DIGITAL LOGIC DESIGN
[2EC303]

SPECIAL ASSIGNMENT

TOPIC:
4X4 MULTIPLIER

SUBMITTED BY:

MARKAND JOSHI(21BEC065)
MITTAL KALAL(21BEC067)

INDEX

INTRODUCTION TO MULTIPLIERS:

A multiplier, also known as an array multiplier, is a combinational digital circuit used in digital systems to perform multiplication of two binary numbers.

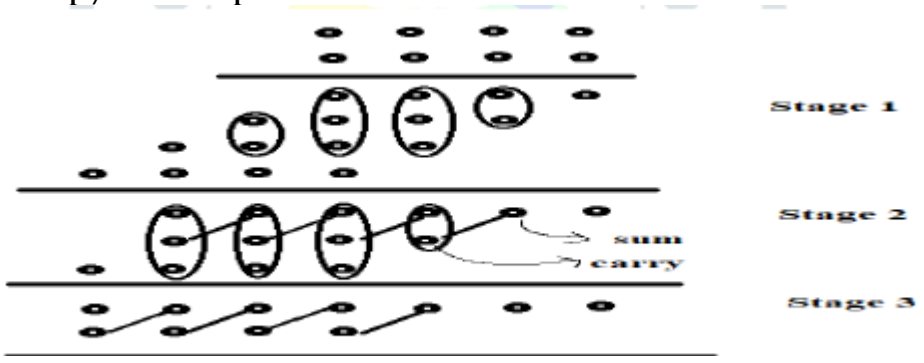
In our project, a 4-bit multiplicand and multiplier are multiplied to obtain the partial products, then after by adding them vertically, an 8 bit product is obtained.

Multipliers have their use in binary multiplications, Fourier transform and other fields because of their fast operations. These are also used in commercial applications like computers, mobiles, high speed calculators, and some processors.

Types of Multipliers :

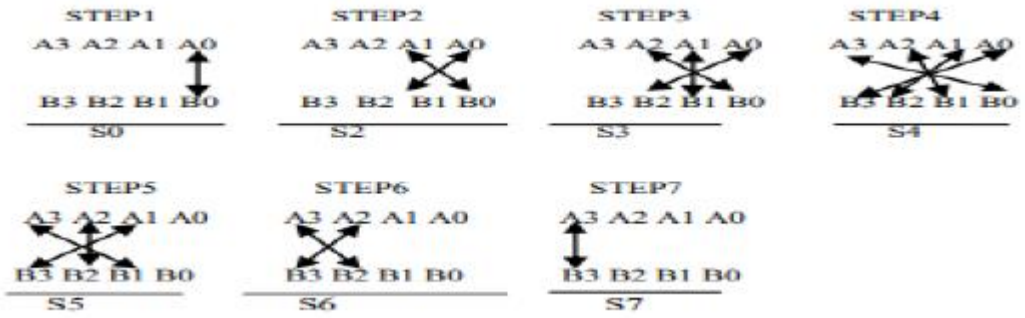
1. WALLACE MULTIPLIER

- Every Multiplication algorithm consists of three stages. They are i) Generation of partial products ii) Reduction of partial products iii) Addition of partial products.
- In the reduction process, use full adders wherever the three elements are present. And use half adders wherever the two elements are there. Generated sum and carry bits from the half and full adders will be passed to the next stage.
- Pass the left over elements to the next stage. Repeat this process until getting the two rows [1][6]. Figure.3 shows the dot diagram of Wallace multiplier. Finally add last two rows to get the output [3]. In Fig.2 p0, p1, p2, p3, p4, p5,p6 and p7 are output bits.

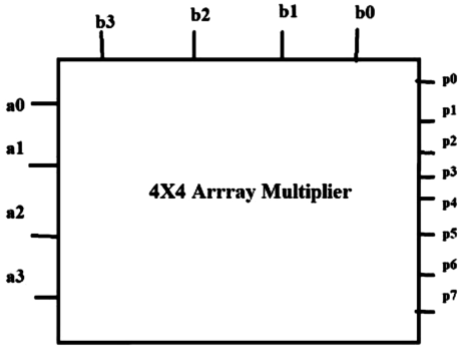


2. VEDIC MULTIPLIER

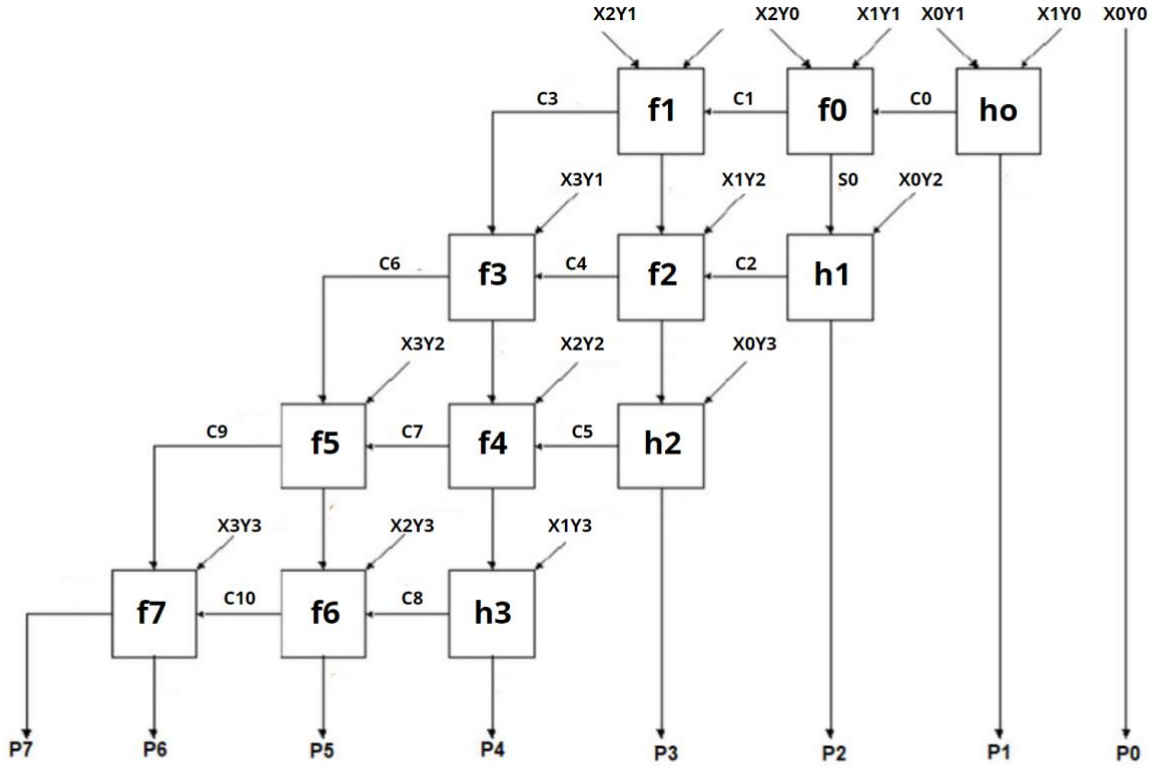
- It is based on the vertically and crosswise method. Consider two numbers A=A3A2A1A0 and B=B3B2B1B0. A3, A2, A1, A0, B3, B2, B1 and B0 are the bits in the number.
- Multiply A0 and B0 and the result is sum s0. In the step2, multiply A0 and B1 and add the result to the product of A1 and B0.The result obtained is s1 and the carry is c1. In this manner multiply and add the bits as shown in Fig.3 for remaining steps.
- Here carry generated from one step will be added to the sum in the next step [7].
- Here in this multiplier 4 full adders and three half adders are used. It is more efficient than Array and Wallace multipliers in terms of power and propagation delay.
- The disadvantage of this multiplier is that the system becomes complex for complex numbers.



BLOCK DIAGRAM



CIRCUIT DIAGRAM



VERILOG CODE:

fulladder.v — Compilation Report - fulladder

```
1  module fulladder(x,y,cin,s,cout); // initiating submodule 2
2  input x,y,cin; // declaration of inputs
3  output s,cout; // declaration of outputs
4  assign s = x ^ y ^ cin; // calculation of sum using Data flow modelling
5  assign cout = ( x & y ) | ( x & cin ) | ( y & cin ); // calculation of carry using data flow moc
6  endmodule
7
```

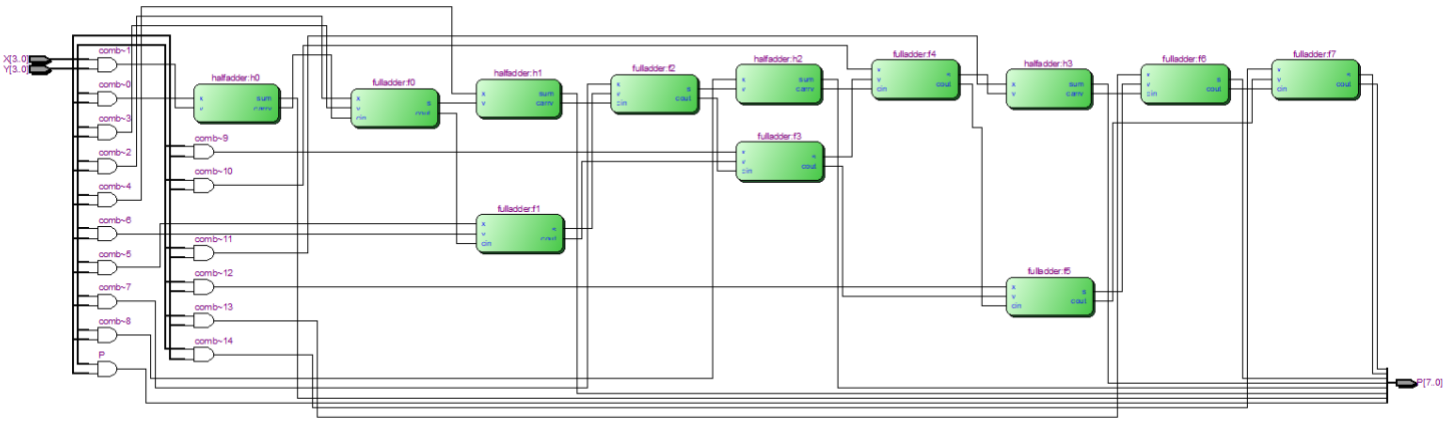
halfadder.v* — Compilation Report - halfadder

```
1  module halfadder(x,y,sum,carry); // initiating submodule 1
2  input x,y; // declaration of inputs
3  output sum,carry; // declaration of outputs
4  assign sum = x ^ y; // calculation of sum using Data Flow Modelling
5  assign carry = x & y; // calculation of carry using Data Flow Modelling
6  endmodule
7
```

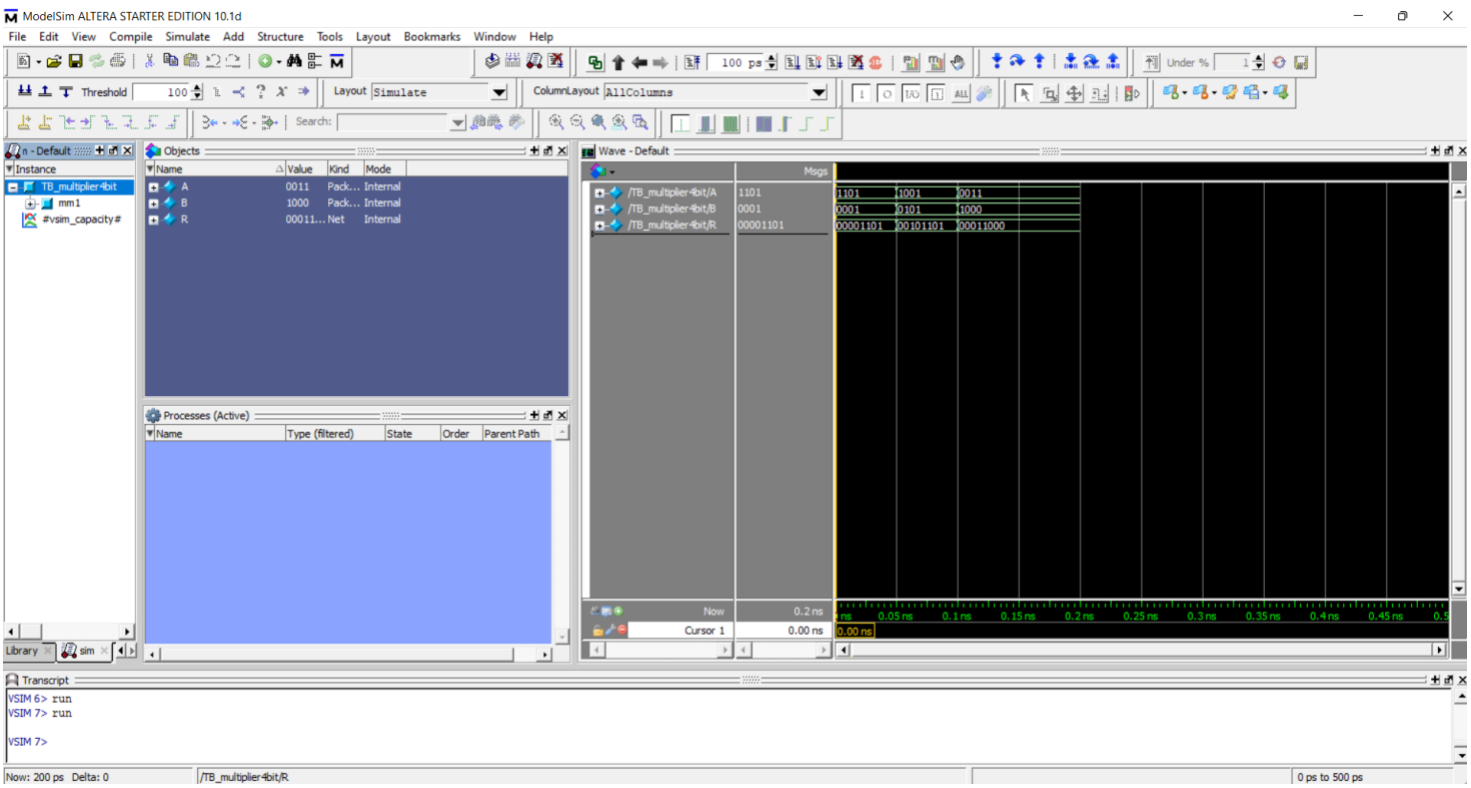
multiplier4bit.v* | Compilation Report - multiplier4bit

```
1  module multiplier4bit(X,Y,P); // initiating main module
2  input [3:0] X,Y; // declaration of inputs
3  output[7:0] P; // declaration of outputs
4  assign P[0]= X[0]&Y[0]; // LSB of the final product
5  // Logic according to the circuit diagram
6  halfadder h0 (X[0]&Y[1] , X[1]&Y[0] , P[1] , C0);
7  fulladder f0 (X[1]&Y[1] , X[2]&Y[0] , C0 , S0 , C1);
8  halfadder h1 (X[0]&Y[2] , S0 , P[2] , C2);
9  fulladder f1 (X[2]&Y[1] , X[3]&Y[0] , C1 , S1 , C3);
10 fulladder f2 (X[1]&Y[2] , S1 , C2 , S2 , C4);
11 halfadder h2 (X[0]&Y[3] , S2 , P[3] , C5);
12 fulladder f3 (X[3]&Y[1] , C3 , C4 , S3 , C6);
13 fulladder f4 (X[2]&Y[2] , S3 , C5 , S4 , C7);
14 halfadder h3 (X[1]&Y[3] , S4 , P[4] , C8);
15 fulladder f5 (X[3]&Y[2] , C6 , C7 , S5 , C9);
16 fulladder f6 (X[2]&Y[3] , S5 , C8 , P[5] , C10);
17 fulladder f7 (X[3]&Y[3] , C9 , C10 , P[6] , P[7]);// MSB of the final product
18 endmodule
19
20 //Here is the Test Bench
21 module TB_multiplier4bit();
22 reg [3:0] A,B;
23 wire [7:0] R;
24 multiplier4bit mm1(.X(A), .Y(B), .P(R));
25 initial
26 begin
27    A=4'b1101; B=4'b0001;
28    #50;
29    A=4'b1001; B=4'b0101;
30    #50;
31    A=4'b0011; B=4'b1000;
32 end
33 endmodule
```

RTL VIEWER:



MODELSIM WAVEFORM:



CONCLUSION

In this project, we learned the fundamental working of a multiplier, how to implement a given design using dataflow modeling style in Verilog and stimulate its waveform in modelsim. While implementing the design, we came to know how any hardware is given a set of codes to follow and choose suitable modeling style according to the given purpose.

REFERENCES

- https://www.elprocus.com/
- https://www.edaplayground.com/
- PALNITKAR-Verilog HDL
- https://www.youtube.com/watch?v=AxrlH7vHOpw&t=1014s

THANK YOU !