Week 6 Lectorial

Question #1
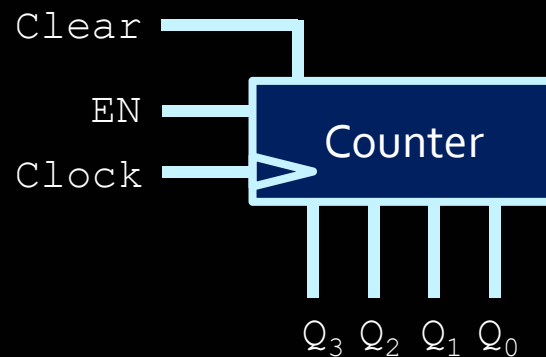
- Imagine you have access to a 4-bit register.

D_3 D_2 D_1 D_0

Write

Clock

Register

Q_3 Q_2 Q_1 Q_0
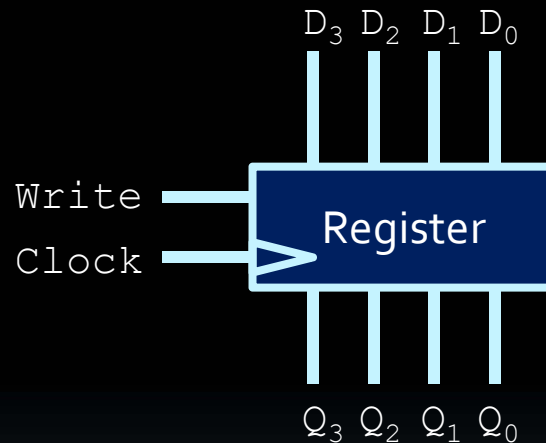
- What does the `Write` signal do?

Question #2

- Assume that you have access to a counter circuit:

Clear ———┐
          │
EN ——— ┌─────────┐
       │ Counter │
Clock ─▷│   │
       └─┬─┬─┬─┬─┘
         │ │ │ │
        Q_3 Q_2 Q_1 Q_0

- How do you make a signal that goes high after 10 clock cycles?

- How do you make a signal that goes high every 10 clock cycles?

Question #2 (cont'd)

- How do you make a signal that goes high every 100 clock cycles, only using 4-bit counters like the one below (and a few additional gates)?

Clear

EN

Clock

Counter

Q_3 Q_2 Q_1 Q_0
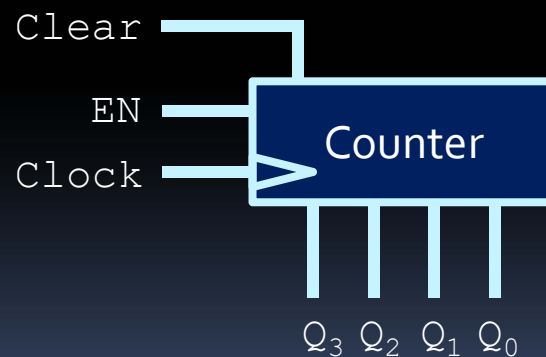
Question #3

- How many flip-flops would you need to implement the following finite state machine (FSM)?

  - 11 states
  - # flip-flops = $\lceil \log_2 (\text{# of states}) \rceil$

  - # flip-flops = 4

Zero — 5¢ → Five

Zero — 10¢ → Ten

Five — 5¢ → Ten

Five — 10¢ → Fifteen

Ten — 5¢ → Fifteen

Ten — 10¢ → Twenty

Fifteen — 5¢ → Twenty

Fifteen — 10¢ → Twenty-Five

Twenty — 5¢ → Twenty-Five

Twenty — 10¢ → Thirty

Twenty-Five — 5¢ → Thirty

Twenty-Five — 10¢ → Thirty-Five

Thirty — 5¢ → Thirty-Five

Thirty — 10¢ → Forty

Thirty-Five — 5¢ → Forty

Thirty-Five — 10¢ → Forty-Five

Forty — 5¢ → Forty-Five

Forty — 10¢ → Fifty

Forty-Five — 5¢ → Fifty

Question #4

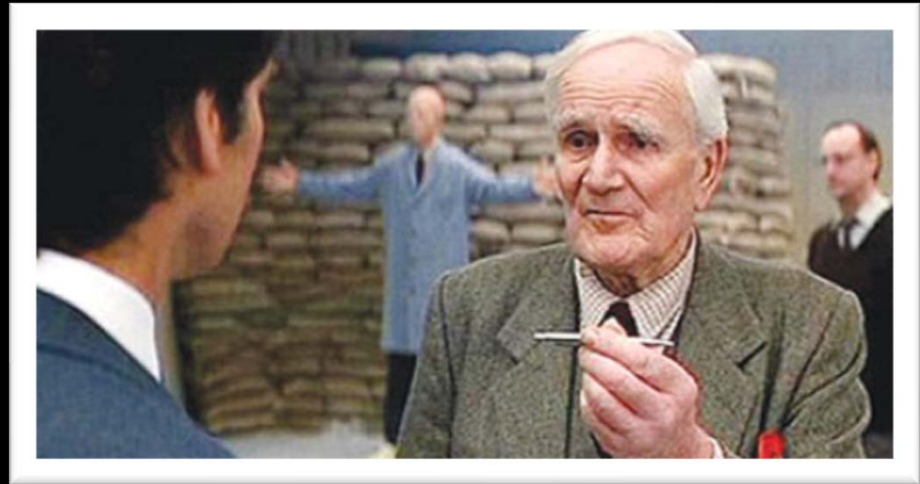- How would we make the following Finite State Machine?

Exploding pen continued…

Making the James Bond pen

- Pen starts off in disarmed state.
- When clicked three times, pen arms itself.

- When clicked three more times, pen disarms itself.
- What are the steps to making this circuit?

Reminder: How to Design FSM

- As a brief reminder:
    1. Draw state diagram
    2. Derive state table from state diagram
    3. Assign flip-flop configuration to each state
        - Number of flip-flops needed is: $\lceil \log(\text{\# of states}) \rceil$
    4. Redraw state table with flip-flop values
    5. Derive combinational circuit for output and for each flip-flop input.

Review of FSMs

- Step 5 requires two combinational circuit design tasks.
  - For Moore machines (pictured bottom right), output is determined solely based on current state (i.e. flip-flop values).

Inputs → **Combinational Circuit** → Outputs

Storage Units

Inputs → **State Logic** **Output Logic**

Flip-Flops
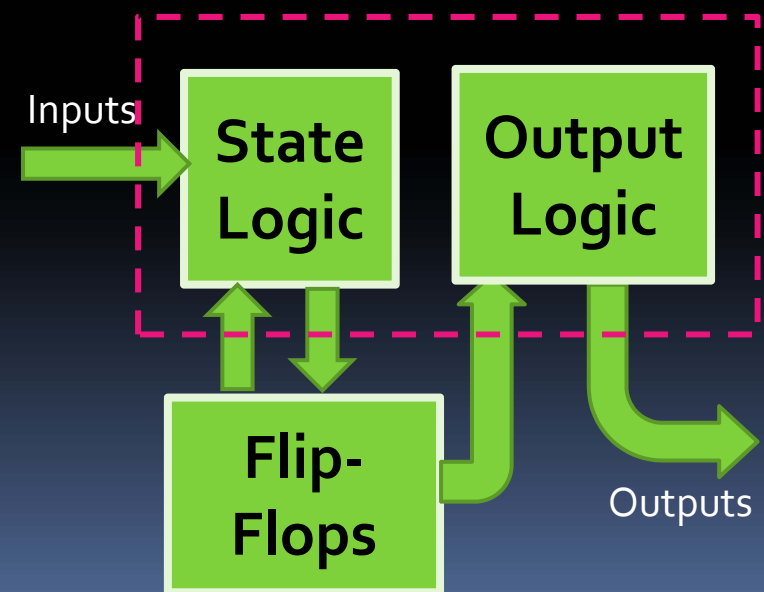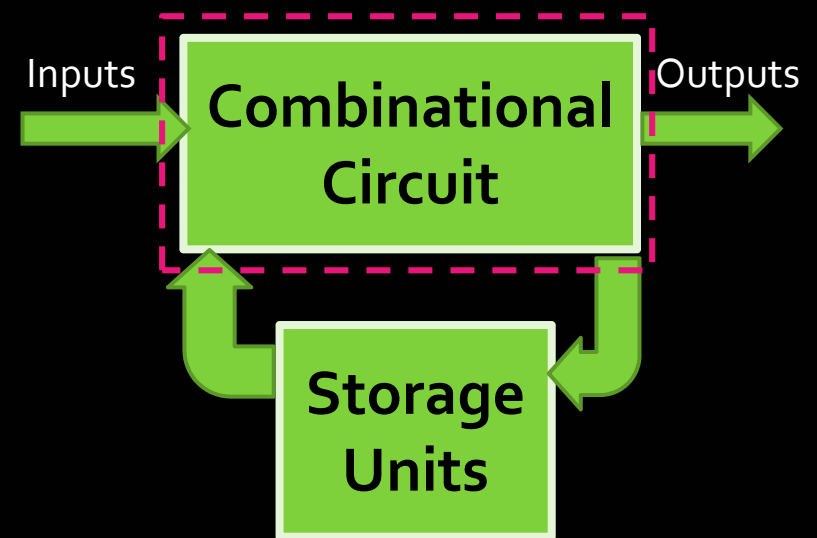
Outputs

Review of FSMs

- For Mealy machines, output is determined by both the current state and the current input values.

   - For simplicity, most of our examples will focus on Moore machines.

Inputs → **Combinational Circuit** → Outputs

Storage Units

Inputs

State Logic   **Output Logic**

Flip-Flops
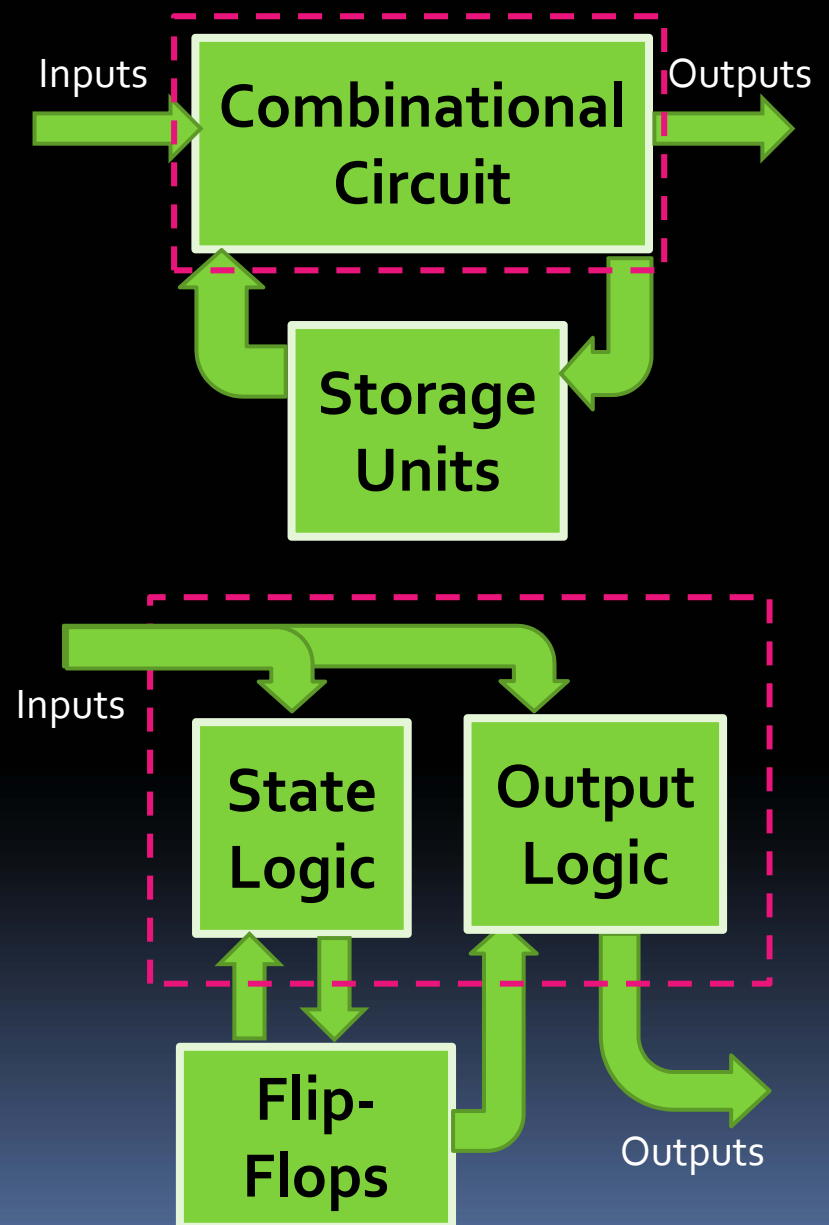
Outputs

State diagrams with output

- Output values are incorporated into the state diagram, depending on the machine used.

➢ Moore Machine

State/Output(s) · A/0 — 1 → B/1 · Input(s)
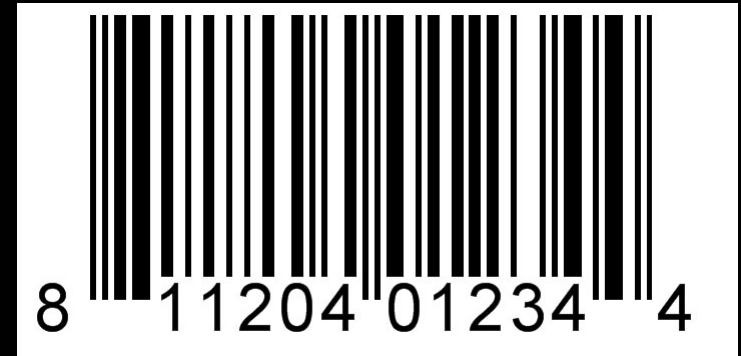
➢ Mealy Machine

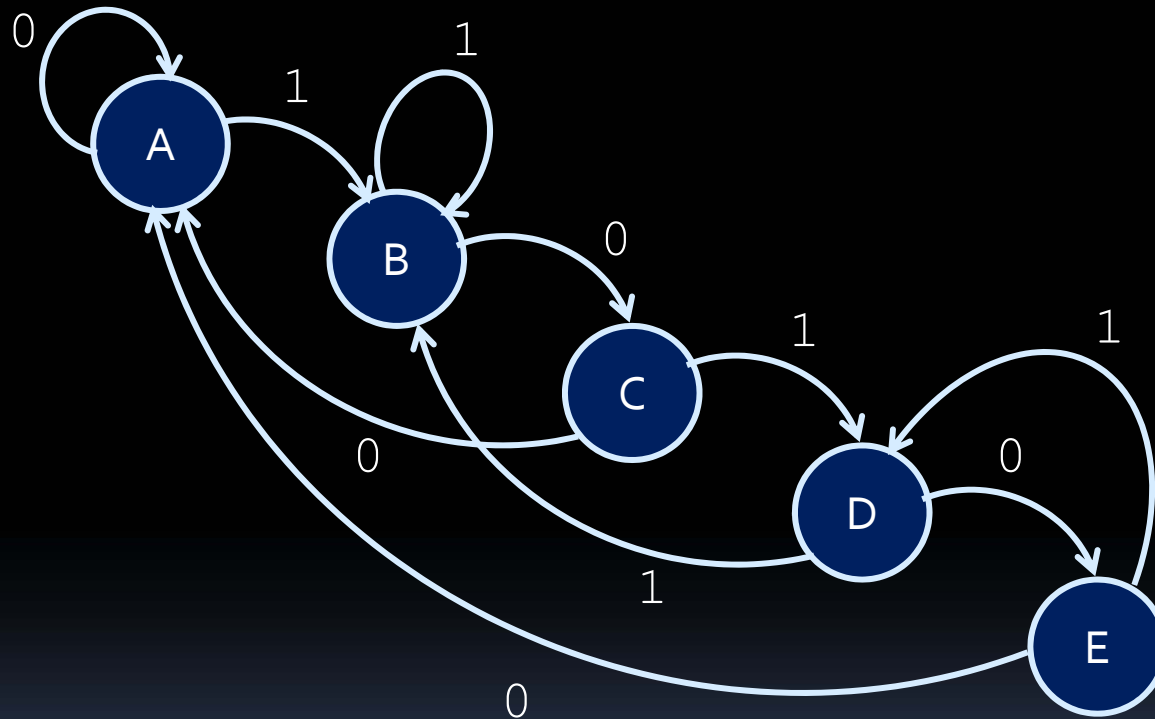State · C — 1/0 → D · Input(s) / Output(s)
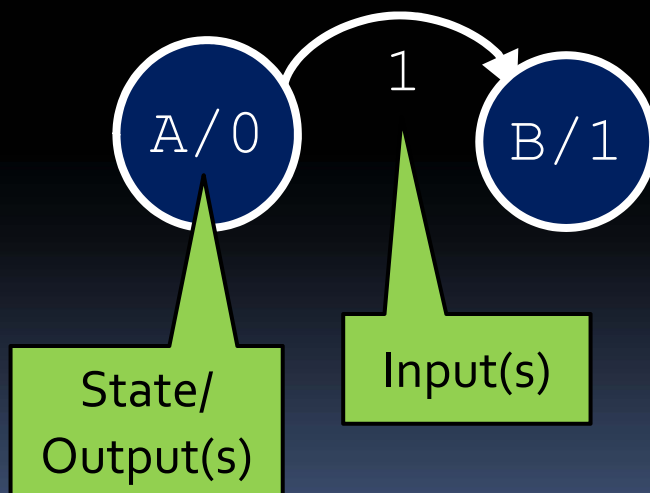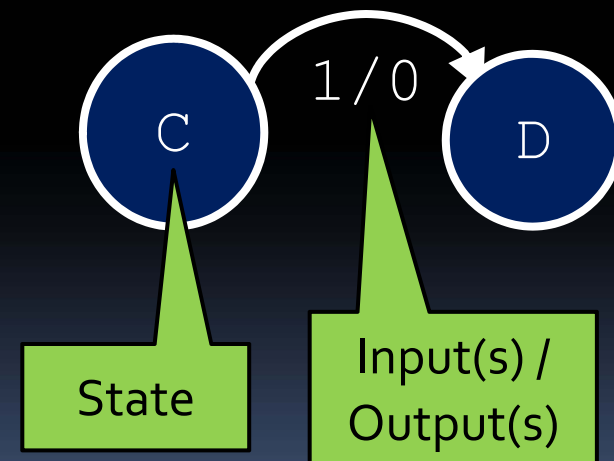
FSM Example: Barcode Reader

- When scanning UPC barcodes, the laser scanner looks for black and white bars that indicate the start of the code.

- If black is read as a 1 and white is read as a 0, the start of the code (from either direction) has a 1010 pattern.

  - Can you create a state machine that detects this pattern?

Step #1: Draw state diagram

Step #2: State Table

- Output Z is determined by the current state.
  - Denotes Moore machine.
- Next step: allocate flip-flops values to each state.
  - How many flip-flops will we need for 5 states?
  - Recall:
    - # flip-flops = $\lceil \log(\text{# of states}) \rceil$

Present State	Z	x	Next State
A	0	0	A
A	0	1	B
B	0	0	C
B	0	1	B
C	0	0	A
C	0	1	D
D	0	0	E
D	0	1	B
E	1	0	A
E	1	1	D

Step #3: Flip-Flop Assignment

- 3 flip-flops needed here.
- Assign states carefully though!
- Can't simply do this:
  - ➤ A = 100
  - ➤ B = 011
  - ➤ C = 010
  - ➤ D = 001
  - ➤ E = 000
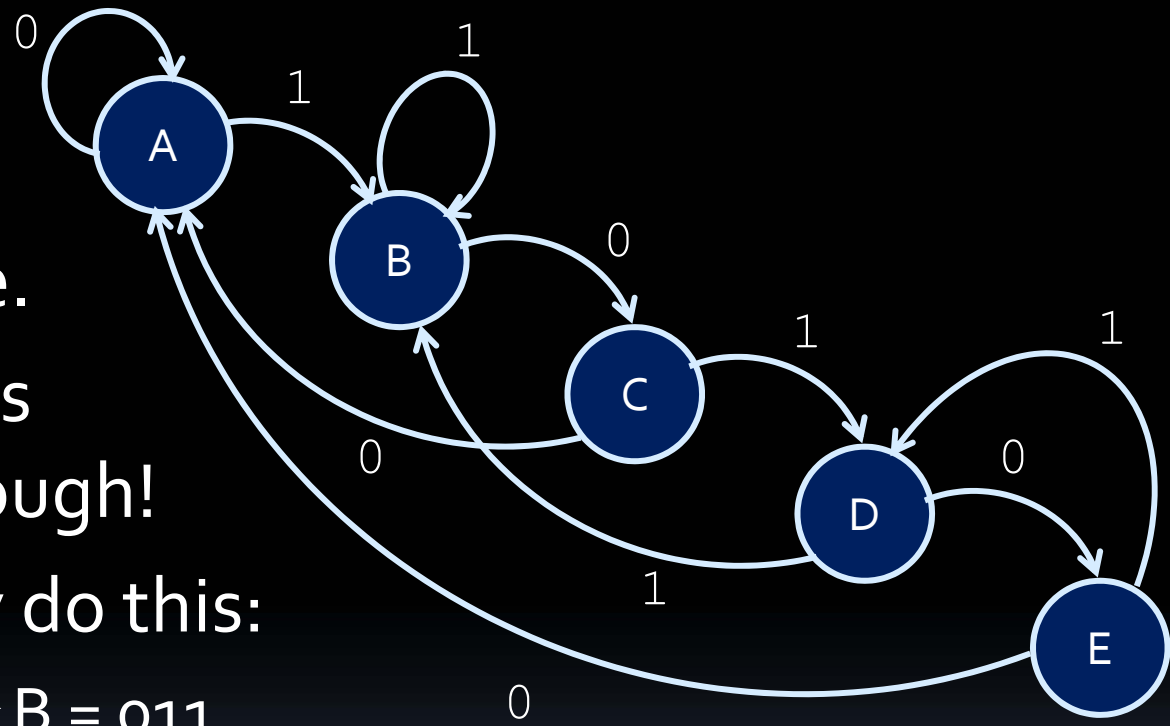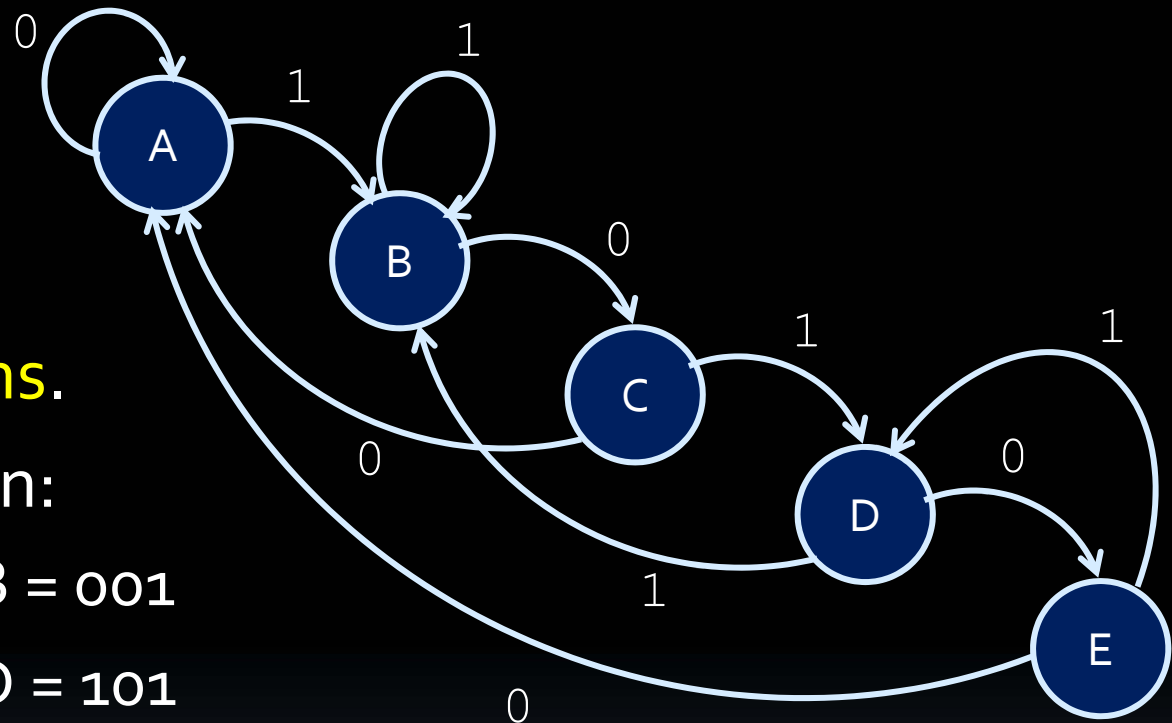
Why not?

Step #3: Flip-Flop Assignment

- Be careful of **race conditions**.

- Better solution:
  - A = 000
  - B = 001
  - C = 011
  - D = 101
  - E = 100

- Still has race conditions (C→D, C→A), but is safer.
  - "Safer" is defined according to output behaviour.
  - Sometimes, extra flip-flops are used for extra insurance.

Step #4: Redraw State Table

- From here, we can construct the K-maps for the state logic combinational circuit.

  - Derive equations for each flip-flop value, given the previous values and the input X.

  - Three equations total, plus one more for Z (trivial for Moore machines).

Present State			Z	X	Next State		
0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	1
0	0	1	0	0	0	1	1
0	0	1	0	1	0	0	1
0	1	1	0	0	0	0	0
0	1	1	0	1	1	0	1
1	0	1	0	0	1	0	0
1	0	1	0	1	0	0	1
1	0	0	1	0	0	0	0
1	0	0	1	1	1	0	1

Step 5: Circuit design

- Karnaugh map for F_2:

	$\overline{\mathbf{F}}_0 \cdot \overline{\mathbf{X}}$	$\overline{\mathbf{F}}_0 \cdot \mathbf{X}$	$\mathbf{F}_0 \cdot \mathbf{X}$	$\mathbf{F}_0 \cdot \overline{\mathbf{X}}$
$\overline{\mathbf{F}}_2 \cdot \overline{\mathbf{F}}_1$	0	0	0	0
$\overline{\mathbf{F}}_2 \cdot \mathbf{F}_1$	X	X	1	0
$\mathbf{F}_2 \cdot \mathbf{F}_1$	X	X	X	X
$\mathbf{F}_2 \cdot \overline{\mathbf{F}}_1$	0	1	0	1

$$F_2 = F_1 X + F_2 \overline{F}_0 X + F_2 F_0 \overline{X}$$

Step 5: Circuit design

- Karnaugh map for F_1:

	$\overline{F_0} \cdot \overline{X}$	$\overline{F_0} \cdot X$	$F_0 \cdot X$	$F_0 \cdot \overline{X}$
$\overline{F_2} \cdot \overline{F_1}$	0	0	0	1
$\overline{F_2} \cdot F_1$	X	X	0	0
$F_2 \cdot F_1$	X	X	X	X
$F_2 \cdot \overline{F_1}$	0	0	0	0

$$F_1 = \overline{F_2}\overline{F_1}F_0\overline{X}$$

Step 5: Circuit design

- Karnaugh map for F_o:

	$\overline{F}_0 \cdot \overline{X}$	$\overline{F}_0 \cdot X$	$F_0 \cdot X$	$F_0 \cdot \overline{X}$
$\overline{F}_2 \cdot \overline{F}_1$	0	1	1	1
$\overline{F}_2 \cdot F_1$	X	X	1	0
$F_2 \cdot F_1$	X	X	X	X
$F_2 \cdot \overline{F}_1$	0	1	1	0

$$F_0 \ = \ X \ + \ \overline{F}_2 \overline{F}_1 F_0$$

Step 5: Circuit design

- Output value Z goes high based on the following output equation:

$$Z = F_2\overline{F_1}\overline{F_0}$$

- Note: All of these equations would be different, given different flip-flop assignments!

  - Practice alternate assignment for the midterm ☺