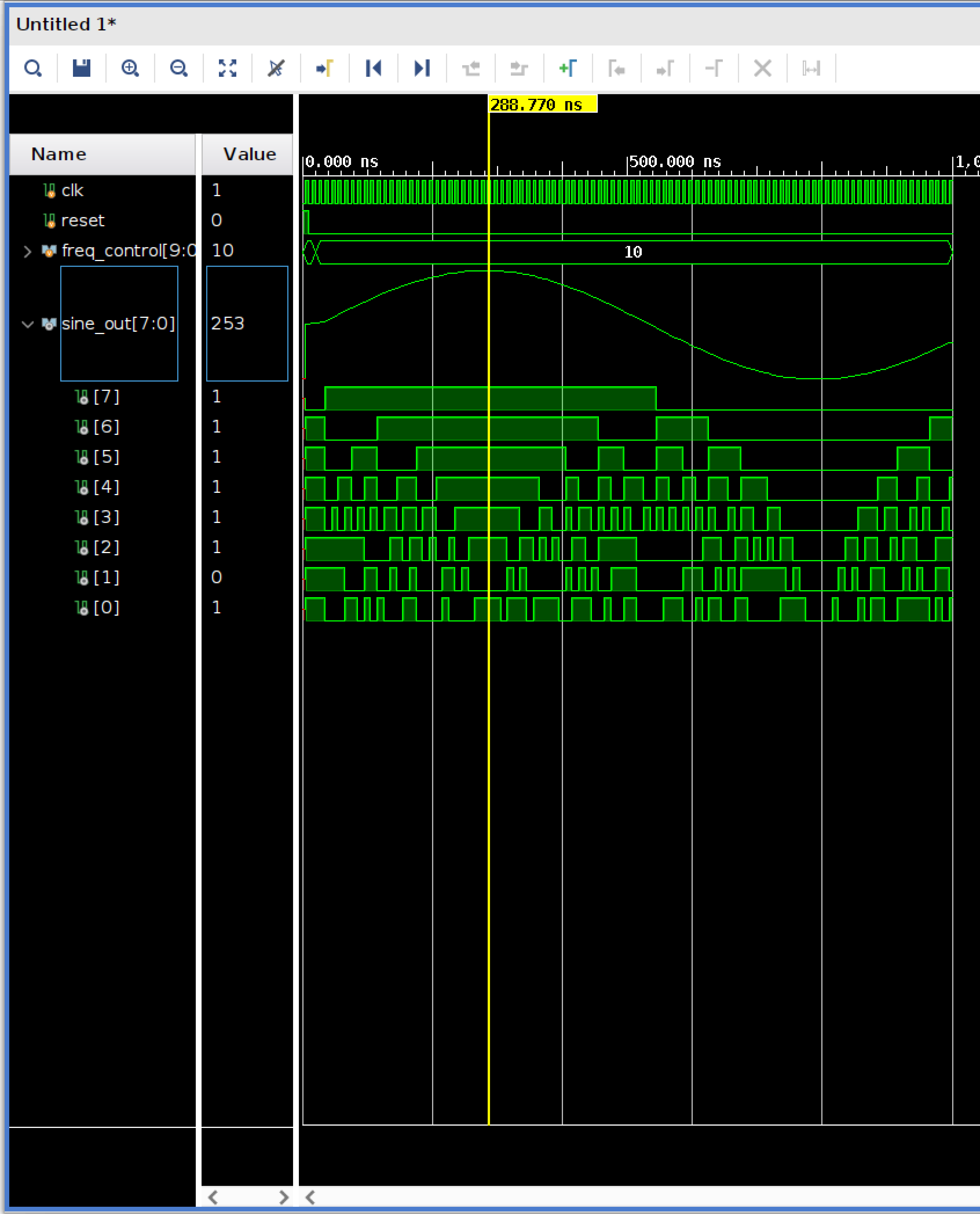
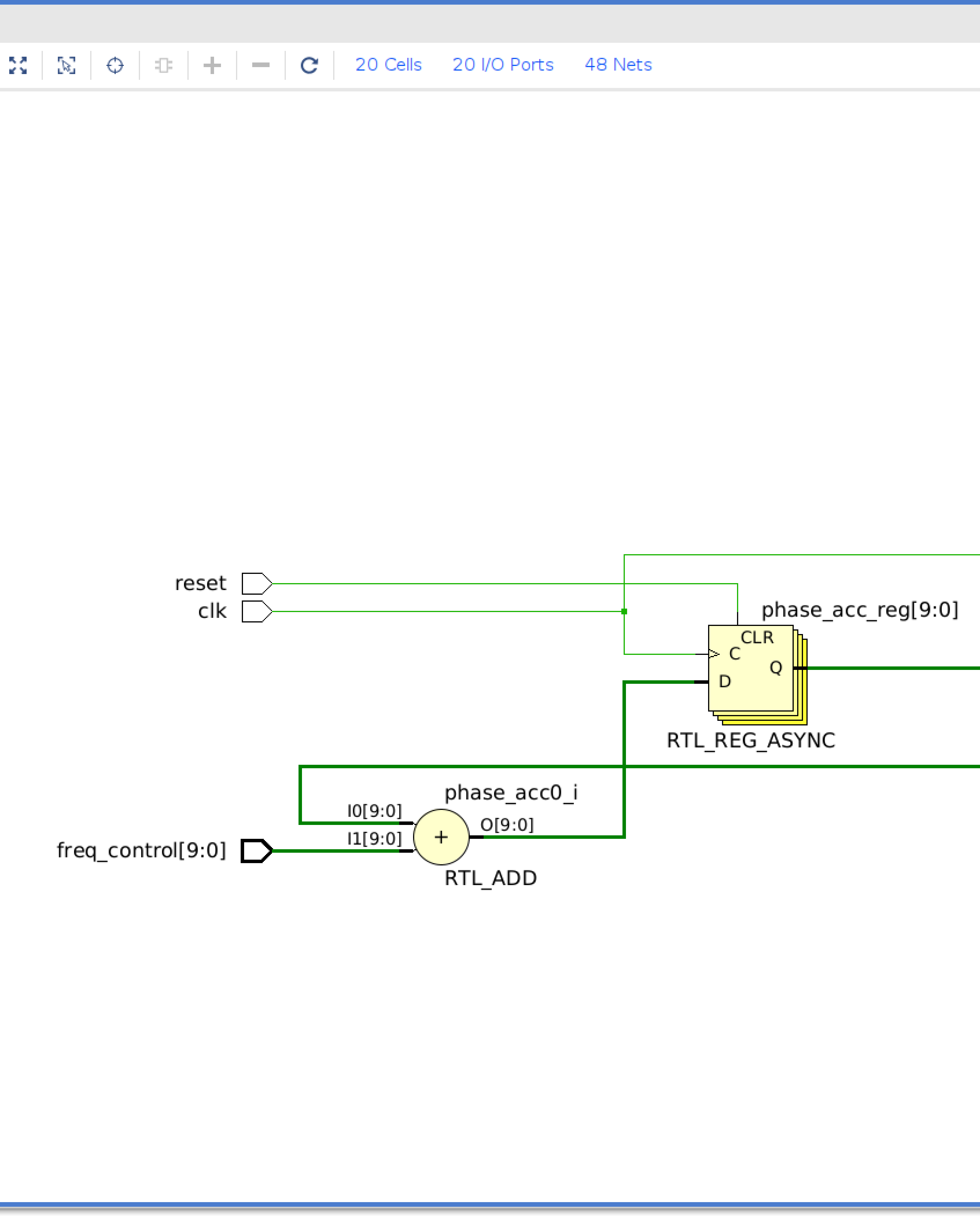
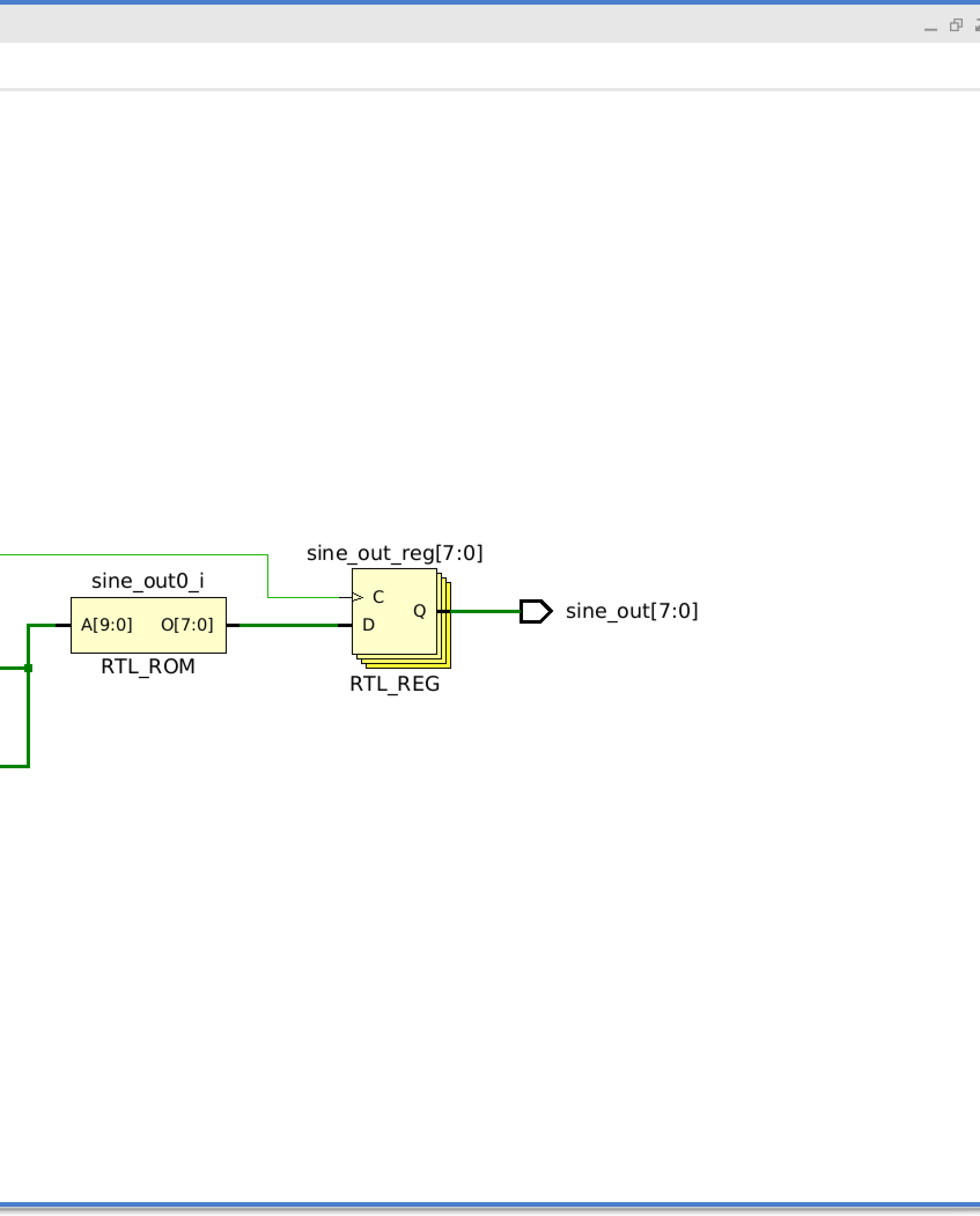
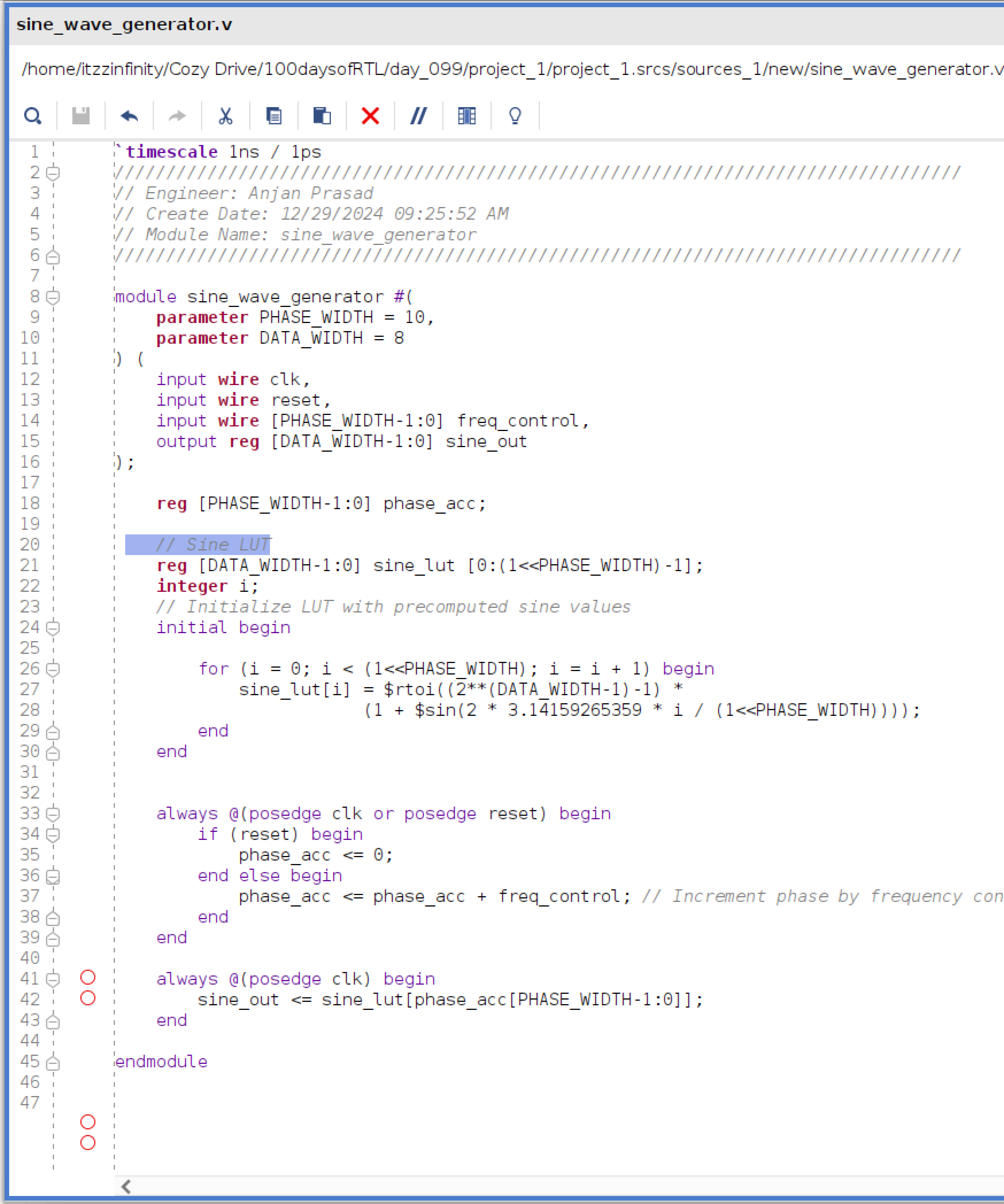

Untitled 1*

Name	Value
clk	1
reset	0
freq_control[9:0	10
sine_out[7:0]	253
[7]	1
[6]	1
[5]	1
[4]	1
[3]	1
[2]	1
[1]	0
[0]	1


20 Cells
20 I/O Ports
48 Nets
reset
clk
phase_acc_reg[9:0]
CLR
C
Q
D
RTL_REG_ASYNC
phase_acc0_i
I0[9:0]
I1[9:0]
+
O[9:0]
freq_control[9:0]
RTL_ADD

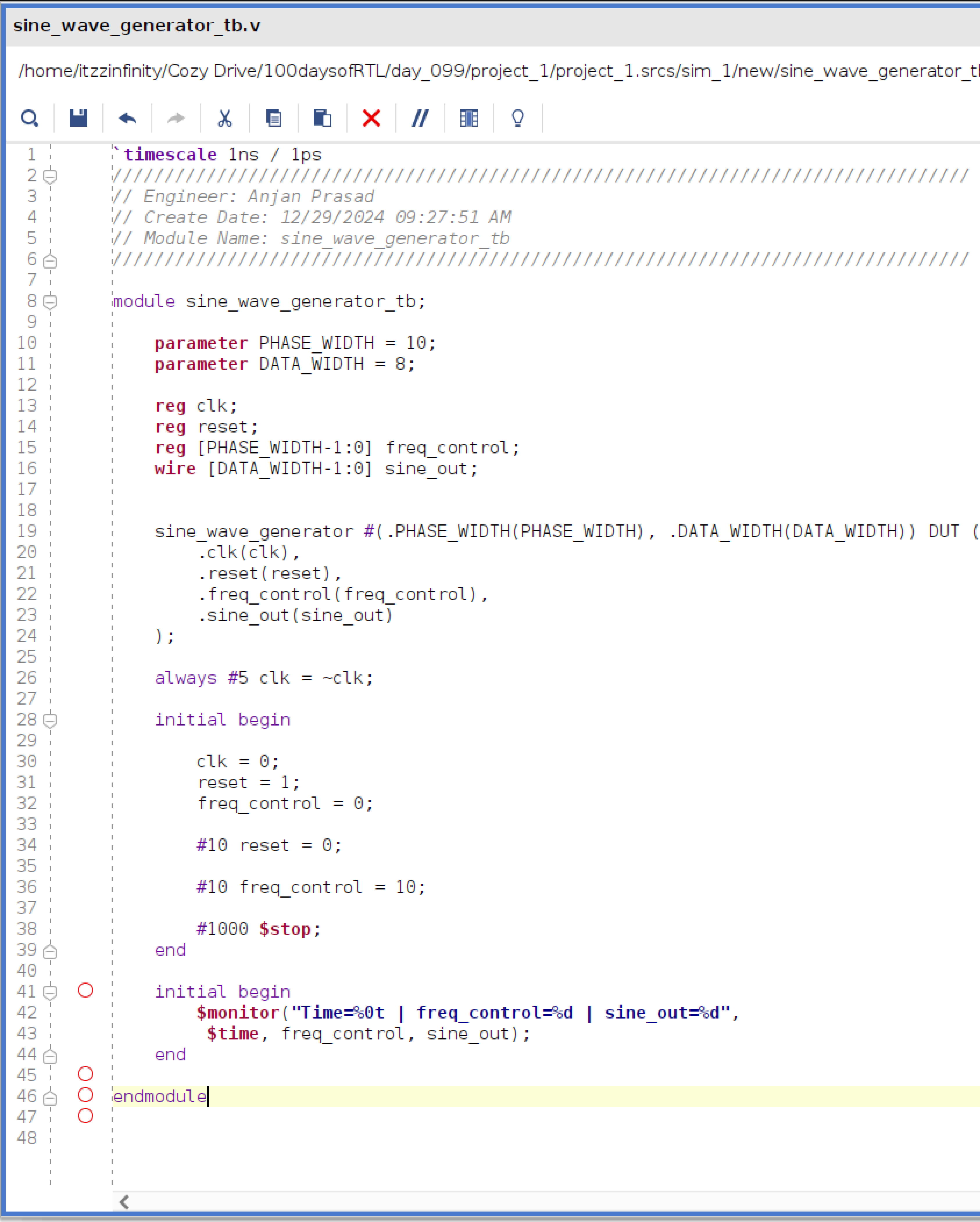

sine_out0_i
A[9:0]
O[7:0]
RTL_ROM
sine_out_reg[7:0]
C
D
Q
RTL_REG
sine_out[7:0]



sine_wave_generator.v

/home/itzzinfinity/Cozy Drive/100daysofRTL/day_099/project_1/project_1.srcs/sources_1/new/sine_wave_generator.v

```verilog
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Engineer: Anjan Prasad
// Create Date: 12/29/2024 09:25:52 AM
// Module Name: sine_wave_generator
//////////////////////////////////////////////////////////////////////////////////

module sine_wave_generator #(
    parameter PHASE_WIDTH = 10,
    parameter DATA_WIDTH = 8
) (
    input wire clk,
    input wire reset,
    input wire [PHASE_WIDTH-1:0] freq_control,
    output reg [DATA_WIDTH-1:0] sine_out
);

    reg [PHASE_WIDTH-1:0] phase_acc;

    // Sine LUT
    reg [DATA_WIDTH-1:0] sine_lut [0:(1<<PHASE_WIDTH)-1];
    integer i;
    // Initialize LUT with precomputed sine values
    initial begin

        for (i = 0; i < (1<<PHASE_WIDTH); i = i + 1) begin
            sine_lut[i] = $rtoi((2**(DATA_WIDTH-1)-1) *
                                (1 + $sin(2 * 3.14159265359 * i / (1<<PHASE_WIDTH))));
        end
    end


    always @(posedge clk or posedge reset) begin
        if (reset) begin
            phase_acc <= 0;
        end else begin
            phase_acc <= phase_acc + freq_control; // Increment phase by frequency con
        end
    end

    always @(posedge clk) begin
        sine_out <= sine_lut[phase_acc[PHASE_WIDTH-1:0]];
    end

endmodule
```



sine_wave_generator_tb.v

/home/itzzinfinity/Cozy Drive/100daysofRTL/day_099/project_1/project_1.srcs/sim_1/new/sine_wave_generator_t

```verilog
`timescale 1ns / 1ps
/////////////////////////////////////////////////////////////////////////////////
// Engineer: Anjan Prasad
// Create Date: 12/29/2024 09:27:51 AM
// Module Name: sine_wave_generator_tb
/////////////////////////////////////////////////////////////////////////////////

module sine_wave_generator_tb;

    parameter PHASE_WIDTH = 10;
    parameter DATA_WIDTH = 8;

    reg clk;
    reg reset;
    reg [PHASE_WIDTH-1:0] freq_control;
    wire [DATA_WIDTH-1:0] sine_out;


    sine_wave_generator #(.PHASE_WIDTH(PHASE_WIDTH), .DATA_WIDTH(DATA_WIDTH)) DUT (
        .clk(clk),
        .reset(reset),
        .freq_control(freq_control),
        .sine_out(sine_out)
    );

    always #5 clk = ~clk;

    initial begin

        clk = 0;
        reset = 1;
        freq_control = 0;

        #10 reset = 0;

        #10 freq_control = 10;

        #1000 $stop;
    end

    initial begin
        $monitor("Time=%0t | freq_control=%d | sine_out=%d",
         $time, freq_control, sine_out);
    end

endmodule
```