2. 1000 : And
0001 : Or
0011 : Xnor
0110 : Add
1110 : Substract
1100 : Mul by 2

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_arith.all;

entity ALU is
    port (A,B: in std_logic_vector(31 downto 0);
      OUT 0: out std_logic_vector(31 downto 0);
          zero, overflow, carryout: out std_logic;
           carryin: in std_logic;
            cntl: in std_logic_vect(3 downto 0));
end ALU;

architecture A of ALU is
signal S: std_logic_vector(32 downto 0);
begin
process (A,B)
process (A,B, cntl)
begin
    case cntl is
        when "1000" => O<= A and B;
OUT<=A and B;      zero<='0'; overflow<='0';
    S<='0'&(A and B);    carryout<='0';
        when "0011" => OUT<= A or B;
    S<'0'&(A or B);   zero<='0'; overflow<='0';
                      carryout<='0';
```

```
when "0011" => OUT <= A xnor B; S <= '0' & (A xnor B);
               zero <= '0'; overflow <= '0'; carryout <= '0';
when "0110" => OUT <= A+B; S <= '0' & A+B; if carryin='1' then
                 if A(31)='1' and B(31)='1' then   S <= S+1;
                                                   end if;
                   overflow <= '1'; carryout <= '1';
                 end if;
when "1110" => OUT <= A-B; S <= '0' & (A-B);
                 if A<B then carryout <= '1';
                 end if;
when "1100" => if A(31)='1' then overflow <= '1';
                 end if;
                 OUT <= S and S & '0';
                   S(31 downto 0) <= A(30 downto 0) & '0';
when others => zero <= '1';
               overflow <= '0'; carryout <= '0';

end case;
end process;
  OUT <= S(31 downto 0);
end A;
    end process;
    end A;
```