```
1   library ieee ;
2   use ieee.std_logic_1164.all;
3
4   entity ULA is
5     port (ULA_A_IN,ULA_B_IN: in std_logic_vector(7 downto 0);
6           ULA_S_IN: in std_logic_vector(3 downto 0);
7           ULA_OUT: out std_logic_vector(7 downto 0);
8           JMP_eq,JMP_hi,JMP_lo,LED_OVRF: out std_logic);
9   end ULA;
10
11  architecture ckt of ULA is
12     component SUM_8Bits is
13       port (A8_in,B8_in: in std_logic_vector(7 downto 0);
14             C8_in: in std_logic;
15             S8_out: out std_logic_vector(7 downto 0);
16             C8_out: out std_logic);
17     end component;
18
19     component SUB_8Bits is
20       port (sub_A8_in,sub_B8_in: in std_logic_vector(7 downto 0);
21             sub_S8_out: out std_logic_vector(7 downto 0));
22     end component;
23
24     component Multiplicador8Bits is
25       port (Ma, Mb: in std_logic_vector(7 downto 0);
26             Ms: out std_logic_vector(15 downto 0));
27     end component;
28
29     component Comparador_8Bits is
30       port (eA8,eB8: in std_logic_vector(7 downto 0);
31             AeqB8,AltB8,AgtB8: out std_logic);
32     end component;
33
34     component SHR_in_8Bits is
35       port (Ar: in std_logic_vector(7 downto 0);
36             Sr: out std_logic_vector(7 downto 0));
37     end component;
38
39     component SHL_in_8Bits is
40       port (Al: in std_logic_vector(7 downto 0);
41             Sl: out std_logic_vector(7 downto 0));
42     end component;
43
44     component MUX12_1_8Bits is
45       port (I12_11,I12_10,I12_9,I12_8,I12_7, I12_6, I12_5, I12_4, I12_3, I12_2, I12_1,
    I12_0: in std_logic_vector(7 downto 0);
46             S12: in std_logic_vector(3 downto 0);
47             d12: out std_logic_vector(7 downto 0));
48     end component;
49
50     signal result_soma, result_soma_ovrf, result_subtracao, result_comparacao, result_mult,
     result_mult_ovrf: std_logic_vector(7 downto 0);
51     signal result_INC, result_DEC, result_AND, result_OR, result_XOR, result_NOT,
    result_SHR, result_SHL, result_mux: std_logic_vector(7 downto 0);
52     signal mux_mult,mux_soma: std_logic_vector(7 downto 0);
53     signal result_mult_total: std_logic_vector(15 downto 0);
54     signal bloco_comparador, compara_maior_0_mult, compara_maior_0_sum, compara_mux_soma,
    compara_mux_mult: std_logic_vector(2 downto 0);
55     signal INC_ovrf, ovrf_mult, ovrf_soma: std_logic;
56
57     begin
58
59     result_soma_ovrf(7 downto 1) <="0000000";
60     ADD: SUM_8Bits port map(
61        A8_in => ULA_A_IN,
62        B8_in => ULA_B_IN,
```

```
63          C8_in => '0',
64          S8_out => result_soma,
65          C8_out => result_soma_ovrf(0));
66
67      SUB: SUB_8Bits port map(
68          sub_A8_in => ULA_A_IN,
69          sub_B8_in => ULA_B_IN,
70          sub_S8_out => result_subtracao);
71
72      result_comparacao(7 downto 3) <= "00000";
73      CMP: comparador_8Bits port map(
74          eA8 => ULA_A_IN,
75          eB8 => ULA_B_IN,
76          AeqB8 => bloco_comparador(1),
77          AltB8 => bloco_comparador(0),
78          AgtB8 => bloco_comparador(2));
79      result_comparacao(2 downto 0) <= bloco_comparador;
80
81      MULT: multiplicador8Bits port map(
82          Ma => ULA_A_IN,
83          Mb => ULA_B_IN,
84          Ms => result_mult_total);
85      result_mult <= result_mult_total(7 downto 0);
86      result_mult_ovrf <= result_mult_total(15 downto 8);
87
88      INC: SUM_8Bits port map(
89          A8_in => ULA_A_IN,
90          B8_in => "00000001",
91          C8_in => '0',
92          S8_out => result_INC,
93          C8_out => INC_ovrf);
94
95      DEC: SUB_8Bits port map(
96          sub_A8_in => ULA_A_IN,
97          sub_B8_in => "00000001",
98          sub_S8_out => result_DEC);
99
100     result_AND <= ULA_A_IN and ULA_B_IN;
101     result_OR <= ULA_A_IN or ULA_B_IN;
102     result_XOR <= ULA_A_IN xor ULA_B_IN;
103     result_NOT <= not ULA_A_IN;
104
105     SHL: SHL_in_8Bits port map(
106         Al => ULA_A_IN,
107         Sl => result_SHL);
108
109     SHR: SHR_in_8Bits port map(
110         Ar => ULA_A_IN,
111         Sr => result_SHR);
112
113     MUX: MUX12_1_8Bits port map(
114         I12_11 => result_SHR,
115         I12_10 => result_SHL,
116         I12_9 => result_NOT,
117         I12_8 => result_XOR,
118         I12_7 => result_OR,
119         I12_6 => result_AND,
120         I12_5 => result_DEC,
121         I12_4 => result_INC,
122         I12_3 => result_mult,
123         I12_2 => result_comparacao,
124         I12_1 => result_subtracao,
125         I12_0 => result_soma,
126         S12 => ULA_S_IN,
127         d12 => result_mux);
128
```

```
129      comparador_mult_ovrf: comparador_8Bits port map(
130         eA8 => result_mult_ovrf,
131         eB8 => "00000000",
132         AeqB8 => compara_maior_0_mult(1),
133         AltB8 => compara_maior_0_mult(0),
134         AgtB8 => compara_maior_0_mult(2));
135
136      comparador_soma_ovrf: comparador_8Bits port map(
137         eA8 => result_soma_ovrf,
138         eB8 => "00000000",
139         AeqB8 => compara_maior_0_sum(1),
140         AltB8 => compara_maior_0_sum(0),
141         AgtB8 => compara_maior_0_sum(2));
142
143      mux_mult(7 downto 4) <= "0000";
144      mux_mult(3 downto 0) <= ULA_S_IN;
145      mux_soma(7 downto 4) <= "0000";
146      mux_soma(3 downto 0) <= ULA_S_IN;
147
148      comparador_mux_soma: comparador_8Bits port map(
149         eA8 => mux_soma,
150         eB8 => "00000000",
151         AeqB8 => compara_mux_soma(1),
152         AltB8 => compara_mux_soma(0),
153         AgtB8 => compara_mux_soma(2));
154      comparador_mux_mult: comparador_8Bits port map(
155         eA8 => mux_mult,
156         eB8 => "00000011",
157         AeqB8 => compara_mux_mult(1),
158         AltB8 => compara_mux_mult(0),
159         AgtB8 => compara_mux_mult(2));
160
161      ULA_OUT <= result_mux;
162      ovrf_mult <= compara_maior_0_mult(2) and compara_mux_mult(1);
163      ovrf_soma <= compara_maior_0_sum(2) and compara_mux_soma(1);
164      LED_OVRF <= ovrf_mult or ovrf_soma;
165      JMP_hi <= bloco_comparador(2);
166      JMP_eq <= bloco_comparador(1);
167      JMP_lo <= bloco_comparador(0);
168
169   end ckt;
```