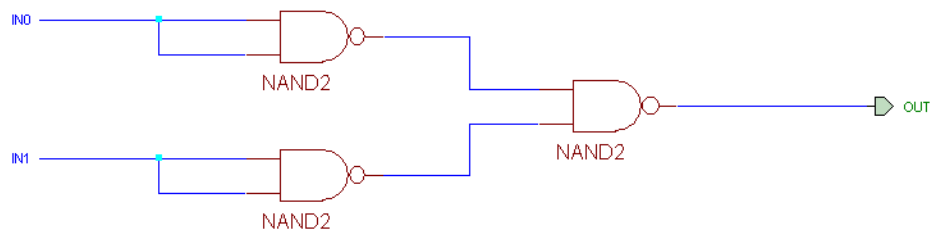


ALBERTO DAZA MÁRQUEZ

PRÁCTICAS DE FUNDAMENTOS DE ELECTRÓNICA



***PRÁCTICAS DE FUNDAMENTOS
DE ELECTRÓNICA***

EDITADO POR

ALBERTO DAZA MÁRQUEZ

**PROFESOR DEL DEPARTAMENTO DE ELECTRÓNICA
DE LA UNIVERSIDAD DE MÁLAGA**

PRÁCTICAS DE FUNDAMENTOS DE ELECTRÓNICA.

Primera edición, Octubre de 2013

© Autor, Editor e Ilustrador: Alberto Daza Márquez

I.S.B.N. 13: 978-84-695-8851-2

I.S.B.N. 10: 84-695-8851-6

Depósito Legal: MA-1990-2013

ÍNDICE.

PRÓLOGO.

III

PRÁCTICA 1. SPICE E INSTRUMENTACIÓN.

PARTE 1. TUTORIAL DE LTSPICE.

1-3

1.1. INTRODUCCIÓN.

1-3

1.2. EJEMPLO DE DISEÑO BÁSICO: CÁLCULO DEL PUNTO DE OPERACIÓN.

1-3

1.3. ANÁLISIS TRANSITORIO .TRAN.

1-12

1.4. ANÁLISIS EN CONTINUA .DC.

1-17

1.5. EJEMPLO DE APLICACIÓN: DIVISOR DE TENSIÓN.

1-17

PARTE 2. TUTORIAL DE INSTRUMENTACIÓN.

1-27

2.1. FUENTE DE ALIMENTACIÓN.

1-27

2.2. GENERADOR DE FUNCIONES.

1-28

2.3. EL OSCILOSCOPIO (I). MEDIDAS BÁSICAS EN 1 CANAL.

1-31

2.4. EL OSCILOSCOPIO (II). MEDIDAS BÁSICAS EN 2 CANALES. EJEMPLO DE MONTAJE.

1-35

PRÁCTICA 2. DIODO.

PARTE 1. SIMULACIÓN CON LTSPICE.

2-3

1.1. ESTUDIO DE LA CURVA CARACTERÍSTICA DE UN DIODO.

2-3

1.2. RECTIFICADOR DE CORRIENTE ALTERNA.

2-3

1.3. SUBIR LA SIMULACIÓN DE LA PRÁCTICA 2 AL CAMPUS VIRTUAL.

2-5

PARTE 2. MONTAJE EN LABORATORIO CON DIODOS.

2-7

2.1. ESTUDIO DE LA CURVA CARACTERÍSTICA DE UN DIODO.

2-7

2.2. RECTIFICADOR DE CORRIENTE ALTERNA.

2-10

2.3. MONTAJE DE UNA PUERTA LÓGICA OR.

2-12

PRÁCTICA 3. TRANSISTOR BIPOLAR.

PARTE 1. SIMULACIÓN CON LTSPICE.

3-3

1.1. ESTUDIO DE LAS CURVAS CARACTERÍSTICAS DE UN TRANSISTOR BIPOLAR.

3-3

1.2. ESTUDIO DE UN INVERSOR RTL.

3-4

1.3. SUBIR LA SIMULACIÓN DE LA PRÁCTICA 3 AL CAMPUS VIRTUAL.

3-6

PARTE 2. MONTAJE EN LABORATORIO CON TRANSISTORES BIPOLARES.

3-7

2.1. TRANSISTOR BIPOLAR NPN 2N2222.

3-7

2.2. CARACTERÍSTICA DE TRANSFERENCIA DE UN INVERSOR RTL.

3-8

2.3. RESPUESTA TEMPORAL DE UN INVERSOR RTL, SIENDO V_i UNA SEÑAL CUADRADA.

3-9

PRÁCTICA 4. TRANSISTOR MOSFET.

PARTE 1. SIMULACIÓN CON LTSPICE.

4-3

1.1. ESTUDIO DE LAS CURVAS CARACTERÍSTICAS DE UN TRANSISTOR MOSFET DE CANAL N.

4-3

1.2. ESTUDIO DE UN INVERSOR NMOS.

4-4

1.3. ESTUDIO DE UN INVERSOR CMOS.

4-7

1.4. FUNCIONES LÓGICAS CMOS.

4-10

1.5. SUBIR LA SIMULACIÓN DE LA PRÁCTICA 4 AL CAMPUS VIRTUAL.

4-12

PARTE 2. MONTAJE EN LABORATORIO CON TRANSISTORES MOSFET.

4-13

2.1. CIRCUITO INTEGRADO MC14007 (CON 6 TRANSISTORES MOSFET)

4-13

2.2. ESTUDIO DE UN INVERSOR NMOS.

4-14

2.3. ESTUDIO DE UN INVERSOR CMOS.

4-16

TUTORIAL XILINX FOUNDATION 3.11.

PARTE 1. INICIAR EL “PROJECT MANAGER” DEL XILINX FOUNDATION 3.11.

X-1

1.1. EL MANEJADOR DE PROYECTO.

X-2

1.2. LIBRERÍAS DEL PROYECTO.

X-3

PARTE 2. DISEÑO DE ESQUEMÁTICOS.

X-4

2.1. AÑADIR COMPONENTES DE LA LIBRERÍA AL ESQUEMÁTICO.

X-7

2.2. CORREGIR ERRORES.

X-7

2.3. DIBUJAR Y ETIQUETAR CABLES.

X-8

2.4. DIBUJAR Y ETIQUETAR BUSES.

X-9

2.5. CREAR UNA MACRO.

X-11

2.6. UTILIZAR UNA MACRO.

X-12

2.7. EDITAR PROPIEDADES DE UNA MACRO.

X-13

2.8. COPIAR UNA (O VARIAS) MACRO(S) A OTRO PROYECTO.

X-15

2.9. GUARDAR EL ESQUEMÁTICO.

X-16

2.10. ARCHIVAR Y RESTAURAR EL PROYECTO.

X-16

PARTE 3. SIMULACIÓN FUNCIONAL.

X-19

3.1. INICIAR EL SIMULADOR LÓGICO.

X-19

3.2. COMPROBAR ERRORES EN LA CARGA DE LA “NETLIST”.

X-20

3.3. REALIZAR LA SIMULACIÓN.

X-21

PARTE 4. IMPLEMENTACIÓN DEL DISEÑO

X-29

4.1. DESCRIPCIÓN DE LA PLACA DE LA CPLD Y DE PERIFÉRICOS.

X-29

4.2. CAMBIOS A REALIZAR EN EL ESQUEMÁTICO PARA ADAPTARSE A LA PLACA DE LA CPLD.

X-31

4.3. MODIFICACIÓN DEL ESQUEMÁTICO PARA INTRODUCIR MACROS DE CONTROL DE LA PLACA DE PERIFÉRICOS.

X-33

4.4. IMPLEMENTACIÓN DEL DISEÑO.

X-35

4.5. VERIFICACIÓN DEL DISEÑO.

X-37

4.6. PROGRAMACIÓN DE LA CPLD.

X-39

PRÁCTICA 5. CIRCUITO COMBINACIONAL DECODIFICADOR 7 SEGMENTOS.

PARTE 1. CREACIÓN DE LOS ESQUEMÁTICOS.

5-3

1.1. DISEÑO Y SIMULACIÓN DE UN CONVERTIDOR BCD – 7 SEGMENTOS.

5-3

PARTE 2. MONTAJE EXPERIMENTAL EN LABORATORIO.

5-9

2.1. IMPLEMENTACIÓN DEL DECODIFICADOR BCD – 7 SEGMENTOS

5-9

PRÁCTICA 6. CIRCUITO SECUENCIAL CONTADOR DNI.

PARTE 1. CREACIÓN DE LOS ESQUEMÁTICOS.

6-3

1.1. DISEÑO Y SIMULACIÓN DE UN SISTEMA SECUENCIAL SÍNCRONO CON 2 SECUENCIAS DIFERENTES DEL DNI.

6-3

PARTE 2. MONTAJE EXPERIMENTAL EN LABORATORIO.

6-11

2.1. IMPLEMENTACIÓN DEL SISTEMA SECUENCIAL SÍNCRONO CON 2 SECUENCIAS DIFERENTES DEL DNI.

6-11

PRÓLOGO

El presente manual se enfoca como un curso práctico para el estudiante de Grado en Informática, si bien es extensible a cualquier grado en carreras técnicas ya que cubre todo el rango básico de dispositivos electrónicos en los que un alumno debe iniciarse. Es por ello que se incluyen conocimientos y extensos tutoriales para manejar tanto instrumentación de laboratorio, que permite realizar diversos experimentos prácticos y obtener conclusiones sobre el funcionamiento de los dispositivos, como software de diseño y simulación de circuitos basados en dispositivos semiconductores (*LTSpice IV*) y en elementos digitales (*Xilinx Foundation 3.1i*).

Por todo esto, la organización de este manual está planteada de la siguiente forma:

- Una primera parte está dedicada a los dispositivos electrónicos más comunes, desde los más básicos elementos activos (fuentes de alimentación) y pasivos (resistencias), hasta los semiconductores (diodos, transistores bipolares y transistores MOSFET). Se abordará desde una doble perspectiva el análisis de diversos circuitos prácticos, por lo que para cada práctica de la nº 1 a la nº 4 se verá en primer lugar la forma de poder resolver los circuitos propuestos con el simulador *LTSpice IV*, y en segundo lugar se propondrá montar unos circuitos similares a los simulados pero en el laboratorio, utilizando componentes discretos, haciendo que el alumno contraste los resultados del simulador con los reales y así sea capaz de alcanzar sus propias conclusiones. Para poder abordar esta doble vertiente en las prácticas se realizará un tutorial muy exhaustivo en la primera práctica, tanto de *LTSpice IV* como de instrumentación de laboratorio, y así adquirir los conocimientos básicos para poder afrontar las prácticas 2, 3 y 4 con mucha más soltura, por lo que el desarrollo completo de los tutoriales de la primera práctica por parte del alumno será primordial para el resto de la asignatura.
- La segunda parte del manual está enfocada a diseñar, simular y probar físicamente en el laboratorio circuitos digitales. Si en las primeras prácticas propuestas en este manual se ha visto cómo se pueden construir circuitos que, fundamentalmente, representan puertas lógicas, en las últimas prácticas (5 y 6) utilizaremos directamente estas puertas lógicas a modo de “cajas negras” (sin importarnos cuál es el circuito que hay dentro de ellas) para poder realizar circuitos más complejos, de tal manera que diseñaremos y simularemos circuitos combinacionales en la práctica 5, mientras que en la 6 nos centraremos en el diseño, creación del esquemático, simulación y prueba real de un circuito secuencial síncrono. Para todo ello se introduce, de manera previa a las prácticas 5 y 6, un completo tutorial sobre la herramienta de diseño y simulación lógica *Xilinx Foundation 3.1i*, la cual nos servirá para diseñar y simular todos estos circuitos digitales que tendremos elaborados, además de que nos permitirá introducirlos en un dispositivo digital programable como es una CPLD, sobre la cual probaremos físicamente el funcionamiento de nuestro diseño.

A continuación detallamos cada una de las prácticas presentes en este manual, para que se entienda el contenido y objetivos de las mismas.

- Práctica 1. El objetivo de esta primera práctica, de las más importantes de todas, es que el alumno coja la suficiente destreza en el manejo del simulador de circuitos *LTSpice IV* así como en el uso del instrumental de laboratorio, realizando una serie de simulaciones de circuitos en el primer caso, y una batería de experimentos y medidas básicas con los instrumentos del laboratorio en el segundo. En esta primera práctica se presentará inicialmente un exhaustivo y detallado tutorial sobre el manejo del software de simulación *LTSpice IV*, explicando cómo calcular el punto de operación de circuitos básicos formados por fuentes y por resistencias, así como el cálculo de la característica de transferencia en un divisor de tensión y la resolución de un circuito en transición (es decir, un circuito cuyas tensiones e intensidades cambian a lo largo del tiempo), siendo necesarios todos estos conocimientos para la realización de las prácticas 2, 3 y 4. En la segunda parte de la práctica 1 tendremos otro extenso tutorial sobre el manejo de la instrumentación de laboratorio, esto es, fuente de alimentación, generador de funciones y osciloscopio, básicamente. Se presentarán diversos ejercicios prácticos con los que el alumno tomará la suficiente destreza para llevar a cabo las medidas que posteriormente, en las prácticas 2, 3 y 4, tendrá que realizar (medida de tiempos, obtención de la característica de transferencia, etc.), guiando de una forma muy detallada, paso a paso, las

acciones que son necesarias seguir para llegar a obtener los resultados solicitados. Es fundamental que el alumno tome destreza en realizar estos pasos porque, a partir de la práctica 2, ya no se le detallarán en el enunciado de la práctica dado que se sobreentiende que éste sabe realizarlos ya de la práctica 1. Será necesario también realizar una lectura de los apéndices A, B y C del presente manual, relacionados con el simulador *LTSpice IV* y con la instrumentación de laboratorio.

- Práctica 2. En esta práctica se propone estudiar el diodo, un primer dispositivo semiconductor de comportamiento sencillo. Se realizarán diversas simulaciones con *LTSpice IV*, obteniendo una serie de resultados los cuales posteriormente los reproduciremos en el laboratorio montando físicamente los circuitos, y debiendo contrastar los resultados de las simulaciones con los del montaje real. Las medidas a realizar serán similares a las ya hechas en la práctica 1, es decir, obtención de una característica de transferencia, y medidas de dos tensiones simultáneas en función del tiempo, por lo que los pasos para obtener los resultados serán prácticamente los mismos se llevaron a cabo en esa primera práctica.
- Práctica 3. En esta práctica caracterizaremos un transistor bipolar NPN, implementando para ello un inversor lógico como circuito básico para observar su funcionamiento. De nuevo tendremos que realizar simulaciones sobre dicho circuito utilizando *LTSpice IV*, y posteriormente montaremos físicamente en el laboratorio el circuito utilizando componentes discretos, debiendo una vez más contrastar los resultados de la simulación con los del montaje real. Las medidas a obtener serán similares a las de la práctica anterior, si bien adicionalmente realizaremos medidas de tiempos de retardo, de subida y de bajada de las señales de salida del inversor (medidas ya explicadas en la práctica 1).
- Práctica 4. Esta práctica será muy parecida a la anterior, pero ahora tomando el inversor MOSFET como dispositivo semiconductor a estudiar. Para ello simularemos y posteriormente construiremos en el laboratorio 2 puertas lógicas, un inversor de la familia lógica NMOS y otro de la familia CMOS, para comparar sus comportamientos. Las medidas a realizar serán las mismas que en la práctica 3 (característica de transferencia y medida de tiempos, básicamente), si bien ahora también indicaremos las instrucciones de cómo medir el consumo en los mismos (es decir, la potencia consumida), para que el alumno saque sus propias conclusiones en cuanto al gasto energético de los diferentes circuitos dependiendo de cómo están contruidos (es decir, dependiendo de a qué familia lógica pertenecen).
- Tutorial de *Xilinx Foundation 3.1i*. En este punto comienza la segunda parte del manual, introduciendo un tutorial de manejo de la herramienta de diseño y simulación digital *Xilinx Foundation 3.1i*, la cual será necesaria utilizar para las prácticas 5 y 6, dado que a partir de este momento realizaremos prácticas basadas en circuitos digitales. En este extenso tutorial veremos todos los pasos para crear un proyecto, introducir un diseño digital en el mismo, simularlo, y programarlo en la placa de la CPLD para su verificación real.
- Práctica 5. Esta práctica está pensada para que el alumno adquiera los conocimientos suficientes para el diseño de circuitos combinacionales, teniendo que implementar un decodificador de código BCD a 7 segmentos para poder apreciar visualmente, a través de una placa específica en el laboratorio, el número del 0 al 9 que él mismo introduzca en binario a través de unos microinterruptores. Dicho circuito será diseñado y simulado con la herramienta *Xilinx Foundation 3.1i*, y posteriormente será probado en el laboratorio sobre la placa de la CPLD. Además, el diseño será implementado mediante diversas técnicas, teniendo que aplicar el alumno sus conocimientos teóricos del Álgebra de Boole.
- Práctica 6. En esta práctica final se pretende que el alumno diseñe en la herramienta *Xilinx Foundation 3.1i* un circuito digital que genere una secuencia de números del 0 al 9 ordenada de dos maneras posibles, de forma que uniendo esta práctica con la anterior se tendrá un sistema que, una vez implementado en la placa de la CPLD, se verá en un display de 7 segmentos dicha secuencia.

TUTORIAL XILINX FOUNDATION 3.1i.

Material Necesario

- Ordenador Personal
- Software Xilinx Foundation 3.1i

Objetivos

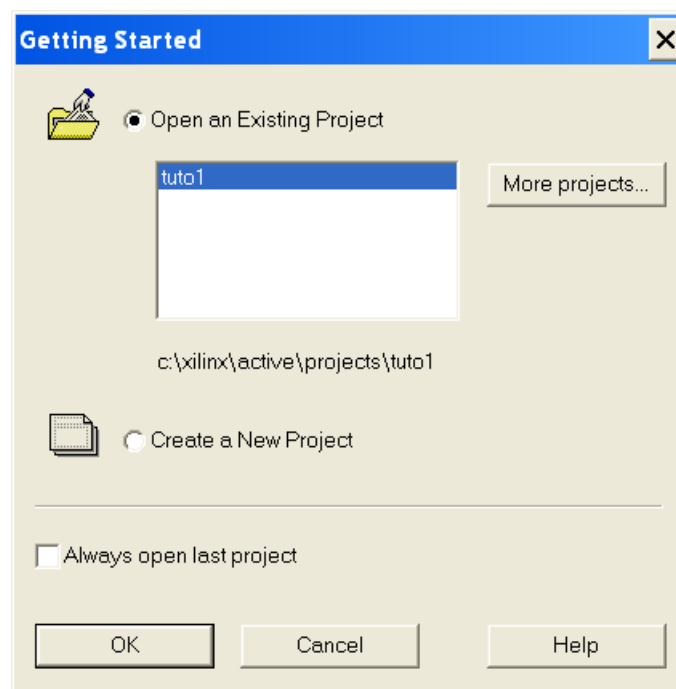
- Conocer el manejo del programa Xilinx Foundation 3.1i para realizar captura de esquemáticos de diseños digitales.
- Aprender a crear Macros de circuitos para ser utilizadas posteriormente.
- Realizar simulaciones funcionales y temporales sobre los circuitos diseñados para comprobar su correcto funcionamiento lógico.
- Implementar el diseño para probarlo posteriormente sobre una placa con una CPLD real.
- Aprender a conectar y programar una placa con un dispositivo CPLD.

Parte 1. Iniciar el “Project Manager” del Xilinx Foundation 3.1i.

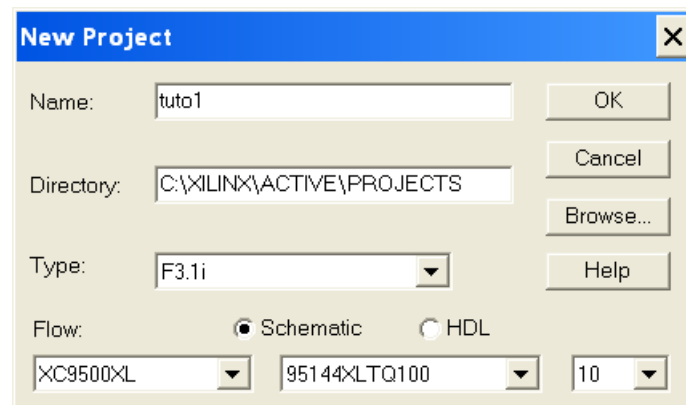
Para comenzar el “Manejador de Proyectos”, se hará doble “click” en el icono “Project Manager”:



Cuando se inicia, aparece un cuadro de diálogo que nos permite seleccionar un proyecto ya existente o crear uno nuevo:



En nuestro caso, para este tutorial, seleccionaremos “Create a New Project” y pulsaremos OK. Aparecerá la siguiente ventana:



Rellenamos el nombre del proyecto, que no debe exceder de los 8 caracteres ni debe empezar por un número, y el resto de parámetros debemos ponerlos como aparecen en la figura, si bien no cambiamos ni el “Directory” ni el “Type” ni el “Flow”, solamente modificaremos los 3 menús desplegables finales:

- Tipo de dispositivo digital a utilizar, en nuestro caso una CPLD de la familia XC9500XL.
- Modelo concreto de dispositivo a utilizar y encapsulado correspondiente, en nuestro caso la CPLD 95144XL, con encapsulado TQ100.
- Grado de velocidad de la CPLD: será el retraso combinacional entre las patillas de entrada hasta cualquier patilla de salida, en nuestro caso 10 (valor expresado en nanosegundos).

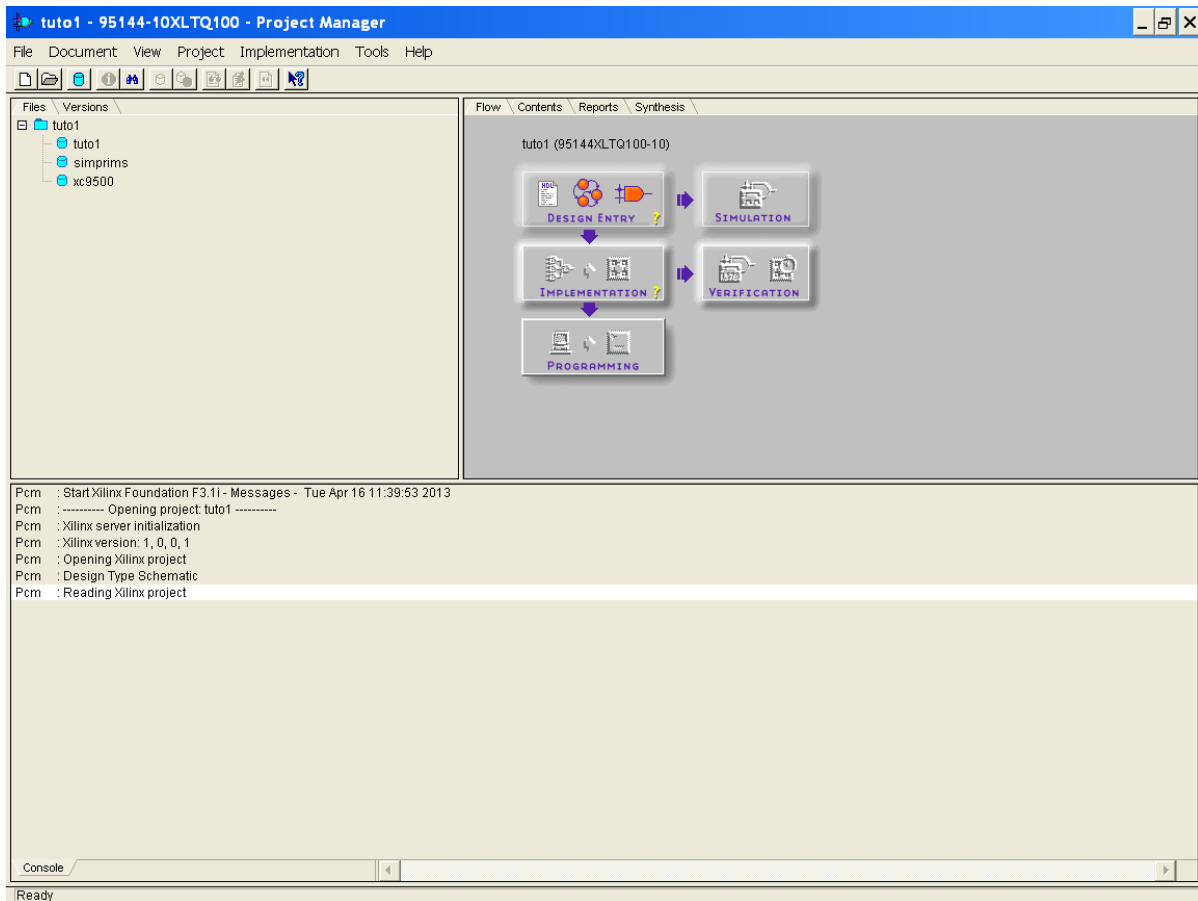
Pulsamos OK, se creará el nuevo proyecto y se mostrará en la ventana del manejador de proyecto.

Para una información más detallada acerca del manejador de proyecto, mirar la ayuda online seleccionando “Help → Foundation Help Contents → Project Manager”.

1.1. El Manejador de Proyecto.

El manejador de proyecto controla todos los aspectos del diseño permitiendo acceder a todas las partes del mismo, además de a los ficheros y documentos asociados con el proyecto.

El manejador de proyecto está dividido en tres ventanas. Arriba a la izquierda está el visor de la jerarquía del diseño, el cual muestra los elementos del proyecto. Arriba a la derecha tenemos un conjunto de opciones de diseño, simulación, implementación y programación en forma de iconos, cada uno de los cuales abre una ventana funcional por separado. La tercera ventana, en la parte inferior del manejador de proyecto, es la consola de mensajes y muestra los mensajes de estado, errores, avisos y es actualizada durante todas las acciones del proyecto.

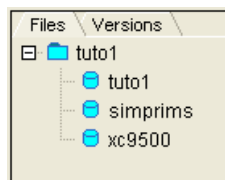


1.2. Librerías del proyecto.

Para diseños basados en esquemáticos, cuando creamos un nuevo proyecto se añaden automáticamente tres librerías al proyecto:

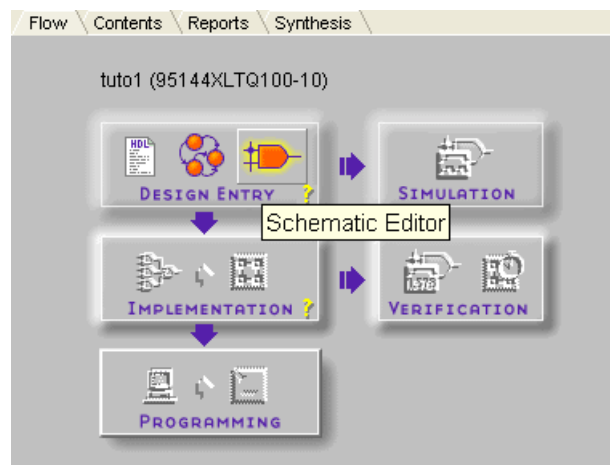
- La librería del proyecto (con el mismo nombre del proyecto)
- La librería SIMPRIMS (para simulación).
- La librería de la familia del dispositivo basada en la familia que se ha seleccionado (por ejemplo XC9500).

Todas las librerías del proyecto están listadas en la etiqueta 'Files' del manejador de proyecto:



Parte 2. Diseño de esquemáticos.

Para inicial el Editor de Esquemáticos (*"Schematic Editor"*) nos situaremos en la pestaña *"Flow"* y pulsaremos el botón correspondiente dentro de la zona llamada *"Design Entry"*:

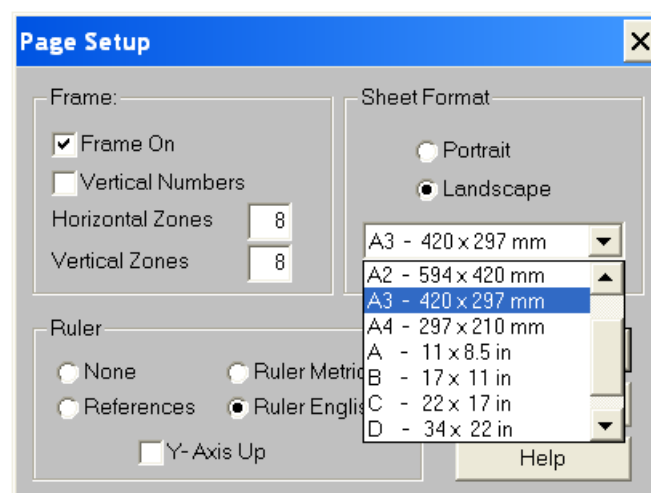


El editor de esquemático abre la hoja con el nombre de proyecto más un número, en nuestro caso será *"tuto11.sch"*.

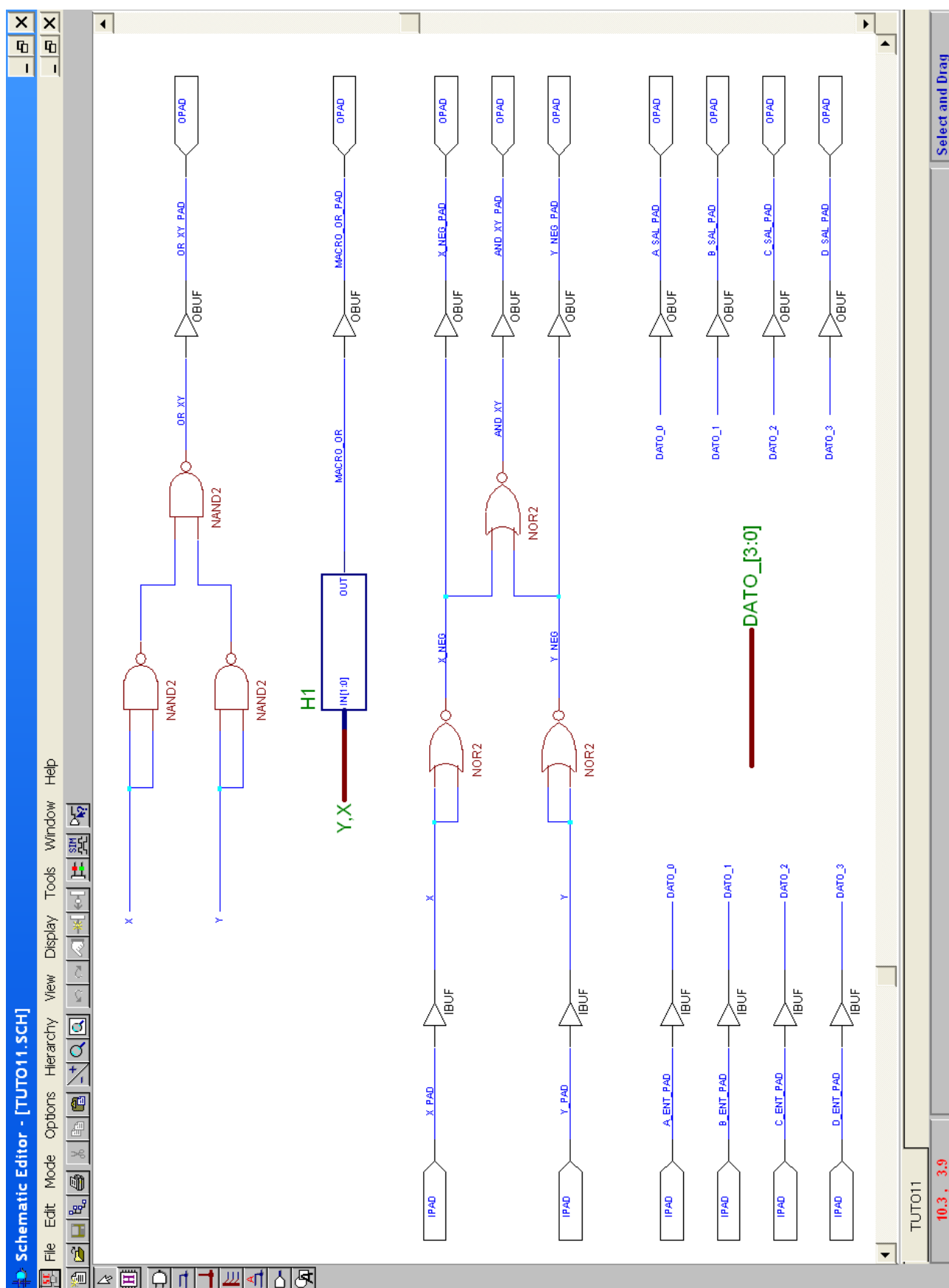
A lo largo del presente tutorial vamos a realizar el diseño de un simple ejemplo. La propuesta de dicho ejemplo se muestra en las figuras de las páginas siguientes. Este ejemplo consiste en:

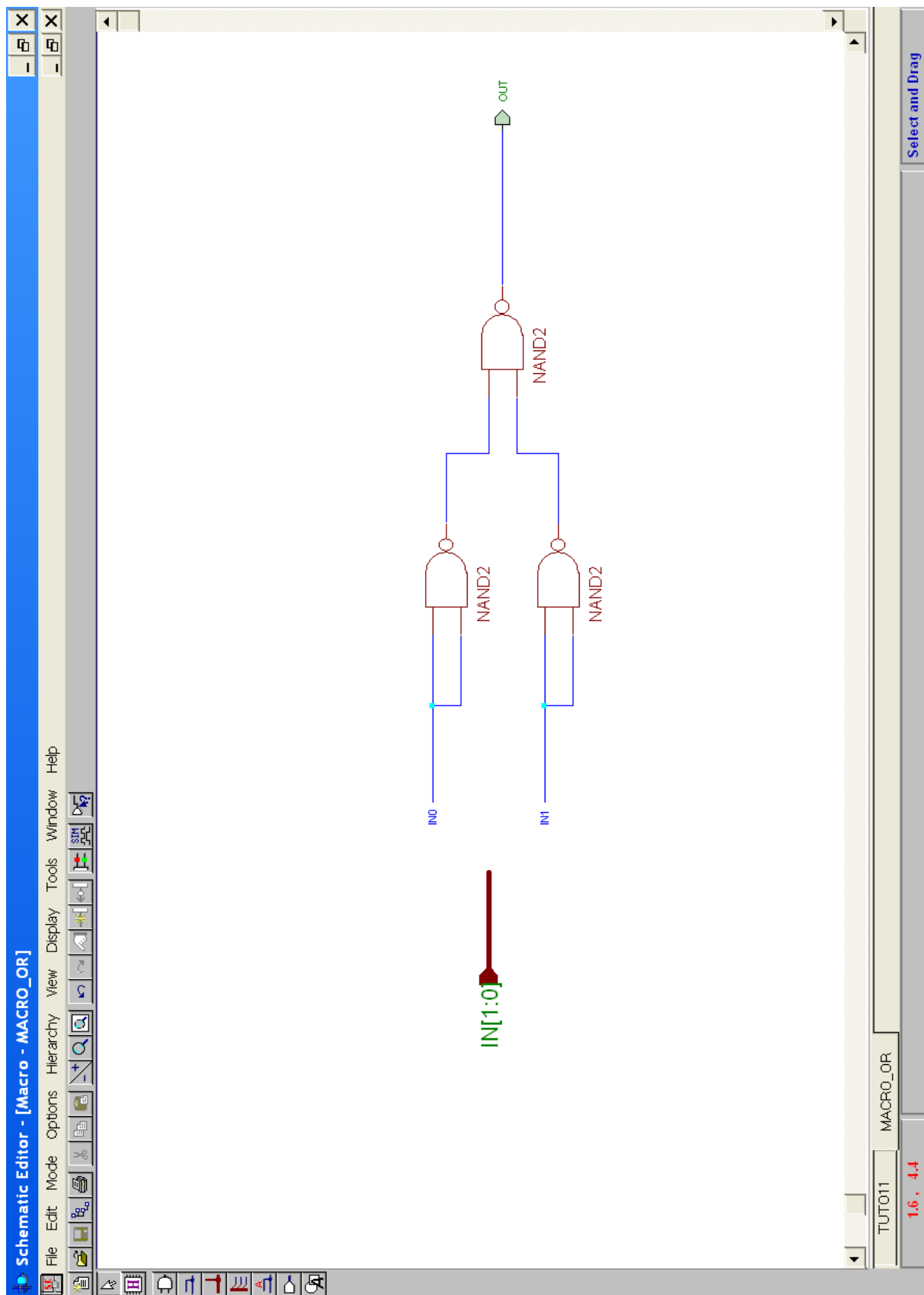
- Función OR con puertas NANDs
- Función AND con puertas NORs
- Una macro (bloque funcional) que implementa una función OR (con puertas NANDs)
- Un BUS

Es importante mencionar que, por defecto, *Xilinx Foundation* abre una hoja de trabajo muy grande (una hoja de tamaño *"D"* en medida americana), por lo que es recomendable trabajar con una hoja un poco más ajustada al diseño que pretendamos realizar para no tener que estar usando constantemente el *"Zoom"*. Para ello, pincha en la opción *"File → Page Setup"*, y escoge una hoja tamaño europeo (A3 ó A4 será suficiente), mediante el siguiente menú:




Si necesitas parar en algún momento el tutorial debes salvar tu trabajo en el esquemático seleccionando *"File → Save"* en el menú del editor de esquemáticos.



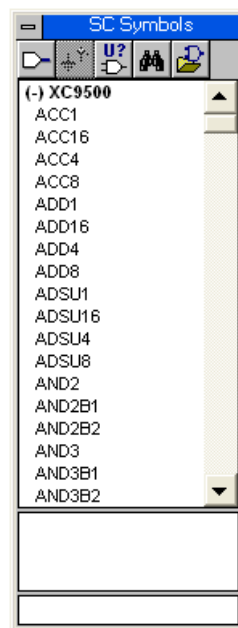


2.1. Añadir componentes de la librería al esquemático.

Los componentes de todas las librerías del proyecto (excepto SIMPRIMS) están disponibles en la ventana de símbolos del esquemático para ser incluidos en el diseño. Los componentes disponibles se muestran ordenados de forma alfabética en cada librería.


La ventana de símbolos puede ser mostrada de dos formas distintas: seleccionando “*Mode → Symbols*”, o pulsando el botón “*Symbols Toolbox*” en la barra de herramientas vertical que se muestra en el lado izquierdo del editor de esquemáticos: 

Pulsando dicho botón se abre la ventana “*SC Symbols*” y muestra las librerías y sus correspondientes componentes:



El componente que vamos a insertar es una puerta NAND de dos entradas. Buscamos este componente en la lista de la ventana anterior y lo seleccionamos pulsando sobre él, momento en el cual se muestra una descripción del mismo en la ventana “*SC Symbols*”. Posteriormente movemos de nuevo el ratón a la ventana de esquemáticos, situamos el símbolo en el punto donde se desee colocar y pulsamos el botón izquierdo del ratón para fijar el componente. Si se desea añadir al diseño más componentes iguales, pulsar sobre el componente ya fijado y aparecerá un nuevo símbolo del componente que podrá ser situado en otro lugar.

Existe una serie de componentes de obligatoria utilización para la herramienta *Xilinx Foundation*. Para las señales de entrada de la CPLD son el IPAD y el IBUF (en ese orden), y para las señales de salida el OBUF y el OPAD (también en ese orden).


La ventana “*SC Symbols*” se podrá quitar pulsando de nuevo sobre el botón “*Symbols Toolbox*”  en la barra de herramientas vertical.

2.2. Corregir errores.

Si se ha cometido algún error situando los componentes, estos pueden ser movidos o borrados fácilmente:

- Pulsar la tecla "Esc" del teclado para salir del modo símbolo.
- Seleccionar el componente que queramos mover o borrar haciendo "click" con el ratón sobre el mismo. Asegurarse previamente de que ningún otro componente ha sido seleccionado (pulsando sobre un área en blanco del esquemático se deselectionará todo)
- Pulsar sobre el componente seleccionado, y sin levantar el dedo del botón arrastramos el componente para situarlo correctamente, o pulsar la tecla "Supr" o el icono "Cut" de la barra de herramientas para borrar el componente
- Tras recolocar el o los componentes, podemos seguir poniendo objetos de la librería si volvemos a hacer "click" sobre la ventana "SC Symbols"

2.3. Dibujar y etiquetar cables.

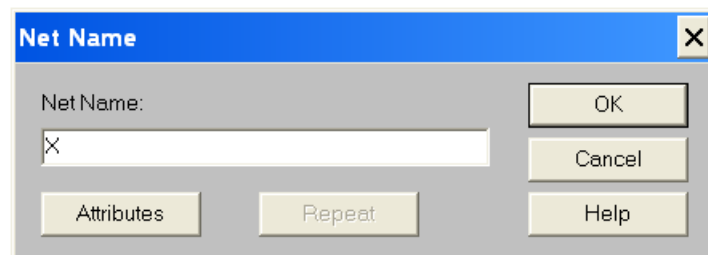
Será necesario utilizar el icono "Draw Wires"  en la barra de herramientas vertical para dibujar cables y así conectar unos componentes con otros del esquemático. Para ello pulsaremos sobre el pin de inicio y luego haremos "click" sobre el pin final, y la línea se dibujará automáticamente entre los mismos. Para salir del modo de dibujo de cables pulsar la tecla "Esc".

Nota: Se puede especificar la forma de la línea moviendo el ratón en la dirección que se quiera y haciendo "click" en cada esquina de 90° que queramos formar. Si se comete algún error en el dibujo de la línea, se puede salir del dibujo de la actual línea pulsando la tecla "Esc".

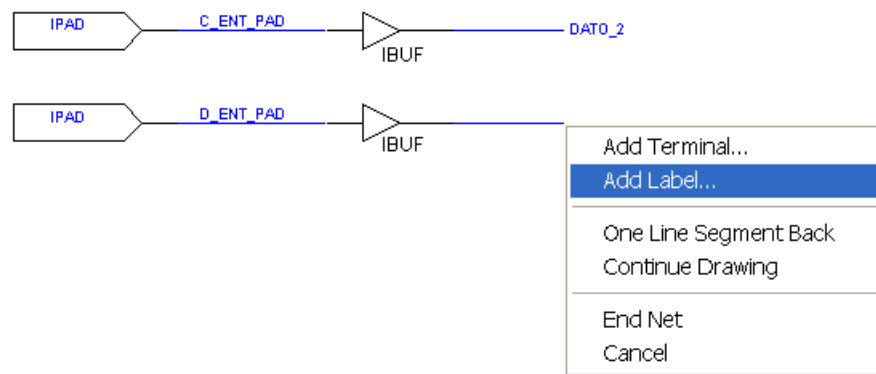
Etiquetar o ponerle nombre a los cables nos sirve para conectar componentes sin tener que tender un cable de pin a pin físico de los mismos, de manera que dos cables con el mismo nombre estarán conectados entre sí en el mismo esquemático.

Para nombrar un cable tenemos dos opciones:

- Si el cable está ya dibujado entre dos pines, podemos realizar un doble "click" sobre el mismo y aparecerá la siguiente ventana; en el campo "Net Name" podremos teclear el nombre que deseamos ponerle, tras lo cual pulsaremos "OK":



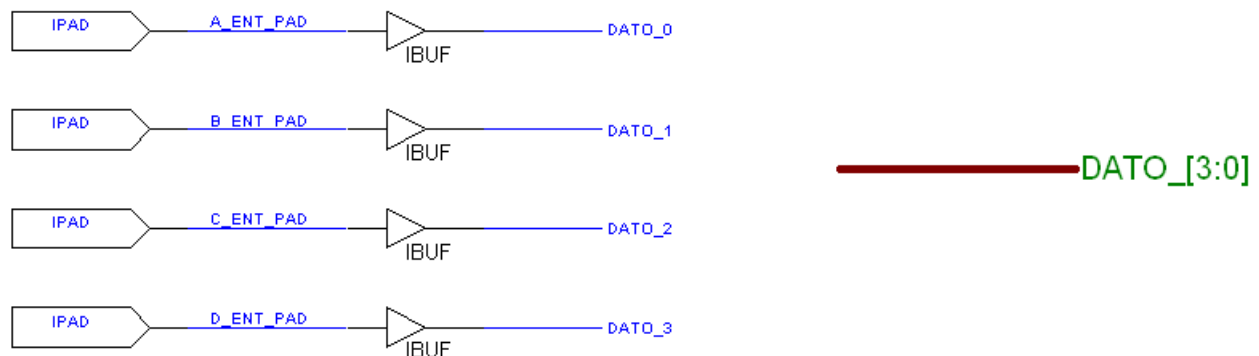
- Si el cable sólo está conectado a 1 pin, podemos colocar el nombre en el extremo de dicho cable cuando lo estemos dibujando. Para ellos haremos "click" con el botón izquierdo del ratón en el pin de inicio, movemos el ratón para crear un cable de la longitud deseada, y directamente pulsaremos con el botón derecho del ratón, apareciendo el menú mostrado a continuación, en el cual seleccionaremos "Add Label...", y de nuevo surgirá una ventana como la anterior para teclear el nombre, en el ejemplo mostrado sería "DATO_3", y pulsamos "OK". Dicho nombre quedará situado al final del cable.



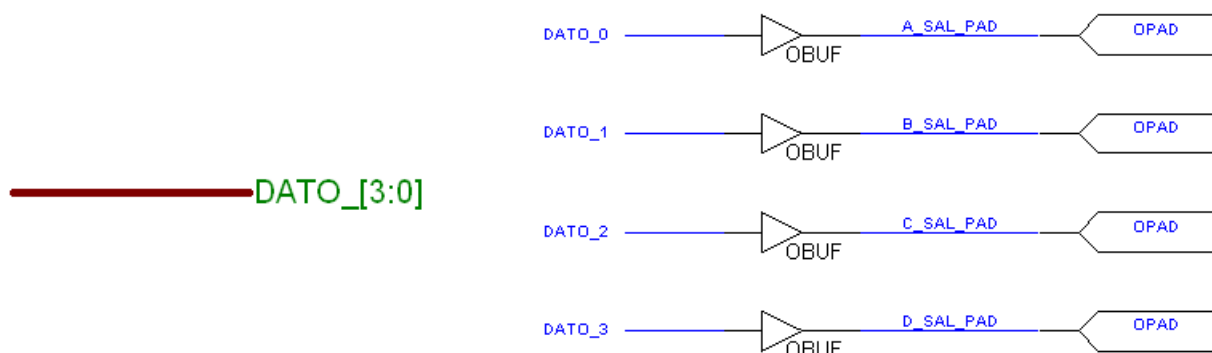
2.4. Dibujar y etiquetar buses.


Entendemos un *bus* como un conjunto de cables reunidos gráficamente en un solo hilo de aspecto más grueso que el habitual, de manera que todos esos cables individuales llevan una información que está relacionada entre sí de alguna manera. De esta forma conseguimos representar fácilmente un flujo de datos que posea más de un bit.

Así, en el ejemplo del tutorial, hay un bus llamado “DATO_[3:0]”, y cada una de las cuatro señales que componen dicho bus son nombradas individualmente como “DATO_0”, “DATO_1”, “DATO_2” y “DATO_3”. Por tanto, cuando queramos crear un bus nombraremos las señales individuales con un nombre común (en el ejemplo, “DATO_”) y le añadiremos un índice que nos dirá la posición de dicha señal dentro del bus, como vemos en el tutorial:

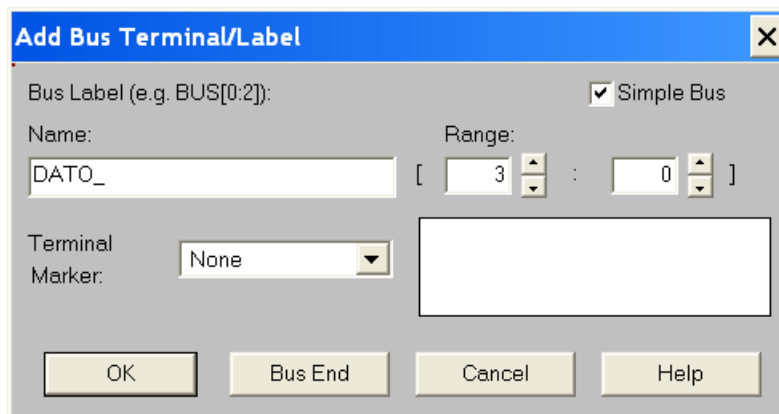


Se procede de la misma manera cuando se quieren extraer señales correspondientes a un bus, no hay más que etiquetar el cable con el nombre del bus seguido del índice de la señal dentro del bus, como vemos en el ejemplo:



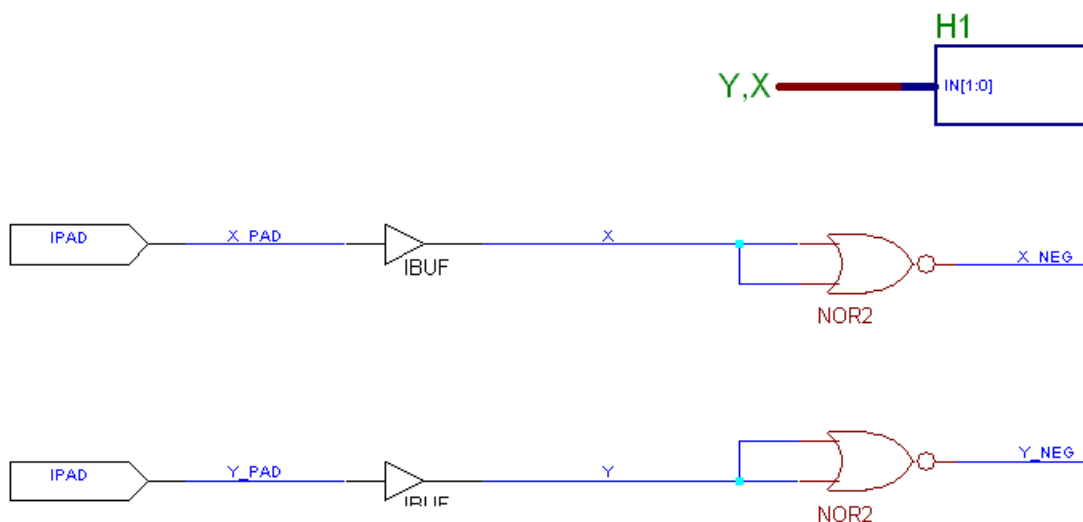
Para dibujar un bus lo haremos igual que para dibujar un cable, pero utilizando el botón “Draw buses” . La manera de etiquetarlos será similar a los cables, si bien hay dos opciones para crear buses:

1.- La que ya hemos explicado, etiquetando señales individuales con un nombre común pero con diferentes índices al final (bus simple). Para crear el nombre del bus haremos “click” donde queramos conectar el bus inicialmente, moveremos el ratón hasta la posición final deseada, y haremos otro “click” pero con el botón derecho del ratón, momento en el que aparecerá una ventana con diversas opciones, una de ella será “Add Bus Label”, la cual escogemos y nos lleva a la siguiente pantalla:



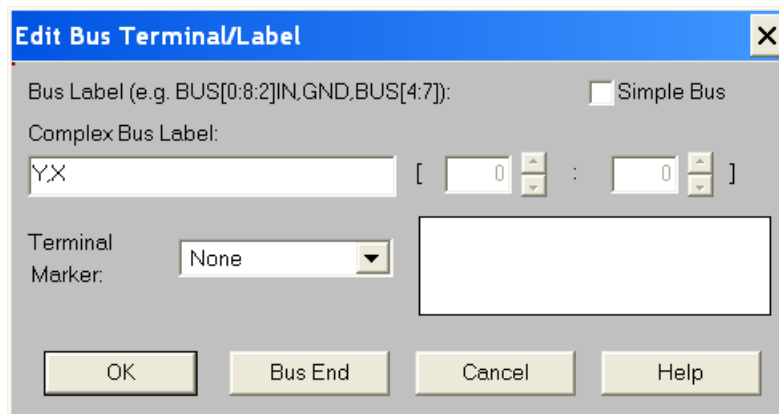
En nuestro ejemplo, pondríamos en el recuadro “Name” el nombre común del bus (“DATO_”), y en “Range” sería necesario especificar el índice inicial y final del bus, en nuestro caso de “3” a “0” (normalmente se pone siempre primero el índice más significativo). Pulsamos “OK” y se creará el bus a partir de las señales indicadas.

2.- Otra opción para crear un bus es la de combinar señales que no tienen un nombre común. En este caso no estaríamos creando un bus de tipo simple, sino que sería un bus complejo. Un ejemplo de este tipo de bus lo encontramos en el tutorial:




Vemos que deseamos crear un bus complejo que reúna a las señales etiquetadas como “X” e “Y”, e introducirlo como un solo bus en el módulo “H1”, de manera que la “X” ocupe la posición con índice 0, y la “Y” la posición con índice 1, como se aprecia en la figura. La forma de construir dicho bus sería con la misma opción que en el caso anterior (dibujamos el bus, pinchamos con el botón derecho del ratón y escogemos “Add Bus Label”), pero tendríamos que desmarcar la casilla llamada “Simple Bus”, y a continuación teclear los nombres de las señales individuales que conforman el bus, separadas por comas,

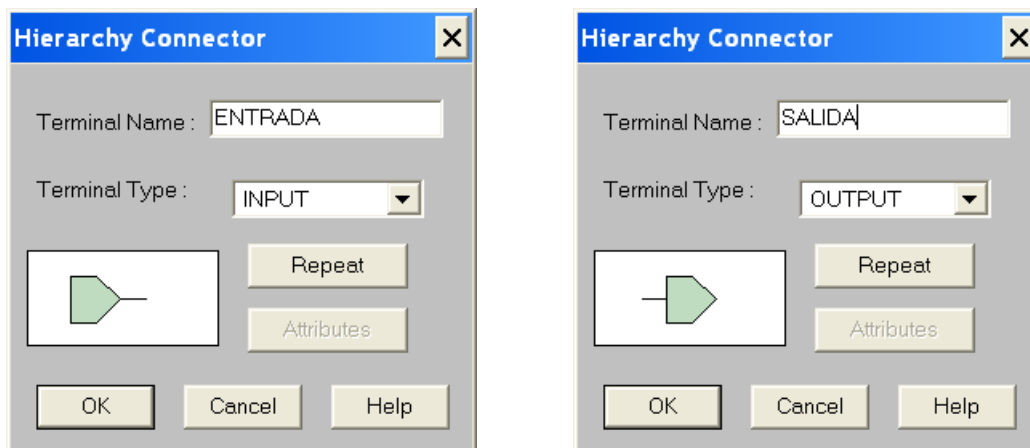
dentro del recuadro titulado “Complex Bus Label” (en nuestro caso, “Y,X”), pulsando “OK” una vez finalizado:




2.5. Crear una macro.

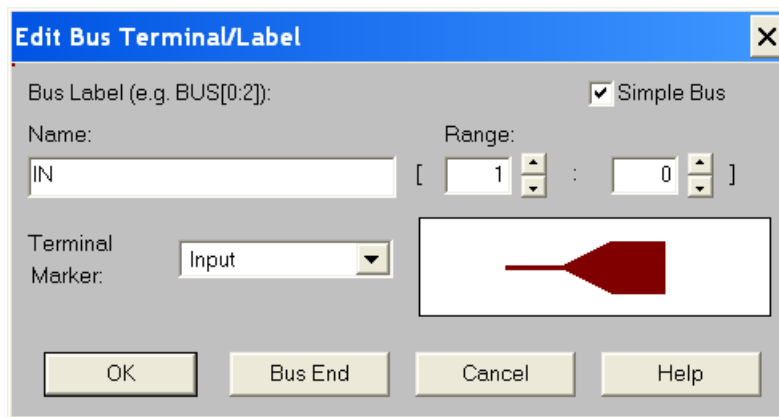
Una macro es un circuito encapsulado en una caja o símbolo creado por nosotros, que realiza una función determinada, y que puede ser incluido en posteriores diseños como cualquier otro bloque o puerta lógica que ya conocemos, conectándolo mediante cables a otros módulos o elementos del circuito.

Para hacer una macro, primero debemos crear una nueva hoja de trabajo dentro del “Schematic Editor” con el menú “File → New Sheet” y a continuación introducimos el esquemático con el diseño del circuito incluyendo las puertas lógicas necesarias y sus conexiones entre sí. Para comunicar dicho bloque con los otros será necesario definir unas señales de entrada y salida del circuito, para lo cual hay que utilizar conectores de jerarquía , del tipo “INPUT” para las entradas, y “OUTPUT” para las salidas:



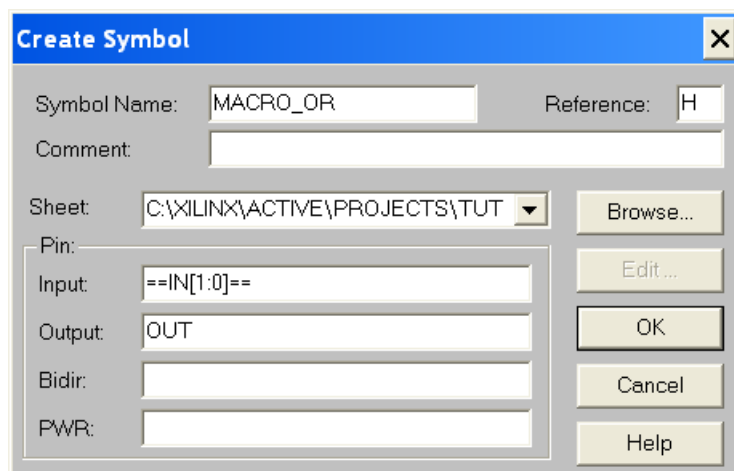
En una macro no incluiremos, salvo casos excepcionales, los componentes IPAD, IBUF, OPAD y OBUF, ya que estos se usan para conectar nuestro circuito con el exterior del integrado (la CPLD), y estos suelen ponerse en el esquemático de nivel superior del diseño.

Si las señales de entrada y salida del bloque son un bus, en lugar de señales individuales, primero dibujaremos dicho bus con el símbolo que ya conocemos () pero al finalizar el dibujo del mismo con el botón derecho del ratón seleccionaremos la opción “Add Bus Terminal”, y aparecerá una pantalla como la siguiente:




En nuestro ejemplo del tutorial, vemos cómo estaríamos definiendo para la macro “MACRO_OR” un bus simple de entrada llamado “IN”, de 2 bits de ancho (el rango “Range” está puesto del índice 1 al 0).

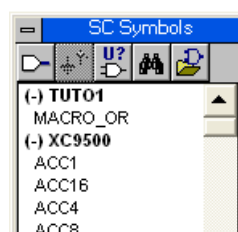
Una vez completado el circuito, la macro se crea con la opción de menú “*Hierarchy → Create Macro Symbol from Current Sheet*”. Automáticamente se abre una ventana en la que debemos introducir en “Symbol Name” el nombre que vamos a ponerle a la macro, a la vez que chequeamos los pines de entrada y salida del mismo para ver que son los correctos para que el módulo se comunique adecuadamente con el exterior (dentro del apartado “Pin”):



Al pinchar en el botón “OK”, se crea la macro y el programa nos preguntará si queremos editarla. Seleccionaremos que “No”, y se cerrará la hoja en donde estábamos creando este circuito, ya que el mismo se habrá guardado en forma de macro en la librería del proyecto con el nombre que le hayamos dado.



2.6. Utilizar una macro.

Para usar una macro, es decir, poner el circuito que ella representa dentro de nuestro diseño, lo haremos a través de la ventana de símbolos, pulsando :



Podemos tratar a nuestro nuevo símbolo como cualquier otro componente de la librería, situarlo en nuestro diseño y conectarlo al resto de puertas lógicas y demás dispositivos del mismo, como vemos en el ejemplo del tutorial:

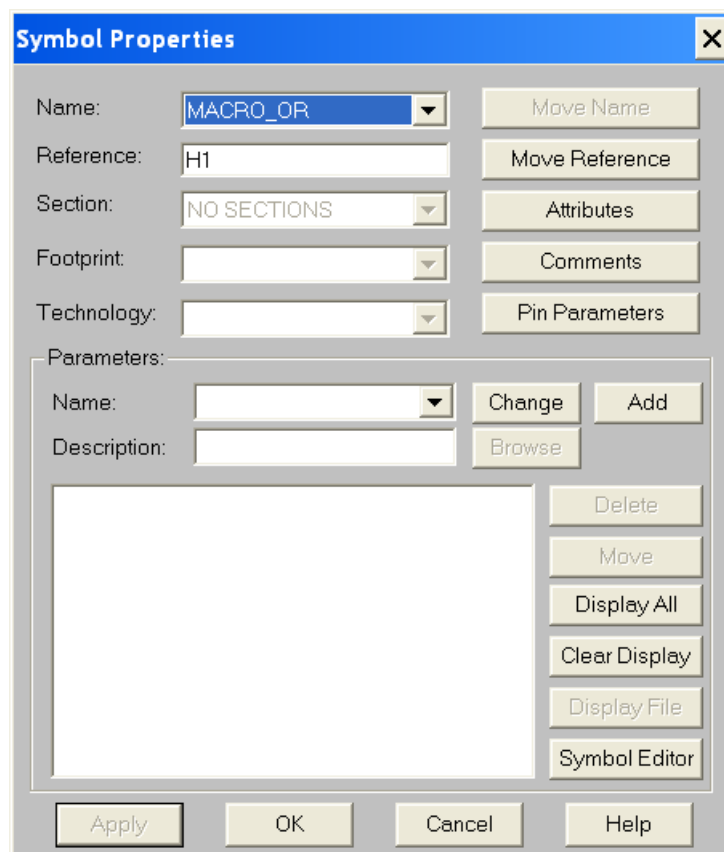


Si una vez utilizada una macro queremos entrar dentro de ella para visualizar el circuito que contiene o incluso modificarlo, pulsaremos sobre el botón con el símbolo  (de “Hierarchy”, “Jerarquía”), y haremos doble “click” sobre la macro en cuestión; se abrirá una nueva ventana con el circuito de la misma, donde podremos realizar los cambios que deseemos, grabarlo, y volver al circuito anterior haciendo de nuevo doble “click” con el mismo símbolo  sobre una zona en blanco de la hoja de trabajo.

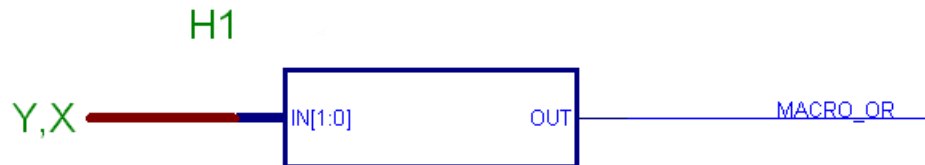
Nota: Si se realiza una modificación en una macro de la manera descrita anteriormente, ésta afectará a todas las instancias de la misma, es decir, que si esa misma macro está colocada en nuestro diseño en diversos sitios los cambios se efectúan en todos los bloques pertenecientes al proyecto actual.

2.7. Editar propiedades de una macro.

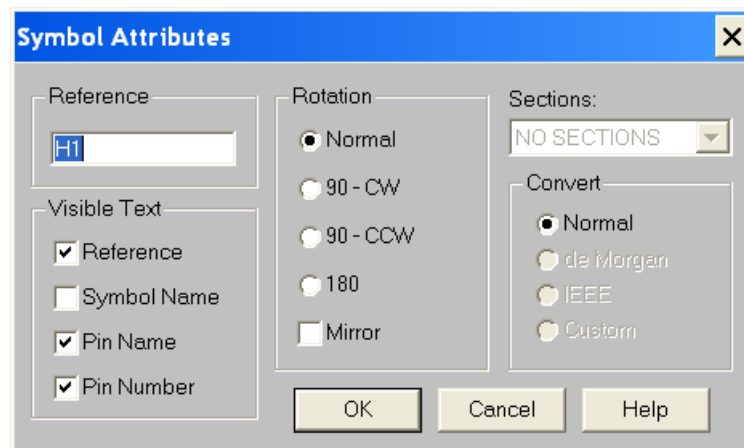
Podemos editar ciertas propiedades de una macro haciendo doble “click” sobre la misma (o bien con el botón derecho del ratón, seleccionando “Symbol Properties”), a través de la siguiente pantalla:



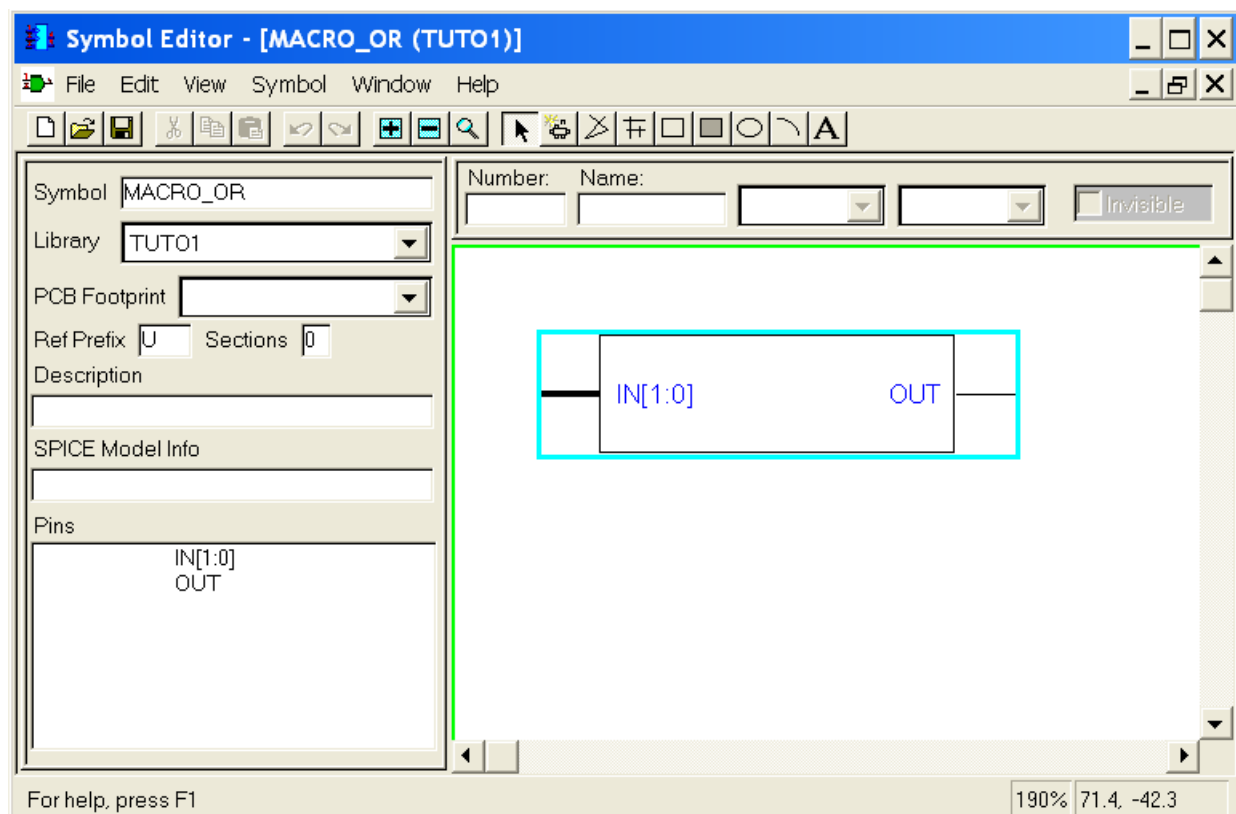
- Mover la referencia (en este caso H1) a otro sitio:



- Cambiar ciertos atributos de visibilidad (por ejemplo, podemos hacer que se vea el “*Symbol Name*”, es decir, que aparezca en pantalla el nombre de la macro):



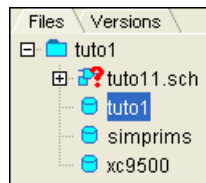
- Llamar al “*Symbol Editor*” para modificar el tamaño o aspecto del símbolo creado para nuestra macro, así como reordenar a nuestro gusto las patillas de entrada o salida (normalmente las de entrada estarán a la izquierda del símbolo y las de salida a la derecha):



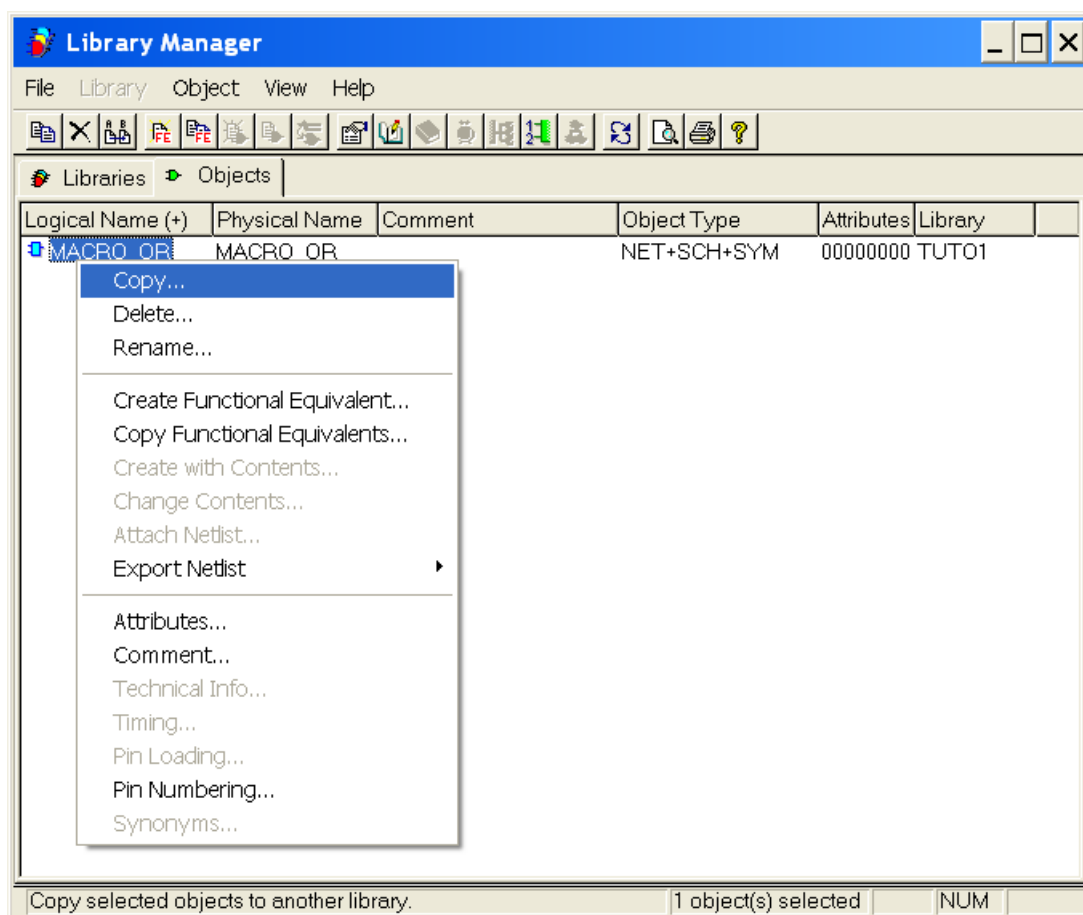
2.8. Copiar una (o varias) macro(s) a otro proyecto.

A veces es necesario copiar una macro de nuestro proyecto a otro, o al contrario. Para realizar este proceso de copia haremos lo siguiente:

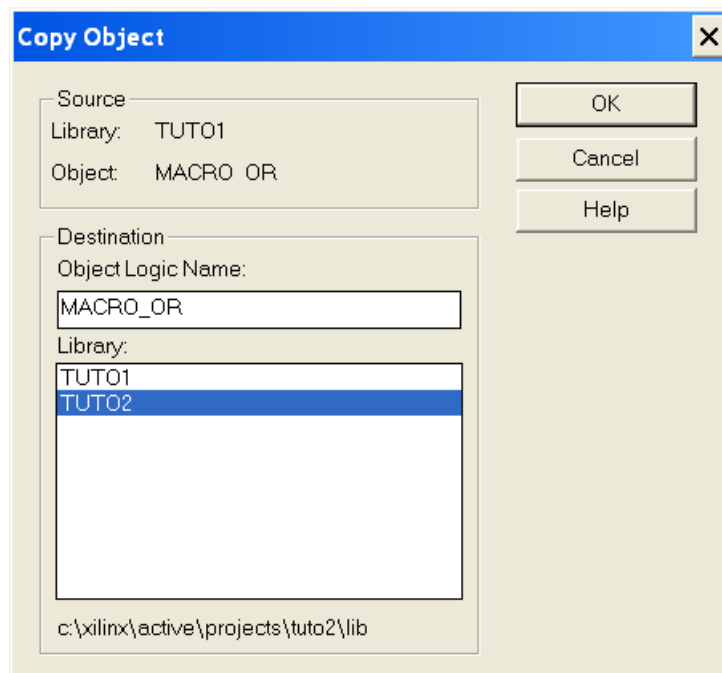
- Abrimos el proyecto que contiene la macro que queremos copiar.
- En el “*Project Manager*”, hacemos doble “click” sobre la librería del proyecto, que tiene el mismo nombre que el proyecto abierto (seleccionada en azul en la siguiente imagen):



- Aparece una ventana con todas las macros creadas en el proyecto. Selecciona las que quieras copiar. Haz “click” con el botón derecho y elige “Copy...”.




- En la siguiente ventana sólo debes seleccionar la librería de destino de la copia en el listado de proyectos que aparece en la parte inferior de la misma (en el ejemplo mostrado sería “TUTO2”), y pulsar el botón “OK”.



- Ya se podrá abrir el proyecto de destino y la macro que se acaba de copiar estará disponible en el mismo. Hay que destacar que si se modifica la macro en uno de los proyectos no afectaría al otro, ya que son copias independientes.

2.9. Guardar el esquemático.

En cualquier momento podemos (y debemos) salvar el esquemático, para lo cual seleccionaremos el menú “File → Save” o pulsaremos sobre el icono “Save”  en la barra de herramientas horizontal.

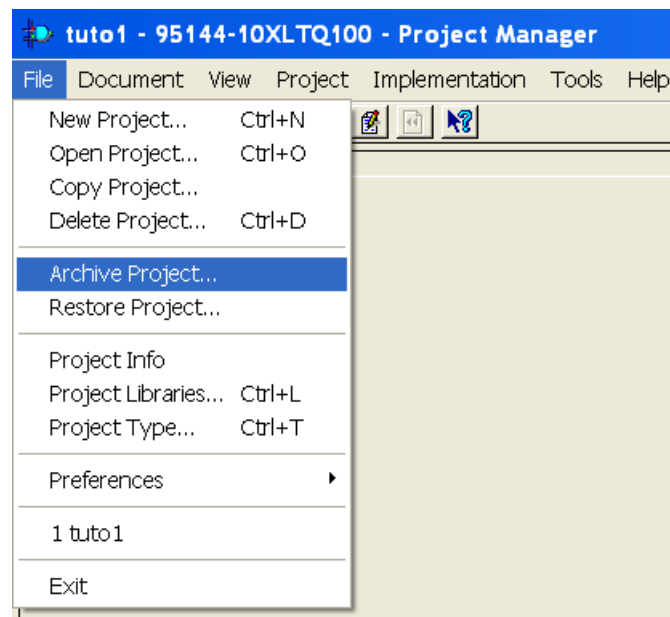
Es necesario recordar que todos los errores, avisos y mensajes de información se muestran en la ventana de mensajes del “Project Manager”. Si se produce algún error, corrígelo y salva de nuevo el esquemático.

2.10. Archivar y Restaurar el proyecto.

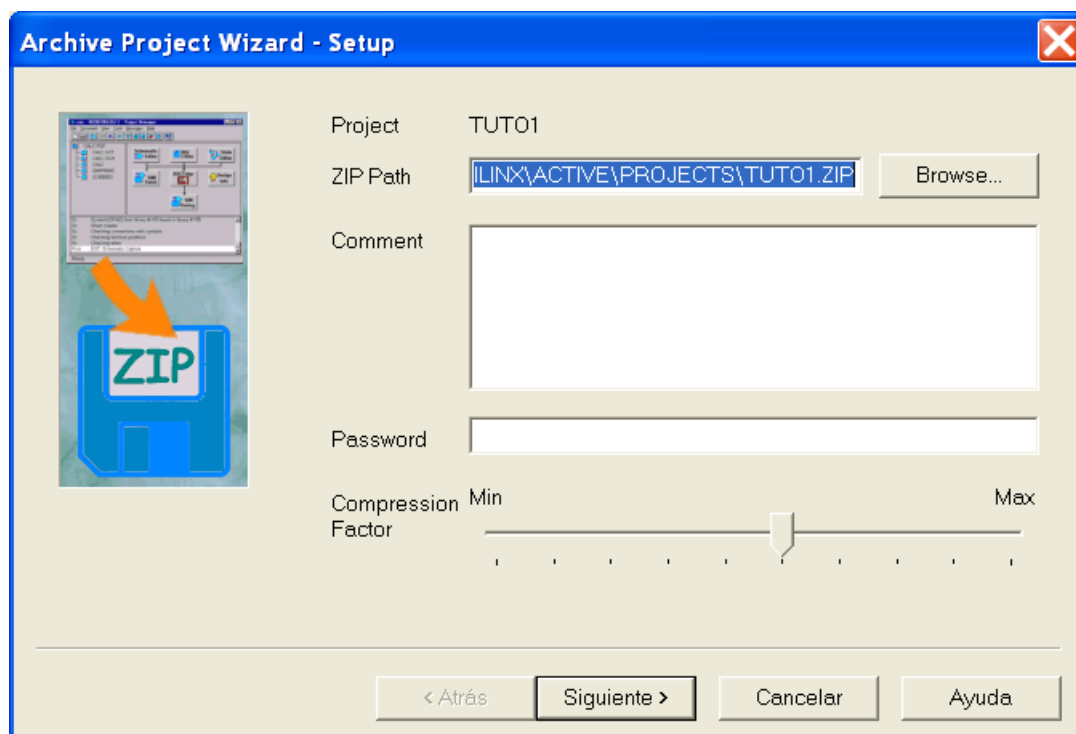
Cuando se va a trasladar el proyecto de un ordenador a otro, o bien cuando se termina una sesión en el laboratorio, es necesario archivar todo el proyecto en un solo fichero ZIP como copia de seguridad del mismo, de manera que se pueda restaurar en otro momento.

La manera correcta de hacerlo es ir al “Project Manager” y, a través del menú “File → Archive Project” y “File → Restore Project”, almacenar o restaurar, respectivamente, el proyecto actualmente en uso.

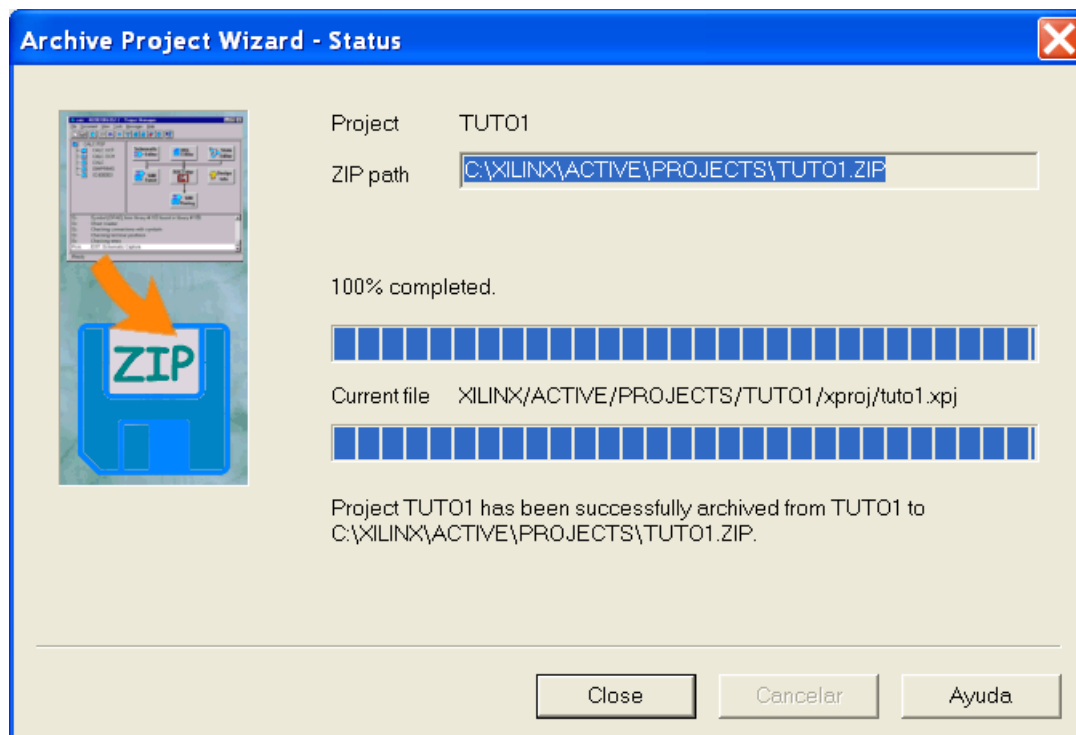
Para archivar el proyecto seleccionamos “File → Archive Project”:



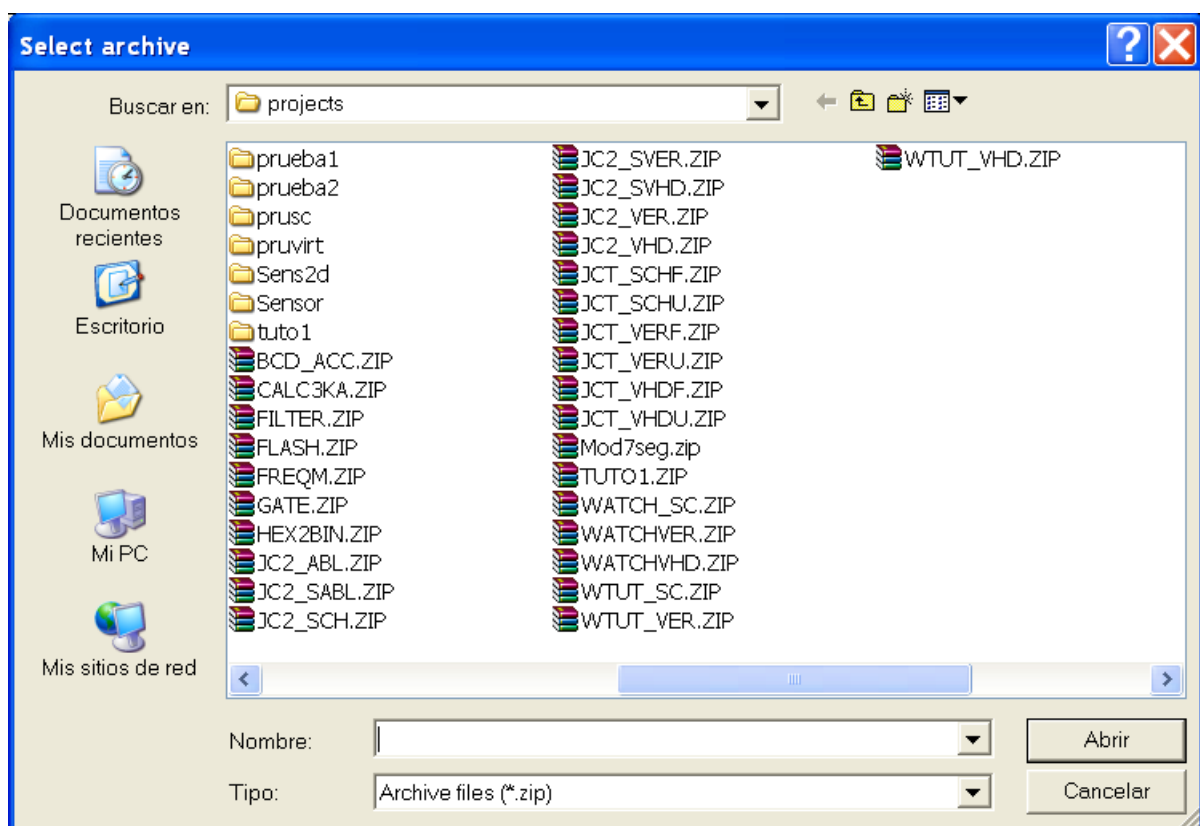
En la ventana que aparece escogemos el directorio y nombre donde vamos a grabar el archivo ZIP con el proyecto completo, así como el nivel de compresión de dicho archivo; pinchamos en “Siguiente”, “Siguiente” y “Start” para comenzar el archivado:



Cuando haya finalizado aparecerá la siguiente pantalla, en la que pulsaremos “Close” y volveremos al “Project Manager”:



Para restaurar un proyecto seleccionamos “File → Restore Project” y aparecerá la siguiente pantalla, donde escogeremos el archivo ZIP de donde lo tengamos guardado previamente, y se descomprimirá en el directorio de proyectos del Xilinx:




Parte 3. Simulación Funcional.

Se debe hacer una simulación funcional del diseño antes de pasar a su implementación física para verificar que la lógica que se ha creado es correcta. *Xilinx Foundation* proporciona un simulador lógico a nivel de puertas, por lo que será necesario aprender su uso para utilizarlo inmediatamente después de haber terminado de crear el diseño con el editor de esquemáticos.

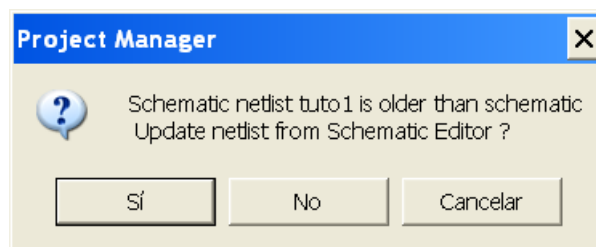
3.1. Iniciar el simulador lógico.


Desde el “*Project Manager*” pulsaremos el botón “*SIMULATION*” que nos lleva al simulador funcional:

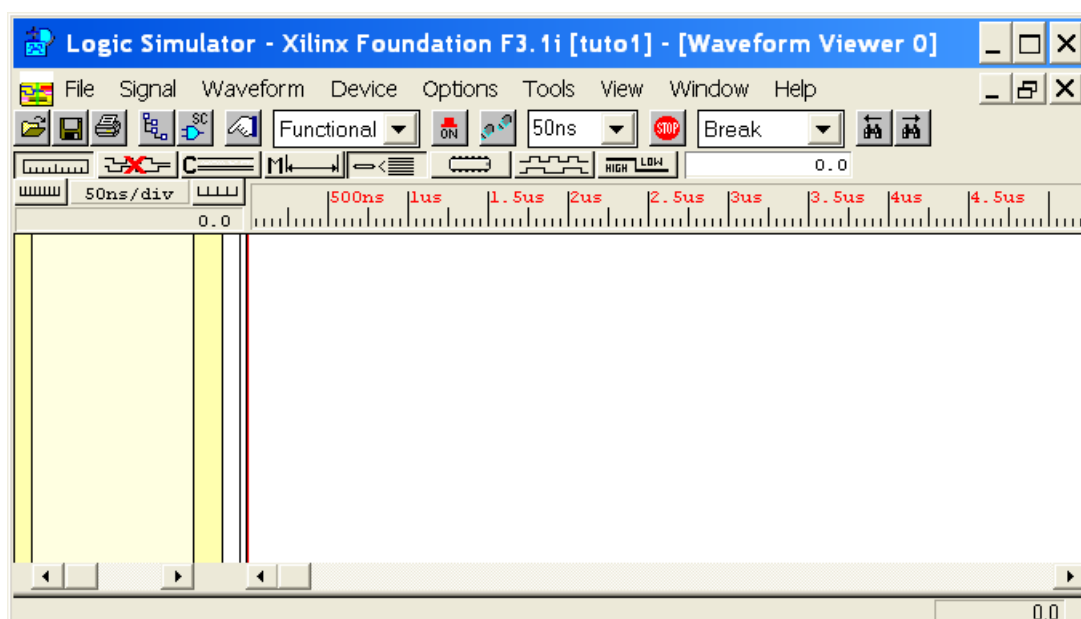


o bien, desde el editor de esquemáticos, podemos pulsar sobre el botón “*SIM*”  para irnos al mismo simulador.

Si lo hacemos de la primera manera, lo más seguro es que el software nos muestre el siguiente aviso:



Con esto nos viene a decir que hay que actualizar la “*netlist*” del diseño (fichero que contiene todos los componentes del mismo y sus interconexiones) antes de iniciar el simulador, por lo que responderemos que “*Sí*” y nos aparecerá la ventana del simulador. Si invocamos al mismo simulador pero desde el editor de esquemáticos con el botón  entonces el software ya se encarga de actualizar la “*netlist*” automáticamente sin preguntarnos, y nos lleva al mismo sitio: el simulador funcional.



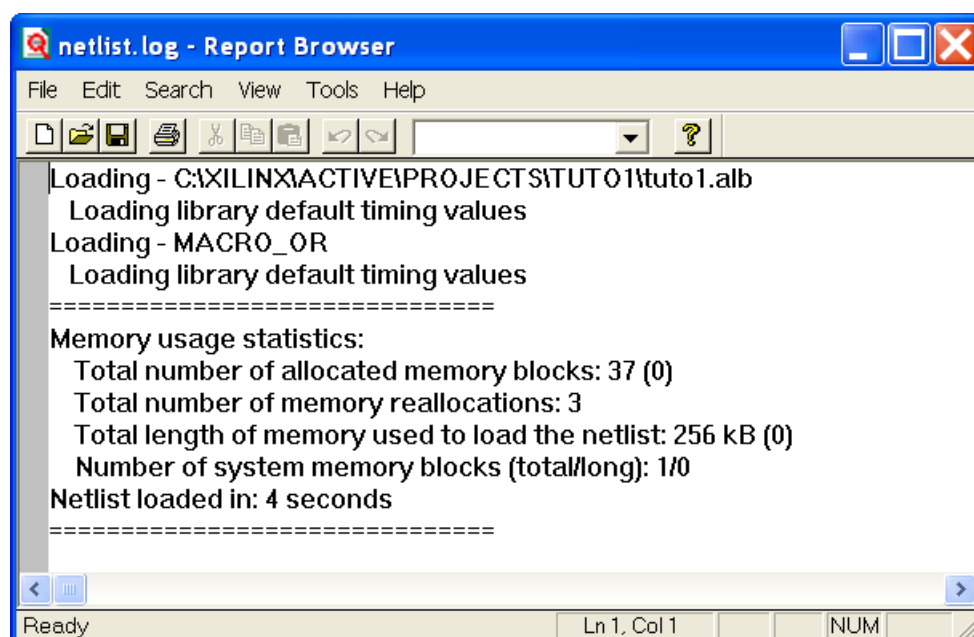
3.2. Comprobar errores en la carga de la “netlist”.

Es frecuente, sobre todo al principio cuando no se tiene mucha práctica, realizar malas conexiones entre dispositivos o no usar los adecuados. Todos estos errores de conexionado, dispositivos erróneos o equivocaciones a la hora de etiquetar cables o buses pueden ser detectados en este paso.

En primer lugar, es conveniente echar un vistazo a la consola de mensajes del “Project Manager”, donde se informa de errores comunes tras la actualización de la “netlist”. En la siguiente ventana vemos el proceso normal de carga de la “netlist” (sin errores) en el simulador funcional.

```
Pcm : Schematic netlist tuto1 is older than schematic
Pcm : Update netlist from Schematic Editor
Sc  : Netlist generation in progress
Sc  : Creating netlist from schematic [C:\XILINX\ACTIVE\PROJECTS\TUTO1\TUTO11.SCH]
Sc  : Netlist created successfully
Pcm : Netlist created successfully
Conv : Scanning design for HDL macros and LogiBLOX symbols
Conv : C:\XILINX\ACTIVE\PROJECTS\TUTO1\TUTO1.alb
Conv : MACRO_OR
Pcm : Execute c:\xilinx\active\exe\simul.exe -p
Pcm : START: Logic Simulator - Xilinx Foundation F3.1i [ ]
Simul : Loading - C:\XILINX\ACTIVE\PROJECTS\TUTO1\tuto1.alb
Simul : Loading library default timing values
Simul : =====
Simul : Memory usage statistics:
Simul : Total number of allocated memory blocks: 37 (0)
Simul : Total number of memory reallocations: 3
Simul : Total length of memory used to load the netlist: 256 kB (0)
Simul : Number of system memory blocks (total/long): 1/0
Simul : Netlist loaded in: 4 seconds
Simul : =====
Simul : Netlist loaded successfully
```

En segundo lugar, tenemos otro sitio donde buscar los errores del circuito, de hecho la gran mayoría de ellos los podremos localizar en este apartado. Una vez cargado el simulador funcional, nos vamos al menú “Tools → View Netlist Log”, abriéndose una ventana de texto con información sobre nuestro diseño. Si hay algún error de conexionado, etiquetas mal nombradas, etc., aparecerá aquí:



```
netlist.log - Report Browser
File Edit Search View Tools Help
[Icons] [?]
Loading - C:\XILINX\ACTIVE\PROJECTS\TUTO1\tuto1.alb
Loading library default timing values
Loading - MACRO_OR
Loading library default timing values
=====
Memory usage statistics:
Total number of allocated memory blocks: 37 (0)
Total number of memory reallocations: 3
Total length of memory used to load the netlist: 256 kB (0)
Number of system memory blocks (total/long): 1/0
Netlist loaded in: 4 seconds
=====
[Navigation Buttons]
Ready Ln 1, Col 1 NUM
```

3.3. Realizar la simulación.


Es necesario realizar tres pasos básicos para simular un diseño:

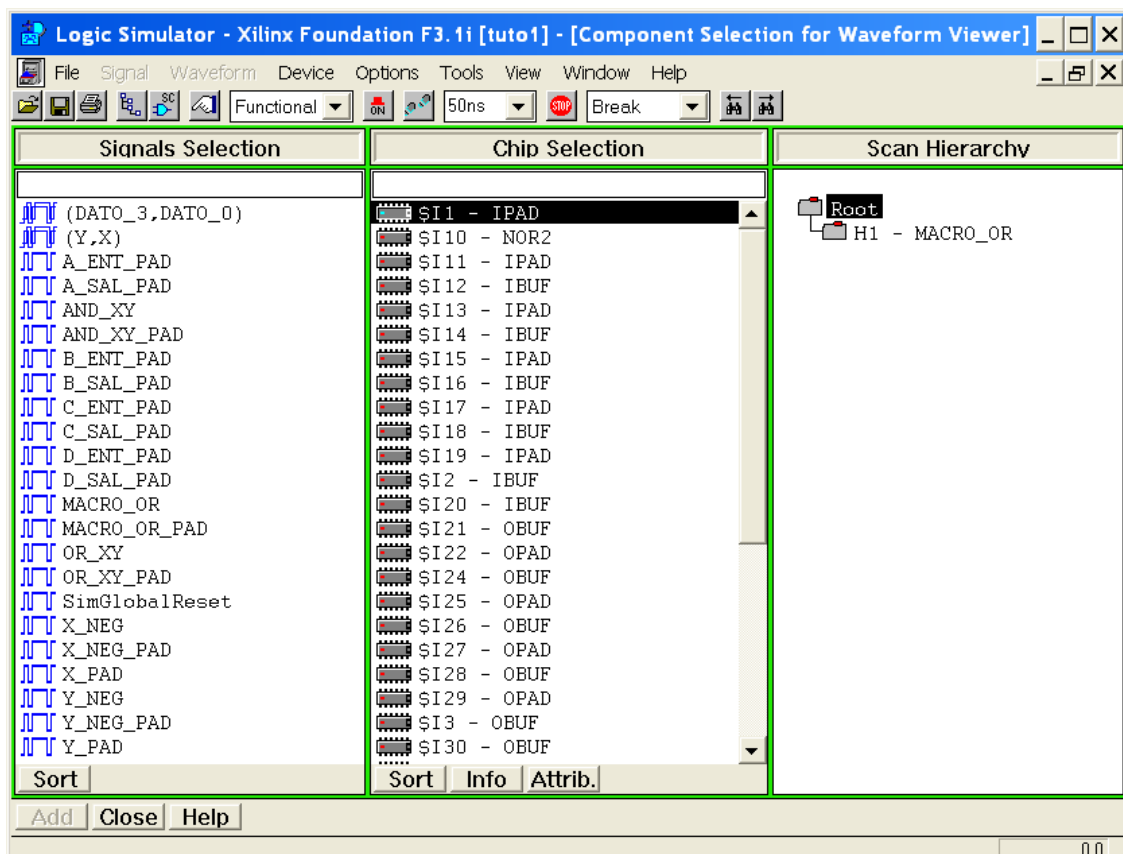
1. Añadir todas las señales (entradas y salidas) que queremos ver en el simulador.
2. Añadir estímulos a todas las señales de entrada (no puede haber ninguna señal de entrada sin estímulo, es decir, sin un valor lógico definido como "0" ó "1").
3. Ejecutar la simulación avanzando en el tiempo para ver las formas de onda de salida y así comprobar el correcto funcionamiento del circuito.

3.3.1. Añadir señales al simulador.

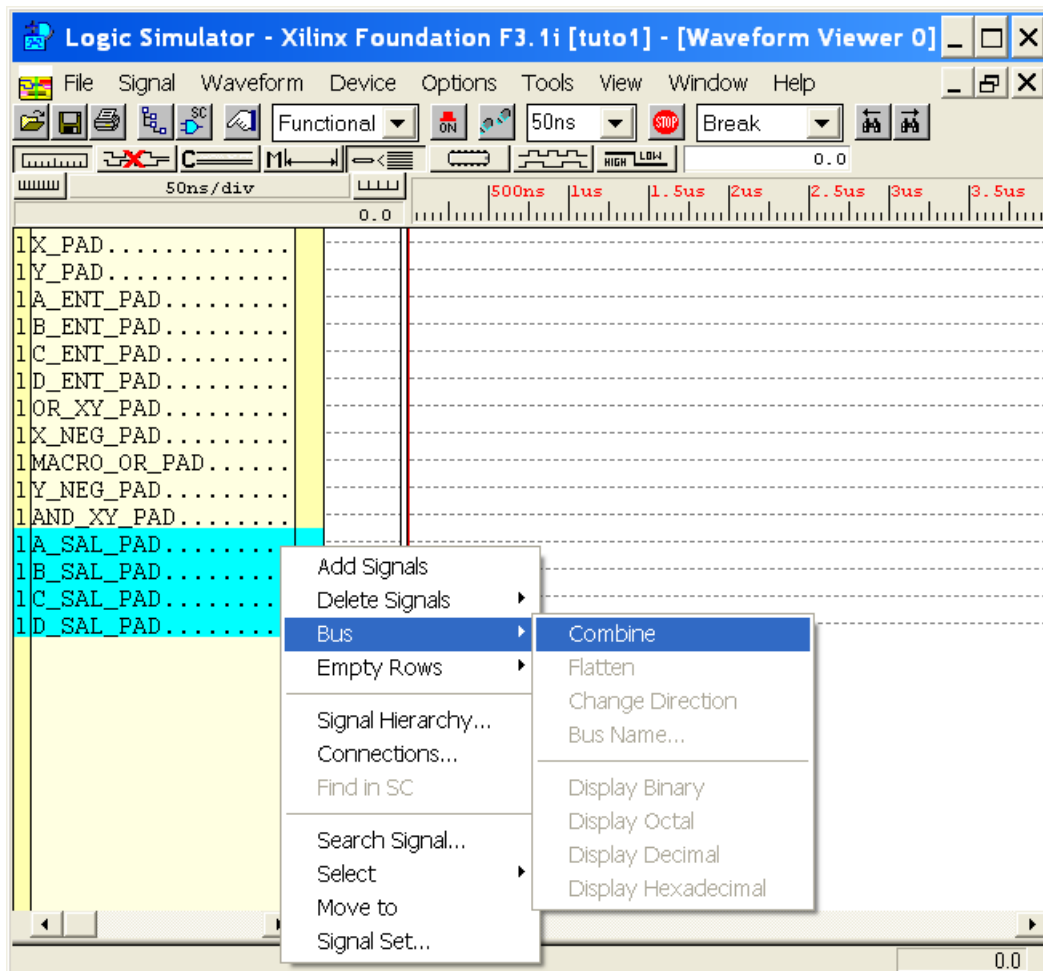
Para ver las señales durante la simulación se debe añadir primero dicha señal al visor de formas de onda del simulador, para posteriormente avanzar el tiempo en dicha simulación y ver cómo cambian los valores lógicos de las mismas.

Para añadir estas ondas podemos hacerlo desde la propia ventana del simulador:

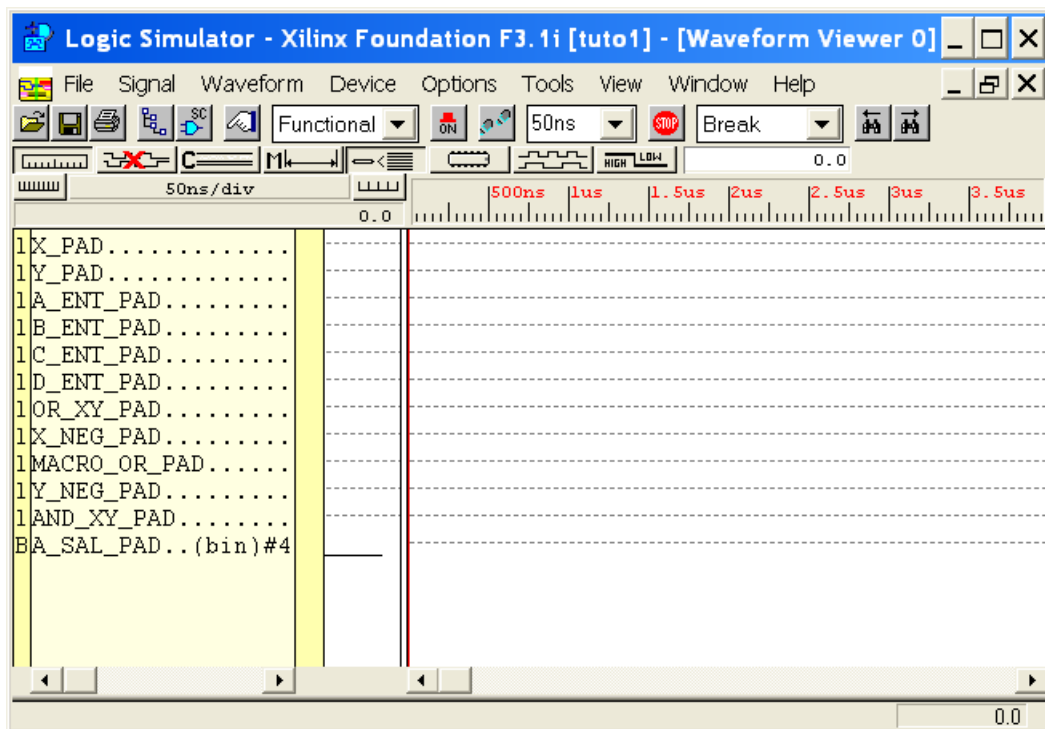
1. Pulsar el icono "Select Component"  en la barra de herramientas del simulador o a través del menú "Signal → Add Signals". Se abrirá una ventana de selección de componentes dividida en tres paneles. El de más a la izquierda, "Signals Selection", muestra una lista de todas las señales disponibles. El central, "Chip Selection", muestra todos los componentes presentes en la hoja del esquemático. El de la derecha, "Scan Hierarchy", muestra la jerarquía de macros presentes en el diseño por la cual nos podemos mover y "entrar" a ver las señales internas de cada una de dichas macros.



2. En el panel “*Signals Selection*” haremos doble “click” sobre las señales que deseemos añadir al visor de formas de onda, o pulsar una sola vez sobre varias de ellas y entonces presionar el botón “*Add*”. En nuestro ejemplo del tutorial añadiremos todas las señales de entrada, es decir, X_PAD, Y_PAD, A_ENT_PAD, B_ENT_PAD, C_ENT_PAD y D_ENT_PAD, así como las señales de salida OR_XY_PAD, MACRO_OR_PAD, X_NEG_PAD, AND_XY_PAD, Y_NEG_PAD, A_SAL_PAD, B_SAL_PAD, C_SAL_PAD y D_SAL_PAD.
3. Cerramos la ventana de selección de componentes pulsando el botón “*Close*”.
4. A continuación vamos a crear un bus para las señales de salida A_SAL_PAD, B_SAL_PAD, C_SAL_PAD y D_SAL_PAD, de manera que su valor conjunto lo veamos codificado en binario. Para ello hacemos “click” sobre la primera de ellas, y pulsando la tecla de “Mayúsculas” hacemos “click” sobre la última del grupo, seleccionándose todo el rango de señales intermedias (como alternativa, podemos hacer “click” una por una sobre todas ellas, con la tecla “Ctrl” pulsada). Con las cuatro salidas seleccionadas, pulsamos con el botón derecho del ratón sobre ellas y seleccionamos “*Bus → Combine*”:



5. A continuación cambiamos la base de dicho bus para ponerla en binario, para lo cual hacemos “click” sobre el mismo y luego pulsamos con el botón derecho del ratón y seleccionamos “*Bus → Display Binary*” (podemos seleccionar mostrarlo en binario, octal, decimal o hexadecimal).
6. Todas las señales que hemos añadido están ahora en el visor de formas de onda. Podemos ordenar la presentación de dichas señales en el visor arrastrándolas hacia la posición deseada.




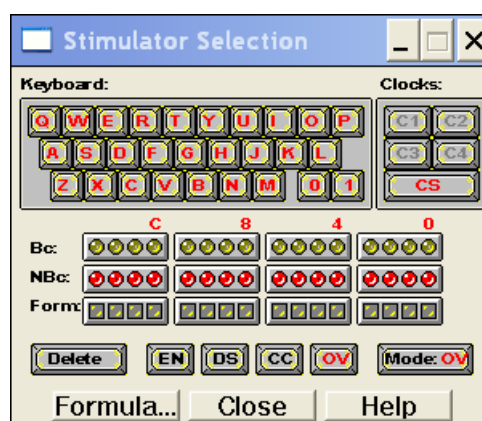
Si necesitamos borrar una señal en un momento dado se puede hacer fácilmente seleccionándola (con un “click” sobre la misma) y pulsando el botón derecho del ratón, a través del menú “Delete Signals → Selected”.

3.3.2. Añadir estímulos a todas las señales de entrada.

Las señales de entrada deben tener unos valores lógicos definidos para que la simulación funcione correctamente, por lo que será necesario estimular cada una de dichas entradas.

Hay tres formas de definir estímulos con el simulador del *Xilinx Foundation*. En este tutorial se utilizará para los estímulos solamente dos de las formas: el teclado del PC o un contador binario interno. La tercera forma sería creando un “script” de simulación, para lo cual hay que conocer el lenguaje propio del simulador, lo cual no entra dentro del ámbito de este tutorial.

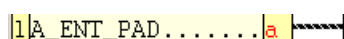
Abrimos la ventana del “Stimulator Selection” pulsando en la barra de herramientas sobre el icono  o seleccionando el menú “Signal → Add Stimulators”:



Estimular las entradas con el teclado del PC.

Una forma de dar estímulos a la señales del ejemplo del tutorial puede ser mediante el teclado del PC. Podemos relacionar las señales de nuestro diseño con las diferentes teclas del PC en la ventana *"Stimulator Selection"*, de manera que después de asignar una tecla como estímulo, los valores de la señal conmutarán entre "0" y "1" cuando presionemos la correspondiente tecla en nuestro PC.

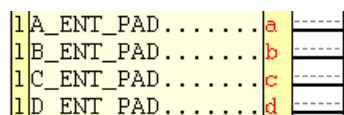
1. Hacemos "click" sobre la tecla "A" del teclado en el selector de estimulador, y sin dejar de pulsar el botón del ratón la arrastramos hasta la señal "A_ENT_PAD" en el visor de forma de onda.
2. Ahora deberíamos ver una "a" al lado de la señal "A_ENT_PAD" en el visor de formas de onda, lo cual indica que éste es el estímulo asignado:



3. Pulsamos la tecla "a" en el teclado de nuestro PC varias veces para ver que el estado del estímulo cambia en el visor de formas de onda:

A_ENT_PAD = "0" lógico	A_ENT_PAD = "1" lógico
	

4. De la misma forma, asignamos la "b", "c" y "d" del teclado de estímulos a las señales "B_ENT_PAD", "C_ENT_PAD" Y "D_ENT_PAD" respectivamente, y ponemos todas las señales a un valor lógico definido, por ejemplo, a "0" lógico:



5. Cerramos la ventana *"Stimulator Selection"* pulsando sobre su botón *"Close"*.

Estimular las entradas con un contador binario interno.

En muchos diseños resulta muy útil poder probar todas las combinaciones posibles sobre un conjunto de señales de entrada, por ejemplo, para comprobar que una función combinatorial está funcionando correctamente. En nuestro caso del tutorial tenemos un grupo de dos señales que responderían a dicha característica, X_PAD e Y_PAD. Para comprobar que los circuitos combinatoriales a los que están conectadas proporcionan los resultados correctos sería necesario que tomaran las 2^2 combinaciones posibles que se podrían hacer, es decir, 00, 01, 10 y 11.

El simulador de *Xilinx Foundation* nos proporciona un contador binario interno de hasta 16 bits que podemos utilizar para estimular las señales que deseemos, el cual se encuentra en la ventana *"Stimulator Selection"* en la fila nombrada como "Bc":

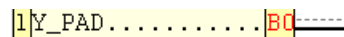


Dicho contador nos proporcionará un bit menos significativo (el B0, situado más a la derecha) que irá cambiando de "0" a "1" lógico constantemente durante la simulación, a cierta frecuencia (la definiremos más adelante). El siguiente bit más significativo (el B1) cambiará también de "0" a "1" lógico pero a la mitad de frecuencia que el anterior, y lo mismo pasará con el B2 (tendrá mitad de frecuencia que el B1), etc. De

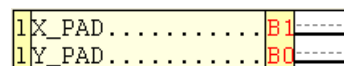
esta forma, asignando cada bit del contador a una señal de entrada, podremos probar todas las combinaciones posibles de unos y ceros en las señales de entrada.

Para estimular nuestras señales X_PAD e Y_PAD en nuestro caso, escogeremos los 2 bits menos significativos de dicho contador, que se corresponderían con los bits B1 y B0 respectivamente (los de más a la derecha). La forma de proceder sería la siguiente:

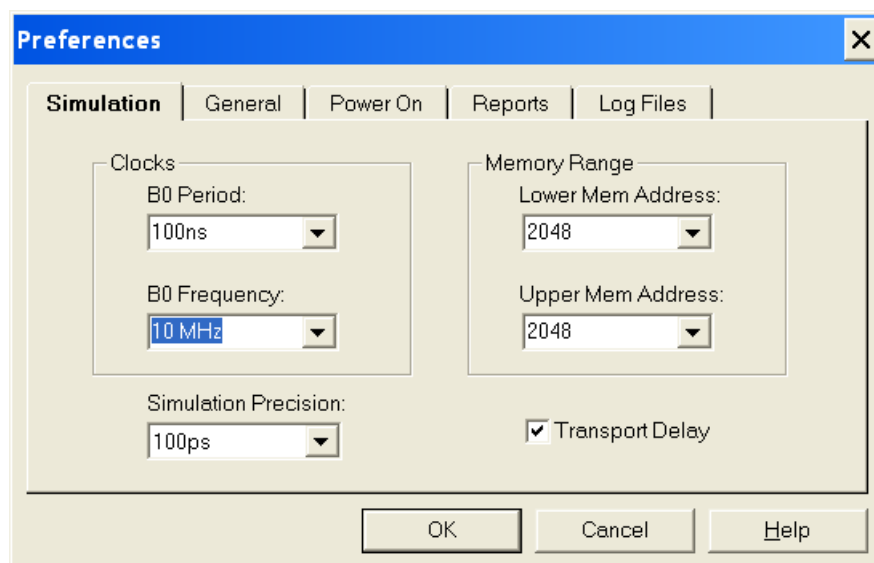
1. Para estimular la entrada Y_PAD del sistema asignándole el bit B0 del contador, pinchamos sobre su correspondiente círculo amarillo y lo arrastramos sobre la señal:



2. Repetimos el proceso con la entrada X_PAD, pero ahora asignándole el bit B1 del contador, pinchamos sobre su correspondiente círculo amarillo y lo arrastramos sobre la señal:

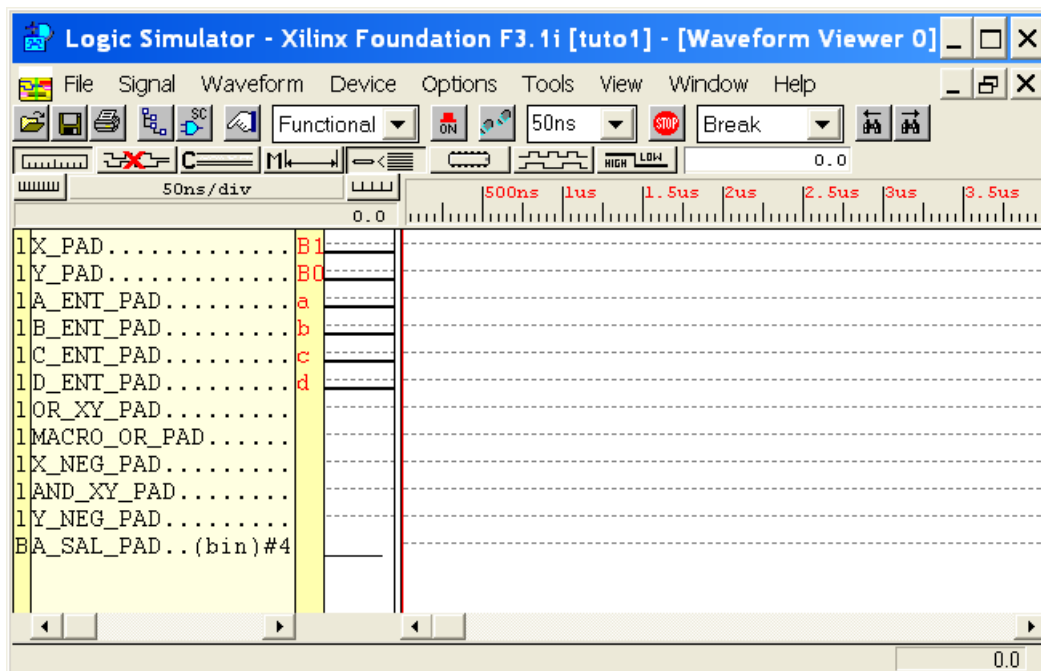



3. Establecemos la frecuencia a la que cambiará el bit B0 del contador binario interno (es importante hacerlo). Seleccionamos “Options → Preferences” desde la ventana del simulador, y se abrirá una ventana. En la pestaña llamada “Simulation” podemos seleccionar la frecuencia del contador B0, poniendo el valor de “B0 Frequency” a 10MHz, tras lo cual pulsaremos “OK”:

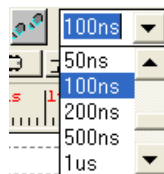


3.3.3. Ejecutar la simulación.

Con todo lo realizado hasta ahora, el simulador debería tener este aspecto, con todas las señales de entrada y salida añadidas, y las de entrada estimuladas bien mediante el teclado, bien mediante el contador interno del simulador:

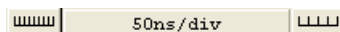


Para hacer que avance el tiempo de simulación tendremos que utilizar el botón “*Step*”  en la barra de herramientas del simulador. Además, será necesario definir el tamaño del paso de simulación (precisamente este “*Step*”) usando las opciones de la ventana desplegable que se encuentra al lado de dicho botón, como se muestra en la figura que sigue:

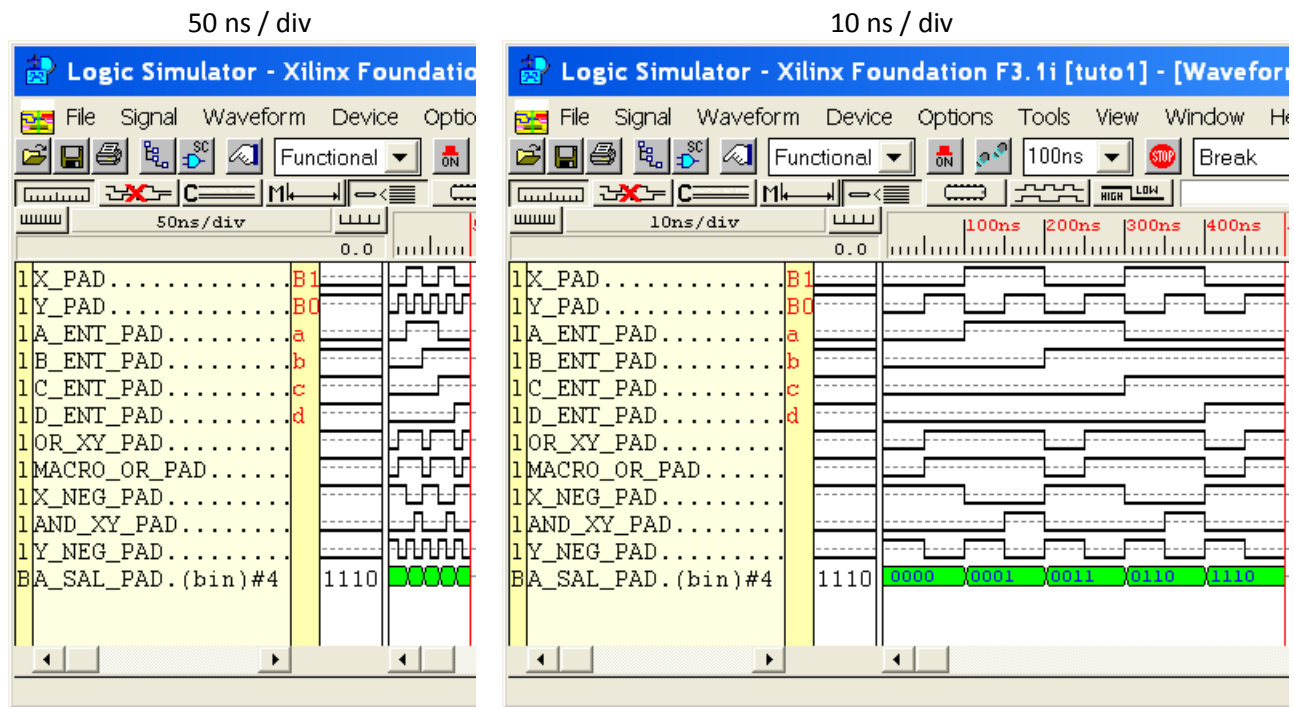


Por tanto, tras seleccionar un paso de simulación adecuado (por ejemplo, 100 ns), iremos pulsando en repetidas ocasiones el botón “*Step*” y cambiando de vez en cuando con el teclado del PC los valores de las señales de entrada, de manera que obtendremos un cronograma con todas las formas de onda de entrada y de salida del circuito.

Como inicialmente se nos quedarán muy pequeñas estas formas de onda, tendremos que cambiar el “*Zoom*”. Para ello utilizaremos el botón “*Zoom In*” (a la izquierda en el dibujo) y el de “*Zoom Out*” (a la derecha) representados en la siguiente barra, donde también se ve la escala de tiempo que representa cada marca horizontal en pantalla:



Pulsando varias veces sobre el botón “*Step*”, cambiando algunas señales con el teclado, y modificando el “*Zoom*” de pantalla podemos ver el efecto sobre las mismas ondas, así como comprobar si el circuito funciona adecuadamente:



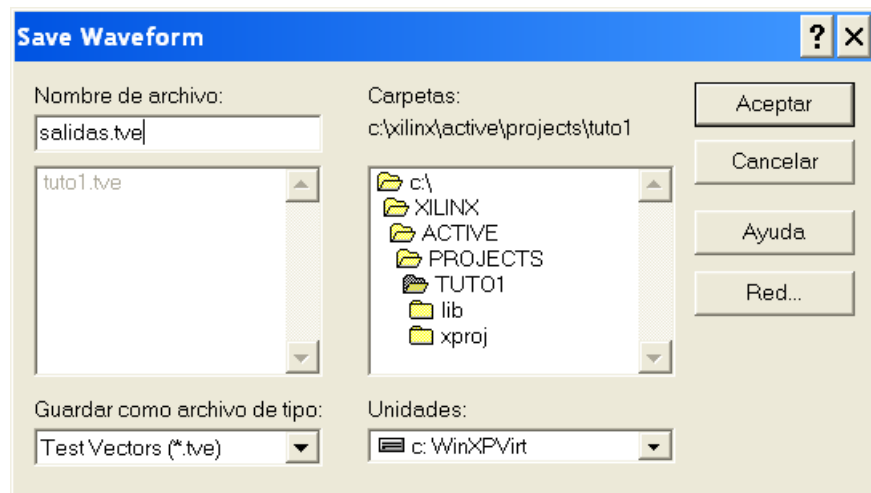
Podemos comprobar como el circuito funciona adecuadamente, ya que las salidas cumplen su cometido:

- OR_XY_PAD: Se obtiene la función OR sobre las señales X_PAD e Y_PAD
- MACRO_OR_PAD: Funciona igual que la señal de salida anterior
- X_NEG_PAD e Y_NEG_PAD: Se obtienen las señales negadas de X_PAD e Y_PAD, respectivamente
- AND_XY_PAD: Crea la función AND entre las señales de entrada X_PAD e Y_PAD
- A_SAL_PAD (bus): Codifica en un bus binario los cambios producidos en las entradas estimuladas por el teclado del PC con "a", "b", "c" y "d", estando la "a" situada en el bit menos significativo del bus (a su derecha).

Nota: En ocasiones, los buses aparecen con los datos en sentido inverso respecto a cómo nos gustaría verlos, es decir, aparecen intercambiados los bits de menos a más significativos. Para poder corregir este común comportamiento del software hacemos un "click" sobre dicho bus, pulsamos con el botón derecho del ratón y escogemos el menú "Bus → Change Direction", de manera que le dará toda la vuelta al bus y los datos aparecerán en el orden deseado.

Si es necesario reiniciar la simulación porque se hayan producido cambios en los estímulos, por ejemplo, pulsaremos sobre la opción del menú "Waveforms → Delete → All Waveforms with Power On".

Si queremos salvar las formas de onda que estamos viendo para un posterior análisis, o bien para adjuntarlo como documentación a una práctica, pulsaremos sobre "File → Save Waveform", y le daremos al fichero cualquier nombre distinto al del proyecto, porque de lo contrario se sobrescribirá cuando se vuelva a realizar otra simulación. El fichero tendrá extensión .TVE:



Parte 4. Implementación del diseño

A la hora de implementar un diseño que ya tenemos hecho y simulado correctamente hay que tener en cuenta el dispositivo final sobre el que vamos a programarlo. Igualmente, debemos conocer las características de la placa que alberga nuestro dispositivo CPLD porque condicionará algunos aspectos del diseño que tenemos.

Es por esto que seguiremos unos pasos a continuación para adaptar nuestro diseño de manera conveniente para que nos funcione en la placa de la CPLD que tenemos en el laboratorio:

1. Familiarizarse con la placa de la CPLD que vamos a utilizar, así como la de periféricos.
2. Realizar los cambios oportunos en el esquemático para que se adapte perfectamente a dicha placa y modelo de CPLD.
3. Modificar el esquemático para introducir las macros adecuadas que nos permitan controlar los módulos existentes en la placa de periféricos (cuando sea necesario).
4. Implementar el diseño a través del software *Xilinx Foundation 3.1i*.
5. Simular de nuevo el diseño pero ahora con tiempos de retraso típicos, lo cual nos dará unos resultados muy fiables ya que se usarán los mismos tiempos de retardo que introduce la CPLD física. A este paso se le denomina verificación.
6. Una vez verificado el diseño, programaremos la placa de la CPLD para comprobar el funcionamiento real del circuito.

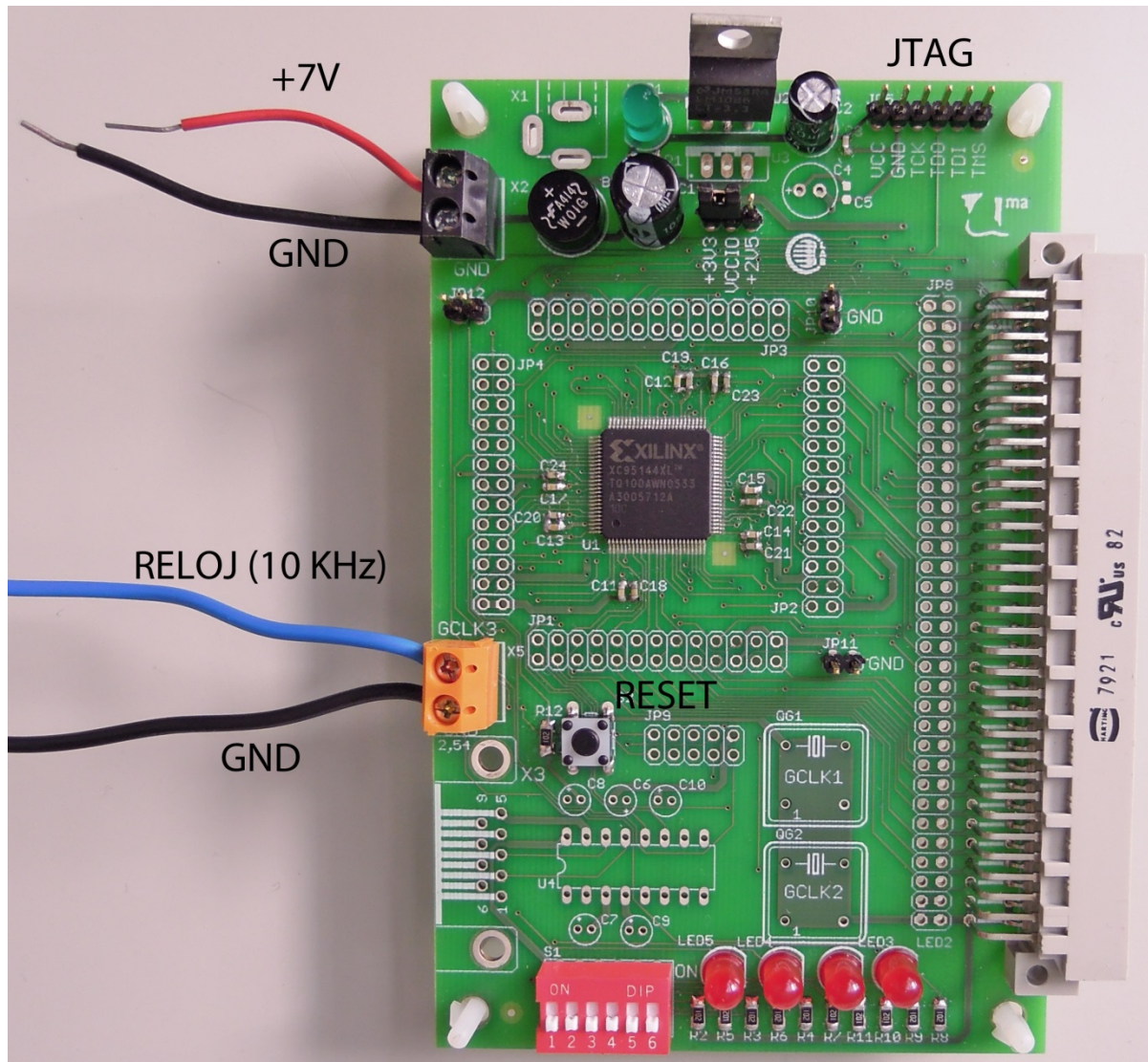
4.1. Descripción de la placa de la CPLD y de periféricos.

A continuación describiremos algunas características principales de la placa de la CPLD que utilizaremos en las prácticas:

- Tiene un modelo de CPLD que es el 95144XLTTQ100, que es el chip central que se aprecia en la imagen.
- En la parte superior de la placa se encuentra el conector JTAG, que nos permitirá comunicarnos con el PC para programar la placa, utilizando el cable adecuado.
- A la izquierda, arriba, vemos un conector negro con 2 cables, uno rojo y otro negro. A través de dichos cables alimentaremos esta placa, con 7V en el cable rojo y tierra (GND) en el negro.
- A la izquierda, a la mediación de la placa, vemos un conector parecido al anterior pero de color naranja, que poseerá 2 cables, uno negro (tierra, GND) y otro habitualmente azul (aunque puede haberse sustituido el cable por alguno de otro color, siempre que no sea negro). A través del mismo introduciremos, con ayuda del generador de funciones, una señal de onda cuadrada de 10 KHz de frecuencia y que oscile entre 0V y 5V (no se debe exceder nunca de esos 5V, porque podremos estropear la placa). Esta señal de reloj nos servirá para utilizarla dentro de la CPLD en los módulos que así la requieran.
- En la parte inferior vemos que la placa posee 6 interruptores, así como 4 diodos luminosos o leds. La placa también posee un pulsador que se puede usar como RESET del circuito.
- A la derecha existe un conector de gran tamaño que nos permitirá unirla a la placa de periféricos y así poder utilizar los diferentes pulsadores, leds, displays de 7 segmentos, etc., que ésta posee.

NOTA: Es muy importante que no nos equivoquemos de conector para alimentar la placa, ya que si lo hacemos por el de color naranja en lugar del de color negro podemos hacer que la placa deje de funcionar definitivamente.

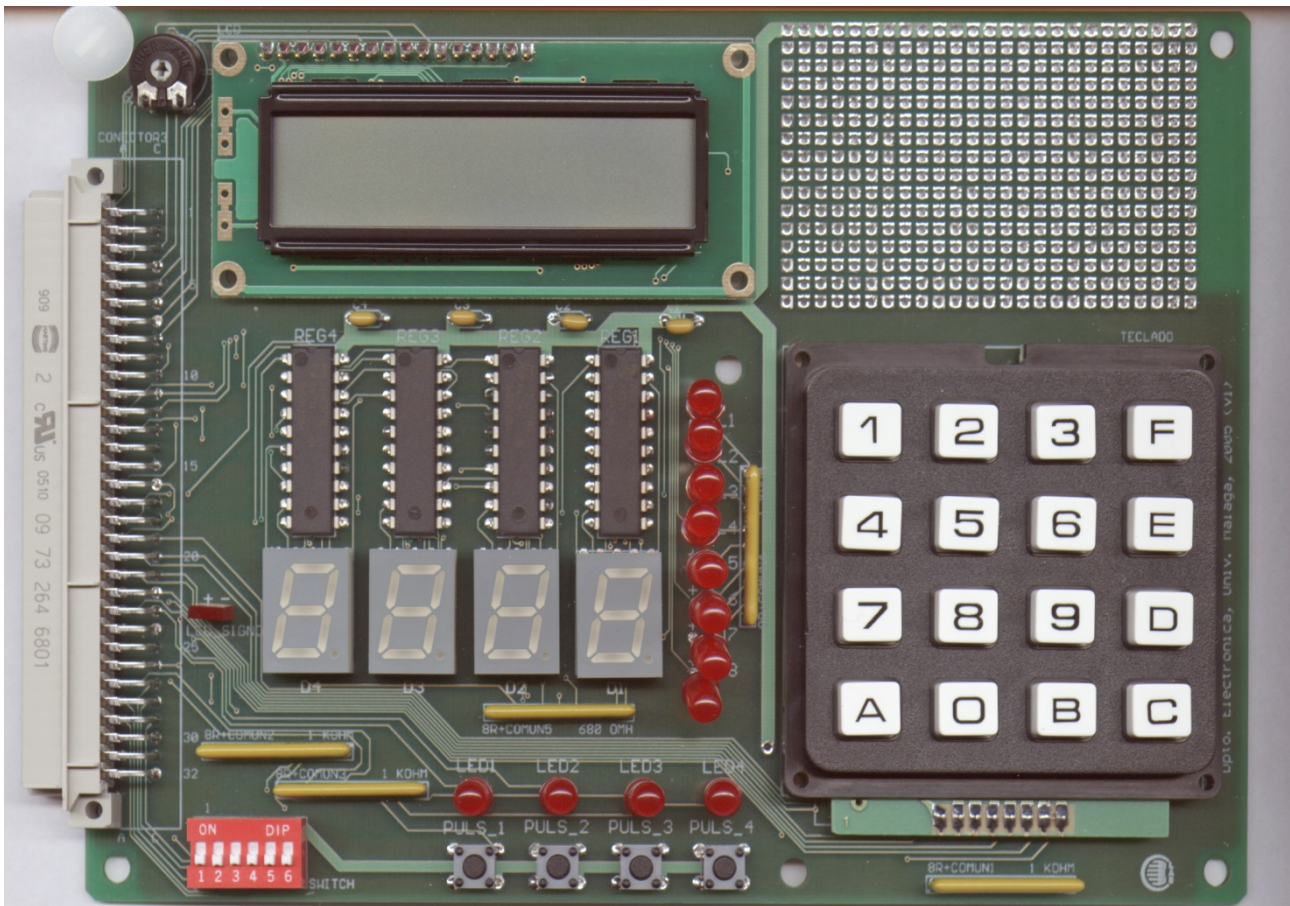
En la siguiente imagen vemos la placa de la CPLD:



La placa de periféricos depende del control de la principal, es decir, de la que contiene la CPLD. El principal uso que se le dará en las prácticas será el de utilizar los 4 displays de 7 segmentos que posee (más un segmento adicional para el punto decimal), denominados D4, D3, D2 y D1, siendo este último las unidades (el dígito menos significativo, situado a la derecha). Además, posee otra serie de pulsadores, interruptores, leds, un teclado hexadecimal y se le puede acoplar también un display LCD de caracteres.

Los displays de 7 segmentos de esta placa tienen la característica de que comparten el bus de datos entre todos ellos, ya que si usáramos 8 líneas de datos independientes de cada uno necesitaríamos 32 líneas separadas para llevar los números a dichos displays, pero no disponemos de tantas patillas libres en la CPLD. Es por esto por lo que el bus de datos de 8 bits es compartido entre todos ellos, y lo que se hace es establecer un mecanismo de control para que el dato llegue correctamente a cada display, cargándose en 4 registros externos existentes en la placa. Este control se hace en un módulo que se proporciona ya diseñado para las prácticas de la asignatura, y que veremos en el siguiente apartado.

En la siguiente imagen vemos la placa de periféricos:



Como se puede apreciar, el único conector que posee esta placa es el que le sirve para unirse a la de la CPLD. A través de él recibirá y enviará las señales necesarias, y además tomará la alimentación para sus dispositivos.

4.2. Cambios a realizar en el esquemático para adaptarse a la placa de la CPLD.

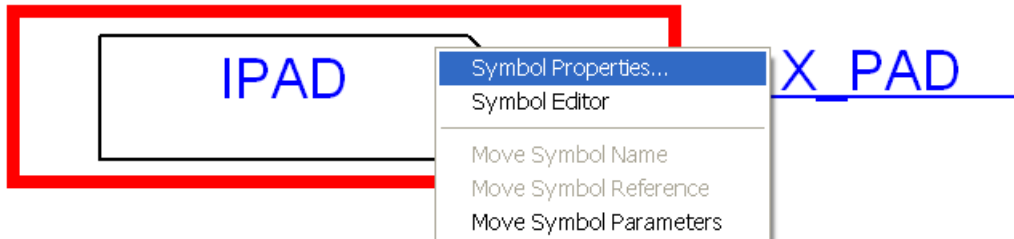
Los cambios a realizar en el esquemático estarán referidos sobre todo a los PADS de entrada y salida, ya que son éstos los que comunican la CPLD con el exterior, así que dependiendo por qué línea deseamos que entre o salga cierta señal pues así tendremos que configurarlo, ya que a cada uno de ellos habrá que ponerle una localización física de la CPLD. Para la placa sobre la que vamos a trabajar tenemos las siguientes localizaciones disponibles:

- Interruptores de entrada, numerados del 1 al 6 (de izquierda a derecha): Están localizados en los pines 90, 91, 92, 93, 94 y 95 de la CPLD 95144XLTQ100.
- Diodos led de la placa, nombrados como LED5, LED4, LED3 y LED2 (de izquierda a derecha): Están localizados en los pines 89, 87, 86 y 85 de la CPLD 95144XLTQ100.
- El pulsador de RESET, si es necesario usarlo, lo situaremos en el pin 99.

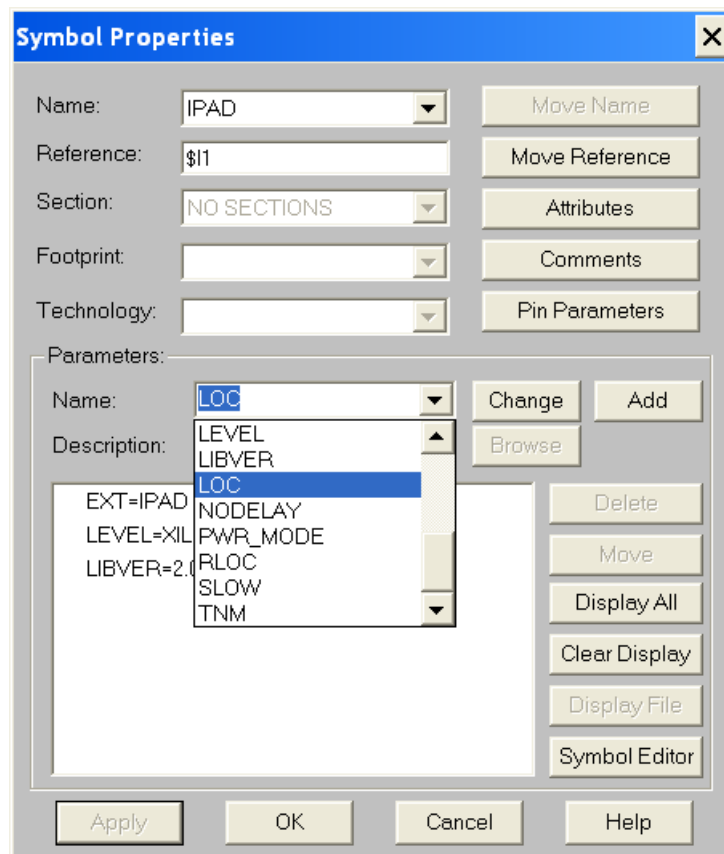
Para el ejemplo de este tutorial podemos establecer que las señales de entrada A_ENT_PAD, B_ENT_PAD, C_ENT_PAD, D_ENT_PAD, X_PAD y Y_PAD vendrán por los microinterruptores 1, 2, 3, 4, 5 y 6 respectivamente, lo cual corresponde a las localizaciones 90, 91, 92, 93, 94 y 95 como hemos visto anteriormente. Las salidas OR_XY_PAD, AND_XY_PAD, X_NEG_PAD e Y_NEG_PAD las llevaremos a unos diodos luminosos, concretamente a LED5, LED4, LED3 Y LED2 (pines 89, 87, 86 y 85). El resto de señales, como no vamos a utilizarlas, podemos dejarlas sin asignar a ningún pin específico, el software se encargará de ponerlas en los pines que él elija.

Para asignar un pin concreto a cada IPAD u OPAD del diseño, seguiremos los siguientes pasos:

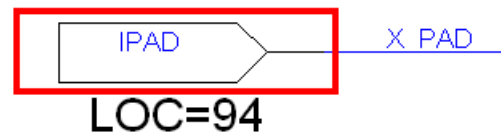
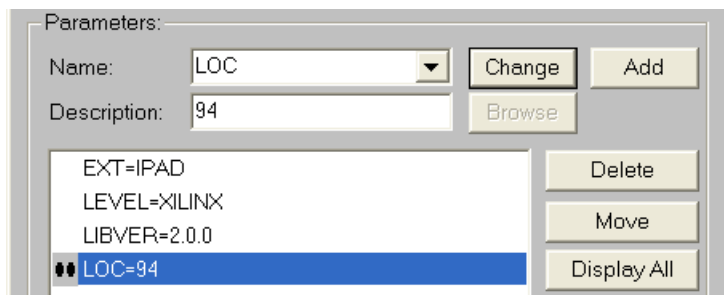
1. Seleccionamos un IPAD u OPAD, y hacemos doble “click” sobre él para obtener el menú de propiedades, o bien una vez seleccionado el PAD pulsamos con el botón derecho del ratón y escogemos la opción “*Symbol Properties*”:



2. En la ventana que aparece, dentro de la sección “*Parameters*”, entramos en el menú desplegable “*Name*” y seleccionamos la propiedad “*LOC*”:



3. En el recuadro “*Description*” que hay justo debajo tecleamos el número de pin donde queremos situar dicho PAD, pulsamos el botón “*Add*”, y posteriormente el “*OK*”. Para el caso del ejemplo, la señal X_PAD va al microinterruptor 5, que se corresponde con el pin 94 de la CPLD, quedando en el diseño como se muestra:



En el ejemplo del tutorial repetiríamos este proceso para los IPAD y OPAD del diseño que se han indicado anteriormente, poniendo las localizaciones adecuadas.

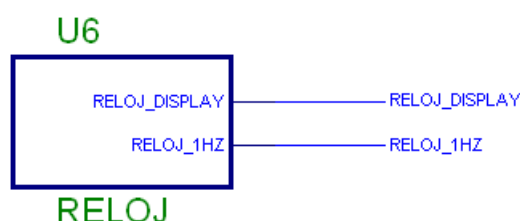
4.3. Modificación del esquemático para introducir macros de control de la placa de periféricos.

Hemos visto como la placa de periféricos consta de diversos módulos que podemos controlar desde la CPLD. Para el uso de estos módulos se proporcionarán una serie de macros prediseñadas que simplemente con instanciarlas en el diseño podremos hacer uso de estos módulos.

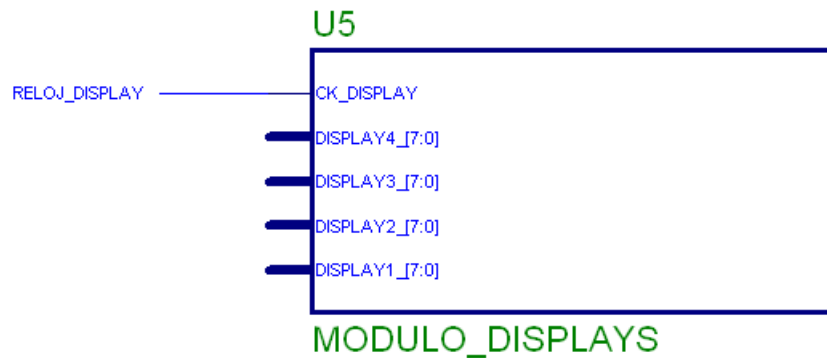
Uno de los bloques que usaremos serán los displays de 7 segmentos. Como vimos antes, esta placa posee 4 displays de 7 segmentos (más un segmento extra para el punto decimal), llamados D4, D3, D2 y D1 (de más a menos significativos). Para el uso de estos bloques se proporcionará para las prácticas un diseño de *Xilinx Foundation* llamado "MODULOS.ZIP" que poseerá una serie de macros que se deberán copiar al diseño en el que se esté trabajando para así poder instanciarlas.

Por tanto, el proceso para usar estas macros será el siguiente:

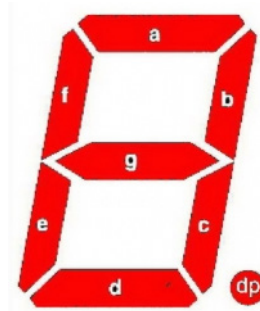
1. Descargarse el proyecto "MODULOS.ZIP" del Campus Virtual, y restaurarlo desde el "Project Manager" con la opción "File → Restore Project".
2. Copiar todas las macros que contiene al proyecto de destino (consultar apartado 2.8).
3. Abrir el proyecto de destino y editar el esquemático en el que estuviéramos trabajando.
4. Añadir las siguientes macros: RELOJ y MODULO_DISPLAYS, ambas necesarias para que funcionen los displays de 7 segmentos de la placa de periféricos.
5. La macro RELOJ tendrá dos salidas que serán la señal de reloj necesaria para que funcione correctamente la macro MODULO_DISPLAYS (RELOJ_DISPLAY) y la que necesitaremos para la práctica de la máquina de estados (RELOJ_1HZ). Para su correcto funcionamiento será imprescindible utilizar el generador de funciones para introducir una señal de reloj a través del conector naranja de la placa de la CPLD, con las características que ya se comentaron en el apartado 4.1, es decir, señal de onda cuadrada, de 10 KHz de frecuencia, y con una amplitud que oscile entre un valor mínimo de 0V y máximo de 5V:



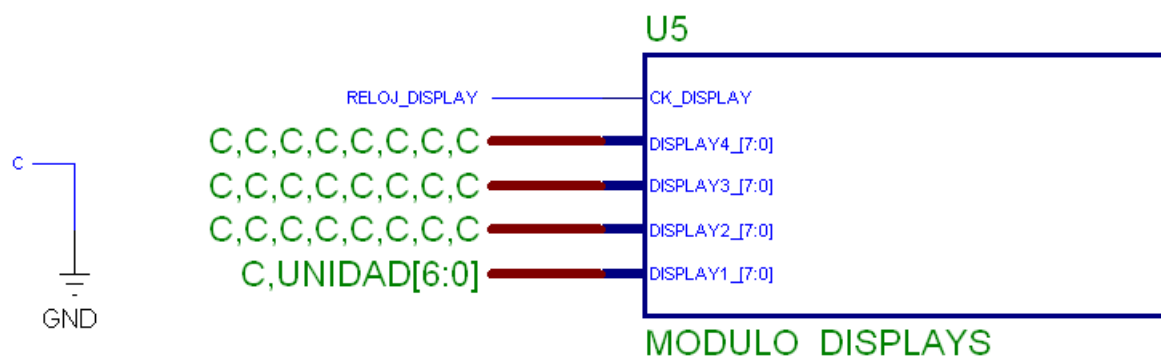
6. La macro MODULO_DISPLAYS será la que reciba la información que se enviará a los displays de 7 segmentos de la placa de periféricos. Será necesario introducirle la señal RELOJ_DISPLAY que proviene de la macro RELOJ anteriormente descrita, junto con otros 4 buses que codifiquen la información de 7 segmentos (más el punto decimal) para cada uno de los dígitos que se pueden representar en dicha placa (D4, D3, D2 y D1):



7. El formato de los buses es el siguiente: Los bits desde el 0 hasta el 6, en ese orden, se corresponden con los segmentos “a” hasta el “g” de la representación en 7 segmentos mostrada a continuación. El bit 7 de cada bus sería el punto decimal de dicho dígito (“dp”):

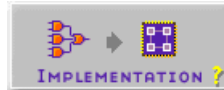


8. Finalmente, si tenemos un bus llamado “UNIDAD[6:0]”, por ejemplo, y lo queremos situar en el display D1 (el de más a la derecha), tendríamos que generar un bus de 8 bits (ojo, 7 segmentos más el punto decimal) de tal manera que combinaran estos 7 bits del bus “UNIDAD” junto con un cero lógico fijo para el punto decimal (para que no se encendiera nunca). Este cero lógico lo podemos obtener a partir del símbolo “GND” de la librería, al cual le podemos poner un cable con una etiqueta que se llame “C”, por ponerle un nombre corto, y lo utilizaríamos para todos los sitios en los que sea necesario. Así, para crear el bus de las unidades, combinaríamos en un bus complejo este cero lógico “C” junto con el bus “UNIDAD[6:0]”, y para el resto de displays (si no vamos a poner nada y los pretendemos “apagar”) pondríamos todos los bits a cero lógico con otros tantos buses complejos:

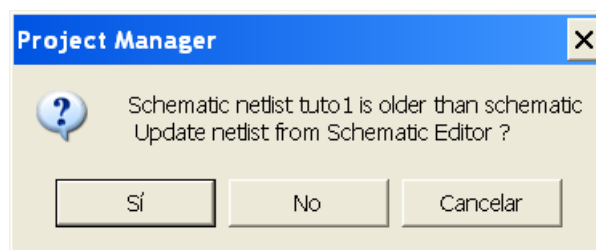


4.4. Implementación del diseño.

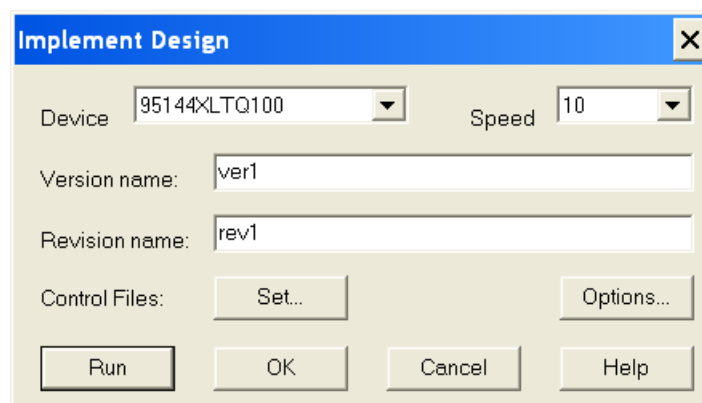
Cuando ya tenemos el esquemático preparado con las localizaciones adecuadas para los PAD de entrada y salida, y con los módulos de control de la placa de periféricos situados en el diseño, ya estamos en condiciones de poder implementar nuestro proyecto. Para ellos iremos al “*Project Manager*” y pulsaremos sobre el botón “*Implementation*”:



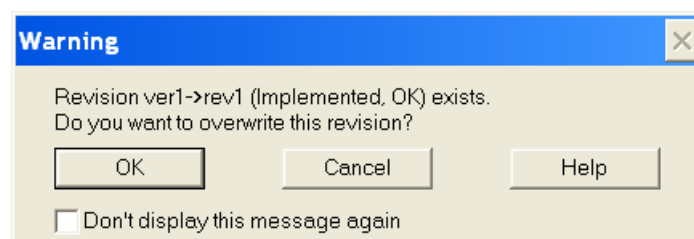
Lo más probable es que al pulsarlo nos aparezca una ventana indicándonos que la netlist del diseño es más antigua que el esquemático, y nos pregunta si queremos actualizarla. Pincharemos en “*Sí*”:



Tras esto, la primera vez que implementamos nos aparecerá una pantalla para confirmar los datos del dispositivo de destino, opciones de implementación, optimización, etc, las cuales habitualmente no modificaremos. Comprobaremos, eso sí, que la CPLD de destino es la correcta (95144XLTQ100, con grado de velocidad 10), y pulsaremos “*Run*”:

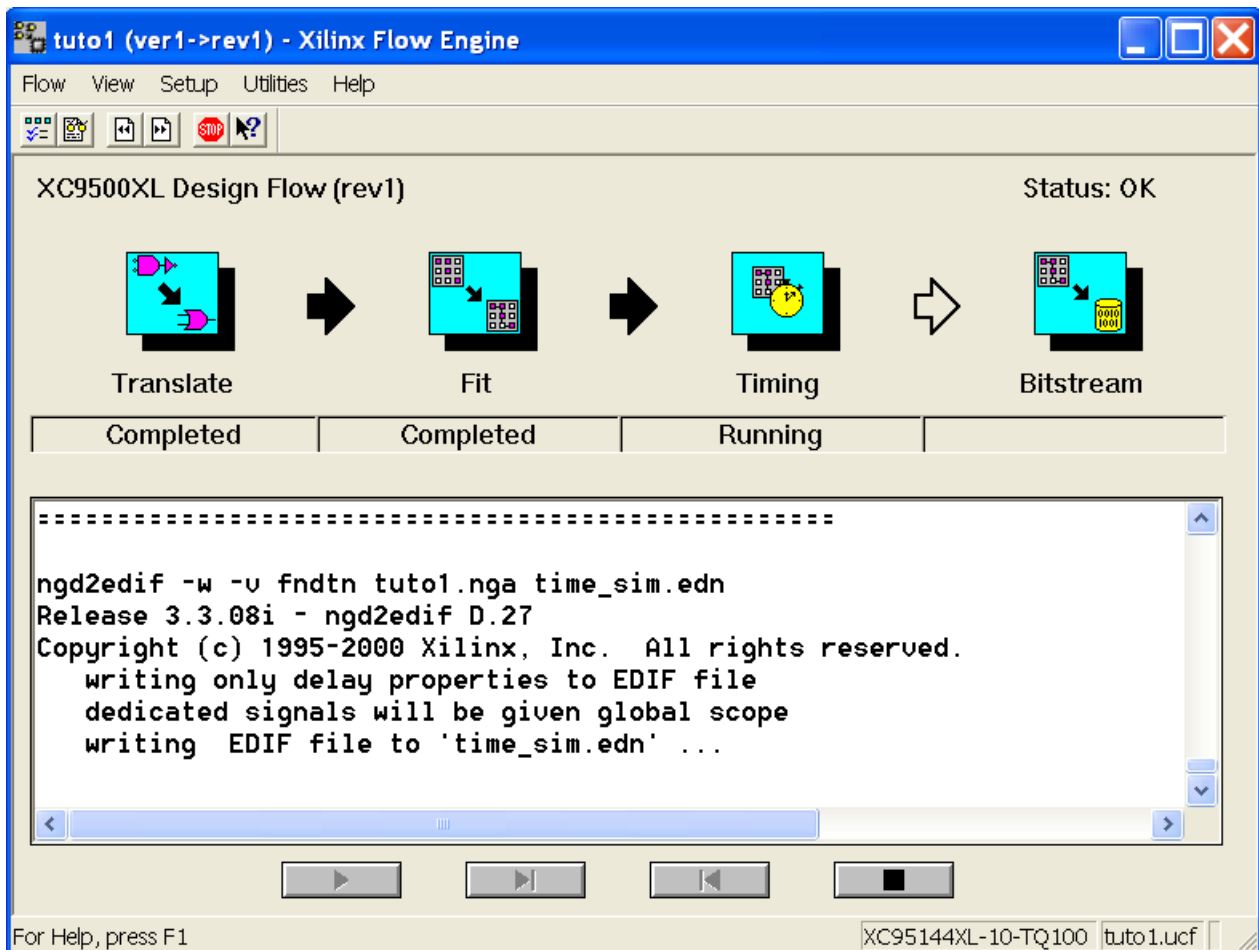


Si ya hemos hecho una implementación previa, tras lo cual realizamos unos cambios en el esquemático y volvemos a implementar el diseño, la ventana que nos aparecerá será como ésta, en la que simplemente pulsaremos “*OK*” para sobrescribir la implementación anterior:

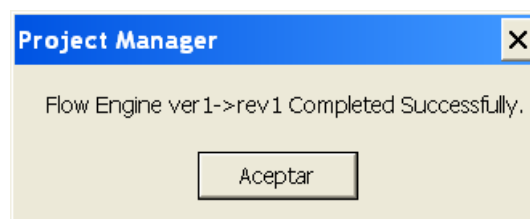


NOTA: En ocasiones nos da un error al intentar sobrescribir dicha implementación. Hay que verificar que no está abierta ninguna otra ventana del *Xilinx Foundation* a la hora de implementar, pero si aun así continúa dando el error pincharemos en el menú "*Project → Clear Implementation Data*" para crear una implementación completamente nueva.

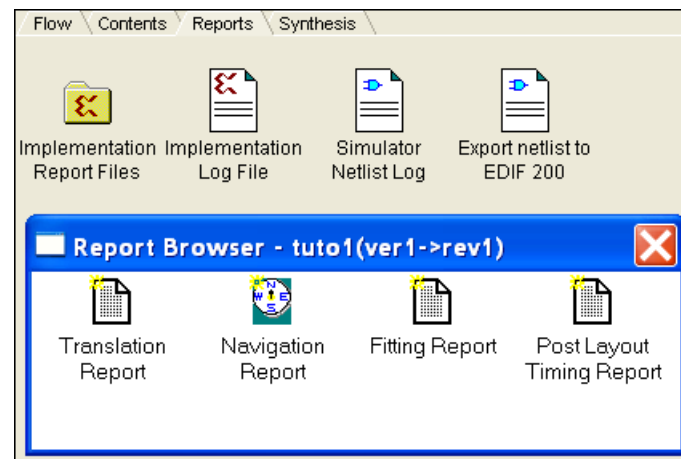
A continuación debe aparecer el "*Xilinx Flow Engine*" o motor de implementación, una ventana como la que sigue en donde se muestran los procesos por los que automáticamente va pasando el diseño hasta que se genera el fichero para programar la CPLD:



La implementación debería terminar satisfactoriamente con el siguiente mensaje:



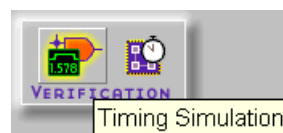
También podemos encontrarnos que nos dé errores la implementación, en cuyo caso deberemos buscar el origen de los mismos, para lo cual lo mejor es pinchar en la pestaña "*Reports*" del *Project Manager* y abrir el fichero llamado "*Implementation Log File*", en donde nos encontraremos un resumen de todos los pasos de la implementación. Si aun así no localizamos el error, podemos mirar en la carpeta "*Implementation Report Files*" donde hay muchos más informes detallados de cada una de las fases de la implementación:



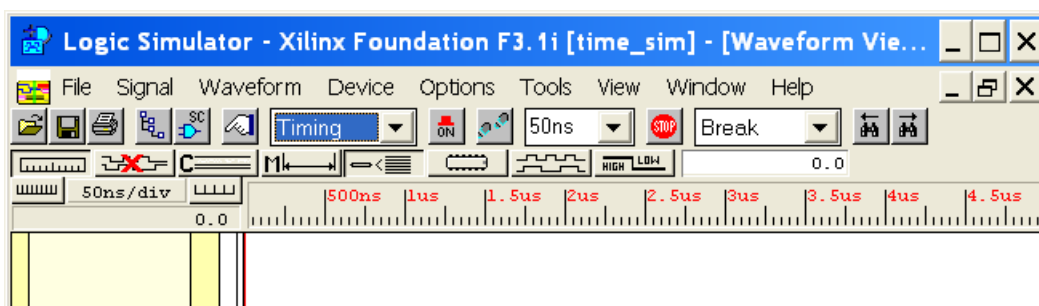
4.5. Verificación del diseño.

En esta fase del diseño, una vez implementado, es conveniente volver a realizar una simulación del circuito pero teniendo en cuenta los datos de dicha implementación, los cuales nos sirven para tener los retrasos prácticamente reales que tendrán las señales dentro de la CPLD, es decir, mientras que en un primer momento realizamos una simulación funcional, sin ningún tipo de retraso en las señales del circuito, ahora realizamos una simulación temporal (llamada también verificación) de manera que se aplicarán unos retrasos típicos en las señales con los datos extraídos de la implementación realizada.

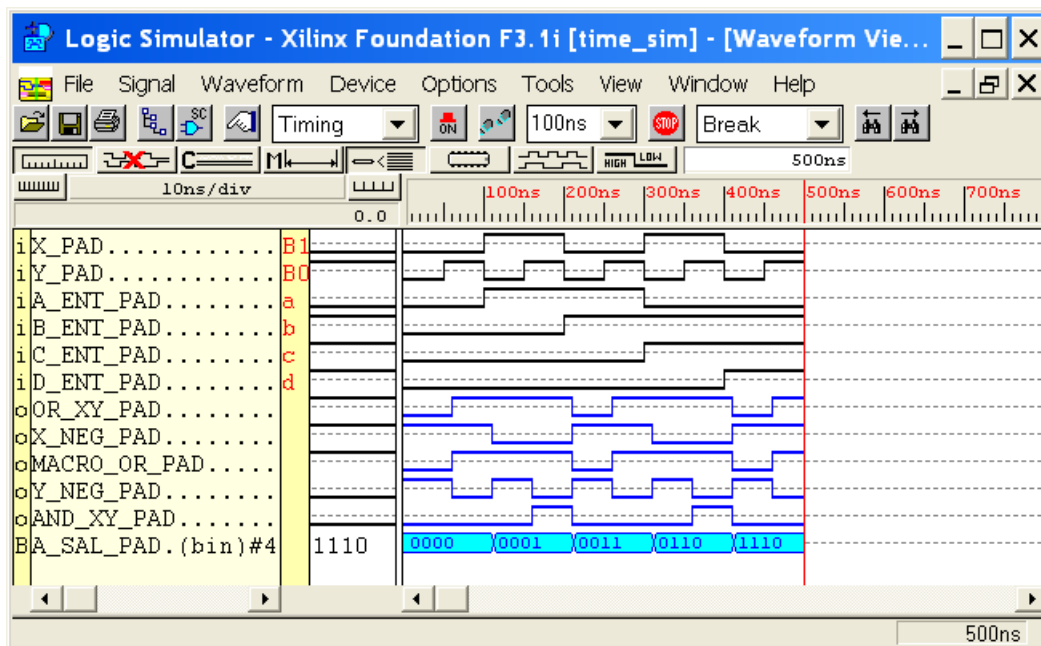
Para realizar esta verificación, pulsaremos en el botón de la izquierda de la casilla “Verification” dentro del “Project Manager”:



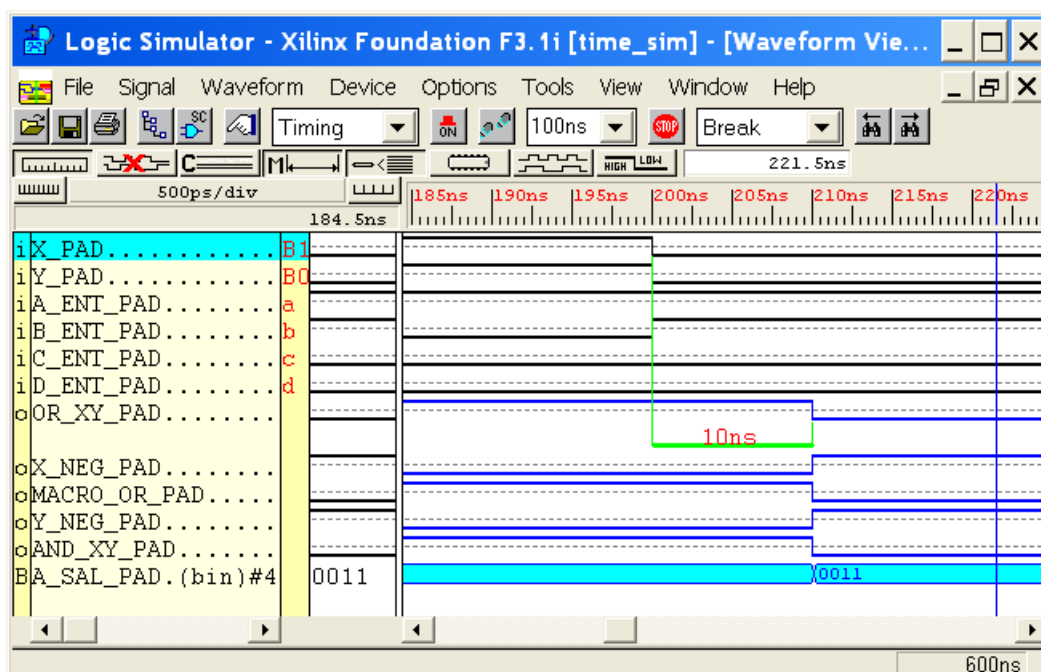
Se abrirá un simulador con un aspecto visual similar al que el estuvimos utilizando antes, pero ahora tendrá una diferencia, ya que estará marcado como tipo “Timing” en lugar de “Functional”:



La idea sería repetir la misma simulación que hicimos antes, pero con tiempos de retraso y midiendo éstos. Para ello, en el ejemplo del tutorial, repetiríamos los mismos pasos que vimos en el apartado de simulación, añadimos todas las señales, estímulos, y ejecutamos la simulación, quedando de la siguiente forma:



Aparentemente las formas de onda son iguales que las que obtuvimos en el apartado 3.3.3 de la simulación funcional, pero sí se puede apreciar que ahora las señales de salida, que aparecen en azul, no cambian exactamente a la vez que las entradas, sino que hay un pequeño retraso hasta que se produce dicho cambio. Este tiempo se puede medir a través del menú “Waveform → Measurements → Measurements On”, con el cual activaremos un nuevo tipo de cursor con una flecha; lo que haremos será marcar con la punta de la flecha un flanco en el que cambie una entrada, y posteriormente pincharemos en otro flanco donde cambie una salida, y mediremos la diferencia de tiempos entre estos dos eventos. Será necesario utilizar un zoom adecuado para realizar esta operación, ya que el tiempo a medir es muy pequeño:

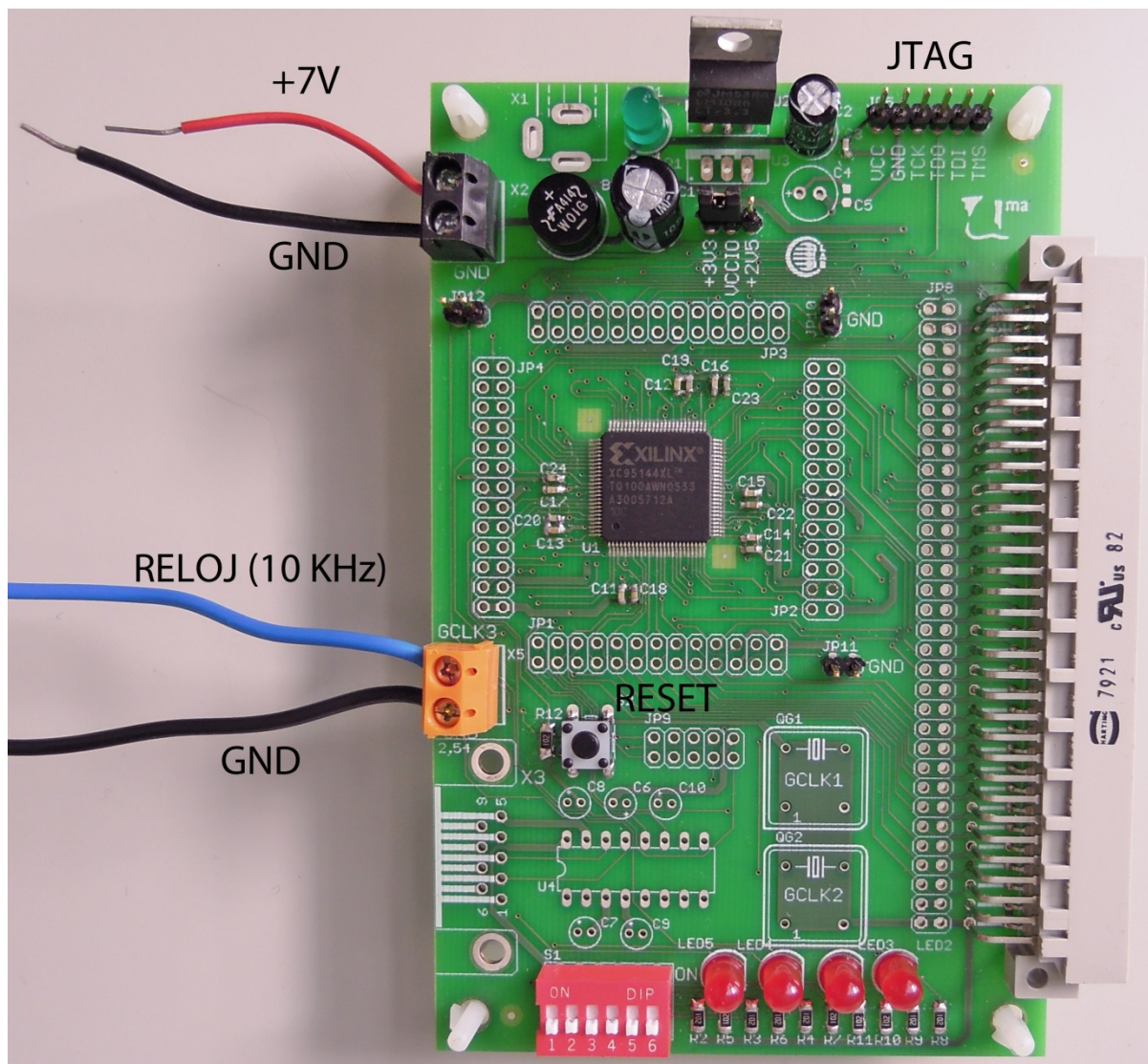


Vemos cómo se ha medido el retraso entre que cambia la señal de entrada X_PAD hasta que cambia la OR_XY_PAD a la salida, por ejemplo. Dicho retraso es de 10 ns, que coincide lógicamente con el grado de velocidad de este modelo de CPLD, que nos indica el retraso máximo que presenta desde que cambia una señal de entrada hasta que cambia, combinatorialmente, una de salida.

4.6. Programación de la CPLD.

El último paso a realizar será la programación de la placa de la CPLD para comprobar físicamente el funcionamiento del diseño. Para ello conectaremos la placa de la CPLD siguiendo estos pasos:

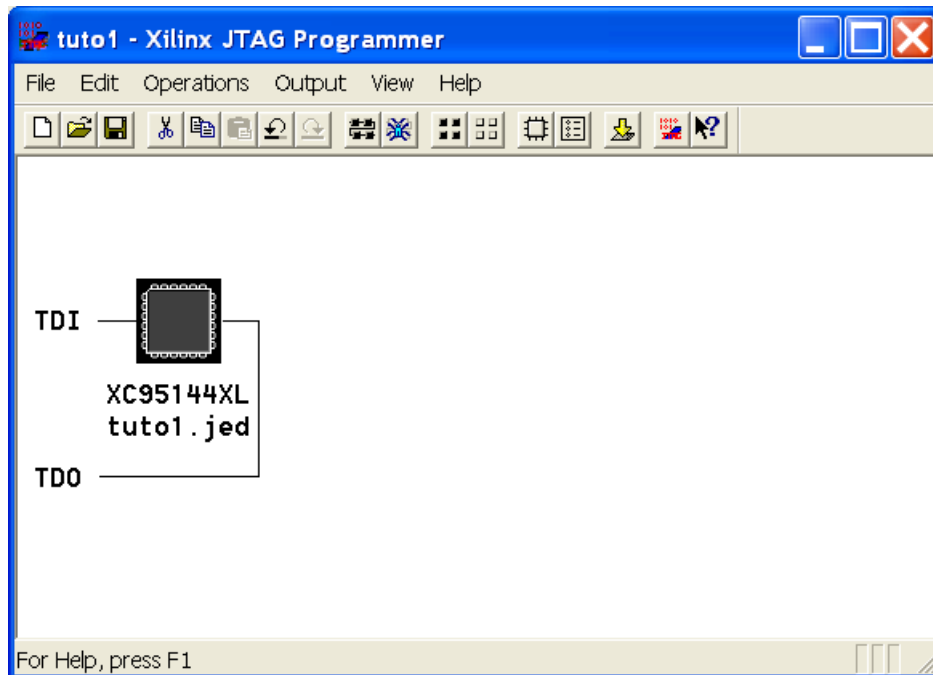
- Con ayuda de una fuente de alimentación, introduciremos 7V en la placa a través del conector negro que se aprecia en la figura. En el cable rojo pondremos el positivo de estos 7V y el negro irá conectado a tierra (GND).
- Conectamos el cable del JTAG al conector de 6 pines que hay en la placa, teniendo especial cuidado de hacer coincidir los pines en el orden correcto. Como ayuda vemos que está serigrafiado en la placa el orden de estos pines, así como en el cable JTAG, para que no se cometa la equivocación de conectarlos al revés. El otro extremo del cable lo conectaremos al puerto paralelo del PC.
- Mediante el generador de funciones regularémos (ayudándonos del osciloscopio) una señal de onda cuadrada de 10 KHz de frecuencia y que oscile entre los 0V (valor mínimo) y 5V (valor máximo), y la introduciremos con ayuda de una sonda en el conector naranja, usando el cable azul para el polo positivo y el negro para el negativo (Nota: puede que el cable azul esté cambiado por uno de cualquier otro color, siempre distinto al negro).



A continuación, en el “*Project Manager*”, pulsaremos sobre el icono “*Programming*” para invocar la parte del software encargada de programar la CPLD:

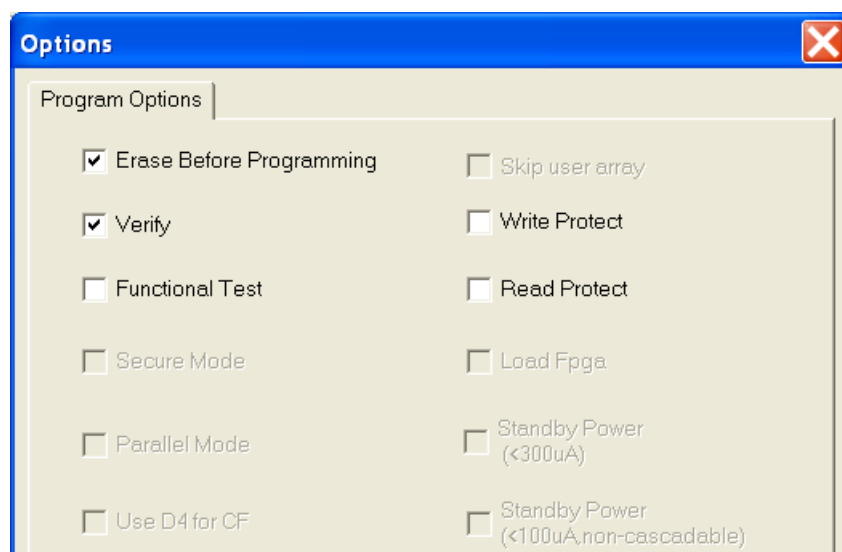


Se abrirá la siguiente pantalla:

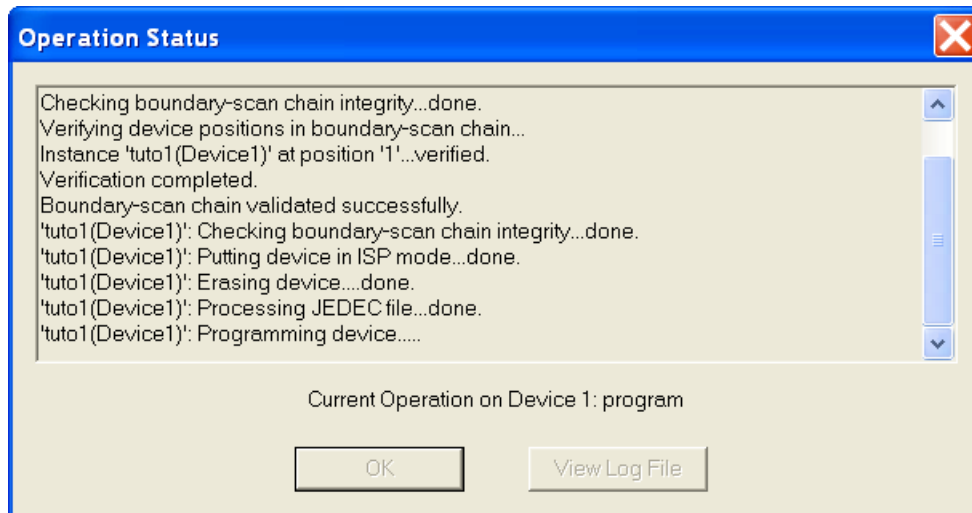


Los pasos siguientes son muy simples:

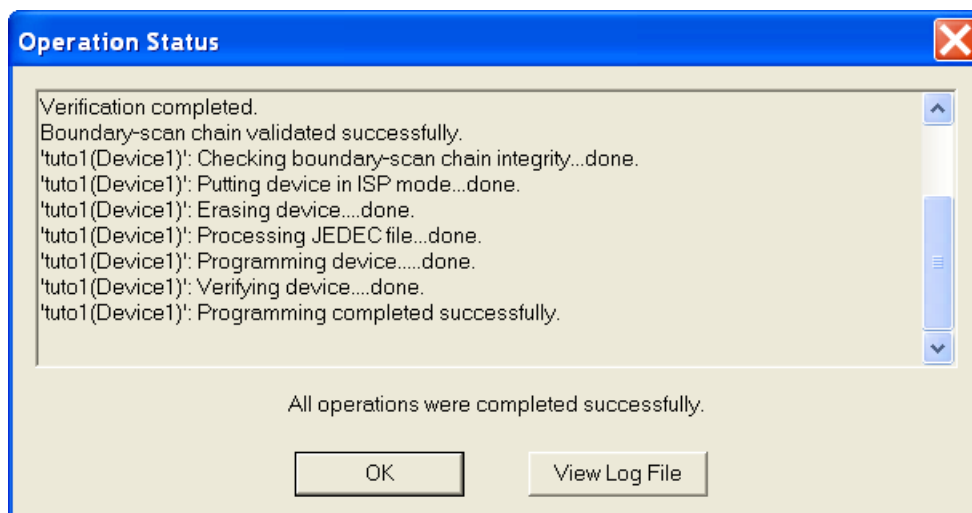
- Seleccionamos el componente XC95144XL pinchando sobre él.
- Iniciamos la programación entrando en el menú “*Operations* → *Program*”, momento en el que aparece una pantalla. En ella comprobaremos que estén marcadas las casillas “*Erase Before Programming*” y, muy importante, la de “*Verify*”, para que chequee el software que la programación ha tenido éxito:



- Pulsamos sobre “Aceptar”, y el proceso de programación comenzará:



- La programación estará acabada cuando en la ventana aparezca el mensaje “*All operations were completed successfully*”. Si ha habido un error en la programación lo indicaría el software y sería necesario repetir la programación:



Página intencionalmente en blanco

PRÁCTICAS DE FUNDAMENTOS DE ELECTRÓNICA

EL MUNDO DE LA ELECTRÓNICA ES MUY AMPLIO, Y EN ESTE MANUAL SE PRETENDE ABORDAR UNA PRIMERA TOMA DE CONTACTO CON EL MISMO. ES POR ELLO QUE SE HAN PREPARADO UNA SERIE DE PRÁCTICAS DE SIMULACIÓN Y MONTAJE ENFOCADAS A INTRODUCIR AL ALUMNO EN LOS DISPOSITIVOS ELECTRÓNICOS MÁS BÁSICOS (FUENTES DE TENSIÓN E INTENSIDAD, RESISTENCIAS, DIODOS Y TRANSISTORES) ASÍ COMO EN LA BASE DE CUALQUIER CIRCUITO DIGITAL (PUERTAS LÓGICAS Y ELEMENTOS DE MEMORIA, ESTO ES, BIESTABLES). LA PRIMERA PARTE DE ESTE MANUAL SE ENFOCA A REALIZAR PRÁCTICAS PARA TOMAR DESTREZA EN LA SIMULACIÓN E IMPLEMENTACIÓN FÍSICA EN EL LABORATORIO DE CIRCUITOS BASADOS EN DISPOSITIVOS ELECTRÓNICOS BÁSICOS Y EN SEMICONDUCTORES, CON EL OBJETIVO DE PODER CREAR PUERTAS LÓGICAS CON LOS MISMOS, TRAS LO CUAL LAS ANALIZAREMOS Y ESTUDIAREMOS SU COMPORTAMIENTO. LA SEGUNDA PARTE DE ESTE MANUAL SE ENCAMINA A DOTARLE AL ALUMNO DE LOS CONOCIMIENTOS NECESARIOS PARA PODER CREAR UN ESQUEMÁTICO CON UN CIRCUITO DIGITAL REALIZADO A BASE DE PUERTAS LÓGICAS Y ELEMENTOS DE MEMORIA (BIESTABLES), ASÍ COMO PARA SIMULARLO E INCLUSO PODER PROBARLO FÍSICAMENTE EN EL LABORATORIO A TRAVÉS DEL USO DE UNA PLATAFORMA DIGITAL PROGRAMABLE COMO ES UNA CPLD.

