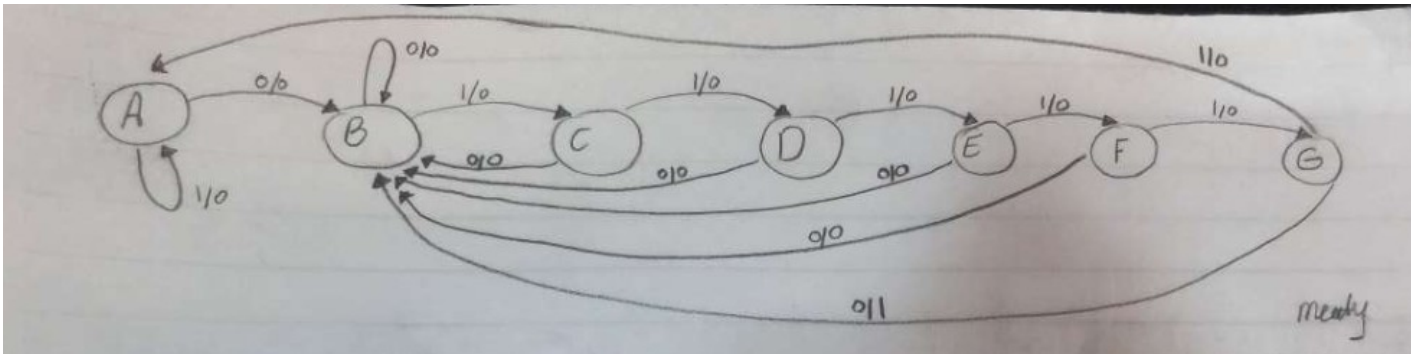
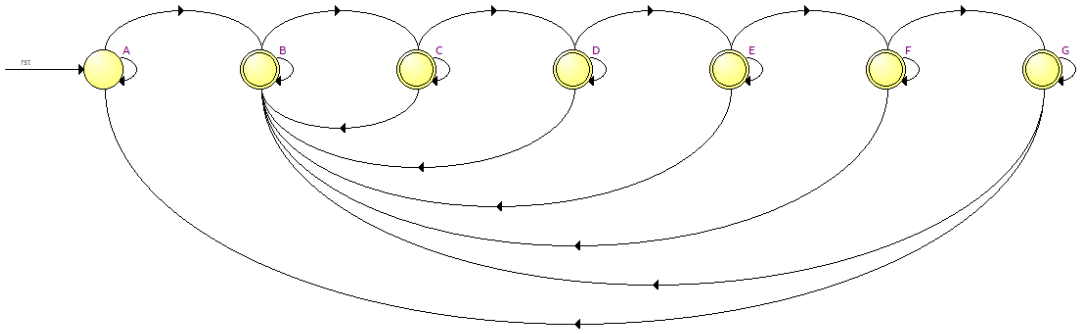
Part a:

Diagram and state machine for mealy machine:



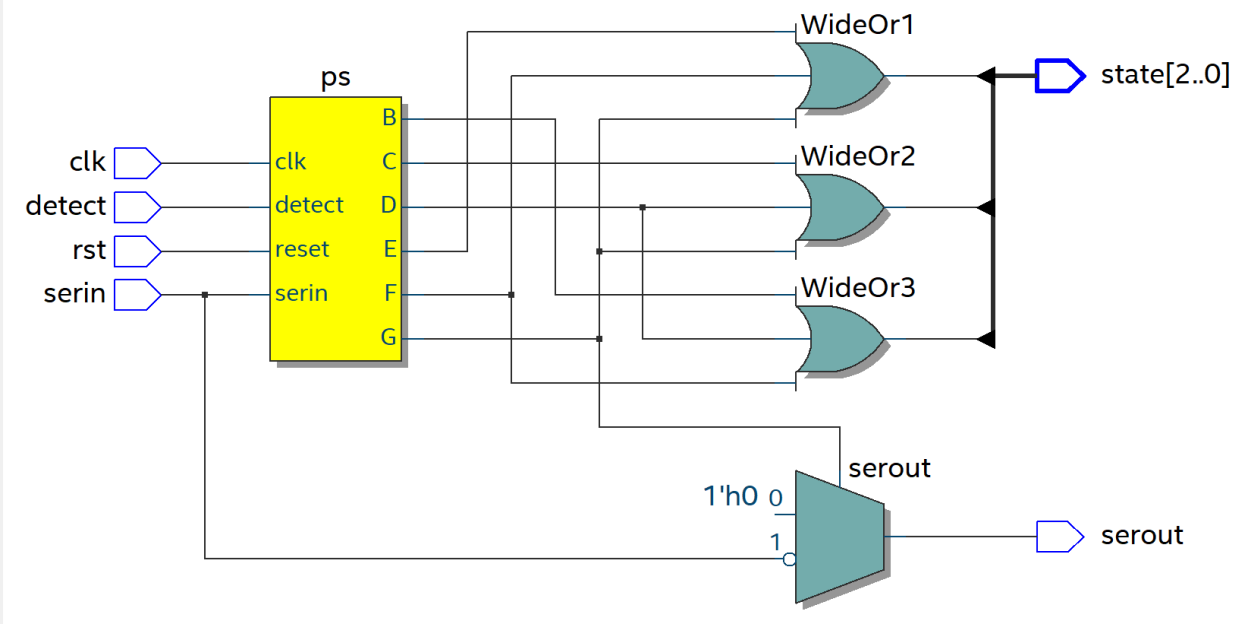
state	Ser_in = 0	Ser_in = 1	Ser_out = 0	Ser_out = 1
000 A	B	A	0	0
001 B	B	C	0	0
010 C	B	D	0	0
011 D	B	E	0	0
100 E	B	F	0	0
101 F	B	G	0	0
110 G	B	A	1	0
	$D_2 D_1 D_0$		W	

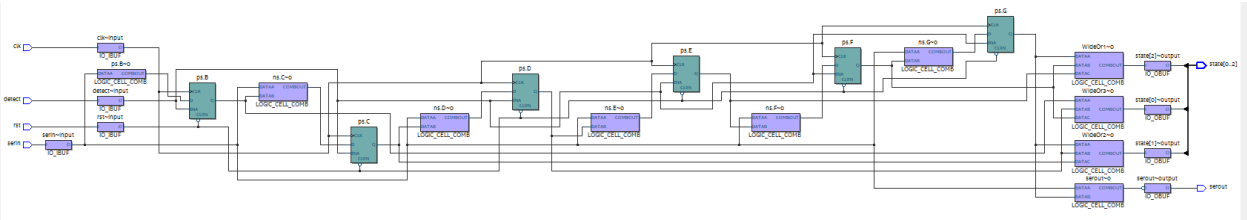


Mealy Machine code and test bench:

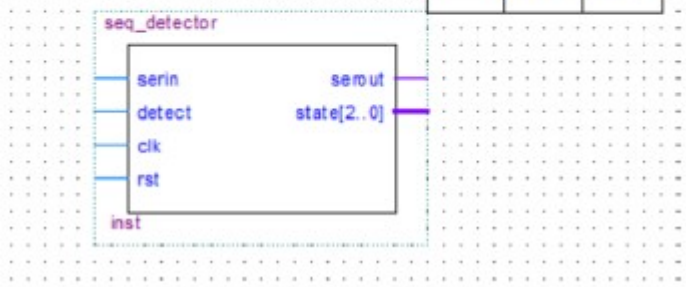
```
1  `timescale 1ns/1ns
2  module seq_detector(input serin,detect,clk,rst, output serout , output [2:0] state);
3      reg[2:0]ns,ps;
4      parameter [2:0] A = 3'b000, B = 3'b001, C = 3'b010, D = 3'b011, E = 3'b100, F = 3'b101, G = 3'b110;
5
6      always @(ps,serin) begin
7          ns=A;
8          case(ps)
9              A:ns = serin ? A : B;
10             B:ns = serin ? C : B;
11             C:ns = serin ? D : B;
12             D:ns = serin ? E : B;
13             E:ns = serin ? F : B;
14             F:ns = serin ? G : B;
15             G:ns = serin ? A : B;
16             default:ns = A;
17         endcase
18     end
19
20     always @(posedge clk,posedge rst)begin
21         if(rst)
22             ps <= A;
23         else if(detect)
24             ps <= ns;
25     end
26
27     assign serout = (ps == G)? ~serin : 1'b0;
28     assign state = ps;
29
30 endmodule
```

```
1  `timescale 1ns/1ns
2  module seq_detectorTB();
3      reg serialin = 0, clk = 0, rst = 0, detect = 1;
4      wire serialout, serialoutQ;
5      wire [2:0] state1, state2;
6      seq_detector UUT(serialin, detect, clk, rst, serialout, state1);
7      seq_detectorQ UUT1(serialin, detect, clk, rst, serialoutQ, state2);
8      initial forever #100 clk = ~clk;
9      initial begin
10                 #50
11                 #200 serialin = 0;
12                 #200 serialin = 1;
13                 #200 serialin = 1;
14                 #200 serialin = 1;
15                 #200 serialin = 1;
16                 #200 serialin = 1;
17                 #200 serialin = 0;
18                 #200 serialin = 1;
19                 #200 detect = 0;
20                 #200 serialin = 0;
21                 #200 serialin = 0;
22                 #200 serialin = 1;
23                 #200 serialin = 1;
24                 #200 serialin = 1;
25                 #200 serialin = 1;
26                 #200 serialin = 1;
27                 #200 serialin = 0;
28                 #200 $stop;
29     end
```





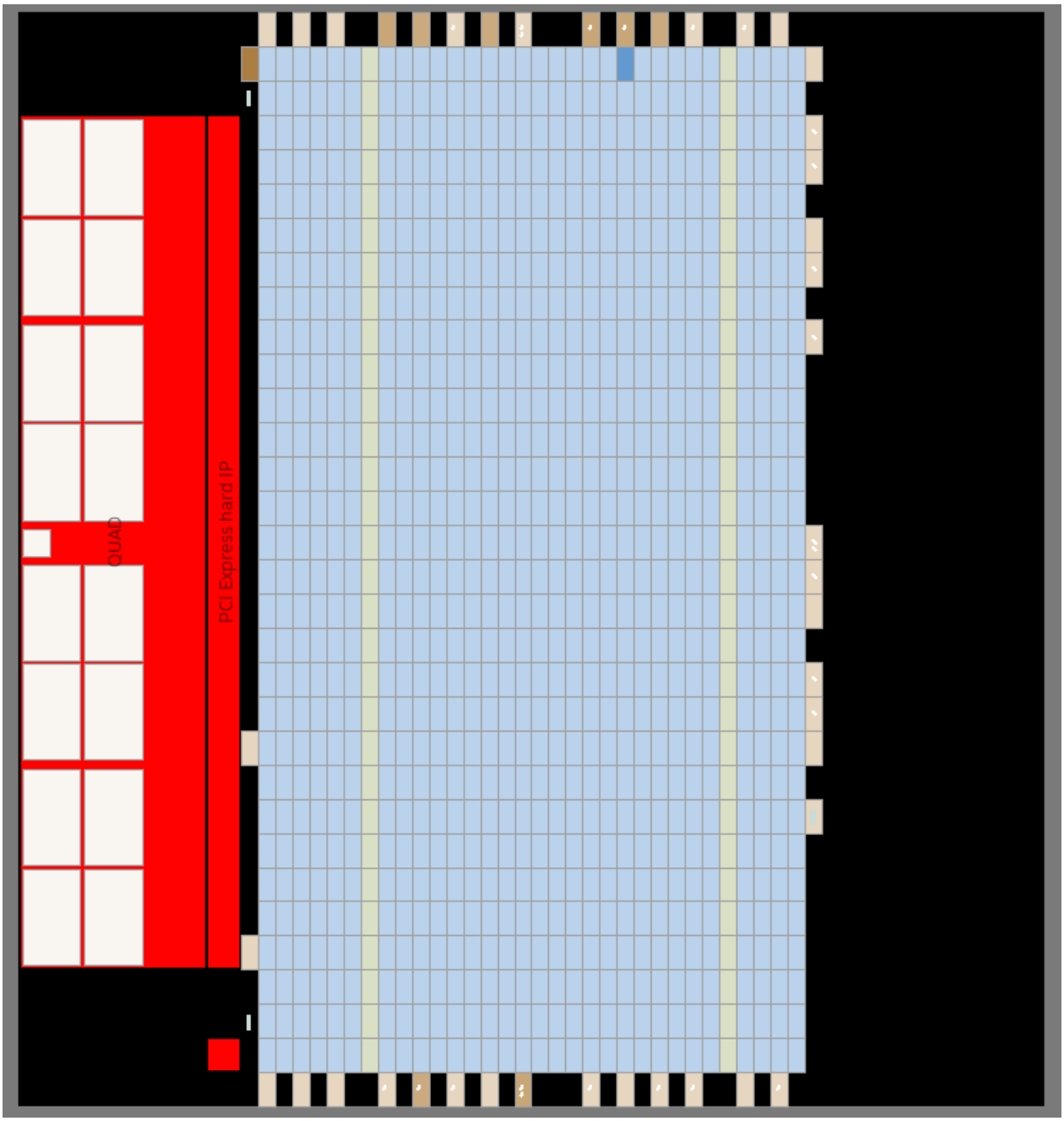
ramete	Value...	Type...
A	000	Unsigne
B	001	Unsigne
C	010	Unsigne
D	011	Unsigne

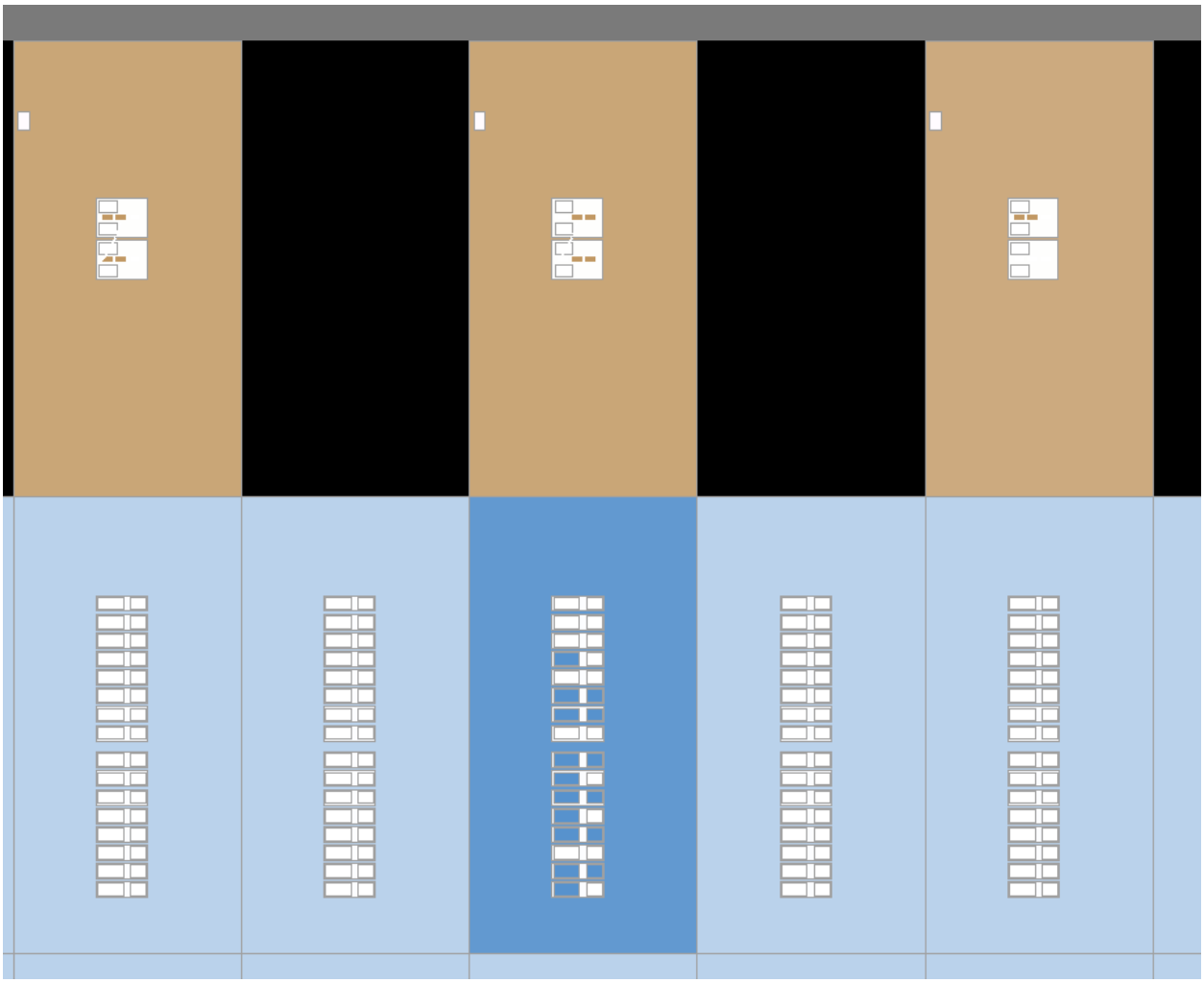


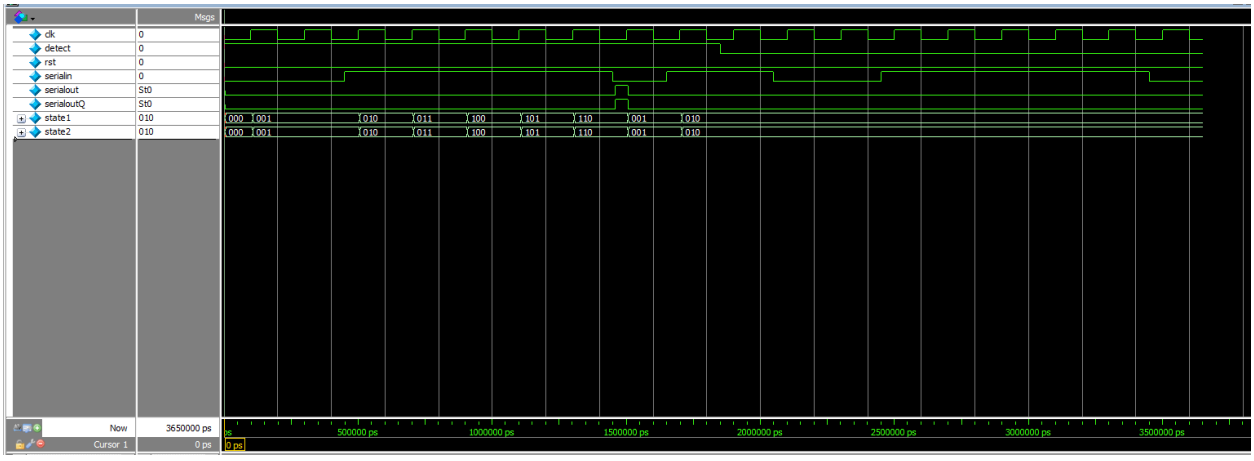
Flow Summary

<<Filter>>

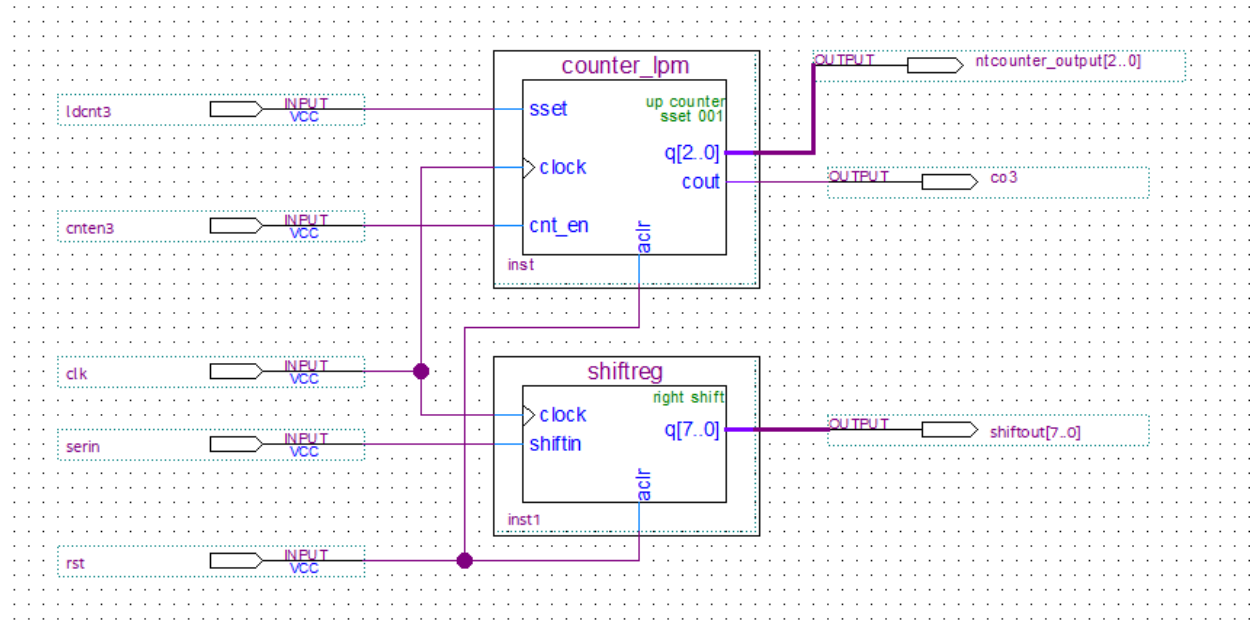
Flow Status	Successful - Tue Jan 02 22:29:25 2024
Quartus Prime Version	20.1.0 Build 711 06/05/2020 SJ Lite Edition
Revision Name	seq_detector
Top-level Entity Name	seq_detector
Family	Cyclone IV GX
Device	EP4CGX15BF14A7
Timing Models	Final
Total logic elements	10 / 14,400 (< 1 %)
Total registers	6
Total pins	8 / 81 (10 %)
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 %)


QUAD
PCI Express hard IP




Msgs
clk
detect
rst
serialin
serialout
serialoutQ
state1
state2
Now
Cursor 1
3650000 ps
0 ps
500000 ps
1000000 ps
1500000 ps
2000000 ps
2500000 ps
3000000 ps
3500000 ps

Part b:


counter_lpm
ldcnt3
INPUT
VCC
sset
up counter
sset 001
clock
q[2..0]
cout
cnten3
cnt_en
aclr
inst
OUTPUT
ntcounter_output[2..0]
co3
clk
serin
rst
shiftreg
right shift
shiftin
q[7..0]
inst1
shiftout[7..0]

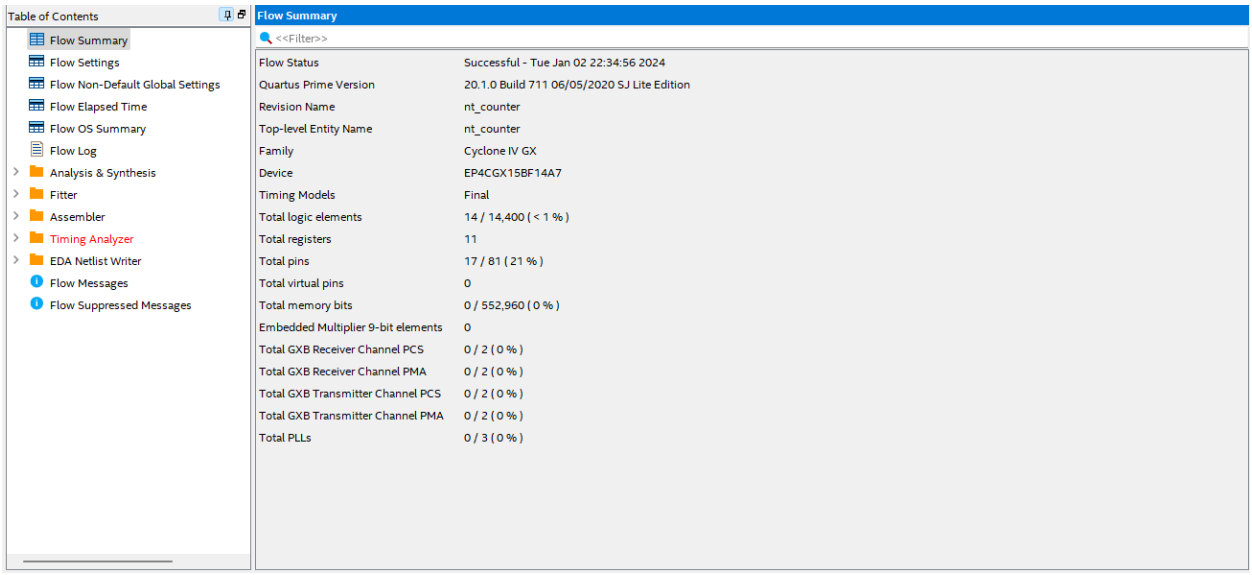
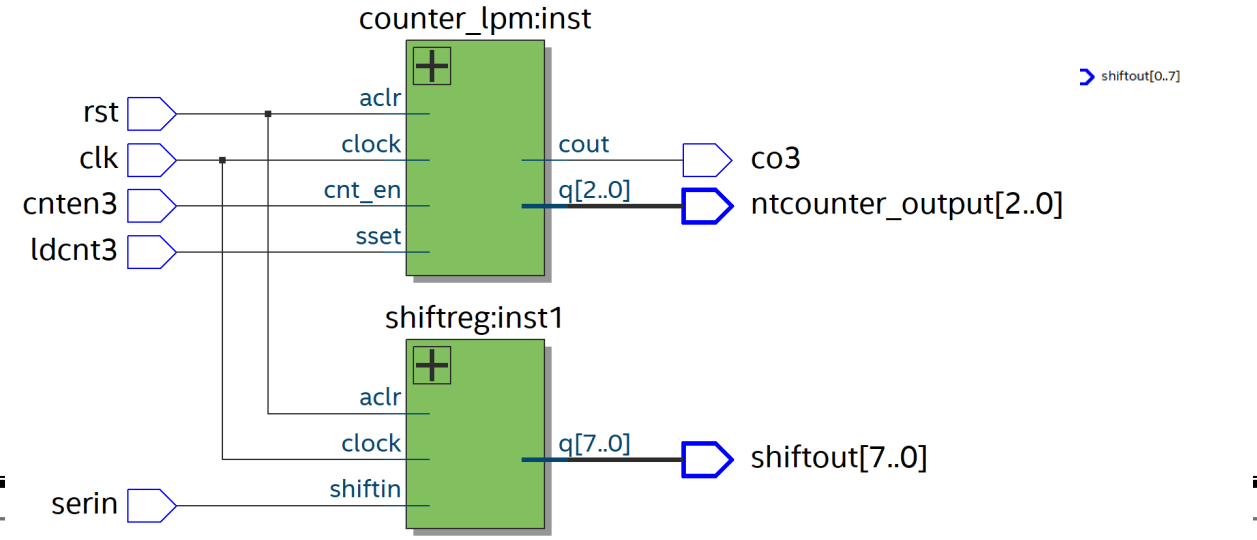
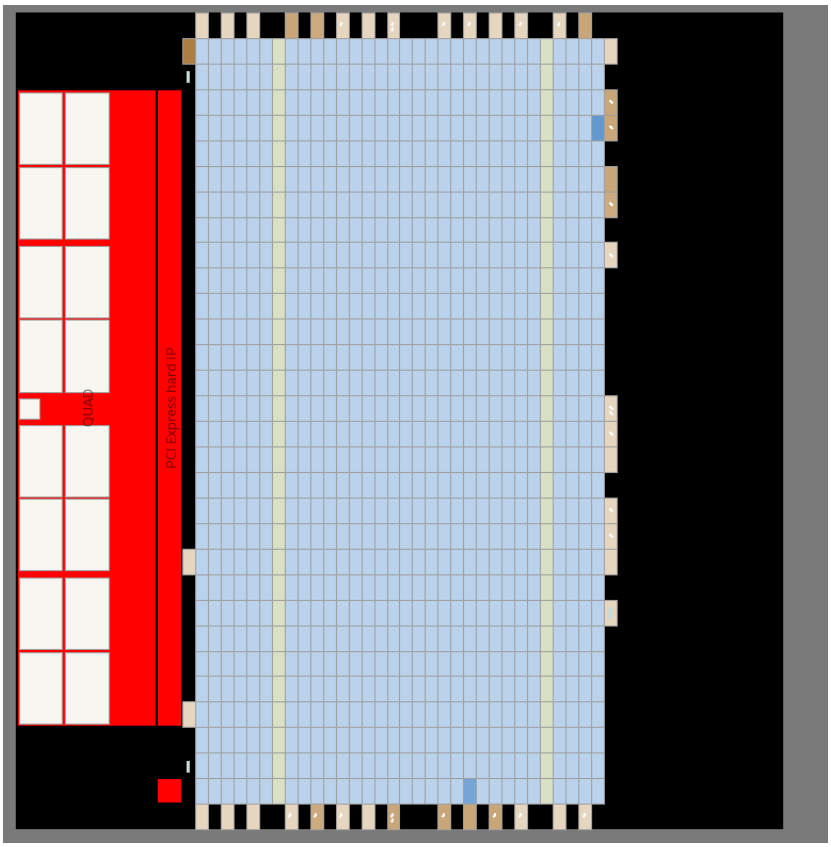
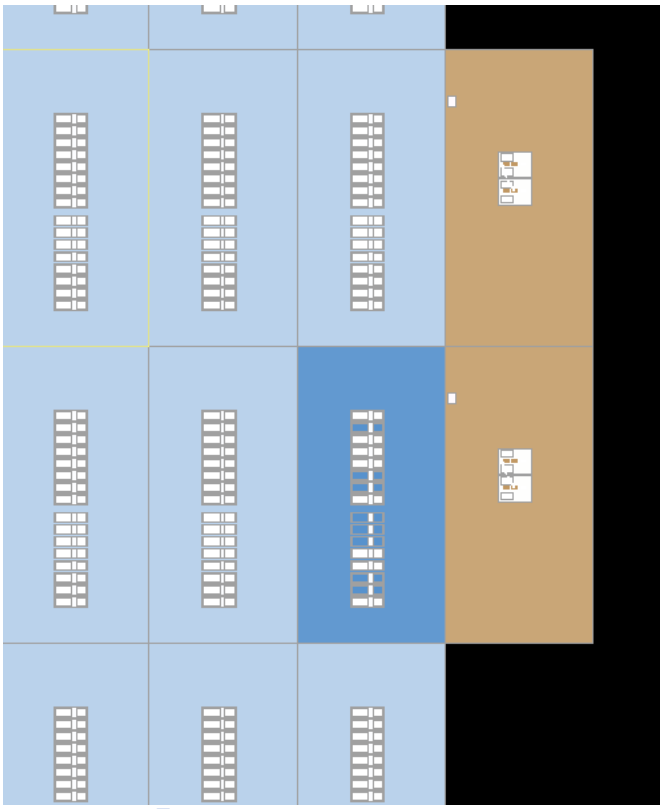
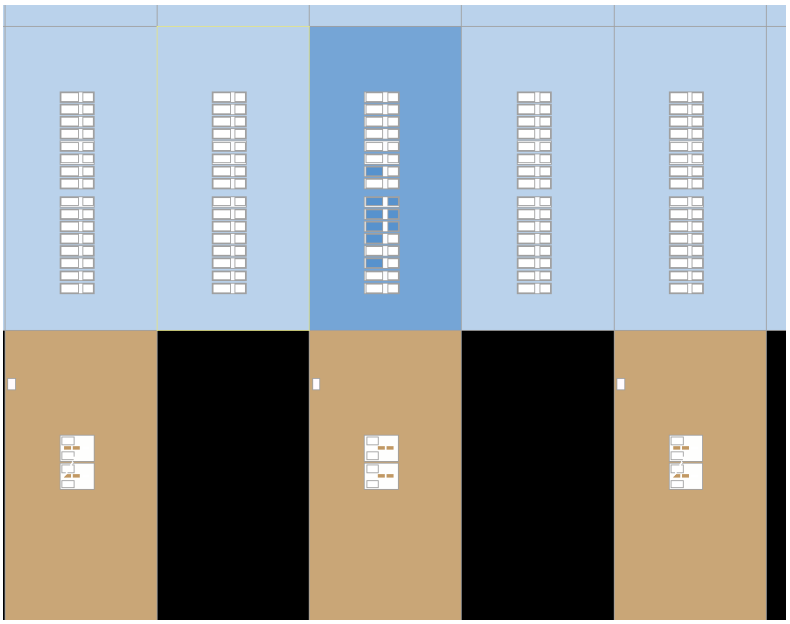


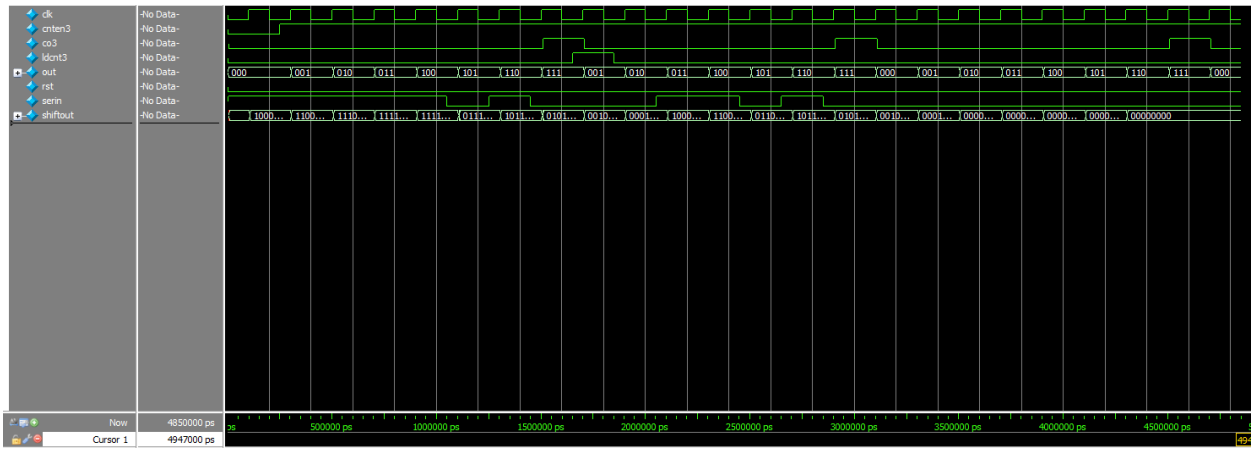
Table of Contents
Flow Summary
Flow Settings
Flow Non-Default Global Settings
Flow Elapsed Time
Flow OS Summary
Flow Log
Analysis & Synthesis
Fitter
Assembler
Timing Analyzer
EDA Netlist Writer
Flow Messages
Flow Suppressed Messages
Flow Summary
<<Filter>>
Flow Status Successful - Tue Jan 02 22:34:56 2024
Quartus Prime Version 20.1.0 Build 711 06/05/2020 SJ Lite Edition
Revision Name nt_counter
Top-level Entity Name nt_counter
Family Cyclone IV GX
Device EP4CGX15BF14A7
Timing Models Final
Total logic elements 14 / 14,400 (< 1 %)
Total registers 11
Total pins 17 / 81 (21 %)
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 %)

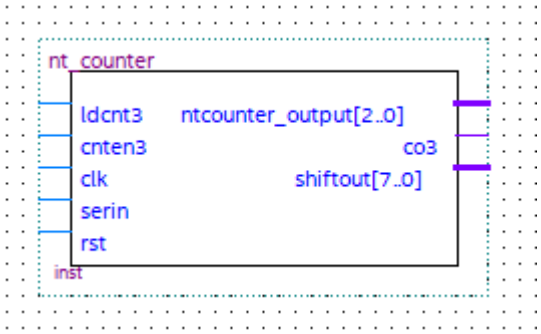

counter_lpm:inst
rst
clk
cnten3
ldcnt3
aclr
clock
cnt_en
sset
cout
q[2..0]
co3
ntcounter_output[2..0]
shiftout[0..7]
shiftreg:inst1
aclr
clock
shiftin
serin
q[7..0]
shiftout[7..0]


PCI Express hard IP
QUAD

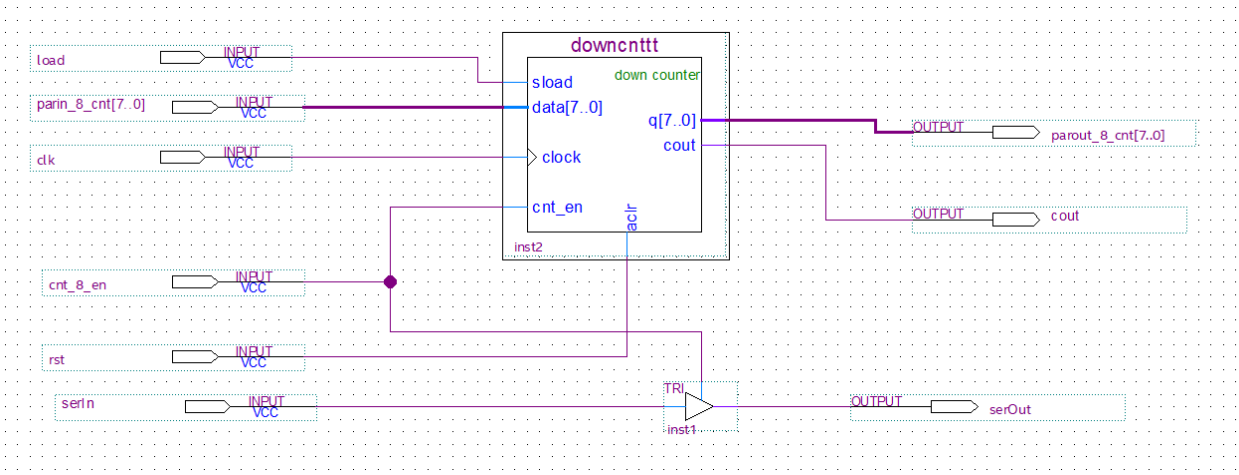





clk
cnten3
co3
ldcnt3
out
rst
serin
shiftout
-No Data-
Now
4850000 ps
Cursor 1
4947000 ps
500000 ps
1000000 ps
1500000 ps
2000000 ps
2500000 ps
3000000 ps
3500000 ps
4000000 ps
4500000 ps


nt_counter
ldcnt3
cnten3
clk
serin
rst
ntcounter_output[2..0]
co3
shiftout[7..0]
inst

Part c:



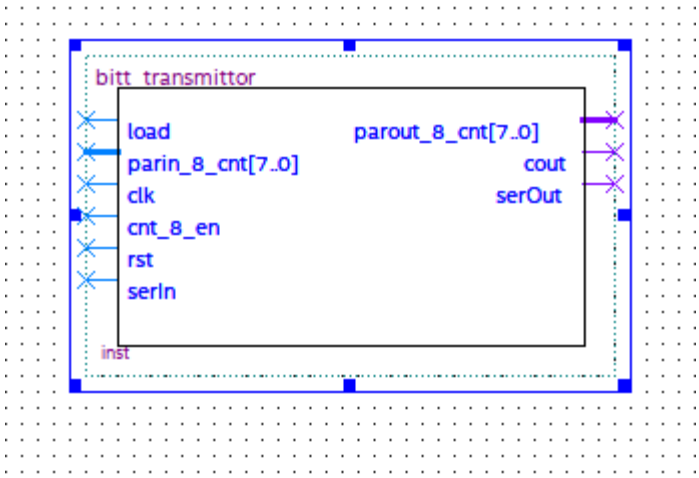


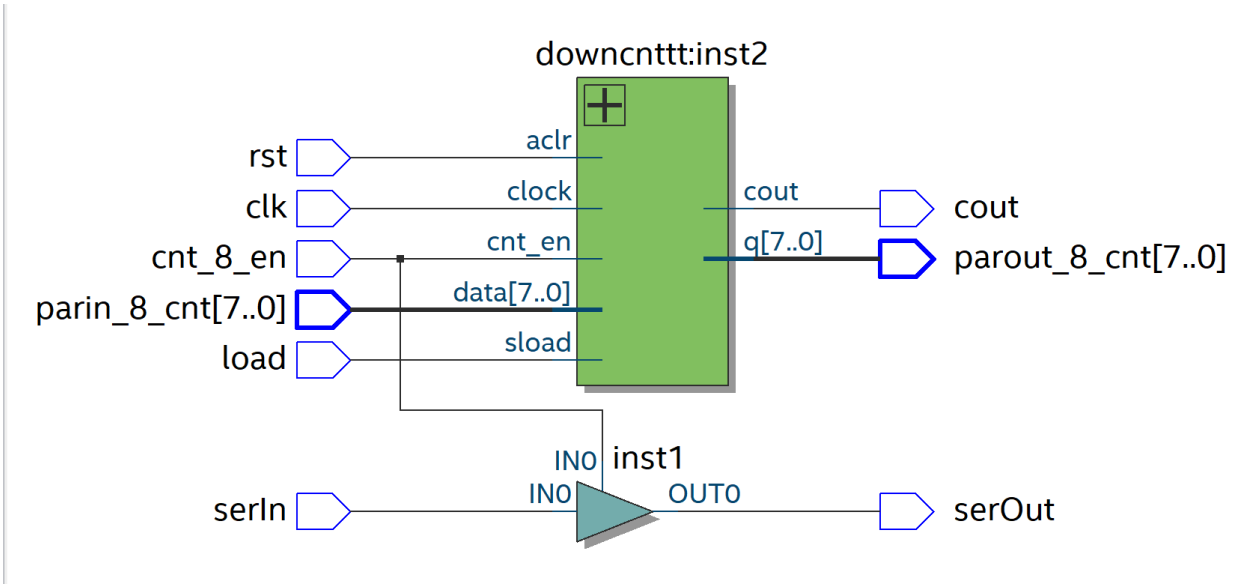
Table of Contents

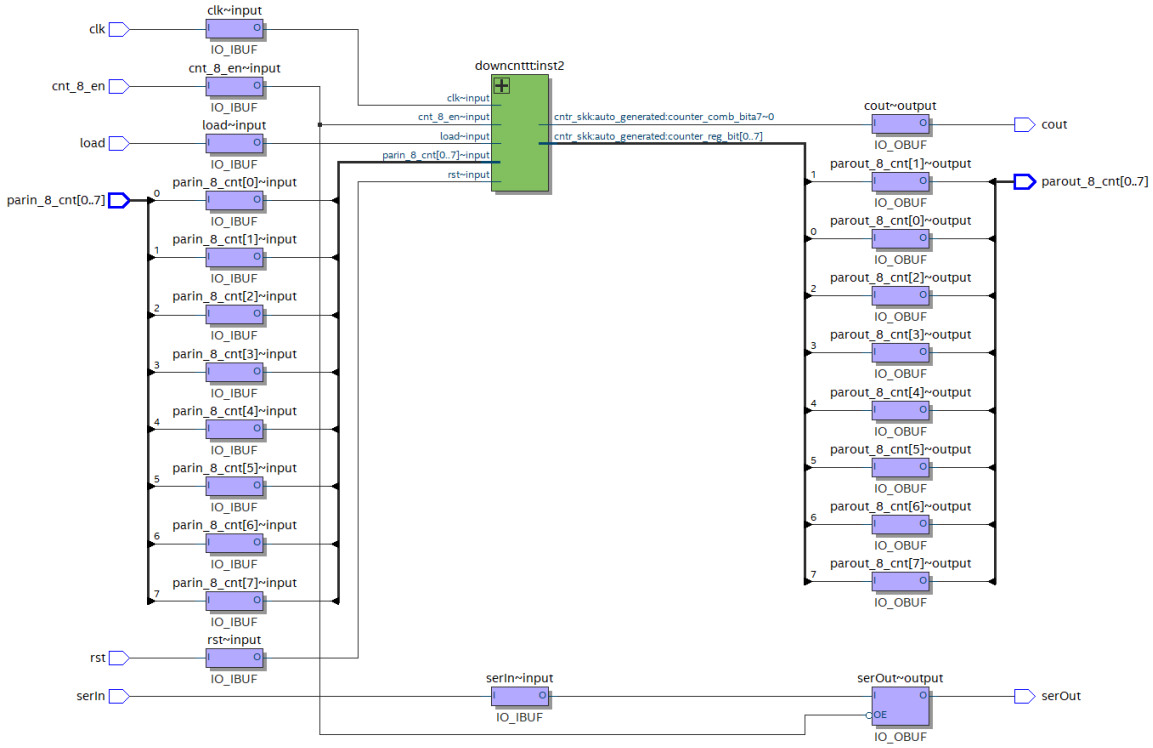
- Flow Summary
- Flow Settings
- Flow Non-Default Global Settings
- Flow Elapsed Time
- Flow OS Summary
- Flow Log
- Analysis & Synthesis
- Fitter
- Assembler
- Timing Analyzer
- EDA Netlist Writer
- Flow Messages
- Flow Suppressed Messages

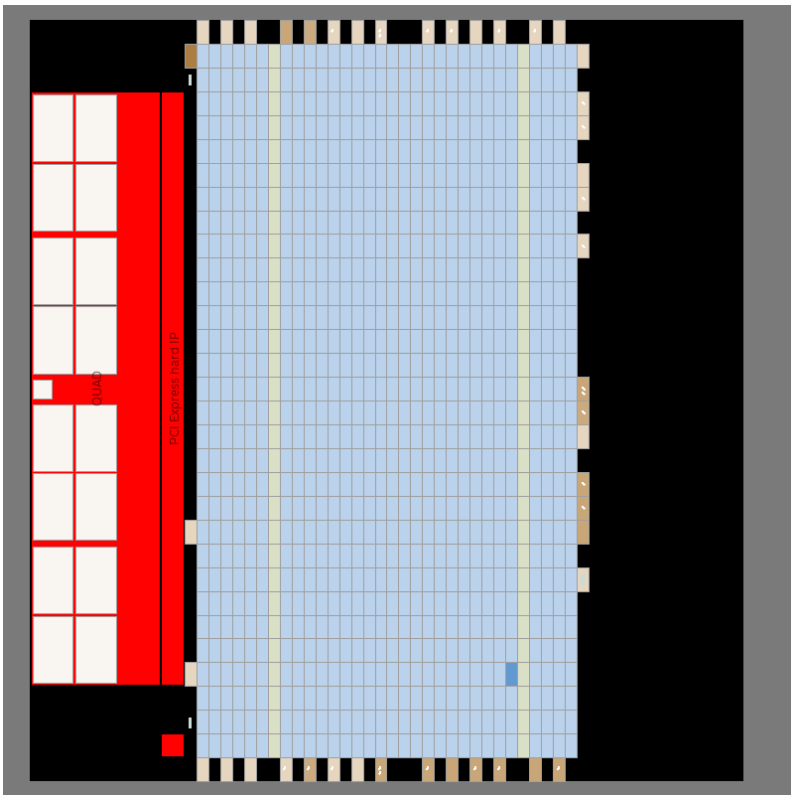
Flow Summary

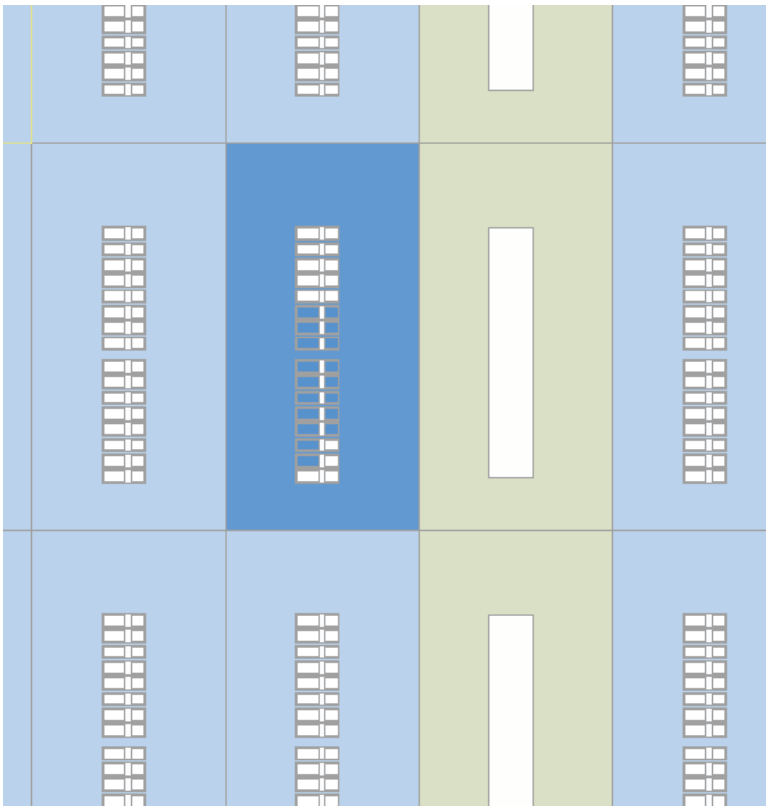
<<Filter>>

Flow Status	Successful - Tue Jan 02 22:37:24 2024
Quartus Prime Version	20.1.0 Build 711 06/05/2020 SJ Lite Edition
Revision Name	bitt_transmittor
Top-level Entity Name	bitt_transmittor
Family	Cyclone IV GX
Device	EP4CGX15BF14A7
Timing Models	Final
Total logic elements	10 / 14,400 (< 1 %)
Total registers	8
Total pins	23 / 81 (28 %)
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 %)

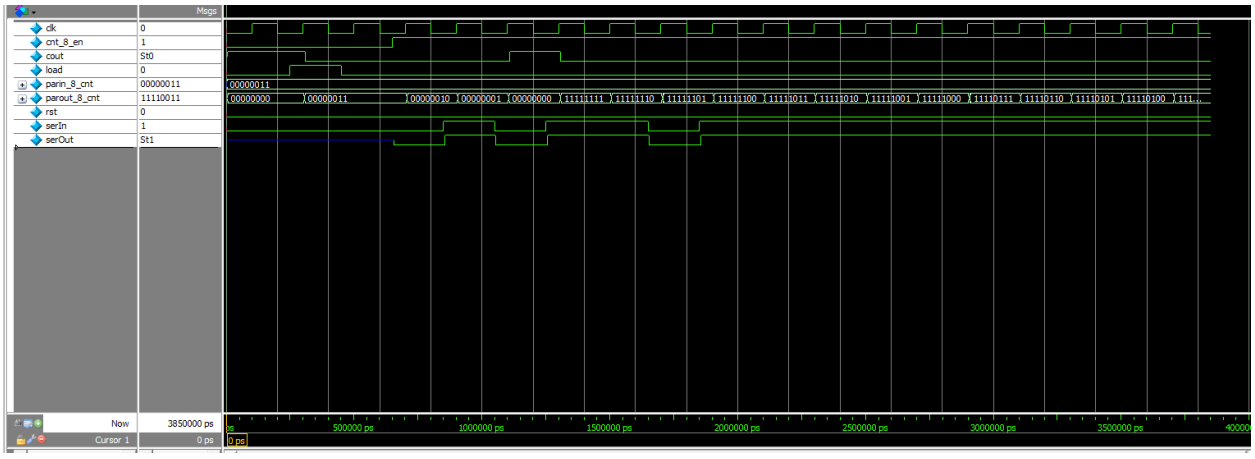

downcnttt:inst2
rst
clk
cnt_8_en
parin_8_cnt[7..0]
load
aclr
clock
cnt_en
data[7..0]
sload
cout
q[7..0]
cout
parout_8_cnt[7..0]
IN0
inst1
IN0
OUT0
serIn
serOut


clk
clk~input
IO_IBUF
cnt_8_en
cnt_8_en~input
IO_IBUF
load
load~input
IO_IBUF
parin_8_cnt[0..7]
parin_8_cnt[0]~input
parin_8_cnt[1]~input
parin_8_cnt[2]~input
parin_8_cnt[3]~input
parin_8_cnt[4]~input
parin_8_cnt[5]~input
parin_8_cnt[6]~input
parin_8_cnt[7]~input
IO_IBUF
rst
rst~input
IO_IBUF
serIn
serIn~input
IO_IBUF
downcnttt:inst2
clk~input
cnt_8_en~input
load~input
parin_8_cnt[0..7]~input
rst~input
cntr_skk:auto_generated:counter_comb_bita7~0
cntr_skk:auto_generated:counter_reg_bit[0..7]
cout~output
IO_OBUF
cout
parout_8_cnt[1]~output
parout_8_cnt[0..7]
parout_8_cnt[0]~output
parout_8_cnt[2]~output
parout_8_cnt[3]~output
parout_8_cnt[4]~output
parout_8_cnt[5]~output
parout_8_cnt[6]~output
parout_8_cnt[7]~output
IO_OBUF
serOut~output
OE
serOut

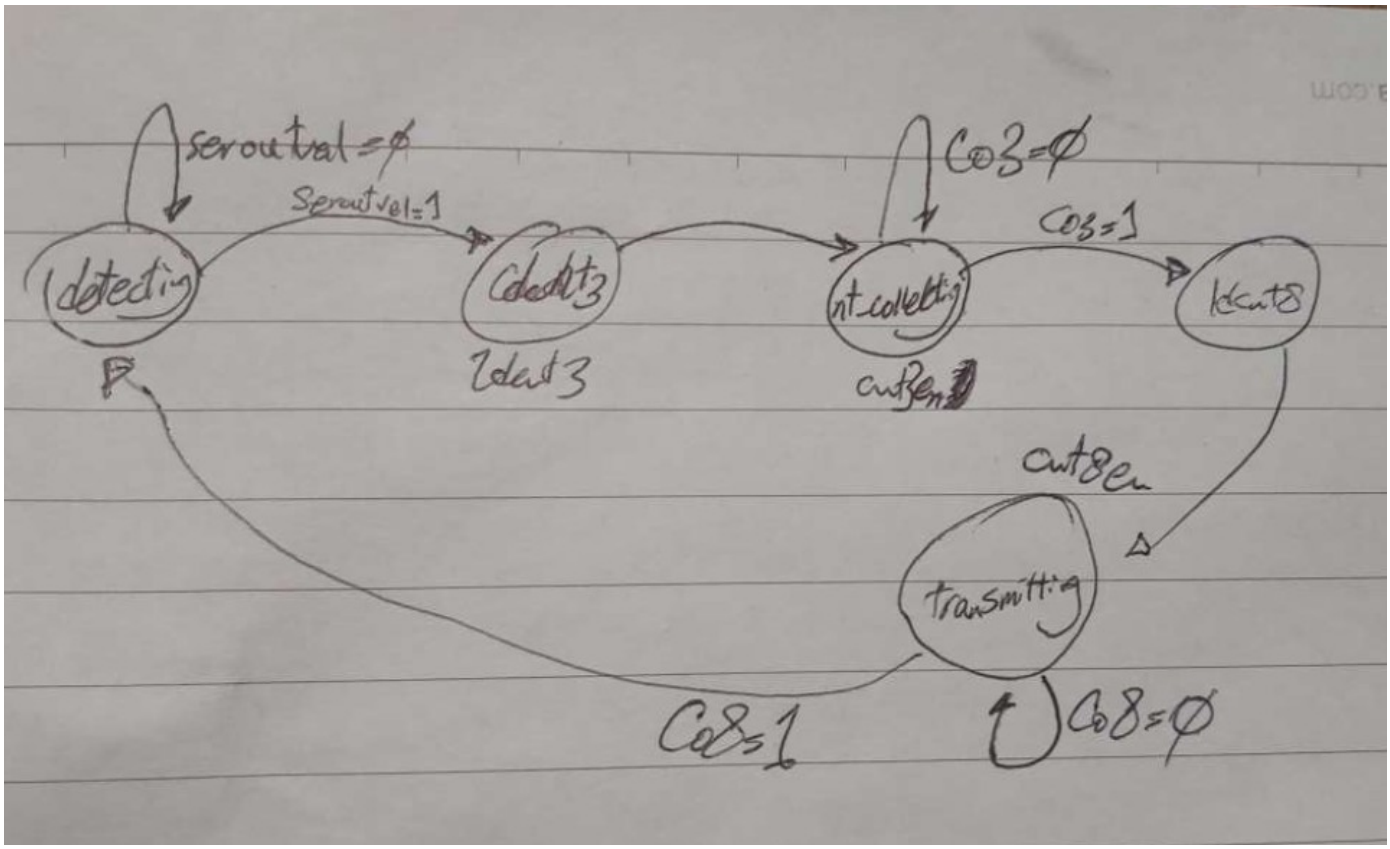

PCI Express hard IP
QUAD



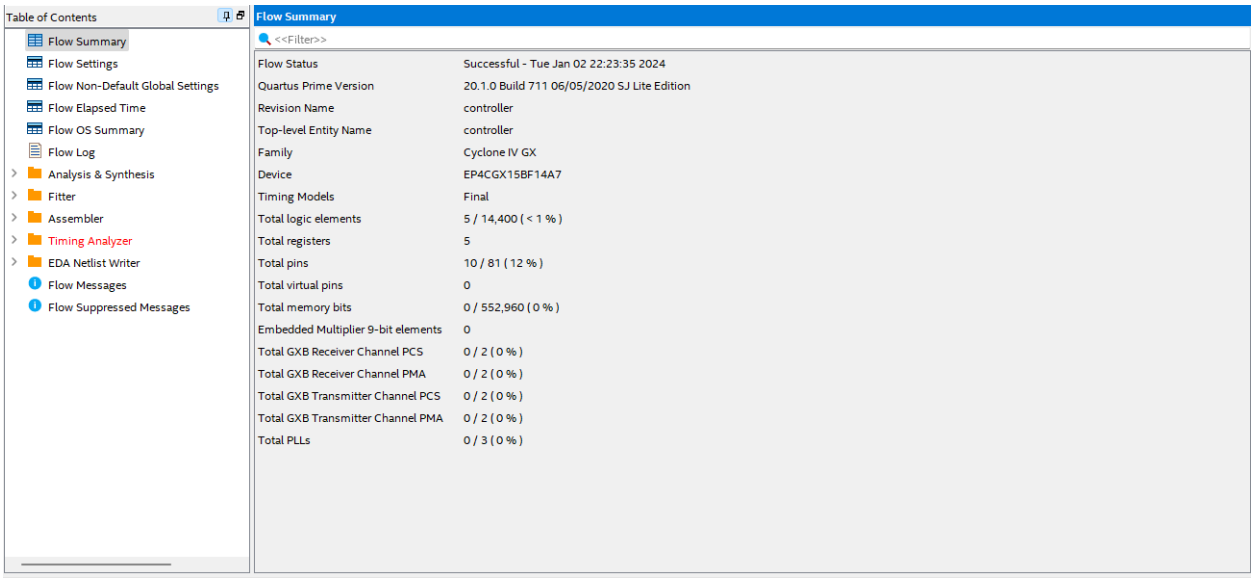
```
1  `timescale 1ns/1ns
2  module transmitorTB();
3      reg cnt_8_en = 0, clk = 0, rst = 0, load = 0, serIn = 0;
4      reg [7:0]parin_8_cnt = 8'b00000011;
5      wire cout, serOut;
6      wire [7:0]parout_8_cnt;
7      bitt_transmittor UUT (
8          serOut,
9          serIn,
10         cnt_8_en,
11         cout,
12         load,
13         clk,
14         rst,
15         parin_8_cnt,
16         parout_8_cnt);
17     initial forever #100 clk = ~clk;
18     initial begin
19             #50
20             #200 load = 1;
21             #200 load = 0;
22             #200 cnt_8_en = 1;
23             #200 serIn = 1;
24             #200 serIn = 0;
25             #200 serIn = 1;
26             #200 serIn = 1;
27             #200 serIn = 0;
28             #200 serIn = 1;
29             #2000 $stop;
30     end
31 endmodule
32
33
```



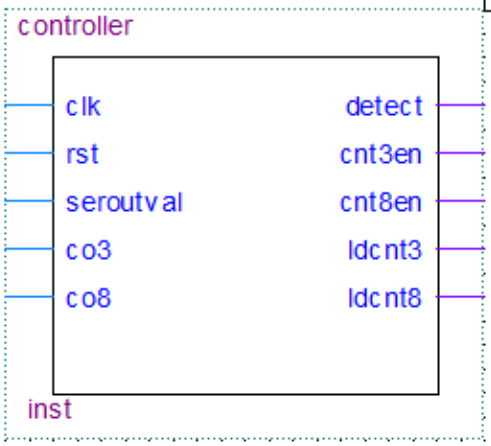
Part d:

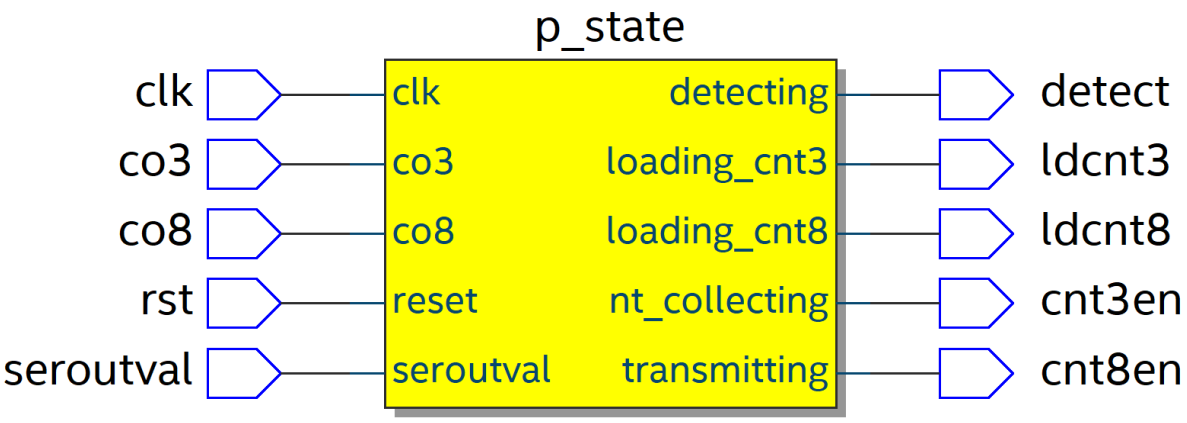


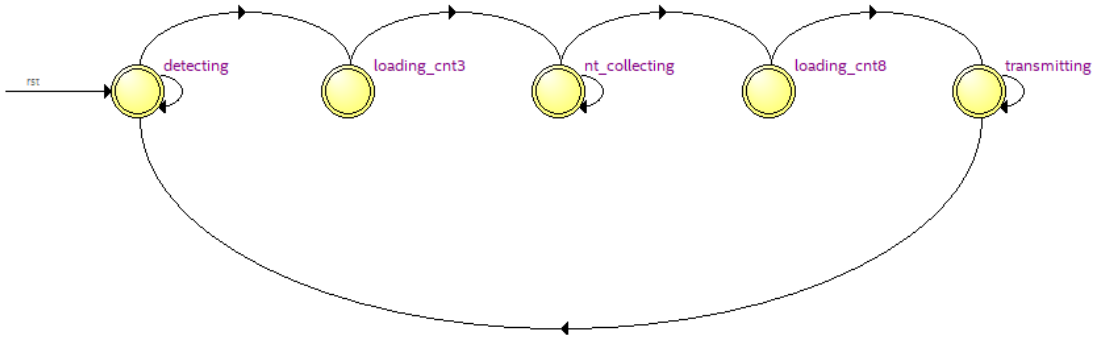

```
1   `timescale 1ns/1ns
2   module controller(input clk,rst,seroutval,co3,co8,output reg detect,cnt3en,cnt8en,ldcnt3,ldcnt8);
3       parameter [2:0] detecting = 0, loading_cnt3 = 1, nt_collecting = 2, loading_cnt8 = 3, transmitting = 4;
4       reg [2:0] p_state,n_state;
5       always @(p_state, seroutval, co3, co8) begin
6           n_state = 0;
7           {detect, cnt3en, cnt8en, ldcnt3, ldcnt8} = 6'b0;
8           case(p_state)
9               detecting:begin
10                  detect = 1; n_state = seroutval ? loading_cnt3 : detecting;
11              end
12              loading_cnt3:begin
13                  ldcnt3 = 1; n_state = nt_collecting;
14              end
15              nt_collecting:begin
16                  cnt3en = 1; n_state = co3 ? loading_cnt8 : nt_collecting;
17              end
18              loading_cnt8:begin
19                  ldcnt8 = 1; n_state = transmitting;
20              end
21              transmitting:begin
22                  cnt8en = 1; n_state = co8 ? detecting : transmitting;
23              end
24              default: n_state = detecting;
25          endcase
26      end
27      always@(posedge clk, posedge rst) begin
28          if(rst)
29              p_state <= detecting;
30          else
31              p_state <= n_state;
32      end
33  endmodule
```

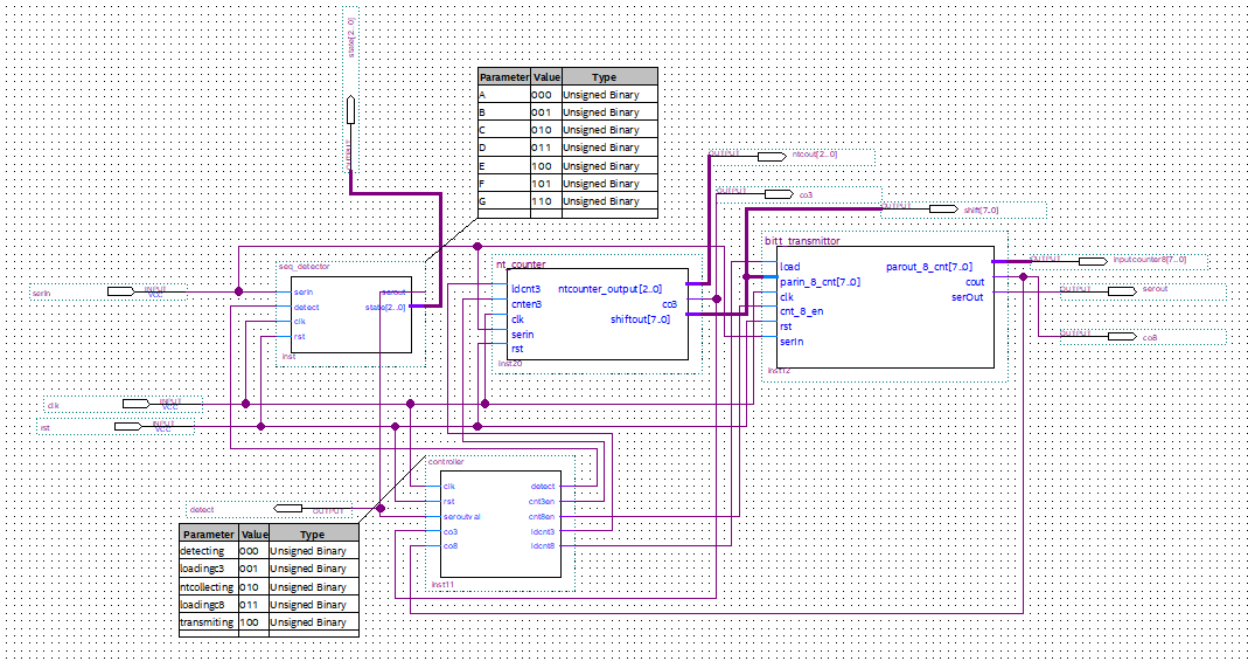


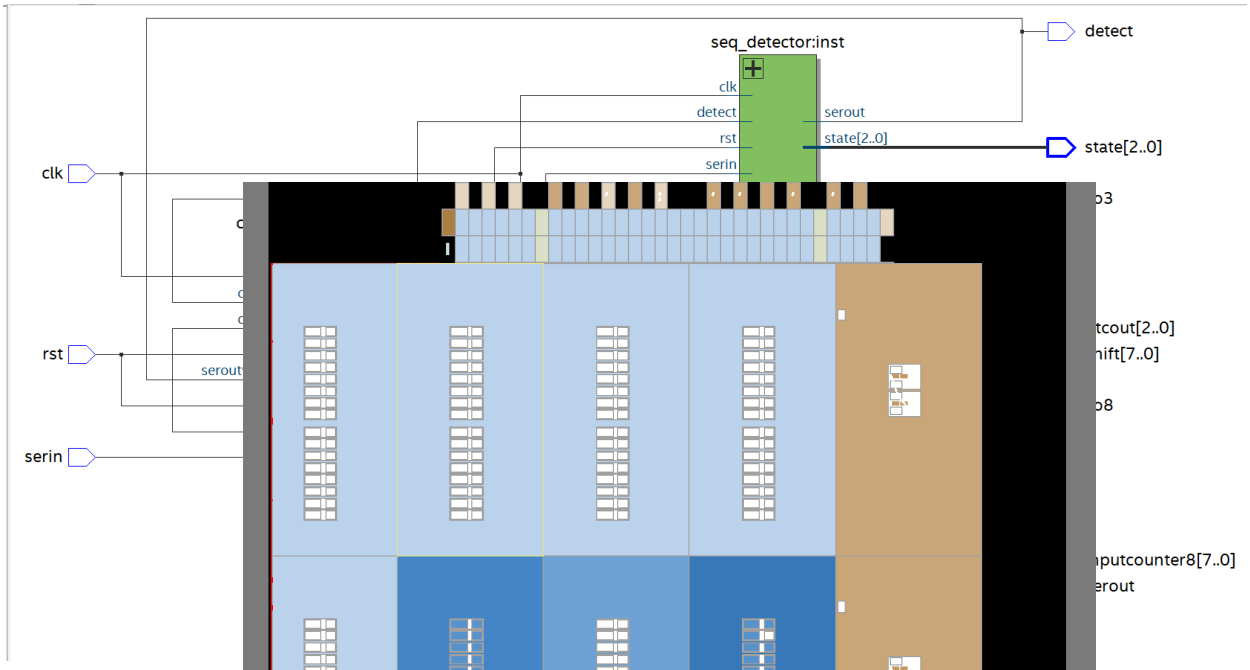
Parameter	Value	Type
detecting	000	Unsigned Binary
loadingc3	001	Unsigned Binary
ntcollecting	010	Unsigned Binary
loadingc8	011	Unsigned Binary
transmiting	100	Unsigned Binary



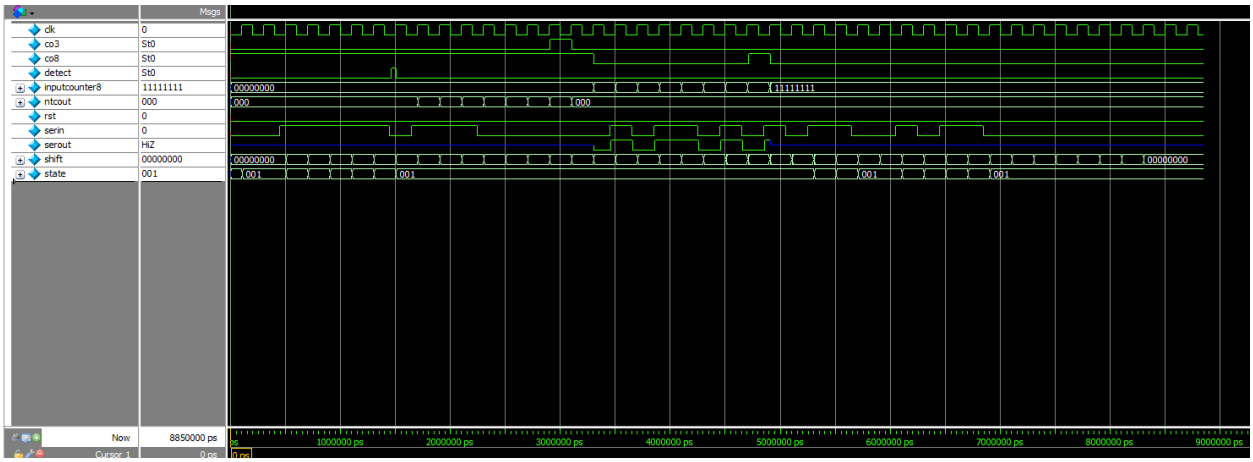

p_state
clk
co3
co8
rst
seroutval
clk
co3
co8
reset
seroutval
detecting
loading_cnt3
loading_cnt8
nt_collecting
transmitting
detect
ldcnt3
ldcnt8
cnt3en
cnt8en


rst
detecting
loading_cnt3
nt_collecting
loading_cnt8
transmitting


Parameter Value Type
A 000 Unsigned Binary
B 001 Unsigned Binary
C 010 Unsigned Binary
D 011 Unsigned Binary
E 100 Unsigned Binary
F 101 Unsigned Binary
G 110 Unsigned Binary
seq_detector
nt_counter
bit1_transmittor
controller
Parameter Value Type
detecting 000 Unsigned Binary
loadingc3 001 Unsigned Binary
ntcollecting 010 Unsigned Binary
loadingc8 011 Unsigned Binary
transmiting 100 Unsigned Binary


seq_detector:inst
clk
detect
rst
serin
serout
state[2..0]
rst
serin

```
1  `timescale 1ns/1ns
2  module finalsystTB();
3      reg serin = 0, clk = 0, rst = 0;
4      wire serout, co3, co8, detect;
5      wire [2:0] state, ntcout;
6      wire [7:0] shift, inputcounter8;
7      finalsyst UUT(
8          serout,
9          clk,
10         rst,
11         serin,
12         co3,
13         detect,
14         co8,
15         inputcounter8,
16         ntcout,
17         shift,
18         state);
19     initial forever #100 clk = ~clk;
20     initial begin
21             #50
22             #200 serin = 0;
23             #200 serin = 1;
24             #200 serin = 1;
25             #200 serin = 1;
26             #200 serin = 1;
27             #200 serin = 0;
28             #200 serin = 1;
29             #200 serin = 1;
30             #200 serin = 1;
31             #200 serin = 0;
32             #200 serin = 0;
33             #200 serin = 0;
34             #200 serin = 0;
35             #200 serin = 0;
36             #200 serin = 0;
37             #200 serin = 1;
38             #200 serin = 0;
39             #200 serin = 0;
40             #200 serin = 1;
41             #200 serin = 1;
42             #200 serin = 0;
43             #200 serin = 1;
44             #200 serin = 0;
45             #200 serin = 1;
46             #200 serin = 0;
47             #200 serin = 1;
48             #200 serin = 1;
49             #200 serin = 0;
50             #200 serin = 0;
51             #200 serin = 1;
52             #200 serin = 0;
53             #200 serin = 1;
54             #200 serin = 1;
55             #200 serin = 0;
56             #2000 $stop;
57     end
58 endmodule
59
60
```


Msgs
clk
co3
co8
detect
inputcounter8
ntcout
rst
serin
serout
shift
state
0
St0
St0
St0
11111111
000
0
0
HiZ
00000000
001
00000000
11111111
000
001
001
001
001
00000000
Now
8850000 ps
Cursor 1
0 ps
1000000 ps
2000000 ps
3000000 ps
4000000 ps
5000000 ps
6000000 ps
7000000 ps
8000000 ps
9000000 ps

The End😊