Outline for Today's Lecture

• Numbers
• Bit manipulations
• Test Bench Simulation
# Recap - Precedence

## Order of operations

<table>
<thead>
<tr>
<th>Highest</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>~</td>
<td>NOT</td>
</tr>
<tr>
<td>*, /, %</td>
<td>mult, div, mod</td>
</tr>
<tr>
<td>+, -</td>
<td>add, sub</td>
</tr>
<tr>
<td>&lt;&lt;, &gt;&gt;</td>
<td>shift</td>
</tr>
<tr>
<td>&lt;&lt;&lt;, &gt;&gt;&gt;</td>
<td>arithmetic shift</td>
</tr>
<tr>
<td>&lt;, &lt;=, &gt;, &gt;=</td>
<td>comparison</td>
</tr>
<tr>
<td>==, !=</td>
<td>equal, not equal</td>
</tr>
<tr>
<td>&amp;</td>
<td>AND, NAND</td>
</tr>
<tr>
<td>^, ~^</td>
<td>XOR, XNOR</td>
</tr>
<tr>
<td></td>
<td>, ~</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Lowest</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>?:</td>
<td>ternary operator</td>
</tr>
</tbody>
</table>

**ENGR 303**
# Numbers

**Format: N'Bvalue**

- **N** = number of bits, **B** = base
- **N'B** is optional but recommended (default is decimal)

<table>
<thead>
<tr>
<th>Number</th>
<th># Bits</th>
<th>Base</th>
<th>Decimal Equivalent</th>
<th>Stored</th>
</tr>
</thead>
<tbody>
<tr>
<td>3'b101</td>
<td>3</td>
<td>binary</td>
<td>5</td>
<td>101</td>
</tr>
<tr>
<td>'b11</td>
<td>unsized</td>
<td>binary</td>
<td>3</td>
<td>00...0011</td>
</tr>
<tr>
<td>8'b11</td>
<td>8</td>
<td>binary</td>
<td>3</td>
<td>000000011</td>
</tr>
<tr>
<td>8'b1010_1011</td>
<td>8</td>
<td>binary</td>
<td>171</td>
<td>10101011</td>
</tr>
<tr>
<td>3'd6</td>
<td>3</td>
<td>decimal</td>
<td>6</td>
<td>110</td>
</tr>
<tr>
<td>6'o42</td>
<td>6</td>
<td>octal</td>
<td>34</td>
<td>100010</td>
</tr>
<tr>
<td>8'hAB</td>
<td>8</td>
<td>hexadecimal</td>
<td>171</td>
<td>10101011</td>
</tr>
<tr>
<td>42</td>
<td>Unsized</td>
<td>decimal</td>
<td>42</td>
<td>00...0101010</td>
</tr>
</tbody>
</table>

// underscores (_) are used for formatting only to make it easier to read. SystemVerilog ignores them.
assign y = {a[2:1], {3{b[0]}}, a[0], 6'b100_010};

// if y is a 12-bit signal, the above statement produces:
y = a[2] a[1] b[0] b[0] b[0] a[0] 1 0 0 0 1 0

// underscores (_) are used for formatting only to make it easier to read. SystemVerilog ignores them.
Bit Manipulations: Example 2

Verilog:
module mux2_8(input [7:0] d0, d1,
    input s,
    output [7:0] y);

    mux2 lsbmux(d0[3:0], d1[3:0], s, y[3:0]);
    mux2 msbmux(d0[7:4], d1[7:4], s, y[7:4]);
endmodule
module tristate(input [3:0] a,
      input en,
      output [3:0] y);
    assign y = en ? a : 4'bz;
endmodule

// N’bz is an N bit binary high impedance assignment in Verilog
Test Bench

- Test Bench is a System Verilog program for applying stimulus to an HDL model and observe its response
- Used to produce waveform of stimulus and response or display simulation log of the value of parameters at given points in time
Test Bench – Gate Delay

• **timescale** is a compiler directive that start with (‘) and specifies unit of measurement for time delay

  ‘timescale 1ns/100ps

• The first number specifies the unit of time measurement for delay. The second number specifies the precision for which the delay is rounded off to, in this case 0.1ns.

• If no timescale is specified, the simulator may display dimensionless values or default to a certain time unit, typically 1ns.

• Timescale is specified before the declaration of a module and applies to code that follows
Test Bench - Example

- Write Test Bench module to simulate the given circuit for input changing from A,B,C = 0 at time zero to A,B,C = 1 after 100nS
- Assume \textbf{and}, \textbf{or} and \textbf{not} gates have delay of 30, 20, and 10nS, respectively
Test Bench - Example

- First add delay time to Verilog model

```verilog
`timescale 1ns/100ps
module simple_circuit_prop_delay(input A, B, C,
 output D, E);
wire w1;
and #(30) G1 (w1, A, B); // 30ns and gate delay
not #(10) G2 (E, C); // 10ns not gate delay
or #(20) G3 (D, w1, E); // 20ns or gate delay
endmodule
```
Test Bench - Example

• Next write the stimulus module

module t_simplified_circuit_prop_delay; //test bench name
wire D, E;  // simulation results specified by wire
reg A, B, C; // simulation inputs specified by reg

simple_circuit_prop_delay M1(A, B, C, D, E); //instance name
initial
begin
  A=l’b0; B=l’b0; C=l’b0; // Set A, B, C = 0 at t=0nS
  #100 A=l’b1; B=l’b1; C=l’b1; // Set A, B, C =1 at t=100nS
end
initial #200 $finish; // simulation end time 200nS
endmodule
Test Bench - Example

• Compile and simulate