SIMULATION AND IMPLEMENTATION OF LOGIC GATES , ADDERS AND SUBTRACTORS

AIM:

To simulate the following logic circuits using Vivado 2023.2.

1) Logic Gates

2) Adders

3) Subtractors

APPARATUS REQUIRED:

VIVADO 2023.2

PROCEDURE:

STEP:1 Launch the Vivado 2023.2 software.

STEP:2 Click on “create project ” from the starting page of vivado .

STEP:3 Choose the design entry method:RTL(verilog/VHDL)

STEP:4 Crete design source and give name to it and click finish.

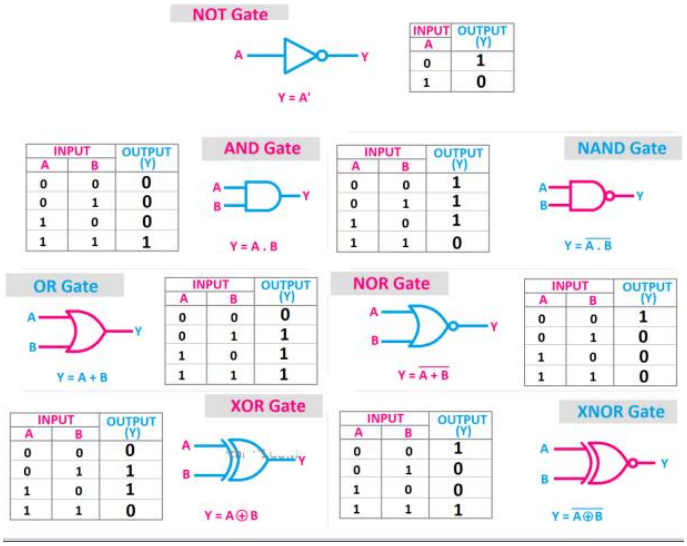
STEP:5 Write the verilog code and check the syntax.

STEP:6 Click “run simulation” in the navigator window and click “Run behavioral simulation”.

STEP:7 Verify the output in the simulation window.

LOGIC GATES

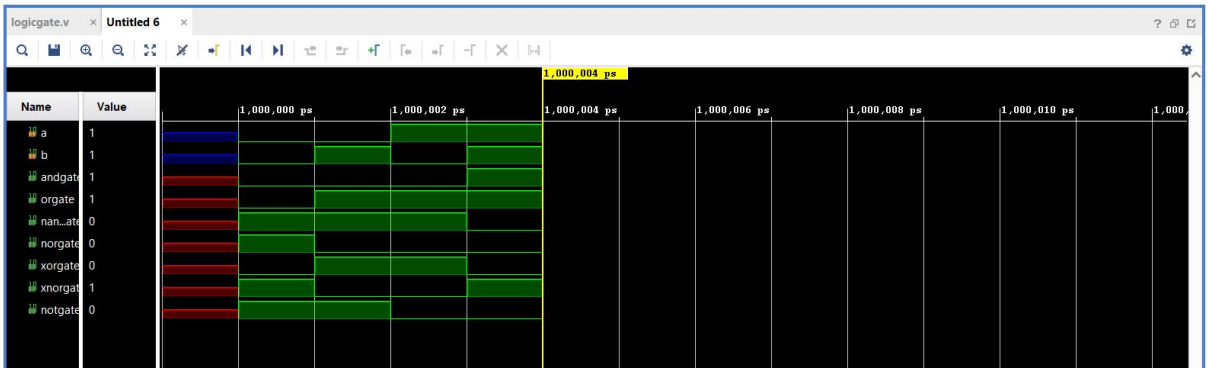
LOGIC DIAGRAM



VERILOG CODE

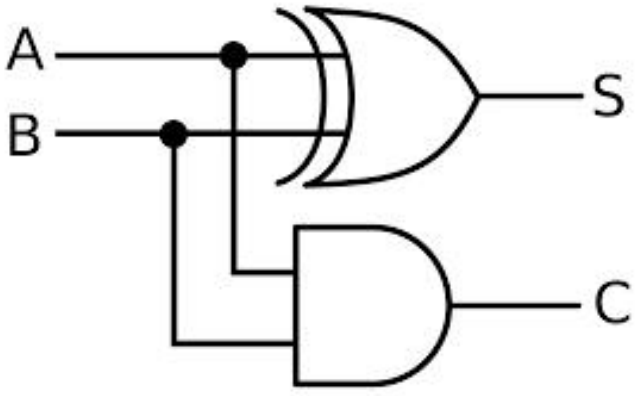
```
module logicgate(a,b,andgate,orgate,nandgate,norgate,xorgate,xnorgate,notgate);
input a,b;
output andgate,orgate,nandgate,norgate,xorgate,xnorgate,notgate;
and(andgate,a,b);
or(orgate,a,b);
nand(nandgate,a,b);
nor(norgate,a,b);
xor(xorgate,a,b);
xnor(xnorgate,a,b);
not(notgate,a);
Endmodule
```

OUTPUT WAVEFORM



HALF ADDER

LOGIC DIAGRAM

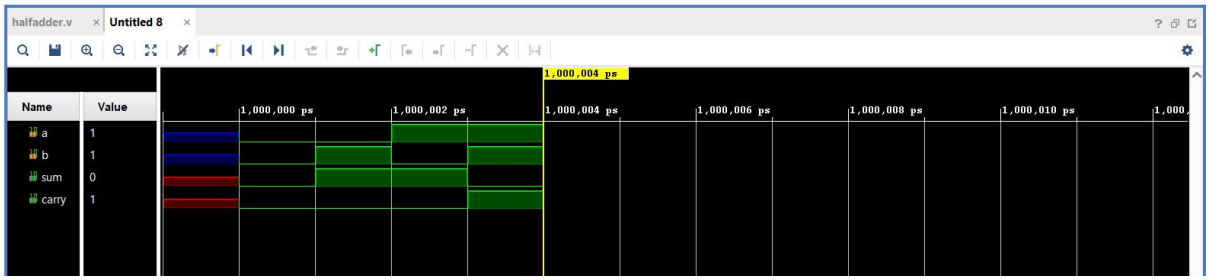


Truth Table			
Input		Output	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

VERILOG CODE

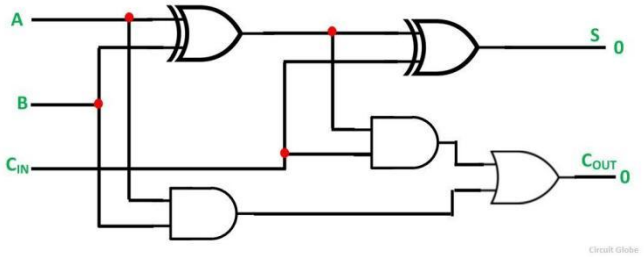
```
module half_adder(a,b,sum,carry);
input a,b;
output sum,carry;
xor g1(sum,a,b);
and g2(carry,a,b);
endmodule
```

OUTPUT WAVEFORM



FULL ADDER

LOGIC DIAGRAM



Inputs			Outputs	
A	B	C_{in}	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

VERILOG CODE

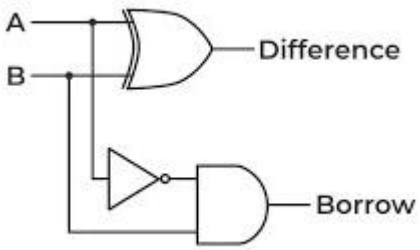
```
module fulladder(a,b,c,sum,carry);
input a,b,c;
output sum,carry;
wire w1,w2,w3;
xor(w1,a,b);
xor(sum,w1,c);
and(w2,w1,c);
and(w3,a,b);
```

```
or(carry,w2,w3);
endmodule
```

OUTPUT WAVEFORM

HALF SUBTRACTOR

LOGIC DIAGRAM

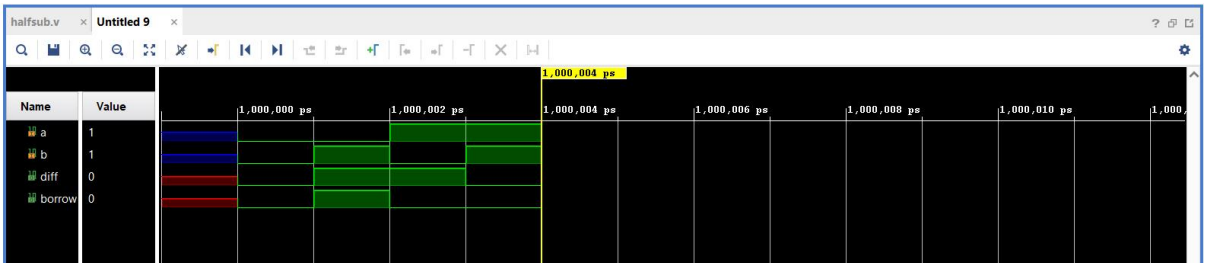


Inputs		Outputs	
A	B	Diff	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

VERILOG CODE

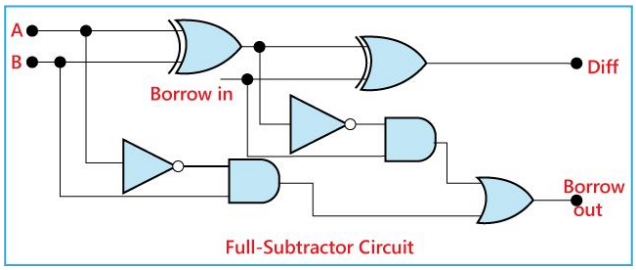
```
module halfsub(a,b,diff,borrow);
input a,b;
output diff,borrow;
xor(diff,a,b);
and(borrow,~a,b);
endmodule
```

OUTPUT WAVEFORM



FULL SUBTRACTOR

LOGIC DIAGRAM

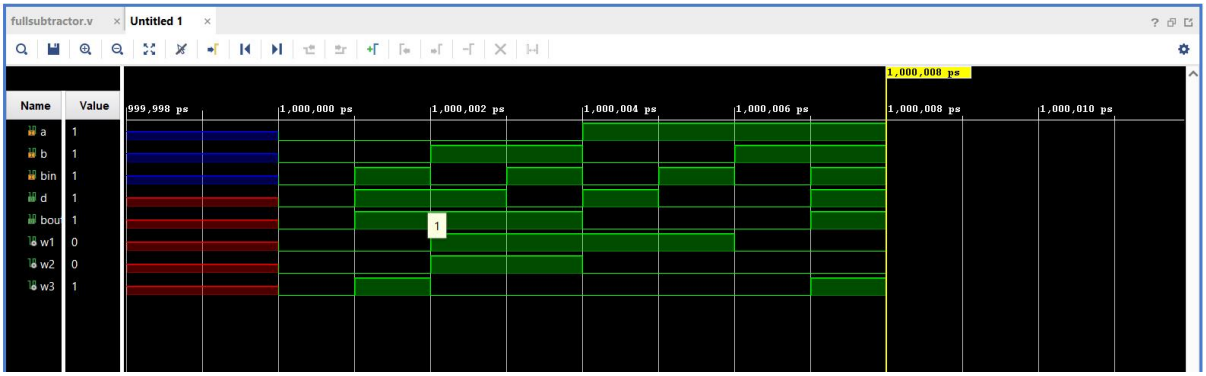


Inputs			Outputs	
A	B	$Borrow_{in}$	Diff	Borrow
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

VERILOG CODE

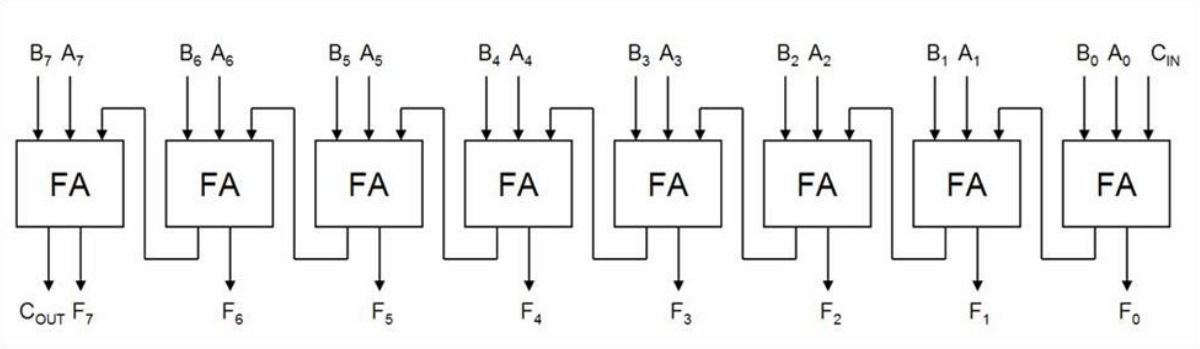
```
module fs(a,b,bin,d,bout);
input a,b,bin;
output d,bout;
wire w1,w2,w3;
xor(w1,a,b);
xor(d,w1,bin);
and(w2,~a,b);
and(w3,~w1,bin);
or(bout,w3,w2);
endmodule
```

OUTPUT WAVEFORM



RIPPLE CARRY ADDER

LOGIC DIAGRAM



VERILOG CODE

```
module fulladder(a,b,c,sum,carry);
input a,b,c;
output sum,carry;
wire w1,w2,w3;
xor(w1,a,b);
xor(sum,w1,c);
and(w2,w1,c);
and(w3,a,b);
or(carry,w2,w3);
endmodule


module rca_8bit(a,b,cin,s,cout);
input [7:0]a,b;
input cin;
output [7:0]s;
output cout;
wire [7:1]w;
fulladder f1(a[0], b[0], cin, s[0], w[1]);
fulladder f2(a[1], b[1], w[1], s[1], w[2]);
fulladder f3(a[2], b[2], w[2], s[2], w[3]);
fulladder f4(a[3], b[3], w[3], s[3], w[4]);
fulladder f5(a[4], b[4], w[4], s[4], w[5]);
```

fulladder f6(a[5], b[5], w[5], s[5], w[6]);

fulladder f7(a[6], b[6], w[6], s[6], w[7]);

fulladder f8(a[7], b[7], w[7], s[7], cout);

endmodule

OUTPUT WAVEFORM

RESULT:

Thus the simulation and implementation of logic gates, adders and subtractors has done and outputs are verified successfully.