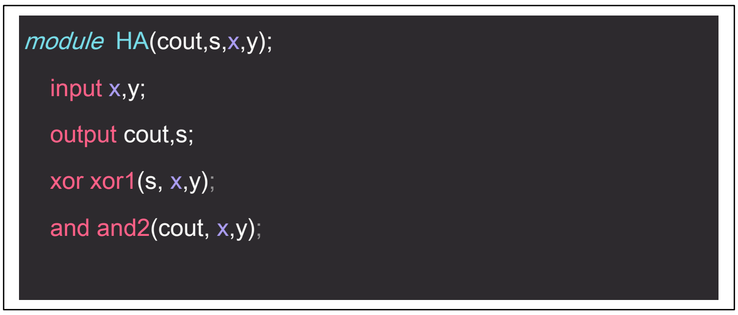
İSTANBUL TEKNİK ÜNİVERSİTESİ
1773

DIGITAL SYSTEM DESIGN APPLICATION PROJECT 4

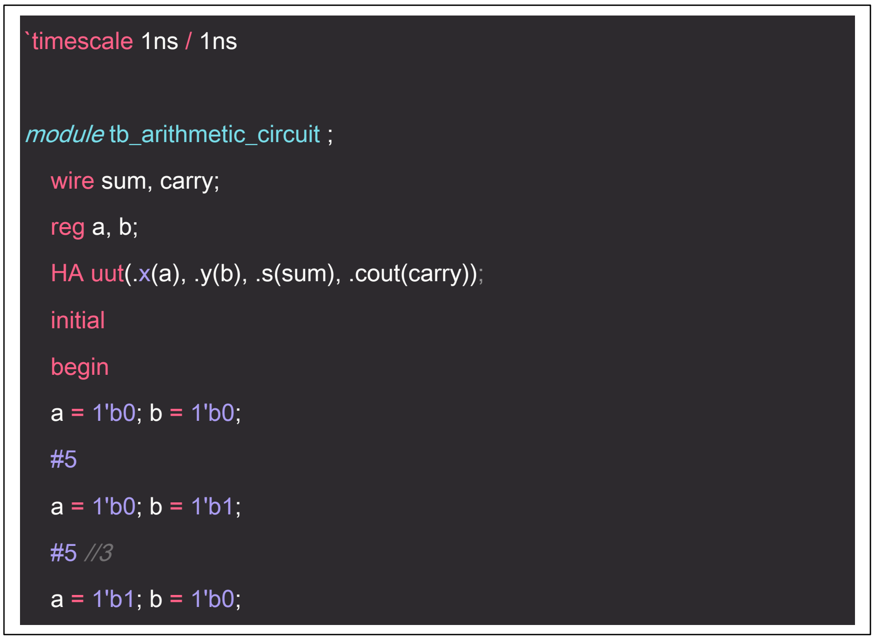
HALF ADDER

Half Adder Verilog code



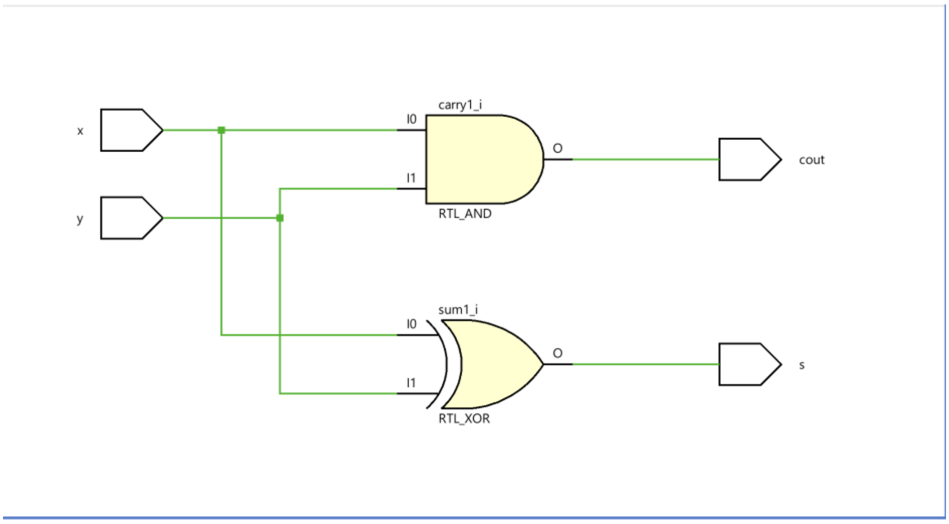
```
module  HA(cout,s,x,y);
   input x,y;
   output cout,s;
   xor xor1(s, x,y);
   and and2(cout, x,y);
```

Test code for Half Adder

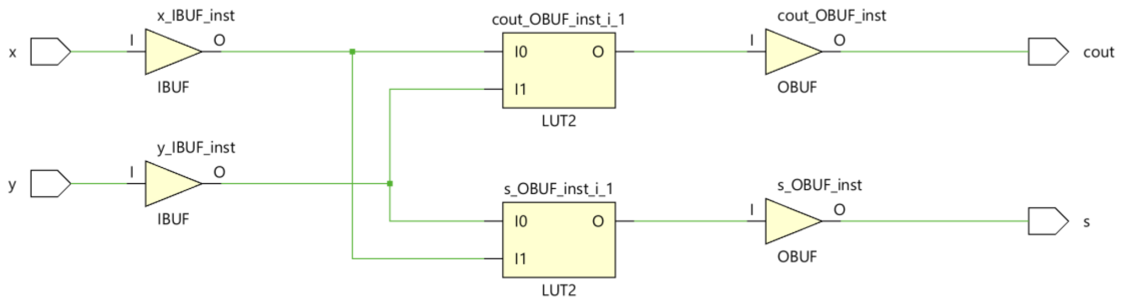


```
`timescale 1ns / 1ns

module tb_arithmetic_circuit ;
   wire sum, carry;
   reg a, b;
   HA uut(.x(a), .y(b), .s(sum), .cout(carry));
   initial
   begin
   a = 1'b0; b = 1'b0;
   #5
   a = 1'b0; b = 1'b1;
   #5 //3
   a = 1'b1; b = 1'b0;
```

Rtl schema

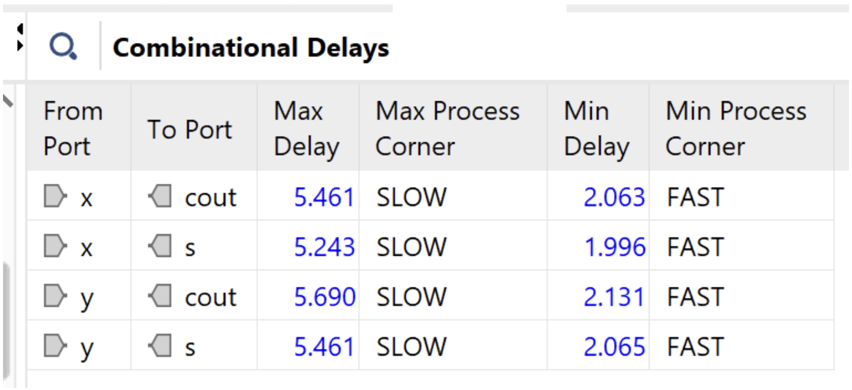


Technology schema



How many luts. = 1

Delay of circuit



Combinational Delays

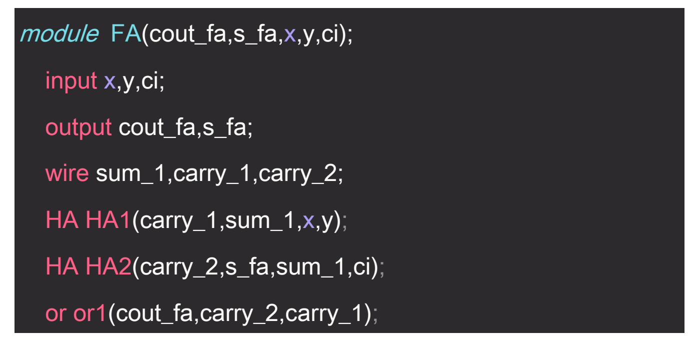
From Port	To Port	Max Delay	Max Process Corner	Min Delay	Min Process Corner
x	cout	5.461	SLOW	2.063	FAST
x	s	5.243	SLOW	1.996	FAST
y	cout	5.690	SLOW	2.131	FAST
y	s	5.461	SLOW	2.065	FAST

Comments:

We did half adder. Half adder add 1-bit two differnt input and if there are carry it Show that in cout output

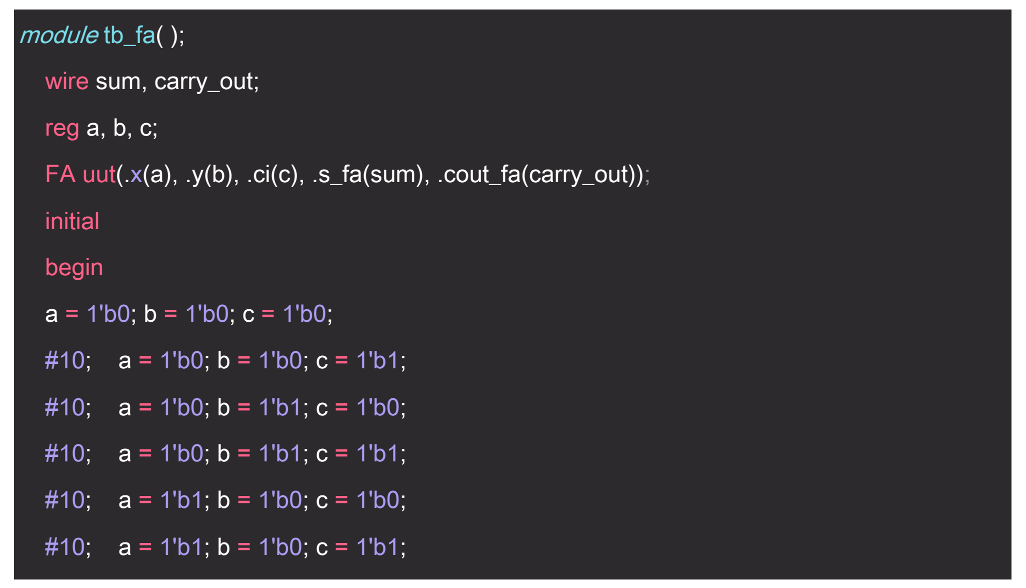
FULL ADDER

Full adder verilog code



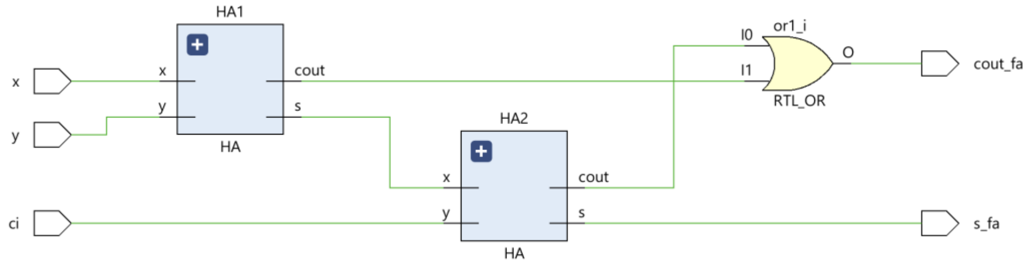
```
module  FA(cout_fa,s_fa,x,y,ci);
   input x,y,ci;
   output cout_fa,s_fa;
   wire sum_1,carry_1,carry_2;
   HA HA1(carry_1,sum_1,x,y);
   HA HA2(carry_2,s_fa,sum_1,ci);
   or or1(cout_fa,carry_2,carry_1);
```

Test code

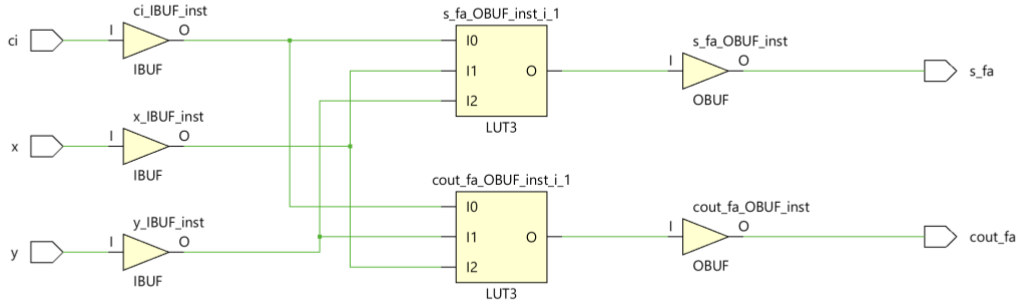


```
module tb_fa( );
   wire sum, carry_out;
   reg a, b, c;
   FA uut(.x(a), .y(b), .ci(c), .s_fa(sum), .cout_fa(carry_out));
   initial
   begin
   a = 1'b0; b = 1'b0; c = 1'b0;
   #10;    a = 1'b0; b = 1'b0; c = 1'b1;
   #10;    a = 1'b0; b = 1'b1; c = 1'b0;
   #10;    a = 1'b0; b = 1'b1; c = 1'b1;
   #10;    a = 1'b1; b = 1'b0; c = 1'b0;
   #10;    a = 1'b1; b = 1'b0; c = 1'b1;
```

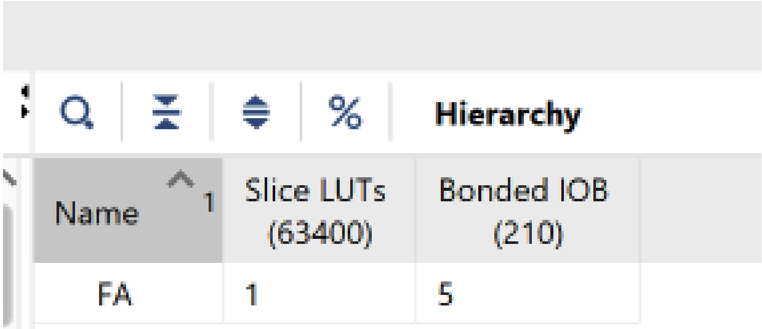
Rtl schema



Technology schema

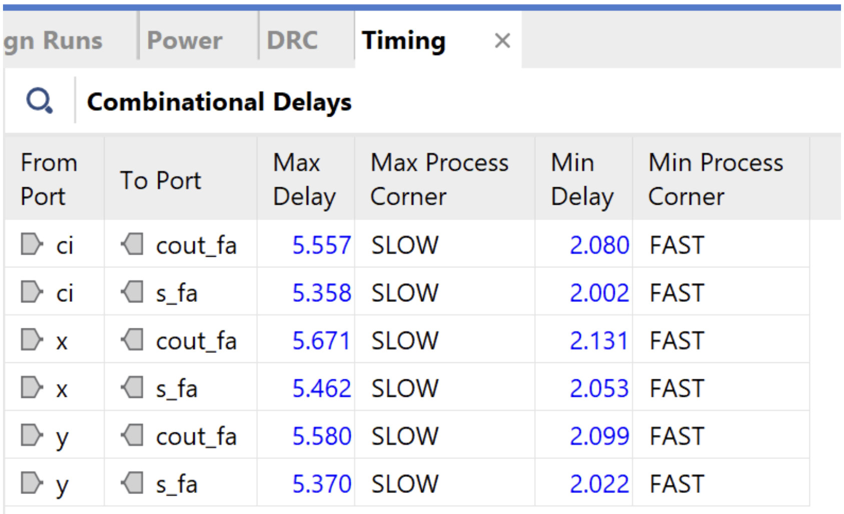


How many luts.



Name	Slice LUTs (63400)	Bonded IOB (210)
FA	1	5

Delay of circuit



Combinational Delays

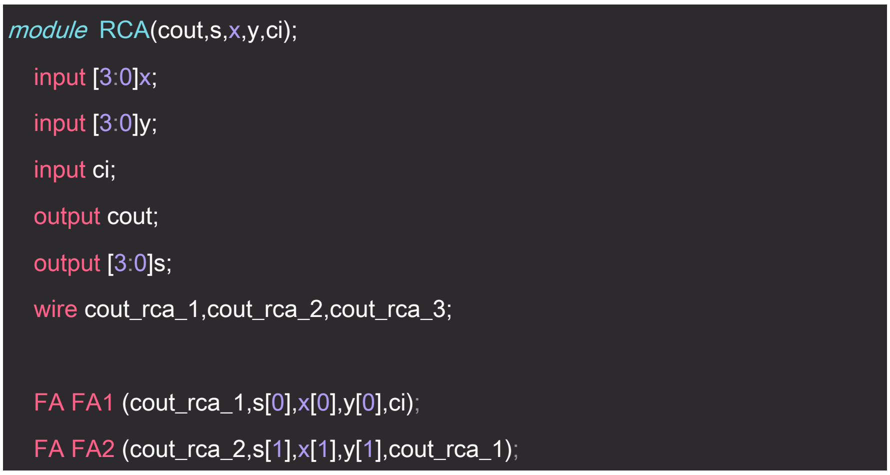
From Port	To Port	Max Delay	Max Process Corner	Min Delay	Min Process Corner
ci	cout_fa	5.557	SLOW	2.080	FAST
ci	s_fa	5.358	SLOW	2.002	FAST
x	cout_fa	5.671	SLOW	2.131	FAST
x	s_fa	5.462	SLOW	2.053	FAST
y	cout_fa	5.580	SLOW	2.099	FAST
y	s_fa	5.370	SLOW	2.022	FAST

Comments:

We did full adder wit 2 half adder. Full adder different from half adder in input number . full adder has 3 input but it do same thing that adding

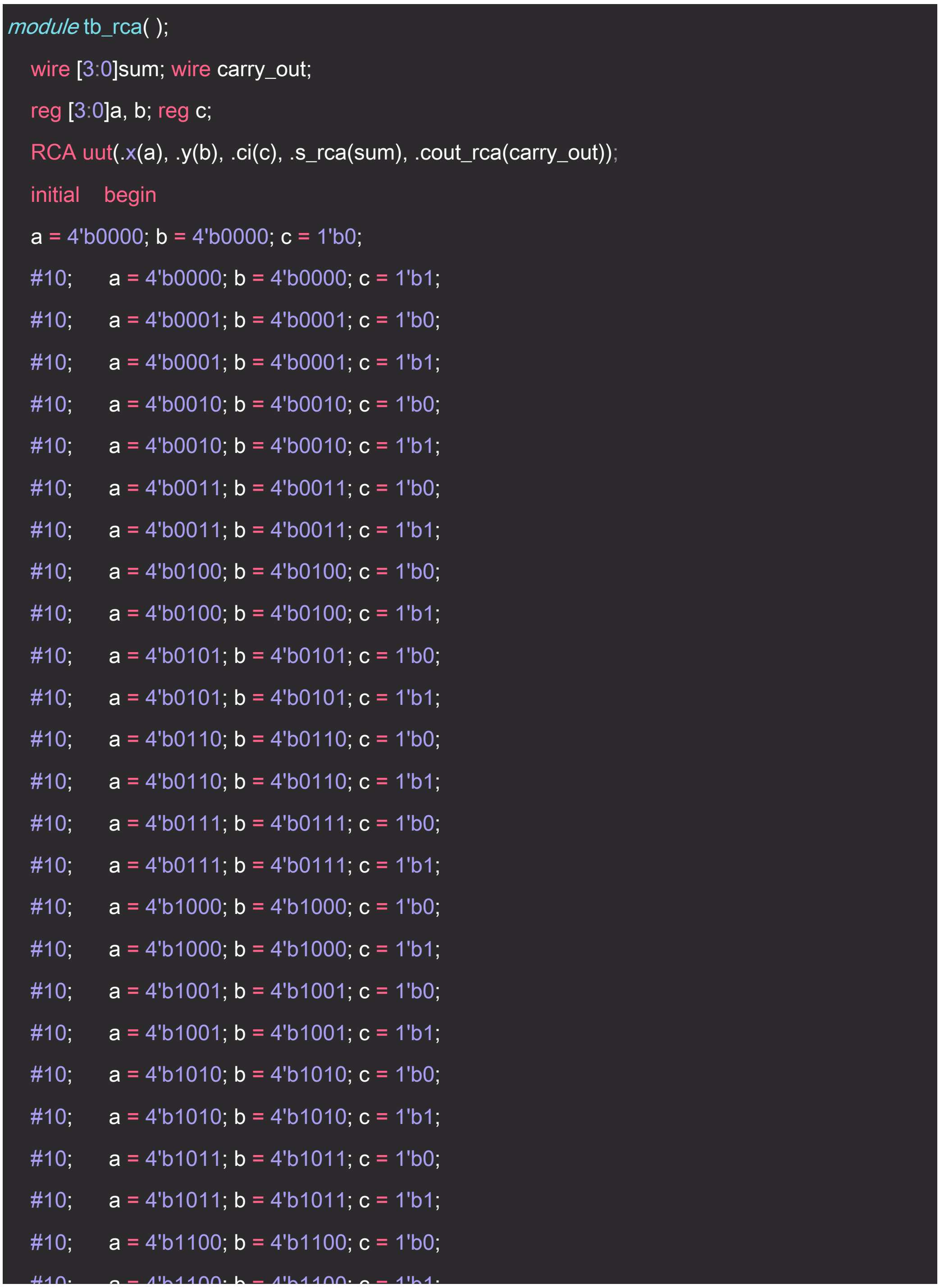
RIPPLE CARRY ADDER

Verilog code



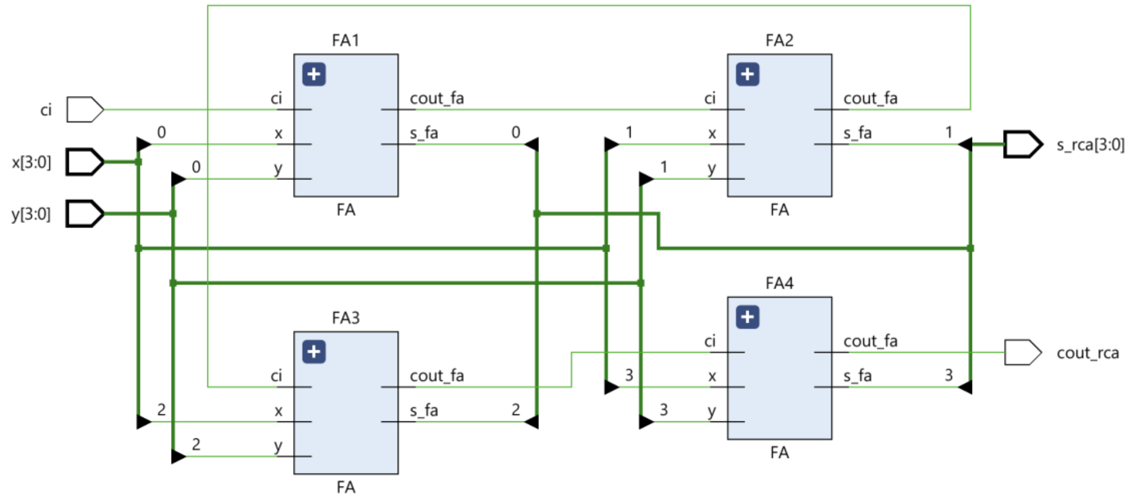
```
module  RCA(cout,s,x,y,ci);
   input [3:0]x;
   input [3:0]y;
   input ci;
   output cout;
   output [3:0]s;
   wire cout_rca_1,cout_rca_2,cout_rca_3;

   FA FA1 (cout_rca_1,s[0],x[0],y[0],ci);
   FA FA2 (cout_rca_2,s[1],x[1],y[1],cout_rca_1);
```

Test code

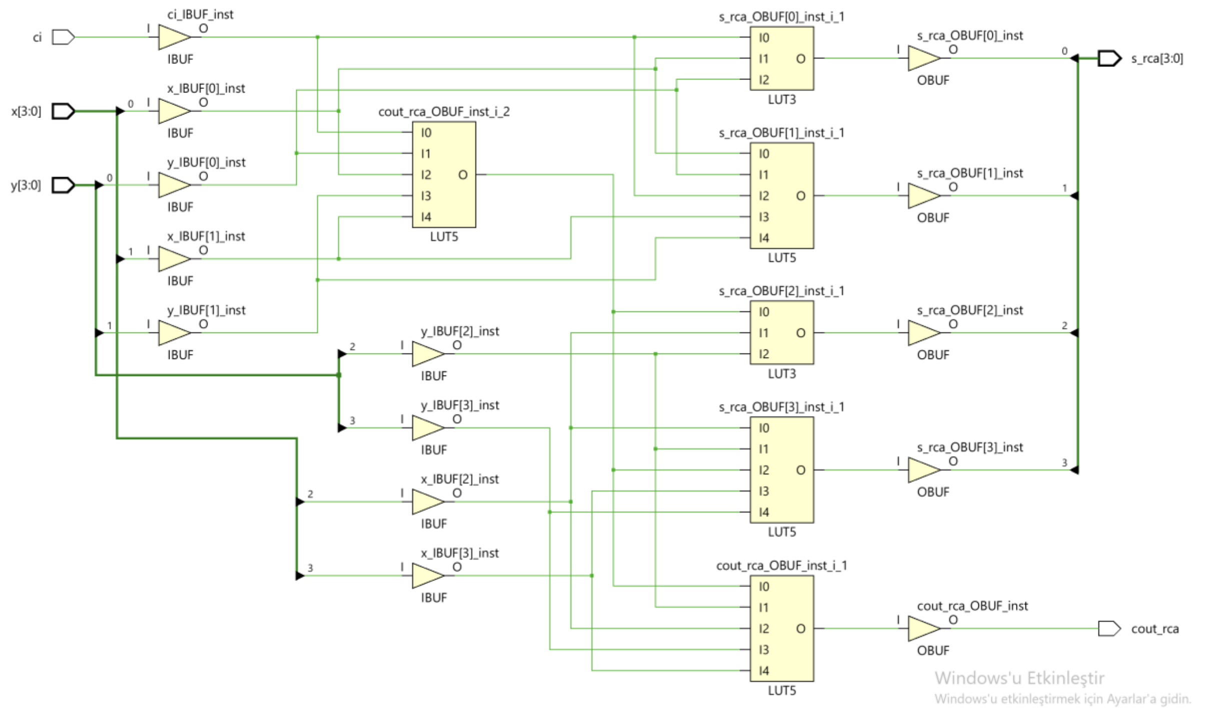


```
module tb_rca( );
   wire [3:0]sum; wire carry_out;
   reg [3:0]a, b; reg c;
   RCA uut(.x(a), .y(b), .ci(c), .s_rca(sum), .cout_rca(carry_out));
   initial    begin
   a = 4'b0000; b = 4'b0000; c = 1'b0;
   #10;      a = 4'b0000; b = 4'b0000; c = 1'b1;
   #10;      a = 4'b0001; b = 4'b0001; c = 1'b0;
   #10;      a = 4'b0001; b = 4'b0001; c = 1'b1;
   #10;      a = 4'b0010; b = 4'b0010; c = 1'b0;
   #10;      a = 4'b0010; b = 4'b0010; c = 1'b1;
   #10;      a = 4'b0011; b = 4'b0011; c = 1'b0;
   #10;      a = 4'b0011; b = 4'b0011; c = 1'b1;
   #10;      a = 4'b0100; b = 4'b0100; c = 1'b0;
   #10;      a = 4'b0100; b = 4'b0100; c = 1'b1;
   #10;      a = 4'b0101; b = 4'b0101; c = 1'b0;
   #10;      a = 4'b0101; b = 4'b0101; c = 1'b1;
   #10;      a = 4'b0110; b = 4'b0110; c = 1'b0;
   #10;      a = 4'b0110; b = 4'b0110; c = 1'b1;
   #10;      a = 4'b0111; b = 4'b0111; c = 1'b0;
   #10;      a = 4'b0111; b = 4'b0111; c = 1'b1;
   #10;      a = 4'b1000; b = 4'b1000; c = 1'b0;
   #10;      a = 4'b1000; b = 4'b1000; c = 1'b1;
   #10;      a = 4'b1001; b = 4'b1001; c = 1'b0;
   #10;      a = 4'b1001; b = 4'b1001; c = 1'b1;
   #10;      a = 4'b1010; b = 4'b1010; c = 1'b0;
   #10;      a = 4'b1010; b = 4'b1010; c = 1'b1;
   #10;      a = 4'b1011; b = 4'b1011; c = 1'b0;
   #10;      a = 4'b1011; b = 4'b1011; c = 1'b1;
   #10;      a = 4'b1100; b = 4'b1100; c = 1'b0;
   #10;      a = 4'b1100; b = 4'b1100; c = 1'b1;
```

Rtl schema



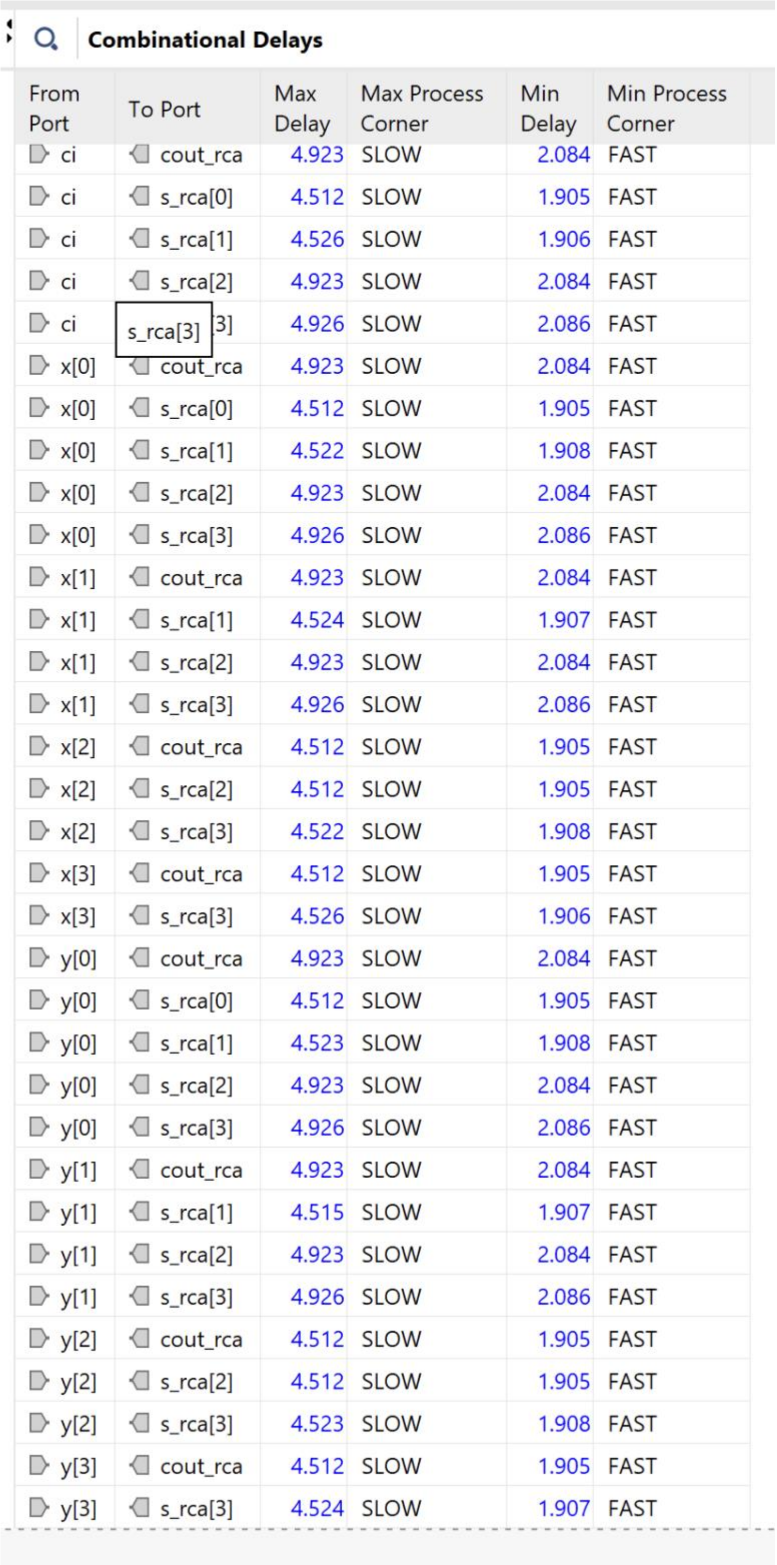
Technology schema



How many luts

4

Delay of circuit



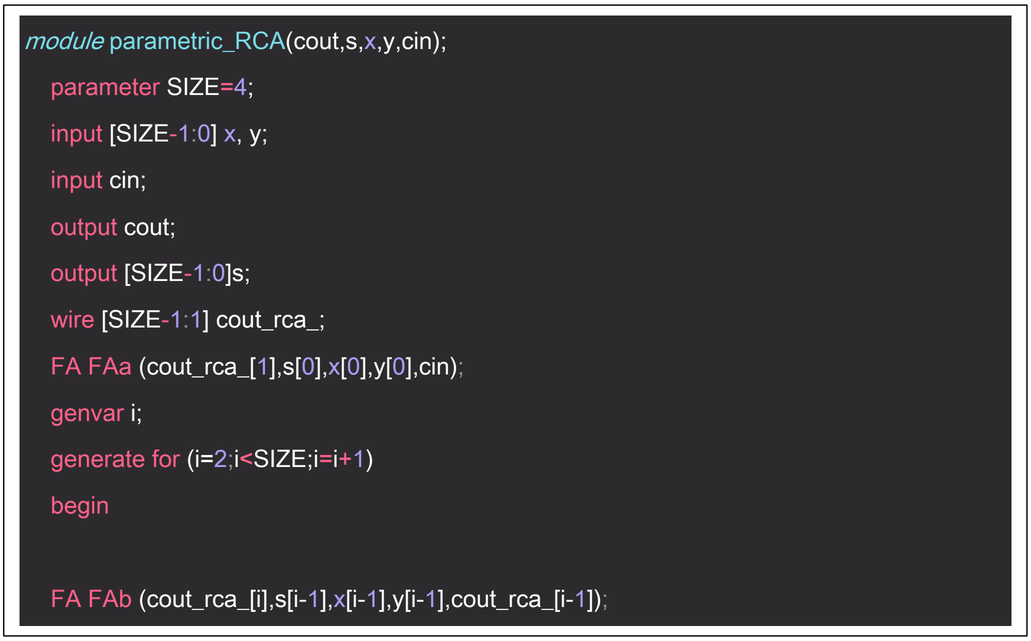
Combinational Delays

From Port	To Port	Max Delay	Max Process Corner	Min Delay	Min Process Corner
ci	cout_rca	4.923	SLOW	2.084	FAST
ci	s_rca[0]	4.512	SLOW	1.905	FAST
ci	s_rca[1]	4.526	SLOW	1.906	FAST
ci	s_rca[2]	4.923	SLOW	2.084	FAST
ci	s_rca[3]	4.926	SLOW	2.086	FAST
x[0]	cout_rca	4.923	SLOW	2.084	FAST
x[0]	s_rca[0]	4.512	SLOW	1.905	FAST
x[0]	s_rca[1]	4.522	SLOW	1.908	FAST
x[0]	s_rca[2]	4.923	SLOW	2.084	FAST
x[0]	s_rca[3]	4.926	SLOW	2.086	FAST
x[1]	cout_rca	4.923	SLOW	2.084	FAST
x[1]	s_rca[1]	4.524	SLOW	1.907	FAST
x[1]	s_rca[2]	4.923	SLOW	2.084	FAST
x[1]	s_rca[3]	4.926	SLOW	2.086	FAST
x[2]	cout_rca	4.512	SLOW	1.905	FAST
x[2]	s_rca[2]	4.512	SLOW	1.905	FAST
x[2]	s_rca[3]	4.522	SLOW	1.908	FAST
x[3]	cout_rca	4.512	SLOW	1.905	FAST
x[3]	s_rca[3]	4.526	SLOW	1.906	FAST
y[0]	cout_rca	4.923	SLOW	2.084	FAST
y[0]	s_rca[0]	4.512	SLOW	1.905	FAST
y[0]	s_rca[1]	4.523	SLOW	1.908	FAST
y[0]	s_rca[2]	4.923	SLOW	2.084	FAST
y[0]	s_rca[3]	4.926	SLOW	2.086	FAST
y[1]	cout_rca	4.923	SLOW	2.084	FAST
y[1]	s_rca[1]	4.515	SLOW	1.907	FAST
y[1]	s_rca[2]	4.923	SLOW	2.084	FAST
y[1]	s_rca[3]	4.926	SLOW	2.086	FAST
y[2]	cout_rca	4.512	SLOW	1.905	FAST
y[2]	s_rca[2]	4.512	SLOW	1.905	FAST
y[2]	s_rca[3]	4.523	SLOW	1.908	FAST
y[3]	cout_rca	4.512	SLOW	1.905	FAST
y[3]	s_rca[3]	4.524	SLOW	1.907	FAST

Comments: rca is adder different from previous adders. Use full adder for that. In rca adding process output of full adder is input of other full adder input. So this have pros. And cos. We can undertand system easily but system using more component

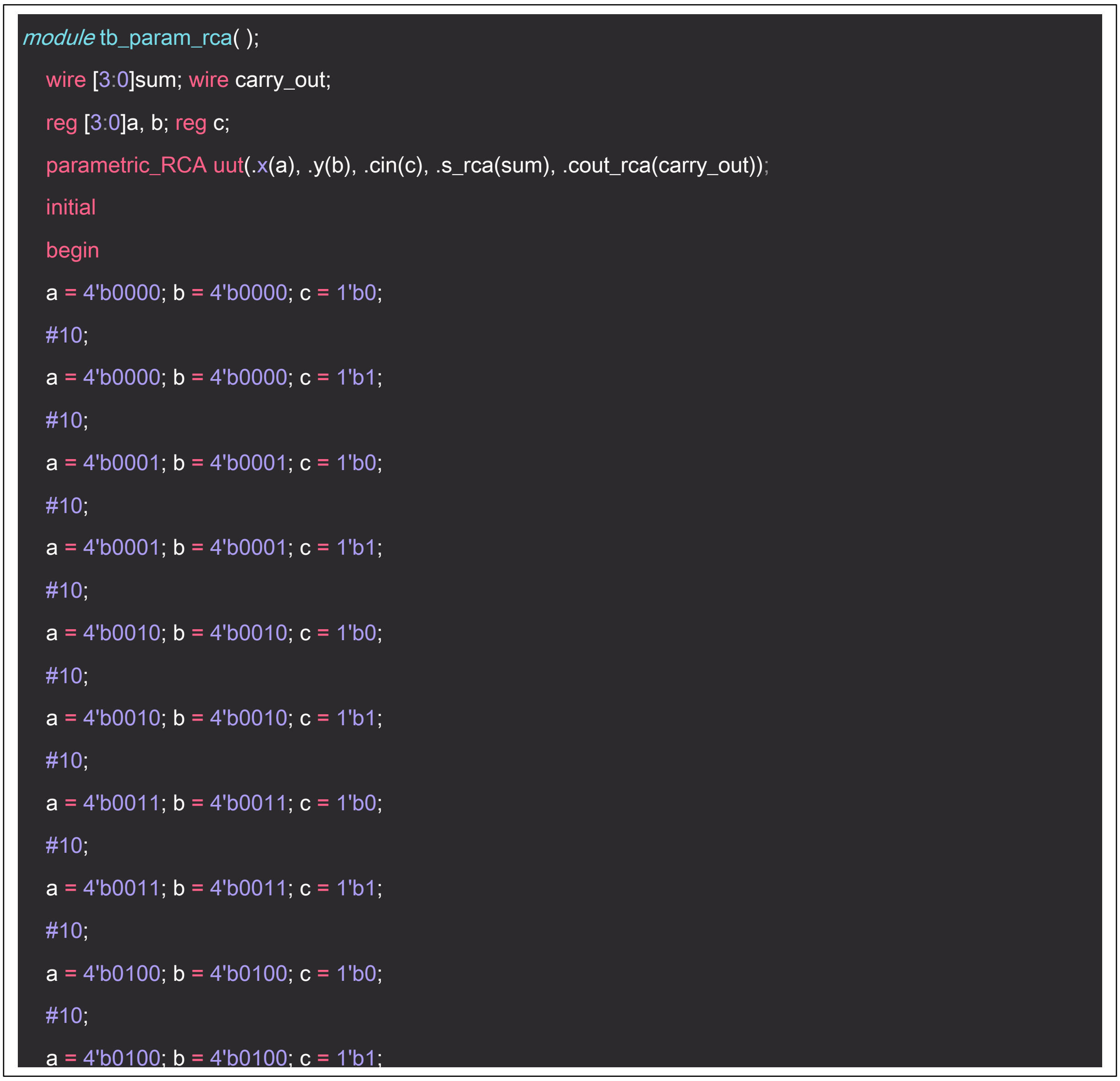
RIPPLE CARY ADDER WITH “GENERATE-FOR”

Verilog code



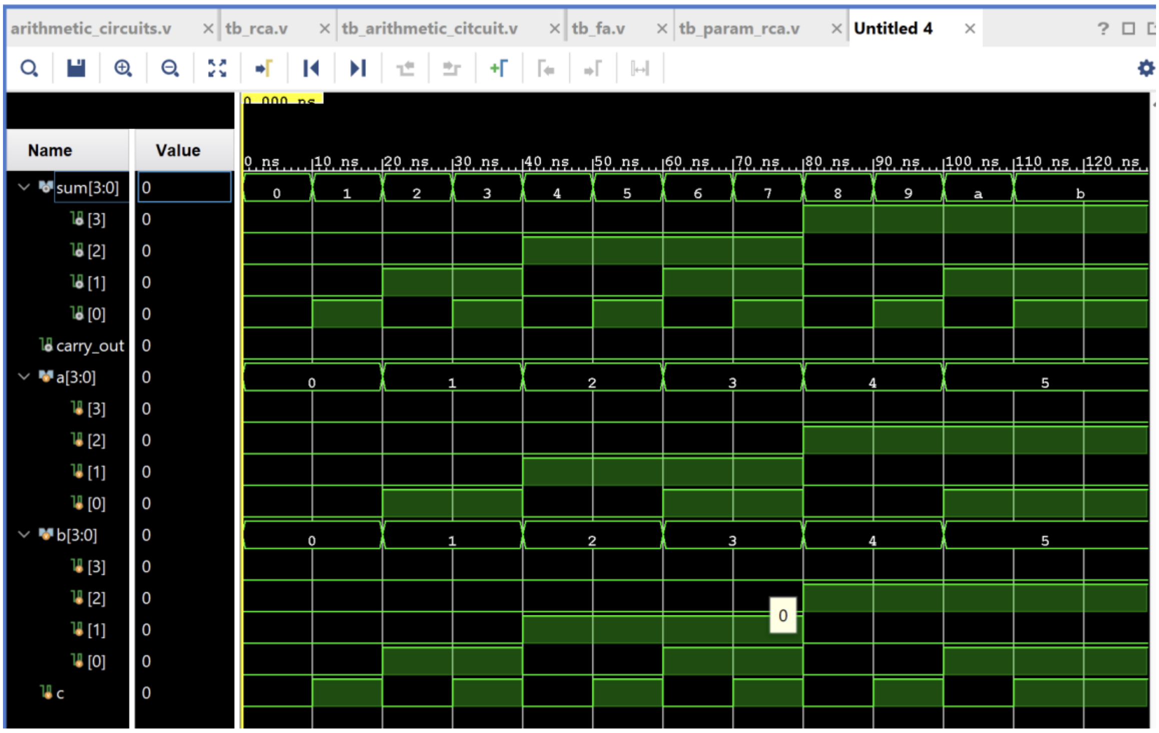
```
module parametric_RCA(cout,s,x,y,cin);
   parameter SIZE=4;
   input [SIZE-1:0] x, y;
   input cin;
   output cout;
   output [SIZE-1:0]s;
   wire [SIZE-1:1] cout_rca_;
   FA FAa (cout_rca_[1],s[0],x[0],y[0],cin);
   genvar i;
   generate for (i=2;i<SIZE;i=i+1)
   begin

   FA FAb (cout_rca_[i],s[i-1],x[i-1],y[i-1],cout_rca_[i-1]);
```

Test code

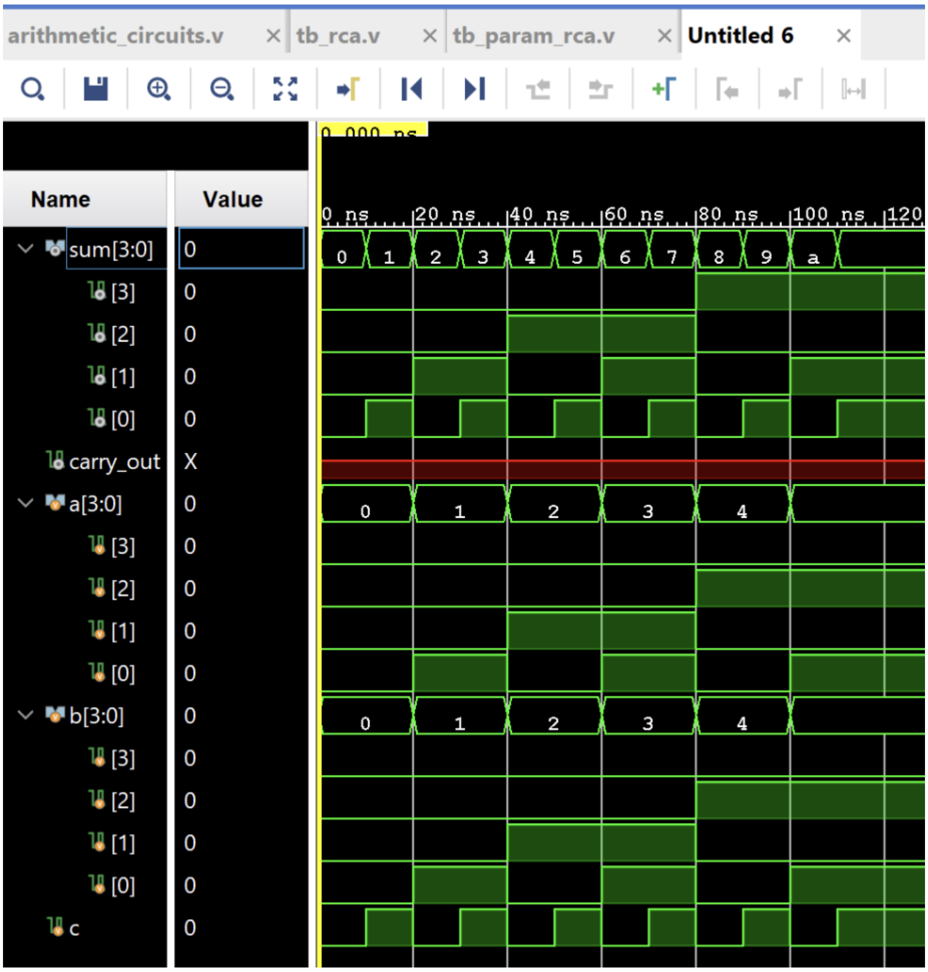


```
module tb_param_rca( );
   wire [3:0]sum; wire carry_out;
   reg [3:0]a, b; reg c;
   parametric_RCA uut(.x(a), .y(b), .cin(c), .s_rca(sum), .cout_rca(carry_out));
   initial
   begin
   a = 4'b0000; b = 4'b0000; c = 1'b0;
   #10;
   a = 4'b0000; b = 4'b0000; c = 1'b1;
   #10;
   a = 4'b0001; b = 4'b0001; c = 1'b0;
   #10;
   a = 4'b0001; b = 4'b0001; c = 1'b1;
   #10;
   a = 4'b0010; b = 4'b0010; c = 1'b0;
   #10;
   a = 4'b0010; b = 4'b0010; c = 1'b1;
   #10;
   a = 4'b0011; b = 4'b0011; c = 1'b0;
   #10;
   a = 4'b0011; b = 4'b0011; c = 1'b1;
   #10;
   a = 4'b0100; b = 4'b0100; c = 1'b0;
   #10;
   a = 4'b0100; b = 4'b0100; c = 1'b1;
```

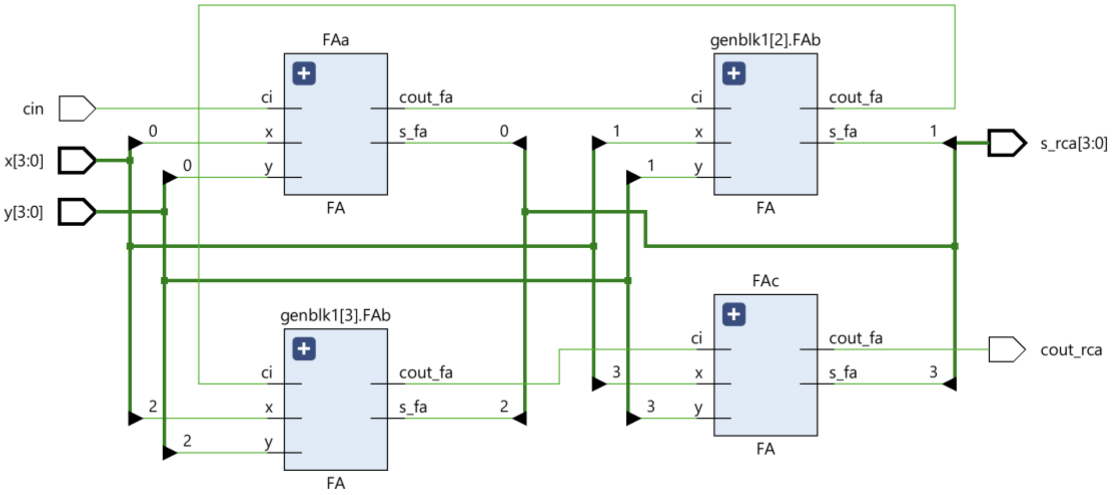
Testbench simulation for 4-bits



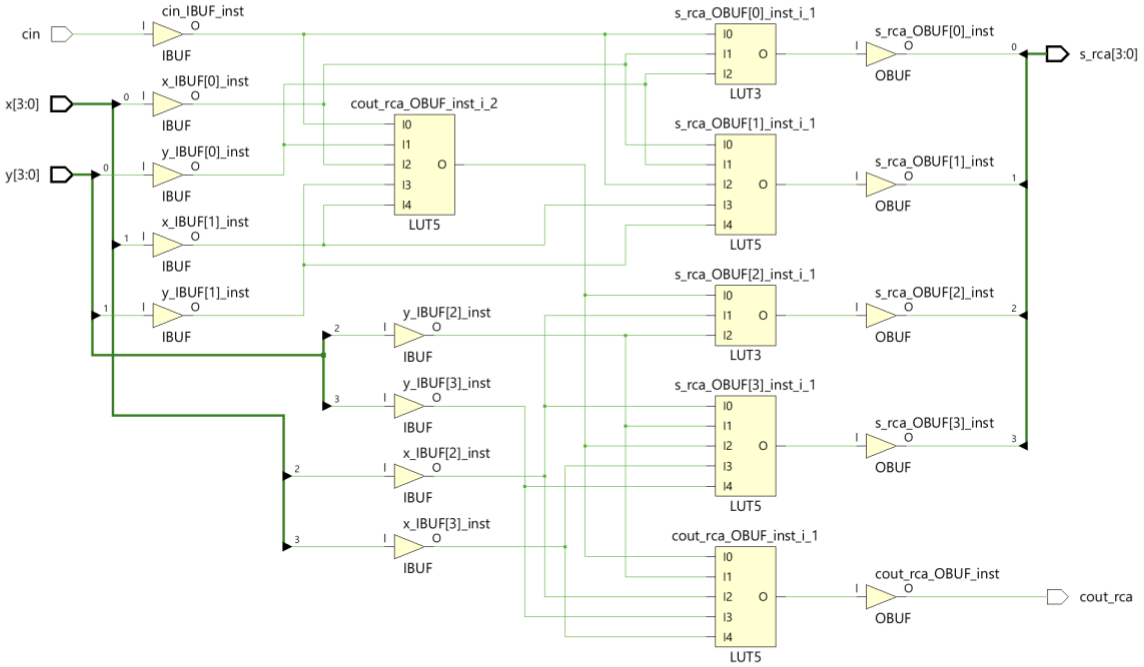
Testbench simulation for 8-bits



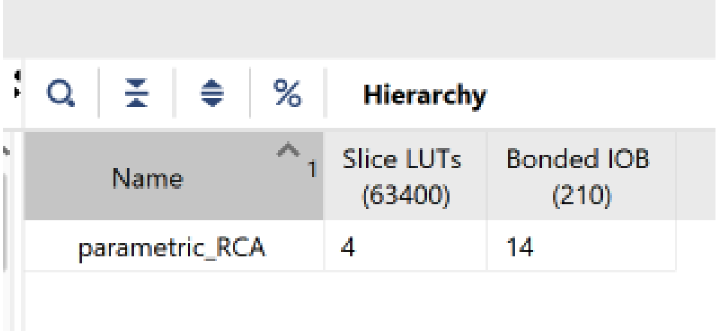
Rtl schema



Technology schema



How many luts



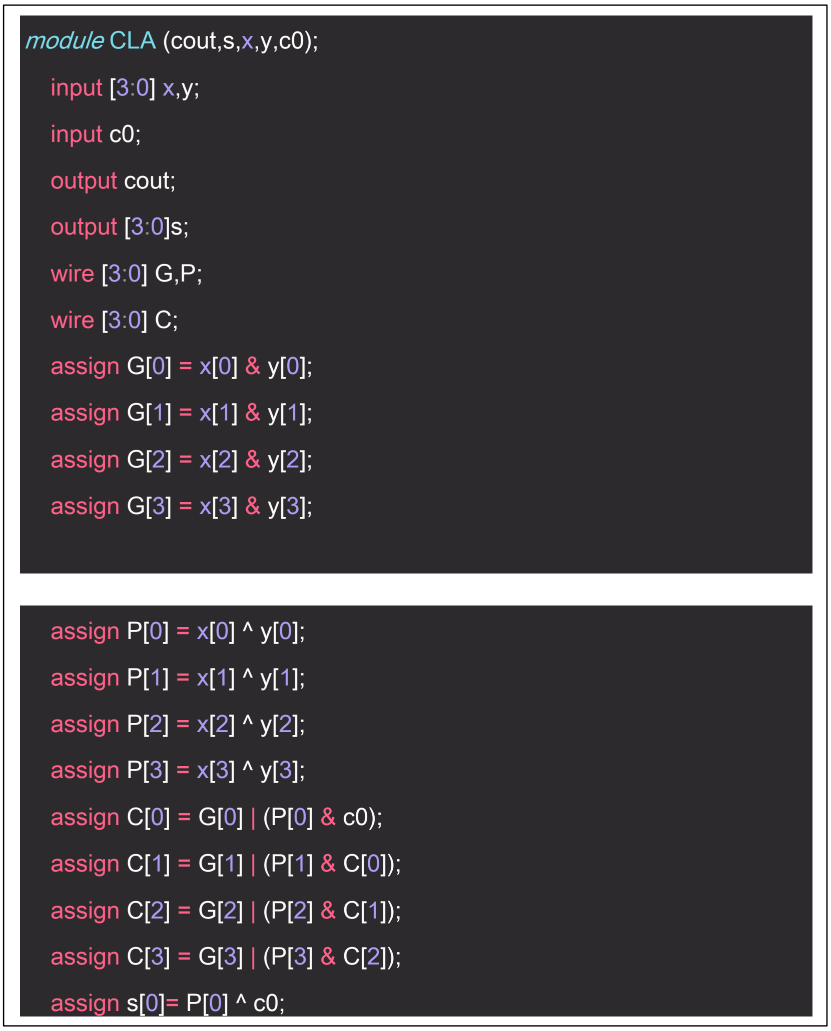
Name	Slice LUTs (63400)	Bonded IOB (210)
parametric_RCA	4	14

4-bit schematic Compare 4-bit rca

these are doing same thing . there are there are not differences. Luts number are same rtl and technology schema are same. But writing code is different.

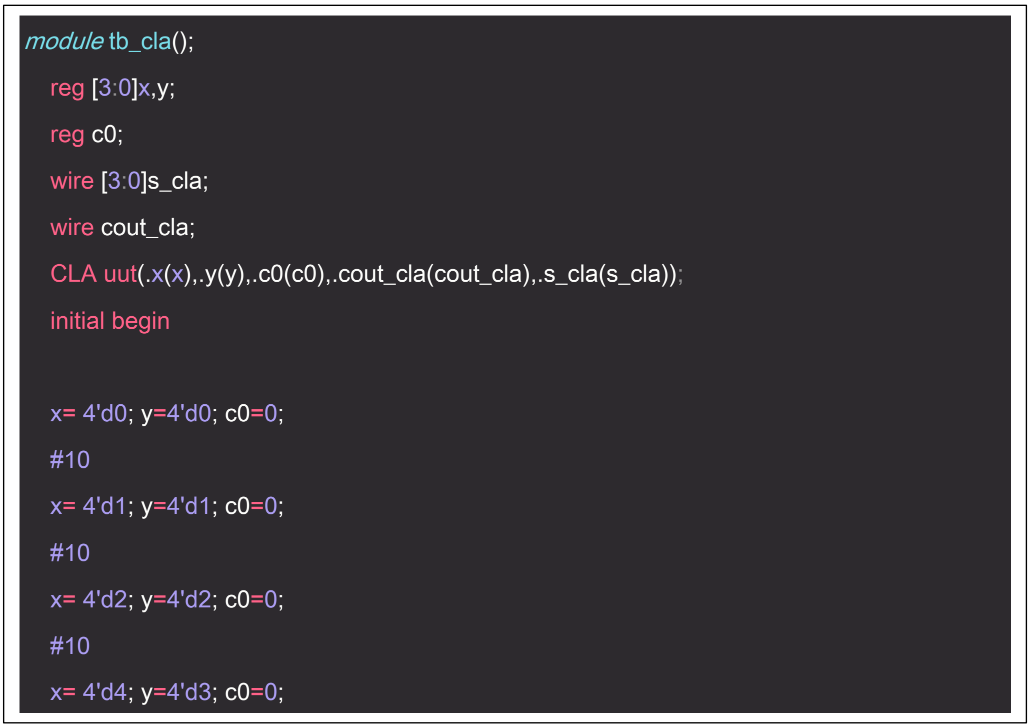
CARRY LOOKAHEAD ADDER

Verilog code



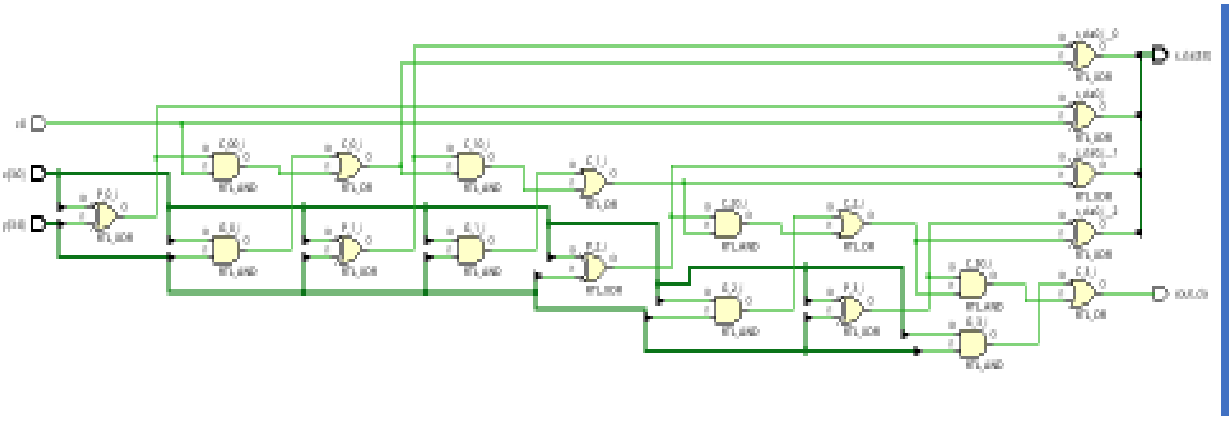
```
module CLA (cout,s,x,y,c0);
   input [3:0] x,y;
   input c0;
   output cout;
   output [3:0]s;
   wire [3:0] G,P;
   wire [3:0] C;
   assign G[0] = x[0] & y[0];
   assign G[1] = x[1] & y[1];
   assign G[2] = x[2] & y[2];
   assign G[3] = x[3] & y[3];

   assign P[0] = x[0] ^ y[0];
   assign P[1] = x[1] ^ y[1];
   assign P[2] = x[2] ^ y[2];
   assign P[3] = x[3] ^ y[3];
   assign C[0] = G[0] | (P[0] & c0);
   assign C[1] = G[1] | (P[1] & C[0]);
   assign C[2] = G[2] | (P[2] & C[1]);
   assign C[3] = G[3] | (P[3] & C[2]);
   assign s[0]= P[0] ^ c0;
```

Test code

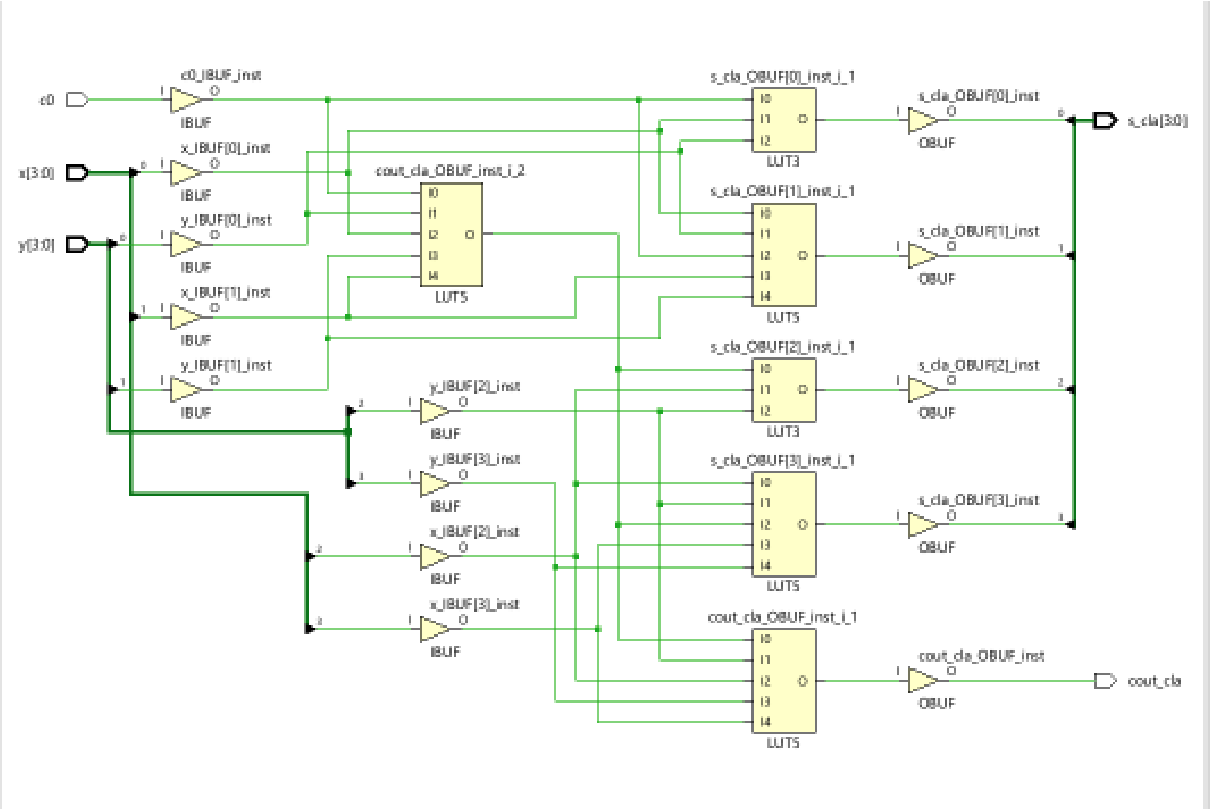


```
module tb_cla();
   reg [3:0]x,y;
   reg c0;
   wire [3:0]s_cla;
   wire cout_cla;
   CLA uut(.x(x),.y(y),.c0(c0),.cout_cla(cout_cla),.s_cla(s_cla));
   initial begin

   x= 4'd0; y=4'd0; c0=0;
   #10
   x= 4'd1; y=4'd1; c0=0;
   #10
   x= 4'd2; y=4'd2; c0=0;
   #10
   x= 4'd4; y=4'd3; c0=0;
```

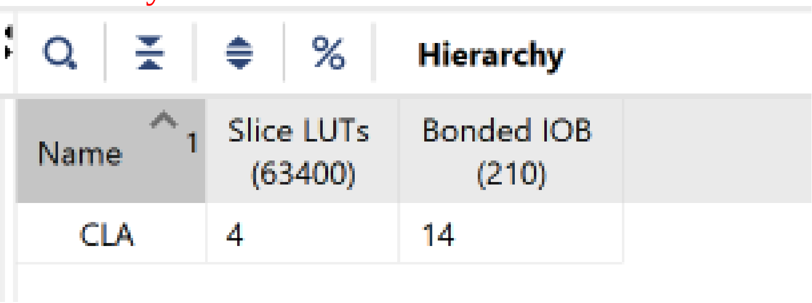
Rtl schema



Technology schema

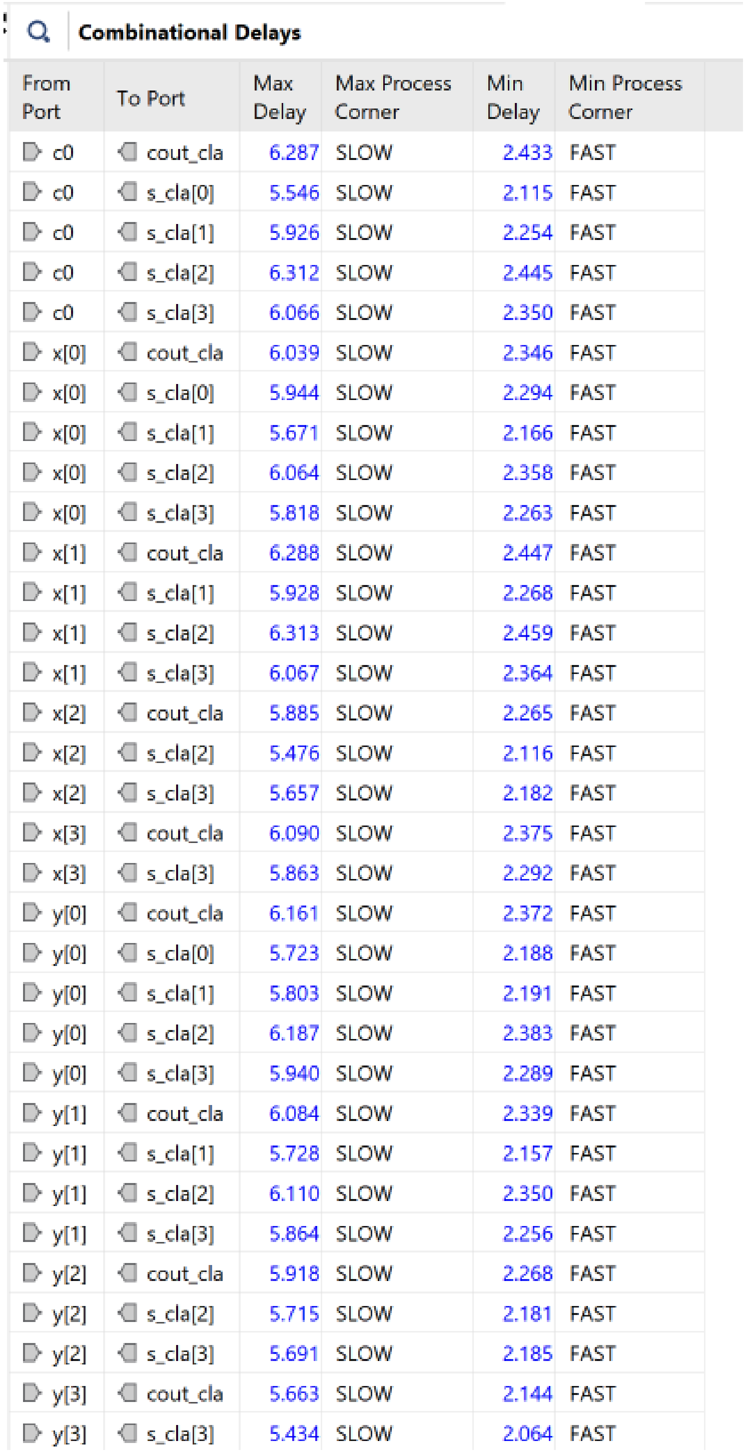


How many luts



Name	Slice LUTs (63400)	Bonded IOB (210)
CLA	4	14

Delay of circuit



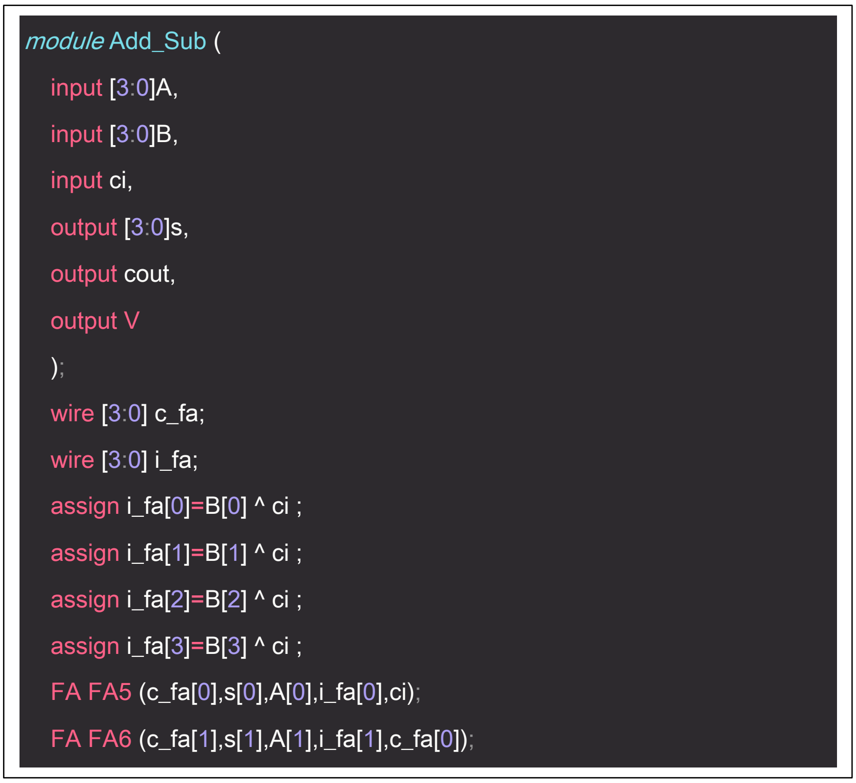
Combinational Delays

From Port	To Port	Max Delay	Max Process Corner	Min Delay	Min Process Corner
c0	cout_cla	6.287	SLOW	2.433	FAST
c0	s_cla[0]	5.546	SLOW	2.115	FAST
c0	s_cla[1]	5.926	SLOW	2.254	FAST
c0	s_cla[2]	6.312	SLOW	2.445	FAST
c0	s_cla[3]	6.066	SLOW	2.350	FAST
x[0]	cout_cla	6.039	SLOW	2.346	FAST
x[0]	s_cla[0]	5.944	SLOW	2.294	FAST
x[0]	s_cla[1]	5.671	SLOW	2.166	FAST
x[0]	s_cla[2]	6.064	SLOW	2.358	FAST
x[0]	s_cla[3]	5.818	SLOW	2.263	FAST
x[1]	cout_cla	6.288	SLOW	2.447	FAST
x[1]	s_cla[1]	5.928	SLOW	2.268	FAST
x[1]	s_cla[2]	6.313	SLOW	2.459	FAST
x[1]	s_cla[3]	6.067	SLOW	2.364	FAST
x[2]	cout_cla	5.885	SLOW	2.265	FAST
x[2]	s_cla[2]	5.476	SLOW	2.116	FAST
x[2]	s_cla[3]	5.657	SLOW	2.182	FAST
x[3]	cout_cla	6.090	SLOW	2.375	FAST
x[3]	s_cla[3]	5.863	SLOW	2.292	FAST
y[0]	cout_cla	6.161	SLOW	2.372	FAST
y[0]	s_cla[0]	5.723	SLOW	2.188	FAST
y[0]	s_cla[1]	5.803	SLOW	2.191	FAST
y[0]	s_cla[2]	6.187	SLOW	2.383	FAST
y[0]	s_cla[3]	5.940	SLOW	2.289	FAST
y[1]	cout_cla	6.084	SLOW	2.339	FAST
y[1]	s_cla[1]	5.728	SLOW	2.157	FAST
y[1]	s_cla[2]	6.110	SLOW	2.350	FAST
y[1]	s_cla[3]	5.864	SLOW	2.256	FAST
y[2]	cout_cla	5.918	SLOW	2.268	FAST
y[2]	s_cla[2]	5.715	SLOW	2.181	FAST
y[2]	s_cla[3]	5.691	SLOW	2.185	FAST
y[3]	cout_cla	5.663	SLOW	2.144	FAST
y[3]	s_cla[3]	5.434	SLOW	2.064	FAST

Comments: cla is 4 bit adder. Doing same thing previous. But its working fast and not using more components. But in bigger system that couse decreasing understandable. İt will more confusing

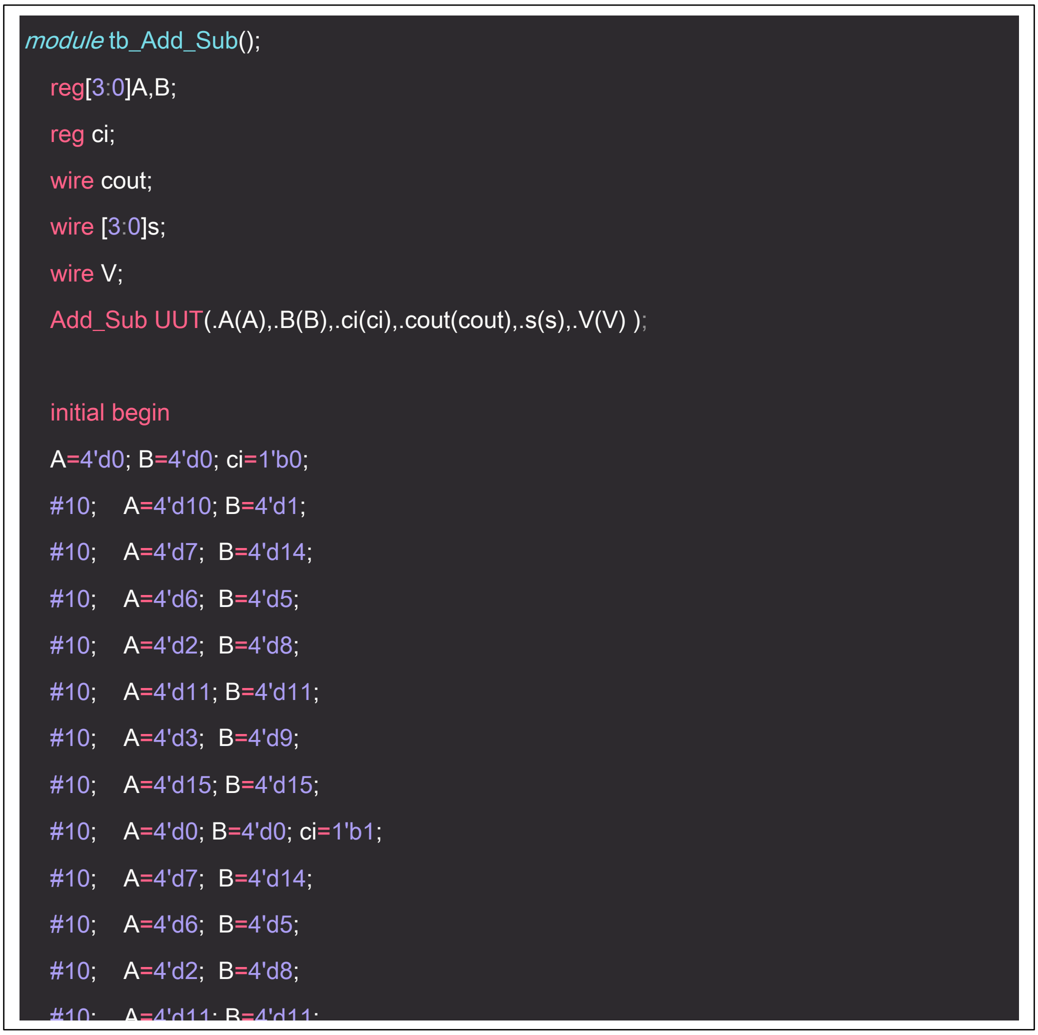
ADDER-SUBSRACTOR CIRCUIT

Verilog code



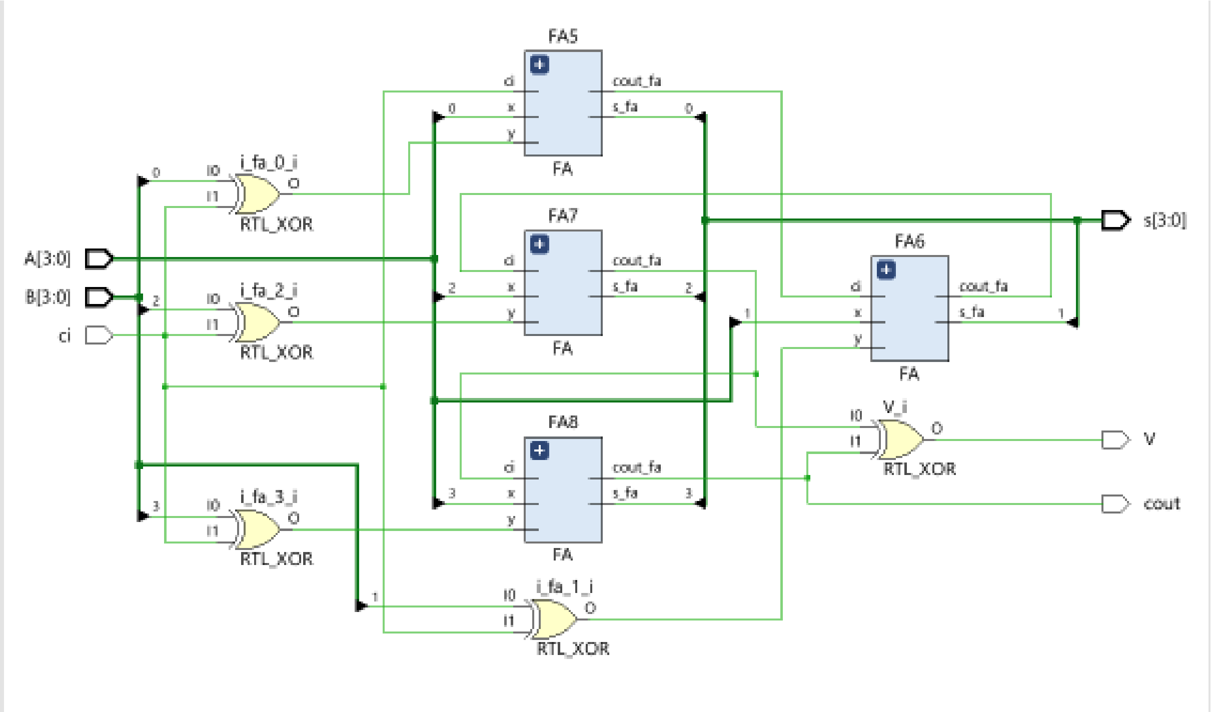
```
module Add_Sub (
    input [3:0]A,
    input [3:0]B,
    input ci,
    output [3:0]s,
    output cout,
    output V
    );
    wire [3:0] c_fa;
    wire [3:0] i_fa;
    assign i_fa[0]=B[0] ^ ci ;
    assign i_fa[1]=B[1] ^ ci ;
    assign i_fa[2]=B[2] ^ ci ;
    assign i_fa[3]=B[3] ^ ci ;
    FA FA5 (c_fa[0],s[0],A[0],i_fa[0],ci);
    FA FA6 (c_fa[1],s[1],A[1],i_fa[1],c_fa[0]);
```

Test code

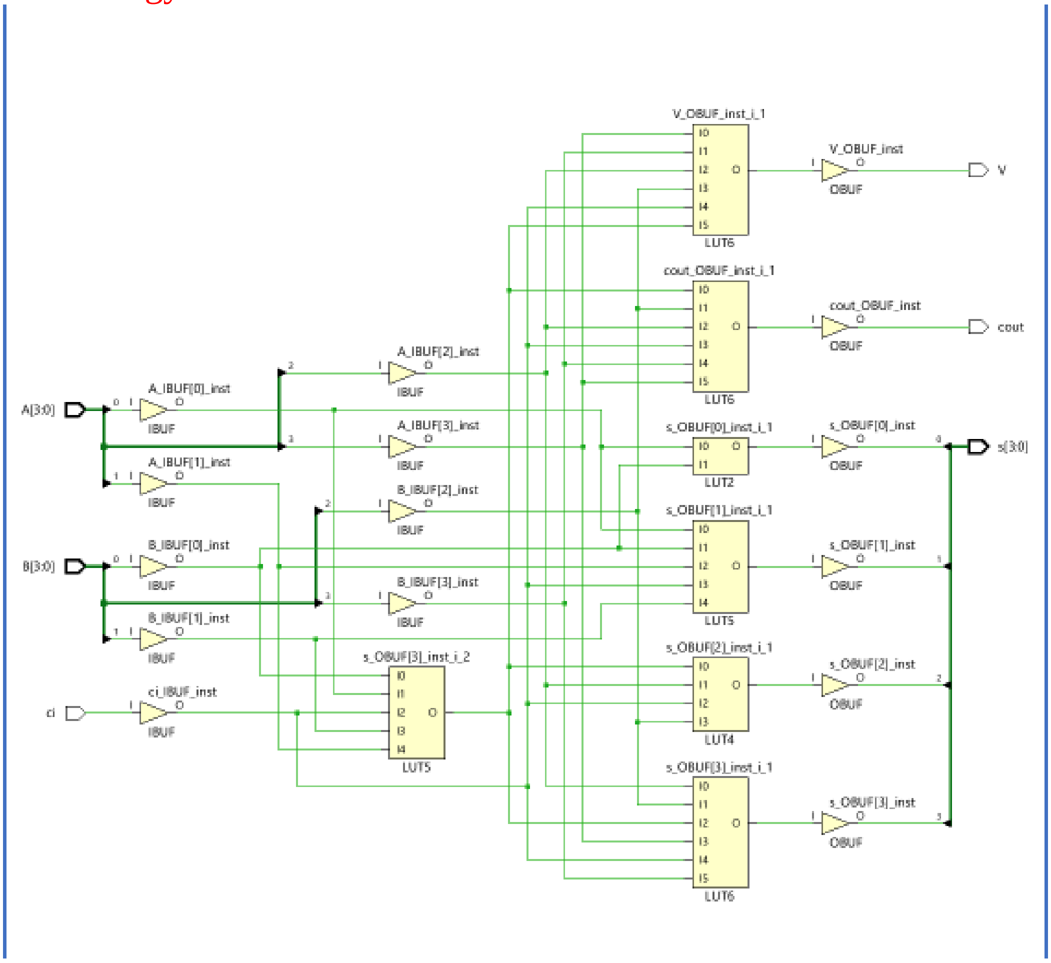


```
module tb_Add_Sub();
    reg[3:0]A,B;
    reg ci;
    wire cout;
    wire [3:0]s;
    wire V;
    Add_Sub UUT(.A(A),.B(B),.ci(ci),.cout(cout),.s(s),.V(V) );

    initial begin
    A=4'd0; B=4'd0; ci=1'b0;
    #10;    A=4'd10; B=4'd1;
    #10;    A=4'd7;  B=4'd14;
    #10;    A=4'd6;  B=4'd5;
    #10;    A=4'd2;  B=4'd8;
    #10;    A=4'd11; B=4'd11;
    #10;    A=4'd3;  B=4'd9;
    #10;    A=4'd15; B=4'd15;
    #10;    A=4'd0; B=4'd0; ci=1'b1;
    #10;    A=4'd7;  B=4'd14;
    #10;    A=4'd6;  B=4'd5;
    #10;    A=4'd2;  B=4'd8;
    #10;    A=4'd11; B=4'd11;
```

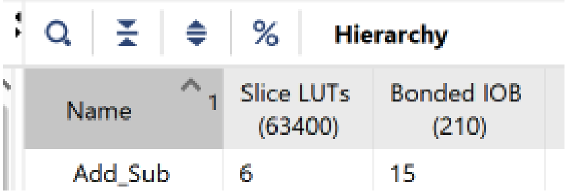
Rtl schema



Technology schema

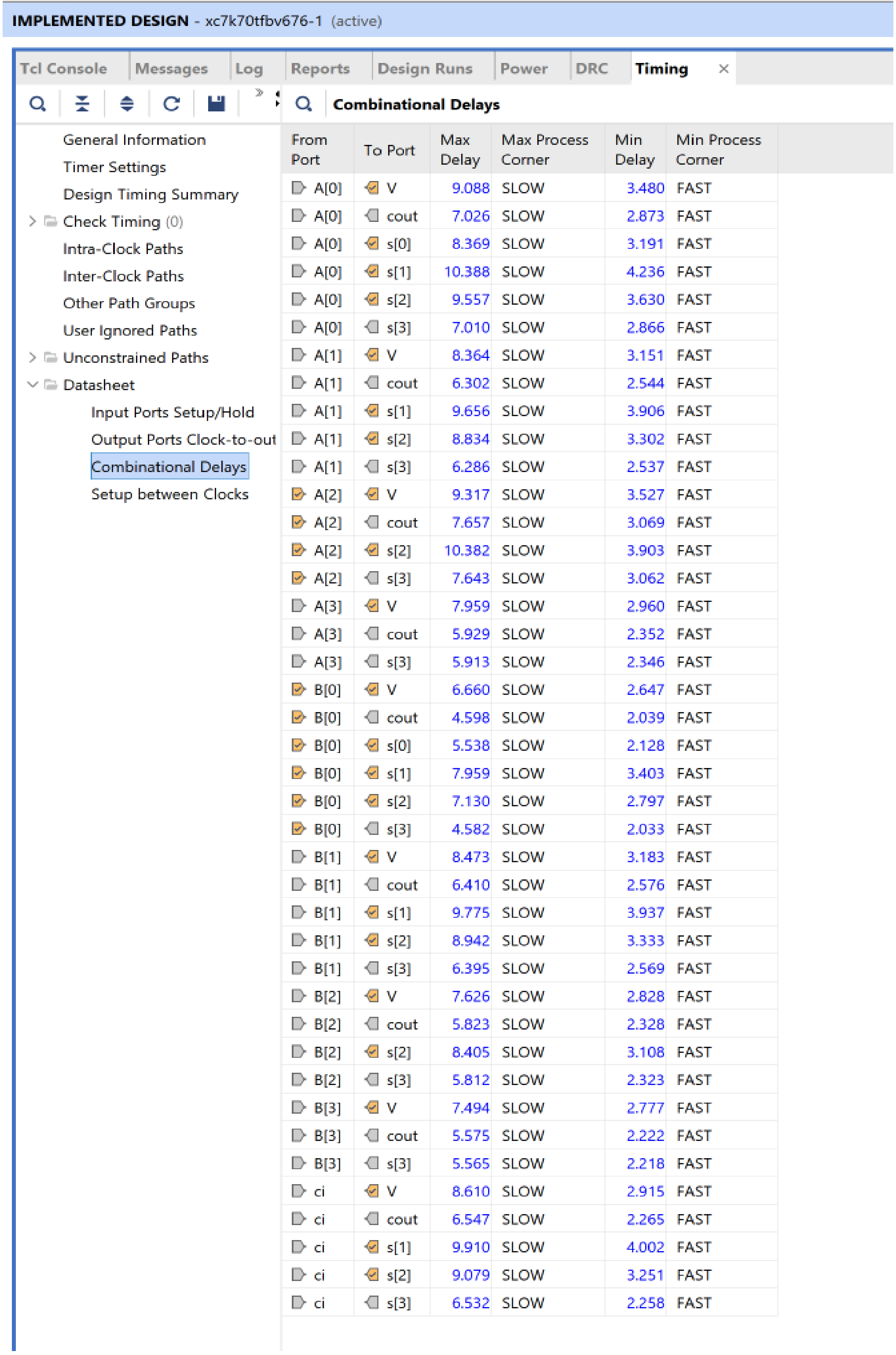


How many luts.



Name	Slice LUTs (63400)	Bonded IOB (210)
Add_Sub	6	15

Delay of circuit



From Port	To Port	Max Delay	Max Process Corner	Min Delay	Min Process Corner
A[0]	V	9.088	SLOW	3.480	FAST
A[0]	cout	7.026	SLOW	2.873	FAST
A[0]	s[0]	8.369	SLOW	3.191	FAST
A[0]	s[1]	10.388	SLOW	4.236	FAST
A[0]	s[2]	9.557	SLOW	3.630	FAST
A[0]	s[3]	7.010	SLOW	2.866	FAST
A[1]	V	8.364	SLOW	3.151	FAST
A[1]	cout	6.302	SLOW	2.544	FAST
A[1]	s[1]	9.656	SLOW	3.906	FAST
A[1]	s[2]	8.834	SLOW	3.302	FAST
A[1]	s[3]	6.286	SLOW	2.537	FAST
A[2]	V	9.317	SLOW	3.527	FAST
A[2]	cout	7.657	SLOW	3.069	FAST
A[2]	s[2]	10.382	SLOW	3.903	FAST
A[2]	s[3]	7.643	SLOW	3.062	FAST
A[3]	V	7.959	SLOW	2.960	FAST
A[3]	cout	5.929	SLOW	2.352	FAST
A[3]	s[3]	5.913	SLOW	2.346	FAST
B[0]	V	6.660	SLOW	2.647	FAST
B[0]	cout	4.598	SLOW	2.039	FAST
B[0]	s[0]	5.538	SLOW	2.128	FAST
B[0]	s[1]	7.959	SLOW	3.403	FAST
B[0]	s[2]	7.130	SLOW	2.797	FAST
B[0]	s[3]	4.582	SLOW	2.033	FAST
B[1]	V	8.473	SLOW	3.183	FAST
B[1]	cout	6.410	SLOW	2.576	FAST
B[1]	s[1]	9.775	SLOW	3.937	FAST
B[1]	s[2]	8.942	SLOW	3.333	FAST
B[1]	s[3]	6.395	SLOW	2.569	FAST
B[2]	V	7.626	SLOW	2.828	FAST
B[2]	cout	5.823	SLOW	2.328	FAST
B[2]	s[2]	8.405	SLOW	3.108	FAST
B[2]	s[3]	5.812	SLOW	2.323	FAST
B[3]	V	7.494	SLOW	2.777	FAST
B[3]	cout	5.575	SLOW	2.222	FAST
B[3]	s[3]	5.565	SLOW	2.218	FAST
ci	V	8.610	SLOW	2.915	FAST
ci	cout	6.547	SLOW	2.265	FAST
ci	s[1]	9.910	SLOW	4.002	FAST
ci	s[2]	9.079	SLOW	3.251	FAST
ci	s[3]	6.532	SLOW	2.258	FAST

Comments

Adder and substactor device do adder ci of doing substactor when ci input is on. İn output there are sum count and V. V is overflaow sign.