Lesson 2: Evaluation of physical parameters of the design

Outline

1. Metrics
2. Static Time Analysis (STA)
3. Segmentation
4. Dynamic behavior
5. Area analysis
6. Power consumption

Why to evaluate?

- To obtain faster designs
  - Greater processing capacity.
  - To reduce the amount of hardware resources needed to carry out the same computations.
- To obtain smaller designs
  - Lower manufacturing cost.
  - To decrease the probability of physical faults during the manufacturing process.
  - The integration chip count per silicon wafer increases, which has a clear impact on the profits.
- To obtain energy-efficient designs
  - Lower negative impact on the environment.
  - Greater battery lifetime.
  - Lower operating cost.

Clock frequency

- Which is the maximum operating frequency of the circuit?
- How many operations per time unit can the circuit carry out?
- Will the design work properly at the desired clock frequency?

Resources

- How many resources do the circuit use?
- How can it be measured?
- If the layout of the circuit is not available, how can it be estimated?

Energy

- How much energy does the circuit consume?
  - Battery lifetime
- How much power?
  - Temperature
- How to measure it?

Outline

1. Metrics
2. Static Time Analysis (STA)
3. Dynamic behavior
4. Area analysis
5. Power consumption
6. Segmentation

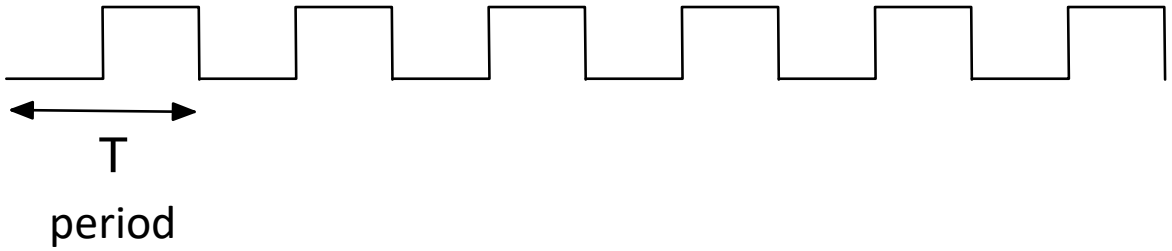
STA of sequential systems

- What should we care about?
  - Once we have obtained a circuit (or while we are getting close to it), we should make sure that the circuit works correctly and that it is efficient:
    - The time needed to obtain the result → How many clock cycles?
    - The clock period.
    - Does the clock signal, as well as all the control and data ones, arrive at their destinations **on time**?
- The analysis of the timing of a circuit is named Static Timing Analysis (STA).

Synchronous operation

- If the system contains memory elements (FFs, registers, counters, …), then it must have a clock signal.

$$f = \frac{1}{T}$$



- The clock determines **when** and **how often** the state of the sequential elements of the system are updated.
- If the circuit under test is synchronous, the difference between two consecutive rising (or falling) edges in the clock must be greater than the time needed to stabilize the new input values in the sequential elements of the system.

How is it done?

- Procedure
  - Calculate the delay of all the paths of the circuit.
  - Determine if the greatest delay is greater or lower than the threshold value imposed by the clock:
    - If it is greater, then timing violation.

    - If it is lower, then the slowest path determines the maximum operating frequency of the system.

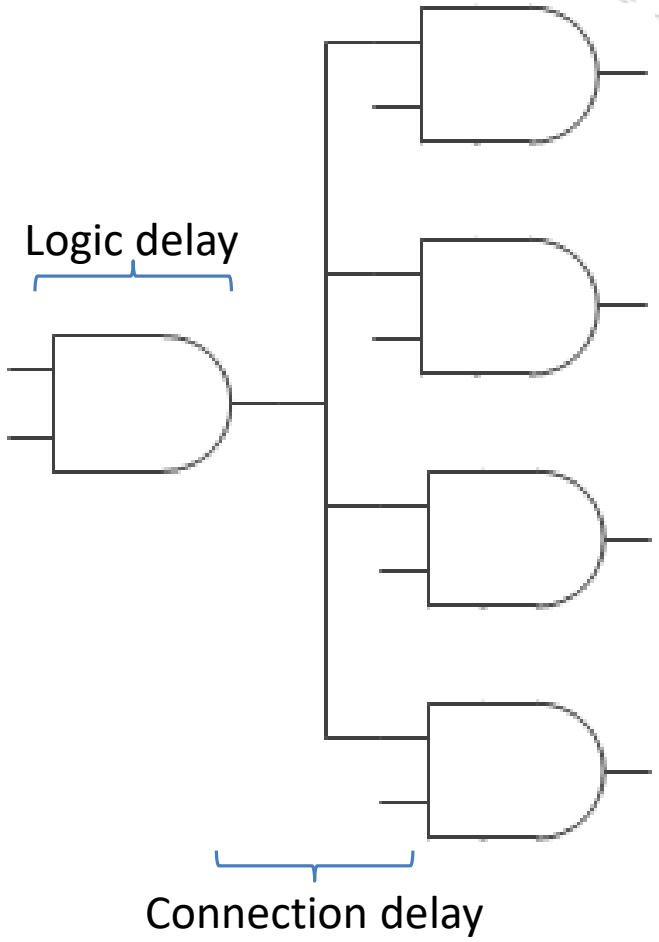
Path delay

- The sum of the delays in the logic and the connections from:
  1. The inputs to the outputs for all the possible combinatorial paths.
  2. The inputs to memory elements (FFs, registers ,…).
  3. The memory elements to the outputs.
  4. A memory element to another memory element.

- Too many values?
  - We care about the greatest delay (i.e., the *worst-case delay*)

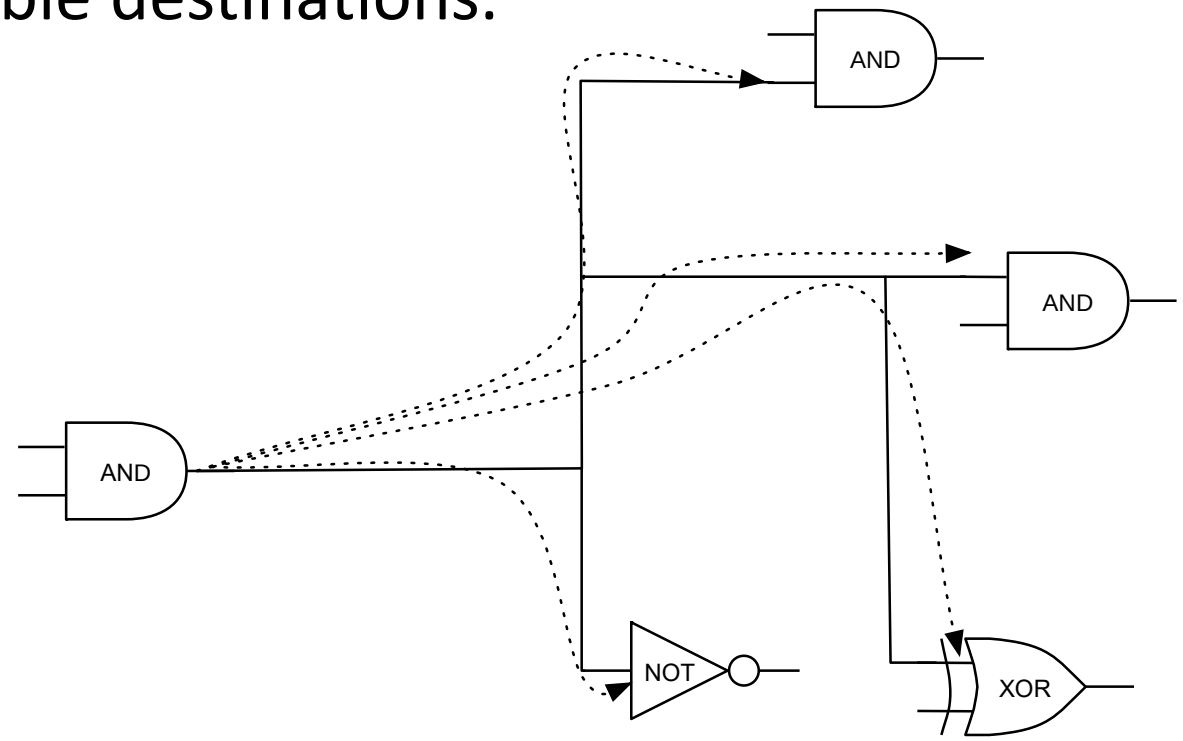
Path delay

- Logic delay: delay of the gates or logical elements of the circuit.
  - Combinatorial
  - Sequential
- Connection delay: All the physical connections in the device have a delay.



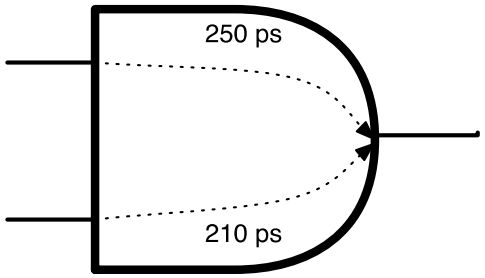
Delay modeling: Timing arcs

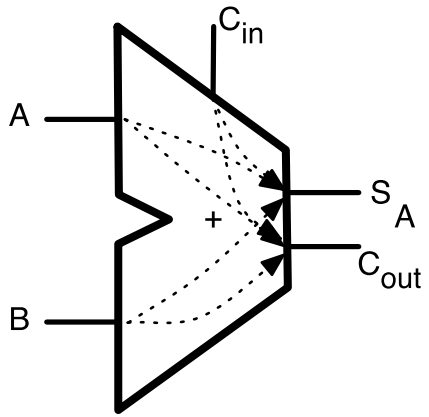
- Connections:
  - Delay from the output pin to each one of the possible destinations.



Delay modeling: Timing arcs

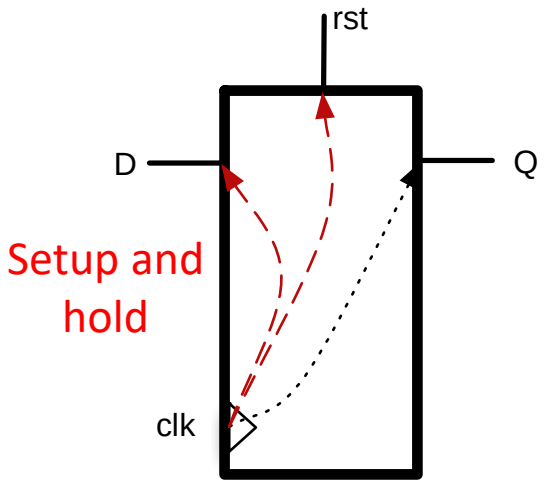
- Combinatorial:
  - Delay from any input to any output
    - AND, OR, adders, decoders, multiplexers, …





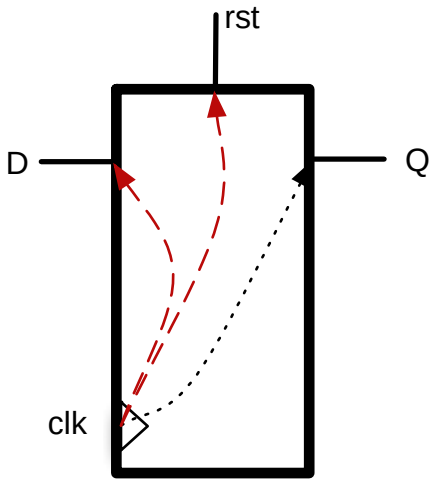
Delay modeling: Timing arcs

- Sequential:
  - FFs, counters, registers, memories, …
  - Delay from clock pin to any output.
  - Restrictions to fulfill to avoid malfunctioning (setup, hold, …)



Definitions

- **Set-up time**.
  - The minimum amount of time the data signal should be held steady before the clock event so that the data are reliably sampled by the clock.
- **Hold time**.
  - The minimum amount of time the data signal should be held steady after the clock event so that the data are reliably sampled.
- **Clk-to-Q**.
  - The time it takes for the register or FF output to be in a stable state after a clock edge occurs.



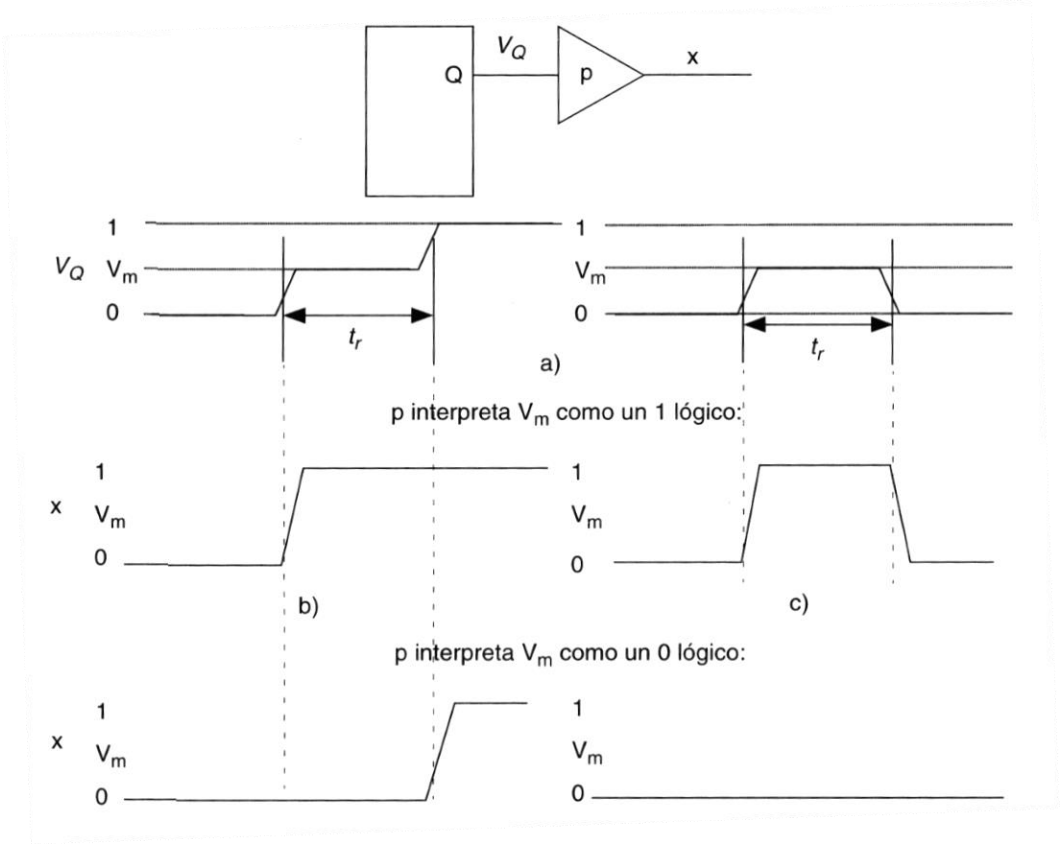
Metastability

- Metastability will occur If there is any *setup* or *hold* violation.
- It is a state of sustained instability in a FF, it tries to converge towards one of its stable states.
- While the FF operates under metastability, its output signals are undefined at the logical level.
- It is in this state for an undetermined time. The probability of staying at a metastable state follows an exponentially decreasing curve controlled by a time constant, τ (also known as *coefficient of resolution*).

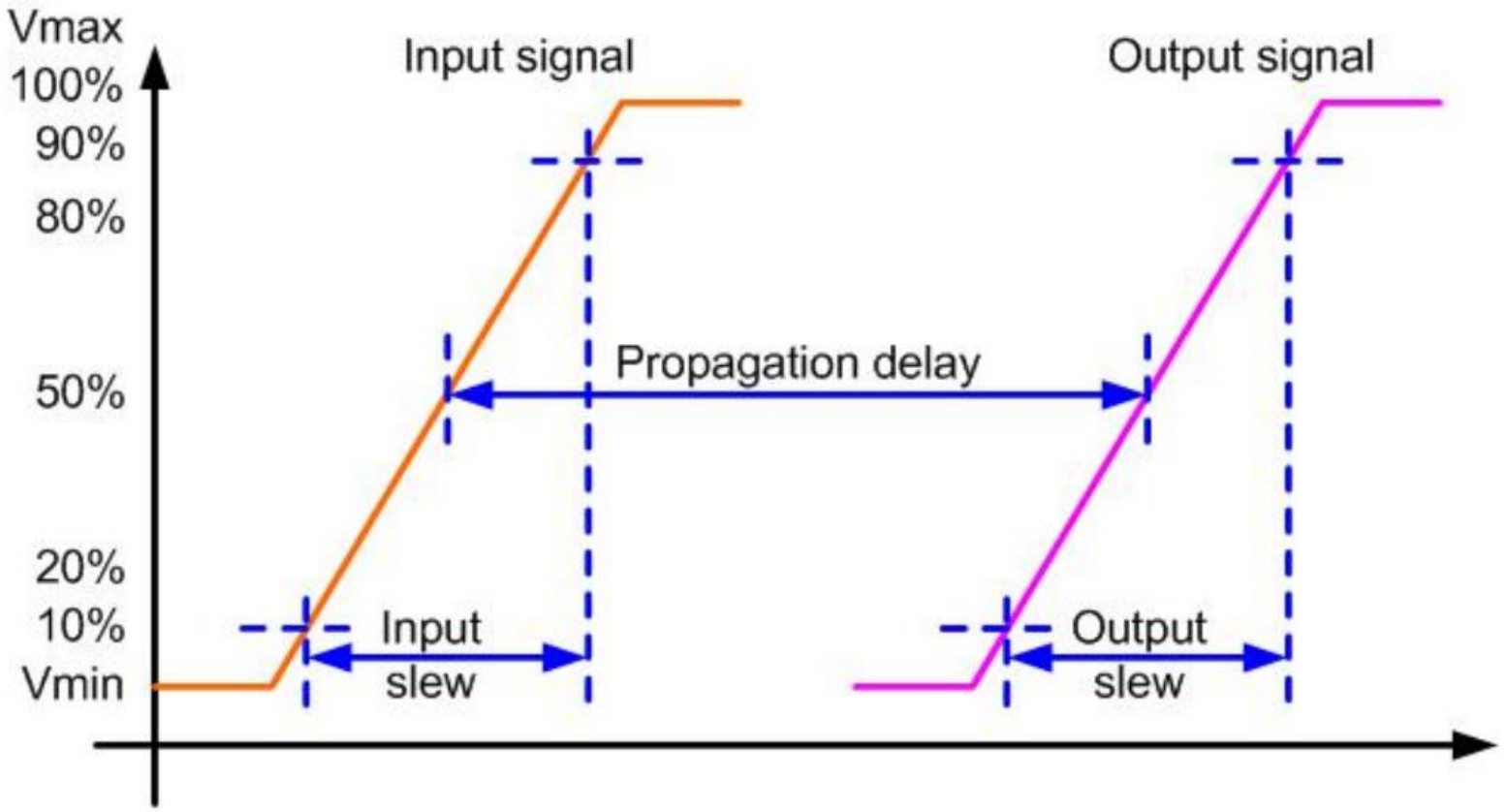
$$P(t > t_c) \mu e^{-\frac{t_c}{t}}$$

Metastability

- It leads to functional errors, since metastable signals are not correctly defined.

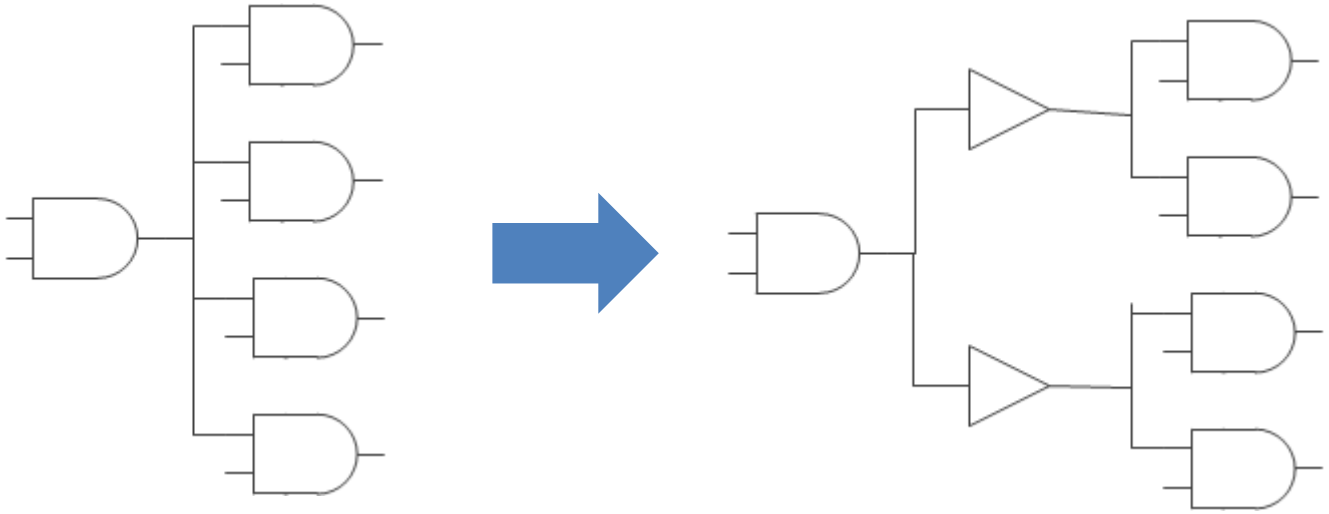


Logic delay: what is it?



Logic delay: dependencies

- Type of logic: AND, OR, FF, ADDER, …
- Fan-out: a greater fan-out leads to a greater delay



Logic delay: dependencies

- Type of data:
  - Transition 0 to 1 ≠ transition 1 to 0.
- Transition time in input data
  - Greater transition time ⟹ greater delay.

Fast input transition

Slow input transition

Logic delay: dependencies

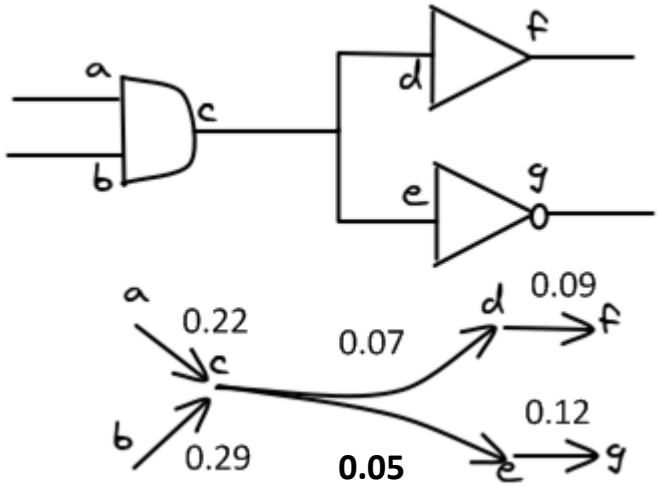
- Temperature
  - Greater temperature $\Longrightarrow$ slower hardware.
- Voltage supply
  - Lower voltage $\Longrightarrow$ slower hardware.
- Manufacturing process
  - Slow processes.
  - Fast processes.

Connection delay: dependencies

- Dimensions of the connection
- Topology of the connection
- Manufacturing process
- Fan-out of the connection
- Material resistivity
- Number of tracks
- Crosstalk

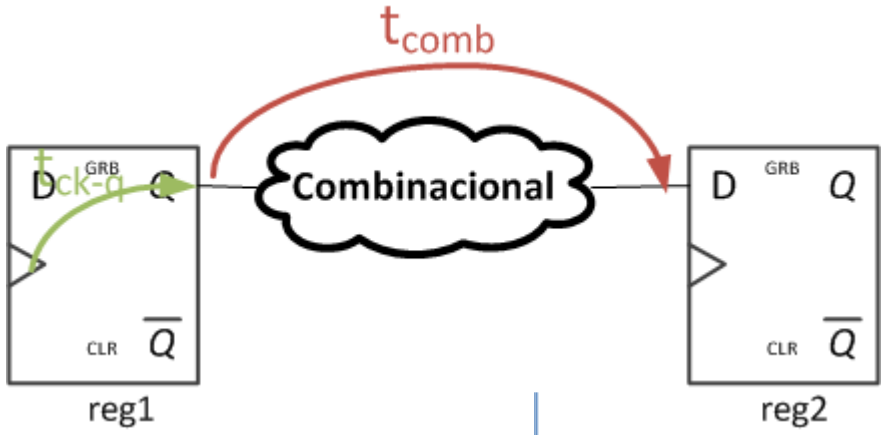
Calculation of the propagation delay

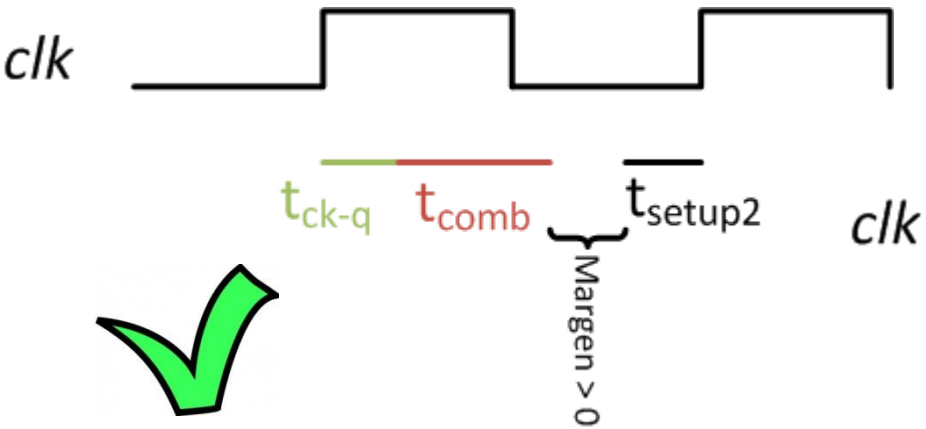
- The sum of delays (*timing arcs*) of the logic and the connections for all the paths of the combinatorial circuit.
- It depends on the path. There are as many times as paths.
- The path with the greatest delay is the **critical path**.

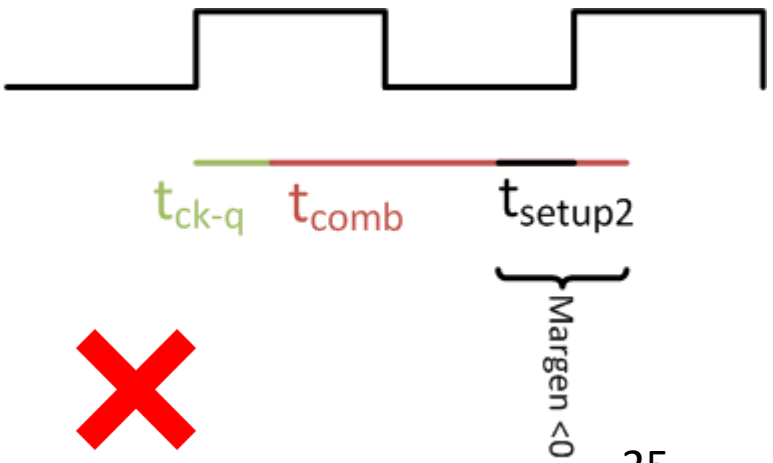


Delay = 0.29 + 0.05 + 0.12 = 0.46 ns

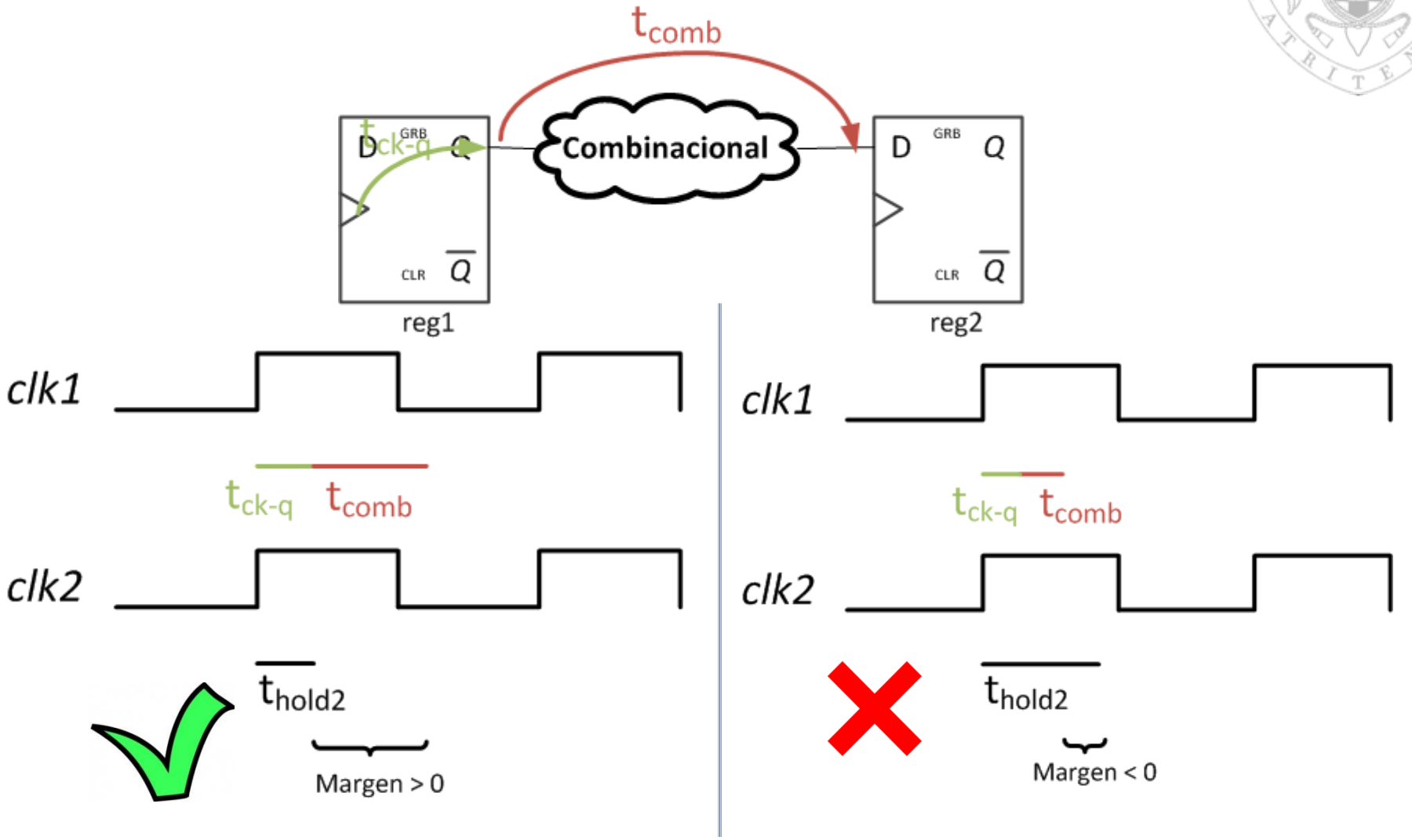
Setup margin



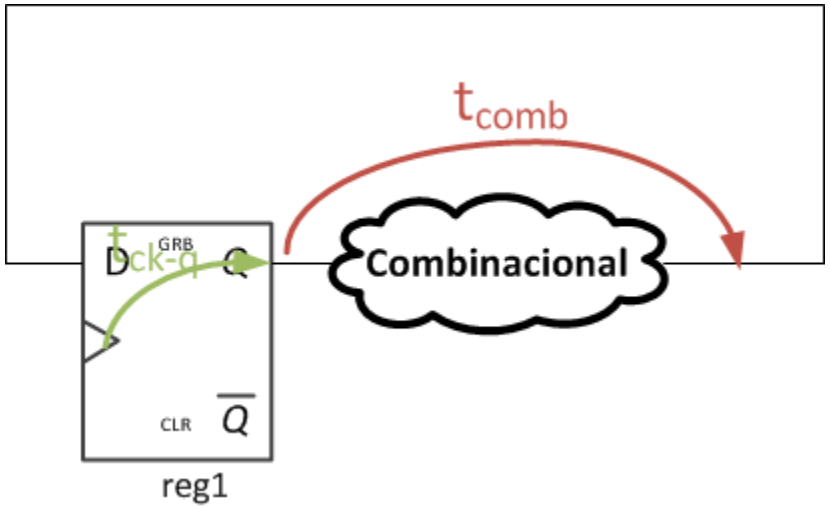


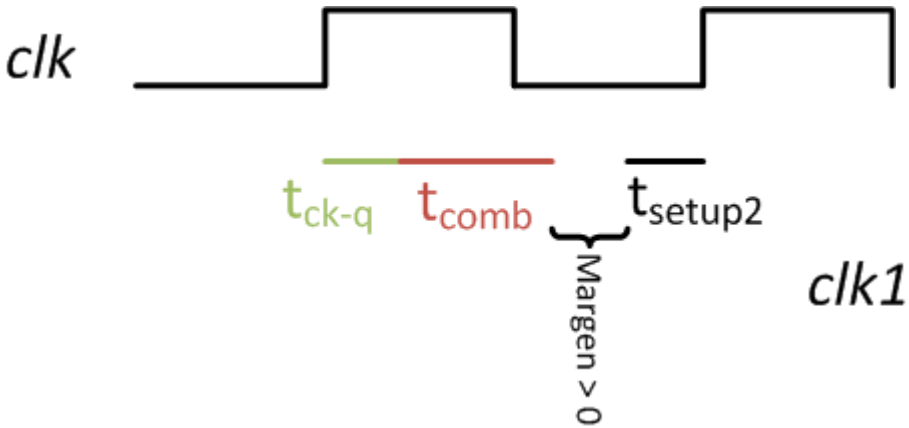


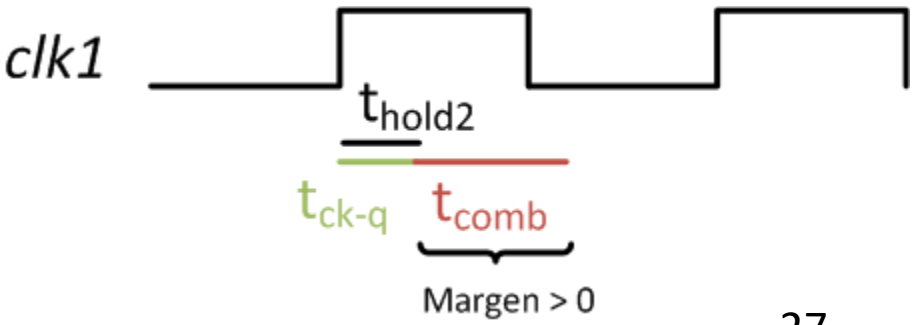
Hold margin


tcomb
Combinacional
D
GRB
Q
CLR
Q
reg1
reg2
clk1
clk2
tck-q
tcomb
thold2
Margen > 0
Margen < 0

Circuit with combinatorial feedback







Clock skew

- This is a phenomenon in synchronous circuits in which the clock signal arrives at different components at different times.
  - This can be caused by many different reasons, such as wire-interconnect length, temperature variations, capacitive coupling, material imperfections…
- If the clock cycle is short, then this problem becomes a major issue.

Clock skew. Types.

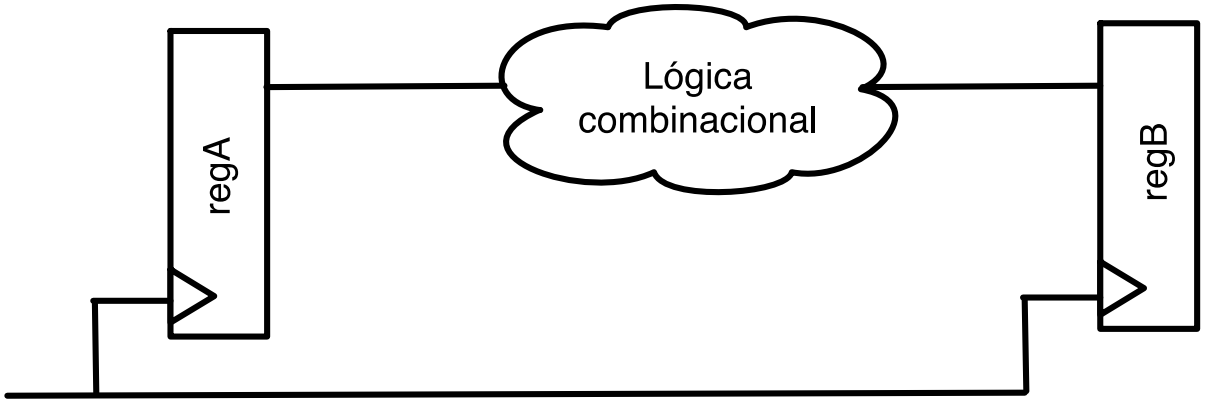
- There are two types of clock skew:
  - ***Positive skew*:** the sending register receives the clock before the receiving register.
  - ***Negative skew***: the sending register receives the clock after the receiving register.

$$skew = t_{dest} - t_{orig}$$

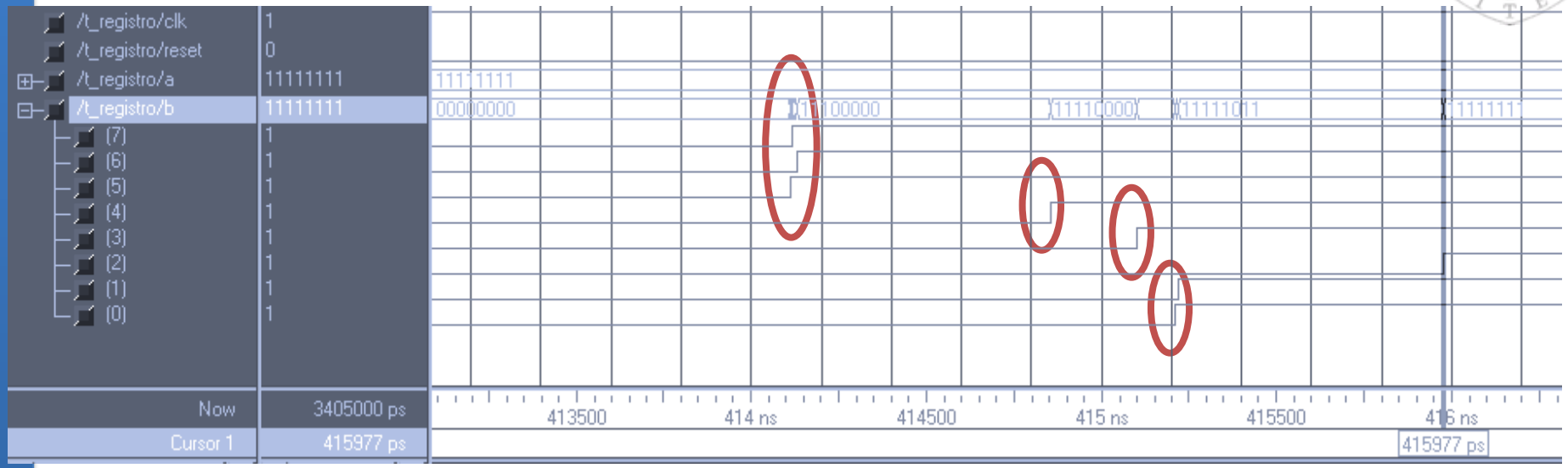
Clock skew. Example

- Times
  - The clock reaches regA in 10ns.
  - The clock reaches regB in 15ns.
  - When would this be a problem?

$$skew = 15 - 10 = 5\ ns$$



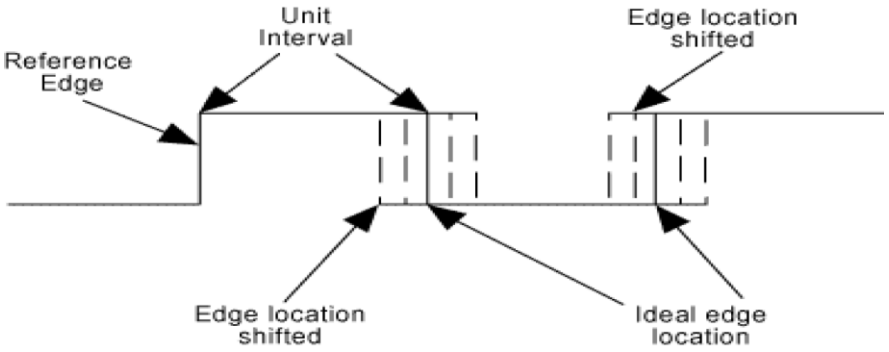
Clock skew. Example



Clk rising edge at T=405 ns
The first B *flip-flops* modify their value after T=414 ns
The last B *flip-flops* set at T=416ns
Total delay: 11 ns
Difference among B *flip-flops*: 2ns

Clock jitter

- **Jitter: It's an unwanted shift in the periodicity of the clock**. In other words, it's the variation in the clock edges with respect to their ideal positions.



Clock jitter

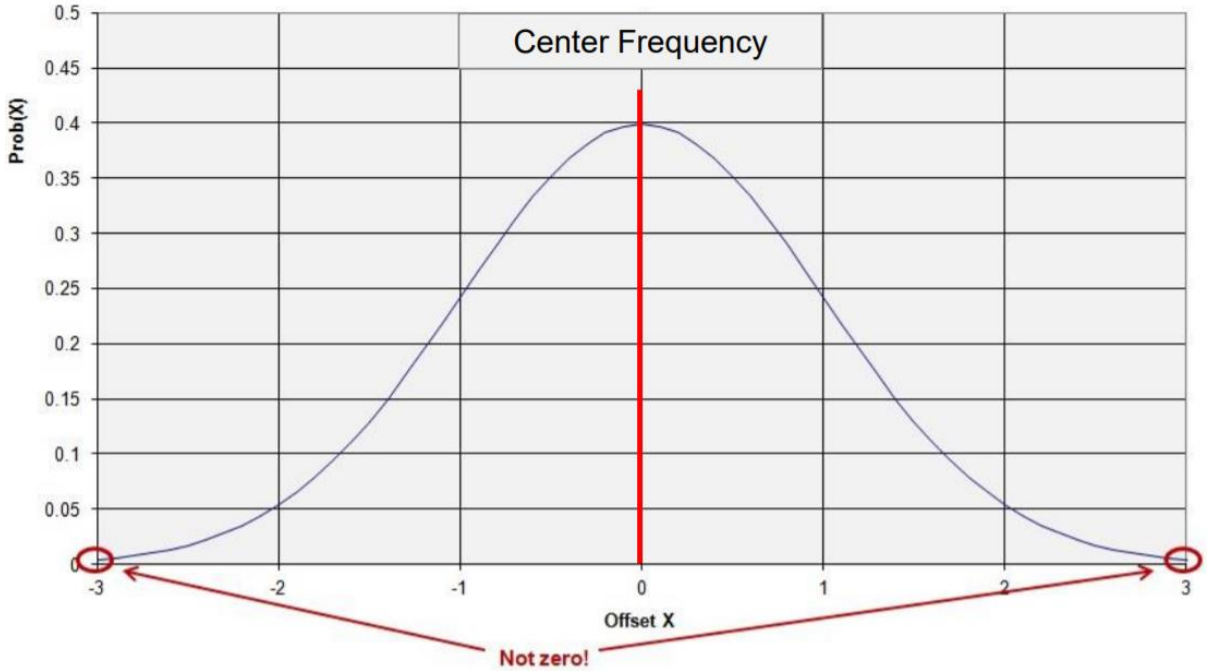
- Many possible origins.
  - Variations in the manufacturing of oscillators.
  - Noise in the voltage supply of PLLs (Phase-Locked Loops).
  - …
- It is specified in three ways:
  - Absolute jitter: Largest difference between the actual clock period the ideal one (absolute value).
  - Periodic jitter: Difference between the largest clock period and the smallest clock period for all individual clock periods within an observation window, typically 1,000 or 10,000 cycles. It is the most important one from the point of view of the STA.
  - Cycle-to-cycle jitter: Largest difference in the duration of two consecutive clock cycles.

Clock jitter

- It has to be specifically taken into account for the temporal analysis of the circuit. The reason is that it may actually shorten the clock cycle.
- Two ways of measuring (on a sample with typically 1,000 or 10,000 periods):
  - Peak-to-peak value (previous slide)
  - Root Mean Square (RMS) value (also known as Quadratic Mean)

$$x_{\mathrm{RMS}} = \sqrt{\frac{1}{n}\left(x_1^2 + x_2^2 + \cdots + x_n^2\right)}$$

- It is a probabilistic phenomena: It follows a Gaussian distribution



https://www.silabs.com/documents/public/white-papers/timing-jitter-tutorial-and-measurement-guide-ebook.pdf

Calculation of setup and hold margins

- Taking the *clock skew* and the *clock jitter*:

$$\text{setup\_margin} = T_{clk} + skew - (t_{ck\_q} + t_{comb} + t_{setup} + \text{jitter})$$

$$\text{hold\_margin} = t_{ck\_q} + t_{comb} - (\text{skew} + \text{jitter} + t_{hold})$$

- Negative margin $\Longrightarrow$ **timing error**

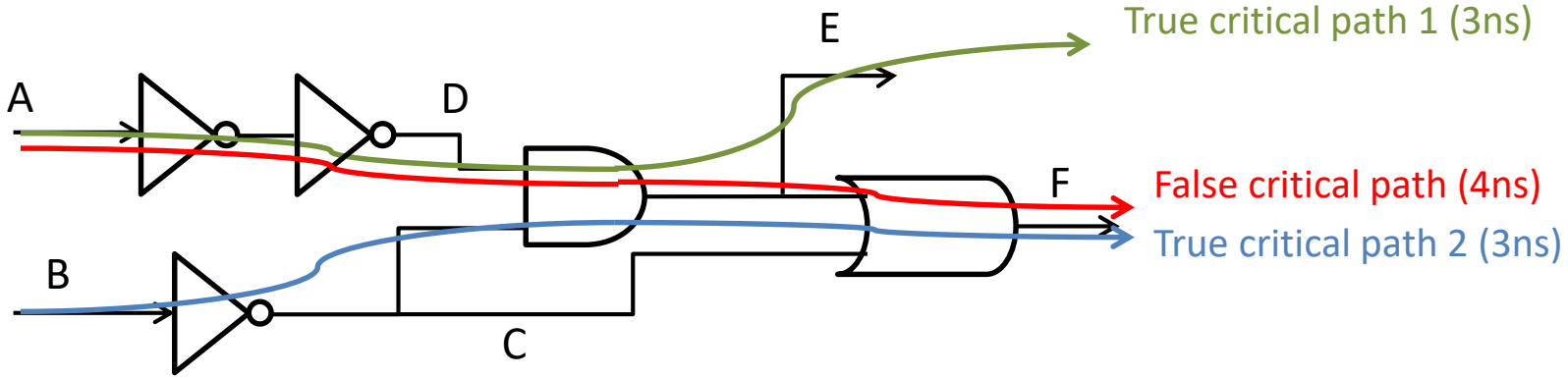
False critical path

- It seems that it is the critical path of a circuit, but it actually isn't!
- It does not propagate any transition between the input(s) and the output(s).
- The designs that share some logic to carry out different calculations are likely to have false critical paths.

False critical path

- The AND gate propagates the value of an input if the other one is '1'. Otherwise, it sets the output to '0'.
- The OR gate propagates the value of an input if the other one is '0'. Otherwise, it sets the output to '1'.

Gates and inverters delay = 1ns



One would think that the critical path is 4 ns. However, let's look closer:

- **If B=1**, then C=0. The AND gate does not have to wait to obtain D. It returns 0, which makes F = 0. **3 gate levels**.
- **If B=0,** then C=1. The AND gate has to wait for D to return E (**3 gate levels**). In parallel, since C=1, the OR gate returns 1 without having to wait for E (**2 gate levels**).

Timing example

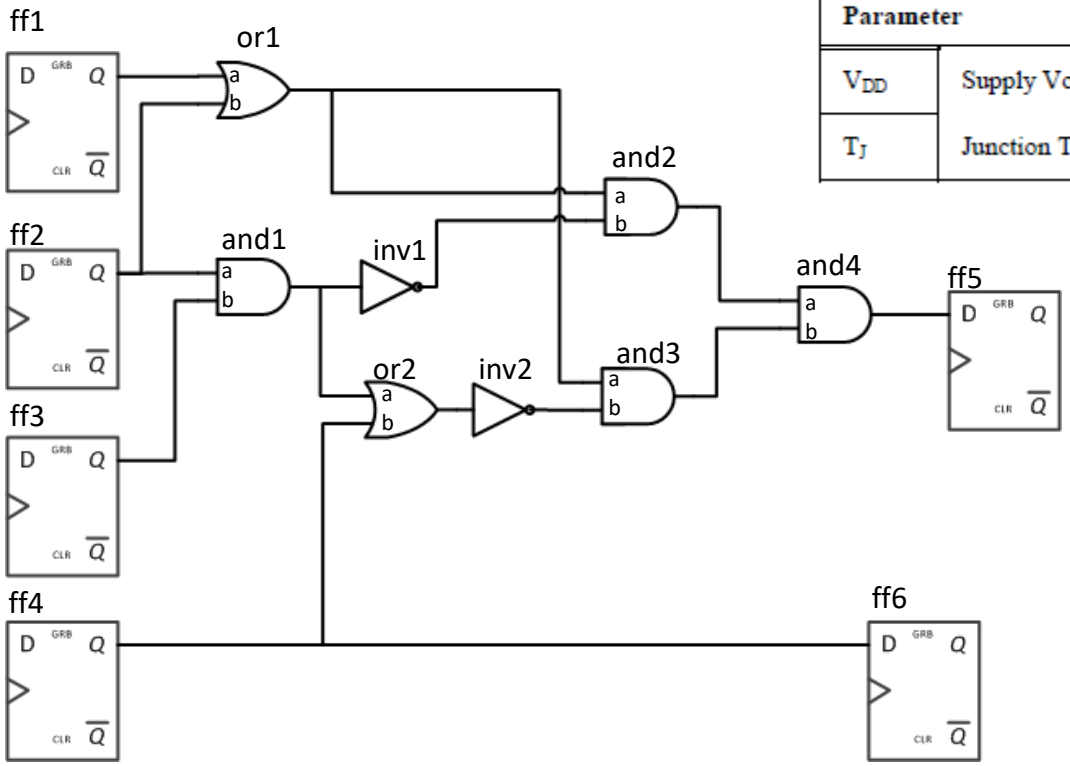


Table 2.2 Recommended Operating Conditions

Parameter		Minimum	Typical	Maximum
V_{DD}	Supply Voltage	1.08V	1.2V	1.32V
T_J	Junction Temperature	-40°C	25°C	+125°C

Will this design work at a frequency of 1 GHz?
Clock skew: 110 ps
Clock jitter: 20 ps
Worst-case scenario analysis: -40 C, 1.32V

Timing example

Pin Description

Cell Name	Pin Cap.(pf)		Max Cap.(pf)
	A1	A2	Z
AN2D0	0.0007336	0.000779	0.02955

Propagation Delay(unit:ns)

(Characterization Condition:Process=Fast-Fast,Voltage=1.32v,Temp=-40degreeC)

Cell Name	Path	Parameter	Group1	Group2	Group3
			(<0.00099)pf	(0.00099-0.01481)pf	(>0.01481)pf
AN2D0	A1 to Z	t_{PLH}	0.0272+6.5217*Cload	0.0288+5.4376*Cload	0.0299+5.2985*Cload
		t_{PHL}	0.0236+4.8696*Cload	0.0251+3.8354*Cload	0.0258+3.7313*Cload
	A2 to Z	t_{PLH}	0.0289+6.5435*Cload	0.0305+5.4358*Cload	0.0316+5.2985*Cload
		t_{PHL}	0.0256+4.913*Cload	0.0272+3.8517*Cload	0.0282+3.7273*Cload

Pin Description

Cell Name	Pin Cap.(pf)	Max Cap.(pf)
	I	ZN
INVD0	0.0007637	0.02955

Propagation Delay(unit:ns)

(Characterization Condition:Process=Fast-Fast,Voltage=1.32v,Temp=-40degreeC)

Cell Name	Path	Parameter	Group1	Group2	Group3
			(<0.00099)pf	(0.00099-0.01481)pf	(>0.01481)pf
INVD0	I to ZN	t_{PLH}	0.0106+5.4783*Cload	0.0106+5.4611*Cload	0.0115+5.3704*Cload
		t_{PHL}	0.0084+4.3696*Cload	0.009+3.868*Cload	0.0094+3.827*Cload

Timing example

Pin Description

Cell Name	Pin Cap.(pf)		Max Cap.(pf)
	A1	A2	Z
OR2D0	0.000686	0.0007661	0.02955

Propagation Delay(unit:ns)

(Characterization Condition:Process=Fast-Fast,Voltage=1.32v,Temp=-40degreeC)

Cell Name	Path	Parameter	Group1	Group2	Group3
			(<0.00099)pf	(0.00099-0.01481)pf	(>0.01481)pf
OR2D0	A1 to Z	t_{PLH}	0.0208+5.8478*Cload	0.0215+5.3363*Cload	0.0225+5.2442*Cload
		t_{PHL}	0.0317+5.9565*Cload	0.0345+4.17*Cload	0.0374+3.7931*Cload
	A2 to Z	t_{PLH}	0.0225+5.8478*Cload	0.0232+5.3436*Cload	0.0239+5.2578*Cload
		t_{PHL}	0.0352+5.9565*Cload	0.038+4.1682*Cload	0.0413+3.7748*Cload

Pin Description

Cell Name	Pin Cap.(pf)		Max Cap.(pf)	
	CP	D	Q	QN
DFD1	0.000778	0.0009582	0.0591	0.0591

Propagation Delay(unit:ns)

(Characterization Condition:Process=Fast-Fast,Voltage=1.32v,Temp=-40degreeC)

Cell Name	Path	Parameter	Group1	Group2	Group3
			(<0.0017)pf	(0.0017-0.02947)pf	(>0.02947)pf
DFD1	CP to Q	t_{PLH}	0.0695+3.4946*Cload	0.0706+3.0432*Cload	0.0717+2.9835*Cload
		t_{PHL}	0.0814+2.9892*Cload	0.0835+2.1773*Cload	0.0852+2.0688*Cload
	CP to QN	t_{PLH}	0.0999+3.2258*Cload	0.1005+2.9793*Cload	0.1006+2.9733*Cload
		t_{PHL}	0.0874+2.9892*Cload	0.0896+2.162*Cload	0.0911+2.0688*Cload

Timing Constraint(unit:ns)

(Characterization Condition:Process=Fast-Fast,Voltage=1.32v,Temp=-40degreeC)

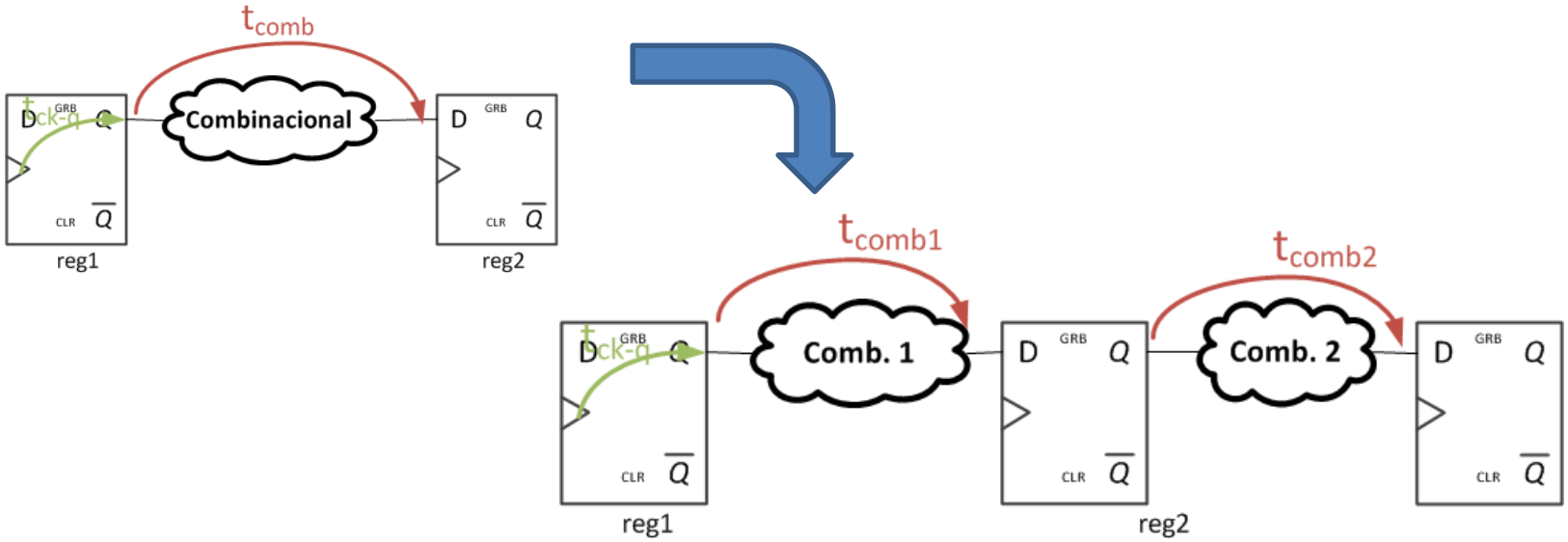
Cell Name	Path	Timing Parameter	Constraint
DFD1	CP	$t_{min\_pulse\_width}^{H}$	0.03906
		$t_{min\_pulse\_width}^{L}$	0.04883
	CP to D	$t_{setup\_rising}^{LH}$	0.008944
		$t_{hold\_rising}^{LH}$	0.00118
		$t_{setup\_rising}^{HL}$	-0.002623
		$t_{hold\_rising}^{HL}$	0.01714

Outline

1. Metrics
2. Static Time Analysis (STA)
3. Segmentation
4. Dynamic behavior
5. Area analysis
6. Power consumption

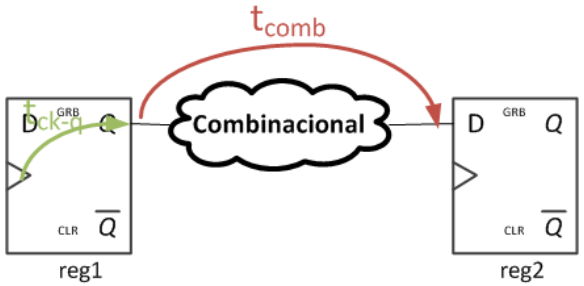
Segmentation: Definition

- Segmentation involves dividing a circuit into steps separated by means of FFs or registers. The outputs of the FFs or registers of a given step constitute the inputs for the following step.

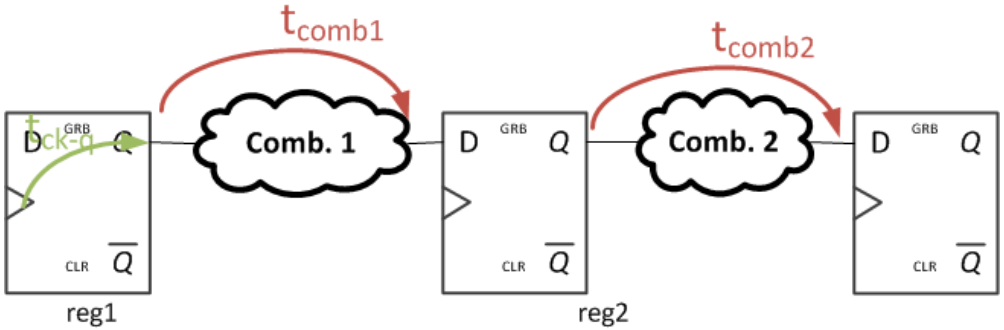


- **All the steps run concurrently**.

Minimum period



$$t_{cycle} > t_{ck\text{-}q} + t_{comb} + t_{setup2} - t_{skew} + t_{jitter}$$



$$t_{cycle1} > t_{ck\text{-}q} + t_{comb1} + t_{setup2} - t_{skew} + t_{jitter}$$

$$t_{cycle2} > t_{ck\text{-}q} + t_{comb2} + t_{setup3} - t_{skew} + t_{jitter}$$

$$t_{cycle} = \max(t_{cycle1}, t_{cycle2})$$

Performance

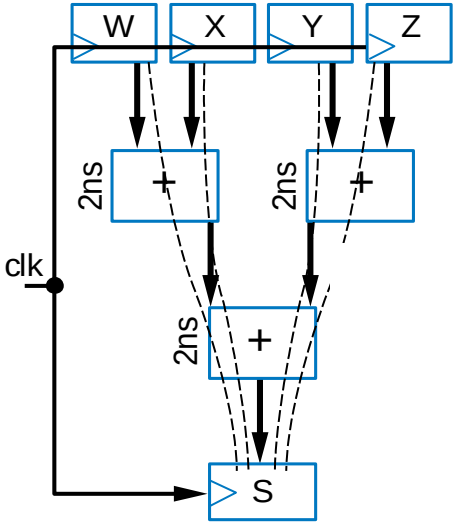
- Latency: Time elapsed until a new result is generated.
- Throughput: Rate of production of new output data per time unit.

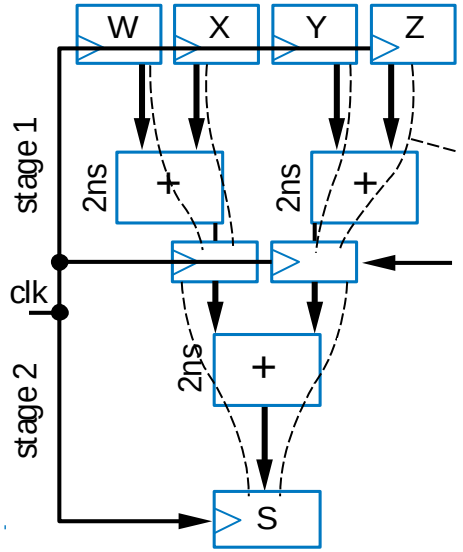
$$L = n \times t_{cycle}$$

$$T = \frac{1}{t_{cycle}}$$

Example I

t_{setup} = 12 ps
$t_{ck\text{-}q}$ = 70 ps

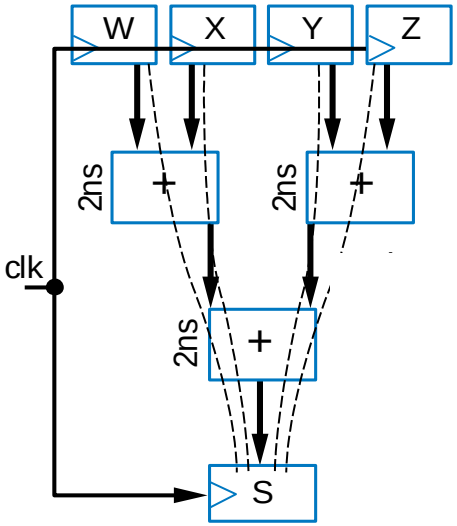


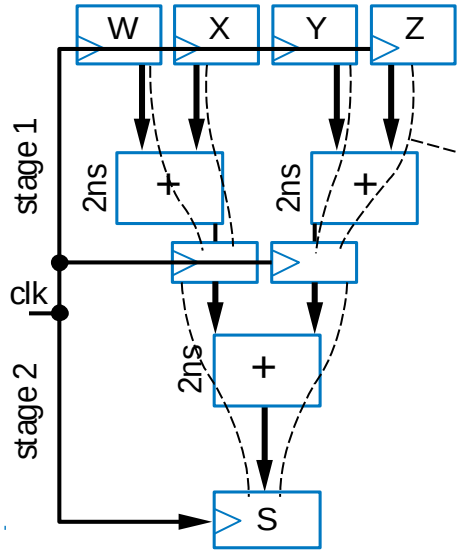


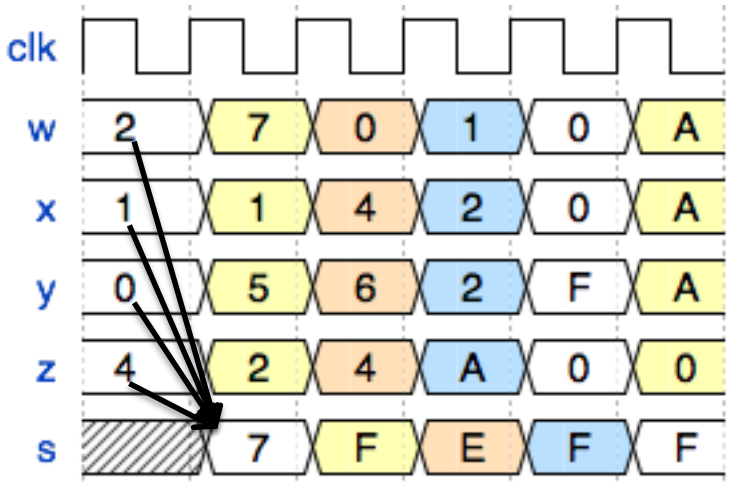
$$t_{cycle} = 4.082ns$$
$$L = 4.082 + 0.070 = 4.152ns$$
$$T = 244.9\ \text{Mdata/s}$$

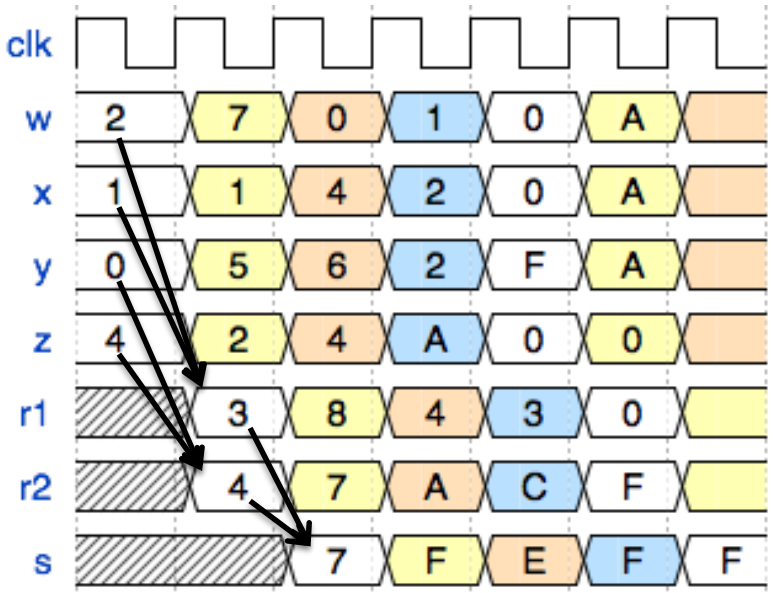
$$t_{cycle} = 2.082ns$$
$$L = 2 \cdot 2.082 + 0.070 = 4.234ns$$
$$T = 480.3\ \text{Mdata/s}$$

Example I

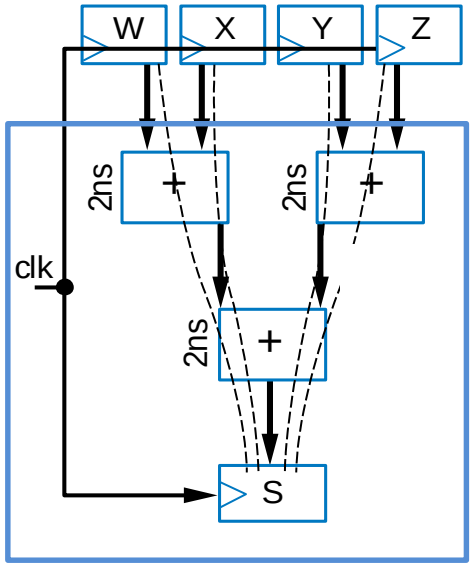

W
X
Y
Z
2ns
+
2ns
+
clk
2ns
+
S


W
X
Y
Z
stage 1
2ns
+
2ns
+
clk
stage 2
2ns
+
S


clk
w 2 7 0 1 0 A
x 1 1 4 2 0 A
y 0 5 6 2 F A
z 4 2 4 A 0 0
s 7 F E F F


clk
w 2 7 0 1 0 A
x 1 1 4 2 0 A
y 0 5 6 2 F A
z 4 2 4 A 0 0
r1 3 8 4 3 0
r2 4 7 A C F
s 7 F E F F

Example I (VHDL)



Option I

```
s1 <= w + x;
s2 <= y + z;
p_reg : process (clk, rst)
begin
  if rst = '1' then
    s <= (others => '0');
  elsif rising_edge(clk) then
    s <= s1 + s2;
  end if;
end process p_reg;
```

Option II

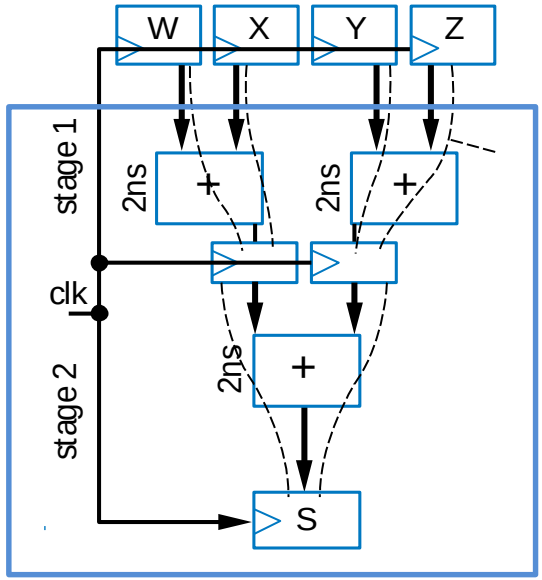
```
s3 <= w + x + y + z;
p_reg : process (clk, rst)
begin
  if rst = '1' then
    s <= (others => '0');
  elsif rising_edge(clk) then
    s <= s3;
  end if;
end process p_reg;
```

Option III

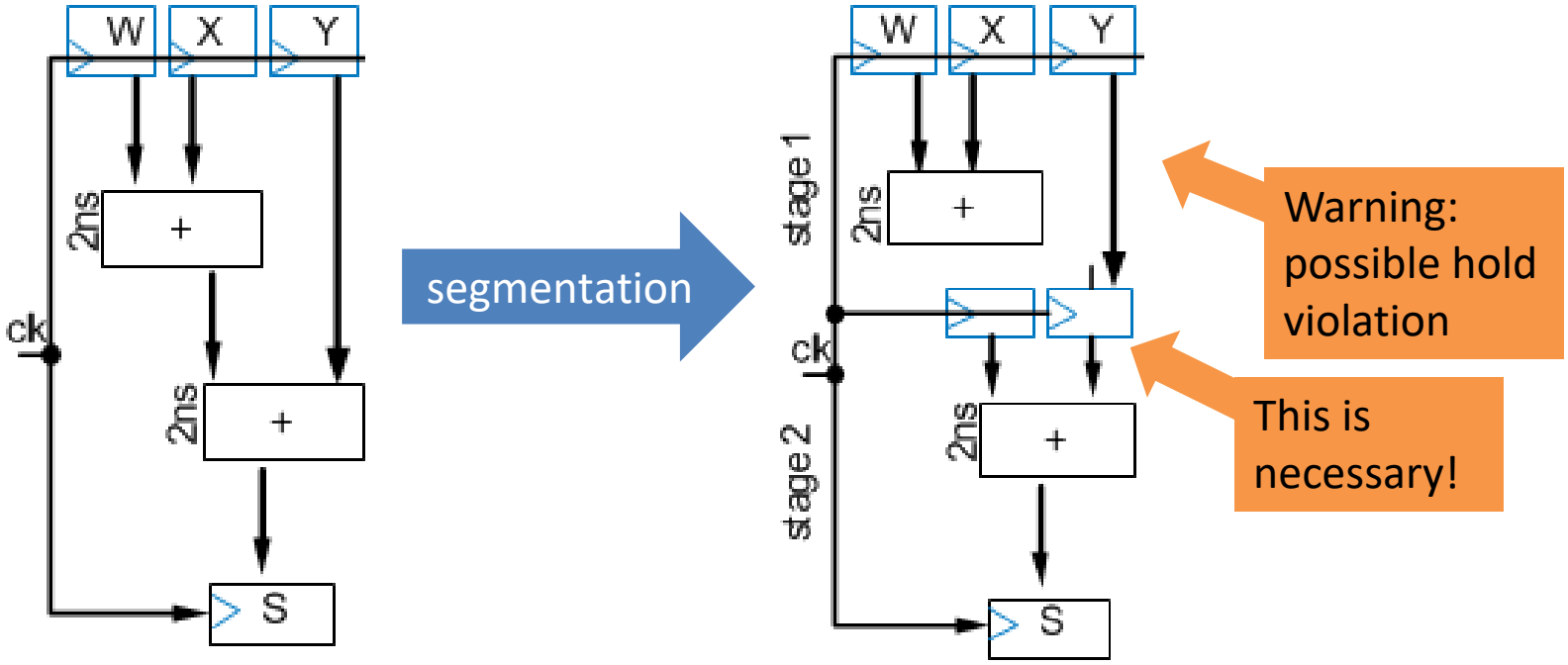
```
p_reg : process (clk, rst)
begin
  if rst = '1' then
    s <= (others => '0');
  elsif rising_edge(clk) then
    s <= w + x + y + z;
  end if;
end process p_reg;
```

Example I (VHDL)

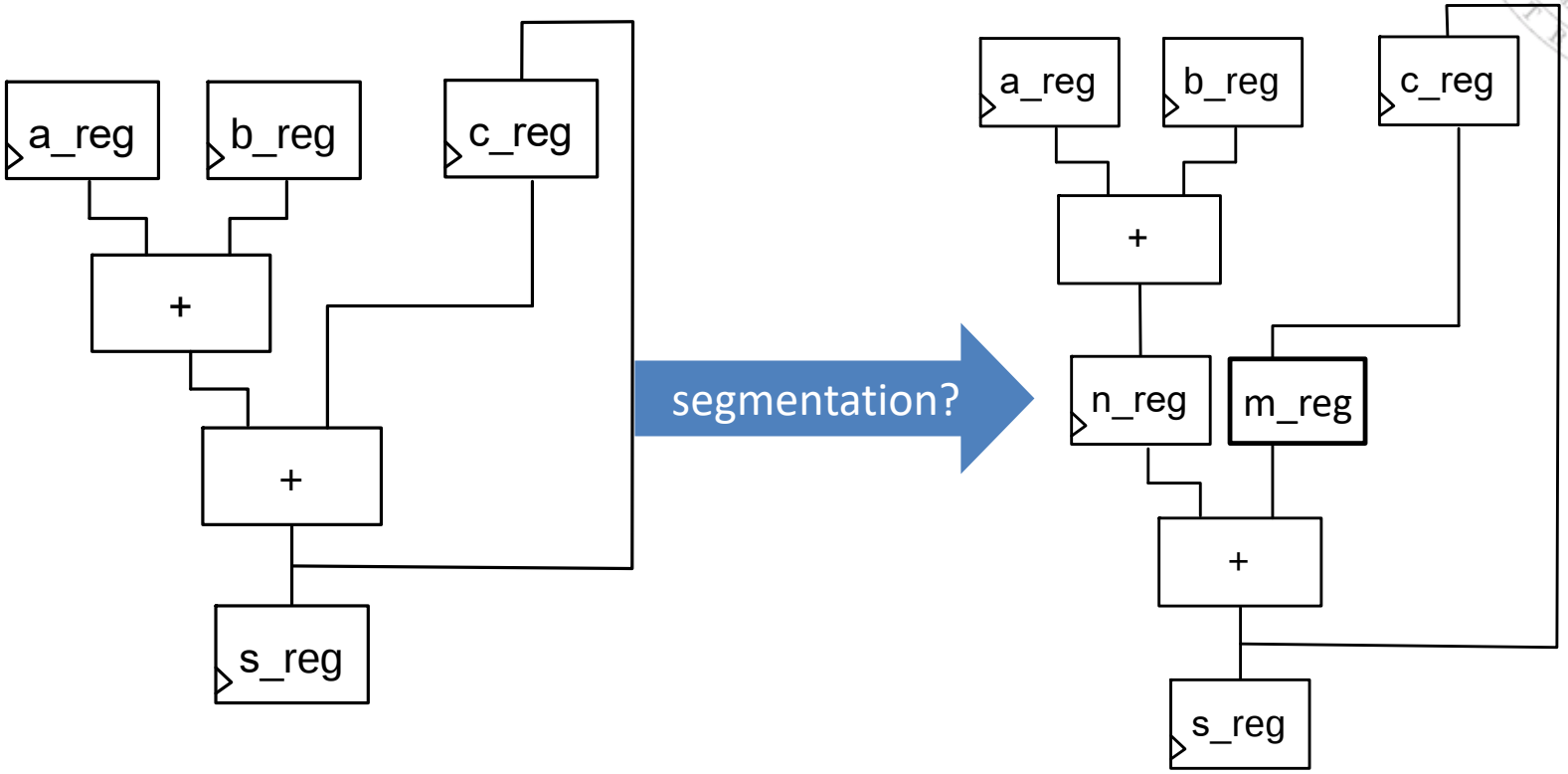
```
p_s_reg : process (clk, rst) begin
  if rst = '1' then
    s1 <= (others => '0');
    s2 <= (others => '0');
    s <=  (others => '0');
  elsif rising_edge(clk) then
    s1 <= w + x;
    s2 <= y + z;
    s  <= s1 + s2;
  end if;
end process p_s_reg;
```



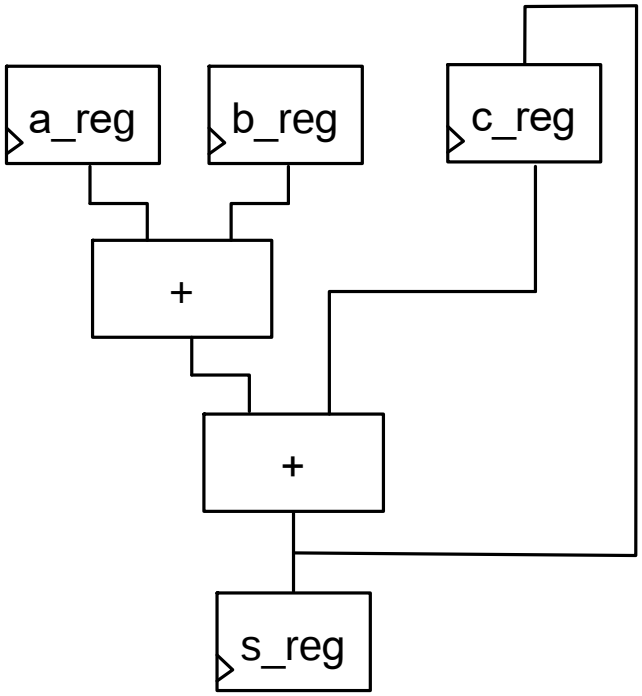
Example II

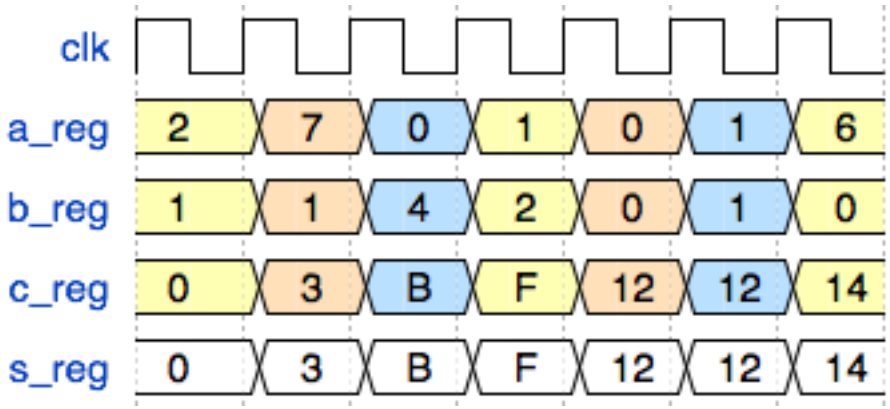

W
X
Y
2ns
+
ck
2ns
+
S
segmentation
W
X
Y
stage 1
2ns
+
ck
2ns
+
stage 2
S
Warning: possible hold violation
This is necessary!

Example III

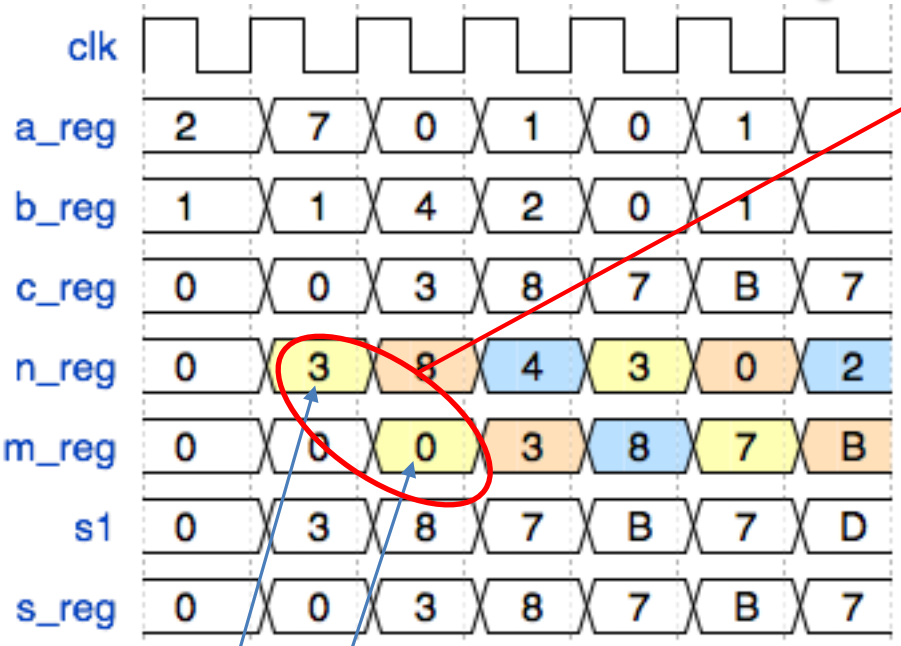

a_reg
b_reg
c_reg
+
+
s_reg
segmentation?
a_reg
b_reg
c_reg
+
n_reg
m_reg
+
s_reg

Example III

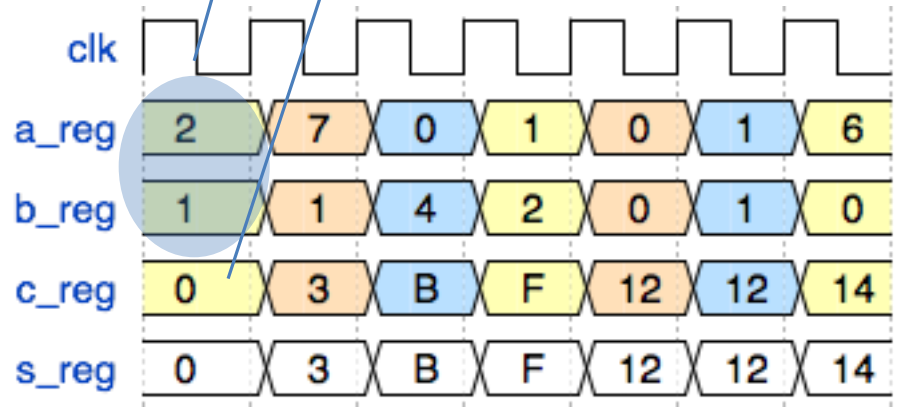

a_reg
b_reg
c_reg
+
+
s_reg


clk
a_reg 2 7 0 1 0 1 6
b_reg 1 1 4 2 0 1 0
c_reg 0 3 B F 12 12 14
s_reg 0 3 B F 12 12 14

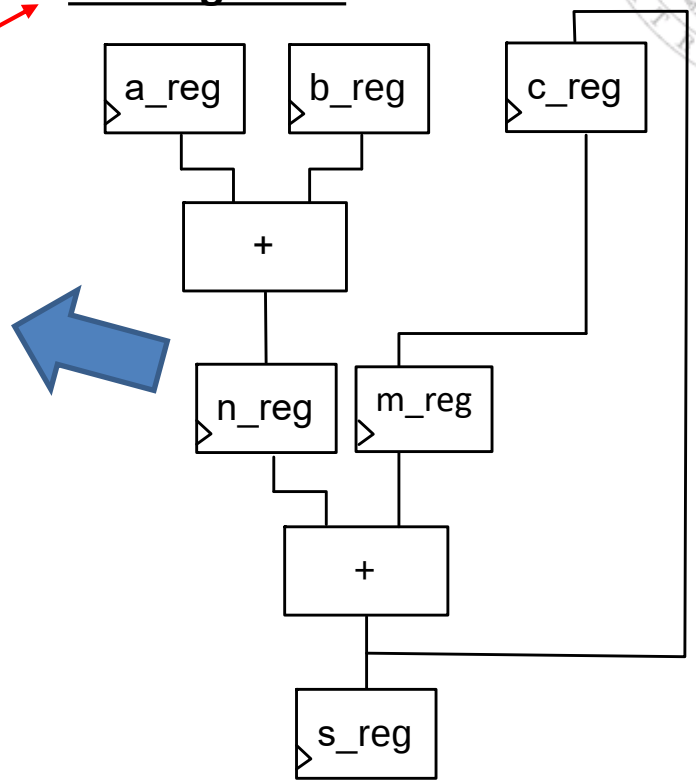
Example III



Chronogram without segmentation

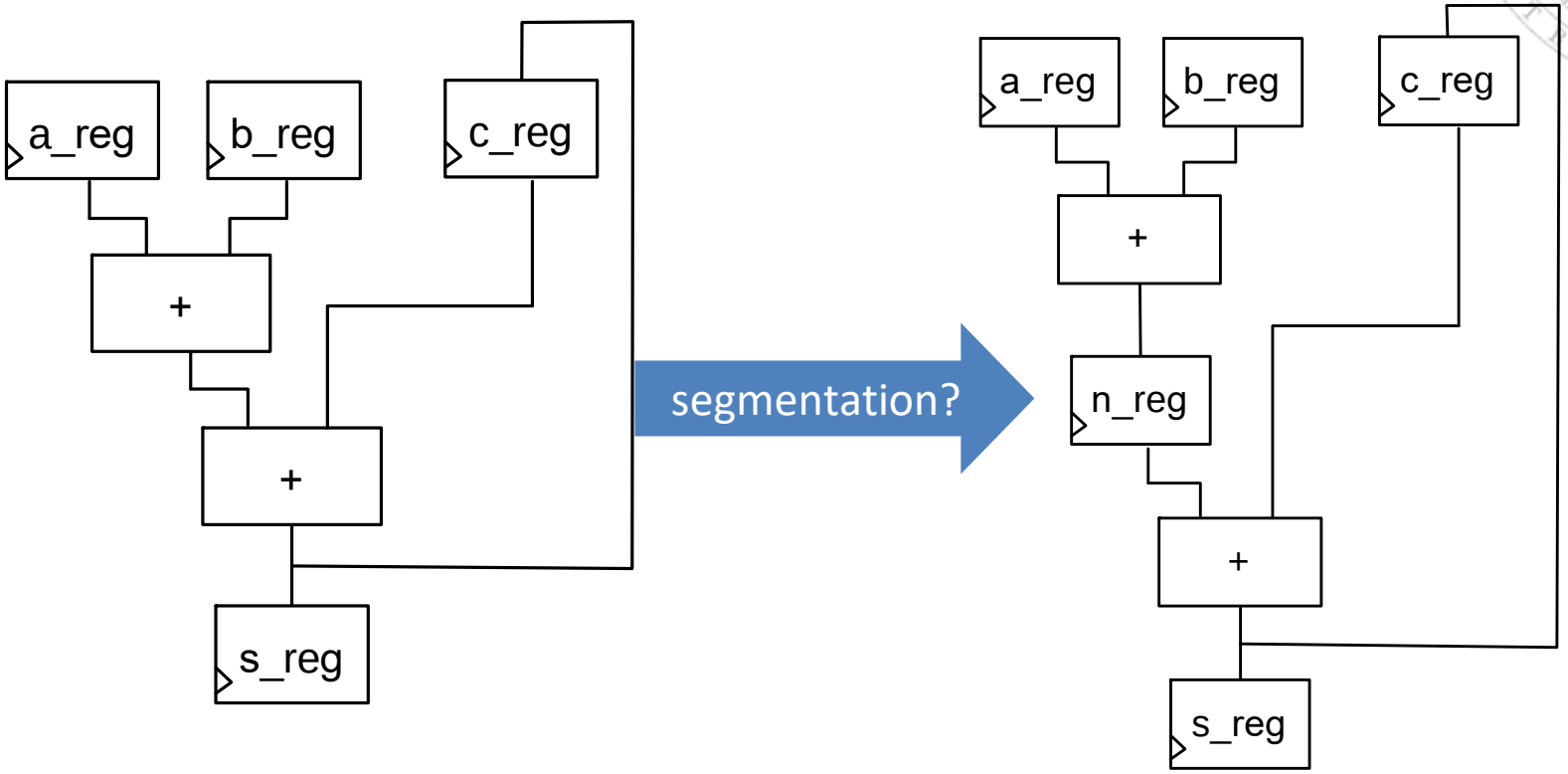


Not aligned!!!

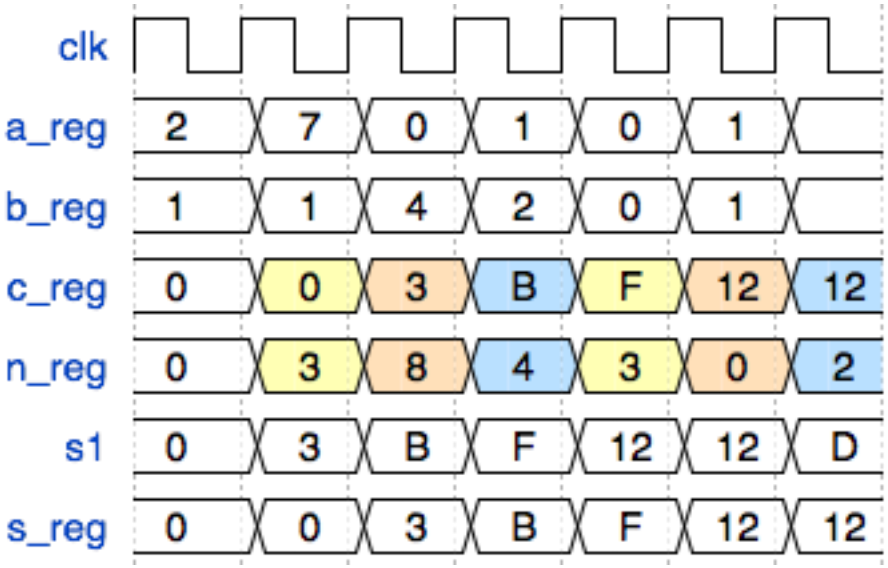


The result is not correct

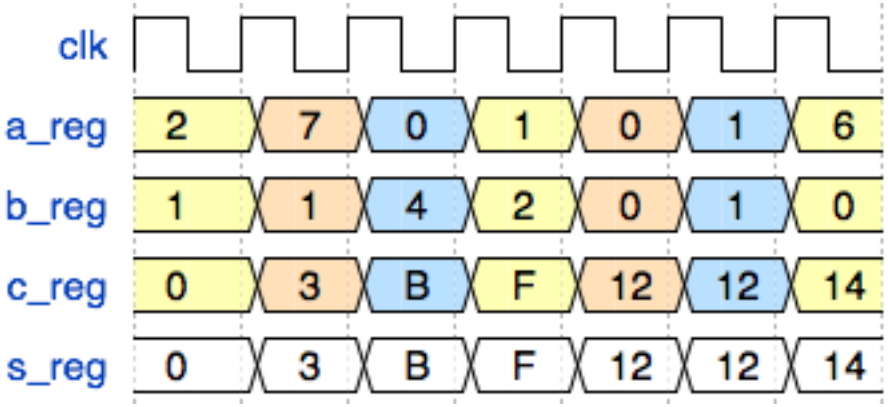
Example III

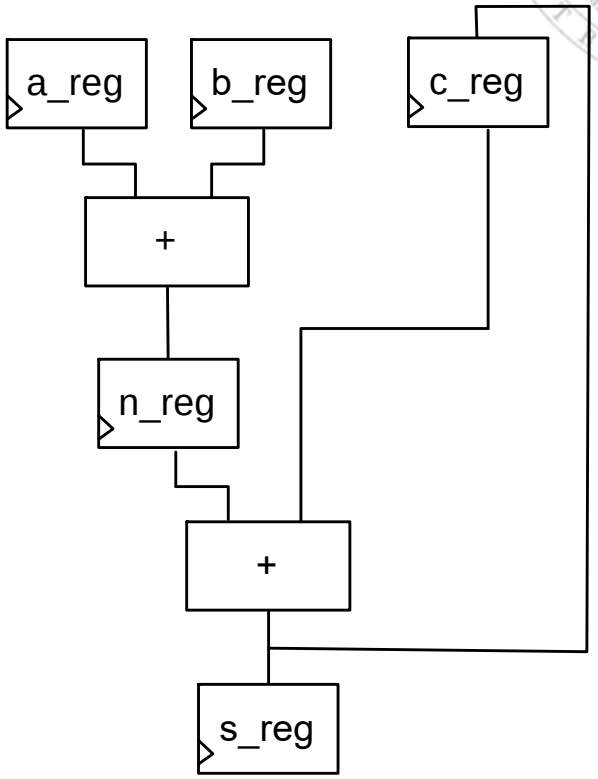


Example III



Chronogram without segmentation

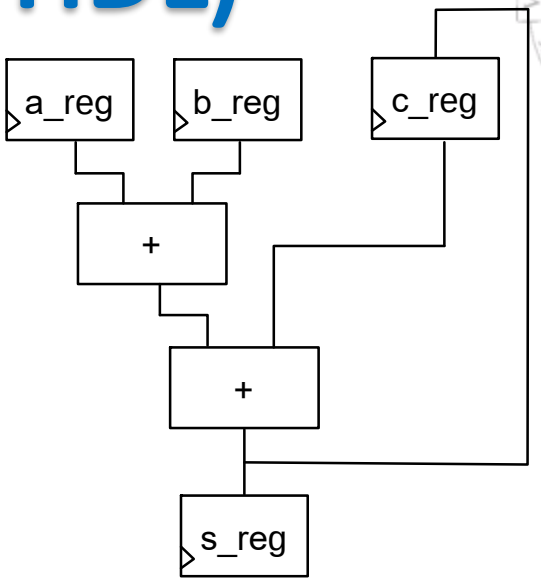




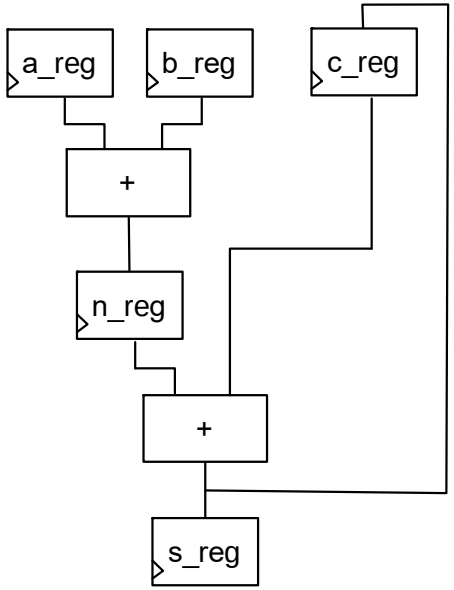
Correct

Example III (VHDL)

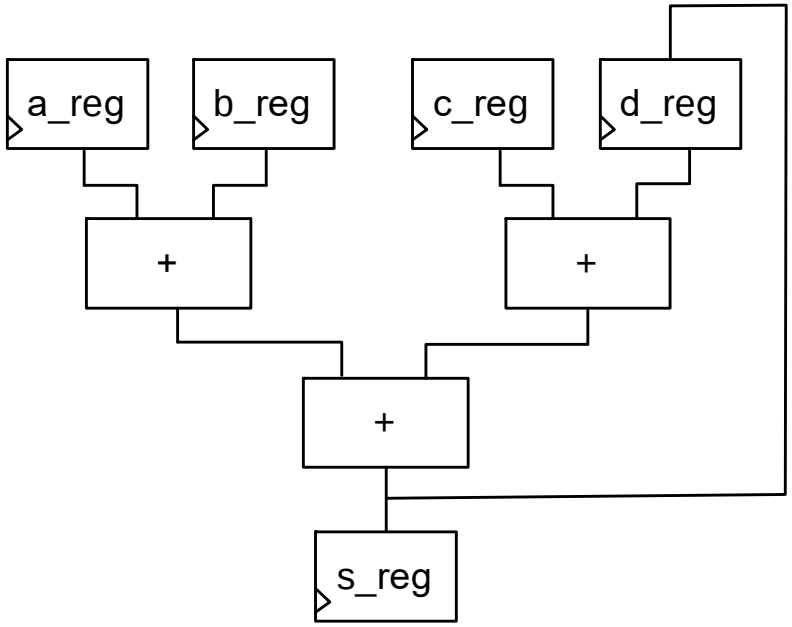
```
p_reg : process (clk, rst) begin
  if rst = '1' then
    s <= (others => '0');
    c <= (others => '0');
  elsif rising_edge(clk) then
    s <= a + b + c;
    c <= a + b + c;
  end if;
end process p_reg;
```

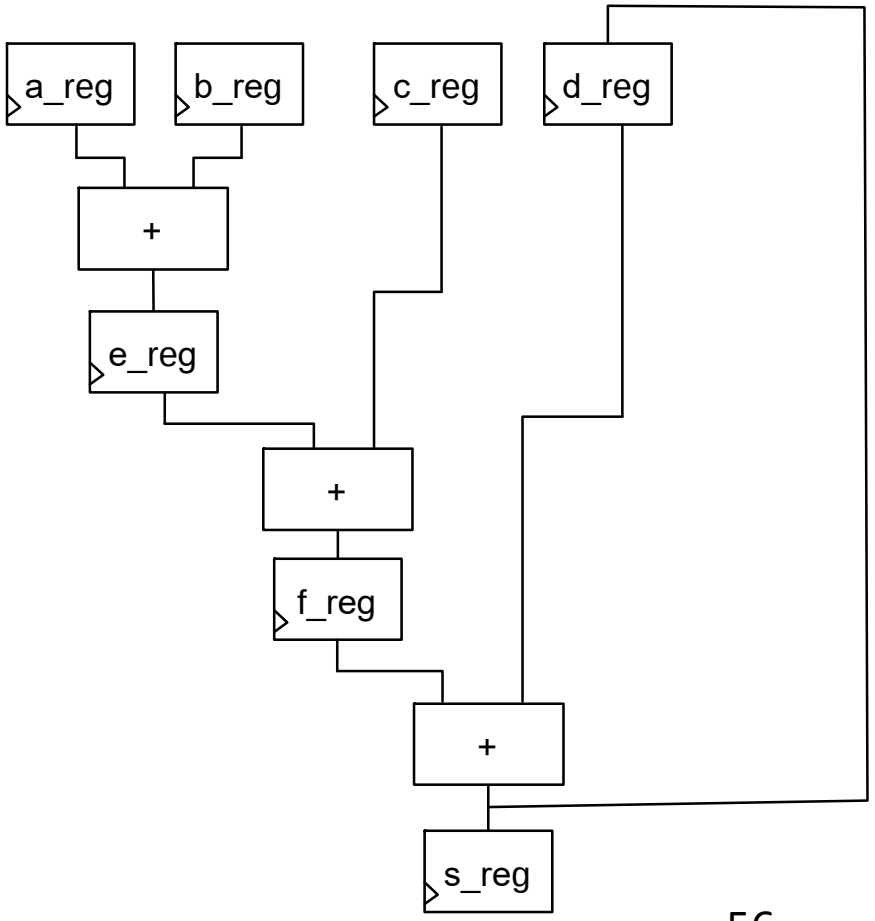



```
p_s_reg : process (clk, rst) begin
  if rst = '1' then
    c <= (others => '0');
    s <= (others => '0');
    n <= (others => '0');
  elsif rising_edge(clk) then
    c <= c + n;
    n <= a + b;
    s <= n + c;
  end if;
end process p_s_reg;
```



Example IV





Outline

1. Metrics
2. Static Time Analysis (STA)
3. Segmentation
4. Dynamic behavior
5. Area analysis
6. Power consumption

Introduction

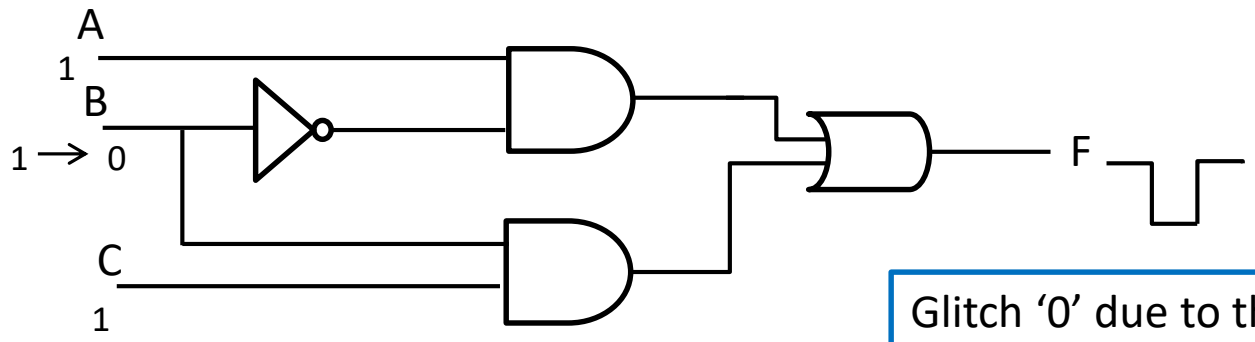
- Logic simulation does not take into account any delays whatsoever.
- Delays are taken into account in the *Post-Place & Route* simulation.
  - The synthesis tool estimates the delay of all the paths.
  - The greater one, named *critical path*, is the one that determines the delay of the circuit.
- The delay of the different circuit paths can cause functional errors.
  - Hazards, glitches and false critical paths.

Hazards and glitches

- The propagation delay of a circuit is defined as the time needed to obtain a valid output value from a valid input value.
- **Hazards** are unexpected fluctuations in the outputs before reaching their final value.
  - Static hazards.
  - Dynamic hazards.
- These fluctuations can be one or several unwanted pulses that are named **glitches.**

Static hazards

- When one input variable changes, the output changes (a 'glitch') momentarily before stabilizing to the correct value
  - Static-1 hazard:
  - Static-0 hazard:
- They are due to the existence of different paths that converge towards the output with different delays.

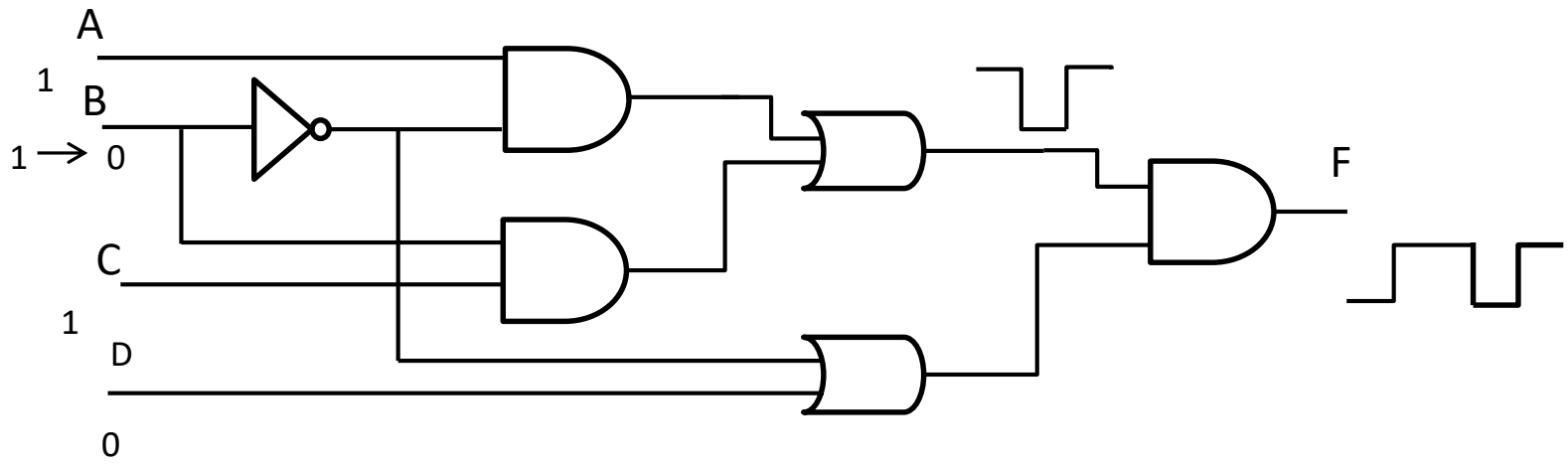


There is a static hazard if a glitch occurs when the output shouldn't have changed at all

Glitch '0' due to the inverter. Its duration will be the same as the inverter delay

Dynamic hazards

- The output should change, but such a transition is not clean; there are intermediate glitches:
  - From '0' to '1'.
  - From '1' to '0'.
- They usually exist in circuits with static hazards and one additional level of logic gates.
  - Note that, if all static hazards have been eliminated from a circuit, then dynamic hazards cannot occur.



Outline

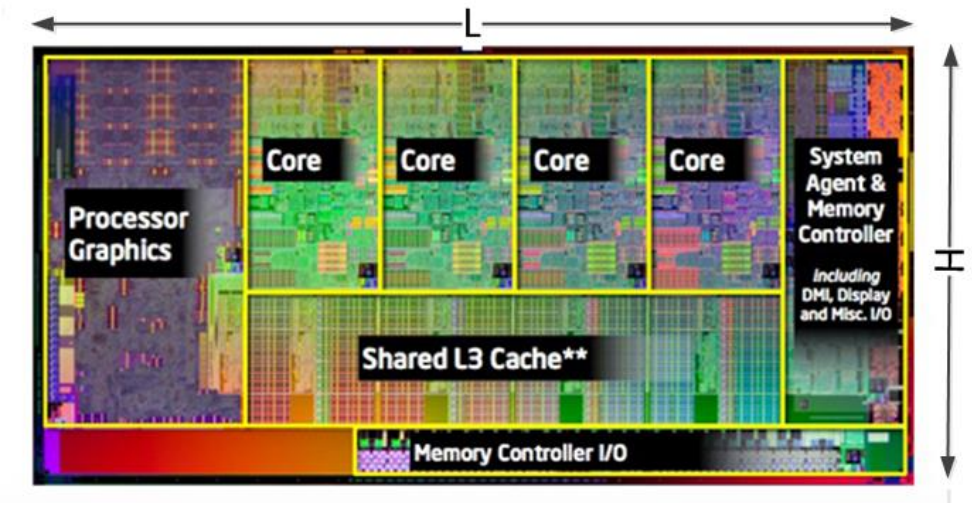
1. Metrics
2. Static Time Analysis (STA)
3. Segmentation
4. Dynamic behavior
5. Area analysis
6. Power consumption

Objective

- Silicon area used by the circuit
- How can it be measured? it depends on the technology used:
  - ASIC (Application-Specific Integrated Circuit)
  - FPGA (Field-Programmable Gate Array)
  - …

ASIC

- Depends on:
  - Floorplan ⟹ L y H.
  - Fabrication technology.
- Area of the logic plus interconnections
- ∄ layout ⟹ Estimation
  - Logic area= ∑ (moduli area)
  - Interconnection area.



FPGA

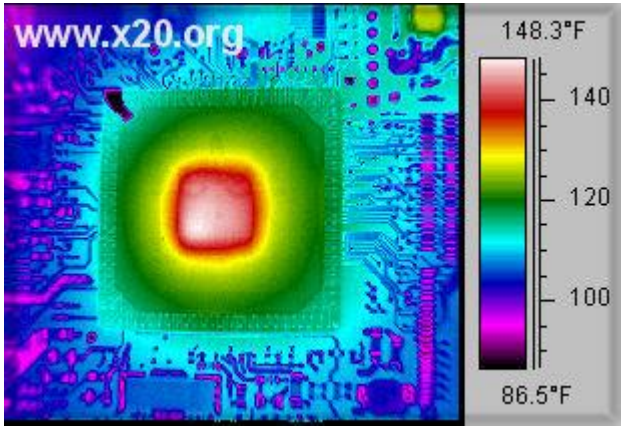
- The FPGA has a fixed size!
- However, the smaller the design, the more circuits can be synthetized in the FPGA
- Logic area:
  - Measured in FF and LUT, slices or CLBs (4 slices)

Outline

1. Metrics
2. Static Time Analysis (STA)
3. Segmentation
4. Dynamic behavior
5. Area analysis
6. Power consumption

Metrics

- Different metrics
  - Power
    - Temperature
  - Energy
    - Battery lifetime



Static and dynamic power consumption

- Static power consumption.
  - Power consumption of the system when there are no transitions in the input signals.
- Dynamic power consumption.
  - Power consumption of the system when there are transitions in the input signals.

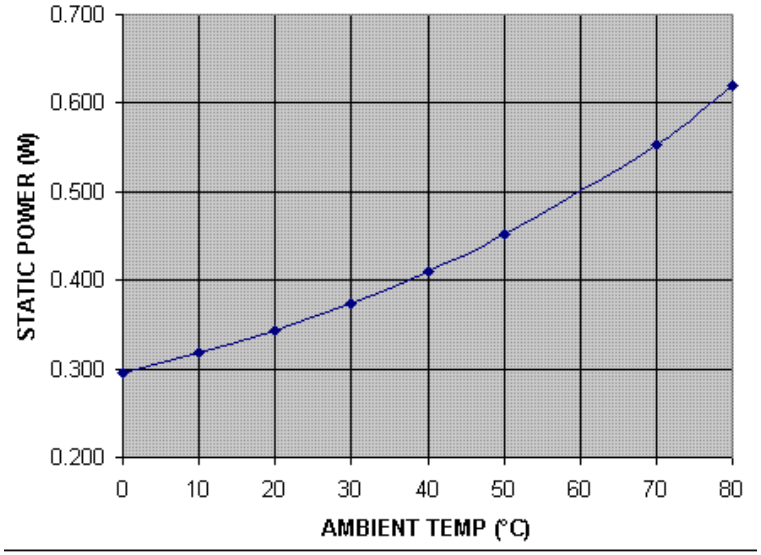
Static consumption

$$P_{static} = V_{dd} \cdot I_{static}$$

- Where:
  - I_{static}: static current.
  - V_{dd}: voltage supply.

Static consumption: dependencies

- I_{static} depends on:
  - Transistors leakage current (technology dependent).
  - Voltage supply ($\mu\ V^3$)
    - Xilinx Virtex-5 nominal voltage: 1.0 V ±5% ⟹ variation ±15% in the static power consumption.
  - Temperature.



Dynamic consumption

$$P = \frac{1}{2} \cdot C_{load} \cdot V_{dd}^2 \cdot f \cdot t$$

- Where:
  - C_{load}: load capacitance.
  - V_{dd}: voltage supply.
  - f: clock frequency.
  - t: activity rate of the gate.

Dynamic consumption

- How to reduce it in our designs?
  - Reducing *f* (however, the timing constraints must still be met).
  - Reducing *t*. Avoid unnecessary transitions.
- More efficient. Reducing V_{dd}. However, this cannot be done in the lab, and…

$$f \approx K \times \frac{(V_{DD} - V_{TH})^a}{V_{DD}}$$ Sakurai's model