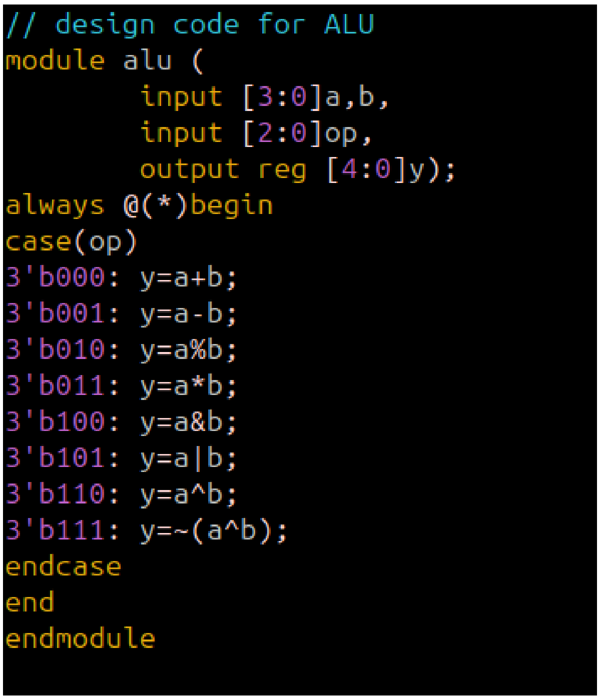
VERILOG ASSIGNMENT

Name   : Munees Sanid
Roll no : KVLSI2501076

GITHUB

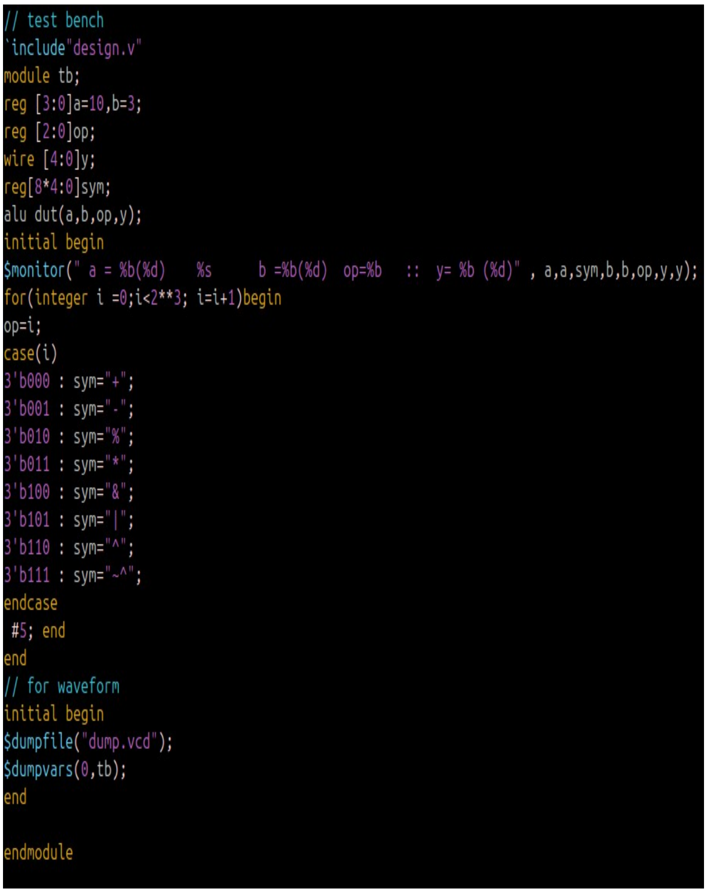
1. Write a verilog code to implement ALU

Design code



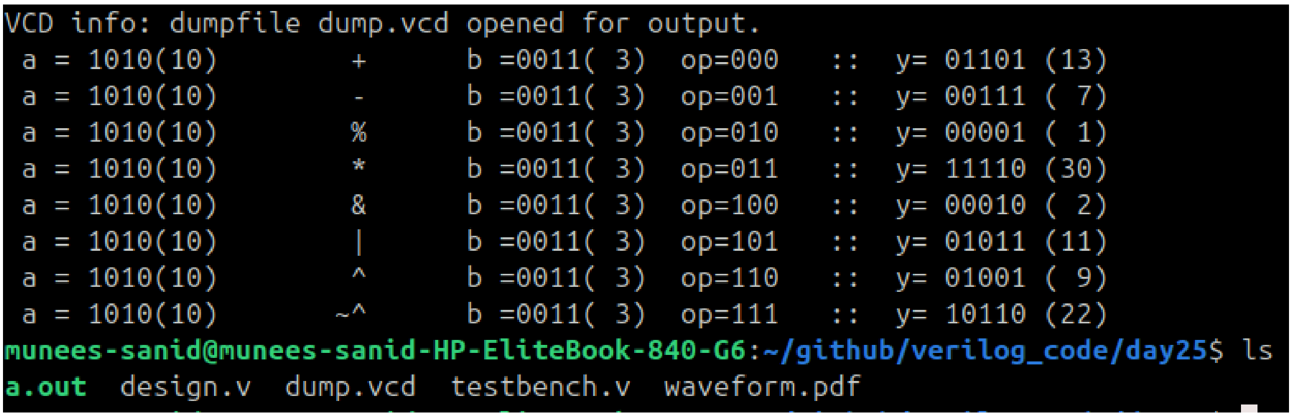
```
// design code for ALU
module alu (
        input [3:0]a,b,
        input [2:0]op,
        output reg [4:0]y);
always @(*)begin
case(op)
3'b000: y=a+b;
3'b001: y=a-b;
3'b010: y=a%b;
3'b011: y=a*b;
3'b100: y=a&b;
3'b101: y=a|b;
3'b110: y=a^b;
3'b111: y=~(a^b);
endcase
end
endmodule
```

Testbench



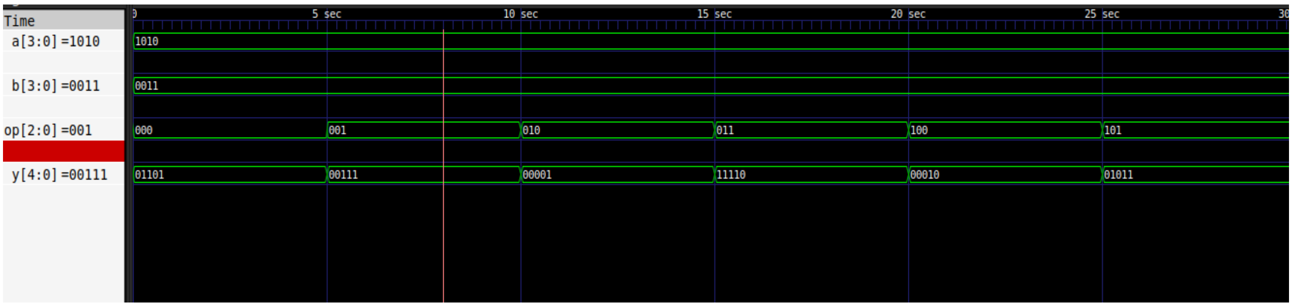
```
// test bench
`include"design.v"
module tb;
reg [3:0]a=10,b=3;
reg [2:0]op;
wire [4:0]y;
reg[8*4:0]sym;
alu dut(a,b,op,y);
initial begin
$monitor(" a = %b(%d)    %s      b =%b(%d)  op=%b   ::  y= %b (%d)" , a,a,sym,b,b,op,y,y);
for(integer i =0;i<2**3; i=i+1)begin
op=i;
case(i)
3'b000 : sym="+";
3'b001 : sym="-";
3'b010 : sym="%";
3'b011 : sym="*";
3'b100 : sym="&";
3'b101 : sym="|";
3'b110 : sym="^";
3'b111 : sym="~^";
endcase
 #5; end
end
// for waveform
initial begin
$dumpfile("dump.vcd");
$dumpvars(0,tb);
end

endmodule
```

Output



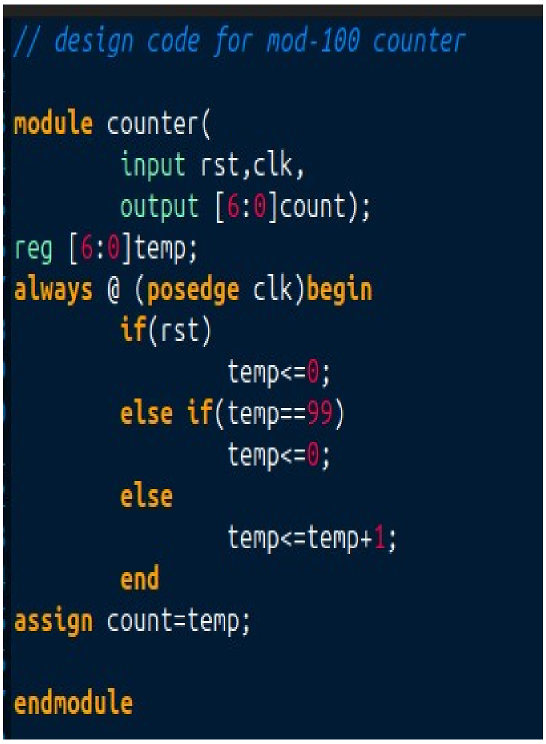
```
VCD info: dumpfile dump.vcd opened for output.
 a = 1010(10)        +      b =0011( 3)  op=000   ::  y= 01101 (13)
 a = 1010(10)        -      b =0011( 3)  op=001   ::  y= 00111 ( 7)
 a = 1010(10)        %      b =0011( 3)  op=010   ::  y= 00001 ( 1)
 a = 1010(10)        *      b =0011( 3)  op=011   ::  y= 11110 (30)
 a = 1010(10)        &      b =0011( 3)  op=100   ::  y= 00010 ( 2)
 a = 1010(10)        |      b =0011( 3)  op=101   ::  y= 01011 (11)
 a = 1010(10)        ^      b =0011( 3)  op=110   ::  y= 01001 ( 9)
 a = 1010(10)       ~^      b =0011( 3)  op=111   ::  y= 10110 (22)
munees-sanid@munees-sanid-HP-EliteBook-840-G6:~/github/verilog_code/day25$ ls
a.out  design.v  dump.vcd  testbench.v  waveform.pdf
```

Waveform



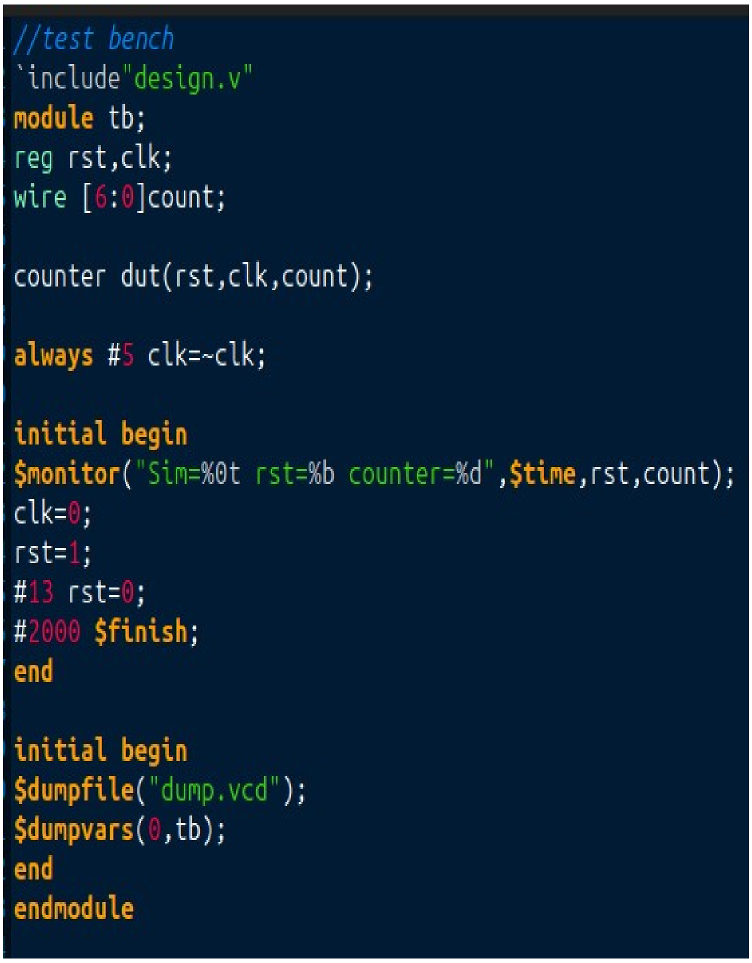
2. Write a Verilog code to design and Implement mod – 100 counter

Design code



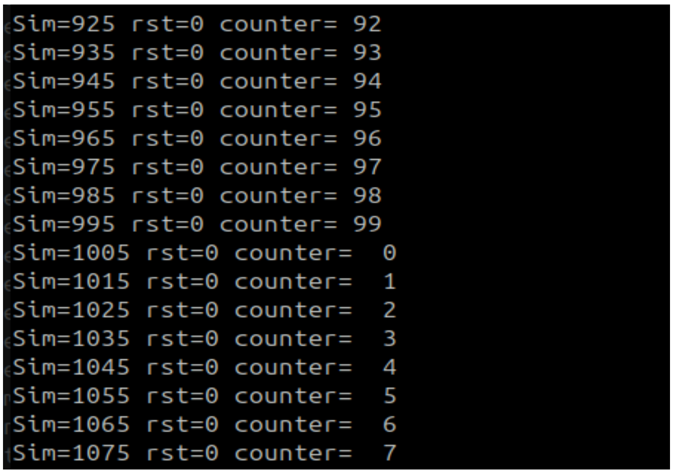
```
// design code for mod-100 counter

module counter(
        input rst,clk,
        output [6:0]count);
reg [6:0]temp;
always @ (posedge clk)begin
        if(rst)
                temp<=0;
        else if(temp==99)
                temp<=0;
        else
                temp<=temp+1;
        end
assign count=temp;

endmodule
```

Testbench



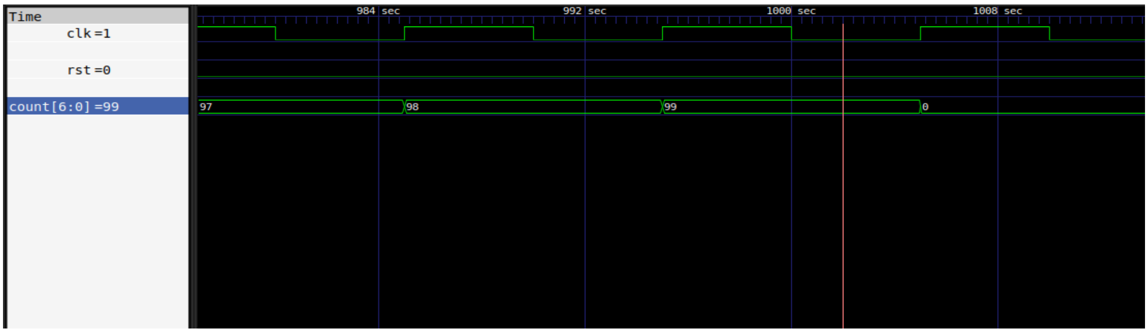
```
//test bench
`include"design.v"
module tb;
reg rst,clk;
wire [6:0]count;

counter dut(rst,clk,count);

always #5 clk=~clk;

initial begin
$monitor("Sim=%0t rst=%b counter=%d",$time,rst,count);
clk=0;
rst=1;
#13 rst=0;
#2000 $finish;
end

initial begin
$dumpfile("dump.vcd");
$dumpvars(0,tb);
end
endmodule
```

Output



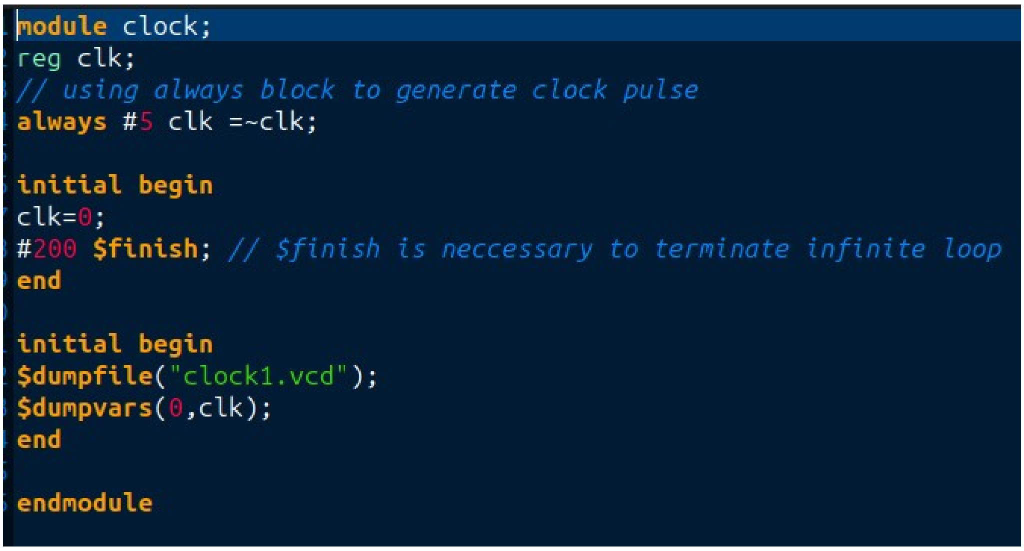
```
Sim=925 rst=0 counter= 92
Sim=935 rst=0 counter= 93
Sim=945 rst=0 counter= 94
Sim=955 rst=0 counter= 95
Sim=965 rst=0 counter= 96
Sim=975 rst=0 counter= 97
Sim=985 rst=0 counter= 98
Sim=995 rst=0 counter= 99
Sim=1005 rst=0 counter=  0
Sim=1015 rst=0 counter=  1
Sim=1025 rst=0 counter=  2
Sim=1035 rst=0 counter=  3
Sim=1045 rst=0 counter=  4
Sim=1055 rst=0 counter=  5
Sim=1065 rst=0 counter=  6
Sim=1075 rst=0 counter=  7
```

Waveform



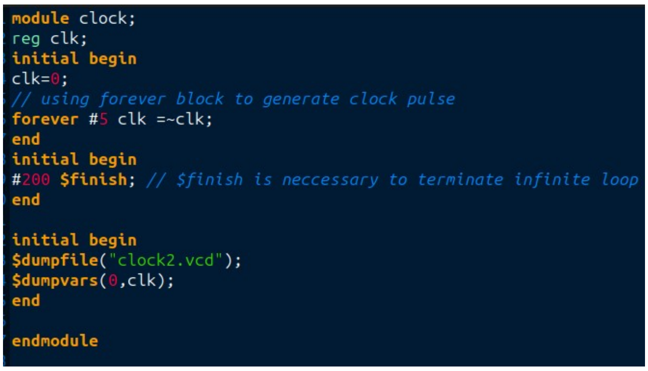
3. Write Verilog code in min 4 ways to generate clock

Using always block



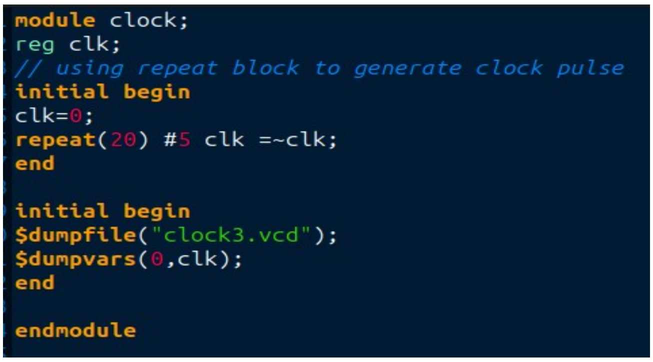
```
module clock;
reg clk;
// using always block to generate clock pulse
always #5 clk =~clk;

initial begin
clk=0;
#200 $finish; // $finish is neccessary to terminate infinite loop
end

initial begin
$dumpfile("clock1.vcd");
$dumpvars(0,clk);
end

endmodule
```

Using forever



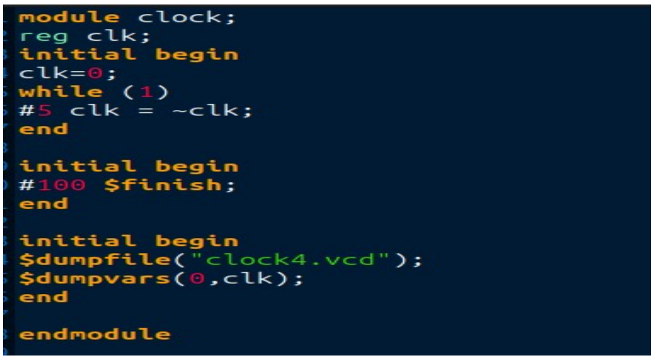
```
module clock;
reg clk;
initial begin
clk=0;
// using forever block to generate clock pulse
forever #5 clk =~clk;
end
initial begin
#200 $finish; // $finish is neccessary to terminate infinite loop
end

initial begin
$dumpfile("clock2.vcd");
$dumpvars(0,clk);
end

endmodule
```

Using repeat



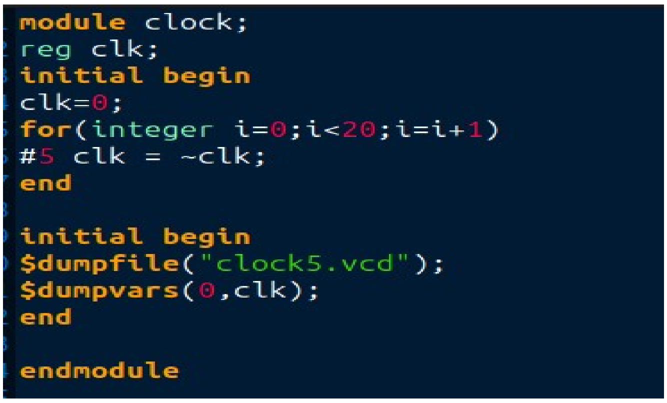
```
module clock;
reg clk;
// using repeat block to generate clock pulse
initial begin
clk=0;
repeat(20) #5 clk =~clk;
end

initial begin
$dumpfile("clock3.vcd");
$dumpvars(0,clk);
end

endmodule
```

Using while



```
module clock;
reg clk;
initial begin
clk=0;
while (1)
#5 clk = ~clk;
end

initial begin
#100 $finish;
end

initial begin
$dumpfile("clock4.vcd");
$dumpvars(0,clk);
end

endmodule
```

Using for loop



```
module clock;
reg clk;
initial begin
clk=0;
for(integer i=0;i<20;i=i+1)
#5 clk = ~clk;
end

initial begin
$dumpfile("clock5.vcd");
$dumpvars(0,clk);
end

endmodule
```