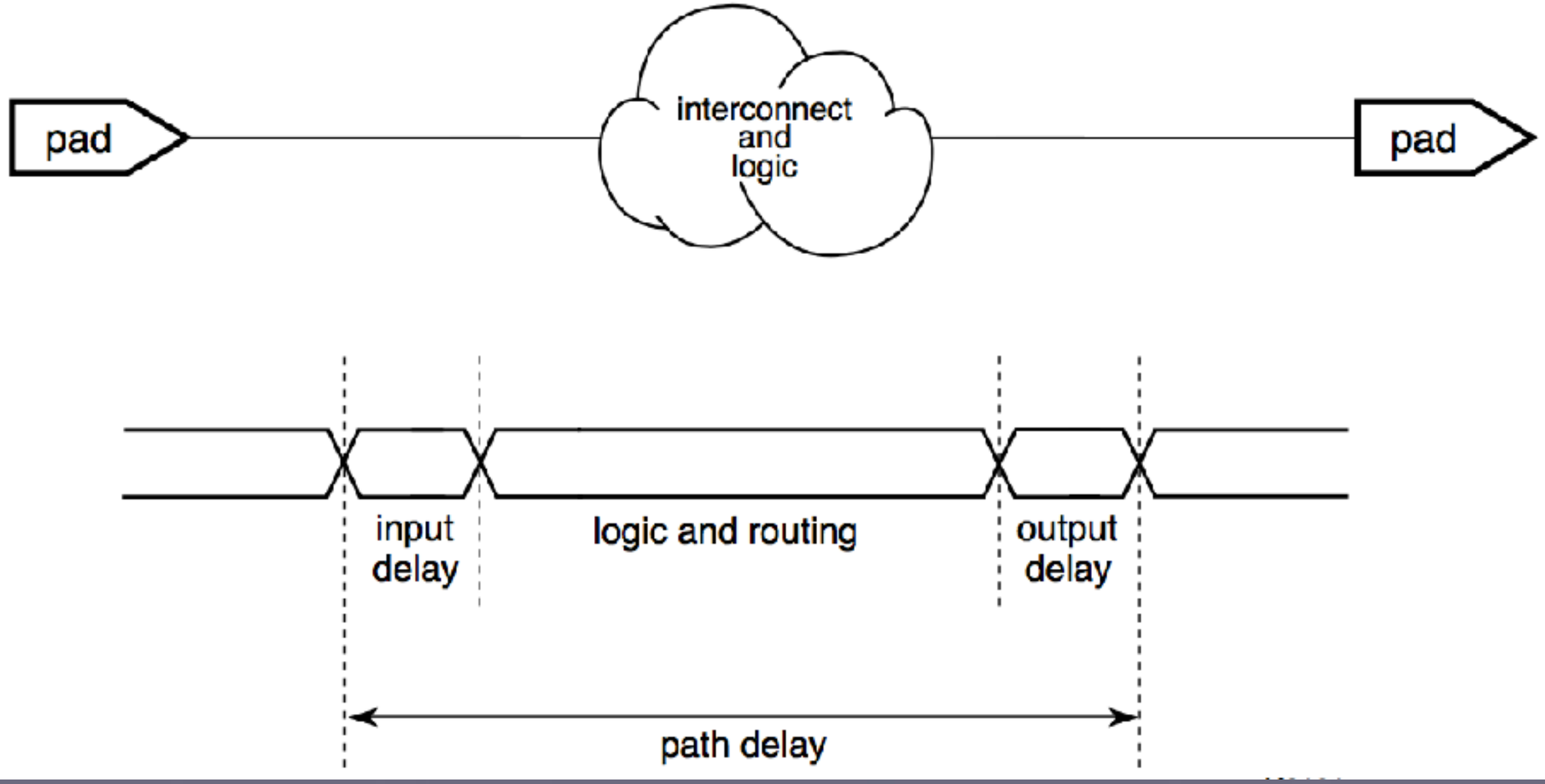


Clock to Pad

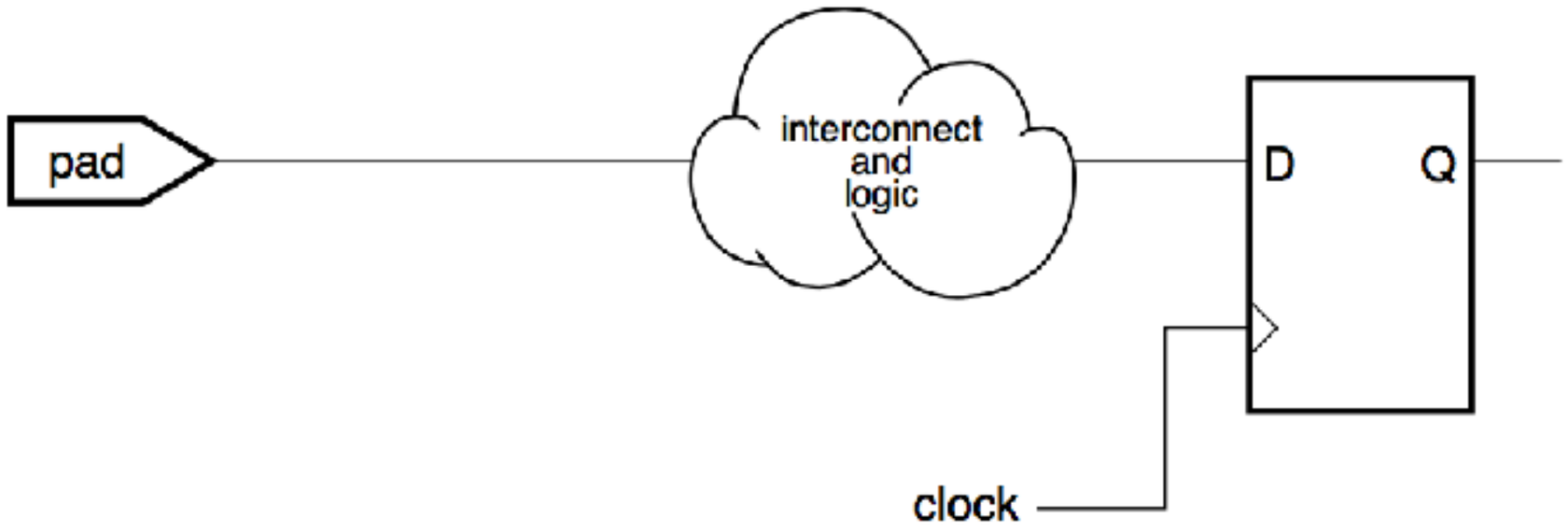

D
Q
interconnect
and
logic
pad
clock

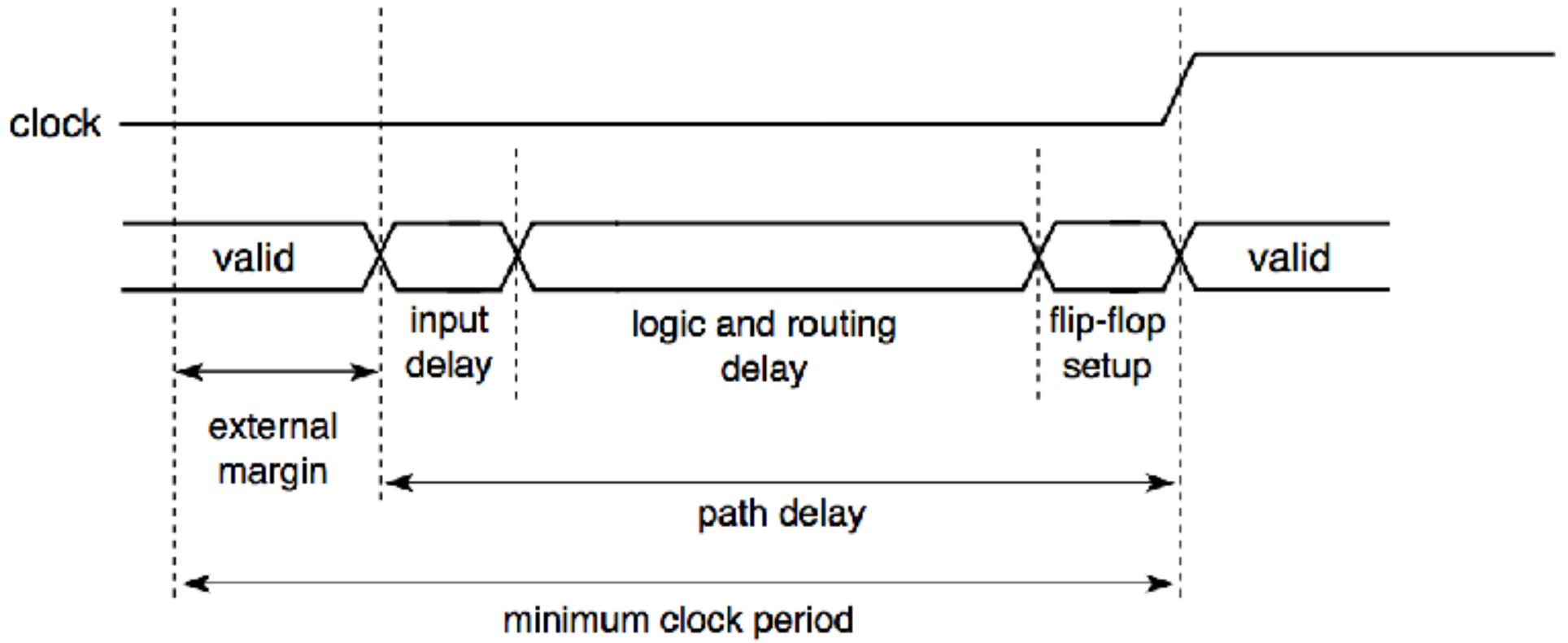

clock
data
valid
clock
to
output
logic and routing
delay
output
delay
external
margin
path delay
minimum clock period

Pad to Pad


pad
interconnect
and
logic
pad
input
delay
logic and routing
output
delay
path delay

Pad to Setup


pad
interconnect
and
logic
D
Q
clock


clock
valid
valid
input
delay
logic and routing
delay
flip-flop
setup
external
margin
path delay
minimum clock period

Defining the Clock

- The ucf file is used to specify the association of the design's top level signals with pins on the device
- The ucf file is also used to provide information regarding the design to the tools
- An example is the definition of the clock

```
## Clock signal
NET "clk"    LOC = "E3"   | IOSTANDARD = "LVCMOS33";
NET "clk" TNM_NET = sys_clk_pin;
TIMESPEC TS_sys_clk_pin = PERIOD sys_clk_pin 100 MHz HIGH 50% INPUT_JITTER 50 ps PRIORITY 1;
```

Input Timing Constraints

- Input timing constraints cover the data path from the external pin of the FPGA to the internal register that captures that data

- Our design treats the RESET as an asynchronous signal so we can tell the Timing Analysis tool that the timing on the RESET signal should be ignored

- The BUTTONS are now debounced and they also are considered to have no timing constraints

```
NET "reset"             LOC=N17 | IOSTANDARD=LVCMOS33; #IO_L9P_T1_DQS_14
NET "reset" TIG;
```

```
NET "Inbtn<1>"            LOC=P18 | IOSTANDARD=LVCMOS33; #IO_L9N_T1_DQS_D13_14
NET "Inbtn<1>" TIG;
NET "Inbtn<0>"            LOC=M18 | IOSTANDARD=LVCMOS33; #IO_L4N_T0_D05_14
NET "Inbtn<0>" TIG;
```

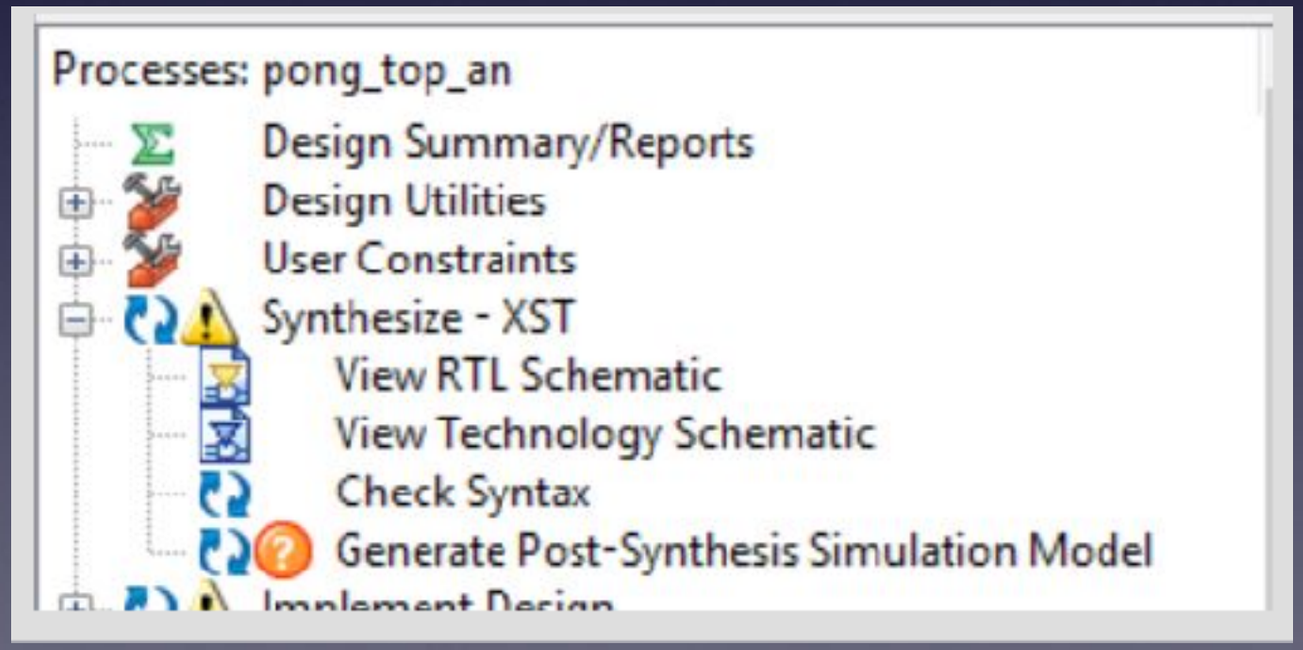
Output Timing Constraints

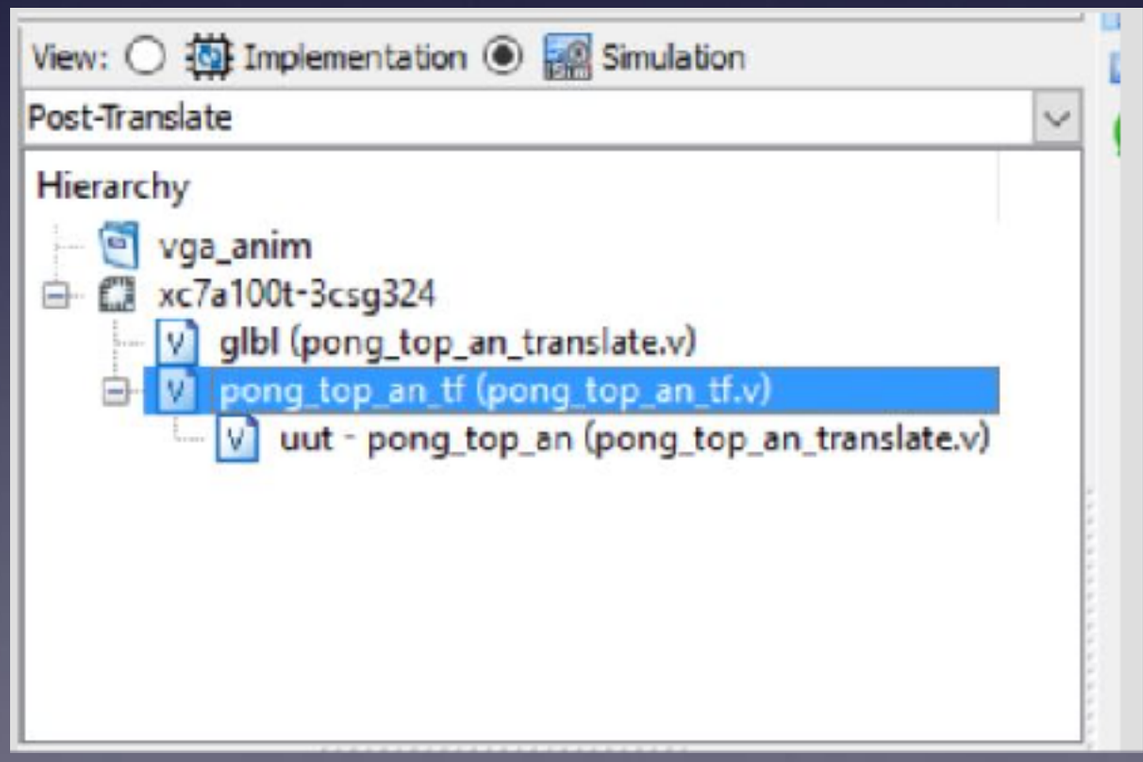
- Output timing constraints cover the data path from the the internal register that generates the data to the external pin of the FPGA
- Our design has the sync outputs and the rub outputs

```
NET "hsync" OFFSET = OUT 8 AFTER "clk";
NET "vsync" OFFSET = OUT 8 AFTER "clk";
NET "rgb*"  OFFSET = OUT 8 AFTER "clk";
```

Simulating Post Synthesis Model

- After synthesizing the design you can create a post synthesis model for simulation
- You can then run the simulation using the post synthesis model





Synthesis Verilog Model 1

pong_top_an_synthesis.v (~\Dropbox\CSULB_Current\360\Projects\Spring2017\vga_anim\vga_anim\netgen\synthesis) - GVIM

File Edit Tools Syntax Buffers Window Help

```
////////////////////////////////////////////////////////////////////////////////
// Copyright (c) 1995-2013 Xilinx, Inc.  All rights reserved.
////////////////////////////////////////////////////////////////////////////////
//   ____  ____
//  /   /\/   /
// /___/  \  /    Vendor: Xilinx
// \   \   \/     Version: P.20131013
//  \   \         Application: netgen
//  /   /         Filename: pong_top_an_synthesis.v
// /___/   /\     Timestamp: Sun Apr 30 18:12:27 2017
// \   \  /  \
//  \___\/\___\
//
// Command     : -intstyle ise -insert_glbl true -w -dir netgen/synthesis -ofmt verilog -sim pong_top_an.ngc pong_top_an_synthesis.v
// Device      : xc7a100t-3-csg324
// Input file  : pong_top_an.ngc
// Output file : C:\Users\tramel\Dropbox\CSULB_Current\360\Projects\Spring2017\vga_anim\vga_anim\netgen\synthesis\pong_top_an_synthesis.v
// # of Modules : 1
// Design Name : pong_top_an
// Xilinx      : C:\Xilinx\14.7\ISE_DS\ISE\
//
// Purpose:
//     This verilog netlist is a verification model and uses simulation
//     primitives which may not represent the true implementation of the
//     device, however the netlist is functionally correct and should not
//     be modified. This file cannot be synthesized and should only be used
//     with supported simulation tools.
//
// Reference:
//     Command Line Tools User Guide, Chapter 23 and Synthesis and Simulation Design Guide, Chapter 6
//
////////////////////////////////////////////////////////////////////////////////

`timescale 1 ns/1 ps

module pong_top_an (
  clk, reset, hsync, vsync, btn, rgb
);
  input clk;
  input reset;
  output hsync;
  output vsync;
  input [1 : 0] btn;
  output [11 : 0] rgb;
  wire btn_1_IBUF_0;
  wire btn_0_IBUF_1;
  wire clk_BUFGP_2;
  wire reset_IBUF_3;
```

Synthesis Verilog Model 2

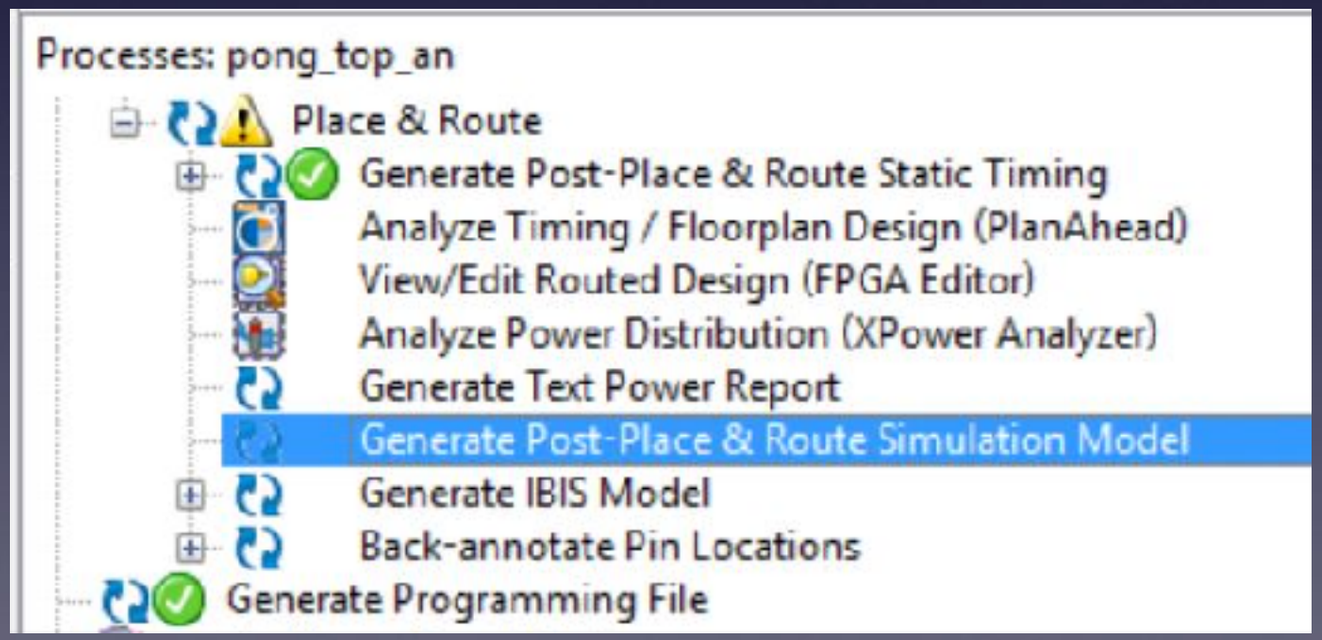
```
LUT2 #(
  .INIT ( 4'h2 ))
\pgauut/Mcompar_pix_y[8]_ball_y_b[8]_LessThan_36_o_lutdi4  (
  .I0(\pgauut/ball_y_b [8]),
  .I1(\vsuut/v_count_reg [8]),
  .O(\pgauut/Mcompar_pix_y[8]_ball_y_b[8]_LessThan_36_o_lutdi4_63 )
);
MUXCY   \pgauut/Mcompar_pix_y[8]_ball_y_b[8]_LessThan_36_o_cy<3>  (
  .CI(\pgauut/Mcompar_pix_y[8]_ball_y_b[8]_LessThan_36_o_cy<2>_67 ),
  .DI(\pgauut/Mcompar_pix_y[8]_ball_y_b[8]_LessThan_36_o_lutdi3_66 ),
  .S(\pgauut/Mcompar_pix_y[8]_ball_y_b[8]_LessThan_36_o_lut<3>_65 ),
  .O(\pgauut/Mcompar_pix_y[8]_ball_y_b[8]_LessThan_36_o_cy<3>_64 )
);
LUT4 #(
  .INIT ( 16'h9009 ))
\pgauut/Mcompar_pix_y[8]_ball_y_b[8]_LessThan_36_o_lut<3>  (
  .I0(\vsuut/v_count_reg [6]),
  .I1(\pgauut/ball_y_b [6]),
  .I2(\vsuut/v_count_reg [7]),
  .I3(\pgauut/ball_y_b [7]),
  .O(\pgauut/Mcompar_pix_y[8]_ball_y_b[8]_LessThan_36_o_lut<3>_65 )
);
LUT4 #(
  .INIT ( 16'h08AE ))
\pgauut/Mcompar_pix_y[8]_ball_y_b[8]_LessThan_36_o_lutdi3  (
  .I0(\pgauut/ball_y_b [7]),
  .I1(\pgauut/ball_y_b [6]),
  .I2(\vsuut/v_count_reg [6]),
  .I3(\vsuut/v_count_reg [7]),
  .O(\pgauut/Mcompar_pix_y[8]_ball_y_b[8]_LessThan_36_o_lutdi3_66 )
);
```

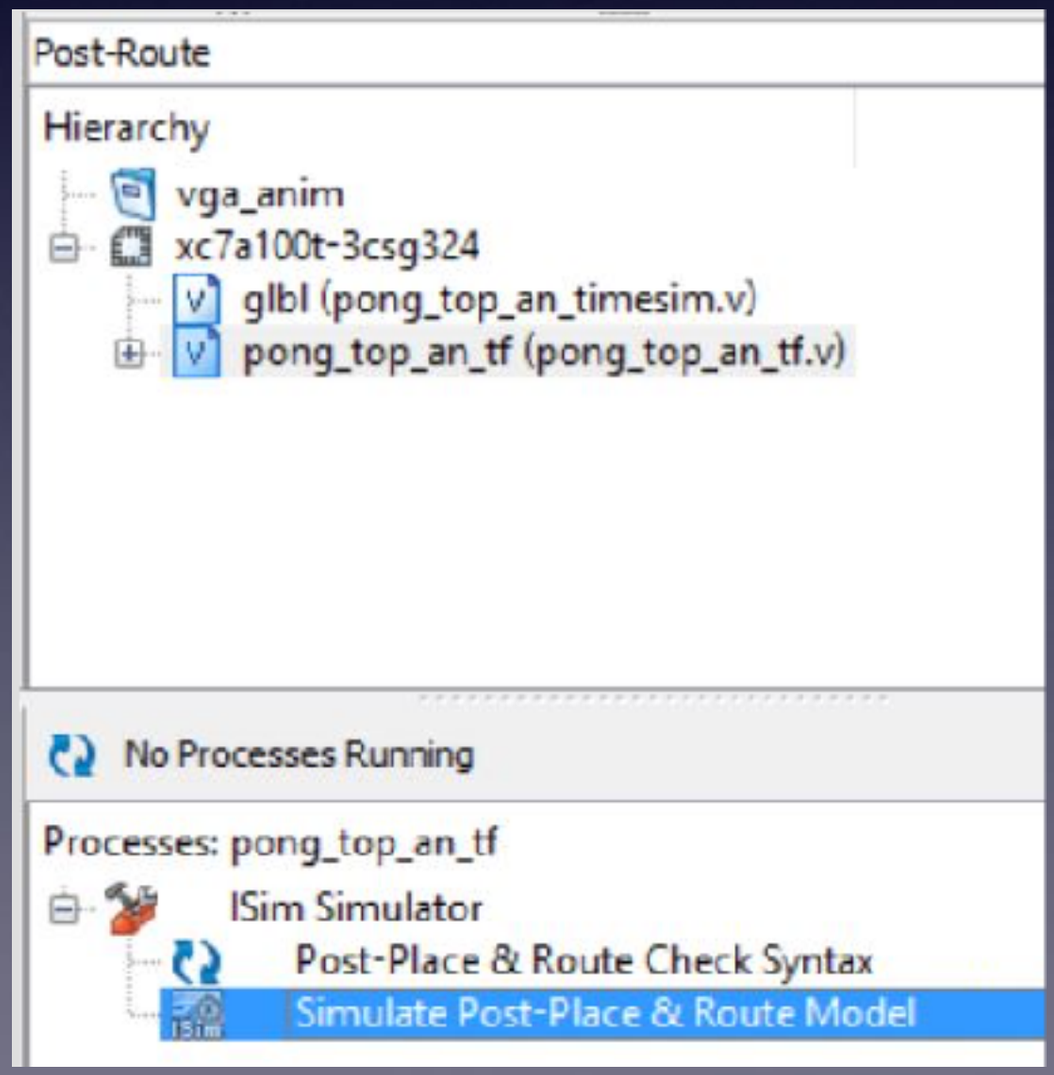
Synthesis Model Simulation

- Notice that the simulation is no longer “ideal”
- From the clock to hsync switching is 100 ps
- Delays are from the library cells

Simulating Post Layout Model

- After laying out the design you can create a post layout model for simulation
- You can then run the simulation using the post layout model





Post Layout Verilog Model

```
X_LUT6 #(
   .LOC ( "SLICE_X50Y136" ),
   .INIT ( 64'hA00A5005A00A5005 ))
\pgauut/Mcompar_pix_y[8]_bar_y_b[8]_LessThan_18_o_lut<3>  (
   .ADR1(1'b1),
   .ADR4(\vsuut/v_count_reg [6]),
   .ADR0(\pgauut/bar_y_b [6]),
   .ADR3(\vsuut/v_count_reg [7]),
   .ADR2(\pgauut/bar_y_b [7]),
   .ADR5(1'b1),
   .O(\pgauut/Mcompar_pix_y[8]_bar_y_b[8]_LessThan_18_o_lut<3>_35 )
);
X_LUT5 #(
   .LOC ( "SLICE_X50Y136" ),
   .INIT ( 32'h00F0A0FA ))
\pgauut/Mcompar_pix_y[8]_bar_y_b[8]_LessThan_18_o_lutdi3  (
   .ADR1(1'b1),
   .ADR4(\vsuut/v_count_reg [6]),
   .ADR0(\pgauut/bar_y_b [6]),
   .ADR3(\vsuut/v_count_reg [7]),
   .ADR2(\pgauut/bar_y_b [7]),
   .O(\pgauut/Mcompar_pix_y[8]_bar_y_b[8]_LessThan_18_o_lutdi3_41 )
);
X_ONE #(
   .LOC ( "SLICE_X50Y136" ))
\ProtoComp45.CYINITVCC  (
   .O(\NLW_ProtoComp45.CYINITVCC_O_UNCONNECTED )
);
```

Post Layout SDF Model with Netlist Cells

```
(DELAYFILE
  (SDFVERSION "3.0")
  (DESIGN "pong_top_an")
  (DATE "Mon May 01 13:08:59 2017")
  (VENDOR "Xilinx")
  (PROGRAM "Xilinx SDF Writer")
  (VERSION "P.20131013")
  (DIVIDER /)
  (VOLTAGE 0.95)
  (TEMPERATURE 85)
  (TIMESCALE 1 ps)
  (CELL (CELLTYPE "X_BUF")
    (INSTANCE b1\/Mcount_count_cy\<11\>\/b1\/Mcount_count_cy\<11\>_DMUX_Delay)
      (DELAY
        (PATHPULSE (50))
        (ABSOLUTE
          (IOPATH I O (65:137:137)(65:137:137))
        )
      )
  )
  (CELL (CELLTYPE "X_BUF")
    (INSTANCE b1\/Mcount_count_cy\<11\>\/b1\/Mcount_count_cy\<11\>_CMUX_Delay)
      (DELAY
        (PATHPULSE (50))
        (ABSOLUTE
          (IOPATH I O (63:133:133)(63:133:133))
        )
      )
  )
  (CELL (CELLTYPE "X_BUF")
    (INSTANCE b1\/Mcount_count_cy\<11\>\/b1\/Mcount_count_cy\<11\>_BMUX_Delay)
      (DELAY
        (PATHPULSE (50))
        (ABSOLUTE
          (IOPATH I O (62:128:128)(62:128:128))
        )
      )
  )
```

```
initial $sdf_annotate("netgen/par/pong_top_an_timesim.sdf");
X_BUF   \b1/Mcount_count_cy<11>/b1/Mcount_count_cy<11>_DMUX_Delay  (
  .I(\b1/Result [11]),
  .O(\b1/Result<11>_0 )
);
X_BUF   \b1/Mcount_count_cy<11>/b1/Mcount_count_cy<11>_CMUX_Delay  (
  .I(\b1/Result [10]),
  .O(\b1/Result<10>_0 )
);
X_BUF   \b1/Mcount_count_cy<11>/b1/Mcount_count_cy<11>_BMUX_Delay  (
  .I(\b1/Result [9]),
  .O(\b1/Result<9>_0 )
);
```

Synthesis Model Simulation

- Notice delays now include routing delays
- From the clock to hsync switching is 6.750ns
- Delays are from the library cells and parasitic delays