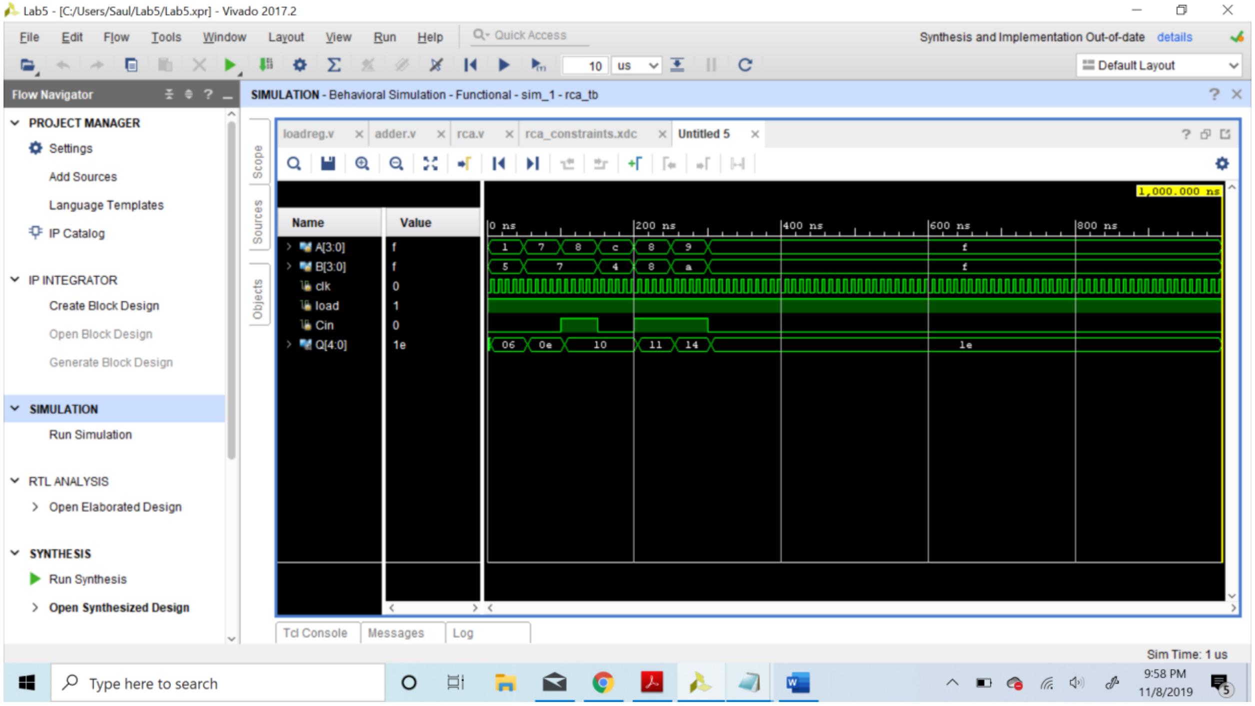
A[3:0]	B[3:0]	Cin	S	Cout
0001	0101	0	0110	0
0111	0111	0	1110	0
1000	0111	1	0000	1
1100	0100	0	0000	1
1000	1000	1	0001	1
1001	1010	1	0100	1
1111	1111	0	1110	1



Name	Value
A[3:0]	f
B[3:0]	f
clk	0
load	1
Cin	0
Q[4:0]	1e

LOAD

```
`timescale 1ns / 1ps



module loadreg(
    input clk,
    input load,
    input [9:0] D,
    output reg [9:0] Q
    );

    initial Q = 0;
    always @(posedge clk) begin
        if(load)Q <= D;
    end


endmodule
```

ADDER

```
`timescale 1ns / 1ps



module adder(
    input A, B, Cin,
    output S, Cout
    );

    assign Cout = (B&Cin) | (A&Cin) | (A&B);
    assign S = A ^ B ^ Cin;

endmodule
```

RCA

```
`timescale 1ns / 1ps

module rca(
    input clk,
    input load,
    input [3:0] A, B,
    input Cin,
    output [4:0] Q
);

    wire [2:0] cin;
    wire [4:0] D;

    //loadreg c5 (.clk(clk), .load(load), .D(D), .Q(Q));

    adder c1 (.A(A[0]), .B(B[0]), .Cin(Cin), .S(D[0]), .Cout(cin[0]));

    adder c2 (.A(A[1]), .B(B[1]), .Cin(cin[0]), .S(D[1]), .Cout(cin[1]));

    adder c3 (.A(A[2]), .B(B[2]), .Cin(cin[1]), .S(D[2]), .Cout(cin[2]));

    adder c4 (.A(A[3]), .B(B[3]), .Cin(cin[2]), .S(D[3]), .Cout(D[4]));


endmodule
```

Constraints

```
## Clock signal
set_property PACKAGE_PIN W5 [get_ports clk]
        set_property IOSTANDARD LVCMOS33 [get_ports clk]
        create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]
```

```
set_property PACKAGE_PIN U18 [get_ports load]
   set_property IOSTANDARD LVCMOS33 [get_ports load]

## LEDs
set_property PACKAGE_PIN U16 [get_ports {Q[0]}]
          set_property IOSTANDARD LVCMOS33 [get_ports {Q[0]}]
set_property PACKAGE_PIN E19 [get_ports {Q[1]}]
          set_property IOSTANDARD LVCMOS33 [get_ports {Q[1]}]
set_property PACKAGE_PIN U19 [get_ports {Q[2]}]
          set_property IOSTANDARD LVCMOS33 [get_ports {Q[2]}]
set_property PACKAGE_PIN V19 [get_ports {Q[3]}]
          set_property IOSTANDARD LVCMOS33 [get_ports {Q[3]}]
set_property PACKAGE_PIN W18 [get_ports {Q[4]}]
          set_property IOSTANDARD LVCMOS33 [get_ports {Q[4]}]

## Switches
   set_property PACKAGE_PIN V17 [get_ports {A[0]}]
      set_property IOSTANDARD LVCMOS33 [get_ports {A[0]}]
   set_property PACKAGE_PIN V16 [get_ports {A[1]}]
      set_property IOSTANDARD LVCMOS33 [get_ports {A[1]}]
   set_property PACKAGE_PIN W16 [get_ports {A[2]}]
      set_property IOSTANDARD LVCMOS33 [get_ports {A[2]}]
   set_property PACKAGE_PIN W17 [get_ports {A[3]}]
      set_property IOSTANDARD LVCMOS33 [get_ports {A[3]}]
   set_property PACKAGE_PIN W15 [get_ports {B[0]}]
      set_property IOSTANDARD LVCMOS33 [get_ports {B[0]}]
   set_property PACKAGE_PIN V15 [get_ports {B[1]}]
      set_property IOSTANDARD LVCMOS33 [get_ports {B[1]}]
```

```
    set_property PACKAGE_PIN W14 [get_ports {B[2]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {B[2]}]
    set_property PACKAGE_PIN W13 [get_ports {B[3]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {B[3]}]
    set_property PACKAGE_PIN V2 [get_ports {Cin}]
        set_property IOSTANDARD LVCMOS33 [get_ports {Cin}]
```

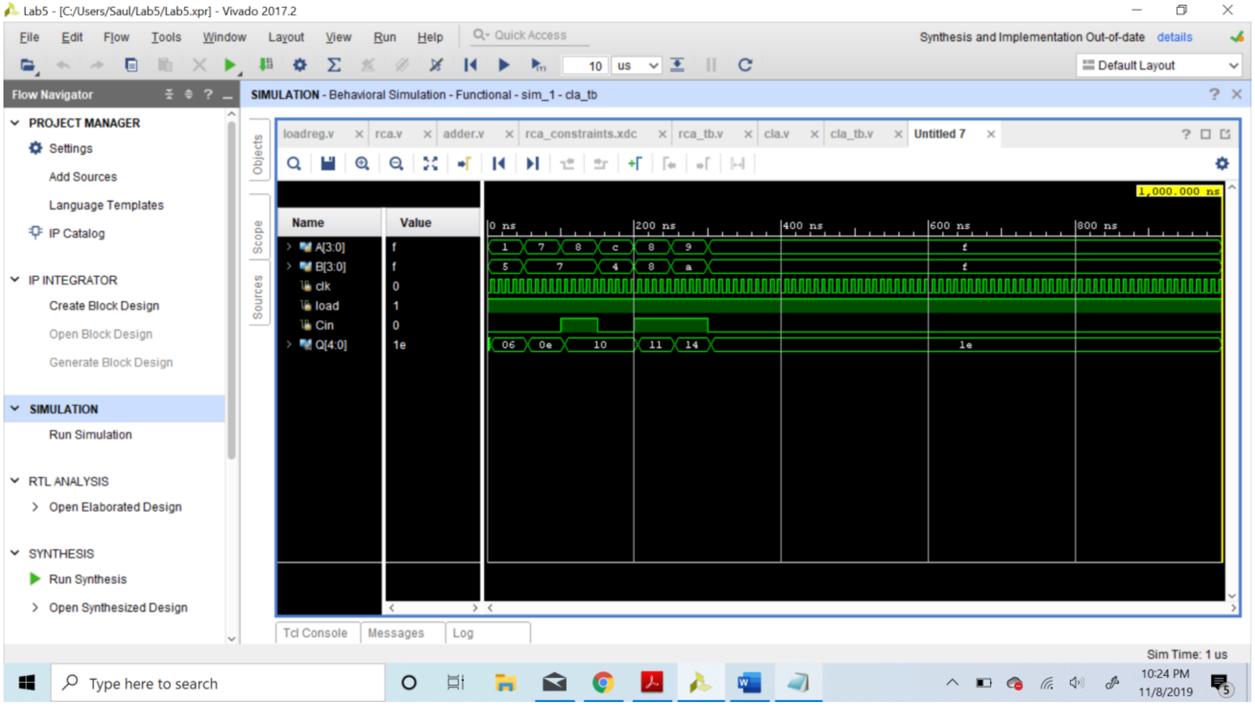
TESTBENCH

```
`timescale 1ns / 1ps

module rca_tb;
    reg [3:0] A;
    reg [3:0] B;
    reg clk;
    reg load;
    reg Cin;
    wire [4:0] Q;



    rca uut(
    .clk(clk),
    .load(load),
    .A(A),
    .B(B),
    .Cin(Cin),
    .Q(Q)
    );
```

```
initial
begin
clk = 0;
load = 1;

A = 4'b0001;
B = 4'b0101;
Cin = 1'b0;
load = 1;

#50;

A = 4'b0111;
B = 4'b0111;
Cin = 1'b0;
load = 1;

#50;

A = 4'b1000;
B = 4'b0111;
Cin = 1'b1;
load = 1;


#50;
A = 4'b1100;
B = 4'b0100;
Cin = 1'b0;
```

```
    load = 1;


    #50;

    A = 4'b1000;

    B = 4'b1000;

    Cin = 1'b1;

    load = 1;



    #50;

    A = 4'b1001;

    B = 4'b1010;

    Cin = 1'b1;

    load = 1;


    #50;

    A = 4'b1111;

    B = 4'b1111;

    Cin = 1'b0;

    load = 1;


    end

    always

    #5 clk = ~clk;


Endmodule
```

PART 2



C[0] = Cin

C[1] = G[0] + (P[0]*C[0])

C[2] = G[1] + (P[1]*C[1])

C[3] = G[2] + (P[2]*C[2])

C[4] = G[3] + (P[3]*C[3])

S[0] = P[0]*C[0]

S[1] = P[1]*(G[0] + (P[0]*C[0]))

S[2] = P[2]*(G[1] | (P[1]*G[0]) + (P[1]*P[0]*C[0]))

S[3] = P[3]*(G[2] | (P[2]*G[1]) + (P[2]*P[1]*G[0]) + (P[2]*P[1]*P[0]*C[0]))

S[4] = C[4]

CLA

```
`timescale 1ns / 1ps

module cla(
   input clk,
   input load,
   input [3:0] A,
   input [3:0] B,
   input Cin,
   output [4:0] Q

   );

   wire [3:0] G, P, S;
   wire [1:0] C;

   assign C[0] = Cin;
   //loadreg r0 ( .clk(clk), .load(load), .D({C[1], S}), .Q(Q));



   assign G[0] = A[0] & B[0]; //Generate
   assign G[1] = A[1] & B[1]; //Generate
   assign G[2] = A[2] & B[2]; //Generate
   assign G[3] = A[3] & B[3]; //Generate

   assign P[0] = A[0] ^ B[0];
   assign P[1] = A[1] ^ B[1];
   assign P[2] = A[2] ^ B[2];
   assign P[3] = A[3] ^ B[3];
```

```
    //assign P = A ^ B; //Propagate

    assign S[0] = P[0] ^ C[0];

    assign S[1] = P[1] ^ (G[0] | P[0]&C[0]);

    assign S[2] = P[2] ^ (G[1] | P[1]&G[0] | P[1]&P[0]&C[0]);

    assign S[3] = P[3] ^ (G[2] | P[2]&G[1] | P[2]&P[1]&G[0] | P[2]&P[1]&P[0]&C[0]);


    assign C[1] = G[3] | P[3]&G[2] | P[3]&P[2]&G[1] | P[3]&P[2]&P[1]&G[0] | P[3]&P[2]&P[1]&P[0]&C[0];


endmodule
```

TESTBENCH

```
`timescale 1ns / 1ps


module cla_tb;

    reg [3:0] A;

    reg [3:0] B;

    reg clk;

    reg load;

    reg Cin;

    wire [4:0] Q;



    cla uut(

    .clk(clk),

    .load(load),

    .A(A),

    .B(B),

    .Cin(Cin),
    .Q(Q)
    );

    initial
    begin
    clk = 0;
    load = 1;

    A = 4'b0001;
    B = 4'b0101;
    Cin = 1'b0;
    load = 1;

    #50;

    A = 4'b0111;
    B = 4'b0111;
    Cin = 1'b0;
    load = 1;

    #50;

    A = 4'b1000;
    B = 4'b0111;
    Cin = 1'b1;
    load = 1;


    #50;
```

```
    A = 4'b1100;
    B = 4'b0100;
    Cin = 1'b0;
    load = 1;

    #50;
    A = 4'b1000;
    B = 4'b1000;
    Cin = 1'b1;
    load = 1;


    #50;
    A = 4'b1001;
    B = 4'b1010;
    Cin = 1'b1;
    load = 1;

    #50;
    A = 4'b1111;
    B = 4'b1111;
    Cin = 1'b0;
    load = 1;

    end
    always
    #5 clk = ~clk;

endmodule
```

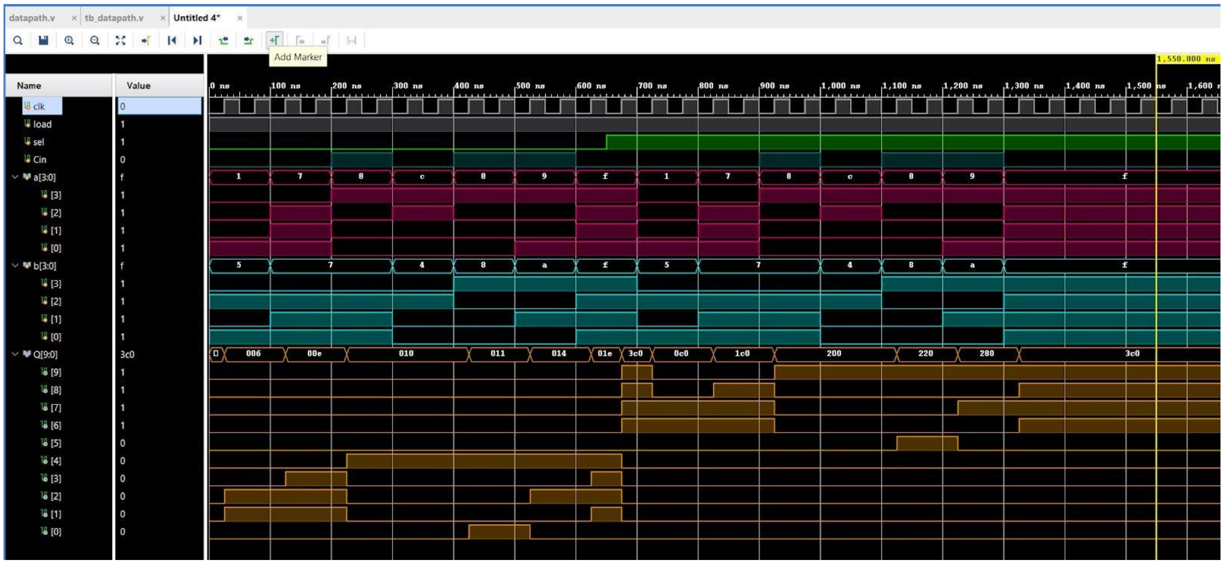
DATAPATH

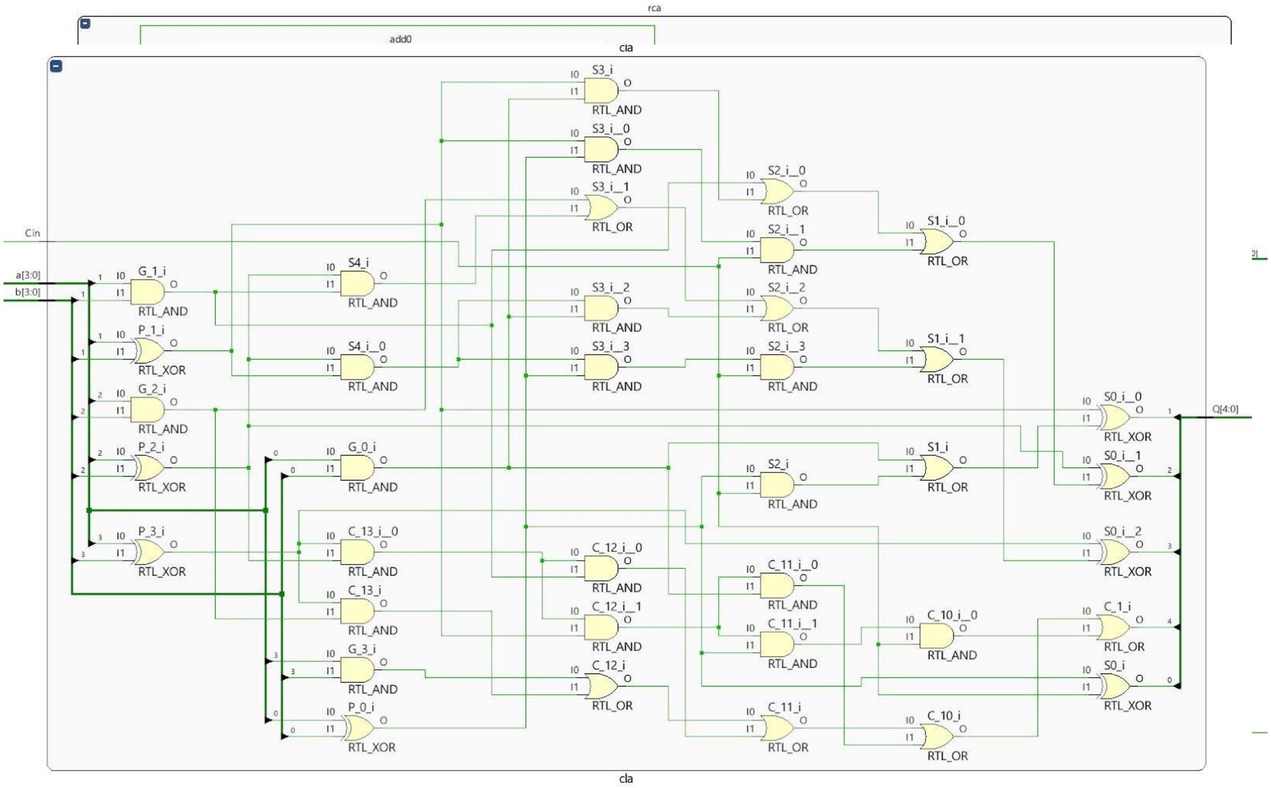
```
`timescale 1ns / 1ps


module datapath(
    input clk, load,
    input [3:0] A, B,
    input Cin,
    input sel,
    output [9:0] Q
    );

    wire [4:0] rcaOut;
    wire [4:0] claOut;
    reg [9:0] S;

    rca rca(.A(A), .B(B), .Cin(Cin), .Q(rcaOut));

    cla cla(.A(A), .B(B), .Cin(Cin), .Q(claOut));

    loadreg r0(.clk(clk), .load(load), .D(S), .Q(Q));

    always @(*) begin

        case (sel)
            1'b0 : S = {5'b00000, rcaOut};
            1'b1 : S = {claOut, 5'b00000};
        endcase
    end


endmodule
```

DATAPATH TB

```
`timescale 1ns / 1ps

module datapath_tb;
reg clk, load;
reg [3:0] A,B;
reg Cin;
reg sel;
wire [9:0] Q;

datapath uut(
.clk(clk),
.load(load),
.A(A),
.B(B),
.Cin(Cin),
.sel(sel),
.Q(Q)
);

initial begin

$display ("Begin testing - RCA");
```

```
clk = 0; load = 1; sel = 0;

Cin = 0; A = 4'b0001; B = 4'b0101; #50


#50 $display ("Q = 01110");
Cin = 0; A = 4'b0111; B = 4'b0111; #50



#50 $display ("Q = 10000");
Cin = 1; A = 4'b1000; B = 4'b0111; #50

Cin = 0; A = 4'b1100; B = 4'b0100; #50

Cin = 1; A = 4'b1000; B = 4'b1000; #50

Cin = 1; A = 4'b1001; B = 4'b1010; #50

Cin = 0; A = 4'b1111; B = 4'b1111; #50

$display ("\n\n\nBegin testing - CLA");
sel = 1;
#50 $display ("Q = 00110");
Cin = 0; A = 4'b0001; B = 4'b0101; #50

Cin = 0; A = 4'b0111; B = 4'b0111; #50

Cin = 1; A = 4'b1000; B = 4'b0111; #50
Cin = 0; A = 4'b1100; B = 4'b0100; #50

Cin = 1; A = 4'b1000; B = 4'b1000; #50

Cin = 1; A = 4'b1001; B = 4'b1010; #50

Cin = 0; A = 4'b1111; B = 4'b1111; #50

end
always
#25 clk = ~clk;
endmodule
```





	AREA (area units	DELAY (ns)
RCA	104	40
CLA	128	29