# Outline

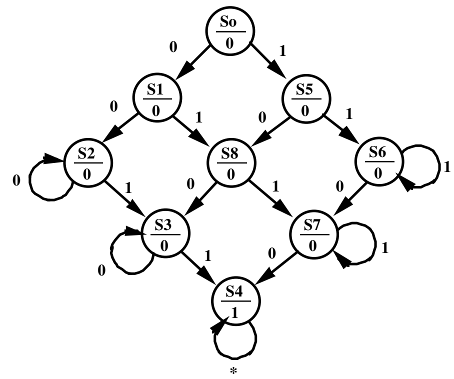
- Last time:
  - ➜ Review State Tables & State Transition Diagrams
  - ➜ Implementation Using D Flip-Flops
  - ➜ Machine Equivalence
  - ➜ Incompletely Specified Machines
  - ➜ State Assignment & State Coding Schemes
  - ➜ Design Example: Assign Codes to States
  - ➜ Design Example: Implement Using D flip-flops
  - ➜ Design Example: Implement Using T flip-flops
- This lecture:
  - ➜ Sequential Examples
  - ➜ Arithmetic Examples



# Example 1: Sequential Design

**"(1) A sequential network has one input and one output. The output becomes 1 and remains 1 thereafter when at least two zeros and at least two ones have occurred as inputs, regardless of the order of occurrence. Draw a state graph (Moore type) for the network (9 states are sufficient). Your final state graph should be neatly drawn with no crossed lines."**

## Example 1: Sequential Design






## Example 2: State Assignment & J/K Implementation

"(2) The following state table is to be implemented using J-K flip-flops and logic gates (format of next-state entries is (next-state,output))."

| Present input input State | x=0 | x=1 |
|---|---|---|
| a | a,0 | e,0 |
| b | c,0 | b,1 |
| c | a,0 | f,0 |
| d | c,0 | b,1 |
| e | f,0 | e,0 |
| f | a,0 | f,0 |

## Example 2: State Assignment & J/K Implementation

**"(a) Find a good state assignments using the three guidelines mentioned in class. (Do not reduce the table first).**
**Try to satisfy as many of the adjacency conditions as possible."**

**Adjacency conditions**:
**Rule 1:** "States which have the same next-state for a given input should be given adjacent assignments."
{f, c}, {d, b},
**Rule 2:** "States which are the next-states of the same states should be given adjacent assignments."
2*{a, f}, 2*{c, b}, {e, f}, {a, e}
**Output:** "States which have the same output for a given input should be given adjacent assignments."
Not of any value in this case; one input & one output almost everything would need to be adjacent!
Lowest priority anyway.



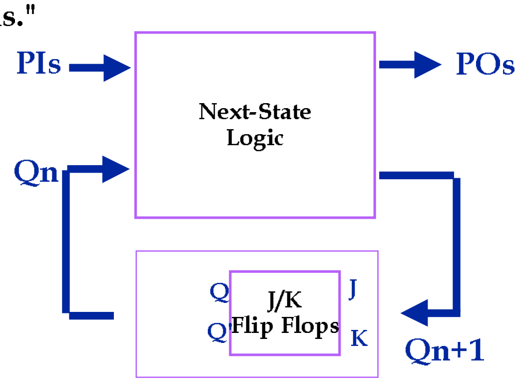
## Example 2: State Assignment & J/K Implementation

- **The following assignment satisfies all of the adjacency conditions except {e,f}:**

| Q3 \ Q1Q2 | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | A 0 | F 2 | C 6 | E 4 |
| 1 | 1 | 3 | B 7 | D 5 |

## Example 2: State Assignment & J/K Implementation

"(b) Using this assignment, derive the.J-K flip-flop input equations and output equations. Express them in a form that contains the minimum number of literals."






## Guidelines for Determining Flip-Flop Input Equations from Next-State Map

| | | $Q_n = 0$ | | $Q_n = 1$ | | Rules for forming input map from next-state map (2) | |
|---|---|---|---|---|---|---|---|
| Type | Input | $Q_{n+1}=0$ | $Q_{n+1}=1$ | $Q_{n+1}=0$ | $Q_{n+1}=1$ | $Q_n=0$ half | $Q_n=1$ half |
| D | D | 0 | 1 | 0 | 1 | no change | no change |
| T | EN | 0 | 1 | 1 | 0 | no change | complement |
| S-R | S | 0 | 1 | 0 | * | no change | replace 1s with *s |
| | R | * | 0 | 1 | 0 | replace 0s with *s | complement |
| J-K | J | 0 | 1 | * | * | no change | fill in with *s |
| | K | * | * | 1 | 0 | fill in with *s | complement |

- Notes:

(1) * = "don't care"
(2) Always copy *s from next-state map to input map first
(3) For S, Qn=1 half and R, Qn=0 half, fill remaining entries with 0s.

# Flip-Flop Input Equations From Next-State Map: Example

Qn+1 next-state map

| Qn \ AB | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 $_{0}$ | 1 $_{2}$ | 0 $_{6}$ | 1 $_{4}$ |
| 1 | 1 $_{1}$ | 0 $_{3}$ | 0 $_{7}$ | * $_{5}$ |

D input map

| Qn \ AB | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 $_{0}$ | 1 $_{2}$ | 0 $_{6}$ | 1 $_{4}$ |
| 1 | 1 $_{1}$ | 0 $_{3}$ | 0 $_{7}$ | * $_{5}$ |

T input map

| Qn \ AB | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 $_{0}$ | 1 $_{2}$ | 0 $_{6}$ | 1 $_{4}$ |
| 1 | 0 $_{1}$ | 1 $_{3}$ | 1 $_{7}$ | * $_{5}$ |

S

| Qn \ AB | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 $_{0}$ | 1 $_{2}$ | 0 $_{6}$ | 1 $_{4}$ |
| 1 | * $_{1}$ | 0 $_{3}$ | 0 $_{7}$ | * $_{5}$ |

J

| Qn \ AB | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 $_{0}$ | 1 $_{2}$ | 0 $_{6}$ | 1 $_{4}$ |
| 1 | * $_{1}$ | * $_{3}$ | * $_{7}$ | * $_{5}$ |

R

| Qn \ AB | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | * $_{0}$ | 0 $_{2}$ | * $_{6}$ | 0 $_{4}$ |
| 1 | 0 $_{1}$ | 1 $_{3}$ | 1 $_{7}$ | * $_{5}$ |

K

| Qn \ AB | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | * $_{0}$ | * $_{2}$ | * $_{6}$ | * $_{4}$ |
| 1 | 0 $_{1}$ | 1 $_{3}$ | 1 $_{7}$ | * $_{5}$ |

S-R input map

J-K input map



# Example 2: State Assignment & J/K Implementation

- J1 = XQ2', K1 = Q2Q3' + XQ3'
- J2 = X'Q1 + Q3, K2 = X'Q3'
- J3 = 0, K3 = X'
- Z = XQ3

# Example 3: Arithmetic

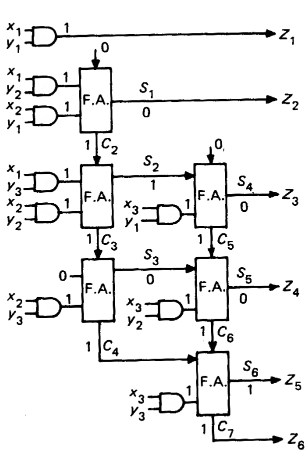
"(3) Design a parallel binary multiplier which multiplies two 3-bit binary numbers to for a 6-bit product. This multiplier is to be a combinational network consisting of an array of 1-bit full adders and AND gates only (no flip-flops).

(a) Show a schematic diagram.(Hint: The AND gates can be used to multiply by 0 or 1 and the full adders can be used to add 2 bits plus a carry. Six full adders are required.)

(b) Demonstrate you multiplier works by showing the values on all internal outputs (outputs of adders and outputs of AND gates) when multiplying 111 by 111. "



# Example 3: Arithmetic



# Example 3: Arithmetic

"(c) Using exactly 5 bits, express the following numbers in 2's-complement form:

(a) 12 = __ __ __ __ __
(b) -13 = __ __ __ __ __
(c) 31 = __ __ __ __ __"