# EECS 270 – Introduction to Logic Design
# Winter 2013

# 12. Finite State Machine Design

**Profs. Kang Shin & Valeria Bertacco**
EECS Department
University of Michigan, Ann Arbor

# FSM Design

- Goal: Design a FSM that satisfies the requirements of the given problem description (spec.)
- Follow FSM analysis steps in reverse! (more or less)

1) (optional) Construct state diagram
2) Construct state/output table
3) Create state assignments
4) Create transition/output table

*"Art of design"*

---

5) Choose FF type
6) Construct **excitation/output table**
   - Similar to transition/output table
7) Find excitation and output logic equations

*"Turn the crank"*

# FSM Design Example

- Problem description: design a Moore FSM with one input IN and one output OUT, such that OUT is one iff IN is 1 for three consecutive clock cycles
- State table:

| S | IN = 0 ($S^+$) | IN = 1 ($S^+$) | OUT |
|---|---|---|---|
| zero1s | zero1s | one1 | 0 |
| one1 | zero1s | two1s | 0 |
| two1s | zero1s | three1s | 0 |
| three1s | zero1s | three1s | 1 |

- State Assignments
  - How many state variables are needed to encode four states?

$$2$$

  - In general, if we have *n* states, how many state variables are needed to encode those states?

$$\lceil \log_2 n \rceil$$

| State name | $Q_1$ | $Q_0$ |
|---|---|---|
| zero1s | 0 | 0 |
| one1 | 0 | 1 |
| two1s | 1 | 0 |
| three1s | 1 | 1 |

*These state assignments may seem rather arbitrary – that's because they are! We will soon see the impact that state assignments have on our final circuit…*

- Transition/output table

*State/Output Table:*

| S | IN 0 | IN 1 | OUT |
|---|---|---|---|
| zero1s | zero1s | one1 | 0 |
| one1 | zero1s | two1s | 0 |
| two1s | zero1s | three1s | 0 |
| three1s | zero1s | three1s | 1 |
| | $S^+$ | | |

*State Assignments*

| State name | $Q_1$ | $Q_0$ |
|---|---|---|
| zero1s | 0 | 0 |
| one1 | 0 | 1 |
| two1s | 1 | 0 |
| three1s | 1 | 1 |

*Transition/Output Table:*

| $Q_1$ | $Q_0$ | IN 0 | IN 1 | OUT |
|---|---|---|---|---|
| 0 | 0 | 00 | 01 | 0 |
| 0 | 1 | 00 | 10 | 0 |
| 1 | 0 | 00 | 11 | 0 |
| 1 | 1 | 00 | 11 | 1 |
| | | $Q_1^+ Q_0^+$ | | |

- Choose FF type:
  - Using D flip-flops will simplify things (as we'll see below…)

- Excitation table
  - Shows *FF input values required to create next state values* for every current state/input combination
  - If we're designing with D FFs, entries in excitation/output table are the same as those in transition/output table!
    - Because of D FF characteristic equation: $Q^+ = D$

| $Q_1$ | $Q_0$ | IN 0 | IN 1 | OUT |
|---|---|---|---|---|
| 0 | 0 | 00 | 01 | 0 |
| 0 | 1 | 00 | 10 | 0 |
| 1 | 0 | 00 | 11 | 0 |
| 1 | 1 | 00 | 11 | 1 |
| | | $D_1 D_0$ | | |

- Excitation Logic

*Excitation/Output Table:*

| $Q_1$ | $Q_0$ | IN = 0 ($D_1 D_0$) | IN = 1 ($D_1 D_0$) | OUT |
|---|---|---|---|---|
| 0 | 0 | 00 | 01 | 0 |
| 0 | 1 | 00 | 10 | 0 |
| 1 | 0 | 00 | 11 | 0 |
| 1 | 1 | 00 | 11 | 1 |

$D_1$

| IN \ $Q_1Q_0$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | | | | |
| 1 | | 1 | 1 | 1 |

$$D_1 = Q_0 \cdot IN + Q_1 \cdot IN$$
$$= IN \cdot (Q_0 + Q_1)$$

$D_0$

| IN \ $Q_1Q_0$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | | | | |
| 1 | 1 | | 1 | 1 |

$$D_0 = Q_1 \cdot IN + \overline{Q_0} \cdot IN$$
$$= IN \cdot (Q_1 + \overline{Q_0})$$

- Output Logic

$$OUT = Q_1 \cdot Q_0$$

- Circuit:

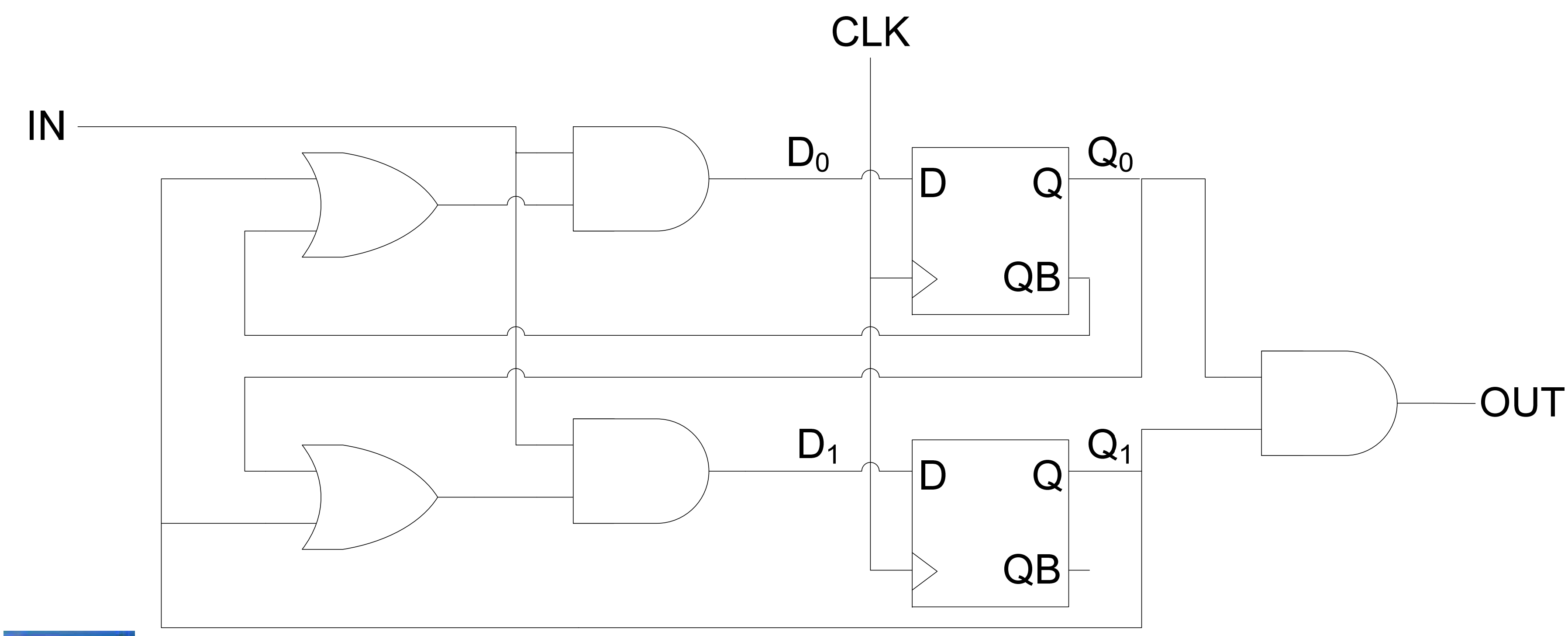
*Excitation Equations:*

$$D_0 = IN \cdot \left(Q_1 + \overline{Q_0}\right)$$

$$D_1 = IN \cdot \left(Q_0 + Q_1\right)$$

*Output Equation:*

$$OUT = Q_1 \cdot Q_0$$



# In Class Exercise

- Design a state/output table for the following problem specification:

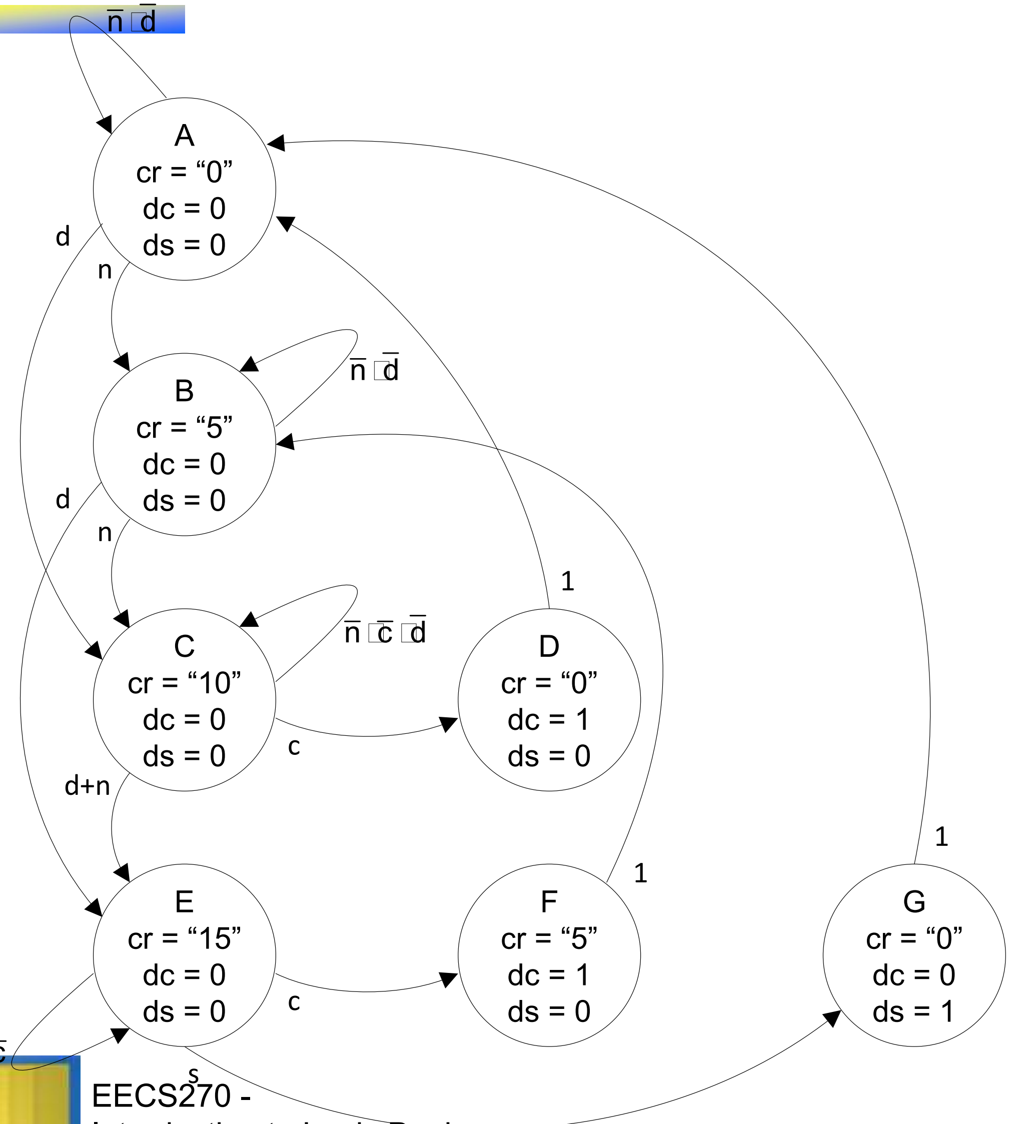
  Combination lock: Two inputs, X and Y, encode a binary number between 0 and 3 (X is MSB, i.e., XY = 10 →2). A single output signal UNLOCK should be set to 1 iff the sequence 1, 2, 1 occurs on the inputs in three consecutive clock cycles

| | | X Y | | | | |
|---|---|---|---|---|---|---|
| | S | 00 | 01 | 10 | 11 | UNLOCK |
| *got nothing* | A | A | B | A | A | 0 |
| *got “1”* | B | A | B | C | A | 0 |
| *got “1,2”* | C | A | D | A | A | 0 |
| *got “1,2,1”* | D | A | B | C | A | 1 |
| | | | $S^+$ | | | |

# FSM Transition List Design: 50s Vending Machine

- Inputs
  - **d**: asserted when user inserts dime
  - **n**: asserted when user inserts nickel
  - **c**: asserted when user presses candy button
  - **s**: asserted when user presses soda button
- Outputs
  - **dc**: dispenses candy when asserted
  - **ds**: dispenses soda when asserted
  - **cr**: 4-bit unsigned number, represents the user's credit
- Specifications
  - All inputs are one-hot
  - Candy costs 10 cents, soda costs 15 cents
  - Money need only be counted up to 15 cents

# Vending Machine State Diagram and Transition List




*Output Table*

| | $Q_2Q_1Q_0$ | cr | dc | ds |
|---|---|---|---|---|
| A | 0 0 0 | “0” | 0 | 0 |
| B | 0 0 1 | “5” | 0 | 0 |
| C | 0 1 0 | “10” | 0 | 0 |
| D | 0 1 1 | “0” | 1 | 0 |
| E | 1 0 0 | “15” | 0 | 0 |
| F | 1 0 1 | “5” | 1 | 0 |
| G | 1 1 0 | “0” | 0 | 1 |

*Transition List*

| S | $Q_2Q_1Q_0$ | Transition Expression | $S^+$ | $Q_2^+Q_1^+Q_0^+$ |
|---|---|---|---|---|
| A | 0 0 0 | n | B | 0 0 1 |
| A | 0 0 0 | d | C | 0 1 0 |
| A | 0 0 0 | $\bar{n}\cdot\bar{d}$ | A | 0 0 0 |
| B | 0 0 1 | n | C | 0 1 0 |
| B | 0 0 1 | d | E | 1 0 0 |
| B | 0 0 1 | $\bar{n}\cdot\bar{d}$ | B | 0 0 1 |
| C | 0 1 0 | n+d | E | 1 0 0 |
| C | 0 1 0 | c | D | 0 1 1 |
| C | 0 1 0 | $\bar{n}\cdot\bar{d}\cdot\bar{c}$ | C | 0 1 0 |
| D | 0 1 1 | 1 | A | 0 0 0 |
| E | 1 0 0 | c | F | 1 0 1 |
| E | 1 0 0 | s | G | 1 1 0 |
| E | 1 0 0 | $\bar{s}\cdot\bar{c}$ | E | 1 0 0 |
| F | 1 0 1 | 1 | B | 0 0 1 |
| G | 1 1 0 | 1 | A | 0 0 0 |

*Transition List*

| S | $Q_2Q_1Q_0$ | Transition Expression | $S^+$ | $Q_2^+Q_1^+Q_0^+$ |
|---|---|---|---|---|
| A | 0 0 0 | $n$ | B | 0 0 1 |
| A | 0 0 0 | $d$ | C | 0 1 0 |
| A | 0 0 0 | $\bar{n}\cdot\bar{d}$ | A | 0 0 0 |
| B | 0 0 1 | $n$ | C | 0 1 0 |
| B | 0 0 1 | $d$ | E | 1 0 0 |
| B | 0 0 1 | $\bar{n}\cdot\bar{d}$ | B | 0 0 1 |
| C | 0 1 0 | $n+d$ | E | 1 0 0 |
| C | 0 1 0 | $c$ | D | 0 1 1 |
| C | 0 1 0 | $\bar{n}\cdot\bar{d}\cdot\bar{c}$ | C | 0 1 0 |
| D | 0 1 1 | 1 | A | 0 0 0 |
| E | 1 0 0 | $c$ | F | 1 0 1 |
| E | 1 0 0 | $s$ | G | 1 1 0 |
| E | 1 0 0 | $\bar{s}\cdot\bar{c}$ | E | 1 0 0 |
| F | 1 0 1 | 1 | B | 0 0 1 |
| G | 1 1 0 | 1 | A | 0 0 0 |

$$D_2 = Q_2^+ = \overline{Q}_2\overline{Q}_1Q_0d + \overline{Q}_2Q_1\overline{Q}_0(n+d) + Q_2\overline{Q}_1\overline{Q}_0c + Q_2\overline{Q}_1\overline{Q}_0s + Q_2\overline{Q}_1\overline{Q}_0\bar{s}\bar{c}$$

$$D_1 = Q_1^+ = \overline{Q}_2\overline{Q}_1\overline{Q}_0d + \overline{Q}_2\overline{Q}_1Q_0n + \overline{Q}_2Q_1\overline{Q}_0c + \overline{Q}_2Q_1\overline{Q}_0\bar{n}\bar{d}\bar{c} + Q_2\overline{Q}_1\overline{Q}_0s$$

$$D_0 = Q_0^+ = \overline{Q}_2\overline{Q}_1\overline{Q}_0n + \overline{Q}_2\overline{Q}_1Q_0\bar{n}\bar{d} + \overline{Q}_2Q_1\overline{Q}_0c + Q_2\overline{Q}_1\overline{Q}_0c + Q_2\overline{Q}_1Q_0$$

*Output Table*

| | $Q_2Q_1Q_0$ | cr | dc | ds |
|---|---|---|---|---|
| A | 0 0 0 | “0” | 0 | 0 |
| B | 0 0 1 | “5” | 0 | 0 |
| C | 0 1 0 | “10” | 0 | 0 |
| D | 0 1 1 | “0” | 1 | 0 |
| E | 1 0 0 | “15” | 0 | 0 |
| F | 1 0 1 | “5” | 1 | 0 |
| G | 1 1 0 | “0” | 0 | 1 |

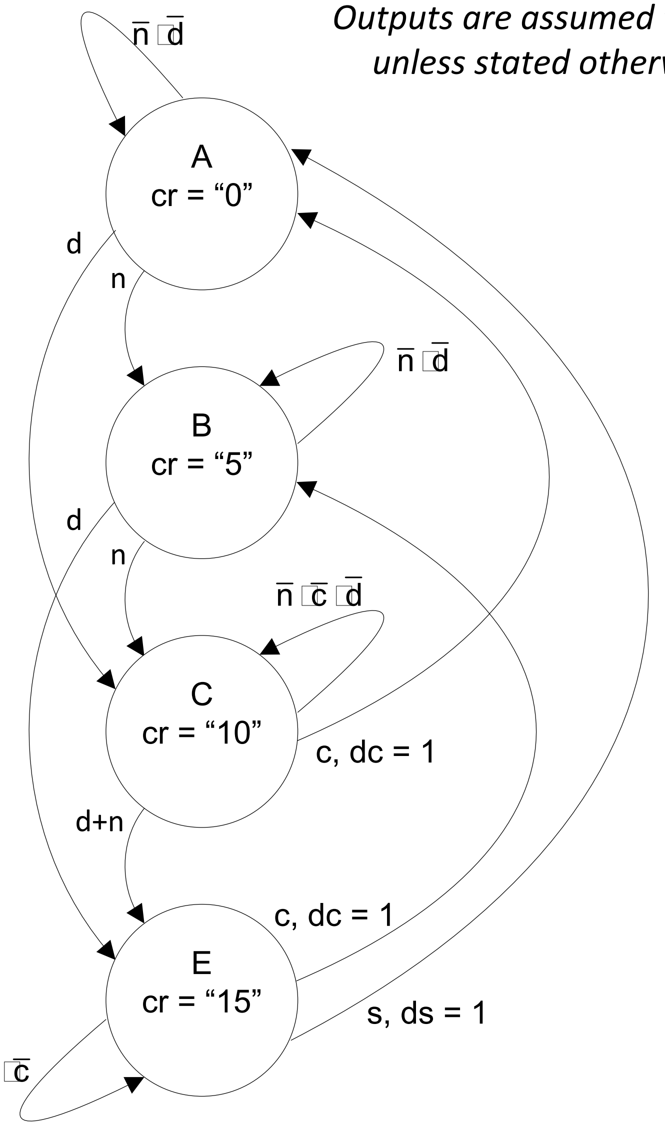
$$dc = \overline{Q}_2Q_1Q_0 + Q_2\overline{Q}_1Q_0$$

$$ds = Q_2Q_1\overline{Q}_0$$

$\ldots$

# 50s Vending Machine:  Mealy Implementation

*Outputs are assumed to be 0 unless stated otherwise*




*The Mealy implementation uses fewer states, and therefore fewer FFs!*

# State Assignments

- Back to our combinational lock example…

| S | XY = 00 | 01 | 10 | 11 | UNLOCK |
|---|---|---|---|---|---|
| A | A | B | A | A | 0 |
| B | A | B | C | A | 0 |
| C | A | D | A | A | 0 |
| D | A | B | C | A | 1 |
| | $S^+$ | | | | |

| S | $Q_1$ | $Q_0$ |
|---|---|---|
| A | 0 | 0 |
| B | 0 | 1 |
| C | 1 | 1 |
| D | 1 | 0 |

| $Q_1Q_0$ | XY = 00 | 01 | 10 | 11 | UNLOCK |
|---|---|---|---|---|---|
| 0 0 | 00 | 01 | 00 | 00 | 0 |
| 0 1 | 00 | 01 | 11 | 00 | 0 |
| 1 1 | 00 | 10 | 00 | 00 | 0 |
| 1 0 | 00 | 01 | 11 | 00 | 1 |
| | $Q_1^+ Q_0^+$ | | | | |

$Q1^+$

| $Q_1Q_0$ \ XY | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 0 | 0 | 1 |
| 11 | 0 | 1 | 0 | 0 |
| 10 | 0 | 0 | 0 | 1 |

$Q0^+$

| $Q_1Q_0$ \ XY | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 1 | 0 | 0 |
| 01 | 0 | 1 | 0 | 1 |
| 11 | 0 | 0 | 0 | 0 |
| 10 | 0 | 1 | 0 | 1 |

**Minimal SOP: 26 literals**

**Minimal POS: 20 literals**

*Perhaps we can do better using smarter state assignments…*

- Another state assignment approach
  - Maximize the number of 1's

| S | XY 00 | 01 | 10 | 11 | UNLOCK |
|---|---|---|---|---|---|
| A | A | B | A | A | 0 |
| B | A | B | C | A | 0 |
| C | A | D | A | A | 0 |
| D | A | B | C | A | 1 |
| | | $\overline{S^+}$ | | | |

| S | $Q_1$ | $Q_0$ |
|---|---|---|
| A | 1 | 1 |
| B | 0 | 1 |
| C | 1 | 0 |
| D | 0 | 0 |

| $Q_1Q_0$ | XY 00 | 01 | 10 | 11 | UNLOCK |
|---|---|---|---|---|---|
| 1 1 | 11 | 01 | 11 | 11 | 0 |
| 0 1 | 11 | 01 | 10 | 11 | 0 |
| 1 0 | 11 | 00 | 11 | 11 | 0 |
| 0 0 | 11 | 01 | 10 | 11 | 1 |
| | | $\overline{Q_1^+ Q_0^+}$ | | | |

$Q1^+$

| $Q_1Q_0$ \ XY | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 1 | 1 | 1 | 0 |
| 01 | 1 | 1 | 1 | 0 |
| 11 | 1 | 1 | 1 | 1 |
| 10 | 1 | 0 | 1 | 1 |

$Q0^+$

| $Q_1Q_0$ \ XY | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 1 | 0 | 1 | 1 |
| 01 | 1 | 0 | 1 | 1 |
| 11 | 1 | 0 | 1 | 1 |
| 10 | 1 | 0 | 1 | 1 |

**Minimal SOP: 10 literals**

**Minimal POS: 9 literals**

*Using smarter state assignments improved the next-state circuit cost from 20 literals to 9 literals!*

- Another approach: use more flip-flops
  - one-hot encodings (with the addition of 000)

| S | XY = 00 | 01 | 10 | 11 | UNLOCK |
|---|---|---|---|---|---|
| A | A | B | A | A | 0 |
| B | A | B | C | A | 0 |
| C | A | D | A | A | 0 |
| D | A | B | C | A | 1 |
| | $S^+$ | | | | |

| S | $Q_2$ | $Q_1$ | $Q_0$ |
|---|---|---|---|
| A | 0 | 0 | 0 |
| B | 0 | 0 | 1 |
| C | 0 | 1 | 0 |
| D | 1 | 0 | 0 |

| $Q_2Q_1Q_0$ | XY = 00 | 01 | 10 | 11 | UNLOCK |
|---|---|---|---|---|---|
| 0 0 0 | 000 | 001 | 000 | 000 | 0 |
| 0 0 1 | 000 | 001 | 010 | 000 | 0 |
| 0 1 0 | 000 | 100 | 000 | 000 | 0 |
| 1 0 0 | 000 | 001 | 010 | 000 | 1 |
| | $Q_2^+ Q_1^+ Q_0^+$ | | | | |

Read minterms directly off of transition table:

$$Q_0^+ = \overline{X}Y\left(\overline{Q}_2\overline{Q}_1\overline{Q}_0 + \overline{Q}_2\overline{Q}_1Q_0 + Q_2\overline{Q}_1\overline{Q}_0\right)$$
$$= \overline{X}Y\left(\overline{Q}_2\overline{Q}_1 + \overline{Q}_1\overline{Q}_0\right)$$
$$= \overline{X}Y\overline{Q}_2\overline{Q}_1 + \overline{X}Y\overline{Q}_1\overline{Q}_0$$

$$Q_1^+ = X\overline{Y}\overline{Q}_2\overline{Q}_1Q_0 + X\overline{Y}Q_2\overline{Q}_1\overline{Q}_0$$

$$Q_2^+ = \overline{X}Y\overline{Q}_2Q_1\overline{Q}_0$$

**23 literals**

How many states are *really* in our new state machine?

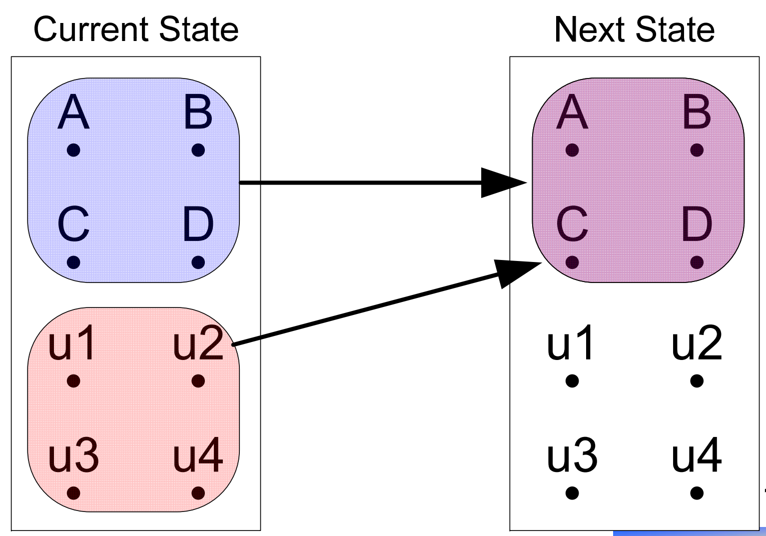
*What happened to the other 4 states???*

# Unused States

- Previous design: *all unused states were implicitly assigned a next state of 000 (state A)*
- This is known as a **safe design**
  - If noise causes the machine to enter an unused state, it will return to a used state under any input conditions

| | $Q_2Q_1Q_0$ | XY 00 | 01 | 10 | 11 | UNLOCK |
|---|---|---|---|---|---|---|
| | 0 0 0 | 000 | 001 | 000 | 000 | 0 |
| | 0 0 1 | 000 | 001 | 010 | 000 | 0 |
| | 0 1 0 | 000 | 100 | 000 | 000 | 0 |
| | 1 0 0 | 000 | 001 | 010 | 000 | 1 |
| u1 | 0 1 1 | 000 | 000 | 000 | 000 | 0 |
| u2 | 1 0 1 | 000 | 000 | 000 | 000 | 0 |
| u3 | 1 1 0 | 000 | 000 | 000 | 000 | 0 |
| u4 | 1 1 1 | 000 | 000 | 000 | 000 | 0 |

$Q_2^+ Q_1^+ Q_0^+$



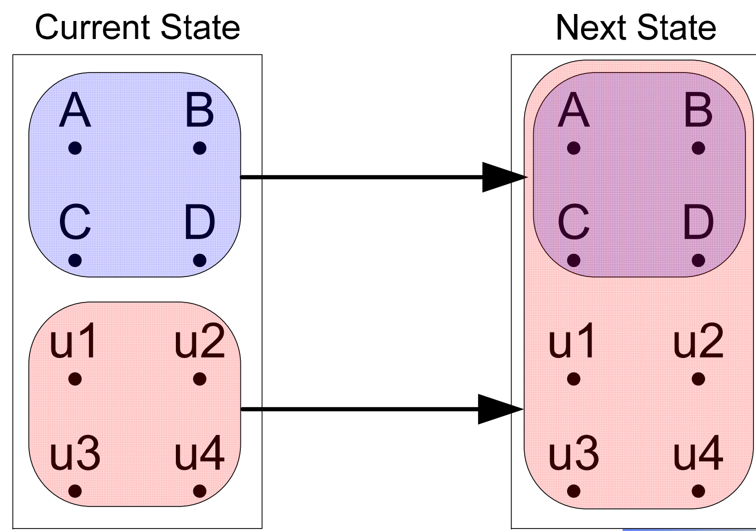

EECS270 -
Introduction to Logic Design

- **Efficient Design**: Treat the next-states and outputs of unused states as don't cares
  - Minimizes circuit cost!
- If an unused state is ever entered, state machine may never return to normal operation

| | $Q_2Q_1Q_0$ | XY = 00 | 01 | 10 | 11 | UNLOCK |
|---|---|---|---|---|---|---|
| | 0 0 0 | 000 | 001 | 000 | 000 | 0 |
| | 0 0 1 | 000 | 001 | 010 | 000 | 0 |
| | 0 1 0 | 000 | 100 | 000 | 000 | 0 |
| | 1 0 0 | 000 | 001 | 010 | 000 | 1 |
| u1 | 0 1 1 | ddd | ddd | ddd | ddd | d |
| u2 | 1 0 1 | ddd | ddd | ddd | ddd | d |
| u3 | 1 1 0 | ddd | ddd | ddd | ddd | d |
| u4 | 1 1 1 | ddd | ddd | ddd | ddd | d |

$Q_2^+ Q_1^+ Q_0^+$

*Finding transition equations now requires 5-variable K-maps!*



- **State clustering** assigns unused states to behave like used states
- If noise causes an unused state to be entered, the machine will return to a used state in a single clock cycle

| | X Y | | | | |
|---|---|---|---|---|---|
| $Q_2Q_1Q_0$ | 00 | 01 | 10 | 11 | UNLOCK |
| 0 0 0 | 000 | 001 | 000 | 000 | 0 |
| 0 0 1 | 000 | 001 | 010 | 000 | 0 |
| 0 1 x | 000 | 100 | 000 | 000 | 0 |
| 1 x x | 000 | 001 | 010 | 000 | 1 |
| | $Q_2^+ Q_1^+ Q_0^+$ | | | | |

Represents 010 (C) and 011 (u1)

Represents 100 (D), 101 (u2), 110 (u3), and 111 (u4)