Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 12, December-2021

How to Time Logic - Primer for Static Timing Analysis

Deekshith Krishnegowda
Department of Electrical and Electronics Engineering
San Jose State University, San Jose, California

Abstract **— Chips for various end applications come in two types, one as ASIC and the other as FPGA. Different parameters like cost, R&D expenses, programmability are considered before implementing logic on an ASIC or FPGA. In addition to the previously mentioned parameters, back-end flow is another crucial parameter that must be considered. Before taping out a chip in ASIC flow, the design goes through several phases like Floor Planning, Clock Tree Synthesis, Place and Route, Physical Verification, etc. in back-end flow while in FPGA flow the implementation tool takes care of most of the backend flow with less human interaction. Be it the ASIC flow or FPGA flow, closing timing for a design with appropriate timing constraints is crucial else the design would violate setup or hold time constraints that will eventually lead to metastability in design. So, in this article, we will go through the basics of timing closure and learn about different parameters that affect timing with practical examples.**

Keywords - Gated clocks, Virtual clocks, Generated clocks, Clock skew, Clock jitter, Clock latency, Slack, Longpath, Setup time, Hold time, False path, Multi-Cycle Path (MCP).

I. Concept of Clock in Digital Design

Clock is the reference point used in digital systems for monitoring the inputs and the outputs. The time between two adjacent active edges or two adjacent inactive edges is referred as period of the clock. In an ideal clock, the rise time and fall time is zero i.e., the transition from 0 to 1 or 1 to 0 does not take any time. But in actual design, the rise and fall time of the clock is not ideal [1] and this uncertainty is measured in volts/sec and is referred as slew rate.

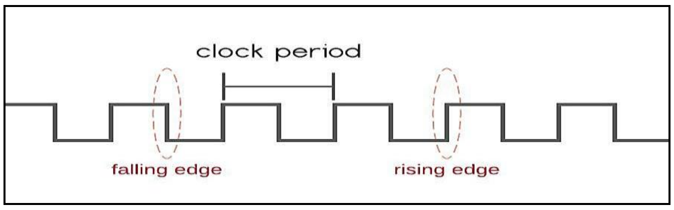

Fig. 1. Ideal clock cycle. No rise delay or fall delay is observed.

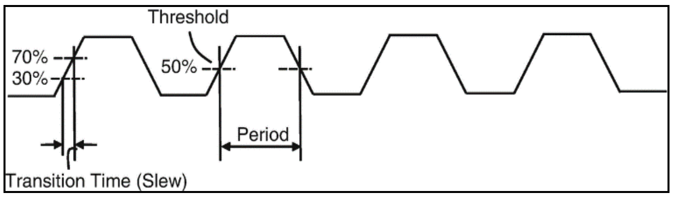

Fig. 2. Actual clock cycle. Transition time from low to high or from high to low is defined as skew [1].

A. *Types of clocks*

In any digital logic there are three types of clock viz gated clocks, virtual clocks, and generated clocks. When performing timing analysis, constraints should be applied to all these clocks so that the tool can recognize the type of clock and can group them accordingly. These clocks will be grouped under synchronous or asynchronous group by the tool. The clocks which share same source will be grouped under synchronous and the clocks which do not share same source will be grouped under asynchronous clock.

1) Gated clocks

Gated clocks are used along with enable signal which controls the clock supply to the flipflop. Clock can be turned off or on by controlling the enable signal. These types of clocks are widely used in low power design [2].

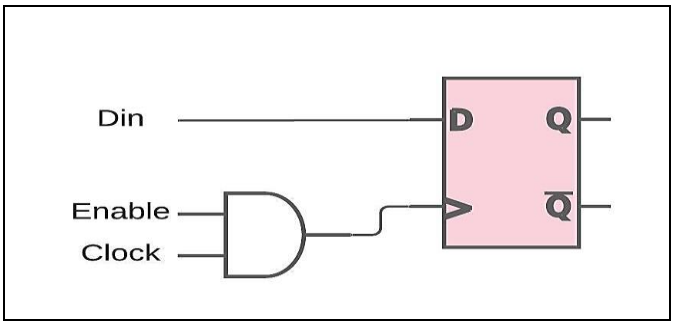

Fig. 3. Circuit of a gated clock. By controlling the enable, clock supply to the flipflop can be turned on or off.

2) Virtual clocks

Virtual clocks are used as a reference point to drive output or to monitor input [1]. These clocks do not affect the functionality of the design and they don’t have a physical source inside the chip.

3) Generated clocks

Today’s chips consist of different blocks within the top design. For example, there can high-speed block that works on fast clock and there can be low-speed design such as memories that work on slow clock. Typically, there will one PLL (Phase Locked Loop) that will serve as the master clock source in the design. Slow and fast clocks are generated using clock divider or clock multiplier logic and these internally generated clocks are referred as generated clocks [1. A simple clock division logic is shown in the Figure 4 and waveform corresponding to that circuit is shown in Figure 5.

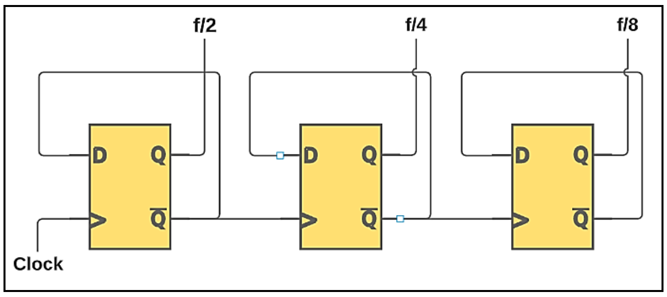

Fig. 4. Clock divide by 8 circuit.

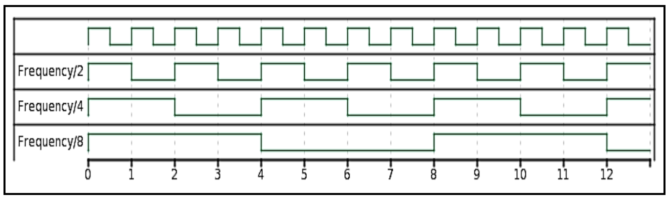

Fig. 5. Waveform for clock divide by 8 logic.

B. *Clock variations*

When a clock is propagated in a digital design, there are three types of uncertainty that will affect the clock namely skew, jitter and latency. All these clock variations will affect the timing of the circuit and while analyzing timing of circuit, it is important to keep track of all these variations.

1) Clock skew

The delta in time where same clock edge reaches two flops can be different. This difference is referred as clock skew, and this is caused because of uneven delay in clock path. Based on direction of direction of data and clock, clock skew can be of two types: positive clock skew and negative clock skew [1]. If data and clock are in same direction, then the skew between flops is referred as positive skew and if data and clock are in opposite direction, skew between two flops is referred as negative clock skew. Positive skew and negative skew can affect the timing if circuit in different ways.

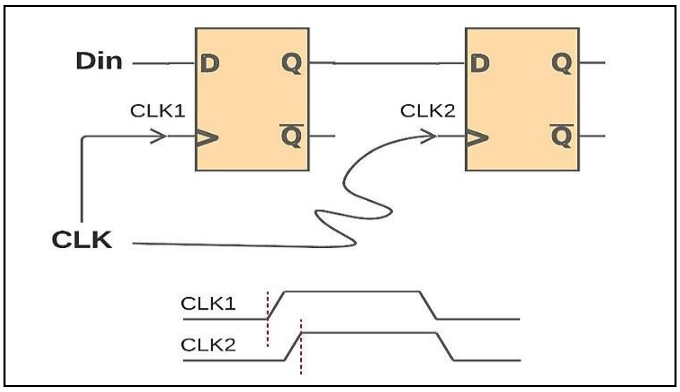

Fig. 6. Circuit of datapath with positive skew. The long path of CLK2 will constitute for the delay between CLK1 and CLK2 for same edge of the clock.

2) Clock jitter

Because of the variations in power supply or temperature, the clock edge might be produced early or late compared to ideal clock edge. Because of this, the clock period can erratic. This difference is referred as clock jitter. Clock jitter and clock skew can increase or decrease the time available for a net to satisfy timing without causing violation [3]. So, while doing timing analysis, these uncertainties should be provided to the tool using appropriate commands.

3) Clock latency

Clock latency is the time it takes for clock signal to propagate from its source (crystal oscillator) to the clock pin of a register. Depending on the path, clock latency can be further subdivided into source clock latency and network clock latency [3].

II. SLACK AND CRITICAL PATH

A. *Input arrival time & output required time*

Before we discuss about slack, it is important to understand the concept of node, input arrival time and output required time. A pin connection between cells, or from input/output to the cells is referred as nodes. Because modern ASIC (Application Specific Integrated Circuit) consists of millions of gates, logic cones found inside those designs can be extremely complex. Hence for timing analysis, these nodes are broken down to several levels of logic gate. In Figure 7 it can be observed that there are 4 modelling pins, 8 cell pins and 2 hierarchical pins. Coming to input arrival time, it is defined as the time interval in which the input data arrives at the input pin of the cell/block with respect to the nearest active edge of the clock. The output required time is defined as the time at which stable output is required at the output pin with respect to the nearest active clock edge.

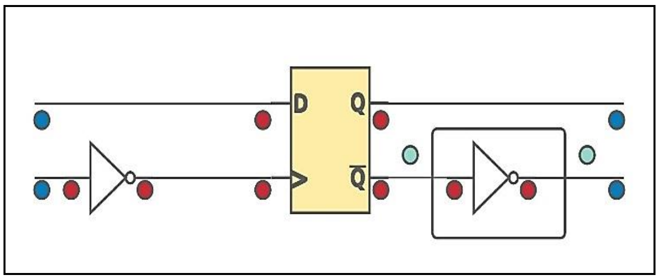

Fig. 7. Circuit representing different pin and node. Navy blue is modelling pin, red is cell pin and light blue is hierarchical pin.

Slack is defined as the difference between output required time (RT) and input arrival time (AT) [3]. For any circuit under consideration, it is important to have positive slack or worst-case scenario zero. If a circuit has negative slack, then the design would fail timing constraint and circuit has to be optimized or redesigned.

To calculate the slack, we must calculate the input arrival time and output required time for all nodes in the circuit. Let us consider the below circuit in Figure 8 to calculate slack. The gate delay of different gates is shown in the Figure 8 as well. Input arrival time of 1ns (nano seconds) is assumed and 0.2ns wire delay is taken into consideration. To calculate AT, we should determine timing at each node and as described previously, the logic is broken down into several levels with timing analysis performed from each input towards the output direction. For example, take the NOT gate in the below circuit. AT at input B is 1ns. Adding the 0.2ns wire delay, AT at input of NOT gate is 1.2ns. Given that NOT gate has 20ns of gate delay, change in output can happen only after 21.2ns. In the same fashion, AT is calculated at each output node and finally at the output pin of the circuit. After all calculations, we get AT of 91.8ns at the output pin of the circuit.

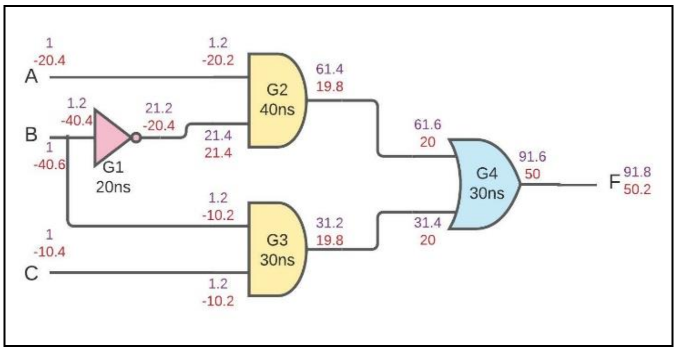

Fig. 8. Example for slack calculation. Timescale used is ns. Purple represents AT and red represents RT.

To calculate the RT, the same principle is used in opposite direction i.e. from output towards the input. Let's assume RT of 50.2ns at the output pin. When back propagated, RT at output pin of OR gate is 50ns. This is obtained after subtracting 0.2ns from 50.2ns. Taking 30ns gate delay, the RT at input pins of OR gate is 20ns. After all calculations, the RT at each input pins are -20.4ns, -40.6ns and -10.4ns.

$$Slack = RT - AT \tag{1}$$

Slack at the output pin of the circuit is 50.2 – 91.8 = -41.6ns.

B. *Longpath*

Critical path, also referred as longpath, in a circuit is the path with worst slack [4]. Critical path defines the operating frequency of the design. For example, if the longpath delay of a design is 40ns then the maximum operating frequency of that design is 1/40ns = 25Mhz (Mega Hertz). Hence it can be concluded that the critical path is indirectly proportional to the operating frequency of the design i.e. the longer the critical path is, the less frequency the design will function.

In section 1, we discussed how actual clock signal is different than ideal clock. This was largely because of the rise and fall delay of components such as buffers and inverters found in the clock network. In similar fashion, all wires, and nets of the datapath are susceptible to uneven rise and fall delay. This delay is because of the intrinsic characteristics of transistors that make up the gates found in digital circuits. Uneven rise and fall time will affect the longpath of the design. So, let us consider the circuit in Figure 9 to study how rise and fall time affects the longpath of the design. The input capacitance on pin A and B of NAND gate is 3.8ff (femto farad) and 4.0ff. The output capacitance per load is 7ff. The rise delay and fall delay of NAND gate is 0.040ns and 0.030ns. The rise and fall delay factor are 0.009 and 0.007 respectively. The output pin has a capacitance of 15ff.

To calculate the longpath of the above circuit, firstly, we must calculate the capacitance on gate output pins i.e., C, D, E and F. The capacitance on output pin of NAND gate in this scenario is given by Equation 2.

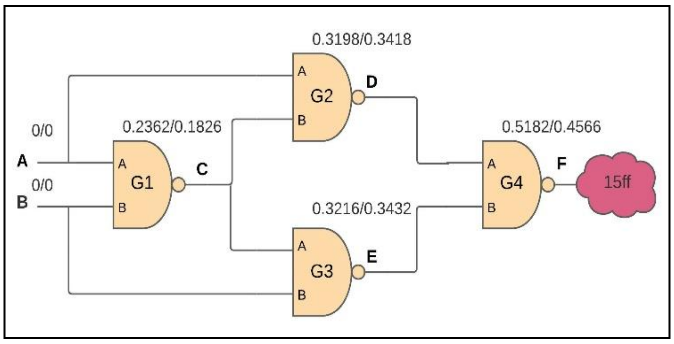

Fig. 9. Example for longpath calculation. Timescale used is ns. Numerator represents total rise delay and denominator represents total fall delay.

*Output capacitance (C) = (Number of loads * Output capacitance per load) + input capacitance of each load* (2)

For example, the output Capacitance on pin C is given by 2 * 7ff/load + 4ff + 3.8ff = 21.8ff. Output capacitance of D, E and F is calculated in the same way and the output capacitance value of 10.88ff, 11ff and 15ff is obtained respectively.

Secondly, we must calculate the net rise and fall delay on each output node of NAND gate. The below Equation 3 describes the formula to calculate net rise and fall delay of the above circuit.

*Net rise delay = Rise delay of gate + (factor * output capacitance)* *(3)*

*Net fall delay = Fall delay of gate + (factor * output capacitance)* *(4)*

For example, the net rise delay on gate output D is given by 0.040ns + (0.009*10.88ff) = 0.137ns. The net fall delay on same output is given by 0.030ns + (0.007+10.88ff) = 0.106ns. The net rise and fall delay on all gate output node is show in Table 1.

TABLE I. NET RISE AND FALL DELAY OF ALL GATE OUTPUT PINS.

	Rise delay (ns)	**Fall delay (ns)**
C	0.236	0.182
D	0.137	0.106
E	0.139	0.107
F	0.175	0.135

Thirdly, we must calculate the relation between rise and fall function of output to the rise and fall characteristics of the input. As observed from the truth table of NAND gate shown in Table 2 below, it can be concluded that the output Z(rising) is a maximum function of input A(falling) and input B(falling) and output Z(falling) is a maximum function of input A(rising) and input B(rising). This is shown in Equation 5 and Equation 6 below.

$$Z(rising) = f\,max(\,A(falling),\,B(falling)\,) \tag{5}$$

$$Z(falling) = f\,max(\,A(rising),\,B(rising)\,) \tag{6}$$

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 12, December-2021

The last step is to calculate the total rise and fall time of each gate output based on the output function. For example, lets calculate the rise delay on gate output F. The rise delay is maximum function of D and E falling i.e. max function of 0.341ns and 0.343ns. Adding 0.343ns and net rise delay of 0.175ns, we get total rise delay on output F as 0.518ns. In similar fashion, total rise and fall delay is calculated on output gate pins and the values are shown in Table 2. It can be observed from Table 2 that the longpath is 0.518ns and the path of the longpath is through gate G1, G3 and G4 as shown in Figure 9.

TABLE II. TOTAL RISE AND FALL DELAY OF ALL GATE OUTPUT PINS.

	C (ns)	D (ns)	E (ns)	F (ns)
Total rise delay	0.236	0.319	0.321	0.518
Total fall delay	0.182	0.341	0.343	0.456

III. SETUP AND HOLD TIME

Nets in sequential circuit must adhere to timing constraints of the flop. These timing constraints are determined by the technology node and type of the flipflop. The time period for which the data should be stable after the active edge of the flipflop is known as hold time and the time period for which the data should be stable before the active edge of the clock is known as setup time. If the data changes in the stable region before the active edge of the clock (indicated as T_{su} in Figure 10), then the flop violates setup time constraint and if the data changes in the stable region after the active edge of the clock (indicated as T_{hd} in Figure 10), then the flop violates hold time constraint. Flop will enter a metastable state if either setup or hold time constraint is violated [1,3,4].

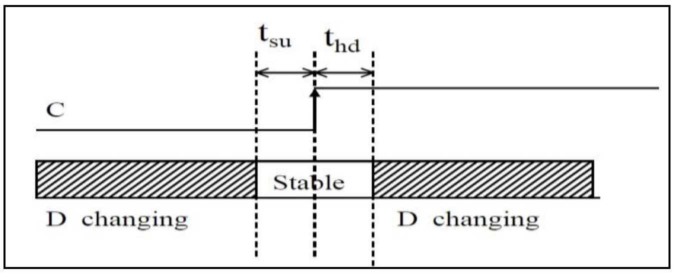

Fig. 10. Setup and hold time of a flop. Data to the flop must be held constant in the stable window else flop enters metastable state.

Input Setup and Hold time (IS & IH) can be affected by clock period (Tclk) and different delays in the design such as clock skew, clock-q delay (Tclk-q), and logic delay between the flop. In addition to different delays, clock latency and clock jitter will shift the clock edge from the ideal position and hence these uncertainties will also affect the setup and hold time constraint of the flop. Clock-q delay is the delay between the active edge of the clock till there is a change in output of the flop. Considering all the delays, the setup and hold time constraint is expressed in Equation 7 and Equation 8 below respectively. All flops in the design should adhere to these equations else the flop enters a metastable condition.

$$Tclk > 2*clock\_skew + IS + logic\_delay + Tclk\text{-}q \quad (7)$$
$$IH < logic\_delay - 2*clock\_skew + Tclk\text{-}q \quad (8)$$

For any technology node, the setup & hold time window, and clock-q delay remains constant. Given that designers need to sign off timing for a fixed frequency, Tclock also stays constant. So, the only variable that can be modified in the above equations is logic delay between the flops. Solving for logic_delay, we have the following Equation 9. It can be observed from the equation that the logic delay should not exceed a particular value, else it will cause setup time violation and also the logic delay should not be under a particular value, else it will cause hold time violation.

$$(Tclk - (2*clock\_skew + IS + Tclk\text{-}q)) > logic\_delay > ((2*clock\_skew + IH) - Tclk\text{-}q) \quad (9)$$

IV. FALSE PATH AND MULTICYCLE PATH

In some digital logic, under certain conditions timing can be ignored on certain paths. While performing timing analysis these paths can be set as false path or multicycle path so that timing checks are not done on these paths [3]. Because of the way some logic is designed, data transition on certain path cannot occur under any conditions. Such paths can be ignored for timing checks by defining these paths as false path [5]. For example, in Figure 11 below, A can propagate to output if select is 0 and B can propagate to output if select is 1. Any other combinations to output are not possible because of the way the select line is connected to common source.

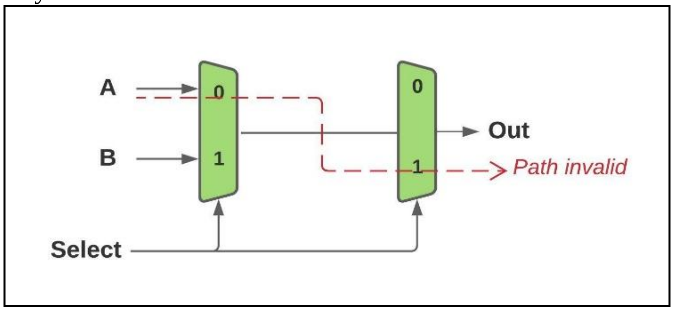

Fig. 11. Circuit of an architectural false path. Since both mux are connected to same select line, crisscross connection between two mux is invalid.

Another example is that of a synchronizer that is used while doing clock domain crossing. The first flop in the synchronizer circuit may or may not enter a metastable condition. It enters a metastable condition if the signal crossing domain toggles during the setup or hold time of the first flop. In a two flip flop synchronizer, the second flop ensures that the signal is stable in receive domain even if the first flop violates timing. Though the output of first flop violates timing, the final output of two flip flop synchronizer circuit is stable. So, this path can also be set as False path since design ensures no timing violation occurs.

The golden rule in digital logic is that the data from the launch flop should be sampled by the receive flop in the next active clock edge. This ensures that the setup time check happens at next active clock edge while the hold time check happens at same clock edge [2]. But, in some cases if the data is held constant for two consecutive active clock edges, then the setup time check is done at 2nd active clock edge after the data is launched. For example, in Figure 12 below, if Enable signal changes every 2 clock cycle then the data to the flop

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 12, December-2021

also toggles every 2 clock cycle. If Enable is held constant, then data to flop is also held constant. Setup time check can be ignored when Enable is constant because data is held constant as well. These types of paths are referred as multicycle paths. Specifying the number of clock edges for which the data is held constant is necessary when defining multicycle paths to ensure proper checking of setup and hold time violation is done.

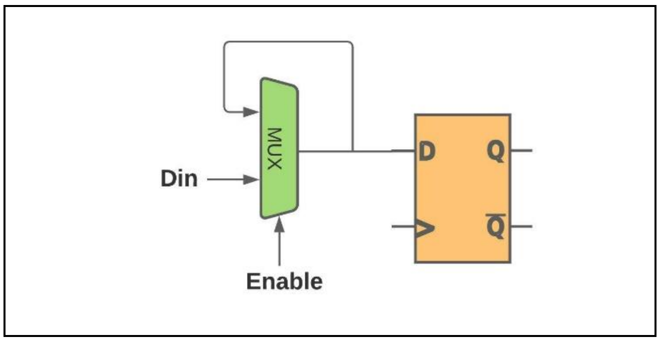

Fig. 12. Circuit of a Multicycle path. If the Enable signal does not toggle every clock cycle, then this path can be set as multicycle path while performing timing analysis.

In this paper, basics of Static Timing Analysis were introduced. Concept of timing violation in digital circuit design was discussed. Calculation of slack and longpath were described with examples and paths with no timing checks were also discussed. But while designing any chip, timing requirements are not met in first try. Numerous logic optimization is done until the design meets required timing constraints. These optimization techniques, ways to solve setup & hold time violation, and quick timing fixes using Verilog for complex arithmetic logic will be discussed in the follow up review paper.

REFERENCES

[1] Gangadharan S., Churiwala S. (2013) Other Clock Characteristics. In: Constraining Designs for Synthesis and Timing Analysis. Springer, New York, NY. https://doi.org/10.1007/978-1-4614-3269-2_8

[2] Chia-Ming Chang, Shih-Hsu Huang, Yuan-Kai Ho, Jia-Zong Lin, Hsin-Po Wang and Yu-Sheng Lu, "Type-matching clock tree for zero skew clock gating," 2008 45th ACM/IEEE Design Automation Conference, 2008, pp. 714-719. https://doi.org/10.1145/1391469.1391653

[3] Bhasker J., Chadha R. (2009) Delay Calculation. In: Static Timing Analysis for Nanometer Designs. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-93820-2_5

[4] http://www.vlsi-expert.com/2011/03/static-timing-analysis-sta-basic-timing.html