

CSE 372 Final Exam

Spring 2008

Name: _____

Problem 1: ____ (of 10) Hardware, Software, FPGAs

Problem 2: ____ (of 10) Verilog

Problem 3: ____ (of 10) Labs

Problem 4: ____ (of 12) Design Flows

Problem 5: ____ (of 8) Testing and Verification

Problem 6: ____ (of 5) Joel Emer's Talk

Final Score: ____ (of 55)

This is a closed book exam. There are 55 total points, 1 point per minute ... plus 5 minutes. Budget your time accordingly.

1 “REAL HARDWARE” and FPGAs (10 points)

a. (4 points) For a given design, what is the difference between a “real hardware” implementation and an FPGA implementation?

The “real hardware” is manufactured using a custom mask set that directly implements the design. The mask set that implements the FPGA doesn’t implement any one design. The design is programmed onto it.

What advantage does the FPGA implementation have over the “real hardware” one?

FPGAs have much lower startup costs and fabrication turnaround times.

What advantage does the “real hardware” implementation have over the FPGA one?

Real hardware has better unit characteristics: lower cost, lower power, higher performance, etc.

Given these advantages, what kinds of applications are FPGAs suited to?

FPGAs are suited for experimental or low volume hardware products.

b. (1 points) How many CLBs does the Xilinx Virtex-II FPGA have? 3,000? 30,000? 300,000? 3,000,000?

30,000

c. (1 point) What is the peak clock speed of the Xilinx Virtex-II: i) 10 MHz, ii) 100 MHz, iii) 1 GHz, iv) 3 GHz?

100MHz

d. (2 points) Which of the following elements are found on a Xilinx Virtex-II? i) a 4-input 1-output lookup table, ii) an 8-input 1-output lookup table, iii) a flip-flop, iv) a fast carry-select adder, v) an 18 by-18 multiplier, vi) a PowerPC 405 processor, vii) block-RAM?

4-input, 1-output LUT, flip-flop, multiplier, PowerPC core and BRAM.
There are no fast carry-select adders or 8-input 1-output LUTs.

e. (2 points) Why do delay and area trends on FPGAs not match those on “real hardware”?

The CLB structure and interconnect introduce “discretization effects” that aren’t present in custom hardware. For instance, in custom hardware, a 4-input function is 2X more expensive than a 2-input function, but in an FPGA they have the same cost. Similarly, a 6-input function is 1.5X more expensive than a 4-input function, but in hardware, it is 7X more expensive.

2 VERILOG (10 points)

- a. (2 points) What are two differences between an HDL like Verilog and an SDL like C?

An SDL like C is only structural, it describes the program implementation directly. An HDL has both structural features and behavioral features, which describe how a piece of hardware acts but not its direct implementation. There are functional SDLs, like ML and Haskell, but C is not one of them.

Both HDLs and SDLs can be “synthesized”, an HDL into a piece of hardware and an SDL into a software executable. An HDL, however, can also be “simulated”.

- b. (1 point) In Verilog, why do you have to give a name to every instance of a user-defined module, but not to every instance of a primitive (e.g., an and gate)?

Because you cannot “drill into” a primitive.

- c. (4 points) What do the following Verilog statements accomplish?

```
wire [7:0] a, b, c; wire [15:0] e; wire f, g;
assign e = {{8{b[7]}}, b};
```

sign-extends b

```
assign g = |c;
```

ORs all the bits in c and assigns the result to g. At a higher level, g is set to 1 if c is not 0.

```
assign c = f ? a : b;
```

This is a multiplexer

```
g = b ^ c;
```

This is an error, it's an illegal procedural assignment on a wire. It's also a bitwise XOR of 8-bit vectors into a 1 bit wire.

- d. (3 points) Write the verilog for an 8-bit register with synchronous write enable. You may use behavioral verilog.

```
module reg_8b (out, in, we, clk);
    output [7:0] out;
    input [7:0] in;
    input we;
    input clk;
    reg [7:0] out;
    always @(posedge clk)
        if (we) out = in;
endmodule
```

3 LABS (10 points)

a. (3 points) Consider a 4-bit multiplier implemented as a chain of ripple-carry adders. Fill in the blanks for this verilog code.

```
module multiplier_4b_chain_rc (p, a, b);
    input [3:0] a, b; output [3:0] p;
    wire [3:0] a0 = b[0] ? a : 4'h0;
    wire [3:0] a1 = b[1] ? {a[2:0], 1'b0} : 4'h0;
    wire [3:0] a2 = b[2] ? {a[1:0], 2'b00} : 4'h0;
    wire [3:0] a3 = b[3] ? {a[3], 3'b000} : 4'h0;
    wire [3:0] t0, t1; wire c0, c1, c2;
    adder_4b_rc adder0 (t0, c0, a3, a2, 1'b0);
    adder_4b_rc adder1 (t1, c1, t0, a1, 1'b0);
    adder_4b_rc adder2 (p, c2, t1, a0, 1'b0);
endmodule
```

Does it matter in which order you perform the additions? Why?

It does matter. If you add the more shifted operands first, your multiplier chain will have a lower delay. You will take the delay of the carry chain of the last addition in parallel with the most significant bit of the previous additions, not in series with them.

b. (4 points) Write the verilog for an 8-bit barrel left shifter. The interface for the shifter is. You may use behavioral verilog, but not the << operator itself.

```
module sll_8b (out, in, sh);
    output [7:0] out;
    input [7:0] in; input [2:0] sh;
    wire [7:0] s1 = (sh[0]==1'b1) ? {in[6:0], 1'b0} : in;
    wire [7:0] s2 = (sh[1]==1'b1) ? {s1[5:0], 2'b0} : s1;
    assign out = (sh[2]==1'b1) ? {s2[3:0], 4'b0} : s2;
endmodule
```

c. (2 points) What is the difference between datapath control and pipeline control?

Datapath control controls the datapath muxes and write-enable signals, it is a function of the opcode of the instruction in a particular stage. Datapath control travels down the pipeline in the pipeline latches.

Pipeline control controls the pipeline latches themselves (i.e., it implements stalls and flushes). It is a function of the instructions in each latch, branch prediction, etc.

d. (1 point) What is the combination to the door of the K-lab?

4-2-5-1

4 DESIGN FLOWS (12 points)

a. (2 points) What is a netlist?

A netlist is a canonical hardware description roughly on the level of software object or binary code. It consists of low-level logical functions (e.g., 3-input 1-output or NANDs) and their interconnectivity.

What process produces a netlist?

Synthesis.

b. (2 points) What are standard cells?

Standard cells are mask-set “lego” blocks for simple logical functions. They are combined to create complete mask sets.

What is the benefit of standard cells?

Standard cells raise the design abstraction level for custom hardware. You don’t need to be a VLSI expert to lay down standard cells. Cells can also be layed out automatically.

c. (4 points) What is a soft core?

A softcore is a portable/synthesizable processor design, as opposed to a physical processor.

What are two formats in which a soft core can be distributed?

HDL or netlist.

What is the advantage of each distribution format?

The HDL format is easier to modify and extend. The netlist format protects the intellectual property of the vendor. HDL can be thought of as “source code” and netlist as “object code”. Vendors typically charge more for HDL softcores.

d. (4 points) What is an ASIC?

An ASIC is a direct hardware implementation of an application-specific function. An ASIC is contrasted with a software implementation of the same function on a general-purpose processor.

What is an “embedded” processor?

An embedded processor is one that is in a device that doesn’t have a traditional computer interface, e.g., a computer in a digital camera.

What are two characteristics of embedded computers that make ASICs a good choice?

Embedded processors typically have strict power and cost requirements. They also typically have “closed” software environments, i.e., they come with a fixed suite of pre-loaded software. The user cannot install arbitrary software and even upgrades (if they are possible at all) can only be done in a restricted way. The combination of known software and strict implementation requirements makes application-specific designs compelling.

5 TESTING AND VERIFICATION (8 points)

a. (4 points) What is the difference between testing and verification?

Testing provides statistical confidence that a design is “correct” by exercising all of its functionality in an attempt to find bugs. If no bugs are found then the design is assumed to be correct for all intents and purposes, i.e., even if it contains bugs, those bugs must be really rare and obscure and unlikely to come up during real usage. Verification provides absolute confidence that a design is correct using formal techniques to “prove” that it is.

What is an advantage of verification over testing?

Verification is preferred because it provides absolute guarantees.

What is an advantage of testing over verification?

However, verification of a state machine is exponential in the number of transitions, making this approach intractable for large designs.

Which approach do companies actually use?

Both. They verify whatever they can and test whatever they can't.

b. (2 points) Why would a company release a product with known bugs for which there are known fixes?

The bug may not be severe (i.e., it could impact very few people or cause mere inconvenience) and the fix may have unknown side-effects and interactions that cannot be tested in time.

c. (2 points) What form of testing does hardware require that software does not? Why?

Hardware requires additional “unit” testing. In both hardware and software, the “master design” is tested rigorously. However, in software subsequent copies of this master are not tested, because they are all assumed to be functionally identical “virtual” replicas. However, hardware copies of the master are “physical” replicas which may not be functionally identical. For hardware, each copy must be tested in addition to the master.

6 JOEL EMER'S TALK (5 points)

a. (3 points) What is the difference between a hardware model and a hardware prototype?

A prototype matches the target design structurally, a model only matches the performance of the target design, but not its actual structure.

Which approach is Joel Emer's group taking to simulate hardware designs?

Joel's group is modeling (rather than prototyping) future designs.

Why?

Models are much smaller and can fit onto a single FPGA.

b. (1 point) How much faster is FPGA-based simulation than comparable software simulation: i) 10X faster, ii) 100X faster, iii) 1,000X faster, iv) 1,000,000X faster?

1,000X faster

c. (1 point) What is a "chicken bit"?

A chicken bit is a bit that can be used to disable a feature (without actually taking it out of the design) if it is determined that this feature is either broken somehow or hurts performance.