```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  use IEEE.STD_LOGIC_UNSIGNED.ALL;
4  use IEEE.NUMERIC_STD.ALL;
5
6
7  entity ALU is
8      Port ( A : in       STD_LOGIC_VECTOR (7 downto 0);
9             B : in       STD_LOGIC_VECTOR (7 downto 0);
10            F : out      STD_LOGIC_VECTOR (7 downto 0);
11            Cin : in     STD_LOGIC;
12            OPCODE : in  STD_LOGIC_VECTOR (3 downto 0);
13            Cout : out   STD_LOGIC);
14 end ALU;
15
16 architecture Behavioral of ALU is
17
18     -- signals
19     signal A_add_Cin:   STD_LOGIC_VECTOR (7 downto 0);
20     signal A_add_B:     STD_LOGIC_VECTOR (7 downto 0);
21     signal Sub:         STD_LOGIC_VECTOR (7 downto 0);
22     signal R_SHIFT:     STD_LOGIC_VECTOR (7 downto 0);
23     signal L_SHIFT:     STD_LOGIC_VECTOR (7 downto 0);
24
25     signal Carry:       STD_LOGIC_VECTOR (2 downto 0);
26
27     COMPONENT ADD8
28     PORT(
29         A : IN      std_logic_vector(7 downto 0);
30         B : IN      std_logic_vector(7 downto 0);
31         Cin : IN    std_logic;
32         Sum : OUT   std_logic_vector(7 downto 0);
33         Cout : OUT  std_logic
34         );
35     END COMPONENT;
36 begin
37
38     -- A + Cin
39     Inst_ADD8_0: ADD8 PORT MAP(
40         A => A,
41         B => "00000000",
42         Cin => Cin,
43         Sum => A_add_Cin,
```

```
44          Cout => Carry(0)
45      );
46
47      -- A + B + Cin
48      Inst_ADD8_1: ADD8 PORT MAP(
49          A => A,
50          B => B,
51          Cin => Cin,
52          Sum => A_add_B,
53          Cout => Carry(1)
54      );
55
56      -- A + (not B) + Cin
57      Inst_ADD8_2: ADD8 PORT MAP(
58          A => A,
59          B => (not B),
60          Cin => Cin,
61          Sum => Sub,
62          Cout => Carry(2)
63      );
64
65      -- shift to right
66      R: for i in 0 to 6 generate
67          R_SHIFT(i) <= A(i+1);
68      end generate;
69          R_SHIFT(7) <= '0';
70
71      -- shift to left
72      L: for i in 1 to 7 generate
73          L_SHIFT(i) <= A(i-1);
74      end generate;
75          L_SHIFT(0) <= '0';
76
77
78      with OPCODE (3 downto 0) select
79          F <=    A_add_Cin   when "0000",
80                  A_add_B     when "0001",
81                  Sub         when "0010",
82                  A-1       when "0011",
83                  A and B     when "0100",
84                  A or B      when "0101",
85                  A xor B     when "0110",
86                  not A       when "0111",
```

```
87                  R_SHIFT     when "1000",
88                  R_SHIFT     when "1001",
89                  R_SHIFT     when "1010",
90                  R_SHIFT     when "1011",
91                  L_SHIFT     when "1100",
92                  L_SHIFT     when "1101",
93                  L_SHIFT     when "1110",
94                  L_SHIFT     when "1111",
95                  "ZZZZZZZZ"  when others;
96
97      with OPCODE (3 downto 0) select
98          Cout <= Carry(0) when "0000",
99                  Carry(1) when "0001",
100                 Carry(2) when "0010",
101                 '0'      when others;
102
103 end Behavioral;
```