```
1
2   ------------------------------------3)  INTEGER DIVISION---------------------------------
3
4   library IEEE;
5   use IEEE.STD_LOGIC_1164.ALL;
6   use IEEE.numeric_std.all;
7
8   entity divison1 is
9       Port ( a : in  unsigned(7 downto 0);
10             b : in  unsigned(7 downto 0);
11             q : out unsigned(7 downto 0);
12  --         r : out unsigned(7 downto 0));
13              suc_flag:out std_logic);
14  end divison1;
15
16  architecture Behave of divison1 is
17  --signal q : unsigned(7 downto 0) := "00000000";  ----------- quotient
18  --signal r : unsigned(7 downto 0) := "00000000";   ------------- remainder
19  begin
20
21  process(a, b)
22
23  variable s7 : unsigned(7 downto 0) := "00000000";
24  variable s8 : unsigned(7 downto 0) := "00000000";
25  variable s6 : unsigned(7 downto 0) := "00000000";
26  variable s3 : unsigned(7 downto 0) := "00000000";
27  variable m : unsigned(7 downto 0) := "00000000";
28  variable t : unsigned(14 downto 0) := "000000000000000";
29  variable check1 : STD_LOGIC;
30  variable check2 : STD_LOGIC;
31
32  begin
33
34  if (a(7) = '1')                    ------------------- Checking if dividend is negative or positive in 2's
    complement form
35  then
36   check1 := '1';
37   s3 := unsigned(a);
38   s3 := not(s3);
39   s3 := s3 + "00000001";
40  else
41   check1 := '0';
```

```
42     s3 := unsigned(a);
43    end if;
44
45    t(6 downto 0) := (s3(6 downto 0));
46    t(14 downto 7) := "00000000";
47
48
49    if (b(7) = '1')                        ------------------- Checking if divisor is negative or positive in 2's
      complement form
50    then
51     check2 := '1';
52     s3 := unsigned(b);
53     s3 := not(s3);
54     s3 := s3 + "00000001";
55    else
56     check2 := '0';
57     s3 := unsigned(b);
58    end if;
59
60    m := s3;
61
62    for k in 6 downto 0                 ----------------- Division
63    loop
64    t := t sll 1;
65    t(14 downto 7) := t(14 downto 7) - m;
66
67    if (t(14) = '1')
68    then
69       t(0) := '0';
70       t(14 downto 7) := t(14 downto 7) + m;
71    else
72       t(0) := '1';
73    end if;
74    end loop;
75
76    if(check1 = '0' and check2 = '0')       ---------- Making quotient negative if any of dividend and divisor is
      negative
77    then                                    ---------- Making remainder negative if any of dividend is negative
78     s7 := t(14 downto 7);
79     s6(6 downto 0) := t(6 downto 0);
80
81    elsif(check1 = '1' and check2 = '1')
```

```
 82   then
 83    s6(6 downto 0) := t(6 downto 0);
 84    s3 := t(14 downto 7);
 85    s8 := "00000000";
 86    s7 := s8 - s3;
 87
 88   elsif(check1 = '1' and check2 = '0')
 89   then
 90    s3(6 downto 0) := (t(6 downto 0));
 91    s8 := "00000000";
 92    s3(7) := '0';
 93    s6 := s8 - s3;
 94    s3 := (t(14 downto 7));
 95    s7 := s8 - s3;
 96
 97   elsif(check1 = '0' and check2 = '1')
 98   then
 99    s7 := t(14 downto 7);
100    s3(6 downto 0) := (t(6 downto 0));
101    s3(7) := '0';
102    s6 := s8 - s3;
103   end if;
104
105   --r  <= s7;
106   q <= s6;
107   suc_flag<='1';
108   end process;
109   end Behave;
110
```