**DIGITAL LOGIC DESIGN LAB (EET1211)**

**LAB IV: Construct and Test various Binary Adder and Subtractor circuits Using HDL**

**Siksha ‘O’ Anusandhan Deemed to be University, Bhubaneswar**

|  |  |  |  |
| --- | --- | --- | --- |
| **Branch:** Computer Science and Engineering **Section:** D | | | |
| **S. No.** | **Name** | **Registration No.** | **Signature** |
| 1 | Saswat Mohanty | 1941012407 | **E:\sign.jpg** |

**Marks: \_\_\_\_\_\_/10**

**Remarks:**

**Teacher’s Signature**

**I. OBJECTIVE:**

1. Design, construct and test a Full Adder circuit using two ICs, 7486(XOR) and 7400(NAND).
2. Design, construct, and test a Half­ Subtractor circuit using two ICs, 7486(XOR) and 7400(NAND).
3. Design, construct, and test a 2 bit Parallel Adder circuit.

**II. PRE-LAB**

**For Objective 1:**

1. **Write the truth table for full-adder logic.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **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 |

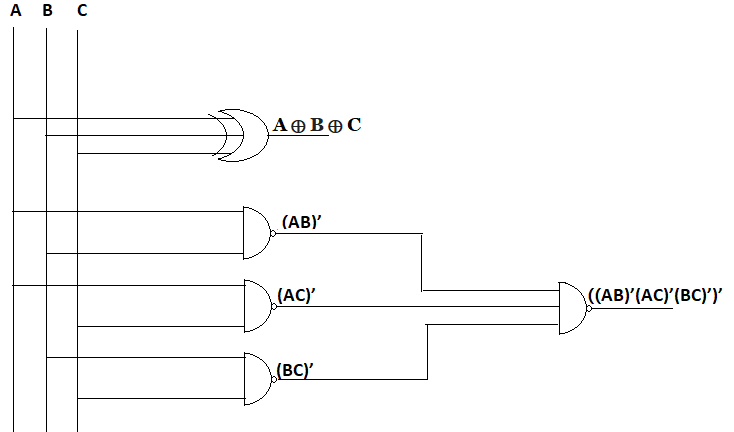
1. **Derive the Boolean expression for sum and carry using XOR and NAND operation respectively.**

**Sum:-** A ⊕ B ⊕ C

**Carry:-** AB + AC + BC

**Nand:-** ((AB)’(AC)’(BC)’)’

1. **Draw the logic diagram for sum and carry as output of the full-adder circuit.**



1. **Write HDL code.**

**design.sv:**

*`default\_nettype none*

*module mod (A,B,C,Sum,Carry);*

*input A,B,C;*

*output Sum,Carry;*

*wire P,Q,R,S;*

*// dataflow model*

*assign Sum=A^B^C;*

*assign Carry=~(~(A&&B)&&~(A&&C)&&~(B&&C));*

*// gate-level model*

*xor x1(S,A,B),*

*x2(Sum,S,C);*

*nand n1(P,A,B),*

*n2(Q,A,C),*

*n3(R,B,C),*

*n4(Carry,P,Q,R);*

*endmodule*

**For Obj. 2:**

1. **Write the truth table for Half­ Subtractor logic.**

|  |  |  |  |
| --- | --- | --- | --- |
| **A** | **B** | **Diff** | **Borrow** |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |

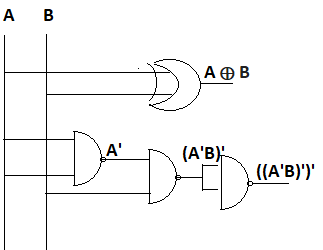
1. **Derive the Boolean expression for difference and borrow using XOR and NAND operation respectively.**

**Difference:-** A ⊕ B

**Borrow:-** X’Y

**Nand:-** ((X’Y)’)’

1. **Draw the logic diagram for difference and borrow as output of the half-subtractor circuit.**



1. **Write HDL code.**

**design.sv:**

*`default\_nettype none*

*module mod (*

*input A,*

*input B,*

*output s\_D,s\_B*

*);*

*wire P;*

*// dataflow model*

*assign s\_D = A ^ B;*

*assign s\_B = ~(A && ~B);*

*// gate-level model*

*xor(s\_D,A,B);*

*nand n1(P,~A,B),*

*n2(s\_B,P,P);*

*endmodule*

**For Obj. 3:**

1. **Write the truth table for 2-bit parallel adder.**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **A1** | **A0** | **B1** | **B0** | **C0** | **S0** | **S1** | **C1** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

1. **Derive the Boolean expression for sum bit and carry bit using XOR and NAND operation respectively.**

**S0:-** A0 ⊕ B0

**C0:-** A0.B0

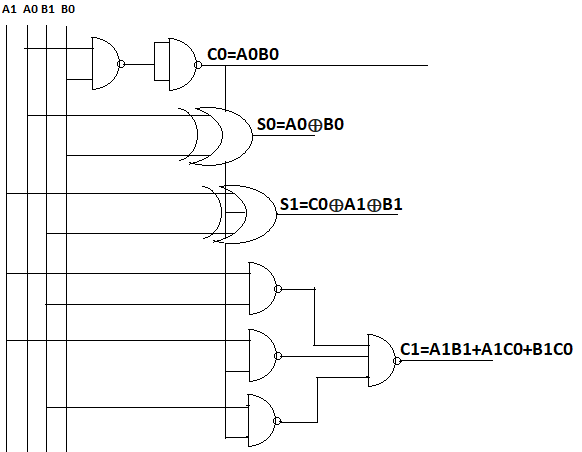
**Nand:-** ((A0.B0)’)’

**S1:-** C0 ⊕ A1 ⊕ B1

**C1:-** A1.B1 +A1.C0 + B1.C0

**Nand:-** ((A1.B1)’(B1.C0)’(A1.C0)’)’

1. **Draw the logic diagram based on Boolean expression.**



1. **Write HDL code.**

**design.sv:**

*`default\_nettype none*

*module lab4 (*

*input A1,*

*input A0,*

*input B1,*

*input B0,*

*output S0,S1,C0,C1*

*);*

*wire P,Q,R,S,T;*

*// dataflow model*

*assign S0=A0^B0;*

*assign P=~(A0&&B0);*

*assign C0=~(P&&P);*

*assign S1=C0^A1^B1;*

*assign Q=~(A1&&B1);*

*assign R=~(B1&&C0);*

*assign S=~(A1&&C0);*

*assign C1=~(Q&&R&&S);*

*// gate-level model*

*xor(S0,A0,B0);*

*nand(P,A0,B0);*

*nand(C0,P,P);*

*xor(S1,C0,A1,B1);*

*nand(Q,A1,B1);*

*nand(R,A1,C0);*

*nand(S,B1,C0);*

*nand(C1,Q,R,S);*

*endmodule*

**III. LAB:**

1. **Design, construct and test a Full Adder circuit using two ICs, 7486(XOR) and 7400(NAND).**

**Code:-**

**design.sv:**

*`default\_nettype none*

*module mod (A,B,C,Sum,Carry);*

*input A,B,C;*

*output Sum,Carry;*

*wire P,Q,R,S;*

*// dataflow model*

*assign Sum=A^B^C;*

*assign Carry=~(~(A&&B)&&~(A&&C)&&~(B&&C));*

*// gate-level model*

*xor x1(S,A,B),*

*x2(Sum,S,C);*

*nand n1(P,A,B),*

*n2(Q,A,C),*

*n3(R,B,C),*

*n4(Carry,P,Q,R);*

*endmodule*

**testbench.sv:**

*`default\_nettype none*

*module dl\_mod;*

*reg i\_a, i\_b, i\_c;*

*wire out\_sum, out\_carry;*

*mod h\_dut(i\_a,i\_b,i\_c,out\_sum,out\_carry);*

*initial*

*begin*

*$dumpfile("dump.vcd");*

*$dumpvars(0, h\_dut);*

*$display("Lab 4 Obj 1");*

*#1*

*i\_a <= 0;*

*i\_b <= 0;*

*i\_c <= 0;*

*#1*

*#1*

*i\_a <= 0;*

*i\_b <= 0;*

*i\_c <= 1;*

*#1*

*#1*

*i\_a <= 0;*

*i\_b <= 1;*

*i\_c <= 0;*

*#1*

*#1*

*i\_a <= 0;*

*i\_b <= 1;*

*i\_c <= 1;*

*#1*

*#1*

*i\_a <= 1;*

*i\_b <= 0;*

*i\_c <= 0;*

*#1*

*#1*

*i\_a <= 1;*

*i\_b <= 0;*

*i\_c <= 1;*

*#1*

*#1*

*i\_a <= 1;*

*i\_b <= 1;*

*i\_c <= 0;*

*#1*

*#1*

*i\_a <= 1;*

*i\_b <= 1;*

*i\_c <= 1;*

*#1*

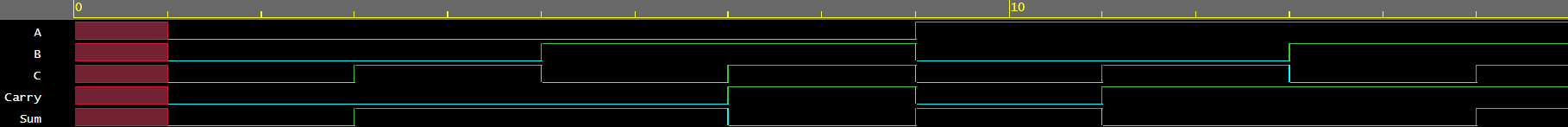
*$finish();*

*end*

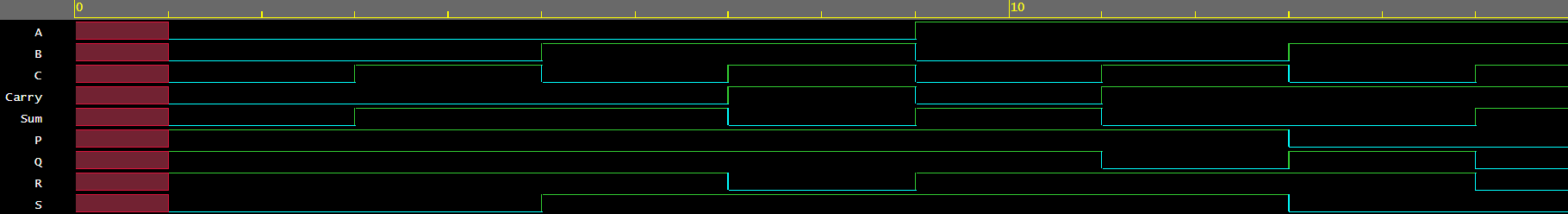
*endmodule*

**Link:-** <https://www.edaplayground.com/x/wZcW>

**EP Waveform:-**

****

***Dataflow model***

******

***Gate-level model***

**Observation:-**

The following Truth Table was obtained from the above EP Waveform:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **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 |

1. **Design, construct, and test a Half­ Subtractor circuit using two ICs, 7486(XOR) and 7400(NAND).**

**Code:-**

**design.sv:**

*`default\_nettype none*

*module mod (*

*input A,*

*input B,*

*output s\_D,s\_B*

*);*

*wire P;*

*// dataflow model*

*assign s\_D = A ^ B;*

*assign s\_B = ~(A && ~B);*

*// gate-level model*

*xor(s\_D,A,B);*

*nand n1(P,~A,B),*

*n2(s\_B,P,P);*

*endmodule*

**testbench.sv:**

*`default\_nettype none*

*module tb\_mod;*

*reg a, b;*

*wire sd,sb;*

*mod h\_dut(a,b,sd,sb);*

*initial*

*begin*

*$dumpfile("dump.vcd");*

*$dumpvars(0, h\_dut);*

*$display("Lab 4 Obj 2");*

*#1*

*a <= 0;*

*b <= 0;*

*#1*

*#1*

*a <= 0;*

*b <= 1;*

*#1*

*#1*

*a <= 1;*

*b <= 0;*

*#1*

*#1*

*a <= 1;*

*b <= 1;*

*#1*

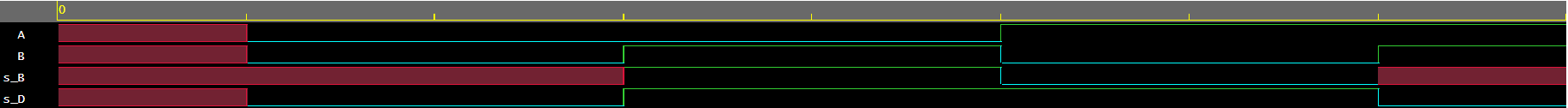
*$finish();*

*end*

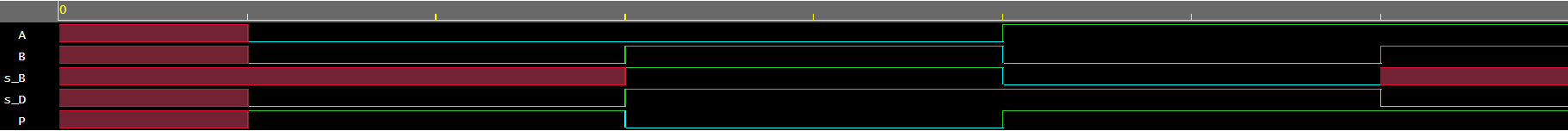
*endmodule*

**Link:-** <https://www.edaplayground.com/x/fVu9>

**EP Waveform:-**

****

***Dataflow model***

******

***Gate-level model***

**Observation:-**

The following Truth Table was obtained from the above EP Waveform:

|  |  |  |  |
| --- | --- | --- | --- |
| **A** | **B** | **Diff** | **Borrow** |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |

1. **Design, construct, and test a 2 bit Parallel Adder circuit.**

**Code:-**

**design.sv:**

*`default\_nettype none*

*module lab4 (*

*input A1,*

*input A0,*

*input B1,*

*input B0,*

*output S0,S1,C0,C1*

*);*

*wire P,Q,R,S,T;*

*// dataflow model*

*assign S0=A0^B0;*

*assign P=~(A0&&B0);*

*assign C0=~(P&&P);*

*assign S1=C0^A1^B1;*

*assign Q=~(A1&&B1);*

*assign R=~(B1&&C0);*

*assign S=~(A1&&C0);*

*assign C1=~(Q&&R&&S);*

*// gate-level model*

*xor(S0,A0,B0);*

*nand(P,A0,B0);*

*nand(C0,P,P);*

*xor(S1,C0,A1,B1);*

*nand(Q,A1,B1);*

*nand(R,A1,C0);*

*nand(S,B1,C0);*

*nand(C1,Q,R,S);*

*endmodule*

**testbench.sv:**

*`default\_nettype none*

*module dl\_lab4;*

*reg a, b, c, d,cin;*

*wire S0,S1,C0,C1;*

*lab4 parms(a,b,c,d,S0,S1,C0,C1);*

*initial*

*begin*

*$dumpfile("dump.vcd");*

*$dumpvars(0, parms);*

*$display("Lab 4 Obj 3");*

*#1*

*a<=0;*

*b<=0;*

*c<=0;*

*d<=0;*

*#1*

*#1*

*a<=0;*

*b<=0;*

*c<=0;*

*d<=1;*

*#1*

*#1*

*a<=0;*

*b<=0;*

*c<=1;*

*d<=0;*

*#1*

*#1*

*a<=0;*

*b<=0;*

*c<=1;*

*d<=1;*

*#1*

*#1*

*a<=0;*

*b<=1;*

*c<=0;*

*d<=0;*

*#1*

*#1*

*a<=0;*

*b<=1;*

*c<=0;*

*d<=1;*

*#1*

*#1*

*a<=0;*

*b<=1;*

*c<=1;*

*d<=0;*

*#1*

*#1*

*a<=0;*

*b<=1;*

*c<=1;*

*d<=1;*

*#1*

*#1*

*a<=1;*

*b<=0;*

*c<=0;*

*d<=0;*

*#1*

*#1*

*a<=1;*

*b<=0;*

*c<=0;*

*d<=1;*

*#1*

*#1*

*a<=1;*

*b<=0;*

*c<=1;*

*d<=0;*

*#1*

*#1*

*a<=1;*

*b<=0;*

*c<=1;*

*d<=1;*

*#1*

*#1*

*a<=1;*

*b<=1;*

*c<=0;*

*d<=0;*

*#1*

*#1*

*a<=1;*

*b<=1;*

*c<=0;*

*d<=1;*

*#1*

*#1*

*a<=1;*

*b<=1;*

*c<=1;*

*d<=0;*

*#1*

*#1*

*a<=1;*

*b<=1;*

*c<=1;*

*d<=1;*

*#1*

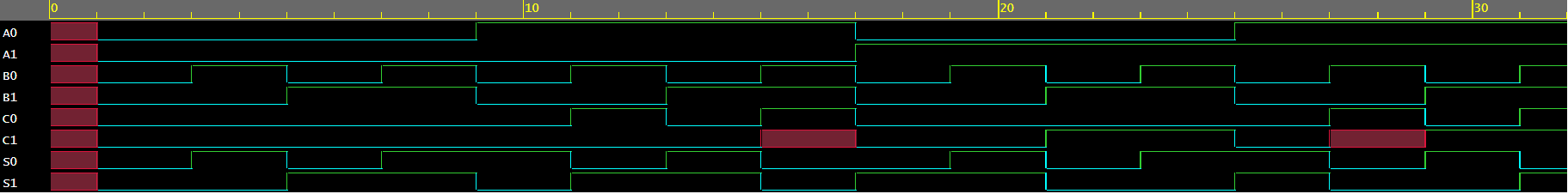
*$finish();*

*end*

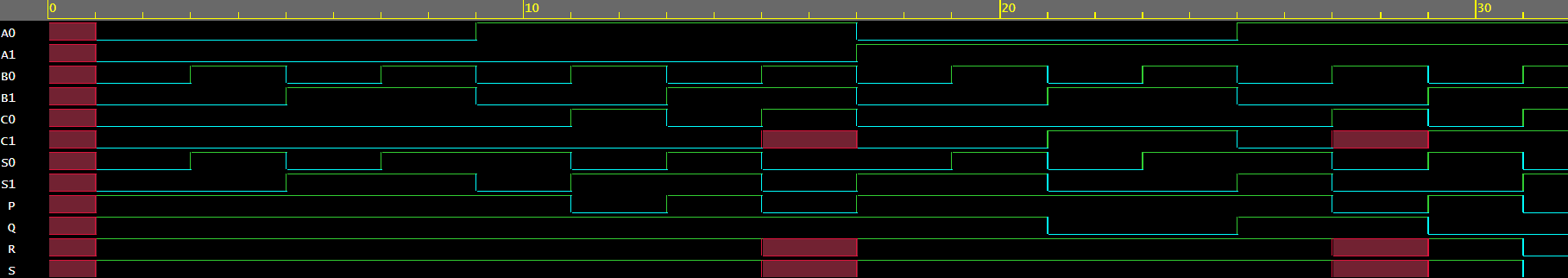
*endmodule*

**Link:-** <https://www.edaplayground.com/x/DaWc>

**EP Waveform:-**

****

***Dataflow model***

******

***Gate-level model***

**Observation:-**

The following Truth Table was obtained from the above EP Waveform:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **A1** | **A0** | **B1** | **B0** | **C0** | **S0** | **S1** | **C1** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

**Conclusion:**

**Objective 1:** It can be concluded that a full adder can add 3 binary bit using 7486 and 7400 ICs.

**Objective 2:** It can be concluded that a half subtractor can subtract 2 binary bit using 7486 and 7400 ICs.

**Objective 3:** It can be concluded that a 2-bit parallel adder can add 2 2 binary bit binary number using 7486 and 7400 ICs.

**IV. POST LAB:**

**A Half-adder is characterized by**

|  |  |  |  |
| --- | --- | --- | --- |
| **a.** | **Two inputs and two outputs** | **b.** | **Three inputs and two outputs** |
| **c.** | **Two inputs and three outputs** | **d.** | **Two inputs and one outputs** |

**Ans:-** a) Two inputs and two outputs

* + - 1. **A 4-bit parallel adder can add**

|  |  |  |  |
| --- | --- | --- | --- |
| **a.** | **Two 4-bit binary numbers** | **b.** | **Two 2-bit binary number** |
| **c.** | **Four bits at a time** | **d.** | **Four bits at a time** |

**Ans:-** a) Two 4-bit binary numbers

* + - 1. **Two four bit numbers can be added using two full adders. Yes or No? Justify answer.**

**Ans:-** No because to add two 4 bit numbers to produce 4 bit sum with possible carry four full adders are requires.

**V. HDL PROGRAM LINK:**

**Objective 1:**  <https://www.edaplayground.com/x/wZcW>

**Objective 2:**  <https://www.edaplayground.com/x/fVu9>

**Objective 3:**  <https://www.edaplayground.com/x/DaWc>