EE6094
CAD for VLSI Design

PA4: NCVerilog and Behavior Description

Andy, Yu-Guang Chen
Assistant Professor, Department of EE
National Central University
andyygchen@ee.ncu.edu.tw
Slides Credit: TA李宗穎

Outline

- Verilog Simulation
- Examples
- Cases

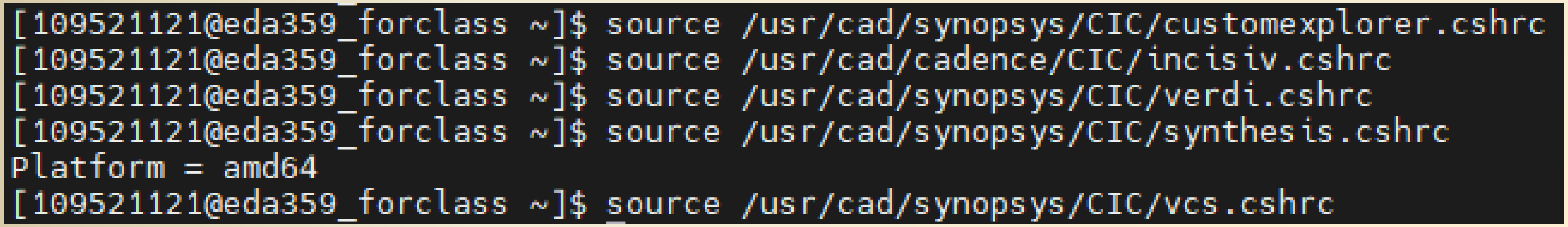
Outline

- ◆Verilog Simulation
- ◆Examples
- ◆Cases

Verilog Simulation

- How to use the Verilog Simulation on workstation?
  - Step1: Key in the following source commands.

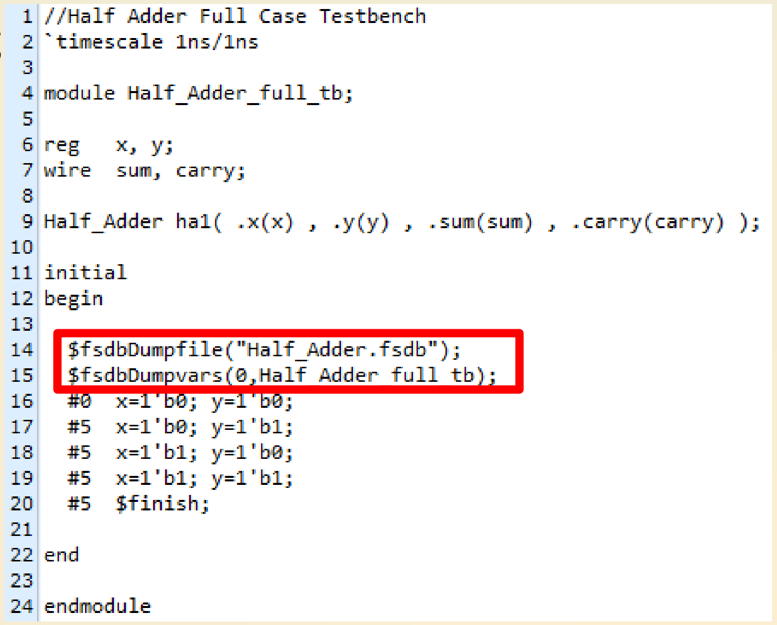
    source /usr/cad/synopsys/CIC/customexplorer.cshrc
    source /usr/cad/cadence/CIC/incisiv.cshrc
    source /usr/cad/synopsys/CIC/verdi.cshrc
    source /usr/cad/synopsys/CIC/synthesis.cshrc
    source /usr/cad/synopsys/CIC/vcs.cshrc



```
[109521121@eda359_forclass ~]$ source /usr/cad/synopsys/CIC/customexplorer.cshrc
[109521121@eda359_forclass ~]$ source /usr/cad/cadence/CIC/incisiv.cshrc
[109521121@eda359_forclass ~]$ source /usr/cad/synopsys/CIC/verdi.cshrc
[109521121@eda359_forclass ~]$ source /usr/cad/synopsys/CIC/synthesis.cshrc
Platform = amd64
[109521121@eda359_forclass ~]$ source /usr/cad/synopsys/CIC/vcs.cshrc
```

Verilog Simulation

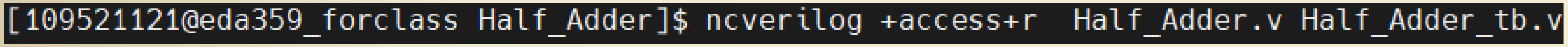
- How to use the Verilog Simulation on workstation?
  - Step2: You need to add some program syntax for testbench to generate waveform.
    - $fsdbDumpfile("your_circuit.fsdb");
    - $fsdbDumpvars(0, your_circuit _tb);



```
//Half Adder Full Case Testbench
`timescale 1ns/1ns

module Half_Adder_full_tb;

reg   x, y;
wire  sum, carry;

Half_Adder ha1( .x(x) , .y(y) , .sum(sum) , .carry(carry) );

initial
begin

  $fsdbDumpfile("Half_Adder.fsdb");
  $fsdbDumpvars(0,Half_Adder_full_tb);
  #0  x=1'b0; y=1'b0;
  #5  x=1'b0; y=1'b1;
  #5  x=1'b1; y=1'b0;
  #5  x=1'b1; y=1'b1;
  #5  $finish;

end

endmodule
```

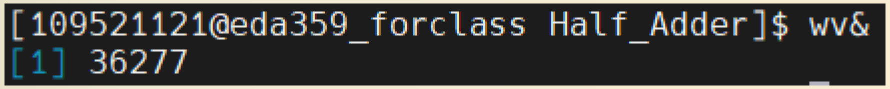
Verilog Simulation

- How to use the Verilog Simulation on workstation?
  - Step3: Key in the following command for your circuit and testbench.
    - ncverilog +access+r your_circuit.v your_circuit_tb.v



```
[109521121@eda359_forclass Half_Adder]$ ncverilog +access+r  Half_Adder.v Half_Adder_tb.v
```

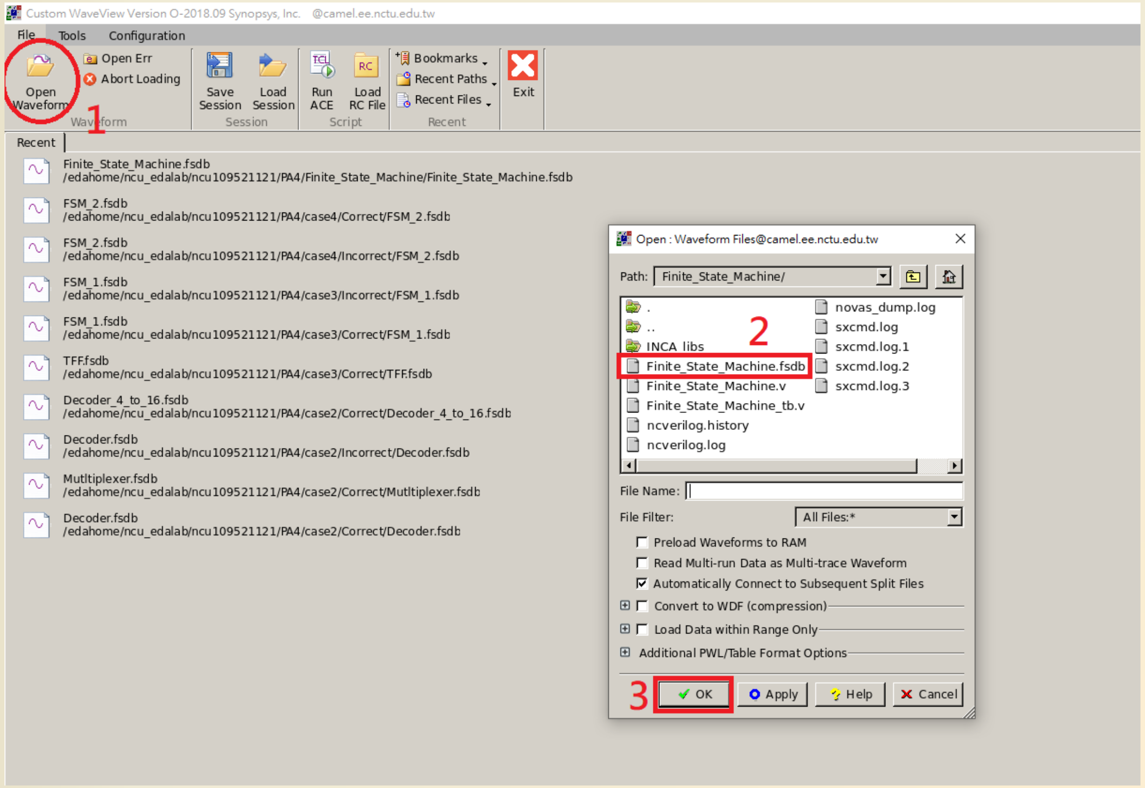
  - Step4: Open wave viewer to check the waveform
    - wv&



```
[109521121@eda359_forclass Half_Adder]$ wv&
[1] 36277
```

Verilog Simulation

- ◆How to use the Verilog Simulation on workstation?
  - ➢Step4: Open wave viewer to check the waveform
    - wv&



Custom WaveView Version O-2018.09 Synopsys, Inc. @camel.ee.nctu.edu.tw

```
Finite_State_Machine.fsdb
/edahome/ncu_edalab/ncu109521121/PA4/Finite_State_Machine/Finite_State_Machine.fsdb
FSM_2.fsdb
/edahome/ncu_edalab/ncu109521121/PA4/case4/Correct/FSM_2.fsdb
FSM_2.fsdb
/edahome/ncu_edalab/ncu109521121/PA4/case4/Incorrect/FSM_2.fsdb
FSM_1.fsdb
/edahome/ncu_edalab/ncu109521121/PA4/case3/Incorrect/FSM_1.fsdb
FSM_1.fsdb
/edahome/ncu_edalab/ncu109521121/PA4/case3/Correct/FSM_1.fsdb
TFF.fsdb
/edahome/ncu_edalab/ncu109521121/PA4/case3/Correct/TFF.fsdb
Decoder_4_to_16.fsdb
/edahome/ncu_edalab/ncu109521121/PA4/case2/Correct/Decoder_4_to_16.fsdb
Decoder.fsdb
/edahome/ncu_edalab/ncu109521121/PA4/case2/Incorrect/Decoder.fsdb
Mutltiplexer.fsdb
/edahome/ncu_edalab/ncu109521121/PA4/case2/Correct/Mutltiplexer.fsdb
Decoder.fsdb
/edahome/ncu_edalab/ncu109521121/PA4/case2/Correct/Decoder.fsdb
```

Open : Waveform Files@camel.ee.nctu.edu.tw

Path: Finite_State_Machine/

```
.                            novas_dump.log
..                           sxcmd.log
INCA_libs                    sxcmd.log.1
Finite_State_Machine.fsdb    sxcmd.log.2
Finite_State_Machine.v       sxcmd.log.3
Finite_State_Machine_tb.v
ncverilog.history
ncverilog.log
```

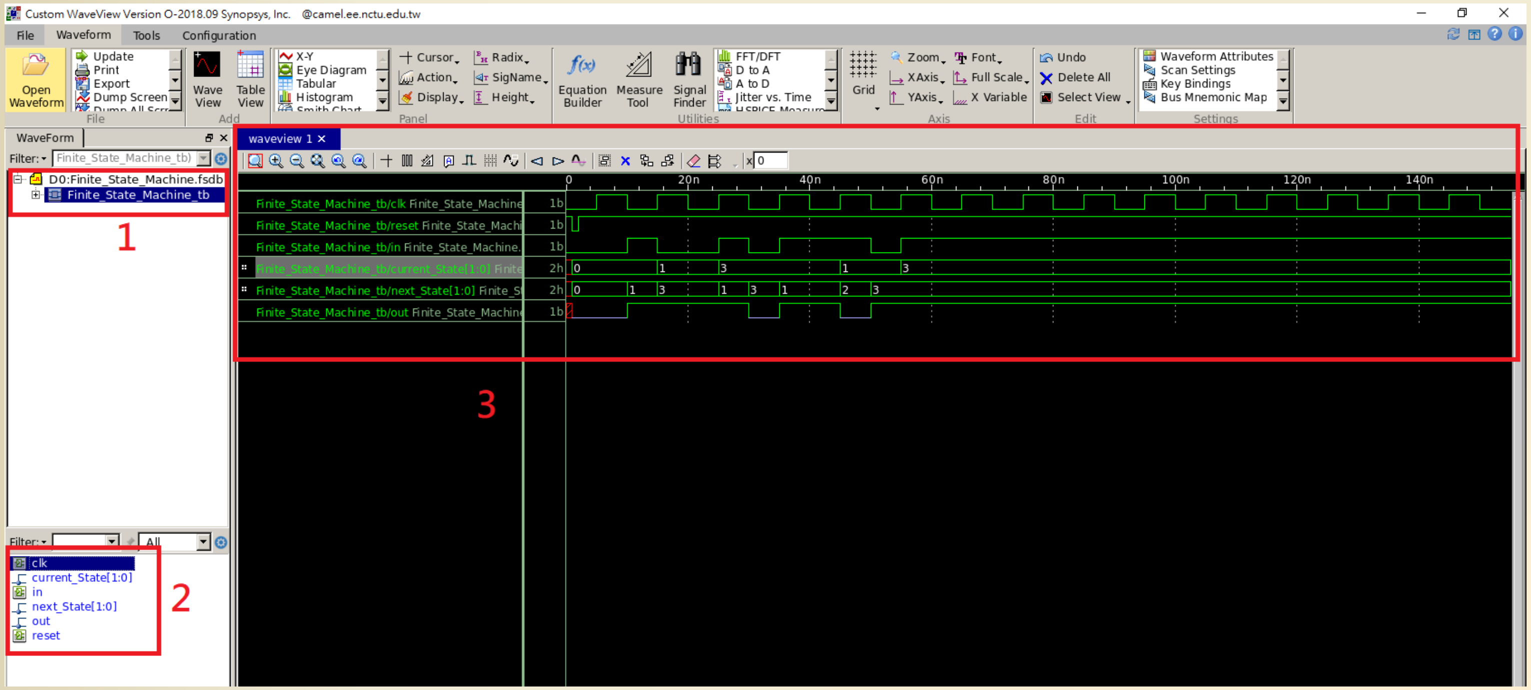
File Name:

File Filter: All Files:*

- ☐ Preload Waveforms to RAM
- ☐ Read Multi-run Data as Multi-trace Waveform
- ☑ Automatically Connect to Subsequent Split Files
- ☐ Convert to WDF (compression)
- ☐ Load Data within Range Only
- Additional PWL/Table Format Options

Verilog Simulation

- How to use the Verilog Simulation on workstation?
  - Step4: Open wave viewer to check the waveform
    - wv&



Verilog Simulation

- How to use the Verilog Simulation on workstation?
  - Step5(Option): You can write some values from Verilog for c++/c to use for automated detection.
    - https://www.cnblogs.com/oomusou/archive/2008/02/11/1066839.html

Outline

- Verilog Simulation
- Examples
  - Half Adder
  - Full Adder
  - Finite State Machine
- Cases

Half Adder

◆Half Adder Correct Behavior

X	Y	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Full Adder

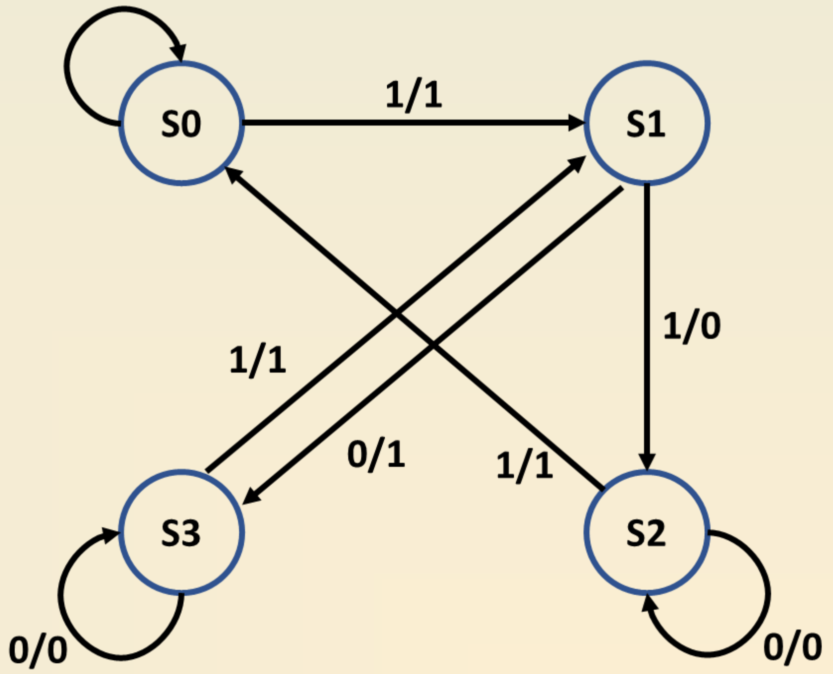
◆Full Adder Correct Behavior

Carry_in	Augend	Addend	Sum	Carry_out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Finite State Machine

◆Finite State Machine Correct Behavior

Current State	Next State		out	
	in=0	in=1	in=0	in=1
S0	S0	S1	0	1
S1	S3	S2	1	0
S2	S2	S0	0	1
S3	S3	S1	0	1



Outline

- Verilog Simulation
- Example
- Cases
  - Case 1: Comparator 4-bit
  - Case 2: Decoder
  - Case 3: FSM 1
  - Case 4: FSM 2

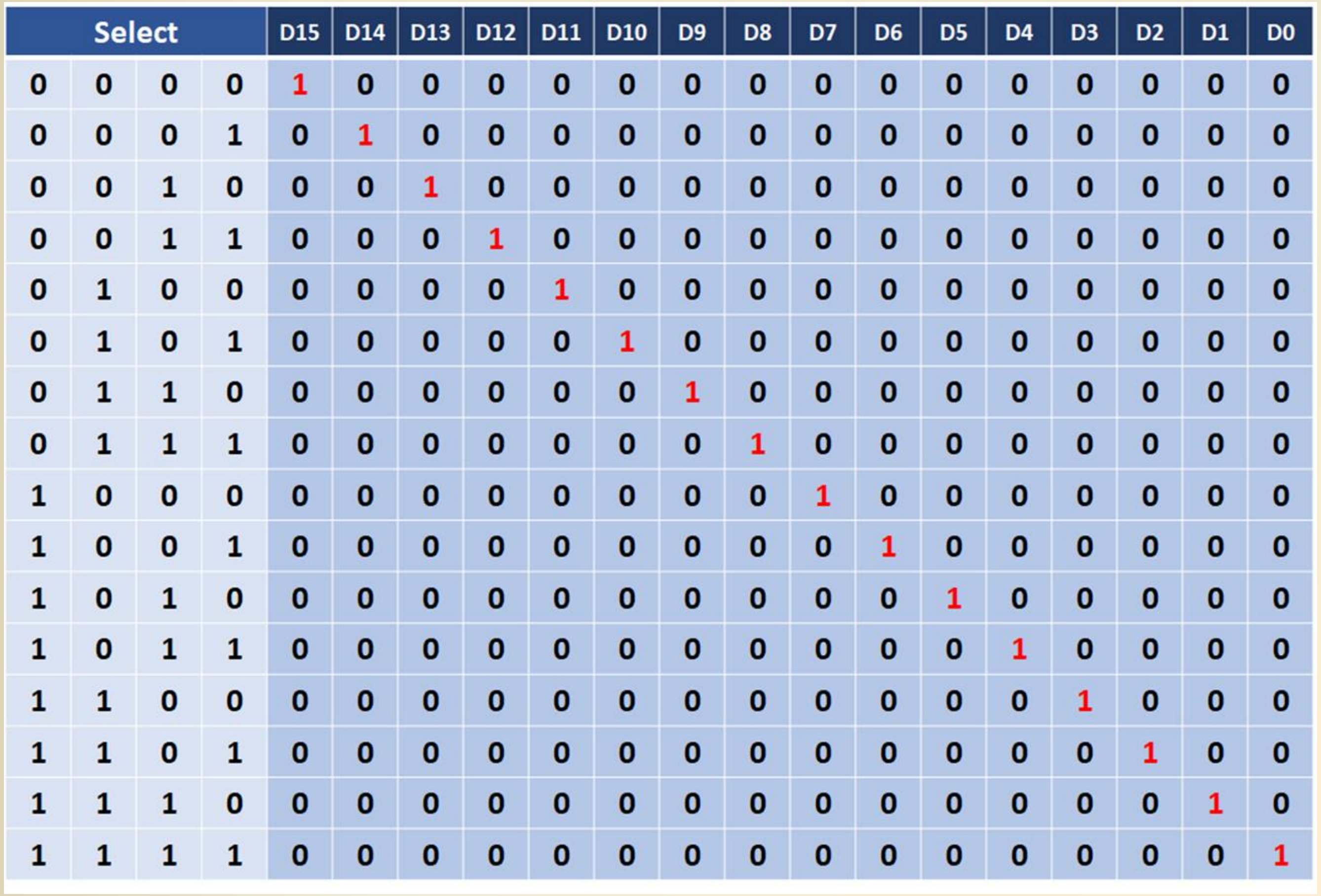
Comparator 4-bit

- Comparator
  - A<B ➡ Result: 100
  - A>B ➡ Result: 010
  - A=B ➡ Result: 001

A	B	Result
4’d3	4’d9	3’b100
4’d12	4’d5	3’b010
4’d7	4’d7	3’b001

Decoder

◆Decoder Correct Behavior

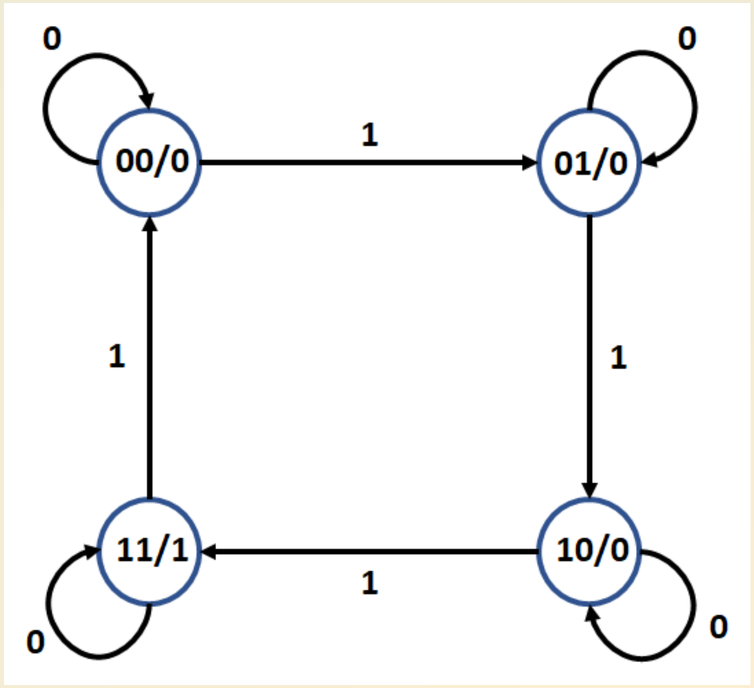


Select				D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

FSM 1

◆FSM 1 Correct Behavior

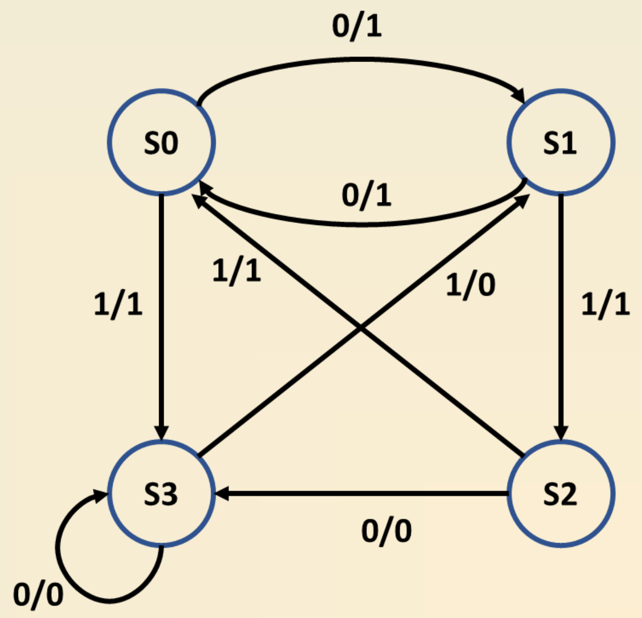
Current State		Input	Next State		Output
A	B	in	A	B	out
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	1	0	0
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	1



FSM 2

- FSM 2 Correct Behavior
  - FSM 2 can be divided into two parts, the first is normal operation mode and the second is idle mode.
    - Normal mode:

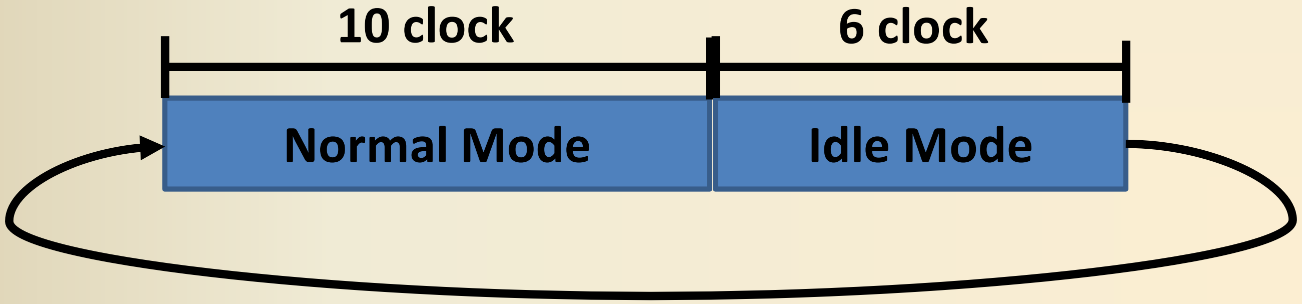
Current State	Next State		out	
	in=0	in=1	in=0	in=1
S0	S1	S3	1	1
S1	S0	S2	1	1
S2	S3	S0	0	1
S3	S3	S1	0	0



FSM 2

◆FSM 2 Correct Behavior

- ➢ FSM 2 can be divided into two parts, the first is normal operation mode and the second is idle mode.
  - Idle mode:
    - – After 10 clock periods of normal mode, the current state will remain for 6 clock periods, that is, idle mode.
  - Normal mode and idle mode will rotate with each other



Thank You.