50.002 COMPUTATIONAL STRUCTURES

INFORMATION SYSTEMS TECHNOLOGY AND DESIGN

TL;DR : Sequential Logic & Syncrhonization

1 Sequential Logic

Look at the diagram below.

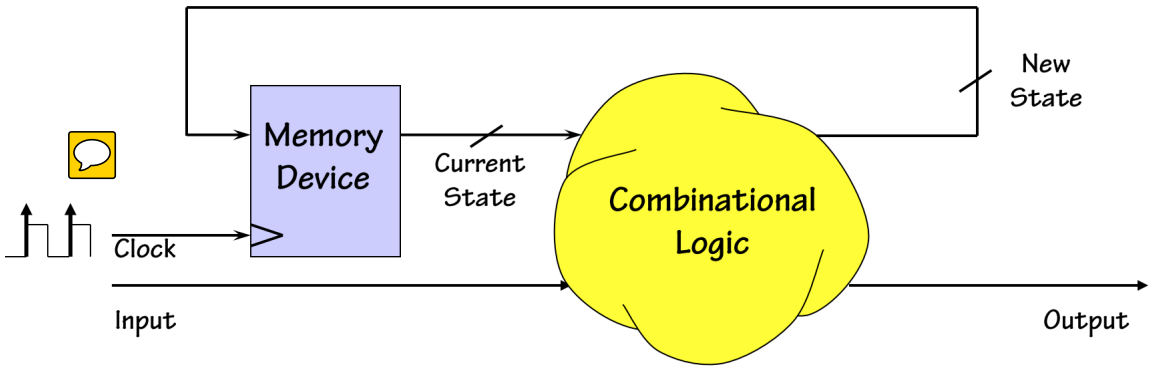


Figure 1

So far we have learned **combinational device** (also called combinational logic), that implements your truth table functional specs. But it isn't enough to make a complete computer that we know today. We are missing the **storage (memory) device**.

2 Storage Device: D-Latch

A D-latch is a mux with a feedback loop,

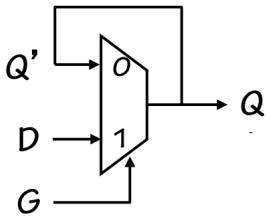


Figure 2

How it works:

1. G is clock signal. It will periodically switch between '1's and '0's
2. If G is 1, then the input signal on wire D will be passed to output wire Q, independent of the signal on wire Q'. Lets call this the **write mode**.
3. If G is 0, then output signal on wire Q follows the signal on wire Q', independent of input wire D. Lets call this the **read memory mode**.

Possible problems:

1. **Storing invalid information:** If G changes from 1 to 0 at the exact moment when D is changing (invalid), then we will end up storing that new invalid information on wire D.
2. **Unstable Output because of unstable input:** In practice, input from users can be highly unstable. We do not want it to affect our devices. If G is 0, then it will pass **all** input from D to the output wire Q, regardless on whether it is a valid or stable input or not.

We address these problems in the next two sections.

3 The Dynamic Discipline

The dynamic discipline addresses problem 1 above: **storing invalid information**. It is **very important to never violate the dynamic discipline to ensure any sequential logic circuits to work properly**.

Take a look at the clock signal below for G, it basically changes between 0 to 1 periodically:

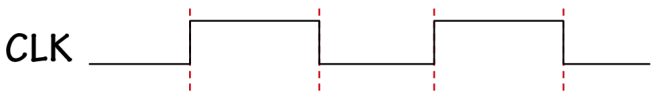


Figure 3

The dynamic discipline states that:

1. $T_{setup} = 2t_{pd}$
2. $T_{hold} = t_{pd}$

where,

1. T_{setup} = the minimum amount of time that the voltage on wire D needs to be stable **before** the **clock edge changes from 0 to 1**.
2. T_{hold} = the minimum amount of time that the voltage on wire D needs to be stable **after** the **clock edge changes from 0 to 1**.
3. t_{pd} is the propagation delay of the D-latch [1]

4 Edge-Triggered Flip-Flop

To address the second problem of **Unstable Output because of unstable input**, we put two D-Latches in series to make another device called the **Flip Flop**,

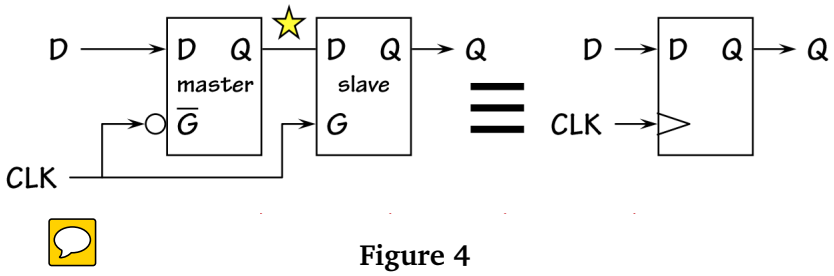


Figure 4

Things to note:

1. There is an inverter on the G input on the master flip flop
2. The diagram on the right is how flip-flop is drawn in books or on paper to save space

How it works:

1. The user gets output only from the output wire of the **slave** Q wire, and the user supplies input only to the **master** D wire.
2. Remember CLK periodically changes from 0 to 1 and vice versa
3. When CLK signal is 0, the G wire of **master** latch will receive a 1 (due to the inverter) and the G wire of **slave** flip flop will receive a 0.

[1] Recall that D-latch is must made of mux, that can be made of NAND gates

4. This means that the **master** latch is in ”write mode”, i.e: it lets signal from its D wire through to its Q wire, while the **slave** latch is in ”read memory mode”, i.e: slave’s output depends on *its own* memory wire Q’ and not affected by input on wire ⋆.

5. When CLK signal is 1, the G wire of **master** latch will receive a 0 due to the inverter and the G wire of **slave** latch will receive a 1.

6. This means that the **master** latch is in ”read memory mode”, i.e: master’s output depends on *its own* memory wire Q’ and not affected by input on wire D. Meanwhile, the **slave** latch is on ”write mode”, i.e: it lets signal from the ⋆ wire to be passed through its slave output.

7. Hence, **Only one of the two D-Latches is on ”write mode” at a time** or equivalently, only one D-latch is on ”memory-mode” at a time.

The explanation above is illustrated in terms of waveforms below:

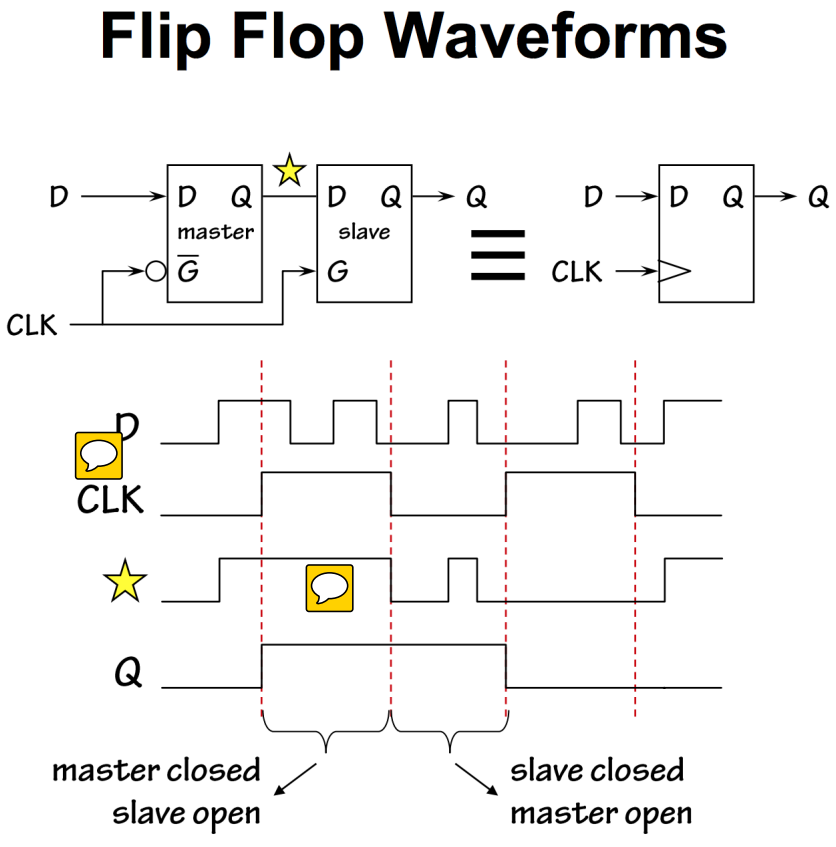


Figure 5

The name ”edge-triggered” comes from the fact that the output wire Q from the slave *changes* only when the CLK rises from 0 to 1.

5 Flip-Flop Timing Constraint

In the previous lessons, we learn about t_{CD} and t_{PD}. As a recap, for D-latch, these timings means:

1. t_{CD} of a D-latch is the time taken for **invalid** CLK input to produce an **invalid** output on wire G

2. t_{PD} of a D-latch is the time taken for **valid** CLK input to produce a **valid** output on wire G

For a single D-latch, we have the dynamic discipline to ensure that we do not end up storing invalid input signals. **Now that we cascade two D-latches to make a Flip-Flop, the dynamic discipline is translated into this timing constraint for Flip-Flop:**

$$t_{CD_{master}} > t_{hold_{slave}}$$

Explanation:

1. When signal at G in master changes from 1 to 0, and signal at G in slave changes from 0 to 1, master goes into read memory mode and slave goes into write mode.

2. Again, these CLK signals do not magically transform from 0 to 1 or 1 to 0. It has to **gradually** change to high voltage '1' or to low voltage '0', and it will crosses the invalid region (the region which voltage value doesn't translate to either digital 1 or 0).

3. Then $\star$ cannot change too quickly while G is transitioning, otherwise it will not meet the **hold time** [2] of the slave latch.

4. This means the contamination delay of the master latch [3] has to meet the hold time of the slave latch.

5. What about the setup time of the slave latch? We do not have to worry about that because signal $\star$ is taken care by the dynamic discipline (especially hold time) of the master latch [4], meaning that $\star$ is **always** stable before the clock edge changes from 0 to 1.

6. Note: Some of you may realise that the sum of the hold time of the master latch and the setup time of the slave latch cannot be lesser than the clock period. This is true. But typically we will add combinational logic between two flip flops (see next section) so the clock period will always be much longer than the $t_{hold_{master}} + t_{setup_{slave}}$, hence we do not even discuss this case.

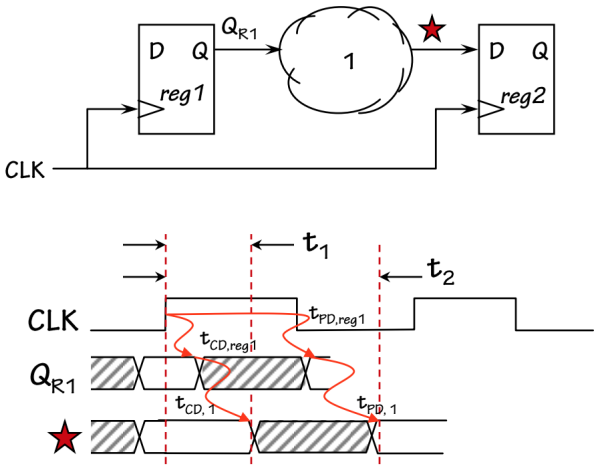
[2] Recall: time for $\star$, input to the slave latch, to be stable after the clock rise edge - i.e: clock changes from 0 to 1

[3] Time taken on signal on $\star$ is be invalid after G becomes invalid

[4] Recall dynamic discipline ensures that a latch will never end up going into read memory mode and storing invalid values

6 Sequential Logic Timing Constraint

We can now use Flip-Flop in our system as a 'memory' device that's cascaded before and after any combinational logic circuit. Due to dynamic discipline, we have two timing constraints called t_1 **and** t_2:



$$t_1 = t_{CD,reg1} + t_{CD,1} > t_{HOLD,reg2}$$

$$t_2 = t_{PD,reg1} + t_{PD,1} < t_{CLK} - t_{SETUP,reg2}$$

Figure 6

Note: The '1' in the picture above means any combinational logic unit, **and it is unrelated with the '1' in** t_1. The two boxes reg_1 and reg_2 are **both flip flops**, each made up of 2 D-latch. t_{CLK} means the clock period.

Explanation:

1. For t_1, it is basically saying that an "invalid" CLK signal that propagates from the input to $R1$ through combinational logic 1 should not arrive 'too soon' at R2 because R2 requires the signal $\star$ for as long as t_{hold} in order to latch into the valid values during write mode.

2. For t_2, it ensures that the things that has to be done between two clock edges are the propagation delay of R1, the propagation delay of the combinational logic CL1, and the setup time for R2 in order to go to write mode and store the value value properly.

Look at t_2 constraint. The time taken to do *actual work* at the combinational logic unit '1' is therefore,

$$t_{work_1} = t_{PD_1} = t_{CLK} - t_{setup_{R2}} - t_{pd_{R1}}$$

7 Timing Example

Below is one sequential timing example: Explanation:

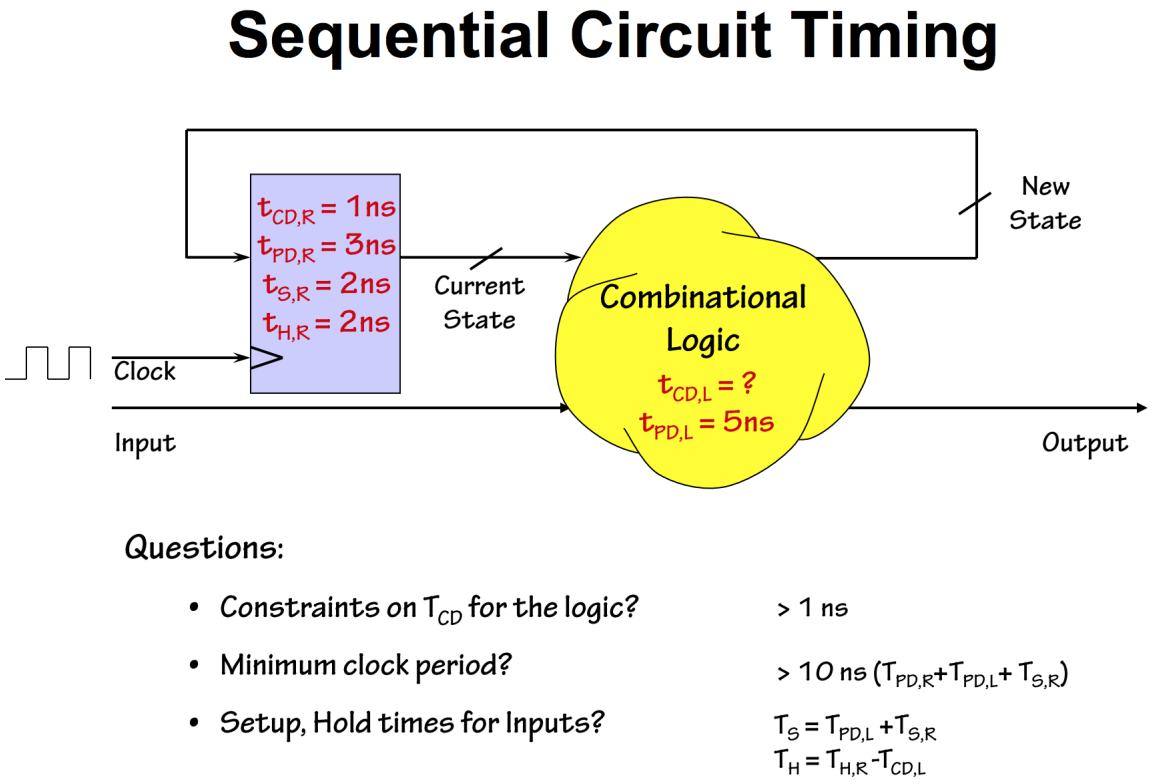


Figure 7

1. According to t_1 in the previous section, the contamination delay of the combinational logic CL has to be larger than the hold time of the register R2, and the setup time of the register R1.
2. Here, R1 = R2 because of the feedback path, hence the contamination delay of CL is at least $t_{H,R} - t_{CD,R} = 2 - 1 = 1$.
3. **Note that symbols T_S and T_H in flip-flop timings like this simply means the setup and hold time required for the input at the input wire to be stable**

8 Synchronization

In sequential logic, **we use a single synchronous clock**, meaning that we use one same clock to any flip-flop in the computer. Recall that dynamic discipline is crucial for any sequential logic to work properly. We are now going to address two problems of synchronisation that can violate dynamic discipline:

1. There's dynamic discipline that make sure that the input D has to meet the setup and hold time of the latches. But in real world, user does not always supply input that is synchronous with the clock edge. This carries the risk of violating dynamic discipline and may end up storing the invalid values in read memory mode. This invalid value storing is called the **metastable state**.

2. Clock skew: The clock does not arrive at all registers at the same time and skew is the difference between two clock edges. This may also potentially violate dynamic discipline.

The next two sections address these problems.

9 The Metastable State

The metastable state properties are illustrated below:

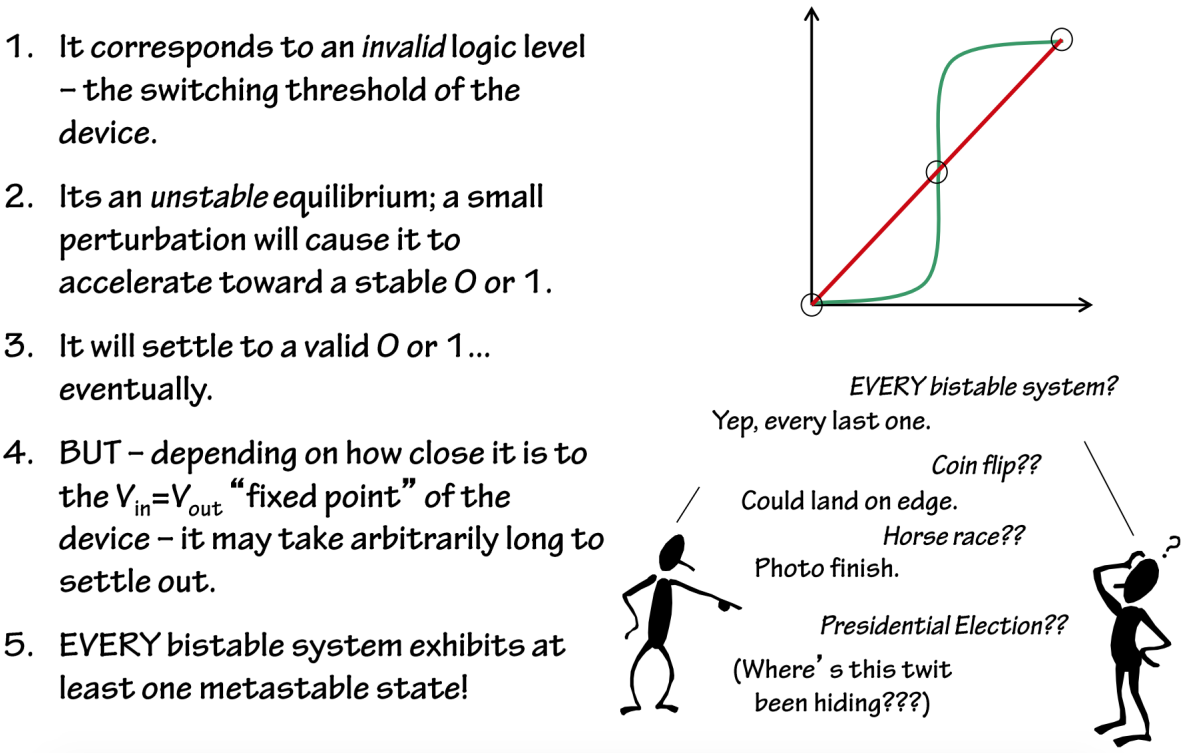


Figure 8

And the reason behind metastable state is:

The Metastable State:

Why is it an inevitable risk of synchronization?

- Our active devices always have a fixed-point voltage, V_M, such that $V_{IN}=V_M$ implies $V_{OUT} = V_M$
- Violation of dynamic discipline puts our feedback loop at some voltage V_O near V_M
- The rate at which V progresses toward a stable "O" or "1" value is proportional to $(V - V_M)$
- The time to settle to a stable value depends on $(V_O - V_M)$; its theoretically infinite for $V_O = V_M$
- Since there's no lower bound on $(V_O - V_M)$, there's no upper bound on the settling time.
- Noise, uncertainty complicate analysis (but don't help).

Figure 9

In other words, we cannot completely avoid the metastable state. The only thing we can do is to *minimize its probability*. We can do that by introducing more **delays** between input and output in the hopes that the signal will somehow resolve itself before reaching the output, like illustrated here,

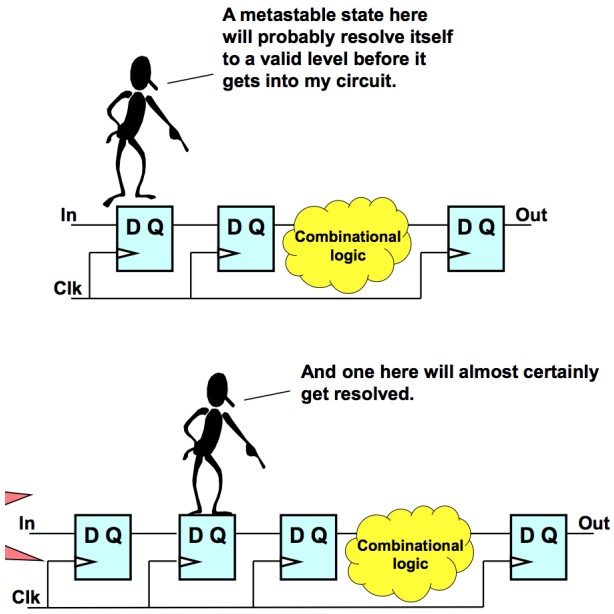


Figure 10

10 Clock Skew

To fix clock skew, we need to know t_{skew}, which is the maximum difference in clock signal arrival times across all flip-flops in the computer. We just need to adjust the two rules t_1 and t_2 for flip flops into,

$$t_1 = t_{CD,Reg1} + t_{CD,Reg1} > t_{hold,Reg2} + t_{skew} \tag{1}$$

$$t_2 = t_{PD,Reg1} + t_{PD,CL1} + t_{SETUP,Reg2} < t_{CLK} + t_{skew} \tag{2}$$