

CMSC 313

COMPUTER ORGANIZATION

&

ASSEMBLY LANGUAGE

PROGRAMMING

Lecture 23, Fall 2014

TOPICS TODAY

- **Finite State Machines**
- **Example: Mod-4 Counter**
- **Example: Vending Machine**

FINITE STATE MACHINES

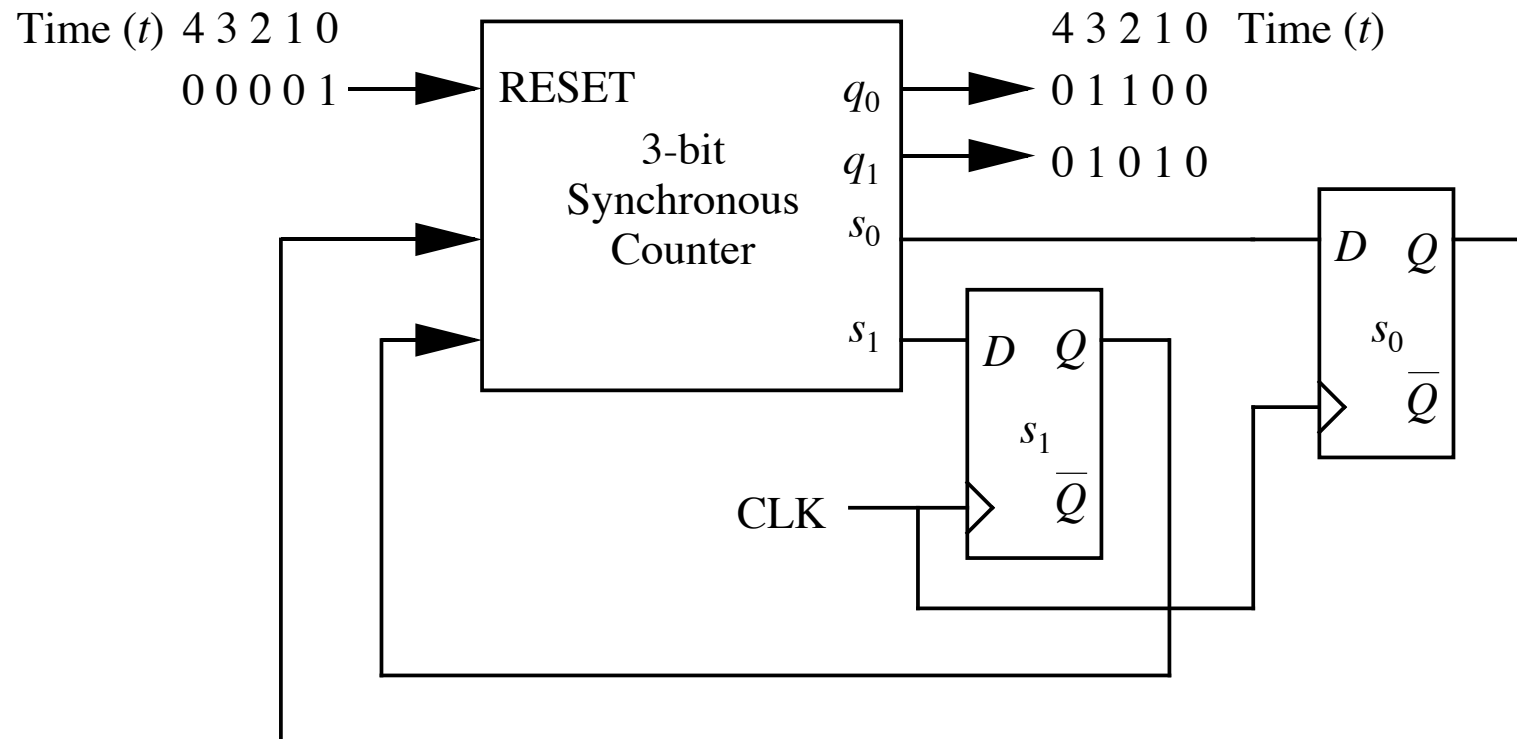


EXAMPLE: MOD 4 COUNTER

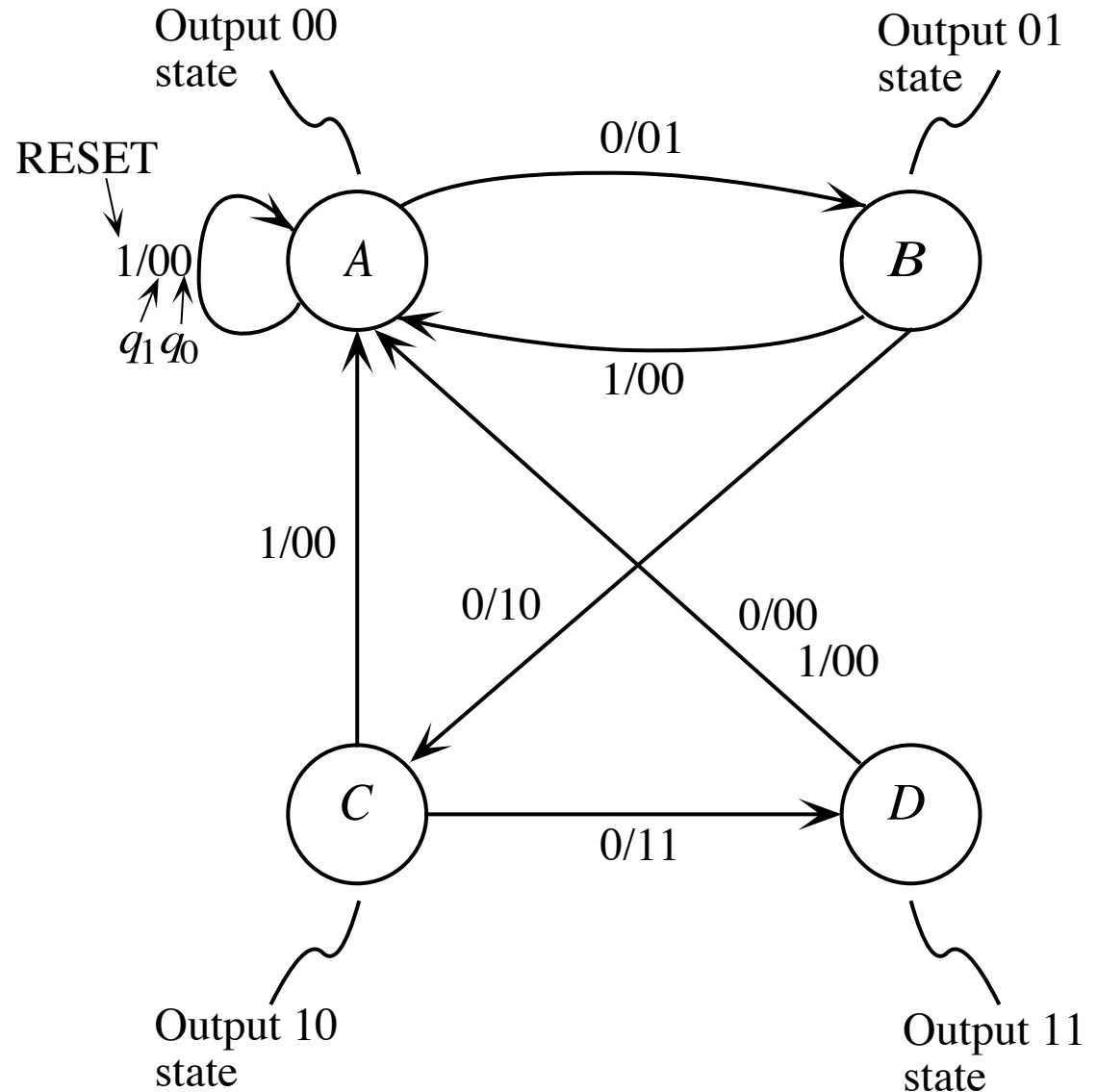


Example: Modulo-4 Counter

- 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

Input Present state	<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
<i>A</i> :00	01/01	00/00
<i>B</i> :01	10/10	00/00
<i>C</i> :10	11/11	00/00
<i>D</i> :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

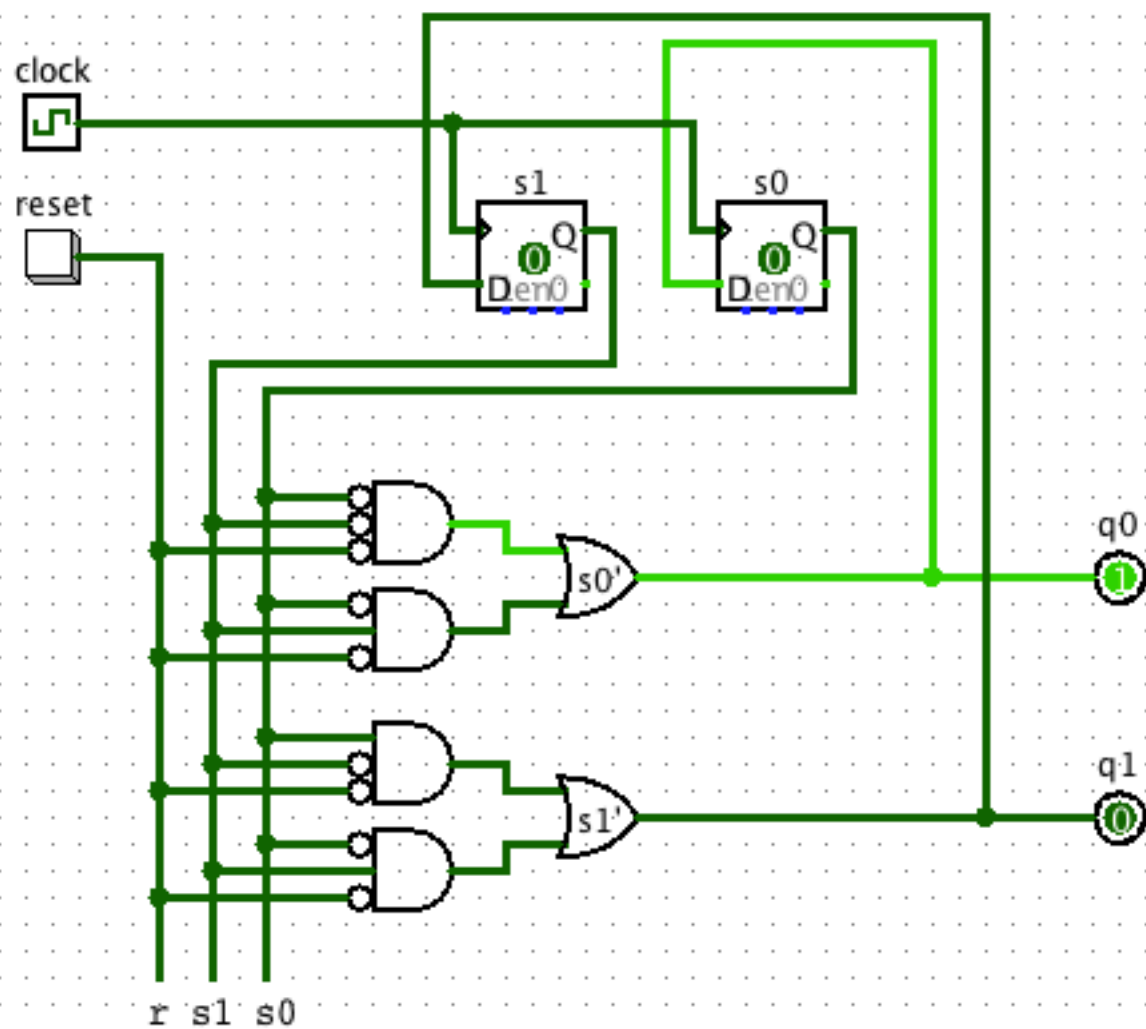
$$s_0(t+1) = \overline{r(t)}\overline{s_1(t)}\overline{s_0(t)} + \overline{r(t)}s_1(t)\overline{s_0(t)}$$

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

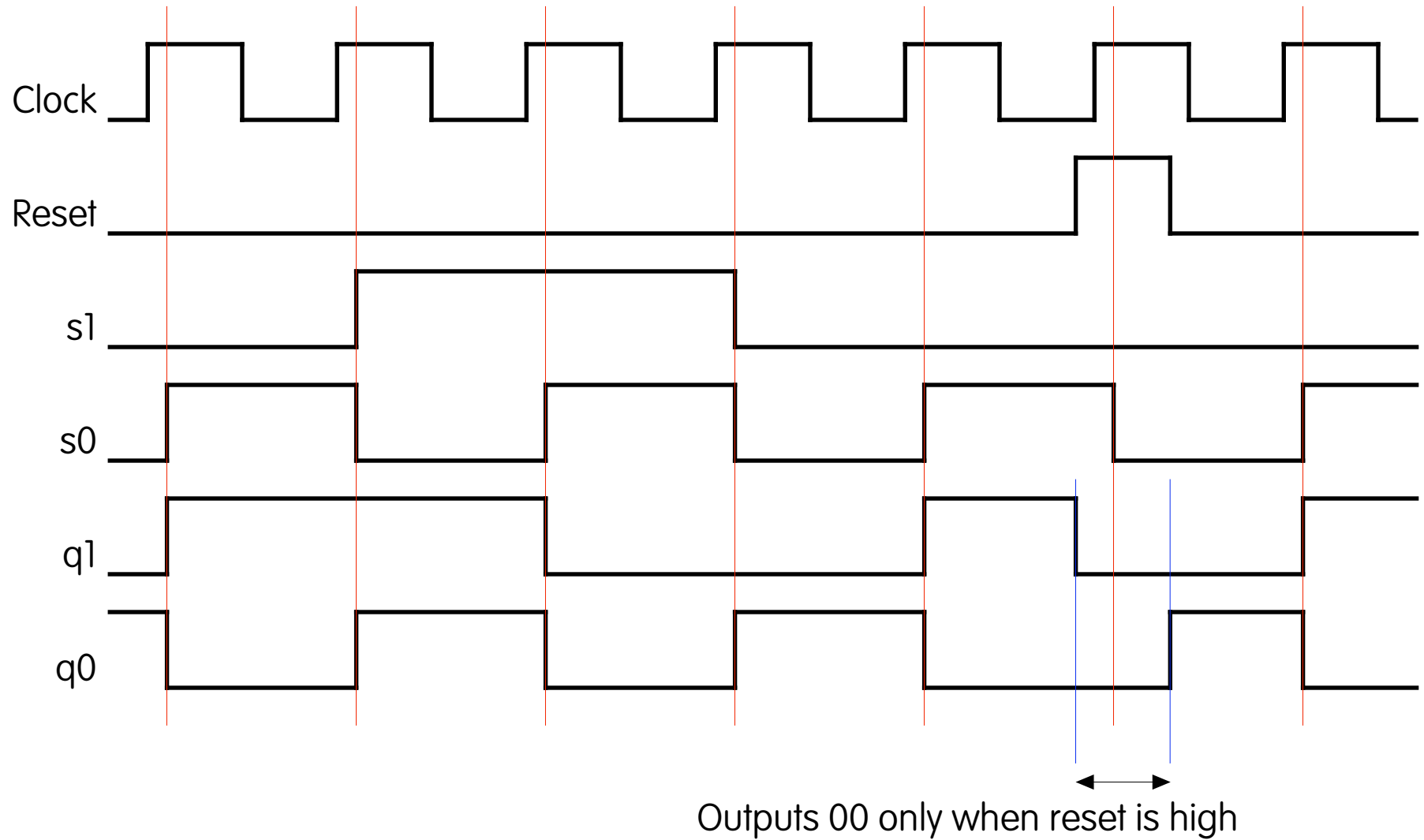
$$q_0(t+1) = \overline{r(t)}\overline{s_1(t)}\overline{s_0(t)} + \overline{r(t)}s_1(t)\overline{s_0(t)}$$

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

Mod 4 Counter

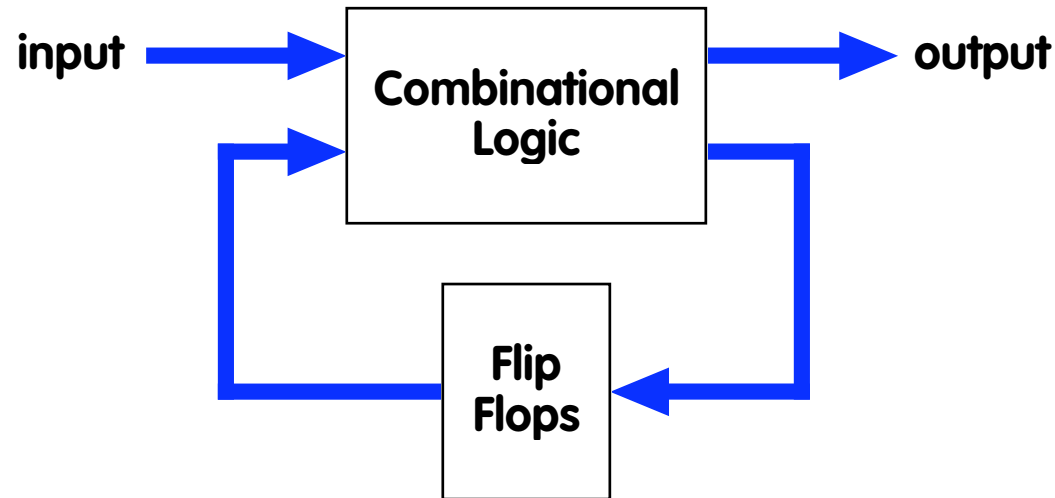


Mod 4 Counter Timing

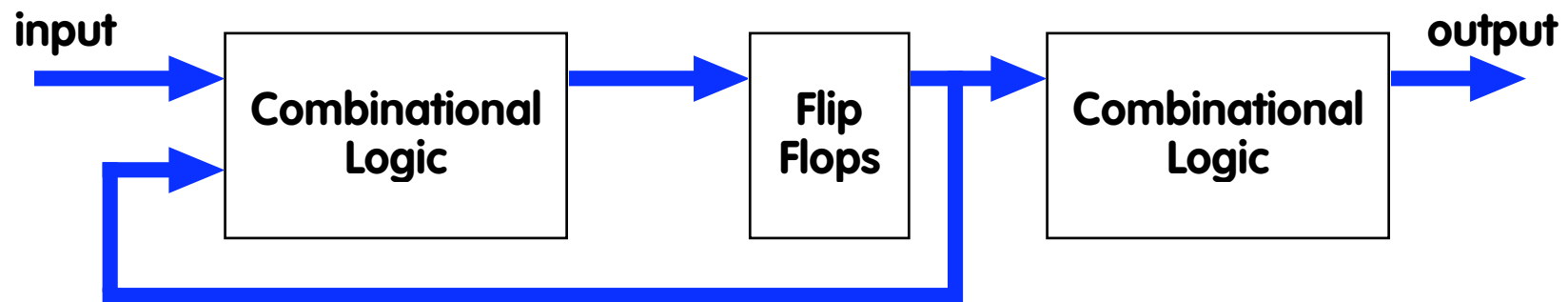


Mealy vs Moore Finite State Machines

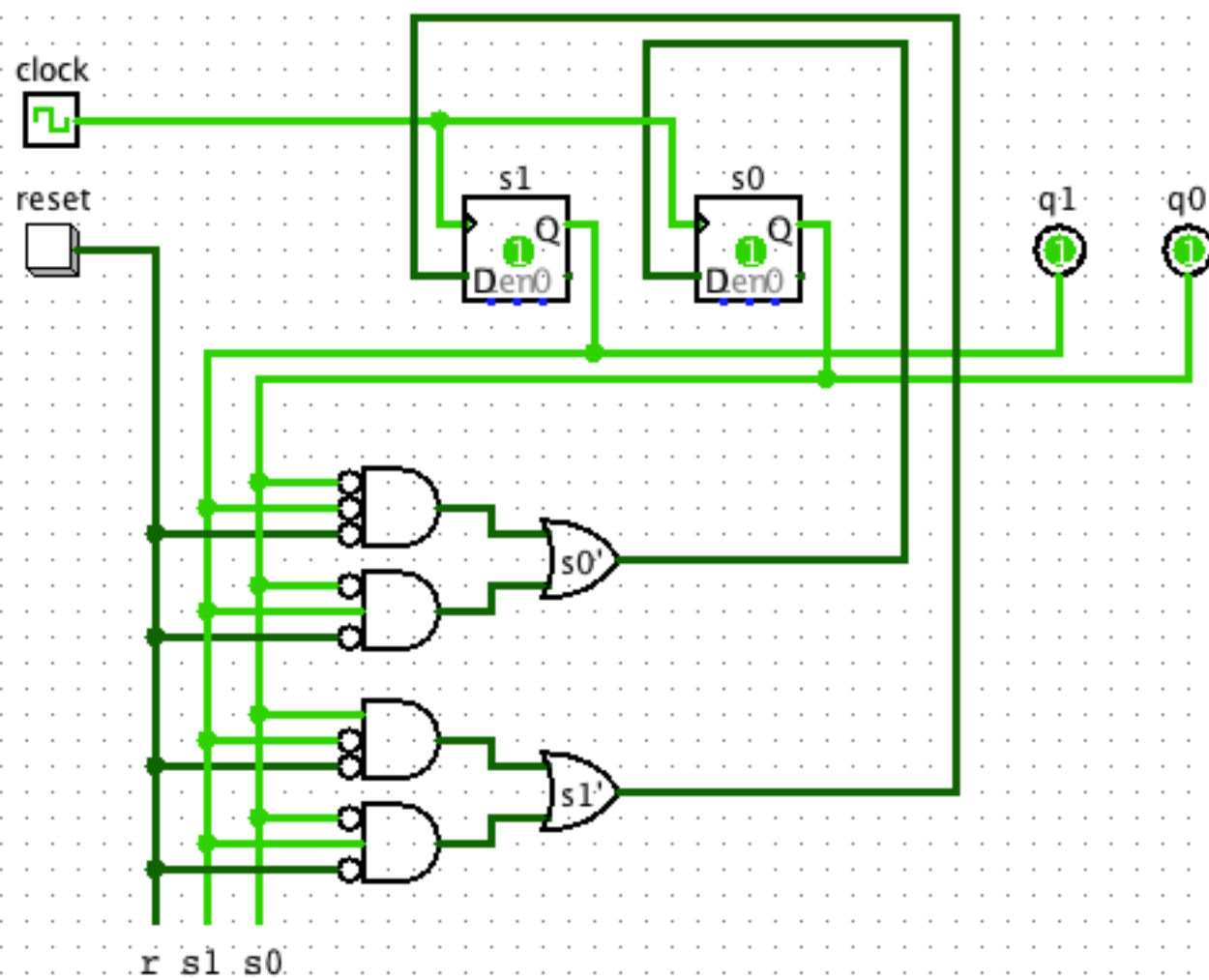
- **Mealy: output depends on input and state bits**



- **Moore: output depends only on state bits**



Mod 4 Counter (Moore)



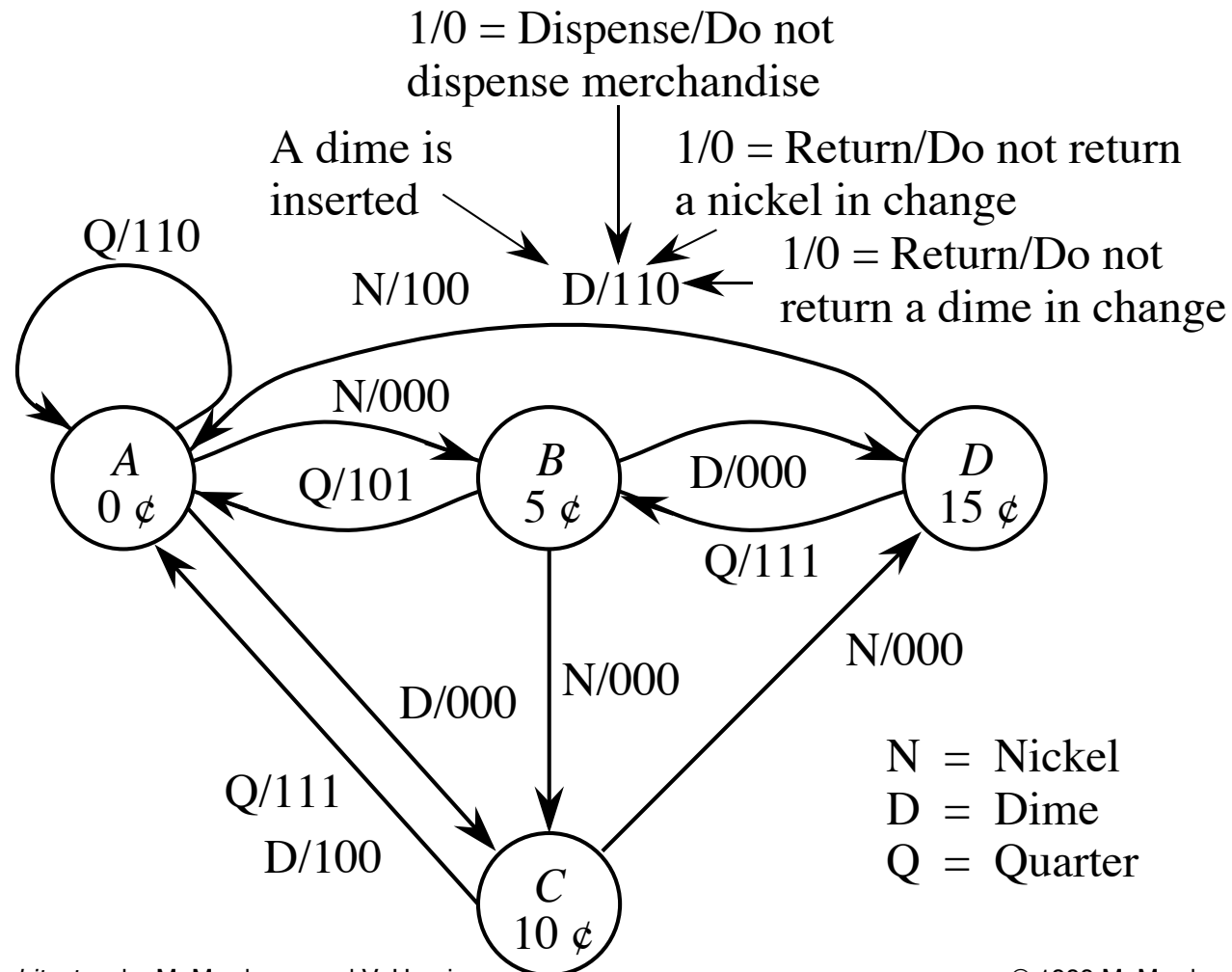
EXAMPLE: VENDING MACHINE



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

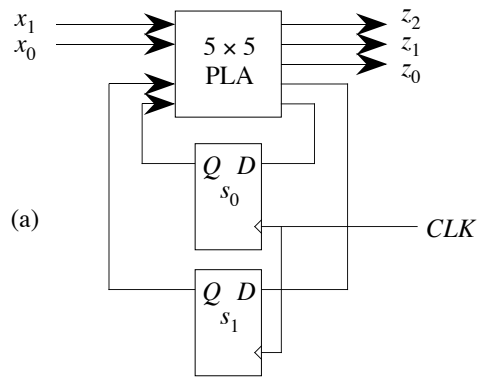
Input P.S.	N 00	D 01	Q 10
<i>A</i>	<i>B</i> /000	<i>C</i> /000	<i>A</i> /110
<i>B</i>	<i>C</i> /000	<i>D</i> /000	<i>A</i> /101
<i>C</i>	<i>D</i> /000	<i>A</i> /100	<i>A</i> /111
<i>D</i>	<i>A</i> /100	<i>A</i> /110	<i>B</i> /111

(a)

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

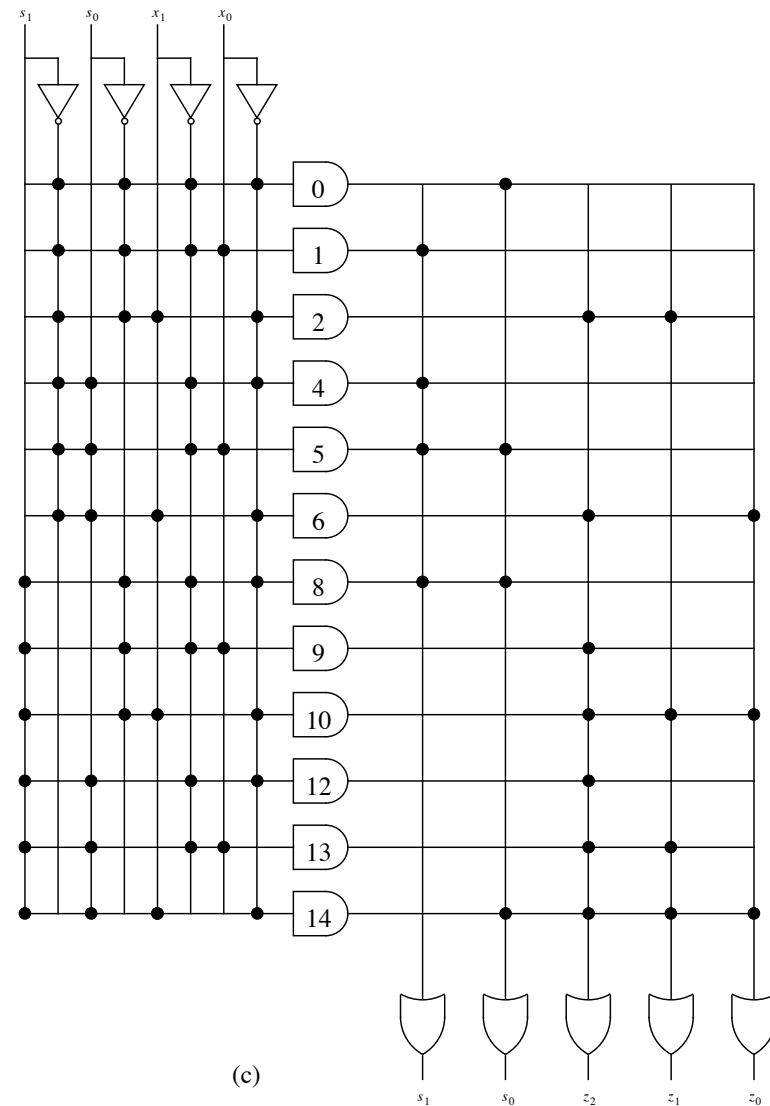
(b)

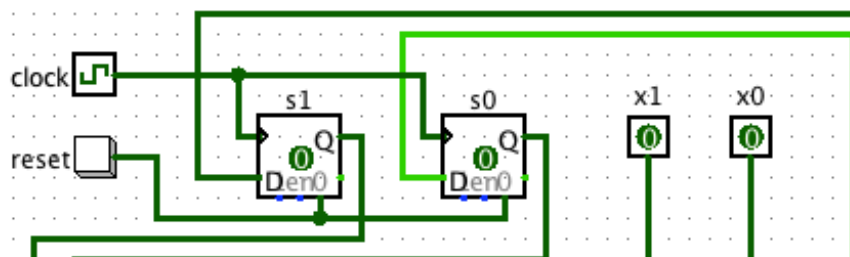
PLA Vending Machine Controller



(b)

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	1	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 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 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 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 d	d d





States (s1 s0):
 00 = A (0cts)
 01 = B (5cts)
 10 = C (10cts)
 11 = D (15cts)

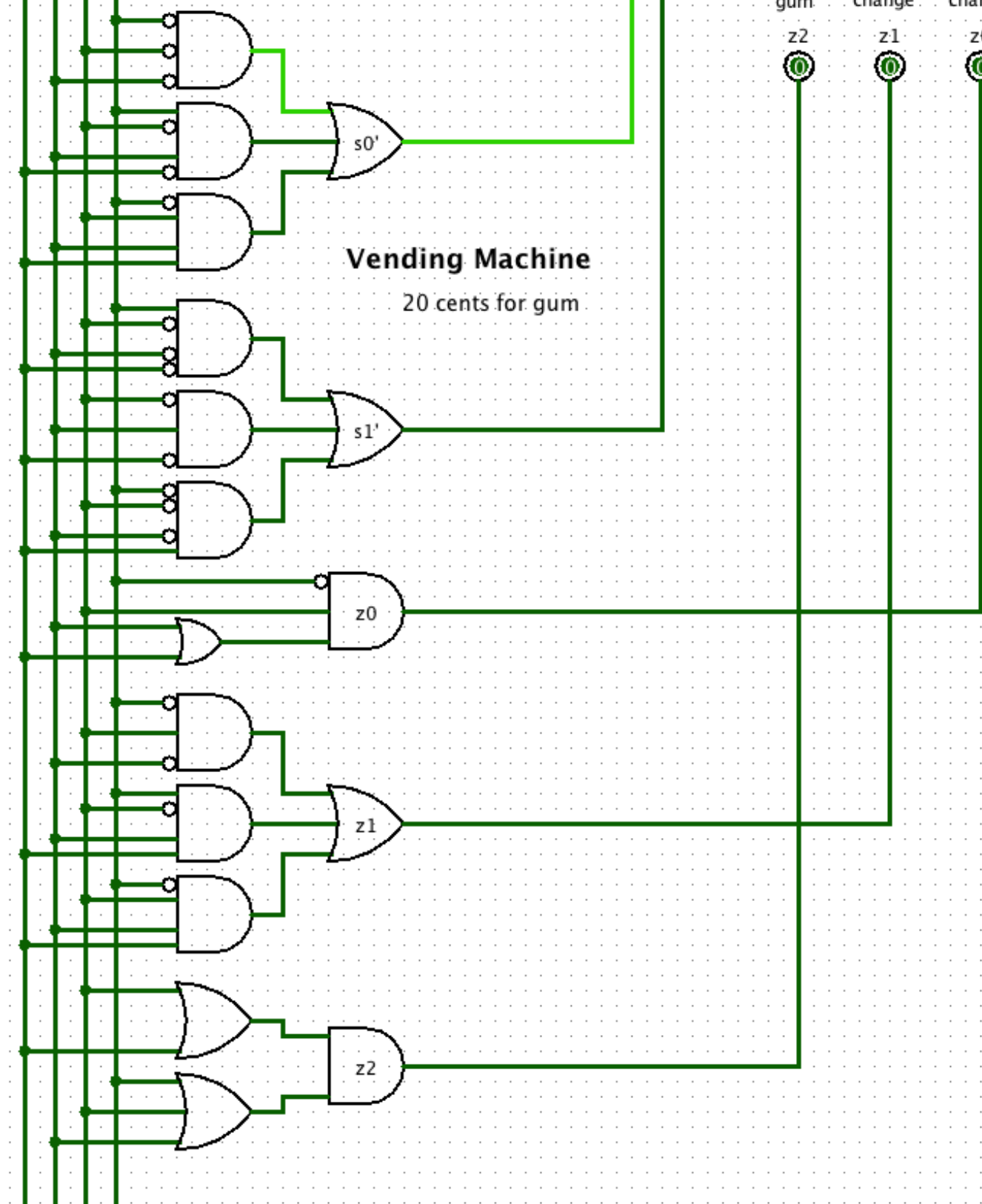
Input (x1 x0):
 00 = nickel
 01 = dime
 10 = quarter
 11 = xxxx

dispense gum nickel change dime change

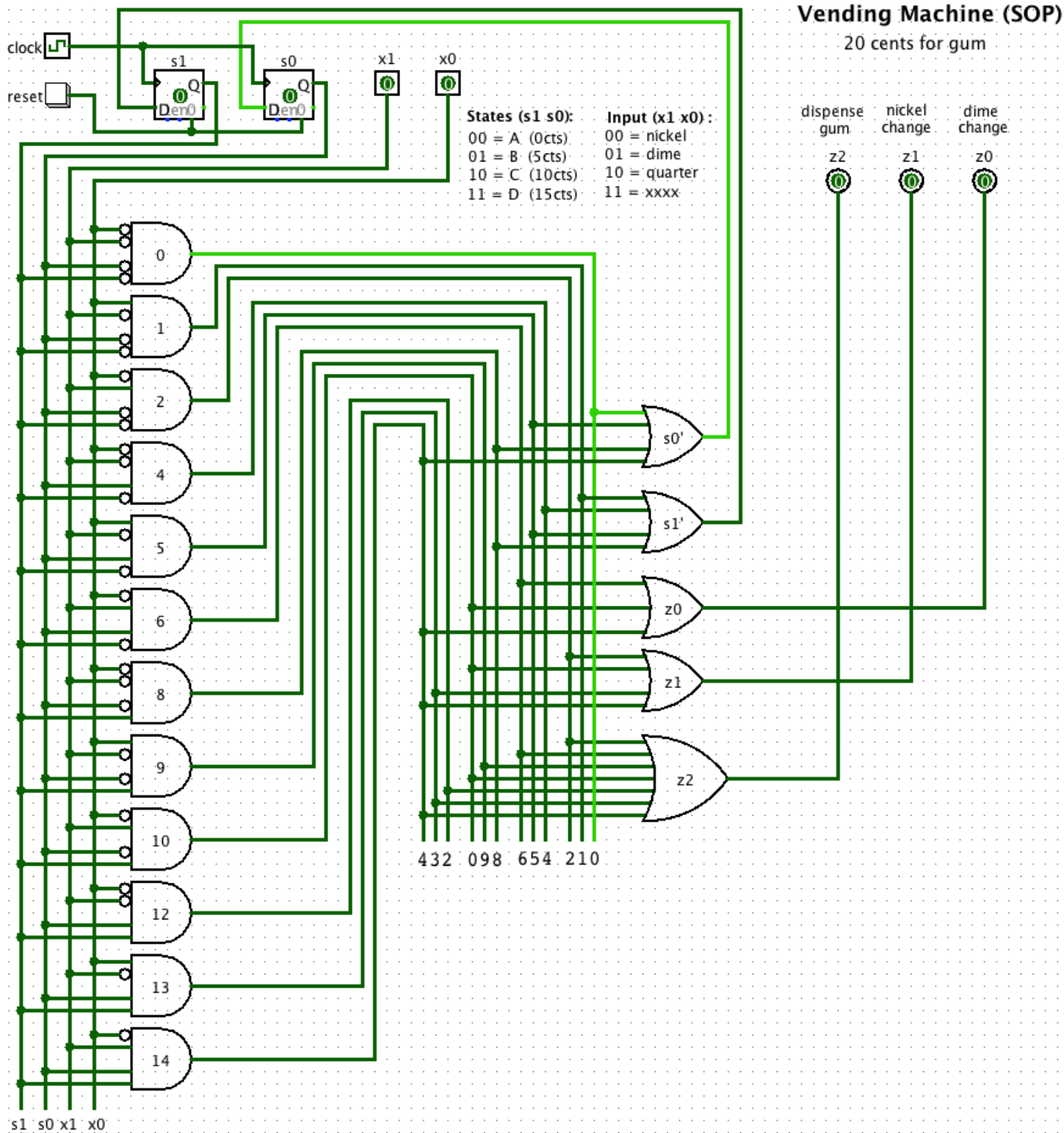
z2 z1 z0

Vending Machine

20 cents for gum



s1 s0 x1 x0



NEXT TIME

- Finite State Machine Simplification

