**INSTITUTO POLITÉCNICO NACIONAL**

**USO DE GALAXY**

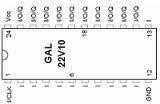
# ESCUELA SUPERIOR DE CÓMPUTO

### DEPARTAMENTO DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

**PRACTICA No. 4**

USO DE GALAXY





### OBJETIVO GENERAL:

Al finalizar la práctica, el alumno aprenderá a programar un dispositivo lógico programable (PLD). Para esto estudiará y aprenderá a utilizar las herramientas adecuadas para dicho objetivo. Así mismo, reafirmará el conocimiento adquirido en clase al realizar un programa que ejecute una función específica, sobre el dispositivo utilizado, y comprobar físicamente el correcto funcionamiento del programa desarrollado. Finalmente, sabrá lo que es un PLD y como programar una función específica sobre el dispositivo, utilizando un lenguaje de descripción de hardware (HDL).

### OBJETIVOS ESPECÍFICOS:

1. Distinguir los beneficios proporcionados por las herramientas CAD.
2. Confirmar el conocimiento adquirido en clase y realizar un programa en un HDL que se ejecute sobre un PLD.
3. Comprobar físicamente el correcto funcionamiento del sistema diseñado, el cual será implementando sobre un proto-board.

### MATERIAL Y EQUIPO EMPLEADO.

* + Fuente de 5V.
  + Circuito Integrado GAL22V10.
  + DIP switch de 8.
  + 6 Resistencias de 1KΩ.
  + 1 Resistencia de 330Ω.
  + 1 Led.
  + Tablilla de Prueba (proto-board).
  + Alambre telefónico.
  + Pinzas de punta.
  + Pinzas de corte.
  + Cables Banana-Caimán.
  + Programador universal.

1. **DESARROLLO EXPERIMENTAL.**

El problema propuesto es simple, ya que la finalidad es relacionarse con las herramientas CAD, comprender la mecánica del proceso de diseño utilizando PLD’s y realizar la implementación física de la función deseada.

De esta manera, la práctica consiste en programar el código para implementar las ya conocidas compuertas lógicas en la GAL, usando VHDL. El circuito tendrá 4 entradas y 1 salida, las cuales corresponden a las compuertas lógicas; y debe mostrar a la salida el resultado de las operaciones lógicas ya conocidas.

## EDICIÓN DEL PROGRAMA.

Los pasos a seguir para trabajar con Galaxy se describen a continuación:

1. Ejecutar el software Galaxy (**Inicio→ Todos los programas→ Cypress→ Galaxy**).
2. Una vez en la interfaz del programa, se crea un proyecto de la siguiente manera:
3. **File**→ **New**→ **Project [Target-Device].** Esto abre una ventana, como se muestra en la figura 1, en donde se introduce la ruta en que se salvará el proyecto, el nombre de este (**compuertas para este caso**) y además se selecciona el lenguaje que se va a utilizar (VHDL o Verilog HDL). Se selecciona VHDL.

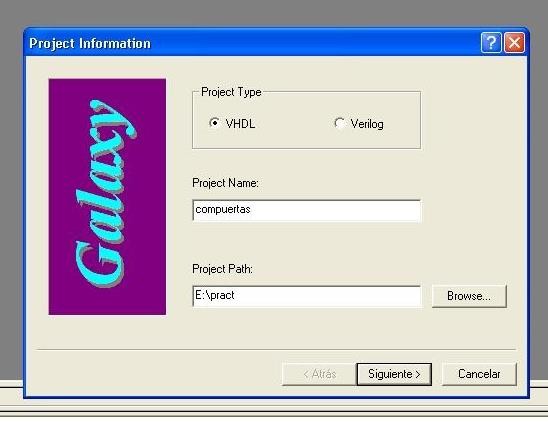


Figura 1. Nombre del proyecto (*compuertas*) y ruta.

1. Se da clic en **<siguiente>** y aparece una nueva ventana que da la opción para agregar el archivo

***.vhd***, lo cual haremos después, por lo que se da clic en **<siguiente>**, lo que abre una nueva ventana que permite seleccionar el dispositivo a utilizar (ver figura 2). En esta práctica se utilizará un dispositivo del tipo GAL22V10, que es un **SPLD**, localizado dentro de **C22V10** y se selecciona el PALCE22V10-25PC ó -15PC, según sea su retardo de programación (**speed (ns)**) y se da clic en

### <Finalizar>.

1. Después aparece una ventana que pregunta si se quiere salvar el nuevo proyecto, y se selecciona

### <si>.

1. Para crear un archivo de texto, se hace mediante: **File→ New→ Text File** y después **<OK>**.

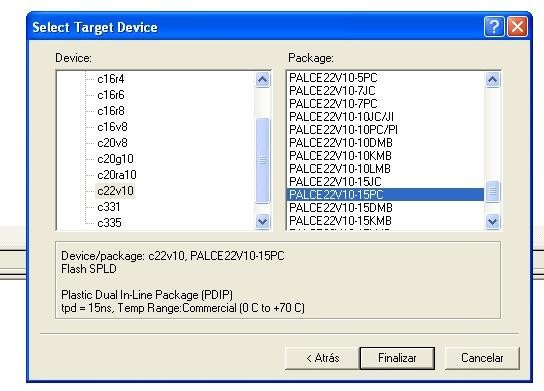


Figura 2. Selección del tipo de dispositivo.

1. Se escribe el código que se muestra en la figura 3 y después se salva el archivo, preferentemente con el nombre de la ENTIDAD, con terminación **\*.vhd** (en este caso se llama **and*.vhd***), para posteriormente agregarlo al proyecto haciendo clic en **Project → Add Files**, como se muestra en la figura 4.

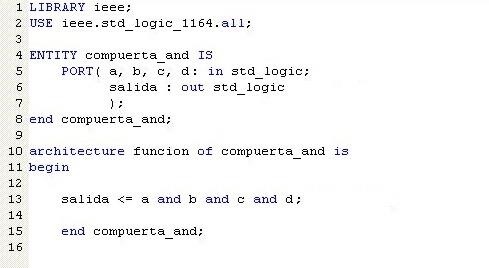


Figura 3. Código del programa en VHDL.

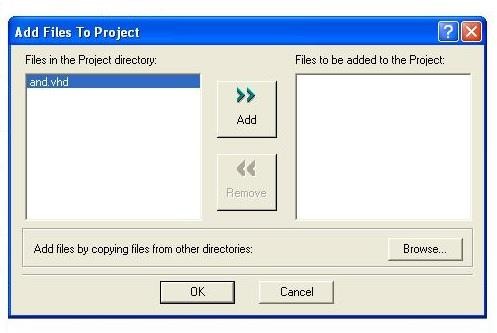


Figura 4. Agregar el archivo .vhd al proyecto.

1. En la figura 5 se puede observar que la ventana izquierda cambia. Esto es debido al hecho de que ya es un proyecto. Se da clic con el botón derecho del ratón sobre el icono del archivo creado **and.vhd** y se selecciona **Set Top**. Se puede observar, en la figura 5b, que después de esto cambia dicho icono.

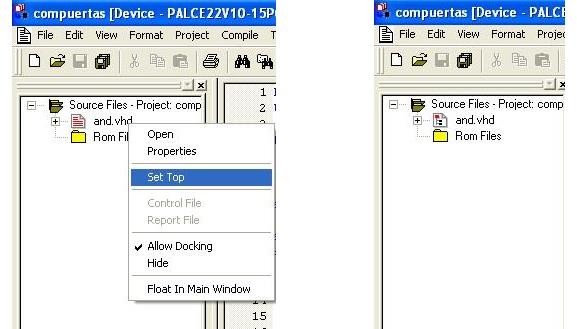


Figura 5 a) y b).

1. Una vez hecho esto, se compila el archivo en: **Compile→ Selected File(s)**. En la parte inferior de la ventana se despliega la información de la compilación. En caso de que esta no fuera satisfactoria, es posible ver los errores o advertencias. Para esto es necesario dirigirse a la pestaña de nombre **Errors & Warnings** que está localizada en la parte inferior de la ventana de trabajo.
2. Para corroborar el funcionamiento correcto del programa es posible realizar su simulación. Para esto, debemos ir a: **Tool→ Active-HDL Sim**, abriéndose la ventana que se muestra en la figura 6.
3. Se abre el archivo **and*.vhd***, esto en: **File→ Open VHDL file for simulation** y se selecciona el archivo, localizado en la carpeta ***vhd***, ***creada por el proyecto***. Se da **<Abrir>**. Al hacer eso se compila el programa, apareciendo comentarios en la consola de la ventana de simulación.

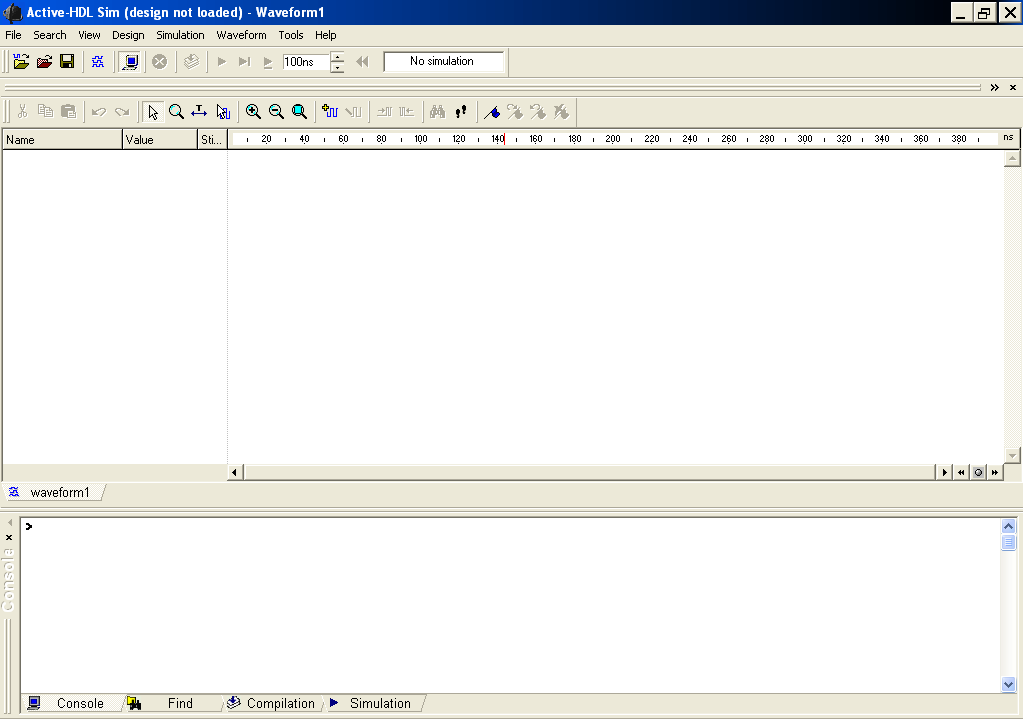


Figura 6. Interfaz de simulación.

1. Para visualizar las señales a simular es necesario agregarlas a la ventana de simulación. Para esto seleccionamos **Waveform** **Add signal**, como se muestra en la figura 7a. Al hacer esto, se abre la ventana que se muestra en la figura 7b, donde se seleccionan las señales a monitorear; en este caso, las entradas a, b, c, d y la salida.

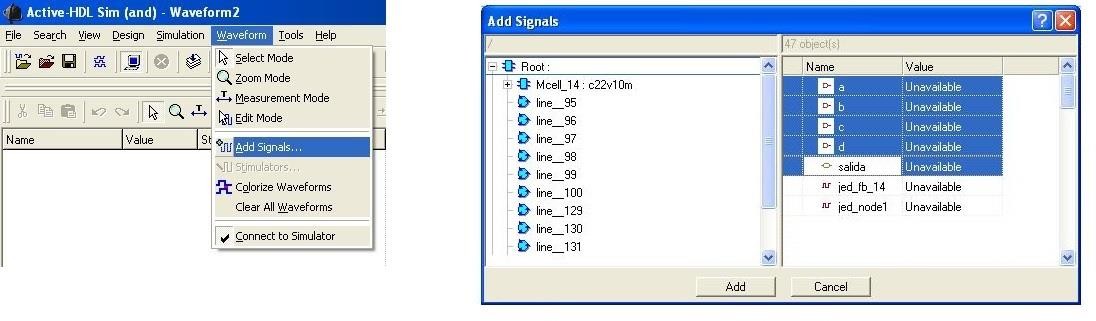


Figura 7 a) y b). Agregar señales a la simulación.

1. En el caso de las entradas es necesario asignar valores para la simulación. Esto se hace dando clic con el botón derecho sobre las señales. Al hacer esto se abre un menú desplegable y se da clic en **Stimulators**, abriéndose una nueva ventana, que se muestra en la figura 8.

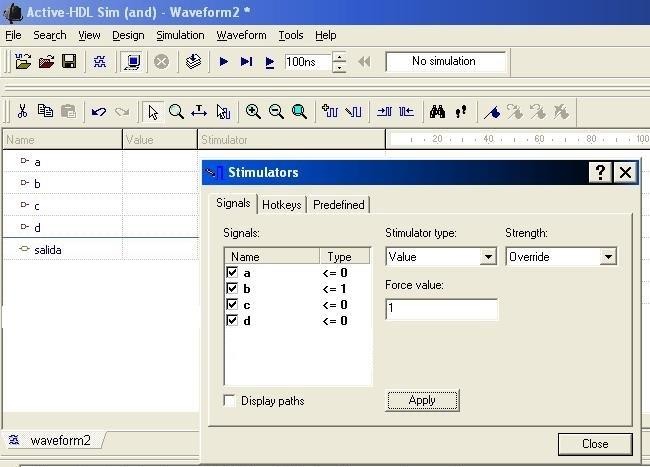


Figura 8. Asignación de valores a las señales de entrada.

1. En esta ventana se indican el tipo de estímulo, en este caso elegimos **Stimulator type: value, Strength: Override, Force value:** (valor que deseamos asignar, el cual será **0’s** o **1’s** para las entradas a, b, c, y d). Para que tenga efecto se da clic en **<Apply>**.
2. Para observar la simulación con los estímulos asignados es necesario correr el tiempo de la simulación. Esto se hace en: **Simulation** **Run for**. Al hacer esto se muestran los valores de entrada y se generan los valores de la salida, como se puede ver en la figura 9. Se verifica si los valores obtenidos son los deseados.

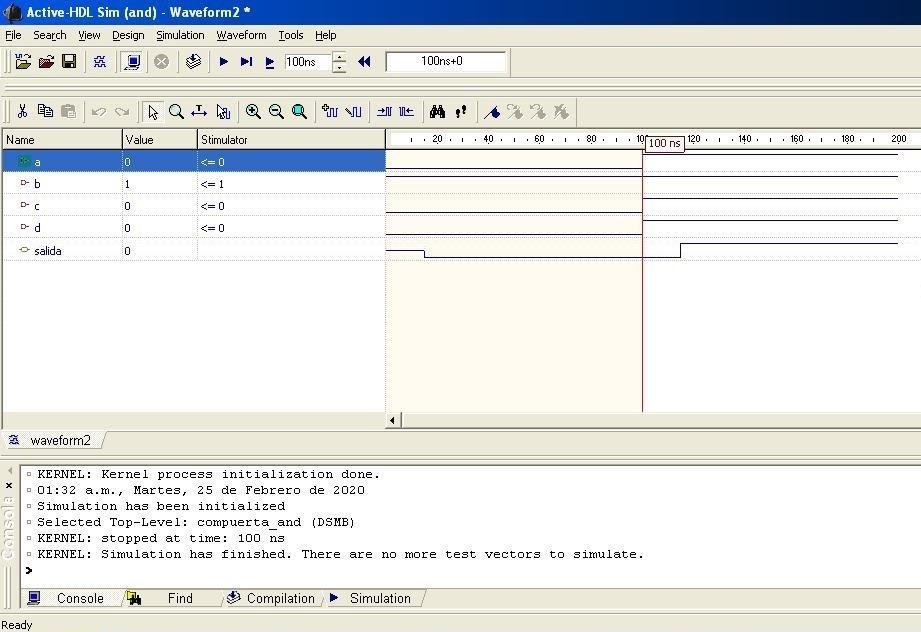


Figura 9. Simulación.

1. Si se desea hacer una nueva simulación con otro valor u otro estimulo de entrada, es necesario asignar el nuevo valor y correr otra vez la simulación.
2. Una vez que se ha revisado el correcto funcionamiento del sistema diseñado mediante la simulación, es necesario saber en qué terminales se encuentra cada señal de entrada/salida. El sitio donde aparece esta información es el reporte de salida ***Output Files***, localizado en la ventana izquierda del proyecto, dando clic derecho en el archivo **and*.vhd***. Para acceder al archivo de reporte se tiene que ir a la pestaña localizada en la parte inferior de la misma ventana, como se muestra en la figura 10. Dando clic en el archivo **and*.rpt*** se pueden observar las terminales asignadas a cada señal y el uso del dispositivo, como muestra la figura 11.

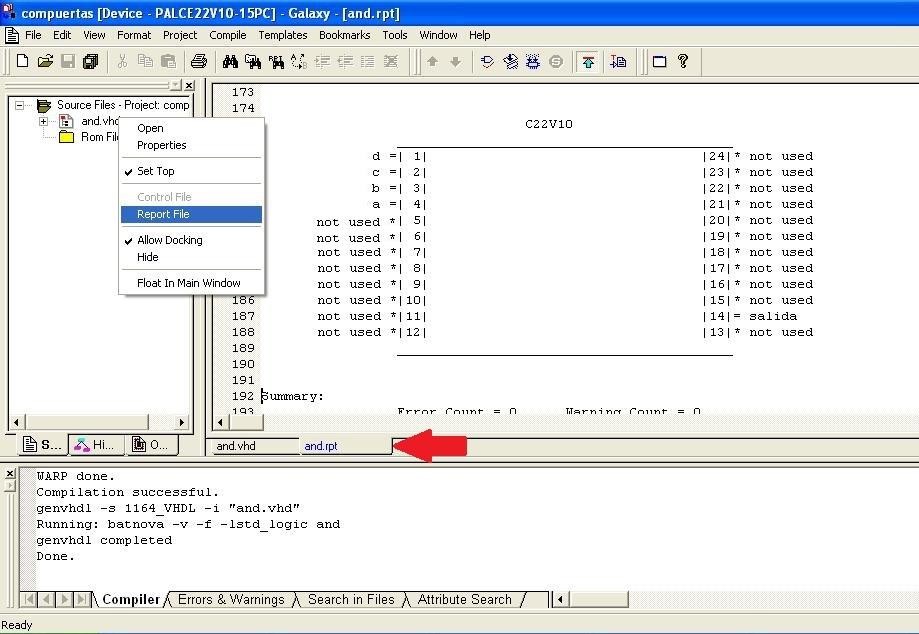


Figura 10. Reporte del diseño.

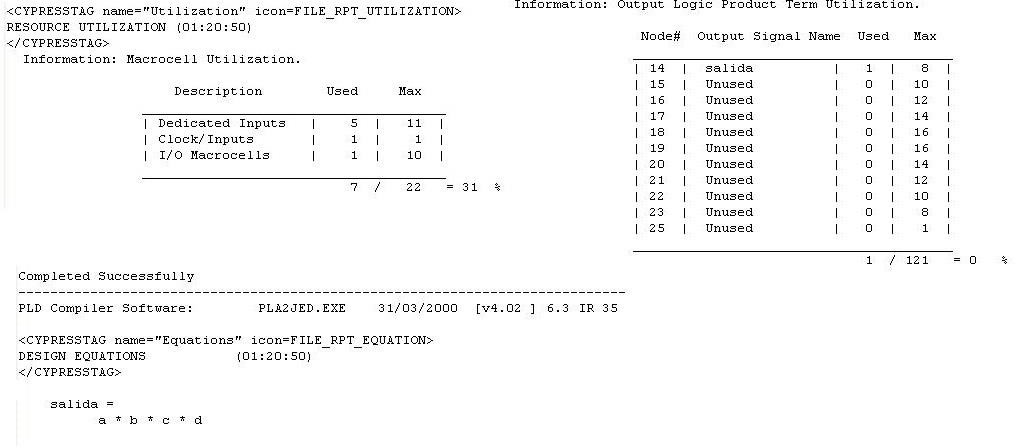


Figura 11.

1. **PROGRAMACIÓN DEL DISPOSITIVO:** Para cargar a la GAL el programa que se generó, es necesario guardar el archivo jedec (con extensión .jed; en nuestro caso ***and.jed***), el cual se encuentra en la carpeta del proyecto, e ir al programador universal. En el laboratorio se tiene un programador con su respectivo software para dicho propósito. En la interfaz se selecciona el archivo y se indica a que dispositivo se va a cargar, se coloca el dispositivo en la base del programador y se procede a guardar el archivo en la GAL.

# COMPROBACIÓN FÍSICA.

Esta última etapa consiste en realizar la conexión física del circuito y corroborar que efectivamente hace la función para la cual fue programado.

La interconexión se realiza como se muestra en la figura 12. Es necesario aclarar que esta figura solo es ilustrativa, ya que para hacer la conexión se debe revisar el archivo de reporte para saber que pines se le asignaron a cada señal.

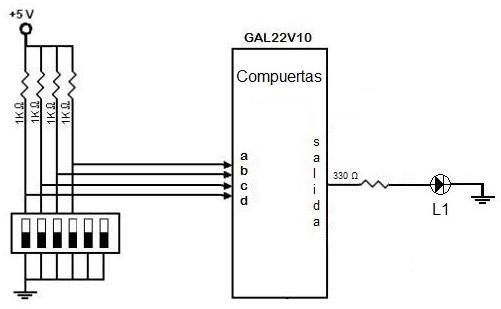
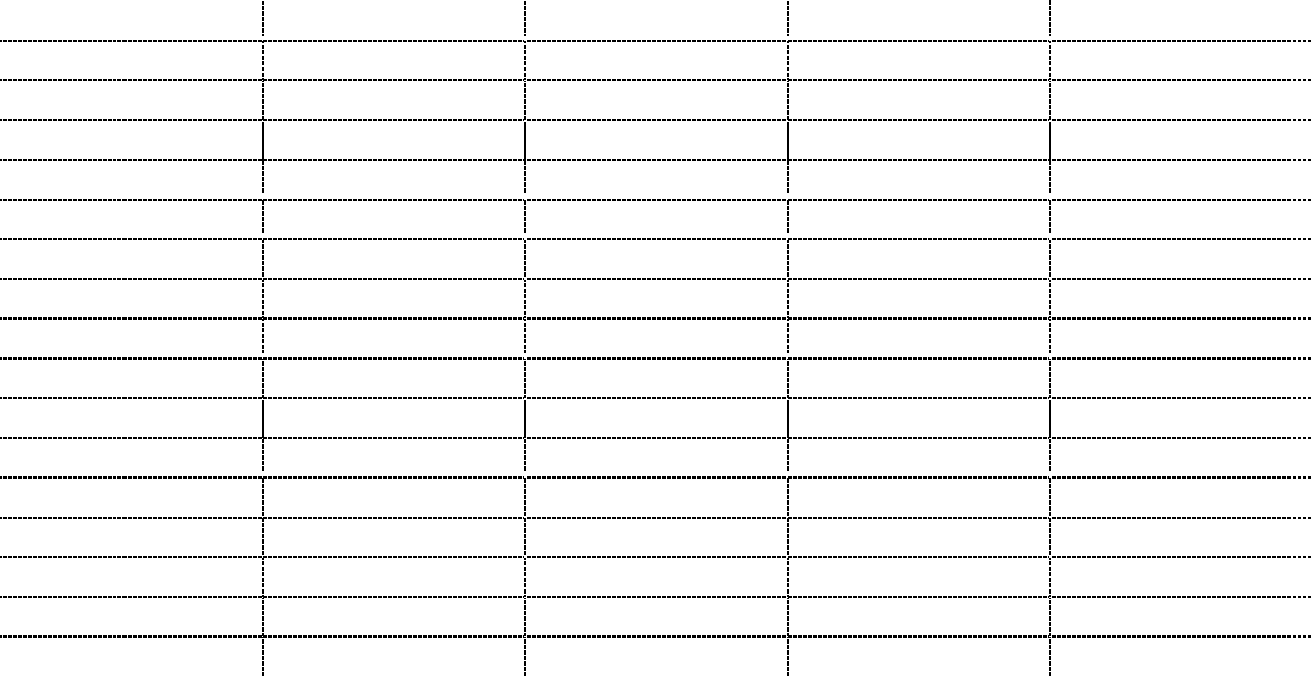
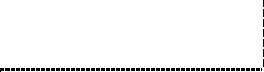


Figura 12. Diagrama de conexión.

Una vez realizado el programa, se procede a determinar la tabla de verdad para la compuerta.



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Entradas** |  |  | **Salida compuerta**  **AND** |
| **A** | **B** | **C** | **D** |
| 0 | 0 | 0 | 0 |  |
| 0 | 0 | 0 | 1 |  |
| 0 | 0 | 1 | 0 |  |
| 0 | 0 | 1 | 1 |  |
| 0 | 1 | 0 | 0 |  |
| 0 | 1 | 0 | 1 |  |
| 0 | 1 | 1 | 0 |  |
| 0 | 1 | 1 | 1 |  |
| 1 | 0 | 0 | 0 |  |
| 1 | 0 | 0 | 1 |  |
| 1 | 0 | 1 | 0 |  |
| 1 | 0 | 1 | 1 |  |
| 1 | 1 | 0 | 0 |  |
| 1 | 1 | 0 | 1 |  |
| 1 | 1 | 1 | 0 |  |
| 1 | 1 | 1 | 1 |  |

Para continuar con la práctica, los alumnos deben realizar los mismos pasos para obtener las compuertas lógicas OR, NAND, NOR, XOR y XNOR, con sus respectivas tablas de verdad.

## OBSERVACIONES Y

**CONCLUSIONES:**

Individuales.