Lab 5 Report

Name: **Donald Maze-England**

UT EID: **dsm2588**

Section: **15320**

Checklist:

Part 1 –

i. Design file (.v) for the Ripple Carry Adder
ii. Test-bench
iii. Complete Table 1 from the simulation

A[3:0]	B[3:0]	Cin	Sum[3:0]	Cout
0001	0101	0	?	?
0111	0111	0	?	?
1000	0111	1	?	?
1100	0100	0	?	?
1000	1000	1	?	?
1001	1010	1	?	?
1111	1111	0	?	?

Table 1. Testcases for Ripple Carry Adder Verification

iv. Constraints File (Just the uncommented portion)
v. Simulation waveform for the above test-cases

Part 2 –

vi. All the equations for C_i's and S_i's
vii. Design files (.v) for the Carry Lookahead Adder and Register Logic
viii. Test-bench
ix. Complete Table 2 from the simulation

A[3:0]	B[3:0]	Cin	Sum[3:0]	Cout
0000	0101	0	?	?
0101	0111	0	?	?
1000	0111	1	?	?
1001	0100	0	?	?
1000	1000	1	?	?
1101	1010	1	?	?
1110	1111	0	?	?

Table 2. Testcases for Carry Lookahead Adder Verification

x. Constraints File (Just the uncommented portion)
xi. Simulation waveform for the above test-cases

Part 3 –

xii. Screenshots of the gate-level schematics for both the adder techniques
xiii. Delay and area for both the adder techniques showing all the work
xiv. Brief conclusion regarding the pros and cons of each of the techniques

Note *–> The Verilog codes and the uncommented portions of the constraint files should be copied in your lab report and the* ***actual Verilog (.v), Constraint (.xdc) files and Bitstream (.bit) files*** *need to be zipped and submitted as well on Canvas. You are not allowed to change your codes after final submission as the TAs may download the submitted codes or bitstream files from Canvas during checkouts. For the truth Table, K-maps minimizations and algebraic expressions, you are free to draw them on paper and then put the pictures in your lab report, but please make sure it is legible for the TAs to grade it properly.*

Part 1:

RCA Verilog Code:

```
`timescale 1ns / 1ps



module RCA4bit(
    input clk,
    input enable,
    input [3:0] A,B,
    input Cin,
    output [4:0] q
    );

    wire [4:0] regIn;
    wire Cout1, Cout2, Cout3;
    fullAdder a1 (A[0], B[0], Cin, Cout1, regIn[0]);
    fullAdder a2 (A[1], B[1], Cout1, Cout2, regIn[1]);
    fullAdder a3 (A[2], B[2], Cout2, Cout3, regIn[2]);
    fullAdder a4 (A[3], B[3], Cout3, regIn[4], regIn[3]);

    register_logic r1 (clk, enable, regIn, q);



endmodule
```

Test Bench for RCA:

```
`timescale 1ns / 1ps


module tb_RCA4bit;

    reg clk;
    reg enable;
    reg [3:0] A,B;
    reg Cin;
    wire [4:0] q;

    RCA4bit a1 (
        .clk(clk),
        .enable(enable),
        .A(A),
        .B(B),
        .Cin(Cin),
        .q(q)
        );


    initial begin
    clk = 0;
    enable = 1;

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

    #20;

    A = 4'b0111;
```

```
B = 4'b0111;
Cin = 1'b0;

#20;

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

#20;

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

#20;

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

#20;

A = 4'b1001;
B = 4'b1010;
Cin = 1'b1;

#20;

A = 4'b1111;
B = 4'b1111;
Cin = 1'b0;

#20;
```

```
enable = 0;

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

#20;

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

#20;

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

#20;

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

#20;

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

#20;
```

```
    A = 4'b1001;
    B = 4'b1010;
    Cin = 1'b1;

    #20;

    A = 4'b1111;
    B = 4'b1111;
    Cin = 1'b0;

    #20;



    end

    always begin
        #5 clk = ~clk;
    end

endmodule
```

Table for RCA:

A[3:0]	B[3:0]	Cin	Sum[3:0]	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

Constraints file for RCA:

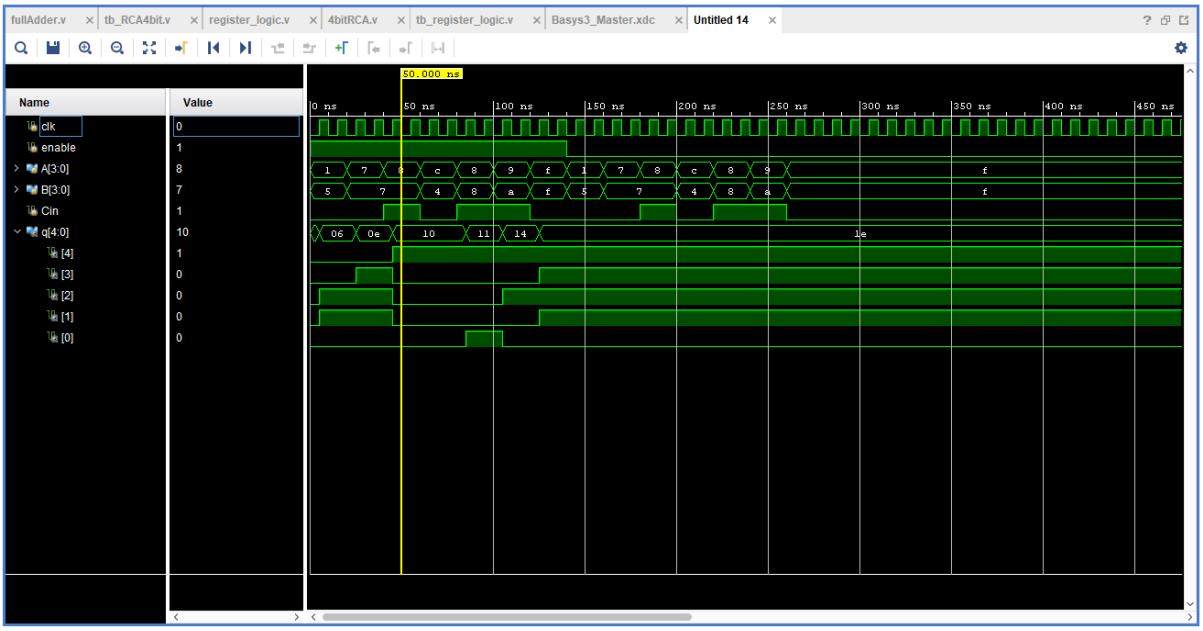
```
set_property PACKAGE_PIN W5 [get_ports clk]

        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 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}]

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]}]

set_property PACKAGE_PIN U18 [get_ports enable]

set_property IOSTANDARD LVCMOS33 [get_ports enable]

Simulation Waveform for RCA:



Part 2:

Equations of Ci and Si:

C0 = Cin;

C1 = G0 + P0*C0

C2 = G1 + P1*G0 + P1*P0*C0

C3 = G2 + P2*G1 + P2*P1*G0 + P2*P1*P0*C0

C 4 = G3 + P3*G2 + P3*P2*G1 + P3*P2*P1*G0 + P3*P2*P1*P0*C0

S0 = P0 xor C0

S1 = P1 xor C1

S2 = P2 xor C2

S3 = P3 xor C3

Verilog Code for CLA adder:

```
`timescale 1ns / 1ps



module CLA4bitAdder(
    input clk,
    input enable,
    input [3:0] A,B,
    input Cin,
    output [4:0] q
    );

    wire [3:0] P;
    wire [3:0] G;
    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 G[0] = A[0]&B[0];
    assign G[1] = A[1]&B[1];
    assign G[2] = A[2]&B[2];
    assign G[3] = A[3]&B[3];

    wire[3:0] S;
    wire[4:0] C;

    assign C[0] = Cin;
    assign C[1] = G[0] | (P[0]&C[0]);
    assign C[2] = G[1] | (P[1]&G[0]) | (P[1]&P[0]&C[0]);
    assign C[3] = G[2] | (P[2]&G[1]) | (P[2]&P[1]&G[0]) | (P[2]&P[1]&P[0]&C[0]);
    assign C[4] = 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]);

    assign S[0] = P[0]^C[0];
    assign S[1] = P[1]^C[1];
    assign S[2] = P[2]^C[2];
    assign S[3] = P[3]^C[3];

    wire [4:0] regIn;
    assign regIn [0] = S[0];
    assign regIn [1] = S[1];
    assign regIn [2] = S[2];
    assign regIn [3] = S[3];
    assign regIn [4] = C[4];

    register_logic r1 (clk, enable, regIn, q);


endmodule
```

Verilog code for register logic for CLA:

```
`timescale 1ns / 1ps



module register_logic(
    input clk,
    input enable,
    input [4:0] data,
    output reg [4:0] q
    );

    initial begin
        q = 0;
    end

    always @(posedge clk) begin
        if (enable)
            q <= data;
    end


endmodule
```

Test Bench for CLA:

```
`timescale 1ns / 1ps



module tb_CLA4bitAdder;

    reg clk;
```

```
reg enable;
reg [3:0] A,B;
reg Cin;
wire [4:0] q;

CLA4bitAdder a1 (
   .clk(clk),
   .enable(enable),
   .A(A),
   .B(B),
   .Cin(Cin),
   .q(q)
   );


initial begin
clk = 0;
enable = 1;

A = 4'b0000;
B = 4'b0101;
Cin = 1'b0;

#20;

A = 4'b0101;
B = 4'b0111;
Cin = 1'b0;

#20;

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

```
#20;

A = 4'b1001;
B = 4'b0100;
Cin = 1'b0;

#20;

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

#20;

A = 4'b1101;
B = 4'b1010;
Cin = 1'b1;

#20;

A = 4'b1110;
B = 4'b1111;
Cin = 1'b0;

#20;


enable = 0;

A = 4'b0000;
B = 4'b0101;
Cin = 1'b0;
```

```
#20;

A = 4'b0101;
B = 4'b0111;
Cin = 1'b0;

#20;

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

#20;

A = 4'b1001;
B = 4'b0100;
Cin = 1'b0;

#20;

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

#20;

A = 4'b1101;
B = 4'b1010;
Cin = 1'b1;

#20;

A = 4'b1110;
B = 4'b1111;
```

```
    Cin = 1'b0;

    #20;

    end

    always begin
        #5 clk = ~clk;
    end

endmodule
```

Table 2:

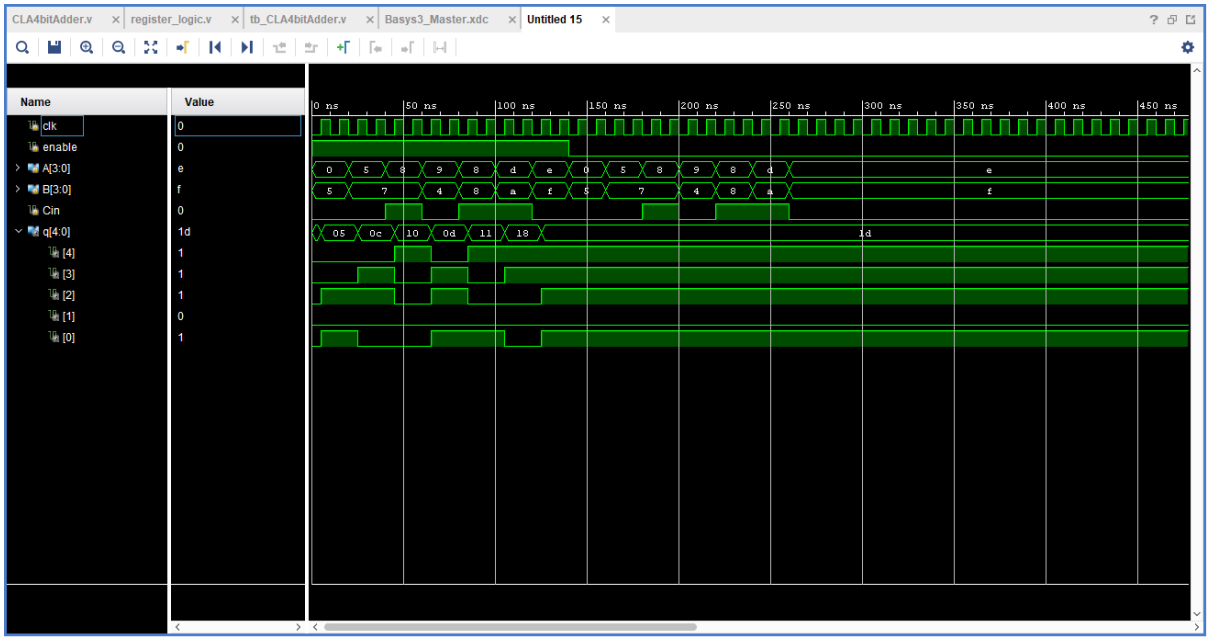
A[3:0]	B[3:0]	Cin	Sum[3:0]	Cout
0000	0101	0	0101	0
0101	0111	0	1100	0
1000	0111	1	0	1
1001	0100	0	1101	0
1000	1000	1	0001	1
1101	1010	1	1000	1
1110	1111	0	1101	1

Constraints File for CLA:

```
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 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]}]
```

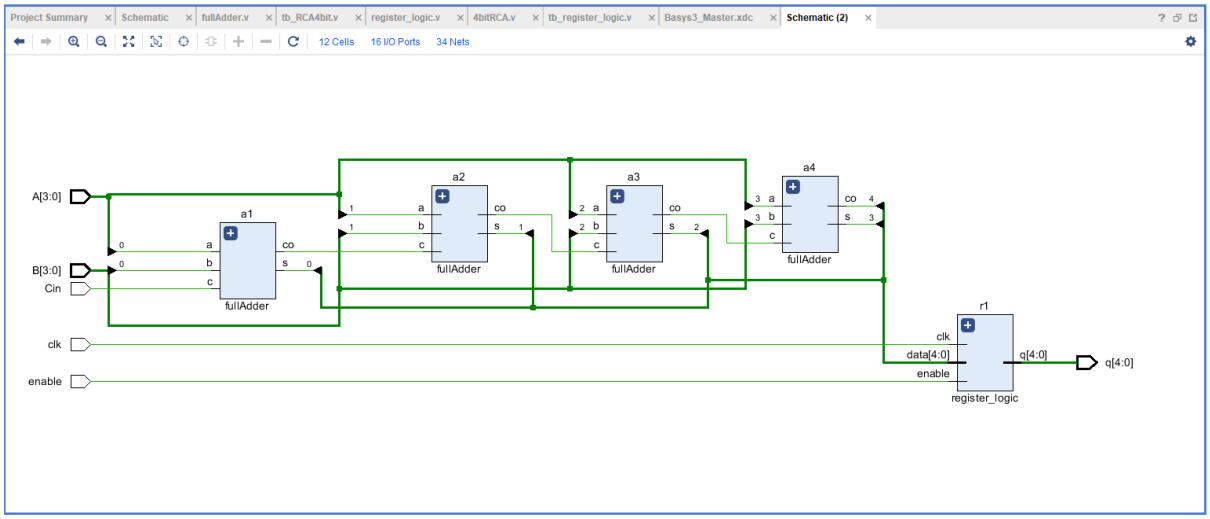
```
set_property PACKAGE_PIN W17 [get_ports {A[3]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {A[3]}]
set_property PACKAGE_PIN W15 [get_ports {B[0]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {B[0]}]
set_property PACKAGE_PIN V15 [get_ports {B[1]}]
        set_property IOSTANDARD LVCMOS33 [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}]

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]}]

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

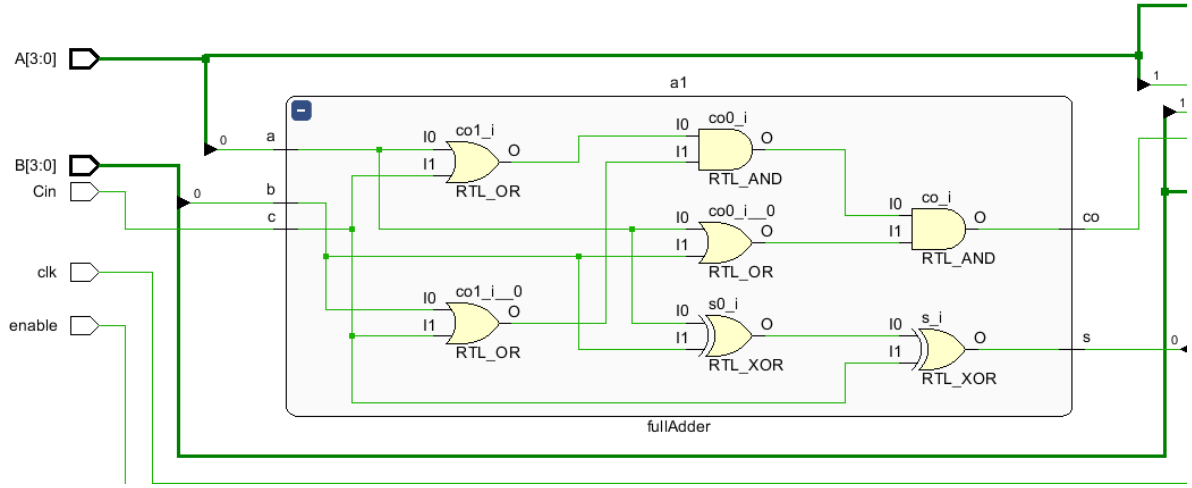
Simulation waveform for CLA:



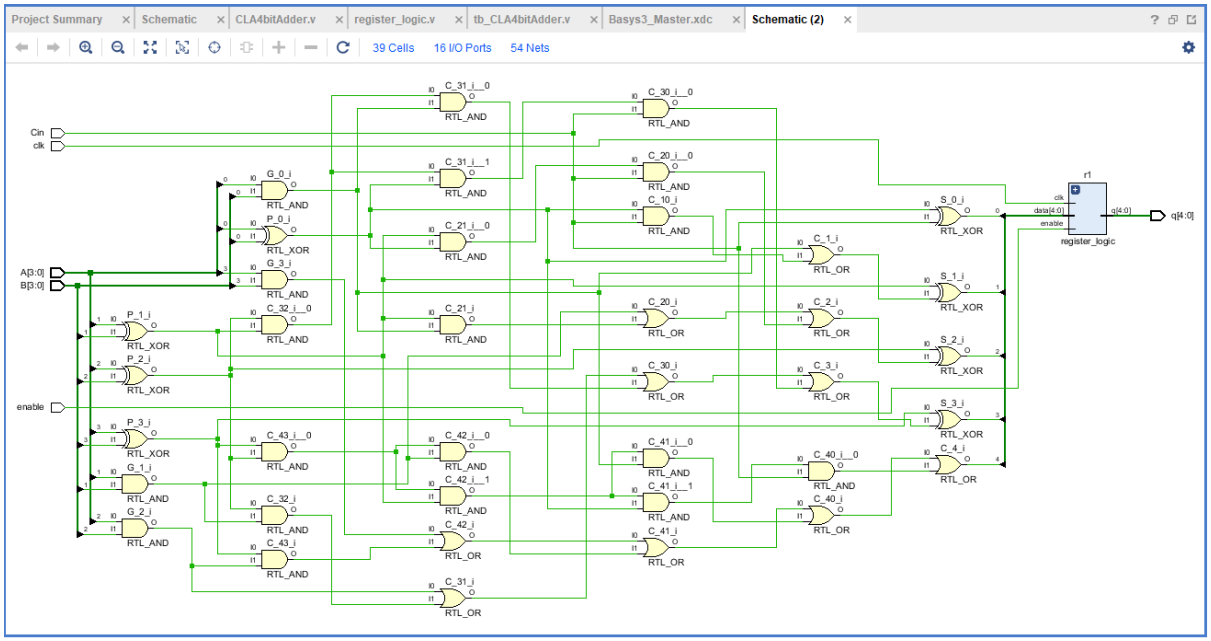
Part 3:

Schematics for RCA:





CLA Schematics:



Delay and area calculations:

full adder max path delay: 4*(1 or + 2 and) = 4*(2ns + 2*3ns) = 32ns

full adder area: 4*(3or + 2xor + 2and) = 4*(3*4 + 2*6 + 2*4) = 128

CLA adder max path delay: Xor + and + and + and + or + xOr = 3 + 3 + 3 +3 +2 +3 = 17ns

CLA adder area: 10*(Or) + 20*(and) + 8*(xor) = 10*(4) + 20*(4) + 8*(6) = 168

Conclusion:

The difference between the carry look ahead adder and the ripple carry adder is a trade off between size and speed. The carry look ahead adder is larger but calculates the final value more quickly. The ripple carry adder is smaller but calculates the final value more slowly.