```
----------------------------------------------------------------------------------
-- Company: 
-- Engineer: 
-- 
-- Create Date:    18:24:43 03/14/2022 
-- Design Name: 
-- Module Name:    ALU_8bit - Behavioral 
--      GitHub:    https://github.com/MasoudHeidary/
--     License:    MIT
----------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity ALU_8bit is
    Port ( DR1 : in  STD_LOGIC_VECTOR (7 downto 0);
           DR2 : in  STD_LOGIC_VECTOR (7 downto 0);
           Cin : in  STD_LOGIC;
           OPCODE : in  STD_LOGIC_VECTOR (3 downto 0);
           AC : out  STD_LOGIC_VECTOR (7 downto 0);
           Cout : out  STD_LOGIC);
end ALU_8bit;

architecture Behavioral of ALU_8bit is
    signal OPCODE_A         : STD_LOGIC_VECTOR (7 downto 0);
    signal OPCODE_AND_A_B   : STD_LOGIC_VECTOR (7 downto 0);
    signal OPCODE_OR_A_B    : STD_LOGIC_VECTOR (7 downto 0);
    signal OPCODE_XOR_A_B   : STD_LOGIC_VECTOR (7 downto 0);
    signal OPCODE_NOT_A     : STD_LOGIC_VECTOR (7 downto 0);

    signal A        : STD_LOGIC_VECTOR (7 downto 0);
    signal AND_A_B  : STD_LOGIC_VECTOR (7 downto 0);
    signal OR_A_B   : STD_LOGIC_VECTOR (7 downto 0);
    signal XOR_A_B  : STD_LOGIC_VECTOR (7 downto 0);
    signal NOT_A    : STD_LOGIC_VECTOR (7 downto 0);
begin
    -- instruction 1: A
    OPCODE_A(0) <= (not OPCODE(0)) and (not OPCODE(1)) and (not OPCODE(2)) and (not 
OPCODE(3) and (not Cin));
    OPCODE_A(1) <= OPCODE_A(0);
    OPCODE_A(2) <= OPCODE_A(0);
    OPCODE_A(3) <= OPCODE_A(0);
    OPCODE_A(4) <= OPCODE_A(0);
    OPCODE_A(5) <= OPCODE_A(0);
    OPCODE_A(6) <= OPCODE_A(0);
    OPCODE_A(7) <= OPCODE_A(0);

    A <= DR1 and OPCODE_A;


    -- instruction 2: A and B
    OPCODE_AND_A_B(0) <= (not OPCODE(0)) and (not OPCODE(1)) and OPCODE(2) and (not 
OPCODE(3));
    OPCODE_AND_A_B(1) <= OPCODE_AND_A_B(0);
    OPCODE_AND_A_B(2) <= OPCODE_AND_A_B(0);
    OPCODE_AND_A_B(3) <= OPCODE_AND_A_B(0);
    OPCODE_AND_A_B(4) <= OPCODE_AND_A_B(0);
    OPCODE_AND_A_B(5) <= OPCODE_AND_A_B(0);
    OPCODE_AND_A_B(6) <= OPCODE_AND_A_B(0);
    OPCODE_AND_A_B(7) <= OPCODE_AND_A_B(0);

    AND_A_B <= DR1 and DR2 and OPCODE_AND_A_B;
```

```vhdl


    -- instrction 3: A or B
    OPCODE_OR_A_B(0) <= OPCODE(0) and (not OPCODE(1)) and OPCODE(2) and (not OPCODE(3) and (not Cin));
    OPCODE_OR_A_B(1) <= OPCODE_OR_A_B(0);
    OPCODE_OR_A_B(2) <= OPCODE_OR_A_B(0);
    OPCODE_OR_A_B(3) <= OPCODE_OR_A_B(0);
    OPCODE_OR_A_B(4) <= OPCODE_OR_A_B(0);
    OPCODE_OR_A_B(5) <= OPCODE_OR_A_B(0);
    OPCODE_OR_A_B(6) <= OPCODE_OR_A_B(0);
    OPCODE_OR_A_B(7) <= OPCODE_OR_A_B(0);

    OR_A_B <= (DR1 or DR2) and OPCODE_OR_A_B;


    -- instruction 3: A xor B
    OPCODE_XOR_A_B(0) <= (not OPCODE(0)) and OPCODE(1) and OPCODE(2) and (not OPCODE(3));
    OPCODE_XOR_A_B(1) <= OPCODE_XOR_A_B(0);
    OPCODE_XOR_A_B(2) <= OPCODE_XOR_A_B(0);
    OPCODE_XOR_A_B(3) <= OPCODE_XOR_A_B(0);
    OPCODE_XOR_A_B(4) <= OPCODE_XOR_A_B(0);
    OPCODE_XOR_A_B(5) <= OPCODE_XOR_A_B(0);
    OPCODE_XOR_A_B(6) <= OPCODE_XOR_A_B(0);
    OPCODE_XOR_A_B(7) <= OPCODE_XOR_A_B(0);

    XOR_A_B <= (DR1 xor DR2) and OPCODE_XOR_A_B;


    -- instruction 4: not A
    OPCODE_NOT_A(0) <= OPCODE(0) and OPCODE(1) and OPCODE(2) and (not OPCODE(3));
    OPCODE_NOT_A(1) <= OPCODE_NOT_A(0);
    OPCODE_NOT_A(2) <= OPCODE_NOT_A(0);
    OPCODE_NOT_A(3) <= OPCODE_NOT_A(0);
    OPCODE_NOT_A(4) <= OPCODE_NOT_A(0);
    OPCODE_NOT_A(5) <= OPCODE_NOT_A(0);
    OPCODE_NOT_A(6) <= OPCODE_NOT_A(0);
    OPCODE_NOT_A(7) <= OPCODE_NOT_A(0);

    NOT_A <= (not DR1) and OPCODE_NOT_A;

    -- output
    AC <= A or AND_A_B or OR_A_B or XOR_A_B or NOT_A;

end Behavioral;
```