INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

U.A: Arquitectura de Computadoras

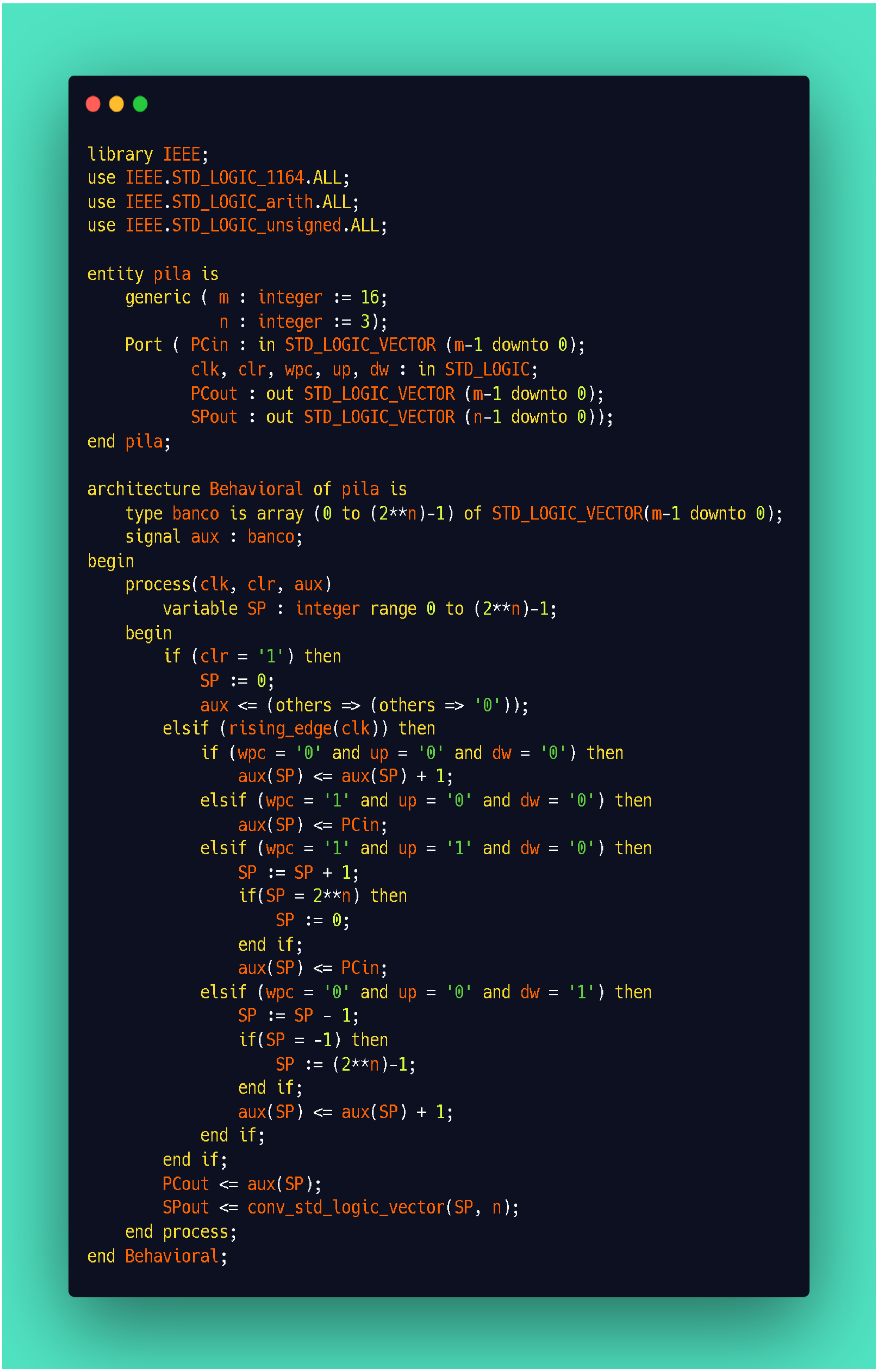
"Práctica 9: Pila Hardware"

Grupo: 3CV11

Profesor: Vega García Nayeli

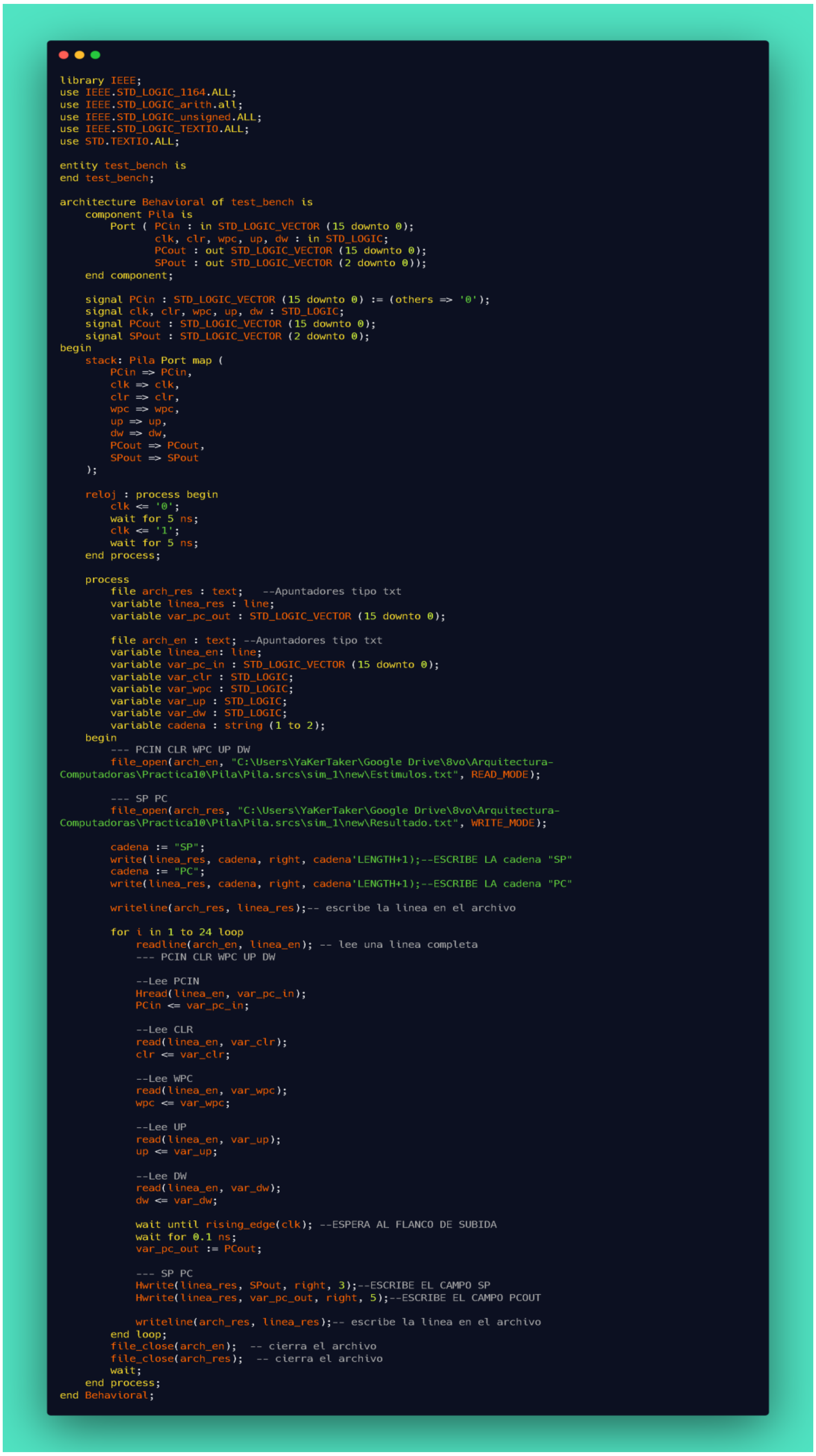
Sánchez Becerra Ernesto Daniel

Código de implementación



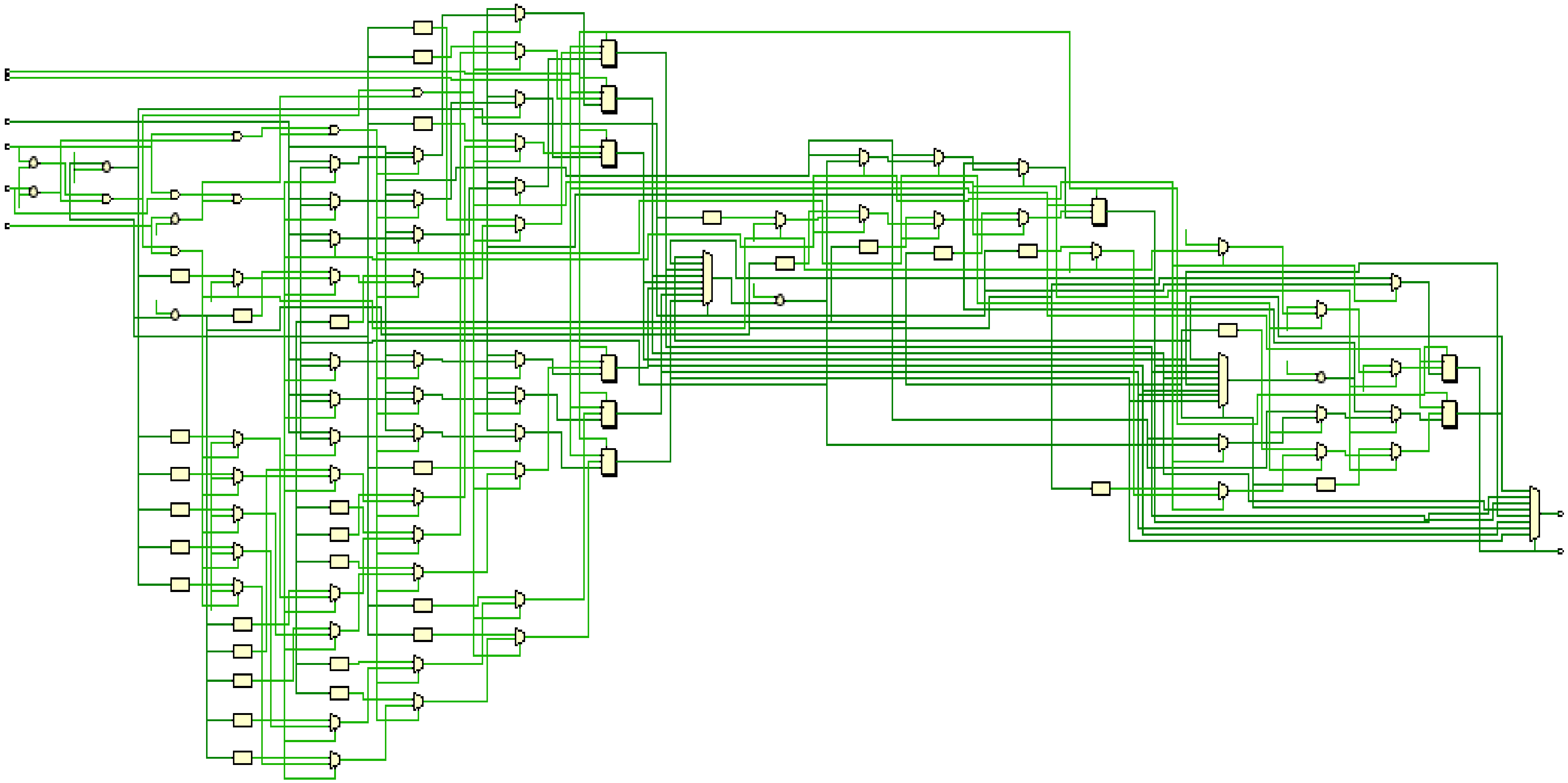
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_arith.ALL;
use IEEE.STD_LOGIC_unsigned.ALL;

entity pila is
    generic ( m : integer := 16;
              n : integer := 3);
    Port ( PCin : in STD_LOGIC_VECTOR (m-1 downto 0);
           clk, clr, wpc, up, dw : in STD_LOGIC;
           PCout : out STD_LOGIC_VECTOR (m-1 downto 0);
           SPout : out STD_LOGIC_VECTOR (n-1 downto 0));
end pila;

architecture Behavioral of pila is
    type banco is array (0 to (2**n)-1) of STD_LOGIC_VECTOR(m-1 downto 0);
    signal aux : banco;
begin
    process(clk, clr, aux)
        variable SP : integer range 0 to (2**n)-1;
    begin
        if (clr = '1') then
            SP := 0;
            aux <= (others => (others => '0'));
        elsif (rising_edge(clk)) then
            if (wpc = '0' and up = '0' and dw = '0') then
                aux(SP) <= aux(SP) + 1;
            elsif (wpc = '1' and up = '0' and dw = '0') then
                aux(SP) <= PCin;
            elsif (wpc = '1' and up = '1' and dw = '0') then
                SP := SP + 1;
                if(SP = 2**n) then
                    SP := 0;
                end if;
                aux(SP) <= PCin;
            elsif (wpc = '0' and up = '0' and dw = '1') then
                SP := SP - 1;
                if(SP = -1) then
                    SP := (2**n)-1;
                end if;
                aux(SP) <= aux(SP) + 1;
            end if;
        end if;
        PCout <= aux(SP);
        SPout <= conv_std_logic_vector(SP, n);
    end process;
end Behavioral;
```

Código de simulación

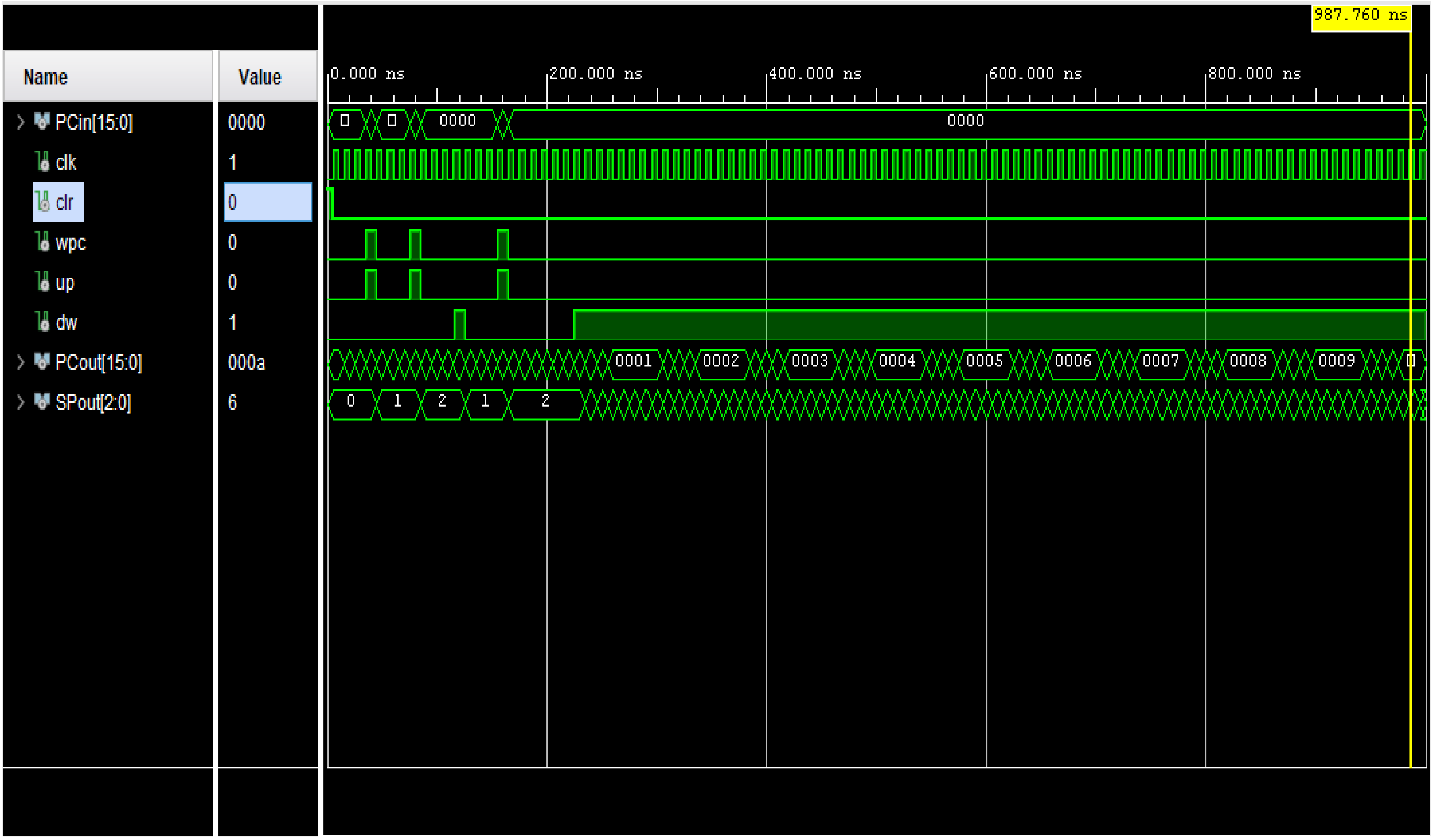


```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_arith.all;
use IEEE.STD_LOGIC_unsigned.ALL;
use IEEE.STD_LOGIC_TEXTIO.ALL;
use STD.TEXTIO.ALL;

entity test_bench is
end test_bench;

architecture Behavioral of test_bench is
    component Pila is
        Port ( PCin : in STD_LOGIC_VECTOR (15 downto 0);
               clk, clr, wpc, up, dw : in STD_LOGIC;
               PCout : out STD_LOGIC_VECTOR (15 downto 0);
               SPout : out STD_LOGIC_VECTOR (2 downto 0));
    end component;

    signal PCin : STD_LOGIC_VECTOR (15 downto 0) := (others => '0');
    signal clk, clr, wpc, up, dw : STD_LOGIC;
    signal PCout : STD_LOGIC_VECTOR (15 downto 0);
    signal SPout : STD_LOGIC_VECTOR (2 downto 0);
begin
    stack: Pila Port map (
        PCin => PCin,
        clk => clk,
        clr => clr,
        wpc => wpc,
        up => up,
        dw => dw,
        PCout => PCout,
        SPout => SPout
    );

    reloj : process begin
        clk <= '0';
        wait for 5 ns;
        clk <= '1';
        wait for 5 ns;
    end process;

    process
        file arch_res : text;   --Apuntadores tipo txt
        variable linea_res : line;
        variable var_pc_out : STD_LOGIC_VECTOR (15 downto 0);

        file arch_en : text; --Apuntadores tipo txt
        variable linea_en: line;
        variable var_pc_in : STD_LOGIC_VECTOR (15 downto 0);
        variable var_clr : STD_LOGIC;
        variable var_wpc : STD_LOGIC;
        variable var_up : STD_LOGIC;
        variable var_dw : STD_LOGIC;
        variable cadena : string (1 to 2);
    begin
        --- PCIN CLR WPC UP DW
        file_open(arch_en, "C:\Users\YaKerTaker\Google Drive\8vo\Arquitectura-
Computadoras\Practica10\Pila\Pila.srcs\sim_1\new\Estimulos.txt", READ_MODE);

        --- SP PC
        file_open(arch_res, "C:\Users\YaKerTaker\Google Drive\8vo\Arquitectura-
Computadoras\Practica10\Pila\Pila.srcs\sim_1\new\Resultado.txt", WRITE_MODE);

        cadena := "SP";
        write(linea_res, cadena, right, cadena'LENGTH+1);--ESCRIBE LA cadena "SP"
        cadena := "PC";
        write(linea_res, cadena, right, cadena'LENGTH+1);--ESCRIBE LA cadena "PC"

        writeline(arch_res, linea_res);-- escribe la linea en el archivo

        for i in 1 to 24 loop
            readline(arch_en, linea_en); -- lee una linea completa
            --- PCIN CLR WPC UP DW

            --Lee PCIN
            Hread(linea_en, var_pc_in);
            PCin <= var_pc_in;

            --Lee CLR
            read(linea_en, var_clr);
            clr <= var_clr;

            --Lee WPC
            read(linea_en, var_wpc);
            wpc <= var_wpc;

            --Lee UP
            read(linea_en, var_up);
            up <= var_up;

            --Lee DW
            read(linea_en, var_dw);
            dw <= var_dw;

            wait until rising_edge(clk); --ESPERA AL FLANCO DE SUBIDA
            wait for 0.1 ns;
            var_pc_out := PCout;

            --- SP PC
            Hwrite(linea_res, SPout, right, 3);--ESCRIBE EL CAMPO SP
            Hwrite(linea_res, var_pc_out, right, 5);--ESCRIBE EL CAMPO PCOUT

            writeline(arch_res, linea_res);-- escribe la linea en el archivo
        end loop;
        file_close(arch_en);  -- cierra el archivo
        file_close(arch_res);  -- cierra el archivo
        wait;
    end process;
end Behavioral;
```

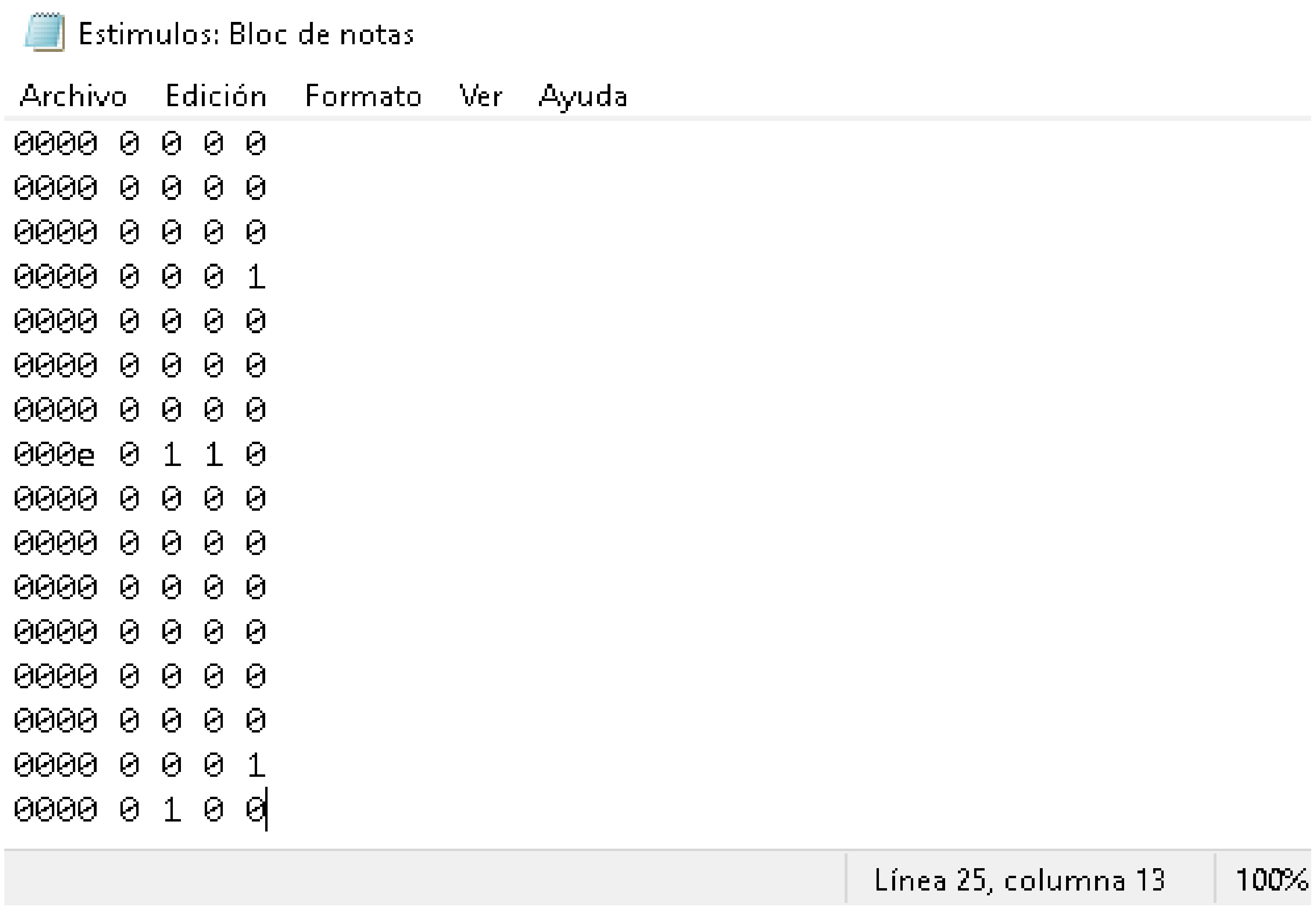
Diagrama RTL



Forma de onda

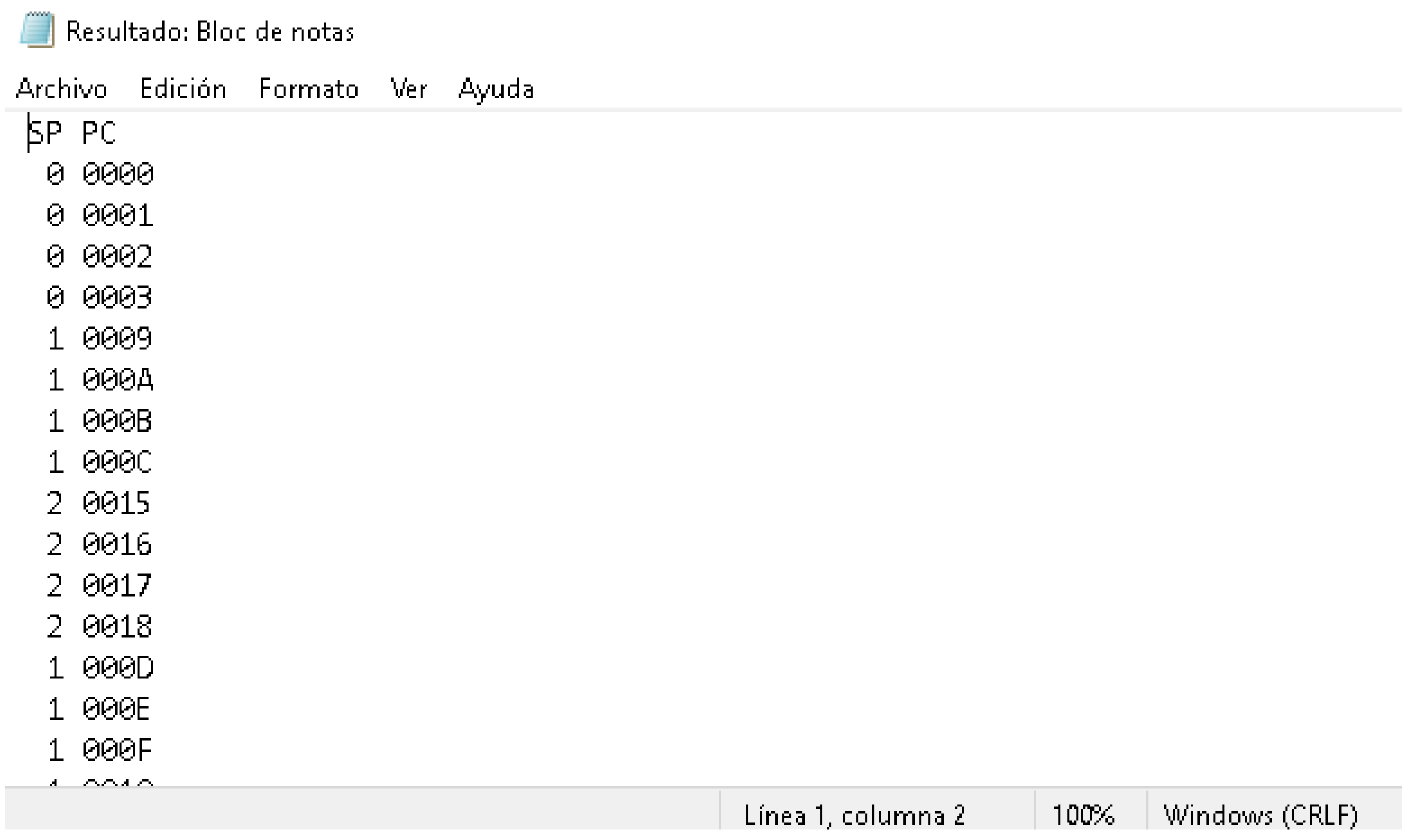


Archivo de estímulos



```
0000 0 0 0 0
0000 0 0 0 0
0000 0 0 0 0
0000 0 0 0 1
0000 0 0 0 0
0000 0 0 0 0
0000 0 0 0 0
000e 0 1 1 0
0000 0 0 0 0
0000 0 0 0 0
0000 0 0 0 0
0000 0 0 0 0
0000 0 0 0 0
0000 0 0 0 0
0000 0 0 0 1
0000 0 1 0 0
```

Archivo de resultados



```
SP PC
 0 0000
 0 0001
 0 0002
 0 0003
 1 0009
 1 000A
 1 000B
 1 000C
 2 0015
 2 0016
 2 0017
 2 0018
 1 000D
 1 000E
 1 000F
```