Analysis and Design of Finite State Machines

Finite State Machines

The general class of circuits in which the output depends on past behavior are called **sequential circuits**

In most cases, a **clock signal** is used to control the operation of a sequential circuit (***synchronous sequential circuit***).

Synchronous sequential circuits are realized using combinational logic and flip-flops.

Sequential circuits are also referred to as **Finite State Machines (FSM)**

FSM Design Example

Design a 3-bit synchronous Gray Code up/down counter using D Flip-Flops and gates

1. Sequence

Q_2									
Q_1									
Q_0									
State									

2. Up/Down Control

3. Synchronous

State Transition Diagrams

The **State of the circuit is defined by the Q outputs of the flip-flops**.

State transition Diagrams show states and possible from one state to another

State Transition Table

State Transition Tables are truth tables that show for each combination of state and inputs, what the Next State will be.

	Present state			*Next state*		
R	Q_2	Q_1	Q_0	Q_2	Q_1	Q_0

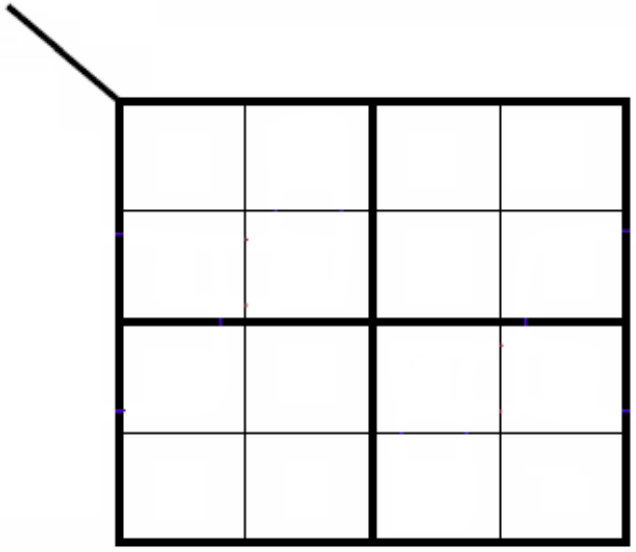
For a counter, the output is the same as the state

For D Flip Flops, Q at the next state is equal to the value of D. Therefore, the Next State Columns show what the D inputs should be in this case

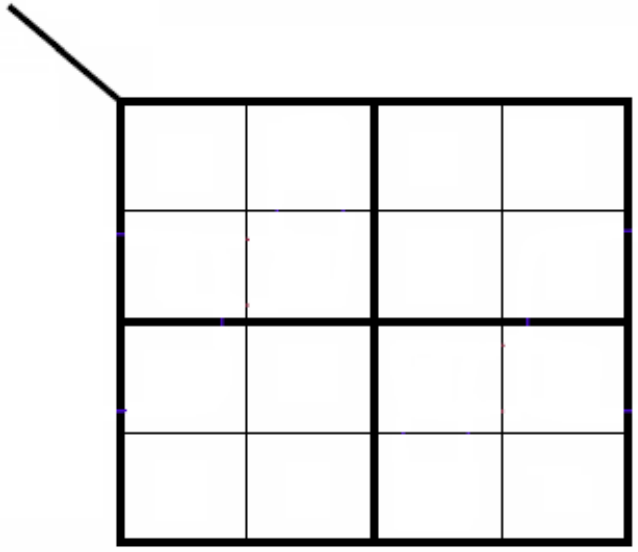
Next State Feedback Logic

We need to determine logical expressions for each flip-flop in terms of external inputs and the present state

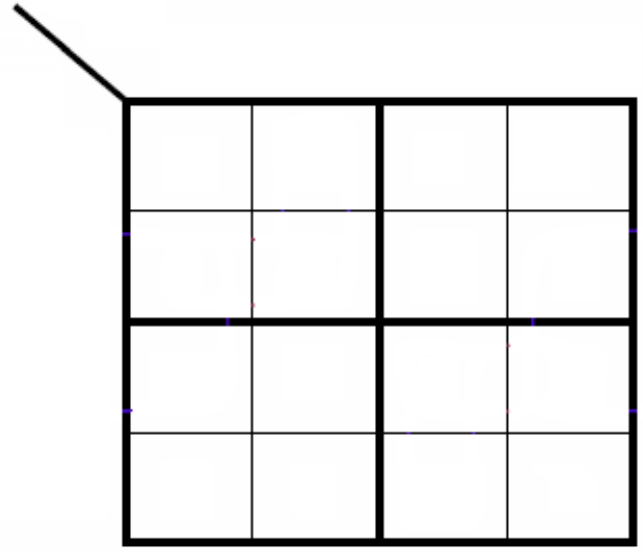
D_2



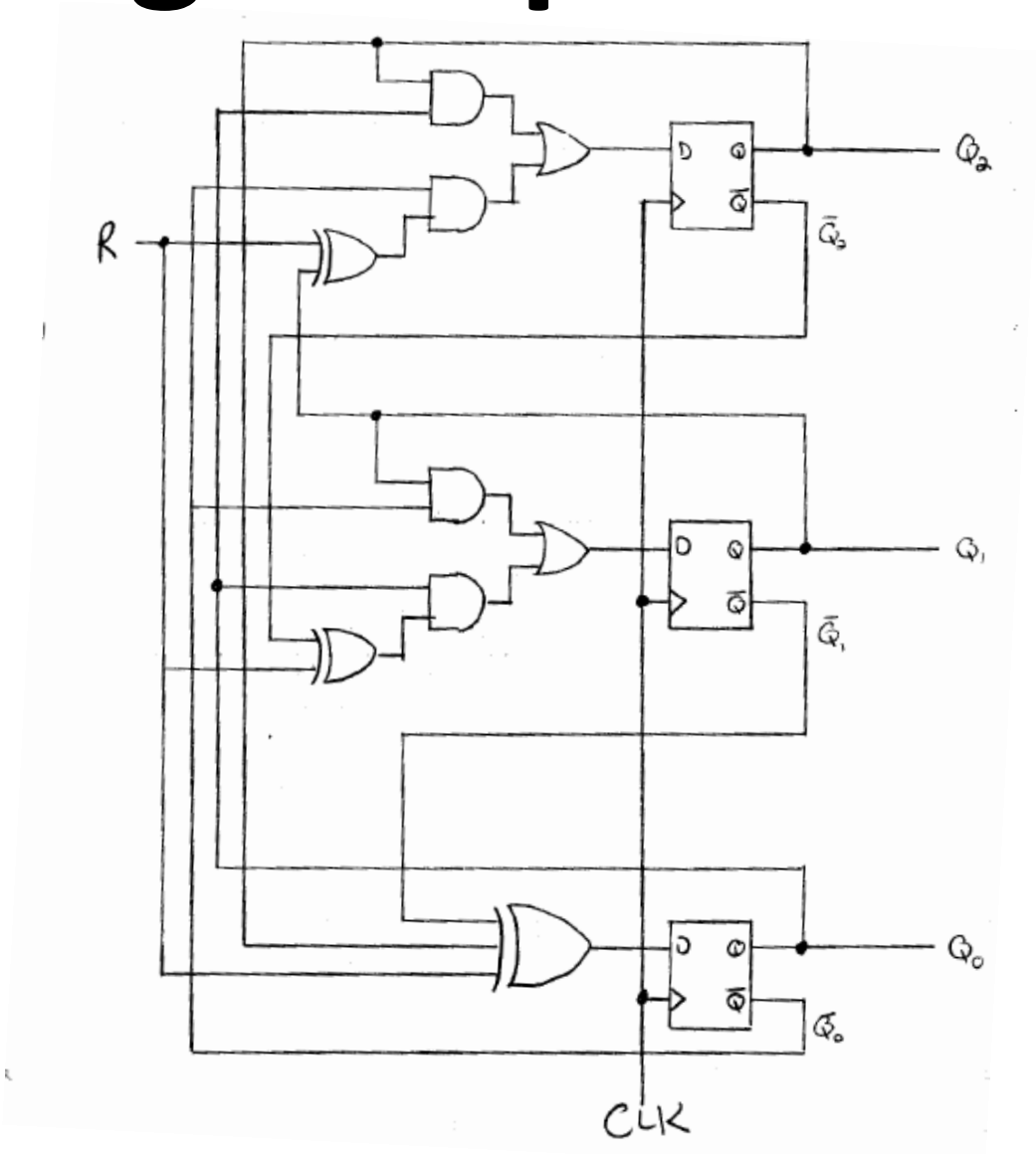
D_1



D_0



FSM Logic Implementation



Schematic for 3 Bit Up/Down Gray Code Counter

Finite State Machines

Memory: Flip-flops to store state values

State Logic: Logic network that takes the present state, inputs and produces signals that will drive the circuit to the next state

Output Logic: Produces Outputs Based on Present State (and possibly inputs)

Moore Machine → *Outputs depend only on* ***present state*** *(not inputs)*

Mealy Machine → *Outputs depend on both the* ***present state AND inputs***

FSM Design Example (II)

1. Specifications

Design a circuit to meet the following specifications:

1. inputs *w*,reset,clock, one output z
2. All changes in the circuit occur on the positive edge of the clock signal
3. The output z is equal to 1 if during the immediately preceding two clock cycles the input w was equal to 1. Otherwise the value of z is equal to 0
4. resetn = 0 puts the circuit into its default state where z = 0 (Active Low Asynchronous Reset)

Clock Cycle	t0	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10
w	0	1	0	1	1	0	1	1	1	0	1
z											

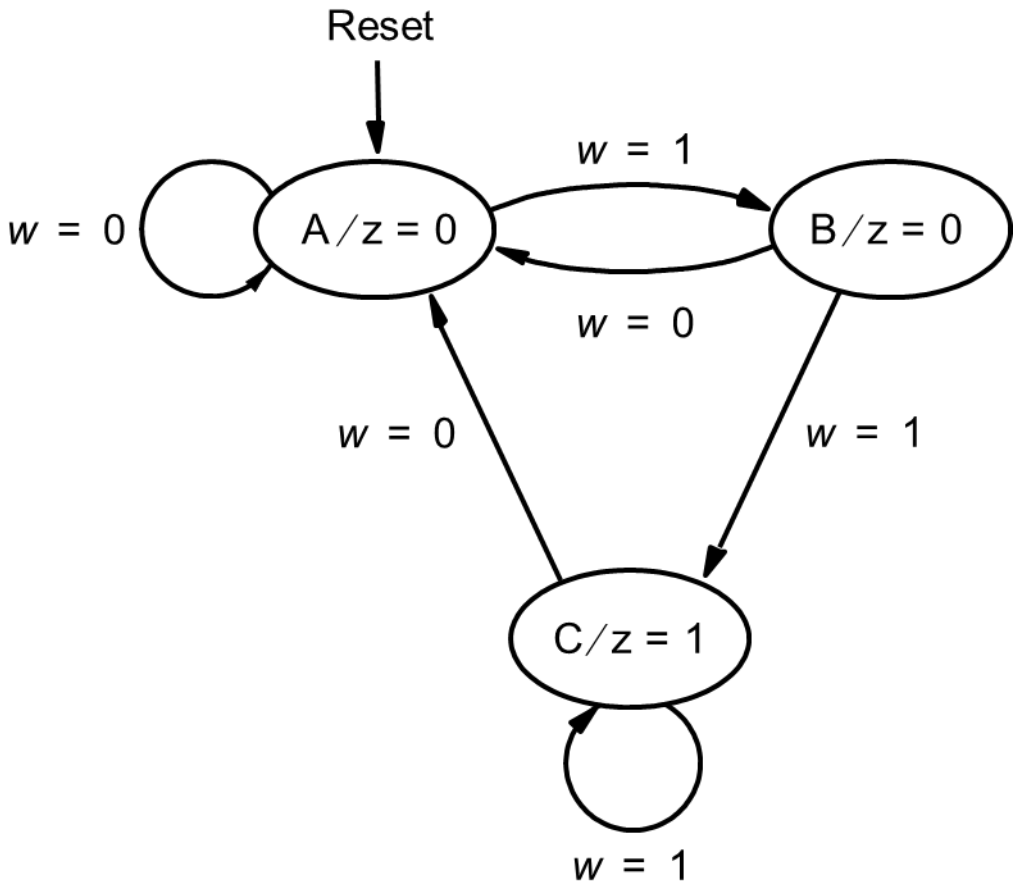
FSM Design Example (II)

2. State Transition Diagram

FSM Design Example (II)

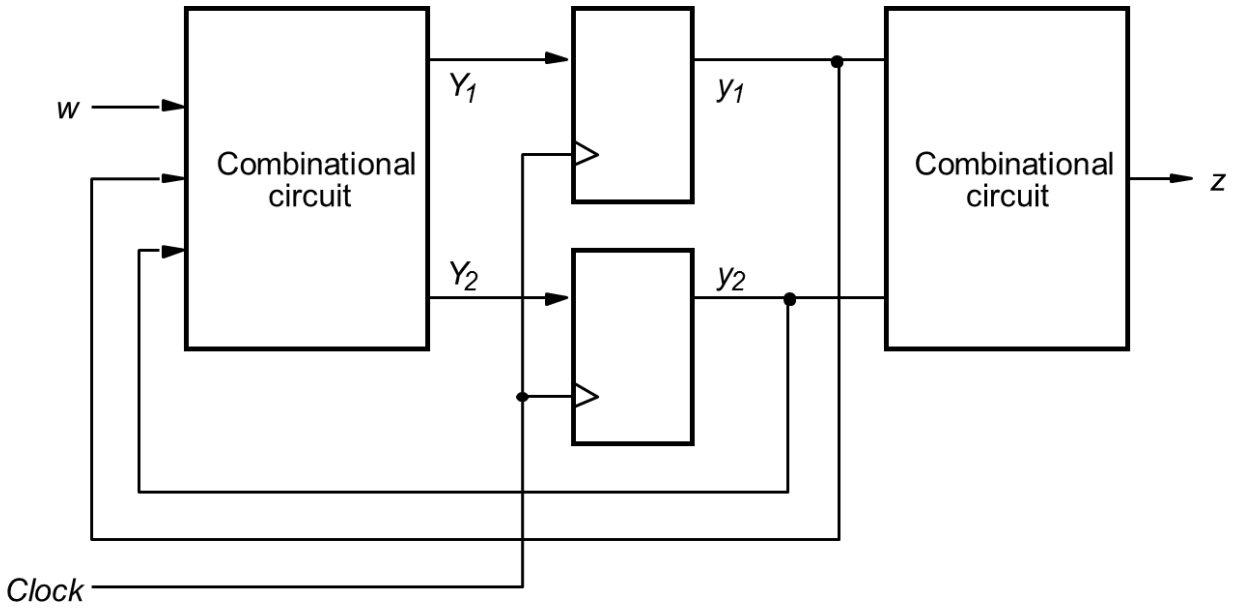
3. Draw the State Table

Present state	Next state	Output z
A B C		



FSM Design Example (II)

4. State Assignment

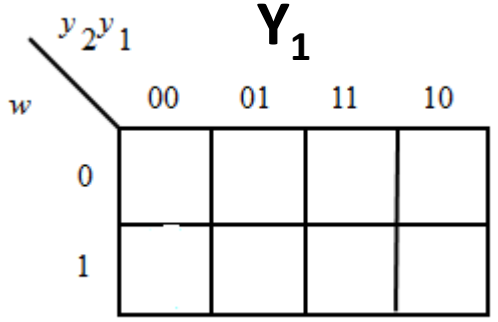


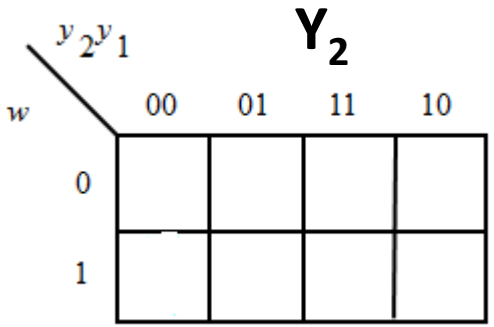
Present state y_2y_1	Next state $w = 0$ Y_2Y_1	Next state $w = 1$ Y_2Y_1	Output z

FSM Design Example (II)

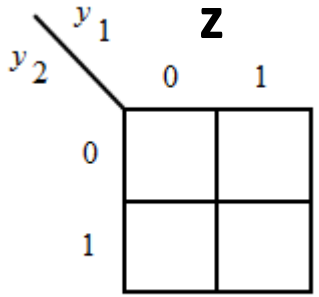
5. Derive Next-State and Output Expressions

Next State Expressions



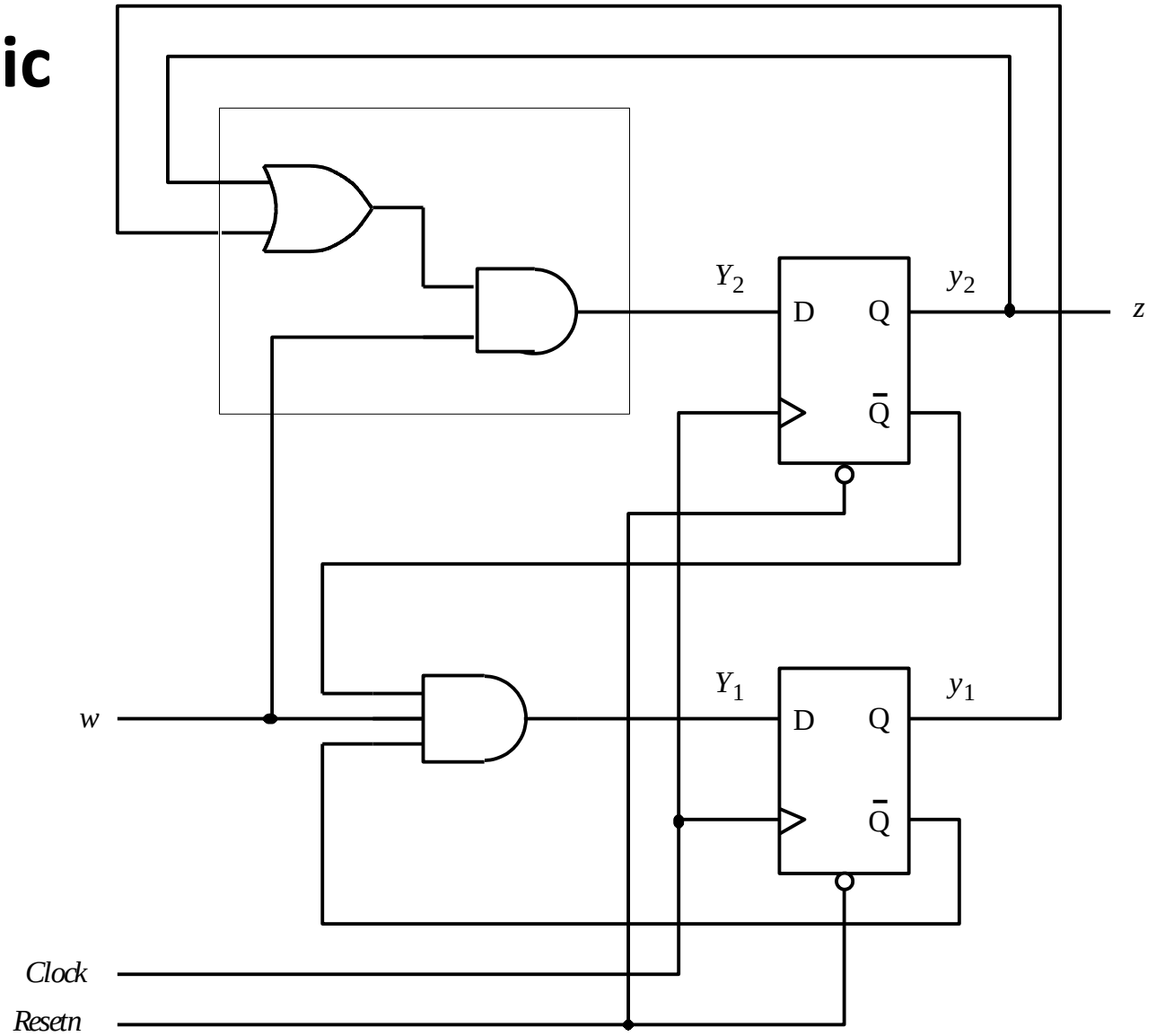


Output Expressions



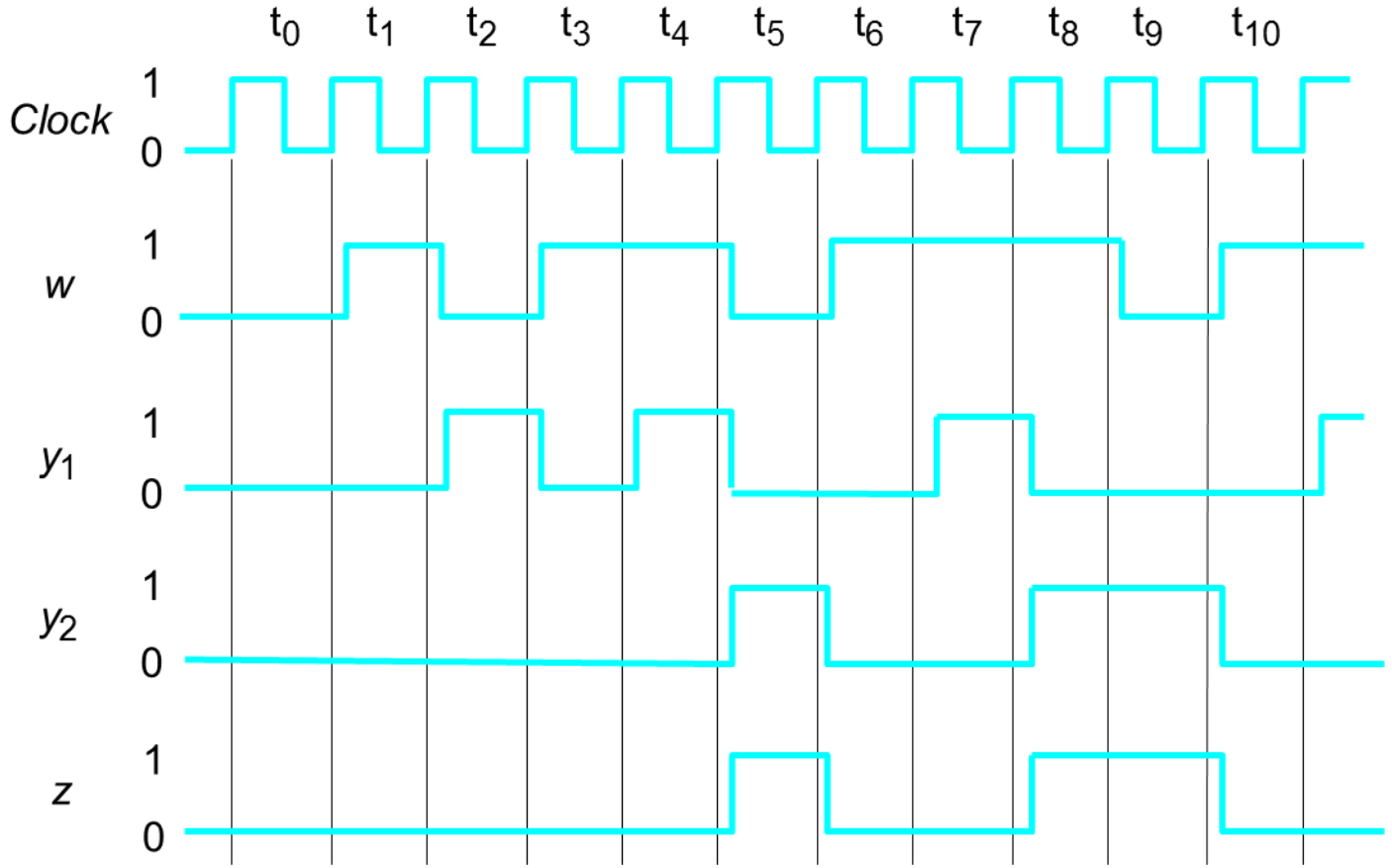
FSM Design Example (II)

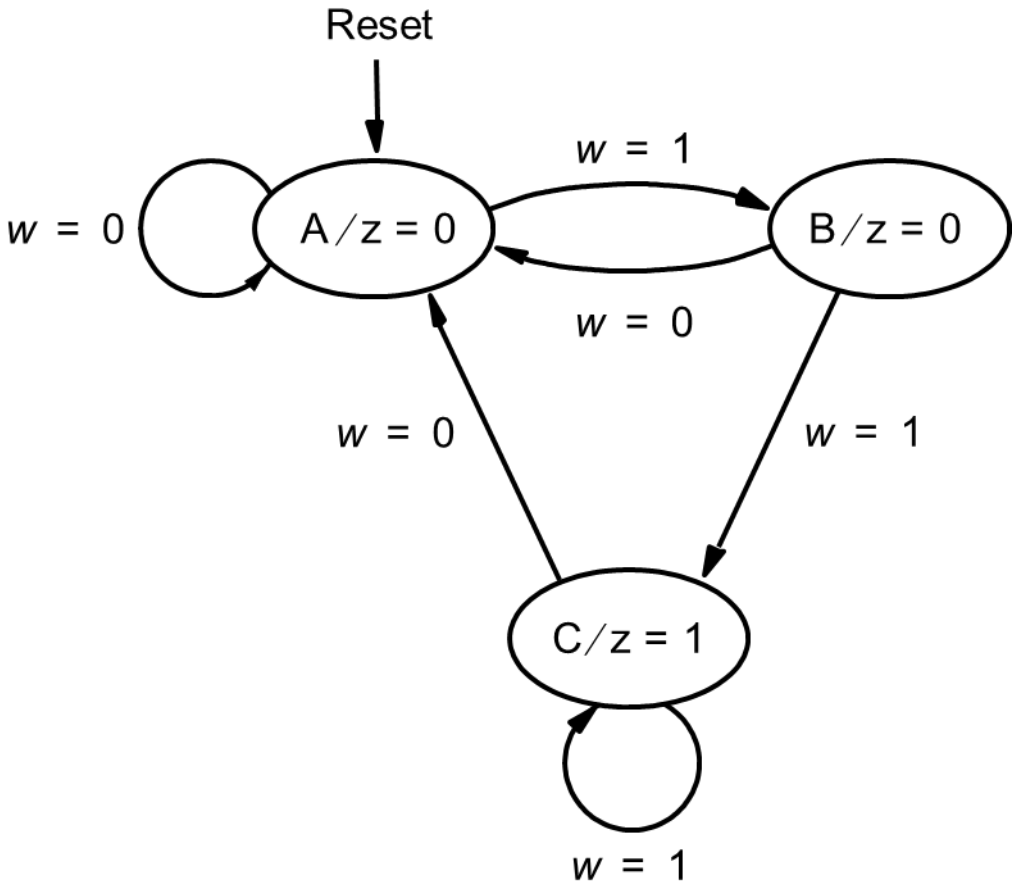
6. **Draw the schematic**



FSM Design Example (II)

6. Timing Diagram Verification



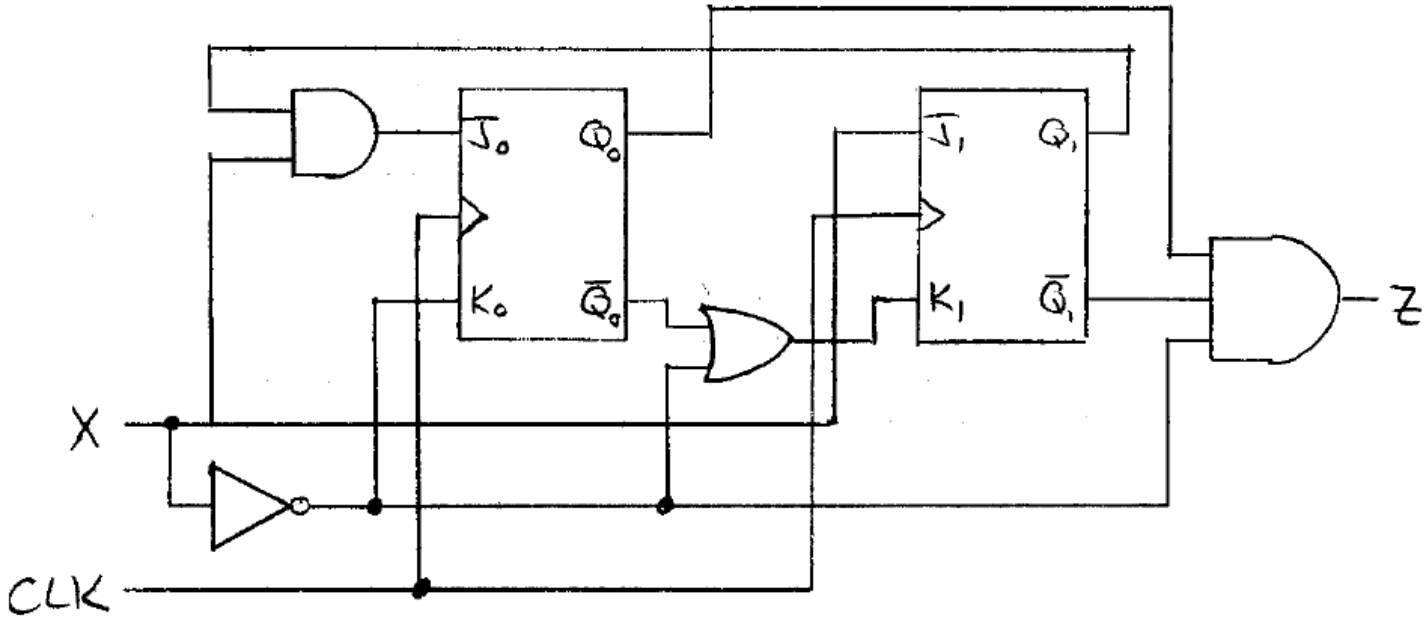


FSM Analysis Example: Determine how the sequential circuit below operates

1. FSM Type

2. State Logic

3. Output Logic



Schematic

4. State Transition Table

Include "Excitations": Show flip-flop inputs produced by state logic, use them to determine Next State

Input	PS	Next State Excitations		NS	Output
X	$Q_1\ Q_0$	$J_1\ K_1$	$J_0\ K_0$	$Q_1\ \ Q_0$	Z

FSM Analysis Example

5. State Diagram

Mealy Notation→ Input / Output on transistions

6. Behavioral Description

From any State, go to State 0 if X = 0

Can only reach State 2 from State 0 when X = 1

Can only reach State 1 from State 2 when X = 1

Reach State 3 from either State 1 or State 3 when X = 1

Z = 1 only when in State 1 and X = 0

7. I/O Example

X 0 1 0 0 1 1 0 1 1 0 1 1 1 0 0 1 1 0 0 1
Z 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0