General Standards

- Instantiate modules using named parameters, not positional parameters
- Only use synthesizable constructs
  - if-then-else
  - case
  - assign
  - simple math: +, -, <<, >>, &, |, ^, &&, ||, =, ==, {}
  - always
  - assignments: =, <=
  - (this list is not complete)
- Loops are forbidden!
- Complex math is forbidden
  - integer variables
  - floating point variables
  - /, %
Datapath Coding Standards

• Non-leaf nodes should contain only
  • Module instantiations
  • Wires
  • Simple assigns for renaming: assign foo = bar (and not many of these)

• Leaf nodes are either stateful or not.

• Stateful leaves
  • Registers
  • Register files
  • Memory modules.
  • Need to have clk and reset.
  • May contain always @(posedge clk), always @(*), and ‘<=’ assignments

• Non-stateful leaves
  • May contain always @(*), and ‘=’ assignments
  • No clk or reset input.
Datapath Coding Standards

- Consistently use a good naming conventions
- Label all inputs and outputs
  - e.g. foo_in, foo_out
- Include module types in their names
  - A_mux -- the instantiated mux
- Give control lines descriptive and consistent names
  - A_mux_sel_in -- the input that controls the mux
  - A_mux_sel -- should not exist since it would be a control line (and would come from the control path)
  - The control unit would have a corresponding A_mux_sel_out
Build Useful Modules

- Parameterize!
- You should only ever write code for one
  - Register (of any width)
  - 2-input mux (of any width)
  - 3-input mux
  - etc.
- Give your modules descriptive names
  - my3Mux
  - my4Mux
  - myFF
  - gcd_control
  - gcd_datapath
  - gcd -- top-level.
Control Coding Standards

• Use clear, consistent names for signals
• Implement your state machines in 3 always blocks
  • One block computes the state transitions. (always@(*)
  • One block computes the outputs. (always@(*))
  • One block implements the register for the state. (always @ (posedge clk or posedge reset)
• Use ‘localparam’ to define state names. -- No magic numbers! (0 and 1 are not magic)