Project 2 : UART Communication to print a single character

1. Design :-

```
//module declaration
module uart_transmit(
        //inputs
         input clk,
         input rst,
         input uart_en,
         //outputs
         output reg uart_tx,
         output uart_busy
   );
         //Identifier Declaration
         reg[3:0] bitcount;
         reg[8:0] shift;
         reg[7:0] data = "v";
         wire sending;
         //Baud rate as 115200Hz
         reg [28:0] clk_count;
         wire [28:0] clk_inc_count = clk_count[28] ? (115200) : (115200-100000000);
         wire [28:0] clk_inc = clk_count + clk_inc_count;
         wire gen_clk;
         //Generating clock
         always@(posedge clk)
              clk_count <= clk_inc;
         assign gen_clk = clk_count[28]; //115200Hz
        assign uart_busy = |bitcount[3:1];
        assign sending = |bitcount;
   //UART Transmission block
   always@(posedge clk) begin
```

```
    if(rst) begin
      uart_tx <= 1'b1;
      bitcount <= 0;
      shift <= 0;
    end else begin
      //Shifter init
      if(uart_en & ~uart_busy) begin
        bitcount <= (1 + 8 + 1 + 1);
        shift <= {data[7:0], 1'b0};
      end
      //Data Transmission
      if(sending & gen_clk) begin
        {shift,        uart_tx} <= {1'h1, shift};
                          bitcount <= bitcount - 1;
              end
            end
      end
endmodule
```

NOTE:- Here for the software simulation according to our test bench we have to keep the '**data**' as variable without any given value. Its value will be assign from its testbench.

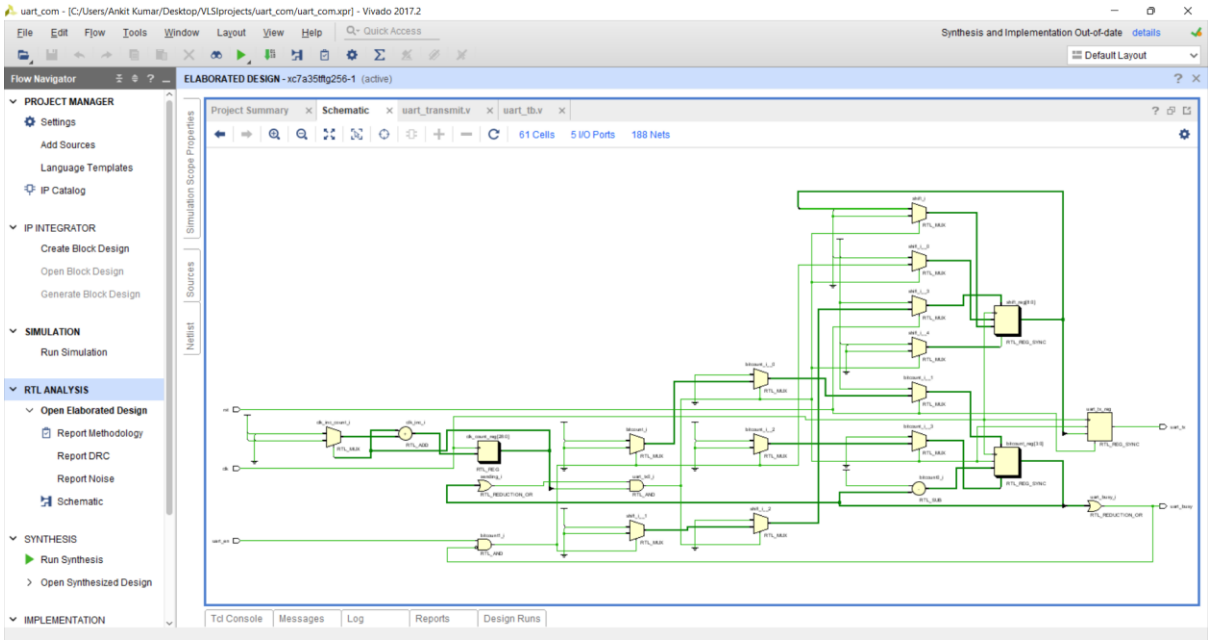
2. Test bench :-

```
module uart_tb;
      reg clk;
      reg rst;
      reg uart_en;
      wire uart_tx;
      wire uart_busy;
      reg[7:0] data;
      // Instantiate Unit Under Test (UUT)
```

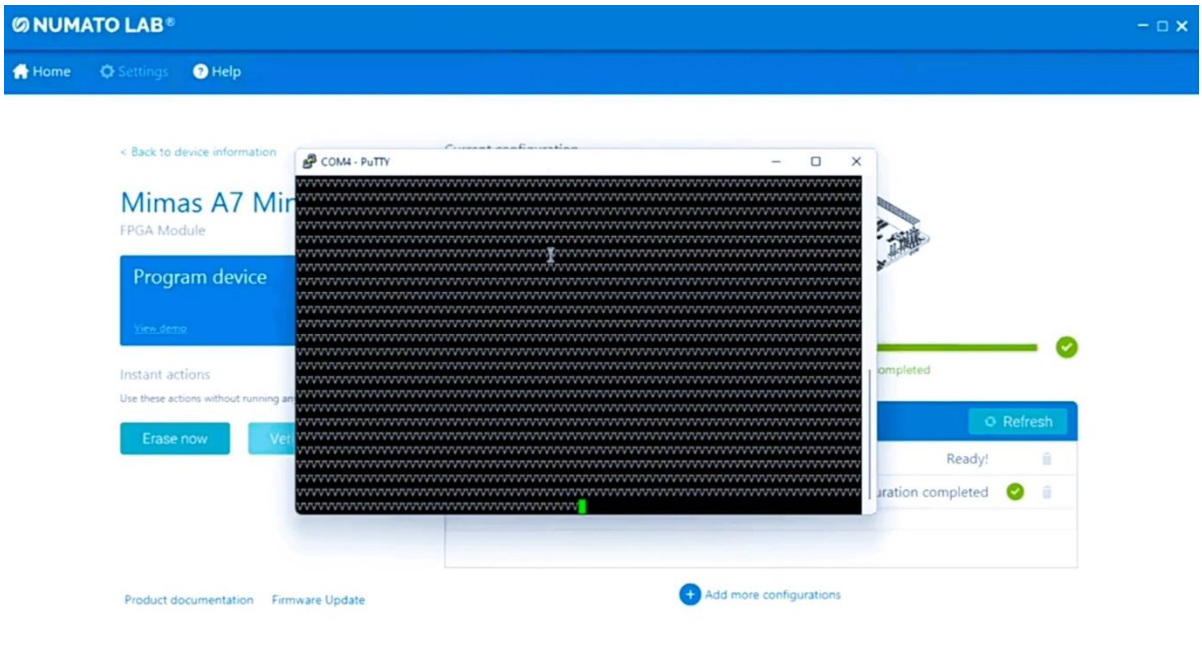
```
    uart_transmit uut(
            .clk(clk),
            .rst(rst),
            .uart_en(uart_en),
            .uart_tx(uart_tx),
            .uart_busy(uart_busy)
    );
    // 16x 115200 BAUD clock
    always begin
            #271.27
            clk = 1'b0;
            #271.27
            clk = 1'b1;
    end
    initial begin
            // initialize inputs
            clk = 1;
            rst = 1;
            uart_en = 0;
            #100
            rst = 0;
            #100
            uart_en = 1;
// send a character
            send(8'h75);
    end
    // send data
    task send;
            input [7:0] char;
            begin
```

```
                wait(!clk);
                wait(clk);
                #1
                data = {char};
                // write to transmitter
                wait(!clk);
                wait(clk);
                #1 uart_en = 1;
                wait(!clk);
                wait(clk); #1 uart_en = 0;
                // release bus
                wait(!clk);
                wait(clk);
                #1
                 data = 8'hx;
            end
        endtask
endmodule
```

3. Outputs :-

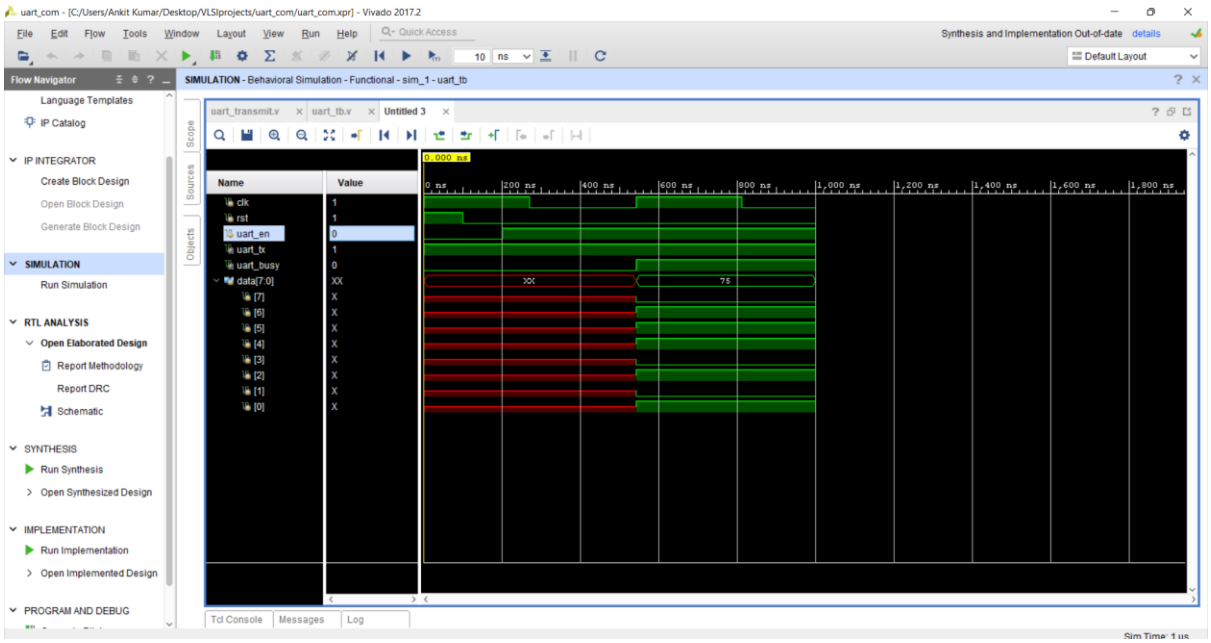


:- RTL Viewer



:- Hardware Implementation

4. Waveform :-



Note :- Here simulation wave form will be according to given test bench while, Hardware implementation on board will be according to bitstream and design code.