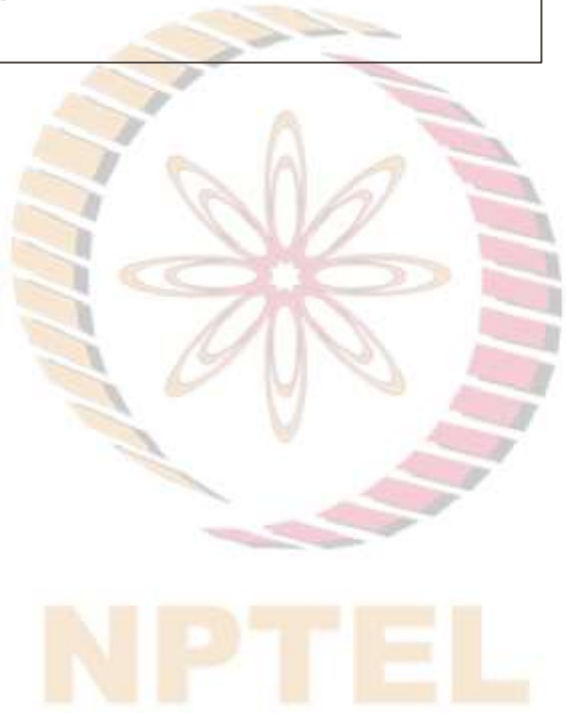
VLSI DESIGN FLOW: RTL TO GDS

Lecture 23
Static Timing Analysis – Part II

Sneh Saurabh
Electronics and Communications Engineering
IIIT Delhi

Lecture Plan

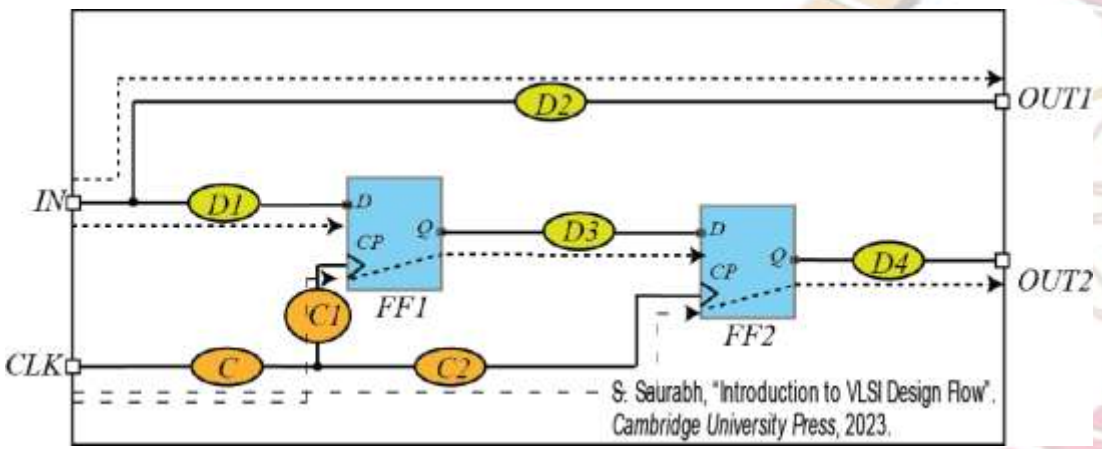
- Mechanism of Static Timing Analysis



Static Timing Analysis: Definitions

STA considers two types of paths:

1. Data Path
2. Clock Path



- Four data paths

- Two clock paths

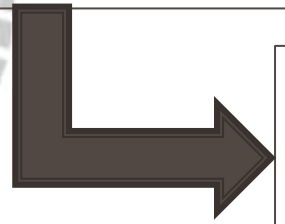
Data Path:

- **Timing startpoints**: input ports or clock-pin of a flip-flop
- Goes through combinational circuit elements
- **Timing endpoints**: D-pin of a flip-flop or output ports
  - ➢ Setup and hold checks are performed at the timing endpoints

Clock Path:

- Starts at clock source (specified in *constraints*)
- Passes through the combinational circuit elements (buffers and inverters)
- Ends at a clock-pin of a flip-flop

Static Timing Analysis



How it works

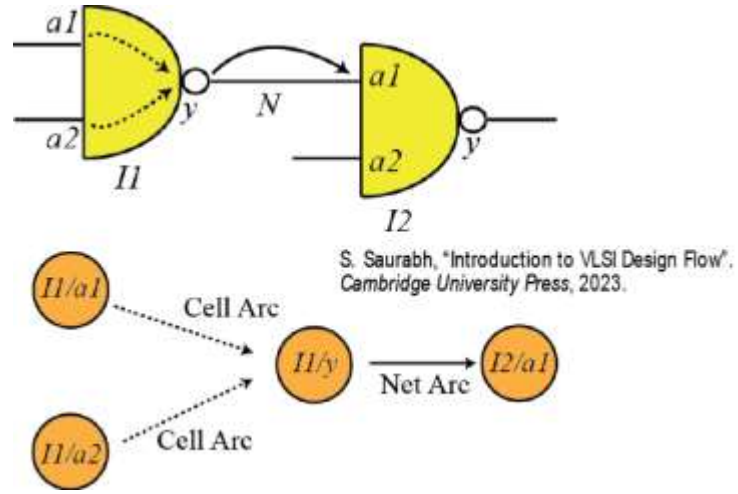
Timing Graph (1)

STA starts with building a *Timing Graph* for a given circuit

- Timing Graph: is a directed acyclic graph
  - $G = (V, E)$, where V is the vertex set and E is the edge set.
- The vertex $v \in V$ corresponds to a pin or a port in the circuit.
- An edge $e \in E$ represents a timing arc in the circuit
  - An edge $e_{i,j} = (v_i, v_j)$ exists in E if and only if there exists a timing arc between the corresponding pins or ports in the circuit.

Two types of edges:

- **Cell Arc**: Timing arc between two pins of the *same cell*
- **Net Arc**: Timing arc between two pins of *different cells* that are connected directly by a net



S. Saurabh, "Introduction to VLSI Design Flow", *Cambridge University Press*, 2023.

- Each edge $e_{i,j}$ has annotated information of delay $D_{i,j}$ (computed by *Delay Calculation)*
- Each vertex in the graph has arrival time, required time, slack, etc.

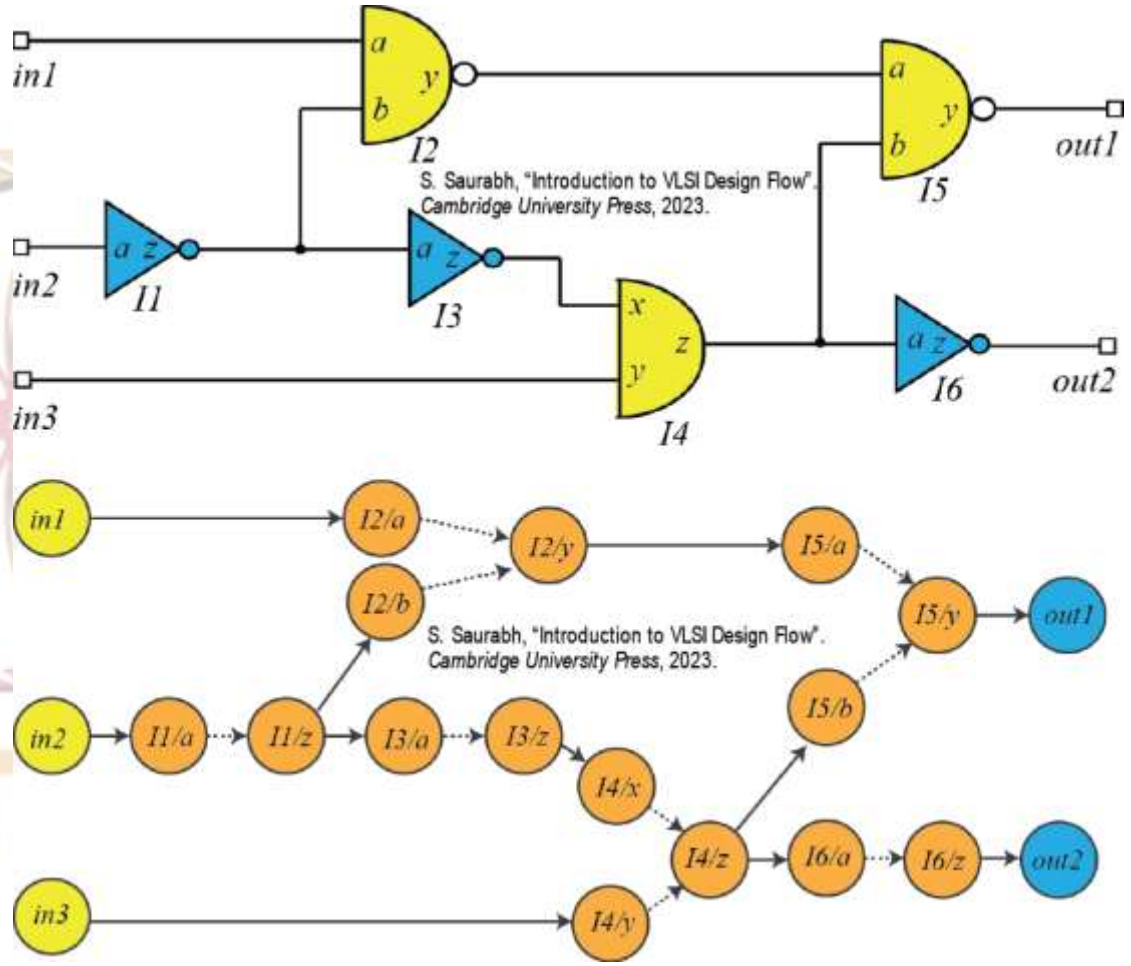
Timing Graph (2)

Source: vertices with no incoming edges

- Timing startpoints (input ports, clock-pin of FFs) and clock start points treated as sources

Sink: vertices with no outgoing edges

- Timing endpoints (output ports, D-pin of FFs) treated as sinks

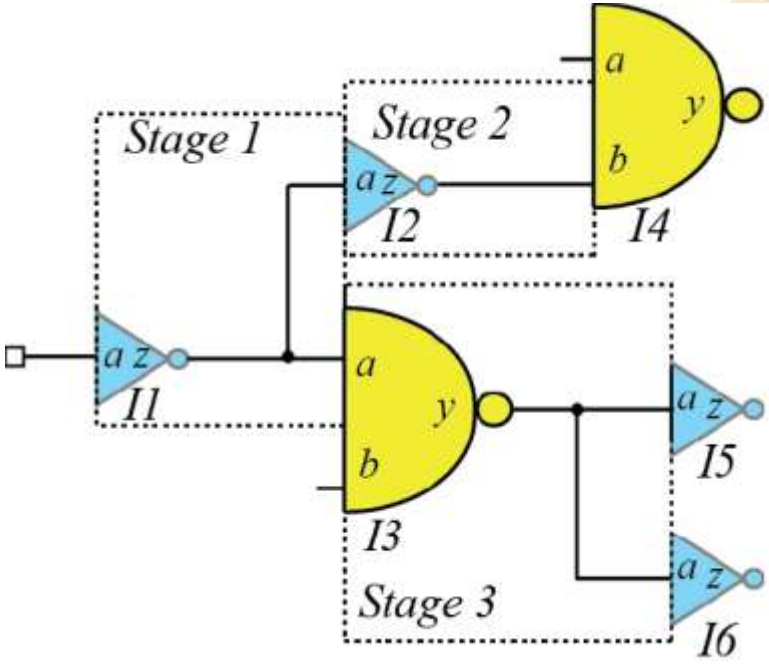


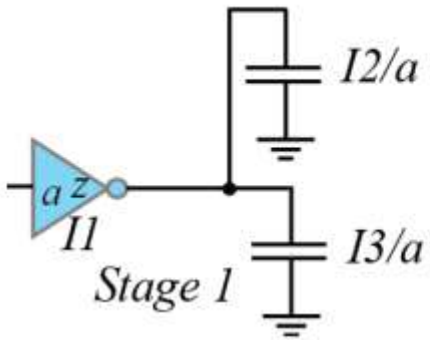
Delay Calculation: Stage

STA needs to know the delays for the timing arcs existing in the timing graph

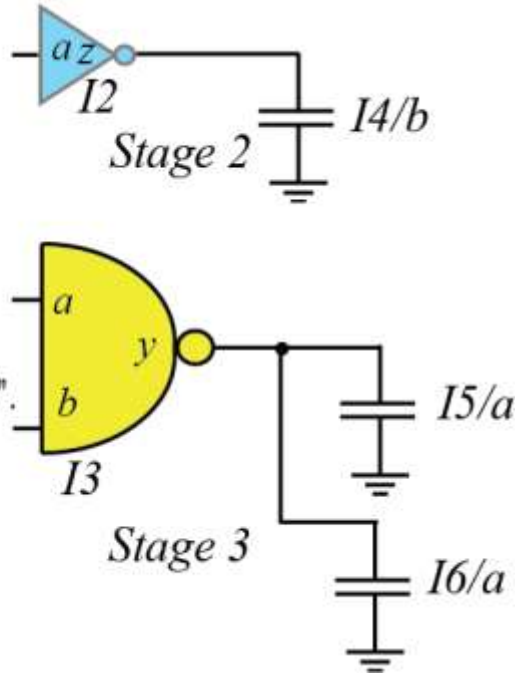
- Retrieves it from a **delay calculator**
- Inbuilt or coupled with an STA tool

- Decomposes a given circuit into separate stages
- A stage is composed of a driving cell and its driven pins connected through wires.

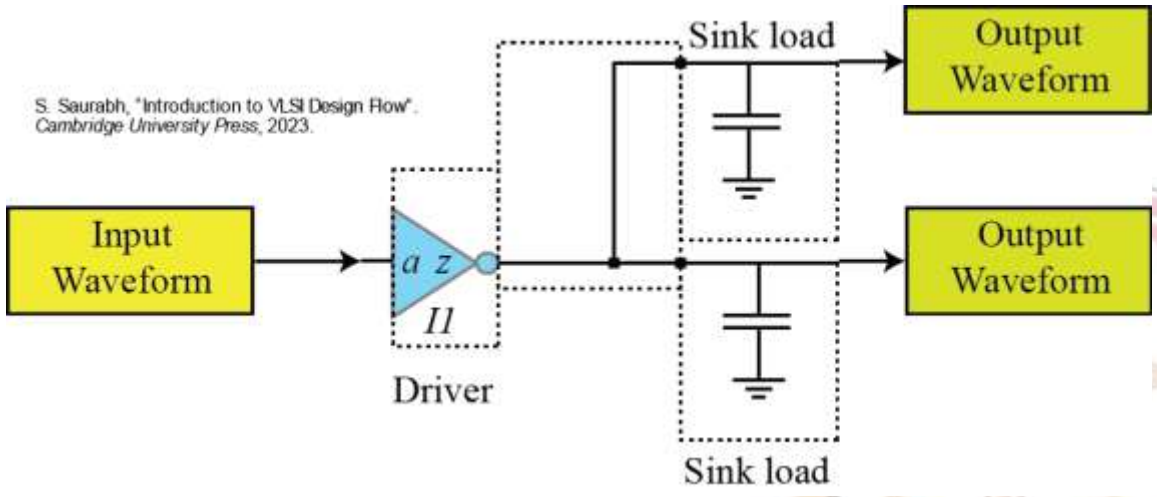




S. Saurabh, "Introduction to VLSI Design Flow". *Cambridge University Press*, 2023.



Delay Calculation: Essential Components



Driver model: NLDM, CCS, ECSM

Interconnect:

- Zero capacitance models, parasitic extraction
- Interconnect delay model: Lumped Capacitance, Elmore, Asymptotic Waveform Evaluation (AWE)

Receiver model: capacitance, or more advanced

- Given a stage, we can compute the output waveform once we know the input waveform

- Delay calculation can be done in topological order (from input to output)
- Transition at input ports obtained using SDC or assumed by a tool

- Each edge $e_{i,j}$ has annotated information of delay $D_{i,j}$ and slew $S_{i,j}$

Arrival Time Computation: Basic Concept

Arrival Time (AT): time at which a signal settles at a given vertex

When there is only one incoming edge: $A_j = A_i + D_{ij}$

When there are multiple incoming edges:

- Can associate different arrival time for different edges
- Value at a vertex can toggle multiple times before settling

- A bound on the arrival time at a given vertex v_j can be computed if the arrival times at all its input vertices v_i are known:
  - $\boldsymbol{A_{j,min} = Min(A_{i,min} + D_{ij})}$
  - $\boldsymbol{A_{j,max} = Max(A_{i,max} + D_{ij})}$

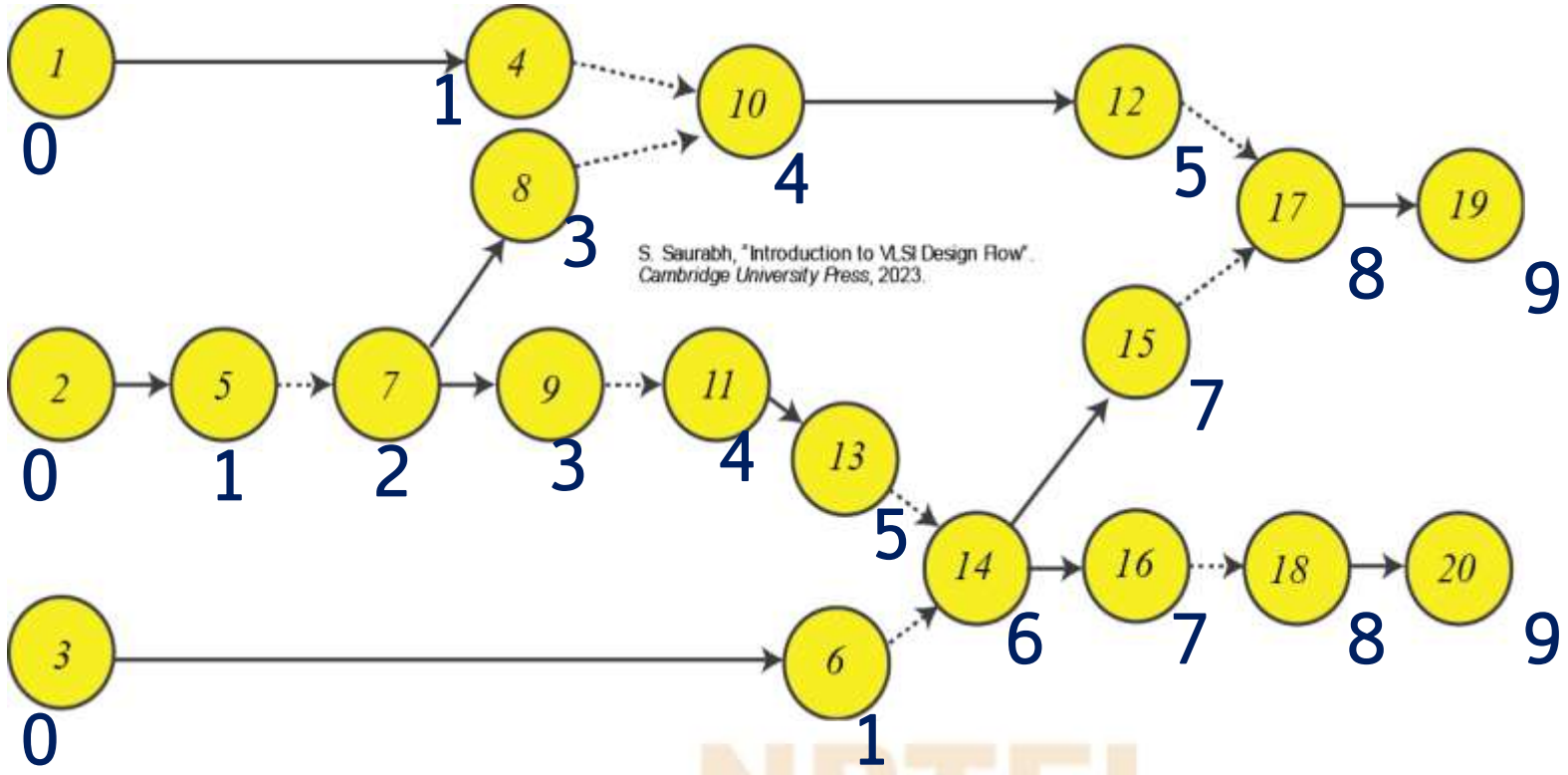
Arrival Time Computation: Method

- Arrival Time (AT) is computed and stored at each vertex in a timing graph
- AT at input ports specified by constraints or assumed to be zero

- AT computation is done by **Forward Traversal of Timing Graph**
- AT computation starts from the vertices corresponding to input ports

- A vertex is chosen for computing AT such that:
  - **All the input vertices of the given vertex have their AT already computed**
- AT can be computed in one traversal of the vertices and edges of the timing graph

Maximum Arrival Time Computation: Illustration



- Assume that delay of all the edges = 1 time unit
- Assume that AT is 0 at all sources

- Similarly, minimum arrival time computation can be done

Arrival Time Computation : Complications

- Rise/Fall Delays:
  - Separate AT is computed for rise/fall cases

- Dependence of delay on the input slew (rise/fall transition time)
  - **Slew is also propagated and stored at each vertex, in addition to delay**

Required Time Computation: Basic Concept

Required Time (RT): time constraint for a given vertex to avoid setup/hold violation

- Setup/Late analysis: the **maximum time by which** a signal should arrive to avoid violation
- Hold/Early analysis: the **minimum time after which** a signal should arrive to avoid timing violation

- Required time for a vertex v_i can be computed if required times at all its output vertices v_j are known:
  - $\boldsymbol{R_{i,hold} = Max(R_{j,hold} - D_{ij})}$
  - $\boldsymbol{R_{i,setup} = Min(R_{j,setup} - D_{ij})}$

Required Time Computation: Method

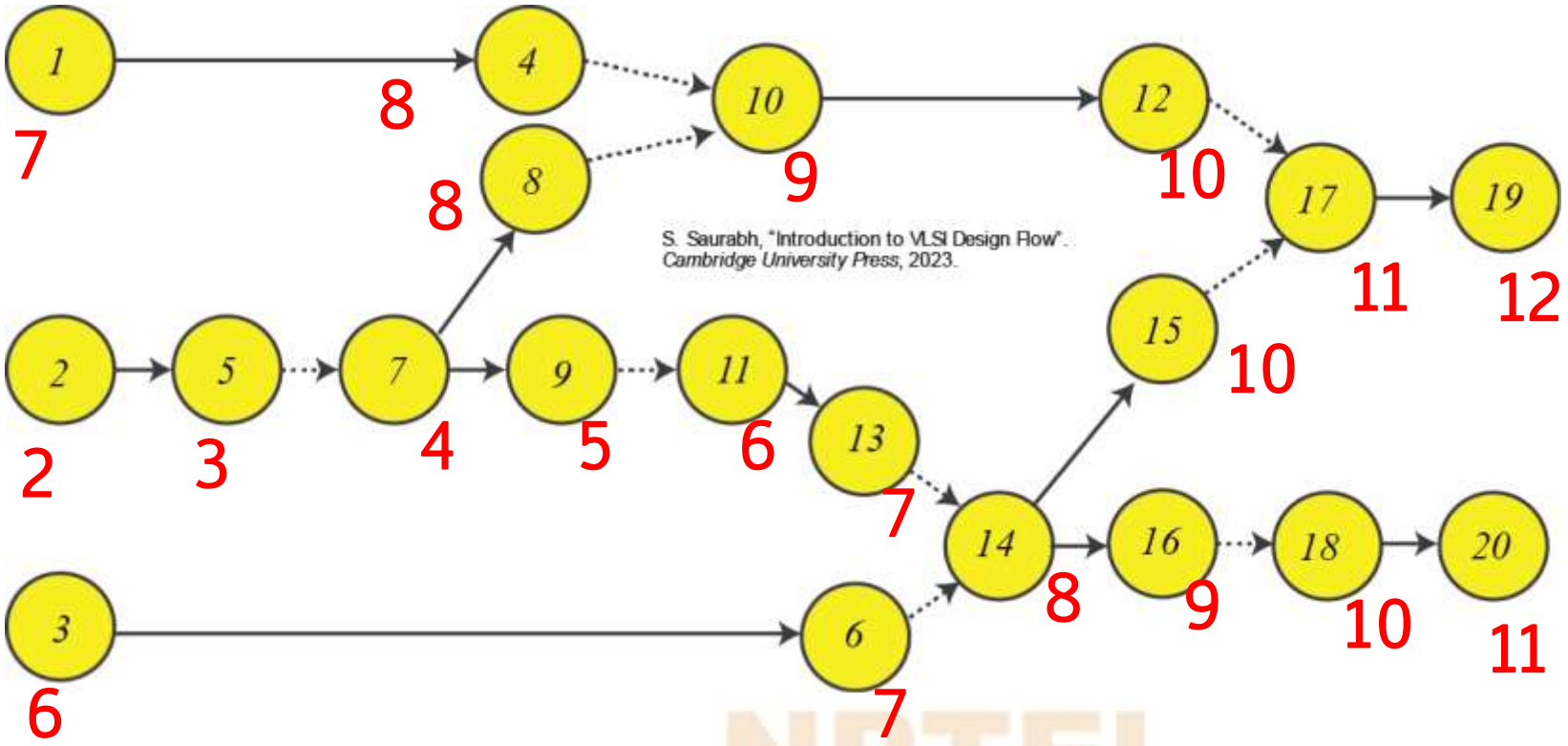
- At each vertex in the Timing Graph Required Time (RT) is computed and stored
- RT at output ports or timing end-points is specified or inferred from constraints

- RT computation is done by **Backward Traversal of Timing Graph**
- RT computation starts from the vertices corresponding to output ports

- A vertex is chosen for computing RT such that:
  - **All the output vertices of the given vertex have their RT already computed**
- RT can be computed in one traversal of the vertices and edges of the timing graph

- Delay of cell/net arcs calculated during AT computation is reused in RT computation
- RT constraints primarily determined by clock period

Required Time Computation: Setup Analysis



- Assume that delay of all the edges = 1 time unit
- Assume that RT is 11 and 12 as shown at the sinks

- Similarly, required time for hold analysis can be computed

Slack Computation

Setup or Late Analysis:

- $Slack = RT - AT$
- Slack is the time by which AT at a vertex can be increased *without causing setup violation*
  - ➢ AT can be increased till slack becomes zero

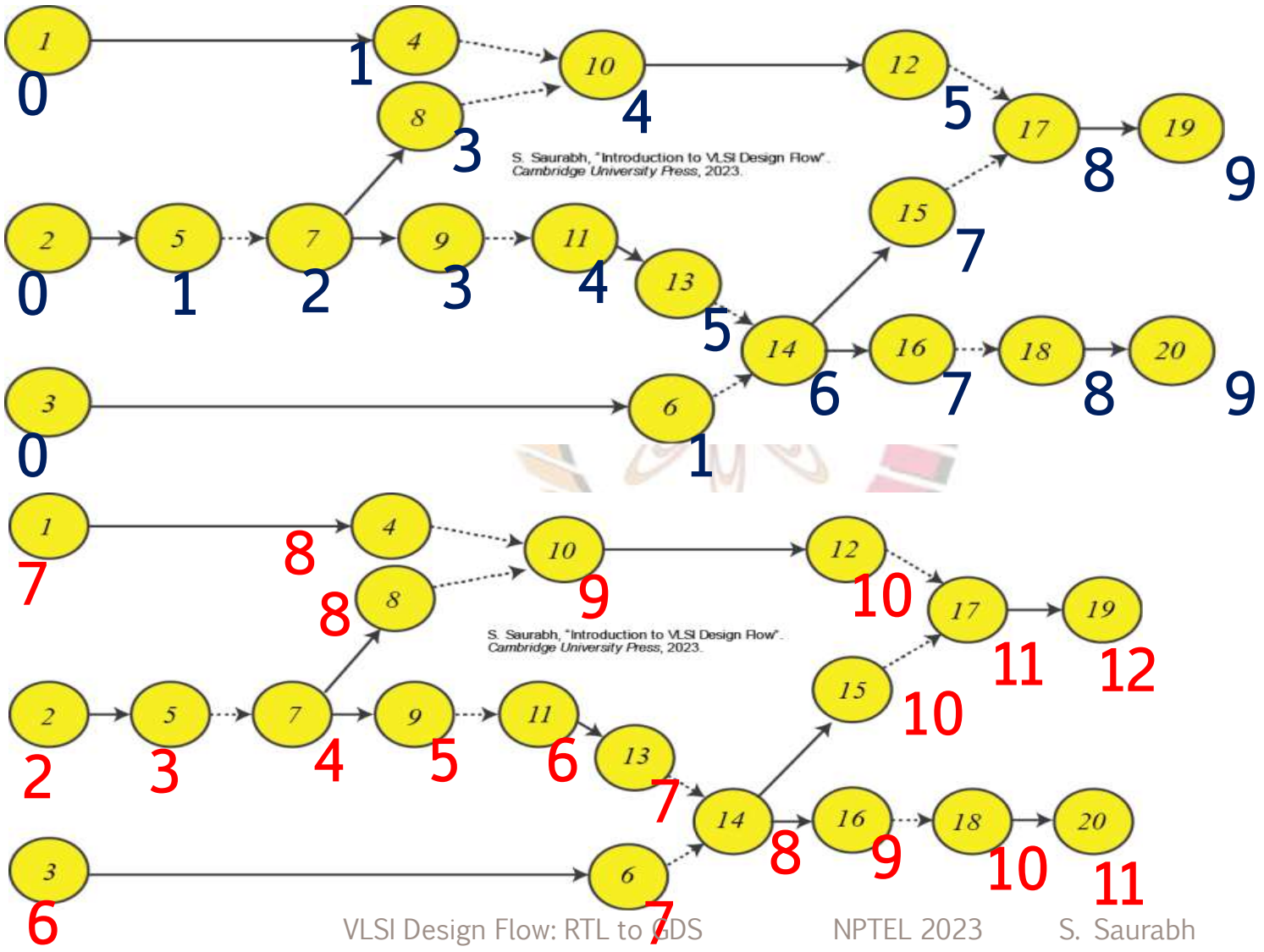
Hold or Early Analysis:

- $Slack = AT - RT$
- Slack is the time by which AT at a vertex can be decreased *without causing hold violation*
  - ➢ AT can be decreased till slack becomes zero

A negative slack implies that:

- Setup/Hold violation exists in the circuit
- Need to fix the circuit for proper operation

Slack Computation: Illustration



References

- S. Saurabh, “Introduction to VLSI Design Flow”. Cambridge: *Cambridge University Press*, 2023.
- Bhasker, Jayaram, and Rakesh Chadha. *Static timing analysis for nanometer designs: A practical approach*. Springer Science & Business Media, 2009.