

Modeling Approaches in Verilog

Omid Ardakanian

CS 450/650 - Computer Architecture
University of Waterloo

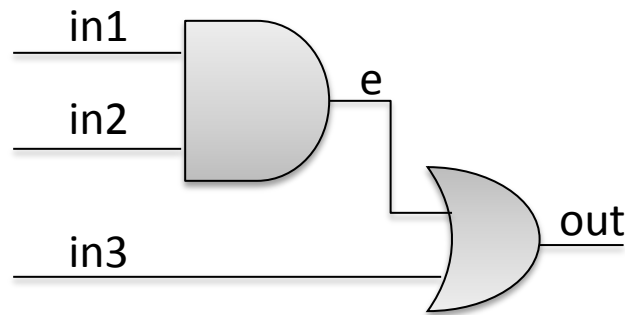
Modeling Approaches

- Modeling approaches
 - Structural (gate-level) **low level of abstraction**
 - Dataflow
 - Behavioural **high level of abstraction**

Gate-level Modeling

- Instantiation
 - Gate primitives
 - Defined modules
- Connectivity

- Example:
 - `and (e,in1,in2)`
 - `or (out,in3,e)`



Gate-level Modeling – Transition Delay

- rise delay
- fall delay
- turnoff delay

- Example:
 - buf #(delay) b1 (out,in)
 - buf #(rise_val, fall_val) b1 (out,in)
 - buf #(rise_val, fall_val, turnoff_val) b1 (out,in)

Gate-level Modeling – Delay Settings

- Min/Typical/Max
- Example:
 - and #(2:3:4, 3:4:5, 4:5:6) a(out,in1,in2)

A Simple Example

```
module full_adder(sum, c_out, a, b, c_in);
    output sum, c_out;
    input a, b, c_in;
    wire s1, c1, c2;

    xor (s1, a, b);
    and (c1, a, b);
    xor (sum, s1, c_in);
    and (c2, s1, c_in);
    or (c_out, c2, c1);
endmodule

module adder_4_bit(x,y,cin,z,cout);
    input [3:0] x, y;
    input cin;
    output [3:0] z;
    output cout;
    wire [3:1] carry;

    full_adder fa0(x[0],y[0],cin,z[0],carry[1]);
    full_adder fa1(x[1],y[1],carry[1],z[1],carry[2]);
    full_adder fa2(x[2],y[2],carry[2],z[2],carry[3]);
    full_adder fa3(x[3],y[3],carry[3],z[3],cout);
endmodule
```

Dataflow Modeling

- Continuous Assignment
 - Driving a value onto a net
 - Example:
 - `assign <drive-strength>? <delay>? <list_of_assignments>;`
 - Operators
- When does the simulator run the 'assign' statement?
- Example:
 - `assign out = in1 & in2;`
 - `wire out = in1 & in2;`

Dataflow Modeling - Delay

- Example:
 - assign #10 out = in1 & in2;
- Inertial delay

A Simple Example

```
module adder_4_bit(sum, c_out, a, b, c_in);  
    output [3:0] sum;  
    output c_out;  
    input [3:0] a, b;  
    input c_in;  
  
    assign {c_out, sum} = a + b + c_in;  
endmodule
```

Behavioural Modeling - always vs. initial

- initial
 - Starts at $t=0$, and executes exactly once
- always
 - Starts at $t=0$, and executes statements in the 'always' block continuously in a loop
- What happens to multiple initial and/or always blocks?

Behavioural Modeling - Assignments

- Blocking assignments (=)
 - Block execution of statements that follow in a sequential block
 - Execute in the order they are specified
- Non-blocking assignments (<=)
 - Allow scheduling of the following assignments

Behavioural Modeling – Assignments (cont'd)

Swapping

```
always @(posedge clock)
```

```
  a<=b;
```

```
always @(posedge clock)
```

```
  b<=a;
```

Race Condition

```
always @(posedge clock)
```

```
  a=b;
```

```
always @(posedge clock)
```

```
  b=a;
```

Behavioural Modeling – Timing Control

- Delay-based
 - Regular delay control
 - #10 a = b & c;
 - Intra-assignment delay control
 - a = #10 b & c;
 - Zero delay control
 - #0 a = 0;

Behavioural Modeling – Timing Control

- Event-based
 - @(clock)
 - @(posedge clock)
 - @(negedge clock)
 - @(sample_event)
 - Definition: event sample_event
 - Triggering: -> sample_event

Thank you!

Any Questions?