

Structural Verilog

UCB EECS150 Fall 2010

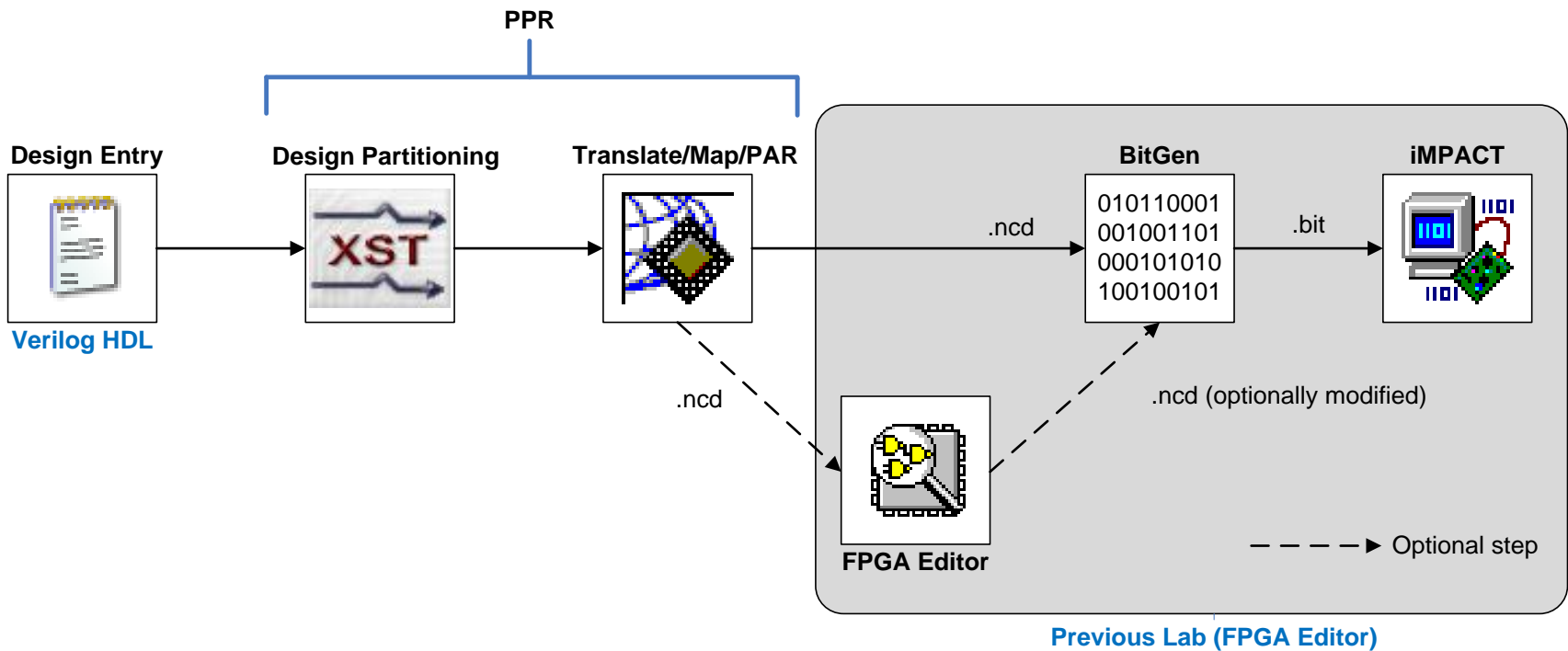
Lab Lecture #2

Agenda

- CAD Flow Extension
- Verilog
- Structural Verilog
- Administrative Info
- Lab #2: A Structural Accumulator
 - Circuit
 - Testing
 - Analysis of resource usage and timing
 - PreLab
- Questions?

CAD Flow Extension

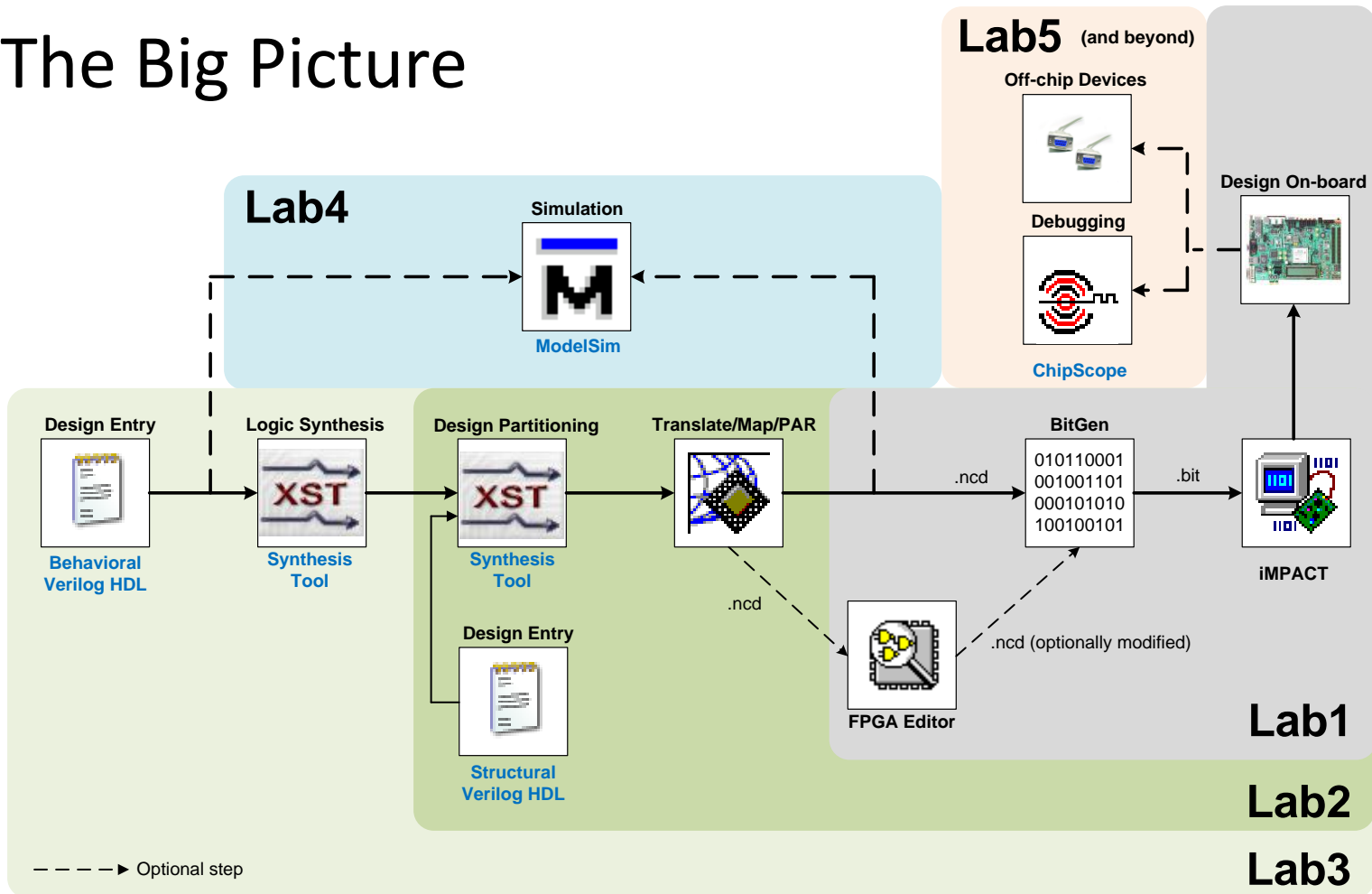
(1)



CAD Flow Extension

(2)

- The Big Picture



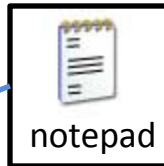
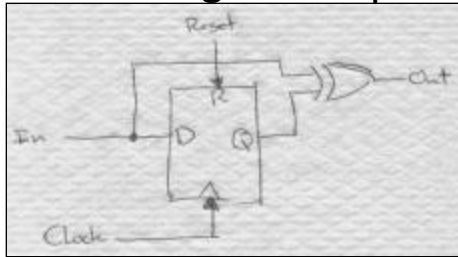
Verilog

(1)

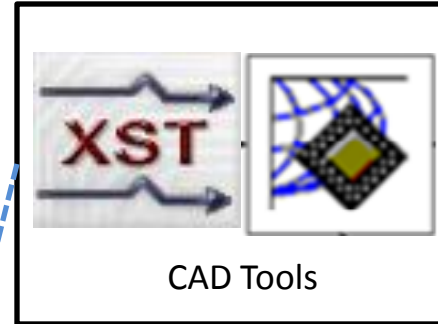
- What's an HDL?
 - Textual Description of a Circuit
 - Human and Machine Readable
 - Hierarchical
 - Meaningful Naming
- NOT A PROGRAM
 - Describe what the circuit IS
 - Not what it DOES

Verilog + CAD

design on napkin



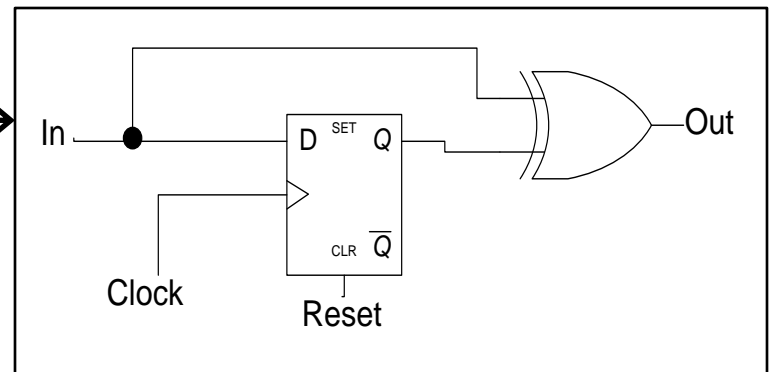
notepad



CAD Tools

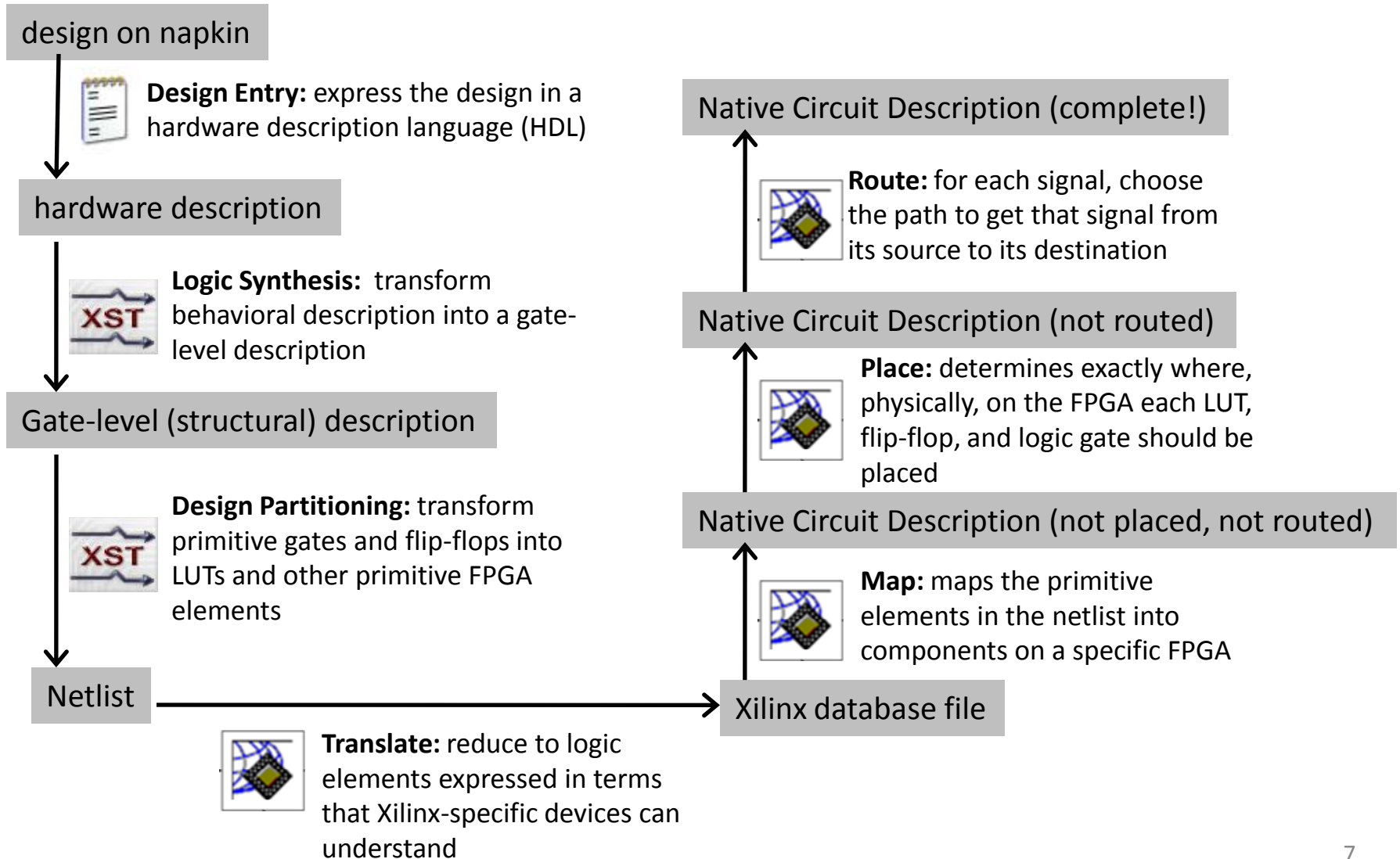
```
assign Out = Q ^ In;  
always @ (posedge Clock) begin  
    if (Reset) Q <= 1'b0;  
    else Q <= In;  
end
```

textual hardware description





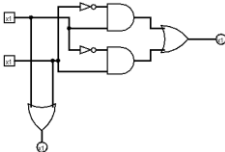
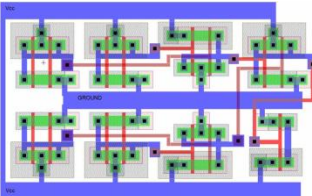
placed & routed design

Verilog + CAD



Digital Design Productivity, Gates/Week

Source: DataQuest

- Behavioral HDL  2K-10K
- RTL HDL  1K-2K
- Gates  100-200
- Transistors  10-20

Structural Verilog

(1)

- Verilog Subsets
 - Structural: primitive gates + modules
 - Gate level design
 - **You will ONLY use Structural Verilog in this lab**
 - Dataflow: compact boolean expressions
 - More compact expression of structural Verilog
 - Behavioral: abstract syntax
 - Timing nuances
 - You will see this starting next lab

Structural Verilog

(2)

- Structural 2:1 Mux example

```
module Mux21(A, B, S, Out);  
  input wire A, B, S;  
  output wire Out;  
  
  wire notS, ATemp, BTemp;  
  
  not invertS(notS, S);  
  and and(ATemp, A, notS),  
    andB(BTemp, B, S);  
  or result(Out, ATemp, BTemp);  
endmodule
```

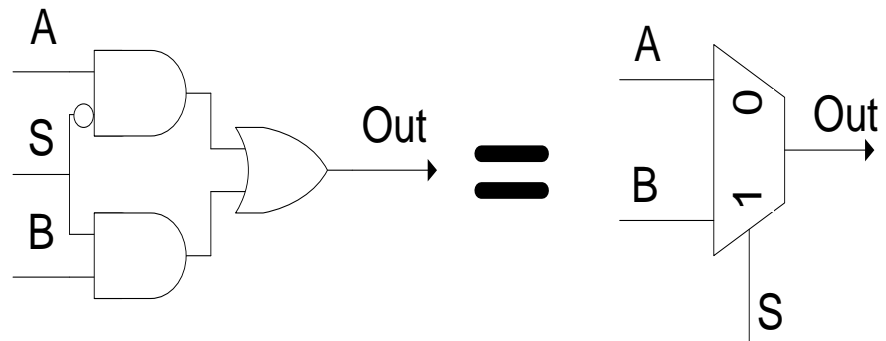
Key

Module wrapper

Input/Output wire declarations

Wire declarations

Gates



Administrative Info

(1)

- Homework submission
- Lab lecture conflicts
- Card key access
- Check-off procedure
- Questions?

Lab #2

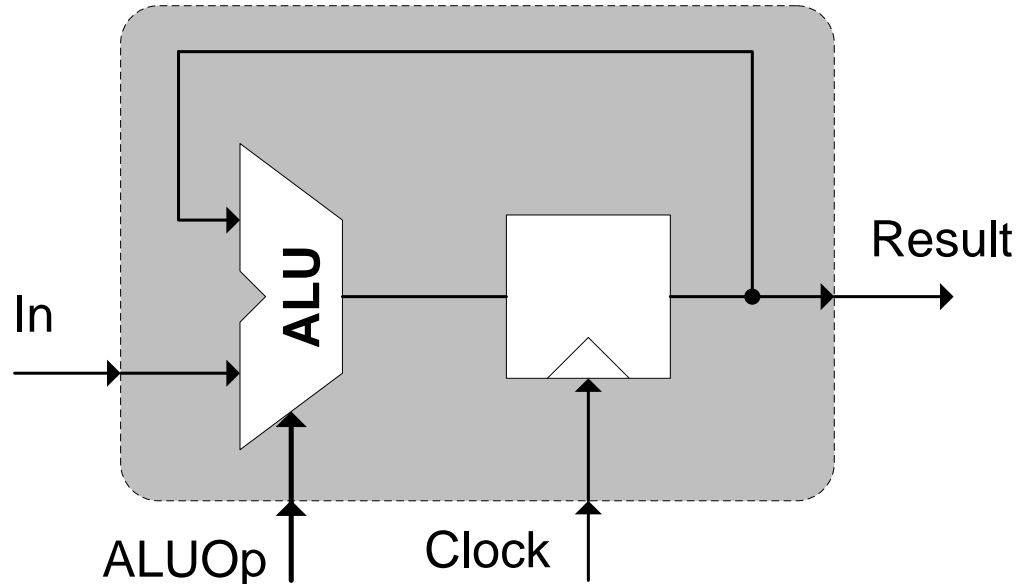
(1)

- Build a structural Accumulator
 - Work with a real design
 - Write parameterized Verilog
- Debugging
 - Synplicity RTL/Technology schematic
- Analysis
 - Resource consumption
 - Timing

Lab #2

(2)

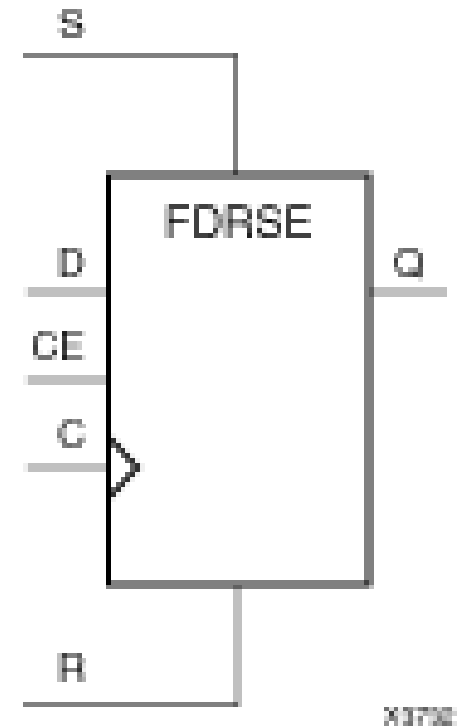
- Structural Accumulator
 - ALU
 - 'FDRSE' Xilinx primitive flip-flop



Lab #2

(4)

- FDRSE
 - Xilinx primitive
 - D-type flip-flop
 - Instantiated like a simple module
 - Specific Set/Reset characteristics
 - “Read all about it!” → [virtex5_hdl.pdf](#)
 - It’s part of the PreLab!



Lab #2

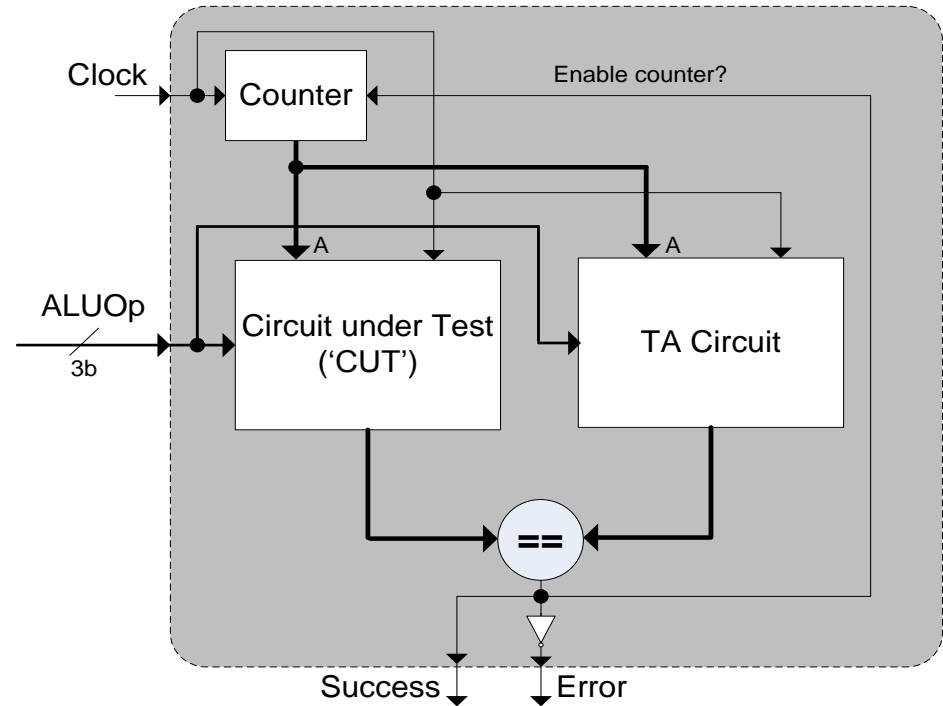
(5)

- Accumulator
 - We give you port specification
 - You will implement the rest of the circuit
- Use code examples
 - Mux21: Structural Verilog (gates, wires)
 - ALU: generate statements
- Abide by our interfaces!

Lab #2

(6)

- HW test harness
 - TA Accumulator vs. your Accumulator
 - Check all input combinations
 - Signal error



Lab #2

(7)

- Circuit Analysis

- Resource Usage

- Accumulator(width) = how many LUTs / SLICES?
 - generate allows you to experiment

- Timing

- Locate nets → **“Technology Schematic”**
 - Calculate delay on the nets → **FPGA Editor**

Lab #2

(8)

- PreLab
 - Read specified material
 - [Virtex-5 Libraries Guide for HDL Designs](#) (FDRSE section)
 - Design your ALUBitSlice and Accumulator
 - Write all of your Verilog
- Lab starts at debugging phase
 - Assumption:
you have written all of your Verilog ahead of time

Acknowledgements & Contributors

Slides developed by Brandon Myers & John Wawrzynek (1/2010).

This work is based closely on slides by:

Chris Fletcher (2008-2009)

Greg Gibeling (2003-2005)

This work has been used by the following courses:

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