STRUCTURAL PROGRAMMING

Q1.FULL ADDER USING HALF ADDER:

VERILOG CODE:

```verilog
//Lower level model
module HA(A,B,Sum,Cout);
  input A,B;                // Inputs: A and B
  output Sum,Cout;          // Outputs: Sum and Carry-out (Cout)
  assign Sum=A^B;           // Sum is the XOR of A and B
  assign Cout=A&B;          // Carry-out is the AND of A and B
endmodule


//Top level Model Full Adder
`include "HA.v"             // Include the Half Adder module
module FA(A,B,Cin,Sum,Cout);
  input A,B,Cin;                 // Inputs: A, B, and Carry-in (Cin)
  output Sum,Cout;               // Outputs: Sum and Carry-out (Cout)
  wire w1,w2,w3;                 // Internal wires to connect the half adders
  HA h1(A,B,w1,w2);              // First half adder: adds A and B
  HA h2(w1,Cin,Sum,w3);         // Second half adder: adds w1 and Cin to produce the final Sum
  assign Cout=w2|w3;             // Final Carry-out is the OR of two intermediate carries
endmodule
```

TEST BENCH CODE:

```verilog
module FA_test;
reg A,B,Cin;
wire Sum,Cout;
  FA dut(A,B,Cin,Sum,Cout);
  initial begin
    A=1'b0; B=1'b0; Cin=1'b0;
  #5 A=1'b0; B=1'b0; Cin=1'b1;
  #5 A=1'b0; B=1'b1; Cin=1'b0;
```

```verilog
    #5 A=1'b0; B=1'b1; Cin=1'b1;

    #5 A=1'b1; B=1'b0; Cin=1'b0;

    #5 A=1'b1; B=1'b0; Cin=1'b1;

    #5 A=1'b1; B=1'b1; Cin=1'b0;

    #5 A=1'b1; B=1'b1; Cin=1'b1;

 end

 initial begin

  $monitor("Sim
    time=%0t,A=%b,B=%b,Cin=%b,Sum=%b,Cout=%b",$time,A,B,Cin,Sum,Cout);

 end

 initial begin

  $dumpfile("dump.vcd");

  $dumpvars(0,A,B,Cin,Sum,Cout);

 end

endmodule
```
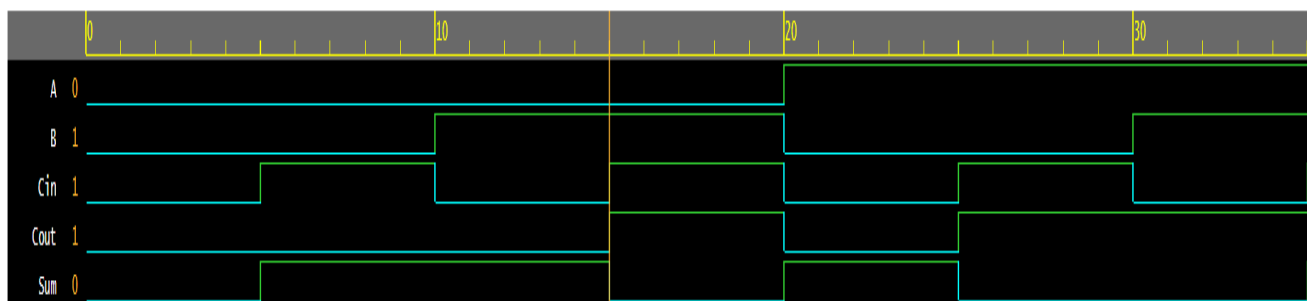
OUTPUT WAVEFORMS:

SIMULATION OUTPUT:

```
    Sim time=0,A=0,B=0,Cin=0,Sum=0,Cout=0
    Sim time=5,A=0,B=0,Cin=1,Sum=1,Cout=0
    Sim time=10,A=0,B=1,Cin=0,Sum=1,Cout=0
    Sim time=15,A=0,B=1,Cin=1,Sum=0,Cout=1
    Sim time=20,A=1,B=0,Cin=0,Sum=1,Cout=0
    Sim time=25,A=1,B=0,Cin=1,Sum=0,Cout=1
    Sim time=30,A=1,B=1,Cin=0,Sum=0,Cout=1
    Sim time=35,A=1,B=1,Cin=1,Sum=1,Cout=1
```