

# 4 bit Unsigned ALU

Tuesday, March 23, 2021 5:38 PM

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.NUMERIC_STD.all;
entity ALU4 is
    port(operation: in std_logic_vector(2 downto 0);
          operand0 : in std_logic_vector(3 downto 0);
          operand1 : in std_logic_vector(3 downto 0);
          result    : out std_logic_vector(3 downto 0);
          multHi    : out std_logic_vector(3 downto 0));
end ALU4;
architecture Behavioral of ALU4 is
    signal s_result    : std_logic_vector(3 downto 0);
    signal s_multRes    : std_logic_vector(7 downto 0);
begin
    process(operation, operand0, operand1)
    begin
        case operation is
            when "000" =>
                s_result <= std_logic_vector(unsigned(operand0) +
unsigned(operand1));
            when "001" =>
                s_result <= std_logic_vector(unsigned(operand0) -
unsigned(operand1));
            when "011" =>
                s_result <= std_logic_vector(unsigned(operand0) /
unsigned(operand1));
            when "100" =>
                s_result <= std_logic_vector(unsigned(operand0) rem
unsigned(operand1));
            when "101" =>
                s_result <= operand0 and operand1;
            when "110" =>
                s_result <= operand0 or operand1;
            when others =>
                s_result <= operand0 xor operand1;
        end case;
    end process;
    s_multRes <= std_logic_vector(unsigned(operand0) * unsigned(operand1));
    result    <= s_multRes(3 downto 0) when (operation = "010") else s_result;
    multHi    <= s_multRes(7 downto 4) when (operation = "010") else "0000" ;
end Behavioral;
```

## OP CODES

| OP  |                        |
|-----|------------------------|
| 000 | Addition               |
| 001 | Subtraction            |
| 010 | Unsigned multiplctaion |
| 011 | Unisgned division      |
| 100 | Unsigned reminder      |
| 101 | Bitwise AND            |
| 110 | Bitwise OR             |
| 111 | Bitwise XOR            |

## Example Simulation

