|  |  |  |
| --- | --- | --- |
| **Name: Sriram R** | **SRN: PES1UG20CS435** | **Section: H** |
| **Date: 01-09-2021** | **Week Number: 1** |

|  |  |
| --- | --- |
| **1** | Implement AND, OR, NAND & NOR gates using expressions. |
|  | **Program:**  **a.v :**  module and1(input wire a,b,output wire w, x, y, z);  assign w=a&b;  assign x=a|b;  assign y=~(a&b);  assign z=~(a|b);  endmodule  **atestBench.v :**  module tb;  reg t\_a;  reg t\_b;  wire t\_w,t\_x,t\_y,t\_z;  //instatiate  and1 a1(.a(t\_a),.b(t\_b),.w(t\_w),.x(t\_x),.y(t\_y),.z(t\_z));  initial begin $dumpfile("dump1.vcd");  $dumpvars(0,tb);  end  initial begin $monitor(t\_a,t\_b,t\_w,t\_x,t\_y,t\_z); //displays the content of the register  t\_a=1'b0;//1 bit input  t\_b=1'b0;  #10 //time nanosecs  t\_a=1'b0;//1 bit input  t\_b=1'b1;  #10 //time nanosecs  t\_a=1'b1;//1 bit input  t\_b=1'b0;  #10 //time nanosecs  t\_a=1'b1;//1 bit input  t\_b=1'b1;  #10 //time nanosecs  t\_a=0;//inorder to see the last input  t\_b=0;  end  endmodule |
|  | **Output Screenshots:** |
| 2 | Implement NOT, AND, OR, XOR & NAND gates using modules. |
|  | **Program :**  **b.v :**  module invert(input wire i, output wire o1);  assign o1 = ~(i);  endmodule  module and2(input wire i0, i1, output wire o2);  assign o2 = (i0 & i1);  endmodule  module or2(input wire i0, i1, output wire o3);  assign o3 = (i0 | i1);  endmodule  module xor2(input wire i0, i1, output wire o4);  assign o4 = (i0 ^ i1);  endmodule  module nand2(input wire i0, i1, output wire o5);  wire t;  assign t = (i0 & i1);  assign o5 = ~(t);  endmodule  **btestBench.v:**  module tb;  reg t\_a;  reg t\_b;  wire P,Q,R,S,T;  //instantiate  invert a1(.i(t\_a),.o1(P));  and2 a2(.i0(t\_a),.i1(t\_b),.o2(Q));  or2 a3(.i0(t\_a),.i1(t\_b),.o3(R));  xor2 a4(.i0(t\_a),.i1(t\_b),.o4(S));  nand2 a5(.i0(t\_a),.i1(t\_b),.o5(T));  initial begin $dumpfile("dump2.vcd");  $dumpvars(0,tb);  end  initial begin $monitor(t\_a,t\_b,P,Q,R,S,T); //displays the content of the register  t\_a=1'b0;//1 bit input  t\_b=1'b0;  #10 //time nanosecs  t\_a=1'b0;//1 bit input  t\_b=1'b1;  #10 //time nanosecs  t\_a=1'b1;//1 bit input  t\_b=1'b0;  #10 //time nanosecs  t\_a=1'b1;//1 bit input  t\_b=1'b1;  #10 //time nanosecs  t\_a=0;//inorder to see the last input  t\_b=0;  end  endmodule |
|  | **Output Screenshots:** |

**Note : Full adder included as a part of week2 submission**