VLSI LAB Digital Assignment 6

Submitted by:

Swarup Tripathy — 19BEE0167

School of Electrical Engineering

Faculty: **Professor Balamurugan S**

Course: **EEE-4028**

Course Name: **VLSI Lab**

Lab Slot: **L43 + L44**

This work is submitted in partial fulfilment of the requirement of the award of the degree of Bachelor of Technology in EEE

April 21, 2022

4-BIT and 5-BIT BINARY SQUARER

Objectives

1. To provide students with the background needed to design, develop, and test digital arithmetic circuits using IEEE standard Verilog HDL.
2. To provide an understanding complex arithmetic circuit design principles and its architecture design.

Outcomes

1. After completion of this course the students will be familiar with design and implementation of Digital Arithmetic building blocks using Verilog HDL and Modelsim Software.

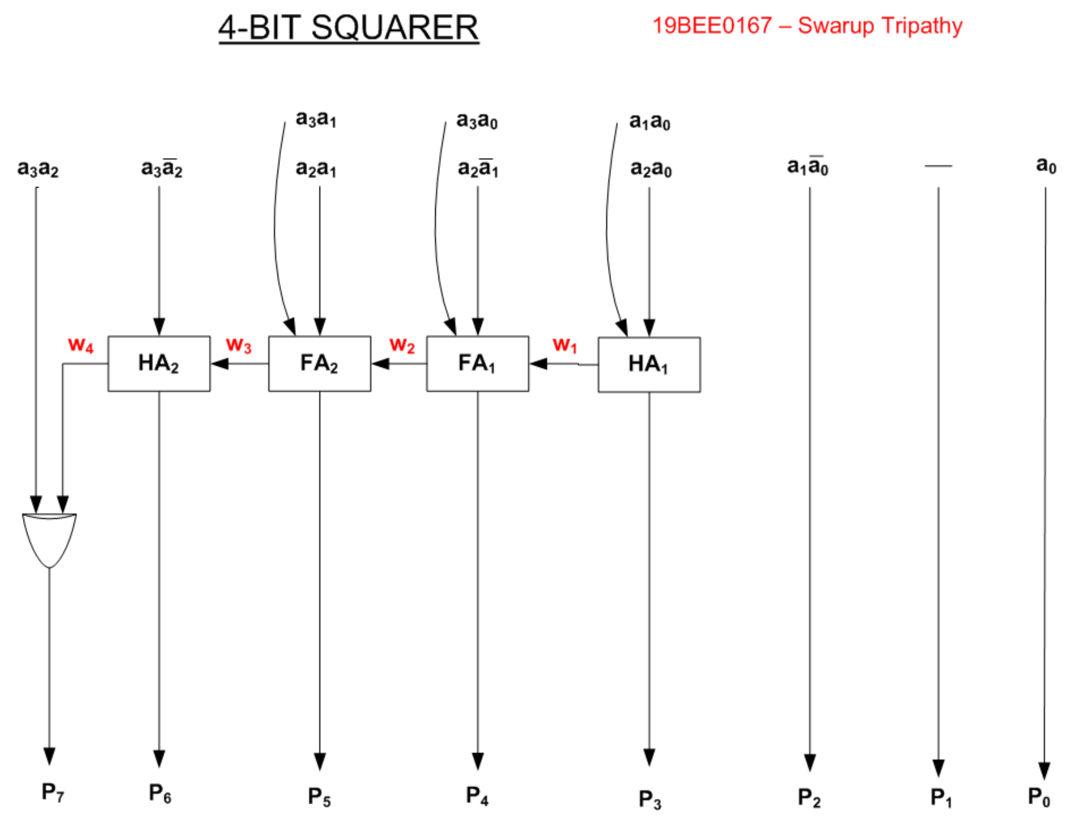
AIM

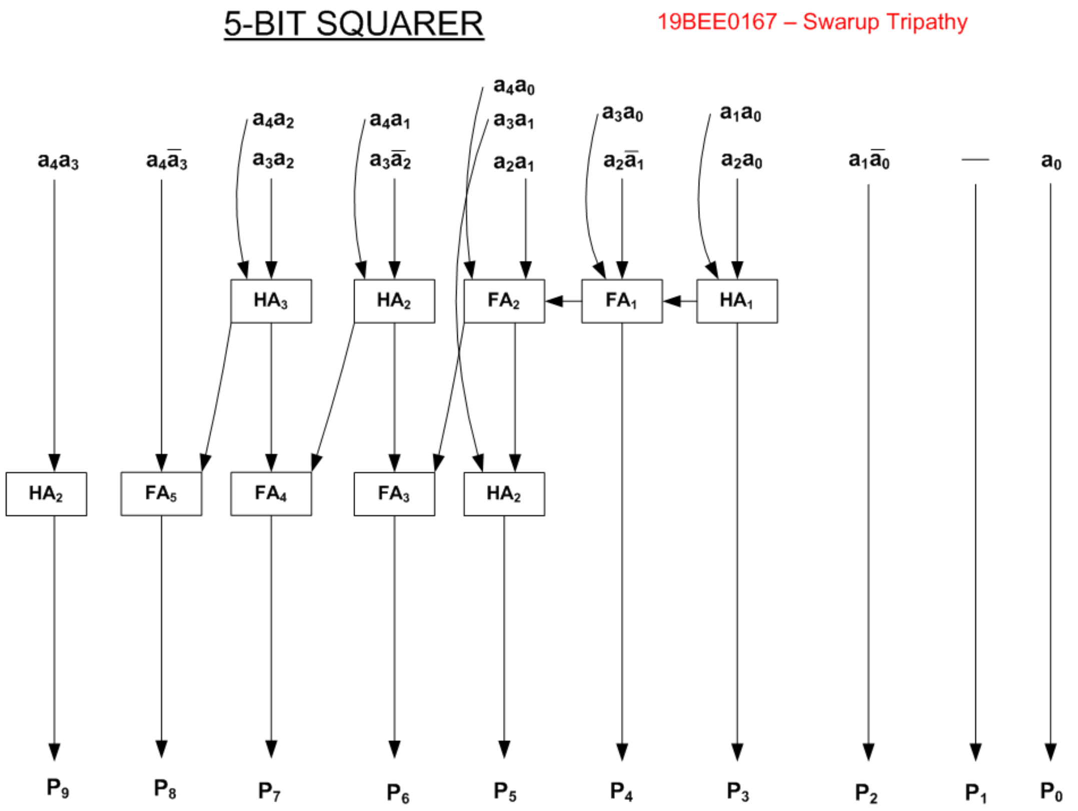
1. Design a 4bit and 5bit binary squarer using full adders and half adders.

REQUIRED SOFTWARE

1. Model sim software for simulation
2. Microsoft Visio for making flowchart
3. Documentation to be done using LaTeX

CIRCUIT DIAGRAM





Design Code for 5bit

1. **Verilog Module** — designcode.v

```
module squarer_5bit_19BEE0167(input[4:0]a,
    output[9:0]p);
wire[7:0]w;

assign p[0]=a[0];
assign p[1]=0;
assign p[2]=(a[1]&~(a[0]));

ha_df_19BEE0167 ha1((a[2]&a[0]),(a[1]&a[0]),p[3],w[1]);
fa_df_19BEE0167 fa1((a[3]&a[0]),(a[2]&~(a[1])),w[1],p[4],w[2]);
fa_df_19BEE0167 fa2((a[4]&a[0]),(a[3]&a[1]),w[2],w[3],w[4]);
fa_df_19BEE0167 fa3((a[4]&a[1]),(a[3]&~(a[2])),w[4],p[6],w[5]);
fa_df_19BEE0167 fa4((a[4]&a[2]),(a[3]&a[2]),w[5],p[7],w[6]);
ha_df_19BEE0167 ha2((a[4]&~(a[3])),w[6],p[8],w[7]);
assign p[9]= ((a[4]&a[3])^w[7]);

assign p[5]=((a[2]&a[1])^w[3]);

endmodule
```

Design Code for 4bit

1. **Verilog Module** — designcode.v

```
module squarer_4bit_19BEE0167(input [3:0]a,
output [7:0]p);
//wire[5:0]w;
wire[4:0]w;

assign p[0]=a[0];
assign p[1]=0;
assign p[2]=(a[1]&(~a[0]));

ha_df_19BEE0167 ha1((a[1]&a[0]),(a[2]&a[0]),p[3],w[1]);
fa_df_19BEE0167 fa1((a[2]&(~a[1])),(a[3]&a[0]),w[1],p[4],w[2]);
fa_df_19BEE0167 fa2((a[2]&a[1]),(a[3]&a[1]),w[2],p[5],w[3]);
ha_df_19BEE0167 ha2((a[3]&(~a[2])),w[3],p[6],w[4]);
//ha_df ha3((a[3]&a[2]),w[4],p[7],w[5]);
assign p[7]=((a[3]&a[2])^w[4]);
```

```
endmodule
```

2. **Fulladder.v**

```
module fa_df_19BEE0167(input a,b,cin,output sum,cout);
assign sum=a^b^cin;
assign cout=(a^b)&cin|(a&b);
endmodule
```

3. **Halfadder.v**

```
module half_adder(input a,b, output sum,cout);
xor x1(sum,a,b);
and a1(cout,a,b);
endmodule
```

4. **Test Fixture** — Test.v

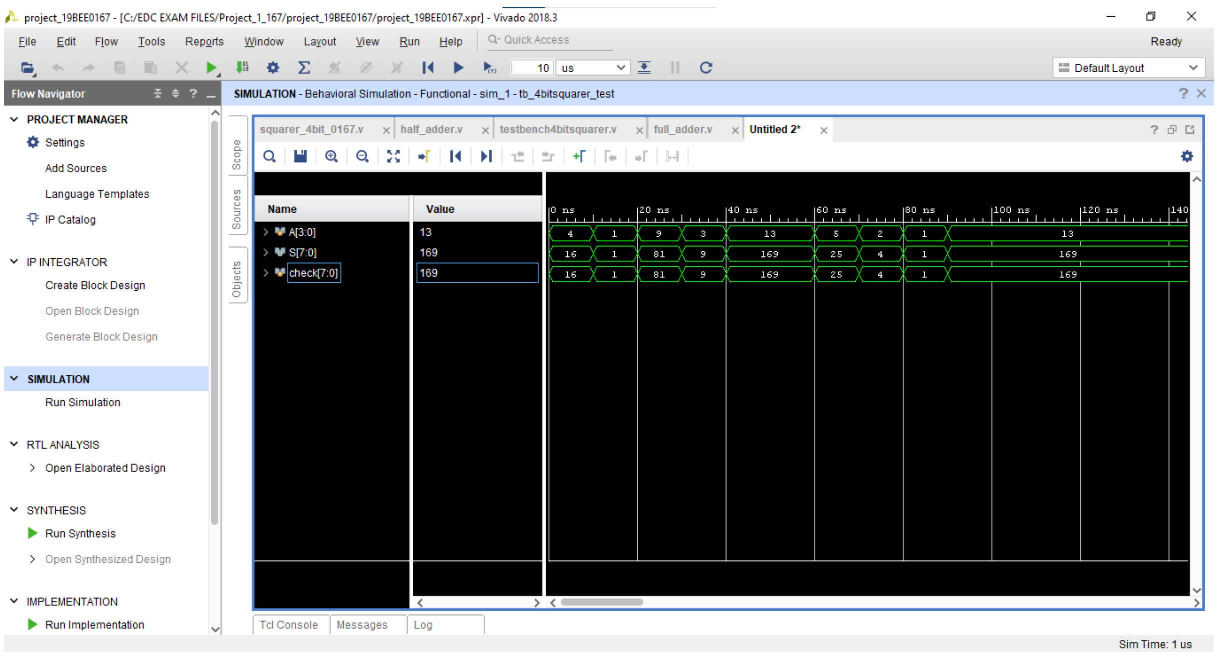
```
module tb_squarer_5bit_19BEE0167_test;
reg[4:0]A;

wire[9:0]S;
reg[9:0]check;

squarer_5bit_19BEE0167 uut(A,S);
initial repeat(10) begin
A = $random;


check = A*A;
#10 $display($time," %d=%d(%d) ",A,S,check);
end
endmodule
```

5. **Test Fixture** — Test.v

```
module tb_squarer_4bit_19BEE0167_test;
reg[3:0]A;

wire[7:0]S;
reg[7:0]check;
```

```
squarer_4bit_19BEE0167 uut(A,S);
initial repeat(10) begin
A = $random;


check = A*A;
#10 $display($time," %d=%d(%d) ",A,S,check);
end
endmodule
```

6. Output



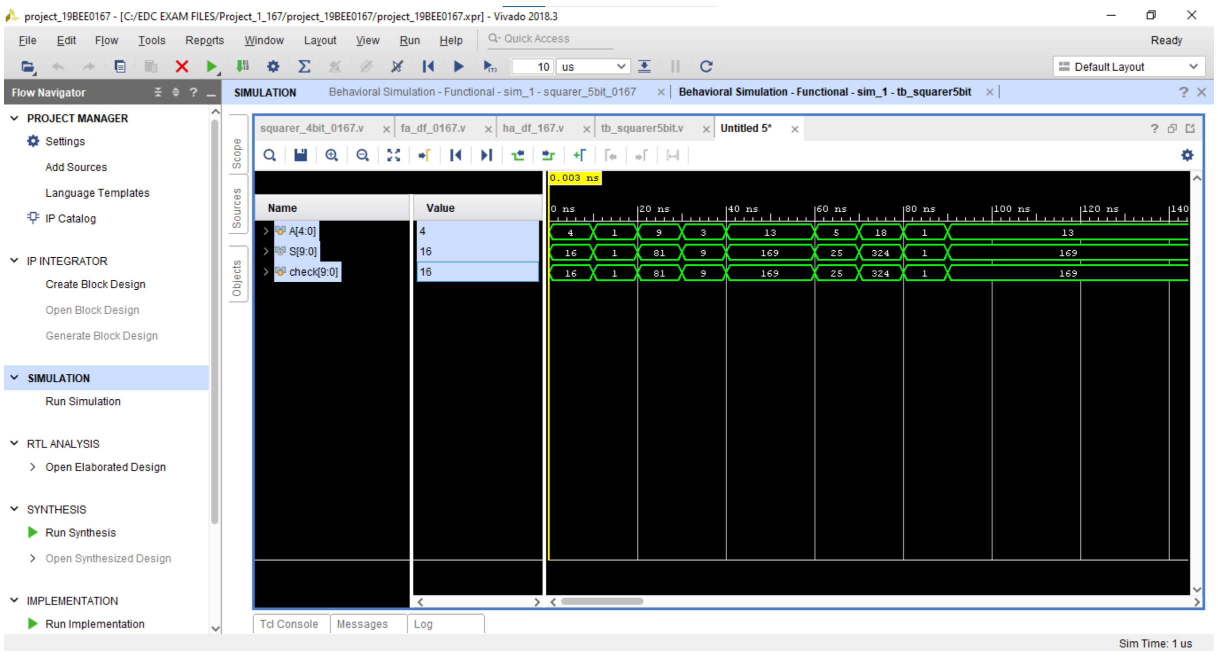
project_19BEE0167 - [C:/EDC EXAM FILES/Project_1_167/project_19BEE0167/project_19BEE0167.xpr] - Vivado 2018.3

SIMULATION - Behavioral Simulation - Functional - sim_1 - tb_4bitsquarer_test

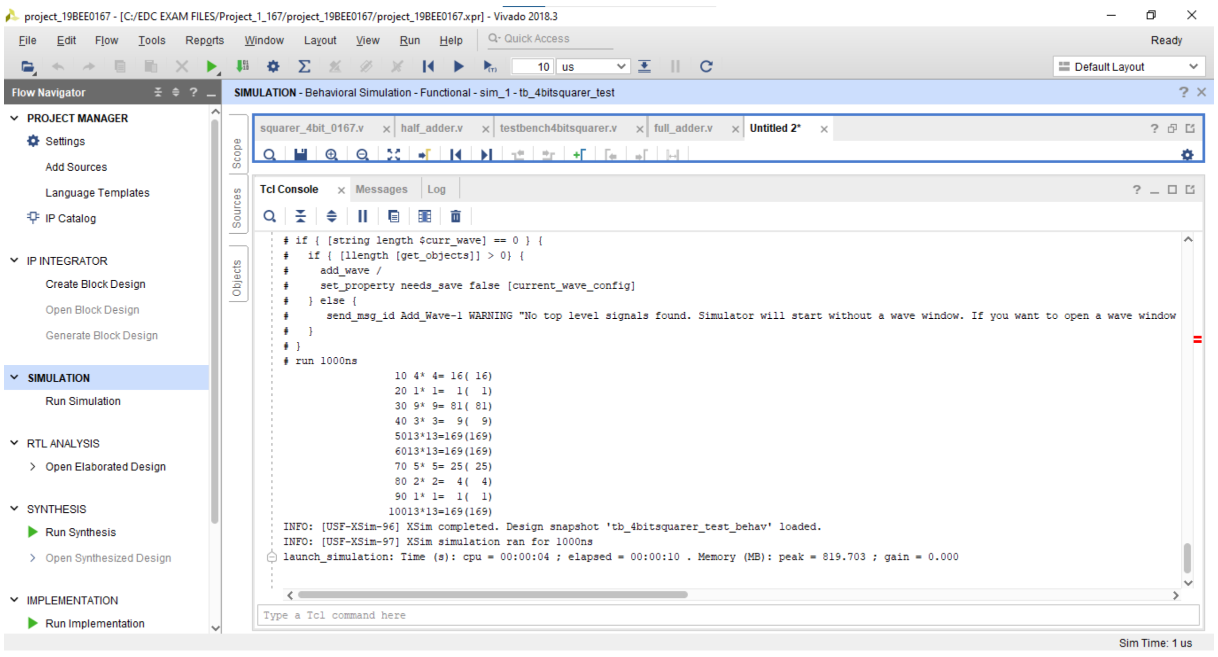
Name	Value
A[3:0]	13
S[7:0]	169
check[7:0]	169

```
A[3:0]      4   1   9   3   13   5   2   1   13
S[7:0]      16  1   81  9   169  25  4   1   169
check[7:0]  16  1   81  9   169  25  4   1   169
```

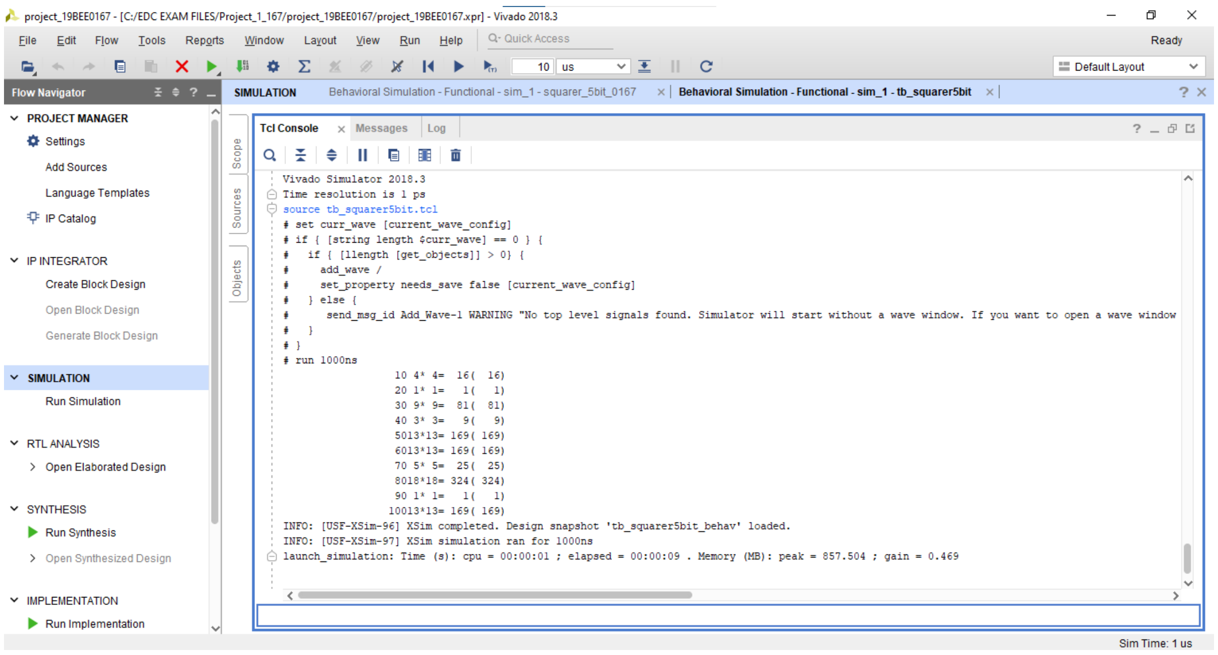
Sim Time: 1 us



7. Console Output


project_19BEE0167 - [C:/EDC EXAM FILES/Project_1_167/project_19BEE0167/project_19BEE0167.xpr] - Vivado 2018.3
SIMULATION - Behavioral Simulation - Functional - sim_1 - tb_4bitsquarer_test
squarer_4bit_0167.v
half_adder.v
testbench4bitsquarer.v
full_adder.v
Untitled 2*
Tcl Console
Messages
Log
Sim Time: 1 us

```
# if { [string length $curr_wave] == 0 } {
#   if { [llength [get_objects]] > 0} {
#     add_wave /
#     set_property needs_save false [current_wave_config]
#   } else {
#      send_msg_id Add_Wave-1 WARNING "No top level signals found. Simulator will start without a wave window. If you want to open a wave window
#   }
# }
# run 1000ns
                  10 4* 4= 16( 16)
                  20 1* 1=  1(  1)
                  30 9* 9= 81( 81)
                  40 3* 3=  9(  9)
                  5013*13=169(169)
                  6013*13=169(169)
                  70 5* 5= 25( 25)
                  80 2* 2=  4(  4)
                  90 1* 1=  1(  1)
                 10013*13=169(169)
INFO: [USF-XSim-96] XSim completed. Design snapshot 'tb_4bitsquarer_test_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:00:04 ; elapsed = 00:00:10 . Memory (MB): peak = 819.703 ; gain = 0.000
```


project_19BEE0167 - [C:/EDC EXAM FILES/Project_1_167/project_19BEE0167/project_19BEE0167.xpr] - Vivado 2018.3

SIMULATION - Behavioral Simulation - Functional - sim_1 - tb_squarer5bit

```
Vivado Simulator 2018.3
Time resolution is 1 ps
source tb_squarer5bit.tcl
# set curr_wave [current_wave_config]
# if { [string length $curr_wave] == 0 } {
#   if { [llength [get_objects]] > 0} {
#     add_wave /
#     set_property needs_save false [current_wave_config]
#   } else {
#      send_msg_id Add_Wave-1 WARNING "No top level signals found. Simulator will start without a wave window. If you want to open a wave window
#   }
# }
# run 1000ns
                  10 4* 4=  16(  16)
                  20 1* 1=   1(   1)
                  30 9* 9=  81(  81)
                  40 3* 3=   9(   9)
                  5013*13= 169( 169)
                  6013*13= 169( 169)
                  70 5* 5=  25(  25)
                  8018*18= 324( 324)
                  90 1* 1=   1(   1)
                 10013*13= 169( 169)
INFO: [USF-XSim-96] XSim completed. Design snapshot 'tb_squarer5bit_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:00:01 ; elapsed = 00:00:09 . Memory (MB): peak = 857.504 ; gain = 0.469
```

Sim Time: 1 us

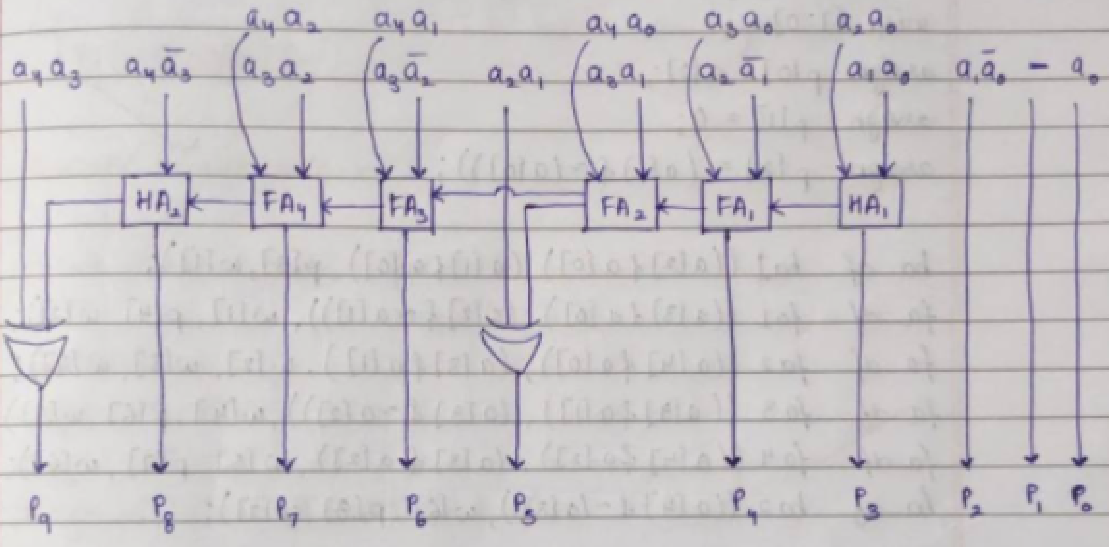
Result

1. Successfully the 4-bit and 5 bit binary squarer has been designed and the output was verified.

EXPERIMENT-(1)

BINARY SQUARER

» 5 Bit Binary Squarer

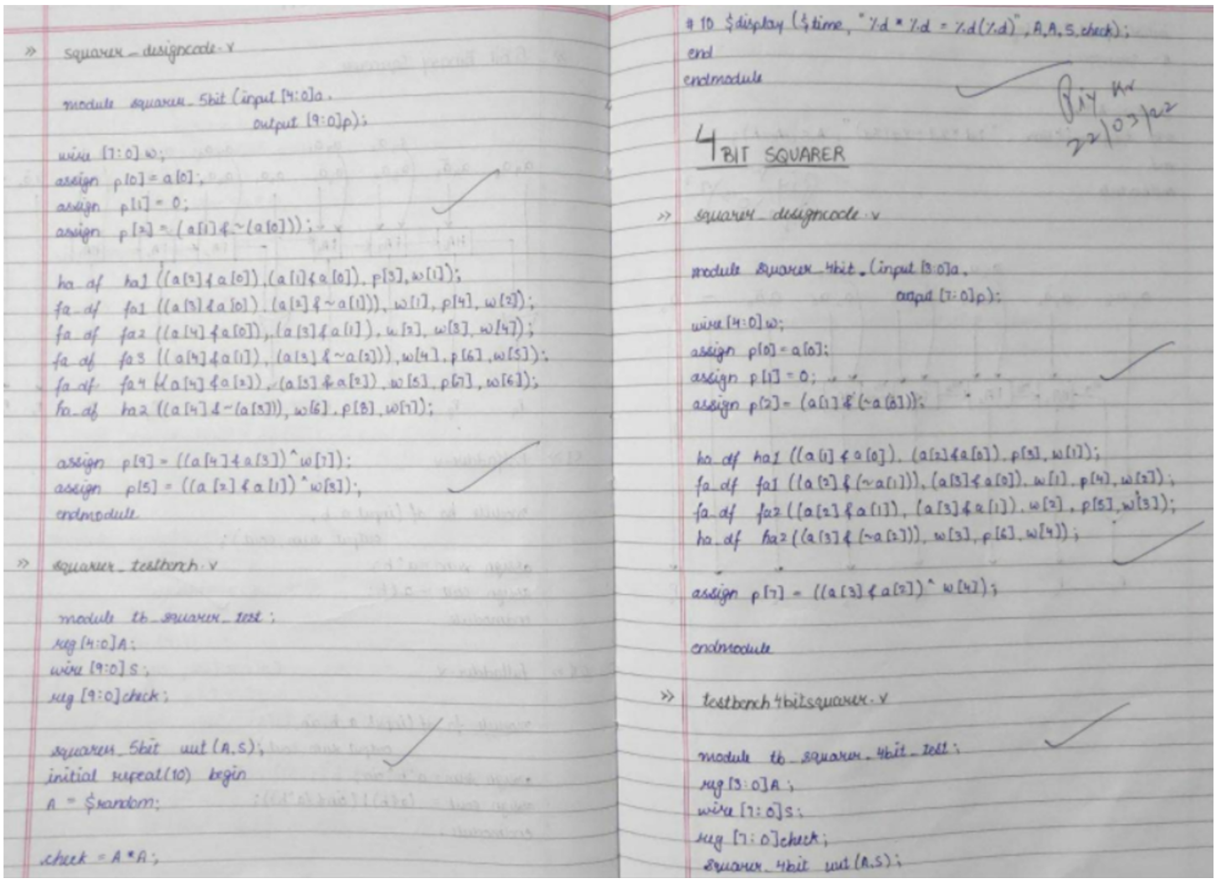


(I » halfadder.v

```
module ha_df (input a,b,
                output sum, cout);
assign sum = a^b;
assign cout = a&b;
endmodule
```

(II » fulladder.v

```
module fa_df (input a,b,cin,
               output sum, cout);
assign sum = a^b^cin;
assign cout = (a&b)|(cin&(a^b));
endmodule
```



» squarer_designcode.v

```
module squarer_5bit (input [4:0]a,
                     output [9:0]p);

wire [7:0] w;
assign p[0] = a[0];
assign p[1] = 0;
assign p[2] = (a[1] & ~(a[0]));

ha_df ha1 ((a[2] & a[0]), (a[1] & a[0]), p[3], w[1]);
fa_df fa1 ((a[3] & a[0]), (a[2] & ~a[1])), w[1], p[4], w[2]);
fa_df fa2 ((a[4] & a[0]), (a[3] & a[1]), w[2], w[3], w[4]);
fa_df fa3 ((a[4] & a[1]), (a[3] & ~a[2])), w[4], p[6], w[5]);
fa_df fa4 ((a[4] & a[2]), (a[3] & a[2]), w[5], p[7], w[6]);
ha_df ha2 ((a[4] & ~(a[3])), w[6], p[8], w[7]);

assign p[9] = ((a[4] & a[3]) ^ w[7]);
assign p[5] = ((a[2] & a[1]) ^ w[3]);
endmodule
```

» squarer_testbench.v

```
module tb_squarer_test;
reg [4:0]A;
wire [9:0] S;
reg [9:0]check;

squarer_5bit uut (A,S);
initial repeat(10) begin
A = $random;

check = A*A;
#10 $display ($time, "%d * %d = %d(%d)", A,A,S,check);
end
endmodule
```

[illegible]
22/07/22

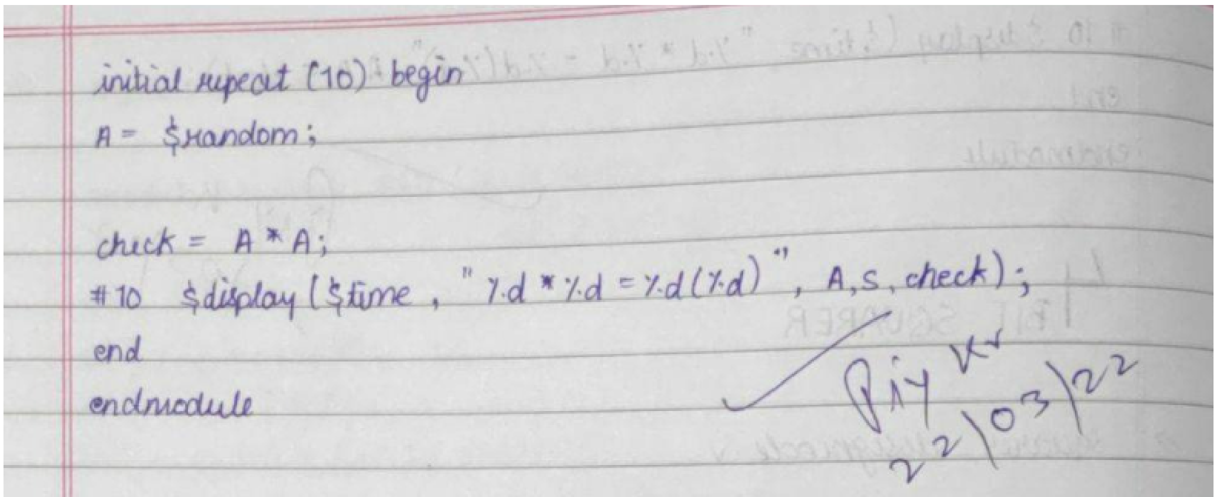
4 BIT SQUARER

» squarer_designcode.v

```
module squarer_4bit (input [3:0]a,
                     output [7:0]p);
wire [4:0] w;
assign p[0] = a[0];
assign p[1] = 0;
assign p[2] = (a[1] & (~a[0]));

ha_df ha1 ((a[1] & a[0]), (a[2] & a[0]), p[3], w[1]);
fa_df fa1 ((a[2] & (~a[1])), (a[3] & a[0]), w[1], p[4], w[2]);
fa_df fa2 ((a[2] & a[1]), (a[3] & a[1]), w[2], p[5], w[3]);
ha_df ha2 ((a[3] & (~a[2])), w[3], p[6], w[4]);

assign p[7] = ((a[3] & a[2]) ^ w[4]);

endmodule
```

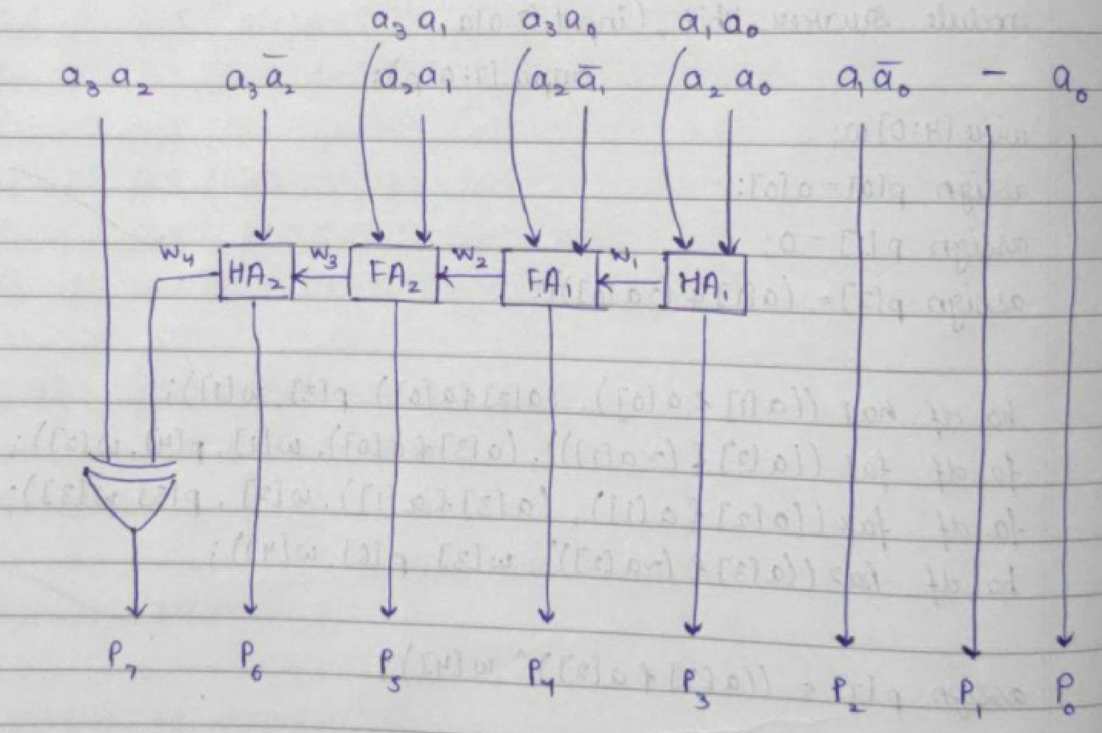
» testbench4bitsquarer.v

```
module tb_squarer_4bit_test;
reg [3:0]A;
wire [7:0]S;
reg [7:0]check;
squarer_4bit uut (A,S);
```



```
initial repeat (10) begin
A = $random;

check = A * A;
#10 $display ($time , " %d * %d = %d (%d) ", A,S, check);
end
endmodule
```

[illegible]
22/03/22



Inference

1. In this experiment learnt about how to construct multiple-bit adders.