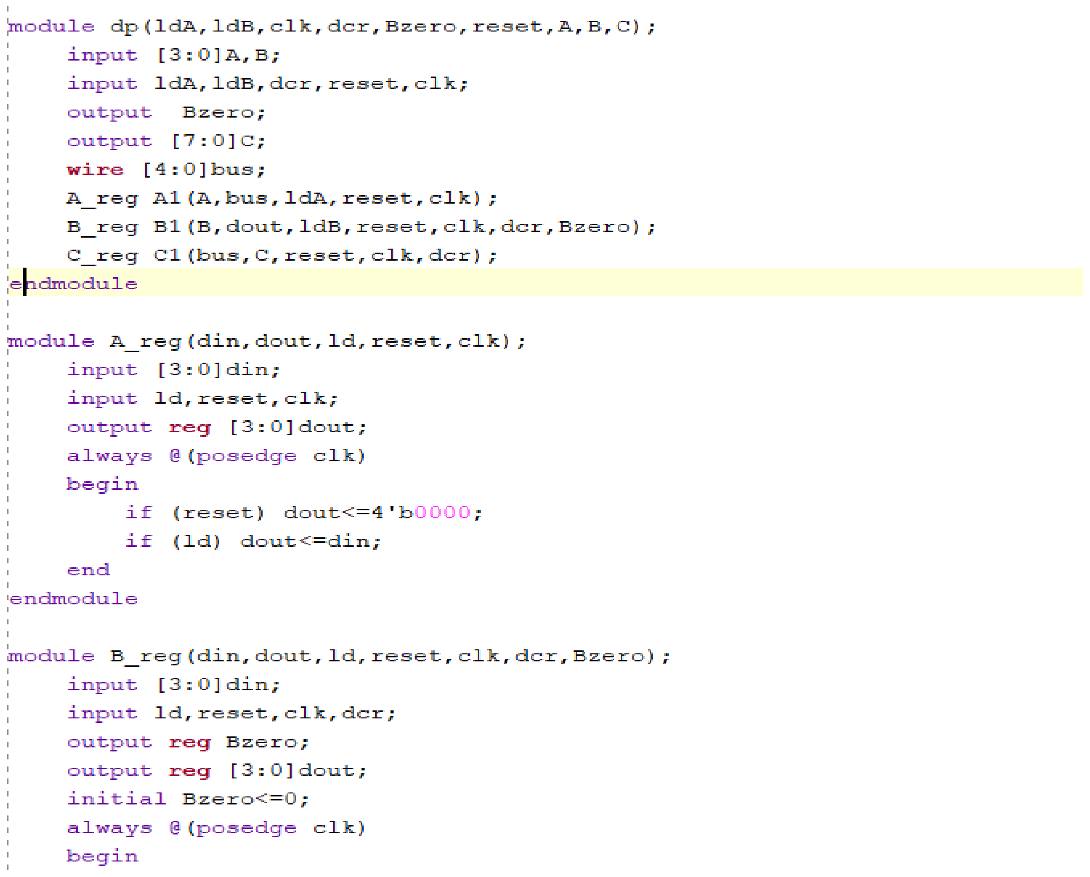
Multiplier design using controller and data path in Verilog

Controller :-

```
module cp(ldA,ldB,clk,start,dcr,Bzero,done,reset);
    input clk,start,Bzero;
    output reg ldA,ldB,dcr,done,reset;
    reg [3:0]state;
    parameter S0=4'b0000, S1=4'b0001, S2=4'b0010, S3=4'b0011, S4=4'b0100;
    always @(posedge clk)
    begin
        case(state)
        S0:  if(start) state<= S1;
        S1:  begin if (~Bzero) state<=S2; else state<=S4; end
        S2:  state<=S3;
        S3: #1 begin  if (~Bzero) state<=S2; else state<=S4; end
        S4:  if (start) state<=S1;
        default: state<=S0;
        endcase
    end
    always @(posedge clk)
    begin
    case (state)
        S0: begin reset<=1; done<=0; end
        S1: begin reset<=0; ldA<=1; ldB<=1; end
        S2: begin ldB<=0; ldA<=0; dcr <=1 ;end
        S3: dcr<=0;
        S4: done<=1;
        default: begin reset<=0; ldA<=0; ldB<=0; dcr<=0; done<=0; end
        endcase
    end
endmodule
```

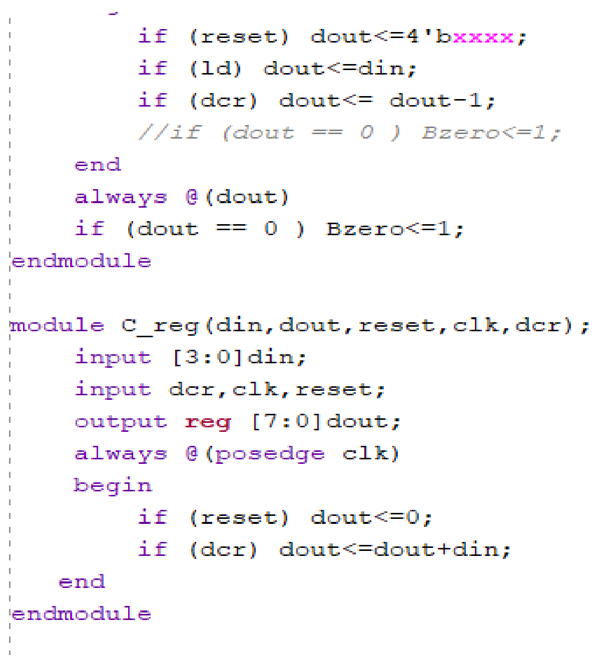
Data Path :-



```verilog
module dp(ldA,ldB,clk,dcr,Bzero,reset,A,B,C);
    input [3:0]A,B;
    input ldA,ldB,dcr,reset,clk;
    output  Bzero;
    output [7:0]C;
    wire [4:0]bus;
    A_reg A1(A,bus,ldA,reset,clk);
    B_reg B1(B,dout,ldB,reset,clk,dcr,Bzero);
    C_reg C1(bus,C,reset,clk,dcr);
endmodule

module A_reg(din,dout,ld,reset,clk);
    input [3:0]din;
    input ld,reset,clk;
    output reg [3:0]dout;
    always @(posedge clk)
    begin
        if (reset) dout<=4'b0000;
        if (ld) dout<=din;
    end
endmodule

module B_reg(din,dout,ld,reset,clk,dcr,Bzero);
    input [3:0]din;
    input ld,reset,clk,dcr;
    output reg Bzero;
    output reg [3:0]dout;
    initial Bzero<=0;
    always @(posedge clk)
    begin
```



```verilog
        if (reset) dout<=4'bxxxx;
        if (ld) dout<=din;
        if (dcr) dout<= dout-1;
        //if (dout == 0 ) Bzero<=1;
    end
    always @(dout)
    if (dout == 0 ) Bzero<=1;
endmodule

module C_reg(din,dout,reset,clk,dcr);
    input [3:0]din;
    input dcr,clk,reset;
    output reg [7:0]dout;
    always @(posedge clk)
    begin
        if (reset) dout<=0;
        if (dcr) dout<=dout+din;
    end
endmodule
```

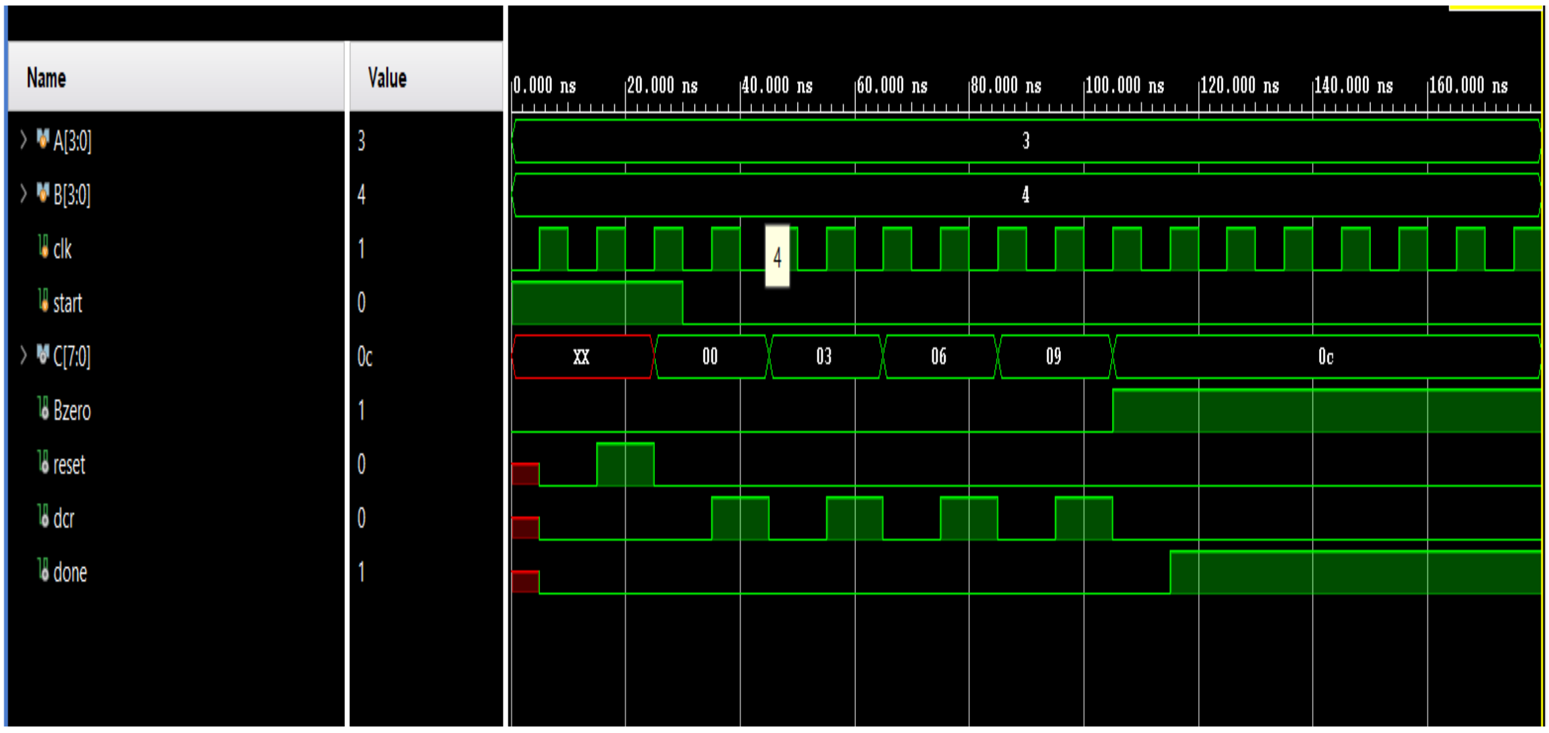
Top Module :-

```verilog
module multiplier(A,B,C,clk,start,Bzero,reset,dcr,done);
    input [3:0]A,B;
    output Bzero,reset,dcr,done;
    output [7:0]C;
    input clk,start;
    cp control_path(ldA,ldB,clk,start,dcr,Bzero,done,reset);
    dp data_path(ldA,ldB,clk,dcr,Bzero,reset,A,B,C);
endmodule
```

Test Bench :-

```verilog
module tb();
    reg [3:0]A,B;
    reg clk,start;
    wire [7:0]C;
    multiplier dit(A,B,C,clk,start,Bzero,reset,dcr,done);
    initial begin
    clk=0;
    forever #5 clk=~clk;
    end
    initial begin
    A=3; B=4; start=1;
    #30
    start=0;
    #150
    $finish;
    end
endmodule
```

Output Waveform :-



Name
Value
A[3:0]
3
B[3:0]
4
clk
1
start
0
C[7:0]
0c
Bzero
1
reset
0
dcr
0
done
1
0.000 ns
20.000 ns
40.000 ns
60.000 ns
80.000 ns
100.000 ns
120.000 ns
140.000 ns
160.000 ns
3
4
4
XX
00
03
06
09
0c