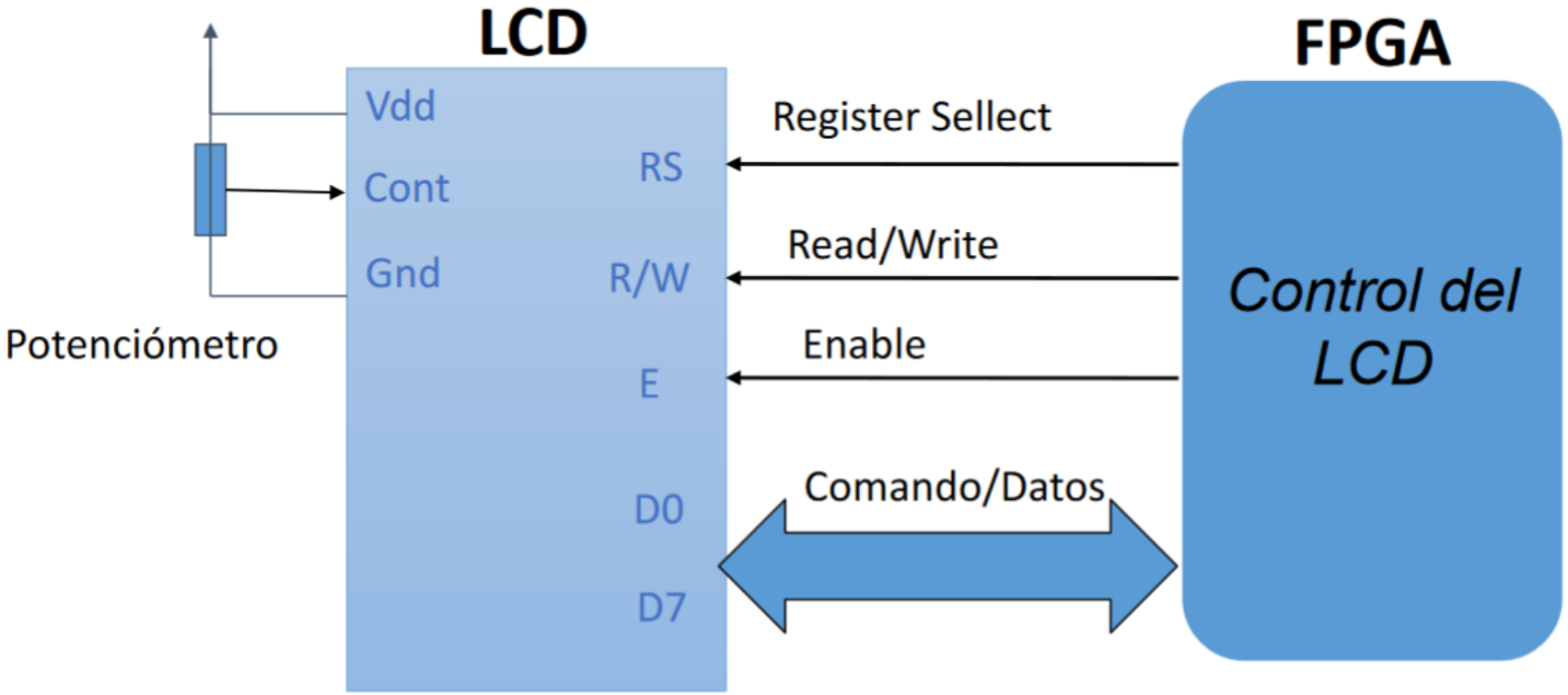
Controlador LCD

J. Emilio Soriano Campos A00829390

Este proyecto consistió en construir un controlador en VHDL para un display de cristal liquido de cuarzo de 16x2.



Contador de 12 bits

Para este proyecto solo fueron necesarios dos componentes : un contador de 12 bits y una ma2quina de estados. El contador lo diseñe de tal manera que al momento de llegar al ultimo numero posible por el contador este emitiría una señal de overflow el cual activaría la maquina de estados.

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3
4  entity CNT12 is
5     port (clk, rst : in std_logic;
6           ov : out std_logic;
7           count : out std_logic_vector(11 downto 0));
8  end entity;
9
10 architecture CNT12_ARC of CNT12 is
11    component masuno12 is
12    port (A : in std_logic_vector(11 downto 0);
13          ov : out std_logic;
14          Z : out std_logic_vector(11 downto 0));
15    end component;
16
17    signal D, Q : std_logic_vector(11 downto 0);
18    signal O : std_logic;
19
20    begin
21    I0 : masuno12 port map (Q, O, D);
22    count <= Q;
23
24    P1 : process(clk, rst)
25       begin
26       if rst = '0' then
27          Q <= "000000000000";
28       elsif clk'event and clk = '1' then
29          Q <= D;
30          ov <= O;
31       end if;
32    end process;
33 end architecture;
```

Maquina de estados

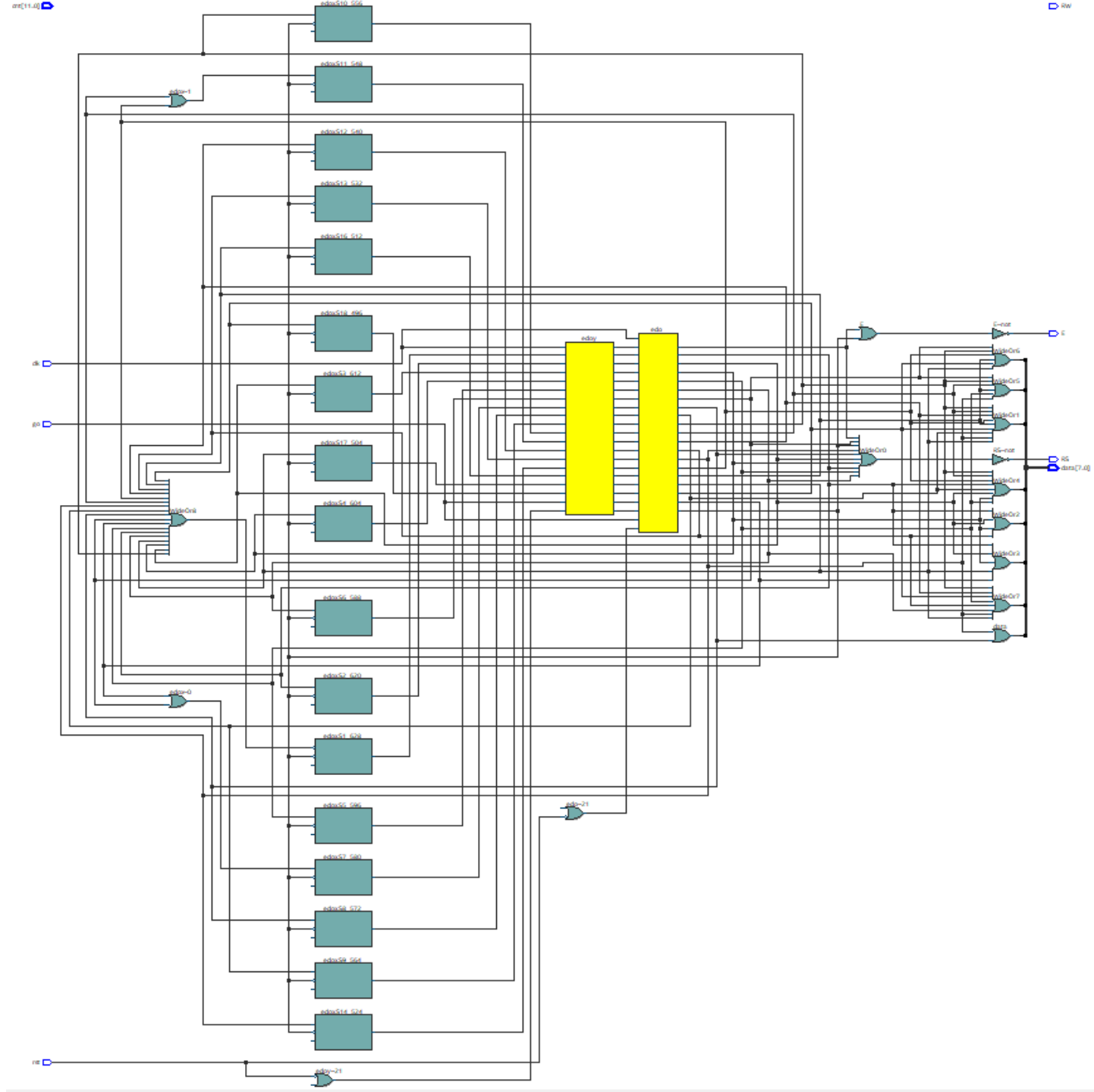
Para la maquina de estados fueron necesarios 20 estados diferentes para poder emitir el mensaje “Hola! JESC”. Fueron 7 estados para configurar el LCD, 10 estados para definir los caracteres que se querían mostrar, uno para volver a iniciar la maquina y un estado mas que iría entre cada cambio de estados.

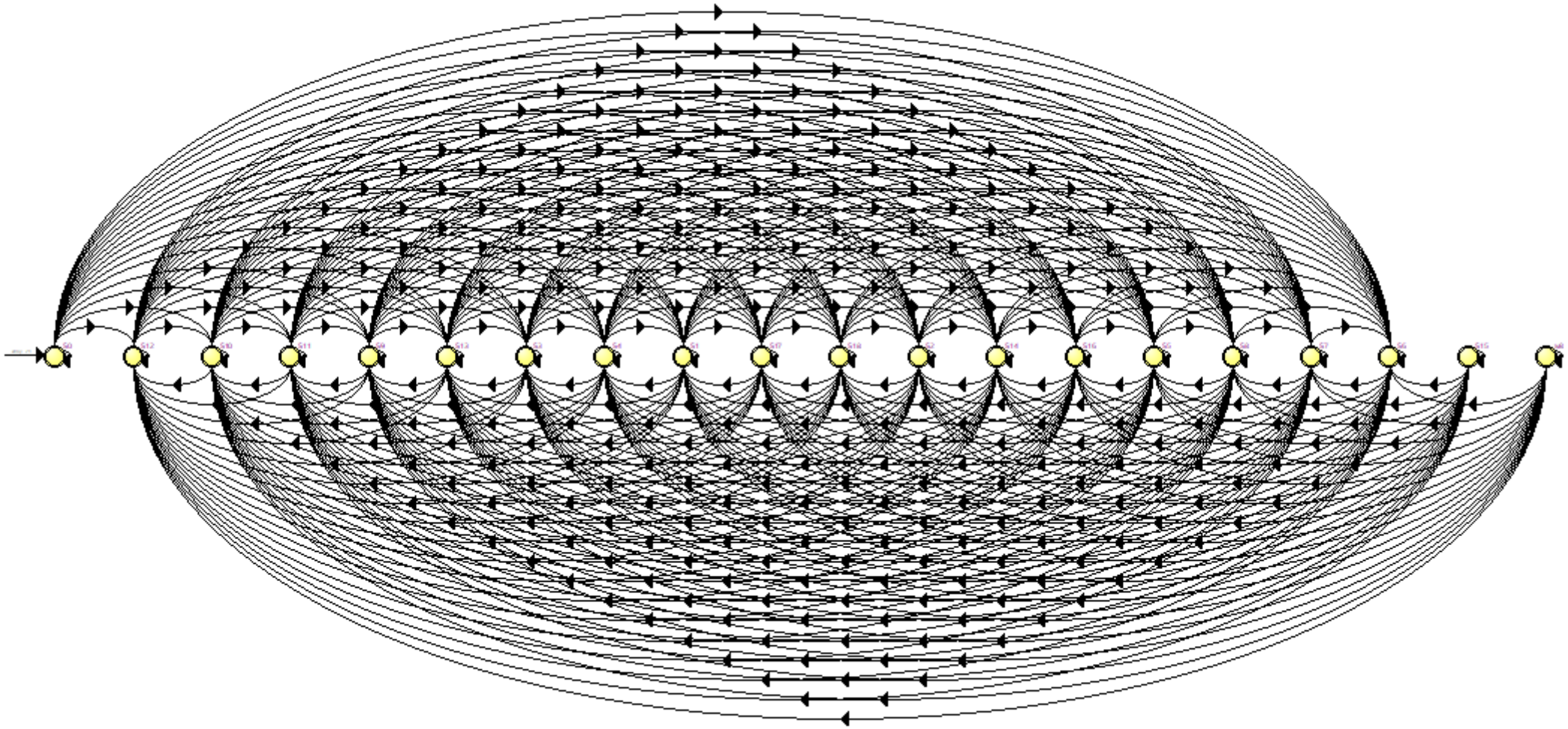
```
1   library ieee;
2   use ieee.std_logic_1164.all;
3
4   entity LCD is
5      port (clk, rst, go : in std_logic;
6            cnt : in std_logic_vector(11 downto 0);
7            RS, RW, E : out std_logic;
8            data : out std_logic_vector(7 downto 0));
9   end entity;
10
11  architecture LCD_ARC of LCD is
12
13     type edos is (S0, S1, S2, S3, S4, S5, S6, S7, S8, S9, S10, S11, S12, S13, S14, S15, S16, S17, S18, w8);
14     signal edo, edof, edox, edoy : edos := S0;
15
16     begin
17
18     P1 : process (clk, rst, go)
19        begin
20        if rst = '0' then
21           edo <= S0;
22           edoy <= S0;
23        elsif clk'event and clk = '1' then
24           if go = '1' then
25              edo <= edof;
26              edoy <= edox;
27           end if;
28        end if;
29     end process;
```

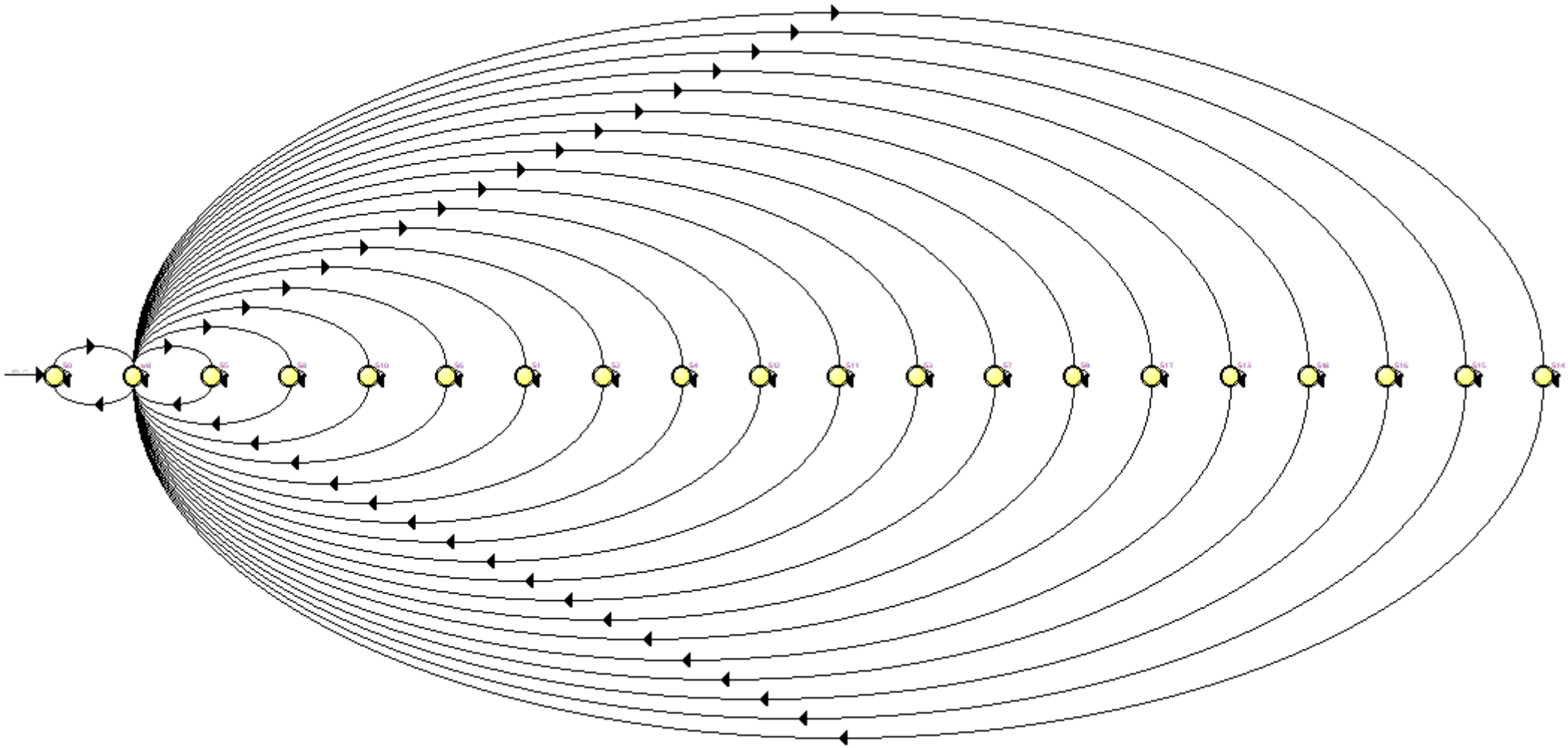
```
31  P2 : process (clk, edo)
32     begin
33     case edo is
34        when S0 => RS <= '0';
35                   RW <= '0';
36                   E <= '0';
37                   data <= X"00";
38                   edox <= S1;
39                   edof <= w8;
40
41        when S1 => RS <= '0';
42                   RW <= '0';
43                   E <= '1';
44                   data <= X"38";
45                   edox <= S2;
46                   edof <= w8;
47
48        when S2 => RS <= '0';
49                   RW <= '0';
50                   E <= '1';
51                   data <= X"38";
52                   edox <= S3;
53                   edof <= w8;
54
55        when S3 => RS <= '0';
56                   RW <= '0';
57                   E <= '1';
58                   data <= X"38";
59                   edox <= S4;
60                   edof <= w8;
61
62        when S4 => RS <= '0';
63                   RW <= '0';
64                   E <= '1';
65                   data <= X"0D";
66                   edox <= S5;
67                   edof <= w8;
68
69        when S5 => RS <= '0';
70                   RW <= '0';
71                   E <= '1';
72                   data <= X"01";
73                   edox <= S6;
74                   edof <= w8;
75
76        when S6 => RS <= '0';
77                   RW <= '0';
78                   E <= '1';
79                   data <= X"06";
80                   edox <= S7;
81                   edof <= w8;
82
83        when S7 => RS <= '0';
84                   RW <= '0';
85                   E <= '1';
86                   data <= X"80";
87                   edox <= S8;
88                   edof <= w8;
```

```
90         when S8 => RS <= '1';
91                    RW <= '0';
92                    E <= '1';
93                    data <= X"48";
94                    edox <= S9;
95                    edof <= w8;
96
97         when S9 => RS <= '1';
98                    RW <= '0';
99                    E <= '1';
100                   data <= X"4F";
101                   edox <= S10;
102                   edof <= w8;
103
104        when S10 => RS <= '1';
105                   RW <= '0';
106                   E <= '1';
107                   data <= X"4C";
108                   edox <= S11;
109                   edof <= w8;
110
111        when S11 => RS <= '1';
112                   RW <= '0';
113                   E <= '1';
114                   data <= X"41";
115                   edox <= S12;
116                   edof <= w8;
117
118        when S12 => RS <= '1';
119                   RW <= '0';
120                   E <= '1';
121                   data <= X"21";
122                   edox <= S13;
123                   edof <= w8;
124
125        when S13 => RS <= '0';
126                   RW <= '0';
127                   E <= '1';
128                   data <= X"C5";
129                   edox <= S14;
130                   edof <= w8;
131
132        when S14 => RS <= '1';
133                   RW <= '0';
134                   E <= '1';
135                   data <= X"4A";
136                   edox <= S11;
137                   edof <= w8;
138
139        when S15 => RS <= '1';
140                   RW <= '0';
141                   E <= '1';
142                   data <= X"46";
143                   edox <= S16;
144                   edof <= w8;
145
146        when S16 => RS <= '1';
147                   RW <= '0';
148                   E <= '1';
149                   data <= X"53";
150                   edox <= S17;
151                   edof <= w8;
```

```
152
153        when S17 => RS <= '1';
154                    RW <= '0';
155                    E <= '1';
156                    data <= X"43";
157                    edox <= S18;
158                    edof <= w8;
159
160        when S18 => RS <= '1';
161                    RW <= '0';
162                    E <= '1';
163                    data <= X"10";
164                    edox <= S7;
165                    edof <= w8;
166
167        when w8 => RS <= '0';
168                   RW <= '0';
169                   E <= '0';
170                   data <= X"00";
171                   edof <= edoy;
172
173        when others => null;
174     end case;
175  end process;
176 end architecture;
```


cnt[11..0]
RW
clk
go
rst
E
RS
data[7..0]

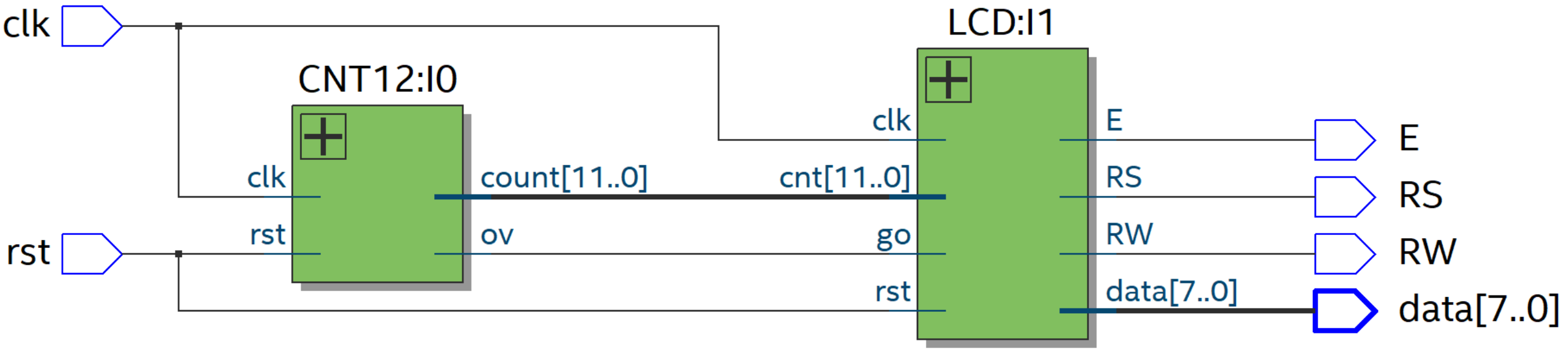




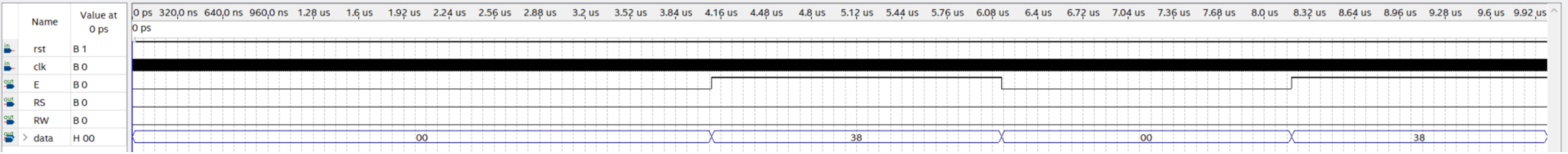
Top-level design

Una vez con la maquina de estados y el contador de 12 bits solo fue cuestión de conectar ambos componentes en un simple top-level design

```
1   library ieee;
2   use ieee.std_logic_1164.all;
3
4   entity LCD_toplvl is
5      port (clk, rst : in std_logic;
6            RS, RW, E : out std_logic;
7            data : out std_logic_vector(7 downto 0));
8   end entity;
9
10  architecture LCD_toplvl_ARC of LCD_toplvl is
11     component CNT12 is
12     port (clk, rst : in std_logic;
13           ov : out std_logic;
14           count : out std_logic_vector(11 downto 0));
15     end component;
16     component LCD is
17     port (clk, rst, go : in std_logic;
18           cnt : in std_logic_vector(11 downto 0);
19           RS, RW, E : out std_logic;
20           data : out std_logic_vector(7 downto 0));
21     end component;
22
23     signal cont : std_logic_vector(11 downto 0);
24     signal flag : std_logic;
25
26     begin
27     I0 : CNT12 port map (clk, rst, flag, cont);
28     I1 : LCD port map (clk, rst, flag, cont, RS, RW, E, data);
29  end architecture;
```


clk
rst
CNT12:I0
clk
rst
count[11..0]
ov
LCD:I1
clk
cnt[11..0]
go
rst
E
RS
RW
data[7..0]
E
RS
RW
data[7..0]

Simulación


Name
Value at 0 ps
0 ps
320,0 ns
640,0 ns
960,0 ns
1.28 us
1.6 us
1.92 us
2.24 us
2.56 us
2.88 us
3.2 us
3.52 us
3.84 us
4.16 us
4.48 us
4.8 us
5.12 us
5.44 us
5.76 us
6.08 us
6.4 us
6.72 us
7.04 us
7.36 us
7.68 us
8.0 us
8.32 us
8.64 us
8.96 us
9.28 us
9.6 us
9.92 us
rst
B 1
clk
B 0
E
B 0
RS
B 0
RW
B 0
data
H 00
00
38
00
38