SIMULATION OF LOGIC GATES ,ADDERS AND SUBTRACTORS

AIM:

To simulate Logic Gates ,Adders and Subtractors using Vivado 2023.2.

APPARATUS REQUIRED:

VIVADO 2023.2

PROCEDURE:

STEP:1 Start the Xilinx navigator, Select and Name the New project.

STEP:2 Select the device family, device, package and speed.

STEP:3 Select new source in the New Project and select Verilog Module as the Source type.

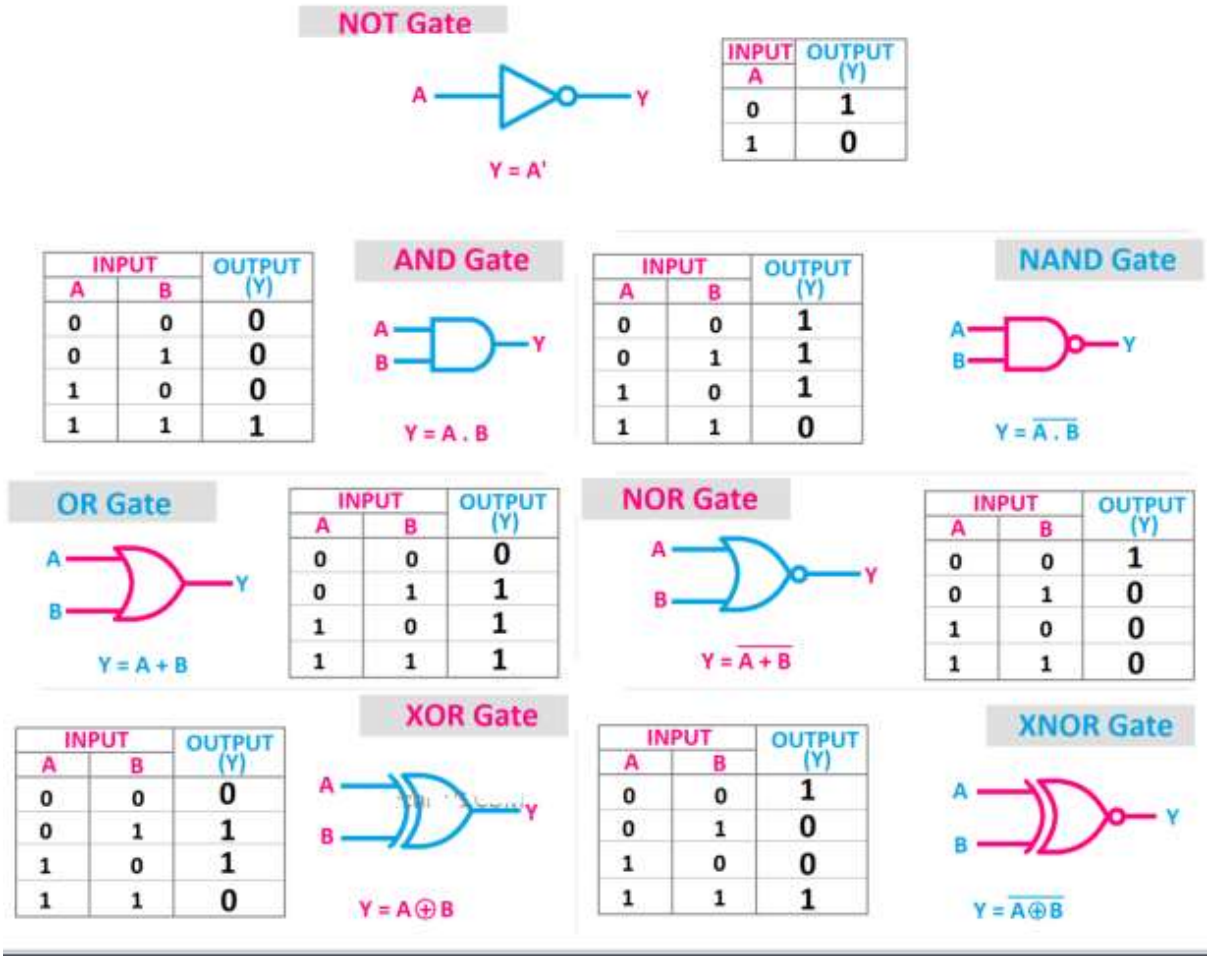
STEP:4 Type the File Name and Click Next and then finish button. Type the code and save it.

STEP:5 Select the Behavioral Simulation in the Source Window and click the check syntax.

STEP:6 Click the simulation to simulate the program and give the inputs and verify the outputs as per the truth table.

LOGIC GATES

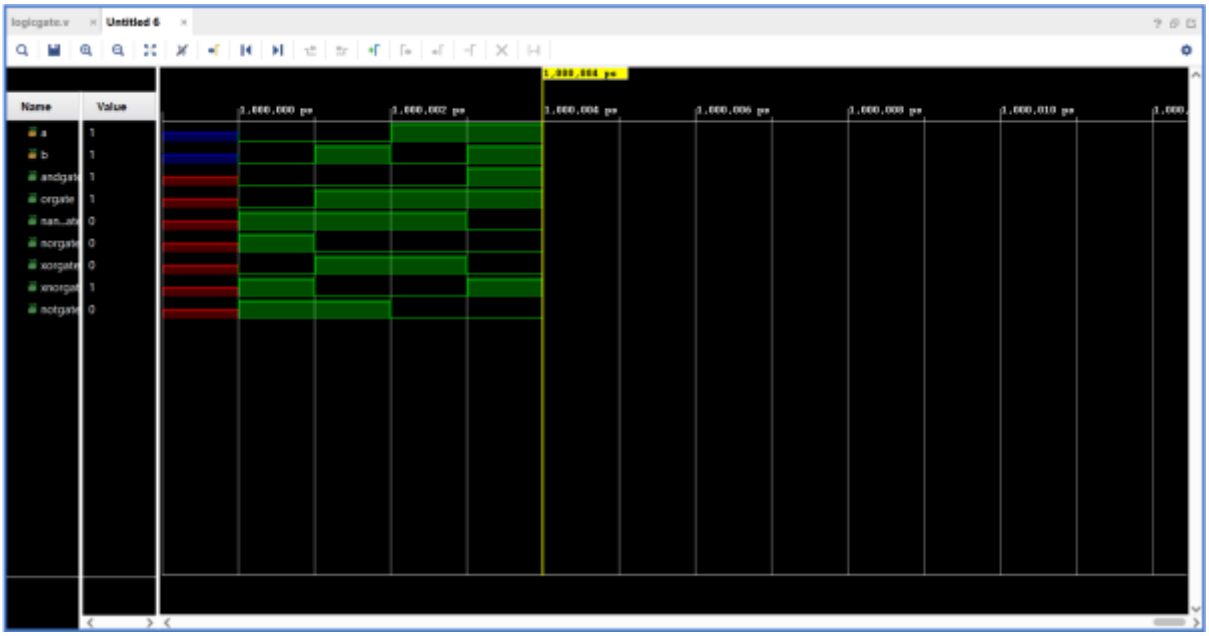
LOGIC DIAGRAM



VERILOG CODE

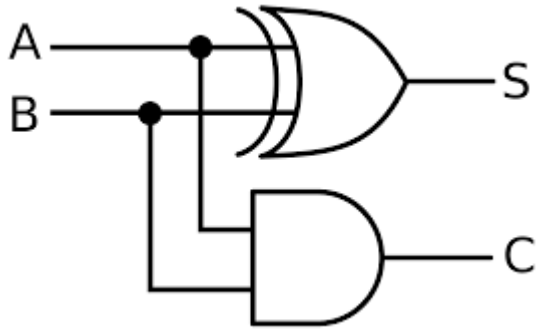
```
module logicgate(a,b,andgate,orgate,nandgate,norgate,xorgate,xnorgate,notgate);
input a,b;
output andgate,orgate,nandgate,norgate,xorgate,xnorgate,notgate;
and(andgate,a,b);
or(orgate,a,b);
nand(nandgate,a,b);
nor(norgate,a,b);
xor(xorgate,a,b);
xnor(xnorgate,a,b);
not(notgate,a);
endmodule
```

OUTPUT WAVEFORM



HALF ADDER

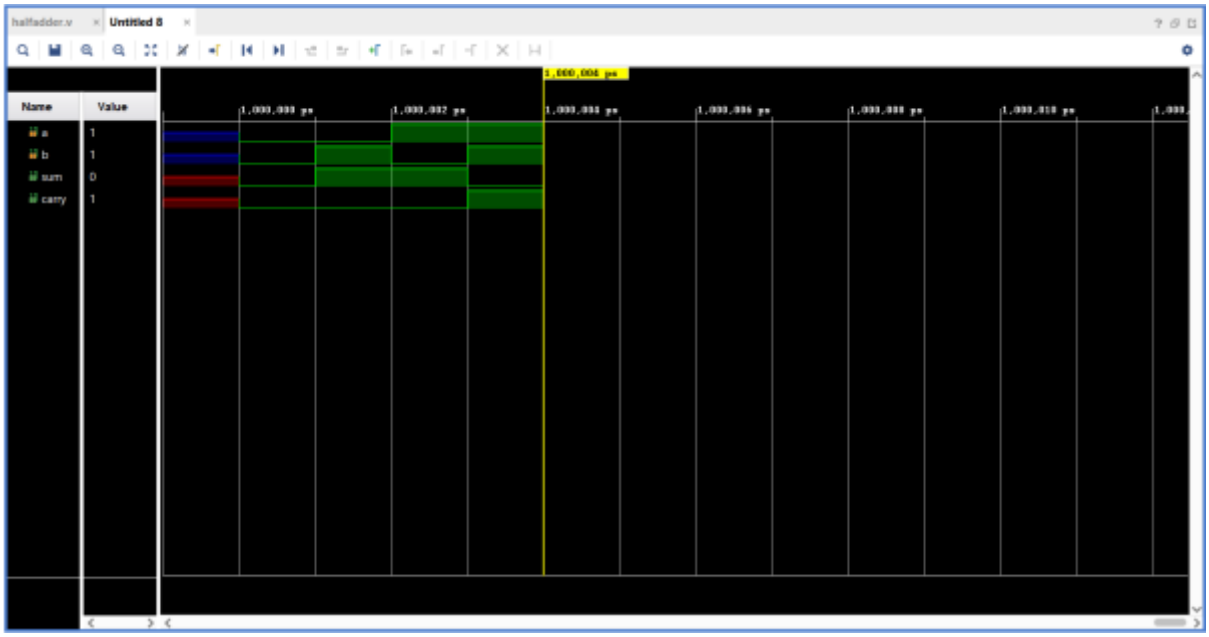
LOGIC DIAGRAM



VERILOG CODE

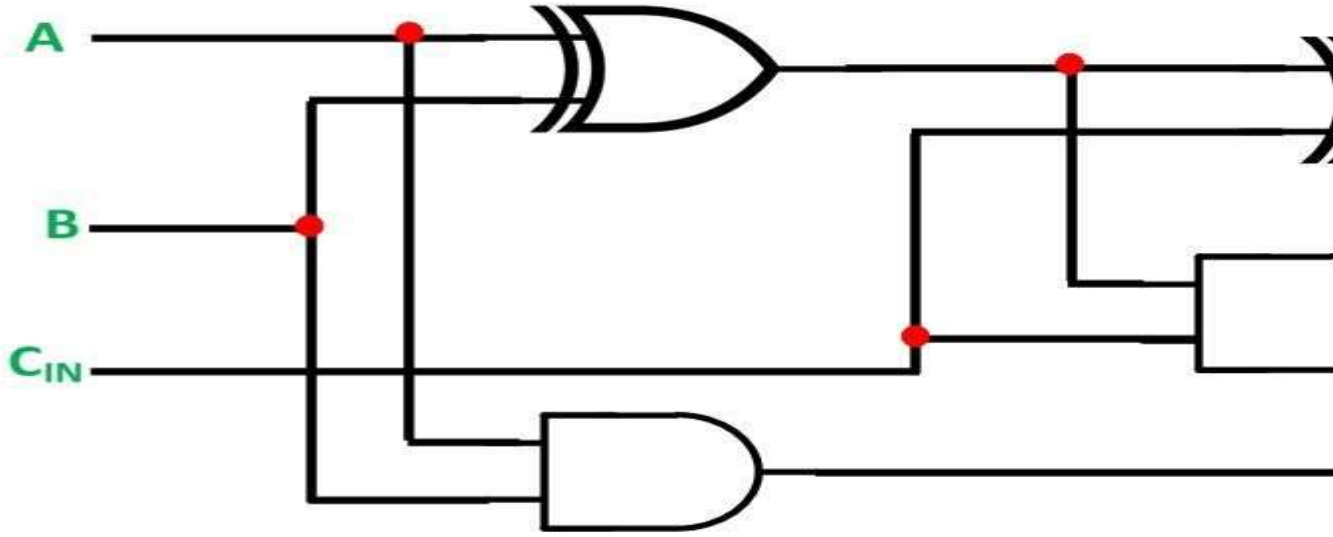
```
module half_adder(a,b,sum,carry);
input a,b;
output sum,carry;
xor g1(sum,a,b);
and g2(carry,a,b);
endmodule
```

OUTPUT WAVEFORM



FULL ADDER

LOGIC DIAGRAM



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);

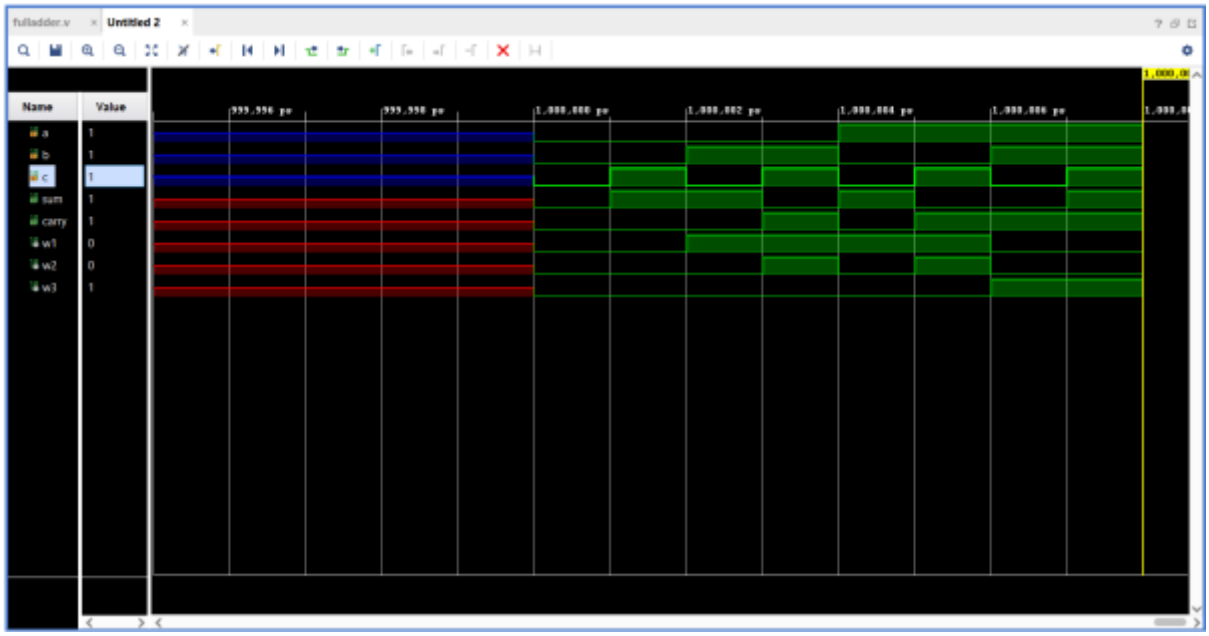
and(w2,w1,c);

and(w3,a,b);

or(carry,w2,w3);

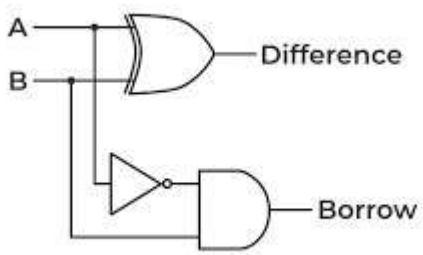
endmodule

OUTPUT WAVEFORM



HALF SUBTRACTOR

LOGIC DIAGRAM



VERILOG CODE

module halfsub(a,b,diff,borrow);

input a,b;

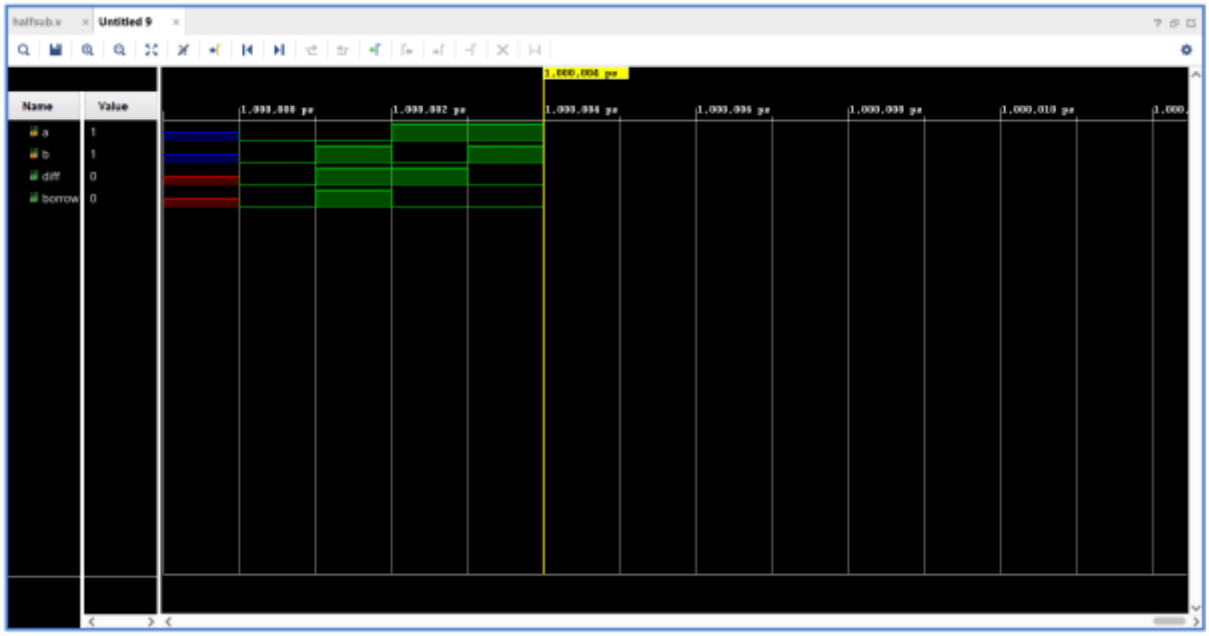
output diff,borrow;

xor(diff,a,b);

and(borrow,~a,b);

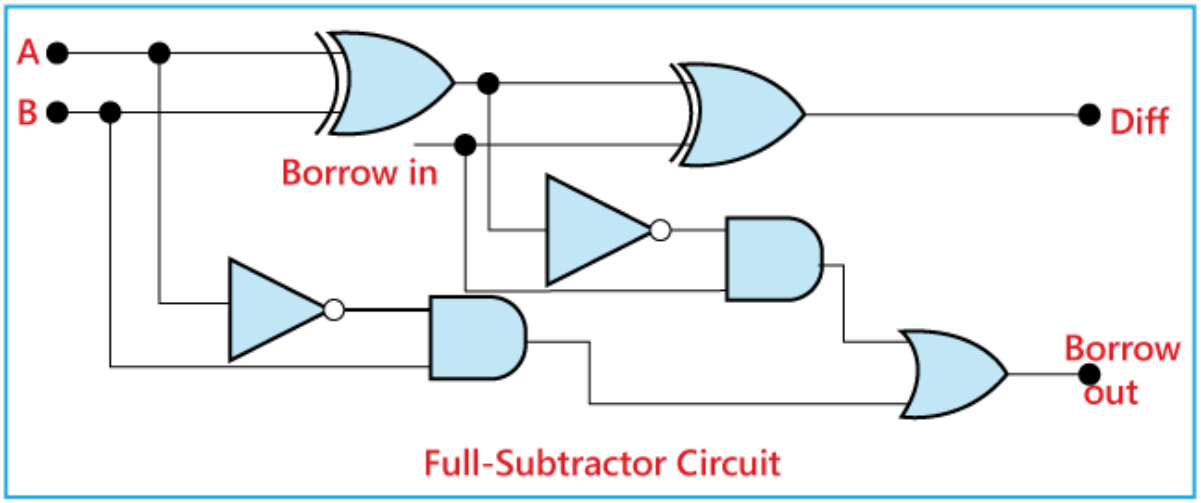
endmodule

OUTPUT WAVEFORM



FULL SUBTRACTOR

LOGIC DIAGRAM



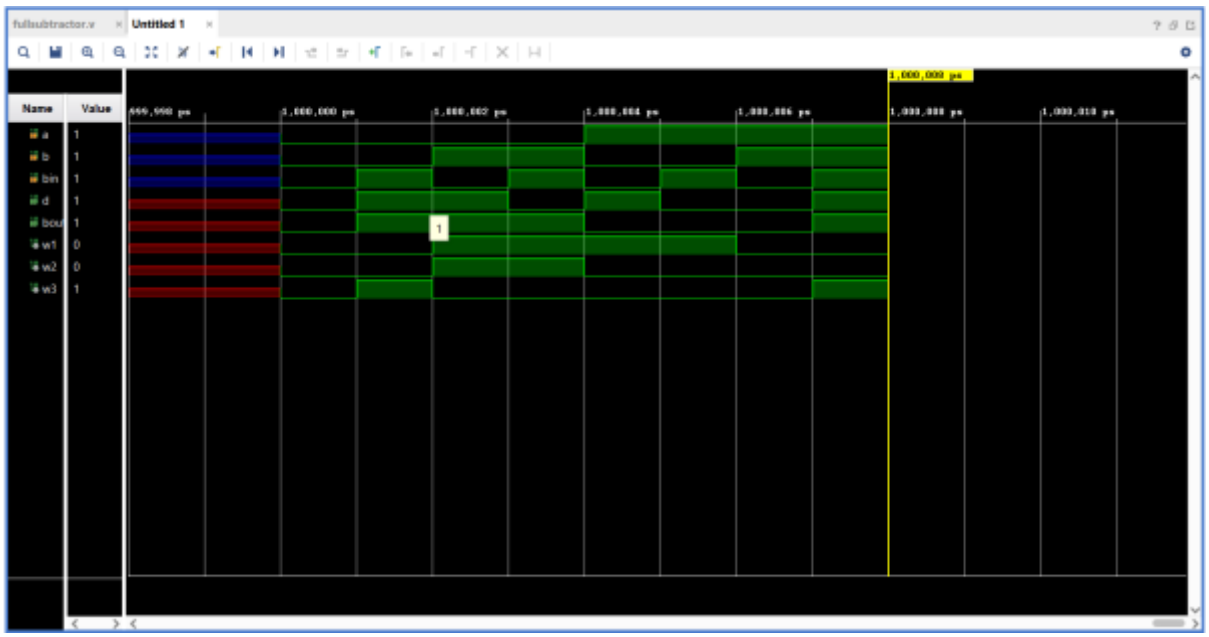
VERILOG CODE

```
module fs(a,b,bin,d,bout);

input a,b,bin;
```

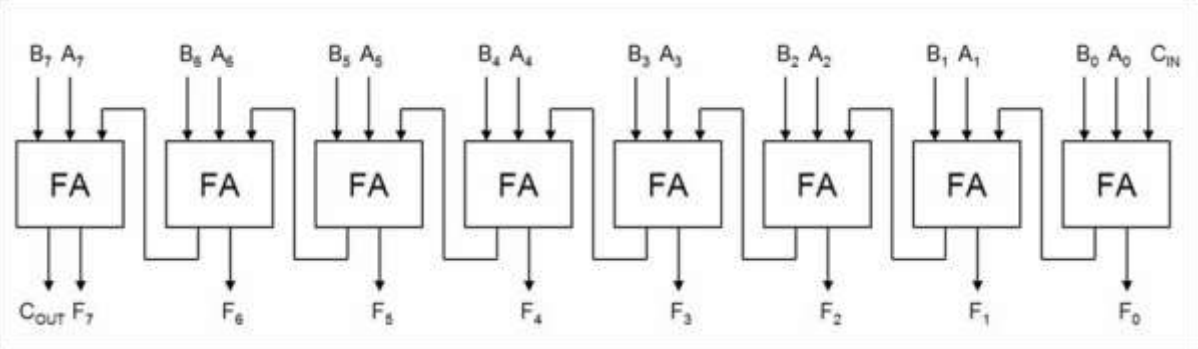
```
output d,bout;
wire w1,w2,w3;
xor(w1,a,b);
xor(d,w1,bin);
and(w2,~a,b);
and(w3,~w1,bin);
or(bout,w3,w2);
endmodule
```

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