Reporte de practica 12

González Pardo Adrian

Mayo 2020

1. Código fuente:

1.1. Unidad de Control

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity unidadControl is
    port( clk,clr,ini,z,a0:in std_logic;
          la,lb,ea,eb,ec:out std_logic
    );
end unidadControl;

architecture Behavioral of unidadControl is
    type estados is (e0,e1,e2);
    signal edoA,edoS:estados;
begin
    process(clk,clr)
    begin
        if(clr='1') then
            edoA<=e0;
        elsif rising_edge(clk) then
            edoA<=edoS;
        end if;
    end process;
    process(edoA,ini,z,a0)
    begin
        la<='0';
        ea<='0';
        lb<='0';
        eb<='0';
        ec<='0';
        case edoA is
            when e0 =>
                lb <= '1';
                if(ini='0') then
                    la<='1';
                    edoS<=e0;
                else
                    edoS<=e1;
                end if;
            when e1 =>
                ea <= '1';
                if(z='0') then
                    if(a0='0') then
                        edoS<=e1;
                    else
                        eb<='1';
                        edoS<=e1;
                    end if;
                else
                    edoS<=e2;
                end if;
            when e2=>
                ec <='1';
                if(ini='0') then
                    edoS<=e0;
                else
                    edoS<=e2;
                end if;
        end case;
    end process;
end Behavioral;
```

1.2. Registro

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity registro is
    port( la,ea,clk,clr:in std_logic;
          d: in std_logic_vector(8 downto 0);
          z,a0:out std_logic;
          qa:out std_logic_vector(8 downto 0)
    );
end registro;

architecture Behavioral of registro is
    signal a:std_logic_vector(8 downto 0);
begin
    process(clk,clr,a)
    begin
        if(clr='1') then
            a<=(others =>'0');
        elsif rising_edge(clk) then
            if(la='1' and ea='0') then
                a<=d;
            end if;
            if(la='0' and ea='1') then
                a<=to_stdlogicvector(to_bitvector(a) srl 1);
            end if;
        end if;
        qa<=a;
        a0<=a(0);
        z<=not(a(0) or a(1) or a(2) or a(3) or a(4) or a(5) or a(6) or a(7) or a(8));
    end process;

end Behavioral;
```

1.3. Contador

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity contador is
    port( lb,eb,clr,clk:in std_logic;
          qb:out std_logic_vector(3 downto 0));
end contador;

architecture Behavioral of contador is
    signal b:std_logic_vector(3 downto 0);
begin
    process(clk,clr)
    begin
        if(clr='1') then
            b<=(others=>'0');
        elsif rising_edge(clk) then
```

```vhdl
            if(lb='1' and eb='0') then
                b<=(others=>'0');
            end if;
            if(lb='0' and eb='1') then
                b<=b+1;
            end if;
        end if;
        qb<=b;
    end process;
end Behavioral;
```

1.4. Decodificador

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity decodificador is
    port( qb:in std_logic_vector(3 downto 0);
          di:out std_logic_vector(6 downto 0));
end decodificador;

architecture Behavioral of decodificador is
    -- Constantes para 7 segmentos de anodo comun
    constant n0:std_logic_vector(6 downto 0):="0000001";
    constant n1:std_logic_vector(6 downto 0):="1001111";
    constant n2:std_logic_vector(6 downto 0):="0010010";
    constant n3:std_logic_vector(6 downto 0):="0000110";
    constant n4:std_logic_vector(6 downto 0):="1001100";
    constant n5:std_logic_vector(6 downto 0):="0100100";
    constant n6:std_logic_vector(6 downto 0):="0100000";
    constant n7:std_logic_vector(6 downto 0):="0001111";
    constant n8:std_logic_vector(6 downto 0):="0000000";
    constant n9:std_logic_vector(6 downto 0):="0000100";
    constant nx:std_logic_vector(6 downto 0):="1111110";
begin
    with qb select
        di<=n0 when "0000",
           n1 when "0001",
           n2 when "0010",
           n3 when "0011",
           n4 when "0100",
           n5 when "0101",
           n6 when "0110",
           n7 when "0111",
           n8 when "1000",
           n9 when "1001",
           nx when others;



end Behavioral;
```

1.5. Mux

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity muxD is
    port( di: in std_logic_vector(6 downto 0);
          ec: in std_logic;
          disp:out std_logic_vector(6 downto 0));
end muxD;

architecture Behavioral of muxD is
    constant nx:std_logic_vector(6 downto 0):="1111110";
begin
    with ec select disp <=nx when '0', di when others;
```

```

end Behavioral;
```

1.6. Arquitectura completa

```
library IEEE;
library WORK;
use IEEE.STD_LOGIC_1164.ALL;
use WORK.paqueteEntidades.ALL;

entity cartaASM is
    port( ini,clr,clk:in std_logic;
          d:in std_logic_vector(8 downto 0);
          disp:out std_logic_vector(6 downto 0);
          qa:out std_logic_vector(8 downto 0));
end cartaASM;

architecture Behavioral of cartaASM is
    signal la,ea,a0,z,lb,eb,ec:std_logic;
    signal qb:std_logic_vector(3 downto 0);
    signal di:std_logic_vector(6 downto 0);
begin
    elementoUnidadControl:unidadControl port map(
        clk=>clk,
        clr=>clr,
        ini=>ini,
        a0=>a0,
        z=>z,
        la=>la,
        ea=>ea,
        eb=>eb,
        lb=>lb,
        ec=>ec
    );
    elementoRegistro:registro port map(
        clk=>clk,
        clr=>clr,
        la=>la,
        ea=>ea,
        d=>d,
        z=>z,
        a0=>a0,
        qa=>qa
    );

    elementoContador:contador port map(
        lb=>lb,
        eb=>eb,
        clr=>clr,
        clk=>clk,
        qb=>qb
    );

    elementoDecodificador:decodificador port map(
        qb=>qb,
        di=>di
    );

    elementoMux:muxD port map(
        di=>di,
        ec=>ec,
        disp=>disp
    );

end Behavioral;
```

2. Codigo de empaquetado

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

package paqueteEntidades is
    component muxD is
    port( di: in std_logic_vector(6 downto 0);
          ec: in std_logic;
          disp:out std_logic_vector(6 downto 0));
    end component;

    component unidadControl is
    port( clk,clr,ini,z,a0:in std_logic;
          la,lb,ea,eb,ec:out std_logic
    );
    end component;

    component decodificador is
    port( qb:in std_logic_vector(3 downto 0);
          di:out std_logic_vector(6 downto 0));
    end component;

    component contador is
    port( lb,eb,clr,clk:in std_logic;
          qb:out std_logic_vector(3 downto 0));
    end component;

    component registro is
    port( la,ea,clk,clr:in std_logic;
          d: in std_logic_vector(8 downto 0);
          z,a0:out std_logic;
          qa:out std_logic_vector(8 downto 0)
    );
    end component;
end package;
```

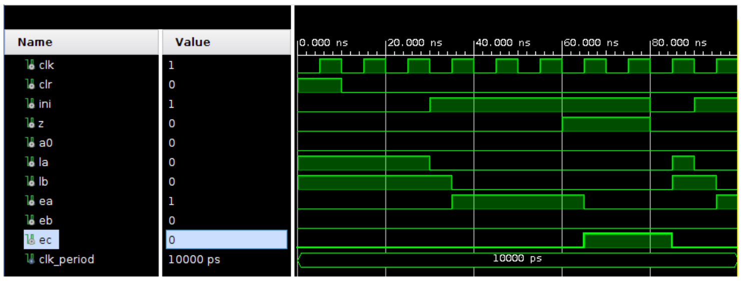
3. Test-Bench:

3.1. Unidad de Control

3.1.1. Codigo

```vhdl
library IEEE;
library work;
use IEEE.STD_LOGIC_1164.ALL;
use work.paqueteentidades.unidadControl;
entity unidadControl_TB is
end unidadControl_TB;

architecture Behavioral of unidadControl_TB is
    signal clk,clr,ini,z,a0:std_logic:='0';
    signal la,lb,ea,eb,ec:std_logic:='0';
    constant clk_period: time :=10ns;
begin
    uc:unidadControl port map(
        clk=>clk,
        clr=>clr,
        ini=>ini,
        z=>z,
        a0=>a0,
        la=>la,
        lb=>lb,
        ea=>ea,
        eb=>eb,
        ec=>ec
    );
    clkProcess:process
    begin
        clk<='0';
        wait for clk_period/2;
        clk<='1';
        wait for clk_period/2;
    end process;

    ucTB:process
    begin
        clr<='1';
        wait for clk_period;
        clr<='0';
        wait for clk_period;
        --Aqui se habilita la
        wait for clk_period;
        ini<='1';
        --Pasa a e1
        wait for clk_period;
        z<='0';
        --Sigue en e1 y eb esta en 0
        wait for clk_period;
        a0<='0';
        --Sigue en e1 y eb pasa a 1
        wait for clk_period;
        z<='1';
        --Pasa a e2
        wait for clk_period;
        --Sigue en e2 porque ini esta en 1
        wait for clk_period;
        ini<='0';
        z<='0';
        a0<='0';
        --Pasa a e0
        wait for clk_period;
        ini<='1';
        --pasa a e1
        wait for clk_period;
        clr<='1';
        wait;
    end process;

end Behavioral;
```

3.1.2. Imagenes



Name	Value
clk	1
clr	0
ini	1
z	0
a0	0
la	0
lb	0
ea	1
eb	0
ec	0
clk_period	10000 ps

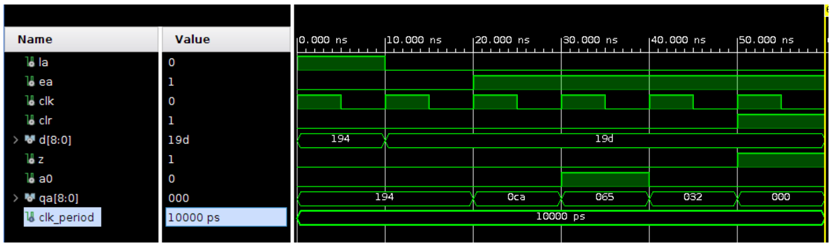
3.2. Registro

3.2.1. Codigo

```
library IEEE;
library work;
use IEEE.STD_LOGIC_1164.ALL;
use work.paqueteentidades.registro;
```

```vhdl
entity registro_TB is
end registro_TB;

architecture Behavioral of registro_TB is
    signal la,ea,clk,clr:std_logic:='0';
    signal d:std_logic_vector(8 downto 0):="000000000";
    signal z,a0:std_logic:='0';
    signal qa:std_logic_vector(8 downto 0):="000000000";
    constant clk_period:time:=10ns;
begin
    reg:registro port map(
        la=>la,
        ea=>ea,
        clk=>clk,
        clr=>clr,
        d=>d,
        z=>z,
        a0=>a0,
        qa=>qa
    );
    clkP:process
    begin
        clk<='1';
        wait for clk_period/2;
        clk<='0';
        wait for clk_period/2;
    end process;

    regTB:process
    begin
        la<='1';
        d<="110010100";
        wait for clk_period;
        d<="110011101";
        la<='0';
        wait for clk_period;
        ea<='1';
        wait for clk_period*3;
        clr<='1';
        wait for clk_period;
        wait;
    end process;

end Behavioral;
```

3.2.2. Imagenes



```
Name        Value
la          0
ea          1
clk         0
clr         1
d[8:0]      19d
z           1
a0          0
qa[8:0]     000
clk_period  10000 ps
```

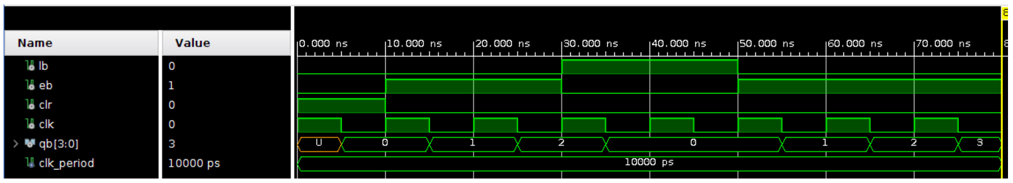
3.3. Contador

3.3.1. Codigo

```vhdl
library IEEE;
library work;
use IEEE.STD_LOGIC_1164.ALL;
use work.paqueteentidades.contador;

```

```
entity contador_TB is
end contador_TB;

architecture Behavioral of contador_TB is
    signal lb,eb,clr,clk:std_logic:='0';
    signal qb:std_logic_vector(3 downto 0):="0000";
    constant clk_period:time:=10ns;
begin
    cont:contador port map(
        lb=>lb,
        eb=>eb,
        clr=>clr,
        clk=>clk,
        qb=>qb
    );
    clkP:process
    begin
        clk<='1';
        wait for clk_period/2;
        clk<='0';
        wait for clk_period/2;
    end process;
    tb:process
    begin
        clr<='1';
        wait for clk_period;
        clr<='0';
        eb<='1';
        wait for clk_period*2;
        eb<='0';
        lb<='1';
        wait for clk_period*2;
        lb<='0';
        eb<='1';
        wait for clk_period*3;
        clr<='1';
        wait;
    end process;

end Behavioral;
```

3.3.2. Imagenes



Name	Value
lb	0
eb	1
clr	0
clk	0
qb[3:0]	3
clk_period	10000 ps

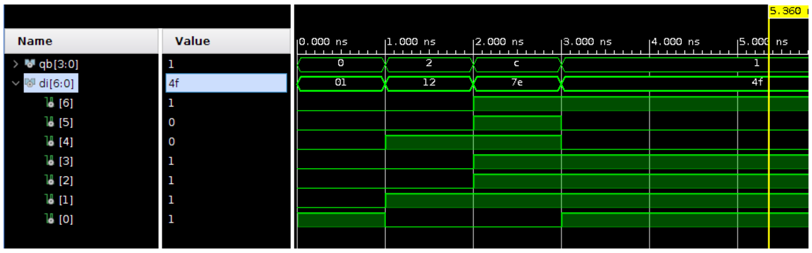
3.4. Decodificador

3.4.1. Codigo

```
library IEEE;
library work;
use IEEE.STD_LOGIC_1164.ALL;
use work.paqueteEntidades.decodificador;

entity decodificador_TB is
end decodificador_TB;

architecture Behavioral of decodificador_TB is
    signal qb:std_logic_vector(3 downto 0);
    signal di:std_logic_vector(6 downto 0);
begin
```

```vhdl
    deco:decodificador port map(
        qb=>qb,
        di=>di
    );
    decoTB:process
    begin
        qb<="0000";
        wait for 1ns;
        qb<="0010";
        wait for 1ns;
        qb<="1100";
        wait for 1ns;
        qb<="0001";
        wait;
    end process;
end Behavioral;
```

3.4.2. Imagenes



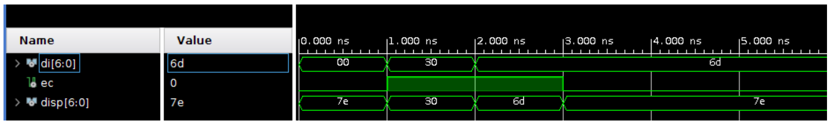
Name	Value
qb[3:0]	1
di[6:0]	4f
[6]	1
[5]	0
[4]	0
[3]	1
[2]	1
[1]	1
[0]	1

3.5. Mux

3.5.1. Codigo

```vhdl
library IEEE;
library work;
use IEEE.STD_LOGIC_1164.ALL;
use work.paqueteentidades.muxD;

entity muxD_TB is
end muxD_TB;

architecture Behavioral of muxD_TB is
    signal di:std_logic_vector(6 downto 0):="0000000";
    signal ec:std_logic:='0';
    signal disp:std_logic_vector(6 downto 0):="0000000";
begin
    mux:muxD port map(
        di=>di,
        ec=>ec,
        disp=>disp
    );
    muxTB:process
    begin
        ec<='0';
        wait for 1ns;
        ec<='1';
        di<="0110000";
        wait for 1ns;
        di<="1101101";
        wait for 1ns;
        ec<='0';
        wait;

    end process;

end Behavioral;
```

3.5.2. Imagenes



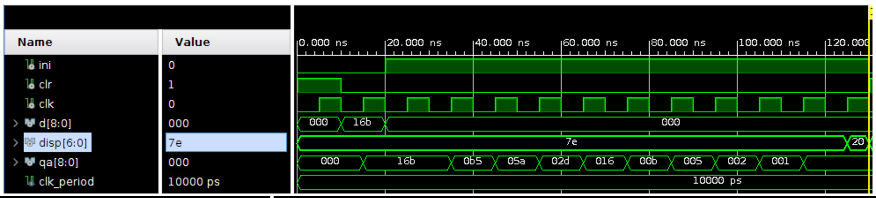
```
Name        Value
di[6:0]     6d
ec          0
disp[6:0]   7e
```

3.6. Arquitectura completa

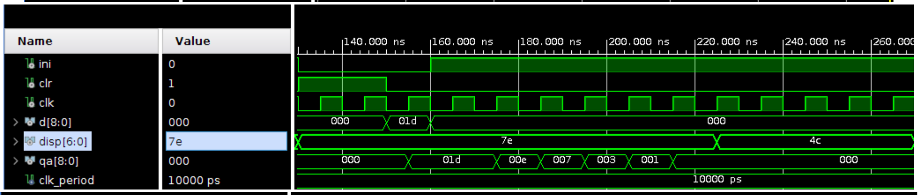
3.6.1. Codigo

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity cartaASM_TB is
end cartaASM_TB;

architecture Behavioral of cartaASM_TB is
    component cartaASM is
        port( ini,clr,clk:in std_logic;
              d:in std_logic_vector(8 downto 0);
              disp:out std_logic_vector(6 downto 0);
              qa:out std_logic_vector(8 downto 0));
    end component;
    signal ini,clr,clk:std_logic:='0';
    signal d:std_logic_vector(8 downto 0):="000000000";
    signal disp:std_logic_vector(6 downto 0):="0000000";
    signal qa:std_logic_vector(8 downto 0):="000000000";
    constant clk_period: time :=10ns;
begin
    asm:cartaASM port map(
        ini=>ini,
        clr=>clr,
        clk=>clk,
        d=>d,
        disp=>disp,
        qa=>qa
    );

    clkProcess:process
    begin
        clk<='0';
        wait for clk_period/2;
        clk<='1';
        wait for clk_period/2;
    end process;

    tb:process
    begin
        --a
        clr<='1';

        wait for clk_period;
        clr<='0';
        d<="101101011";
        wait for clk_period;
        ini<='1';
        d<="000000000";
        wait for clk_period*11;
        ini<='0';
        clr<='1';
        --b
        wait for clk_period*2;
        clr<='0';
        d<="000011101";
        wait for clk_period;
        ini<='1';
        d<="000000000";
        wait for clk_period*11;
        ini<='0';
        clr<='1';
        --c
        wait for clk_period*2;
        clr<='0';
        d<="000010000";
        wait for clk_period;
        ini<='1';
        d<="000000000";
        wait for clk_period*11;
        ini<='0';
        clr<='1';
        --d
        wait for clk_period*2;
        clr<='0';
        d<="100001000";
        wait for clk_period;
        ini<='1';
        d<="000000000";
        wait for clk_period*11;
        ini<='0';
        clr<='1';
        --e
        wait for clk_period*2;
        clr<='0';
        d<="000000000";
        wait for clk_period;
        ini<='1';
        d<="000000000";
        wait for clk_period*11;
        ini<='0';
        clr<='1';
        wait;
    end process;

end Behavioral;
```

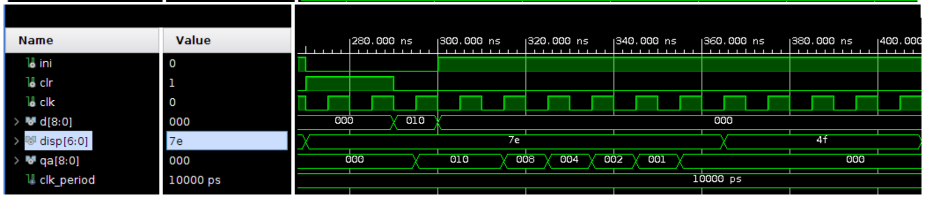
3.6.2. Imagenes



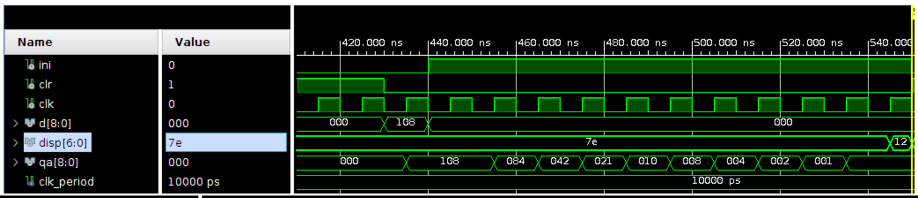
Name	Value
ini	0
clr	1
clk	0
d[8:0]	000
disp[6:0]	7e
qa[8:0]	000
clk_period	10000 ps



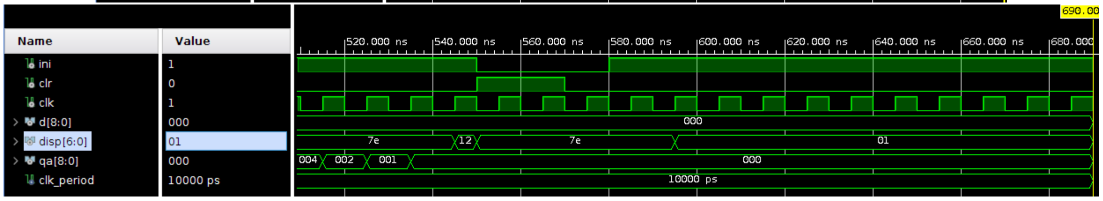
Name	Value
ini	0
clr	1
clk	0
d[8:0]	000
disp[6:0]	7e
qa[8:0]	000
clk_period	10000 ps



Name	Value
ini	0
clr	1
clk	0
d[8:0]	000
disp[6:0]	7e
qa[8:0]	000
clk_period	10000 ps



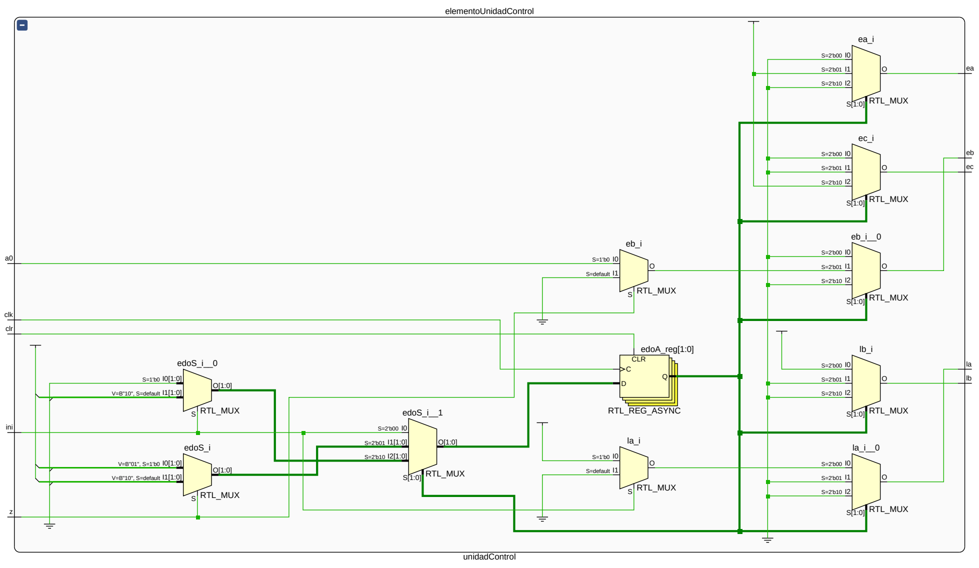
Name	Value
ini	0
clr	1
clk	0
d[8:0]	000
disp[6:0]	7e
qa[8:0]	000
clk_period	10000 ps



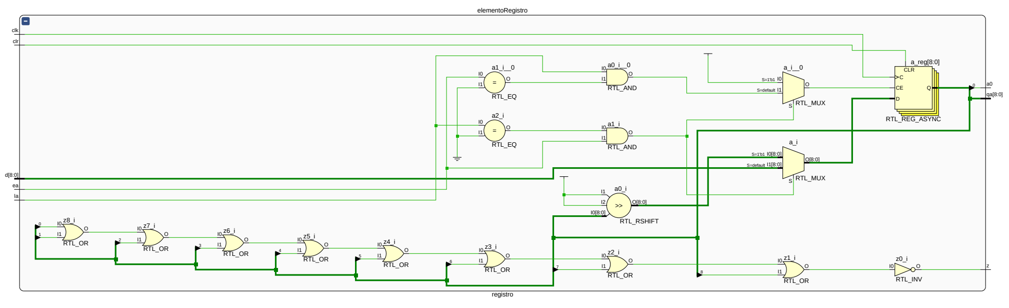
Name	Value
ini	1
clr	0
clk	1
d[8:0]	000
disp[6:0]	01
qa[8:0]	000
clk_period	10000 ps

4. Diagrama RTL

4.1. Unidad de Control

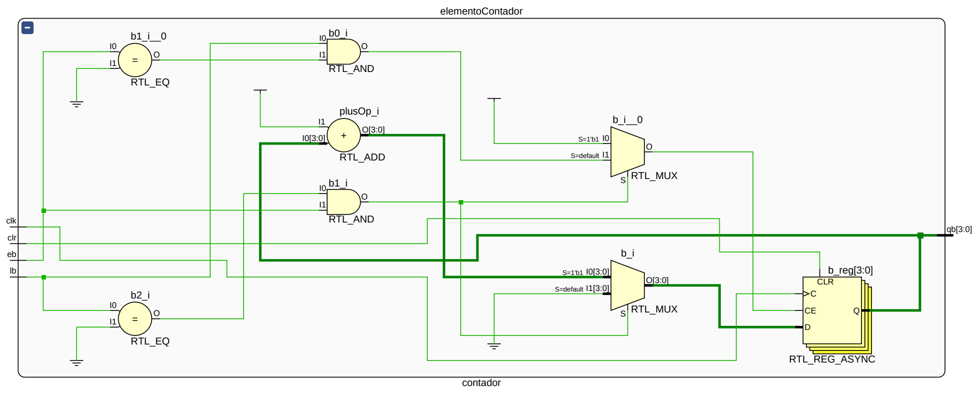


4.2. Registro

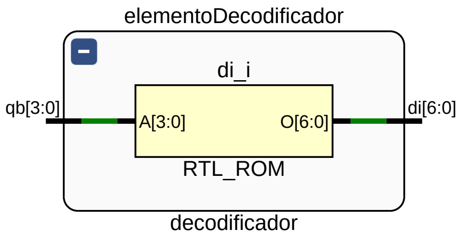


elementoRegistro
clk
clr
d[8:0]
ea
la
a1_i__0
RTL_EQ
a2_i
RTL_EQ
a0_i__0
RTL_AND
a1_i
RTL_AND
a_i__0
RTL_MUX
a_i
RTL_MUX
a_reg[8:0]
RTL_REG_ASYNC
a0_i
RTL_RSHIFT
z8_i
z7_i
z6_i
z5_i
z4_i
z3_i
z2_i
z1_i
RTL_OR
z0_i
RTL_INV
a0
qa[8:0]
z
registro

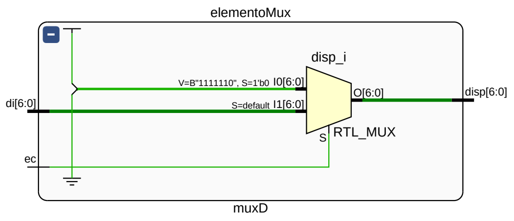
4.3. Contador


elementoContador
b1_i__0
RTL_EQ
b0_i
RTL_AND
plusOp_i
RTL_ADD
b1_i
RTL_AND
b_i__0
RTL_MUX
b_i
RTL_MUX
b_reg[3:0]
RTL_REG_ASYNC
b2_i
RTL_EQ
clk
clr
eb
lb
qb[3:0]
contador

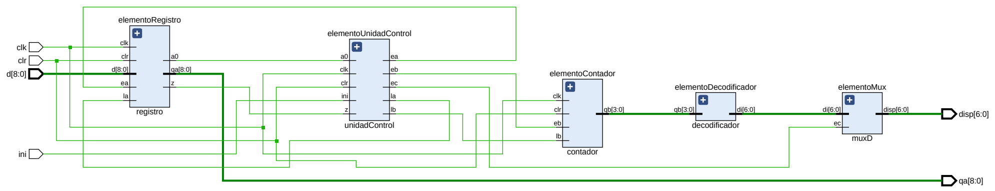
4.4. Decodificador



4.5. Mux


elementoMux
disp_i
V=B"1111110", S=1'b0 I0[6:0]
S=default I1[6:0]
O[6:0]
di[6:0]
disp[6:0]
S
RTL_MUX
ec
muxD

4.6. Arquitectura Completa


clk
clr
d[8:0]
ini
elementoRegistro
clk
clr
d[8:0]
ea
la
a0
qa[8:0]
z
registro
elementoUnidadControl
a0
clk
clr
ini
z
ea
eb
ec
la
lb
unidadControl
elementoContador
clk
clr
eb
lb
qb[3:0]
contador
elementoDecodificador
qb[3:0]
di[6:0]
decodificador
elementoMux
di[6:0]
ec
disp[6:0]
muxD
disp[6:0]
qa[8:0]