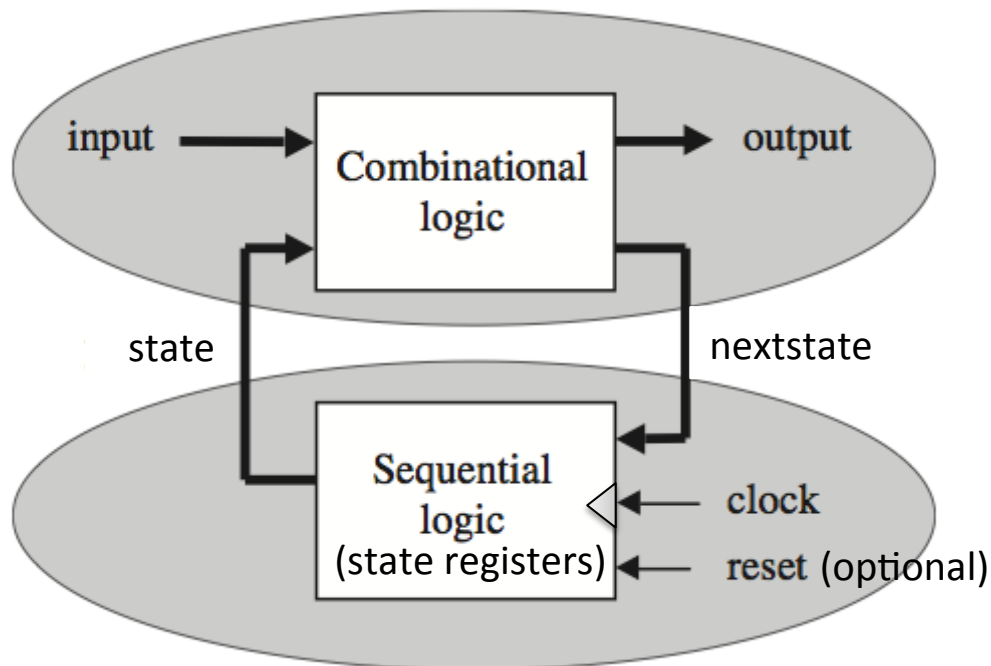


Finite State Machines

FSM



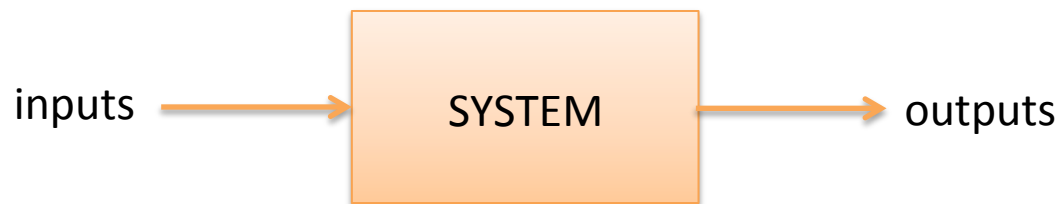
Generic State Machine Model

Guidelines for coding FSMs in VHDL:

- * Use separate processes for sequential logic and combinational logic
- * Use enumerated data type to list all possible states

State Machine: key idea

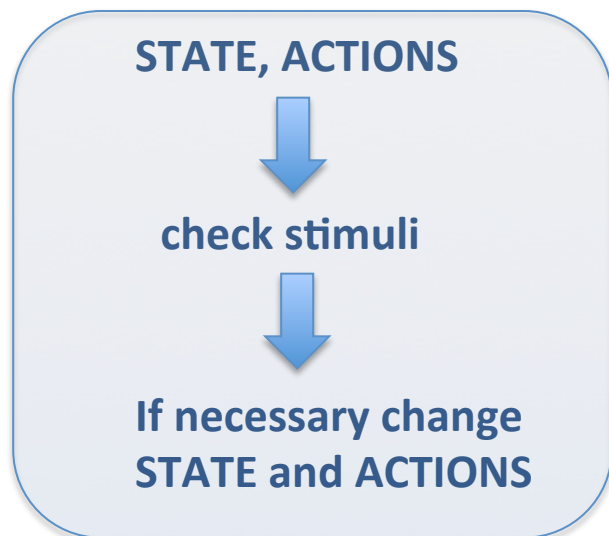
- State machines are an effective mathematical MoC that allow to characterize in an un-ambiguous and formal way the behavior of a system



- Goal: given a set of external stimuli we want to design a system that exhibit a desired behavior, i.e. the system must be able to process the **stimuli** provided at its inputs to produce certain **actions** at its outputs

State Machine: key idea

System Behavior:



In summary:

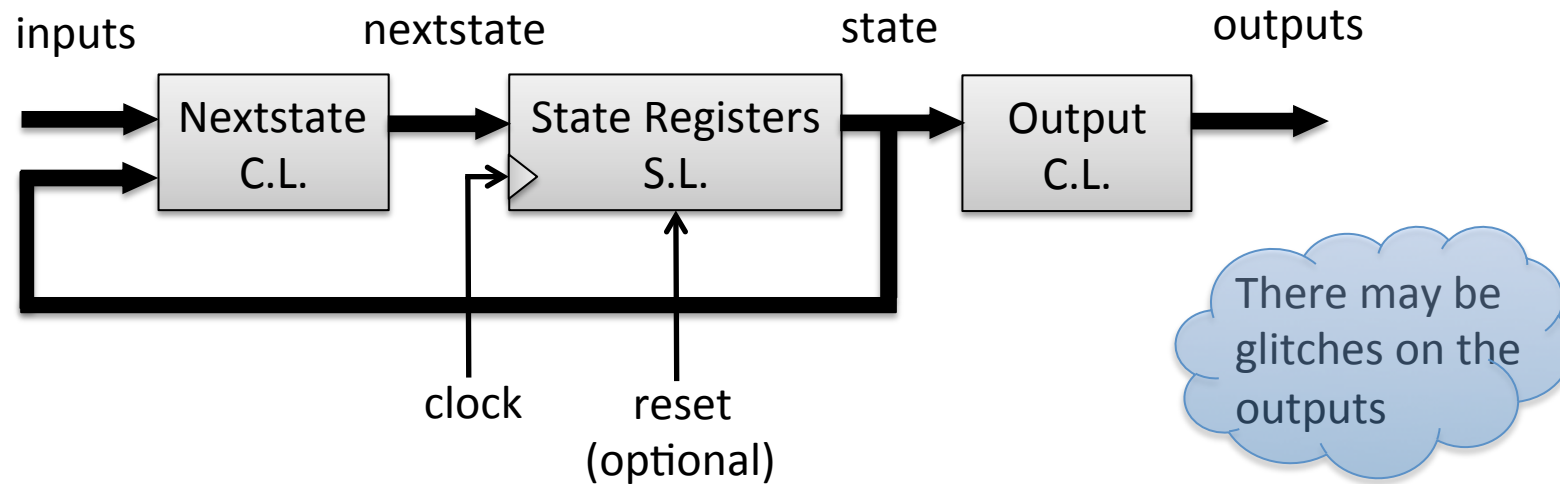
1. at any given time you are in a certain state, and perform certain actions
2. monitor external stimuli and “decide” what next state and actions should be

stimuli ↔ inputs

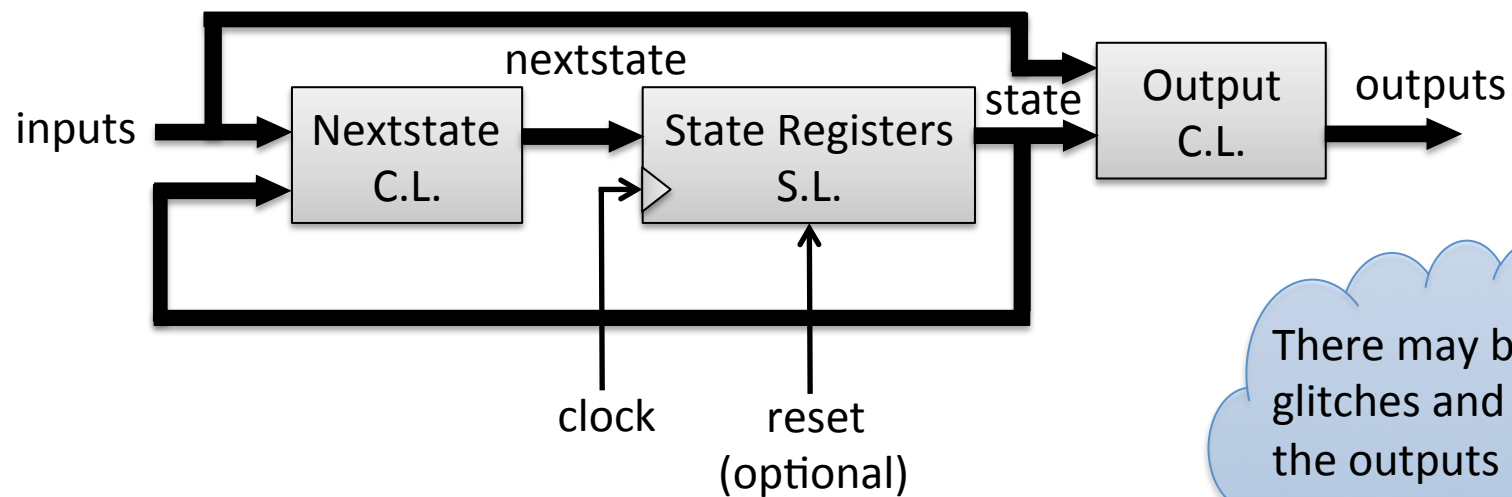
actions ↔ outputs

actions depends on stimuli & state

Moore style FSM



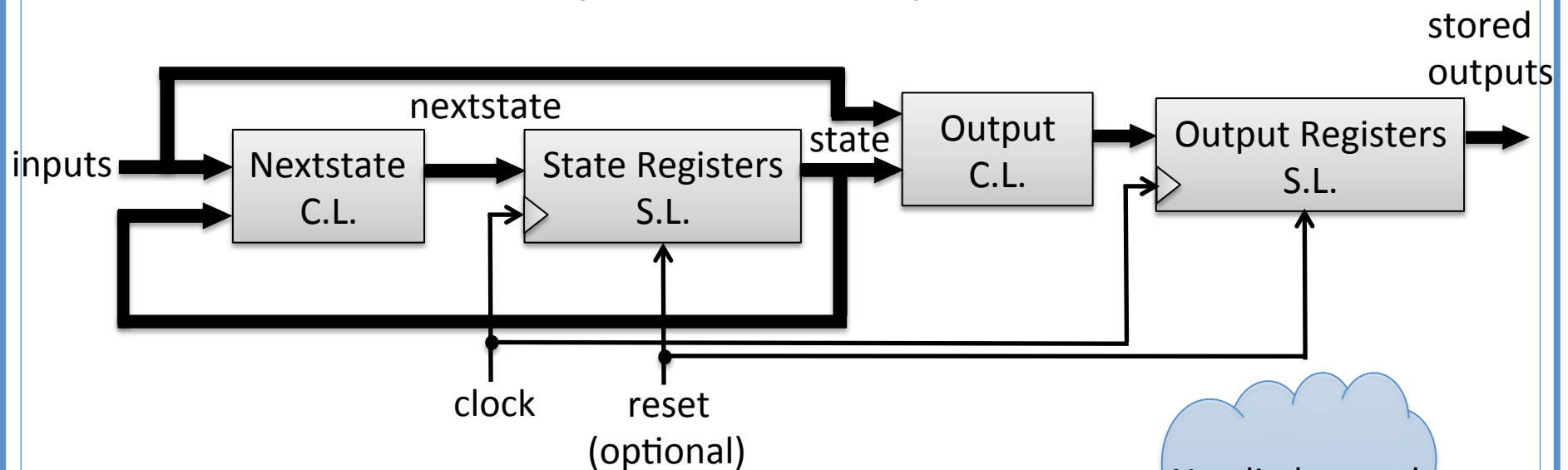
Mealy style FSM



There may be glitches and the outputs may last less than one cycle

Stored Outputs FSMs

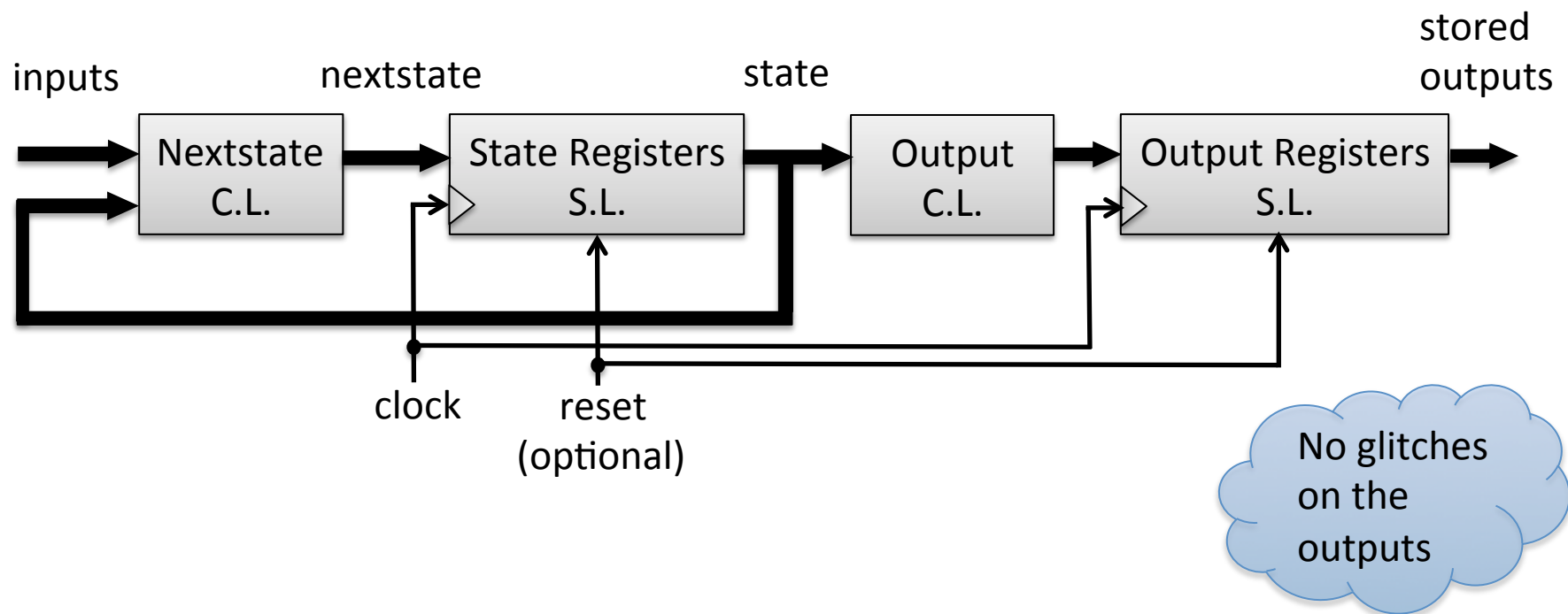
Pipelined Mealy



No glitches and
the outputs
last one cycle

Stored Outputs FSMs

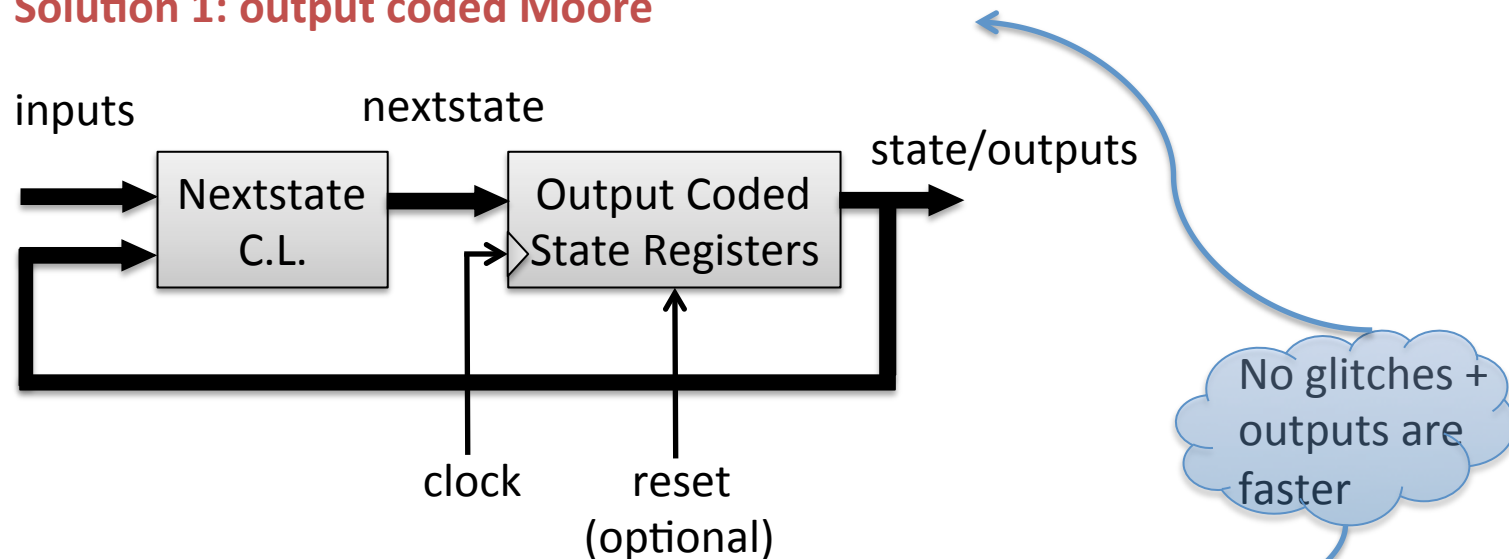
Pipelined Moore



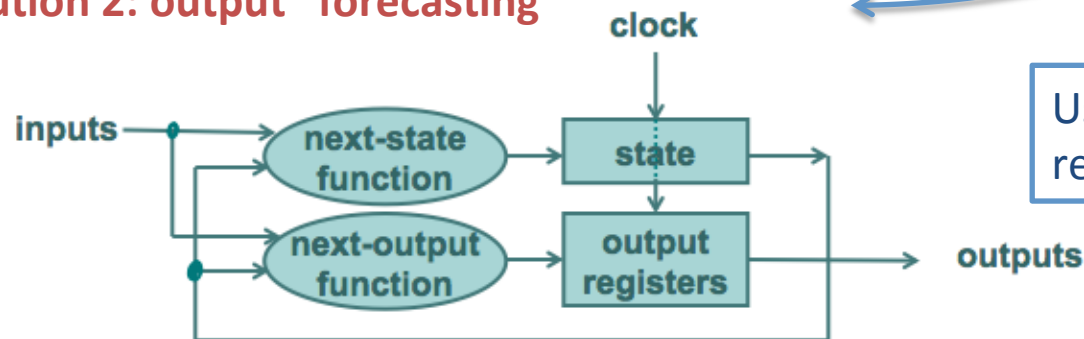
Stored outputs FSMs

NOTE: with Moore FSMs is possible to get registered outputs without having to add a pipeline stage → outputs get speed up by one clock cycle !!!

Solution 1: output coded Moore



Solution 2: output "forecasting"



Usually solution 2 requires less thinking