

بسم الله الرحمن الرحيم

پروژه ۵ درس مدار منطقی  
دکتر نوابی

مهدی وجهی

۸۱۰۱۰۱۵۵۸

# فهرست

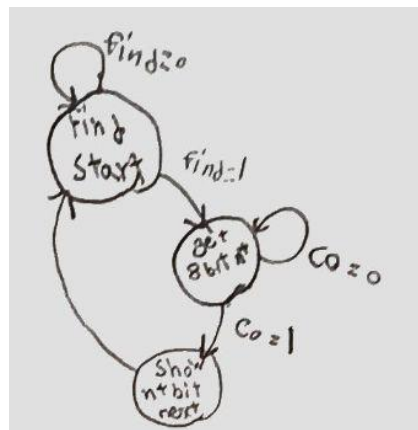
بخش a.....	3
شرح اجمالی.....	3
پیاده سازی مدار شناسایی ۰۱۱۱۱۱۰.....	3
نوشتن کد.....	5
زیر بخش ۱.....	6
زیر بخش ۲ و ۳.....	7
synthesizing.....	7
بررسی قطعات مدار و پیاده سازی آن.....	9
بررسی مدار از لحاظ زمانی.....	12
ساخت نماد در کوارتز.....	13
بخش b.....	15
نوشتن کد.....	15
synthesizing.....	16
بررسی قطعات مدار و پیاده سازی آن.....	17
زمان بندی.....	20
ساخت نماد در کوارتز.....	20
بخش c.....	21
نوشتن کد.....	21
synthesizing.....	22
بررسی قطعات مدار و پیاده سازی آن.....	23
زمانبندی.....	27
ساخت نماد در کوارتز.....	27
بخش d.....	28
نوشتن کد.....	28
طراحی مدار.....	30
synthesizing.....	31
بررسی قطعات مدار و پیاده سازی آن.....	31
زمان بندی.....	34

## بخش a

### شرح اجمالی

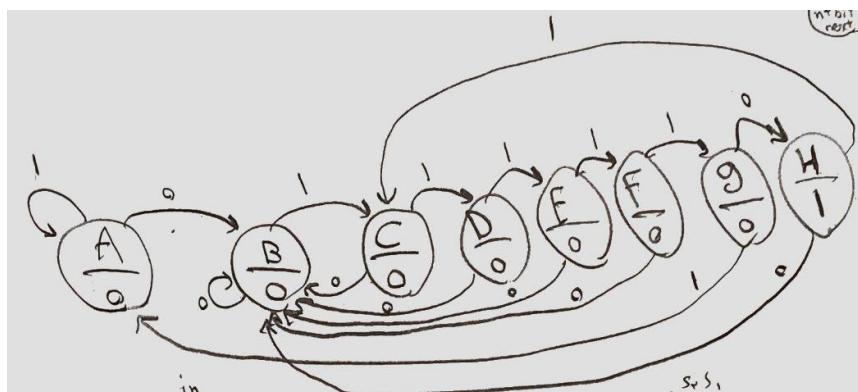
مدار این پروژه ۳ مرحله دارد:

1. پیدا کردن یک توالی (۰۱۱۱۱۱۰) در ورودی
  2. گرفتن یک ورودی ۸ بیت به صورت سریالی و ذخیره آن
  3. نمایش اعداد ورودی بر روی خروجی به تعداد عدد ۸ بیتی مرحله قبل
- که نمودار آن به طور خلاصه به شکل زیر است:

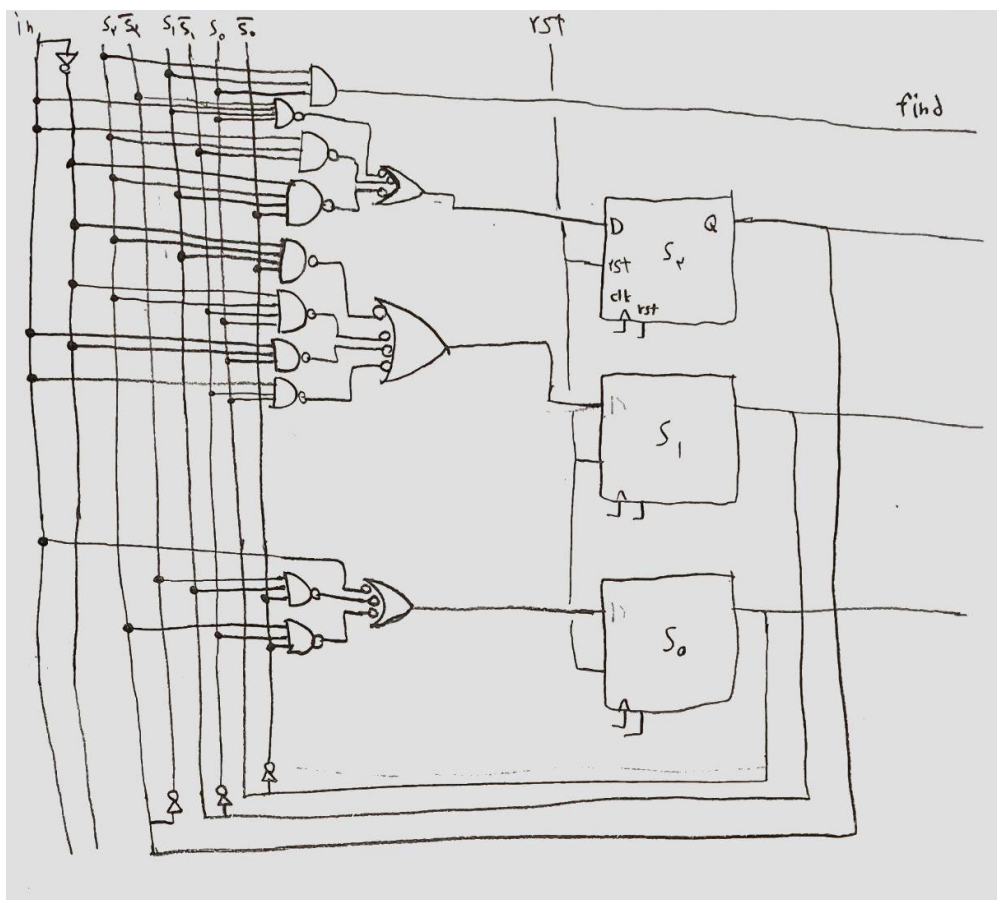
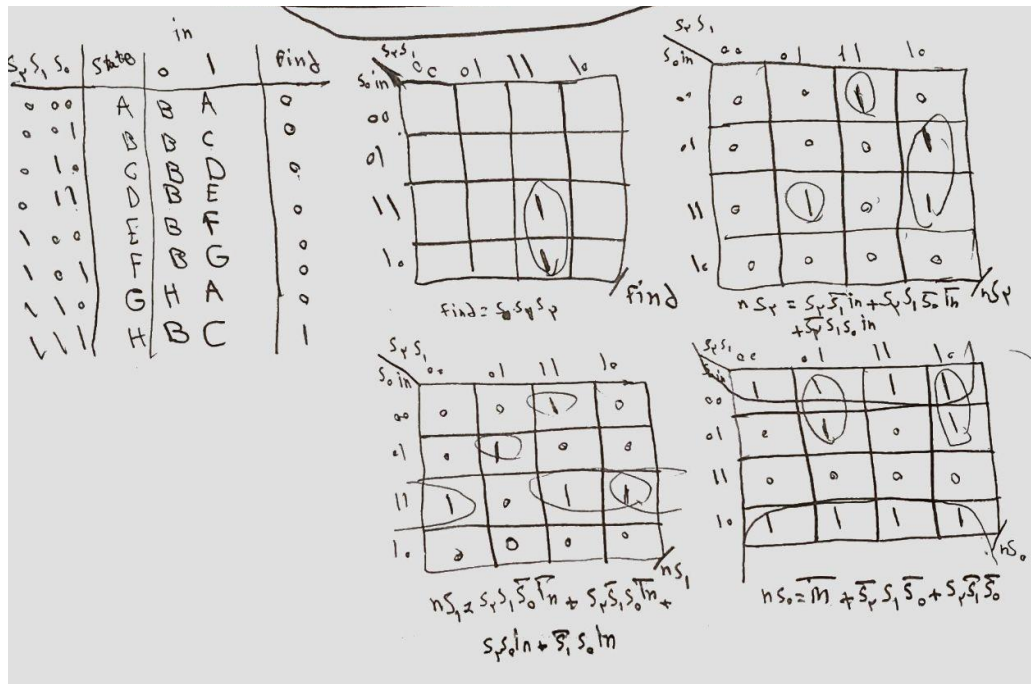


### پیاده سازی مدار شناسایی ۰۱۱۱۱۱۰

حال در این بخش باید توالی (۰۱۱۱۱۱۰) را شناسایی کنیم برای این کار یک moore machine ساده طراحی می کنیم که نمودار آن به شکل زیر است:



حال مدار آن را به صورت زیر پیاده می کنیم:



## نوشتن کد

حال که مدار طراحی شد کد آن می نویسیم که شامل ۳ بخش است:

- مدار بدون حافظه
- مدار حافظه
- مدار خروجی

```
module qa(input in, rst, clk, output find);
    logic [2:0] ns,ps;
    parameter [2:0] A = 3'b000,
                  B = 3'b001,
                  C = 3'b010,
                  D = 3'b011,
                  E = 3'b100,
                  F = 3'b101,
                  G = 3'b110,
                  H = 3'b111;

    // combinational part
    always @(ps,in) begin
        ns = A;
        case (ps)
            A: ns = in ? A : B;
            B: ns = in ? C : B;
            C: ns = in ? D : B;
            D: ns = in ? E : B;
            E: ns = in ? F : B;
            F: ns = in ? G : B;
            G: ns = in ? A : H;
            H: ns = in ? C : B;
            default: ns = A;
        endcase
    end

    // output
    assign find = (ps == H) ? 1'b1 : 1'b0;

    // sequential part
    always @(posedge clk, posedge rst) begin
        if (rst)
            ps <= A;
        else
            ps <= ns;
        end
    end
endmodule
```

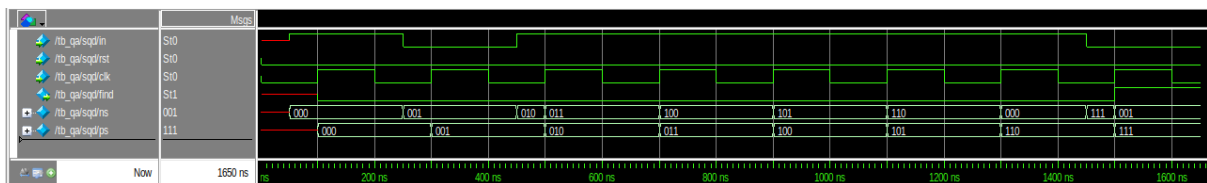
## زیر بخش ۱

در این بخش لازم است یک میز آزمایش برای مدار خود طراحی کنیم. این قسمت پیچیدگی خاصی ندارد فقط کافیه یک ورودی غلط بدهیم سپس ورودی درست هم بدهیم و صحت آن را بسنجیم. کد آن به صورت زیر است:

```
`timescale 1ns/1ns
module tb_qa();
    logic in, clk = 0, rst = 0;
    wire out;
    qa sqd(.in(in), .clk(clk), .rst(rst), .find(out));

    always #100 clk = ~clk;
    initial begin
        #50 in = 1;
        #200 in = 0;
        #200 in = 1;
        #200 in = 1;
        #200 in = 1;
        #200 in = 1;
        #200 in = 1;
        #200 in = 0;
        #200 $stop;
    end
endmodule
```

موج خروجی مدار به شکل زیر است:



## زیر بخش ۲ و ۳

synthesizing

ابتدا باید کد را synthesis کنیم.

Flow Status	Successful - Sun Dec 31 21:27:49 2023
Quartus Prime Version	23.1std.0 Build 991 11/28/2023 SC Lite Edition
Revision Name	ca5
Top-level Entity Name	qa
Family	Cyclone IV GX
Device	EP4CGX15BF14A7
Timing Models	Final
Total logic elements	7 / 14,400 ( < 1 % )
Total registers	7
Total pins	4 / 81 ( 5 % )
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 % )

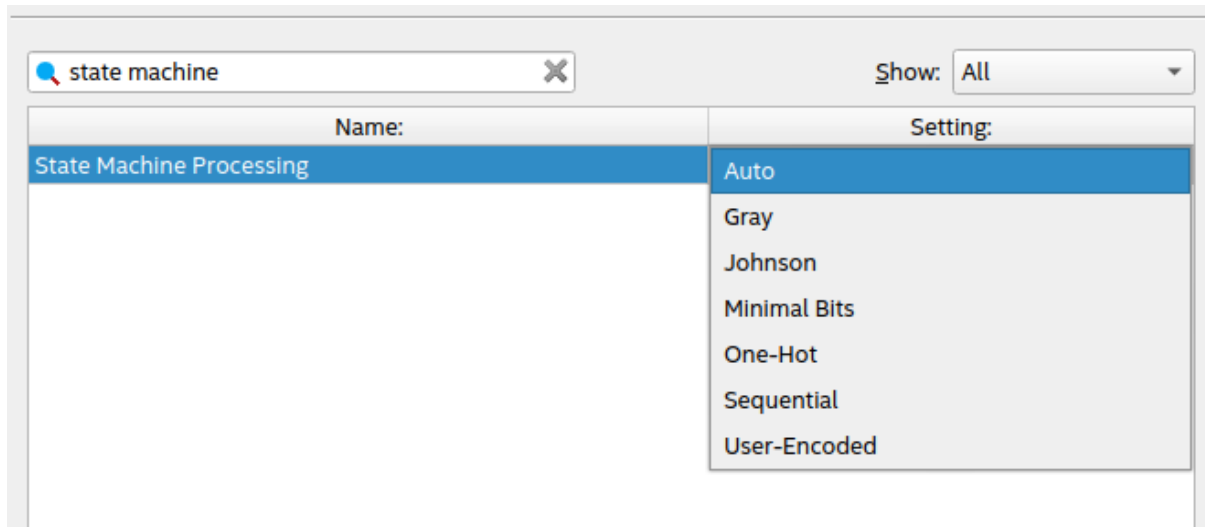
اما مشاهده می شود که ۷ رجیستر استفاده شده این یعنی از روش دیگری استفاده شده که encode آن به شکل زیر است:

	Name	H	G	F	E	D	C	B	A
1	A	0	0	0	0	0	0	0	0
2	B	0	0	0	0	0	0	1	1
3	C	0	0	0	0	0	1	0	1
4	D	0	0	0	0	1	0	0	1
5	E	0	0	0	1	0	0	0	1
6	F	0	0	1	0	0	0	0	1
7	G	0	1	0	0	0	0	0	1
8	H	1	0	0	0	0	0	0	1

لازم از از طریق

Edit settings -> compiler settings -> advanced settings (synthesis) -> state machine processing

این موضوع را اصلاح کنیم:



همانطور که مشاهده می شود انواع مختلفی وجود دارد ما بر روی user-encoded می گذاریم که با کد ما همخوانی داشته باشد.

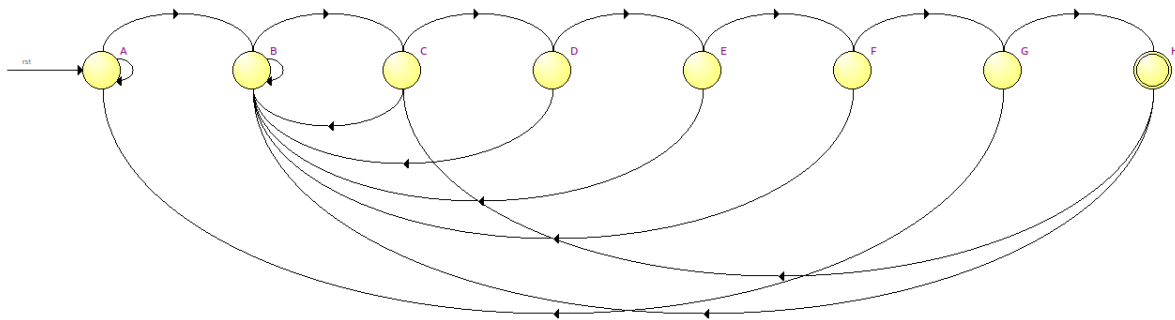
حال مجدد خروجی می گیریم که نتیجه به صورت زیر است:

Flow Status	Successful - Sun Dec 31 21:38:04 2023
Quartus Prime Version	23.1std.0 Build 991 11/28/2023 SC Lite Edition
Revision Name	ca5
Top-level Entity Name	qa
Family	Cyclone IV GX
Device	EP4CGX15BF14A7
Timing Models	Final
Total logic elements	4 / 14,400 ( < 1 % )
Total registers	3
Total pins	4 / 81 ( 5 % )
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 % )



## بررسی قطعات مدار و پیاده سازی آن

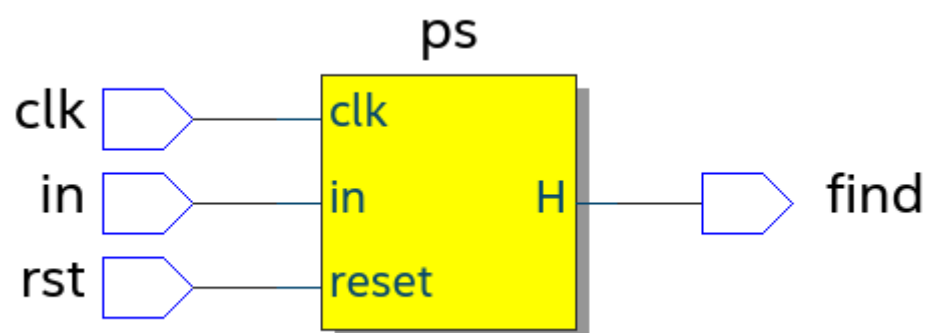
نمودار و همچنین encode آن نیز تطابق دارد.



	Name	ps~6	ps~5	ps~4
1	A	0	0	0
2	B	0	0	1
3	C	0	1	0
4	D	0	1	1
5	E	1	0	0
6	F	1	0	1
7	G	1	1	0
8	H	1	1	1

	Source State	Destination State	Condition
1	A	B	(in)
2	A	A	(lin)
3	B	C	(in)
4	B	B	(lin)
5	C	D	(in)
6	C	B	(lin)
7	D	B	(lin)
8	D	E	(in)
9	E	F	(in)
10	E	B	(lin)
11	F	B	(lin)
12	F	G	(in)
13	G	A	(lin)
14	G	H	(lin)
15	H	C	(in)
16	H	B	(lin)

مدار آن نیز به شکل زیر است:



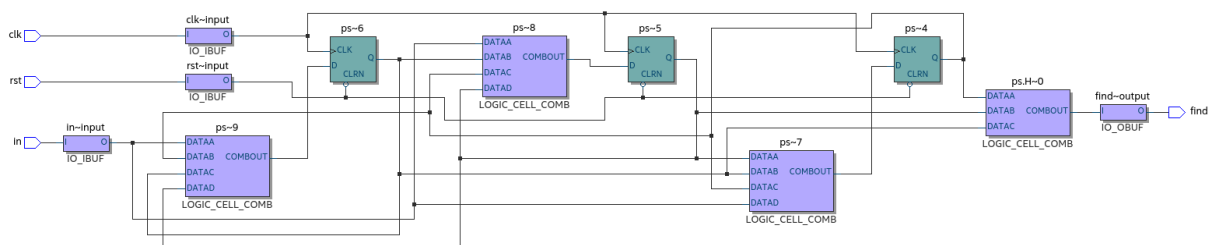
در گزارش می بینیم:

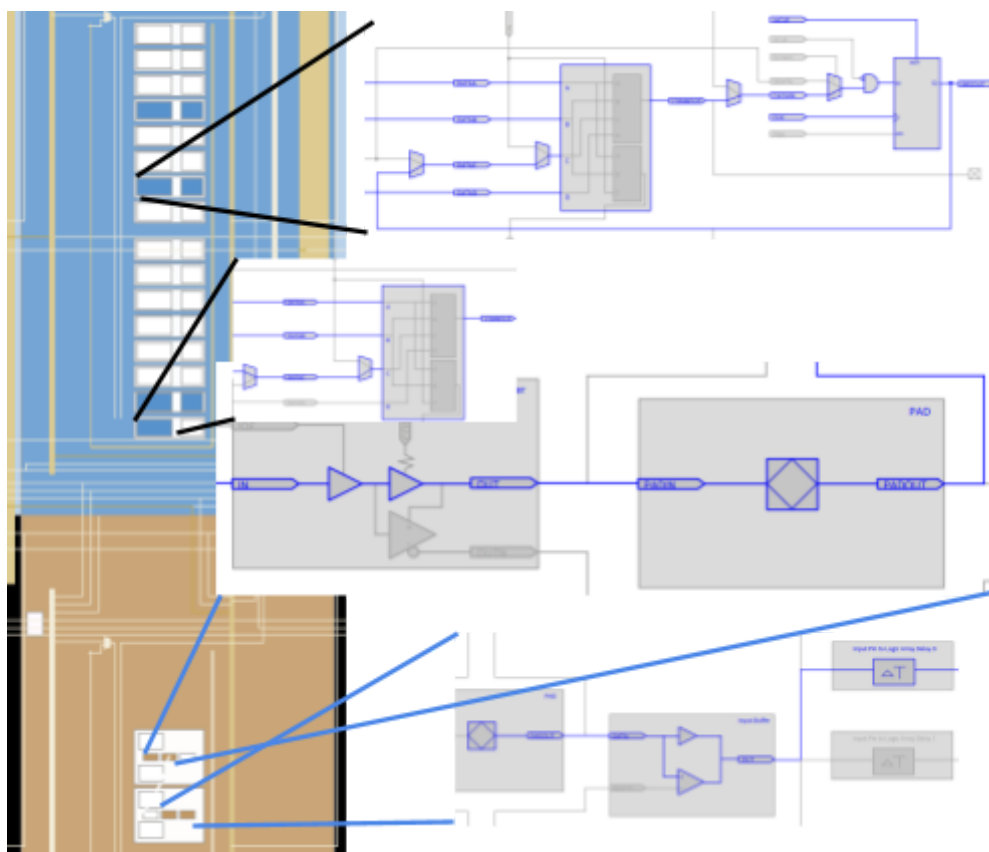
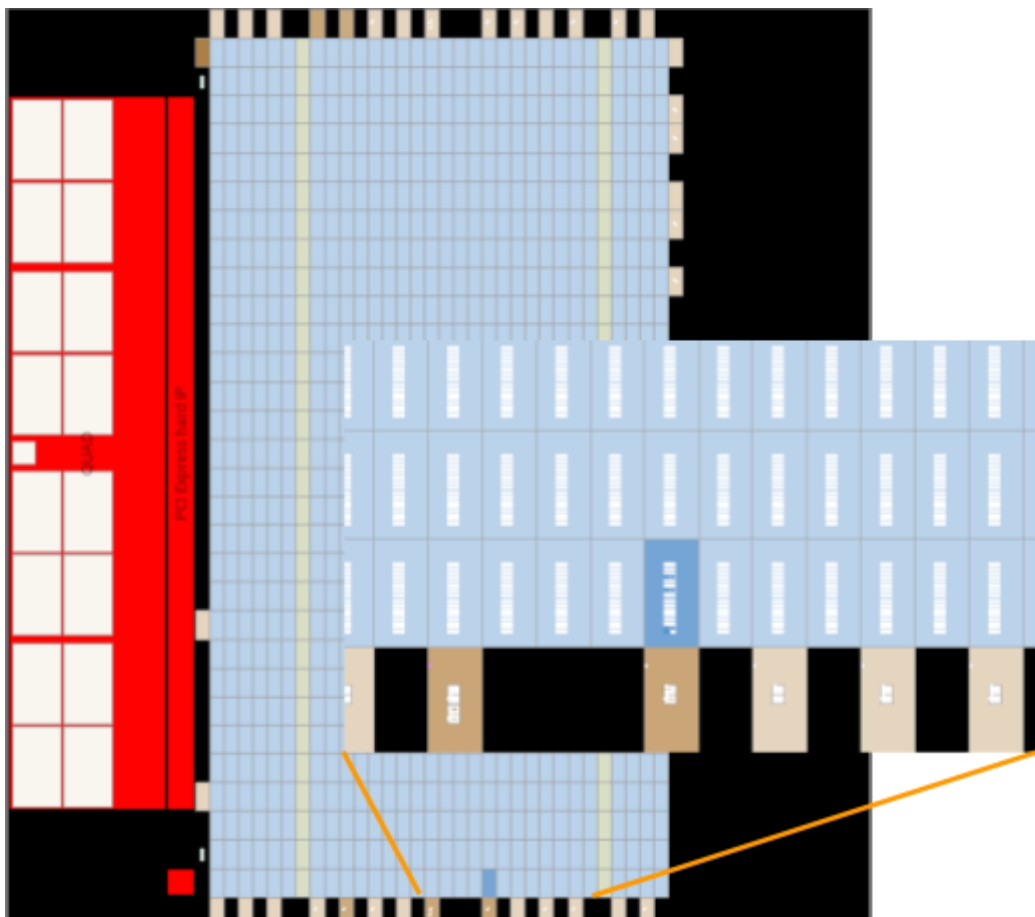
	Statistic	Value
1	Total registers	3
2	Number of registers using Synchronous Clear	0
3	Number of registers using Synchronous Load	0
4	Number of registers using Asynchronous Clear	3
5	Number of registers using Asynchronous Load	0
6	Number of registers using Clock Enable	0
7	Number of registers using Preset	0

که همان ۳ رجیستری است که در شکل دستی هست. همچنین داریم که:

	Resource	Usage
1	I/O pins	4
2		
3	DSP block 9-bit elements	0
4		
5	Maximum fan-out node	ps~4
6	Maximum fan-out	4
7	Total fan-out	29
8	Average fan-out	1.93


که نشان می دهد ما ۴ ورودی داریم که در شکل نیز مشخص بود آنها `find`, `clk`, `rst`, `in` هستند. همچنین در گزارش کلی می بینیم که ما از 4/14,400 مدار یعنی کمتر از ۱ درصد آن استفاده کردیم. برای درک بهتر این موضوع پیاده سازی آن را روی برد بررسی می کنیم.






## بررسی مدار از لحاظ زمانی

ابتدا داده های زمانی خود کوارتز را بررسی می کنیم.

Slow 1200mV 125C Model Fmax Summary				
 <<Filter>>				
	Fmax	Restricted Fmax	Clock Name	Note
1	782.47 MHz	250.0 MHz	clk	limit due to minimum period restriction (max I/O toggle rate)

Slow 1200mV -40C Model Fmax Summary				
 <<Filter>>				
	Fmax	Restricted Fmax	Clock Name	Note
1	933.71 MHz	250.0 MHz	clk	limit due to minimum period restriction (max I/O toggle rate)

در این قسمت گفته شده که مدار در دمای ۱۲۵ درجه می تواند تا 782MHz کار کند اما به دلیل محدودیت سایکلون ۴ این مقدار حداکثر ۲۵۰ است همچنین در دمای ۴۰- درجه مدار می تواند با فرکانس ۹۳۳ کار کند. حال باید در مدل سیم مدار را شبیه سازی کنیم برای این کار ابتدا میز آزمایش خود را بروزرسانی می کنیم که به شکل زیر می شود:

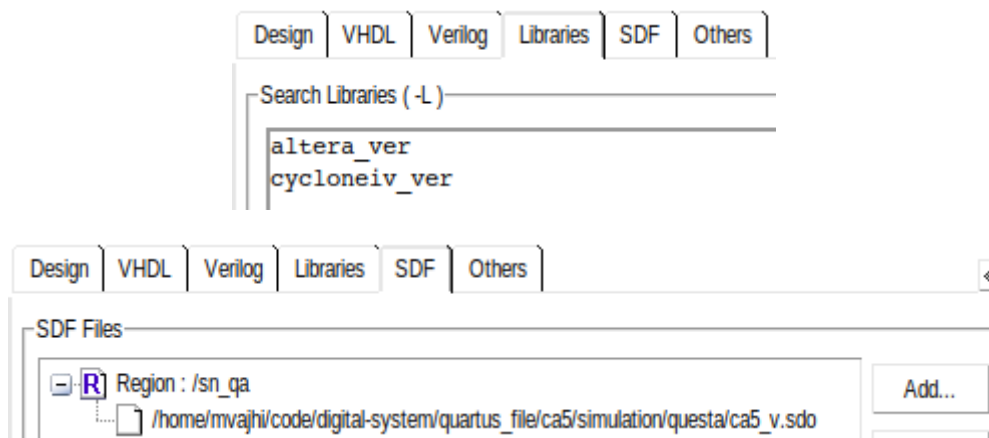
```
`timescale 1ns/1ns
module tb_qa();
    logic in, clk = 0, rst = 0;
    wire [1:0] out;

    pre_qa pre_sqd(.in(in), .clk(clk), .rst(rst), .find(out[0]));
    qa sqd(.in(in), .clk(clk), .rst(rst), .find(out[1]));

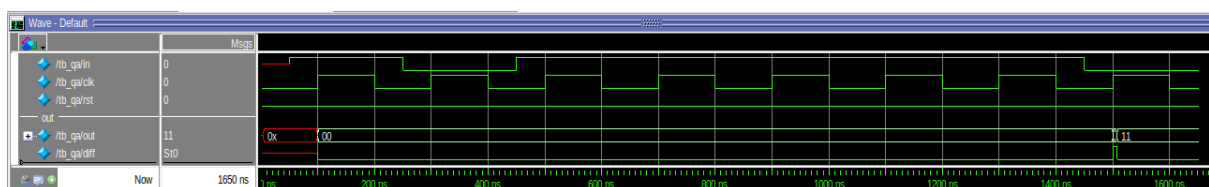
    wire diff;
    assign diff = ^out;

    always #100 clk = ~clk;
    initial begin
        #50 in = 1;
        #200 in = 0;
        #200 in = 1;
        #200 in = 1;
        #200 in = 1;
        #200 in = 1;
        #200 in = 1;
        #200 in = 0;
        #200 $stop;
    end
endmodule
```

سپس فایل ca5/simulation/ca5.svo را به مدل سیم اضافه می کنیم سپس برای شبیه سازی کتابخانه های زیر را انتخاب می کنیم و sdo مربوطه را به آن می دهیم (نکته مهم این است که فایل sdo باید در پوشه پروژه باشد):



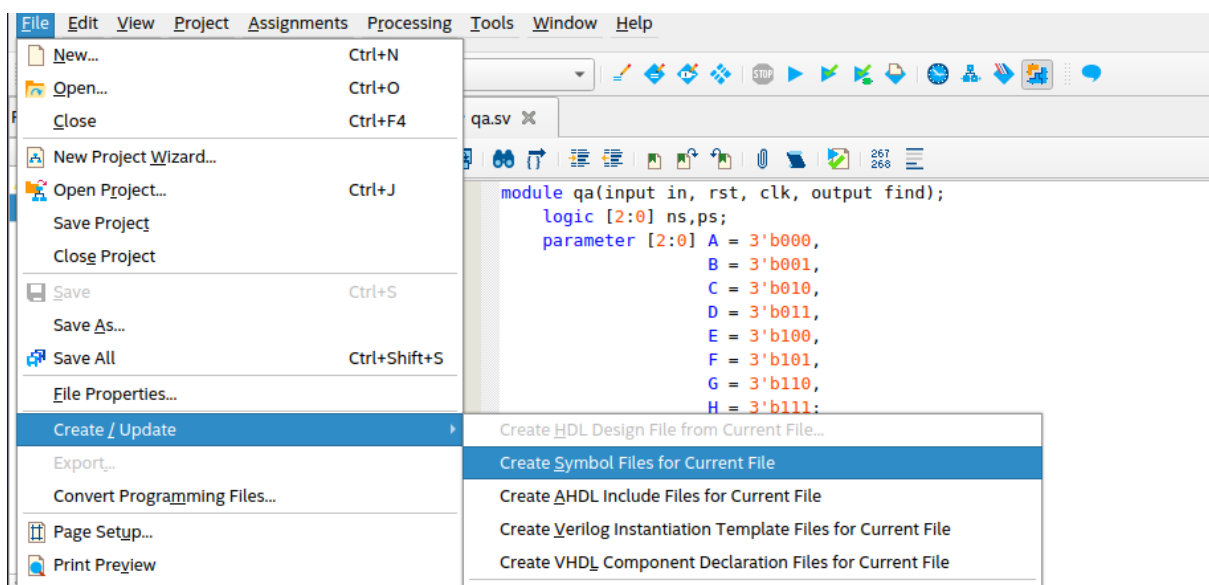
در نهایت خروجی به صورت زیر است:



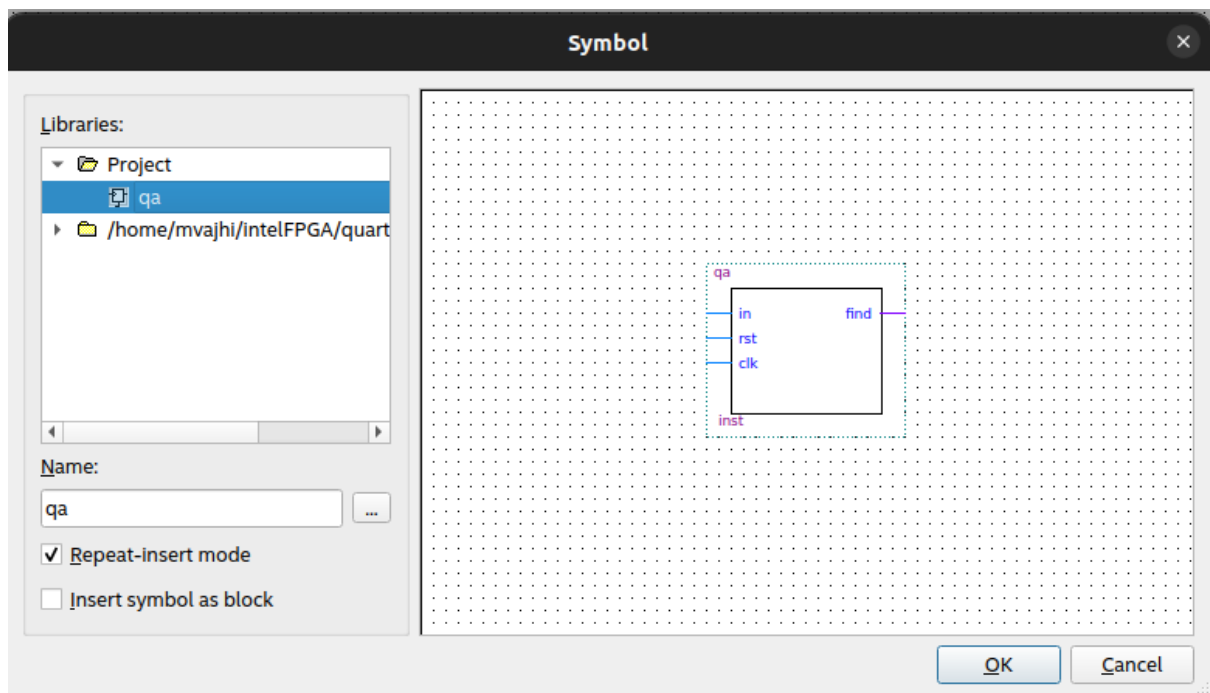
مشاهده می کنیم که بعد از synthesis در مدار تاخیر داریم.

## ساخت نماد در کوارتز

در آخر خواسته شده که آن را به عنوان یک نماد به کوارتز اضافه کنیم. برای این کار طبق توضیحات استاد در نوار بالا و در فایل گزینه ی موجود در تصویر را انتخاب می کنیم:



در آخر در قسمت مربوطه می توانیم نماد ایجاد شده را ببینیم:



## بخش b

### نوشتن کد

برای دریافت ۸ بیت nt یک شمارنده ۳ بیتی قرار می دهیم و یک شیفت رجیستر که si را از سمت چپ دریافت می کند. کد آن به شکل زیر است:

```
module qb(input rst, en, si, clk, output logic [7:0] out, output co);
    // counter
    logic [2:0] counter;
    always@(posedge clk, posedge rst) begin
        if (rst)
            counter <= 3'b0;
        else
            if (en)
                counter <= counter + 1;
    end

    assign co = en ? &counter : 1'b0;

    // shift reg
    always@(posedge clk) begin
        if (en)
            out <= {si, out[7:1]};
    end
endmodule
```

حال با نوشتن یک میز بررسی مدار خود را صحت سنجی می کنیم.

```
`timescale 1ns/1ns
module tb_qb();
    logic si, en = 1, clk = 0, rst = 1;
    wire [7:0] out;
    wire co;

    qb test(rst, en, si, clk, out, co);

    always #100 clk = ~clk;
    initial begin
        #1 rst = 0;
        #50 si = 1;
        #200 si = 0;
        #200 si = 1;
        #200 si = 1;
        #200 si = 0;
        #200 si = 1;
    end
endmodule
```

```

#200 si = 1;
#200 si = 0;
#200 en = 0;
#200 si = 1;
#200 si = 1;
#200 si = 1;
#200 si = 1;
#200 si = 1;
#200 si = 1;
#200 si = 1;
#200 si = 1;
#200 $stop;
end
endmodule

```

synthesizing

حال کد خود را synthesis می‌کنیم.

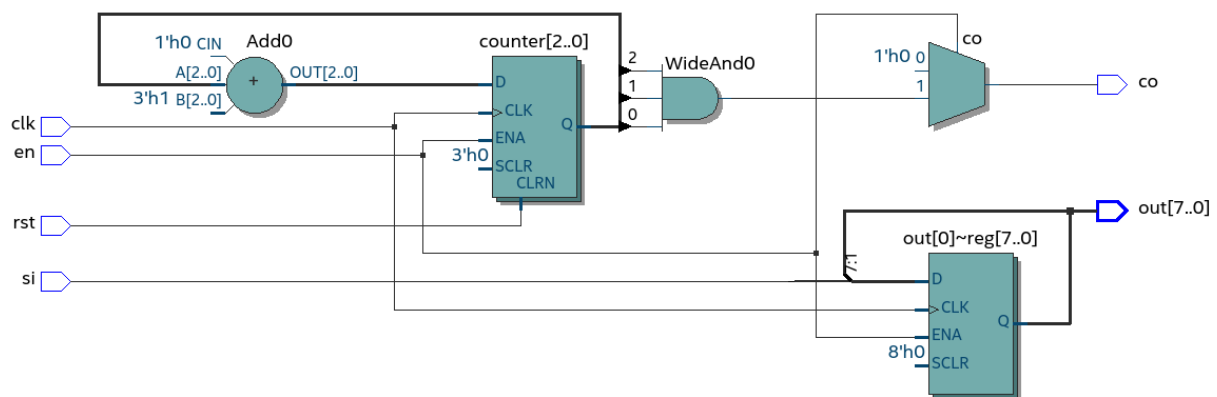
Flow Summary	
<<Filter>>	
Flow Status	Successful - Tue Jan 2 14:27:01 2024
Quartus Prime Version	23.1std.0 Build 991 11/28/2023 SC Lite Edition
Revision Name	ca5_qb
Top-level Entity Name	qb
Family	Cyclone IV GX
Device	EP4CGX15BF14A7
Timing Models	Final
Total logic elements	12 / 14,400 ( < 1 % )
Total registers	11
Total pins	13 / 81 ( 16 % )
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 % )



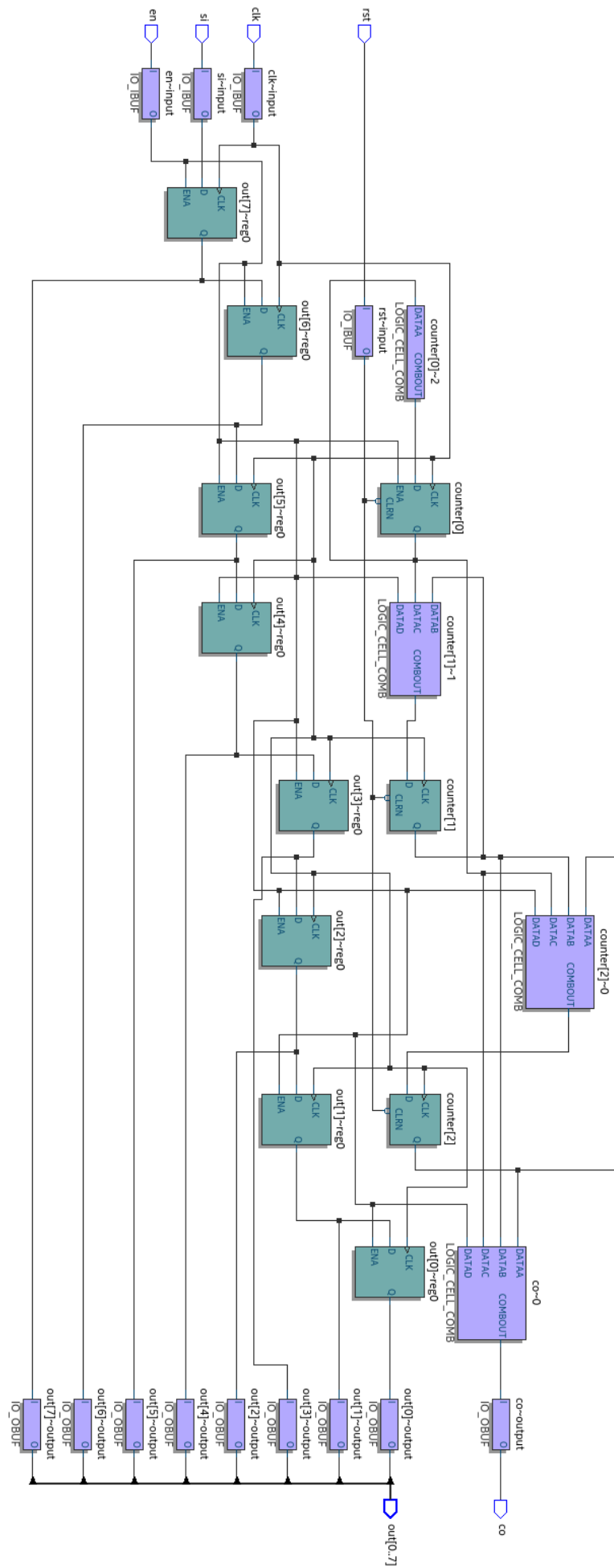
## بررسی قطعات مدار و پیاده سازی آن

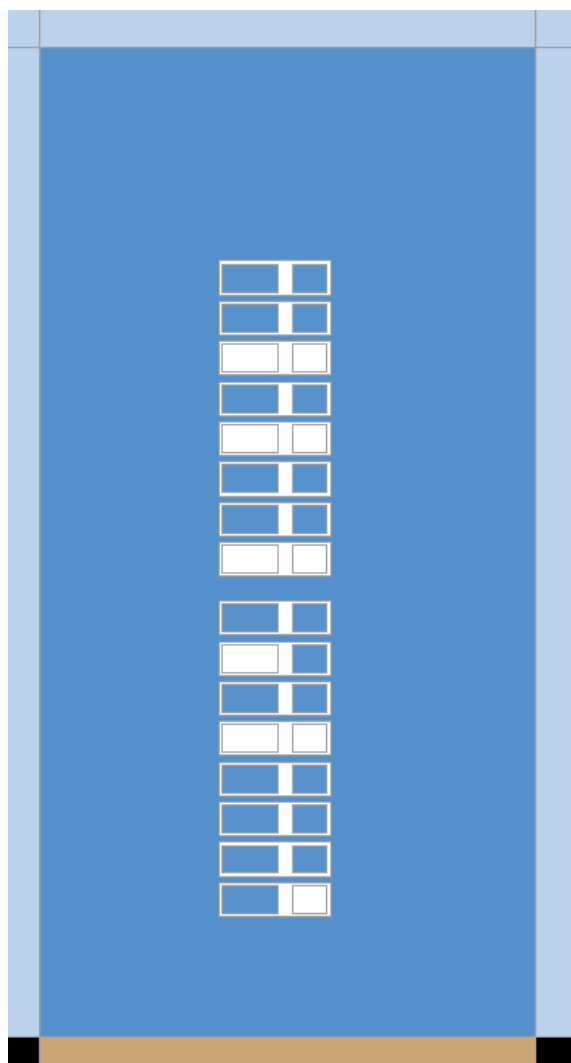
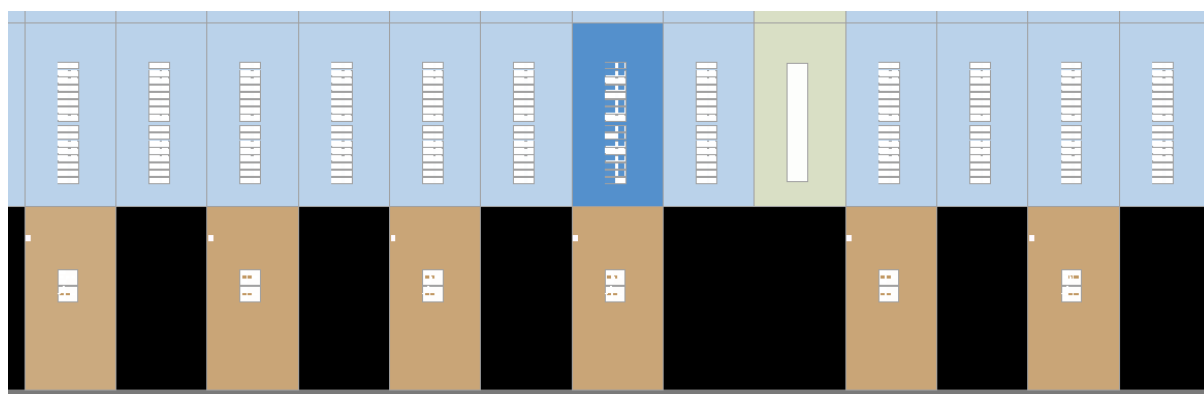
Analysis & Synthesis Summary	
<<Filter>>	
Analysis & Synthesis Status	Successful - Tue Jan 2 14:26:52 2024
Quartus Prime Version	23.1std.0 Build 991 11/28/2023 SC Lite Edition
Revision Name	ca5_qb
Top-level Entity Name	qb
Family	Cyclone IV GX
Total logic elements	12
Total registers	11
Total pins	13
Total virtual pins	0
Total memory bits	0
Embedded Multiplier 9-bit elements	0
Total GXB Receiver Channel PCS	0
Total GXB Receiver Channel PMA	0
Total GXB Transmitter Channel PCS	0
Total GXB Transmitter Channel PMA	0
Total PLLs	0

می بینید که ۱۱ رجیستر استفاده شده ۳ تا برای شمارنده و ۸ تا برای شیفت رجیستر. ۱۳ پین استفاده شده که صرف `output co`، `output logic [7:0] out`، `input rst`، `en`، `si`، `clk` شده است.



شکل مدار به صورت بالا است که آن را بررسی می کنیم. بخش بالایی مدار مربوط به شمارنده ۳ بیتی است که `cout` هم طبق کد طراحی شده و بخش پایینی مربوط به رجیستر ۸ بیتی است که `nt` در آن ذخیره می شود و همان شیفت رجیستر است. سیگنال ورودی مدار `en` و سیگنال خروجی `co` است. پیاده سازی آن روی FPGA به شکل زیر است. پین ها که قهوه ای هستند ورودی هستند و مشاهده می شود که در این مدار ۵ سوکت استفاده شده.



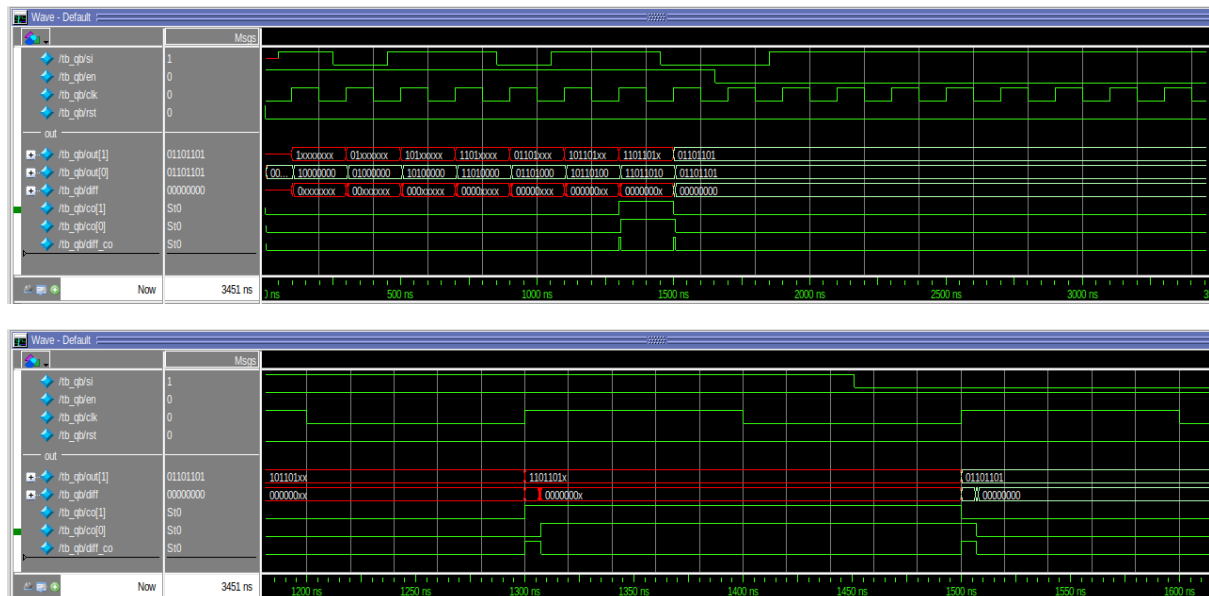


## زمان بندی

فرکانس ها در دما های مختلف به شرح زیر است:

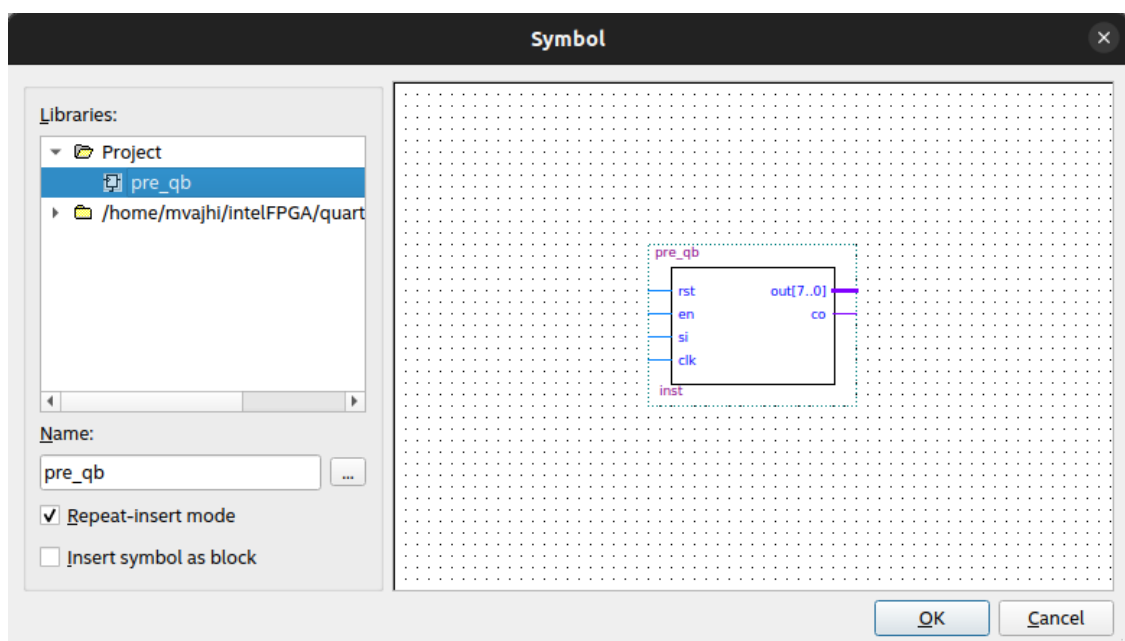
Slow 1200mV -40C Model Fmax Summary				Slow 1200mV 125C Model Fmax Summary			
<<Filter>>				<<Filter>>			
	Fmax	Restricted Fmax	Clock		Fmax	Restricted Fmax	Clock
1	1001.0 MHz	250.0 MHz	clk	1	845.31 MHz	250.0 MHz	clk

شکل موج آن به صورت است:



## ساخت نماد در کوارتز

مانند قسمت قبل عملیات مربوطه را انجام می‌دهیم.



## بخش C

### نوشتن کد

در این قسمت لازم است یک شمارنده کاهشی طراحی کنیم که از nt تا ۰ بشمارد. طراحی این مدار به آسانی میسر است اما مشکل اینجاست که بعد از کامل شدن nt در مدار قبلی یک clk لازم است تا مقدار آن روی شمارنده قرار گیرد و ما در اینجا یک clk از دست می‌دهیم و با یک clk تاخیر seroutvalid را ۱ می‌کنیم. برای جلوگیری از این موضوع در کنترلر اصلی مدار به محض دریافت سیگنال co از مدار قسمت قبلی مقدار seroutvalid را ۱ می‌کنیم تا clk بعدی که با توجه به مدار این قسمت تصمیم بگیریم و مقدار آن را تنظیم کنیم. فقط توجه کنید که چون یک clk طول می‌کشد مقدار روی شمارنده قرار گیرد پس باید یکی از آن کم شود و سپس روی مدار قرار گیرد. کد مدار به شکل زیر است:

```
module qc (input logic clk, input logic ld, input logic [7:0] pi, output
logic co);
    logic [7:0] out;

    always @(posedge clk, posedge ld) begin
        if (ld) begin
            out <= pi - 1;
        end else begin
            out <= out - 1;
        end
    end

    assign co = ~|out;

endmodule
```

و میز بررسی آن به صورت زیر است:

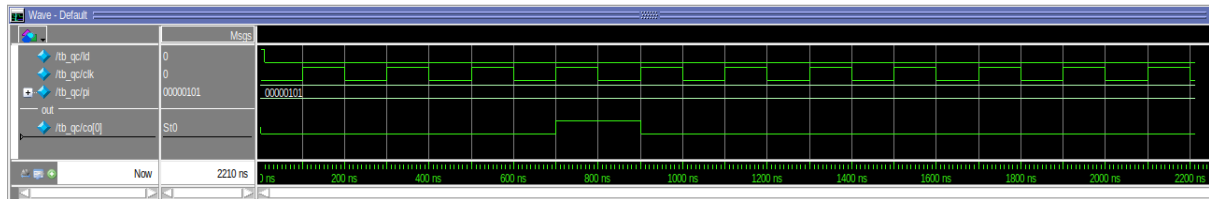
```
`timescale 1ns/1ns
module tb_qc();
    logic ld = 1, clk = 0;
    logic [7:0] pi = 5;
    wire co;

    qc test(clk, ld, pi, co);

    always #100 clk = ~clk;
    initial begin
        #10 ld = 0;
        repeat (10) #200 ld = 0;
        #200 $stop;
    end
endmodule
```

```
end
endmodule
```

و خروجی آن به صورت زیر است. با توجه به این که مقدار ۵ روی مدار قرار گرفته انتظار می رود که CO بعد از ۴ clk برابر ۱ شود زیرا طبق توضیحات بالا باید یکی کمتر از مقدار ورودی عمل کند.



synthesizing

خروجی بعد از synthesis به صورت زیر است:

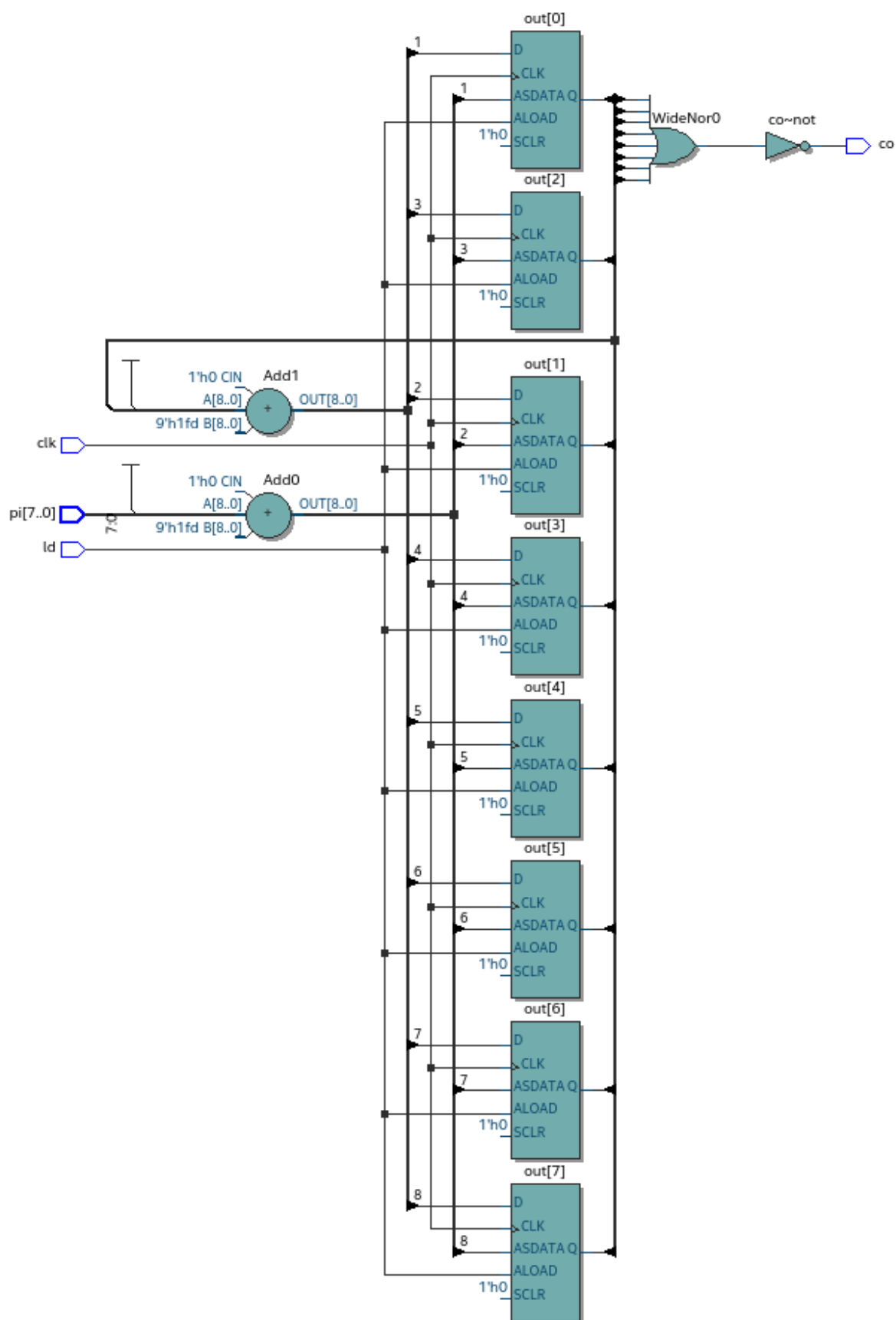
Flow Summary	
<<Filter>>	
Flow Status	Successful - Tue Jan 2 21:03:42 2024
Quartus Prime Version	23.1std.0 Build 991 11/28/2023 SC Lite Edition
Revision Name	ca5_qc
Top-level Entity Name	qc
Family	Cyclone IV GX
Device	EP4CGX15BF14A7
Timing Models	Final
Total logic elements	43 / 14,400 ( < 1 % )
Total registers	8
Total pins	11 / 81 ( 14 % )
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 % )

## بررسی قطعات مدار و پیاده سازی آن

خلاصه گزارش مدار به شرح زیر است:

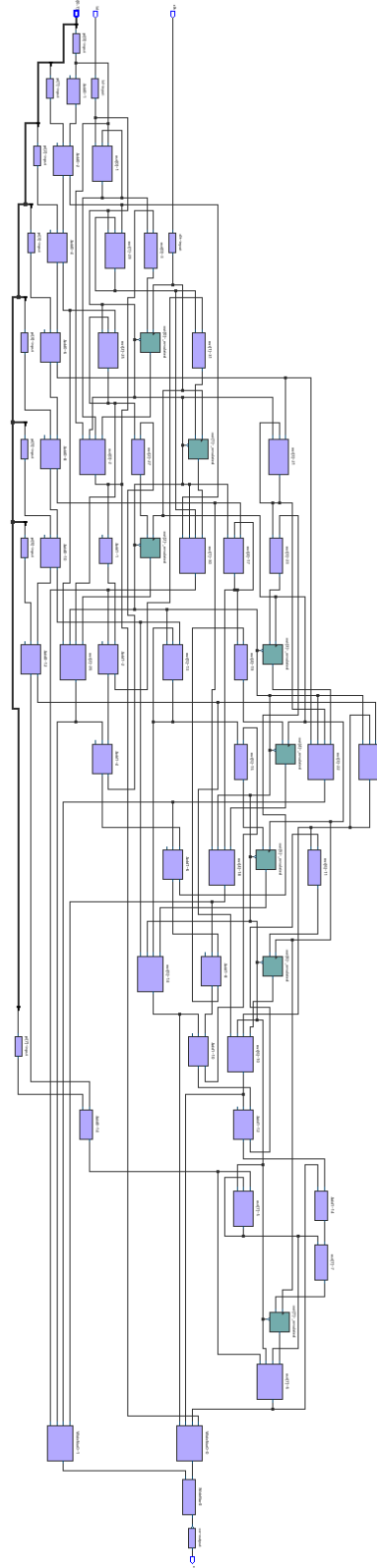
Analysis & Synthesis Summary	
<<Filter>>	
Analysis & Synthesis Status	Successful - Tue Jan 2 21:03:35 2024
Quartus Prime Version	23.1std.0 Build 991 11/28/2023 SC Lite Edition
Revision Name	ca5_qc
Top-level Entity Name	qc
Family	Cyclone IV GX
Total logic elements	43
Total registers	8
Total pins	11
Total virtual pins	0
Total memory bits	0
Embedded Multiplier 9-bit elements	0
Total GXB Receiver Channel PCS	0
Total GXB Receiver Channel PMA	0
Total GXB Transmitter Channel PCS	0
Total GXB Transmitter Channel PMA	0
Total PLLs	0

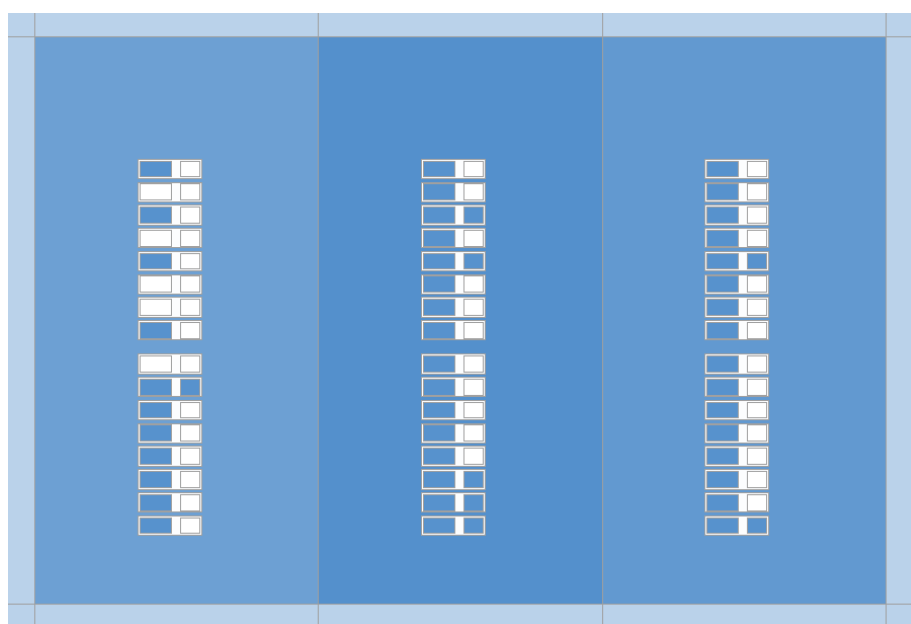
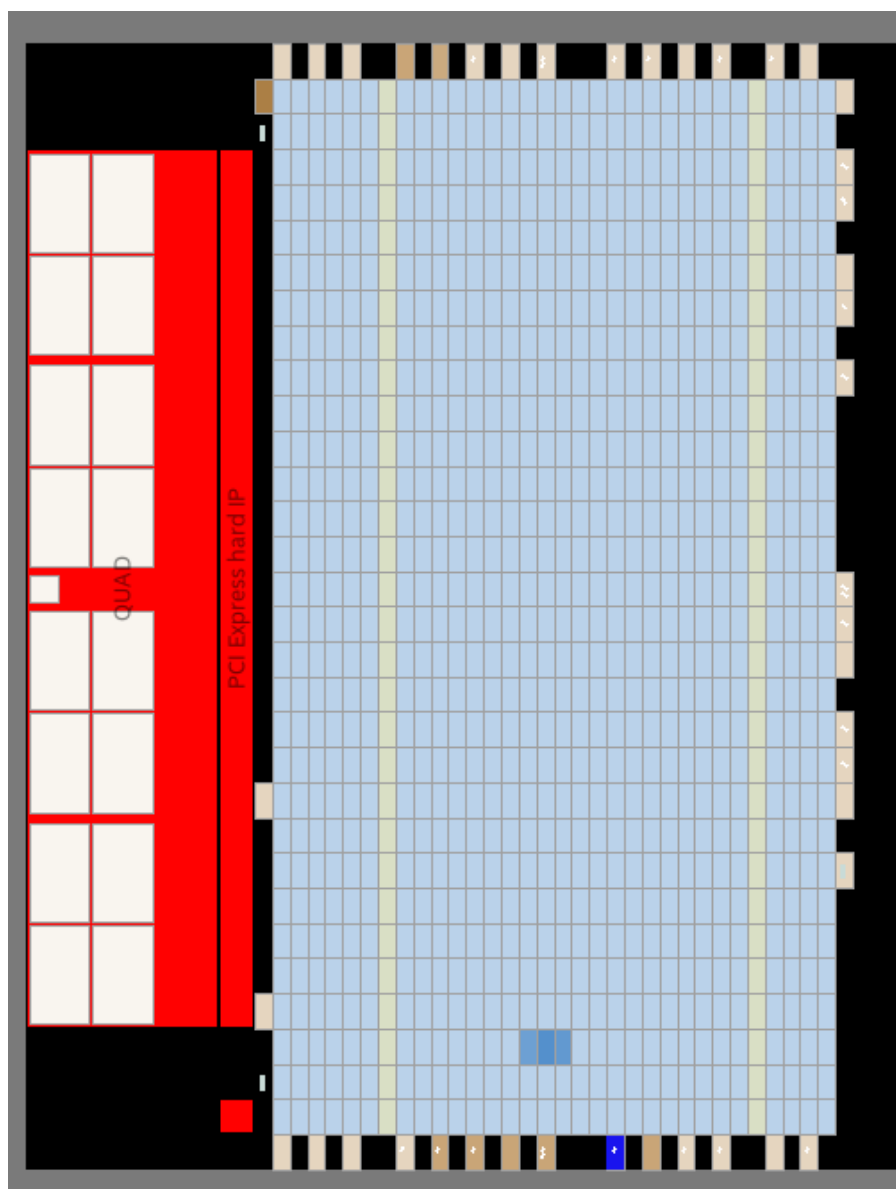
۸ رجیستر استفاده شده مربوط به شمارنده ۸ بیت رجیستر درون شمارنده کاهشی است. ۱۱ پین استفاده شده که مربوط به `output` `pi`، `input logic [7:0]` `ld`، `input logic clk` می باشد. همچنین مدار آن به صورت زیر است:





همانطور که مشاهده می شود از مدار به این صورت است که بسته به این که  $I_d$  فعال است یا نه مقدار را از روی خود یا از روی  $P_i$  می خواند همچنین مدار های کاهنده نیز در مدار قرار گرفته. البته که می شد به جای دو مدار جمع از یک مدار جمع استفاده کرد که تعداد گیت ها را کاهش می داد اما تاخیر را افزایش می داد. حال به سراغ پیاده سازی آن روی FPGA می رویم که به صورت زیر است:



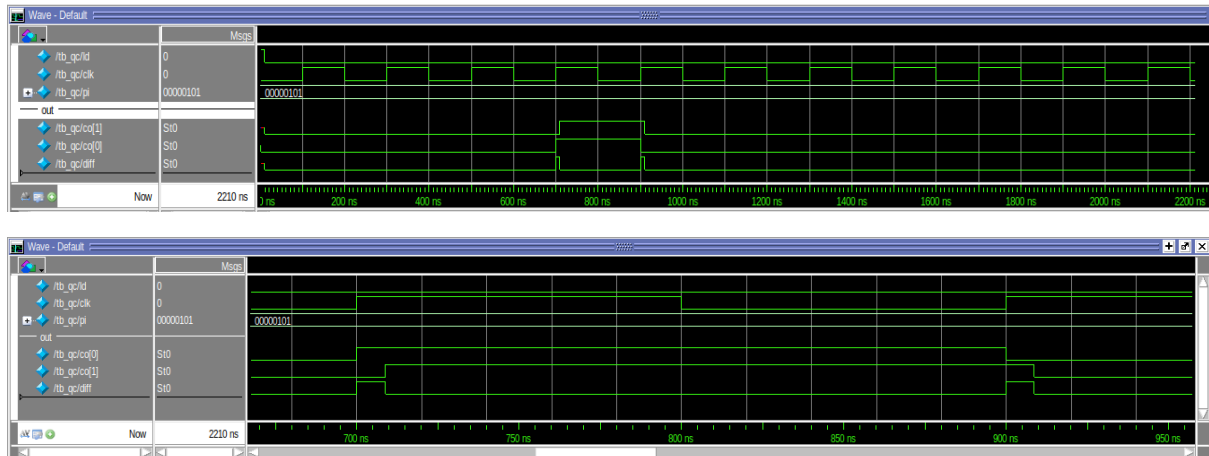


## زمانبندی

تحلیل فرکانس مدار در دما های مختلف به شرح زیر است:

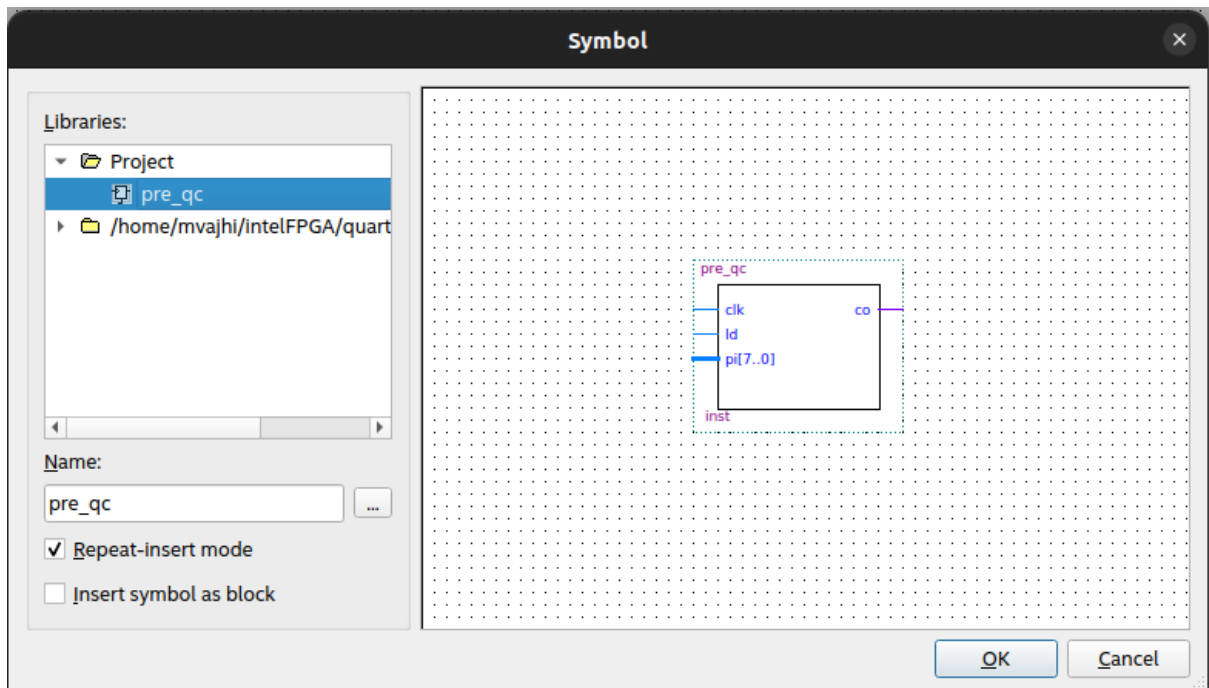
Slow 1200mV -40C Model Fmax Summary				Slow 1200mV 125C Model Fmax Summary			
<<Filter>>				<<Filter>>			
	Fmax	Restricted Fmax	Clock		Fmax	Restricted Fmax	Clock
1	265.04 MHz	250.0 MHz	clk	1	232.72 MHz	232.72 MHz	clk

شکل موج مدار به صورت زیر است:



## ساخت نماد در کوارتز

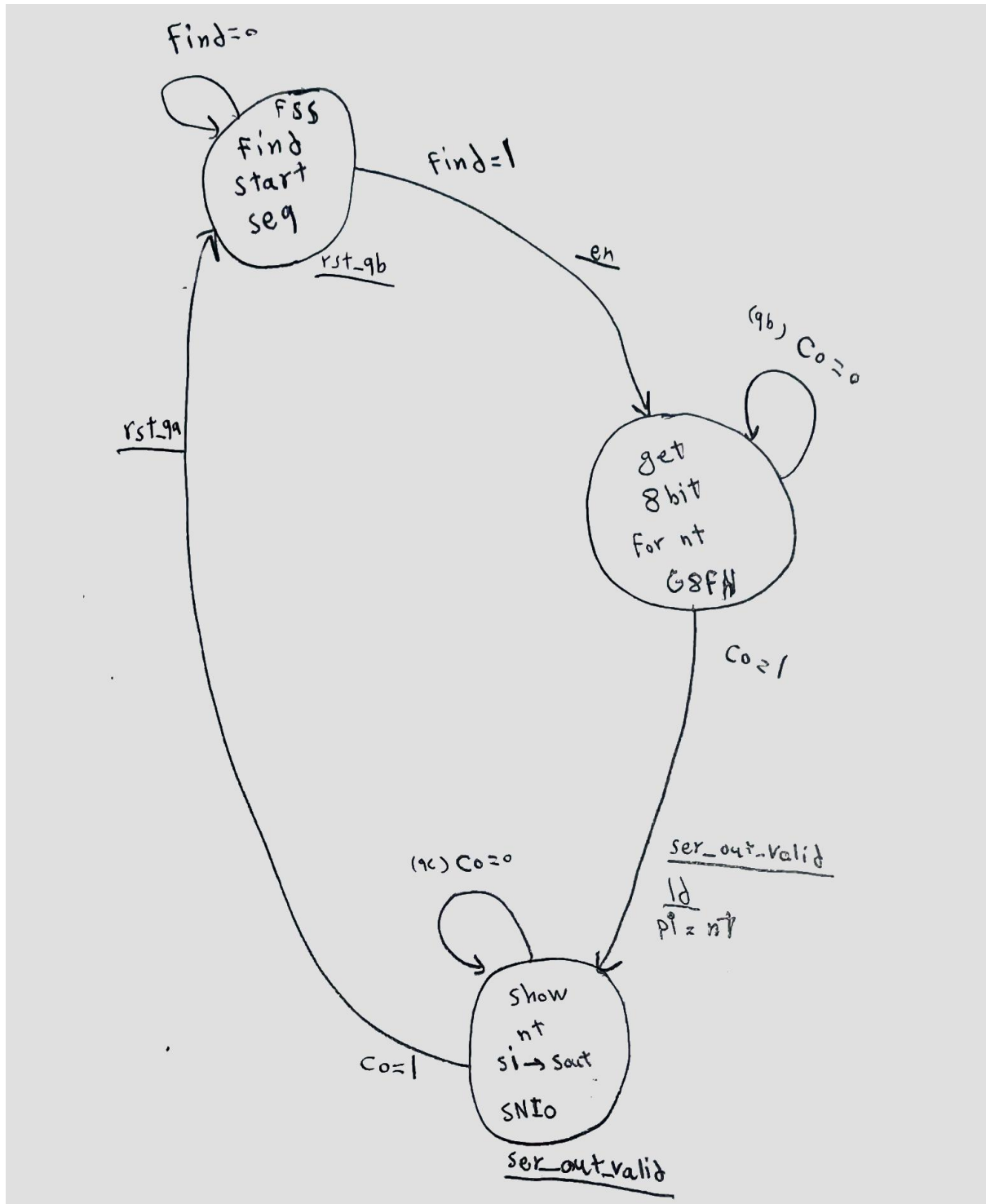
مانند قسمت قبل عملیات مربوطه را انجام می‌دهیم.



## بخش d

نوشتن کد

می خواهیم state ماشینی به صورت زیر طراحی کنیم.



که کد این کنترلر به صورت زیر است:

```
module qd (input clk, find_FSS, co_G8FN, co_SNIO,
           output logic seroutvalid, rst_FSS, rst_G8FN, en_G8FN, ld_SNIO);
  parameter S_FSS = 2'b00,
            S_G8FN = 2'b01,
            S_SNIO = 2'b10;
  logic[1:0] ps, ns;

  always @(find_FSS, co_G8FN, co_SNIO) begin
    ns = ps;
    case (ps)
      S_FSS: ns = find_FSS ? S_G8FN : S_FSS;
      S_G8FN: ns = co_G8FN ? S_SNIO : S_G8FN;
      S_SNIO: ns = co_SNIO ? S_FSS : S_SNIO;
      default: ns = S_FSS;
    endcase

    if (ps)
      rst_G8FN = 1'b1;
    else
      rst_G8FN = 1'b0;

    if (ns == S_G8FN || ps == S_G8FN)
      en_G8FN = 1'b1;
    else
      en_G8FN = 1'b0;

    if (ps == S_SNIO && co_SNIO == 1'b1)
      seroutvalid = 1'b1;
    else if (ps == S_G8FN && co_G8FN == 1'b1)
      seroutvalid = 1'b1;
    else
      seroutvalid = 1'b0;

    if (ps == S_G8FN && ns == S_SNIO)
      ld_SNIO = 1'b1;
    else
      ld_SNIO = 1'b0;

    if (ps == S_SNIO && co_SNIO == 1'b1)
      rst_FSS = 1'b1;
    else
      rst_FSS = 1'b0;
  end
```

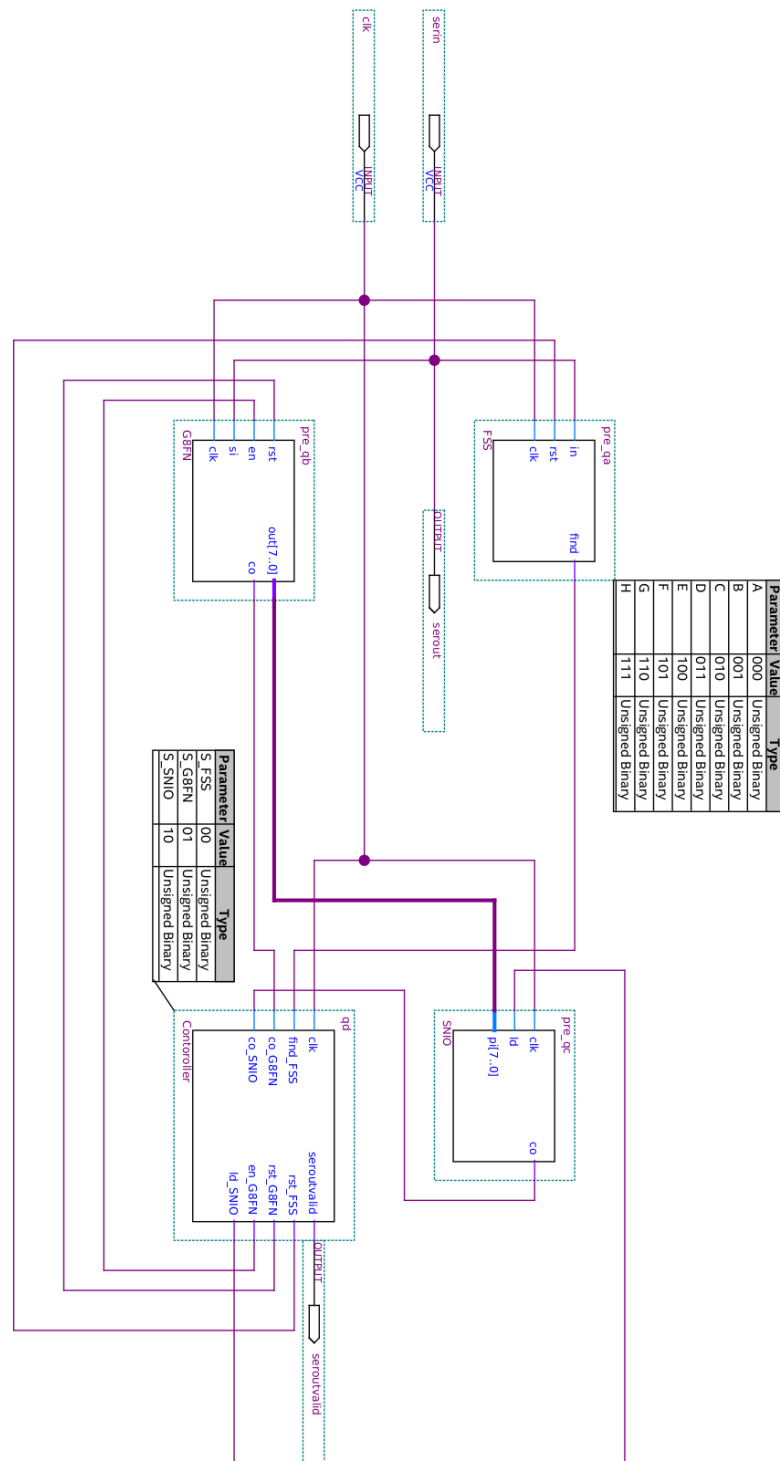
```

always @(posedge clk) begin
    ps <= ns;
end
endmodule

```

## طراحی مدار

در نهایت مدار کلی را در کوارتز می کشیم:



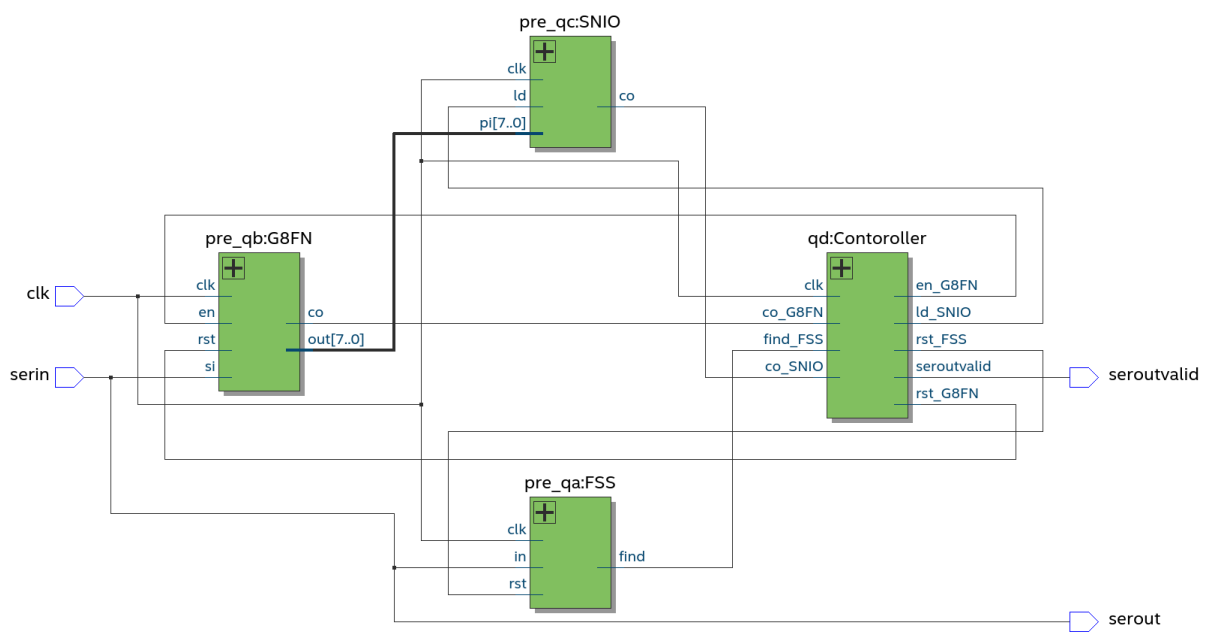
## synthesizing

Flow Summary	
<<Filter>>	
Flow Status	Successful - Wed Jan 3 01:42:33 2024
Quartus Prime Version	23.1std.0 Build 991 11/28/2023 SC Lite Edition
Revision Name	ca5_qd
Top-level Entity Name	ca5_qd
Family	Cyclone IV GX
Device	EP4CGX15BF14A7
Timing Models	Final
Total logic elements	65 / 14,400 ( < 1 % )
Total registers	24
Total pins	4 / 81 ( 5 % )
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 % )

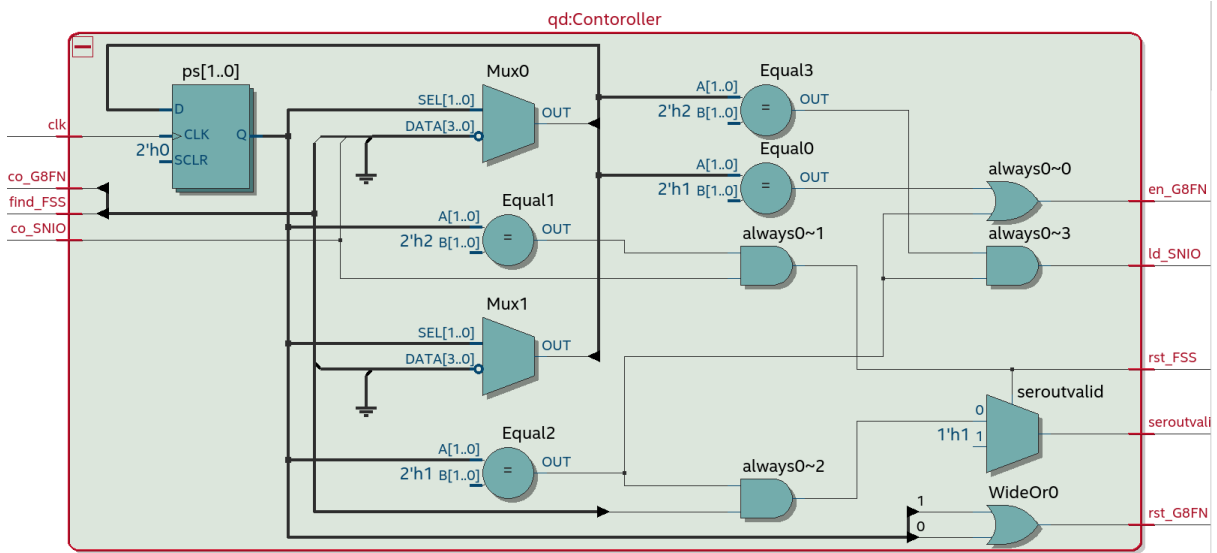
## بررسی قطعات مدار و پیاده سازی آن

Analysis & Synthesis Summary	
<<Filter>>	
Analysis & Synthesis Status	Successful - Wed Jan 3 01:42:25 2024
Quartus Prime Version	23.1std.0 Build 991 11/28/2023 SC Lite Edition
Revision Name	ca5_qd
Top-level Entity Name	ca5_qd
Family	Cyclone IV GX
Total logic elements	70
Total registers	24
Total pins	4
Total virtual pins	0
Total memory bits	0
Embedded Multiplier 9-bit elements	0
Total GXB Receiver Channel PCS	0
Total GXB Receiver Channel PMA	0
Total GXB Transmitter Channel PCS	0
Total GXB Transmitter Channel PMA	0
Total PLLs	0

تعداد رجیستر ها دقیقا برابر است با مجموع قسمت های قبل و ۲ عدد برای کنترلر این قسمت  $(۲۴=۲+۳+۱۱+۸)$ . تعداد پین ها هم با  $clk, serin, serout, seroutvalid$  برابر است. ساختار مدار مانند طراحی است:

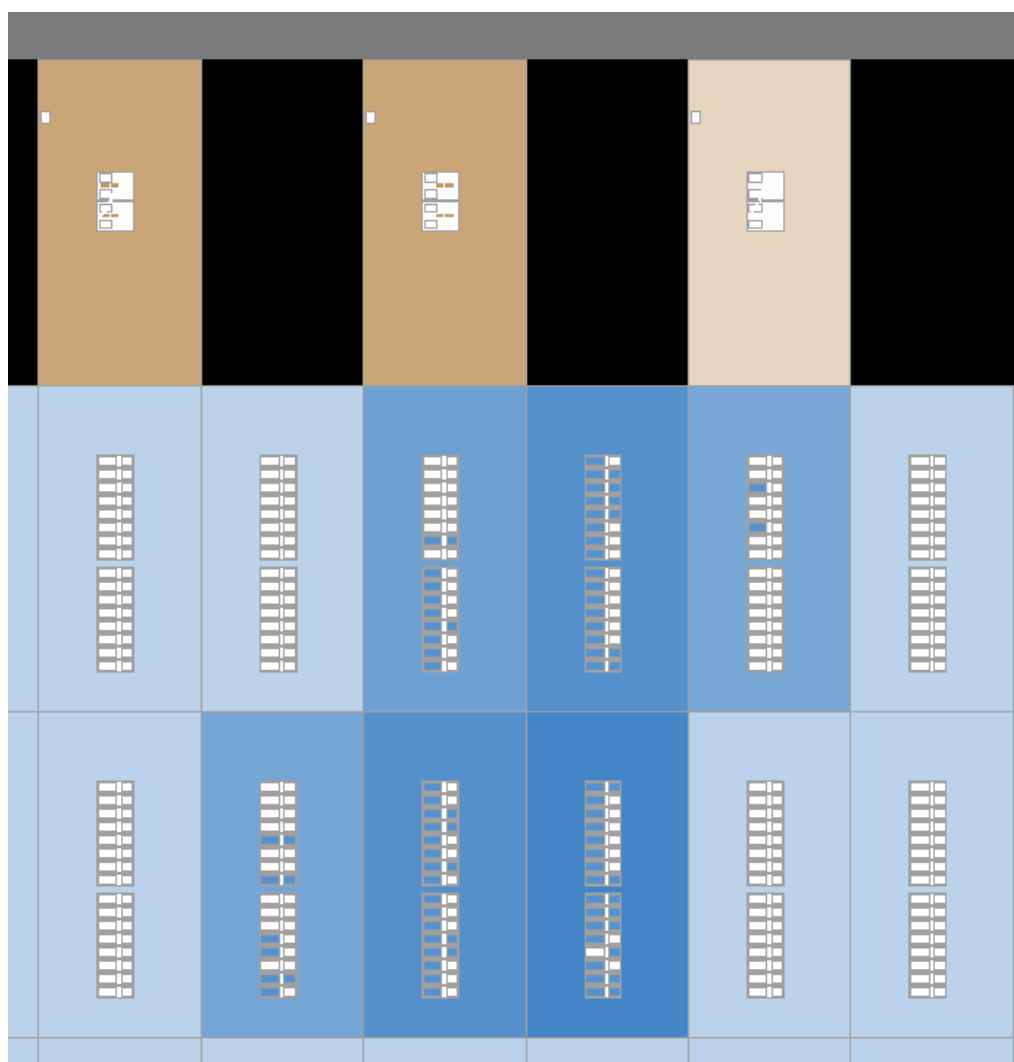
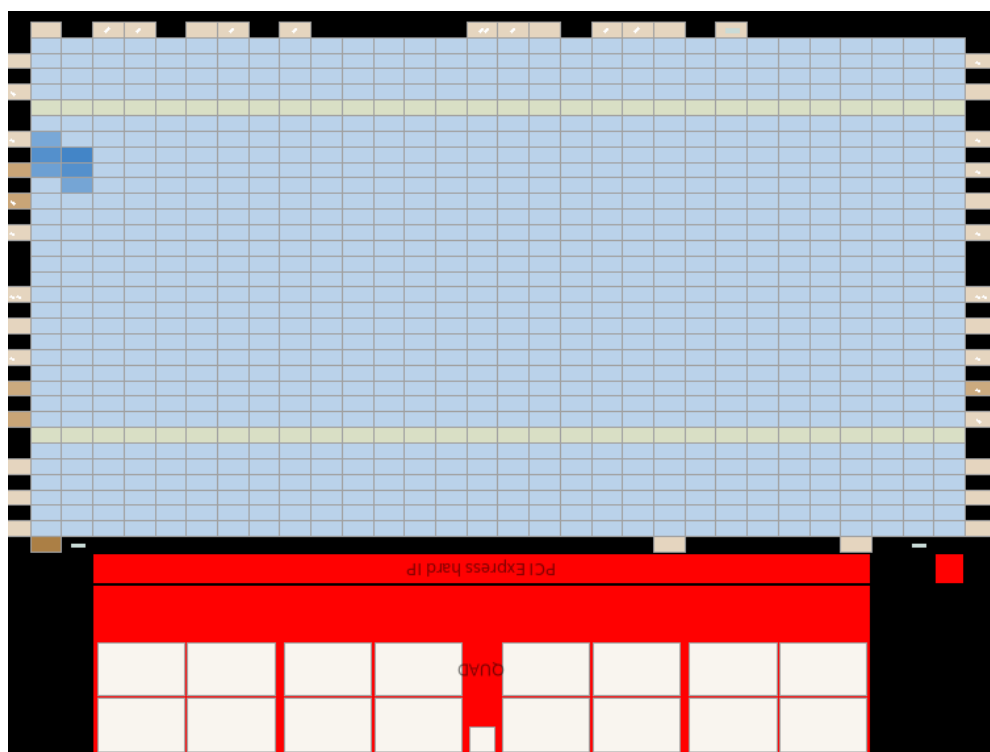


و ساختار کنترلر به شکل زیر:



حال به نمای FPGA نگاهی می اندازیم:





## زمان بندی

زمان بندی مدار در دما های مختلف به شرح زیر است:

Slow 1200mV -40C Model Fmax Summary				Slow 1200mV 125C Model Fmax Summary			
<<Filter>>				<<Filter>>			
	Fmax	Restricted Fmax	Clock		Fmax	Restricted Fmax	Clock
1	103.34 MHz	103.34 MHz	clk	1	92.1 MHz	92.1 MHz	clk

برای مدار میز آزمایش زیر را طراحی می کنیم:

```
`timescale 1ns/1ns
module tb_qd();
    logic clk = 0, serin = 0;
    wire seroutvalid, serout;

    ca5_qd test(seroutvalid, clk, serin, serout);

    always #500 clk = ~clk;
    initial begin
        #100 serin = 0;
        #1000 serin = 0;
        #1000 serin = 0;

        //start seq
        #1000 serin = 0;
        #1000 serin = 1;
        #1000 serin = 1;
        #1000 serin = 1;
        #1000 serin = 1;
        #1000 serin = 1;
        #1000 serin = 0;

        // set nt
        #1000 serin = 1;
        #1000 serin = 0;
        #1000 serin = 1;
        #1000 serin = 0;

        #1000 serin = 0;
        #1000 serin = 0;
        #1000 serin = 0;
        #1000 serin = 0;

        //out value
        #1000 serin = 1;
        #1000 serin = 0;
        #1000 serin = 1;
```

```

#1000 serin = 0;
#1000 serin = 1;

#1000 $stop;

end
endmodule

```

نتیجه به شکل زیر است:

