EE-334
Digital System Design

Custom Digital Circuits

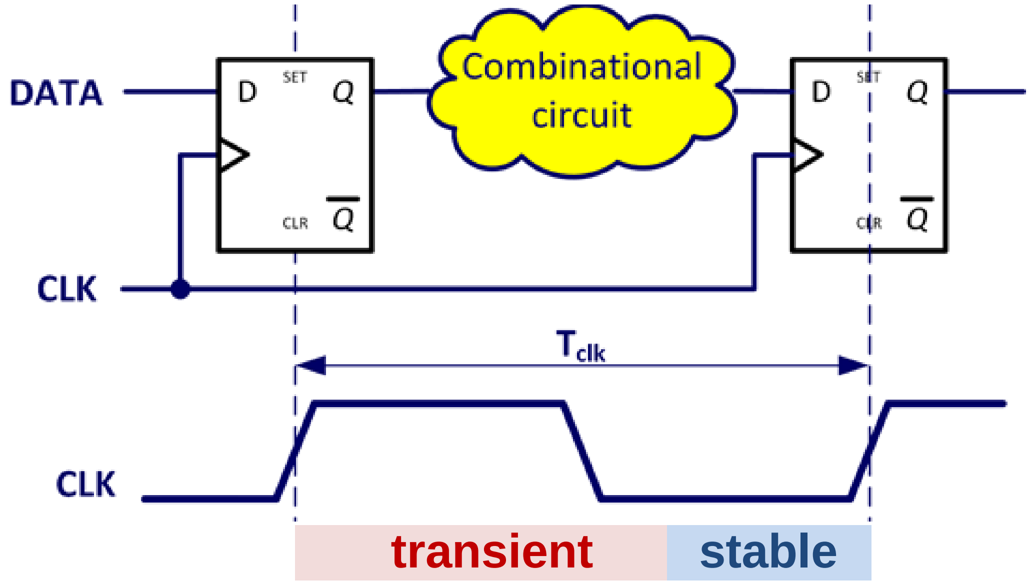
Timing Analysis and Constraints in Synchronous Circuits

Andreas Burg

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Reminder: Principle of Synchronous RTL Design

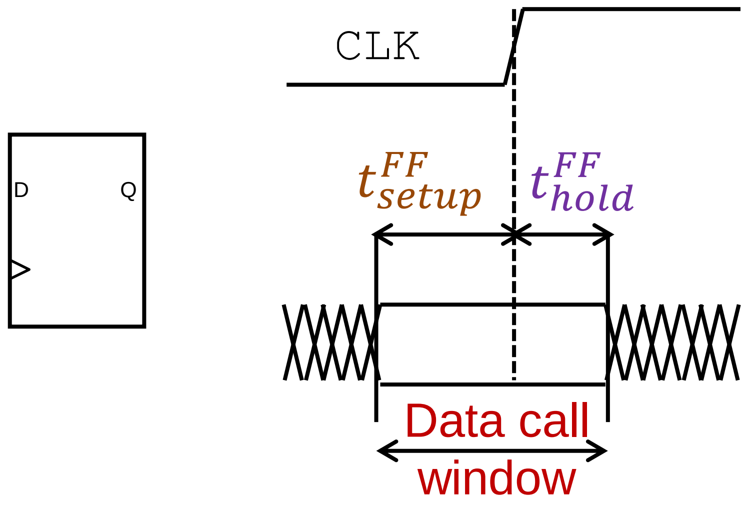
- Synchronous circuits are composed of
  - Registers that store the current state and
  - Combinational logic that defines the next state
- A **single** global **clock** signal **triggers all registers** at the same time
  - **Registers capture** all **outputs** of the combinational logic at the same time **when they are stable**
  - Inputs of all combinational circuits change at the same time
- **After the positive clock edge**
  - **the circuit enters a transient state** and
  - **stabilizes again after some time**, ready for the next clock edge



How can we verify that data is always captured safely?

Register Timing Requirements

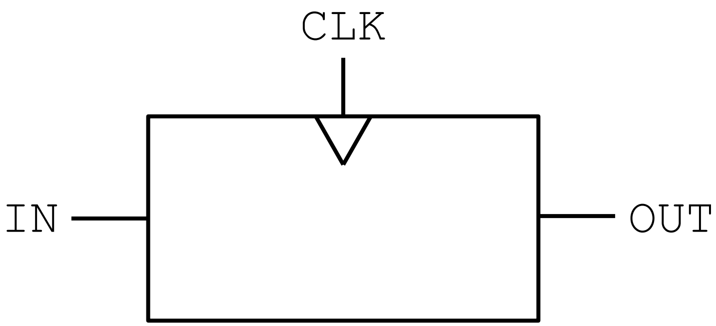
- **Requirements for** safely **capturing the next state**/data are **set by the registers**
- **Input must be stable during the data call window** around the active clock edge
- Formal register **timing requirements**:
  - Setup time t_{setup}^{FF}: time the data must be stable before the active clock edge
  - Hold time t_{hold}^{FF}: time the data must remain stable after the active clock edge

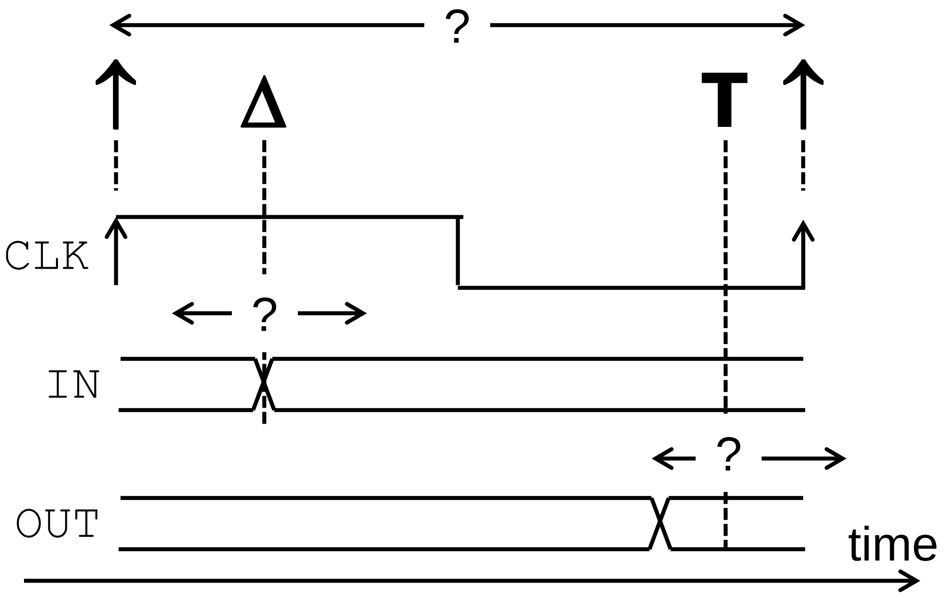


Static Timing Analysis (STA)

- **Static Timing Analysis** characterizes a digital circuit in terms of its timing parameters and **verifies the timing requirements** of a circuit and its registers

- Questions answered by static timing analysis
  - **How fast can we clock a circuit?**
  - **What is the time window in which we can** safely
    - **apply signals** to the input? (stimuli application)
    - **sample the output** of a circuit? (response acquisition)

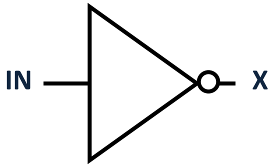


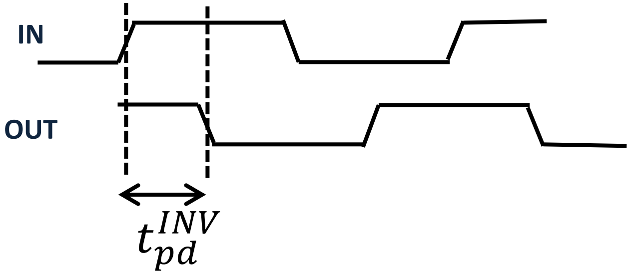


Δ . . . Stimuli application
T . . . Response acquisition
↑ . . . Active clock edge

Modelling the Delay of Combinational Circuits

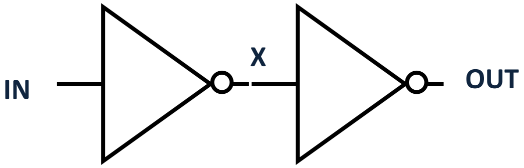
- Combinational Circuits are composed of combinational logic gates
  - **Logic gates have** a **propagation delay**

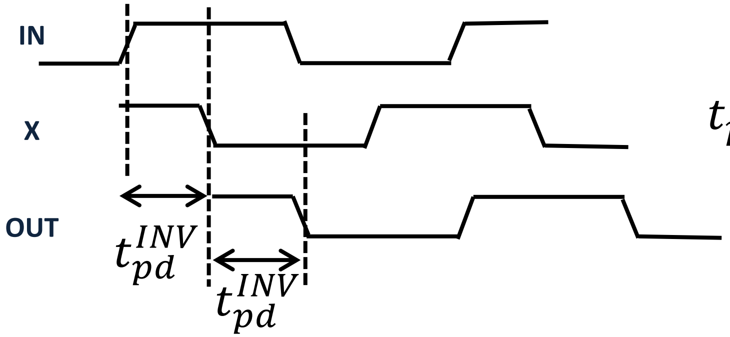




$$t_{pd}(IN \rightarrow x) = t_{pd}^{INV}$$

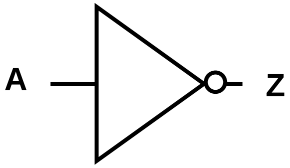
  - **Delays** of subsequent gates **add up**

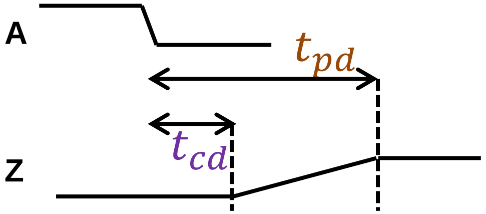




$$t_{pd}(IN \rightarrow x) = 2 \cdot t_{pd}^{INV}$$

  - To abstract transient effects it is convenient to **define a minimum and a maximum delay**
    - **Contamination delay (min): t_{cd}**
    - **Propagation delay (max): t_{pd}**

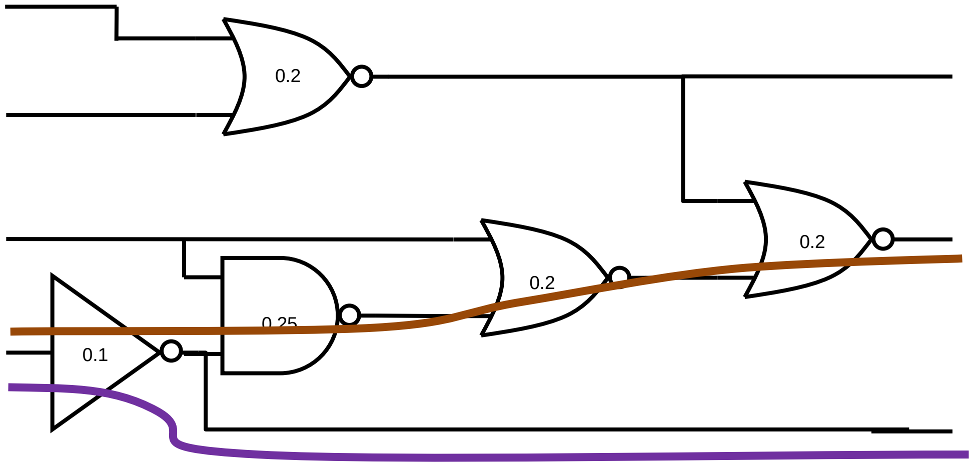




Characterizing the Delay of Combinational Circuits

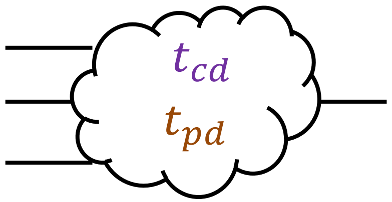
- **Combinational circuits have multiple inputs and multiple outputs**
  - Different inputs may cause changes in the same output
  - The same input may cause changes in different outputs

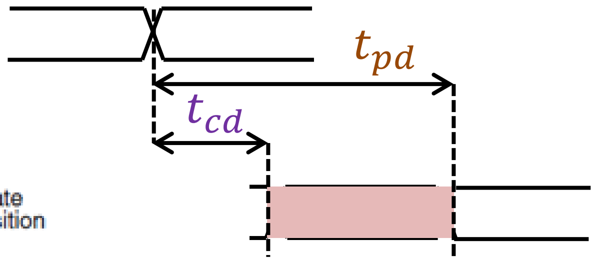
- **Consider all paths from all inputs to all outputs and define**
  - **Contamination delay**: shortest path t_{cd}
    - Based on contimination delays of the gates
    - Earliest change of an output after a change on any input
  - **Propagation delay**: longest path t_{pd}
    - Based on propagation delays of the gates
    - Latest change of an output after a change on any input



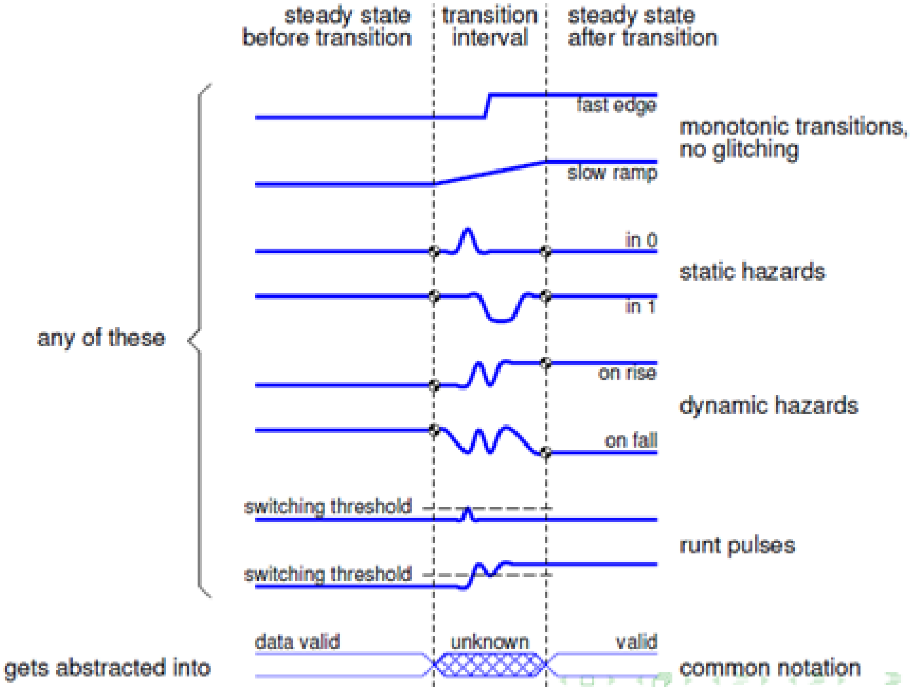
Delays of Complex Gates and Sub-Circuits

- **Contamination delay and propagation delay determine** the time **when the output** of a combinational circuit **is** guaranteed to be **stable**

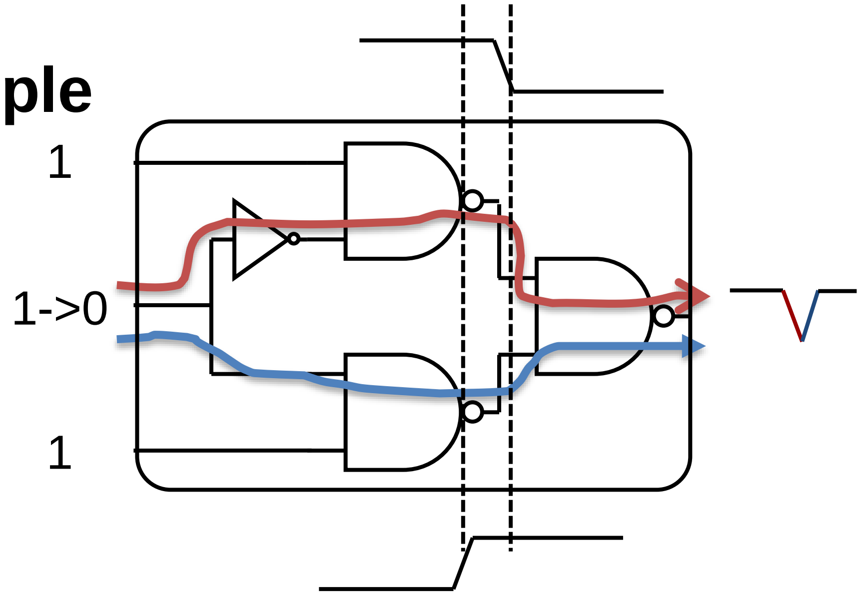




  - **Transition window** (**transient** time between contamination and propagation delay)**:**
    - **Outputs do not match Boolean equations**
    - **Must assume outputs are unknown**

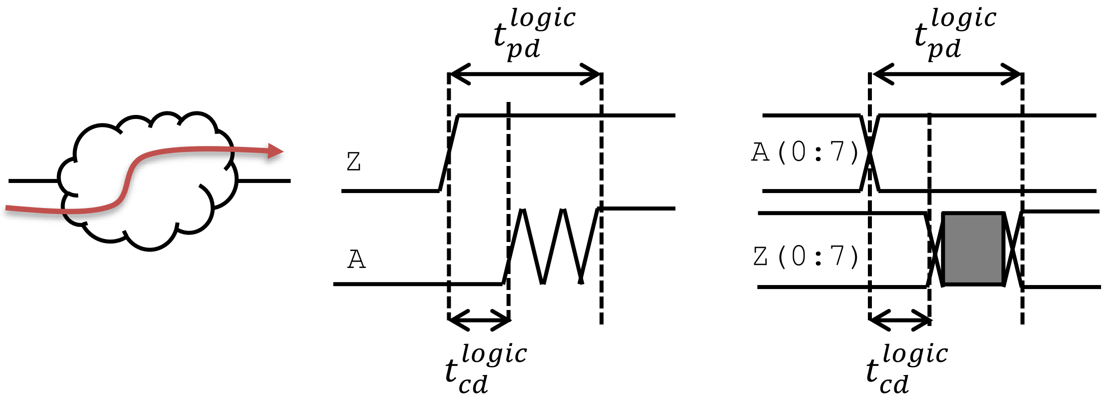


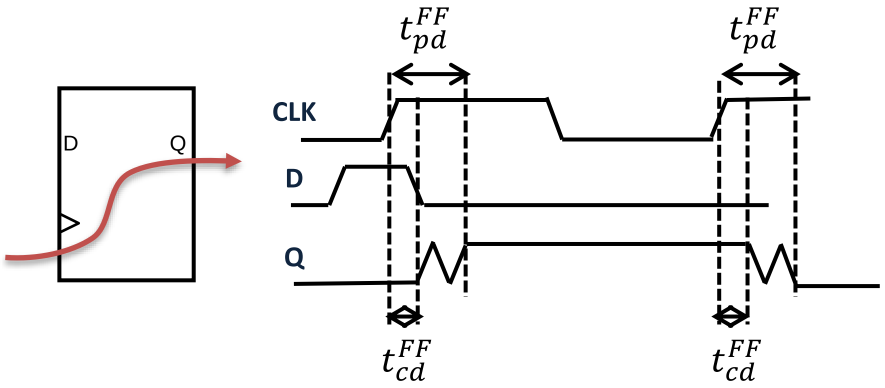
- **Example**



Static Timing Analysis

- STA is based on **delays of combinational logic and sequential elements**





Combinational logic

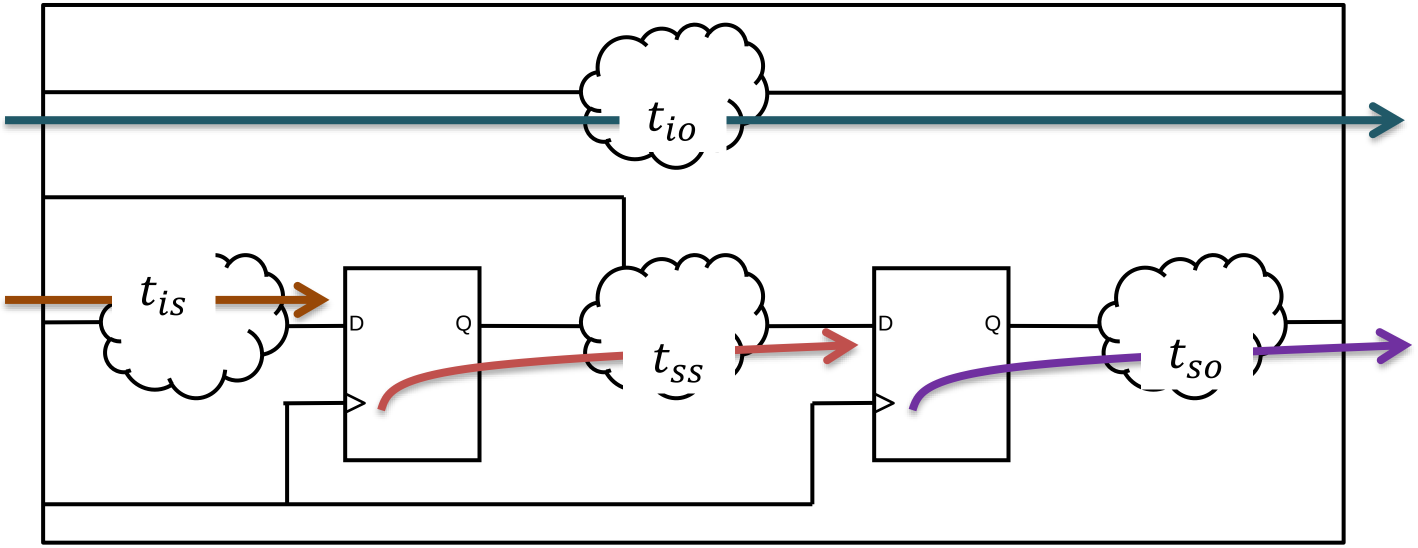
- Timing arc from input to output
  - Propagation delay t_{pd}^{logic}
  - Contamination delay t_{cd}^{logic}
  - Entire circuit (multiple paths)
  - Critical path $t_{crit}^{logic} = max\left(t_{pd}^{logic}\right)$

Sequential elements

- Timing arc from clock to output (between clock and data)
  - Propagation delay t_{pd}^{FF}
  - Contamination delay t_{cd}^{FF}

Static Timing Analysis: Path Groups

- STA considers **four path groups**, **defined by the type of start- and end-points**
  - **Input-to-Register** (t_{is}): **from** a primary **input to** a **register data input**
  - **Register-to-Register** (t_{ss}): **from clock input of a register to** the **data input of a register**
  - **Register-to-Output** (t_{so}): **from clock input of a register to** a primary **output**
  - **Input-to-Output** (t_{io}): direct path **from** a primary **input to** a primary **output**



STA: Checking Timing Conditions

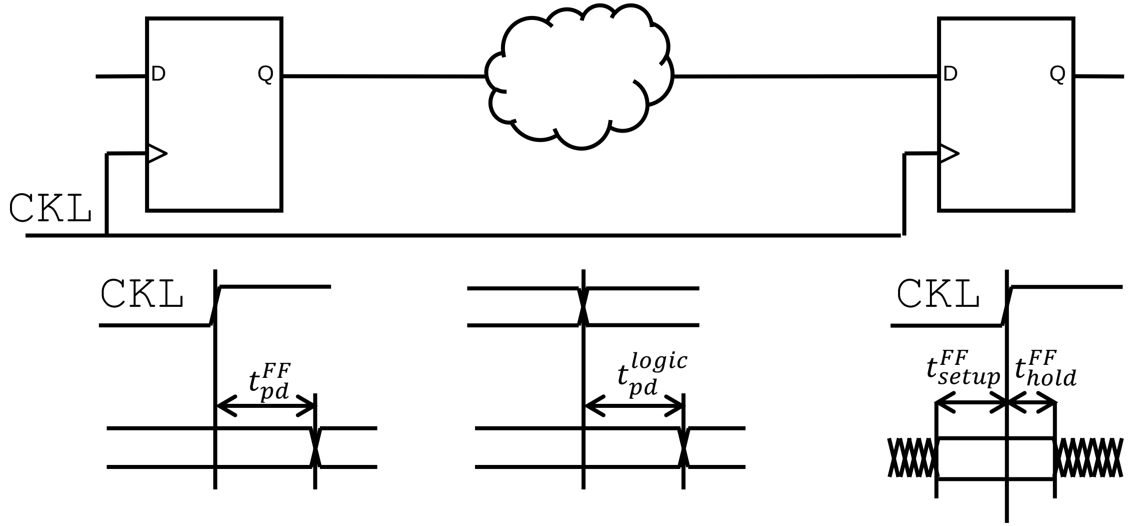
- **Timing requirements** for all **register-to-register** paths
  - Path **starts from the clock pin** of a register and **ends on** the **data pin** of a register
  - **Setup condition**: latest arrival

$$\underbrace{t_{pd}^{FF} + t_{pd}^{logic}}_{t_{ss}^{max}} < \boldsymbol{t_{clk}} - t_{setup}^{FF}$$

Determines maximum frequency

  - Hold condition: earliest arrival

$$\underbrace{t_{cd}^{FF} + t_{cd}^{logic}}_{t_{ss}^{min}} > t_{hold}^{FF}$$



Critical Path

- The **critical path** is the path within a circuit that **limits** its **clock frequency**

- The critical path contains
  - For reg->reg: the Clk->Q delay delay of the register from which it starts
  - The delay of the subsequent logic

- The critical path does not include:
  - For in->reg: the stimuli application delay (if known, otherwise 0)
  - For reg->out: the output sampling time (if known, otherwise 0)
  - The setup time of the register at the end

STA: Checking Timing Conditions

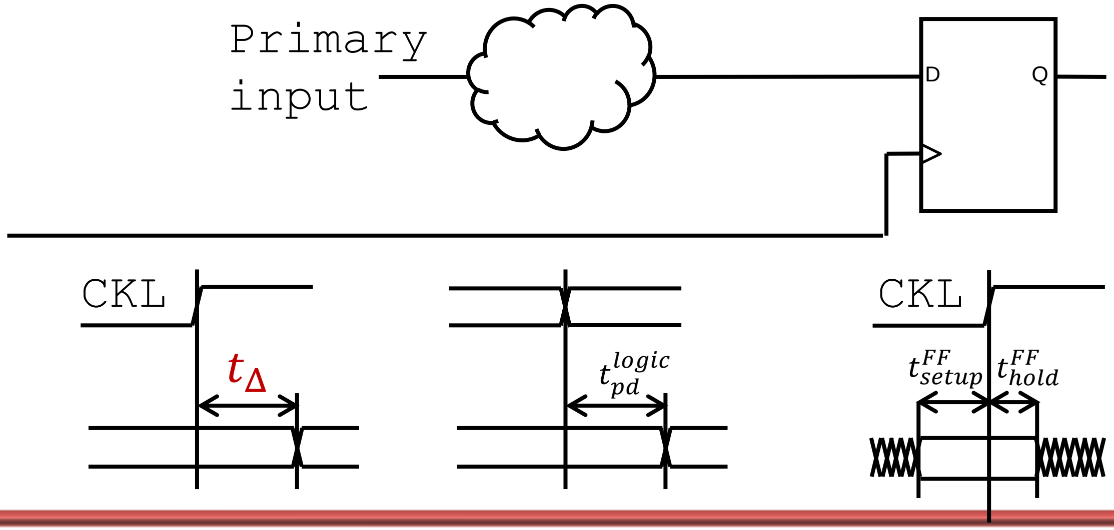
- **Timing requirements** for all **input-to-register** paths
  - Path **starts from primary input** and **ends on the data pin** of a register
  - **Setup condition**: latest arrival

    $$t_{\Delta} + t_{pd}^{logic} < t_{clk} - t_{setup}^{FF}$$

  - **Hold condition**: earliest arrival

    $$t_{\Delta} + t_{cd}^{logic} > t_{hold}^{FF}$$

Determine setup window for primary inputs



STA: Checking Timing Conditions

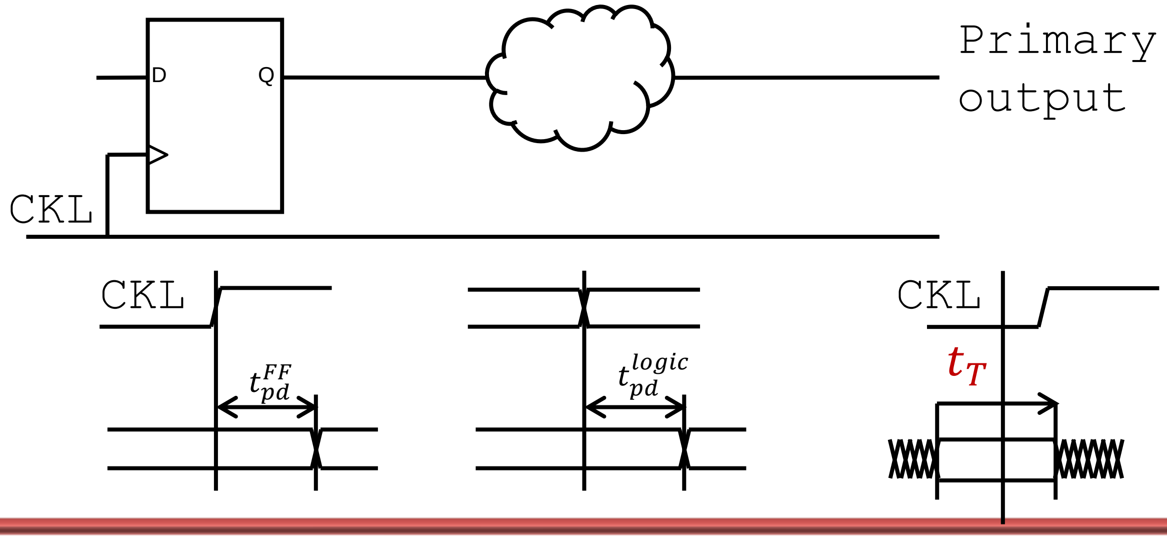
- **Timing requirements** for all **register-to-output** paths
  - Path **starts from data pin of a register** and **ends on the sampling of the output**
  - **Setup condition**: earliest (ideal) sampling

    $$t_{pd}^{FF} + t_{pd}^{logic} < t_T$$

  - **Hold condition**: latest (ideal) sampling

    $$t_{cd}^{FF} + t_{cd}^{logic} > t_T$$

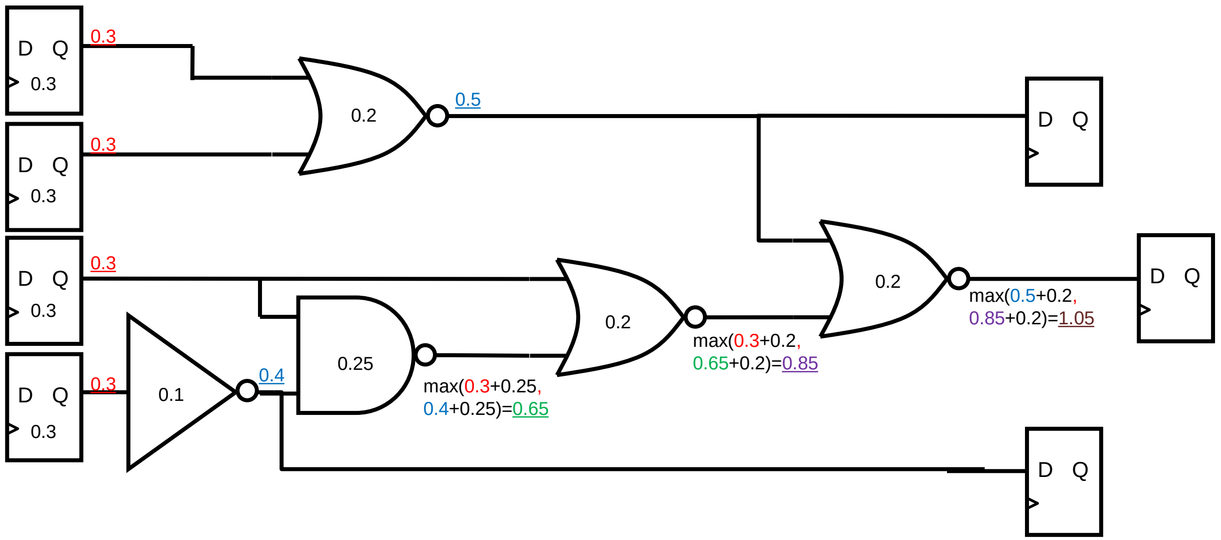
Determine the sampling window for primary outputs



Static Timing Analysis on Gate Level

- The STA procedure identifies
  - the critical (longest) path t_{pd}^{logic} to verify setup
  - the shortest path t_{cd}^{logic} to verify hold

- Identify the shortes and the longest path for the data-input of each Flip-Flop
  - Annotate the circuit with the worst-case arrival times (delay) of the signal
  - Start from the clock input of the FFs
  - Keep only the longest (shortest) path (Add-Compare-Select)



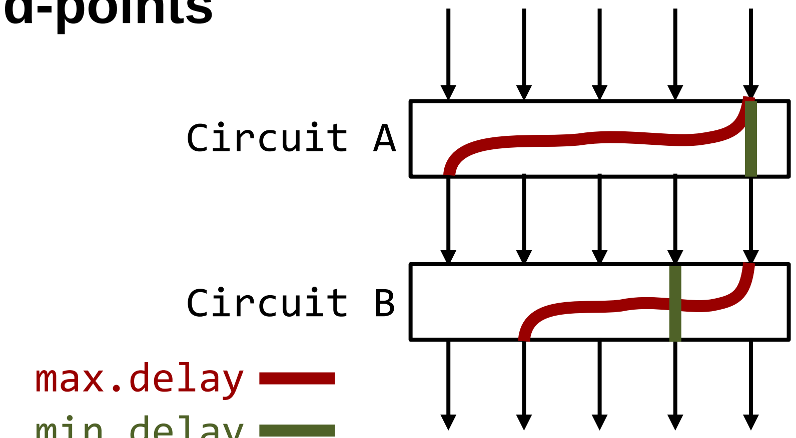
Static Timing Analysis of Datapath Components

- STA of gate level circuits is tedious and therefore carried our by automatic tools

- Designers often want to **have a quick estimate of complex circuits composed of arithmetic components** or other well-studied combinational circuits

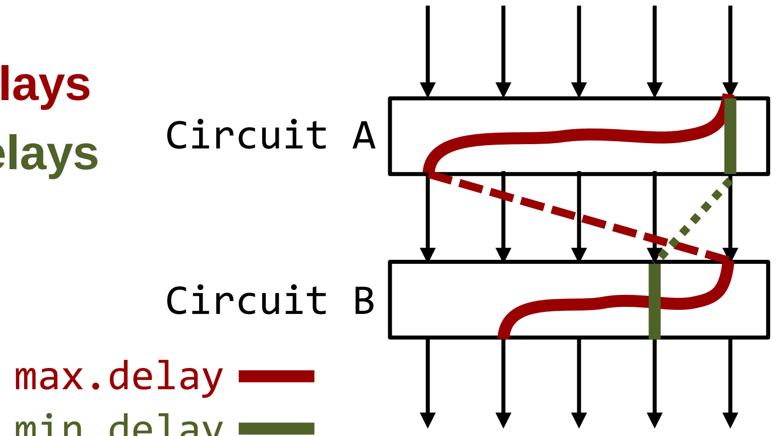
- Combinational sub-circuits have multiple inputs and multiple outputs
  - **Critical paths** of sub-circuits **have different start and end-points**
    - Critical end-point in one sub-circuit is often NOT the critical starting point in the next sub-circuit

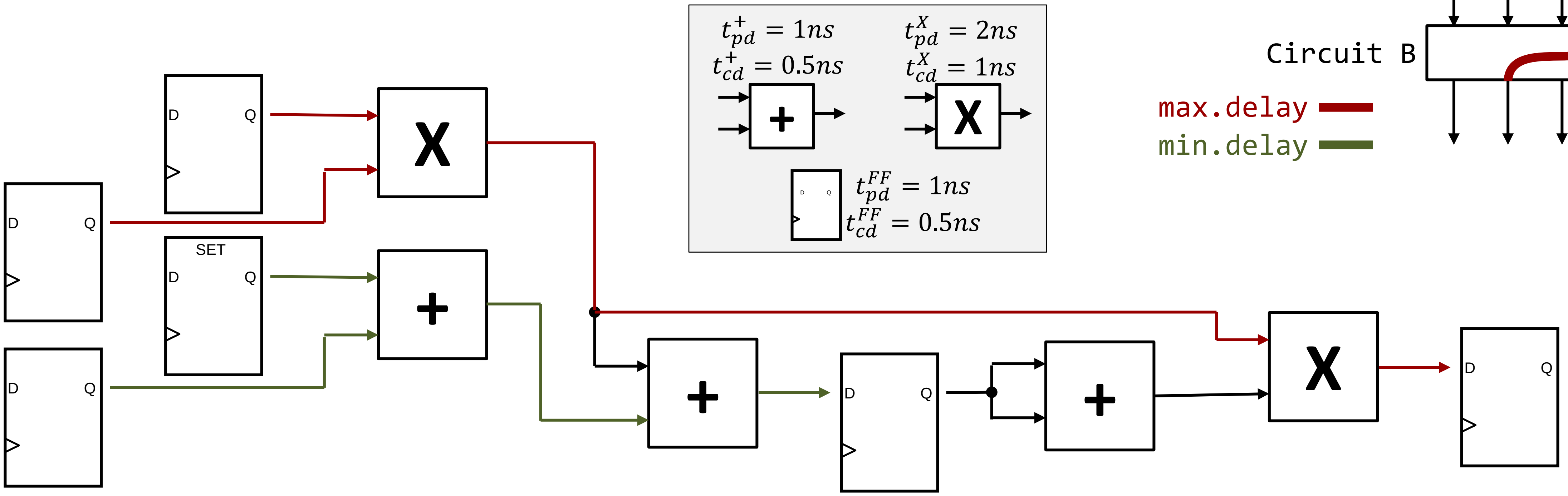
- **Accurate timing analysis is difficult**
  - Needs to be done at the gate level or with complex models of the individual sub-circuits.



Static Timing Analysis of Datapath Components

- In practice, we are often only **interested in a** quick **worst-case estimate**
  - **Model** timing of **each** (combinational) **component with** its **min. and max delay** only
  - **Worst case estimate** obtained by
    - Approximating the **longest path from the** sum of the **maximum delays**
    - Approximating the **shortest path from the** sum of the **minimum delays**





Timing in Hierarchical Designs

- Concatenate two blocks A and B (design entities or chips)
  - Register-to-output path of block A and the input-to-register path of block B merge into a new register-to-register path

$$t_{so}^{max}(A) + t_{is}^{max}(B) < t_{clk} - t_{setup}^{FF}$$

$$t_{so}^{min}(A) + t_{is}^{min}(B) > t_{hold}^{FF}$$