Lab 5 Report

Name: Connie Wang, Harshika Jha

UT EID: cw39276, hj6963

Section: 17745, 17745

Checklist:

Part 1 –

i. Design file (.v) for the Ripple Carry Adder

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

        wire r0,r1,r2,r3;

        full_adder a0(.A(A[0]),.B(B[0]),.cin(cin),.S(Q[0]),.cout(r0));
        full_adder a1(.A(A[1]),.B(B[1]),.cin(r0),.S(Q[1]),.cout(r1));
        full_adder a2(.A(A[2]),.B(B[2]),.cin(r1),.S(Q[2]),.cout(r2));
        full_adder a3(.A(A[3]),.B(B[3]),.cin(r2),.S(Q[3]),.cout(Q[4]));

        register_logic r(.clk(clk),.enable(enable),.Data(Q), .Q());
```

Endmodule

```
module full_adder(
        input A,
        input B,
        input cin,
        output S,
        output cout
        );
        assign cout = (A&cin)| (B&cin)|(A&B);
        assign S = A^B^cin;
endmodule


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

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

ii. Test-benchtimescale 1ns / 1ps

```
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
clk = 0;
enable = 0;
A = 4'b0001;
B = 4'b0101;
cin = 1'b0;
enable = 1;

#10

enable = 0;
A = 4'b0111;
B = 4'b0111;
cin = 1'b0;
```

```
enable = 1;

#10


enable = 0;
A = 4'b1000;
B = 4'b0111;
cin = 1'b1;
enable = 1;

#10

enable = 0;
A = 4'b1100;
B = 4'b0100;
cin = 1'b0;
enable = 1;

#10

enable = 0;
A = 4'b1000;
B = 4'b1000;
cin = 1'b1;
enable = 1;

#10

enable = 0;
```

```
A = 4'b1001;
B = 4'b1010;
cin = 1'b1;
enable = 1;

#10

enable = 0;
A = 4'b1111;
B = 4'b1111;
cin = 1'b0;
enable = 1;

end
always
#5 clk = ~clk;

endmodule
```

iii. Complete Table 1 from the simulation

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 (Just the uncommented portion)

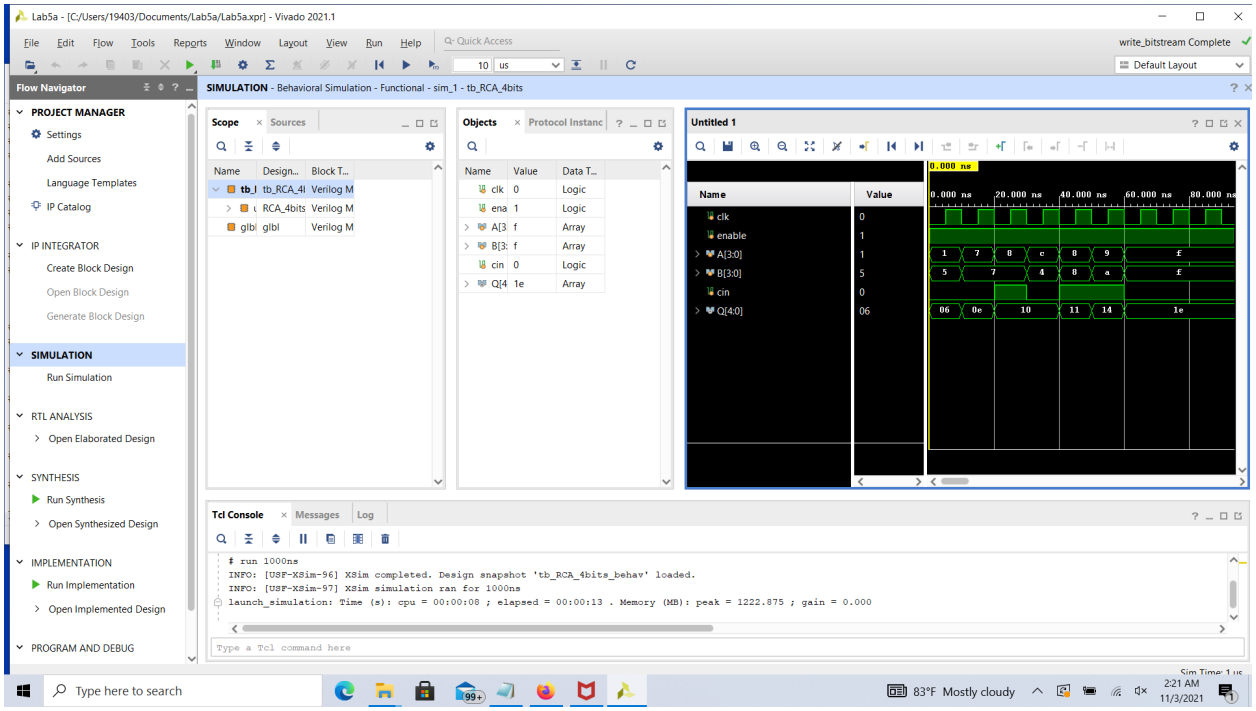
```
## Clock signal - Uncomment if needed (will be used in future labs)
set_property PACKAGE_PIN W5 [get_ports clk]

    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 the above test-cases



Part 2 –

vi. All the equations for C_i's and S_i's

C0 = cin

C1= G0+P0C0

C2 = G1+P1C1 = G1 + P1G0+P1P0C0

C3 = G2+P2C2 = G2 + P2G1+P2P1G0+P2P1P0C0

C4 = G3+P3C3 = G3+ P3G2+P3P2G1+P3P2P1G0+P3P2P1P0C0

S0 = P0 XOR C0 = P0 XOR cin

S1 = P1 XOR C1 = P1 XOR (G0+P0C0)

S2 = P2 XOR C2 = P2 XOR (G1 + P1G0+P1P0C0)
S3 = P3 XOR C3 = P3 XOR (G2 + P2G1+P2P1G0+P2P1P0C0)
vii. Design files (.v) for the Carry Lookahead Adder and Register Logic

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

        always @(posedge clk)
        if(enable)
        Q = Data;
Endmodule



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

```
	//wire [3:0] Sum;
	wire Cout;
	wire [4:0] Data;
	assign C[0] = Cin;

	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];

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

	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] | (G[1]&P[2]) | (P[2]&P[1]&G[0]) |
(P[2]&P[1]&P[0]&C[0]));

	//assign Q[4] = COut;
```

```
        //assign Sum[3:0] = S;

    // wire[4:0] Data;
        assign Data[3:0] = S;
        assign Data[4] = Cout;

        register_logic r(.clk(clk), .enable(enable),.Data(Data),.Q(Q));


endmodule
```

viii. Test-bench

```
`timescale 1ns / 1ps

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
clk = 0;
enable = 0;
A = 4'b0000;
B = 4'b0101;
Cin = 1'b0;
enable = 1;

#10

enable = 0;
A = 4'b0101;
B = 4'b0111;
Cin = 1'b0;
enable = 1;

#10


enable = 0;
A = 4'b1000;
B = 4'b0111;
Cin = 1'b1;
enable = 1;

#10
```

```
enable = 0;
A = 4'b1001;
B = 4'b0100;
Cin = 1'b0;
enable = 1;

#10

enable = 0;
A = 4'b1000;
B = 4'b1000;
Cin = 1'b1;
enable = 1;

#10

enable = 0;
A = 4'b1101;
B = 4'b1010;
Cin = 1'b1;
enable = 1;

#10

enable = 0;
A = 4'b1110;
B = 4'b1111;
Cin = 1'b0;
enable = 1;
```

```
end
always
#5 clk = ~clk;

endmodule
```

ix. Complete Table 2 from the simulation

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

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

Table 2. Testcases for Carry Lookahead Adder Verification

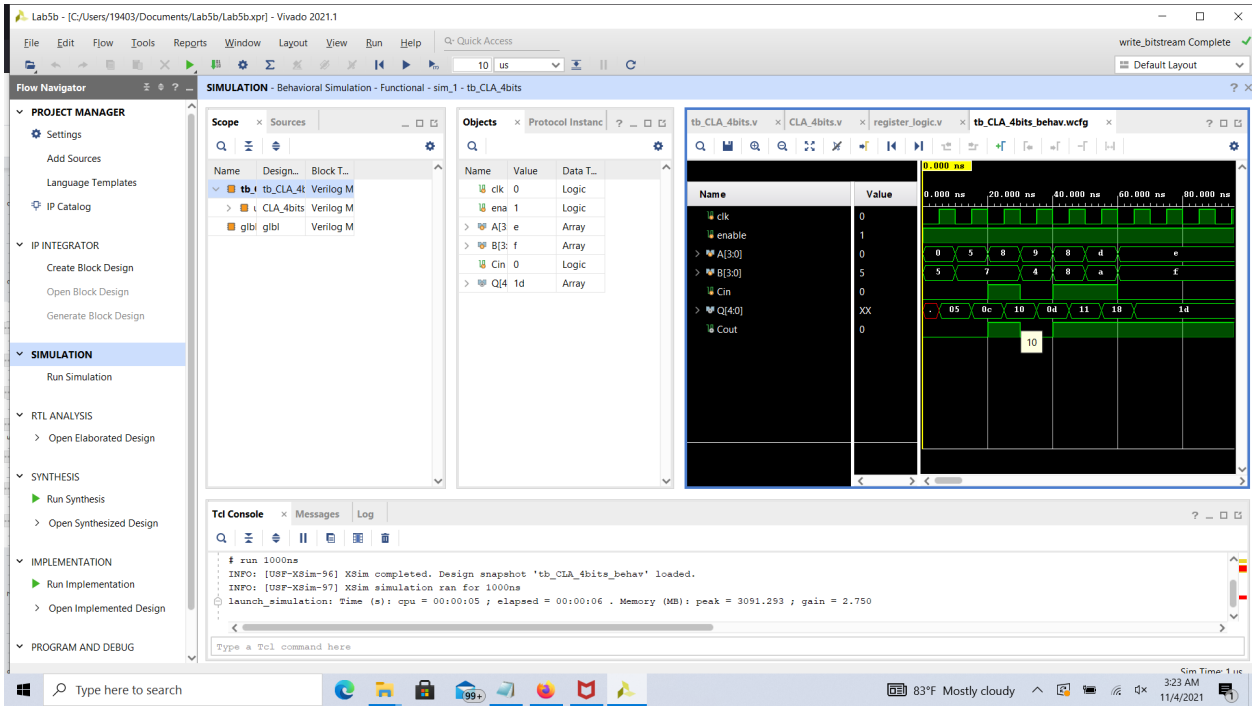
x. Constraints File (Just the uncommented portion)

```
## Clock signal - Uncomment if needed (will be used in future labs)
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}]
```

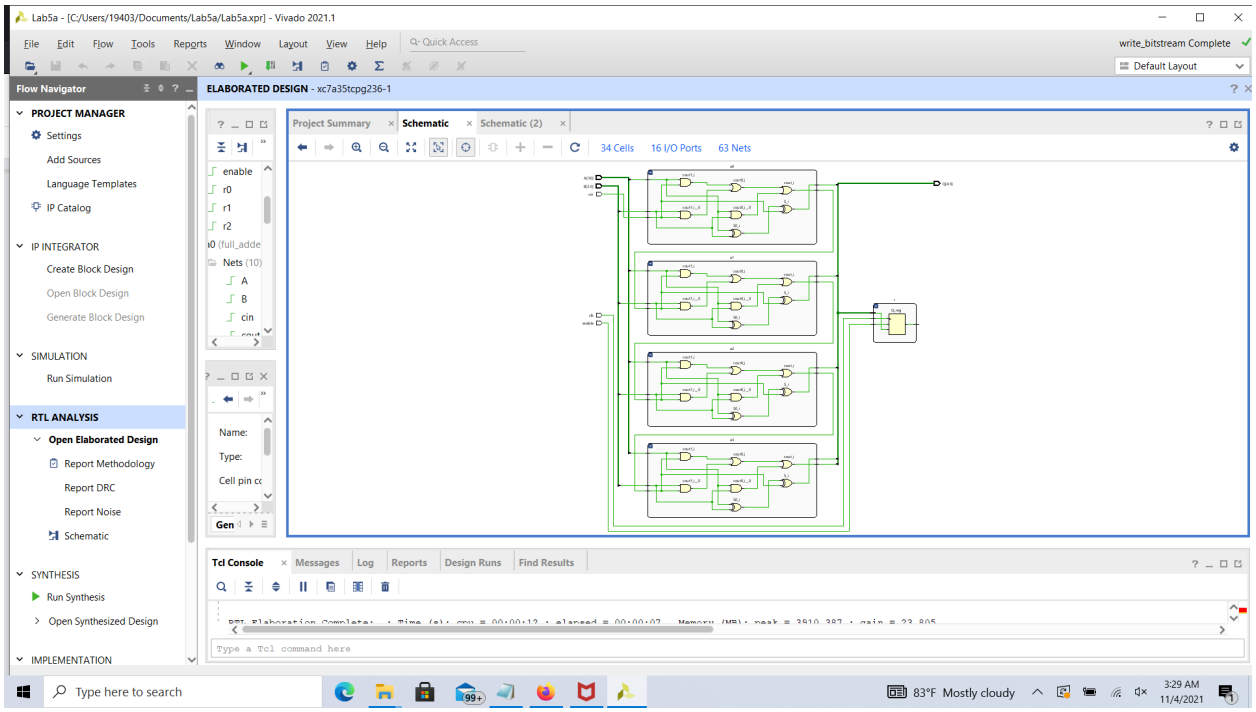
```
## LEDs
set_property PACKAGE_PIN U16 [get_ports {Q[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Q[0]}]
set_property PACKAGE_PIN E19 [get_ports {Q[1]}]

    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]
```

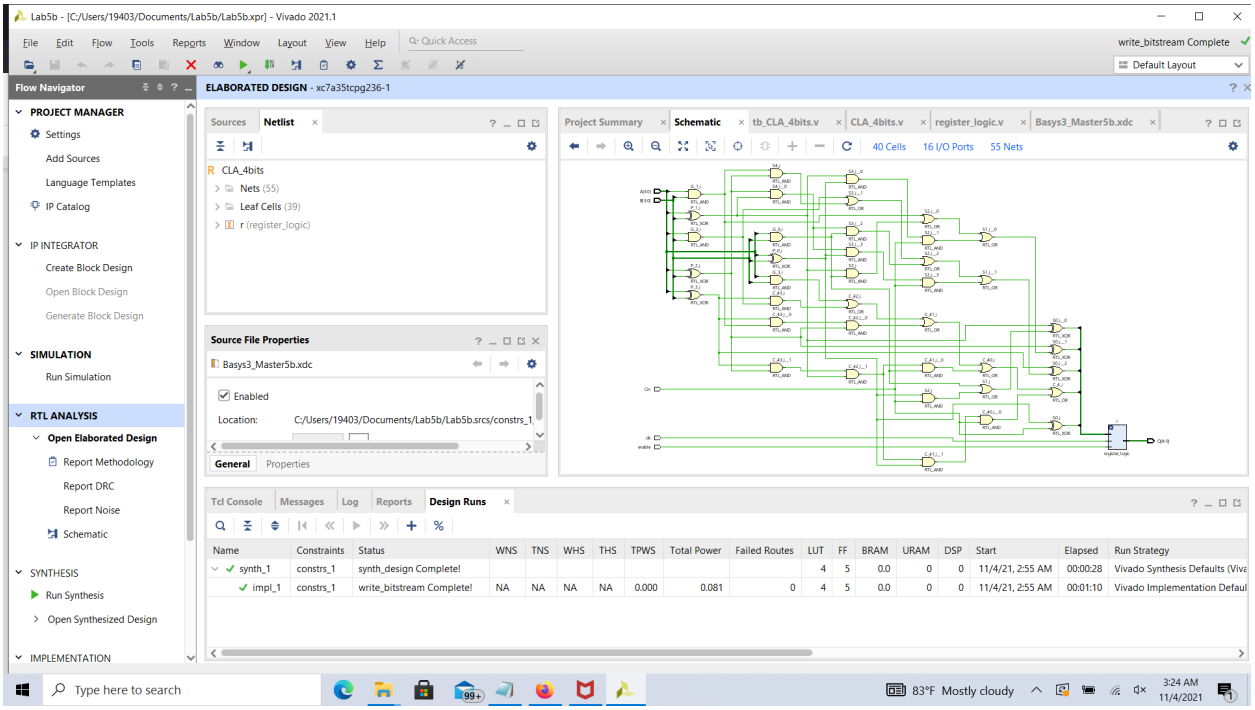
xi. Simulation waveform for the above test-cases



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

a ripple carry adder delay =4(2+2+3)= 28
area= 4[3(4)+2(4)+2(6)]=128
carry-lookahead adder delay= 5(3)+4(2)=23
area= 4(20)+4(10)+6(8)=168

xiv. Brief conclusion regarding the pros and cons of each of the techniques

Pros:
a ripple carry adder = less complex hardware
carry-lookahead adder = short delay

Cons:
a ripple carry adder = more delay
carry-lookahead adder = less complex hardware

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,