EECS 150 Fall 2003

Checkpoint1 Part2
10/3/2003

Greg Gibeling

Today

- Finer Points of SDRAM
- Common Mistakes
  - A little verilog style
  - Announcements
  - Synplify Warnings
  - Designers Toolbox
    - System building blocks
    - Block diagrams

Finer Points of SDRAM (1)

- Errata
  - A word is 32 bits
  - Your testbench needs two SDRAM instances (there are two SDRAM chips)
  - The write FIFO needs only 4 words before the first write
  - Programmable Delay/Time Select is in 10 msec units
  - Please read verilog files

Finer Points of SDRAM (2)

- Write and Read are independent
  - SDRAM controller is also a bus arbiter
  - Decides whether to do a read or a write
  - Schedules access to the data bus
    - Can't read and write on the same cycle
    - But read and write requests might come in at the same time
  - This will be necessary later in the project

Common Mistakes (1)

- Verilog Style
  - Remember Lab Lecture #4?
  - WE CANNOT DEBUG POOR VERILOG
  - (Neither can you)
  - Please remember:
    - Always @ Posedge <= NonBlocking
    - Always @ = Blocking
    - You cannot substitute one for the other

Common Mistakes (2)

- If you don't know what circuit you want how should synplify know?
- Verilog meant for simulation
  - Not everything synthesizes
- Simulation vs Circuit
  - For Loops
    - Initial blocks
    - # for delays
  - Counter
    - Reset
    - Properly sized counter
  - Integers
    - Use wires or regs
Announcements

- Deadlines
  - Checkpoint1
    - Part1: Should be done by Oct 10\textsuperscript{th}
    - Part2: MUST BE DONE BY OCT 17\textsuperscript{th}
  - Checkpoint deadlines will be FINAL
- Project Design Review
  - Major part of checkpoint2
  - Start thinking about this early

Synplify Warnings (1)

- Synplify Warnings
- Knowing these will save days of work
- Incomplete Sensitivity
  - ModelSim will use the sensitivity list
  - Synplify pretty much ignores it

Synplify Warnings (2)

```verilog
input Clock;
reg [31:0] Count;
// Counter
always @ (posedge Clock)
  Count <= Count + 1;

input [15:0] A, B;
output [31:0] Sum;
output COut;
// Adder
always @ (A or B)
  {Sum, COut} = A + B;
```

Synplify Warnings (3)

```verilog
Latch Generated
input [1:0] select;
input A, B, C;
output Out;
reg Out;
// Mux
always @ (select or A or B or C)
begin
  case (select)
    2'b00: Out = A;
    2'b01: Out = B;
    2'b10: Out = C;
    default: Out = 1'bx;
  endcase
end
```

Synplify Warnings (4)

- Combination Loop
- Must remove the loop or add a register
- Synplify wont show you the loop

Synplify Warnings (5)

- FPGA\_TOP always has warnings
  - Undriven Input
  - Unconnected Output
  - These are truly unneeded pins
- In your modules these are a problem
  - Synplify will optimize your design
  - Unconnected modules removed
Synplify Warnings (6)
- Synplify warnings will result in lost points
- Getting rid of warnings will ease debugging
- Warnings are the only syntax check

Project Comments
- We know it’s a lot of work
- We’re still fine tuning the workload
  - Deadlines are fixed
  - Work involved is not (we may give you parts of checkpoints)
- Please read documents carefully before you ask us how to do it

Design Review
- Show Design to TA
- Explaining design will help you understand in
- High Level Block Diagrams
- State Transition Diagrams
  - We can't read your verilog easily
  - Clean block diagrams will save you lots of time

Basic Digital Building Blocks
- Combinational (Unclocked, Logic)
  - AND/NAND/OR/NOR/NOT/XOR/XNOR
  - Comparator
  - Adder
  - Next State/Output Functions
  - NO MULTIPLY OR DIVIDE (Shift Instead)
- Register (Sequential, Clocked)
  - Shift Register
  - Counter
  - FIFOs
  - Memory
  - State Register

Combinational Blocks
- Comparator
  - Equality is cheap
  - Greater-than/Less-than are EXPENSIVE

```

assign equal = (A == B);
```

Sequential Blocks (1)
- Shift Registers
  - Parallel to Serial
  - Serial to Parallel
Sequential Blocks (2)

- Counter
  
  ```
  input Clock;
  reg [31:0] Count;
  // Counter
  always @ (posedge Clock) Count <= Count + 1;
  ```

- FIFOs
  - Buffer to match two data rates
  - Great for datapath clock domain crossings

FSM Blocks / Controllers

- Clearly specify the control signals
- Mark which FSM outputs them
- Leave control connections off block diagram
- ASMD
- Bubble and Arc