# Penn State Abington
# CMPEN 271
# Lecture Set #19
# State Machines with JK FFs

R. Avanzato ©2014-2015

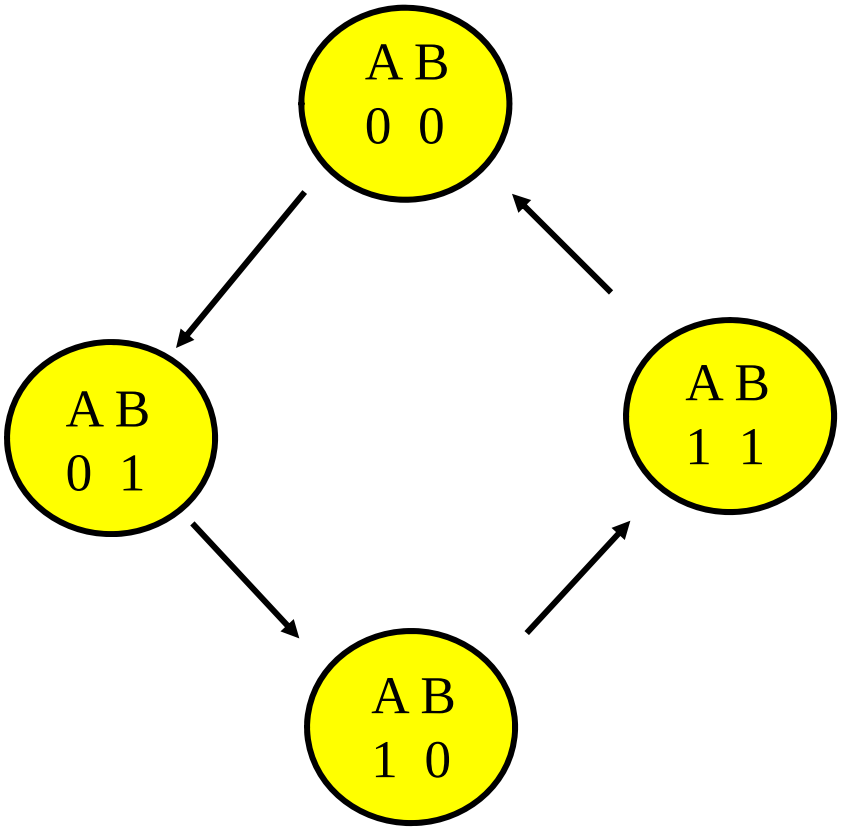
**Topics:**

- Implementing a state machine with JK flip-flops — **Video part 1 of 3 ←**
- Design Examples

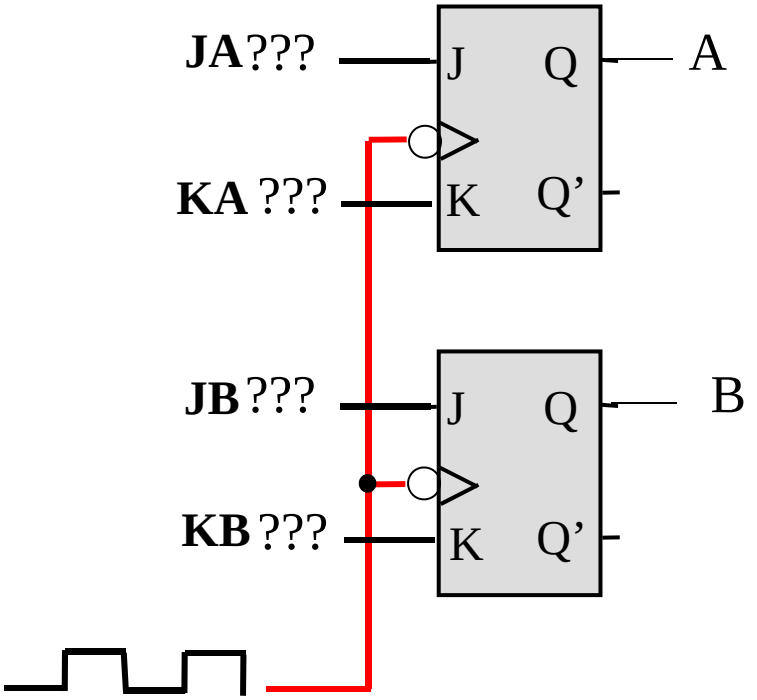
- FSM Design (Sequence detector) — Video part 2 of 3
- Mealy versus Moore machines
- FSM Summary

- Review Questions — Video part 3 of 3

# Implementing State Machines with JK Flip-flops



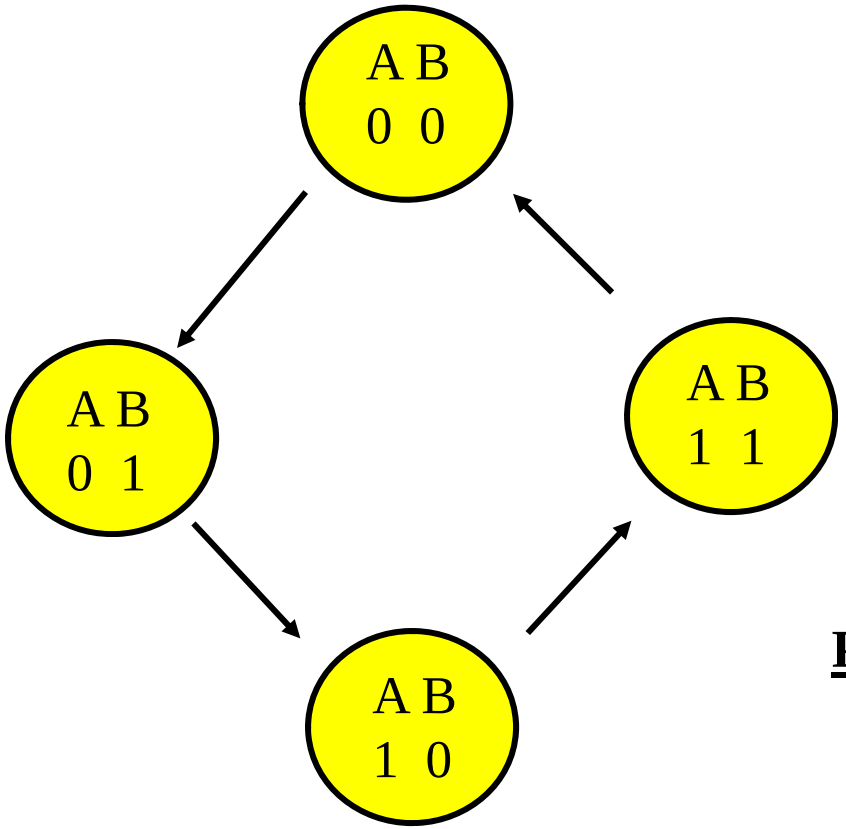

State Diagram Example




Observations:

1) FFs synchronized to clock.
2) A and B are outputs of FFs.
3) data inputs to JK FFs must be found
4) Our goal is to find circuitry for each ???

# Design Example #1



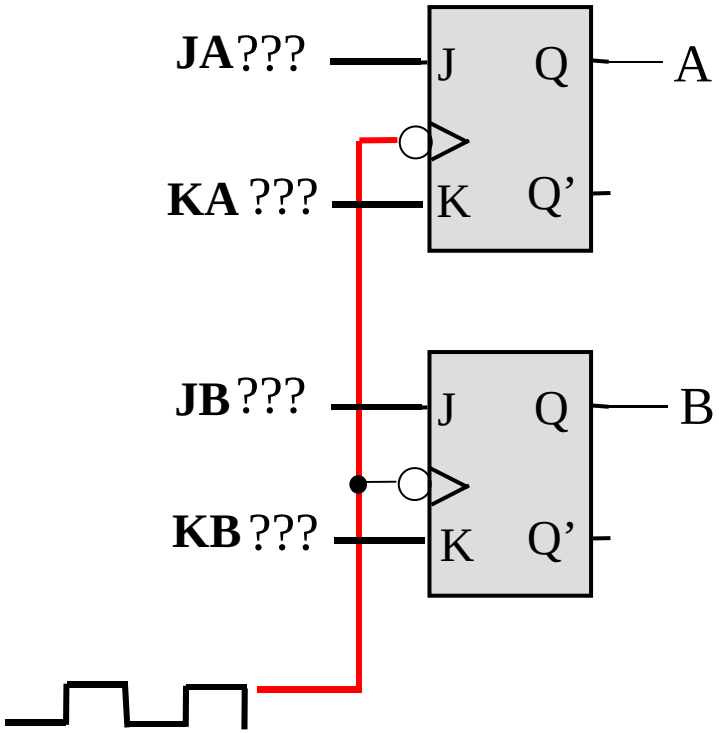

**State Diagram**

Problem: Design circuit with above state diagram

**Step#1** - Construct state diagram based on given problem.
**Step#2** - Construct state table based on state diagram.
**Step#3** - Decide on D ffs or JK FFs. We pick JKs.
**Step#4** - Identify FF input equations (two FF input functions per JK FF) using excitation table.
**Step#5** - Minimize FF input equations (or use DEC, MUX, etc). Input eq's are Boolean fcns.
**Step#6** - Draw circuit. Include clock.

| Present State | | Next State | | FF input eq's | | | |
|---|---|---|---|---|---|---|---|
| **A** | **B** | **A** | **B** | **JA** | **KA** | **JB** | **KB** |
| 0 | 0 | 0 | 1 | ? | ? | ? | ? |
| 0 | 1 | 1 | 0 | ? | ? | ? | ? |
| 1 | 0 | 1 | 1 | ? | ? | ? | ? |
| 1 | 1 | 0 | 0 | ? | ? | ? | ? |

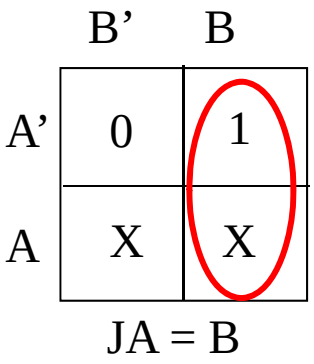
# Design Example #1



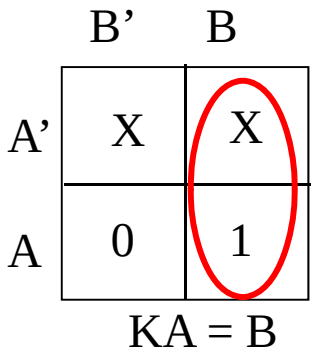

| Present State | | Next State | | FF input eq's | | | |
|---|---|---|---|---|---|---|---|
| A | B | A | B | JA | KA | JB | KB |
| 0 | 0 | 0 | 1 | 0 | X | 1 | X |
| 0 | 1 | 1 | 0 | 1 | X | X | 1 |
| 1 | 0 | 1 | 1 | X | 0 | 1 | X |
| 1 | 1 | 0 | 0 | X | 1 | X | 1 |

JA = B  JB = 1
KA = B  KB = 1

Solve (K-map or inspection) for each function

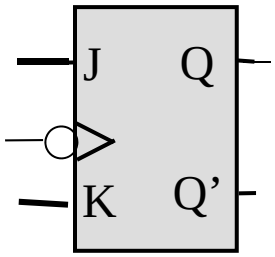



JA = B




KA = B

# Design Example #1




JK FF Excitation Table:

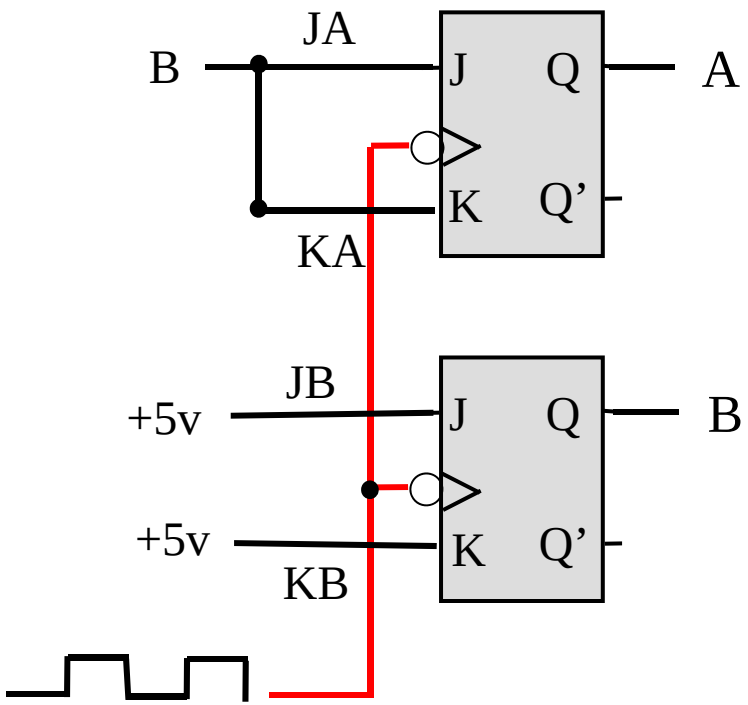
| **PS** | **NS** | **J** | **K** | | J | K | | J | K |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | X | (means | 0 | 0 | or | 0 | 1) |
| 0 | 1 | 1 | X | (means | 1 | 0 | or | 1 | 1) |
| 1 | 0 | X | 1 | (means | 0 | 1 | or | 1 | 1) |
| 1 | 1 | X | 0 | (means | 0 | 0 | or | 1 | 0) |

where: PS = present state (before clock pulse)
NS = next state (after clock pulse)
X = don't care (either 1 or 0)

# Design Example #1

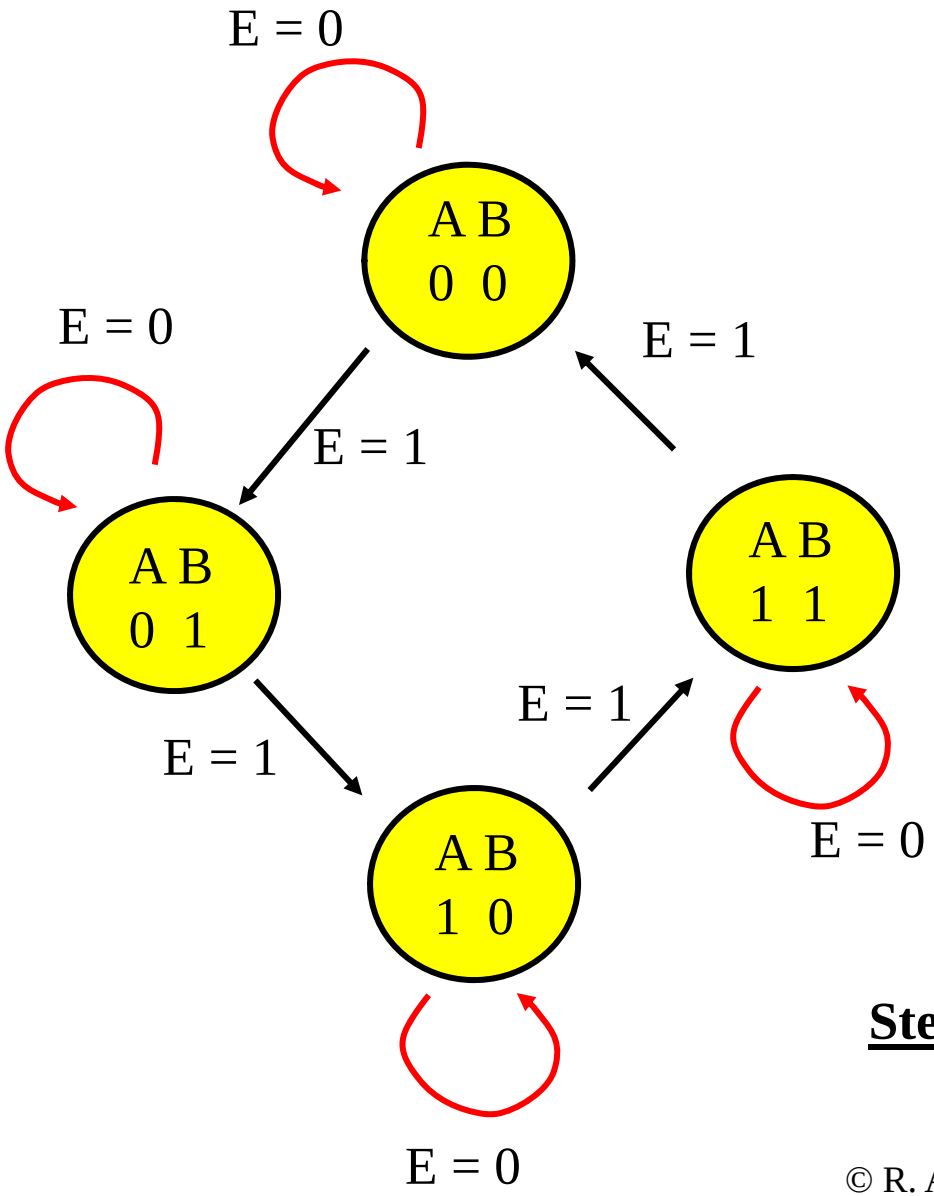
FF Input Equations:

JA = B  JB = 1
KA = B  KB = 1




Note: Compare JK FF solution to D FF solution of same problem. JK FFs generally lead to less gates because JK flip flops allow for more don't cares.

# Design Example #2




**Design Problem**: Design and implement a finite state machine (FSM) that normally counts in the sequence 00, 01, 10, 11, and repeats. There is a single external input, E. If E = 1, then the counter counts in the normal sequence. If E=0, then the counters "stays" in the present state. (Use JK flip-flops).

How many FFs are required?

**Step#1** - construct state diagram and explain

# Design Example #2

**Step #2** - construct state table.
**Step #3** - decide on FF type - use JK FFs (given in problem)
**Step #4** - fill-out values for 4 needed FF input equations (exercise)

| P.S. | | input | Next State | | FF input eq's | | | |
|---|---|---|---|---|---|---|---|---|
| A | B | E | A | B | JA | KA | JB | KB |
| 0 | 0 | 0 | 0 | 0 | | | | |
| 0 | 0 | 1 | 0 | 1 | | | | |
| 0 | 1 | 0 | 0 | 1 | | | | |
| 0 | 1 | 1 | 1 | 0 | | | | |
| 1 | 0 | 0 | 1 | 0 | | | | |
| 1 | 0 | 1 | 1 | 1 | | | | |
| 1 | 1 | 0 | 1 | 1 | | | | |
| 1 | 1 | 1 | 0 | 0 | | | | |

**Excitation Table**

| PS | NS | J | K |
|---|---|---|---|
| 0 | 0 | 0 | X |
| 0 | 1 | 1 | X |
| 1 | 0 | X | 1 |
| 1 | 1 | X | 0 |

**Exercise** -- fill in values for functions JA, KA, JB, KB (Hint: use excitation table for help.)

# Design Example #2

**Step#5** - Minimize each FF input equations (Boolean functions)

JA

| | B'E' | B'E | BE | BE' |
|---|---|---|---|---|
| A' | | | 1 | |
| A | X | X | X | X |

JA = BE

KA

| | B'E' | B'E | BE | BE' |
|---|---|---|---|---|
| A' | X | X | X | X |
| A | | | 1 | |

KA = BE

JB

| | B'E' | B'E | BE | BE' |
|---|---|---|---|---|
| A' | | 1 | X | X |
| A | | 1 | X | X |

JB = E

KB

| | B'E' | B'E | BE | BE' |
|---|---|---|---|---|
| A' | X | X | 1 | |
| A | X | X | 1 | |

KB = E

# Design Example #2

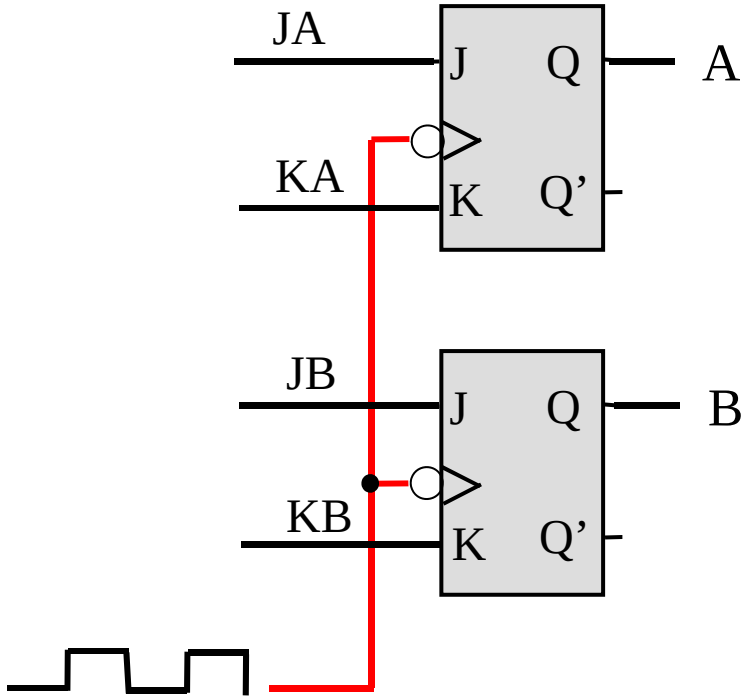
**Step #6** - Draw circuit

E

From
K-maps

JA = BE
KA = BE

JB = E
KB = E




Question: Could MUXs or Decoders be used to implement the input equations? What are the pros and cons?

# Penn State Abington
# CMPEN 271
# Lecture Set #19
# State Machines with JK FFs

R. Avanzato ©2014-2015

**Topics:**

- Implementing a state machine with JK flip-flops — Video part 1 of 3
- Design Examples
- FSM Design (Sequence detector) — **Video part 2 of 3 ←**
- Mealy versus Moore machines
- FSM Summary
- Review Questions — Video part 3 of 3

# State Machine Design Example - 1

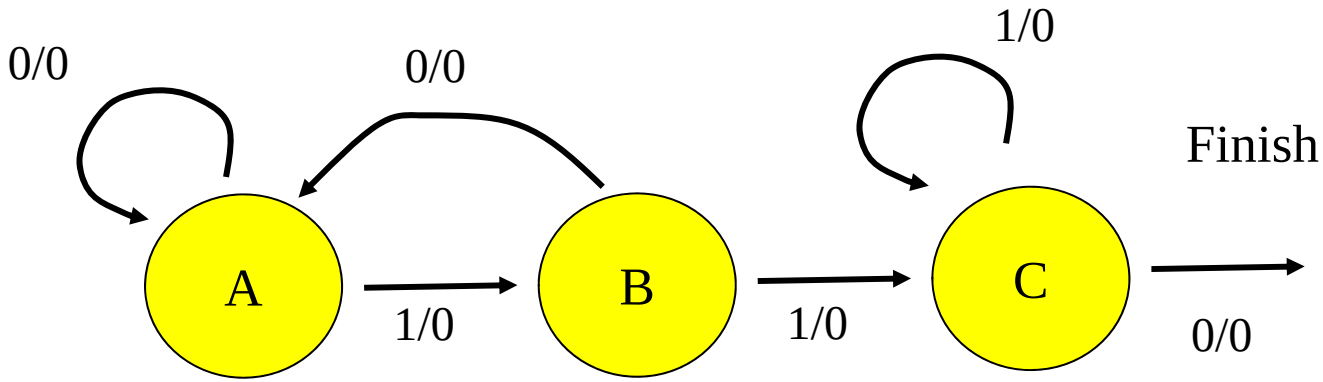
Design a state machine (also called a sequential circuit) to detect a particular **binary sequence** in a stream of input bits. **The sequence to be detected is 1101.** When a complete sequence is detected, the state machine should produce an output of 1, otherwise the output will be 0. The bit stream may contain more than one sequence to be detected. For example if the (arbitrary) input bit stream is

**input**: 1010011010011010000

then output of the state machine would be

**output**: 0000000**1**0000**1**0000

Notation:
input / output




Finish state diagram...

How many states needed? How many FF's? How do you assign states A, B, C, …?
Discuss **state assignment** problem.

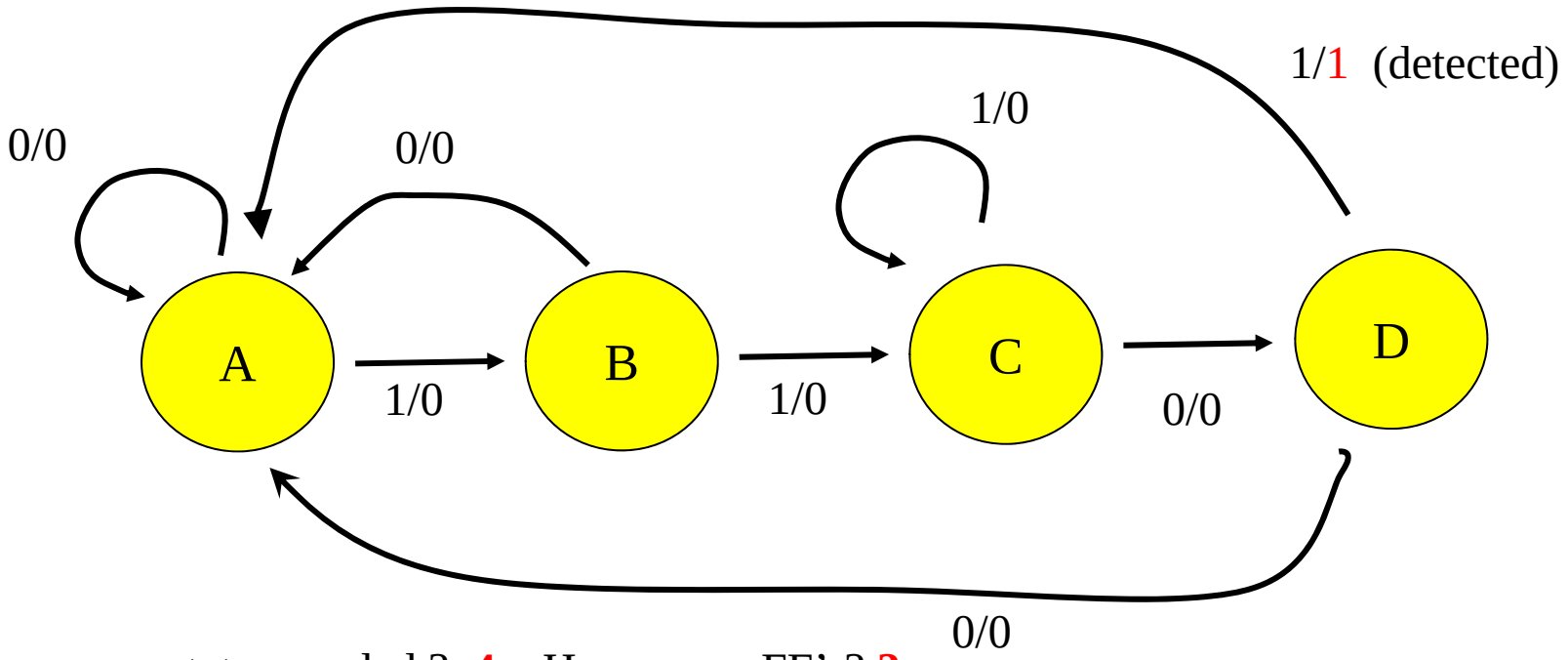
# State Machine Design Example - 1

input: 0 0 1 0 1 0 0 1 1 0 1 0 0 1 1 0 1 0 0 0 0

then output of the state machine would be

output: 0 0 0 0 0 0 0 0 0 0 **1** 0 0 0 0 0 **1** 0 0 0 0

Notation:
input / output



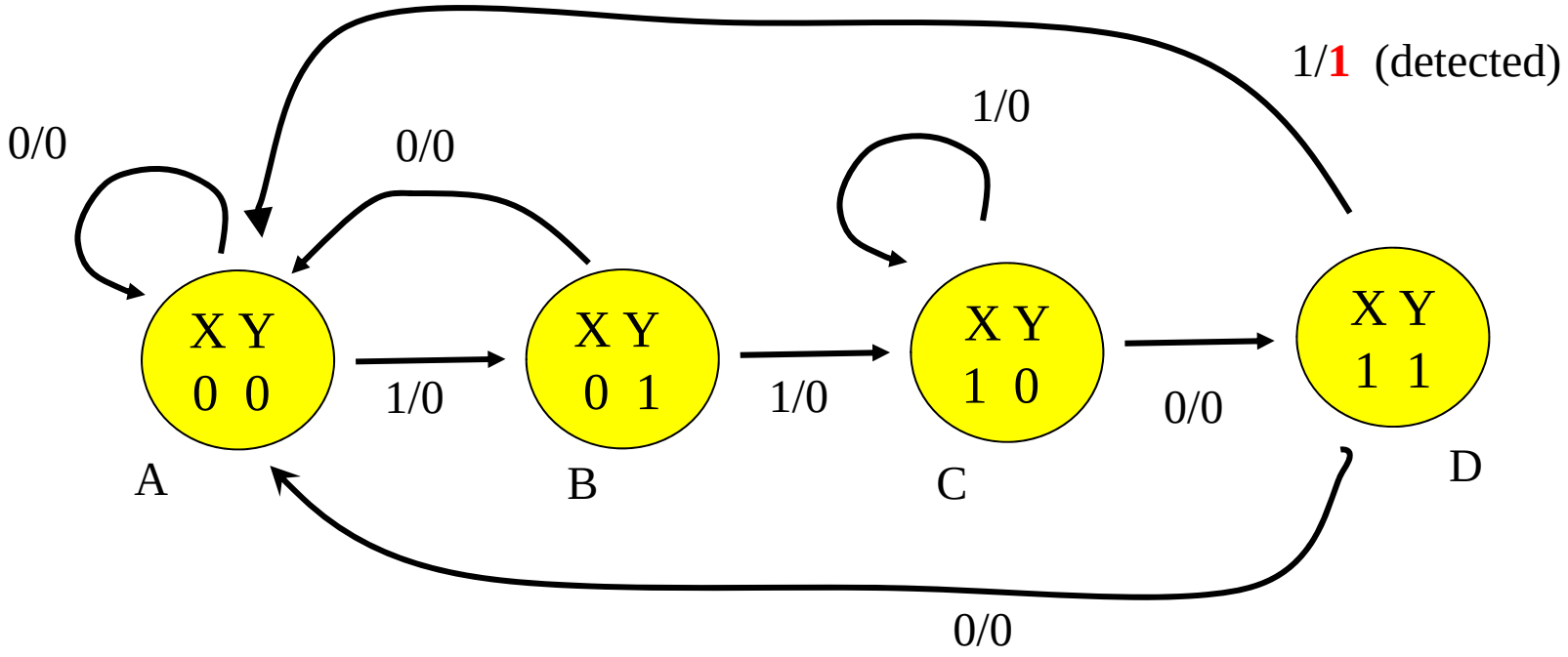

How many states needed ? **4** How many FF's? **2**

How do you assign states A, B, C, D? **We can try 00, 01, 10, 11 or other. The state assignment won't change operation but might be more or less efficient (number of gates)**

# State Machine Design Example - 1

- Pick **state assignment and redraw state diagram**
  - Let A (00), B(01), C(10), D(11)
  - How many flip flops are required?
  - Is this the most efficient state assignment?

Notation:
input(in) / output(out)




How many states needed ? **4** How many FF's? **2**

**There is one external input (in) and one external output (out) in this FSM**

# State Machine Design Example - 1

**Step #2** - construct state table for sequence detector "1101".
**Step #3** - decide on FF type - use JK FFs (given in this problem)
**Step #4** - fill-out values for 4 needed FF input equations (exercise)

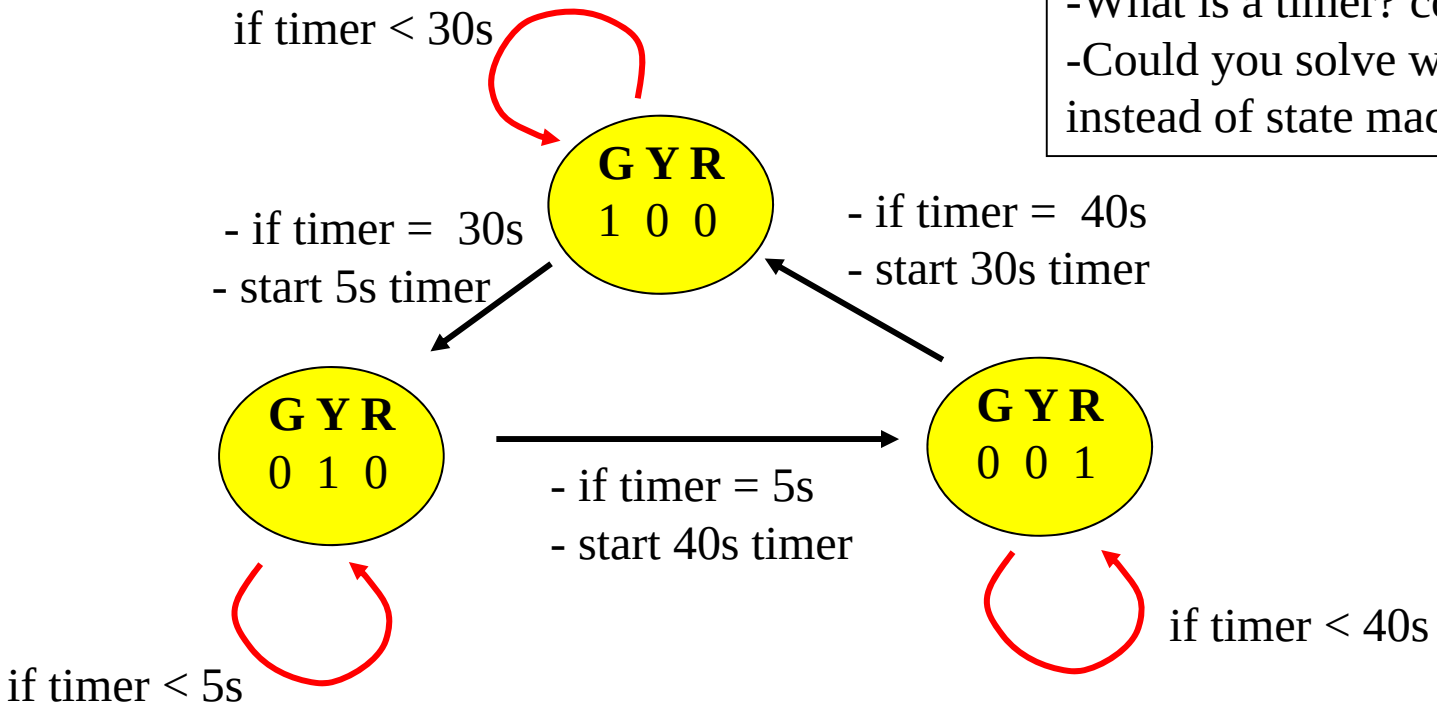
| P.S. | | input | Next State | | | FF input eq's | | | |
|---|---|---|---|---|---|---|---|---|---|
| X | Y | In | X | Y | Out | JX | KX | JY | KY |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | X | 0 | X |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | X | 1 | X |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | X | X | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 | X | X | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 | X | 0 | 1 | X |
| 1 | 0 | 1 | 1 | 0 | 0 | X | 0 | 1 | X |
| 1 | 1 | 0 | 0 | 0 | 0 | X | 1 | X | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 | X | 1 | X | 1 |

**Exercise** -- Find 5 minimized Boolean Functions for JX, KX, JY, KY, and Out, and construct circuit using 2, JK FFs.

# State Machine Design Example - 2

Design a state machine to control a **traffic light**. The traffic light system consists of three separate lights – green(G), yellow(Y), and red(R). The green light should be on for 30 seconds, then the yellow light should be on for 5 seconds, then the red light should be on for 40 seconds, then the process is repeated. Assume a 1Hz clock.

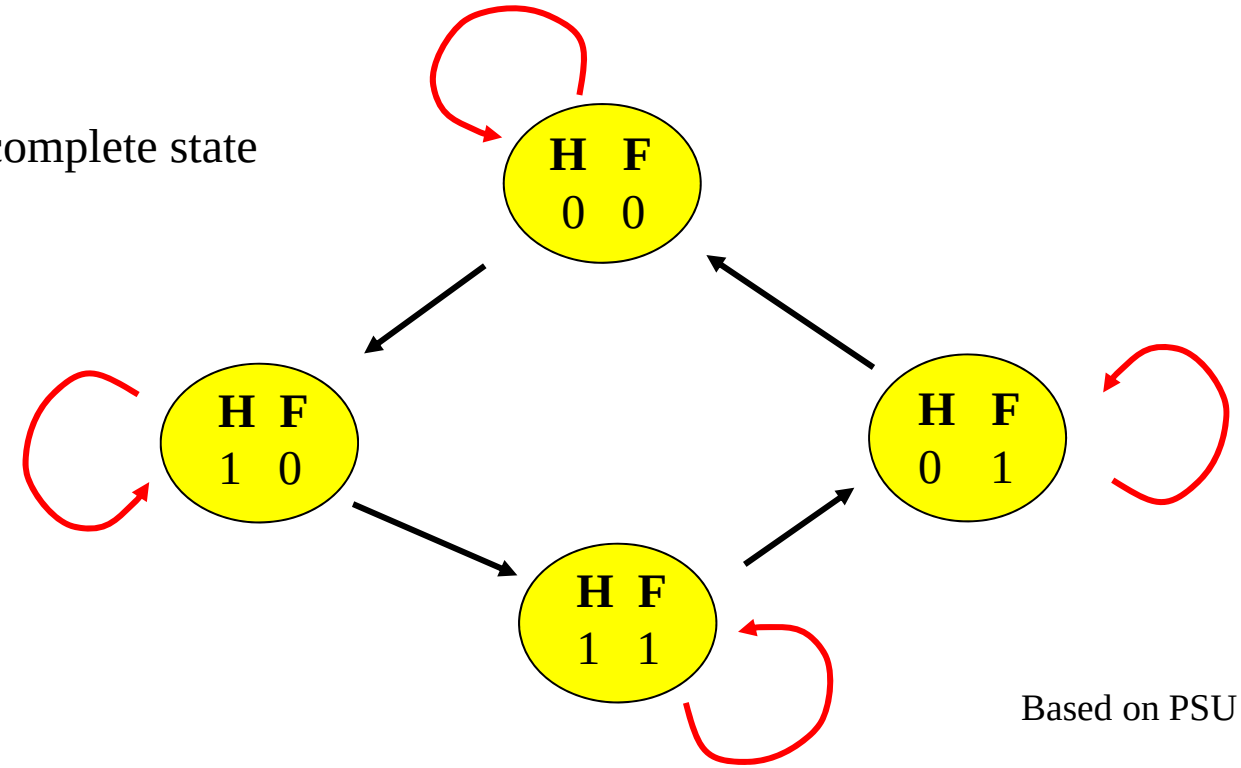
-How many flip-flops needed?
(one-hot encoding)
-What is a timer? counter?
-Could you solve with ring counter instead of state machine?



# State Machine Design Exercise - 3

Design a state machine to control a **home heating system**. The heating system consists of three components: the thermostat, the heater, and the fan. If the thermostat senses the room temperature below 70 degrees F, then the heater should turn on, and 5 seconds later the fan should turn on. The heater and fan stay on until the temperature rises to >= 70 degrees. When the temperature >= 70 degrees, then the heater should turn off, and 10 seconds later the fan should turn off.

Exercise: complete state diagram




Based on PSU CSE 275 Lab notes

# Mealy vs. Moore Machines

Useful definitions to classify State Machines:

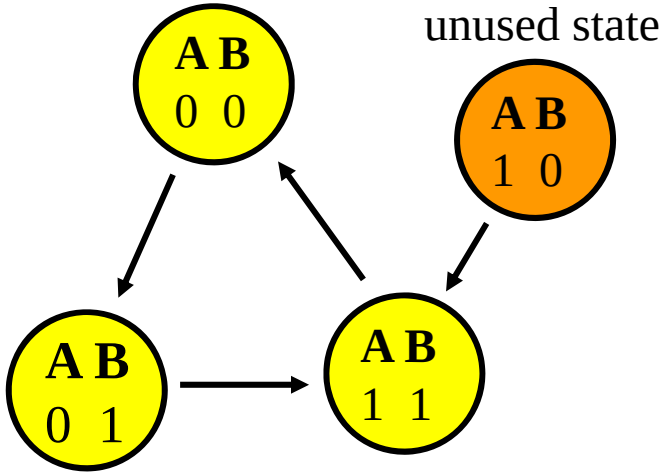
**Mealy Machine** = state machine in which the external outputs depend on the external inputs as well as on the states (outputs) of the flip-flops.

**Moore Machine** = state machine in which the external outputs depend only on the states (outputs) of the flip-flops.

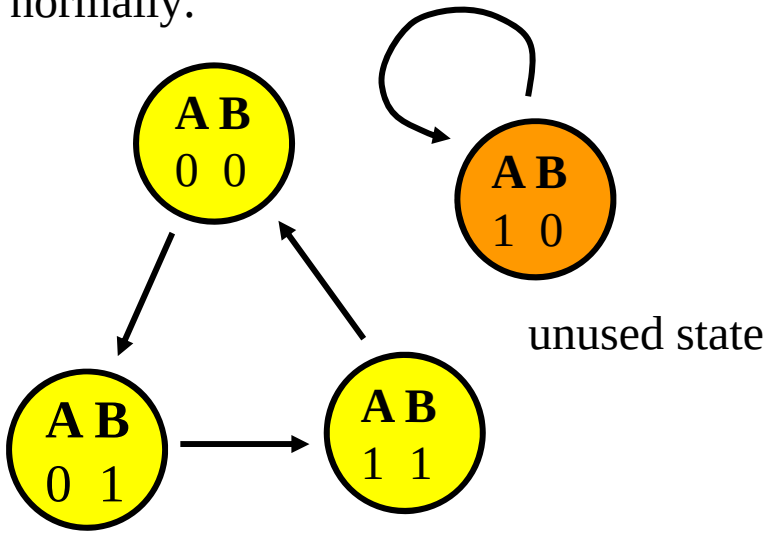
**Exercise**: classify the previous state machine examples as either Mealy or Moore machines. Give examples of both types of machines.

# Self-correcting State Machine

- Consider a state machine with one or more **unused** states. For example, suppose we had a 2-bit state machine with sequence 00, 01, 11, repeat. The state 10 is unused. What would happen if the state machine flip-flops were inadvertently placed in this unused state? This could happen at startup or by way of a glitch.
- A **self-correcting** state machine is designed so that in the event the flip-flops "pop" into any unused state, the state machine will eventually return to any one of the "used" states and then perform normally.




self-correcting
(returns to valid states)




not self-correcting
(never returns to valid states)

# State Machines Summary -1

- The **first step** in the design of a state machine is to draw the **state diagram** and then the **state table**. After this, the next step is to determine the **flip-flop input equations.**

- State machines can be **implemented with either D FFs or JK FFs**. (There are also other types flip-flops, but the operation is equivalent.) Edge-triggered D FFs and JK FFs are the most common.

- Is a state machine, the **clock inputs** for all of the FFs (whether D or JK) should be connected to the master clock.

- The goal in designing the state machine circuit is to find the **input equations** for each flip-flop

- It is often **easier and quicker to design a FSM with D flip-flops** (1 input equation per D flip-flop) then using JK flip-flops (2 input equations per JK flip-flop), **but** there is generally more simplification (through the use of don't cares) with the JK flip-flops, which results in fewer gates overall, and this can be an advantage of using JK flip-flops. Both approaches yield the same FSM operation. There are modern software tools that can perform some or all of the circuit synthesis.

# State Machines Summary -2

- The **input equations** for any state machine (using either D or JK flip-flops) can be implemented using a **decoder or a multiplexer**. This can save circuitry. Also, there would be no need to perform a K-map if you use a decoder or a multiplexer. Remember, input equations are simply combinatorial Boolean functions.

- A **Mealy machine** is a FSM machine in which the external outputs of the FSM depend on the outputs of the FFs and also the inputs to the FSM. Examples of inputs might be "clear" or "enable" or "direction", etc. A **Moore machine** is an FSM in which the external outputs depend only on the outputs of the flip-flops. These terms are simply a way to categorize FSMs. (A Mealy model may lead to a reduction in states as compared to a Moore model, but this is left as a research project to explore.)

- Many **control problems** (such as the heater-fan controller, elevator controller) can be solved using finite state machines (FSMs).

- There are often **unused states** in a state machine design. This depends on the particular problem you are solving. Unused states can be used as **"don't cares"** when solving for the flip-flop input equations and can reduce the amount of circuitry required.

# State Machines Summary - 3

• Modern **programmable logic devices** (PLD, CPLD, FPGA) generally support state machines with D FFs. Special design software will automatically generate and optimize the FF input equations and any combinational circuitry. (See www.xilinx.com and www.altera.com). Software languages supported include ABEL, VHDL, and VeriLog.

•The **most challenging aspect** of state machine design is gaining a full understanding of the problem, and generating an accurate state diagram and state table. Once this is accomplished software tools can be used to "turn the crank" and generate the circuit. As you might expect, no s/w tool is fully automatic, and designer often needs to understand the hardware.

•"**State assignment**" and "**state minimization/reduction**" techniques are often faced by state machines designers. There are modern software tools which can provide some assistance with these techniques. See Mano, Kime or Wakerly texts for more information.

•FSMs are **fundamental building blocks** in the design of microprocessors and FPGAs.

•Beyond digital electronics, there are many technical fields which use the **idea of state machines** to solve problems. Areas include software design, networking communications, robotics. grammar, natural language recognition, and artificial intelligence.

# State Machine Projects

#1) Design and simulate a state machine circuit for a **binary sequence detector**. The sequence to be detected is 1011. Use D or JK flip-flops.

#2) Design and simulate a state machine circuit for the **heater fan problem**. Use a one-shot device or a counter to provide the timing delay. Use D or JK flip-flops. See instructor for details.

#3) Design and simulate a state machine circuit for **traffic light problem**. Use a one-shot device or a counter to provide the timing delay. Use D or JK flip-flops. See instructor for details.

**NOTE**: We will discuss each one of these problems in class. Students may optionally complete the circuit diagrams and simulate the circuits with proper documentation Students may also elect to construct circuits on a breadboard for demonstration. See instructor for details.

# Further Reading

- Mano, Kime, Logic and Computer Design Fundamentals, 2nd edition, Prentice Hall, Chapter 4.

- Wakerly, John, Digital Design: Principles and Practice, 3rd edition, Prentice Hall, Chapter 7.

- Wikipedia.com "Finite State Machine"  (Note: also includes references to software tools.)

- **Review Sample Final Exam Questions with Solutions** on Angel (includes D FF, JK FF, shift registers, ring counters, state machines)

# Penn State Abington
# CMPEN 271
# Lecture Set #19
# State Machines with JK FFs

R. Avanzato ©2014-2015

**Topics:**

- Implementing a state machine with JK flip-flops — Video part 1 of 3
- Design Examples
- FSM Design (Sequence detector) — Video part 2 of 3
- Mealy versus Moore machines
- FSM Summary
- Review Questions — **Video part 3 of 3 ←**

# Review Questions

#1.- Does the state diagram and state table contain any information about the type of FF to be used in the implementation of the FSM?

a) Yes b) No

#2.- Does the state diagram and state table contain any information about the clock speed or how fast the circuit will operate?

a) Yes b) No

#3.- Do the FF input equations depend on the type of FF to be used in the implementation of the FSM?

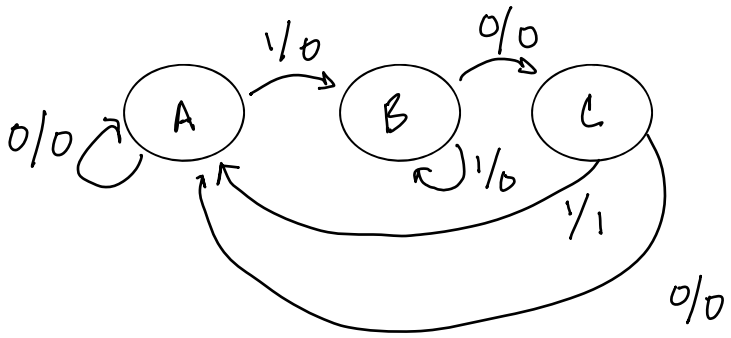
a) Yes b) No

#4- If a state diagram is composed of 5 states (5 bubbles), how many flip flops are minimally required to implement the FSM?

a) 2 b) 3 c) 4 d) 5

# Review Questions

Consider the following state diagram for a binary sequence detector.




See sample final exam questions on Angel

**#5. In the state machine form above, the notation "x/y" denotes**

a) x is an input and y is an input
b) x is an input and y is an output
c) x is an output and y is an output
d) x is divided by y

**#6. What does the symbol "0/1" mean in a state diagram?**

a) if the input = 1 then set the output = 0 and make the state transition
b) if the input = 0 then set the output = 1 and make the state transition
c) if the input = 0 then set the output = 0 and make the state transition

**#7. What binary sequence is detected in the FSM above? That is, starting at state A, what sequence of bits will produce an output of 1?**

a) 111 b) 101 c) 1011 d) 0100 e) 100

# Review Questions

Consider the following state table for a FSM using JK flip-flops

| Present State | | Next State | | FF input eq's | | | |
|---|---|---|---|---|---|---|---|
| A | B | A | B | JA | KA | JB | KB |
| 0 | 0 | 0 | 1 | 0 | X | 1 | X |
| 0 | 1 | 1 | 0 | 1 | X | X | 1 |
| 1 | 0 | 1 | 1 | X | 0 | 1 | X |
| 1 | 1 | 0 | 0 | X | 1 | X | 1 |

#8.- How many flip-flops are represented in above state diagram?

a) 5 b) 4 c) 3 d) 2

#9.- What are the flip-flop input equations for flip-flop A?

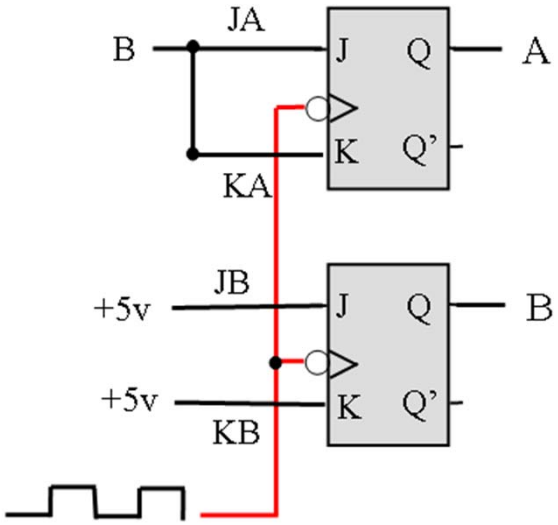
a) JA=B, KA=B b) JA =AB, KA=B c) JA=0, KA=B d) JA=1, KA=1

#10.- What are the flip-flop input equations for flip flop B?

a) JB=A, KB=B b) JB =1, KB=1 c) JB=0, KB=B d) JB=A, KB=1

# Review Questions

#11.- Consider the state machine below designed with JK flip-flops?




#12.- What is the flip-flop input equation for flip-flop A?

a) JA=AB, KA=B b) JA =0, KA=B c) JA=B, KA=B d) JA=1, KA=1

#13.- What is the flip-flop input equation for flip flop B?

a) JB=A, KB=B b) JB =1, KB=1 c) JB=0, KB=B d) JB=A, KB=1