Jost Luebbe jl64249 Tues/Thurs 3-4:30pm

Lab 5 Report

Part 1 – Ripple Carry Adder (RCA)

i. Design Files for RCA

a. RCA_4bits

```
module RCA_4bits(
        input clk,
        input enable,
        input [3:0] A, B,
        input Cin,
        output [4:0] Q
    );

    wire c1, c2, c3;
    wire [4:0] Data;

    full_adder  FA1 (A[0], B[0], Cin, Data[0], c1),
                FA2 (A[1], B[1], c1, Data[1], c2),
                FA3 (A[2], B[2], c2, Data[2], c3),
                FA4 (A[3], B[3], c3, Data[3], Data[4]);

    register_logic r1(clk, enable, Data, Q);
endmodule
```

b. full_adder

```
module full_adder(
        input A, B, Cin,
        output S, Cout
    );
    wire s1, c1, c2;

    xor (S, A, B, Cin);
    or (Cout, B & Cin, A & Cin, A & B);
endmodule
```

c. register_logic

```
module register_logic(
        input clk,
        input enable,
        input [4:0] Data,
        output reg [4:0] Q
    );

    initial begin
        Q = 4'b0000;
    end

    always @(posedge clk) begin
        if(enable)
            Q = Data;
    end
endmodule
```

ii. Test-bench for RCA

```
module tb_RCA_4bits;

    reg clk;
    reg enable;
    reg [3:0] A;
    reg [3:0] B;
    reg Cin;

    wire [4:0] Q;

    RCA_4bits uut (
        .clk(clk),
        .enable(enable),
        .A(A),
        .B(B),
        .Cin(Cin),
        .Q(Q)
    );

    initial begin

    // Initialization
    clk = 0;
    enable = 1;
    A = 0;
    B = 0;
    Cin = 0;

    #10;

    // Test Case 1
    A = 4'b0001;
    B = 4'b0101;
    Cin = 1'b0;
    enable = 1;

    #10;

    enable = 0;

    #10;

    // Test Case 2
    A = 4'b0111;
    B = 4'b0111;
    Cin = 1'b0;
    enable = 1;

    #10;

    enable = 0;

    #10;

    // Test Case 3
    A = 4'b1000;
    B = 4'b0111;
    Cin = 1'b1;
    enable = 1;

    #10;

    enable = 0;

    #10;

    // Test Case 4
    A = 4'b1100;
    B = 4'b0100;
    Cin = 1'b0;
    enable = 1;

    #10;

    enable = 0;

    #10;

    // Test Case 5
    A = 4'b1000;
    B = 4'b1000;
    Cin = 1'b01;
    enable = 1;

    #10;

    enable = 0;

    #10;

    // Test Case 6
    A = 4'b1001;
    B = 4'b1010;
    Cin = 1'b1;
    enable = 1;

    #10;

    enable = 0;

    #10;

    // Test Case 7
    A = 4'b1111;
    B = 4'b1111;
    Cin = 1'b0;
    enable = 1;

    #10;

    enable = 0;

    #10;

    clk = 0;
    enable = 1;
    A = 0;
    B = 0;
    Cin = 0;

    #10;

    end

    always
    #5 clk = ~clk;

endmodule
```

iii. Testcases 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

iv. Constraints File for RCA

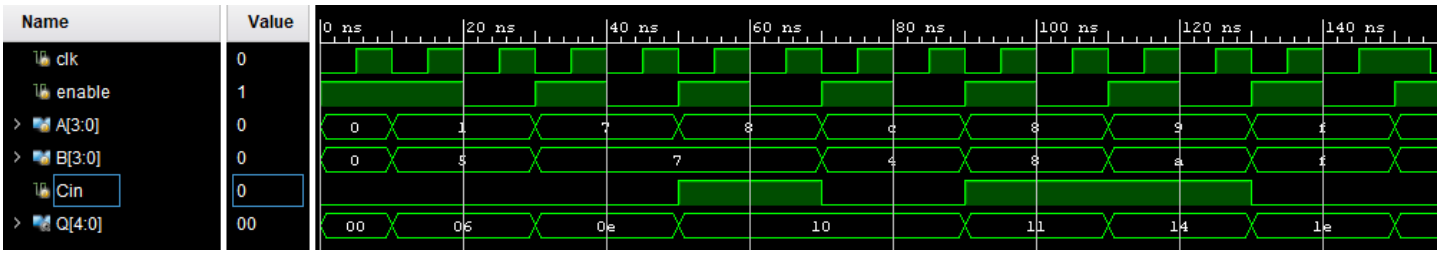
```
## Clock signal
set_property PACKAGE_PIN W5 [get_ports clk]
    set_property IOSTANDARD LVCMOS33 [get_ports clk]
    create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]

## 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}]

## 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]}]

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

v. Simulation Waveform for RCA TB



Part 2 – Carry Lookahead Adder (CLA)

vi. C_i and S_i Equations

$$C_0 = C_{in}$$

$$C_1 = P_0C_0 + G_0$$

$$C_2 = P_1P_0C_0 + P_1G_0 + G_1$$

$$C_3 = P_2P_1P_0C_0 + P_2P_1G_0 + P_2G_1 + G_2$$

$$C_4 = C_{out} = P_3P_2P_1P_0C_0 + P_3P_2P_1G_0 + P_3P_2G_1 + P_3G_2 + G_3$$

$$S_0 = P_0 \oplus C_0$$

$$S_1 = P_1 \oplus C_1$$

$$S_2 = P_2 \oplus C_2$$

$$S_3 = P_3 \oplus C_3$$

vii. Design Files for CLA

a. CLA_4bits

```
module CLA_4bits(
        input clk,
        input enable,
        input [3:0] A, B,
        input Cin,
        output [4:0] Q
    );

    wire [3:0] G, P, S;
    wire [4:0] C, Data;
    wire Cout;

    assign G = A & B;
    assign P = A ^ B;
    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 Cout = 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 = P ^ C;

    assign Data = {Cout, S};

    register_logic r1(clk, enable, Data, Q);
endmodule
```

b. register_logic

```
module register_logic(
        input clk,
        input enable,
        input [4:0] Data,
        output reg [4:0] Q
    );

    initial begin
        Q = 4'b0000;
    end

    always @(posedge clk) begin
        if(enable)
            Q = Data;
    end
endmodule
```

viii. Test-bench for CLA

```
module tb_CLA_4bits;

    reg clk;
    reg enable;
    reg [3:0] A;
    reg [3:0] B;
    reg Cin;

    wire [4:0] Q;

    CLA_4bits uut (
        .clk(clk),
        .enable(enable),
        .A(A),
        .B(B),
        .Cin(Cin),
        .Q(Q)
    );

    initial begin

    // Initialization
    clk = 0;
    enable = 1;
    A = 0;
    B = 0;
    Cin = 0;

    #10;

    // Test Case 1
    A = 4'b0000;
    B = 4'b0101;
    Cin = 1'b0;
    enable = 1;

    #10;

    enable = 0;

    #10;

    // Test Case 2
    A = 4'b0101;
    B = 4'b0111;
    Cin = 1'b0;
    enable = 1;

    #10;

    enable = 0;

    #10;

    // Test Case 3
    A = 4'b1000;
    B = 4'b0111;
    Cin = 1'b1;
    enable = 1;

    #10;

    enable = 0;

    #10;

    // Test Case 4
    A = 4'b1001;
    B = 4'b0100;
    Cin = 1'b0;
    enable = 1;

    #10;

    enable = 0;

    #10;

    // Test Case 5
    A = 4'b1000;
    B = 4'b1000;
    Cin = 1'b1;
    enable = 1;

    #10;

    enable = 0;

    #10;

    // Test Case 6
    A = 4'b1101;
    B = 4'b1010;
    Cin = 1'b1;
    enable = 1;

    #10;

    enable = 0;

    #10;

    // Test Case 7
    A = 4'b1110;
    B = 4'b1111;
    Cin = 1'b0;
    enable = 1;

    #10;

    enable = 0;

    #10;

    clk = 0;
    enable = 1;
    A = 0;
    B = 0;
    Cin = 0;

    end

    always
    #5 clk = ~clk;

endmodule
```

ix. Testcases Table for CLA

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

x. Constraints File for CLA

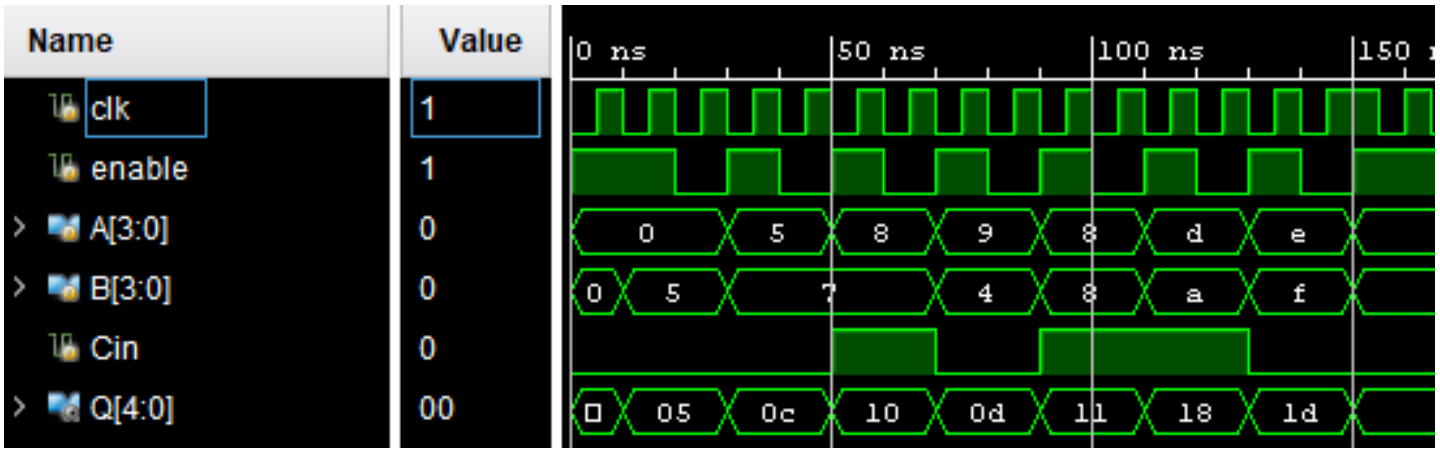
```
## Clock signal
set_property PACKAGE_PIN W5 [get_ports clk]
    set_property IOSTANDARD LVCMOS33 [get_ports clk]
    create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]

## 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}]

## 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]}]

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

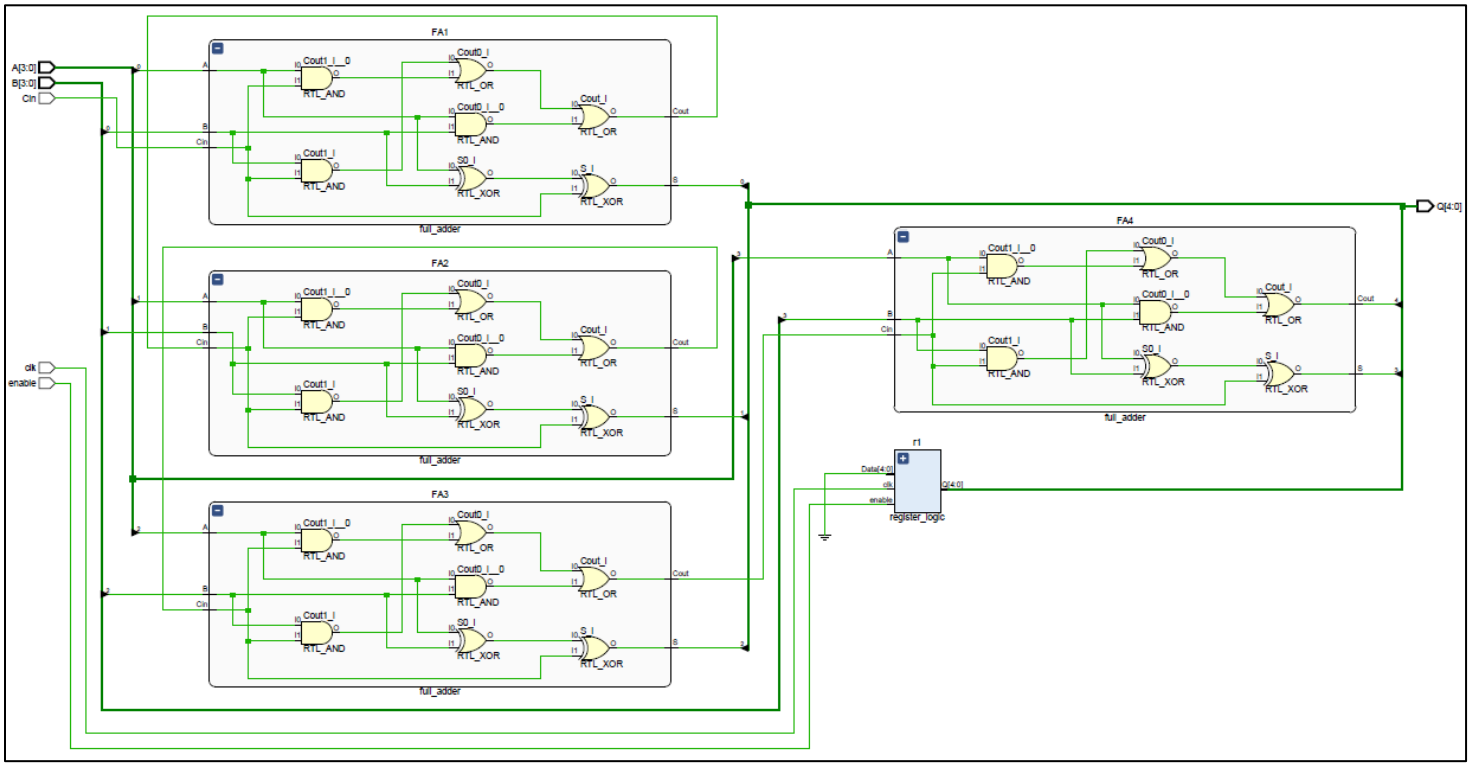
xi. Simulation Waveform for CLA TB


Name
Value
clk
1
enable
1
A[3:0]
0
B[3:0]
0
Cin
0
Q[4:0]
00
0 ns
50 ns
100 ns
0
5
8
9
8
d
e
0
5
7
4
8
a
f
05
0c
10
0d
11
18
1d

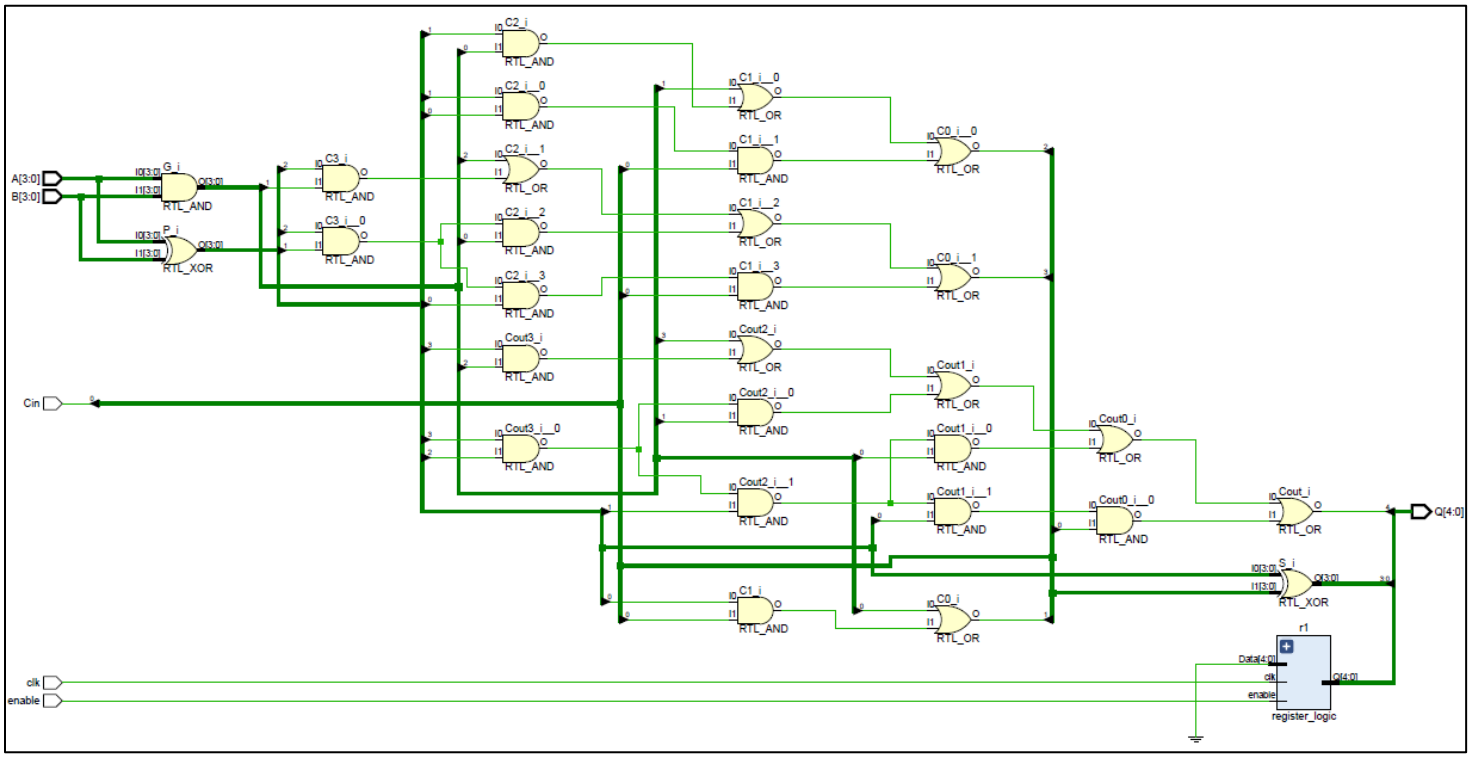
Part 3 – Speed and Area Comparison of RCA vs. CLA

xii. Screenshots of Gate-Level Schematics

a. RCA



b. CLA



xiii. Delay and Area Calculations

a. RCA

There are 4 full adders each with 3 AND, 2 OR, and 2 XOR gates. The areas for these gates are 4, 4, and 6 respectively. This means the total area comes for the equation: $4\big(3(4) + 2(4) + 2(6)\big) = 4(32) = 128$ units of area.

The critical path through our full adder design is the C_{out} path for each full adder. This means going through an AND gate, then 2 OR gates. The time take for each full adder is $3 + 2(2) = 7ns$, this multiplied by 4 since there are 4 full adders to go through means the critical path would take $28ns$.

b. CLA

My CLA design has 20 AND, 10 OR, and 8 XOR gates. Multiplied by their areas this gives us the following equation: $20(4) + 10(4) + 8(6) = 168$ units of area.

The critical path through the carry adder is through 1 XOR, 4 AND, and 2 OR gates. This gives us the following time equation $1(3) + 4(3) + 2(2) = 19ns$.

xiv. RCA and CLA Delay and Area Analysis

The RCA design has a smaller area footprint but takes a lot longer than the CLA design. The CLA design has the converse, with a larger area but a lesser critical path. This means in space constrained design I would implement the RCA, but in a time constrained design I would implement the CLA.