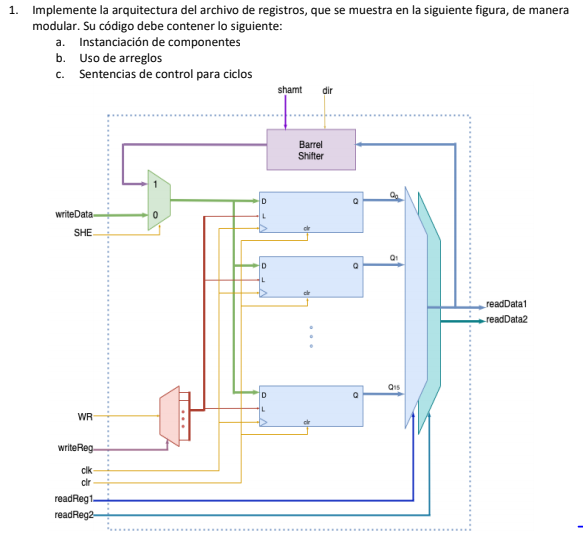
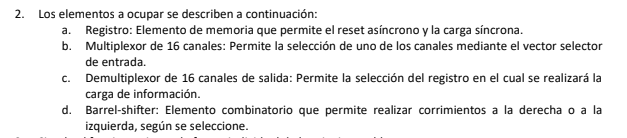
# Práctica 6

1.

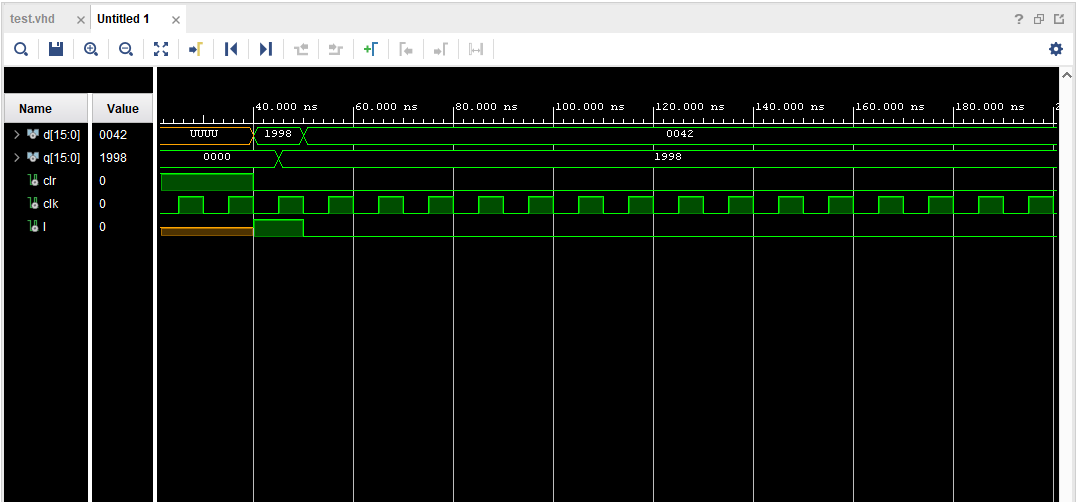


2.

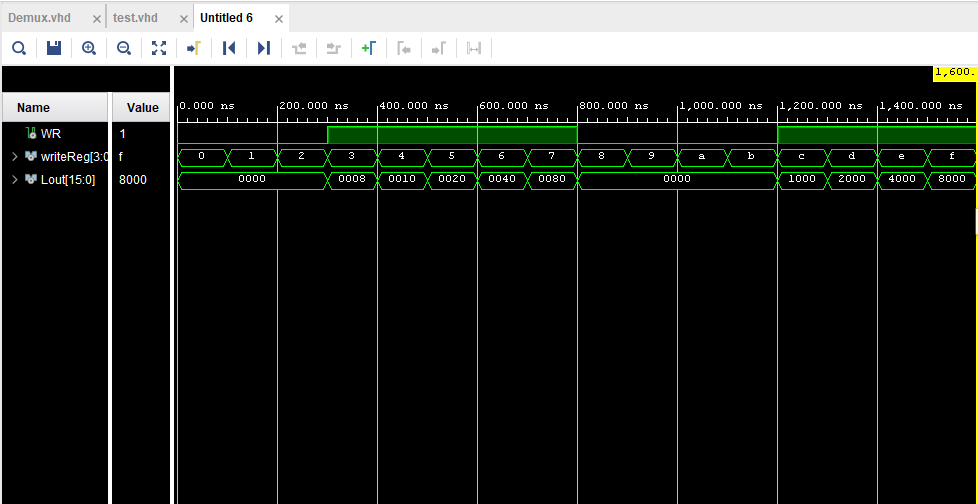


3.

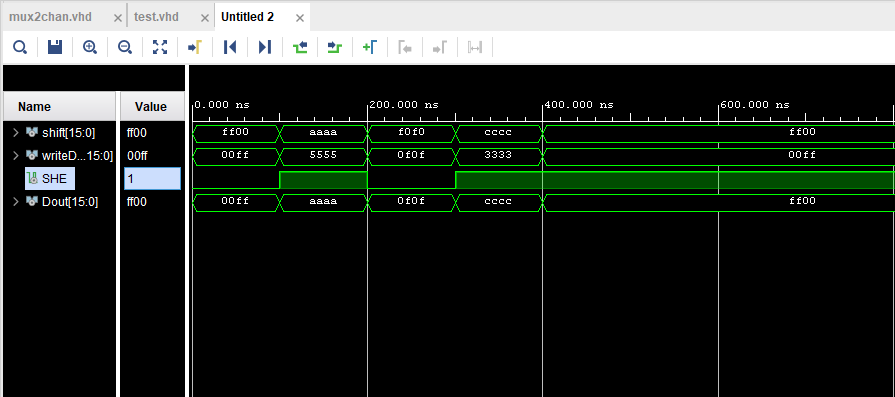
**Simulación Registro**



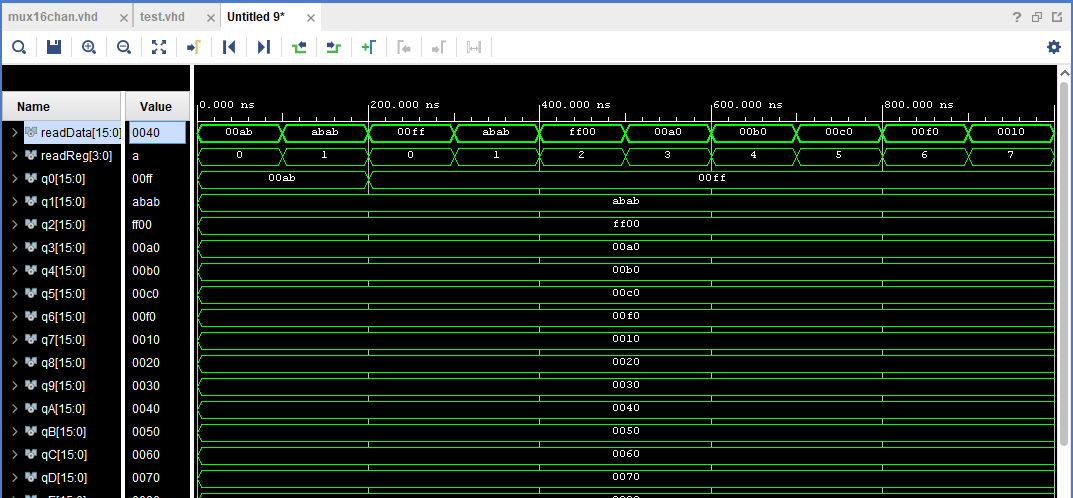
**Simulación Demux**



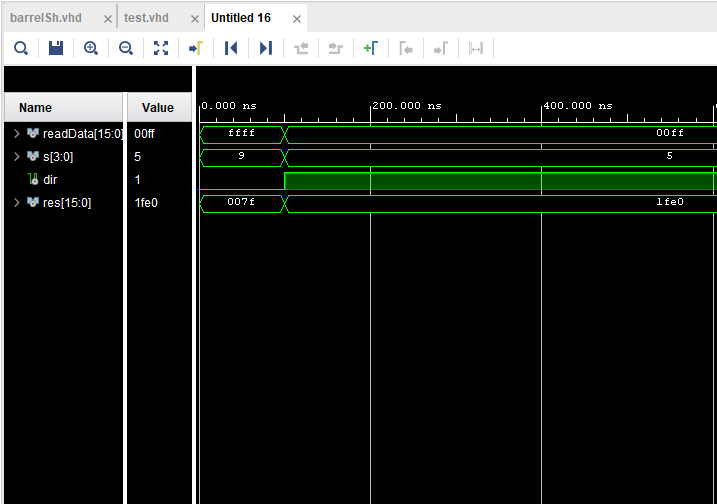
**Simulación MUX 2 canales**



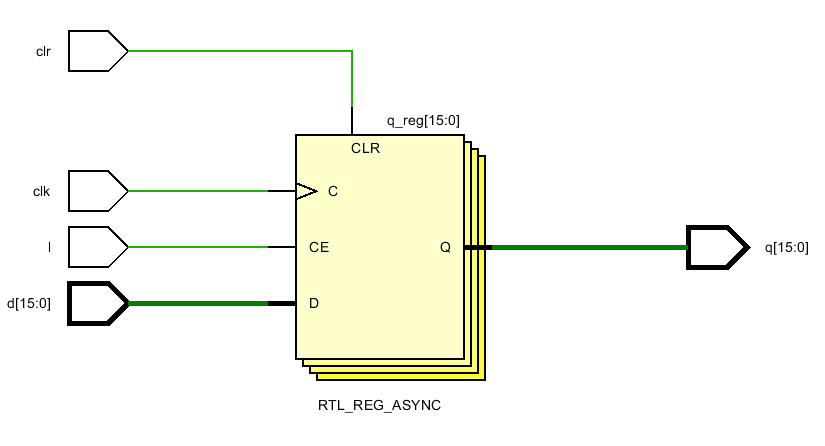
**Simulación MUX 16 canales**



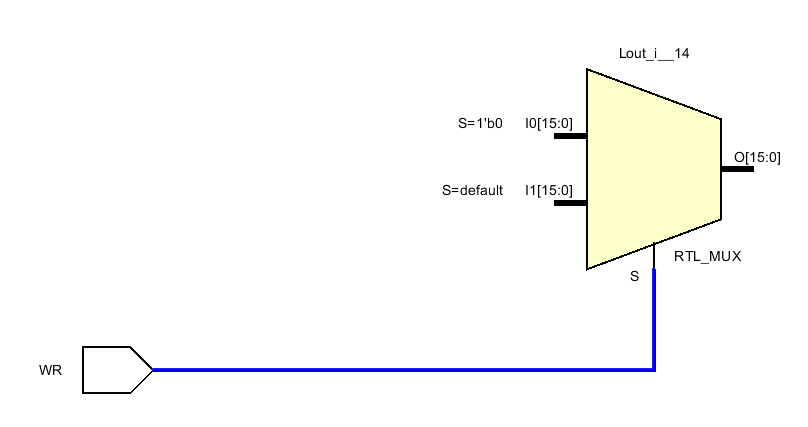
**Simulación BarrelShifter**



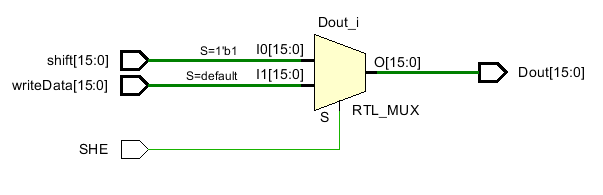
**Diagrama RTL Registro**



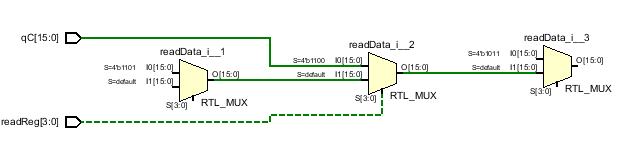
**Diagrama RTL DEMUX**



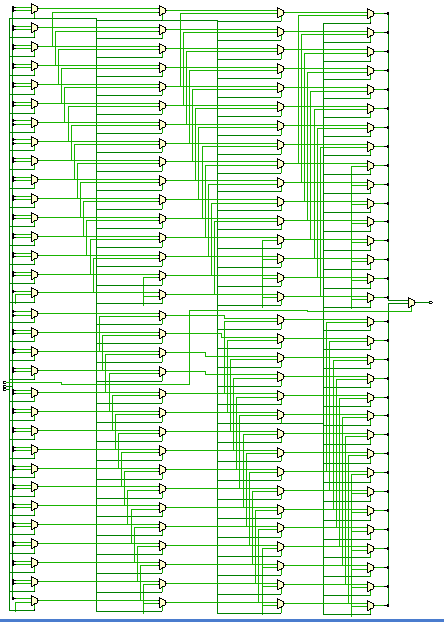
**Diagrama RTL Multiplexor 2 Canales**



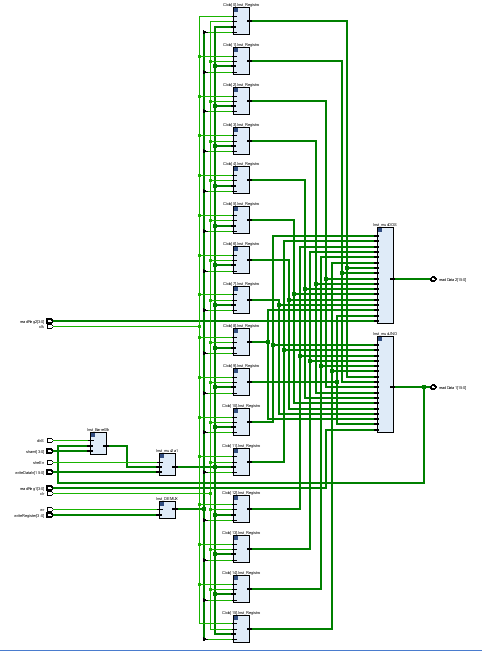
**Diagrama RTL Multiplexor 16 Canales**



**Diagrama RTL BarrelShifter**

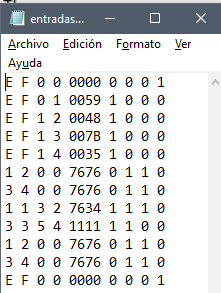


**Diagrama RTL del Archivo de Registros**

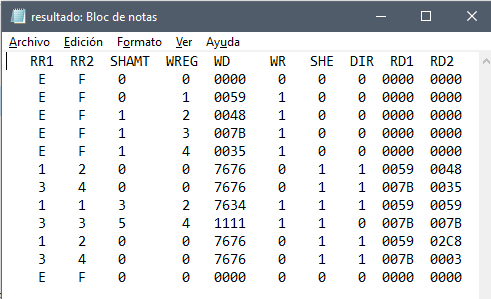


**4. Simulación Archivo de Registros**

Para realizar la simulación se usó el siguiente conjunto de archivos, siendo el archivo de entrada el nombrado como ***entradas.txt*** y el de salida el archivo ***resultado.txt***; a continuación, se muestra el contenido de ambos.



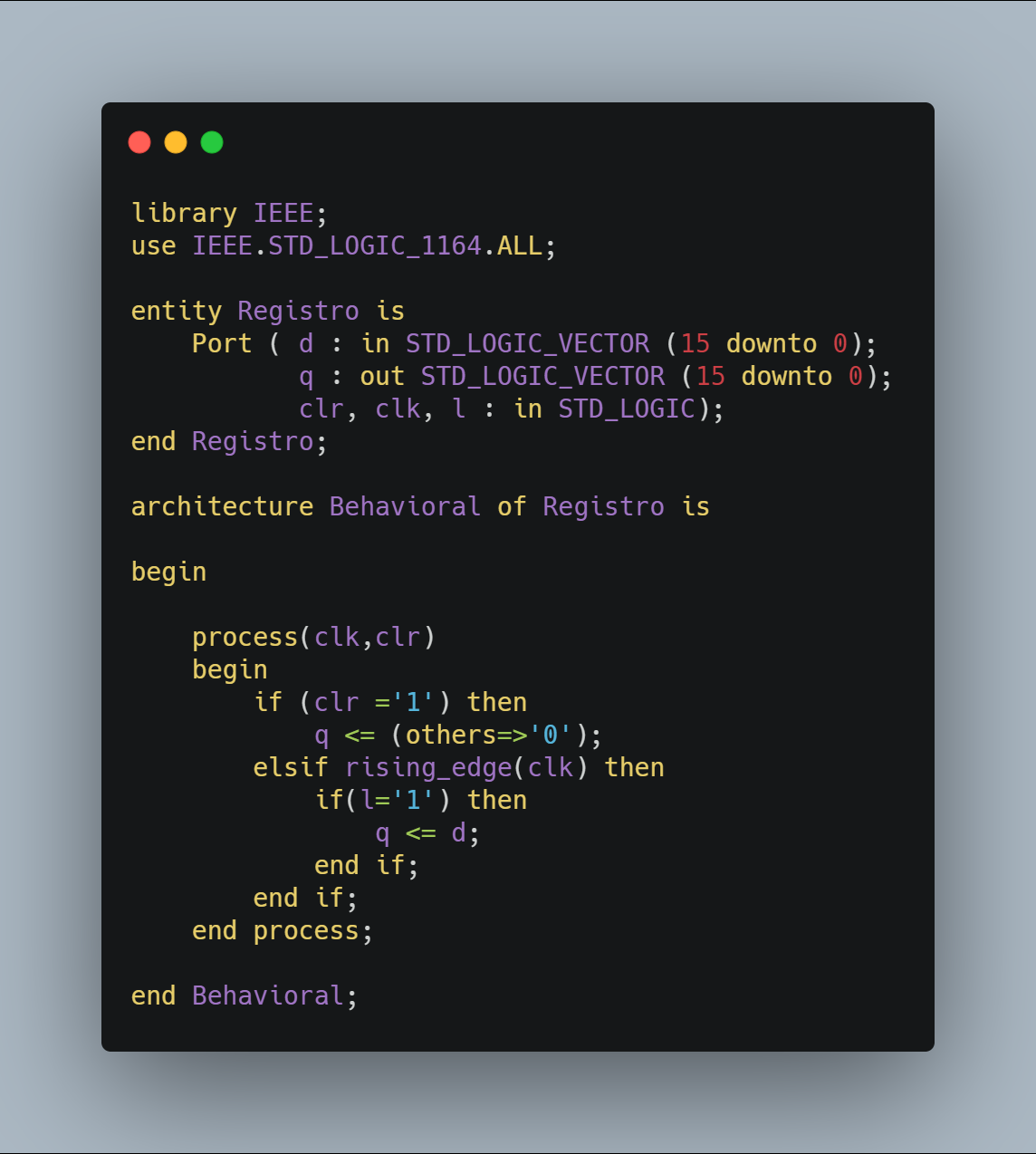
**5. Resultado de la simulación, las salidas del archivo están en formato hexadecimal.**



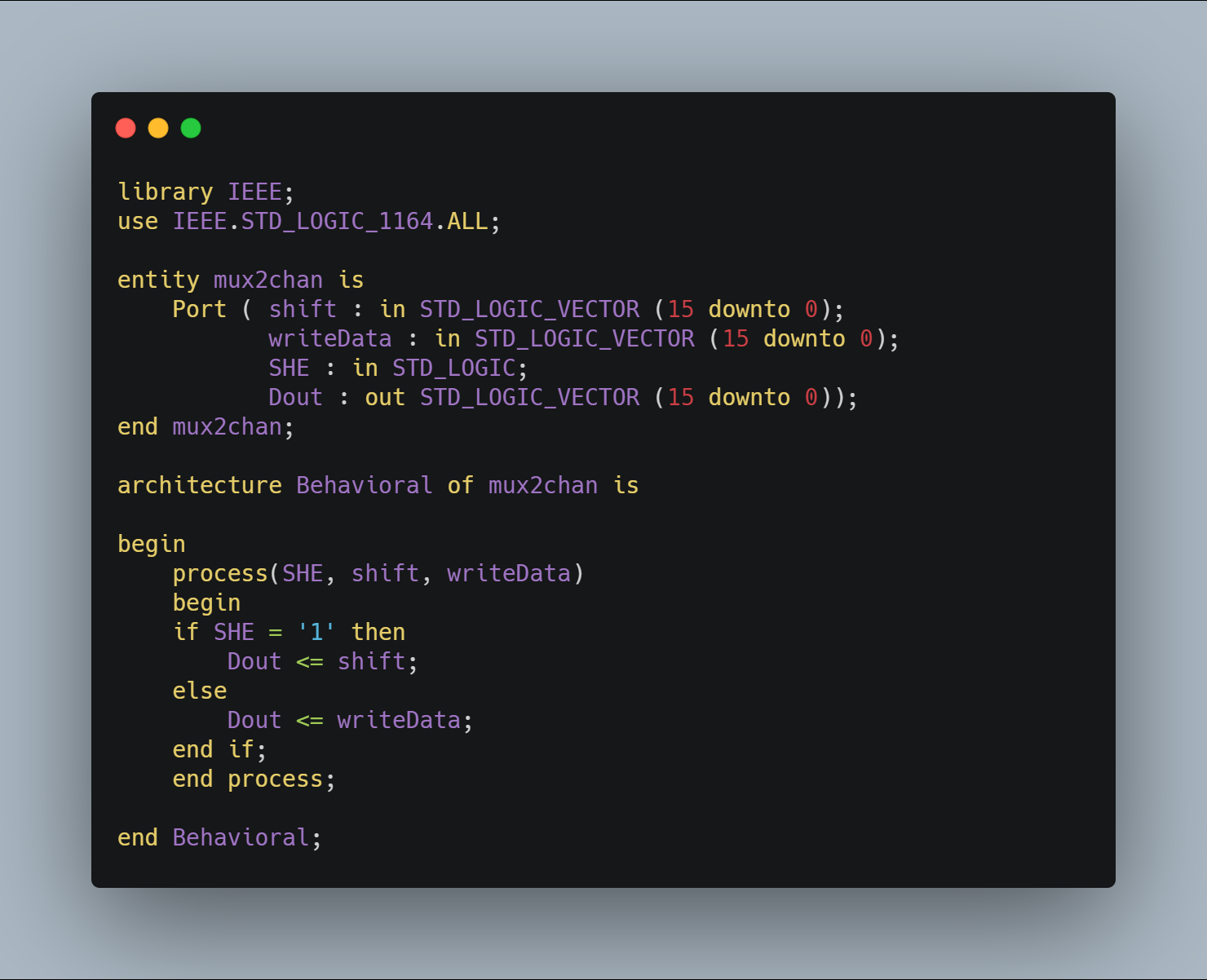
**6.**

**a) Código de implementación**

**Registro**



**Multiplexor 2 canales**



**Multiplexor 16 canal**



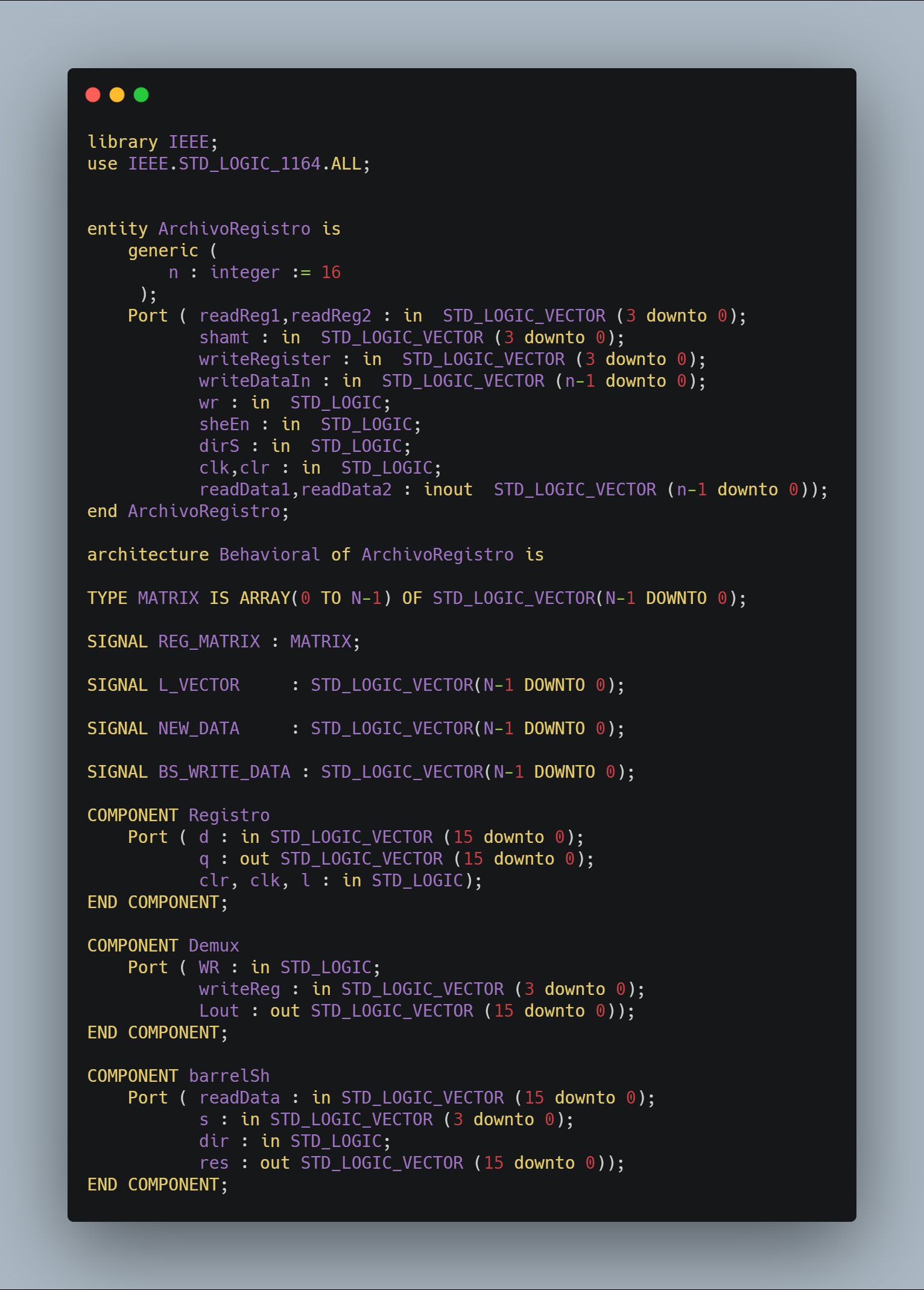
**Demultiplexor**



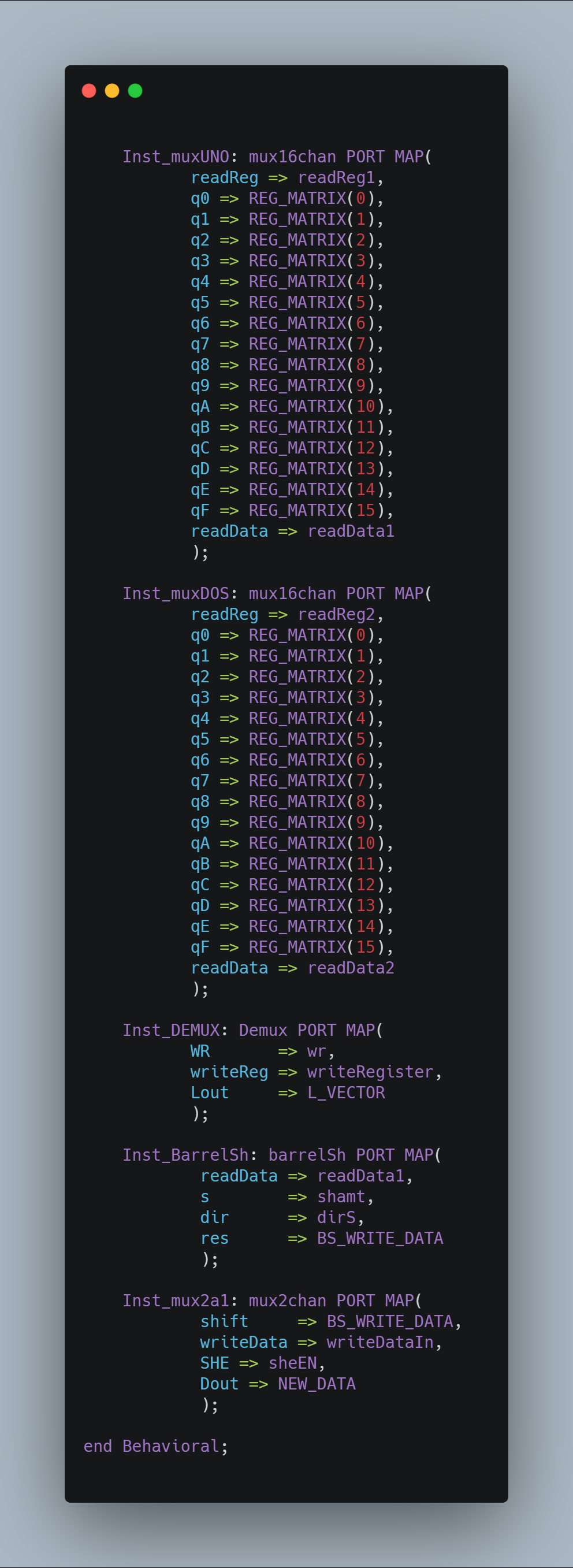
**BarrelShifter**



**Archivo de Registros**

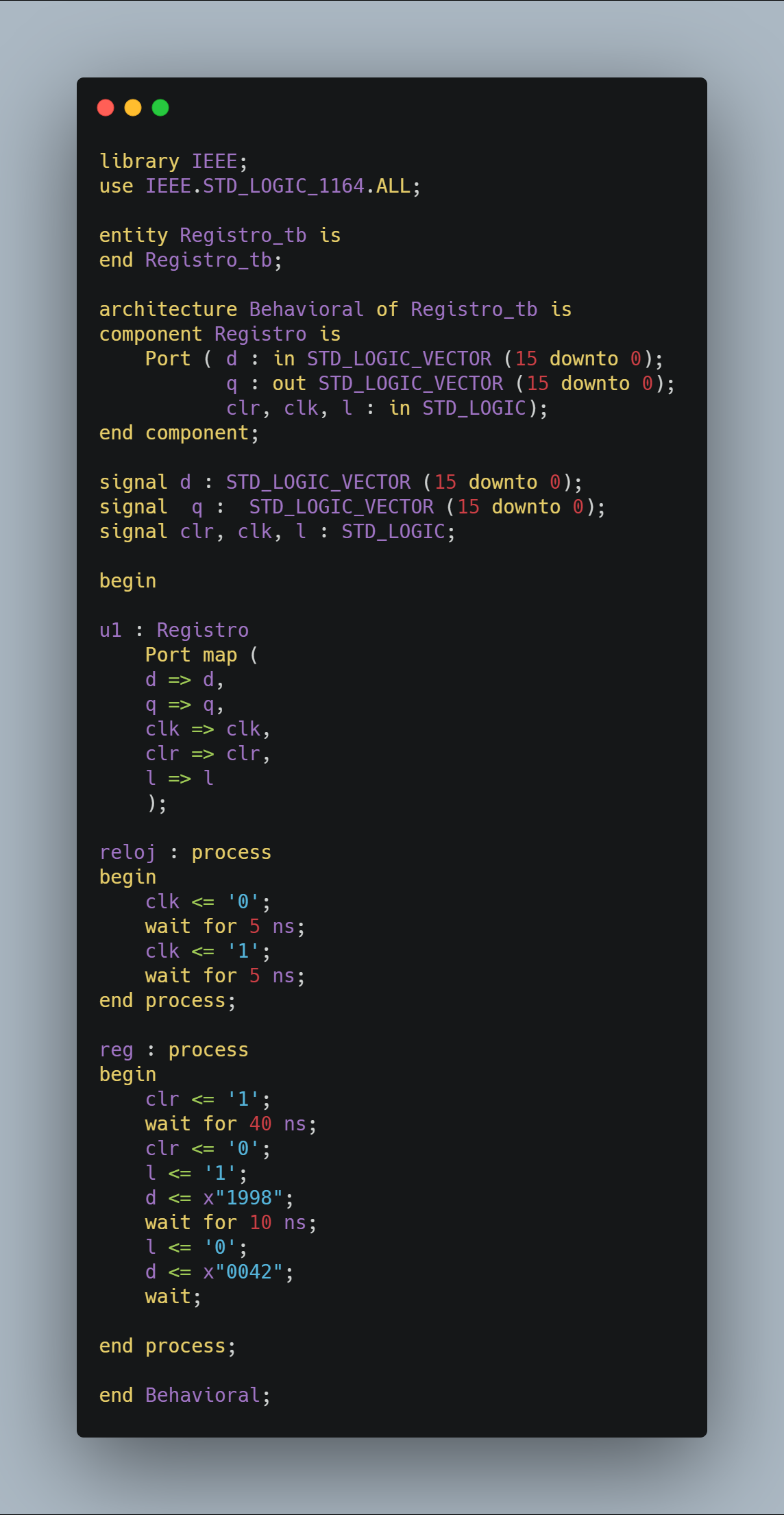




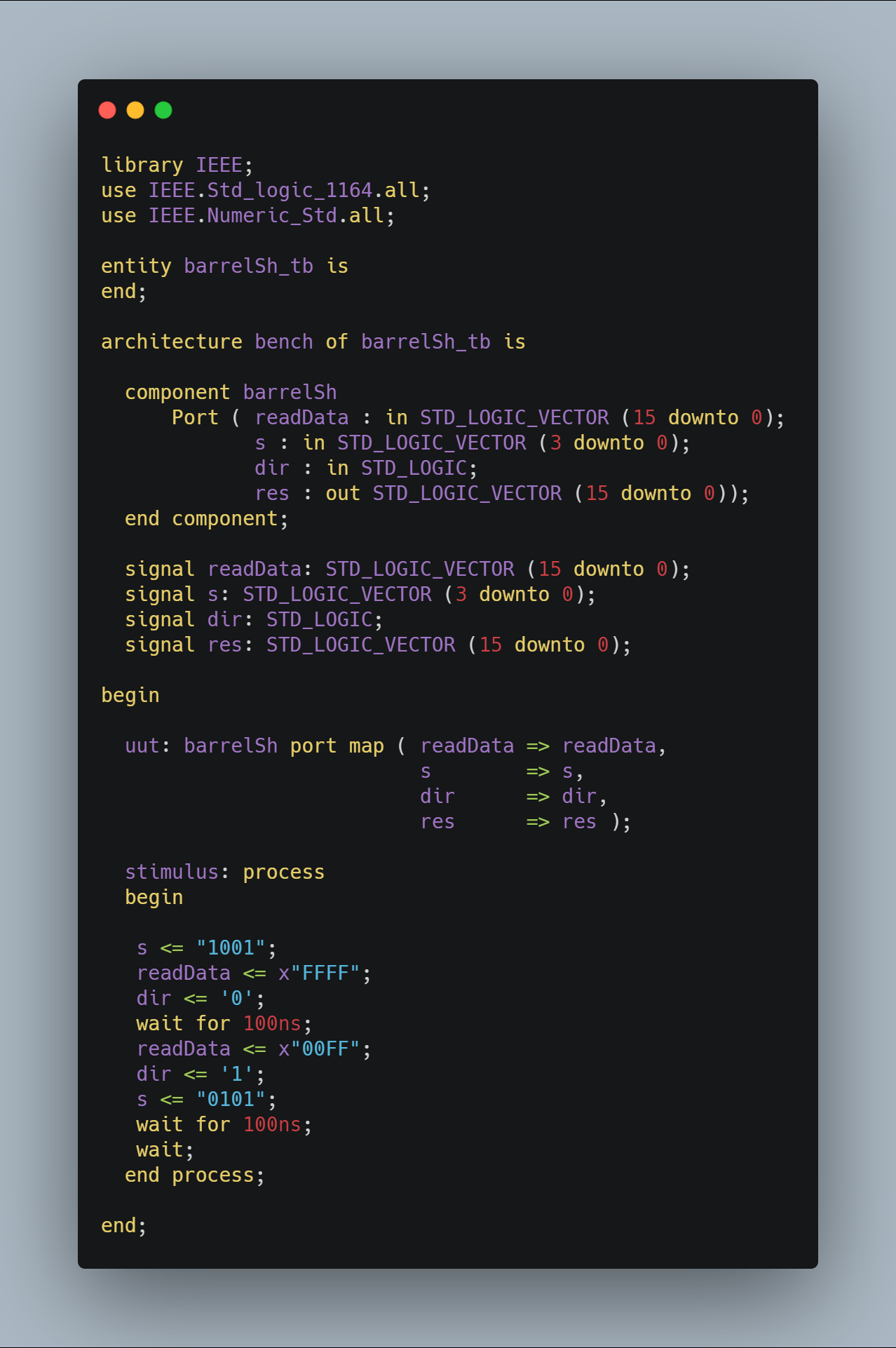


**b) Código de simulación(test bench)**

**Registro**



**Barrel Shifter**



**Archivo de Registros**

LIBRARY ieee;

LIBRARY STD;

USE STD.TEXTIO.ALL;

USE ieee.std\_logic\_TEXTIO.ALL;  *--PERMITE USAR STD\_LOGIC*

USE ieee.std\_logic\_1164.ALL;

USE ieee.std\_logic\_UNSIGNED.ALL;

USE ieee.std\_logic\_ARITH.ALL;

entity tb\_ArchivoRegistro is

end tb\_ArchivoRegistro;

architecture tb of tb\_ArchivoRegistro is

    component ArchivoRegistro

        port (readReg1      : in std\_logic\_vector (3 downto 0);

              readReg2      : in std\_logic\_vector (3 downto 0);

              shamt         : in std\_logic\_vector (3 downto 0);

              writeRegister : in std\_logic\_vector (3 downto 0);

              writeDataIn   : in std\_logic\_vector (15 downto 0);

              wr            : in std\_logic;

              sheEn         : in std\_logic;

              dirS          : in std\_logic;

              clk           : in std\_logic;

              clr           : in std\_logic;

              readData1     : inout std\_logic\_vector (15 downto 0);

              readData2     : inout std\_logic\_vector (15 downto 0));

    end component;

    signal readReg1      : std\_logic\_vector (3 downto 0);

    signal readReg2      : std\_logic\_vector (3 downto 0);

    signal shamt         : std\_logic\_vector (3 downto 0);

    signal writeRegister : std\_logic\_vector (3 downto 0);

    signal writeDataIn   : std\_logic\_vector (15 downto 0);

    signal wr            : std\_logic;

    signal sheEn         : std\_logic;

    signal dirS          : std\_logic;

    signal clk           : std\_logic;

    signal clr           : std\_logic;

    signal readData1     : std\_logic\_vector (15 downto 0);

    signal readData2     : std\_logic\_vector (15 downto 0);

    constant CLK\_period : time := 10 ns;

begin

    dut : ArchivoRegistro

    port map (readReg1      => readReg1,

              readReg2      => readReg2,

              shamt         => shamt,

              writeRegister => writeRegister,

              writeDataIn   => writeDataIn,

              wr            => wr,

              sheEn         => sheEn,

              dirS          => dirS,

              clk           => clk,

              clr           => clr,

              readData1     => readData1,

              readData2     => readData2

              );

    CLK\_process :process

    begin

        CLK <= '0';

        wait for CLK\_period/2;

        CLK <= '1';

        wait for CLK\_period/2;

    end process;

    stimuli : process

    file ARCH\_RES : TEXT;

    variable LINEA\_RES : LINE;

    VARIABLE VAR\_RDU : STD\_LOGIC\_VECTOR(15 DOWNTO 0);

    VARIABLE VAR\_RDD : STD\_LOGIC\_VECTOR(15 DOWNTO 0);

    file ARCH\_VEC : TEXT;

    variable LINEA\_VEC : line;

    VARIABLE VAR\_RR1    :  STD\_LOGIC\_VECTOR(3 DOWNTO 0);

    VARIABLE VAR\_RR2    :  STD\_LOGIC\_VECTOR(3 DOWNTO 0);

    VARIABLE VAR\_SHAMT  :  STD\_LOGIC\_VECTOR(3 DOWNTO 0);

    VARIABLE VAR\_WRE    :  STD\_LOGIC\_VECTOR(3 DOWNTO 0);

    VARIABLE VAR\_WD     :  STD\_LOGIC\_VECTOR(15 DOWNTO 0);

    VARIABLE VAR\_WR     :  STD\_LOGIC;

    VARIABLE VAR\_SHE    :  STD\_LOGIC;

    VARIABLE VAR\_DIR    :  STD\_LOGIC;

    VARIABLE VAR\_CLR    :  STD\_LOGIC;

    VARIABLE CADENA : STRING(1 TO 4);

    VARIABLE CADENA\_I : STRING(1 TO 6);

    VARIABLE CADENA\_W : STRING(1 TO 10);

    VARIABLE CADENA\_X : STRING(1 TO 5);

    begin

        file\_open(ARCH\_RES, "D:\ESCOM\ARQUITECTURA\ArchivoRegistros\ArchivoRegistros.srcs\sim\_1\new\resultado.txt", WRITE\_MODE);

        file\_open(ARCH\_VEC, "D:\ESCOM\ARQUITECTURA\ArchivoRegistros\ArchivoRegistros.srcs\sim\_1\new\entradas.txt", READ\_MODE);

        CADENA\_X := "  RR1";

        write(LINEA\_RES, CADENA\_X, right, CADENA\_X'LENGTH+1);

        CADENA := " RR2";

        write(LINEA\_RES, CADENA, right, CADENA'LENGTH+1);

        CADENA\_I := " SHAMT";

        write(LINEA\_RES, CADENA\_I, right, CADENA\_I'LENGTH+1);

        CADENA\_X := " WREG";

        write(LINEA\_RES, CADENA\_X, right, CADENA\_X'LENGTH+1);

        CADENA := " WD ";

        write(LINEA\_RES, CADENA, right, CADENA'LENGTH+1);

        CADENA\_I := "   WR ";

        write(LINEA\_RES, CADENA\_I, right, CADENA\_I'LENGTH+1);

        CADENA := " SHE";

        write(LINEA\_RES, CADENA, right, CADENA'LENGTH+1);

        CADENA := " DIR";

        write(LINEA\_RES, CADENA, right, CADENA'LENGTH+1);

        CADENA := " RD1";

        write(LINEA\_RES, CADENA, right, CADENA'LENGTH+1);

        CADENA := " RD2";

        write(LINEA\_RES, CADENA, right, CADENA'LENGTH+1);

        writeline(ARCH\_RES,LINEA\_RES);

      wait for 100 ns;

      FOR I IN 0 TO 11 LOOP

            readline(ARCH\_VEC,LINEA\_VEC);

            Hread(LINEA\_VEC, VAR\_RR1);

            readReg1<=VAR\_RR1;

            Hread(LINEA\_VEC, VAR\_RR2);

            readReg2<=VAR\_RR2;

            Hread(LINEA\_VEC, VAR\_SHAMT);

            SHAMT<=VAR\_SHAMT;

            Hread(LINEA\_VEC, VAR\_WRE);

            writeRegister<=VAR\_WRE;

            Hread(LINEA\_VEC, VAR\_WD);

            writeDataIn<=VAR\_WD;

            read(LINEA\_VEC, VAR\_WR);

            WR<=VAR\_WR;

            read(LINEA\_VEC, VAR\_SHE);

            sheEn<=VAR\_SHE;

            read(LINEA\_VEC, VAR\_DIR);

            dirS<=VAR\_DIR;

            read(LINEA\_VEC, VAR\_CLR);

            CLR<=VAR\_CLR;

            WAIT UNTIL RISING\_EDGE(CLK);

            VAR\_RDU := readData1;

            VAR\_RDD := readData2;

            Hwrite(LINEA\_RES, VAR\_RR1,  right, 5);

            Hwrite(LINEA\_RES, VAR\_RR2,  right, 5);

            Hwrite (LINEA\_RES, VAR\_SHAMT, right, 5);

            Hwrite (LINEA\_RES, VAR\_WRE,     right, 8);

            Hwrite (LINEA\_RES, VAR\_WD,  right, 7);

            write(LINEA\_RES, VAR\_WR,    right, 5);

            write(LINEA\_RES, VAR\_SHE,   right, 5);

            write(LINEA\_RES, VAR\_DIR,   right, 5);

            Hwrite (LINEA\_RES, VAR\_RDU, right, 6);

            Hwrite (LINEA\_RES, VAR\_RDD, right, 6);

            writeline(ARCH\_RES,LINEA\_RES);

        end loop;

        file\_close(ARCH\_VEC);

        file\_close(ARCH\_RES);

        wait for CLK\_period\*10;

        wait;

    end process;

end tb;