# Digital Logic Design EEE241

Lab Terminal Report



|  |  |
| --- | --- |
| Name | **M.RUSLAN BABAR** |
| Exam | **DLD-Terminal-Exam** |
| Registration Number | **FA20-BSE-094** |
| Class | **BSE-2B** |
| Instructor’s Name | **SIR SAJID ALI GILAL** |

# Abstract:

In this Lab Terminal exam, students are assigned with a task to design an N bit binary multiplier. And this binary multiplier is based in the binary equivalent of students roll number. My roll number is 094 = 1011110 (binary equivalent).

Number of bits are 7. Now the job is to design 7 bit multiplier that has several modules like Seven bit adder and a full adder. These modules helps to solve the problem in an organized manner and display the desired multiplication results in simulation wave form.

**Contents-Lab-Report**

[Digital Logic Design EEE241 1](#_Toc75099747)

[Abstract: 2](#_Toc75099748)

[Terminal Exams Digital Logic Design 3](#_Toc75099749)

[**Introduction:** 3](#_Toc75099750)

[Task Question: 3](#_Toc75099751)

[Design and implementSolution: 4](#_Toc75099752)

[**Decimal Multiplication** 4](#_Toc75099753)

[**Binary Multiplication** 4](#_Toc75099754)

[Design and implementation of N\_bit\_Multiplier: 5](#_Toc75099755)

[ **Full Adder (Module):** 5](#_Toc75099756)

[ **7 bit Full Adder (Module):** 5](#_Toc75099757)

[ **7 bit Multiplier (Module):** 6](#_Toc75099758)

[ **Test Bench:** 8](#_Toc75099759)

[ **Simulation & results:** 9](#_Toc75099760)

[**Conclusions:** 10](#_Toc75099761)

# Terminal Exams Digital Logic Design

## **Introduction:**

In terminal exam we were assign to design a **N**\_**bit** **multiplier**, which consists of our **roll** **number**. Our roll number has to be converted to Binary and then implemented in **VHDL**. This task has to be perform on Verilog Hardware descriptive language. This tasks mainly convers the concepts of **full** **adder**, **N\_bit full adder** and an **N** **multiplier**.

# Task Question:

**Design and implement following code example on XILINX ISE Design Suite Simulator?**

Roll number **FA2-BSE-094**

Your Enrolment Number in decimal is: **094**

Your Enrolment Number in Binary is: **1011110**

Design and implement N-Bit Binary Multiplier and multiply your enrolment number with your enrolment number.

**Decimal number multiplication**

**094**

**x 094**

**-----------------**

8836

**\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.**

**Binary numbers multiplication**

**1011110**

**x 1011110**

**---------------------------**

**010001010000100**

# Design and implementSolution:

**Step 1:**

Initially we will do conversion of our roll number to binary. My roll number is **(Fa20-Bse-094).**

Binary Equivalent of (094) is (1011110).

**Step 2:**

Secondly we will do multiplication of our binary equivalent roll number.

## **Decimal Multiplication**

**094**

**X 094**

**-----------------**

8836

## **Binary Multiplication**

1011110

X 1011110

**---------------------------**

**010001010000100**

**Step 3:**

Thirdly we will identify number of bit in our binary equivalent roll number. **(1011110).**

My roll number consists of 7 bit. So we will design our multiplier in such a way that it works for 7 bits.

**Step 4:**

Now we will Design and implement following code example on XILINX ISE Design Suite Simulator

# Design and implementation of N\_bit\_Multiplier:

We will design three (3) modules for this task of 7 bit multiplication. These modules are:

* Full Adder.
* 7 Bit Full Adder.
* 7 Bit Multiplier.

These modules are called continuously for calculations.

## **Full Adder (Module):**

Whenever this module is called, this will perform the function of addition.

Full adder has the ability to add three inputs and return a Sum and a Carry.

**Full Adder:** *(Verilog Code):*

*module Full\_Adder(A,B,Cinput,S,Cout);*

*input A,B,Cinput;*

*output Cout,S;*

*//Terminal Exam*

*//Fa20-BSe-094*

*//Sir sajid Ali Gilal*

*assign S = A ^ B ^ Cinput;*

*assign Cout = (A&B) | (A&Cinput) | (B&Cinput);*

*endmodule*

## **7 bit Full Adder (Module):**

Since my roll number (1011110) consists of Seven bits, therefore

I have design a Seven bits full adder. This 7 bit adder further

**calls** the **Full Adder** for continuous and quick addition of 7 bit.

**Seven Bit Full Adder:** *(Verilog Code)*

*module Seven\_BIt\_FullAdder(P,A,B);*

*input [6:0] A,B;*

*output [7:0] P;*

*wire [6:0] W;*

*//Fa20-BSe-094*

*//Sir Sajid Ali Gilal*

*Full\_Adder a1 (A[0],B[0],0,P[0],W[0]);*

*Full\_Adder a2 (A[1],B[1],W[0],P[1],W[1]);*

*Full\_Adder a3 (A[2],B[2],W[1],P[2],W[2]);*

*Full\_Adder a4 (A[3],B[3],W[2],P[3],W[3]);*

*Full\_Adder a5 (A[4],B[4],W[3],P[4],W[4]);*

*Full\_Adder a6 (A[5],B[5],W[4],P[5],W[5]);*

*Full\_Adder a7 (A[6],B[6],W[5],P[6],W[6]);*

*assign P[7] = W[6];*

*endmodule*

## **7 bit Multiplier (Module):**

Since my roll number (**1011110**) consists of Seven bits, therefore

I have design a Seven bits multiplier. This 7 bit multiplier continuously calls the following modules

* + The **Seven** **bit** **Full** **Adder**
  + The Seven bit adder calls the **full** **adder**.

This process continues until the complete multiplication is not perform.

**Calling procedure from n\_Bit Multiplier module:**

When two numbers are received from **test** **bench**. These number are multiplied step wise. During multiplication, the process of additions also occurs for this purpose **Multiplier** **module** calls the **Seven** **bit** **adder** module for addition of 7 bits. Furthermore, the **Seven** **bit** **adder** calls the **full** **adder**. This helps in sequential multiplication of my roll number (094).

**Caller Modules Called Modules**

|  |  |  |
| --- | --- | --- |
| Multiplier Module | Calls 🡪  Calls 🡪 | Seven Bit Full Adder Module |
| Seven Bit Full Adder module | Full Adder Module |

**nBit Multiplier:**

*//Bse-094*

*//Sir Sajid Ali Gilal*

*module Muliplier\_094(P,A,B);*

*input [6:0] A;*

*input [6:0] B;*

*output [13:0] P;*

*wire [7:0] S1, S2,S3,S4,S5,S6,S7;*

*wire [6:0] T1,T2,T3,T4,T5,T6,T7;*

*and b1(P[0],A[0],B[0]);*

*and b2(T1[0],A[0],B[1]);*

*and b3(T1[1],A[0],B[2]);*

*and b4(T1[2],A[0],B[3]);*

*and b5(T1[3],A[0],B[4]);*

*and b6(T1[4],A[0],B[5]);*

*and b7(T1[5],A[0],B[6]);*

*assign T1[6] = 1'b0;*

*and c8(T2[0],A[1],B[0]);*

*and c9(T2[1],A[1],B[1]);*

*and c10(T2[2],A[1],B[2]);*

*and c11(T2[3],A[1],B[3]);*

*and c12(T2[4],A[1],B[4]);*

*and c13(T2[5],A[1],B[5]);*

*and c14(T2[6],A[1],B[6]);*

*Seven\_BIt\_FullAdder a1 (S1,T1,T2);//Seven Bit addition*

*assign P[1] = S1[0];*

*and d15(T3[0],A[2],B[0]);*

*and d16(T3[1],A[2],B[1]);*

*and d17(T3[2],A[2],B[2]);*

*and d18(T3[3],A[2],B[3]);*

*and d19(T3[4],A[2],B[4]);*

*and d20(T3[5],A[2],B[5]);*

*and d21(T3[6],A[2],B[6]);*

*Seven\_BIt\_FullAdder a2 (S2,T3,S1[7:1]);//Seven Bit addition*

*assign P[2] = S2[0];*

*and e22(T4[0],A[3],B[0]);*

*and e23(T4[1],A[3],B[1]);*

*and e24(T4[2],A[3],B[2]);*

*and e25(T4[3],A[3],B[3]);*

*and e26(T4[4],A[3],B[4]);*

*and e27(T4[5],A[3],B[5]);*

*and e28(T4[6],A[3],B[6]);*

*Seven\_BIt\_FullAdder a3 (S3,T4,S2[7:1]);//Seven Bit addition*

*assign P[3] = S3[0];*

*and f29(T5[0],A[4],B[0]);*

*and f30(T5[1],A[4],B[1]);*

*and f31(T5[2],A[4],B[2]);*

*and f32(T5[3],A[4],B[3]);*

*and f33(T5[4],A[4],B[4]);*

*and f34(T5[5],A[4],B[5]);*

*and f35(T5[6],A[4],B[6]);*

*Seven\_BIt\_FullAdder a4 (S4,T5,S3[7:1]);//Seven Bit addition*

*assign P[4] = S4[0];*

*and g36(T6[0],A[5],B[0]);*

*and g37(T6[1],A[5],B[1]);*

*and g38(T6[2],A[5],B[2]);*

*and g39(T6[3],A[5],B[3]);*

*and g40(T6[4],A[5],B[4]);*

*and g41(T6[5],A[5],B[5]);*

*and g42(T6[6],A[5],B[6]);*

*Seven\_BIt\_FullAdder a5 (S5,T6,S4[7:1]);//Seven Bit addition*

*assign P[5] = S5[0];*

*and h43(T7[0],A[6],B[0]);*

*and h44(T7[1],A[6],B[1]);*

*and h45(T7[2],A[6],B[2]);*

*and h46(T7[3],A[6],B[3]);*

*and h47(T7[4],A[6],B[4]);*

*and h48(T7[5],A[6],B[5]);*

*and h49(T7[6],A[6],B[6]);*

*Seven\_BIt\_FullAdder a6(P[14:6],T7,S5[7:1]);//Seven Bit addition*

*endmodule*

## **Test Bench:**

The purpose of the test bench is to assign multiplicand and the multiplier a binary value and that will be my binary equivalent roll number. (094) = 1011110.

A = 1011110

A = 1011110

The results will later be produced by multiplications of these two numbers.

**TEST BENCH:**

*//Fa20-BSe-094*

*//Sir Sajid ALi Gilal*

*module Test\_Bench\_094;*

*// Inputs*

*reg [6:0] A;reg [6:0] B;*

*// Outputs*

*wire [13:0] P;*

*// Instantiate the Unit Under Test (UUT)*

*Muliplier\_094 uut (*

*.P(P), .A(A), .B(B)*

*);*

*initial begin*

*// Initialize Inputs*

*A = 7'b1011110;*

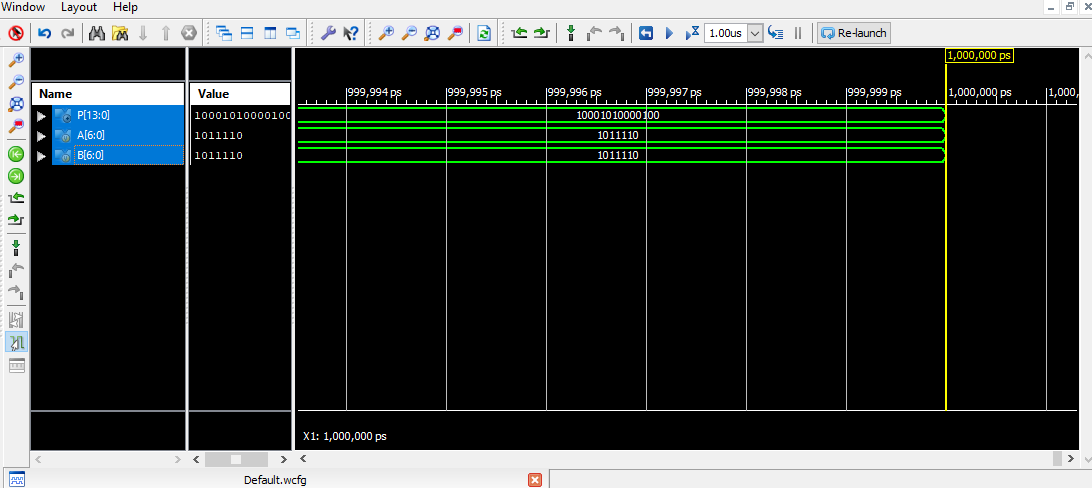
*B = 7'b1011110;*

*#100;*

*end*

*endmodule*

## **Simulation & results:**



**Simulation results = (094) Binary x (094) Binary**

1011110 x 1011110 = 10001010000100

## **Conclusions:**

This terminal exams of Digital Logic Design Laboratory assigned us a task of binary multiplication of binary equivalent of student’s roll number. This is now achieved by following the point below.

* Conversion of roll number to binary equivalent e.g: 094 = 1011110.
* Setting up modules.
  + Full Adder.
  + Seven bit full adder.
  + Seven bit multiplier.
* The Main caller module is Seven bit multiplier, which calls the other methods for proper calculations.

In this Lab:

* We design a 7 bit multiplier, with several modules.
* We learn how can a students binary equivalent roll number could be multiplied using XILINIX design suit.
* In the end we get the final accurate answers from the simulation wave form graph.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Terminal Assessment** | | | | |
| **Lab-Exam** | | | **/** |  |
|  | **Data Analysis** | **/** | **/** |
| **Data Presentation** | **/** |
| **Writing Style** | **/** |
| **Instructor Signature and Comments** | | | | |