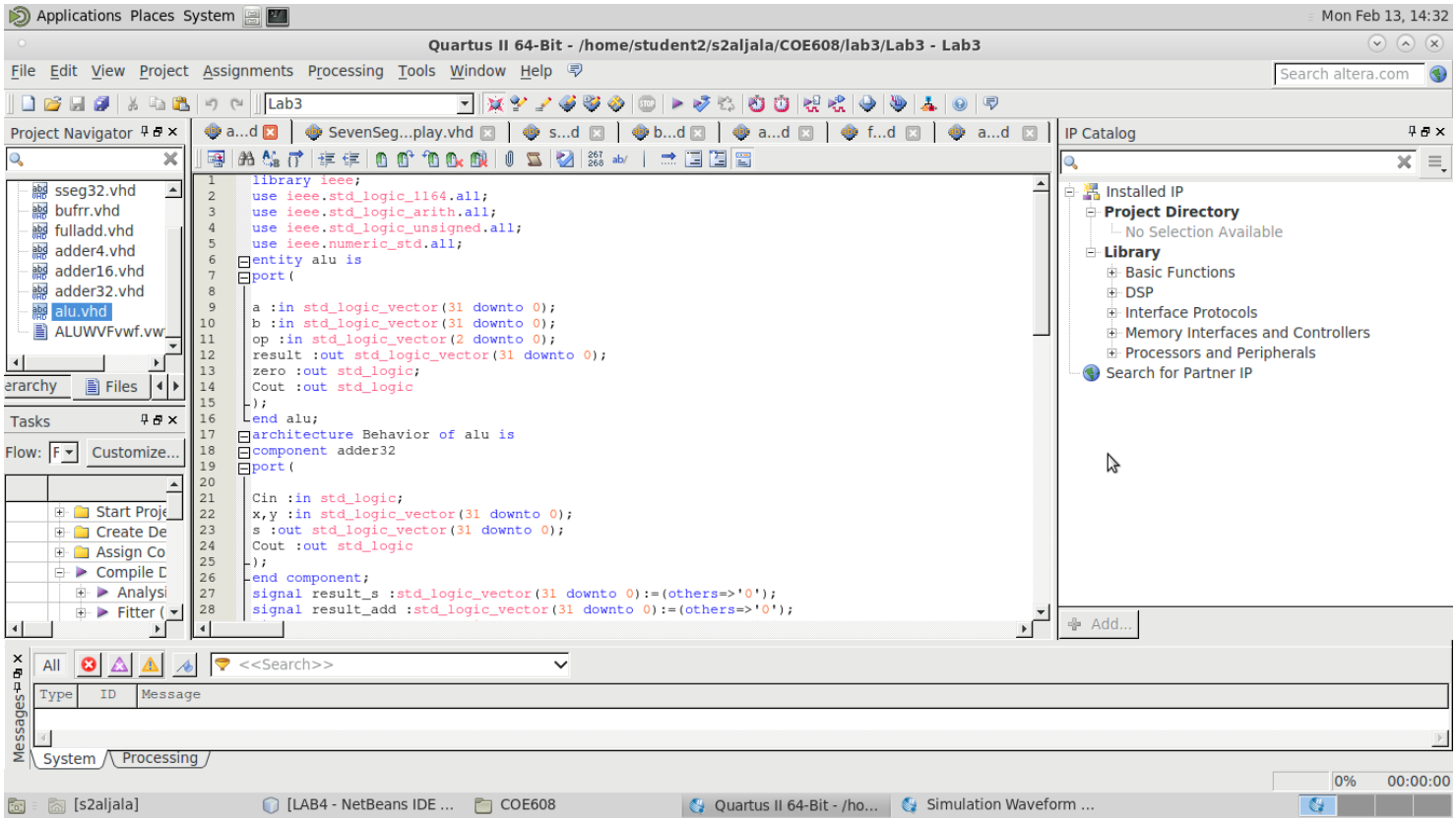
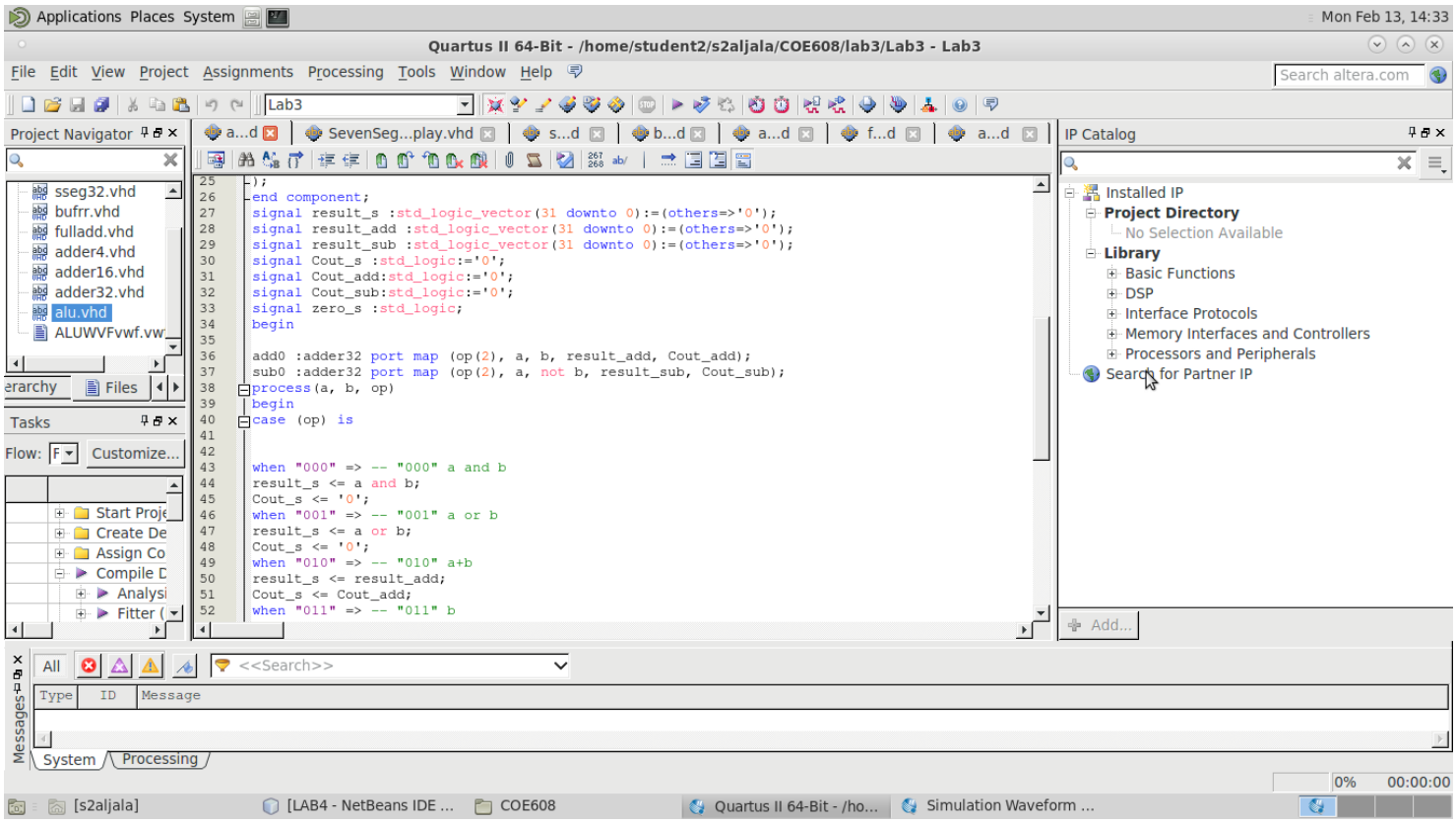
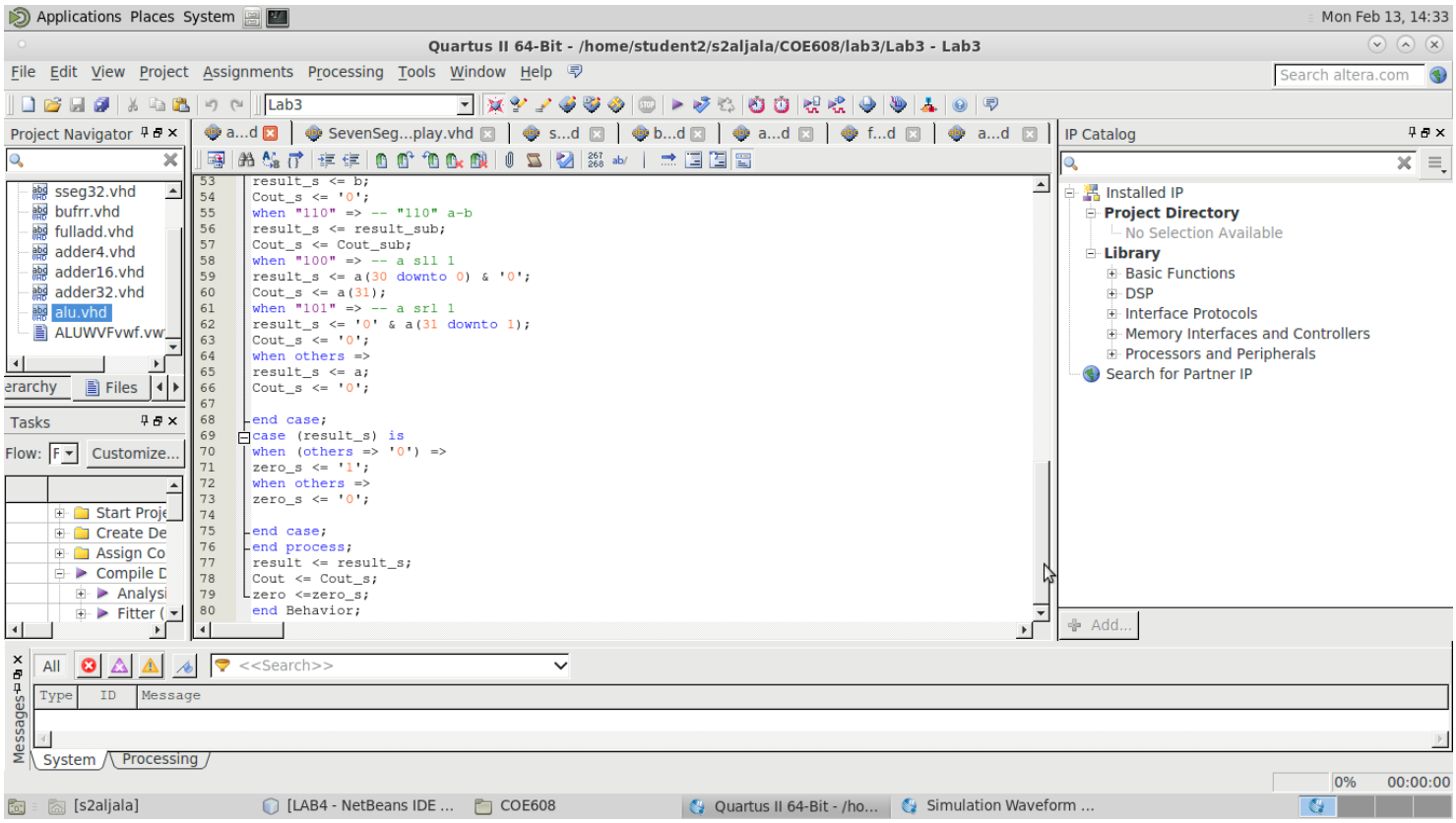
ALU Part 1



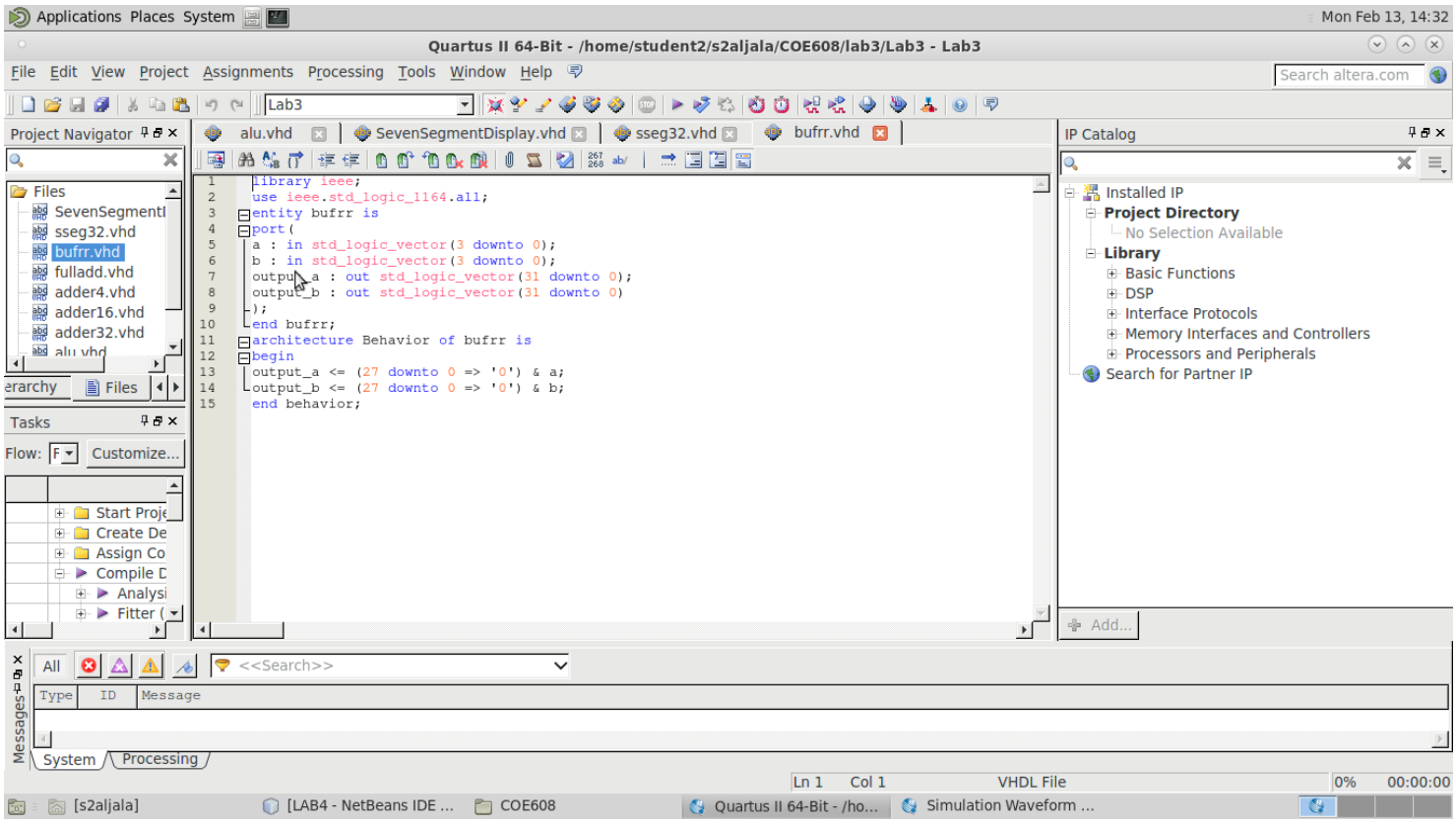
ALU Part 2



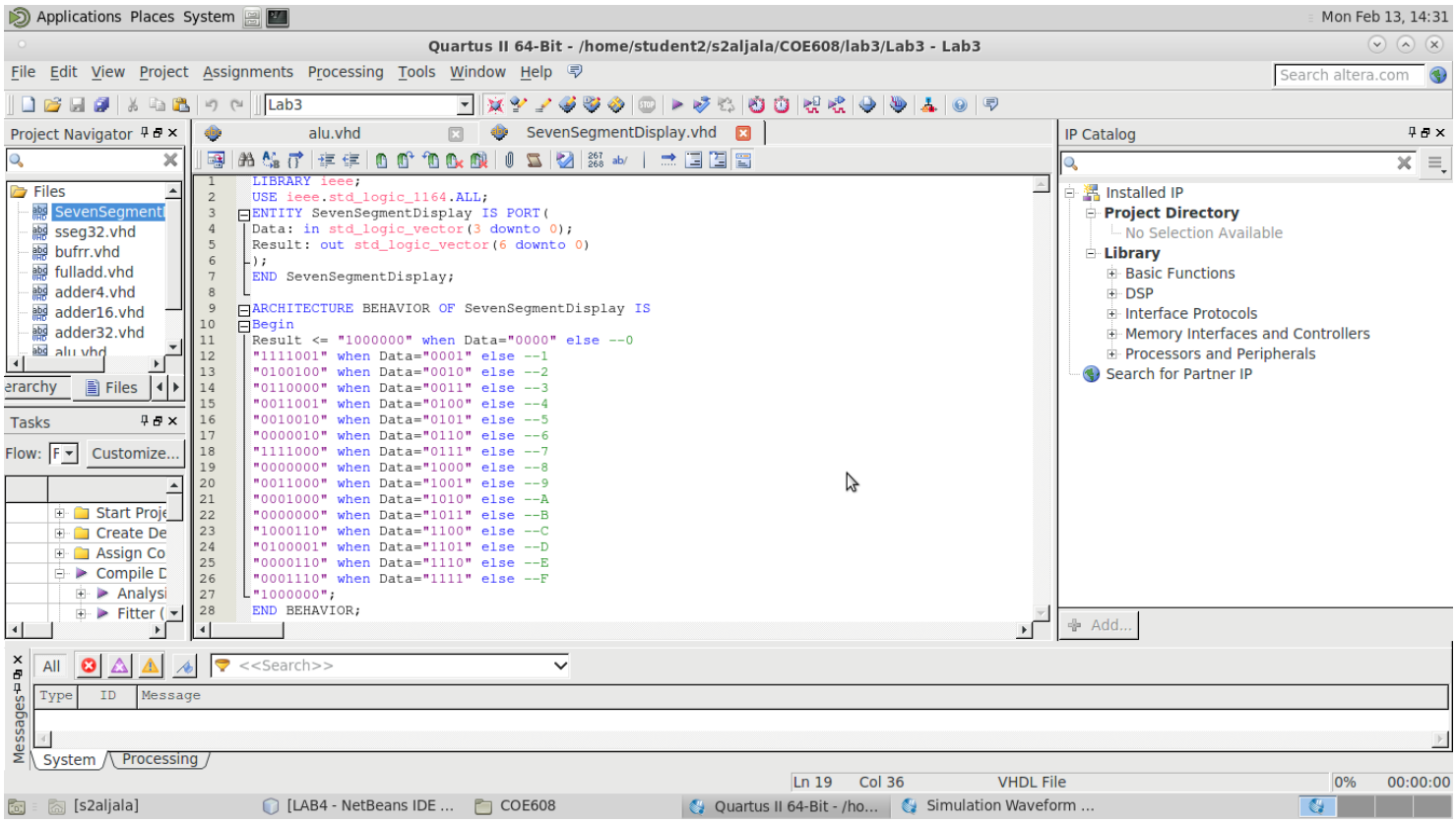
ALU Part 3


Applications Places System
Mon Feb 13, 14:33
Quartus II 64-Bit - /home/student2/s2aljala/COE608/lab3/Lab3 - Lab3
File Edit View Project Assignments Processing Tools Window Help
Search altera.com
Lab3
Project Navigator
sseg32.vhd
bufrr.vhd
fulladd.vhd
adder4.vhd
adder16.vhd
adder32.vhd
alu.vhd
ALUWVFvwf.vw
Hierarchy
Files
Tasks
Flow:
Customize...
Start Proje
Create De
Assign Co
Compile D
Analysi
Fitter (
a...d
SevenSeg...play.vhd
s...d
b...d
a...d
f...d
a...d
result_s <= b;
Cout_s <= '0';
when "110" => -- "110" a-b
result_s <= result_sub;
Cout_s <= Cout_sub;
when "100" => -- a sll 1
result_s <= a(30 downto 0) & '0';
Cout_s <= a(31);
when "101" => -- a srl 1
result_s <= '0' & a(31 downto 1);
Cout_s <= '0';
when others =>
result_s <= a;
Cout_s <= '0';
end case;
case (result_s) is
when (others => '0') =>
zero_s <= '1';
when others =>
zero_s <= '0';
end case;
end process;
result <= result_s;
Cout <= Cout_s;
zero <=zero_s;
end Behavior;
IP Catalog
Installed IP
Project Directory
No Selection Available
Library
Basic Functions
DSP
Interface Protocols
Memory Interfaces and Controllers
Processors and Peripherals
Search for Partner IP
Add...
Messages
All
<<Search>>
Type
ID
Message
System
Processing
0%
00:00:00
[s2aljala]
[LAB4 - NetBeans IDE ...
COE608
Quartus II 64-Bit - /ho...
Simulation Waveform ...

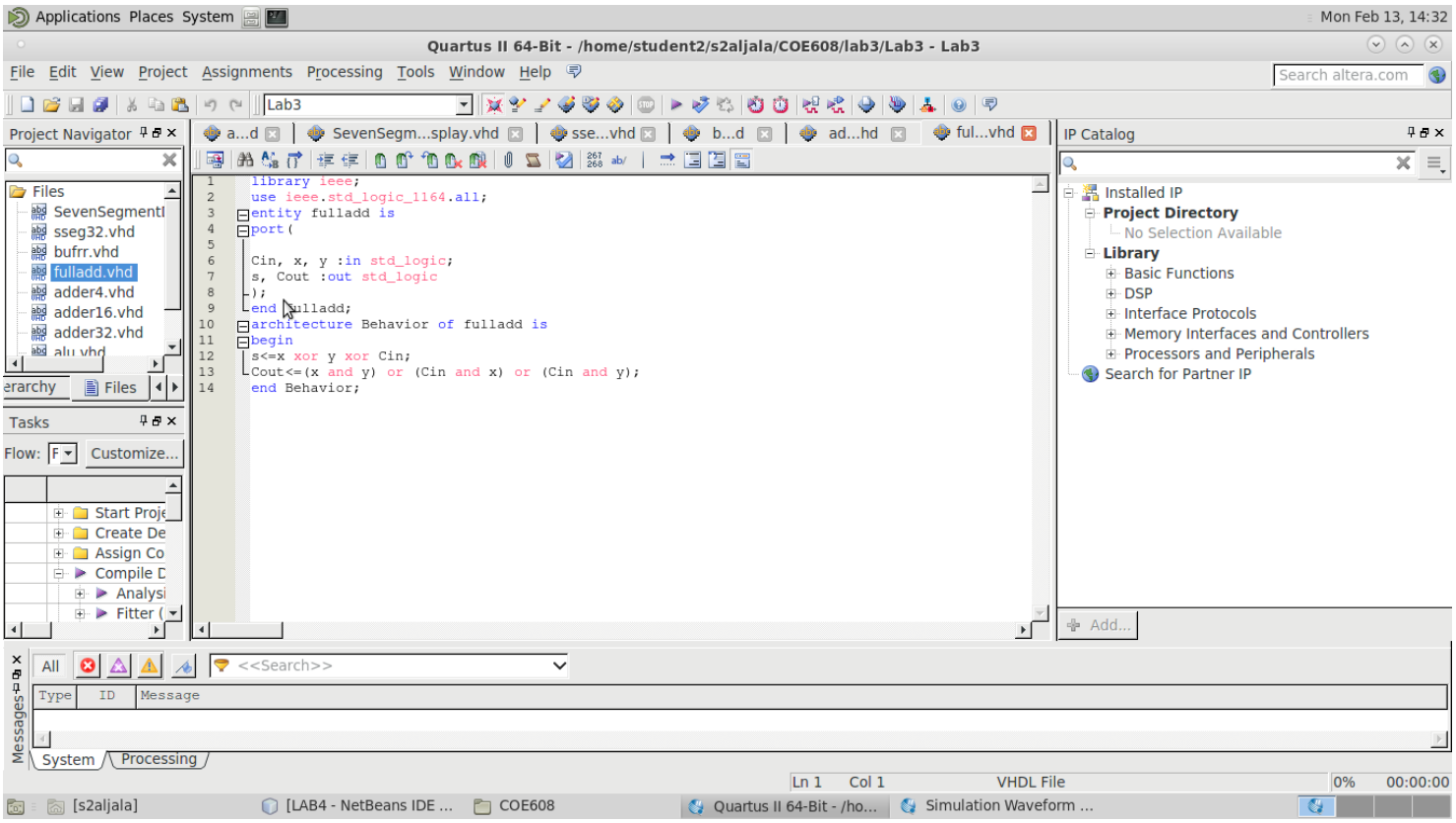
Buffer



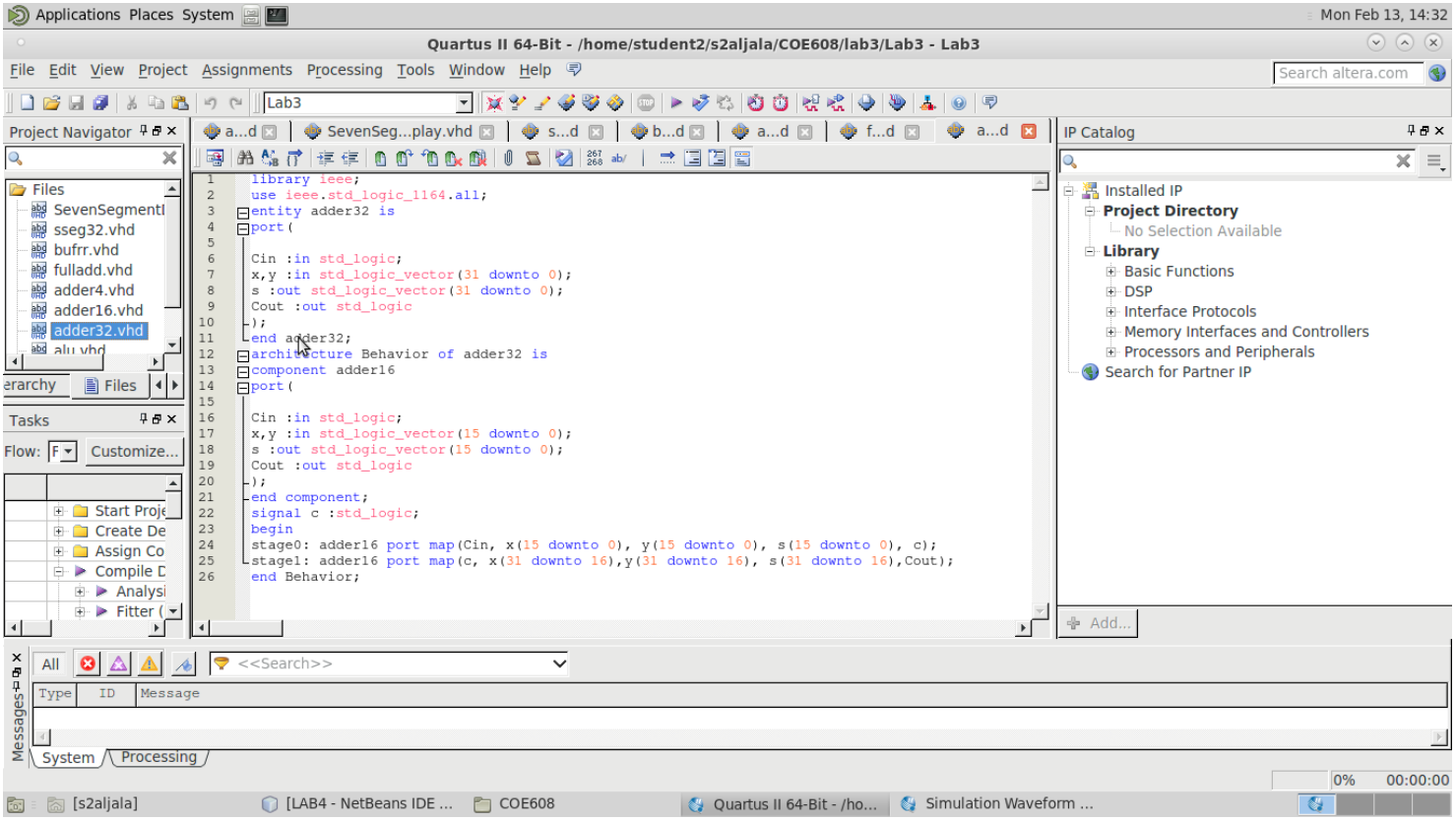
Seven Segment



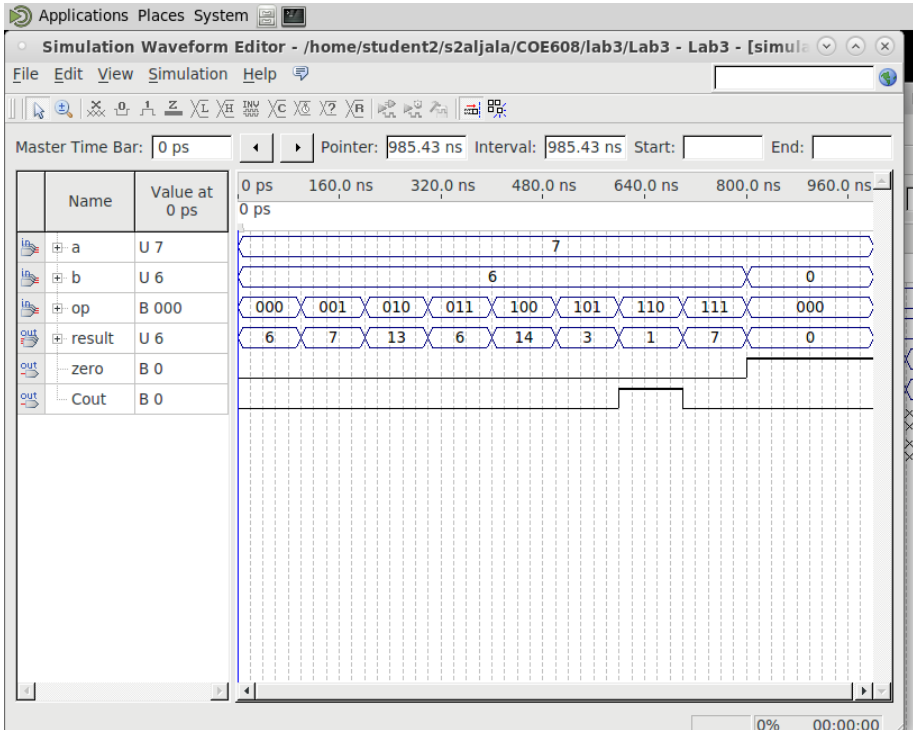
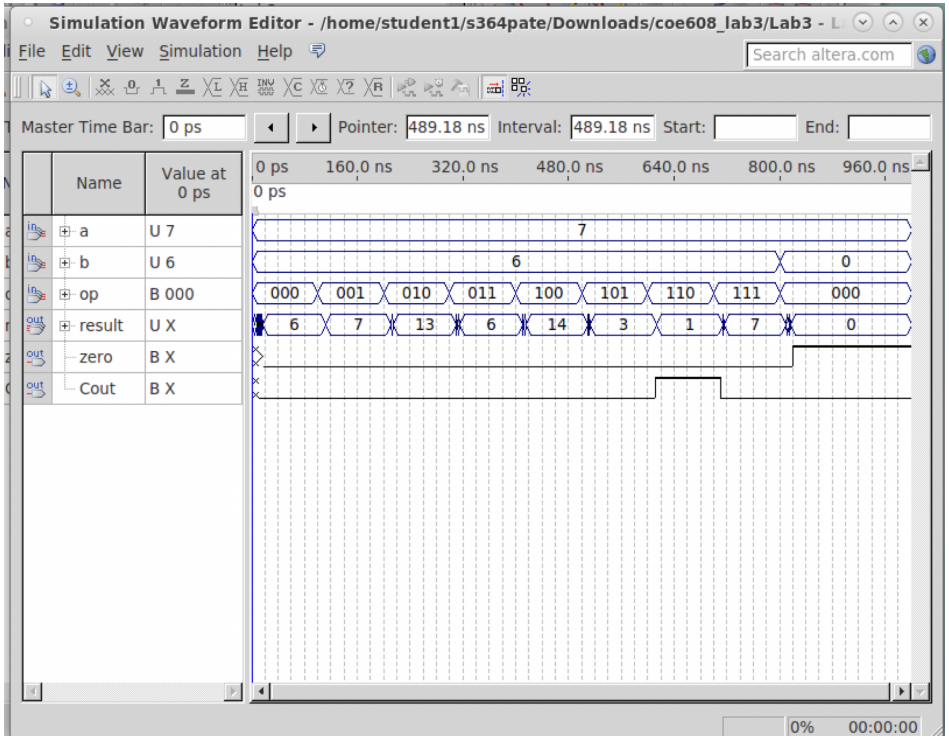
fulladd



adder32



32-bit ALU Waveforms:

Functional Simulation	Time Simulation
	

***Please Scroll Down to see the explanation on the two waveforms.**

Description of Input and Output Functionalities

The graphs above are a representation of the 32-bit ALU Design which has inputs a,b, op and results along with outputs zero, Cout. For both waveforms input values a and b are decimal while others are in binary. Input ‘a’ is set as 7 and input ‘b’ is set as 6. The op values are default which represent the ALU arithmetics and the "result" is the output of the 32-bit ALU (Arithmetic Logic Unit), that stores the result of the operation performed on the inputs "a" and "b" by the ops. "zero" is a single-bit output signal that represents whether the result of the operation performed by the ALU is equal to zero or not. If the result is equal to zero, the zero output is set to '1', otherwise it is set to '0'. The carry out bit is used to indicate if there was an overflow or carry-out during the operation. If there is an overflow or carry-out, Cout is set to '1', otherwise it is set to '0'.

1. For op = "000", the operation performed is "a and b" and the result is 6.
2. For op = "001", the operation performed is "a or b" and the result is 7.
3. For op = "010", the operation performed is "a + b" and the result is 13.
4. For op = "011", the operation performed is "b" and the result is 6.
5. For op = "100", the operation performed is "a SLL 1" (left shift by 1 bit) and the result is 14.
6. For op = "110", the operation performed is "a - b" and the result is 1. Cout is ‘1’ indicating that there was a borrow during the subtraction operation.
7. For op = "111", the operation performed is "a" and the result is 7.
8. For op = "000", the result is 0 because input b has changed to ‘0’ while input a=7. The zero output is ‘1’ to indicate that the result of ALU equals to 0.

Difference Between Functional Simulations Time Simulation

Both waveforms are very much identical to each other except the only difference is that there is a delay shown in the time waveform. This is because the functional simulation verifies the correct functioning of the circuit, and showcases an ideal output without considering the timing delay of individual gates. On the other hand, time simulation waveform showcases more of a realistic output as it considers the timing delay of individual gates.The delay is calculated based on the type of gate, its size, the load on its output, and the voltage and temperature conditions. This delay is important to consider as it affects the final result and can lead to errors in the circuit.

Explanation of the Delays

The waveform in a time simulation reveals delays at the start and end due to the propagation time of input signals and the setup time of the first flip-flop. There may also be minor delays between specific operations, such as "010" and "011" and "100" and "101," due to the logic gates employed in the design. The subtraction operation (op="110") has the worst-case delay among the various operations since it needs the CPU to invert the integer, making it slower than the addition operation (op="010"). In comparison, the operations "a and b" (op="000") and "a or b" (op="001") have relatively lesser delays, as they require simple logic operations. As a result, these timing delays must be considered since they can affect the accuracy of the circuit's final result.