```
--ALU using VHDL
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.all;
USE IEEE.NUMERIC_STD.all;                -- Needed for shifts
USE IEEE.STD_LOGIC_UNSIGNED.all

ENTITY ALU IS
 PORT(
   OP_CODE:         IN STD_LOGIC_VECTOR(4 DOWNTO 0);
   CIN:             IN STD_LOGIC_VECTOR;
   OP_A:            IN STD_LOGIC_VECTOR(7 DOWNTO 0);
   OP_B:            IN STD_LOGIC_VECTOR(7 DOWNTO 0);
   ALU_OUT:         OUT STD_LOGIC_VECTOR(7 DOWNTO 0));


--signal OP_A:    STD_LOGIC_VECTOR(7 DOWNTO 0);
signal SHL_A:   unsigned(7 DOWNTO 0);
signal SHR_A:   unsigned(7 DOWNTO 0);
END ENTITY ALU;

ARCHITECTURE ALUBehaviour OF ALU IS
BEGIN
Logic: PROCESS(OP_CODE,OP_A,OP_B,CIN)
BEGIN
  IF (OP_CODE="00000" AND CIN="1") THEN
     ALU_OUT <= OP_A;
  ELSIF (OP_CODE="00000" AND CIN="1") THEN
     ALU_OUT <= OP_A+"00000001";
  ELSIF (OP_CODE="00001" AND CIN="0") THEN
     ALU_OUT <= (OP_A + OP_B);
```

```
  ELSIF (OP_CODE="00001" AND CIN="1") THEN
     ALU_OUT <= (OP_A + OP_B + "1");
  ELSIF (OP_CODE="00010" AND CIN="0") THEN
     ALU_OUT <= (OP_A + NOT(OP_B));
  ELSIF (OP_CODE="00010" AND CIN="1") THEN
     ALU_OUT <= (OP_A + NOT(OP_B) + "1");
  ELSIF (OP_CODE="00011" AND CIN="0") THEN
     ALU_OUT <= OP_A - "1";
  ELSIF (OP_CODE="00011" AND CIN="1") THEN
     ALU_OUT <= OP_A;
  ELSIF (OP_CODE="00100") THEN
     ALU_OUT <= (OP_A AND OP_B);
  ELSIF (OP_CODE="00101") THEN
     ALU_OUT <= (OP_A OR OP_B);
  ELSIF (OP_CODE="00110") THEN
     ALU_OUT <= (OP_A XOR OP_B);
  ELSIF (OP_CODE="00111") THEN
     ALU_OUT <= NOT(OP_A);
  ELSIF (OP_CODE="01000") THEN
     ALU_OUT <= shift_left(unsigned(OP_A),7);
  ELSIF (OP_CODE="01000") THEN
     ALU_OUT <= shift_right(unsigned(OP_A),7);
  ELSIF (OP_CODE="11000")
     ALU_OUT <= "00000000";
  ELSE ALU_OUT <= "00000000";
END IF;
END PROCESS Logic;
END ARCHITECTURE ALUBehaviour;
```