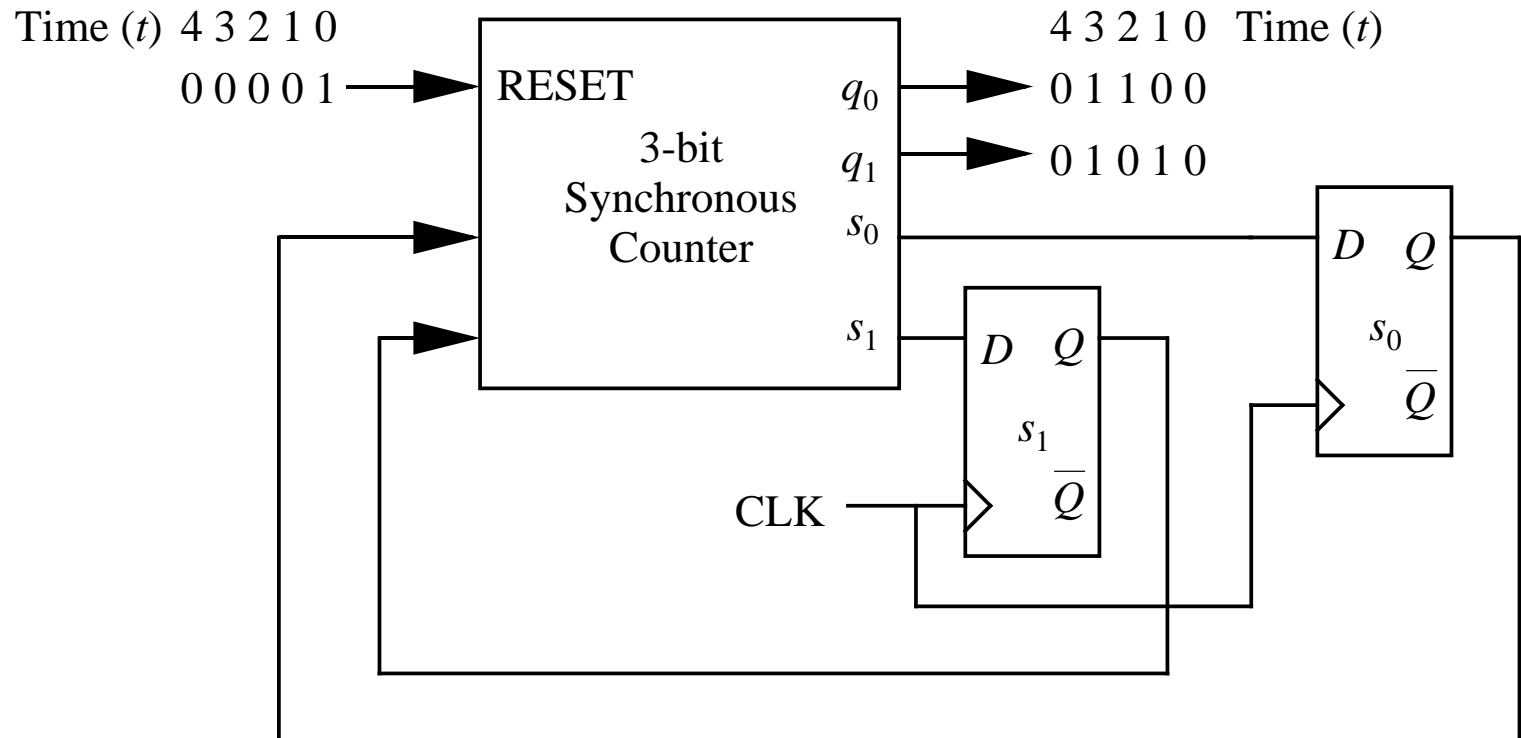
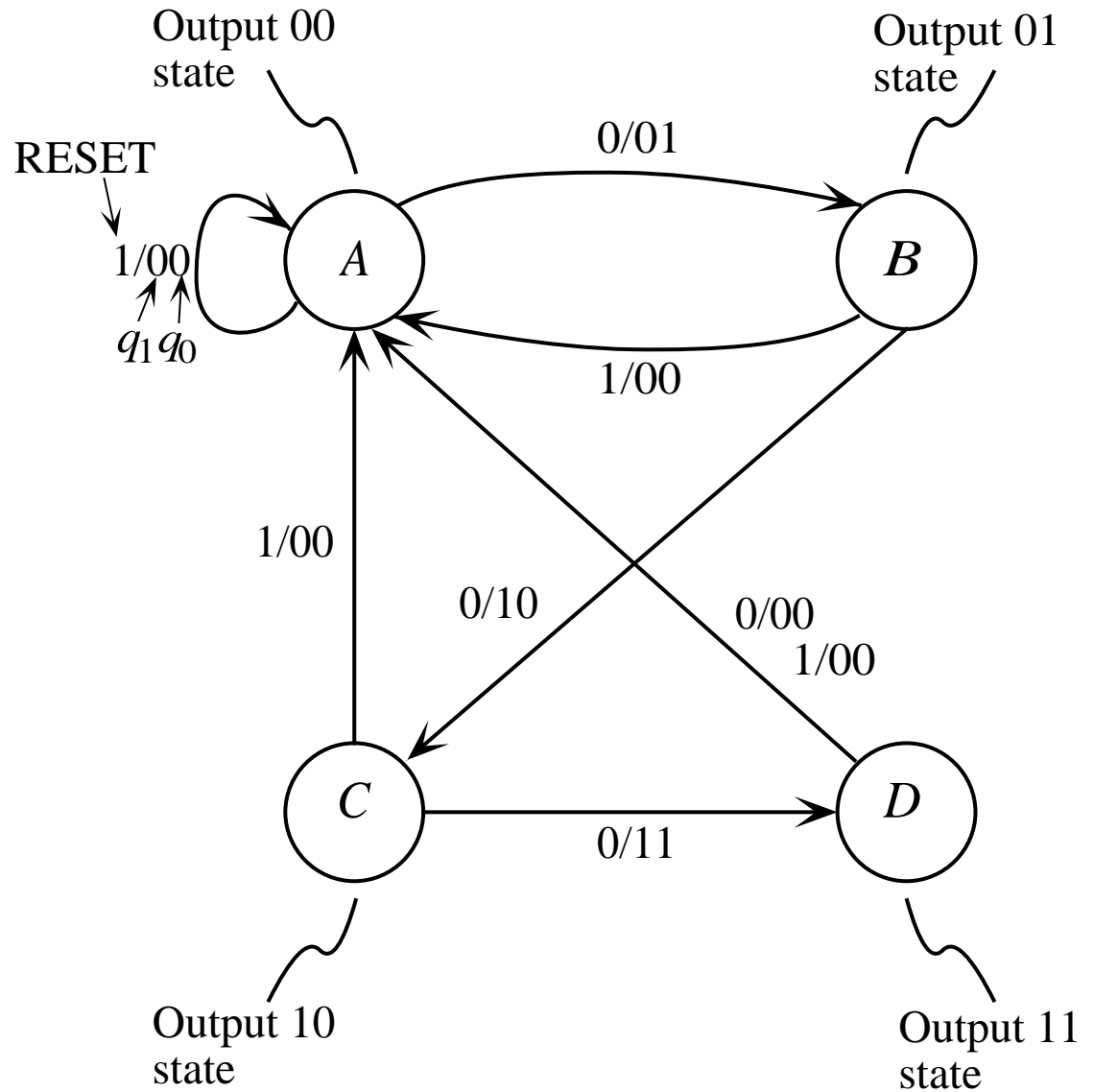


Example: Modulo-4 Counter

- 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	<i>RESET</i>	
	0	1
<i>A</i>	<i>B/01</i>	<i>A/00</i>
<i>B</i>	<i>C/10</i>	<i>A/00</i>
<i>C</i>	<i>D/11</i>	<i>A/00</i>
<i>D</i>	<i>A/00</i>	<i>A/00</i>

Next state Output

State Assignment for Mod-4 Counter

Present state (S_t) \ Input	<i>RESET</i>	
	0	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

<i>RESET</i> <i>r(t)</i>	<i>s</i> ₁ (<i>t</i>)	<i>s</i> ₀ (<i>t</i>)	<i>s</i> ₁ <i>s</i> ₀ (<i>t</i> +1)	<i>q</i> ₁ <i>q</i> ₀ (<i>t</i> +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

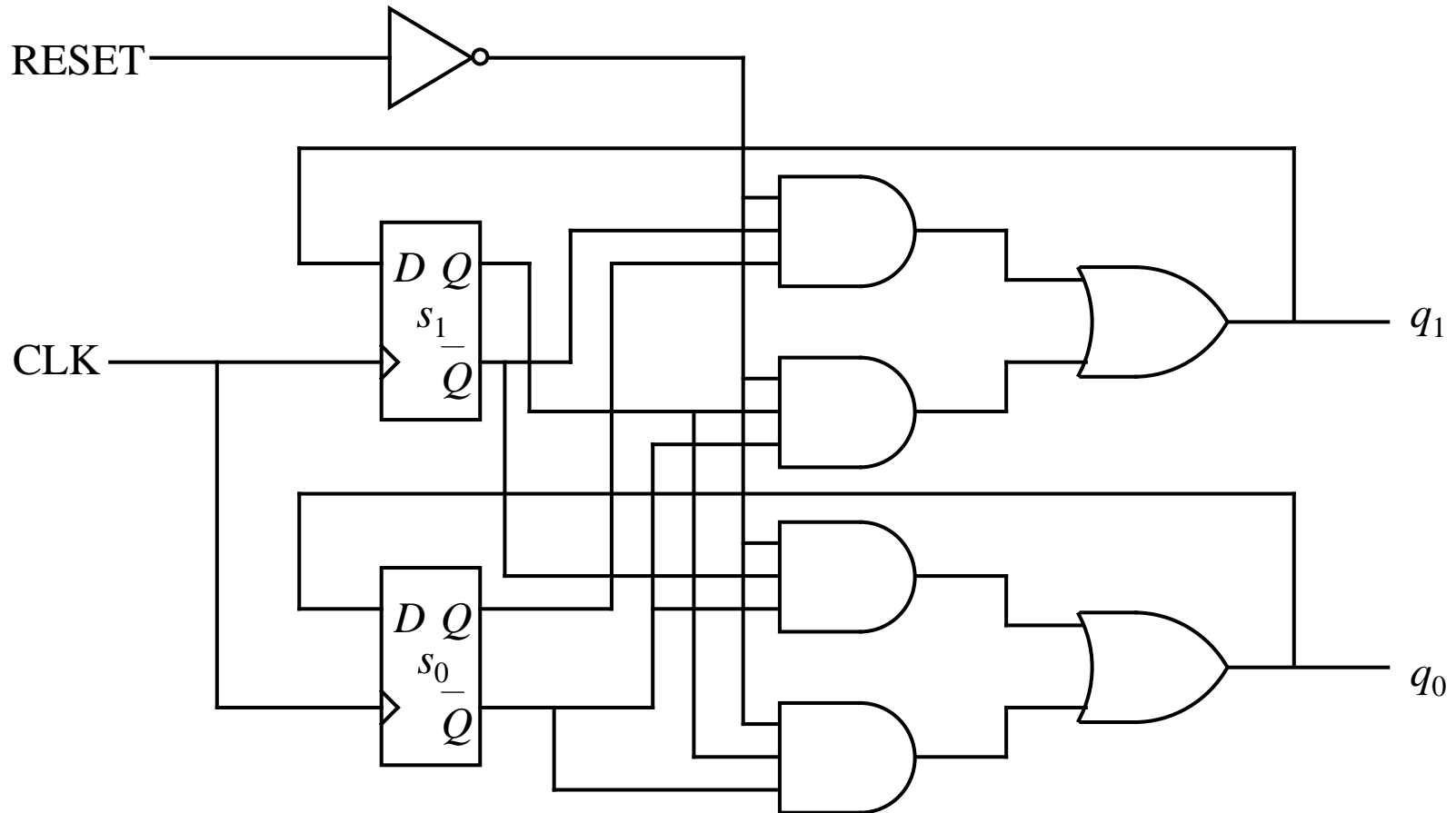
$$s_0(t+1) = \overline{r(t)}\overline{s_1(t)}\overline{s_0(t)} + \overline{r(t)}s_1(t)\overline{s_0(t)}$$

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

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

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

Logic Design for Mod-4 Counter

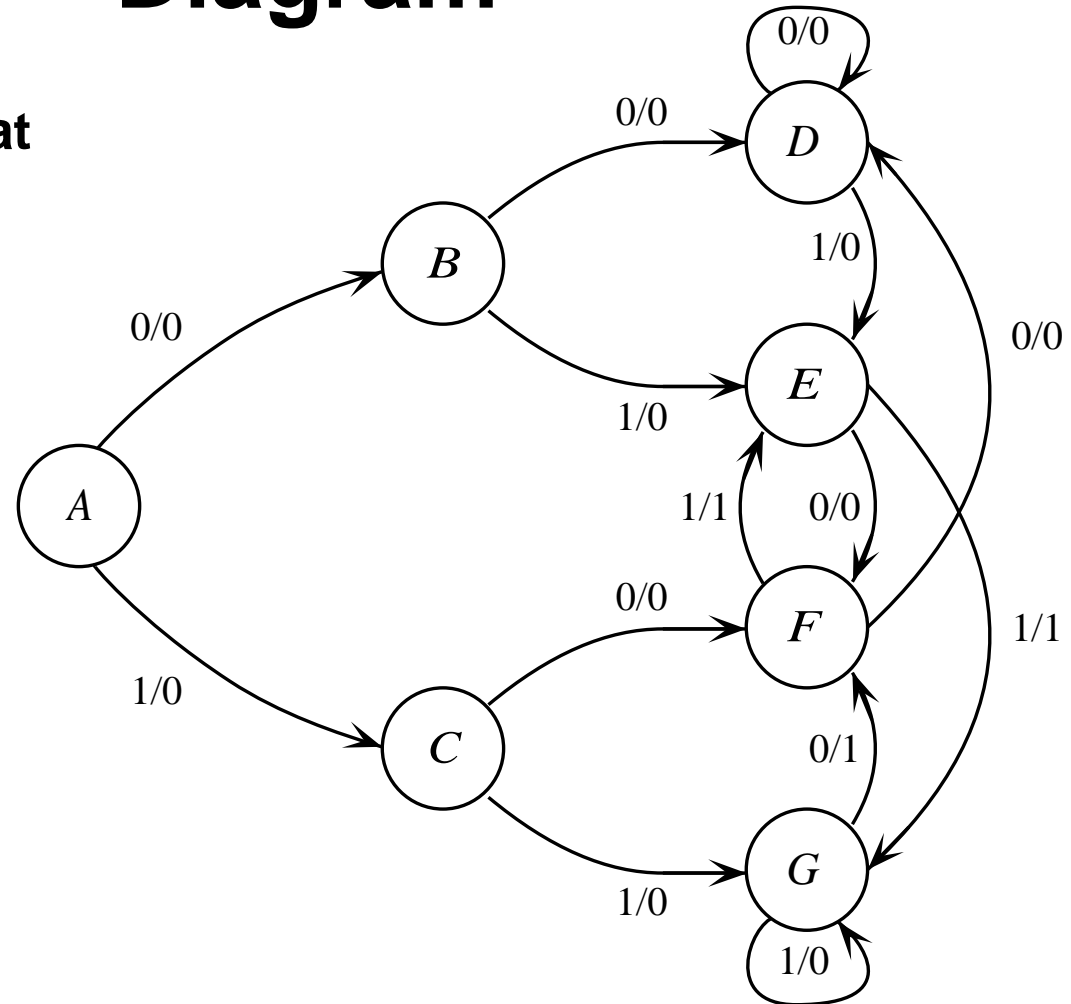


Example: A Sequence Detector

- **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	<i>X</i>	
	0	1
<i>A</i>	<i>B/0</i>	<i>C/0</i>
<i>B</i>	<i>D/0</i>	<i>E/0</i>
<i>C</i>	<i>F/0</i>	<i>G/0</i>
<i>D</i>	<i>D/0</i>	<i>E/0</i>
<i>E</i>	<i>F/0</i>	<i>G/1</i>
<i>F</i>	<i>D/0</i>	<i>E/1</i>
<i>G</i>	<i>F/1</i>	<i>G/0</i>

Sequence Detector State Assignment

Present state \ Input	X	
	0	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

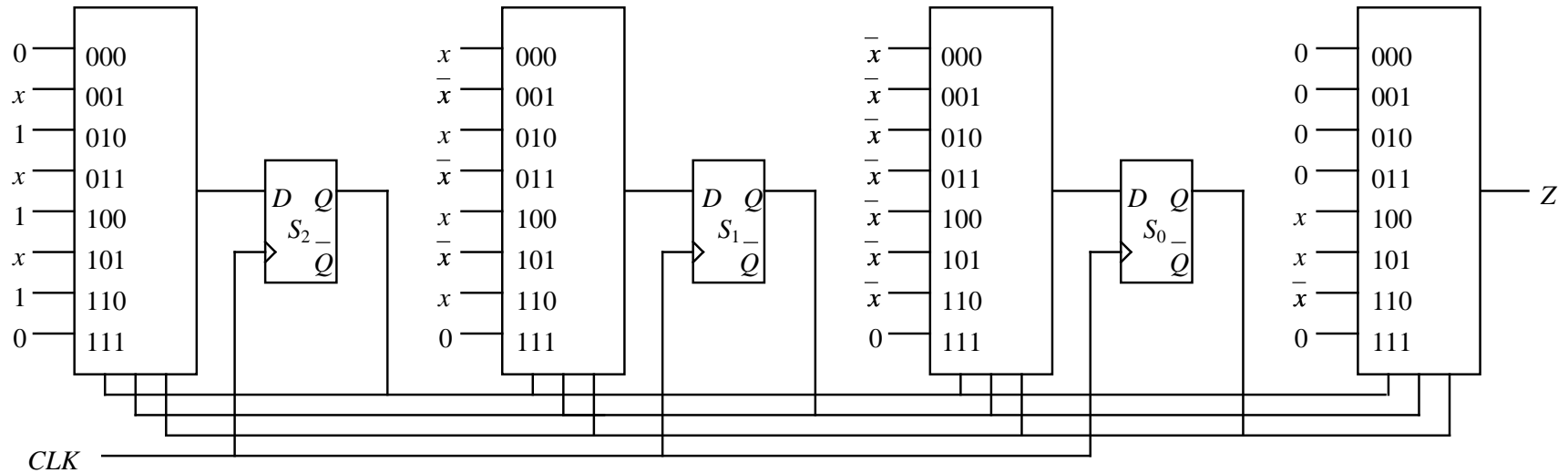
(a)

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)

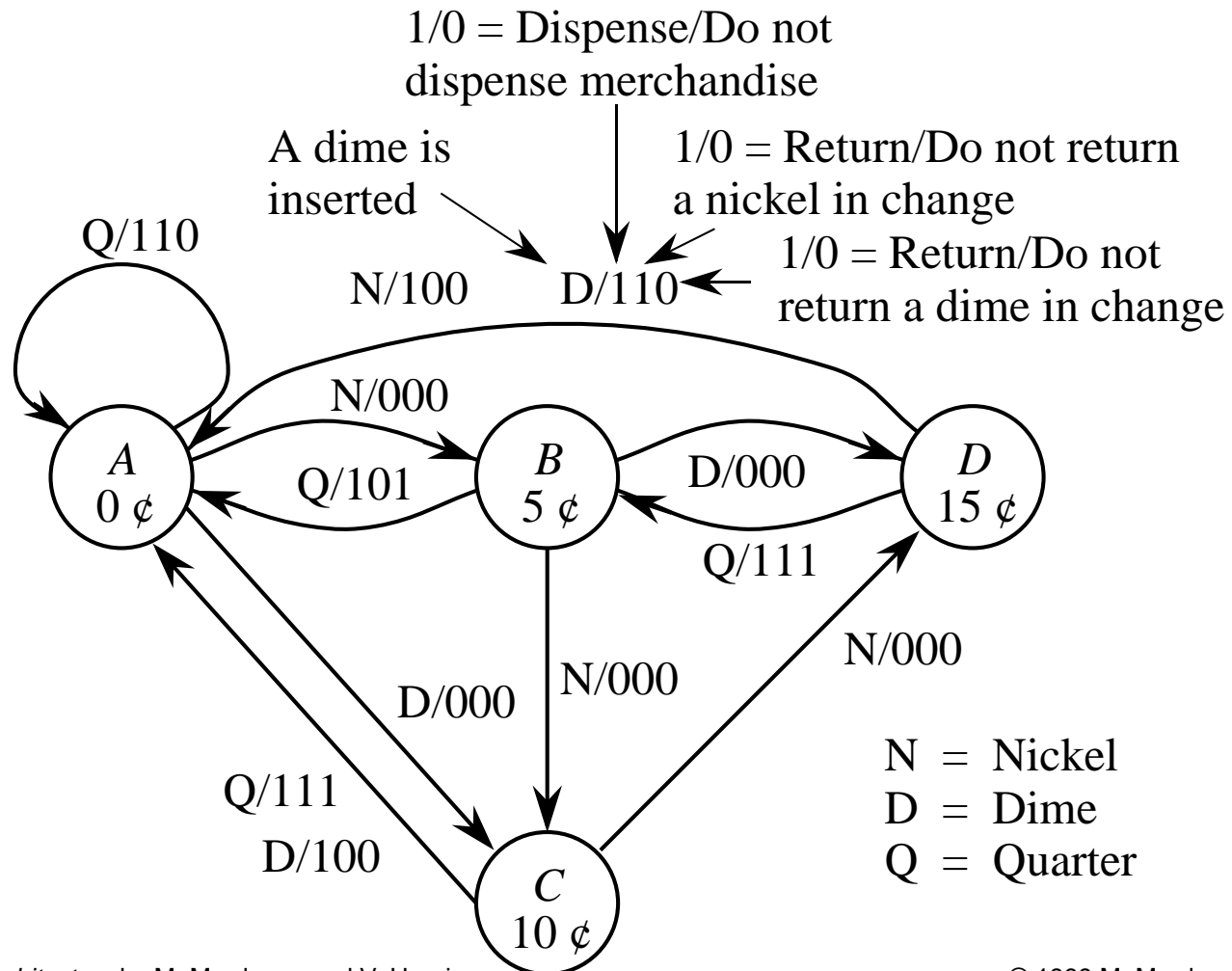
Sequence Detector Logic Diagram



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

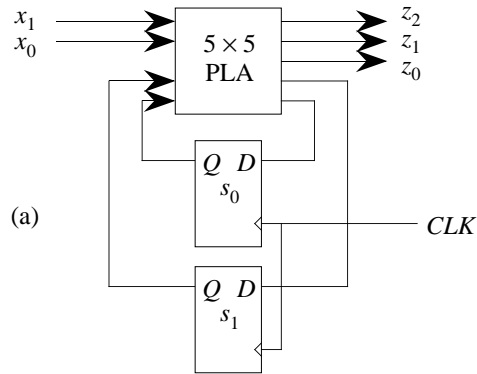
Input P.S.	N 00	D 01	Q 10
A	B/000	C/000	A/110
B	C/000	D/000	A/101
C	D/000	A/100	A/111
D	A/100	A/110	B/111

(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



Base 10 equivalent	Present state		Coin		Next state				
	s_1	s_0	x_1	x_0	s_1	s_0	z_2	z_1	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)

