Introduction to Finite State Machines and their Design

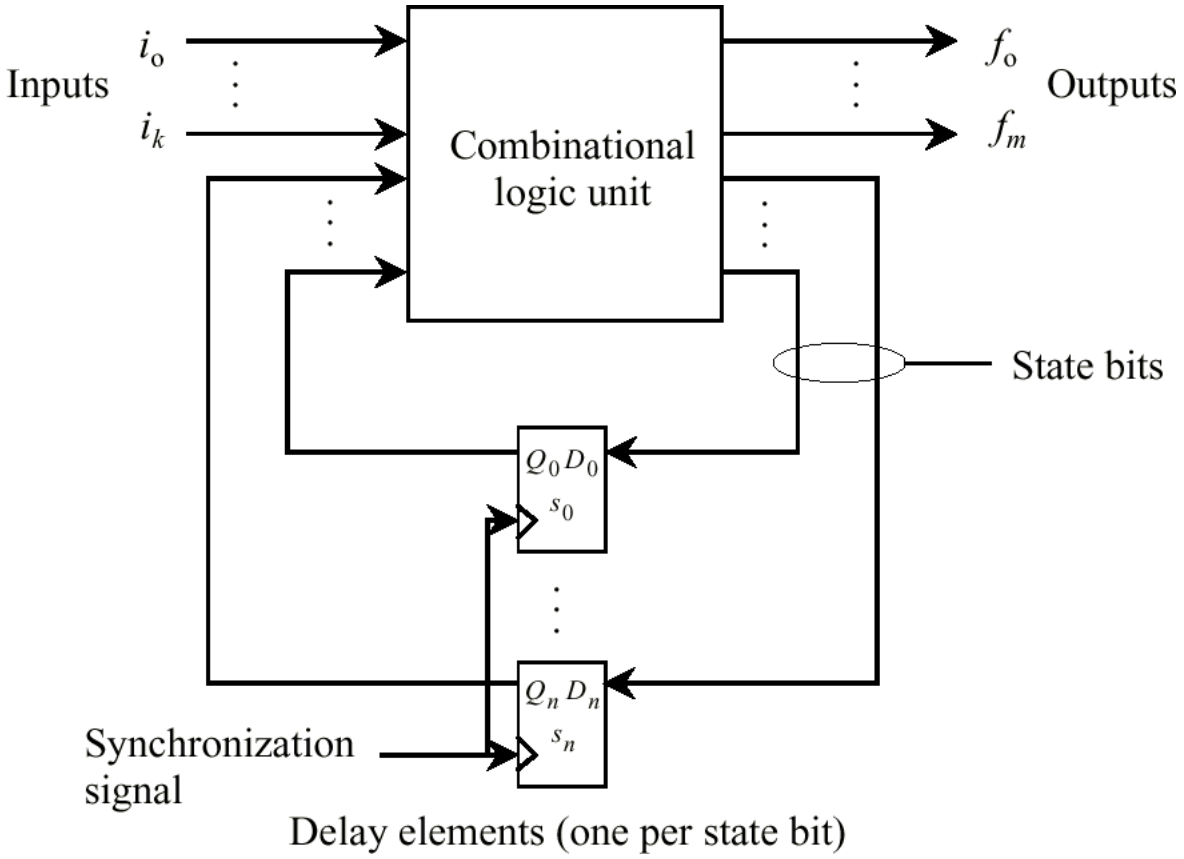
Review of Objectives

After this lecture, you should be able to……

- **Draw a block diagram of a general FSM**
- **Design a finite state machine from a functional description by**
  - **Drawing a state transition diagram**
  - **Deriving a state table from the state transition diagram**
  - **Assigning states, determining # of FF's required**
  - **Drawing up a truth table**
  - **Designing logic for combinational logic unit**
  - **Drawing the complete schematic**
- **State the differences in Mealy and Moore FSM's**
- **Build module 2N counters using JK FF's**

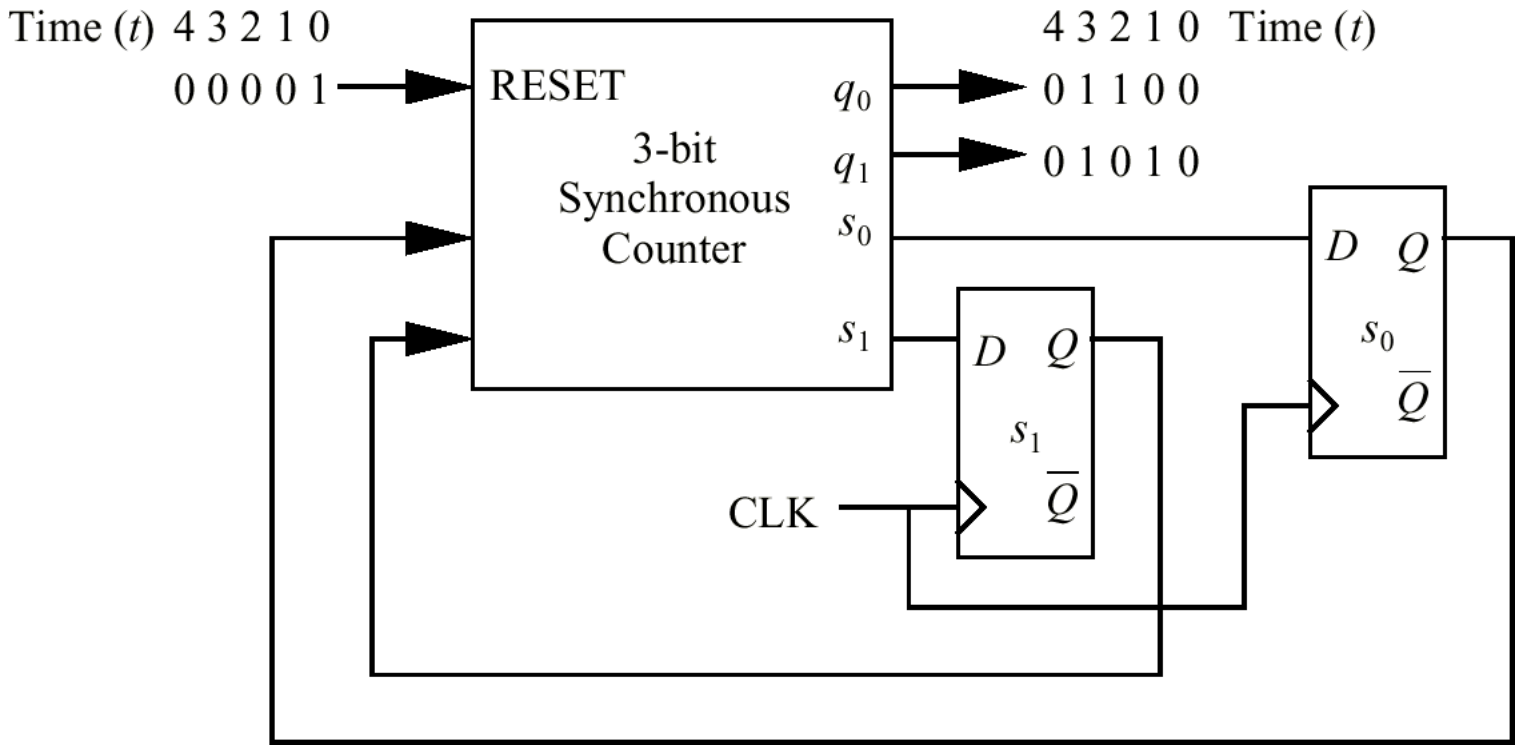
Classical Model of a Finite State Machine

- **An FSM is composed of a combinational logic unit and delay elements (called *flip-flops*) in a feedback path, which maintains state information.**
- **This particular FSM is based on the Mealy Model.**

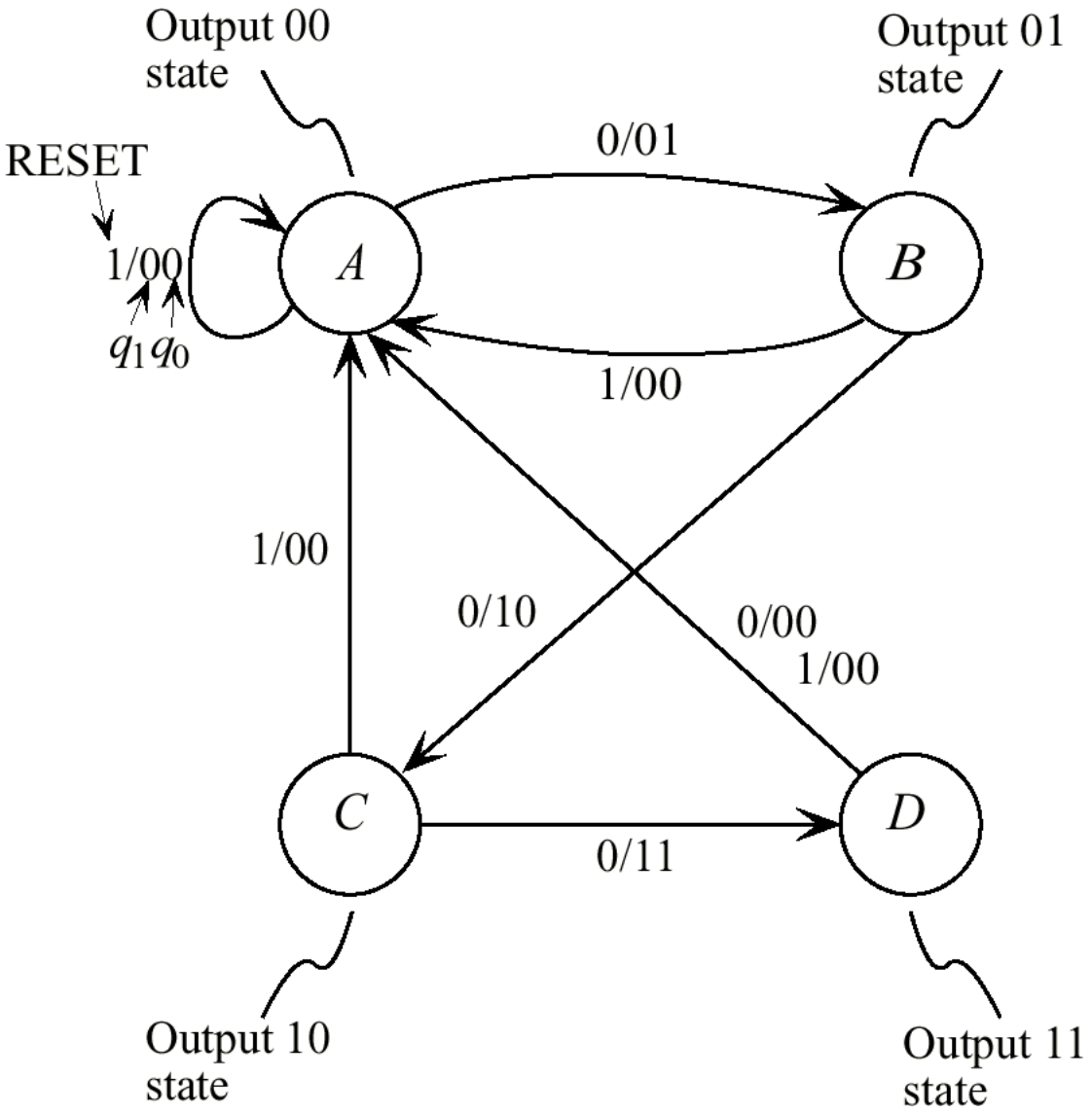


Example: Modulo-4 Counter

- **Counter has a clock input (CLK) and a RESET input.**
- **Counter has two output lines, which take on values of 00, 01, 10, and 11 on subsequent clock cycles.**



State Transition Diagram for Mod-4 Counter



State Table for Mod-4 Counter

Present state \ Input	*RESET* = 0	*RESET* = 1
A	*B*/01	*A*/00
B	*C*/10	*A*/00
C	*D*/11	*A*/00
D	*A*/00	*A*/00

Next state

Output

State Assignment for Mod-4 Counter

Present state (S_t) \ Input	*RESET* 0	*RESET* 1
A:00	01/01	00/00
B:01	10/10	00/00
C:10	11/11	00/00
D:11	00/00	00/00

Truth Table for Mod-4 Counter

RESET $r(t)$	$s_1(t)$	$s_0(t)$	$s_1s_0(t+1)$	$q_1q_0(t+1)$
0	0	0	01	01
0	0	1	10	10
0	1	0	11	11
0	1	1	00	00
1	0	0	00	00
1	0	1	00	00
1	1	0	00	00
1	1	1	00	00

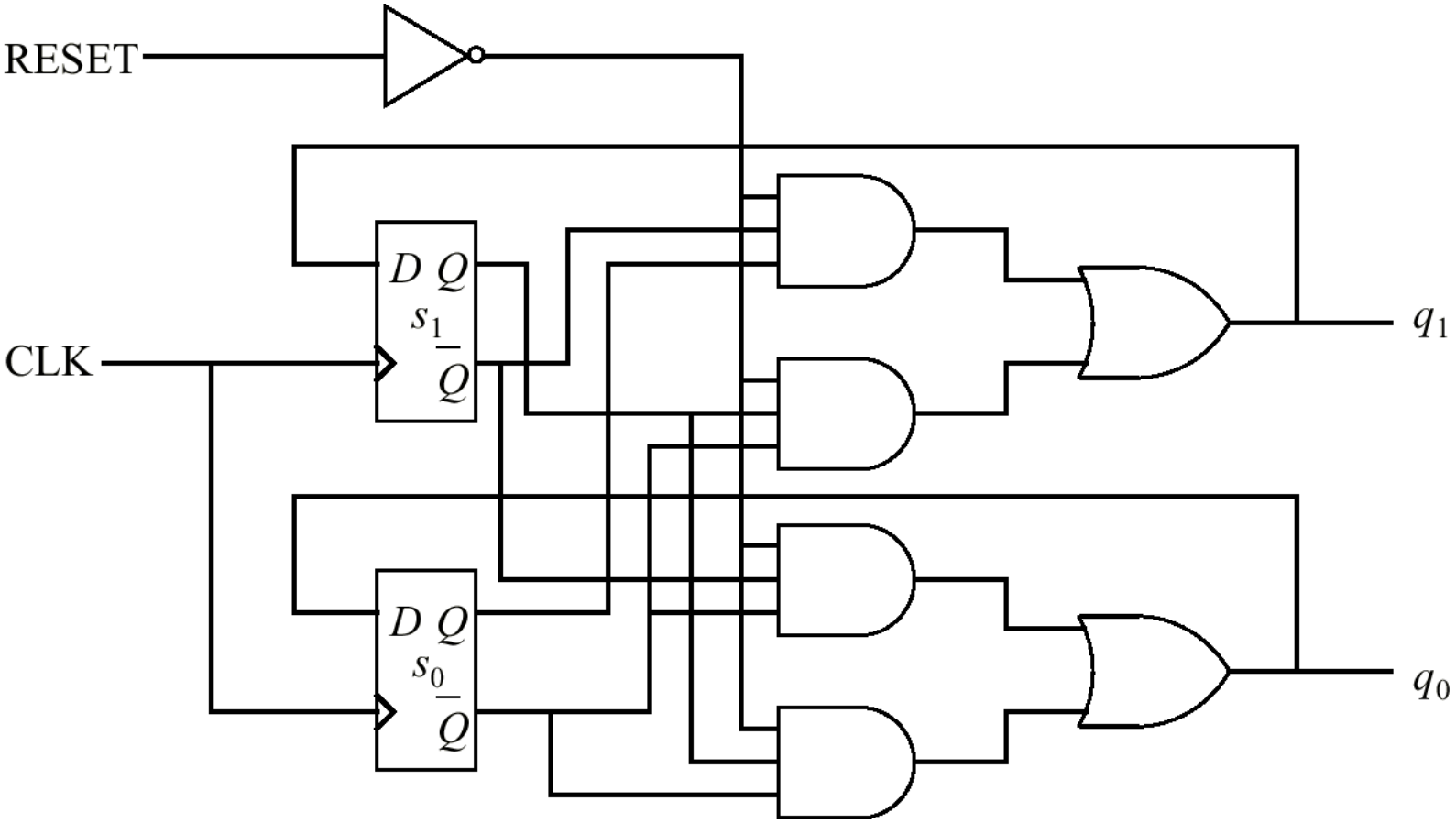
$$s_0(t+1) = \overline{r(t)}\,\overline{s_1(t)}\,\overline{s_0(t)} + \overline{r(t)}\,s_1(t)\,\overline{s_0(t)}$$

$$s_1(t+1) = \overline{r(t)}\,\overline{s_1(t)}\,s_0(t) + \overline{r(t)}\,s_1(t)\,\overline{s_0(t)}$$

$$q_0(t+1) = \overline{r(t)}\,\overline{s_1(t)}\,\overline{s_0(t)} + \overline{r(t)}\,s_1(t)\,\overline{s_0(t)}$$

$$q_1(t+1) = \overline{r(t)}\,\overline{s_1(t)}\,s_0(t) + \overline{r(t)}\,s_1(t)\,\overline{s_0(t)}$$

Logic Design for Mod-4 Counter

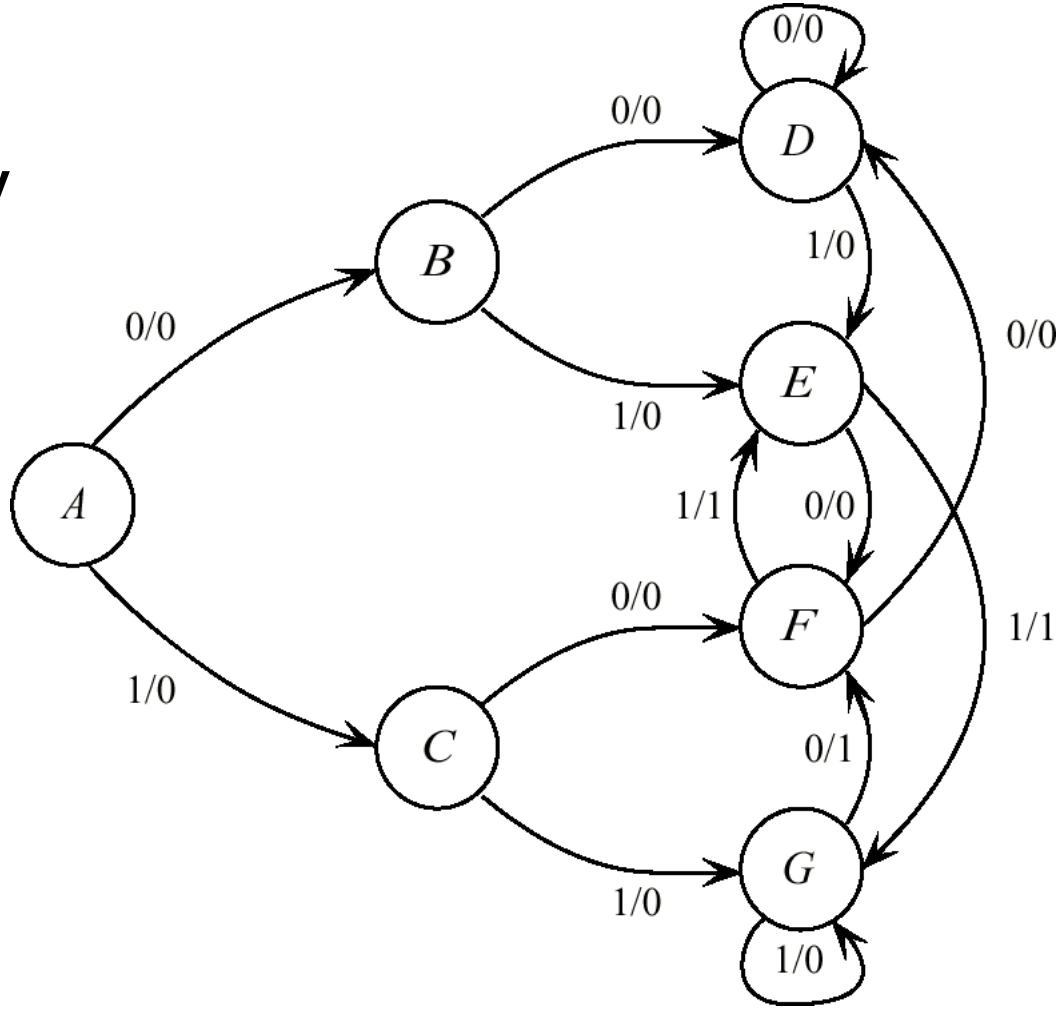


Example: A Sequence Detector

- **Example: Design a machine that outputs a 1 when exactly two of the last three inputs are 1.**
- ***e.g.* input sequence of 011011100 produces an output sequence of 001111010.**
- **Assume input is a 1-bit serial line.**
- **Use D flip-flops and 8-to-1 Multiplexers.**
- **Start by constructing a state transition diagram (next slide).**

Sequence Detector State Transition Diagram

- **Design a machine that outputs a 1 when exactly two of the last three inputs are 1.**



Sequence Detector State Table

Present state \ Input	X = 0	X = 1
A	B/0	C/0
B	D/0	E/0
C	F/0	G/0
D	D/0	E/0
E	F/0	G/1
F	D/0	E/1
G	F/1	G/0

Sequence Detector State Assignment

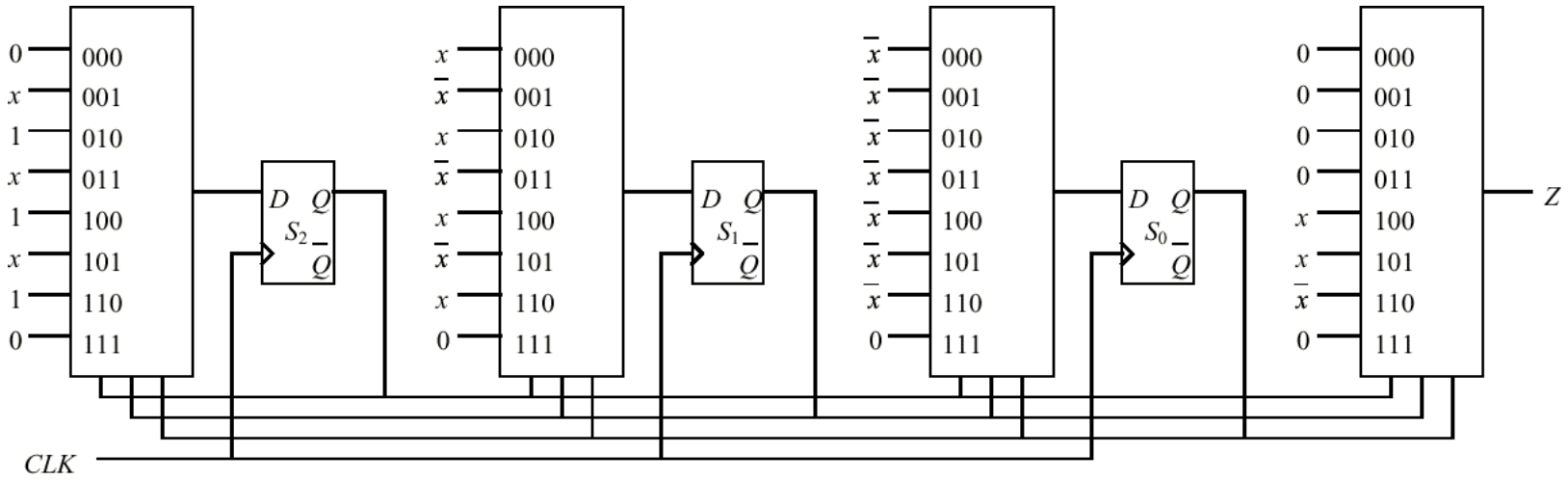
Present state \ Input	X = 0	X = 1
$S_2S_1S_0$	$S_2S_1S_0Z$	$S_2S_1S_0Z$
A: 000	001/0	010/0
B: 001	011/0	100/0
C: 010	101/0	110/0
D: 011	011/0	100/0
E: 100	101/0	110/1
F: 101	011/0	100/1
G: 110	101/1	110/0

(a)

Input and state at time t				Next state and output at time t+1			
s_2	s_1	s_0	x	s_2	s_1	s_0	z
0	0	0	0	0	0	1	0
0	0	0	1	0	1	0	0
0	0	1	0	0	1	1	0
0	0	1	1	1	0	0	0
0	1	0	0	1	0	1	0
0	1	0	1	1	1	0	0
0	1	1	0	0	1	1	0
0	1	1	1	1	0	0	0
1	0	0	0	1	0	1	0
1	0	0	1	1	1	0	1
1	0	1	0	0	1	1	0
1	0	1	1	1	0	0	1
1	1	0	0	1	0	1	1
1	1	0	1	1	1	0	0
1	1	1	0	d	d	d	d
1	1	1	1	d	d	d	d

(b)

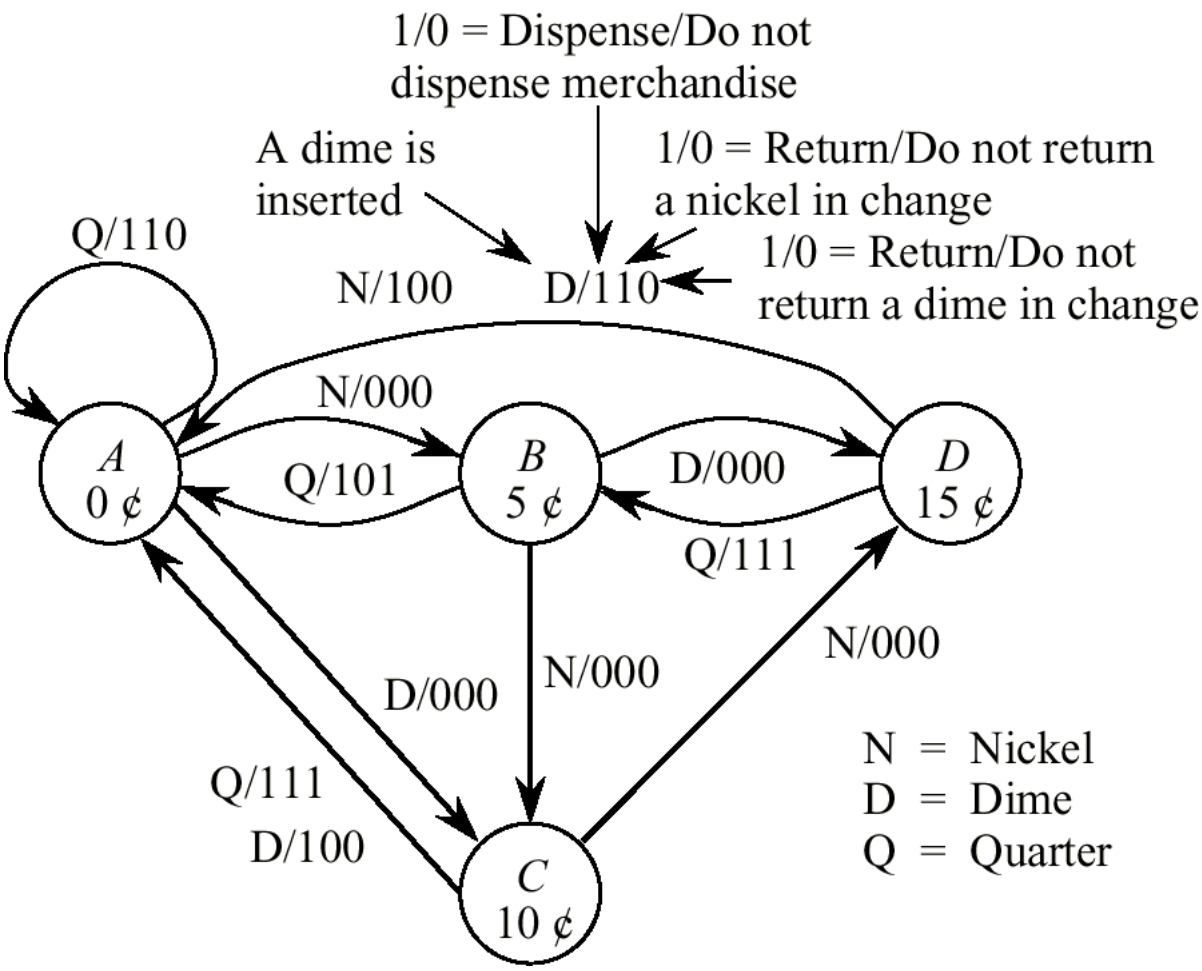
Sequence Detector Logic Diagram



Example: A Vending Machine Controller

- **Example: Design a finite state machine for a vending machine controller that accepts nickels (5 cents each), dimes (10 cents each), and quarters (25 cents each). When the value of the money inserted equals or exceeds twenty cents, the machine vends the item and returns change if any, and waits for next transaction.**
- **Implement with PLA and D flip-flops.**

Vending Machine State Transition Diagram



Vending Machine State Table and State Assignment

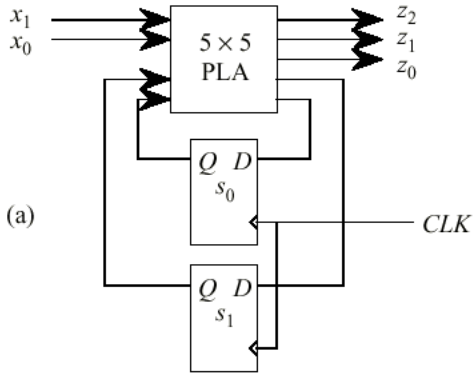
Input / P.S.	N 00	D 01	Q 10
A	*B*/000	*C*/000	*A*/110
B	*C*/000	*D*/000	*A*/101
C	*D*/000	*A*/100	*A*/111
D	*A*/100	*A*/110	*B*/111

(a)

Input / P.S.	N x_1x_0 00	D x_1x_0 01	Q x_1x_0 10
s_1s_0	s_1s_0 / $z_2z_1z_0$		
A:00	01/000	10/000	00/110
B:01	10/000	11/000	00/101
C:10	11/000	00/100	00/111
D:11	00/100	00/110	01/111

(b)

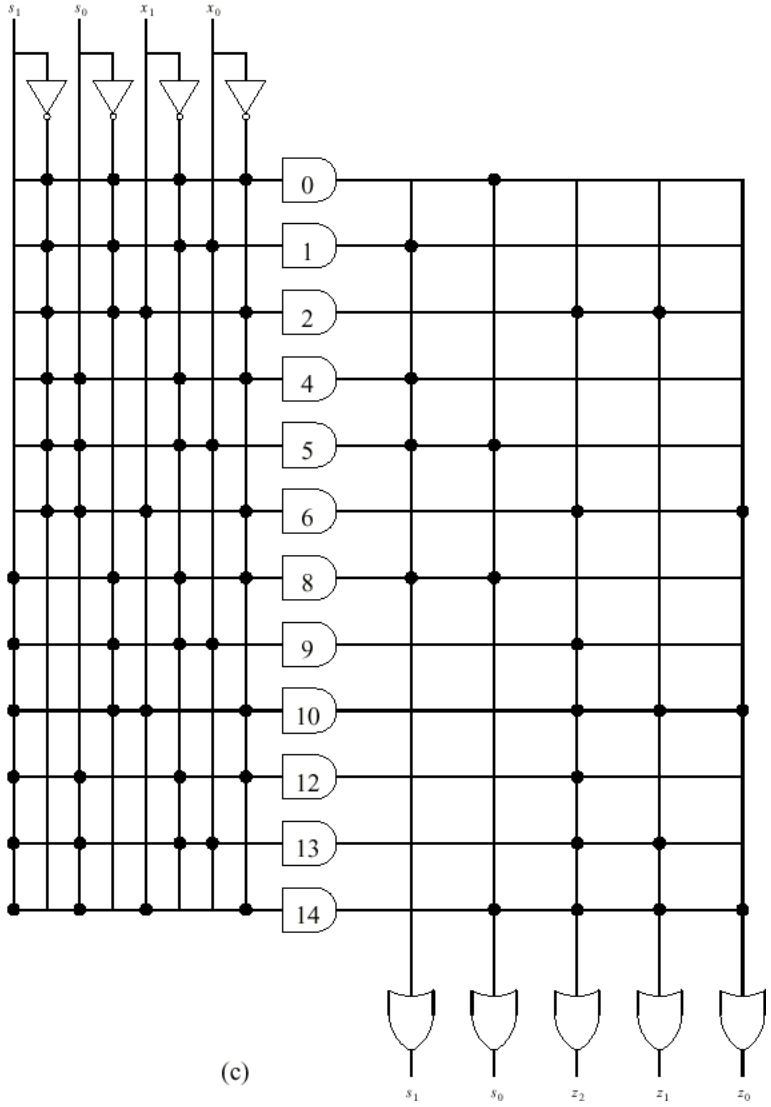
PLA Vending Machine Controller



(a)

Base 10 equivalent	Present state s_1	s_0	Coin x_1	x_0	Next state s_1	s_0	Dispense z_2	Return nickel z_1	Return dime z_0
0	0	0	0	0	0	1	0	0	0
1	0	0	0	1	1	0	0	0	0
2	0	0	1	0	0	0	1	1	0
3	0	0	1	1	d	d	d	d	d
4	0	1	0	0	1	0	0	0	0
5	0	1	0	1	1	1	0	0	0
6	0	1	1	0	0	0	1	0	1
7	0	1	1	1	d	d	d	d	d
8	1	0	0	0	1	1	0	0	0
9	1	0	0	1	0	0	1	0	0
10	1	0	1	0	0	0	1	1	1
11	1	0	1	1	d	d	d	d	d
12	1	1	0	0	0	0	1	0	0
13	1	1	0	1	0	0	1	1	0
14	1	1	1	0	0	1	1	1	1
15	1	1	1	1	d	d	d	d	d

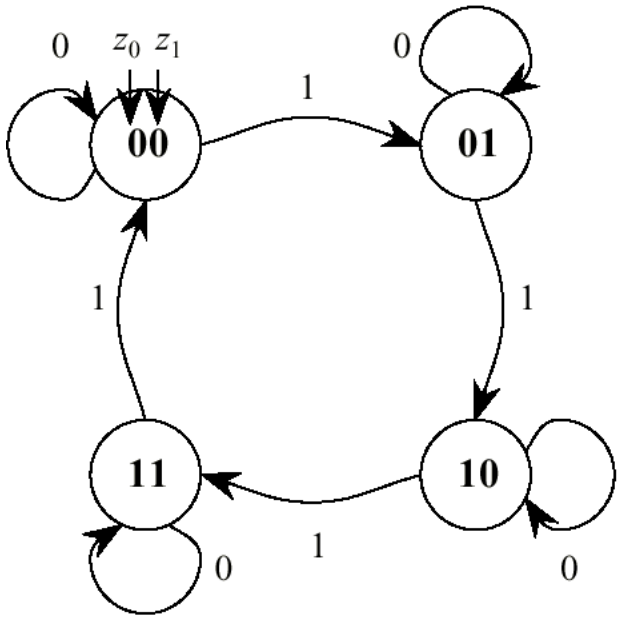
(b)

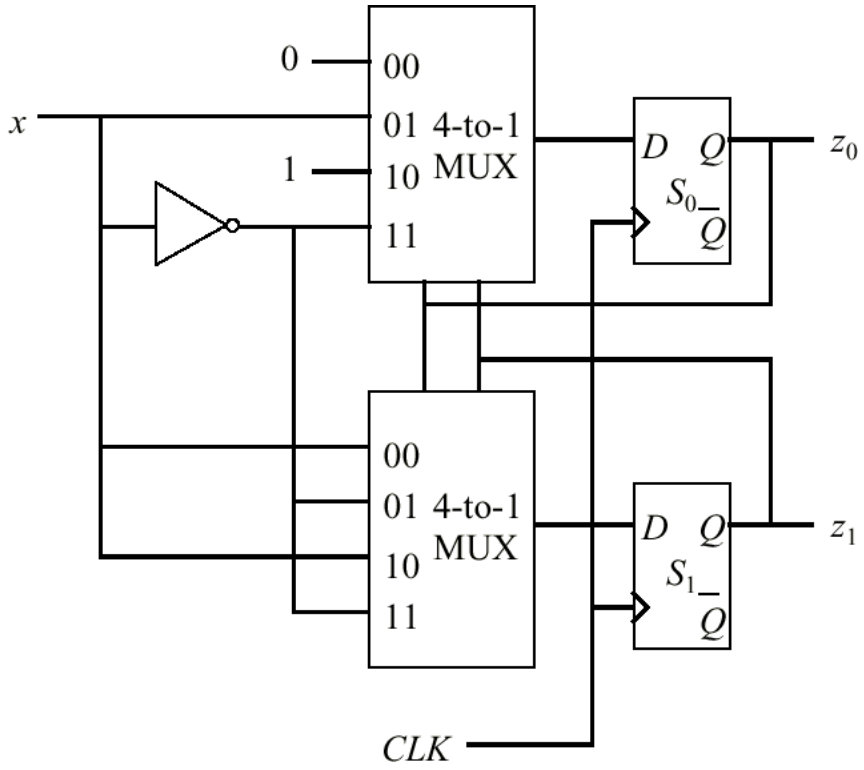


(c)

Moore Modulo-4 Counter

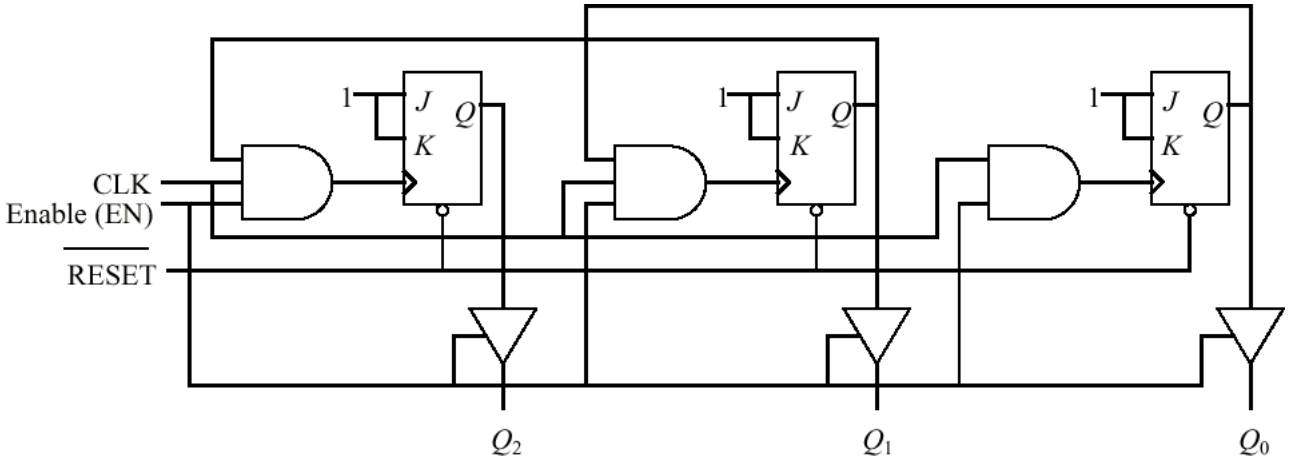
- **Mealy Model: Outputs are functions of Inputs and Present State.**
- **Previous FSM designs were Mealy Machines, in which next state was computed from present state and inputs.**
- **Moore Model: Outputs are functions of Present State only.**

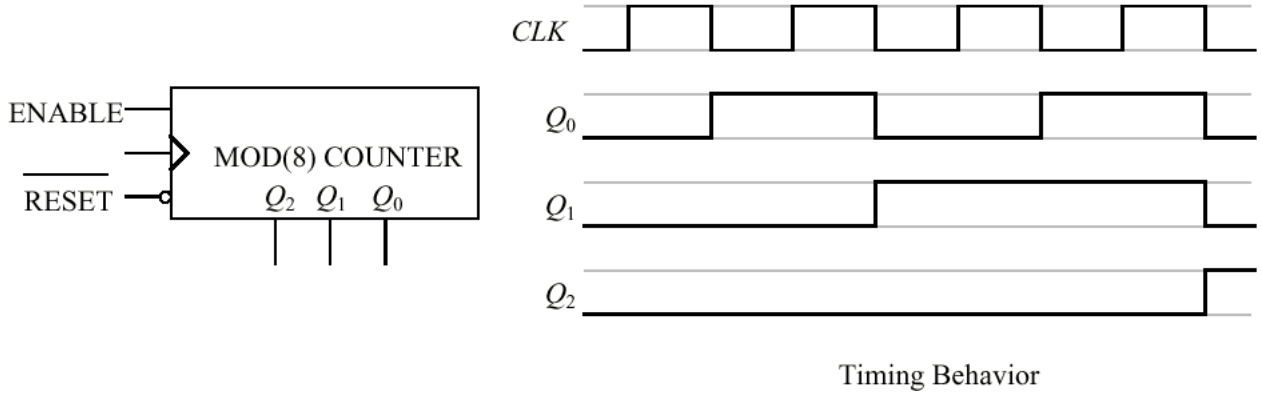




Modulo-8 Counter

- **Note the use of the T flip-flops, implemented as J-K's. They are used to toggle the input of the next flip-flop when its output is 1.**





Timing Behavior

Review of Objectives

We…….

- **Examined the processes involved in designing Finite State Machines or FSMs**
  - **Design carried out six steps, which are to**
    - **Draw a state transition diagram**
    - **Write the state transition table**
    - **Develop state assignments (SA's)**
    - **Modify the state transition table to reflect the chosen SA's**
    - **Draw up a truth table and using this to design the combinational logic unit of the FSM**
    - **Finally draw up the complete logic schematic of the FSM**
- **Reviewed function of Mealy & Moore FSM's**
- **Reviewed a modulo 8 counter using JK FF's**