30010 Programmeringsprojekt: Reflex Ball



Mads Friis Bornebusch (s123627)



Tobias Tuxen (s120213)



Kristian Sloth Lauszus (s123808)

Reflex Ball Group Number: 15 DTU Space June 24, 2013

Resumé

I dette programmeringsprojekt har vi skrevet og designet al koden til spillet Reflex Ball. Koden er skrevet i C i compileren Z8Encore! og implementeret på en Zilog 6403 mircocontroller.

Vi har konkluderet at den skrevne kode implementerer det ønskede kredsløb på Mircrocontrolleren, da alle spillets facetter er blevet gennemtestet og giver det ønskede output. Som slutresultat har vi et fuldt funktionelt ReflexBall-spil med mange udvidelser der bl.a. inkluderer brikker (Arkanoid-stil), power-ups, forskellige levels og styring med ret. Der er indtil videre ikke fundet nogle bugs i slutversionen af spillet.

Abstract

In this programming project we have written and designed all the code to the game Reflex Ball. The code has been written in C in the compiler Z8Encore! and has been implemented on a Zilog 6403 Mircrocontroller.

We have concluded that the written code implements the desired circuit on the Microcontroller as as all the different facilities of the game has been thoroughly tested and is in compliance with the expected result. In the end we have a fully functional Reflex Ball Game with many expansions, which amongst others, include bricks (Arkanoid style), power-ups, different levels and steering-wheel game controller. So far no bugs has been found in the final version of the game.

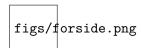


Figure 1: Screenshot from our implementation of ReflexBall

Fordord

Denne rapport er skrevet som en del af eksaminationen i DTU kursus 30010 Programmeringsprojekt. Alle tre, på forsiden nævnte, gruppemedlemmer har bidraget til rapporten på lige vis. Vi har lavet alle forberedelsesøvelser, skrevet koden og afsnittene i rapporten i fællesskab. Denne rapport beskriver vores arbejde og resultater.

Contents

1 Introduktion

This report documents the design and implementation of the control part of a vending-machine for soft-drinks.

The project was divided into three laboratory exercises, in which parts of the system was designed. After the finalization of all three laboratory exercises a complete vending machine control circuit was designed and implemented on a Basys2 FPGA board. The optional parts of the lab exercises is included in the elaboration of each of the exercises, which also redefines some of the mandatory functions, and update/change them as the requirements of each part changes a we move towards the finished control part.

2 Konklusion

Efter spillet et blevet designet, skrevet og uploadet til Zilog 6403 microcontrolleren, er alle spillets facetter blevet gennemtestet og fundet i overensstemmelse med det forventede resultat. Vi kan derfor konkludere at den skrevne kode implementerer spillet Reflex Ball på microcontrolleren og i hyperterminalen korrekt i forhold til specifikationskravene. Derudover kan vi også konkludere at alle udvidelse virker som forventet.

A Appendiks 1

This appendix contains the complete VHDL-code used, and written/modified by us

/Users/hbm/Dropbox/DTU-Amigos/0...ngMachine/src/clock_manager.vhd Page 1 of 2 Saved: 5/5/13 7:51:45 PM Printed For: Hjalte Bested Møller

```
-- This component divides the 50 MHz clock signal down to a clk signal
   -- of 762Hz and a clk_3 signal of 3Hz.
4
5
6 library ieee;
   use ieee.std_logic_1164.all;
8 use ieee.numeric_std.all;
10 entity clock_manager is
11
       port(clk_50 : in std_logic;
12
            clk_man : in std_logic;
13
            sel_man : in std_logic;
            clk : out std_logic;
clk_3 : out std_logic);
14
15
16 end clock_manager;
17
   architecture structure of clock_manager is
18
       signal count, count_next : unsigned(15 downto 0) := (others => '0');
19
       signal count3, count3_next : unsigned(23 downto 0) := (others => '0');
20
       attribute clock_signal : string;
21
       attribute clock_signal of clk : signal is "yes";
22
23
24
  begin
       count_next <= count + 1;</pre>
25
       count3 next <= count3 + 1;</pre>
26
27
28
29
   -- this process generates the clk and clk3 signal, which are derived from
   -- the 50MHz clock. When MSB of count = '1' clk will go high, when MSB of
30
   -- count = '0' clk will go low. Same with count3 and clk_3. If sel_man is
31
   -- set to 1, clk will be set to the input signal clk_man.
32
33
34
35
       process(sel_man, clk_man, count, count3)
       begin
36
           if sel_man = '1' then
37
               clk <= clk_man;</pre>
38
39
           else
40
               clk <= count(15);</pre>
41
           end if;
42
           clk_3 <= count3(23);
       end process;
43
44
45
46
   -- on the rising edge of the 50MHz clock
   -- count and count3 will be increased by 1
47
48
```

Figure 2: The clock manager, include our own 3 Hz clock for use in the alarm signal

/Users/hbm/Dropbox/DTU-Amigos/0...ngMachine/src/clock_manager.vhd Page 2 of 2 Saved: 5/5/13 7:51:45 PM Printed For: Hjalte Bested Møller

```
49
       process(clk_50, count, count3)
50
51
       begin
            if rising\_edge(clk\_50) then
52
53
                count <= count_next;</pre>
                count3 <= count3_next;</pre>
54
55
            end if;
       end process;
56
57
58 end structure;
```

Figure 3: Page 2

/Users/hbm/Dropbox/DTU-Amigos/0...gMachine/src/display_driver.vhd Page 1 of 5 Saved: 5/2/13 11:26:16 AM Printed For: Hjalte Bested Møller

```
-- This component converts the sum and price signals with BCD into
  -- four seven segment displays. The code for displaying hexadecimal
4 -- 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,
7 -- causes the display to show these words.
8
9
10 library ieee;
use ieee.std_logic_1164.all;
12 use ieee.numeric_std.all;
13
  entity display_driver is
14
      port(sum : in unsigned(5 downto 0);
15
            price : in unsigned(5 downto 0);
16
            reset : in std_logic;
17
18
            clock : in std_logic;
            clk_3 : in std_logic;
19
20
            alarm : in std_logic;
            cola : in std_logic;
21
            aqua : in std_logic;
hash : in std_logic;
22
23
24
            an
                  : out std_logic_vector(3 downto 0);
            led : out std_logic_vector(1 to 8));
25
26 end display_driver;
27
28
  architecture Behavior of display_driver is
       signal m, m_next : unsigned(1 downto 0);
29
                              : unsigned(4 downto 0);
30
       signal d
                          : unsigned(4 downto 0);
31
       signal sumL, sumH
32
       signal priceL, priceH : unsigned(4 downto 0);
                              : std_logic_vector(3 downto 0);
33
       signal anTemp
34
35
  begin
      m_next <= m + 1;</pre>
36
37
38
39
       -- BCD converter for price input signal
40
41
       process(price)
       begin
42
           if price >= 60 then
43
               priceH <= "00110";</pre>
44
45
               priceL <= price - 60;</pre>
46
           elsif price >= 50 then
               priceH <= "00101";</pre>
47
48
               priceL <= price - 50;</pre>
```

Figure 4: Display driver for displaying symbols/numbers on each of the four seven segment displays

/Users/hbm/Dropbox/DTU-Amigos/0...gMachine/src/display_driver.vhd Page 2 of 5 Saved: 5/2/13 11:26:16 AM Printed For: Hjalte Bested Møller

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

Figure 5: Page 2

/Users/hbm/Dropbox/DTU-Amigos/0...gMachine/src/display_driver.vhd Page 3 of 5 Saved: 5/2/13 11:26:16 AM Printed For: Hjalte Bested Møller

```
97
       -- m is the multiplexer select signal which selects which of the sever
       -- segment displays are turned on at any given time. There is a sequer
98
99
       -- hierarchy in which cola overrules aqua, which overrules hash. If or
       -- these signals = '1', the word is shown on the display, if none of t
100
       -- signals = '1', price is shown on the two first seven seg displays w
101
102
       -- sum is shown on the last two.
103
       process(m)
104
       begin
105
106
           if cola = '1' then
107
               case m is
                    when "00" =>
108
                        anTemp <= NOT "0001";
109
                             <= "01010";
                        d
110
                    when "01" =>
111
                        anTemp <= NOT "0010";
112
                        d <= "10010";
113
                    when "10" =>
114
                        anTemp <= NOT "0100";
115
                        d <= "00000";
116
                    when "11" =>
117
                        anTemp <= NOT "1000";
118
119
                        d <= "01100";
120
                    when others =>
                        anTemp <= NOT "0000";
121
                               <= "000000";
122
               end case;
123
124
           elsif aqua = '1' then
125
               case m is
                    when "00" =>
126
                        anTemp <= NOT "0001";
127
                              <= "01010";
128
                        d
                    when "01" =>
129
                        anTemp <= NOT "0010";
130
                        d <= "10011";
131
                    when "10" =>
132
                        anTemp <= NOT "0100";
133
                        d <= "01001";
134
135
                    when "11" =>
136
                        anTemp <= NOT "1000";
137
                        d
                             <= "01010";
                    when others =>
138
                        anTemp <= NOT "0000";
139
                               <= "00000";
140
141
               end case;
           elsif hash = '1' then
142
143
               case m is
                    when "00" =>
144
```

Figure 6: Page 3

/Users/hbm/Dropbox/DTU-Amigos/0...gMachine/src/display_driver.vhd Page 4 of 5 Saved: 5/2/13 11:26:16 AM Printed For: Hjalte Bested Møller

```
anTemp <= NOT "0001";
145
                                <= "10000";
146
                    when "01" =>
147
                        anTemp <= NOT "0010";
148
                        d <= "10001";
149
                    when "10" =>
150
                        anTemp <= NOT "0100";
151
                              <= "01010";
                        d
152
                    when "11" =>
153
                        anTemp <= NOT "1000";
154
155
                               <= "10000";
156
                    when others =>
                        anTemp <= NOT "0000";
157
                               <= "000000";
158
159
                end case;
160
            else
                case m is
161
                    when "00" =>
162
                        anTemp <= NOT "0001";
163
164
                        d
                              <= sumL;
                    when "01" =>
165
                        anTemp <= NOT "0010";
166
167
                        d
                               <= sumH;
                    when "10" =>
168
                        anTemp <= NOT "0100";
169
                        d
                              <= priceL;
170
                    when "11" =>
171
172
                        anTemp <= NOT "1000";
173
                        d
                               <= priceH;
174
                    when others =>
                        anTemp <= NOT "0000";
175
                               <= "000000";
176
177
                end case;
            end if;
178
       end process;
179
180
181
       -- if the alarm signal is asserted, the whole display will blink on th
182
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 display€
185
186
       process(alarm, clk_3)
       begin
187
            an <= anTemp;</pre>
188
            if (alarm = '1' AND clk_3 = '0') then
189
                an <= NOT "0000";
190
            end if;
191
       end process;
192
```

Figure 7: Page 4

/Users/hbm/Dropbox/DTU-Amigos/0...gMachine/src/display_driver.vhd Page 5 of 5 Saved: 5/2/13 11:26:16 AM Printed For: Hjalte Bested Møller

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

Figure 8: Page 5

/Users/hbm/Dropbox/DTU-Amigos/0...hine/src/input_synchronizer.vhd Page 1 of 4 Saved: 5/2/13 11:53:51 AM Printed For: Hjalte Bested Møller

```
-- This component synchronizes input signals produced from outside the
  -- board with the clock signal on the board. This is done to avoid
  -- metastability. Also this component makes sure that these signals will
  -- only be asserted for one clock cycle each, even if the input signals
  -- are asserted from outside the board for several clock cycles.
9 library ieee;
10 use ieee.std_logic_1164.all;
12 entity input_synchronizer is
13
      port(clock
                     : in std_logic;
                     : in std_logic;
           buy_btn
14
15
           buy_out
                    : out std_logic;
           coin1_btn : in std_logic;
16
           coin1_out : out std_logic;
17
           coin2_btn : in std_logic;
18
           coin2_out : out std_logic;
19
20
           coin5_btn : in std_logic;
21
           coin5_out : out std_logic;
           cola_sw : in std_logic;
22
           cola_out : out std_logic;
23
24
           hash_sw
                     : in std_logic;
25
           hash_out : out std_logic;
26
                     : in std_logic;
           aqua_sw
           aqua_out
                     : out std_logic;
                     : in std_logic);
           Reset
28
  end input_synchronizer;
29
30
  architecture Structure of input_synchronizer is
31
       signal buy_btn_synk, buy_btn_synk_new
                                                : std_logic;
32
       signal coin1_btn_synk, coin1_btn_synk_new : std_logic;
33
34
       signal coin2_btn_synk, coin2_btn_synk_new : std_logic;
       signal coin5_btn_synk, coin5_btn_synk_new : std_logic;
       signal cola_sw_synk, cola_sw_synk_new
                                               : std_logic;
36
37
       signal hash_sw_synk, hash_sw_synk_new
                                                 : std_logic;
38
       signal aqua_sw_synk, aqua_sw_synk_new
                                                 : std_logic;
39
40 begin
41
42
43
      -- the seven processes below asserts the signal_out only when
       -- the synkd version of the input signal is '1' and the synk_new
44
       -- version of the input signal is '0'. This makes signal_out
45
       -- last for exactly one clock cycle.
46
47
48
```

Figure 9: VHDL code for the input synchronizer which synchronizes the input signals to the clock

/Users/hbm/Dropbox/DTU-Amigos/0...hine/src/input_synchronizer.vhd Page 2 of 4 Saved: 5/2/13 11:53:51 AM Printed For: Hjalte Bested Møller

```
-- buy button
49
50
       process(buy_btn_synk)
51
       begin
           buy_out <= '0';</pre>
52
           if buy_btn_synk = '1' AND buy_btn_synk_new = '0' then
53
               buy_out <= '1';
54
           end if;
55
56
       end process;
57
       -- coin1 button
58
       process(coin1_btn_synk)
59
60
       begin
61
           coin1_out <= '0';
           if coin1_btn_synk = '1' AND coin1_btn_synk_new = '0' then
62
               coin1_out <= '1';
63
           end if;
64
       end process;
65
66
       -- coin2 button
67
68
       process(coin2_btn_synk)
69
       begin
           coin2_out <= '0';</pre>
70
71
           if (coin2_btn_synk = '1' AND coin2_btn_synk_new = '0') then
               coin2_out <= '1';
72
           end if;
73
74
       end process;
75
       -- coin5 button
76
       process(coin5_btn_synk)
77
78
79
           coin5_out <= '0';
           if coin5_btn_synk = '1' AND coin5_btn_synk_new = '0' then
80
               coin5_out <= '1';
81
           end if;
82
       end process;
83
84
       -- cola switch
85
86
       process(cola_sw_synk)
87
       begin
           cola_out <= '0';</pre>
88
           if cola_sw_synk = '1' AND cola_sw_synk_new = '0' then
89
               cola_out <= '1';
90
           end if;
91
       end process;
92
93
       -- hash switch
94
95
       process(hash_sw_synk)
96
       begin
```

Figure 10: Page 2

/Users/hbm/Dropbox/DTU-Amigos/0...hine/src/input_synchronizer.vhd Page 3 of 4 Saved: 5/2/13 11:53:51 AM Printed For: Hjalte Bested Møller

```
hash out <= '0';
97
            if hash_sw_synk = '1' AND hash_sw_synk_new = '0' then
    hash_out <= '1';</pre>
98
99
100
             end if;
101
        end process;
102
        -- aqua switch
103
104
        process(aqua_sw_synk)
105
        begin
             aqua_out <= '0';
106
             if aqua_sw_synk = '1' AND aqua_sw_synk_new = '0' then aqua_out <= '1';
107
108
109
            end if;
        end process;
110
111
112
        -- This process keeps updating reqisters
113
        -- on every rising clock edge
114
115
116
        process(clock)
117
        begin
             if rising_edge(clock) then
118
                 -- buy button
119
                 buy_btn_synk
                                    <= buy_btn;
120
121
                 buy_btn_synk_new <= buy_btn_synk;</pre>
122
                 -- coin1 button
123
                 coin1_btn_synk
                                      <= coin1_btn;
124
                 coin1_btn_synk_new <= coin1_btn_synk;</pre>
125
126
127
                 -- coin2 button
                 coin2_btn_synk
                                      <= coin2_btn;
128
129
                 coin2_btn_synk_new <= coin2_btn_synk;</pre>
130
                 -- coin5 button
131
                 coin5_btn_synk
                                      <= coin5_btn;
132
                 coin5_btn_synk_new <= coin5_btn_synk;</pre>
133
134
135
                 -- cola switch
136
                 cola_sw_synk
                                    <= cola_sw;
                 cola_sw_synk_new <= cola_sw_synk;</pre>
137
138
139
                 -- hash switch
140
                 hash_sw_synk
                                    <= hash_sw;
                 hash_sw_synk_new <= hash_sw_synk;
141
142
143
                 -- aqua switch
144
                 aqua_sw_synk
                                    <= aqua_sw;
```

Figure 11: Page 3

/Users/hbm/Dropbox/DTU-Amigos/0...hine/src/input_synchronizer.vhd Page 4 of 4 Saved: 5/2/13 11:53:51 AM Printed For: Hjalte Bested Møller

```
145
146
147
148
149
end Structure;
aqua_sw_synk_new <= aqua_sw_synk;
end aqua_sw_synk,
end structure;
```

Figure 12: Page 4

/Users/hbm/Dropbox/DTU-Amigos/0...Machine/src/processing_unit.vhd Page 1 of 3 Saved: 5/5/13 8:27:12 PM Printed For: Hjalte Bested Møller

```
-- This component is the central processing unit of the vending machine.
3
  -- This includes the following operations:
           When one of the coin inputs are asserted, sum is updated
           When one of the price_product inputs are asserted, price is update
5
           If a buy is attempted:
6
               sum will be deducted from price if sum >= price
   --
8
               alarm signal will be asserted if sum < price
10
11 library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
14
15 entity processing_unit is
                    : in std_logic;
16
       port(clock
                                            -- Clock signal in 762Hz
17
            clk_3
                      : in std_logic;
                                            -- Clock signal in 3Hz
                       : in std_logic;
: in std_logic;
18
            buy
            coin1
19
20
            coin2
                       : in std_logic;
21
            coin5
                       : in std_logic;
            price_cola : in std_logic;
22
            price_hash : in std_logic;
23
            price_aqua : in std_logic;
24
                      : in std_logic;
25
            Reset
26
                       : out unsigned(5 downto 0);
            sum_out
            price_out : out unsigned(5 downto 0);
27
            alarm_out : out std_logic;
28
29
            cola_out
                       : out std_logic;
                      : out std_logic;
: out std_logic);
30
            hash_out
31
            aqua_out
32
  end processing_unit;
33
34
  architecture Behavioral of processing_unit is
       signal sum, price
                                            : unsigned(5 downto 0);
35
       signal alarm_count, alarm_count_next : unsigned(10 downto 0);
36
37
       signal alarm, cola, hash, aqua
                                            : std_logic;
                                            : unsigned(10 downto 0);
38
       signal cola_count, cola_count_next
39
       signal hash_count, hash_count_next
                                             : unsigned(10 downto 0);
       signal aqua_count, aqua_count_next
                                            : unsigned(10 downto 0);
40
41
42 begin
43
       alarm_count_next <= alarm_count + 1;</pre>
       cola_count_next <= cola_count + 1;</pre>
44
       hash_count_next <= hash_count + 1;
45
46
       aqua_count_next <= aqua_count + 1;
47
48
```

Figure 13: Processing unit, for calculating/setting price/sum

/Users/hbm/Dropbox/DTU-Amigos/0...Machine/src/processing_unit.vhd Page 2 of 3 Saved: 5/5/13 8:27:12 PM Printed For: Hjalte Bested Møller

```
-- This process sets the price of the 3 products. When one of the
49
       -- price_'product' signals are asserted, the 'product' signal is set -- to '1', which will cause 'product'_count to keep adding + 1 on
50
51
       -- every clock, until its MSB = '1' and the 'product' signal will be -- set back to '0'. For both the price_'product' and the 'product'
52
53
       -- signals, cola overrules hash, which overrules aqua.
54
55
56
57
       process(price_cola, price_hash, price_aqua, clock)
58
        begin
            if rising_edge(clock) then
    if price_cola = '1' then
59
60
                      cola <= '1';
price <= "010010";
61
62
                 elsif price_hash = '1' then
63
                      hash <= '1';
price <= "110111";
64
65
                 elsif price_aqua = '1' then
66
                      aqua <= '1';
67
                      price <= "001100";
68
                 elsif cola = '1' then
69
                      cola_count <= cola_count_next;</pre>
70
                      if cola_count(10) = '1' then
71
                                   <= '0';
72
                           cola
                           cola_count <= "00000000000";
73
74
                      end if;
                 elsif hash = '1' then
75
                      hash_count <= hash_count_next;</pre>
76
                      if hash_count(10) = '1' then
77
                                        <= '0';
78
                           hash
                           hash_count <= "00000000000";
79
                      end if;
80
                 elsif aqua = '1' then
81
                      aqua_count <= aqua_count_next;</pre>
82
                      if aqua_count(10) = '1' then
83
                                     <= '0';
                           aqua
84
                           aqua_count <= "00000000000";
85
                      end if;
86
87
                 end if;
            end if;
88
            cola_out <= cola;
89
            hash_out <= hash;
90
91
            aqua_out <= aqua;
       end process;
92
93
94
       -- This process adds the coin value to sum when a coin input is
95
96
        -- asserted. If buy is asserted sum will be deducted from price,
```

Figure 14: Page 2

/Users/hbm/Dropbox/DTU-Amigos/0...Machine/src/processing_unit.vhd Page 3 of 3 Saved: 5/5/13 8:27:12 PM Printed For: Hjalte Bested Møller

```
-- if sum >= price, else alarm will be set to '1', which causes
97
        -- alarm_count to be increased by one on every clock until MSB
98
        -- of alarm_count equals '1' which resets alarm to '0'.
99
100
101
        process(coin1, coin2, coin5, buy, clock)
102
103
104
            if rising_edge(clock) then
                if Reset = '1' then
105
                     sum <= "000000";
106
                 elsif coin1 = '1' then
107
108
                     sum <= sum + 1;
109
                 elsif coin2 = '1' then
                     sum <= sum + 2;
110
                 elsif coin5 = '1' then
111
112
                     sum <= sum + 5;
                 elsif alarm = '1' then
113
                     alarm_count <= alarm_count_next;</pre>
114
                     if alarm_count(10) = '1' then
115
                                      <= '0';
116
                         alarm
117
                         alarm_count <= "00000000000";</pre>
                     end if;
118
                elsif buy = '1' then
119
                     if sum >= price then
120
121
                         sum <= sum - price;</pre>
122
                     elsif sum < price then</pre>
                         alarm <= '1';
123
                     end if;
124
                 end if;
125
126
                 sum_out
                           <= sum(5 downto 0);
                 price_out <= price(5 downto 0);</pre>
127
            end if;
128
129
            alarm_out <= alarm;</pre>
        end process;
130
131
132 end Behavioral;
```

Figure 15: Page 3

/Users/hbm/Dropbox/DTU-Amigos/0...Machine/src/vending_machine.vhd Page 1 of 4 Saved: 5/5/13 7:54:15 PM Printed For: Hjalte Bested Møller

```
-- Top component of the vending machine for the course:
2
3 -- 02139 Digital electronics 2 at the Technical University of Denmark
5
  -- This component declares and instantiates all the components of the
6
  -- vending machine.
7
8
9 library ieee;
use ieee.std_logic_1164.all;
11 use ieee.numeric_std.all;
12
13
  entity vending_machine is
      port(clk_50
                     : in std_logic;
14
            clk_man
                      : in std_logic;
15
16
            sel_man
                      : in std_logic;
17
            reset
                      : in std_logic;
           coin1_btn : in std_logic;
coin2_btn : in std_logic;
18
19
20
            coin5_btn : in std_logic;
21
            buy_btn
                      : in std_logic;
                     : in std_logic;
22
            cola_sw
            hash_sw
23
                      : in std_logic;
24
            aqua_sw
                      : in std_logic;
25
            an
                      : out std_logic_vector(3 downto 0);
26
            led
                      : out std_logic_vector(1 to 8));
27 end vending_machine;
28
  architecture struct of vending_machine is
29
30
      signal clk
                                                   : std_logic;
                                                   : std_logic;
: std_logic;
31
      signal clk_3
      signal buy
32
33
      signal coin1, coin2, coin5
                                                   : std_logic;
34
      signal price_cola, price_hash, price_aqua : std_logic;
      signal alarm, cola, aqua, hash
                                                 : std_logic;
35
                                                   : unsigned(5 downto 0);
      signal sum, price
36
37
38
39
      -- Component declarations
40
      component clock_manager is
41
           port(clk_50 : in std_logic;
42
43
                clk_man : in std_logic;
44
                sel_man : in std_logic;
                       : out std_logic;
45
                clk
                clk_3
46
                       : out std_logic);
47
      end component;
48
```

Figure 16: The complete vending machine, assembling all the components

/Users/hbm/Dropbox/DTU-Amigos/0...Machine/src/vending_machine.vhd Page 2 of 4 Saved: 5/5/13 7:54:15 PM Printed For: Hjalte Bested Møller

```
component display_driver is
49
            port(price : in unsigned(5 downto 0);
50
51
                  sum : in unsigned(5 downto 0);
                         : out std_logic_vector(3 downto 0);
52
                  reset : in std_logic;
clock : in std_logic;
53
54
55
                  clk_3 : in std_logic;
                  alarm : in std_logic;
56
                  cola : in std_logic;
57
58
                  hash : in std_logic;
59
                  aqua : in std_logic;
60
                  led
                        : out std_logic_vector(1 to 8));
61
        end component;
62
        component input_synchronizer is
63
                            : in std_logic;
: in std_logic;
64
            port(clock
65
                  buy_btn
                  buy_out : out std_logic;
coin1_btn : in std_logic;
66
67
68
                  coin1_out : out std_logic;
69
                  coin2_btn : in std_logic;
                  coin2_out : out std_logic;
70
                  coin5_btn : in std_logic;
coin5_out : out std_logic;
71
72
                  cola_sw : in std_logic;
cola_out : out std_logic;
73
74
                  hash_sw : in std_logic;
75
                  hash_out : out std_logic;
76
                              : in std_logic;
77
                  aqua_sw
78
                  aqua_out : out std_logic;
79
                  Reset
                              : in std_logic);
80
81
       end component;
82
83
        component processing_unit is
                          : in std_logic;
: in std_logic;
84
            port(clock
85
                  buy
86
                  coin1
                               : in std_logic;
87
                  coin2
                               : in std_logic;
                               : in std_logic;
88
                  coin5
                  price_cola : in std_logic;
price_hash : in std_logic;
89
90
91
                  price_aqua : in std_logic;
92
                             : in std_logic;
                  Reset
                              : out unsigned(5 downto 0);
                  sum_out
93
                  price_out : out unsigned(5 downto 0);
alarm_out : out std_logic;
94
95
                  cola_out : out std_logic;
```

Figure 17: Page 2

/Users/hbm/Dropbox/DTU-Amigos/0...Machine/src/vending_machine.vhd Page 3 of 4 Saved: 5/5/13 7:54:15 PM Printed For: Hjalte Bested Møller

```
hash_out : out std_logic;
 97
                  aqua_out : out std_logic);
 98
99
        end component;
100
101
    -- signal assignments
102
103
104
105
   begin
        clock_manager1 : clock_manager
    port map(clk_50 => clk_50,
106
107
                      clk_man => clk_man,
108
109
                      sel_man => sel_man,
                           => clk,
                      clk
110
                      clk_3 => clk_3);
111
112
113
        display_driver1 : display_driver
114
            port map(sum => sum,
                      price => price,
115
                          => an,
116
                      an
                      reset => reset,
117
                      clock => clk,
118
                      clk_3 => clk_3,
119
                      alarm => alarm,
120
121
                      cola => cola,
122
                      hash => hash,
                      aqua => aqua,
123
                      led
                            => led);
124
125
126
        input_synchronizer1 : input_synchronizer
127
            port map(clock
                                => clk,
                      buy_btn => buy_btn,
128
129
                      buy_out => buy,
130
                      coin1_btn => coin1_btn,
131
                      coin1_out => coin1,
                      coin2_btn => coin2_btn,
coin2_out => coin2,
132
133
134
                      coin5_btn => coin5_btn,
135
                      coin5_out => coin5,
                      cola_sw => cola_sw,
136
137
                      cola_out => price_cola,
                                => hash_sw,
                      hash_sw
138
139
                      hash_out => price_hash,
140
                      aqua_sw
                                 => aqua_sw,
                      aqua_out => price_aqua,
141
                      Reset
142
                                 => reset);
143
144
        processing_unit1 : processing_unit
```

Figure 18: Page 3

/Users/hbm/Dropbox/DTU-Amigos/0...Machine/src/vending_machine.vhd Page 4 of 4 Saved: 5/5/13 7:54:15 PM Printed For: Hjalte Bested Møller

```
=> clk,
           port map(clock
145
146
                     buy
                                 => buy,
                     coin1
147
                                 => coin1,
148
                     coin2
                                 => coin2,
                                => coin5,
                     coin5
149
                     price_cola => price_cola,
150
                     price_hash => price_hash,
151
152
                     price_aqua => price_aqua,
                     Reset
                                => reset,
153
                                => sum,
                     sum_out
154
                     price_out => price,
155
156
                     alarm_out => alarm,
157
                     cola_out
                                 => cola,
                     hash_out
                                => hash,
158
                     aqua_out
                                => aqua);
159
160
161 end struct;
162
```

Figure 19: Page 4