Structural Verilog	
Structural verilog	
UCB EECS150 Spring 2010  Lab Lecture #2	
Lab Lecture #2	
Agenda	
<ul><li>CAD Flow Extension</li><li>Verilog</li></ul>	
<ul> <li>Structural Verilog</li> <li>Administrative Info</li> <li>Lab #2: A Structural Accumulator</li> </ul>	
- Circuit - Testing	
<ul> <li>Analysis of resource usage and timing</li> <li>PreLab</li> </ul>	
Questions?  UCB EECS150 Spring 2010, Lab Lecture #2  2	
CAD Flow Extension (1)	
ppr	
Design Entry  Design Partitioning  Translatin/AppPAR  BitGen  BitCon  District Control of the Co	
Verificial MOL.  And (optionally modified)	
FPGA Editor  Previous Lab (FPGA Editor)	
UCB EECS150 Spring 2010, Lab Lecture #2 3	

## CAD Flow Extension

(2)

The Big Picture

Lab4

Lab2

Lab2

Lab2

Lab2

Lab2

Lab2

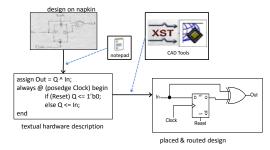
Lab3

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

UCB EECS150 Spring 2010, Lab Lecture #2

## Verilog + CAD



UCB EECS150 Spring 2010, Lab Lecture #2

## Verilog + CAD design on napkin Design Entry: express the design in a hardware description language (HDL) Native Circuit Description (complete!) Route: for each signal, choose the path to get that signal from its source to its destination Logic Synthesis: transform ve Circuit Description (not routed) behavioral description into a gate-level description Place: determines exactly where, physically, on the FPGA each LUT, flip-flop, and logic gate should be placed Gate-level (structural) description Design Partitioning: transform primitive gates and flip-flops into LUTs and other primitive FPGA Native Circuit Description (not placed, not routed) Map: maps the primitive elements in the netlist into components on a specific FPGA Xilinx database file Translate: reduce to logic elements expressed in terms that Xilinx-specific devices can understand UCB EECS150 Spring 2010, Lab Lecture #2 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

UCB EECS150 Spring 2010, Lab Lecture #2

Structural Verilog	(2)	
Structural 2:1 Mux example		
Key module Mux21(A, B, S, Out); Module wrapper	r	
input wire A, B, S; Input/Output wire Out; Wire declaration wire nots, ATemp, BTemp; Gates	ire declarations	
not invertS(notS, S); and and (ATemp, A, notS), andB(BTemp, B, S); Out	A Out	
or result(Out, ATemp, BTemp); endmodule	B	
UCB EECS150 Spring 2010, Lab Lecture N2	10	
Administrative Info	(1)	
Homework submission		
Lab lecture conflicts		
Card key access		
<ul><li>Check-off procedure</li><li>Questions?</li></ul>		
Questions:		
UCB EECS150 Spring 2010, Lab Lecture #2	11	
Lab #2	(1)	
Build a structural Accumulator		
Work with a real design  Write parameterized Veriles.		
<ul><li>Write parameterized Verilog</li><li>Debugging</li></ul>		
<ul> <li>Synplicity RTL/Technology schematic</li> </ul>		
<ul><li>Analysis</li><li>Resource consumption</li></ul>		
– Timing		
UCB EECS150 Spring 2010, Lab Lecture #2	12	

## (2) Lab #2 • Structural Accumulator - 'FDRSE' Xilinx primitive flip-flop Result Clock ALUOp Lab #2 (3) • ALU - We provide the Verilog for N-bit version - You must implement the 1-bit ALU model - Must support our ALUOp - Supports: +, -, &, |, ~, ALUOp pass-through (7 operations) UCB EECS150 Spring 2010, Lab Lecture #2 Lab #2 (4)• FDRSE FDRSE - Xilinx primitive Q - D-type flip-flop CE · Instantiated like a simple module c • Specific Set/Reset characteristics – "Read all about it!" → virtex5\_hdl.pdf • It's part of the PreLab!

UCB EECS150 Spring 2010, Lab Lecture #2

Lab #2	(5)	
<ul> <li>Accumulator         <ul> <li>We give you port specification</li> <li>You will implement the rest of the circuit</li> </ul> </li> <li>Use code examples         <ul> <li>Mux21: Structural Verilog (gates, wires)</li> <li>ALU: generate statements</li> </ul> </li> <li>Abide by our interfaces!</li> </ul>		
UCB EECS150 Spring 2010, Lab Lecture #2	16	
Lab #2	(6)	
HW test harness     TA Accumulator vs. your Accumulator     Check all input combinations     Signal error		
UCB EECS150 Spring 2010, Lab Lecture #2	17	
Lab #2	(7)	
	(7)	
Circuit Analysis  — Resource Usage  • Accumulator(width) = how many LUTs / SLICEs?  • generate allows you to experiment		
<ul> <li>Timing</li> <li>Locate nets → "Technology Schematic"</li> <li>Calculate delay on the nets → FPGA Editor</li> </ul>		
UCB EECS150 Spring 2010, Lab Lecture #2	18	

Lab #2		(8)		
PreLab     Read specifier     Victor E Library	d material aries Guide for HDL Designs (FDRSE sec	rtion)		
	LUBitSlice and Accumulator	ction		
• Lab starts at d				
<ul><li>Assumption:</li><li>you have writt</li></ul>	ten all of your Verilog ahead of time			
	UCB EECS150 Spring 2010, Lab Lecture #2	19		
Acknowled	gements & Contributo	ors		
	andon Myers & John Wawrzynek (1/2010).			
This work is based clos Chris Fletcher (20	008-2009)			
Greg Gibeling (20				
	ed by the following courses: 2010): Components and Design Techniques for Digital Systems			
	UCB EECS150 Spring 2010, Lab Lecture #2	20		