Name: Nicholas Matvienko Last 4 ID: 9670

HW 4 **Due: November 3, 2025 at 11:59 PM in Santa Cruz**

CSE 100
Fall 2025

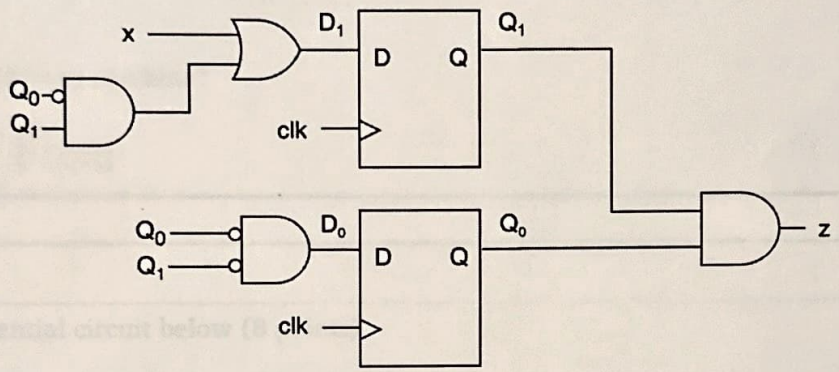
Reading: Sections 4.7 and 5.1-5.7.

Instructions: Complete the problems below **in your own handwriting**. Box your answers where necessary. The grader will award 1 point for your name and the last four digits of your student ID, and 1 point for neatness.

Problem 1

For the sequential circuit below (8 points):

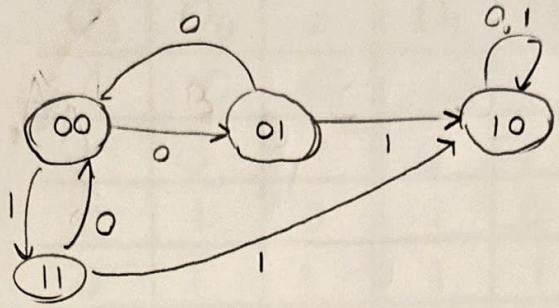
- Obtain its state transition table
- Draw its state diagram
- Identify whether the circuit implements a Mealy or Moore machine



State transition table:

Current State			Next State		Output
Q_1	Q_0	x	D_1	D_0	z
0	0	0	0	1	0
0	0	1	1	1	0
0	1	0	0	0	0
0	1	1	1	0	0
1	0	0	1	0	0
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	1

State diagram:



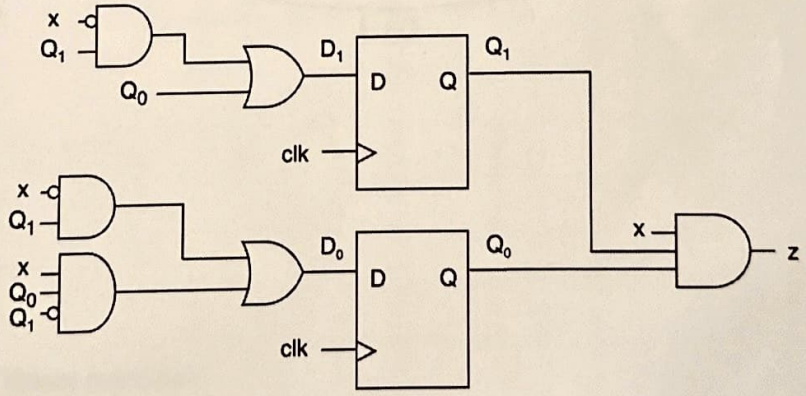
Mealy or Moore machine?

Moore

Problem 2

For the sequential circuit below (8 points):

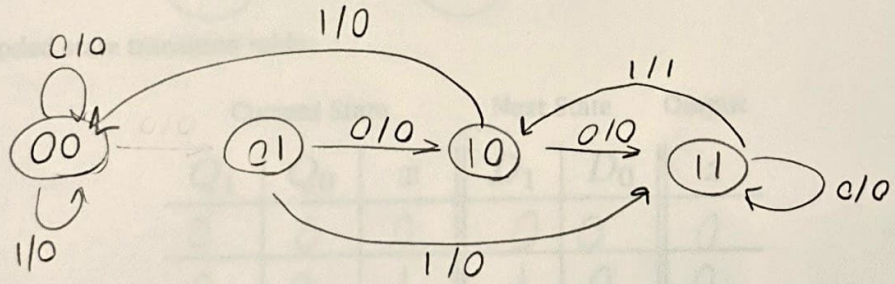
- Obtain its state transition table
- Draw its state diagram
- Identify whether the circuit implements a Mealy or Moore machine



State transition table:

Current State			Next State		Output
Q_1	Q_0	x	D_1	D_0	z
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	1	0	0
0	1	1	1	1	0
1	0	0	1	1	0
1	0	1	0	0	0
1	1	0	1	1	0
1	1	1	1	0	1

State diagram:



Mealy or Moore machine?

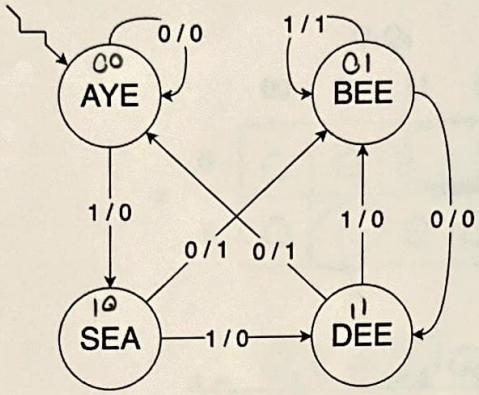
Mealy

Problem 3

The behavior of a synchronous sequential circuit with one input (x) and one output (z) is given by the state diagram below. (Remember: in the diagram the arcs are labeled as 0/1, where 0 is the input value of x, and 1 is the input value of z in this example)

Implement this sequential circuit using boolean logic gates and *two* D Flip-Flops. To receive full credit complete the following (8 points):

- Use the minimum length encoding provided on the right.
- Give the state transition and output table.
- Obtain minimal SOP expressions for your next state and output logic using K-maps.
- Draw the final sequential circuit.

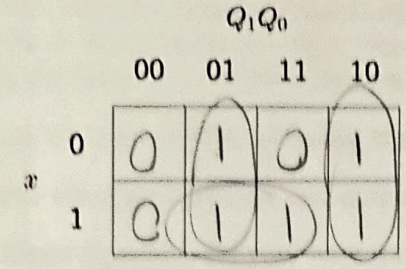


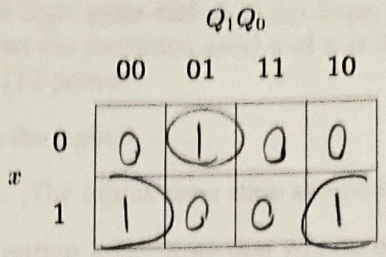
	Q_1	Q_0
AYE	0	0
BEE	0	1
SEA	1	0
DEE	1	1

Encoded state transition table:

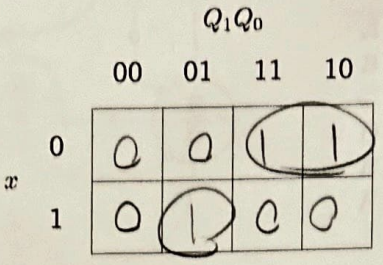
Current State			Next State		Output
Q_1	Q_0	x	D_1	D_0	z
0	0	0	0	0	0
0	0	1	1	0	0
0	1	0	1	1	0
0	1	1	0	1	1
1	0	0	0	1	1
1	0	1	1	1	0
1	1	0	0	0	1
1	1	1	0	1	0

Minimal SOP equations and K-maps:



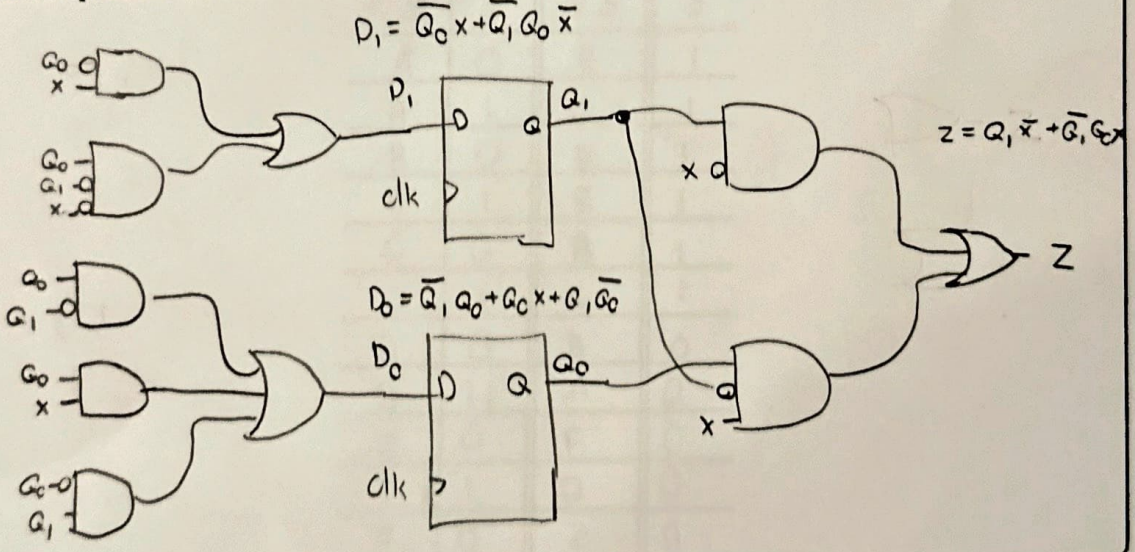


$D_0 = (\overline{Q_1}Q_0) + (Q_0x) + (Q_1\overline{Q_0})$ $D_1 = (\overline{Q_0}x) + \overline{Q_1}Q_0\overline{x}$



$z = Q_1\overline{x} + \overline{Q_1}Q_0x$

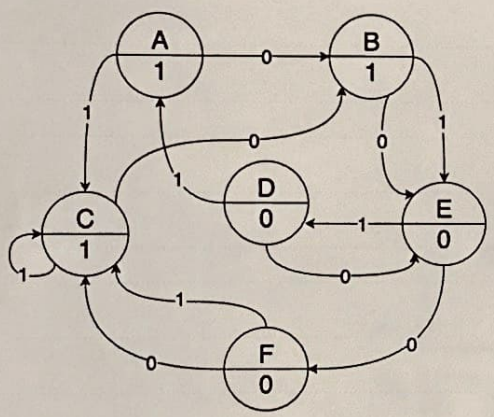
Final sequential circuit:



Problem 4

Implement the state diagram shown below with logic gates and *six* D flip-flops. Its initial state should be A. Use **x** as the the input (the value on the transition arcs) and **z** as the output (the value in the bottom of the circle in each state) (12 points).

- Use the one-hot state encoding shown on the right.
- Give the state transition and output table. (The initial/reset state should be A)
- Obtain equations for your next state and output logic. Note that you do not need to use K-maps to reduce your state logic; this is one of the benefits of one-hot encoding.
- Draw the final sequential circuit.



	Q_5	Q_4	Q_3	Q_2	Q_1	Q_0
A	0	0	0	0	0	1
B	0	0	0	0	1	0
C	0	0	0	1	0	0
D	0	0	1	0	0	0
E	0	1	0	0	0	0
F	1	0	0	0	0	0

State transition table (use letters to represent current and next state):

Current State		Next State	Output
C.S.	x	*N.S.*	z
A	0	B	1
A	1	C	1
B	0	E	1
B	1	E	1
C	0	B	1
C	1	C	1
D	0	E	0
D	1	A	0
E	0	F	0
E	1	D	0
F	0	C	0
F	1	C	0

State and output logic equations (use state bits (e.g. Q_0) and not the state name (e.g. A)):

$$D_0 = Q_3 x \qquad z = Q_0 + Q_1 + Q_2$$

$$D_1 = Q_0 \bar{x} + Q_2 \bar{x}$$

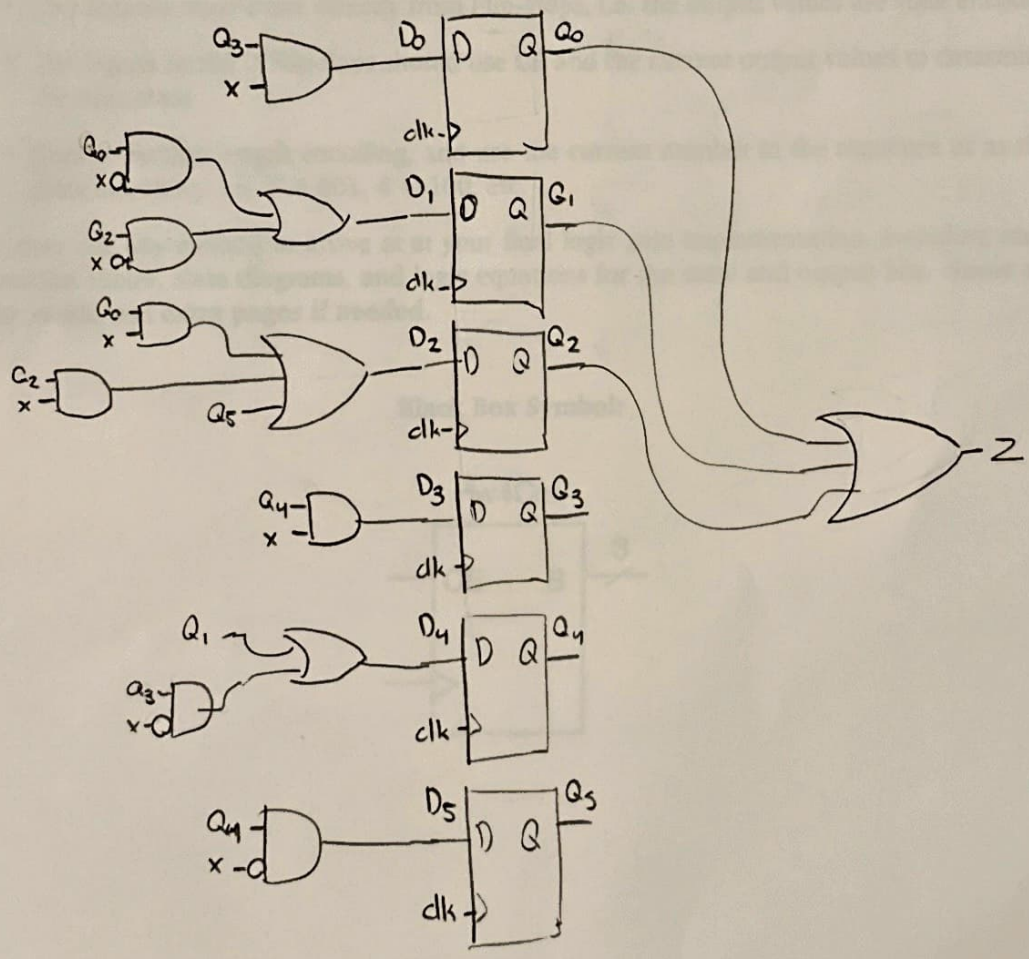
$$D_2 = Q_0 x + Q_2 x + Q_5$$

$$D_3 = Q_4 x$$

$$D_4 = Q_1 + Q_3 \bar{x}$$

$$D_5 = Q_4 \bar{x}$$

Final sequential circuit:



Problem 5

Implement a 3-bit counter, **Hw4Cnt**, with logic gates and 3 D Flip-Flops such that it (12 points):

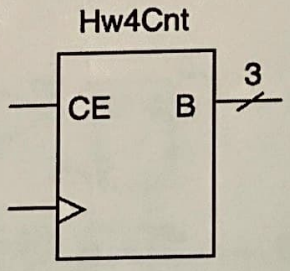
- Counts in the sequence: 0, 1, 3, 2, 6, 7, 5, 4
- Loops, i.e. 0, 1, 3, 2, 6, 7, 5, 4, 0, 1, 3, 2, 6, 7, 5, 4, ...
- Advances to the next number in the sequence when CE is high, and never otherwise

Your implementation must satisfy the following requirements:

- Only uses logic gates and 3 D Flip-Flops
- The outputs, B_2, B_1, and B_0 represents the current number in the sequence
- The outputs *must* come directly from Flip-Flops, i.e. the output values are state encoded
- The inputs to the D Flip-flops should use **CE** and the current output values to determine the next state
- Uses minimum-length encoding, and use the current number in the sequence as as the state encoding. i.e. 1 = 001, 4 = 100, etc.

You may use any method to arrive at at your final logic gate implementation, including state transition tables, state diagrams, and logic equations for the state and output bits. **Show all your work, use extra pages if needed.**

Black Box Symbol:



Q_2	Q_1	Q_0	CE	D_2	D_1	D_0	
0	0	0	0	0	0	0	
0	0	0	1	0	0	1	←
0	0	1	0	0	0	1	
0	0	1	1	0	1	1	←
0	1	1	0	0	1	1	
0	1	1	1	0	1	0	←
0	1	0	0	0	1	0	
0	1	0	1	1	1	0	←
1	1	0	0	1	1	0	
1	1	0	1	1	1	1	←
1	1	1	0	1	1	1	
1	1	1	1	1	0	1	←
1	0	1	0	1	0	1	
1	0	1	1	1	0	0	←
1	0	0	0	1	0	0	
1	0	0	1	1	0	0	←

N_2 ↳ 010, 110, 111, 101

N_1 ↳ 001, 011, 010, 110

N_0 ↳ 000, 001, 110, 111

$W_2 = Q_2 Q_0 + Q_1 \overline{Q_0}$

$W_1 = \overline{Q_2} Q_0 + Q_1 \overline{Q_0}$

$W_0 = Q_1 Q_2 + \overline{Q_1}\ \overline{Q_2}$

2:1 mux with CE

↳ $D_i = CE W_i + \overline{CE} Q_i$

$D_2 = CE(Q_2 Q_0 + Q_1 \overline{Q_0}) + \overline{CE} Q_2$

$D_1 = CE(\overline{Q_2} Q_0 + Q_1 \overline{Q_0}) + \overline{CE}\ Q_1$

$D_0 = CE(Q_1 Q_2 + \overline{Q_1}\ \overline{Q_2}) + \overline{CE}\ Q_0$