ASSIGNMENT REPORT

Architects

Keshav Agarwal 2020102048 Maulesh Gandhi 2020112009

Introduction:

Our approach for this assignment is to make modules for each function and then instantiate them in the ALU and get the required output using 4x1 mux modules. We have created all these modules using a loop. The adder_64 module works by initializing a 1-bit full adder module inside a for loop. The SUBTRACT module works by taking 1's complement of the subtrahend and initializing the 0^{th} carry bit or the input carry bit as 1 to obtain the 2's complement and then adding the first number and the complement similar to the adder_64 module. The XOR and AND modules work in a much simpler way by calling the respective gates 64 times using the loop.

For adder, we are using the normal sequential adder which takes input for carry in from the carry out of the last bit and uses xor gates to generate the sum bit while uses xor gates and or gate to get the carry out bit. We are using this to add 1 bit each of the two operands and then combine the results for the answer.

Variables:

1) Input

a,b,mode (two 64-bit inputs and one 2-bit input to determine function to be carried out)

2) Output

out (one 64 bit output)

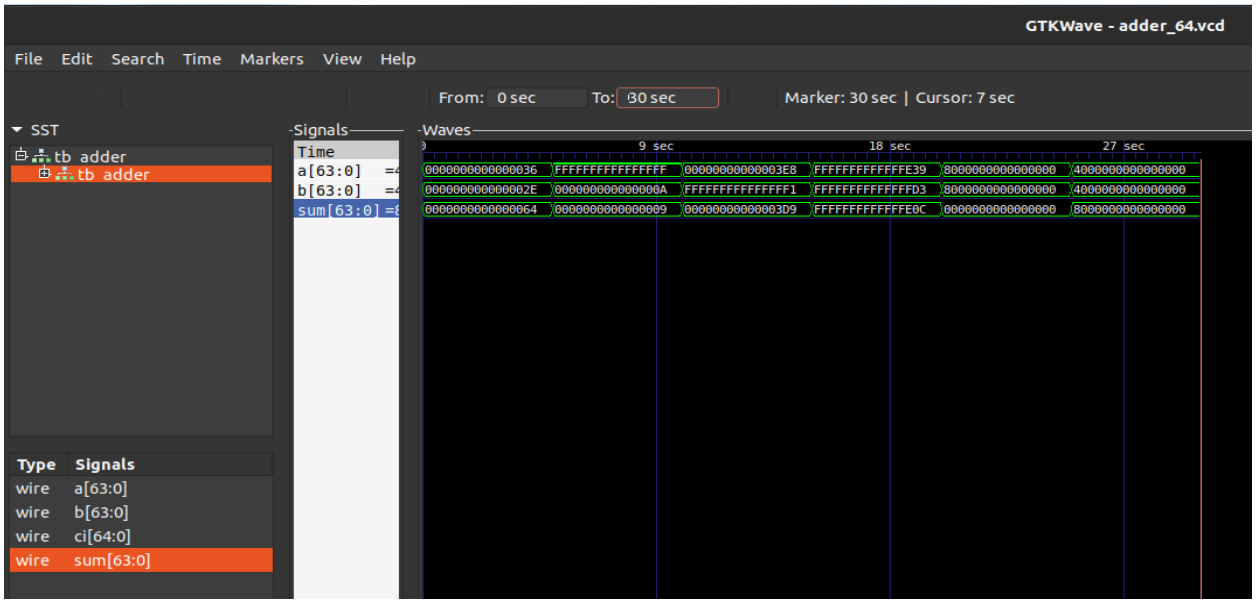
3) Wire

out0,out1,out2,out3 (outputs of all four possible functions which are fed to muxes)

Test Cases:

alu_64 module: -

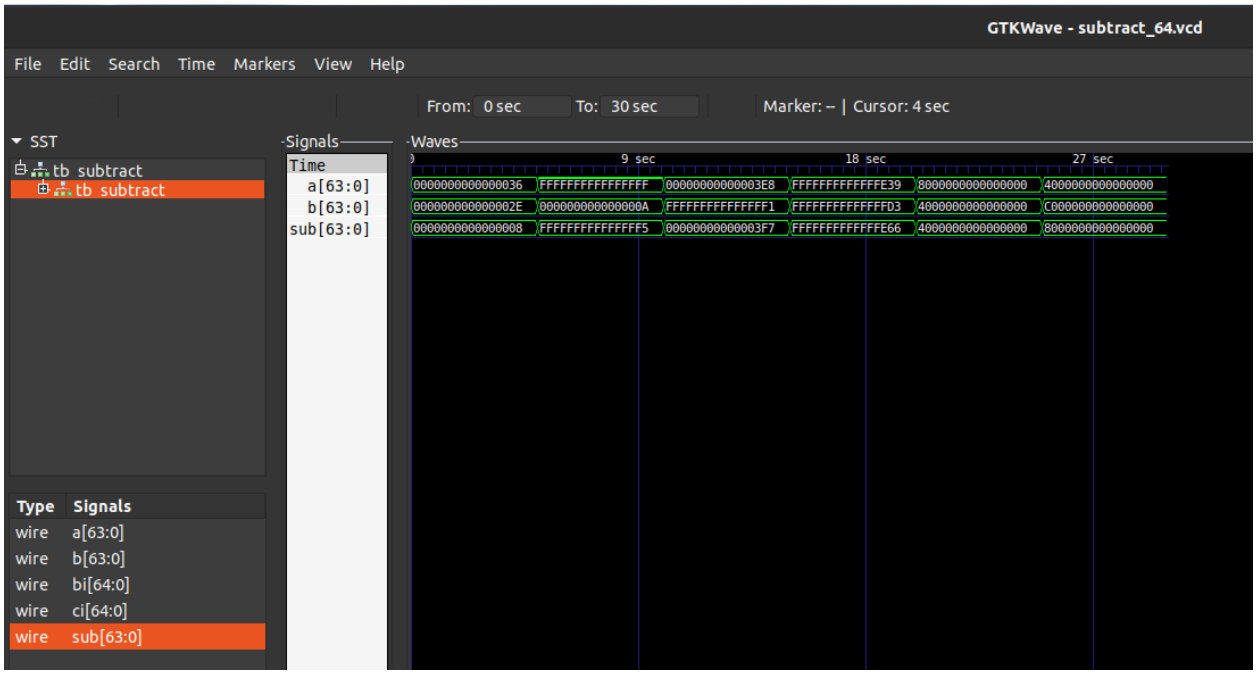
1. adder_64 module (mode =00) :-
    1) Adding two positives {54+46}
    2) Adding a negative and a positive {(-1)+10}
    3) Adding a positive and a negative {1000+(-15)}
    4) Adding two negatives {(-455)+(-45)}
    5) Negative overflow
    6) Positive overflow



(if not clear, please see the photo in github)

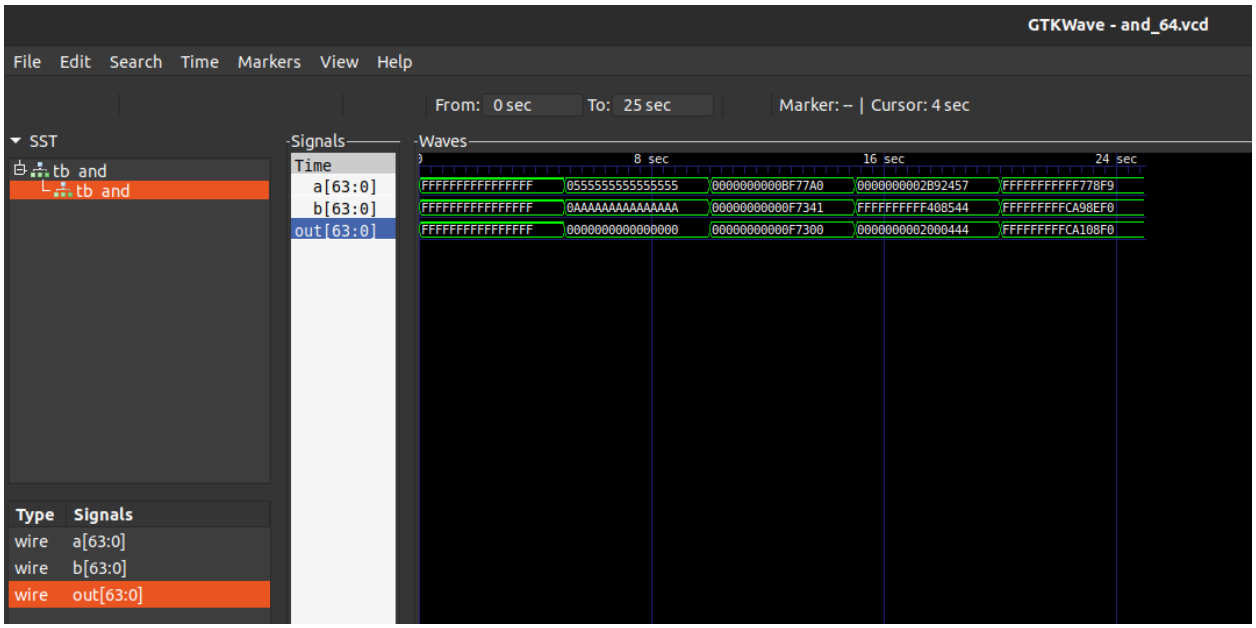
2. subtract_64 module (mode = 01) : -

    1) Subtracting two positives {54-46}
    2) Subtracting a negative and a positive {(-1)-10}
    3) Subtracting a positive and a negative {1000-(-15)}
    4) Subtracting two negatives {(-455)-(-45)}
    5) Negative overflow
    6) Positive overflow



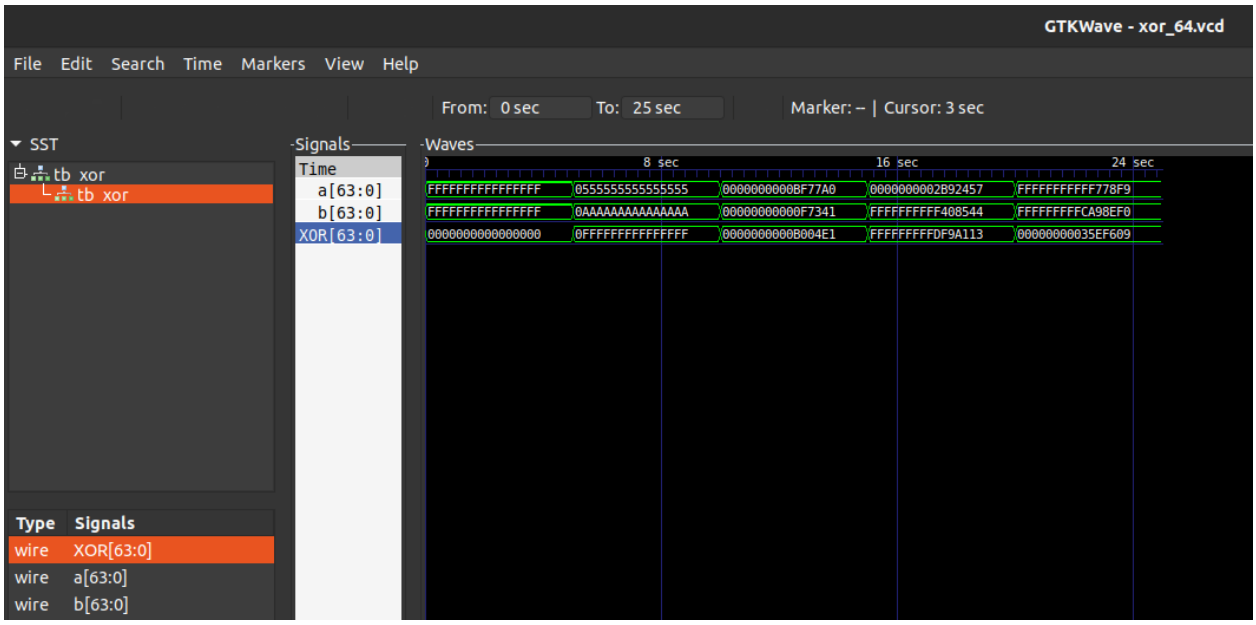
3. and_64 module (mode = 10) : -

1) All bits of inputs are 1
2) Bit shifted input
3) Two positive inputs
4) One positive and one negative input
5) Two negative inputs



4.Xor_64 module (mode = 11): -

1) All bits of inputs are 1
2) Bit shifted input
3) Two positive inputs
4) One positive and one negative input
5) Two negative inputs



Overall waveform: