Chapter 11: Flip-Flops

Computer Structure - Spring 2005

©Dr. Guy Even

Tel-Aviv Univ.

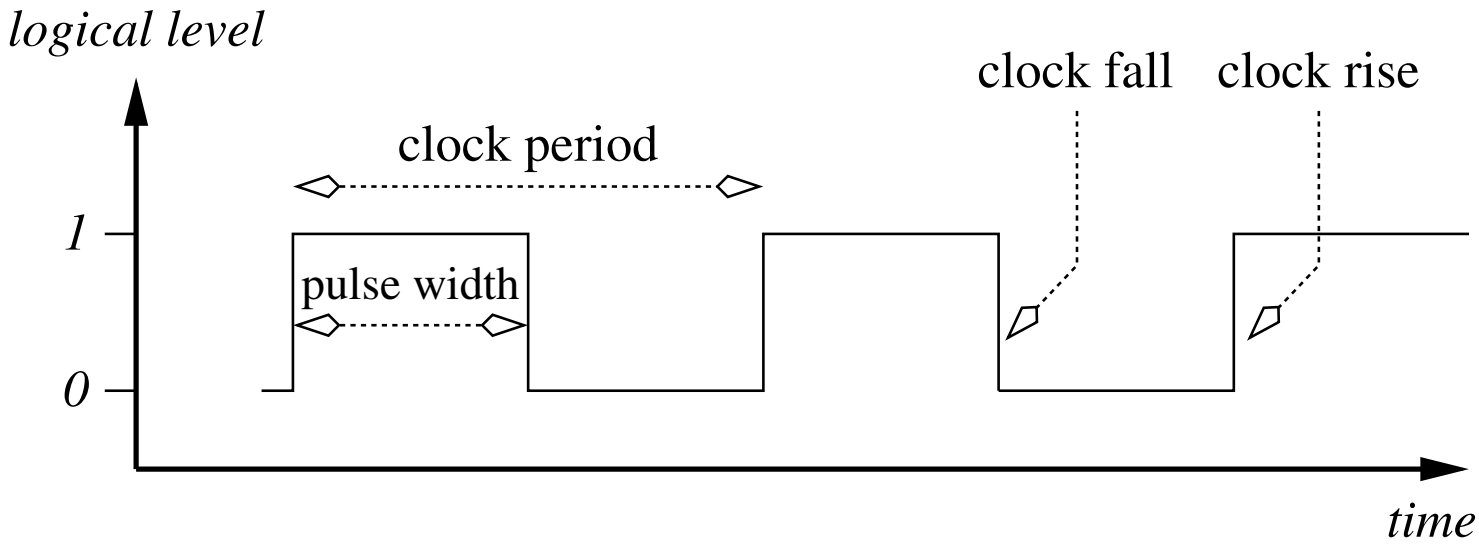
Preliminary Questions

- How is time measured in a synchronous circuit?
- What is the functionality of a flip-flop?
- What is a stable state? How many stable states does a flip-flop have?
- How does a flip-flop move from one stable state to another? How fast is this transition?

Goals

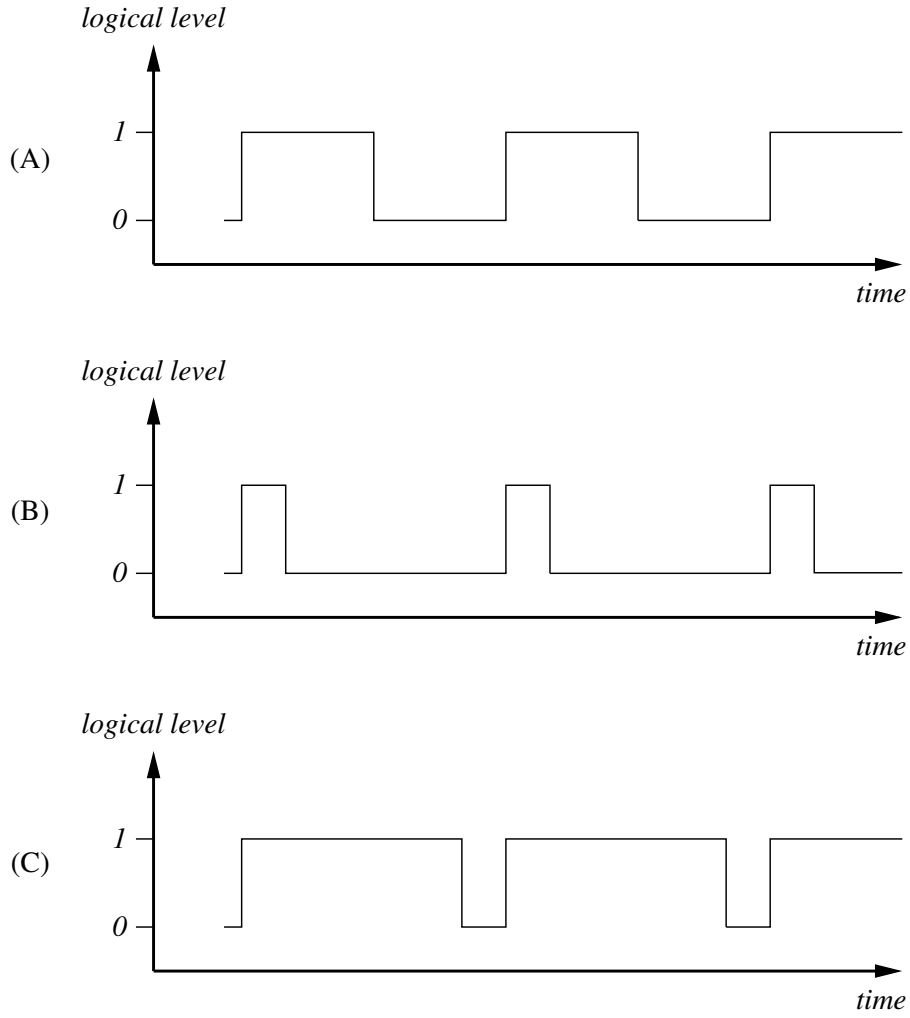
- introduce clock signal.
- define edge-triggered flip-flops.
- discuss parameters of flip-flops: setup time, hold time, contamination delay, propagation delay.
- explain importance of critical segment.
- understand timing of a flip-flop.
- other memory devices.

The Clock



- digital signal with periodic oscillations between 0 and 1.
- transitions are instantaneous.
- each clock period starts with a $0 \rightarrow 1$ transition.
- $1 \rightarrow 0$ transition in the interior of the clock period.
- we denote the clock signal by CLK.

Clock terminology



- clock period - denoted by $\varphi(\mathsf{CLK})$.
- clock pulse - interval during which $\mathsf{CLK}(t) = 1$.
- CLK_{pw} - duration of clock pulse.
- symmetric clock - if $\mathsf{CLK}_{pw} = \varphi(\mathsf{CLK})/2$.
- narrow pulses - if $\mathsf{CLK}_{pw} < \varphi(\mathsf{CLK})/2$.
- wide pulses - if $\mathsf{CLK}_{pw} > \varphi(\mathsf{CLK})/2$.

Clock cycles

A clock partitions time into discrete intervals as follows:

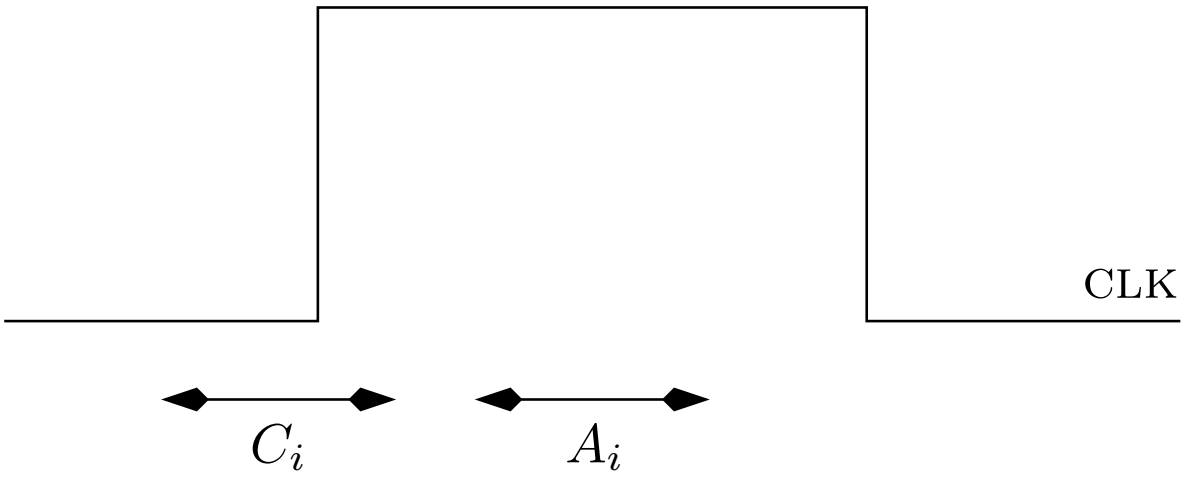
- Let t_i denote the starting time of the ith clock period.
- We refer to the half-closed interval $[t_i, t_{i+1})$ as clock cycle i.

Parameters of an Edge-triggered Flip-Flop

- Setup-time denoted by t_{su},
- Hold-time denoted by t_{hold},
- Contamination-delay denoted by t_{cont},
- Propagation-delay denoted by t_{pd}.

These parameters satisfy $-t_{su} < t_{hold} < t_{cont} < t_{pd}$.
Notation:

- critical segment: $C_i = [t_i - t_{su}, t_i + t_{hold}]$.
- instability segment: $A_i = [t_i + t_{cont}, t_i + t_{pd}]$.

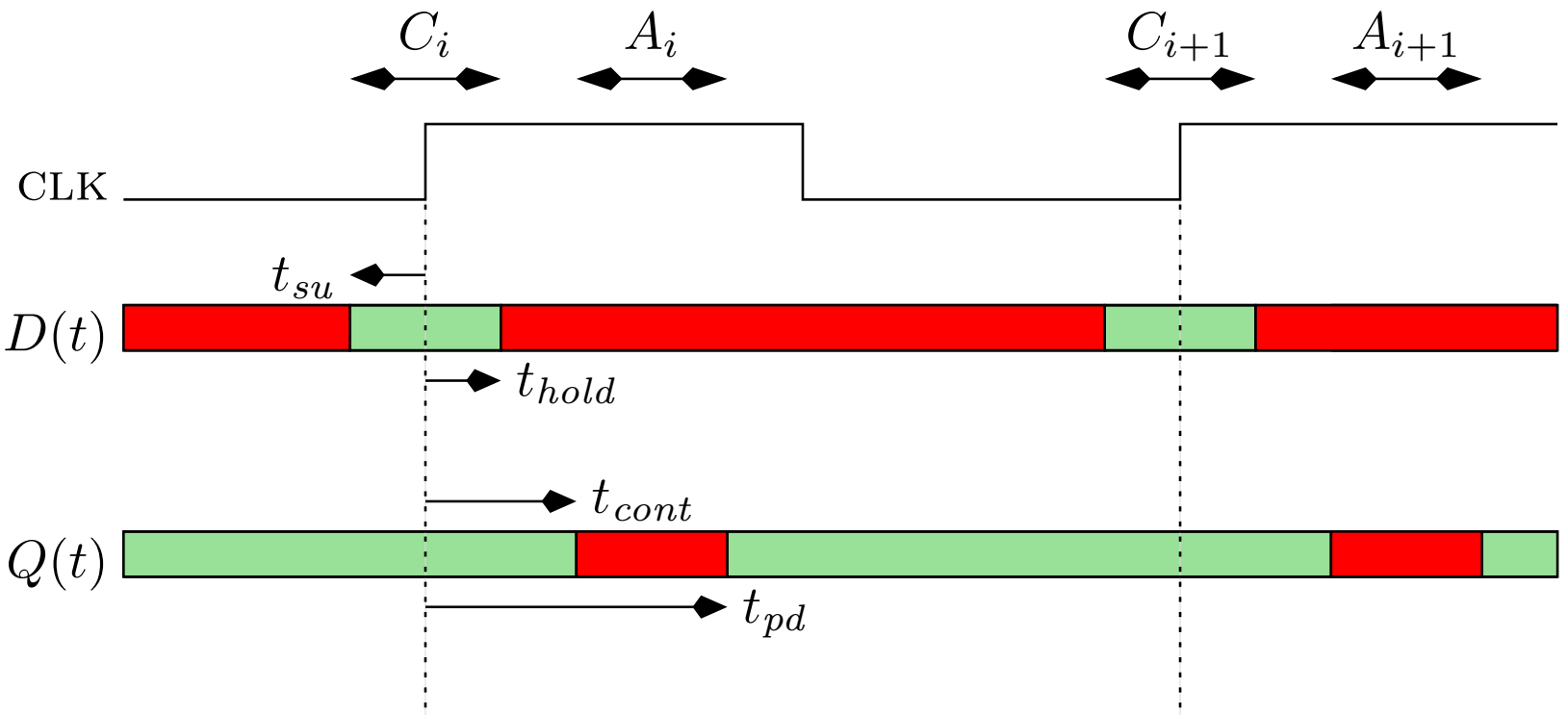


Definition: Edge-triggered Flip-Flop

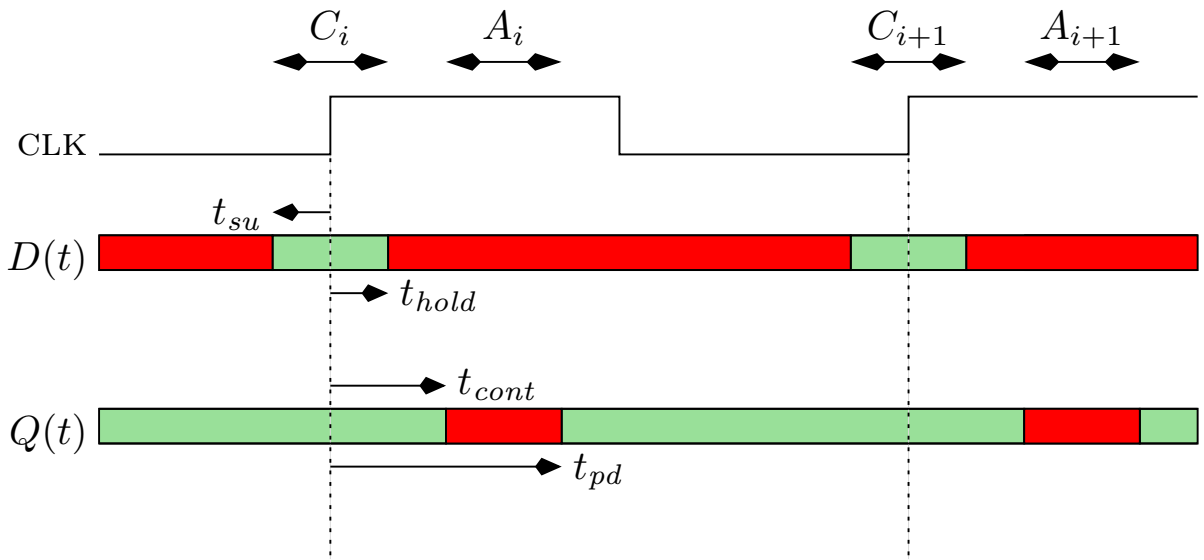
Inputs: A digital signal $D(t)$ and a clock CLK.

Output: A digital signal $Q(t)$.

Functionality: If $D(t)$ is stable during the critical segment C_i, then $Q(t) = D(t_i)$ during the interval $(t_i + t_{pd}, t_{i+1} + t_{cont})$.

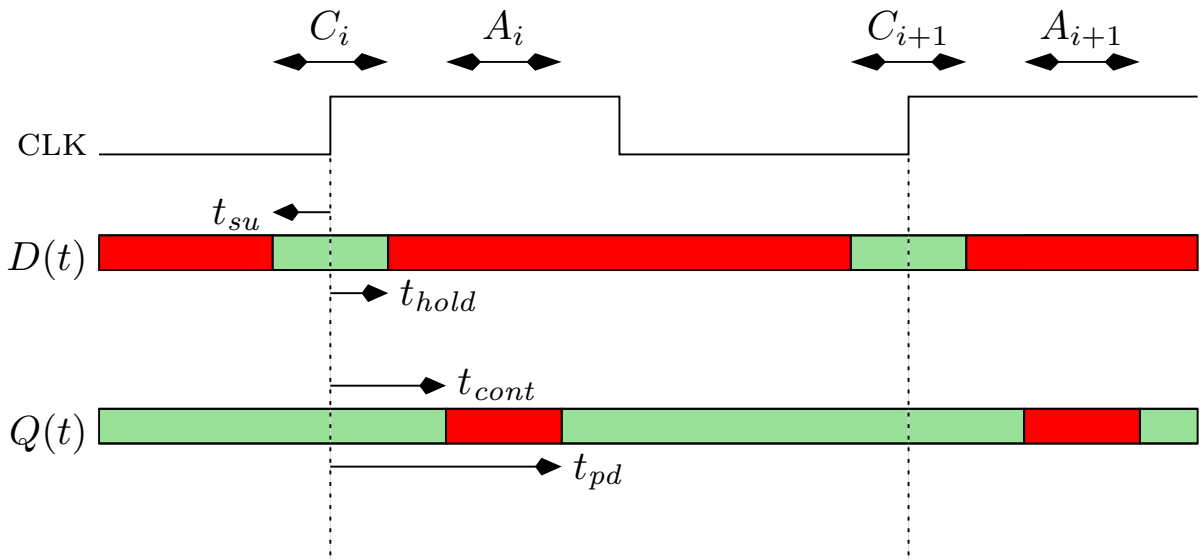


Remarks on definition of flip-flop



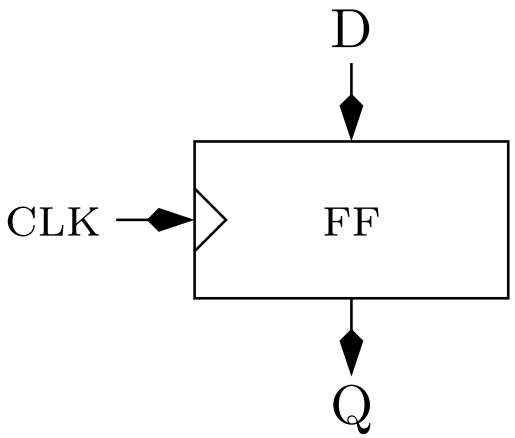
- $-t_{su} < t_{hold} < t_{cont} < t_{pd} \implies C_i \cap A_i = \emptyset$.
- Stability of $D(t)$ during $C_i \Rightarrow$ digital value of $D(t)$ during the critical segment C_i is logical and equals $D(t_i)$.
- Flip-flop samples $D(t)$ during C_i. The sampled value $D(t_i)$ is output during the interval $[t_i + t_{pd}, t_{i+1} + t_{cont}]$.
- Sampling is successful only if $D(t)$ is stable while it is sampled. This is why we refer to C_i as a critical segment.

Remarks on definition of flip-flop - cont.



- If the input $D(t)$ is stable during the critical segments $\{C_i\}_i$, then the output $Q(t)$ is stable in between the instability segments $\{A_i\}_i$.
- The stability of the input $D(t)$ during the critical segments depends on the clock period. We will later see that slowing down the clock (i.e. increasing the clock period) helps in achieving a stable $D(t)$ during the critical segments.

schematic of an edge triggered flip-flop



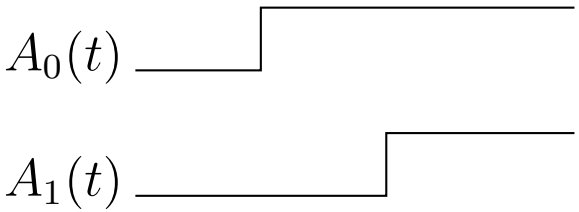
- clock port is marked by an “arrow”.
- we abbreviate and refer to an edge-triggered flip-flop simply as a flip-flop.

Question: Prove that an edge-triggered flip-flop is not a combinational circuit.

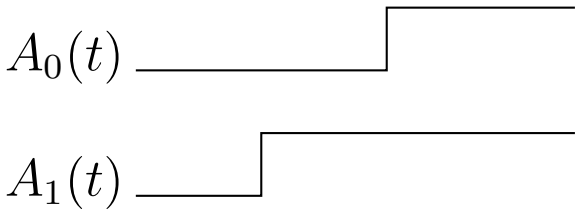
Arbitration

Arbitration is the problem of deciding which event occurs first.

Focus on the task of determining which of two signals reaches 1 first.



$A_0(t)$ reaches 1 first



$A_1(t)$ reaches 1 first

Definition: arbiter

Inputs: Non-decreasing analog signals $A_0(t), A_1(t)$ defined for every $t \geq 0$.

Output: An analog signal $Z(t)$.

Functionality: Assume that $A_0(0) = A_1(0) = 0$. Define T_i, for $i = 0, 1$, as follows:

$$T_i \triangleq \inf\{t \mid \mathit{dig}(A_i(t)) = 1\}.$$

Let $t' \triangleq 10 + \max\{T_0, T_1\}$. The output $Z(t)$ must satisfy, for every $t \geq t'$,

$$\mathit{dig}(Z(t)) = \begin{cases} 0 & \text{if } T_0 < T_1 - 1 \\ 1 & \text{if } T_1 < T_0 - 1 \\ 0 \text{ or } 1 & \text{otherwise.} \end{cases}$$

Arbiter - remarks

- $$T_i \triangleq \inf\{t \mid \mathit{dig}(A_i(t)) = 1\}.$$

  If T_0 or T_1 equals infinity, then $t' = \infty$, and there is no requirement on the output $Z(t)$.

- Arbiter circuit is given 10 time units starting from $\max\{T_0, T_1\}$ to determine if $T_0 < T_1$ or $T_1 < T_0$.
- tie: the case that $|T_0 - T_1| \leq 1$.
- In the case of a tie, the arbiter is free to decide, but must decide. $Z(t)$ is stable in the interval $[t, \infty)$.

Arbiters - an impossibility result

Claim: There does not exist a circuit C that implements an arbiter.

- Inherent limitation - not just a weakness of the digital abstraction.
- Use the claim to show that flip-flops must have critical segments.

Proof: every circuit C is not an arbiter

- Define $A_0(t)$ so that $T_0 = 100$ as follows:

$$A_0(t) \triangleq \begin{cases} \frac{t}{100} \cdot V_{high,in} & \text{if } t \in [0, 100] \\ V_{high,in} & \text{if } t > 100. \end{cases}$$

- Fix a parameter $x \in [-2, 2]$ and define $A_1(t)$ so that $T_1 = 100 + x$ as follows:

$$A_1(t) \triangleq \begin{cases} \frac{t}{100+x} \cdot V_{high,in} & \text{if } t \in [0, 100 + x] \\ V_{high,in} & \text{if } t > 100 + x. \end{cases}$$

- Define the function $f(x)$ by $f(x) \triangleq Z(200)$.
- We study the function $f(x)$ in the interval $x \in [-2, 2]$.

Proof: every circuit C is not an arbiter - cont.

- $x = -2 \Rightarrow T_1 = 100 + x = 98$. It follows that $A_1(t)$ "wins", and *dig*$(Z(200)) = 1$. Hence $f(-2) \geq V_{\mathit{high,out}}$.
- $x = 2 \Rightarrow T_1 = 100 + x = 102$. It follows that $A_0(t)$ "wins", and *dig*$(Z(200)) = 0$. Hence $f(2) \leq V_{\mathit{low,out}}$.
- claim: $f(x)$ is continuous (will prove this later).
- Mean Value theorem $\Rightarrow$

$$\forall y \in [V_{\mathit{low,out}}, V_{\mathit{high,out}}] \;\; \exists x \in [-2, 2] \;\; : \;\; f(x) = y.$$

- Pick y such that *dig*$(y) =$ non-logical.
- $\Rightarrow$ There exist valid inputs $A_0(t), A_1(t)$ with $t' \leq 112$, such that *dig*$(Z(200)) =$non-logical.
- $\Rightarrow C$ is not an arbiter. QED.

Proof: $f(x)$ is continuous

Rely on the assumption that an infinitesimal change in the energy of input signals causes an infinitesimal change in the energy of the output. Otherwise, noise would cause uncontrollable changes in $Z(t)$ and the circuit C would not be useful anyhow.

The output $Z(200)$ depends on the following:

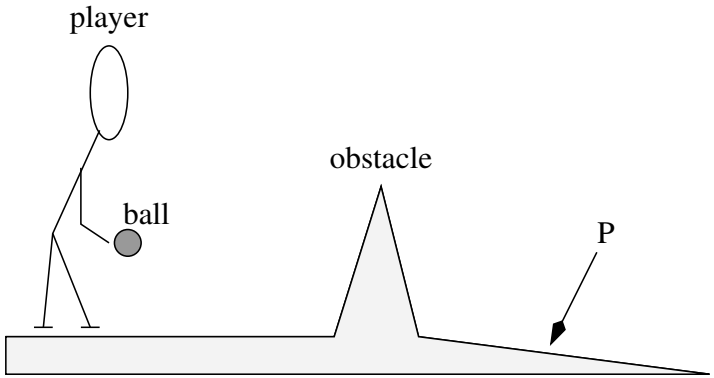
1. The initial state of the device C at time $t = 0$. We assume that the device C is in a stable state and that the charge is known everywhere.
2. The signal $A_i(t)$ in the interval $[0, 200]$, for $i = 0, 1$.

Proof: $f(x)$ is continuous - cont.

- Consider an infinitesimal change in x. This change affects $A_1(t)$ but does not affect $A_0(t)$ and the initial state.
- infinitesimal change of $x \Rightarrow$ infinitesimal difference in energy of $A_1(t)$.
- infinitesimal difference in energy of $A_1(t) \Rightarrow$ infinitesimal difference in $Z(200)$.
- $\Rightarrow f(x)$ is continuous.

Discussion: Arbiters - an impossibility result

- Claim is counter-intuitive.
- For every judge in a 100-meter dash, there exist two runners whose running times are such that the judge still hangs after an hour.
- Implies that there does not exist a perfect judge who can determine the winner in a 100-meters dash even if:
  1. high speed cameras located at the finish line and runners run very slowly.
  2. we allow the judge several hours to decide.
  3. we allow the judge to decide arbitrarily if the running times of the winner and runner-up are within a second.

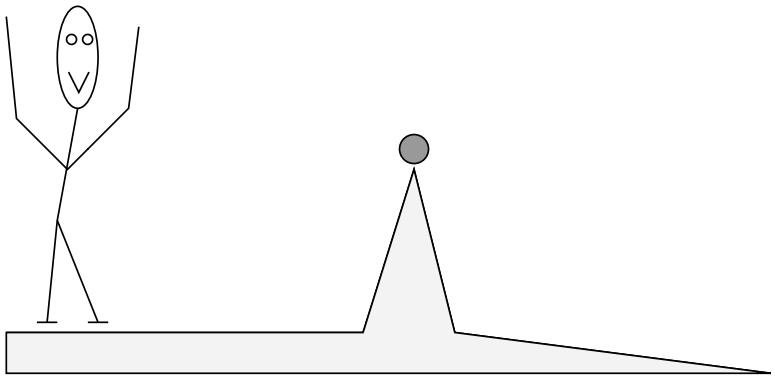


- Player - rolls a ball. Judge - announces decision if ball passes point P one day after.
- If speed of ball is above v', then ball passes the obstacle and then rolls past point P.
- If speed of ball is below v', then ball does not pass the obstacle.

Judge is in trouble:

- If speed$= v'$, then the ball reaches the tip of the obstacle and may remain there indefinitely long!
- If the ball remains on the obstacle's tip 24 hours past the throw, then the judge cannot announce her decision.

Meta-stability



- Meta-stability - a state of equilibrium (i.e. zero force) which is not a local minimum of energy (i.e. a slight force causes a movement away from the state).
- Inclined to say that the "probability of meta-stability occurring is very small". This requires a probability distribution over the rolling speed v where

$$\lim_{\varepsilon \to 0} \mathit{Pr}(|v - v'| < \varepsilon) = 0.$$

Lessons learned

- Certain tasks are not achievable with probability 1.
  - coin toss might end up with the coin standing on its perimeter.
  - noise could be big enough to cause the digital value of a signal to flip from zero to one. (increase noise margin to reduce the probability of such an event.)

Reducing the probability of meta-stability

- Increase length of segment of instability. Increasing the delay of the arbiter (significantly) decreases the chances of meta-stability. E.g., ball resting on the tip of the obstacle is likely to fall to one of the sides.
- Increase the slope of the transfer function in the range of non-logical values. Similar to sharpening the tip of the obstacle.
- However, increasing the clock rate means that “decisions” must be made faster (i.e. within a clock period) and the chance of meta-stability increases.

Question

Does the proof of the Claim hold only if the signals $A_i(t)$ rise gradually?

Question: Prove the claim with respect to "fast" non-decreasing signals $A_i(t)$. Namely, the length of the interval during which *dig*$(A_i(t))$ is non-logical equals ε.

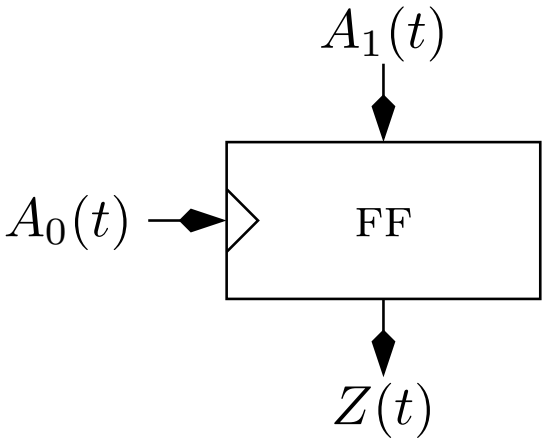
Flip-flops: necessity of critical segments

DEF: A flip-flop without a critical segment is a flip-flop in which the setup-time and hold-time satisfy $t_{su} = t_{hold} = 0$. The functionality is defined as follows:

- For every i, $Q(t)$ is logical (either zero or one) during the interval $t \in (t_i + t_{pd}, t_{i+1} + t_{cont})$ regardless of whether $D(t_i)$ is logical.
- If $D(t_i)$ is logical, then $Q(t) = D(t_i)$ during the interval $t \in (t_i + t_{pd}, t_{i+1} + t_{cont})$.

Just as the arbiter's decision is free if a tie occurs, the flip-flop is allowed to output either zero or one if $D(t_i)$ is not logical. However, the output of the flip-flip must be logical once the instability segment ends.

An arbiter based on a flip-flop without a critical segment



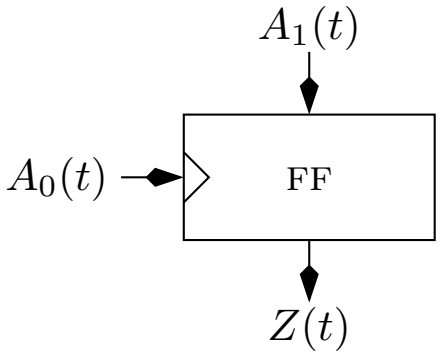
Assumptions:

- flip-flop is without a critical segment.
- $t_{cont}, t_{pd} \approx 10^{-9}$ time unit.
- intervals during which the inputs $A_0(t)$ and $A_1(t)$ are non-logical are also very short (e.g. 10^{-9} time unit).

Claim: The circuit above is an arbiter.

CORO: There does not exist a flip-flop without a critical section.

Remarks



- the signal $A_0(t)$ is input as a clock to the flip-flop, but $A_0(t)$ is not a clock.
- requirements from $A_0(t)$ are weaker than the requirements from a clock. Instead of periodic instantaneous transitions from zero to one and back, $A_0(t)$ is non-decreasing.
- the claim assumes only one “tick of the clock”, so we may regard $A_0(t)$ as a clock with a very long period.
- proof of claim does not rely on $A_0(t)$ rising slowly; the claim holds regardless of the rate of change of $A_0(t)$.

Proof that circuit is an arbiter

We consider three cases:

- $|T_1 - T_0| \leq 1$: flip-flop's output $Z(t)$ is always logical at time $T_0 + t_{pd}$, so circuit functions properly.
- $T_1 < T_0 - 1$: if $T_1 < T_0 - 1$, then *dig*$(A_1(T_0)) = 1$. Hence sampled value equals 1, and hence, *dig*$(Z(t)) = 1$, for every $t \geq T_0 + t_{pd}$.
- $T_0 < T_1 - 1$: we claim that *dig*$(A_1(T_0)) = 0$, and hence, *dig*$(Z(t)) = 0$, for every $t \geq T_0 + t_{pd}$.

Proof that circuit is an arbiter - cont.

We need to show that $T_0 < T_1 - 1 \Rightarrow$ *dig*$(A_1(T_0)) = 0$.

- $T_0 < T_1 \Rightarrow$ *dig*$(A_1(T_0)) \in \{0,$ non-logical$\}$.
- assumption on the fast transition of *dig*$(A_1(t))$ implies:

$$\textit{dig}(A_1(T_0)) = \text{non-logical} \;\Rightarrow\; \textit{dig}(A_1(T_0 + 10^{-9})) = 1.$$

Hence, $T_1 \leq T_0 + 10^{-9}$ contradicting $T_1 > T_0 + 1$.

It follows that if $T_0 < T_1 - 1$, then *dig*$(A_1(T_0)) = 0$. QED

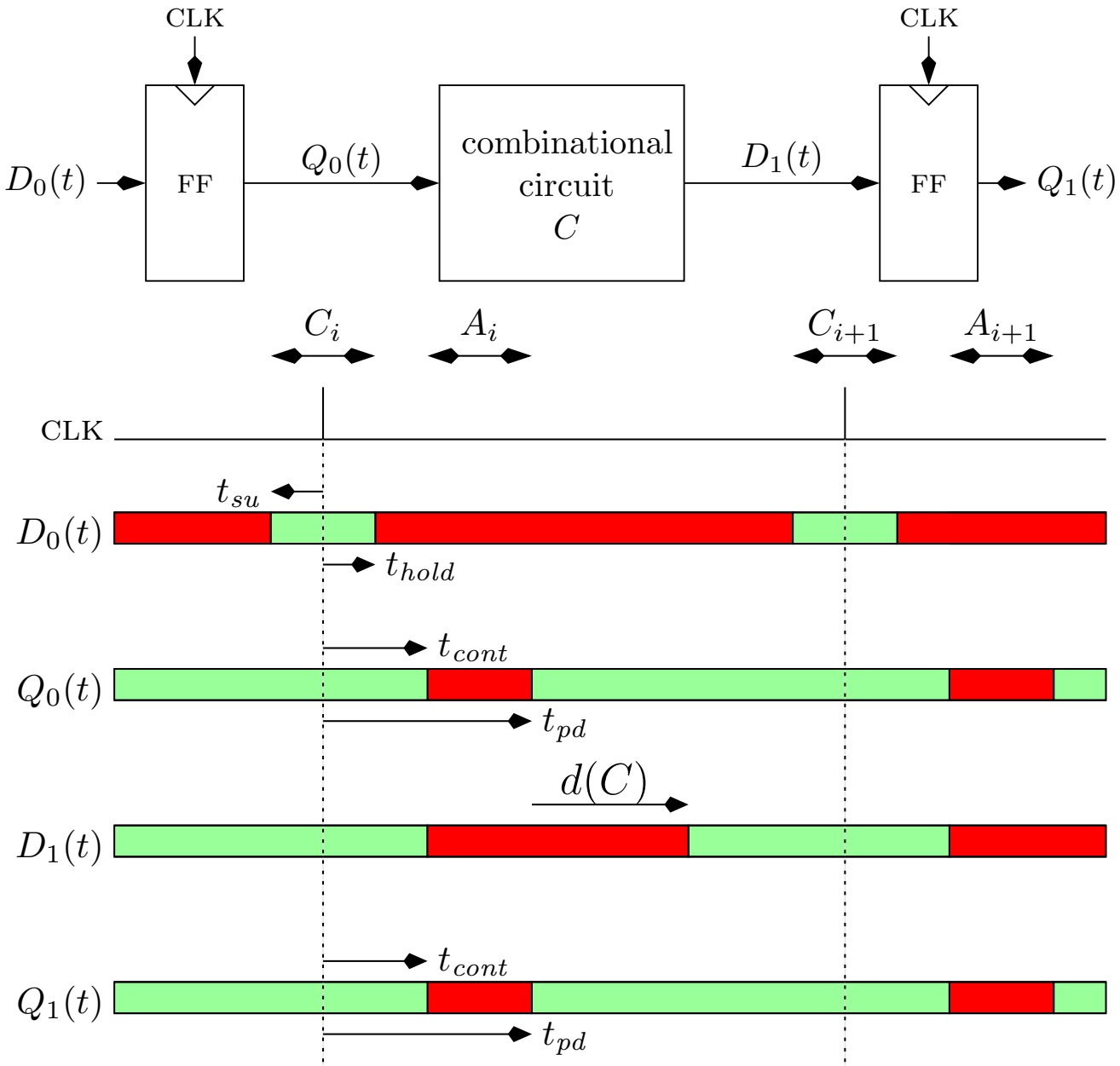
Corollary: conclusion

Critical segment is required to avoid meta-stability of the flip-flop.

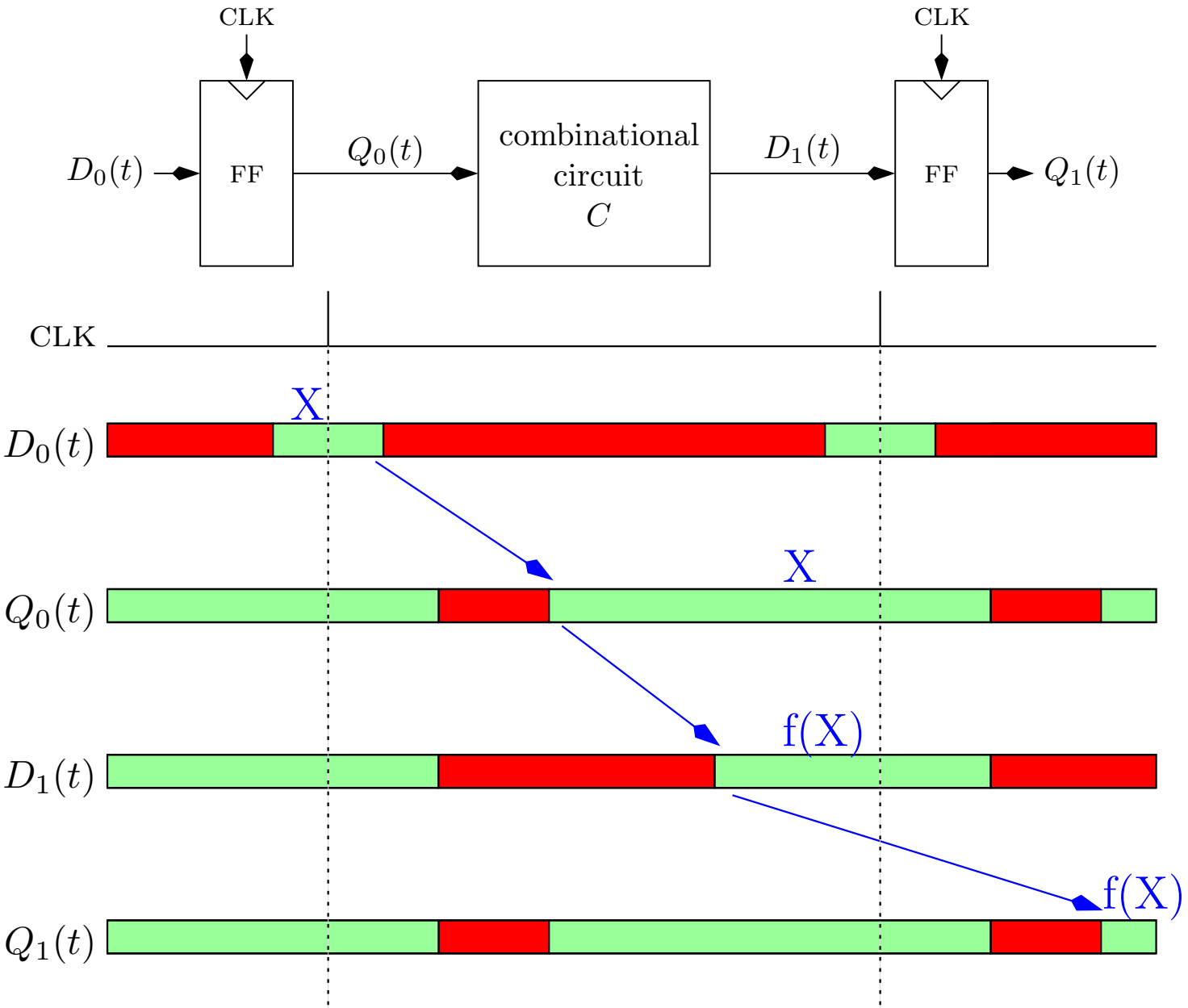
Without critical segment, flip-flop's output can be non-logical

even after $t_i + t_{pd}$.

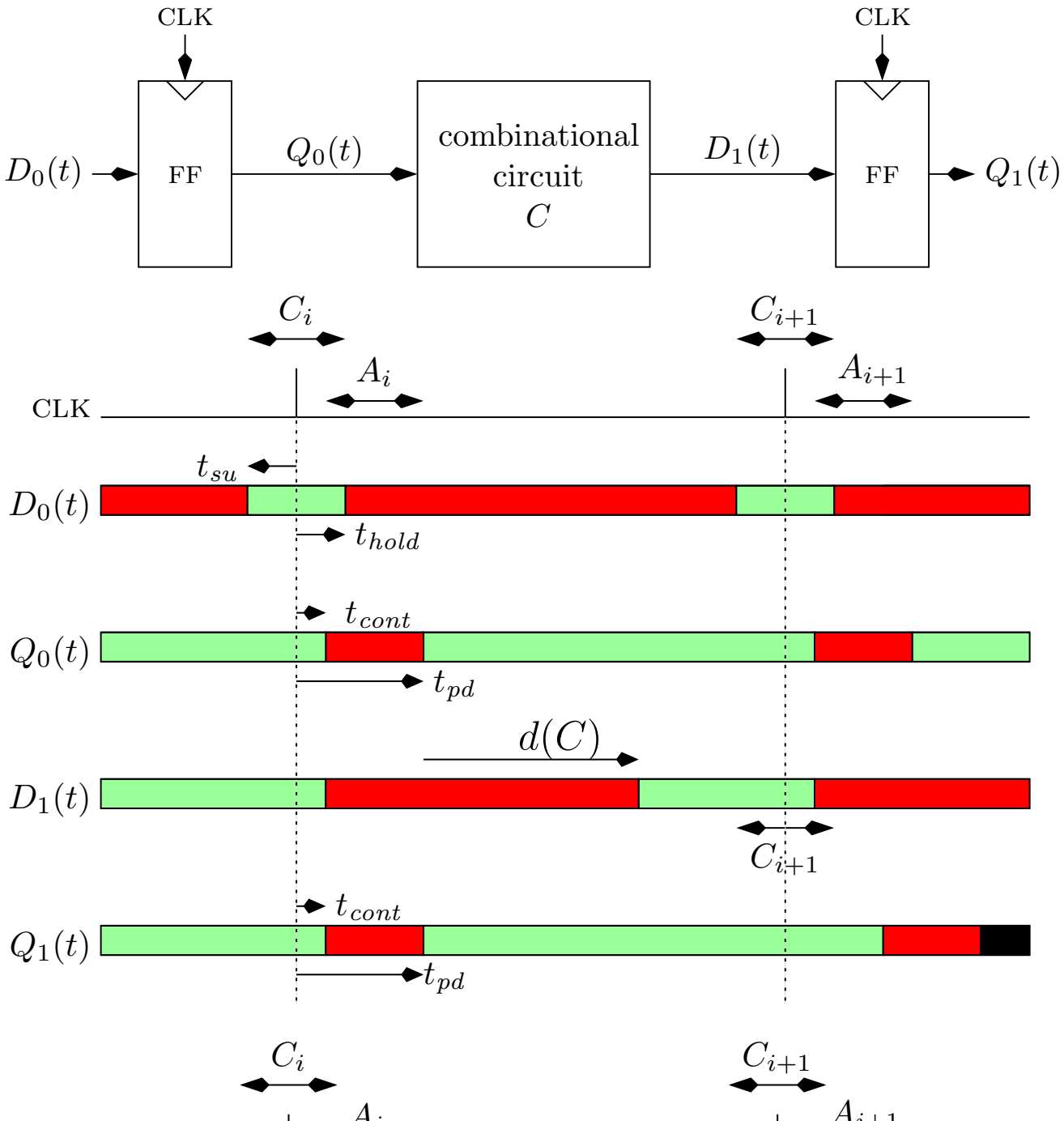
An example: timing



An example: functionality



Non-disjoint segments: $A_i \cap C_i \neq \emptyset$



What if $A_i \cap C_i \neq \emptyset$?

Stability interval of $D_1(t)$ is:

$$[t_i + t_{pd} + d(C), t_{i+1} + t_{cont}].$$

If $t_{cont} < t_{hold}$, then $D_1(t)$ is not stable during

$$C_{i+1} = [t_{i+1} - t_{su}, t_{i+1} + t_{hold}].$$

In this case, we need to rely on the contamination delay $cont(C)$ of the combinational circuit C.

Now $D_1(t)$ is stable during the interval

$$[t_i + t_{pd} + d(C), t_{i+1} + t_{cont} + cont(C)].$$

If $t_{cont} + cont(C) > t_{hold}$, then the signal $D_1(t)$ is stable during the critical segment C_{i+1}, and correct functionality is obtained.

Contamination delay of combinational circuits

- Can help in obtaining stability during the critical segment.
- Many combinational gates have a positive contamination delay. But some don't.
- Relying on the contamination delay of combinational circuits complicates timing analysis.
- We use a strict assumption that *cont*$(C) = 0$, for every combinational circuit C. This does not cause incorrect circuits even if *cont*$(C) > 0$.

Fixing $A_i \cap C_i \neq \emptyset$

Question: Assume that we have an edge-triggered flip-flop FF in which $t_{hold} > t_{cont}$. Suppose that we have an inverter with a contamination delay $cont(\mathsf{INV}) > 0$.

- Suggest how to design an edge-triggered flip-flop FF′ that satisfies $t_{hold}(\mathsf{FF}') < t_{cont}(\mathsf{FF}')$.
- What are the parameters of FF′?

D-Latch: parameters

- characterized by two parameters t_{su}, t_{hold}
- the critical segment is defined with respect to the falling edge of the clock.
- t'_i - time of the falling edge of the clock during the ith clock cycle.
- critical segment of a D-latch is

$$[t'_i - t_{su}, t'_i + t_{hold}].$$

- d - combinational delay of the D-latch.

D-Latch: definition

- During the interval $[t_i + d, t'_i)$, the output $Q(t)$ satisfies: $Q(t) = D(t)$, provided that $D(t)$ is stable during the interval $[t - d, t]$. We say that the D-latch is transparent during the interval $[t_i + d, t'_i)$.
- During the interval $(t'_i + t_{hold}, t_{i+1})$, if $D(t)$ is stable during the critical segment $[t'_i - t_{su}, t'_i + t_{hold}]$, then $Q(t) = D(t'_i)$. We say that the D-latch is opaque during the interval $(t'_i + t_{hold}, t_{i+1})$.

D-Latch : story

- D-latches are very important devices.
- D-latches are cheaper than flip-flops, and in fact, D-latches are the building blocks of flip-flops (e.g. master/slave designs).
- using D-latches wisely leads to faster designs.
- designs based on D-latches require multiple clock phases (or at least a clock CLK and its negation $\overline{\mathsf{CLK}}$).
- Although timing with multiple clock phases is an important and interesting topic, we do not deal with it in this course.

Definition : clock enabled flip-flips

Inputs: Digital signals $D(t), \mathrm{CE}(t)$ and a clock CLK.

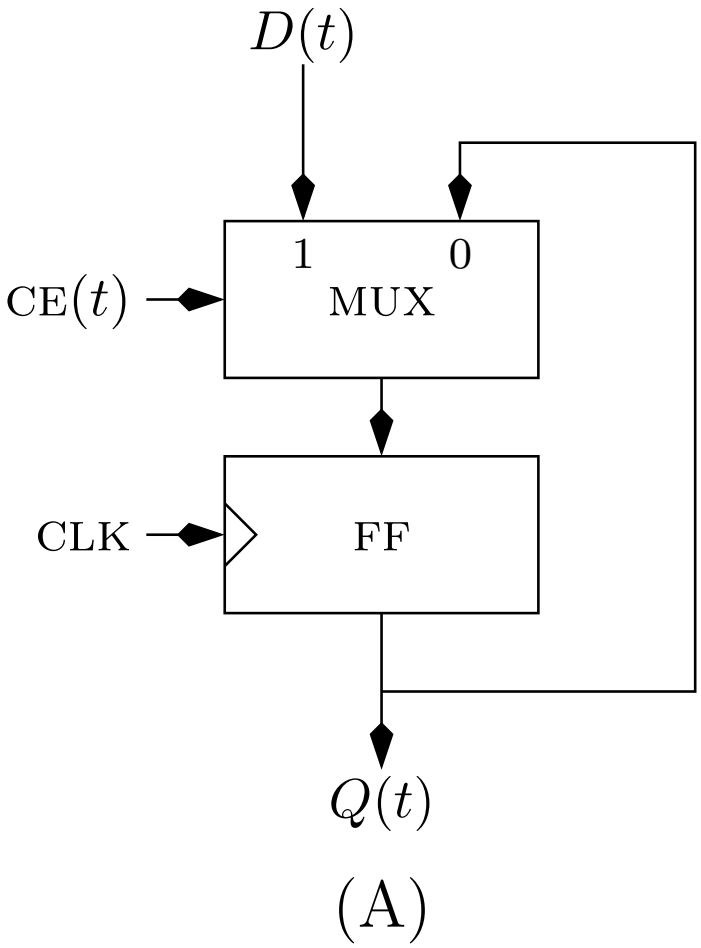
Output: A digital signal $Q(t)$.

Functionality: If $D(t)$ and $\mathrm{CE}(t)$ are stable during the critical segment C_i, then for every $t \in (t_i + t_{pd}, t_{i+1} + t_{cont})$

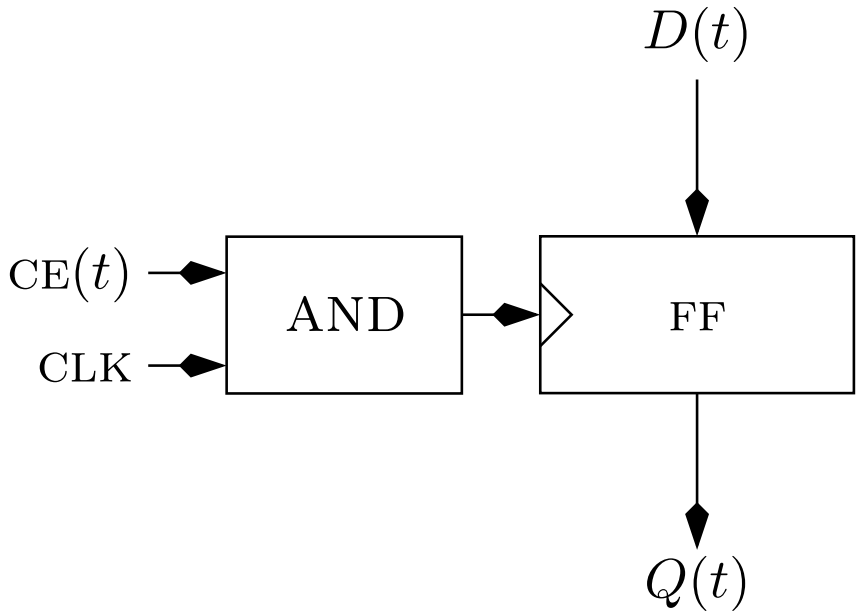
$$Q(t) = \begin{cases} D(t_i) & \text{if } \mathrm{CE}(t_i) = 1 \\ Q(t_i) & \text{if } \mathrm{CE}(t_i) = 0. \end{cases}$$

- $\mathrm{CE}(t)$ - clock-enable signal.
- $\mathrm{CE}(t)$ indicates whether the flip-flop samples the input $D(t)$ or maintains its previous value.

Clock enabled flip-flips : implementation



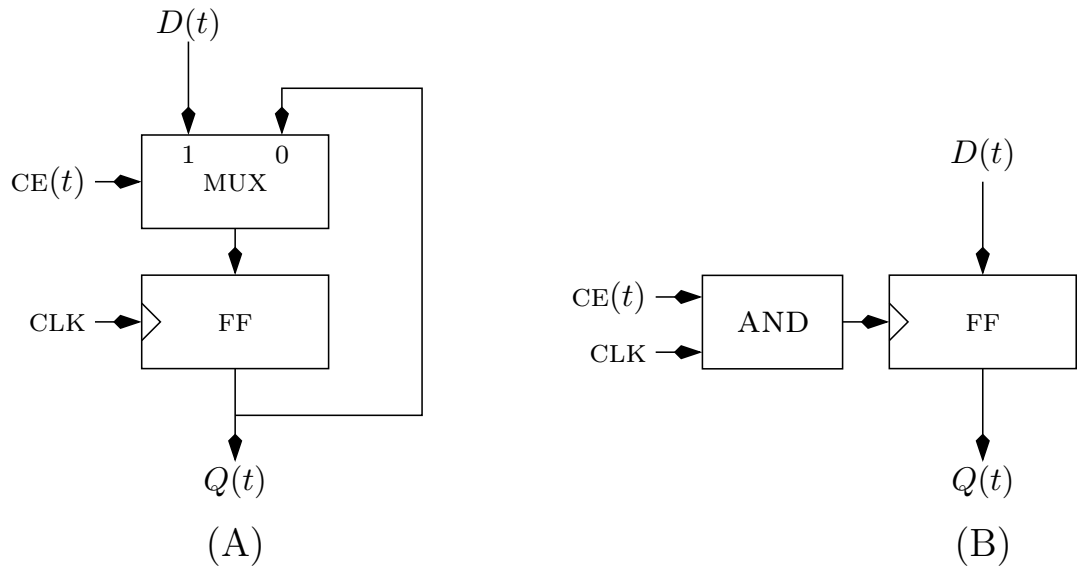
(A)



(B)

Question: Which design is correct?

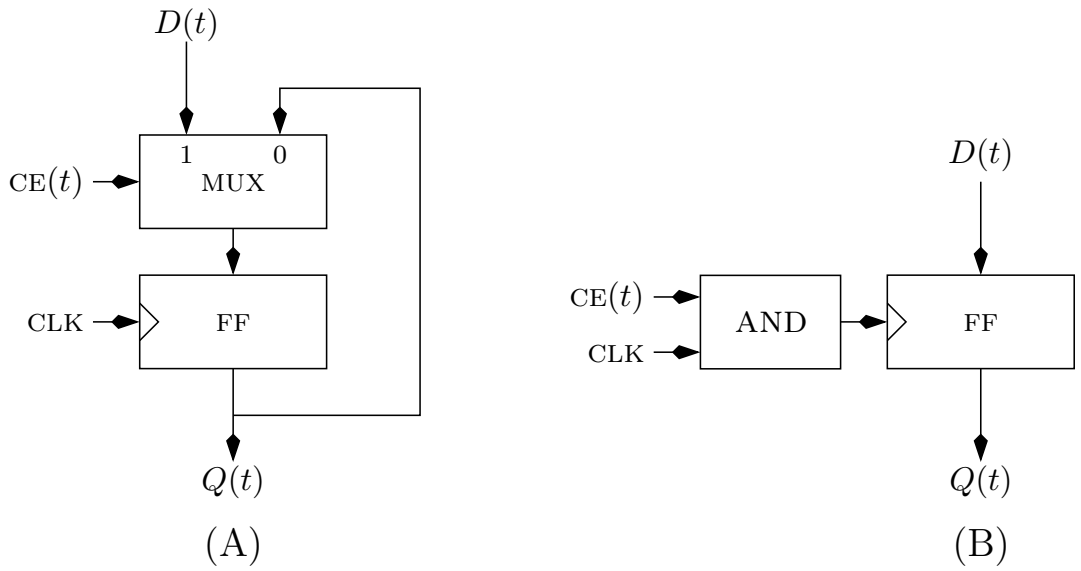
Clock enabled flip-flips : implementation - cont



Design (B) is wrong because:

- output of the AND-gate is not a clock signal (glitches).
- slow transitions of the output of the AND-gate (increase hold time)
- in some technologies, the flip-flop does not retain the stored bit forever. $\Rightarrow$ if CE$(t) = 0$ for a long period, then the flip-flop's output may become non-logical.

Clock enabled flip-flips : implementation - cont



Question: Compute the parameters of the clock-enabled flip-flop depicted in part (A) in terms of the parameters of the edge-triggered flip-flop and the MUX.

Summary

- clock signal - definition, terminology
- define edge-triggered flip-flops
- prove that critical segments are crucial:
  - arbitration - the problem of deciding “whose first”
  - prove that arbiters do not exist
  - use this proof to show that critical segments are crucial
- a timing example
- other memory devices: D-latch & clock-enabled flip-flop

Chapter 12: Synchronous Circuits

Computer Structure - Spring 2005

©Dr. Guy Even

Tel-Aviv Univ.

Preliminary Questions

- What is a synchronous circuit?
- How can we tell if the clock period is not too short? Is it possible to compute the minimum clock period?
- Is it possible to separate between the timing analysis and functionality in synchronous circuits?
- How can we initialize a synchronous circuit?

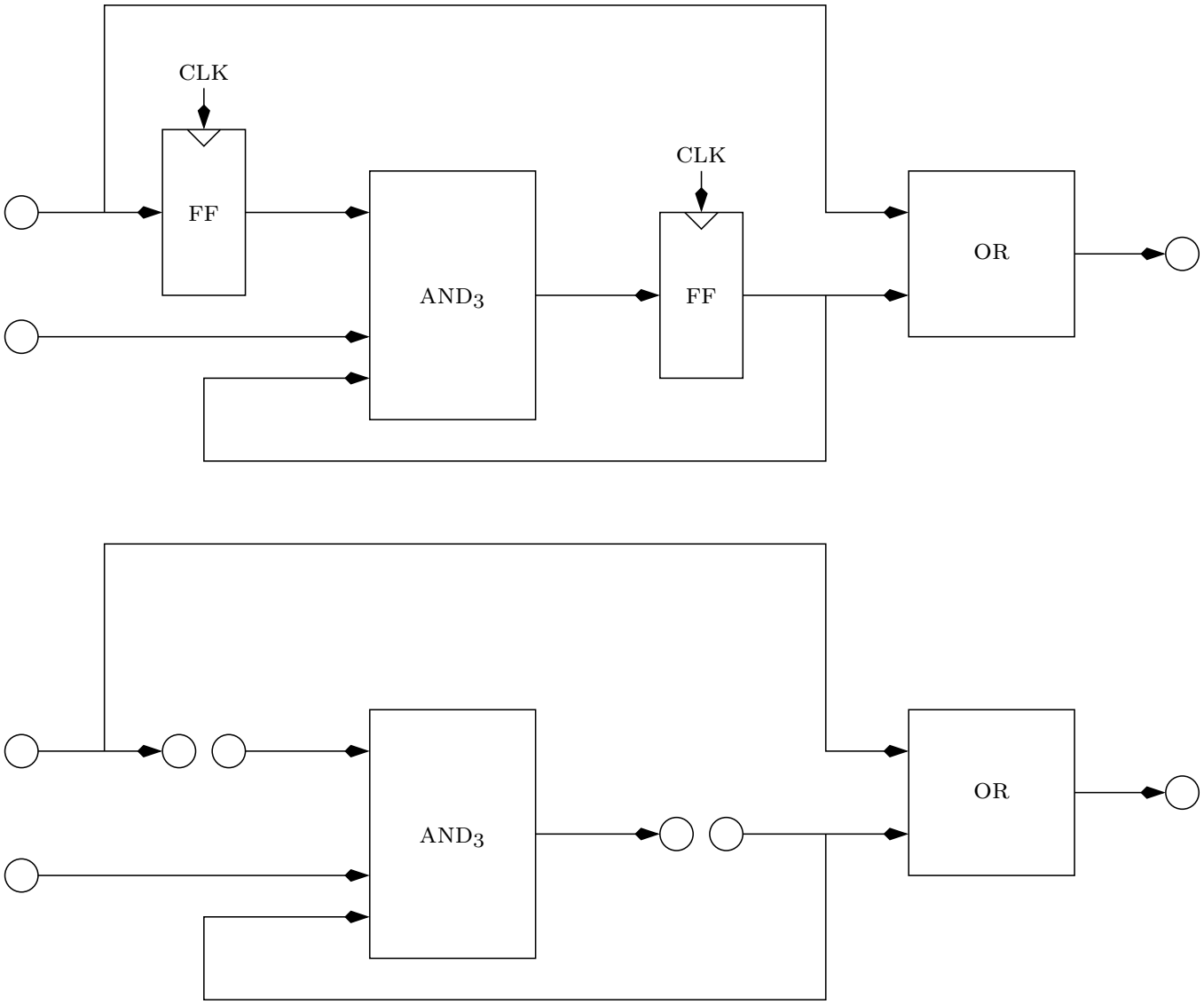
Goals

- define synchronous circuits.
- analyze timing (start with simple case...).
- define: timing constraints.
- find out if timing constraints are feasible.
- define: minimum clock period.
- algorithm: check if timing constraints are feasible.
- algorithm: compute minimum clock period.

Striping flip-flops away

- C - a circuit composed of combinational gates, nets, and flip-flops with a clock net called CLK.
- C' - a circuit obtained from C by:
  1. deleting the CLK net,
  2. deleting the input gate that feeds the CLK net, and
  3. replacing each flip-flip with an output gate (instead of the port D) and an input gate (instead of the port Q).

Striping flip-flops away - example



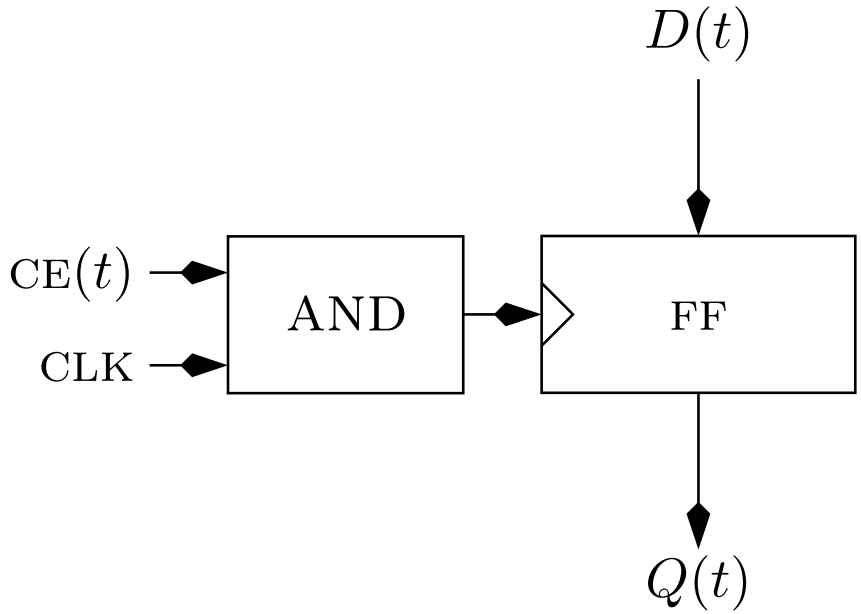
Definition: Synchronous Circuit

A synchronous circuit is a circuit C composed of combinational gates, nets, and flip-flops that satisfies the following conditions:

1. There is a net called CLK that carries a clock signal.
2. The CLK net is fed by an input gate.
3. The set of ports that are fed by the CLK net equals the set of clock-inputs of the flip-flops.
4. The circuit C' obtained from C by stripping away flip-flops is combinational.

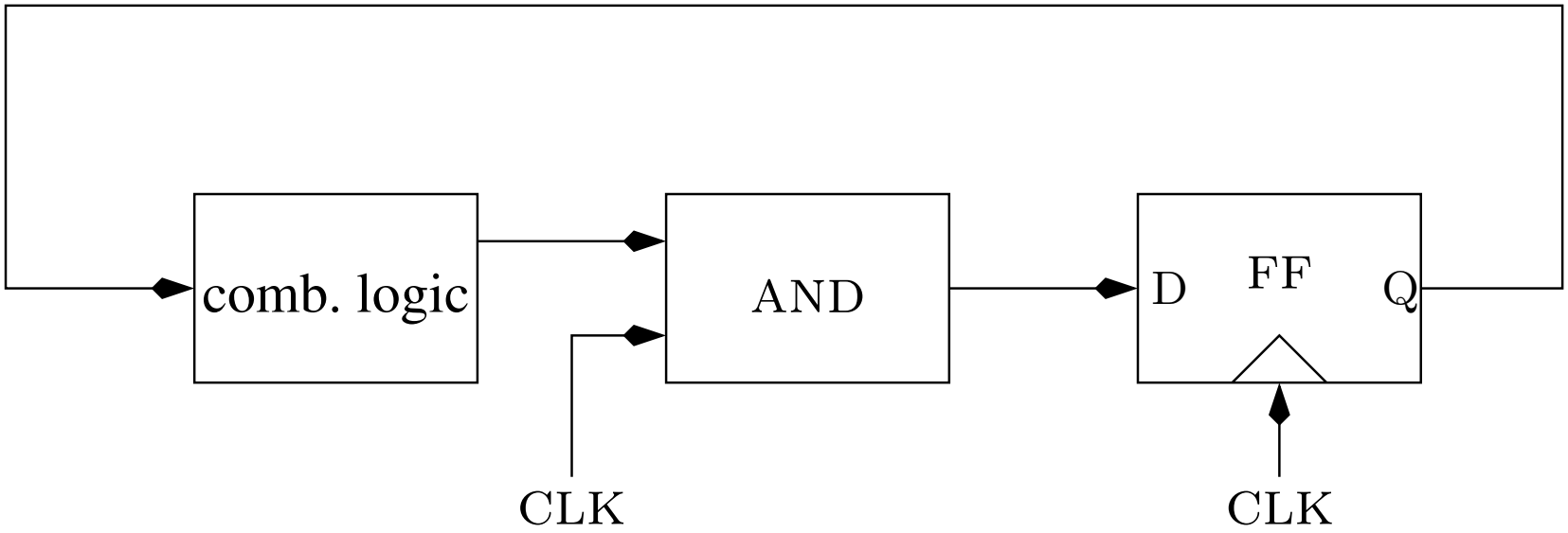
remarks on the definition of synchronous circuits

- CLK connected to all the clock-ports of flip-flops and only to them.
- We already saw that a “bad example” in which CLK feeds a gate:

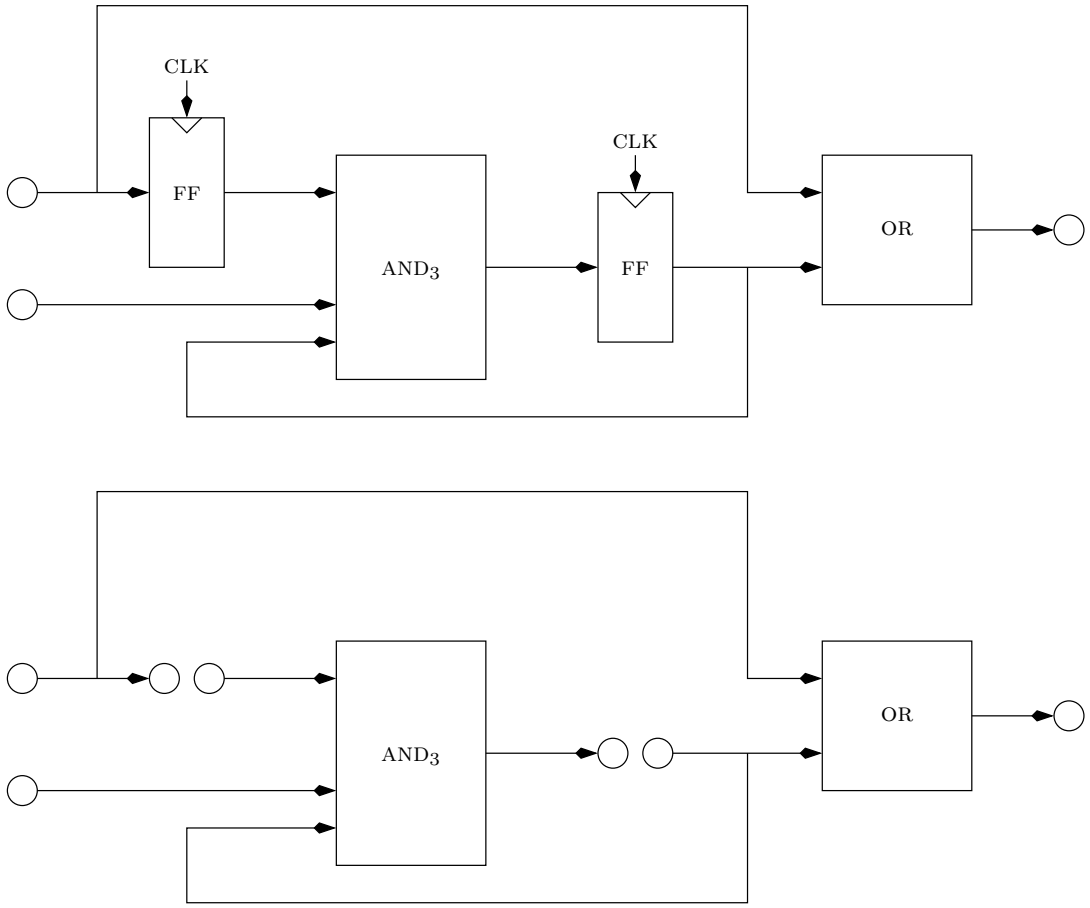


remarks on the definition of synchronous circuits

Question: What is required so that the D-port is stable during the critical segment in this "bad example":



back to the first example



Question: Is this a synchronous circuit?

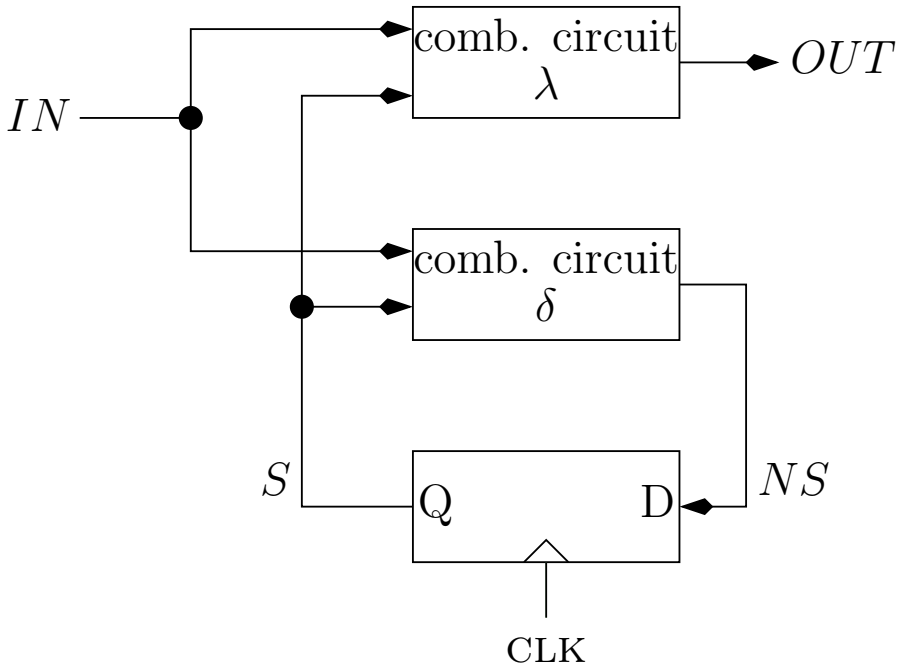
Recognizing a synchronous circuit

Question: Suggest an efficient algorithm that decides if a given circuit is synchronous.

Recall the definition:
A synchronous circuit is a circuit C composed of combinational gates, nets, and flip-flops that satisfies the following conditions:

1. There is a net called CLK that carries a clock signal.
2. The CLK net is fed by an input gate.
3. The set of ports that are fed by the CLK net equals the set of clock-inputs of the flip-flops.
4. The circuit C' obtained from C by stripping away flip-flops is combinational.

Synchronous Circuits: canonic form

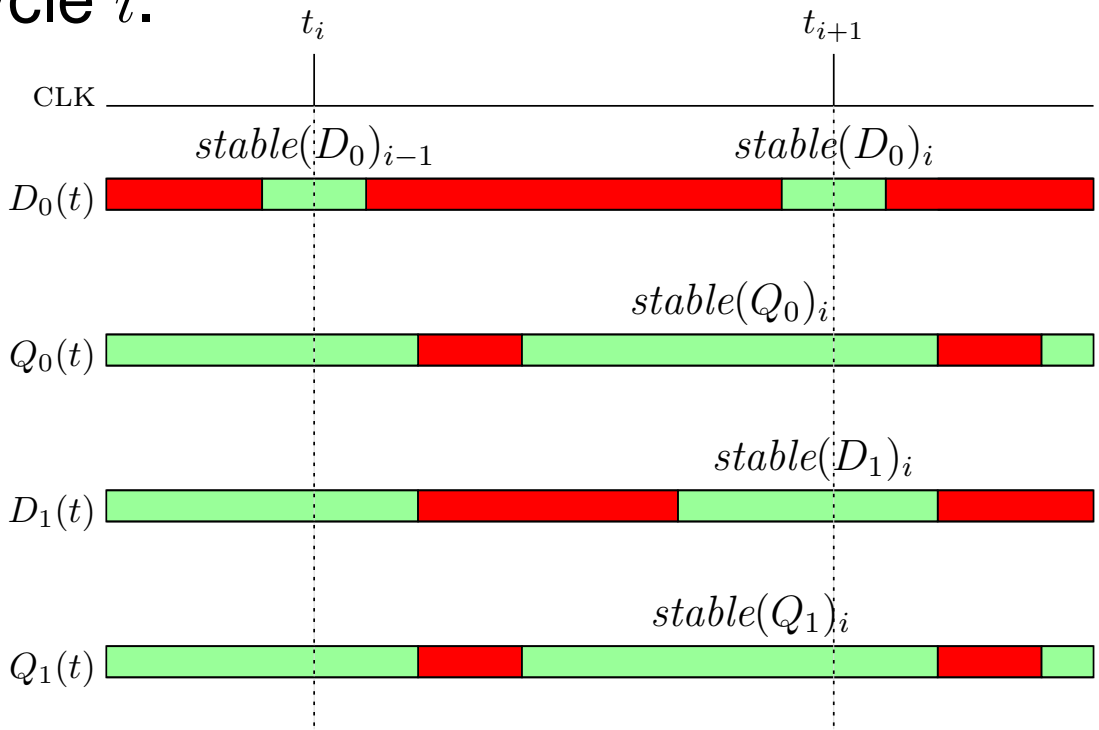


Transform a synchronous to canonic form:

- gather the flip-flops into one group.
- duplicate the combinational circuits to separate between output and next-state.

Stability Interval

- stability interval of signal X - interval during which X is stable.
- *stable*$(X)_i$ - stability interval of X corresponding to clock cycle i.



Timing analysis: the canonic form

Plan:

- Define timing constraints for IN and OUT.
- Define timing constraints for S and NS.
- Find sufficient conditions so that timing constraints are feasible.
- Define minimum clock period.
- Infer functionality from syntax.

Input/output timing constraints

- The input/output timing constraints formulate the timing interface between the the circuit and the “external world”.
- Input timing constraint - tells us when the input is guaranteed to be stable.
- Output timing constraint - tells us when the circuit’s output is required to be stable.
- Usually the external world is also a synchronous circuit. $\Rightarrow IN$ is an output of another synchronous circuit, and OUT is an input of another synchronous circuit.

Input timing constraint

The timing constraint corresponding to IN is defined by two parameters: *pd*(IN) > *cont*(IN) as follows.

$$\forall i: \quad [t_i + \textit{pd}(IN), t_{i+1} + \textit{cont}(IN)] \subseteq \textit{stable}(IN)_i.$$

Remarks:

- t_i - denotes the starting time of the ith clock period.
- Why do we require that *pd*(IN) > *cont*(IN)?
  If *pd*$(IN) \leq$ *cont*(IN), then the stability intervals *stable*$(IN)_i$ and *stable*$(IN)_{i+1}$ overlap. This means that IN is always stable, which is obviously not an interesting case.

Output timing constraint

The timing constraint corresponding to OUT is defined by two parameters: *setup*(OUT) and *hold*(OUT) as follows.

$$\forall i: \quad [t_{i+1} - \textit{setup}(OUT), t_{i+1} + \textit{hold}(OUT)] \subseteq \textit{stable}(OUT)_i.$$

Remark: Note that that timing constraint of OUT is given relative to the end of the ith cycle (i.e. t_{i+1}) .

Remarks

- Asymmetry in the terminology regarding IN and OUT. The parameters associated with IN are *pd*(IN) and *cont*(IN), whereas the parameters associated with OUT are *setup*(OUT) and *hold*(OUT).
- this is not very aesthetic if OUT is itself an input to another synchronous circuit.
- useful to regard IN as an output of a flip-flip and OUT as an input of a flip-flop (even if they are not).

Timing constraint of NS

NS is stable during the critical segments. Namely,

$$\forall i \geq 0: \quad C_{i+1} \subseteq \textit{stable}(NS)_i.$$

Remark: Note that, as in the case of the output signal, the timing constraint of NS corresponding to clock cycle i is relative to the end of the ith clock cycle (i.e. the critical segment C_{i+1}).

Remark: If NS satisfies its timing constraint for i , then S satisfies:

$$[t_{i+1} + t_{\textit{pd}}, t_{i+2} + t_{\textit{cont}}] \subseteq \textit{stable}(S)_{i+1}.$$

Stability Intervals of OUT & NS

- We associate a contamination delay *cont*(x) and a propagation delay *pd*(x) with each combinational circuit x.
- If $[t_i + t_{pd}, t_{i+1} + t_{cont}] \subseteq$ *stable*$(S)_i$, then the stability intervals of the signals OUT and NS satisfy:

$$[t_i + \max\{t_{pd}, \textit{pd}(IN)\} + pd(\lambda), t_{i+1} + \min\{t_{cont}, \textit{cont}(IN)\} + \textit{cont}(\lambda)] \subseteq \textit{stable}(OUT)_i$$

$$[t_i + \max\{t_{pd}, \textit{pd}(IN)\} + pd(\delta), t_{i+1} + \min\{t_{cont}, \textit{cont}(IN)\} + \textit{cont}(\delta)] \subseteq \textit{stable}(NS)_i.$$

Sufficient conditions: OUT

Claim: If

$$[t_i + t_{pd}, t_{i+1} + t_{cont}] \subseteq stable(S)_i$$

$$\max\{t_{pd}, pd(IN)\} + pd(\lambda) + setup(OUT) \leq t_{i+1} - t_i$$

$$\min\{t_{cont}, cont(IN)\} + cont(\lambda) \geq hold(OUT),$$

then

$$[t_{i+1} - setup(OUT), t_{i+1} + hold(OUT)] \subseteq stable(OUT)_i.$$

Proof: stability interval of OUT satisfies:

$$[t_i + \max\{t_{pd}, pd(IN)\} + pd(\lambda), t_{i+1} + \min\{t_{cont}, cont(IN)\} + cont(\lambda)]$$
$$\subseteq stable(OUT)_i$$

□

Sufficient conditions: NS

Claim: If

$$[t_i + t_{pd}, t_{i+1} + t_{cont}] \subseteq stable(S)_i$$

$$\max\{t_{pd}, pd(IN)\} + pd(\delta) + t_{su} \leq t_{i+1} - t_i$$

$$t_{hold} \leq \min\{t_{cont}, cont(IN)\} + cont(\delta),$$

then the signal NS is stable during the critical segment C_{i+1}.

Proof: stability interval of NS satisfies:

$$[t_i + \max\{t_{pd}, pd(IN)\} + pd(\delta), t_{i+1} + \min\{t_{cont}, cont(IN)\} + cont(\delta)]$$
$$\subseteq stable(NS)_i.$$

□

Timing constraints for $i \geq 0$

CORO: If 4 conditions hold and

$$[t_0 + t_{pd}, t_1 + t_{cont}] \subseteq \mathit{stable}(S)_0,$$

then

1. timing constraints of NS and OUT hold wrt every $i \geq 0$,
2. $\forall i \geq 0 \ : \ [t_i + t_{pd}, t_{i+1} + t_{cont}] \subseteq \mathit{stable}(S)_i$.

Proof: Induction on i.

- Basis: part (1) follows from sufficient conditions for OUT and NS.
- Step: NS is stable during $C_{i+1} \Rightarrow$ part (2).
- $\Rightarrow$ part(1).

□

Simplifying the conditions

- Our goal is to simplify the conditions in the 2 Claims.
- Prefer: lower bounds on the clock period.
- $\Rightarrow$ well defined functionality provided that the clock period is large enough.
- We discuss each of the 4 conditions (2 per claim).

$$\max\{t_{\boldsymbol{pd}}, \boldsymbol{pd}(IN)\} + pd(\lambda) + \boldsymbol{setup}(OUT) \leq t_{i+1} - t_i$$

- condition is a lower bound on $\varphi(\mathsf{CLK})$. Great.

$$\min\{t_{cont}, \mathbf{cont}(IN)\} + \mathbf{cont}(\lambda) \geq \mathbf{hold}(OUT)$$

- condition may not hold $\Rightarrow$ serious problem that can lead to failure to meet the timing constraint of OUT...
- Hope: under reasonable circumstances, condition does hold. Why?
  - Suppose IN is the output of a combinational circuit, all the inputs of which are outputs of flip-flops.
  - Assume that all the flip-flops are identical.
  - It follows that $\mathit{cont}(IN) \geq t_{\mathit{cont}}$.
  - By definition: $\mathit{cont}(\lambda) \geq 0$.
  - $\Rightarrow \min\{t_{\mathit{cont}}, \mathit{cont}(IN)\} + \mathit{cont}(\lambda) \geq t_{\mathit{cont}}$.
  - Suppose OUT feeds a combinational circuit that feeds a flip-flop.
  - Hence $\mathit{hold}(OUT) \leq t_{\mathit{hold}}$.
  - $t_{\mathit{hold}} < t_{\mathit{cont}} \Rightarrow$ condition holds.

$$\max\{t_{\boldsymbol{pd}}, \boldsymbol{pd}(IN)\} + \boldsymbol{pd}(\delta) + t_{\boldsymbol{su}} \leq t_{i+1} - t_i$$

- condition is a lower bound on $\varphi(\mathsf{CLK})$. Great.

$$t_{\boldsymbol{hold}} \leq \min\{t_{\boldsymbol{cont}}, \boldsymbol{cont}(IN)\} + \boldsymbol{cont}(\delta)$$

- As before, if $\mathit{cont}(IN) \geq t_{\mathit{cont}}$, the condition holds!

Conclusion

Claim: Assume that $\mathit{cont}(IN) \geq t_{\mathit{cont}}$ and $\mathit{hold}(OUT) \leq t_{\mathit{hold}}$. If

$$[t_0 + t_{\mathit{pd}}, t_1 + t_{\mathit{cont}}] \subseteq \mathit{stable}(S)_0,$$

$$\varphi(\mathsf{CLK}) \geq \max\{t_{\mathit{pd}}, \mathit{pd}(IN)\}$$

$$+ \max\{\mathit{pd}(\lambda) + \mathit{setup}(OUT), \mathit{pd}(\delta) + t_{\mathit{su}}\},$$

then

1. timing constraints of NS and OUT hold wrt every $i \geq 0$,
2. $\forall i \geq 0 \ : \ [t_i + t_{\mathit{pd}}, t_{i+1} + t_{\mathit{cont}}] \subseteq \mathit{stable}(S)_i$.

Under reasonable assumptions, all we need is initialization and a sufficiently long clock period.

Minimum clock period

DEF: The minimum clock period of a synchronous circuit C is the shortest clock period for which the timing constraints of the output signals and signals that feed the flip-flops are satisfied.
We denote the minimum clock period of a synchronous circuit by $\varphi^*(C)$.

- Minimum clock period does not exist if timing constraints are infeasible.
- "timing constraints are satisfied" - for every value of the delays provided that they are in their range. (i.e. actual propagation delay of λ is in $[0, \mathit{pd}(\lambda)]$.)
- if assumptions hold, then in canonic form

$$\varphi(\mathsf{CLK}) \geq \max\{t_{\mathit{pd}}, \mathit{pd}(IN)\} + \max\{\mathit{pd}(\lambda) + \mathit{setup}(OUT), \mathit{pd}(\delta) + t_{\mathit{su}}\}.$$

Discussion

- The timing analysis of synchronous circuits in canonic form is overly pessimistic.
- The problem is that each of the combinational circuits λ and δ is regarded as a "gate" with a propagation delay.
- In practice it may be the case, for example, that the accumulated delay from the input IN to the output OUT is significantly different than the accumulated delay from S to the output OUT. The situation is even somewhat more complicated in the case of multi-bit signals.

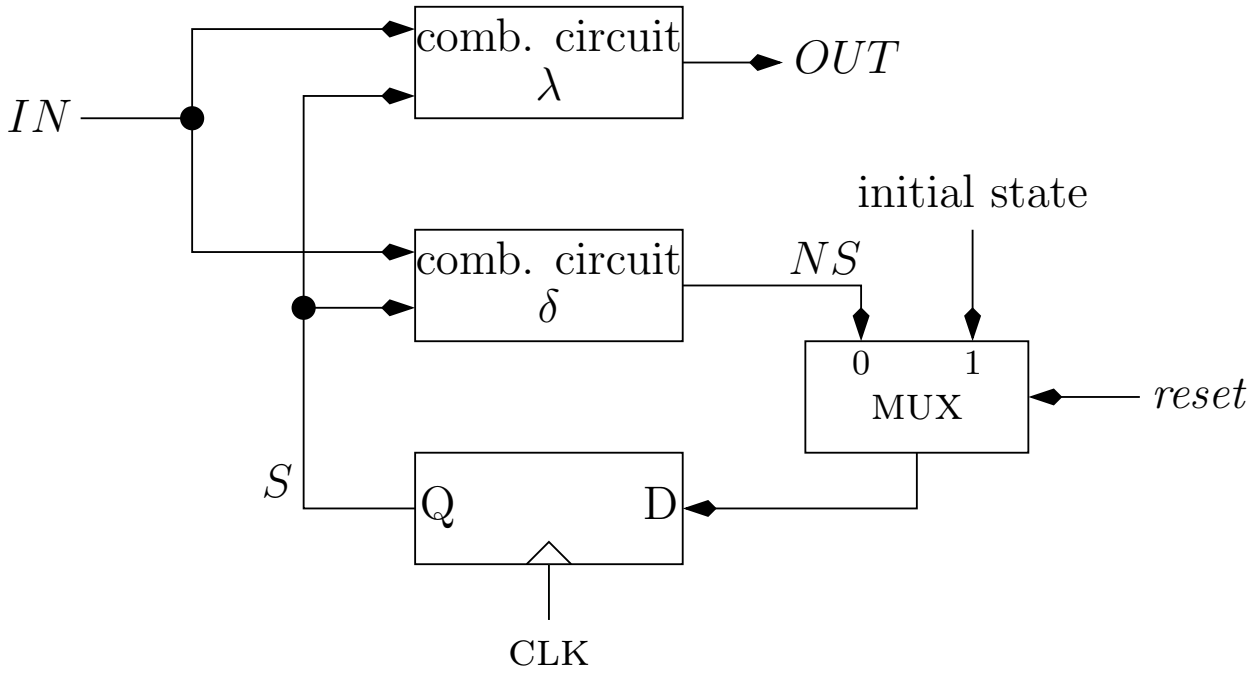
Initialization

We require that

$$[t_0 + t_{pd}, t_1 + t_{cont}] \subseteq \mathit{stable}(S)_0.$$

- after power-up, flip-flop output may be non-logical (and even meta-stable).
- solution: introduce a reset signal.
- boot-strapping problem: How is a reset signal generated?
- no solution to this problem within the digital abstraction (meta-stability). All we can try to do is reduce the probability of such an event.
- reset controller - a special circuit that generates a reset signal.

Synchronous Circuit: canonic form with reset



Remark: NS may not be logical during reset.
Implementation of MUX must output initial-state if *reset* $= 1$.
Implementation based on drivers has this property, while implementation based on combinational gates may not have this property.

Functionality of Synchronous Circuits: canonic form

- X_i - *dig*(X) during *stable*$(X)_i$.
- Assumptions:

$$\begin{aligned}
\textit{cont}(IN) &\geq t_{\textit{cont}} \\
\textit{hold}(OUT) &\leq t_{\textit{hold}} \\
[t_0 + t_{\textit{pd}}, t_1 + t_{\textit{cont}}] &\subseteq \textit{stable}(S)_0, \\
\varphi(\textsc{clk}) &\geq \max\{t_{\textit{pd}}, \textit{pd}(IN)\} \\
&\quad + \max\{\textit{pd}(\lambda) + \textit{setup}(OUT), \textit{pd}(\delta) + t_{\textit{su}}\}.
\end{aligned}$$

CORO: Assumptions $\Rightarrow$ $\forall i \geq 0$:

$$\begin{aligned}
NS_i &= \delta(IN_i, S_i) \\
OUT_i &= \lambda(IN_i, S_i) \\
S_{i+1} &= NS_i.
\end{aligned}$$

Finite State Machines

Corollary states that synchronous circuits implement finite state machines.

DEF: A finite state machine (FSM) is a 6-tuple $\mathcal{A} = \langle Q, \Sigma, \Delta, \delta, \lambda, q_0 \rangle$, where

- Q is a set of states.
- Σ is the alphabet of the input.
- Δ is the alphabet of the output.
- $\delta : Q \times \Sigma \rightarrow Q$ is a transition function.
- $\lambda : Q \times \Sigma \rightarrow Q$ is an output function.
- $q_0 \in Q$ is an initial state.

Definition of FSM: remarks

- Other terms for a finite state machine are a finite automaton with outputs, transducer, and Mealy Machine.
- Moore Machine - an FSM in which the output function $\lambda : Q \rightarrow \Delta$.

What does an FSM do?

- abstract machine that operates as follows.
- input sequence $\{x_i\}_{i=0}^{n-1}$ of symbols over alphabet Σ.
- output sequence $\{y_i\}_{i=0}^{n-1}$ of symbols over alphabet Δ.
- sequence of states $\{q_i\}_{i=0}^{n}$. The state q_i is defined recursively:

$$q_{i+1} \stackrel{\triangle}{=} \delta(q_i, x_i)$$

- The output y_i is defined as follows:

$$y_i \stackrel{\triangle}{=} \lambda(q_i, x_i).$$

State Diagrams

FSMs are often depicted using state diagrams.

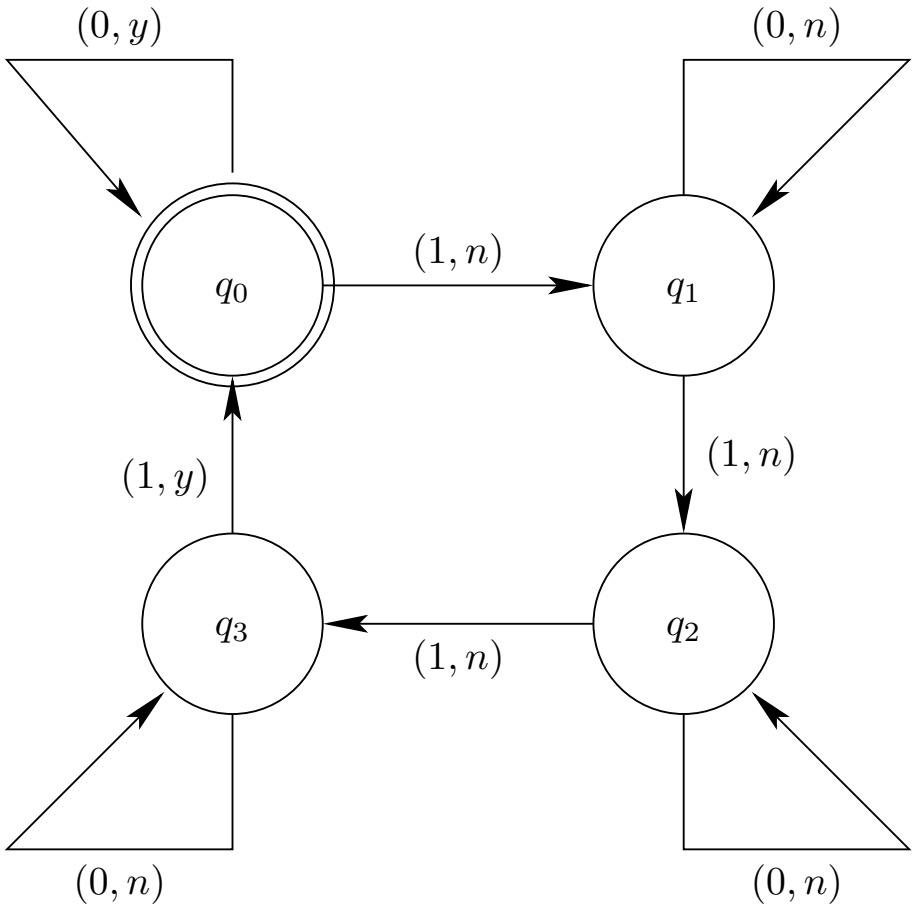
DEF: The state diagram corresponding to an FSM $\mathcal{A}$ is a directed graph $G = (V, E)$ with edge labels $(x, y) \in \Sigma \times \Delta$. The vertex set V equals the state set S. The edge set E is defined by

$$E \stackrel{\triangle}{=} \{(q, \delta(q, x)) : q \in Q \text{ and } x \in \Sigma\}.$$

An edge $(q, \delta(q, x))$ is labeled $(x, \lambda(q, x))$.

State Diagram: example

A state diagram of an FSM that outputs y if the weight of the input so far is divisible by 4, and n otherwise.



Timing analysis: the general case

- Deal with a synchronous circuit that is not in canonic form.
- Algorithm that computes the minimum clock period $\varphi^*(C)$. (if timing constraints are feasible.)
- Algorithm that decides whether the timing constraints are feasible (i.e. conditions used by this algorithm are less restrictive than the conditions used in previous claims).

Review of timing constraints

Input constraints: For every input signal IN, guaranteed:

$$[t_i + \mathit{pd}(IN), t_{i+1} + \mathit{cont}(IN)] \subseteq \mathit{stable}(IN)_i.$$

Output constraints: For every output signal OUT, require:

$$[t_{i+1} - \mathit{setup}(OUT), t_{i+1} + \mathit{hold}(OUT)] \subseteq \mathit{stable}(OUT)_i.$$

Critical segments: For every signal NS that feeds a D-port of a flip-flop, require:

$$C_{i+1} \subseteq \mathit{stable}(NS)_i.$$

Algorithm: minimum clock period

- $C' \leftarrow$ combinational circuit obtained by stripping away flip-flops from C.
- For every gate v of C' define $d(v)$ as follows:

$$d(v) \triangleq \begin{cases} \textit{pd}(IN) & \text{if } v \text{ feeds input signal } IN. \\ t_{\textit{pd}} & \text{if } v \text{ corresponds to a } Q\text{-port.} \\ \textit{setup}(OUT) & \text{if } v \text{ is fed by } OUT. \\ t_{\textit{su}} & \text{if } v \text{ corresponds to a } D\text{-port.} \\ \textit{pd}(v) & \text{if } v \text{ is a combinational gate of } C. \end{cases}$$

- Let $DG(C')$ denote the directed acyclic graph (DAG) that corresponds to C'. Let p' denote the longest path in $DG(C')$ with respect to the delays $d(v)$. Return $d(p')$.

Algorithm: correctness

define delays $c(v)$ to non-sink vertices in $DG(C')$ as follows.

$$c(v) \triangleq \begin{cases} \mathit{cont}(IN) & \text{if } v \text{ feeds an input signal } IN. \\ t_{\mathit{cont}} & \text{if } v \text{ corresponds to a } Q\text{-port of a flip-flop.} \\ \mathit{cont}(v) & \text{if } v \text{ is a combinational gate in } C. \end{cases}$$

Lemma: Consider a combinational gate, an input gate, or a flip-flop v in the synchronous circuit C. Let $\mathcal{P}_v$ denote the set of all directed paths in the directed acyclic graph $DG(C')$ that begin at a source and end in v. If the output of every flip-flop is stable in the interval $[t_i + t_{\mathit{pd}}, t_{i+1} + t_{\mathit{cont}}]$, then every output N of v satisfies

$$[t_i + \max_{p \in \mathcal{P}_v} d(p), t_{i+1} + \min_{p \in \mathcal{P}_v} c(p)] \subseteq \mathit{stable}(N)_i.$$

proof: $[t_i + \max_{p \in \mathcal{P}_v} d(p), t_{i+1} + \min_{p \in \mathcal{P}_v} c(p)] \subseteq \textbf{\textit{stable}}(N)_i$.

- $\{v_0, \ldots, v_{n-1}\}$ - topological sort of vertices of $DG(C')$.
- Let $v = v_j$. Proof by induction on j.
- Basis: two cases: (i) If v is an input gate, then input constraint. (ii) If v is a flip-flop, then assumption on the output of flip-flops.
- Step: same as basis if v is an input gate or a flip flop.
- Assume v is a combinational gate.
- Ind. Hyp. : every input N' of v_{j+1} satisfies equation.
- $\Rightarrow$ every output N of v_{j+1} satisfies: (i) N becomes stable at most $d(v_{j+1})$ time units after its last input becomes stable, and (ii) N remains stable at least $c(v_{j+1})$ time units after its first input becomes instable.

Algorithm: correctness (cont.)

Claim: Suppose that: (i) for every signal fed by a Q-port of a flip-flop, $[t_i + t_{pd}, t_{i+1} + t_{cont}] \subseteq \mathit{stable}(S)_i$, (ii) for every input IN, $\mathit{cont}(IN) \geq t_{cont}$, and (iii) for every output OUT, $\mathit{hold}(OUT) \leq t_{hold}$. Then,

1. For every clock period $\varphi(\mathrm{CLK}) \geq \varphi^*(\mathrm{CLK})$, the signals feeding D-ports of flip-flops are are stable during the critical segment C_{i+1}.
2. For every clock period $\varphi(\mathrm{CLK}) \geq \varphi^*(\mathrm{CLK})$, the output timing constraints corresponding to cycle i are satisfied.
3. For every clock period $\varphi(\mathrm{CLK}) < \varphi^*(\mathrm{CLK})$, a violation of the timing constraints is possible.

Proof: $\varphi(\mathsf{CLK}) \geq \varphi^*(\mathsf{CLK})$ satisfies timing constraints

Let N denote signal that feeds a D-port of a flip-flop v that is fed by u. By Lemma, N is stable during the interval

$$[t_i + \max_{p \in \mathcal{P}_u} d(p), t_{i+1} + \min_{p \in \mathcal{P}_u} c(p)].$$

Since $\varphi(\mathsf{CLK}) \geq \max_{p \in \mathcal{P}_v} d(p) = d(v) + \max_{p \in \mathcal{P}_u} d(p)$ and $d(v) = t_{su}$, we conclude that

$$t_{i+1} - t_i = \varphi(\mathsf{CLK}) \geq t_{su} + \max_{p \in \mathcal{P}_u} d(p).$$

$\Rightarrow$ signal N stable starting at

$$t_i + \max_{p \in \mathcal{P}_u} d(p) \leq t_{i+1} - t_{su}.$$

$\Rightarrow$ setup-time constraint is satisfied.

Proof: $\varphi(\mathsf{CLK}) \geq \varphi^*(\mathsf{CLK})$ - cont.

N is stable until $t_{i+1} + \min_{p \in \mathcal{P}_u} c(p)$.
However, every path $p \in \mathcal{P}_u$ begins at a source. A source may correspond to an input gate in C or a Q-port of a flip flop. Since *cont*$(IN) \geq t_{cont}$, we conclude that $c(s) \geq t_{cont}$, for every source s.
It follows that

$$\min_{p \in \mathcal{P}_u} c(p) \geq t_{cont} > t_{hold}.$$

Lemma: N is stable until $t_{i+1} + \min_{p \in \mathcal{P}_u} c(p) \geq t_{i+1} + t_{hold}$.
$\Rightarrow$ hold-time constraint is satisfied.

$\Rightarrow$ N is stable during the critical segment C_{i+1}, as required.

Proof: $\varphi(\mathsf{CLK}) \geq \varphi^*(\mathsf{CLK})$ **- cont.**

Proof for an output signal OUT is similar.

Proof: $\varphi(\mathsf{CLK}) < \varphi^*(\mathsf{CLK}) \Rightarrow$ violation

- p - longest path in $DG(C')$ with respect to lengths $d(v)$. (p begins at a source and ends in a sink v.)
- Let p' denote the path obtained from p by omitting the sink v. It follows that

$$t_i + d(p') > t_{i+1} - d(v).$$

- If the actual propagation delays along p are maximal, then the signal feeding v is not stable at time $t_{i+1} - d(v)$.
- If v is a flip-flop, then its input is not stable during the critical segment.
- If v is an output gate, then its input does not meet the output constraint. The claim follows.

Corollary

If the circuit is properly initialized, then the clock period computed by the algorithm is the shortest clock period that satisfies all the timing constraints for all clock cycles i, for $i \geq 0$.
Formally...

Corollary - formal

CORO: Suppose that: (i) for every signal S fed by a Q-port of a flip-flop, $[t_0 + t_{pd}, t_1 + t_{cont}] \subseteq \mathit{stable}(S)_0$, (ii) for every input IN, $\mathit{cont}(IN) \geq t_{cont}$, and (iii) for every output OUT, $\mathit{hold}(OUT) \leq t_{hold}$. Then,

1. For every clock period $\varphi(\mathsf{CLK}) \geq \varphi^*(\mathsf{CLK})$, the signals feeding D-ports of flip-flops are are stable during every critical segment C_{i+1}, for $i \geq 0$.
2. For every clock period $\varphi(\mathsf{CLK}) \geq \varphi^*(\mathsf{CLK})$, the output timing constraints corresponding to cycle i are satisfied, for every $i \geq 0$.
3. For every clock period $\varphi(\mathsf{CLK}) < \varphi^*(\mathsf{CLK})$, a violation of the timing constraints is possible.

Proof: Proof is by induction on the clock cycle i. □

Algorithm: feasibility of timing constraints

- So far, reasonable assumptions are made so that it is guaranteed that a minimum clock period exists.
- It is possible that these assumptions do not hold although the timing constraints are feasible.
- We now present an algorithm that verifies whether the timing constraints are feasible without relying an any assumptions.

Follow recipe of the lemma

Lemma states that, for every non-sink v in C', the guaranteed stability interval of the signals that are output by v is:

$$[t_i + \max_{p \in \mathcal{P}_v} d(p), t_{i+1} + \min_{p \in \mathcal{P}_v} c(p)].$$

The $\varphi^*(C)$ algorithm deals with making sure that each such interval does not start too late (i.e. satisfy setup-time constraint).

Feasibility means checking that stability intervals do not end too early (i.e. satisfy hold-time constraint).

Follow recipe of the lemma - cont

Recall: signal fed by v is stable during

$$[t_i + \max_{p \in \mathcal{P}_v} d(p), t_{i+1} + \min_{p \in \mathcal{P}_v} c(p)].$$

Check that

1. For every u that feeds a D-port of a flip-flop, require

$$\min_{p \in \mathcal{P}_u} c(p) \geq t_{\mathit{hold}}.$$

2. For every u that feeds an output signal OUT, require

$$\min_{p \in \mathcal{P}_u} c(p) \geq \mathit{hold}(OUT).$$

violation $\Rightarrow$ timing constraints are infeasible.
no violation $\Rightarrow$ timing constraints are feasible.

Algorithmic Aspect

- All we need to check if timing constraints are feasible is to compute

$$\forall \text{ non-sink } v \ : \ \min_{p \in \mathcal{P}_v} c(p).$$

- Compute shortest path in a DAG (can be done in linear time using depth first search).
- After these values are computed for all the non-sinks, the algorithm simply checks hold-time constraints for every D-port and for every output.
- If a violation is found, then the timing constraints are infeasible.

Recap

- We started with a syntactic definition of a synchronous circuit.
- We then attached timing constraints to the inputs and outputs of synchronous circuit.
- For a given synchronous circuit C with input/output timing constraints, we differentiate between two cases:
  - timing constraints are infeasible $\Rightarrow$ cannot guarantee well defined functionality of C. For example, if the timing constraints are not met, then inputs of flip-flops might not be stable during the critical segments, and then the flip-flop output is not guaranteed to be even logical.
  - timing constraints are feasible $\Rightarrow$ functionality is well defined provided that the clock period satisfies $\varphi(\mathsf{CLK}) \geq \varphi^*(\mathsf{CLK})$.

Functionality

- Assume that the timing constraints are feasible.
- Introduce a trivial timing model called the zero delay model.
- In this model, time is discrete and in each clock cycle, the circuit is reduced to a combinational circuit.
- Advantage: decouple timing issues from functionality and enables simple logical simulations.

The zero delay model

- In the zero delay model we assume that all the parameters of all the components are zero or infinitesimal (i.e. $\forall \varepsilon > 0$: $t_{su} = -\varepsilon$, $t_{hold} = t_{cont} = t_{pd} = \varepsilon$, $pd(IN) = cont(IN) = hold(OUT) = \varepsilon$, $setup(OUT) = -\varepsilon$ and $d(G) = t_{cont}(G) = 0$, for every combinational gate G). Under this unrealistic assumption, the timing constraints are feasible.
- Must pay attention to endpoints of intervals of stability: Output of flip-flip satisfies:

$$(t_i + t_{pd}, t_{i+1} + t_{cont}) \subseteq stable(Q)_i.$$

Hence,

$$(t_i, t_{i+1}] \subseteq stable(Q)_i.$$

The zero delay model - cont

- Similarly,

$$(t_i + \mathit{pd}(IN), t_{i+1} + \mathit{cont}(IN)) \subseteq \mathit{stable}(IN)_i.$$

  Hence,

$$(t_i, t_{i+1}] \subseteq \mathit{stable}(IN)_i.$$

- Following Corollary (synchronous circuit implements an FSM), we conclude that, for every signal X, X_i is well defined.

Simulation of a synchronous circuit

Simulation during cycles $i = 0, \ldots, n-1$ in the zero propagation model proceeds as follows:

assume: flip-flops are initialized ($\vec{S}_0$ - initial values of FFs).

1. Construct comb. circuit C' that corresponds to C.
2. For $i = 0$ to $n-1$ do:
   (a) Simulate C' with input values $\vec{S}_i$ and $I\vec{N}_i$.
   (b) For every output OUT^j, let y denote the value that is fed to y. We set $OUT_i^j = y$.
   (c) For every D-port NS^j of a flip-flop, let y denote the value that is fed to the flip-flop. We set $NS_i^j = y$.
   (d) For every Q-port S^j of a flip-flop, define $S_{i+1}^j \leftarrow NS_i^j$, where NS^j denotes the signal that feeds the D-port of the flip-flop.

Summary

- define synchronous circuits.
- canonic form of synchronous circuits:
  - definition of timing constraints.
  - formulation of sufficient conditions for satisfying the timing constraints.
  - simplify sufficient conditions by relying on the assumption that the input originates from a flip-flop and the output is eventually fed to a flip-flop.
  - define the minimum clock period.
  - initialization.
  - synchronous circuits implement FSMs.

Summary -cont.

- general case of synchronous circuits (not in canonic form).
  - algorithm: min. clock period.
  - algorithm: feasibility of timing constraints.
- functionality:
  - zero delay model.
  - simulation.