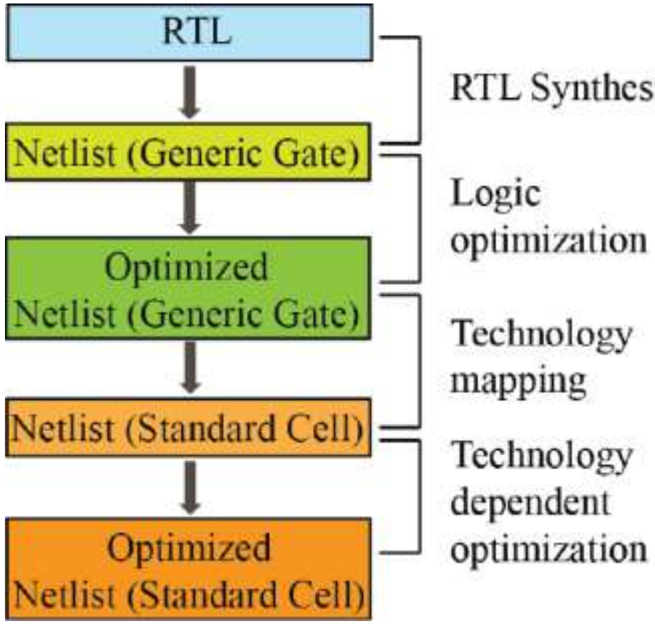
VLSI DESIGN FLOW: RTL TO GDS

Lecture 16

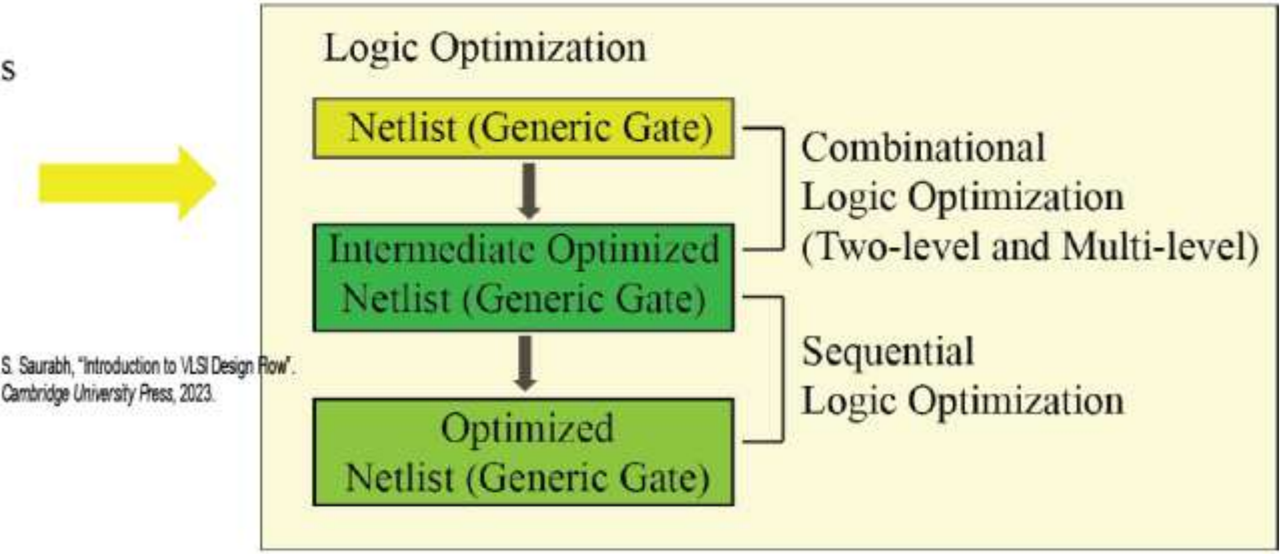
Logic Optimization: Part III

Sneh Saurabh
Electronics and Communications Engineering
IIIT Delhi

Lecture Plan


RTL
Netlist (Generic Gate)
Optimized Netlist (Generic Gate)
Netlist (Standard Cell)
Optimized Netlist (Standard Cell)
RTL Synthesis
Logic optimization
Technology mapping
Technology dependent optimization

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


Logic Optimization
Netlist (Generic Gate)
Intermediate Optimized Netlist (Generic Gate)
Optimized Netlist (Generic Gate)
Combinational Logic Optimization (Two-level and Multi-level)
Sequential Logic Optimization

- Sequential Logic Optimization

Sequential Logic Optimization

Finite State Machine (FSM) : Definition

- FSM is an abstract mathematical model to represent wide variety of sequential circuits and systems

An FSM consists:

- Finite non-empty sets of ***states*** (S)
- Finite non-empty sets of inputs (I) and outputs (O)
- Given initial state (s_0)

An FSM consists:

- State transition function (δ)
  - Given the current state and current input, it produces the next state

Output function can be modelled in two ways in an FSM:

- Output is a function of the current state only: *Moore Machine*
- Output is a function of the current state and current input: *Mealy Machine*

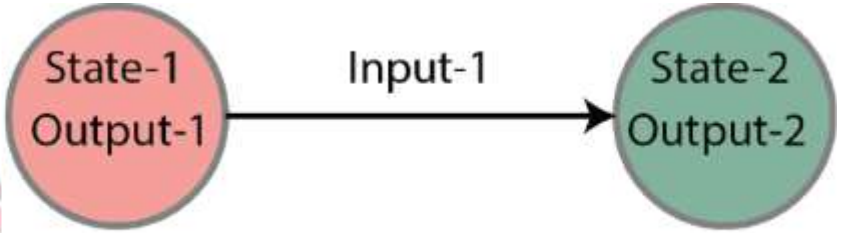
Finite State Machine (FSM) : State Diagram

- We can represent an FSM pictorially using a directed graph called ***state diagram***

State diagram is a directed graph:

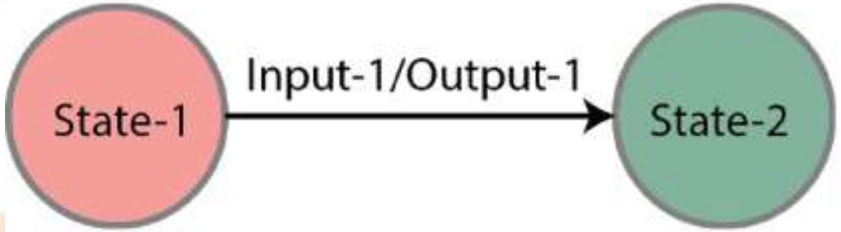
- A state corresponds to a vertex in the graph
- Transition from one state to another state is represented as an ***edge*** in the graph
  - ➢ Edge starts from the current state and ends in the next state
  - ➢ Mark inputs at the edge

- Mark output value:
  - ➢ Inside the state for Moore machine
  - ➢ At the edges for the Mealy machine.



Moore Machine

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



Mealy Machine

Finite State Machine (FSM): Example-1

A fan regulator can be described as an FSM:

- Four settings of regulator described as four states
  - $S = \{s0, s1, s2, s3\}$
  - $s0$ can be taken as an initial state

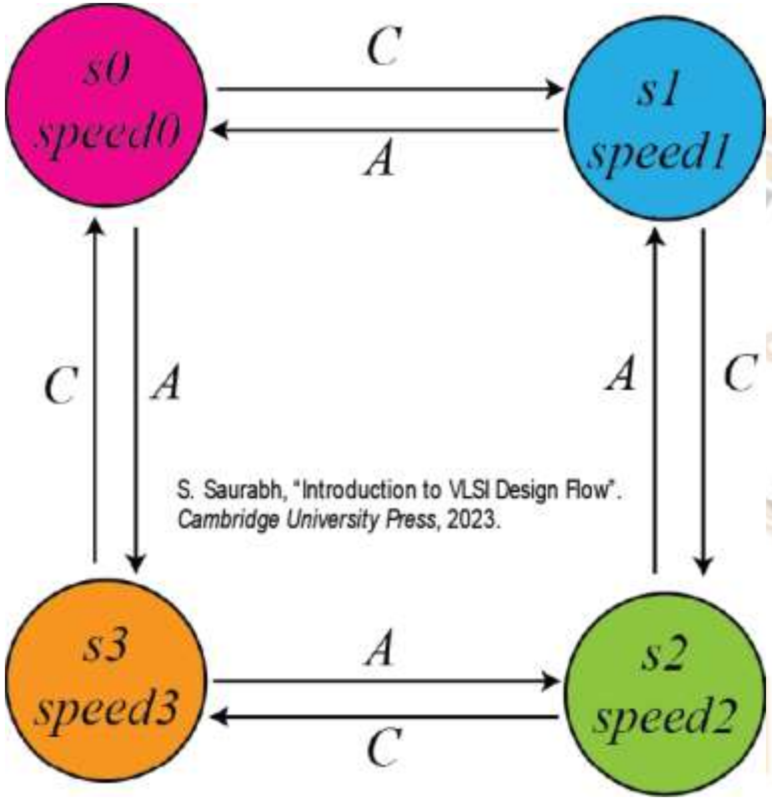
- Inputs to the regulator (rotate clockwise and rotate anti-clockwise) are inputs of the FSM
  - $I = \{C, A\}$
- Four different speeds at four settings are outputs of the FSM
  - $O = \{speed0,\ speed1,\ speed2,\ speed3\}$

Transitions:

- Rotate clockwise successively: $s0 \rightarrow s1 \rightarrow\ s2 \rightarrow s3 \rightarrow s0 \ldots$
- Rotate anti-clockwise successively: $s0 \rightarrow s3 \rightarrow\ s2 \rightarrow s1 \rightarrow s0 \ldots$

We can draw the state diagram for this FSM.

Finite State Machine (FSM): Example-1



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

Which kind of FSM is this?

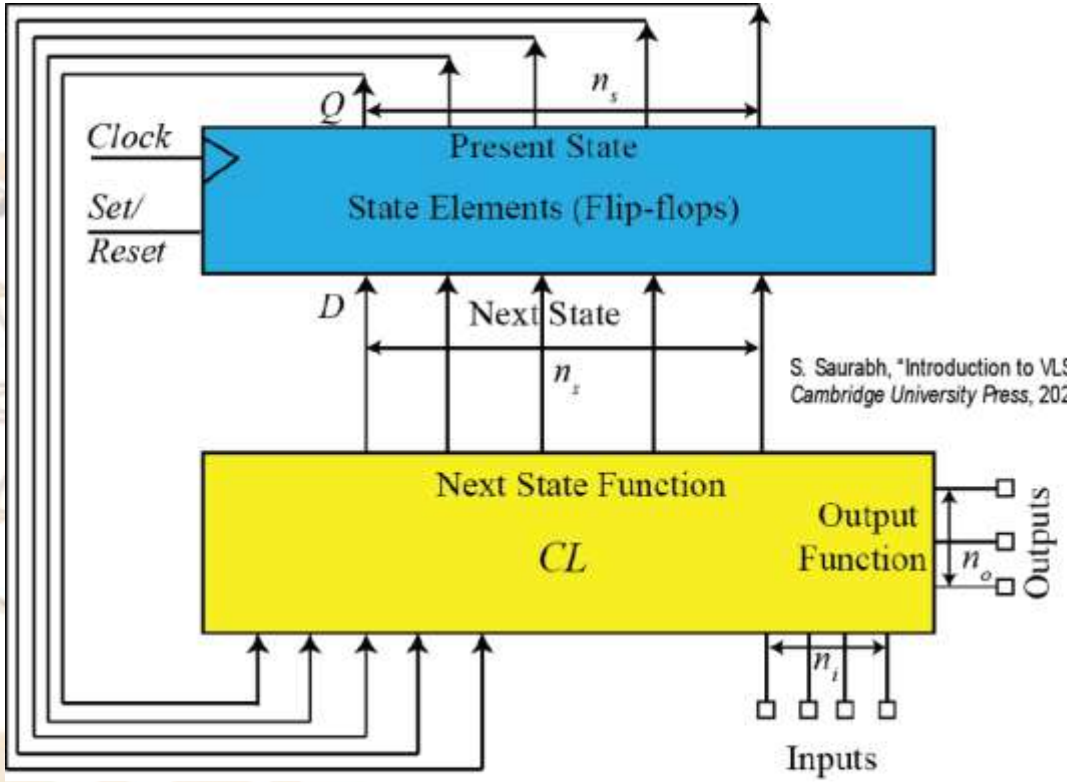
Ans: Moore FSM.

Finite State Machine (FSM) Implementation

- FSM is implemented using flip-flops and combinational circuit elements

- States are represented by the combination of the Q-pin value of flip-flops.
- Transition function is implemented using combinational logic

- Cost (area) of FSM implementation can be reduced by:
    - ➢ Reducing number of flip-flops (state minimization)
    - ➢ Reducing logic complexity of combinational circuit elements (state encoding)



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

State Minimization

State Minimization: Basics

Motivation for state minimization:

- To represent a state we use a sequence of bits
  - For n_s states, minimum $[log_2 n_s]$ bits required
- By reducing number of states n_s, we can reduce number of bits in FSM representation
  - Reduce number of flip-flops in FSM implementation

Objective of state minimization:

- To derive an FSM that has the minimum number of states and exhibits the same behavior as the original FSM
- Relies on determining **equivalent states**

Two states of an FSM are equivalent :

1. They produce identical outputs
2. The corresponding next states are the same or equivalent.

- Among set of equivalent states, retain any one of the equivalent states and remove others
- Update the transition function to maintain the same behavior as the original FSM.

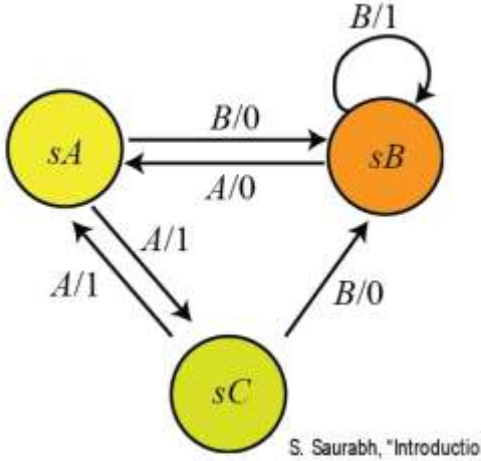
State Minimization: Illustration

Consider an FSM shown: inputs $I = \{A, B\}$ and outputs $O = \{0, 1\}$.

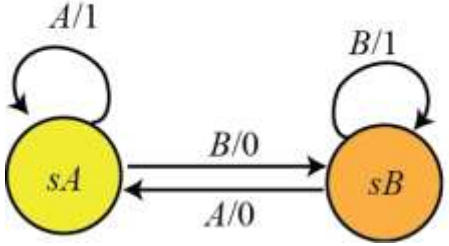
Consider the states sA and sC :

- Outputs:
  - When input is A, both states produce 1
  - When input is B, both states produce 0.
- Next States:
  - When input is A, sA transitions to sC (equivalent state) and sC transitions to sA (equivalent state).
  - When the input is B, both sA and sC transitions to sB.
- The states sA and sC are equivalent

- Efficient algorithms exist to find set of equivalent states.



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



State Encoding

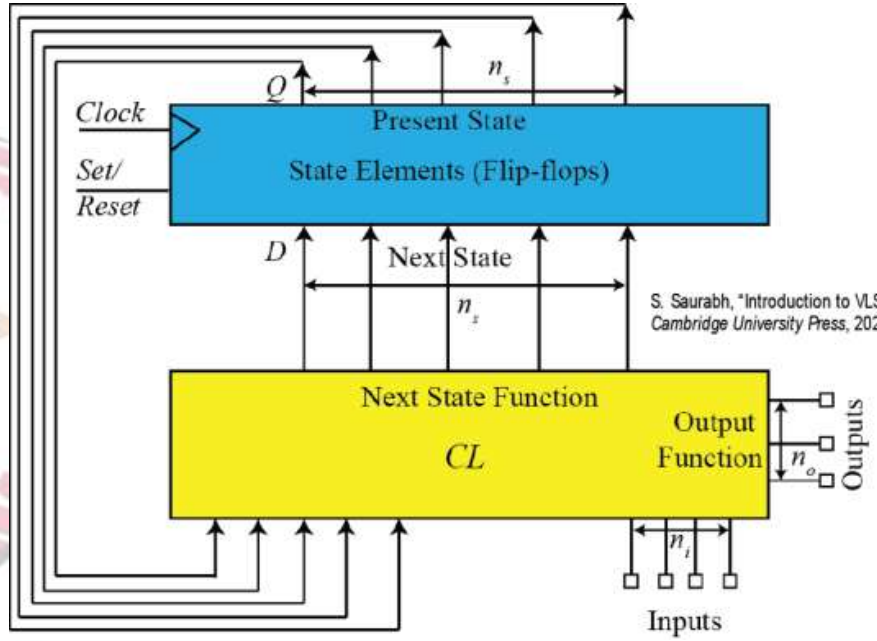
State Encoding

State encoding:

- It assigns a binary representation to each state of an FSM.

Impact of state encoding:

- Changing the state encoding, can change the next state function and the output function
- Combinational circuit block CL and the associated QoR can change.



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

Encoding Length

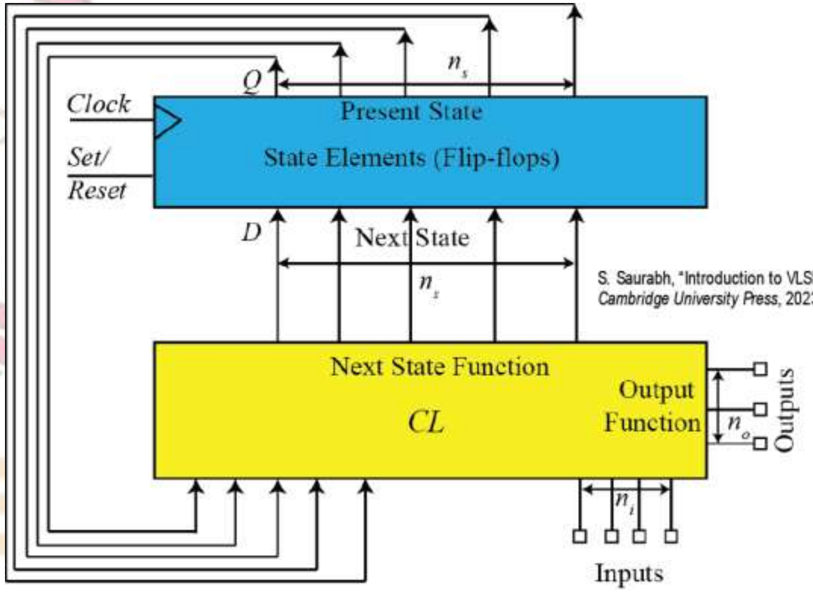
Encoding length:

- To represent an FSM with n_s states, we need at least $[log_2 n_s]$ bits
- Longer encoding lengths can also be chosen
  - Example: one-hot encoding reserves one bit for each state (encoding length is n_s)

Impact of encoding length:

- Number of flip-flops
- Inputs/Outputs of CL

- Sometimes CL can be simplified by changing encoding length.
  - ➢ One-hot encoding: identifying a state requires examining only one state bit.
  - ➢ One-hot encoding can have fewer logic levels between flip-flops and be faster



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

Choosing Encoding Scheme

Challenges:

- Exponential number of possibilities
  - ➢ Brute force cannot be applied to examine all solutions.
- During encoding, CL implementation is not done
  - ➢ Need to assess the encoding quality based on the expected FSM implementation.

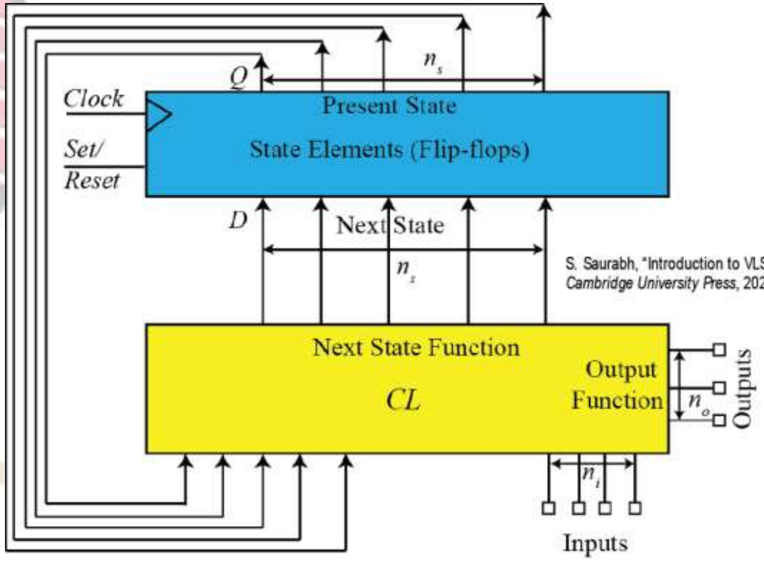
Approach:

- CL needs to produce the next state functions and output functions
  - ➢ Encode such that they share logic
  - ➢ Allow more common cubes and common sub-expressions

Heuristics-based Algorithms:

➢ Quantify the possibility of common cube extraction as "gain" of an encoding scheme

➢ Determine encoding that maximizes the "gain"



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

References

- S. Saurabh, “Introduction to VLSI Design Flow”. Cambridge: *Cambridge University Press*, 2023.
- G. D. Micheli. “Synthesis and Optimization of Digital Circuits”. *McGraw-Hill Higher Education*, 1994.
- S. Devadas, H.-K. Ma, A. R. Newton, and A. Sangiovanni-Vincentelli. “MUSTANG: State assignment of finite state machines targeting multilevel logic implementations.” *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems* 7, no. 12 (1988), pp. 1290–1300.