Computer Science 141
Computing Hardware

Fall 2009

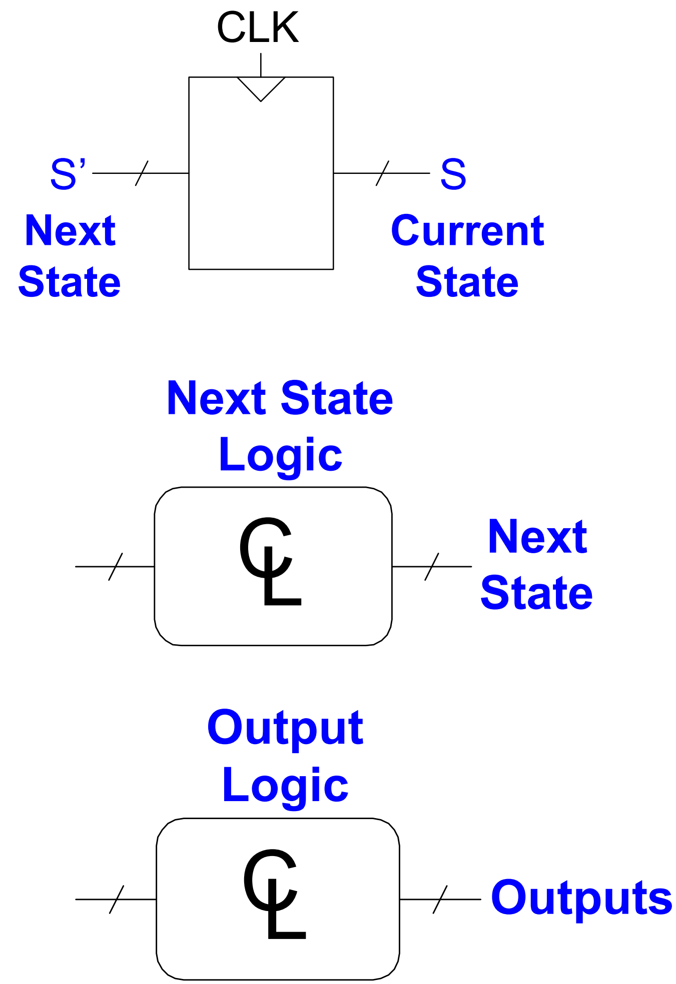
Harvard University

Instructor: Prof. David Brooks

dbrooks@eecs.harvard.edu

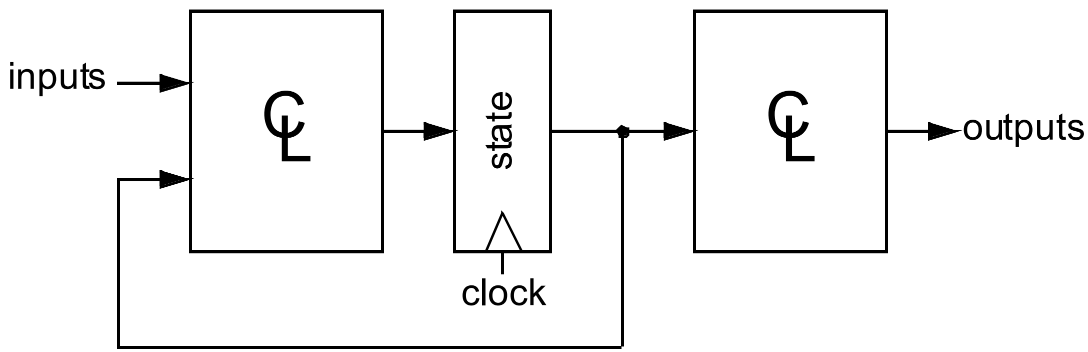
Finite State Machine (FSM)

- Consists of:
  - State register that
    - Store the current state and
    - Load the next state at the clock edge
  - Combinational logic that
    - Computes the next state
    - Computes the outputs

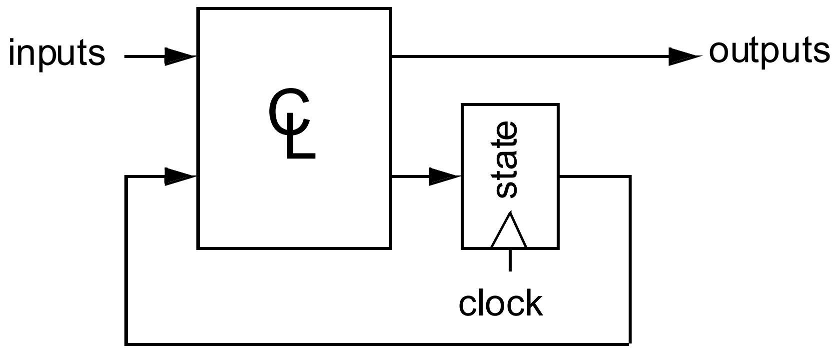


Moore vs. Mealy machines

- block diagrams

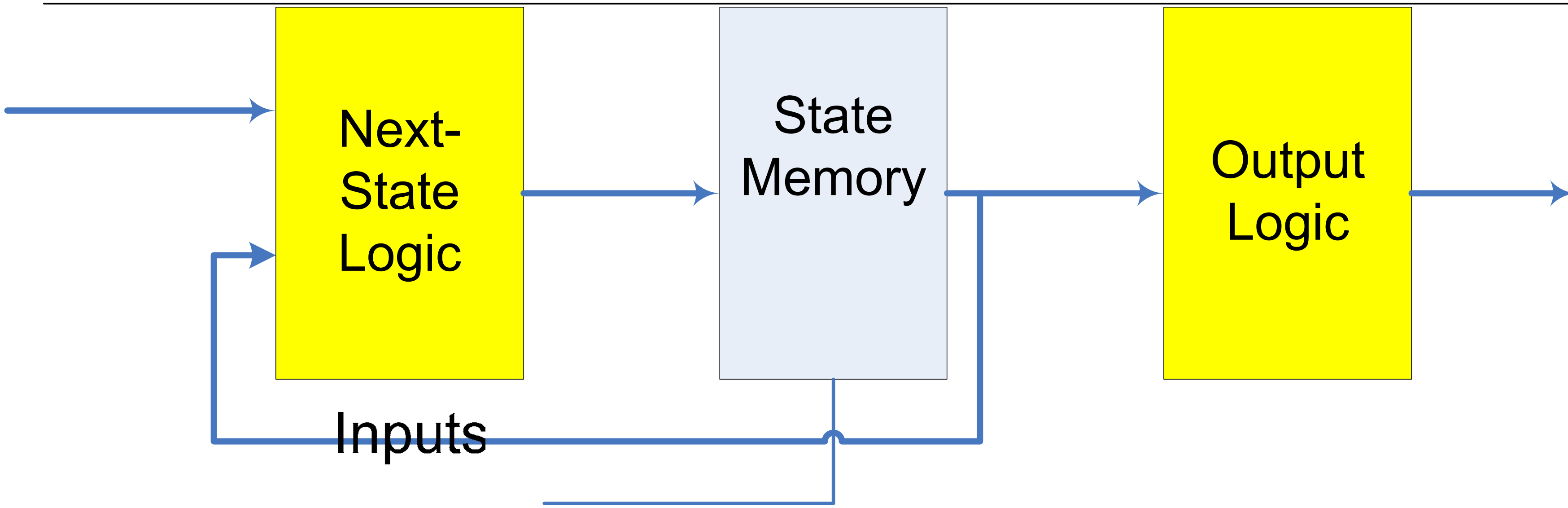


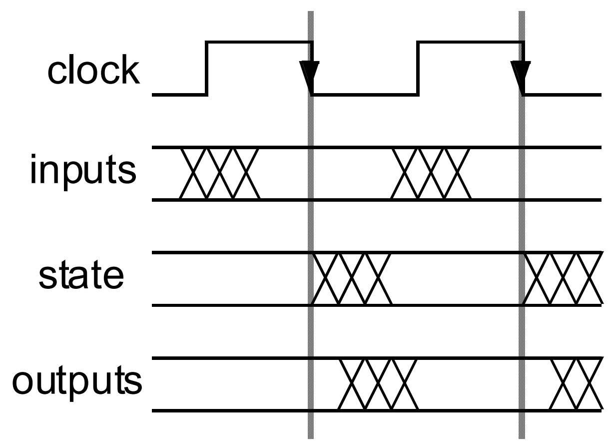
Moore machine



Mealy machine

Moore Machine

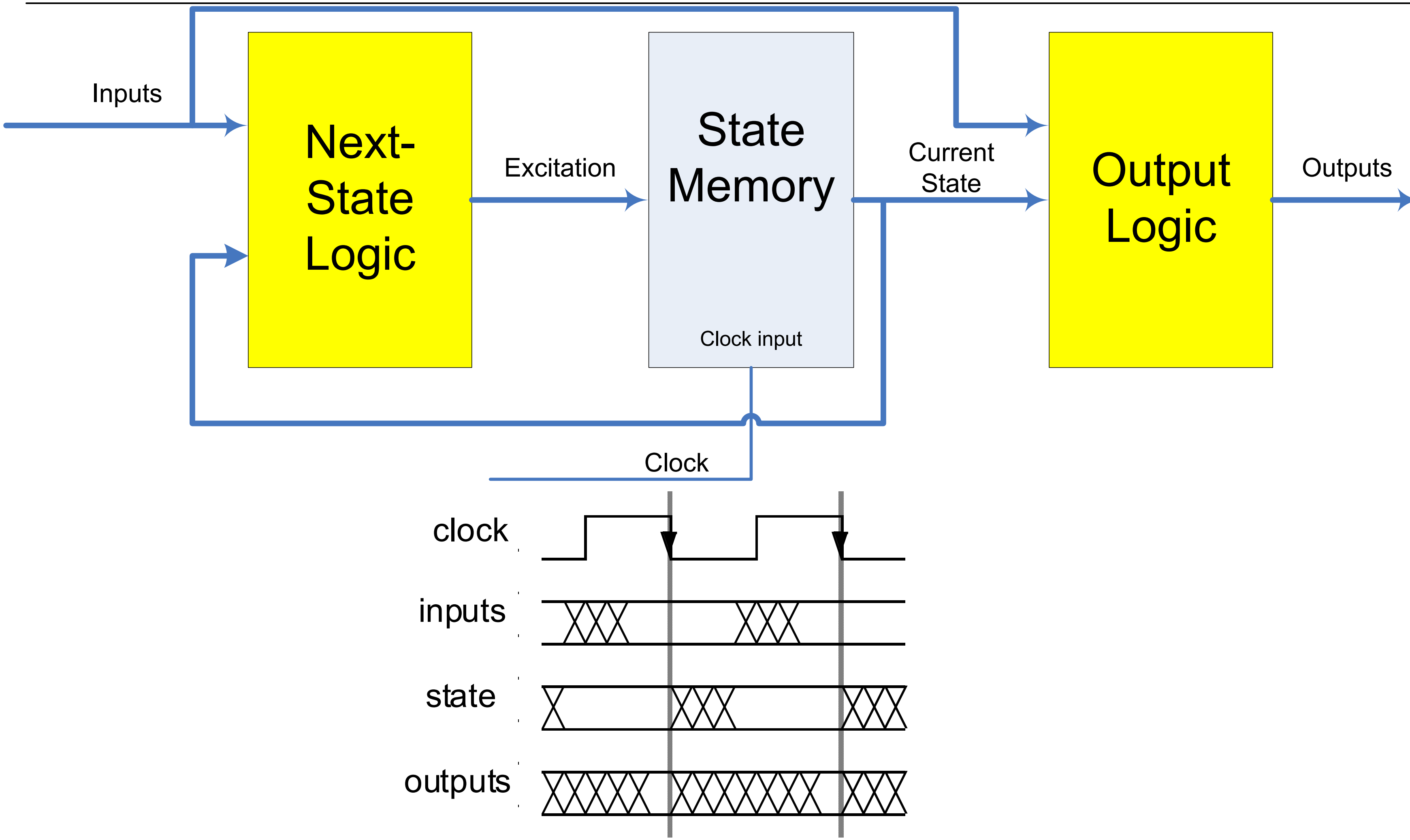




Moore machine timing

Excitation

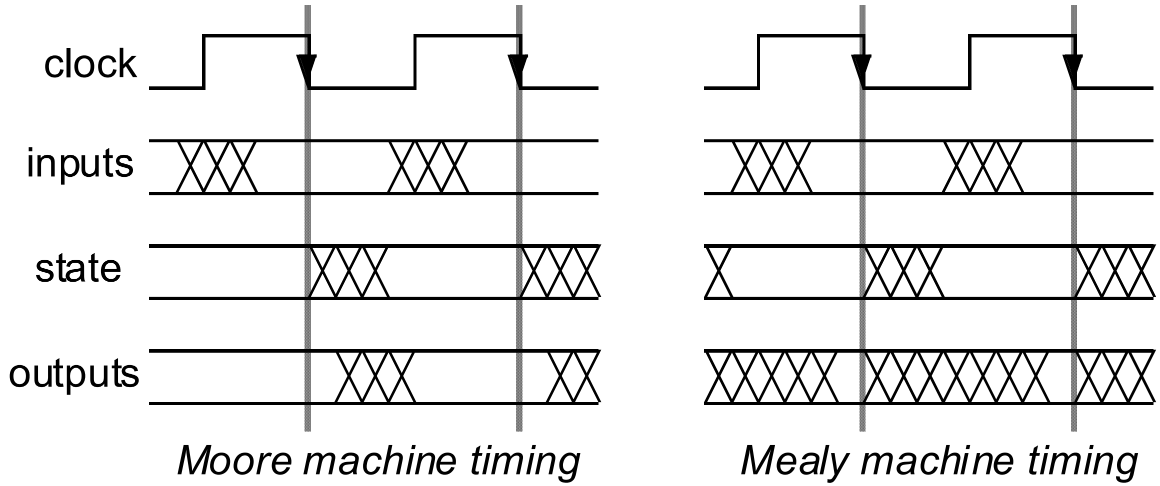
Mealy Machine



Mealy machine timing

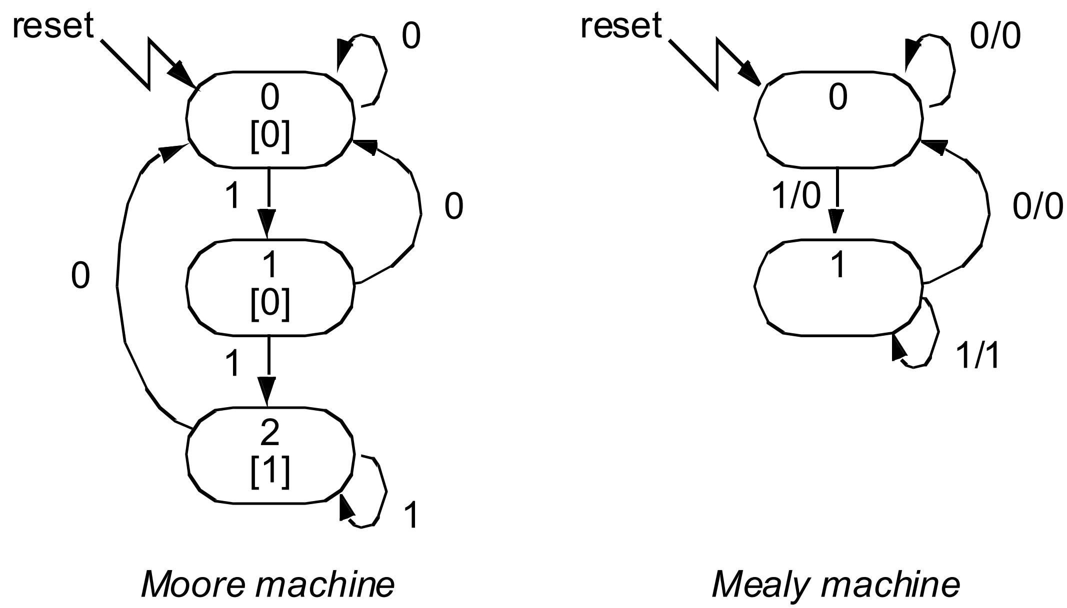
Moore vs. Mealy machines

- timing of input, state, and output changes



Mealy vs. Moore: Design Example

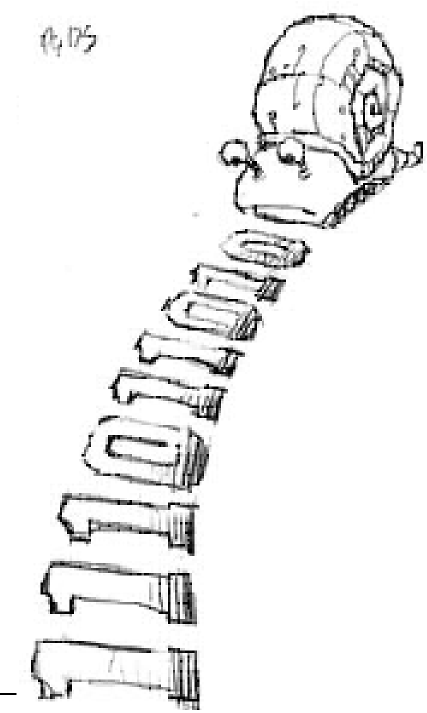
- function: assert output if 2 or more 1's in a row
- state diagram:



- advantages/disadvantages
  - Mealy often has fewer states than Moore machine since it associates outputs with transitions
  - Mealy machine can fall victim to glitches since outputs are asynchronous

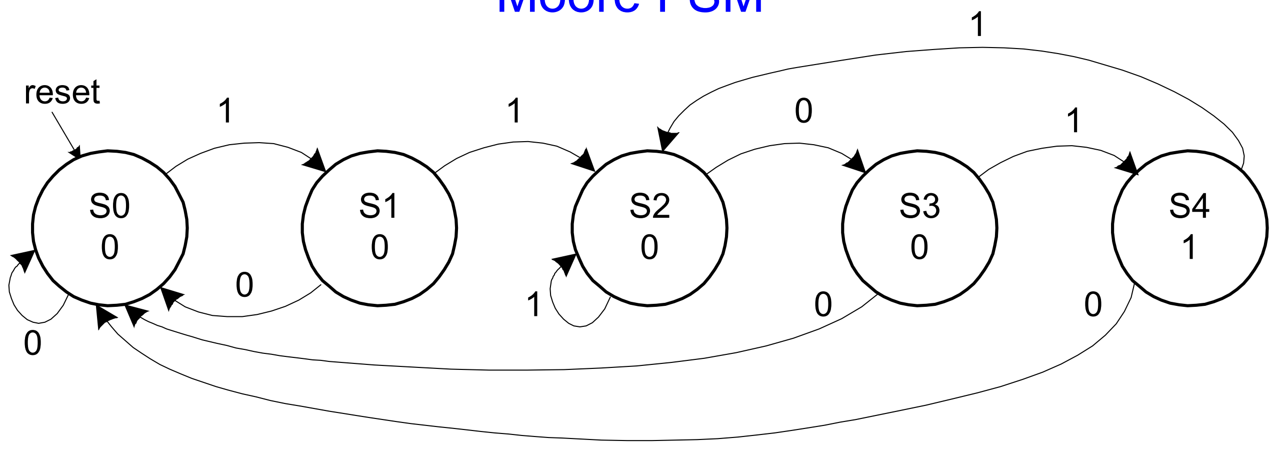
Moore vs. Mealy FSM

- Alyssa P. Hacker has a snail that crawls down a paper tape with 1's and 0's on it. The snail smiles whenever the last four digits it has crawled over are 1101. Design Moore and Mealy FSMs of the snail's brain.



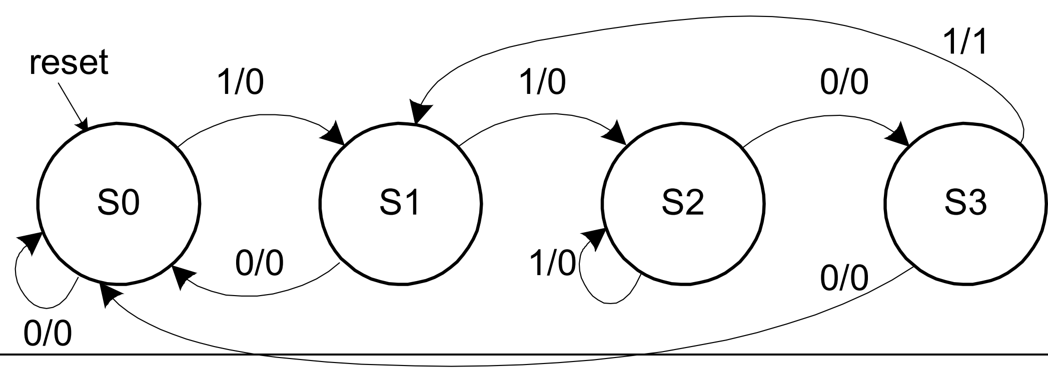
State Transition Diagrams

Moore FSM



Mealy FSM: arcs indicate input/output

Mealy FSM



Moore FSM State Transition Table

Current State			Inputs	Next State		
S_2	S_1	S_0	A	S'_2	S'_1	S'_0
0	0	0	0			
0	0	0	1			
0	0	1	0			
0	0	1	1			
0	1	0	0			
0	1	0	1			
0	1	1	0			
0	1	1	1			
1	0	0	0			
1	0	0	1			

State	Encoding
S0	000
S1	001
S2	010
S3	011
S4	100

Moore FSM State Transition Table

Current State			Inputs	Next State		
S_2	S_1	S_0	A	S'_2	S'_1	S'_0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	0	0
0	0	1	1	0	1	0
0	1	0	0	0	1	1
0	1	0	1	0	1	0
0	1	1	0	0	0	0
0	1	1	1	1	0	0
1	0	0	0	0	0	0
1	0	0	1	0	1	0

State	Encoding
S0	000
S1	001
S2	010
S3	011
S4	100

Moore FSM Output Table

Current State			Output
S_2	S_1	S_0	Y
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	

Moore FSM Output Table

Current State			Output
S_2	S_1	S_0	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1

$$Y = S_2$$

Mealy FSM State Transition and Output Table

Current State		Input	Next State		Output
S_1	S_0	A	S'_1	S'_0	Y
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

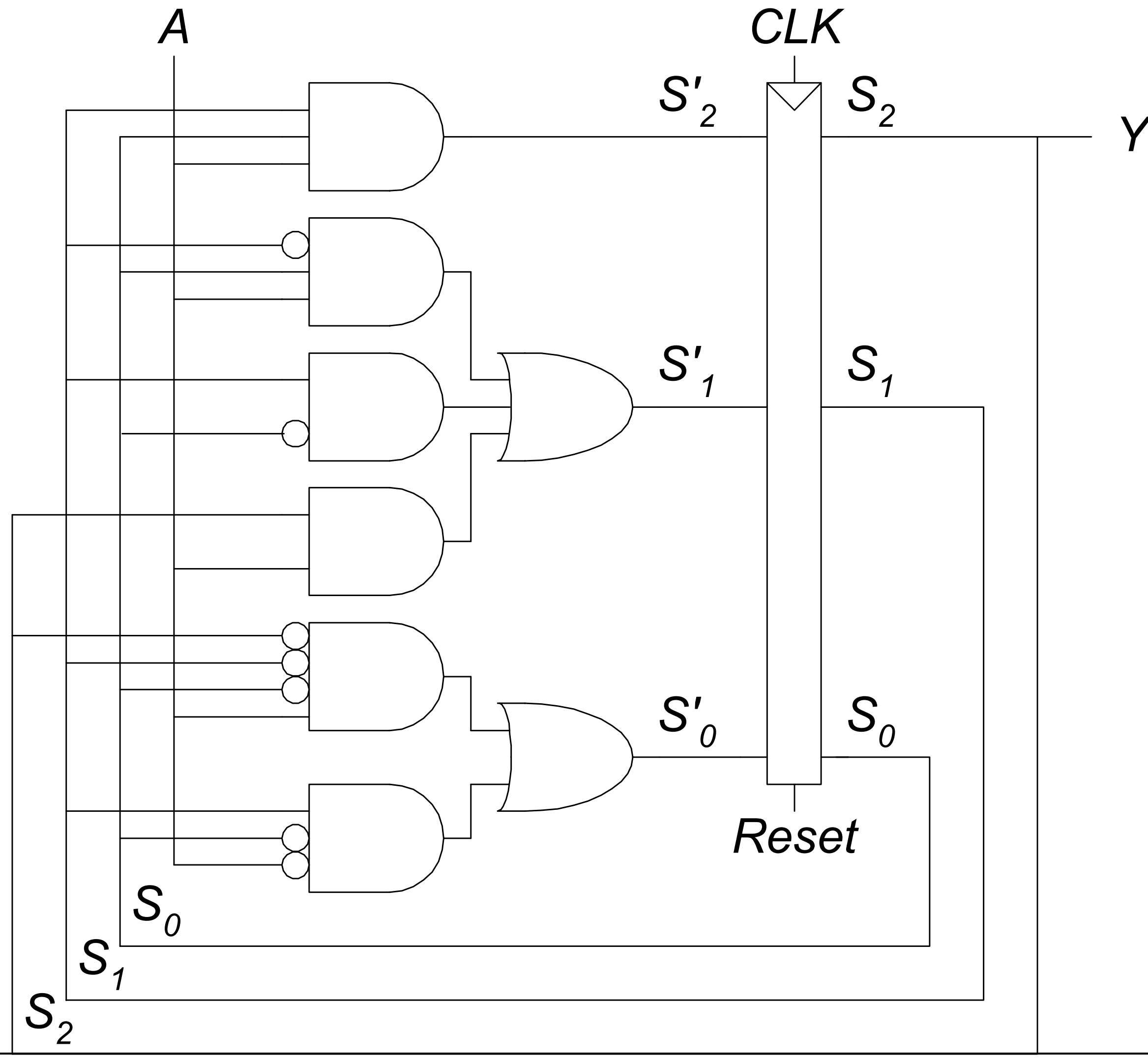
State	Encoding
S0	00
S1	01
S2	10
S3	11

Mealy FSM State Transition and Output Table

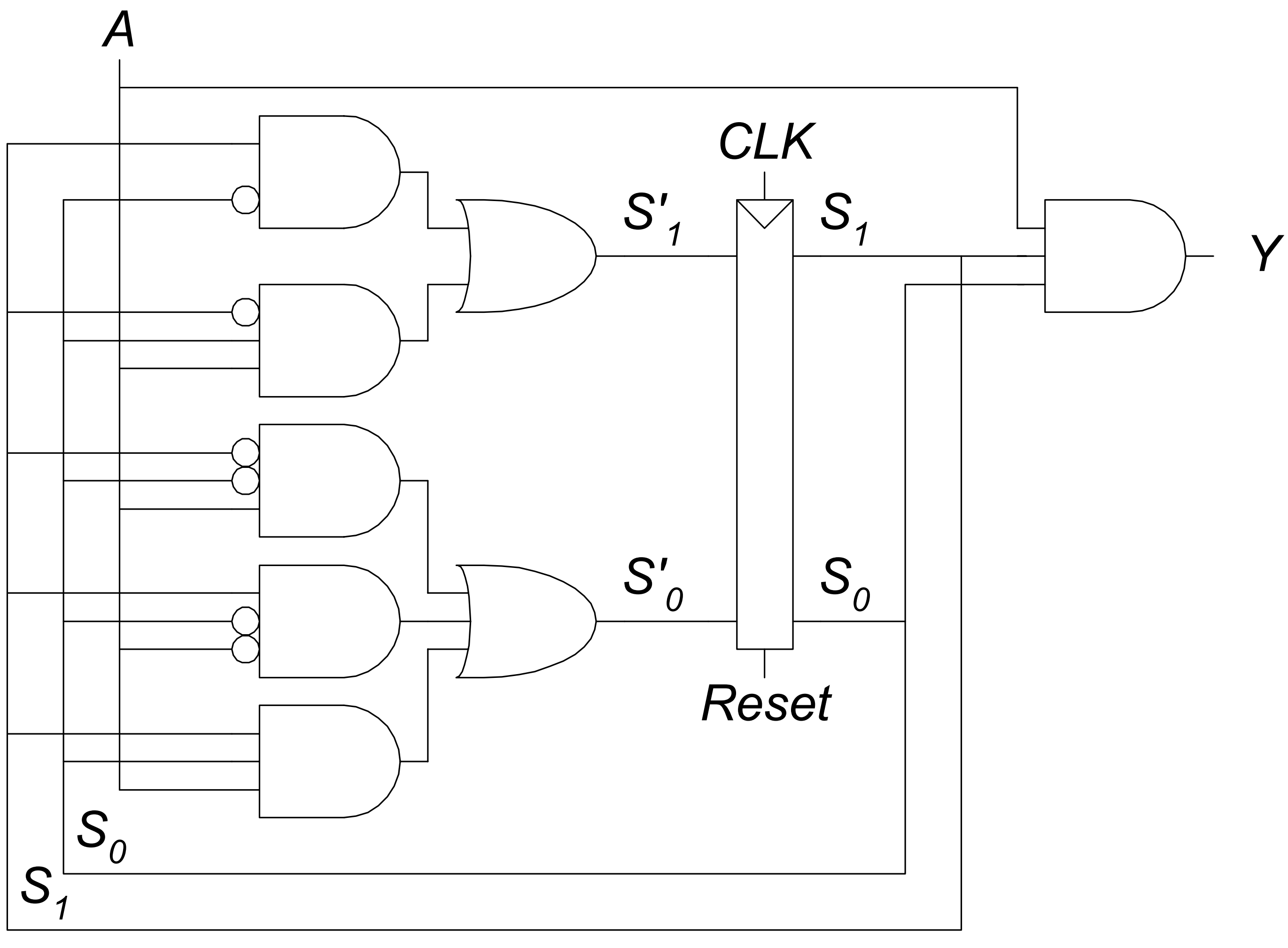
Current State		Input	Next State		Output
S_1	S_0	A	S'_1	S'_0	Y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	0	0
1	0	0	1	1	0
1	0	1	1	0	0
1	1	0	0	0	0
1	1	1	0	1	1

State	Encoding
S0	00
S1	01
S2	10
S3	11

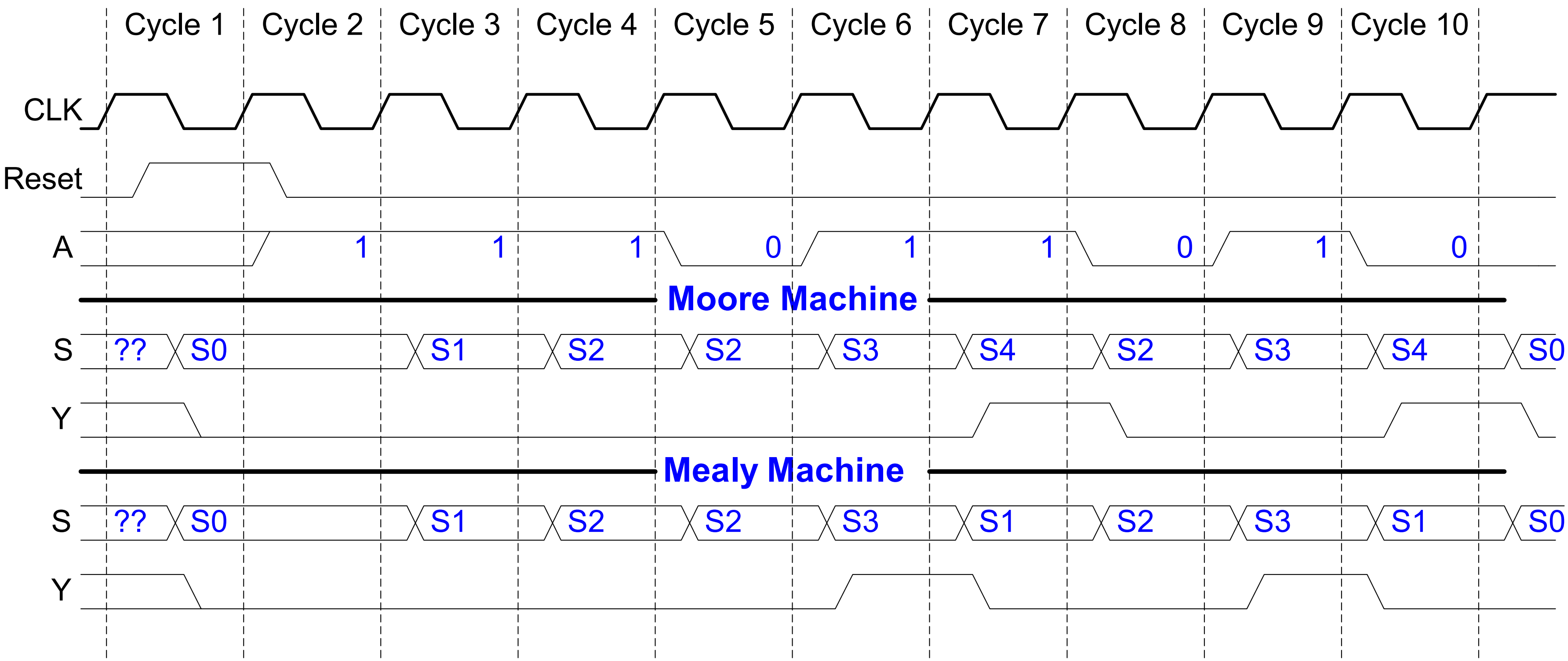
Moore FSM Schematic



Mealy FSM Schematic



Moore and Mealy Timing Diagram

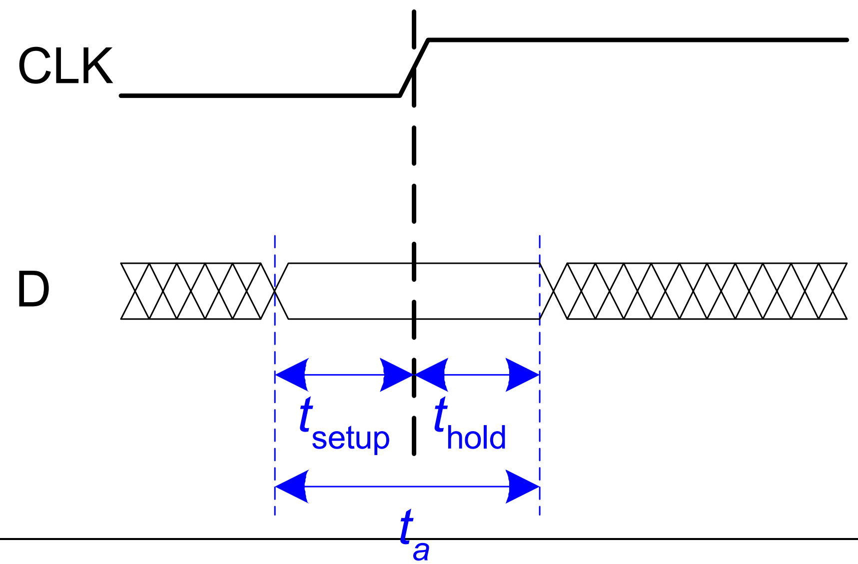


Timing

- Flip-flop samples *D* at clock edge
- *D* must be stable when it is sampled
- Similar to a photograph, *D* must be stable around the clock edge
- If *D* is changing when it is sampled, metastability can occur

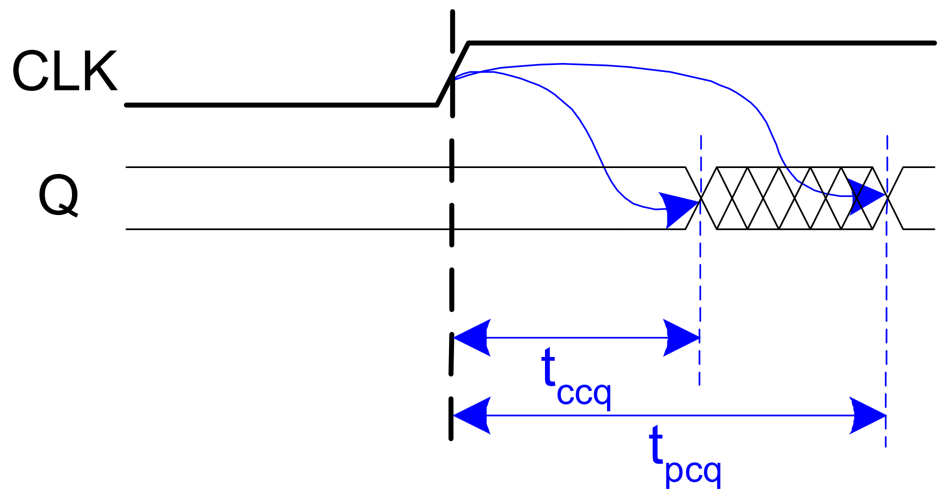
Input Timing Constraints

- Setup time: t_{setup} = time *before* the clock edge that data must be stable (i.e. not changing)
- Hold time: t_{hold} = time *after* the clock edge that data must be stable
- Aperture time: t_a = time around clock edge that data must be stable ($t_a = t_{setup} + t_{hold}$)



Output Timing Constraints

- Propagation delay: t_{pcq} = time after clock edge that the output Q is guaranteed to be stable (i.e., to stop changing)
- Contamination delay: t_{ccq} = time after clock edge that Q might be unstable (i.e., start changing)

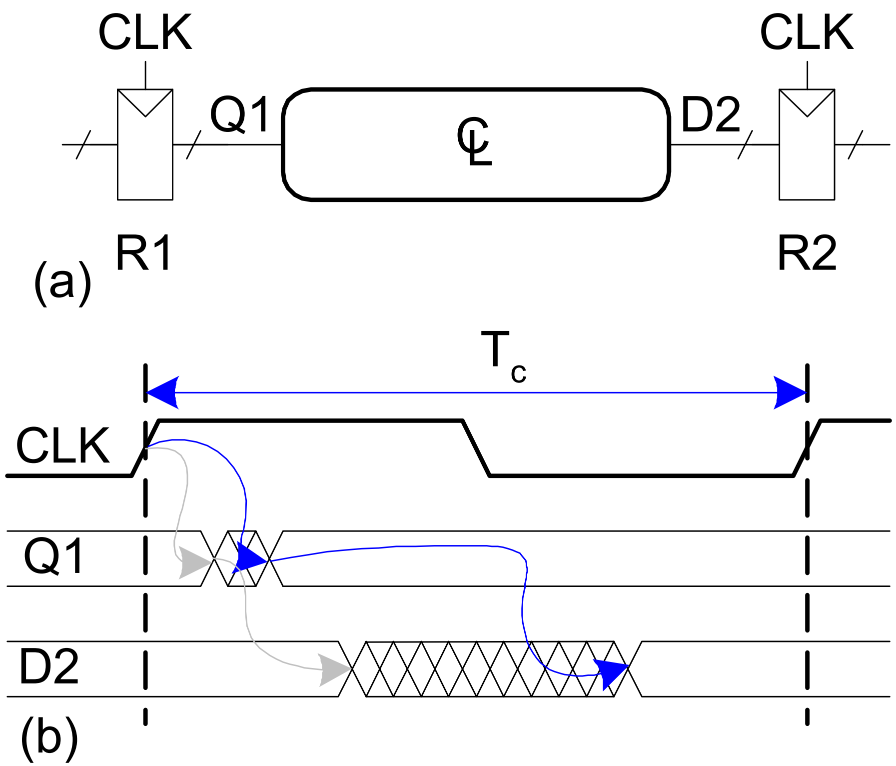


Dynamic Discipline

- The input to a synchronous sequential circuit must be stable during the aperture (setup and hold) time around the clock edge.
- Specifically, the input must be stable
  - at least t_{setup} before the clock edge
  - at least until t_{hold} after the clock edge

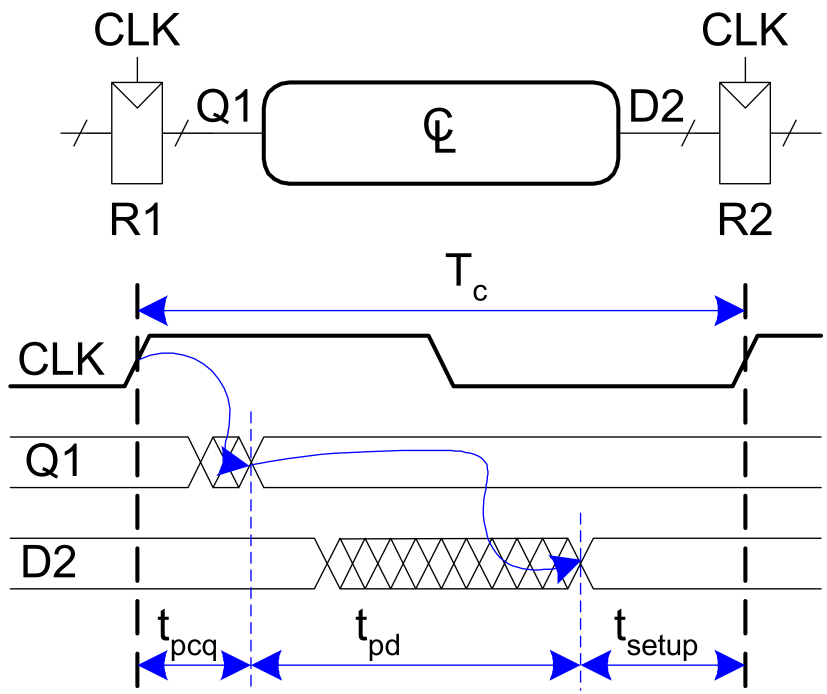
Dynamic Discipline

- The delay between registers has a **minimum** and **maximum** delay, dependent on the delays of the circuit elements



Setup Time Constraint

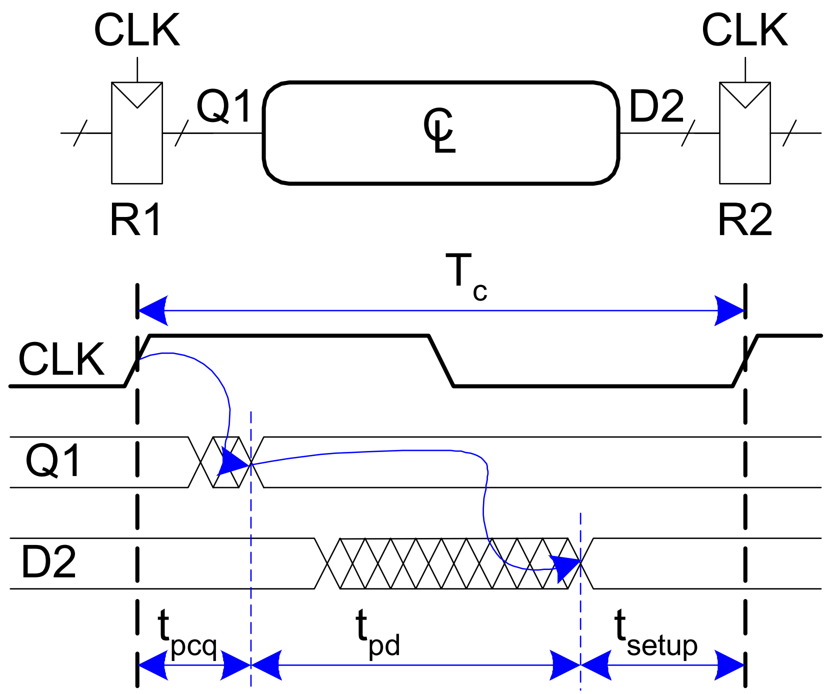
- The setup time constraint depends on the **maximum** delay from register R1 through the combinational logic.
- The input to register R2 must be stable at least t_{setup} before the clock edge.



$$T_c \geq$$

Setup Time Constraint

- The setup time constraint depends on the **maximum** delay from register R1 through the combinational logic.
- The input to register R2 must be stable at least t_{setup} before the clock edge.

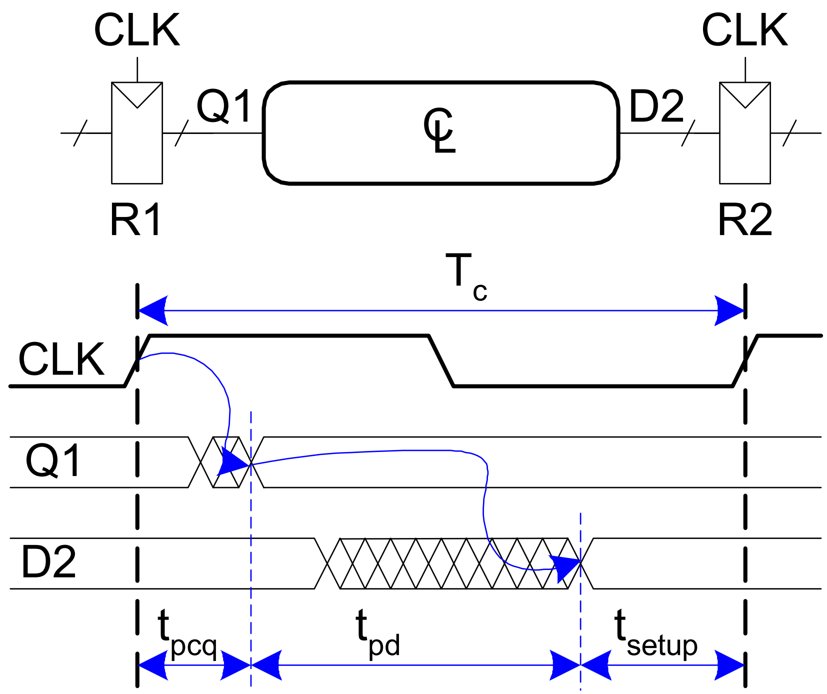


$$T_c \geq t_{pcq} + t_{pd} + t_{setup}$$

$$t_{pd} \leq$$

Setup Time Constraint

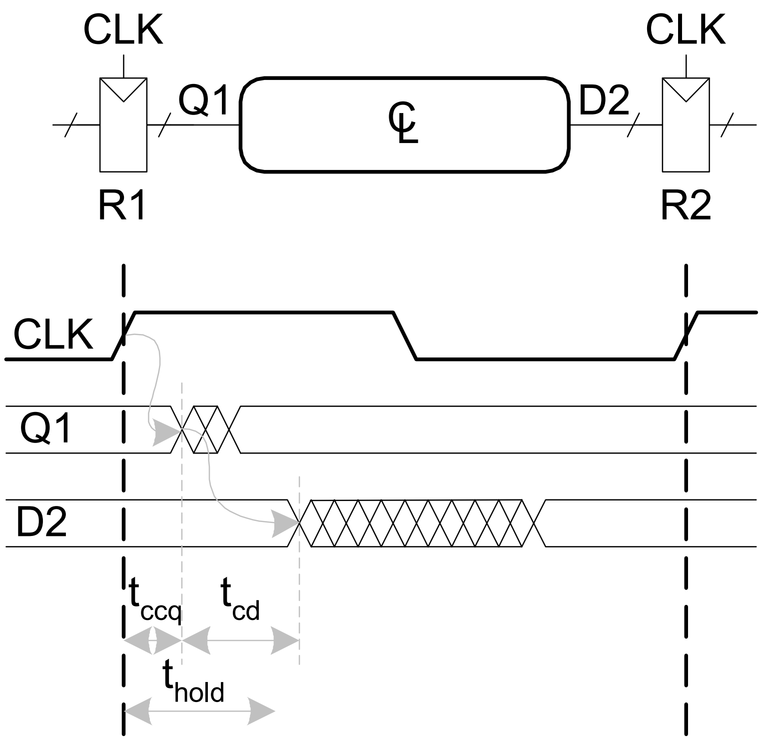
- The setup time constraint depends on the **maximum** delay from register R1 through the combinational logic.
- The input to register R2 must be stable at least t_{setup} before the clock edge.



$$T_c \geq t_{pcq} + t_{pd} + t_{setup}$$
$$t_{pd} \leq T_c - (t_{pcq} + t_{setup})$$

Hold Time Constraint

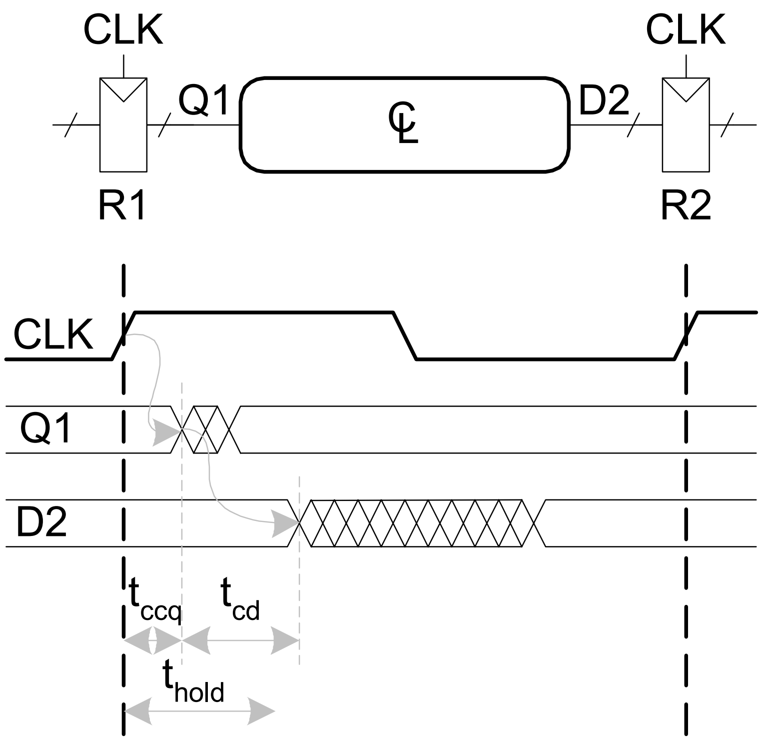
- The hold time constraint depends on the **minimum** delay from register R1 through the combinational logic.
- The input to register R2 must be stable for at least t_{hold} after the clock edge.



$t_{hold} <$

Hold Time Constraint

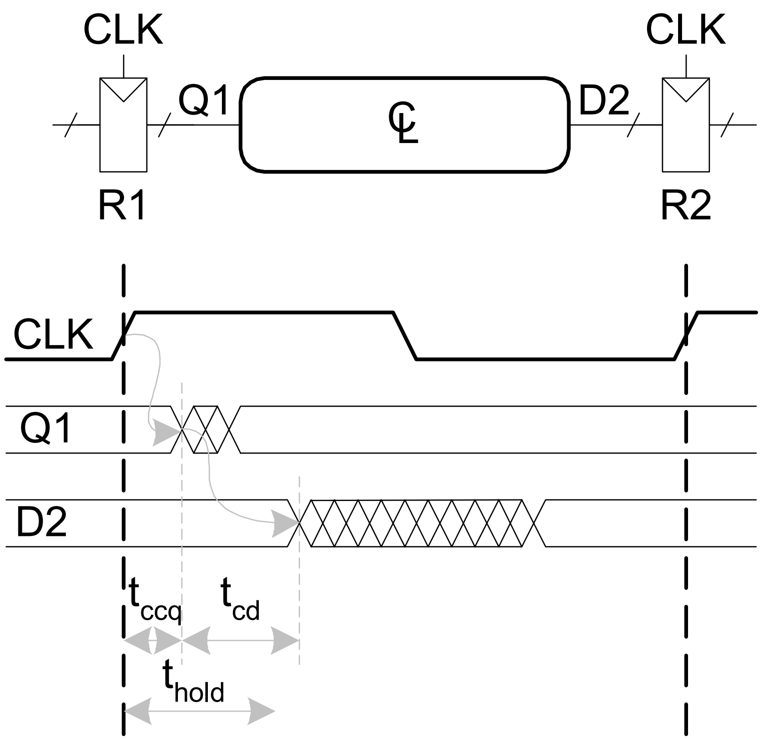
- The hold time constraint depends on the **minimum** delay from register R1 through the combinational logic.
- The input to register R2 must be stable for at least t_{hold} after the clock edge.



$$t_{hold} < t_{ccq} + t_{cd}$$
$$t_{cd} >$$

Hold Time Constraint

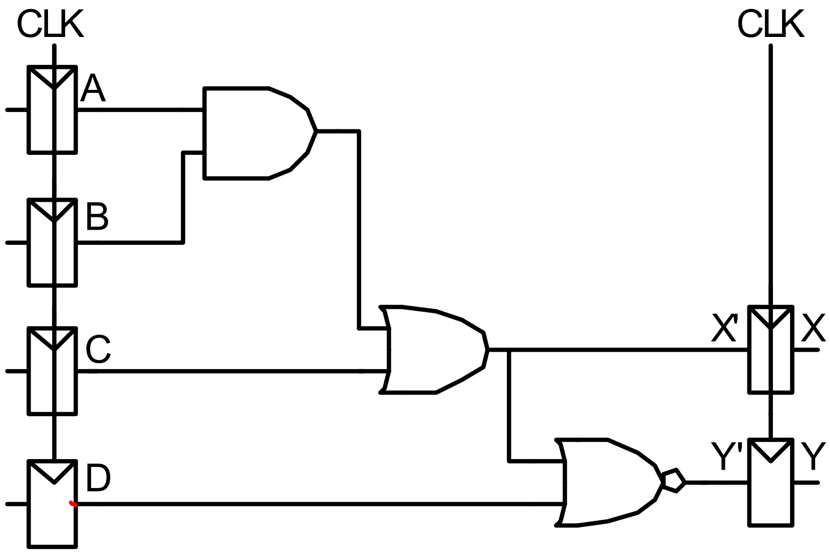
- The hold time constraint depends on the **minimum** delay from register R1 through the combinational logic.
- The input to register R2 must be stable for at least t_{hold} after the clock edge.



$$t_{hold} < t_{ccq} + t_{cd}$$

$$t_{cd} > t_{hold} - t_{ccq}$$

Timing Analysis



Timing Characteristics

$t_{ccq} = 30$ ps

$t_{pcq} = 50$ ps

$t_{\text{setup}} = 60$ ps

$t_{\text{hold}} = 70$ ps

per gate

$t_{pd} = 35$ ps

$t_{cd} = 25$ ps

T_{pd}

$t_{cd} =$

Setup time constraint:

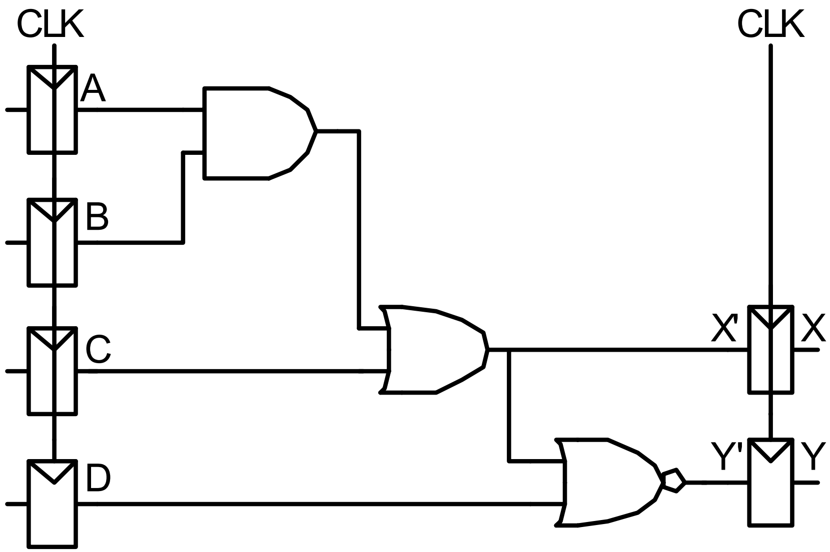
$T_c \geq$

$f_c = 1/T_c =$

Hold time constraint:

$t_{\text{ccq}} + t_{cd} > t_{\text{hold}}$?

Timing Analysis



Timing Characteristics

$t_{ccq} = 30$ ps

$t_{pcq} = 50$ ps

$t_{setup} = 60$ ps

$t_{hold} = 70$ ps

per gate

$t_{pd} = 35$ ps

$t_{cd} = 25$ ps

$t_{pd} = 3 \times 35$ ps $= 105$ ps

$t_{cd} = 25$ ps

Setup time constraint:

$T_c \geq (50 + 105 + 60)$ ps $= 215$ ps

$f_c = 1/T_c = 4.65$ GHz

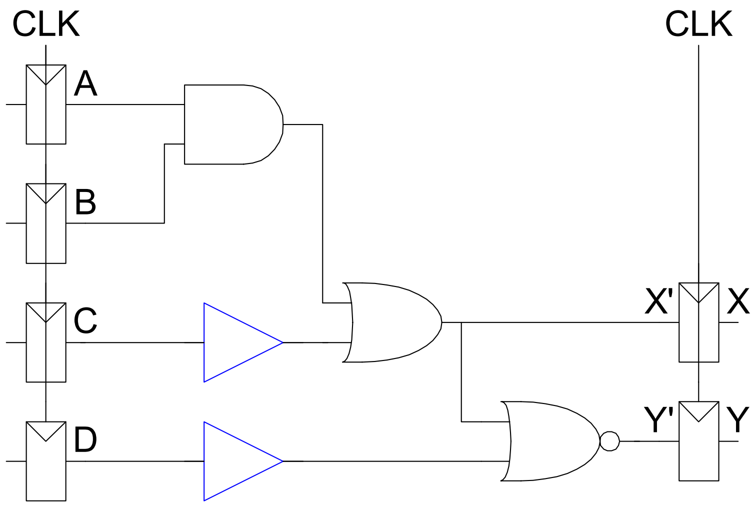
Hold time constraint:

$t_{ccq} + t_{cd} > t_{hold}$?

$(30 + 25)$ ps > 70 ps ? No!

Fixing Hold Time Violation

Add buffers to the short paths:



$t_{pd} =$

$t_{cd} =$

Setup time constraint:

$T_c \geq$

$f_c =$

Timing Characteristics

$t_{ccq} = 30$ ps

$t_{pcq} = 50$ ps

$t_{setup} = 60$ ps

$t_{hold} = 70$ ps

per gate

$t_{pd} = 35$ ps

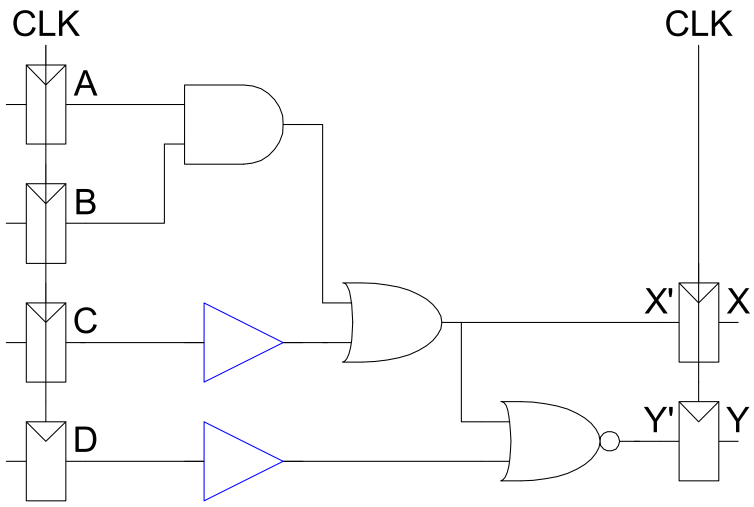
$t_{cd} = 25$ ps

Hold time constraint:

$t_{ccq} + t_{cd} > t_{hold}$?

Fixing Hold Time Violation

Add buffers to the short paths:



$t_{pd} = 3 \text{ x } 35 \text{ ps} = 105 \text{ ps}$

$t_{cd} = 2 \text{ x } 25 \text{ ps} = 50 \text{ ps}$

Setup time constraint:

$T_c \geq (50 + 105 + 60) \text{ ps} = 215 \text{ ps}$

$f_c = 1/T_c = 4.65 \text{ GHz}$

Timing Characteristics

$t_{ccq} = 30 \text{ ps}$

$t_{pcq} = 50 \text{ ps}$

$t_{setup} = 60 \text{ ps}$

$t_{hold} = 70 \text{ ps}$

per gate:

$t_{pd} = 35 \text{ ps}$

$t_{cd} = 25 \text{ ps}$

Hold time constraint:

$t_{ccq} + t_{cd} > t_{hold}$?

$(30 + 50) \text{ ps} > 70 \text{ ps}$? Yes!