 <b>Marwadi University</b>	<b>Marwadi University</b> <b>Faculty of Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Digital Design using Verilog (01CT0619)</b>	<b>Aim:</b> Write the Verilog code for designing 1 bit half adder and full adder circuit.	
<b>Experiment No: 03</b>	<b>Date:</b>	<b>Enrollment No: 92100133020</b>

**Aim:** Write the Verilog code for designing 1 bit half adder and full adder circuit.

**IDE:** EDA Playground

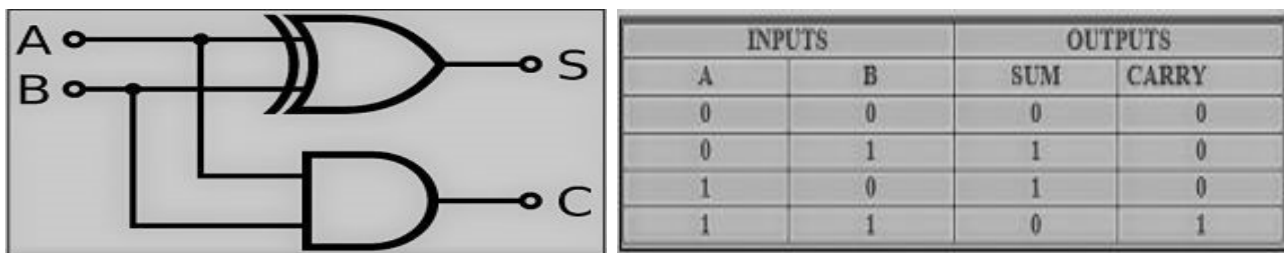
### Theory:

#### Half adder

It adds two binary digits where the input bits are termed as augend and addend and the result will be two outputs one is the sum and the other is carry. To perform the sum operation, XOR is applied to both the inputs, and AND gate is applied to both inputs to produce carry.

$$\text{Sum} = A \text{ XOR } B$$

$$\text{Carry} = A \text{ AND } B$$




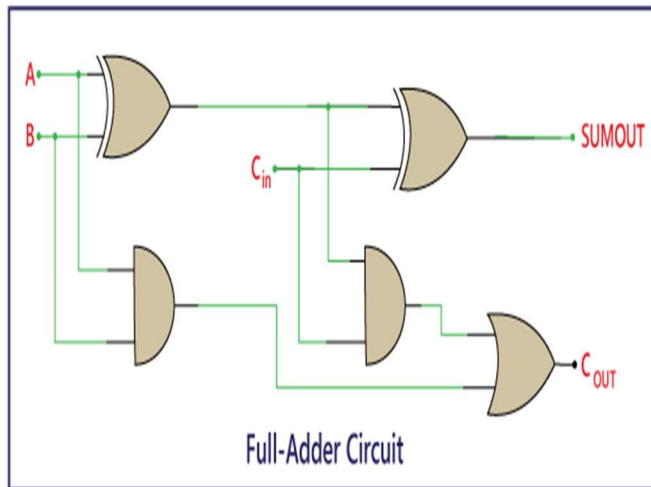
#### Full Adder

Full-adder has three inputs and two outputs, the first two inputs are A and B and the third input is an input carry as C-IN. When a full-adder logic is designed, you string eight of them together to create a byte-wide adder and cascade the carry bit from one adder to the next.

$$\text{Sum} = (A \oplus B) \oplus C_{in}$$

$$\text{Carry} = A.B + (A \oplus B)$$

 <b>Marwadi University</b>	<b>Marwadi University</b> <b>Faculty of Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Digital Design using Verilog (01CT0619)</b>	<b>Aim:</b> Write the Verilog code for designing 1 bit half adder and full adder circuit.	
<b>Experiment No: 03</b>	<b>Date:</b>	<b>Enrollment No: 92100133020</b>



Inputs			Outputs	
A	B	C <sub>in</sub>	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

### Pre Lab Exercise:

a. What is use of adder in processor?

---



---



---

b. What is limitation of half adder?

---



---



---

c. How many AND, OR and EXOR gates are required for the configuration of full adder?

---




---



---

### Verilog Code:

```
// Half Adder
module half_adder (in1, in2, out1, out2);
    input in1;
```

 <b>Marwadi</b> University	<b>Marwadi University</b> <b>Faculty of Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Digital Design using Verilog (01CT0619)</b>	<b>Aim:</b> Write the Verilog code for designing 1 bit half adder and full adder circuit.	
<b>Experiment No: 03</b>	<b>Date:</b>	<b>Enrollment No: 92100133020</b>

```
input in2;
output out1;
output out2;
```

```
xor(out1, in1, in2); // sum output
and(out2, in1, in2); // carry output
endmodule
```

### **Test-bench:**

```
module half_adder_tb;

reg in1;
reg in2;
wire out1;
wire out2;

half_adder uut(in1,in2,out1,out2);


initial begin

in1 = 0;
in2 = 0;
#10 // Delay of 10 nanoseconds
$display("%b",in1," - ", "%b",in2," -> ", " sum - ", "%b",out1," carry - ", "%b",out2);

in1 = 0;
in2 = 1;
#10 // Delay of 10 nanoseconds
$display("%b",in1," - ", "%b",in2," -> ", " sum - ", "%b",out1," carry - ", "%b",out2);

in1 = 1;
in2 = 0;
#10 // Delay of 10 nanoseconds
$display("%b",in1," - ", "%b",in2," -> ", " sum - ", "%b",out1," carry - ", "%b",out2);

in1 = 1;
in2 = 1;
```

 <b>Marwadi University</b>	<b>Marwadi University</b> <b>Faculty of Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Digital Design using Verilog (01CT0619)</b>	<b>Aim:</b> Write the Verilog code for designing 1 bit half adder and full adder circuit.	
<b>Experiment No: 03</b>	<b>Date:</b>	<b>Enrollment No: 92100133020</b>

```
#10 // Delay of 10 nanoseconds
```

```
$display("%b",in1," - ","%b",in2," -> "," sum - ","%b",out1," carry - ","%b",out2);
```

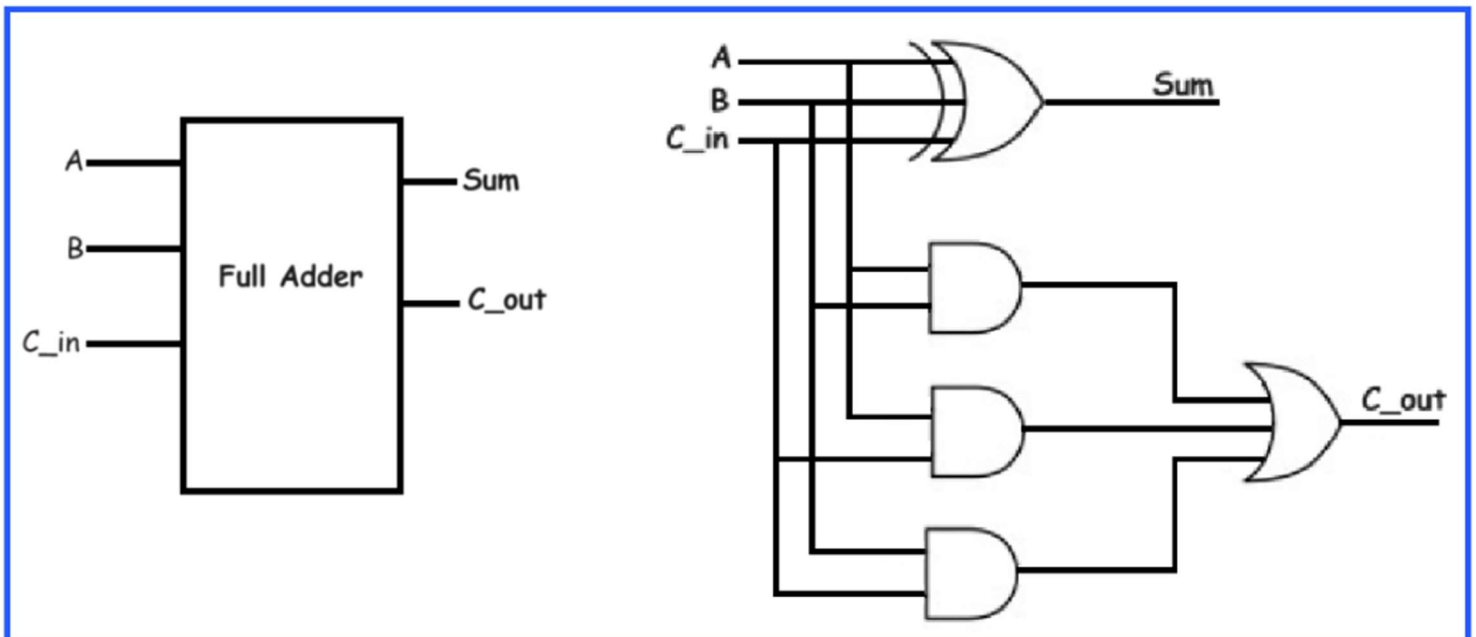
```
$finish();
```


```
end
```

```
endmodule
```

### Results:

```
PS D:\Mirror\ICT\3rd YEAR\SEM 6\DDV_Codes> iverilog half_adder.v
PS D:\Mirror\ICT\3rd YEAR\SEM 6\DDV_Codes> vvp a.out
0 - 0 -> sum - 0 carry - 0
0 - 1 -> sum - 1 carry - 0
1 - 0 -> sum - 1 carry - 0
1 - 1 -> sum - 0 carry - 1
half_adder.v:46: $finish called at 40 (1s)
```



 <b>Marwadi</b> University	<b>Marwadi University</b> <b>Faculty of Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Digital Design using Verilog (01CT0619)</b>	<b>Aim:</b> Write the Verilog code for designing 1 bit half adder and full adder circuit.	
<b>Experiment No: 03</b>	<b>Date:</b>	<b>Enrollment No: 92100133020</b>

### Verilog Code:

// Full Adder

```
module full_adder (in1, in2, cin, sum, cout);
    input in1, in2, cin;
    output sum, cout;
    wire w1;

    // sum output
    xor(w1, in1, in2);
    xor(sum, cin, w1);

    // carry output
    and(w2, in1, in2);
    and(w3, in1, cin);
    and(w4, cin, in2);

    or(w5, w2, w3);
    or(cout, w5, w4);

endmodule
```


### Test-bench:

```
module full_adder_tb;

    reg in1, in2, cin;
    wire sum, cout;
    reg [4:0] inputs; // 4-bit binary counter

    full_adder a(in1, in2, cin, sum, cout);

    initial begin
        for (inputs = 0; inputs < 8; inputs = inputs + 1) begin
            in1 = inputs[0]; // 0th bit for x1
            in2 = inputs[1]; // 1st bit for x2
```

 <b>Marwadi University</b>	<b>Marwadi University</b> <b>Faculty of Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Digital Design using Verilog (01CT0619)</b>	<b>Aim:</b> Write the Verilog code for designing 1 bit half adder and full adder circuit.	
<b>Experiment No: 03</b>	<b>Date:</b>	<b>Enrollment No: 92100133020</b>

```

cin = inputs[2]; // 2nd bit for x3
#10000; // Delay of 1 second

// Display inputs and corresponding outputs
$display("Cin = %b | Input 2 = %b | Input 1 = %b -> Sum : %b | Carry : %b ", cin, in2, in1, sum, cout);
end
$finish;
end
endmodule

```

### Results:

```

PS D:\Mirror\ICT\3rd YEAR\SEM 6\DDV_Codes> iverilog full_adder.v
PS D:\Mirror\ICT\3rd YEAR\SEM 6\DDV_Codes> vvp a.out
Cin = 0 | Input 2 = 0 | Input 1 = 0 -> Sum : 0 | Carry : 0
Cin = 0 | Input 2 = 0 | Input 1 = 1 -> Sum : 1 | Carry : 0
Cin = 0 | Input 2 = 1 | Input 1 = 0 -> Sum : 1 | Carry : 0
Cin = 0 | Input 2 = 1 | Input 1 = 1 -> Sum : 0 | Carry : 1
Cin = 1 | Input 2 = 0 | Input 1 = 0 -> Sum : 1 | Carry : 0
Cin = 1 | Input 2 = 0 | Input 1 = 1 -> Sum : 0 | Carry : 1
Cin = 1 | Input 2 = 1 | Input 1 = 0 -> Sum : 0 | Carry : 1
Cin = 1 | Input 2 = 1 | Input 1 = 1 -> Sum : 1 | Carry : 1
full_adder.v:43: $finish called at 80000 (1s)

```

### Observation and Result Analysis:

Write the final observation and Analysis

---

---


---

---

---


---

---

 <b>Marwadi</b> University	<b>Marwadi University</b> <b>Faculty of Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Digital Design using Verilog (01CT0619)</b>	<b>Aim:</b> Write the Verilog code for designing 1 bit half adder and full adder circuit.	
<b>Experiment No: 03</b>	<b>Date:</b>	<b>Enrollment No: 92100133020</b>

**Post Lab Exercise:**

- a. Design One full adder using two half adder and one or gate.

 <b>Marwadi University</b>	<b>Marwadi University</b> <b>Faculty of Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Digital Design using Verilog (01CT0619)</b>	<b>Aim:</b> Write the Verilog code for designing 1 bit half adder and full adder circuit.	
<b>Experiment No: 03</b>	<b>Date:</b>	<b>Enrollment No: 92100133020</b>

### Verilog Code:

// Full Adder using 2 Half Adder and 1 OR Gate

```
module half_adder (in1, in2, sum, carry);
    input in1;
    input in2;
    output sum;
    output carry;

    xor(sum, in1, in2); // sum output
    and(carry, in1, in2); // carry output
endmodule
```

### Test-bench:


```
module full_adder_tb;

    reg in1, in2, cin;
    wire sum1, sum2, cout, carry1, carry2;
    reg [4:0] inputs; // 4-bit binary counter

    half_adder a(in1,in2,sum1,carry1);
    half_adder b(sum1,cin,sum,carry2);
    or c(cout, carry1, carry2);

    initial begin
        for (inputs = 0; inputs < 8; inputs = inputs + 1) begin
            in1 = inputs[0]; // 0th bit for x1
            in2 = inputs[1]; // 1st bit for x2
            cin = inputs[2]; // 2nd bit for x3
            #10000; // Delay of 1 second
            // Display inputs and corresponding outputs
            $display("Cin = %b | Input 2 = %b | Input 1 = %b -> Sum : %b | Carry : %b ", cin, in2, in1, sum, cout);
        end
        $finish;
    end
endmodule
```



 <b>Marwadi University</b>	<b>Marwadi University</b> <b>Faculty of Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Digital Design using Verilog (01CT0619)</b>	<b>Aim:</b> Write the Verilog code for designing 1 bit half adder and full adder circuit.	
<b>Experiment No: 03</b>	<b>Date:</b>	<b>Enrollment No: 92100133020</b>

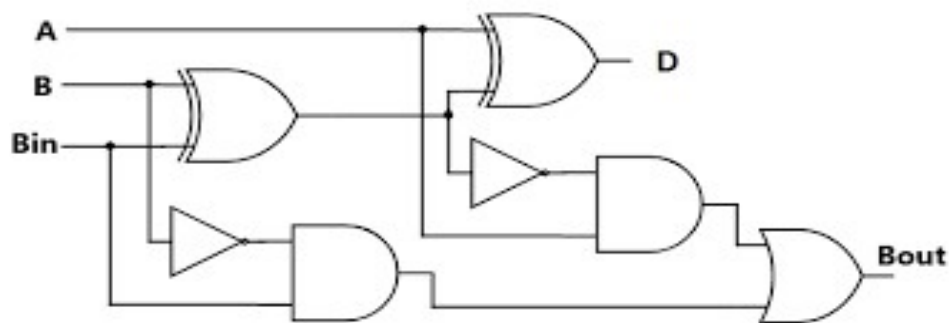
### Results:

```


PS D:\Mirror\ICT\3rd YEAR\SEM 6\DDV_Codes> iverilog full_adder_using_half_adder.v
PS D:\Mirror\ICT\3rd YEAR\SEM 6\DDV_Codes> vvp a.out
Cin = 0 | Input 2 = 0 | Input 1 = 0 -> Sum : 0 | Carry : 0
Cin = 0 | Input 2 = 0 | Input 1 = 1 -> Sum : 1 | Carry : 0
Cin = 0 | Input 2 = 1 | Input 1 = 0 -> Sum : 1 | Carry : 0
Cin = 0 | Input 2 = 1 | Input 1 = 1 -> Sum : 0 | Carry : 1
Cin = 1 | Input 2 = 0 | Input 1 = 0 -> Sum : 1 | Carry : 0
Cin = 1 | Input 2 = 0 | Input 1 = 1 -> Sum : 0 | Carry : 1
Cin = 1 | Input 2 = 1 | Input 1 = 0 -> Sum : 0 | Carry : 1
Cin = 1 | Input 2 = 1 | Input 1 = 1 -> Sum : 1 | Carry : 1
full_adder_using_half_adder.v:42: $finish called at 80000 (1s)

```

### b. Design full subtractor circuit using Data Flow Modeling Style



Inputs			Outputs	
A	B	Borrow <sub>in</sub>	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

 <b>Marwadi University</b>	<b>Marwadi University</b> <b>Faculty of Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Digital Design using Verilog (01CT0619)</b>	<b>Aim:</b> Write the Verilog code for designing 1 bit half adder and full adder circuit.	
<b>Experiment No: 03</b>	<b>Date:</b>	<b>Enrollment No: 92100133020</b>

### Verilog Code:

// Dataflow Model of Full Subtractor

```

module full_subtractor(A, B, Bin, Diff, Bout);
    input A, B, Bin;
    output Diff, Bout;
    wire w1;

    // Difference
    assign Diff = A ^ B ^ Bin;

    // Borrow
    assign Bout = (~A & B) | (~A & Bin) | (B & Bin);
endmodule

```

### Test-bench:

```

module full_subtractor_tb;


    reg in1, in2, Bin;
    wire Diff, Bout;
    reg [4:0] inputs; // 4-bit binary counter

    full_subtractor a(in1, in2, Bin, Diff, Bout);

    initial begin
        for (inputs = 0; inputs < 8; inputs = inputs + 1) begin
            Bin = inputs[0]; // 0th bit for x1
            in2 = inputs[1]; // 1st bit for x2
            in1 = inputs[2]; // 2nd bit for x3
            #10000; // Delay of 1 second

            // Display inputs and corresponding outputs
            $display("Input 1 = %b | Input 2 = %b | Bin = %b -> Difference : %b | Borrow : %b ", in1, in2, Bin, Diff, Bout);
        end
    end

```

 <b>Marwadi University</b>	<b>Marwadi University</b> <b>Faculty of Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Digital Design using Verilog (01CT0619)</b>	<b>Aim:</b> Write the Verilog code for designing 1 bit half adder and full adder circuit.	
<b>Experiment No: 03</b>	<b>Date:</b>	<b>Enrollment No: 92100133020</b>

```

$finish;
end
endmodule

```

### Results:

```

PS D:\Mirror\ICT\3rd YEAR\SEM 6\DDV_Codes> iverilog full_subtractor.v
PS D:\Mirror\ICT\3rd YEAR\SEM 6\DDV_Codes> vvp a.out
Input 1 = 0 | Input 2 = 0 | Bin = 0 -> Difference : 0 | Borrow : 0
Input 1 = 0 | Input 2 = 0 | Bin = 1 -> Difference : 1 | Borrow : 1
Input 1 = 0 | Input 2 = 1 | Bin = 0 -> Difference : 1 | Borrow : 1
Input 1 = 0 | Input 2 = 1 | Bin = 1 -> Difference : 0 | Borrow : 1
Input 1 = 1 | Input 2 = 0 | Bin = 0 -> Difference : 1 | Borrow : 0
Input 1 = 1 | Input 2 = 0 | Bin = 1 -> Difference : 0 | Borrow : 0
Input 1 = 1 | Input 2 = 1 | Bin = 0 -> Difference : 0 | Borrow : 0
Input 1 = 1 | Input 2 = 1 | Bin = 1 -> Difference : 1 | Borrow : 1
full_subtractor.v:33: $finish called at 80000 (1s)

```