Introduction:

A binary multiplier is an electronic circuit used in digital electronics, such as a computer, to multiply two binary numbers. It was built using binary adders. Here in this project we have to design a circuit for 2-bit multiplier. Both A and B is the multiplicand and multiplier in this project. A and B has 2 bits which is A=A1A2 and B=B1B2.The output of this circuit is the result of A×B which is a 4 bit number because when we multiply two 2 bit number the maximum output is 4 bit number. In this project, the output is represented by R3R2R1R0.

What is 2-bit multiplier:

It is similar to the method which we learned in primary school level but the thing is we have to done this on base-10 integers and here it has been modified in to a base-2 (binary) numeral system. The method we taught in school for multiplying decimal numbers is based on calculating partial products, shifting them to the left and then adding them together. One example is given below:

221

x 300

=====

000 (this is 221 x 0)

000 (this is 221x 0, shifted one position to the left)

663 (this is 221x 3, shifted two positions to the left)

=====

66300

Binary multiplier does exactly the same, but with binary numbers (either 0 or 1). The multiplication of two binary numbers comes down to calculating partial products shifting them left, and then adding them together which is called binary addition:

1011 (this is 11 in binary)

x 1110 (this is 14 in binary)

======

0000 (this is 1011 x 0)

1011 (this is 1011 x 1, shifted one position to the left)

1011 (this is 1011 x 1, shifted two positions to the left)

+ 1011 (this is 1011 x 1, shifted three positions to the left)

=========

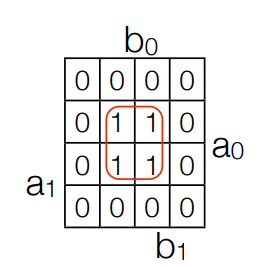
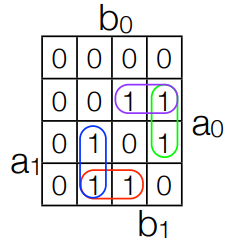
10011010 (this is 154 in binary)

In this example we use 4 bit number but in the project we will use 2 bit number, for this reason we called this 2 bit multiplier.

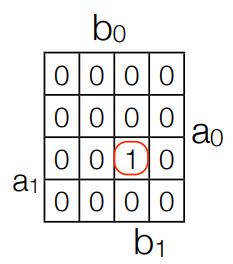
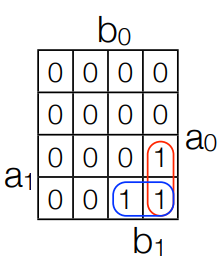
Circuit Design: The circuit of 2-bit multiplier was made from the given truth table. The truth table has been given bellow,

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

From this truth table, the equations of r0,r1,r2,r3 was formed through K-MAP. The K-MAP operations has been given bellow,



R0 = A0B0 R1 = A1A0'B0 + A1B1'B0 + A1'A0B1 + A0B1B0'

R2 = A1A0'B1 + A1B1B0' R3 = A1A0B1B0

00 01 11 10

00

01

11

10

A1A0

B1B0

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Verilog code and circuit simulation: The varilog code has been given bellow:

module project\_1(input a0,a1,b0,b1,output r0,r1,r2,r3);

wire w0,w1,w2,w3,w4,w5;

and g1(r3,a0,a1,b0,b1),

g2(w0,a1,a0,b1),

g3(w1,a1,a0,b1,~b0),

g4(w2,a1,~b1,b0),

g5(w3,a1,~a0,b0),

g6(w4,~a1,a0,b1),

g7(w5,a0,b1,~b0),

g8(r0,a0,b0);

or g9(r2,w0,w1),

g10(r1,w2,w3,w4,w5);

endmodule

After simulating the Verilog code we found the figures bellow:

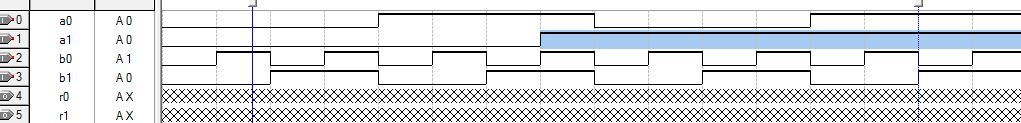


Fig1**.**  Input of the circuit

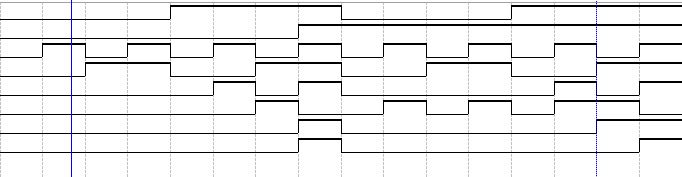


Fig2**.**  Output of the simulation.