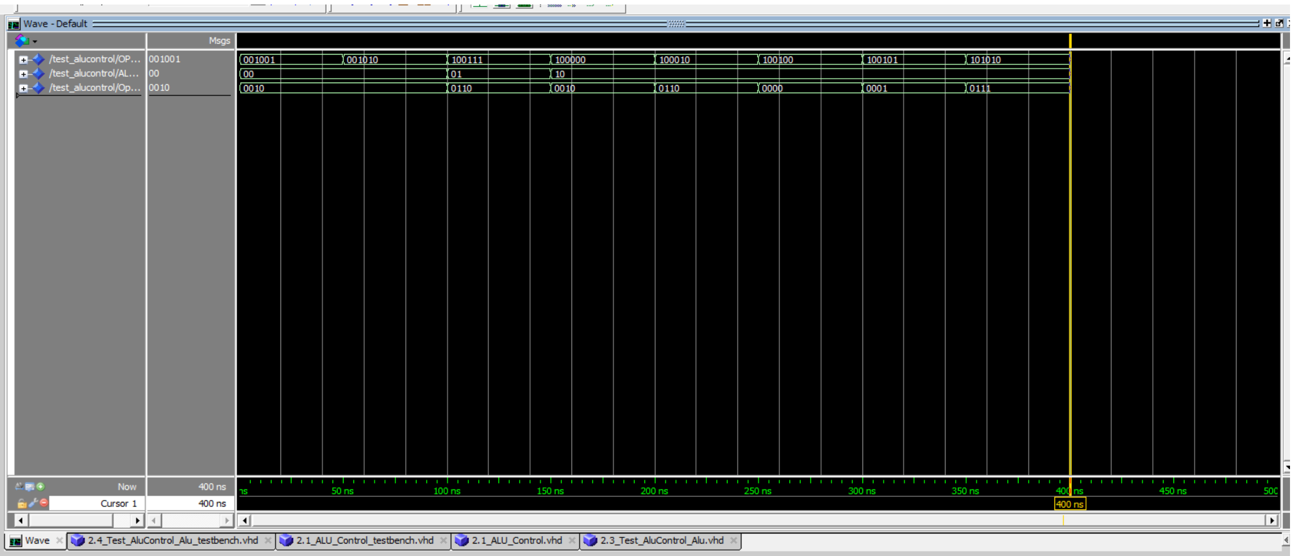
# 2: Unit Control ALU

## 2.1 ALU_Control.vhd

```
--MIPS Part_2
--UNIT control ALU
--30/05/2020, Konstantinos Gkousaris, 711171073, UniWA--

LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY ALU_Control IS PORT(
      OP_5to0 : IN STD_LOGIC_VECTOR(5 DOWNTO 0);
      ALU_op  : IN STD_LOGIC_VECTOR(1 DOWNTO 0);
      Operation : OUT STD_LOGIC_VECTOR(3 DOWNTO 0));
END ALU_Control;

ARCHITECTURE ALUControl_1 OF ALU_Control IS



BEGIN
      --code base on github, credits to Dugagjin Lashi
      --Understand this block and works perfectly for more than the
      --operation which we use.
      Operation(3) <= '0';
      Operation(2) <= ALU_op(0) or (ALU_op(1) and OP_5to0(1));
      Operation(1) <= not ALU_op(1) or not OP_5to0(2);
      Operation(0) <= (OP_5to0(3) or OP_5to0(0)) and ALU_op(1);
END;
```

## 2.2 ALU_Control_testbench.vhd

```
--test bench
--runs for 400 ns
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY test_ALUControl IS
END test_ALUControl;


ARCHITECTURE behavioral OF test_AluControl IS

COMPONENT Alu_Control PORT(
      OP_5to0 : IN STD_LOGIC_VECTOR(5 DOWNTO 0);
      ALU_op  : IN STD_LOGIC_VECTOR(1 DOWNTO 0);
      Operation : OUT STD_LOGIC_VECTOR(3 DOWNTO 0));
END COMPONENT;

      SIGNAL OP_5to01   : STD_LOGIC_VECTOR(5 DOWNTO 0);
      SIGNAL ALU_op1    : STD_LOGIC_VECTOR(1 DOWNTO 0);
      SIGNAL Operation1: STD_LOGIC_VECTOR(3 DOWNTO 0);

BEGIN

      ALUCONTROL1 : Alu_control PORT MAP (
            OP_5to0 => OP_5to01,
            ALU_op => ALU_op1,
            Operation => Operation1);

      alu_op_process : PROCESS
      BEGIN
            ALU_op1 <= "00"; wait for 50 ns; --00 (1)
            ALU_op1 <= "00"; wait for 50 ns; --00 (2)
            ALU_op1 <= "01"; wait for 50 ns; --01 (3)
            ALU_op1 <= "10"; wait for 50 ns; --10 (4)
            ALU_op1 <= "10"; wait for 50 ns; --10 (5)
            ALU_op1 <= "10"; wait for 50 ns; --10 (6)
            ALU_op1 <= "10"; wait for 50 ns; --10 (7)
            ALU_op1 <= "10"; wait for 50 ns; --10 (8)
      END PROCESS;

      opfunction :  PROCESS
      BEGIN
            OP_5to01 <= "001001"; wait for 50 ns; --001001 (1)
            OP_5to01 <= "001010"; wait for 50 ns; --001010 (2)
            OP_5to01 <= "100111"; wait for 50 ns; --100111 (3)
            OP_5to01 <= "100000"; wait for 50 ns; --001000 (4)
            OP_5to01 <= "100010"; wait for 50 ns; --100010 (5)
            OP_5to01 <= "100100"; wait for 50 ns; --100100 (6)
            OP_5to01 <= "100101"; wait for 50 ns; --100101 (7)
            OP_5to01 <= "101010"; wait for 50 ns; --101010 (8)
      END PROCESS;
END;
```

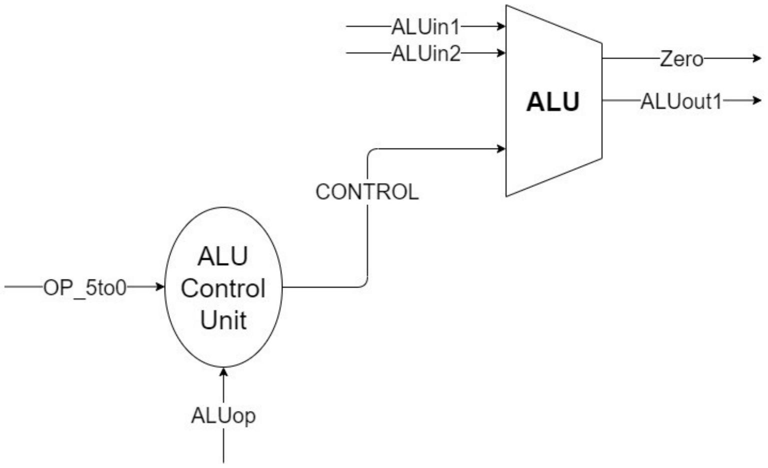


**Πίνακας Λειτουργίας**

| ALUop | OP_5to0 | Operation(Λειτουργία) |
|---|---|---|
| 00 | 001001 | 0010 |
| 00 | 001010 | 0010 |
| 01 | 100111 | 0110 |
| 10 | 100000 | 0010 |
| 10 | 100010 | 0110 |
| 10 | 100100 | 0000 |
| 10 | 100101 | 0001 |
| 10 | 101010 | 0111 |

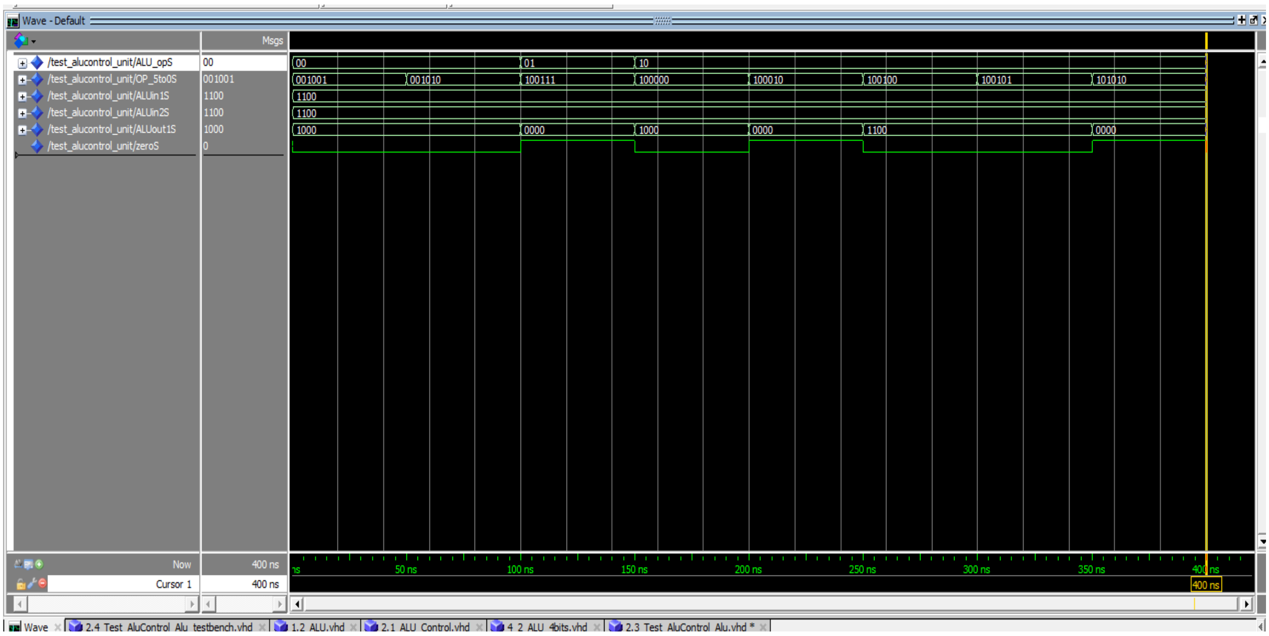
## 2.3 Test_AluControl_Alu.vhd

```
--MIPS Part_2
--TEST ALU_CONTROL_ALU
--30/05/2020, Konstantinos Gkousaris, 711171073, UniWA--

LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY TEST_ALUCONTROL_ALU IS PORT (
      OP_5to0 : IN STD_LOGIC_VECTOR(5 DOWNTO 0);
      ALU_op  : IN STD_LOGIC_VECTOR(1 DOWNTO 0);
      ALUin1  : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
      ALUin2      : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
      ALUout1 : OUT STD_LOGIC_VECTOR(3 DOWNTO 0);
      zero    : OUT STD_LOGIC);
END TEST_ALUCONTROL_ALU;

ARCHITECTURE stuctural OF TEST_ALUCONTROL_ALU IS

      COMPONENT alu4 PORT (
            aluin1  : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
            aluin2  : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
            aluctrl : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
            aluout  : OUT STD_LOGIC_VECTOR(3 DOWNTO 0);
            zero    : OUT STD_LOGIC);
      END COMPONENT;

      COMPONENT ALU_Control PORT(
            OP_5to0   : IN STD_LOGIC_VECTOR(5 DOWNTO 0);
            ALU_op    : IN STD_LOGIC_VECTOR(1 DOWNTO 0);
            Operation : OUT STD_LOGIC_VECTOR(3 DOWNTO 0));
      END COMPONENT;

      SIGNAL CONTROL : STD_LOGIC_VECTOR(3 DOWNTO 0); --the only eternal signal

BEGIN

      ALU_CONTROL_TEST1 : Alu_Control PORT MAP (OP_5to0,ALU_op,CONTROL);
      ALU4_TEST1 :  alu4 PORT MAP (ALUin1,ALUin2,CONTROL,ALUout1,zero);

END;
```



## 2.4 Test_AluControl_Alu_testbench.vhd

```
--test bench
--runs for 400 ns
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY test_ALUControl_Unit IS
END test_ALUControl_Unit;

ARCHITECTURE behavioral OF test_AluControl_Unit IS

COMPONENT TEST_ALUCONTROL_ALU PORT (
      OP_5to0 : IN STD_LOGIC_VECTOR(5 DOWNTO 0);
      ALU_op  : IN STD_LOGIC_VECTOR(1 DOWNTO 0);
      ALUin1  : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
      ALUin2      : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
      ALUout1 : OUT STD_LOGIC_VECTOR(3 DOWNTO 0);
      zero    : OUT STD_LOGIC);
END COMPONENT;

      SIGNAL ALUin1S, ALUin2S, ALUout1S : STD_LOGIC_VECTOR(3 DOWNTO 0);
      SIGNAL zeroS     : STD_LOGIC;
      SIGNAL ALU_opS    : STD_LOGIC_VECTOR(1 DOWNTO 0);
      SIGNAL OP_5to0S   : STD_LOGIC_VECTOR(5 DOWNTO 0);
BEGIN

      TESTALUCONTROLALU1 : TEST_ALUCONTROL_ALU PORT MAP (
            OP_5to0 => OP_5to0S,
            ALU_op => ALU_opS,
            ALUin1 => ALUin1S,
            ALUin2 => ALUin2S,
            ALUout1 => ALUout1S,
            zero => zeroS);

      alu_op_process : PROCESS
      BEGIN
            ALU_opS <= "00"; wait for 50 ns; --00 (1)
            ALU_opS <= "00"; wait for 50 ns; --00 (2)
            ALU_opS <= "01"; wait for 50 ns; --01 (3)
            ALU_opS <= "10"; wait for 50 ns; --10 (4)
            ALU_opS <= "10"; wait for 50 ns; --10 (5)
            ALU_opS <= "10"; wait for 50 ns; --10 (6)
            ALU_opS <= "10"; wait for 50 ns; --10 (7)
            ALU_opS <= "10"; wait for 50 ns; --10 (8)
      END PROCESS;

      opfunction :  PROCESS
      BEGIN
            OP_5to0S <= "001001"; wait for 50 ns; --001001 (1)
            OP_5to0S <= "001010"; wait for 50 ns; --001010 (2)
            OP_5to0S <= "100111"; wait for 50 ns; --100111 (3)
            OP_5to0S <= "100000"; wait for 50 ns; --001000 (4)
            OP_5to0S <= "100010"; wait for 50 ns; --100010 (5)
            OP_5to0S <= "100100"; wait for 50 ns; --100100 (6)
            OP_5to0S <= "100101"; wait for 50 ns; --100101 (7)
            OP_5to0S <= "101010"; wait for 50 ns; --101010 (8)
      END PROCESS;

      aluin1_process : PROCESS
      BEGIN
            ALUin1S <= "1100"; wait for 50 ns; --1100 (1)
            ALUin1S <= "1100"; wait for 50 ns; --1100 (2)
            ALUin1S <= "1100"; wait for 50 ns; --1100 (3)
            ALUin1S <= "1100"; wait for 50 ns; --1100 (4)
            ALUin1S <= "1100"; wait for 50 ns; --1100 (5)
            ALUin1S <= "1100"; wait for 50 ns; --1100 (6)
            ALUin1S <= "1100"; wait for 50 ns; --1100 (7)
            ALUin1S <= "1100"; wait for 50 ns; --1100 (8)
      END PROCESS;

      aluin2_process : PROCESS
      BEGIN
            ALUin2S <= "1100"; wait for 50 ns; --1100 (1)
            ALUin2S <= "1100"; wait for 50 ns; --1100 (2)
            ALUin2S <= "1100"; wait for 50 ns; --1100 (3)
            ALUin2S <= "1100"; wait for 50 ns; --1100 (4)
            ALUin2S <= "1100"; wait for 50 ns; --1100 (5)
            ALUin2S <= "1100"; wait for 50 ns; --1100 (6)
            ALUin2S <= "1100"; wait for 50 ns; --1100 (7)
            ALUin2S <= "1100"; wait for 50 ns; --1100 (8)
      END PROCESS;

END;
```



Wave - Default
Msgs
/test_alucontrol_unit/ALU_opS 00
/test_alucontrol_unit/OP_5to0S 001001
/test_alucontrol_unit/ALUin1S 1100
/test_alucontrol_unit/ALUin2S 1100
/test_alucontrol_unit/ALUout1S 1000
/test_alucontrol_unit/zeroS 0
00
01
10
001001
001010
100111
100000
100010
100100
100101
101010
1100
1100
1000
0000
1000
0000
1100
0000
Now 400 ns
Cursor 1 400 ns
ns
50 ns
100 ns
150 ns
200 ns
250 ns
300 ns
350 ns
400 ns
400 ns
Wave
2.4 Test AluControl Alu testbench.vhd
1.2 ALU.vhd
2.1 ALU Control.vhd
4 2 ALU 4bits.vhd
2.3 Test AluControl Alu.vhd *

## Πίνακας Λειτουργίας

| ALUop | OP_5to0 | ALUin1 | ALUin2 | ALUout1 | Zero |
|---|---|---|---|---|---|
| 00 | 001001 | 1100 | 1100 | 1000 | 0 |
| 01 | 001010 | 1100 | 1100 | 1000 | 0 |
| 10 | 100111 | 1100 | 1100 | 0000 | 1 |
| 10 | 100000 | 1100 | 1100 | 1000 | 0 |
| 10 | 100010 | 1100 | 1100 | 0000 | 0 |
| 10 | 100100 | 1100 | 1100 | 1100 | 0 |
| 10 | 100101 | 1100 | 1100 | 1100 | 0 |
| 10 | 101010 | 1100 | 1100 | 0000 | 1 |