Text

Description automatically generated

**Department of CSE**

**PROJECT REPORT**

|  |  |
| --- | --- |
| **Course Code and Name:**  CSE 345  Digital Logic Design | |
| **Project report** | |
| **Project name:**  Multiply a 3-bit binary number by 4 | |
| **Semester and Year:**  Spring 2021 | **GROUP NO:**  4 |
| **Name of Student:**  Adri Saha  (2019-1-60-024) | **Course Instructor information:**  Touhid Ahmed  Senior Lecturer  Department of Computer Science & Engineering |
| **Date of Submission**  20 May 2021 | |

**Problem statement:**

Design a combinational logic circuit which multiplies a 3-bit binary number by 4.

**Abstract:**

One of the initial courses for Electrical and Electronics Engineering students is Digital Logic Design. It can provide students with an introduction to circuit design, problem solving, testing, and feature verification.

In this project, we designed a 3-bit binary multiplier circuit multiplied by a given constant 4. A binary multiplier is a special type of combinational logic circuit. It is used to multiply two binary numbers. In this process, the number that will be multiplied by the other number is referred to as the multiplicand, and the number that will be multiplied is referred to as the multiplier. The multiplicand is multiplied by each bit of the multiplier starting from the least significant bit. Each multiplication forms a partial product, successive partial products are shifted one position to the left. The final product is obtained from the sum of the partial products.

We built and implemented a 3-bit binary number multiplier employing an adder circuit employing structural Verilog code. Furthermore, the logic design was implemented in Logisim software for verification.

**Equipment and Components:**

1. Quartus Software II
2. Logisim Software

**Objective:**

Our main objective is to implement a multiplier circuit which multiplies a 3-bit binary number by a given constant 4.

* First, we have to design the circuit as per as the given problem statement.
* Then we have to write the Structural Verilog Hardware Description Language.
* Then simulate the vector waveform to justify the desired result.

**Theory:**

**Binary Multiplication:** A binary multiplier is a combinational logic circuit or digital device that is used to multiply two binary numbers. The two numbers are referred to as multiplicand and multiplier, respectively, and the outcome is referred to as a product.

The binary multiplication method is identical to the decimal multiplication approach. There are two processes involved in binary multiplication of integers with more than one bit. The first step is a single bit-wise multiplication known as a partial product, and the second step is to combine all partial products together to form a single product.

AND gates can be used to generate partial or single-bit products. However, we also require full adders and half adders to add these partial products.

A digital multiplier's schematic design varies according on bit size. As the multiplier's bit size increases, the design becomes more complicated.

**Uses of Binary Multipliers:**

Multipliers are used in a variety of other applications nowadays.

For example:

1. A multiplier is used in a variety of digital signal processing applications.
2. It is used to create calculators, mobile phones, CPUs, and digital image processors.

**Design Procedure:**

Binary multiplication functions in the same way as standard multiplication does. There are four basic rules that are easy to understand:

0 x 0=0;

0 x1= 0;

1x0= 0;

1 x 1=1;

Let us consider the three-bit binary digits A2, A1, A0 and B2, B1, B0. We derive the following equation by multiplying the two values by each other using basic binary arithmetic techniques.

A2 A1 A0

B2 B1 B0

A2B0 A1B0 A0B0

A2B1 A2B1 A0B1 X

A2B2 A1B2 A0B2 X X

If the problem is separated into two sub problems, such as taking the first two product terms and performing the necessary arithmetic, then considering the resultant and performing the necessary arithmetic with the last product term, the following processes must be followed:

1st taking the first two product terms:

C0

C1 A2B0 A1B0 A0B0

C2 A2B1 A2B1 A0B1 X

C2 C1 C0 A0B0 + A0B1 A0B0

+A2B1 + A2B0

+A2B1

For simplicity, let us consider A0B0 = S0, A0B0+A0B1 = S1, C0+A2B0+A2B1 = S2,

C1+A2B1 =S3 and C2 = S4.

Secondly, taking the resultant term then adding to product term 3,

C4 C3

S4 S3 S2 S1 S0

C5 A2B2 A1B2 A0B2 X X

C5 C4 C3 S2 S1 S0

+S4 + S3 + A0B2

+ A2B2 +A1B2

Considering S0 = P0, S1 = P1, S2+A0B2 = P2, C3+S3+A1B2 = P3,

C4+S4+A2B2 = P4 and C5 = P5.

So, the multiplication procedure is completed.

**Circuit Design:**

The combinational circuit in Logisim software is as follows:

Diagram, engineering drawing

Description automatically generated

Figure: Simulated multiplier circuit in Logisim

Here, this multiplier can multiply two numbers having a maximum bit size of 3 bits. The bit size of the product will be 6. The maximum range of its product is 7 x 7 = 49. It can be accommodated in 6 bits which is the size of its output product.

For example, let us take a decimal number 6.

Equivalent binary of (6)10= (110)2 and equivalent binary of (4)10 = (100)10.

Now, 1 1 0

1 0 0

C0 0 0 0

C1 0 0 0 x

C2 1 1 0 x x

0 1 1 0 0 0

The result is equivalent to decimal (24)10.

**Verilog Codes:**

Here, half adder & Full adder code is used as reference code inside the main code of binary multiplication design.

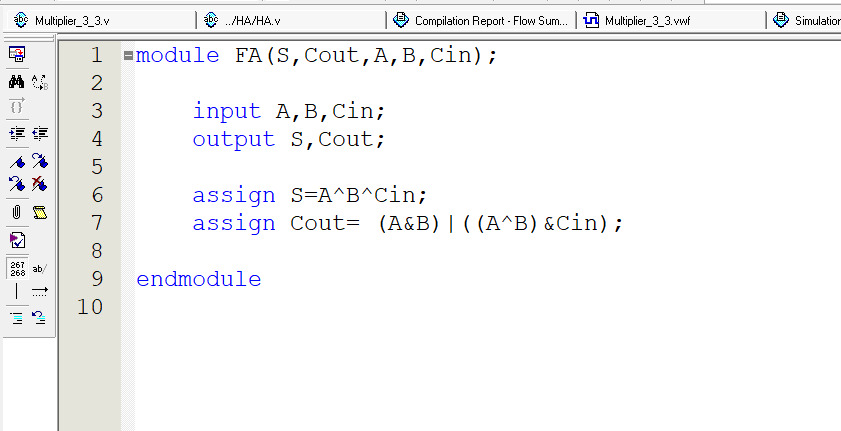


Figure: Verilog Code of full adder

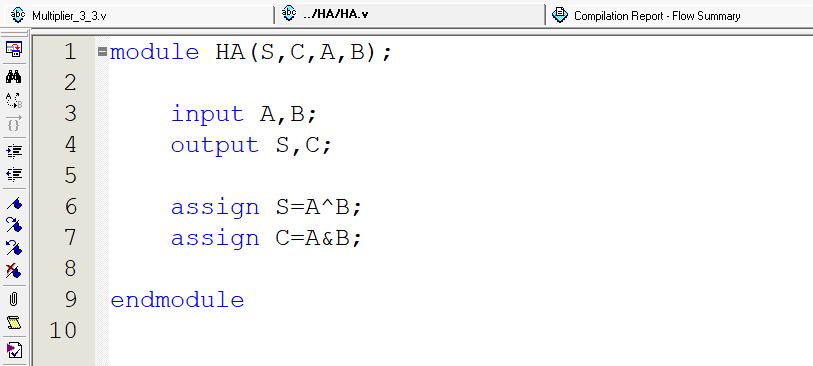


Figure: Verilog Code of half adder

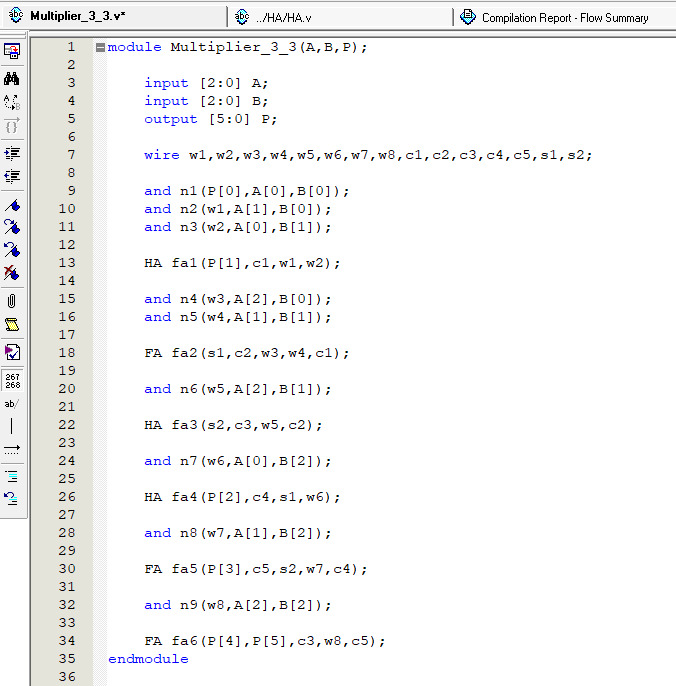
****

Figure: Verilog Code of Multiplying two 3-bit binary numbers

**Verilog Code Description:**

In the Verilog code, we have used three vector variables namely A, B & P where A & B are inputs of 3 bits and output P which has a maximum range of 7 x 7=49 and it can be accommodated in 6 bits. In order to connect the output results to another input, we took several wires (w1, w2, w3, w4, w5, w6, w7, w8, c1, c2, c3, c4, c5, s1, s2). In the circuit, we have used in total 9 and gates, 3 half adders and 3 full adders. We have used half adders to produce sum and carry of 2 inputs and full adders to produce sum and carry of more than 2 inputs. First, we implemented an and operation on A [0] & B [0] to produce output P [0]. Then we implemented another two and operations on A [1], B [0] & A [0], B[1] which is connected to wires w1 and w2 respectively. Then we took a half adder and connected w1 and w2 to generate output P [1] and a carry c1. Again, another two and gates were used to produce output which were temporarily stored in wires w3 and w4. Next, a full adder was used to connect w3, w4 and previous carry c1 to produce sum s1 and carry c2. An and operation was implemented on A[2] & B[1] and stored the output in wire w5. A half adder was taken and wire w5 and previous carry c2 to generate sum s2 and carry c3. An and operation was implemented again on A[0] & B[2] and stored the output in wire w6. Again, we took a half adder and connected previous sum s1 and wire w6 to generate output P[2] and carry c4. An and operation was implemented on A[1] & B[2] and connected it to wire w7. After that, a full adder was used to connect s2, w7 and c4 to produce P [3] and c5. An and operation was implemented again on A[2] & B[2] and stored the output in wire w8 afterwards. Finally, c3, w8 and c5 were connected to a full adder to generate outputs P [4] and P[5].

**Verilog Code Simulation:**

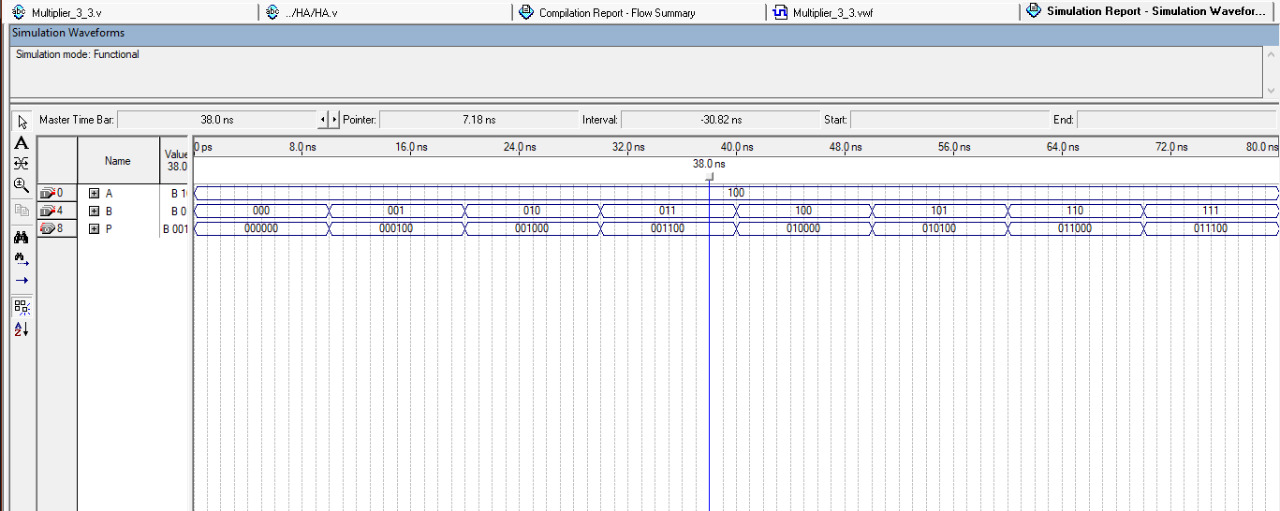


Figure: Simulation of Verilog Code

In this simulation, end time was takes 80 ns for considering 8 values. A & B is 3 bin binary inputs. Here we see each binary multiplication is correctly coming into the output P. Input A is fixed which is (100)2 & B is randomly checking for all numbers in range (1-7)10 as 111 is the highest 3-bit binary number which represent decimal number 7. So, the arbitrary value is taken for both A & B. For A, arbitrary value is (100)2 for all intervals as 4 is fixed. And for B, arbitrary value is constantly 000,001,010,011, 100,101,110,111 for each 10 seconds interval. Then we got the output in P which is the result of each multiplication.

**Discussion:**

So, after all this simulation of design we can find the perfect result for each input. We consider 8 values here.

This is the following input & outputs:

|  |  |  |
| --- | --- | --- |
| Binary input, A (4)10= (100)2 fixed | Binary input, B | 6-bit binary multiplication Output, P |
| (100)2 | (000)2 = (0)10 | (000000)2 = (0)10 |
| (100)2 | (001)2 = (1)10 | (000100)2 = (4)10 |
| (100)2 | (010)2 = (2)10 | (001000)2 = (8)10 |
| (100)2 | (011)2 = (3)10 | (001100)2 = (12)10 |
| (100)2 | (100)2 = (4)10 | (010000)2= (16)­10 |
| (100)2 | (101)2 = (5)10 | (010100)2= (20)10 |
| (100)2 | (110)2 = (6)10 | (011000)2= (24)10 |
| (100)2 | (111)2 = (7)10 | (0111000)2= (28)10 |

Moreover, all the designs implemented by structural behavioral Verilog codes. We use half adder & full adders to implement these designs. All the outputs have given the correct result with truth tables. That is why, we can say there is no error doing this experiment.

**Conclusion:** From this experiment, we learned to design & implement 3-bit binary number multiplier using adder circuit by using structural behavioral Verilog code. By using this circuit, one can multiple two 3-bit binary numbers. One can also find multiplication table as we showed here 4’s multiplication table. From this experiment, we learned the uses of multiplications circuit as well as uses of half & full adders to form this designed multiplication circuit. we also learned to draw combinational logic circuit design in Logisim software. Finally, we were able to verify their logic design by simulating the results. From this experiment, we horned our hands-on experience by simulating these designs in Quartus & Logisim Software.

**Reference:**

1. Digital Logic Design – Md. Mozammel Huq Azad Khan.
2. Fundamentals of Digital Logic with Verilog Design by Stephen Brown, Zvonko Vranesic-McGraw-Hill Science Engineering Math (2013)
3. <https://www.elprocus.com/half-adder-and-full-adder/>
4. <https://www.geeksforgeeks.org/difference-between-half-adder-and-full-adder/>
5. <https://www.electricaltechnology.org/2018/05/binary-multiplier-types-binary-multiplication-calculator.html>
6. <https://www.youtube.com/watch?v=U_qdc6GgmFo&t=596s>
7. <https://www.youtube.com/watch?v=X_6WWn0KIFc>