EE-421: Digital Systems Design

Metastability - When A Good Flip-Flop goes Bad:

From BiStables to Metastable States

"Causes and Cure"

Dr. Rehan Ahmed [rehan.ahmed@seecs.edu.pk]

NUST

SCHOOL OF ELECTRICAL ENGINEERING &
COMPUTER SCIENCE (SEECS)

INTEGRATED
CIRCUITS
DESIGN
LAB

Learning Objectives

- Understand what metastability is, and how it can cause system failure

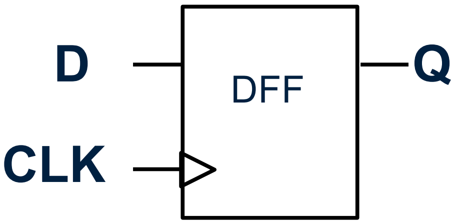
- Understand why metastability happens at the first place

- Be able to design a circuit to reduce the probability that metastability causes system failure

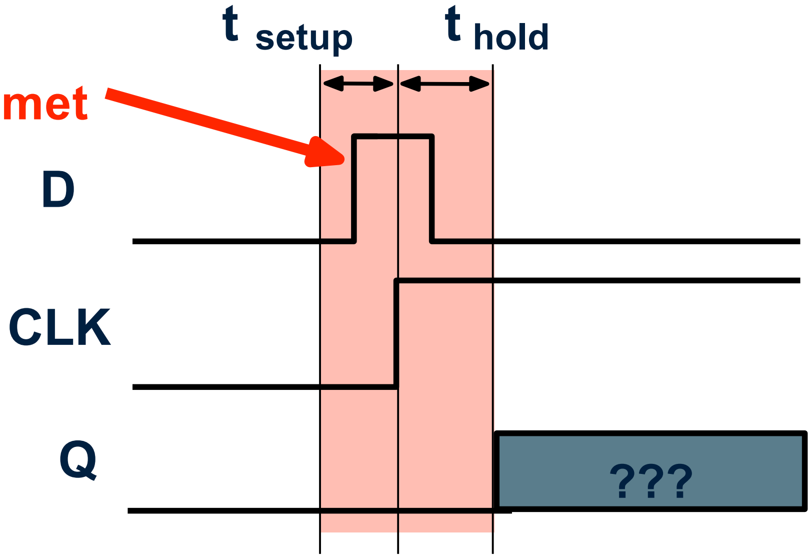
- To be able to calculate the mean time between failure due to metastability

What happens if we violate a flip-flop’s setup and/or hold time requirements?

Violating Flip-Flop Setup/Hold Times (1)



Setup/Hold Requirement not met

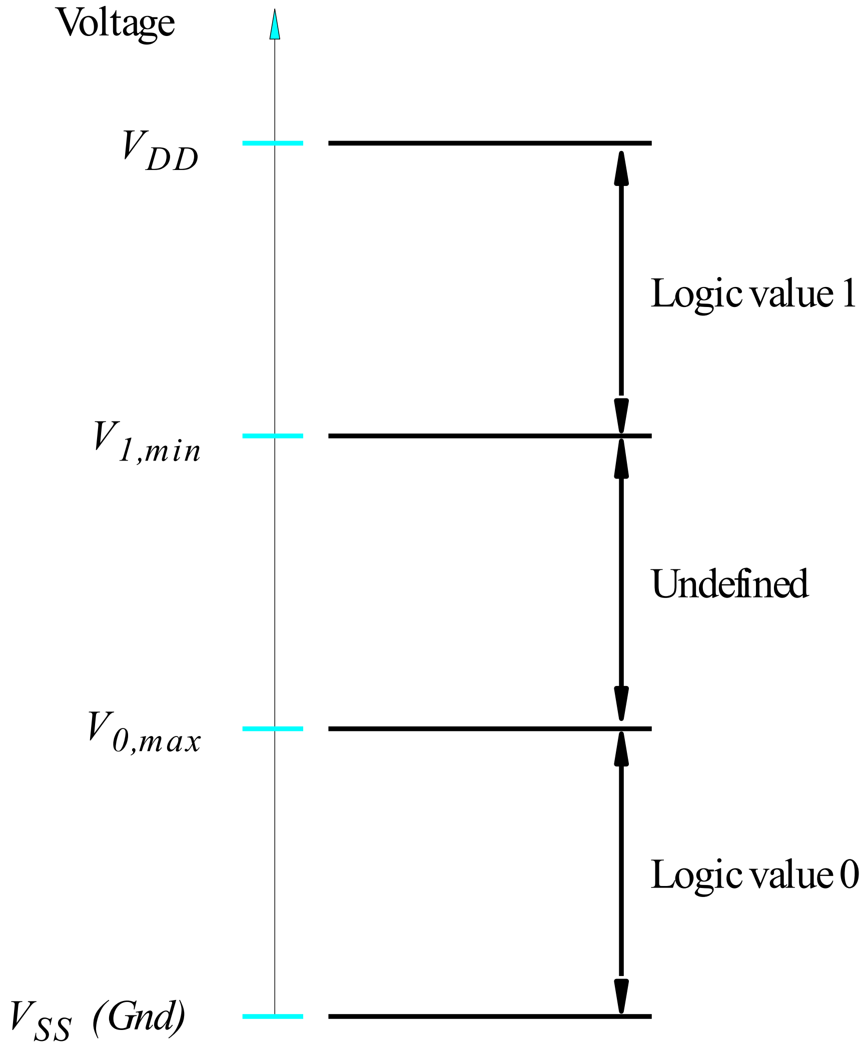


Possible Outcomes: Q may…

- get the correct value
- get the wrong value (0 or 1)
- become **metastable** - get a value between 0 and 1
  - *Remember, in reality, we are working with voltages which can take on a continuous range of values*

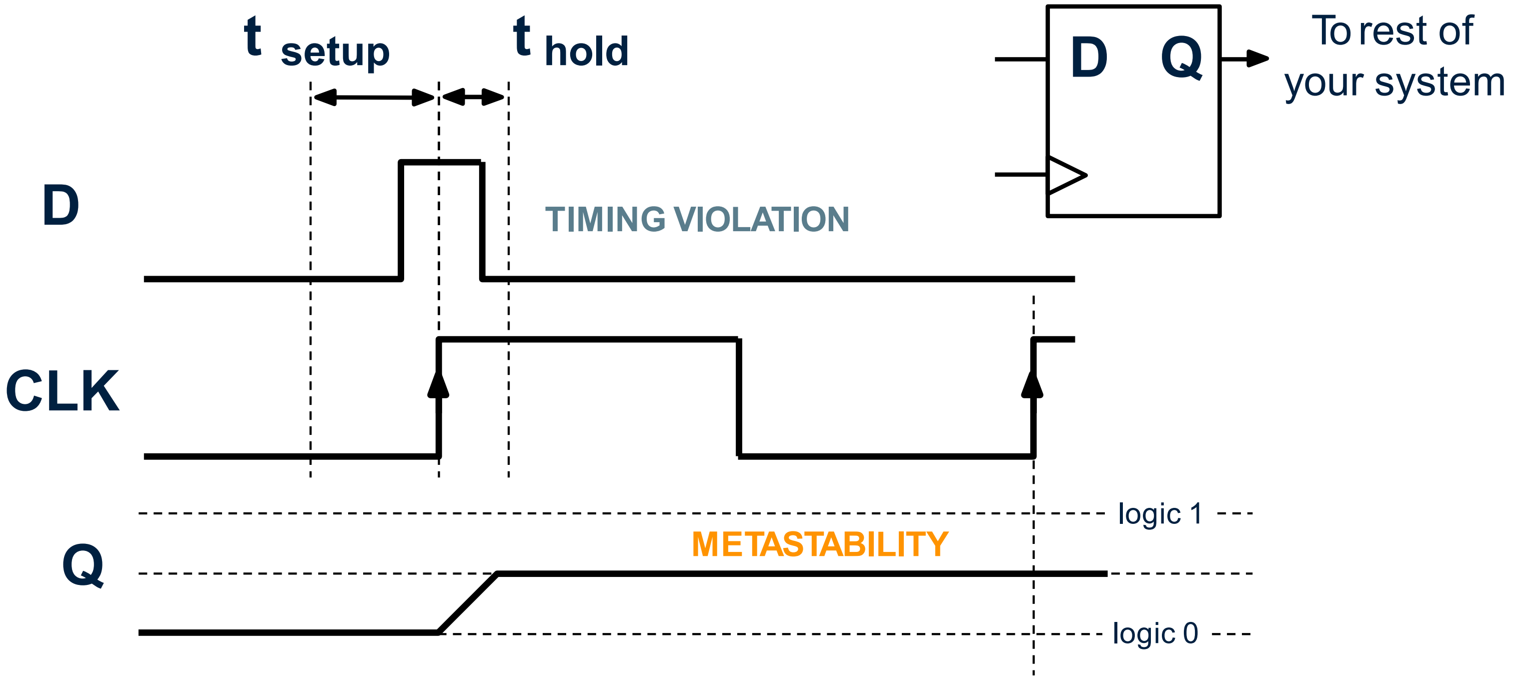
Violating Flip-Flop Setup/Hold Times (2)

- **Metastable** - a value between 0 and 1



Violating Flip-Flop Setup/Hold Times (3)

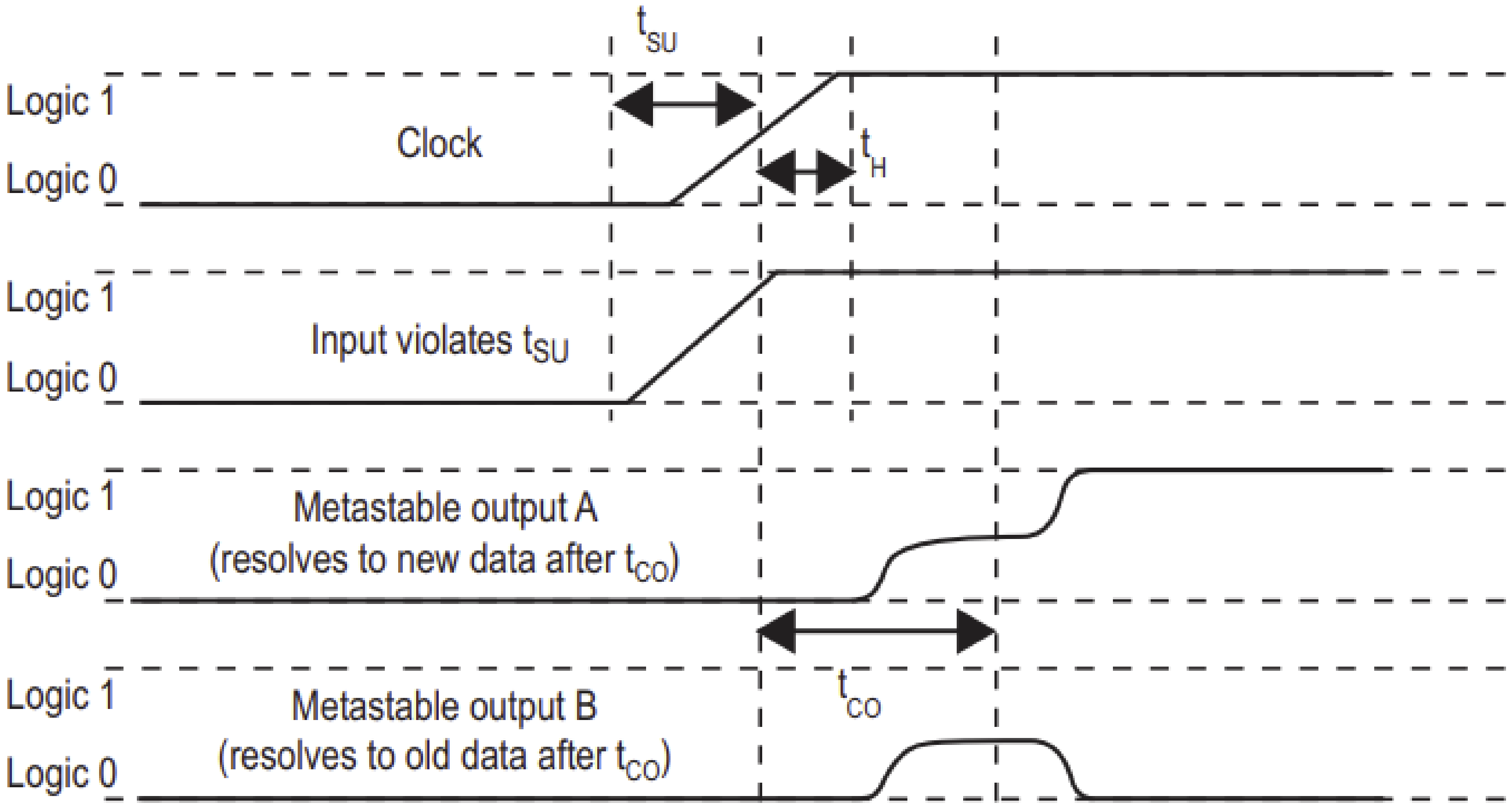
- Metastability



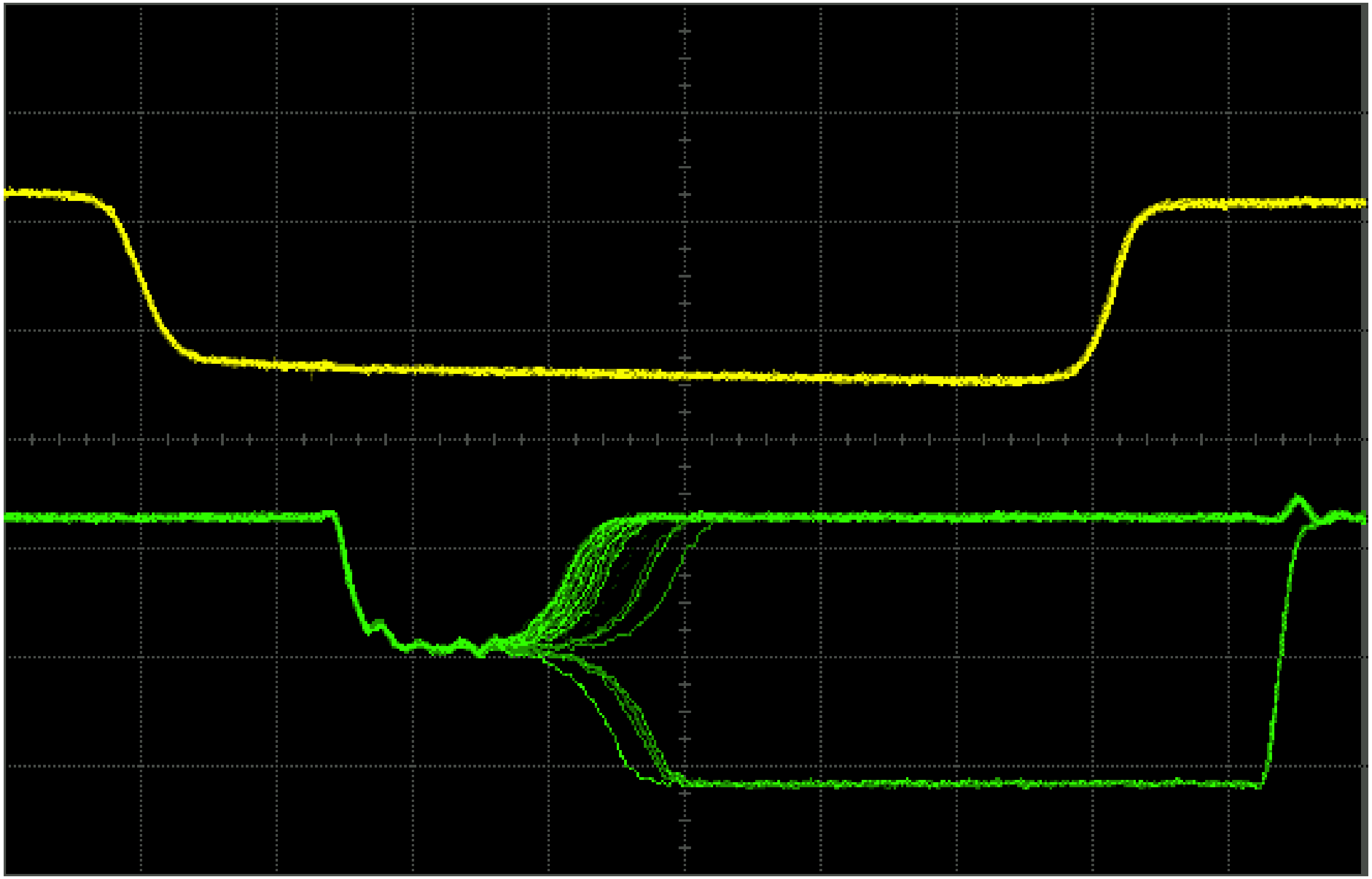
METASTABLE VALUE
PROPOGATED TO OTHER
PARTS OF SYSTEM

MAY CAUSE SYSTEM-WIDE FAILURE

Example of Metastable Signals

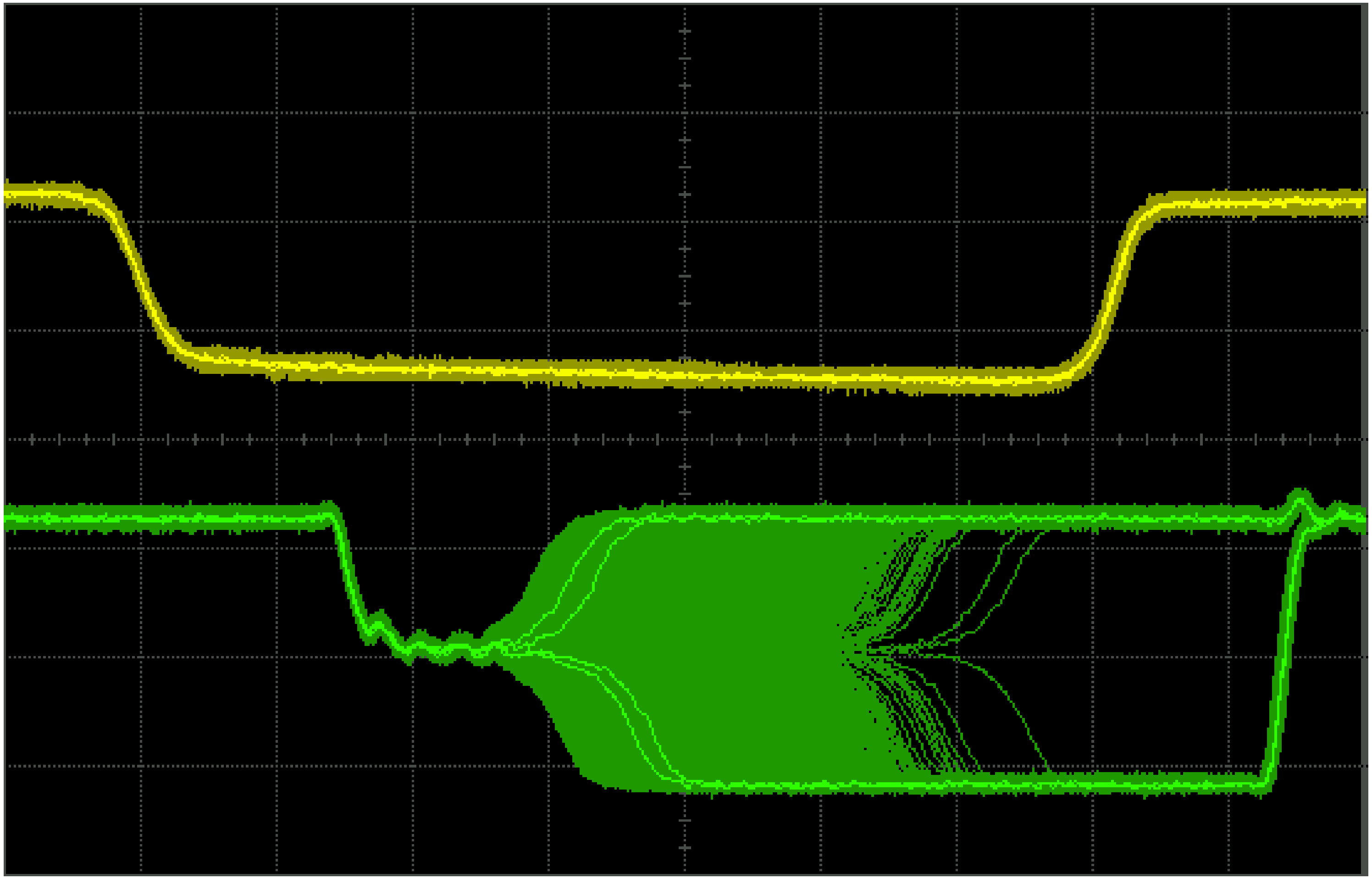


Real Metastable Example



Picture taken from W. J. Dally, Lecture notes for EE108A, Lecture 13: Metastability and Synchronization Failure (ow When Good Flip-Flops go Bad) 11/9/2005.

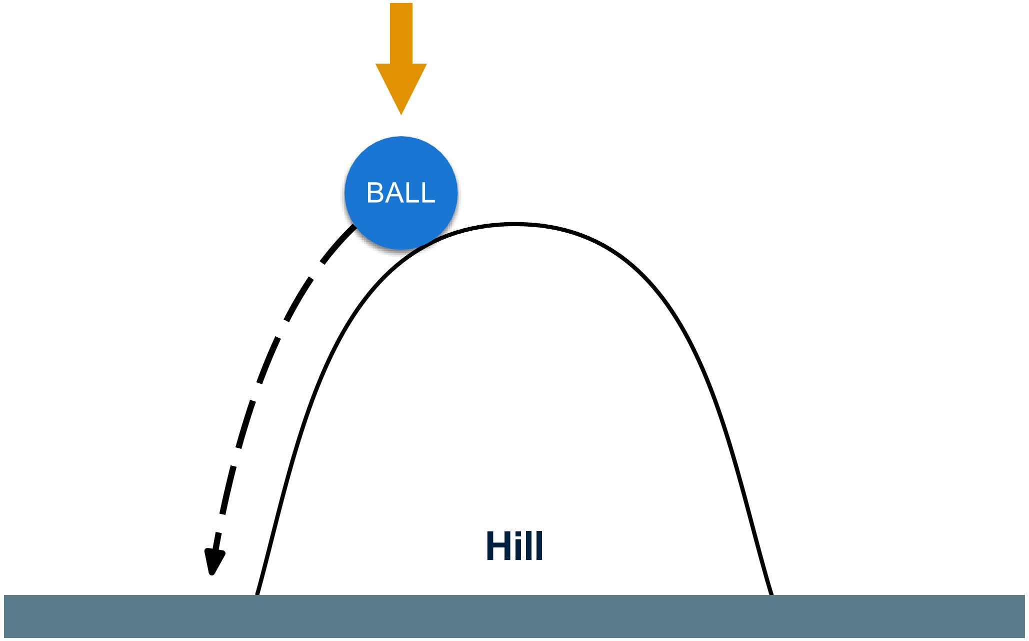
Real Metastable Example



Picture taken from W. J. Dally, Lecture notes for EE108A, Lecture 13: Metastability and Synchronization Failure (ow When Good Flip-Flops go Bad) 11/9/2005.

Let’s Understand Metastable State through a Mechanical Analogy

Mechanical Analogy



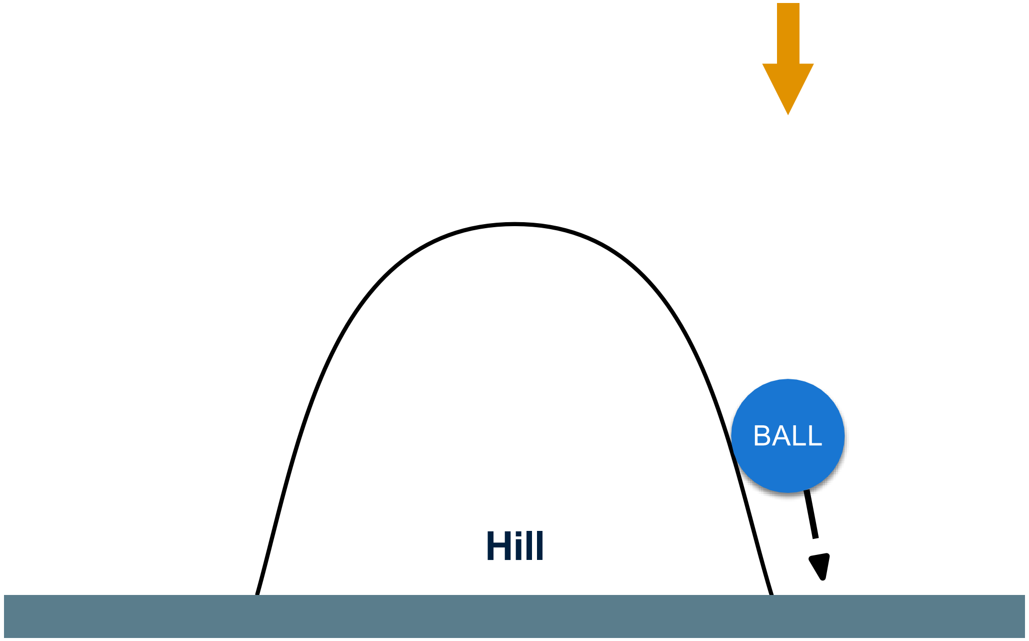
Imagine dropping a ball on a hill

- perfectly smooth and symmetrical hill
- no forces except gravity

Depending on where you drop the ball, it will settle in one of two states

- either on the left side or the right side

Mechanical Analogy

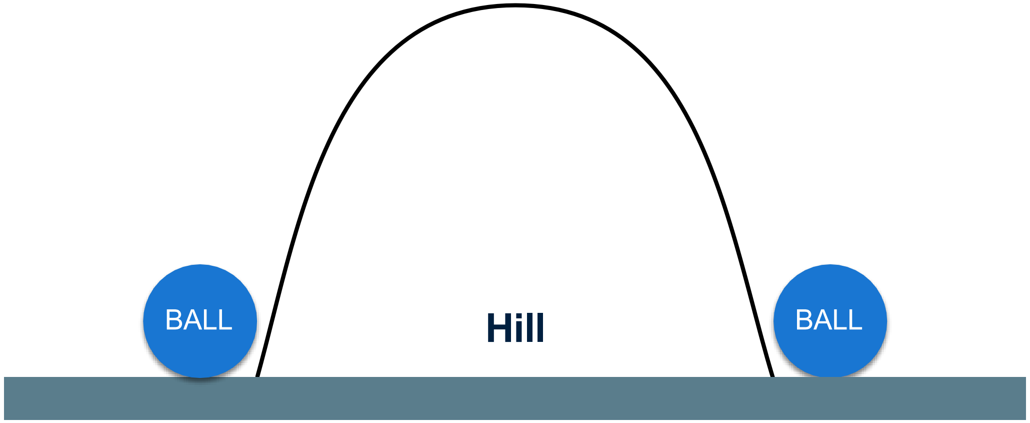


The closer you drop the ball near either side, the faster it settles

- Shorter distance
- Steeper slope

Mechanical Analogy

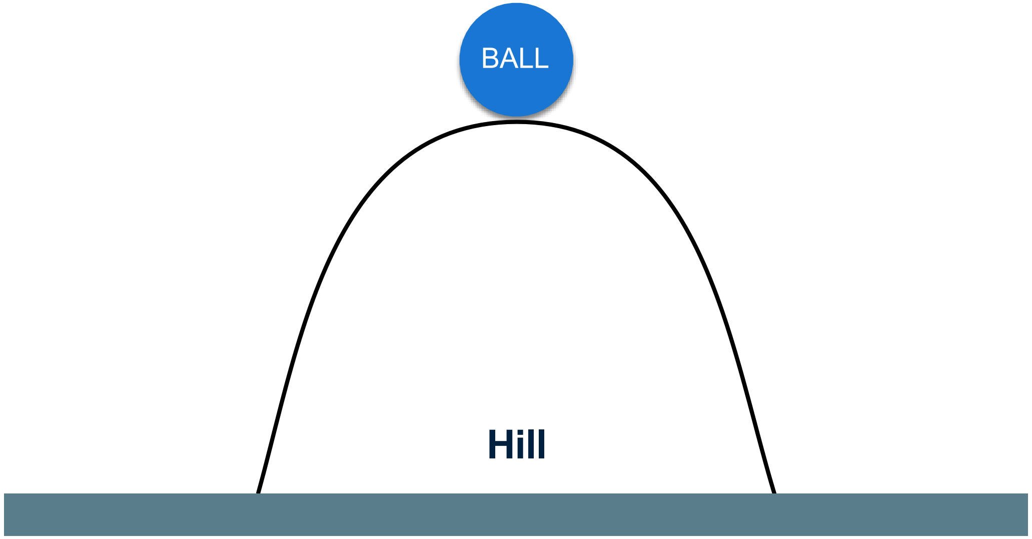
Two Stable States



Once ball reaches either state, it will stay there forever

Mechanical Analogy

There is a THIRD state: “Metastable State”!



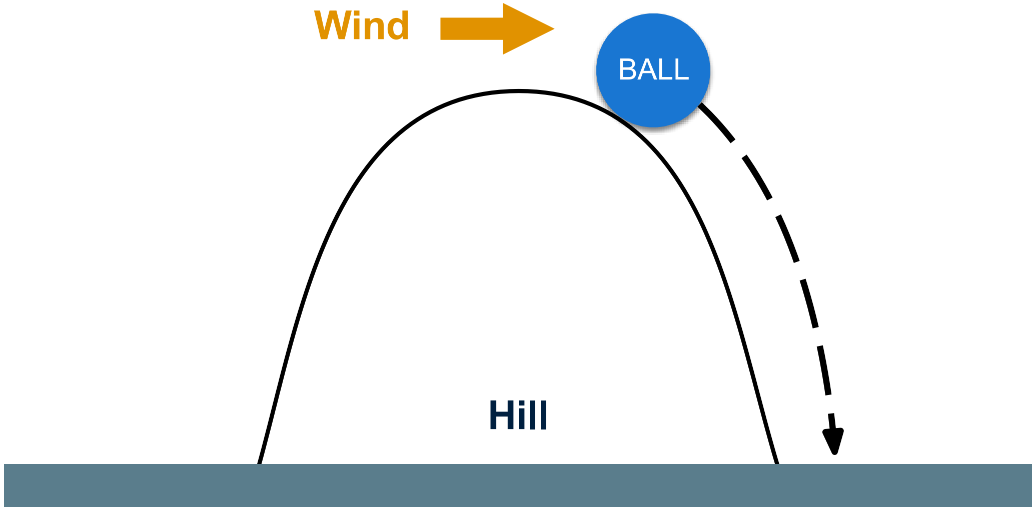
Imagine the ball is dropped perfectly in the middle

The slope at the exact middle is flat

There are no other forces besides gravity

In theory, ball will stay there forever

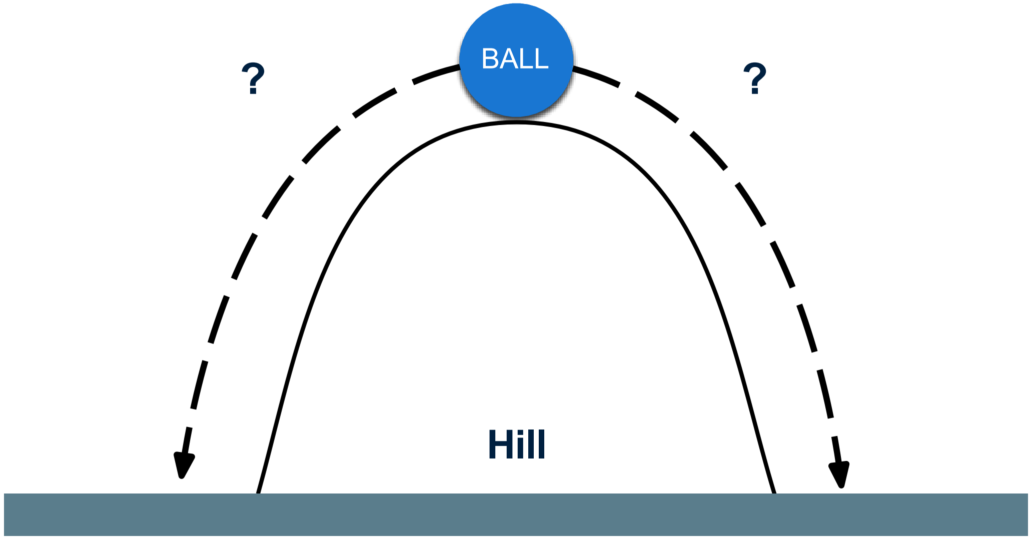
Mechanical Analogy



In practice, ball will eventually settle to a stable state

- Gust of wind
- Non-perfect hill
  - Not perfectly symmetrical
  - Slope not perfectly flat in the middle

Mechanical Analogy



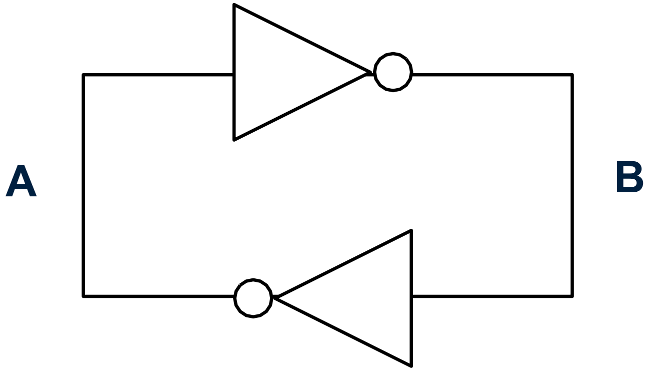
Into which state will it settle?

How long before it settles?

Not predictable!

OK, Got it! But how does this apply to flip-flops?

A Theoretical Storage Element



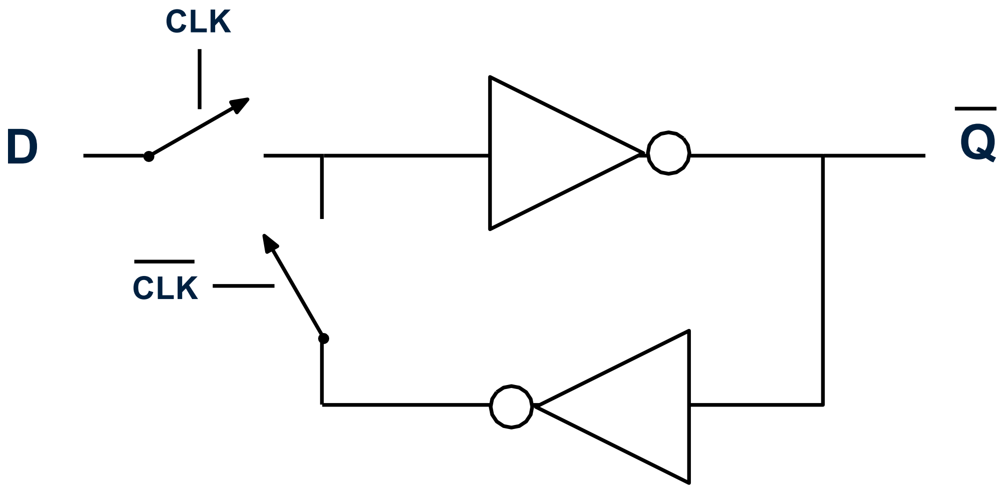
Two Stable States

- A=0, B=1
- A=1, B=0

This element isn't very practical (no way to change state)

But **ALL** state-holding digital circuits* are based off of using positive-feedback loops

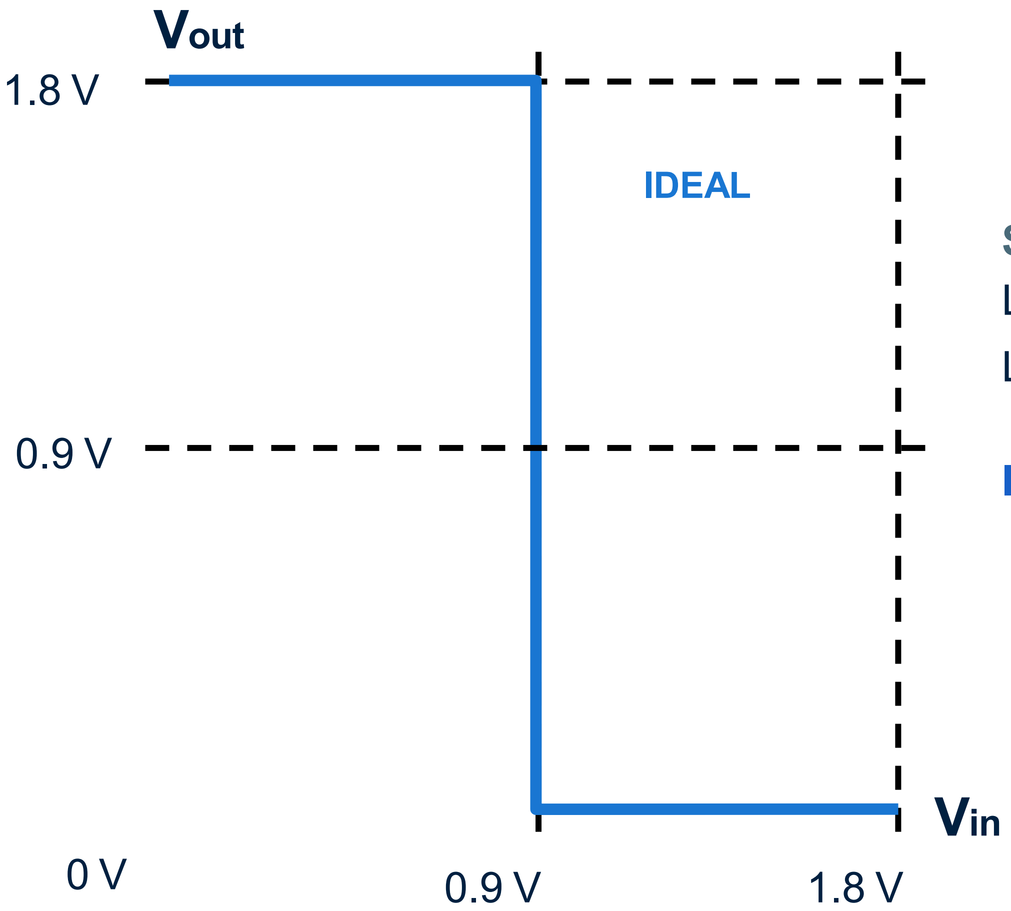
Preview – Possible Topology for a D-Latch

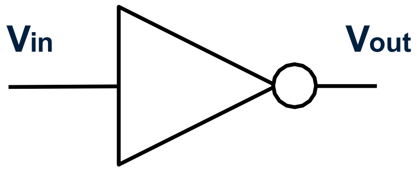


When CLK is 1, Q’ follows D (but inverted)

When CLK is 0, D is disconnected, feedback loop is connected to reinforce “stored” value, Q' does not change with D

Inverter Voltage Transfer Curve (VTC)





Say that in our technology:

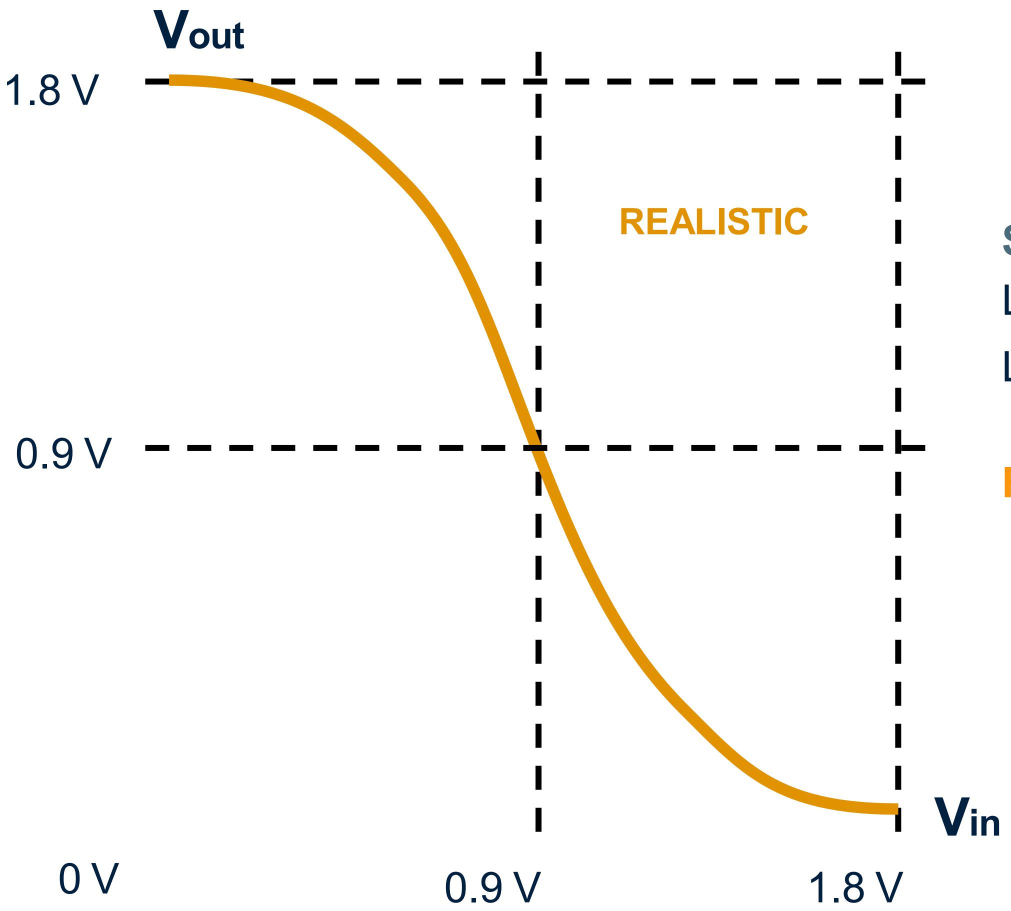
Logic 1 is 1.8V

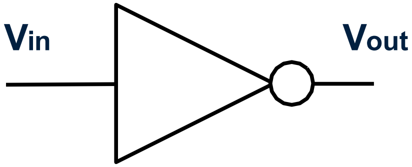
Logic 0 is 0V

IDEAL Behaviour:

$$V_{out} = \begin{matrix} 1.8\,V, & V_{in} < 0.9V \\ 0\,V, & V_{in} >= 0.9V \end{matrix}$$

Inverter Voltage Transfer Curve (VTC)





Say that in our technology:

Logic 1 is 1.8V

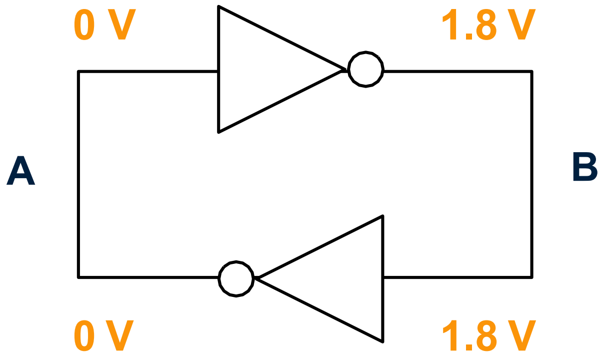
Logic 0 is 0V

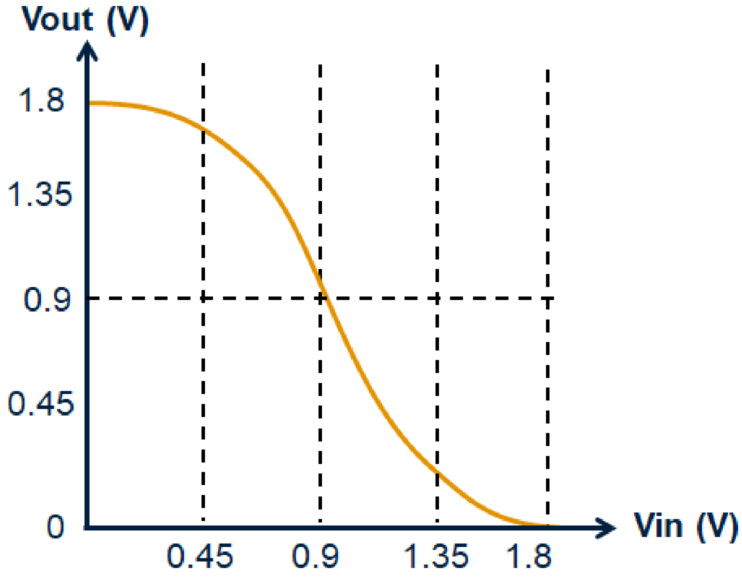
REALISTIC Behaviour

$$V_{out} = f\left(V_{in}\right)$$

Storage Element – Scenario 1

Say we externally drive A to **0 V**



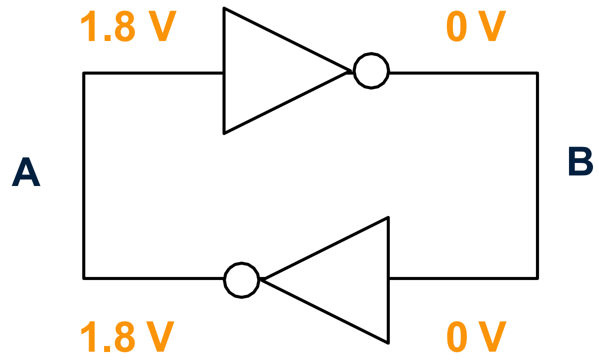


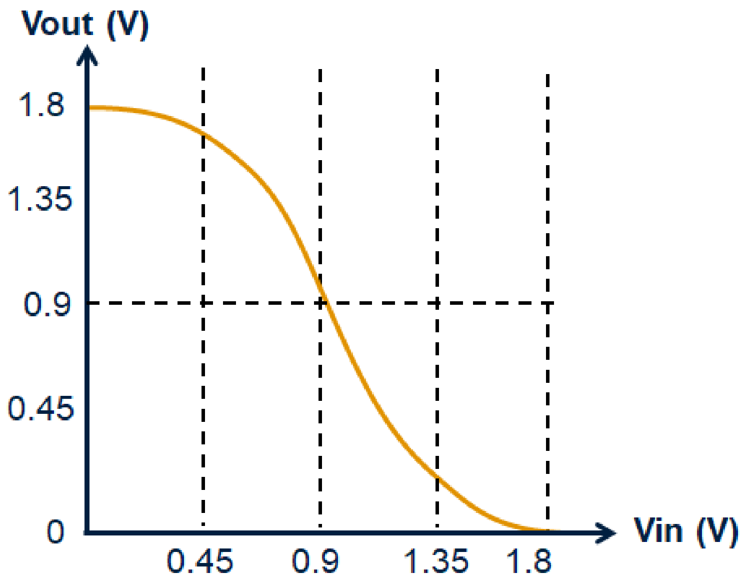
1. B gets driven to 1.8 V by the top inverter
2. A then gets driven to 0 V by the bottom inverter (reinforced)
3. which then causes B to be driven to 1.8 V (reinforced)
4. …

Positive feedback **INSTANTLY** reinforces Internal value.

Storage Element – Scenario 2

Say we externally drive A to **1.8 V**





1. B gets driven to 0 V by the top inverter
2. A then gets driven to to 1.8 V by the bottom inverter (reinforced)

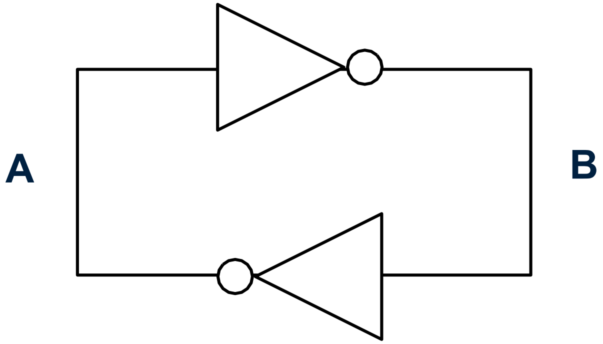
. which then causes B to be driven to 0 V (reinforced)

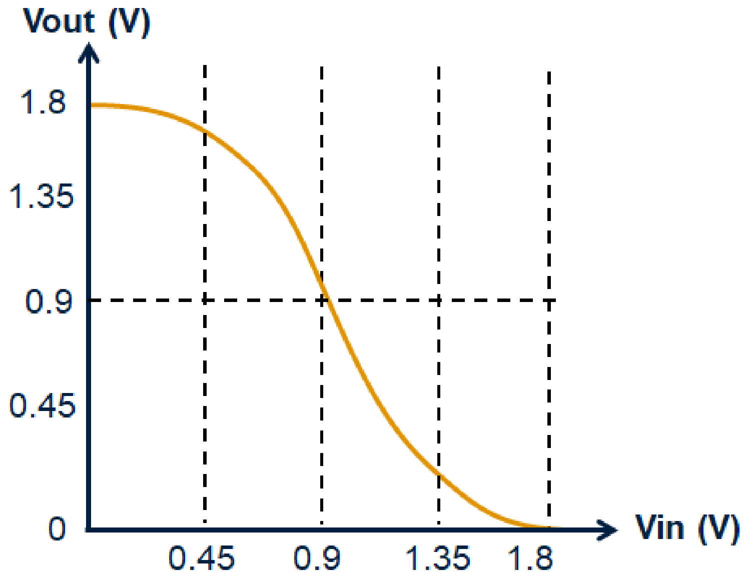
...

Positive feedback **INSTANTLY** reinforces Internal value.

Storage Element – Scenario 3

Say we externally drive A to **1.35 V**



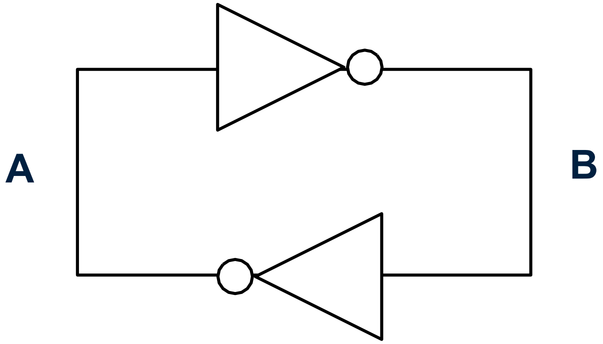


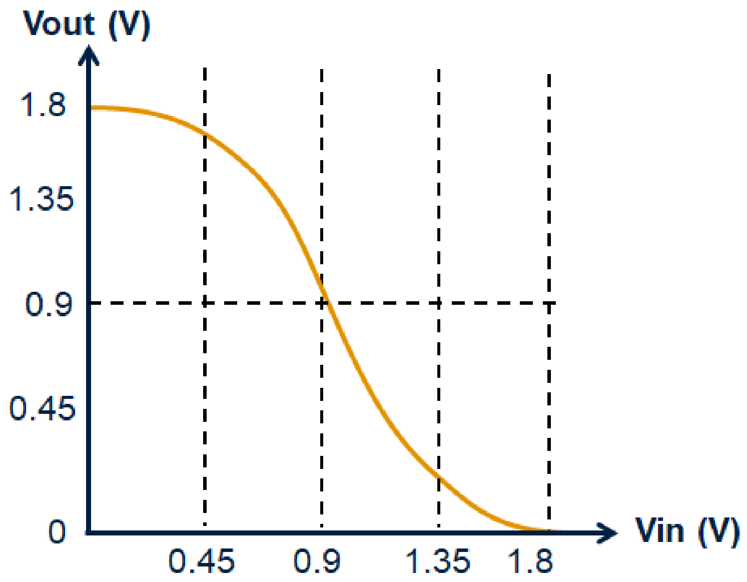
1. B gets driven to 0.2 V by the top inverter
2. A then gets driven to to 1.75 V by the bottom inverter
3. which then causes B to be driven to 0.01 V
4. …

Positive feedback **VERY QUICKLY** settles Internal value.

Storage Element – Scenario 4

Say we externally drive A to **1 V**



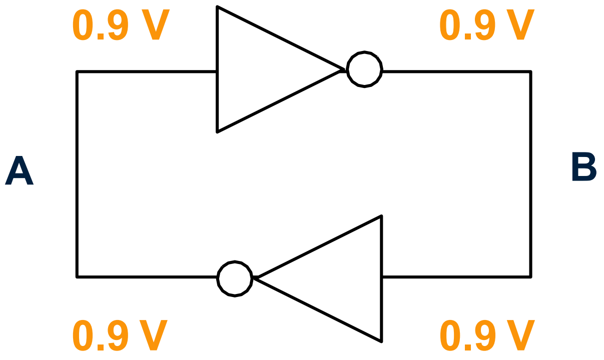


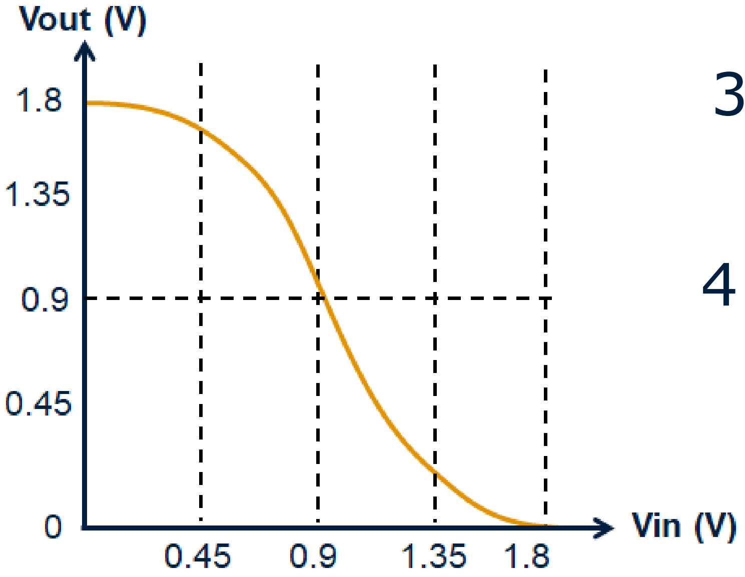
1. B gets driven to 0.75 V by the top inverter
2. A then gets driven to to 1.5 V by the bottom inverter
3. which then causes B to be driven to 0.2 V
4. …

Positive feedback **TAKING LONGER** now to settle internal value.

Storage Element – Scenario 5

Say we externally drive A to **0.9 V**

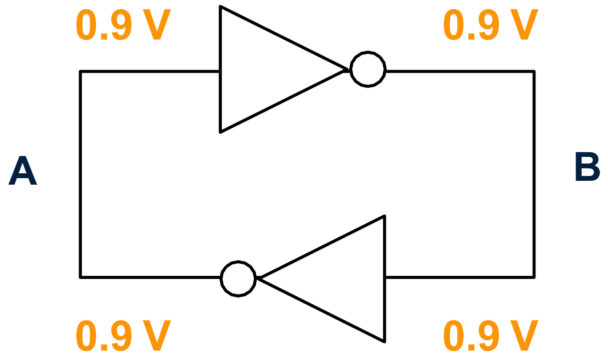




1. B gets driven to 0.9 V by the top inverter
2. A then gets driven to to 0.9 V by the bottom inverter
3. which then causes B to be driven to 0.9 V
4. …

STUCK IN METASTABLE STATE!

Storage Element – Metastable State



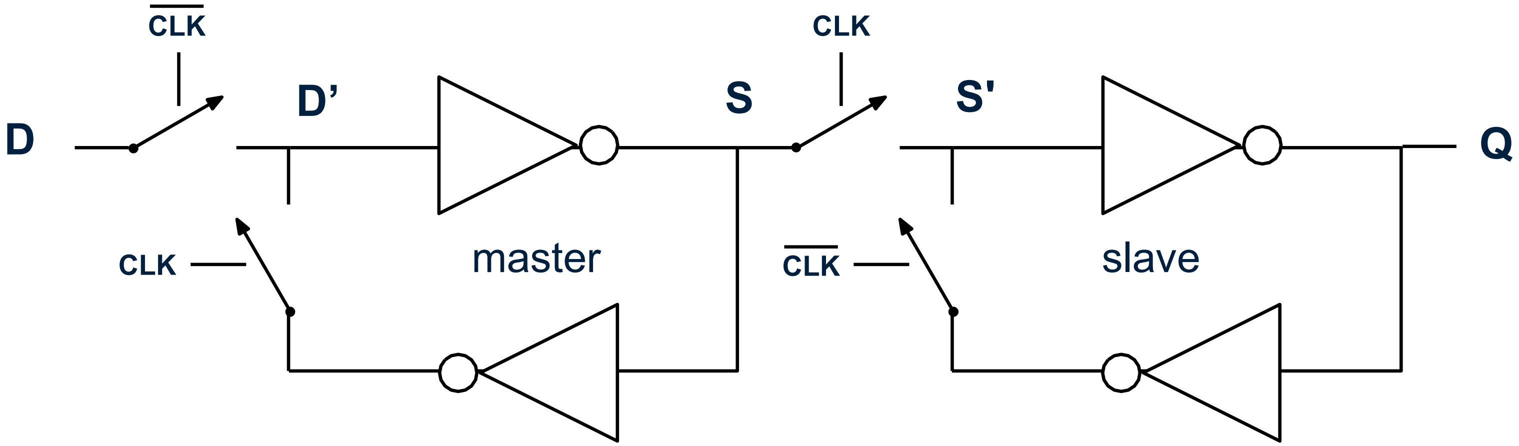
In reality, there are sources of noise:

- Thermal, crosstalk, quantum transistor effects
- Noise is like the “gust of wind” from mechanical example

NOISE will push the circuit out of metastability

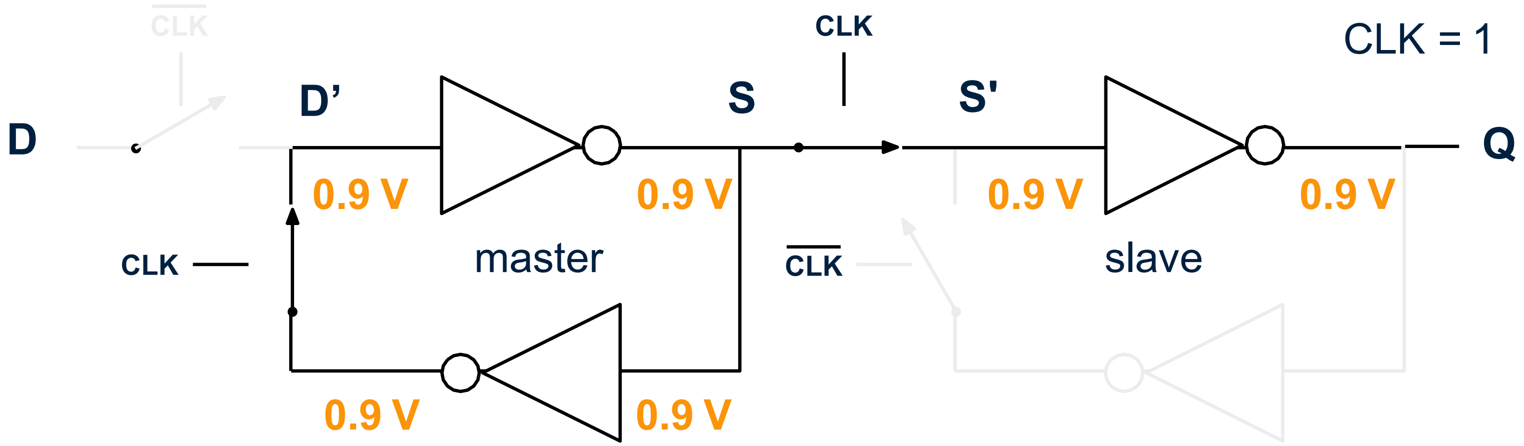
- But we won’t know how long this will take
- Nor will we know which of the two stable states it will assume

D Flip-Flop: Master-Slave Topology



Built using two D-Latches

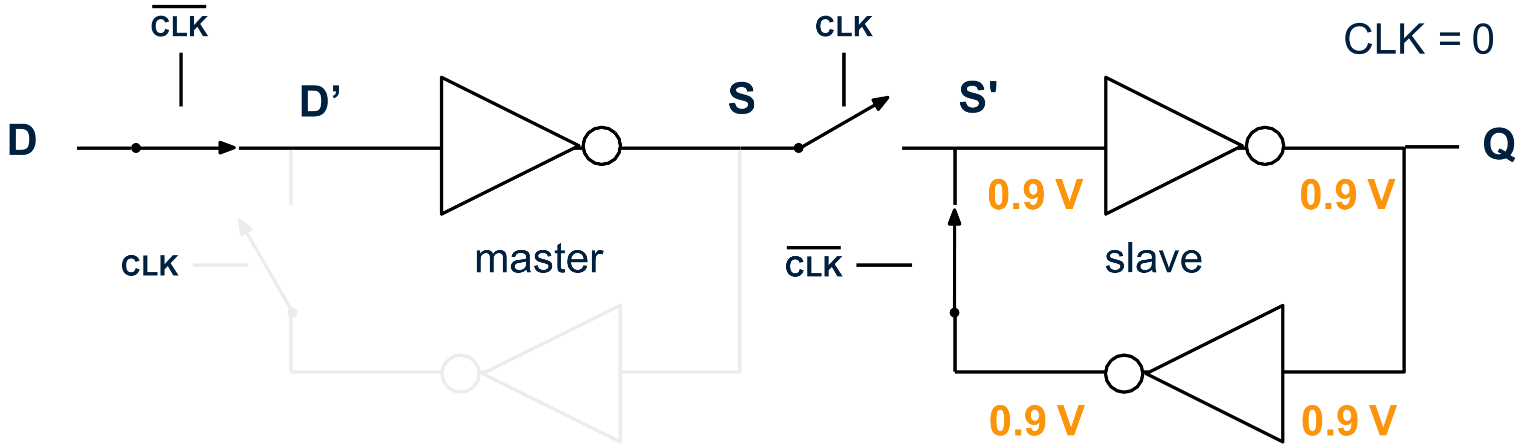
Violating Setup or Hold Time Requirements in a D-Flop (1)



Risks putting 0.9V onto S or D'

- We know that the master feedback look could be stuck in metastable state for unknown amount of time
- Will also put slave into metastable state
- Q would then be metastable

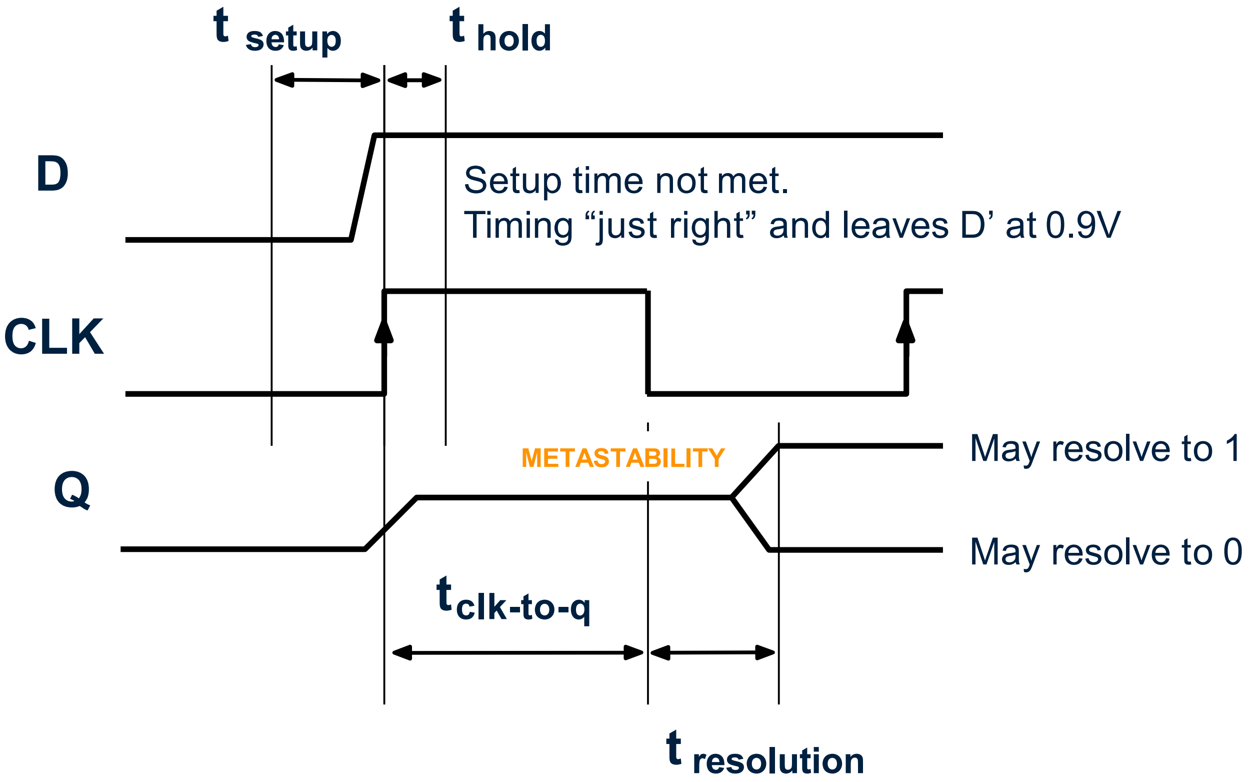
Violating Setup or Hold Time Requirements in a D-Flop (2)



After falling edge

- Slave feedback gets stuck in metstable state
- Q still metastable

Metastability: Violating Setup or Hold Time

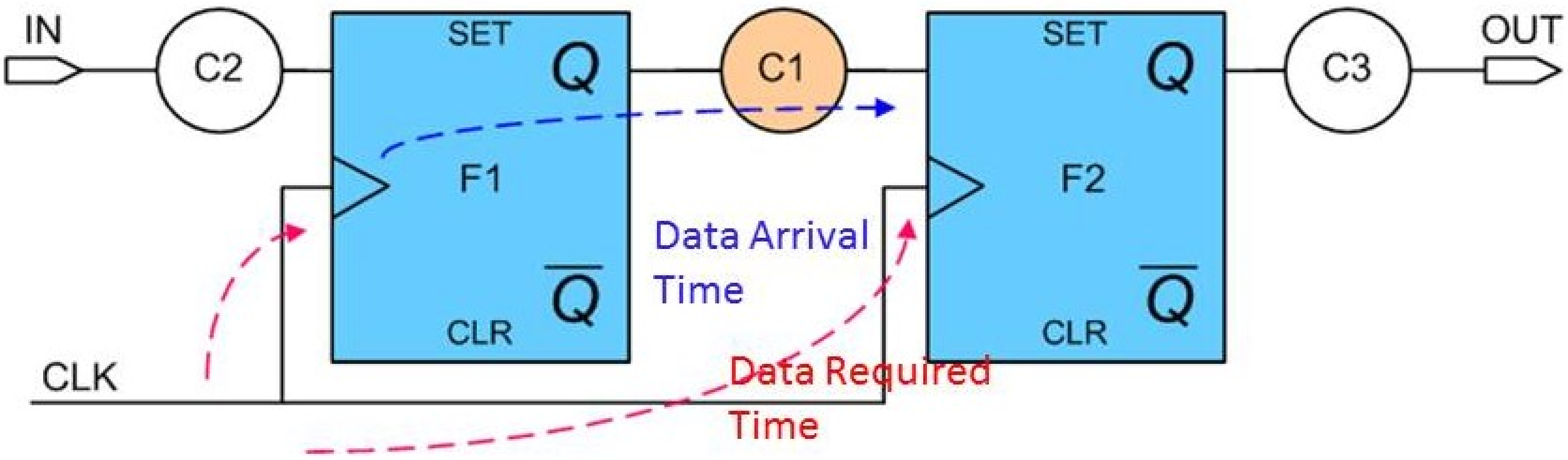


Takes extra time for Q to settle

Resolution time is UNBOUNDED

Path Slack

Path Slack

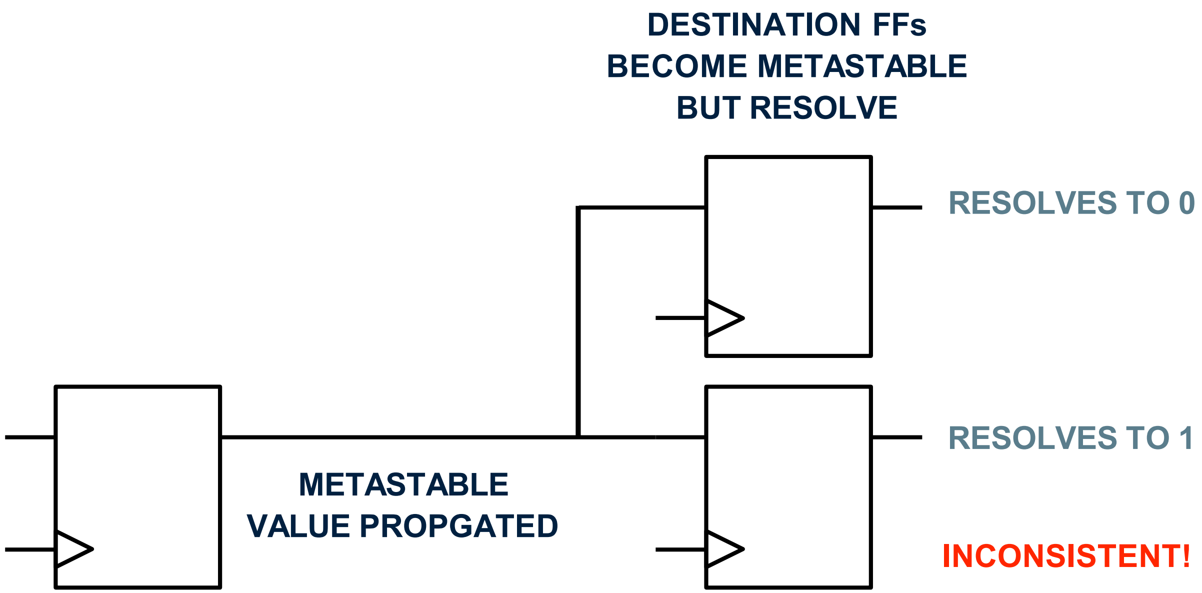


- Slack is defined as difference between actual or achieved time and the desired time for a timing path:
  - A +ve slack means design is working at the specified frequency and it has some more margins as well.
  - Zero slack specifies design is exactly working at the specified frequency and there is no margin available.
  - -ve slack implies that design doesn't achieve the constrainted frequency and timing. Timing Violation!

Slide Credit: http://asic-soc.blogspot.com/2013/08/setup-and-hold-slack.html

If $t_{resolution}$ is **GREATER** than the amount of path slack, then system failure!

Why System Failure?



If metastable value is propagated, FFs at next stage will become metastable and may…

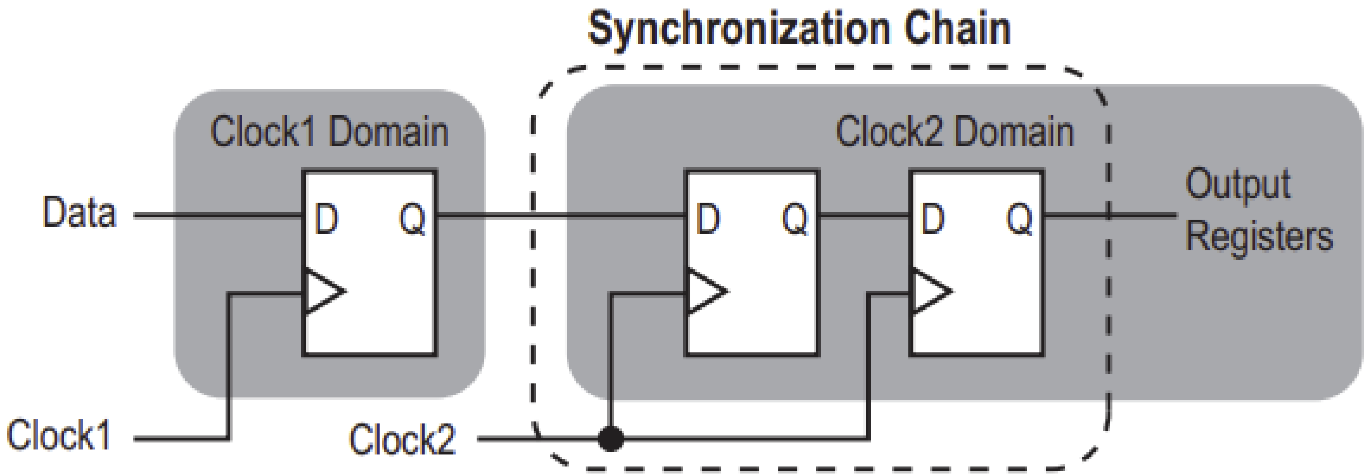
- further propagate metastable value
- or resolve inconsistently
  - different destinations FFs see different values for the same signal

Bottom Line:

Don’t change flip-flop input too close to clock edge

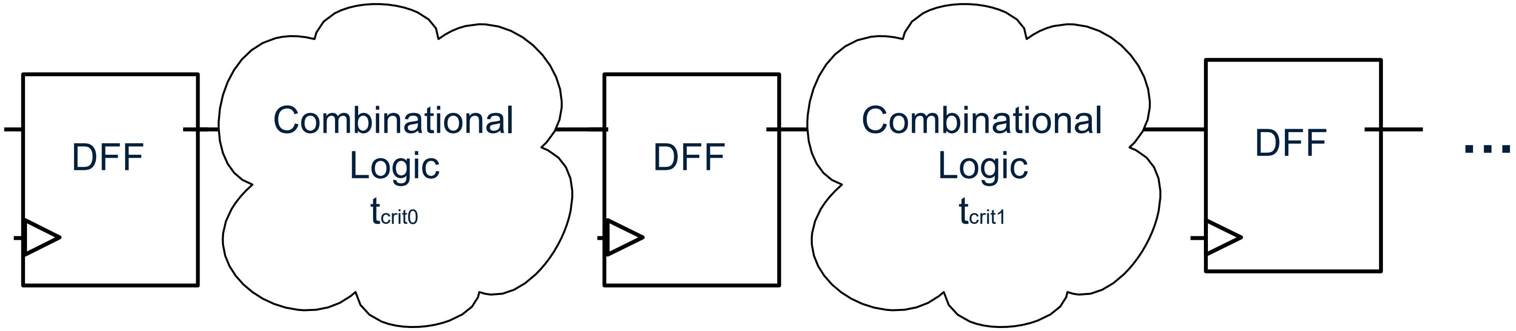
Dealing with Metastability …

Synchronizer Register Chain



- **Synchronizer Circuit**
  - Two flip-flops instead of one i.e two-bit shift register
  - If output of the first flop goes metastable, it has an entire clock-cycle to resolve itself and appear correctly at the second flop
- This is a VERY common technique – add synchronizers to all async interfaces

Asynchronous Signals



The whole point of synchronization:

Within our circuit, make sure both setup and hold requirements are always met for all paths

↠ We can eliminate metastability inside of our circuit

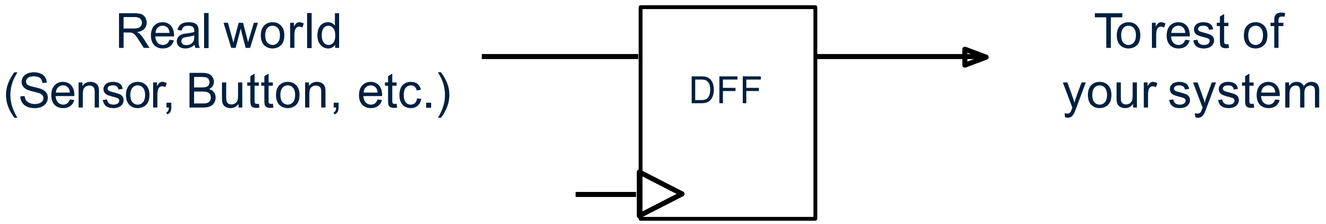
But we often need to interface with **asynchronous** signals

Asynchronous Signals

Asynchronous just mean that it doesn’t adhere to the clock signal

For Example: Real World Signals

- Real world signals, like buttons, don’t change aligned to your clock
- Non-zero chance of input changing too close to clock edge



Real world
(Sensor, Button, etc.)

DFF

To rest of
your system

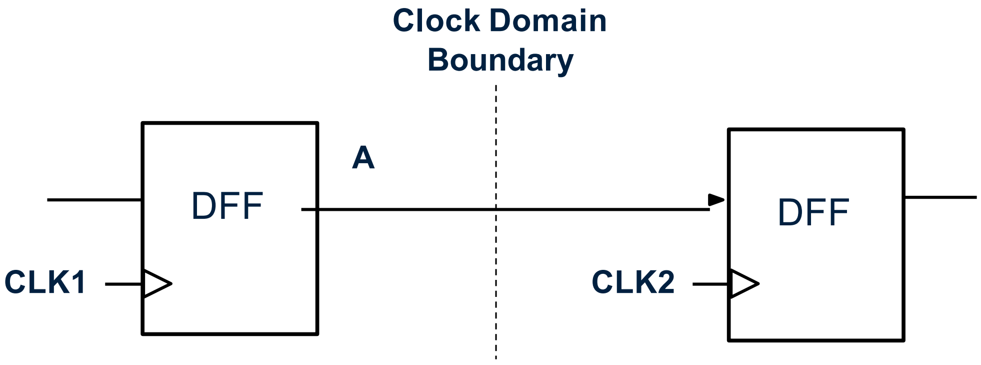
Asynchronous Signals

Another Example: Systems With Multiple Clocks

- Many systems have more than one clock
- Flip-flops still only listen to one clock
- Logic/FFs on the same clock belong to the same **clock-domain**

Signals sometime need to cross clock-domain boundaries

If the clocks are unsynchronized (not multiples of each other), then the signal crossing the clock-domain appears asynchronous



Signal A appears asynchronous to DFF2

Mean-Time Between Failure (MTBF)

Mean-time Between Failure (MTBF)

Metric to estimate the average time between two failure-causing instances of metastability on a given signal transfer

$$MTBF = \frac{e^{t_{slack}/C_0}}{C_1 \cdot f_{CLK} \cdot f_{DATA}}$$

- f_{CLK} – Clock frequency
- f_{DATA} – Toggling frequency of the FF input
- t_{slack} – Amount of slack available on the path for metastability resolution
- C_o, C_1 – Constants dependent on operating conditions and process tech

Mean-time Between Failure

Metric to estimate the average time between two failure-causing instances of metastability on a given signal transfer

$$MTBF\left(t_{slack}\right) = \frac{e^{t_{slack}/C_0}}{C_1 \cdot f_{CLK} \cdot f_{DATA}}$$

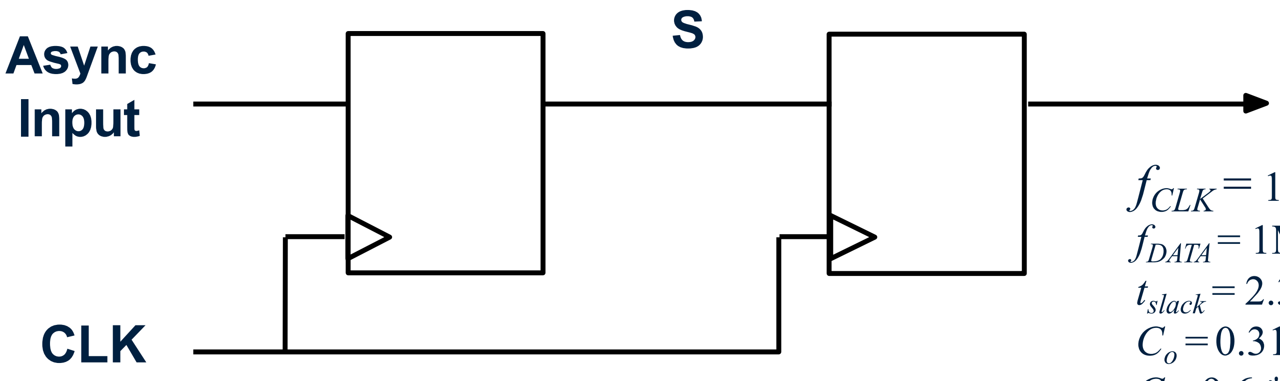
Higher MTBF means more robust design

- maybe hundreds or thousands of years

Required MTBF depends on application

- Life-critical medical equipment would need higher MTBF than consumer video game system

MTBF Example Calculation - Synchronizer



$f_{CLK} = 100\text{MHz}$

$f_{DATA} = 1\text{MHz}$

$t_{slack} = 2.3\text{ns}$

$C_o = 0.31\text{ns}$

$C_1 = 9.6 * 10^{-18}\text{s}$

$$MTBF = \frac{e^{t_{slack}/C_0}}{C_1 \cdot f_{CLK} \cdot f_{DATA}} = 20.1\, days$$

Quartus calculates all this for you!

THANK YOU