**Lab Task\_01**

**ALU implementation using your own library components**

**ENTITIES:**

**1:OR gate**

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**entity** Orgate **is**

**port(**

or\_in1**,** or\_in2 **:** **in** STD\_LOGIC**;**

or\_out **:** **out** STD\_LOGIC

**);**

**end** Orgate**;**

**architecture** bhv **of** Orgate **is**

**begin**

or\_out **<=** or\_in1 **OR** or\_in2**;**

**end** bhv**;**

**1:AND gate**

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**entity** Andgate **is**

**port(**

and\_in1**,** and\_in2 **:** **in** STD\_LOGIC**;**

and\_out **:** **out** STD\_LOGIC

**);**

**end** Andgate**;**

**architecture** bhv **of** Andgate **is**

**begin**

and\_out **<=** and\_in1 **AND** and\_in2**;**

**end** bhv**;**

**2:NAND gate**

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**entity** Nandgate **is**

**port(**

nand\_in1**,** nand\_in2 **:** **in** STD\_LOGIC**;**

nand\_out **:** **out** STD\_LOGIC

**);**

**end** Nandgate**;**

**architecture** bhv **of** Nandgate **is**

**begin**

nand\_out **<=** **NOT** **(**nand\_in1 **AND** nand\_in2**);**

**end** bhv**;**

**3:Full Adder**

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**entity** Fulladder **is**

**port(**

adder\_inA**,** adder\_inB**,** adder\_inC **:** **in** STD\_LOGIC**;**

adder\_outS**,** adder\_outC **:** **out** STD\_LOGIC

**);**

**end** Fulladder**;**

**architecture** bhv **of** Fulladder **is**

**begin**

adder\_outS **<=** **(**adder\_inA **XOR** adder\_inB**)** **XOR** adder\_inC**;**

adder\_outC **<=** **(**adder\_inA **AND** adder\_inB**)** **OR** **(**adder\_inC **AND** **(**adder\_inA **XOR** adder\_inB**));**

**end** bhv**;**

**4:MUX**

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**entity** Mux **is**

**port** **(** mux\_inI0**,** mux\_inI1**,** mux\_inI2**,** mux\_inI3 **:** **in** STD\_LOGIC**;**

mux\_Sel **:** **in** STD\_LOGIC\_VECTOR**(**1 **downto** 0**);**

mux\_outY **:** **out** STD\_LOGIC**);**

**end** Mux**;**

**architecture** bhv **of** Mux **is**

**begin**

**process** **(**mux\_Sel**,** mux\_inI0**,** mux\_inI1**,** mux\_inI2**,** mux\_inI3**)**

**begin**

**case** mux\_Sel **is**

**when** "00" **=>** mux\_outY **<=** mux\_inI0**;**

**when** "01" **=>** mux\_outY **<=** mux\_inI1**;**

**when** "10" **=>** mux\_outY **<=** mux\_inI2**;**

**when** "11" **=>** mux\_outY **<=** mux\_inI3**;**

**when** **others** **=>** mux\_outY **<=** '0'**;**

**end** **case;**

**end** **process;**

**end** bhv**;**

**PACKAGE:**

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**package** alu\_components **is**

-- OR Gate

**component** Orgate

**port** **(** or\_in1**,** or\_in2 **:** **in** STD\_LOGIC**;**

or\_out **:** **out** STD\_LOGIC**);**

**end** **component;**

-- AND Gate

**component** Andgate

**port** **(** and\_in1**,** and\_in2 **:** **in** STD\_LOGIC**;**

and\_out **:** **out** STD\_LOGIC**);**

**end** **component;**

-- NAND Gate

**component** Nandgate

**port** **(** nand\_in1**,** nand\_in2 **:** **in** STD\_LOGIC**;**

nand\_out **:** **out** STD\_LOGIC**);**

**end** **component;**

-- Full Adder

**component** Fulladder

**port** **(** adder\_inA**,** adder\_inB**,** adder\_inC **:** **in** STD\_LOGIC**;**

adder\_outS**,** adder\_outC **:** **out** STD\_LOGIC**);**

**end** **component;**

-- 4-to-1 Multiplexer

**component** Mux

**port** **(** mux\_inI0**,** mux\_inI1**,** mux\_inI2**,** mux\_inI3 **:** **in** STD\_LOGIC**;**

mux\_Sel **:** **in** STD\_LOGIC\_VECTOR**(**1 **downto** 0**);**

mux\_outY **:** **out** STD\_LOGIC**);**

**end** **component;**

**end** alu\_components**;**

**TOP-ENTITY(Wrapper File)**

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**use** work**.**alu\_components**.ALL;** -- Import our custom components

**entity** ALU **is**

**port** **(**

a**,** b**,** Cin **:** **in** STD\_LOGIC**;**

Sel **:** **in** STD\_LOGIC\_VECTOR**(**1 **downto** 0**);**

Result**,** Cout **:** **out** STD\_LOGIC

**);**

**end** ALU**;**

**architecture** struct **of** ALU **is**

-- Internal signals to connect components

**signal** and\_out**,** or\_out**,** nand\_out**,** sum\_out**,** carry\_out**:** STD\_LOGIC**;**

**begin**

-- Instantiate AND Gate

U1**:** Andgate **port** **map(**and\_in1 **=>** a**,** and\_in2 **=>** b**,** and\_out **=>** and\_out**);**

-- Instantiate OR Gate

U2**:** Orgate **port** **map(**or\_in1 **=>** a**,** or\_in2 **=>** b**,** or\_out **=>** or\_out**);**

-- Instantiate NAND Gate

U3**:** Nandgate **port** **map(**nand\_in1 **=>** a**,** nand\_in2 **=>** b**,** nand\_out **=>** nand\_out**);**

-- Instantiate Full Adder

U4**:** Fulladder **port** **map(**adder\_inA **=>** a**,** adder\_inB **=>** b**,** adder\_inC **=>** Cin**,** adder\_outS **=>** sum\_out**,** adder\_outC **=>** carry\_out**);**

-- Instantiate 4-to-1 MUX

U5**:** Mux **port** **map(**

mux\_inI0 **=>** and\_out**,**

mux\_inI1 **=>** or\_out**,**

mux\_inI2 **=>** nand\_out**,**

mux\_inI3 **=>** sum\_out**,**

mux\_Sel **=>** Sel**,**

mux\_outY **=>** Result

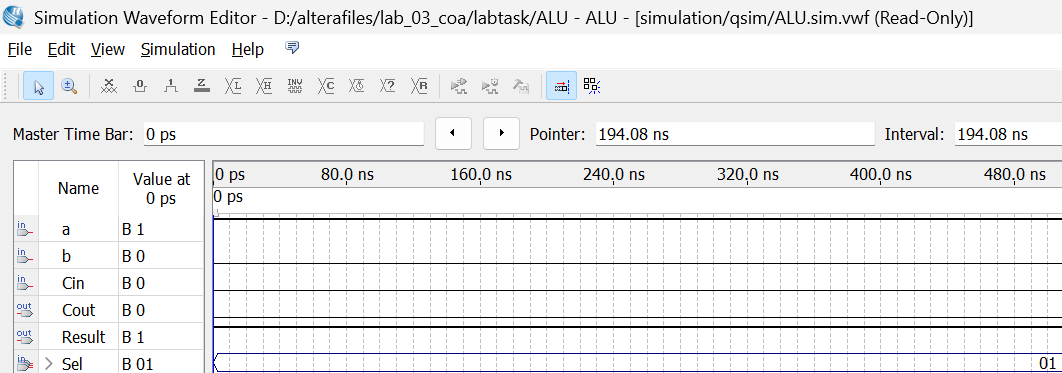
**);**

-- Assign Carry Out

Cout **<=** carry\_out**;**

**end** struct**;**

**Simulation:**

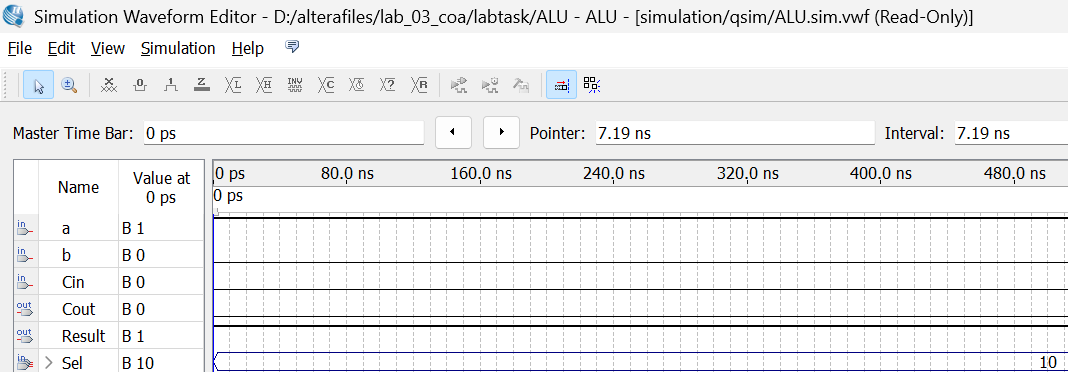
****

**Simulation 1:** Selection line is 01 which reflects the OR gate in MUX logic ,So input a=1 and b=0 ==>Result=1

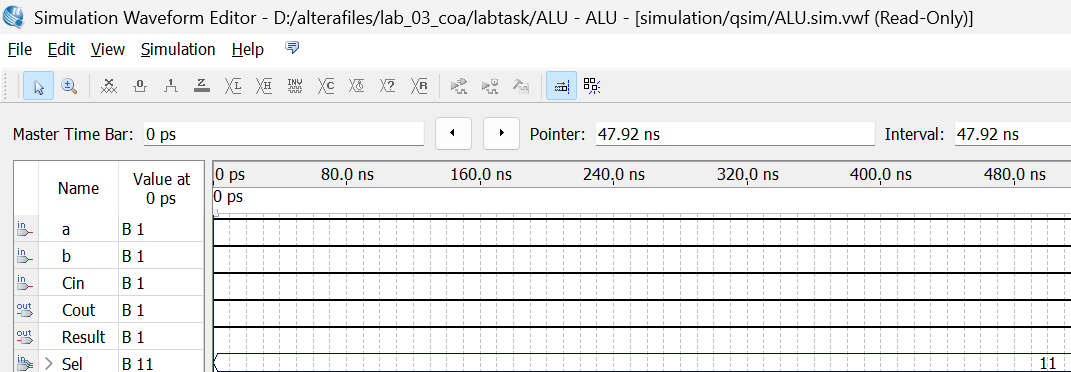
A screenshot of a computer

AI-generated content may be incorrect.

**Simulation 2:** Selection line is 00 which reflects the AND gate in MUX logic ,So input a=1 and b=0 ==>Result=0



**Simulation 3:** Selection line is 10 which reflects the NAND gate in MUX logic ,So input a=1 and b=0 ==>Result=1



**Simulation4:** Selection line is 01 which reflects the ADDER in MUX logic ,So input a=1,b=1 and Cin=1 ==>Result=1,Cout=1