DDCO Mini Project ISA Submission

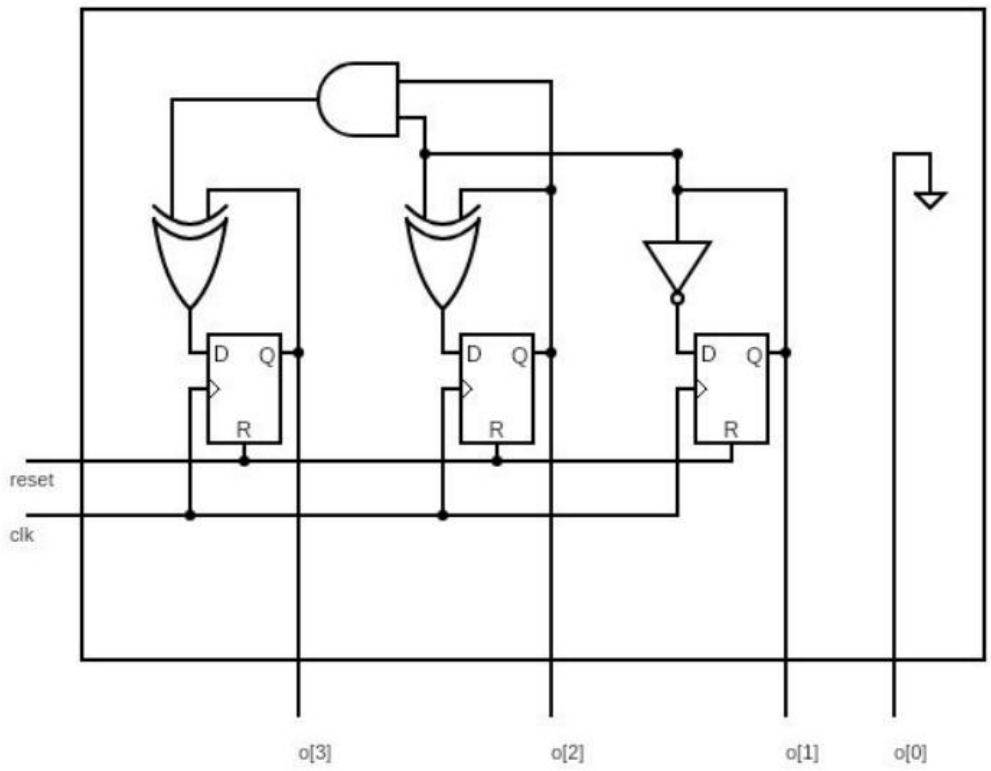
Project Title: Sequence Generator (Even number generator)

Section : H

Batch Details

Student Name	SRN
SUHAS R K	PES1UG19CS514
SUHEB PAPA	PES1UG19CS515
SUJAY S AMBEKAR	PES1UG19CS516
SUJAY S SHIVARA	PES1UG19CS517

Circuit Diagram:



Algorithm (if any):

Since all even numbers in binary ends with 0's,

To construct an n-bit even number generator.

We implement an n-1 bit counter and have a constant LSB output of 1’b0.

For a 4 bit even number generator:

The counter is made of 3 d-flip flops and with one reset, an incrementer circuit and one grounded wire for LSB of output.

Implementation (Code):

TESTBENCH:

```
`timescale 1 ns / 100 ps
module tb;

   // Initialising clock, reset, output wires and even sequence generator module
        wire [ 3 : 0 ] out;
        reg clk, reset;
        even_seq_gen s_g0 (clk, reset, out);


  // Generating a dump file and the dump variable
```

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

   // executing the even sequence generator module

        initial begin
                reset = 1'b1 ;
                #10 reset = 1'b0 ;
        end
        initial clk = 1'b0 ;
        always #5 clk =~ clk;
        initial #300 $finish ;
endmodule
```

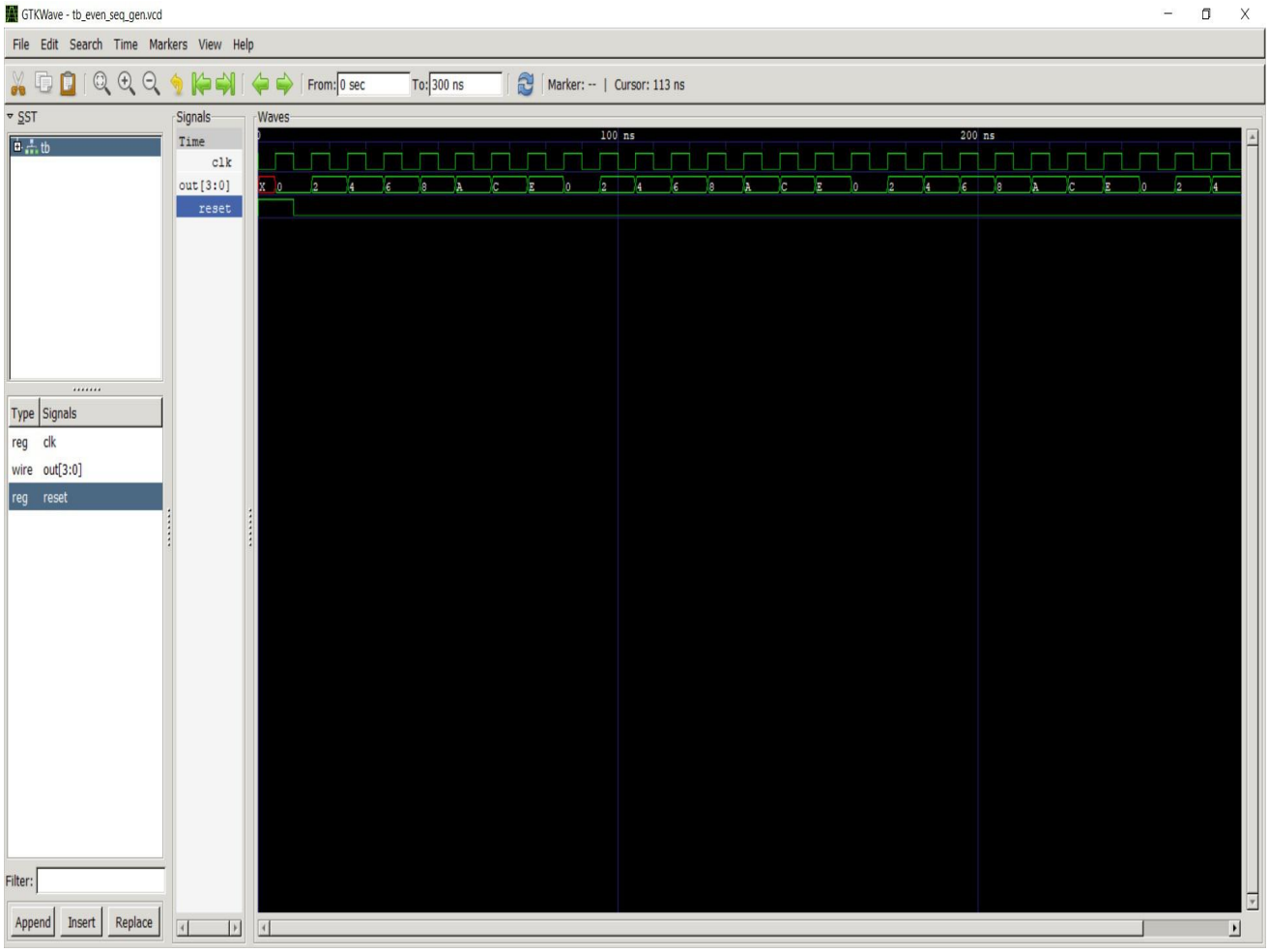
MODULES:

```
module invert(input wire ip, output wire out);
        assign out = ! ip;
endmodule

module and2(input wire ip0, ip1, output wire out);
        assign out = ip0 & ip1;
endmodule
```

```
module xor2(input wire ip0, ip1, output wire out);
	assign out = ip0 ^ ip1;
endmodule

module and3(input wire ip0, ip1, ip2, output wire out);
	wire t;
	and2 and2_0 (ip0, ip1, t);
	and2 and2_1 (ip2, t, out);
endmodule

module df(input wire clk, in, output wire out);
	reg df_out;
	always @( posedge clk) df_out <= in;
	assign out = df_out;
endmodule

module dfr(input wire clk, reset, in, output wire out);
	wire reset_, df_in;
	invert invert_0 (reset, reset_);
	and2 and2_0 (in, reset_, df_in);
	df df_0 (clk, df_in, out);
endmodule
```

```
module even_seq_gen(input wire clock, reset, output wire [3:0]out);
        assign out[0] = 1'b0 ;
        wire s1, s2, s3, c1;

        invert i_0 (out[1], s1);
        xor2 x_0 (out[2], out[1], s2);
        and2 a_0 (out[2], out[1], c1);
        xor2 x_1 (c1, out[3], s3);

        dfr d_0 (clock, reset, s1, out[1]);
        dfr d_1 (clock, reset, s2, out[2]);
        dfr d_2 (clock, reset, s3, out[3]);
endmodule
```

Output:


GTKWave - tb_even_seq_gen.vcd
File Edit Search Time Markers View Help
From: 0 sec
To: 300 ns
Marker: -- | Cursor: 113 ns
SST
tb
Type Signals
reg clk
wire out[3:0]
reg reset
Filter:
Append Insert Replace
Signals
Time
clk
out[3:0]
reset
Waves
100 ns
200 ns