ES204 Digital Systems
LAB Assignment - 4 (A2 only)

Indian Institute of Technology, Gandhinagar
February 7, 2024

Submission deadline: Feb 10, 2024, 8 PM
Marks: 50

General Instructions: Each team will issue the Basys-3 board from their respective TAs after checking that they are working properly. When you return the boards, the TAs will check them. If they are not working, you'll need to replace them. So, please use the boards carefully.

Submission instructions:

(a) Only one student from the team will submit the Word doc with the name Rollno1_Rollno2.pdf. The PDF will contain the code, testbench, the XDC file, and simulation results.
(b) Make a tar-ball / Zip of the project and upload it.

For each of the questions, write a Verilog code. You also need to create a **testbench** and show the simulation results. Lastly, you will perform IO planning and show its working on the FPGA board.

Important Note: Up until simulation, make an 8-bit design, and to show it on FPGA, make a 4-bit design. You may make a parametrized code if you are enthusiastic about it. However, writing parameterized code is not compulsory for this lab.

Question: Write a program to design the following three adders and report each design's LUT utilization, static power, and dynamic power. Also, write a detailed explanation for the results observed. Feel free to use the adder code given in the lecture slides.

1. Ripple Carry Adder 2. Carry Lookahead Adder 3. Carry Select Adder

Lab assignment 4 ES 204

Birudugadda Srivibhav (22110050)
Reddybathuni Venkat (22110220)

#Implementation code of Ripple Carry Adder

1. Code

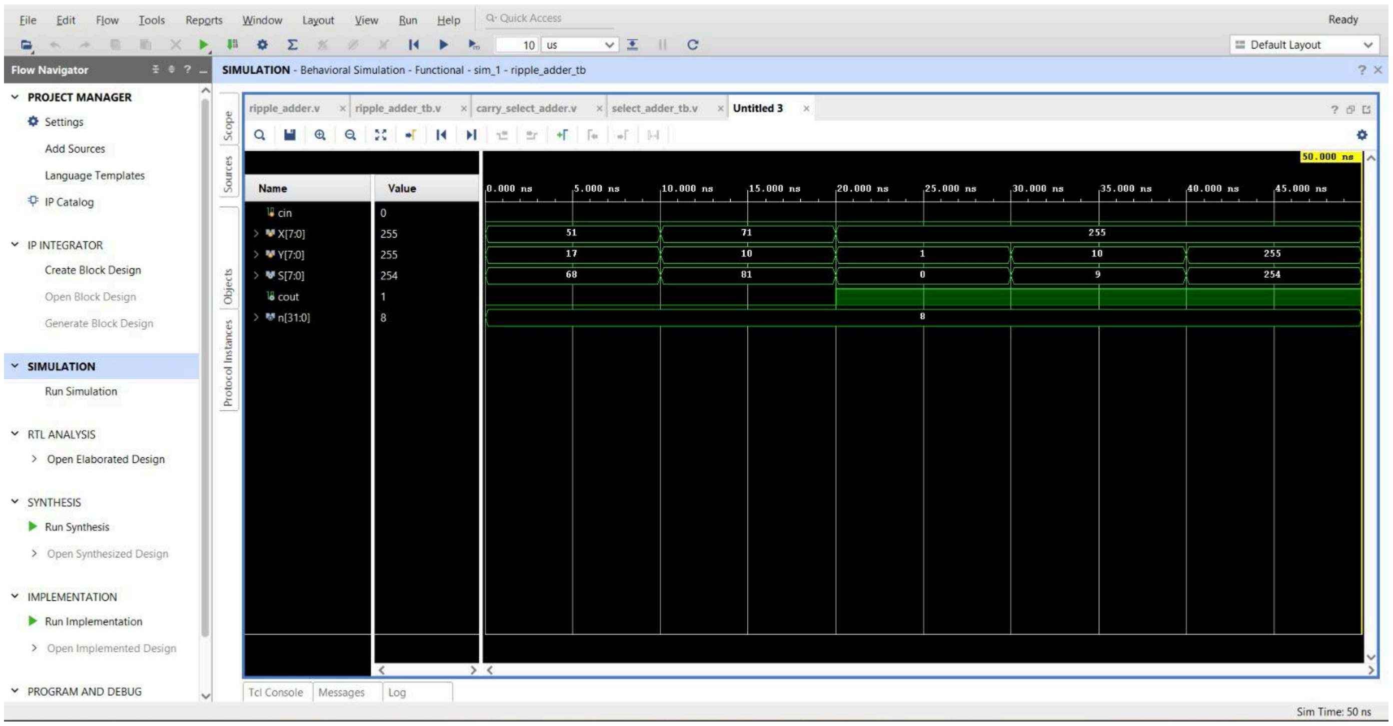
```
`timescale 1ns / 1ps
module ripple_adder(cin,X,Y,S,cout);
   parameter n = 8;
   input cin;
   input [n-1:0] X, Y;
   output reg [n-1:0] S;
   output reg cout;
   reg [n:0] C;
   integer k;
   always@(*)
   begin
   C[0] = cin;
   for (k=0; k<n; k=k+1)
   begin
   S[k] = X[k] ^ Y[k] ^ C[k];
   C[k+1] = (X[k] & Y[k] ) | (X[k] & C[k] ) | (C[k] & Y[k]);
   end
   cout = C[n];
   end

endmodule
```

2. TestBench

```
`timescale 1ns / 1ps
module ripple_adder_tb();
   parameter n = 8;
   reg cin;
   reg [n-1:0] X, Y;
   wire [n-1:0] S;
   wire cout;
   ripple_adder uut(.cin(cin), .X(X), .Y(Y), .S(S), .cout(cout));
   initial
   begin
   cin = 0; X = 51; Y = 17;
   #10;
```

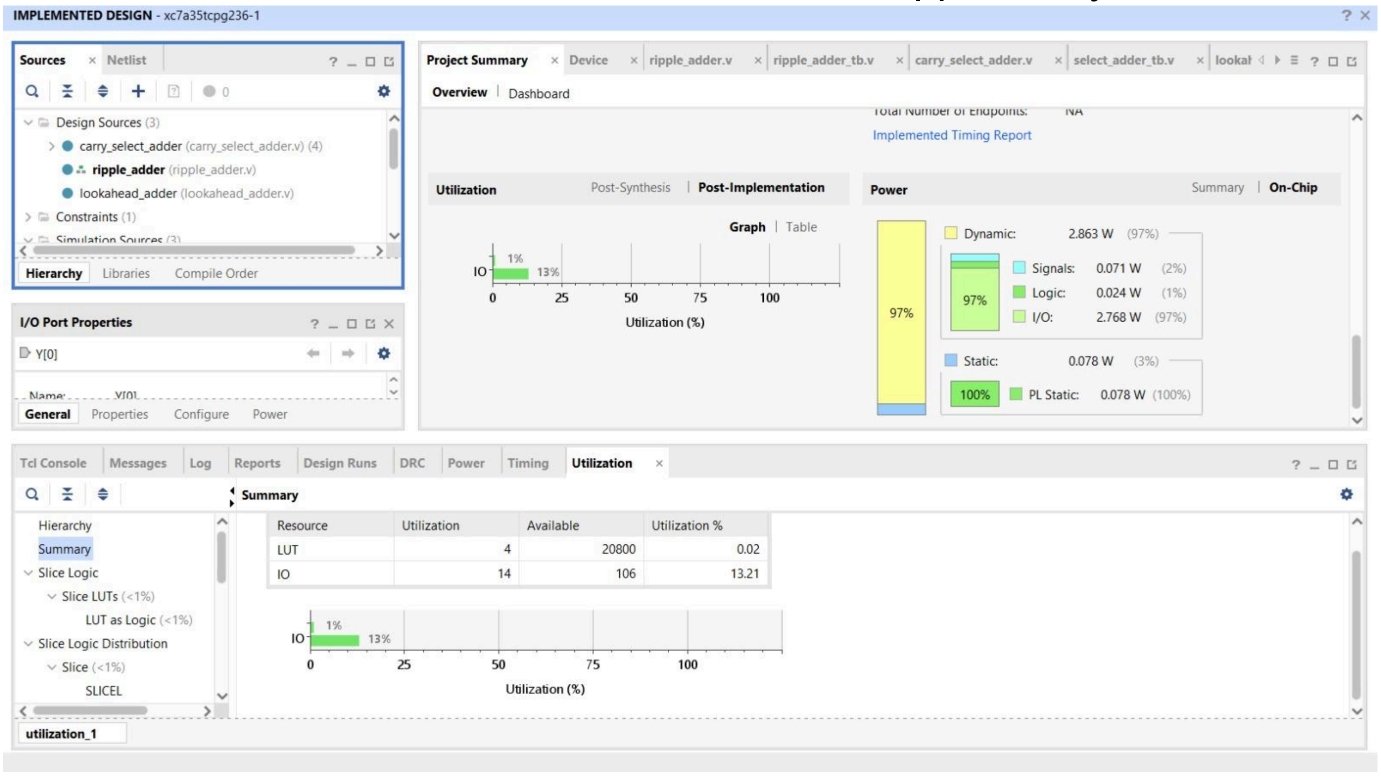
```
    cin = 0; X = 71; Y = 10;
    #10;
    cin = 0; X = 255; Y = 1;
    #10;
    cin = 0; X = 255; Y = 10;
    #10;
    cin = 0; X = 255; Y = 255;
    #10;
    $finish();
    end
endmodule
```

3. Simulation Results



Name	Value					
cin	0					
X[7:0]	255	51	71	255		
Y[7:0]	255	17	10	1	10	255
S[7:0]	254	68	81	0	9	254
cout	1					
n[31:0]	8	8				

4. LUT Utilization and Power Information of Ripple Carry Adder



IMPLEMENTED DESIGN - xc7a35tcpg236-1

Sources | Netlist

- Design Sources (3)
  - carry_select_adder (carry_select_adder.v) (4)
  - ripple_adder (ripple_adder.v)
  - lookahead_adder (lookahead_adder.v)
- Constraints (1)
- Simulation Sources (3)

Hierarchy | Libraries | Compile Order

I/O Port Properties

Y[0]

Name: Y[0]

General | Properties | Configure | Power

Project Summary | Device | ripple_adder.v | ripple_adder_tb.v | carry_select_adder.v | select_adder_tb.v | lookal

Overview | Dashboard

Total Number of Endpoints: NA

Implemented Timing Report

Utilization — Post-Synthesis | Post-Implementation — Graph | Table

IO: 1%, 13%

Utilization (%)

Power — Summary | On-Chip

Dynamic:	2.863 W	(97%)
Signals:	0.071 W	(2%)
Logic:	0.024 W	(1%)
I/O:	2.768 W	(97%)
Static:	0.078 W	(3%)
PL Static:	0.078 W	(100%)

Tcl Console | Messages | Log | Reports | Design Runs | DRC | Power | Timing | Utilization

- Hierarchy
- Summary
- Slice Logic
  - Slice LUTs (<1%)
    - LUT as Logic (<1%)
- Slice Logic Distribution
  - Slice (<1%)
    - SLICEL

Summary

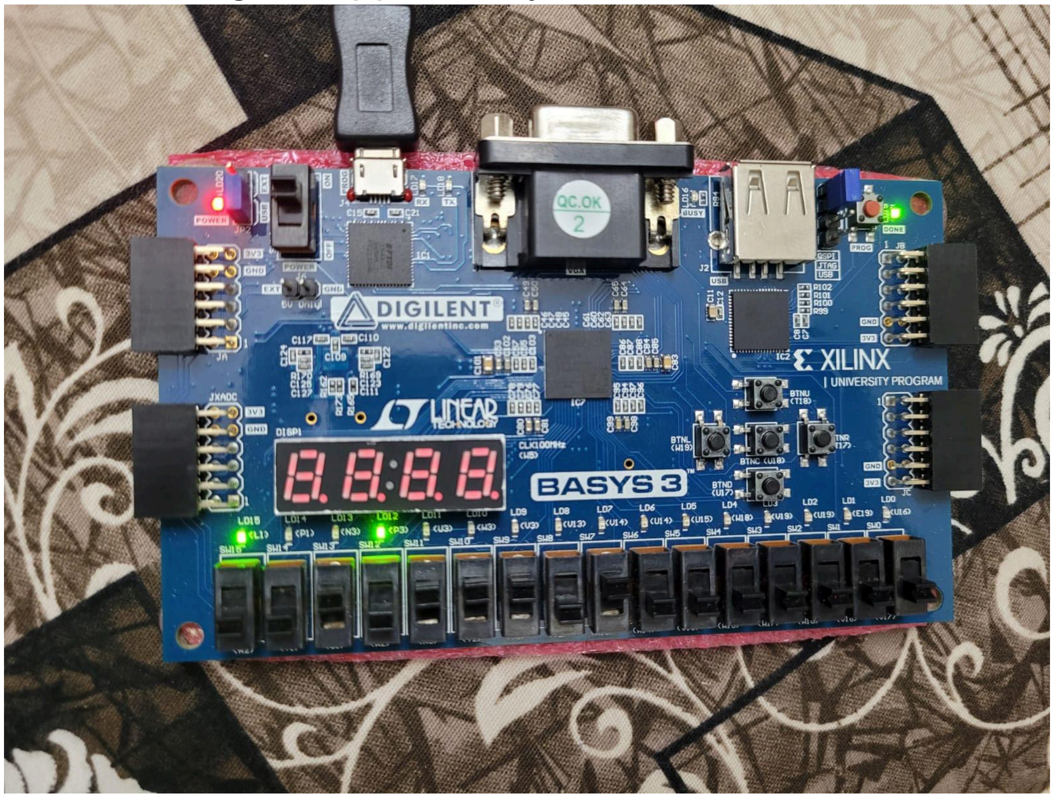
Resource	Utilization	Available	Utilization %
LUT	4	20800	0.02
IO	14	106	13.21

IO: 1%, 13%

Utilization (%)

utilization_1

5. Working of Ripple Carry Adder



#Implementation code of Carry Lookahead Adder

1. Code

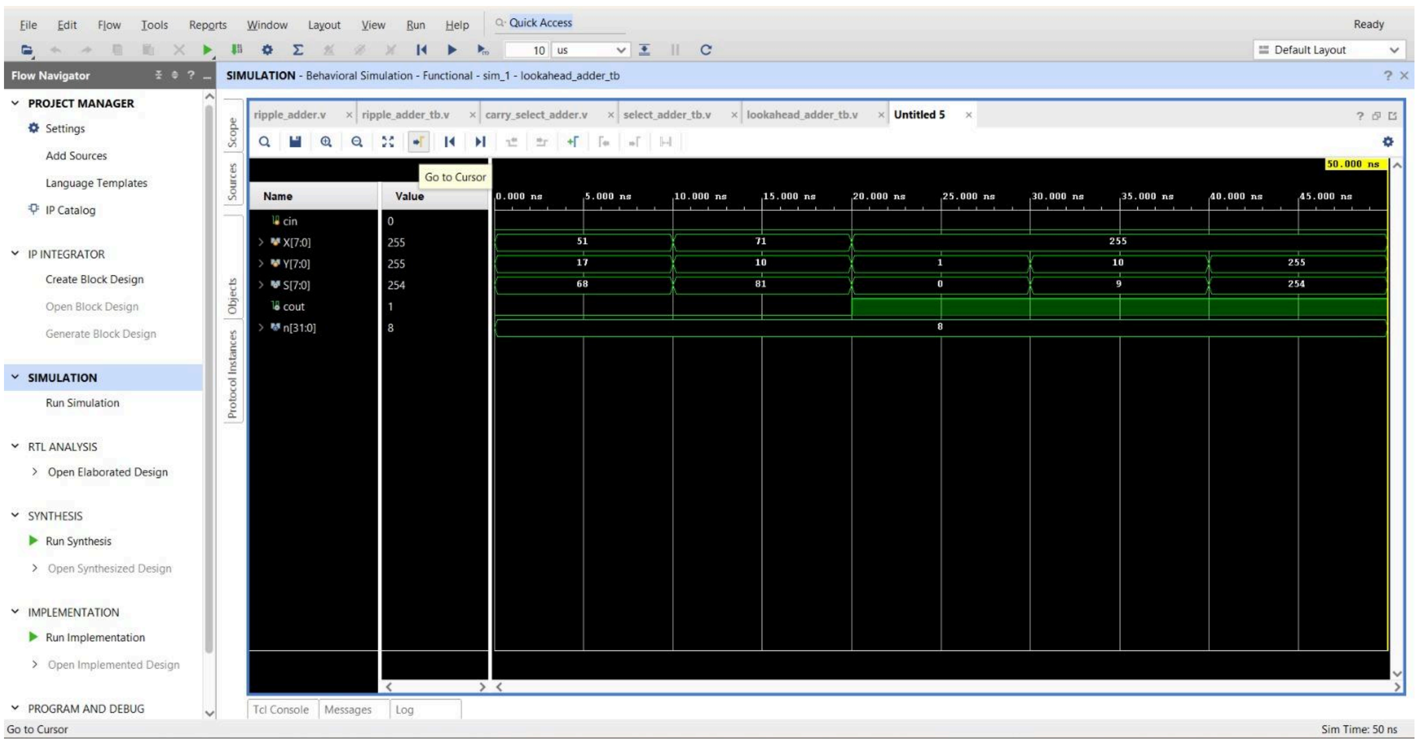
```
`timescale 1ns / 1ps
module lookahead_adder(cin,X,Y,S,cout);
   parameter n = 8;
   input cin;
   input [n-1:0] X, Y;
   output reg [n-1:0] S;
   output reg cout;
   reg [n:0] C;
   reg [n:0] G, P;
   integer k;
   always@(*)
   begin
   C[0] = cin;
   for (k = 0; k < n; k = k +1)
   begin
      G[k] = X[k] & Y[k];
      P[k] = X[k] ^ Y[k];
      S[k] = P[k] ^ C[k];
      C[k+1] = G[k] | (P[k] & C[k]);
   end
    cout = C[n];
   end
endmodule
```

2. TestBench

```
`timescale 1ns / 1ps
module lookahead_adder_tb();
   parameter n = 8;
   reg cin;
   reg [n-1:0] X, Y;
   wire [n-1:0] S;
   wire cout;
   lookahead_adder uut(.cin(cin), .X(X), .Y(Y), .S(S), .cout(cout));
   initial
   begin
   cin = 0; X = 51; Y = 17;
   #10;
   cin = 0; X = 71; Y = 10;
   #10;
   cin = 0; X = 255; Y = 1;
```

```
    #10;
    cin = 0; X = 255; Y = 10;
    #10;
    cin = 0; X = 255; Y = 256;
    #10;
    $finish();
    end
Endmodule
```

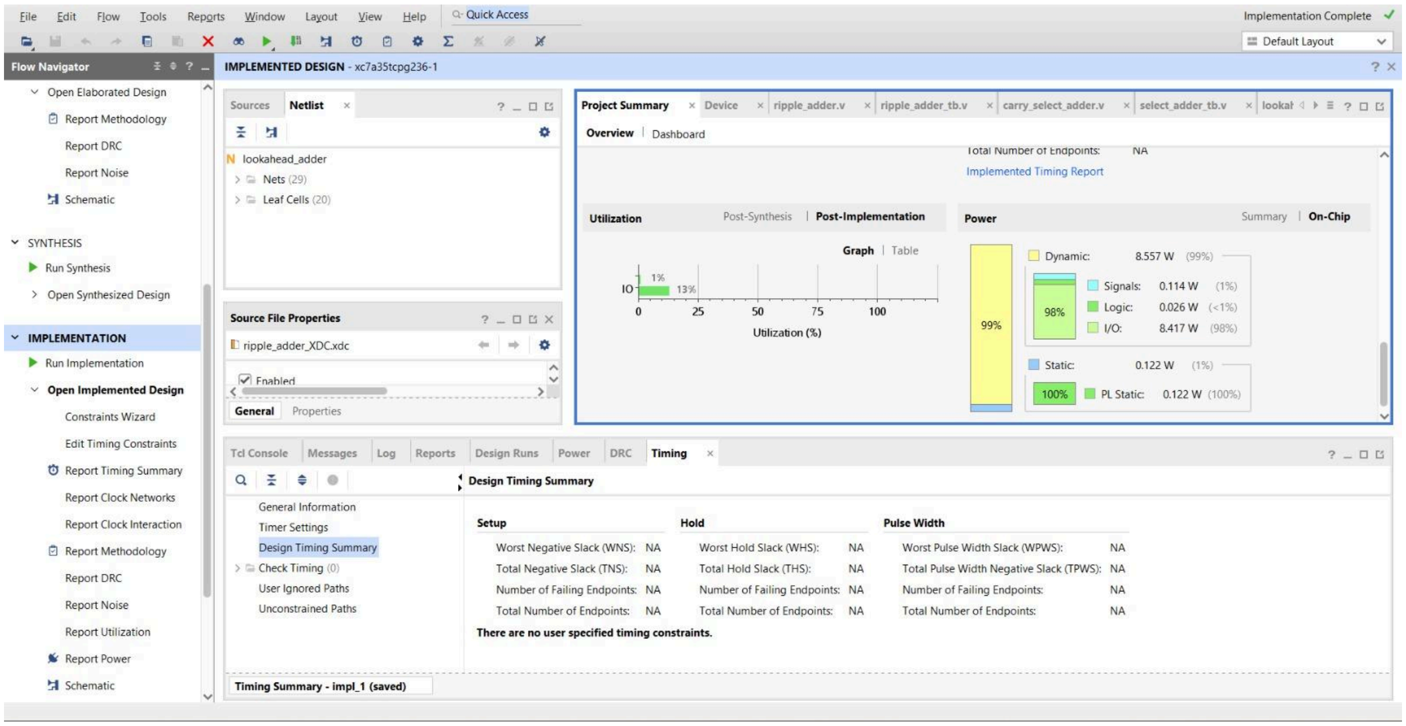
3. Simulation Results



SIMULATION - Behavioral Simulation - Functional - sim_1 - lookahead_adder_tb

Name	Value
cin	0
X[7:0]	255
Y[7:0]	255
S[7:0]	254
cout	1
n[31:0]	8

4. LUT Utilization and Power Information of Carry Lookahead Adder


IMPLEMENTED DESIGN - xc7a35tcpg236-1

Project Summary | Overview | Dashboard

Total Number of Endpoints: NA

Implemented Timing Report

Utilization Post-Synthesis | **Post-Implementation**

Graph | Table

IO: 1%, 13%

Power Summary | **On-Chip**

Dynamic:	8.557 W	(99%)
Signals:	0.114 W	(1%)
Logic:	0.026 W	(<1%)
I/O:	8.417 W	(98%)
Static:	0.122 W	(1%)
PL Static:	0.122 W	(100%)

Netlist: lookahead_adder — Nets (29), Leaf Cells (20)

Source File Properties: ripple_adder_XDC.xdc

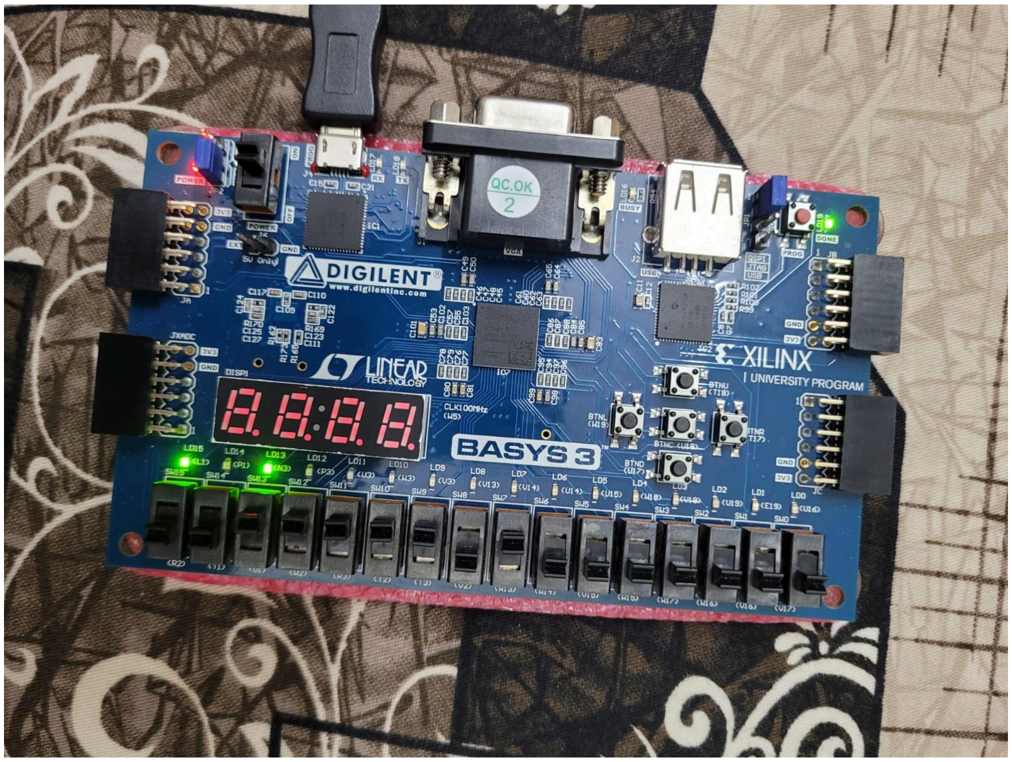
Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	NA	Worst Hold Slack (WHS):	NA	Worst Pulse Width Slack (WPWS):	NA
Total Negative Slack (TNS):	NA	Total Hold Slack (THS):	NA	Total Pulse Width Negative Slack (TPWS):	NA
Number of Failing Endpoints:	NA	Number of Failing Endpoints:	NA	Number of Failing Endpoints:	NA
Total Number of Endpoints:	NA	Total Number of Endpoints:	NA	Total Number of Endpoints:	NA

There are no user specified timing constraints.

Timing Summary - impl_1 (saved)

5. Working of Carry Lookahead Adder



#Implementation code of Carry Select Adder

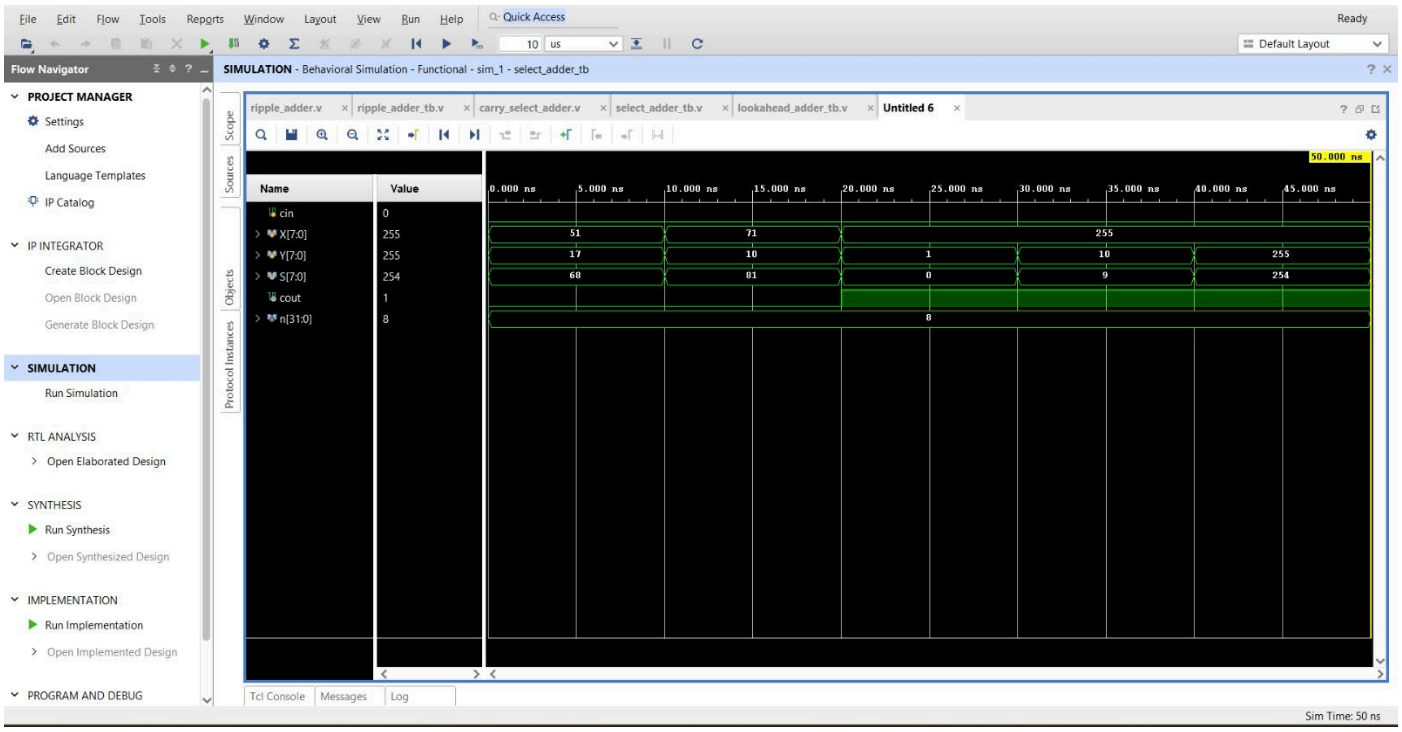
1. Code

```
`timescale 1ns / 1ps
module ripple_adder_4bit(cin,X,Y,S,cout);
   parameter n = 4;
   input cin;
   input [n-1:0] X, Y;
   output reg [n-1:0] S;
   output reg cout;
   reg [n:0] C;
   integer k;
   always@(*)
   begin
   C[0] = cin;
   for (k=0; k<n; k=k+1)
   begin
   S[k] = X[k] ^ Y[k] ^ C[k];
   C[k+1] = (X[k] & Y[k] ) | (X[k] & C[k] ) | (C[k] & Y[k]);
   end
   cout = C[n];
   end
endmodule
module multiplexer2to1(X, Y, select, OUT);
   parameter n = 4;
   input [n-1:0] X,Y;
   input select;
   output reg [n-1:0] OUT;
   always@(*)
   begin
   if(!select)
   OUT = X;
   else
   OUT = Y;
   end
endmodule
module carry_select_adder(cin,X,Y,S,cout);
   parameter n = 8;
   input cin;
   input [n-1:0] X, Y;
   output [n-1:0] S;
   output cout;
   wire c;
   ripple_adder_4bit inst1(.cin(cin), .X(X[n/2 -1:0]), .Y(Y[n/2 -1:0]), .S(S[n/2 -1:0]), .cout(c));
   wire [n/2-1:0] S0, S1;
   wire cout0,cout1;
   ripple_adder_4bit inst2(.cin(0), .X(X[n-1 :n/2]), .Y(Y[n-1 :n/2]), .S(S0[n/2-1:0]), .cout(cout0));
   ripple_adder_4bit inst3(.cin(1), .X(X[n-1 :n/2]), .Y(Y[n-1 :n/2]), .S(S1[n/2-1:0]), .cout(cout1));
   multiplexer2to1 inst4 (.X(S0), .Y(S1), .select(c), .OUT(S[n-1 :n/2]));
   and i1(I1, ~c, cout0);
   and i2(I2, c, cout1);
   or i3(cout, I1,I2);
endmodule
```

2. TestBench

```
`timescale 1ns / 1ps
module select_adder_tb();
parameter n = 8;
   reg cin;
   reg [n-1:0] X, Y;
   wire [n-1:0] S;
   wire cout;
   carry_select_adder uut(.cin(cin), .X(X), .Y(Y), .S(S), .cout(cout));
   initial
   begin
   cin = 0; X = 51; Y = 17;
   #10;
   cin = 0; X = 71; Y = 10;
   #10;
   cin = 0; X = 255; Y = 1;
   #10;
   cin = 0; X = 255; Y = 10;
   #10;
   cin = 0; X = 255; Y = 256;
   #10;
   $finish();
   end
Endmodule
```

3. Simulation Results



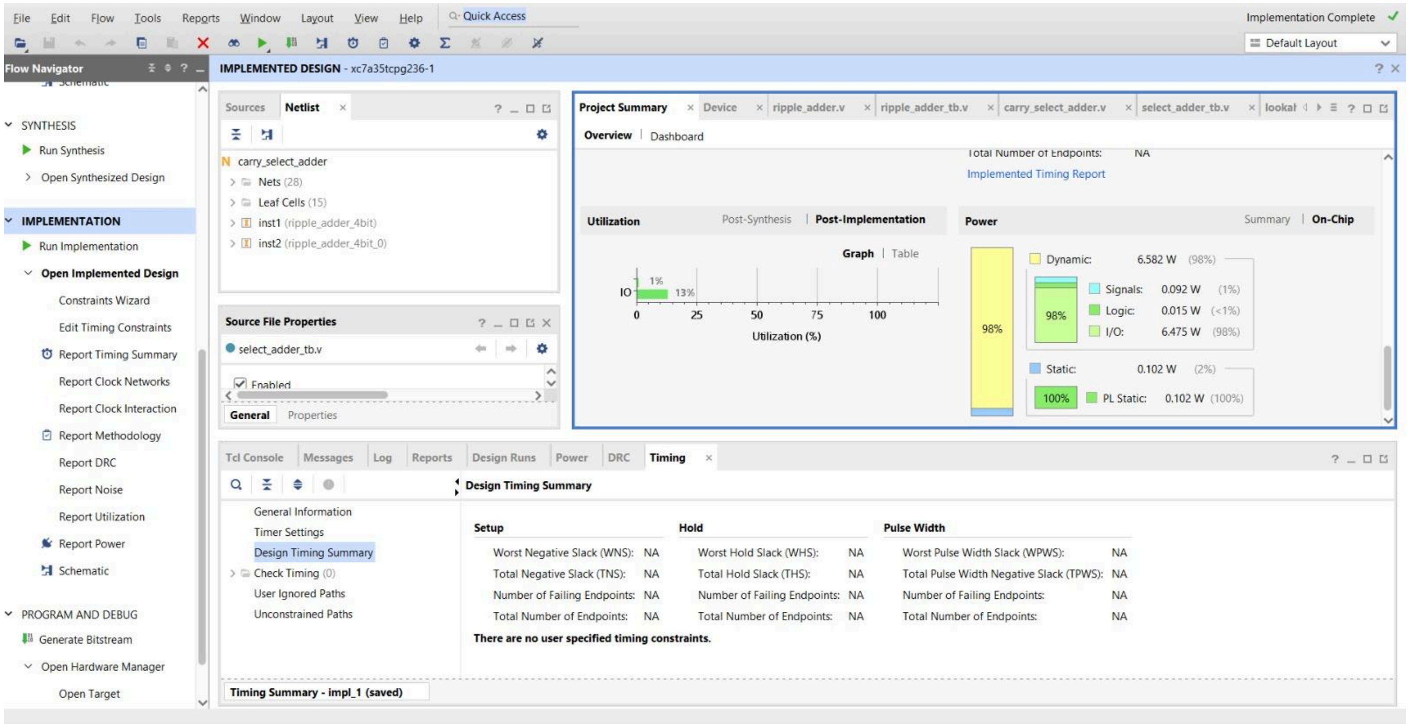
SIMULATION - Behavioral Simulation - Functional - sim_1 - select_adder_tb

Name	Value
cin	0
X[7:0]	255
Y[7:0]	255
S[7:0]	254
cout	1
n[31:0]	8

```
         0.000 ns  5.000 ns  10.000 ns  15.000 ns  20.000 ns  25.000 ns  30.000 ns  35.000 ns  40.000 ns  45.000 ns   50.000 ns
X[7:0]        51                  71                                  255
Y[7:0]        17                  10                  1                    10                   255
S[7:0]        68                  81                  0                    9                    254
n[31:0]                                               8
```

Sim Time: 50 ns

4. LUT Utilization and Power Information of Carry Select Adder



IMPLEMENTED DESIGN - xc7a35tcpg236-1

carry_select_adder
- Nets (28)
- Leaf Cells (15)
- inst1 (ripple_adder_4bit)
- inst2 (ripple_adder_4bit_0)

select_adder_tb.v

Total Number of Endpoints: NA

Implemented Timing Report

Utilization — Post-Synthesis | Post-Implementation — Graph | Table

IO: 1% / 13%

0 25 50 75 100

Utilization (%)

Power — Summary | On-Chip

Dynamic:	6.582 W	(98%)
Signals:	0.092 W	(1%)
Logic:	0.015 W	(<1%)
I/O:	6.475 W	(98%)
Static:	0.102 W	(2%)
PL Static:	0.102 W	(100%)

98%

98%

100%

Design Timing Summary

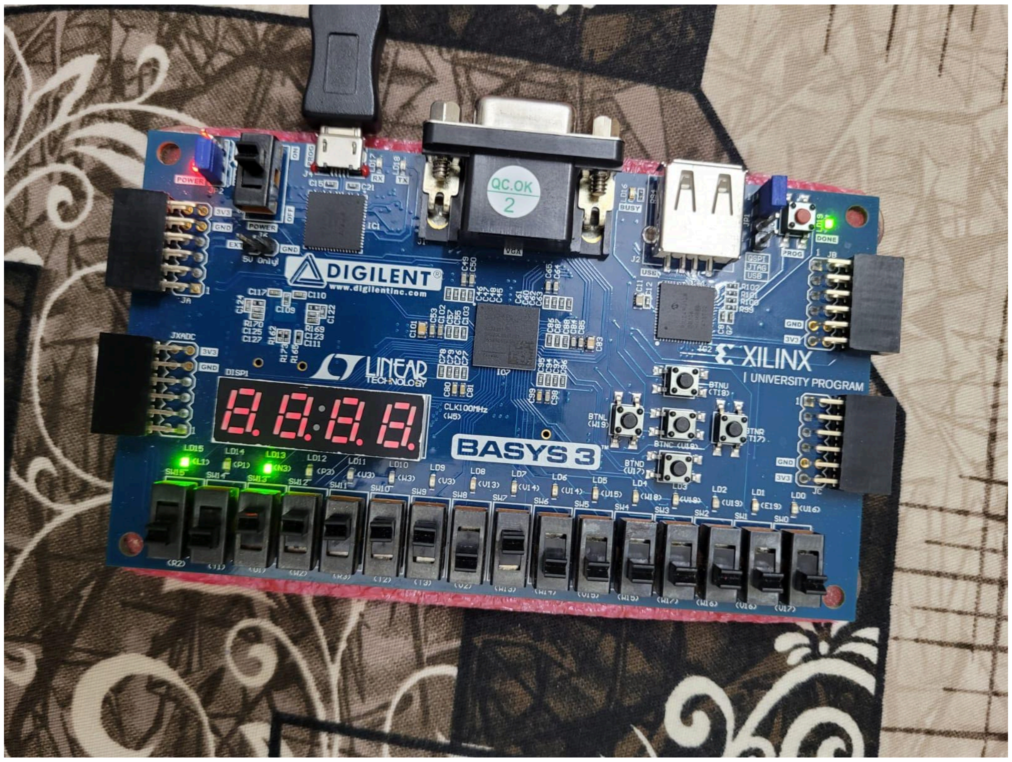
- General Information
- Timer Settings
- Design Timing Summary
- Check Timing (0)
- User Ignored Paths
- Unconstrained Paths

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	NA	Worst Hold Slack (WHS):	NA	Worst Pulse Width Slack (WPWS):	NA
Total Negative Slack (TNS):	NA	Total Hold Slack (THS):	NA	Total Pulse Width Negative Slack (TPWS):	NA
Number of Failing Endpoints:	NA	Number of Failing Endpoints:	NA	Number of Failing Endpoints:	NA
Total Number of Endpoints:	NA	Total Number of Endpoints:	NA	Total Number of Endpoints:	NA

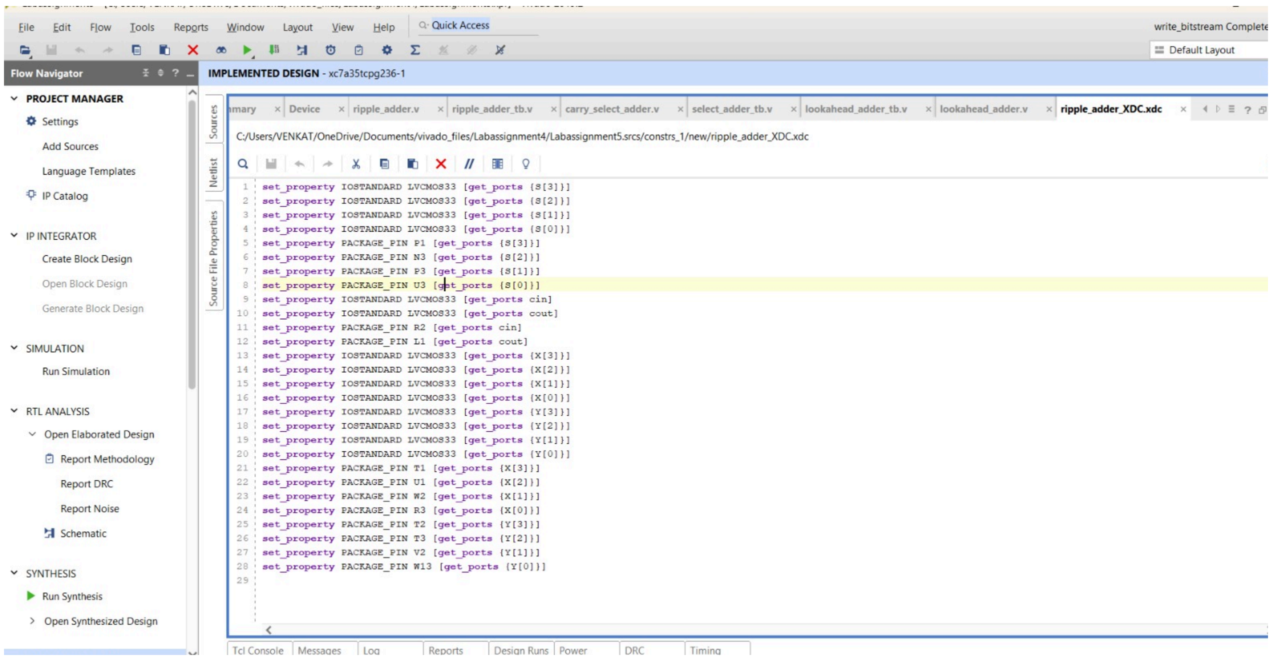
There are no user specified timing constraints.

Timing Summary - impl_1 (saved)

5. Working of Carry Select Adder



XDC File:



C:/Users/VENKAT/OneDrive/Documents/vivado_files/Labassignment4/Labassignment5.srcs/constrs_1/new/ripple_adder_XDC.xdc

```
set_property IOSTANDARD LVCMOS33 [get_ports {S[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {S[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {S[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {S[0]}]
set_property PACKAGE_PIN P1 [get_ports {S[3]}]
set_property PACKAGE_PIN N3 [get_ports {S[2]}]
set_property PACKAGE_PIN P3 [get_ports {S[1]}]
set_property PACKAGE_PIN U3 [get_ports {S[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports cin]
set_property IOSTANDARD LVCMOS33 [get_ports cout]
set_property PACKAGE_PIN R2 [get_ports cin]
set_property PACKAGE_PIN L1 [get_ports cout]
set_property IOSTANDARD LVCMOS33 [get_ports {X[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {X[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {X[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {X[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Y[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Y[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Y[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Y[0]}]
set_property PACKAGE_PIN T1 [get_ports {X[3]}]
set_property PACKAGE_PIN U1 [get_ports {X[2]}]
set_property PACKAGE_PIN W2 [get_ports {X[1]}]
set_property PACKAGE_PIN R3 [get_ports {X[0]}]
set_property PACKAGE_PIN T2 [get_ports {Y[3]}]
set_property PACKAGE_PIN T3 [get_ports {Y[2]}]
set_property PACKAGE_PIN V2 [get_ports {Y[1]}]
set_property PACKAGE_PIN W13 [get_ports {Y[0]}]
```

Observation:

The power consumption diagram shows that the Ripple carry adder consumes less power, followed by the select adder and lookahead adder in that order. This means ripple carry adder is better at power consumption than the rest.

Select adder takes more power than ripple adder as we use three ripple 4-bit adders instead of 1 8-bit ripple adder. But consuming more power makes it give quicker results than that of ripple adder.

In the lookahead adder, we evaluate multiple values for each k-value, which significantly increases power consumption.