

CSE140 Midterm2 Review

Xinyuan Wang

05/13/2018

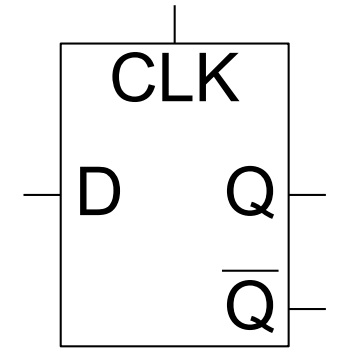
Outline

- Flip-Flops
- Finite State Machines
- Timing and Retiming

D Latch

- Two inputs: CLK , D
 - CLK : controls *when* the output changes
 - D (the data input): controls *what* the output changes to
- Function
 - When $CLK = 1$, D passes through to Q (the latch is *transparent*)
 - When $CLK = 0$, Q holds its previous value (the latch is *opaque*)
- Avoids invalid case when $Q \neq \overline{Q}$

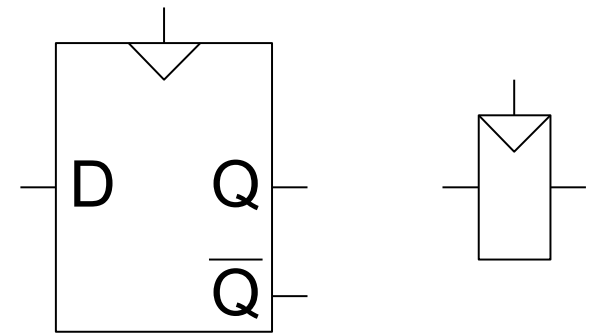
D Latch
Symbol



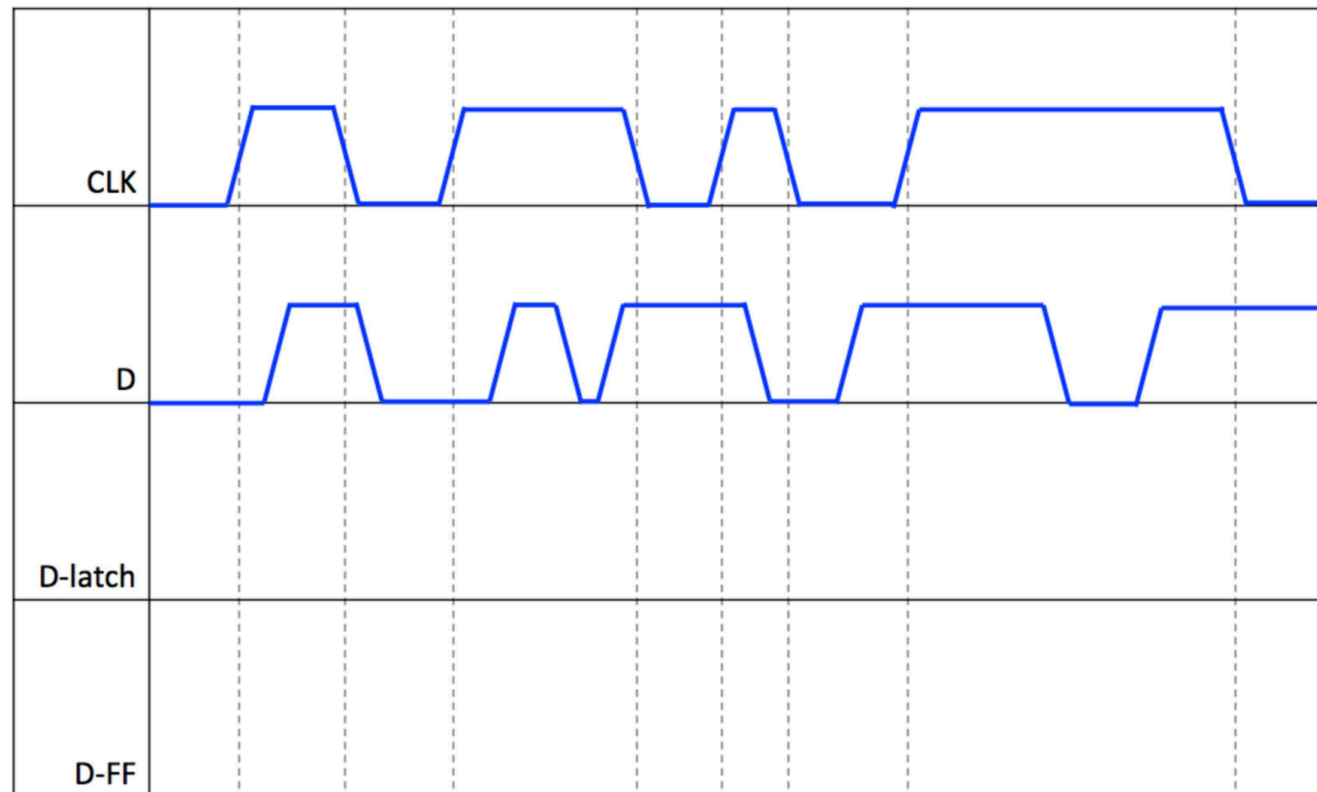
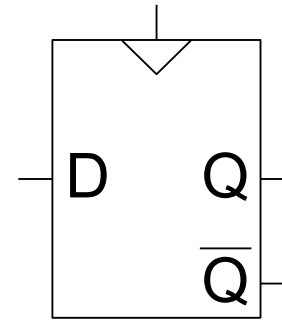
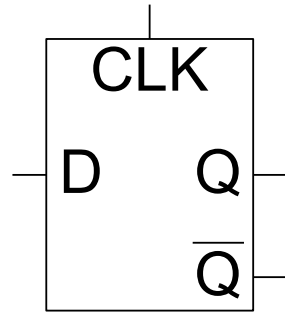
D Flip-Flop

- Two inputs: CLK, D
- Function
 - The flip-flop “samples” D on the rising edge of CLK
 - When CLK rises from 0 to 1, D passes through to Q
 - Otherwise, Q holds its previous value
 - Q changes only on the rising edge of CLK
- A flip-flop is called an edge-triggered device because it is activated on the clock edge

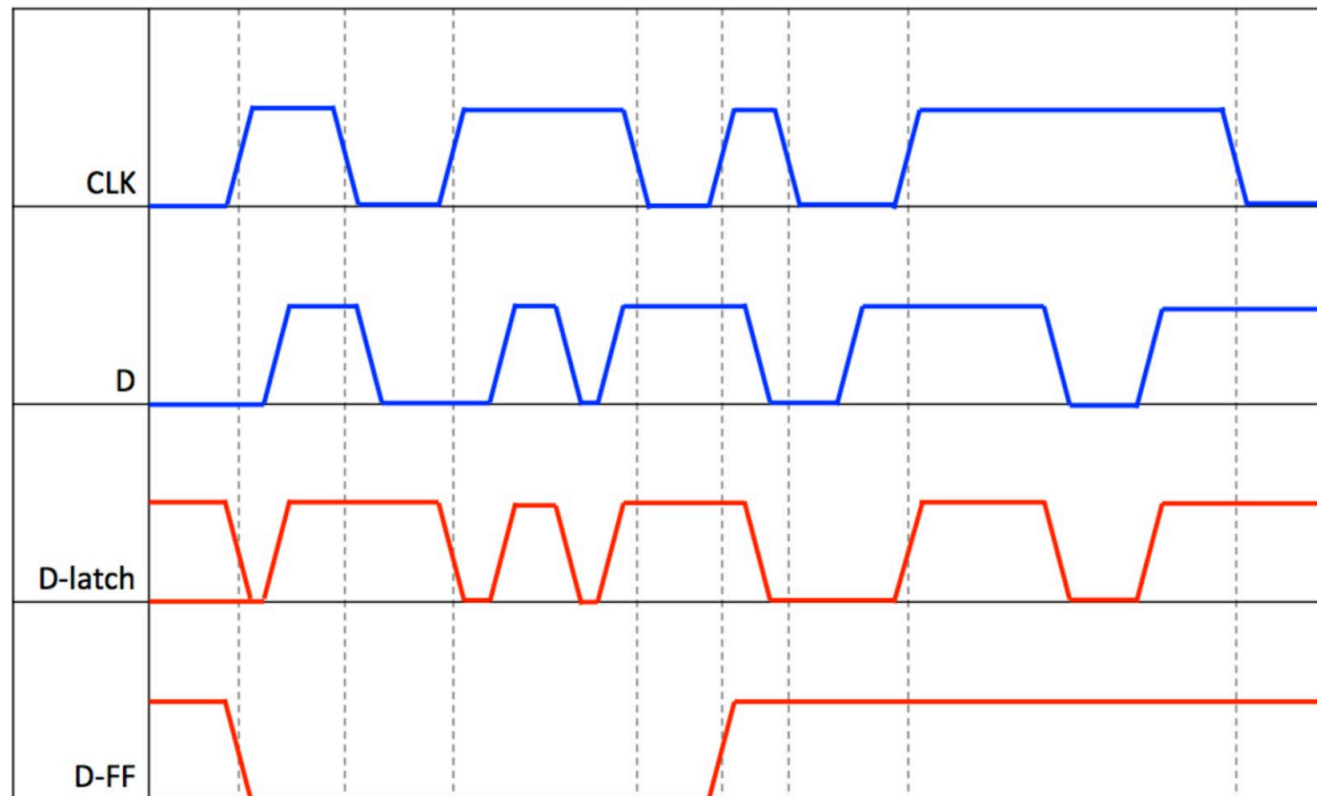
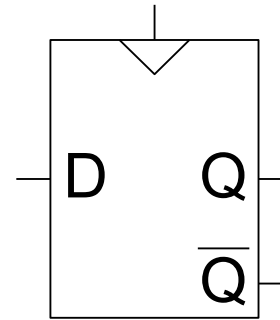
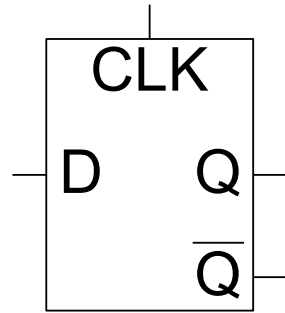
D Flip-Flop
Symbols



D-Latch vs. D Flip-Flop



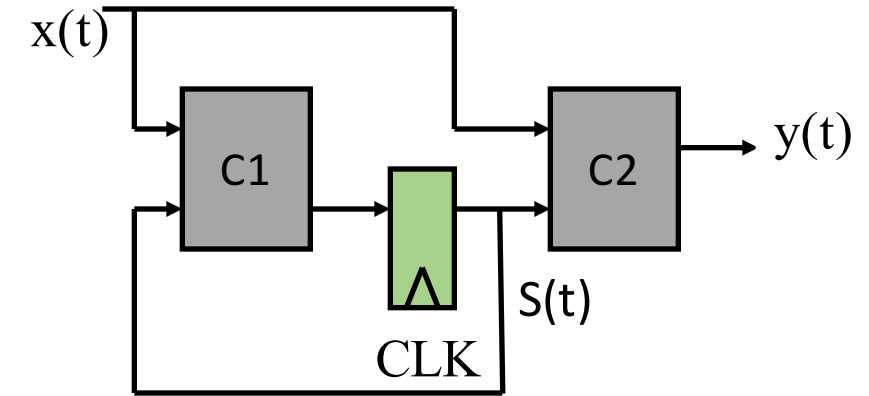
D-Latch vs. D Flip-Flop



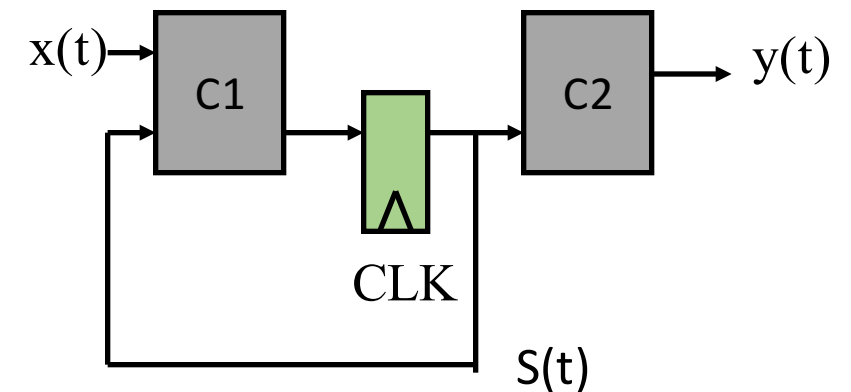
Mealy and Moore Machines

- Mealy Machine: general
 - $y_i(t) = f_i(X(t), S(t))$
- Moore Machine: Output is independent of current input
 - $y_i(t) = f_i(S(t))$
 - $S_i(t + 1) = g_i(X(t), S(t))$
- How to convert Mealy to Moore machine?

Mealy Machine



Moore Machine



Conversion from Mealy to Moore

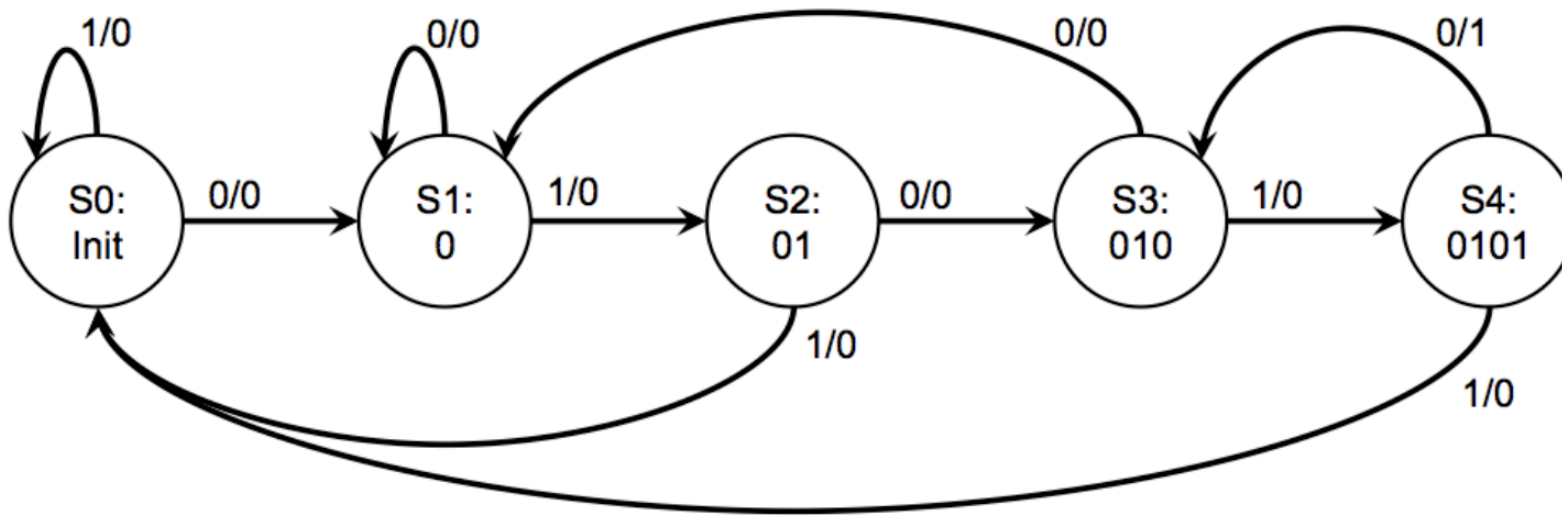
- Algorithm
 1. Identify distinct (NS, y) pair
 2. Replace each distinct (NS, y) pair with distinct new states
 3. Insert rows of present state = new states
 4. Append each present state with its output y

Conversion from Mealy to Moore

- Example: Q4 from Mid2Wi16

A pattern recognizer produces output $y = 1$ when $x(t - 4, t) = 01010$

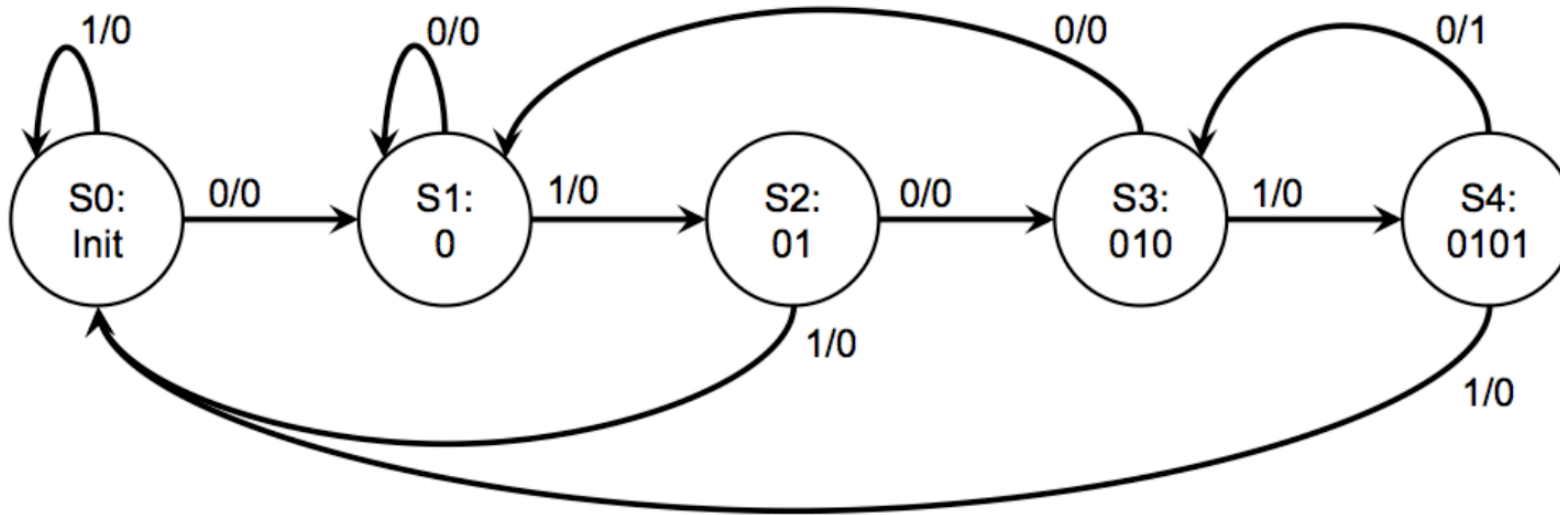
Mealy Machine



PS/Input	$X = 0$	$X = 1$
S_0	$S_1, 0$	$S_0, 0$
S_1	$S_1, 0$	$S_2, 0$
S_2	$S_3, 0$	$S_0, 0$
S_3	$S_1, 0$	$S_4, 0$
S_4	$S_3, 1$	$S_0, 0$

Conversion from Mealy to Moore

Mealy Machine



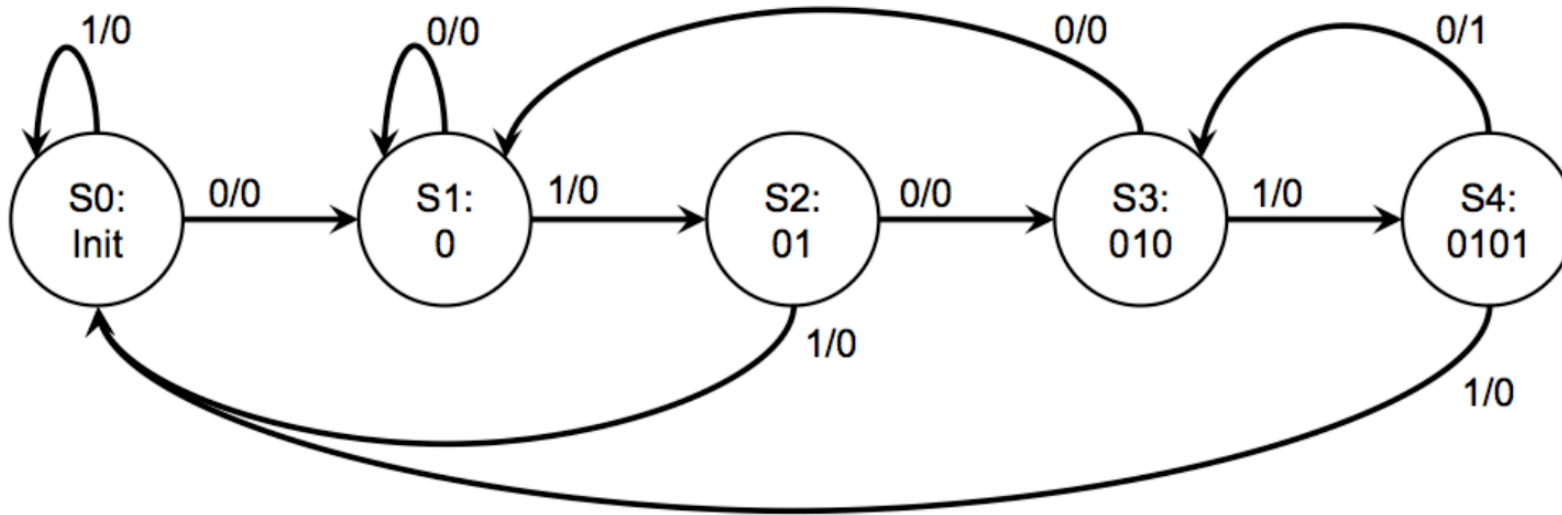
PS/Input	$X = 0$	$X = 1$
S_0	$S_1, 0$	$S_0, 0$
S_1	$S_1, 0$	$S_2, 0$
S_2	$S_3, 0$	$S_0, 0$
S_3	$S_1, 0$	$S_4, 0$
S_4	$S_3, 1$	$S_0, 0$

- Algorithm

1. Identify distinct (NS, y) pair

Conversion from Mealy to Moore

Mealy Machine



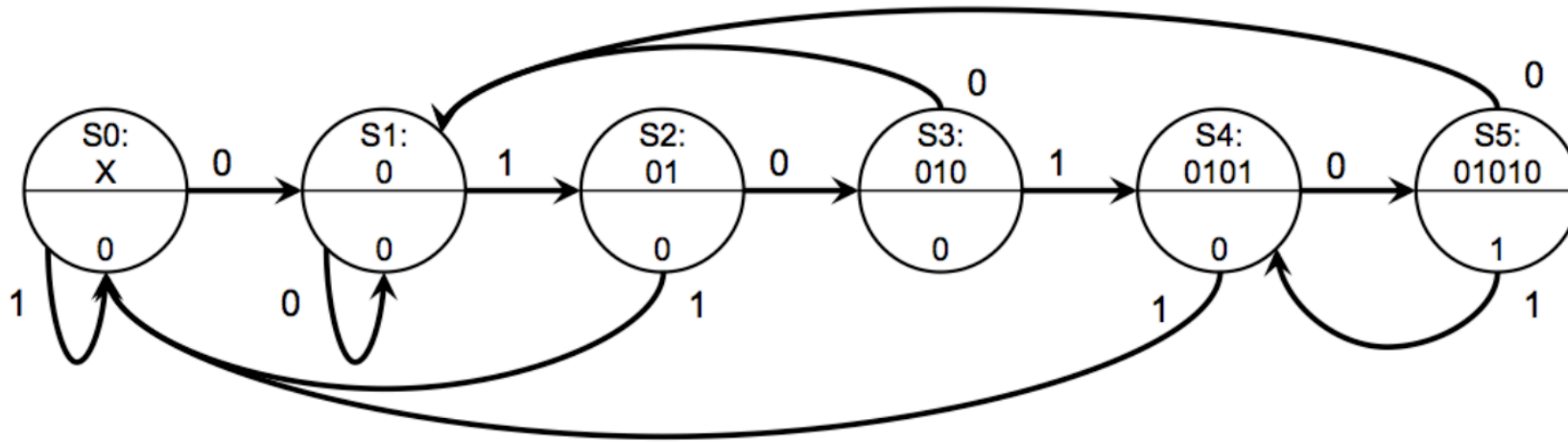
PS/Input	$X = 0$	$X = 1$	y
S_0	S_1	S_0	0
S_1	S_1	S_2	0
S_2	S_3	S_0	0
S_3	S_1	S_4	0
S_4	S_5	S_0	
S_5			

- Algorithm

1. Identify distinct (NS, y) pair
2. Replace each distinct (NS, y) pair with distinct **new states**
3. Insert rows of present state = **new states**

Conversion from Mealy to Moore

Moore Machine



PS/Input	$X = 0$	$X = 1$	y
S_0	S_1	S_0	0
S_1	S_1	S_2	0
S_2	S_3	S_0	0
S_3	S_1	S_4	0
S_4	S_5	S_0	0
S_5	S_1	S_4	1

- Algorithm

1. Identify distinct (NS, y) pair
2. Replace each distinct (NS, y) pair with distinct **new states**
3. Insert rows of present state = **new states**
4. Append each present state with its output y

Implementation

✓ **Given: State diagram**



Circuit implementation?

- Example: Lec8 and HW3 “pattern recognizer”

✓ **Given: Circuit implementation**



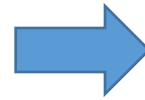
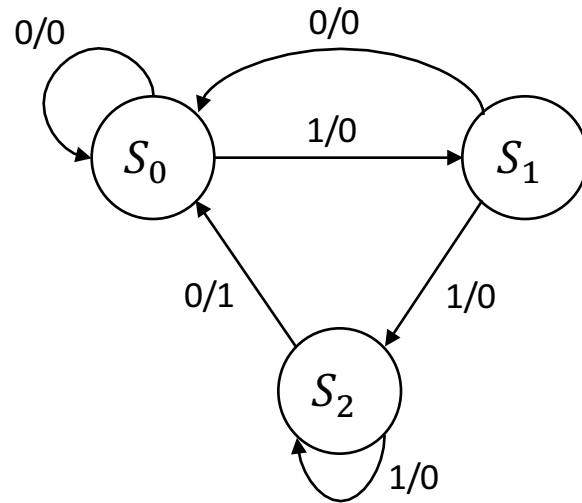
Input output relation?

Implementation

- ✓ **Given: State diagram**
 - Derive the state table and assign states
 - Excitation table
 - Equations(K-maps if needed)
 - Circuit implementation

Examples : given a state diagram

- **Step 1: Derive state table and assign states**

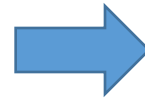
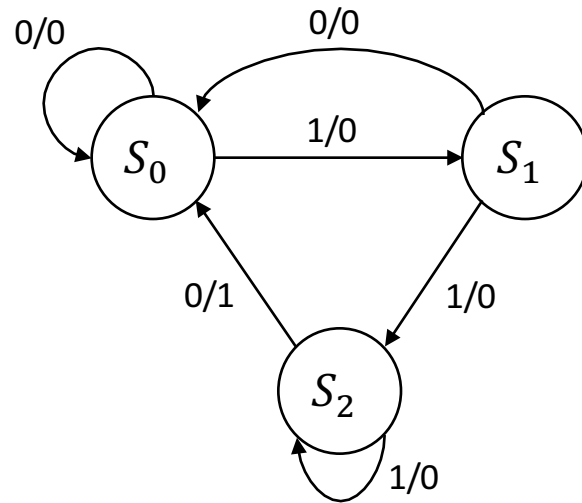


PS/Input	$X = 0$	$X = 1$
S_0	$S_0, 0$	$S_1, 0$
S_1	$S_0, 0$	$S_2, 0$
S_2	$S_0, 1$	$S_2, 0$

- How many binary variables are needed to encode the states?

Examples : given a state diagram

- **Step 1: Derive state table and assign states**



PS/Input	$X = 0$	$X = 1$
S_0	$S_0, 0$	$S_1, 0$
S_1	$S_0, 0$	$S_2, 0$
S_2	$S_0, 1$	$S_2, 0$

- How many binary variables are needed to encode the states?

2 bits

$S_0 \Rightarrow 00$

$S_1 \Rightarrow 01$

$S_2 \Rightarrow 10$

Examples : given a state diagram

- **Step 2:** Excitation table

$Q_1 Q_0$
 $S_0 \Rightarrow 00$
 $S_1 \Rightarrow 01$
 $S_2 \Rightarrow 10$

PS/Input	$X = 0$	$X = 1$
S_0	$S_0, 0$	$S_1, 0$
S_1	$S_0, 0$	$S_2, 0$
S_2	$S_0, 1$	$S_2, 0$

- Excitation table

$Q_1(t)$	$Q_0(t)$	$X(t)$	$Q_1(t+1)$	$Q_0(t+1)$	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	0	0	1
1	0	1	1	0	0
1	1	0	X	X	X
1	1	1	X	X	X

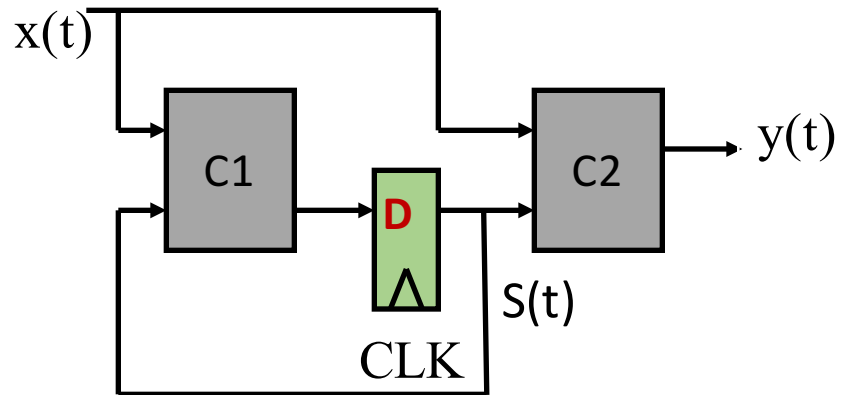


Examples : given a state diagram

- **Step 3: Derive equations**

- Use **D Flip-Flop**: $Q(t + 1) = D(t)$
- K-maps

Mealy Machine



$Q_1(t)$	$Q_0(t)$	$X(t)$	$D_1(t)$	$D_0(t)$	$Q_1(t + 1)$	$Q_0(t + 1)$	y
0	0	0	0		0	0	0
0	0	1	0		0	1	0
0	1	0	0		0	0	0
0	1	1	1		1	0	0
1	0	0	0		0	0	1
1	0	1	1		1	0	0
1	1	0	X		X	X	X
1	1	1	X		X	X	X

$D_1(t)$

$X(t)/Q_1(t)Q_0(t)$	00	01	11	10
0	0	0	X	0
1	0	1	X	1

$$D_1(t) = X(t)Q_0(t) + X(t)Q_1(t)$$

Examples : given a state diagram

- **Step 3: Derive equations**
 - Use D Flip-Flop: $Q(t+1) = D(t)$
 - K-maps

$D_0(t)$

$X(t)/Q_1(t)Q_0(t)$	00	01	11	10
0	0	0	X	0
1	1	0	X	0

$$D_0(t) = X(t)Q'_0(t)Q'_1(t)$$

$y(t)$

$X(t)/Q_1(t)Q_0(t)$	00	01	11	10
0	0	0	X	1
1	0	0	X	0

$$y = X'(t)Q_1(t)$$

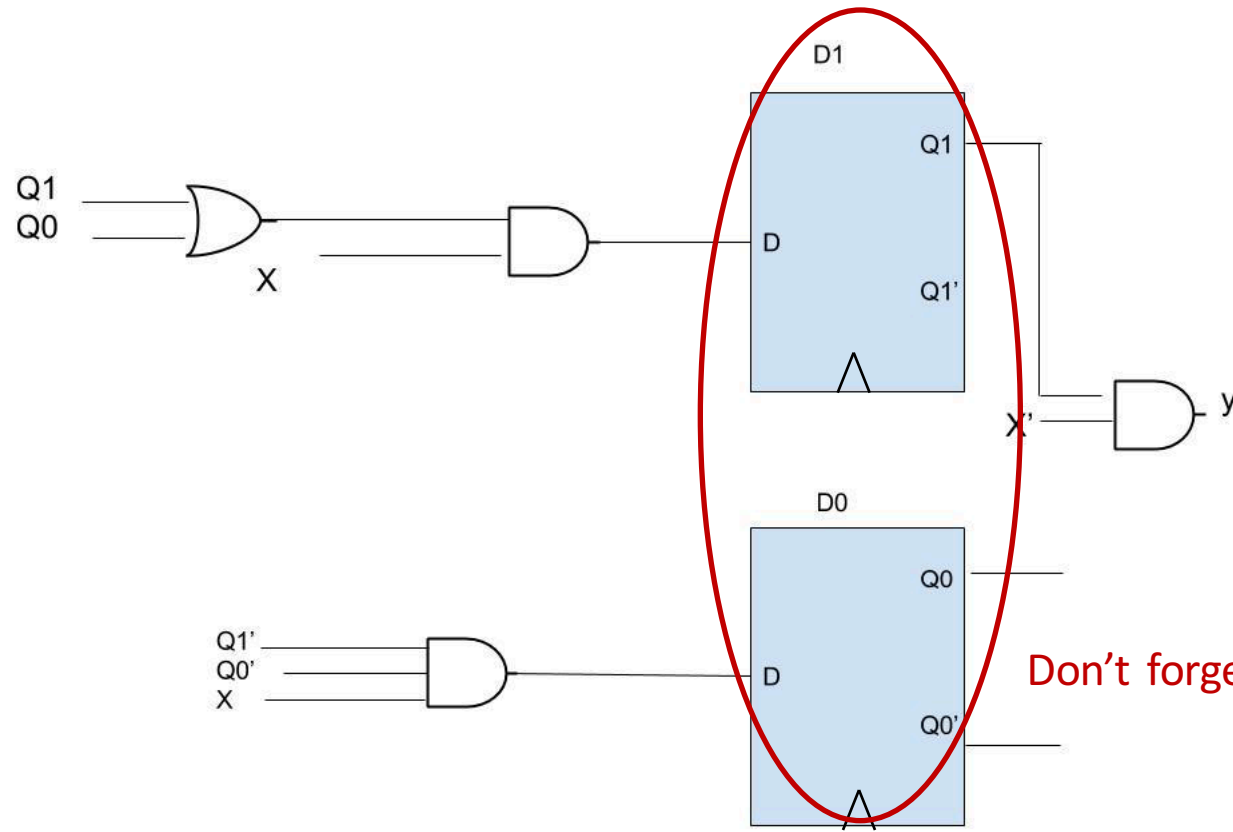
$D_0(t)$ output

$Q_1(t)$	$Q_0(t)$	$X(t)$	$D_1(t)$	$D_0(t)$	$Q_1(t+1)$	$Q_0(t+1)$	y
0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	0
0	1	0	0	0	0	0	0
0	1	1	1	0	1	0	0
1	0	0	0	0	0	0	1
1	0	1	1	0	1	0	0
1	1	0	X	X	X	X	X
1	1	1	X	X	X	X	X

Examples : given a state diagram

- **Step 4:** Circuit implementation

- Use D Flip-Flop: $Q(t+1) = D(t)$



$$D_1(t) = X(t)Q_0(t) + X(t)Q_1(t)$$

$$D_0(t) = X(t)Q'_0(t)Q'_1(t)$$

$$y = X'(t)Q_1(t)$$

Don't forget the label!

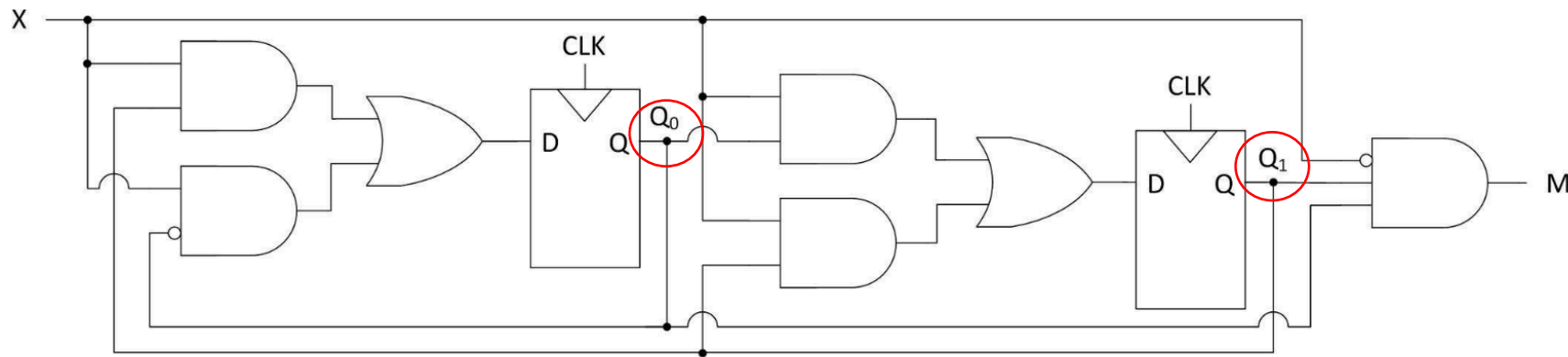
Implementation

- ✓ **Given: Circuit implementation**
 - Excitation table
 - Identify states and derive a state table
 - State diagram
 - Input output relation

Example: given a circuit implementation

- **Step 1: Excitation table**

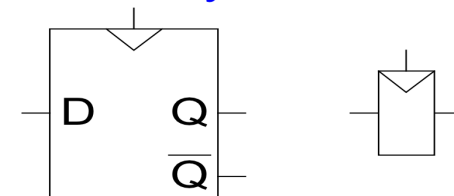
4. (Finite State Machine Specification) Analyze the following circuit.



4.1 Write the transition(excitation) table (8 points).

- D Flip-Flop: $Q(t+1) = D(t)$
 - $D(t)$ can be find with current input $X(t)$ and current state $Q_0(t), Q_1(t)$

D Flip-Flop
Symbols



Examples : given a circuit implementation

- **Step 1: Excitation table**

- State Equations

$$Q_0(t+1) = XQ_1(t) + XQ_0(t)'$$

$$Q_1(t+1) = XQ_1(t) + XQ_0(t)$$

$$M = X'Q_1(t)Q_0(t)$$



- Excitation table

$Q_1(t)$	$Q_0(t)$	X	$Q_1(t+1)$	$Q_0(t+1)$	M
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	0	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	1	1	1	0

Examples : given a circuit implementation

- **Step 2: Identify states and derive state table**

Q_1Q_0
 $S_0 \leftarrow 00$
 $S_1 \leftarrow 01$
 $S_2 \leftarrow 10$
 $S_3 \leftarrow 11$

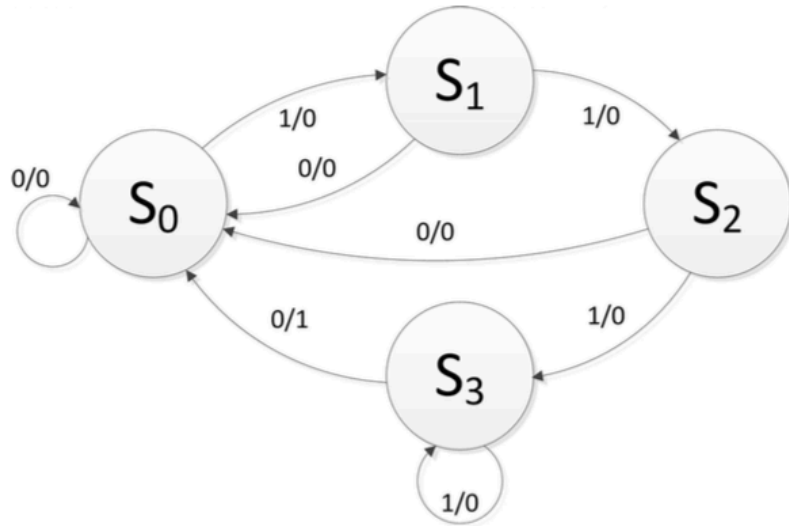
PS/Input	$X = 0$	$X = 1$
S_0	$S_0, 0$	$S_1, 0$
S_1	$S_0, 0$	$S_2, 0$
S_2	$S_0, 0$	$S_3, 0$
S_3	$S_0, 1$	$S_3, 0$

$Q_1(t)$	$Q_0(t)$	X	$Q_1(t+1)$	$Q_0(t+1)$	M
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	0	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	1	1	1	0



Examples : given a circuit implementation

- **Step 3: State diagram**



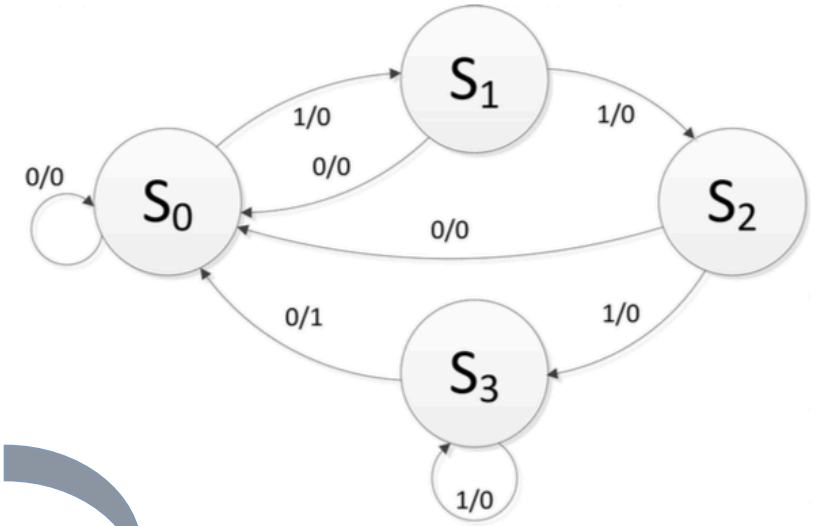
PS/Input	$X = 0$	$X = 1$
S_0	$S_0, 0$	$S_1, 0$
S_1	$S_0, 0$	$S_2, 0$
S_2	$S_0, 0$	$S_3, 0$
S_3	$S_0, 1$	$S_3, 0$

Examples : given a circuit implementation

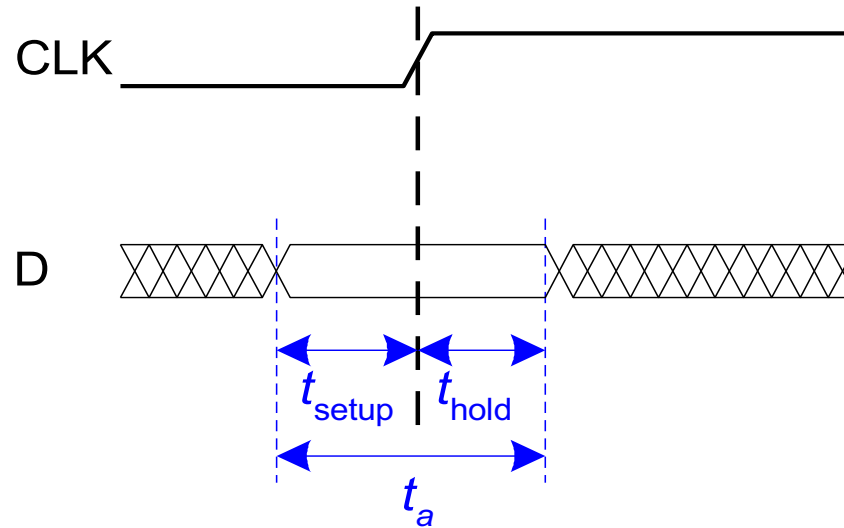
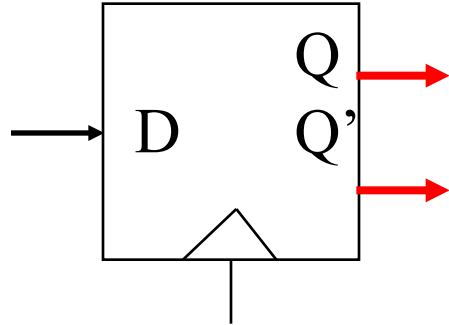
- **Step 4: Input output relation**

cycle	1	2	3	4	5	6	7	8	9	10
X	1	1	1	0	0	0	1	1	1	0
state	S_0									
M										

cycle	1	2	3	4	5	6	7	8	9	10
X	1	1	1	0	0	0	1	1	1	0
state	S_0	S_1	S_2	S_3	S_0	S_0	S_0	S_1	S_2	S_3
M	0	0	0	1	0	0	0	0	0	1



Sequential Networks: setup and hold time



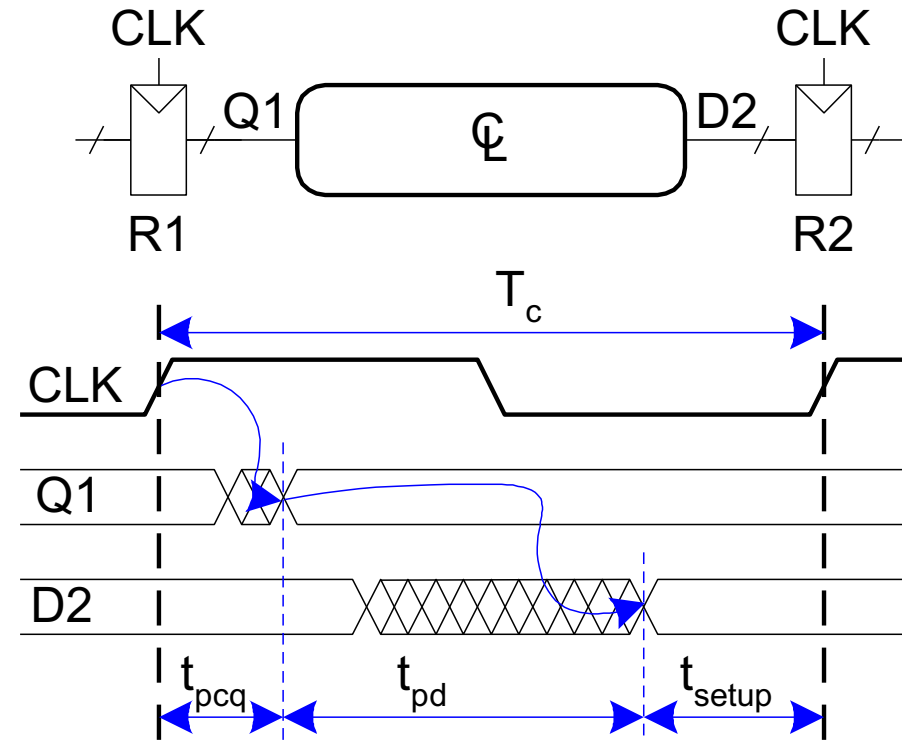
- **Setup time t_{setup}** Time *before* the clock edge that data must be stable (i.e. not change)
- **Setup time violation** This occurs if the input signal D does not settle (**set up**) to the stable value at least t_{setup} *before* the clock edge.
- **Hold time t_{hold}** Time *after* the clock edge that data must be stable
- **Hold time violation** This occurs if the input signal D does not remain unchanged (**hold**) for at least t_{hold} *after* the clock edge.

Sequential Networks: Timing Constraint (without skew)

- **Setup Time Constraint**

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

Longest delay from CLK1 to CLK2

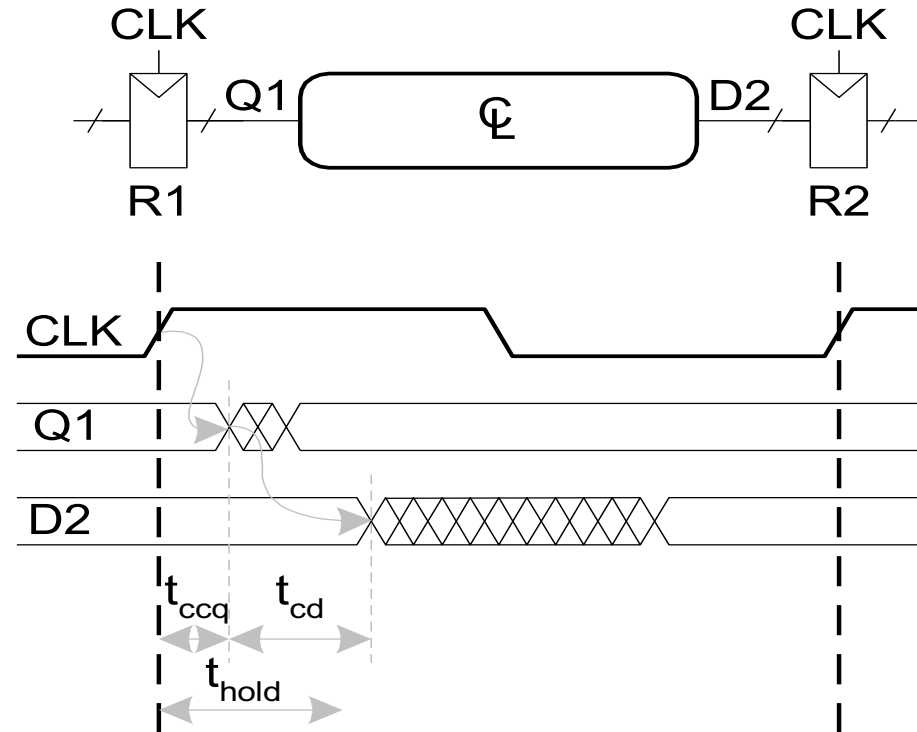


Sequential Networks: Timing Constraint (without skew)

- **Hold Time Constraint**

$$t_{ccq} + t_{cd} \geq t_{hold}$$

Shortest delay from CLK1 to CLK2



Sequential Networks: Timing Constraint (with skew)

- **Time Skew**

- Difference between two clock edges
- $t_{skew} = CLK2 - CLK1$

- **Setup Time Constraint**

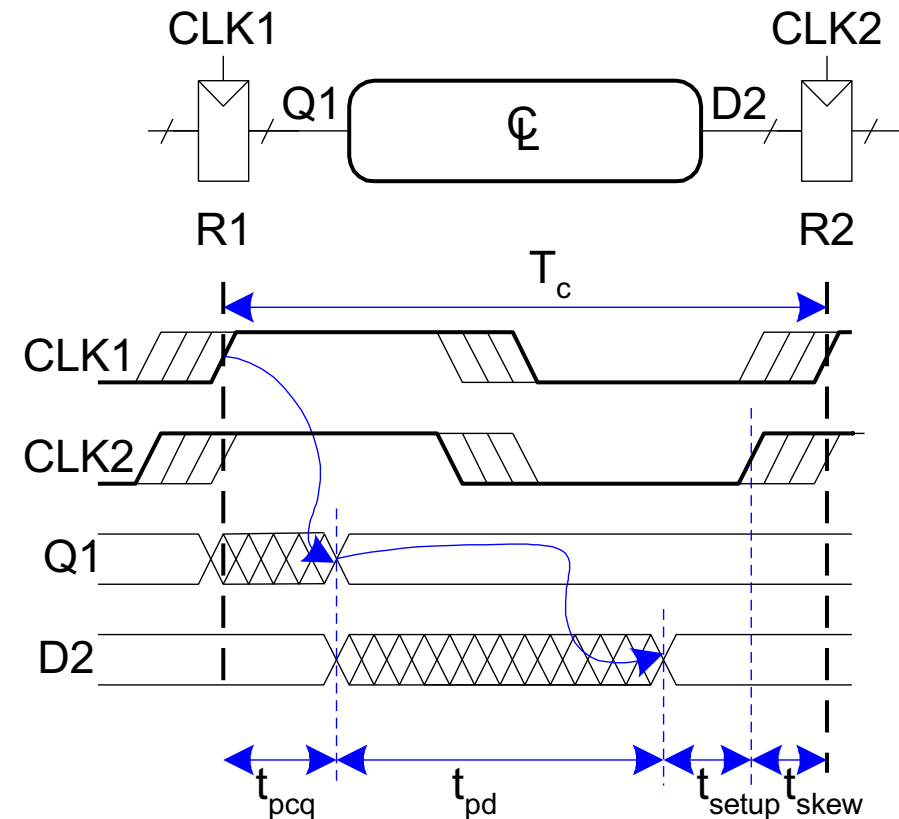
- CLK2 is earlier than CLK1 ($t_{skew} < 0$)

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

- **Hold Time Constraint**

- CLK2 is later than CLK1

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



Sequential Networks: Timing Constraint (with skew)

- **Time Skew**

- Difference between two clock edges
- $t_{skew} = CLK2 - CLK1$

- **Setup Time Constraint**

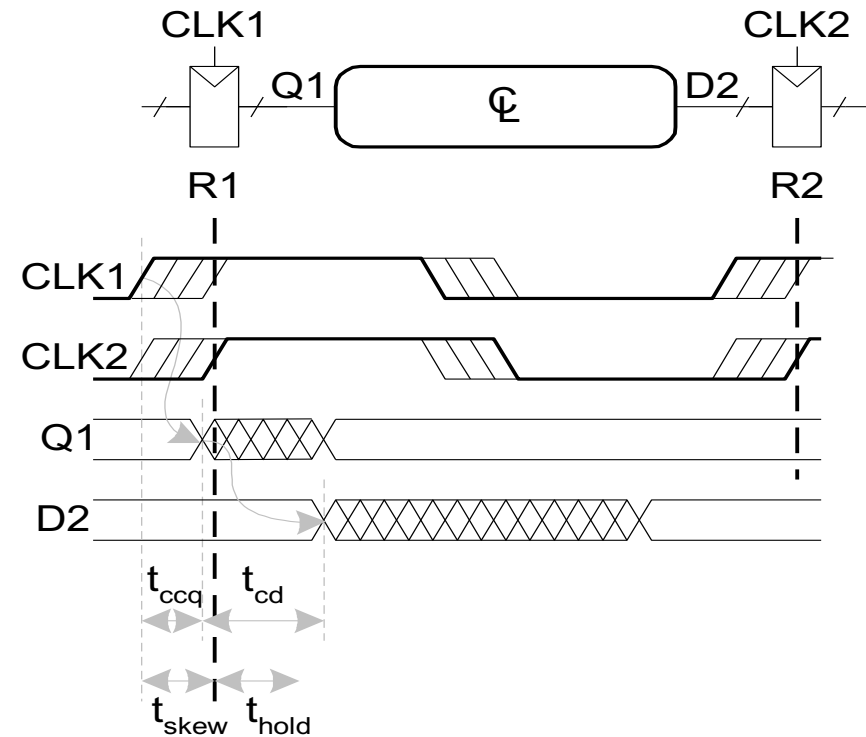
- CLK2 is earlier than CLK1 ($t_{skew} < 0$)

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

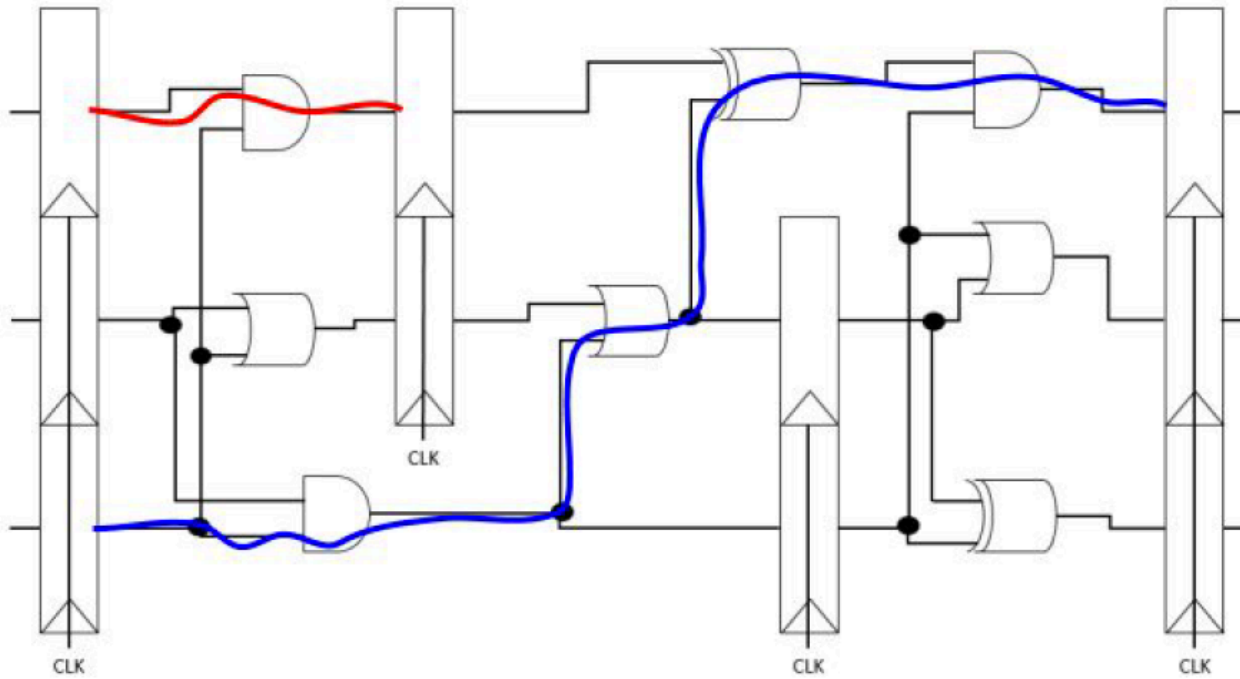
- **Hold Time Constraint**

- CLK2 is later than CLK1

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



Example: Q2 from mid2 sp17



- Propagation delay and contamination delay

✓ Logic gate (AND, OR, XOR)

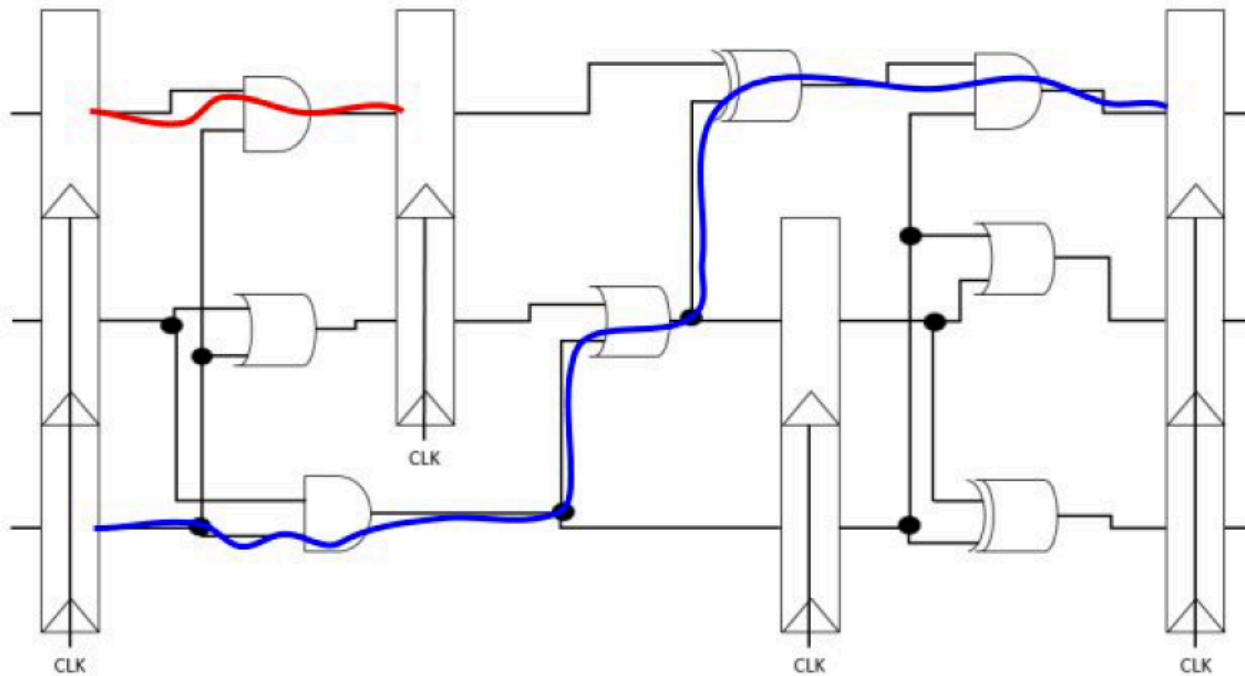
- $t_{pd} = 25ps$
- $t_{cd} = 15ps$

✓ Flip-Flop

- $t_{pcq} = 30ps$
- $t_{ccq} = 20ps$
- $t_{setup} = 20ps$
- $t_{hold} = 30ps$

Maximum frequency?

Example: Timing (Without Clock Skew)



- Find the **longest path** of combinational logic

$$t_{pd(max)} = 4 \times t_{pd(logic\ gate)}$$

- Setup Time Constraint

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

$$T_c \geq (30 + 4 \times 25 + 20) = 150ps$$

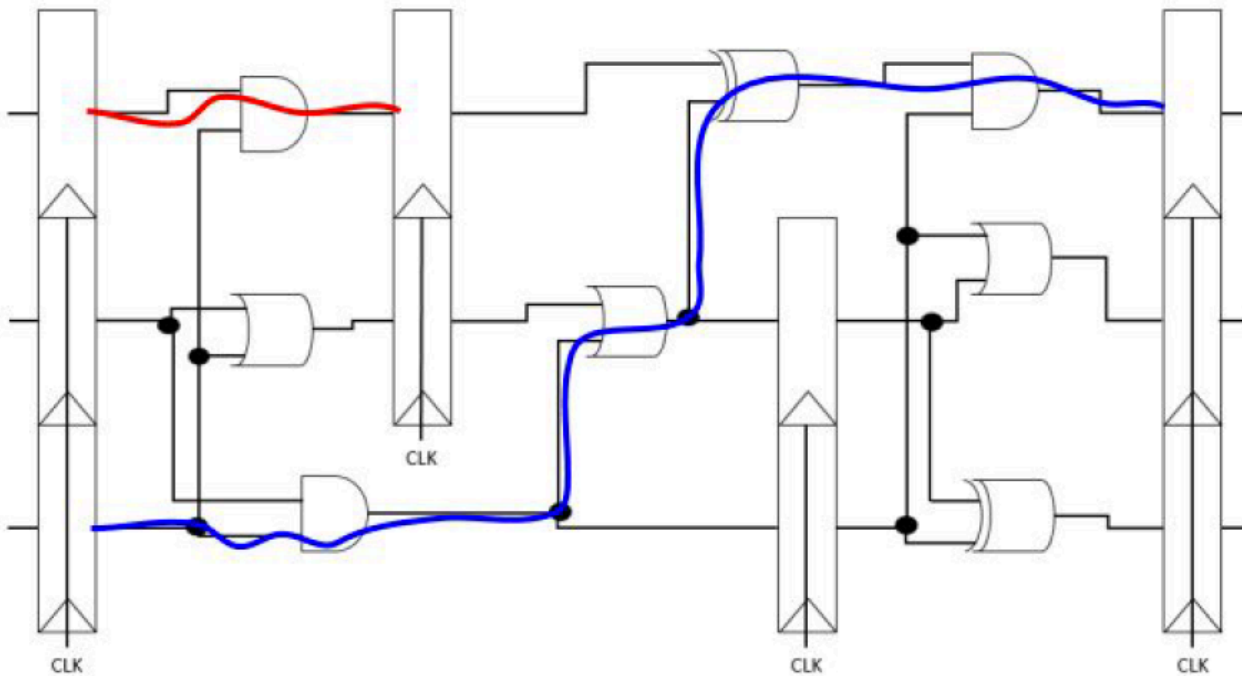
$$\text{the maximum frequency } \frac{1}{T} = 6.67GHz$$

- Hold Time Constraint

$$t_{cd(min)} = t_{cd(logic\ gate)} = 15ps$$

$$t_{ccq} + t_{cd(min)} = 20 + 15 = 35ps \geq t_{hold} = 30ps$$

Example: Timing (With Clock Skew)



- Hold Time Constraint

$$t_{ccq} + t_{cd(min)} = 35ps \geq t_{hold} + t_{skew}$$
$$= (30 + \Delta)ps$$

$$t_{skew} = 5ps \quad \text{CLK2 Later than CLK1 !!!}$$

- Setup Time Constraint

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

$$T_c \geq (150 - 5) = 145ps$$

$$\text{the maximum frequency } \frac{1}{T} = 6.90GHz$$

Good luck!