Timing Analysis

This lecture describes how static timing analysis is used to ensure timing constraints for a digital design are met. After this lecture you should be able to be able to apply the terms defined in this lecture and do calculations involving clock rate, propagation delays and setup/hold time requirements.

Introduction

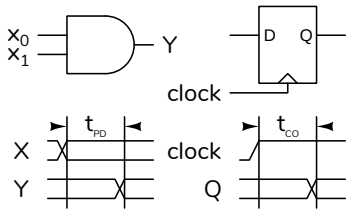
Timing constraints are requirements that a design must meet. A typical example is that the design must operate at a specified minimum clock frequency. Meeting these constraints is often as difficult as ensuring that a design is logically correct.

A designer must correctly specify timing constraints such as clock periods and external circuit delays. This allows the design software to check if a given design will meet the device's internal timing requirements such as flip-flop setup and hold times. If not, the designer must change the design or relax the constraints in order to meet the timing requirements.

The performance of ICs varies from device to device and with changes in temperature and voltage. Building a circuit that appears to work properly is not enough to ensure that a design will work reliably. This is because the same design may fail on a different device, at a different temperature or with a different supply voltage.

Propagation Delays

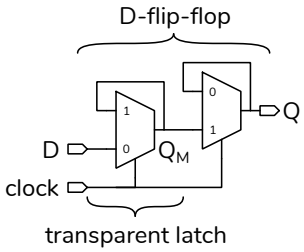
Propagation delay, t_{PD}, is the delay from a change at an input to the corresponding change at an output. The clock-to-output delay, t_{CO}, a type of propagation delay, is the delay from the rising edge at a flip-flop clock input to the change at the Q output.



These delays are primarily caused by the time required to charge the parasitic capacitances of transistors and the metallic "interconnects" that connect them.

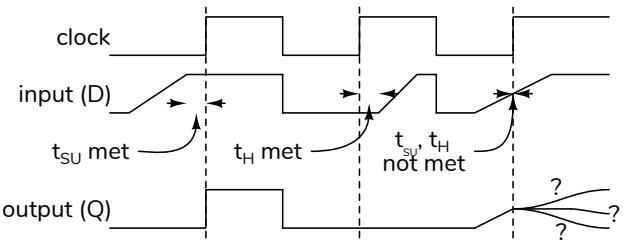
Metastability, Setup and Hold Times

Consider the following implementation of an edge-triggered D flip-flop:



When the clock input is 0, the output of the first multiplexer follows the input – the latch is "transparent". When the clock input is 1, the output level is fed back to the input and held at that level[1].

If the D input is changing and is still near the logic threshold voltage at the time the clock changes from 0 to 1 (the rising edge), then the multiplexer might not be able to decide whether to feed back a 0 or 1. The multiplexer output (Q_M) could remain at an invalid level for a long time – longer than the t_{CO} specification. This behaviour is called "metastability"[2] and can result in incorrect operation.



[1] This is a "master-slave" flip-flop. The second, "slave," latch holds the previously latched value when the clock is 0

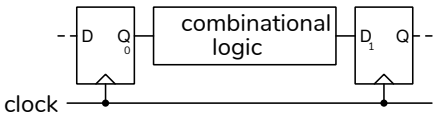
[2] The output is "meta" stable because it appears to be stable at a level that is not one of the true stable states (H or L).

To avoid metastability we must ensure the voltage at the latch input is at valid level long enough to drive the latch output to a valid voltage level before the rising edge of the clock. The time required for this is called the "setup" time, t_{SU}.

The input level must also be held at the correct level until the transparent latch has finished switching to the feedback mode. This is typically a much shorter time – typically zero – and is called the "hold" time, t_H.

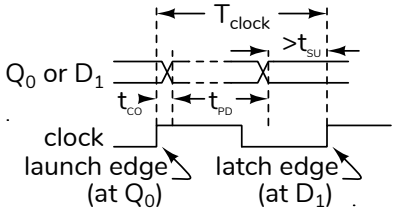
Synchronous Design

By using "synchronous" designs we can avoid metastability. Synchronous designs use edge-triggered flip-flops, all with a common clock. There is combinational logic between the flip-flop outputs and inputs:



By ensuring the propagation delays through the combinational logic are short enough to meet the flip-flop setup and hold requirements we can avoid metastable behaviour.

The timing diagram below shows the relationship between the clock edges and the valid times at the outputs and inputs the flip-flops above:



Q_0 changes t_{CO} after the rising clock edge. After t_{PD} D_1 will have a correct and valid logic level. This must happen at least t_{SU} before the *next* rising clock edge. And this level must be held for at least t_H before it changes.

The flip-flop setup and hold times for internal flip-flops are often called "library" or "micro" times to distinguish them from an IC's external IO setup and hold times.

The following table summarizes these timing specifications:

specification	type	measured between
t_{CO}	guaranteed (max.)	input → output
t_{PD}	guaranteed (max.)	input → output
t_{SU}	requirement (min.)	input → input
T_{clock}	requirement (min.)	input → input

The diagram above identifies two clock edges, the "launch" and "latch" edges. These two edges are separated by one clock period. This is the time allowed for changes in one flip-flop to propagate to the input of another (or the same) flip-flop.

Note that the clock signal itself may have a propagation delay and may arrive at different flip-flops at different times. This is known as "clock skew" and must also be taken into account by the timing analysis software.

Static Timing Analysis

In most cases the timing requirement that is most difficult to meet is the minimum setup time.

From the timing diagram above we can write an expression for the (minimum) available setup time:

$$t_{SU} = T_{clock} - t_{CO} - t_{PD}$$

where t_{SU} is the available set up time, T_{clock} is the clock period, t_{CO} is the maximum clock to output delay of the launching flip-flop and t_{PD} is the maximum propagation delay through the combinational logic. Both t_{CO} and t_{PD} are the maximum delays specified by the manufacturer.

The propagation delay depends on the circuit design. Each logic function (multiplexer, adder, gate, etc.) between the output of one flip-flop and the input of another increases t_{PD} for that path. The timing analyzer computes t_{PD} for every path between flip-flop outputs and inputs and uses the path with the longest delay when deciding if the t_{SU} requirement is met.

Exercise 1: Which of the specifications in the formula above decrease the available setup time as they increase? Which increase it?

The amount by which the minimum available setup time exceeds the minimum required setup time is known as the "slack":

$$\text{slack} = t_{SU}\text{ (available) - }t_{SU}\text{ (required)}$$

If the slack is positive then the available setup time exceeds the required value and the t_{SU} requirement is met, otherwise it is not and the circuit may not operate correctly due to metastable behaviour: excessively long clock-to-output delays.

Exercise 2: For a particular circuit f_{clock} is 50 MHz, t_{CO} is 2 ns (maximum), the worst-case (maximum) t_{PD} in a circuit is 15 ns and the minimum setup time requirement is 5 ns. What is the setup time slack? Will this circuit operate reliably? If not, what it the maximum clock frequency at which it will?

Exercise 3: What is the maximum clock frequency for a counter using flip-flops with 200 ps setup times, 50 ps hold times and adder logic that has a 250 ps propagation delay?

SDC Timing Constraints

Timing and other design constraints are provided to the design software by a file written using the "Synopsys Design Constraint" (SDC) syntax. The only SDC command we will cover in this course is the clock frequency constraint. An example is:

```
create_clock -name clk50 -period 20
   [get_ports {clk50}]
```

which specifies a clock named **`clk50`** at a port named **`clk50`** that has a 20 ns period[3]. The SDC format allows for many additional constraints such as external propagation delays (e.g. **`set_input_delay`**) and signals that should be ignored by the timing analysis (**`false_path`**).

Closing Timing

"Closing" timing is the process of iterating a design until all paths have positive slack. If design does not meet its timing requirements we can:

- Change the design to speed up critical timing paths. This might mean having more logic in parallel or "pipelining" – dividing up the computation across multiple clock cycles.
- Use a faster device – one with lower t_{PD} and/or lower t_{SU}.
- Relax the design constraints by reducing the required clock rate.
- Reduce the interconnect delays. We can have the EDA software spend more time ("effort") optimizing the routing between flip-flops or use a larger PLD (which may allow for layouts with lower delays).

the choice will depend on the project requirements and available resources.

Exercise 4: Which of the above would increase design time? Which would increase the unit costs? Which would lower quality?

PVT and Corners

The propagation delays on one die will depend on the temperate and voltage. There will also be random differences between die due to differences in the processing of each wafer or a die's location on the wafer. STA can be repeated using delays for the expected "PVT" (Process, Voltage, Temperature) extremes. The PVT combination that results in the maximum or minimum delays is called a "corner."

Asynchronous Clocks and Inputs

If all clocks are derived from the same source clock (e.g. through clock division or using a PLL) the time relationships between clocks remains constant and it's possible to verify that timing constraints will be met.

However, if two clocks are physically independent then this is not possible – the clock edges will drift relative to each other and the setup and hold timing requirements of flip-flops using different clocks (those in different "clock domains") are bound to be violated eventually. Even though it's not possible to do timing analysis for asynchronous signals, it is possible to estimate the mean time between failure (MTBF) due to metastable events when signals cross clock "domains."

The probability of a metastable event increases proportionately with the product of the clock rates. For slow events, such as button presses, this probability will be negligible.

[3] Note that this does not change the clock frequency of your circuit – that is determined by the oscillator on your board.

Timing Simulations

After PAR ("post-layout") the interconnect delays can be calculated and a "timing-annotated" netlist can be generated that includes propagation delays. These can be used in simulations and the simulator can check that the setup and hold requirements of each flip-flop are being met.

The advantage of this "dynamic" timing analysis is that the simulation results are independent of, and can serve as a check on, user-provided timing constraints. The disadvantage is that the simulation may not cover all possible events. Timing simulations can be time-consuming for large designs and are primarily used for ASIC "sign-off" before "taping out" a design to send for manufacturing.