*****Instituto Politécnico Nacional***

***Escuela Superior de Cómputo***

*Arquitectura de Computadoras*

***Practica 11: Pila Hardware 2***

***Nombre:*** *Sampayo Hernández Mauro*

***Grupo:*** *3CV8*

***Profesor:*** *Nayeli Vega García*

***Fecha de entrega:*** *18 de abril del 2020*

**Código de Implementación:**

* **Pila**

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**use** IEEE**.**STD\_LOGIC\_arith**.ALL;**--Datos con signos y sin signo, y operaciones aritmeticas

**use** IEEE**.**STD\_LOGIC\_unsigned**.ALL;**--Realizar operaciones sin signo para los ST\_LOGIC\_VECTOR

**entity** Pila **is**

**generic** **(** N**:** integer **:=**3**;**

M**:** integer **:=**16**);**

**Port** **(** PC\_in **:** **in** STD\_LOGIC\_VECTOR **(**M**-**1 **downto** 0**);**

PC\_out **:** **out** STD\_LOGIC\_VECTOR **(**M**-**1 **downto** 0**);**

clk**,** clr**,** UP**,** DW**,** WPC **:** **in** STD\_LOGIC**;**

SP**:** **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**;**

**signal** SP1**:** integer **range** 0 **to** **(**2**\*\***N**)-**1**;**

**begin**

**process(**clk**,** clr**)**

**variable** SPout **:** integer **range** 0 **to** **(**2**\*\***N**)-**1**;**

**begin**

**if(**clr **=** '1'**)then**

SPout **:=** 0**;**

aux **<=** **(others** **=>** **(others** **=>** '0'**));**

**elsif(**clk'**event** and clk **=** '1'**)then**

**if(**WPC **=** '0' and UP **=** '0' and DW **=** '0'**)then**--incremento

aux**(**SPout**)** **<=** aux**(**SPout**)+**1**;**

**elsif(**WPC **=** '1' and UP **=** '1' and DW **=** '0'**)then**--CALL

SPout **:=** SPout **+** 1**;**

aux**(**SPout**)** **<=** PC\_in**;**

**elsif(**WPC **=** '1' and UP **=** '0' and DW **=** '0'**)then**--JUMP

aux**(**SPout**)** **<=** PC\_in**;**

**elsif(**WPC **=** '0' and UP **=** '0' and DW **=** '1'**)then**--RET

SPout **:=** SPout **-** 1**;**

aux**(**SPout**)** **<=** aux**(**SPout**)+**1**;**

**end** **if;**

**end** **if;**

SP1 **<=** SPout**;**

**end** **process;**

SP **<=** conv\_std\_logic\_vector**(**SP1**,** 3**);**

PC\_out **<=** aux**(**SP1**);**

**end** Behavioral**;**

* **MemoriaPrograma**

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**use** IEEE**.**STD\_LOGIC\_arith**.ALL;**

**use** IEEE**.**STD\_LOGIC\_unsigned**.ALL;**

**entity** MemoriaPrograma **is**

**generic** **(** d **:** integer **:=** 25**;**

a **:** integer **:=** 16**);**

**Port** **(**PC **:** **in** STD\_LOGIC\_VECTOR **(**a**-**1 **downto** 0**);**

Inst **:** **out** STD\_LOGIC\_VECTOR **(**d**-**1 **downto** 0**));**

**end** MemoriaPrograma**;**

**architecture** Behavioral **of** MemoriaPrograma **is**

--INSTRUCCIONES

--Tipo I

**constant** LI **:** std\_logic\_vector**(**4 **downto** 0**)** **:=** "00001"**;**

**constant** LWI **:** std\_logic\_vector**(**4 **downto** 0**)** **:=** "00010"**;**

**constant** LW **:** std\_logic\_vector**(**4 **downto** 0**)** **:=** "10111"**;**

**constant** SWI **:** std\_logic\_vector**(**4 **downto** 0**)** **:=** "00011"**;**

**constant** SW **:** std\_logic\_vector**(**4 **downto** 0**)** **:=** "00100"**;**

**constant** ADDI **:** std\_logic\_vector**(**4 **downto** 0**):=** "00101"**;**

**constant** SUBI **:** std\_logic\_vector**(**4 **downto** 0**):=** "00110"**;**

**constant** ANDI**:** std\_logic\_vector**(**4 **downto** 0**)** **:=** "00111"**;**

**constant** ORI **:** std\_logic\_vector**(**4 **downto** 0**)** **:=** "01000"**;**

**constant** XORI **:** std\_logic\_vector**(**4 **downto** 0**)** **:=** "01001"**;**

**constant** NANDI **:** std\_logic\_vector**(**4 **downto** 0**):=** "01010"**;**

**constant** NORI**:** std\_logic\_vector**(**4 **downto** 0**)** **:=** "01011"**;**

**constant** XNORI **:** std\_logic\_vector**(**4 **downto** 0**):=** "01100"**;**

**constant** BEQI **:** std\_logic\_vector**(**4 **downto** 0**):=** "01101"**;**

**constant** BNEI **:** std\_logic\_vector**(**4 **downto** 0**):=** "01110"**;**

**constant** BLTI **:** std\_logic\_vector**(**4 **downto** 0**):=** "01111"**;**

**constant** BLETI **:** std\_logic\_vector**(**4 **downto** 0**):=** "10000"**;**

**constant** BGTI **:** std\_logic\_vector**(**4 **downto** 0**):=** "10001"**;**

**constant** BGETI **:** std\_logic\_vector**(**4 **downto** 0**):=** "10010"**;**

--Tipo R

**constant** TR **:** std\_logic\_vector**(**4 **downto** 0**)** **:=** "00000"**;**--Operaci󮠔ipo R

**constant** ADD **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0000"**;**

**constant** SUB **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0001"**;**

**constant** OpAND **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0010"**;**

**constant** OpOR **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0011"**;**

**constant** OpXOR **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0100"**;**

**constant** OpNAND**:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0101"**;**

**constant** OpNOR **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0110"**;**

**constant** OpXNOR **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0111"**;**

**constant** OpNOT **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "1000"**;**

**constant** OpSLL **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "1001"**;**

**constant** OpSRL **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "1010"**;**

--Tipo J

**constant** B**:** std\_logic\_vector**(**4 **downto** 0**):=** "10011"**;**

**constant** CALL **:** std\_logic\_vector**(**4 **downto** 0**):=** "10100"**;**

--Otros

**constant** RET **:** std\_logic\_vector**(**4 **downto** 0**):=** "10101"**;**

**constant** NOP **:** std\_logic\_vector**(**4 **downto** 0**):=** "10110"**;**

--Sin Uso

**constant** SU **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0000"**;**--Sin Uso

--REGISTROS

**constant** R0 **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0000"**;**

**constant** R1 **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0001"**;**

**constant** R2 **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0010"**;**

**constant** R3 **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0011"**;**

**constant** R4 **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0100"**;**

**constant** R5 **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0101"**;**

**constant** R6 **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0110"**;**

**constant** R7 **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "0111"**;**

**constant** R8 **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "1000"**;**

**constant** R9 **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "1001"**;**

**constant** R10 **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "1010"**;**

**constant** R11 **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "1011"**;**

**constant** R12 **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "1100"**;**

**constant** R13 **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "1101"**;**

**constant** R14 **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "1110"**;**

**constant** R15 **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** "1111"**;**

--COMANDOS :0

**type** banco **is** **array** **(**0 **to** **(**2**\*\***10**)-**1**)** **of** std\_logic\_vector**(**d**-**1 **downto** 0**);**

**constant** memProg **:** banco **:=** **(**

LI **&** R6 **&** x"0057"**,** --1 LI R6, #87

LI **&** R8 **&** x"005a"**,** --2 LI R8, #90

TR **&** R8 **&** R2 **&** R3 **&** SU **&** ADD**,** --3 ADD R8, R2, R3

TR **&** R1 **&** R2 **&** R3 **&** SU **&** SUB**,** --4 SUB R1, R2, R3

CALL **&** SU **&** x"0009"**,** --5 CALL 0x09

LI **&** R6 **&** x"0057"**,** --6 LI R6, #87

LI **&** R8 **&** x"005a"**,** --7 LI R8, #90

CALL **&** SU **&** x"000d"**,** --8 CALL 13

TR **&** R8 **&** R2 **&** R3 **&** SU **&** ADD**,** --9 ADD R8, R2, R3

TR **&** R1 **&** R2 **&** R3 **&** SU **&** SUB**,** --10 SUB R1, R2, R3

LI **&** R6 **&** x"0057"**,** --11 LI R6, #87

RET **&** SU **&** SU **&** SU **&** SU **&** SU**,** --12 RET

TR **&** R1 **&** R2 **&** R3 **&** SU **&** SUB**,** --13 SUB R1, R2, R3

LI **&** R6 **&** x"0057"**,** --14 LI R6, #87

RET **&** SU **&** SU **&** SU **&** SU **&** SU**,** --15 RET

B **&** SU **&** x"0012"**,** --16 B 18

NOP **&** SU **&** SU **&** SU **&** SU **&** SU**,** --17 NOP

NOP **&** SU **&** SU **&** SU **&** SU **&** SU**,**--18 NOP

B **&** SU **&** x"0011"**,** --19 B 17

**others** **=>** **(others** **=>** '0'**));**

**begin**

Inst **<=** memProg**(**conv\_integer**(**PC**));**

**end** Behavioral**;**

* **Pila\_MemoriaPrograma**

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**entity** Pila\_MemoriaPrograma **is**

**generic** **(** d **:** integer **:=** 25**;**

a **:** integer **:=** 16**;**

s **:** integer **:=** 3**);**

**Port(** PC\_in **:** **in** STD\_LOGIC\_VECTOR **(**a**-**1 **downto** 0**);**

clk**,** clr**,** UP**,** DW**,** WPC **:** **in** STD\_LOGIC**;**

PC\_out **:** **out** STD\_LOGIC\_VECTOR **(**a**-**1 **downto** 0**);**

Inst **:** **out** STD\_LOGIC\_VECTOR **(**d**-**1 **downto** 0**);**

SP **:** **out** STD\_LOGIC\_VECTOR**(**s**-**1 **downto** 0**));**

**end** Pila\_MemoriaPrograma**;**

**architecture** Behavioral **of** Pila\_MemoriaPrograma **is**

**component** Pila **is**

**Port** **(** PC\_in **:** **in** STD\_LOGIC\_VECTOR **(**a**-**1 **downto** 0**);**

PC\_out **:** **out** STD\_LOGIC\_VECTOR **(**a**-**1 **downto** 0**);**

clk**,** clr**,** UP**,** DW**,** WPC **:** **in** STD\_LOGIC**;**

SP**:** **out** std\_logic\_vector**(**S**-**1 **downto** 0**));**

**end** **component;**

**component** MemoriaPrograma **is**

**Port** **(**PC **:** **in** STD\_LOGIC\_VECTOR **(**a**-**1 **downto** 0**);**

Inst **:** **out** STD\_LOGIC\_VECTOR **(**d**-**1 **downto** 0**));**

**end** **component;**

**signal** PC**:** STD\_LOGIC\_VECTOR **(**a**-**1 **downto** 0**);**

**signal** SP\_out**:** STD\_LOGIC\_VECTOR **(**a**-**1 **downto** 0**);**

**begin**

Stack**:** pila

**Port** **map(** PC\_in **=>** PC\_in**,**

clk **=>** clk**,**

clr **=>** clr**,**

UP **=>** UP**,**

DW **=>** DW**,**

WPC **=>** WPC**,**

PC\_out **=>** PC**,**

SP **=>** SP**);**

Memory**:** MemoriaPrograma

**Port** **map(** PC **=>** PC**,**--el segundo PC es la se񡬠:v

Inst **=>** Inst**);**

PC\_out **<=** PC**;**

**end** Behavioral**;**

**Código de Simulación:**

**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** Pila\_MemoriaPrograma\_tb **is**

**end** Pila\_MemoriaPrograma\_tb**;**

**architecture** Behavioral **of** Pila\_MemoriaPrograma\_tb **is**

**component** Pila\_MemoriaPrograma **is**

**Port(** PC\_in **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

clk**,** clr**,** UP**,** DW**,** WPC **:** **in** STD\_LOGIC**;**

PC\_out **:** **out** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

Inst **:** **out** STD\_LOGIC\_VECTOR **(**24 **downto** 0**);**

SP **:** **out** STD\_LOGIC\_VECTOR**(**2 **downto** 0**));**

**end** **component;**

--Inputs

**signal** PC\_in **:** STD\_LOGIC\_VECTOR **(**15 **downto** 0**)** **:=** **(others** **=>** '0'**);**

**signal** clk **:** STD\_LOGIC **:=** '0'**;**

**signal** clr **:** STD\_LOGIC **:=** '0'**;**

**signal** UP **:** STD\_LOGIC **:=** '0'**;**

**signal** DW **:** STD\_LOGIC **:=** '0'**;**

**signal** WPC **:** STD\_LOGIC **:=** '0'**;**

--Outputs

**signal** PC\_out **:** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

**signal** Inst **:** STD\_LOGIC\_VECTOR **(**24 **downto** 0**);**

**signal** SP **:** STD\_LOGIC\_VECTOR**(**2 **downto** 0**);**

-- Clock period definitions

**constant** clk\_period **:** time **:=** 10 ns**;**

**begin**

-- Instantiate the Unit Under Test (UUT)

uut**:** Pila\_MemoriaPrograma **PORT** **MAP** **(**

PC\_in **=>** PC\_in**,**

clk **=>** clk**,**

clr **=>** clr**,**

UP **=>** UP**,**

DW **=>** DW**,**

WPC **=>** WPC**,**

Inst **=>** Inst**,**

PC\_out **=>** PC\_out**,**

SP **=>** SP

**);**

-- Clock process definitions

clk\_process **:process**

**begin**

clk **<=** '0'**;**

**wait** **for** clk\_period**/**2**;**

clk **<=** '1'**;**

**wait** **for** clk\_period**/**2**;**

**end** **process;**

-- Stimulus process

stim\_proc**:** **process**

**file** ARCH\_RES **:** TEXT**;**--Archivo de resultados

**variable** LINEA\_RES **:** line**;**--linea de resultado

**file** ARCH\_VEC **:** TEXT**;**--Archivo de vectores

**variable** LINEA\_VEC **:** line**;**--Linea de vectores

--Variables

**variable** v\_PC\_in**:** STD\_LOGIC\_VECTOR**(**15 **DOWNTO** 0**);**

**variable** v\_SP **:** STD\_LOGIC\_VECTOR **(**2 **downto** 0**);**

**variable** v\_PC\_out **:** std\_logic\_vector**(**15 **downto** 0**);**

**variable** v\_OP\_CODE**:** STD\_LOGIC\_VECTOR**(**4 **DOWNTO** 0**);**

**variable** v\_Rd**,** v\_Rt**,** v\_Rs**,** v\_shamt**,** v\_FUNC\_CODE**:** STD\_LOGIC\_VECTOR**(**3 **DOWNTO** 0**);**

**variable** v\_clr**,** v\_UP**,** v\_DW**,** v\_WPC**:** STD\_LOGIC**;**

--Cadena

**variable** CADENA **:** STRING**(**1 **TO** 6**);**

**begin**

file\_open**(**ARCH\_VEC**,** "VECTORES.txt"**,** READ\_MODE**);**

file\_open**(**ARCH\_RES**,** "RESULTADO.txt"**,** WRITE\_MODE**);**

--Impresi󮠤e Cadenas

CADENA **:=** " SP"**;**

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

CADENA **:=** " PC"**;**

**write(**LINEA\_RES**,** CADENA**,** **right,** CADENA'**LENGTH+**2**);**

CADENA **:=** "OPCODE"**;**

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

CADENA **:=** " Rd"**;**

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

CADENA **:=** " Rt"**;**

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

CADENA **:=** " Rs"**;**

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

CADENA **:=** " Shamt"**;**

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

CADENA **:=** "F\_CODE"**;**

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

**writeline(**ARCH\_RES**,**LINEA\_RES**);**--Escribe la linea en el archivo

--Impresion de Resultados

**wait** **for** 100 ns**;**

**for** i **in** 0 **to** 25 **loop**

--Lectura de cadenas de VECTORES.txt

**readline(**ARCH\_VEC**,** LINEA\_VEC**);**--Lee una linea completa

hread**(**LINEA\_VEC**,** V\_PC\_in**);**

PC\_in **<=** V\_PC\_in**;**

**read(**LINEA\_VEC**,** V\_UP**);**

UP **<=** V\_UP**;**

**read(**LINEA\_VEC**,** V\_DW**);**

DW **<=** V\_DW**;**

**read(**LINEA\_VEC**,** V\_WPC**);**

WPC **<=** V\_WPC**;**

**read(**LINEA\_VEC**,** V\_clr**);**

clr **<=** V\_clr**;**

**wait** **until** **RISING\_EDGE(**CLK**);**

--Asignaci󮠤e Salidas :0

v\_SP **:=** SP**;**

v\_PC\_out **:=** PC\_out**;**

v\_OP\_CODE**(**4**)** **:=** Inst**(**24**);**

v\_OP\_CODE**(**3**)** **:=** Inst**(**23**);**

v\_OP\_CODE**(**2**)** **:=** Inst**(**22**);**

v\_OP\_CODE**(**1**)** **:=** Inst**(**21**);**

v\_OP\_CODE**(**0**)** **:=** Inst**(**20**);**

v\_Rd**(**3**)** **:=** Inst**(**19**);**

v\_Rd**(**2**)** **:=** Inst**(**18**);**

v\_Rd**(**1**)** **:=** Inst**(**17**);**

v\_Rd**(**0**)** **:=** Inst**(**16**);**

v\_Rt**(**3**)** **:=** Inst**(**15**);**

v\_Rt**(**2**)** **:=** Inst**(**14**);**

v\_Rt**(**1**)** **:=** Inst**(**13**);**

v\_Rt**(**0**)** **:=** Inst**(**12**);**

v\_Rs**(**3**)** **:=** Inst**(**11**);**

v\_Rs**(**2**)** **:=** Inst**(**10**);**

v\_Rs**(**1**)** **:=** Inst**(**9**);**

v\_Rs**(**0**)** **:=** Inst**(**8**);**

v\_shamt**(**3**)** **:=** Inst**(**7**);**

v\_shamt**(**2**)** **:=** Inst**(**6**);**

v\_shamt**(**1**)** **:=** Inst**(**5**);**

v\_shamt**(**0**)** **:=** Inst**(**4**);**

v\_FUNC\_CODE**(**3**)** **:=** Inst**(**3**);**

v\_FUNC\_CODE**(**2**)** **:=** Inst**(**2**);**

v\_FUNC\_CODE**(**1**)** **:=** Inst**(**1**);**

v\_FUNC\_CODE**(**0**)** **:=** Inst**(**0**);**

--Escritura de Resultados

Hwrite**(**LINEA\_RES**,** v\_SP**,** **right,** 7**);**

Hwrite**(**LINEA\_RES**,** v\_PC\_out**,** **right,** 7**);**

**write(**LINEA\_RES**,** v\_OP\_CODE**,** **right,** 7**);**

**write(**LINEA\_RES**,** v\_Rd**,** **right,** 7**);**

**write(**LINEA\_RES**,** v\_Rt**,** **right,** 7**);**

**write(**LINEA\_RES**,** v\_Rs**,** **right,** 7**);**

**write(**LINEA\_RES**,** v\_shamt**,** **right,** 7**);**

**write(**LINEA\_RES**,** v\_FUNC\_CODE**,** **right,** 7**);**

**writeline(**ARCH\_RES**,**LINEA\_RES**);**--Escribe la linea en el archivo

**end** **loop;**

file\_close**(**ARCH\_VEC**);**--Cierra el archivo

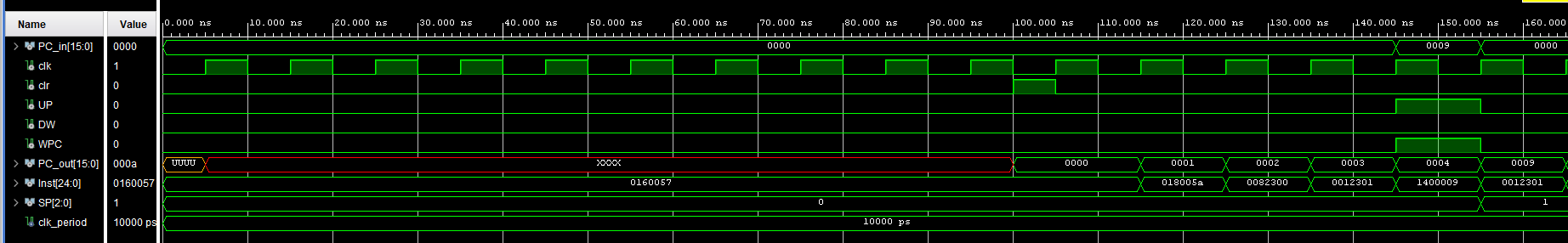
file\_close**(**ARCH\_RES**);**--Cierra el archivo

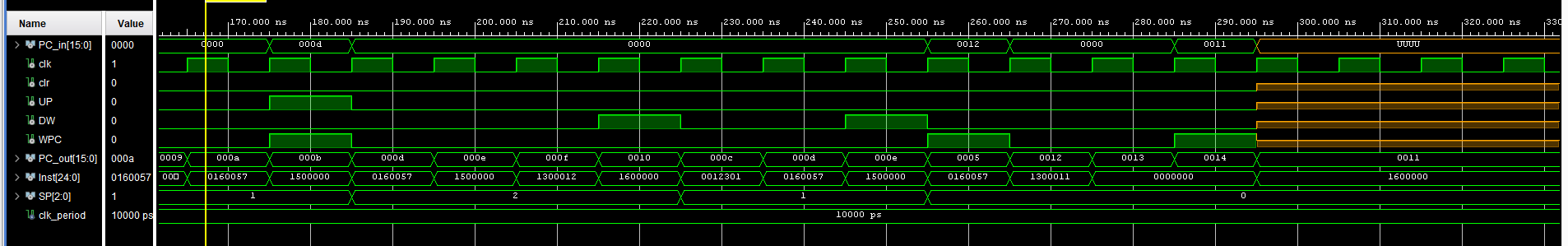
**wait;**

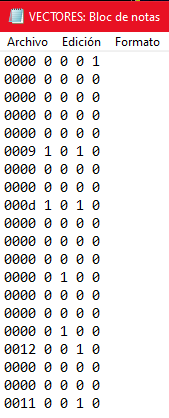
**end** **process;**-- Stimulus process

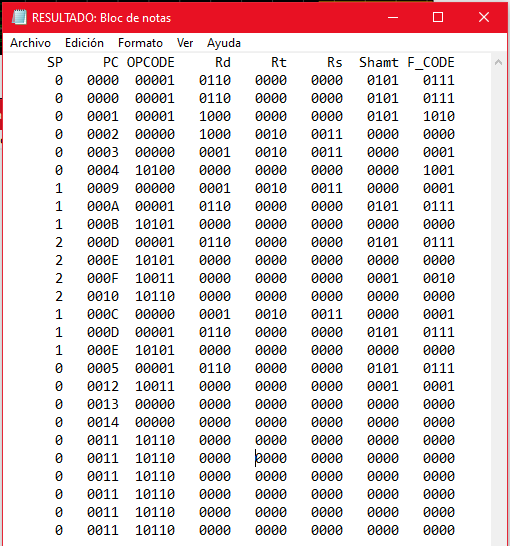
**end** Behavioral**;**

**Simulación:**





**Archivo de entrada: Archivo de salida:**



**Diagrama RTL:**

