Slide Set 9

for ENEL 353 Fall 2019

Steve Norman, PhD, PEng

Electrical & Computer Engineering
Schulich School of Engineering
University of Calgary

Fall Term, 2019

Contents

Outline of Slide Set 9

Introduction to timing of sequential logic

For a synchronous sequential circuit design, some of the major timing concerns are ...

- What are sufficient conditions on the D input of a DFF to ensure reliable operations of the DFF? (This is called the "dynamic discipline".)
- Given timing specifications for DFFs and a desired clock period T_C, what do those things say about maximum delays in combinational elements in the circuit?
- What can go wrong if D inputs of DFFs go $0 \rightarrow 1$ or $1 \rightarrow 0$ at a bad time?

Section 3.5 of Harris & Harris is **excellent** on these topics. Please read it carefully, more than once!

Review: The *static discipline*

This idea was introduced very early in the course. (See Section 1.6 of Harris and Harris.)

The *static discipline* says that for reliable operation of digital circuit elements, voltages on inputs of circuit elements **must not sit** in the **forbidden zone** between V_{IL} and V_{IH}.

(Of course, voltages are allowed to pass through the forbidden zone when making low-to-high or high-to-low transitions!)

The *dynamic discipline*

The *dynamic discipline* has to do with rules about the **timing** of transitions on input signals to sequential devices such as latches and flip-flops.

If a sequential circuit design does not comply with the dynamic discipline, the circuit is likely to be **unreliable** or **completely defective**.

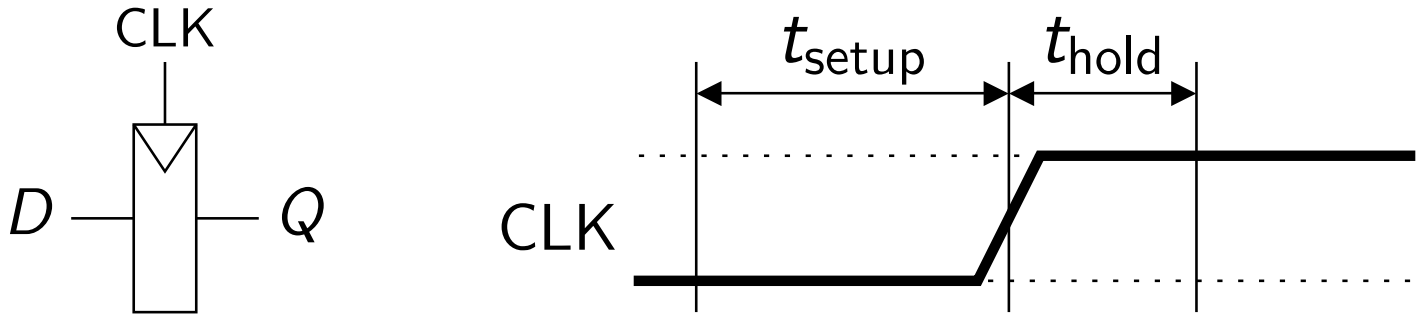
Specifically, for D flip-flops, the dynamic discipline says:

> The D input to a DFF must not make a $0 \rightarrow 1$ or $1 \rightarrow 0$ transition within an *aperture time* surrounding an an active clock edge.

The *aperture time*, as we'll see, is defined by two DFF timing parameters called the *setup time* and the *hold time*.

Outline of Slide Set 9

Setup and hold times for DFFs



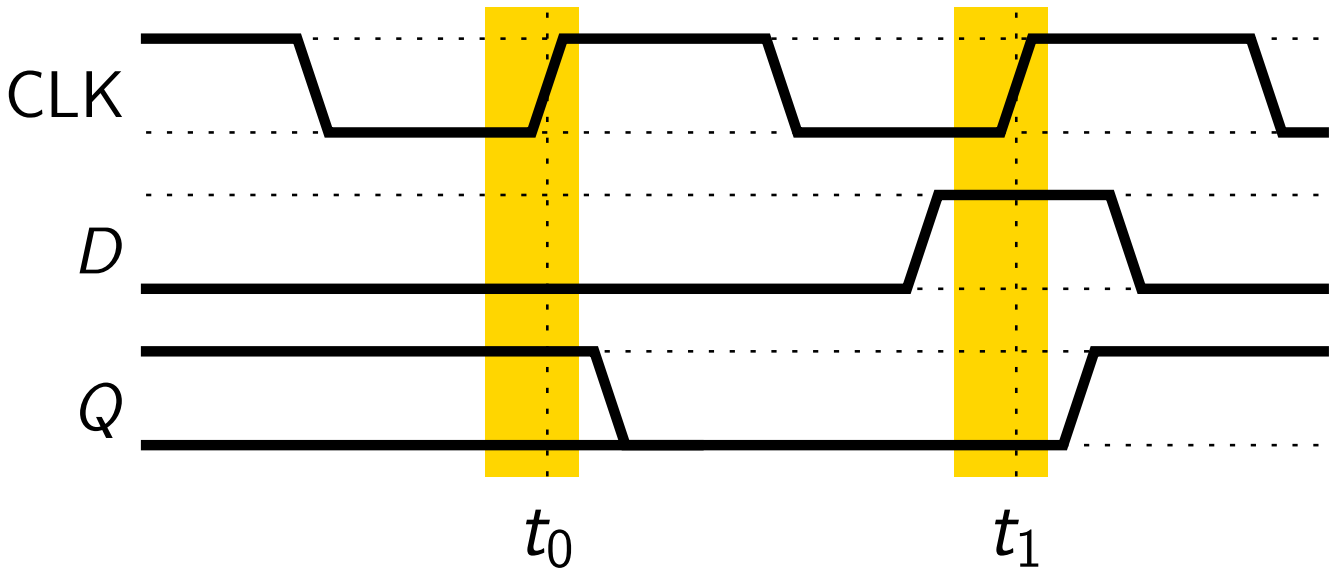
The *setup time*, t_{setup} (just t_s in some textbooks) is a short time interval **before** a rising edge on CLK.

The *hold time*, t_{hold} (just t_h in some textbooks) is a short time interval **after** a rising edge on CLK.

Proper DFF behaviour—Q copies D on rising edges of CLK—is guaranteed only if D **does not change value** within the *aperture time* defined by the setup and hold times.

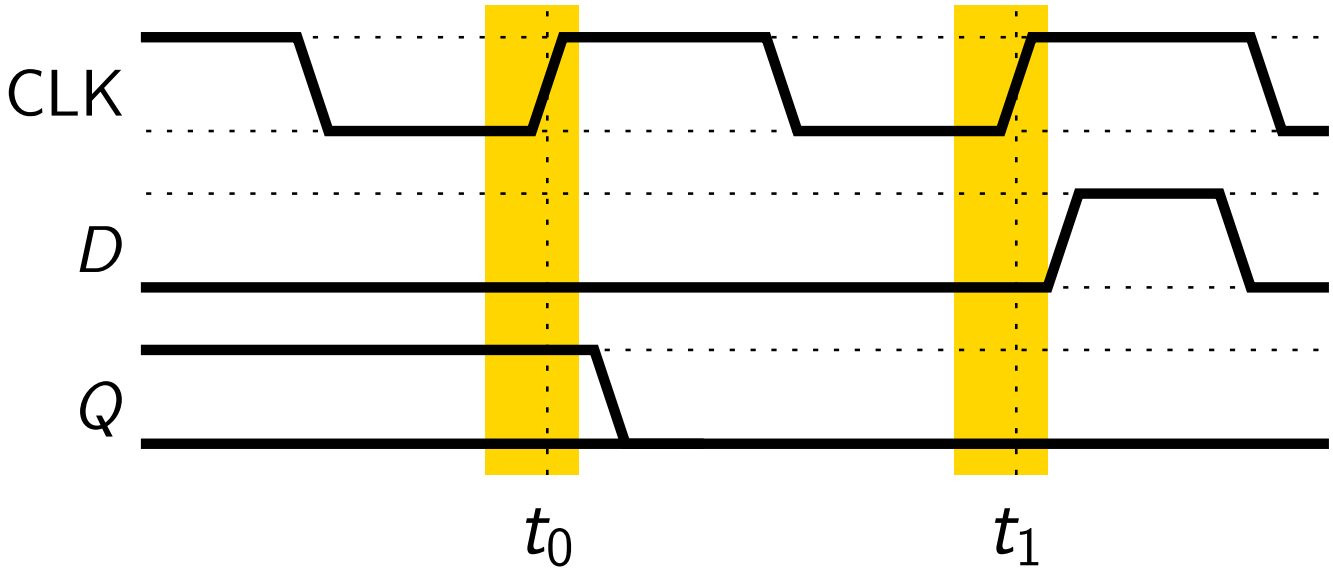
Setup and hold times—example 1

The gold rectangles mark **apertures** defined by setup and hold times for a DFF.



Here D is stable through both apertures, so Q **reliably** takes values of 0 shortly after t_0 and 1 shortly after t_1.

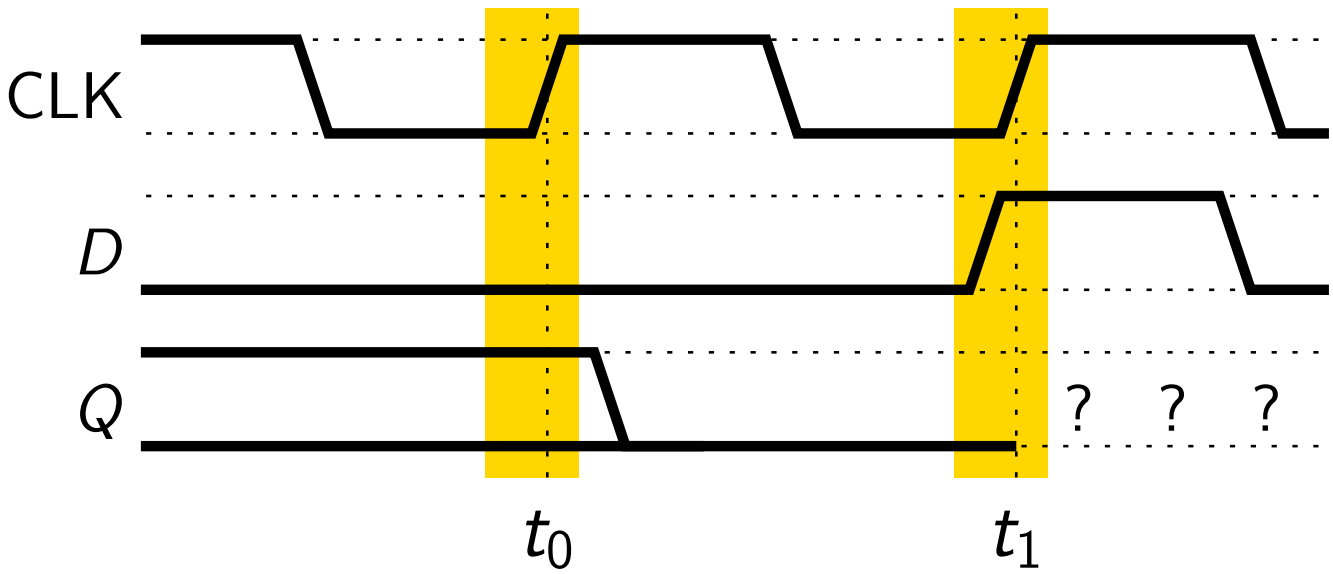
Setup and hold times—example 2



Here D changes after the aperture around t_1 is over, so Q remains 0 for the clock cycle following t_1.

Setup and hold times—example 3

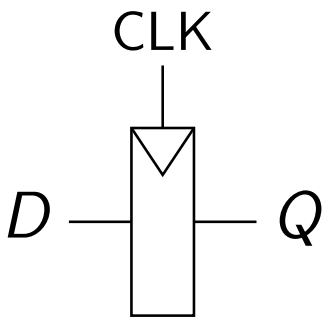
Here there is a **violation** of the setup-and-hold-time rules around t_1.



What happens to Q after t_1? There are **multiple possibilities**, which we'll get to later.

For now, what's important to know is that circuit behaviour following t_1 is **unpredictable**.

D flip-flop *clock-to-Q delays:* t_{ccq} and t_{pcq}

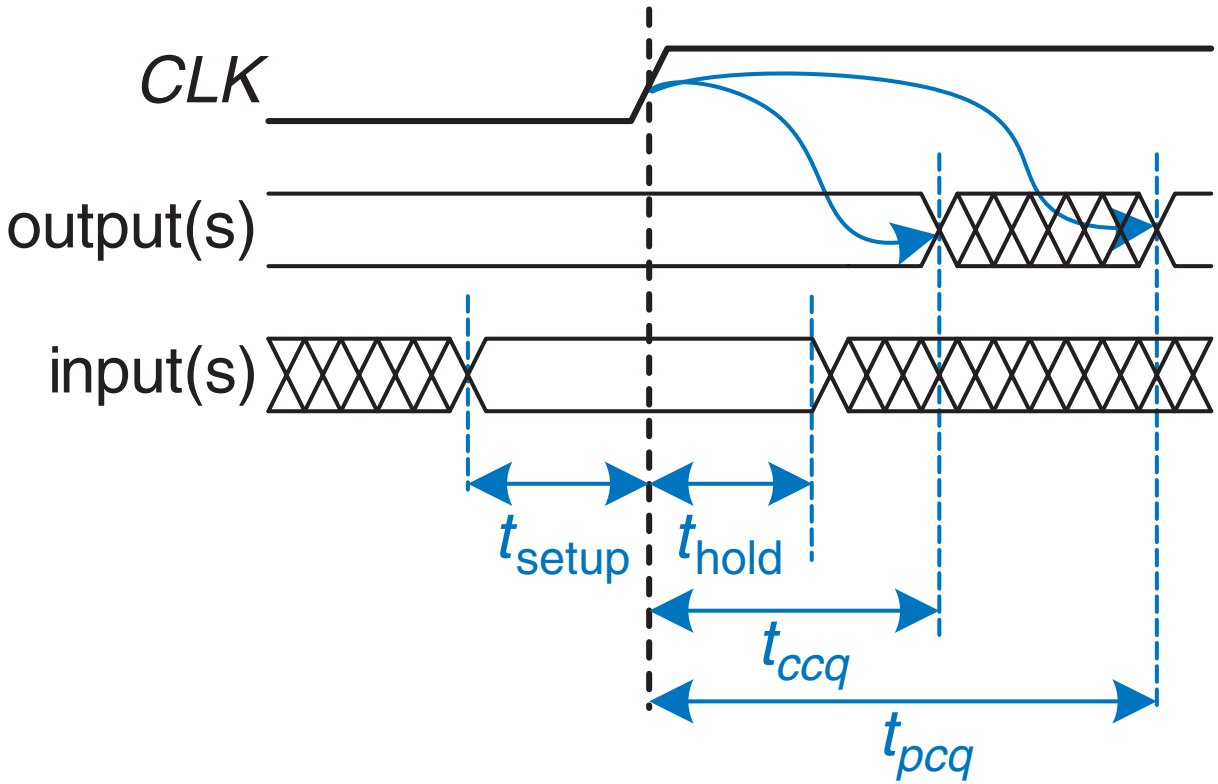


As mentioned several lectures ago, the Q output of a DFF does not change at exactly the same time as its input CLK signal rises—there is a short delay.

t_{ccq} is the *clock-to-Q contamination delay*. It takes **at least** this much time for a rising edge of CLK to cause a change in Q.

t_{pcq} is the *clock-to-Q propagation delay*. It takes **no more than** this much time for a rising edge of CLK to cause a change in Q.

t_{setup}, t_{hold}, t_{ccq}, and t_{pcq}, all on one timing diagram



Let's write down some notes about how to read this diagram.

Image is Figure 3.37 from Harris D. M. and Harris S. L., *Digital Design and Computer Architecture, 2nd ed.*, © 2013, Elsevier, Inc.

Where do DFF timing parameters come from?

The range of possible clock-to-Q delays from as fast as t_{ccq} to as slow as t_{pcq} reflects factors such as

- variation in V_{DD}
- variation in temperature
- minor variations in physical dimensions and chemical composition of transistors
- various other physical factors.

t_{setup} and t_{hold} are **worst-case** numbers over all allowable operating conditions for a circuit.

Remark about resettable, settable, and enabled flip-flops

The textbook doesn't mention this, but it's good to know.

For DFFs with EN inputs, and/or **synchronous** reset or set inputs, the EN, reset and set inputs have t_{setup} and t_{hold} parameters that are similar to the t_{setup} and t_{hold} parameters for the D input.

For DFFs with **asynchronous** reset or set inputs, the timing parameters for those inputs are typically a **minimum width** for a reset or set **pulse**, along with a **minimum gap** between when reset or set is turned off and a rising edge of the clock.

Outline of Slide Set 9

Timing constraints for synchronous sequential logic

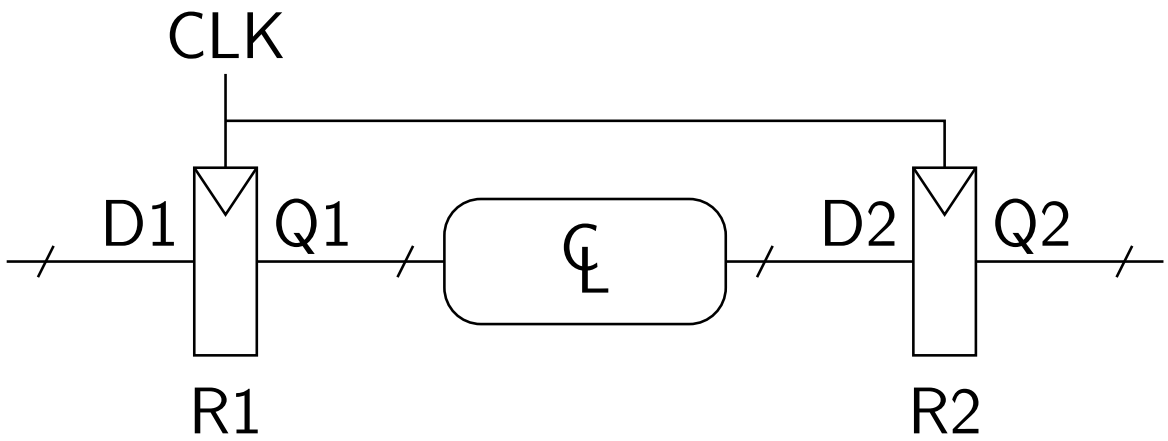
We have now been introduced to the DFF timing parameters t_{setup}, t_{hold} t_{ccq}, and t_{pcq}.

We're about to put those ideas **together** with earlier ideas about timing of **combinational** logic.

The results will be some **inequalities** that will help us assess whether **synchronous sequential** designs are safe with respect to timing.

A generic piece of synchronous sequential logic

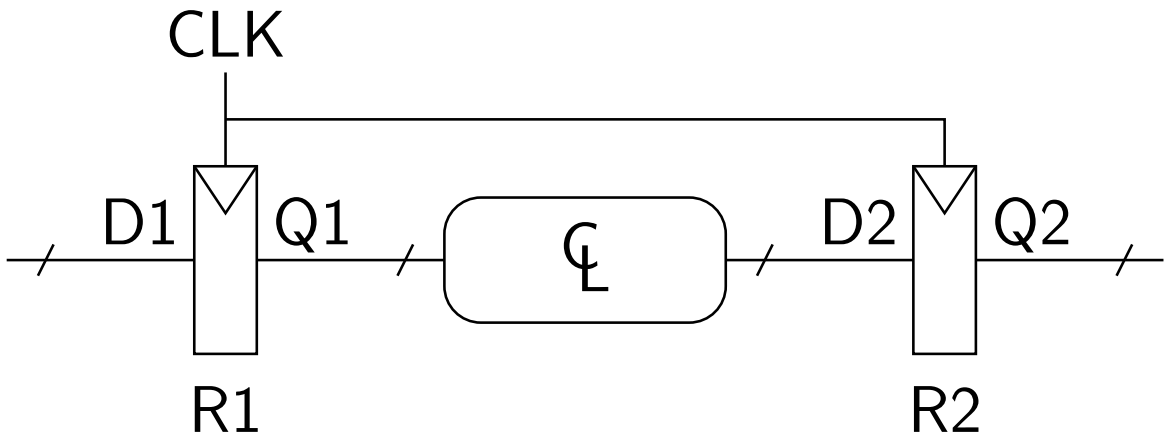
Below is a small part of a larger synchronous sequential circuit. Registers R1 and R2 are collections of DFFs that all have the same t_{setup}, t_{hold}, t_{ccq}, and t_{pcq}.



The combinational element shown has contamination delay t_{cd} and propagation delay t_{pd}.

We'll assume that signal D1 meets the setup and hold time requirements of R1, and look at whether signal D2 meets the setup and hold time requirements of R2.

Setup time constraint

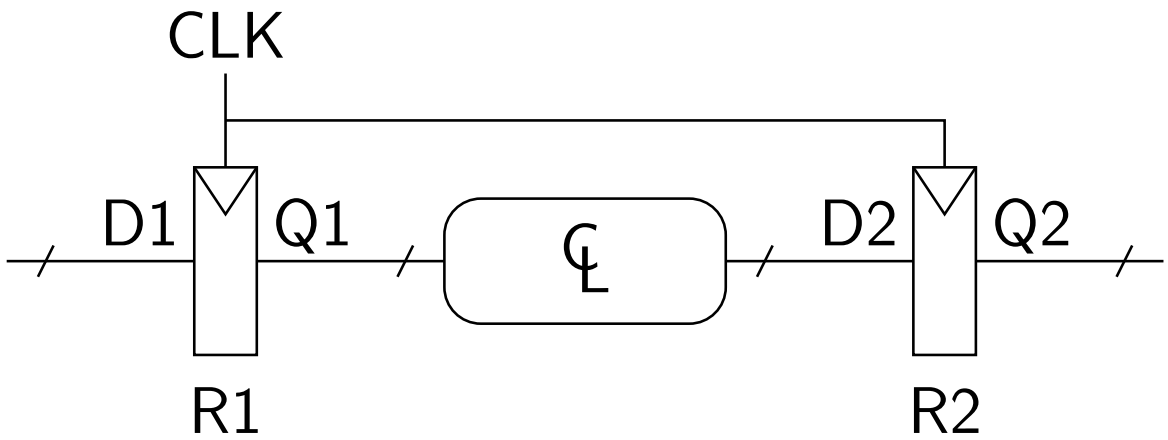


Recall that T_C stands for the **clock period**.

Suppose there is a rising edge of CLK at time t_0. What must be true so that there is **no setup time violation** at R2 at the **next** rising edge of CLK, at time $t_0 + T_C$?

Let's do the simple math, then make some remarks.

Hold time constraint

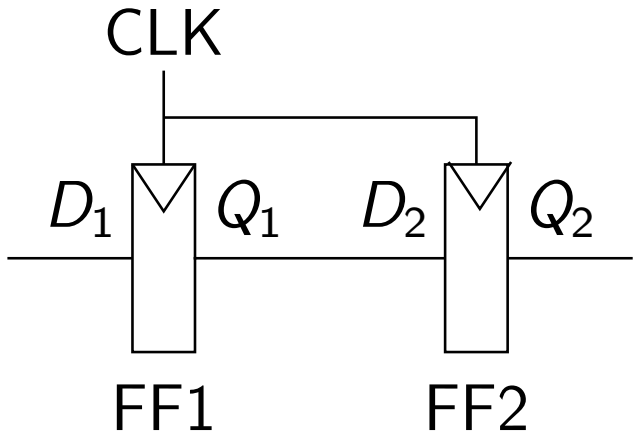


Suppose there is a rising edge of CLK at time t_0. What must be true so that there is **no hold time violation** at R2 at the **same** rising edge of CLK, also at time t_0?

Again, let's do some simple math, then make some remarks.

Hold time constraint: Direct Q-to-D connection

Let's look at this special case, in which there is no combinational delay between a Q output of a DFF and the D input of another DFF.

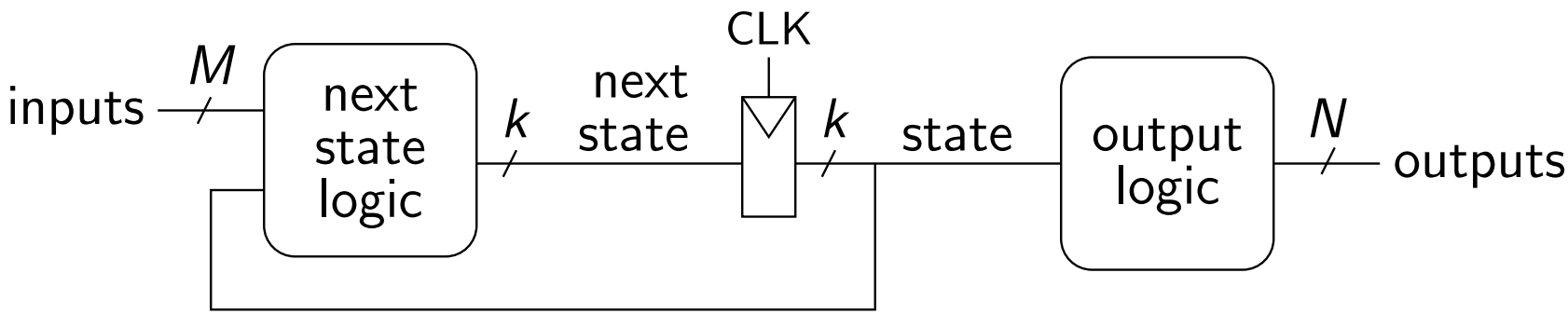


Let's assume that the DFFs are identical, and that setup and hold time conditions are satisfied by the D_1 input to FF1.

What must be true so that there is **no hold time violation** at FF2 at the **same** rising edge of CLK, also at time t_0?

Let's do the very simple math, then make some remarks.

Setup time constraint for a Moore-type FSM circuit

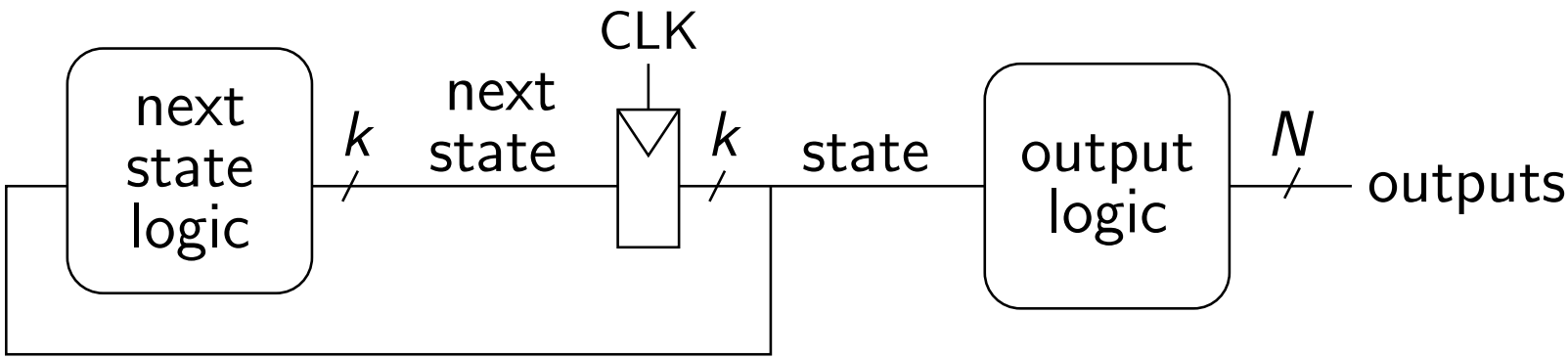


Suppose that T_C is the desired clock period.

Suppose we know t_{setup} and t_{pcq} for the register, and we know t_{pd} for the next-state logic.

Can we do a simple calculation to determine whether this circuit might have setup time violations? Why or why not?

Setup time constraint for a "free running" Moore-type FSM circuit



Again, suppose that T_C is the desired clock period.

Again, suppose we know t_{setup} and t_{pcq} for the register, and we know t_{pd} for the next-state logic.

For the above circuit, it **is** possible to do a simple calculation to check for possible setup time violations.

Let's do the calculation, then make some remarks.

Synchronous logic timing: Detailed example

Suppose A is connected to V_{DD} in the circuit on the next slide.

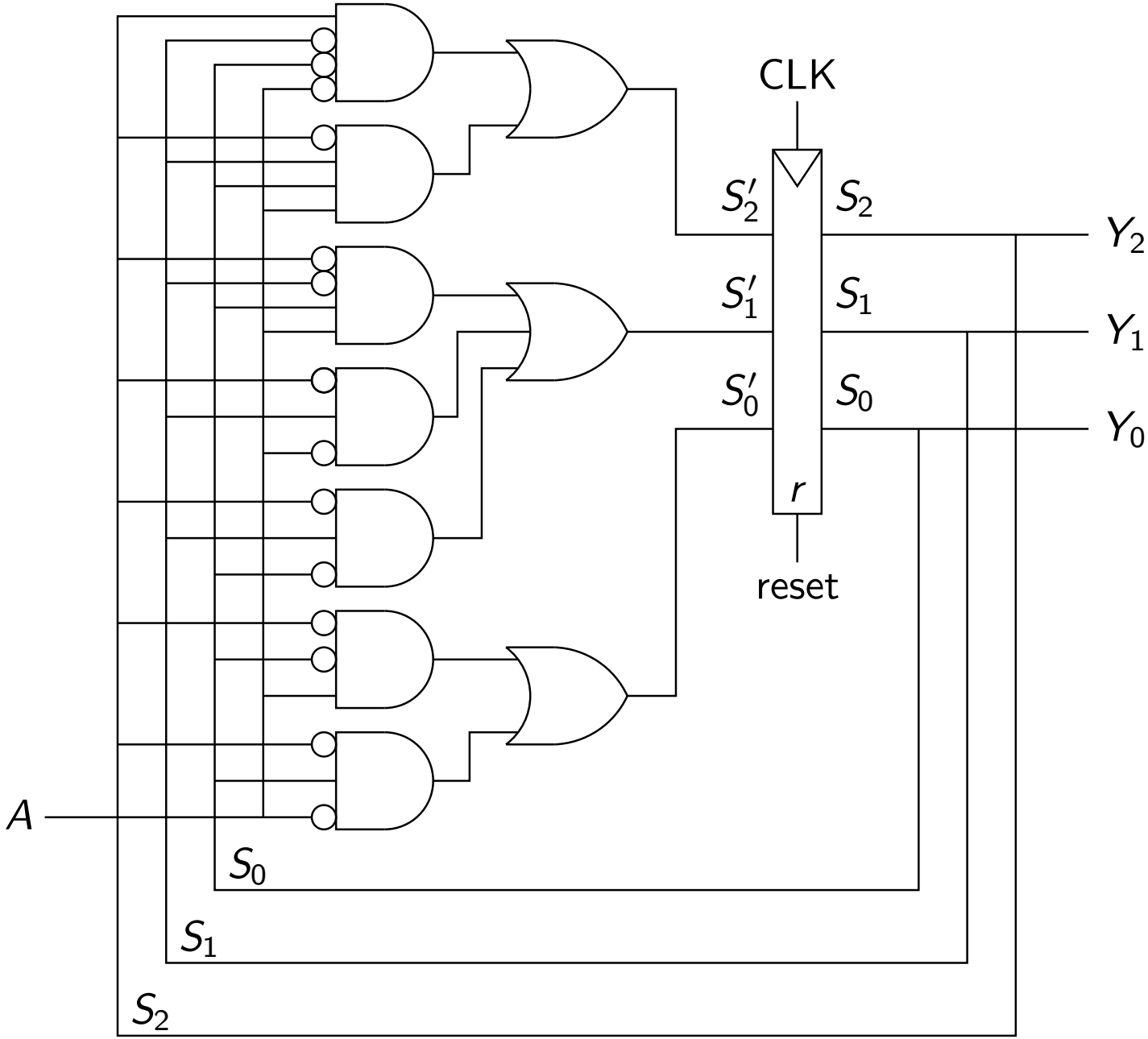
Bubbles on the AND gate inputs are implemented using NOT gates.

For the register, $t_{setup} = 35\,\text{ps}$ and $t_{pcq} = 75\,\text{ps}$.

Is it safe to run the clock with a frequency of 3.33 GHz?

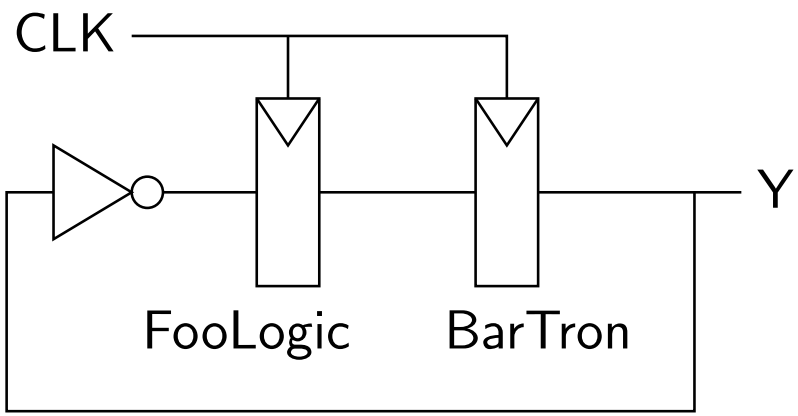
If not, what is a simple redesign that would allow safe operation at 3.33 GHz?

gate	t_{pd} (ps)
NOT	30
NAND2	40
NAND3	60
NAND4	80
AND2	60
AND3	80
AND4	100
OR2	80
OR3	110


CLK
S_2'
S_2
S_1'
S_1
S_0'
S_0
r
reset
Y_2
Y_1
Y_0
A
S_0
S_1
S_2

Another example of timing analysis

Using some 1970's/1980's inverters and DFFs found in a junk drawer, a student builds a clock-divide-by-4 circuit. For the inverters, $t_{cd} = 9\,\text{ns}$ and $t_{pd} = 15\,\text{ns}$. DFF timing parameters, in ns, are given in the table.



	family	
parameter	Foo	Bar
t_{setup}	2	20
t_{hold}	1	7
t_{pcq}	8	50
t_{ccq}	5	30

The student tests the circuit with a 1 MHz CLK input, expecting to see a 250 kHz square wave on Y.

Why doesn't the circuit work? What can be done to fix it?

Outline of Slide Set 9

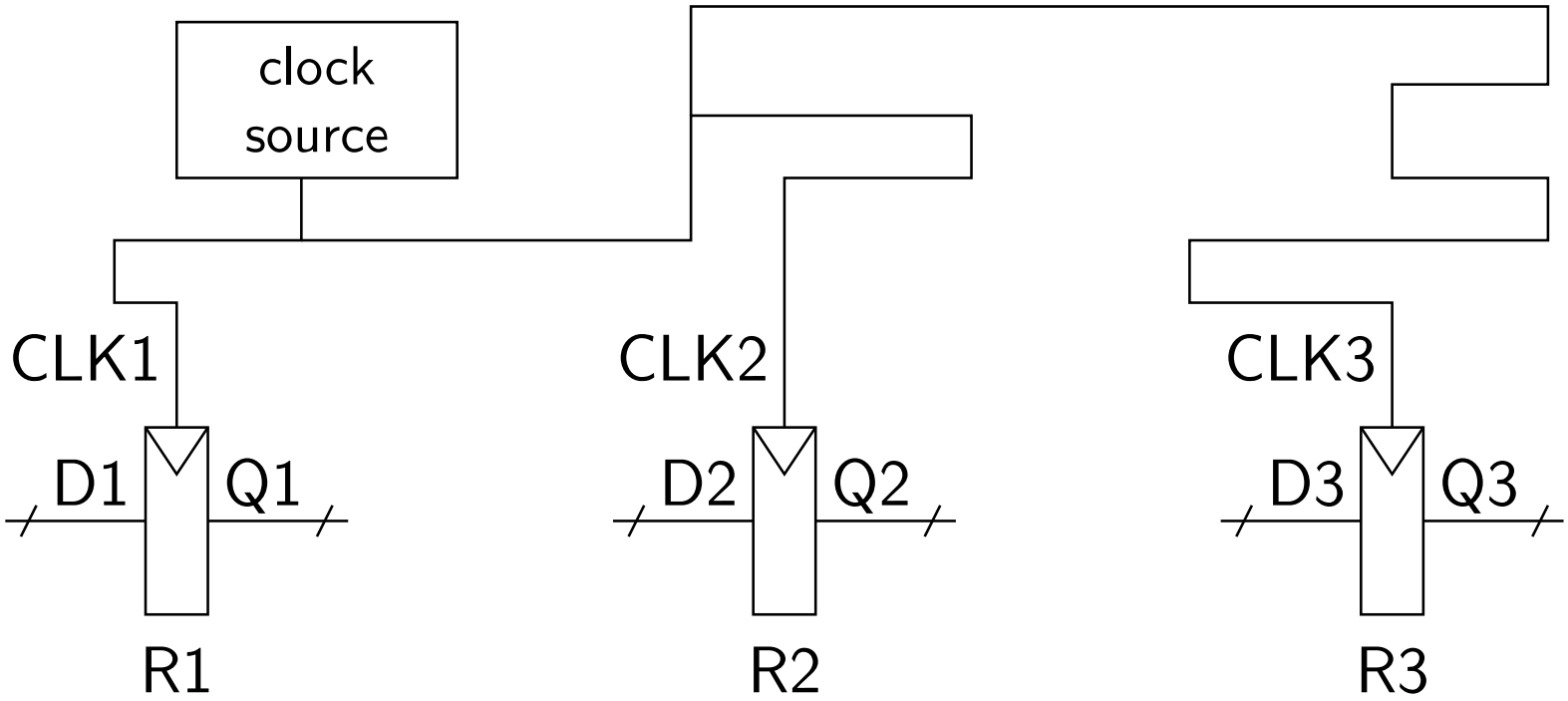
Introduction to *clock skew*

This list is review. It's a list of sufficient conditions for building a **synchronous sequential circuit** ...

1. Every element in the circuit either is a register or is combinational.
2. At least one element is a register.
3. All registers receive the same clock signal.
4. Every cyclic path in the circuit passes through at least one register.

Unfortunately, the laws of physics make it very hard to perfectly satisfy condition 3 ...

It takes **time** for a voltage change to propagate along a wire.



Clock edges received by R1 are **early** relative to clock edges received by R2. Clock edges received by R3 are **late** relative to clock edges received by R2.

Minimization of clock skew; definition of t_{skew}

Clock skew is the name given to the problem having different registers get clock edges at slightly different moments in time.

Delay from the clock source to clock inputs **cannot** be avoided. Circuit designers try to minimize clock skew by making all the source-to-input delays **very close to the same**.

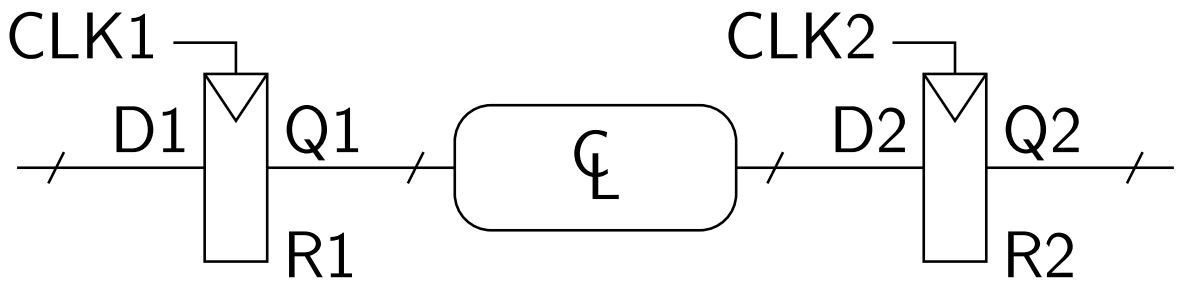
(Because delays can be affected by factors such as temperature and electrical noise, clock skew can't be made zero just by ensuring that all clock wires have the same length.)

In a synchronous sequential circuit, t_{skew} is defined as the worst-case difference in times of arrival of an active clock edge at any two registers in the circuit.

Outline of Slide Set 9

Clock skew, setup and hold time constraints

CLK1 and CLK2 come from the same clock source, but due to clock skew, clock edges might not arrive at R1 and R2 at exactly the same time.

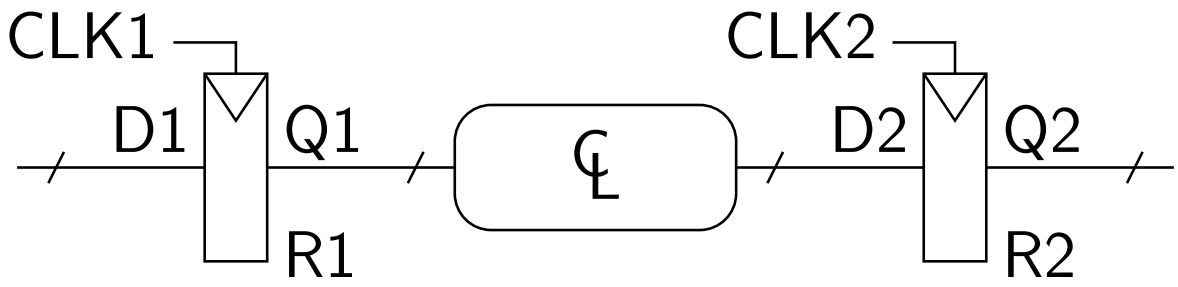


Review: If it happens that that there is **no** clock skew, then we know for safe operation this must be true . . .

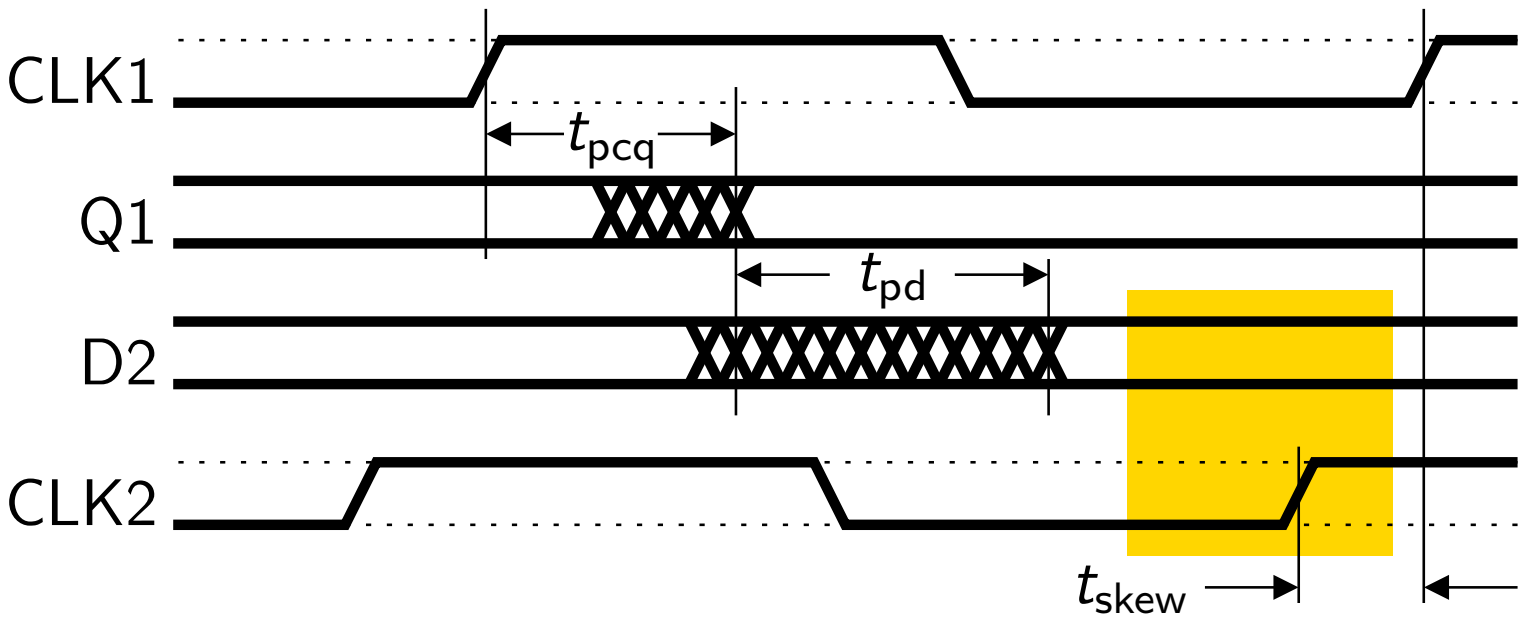
$$t_{pd} \text{ for } \text{℄} \leq T_C - (t_{pcq} + t_{setup}) \qquad (t_{setup} \text{ constraint})$$

$$t_{cd} \text{ for } \text{℄} \geq t_{hold} - t_{ccq} \qquad (t_{hold} \text{ constraint})$$

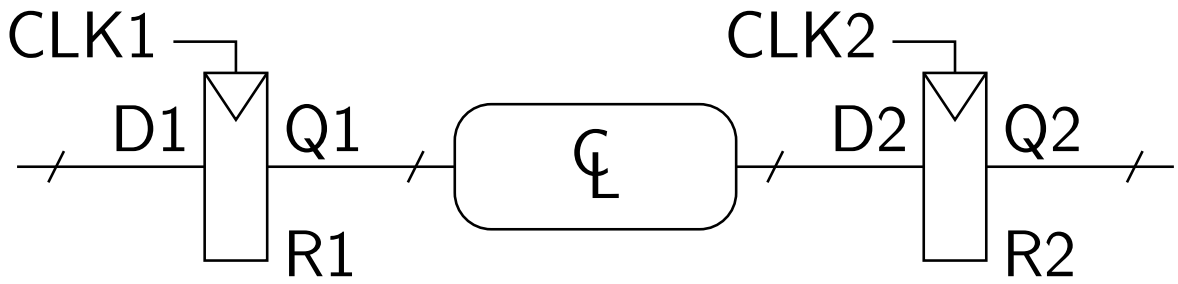
Clock skew and the setup time constraint



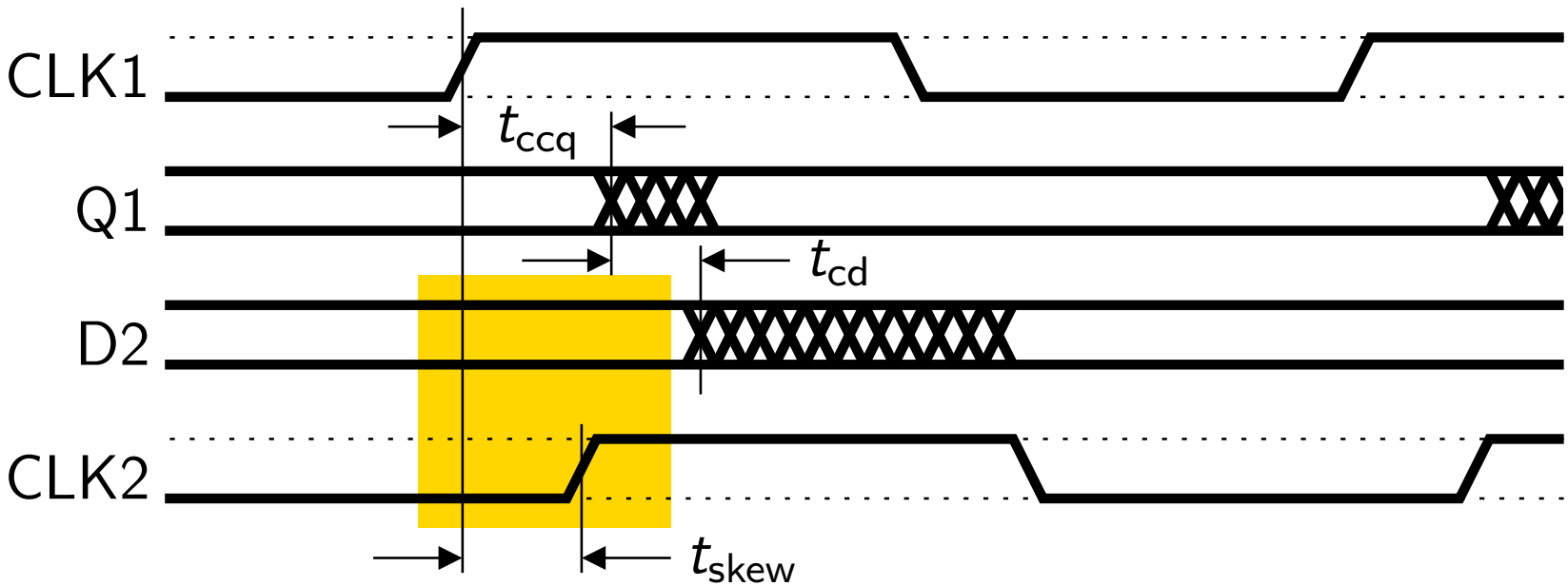
Suppose CLK2 is **early** relative to CLK1. *For reliable operation, what must be true about the speed of the combinational logic?* (The gold rectangle shows the $t_{\text{setup}}/t_{\text{hold}}$ aperture for R2.)



Clock skew and the hold time constraint



Now suppose CLK2 is **late** relative to CLK1. *For reliable operation, what must be true about the speed of the combinational logic?* (Again, the gold rectangle shows the $t_{\text{setup}}/t_{\text{hold}}$ aperture for R2.)



Summary of timing constraints in the presence of clock skew

setup time constraint: $t_{pd} \leq T_C - (t_{pcq} + t_{setup} + t_{skew})$

hold time constraint: $t_{cd} \geq t_{hold} + t_{skew} - t_{ccq}$

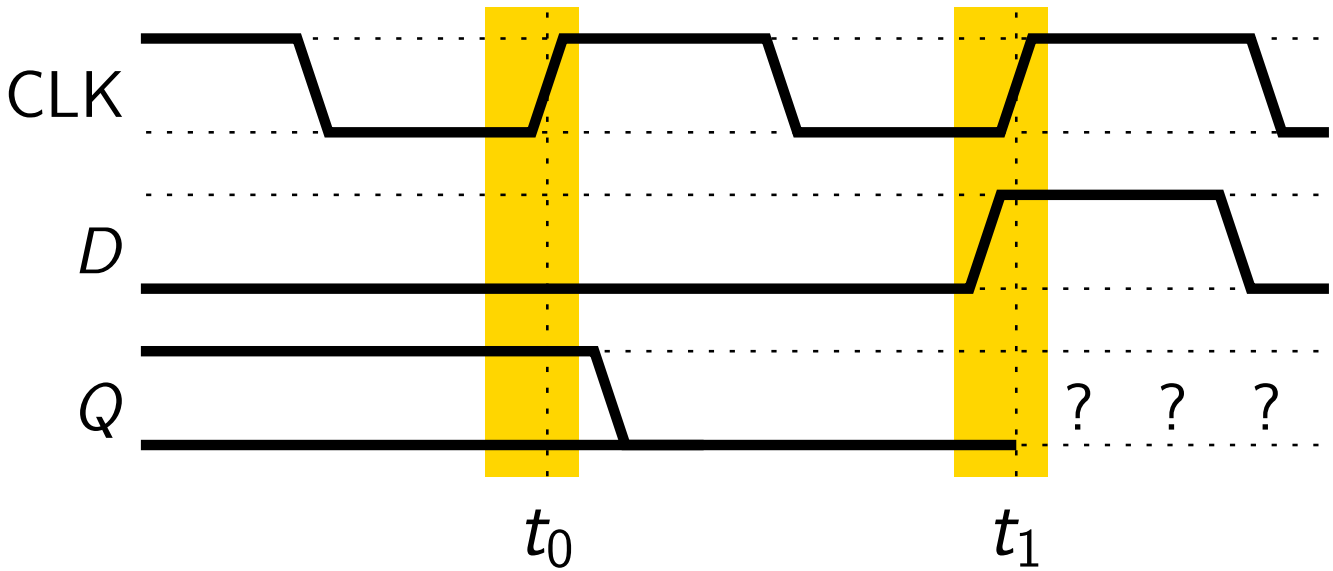
Things to note:

- If we set $t_{skew} = 0$, we get the same inequalities we've derived previously.
- Both inequalities say that as t_{skew} increases, the circuit designer's job gets more difficult. t_{pd} may need to be reduced on some paths, and t_{cd} may need to be increased on other paths.

Outline of Slide Set 9

Review of a simple timing example

Here there is a **violation** of the setup-and-hold-time rules around t_1.

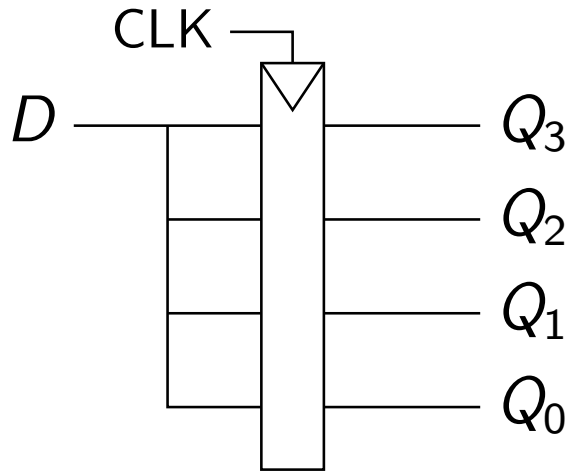


What happens to Q after t_1? There are **multiple possibilities**.

Let's look at these multiple possibilities in more detail.

An odd-looking example circuit

Connecting one signal to four register inputs will let us make a point about the variety of possible responses to an aperture time violation.



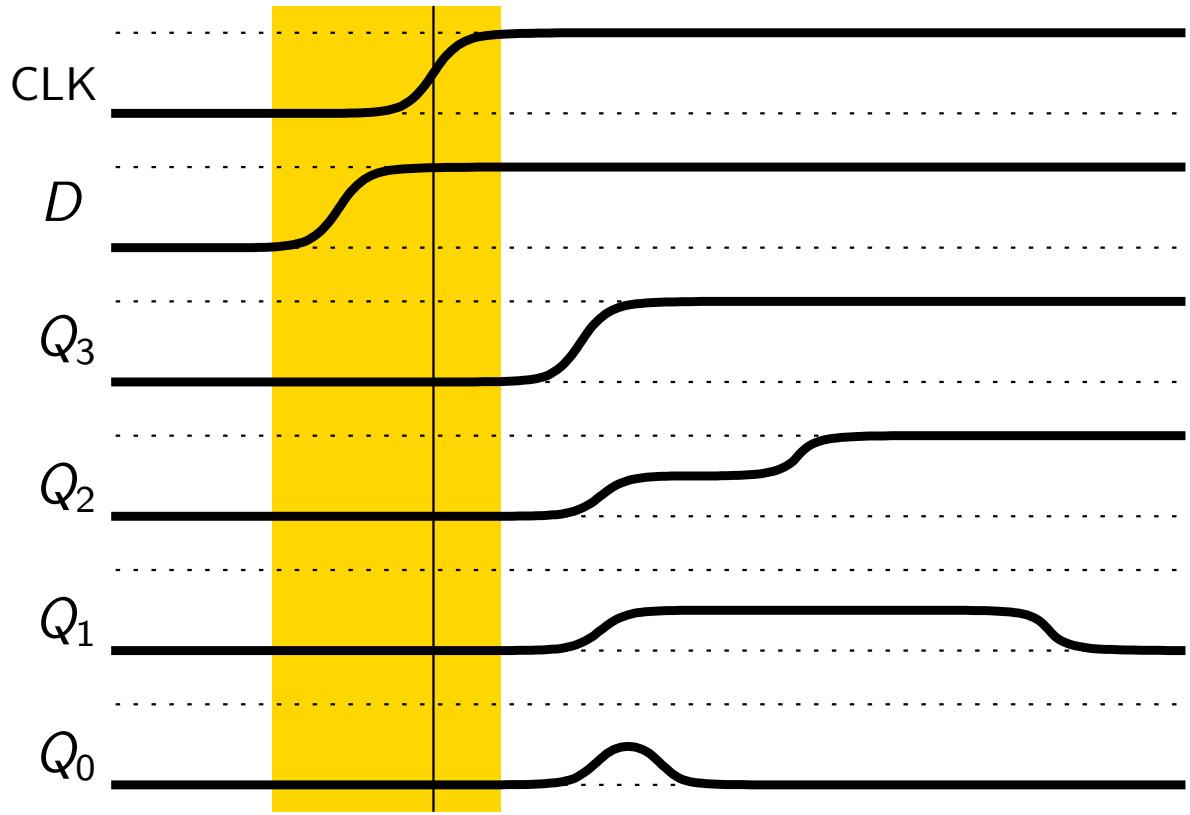
The DFFs in the register are very close to identical, but not perfectly so, due to minor manufacturing variations.

If the setup and hold time rules are respected, all four Q values will copy D on each rising edge of CLK, with a delay in the range from t_{ccq} to t_{pcq}.

What might happen if the setup and hold time rules are violated?

Example responses to a setup time violation

Here we assume that $Q_{3:0} = 0000$ before the rising edge of CLK.



Let's make some remarks about these responses.

Metastability in latches and flip-flops

In normal operation the Q and $\overline{Q}$ signals of a latch or flip-flop will sit in one or the other of two **stable** states: $(Q, \overline{Q}) = (0, 1)$ or $(Q, \overline{Q}) = (1, 0)$.

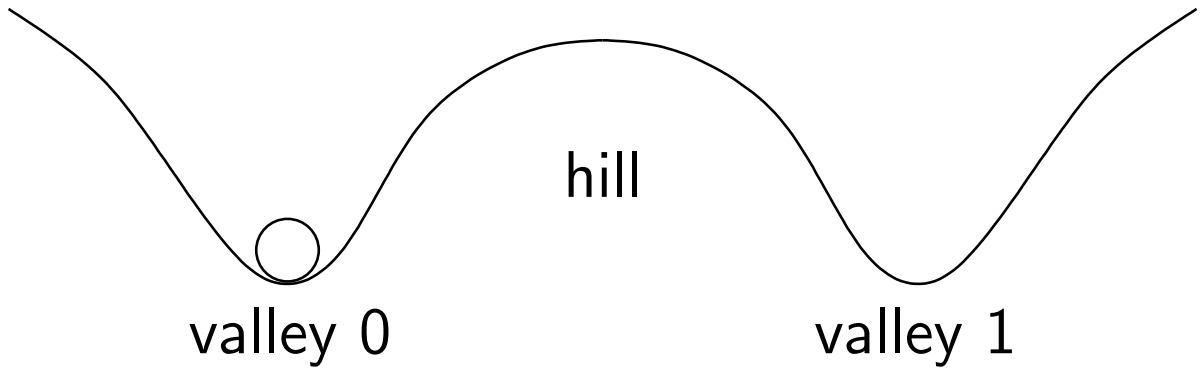
Metastability is the name given to a kind of abnormal behaviour in which the voltages of the Q and $\overline{Q}$ signals both sit **approximately halfway** between 0 and V_{DD} for some period of time called t_{res}, the *resolution time*.

Once the resolution time has passed, the latch or flip-flop goes to ("resolves" to) one or the other of its stable states.

t_{res} is **random**, differing in length from one instance of metastability to the next.

A mechanical analogy for metastability

There are two stable places for the ball: the bottom of valley 0, and the bottom of valley 1.



Given a gentle nudge, the ball will move a little, but stay in valley 0. Given a strong push, the ball will roll over the hill and settle in valley 1.

What if the ball is given a push that gives it just enough energy to get to the top of the hill and stop?

Why is metastability dangerous?

The key problem is this: t_{res} will sometimes be much longer than t_{pcq}. Sometimes t_{res} may be as long as one whole clock period in a synchronous system.

Of course, the output signal of a flip-flop or latch is typically an input signal to one or more other circuit elements.

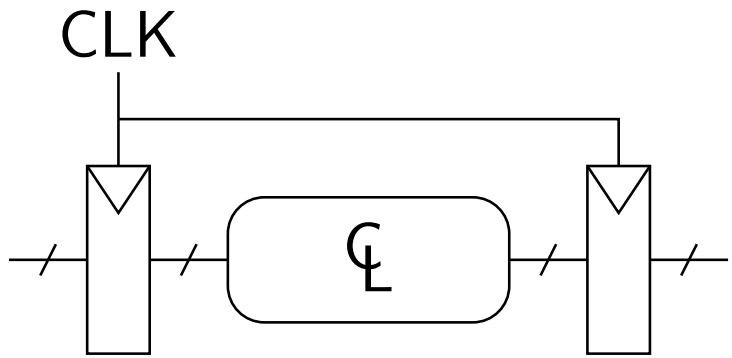
A circuit element with a metastable input will probably generate an incorrect output.

So metastability in a **single** DFF could cause an **entire** synchronous circuit to behave incorrectly, possibly getting the circuit into a state from which it can't recover.

Preventing metastability from causing circuit failure

A single DFF going metastable for a fraction of a clock cycle may cause a synchronous system to behave incorrectly for a much longer time period, and may even cause the system to freeze completely.

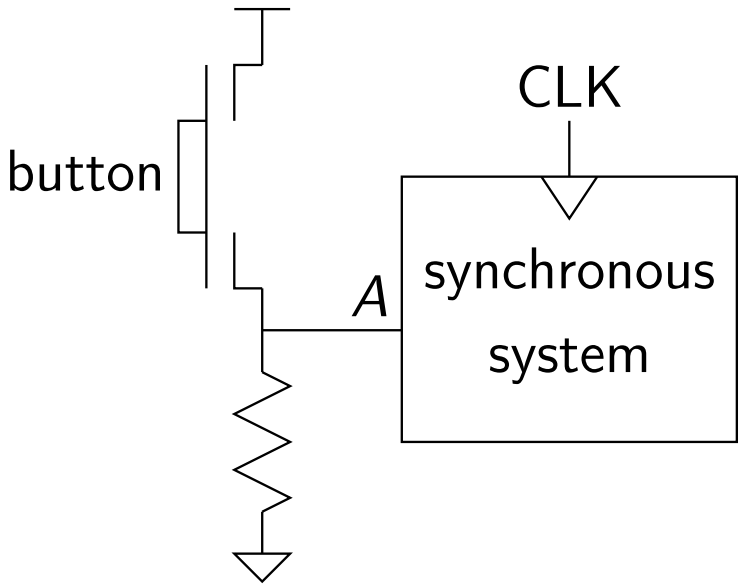
One essential step in reducing the risk of problems is, as we've just seen, careful timing analysis of paths like this ...

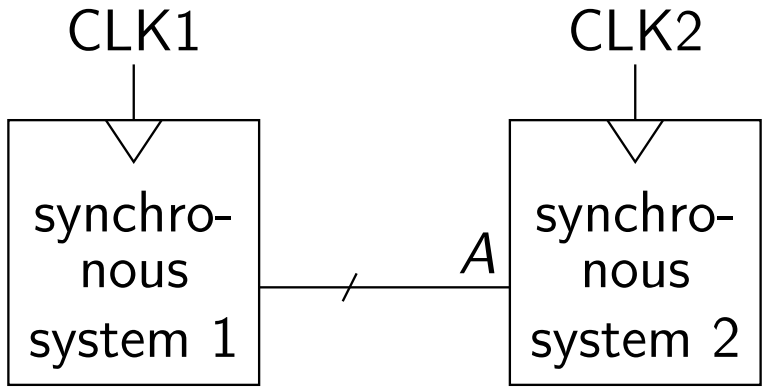


But what about **system inputs?** Edges on inputs can have completely unpredictable timing relative to edges on the system clock!

Examples of asynchronous inputs

In each of the systems below, A is what is called an *asynchronous input* . . .



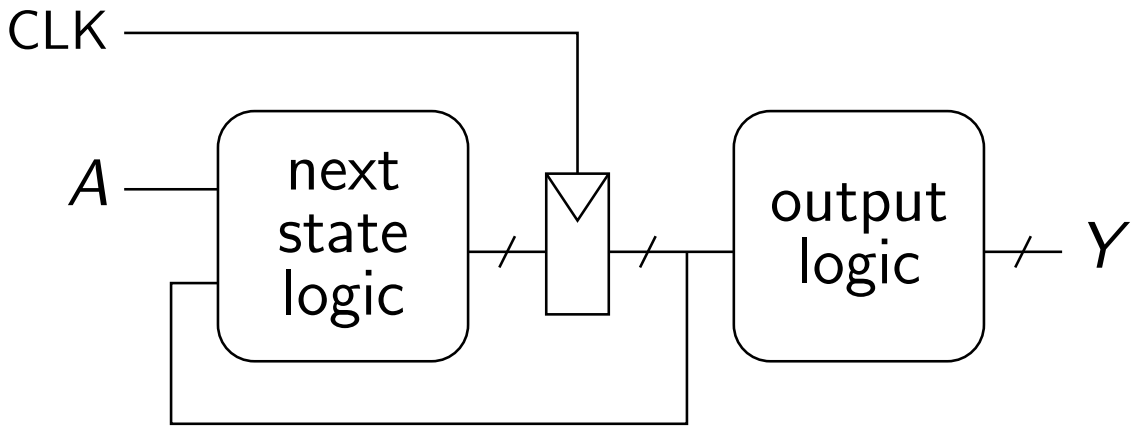


CLK1 and CLK2 are **unrelated**, with different frequencies.

On the left, the system can't control when a human might press or release the button. On the right, there will be no predictable relationship between edges on A and edges on CLK2.

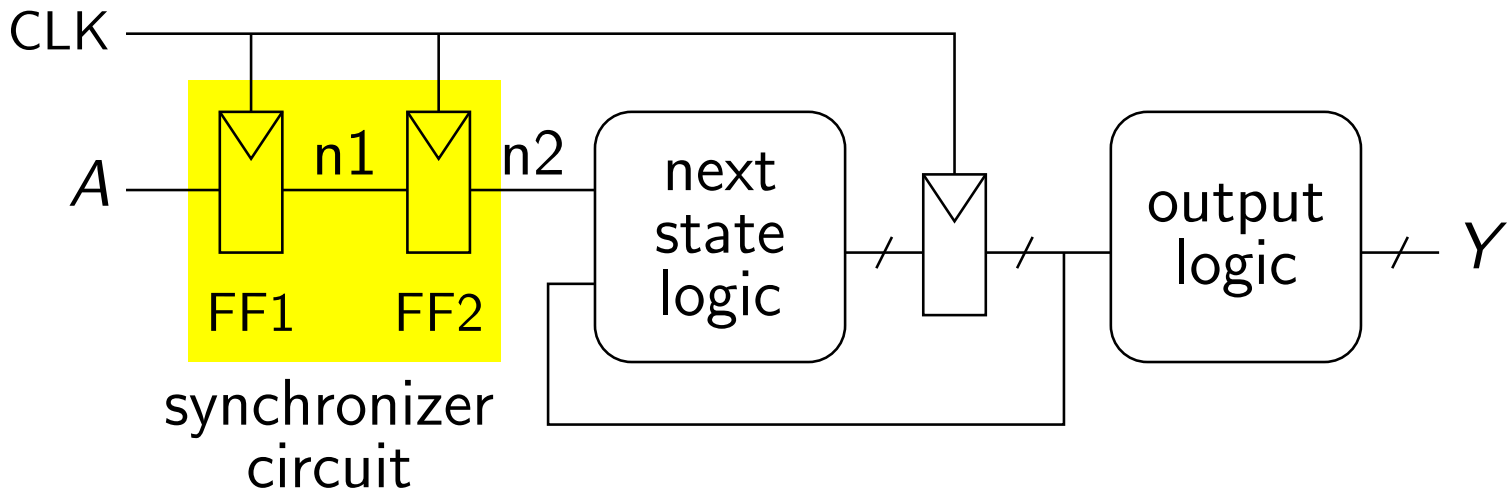
An example FSM with an asynchronous input

If we don't know anything at all about when edges on A might appear relative to edges on CLK, this is a **bad design . . .**



Why is it impossible to design the next state logic to prevent disastrous metastability in the state register?

To greatly reduce the risk, A can be passed though a *synchronizer circuit*, as shown on the next slide . . .



Suppose the next state logic has been designed so that its t_{pd} and t_{cd} meet setup and hold constraints for the state register.

Recall that t_{res} is the resolution time for a DFF or register.

Suppose that a detailed transistor-level model says that it is **extremely unlikely** that it will ever happen that $t_{\text{res}} > 0.5\,T_{\text{C}}$.

Let's give an approximate, qualitative argument that there is very little risk of metastability in the state register.

Detailed analysis of t_{res} and synchronizers

In Sections 3.4.4–3.4.6, Harris and Harris present a formula for the probability distribution of t_{res}:

$$P(t_{res} > t) = \frac{T_0}{T_C} \exp\left(\frac{-t}{\tau}\right)$$

They go on to use that probability distribution to derive a formula for MTBF (mean time between failures) of synchronizer circuits.

In ENEL 353 in Fall 2019, we are **not** going to cover that material, and you will **not** be tested on it on the final exam.

You **are** expected to understand the qualitative ideas about metastability and synchronizers presented in Sections 3.4.4 and 3.4.5.

Outline of Slide Set 9

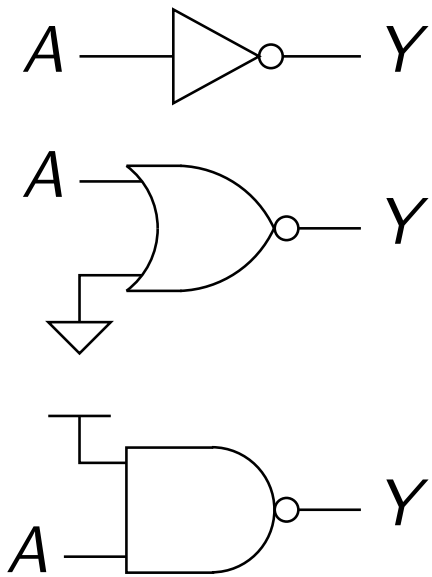
Some insight into circuits, setup times, and metastability

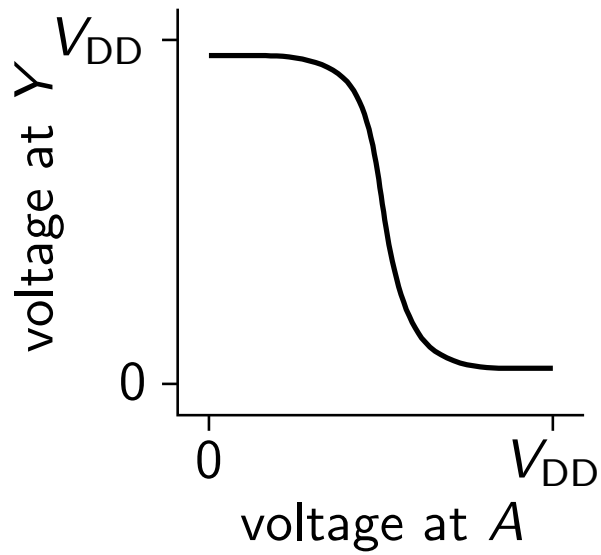
The rest of the slides in this slide set are **not** exam material in ENEL 353 in Fall 2019.

They are intended to provide some insight into why latches and flip-flops must have setup times, and why latches and flip-flops can go metastable.

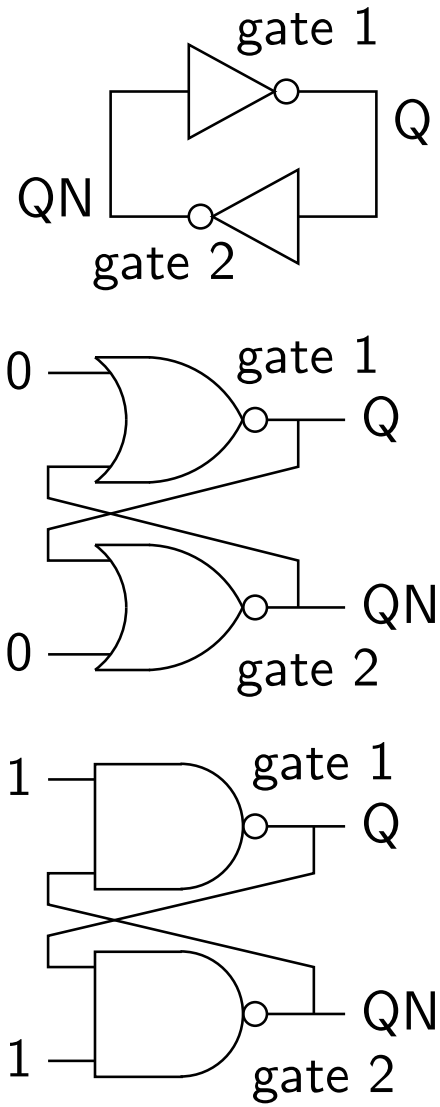
Static behaviour of CMOS NOT, NOR, and NAND gates

If we **slowly** vary the voltage at A in each of the circuits below, and measure the voltage at Y as we go, we'll see an input/output relationship that looks something like the graph sketched to the right.

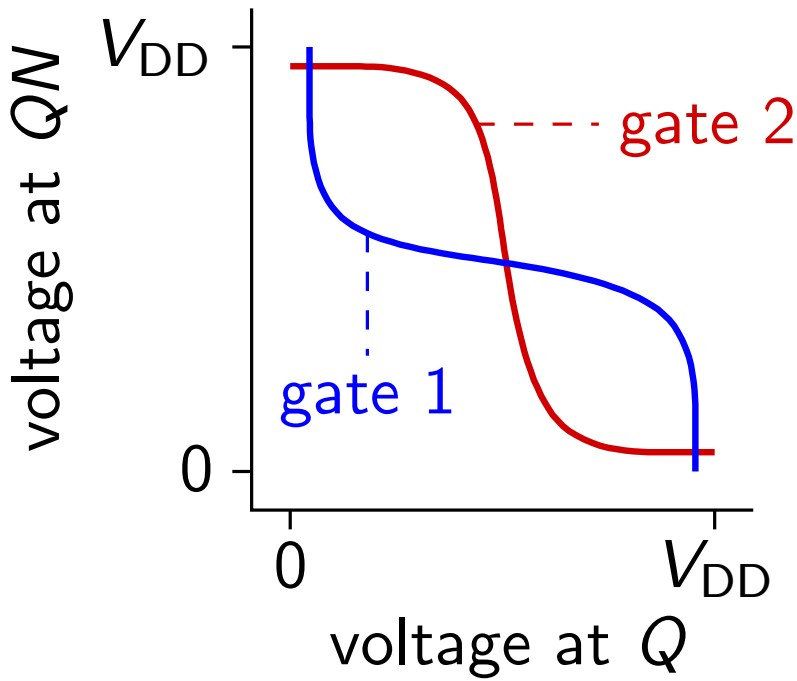




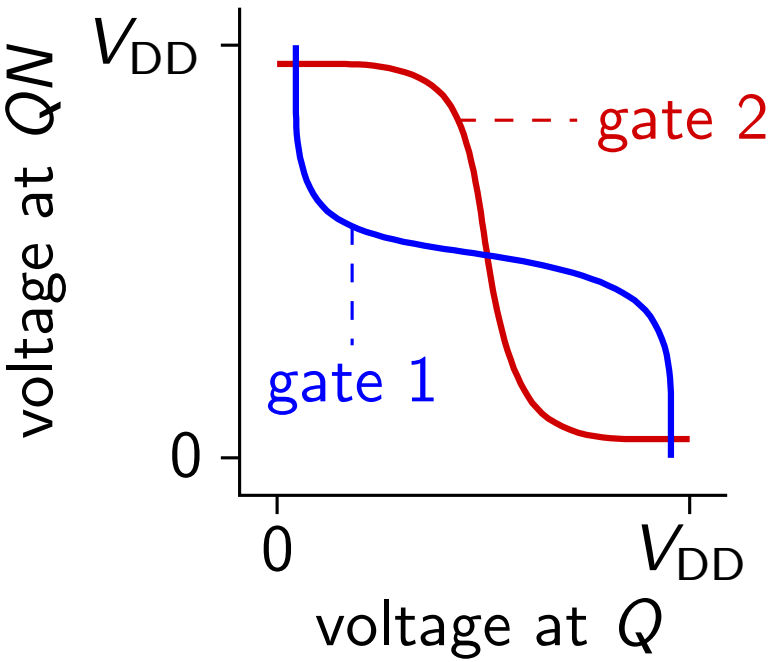
Static solutions for some bistable circuits



For each of the circuits, there are **three** static conditions that satisfy input/output voltage relationships for both gates ...



Static solutions for bistable circuits, continued



The upper-left and lower-right solutions are the **stable** solutions predicted by Boolean algebra: $(Q, QN) = (0, 1)$ and $(Q, QN) = (1, 0)$.

Boolean algebra only works with 1's and 0's, so **cannot** predict the **metastable** solution in the middle of the graph.

Dynamic behaviour of bistable circuits

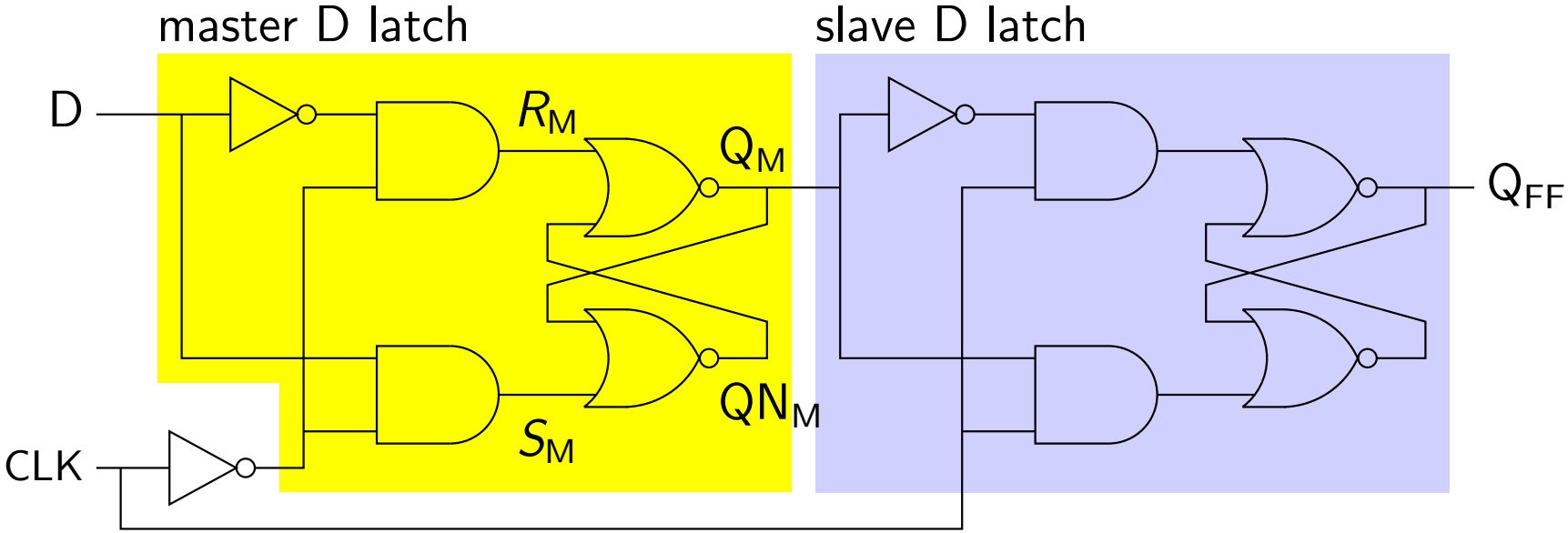
Static analysis shows us only what the possible solutions are when $dV/dt = 0$ for both gate outputs, and does not tell us how voltages might change as a function of time.

Transistor physics and circuit theory, beyond what has been taught to year 2 ENEL and ENSF students, says ...

- The static solutions that are identified by Boolean algebra are **stable equilibrium points**. If a bistable circuit is in one of its stable states, moderate amounts of electrical noise will **not** move the state very far away from that stable state.
- The metastable state of a bistable circuit is an **unstable equilibrium point**. A **very tiny** amount of electrical noise will rapidly drive the circuit from there to one or the other of its stable equilibrium points.

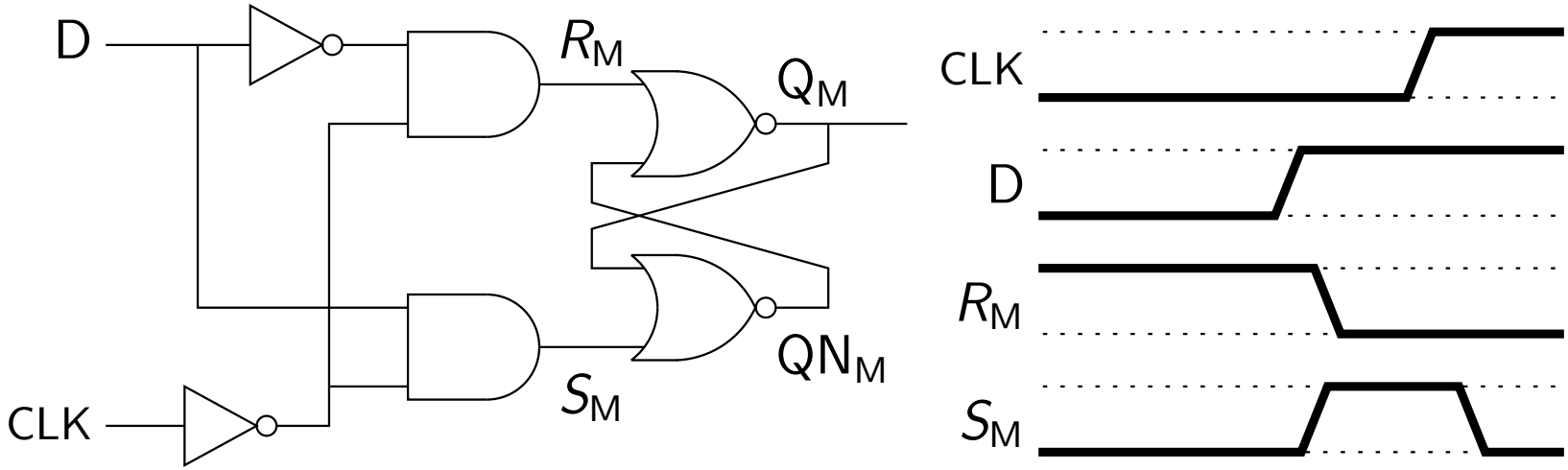
Metastability in a D flip-flop

We studied this design in lectures. It's not how D flip-flops are constructed in most modern integrated circuits, but the design is relatively easy to understand.



Reminder: Because CLK passes through a NOT gate before entering the master latch, the master latch is transparent when CLK is LOW.

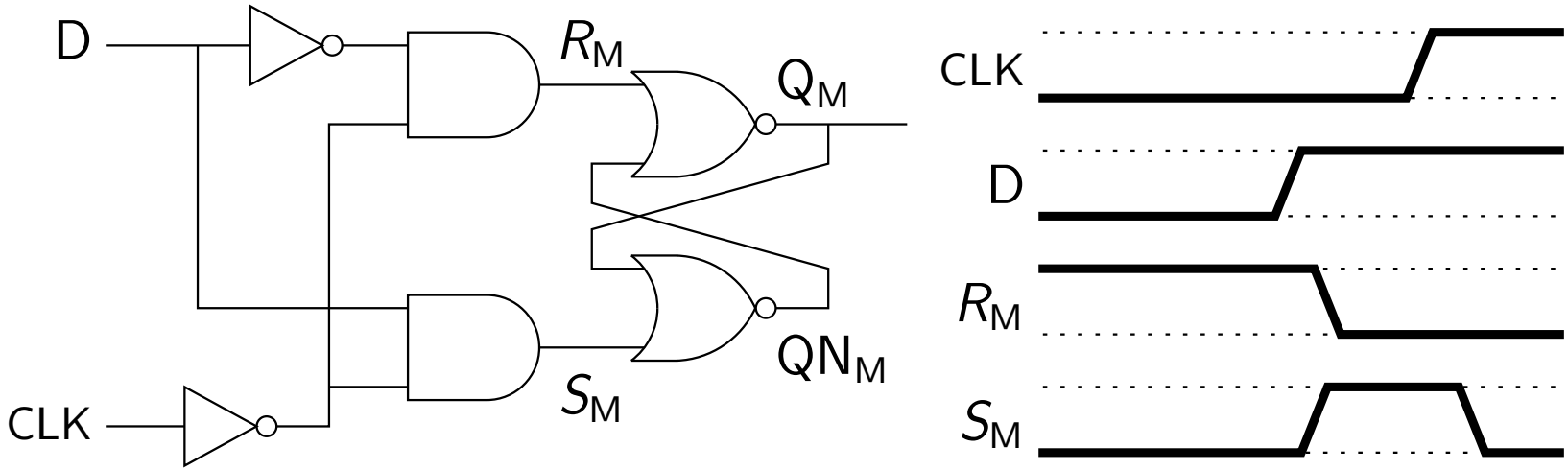
Let's look at what happens in the master latch when there is an edge on D just before a rising edge on CLK.



In an **ideal** SR latch, (Q_M, QN_M) will go to $(1, 0)$ because of the pulse on S_M. But in a **real** SR latch, the **width** of the pulse matters.

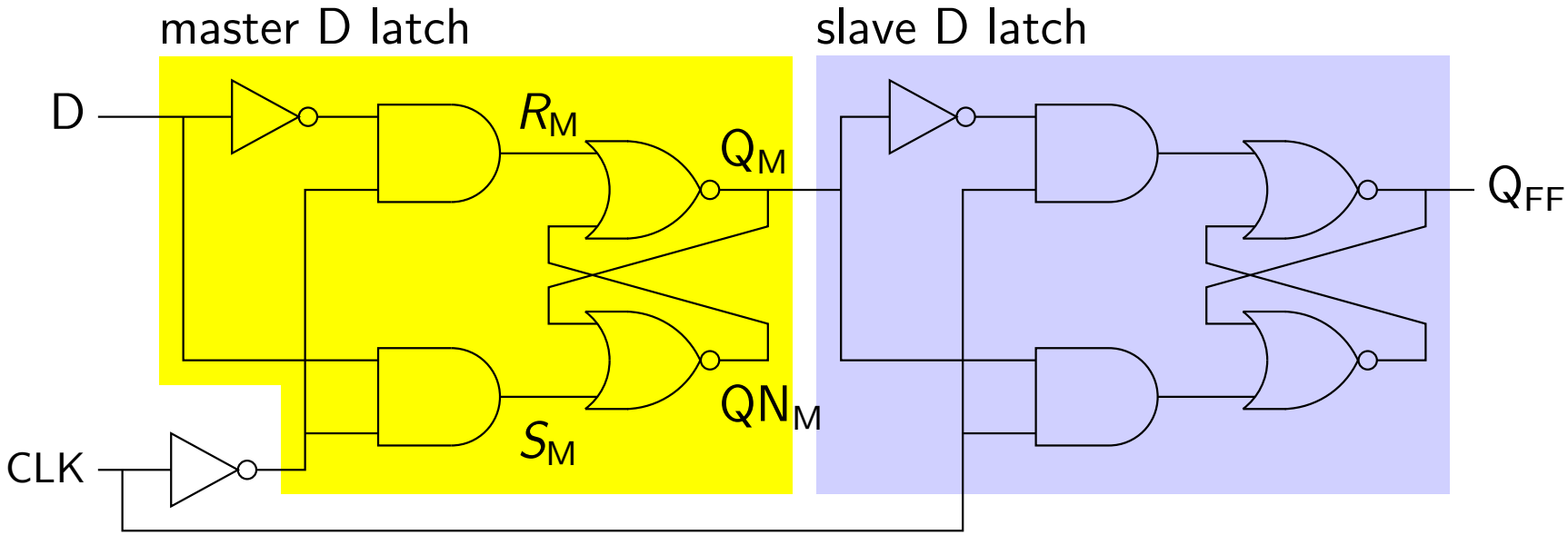
A **wide** pulse will make the latch "do the right thing."

A **very narrow** pulse will not decrease the voltage at QN_M very much, and the voltage at Q_M won't change at all. (Q_M, QN_M) will settle back to $(0, 1)$.



If the width of the pulse on S_M is "perfectly wrong" the voltages at Q_M and QN_M will both be near $0.5V_{DD}$ when the pulse ends. That gives the pair of NOR gates a chance to go **metastable!**

Now let's have another look at the slave latch . . .

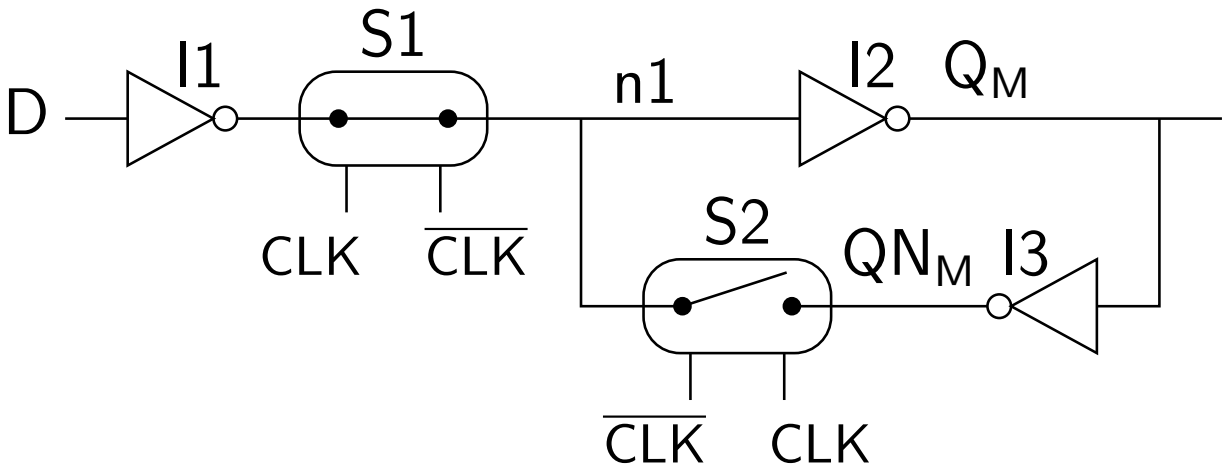


When CLK is HIGH, the slave latch is supposed to be transparent. But if $Q_M \approx 0.5 V_{DD}$, both AND gates in the slave have "forbidden zone" inputs, and we cannot rely on Q_{FF} to have any particular value.

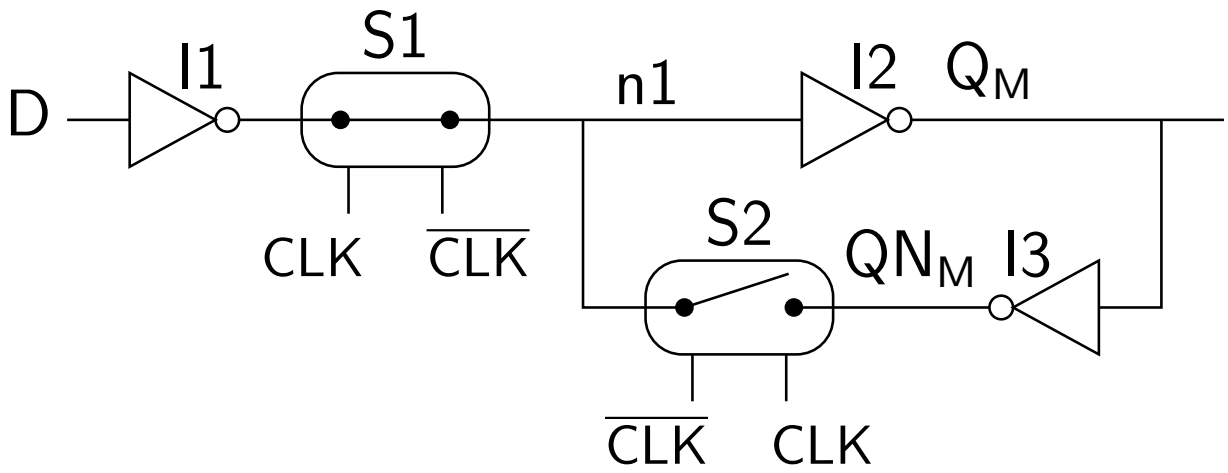
If it's still true that $Q_M \approx 0.5 V_{DD}$ when CLK goes from HIGH to LOW, that could make the NOR gates in the slave go metastable.

Setup time in a modern DFF

Below is a D latch suitable for use as the master latch in a DFF. Switches S1 and S2 are implemented in CMOS using simple two-transistor devices called **transmission gates**.

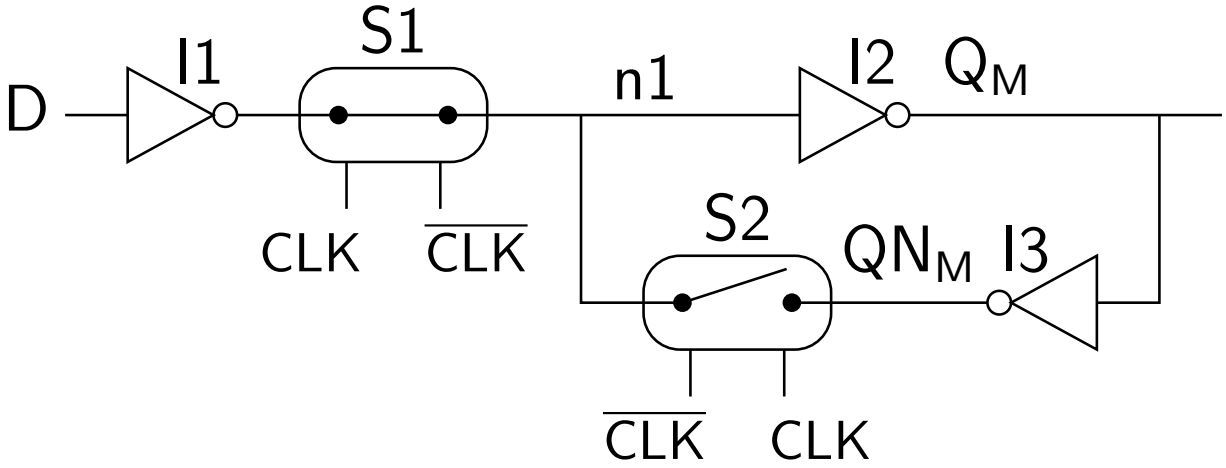


CLK	S1	S2	latch condition
LOW	closed	open	transparent
HIGH	open	closed	opaque



When CLK is LOW, S1 is closed and S2 is open. I1 and I2 form a simple buffer.

When CLK is HIGH, S1 is open and S2 is closed. I2 and I3 form a **bistable** circuit that can lock the state in either $(Q_M, QN_M) = (0, 1)$ or $(Q_M, QN_M) = (1, 0)$.



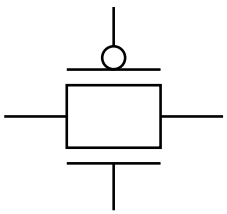
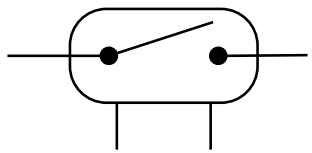
What happens if there is an edge on D, just before a rising edge on CLK that will break the connection though S1?

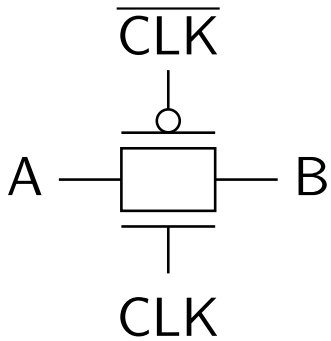
I1 may not have enough time to drive node n1 to the correct voltage for $\overline{D}$, with one of two possible bad outcomes:

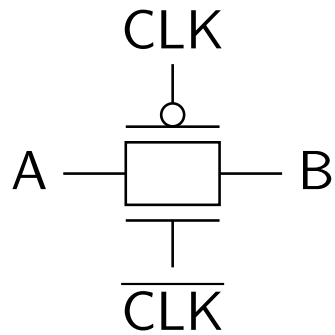
- Q_M could fail to copy the new value of D.
- If the n1 voltage is "just wrong", the bistable circuit made from I2 and I3 could go metastable.

If the signal at D respects a **setup time** specification, then the above bad outcomes can't happen.

Some connections to the textbook and other literature


, not

, is the usual symbol for a transmission gate.





On the left, A and B are connected when CLK is HIGH. On the right, A and B are connected when CLK is LOW.

More connections to the textbook and other literature

Here is the complete DFF design. Note that the inverting tristate buffers T1 and T2 are really just inverters followed by transmission gates.

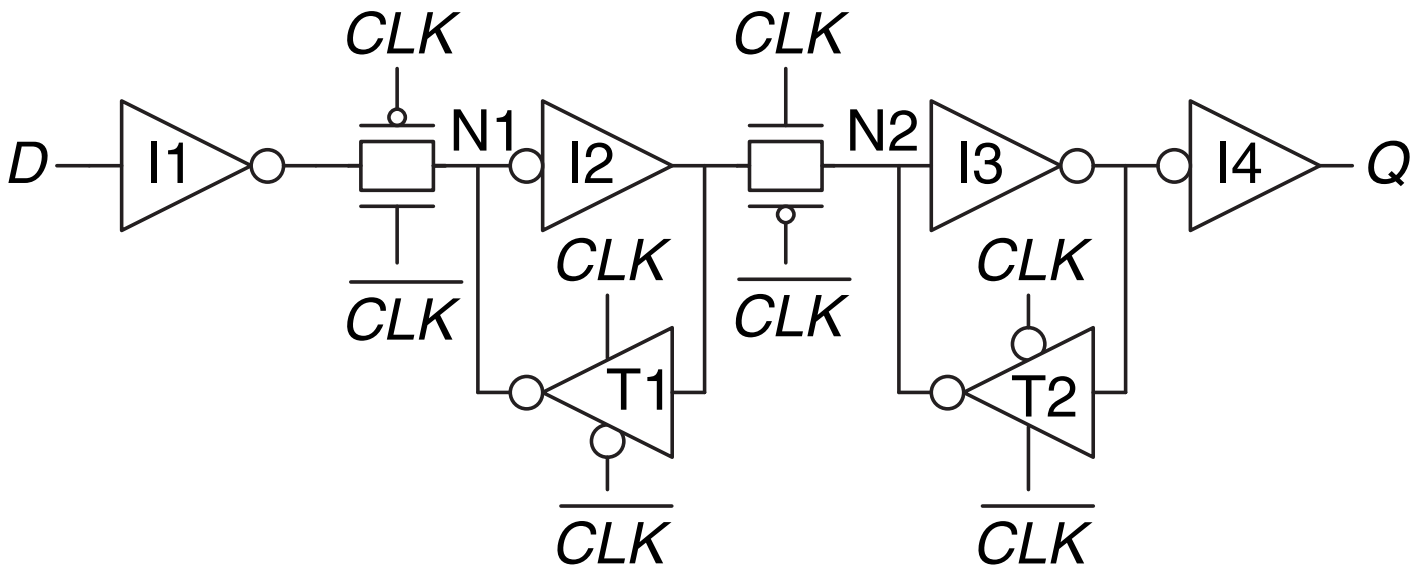


Image is Figure 3.13 from Harris D. M. and Harris S. L., *Digital Design and Computer Architecture, 2nd ed.*, © 2013, Elsevier, Inc.