ALU simple code

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_arith.ALL;
use IEEE.STD_LOGIC_unsigned.ALL;



-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity ALU1 is
   Port ( A : in  STD_LOGIC_VECTOR (3 downto 0);
          B : in  STD_LOGIC_VECTOR (3 downto 0);
          S : in  STD_LOGIC_VECTOR (3 downto 0);
          M : in  STD_LOGIC;
          F : out  STD_LOGIC_VECTOR (3 downto 0));
end ALU1;

architecture Behavioral of ALU1 is

begin
process(A,B,S,M)
begin
if(M='0') then
case S is
```

```
When"0000"=>F<=A+B;
When"0001"=>F<=A-B;
When"0010"=>F<=A+'1';
When"0011"=>F<=A-'1';
When others=>F<="0000";
end case;
else
case S is
When "0000"=>F<=A AND B;
When "0001"=>F<=A OR B;
When "0010"=>F<=A NAND B;
When "0011"=>F<=NOT A;
When others=>F<="0000";
end case;
end if;
end process;
end Behavioral;
```

ALU testbench

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_arith.ALL;
USE ieee.std_logic_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--USE ieee.numeric_std.ALL;

ENTITY ALU1_tb IS
END ALU1_tb;

ARCHITECTURE behavior OF ALU1_tb IS
```

```
   -- Component Declaration for the Unit Under Test (UUT)


   COMPONENT ALU1
   PORT(
        A : IN  std_logic_vector(3 downto 0);
        B : IN  std_logic_vector(3 downto 0);
        S : IN  std_logic_vector(3 downto 0);
        M : IN  std_logic;
        F : OUT  std_logic_vector(3 downto 0)
       );
   END COMPONENT;
  --Inputs
  signal A : std_logic_vector(3 downto 0) := (others => '0');
  signal B : std_logic_vector(3 downto 0) := (others => '0');
  signal S : std_logic_vector(3 downto 0) := (others => '0');
  signal M : std_logic := '0';


          --Outputs
  signal F : std_logic_vector(3 downto 0);


BEGIN


          -- Instantiate the Unit Under Test (UUT)
  uut: ALU1 PORT MAP (
        A => A,
        B => B,
        S => S,
        M => M,
        F => F
       );
   -- Clock process definitions
--   <clock>_process :process
--   begin
--                    <clock> <= '0';
--                    wait for <clock>_period/2;
--                    <clock> <= '1';
--                    wait for <clock>_period/2;
--   end process;



   -- Stimulus process
   stim_proc: process
   begin
     -- hold reset state for 100 ns.
                     M<='0';A<="1101"; B<="0101"; S<="0000";
                     wait for 100 ns;
                     Assert F<=A+B; Report"in Logic";

                     M<='0';A<="1101"; B<="0101"; S<="0001";
                     wait for 100 ns;
                     Assert F<=A-B; Report"in Logic";

                     M<='0';A<="1101"; B<="0101"; S<="0010";
                     wait for 100 ns;
                     Assert F<=A+ '1'; Report"in Logic";

                     M<='0';A<="1101"; B<="0101"; S<="0011";
                     wait for 100 ns;
                     Assert F<=A- '1'; Report"in Logic";
```

```
                    M<='0';A<="1101"; B<="0101"; S<="1111";
                    wait for 100 ns;
                    Assert F<="0000"; Report"in Logic";
                    wait for 100 ns;



                    M<='1';A<="1101"; B<="0101"; S<="0000";
                    wait for 100 ns;
                    Assert F<=(A AND B); Report"in Logic";

                    M<='1';A<="1101"; B<="0101"; S<="0001";
                    wait for 100 ns;
                    Assert F<=(A OR B);Report"in Logic";

                    M<='1';A<="1101"; B<="0101"; S<="0010";
                    wait for 100 ns;
                    Assert F<=(A NAND B);Report"in Logic";

                    M<='1';A<="1101"; B<="0101"; S<="0011";
                    wait for 100 ns;
                    Assert F<=(NOT A); Report"in Logic";
                    M<='1';A<="1101"; B<="0101"; S<="1111";
                    wait for 100 ns;
                    Assert F<="0000"; Report"in Logic";
--    wait for <clock>_period*10;
    -- insert stimulus here
  wait;
  end process;
END;
```