Name: Adithya M SRN: PES1UG20CS621 Section: K

**2:1 Mux**

**mux2.v:**

module mux2(input wire i0,i1,j,output wire o);

assign o=(j==0)?i0:i1;

endmodule

**testbench.v:**

module TB;

reg A,B,S;

wire X;

initial

begin

$dumpfile("dump.vcd");

$dumpvars(0,TB);

end

mux2 newMUX(.i0(A), .i1(B), .j(S), .o(X));

initial

begin

S = 1'b0;

A = 1'b0;

B = 1'b0;

#5

A = 1'b0;

B = 1'b1;

#5

A = 1'b1;

B = 1'b0;

#5

A = 1'b1;

B = 1'b1;

#5

S = 1'b1;

A = 1'b0;

B = 1'b0;

#5

A = 1'b0;

B = 1'b1;

#5

A = 1'b1;

B = 1'b0;

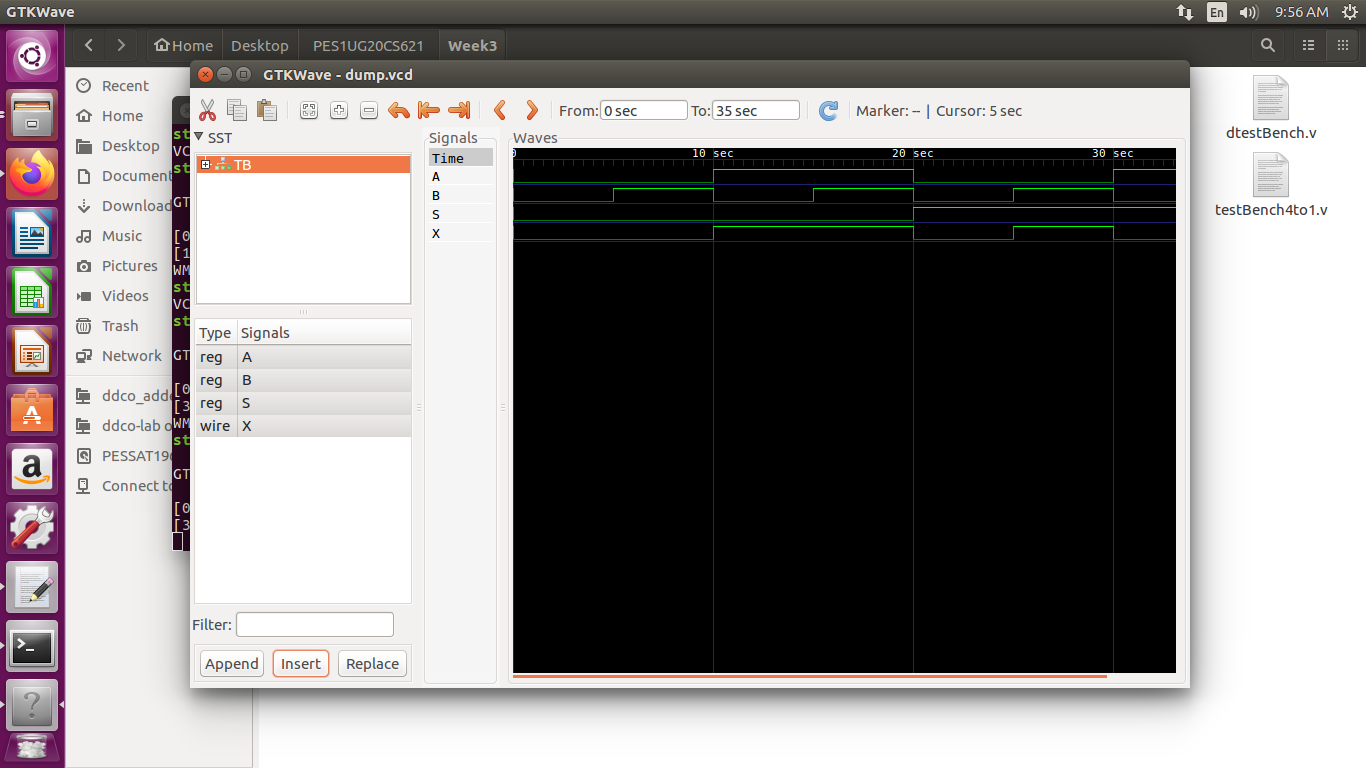
#5

A = 1'b1;

B = 1'b1;

end

endmodule



**4:1 Mux**

**mux4.v**

module mux4 (input wire [0:3] i, input wire j1, j0, output wire o);

wire t0, t1;

mux2 m0 (i[0], i[1], j1, t0);

mux2 m1 (i[2], i[3], j1, t1);

mux2 m2 (t0, t1, j0, o);

endmodule

**testbench.v**

module TB;

reg [0:3]ii;

reg s0;

reg s1;

wire yy;

initial

begin

$dumpfile("dump.vcd");

$dumpvars(0, TB);

end

mux4 newMUX(.i(ii), .j0(s0),.j1(s1),.o(yy));

initial

begin

ii = 4'b0001;

s0=1'b0;

s1=1'b0;

#5

ii = 4'b1000;

#5

ii = 4'b0001;

s0=1'b1;

s1=1'b1;

#5

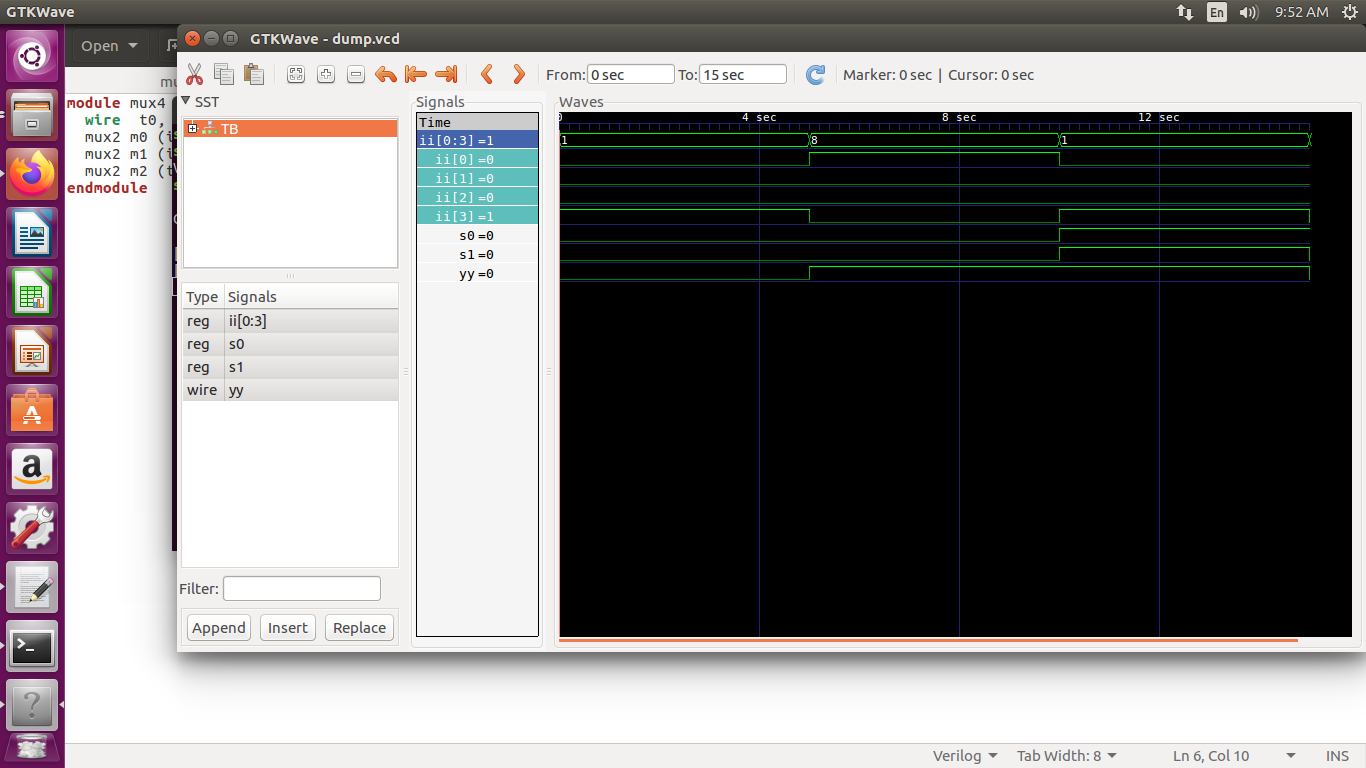
ii = 4'b0000;

s0=1'b1;

s1=1'b0;

end

endmodule



**Full Adder:**

**adder.v**

module fulladd(input wire a,b,cin,output wire sum,cout);

wire t;

assign t = a^b;

assign sum = t^cin;

assign cout = (a&b)|(b&cin)|(a&cin);

endmodule

module halfadd(input wire a,b,output wire ha,hca);

assign ha = a^b;

assign hca = a&b;

endmodule

**testbench.v**

module TB;

reg aa,bb,cc;

wire ss,cy,hs,hc;

fulladd add1(.a(aa), .b(bb), .cin(cc), .sum(ss), .cout(cy));

halfadd add2(.a(aa),.b(bb),.ha(hs),.hca(hc));

initial

begin

$dumpfile("dump.vcd");

$dumpvars(0, TB);

end

initial begin $monitor(aa,bb,cc,ss,cy,hs,hc);

aa = 1'b0;

bb = 1'b0;

cc=1'b0;

#5

aa = 1'b0;

bb = 1'b1;

cc=1'b1;

#5

aa = 1'b1;

bb = 1'b0;

cc=1'b0;

#5

aa = 1'b1;

bb = 1'b1;

cc=1'b0;

#5

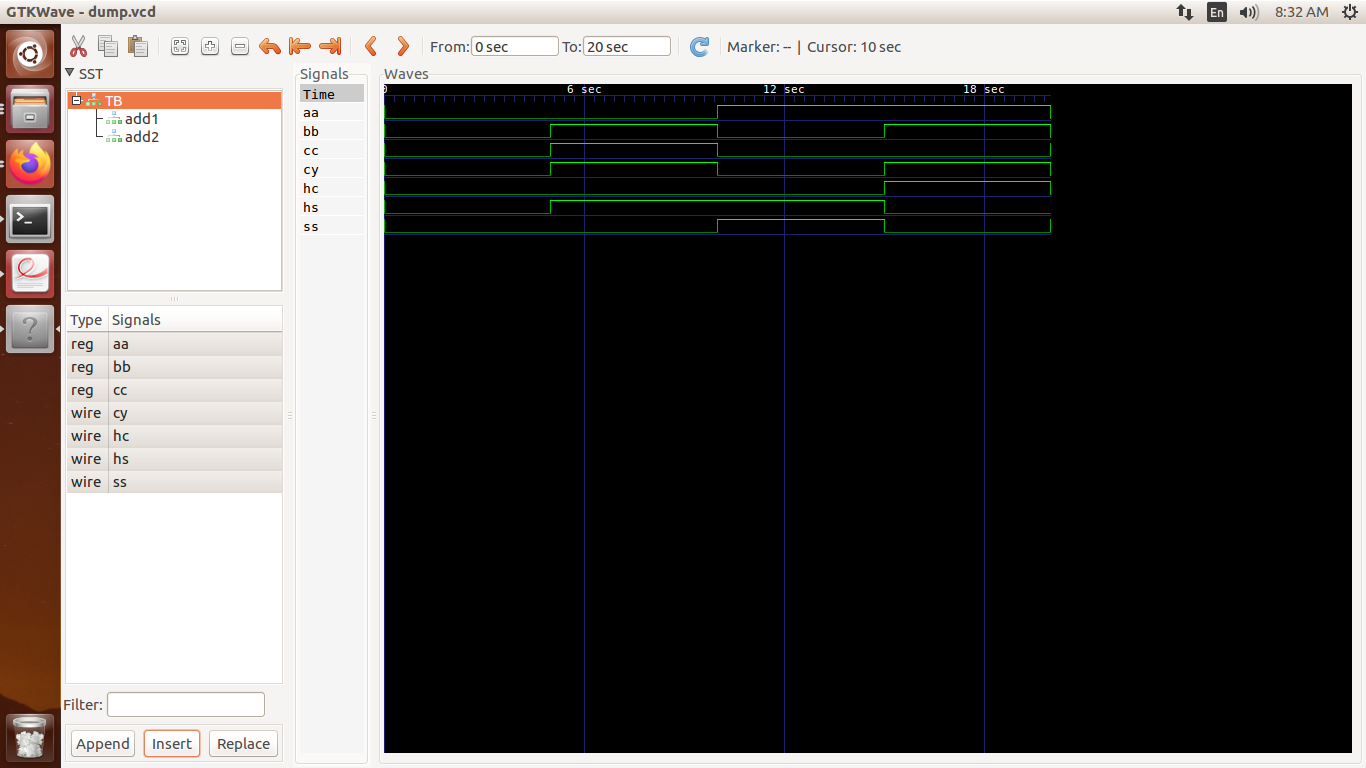
aa = 1'b0;

bb = 1'b0;

cc=1'b1;

end

endmodule



**Ripple Carry Adder:**

**4bit.v**

module fulladdR(input wire a,b,cin,output wire sum,cout);

wire t;

assign t = a^b;

assign sum = t^cin;

assign cout = (a&b)|(b&cin)|(a&cin);

endmodule

module r\_c\_addr(input wire [3:0] a,input wire [3:0]b, input wire cin, output wire [3:0] s, output wire cout);

wire [2:0]c;

fulladdR f\_0(a0,b0,cin,s0,c0);

fulladdR f\_1(a1,b1,c0,s1,c1);

fulladdR f\_2(a2,b2,c1,s2,c2);

fulladdR f\_3(a3,b3,c2,s3,cout);

endmodule

**testbench.v**

`timescale 1 ns / 100 ps

`define TESTVECS 8

module tb;

reg clk, reset;

reg [3:0] i0, i1; reg cin;

wire [3:0] o; wire cout;

reg [8:0] test\_vecs [0:(`TESTVECS-1)];

integer i;

initial begin $dumpfile("dump.vcd");

$dumpvars(0,tb);

end

initial begin reset = 1'b1; #12.5 reset = 1'b0; end

initial clk = 1'b0; always #5 clk =~ clk;

initial begin

test\_vecs[0] = 9'b000000010;

test\_vecs[1] = 9'b000100010;

test\_vecs[2] = 9'b011100010;

test\_vecs[3] = 9'b000001110;

test\_vecs[4] = 9'b011001111;

test\_vecs[5] = 9'b001110011;

test\_vecs[6] = 9'b111100011;

test\_vecs[7] = 9'b011101110;

end

initial {i0, i1, cin, i} = 0;

r\_c\_addr u0 (i0, i1, cin, o, cout);

initial begin

#6 for(i=0;i<`TESTVECS;i=i+1)

begin #10 {i0, i1, cin}=test\_vecs[i]; end

#100 $finish;

end

endmodule

