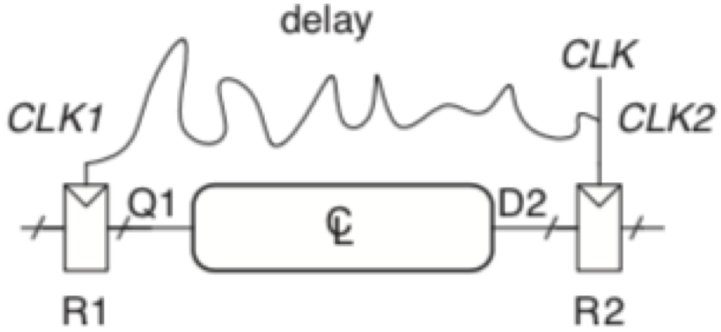
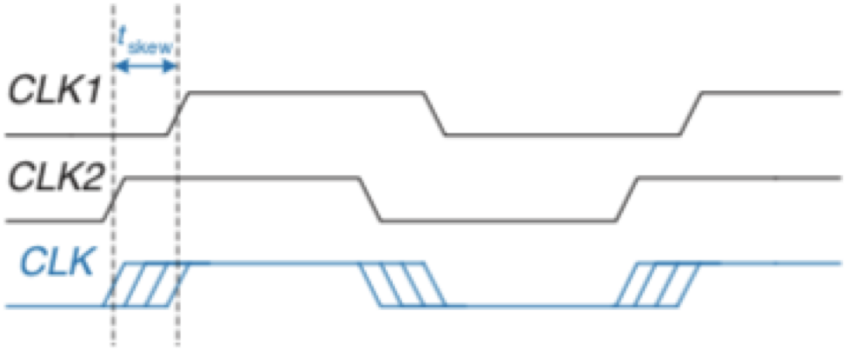
Sequential Logic Design

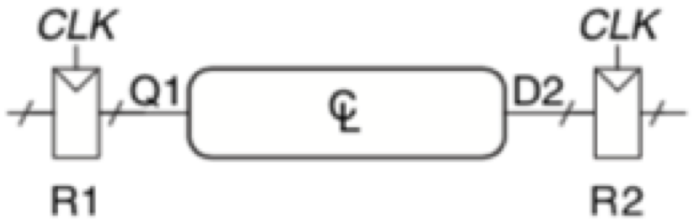
Lecture 8

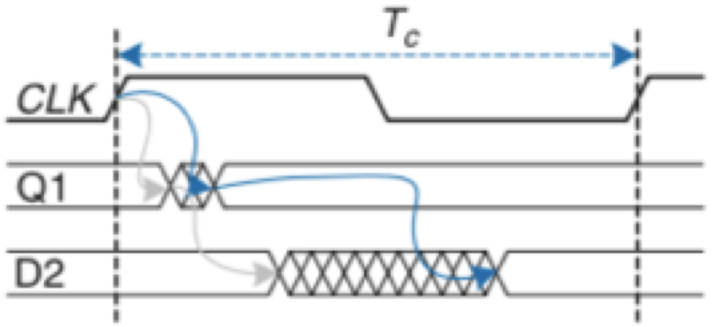
Clock Skew

- So far, we assumed that the clock reaches all registers at exactly the same time
- In reality, there is some variation in this time called clock skew.
  - the wires to different registers may be of different lengths
  - Noise also results in different delays.
  - Clock gating further delays the clock

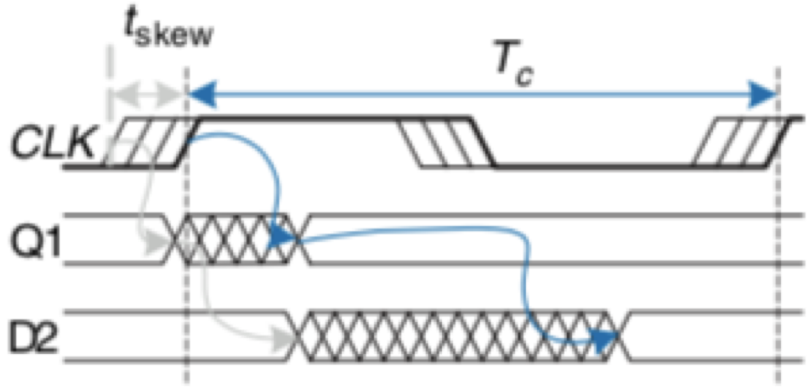






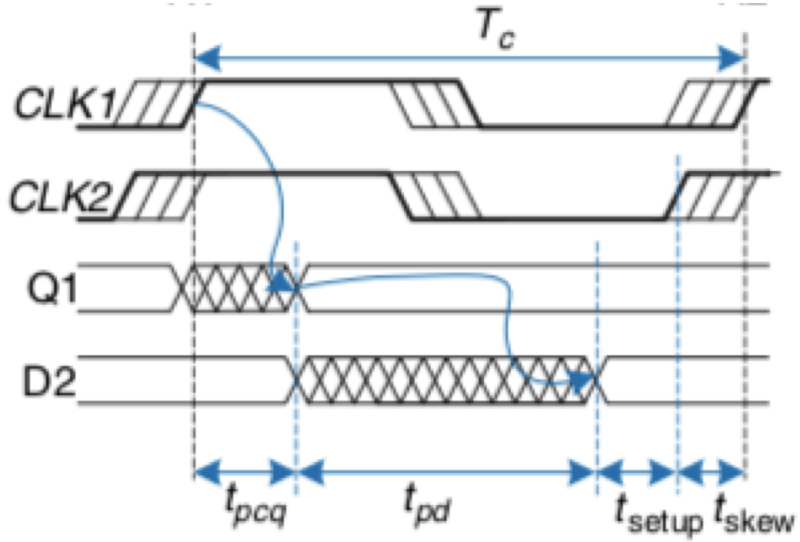


- We consider the worst-case scenario, so that we can guarantee that the circuit will work under all circumstances.
- In the worst case: R1 receives the latest skewed clock and R2 receives the earliest skewed clock, leaving as little time as possible for data to propagate between the registers.
- The data must setup before R2 samples it



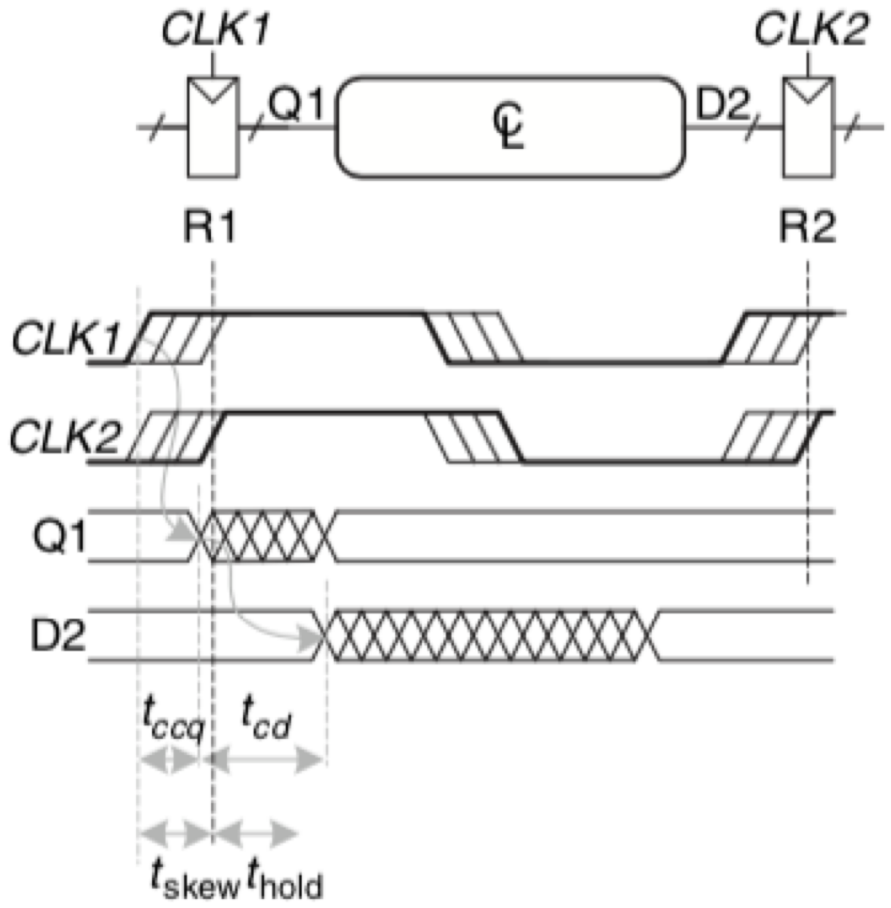
$$T_c \geq t_{pcq} + t_{pd} + t_{\text{setup}} + t_{\text{skew}}$$

$$t_{pd} \leq T_c - (t_{pcq} + t_{\text{setup}} + t_{\text{skew}})$$



It adds to the sequencing overhead and it increases the required minimum delay through the combinational logic.

Clock skew effectively increases both the setup time and the hold time



Example

Revisit Example 3.10 and assume that the system has 50 ps of clock skew.

- The critical path remains the same, but the setup time is effectively increased by the skew. Hence, the minimum cycle time is

$$T_c \geq t_{pcq} + 3t_{pd} + t_{\text{setup}} + t_{\text{skew}}$$
$$= 80 + 3\times 40 + 50 + 50 = 300\text{ps}$$

The maximum clock frequency is $f_c = 1/T_c = 3.33\ \text{GHz}$

The short path also remains the same at 55 ps.

The hold time is effectively increased by the skew to 60+50=110 ps >> 55 ps

the circuit will violate the hold time and malfunction at any frequency

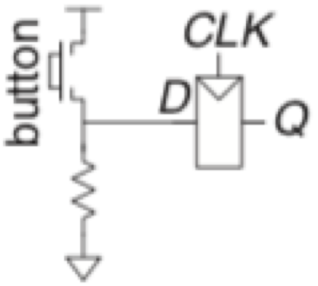
Fixing the Hold Time Violations

- The critical path is unaffected, so the maximum clock frequency remains 3.33 GHz.
- The short path increases to 80 ps < $t_{hold} + t_{skew}$ = 110 ps,

the circuit still violates its hold time constraint.

- To fix the problem, even more buffers could be inserted. Buffers would need to be added on the critical path as well, reducing the clock frequency. Alternatively, a better flip-flop with a shorter hold time might be used.

Metastability



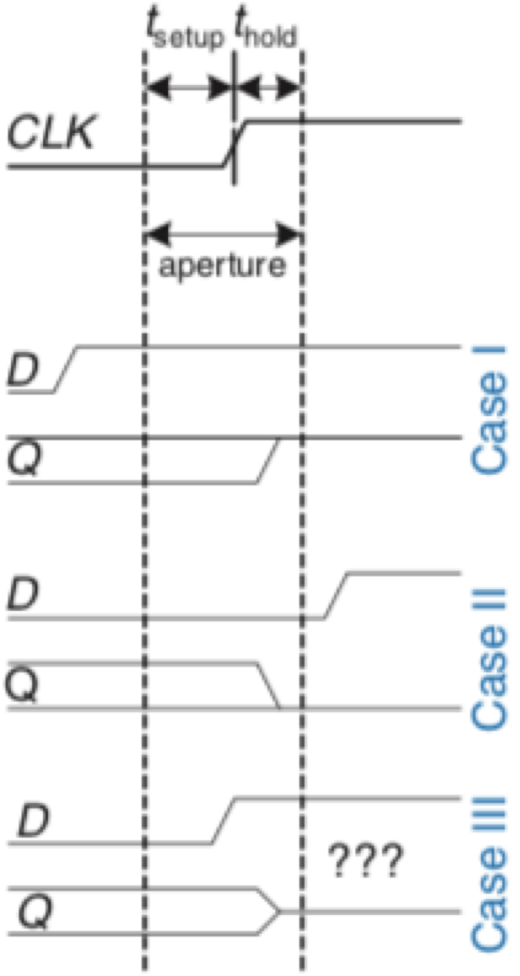
It is not always possible to guarantee that the input to a sequential circuit is stable during the aperture time.

What is the output Q after the rising edge of CLK?

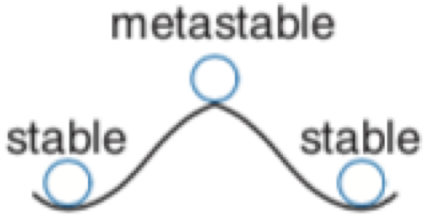
Case I: Q = 1.

Case II: Q = 0.

Case III: D changes between t_{setup} before CLK and t_{hold} after CLK, the input violates the dynamic discipline and the Q = undefined.



Metastable state: When a flip-flop samples an input that is changing during its aperture, Q may momentarily take on a voltage between 0 and V_{DD} that is in the forbidden zone. Eventually, the flip-flop will resolve the output to a stable state of either 0 or 1. However, the resolution time required to reach the stable state is unbounded. The time required for this change to occur depends on how nearly well balanced the ball originally was. Every bistable device has a metastable state between the two stable states.



Resolution Time

If an input to a flip-flop changes at a random time during the clock cycle, then:

Resolution time: t_{res} is a random variable

If the input changes outside the aperture, then: $t_{res} = t_{pcq}$

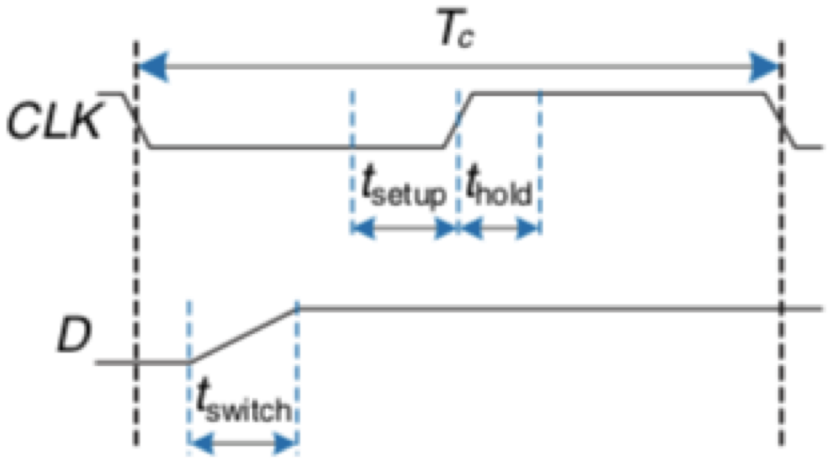
If the input happens to change within the aperture, t_{res} can be substantially longer

The probability that the resolution time, t_{res}, exceeds some arbitrary time, t, decreases exponentially with t:
(The equation is valid only for t substantially longer than t_{pcq})

$$P(t_{res} > t) = \frac{T_0}{T_c} e^{-\frac{t}{\tau}}$$

Where:
T_c the clock period
T_0 and τ are characteristic of the flip- flop.

$$P(t_{res} > t) = P(\text{samples changing input}) \times P(\text{unresolved})$$

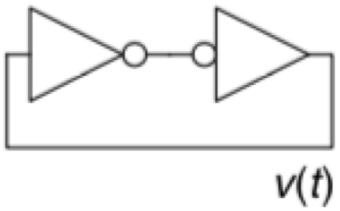


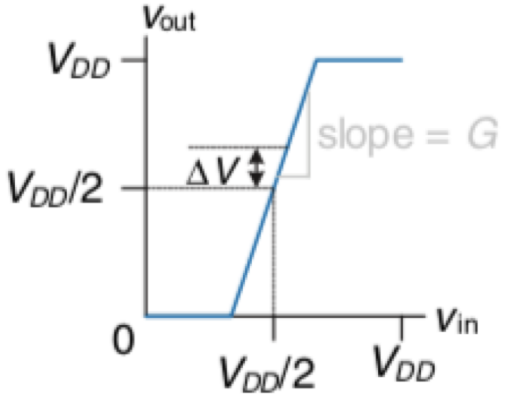
Switch Time

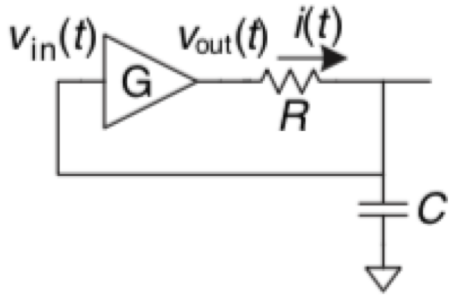
The probability that the input changes during the aperture around the clock edge:

$$P(\text{samples changing input}) = \frac{t_{switch} + t_{setup} + t_{hold}}{T_c}$$

P(unresolved): the probability that the flip-flop has not yet resolved to a valid logic level after a time t







Charging the capacitor through the resistor causes an RC delay, preventing the buffer from switching instantaneously

if $v_{in}(t) = VDD/2 + \Delta V/G$ then

$v_{out}(t) = VDD/2 + \Delta V$ for small ΔV

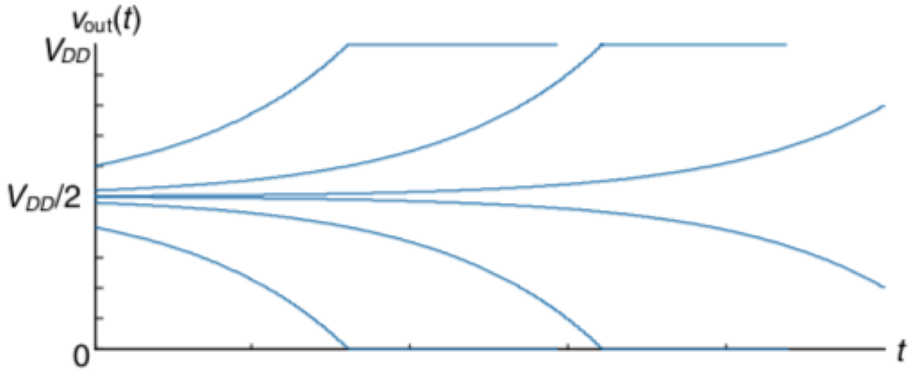
$i(t) = (v_{out}(t) - v_{in}(t))/R$

$dv_{in}(t)/dt = i(t)/C$

Linear first-order differential equation

$$\frac{dv_{out}(t)}{dt} = \frac{(G-1)}{RC}\left[v_{out}(t) - \frac{V_{DD}}{2}\right]$$

Solving with with I.C. $v_{out}(0) = V_{DD}/2 + \Delta V$



$$v_{out}(t) = \frac{V_{DD}}{2} + \Delta V e^{\frac{(G-1)t}{RC}}$$

Solving for the resolution time t_{res}, such that $v_{out}(t_{res})$ = VDD or 0, gives

$$|\Delta V| e^{\frac{(G-1)t_{res}}{RC}} = \frac{V_{DD}}{2}$$

$$t_{res} = \frac{RC}{G-1} \ln \frac{V_{DD}}{2|\Delta V|}$$

Let $\tau = \frac{RC}{G-1}$

The initial offset, ΔV_{res}, that gives a particular resolution time, t_{res}:

$$\Delta V_{res} = \frac{V_{DD}}{2} e^{-t_{res}/\tau}$$

- The resolution time increases if the bistable device has high resistance or capacitance that causes the output to change slowly
- It decreases if the bistable device has high gain, G
- increases logarithmically as the circuit starts closer to the metastable point ($\Delta V \rightarrow 0$)

The probability that the bistable device samples the input at a time to obtain a sufficiently small initial offset is

$$P(\text{unresolved}) = P\left(\left|v_{\text{in}}(0) - \frac{V_{DD}}{2}\right| < \frac{\Delta V_{res}}{G}\right) = \frac{2\Delta V_{res}}{GV_{DD}}$$

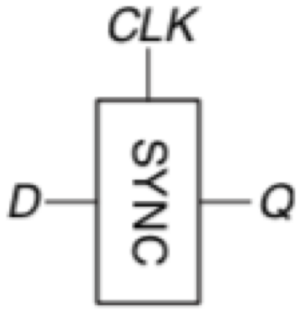
The probability that the resolution time exceeds some time t is given by the following equation:

$$P(t_{res} > t) = \frac{t_{\text{switch}} + t_{\text{setup}} + t_{\text{hold}}}{GT_c} e^{-\frac{t}{\tau}}$$

$T_0 = (t_{switch} + t_{setup} + t_{hold})/G$ and $\tau = RC/(G - 1)$

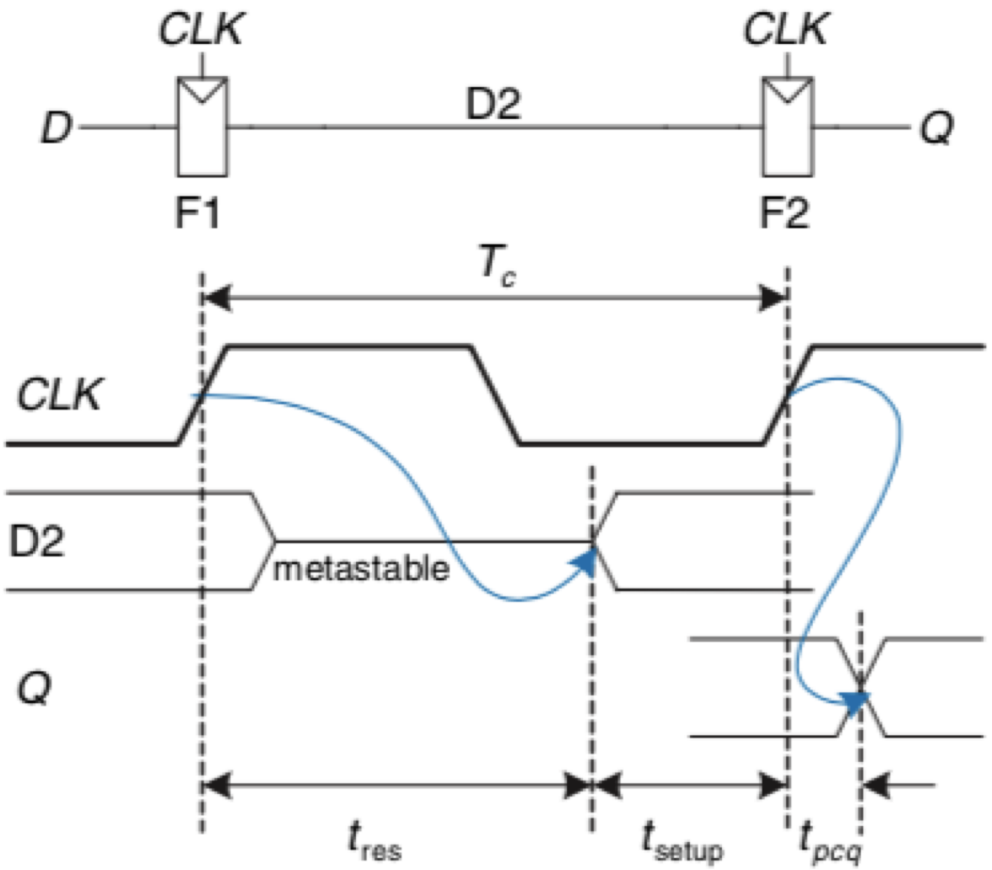
Synchronizers

- Asynchronous inputs to digital systems from the real world are inevitable
- To guarantee good logic levels, all asynchronous inputs should be passed through synchronizers.



A device that receives an asynchronous input D and a clock CLK. It produces an output Q within a bounded amount of time which has a valid logic level with extremely high probability. If D is stable during the aperture, Q should take on the same value as D. If D changes during the aperture, Q may take on either a HIGH or LOW value but must not be metastable. The synchronizer may fail if

$$t_{res} > T_c - t_{\text{setup}}$$



The probability of failure for a single input change at a random time is

$$P(\text{failure}) = \frac{T_0}{T_c} e^{-\frac{T_c - t_{\text{setup}}}{\tau}}$$

if D changes N times per second, the probability of failure per second is N times as great:

$$P(\text{failure})/\text{sec} = N \frac{T_0}{T_c} e^{-\frac{T_c - t_{\text{setup}}}{\tau}}$$

System reliability is usually measured in mean time between failures (MTBF). It is the reciprocal of the probability that the system will fail in any given second

$$MTBF = \frac{1}{P(\text{failure})/\text{sec}} = \frac{T_c e^{\frac{T_c - t_{\text{setup}}}{\tau}}}{NT_0}$$

Note: MTBF improves exponentially as the synchronizer waits for a longer time, T_c.

Example

- The traffic light controller FSM from receives asynchronous inputs from the traffic sensors. Suppose that a synchronizer is used to guarantee stable inputs to the controller. Traffic arrives on average 0.2 times per second. The flip-flops in the synchronizer have the following characteristics: τ = 200 ps, T0 = 150 ps, and tsetup = 500 ps. How long must the synchronizer clock period be for the MTBF to exceed 1 year?

Solutions

1 year ≈ π × 107 seconds

$$\pi \times 10^{7} = \frac{T_c \, e^{\frac{T_c - 500\times10^{-12}}{200\times10^{-12}}}}{(0.2)(150\times10^{-12})}$$

No closed form solution

Solve numerically!

T_c = 3.036 ns

PARALLELISM

Speed of a system is characterized by

- Latency
- Throughput

Token to be a group of inputs that are processed to produce a group of outputs

Latency of a system is the time required for one token to pass through the system from start to end.

Throughput is the number of tokens that can be produced per unit time

The throughput can be improved by processing several tokens at the same time.

Parallelism

Spatial

Temporal (Pipelining)

Multiple copies of the hardware are provided so that multiple tasks can be done at the same time

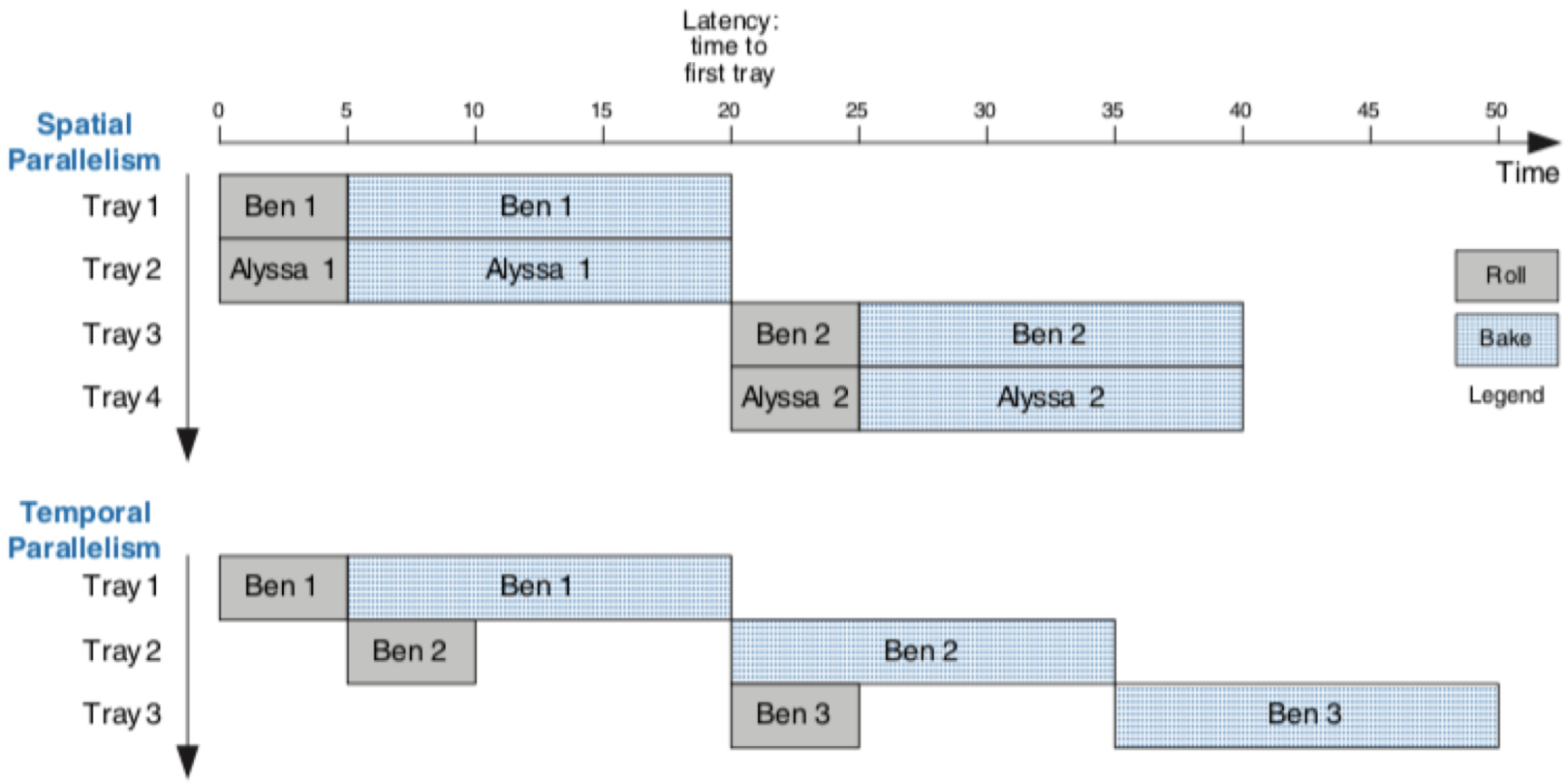


Figure 3.57 Spatial and temporal parallelism in the cookie kitchen

Consider a task with latency L

No parallelism

Throughput = 1/L

Spatially parallel system with N copies of the hardware

Throughput = N/L

Temporally parallel system

Ideally, the task is broken into N steps, or stages, of equal length

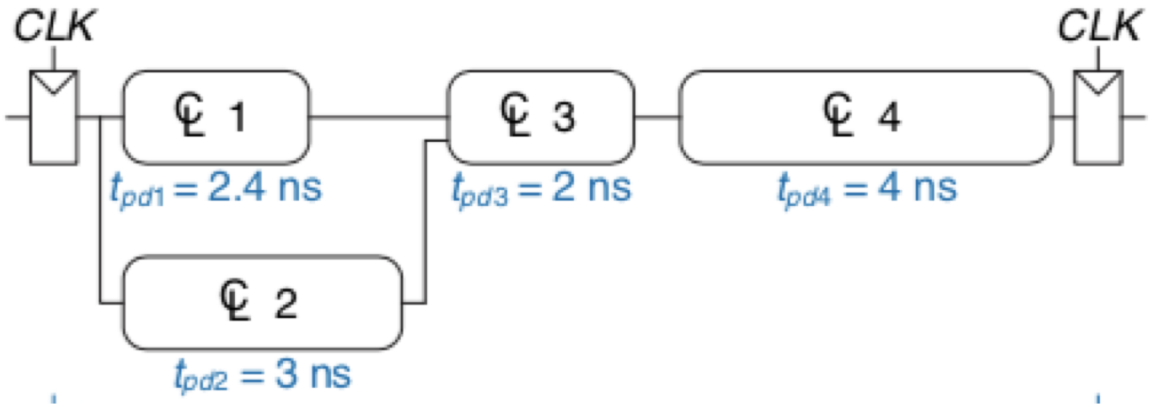
Throughput = N/L
Without replicating the hardware!

However, If the longest step has a latency L1

Throughput is 1/L1

Example

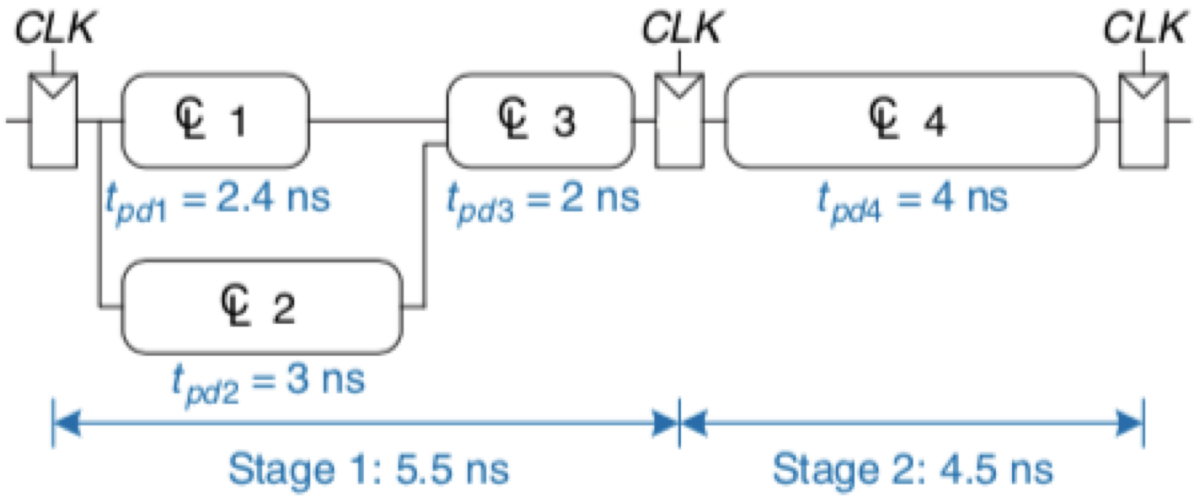
Assume that the register has a clock-to-Q propagation delay of 0.3 ns and a setup time of 0.2 ns. Find Tc, latency, and throughput.



Solution

- Latency = T_c =0.3+3+2+4+0.2=9.5 ns
- Throughput = 1/9.5 ns = 105 MHz

Two-stage Pipeline

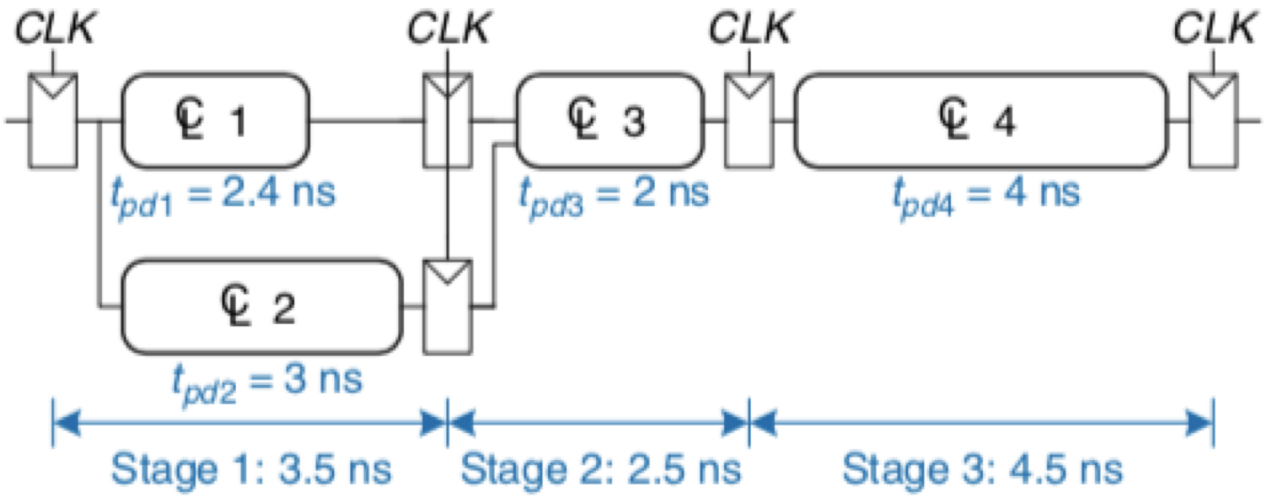


The first stage has a minimum clock period of 0.3+3+2+0.2=5.5 ns
The second stage has a minimum clock period of 0.3+4+0.2=4.5 ns

Tc = 5.5 ns
Latency = 11 ns (2 clock cycles)
Throughput = 1/5.5 ns = 182 MHz

Note: ideal pipelining would exactly double the throughput at no penalty in latency.

Three- stage pipeline



Tc = 4.5 ns
Latency = 13.5 ns (3 clock cycles)
Throughput = 1/4.5 ns = 222 MHz