

# Verilog Examples

October 25, 2011

# Hierarchical description of a Full Adder

```
//Gate Level description of Half Adder  
module half_adder(x,y,s,c);  
input x,y;  
output s,c;  
xor(s,x,y);  
and(c,x,y);  
endmodule
```

# Full Adder

```
module full_adder(x,y,cin,s,cout);  
input x,y,cin;  
output s,cout;  
wire s1,c1,c2;  
half_adder ha1(x,y,s1,c1);  
half_adder ha2(cin,s1,s,c2);  
or(cout,c1,c2);  
endmodule
```

# Four bit Full adder

```
module four_bit_adder(x,y,cin,sum,cout);  
input [3:0] x,y;  
input cin;  
output[3:0] sum;  
output cout;  
wire c1,c2,c3;  
//fourbit adder body  
full_adder fa1(x[0],y[0],cin,sum[0],c1);  
full_adder fa2(x[1],y[1],c1,sum[1],c2);  
full_adder fa3(x[2],y[2],c2,sum[2],c3);  
full_adder fa4(x[3],y[3],c3,sum[3],cout);  
endmodule
```

# Four bit Adder -Data Flow Description

```
module adder4(A,B,Cin,SUM,Cout);  
input [3:0] A,B;  
input Cin;  
output [3:0] SUM;  
output Cout;  
assign {Cout,SUM}=A+B+Cin;  
endmodule
```

# One Bit Equality Test

```
module eq1p
  ( input i0,i1,
    output eq );
  assign eq=( $\sim$  i0 &  $\sim$  i1) | (i1 & i0);
endmodule
```

# Two Bit Equality Test

```
module eq2e
  ( input [1:0] a,b,
    output aeqb );
  wire eq1,eq2;
  eq1p e1(a[1],b[1],eq1);
  eq1p e2(a[0],b[0],eq2);
  assign aeqb=eq1 & eq2;
endmodule
```

# Writing a Test Bench

Use **initial** and **always** to generate inputs for the unit you are testing.

**initial**

```
begin /* if more than one statement it must be put begin ...end  
    block */
```

```
A=0; B=0;
```

```
# 10 A=1;
```

```
# 10 A=0; B=1;
```

```
end
```

**initial**

```
begin
```

```
D=3'b000;
```

```
repeat(7) // repeat following statement 7 times
```

```
# 10 D=D+3'b001;
```

```
end
```

# Testing module

```
module test_module_name;  
//Declare local reg and wire identifiers.  
//Instatiate the design module under test  
//Generate stimulus (inputs to design module) using initial and  
always.  
// Display output response.(Display on screen or print)  
endmodule
```

# Display and Control of Testing

**\$display**– display value of variables

**\$monitor**– display variables whenever a value changes during simulation

**\$time**– display simulation time

**\$finish**– terminate simulation

Examples:

```
$display("%d %b %b",A,B,C);
```

```
$monitor($time,,"%b %b %h" , s1,s2,outvalue);
```

## Two Bit Equality Testbench

```
module eq2_ testbench;
reg [1:0] testin0,testin1;
wire testout;
eq2e uut(.a(testin0),.b(testin1),.aeqb(testout));
initial
begin
    testin0=2'b00;
    testin1=2'b00;
#100
    testin0=2'b01;
    testin1=2'b00;
```

```
#100
    testin0=2'b11;
    testin1=2'b00;
#100
    testin0=2'b01;
    testin1=2'b01;
#100
    testin0=2'b01;
    testin1=2'b10;
#100
    testin0=2'b11;
    testin1=2'b10;
```

```
#100
    testin0=2'b10;
    testin1=2'b01;
#100
    testin0=2'b10;
    testin1=2'b10;
$stop;
end
$display "time testin0 testin1 testout");
$monitor(" %d % b %b %b "
$time,testin0,testin1,testout);
endmodule
```

# Dataflow Description 2 to 1 line multiplexer

```
module mux2to1(mout,i0,i1,s);  
output mout;  
input i0,i1,s;  
reg mout;  
assign mout=s?i1:i0;  
endmodule
```

*Conditional Expression*

*condition?true-expression:false-expression*

# Behavioral Description 2 to 1 line multiplexer

```
module mux2to1(mout,i0,i1,s);  
output mout;  
input i0,i1,s;  
reg mout;  
  
always @(i0 or i1 or s)  
    if (s==1) mout=i1;  
    else mout=i0;  
  
endmodule
```

# Behavioral Description of 2 to 4 Decoder

```
module dec2x4(xin,yout,enable);  
input [1:0] xin; input enable;  
output[3:0] yout;  
reg[3:0] yout;  
  
always @(xin or enable)  
begin  
if(enable==1)  
case(xin)  
2'b00: yout=4'b0001;  
2'b01: yout=4'b0010;  
2'b10: yout=4'b0100;  
2'b11: yout=4'b1000;  
endcase  
else  
yout=4'b0000;  
end  
  
endmodule
```

# Behavioral Description of four-to-one line multiplexer

```
module mux4to1
( output reg mout,
  input in0,in1,in2,in3
  input [1:0] sel)

always @(in0,in1,in2,in3,sel)
  case(sel)
    2'b00: mout=in0;
    2'b01: mout=in1;
    2'b10: mout=in2;
    2'b11: mout=in3;
  endcase

endmodule
```

# Testbench example Test mux2to1

```
module testmux2to1;
wire tout;
reg tA,tB,tselect;
parameter stoptime=50;
mux2to1 mux1(tout,tA,tB,tselect);
initial #stoptime $finish;
initial begin
tselect=1;tA=0; tB=1;
#10 tA=1; tB=0;
#10 tselect=0;
#10 tA=0;tB=1;
end
```

# Testbench example Test mux2to1 - Cont

```
initial begin $display(" time select A B mout");  
$monitor("time=%d select=%b A=%b B=%  
mout=%b",$time,tselect,tA,tB,tout);  
end  
endmodule
```