

# Unidad Aritmética Lógica en Arty Z7-10

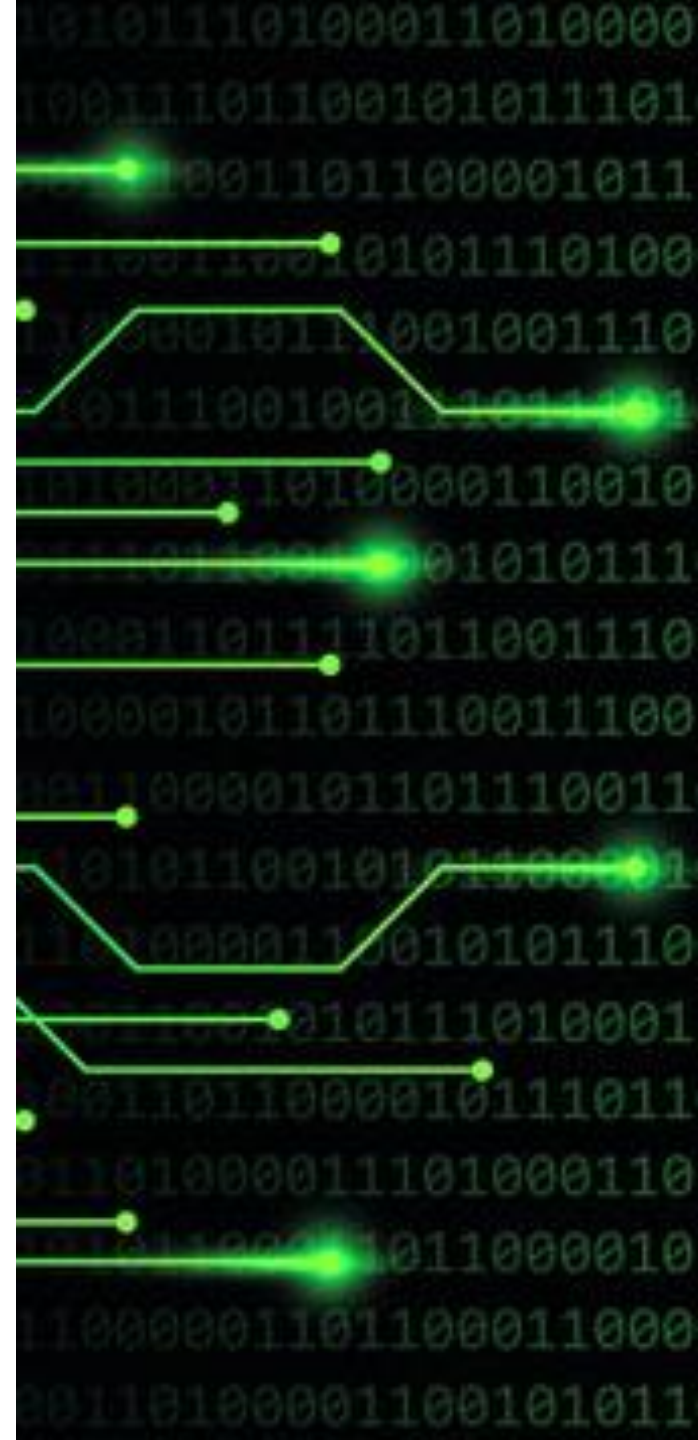
MARCOS DOMINGUEZ  
CESE 15Co - 2022

APLICACIÓN

# Descripción de la aplicación

---

Se implementó una ALU de 4 bits con 14 instrucciones. El código VHDL permite escalar la ALU a cualquier número par de bits



# Instrucciones

01

## **SUMA**

Suma dos operandos con carries.

02

## **RESTA**

Resta dos operandos con carries.

03

## **MULTIPLICACION**

Multiplica dos operandos de tamaño ALU bits / 2.

04

## **DESPLAZAMIENTO NO SIGNADO A LA IZQUIERDA**

Desplazamiento a la izquierda con carries

05

## **DESPLAZAMIENTO NO SIGNADO A LA DERECHA**

Desplazamiento a la derecha con carries

06

## **DESPLAZAMIENTO SIGNADO A LA IZQUIERDA**

Desplazamiento a la izquierda con carries

# Instrucciones

07

**DESPLAZAMIENTO  
SIGNADO A LA  
DERECHA**

Desplazamiento a la derecha. El carry in se usa como indicador de signo

08

**AND**

Operación and entre operandos

09

**OR**

Operación or entre operandos

10

**XOR**

Operación xor entre operandos

11

**NAND**

Operación nand entre operandos

12

**NOR**

Operación nor entre operandos

# Instrucciones

13

**NXOR**

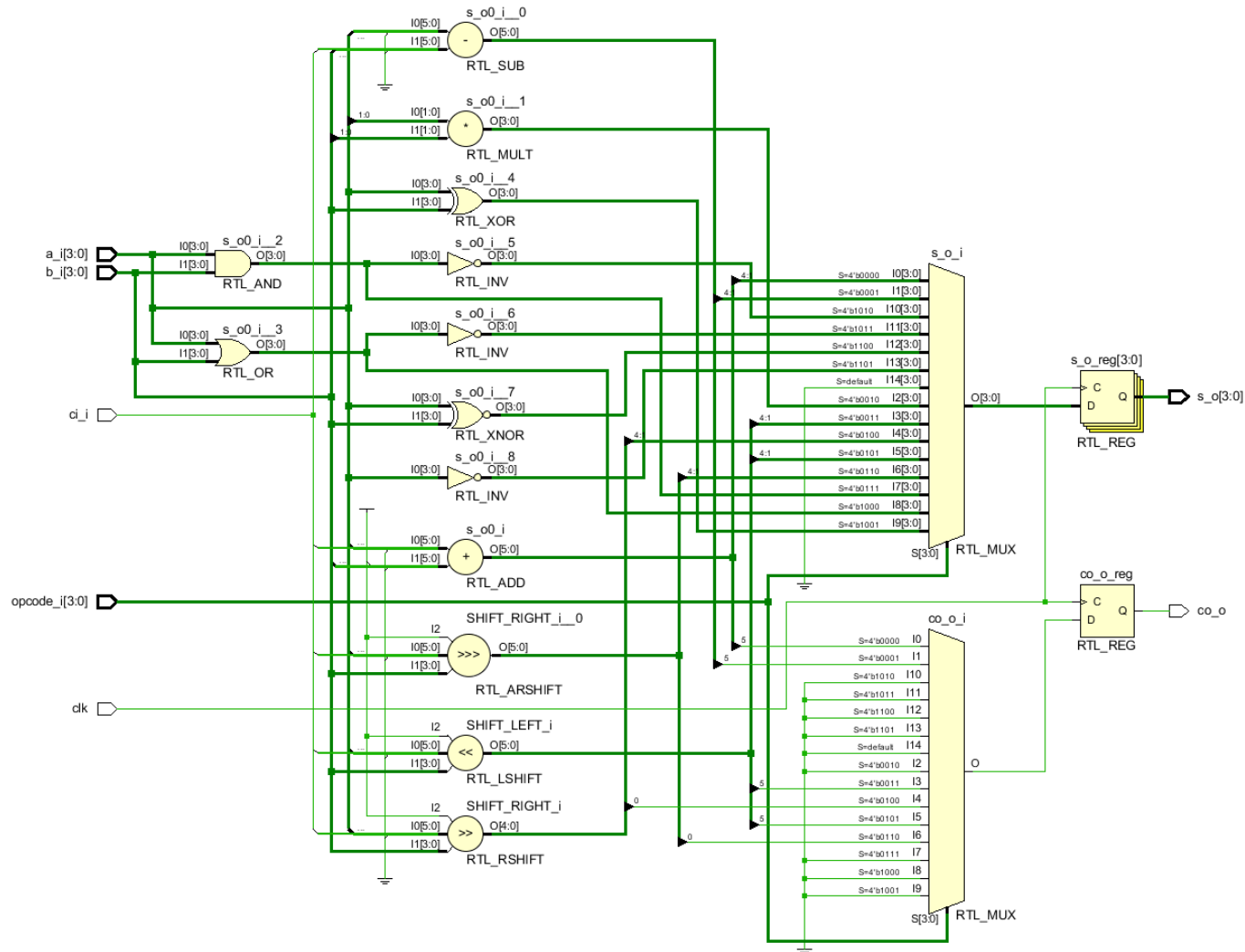
Operación nxor entre  
operandos

14

**NOT**

Operación not del  
primer operando

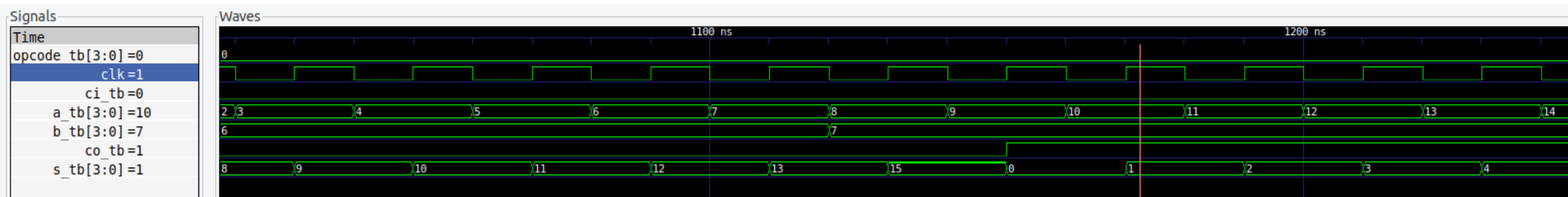
# Esquemático



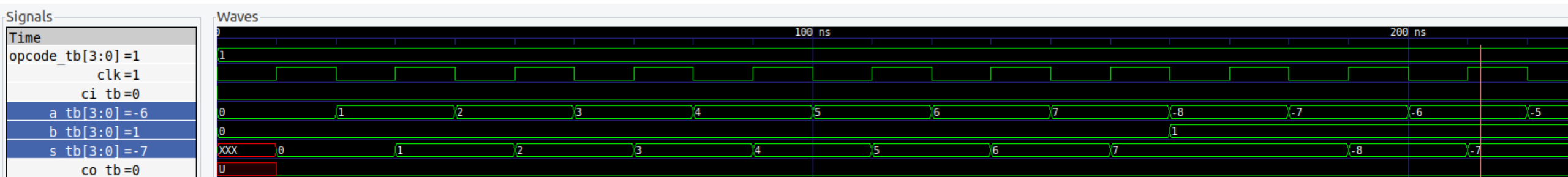
```
entity alu is
  generic (
    N : natural := 4
  );
  port (
    clk : in std_logic;
    a_i : in std_logic_vector(N - 1 downto 0);
    b_i : in std_logic_vector(N - 1 downto 0);
    ci_i : in std_logic;
    opcode_i : in std_logic_vector(3 downto 0);
    s_o : out std_logic_vector(N - 1 downto 0);
    co_o : out std_logic
  );
end;
```

# SIMULACIONES

## S U M A

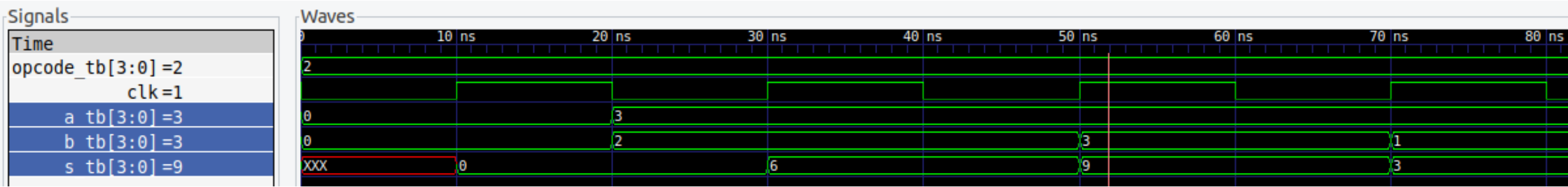


## RESTA

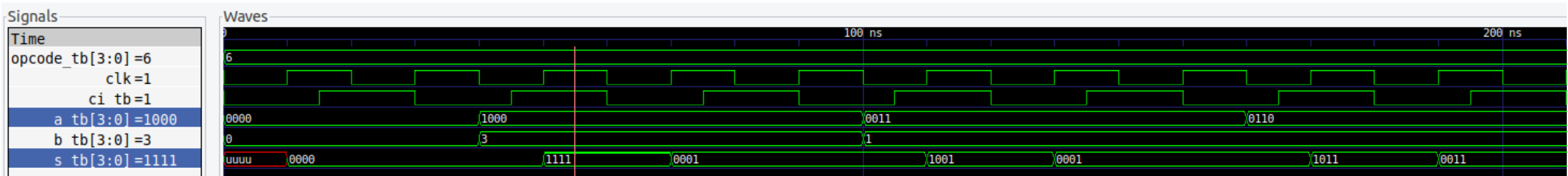


# SIMULACIONES

## MULTIPLICACIÓN



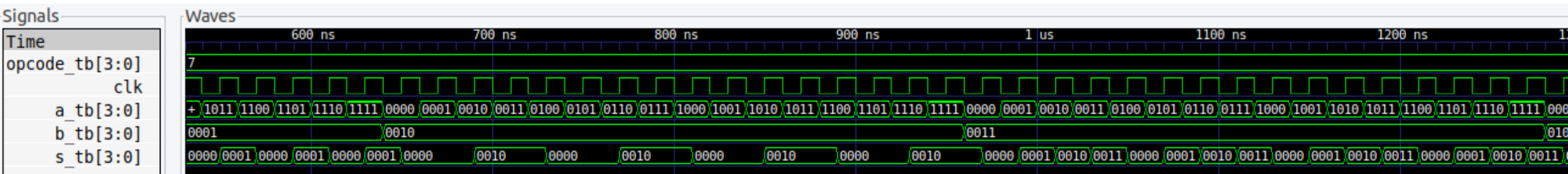
## DESPLAZAMIENTO
















# SIMULACIONES

**AND**



# Uso del VIO

hw_vio_1				
    				
Name	Value	Activity	Direction	VIO
 co_tb	[B] 0		Input	hw_vio_1
>  s_tb[3:0]	[B] 1110		Input	hw_vio_1
>  opcode_i[3:0]	[B] 0011 ▾		Output	hw_vio_1
>  b_i[3:0]	[U] 2 ▾		Output	hw_vio_1
>  a_i[3:0]	[B] 0011 ▾		Output	hw_vio_1
 ci_i	[B] 1 ▾		Output	hw_vio_1



Preguntas