```vhdl
----------------------------------------------------------------------
-- This component converts the sum and price signals with BCD into
-- four seven segment displays. The code for displaying hexadecimal
-- numbers as well as additional letters are also contained.
-- This component also receives an alarm signal which causes the
-- display to blink. The cola, aqua and hash signals, if positive,
-- causes the display to show these words.
----------------------------------------------------------------------

library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity display_driver is
    port(sum   : in  unsigned(5 downto 0);
         price : in  unsigned(5 downto 0);
         reset : in  std_logic;
         clock : in  std_logic;
         clk_3 : in  std_logic;
         alarm : in  std_logic;
         cola  : in  std_logic;
         aqua  : in  std_logic;
         hash  : in  std_logic;
         an    : out std_logic_vector(3 downto 0);
         led   : out std_logic_vector(1 to 8));
end display_driver;

architecture Behavior of display_driver is
    signal m, m_next      : unsigned(1 downto 0);
    signal d              : unsigned(4 downto 0);
    signal sumL, sumH     : unsigned(4 downto 0);
    signal priceL, priceH : unsigned(4 downto 0);
    signal anTemp         : std_logic_vector(3 downto 0);

begin
    m_next <= m + 1;

    ----------------------------------------
    -- BCD converter for price input signal
    ----------------------------------------
    process(price)
    begin
        if price >= 60 then
            priceH <= "00110";
            priceL <= price - 60;
        elsif price >= 50 then
            priceH <= "00101";
            priceL <= price - 50;
```

```
49          elsif price >= 40 then
50              priceH <= "00100";
51              priceL <= price - 40;
52          elsif price >= 30 then
53              priceH <= "00011";
54              priceL <= price - 30;
55          elsif price >= 20 then
56              priceH <= "00010";
57              priceL <= price - 20;
58          elsif price >= 10 then
59              priceH <= "00001";
60              priceL <= price - 10;
61          else
62              priceH <= "00000";
63              priceL <= price(4 downto 0);
64          end if;
65      end process;
66
67      ---------------------------------------
68      -- BCD converter for sum input signal
69      ---------------------------------------
70      process(sum)
71      begin
72          if sum >= 60 then
73              sumH <= "00110";
74              sumL <= sum - 60;
75          elsif sum >= 50 then
76              sumH <= "00101";
77              sumL <= sum - 50;
78          elsif sum >= 40 then
79              sumH <= "00100";
80              sumL <= sum - 40;
81          elsif sum >= 30 then
82              sumH <= "00011";
83              sumL <= sum - 30;
84          elsif sum >= 20 then
85              sumH <= "00010";
86              sumL <= sum - 20;
87          elsif sum >= 10 then
88              sumH <= "00001";
89              sumL <= sum - 10;
90          else
91              sumH <= "00000";
92              sumL <= sum(4 downto 0);
93          end if;
94      end process;
95
96      ---------------------------------------------------------------------
```

```
97      -- m is the multiplexer select signal which selects which of the sever
98      -- segment displays are turned on at any given time. There is a sequen
99      -- hierarchy in which cola overrules aqua, which overrules hash. If or
100     -- these signals = '1', the word is shown on the display, if none of t
101     -- signals = '1', price is shown on the two first seven seg displays w
102     -- sum is shown on the last two.
103     ----------------------------------------------------------------------
104     process(m)
105     begin
106         if cola = '1' then
107             case m is
108                 when "00" =>
109                     anTemp <= NOT "0001";
110                     d      <= "01010";
111                 when "01" =>
112                     anTemp <= NOT "0010";
113                     d      <= "10010";
114                 when "10" =>
115                     anTemp <= NOT "0100";
116                     d      <= "00000";
117                 when "11" =>
118                     anTemp <= NOT "1000";
119                     d      <= "01100";
120                 when others =>
121                     anTemp <= NOT "0000";
122                     d      <= "00000";
123             end case;
124         elsif aqua = '1' then
125             case m is
126                 when "00" =>
127                     anTemp <= NOT "0001";
128                     d      <= "01010";
129                 when "01" =>
130                     anTemp <= NOT "0010";
131                     d      <= "10011";
132                 when "10" =>
133                     anTemp <= NOT "0100";
134                     d      <= "01001";
135                 when "11" =>
136                     anTemp <= NOT "1000";
137                     d      <= "01010";
138                 when others =>
139                     anTemp <= NOT "0000";
140                     d      <= "00000";
141             end case;
142         elsif hash = '1' then
143             case m is
144                 when "00" =>
```

```
145                     anTemp <= NOT "0001";
146                     d      <= "10000";
147                 when "01" =>
148                     anTemp <= NOT "0010";
149                     d      <= "10001";
150                 when "10" =>
151                     anTemp <= NOT "0100";
152                     d      <= "01010";
153                 when "11" =>
154                     anTemp <= NOT "1000";
155                     d      <= "10000";
156                 when others =>
157                     anTemp <= NOT "0000";
158                     d      <= "00000";
159             end case;
160         else
161             case m is
162                 when "00" =>
163                     anTemp <= NOT "0001";
164                     d      <= sumL;
165                 when "01" =>
166                     anTemp <= NOT "0010";
167                     d      <= sumH;
168                 when "10" =>
169                     anTemp <= NOT "0100";
170                     d      <= priceL;
171                 when "11" =>
172                     anTemp <= NOT "1000";
173                     d      <= priceH;
174                 when others =>
175                     anTemp <= NOT "0000";
176                     d      <= "00000";
177             end case;
178         end if;
179     end process;
180
181     ----------------------------------------------------------------------
182     -- if the alarm signal is asserted, the whole display will blink on th
183     -- 3Hz clock. This process runs parallel with the above process, meani
184     -- that e.g. the alarm signal can be asserted while 'cola' is displaye
185     ----------------------------------------------------------------------
186     process(alarm, clk_3)
187     begin
188         an <= anTemp;
189         if (alarm = '1' AND clk_3 = '0') then
190             an <= NOT "0000";
191         end if;
192     end process;
```

```
193
194     ------------------------------------------------------------
195     -- this process assigns each hexadecimal number (0-F) as well
196     -- as some additional letters to different values of d.
197     ------------------------------------------------------------
198     process(d)
199     begin
200         case d is
201             when "00000" => led <= NOT "11111100"; -- 0
202             when "00001" => led <= NOT "01100000"; -- 1
203             when "00010" => led <= NOT "11011010"; -- 2
204             when "00011" => led <= NOT "11110010"; -- 3
205             when "00100" => led <= NOT "01100110"; -- 4
206             when "00101" => led <= NOT "10110110"; -- 5
207             when "00110" => led <= NOT "10111110"; -- 6
208             when "00111" => led <= NOT "11100000"; -- 7
209             when "01000" => led <= NOT "11111110"; -- 8
210             when "01001" => led <= NOT "11100110"; -- 9
211             when "01010" => led <= NOT "11101110"; -- A
212             when "01011" => led <= NOT "00111110"; -- b
213             when "01100" => led <= NOT "10011100"; -- C
214             when "01101" => led <= NOT "01111010"; -- d
215             when "01110" => led <= NOT "10011110"; -- E
216             when "01111" => led <= NOT "10001110"; -- F
217             when "10000" => led <= NOT "01101110"; -- H
218             when "10001" => led <= NOT "10110110"; -- S
219             when "10010" => led <= NOT "00011100"; -- L
220             when "10011" => led <= NOT "01111100"; -- U
221             when others  => led <= (others => '0');
222         end case;
223     end process;
224
225     ------------------------------------------------
226     -- the two bit vector m is increased by one on
227     -- each positive clock edge of the 762Hz clock
228     ------------------------------------------------
229     process(clock)
230     begin
231         if rising_edge(clock) then
232             m <= m_next;
233         end if;
234     end process;
235 end Behavior;
236
```