



Actividad 02

Unidad Lógica – Implementación de Compuertas Lógicas

Emilio Gomez Breschi

Introducción

Introducción

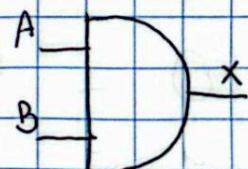
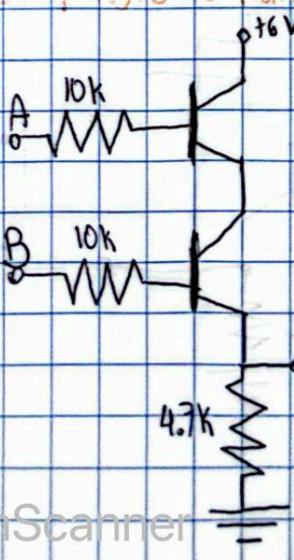
La Unidad Aritmético Lógico es la que se encarga de realizar operaciones aritméticas y lógicas. Dentro de la unidad, existen módulos que sirven para comparar y operar al igual que tomar decisiones lógicas.

Las FPGAs en donde sus siglas significan Field Programmable Gate Arrays, estos son dispositivos reconfigurables que permiten implementar circuitos digitales. Verilog se relaciona con las FPGAs ya que el código escrito describe el como debe interconectarse las compuertas lógicas y bloques funcionales en el dispositivo.

AND

Tabla de Verdad, Freglo de Transistores, Símbología

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1



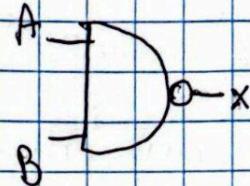
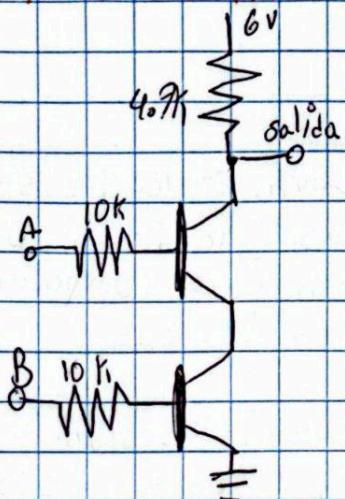
Sintaxis

assign y = a & b;
and(y,a,b)

NAND

Tabla de Verdad Arreglo de Transistores Símbología

A	B	X
0	0	1
0	1	1
1	0	1
1	1	0



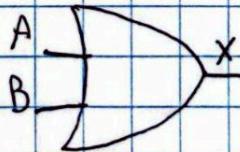
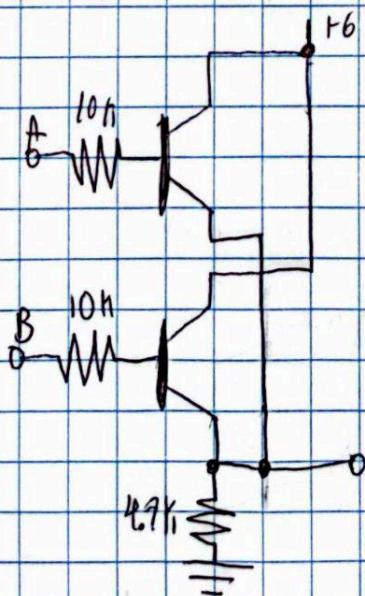
Sintaxis

- assign $y = \sim(a \otimes b)$
- nand (y, a, b)

OR

Tabla de Verdad Arreglo de Transistores Símbología

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1



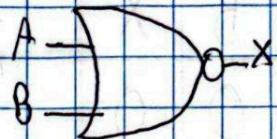
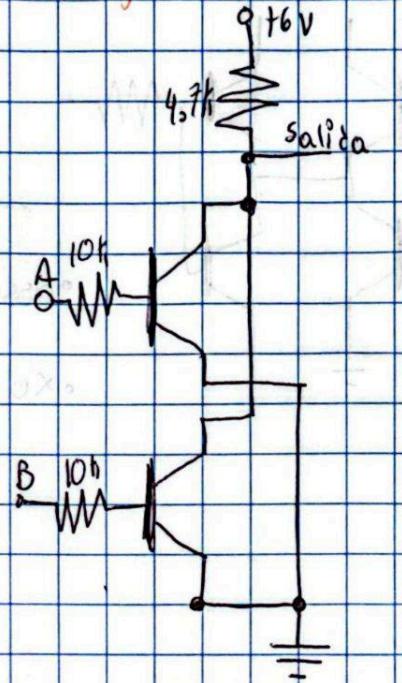
Sintaxis

- assign $y = a \mid b$
- or (y, a, b)

NOR

Tabla de Verdad Arreglo de Transistores Símbología

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0



Sintaxis

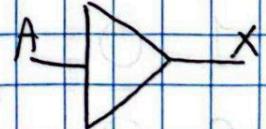
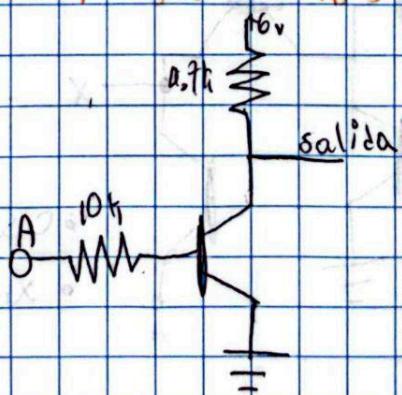
• assign g = ~(a | b)

• not(g, a, b)

NOT

Tabla de Verdad Arreglo de Transistores Símbología

A	X
0	1
1	0



Sintaxis

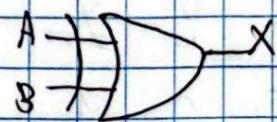
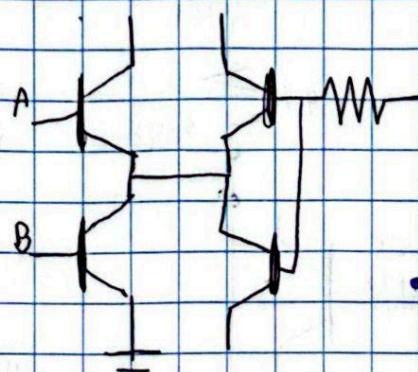
• assign g = ~a;

• not(g, a)

XOR

Tabla de Verdad Alrededor de Transistores Símbolos

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0



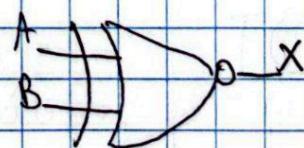
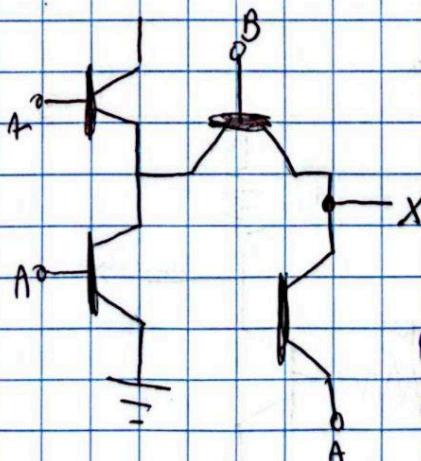
Sintaxis

- assign $y = a \oplus b;$
- xor ($y, a, b);$

XNOR

Tabla de Verdad Alrededor de Transistores Símbolos

A	B	X
0	0	1
0	1	0
1	0	0
1	1	1



Sintaxis

- assign $y = \sim(a \oplus b);$
- xnor ($y, a, b)$

Objetivos

Objetivo general

Comprobar la correcta instalación y funcionamiento de ModelSim mediante la programación y simulación de compuertas lógicas usando el lenguaje de Verilog.

Objetivos específicos

- Investigar el funcionamiento de las compuertas lógicas básicas.
- Analizar la sintaxis de las compuertas lógicas en el lenguaje Verilog.
- Implementar un módulo a nivel de 1 bit con dos entradas y siete salidas.
- Simular el comportamiento de cada compuerta.
- Familiarizarse con el flujo de simulación en ModelSim.

Desarrollo

Para el desarrollo de esta práctica se utilizó el lenguaje Verilog, el cual permite describir circuitos digitales de manera estructurada y clara. Se implementó un solo módulo que contiene dos entradas binarias (*i_bit1* e *i_bit2*) y siete salidas que son las compuertas lógicas.

Cada compuerta fue implementada utilizando los operadores lógicos de Verilog, los cuales permiten describir directamente el comportamiento del circuito. La compuerta NOT utiliza únicamente una de las entradas, mientras que las otras compuertas operan con dos entradas.

El módulo fue simulado en ModelSim, donde se pusieron diferentes combinaciones de entrada para comprobar el comportamiento esperado de cada salida. La simulación permitió observar las ondas, verificando que las salidas coinciden con las tablas de verdad.

The screenshot shows the ModelSim - INTEL FPGA STARTER EDITION 2020.1 interface. On the left, the 'Source' window displays the Verilog code for 'Actividad02.v'. The code defines a module 'Actividad02' with two inputs ('i_bit1', 'i_bit2') and seven outputs ('o_and', 'o_nand', 'o_or', 'o_nor', 'o_xor', 'o_nxor', 'o_not'). It uses the 'assign' statement to map input combinations to output values based on Verilog's built-in functions. On the right, the 'Wave' window shows a waveform for 'Actividad02_b11' over time. The waveform has seven data channels labeled 'Actividad02_o_and' through 'Actividad02_o_nor'. The 'o_and' channel shows a constant high value. The 'o_nand' channel shows a constant low value. The 'o_or' channel shows a constant high value. The 'o_nor' channel shows a constant low value. The 'o_xor' channel shows a constant high value. The 'o_nxor' channel shows a constant low value. The 'o_not' channel shows a constant high value. The time axis at the bottom ranges from 0.0 ns to 0.5 ns, with a cursor at 0.039 ns.

```
ModelSim - INTEL FPGA STARTER EDITION 2020.1
File Edit View Compile Simulate Add Project Tools Layout Bookmarks Window Help
C:\Users\lime1\Documents\CUCEI\Cuarto Semestre\Arquitectura\Actividad02\Actividad02.v - Default
Name Status Type Order
Actividad02.v Verilog 0
1
2 module Actividad02(
3   input i_bit1,
4   input i_bit2,
5   output o_and,
6   output o_nand,
7   output o_or,
8   output o_nor,
9   output o_xor,
10  output o_nxor,
11  output o_not
12 );
13
14   assign o_and = i_bit1 & i_bit2;
15   assign o_nand = ~(i_bit1 & i_bit2);
16   assign o_or = i_bit1 | i_bit2;
17   assign o_nor = ~i_bit1 | ~i_bit2;
18   assign o_xor = i_bit1 ^ i_bit2;
19   assign o_nxor = ~(i_bit1 ^ i_bit2);
20
21 endmodule
22
```

Conclusión

La realización de esta práctica me hizo comprender el funcionamiento de las compuertas lógicas y su implementación mediante el lenguaje de verilog. Se aprendieron conceptos de lógica digital y experiencia en el uso de herramientas de simulación como ModelSim.

Aunque la práctica es fácil, es muy importante para entender el funcionamiento interno de sistemas más difíciles como procesadores y compuertas lógicas. Además, permitió identificar la importancia de la simulación previa antes de una implementación física.

Referencias

Compuertas lógicas – Sistemas Digitales. (s. f.).

<https://virtual.cuautitlan.unam.mx/intar/sistdig/compuertas-logicas/>

TecNM. (s. f.). *TECNM | Tecnológico Nacional de México.* TecNM.

https://www.tecnm.mx/images/areas/docencia01/Apuntes/L%C3%B3gica_Digital.pdf

Wtdtantakatan. (2016, 21 septiembre). *Transistores en compuertas lógicas.* Tantakatanblog.

<https://tantakatanblog.wordpress.com/2016/09/18/transistores/>