Código de implementación:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;
use ieee.numeric_std.all;

entity pila is
   Port ( PCin : in STD_LOGIC_VECTOR (15 downto 0);
          wpc : in STD_LOGIC;
          up : in STD_LOGIC;
          down : in STD_LOGIC;
          clk : in STD_LOGIC;
          clr : in STD_LOGIC;
          PCout : out STD_LOGIC_VECTOR (15 downto 0);
          SPout : out STD_LOGIC_VECTOR (2 downto 0));
end pila;

architecture Behavioral of pila is
type pila is array (0 to 7) of std_logic_vector(15 downto 0);
begin

process(clr,clk)
variable aux: pila;
variable sp: integer range 0 to 7;
begin
   if (clr = '1') then
      sp := 0;
      aux := (others => (others => '0'));
   elsif ( rising_edge(clk) ) then
      if ( wpc = '0' and up = '0' and down = '0' ) then
         aux(sp) := aux(sp) + 1;
      elsif ( wpc = '1' and up = '0' and down = '0' ) then
         aux(sp) := PCin;
      elsif ( wpc = '1' and up = '1' and down = '0' ) then
         sp := sp + 1;
         aux(sp) := PCin;
      elsif ( wpc = '0' and up = '0' and down = '1' ) then
         sp := sp - 1;
         aux(sp) := aux(sp) + 1;
      end if;
   end if;
   PCout <= aux(sp);
   SPout <= conv_std_logic_vector(sp, 3);
end process;
end Behavioral;
```

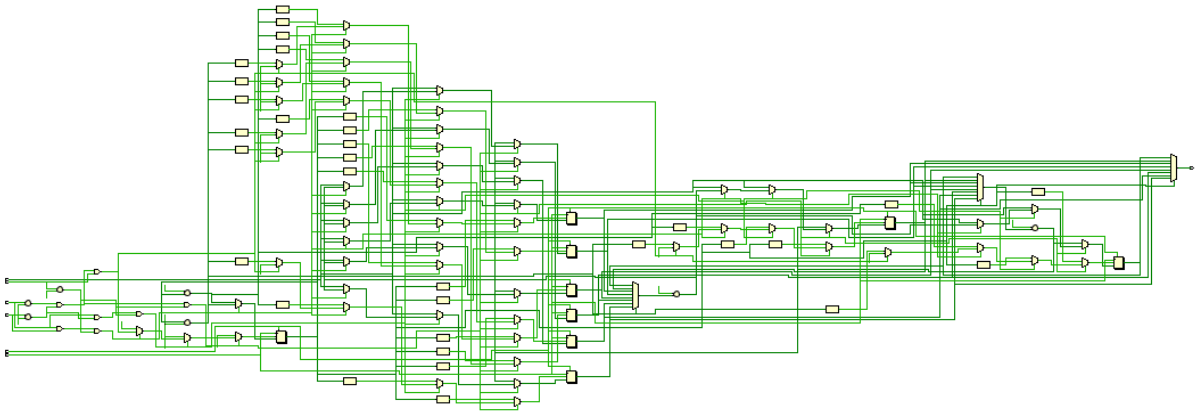
Código de simulación:

```
LIBRARY ieee;
LIBRARY STD;
USE STD.TEXTIO.ALL;
USE ieee.std_logic_TEXTIO.ALL;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_ARITH.ALL;
use ieee.numeric_std.all;

entity tb_pila is

end tb_pila;

architecture Behavioral of tb_pila is

component pila is
   Port ( PCin : in STD_LOGIC_VECTOR (15 downto 0);
          wpc : in STD_LOGIC;
          up : in STD_LOGIC;
          down : in STD_LOGIC;
          clk : in STD_LOGIC;
          clr : 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);
signal wpc : STD_LOGIC;
signal up : STD_LOGIC;
signal down : STD_LOGIC;
signal clk : STD_LOGIC;
signal clr : STD_LOGIC;
signal PCout : STD_LOGIC_VECTOR (15 downto 0);
signal SPout : std_logic_vector (2 downto 0);

constant CLK_period : time := 1 ns;

begin

elemento1: pila Port map (
   PCin => PCin,
   wpc => wpc,
   up => up,
   down => down,
   clk => clk,
   clr => clr,
   PCout => PCout,
   SPout => SPout
);

clock : process begin
```

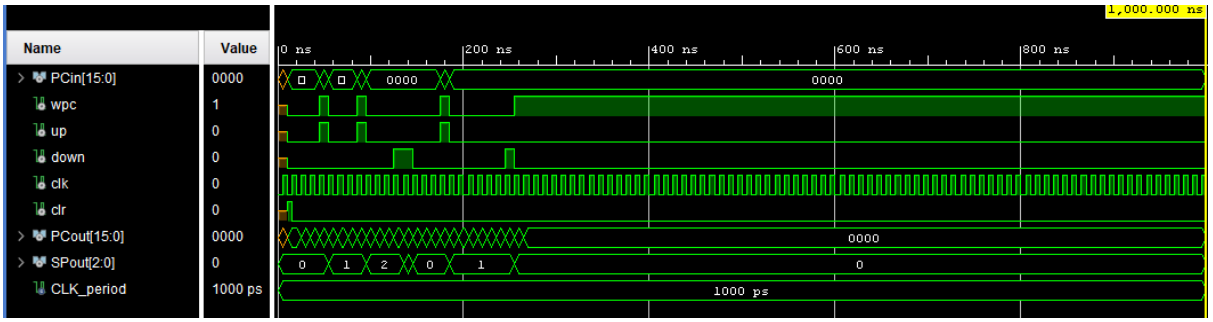
```
    CLK <= '0';
    wait for 5ns;
    CLK <= '1';
    wait for 5ns;
end process;

stim_proc: process
    variable var_PCin : STD_LOGIC_VECTOR (15 downto 0);
    variable var_wpc : STD_LOGIC;
    variable var_up : STD_LOGIC;
    variable var_down : STD_LOGIC;
    variable var_clr : STD_LOGIC;

    variable var_PCout : STD_LOGIC_VECTOR (15 downto 0);
    variable var_SPout : STD_LOGIC_VECTOR (2 downto 0);

            file ARCH_RES : TEXT;

            variable LINEA_RES : line;

            file ARCH_VEC : TEXT;
            variable LINEA_VEC : line;

            VARIABLE CADENA : STRING(1 TO 6);
    begin
                    file_open(ARCH_RES, "C:\Users\sebas\Desktop\PracticasArq\practica
7\resultado.txt", WRITE_MODE);
        file_open(ARCH_VEC, "C:\Users\sebas\Desktop\PracticasArq\practica 7\vectores.txt",
READ_MODE);
        CADENA := "SP    ";
                    write(LINEA_RES, CADENA, left, CADENA'LENGTH);
                    CADENA := "PC    ";
                    write(LINEA_RES, CADENA, left, CADENA'LENGTH);
                    writeline(ARCH_RES,LINEA_RES);

        wait for 10ns;

                    FOR I IN 0 TO 25 LOOP
                            readline(ARCH_VEC,LINEA_VEC);

                            read(LINEA_VEC, var_clr);
            clr <= var_clr;
            read(LINEA_VEC, var_wpc);
            wpc <= var_wpc;
            read(LINEA_VEC, var_up);
            up <= var_up;
            read(LINEA_VEC, var_down);
            down <= var_down;
            Hread(LINEA_VEC, var_PCin);
            PCin <= var_PCin;

                            WAIT UNTIL RISING_EDGE(CLK);
```

```
        var_PCout := PCout;
        var_SPout := SPout;

                        Hwrite(LINEA_RES, var_SPout, left, 6);
                        Hwrite(LINEA_RES, var_PCout, left, 6);
                        writeline(ARCH_RES,LINEA_RES);
                end loop;

                file_close(ARCH_VEC);
                file_close(ARCH_RES);
    wait;
  end process;
end Behavioral;
```

Diagrama RTL:



Onda de simulación:



Practica 7 Cipriano Damián Sebastián No. 8

Vectores:

vectores: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
1 0 0 0 0000
0 0 0 0 0000
0 0 0 0 0000
0 0 0 0 0000
0 1 1 0 0009
0 0 0 0 0000
0 0 0 0 0000
0 0 0 0 0000
0 1 1 0 0015
0 0 0 0 0000
0 0 0 0 0000
0 0 0 0 0000
0 0 0 1 0000
0 0 0 1 0000
0 0 0 0 0000
0 0 0 0 0000
0 0 0 0 0000
0 1 1 0 000E
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 0000
```

Resultados:

resultado: Bloc de notas

Archivo Edición Formato Ver Ayuda

SP	PC
0	0000
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
0	0004
0	0005
0	0006
0	0007
1	000E
1	000F
1	0010
1	0011
1	0012
1	0013
1	0014
0	0008