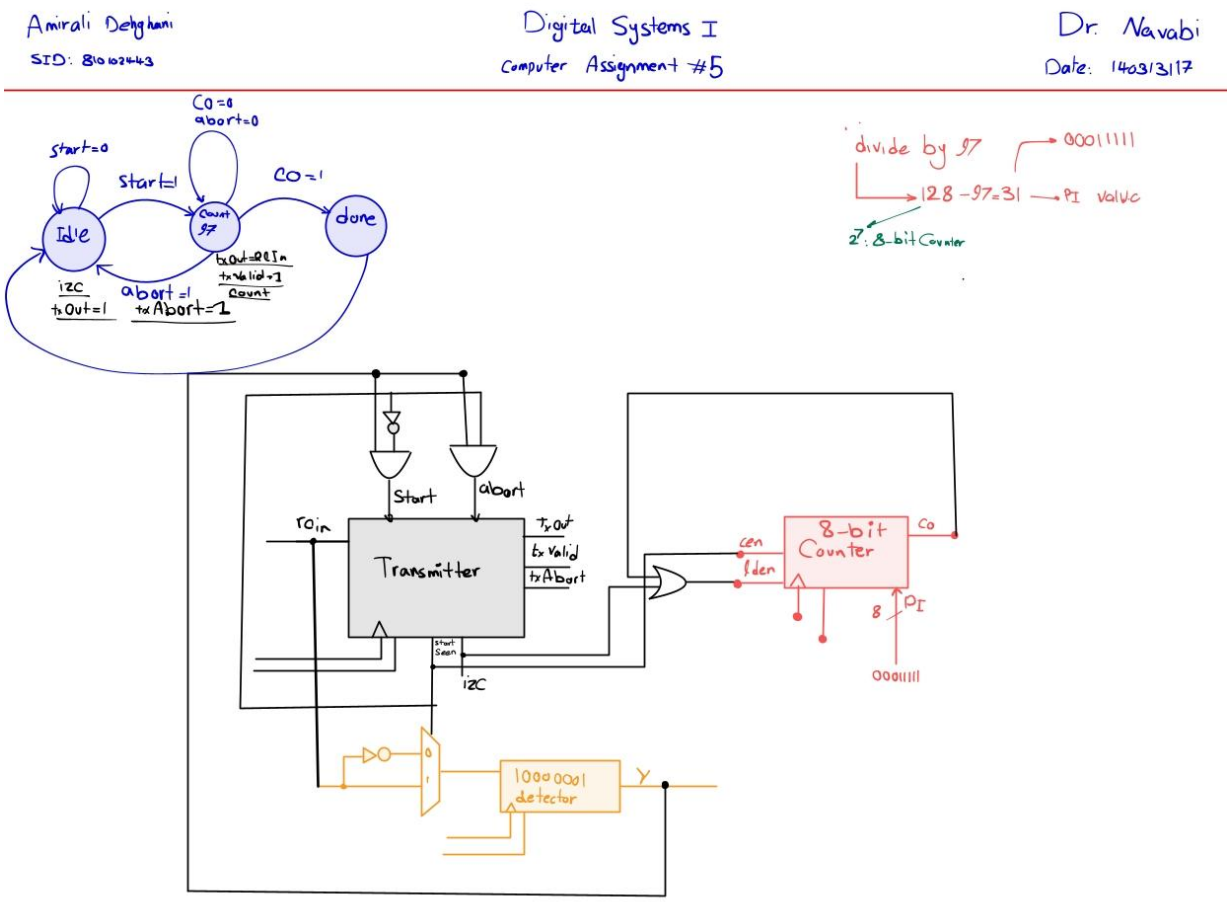
Digital Systems I

Computer Assignment #5

Synthesis of Counters, Shifters and State Machines

Amirali Dehghani

SID: 810102443



Codes

Instead of schematics, I described it in Verilog and then synthesized it in Quartus. At first I implemented the 7 bit counter(In code I realized I need 7 bit not 8 bit.) and signals like what I did in last CA for 3-bit counter then I draw the Transmitter block and signals using the FSM and after that, I connected it to the counter. At the end that I had a complete block of this, I

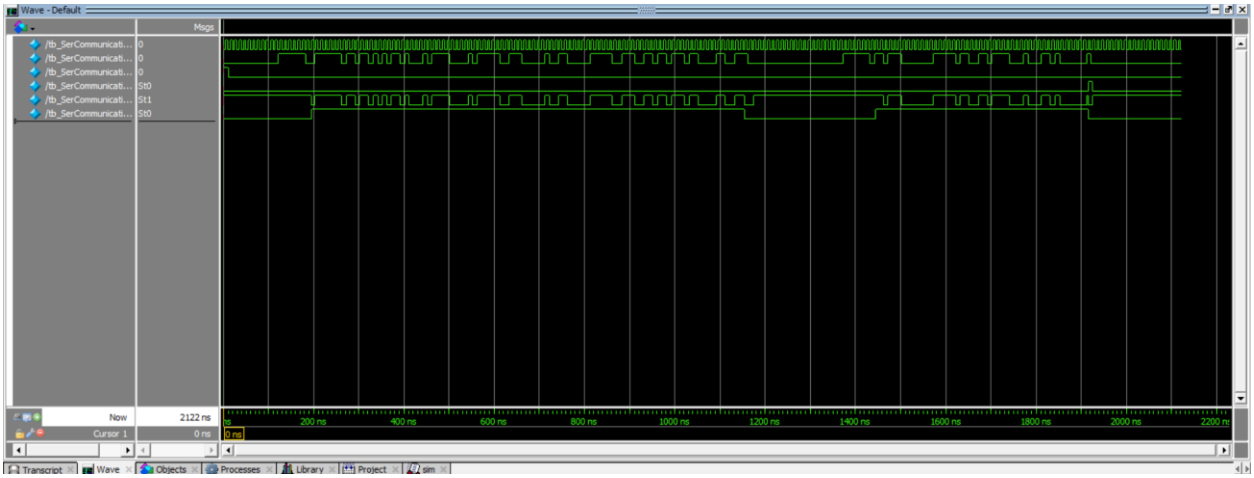
connected it to 10000001 detector from last CA and connect signals and wires to each other for the full circuit. I just described this in Verilog.

```verilog
`timescale 1ns/1ns
module Counter_8(input wire [6:0] PI, input wire clk, rst, ci, cen, iz, output reg [6:0] PO, output wire co);
    assign lden = iz || co;
    always @(posedge clk, posedge rst) begin
        if (rst) PO <= 3'd0;
        else if (lden) PO <= PI;
        else if (cen) PO <= PO + ci;
    end
    assign co = &PO;
endmodule
```

```verilog
`timescale 1ns/1ns
module Transmitter(input wire clk, rst, RcIn, start, abort, output reg txOut, txValid, txAbort, startSeen);
    parameter[1:0] Idle = 0, Count97 = 1, Done = 2;
    reg [1:0] PS, NS;
    reg [6:0] init_val = 7'b0011111;
    wire [6:0] cnt_val;
    reg izc, ld;
    wire co;
    assign ld = izc || co;
    Counter_8 cntr(.PI(init_val), .clk(clk), .rst(rst), .ci(1'b1), .cen(startSeen), .iz(ld), .PO(cnt_val), .co(co));
    always @(PS, NS, start, abort, co, RcIn) begin
        NS = Idle;
        {izc, txOut, txValid, txAbort, startSeen} = 5'b0;
        case (PS)
            Idle: begin
                NS = start ? Count97 : Idle;
                izc = 1'b1;
                txOut = 1'b1;
            end
            Count97: begin
                if (co) begin
                    NS = Done;
                end
                else if (abort) begin
                    NS = Idle;
                    txAbort = 1'b1;
                end
                else begin
                    NS = Count97;
                    txOut = RcIn;
                    txValid = 1'b1;
                end
                startSeen = 1'b1;
            end
            Done: NS = Idle;
            default: NS = Idle;
        endcase
    end

    always @(posedge clk, posedge rst) begin
        if (rst) PS <= Idle;
        else PS <= NS;
    end
endmodule

```

```
`timescale 1ns/1ns
module SerCommunication(input wire clk, rst, RcIn, output wire txOut, txValid, txAbort);
    wire startSeen, detectorOut;
    wire start = detectorOut && ~startSeen, abort = detectorOut && startSeen;
    wire selectedRcIn = startSeen ? RcIn : ~RcIn;
    Transmitter transmitter(.clk(clk), .rst(rst), .RcIn(RcIn),
    .start(start), .abort(abort), .startSeen(startSeen),
    .txOut(txOut), .txAbort(txAbort), .txValid(txValid));
    s10000001detector seqDetector(.clk(clk), .rst(rst),
    .J(selectedRcIn), .Y(detectorOut));
endmodule
```

Post Synthesis Waveform.

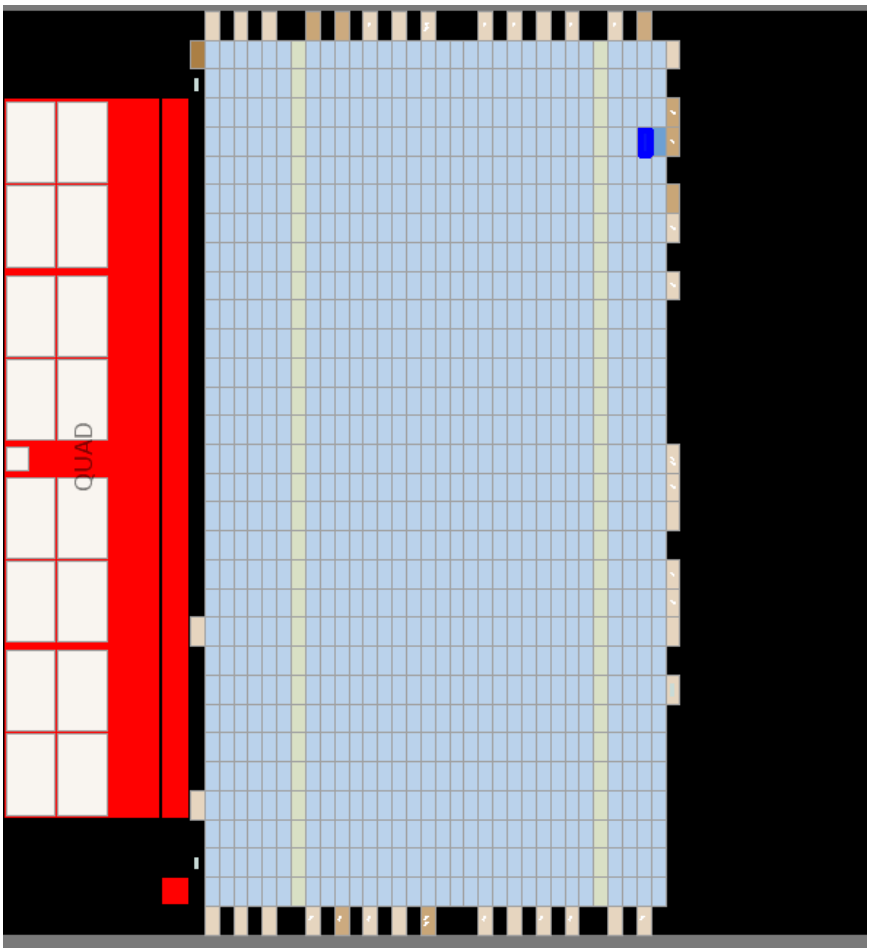


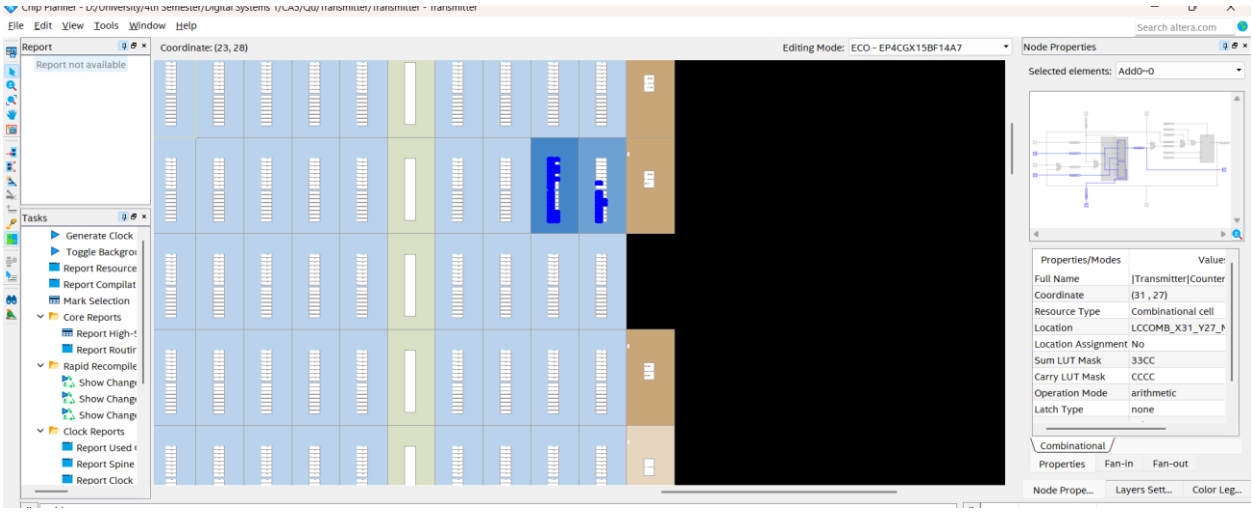
Transmitter in Quartus:

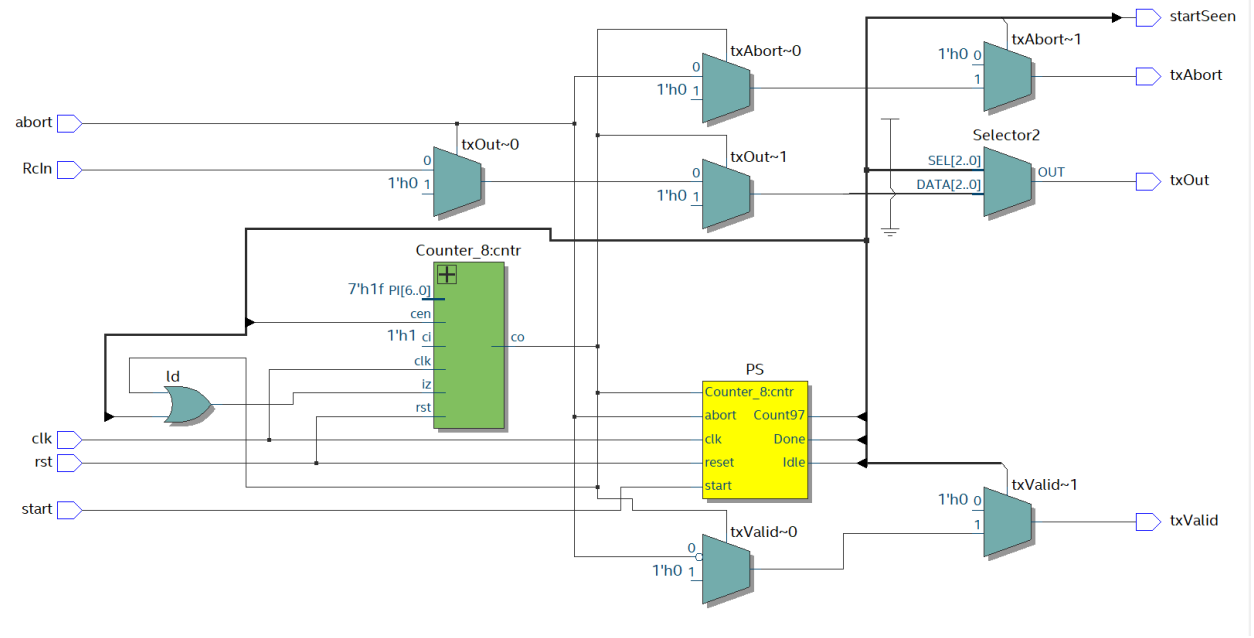
Flow Summary

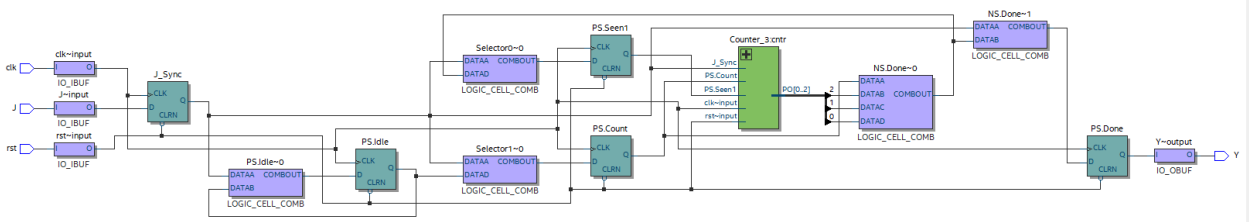
<<Filter>>

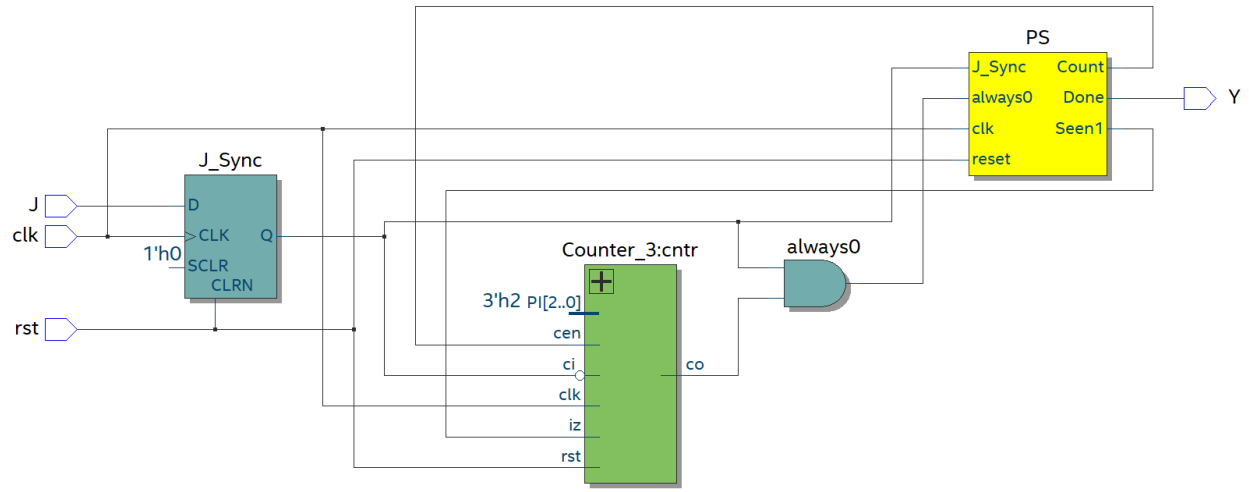
Flow Status	Successful - Sun Jun 08 03:56:20 2025
Quartus Prime Version	20.1.0 Build 711 06/05/2020 SJ Lite Edition
Revision Name	Transmitter
Top-level Entity Name	Transmitter
Family	Cyclone IV GX
Device	EP4CGX15BF14A7
Timing Models	Final
Total logic elements	25 / 14,400 (< 1 %)
Total registers	9
Total pins	9 / 81 (11 %)
Total virtual pins	0
Total memory bits	0 / 552,960 (0 %)
Embedded Multiplier 9-bit elements	0
Total GXB Receiver Channel PCS	0 / 2 (0 %)
Total GXB Receiver Channel PMA	0 / 2 (0 %)
Total GXB Transmitter Channel PCS	0 / 2 (0 %)
Total GXB Transmitter Channel PMA	0 / 2 (0 %)
Total PLLs	0 / 3 (0 %)



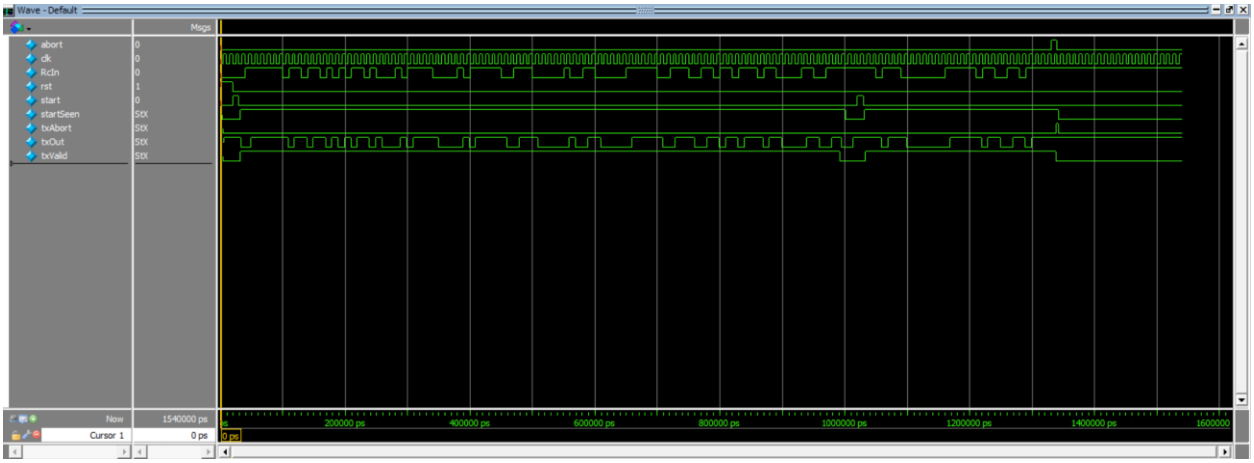

File Edit View Tools Window Help
Report
Report not available
Coordinate: (23, 28)
Editing Mode: ECO - EP4CGX15BF14A7
Node Properties
Selected elements: Add0~0
Tasks
Generate Clock
Toggle Backgro
Report Resource
Report Compilat
Mark Selection
Core Reports
Report High-
Report Routir
Rapid Recompile
Show Change
Show Change
Show Change
Clock Reports
Report Used
Report Spine
Report Clock
Properties/Modes
Value
Full Name
|Transmitter|Counter
Coordinate
(31 , 27)
Resource Type
Combinational cell
Location
LCCOMB_X31_Y27_
Location Assignment
No
Sum LUT Mask
33CC
Carry LUT Mask
CCCC
Operation Mode
arithmetic
Latch Type
none
Combinational
Properties
Fan-in
Fan-out
Node Prope...
Layers Sett...
Color Leg...


abort
RcIn
clk
rst
start
txOut~0
txOut~1
txAbort~0
txAbort~1
txValid~0
txValid~1
Selector2
SEL[2..0]
DATA[2..0]
OUT
startSeen
txAbort
txOut
txValid
Counter_8:cntr
7'h1f PI[6..0]
cen
1'h1 ci
clk
iz
rst
co
ld
PS
Counter_8:cntr
abort
clk
reset
start
Count97
Done
Idle
1'h0
1'h1


clk
clk~input
IO_IBUF
J
J~input
IO_IBUF
rst
rst~input
IO_IBUF
J_Sync
CLK
D
Q
CLRN
PS.Idle~0
DATAA COMBOUT
DATAB
LOGIC_CELL_COMB
PS.Idle
Selector0~0
DATAA COMBOUT
DATAD
LOGIC_CELL_COMB
Selector1~0
DATAA COMBOUT
DATAD
LOGIC_CELL_COMB
PS.Seen1
PS.Count
Counter_3:cntr
J_Sync
PS.Count
PS.Seen1
clk~input
rst~input
PO[0..2]
2
1
0
NS.Done~0
DATAA
DATAB COMBOUT
DATAC
DATAD
LOGIC_CELL_COMB
NS.Done~1
DATAA COMBOUT
DATAB
LOGIC_CELL_COMB
PS.Done
Y~output
IO_OBUF
Y

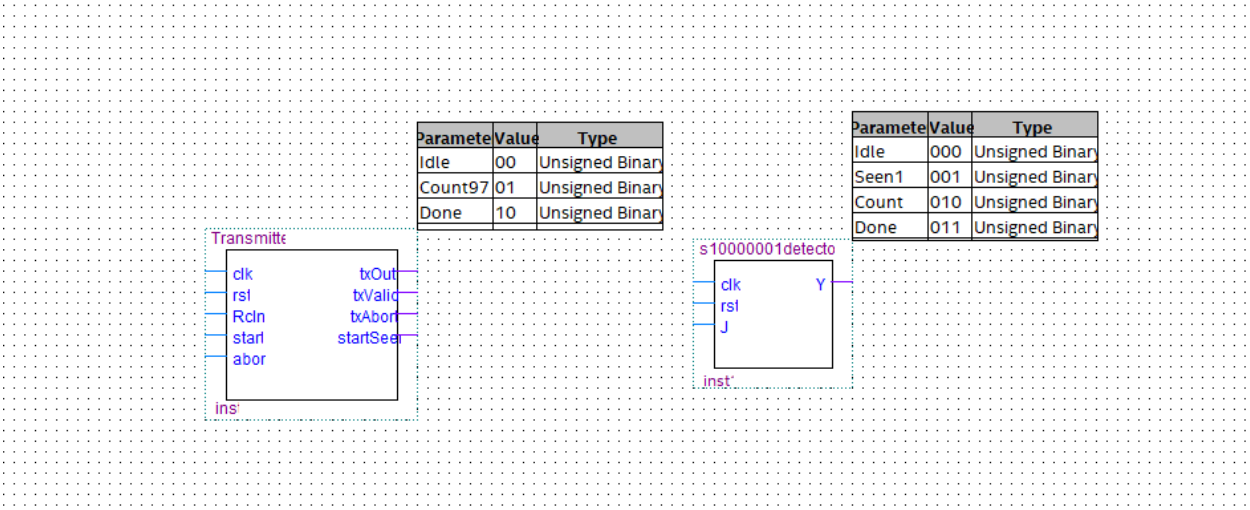

PS
J_Sync
always0
clk
reset
Count
Done
Seen1
Y
J_Sync
J
clk
rst
D
CLK
Q
1'h0
SCLR
CLRN
Counter_3:cntr
3'h2
PI[2..0]
cen
ci
clk
iz
rst
co
always0

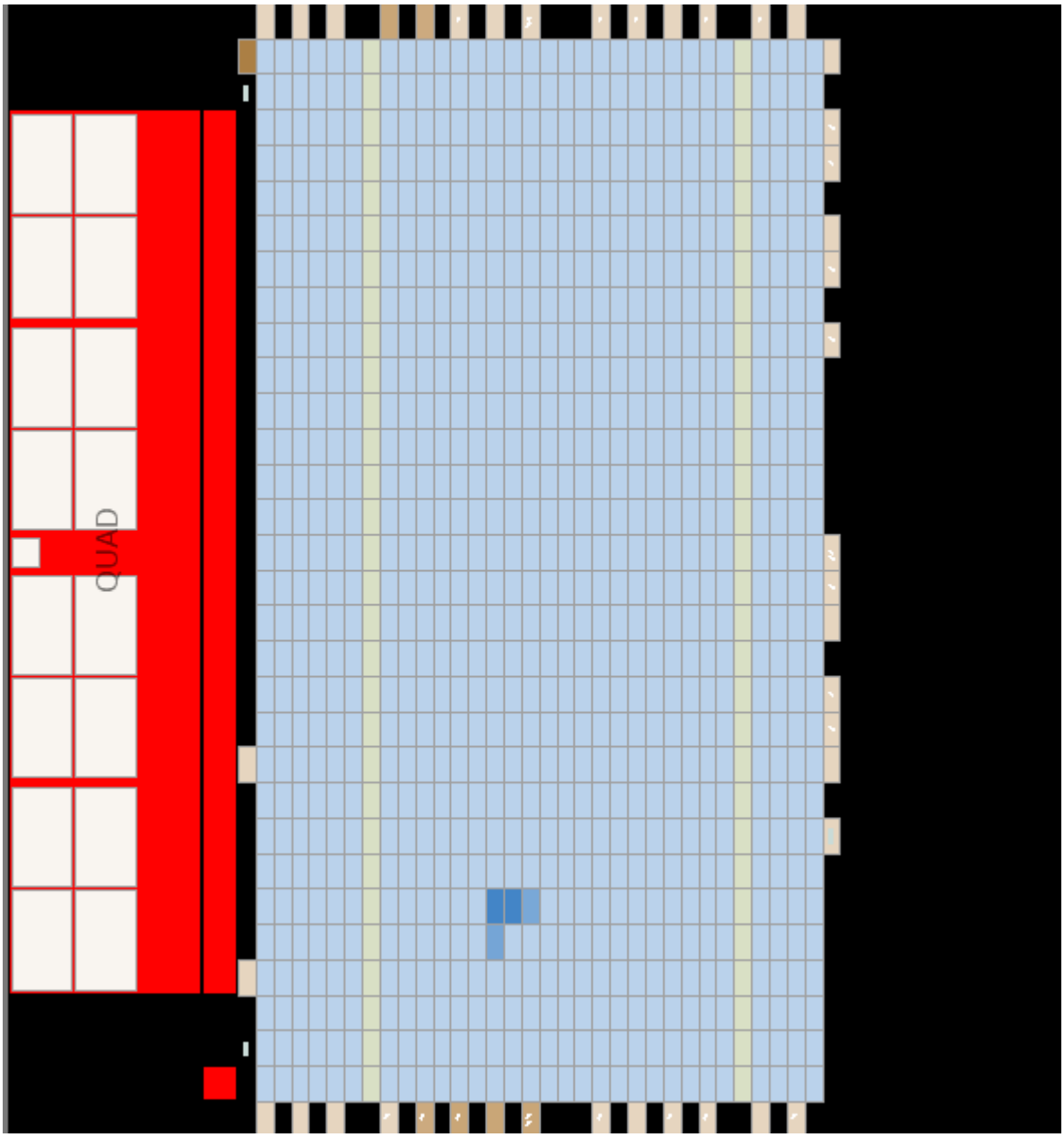
```
`timescale 1ns/1ns
module TB_Transmitter;
    reg clk, rst, RcIn, start, abort;
    wire txOut, txValid, txAbort, startSeen;

    Transmitter DUT (
        .clk(clk),
        .rst(rst),
        .RcIn(RcIn),
        .start(start),
        .abort(abort),
        .txOut(txOut),
        .txValid(txValid),
        .txAbort(txAbort),
        .startSeen(startSeen)
    );

    initial clk = 0;
    always #5 clk = ~clk;

    initial begin
        rst   = 1;
        RcIn  = 0;
        start = 0;
        abort = 0;
        #20 rst = 0;

        start = 1; #10 start = 0;
        repeat (97) begin
            RcIn = $random;
            #10;
        end
        #20;

        start = 1; #10 start = 0;
        repeat (30) begin
            RcIn = $random;
            #10;
        end
        abort = 1; #10 abort = 0;
        repeat (20) #10;

        $stop;
    end
endmodule
```

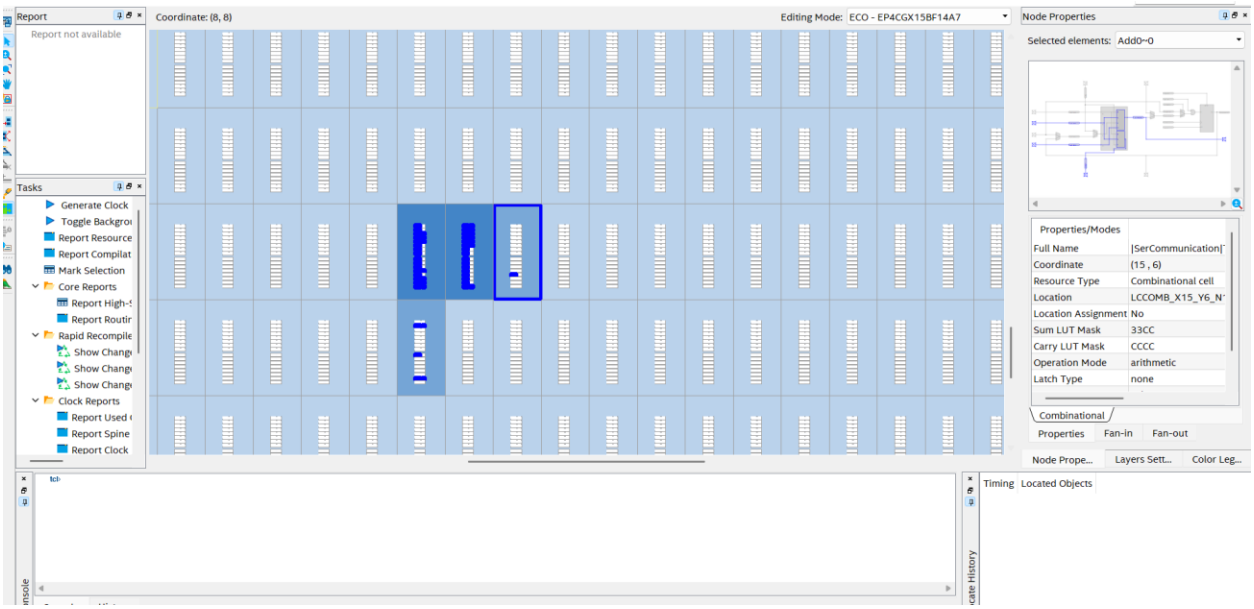


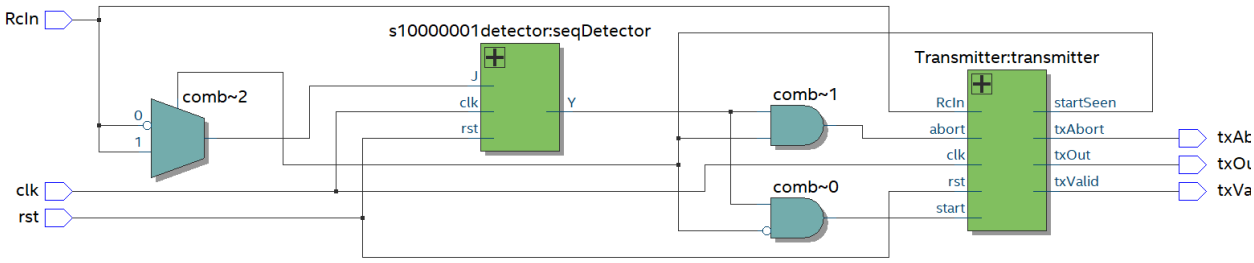
Serial Communicator in Quartus:

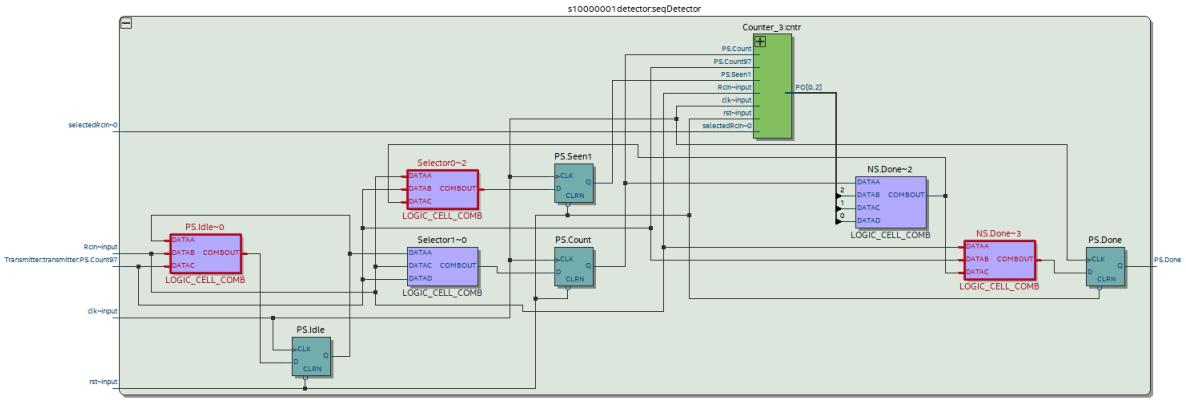
(I used the code for synthesis)



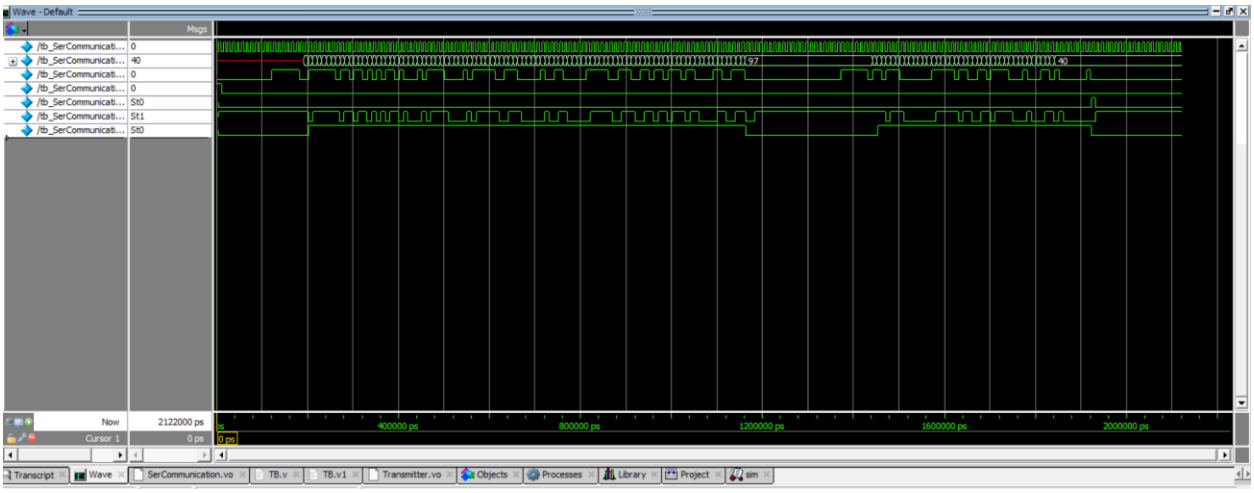

QUAD


Report
Report not available
Coordinate: (8, 8)
Editing Mode: ECO - EP4CGX15BF14A7
Node Properties
Selected elements: Add0~0
Tasks
Generate Clock
Toggle Backgro
Report Resource
Report Compilat
Mark Selection
Core Reports
Report High-
Report Routir
Rapid Recompile
Show Change
Show Change
Show Change
Clock Reports
Report Used
Report Spine
Report Clock
Properties/Modes
Full Name
[SerCommunication]
Coordinate
(15 , 6)
Resource Type
Combinational cell
Location
LCCOMB_X15_Y6_N
Location Assignment
No
Sum LUT Mask
33CC
Carry LUT Mask
CCCC
Operation Mode
arithmetic
Latch Type
none
Combinational
Properties
Fan-in
Fan-out
Node Prope...
Layers Sett...
Color Leg...
Timing
Located Objects


RcIn
clk
rst
comb~2
0
1
s10000001detector:seqDetector
J
clk
rst
Y
comb~1
comb~0
Transmitter:transmitter
RcIn
abort
clk
rst
start
startSeen
txAbort
txOut
txValid




```
`timescale 1ns/1ns
module tb_SerCommunication;

    reg clk, rst, RcIn;
    wire txOut, txValid, txAbort;

    SerCommunication dut (
        .clk(clk),
        .rst(rst),
        .RcIn(RcIn),
        .txOut(txOut),
        .txValid(txValid),
        .txAbort(txAbort)
    );

    always #5 clk = ~clk;

    task send_bit;
        input reg b;
        begin
            RcIn = b;
            #10;
        end
    endtask

    task send_sequence;
        input [7:0] seq;
        integer i;
        begin
            for (i = 7; i >= 0; i = i - 1)
                send_bit(seq[i]);
        end
    endtask

    integer i;
    initial begin
        clk = 0;
        RcIn = 0;
        rst = 1;
        #12 rst = 0;
        repeat (10) send_bit(0);

        send_sequence(8'b01111110);

        for (i = 0; i < 97; i = i + 1)
            send_bit($random % 2);

        repeat (20) send_bit(0);

        send_sequence(8'b01111110);

        for (i = 0; i < 40; i = i + 1)
            send_bit($random % 2);

        send_sequence(8'b10000001);

        repeat (20) send_bit(0);
        $stop;
    end

endmodule

```


Wave - Default
Msgs
/tb_SerCommunicati... 0
/tb_SerCommunicati... 40
/tb_SerCommunicati... 0
/tb_SerCommunicati... 0
/tb_SerCommunicati... St0
/tb_SerCommunicati... St1
/tb_SerCommunicati... St0
Now
2122000 ps
Cursor 1
0 ps
400000 ps
800000 ps
1200000 ps
1600000 ps
2000000 ps
Transcript
Wave
SerCommunication.vo
TB.v
TB.v1
Transmitter.vo
Objects
Processes
Library
Project
sim