**Lab Experiment-3**

**Title of the Experiment:** To design and Verify **4- Bit Binary Adder and Subtractor** using Verilog code and compare with their respective truth tables.

**Objective/Motivation:** In this lab, Adder - Subtractor circuit is designed and which is arithmetic digital circuit that can perform both addition and subtraction using only one n-bit adder, no separate circuit for addition or subtraction is required. The objective will be to test these designs on Xilinx simulation tool. The tests will be performed for all the possible combinations of inputs to verify their functionality. Moreover, the knowledge gained will be used to design much larger and complex logic designs.

**Equipment required:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S No** | **Name of The Components/Tool** | **Version** | **Quantity** |
| 1. | Xilinx Vivado Design Suite **/** EDA play Ground software | V23.1 | 1 |
| 2. | Zybo board | Zynq XC7Z010 | 1 |
| 3. | Personal Computer | - | 1 |

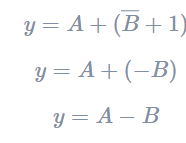
**Logic diagram(s) and Truth tables:**

|  |  |
| --- | --- |
| **Logic Circuit name** | **Logic Diagram** |
| **Half Adder** |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Logic Circuit name** | **Truth Table** | | | | | | |
| **4 Bit binary Adder-Subtractor** |  | **Full Adder Input** | | | | **Output** | |
| When Adding | | | | | | |
| **Crtl /a\_s** | **a**  **(DE=6)** | B  (DE=4) | Bmod | C / a\_s | Sum | Carry |
| 0 | **0** | 0 | **0** | 0 | **1** | **c1= 0** |
| **1** | 1 | **1** | **c1= 0** | **0** | **c2=0** |
| **1** | 0 | **0** | **c2=0** | **1** | **c3=0** |
| **0** | 0 | **0** | **c3=0** | **0** | **Carry(c4)=0** |
| **When Subtracting** | | | | | | |
| **Crtl /a\_s** | **a**  **(DE=6)** | B  (DE=4) | Bmod | **Barrow / a\_s** | **Diff** | **Barrow** |
| 1 | **0** | 0 | **1** | c=a\_s=1 | **0** | b1=1 |
| **1** | 1 | **0** | b1=1 | **0** | b2=1 |
| **1** | 0 | **1** | b2=1 | **1** | b3=1 |
| **0** | 0 | **1** | b3=1 | **0** | **Barrow(b4)=1** |
| * When **Crtl /a\_s =0**, then Full Adder performs Addition, **Crtl /a\_s =1** then Full Adder performs subtraction**.** * B is connected to the one input of n-bit ex-or gates and 2nd input is ctrl or a\_s. So wire **Bmod** is **B** when input **a\_s / CTRL** is **0**. Wire **Bmod** is **complement of B** if **a\_s / CTRL** is 1. * **Sum\_diff =a^bmod^a\_s** * **carry=(a&b)|(b&a\_s)|(a\_s&a);** * **barrow=(~(a)&b)|(b&a\_s)|(a\_s&~(a));** * For initial bit Operation of Addition / Subtraction, a\_s= Ctrl, but from 2nd bit operation onwards a\_s=c1,c2,c3 in addition, a\_s= b1,b2,b3 in subtraction. | | | | | | |

**Operational Summary:**

* 4-bit Binary Adder-Subtractor circuit is the **arithmetic digital circuit**, which can perform both addition and subtraction **using only one n-bit adder**, no separate circuit for addition or subtraction is required.
* The logic of the circuit is that, we can represent a negative number in 2's complement number system by taking it's complemented and then adding 1 into it. So, if we want wo perform operation y = A - B, we can perform y = A + (-B) and the result is same y = A - B. So, we require only adder to perform subtraction operation if we can convert positive input to the negative input.



* We need to implement an adder. Let's use n-bit [Ripple Carry Adder](https://circuitfever.com/ripple-carry-adder-verilog-code) for this. Now, only we need to write code to implement the n-bit inverter to make one input negative number. If carry in (Cin) is logic 1 and one number is negative, the adder will compute y = A-B. If carry in is logic zero and both the number is same as the input, then it computer y = A + B.
* The complete Verilog code for n-bit adder-subtractor circuit is given above. There are two inputs numbers A, B and one input (**a\_s / CTRL**) **for performing adder/subtraction**.
* Initially, the input **a\_s / CTRL** is connected to the cin and ex-or gate 2nd input of the ripple carry adder.
* A is directly connected to the input A of ripple carry adder and
* B is connected to the one input of n-bit ex-or gates. So wire **Bmod** is **B** when input **a\_s / CTRL** is **0**. Wire **Bmod** is **complement of B** if **a\_s / CTRL** is 1.

**Verilog Source Code for 4- Bit Binary Adder and Binary Subtractor** **in various Models or various Level of Abstrction:**

|  |  |  |
| --- | --- | --- |
| **Gate name** | **Data Flow Model** | **Test Bench** |
| **4- Bit Binary Adder and Binary Subtractor** | //Full Adder  module full\_adder(a,b,a\_s,sum,carry);  input a,b,a\_s;  output sum,carry;  assign sum=a^b^a\_s;  assign carry=(a&b)|(b&a\_s)|(a\_s&a);  endmodule  //rca(ripple carry adder)  **module rca(input [3:0]a,b,input a\_s,output [3:0]sum,output c4);**  **wire c1,c2,c3; //Carry out of each full adder**  **full\_adder fa0(a[0],b[0],a\_s,sum[0],c1);**  **full\_adder fa1(a[1],b[1],c1,sum[1],c2);**  **full\_adder fa2(a[2],b[2],c2,sum[2],c3);**  **full\_adder fa3(a[3],b[3],c3,sum[3],c4);**  **endmodule**  //fbbas (4-Bit binary Adder Subtractor)  **module fbbas\_d(input a\_s,input [3:0]a,b,output [3:0]sum\_diff,output c4\_b4);**  **wire [3:0]Bmod;**  **assign Bmod={4{a\_s}}^b;**  **rca rca0(a,Bmod,a\_s,sum\_diff,c4\_b4);**  **endmodule** | **//fbbas (4-Bit binary Adder Subtractor)-Test Bench**  **module fbbas\_d\_tb();**  **reg a\_s;**  **reg [3:0]a,b;**  **wire [3:0]sum\_diff;**  **wire c4\_b4;**  **fbbas\_d uut(a\_s,a,b,sum\_diff,c4\_b4);**  **initial begin**  **a=4'b0110;**  **b=4'b0100;**  a\_s = 0; //Addition  #20  a\_s = 1; //Subtraction  #20  $stop();  end  endmodule |

**Observation / Output Waveforms:**

**Result:** The simulation waveforms are obtained with various input combination of binary data and Control logic pin (**CTRL /a\_s**) and verified with the truth table data.