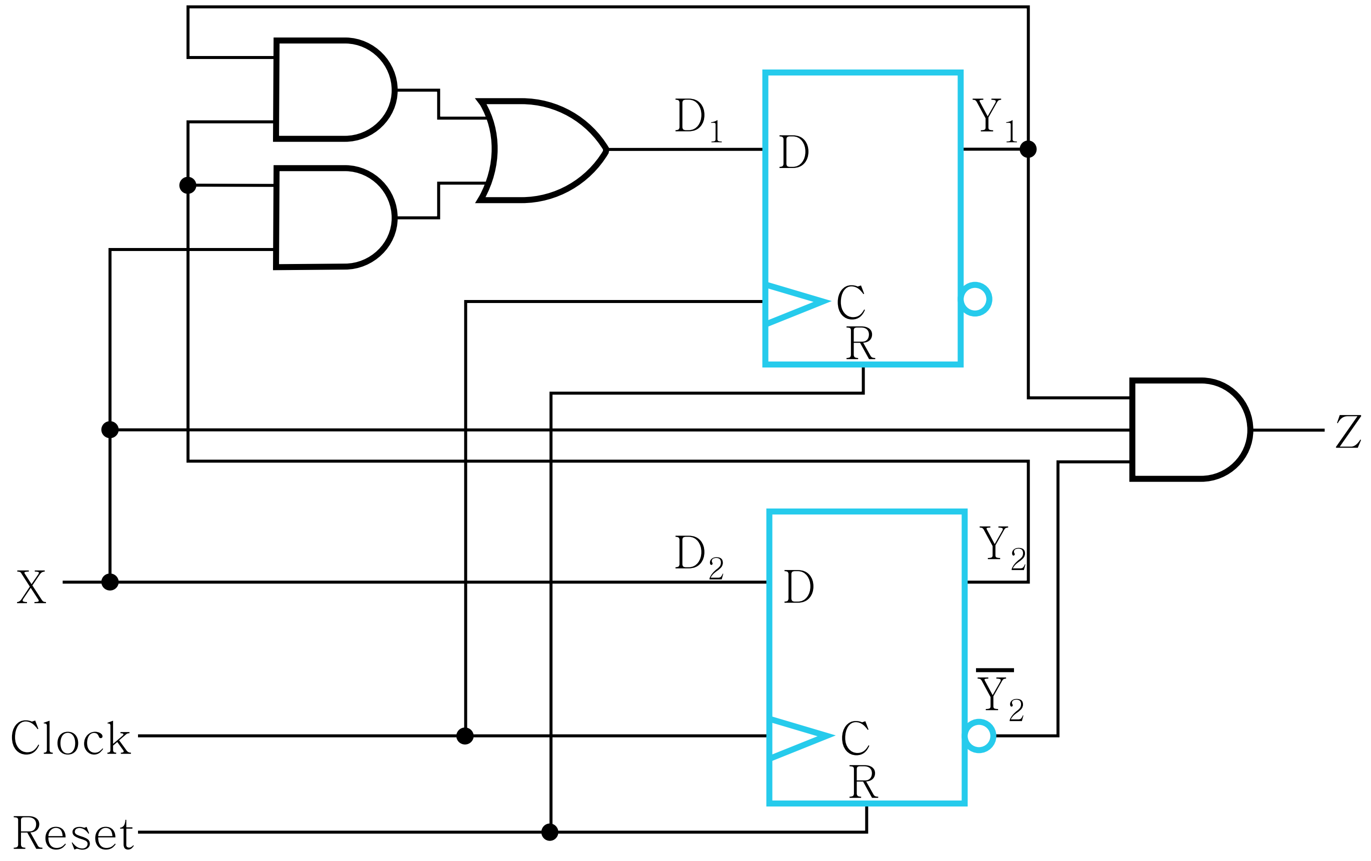
LECTURE 14

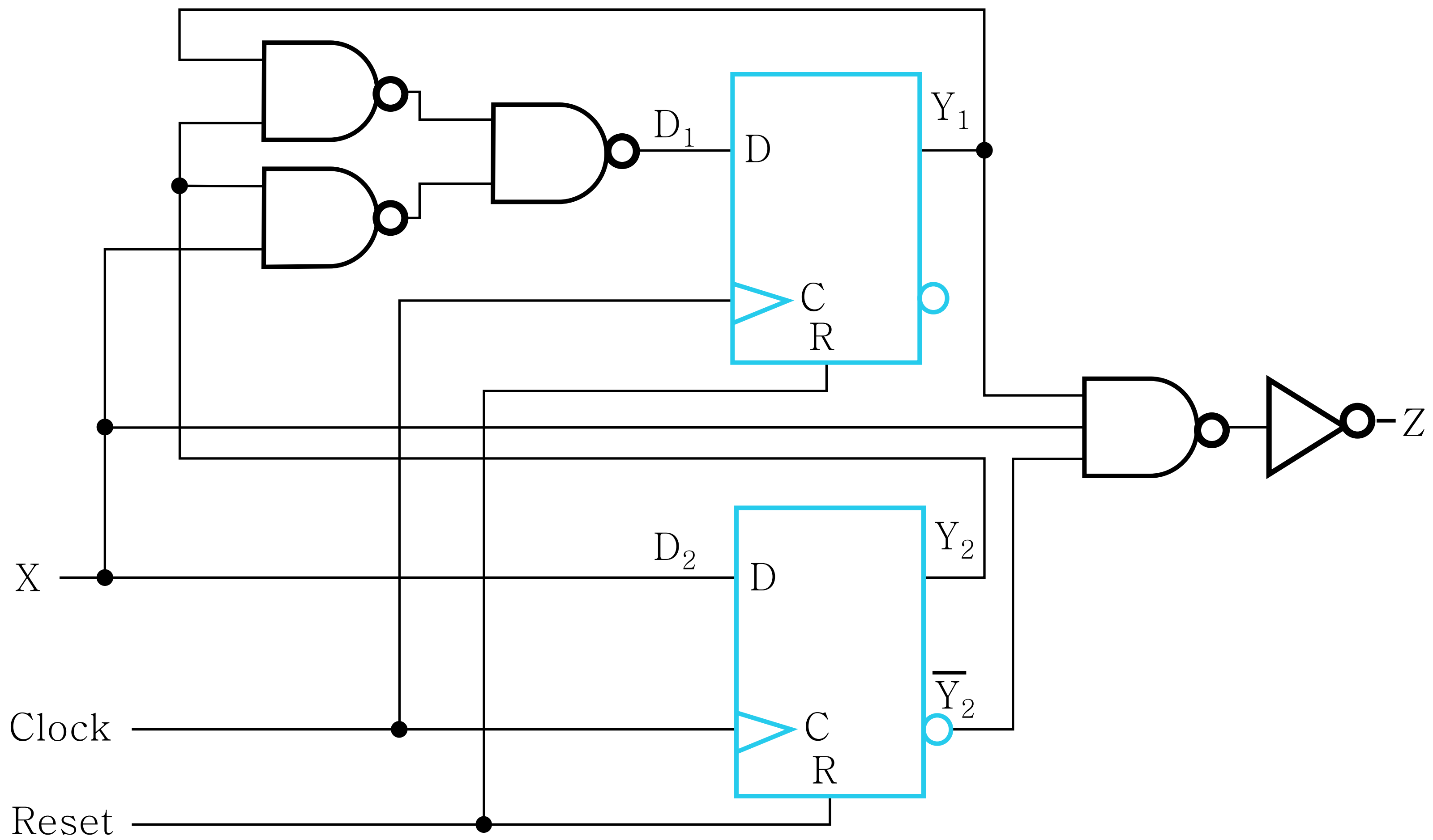
Digital Logic Families

Map to Technology

- Library: D-type Flip-Flops with Reset input
  - Reset input is used to reset to start state: Y_1 Y_2 = '00'



Circuit Implementation with NAND



Using SR, JK, and T Flip-Flop Types

- Characteristic table (used in analysis)
  - Defines the next state of the flip-flop in terms of flip-flop inputs and current state
- Characteristic equation (used also in analysis)
  - Obtained from characteristic table
  - Defines the next state of the flip-flop as a Boolean function of the flip-flop inputs and the current state
- Excitation table (used in design)
  - Defines the flip-flop input variable values as function of the current state and next state

SR Flip-Flop

- Characteristic Table

S	R	Q(t+1)	Operation
0	0	$Q(t)$	No change
0	1	0	Reset
1	0	1	Set
1	1	?	Undefined

- Excitation Table

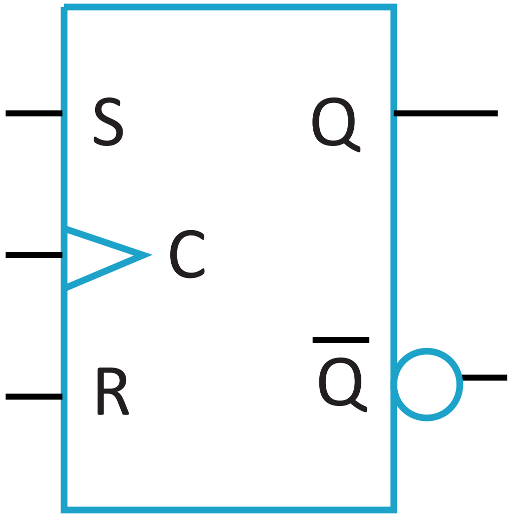
Q(t)	Q(t+1)	S	R	Operation
0	0	0	X	No change
0	1	1	0	Set
1	0	0	1	Reset
1	1	X	0	No change

- Characteristic Equation

  $Q(t+1) = S + \overline{R}\,Q(t)$

  S R = 0 (S and R cannot be 1 simultaneously)

Symbol



One-Hot Assignment

- Use one flip-flop per state: *m* states $\Rightarrow$ *m* flip-flops
  - $Y_3Y_2Y_1Y_0$ = 0001 (state A), 0010 (B), 0100 (C), 1000 (D)
- Flip-flop cost is higher but combinational logic might be simpler
- Provides simplified analysis and design
  - In equations, need to include only the variable that is 1 for the state, e. g., state with code 0~~00~~1~~, i~~s ~~re~~presented in equations by Y_0 instead of Y_3 Y_2 Y_1 Y_0 because if Y_0 is ‘1’ then the remaining state variables will be ‘0’

One-Hot State Assignment

- A = 0001, B = 0010, C = 0100, D = 1000
- The resulting coded state table:

Present State $Y_3\ Y_2\ Y_1\ Y_0$	Next State x = 0	x = 1	Output x = 0	x = 1
0001	0001	0010	0	0
0010	0001	0100	0	0
0100	1000	0100	0	0
1000	0001	0010	0	1

Optimization: One Hot Assignment

- No need for K-map, flip-flop input equations can be obtained directly from the state table
- Assume D Flip-Flops

  $D_0 = \overline{X}(Y_0 + Y_1 + Y_3)$ or $\overline{X}\,\overline{Y_2}$

  $D_1 = X(Y_0 + Y_3) = X\,\overline{Y_1}\,\overline{Y_2}$

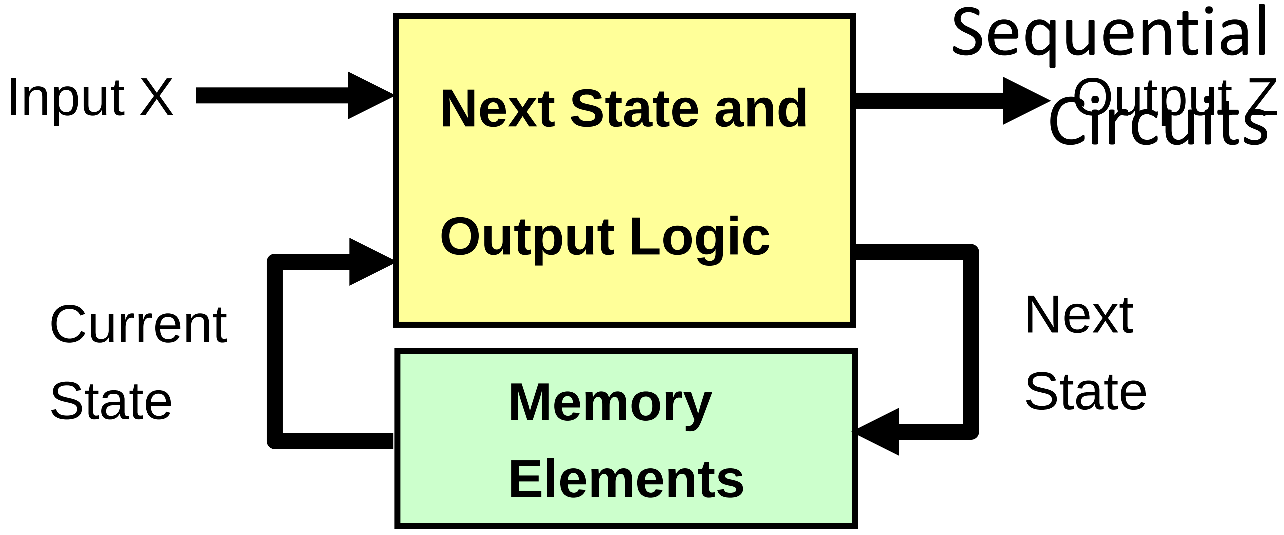
  $D_2 = X(Y_1 + Y_2) = X\,\overline{Y_0}\,\overline{Y_3}$

  $D_3 = \overline{X}\,Y_2$

  $Z = XY_3$ Gate Input Cost = 12
- Total cost = combinational circuit cost + cost of four flip-flops

Mealy and Moore Sequential Circuits

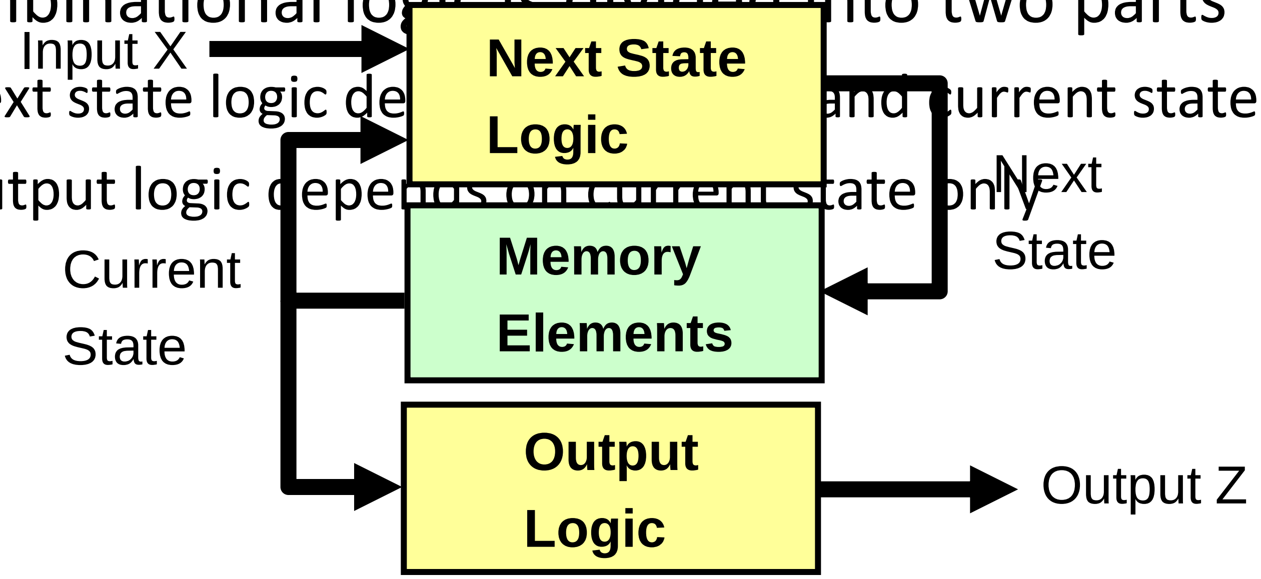
- Two ways to design clocked sequential circuits
  - Mealy and Moore type sequential circuits
- Mealy type sequential circuit
  - Output is a function of current state and input



  - Example: 1101 sequence detector discussed above

Moore Type Sequential Circuits

- Output depends on current state only
  - Output does not depend on input
- Combinational logic is divided into two parts
  - Next state logic de and current state
  - Output logic depends on current state only

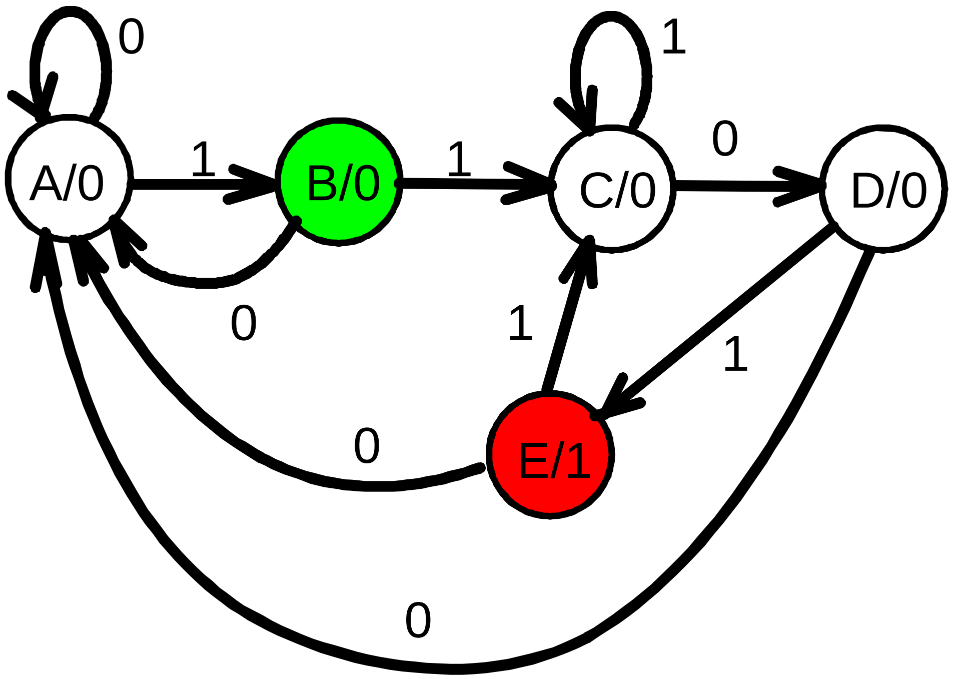


Moore Model for Sequence 1101 Detector

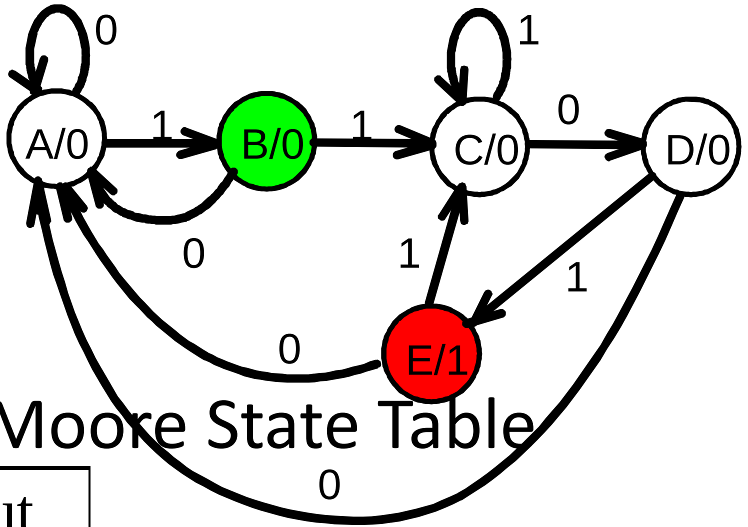
- For the Moore Model, outputs depend on states
- We need to add a state E with output value ‘1’ for the final ‘1’ in the recognized input sequence
  - This new state E, though similar to B, would generate an output of ‘1’ and thus be different from state B
- The Moore model for a sequence recognizer usually has *more states* than the Mealy model

Moore State Diagram

- We mark outputs on states for Moore model
- For Mealy, outputs were marked on arcs
- Arcs now show state transitions and input only
- Add a new state E to produce the output 1
- Note that the new state E produces the same behavior as state B, but gives a different output: ‘1’ rather than ‘0’



- State and output tables are shown below
- Observe that output y does not depend on input x



Moore State Table

Present State	Next State		Output
	x=0	x=1	y
A	A	B	0
B	A	C	0
C	D	C	0
D	A	E	0
E	A	C	1

Moore state diagram typically results in More states

State Assignment for Moore Detector

- As in t uence detec nment and o
- Using uired

Present State	Next State		Output
$Y_4\,Y_3\,Y_2\,Y_1\,Y_0$	x=0	x=1	**Z**
A = 0 0 0 0 1	**0 0 0 0 1**	**0 0 0 1 0**	**0**
B = 0 0 0 1 0	**0 0 0 0 1**	**0 0 1 0 0**	**0**
C = 0 0 1 0 0	**0 1 0 0 0**	**0 0 1 0 0**	**0**
D = 0 1 0 0 0	**0 0 0 0 1**	**1 0 0 0 0**	**0**
E = 1 0 0 0 0	**0 0 0 0 1**	**0 0 1 0 0**	**1**

Equations for Moore Sequence Detector

- Using D Flip Flops, input equations:

  $D_0 = X (Y_0 + Y_1 + Y_3 + Y_4) = X Y_2$

  $D_1 = X Y_0$

  $D_2 = X (Y_1 + Y_2 + Y_4) = X Y_0 Y_3$

  $D_3 = X Y_2$

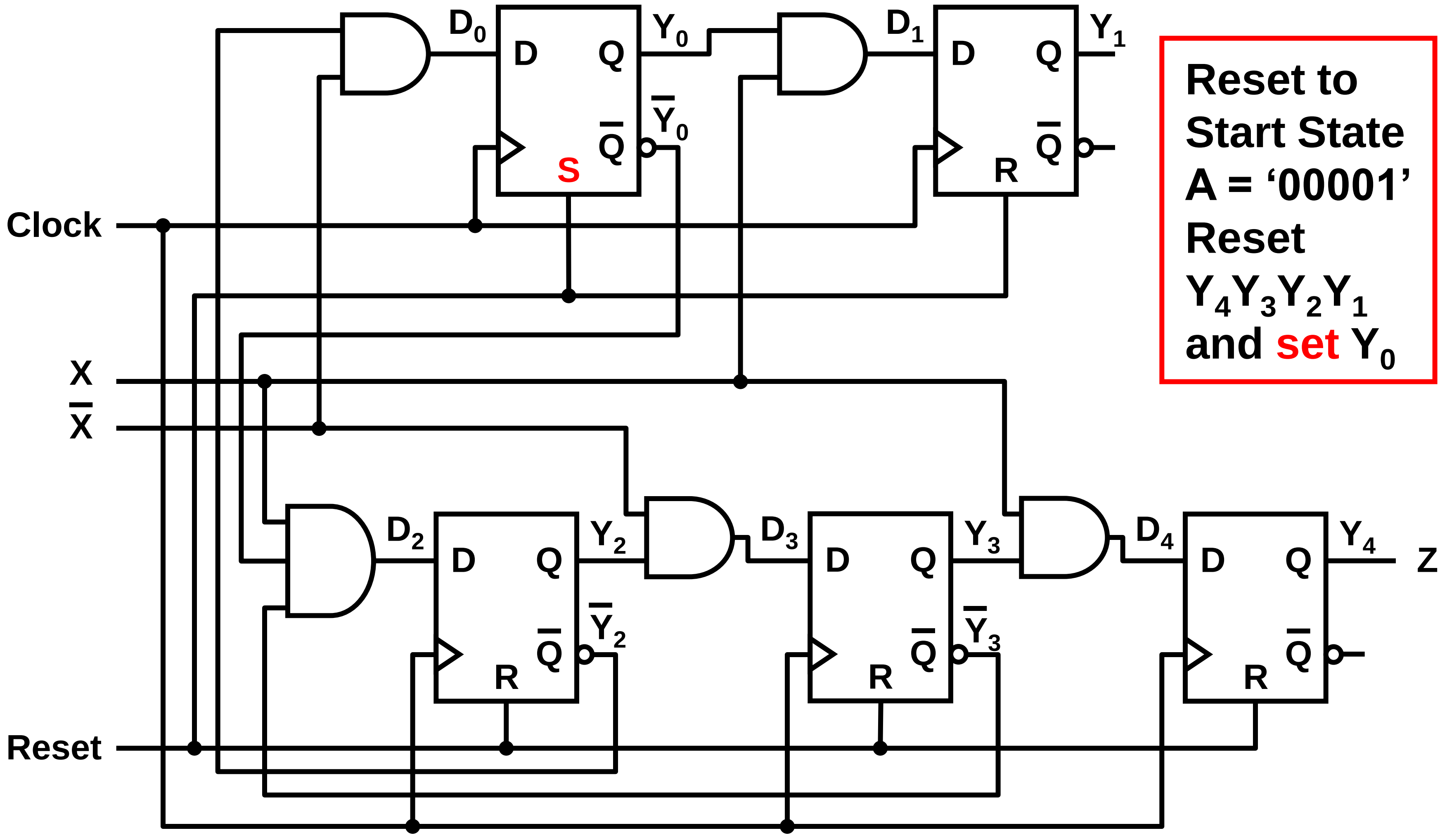
  $D_4 = X Y_3$

- Output Equation Z

  $Z = Y_4$

- Gate input cost = 11

Circuit Implementation



Reset to Start State A = '00001' Reset $Y_4Y_3Y_2Y_1$ and set Y_0

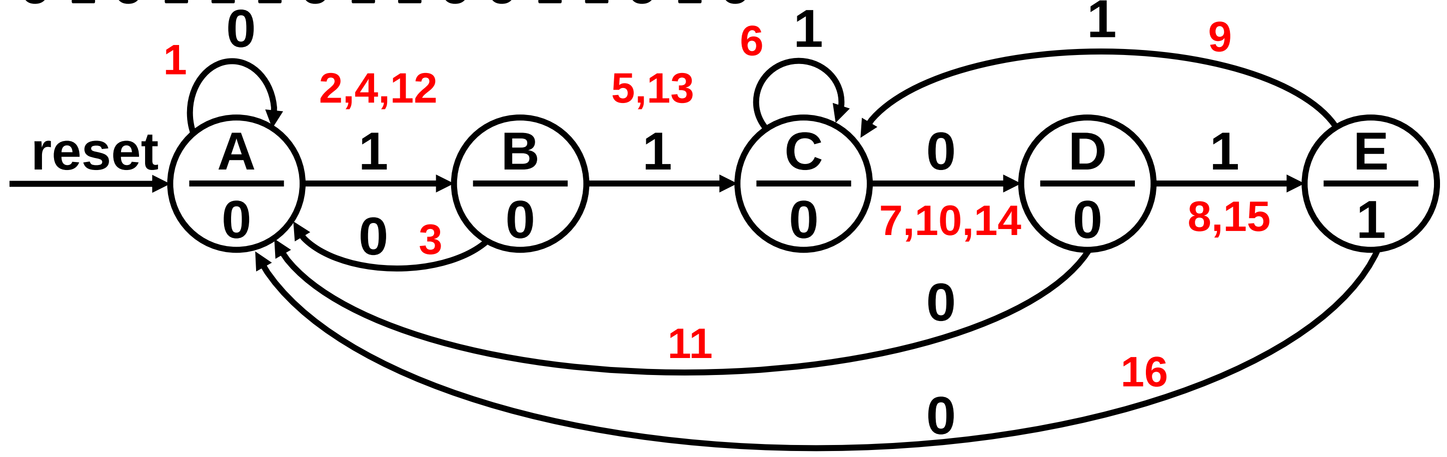
Verification

- Sequential circuits should be verified by showing that the circuit produces the original state diagram
- Verification can be done manually, or with the help of a simulation program
- All possible input combinations are applied at each state and the state variables and outputs are observed
- A reset input is used to reset the circuit to its initial state
- Apply a sequence of inputs to test all the state-input combinations, i.e., all transitions in the state diagram
- Observe the output and the next state that appears after each clock edge in the timing diagram

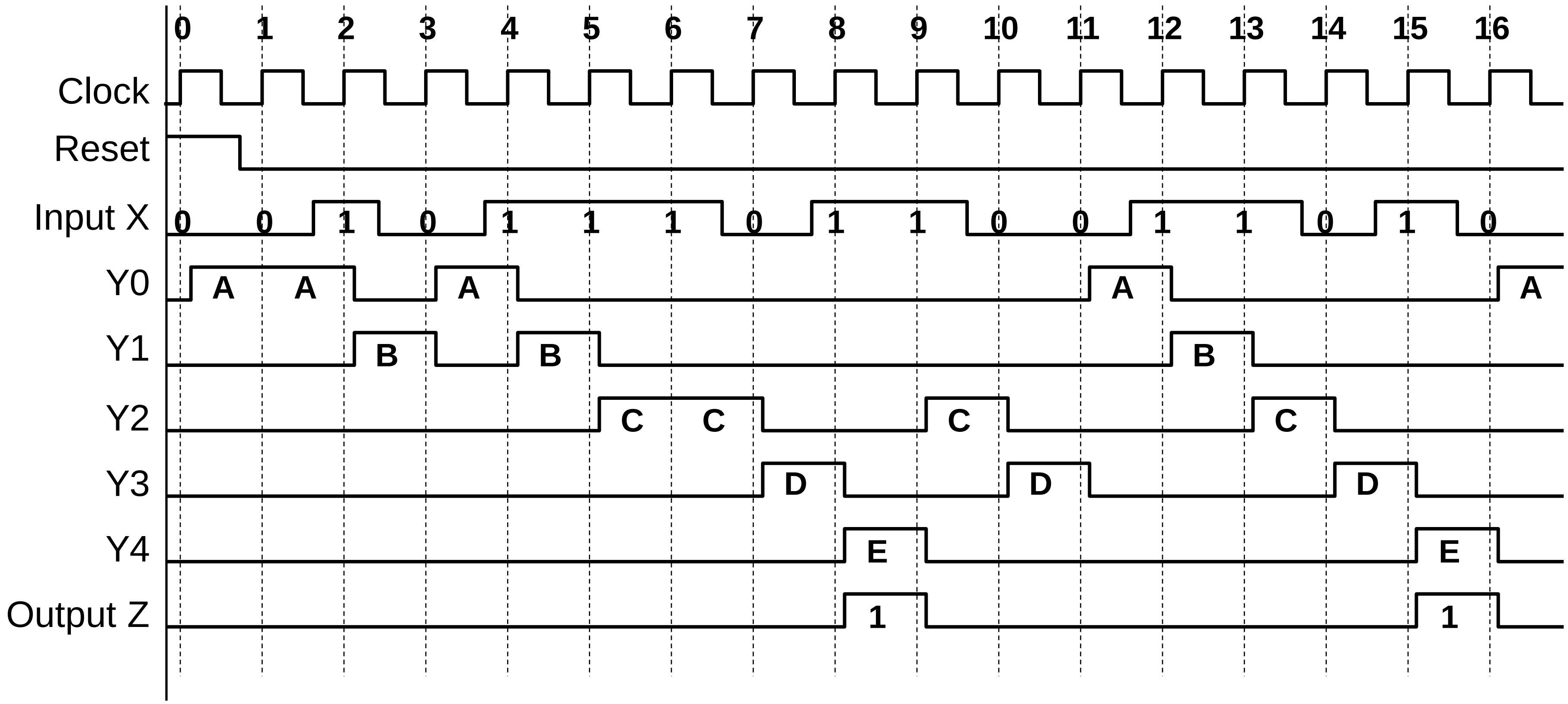
Input Test Sequence

- An input test sequence is required to verify the correct operation of a sequential circuit
- It should test each state transition of the state diagram
- Test sequences can be generated from the state diagram
- Consider the Moore sequence detector. Starting at A (after reset), we can generate an input test sequence:

X = 0 1 0 1 1 1 0 1 1 0 0 1 1 0 1 0



Verifying the Moore Sequence Detector

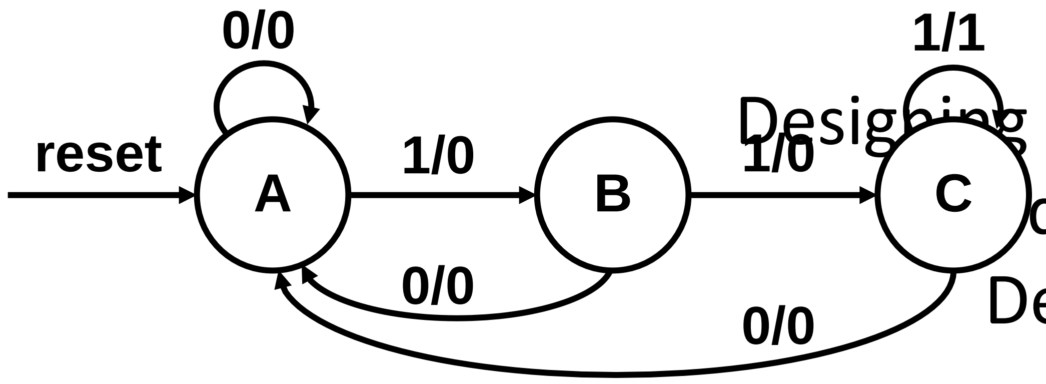


Moore versus Mealy Sequential Circuits

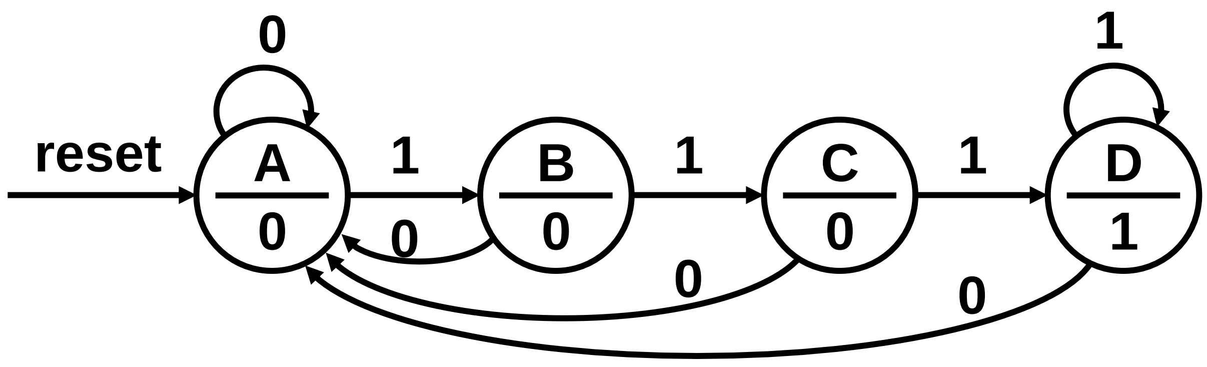
- Output in a Moore sequential circuit is associated with a state, while output in a Mealy circuit is associated with a transition between states
- In general, Moore state diagrams have more states than corresponding Mealy state diagrams and the Moore sequential circuit implementation might have higher cost
- Since the output in a Mealy machine is a combination of present state and input values, an unsynchronized input may result in an invalid output (drawback of Mealy)
- A Moore state diagram produces a unique output for every state irrespective of inputs. Output of a Moore machine is synchronized with the clock (better)

Designing a ‘111’ Sequence Detector

- To illustrate the drawback of the Mealy machine, consider the design of a ‘111’ sequence detector



Mealy State Diagram



Moore State Diagram

State Assignment and Equations

- A minimum of 2 state variables are required
- Using Gray Code state assignment and D flip flops

Mealy State Table

Present State	Next State		Output	
Y_1 Y_0	x=0	x=1	x=0	x=1
A = 0 0	**0 0**	**0 1**	**0**	**0**
B = 0 1	**0 0**	**1 1**	**0**	**0**
C = 1 1	**0 0**	**1 1**	**0**	**1**

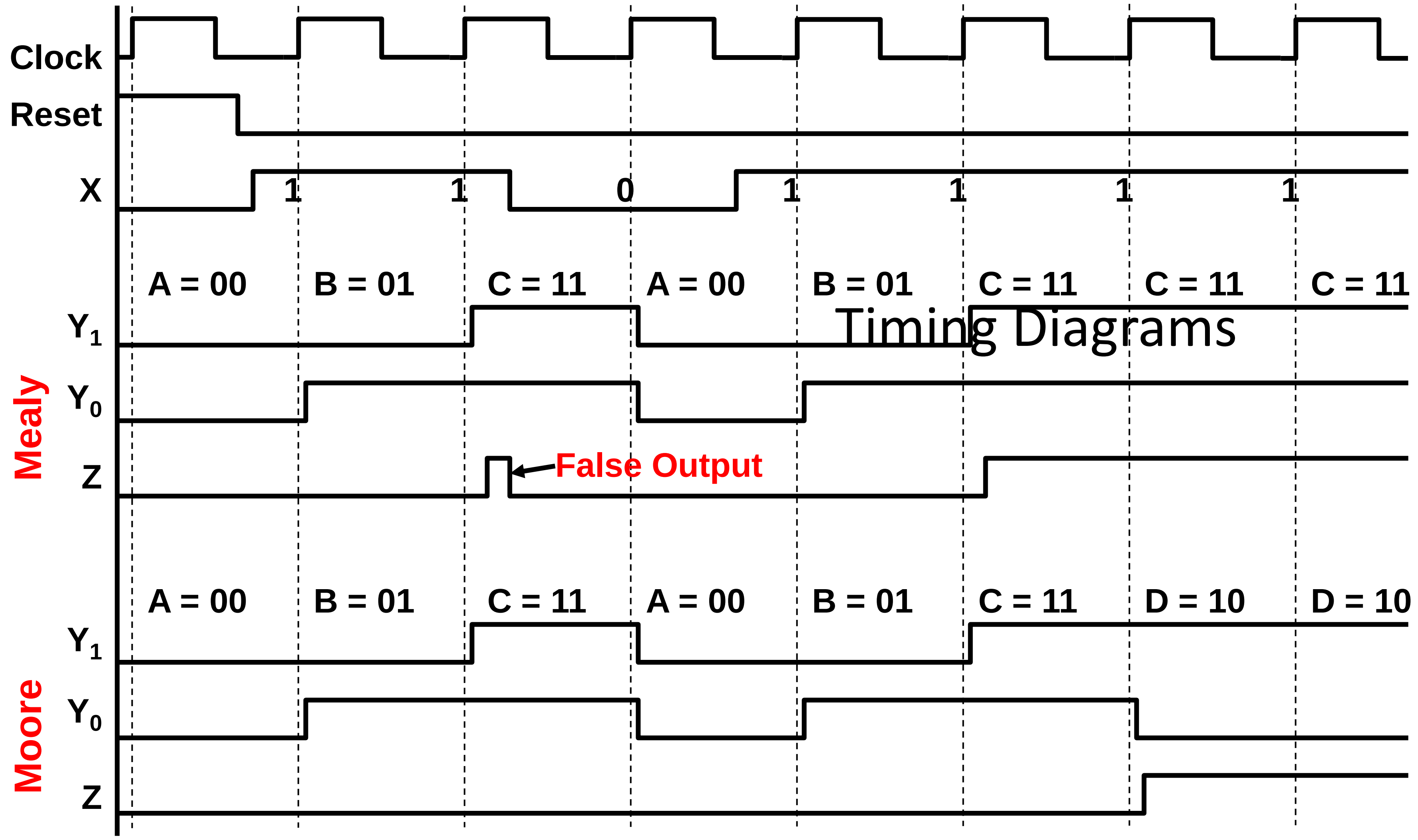
Moore State Table

Present State	Next State		Output
Y_1 Y_0	x=0	x=1	**Z**
A = 0 0	**0 0**	**0 1**	**0**
B = 0 1	**0 0**	**1 1**	**0**
C = 1 1	**0 0**	**1 0**	**0**
D = 1 0	**0 0**	**1 0**	**1**

- $D_1 = X\,Y_0$
- $D_0 = X$
- $Z = X\,Y_1$

- $D_1 = X\,(Y_0 + Y_1)$
- $D_0 = X\,\overline{Y}_1$ $\quad Z = Y_1\,\overline{Y}_0$

Timing Diagrams


Clock
Reset
X
1
1
0
1
1
1
1
A = 00
B = 01
C = 11
A = 00
B = 01
C = 11
C = 11
C = 11
Y1
Y0
Z
False Output
Mealy
A = 00
B = 01
C = 11
A = 00
B = 01
C = 11
D = 10
D = 10
Y1
Y0
Z
Moore