

# Verilog Synthesis and FSMs

UCB EECS150 Spring 2010

Lab Lecture #3

# Agenda

- Logic Synthesis
- Behavioral Verilog HDL
- Blocking vs. Non-Blocking
- Administrative Info
- Lab #3: The Combo Lock
- FSMs in Verilog HDL

# Logic Synthesis

- Allows designing at a high level
  - The tool handles details
- Very good at small-scale optimization
- Synthesis tool is good at small circuits
  - Don't let it design things you can't
  - This is not software

# Behavioral Verilog

(1)

- Specifies what a circuit does
  - Not how it is built
- Most common constructs:
  - “always @ ...”
    - always @ \*
    - always @ (posedge Clock)
  - assign Y = ...

# Behavioral Verilog

(2)

- always @ \*
- Used to describe **combinational** logic.
- Can cause the nastiest errors
  - Make sure no latches are generated
- always @ (posedge Clock)
- Used to infer a register
- Keep these nice and short
  - Even an accumulator is too much

# Wire vs. Reg

- Wire
  - Logical connection of circuit elements
  - Cannot be assigned in an always block.
- Reg
  - **NOT** a Register
  - A variable used in a circuit description.
  - Can be assigned in an always block.

# Blocking vs. Non-Blocking (1)

- Blocking assignment “=”
  - Guarantees sequential assignment
  - Often costs unwanted hardware
  - **ONLY** use in “always @ \*”
    - Else simulation and synthesis won’t match
- Non-Blocking “<=”
  - All assignments happen simultaneously
  - **ONLY** in “always @ (posedge Clock)”

# Blocking vs. Non-Blocking (2)

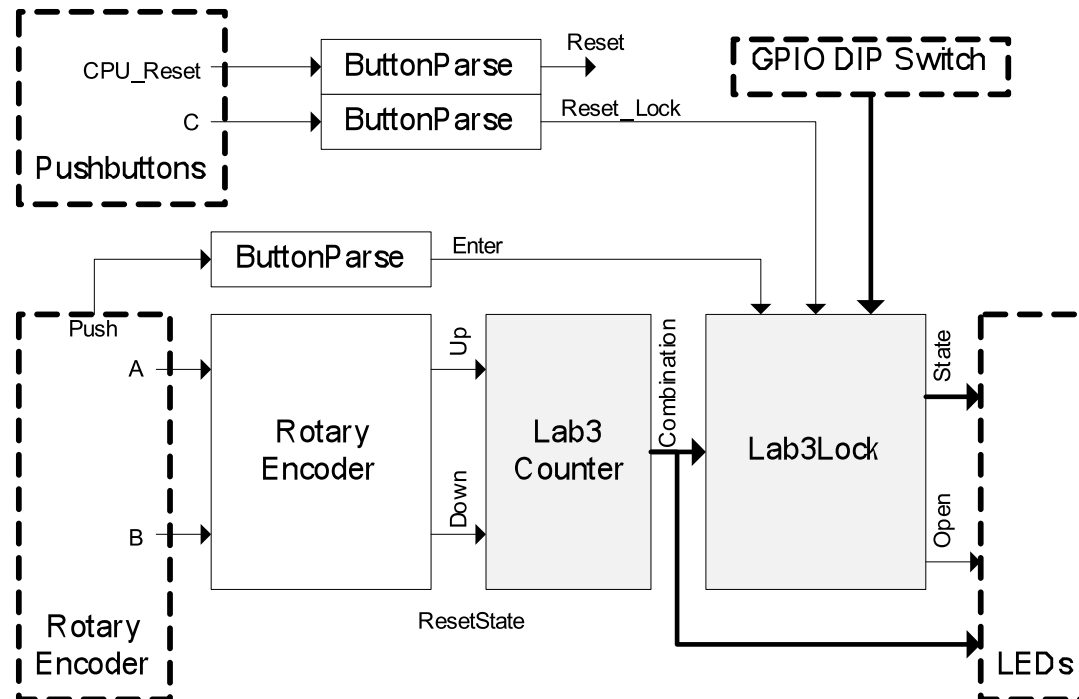
Verilog Fragment	Effect
<pre>always @ ( * ) begin     b = a;     c = b; end</pre>	<p>C = B = A</p>
<pre>always @ (posedge Clock) begin     b &lt;= a;     c &lt;= b; end</pre>	<p>B = Old A C = Old B</p>



# Administrivia

- SVN now available
  - Use your UNIX passwords
  - E-mailed submissions no longer accepted
  - Homework 2 should have already been submitted!
- Lab Checkoff
  - Due at the beginning of you lab period

# Lab #3: The Combo Lock (1)



- Used to control entry to a locked room  
4bit, 2 digit combo (By Default 0x2, 0x3)  
Set code to 0010, Press Enter  
Set code to 0011, Press Enter  
Lock Opens (Open = 1)

# Lab #3: The Combo Lock (2)

**READ THE LAB**

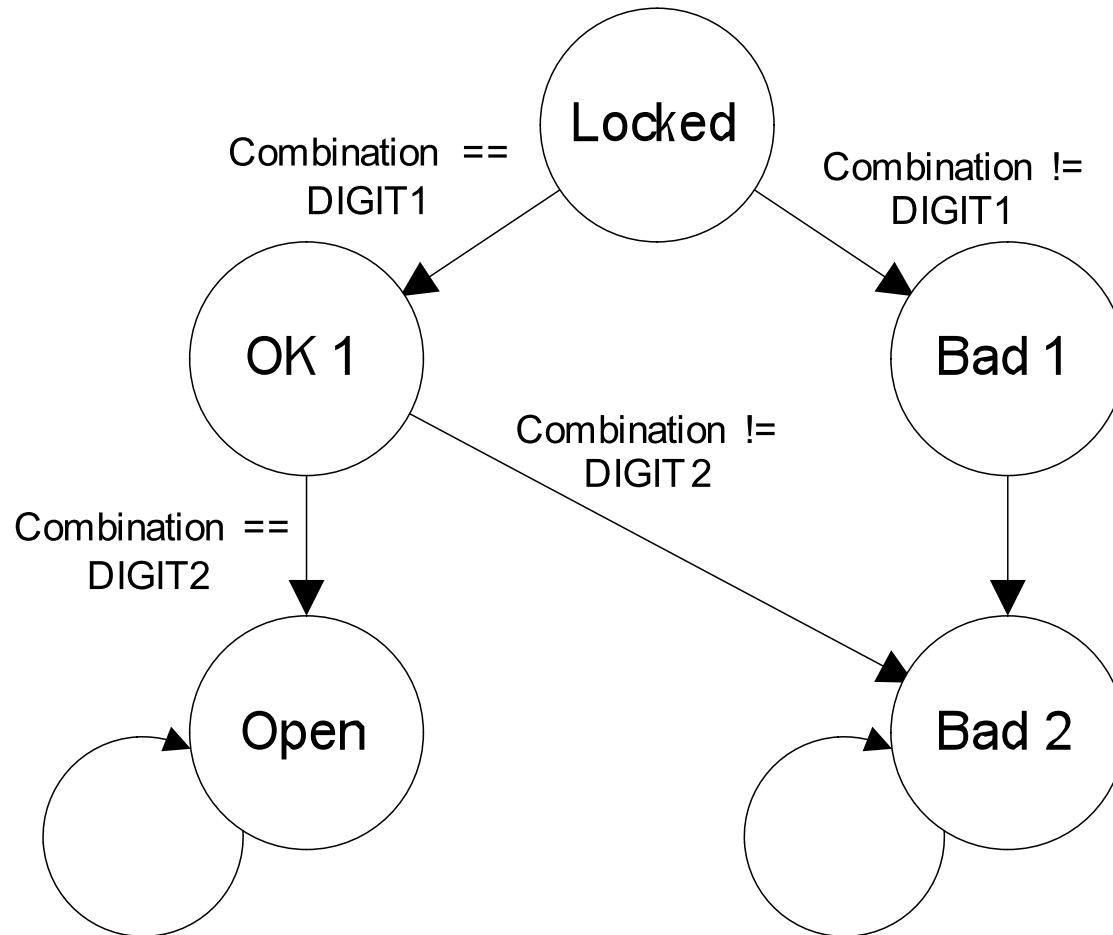
**DO THE PRELAB**

# Lab #3: The Combo Lock (3)

- We will provide the framework
- You will build two modules:
  - Lab3Lock
    - A Moore FSM
  - Lab3Counter
    - An up-down counter
    - Somewhat similar to an accumulator
- Use behavioral Verilog

# Lab #3: The Combo Lock

(4)

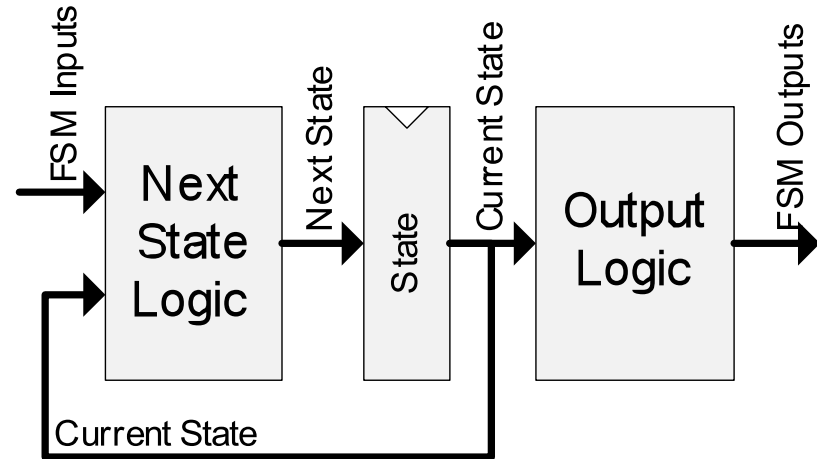


# Lab #3: The Combo Lock (5)

- Use LEDs and Buttons to debug
  - Simple
    - Hard to mess this up
  - Great way to show state
  - Low overhead, compared to other tools
- But:
  - Can't see fast events
  - Limited number
  - No timing information

# FSMs in Verilog

- Next State logic
  - “always @ \* ” block with “case”
- Register
  - “always @ (posedge Clock)”
- Output logic
  - Continuous “assign”



# Acknowledgements & Contributors

Slides developed by Kyle Wecker & John Wawrzynek (2/2010).

This work is based closely on slides by:

Chris Fletcher & Ilia Lebedev (2008-2009)

Greg Gibeling (2003-2005)

This work has been used by the following courses:

- UC Berkeley CS150 (Spring 2010): Components and Design Techniques for Digital Systems