Laboratorio Nº0 – "Introducción a Vivado y a FPGA NEXYS 4 DDR"

Objetivos

En este ejercicio de laboratorio, diseñaremos e implementaremos un multiplexor 2 a 1 (MUX), utilizando las herramientas de Xilinx Vivado para crear un modelo VHDL del diseño, verificar el modelo e implementar el modelo en una "Field Programmable Gate Array" (FPGA).

El comportamiento del multiplexor se implementará de dos maneras diferentes:

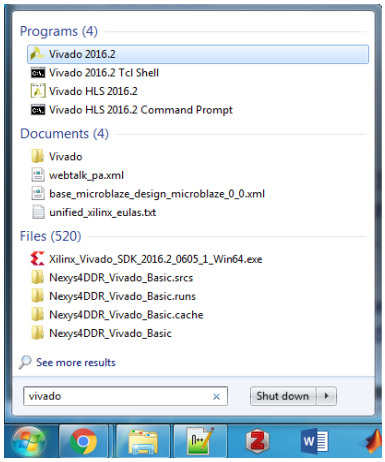
- Ecuaciones lógicas, que deberían ser familiares de los circuitos lógicos digitales.
- Descripción del comportamiento, en la que especificamos el comportamiento de entrada/salida deseada y permitimos que la herramienta de síntesis de Vivado implemente la lógica requerida.
- Las herramientas Active-HDL se utilizarán para simular y verificar el diseño, con la depuración y corrección del modelo según sea necesario.
- Una vez que el diseño esté completamente verificado, utilizaremos Vivado para sintetizar el diseño en un FPGA Artix-7, generando un archivo de datos de configuración que luego se descargará en una placa de circuito Diligent Nexus 4 DDR, donde el diseño implementado se verificará en el hardware usando interruptores para estimular el circuito y LEDs para observar las salidas.
- Si no tienes idea de lo que esto significa, no te preocupes. ¡Es por eso por lo que esto es un tutorial llamado LAB0!

Precauciones

- Tendrá que leer realmente las instrucciones.
- Si intenta simplemente mirar las imágenes, perderá pasos, cometerá errores, encontrará errores, etc.

Nuevo Proyecto

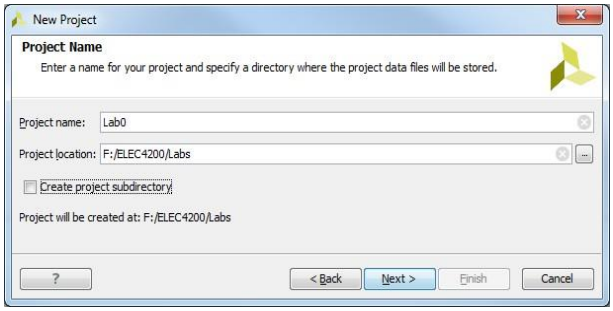
[1] Haz clic en el menú de inicio y escribe "Vivado". Luego busque "Vivado 2016" (o la última versión instalada) y haga clic en él.



[2] Inicie el Asistente para nuevos proyectos haciendo clic en "Crear nuevo proyecto" en la página de inicio rápido o en "Nuevo proyecto" en el menú desplegable "Archivo".

[3] Haga clic en "Next" en la primera pantalla para producir la ventana Nuevo proyecto.

Laboratorio Nº0 – "Introducción a Vivado y a FPGA NEXYS 4 DDR"

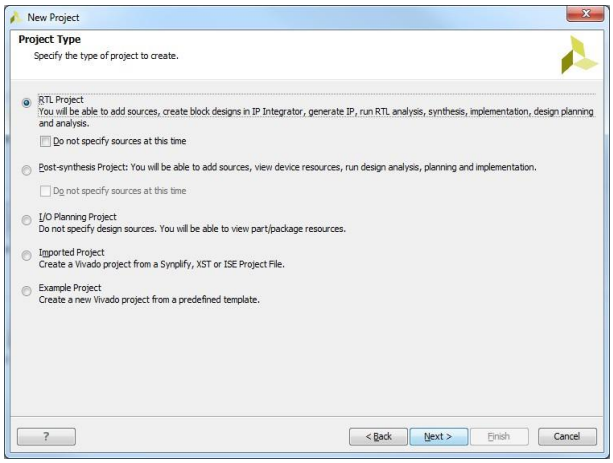


[4] Introduzca el nombre del proyecto y la ubicación en las casillas correspondientes.

Nota: Se recomienda ENCARECIDAMENTE que, mientras trabaja en el laboratorio, sus proyectos estén ubicados en una unidad flash externa o en la unidad C: (NO en su unidad H:). Si su proyecto está ubicado en la unidad H:, uno de los pasos de síntesis posteriores SIEMPRE fallará. Si usa la unidad C:, puede usar el directorio "C:\TEMP\".

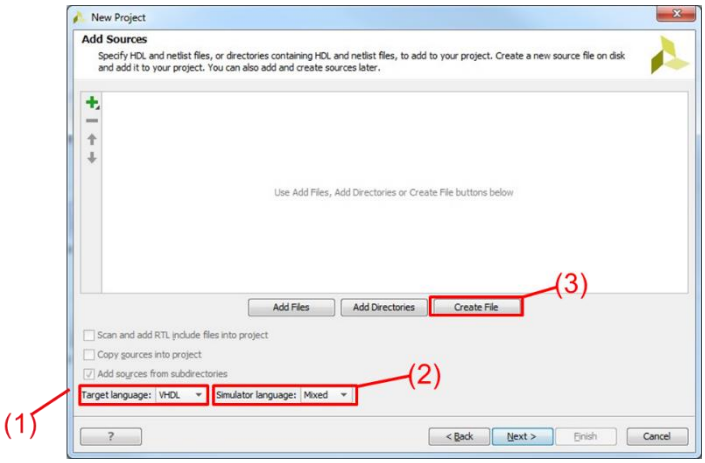
[5] Dale al proyecto un nombre descriptivo, como "LAB0" y haz clic en "Next".

[6] Seleccione "RTL Project" como tipo de proyecto y haga clic en "Next".



[7] En la ventana Agregar fuentes, configure el "Target Language" en VHDL y el "Simulator language" en Mixto.

[8] Después de configurar las opciones de lenguaje, haga clic en "Create File"

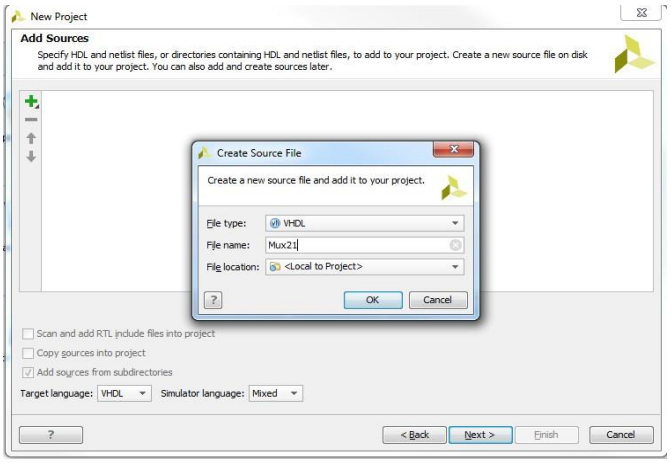


Laboratorio Nº0 – “Introducción a Vivado y a FPGA NEXYS 4 DDR”

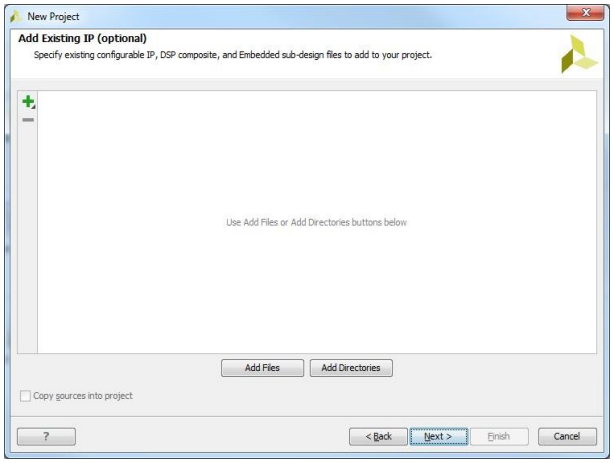
[9] En el cuadro de diálogo Crear archivo de origen, asegúrese de que el "File type" esté configurado en VHDL e ingrese el nombre de su archivo.

[10]Nuevamente, siempre es una buena idea usar un nombre descriptivo, en este caso Mux21 porque es un multiplexor 2-1.

[11]Haga clic en "Ok" para cerrar el cuadro de diálogo y luego en "Next".

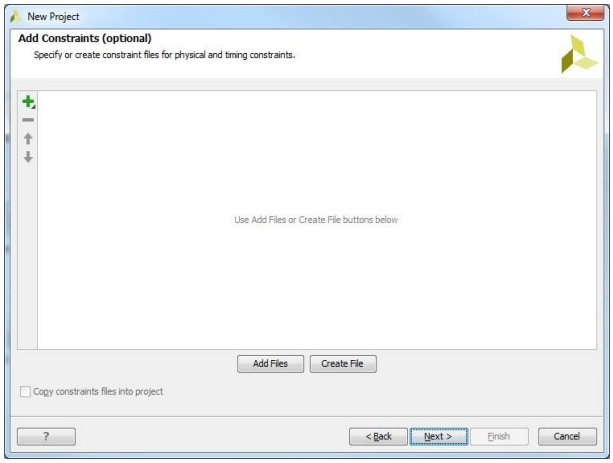


[12]No estamos utilizando ninguna IP existente (componentes de propiedad intelectual), así que haga clic en "Next" nuevamente.



[13]Todavía no tenemos restricciones, pero en proyectos futuros, puede importar y editar archivos de restricciones existentes para ahorrar tiempo. Así que haga clic en "Next" por ahora.

Laboratorio Nº0 – "Introducción a Vivado y a FPGA NEXYS 4 DDR"



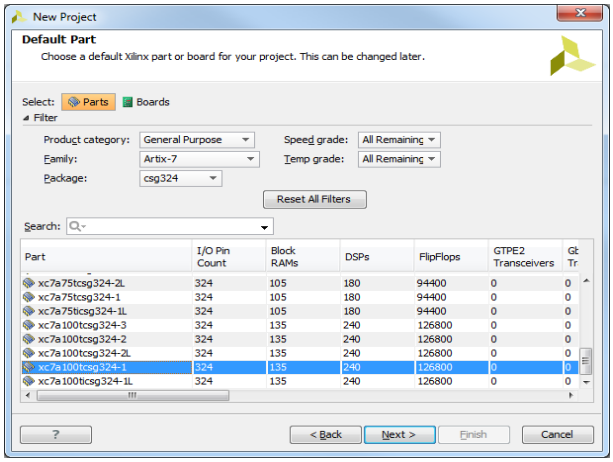
[14]Aquí tenemos que seleccionar nuestro dispositivo. Puedes usar la función de búsqueda, filtros o simplemente desplazarte hasta encontrar nuestro dispositivo: **XC7A100tcsg324-1**.

[15]Este FPGA es de la familia Xilinx Artix-7 (**XC7A100T**).

[16]El dispositivo está contenido en un paquete **csg324** de 324 pines.

[17]El grado de velocidad de la pieza es "**-1**".

[18]Haga clic en "Next"

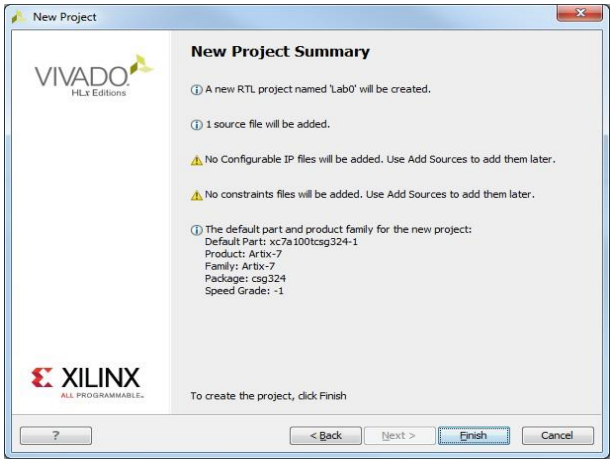


[19]La ventana Resumen del nuevo proyecto enumera la información seleccionada en las pantallas anteriores.

[20]Si es necesario, utilice el botón "Back" para volver a las pantallas anteriores para realizar cambios/correcciones.

[21]Haga clic en "Finish" para abrir la ventana del Administrador de proyectos.

Laboratorio Nº0 – “Introducción a Vivado y a FPGA NEXYS 4 DDR”



Crear el modelo VHDL

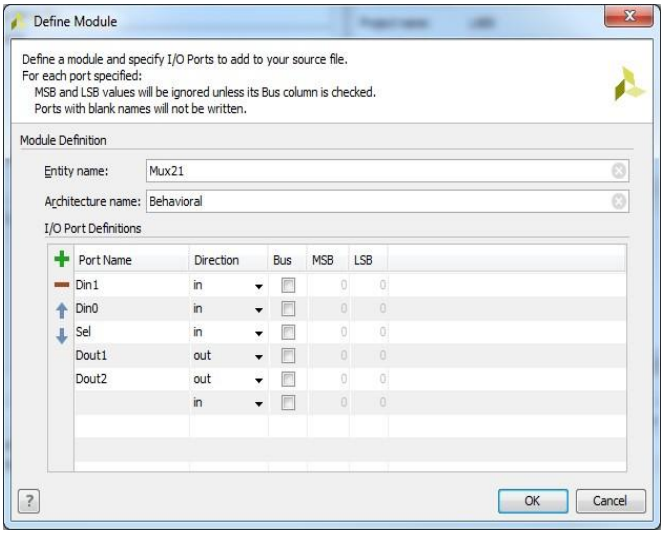
[1] Dado que elegimos crear un nuevo archivo VHDL, Vivado lanzará automáticamente un asistente para ayudar a crear las estructuras de entidad y arquitectura que componen un modelo VHDL. Todo esto se puede hacer a mano (y puede editar todo esto más tarde), pero no hay razón para no aprovechar la utilidad del asistente.

[2] Deje los nombres de Entidad y Arquitectura con sus valores predeterminados.

[3] Cree tres entradas (Dirección “in”): Din0, Din1, Sel

[4] Cree dos salidas (Dirección "out"): Dout1, Dout2

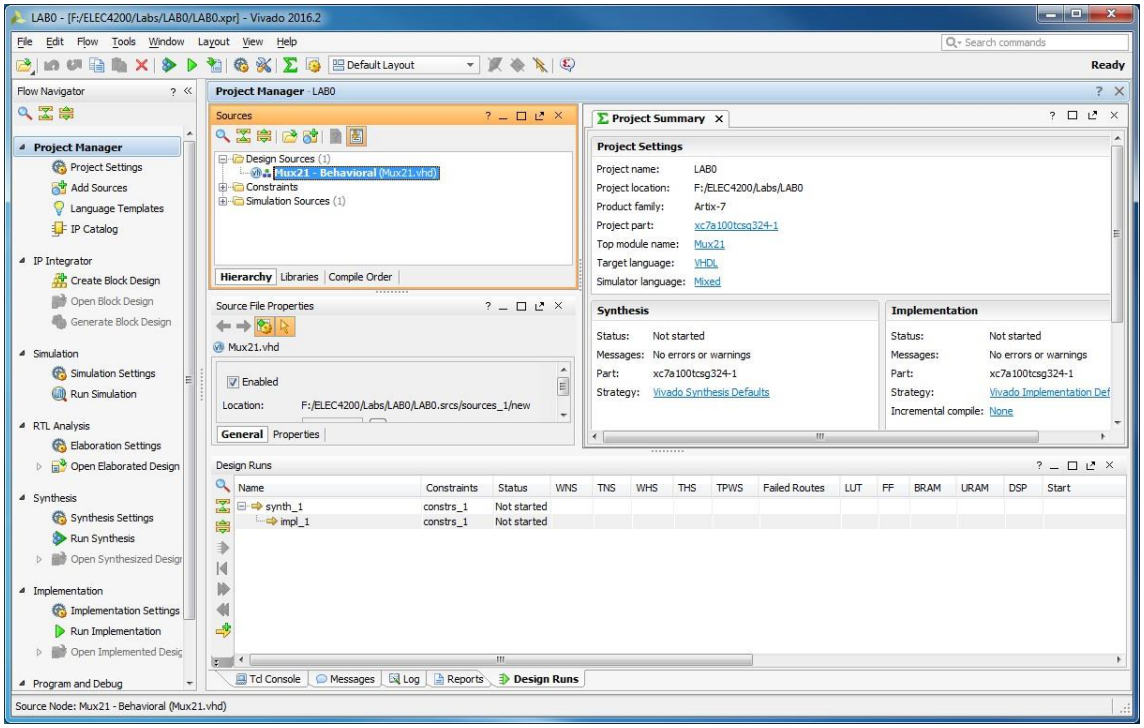
[5] Haga clic en “Ok"



Laboratorio Nº0 – "Introducción a Vivado y a FPGA NEXYS 4 DDR"

Editar modelo VHDL

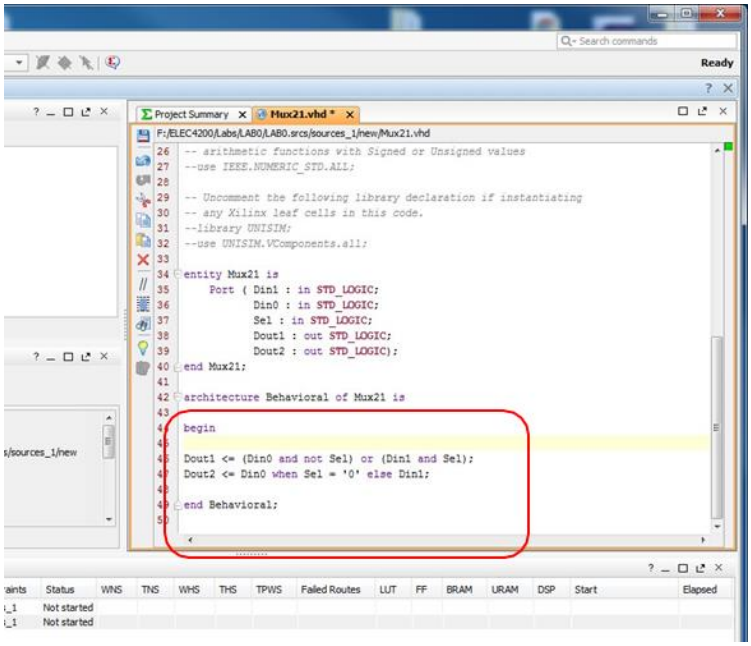
[1] Abra su nuevo archivo VHDL desde la ventana de Fuentes (haga doble clic en el nombre de la fuente de diseño, Mux21)



[2] Desplácese hacia abajo y agregue las siguientes dos líneas de código entre las instrucciones Begin y End de la sección de arquitectura del modelo.

a. Dout1<=(Din0 and not Sel) or (Din1 and Sel);
b. Dout2<=Din0 when Sel = '0' else Din1;

[3] Después de guardar, podemos configurar el simulador.



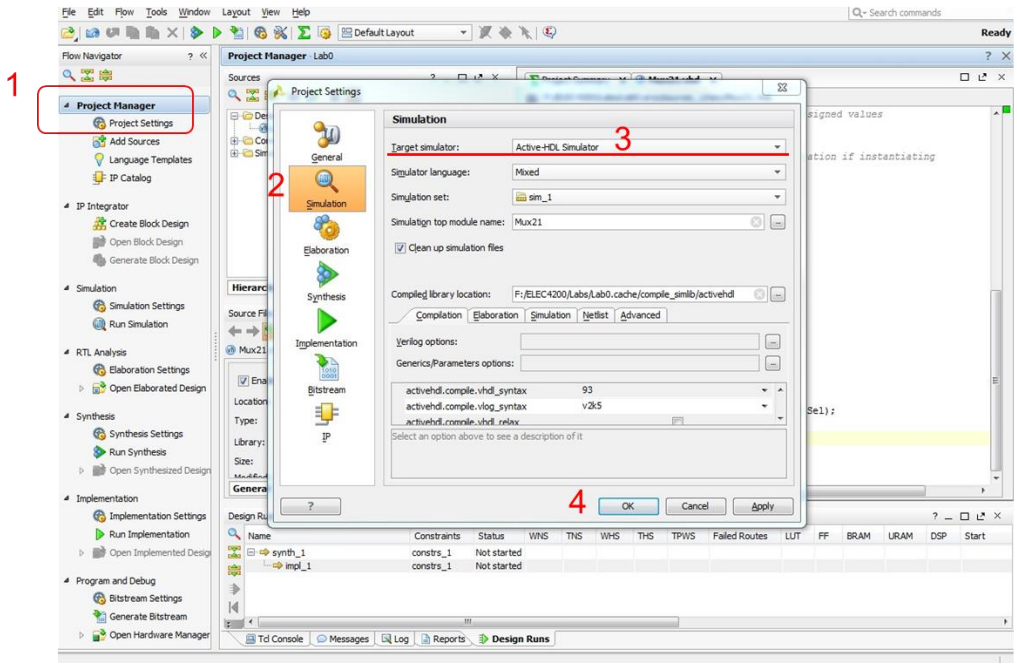
Laboratorio Nº0 – “Introducción a Vivado y a FPGA NEXYS 4 DDR”

Configuración de Active-HDL

Aldec Active-HDL es una herramienta de modelado y simulación disponible comercialmente que se utiliza para verificar diseños antes de sintetizarlos en hardware real. Xilinx Vivado integra soporte para Active-HDL, así como varias otras herramientas comerciales de simulación y su propio simulador. Por lo tanto, debe seleccionar Active-HDL de una lista de simuladores.

Siempre que use la misma computadora semana a semana, esta configuración solo debe realizarse esta vez. Sin embargo, si alguna vez recibe errores de Vivado acerca de no poder encontrar el simulador, verifique primero esta configuración.

[1] En el panel Administrador de proyectos, seleccione Configuración del proyecto
[2] En la ventana Configuración del proyecto, seleccione Simulación
[3] Seleccione Active-HDL Simulator en el menú desplegable Target simulator.
[4] Haga clic en Aceptar para cerrar la ventana.

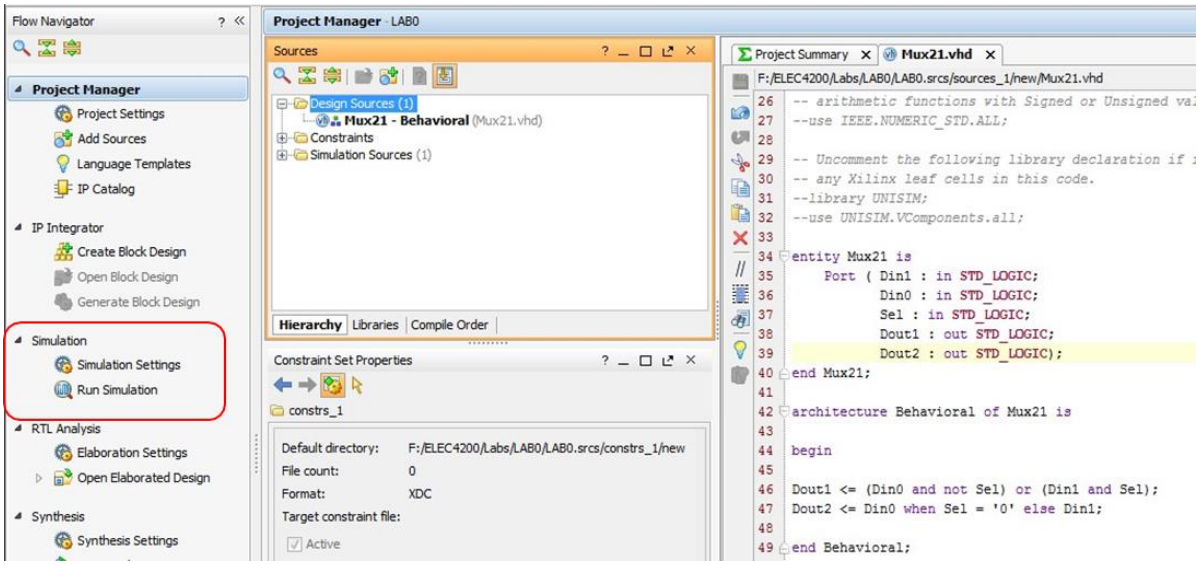


Laboratorio Nº0 – "Introducción a Vivado y a FPGA NEXYS 4 DDR"

Simulación de Active-HDL

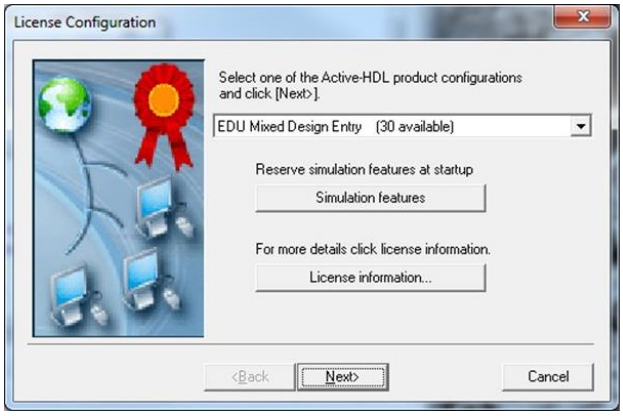
Active-HDL está repleto de funciones que ayudan con el desarrollo, la simulación y la depuración de diseños de FPGA. Este tutorial describe solo cómo realizar una simulación básica, pero se recomienda que experimente con sus muchas capacidades, ya que puede encontrar formas en que puede ayudarlo a depurar en sesiones de laboratorio posteriores.

Haga clic en "Ejecutar simulación" en el navegador de flujo y luego en "Ejecutar simulación de comportamiento" en el menú emergente.



Active-HDL se ejecutará dos veces

- Se abrirá para compilar los modelos VHDL y luego se cerrará.
- Si no hay errores de compilación, se abrirá de nuevo para simular el modelo y permanecerá abierto.
- Cada vez debe hacer clic en "Siguiente" en la ventana Configuración de licencia.

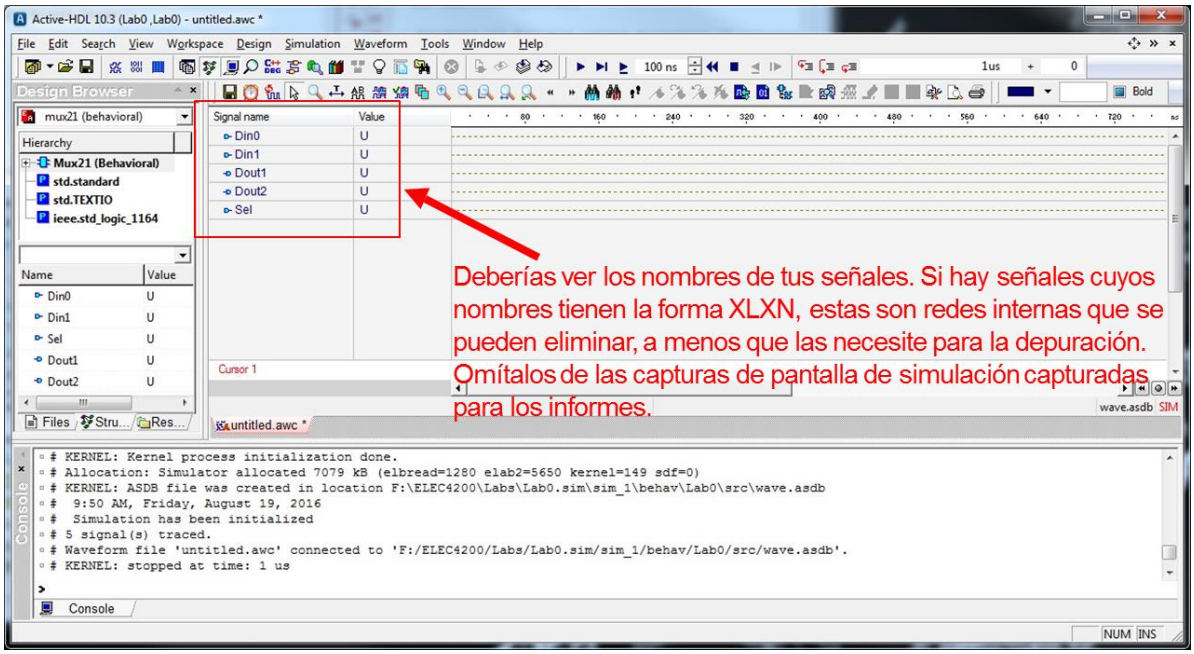


Haga clic en Aceptar en la ventana de información "La simulación había terminado".

- Tenga en cuenta que aún no hemos proporcionado ninguna entrada de señal, por lo que esta "simulación" no proporcionó ninguna información útil.
- Configuraremos la simulación deseada en los siguientes pasos.

Laboratorio Nº0 – "Introducción a Vivado y a FPGA NEXYS 4 DDR"

Cuando el simulador termine de iniciarse, debería ver la siguiente ventana.



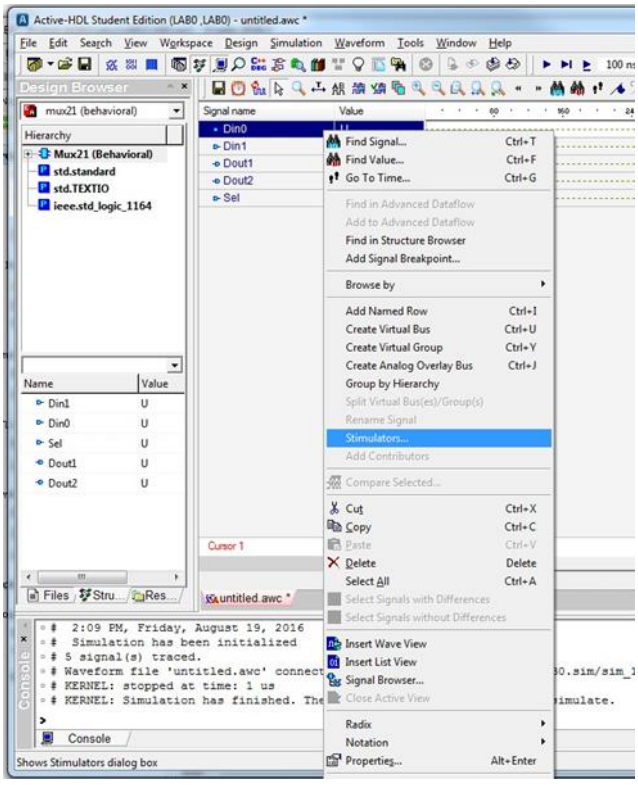
La verificación del diseño requiere que estimule las entradas y observe las salidas. Debe estimular las entradas con todas las combinaciones de entrada posibles y observar cada salida para verificar su corrección.

Esto se llama "prueba exhaustiva". Es adecuado para un circuito simple (como el mux) pero no es práctico para circuitos grandes con muchas entradas.

Haga clic con el botón derecho en una de sus señales (por ejemplo, Din0) y seleccione "Estimuladores..." en el menú contextual.

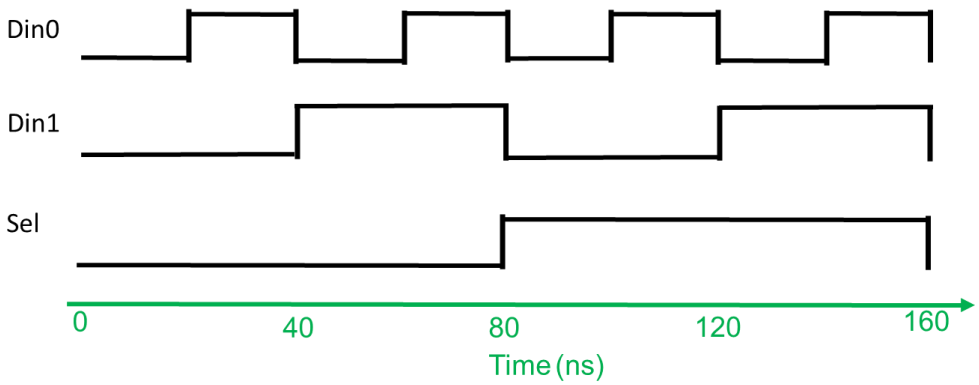
También puede cambiar/controlar el clic para seleccionarlos todos a la vez.

Para el 2-1 MUX tenemos 3 entradas (Din0, Din1 y Sel) donde cada entrada puede ser lógica alta (1) o lógica baja (0).



Hay 2^3=8 combinaciones de entrada posibles. Si bien podríamos estimular cada combinación individualmente, un enfoque más fácil sería usar el estimulador de reloj de Active-HDL para generar los diagramas de tiempo de entrada a continuación.

Laboratorio Nº0 – "Introducción a Vivado y a FPGA NEXYS 4 DDR"

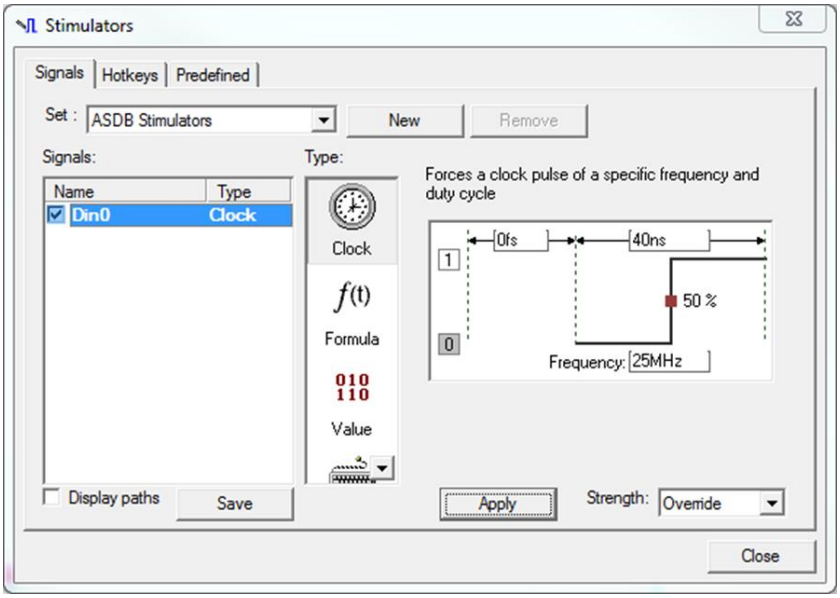


Seleccione su señal y haga clic en el tipo "Reloj". El lado derecho del cuadro de diálogo Estimuladores ahora debería mostrar los parámetros del reloj.

Puede establecer el valor inicial (0 o 1), ajustar el valor de compensación para iniciar el reloj, ajustar el período y ajustar el ciclo de trabajo

Establezca las opciones como se muestra a la derecha para crear una forma de onda que comience baja y se repita cada 40 ns con un ciclo de trabajo del 50 %.

Haga clic en "Aplicar"



Mueva el cuadro de diálogo Estimuladores fuera del camino para que pueda ver las señales en la Ventana de forma de onda. Haga clic en una señal en la Ventana de forma de onda para agregarla al cuadro de diálogo Estimuladores.

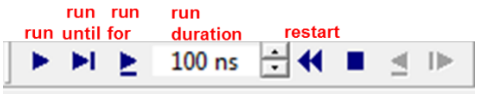
Repita el procedimiento del reloj para Din1 y Sel con los siguientes períodos. Haga clic en "Aplicar" después de agregar cada señal.

- Din1: período de 80 ns
- Sel: período de 160 ns

Haga clic en "Cerrar" cuando haya terminado.

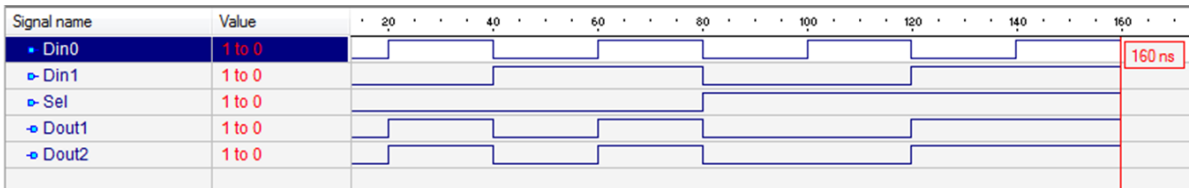
Laboratorio Nº0 – "Introducción a Vivado y a FPGA NEXYS 4 DDR"

Ahora necesitamos ejecutar la simulación usando los siguientes controles de simulación.



Cada vez que comience una nueva simulación, debe borrar la ventana de formas de onda. Haga esto haciendo clic en el botón "Reiniciar simulación" o escribiendo "reiniciar" en la ventana de la consola.

Ejecute la simulación durante 160 ns escribiendo 160 ns en el cuadro "Duración de ejecución" y luego haga clic en "Ejecutar durante". Su salida debe ser similar a esto. Utilice los controles de zoom según sea necesario para que la forma de onda llene la ventana.



Examine sus resultados y asegúrese de que coincidan con precisión con la operación de un MUX 2-1 para Dout1 y Dout2.

Si observa algún resultado incorrecto durante la simulación, regrese y depure su circuito. No continúe hasta que su simulación produzca los resultados correctos.

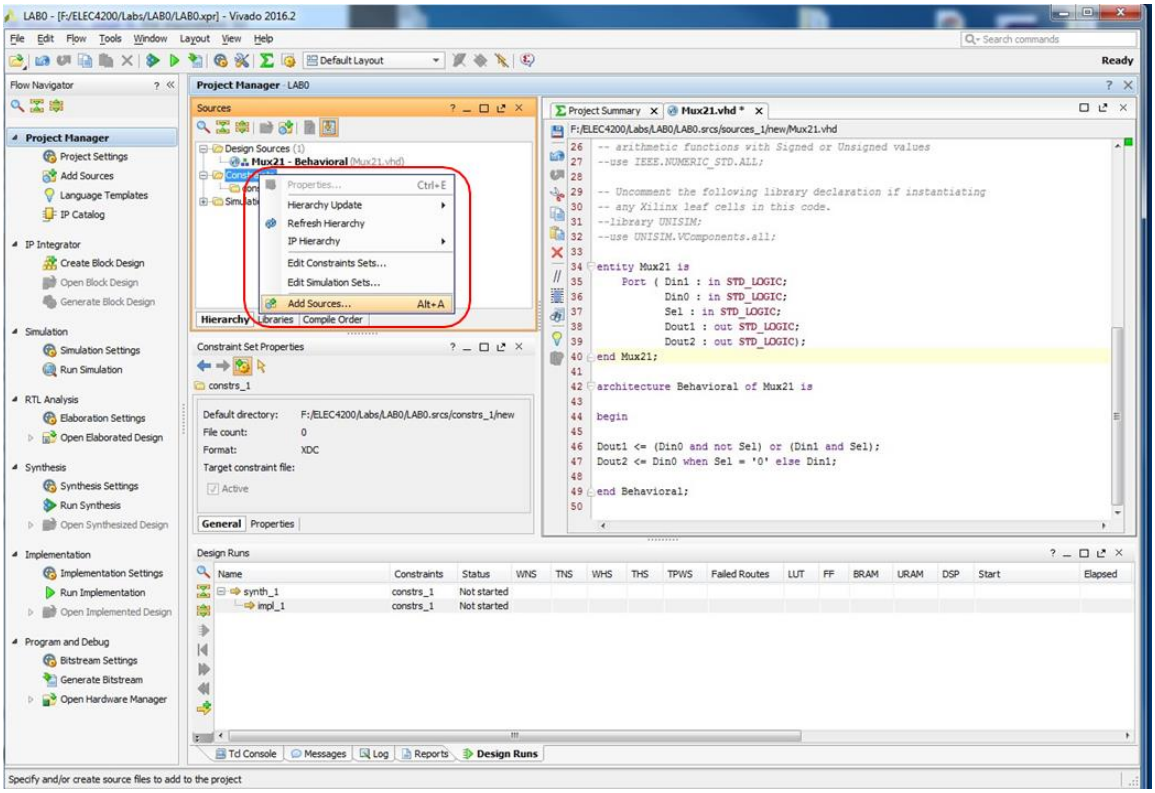
Si el tiempo lo permite, es posible que desee experimentar con algunos de los otros "tipos" en la ventana Estimuladores para que pueda familiarizarse con su funcionamiento y saber cómo usarlos en futuros laboratorios.

Laboratorio Nº0 – "Introducción a Vivado y a FPGA NEXYS 4 DDR"

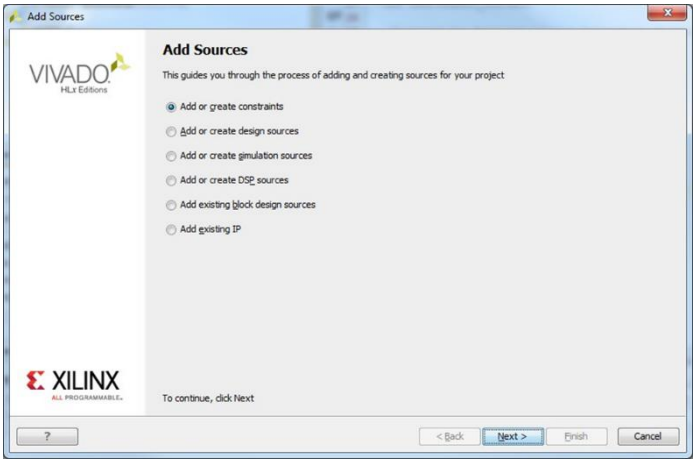
Agregar restricciones

Necesitamos definir un "archivo de restricciones" que defina qué señales (Din, Dout, Sel) van a qué pines en el FPGA.

Haga clic derecho en la carpeta "Restricciones" en el administrador de proyectos y seleccione "Agregar fuentes".

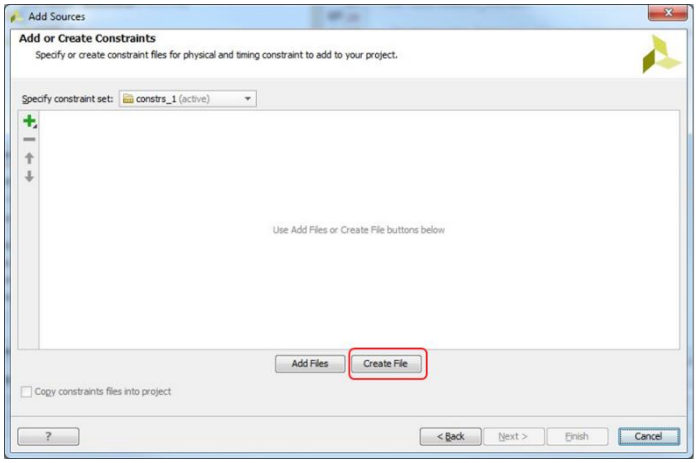


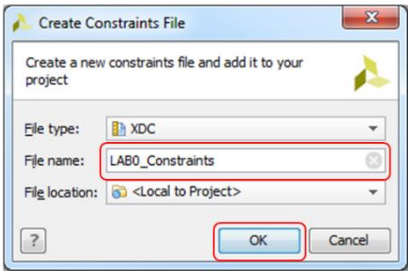
Abre el "Asistente para agregar fuentes". Asegúrese de seleccionar "Agregar o crear restricciones" y haga clic en "Siguiente".



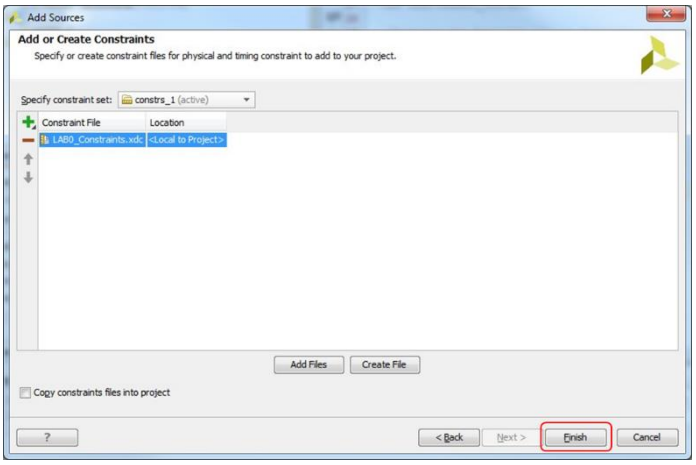
Haga clic en "Crear archivo" y en el cuadro de diálogo que se abre, asigne un nombre a su archivo de restricciones. Como siempre, utilice un nombre descriptivo, como "LAB0_Constraints". Luego haga clic en "Aceptar".

Laboratorio Nº0 – “Introducción a Vivado y a FPGA NEXYS 4 DDR”

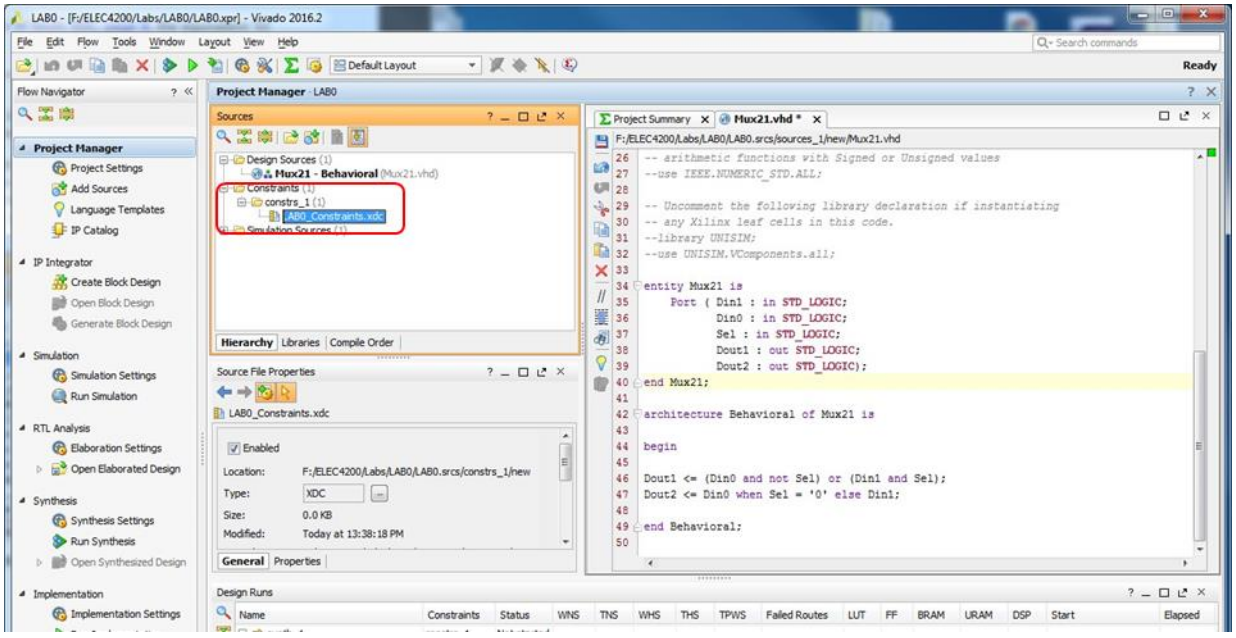




Haga clic en "Finalizar" para agregar su archivo al proyecto



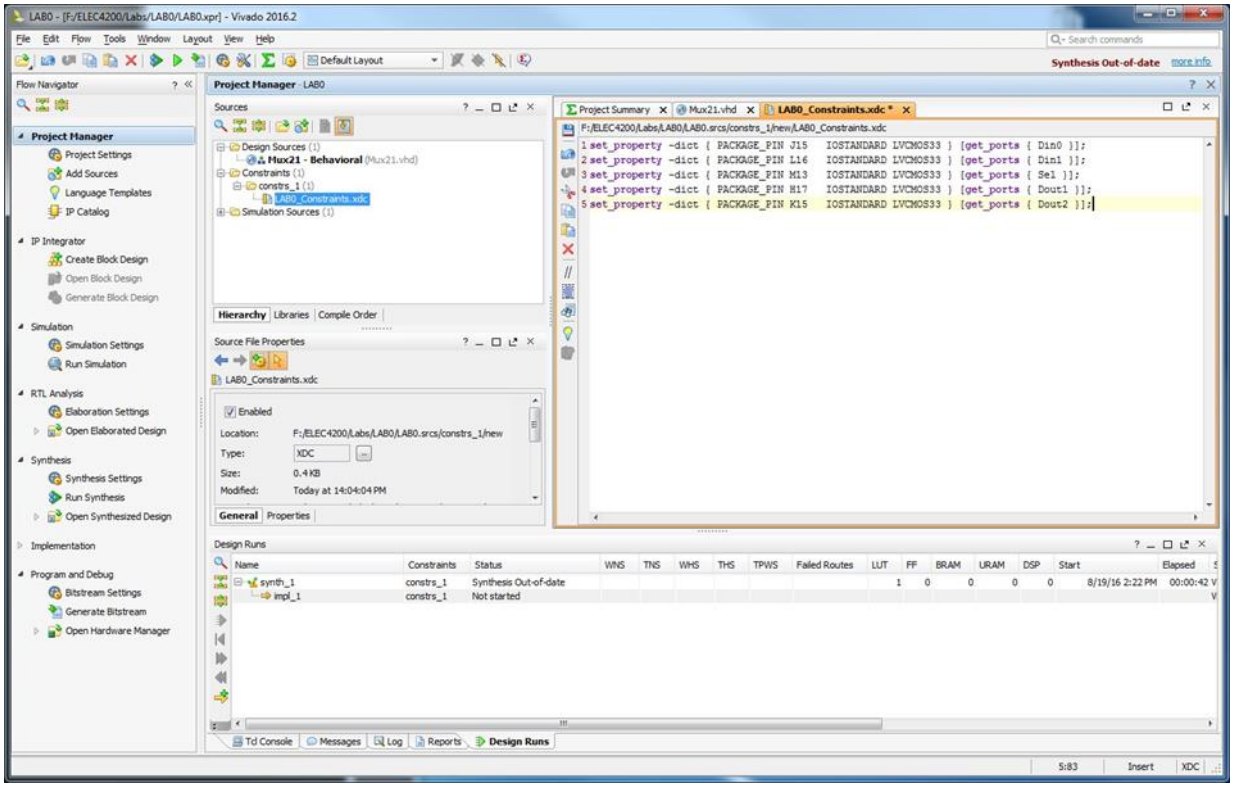
Abra su archivo recién agregado haciendo doble clic en él desde el Administrador de proyectos.



Laboratorio Nº0 – "Introducción a Vivado y a FPGA NEXYS 4 DDR"

Agregue las siguientes líneas en el archivo de restricciones y cambie el nombre de los puertos para que coincidan con los de su diseño. Guardar después de que haya terminado.

```
set_property -dict { PACKAGE_PIN J15 IOSTANDARD LVCMOS33 } [get_ports { Din0 }];
set_property -dict { PACKAGE_PIN L16 IOSTANDARD LVCMOS33 } [get_ports { Din1 }];
set_property -dict { PACKAGE_PIN M13 IOSTANDARD LVCMOS33 } [get_ports { Sel }];
set_property -dict { PACKAGE_PIN H17 IOSTANDARD LVCMOS33 } [get_ports { Dout1 }];
set_property -dict { PACKAGE_PIN K15 IOSTANDARD LVCMOS33 } [get_ports { Dout2 }];
```



Laboratorio Nº0 – “Introducción a Vivado y a FPGA NEXYS 4 DDR”

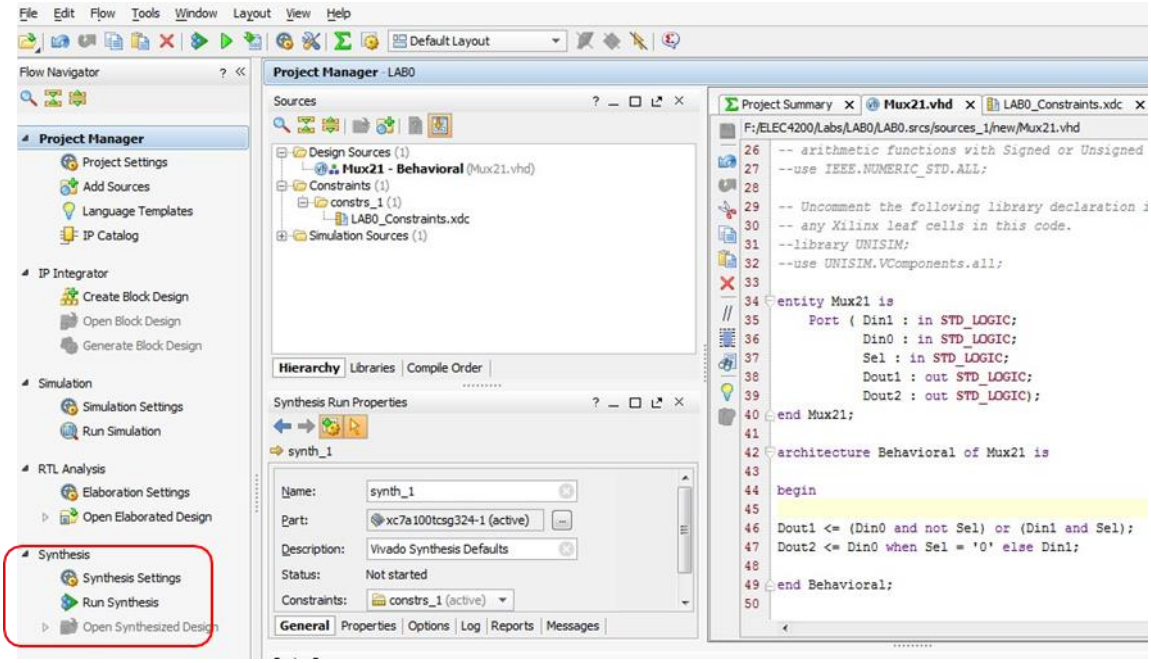
Implementar el diseño

Ahora hará que Vivado "sintetice" su diseño en “componentes lógicos digitales genéricos”.

En el navegador de flujo, haga clic en "Ejecutar síntesis".

Esto puede tardar varios minutos en ejecutarse, así que tenga paciencia.

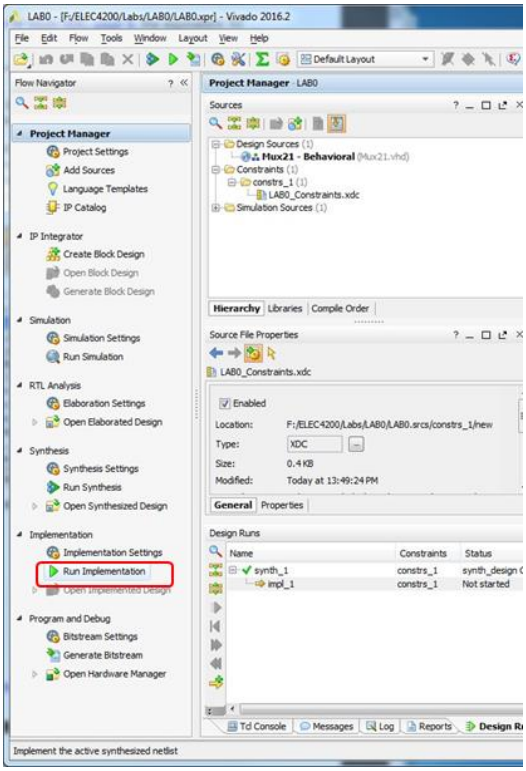
Se abrirá un cuadro de diálogo al finalizar, seleccione "Ejecutar implementación" o presione "Cancelar".



Haga clic en "Implementar diseño". Esto asigna el diseño sintetizado al hardware y enruta los puertos de E/S a los pines especificados en el archivo de restricciones.

Una vez más, esto tardará varios minutos en ejecutarse.

Se abrirá un cuadro de diálogo al finalizar, seleccione "Generar Bitstream" o presione "Cancelar".



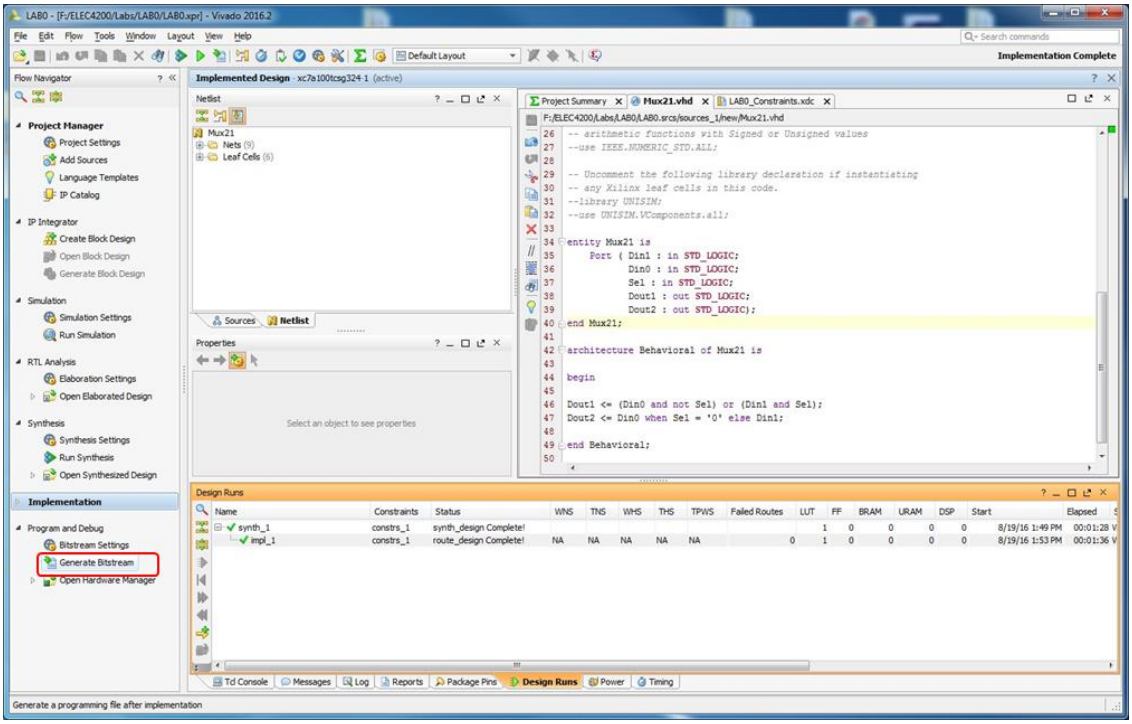
Laboratorio Nº0 – "Introducción a Vivado y a FPGA NEXYS 4 DDR"

Generar Bitstream

Con el diseño ahora implementado, es hora de generar el Bitstream, o el archivo que se descargará a la FPGA.

Haga clic en "Generar Bitstream"

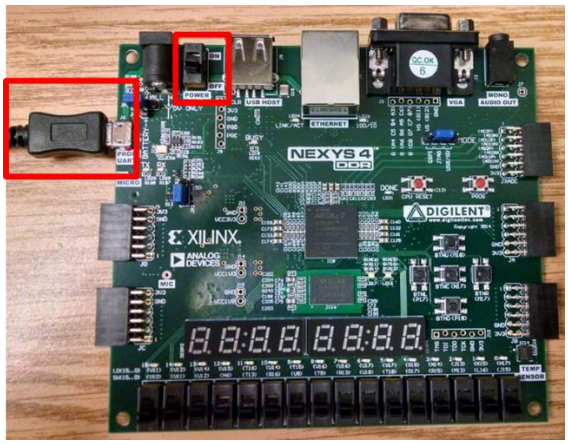
Se abrirá un cuadro de diálogo al finalizar, seleccione "Abrir administrador de hardware" o presione "Cancelar".



Descargar al hardware

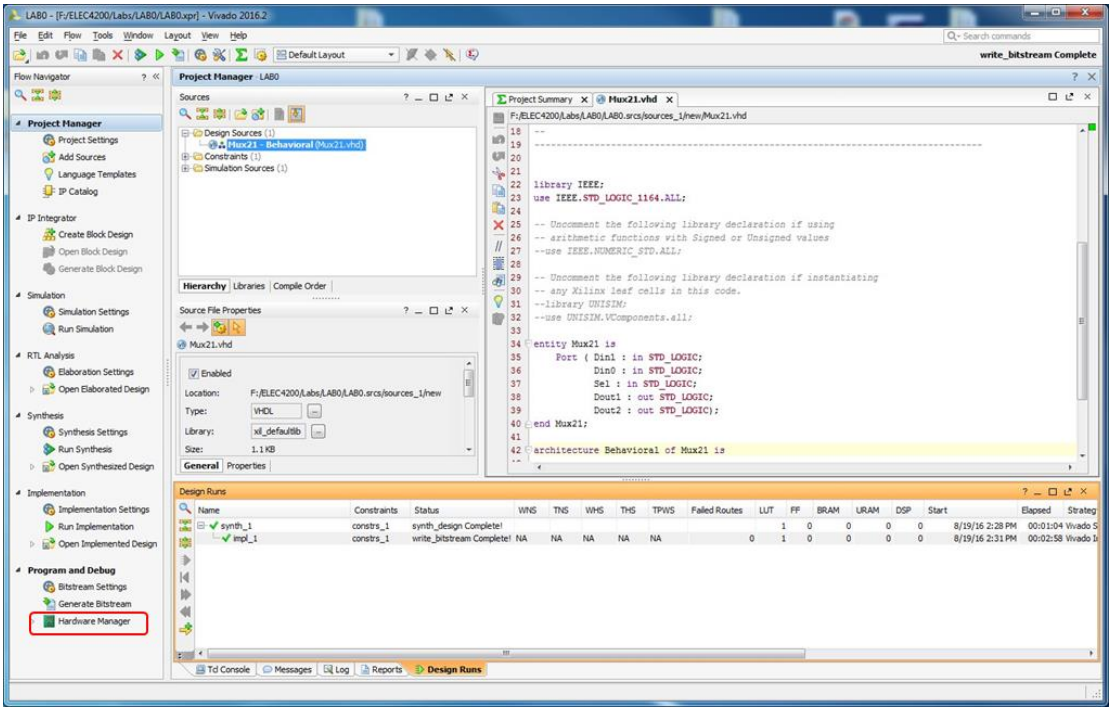
Conecte el cable USB entre el puerto USB de una PC y el puerto USB en la placa FPGA para acceder al módulo de programación JTAG.

Mueva el interruptor de alimentación de la placa de APAGADO a ENCENDIDO



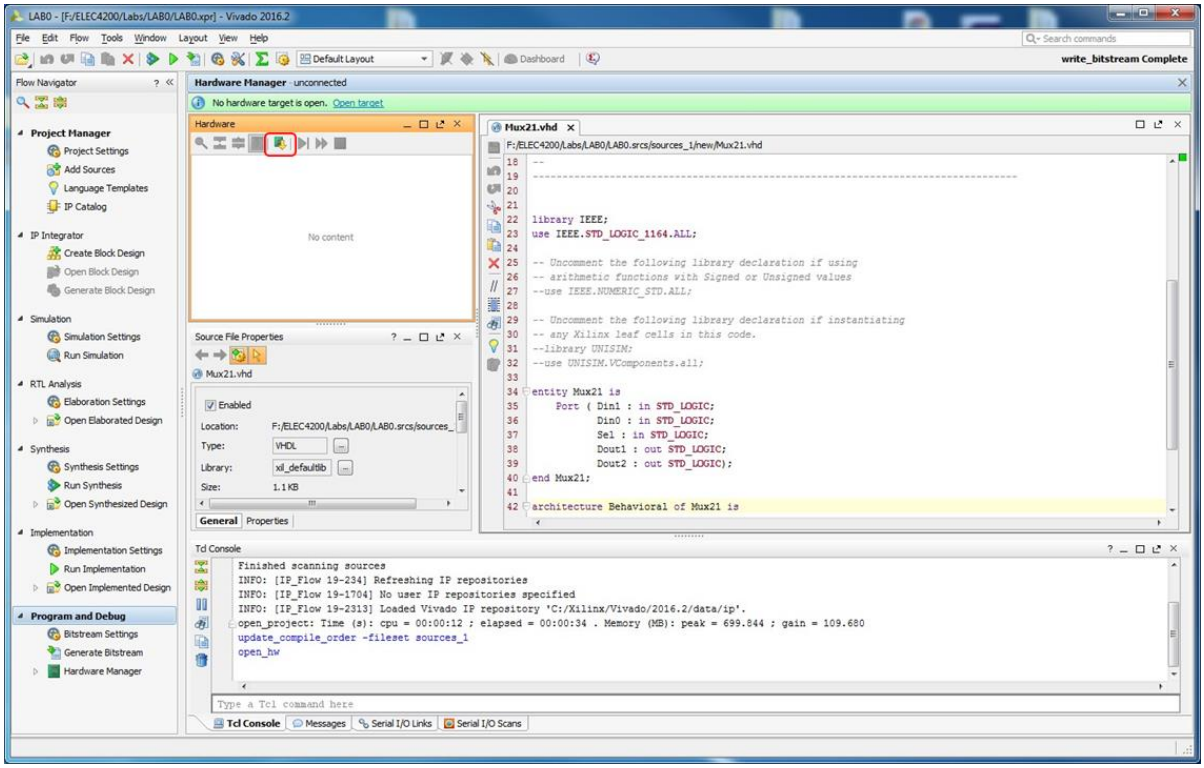
Click en "Hardware Manager"

Laboratorio Nº0 – “Introducción a Vivado y a FPGA NEXYS 4 DDR”



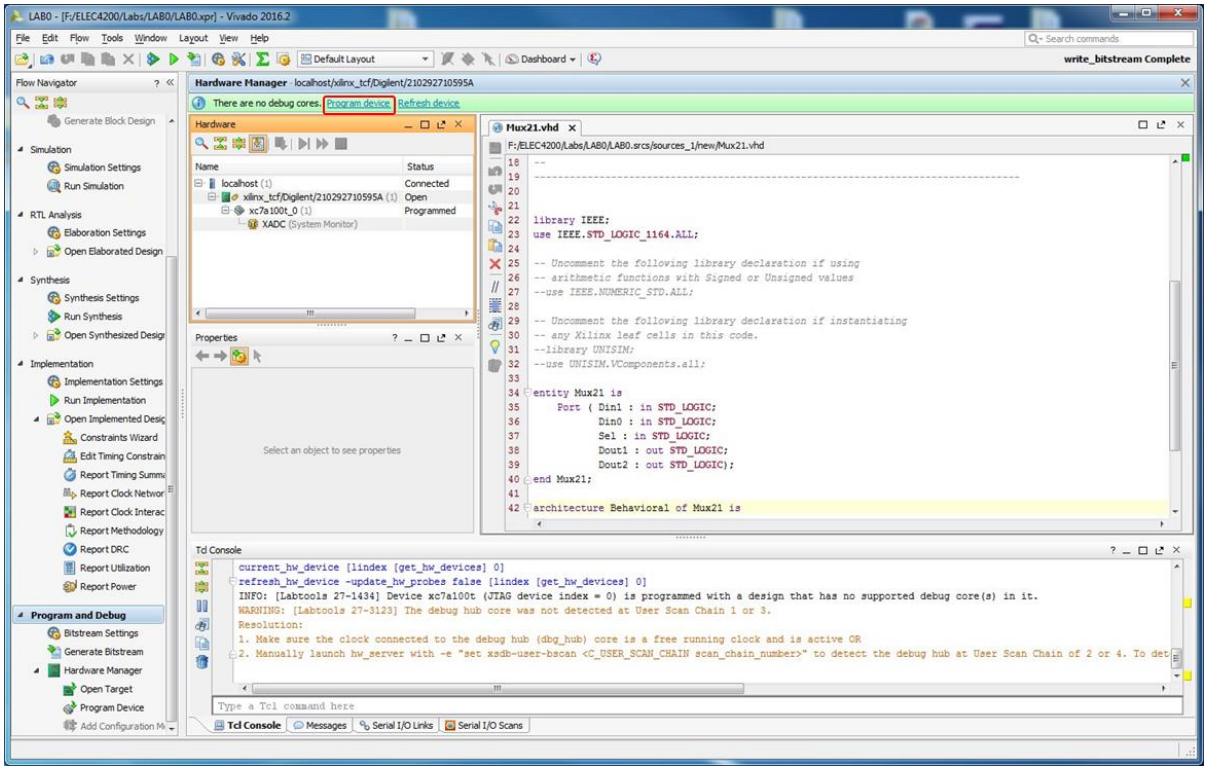
Haga clic en "Conexión automática" para conectarse a su placa Nexys 4.

Si encuentra algún error, verifique que la placa esté enchufada y encendida.



Haga clic en "Programar dispositivo" y seleccione xc7a100t_0 de la lista.

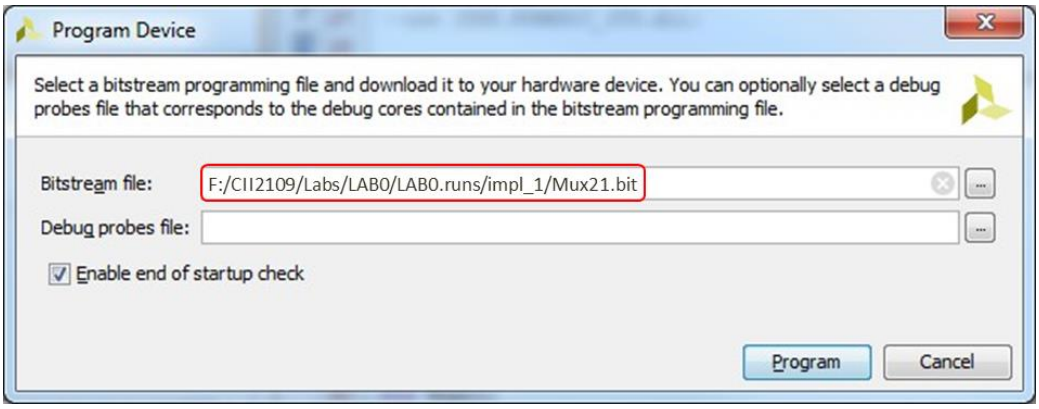
Laboratorio Nº0 – "Introducción a Vivado y a FPGA NEXYS 4 DDR"



En el cuadro de diálogo, asegúrese de que esté seleccionado el archivo de bits correcto.

Deje el cuadro " Debug probes file " vacío por ahora.

Haga clic en "Program"



Verifique el correcto funcionamiento del circuito usando los interruptores y observe la salida en los LED. Si encuentra algún error, intente averiguar qué salió mal antes de pedir ayuda.

Asegúrese de aplicar todas las combinaciones de entrada posibles, como lo hizo en la simulación, y verifique que las salidas coincidan con los resultados de la simulación.

Después de verificar que el circuito es correcto, llame al Profesor y demuestre el circuito.

Laboratorio Nº0 – "Introducción a Vivado y a FPGA NEXYS 4 DDR"

Clean up

- Apague la placa, desenchufe el cable USB, vuelva a colocarlos en su caja y deje la caja al estante.
- Cierre Vivado, cualquier otro programa abierto y guarde todos los archivos.
- No olvide cerrar la sesión de su equipo y llevar cualquier unidad USB con usted.

Laboratorio Nº0 – "Introducción a Vivado y a FPGA NEXYS 4 DDR"

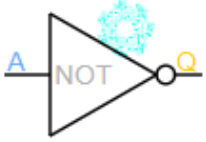
ACTIVIDAD

Visto todo lo anterior, implementar las siguientes compuertas:

Compuerta NOT o INV

Su expresión es representada con una letra testada, para esta compuerta únicamente se cuenta con una entrada y una salida por lo tanto actúa como un inversor. Si la entrada se encuentra en estado activo "1" se tendrá a la salida un estado inactivo "0" y para el caso contrario, si la entrada se encuentra en estado inactivo "0" a la salida estará en estado activo "1".

Símbolo



Expresión

$Q = \overline{A}$

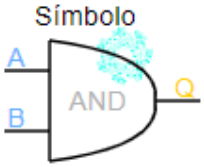
Tabla de verdad

A	Q
0	1
1	0

Compuerta AND

En el Álgebra de Boole se representa por una multiplicación, por lo tanto para tener la salida en estado activo es necesario que sus entradas tengan un estado binario 1, al tener una entrada inactiva "0" su salida será 0. Es posible representarlo mediante un circuito que tenga sus interruptores en serie, al tener todos los interruptores activos permite cerrar el circuito y por lo tanto el flujo de la corriente.

Símbolo



Expresión

$Q = A \cdot B$

Tabla de verdad

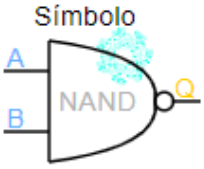
A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

Laboratorio Nº0 – "Introducción a Vivado y a FPGA NEXYS 4 DDR"

Compuerta NAND

También conocida como AND negada o inversa, es una combinación de las compuertas AND y NOT que se representa con la compuerta AND con un círculo a la salida, al tener sus entradas activas "1" la salida se encuentra inactiva "0", otra variación con respeto a las entradas mantendrá su salida en estado activo "1". Se puede representar mediante un circuito con dos interruptores en serie y debemos recordar que el flujo de corriente circula por donde se tenga menor resistencia.

Símbolo



Expresión

$$Q = \overline{A \cdot B}$$

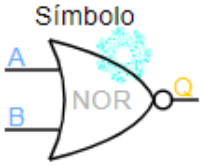
Tabla de verdad

A	B	Q
0	0	1
0	1	1
1	0	1
1	1	0

Compuerta OR

Su expresión en el Álgebra de Boole es representada por una suma. Esta compuerta se encuentra en estado activo siempre y cuando una de sus entradas tenga un estado binario activo "1". Para lograr un estado inactivo "0" a la salida, es necesario que todas sus entradas se encuentren en estado inactivo "0". Se puede representar mediante un circuito que tenga dos interruptores en paralelo, al accionar un interruptor permite cerrar el circuito y por lo tanto el flujo de la corriente.

Símbolo



Expresión

$$Q = \overline{A + B}$$

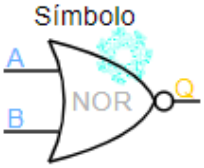
Tabla de verdad

A	B	Q
0	0	1
0	1	0
1	0	0
1	1	0

Compuerta NOR

Es una combinación de las compuertas OR y NOT, en otras palabras la compuerta NOR es la versión inversa de la compuerta OR. Al tener sus entradas en estado inactivo "0" su salida estará en un estado activo "1", pero si alguna de las entradas pasa a un estado binario "1" su salida tendrá un estado inactivo "0". Se puede representar mediante un circuito con los interruptores y salida en paralelo, para tener la salida en estado activo "1" es necesario que ambos interruptores se encuentren abiertos, mientras alguno de los interruptores se encuentre cerrado la salida "y" tendrá un estado binario "0".

Símbolo



Expresión

$$Q = \overline{A + B}$$

Tabla de verdad

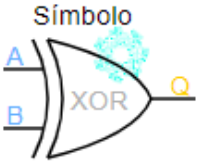
A	B	Q
0	0	1
0	1	0
1	0	0
1	1	0

Laboratorio Nº0 – "Introducción a Vivado y a FPGA NEXYS 4 DDR"

Compuerta XOR

También conocida como "OR exclusiva", su expresión Booleana es una suma binaria de un dígito cada uno y el resultado obtenido será la salida. La salida tiene un estado activo "1" al tener las entradas en estados diferentes (Una activa y otra inactiva). Su representación es mediante cuatro interruptores que se encuentran acoplados mecánicamente a su valor negado, de este modo cuando A se cierra entonces A´ se abre y viceversa, lo mismo ocurre con el interruptor B con respecto al B´.

Símbolo



Expresión

$$Q = A \oplus B$$

$$Q = A \cdot \overline{B} + \overline{A} \cdot B$$

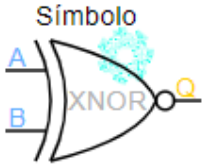
Tabla de verdad

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

Compuerta XNOR

Es la negación de la compuerta XOR, cuando las entradas sean iguales se representará una salida en estado "1" y si son diferentes la salida será un estado "0".

Símbolo



Expresión

$$Q = \overline{A \oplus B}$$

$$Q = \overline{A \cdot \overline{B} + \overline{A} \cdot B}$$

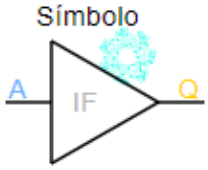
Tabla de verdad

A	B	Q
0	0	1
0	1	0
1	0	0
1	1	1

Compuerta IF o Buffer

Es una de las compuertas menos utilizadas o recocidas, pero a veces es necesaria, sus estados lógicos permanecen del mismo modo, se podría considerar como un cable conectado ya que lo que tenemos a la entrada se obtendrá a la salida. La función principal del buffer es aumentar la corriente suministrada a la salida mientras se retiene el estado lógico, en la práctica es utilizado para amplificar la corriente o como seguidor de tensión para adaptar impedancias.

Símbolo



Expresión

$$Q = A$$

Tabla de verdad

A	Q
0	0
1	1