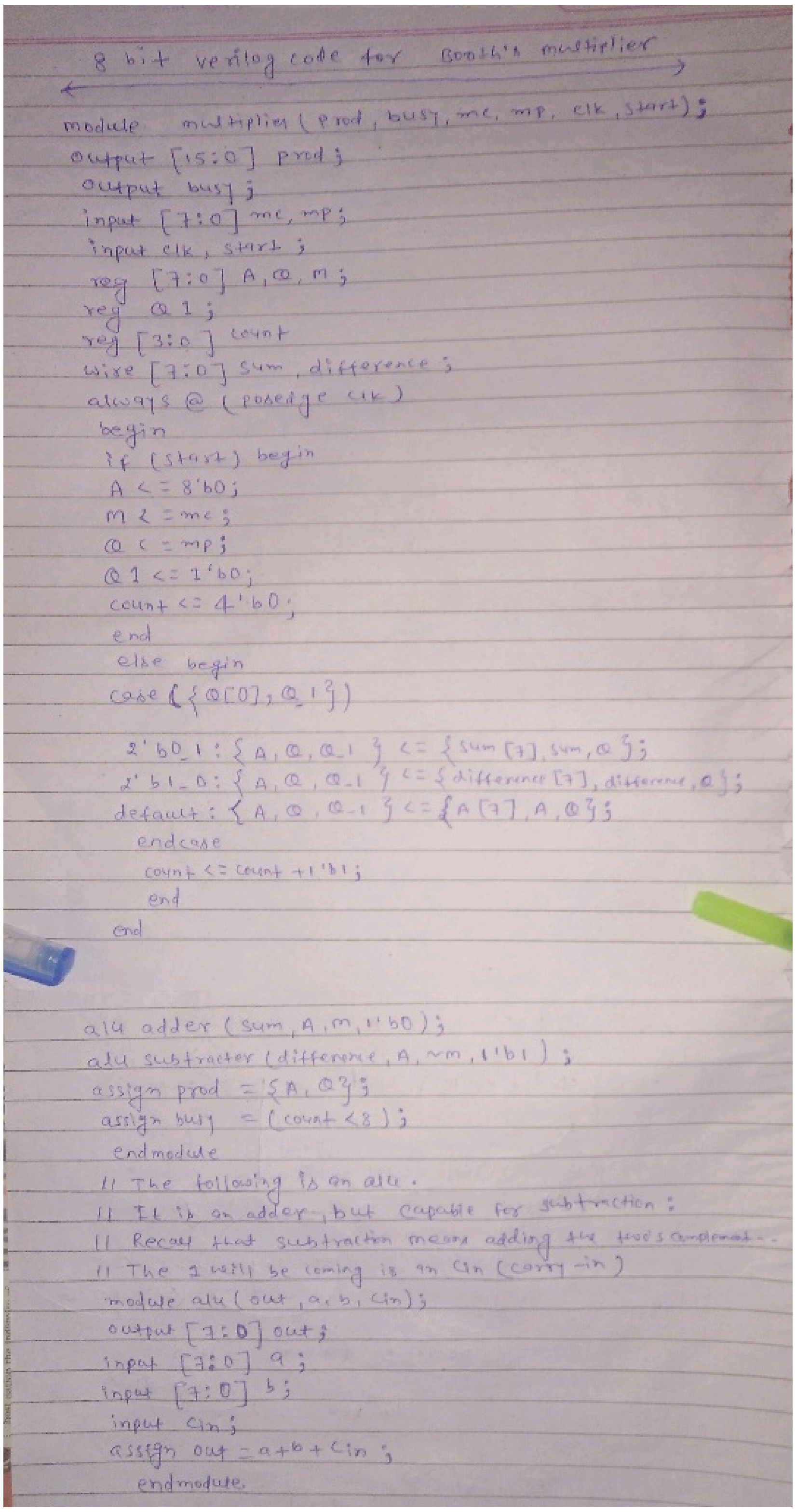

8 bit verilog code for Booth's multiplier

```
module  multiplier ( prod, busy, mc, mp, clk ,start);
output [15:0] prod;
output busy;
input [7:0] mc, mp;
input clk, start ;
reg [7:0] A, Q, m;
reg  Q 1;
reg [3:0] count
wire [7:0] sum , difference ;
always @ ( posedge clk)
 begin
  if (start) begin
  A <= 8'b0;
  m < = mc;
  Q < = mp;
  Q 1 <= 1'b0;
  count <= 4'b0;
  end
   else  begin
  case ({Q[0], Q_1})

   2'b0_1 : {A, Q, Q_1 } <= {sum [7], sum, Q };
   2'b1_0 : {A, Q , Q_1 } <= {difference [7], difference, Q};
  default : {A, Q , Q_1 } <= {A [7], A, Q};
    endcase
    count <= count +1'b1;
    end
  end


alu adder ( sum, A, m, 1'b0);
alu substracter (difference, A, ~m, 1'b1 ) ;
assign prod = {A, Q};
 assign busy = (count <8 );
  endmodule
 // The following is an alu.
 // It is an adder, but capable for subtraction :
 // Recall that subtraction means adding the two's complement - -
 // The 1 will be coming is an Cin (carry -in )
 module alu ( out , a, b, Cin);
  output [7:0] out;
 input [7:0] a ;
 input [7:0] b;
  input Cin;
  assign out = a+b + Cin ;
    endmodule.
```

red begin 1451'6 e nd ele begin sm (.sm end medute