Penn State Abington
CMPEN 271
Lecture Set #18
State Machines with D FFs

R. Avanzato © 2014-2015

Topics:

- Design Project: Up/Down Counter — **Video part 1 of 5 ←**
- Sequential versus combinational Logic
- Definition of a finite state machine (FSM)
- Implementing a state diagram with D flip-flops

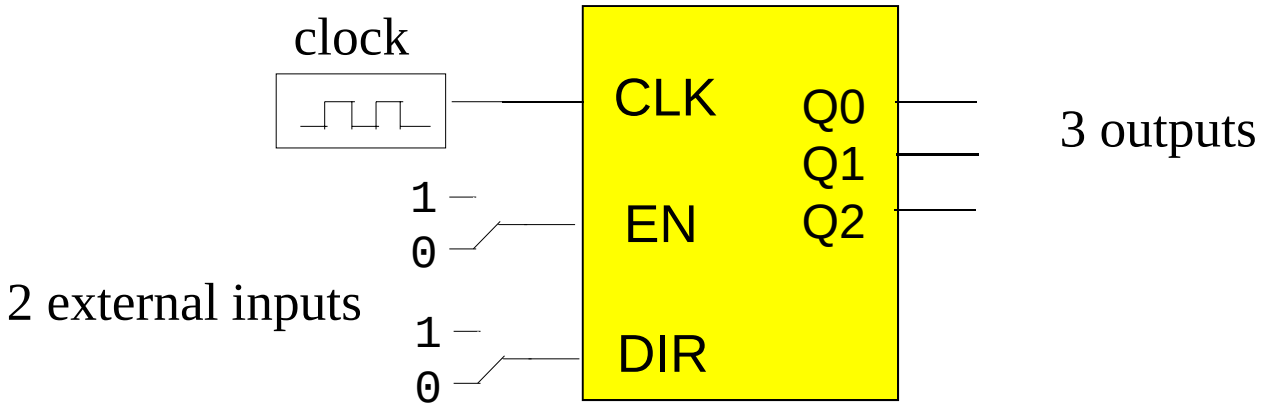
- FSM Design Examples — Video part 2 of 5

- Practice Design Exercises — Video part 3 of 5

- HW #9C FSM Up/Down Counter — Video part 4 of 5

- Review Questions — Video part 5 of 5

Design Project: Up/Down Counter



Design a 3-bit, **up/down counter with enable**. The counter has two external inputs, EN(enable) and DIR (direction). When EN = 1, the circuit count proceeds normally; when EN = 0, the counter stays in the present state (that is, the counter stops counting). When DIR = 1 the counter counts up in the sequence 000, 001, 010, 011, 100, 101, 110, 111, and repeat. When DIR = 0, the counter counts down in the sequence 111, 110, 101, 100, 011, 010, 001, 000, and repeat.

Combinational vs. Sequential Circuits

1) **Combinational Logic Circuit.** -- No presence of latches or flip-flops or memory; only logic gates and MSI devices. Examples include all of the Boolean function circuits designed with truth tables, K-maps, MUX's, Decoders, Adders, MSI devices. Order of inputs does not affect circuit output – because there is no memory.

2) **Sequential Logic Circuit**-- presence of **at least one latch, flip-flop** or memory device. Examples include ripple counters, shift registers, state machines. Sequential circuits are generally a **combination (mix)** of flip-flops and combinational circuitry. There are **2 types of sequential circuits:**

A) **synchronous** -- every flip-flop in the circuit is directly connected to the master clock, and each flip-flop is switched at the same time. Examples include shift registers, ring counters, and state machines (FSM's). Generally, synchronous sequential circuits are preferred because they behave more predictably and are faster.

B) **asynchronous** -- at least one flip-flop is not connected directly to the master clock. Examples include the JK ripple counter. Asynchronous circuits can be useful if they are easy to design and implement.

What is a State Machine?

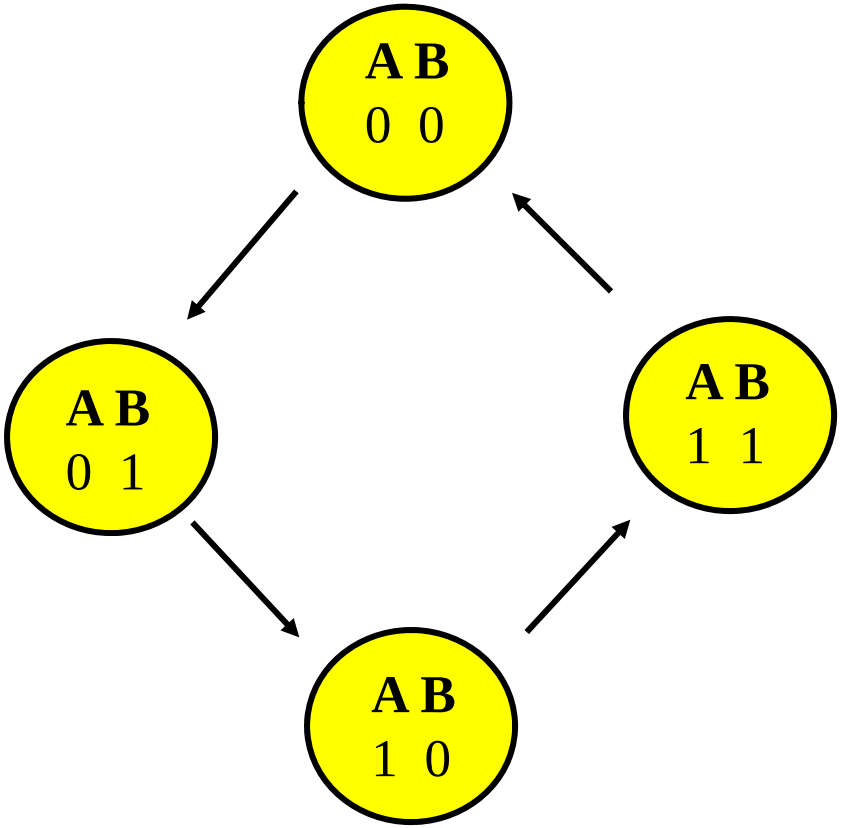
- A **state machine** (also known as a finite state machine or FSM) is a sequential circuit. This means that a state machine is composed of both flip-flops (FFs) and combinational logic (logic gates).
- A state machine **responds to a clock input**. At every clock pulse, the FFs transition (change) into a specified state (1 or 0).
- State machines are always **synchronous**. This means that each FF in a state machine is connected directly to the master clock.
- State machines can be designed with ***any*** **number of FFs** depending on the application.
- State machines are usually designed with **edge-triggered FFs** (usually with either D or JK FFs.)
- State machines can be designed such that the **FFs can sequence through any arbitrary sequence of states**, depending on the application. For example, a state machine can be designed as a 2-bit counter. In this case, 2 FFs would be required, and the count sequence would be 00, 01, 10, 11, 00, and repeat.
- State machines are powerful because the **sequence of states can be completely arbitrary**. For example it is possible to design a state machine with 2 flip-flops that sequences through the states 11,00,10,01, and repeats.

What is a State Machine?

- The FFs in a state machine will only **change state** during the **positive or negative edge** of a clock pulse. If the clock frequency is fast, then the FF transitions will be fast. If the clock frequency is slow, then the FF transitions will be slow. If the clock signal is removed (or turned off) from a state machine, then the state machine will never change state.
- The term "**state**" generally means the stable **output value Q (1 or 0) of each flip-flop** between clock transitions. For example, consider a 2-bit state machine with sequence 00, 01, 10, 11, repeat, with two flip-flops A and B. First, the state machine is in state 00, which means flip-flop A=0, and flip-flop B=0. After the next clock pulse, the state will be A=0, B=1. After the next clock pulse, A=1, B=0, and so on. Again, when you design a state machine, you can pick any arbitrary sequence of states depending on the problem you are trying to solve. (Compare to walking up/down a staircase; hopping from rock to rock on a pond.)
- The **output** of a state machine can be LEDs or devices such as motors, heaters, pumps, taillights, elevators, other computers, other digital devices, etc.
- State machines can be designed to respond to **external inputs** such as pushbuttons (pressed by humans), or temperature sensors, or outputs from other computers or digital devices.
- State machines are often represented by **state diagrams** or "bubble" diagrams. Each "bubble" is a state. Each arrow represents the state transition when a clock pulse arrives.

Finite State Machine - 1

Introduce **State Diagram** (graphical representation of FSM)



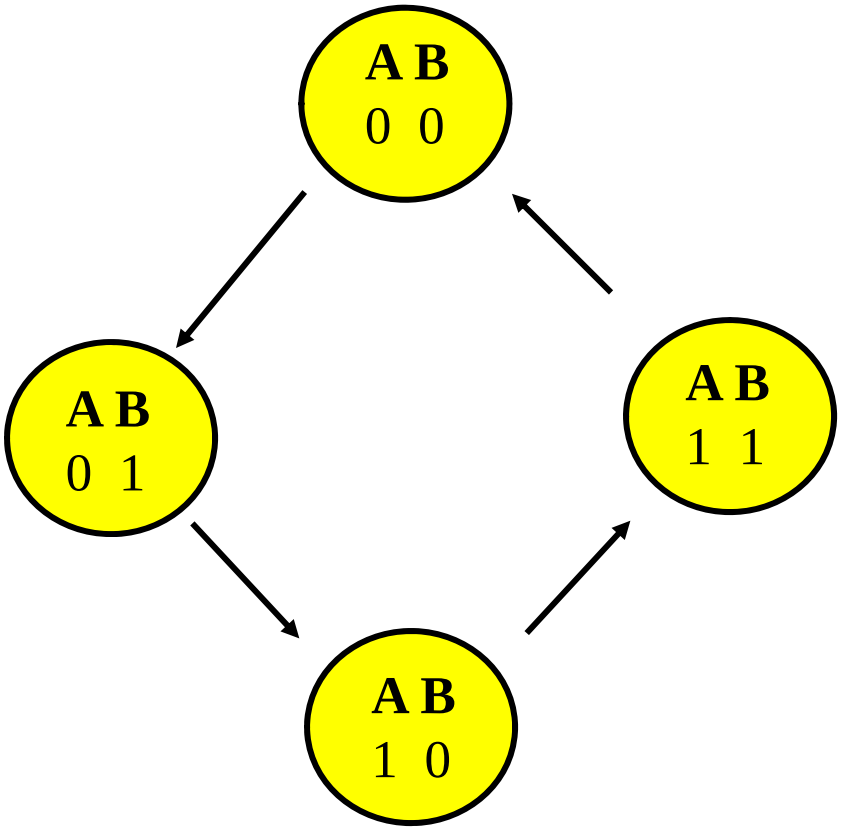
State ("bubble") Diagram

Questions about state diagram:

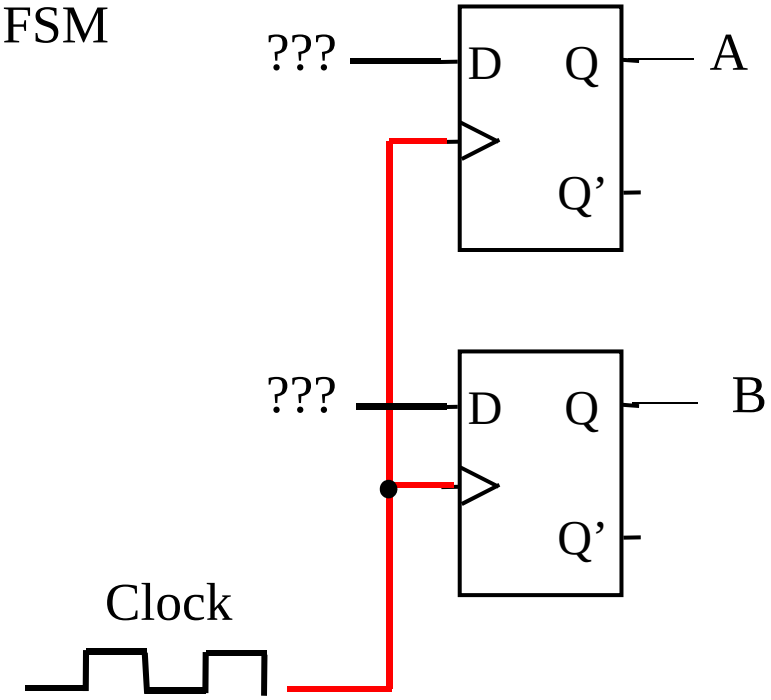
- What is circuit operation?
- What do the bubbles represent?
- What do the arrows represent?
- What is a state?
- How many FFs are required?
- What type of FFs are required?
- Do FFs need to be edge-triggered?
- How many external inputs?
- How many external outputs?
- How fast is this circuit?
- How many states are represented?
- Are there any unused states?
- What do variables A, B represent?
- What is a practical application of the circuit?

Finite State Machine - 2

Goal → Build a circuit to represent FSM



State Diagram Example

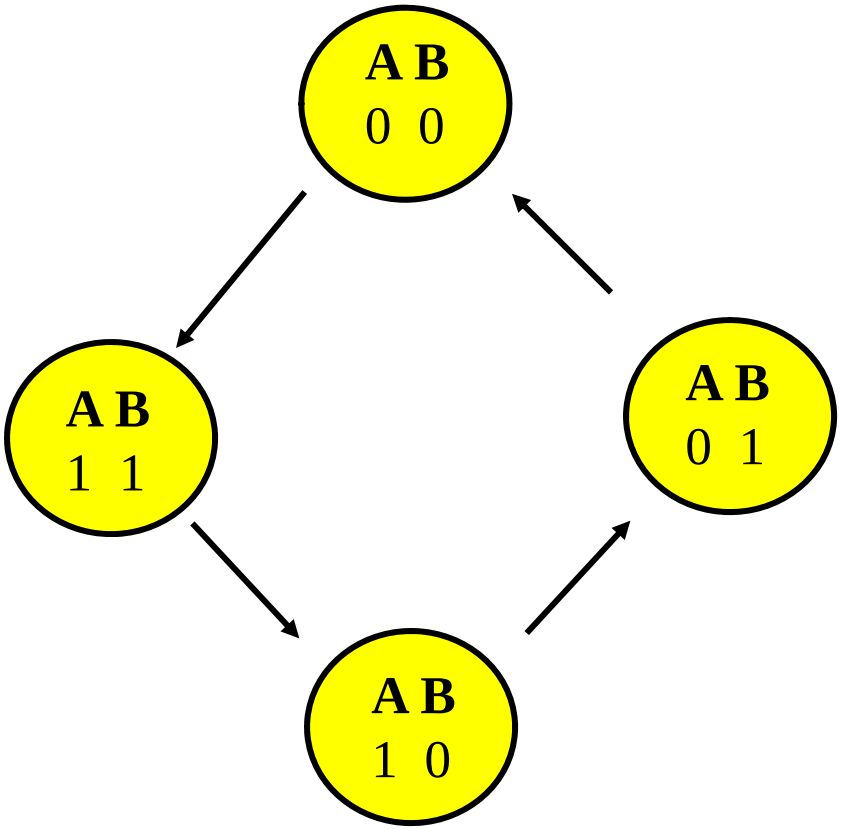


Observations:

1) All FFs synchronized to clock in a FSM.
2) A and B are outputs of FFs.
3) data inputs to D FFs must be found.
4) FFs can be D or JK
5) Our goal is to find circuitry for ???
(called FF input eq's)

Finite State Machine - 3

Another FSM:



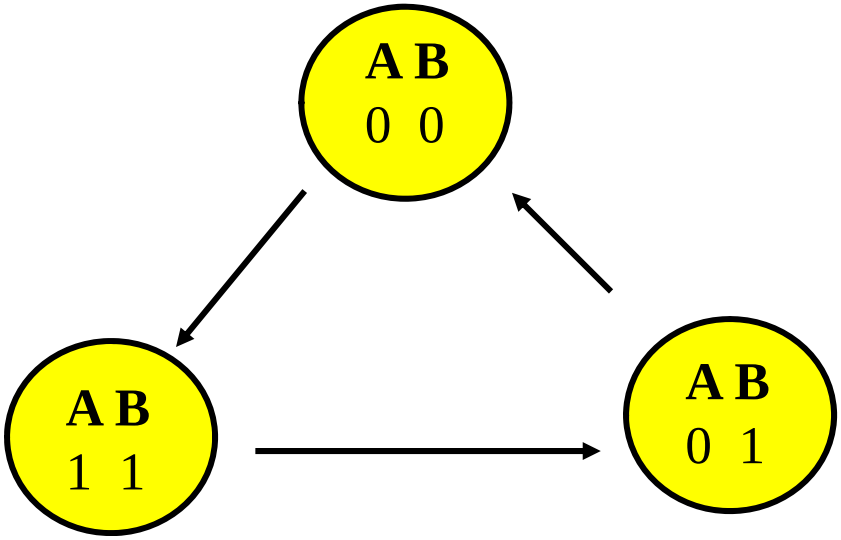
State Diagram Example

Questions about state diagram:

1) What is circuit operation?
2) How many FFs are required?
3) What type of FFs are required?
4) Do FFs need to be edge-triggered?
5) How many external inputs?
6) How many external outputs?
7) How fast is this circuit?
8) How many states are represented?
9) Are there any unused states?
10) What do variables A,B represent?
11) What is a practical application of the circuit?

Finite State Machine - 4

Another FSM:

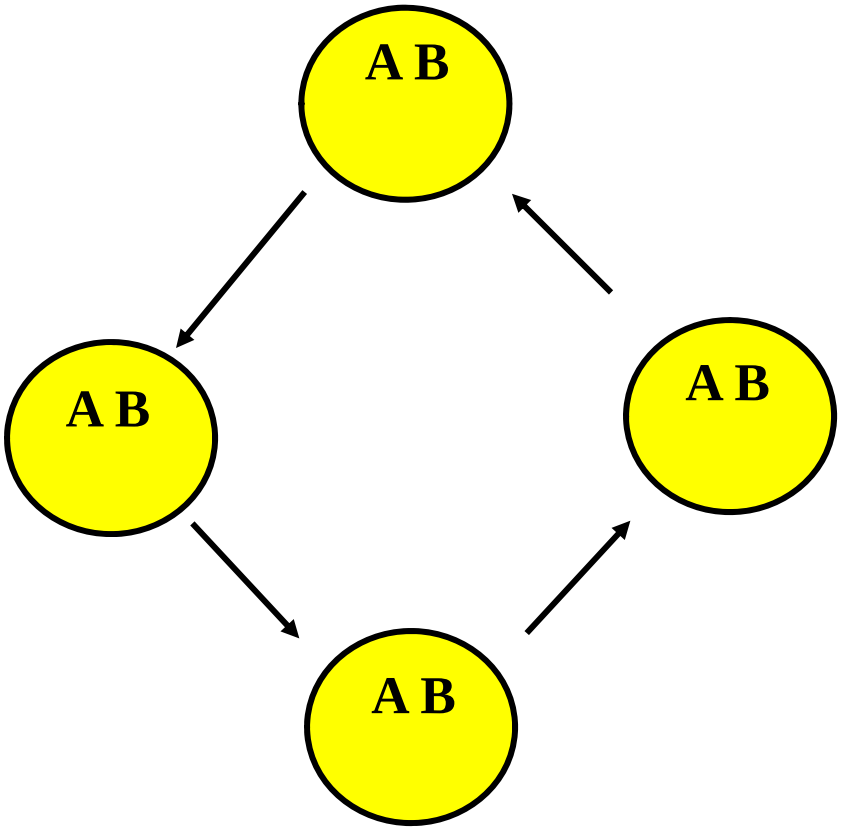


State Diagram Example

Questions about state diagram:

1) What is circuit operation?
2) How many FFs are required?
3) What type of FFs are required?
4) Do FFs need to be edge-triggered?
5) How many external inputs?
6) How many external outputs?
7) How fast is this circuit?
8) How many states are represented?
9) Are there any unused states?
10) What do variables A,B represent?
11) What is a practical application of the circuit?

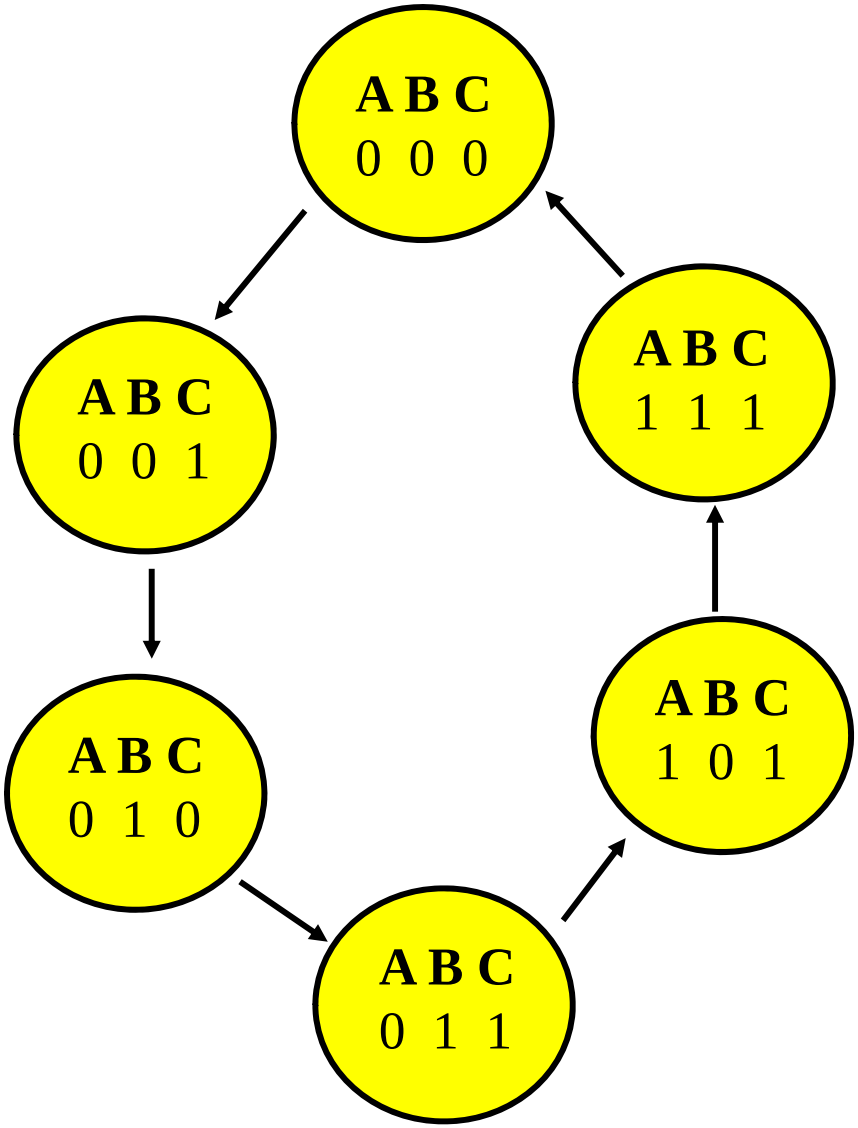
Finite State Machine - 5



State Diagram Example

Design Exercise: Design a state diagram representing a state machine 2-bit counter that counts in reverse from 3(11) to 0(00), then repeats.

Finite State Machine - 6



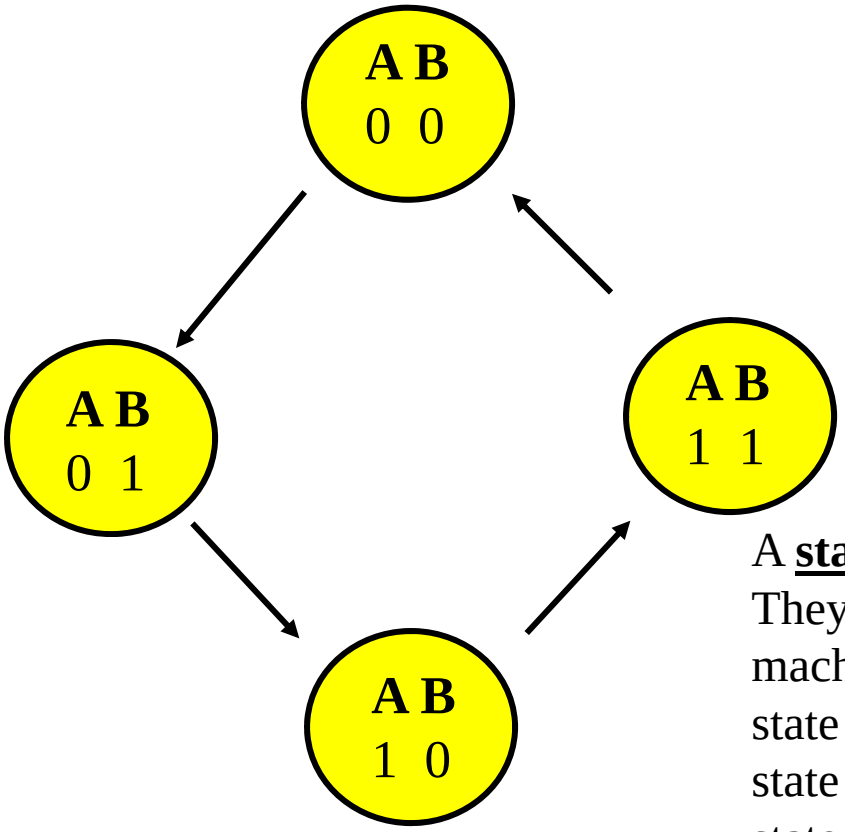
State Diagram Example

Questions about state diagram:

1) What is circuit operation?
2) How many FFs are required?
3) What type of FFs are required?
4) Do FFs need to be edge-triggered?
5) How many external inputs?
6) How many external outputs?
7) How fast is this circuit?
8) How many states are represented?
9) Are there any unused states?
10) What do variables A,B,C represent?
11) What is a practical application of the circuit?

Finite State Machine - 7

Introduce **State Table**

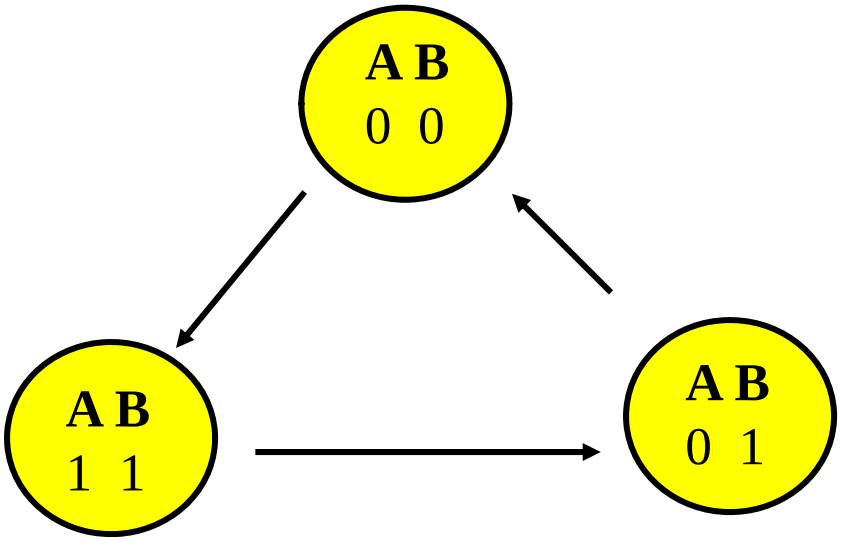


State Diagram

Present State		Next State	
A	**B**	**A**	**B**
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0

A **state diagram** is equivalent to a **state table**. They are 2 different representations of the same state machine. Given any state diagram, you can construct a state table. Given any state table, you can construct a state diagram. Generally, engineers prefer to draw a state diagram first, then construct a state table. "Present state" means before the clock pulse, and "next state" means after the clock pulse.

Finite State Machine - 8

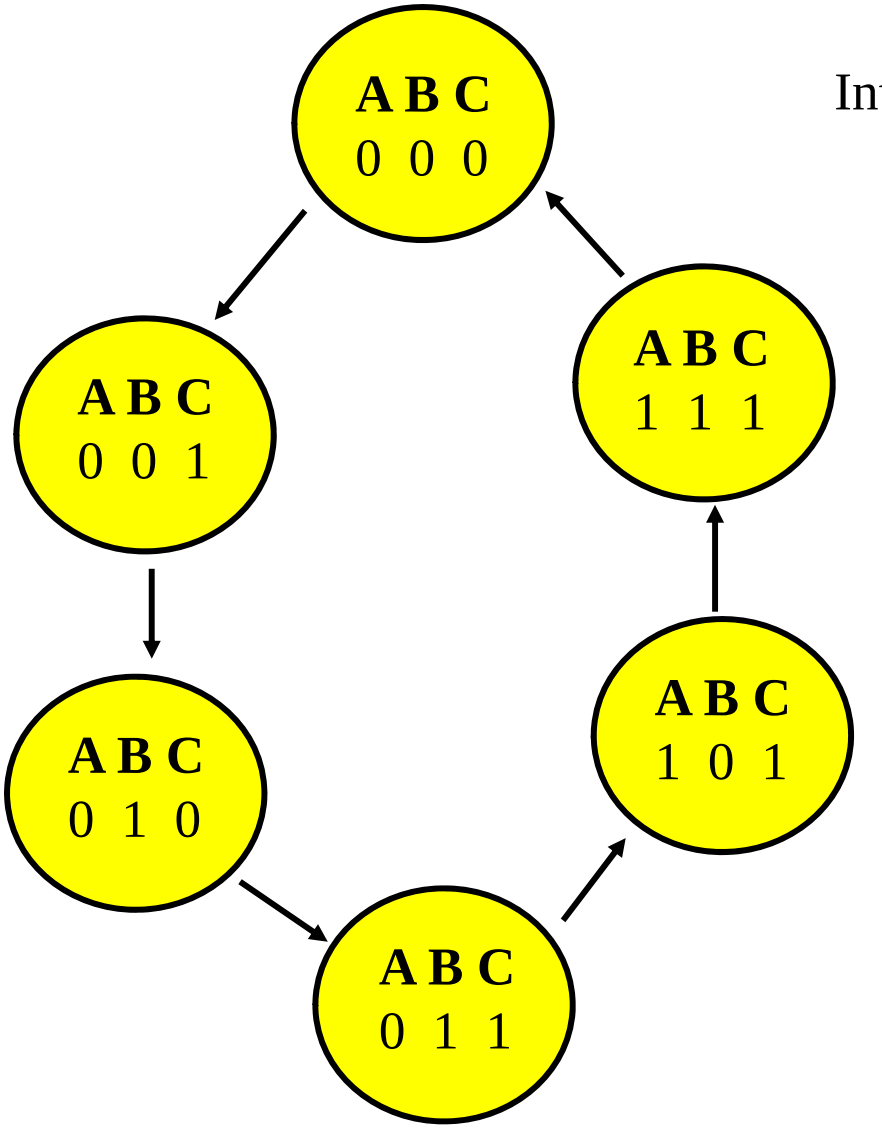


State Diagram Example

Exercise: Complete the state table for the state diagram shown.

Present State		Next State	
A	**B**	**A**	**B**

Finite State Machine - 9



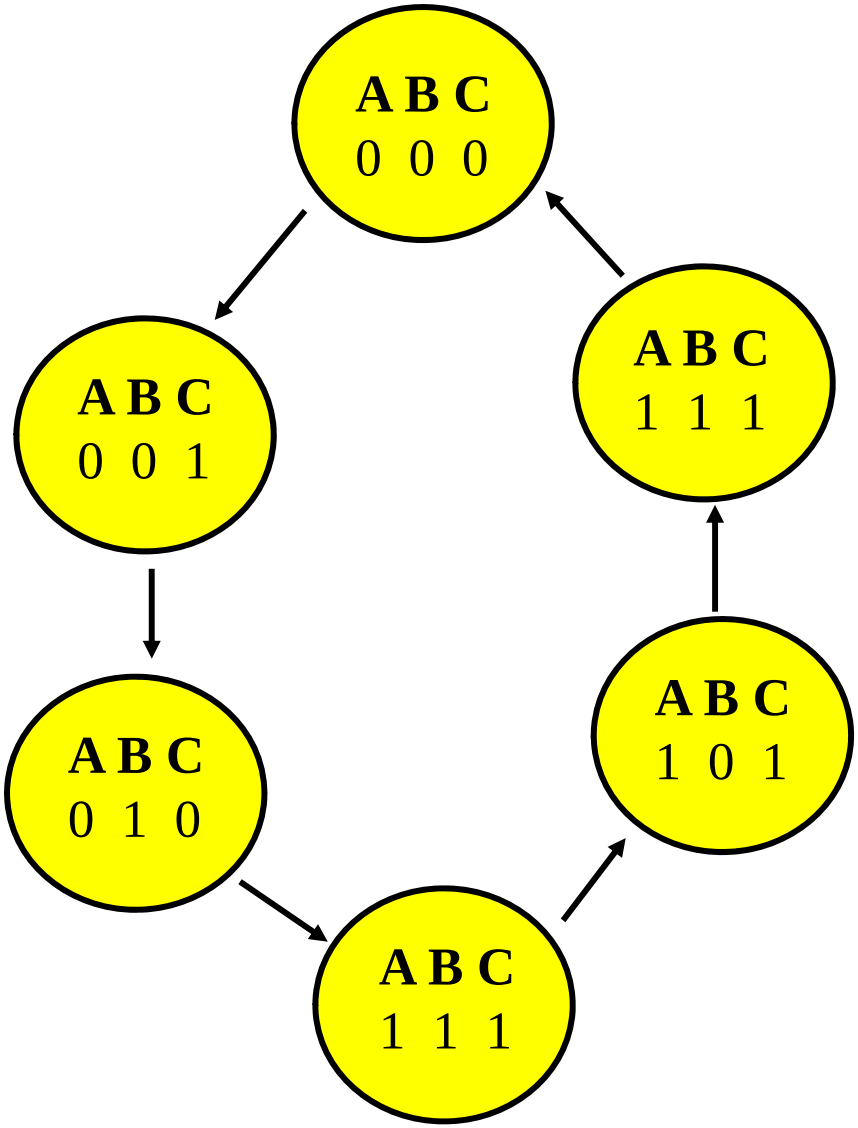
State Diagram Example

Introduce State Table:

Present State			Next State		
A	B	C	A	B	C
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	1
1	0	1	1	1	1
1	1	1	0	0	0

State Table

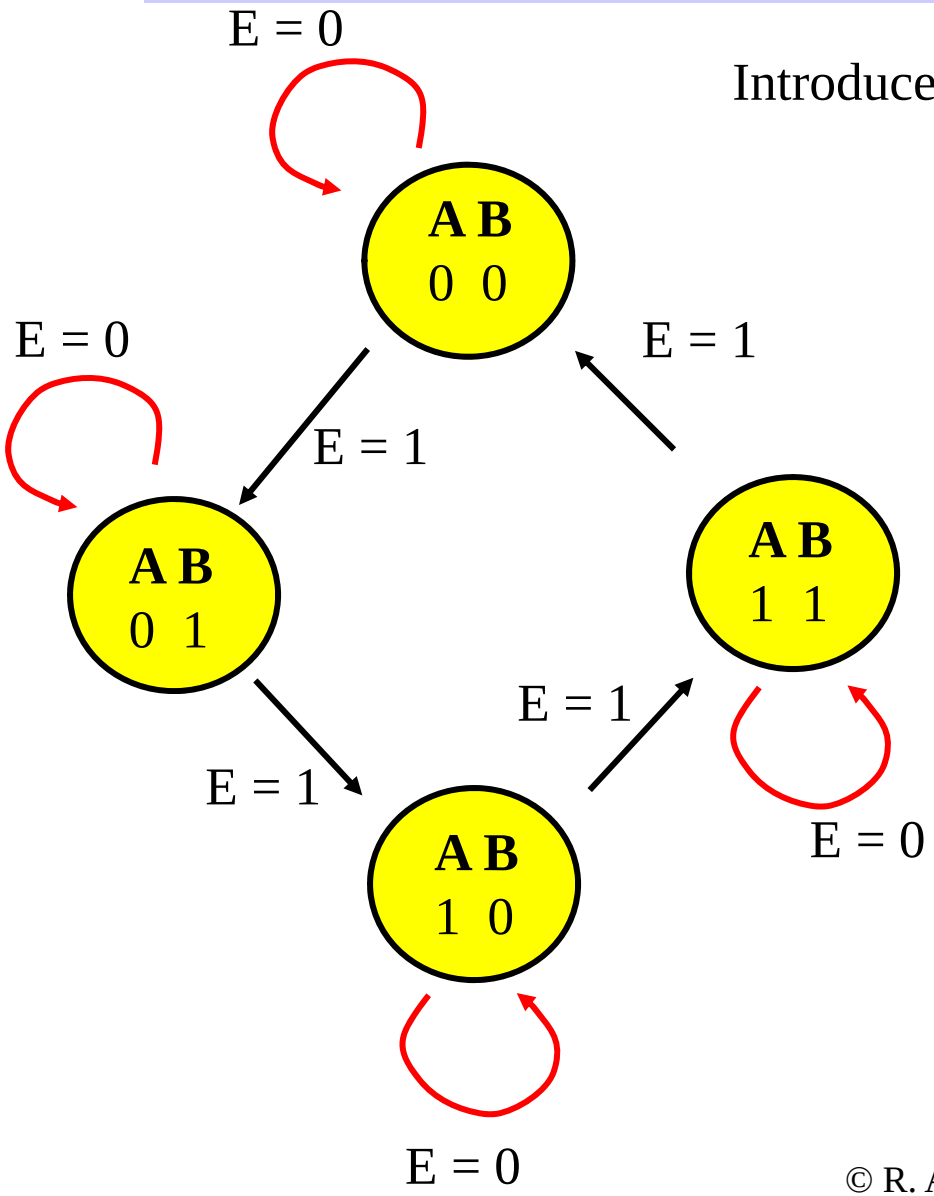
Finite State Machine - 10



State Diagram Example

What is wrong (invalid) about this state diagram?

Finite State Machine - 11

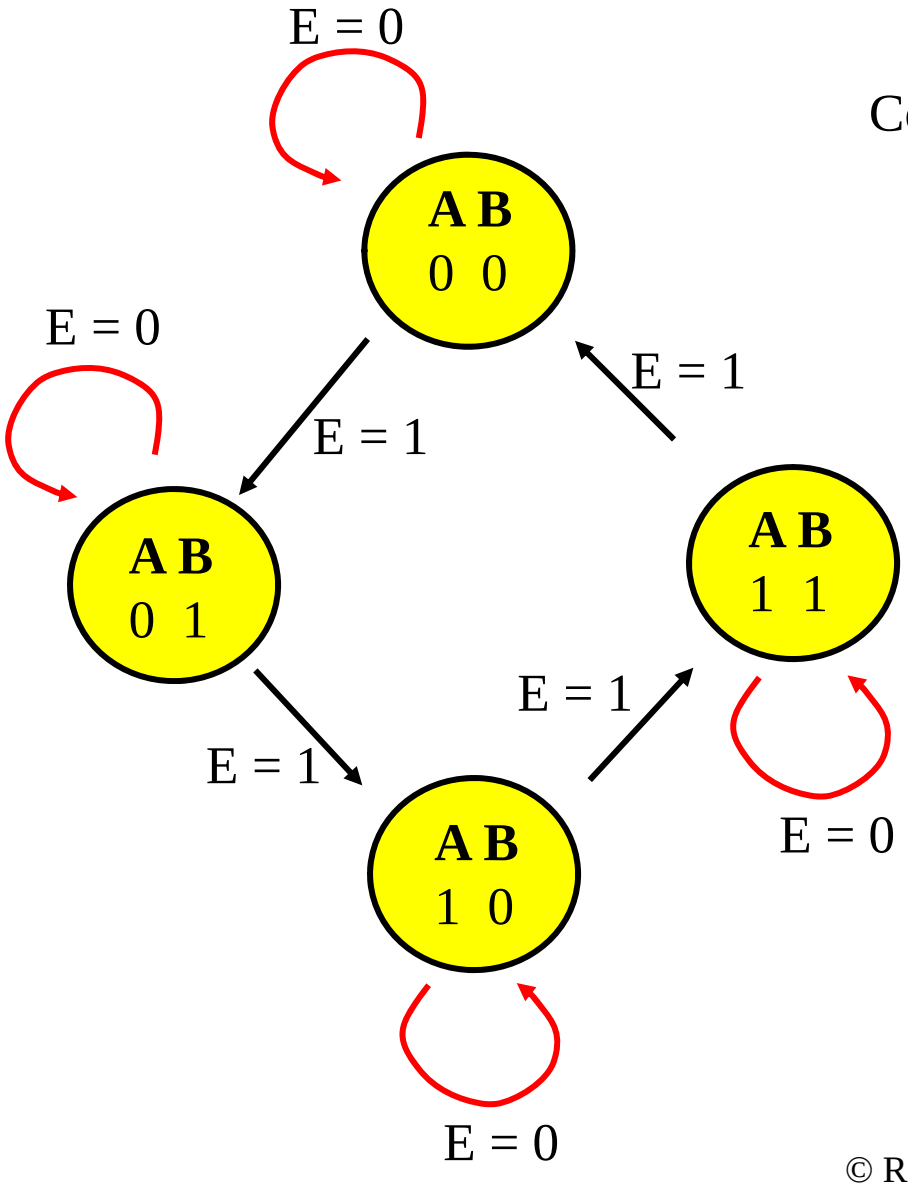


Introduce an **external input E (enable)**

Questions about state diagram:

1) Explain operation
2) How many FFs are required?
3) What type of FFs are required?
4) Do FFs need to be edge-triggered?
5) How many external inputs?
6) How many external outputs?
7) How fast is this circuit?
8) How many states are represented?
9) Are there any unused states?
10) What do variables A,B,E represent?
11) What is a practical application of the circuit?

Finite State Machine - 12

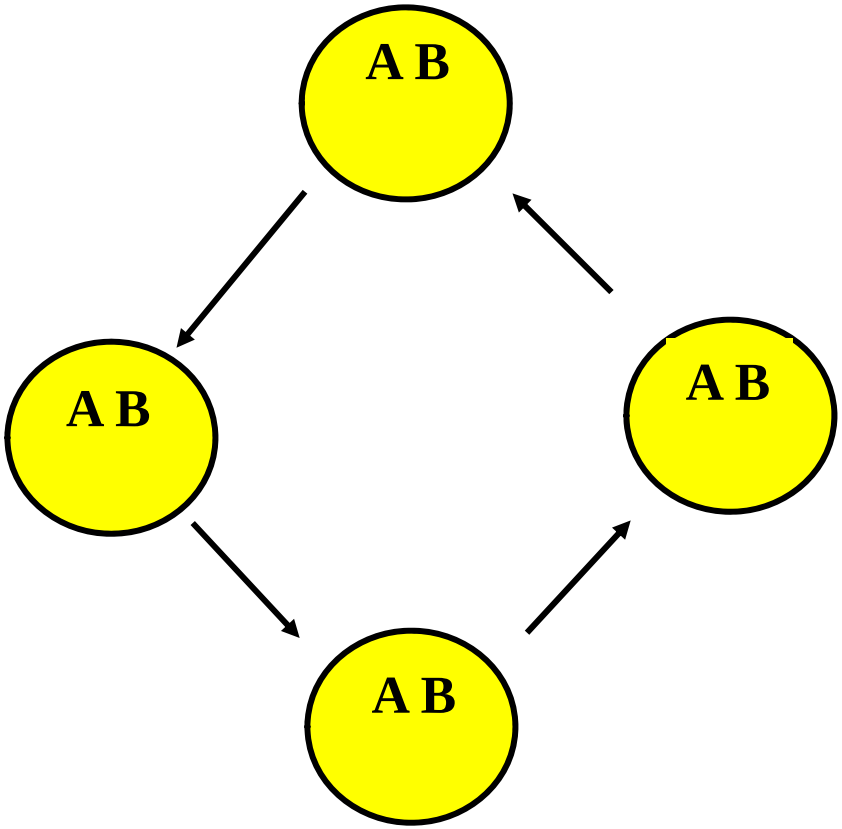


Construct state table from state diagram:

P.S.		input	Next State	
A	B	E	A	B
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

- P.S. = present state
- Explain operation
- E is an external enable input

Finite State Machine - 13



State Diagram

Design Exercise: Part I. Design a **state diagram** representing a state machine 2-bit counter that counts in reverse from 3(11) to 0(00), then repeats. The circuit has one external input, called C. If C = 0, then the count proceeds normally. If C = 1, then the circuit clears (becomes 00) on the next clock pulse.

Part II. Construct **state table** for the circuit above.

P.S.		input	Next State	
A	**B**	**C(clr)**	**A**	**B**
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Penn State Abington
CMPEN 271
Lecture Set #18
State Machines with D FFs

R. Avanzato © 2014-2015

Topics:

- Design Project: Up/Down Counter — Video part 1 of 5
- Sequential versus combinational Logic
- Definition of a finite state machine (FSM)
- Implementing a state diagram with D flip-flops

- FSM Design Examples — **Video part 2 of 5** ←

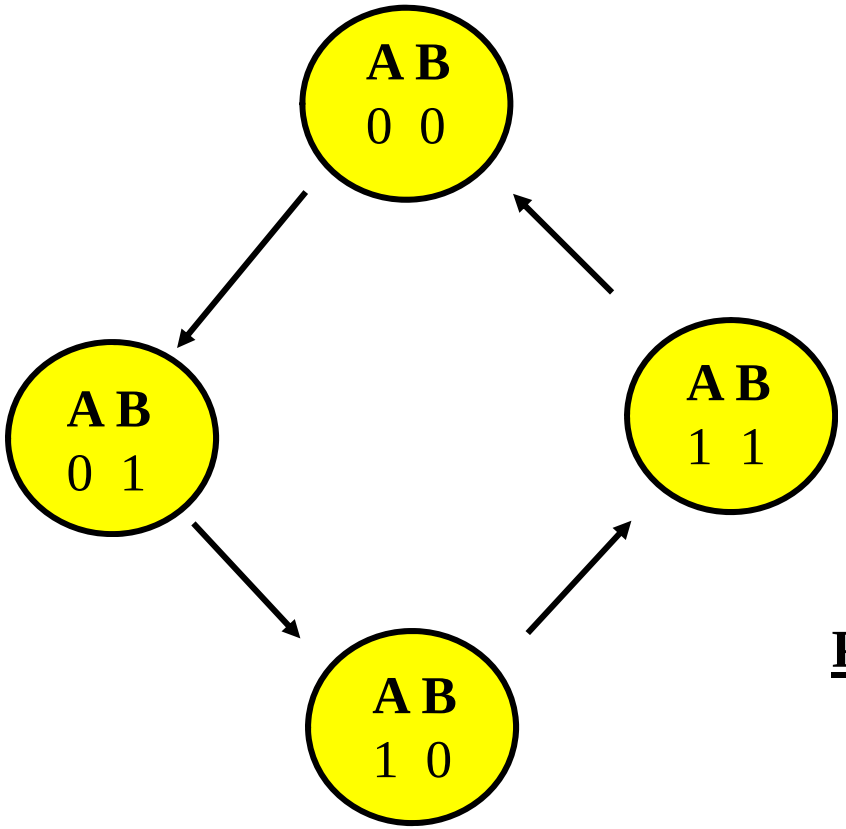
- Practice Design Exercises — Video part 3 of 5

- HW #9C FSM Up/Down Counter — Video part 4 of 5

- Review Questions — Video part 5 of 5

Design Example #1

Introduce **FF input equations**



State Diagram

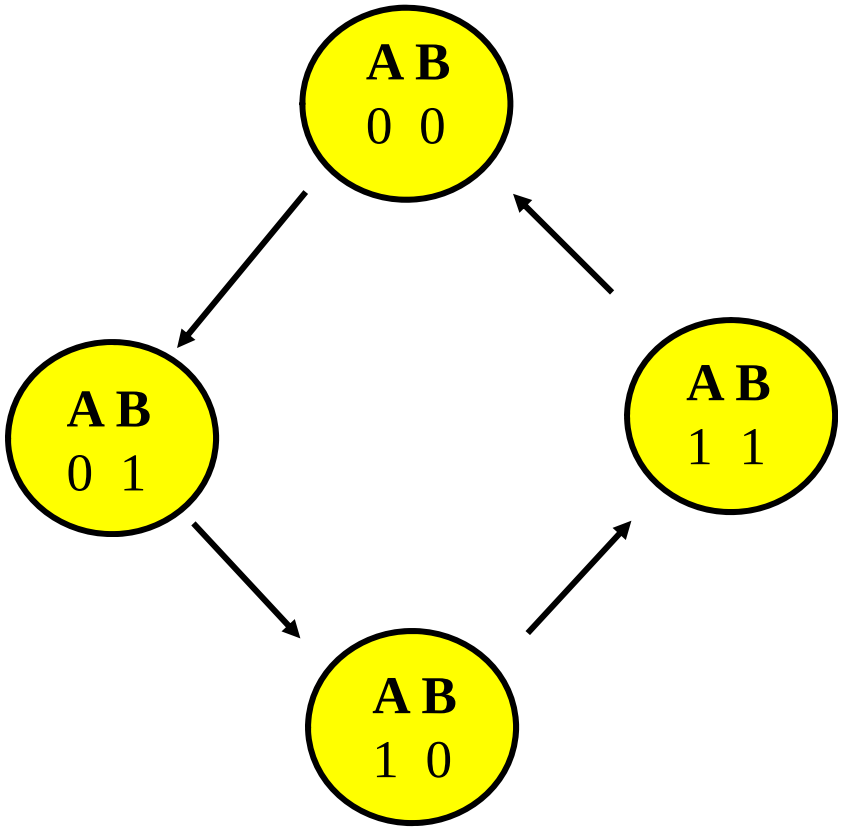
Problem: Design circuit with above state diagram

FSM Design Steps

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 (JK FFs later)
Step#4 - Identify FF input equations (one FF input equation per D FF) as Next State equations
Step#5 - Minimize FF input equations (or use DEC, MUX, etc). Input eq's are Boolean fcns.
Step#6 - Draw circuit (and simulate).

Present State		Next State		FF input eq's	
A	B	A	B	DA	DB
0	0	0	1	0	1
0	1	1	0	1	0
1	0	1	1	1	1
1	1	0	0	0	0

Design Example #1



State Diagram

Present State		Next State		FF input eq's	
A	B	A	B	DA	DB
0	0	0	1	0	1
0	1	1	0	1	0
1	0	1	1	1	1
1	1	0	0	0	0

Notes:
1) DA = next state A, DB = next state B.
2) DA and DB are Boolean functions of present state A and B. Truth tables.

DA = A xor B

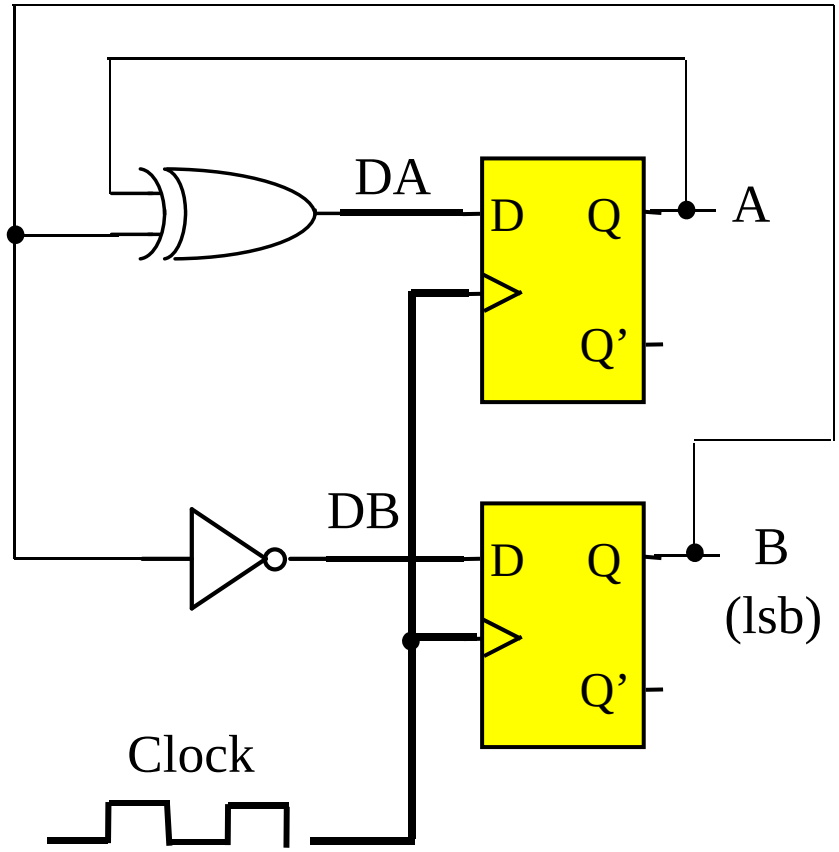
DB = B'

Design Example #1

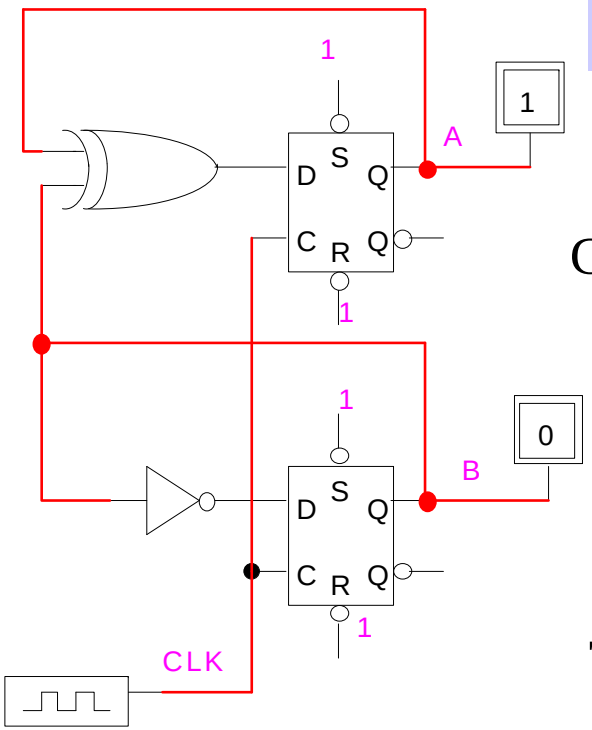
Flip-flop input equations
(need one for each D flip-flop)

DA = A xor B

DB = B’

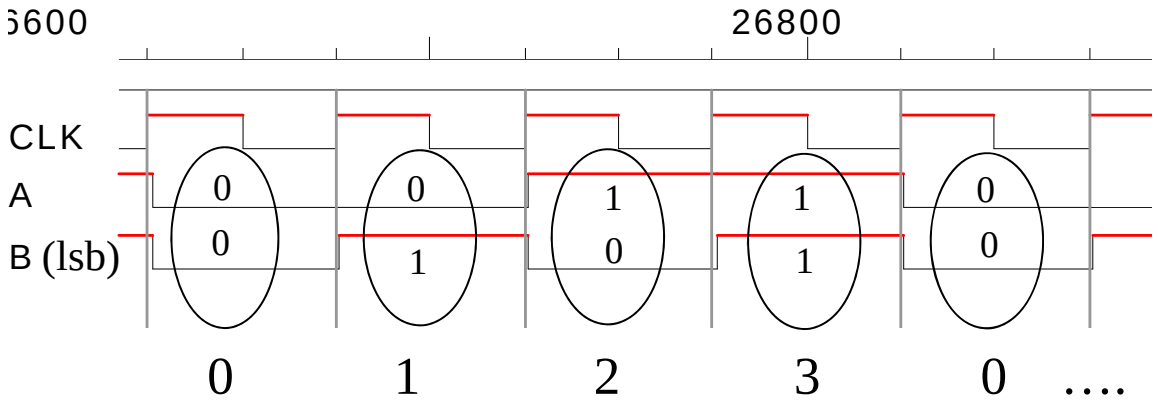


Design Example #1

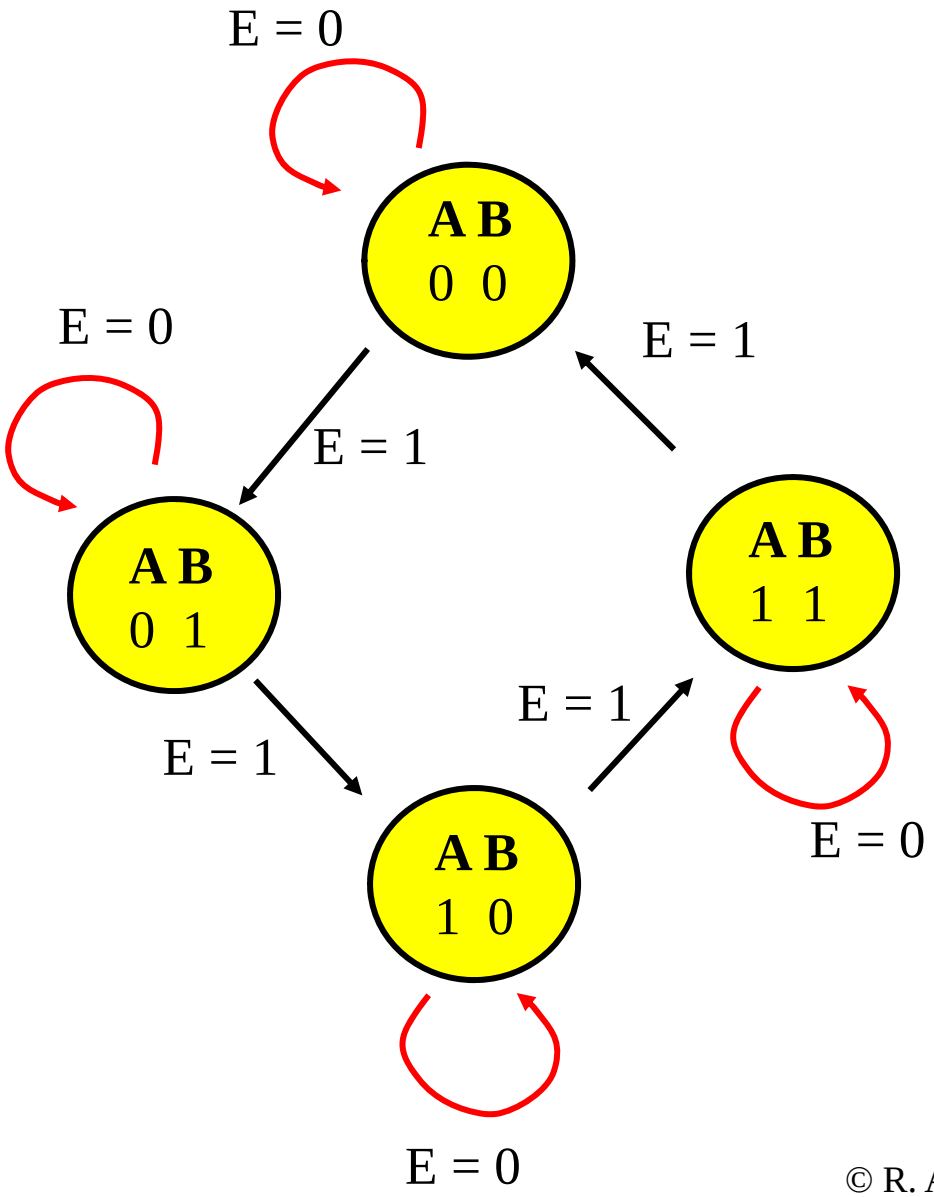


Circuit (D FF are pos. edge triggered in LogicWorks)

Timing diagram with markups/comments



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 D flip-flops).

Step#1 - construct state diagram

Design Example #2

Step #1 - construct state diagram (done)
Step #2 - construct state table.
Step #3 - use D FFs (given in this problem)
Step #4 - identify the two needed FF input equations

P.S.		input	Next State		FF input eq's	
A	**B**	**E**	**A**	**B**	**DA**	**DB**
0	0	0	0	0	0	0
0	0	1	0	1	0	1
0	1	0	0	1	0	1
0	1	1	1	0	1	0
1	0	0	1	0	1	0
1	0	1	1	1	1	1
1	1	0	1	1	1	1
1	1	1	0	0	0	0

Flip-flop Input Equations

$DA(A,B,E) = \Sigma m(3,4,5,6)$

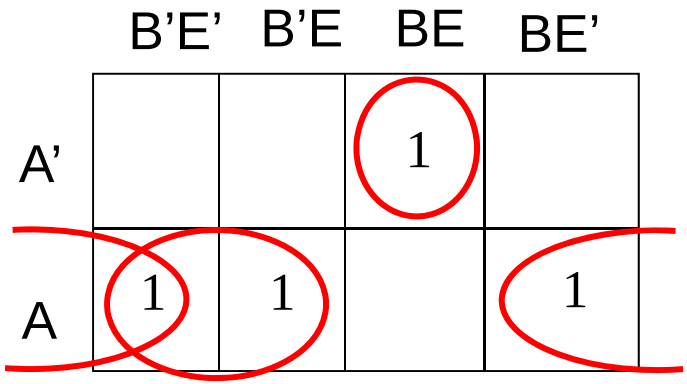
$DB(A,B,E) = \Sigma m(1,2,5,6)$

Observation: Output DA is same as output A, and output DB is same as output B

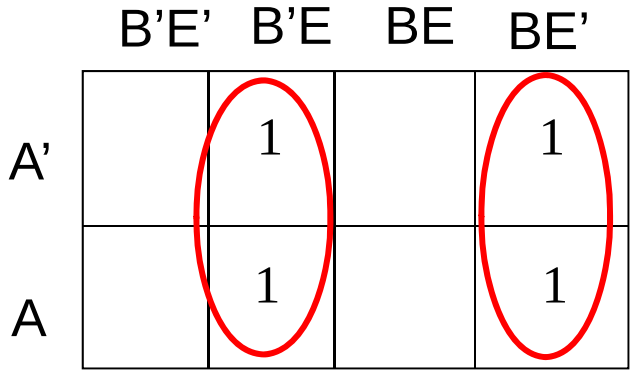
Design Example #2

Step#5 - Minimize FF input equations

DA(A,B,E) = Σm(3,4,5,6)



DB(A,B,E) = Σm(1,2,5,6)



DA(A,B,E) = AE' + AB' + A'BE

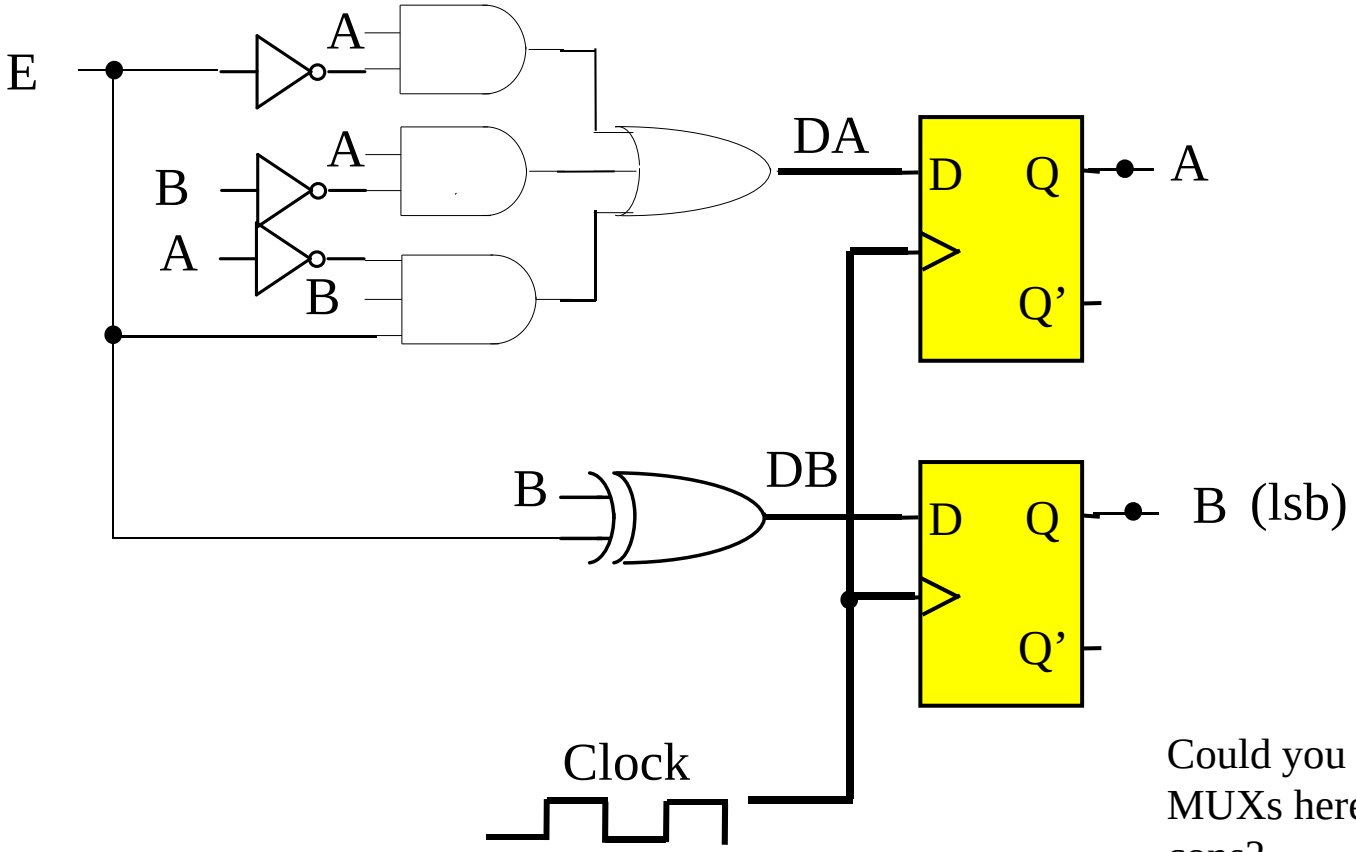
DB(A,B,E) = B'E + BE'
= B xor E

Design Example #2

Step #6 - Draw circuit

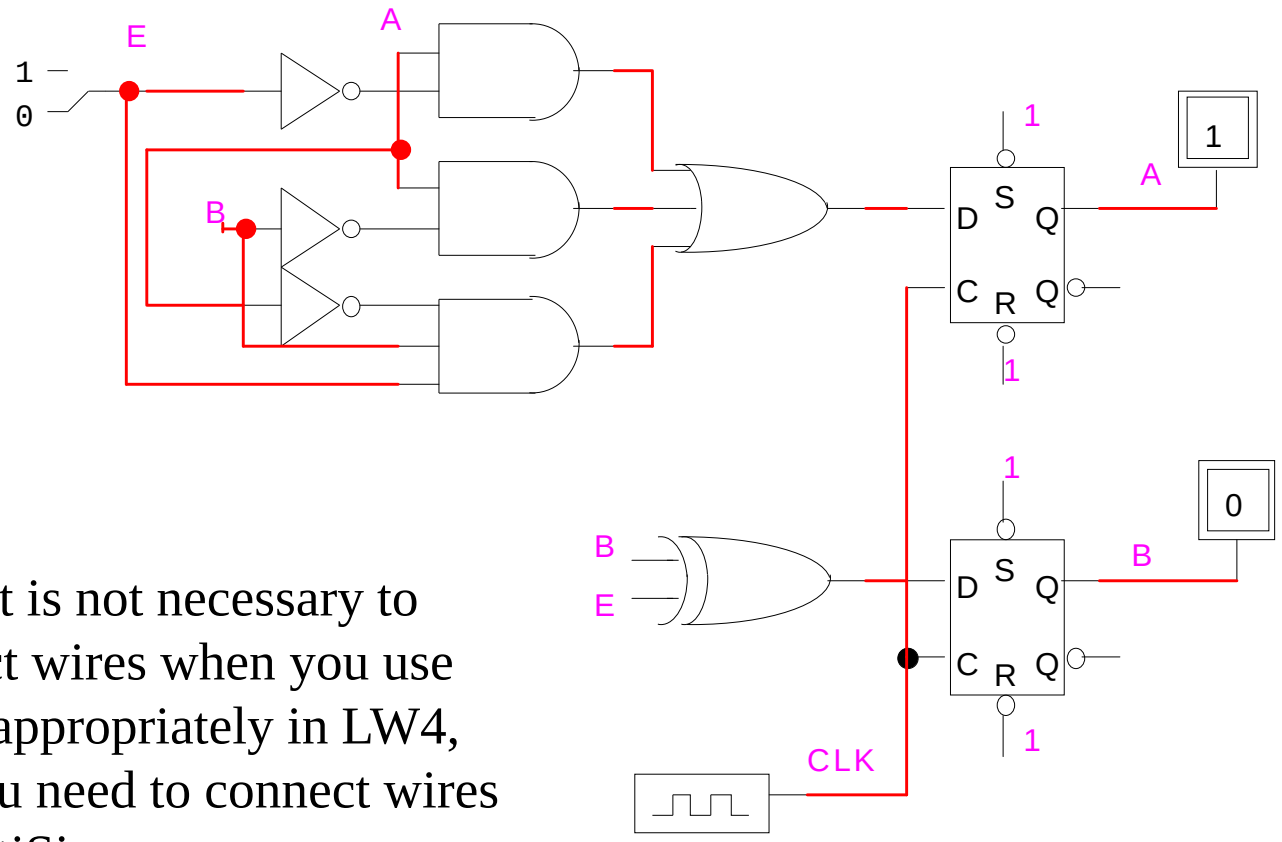
$DA(A,B,E) = AE' + AB' + A'BE$

$DB(A,B,E) = B'E + BE' = B \text{ xor } E$



Could you use Decoders or MUXs here? What are pros and cons?

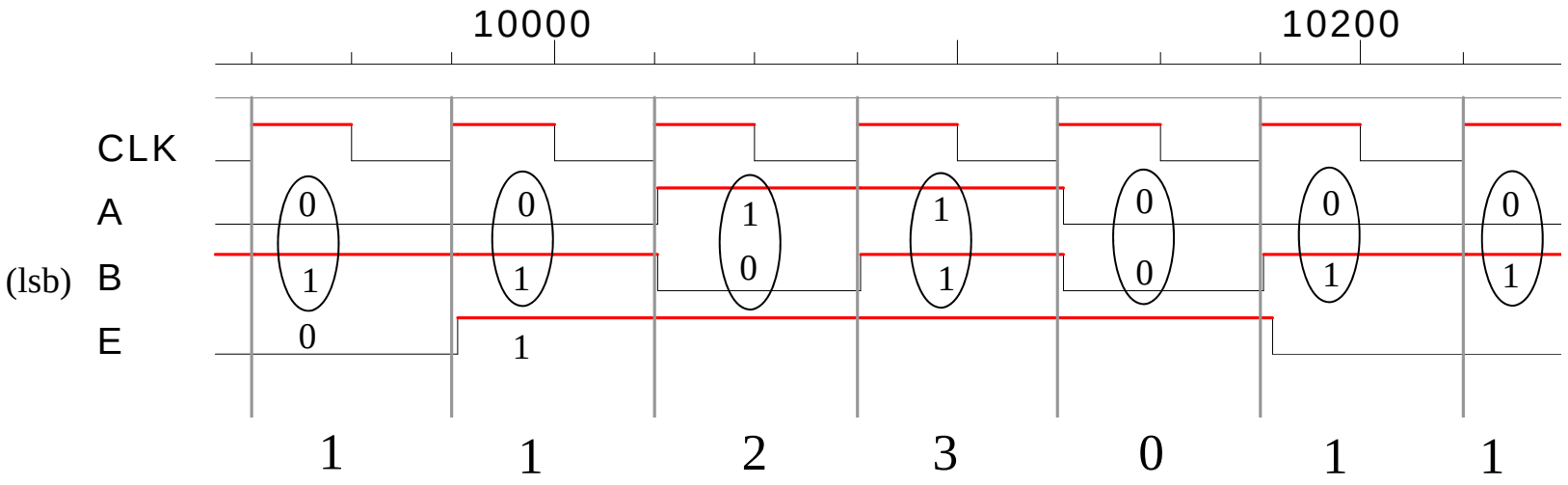
Design Example #2



Note: it is not necessary to connect wires when you use labels appropriately in LW4, But you need to connect wires in MultiSim

Design Example #2

Timing Diagram with E input (enable) and markups/comments



E = 0
count
stays
at 1

E=1
count
normally

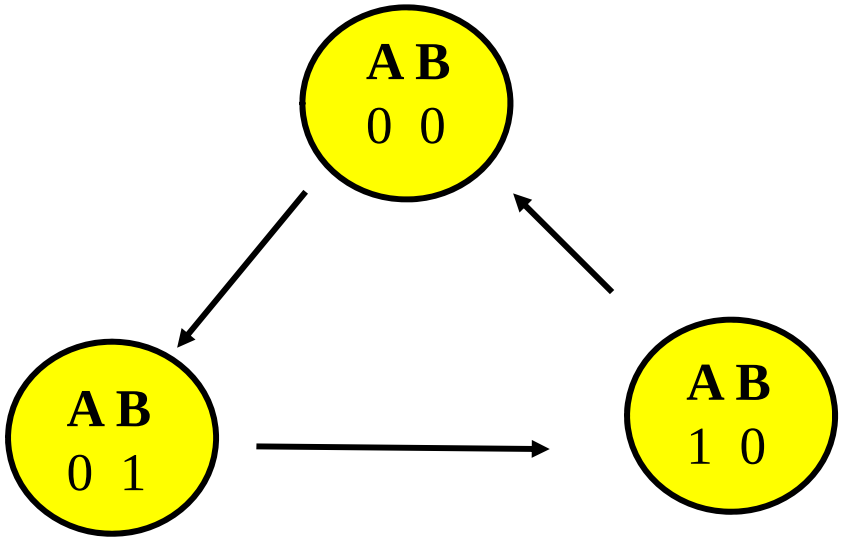
E = 0,
count
stays at 1

Design Example #2

How would you characterize (describe) the circuit you constructed for Example #2? (check all that apply)

a) State machine
b) Combinational Circuit
c) Sequential Circuit
d) Synchronous Circuit
e) Asynchronous Circuit

Design Example #3



Design Problem: Design a finite state machine that produces output 00, 01, 10, then repeats. Use D flip-flops.

Step#1 - construct state diagram

- How many flip-flops are needed?
- Are there any unused states?
- Are there any external inputs?

Design Example #3

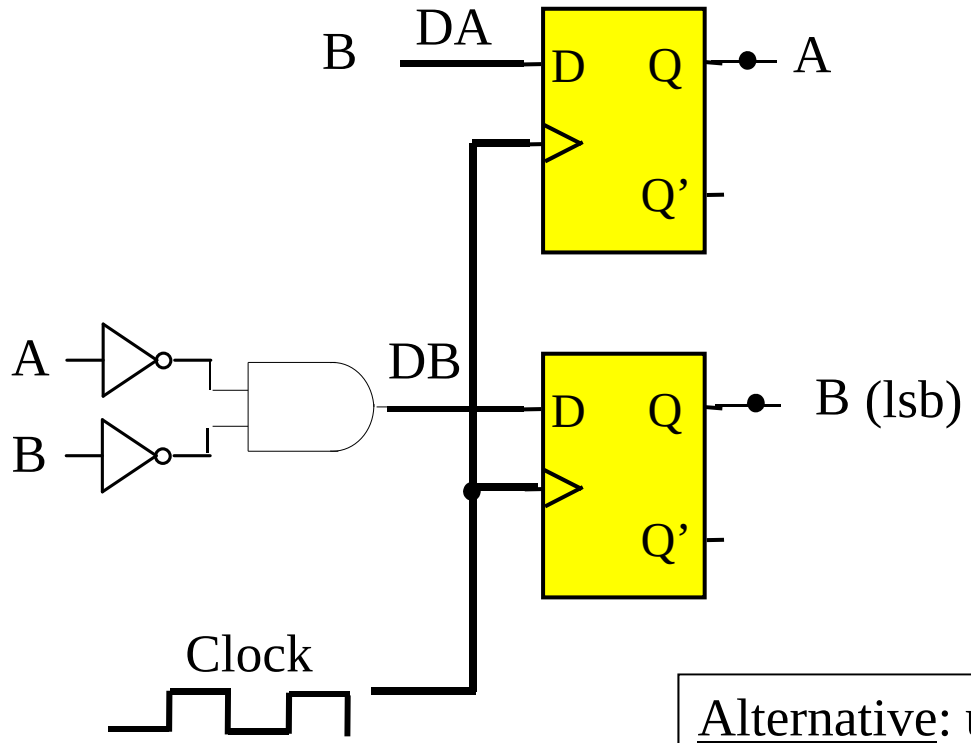
	Present State		Next State		FF input eq.s		
	A	**B**	**A**	**B**	**DA**	**DB**	
	0	0	0	1	0	1	
	0	1	1	0	1	0	
	1	0	0	0	0	0	
unused state -	1	1	X	X	X	X	don't cares

DA = B (by inspection)

DB = A'B'

Design Example #3

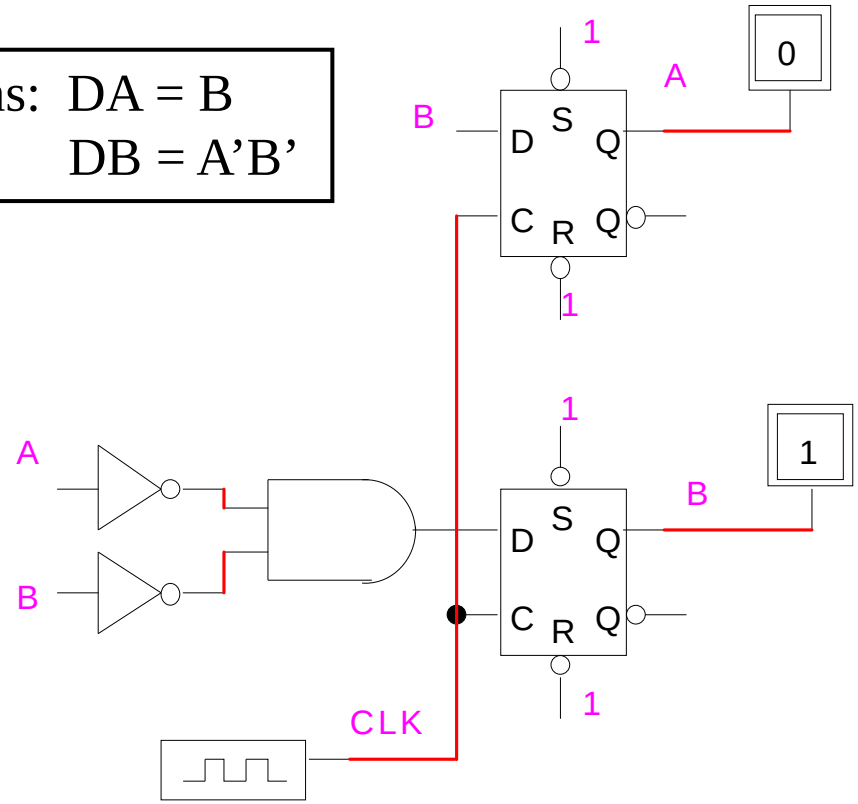
FF Input Equations: DA = B
DB = A'B'



Alternative: use Q' of A FF as A' and Q' of B FF as B'. This reduces number of gates.

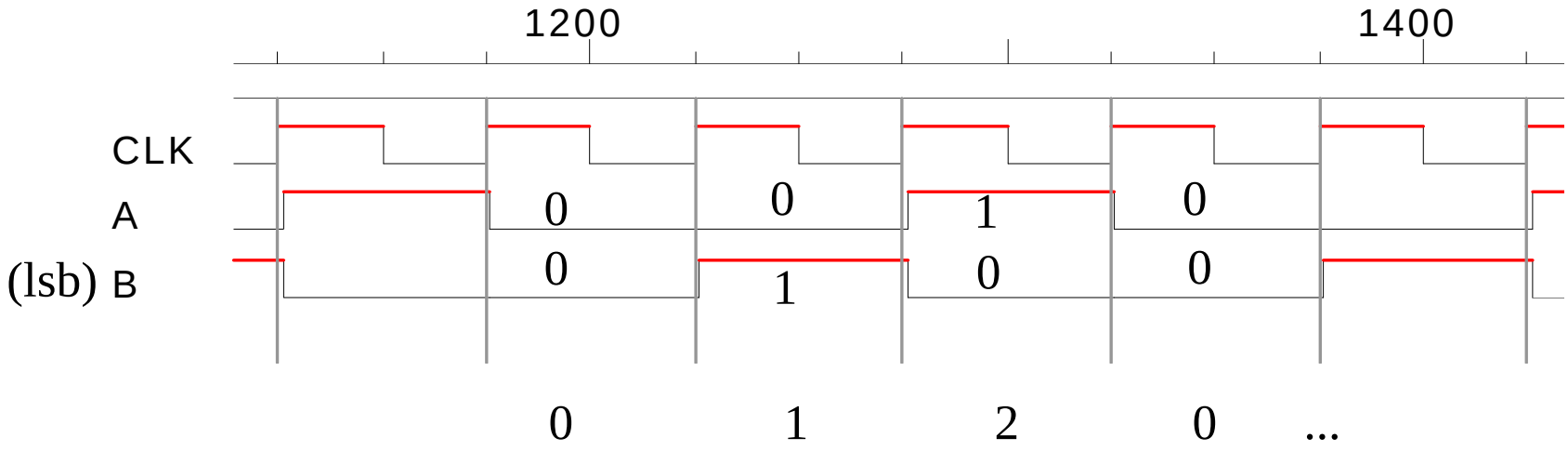
Design Example #3

FF Input Equations: DA = B
DB = A'B'



Circuit (LW4)

Design Example #3



Timing Diagram (LogicWorks) with Markups/Comments

Finite State Machine Concepts

- Finite state machines (FSMs) are **sequential circuits** (they contain flip-flops).
- A **master clock** is connected directly to each one of the FFs in a FSM.
- FFs in a FSM transition to **specific states** (1 or 0) each clock cycle based on design of the FSM and any external inputs.
- A **state diagram** is a graphical representation of the FF states and transitions.
- State machines can optionally have **external inputs and external outputs**.
- In many cases, the **outputs of the FFs** are used as outputs of the circuits. External outputs based on a combination of FF outputs and inputs are possible. For example, an external output might be used to turn on a timer or other device.
- The **state diagram** and the **state table** are equivalent.
- FSMs can be built with either **D FFs or JK FFs**.
- FSMs require either **positive-edge triggered or negative-edge triggered** FFs.
- State diagrams and state tables are **independent of the type of FF** being used.
- In a D FF design, the "**next state**" equations are equal to the D FF "**input equations**".
- **FF input equations are Boolean functions** that can be implemented using a) K-maps, b) decoders, c) multiplexers.
- **Unused states in a FSM** can be used as don't cares -- this can simplify circuitry in the final design.
- State machine designs should be verified using a **software simulation package**.

Penn State Abington
CMPEN 271
Lecture Set #18
State Machines with D FFs

R. Avanzato © 2014-2015

Topics:

- Design Project: Up/Down Counter — Video part 1 of 5
- Sequential versus combinational Logic
- Definition of a finite state machine (FSM)
- Implementing a state diagram with D flip-flops

- FSM Design Examples — Video part 2 of 5

- Practice Design Exercises — **Video part 3 of 5** ←

- HW #9C FSM Up/Down Counter — Video part 4 of 5

- Review Questions — Video part 5 of 5

FSM Practice Exercises

1. Design a FSM counter with sequence 0,1,3,2, repeat Use D flip-flops.

2. Design a FSM counter with sequence 0,1,2,3,4,5,6,7, repeat. Use D FFs. Compare with a ripple counter solution for the same sequence.

3. Design a FSM counter with sequence 0,1,2,3,4,5,6,7, repeat. Include enable input E. When E = 1, the count sequences normally. When E=0, the count stays in the present state.

4. Design a FSM counter with sequence 0,1,2,3,4,5,6,7, repeat. Include enable input E. When E = 1, the count sequences normally. When E=0, the count stays in the present state. Include another input CL. When CL = 0 the count precedes normally. When CL=1, the count returns to the zero state. Input CL has higher priority than E input. E and CL are external inputs

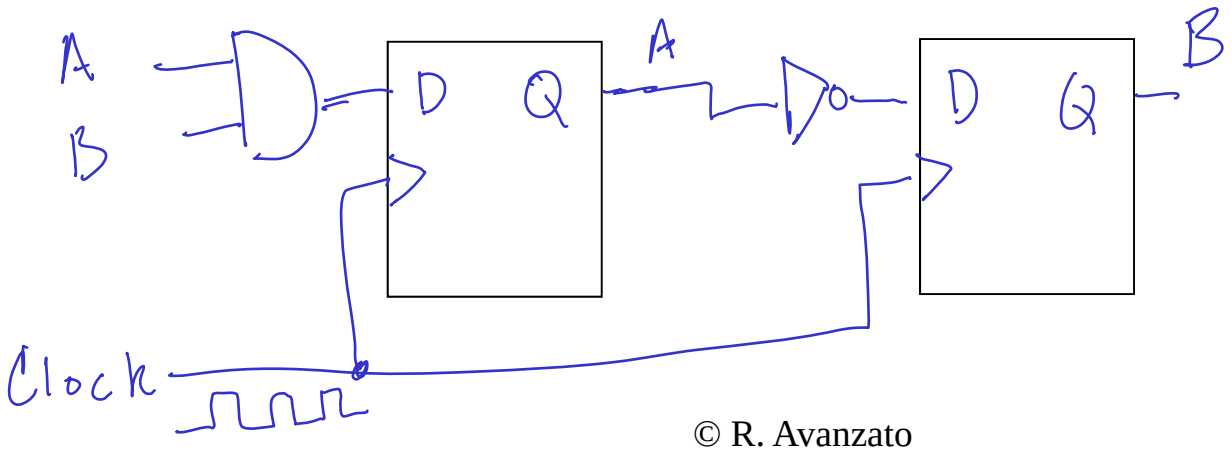
5. Design a FSM counter with sequence 4,3,1,6,7, repeat. Use D FFs. Take advantage of unused states.

See solutions on Angel (CMS)

FSM Practice Exercises

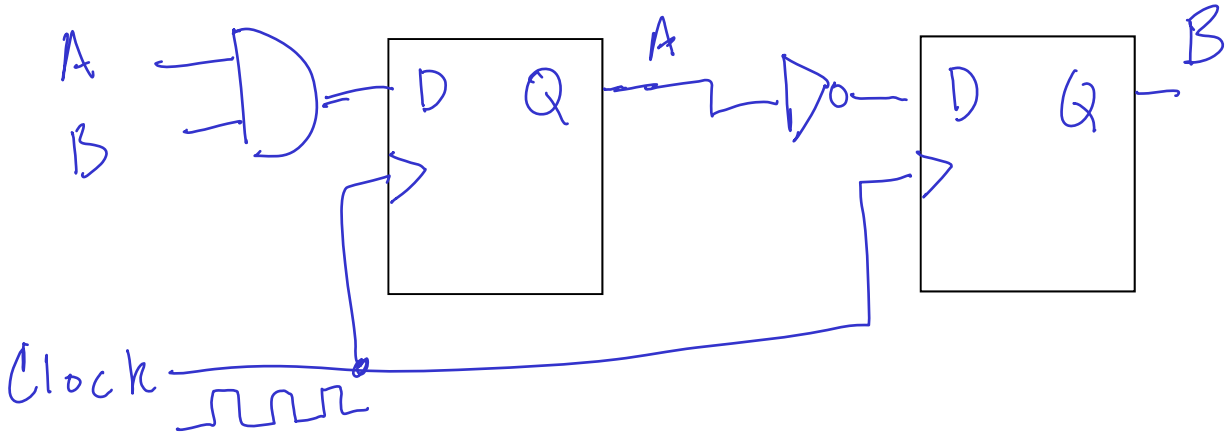
- Be able to "**reverse engineer**" a state machine. For example, if you are given an arbitrary state machine circuit (consisting of flip-flops, gates, etc), then you should be able to:
  - Write down the "input equations" for each flip-flop. Each input equation is a Boolean function which is a function of the outputs (Qs) of the flip-flops and any external inputs. There is one FF input equation for each D flip flop.
  - Construct a complete "state table"
  - Construct a complete "state diagram"

- Given the state machine circuit below, write down the a) FF input equations, b) the state table, and c) the state diagram



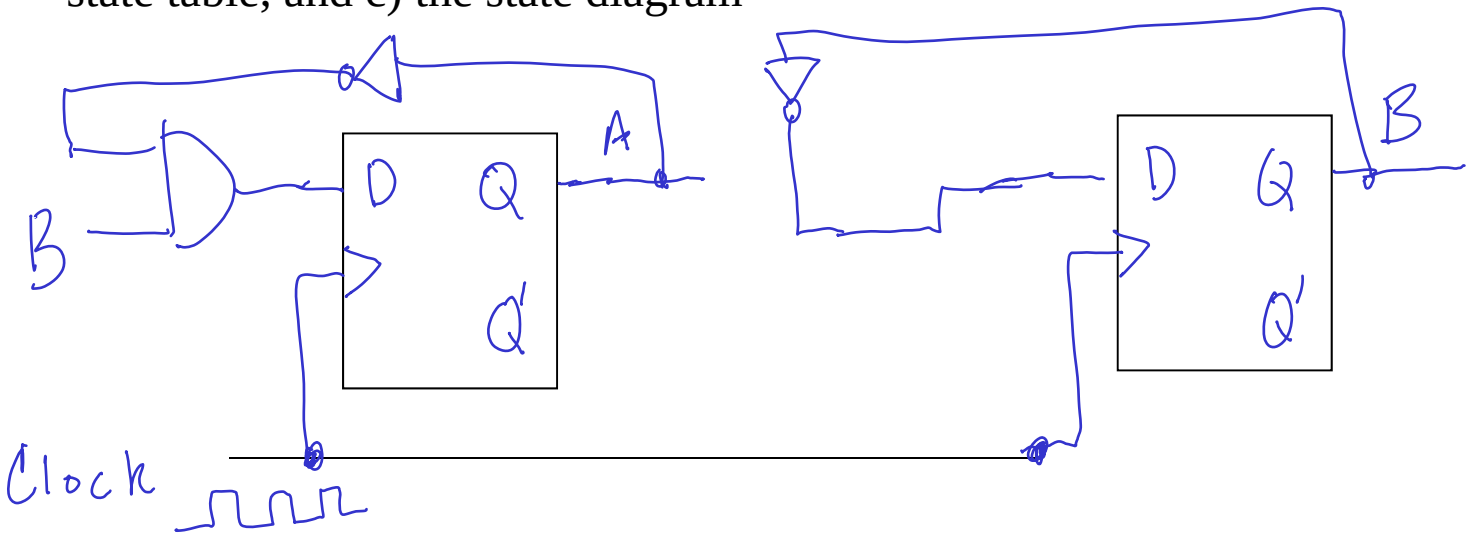
FSM Practice Exercises

#1. Given the state machine circuit below, write down the a) FF input equations, b) the state table, and c) the state diagram



FSM Practice Exercises

- Given the state machine circuit below, write down the a) FF input equations, b) the state table, and c) the state diagram



See solutions on Angel CMS

In-class Design Project – 1

Design a sequential FSM circuit to control the taillights (turn signals) on the 1965 Thunderbird (ref: Wakerly). You solved this problem using a shift register "ring counter". Now solve the same problem with the FSM approach. **Sequence: 000 (all off), 100, 110, 111(all on), repeat**. (Observation: count sequence is 0, 4, 6, 7, repeat) Assume a clock is available.

a) Construct state diagram
b) Construct state table
c) Add flip-flop input equations to truth table and minimize
d) Draw circuit
e) Simulate circuit
f) Deliver circuit and timing diagram with markups/comments.
g) Compare FSM approach with the ring counter approach
- Which solution was easier to develop? Explain.
- Which solution was implemented with less ICs?
- What are the tradeoffs?

(See Solution on Angel CMS)

Penn State Abington
CMPEN 271
Lecture Set #18
State Machines with D FFs

R. Avanzato © 2014-2015

Topics:

- Design Project: Up/Down Counter — Video part 1 of 5
- Sequential versus combinational Logic
- Definition of a finite state machine (FSM)
- Implementing a state diagram with D flip-flops

- FSM Design Examples — Video part 2 of 5

- Practice Design Exercises — Video part 3 of 5

- HW #9C FSM Up/Down Counter — **Video part 4 of 5** ←

- Review Questions — Video part 5 of 5

HW #9C FSM Up/down counter

Design and simulate a 2-bit **FSM up/down counter with enable**. The counter has **two external inputs**, EN(enable) and DIR (direction). When EN = 1, the circuit count proceeds normally; when EN = 0, the counter stays in the present state. When DIR = 1 the counter counts up in the sequence 00, 01, 10, 11, repeat. When DIR = 0, the counter counts down in the sequence 11, 10, 01, 00, repeat. EN (enable) input has higher priority. Use D flip-flops.

a) construct state diagram
b) construct state table
c) add flip-flop input equations to
truth table and minimize
(use K-maps or any MSI devices)
d) draw circuit
e) simulate circuit
f) deliver circuit and timing diagram with markups/comments.

Homework #9C (see schedule)

- **#9C Up/down counter FSM**; include state diagram, state table, circuit simulation, timing diagram with markups. Show all analysis, For FF input equations, you can use combinational logic (include K-maps) or MSI devices (MUXs, decoders, etc). Show effect of all inputs n the timing diagrams. You may need to include 3 or 4 annotated timing diagrams for this problem. Make sure EN and DIR are also on the timing diagrams. Ask instructor questions if needed.

- **HW #9A, #9B, #9C is a total of 3 problems** with simulations. Submit each solution as a Word doc or pdf in the appropriate drop box. Include title and brief description of each problem. Make sure simulation output timing diagrams are annotated for each problem. Use proper file name: first name, initial of last name and HW number (example: “John A HW 9C.doc”)

- **Extra credit** : Read and summarize "How to be a Star Engineer" by Robert Kelley (this article is on Angel). Write 10 to 15 sentence summary (in your own words) in Word doc. Submit to separate drop box.

Further Reading

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

- Wakerly, John F., Digital Design:Principles and Practices, Prentice-Hall, Chapter 7.

Penn State Abington
CMPEN 271
Lecture Set #18
State Machines with D FFs

R. Avanzato © 2014-2015

Topics:

- Design Project: Up/Down Counter — Video part 1 of 5
- Sequential versus combinational Logic
- Definition of a finite state machine (FSM)
- Implementing a state diagram with D flip-flops

- FSM Design Examples — Video part 2 of 5

- Practice Design Exercises — Video part 3 of 5

- HW #9C FSM Up/Down Counter — Video part 4 of 5

- Review Questions — **Video part 5 of 5** ←

Review Questions

#1.- A “synchronous” circuit means that every FF in the circuit is

a) is a D FF b) is a JK FF c) directly connected to master clock

d) has a reset e) is positive-edge triggered

#2.- Every sequential circuit is synchronous.

a) true b) false

#3.- Every synchronous circuit is sequential.

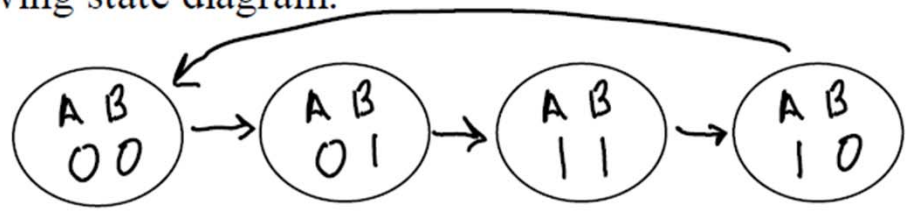
a) true b) false

#4.- A state machine is a(n) _____ circuit.

a) synchronous b) asynchronous c) combinational d) analog

Review Questions

Consider the following state diagram:



#5.- How many states are represented in the state diagram above?

a) 1 b) 2 c) 3 d) 4 e) 5

#6.- How many FFs (min) would be required to implement the state diagram above?

a) 1 b) 2 c) 3 d) 4

#7.- How many unused states are represented in the state diagram above?

a) 0 b) 1 c) 2 d) 3

#8.- What is the FF input equation DA represented in the state diagram above?

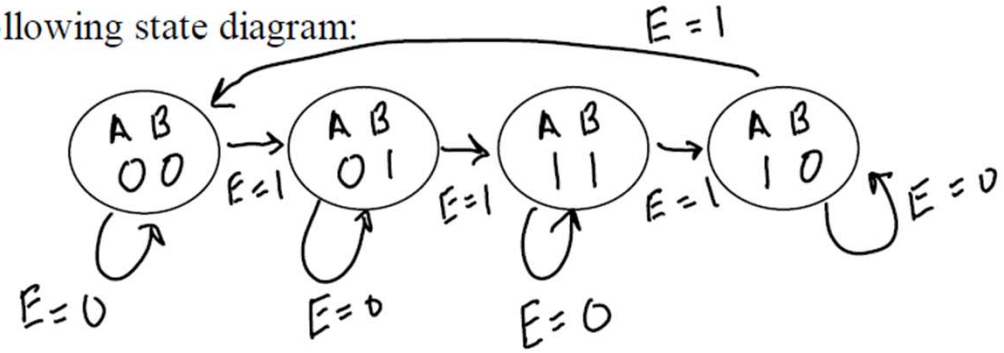
a) DA = A b) DA = B c) DA = AB d) DA = 0 e) DA = 1

#9.- What is the FF input equation DB represented in the state diagram above?

a) DB = A b) DB = A' c) DB = B d) DB = 0 e) DB = 1

Review Questions

Consider the following state diagram:



#10.- How many external inputs are represented in the state diagram above?

a) 0 b) 1 c) 2 d) 3

#11.- Assume initially A=1, B=1. What is value of A, B after 3 clock pulses?
Let input E=1.

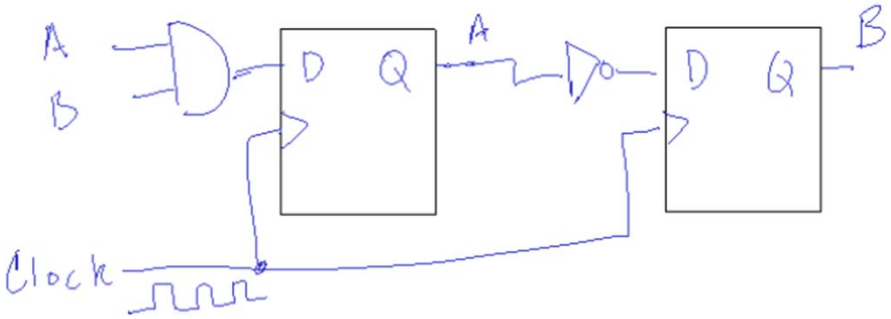
a) A=0, B=0 b) A= 0, B=1 c) A=1, B=0 d) A=1, B=1

#12.- Assume initially A=1, B=0. What is value of A, B after 3 clock pulses?
Let input E=0.

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

Review Questions

Consider the state machine below:



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

a) DA = A b) DA = A' c) DA = Clock d) DA = AB

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

a) DB = A b) DA = AB c) DB = Clock d) DB = A'

#15.- Draw the state table for state machine circuit above.

a)

P. S.		N. S.	
A	B	A	B
0	0	0	1
0	1	0	1
1	0	0	0
1	1	1	0

b)

P. S.		N. S.	
A	B	A	B
0	0	0	1
0	1	1	0
1	0	1	1
1	1	1	0