STRUCTURAL PROGRAMMING

Q. 4 bit Adder using Full Adder:

VERILOG CODE:

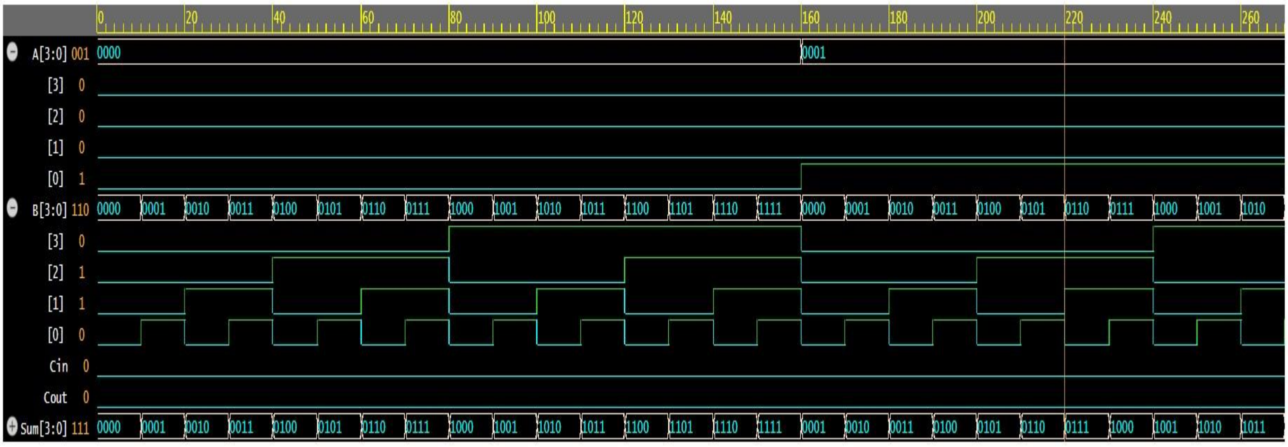
```
//Lower level module
module FA(A,B,Cin,Sum,Cout);
  input A,B,Cin;
  output Sum,Cout;
  assign Sum=A^B^Cin;
  assign Cout=(A&B)|(B&Cin)|(A&Cin);
endmodule


//Top Level Module
`include "FA.v"
module adder4bit(A,B,Cin,S,Cout);
  input [3:0]A,B;
  input Cin;
  output [3:0]S;
  output Cout;
  wire [2:0]C;
  FA f1(A[0],B[0],Cin,S[0],C[0]);
  FA f2(A[1],B[1],C[0],S[1],C[1]);
  FA f3(A[2],B[2],C[1],S[2],C[2]);
  FA f4(A[3],B[3],C[2],S[3],Cout);
 endmodule
```

TESTBENCH CODE:

```
module adder4bit_;
   reg [3:0] A, B;
   reg Cin;
   wire [3:0] Sum;
   wire Cout;
```

```
    integer i;
    adder4bit uut (A,B,Cin,Sum,Cout);
    initial begin
        Cin = 0;
      $monitor("Time=%0t ,A=%b, B=%b, Cin=%b => Sum=%b Cout=%b", $time, A, B, Cin,
Sum, Cout);
      for (i = 0; i <256; i = i + 1)
   initial begin
            {A,B}=i;
               #10;
            end
    end
      initial begin
    $dumpfile("dump.vcd");
    $dumpvars(0,A, B, Cin, Sum, Cout);
  end
endmodule
```

OUTPUT WAVEFORM:



0 20 40 60 80 100 120 140 160 180 200 220 240 260
A[3:0] 001
0000
0001
[3] 0
[2] 0
[1] 0
[0] 1
B[3:0] 110
0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010
[3] 0
[2] 1
[1] 1
[0] 0
Cin 0
Cout 0
Sum[3:0] 111
0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011