



Universidad  
Nacional  
de Córdoba

UNIVERSIDAD NACIONAL DE CÓRDOBA  
Facultad de Ciencias Exactas, Físicas y Naturales

Cátedra de Arquitectura de Computadoras

**Trabajo Práctico Nro. 1: ALU**

*Fernández Oria, Luciano*

[luchoof212@gmail.com](mailto:luchoof212@gmail.com)

*Sieber, Braian*

[braiansieber@gmail.com](mailto:braiansieber@gmail.com)

10/11/2018

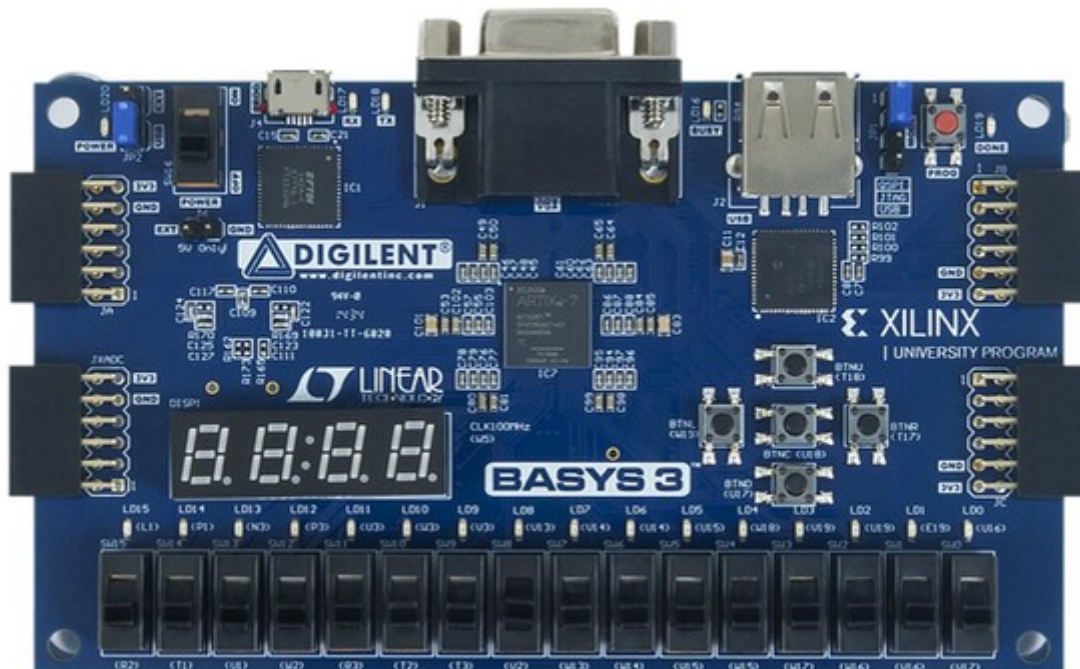
## CONSIGNA:

En este trabajo práctico se nos solicitó desarrollar una ALU que permitiera ejecutar múltiples instrucciones seleccionadas por una entrada. El módulo cuenta con 3 entradas, dos de 8 bits, extensibles a 32 bits y una de 6 bits. Las entradas de 8 bits corresponden a los dos operandos de la función que realiza la ALU, mientras que la entrada de 6 bits se utiliza para seleccionar la operación a realizar.

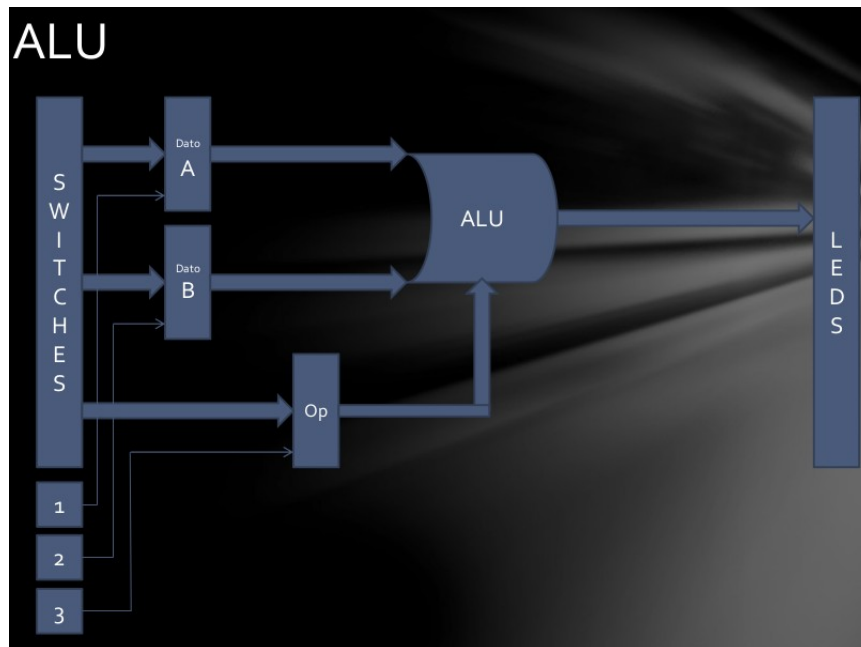
## Requerimientos:

- Implementar en FPGA una ALU.
- Utilizar las placas de desarrollo Basys 3.
- La ALU debe ser parametrizable (bus de datos) para poder ser utilizada posteriormente en el trabajo final.
- Validar el desarrollo por medio de Test Bench

Para probar el modulo utilizamos la placa de desarrollo Basys 3 que cuenta con una FPGA Artix 7 de la empresa Xilinx, ingresando los valores en los switches.



La ALU conciste en un circuito combinacional, donde las entradas son los valores que ingresamos a traves de los switchs y utilizamos los botones para definir ambos operandos y el tipo de operación.



Este módulo toma los operandos A y B, realiza una operación indicada por la entrada Op, y pone el resultado en la salida Leds.

Las operaciones que deberá soportar la ALU son las siguientes:

Operación	Código
ADD	100000
SUB	100010
AND	100100
OR	100101
XOR	100110
SRA	000011
SRL	000010
NOR	100111

Se simularon las distintas operaciones de la ALU.

```
`timescale 1ns / 1ps

module TestALU;

    // Inputs
    reg [5:0] Op;
    reg [7:0] A;
    reg [7:0] B;

    // Outputs
    wire [7:0] Leds;

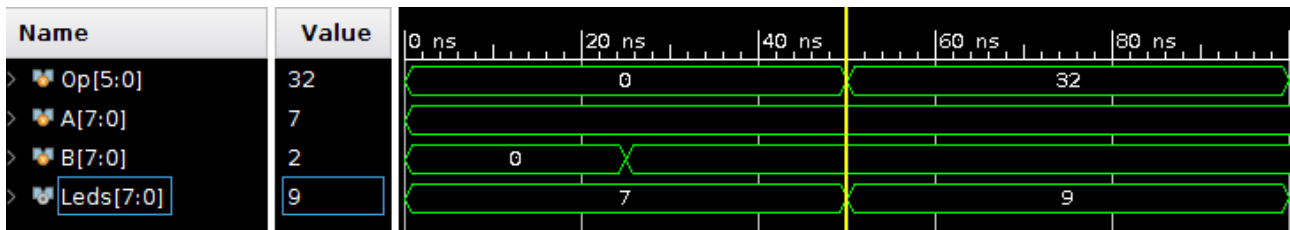
    // Instantiate the Unit Under Test (UUT)
    ALU uut (
        .Op(Op),
        .A(A),
        .B(B),
        .Leds(Leds)
    );

    initial begin
        // Initialize Inputs
        Op = 0;
        A = 0;
        B = 0;
```

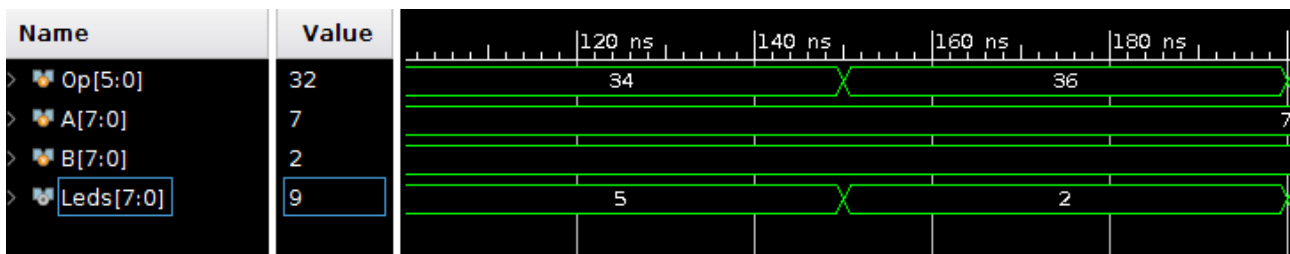
Ingresamos valores y probamos las diferentes operaciones:

```
        // Wait 100 ns for global reset to finish
        // Add stimulus here
        A=7;
        #25;
        B=2;
        #25;
        Op=32; //suma
        #50;
        Op=34; //resta
        #50;
        Op=36; //and
        #50;
        Op=37; //or
        #50;
        Op=38; //xor
        #50;
        A=128;
        Op=3; //sra
        #50;
        Op=2; //srl
        #50;
        A=7;
        Op=39; //nor
        #50;
        B=9;
        Op=34;
        #50;
```

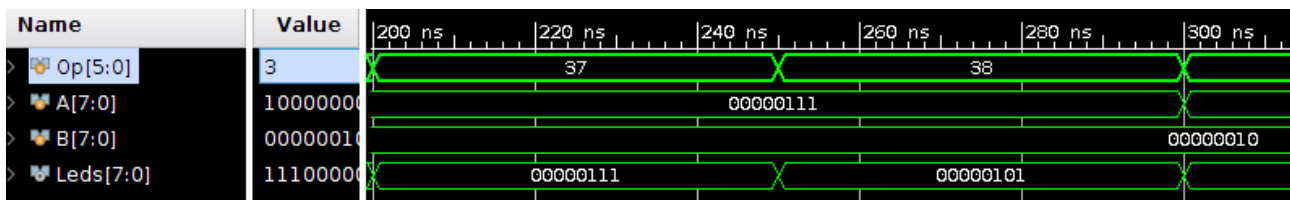
En la siguiente imagen se observa en la salida el valor de A y la operacion ADD:



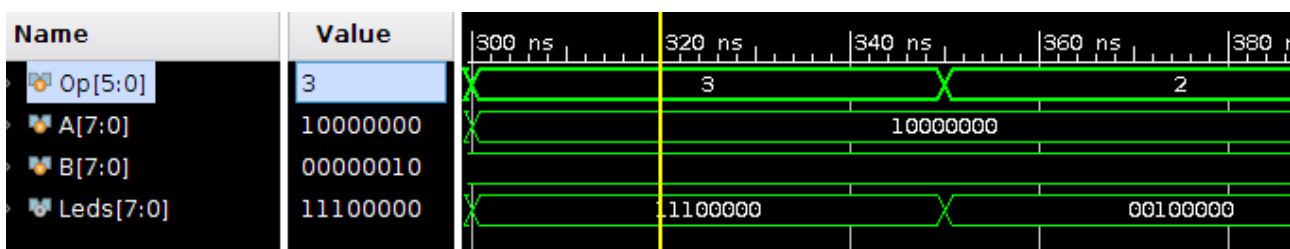
SUB y AND:



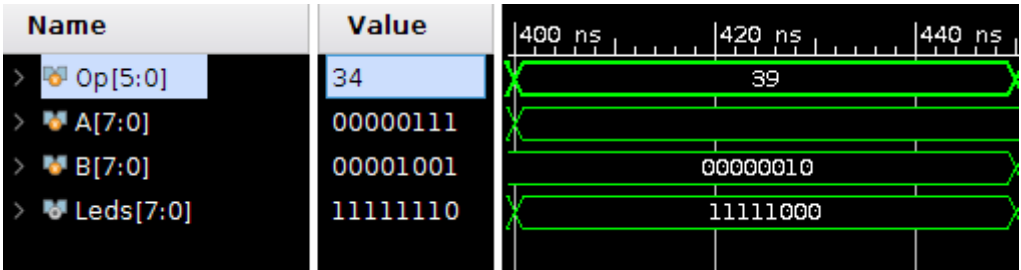
OR y XOR



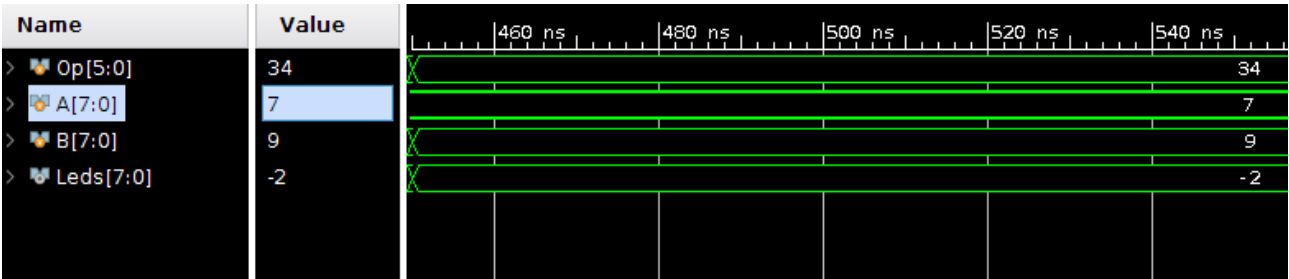
SRA y SRL:



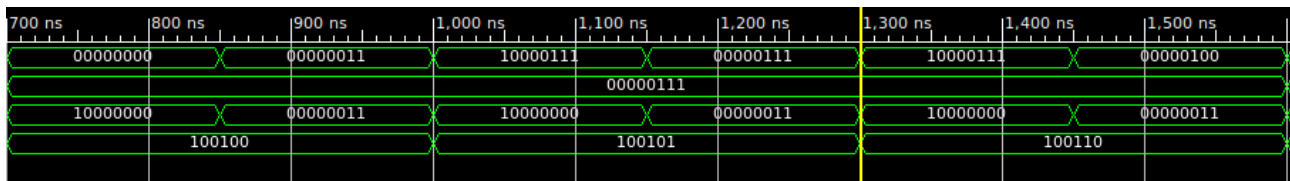
NOR



Cambiamos el valor de B para que el resultado de la resta de un numero negativo:



Las operaciones AND, OR y XOR:

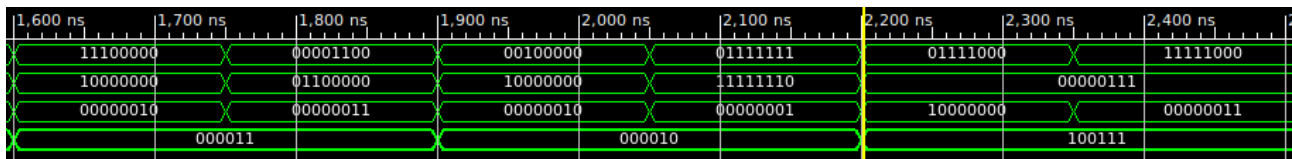


```
//-----AND-----//
    entrada1=7;
    entrada2=128;
    op=6'b100100;
    #150;
    entrada1=7;
    entrada2=3;
    #150;

//-----OR-----//
    entrada1=7;
    entrada2=128;
    op=6'b100101;
    #150;
    entrada1=7;
    entrada2=3;
    #150;

//-----XOR-----//
    entrada1=7;
    entrada2=128;
    op=6'b100110;
    #150;
    entrada1=7;
    entrada2=3;
    #150;
```

Y por ultimo se observan las operaciones SRA, SRL y NOR:



```
//-----SRA-----//
    entrada1=128;
    entrada2=2;
    op=6'b000011;
    #150;
    entrada1=96;
    entrada2=3;
    #150;

//-----SRL-----//
    entrada1=128;
    entrada2=2;
    op=6'b000010;
    #150;
    entrada1=254;
    entrada2=1;
    #150;

//-----NOR-----//
    entrada1=7;
    entrada2=128;
    op=6'b100111;
    #150;
    entrada1=7;
    entrada2=3;
    #150;
end
endmodule
```

La prueba en la placa se realizó de la siguiente manera. Por el bus Entrada se coloca un valor. Luego se pone en 1 la entrada switch1 que escribe el valor que hay en la entrada a la salida del registro1, y así con el registro2 y registro3. Esto se hace de manera sincrónica en cada flanco positivo de reloj. La salida está dada por un circuito combinacional que depende de los valores de RegistroA, RegistroB y RegistroC.