Tampere University

COMP.CE.510
Chip Implementation

Tampere University

Static Timing Analysis (STA)

By. Syed Mohsin Abbas

Assistant Professor (Tenure Track), SoC HUB

Faculty of Information Technology and Communication Sciences

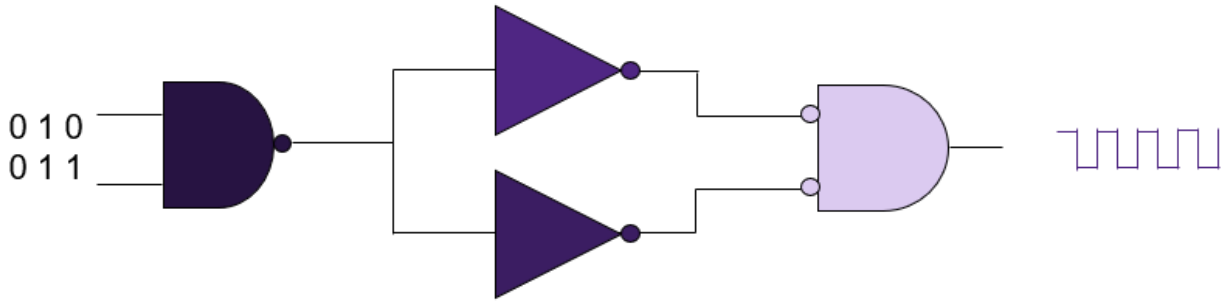
Tampere University, Finland

Static Timing Analysis (STA)

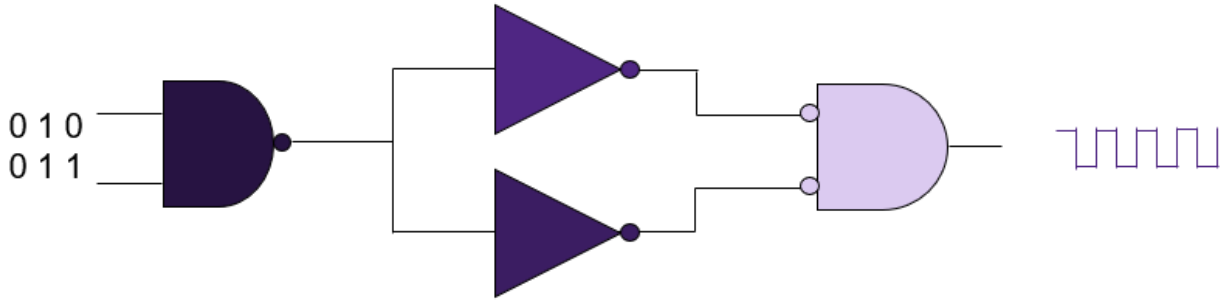
- Static timing analysis (STA) is a method of validating the timing performance of a design by checking all possible paths for timing violations.
  - The timing position of input data signals against clock pulses of the design should be given.
  - From this data and the timing characteristics of the chain it is possible to determine the timing of the arrival of data against clock signal: t_{ar}
  - Depending on the setup and hold times of FFs, it is possible to determine the required time for the data to appear in the inputs of FFs: t_{rq}
  - The amount of time violation is given by slack = $t_{rq} - t_{ar}$
    - For smooth operation, it is necessary that slack >0.

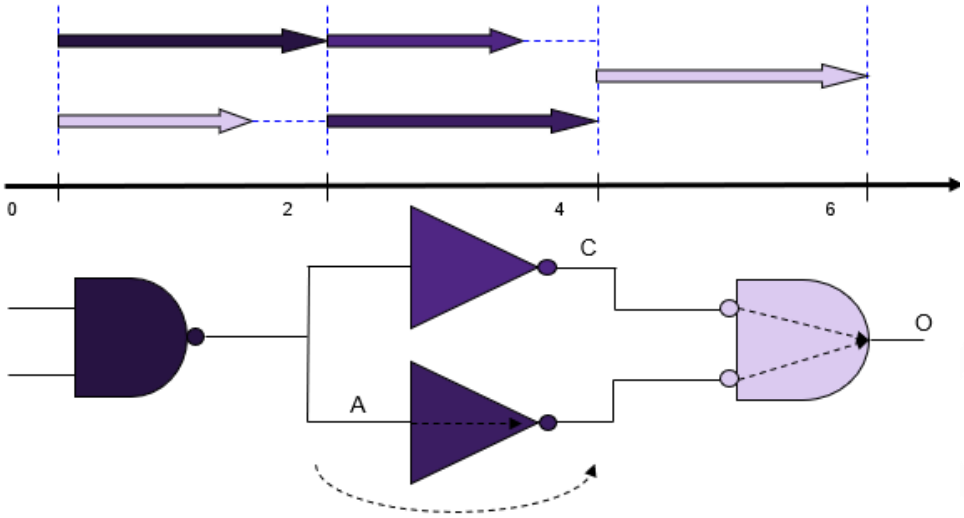
Static Timing Analysis (STA)

- Static Timing Analysis (STA) is the effective methodology for verifying the timing characteristics of a design without the use of test vectors because thousands of test vectors are required to test all timing paths using logic simulation.



- Key principle behind STA is propagating delays through logic gates by simply summing signal arrival times with delay time of the cell and finding maximum of all available arrive times on a net.



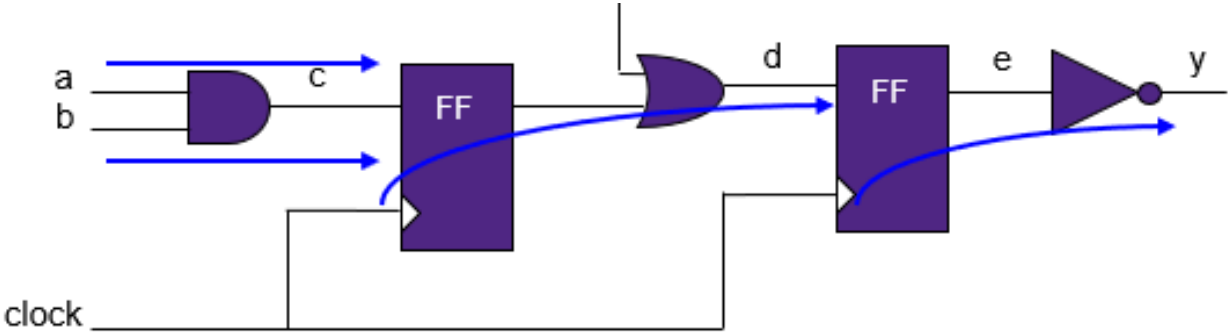


A gate is being processed in STA.

- Arrival times of all inputs are known.
- Delay values at the output are computed using SUM operation that adds the delay at each input with input-to-pin delay.

- Once these candidate delays have been found, the MAX operation is applied to determine the maximum arrival time at the output.
- The arrival time at the input is propagated through the gates at each level till it reaches the output.

STA Steps

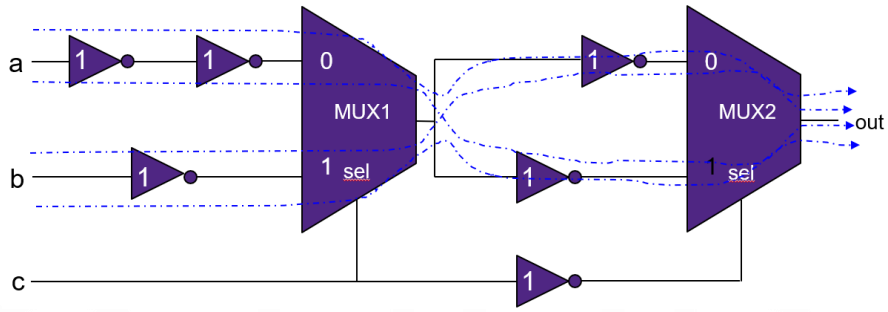


1. Circuit is broken into timing paths
2. Delay of each path is calculated
3. For each path delays are checked against timing constraints

Path	Delay	Constraint
a → c	?	?
b → c	?	?
clock → d	?	?
clock → y	?	?

STA-Timing Paths

- Start point:
  - Input ports
  - Clock pins of flip-flops
- Endpoints:
  - Output ports
  - data input pins of flip-flops



Paths
a → MUX1.0 → MUX2.0 → out
a → MUX1.0 → MUX2.1 → out
b → MUX1.1 → MUX2.0 → out
b → MUX1.1 → MUX2.1 → out

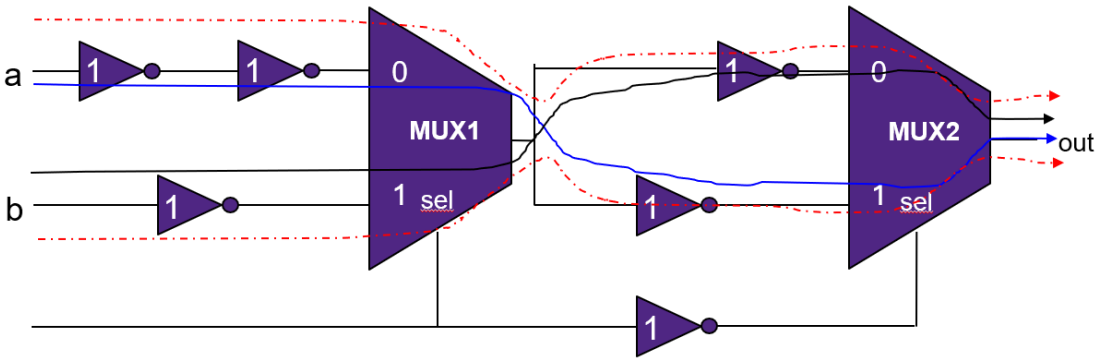
Reference. Vazgen Melikyan, Synopsys University Courseware (Synopsys confidential information)

STA-False Paths

- Paths which physically exist in a design but are not logic paths. **These paths never get synthesized under any input condition**
- In STA ”false path” means a path which user has intentionally set as an ”not-to-be-checked” timing path.

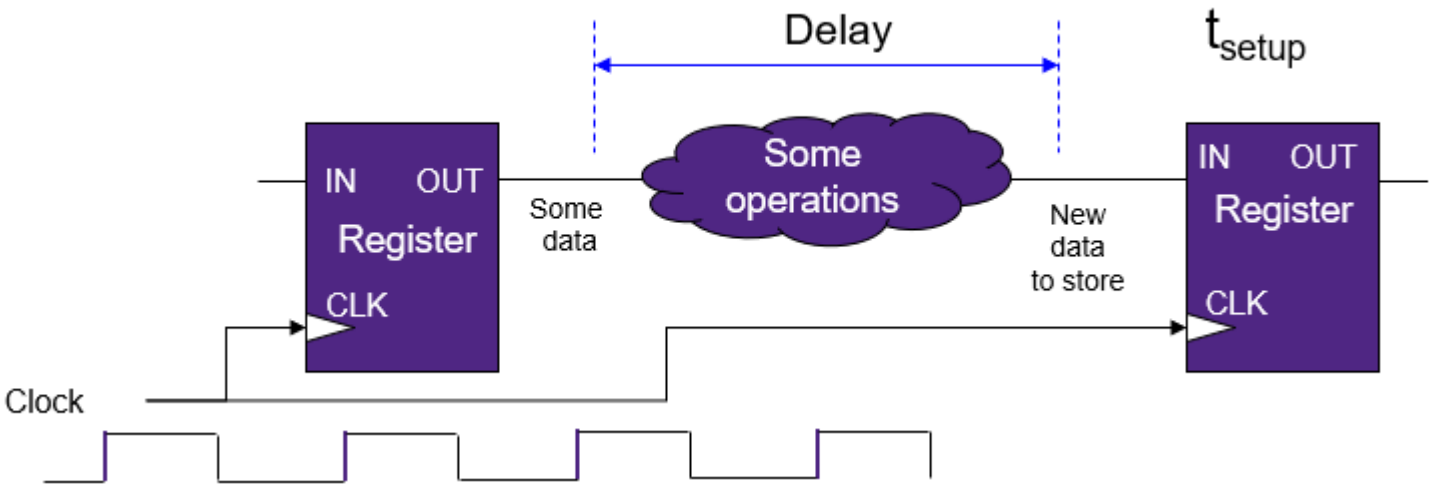
Mux1.0->Mux2.0 & Mux1.1 ->Mux2.1 are checked in STA by default. The reason is that **in GBA (graph based analysis) delay values are propagated, not the logic values.**

In **PBA (path-based-analysis)** actual ”real” paths are considered for timing analysis, and in that mode we would not see those (false) paths.



Paths
~~a → MUX1.0 → MUX2.0 → out~~
a → MUX1.0 → MUX2.1 → out
b → MUX1.1 → MUX2.0 → out
~~b → MUX1.1 → MUX2.1 → out~~

Timing Closure Problem

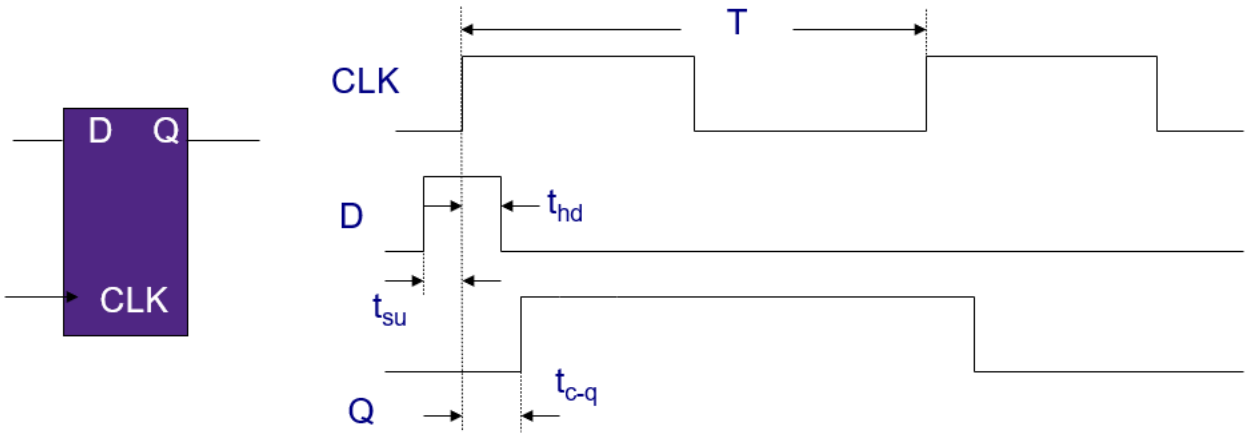


$$Delay + t_{setup} < T_{clock}$$

Reference. Vazgen Melikyan, Synopsys University Courseware (Synopsys confidential information)

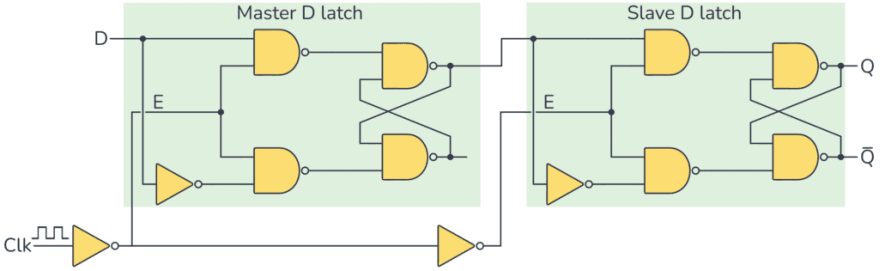
STA: Terminologies

- Timing Diagram of a Flip-Flop



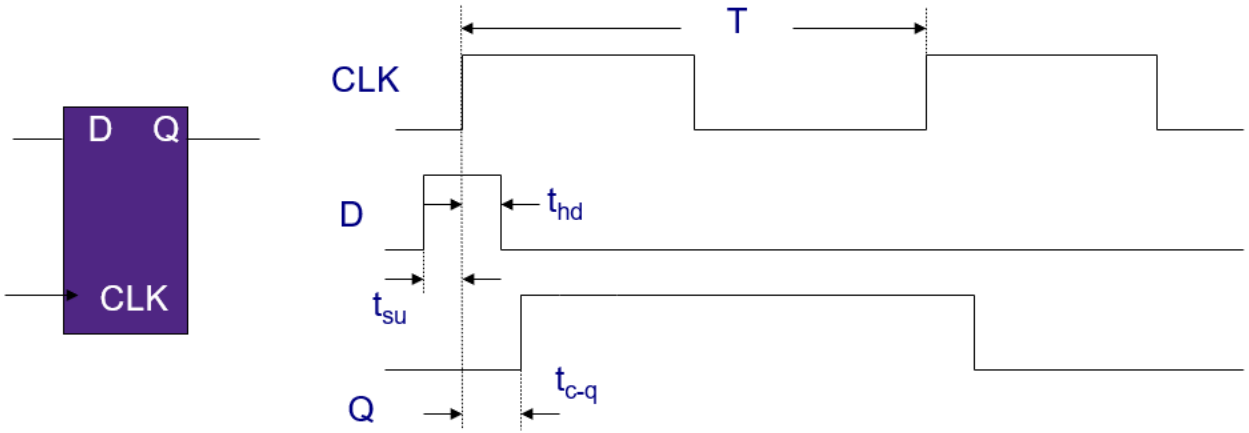
Setup time (t_{su}) is the minimum amount of time before the active clock edge of flip flop, the data input (D) should be held steady.

STA: Terminologie



D Flip-Flop

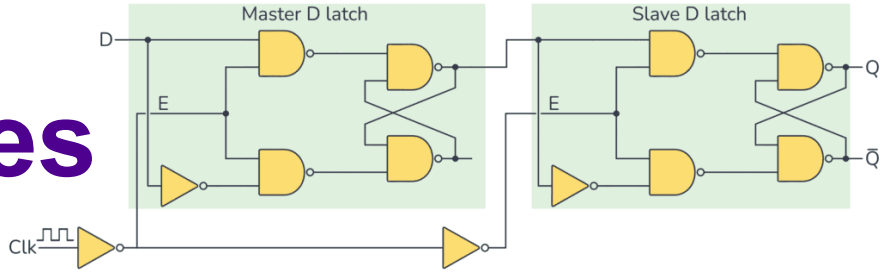
- Timing Diagram of a Flip-Flop



Hold time (t_{hold}) is the minimum amount of time after the active clock edge of flip flop, the data input (D) should be held steady.

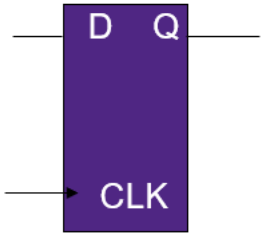
Reference. Vazgen Melikyan, Synopsys University Courseware (Synopsys confidential information)
https://www.build-electronic-circuits.com/d-flip-flop/

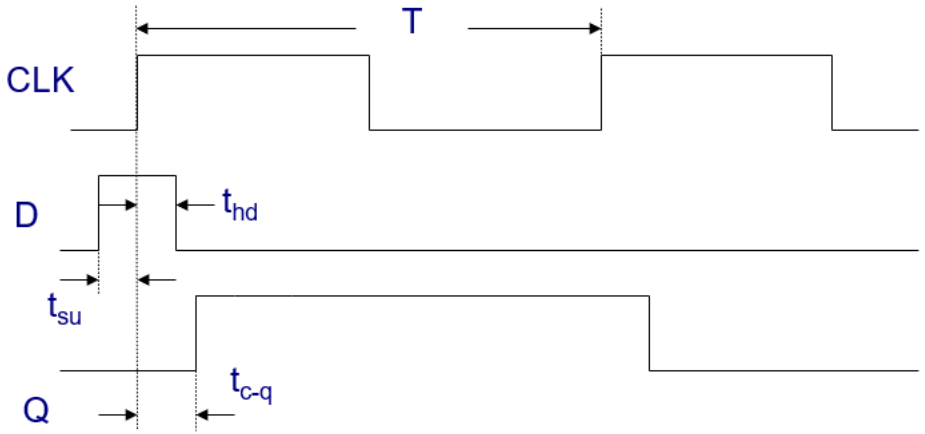
STA. Terminologies



D Flip-Flop

- Timing Diagram of a Flip-Flop





Propagation delay (t_{cq}) is the clock-to-output delay i.e. data input (D) is available at output (Q) after a t_{cq} delay.

Intuitive Example (Setup/Hold)

- Boarding the bus at 8.00 AM
  - Arrival time 7.57 (MUST arrive before departure)
    - Setup time is 3 mins
  - Bus shouldn't move until I settle down
    - Hold time (2 mins ?)

https://www.freepik.com/free-vector/school-bus-cartoon-sticker-white-background_20746940.htm#fromView=keyword&page=1&position=0&uuid=f5196280-1309-4f21-958d-c4aad4398458&query=Bus+Clipart

PDK (Process Design Kit).

PDK : Set of files to model a fabrication process.

- PDK Files
  - Primitive Device Library
    - Symbols
    - Device Parameters
    - Parameterized Cells (PCells)
  - Verification Checks
    - Design Rule Checking (DRC)
    - Layout Versus Schematic (LVS)
    - Antenna and Electrical rule check
    - Physical Extraction
  - Technology data
    - Layers, layer names
    - Process constraints
    - Electrical rules
- Rule files
  - LEF (Library Exchange Format)
  - Tool dependent rule formats
- Simulation models of primitive devices
  - Transistors (typically SPICE)
  - Capacitors
  - Resistors
  - Inductors
- Design Rule Manual
  - A user-friendly representation of the process requirements

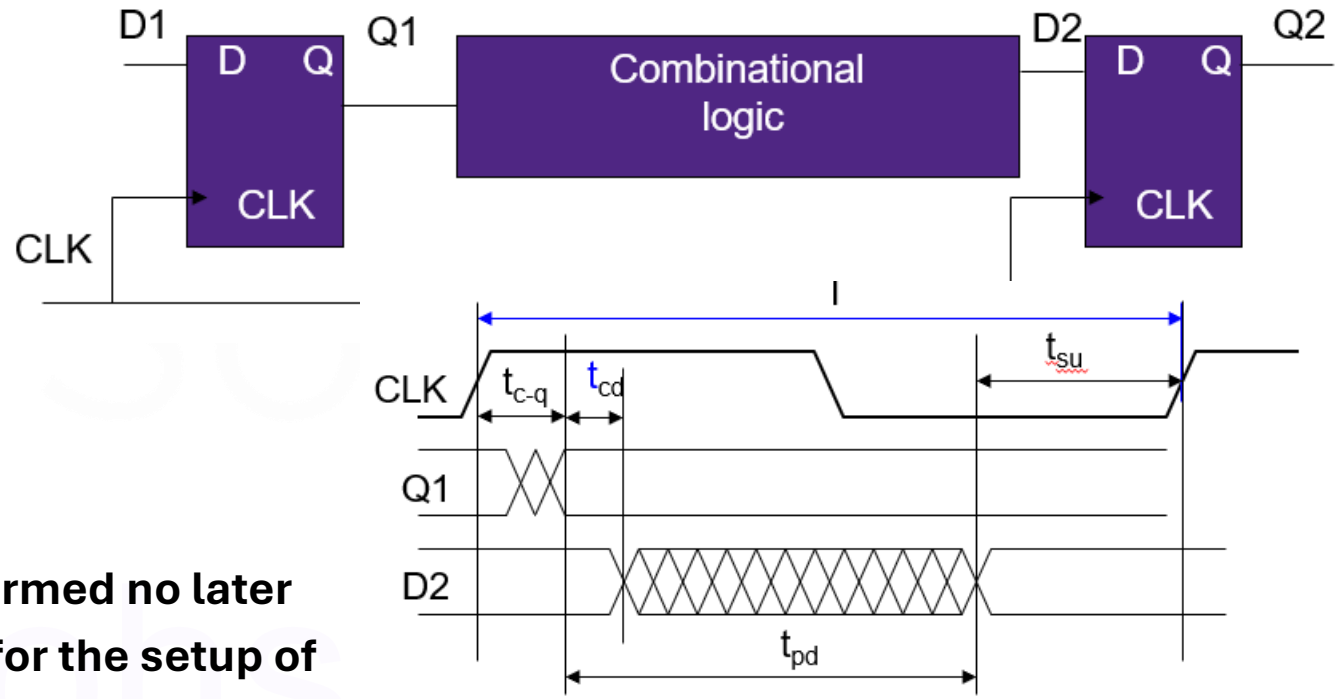
https://www.icdesigntips.com/2020/10/setup-and-hold-time-explained.html,

Liberty File

- Liberty file contains timing related information for all the standard cells and macros
  - Setup/ Hold time information
  - Rise/Fall transitions
  - Slew rate, I/O threshold
  - Area of cell
  - Leakage Power
  - Load Capacitance
- A different liberty file -type is: LVF (Liberty Variance Format).
  - It is used in more modern technology nodes.
- Different liberty files available for different PVT (Process, Voltage, and Temperature) corners
  - NLDM (Non-Linear Delay Model)
  - CCS (Composite Current Source Model)
  - **Will discuss more about PVT corners in later lectures**

Delay Constraints

- Constraints on maximum delay in combinational logic



D2 signal must be confirmed no later than the time required for the setup of FFs, otherwise there will be a violation of the setup time.

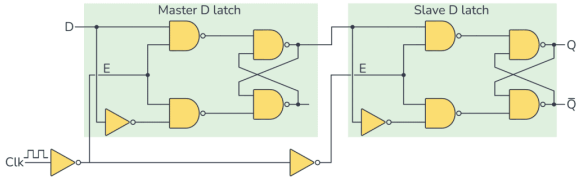
$$t_{pd} \leq T - (t_{su} + t_{c-q})$$

Reference. Vazgen Melikyan, Synopsys University Courseware (Synopsys confidential information)

Checking timing constraints

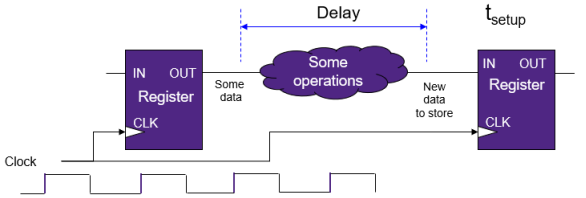
- The delays of combinational circuits in ICs change at large boundaries, depending on PVT.
  - Maximum delay is obtained in the worst case - slow process, low voltage, high temperature. Therefore the **setup condition** should be checked for the **worst case** logical delays and FF timing parameters.
  - Minimum delay is obtained in the best case - fast process, high voltage, low temperature, therefore the **hold condition** should be checked for the **best case** logical delays and FF timing parameters.

- In advanced technology nodes, called **"temperature inversion"** exists.
  - Phenomenon causes transistors to be slower in cold temperatures and faster in hot temperatures.
    - So, max- and min-delay worst cases are technology dependent

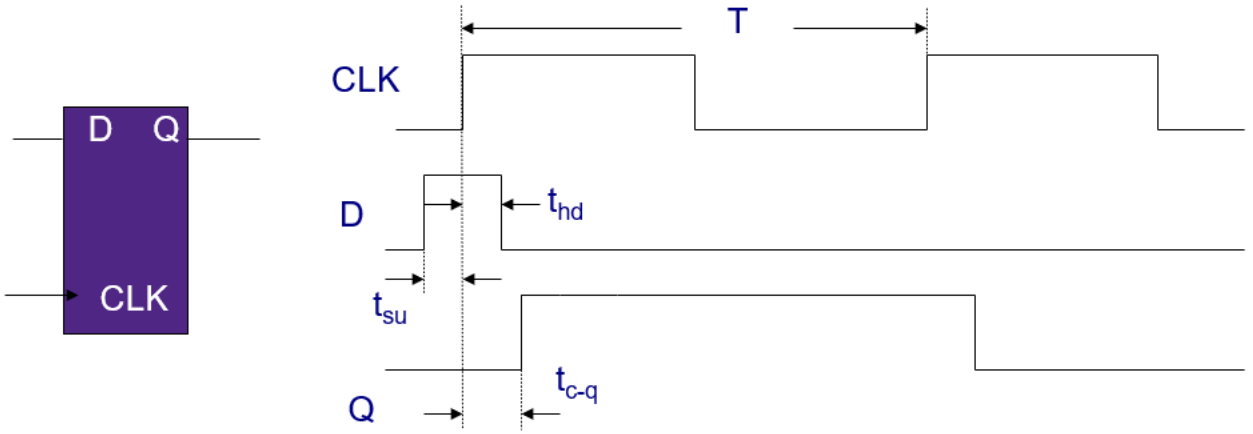


Checking timing constraints

- In a system built on FFs, the data is delivered via a clock pulse edge and must be set up before the next edge arrives.
  - If the data is late, the system will malfunction
  - If the data is set up earlier, the time from the moment to the next edge is wasted
  - Synchronization with FFs requires strict coordination of setup and hold times during clock signal period – in the period of one rising edge to the next one.



Ideal and Real Clock Pulses

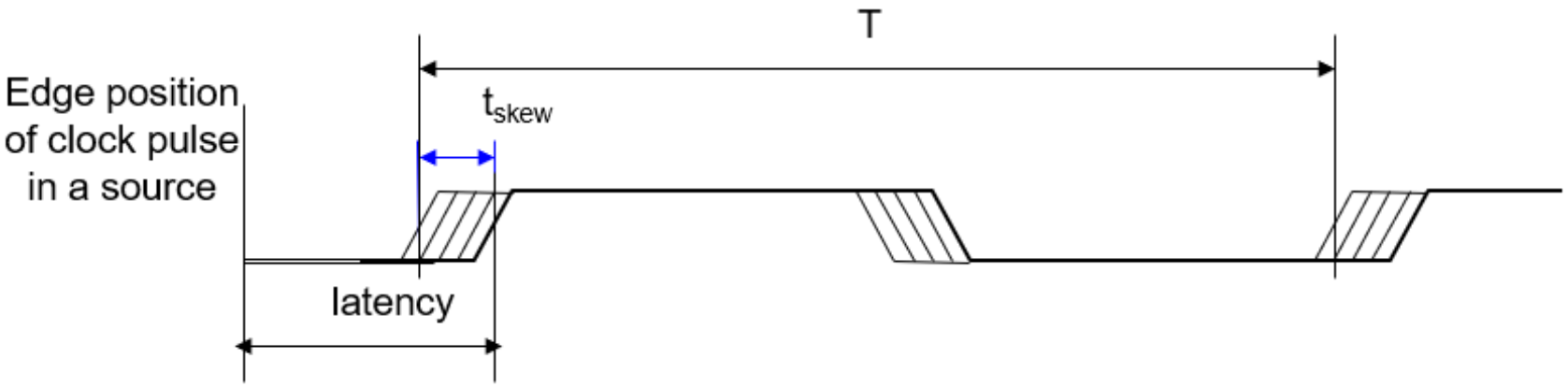


- Ideal Clock
  - Fixed Period
  - Constant rise/fall duration
  - Simultaneous arrival at the inputs of all FFs/latches.

None of these are met…

- For reliable verification of timing constraints in digital systems, it is necessary to take into account possible timing distortions of clock signals

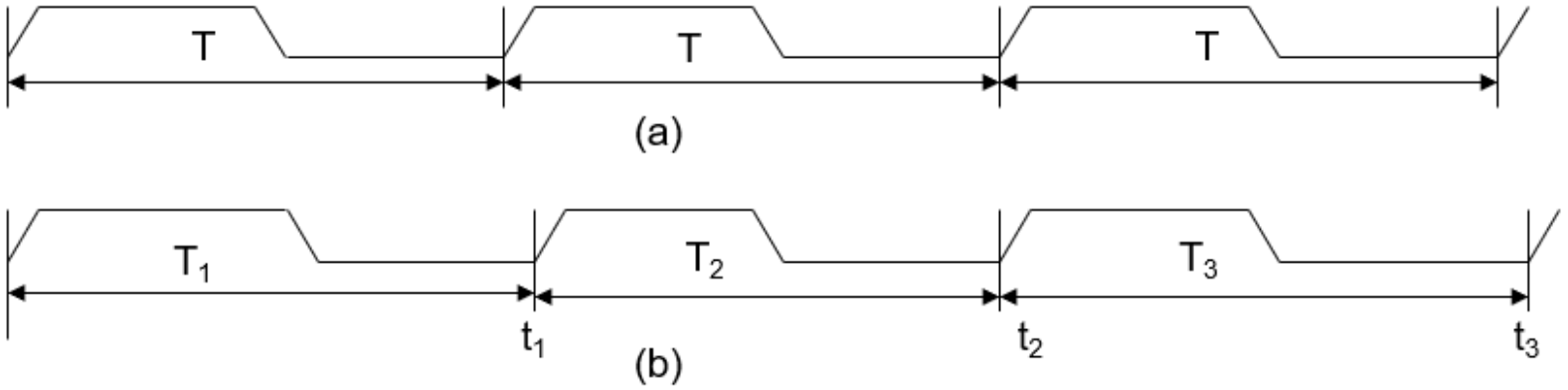
Skew and Latency of Clock Signal



- The skew is equal to the maximum difference between the moments the clock pulse reaches the inputs of **different** FFs of the system.
- The latency is the maximum delay from the source of the clock pulse to the input of FF.

Clock Jitter

- Jitter is observed as from cycle-to-cycle movement of pulse edge position or the instability of the pulse over time

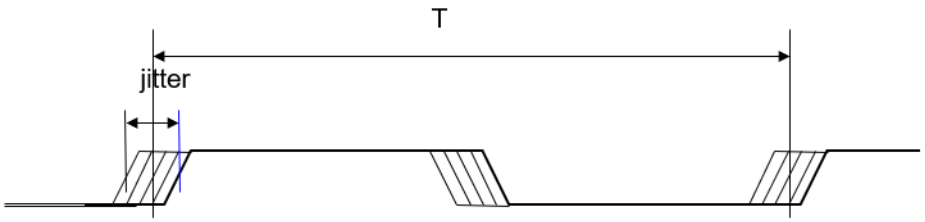


(a) Ideal case - there is no jitter

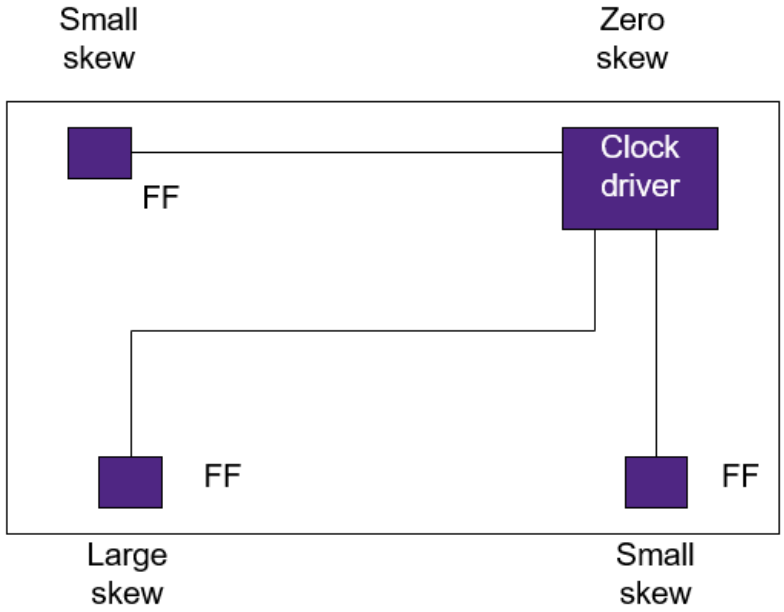
(b) absolute jitter: (t_n-nT) and relative jitter: ((t_n-t_{n-1})-T)

Clock Jitter and Skew

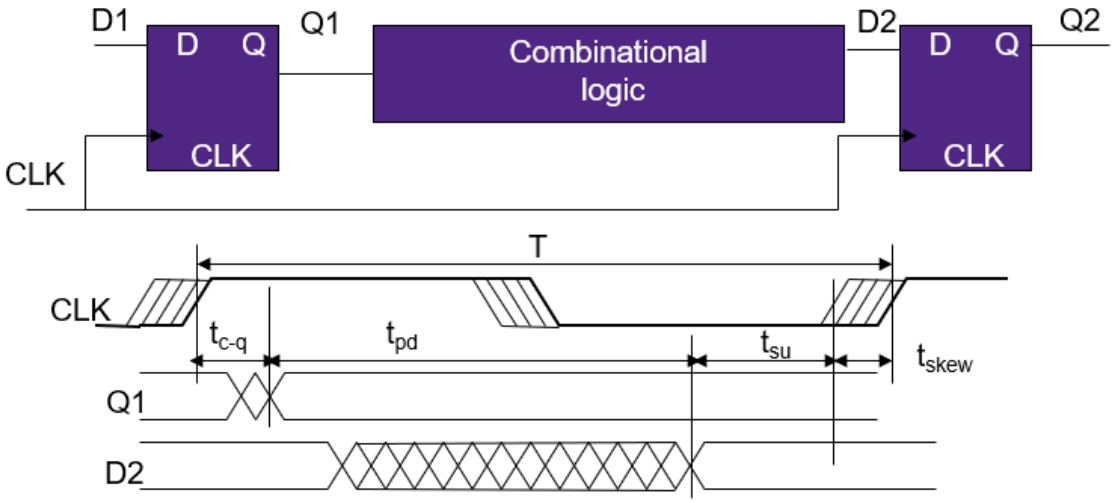
- Jitter is observed as from cycle-to-cycle movement of pulse edge position or the instability of the pulse over time (Same point)



- Clock Skew
  - Difference in arrival of clock pulse at different points in an IC.

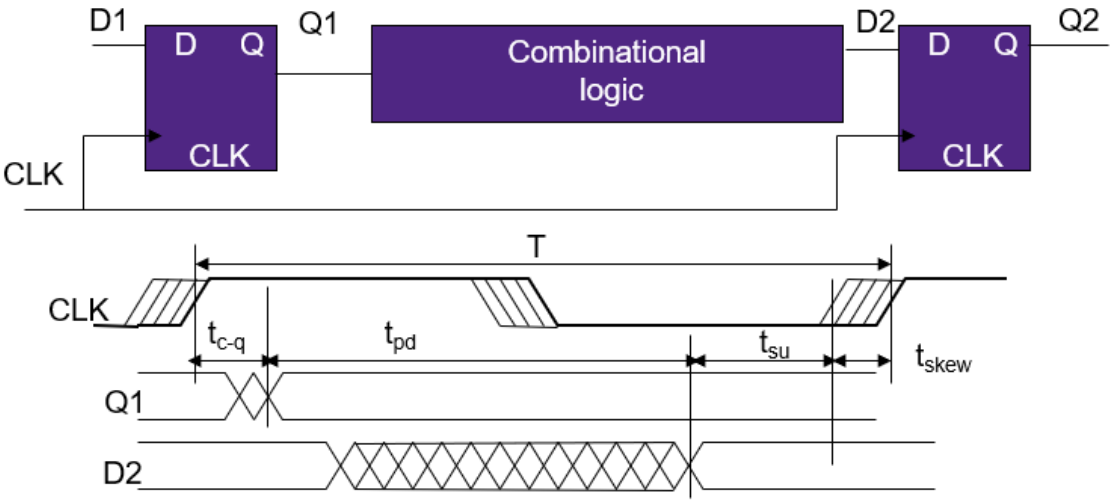


Skew impact on setup constraint



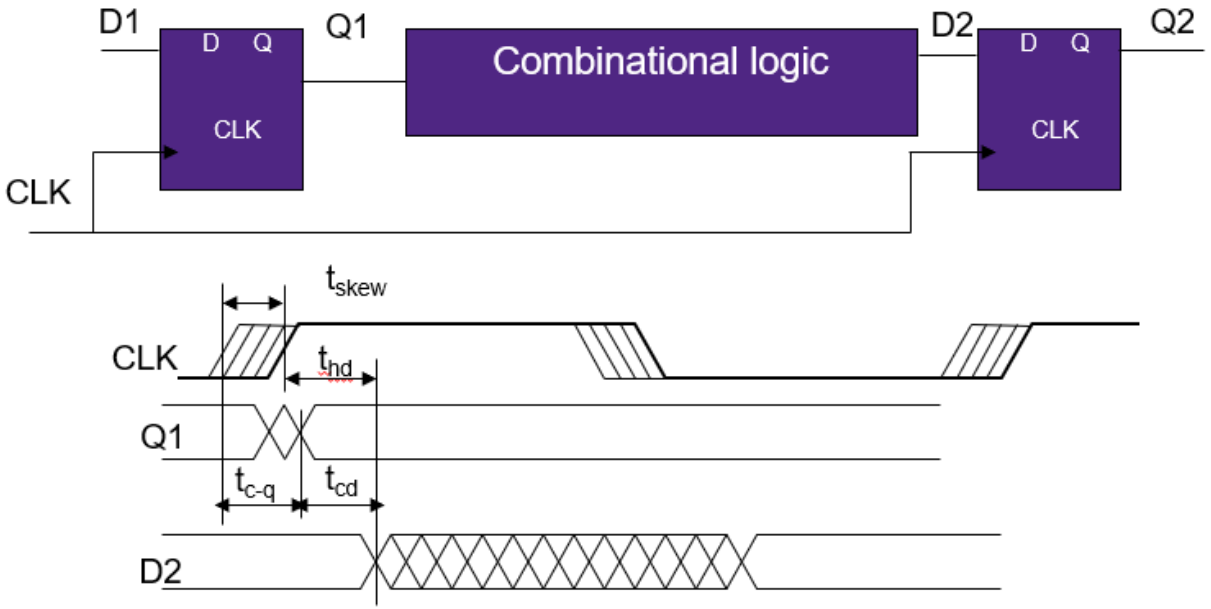
$$T = t_{c-q} + t_{pd} + t_{su} + t_{skew}$$

Setup Slack



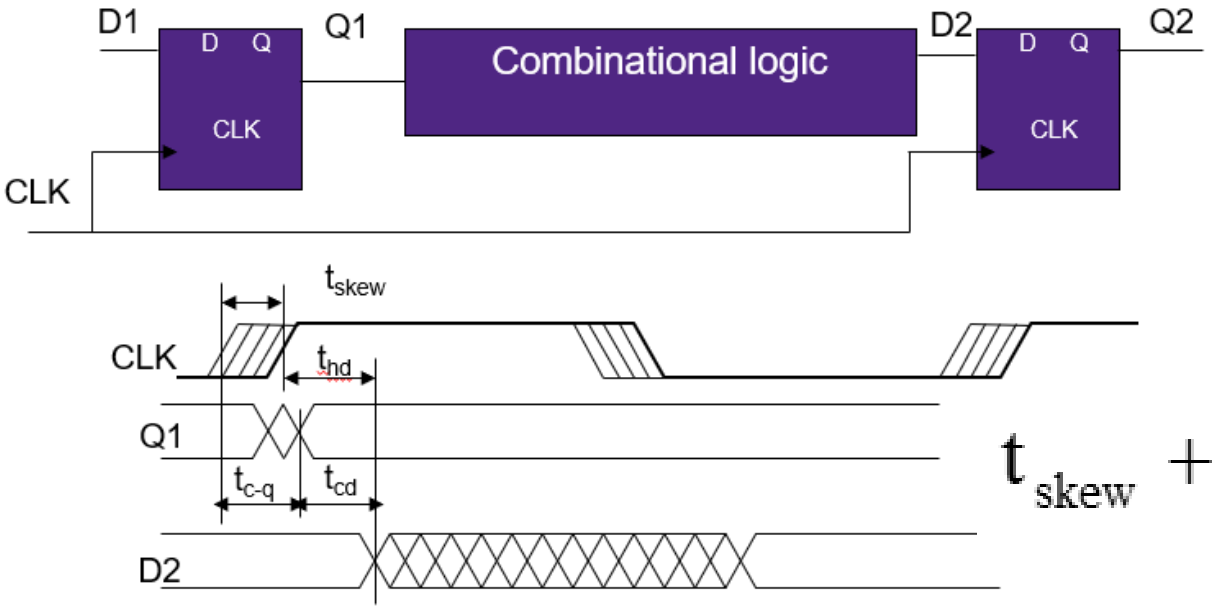
- **Setup slack** is the margin by which a timing path meets setup check requirement.
  - If setup slack is positive, it means the timing path meets setup requirement.
  - A negative setup slack means setup violating timing path.

Skew impact on hold constraint



$$t_{skew} + t_{hold} < t_{c-q} + t_{cd}$$

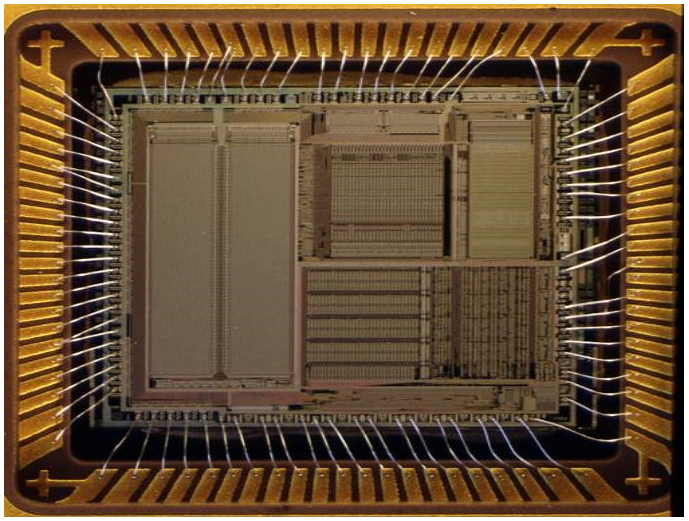
Hold slack



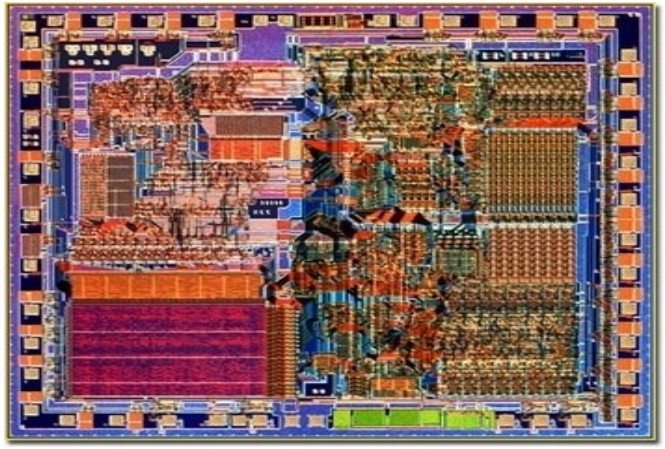
$$t_{skew} + t_{hold} < t_{c-q} + t_{cd}$$

- Hold Slack
  - The presence and magnitude of hold violation is governed by a parameter called as hold slack.
    - If hold slack is positive, it means there is still some margin available before it will start violating for hold.
    - A negative hold slack means the path is violating hold timing check by the amount represented by hold slack.

Trick Question



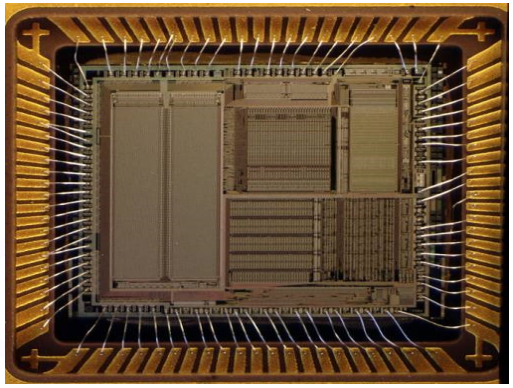
- If, by chance, a fabricated design is found to have a **setup** violation
  - Solution ?
    - Operate the chip at lower clock speed



- If, by chance, a fabricated design is found to have a **hold** violation
  - Solution ?
    - Throwaway the chip…

Hold violation mitigation

- Hold violations occur when data changes too soon after the clock edge, violating the hold time requirement. Fixing hold violations ensures reliable data capture.
- If a timing path violates for hold, we can do either of the following:
  - Increase data path delay
    - Buffers/Inverters/Delay cell insertion
    - Changing cells from lower VT to higher VT
      - Delay Constraints (LVT >> SVT >> HVT)
        - HVT (Higher Voltage)
        - LVT (Lower Voltage)
        - SVT (Standard Voltage)
  - Decrease clock skew
  - Choose a flip-flop with less hold requirement



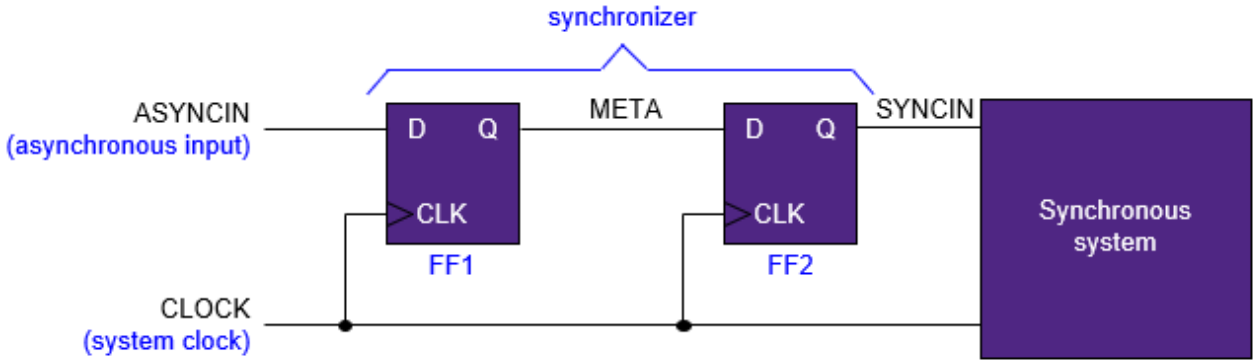
Hold violation mitigation

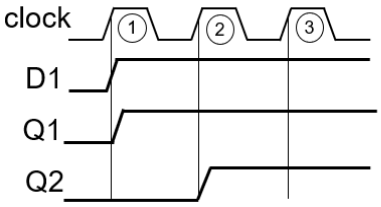
1. Identify Violations: Use STA tools like (PrimeTime/Tempus) to locate paths with negative hold slack.
2. Analyze Paths: Check for short data paths or excessive clock skew.
3. Add Delay Elements: Insert buffers or delay cells to slow down the data path.
4. Adjust Clock Skew: Optimize the clock tree to balance arrival times.
5. Re-run STA: Verify fixes by checking for positive hold slack.
6. Validate Design: Ensure fixes don’t introduce new setup violations.

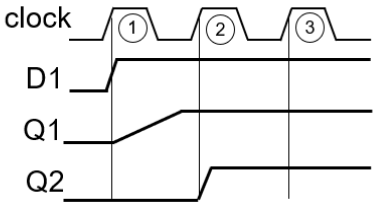
https://logicmadness.com/static-timing-analysis-challenges/
Ref. Vazgen Melikyan, Synopsys University Courseware (Synopsys confidential information)

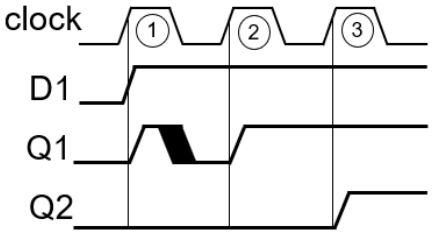
Rule-of-thumb: in over 1GHz clock domains
a 3-stage synchronizers should be used.

Metastability and Synchronizer









(left) Q1 can switch in the beginning of first cycle and Q2 will write the new value of Q1 in 2nd cycle. (right) FF1 appears in metastable state, its output increases and eventually reaches high level. Like in the previous case, Q2 registers accurate data in second cycle

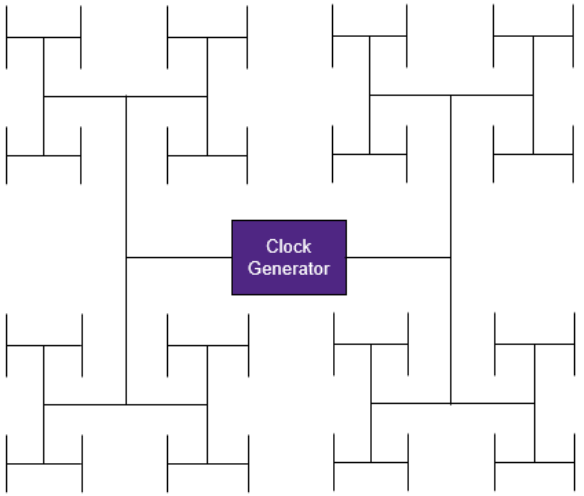
FF1 appears in a metastable state, but its output first goes to high, then low state, afterwards low state is determined. At the end of #1 cycle, Q1 is in low state, it switches to high state in cycle #3.

https://logicmadness.com/static-timing-analysis-challenges/
Ref. Vazgen Melikyan, Synopsys University Courseware (Synopsys confidential information)

Distribution and propagation of clock signal

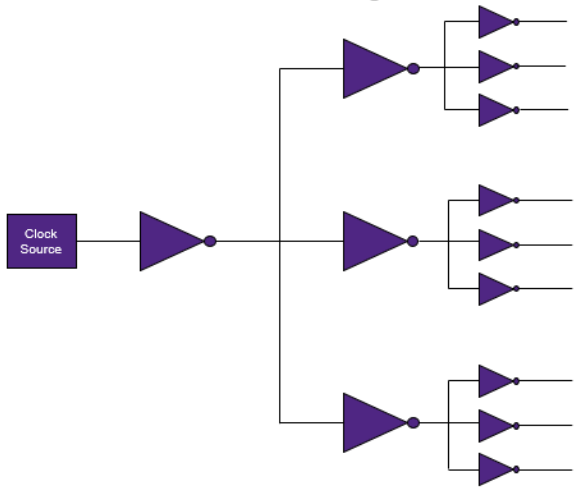
H-Tree for clock distribution

- H tree of clock signal propagation to ensure minimum skew



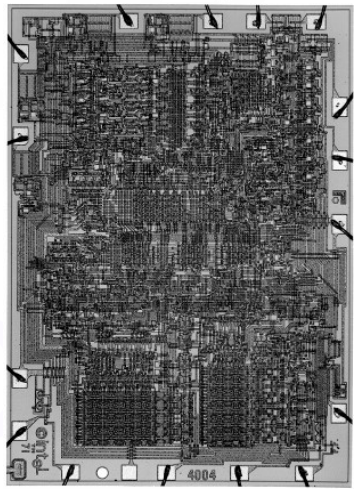
Clock signal buffering

- Each degree of clock signal tree buffering must have the same fan coefficient so that clock signal delays should be well balanced.
- Clock Tree Synthesis (CTS) will be discussed in later lectures



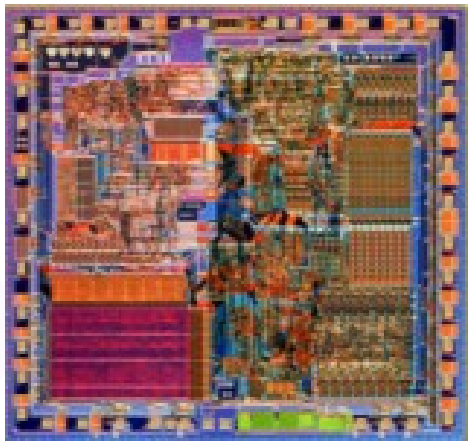
Digital Circuits Timing Constraints/Goals

- Achievement of required operating frequency
  - Digital circuits are constrained to operate over specific frequencies
  - All separate parts/sub-components of a circuit/design are constrained to have delay smaller than clock period
- Meeting timing constraints
  - Avoiding collision of signals
  - Avoiding failure



Digital Circuits Timing Constraints/Goals

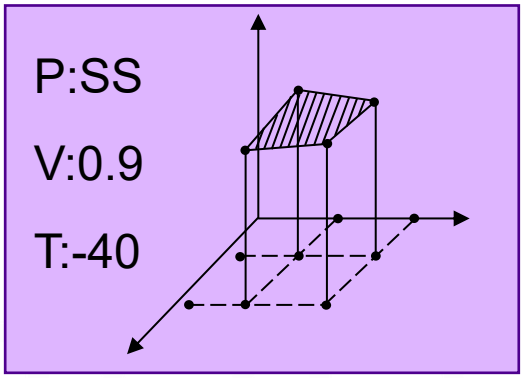
- Achievement of required operating frequency
  - Digital circuits are constrained to operate over specific frequencies
  - All separate parts/sub-components of a circuit/design are constrained to have delay smaller than clock period
- Meeting timing constraints
  - Avoiding collision of signals
  - Avoiding failure



STA Verifies these goals

Problem Variability

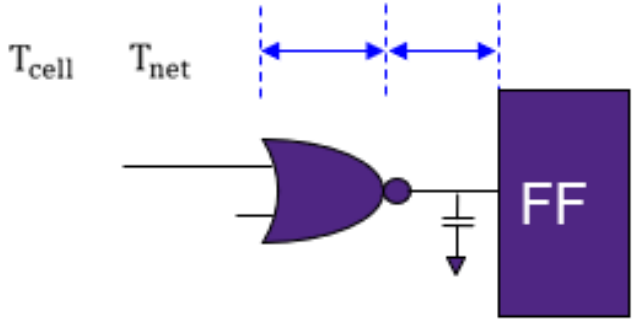
- Other variables affect circuit timing, thus need to be considered during design
  - Operating Conditions
    - Process, Voltage, Temperature variations
  - Unstable clock frequency (jitter, skew)
    - Instability of clock frequency requires design margin
  - On-chip variation (OCV)
    - Device/Interconnect



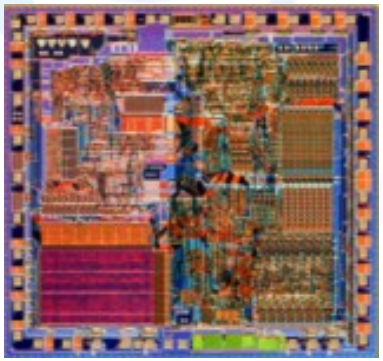
Uncertainties
Tech. deviations
Increase in the number of Design rules

Components of Circuit Timing

- Delay components
  - Cells, Interconnects
- Constrained components
  - Clocked registers require setup/hold, recovery/removal constraints



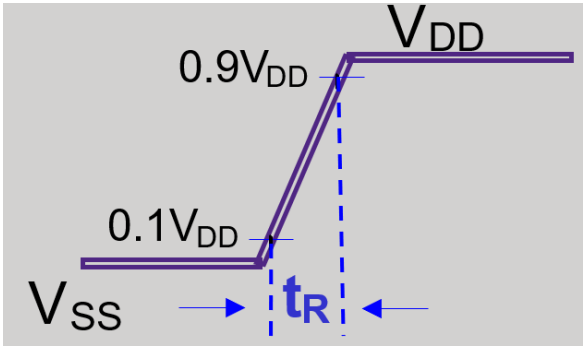
$$t_{path} = t_{net} + t_{cell}$$



Cell Timing Parameters

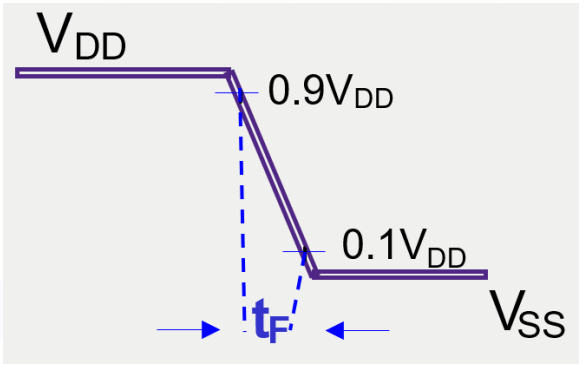
Rise transition time (t_R)

- The time it takes a driving pin to make a transition from kV_{DD} to $(1-k)V_{DD}$ value.



Fall transition time (t_F)

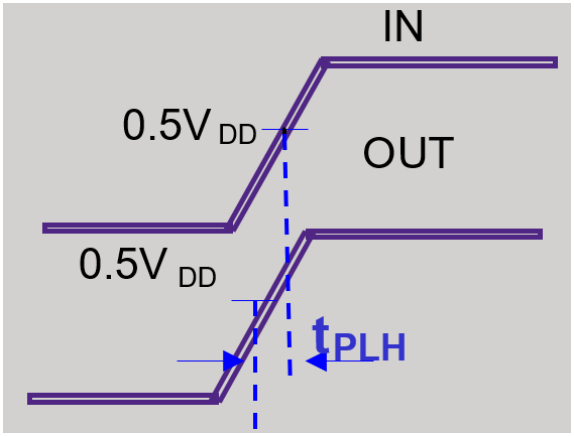
- The time it takes a driving pin to make a transition from $(1-k)V_{DD}$ to kV_{DD} value.
  - Usually k=0.1 (also possible k=0.2, 0.3, etc)
  - 10% to 90% (k=0.1)



Cell Timing Parameters

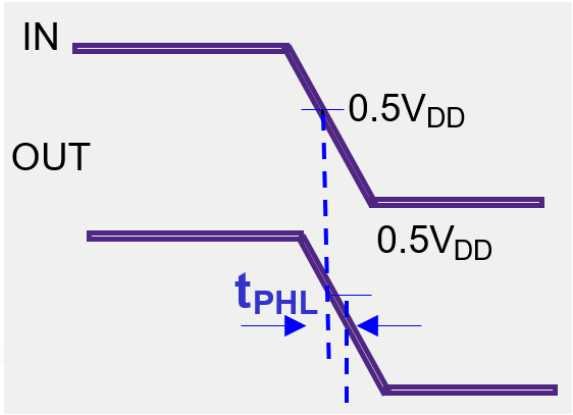
Propagation delay low-to-high (rise) (t_{PLH} or t_{PR})

- Time difference between the input signal crossing a $0.5V_{DD}$ and the output signal crossing its $0.5V_{DD}$ when the output signal is changing from low to high



Propagation delay high-to-low (fall) (t_{PHL} or t_{PF})

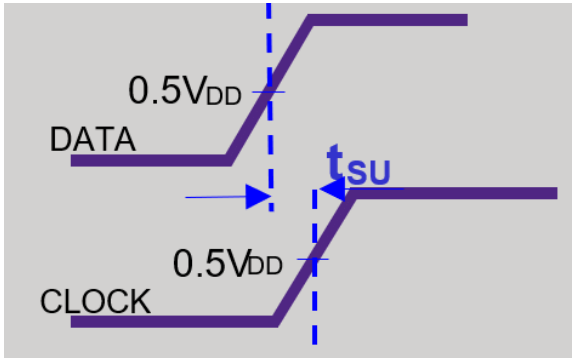
- Time difference between the input signal crossing a 0.5VDD and the output signal crossing its 0.5VDD when the output signal is changing from high to low



Cell Timing Parameters

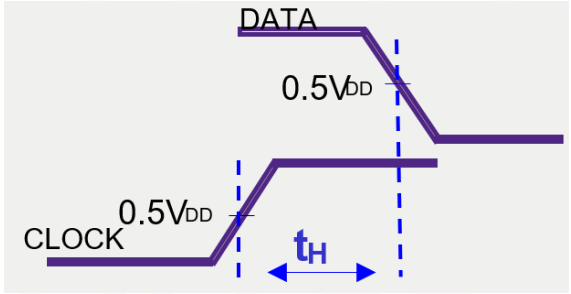
Setup time (t_{SU})

- The minimum period in which the input data to a flip-flop or a latch must be stable before the active edge of the clock occurs

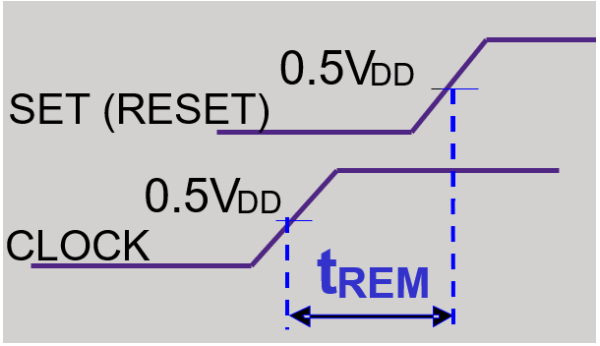


Hold time (t_H)

- The minimum period in which the input data to a flip-flop or a latch must remain stable after the active edge of the clock has occurred

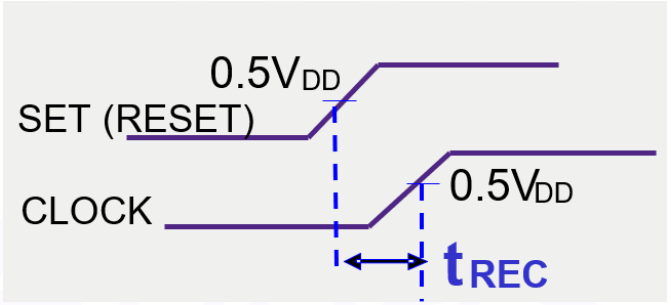


Cell Timing Parameters (only for asynchronous Set or Reset)



Removal time (t_{REM})

- The minimum time in which the asynchronous Set or Reset pin to a flip-flop or latch must remain enabled after the active edge of the clock has occurred

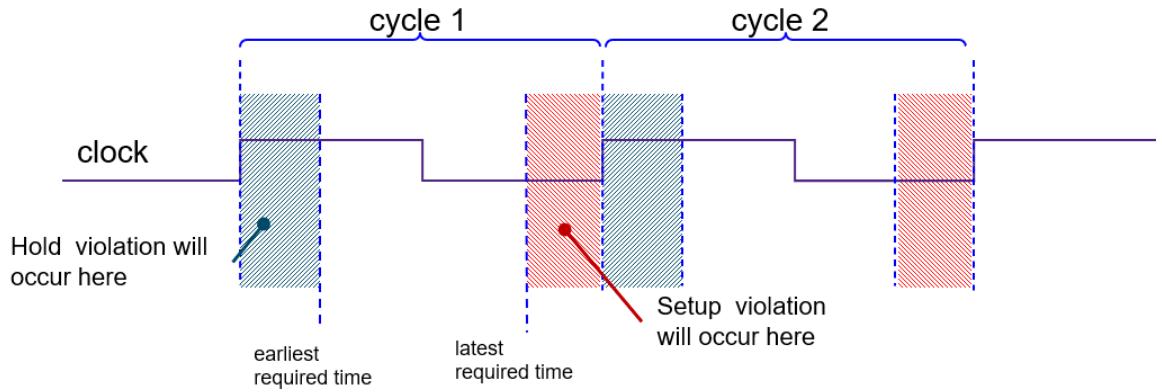


Recovery time (t_{REC})

- The minimum time in which Set or Reset must be held stable after being de-asserted before next active edge of the clock occurs

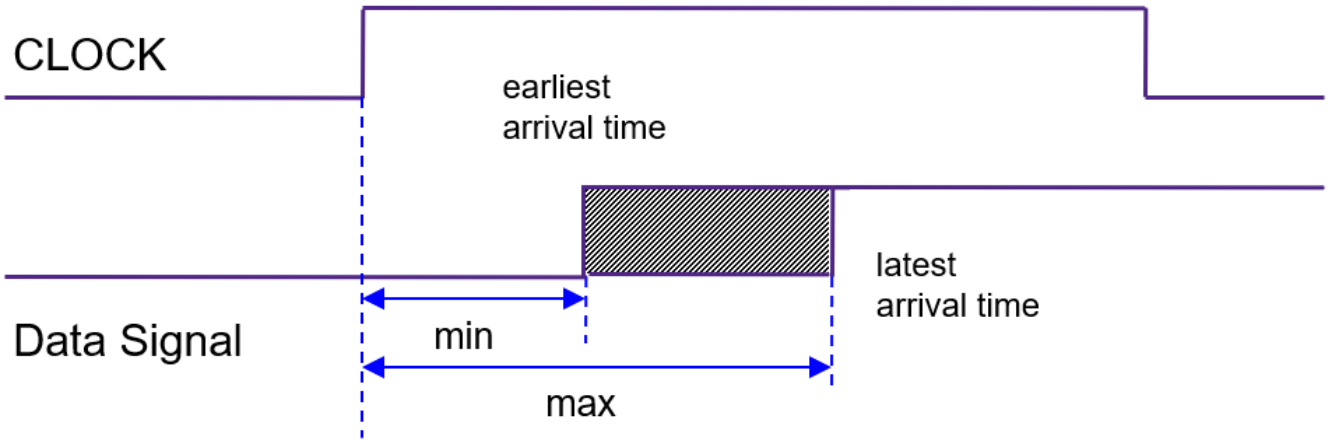
Required Time

- Required time specifies the time point (interval) at which data is required to arrive at end point (data is required to be stable after arrival).
  - Time point after which data can become unstable (change) is called earliest required time (hence we have to HOLD the data at the input)
  - Time point after which data cannot become unstable (change) is called latest required time (hence the data computation should be complete before this point otherwise, setup violations can occur)
- The requirement is set by timing constraints like setup/hold, removal/recovery, etc.



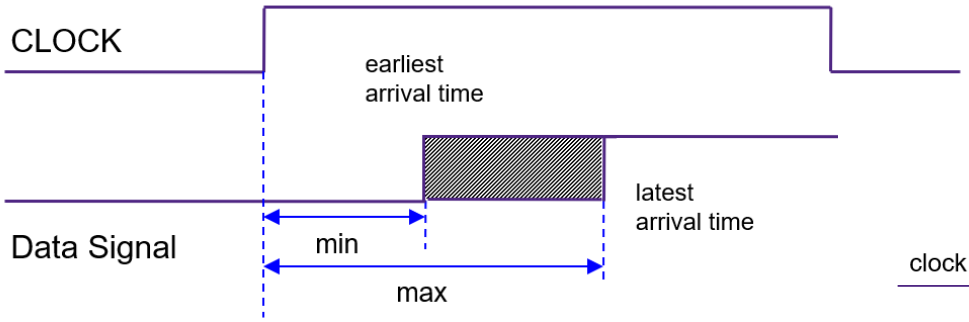
Arrival Time

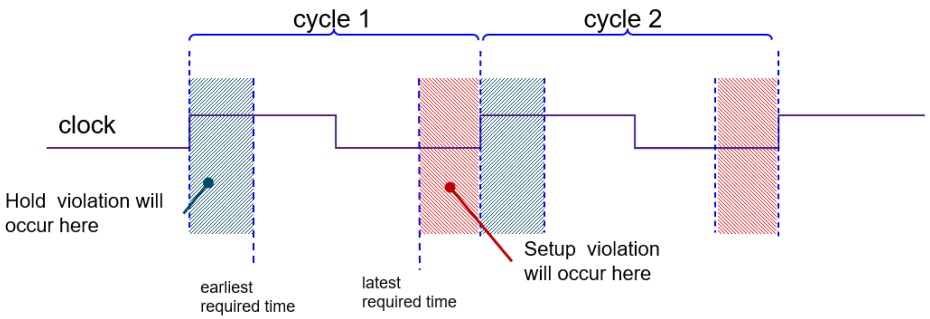
- Arrival time defines the time interval during which a data signal will arrive at a path endpoint (after arrival-time signal will be stable).
- Data arrival depend on circuit delay, which vary (depend on temperature, supply voltage, etc.)
- Minimum delay, early arrival
- Maximum delay, late arrival



Slack and Critical Path

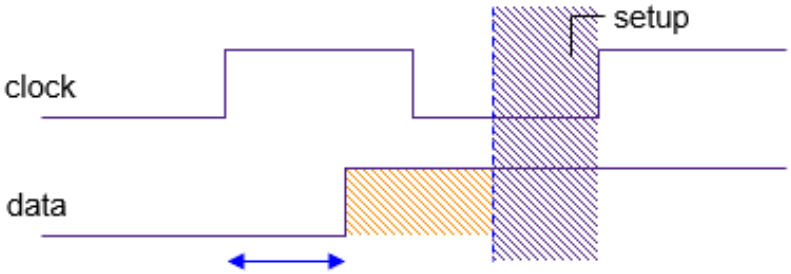
- Slack is the difference between the required time and the arrival time
  - SLACK +ve ---- Constrains have been met
  - SLACL -ve ---- Violations
- Critical path is a path in the design that has the smallest slack.

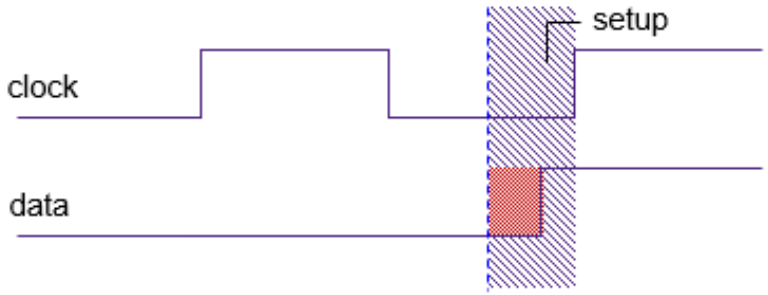




Early and Latest Analysis

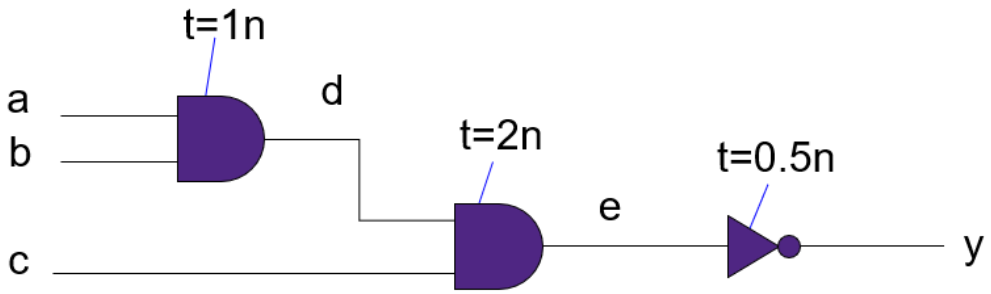
- STA tool calculates the slack of each logic path, in order to find critical path.
- Early and Latest analysis approaches:
  - Assumes circuits have minimum delay, compares arrival time to earliest required time (hold check)
  - Assumes circuits have maximum delay, compares arrival time to latest required time (setup check)





Delay Modeling

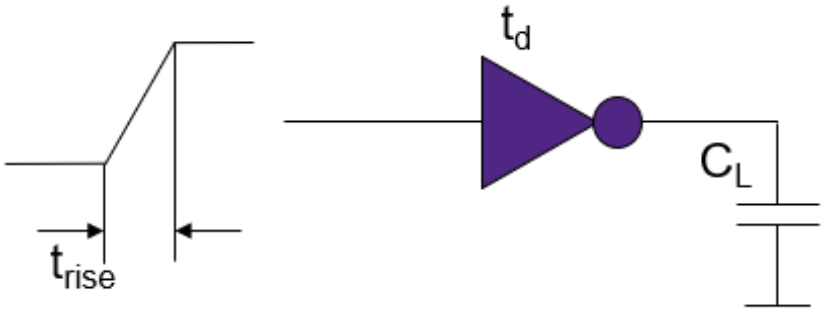
Path Delay: Basic Approach



t_{delay}=3.5n

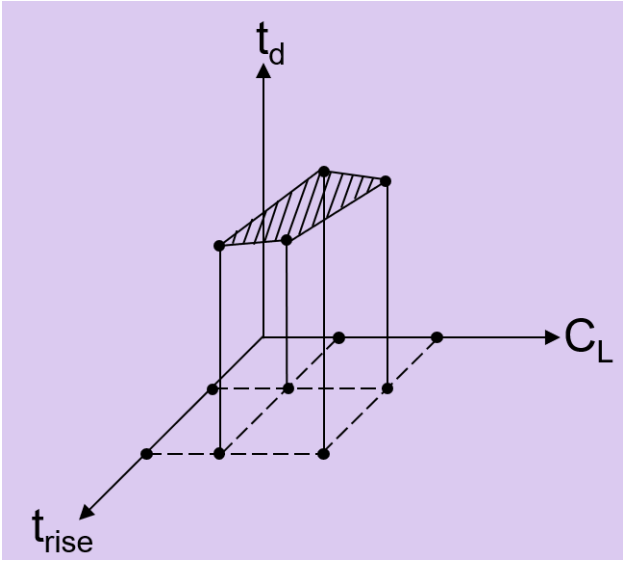
Delay Dependencies

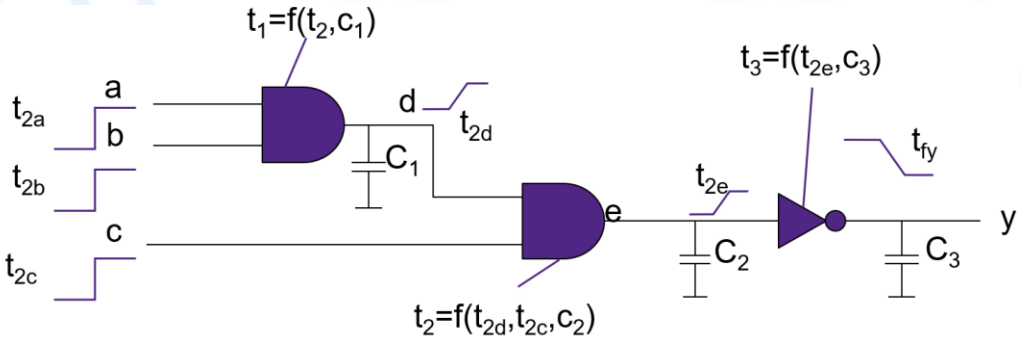
t_{rise}	10ps – 120ps
C_{load}	10fF – 50 fF



Delay Dependencies

t_{delay}=3.5n depends on t_{rise} and C_{load}





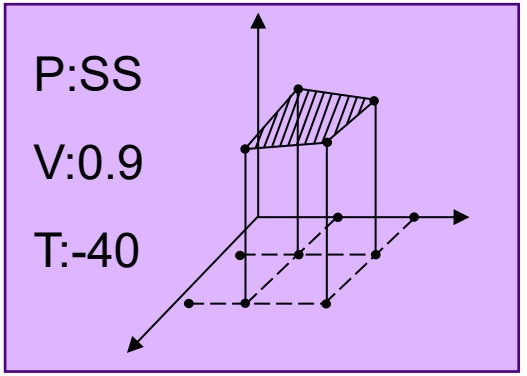
$$t_{total}= f(t_{2a},t_{2b},t_{2c},C_1,C_2,C_3)$$

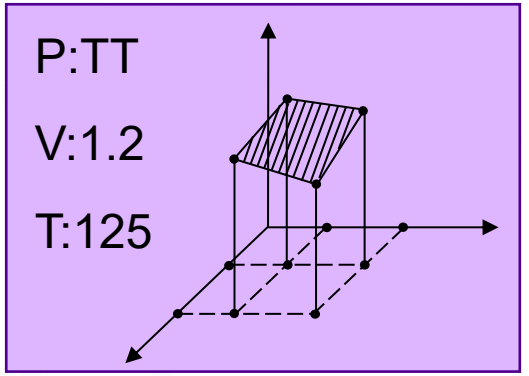
Delay Dependencies: Operating Conditions (PVT)

t_d ~ Process Variations

t_d ~ Voltage

t_d ~ Temperature





Process: TT, FF, SS, etc. (Typical, Fast, Slow)

Voltage: ∓10%

Temperature: -40 -125^{0}C

More on this in Lecture 5

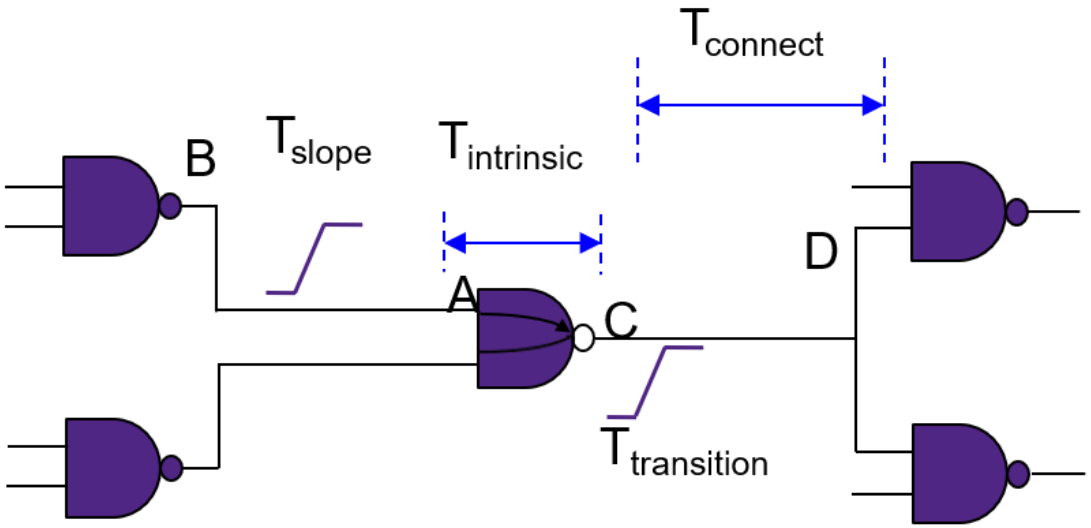
Cell timing models are used to provide accurate timing for various instances of the cells present in the design. The timing model normally obtained from detailed circuit simulation of the cell to model the actual scenario of the cell operation.

Delay Models

Linear Delay Model

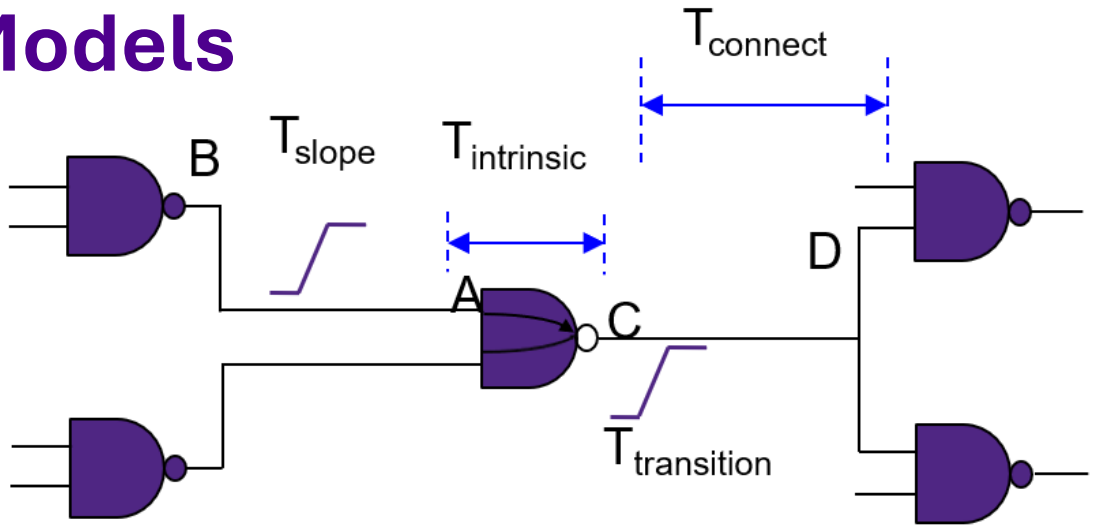
Non-Linear Delay (NDLM) Model

Composite Current Source (CCS) Model



$$T_{total} = T_{slope} + T_{intrinsic} + T_{transition} + T_{connect}$$

Delay Models



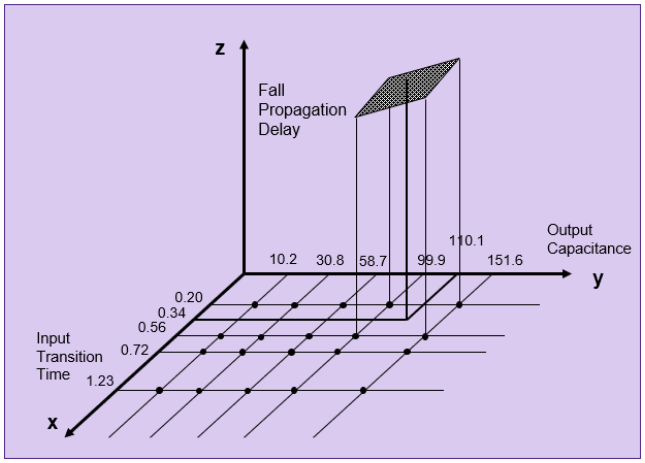
Linear Delay Model

- Slope Delay (T_{slope})
  - The transition time of the previous gate
- Intrinsic Delay ($T_{intrinsic}$)
  - Delay of an element
- Connect Delay ($T_{connect}$)
  - Delay from transition of the driving pin to endpoint
- Transition time ($T_{transition}$)
  - Delay introduced by capacitive load on driving pin
  - $T_{transition} = R_{drive} * (\sum_{pins} C_{pin} + C_{wire})$

Not accurate over the range of input transition time and output capacitance.

For deep Sub-micron technologies, most of the cell libraries use the more complex models like Non-linear Delay Model (NLDM) and CCS model.

Delay Models



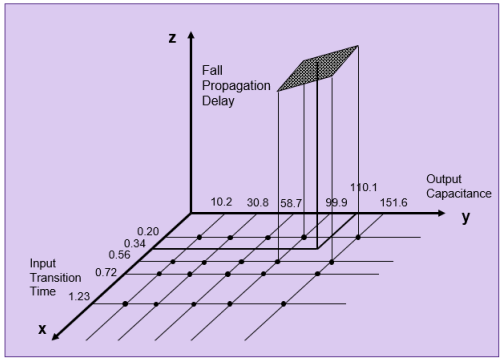
Non-Linear Delay (NDLM) Model

$$T_{total} = T_{propagation} + T_{transition} + T_{connect}$$

- Transition time ($T_{transition}$)
  - Delay introduced by capacitive load on driving pin (measured, not calculated)
- Propagation delay ($T_{propagation}$)
  - Time from the 50 percent input pin voltage until the gate output just begins to switch (10 percent output voltage) (measured, not calculated)
- Connect Delay ($T_{connect}$)
  - Delay from transition of the driving pin (estimated interconnect delay)
- Transition time and Propagation delay for each cell are measured beforehand and stored in form of lookup table

Tampere University

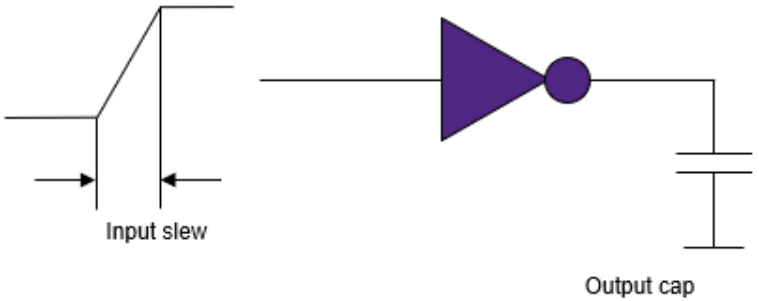
Delay Models



Non-Linear Delay (NDLM) Model

- Look Up Tables (LUTs), the characterization data such as cell delay and transition time is indexed by a fixed number of input transition time and load capacitance values
  - Both NDLM and CCS Model use LUTs
- A Synopsys Liberty (.lib) format file, also known as a timing library file (Lib file), contains several kinds of LUTs for computing cell delay.
- NLDM is a highly accurate timing model as it is derived from SPICE characterizations.

Delay Models



Non-Linear Delay (NDLM) Model

Skew

The time difference between a clock signal's actual and expected arrival time.

Slew

- The time it takes for a signal to transition from one voltage level to another.
- Rate of change of voltage with respect to time.
- The slew (slew rate) is also known as transition delay (10 to 90 percent).

`cell_rise` or `cell_fall` table

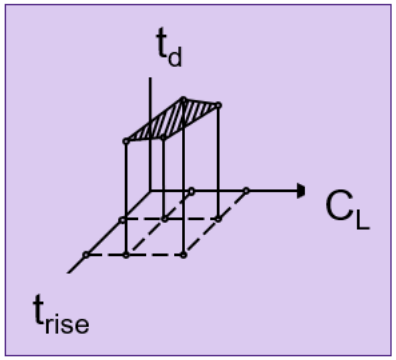
Input slew \ output cap	.023	.047	.065	.078	.091
0.7	0.25	0.27	0.31	0.33	0.4
0.5	0.26	0.28	0.29	0.31	0.35
0.2	0.35	0.37	0.39	0.41	0.45
0.1	0.40	0.41	0.43	0.48	0.51

Synopsys Liberty Format (.lib)

```
library (Digital_Std_Lib) {
     technology (cmos);
     delay_model : table_lookup;
     cell(AND2) {
          area : 2;
          pin(A) {
               direction : input;
          }
          pin(B) {
               direction : input;
          }
          pin(Z) {
               direction : output;
               function : "A*B";
               timing() {
                    related_pin : "A" ;
                    timing_type : "combinational" ;
                    cell_rise(…) {
               index_1("0.016, 0.032, 0.064");
               index_2("2, 4");
                        values("1.0020, 1.1280, 3.547 ", \
                               "1.0080, 1.1310, 3.847 "   );
                    }
               }
          } /* end of pin */
       } /* end of cell */
} /* end of library*/
```

Lookup table

		t_{rise}		
		0.016	0.032	0.064
C_L	2	1.0020	1.1280	3.547
	4	1.0080	1.1310	1.1310



Non-Linear Delay (NDLM) Model

Delay table

- An input fall transition time 0f 0.3ns and an output load of 0.16pf will correspond to the rise delay of the inverter of 0.1018ns

Example of delay table for inverter

```
pin (OUT) {
   max_transition: 1.0;
   timing () {
      related_pin: "INP1";
     timing_sense: negative_unate;
      cell_rise (delay_template_3x3) {
            index_1 ("0.1, 0.3, 0.7"); /* Input transition */
            index_2 ("0.16, 0.35, 1.43"); /* Output capacitance */
            values (/* 0.16 0.35 1.43 */ \
              /* 0.1 */ "0.0513, 0.1537, 0.5280", \
              /* 0.3 */ "0.1018, 0.2327, 0.6476", \
              /* 0.7 */ "0.1334, 0.2973, 0.7252");
      }
      cell_fall (delay_template_3x3) {
            index_1 ("0.1, 0.3, 0.7"); /* Input transition */
            index_2 ("0.16, 0.35, 1.43"); /* Output capacitance */
            values ( /* 0.16 0.35 1.43 */ \
              /* 0.1 */ "0.0617, 0.1537, 0.5280", \
              /* 0.3 */ "0.0918, 0.2027, 0.5676", \
              /* 0.7 */ "0.1034, 0.2273, 0.6452");
      }
```

Tampere University

Cell Timing Data

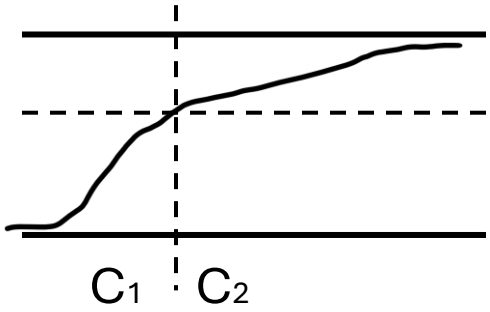
```
library(){
    lu_table_template ("del_1_7_7") {
       variable_1 : "input_net_transition";
       index_1("1, 2, 3, 4, 5, 6, 7");
       variable_2 : "total_output_net_capacitance";
       index_2("1, 2, 3, 4, 5, 6, 7");
    }
```

```
cell (INVX1) {
    pin (Y) {
        timing () {
            related_pin : "A";
            timing_type : "combinational";
            timing_sense : "negative_unate";
            cell_rise ("del_1_7_7") {
                index_1("0.016, 0.032, 0.064, 0.128, 0.256, 0.512, 1.024");
                index_2("0.1, 0.25, 0.5, 1, 2, 4, 8");
                values("0.016861, 0.0179019, 0.0195185, 0.0229259, 0.029658, 0.043145, 0.07712", \
                  "0.0239648, 0.0255491, 0.0279298, 0.0319930, 0.0387540, 0.0520896, 0.0790211", \
                  "0.0342118, 0.0366966, 0.0402223, 0.0462823, 0.0558327, 0.0705154, 0.0967339", \
                  "0.0491695, 0.0524727, 0.0576512, 0.0665647, 0.0810999, 0.1027237, 0.1342571", \
                  "0.0721332, 0.0765389, 0.0836775, 0.0960890, 0.1171612, 0.1497265, 0.1957640", \
                  "0.1111560, 0.1164417, 0.1252609, 0.1422002, 0.1712097, 0.2171862, 0.2847010", \
                  "0.1841131, 0.1901881, 0.2010298, 0.2194395, 0.2555983, 0.3182710, 0.4139452");
            }
```

Delay Analysis

- Calculation of each timing arc's value cell delay or a net delay
  - Positive unate timing arc combines rise delays with rise delays and fall delays with fall delays (Buffers, AND, OR)
  - Negative unate timing arc combines incoming rise delays with local fall delays and vice versa (Inverter, NAND, NOR)
  - Non-unate timing arc combines local delay with the worst – case incoming delays logic functions whose output value change cannot be predicted (XOR, XNOR

Delay Models

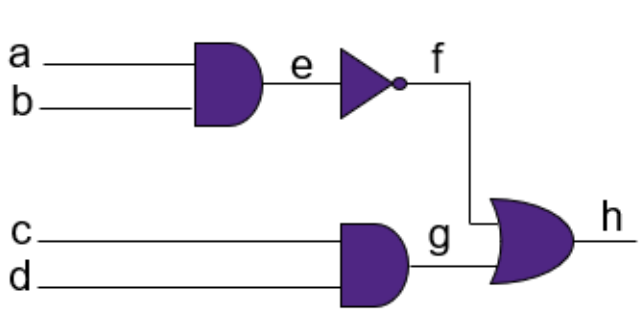


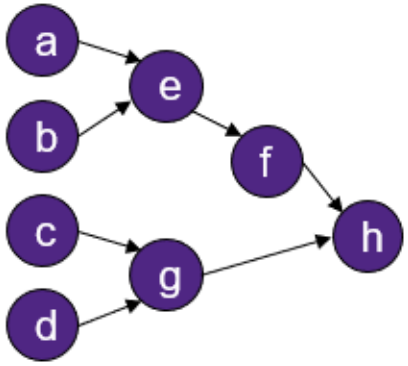
Composite Current Source Model (CCS)

- More accurate than NLDM
- The **driver model** uses a time-varying current source.
- The **receiver model** consists of 2 different capacitors.
  - The first one is used as load up to the input delay threshold. A second capacitance value is used when the input waveform reaches the threshold value.
- CCS models are frequently used in advanced technology nodes.

Path-based Timing Calculation

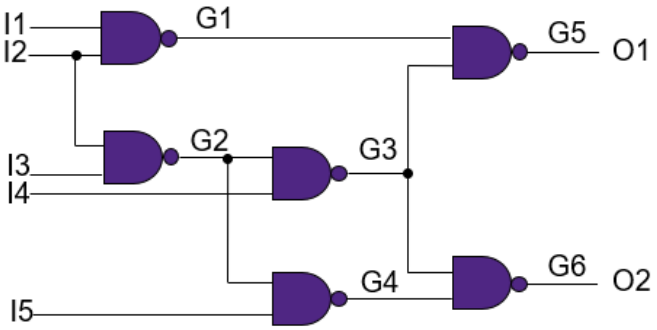
- Calculates minimum and maximum path delay costs
- The timing analyzer represents a netlist as a directed graph
  - Nodes in the graph
  - Edges represent
    - Net delay – interconnect delay between a driver pin and a load pin (its fanout)
    - Cell delay – timing delay between an input pin and an output pin of a cell

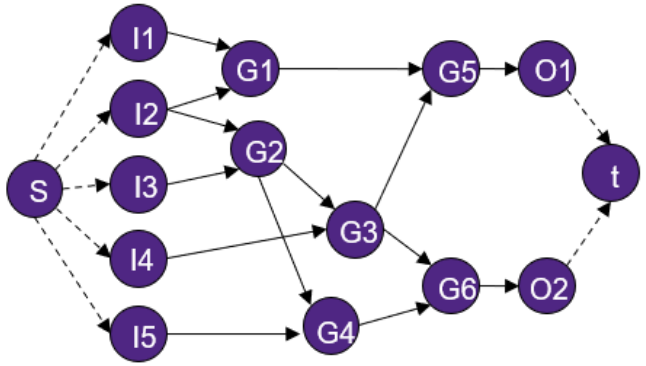




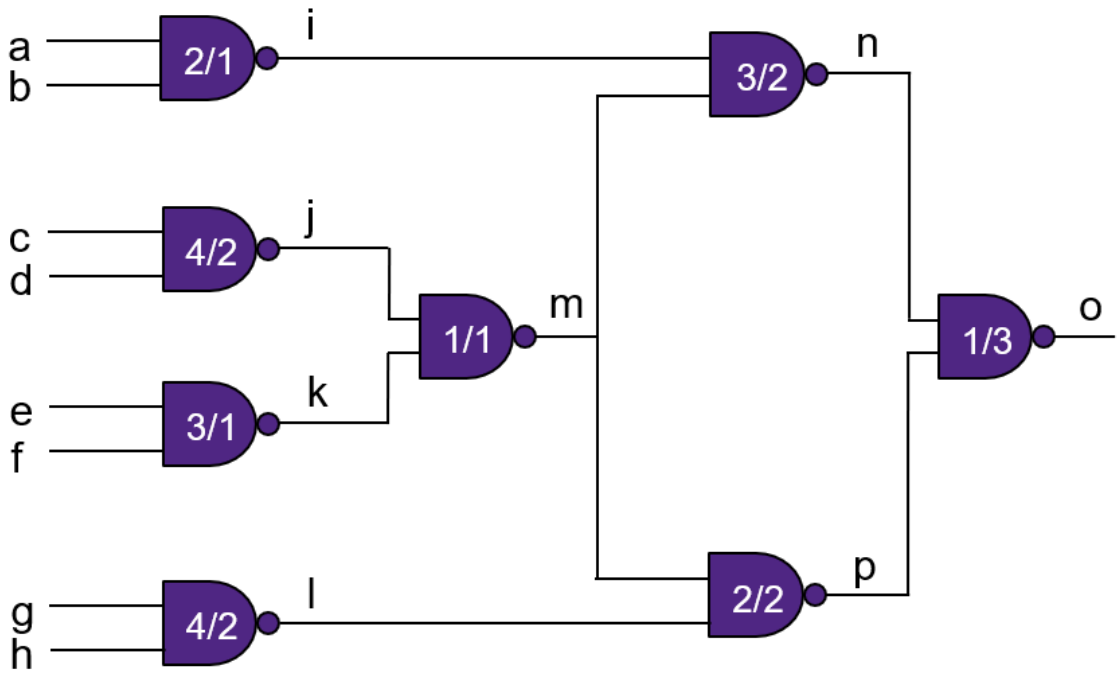
Representation of Circuits

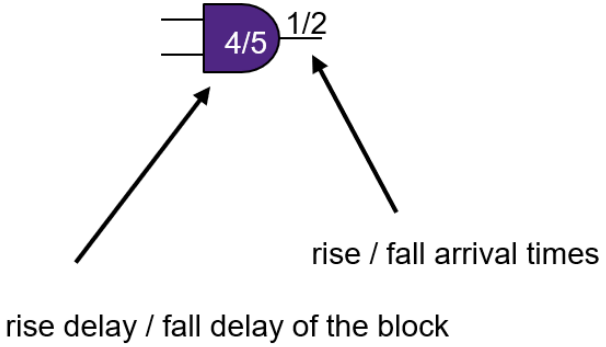
- Combinational logic circuit may be represented as a timing graph G = (V,E)
  - V, the vertex set, are the logic gates in the circuit and the primary inputs and outputs of the circuit
- Vertices, u and v ∈ G , are connected by a directed edge e(u, v) ∈ E
  - Connection from the output of the element represented by vertex u to the input of the element represented by vertex v
- Circuit is represented by directed acyclic graph (DAG)
  - Do not have any cycles





Circuit with Delays of Components





```
Path 1: VIOLATED Setup Check with Pin Controller_Unit/find_min/val_out_reg_9_/
CP
Endpoint:   Controller_Unit/find_min/val_out_reg_9_/D (^) checked with  leading
edge of 'ideal_clk'
Beginpoint: reset                                     (^) triggered by  leading
edge of 'ideal_clk'
Path Groups: {inclkSrc2reg}
Analysis View: view_setup_cmax
Other End Arrival Time          0.000
- Setup                         0.071
+ Phase Shift                  10.000
+ CPPR Adjustment               0.000
- Uncertainty                   1.000
= Required Time                 8.929
- Arrival Time                  8.990
= Slack Time                   -0.061
     Clock Rise Edge                      0.000
     + Input Delay                        2.000
     + Drive Adjustment                   0.100
     = Beginpoint Arrival Time            2.100
     +-------------------------------------------------------------------------------------------+
     |                Instance                |     Arc      |   Cell   | Delay | Arrival | Required |
     |                                        |              |          |       |  Time   |   Time   |
     |----------------------------------------+--------------+----------+-------+---------+----------|
     |                                        | reset ^      |          |       |   2.100 |    2.039 |
     | FE_OFC10209_reset                      | I ^ -> Z ^   | BUFFD16  | 0.141 |   2.241 |    2.180 |
     | FE_OFC10098_n                          | I ^ -> Z ^   | BUFFD16  | 0.118 |   2.359 |    2.298 |
     | sorting/U878                           | A1 ^ -> ZN v | CKND2D1  | 0.129 |   2.488 |    2.427 |
     | sorting/U3147                          | A1 v -> ZN ^ | NR2D3    | 0.317 |   2.805 |    2.744 |
     | sorting/U11                            | A1 ^ -> ZN v | NR2D3    | 0.211 |   3.017 |    2.955 |
     | sorting/U2124                          | A1 v -> ZN ^ | AOI22D1  | 0.168 |   3.185 |    3.124 |
     | sorting/U743                           | B ^ -> ZN v  | OAI211D1 | 0.158 |   3.343 |    3.282 |
     | sorting/U662                           | A1 v -> ZN ^ | ND3D1    | 0.102 |   3.445 |    3.384 |
     | sorting/U638                           | B ^ -> ZN v  | IOA21D2  | 0.072 |   3.517 |    3.455 |
     | sorting/U2007                          | B v -> ZN ^  | IAO21D2  | 0.127 |   3.644 |    3.583 |
     | sorting/U591                           | A1 ^ -> ZN v | NR2D0    | 0.092 |   3.736 |    3.674 |
     | sorting/U535                           | A1 v -> ZN ^ | NR2D1    | 0.145 |   3.881 |    3.820 |
     | sorting/U3159                          | A1 ^ -> ZN v | OAI211D1 | 0.144 |   4.025 |    3.964 |
```

- Path Timing Report

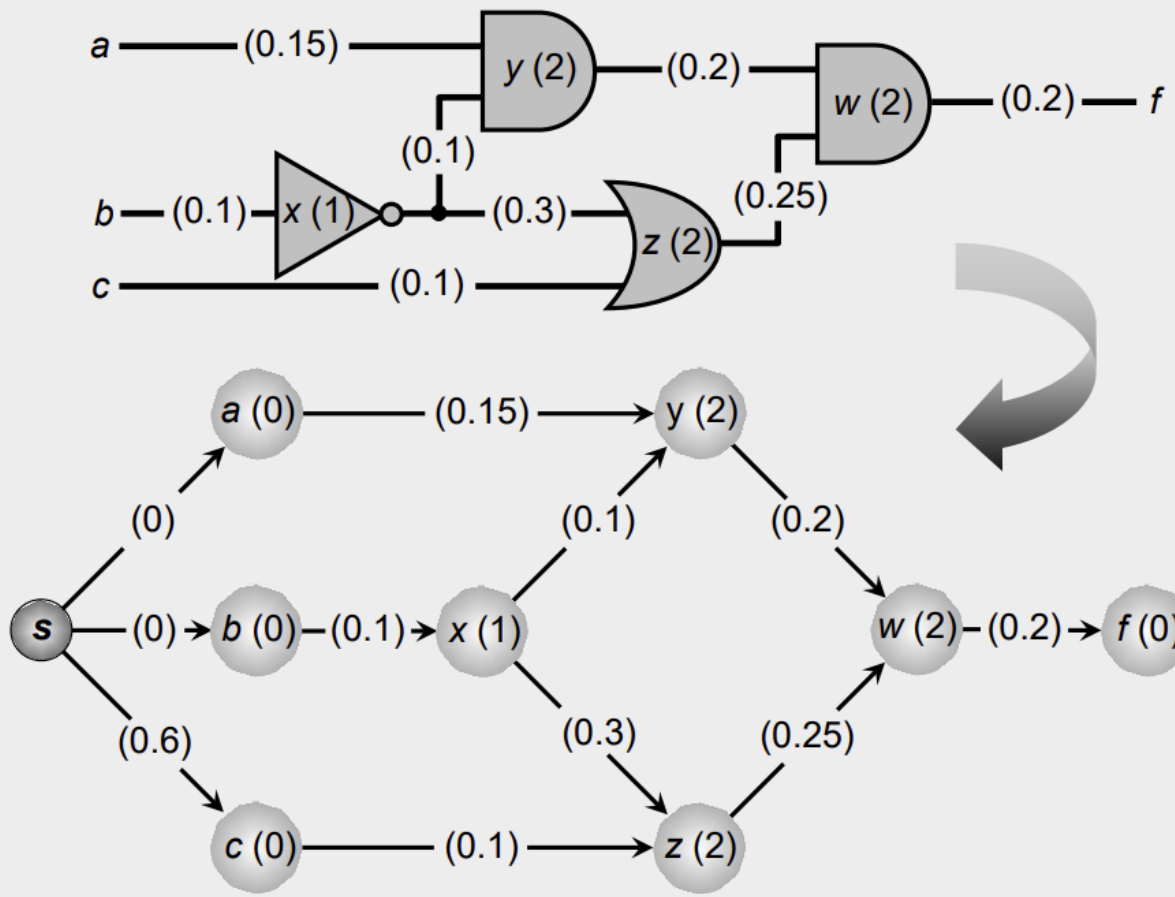
Arrival Time/Violations

- Arrival Time (AT)
  - The Arrival Time at a node is just the maximum of the ATs at the predecessor nodes plus the delay from that node.
- Required Arrival Time (RAT)
  - The Required Arrival Time to a node is just the minimum of the RATs at the successor nodes minus the delay to that node
- Max Delay (Max Constraints)
  - Data doesn't have enough time to pass from one register/FFs to the next.
    - Slow Data Path
    - Setup Violations
- Min Delay (Min Constraints)
  - Datapath is so short that it passes several registers/FFs during one clock cycle
    - Short Data Path
    - Hold Violations

STA/AT

Slack = RAT – AT for each node

AAT. Actual Arrival Time
RAT. Required Arrival Time

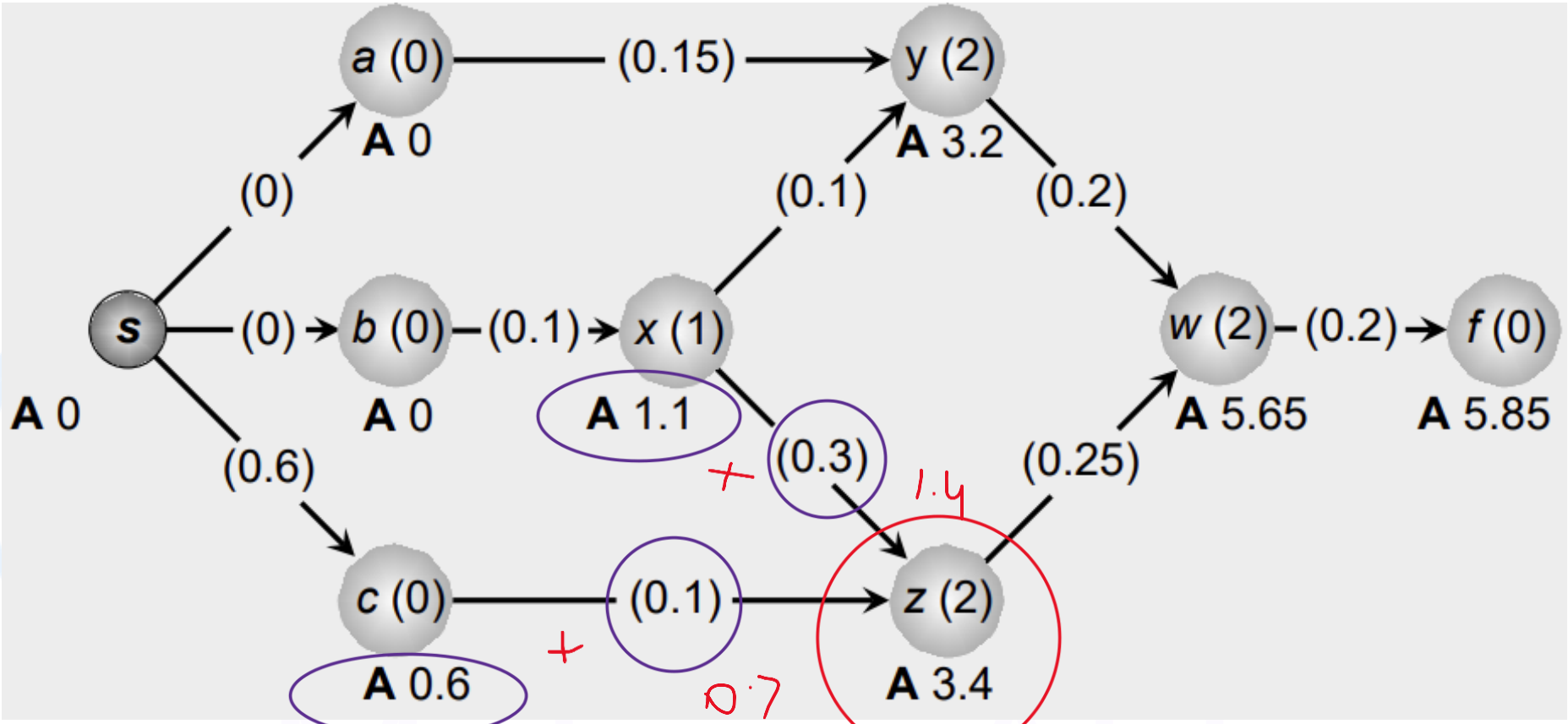


Compute AATs at each node:

$$AAT(v) = \max_{u \in FI(v)} \left(AAT(u) + t(u,v) \right)$$

where $FI(v)$ is the fanin nodes, and $t(u,v)$ is the delay between u and v
(AATs of inputs are given)

- Given combinational circuit, represent as directed acyclic graph (DAG) – Every edge (node) has weight = wire (gate) delay



- Arrival Time computation (Forward Pass)

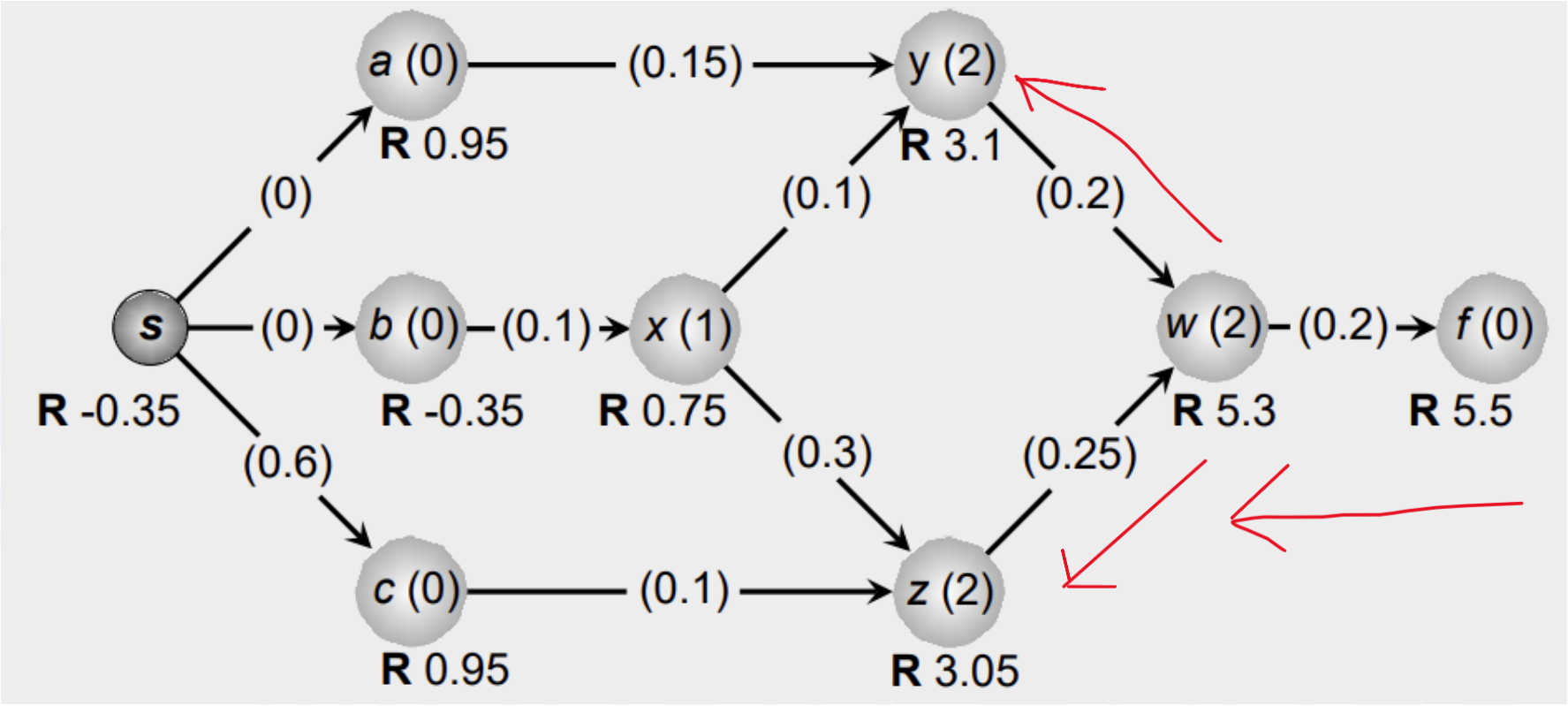
STA/RAT

Compute RATs at each node:

$$RAT(v) = \min_{u \in FO(v)} \left(RAT(u) - t(u,v) \right)$$

where $FO(v)$ are the fanout nodes, and $t(u,v)$ is the delay between u and v

(RATs of outputs are given)

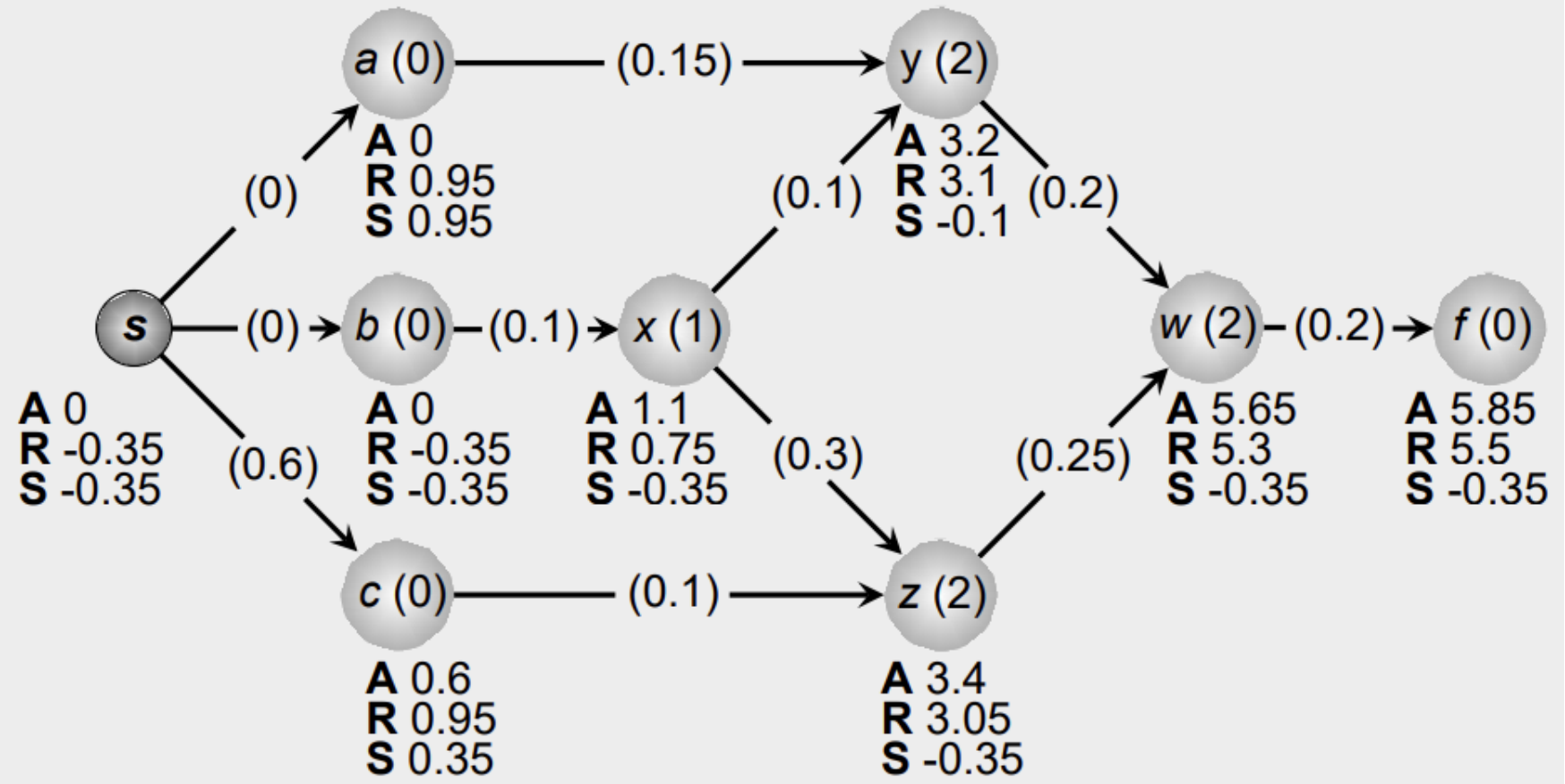


Forward to backward pass /Assume Clock period of 5.5 / Min delay

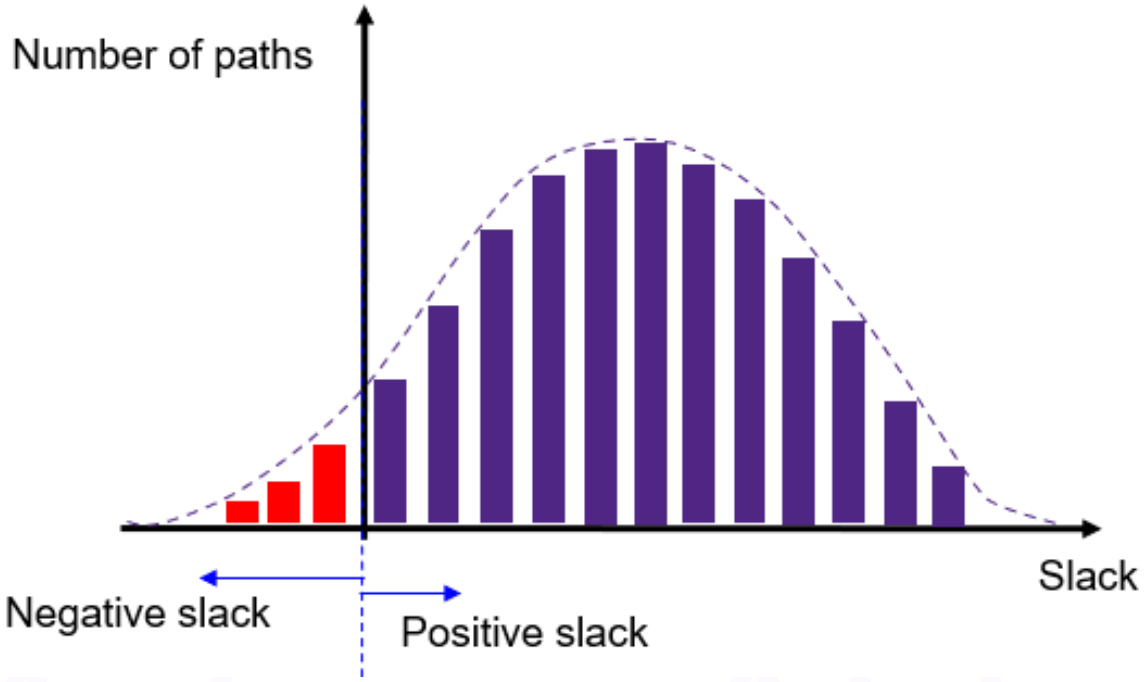
STA/SLACK

Compute slacks at each node:

$$slack(v) = RAT(v) - AAT(v)$$

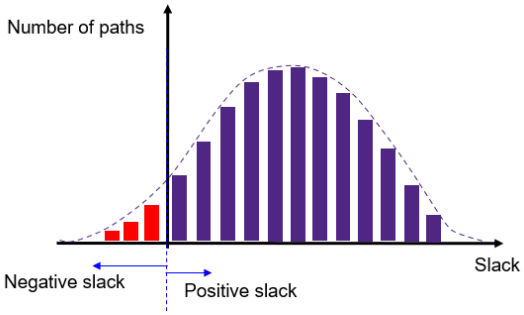


Path Slack Histogram

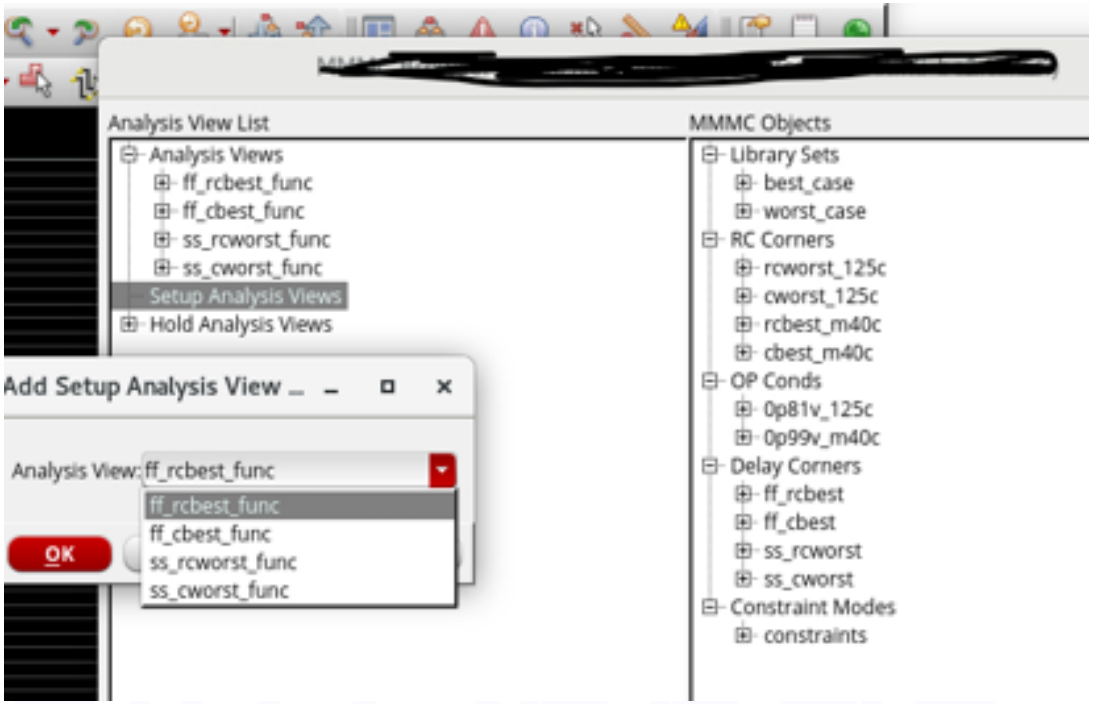


- If there is no path with negative slack, this will mean design does not have timing violations.

Timing Checks



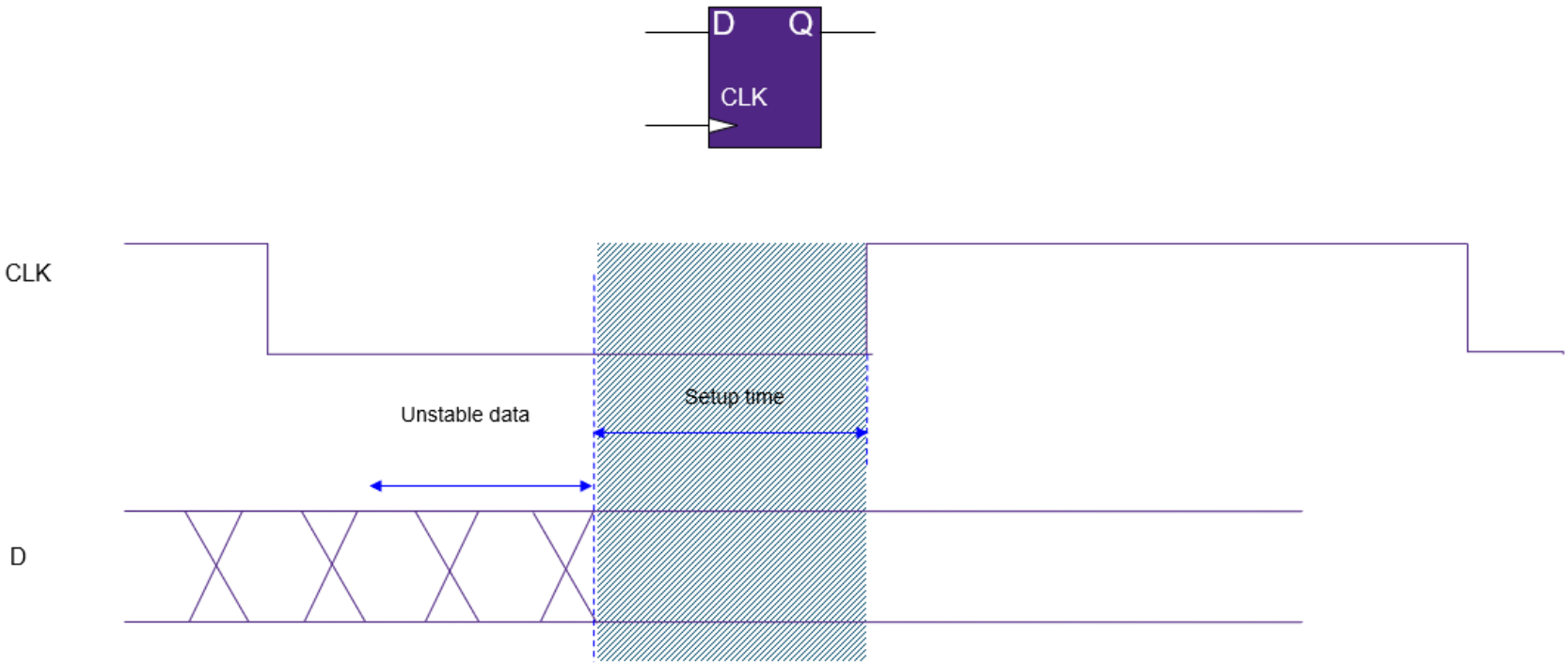
- Constraints set by sequential circuits:
  - Setup
  - Hold
- Possible conditions:
  - Best-case (Hold)
  - Worst-case (Setup)



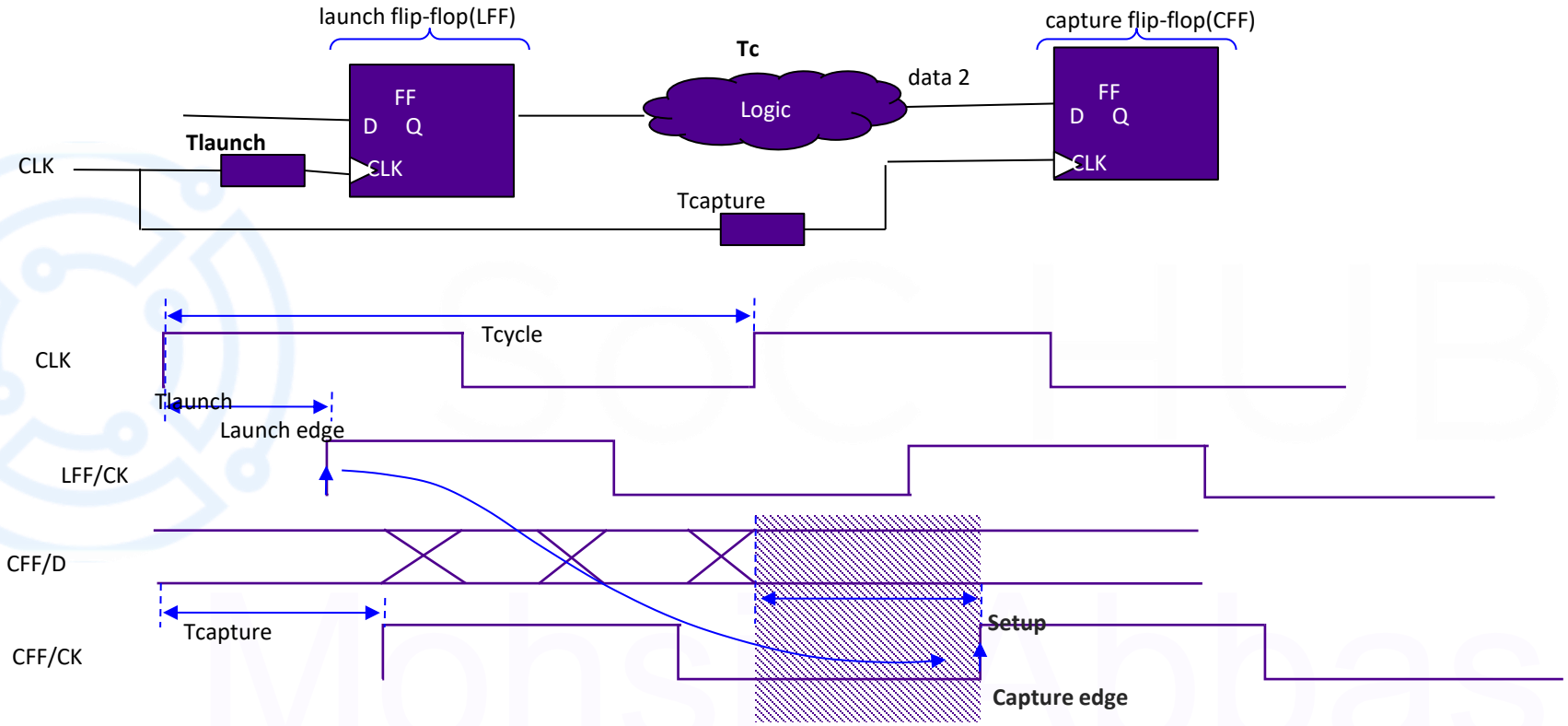

```
set_analysis_view -setup {ss_rcworst_func ss_cworst_func} -hold {ff_rcbest_func ff_cbest_func}
```

Setup Timing Check

- Data must be stable before the active edge of the clock



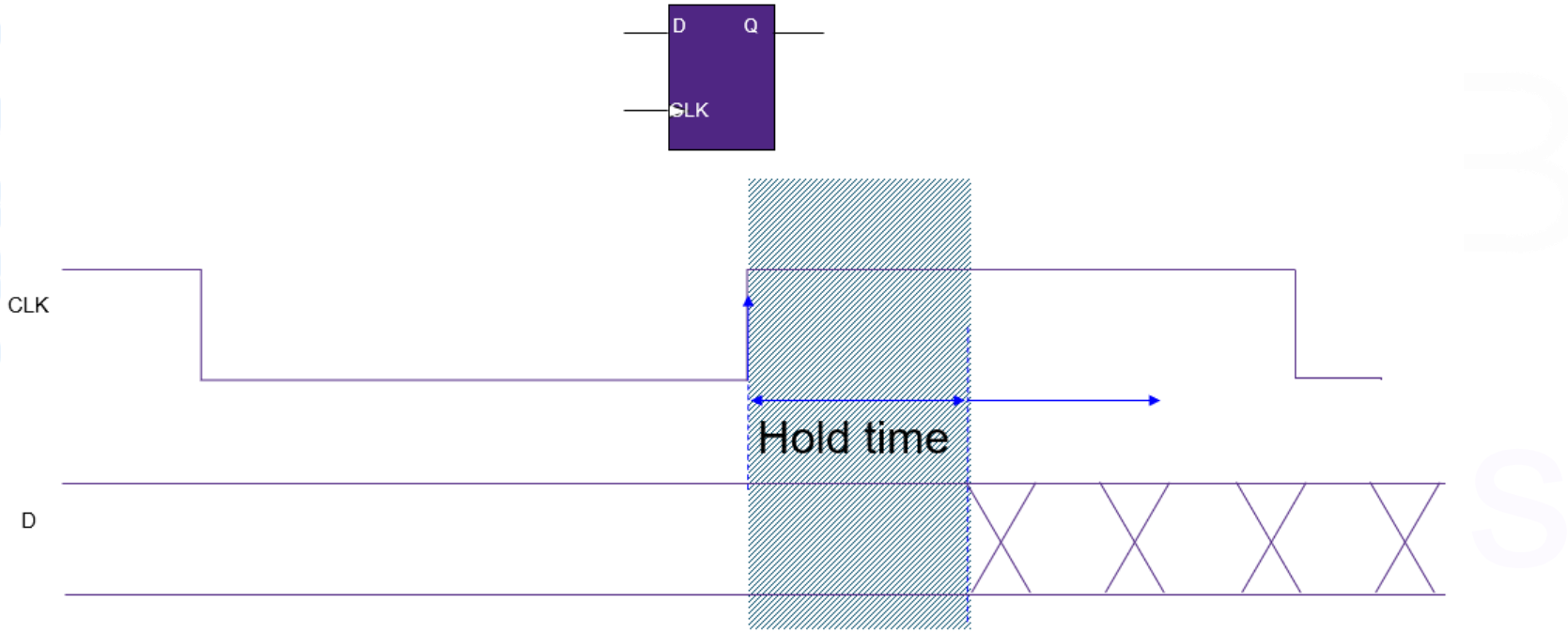
Setup condition



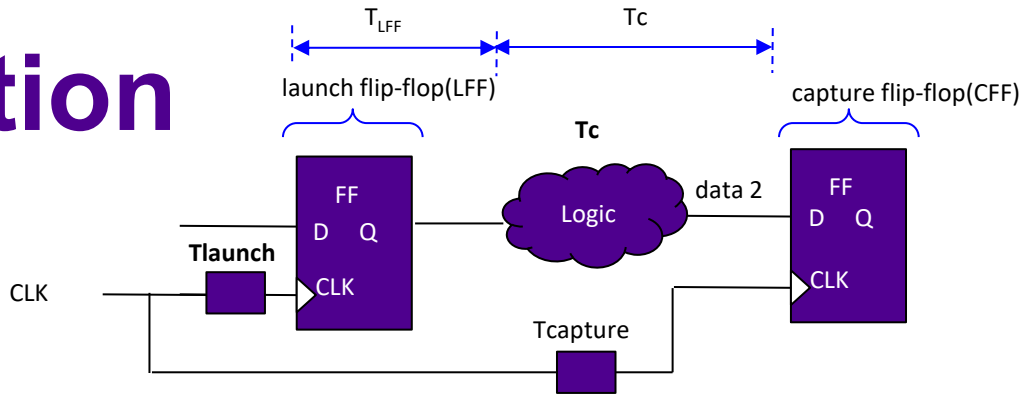
$$T_{launch} + T_{LFF} + T_c < T_{capture} + T_{cycle} - T_{setup}$$

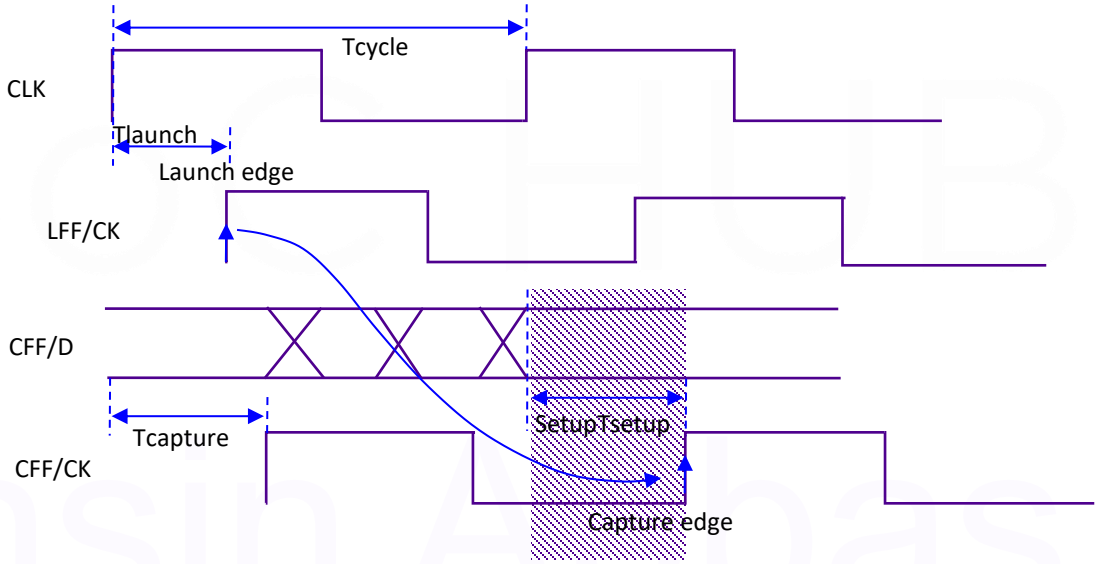
Hold Timing Check

- Verifies that the data is held stable for a specified amount of time after the active edge of the clock



Hold condition

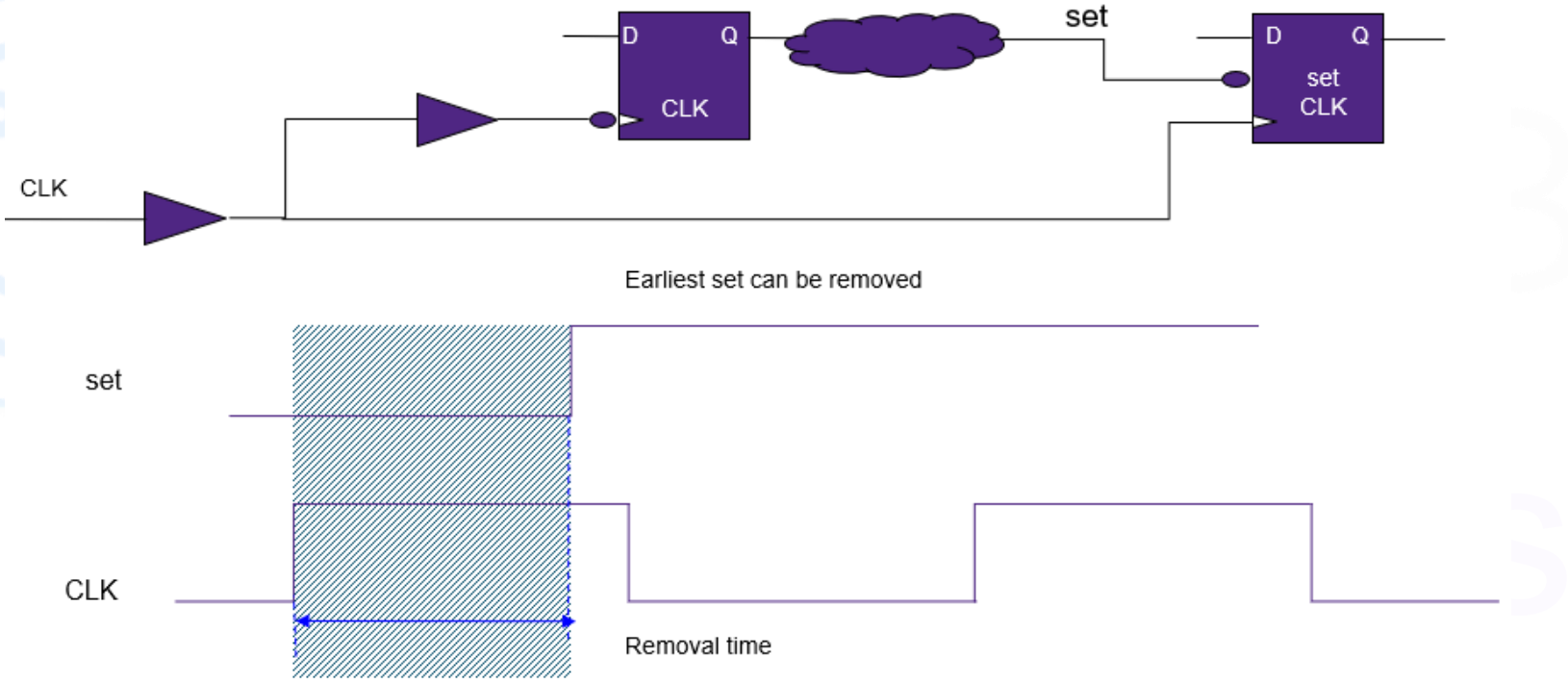




$$T_{launch} + T_{LFF} + T_c > T_{capture} + T_{hold}$$

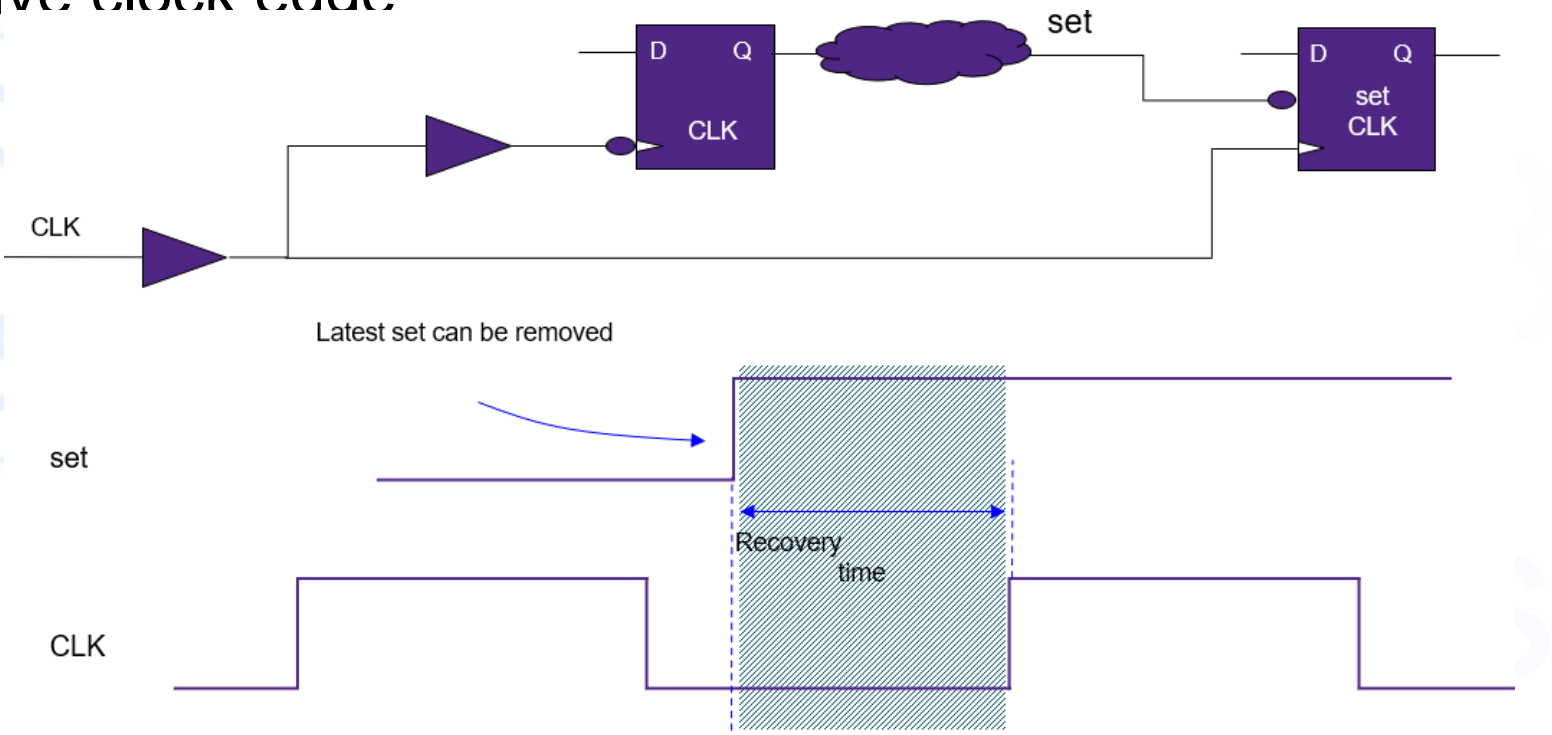
Removal Timing Check

- Verifies that there is required amount of time between an active clock edge and the release of an asynchronous control signal



Recovery Timing Check

- Verifies that there is a minimum amount of time between the asynchronous signal becoming inactive and the next active clock edge



Thanks

Next Lecture: Logical Synthesis