Lab 5 Report

Name: Dhruv Sandesara

UT EID: djs3967

Section: 15295

Checklist:

Part 1 –

i. Design file (.v) for the Ripple Carry Adder
ii. `timescale 1ns / 1ps
iii. //
///
iv. // Company:
v. // Engineer:
vi. //
vii. // Create Date: 04/06/2018 02:57:29 PM
viii. // Design Name:
ix. // Module Name: RCA_4bits
x. // Project Name:
xi. // Target Devices:
xii. // Tool Versions:
xiii. // Description:
xiv. //
xv. // Dependencies:
xvi. //
xvii. // Revision:
xviii. // Revision 0.01 - File Created
xix. // Additional Comments:

```
xx.   //
xxi.  //////////////////////////////////////////////////////////////////////////////
      ///
xxii.
xxiii.
xxiv. module RCA_4bits(
xxv.     input clk,
xxvi.    input enable,
xxvii.   input [3:0] A,
xxviii.  input [3:0] B,
xxix.    input Cin,
xxx.     output [4:0]Q
xxxi.    );
xxxii.
xxxiii.  wire cout0, cout1, cout2;
xxxiv.      wire [4:0] Data;
xxxv.       full_adder f0 (.A(A[0]), .B(B[0]), .Cin(Cin),   .S(Data[0]), .Cout(cout0));
xxxvi.      full_adder f1 (.A(A[1]), .B(B[1]), .Cin(cout0), .S(Data[1]), .Cout(cout1));
xxxvii.     full_adder f2 (.A(A[2]), .B(B[2]), .Cin(cout1), .S(Data[2]), .Cout(cout2));
xxxviii.    full_adder f3 (.A(A[3]), .B(B[3]), .Cin(cout2), .S(Data[3]), .Cout(Data[4]));
xxxix.
xl.         register_logic r0 (.clk(clk), .enable(enable), .Data(Data), .Q(Q));
xli.
xlii.
xliii.
xliv. Endmodule
xlv.  `timescale 1ns / 1ps
xlvi. //////////////////////////////////////////////////////////////////////////////
      ///
xlvii.  // Company:
xlviii. // Engineer:
xlix.   //
l.      // Create Date: 04/06/2018 02:59:43 PM
```

```
// Design Name:
// Module Name: full_adder
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


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

    reg [1:0]out;
    always@(*)begin
    out = A+B+Cin;
    assign S= out[0];
    assign Cout= out[1];

    //{Cout,S}= A+B+Cin;



    end
```

```
lxxxiii.
lxxxiv.   Endmodule
lxxxv.    `timescale 1ns / 1ps
lxxxvi.   //////////////////////////////////////////////////////////////////////////////////
          ///
lxxxvii.  // Company:
lxxxviii. // Engineer:
lxxxix.   //
xc.       // Create Date: 04/06/2018 03:01:08 PM
xci.      // Design Name:
xcii.     // Module Name: register_logic
xciii.    // Project Name:
xciv.     // Target Devices:
xcv.      // Tool Versions:
xcvi.     // Description:
xcvii.    //
xcviii.   // Dependencies:
xcix.     //
c.        // Revision:
ci.       // Revision 0.01 - File Created
cii.      // Additional Comments:
ciii.     //
civ.      //////////////////////////////////////////////////////////////////////////////////
          ///
cv.
cvi.
cvii.     module register_logic(
cviii.        input clk,
cix.          input enable,
cx.           input [4:0] Data,
cxi.          output reg [4:0] Q
cxii.         );
cxiii.
```

```
cxiv.          initial begin
cxv.           Q <= 0;
cxvi.       end
cxvii.      always @(posedge clk)
cxviii.        if(enable)
cxix.             Q <= Data;
cxx.
cxxi.
cxxii.   Endmodule


cxxiii.  Test-bench
cxxiv.   `timescale 1ns / 1ps
cxxv.    //////////////////////////////////////////////////////////////////////////////////
         ///
cxxvi.   // Company:
cxxvii.  // Engineer:
cxxviii. //
cxxix.   // Create Date: 04/06/2018 03:24:05 PM
cxxx.    // Design Name:
cxxxi.   // Module Name: tb_RCA_4bits
cxxxii.  // Project Name:
cxxxiii. // Target Devices:
cxxxiv.  // Tool Versions:
cxxxv.   // Description:
cxxxvi.  //
cxxxvii. // Dependencies:
cxxxviii.//
cxxxix.  // Revision:
cxl.     // Revision 0.01 - File Created
cxli.    // Additional Comments:
cxlii.   //
```

```
cxliii.  ////////////////////////////////////////////////////////////////////////////
         ///
cxliv.
cxlv.
cxlvi.   module tb_RCA_4bits;
cxlvii.    reg clk;
cxlviii.   reg enable;
cxlix.     reg [3:0] A;
cl.        reg [3:0] B;
cli.       reg Cin;
clii.      wire [4:0] Q;
cliii.
cliv.
clv.        RCA_4bits uut (
clvi.         .clk(clk),
clvii.        .enable(enable),
clviii.       .A(A),
clix.         .B(B),
clx.          .Cin(Cin),
clxi.         .Q(Q)
clxii.     );
clxiii.    initial begin
clxiv.
clxv.      clk = 0;
clxvi.     enable = 0;
clxvii.    A = 4'b0000;
clxviii.   B = 4'b0000;
clxix.     Cin = 0;
clxx.
clxxi.     #50;
clxxii.    enable = 1;
clxxiii.   A = 4'b0001;
clxxiv.    B = 4'b0101;
```

```
clxxv.      Cin = 0;
clxxvi.
clxxvii.    #50;
clxxviii.   A = 4'b0111;
clxxix.     B = 4'b0111;
clxxx.      Cin = 0;
clxxxi.
clxxxii.    #50;
clxxxiii.   A = 4'b1000;
clxxxiv.    B = 4'b0111;
clxxxv.     Cin = 1;
clxxxvi.
clxxxvii.   #50;
clxxxviii.  A = 4'b1100;
clxxxix.    B = 4'b0010;
cxc.        Cin = 0;
cxci.
cxcii.      #50;
cxciii.     A = 4'b1000;
cxciv.      B = 4'b1000;
cxcv.       Cin = 1;
cxcvi.
cxcvii.     #50;
cxcviii.    A = 4'b1001;
cxcix.      B = 4'b1010;
cc.         Cin = 1;
cci.
ccii.       #50;
cciii.      A = 4'b1111;
cciv.       B = 4'b1111;
ccv.        Cin = 0;
ccvi.
ccvii.      end
```

ccviii.

ccix. always

ccx. #5 clk = ~clk;

ccxi.

ccxii.

ccxiii. endmodule

ccxiv. Complete Table 1 from the simulation

A[3:0]	B[3:0]	Cin	Sum[3:0]	Cout
0001	0101	0	6	0
0111	0111	0	E	0
1000	0111	1	0	1
1100	0100	0	E	0
1000	1000	1	1	1
1001	1010	1	4	1
1111	1111	0	E	1

Table 1. Testcases for Ripple Carry Adder Verification

ccxv. Constraints File (Just the uncommented portion)

i. ## Clock signal

ii. set_property PACKAGE_PIN W5 [get_ports clk]

iii. set_property IOSTANDARD LVCMOS33 [get_ports clk]

iv. create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]

v.

vi. ## Switches

vii. set_property PACKAGE_PIN V17 [get_ports {A[0]}]

viii. set_property IOSTANDARD LVCMOS33 [get_ports {A[0]}]

ix. set_property PACKAGE_PIN V16 [get_ports {A[1]}]

x. set_property IOSTANDARD LVCMOS33 [get_ports {A[1]}]

xi. set_property PACKAGE_PIN W16 [get_ports {A[2]}]

xii. set_property IOSTANDARD LVCMOS33 [get_ports {A[2]}]

xiii. set_property PACKAGE_PIN W17 [get_ports {A[3]}]

xiv. set_property IOSTANDARD LVCMOS33 [get_ports {A[3]}]

xv. set_property PACKAGE_PIN W15 [get_ports {B[0]}]

xvi. set_property IOSTANDARD LVCMOS33 [get_ports {B[0]}]
xvii. set_property PACKAGE_PIN V15 [get_ports {B[1]}]
xviii. set_property IOSTANDARD LVCMOS33 [get_ports {B[1]}]
xix. set_property PACKAGE_PIN W14 [get_ports {B[2]}]

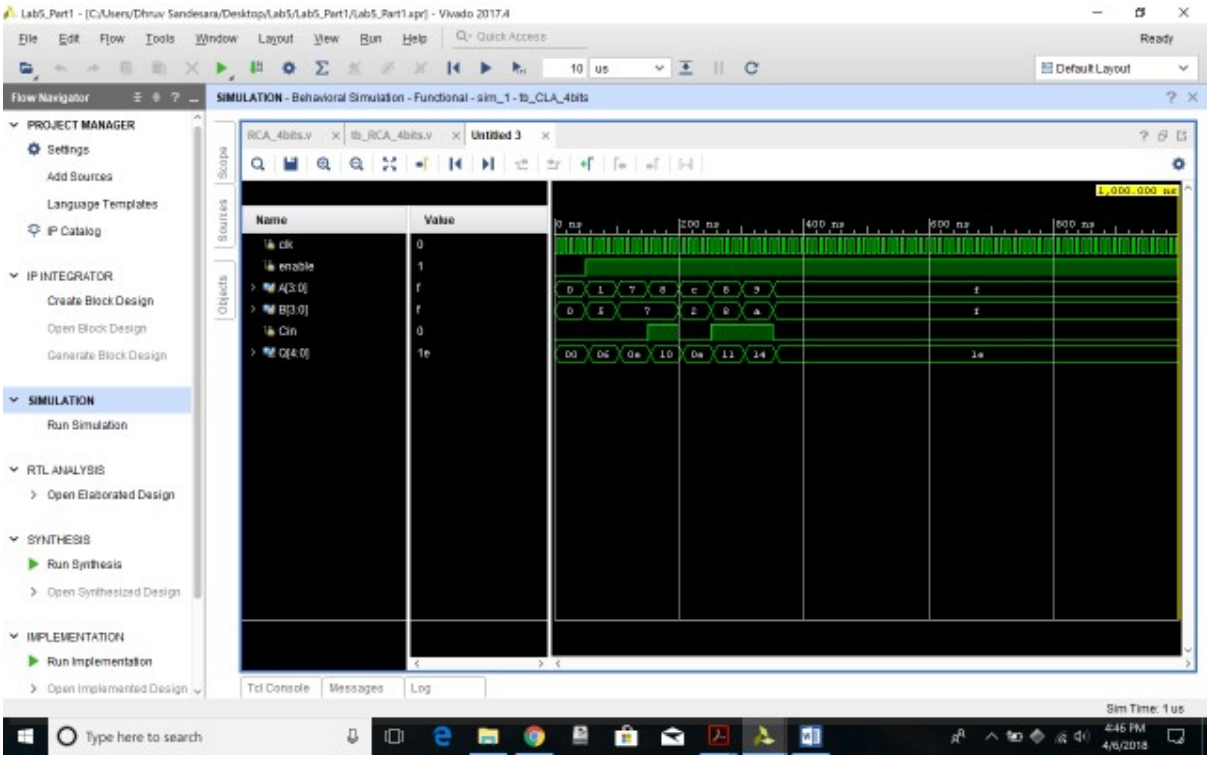
xx. set_property IOSTANDARD LVCMOS33 [get_ports {B[2]}]
xxi. set_property PACKAGE_PIN W13 [get_ports {B[3]}]

xxii. set_property IOSTANDARD LVCMOS33 [get_ports {B[3]}]
xxiii. set_property PACKAGE_PIN V2 [get_ports {Cin}]
xxiv. set_property IOSTANDARD LVCMOS33 [get_ports {Cin}]
xxv. set_property PACKAGE_PIN U16 [get_ports {Q[0]}]
xxvi. set_property IOSTANDARD LVCMOS33 [get_ports {Q[0]}]
xxvii. set_property PACKAGE_PIN E19 [get_ports {Q[1]}]
xxviii. set_property IOSTANDARD LVCMOS33 [get_ports {Q[1]}]
xxix. set_property PACKAGE_PIN U19 [get_ports {Q[2]}]
xxx. set_property IOSTANDARD LVCMOS33 [get_ports {Q[2]}]
xxxi. set_property PACKAGE_PIN V19 [get_ports {Q[3]}]
xxxii. set_property IOSTANDARD LVCMOS33 [get_ports {Q[3]}]
xxxiii. set_property PACKAGE_PIN W18 [get_ports {Q[4]}]

xxxiv. set_property IOSTANDARD LVCMOS33 [get_ports {Q[4]}]
xxxv. set_property PACKAGE_PIN U18 [get_ports enable]

xxxvi. set_property IOSTANDARD LVCMOS33 [get_ports enable]

xxxvii. Simulation waveform for the above test-cases



Part 2 –

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

```
xxxix.    assign P=A^B;
  xl.     assign G=A&B;
  xli.
  xlii.   assign C[0]= Cin;
  xliii.  assign C[1]= G[0]| P[0]&Cin;
  xliv.   assign C[2]= G[1]| P[1]&G[0]| P[1]&P[0]&Cin  ;
  xlv.    assign C[3]= G[2]| P[2]&G[1]| P[2]&P[1]&G[0]| P[2]&P[1]&P[0]&Cin ;
  xlvi.   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]&Cin;
  xlvii.
  xlviii. assign Data[0]= P[0]^C[0];
  xlix.   assign Data[1]= P[1]^C[1];
  l.      assign Data[2]= P[2]^C[2];
  li.     assign Data[3]= P[3]^C[3];
  lii.    assign Data[4]= C[4];
```

liii. Design files (.v) for the Carry Lookahead Adder and Register Logic

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 04/06/2018 03:51:09 PM
// Design Name:
// Module Name: CLA_4bits
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
```

```
//////////////////////////////////////////////////////////////////////////////////


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

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

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

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

   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]&Cin;


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

   assign Data[1]= P[1]^C[1];

   assign Data[2]= P[2]^C[2];

   assign Data[3]= P[3]^C[3];

   assign Data[4]= C[4];




    register_logic r0 (.clk(clk), .enable(enable), .Data(Data), .Q(Q));

endmodule


`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company:

// Engineer:

//

// Create Date: 04/06/2018 03:52:58 PM

// Design Name:
```

```
// Module Name: register_logic
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


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

```
    initial begin
    Q <= 0;
  end
  always @(posedge clk)
    if(enable)
      Q <= Data;


Endmodule
```

```
liv.   Test-bench
lv.    `timescale 1ns / 1ps
lvi.   //////////////////////////////////////////////////////////////////////////////////
       ///
lvii.  // Company:
lviii. // Engineer:
lix.   //
lx.    // Create Date: 04/06/2018 03:55:54 PM
lxi.   // Design Name:
lxii.  // Module Name: tb_CLA_4bits
lxiii. // Project Name:
lxiv.  // Target Devices:
lxv.   // Tool Versions:
lxvi.  // Description:
lxvii. //
lxviii. // Dependencies:
```

```
lxix.    //
lxx.     // Revision:
lxxi.    // Revision 0.01 - File Created
lxxii.   // Additional Comments:
lxxiii.  //
lxxiv.   //////////////////////////////////////////////////////////////////////////////////
         ///
lxxv.
lxxvi.
lxxvii.  module tb_CLA_4bits();
lxxviii.   reg clk;
lxxix.   reg enable;
lxxx.    reg [3:0] A;
lxxxi.   reg [3:0] B;
lxxxii.  reg Cin;
lxxxiii. wire [4:0] Q;
lxxxiv.
lxxxv.
lxxxvi.   CLA_4bits uut (
lxxxvii.     .clk(clk),
lxxxviii.    .enable(enable),
lxxxix.      .A(A),
xc.          .B(B),
xci.         .Cin(Cin),
xcii.        .Q(Q)
xciii.   );
xciv.    initial begin
xcv.
xcvi.    clk = 0;
xcvii.   enable = 0;
xcviii.  A = 4'b0000;
xcix.    B = 4'b0000;
c.       Cin = 0;
```

```
ci.
cii.    #50;
ciii.   enable = 1;
civ.    A = 4'b0000;
cv.     B = 4'b0101;
cvi.    Cin = 0;
cvii.
cviii.  #50;
cix.    A = 4'b0101;
cx.     B = 4'b0111;
cxi.    Cin = 0;
cxii.
cxiii.  #50;
cxiv.   A = 4'b1000;
cxv.    B = 4'b0111;
cxvi.   Cin = 1;
cxvii.
cxviii. #50;
cxix.   A = 4'b1001;
cxx.    B = 4'b0100;
cxxi.   Cin = 0;
cxxii.
cxxiii. #50;
cxxiv.  A = 4'b1000;
cxxv.   B = 4'b1000;
cxxvi.  Cin = 1;
cxxvii.
cxxviii. #50;
cxxix.  A = 4'b1101;
cxxx.   B = 4'b1010;
cxxxi.  Cin = 1;
cxxxii.
cxxxiii. #50;
```

cxxxiv. A = 4'b1110;
cxxxv. B = 4'b1111;
cxxxvi. Cin = 0;
cxxxvii.
cxxxviii. end
cxxxix.
cxl. always
cxli. #5 clk = ~clk;
cxlii.
cxliii.
cxliv.
cxlv. Endmodule

cxlvi. Complete Table 2 from the simulation

A[3:0]	B[3:0]	Cin	Sum[3:0]	Cout
0000	0101	0	5	0
0101	0111	0	C	0
1000	0111	1	0	1
1001	0100	0	D	0
1000	1000	1	1	1
1101	1010	1	8	1
1110	1111	0	D	1

Table 2. Testcases for Carry Lookahead Adder Verification

cxlvii. Constraints File (Just the uncommented portion)
cxlviii. ## Clock signal
cxlix. set_property PACKAGE_PIN W5 [get_ports clk]

cl. set_property IOSTANDARD LVCMOS33 [get_ports clk]
cli. create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]
clii.

cliii. ## Switches
cliv. set_property PACKAGE_PIN V17 [get_ports {A[0]}]
clv. set_property IOSTANDARD LVCMOS33 [get_ports {A[0]}]
clvi. set_property PACKAGE_PIN V16 [get_ports {A[1]}]
clvii. set_property IOSTANDARD LVCMOS33 [get_ports {A[1]}]
clviii. set_property PACKAGE_PIN W16 [get_ports {A[2]}]

clix. set_property IOSTANDARD LVCMOS33 [get_ports {A[2]}]
clx. set_property PACKAGE_PIN W17 [get_ports {A[3]}]

clxi. set_property IOSTANDARD LVCMOS33 [get_ports {A[3]}]
clxii. set_property PACKAGE_PIN W15 [get_ports {B[0]}]

clxiii. set_property IOSTANDARD LVCMOS33 [get_ports {B[0]}]
clxiv. set_property PACKAGE_PIN V15 [get_ports {B[1]}]
clxv. set_property IOSTANDARD LVCMOS33 [get_ports {B[1]}]
clxvi. set_property PACKAGE_PIN W14 [get_ports {B[2]}]

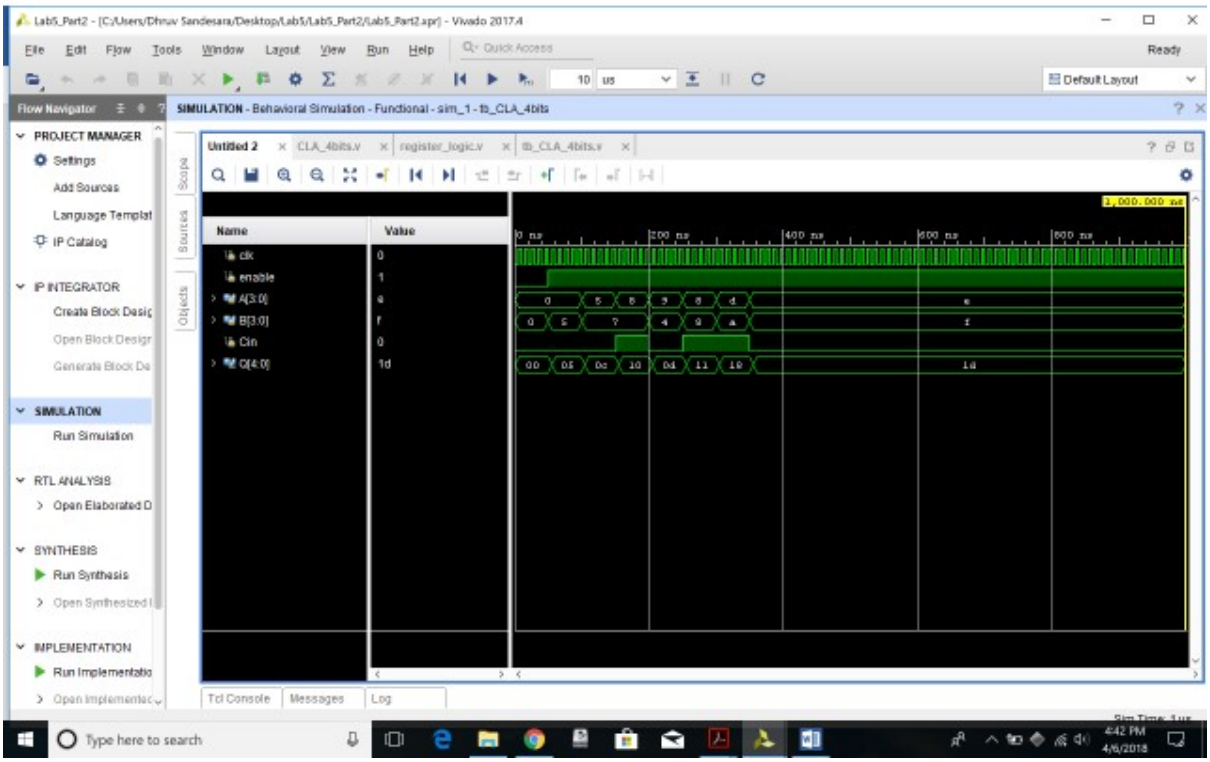
clxvii. set_property IOSTANDARD LVCMOS33 [get_ports {B[2]}]
clxviii. set_property PACKAGE_PIN W13 [get_ports {B[3]}]

clxix. set_property IOSTANDARD LVCMOS33 [get_ports {B[3]}]
clxx. set_property PACKAGE_PIN V2 [get_ports {Cin}]
clxxi. set_property IOSTANDARD LVCMOS33 [get_ports {Cin}]
clxxii. set_property PACKAGE_PIN U16 [get_ports {Q[0]}]
clxxiii. set_property IOSTANDARD LVCMOS33 [get_ports {Q[0]}]
clxxiv. set_property PACKAGE_PIN E19 [get_ports {Q[1]}]
clxxv. set_property IOSTANDARD LVCMOS33 [get_ports {Q[1]}]
clxxvi. set_property PACKAGE_PIN U19 [get_ports {Q[2]}]
clxxvii. set_property IOSTANDARD LVCMOS33 [get_ports {Q[2]}]
clxxviii. set_property PACKAGE_PIN V19 [get_ports {Q[3]}]
clxxix. set_property IOSTANDARD LVCMOS33 [get_ports {Q[3]}]
clxxx. set_property PACKAGE_PIN W18 [get_ports {Q[4]}]

clxxxi. set_property IOSTANDARD LVCMOS33 [get_ports {Q[4]}]
clxxxii. set_property PACKAGE_PIN U18 [get_ports enable]

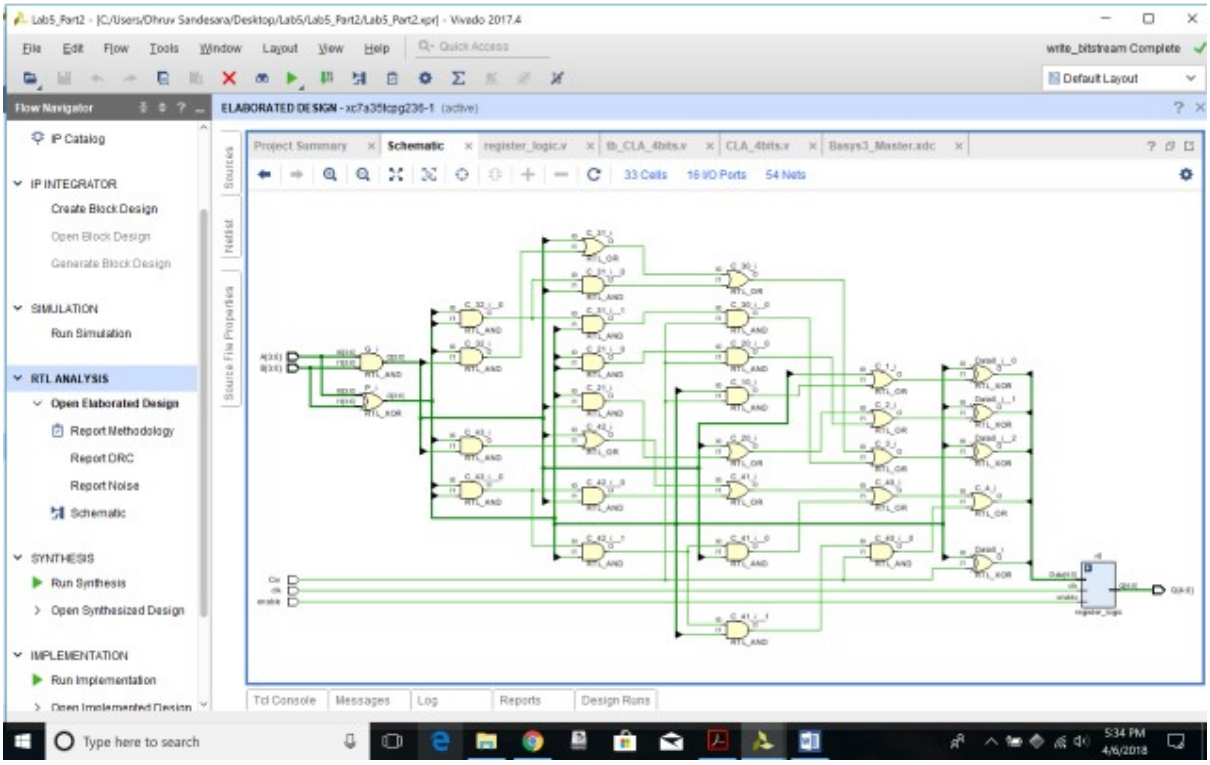
clxxxiii. set_property IOSTANDARD LVCMOS33 [get_ports enable]

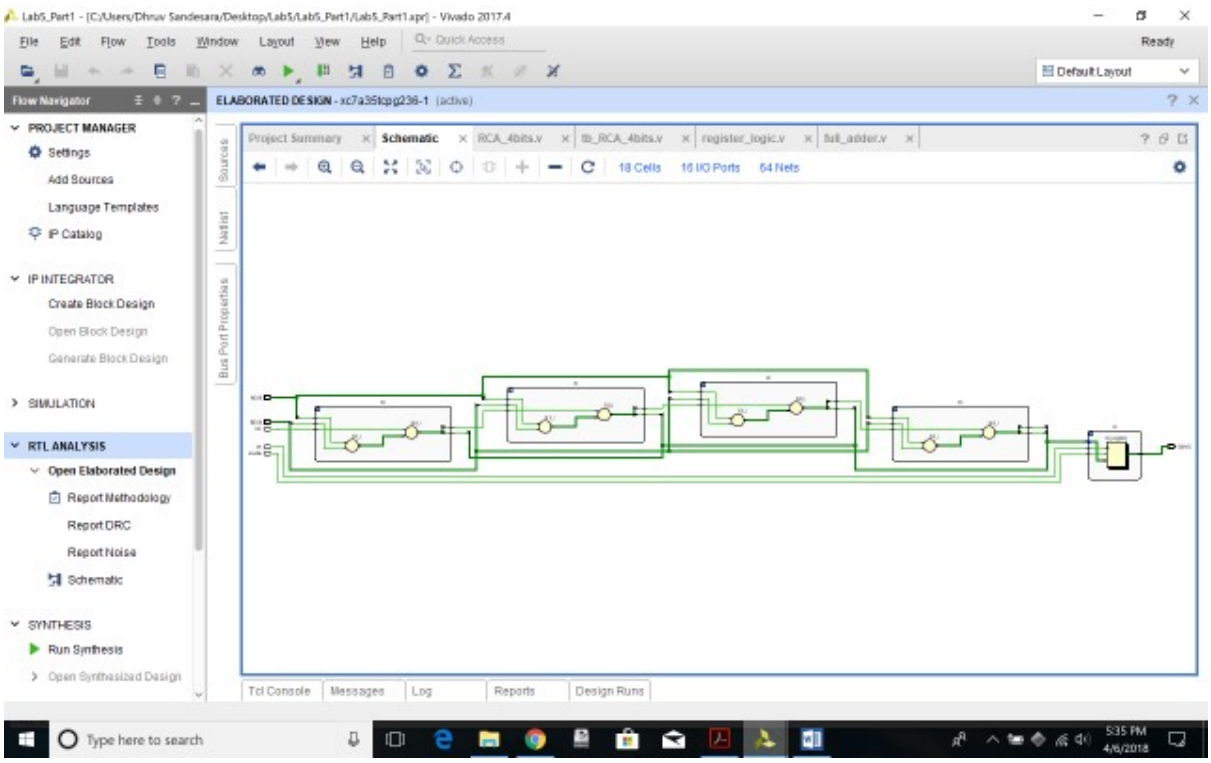
clxxxiv. Simulation waveform for the above test-cases



Part 3 –

clxxxv. Screenshots of the gate-level schematics for both the adder techniques


Lab5_Part2 - [C:/Users/Dhruv Sandesara/Desktop/Lab5/Lab5_Part2/Lab5_Part2.xpr] - Vivado 2017.4
File Edit Flow Tools Window Layout View Help
write_bitstream Complete
Default Layout
Flow Navigator
IP Catalog
IP INTEGRATOR
Create Block Design
Open Block Design
Generate Block Design
SIMULATION
Run Simulation
RTL ANALYSIS
Open Elaborated Design
Report Methodology
Report DRC
Report Noise
Schematic
SYNTHESIS
Run Synthesis
Open Synthesized Design
IMPLEMENTATION
Run Implementation
ELABORATED DESIGN - xc7a35tcpg236-1 (active)
Project Summary
Schematic
register_logic.v
tb_CLA_4bits.v
CLA_4bits.v
Basys3_Master.xdc
33 Cells
16 I/O Ports
54 Nets
Sources
Netlist
Source File Properties
Tcl Console
Messages
Log
Reports
Design Runs
Type here to search
5:34 PM
4/6/2018



clxxxvi. Delay and area for both the adder techniques showing all the work

clxxxvii. Gate Delay (ns) Area

clxxxviii. XOR 3 6

clxxxix. AND 3 4

cxc. OR 2 4

cxci. Table 3. Sample Delay and Area Values for Various

Part 2 :

18 And Gates

5 XOR Gates

10 OR Gates

Area is 18*4 + 5*6 + 10*4= 142

Max Path is 6 And&XOR gates deep which is 6*3 is 18ns

Part 1:

1 Full Adder has : 2* XOR, 2* And, and 1 OR;

MaX Delay per Adder is 1 XOR, 1AND, And 1 OR.

There fore

AREa is 2*6+2*4+4= 24

Delay is 3+3+2= 8

As we use 4 adders we get

Area is 4*24=96

Delay is 4*8= 32ns;

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

Carry Ahead Looker: Pros is that it is fast due to minimal gate delay. Cons is that there are way too many gates used

Carry Ripple Ahead: Pros is that it uses less gates. Cons is that there is a O(n) gate delay as the MSB has to wait on the Carry bits from the lesser significat bits.

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.*