ANEXO PRÁCTICAS 3.

# XILINX FOUNDATION F 2.1i

## 3.0.- INDICE.

# 3.1.- Introducción a las herramientas de diseño de XILINX Foundation F2.1

El sistema CAE de Xilinx es una herramienta de desarrollo que consiste en un conjunto integrado de herramientas software y hardware para crear, simular e implementar diseños digitales en una FPGA o CPLD. Todas las herramientas usan una interfaz de usuario gráfica que permite usar todos los programas desde iconos, menús o barras de herramientas. También se dispone de ayuda en línea desde la mayoría de ventanas.

### 3.1.1.- Pasos para realizar el Diseño.

El flujo de diseño para CPLDs o FPGAs consiste en tres pasos principales, ilustrados en la figura 3.1.

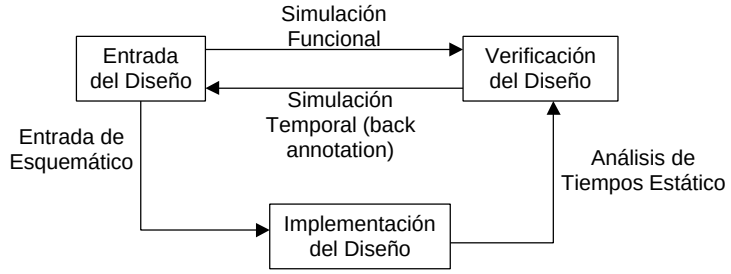



Figura 3.1: Visión general del flujo de diseño.

El proceso de diseño consiste en (a) Entrada del Diseño, (b) Implementación del Diseño y (c) verificación del diseño. Los diseños se pueden realizar de varias formas: Usando un Editor de Esquemáticos para dibujar un diagrama lógico, usando una descripción textual orientada al comportamiento del circuito (ABEL o VHDL), o una combinación de ambos. Las herramientas también permiten la realización de diseños jerárquicos.

Las herramientas de implementación sintetizan el diseño realizado en la arquitectura del dispositivo seleccionado. Las herramientas están automatizadas y compilan el diseño en un fichero de configuración que es el que se usa para programar el dispositivo real, optimizado en términos de uso de puertas lógicas e interconexiones para el dispositivo dado.

La programación del dispositivo final se puede realizar fácilmente desde un PC en una FPGA (usando una placa de prueba o un programador) o en un CPLD. Ambos dispositivos también se pueden programar en el sistema (ISP) conectando un **cable JTAG (MultiLINX) o Xchecker** a las patillas de programación del dispositivo.

La verificación del diseño incluye la simulación funcional, que se realiza sobre el ordenador, el testeo del circuito, que se realiza tras su programación, y la simulación temporal (una vez rutado el dispositivo).

### 3.1.2.- Tipos de dispositivos: FPGA o CPLD.

Hay dos tipos de dispositivos lógicos programables con los que se puede trabajar. Unos se denominan Array de Puertas Programable (Field Programmable Gate Array o FPGA) y el otro se denomina Dispositivo Lógico Programable Complejo (CPLD). Emplearemos principalmente el

XC4003EPC84-4 que funciona a 5V o el XC4005XLPC84-4 que funciona a 3.3V pero con patillas de E/S tolerantes a 5V.

Los dispositivos están disponibles con gran variedad de encapsulados. Los que vamos a usar están empaquetados en plástico con 84 patillas en formato PLCC y tienen los siguientes números de dispositivo: XC4010XLPC84, XC4005XLPC84 y XC4003EPC84. Tendremos que especificar estos nombres de dispositivo cuando comencemos un nuevo proyecto para que las herramientas puedan optimizar el diseño para el dispositivo especifico que usemos. Una vez tengamos el diseño, resulta muy fácil cambiar el componente destino de nuestro diseño como veremos en la sección de implementación del diseño.

Se puede encontrar información detallada de estos dispositivos en el Libro de Datos de lógica programable de Xilinx. La disposición de patillas del XC4000 la podemos encontrar al final de este anexo.

## 3.2.- Creación de un nuevo proyecto: Entrada de Esquemáticos.

Supongamos que queremos construir un sencillo circuito conbinacional, como muestra la figura 3.2. Este circuito se usará en un coche para generar una señal de aviso cuando “La llave de contacto (marcha) este puesta Y la puerta este abierta O no se usen los cinturones.

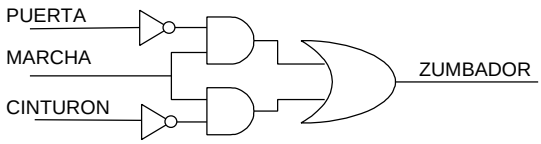



Figura 3.2: Diagrama del esquemático usado en el diseño.

La expresión boleana correspondiente es la siguiente:

**ZUMBADOR = MARCHA & ( !PUERTA # !CINTURON )**

, donde & denota la operación AND, # la operación OR y ! la operación NOT.

### 3.2.1.- Abrir un proyecto nuevo.

Para arrancar las herramientas de Xilinx Foundation (hacer click en el icono sobre el escritorio o bien ir al menu **Inicio** de Windows -> **Programas -> Electrónica Digital -> Xilinx Foundation Series -> Xilinx Foundation Program Manager**). Aparecerá primero una pequeña ventana preguntando si abrir un proyecto existente o crear uno nuevo. Seleccionar “**New Project**”. Esto provoca que aparezca la ventana de proyecto nuevo:

- Rellenar el nombre del proyecto, el directorio donde queremos almacenar el proyecto, y el tipo. Rellenar en **project Name**: EasyProj. El nombre de proyecto puede estar en minusculas o mayusculas.
- **Directory**: Seleccionar el directorio correcto (carpeta) donde queremos guardar nuestro proyecto. Usar el botón Browse para cambiar el directorio.
- Para el tipo, **Type** seleccionar F2.1i.

- Debajo de **Flow**, seleccionar Schematic (en el caso que pretendamos introducir el diseño como un esquemático. Las tres cajas desplegables bajo la ventana de proyecto nuevo hacen referencia a la familia del dispositivo, el código de componente y el sufijo de velocidad.
- La etiqueta **Family** se refiere a la familia del dispositivo. Seleccionar XC4000XL.
- La etiqueta **part** se refiere al dispositivo especifico. Seleccionar XC4005XL-3PC84C, este será el dispositivo más probable que usemos.
- La etiqueta **speed** se refiere a la especificación de velocidad del dispositivo que se puede encontrar impreso en la parte superior del dispositivo siguiendo al guión tras el nombre del dispositivo. Se puede cambiar el dispositivo empleado posteriormente haciendo click sobre el icono **Design Info** en la ventana del Project Manager (Fig. 3.3).

La ventana del Project Manager se muestra en la figura 3.3. Esta ventana muestra el flujo de diseño junto con las herramientas asociadas.

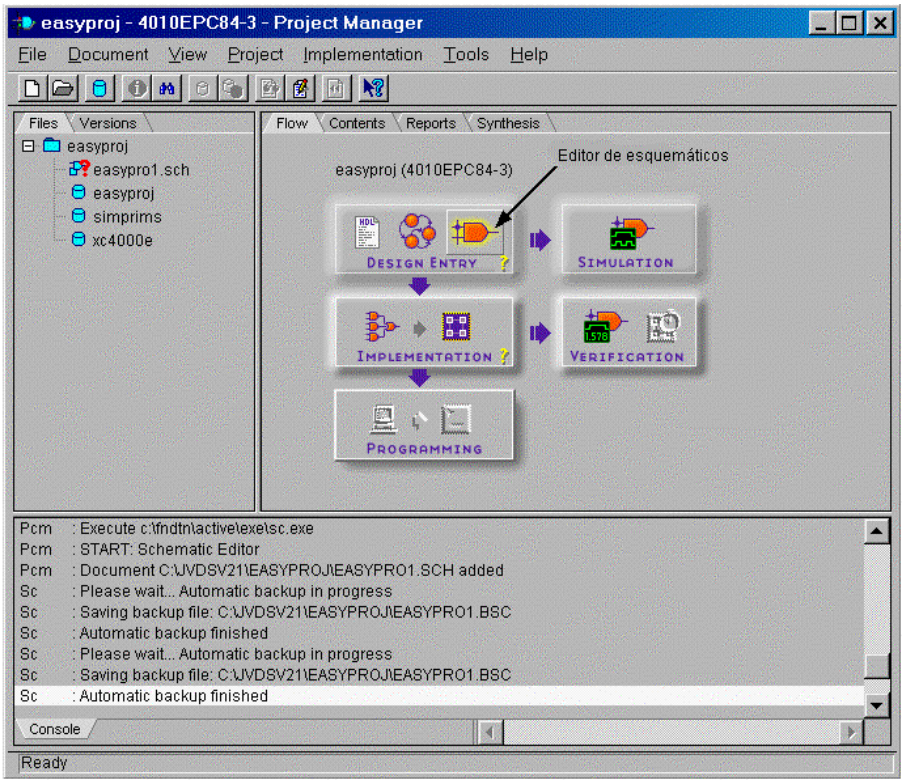


Figura 3.3.- Ventana del gestor de proyectos de Xilinx Foundation.

El proyecto tendrá una extensión .PDF. Otros ficheros de proyecto como esquemáticos, listas de componentes, macros, etc., se almacenarán en un subdirectorio con el nombre de proyecto. Un proyecto puede tener tan solo un esquematico principal. Después se pueden añadir sub-esquemáticos al proyecto como macros.

Para conseguir más información sobre el gestor de proyecto, usar la función de ayuda en línea seleccionando HELP -> FOUNDATION HELP CONTENTS -> PROJECT MANAGER, en la ventana del gestor de proyectos.

### 3.2.2.- Entrada del esquemático.

Para crear el circuito de la figura anterior usando el editor de esquemáticos hay que hacer click sobre el icono del editor de esquematicos en la ventana del gestor de proyectos o seleccionar el menu **TOOLS -> DESIGN ENTRY -> SCHEMATIC EDITOR**. Una ventana de captura de esquematico aparece como la de la figura 3.4.

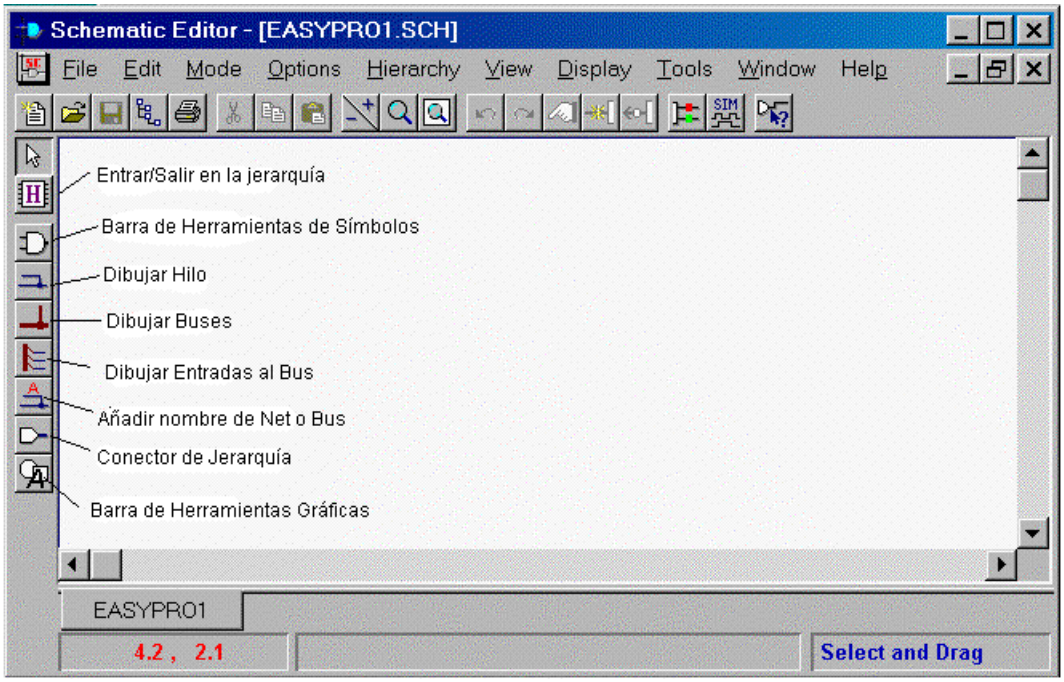



Figura 3.4.- Ventana del capturador de esquematicos con esquema en blanco de Xilinx Foundation.

#### Colocación de los símbolos

Podemos añadir los símbolos lógicos haciendo click en el icono de la puerta NAND (icono de Símbolo) en la barra de herramientas de la izquierda. Surge una ventana de símbolos. Se puede hacer un scroll sobre la lista y seleccionar AND2 o escribir el nombre del simbolo en la caja inferior de la lista. Una breve descripción del simbolo seleccionado aparece abajo. Ahora se puede colocar la puerta lógica con el cursor sobre el esquematico haciendo click con el ratón. Para colocar otra puerta AND solo hay que hacer click en la anterior y una segunda puerta lógica se engancha en el cursor. Cada vez que se hace click otra puerta del mismo tipo se coloca sobre el esquemático. Colocar las puertas AND de dos entradas. Luego seleccionar OR2 de la lista de simbolos y colocar una puerta OR.

Hay que tener cuidado de no colocar los simbolos demasiado cerca entre ellos de forma que parezca que están conectados. En realidad no están conectados electricamente. Los símbolos deben conectarse a través de hilos, asi que hay que asegurarse que se deja algo de espacio entre las puertas lógicas para que se puedan cablear. El test de integridad no avisa de este tipo de errores. Sin embargo, en la versión F2.1i, las puertas que estén colocadas juntas se conectarán automaticamente. También, el test de integridad de la versión F2.1i informará que las puertas no están conectadas. La figura 3.5 muestra el esquemático con los simbolos añadidos.

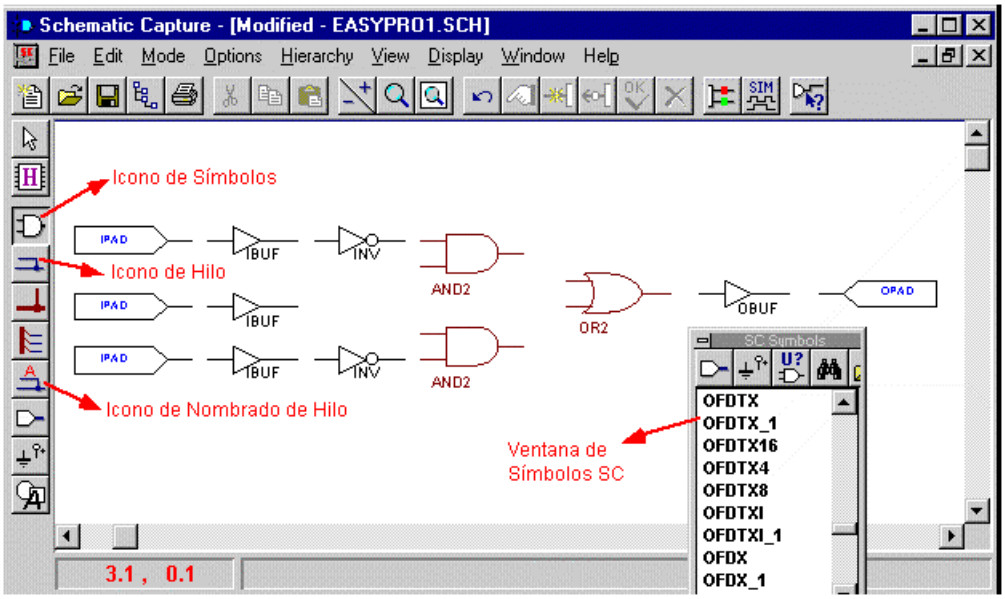



Figura 3.5.- Ventana del capturador de esquemáticos con los simbolos añadidos.

Se puede dar la vuelta a un simbolo pulsando **Ctrl+M** o rotandolo pulsando la tecla **Ctrl+L**. Para redibujar la pantalla pulsar la tecla de función F10. La FPGA contiene un oscilador de reloj en el chip de 8 MHz. Este se puede emplear colocando el símbolo OSC4 en el esquematico. Un divisor interno hace disponibles las siguientes señales: 8MHz, 500KHz, 16KHz, 490Hz y 15Hz. Las frecuencias no están muy bien definidas y pueden variar entre –50% y +25%.

Las FPGAs poseen bloques de entrada/salida que actuan como una interfaz entre la circuitería interna y las patillas que la conectan al mundo exterior. El bloque de E/S se puede configurar por el usuario. Consiste en buffers (amplificadores de corriente) de entrada y salida (designados por los símbolos de la librería como IBUF y OBUF), buffer de salida triestado (OBUFT) y flip-flops (flip-flop tipo D disparado por flanco (IFD) o como cerrojo (ILD)). Las patillas de salida están conectadas a la alimentación a través de una resistencia de pull-up de unos 10Kohm cuando no se encuentra en uso para prevenir salidas flotantes. Durante el funcionamiento normal, el pull-up está desactivado.

### Buffers

Los buffers son necesarios para las señales de entrada y las de salida que se dirigen a una patilla del dispositivo. Los buffers se colocan en el esquematico de forma similar a como se hace para los demás símbolos. Para los buffers de entrada se selecciona el símbolo IBUF y para el buffer de salida seleccionar **OBUF** de la lista de símbolos. En el caso que se desee añadir un buffer triestado, se puede seleccionar OBUFT. **No olvidar añadir buffers al esquemático** porque sino este no se podrá compilar después.

### Pads

También tendremos que añadir patillas (pads) de E/S a los buffers de entrada y salida. Estos pads representan a las patillas reales del dispostivo programable. Un pad es un componente físico en la Librería unificada de Xilinx y se coloca como cualquier otro componente. Los nombres de los pads son IPAD (entrada), OPAD (para salida), IOPAD (bidireccional), IPAD4 y OPAD4, etc. Todas las patillas del dispositivo DEBEN ser representadas con uno de estos pads de

entrada/salida. Se le debe dar un nombre a los pads y posiblemente un número de patilla (localización de la patilla). Esto se puede realizar haciendo doble click sobre el pad, aparece la ventana de propiedades del símbolo.

### Patillas

Un terminal de E/S (patilla), disponible haciendo click sobre el botón terminal de E/S a la izquierda de la barra de herramientas encima de la ventana de símbolos, no es un dispositivo físico y no se puede usar como una patilla de E/S. Los terminales de E/S solo deben usarse para proporcionar conexión entre niveles de jerarquia dentro del diseño. De esta forma se usan en macros para conectar señales a las patillas correspondientes sontre el símbolo de macro. Sin embargo, las señales que se distribuyen por múltiples páginas de un diseño plano no necesitan terminales ni conectores fuera de página. Para indicar patillas del dispositivo en el esquemático de mayor nivel de un diseño jerárquico, no se deben usar terminales de E/S sino pads para indicar las patillas del dispositivo.

*Si el esquemático va a convertirse en una celda o macro (subdiseño) que será usada posteriormente en un esquemático más grande necesitamos usar terminales de E/S para indicar los terminales del dispositivo.* Como se explicó antes se puede añadir un terminal de E/S (patilla) haciendo click sobre el botón Terminal y entrando el nombre del terminal. Para macros, no se necesita colocar buffers en el esquemático (IBUF o OBUF) puesto que las entradas y salidas no estarán conectadas a una patilla física de la FPGA o CPLD.

### Dibujando líneas y nombrando conexiones

Para conectar una puerta con otra, se usa el botón de dibujar conexión. Esto se realiza haciendo click sobre el símbolo de conexión justo debajo del icono de la puerta NAND. Todos los símbolos deben conectarse con hilos. No se deben colocar los símbolos juntos para guardar espacio para colocar los hilos de conexión.

Los hilos de conexión (*NETS*) deben definirse por el usuario con el propósito de mejorar la documentación y la legibilidad del diseño. Se puede nombrar un hilo haciendo click en el icono de hilo con una A debajo del icono de bus en la barra vertical. Escribir un nombre en la ventana Net Name y colocar el cursor sobre la línea de conexión. Asegurarse que apuntamos a la conexión que queremos nombrar, en caso contrario el nombre no estará conectado a nada. Un atajo para nombrar conexiones es hacer doble clic sobre ellas. Ahora podremos rellenar el nombre de conexión (netname) entre las patillas y los buffers para las tres entradas (PUERTA, MARCHA y CINTURON) y la salida (ZUMBADOR). Los nombres de conexión deben aparecer en azul (nombres en verde indican que el nombre no está conectado a nada).

### Añadir números de patilla

Podemos asignar números de patilla a cada pin de entrada y salida. Si no hacemos esto, el compilador asignará las patillas por nosotros. Hay dos formas de asignar números de patilla. Primero, podemos colocar las patillas en el esquematico usando la propiedad LOC. Esto se hace haciendo doble clic en el simbolo de PAD. En la ventana emergente de propiedades, vamos a la sección Parameters y entramos como Parameter Name: LOC, y para la Parameter Description: P#, en los que # representa el número de pin (la letra P es necesaria delante del número). Hacemos clic en el botón ADD. Para mostrar el número de patilla, hacer doble clic sobre LOC=P19 hasta que aparezcan dos diamantes al lado. Hacer clic en OK para terminar. El

nombre se puede mover haciendo doble clic sobre el pad hasta que aparezca la ventana de propiedades del símbolo. Seleccionamos LOC=P# y seleccionamos mover. Ahora podemos mover la etiqueta de patilla.

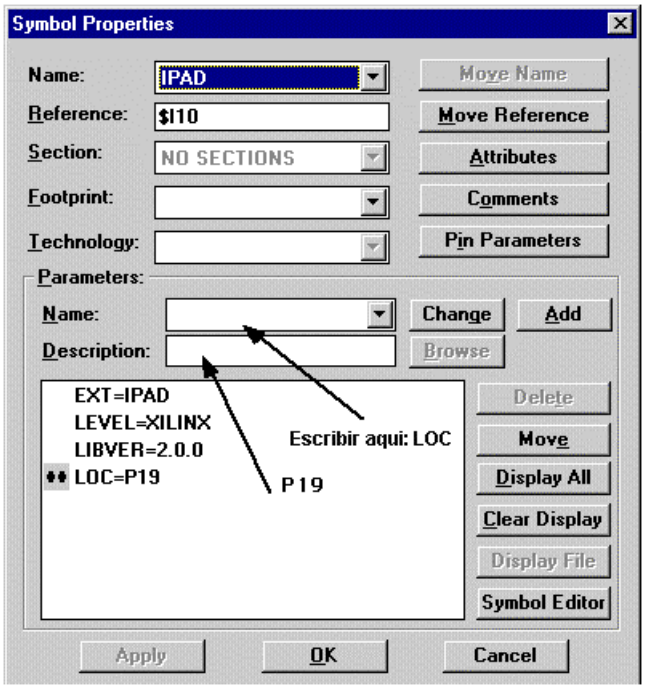



Figura 3.6.- Ventana de propiedades de símbolo para asignar patillas.

Una forma alternativa es no asignar números de patilla en el esquemático, sino especificarlos después, antes de compilar el diseño. Esto se realiza creando un fichero de restricciones de usuario (UCF). La ventaja de este método es que el esquemático es más genérico y se pueden cambiar fácilmente las patillas sin tener que modificar el esquemático.

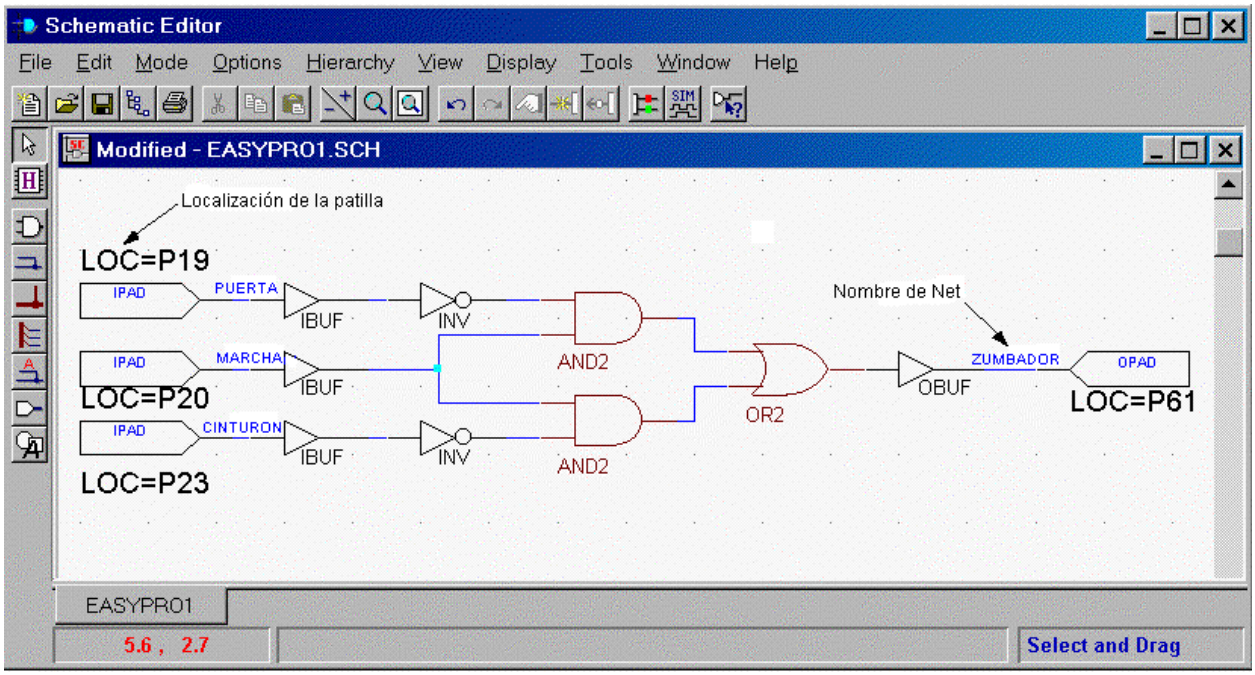



Figura 3.7.- Esquemático del circuito terminado incluyendo números de patilla.

Cuando usamos las placas de desarrollo de FPGA, ciertas patillas han sido preasignadas y algunas están conectadas a LEDs y conmutadores para facilitar el testeo (consultar la descripción de las placas de desarrollo). Asignar los números de patilla de forma que podamos hacer uso de estos dispositivos (leds y pulsadores).

### Añadir nuestro nombre, titulo del proyecto y fecha.

Resulta buena práctica etiquetar los esquemas de forma clara. Existe una forma estándar de hacer esto. Ir a la parte inferior de la página y rellenar el pequeño rectángulo. Si la caja tiene un nombre predefinido, podemos cambiarlo yendo a **File -> Table Setup**. Podemos cambiar ahora la dirección, nombre, Descripción, Fecha, etc.

### Creación del netlist y test de integridad.

Necesitaremos generar un netlist en un formato que sea legible por el compilador. Esto se realiza yendo al menú **OPTIONS -> CREATE NETLIST**. Cuando termine el proceso, siempre es buena idea comprobar que el esquemático no tiene errores de reglas eléctricas. Esto se realiza en el menú **OPTIONS -> INTEGRITY TEST**. También podemos generar ahora un netlist EDIF yendo al menú **OPTIONS -> EXPORT NETLIST**. Si no hacemos esto, no hay que preocuparse, el sistema nos preguntara después si debe hacer este paso.

### Guardar nuestro esquemático.

Ir al menú **FILE -> SAVE** o hacer clic en el icono del disquete en la barra de herramientas superior. Poner un nombre al esquemático con la extensión .SCH (p. ejemplo prueba1.sch). Cuando terminemos con el esquemático, salir del programa de captura de esquemáticos que nos trasladará de vuelta a la ventana del gestor de programas de Foundation.

### Añadir el esquemático al proyecto.

Si el documento creado no se encuentra listado en la ventana del gestor de proyectos dentro del proyecto que hemos creado (p. ejemplo prueba1.sch) tenemos que seleccionar el menú **DOCUMENT -> ADD** en la ventana del gestor de proyectos. Aparece una ventana con una lista de ficheros. Para mostrar solo el tipo "Esquemático" teclear *.SCH en la ventana del diálogo. Luego seleccionar el esquemático (prueba1.sch) que queremos añadir a nuestro proyecto (prueba1.sch).

### Copiar el proyecto a otro disco.

Si queremos guardar el proyecto en otro disco (p. ejemplo en un disquete) podemos hacer esto desde el gestor de programas. Abrir el menú **FILE -> COPY PROJECT**. Dar el nombre de la unidad de disco y el directorio donde queramos copiar el proyecto.

Podemos archivar también el proyecto como un fichero ZIP. En el gestor de proyectos, vamos a **FILE -> ARCHIVE PROJECT**. Esto abrirá la ventana del asistente de archivo de proyectos. El fichero zip es interesante cuando necesitamos mandar el proyecto por e-mail o cuando el fichero es muy grande para almacenarlo en un disco.

El próximo paso es compilar el circuito o simularlo. Veremos primero la simulación.

# 3.3.- Simulación del diseño.

## 3.3.1.- Simulación Funcional.

### Abrir el simulador

En este momento podemos realizar una simulación funcional para verificar que el circuito funciona adecuadamente. Hacer clic en el icono “Simulation” en la ventana del gestor de proyectos o seleccionar **TOOLS -> SIMULATION -> GATE SIMULATOR** de la ventana del gestor de proyectos. La versión actual del diseño se carga a continuación. Si aparece una ventana con el mensaje “Schematic Netlist prueba1 is older than schematic. Update netlist from Schematic Editor?” (El netlist del esquemático prueba1 es más antiguo que el esquemático. ¿ Quiere actualizarlo ? ) hacer clic en YES. Surgirá la ventana del simulador lógico de Foundation con la ventana del visor de cronogramas, como se muestra en la figura 3.8.

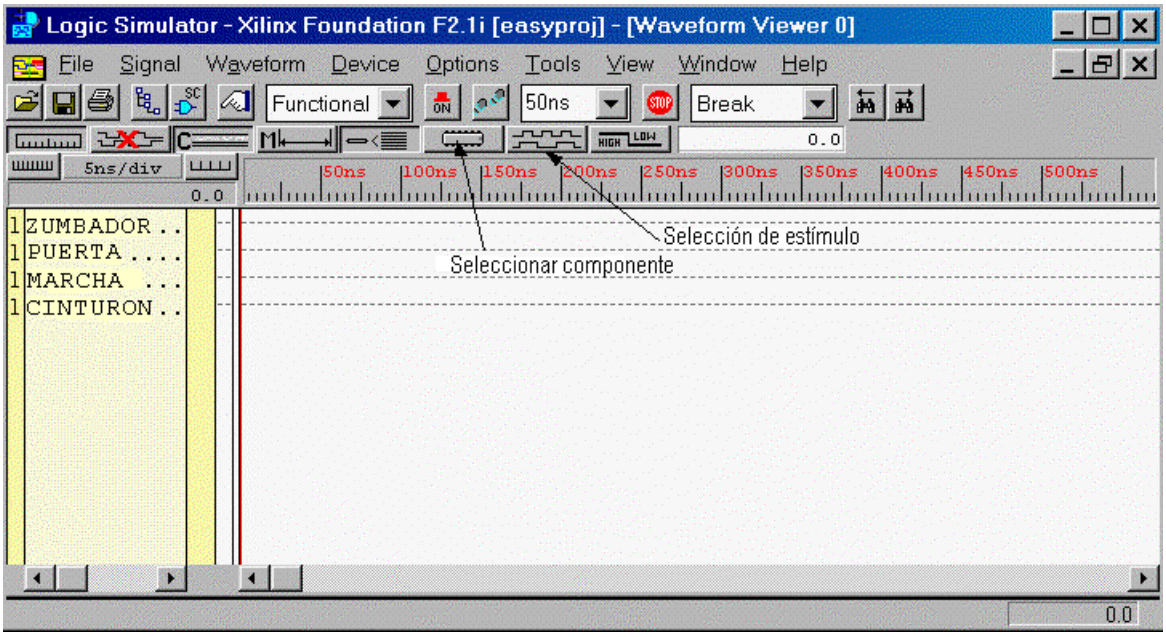



Figura 3.8.- Ventana del simulador lógico con el visor de cronogramas y la ventana de selección de componentes.

### Selección de las señales de entrada y salida.

Para simular el circuito hacemos los mismo que si estuviéramos en el laboratorio: añadimos señales de entrada a las patillas de entrada y vemos las señales de salida de entrada y salida en el cronograma. Así, el primer paso consiste en especificar las señales de entrada. Esto se realiza haciendo clic en el icono CHIP (seleccionar componente) que se encuentra sobre la ventana del visor de cronogramas o yendo al menú **SIGNAL -> ADD SIGNAL**. Ahora aparece otra ventana, denominada ventana de selección de componentes, como muestra la figura 3.9.

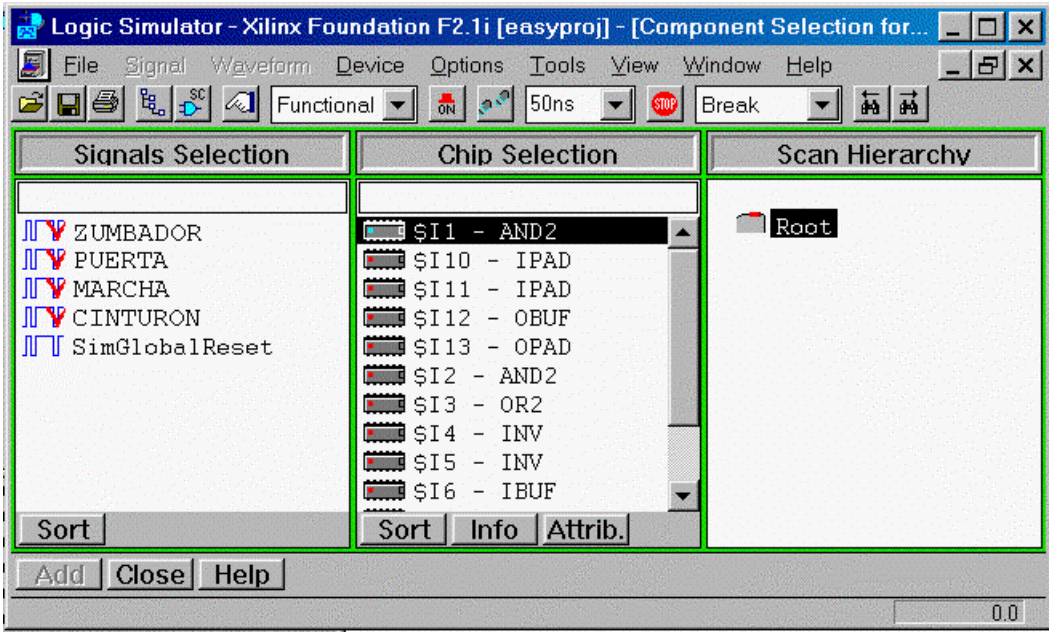



Figura 3.9.- Ventanas de selección de componentes.

En la ventana de selección de componente, seleccionar las entradas que queremos añadir al cronograma y hacer clic en el botón "Add" o doble clic en el nombre de señal. Hacer esto para todas las entradas y salidas. Cuando terminemos, hacer clic en el botón **CLOSE**. Las señales seleccionadas aparecerán en la ventana del visor de cronogramas.

Podemos seleccionar también patillas de test volviendo a los esquemáticos y añadiendo puntos de prueba a los hilos de conexión:

- Para cambiarse al editor de esquemáticos, hacer clic en el botón SC en la barra de herramientas horizontal o ir al menú **TOOLS -> SCHEMATIC CAPTURE**. El editor de esquemáticos pasará a ser la ventana activa.
- Seleccionar el menú **MODE -> TESTPOINT** que mostrará la caja de herramientas SC Probe. La herramienta de puntos de prueba es la que se encuentra arriba a la izquierda y es la seleccionada por defecto. Para añadir un punto de prueba a la línea, hacer clic en la etiqueta de la línea (la conexión debe tener una etiqueta). Una pequeña caja gris aparecerá sobre la conexión que cambiará de color a medida que la simulación se realice.
- También podemos añadir puntos de prueba a buses. El valor del bus se mostrará en formato hexadecimal.
- Para volver a la ventana del simulador, hacer clic en el icono SIM en la caja de herramientas SC Probes.

### Añadiendo estímulos.

A continuación debemos añadir estímulos adecuados a las señales de entrada.

- Esto se realiza haciendo clic en el icono del generador de estímulos (sobre la barra de herramientas, el icono con las dos ondas cuadradas) o yendo al menú **SIGNAL -> ADD STIMULATORS**.

- Aparece una nueva ventana que parece muy complicada, como muestra la figura siguiente.
  - La sección del teclado se usa para definir la simulación interactiva. Pulsando una tecla en particular, el generador de estímulos cambiará entre 0 y 1. Para asignar una tecla a una señal, primero seleccionar la señal y luego hacer clic en la tecla. Para poner la señal permanentemente a 0 o 1, usar la tecla 0 o 1, respectivamente en la sección de teclado.
  - Debajo de la sección de teclado hay dos filas de LEDs redondos y una fila de LEDs cuadrados. Los redondos representan los bits de un contador binario de 16 bits. Este contador se puede usar para generar señales de reloj de varias frecuencias con un ciclo de trabajo del 50 %.
  - La fila superior de LEDs, etiquetada Bc: ofrece la salida activa a nivel alto y la segunda fila, etiquetada NBc: da el formato activo a nivel bajo. Las primeras cuatro salidas empezando por la izquierda están etiquetadas como B0 a B3, las siguientes cuatro como B4 a B7, etc. Las salidas cambian siguiendo la secuenica: 0000 0000 0000 0000 -> 0000 0000 0000 0001 -> 0000 0000 0000 0010 ... -> 1111 1111 1111 1111 -> 0000 0000 0000 0000 ... Las salidas del contador NBc serán 1111 1111 1111 1111 -> 1111 1111 1111 1110 ....
  - Los LEDs cuadrados, etiquetados como Form: permiten asignar funciones lógicas definidas por el usuario. Esto se explica un poco más abajo.

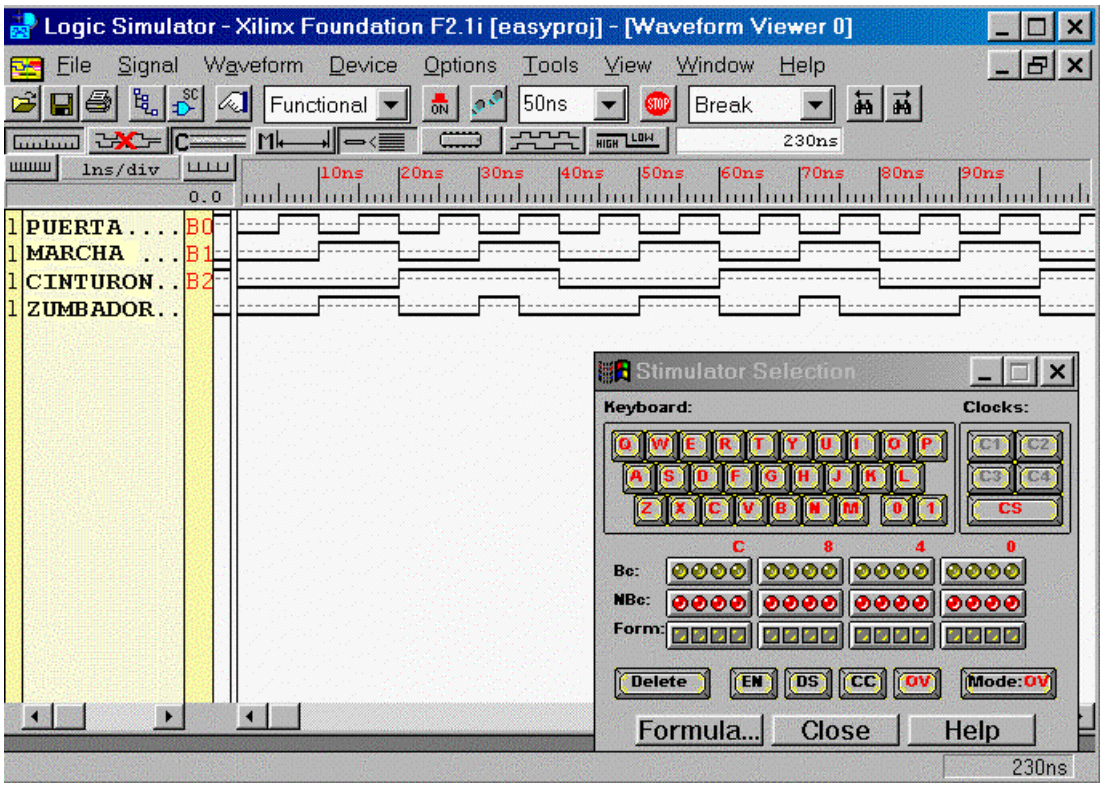



Figura 3.10.- Ventana del generador de estímulos y ventana de simulación.

En nuestro caso necesitamos usar tres salidas del contador. Usaremos los tres bits menos significativos del contador Bc. Hacer clic primero en el nombre de señal en el visor de cronogramas (ejemplo: PUERTA) de forma que quede resaltado, luego hacer clic en el lado más a la derecha de la salida del contador (etiquetado 0). Aparecerá el nombre B0 tras el nombre de

señal en el visor de cronogramas. Hacer lo mismo para las otras tres señales de entrada (seleccionar las siguientes dos salidas del contador).

Podemos asignar también una tecla del teclado a la señal, para cambiar una señal manualmente. Esto se hace seleccionando una de las teclas sobre la sección de teclado en la ventana del generador de estímulos. Para más información, pulsar el botón **HELP**. Cuando hayamos terminado cerramos la ventana de selección del generador de estímulos.

### Definición de la frecuencia del contador.

Podemos definir la frecuencia de reloj del contador binario. Ir al menú **OPTIONS -> PREFERENCES**. Bajo la sección Stimulation, hacer clic en la lista desplegable "B0 period" y seleccionar el periodo necesario. La frecuencia se ajustará de forma automática. La precisión depende del tipo de simulación. Una simulación funcional requiere menos precisión (por ejemplo 1ns) mientras que una simulación física (*timing simulation*) requiere una precisión más alta (p. ejemplo 100ps o menos).

### Realización de la simulación y del cronograma.

En este momento podemos realizar una simulación funcional. Escogemos "Functional" del menú desplegable que se encuentra sobre la ventana del simulador. Luego, hacemos clic en el icono de periodo de simulación (paso corto) sobre la ventana para iniciar la simulación. Las entradas y los cronogramas correspondientes a las salidas aparecen, como muestra la siguiente figura. Para cambiar el tamaño de paso, usar el menú desplegable cera del icono de paso de simulación. Podemos cambiar también la escala del eje de tiempos haciendo clic en el icono de regla sobre la lista de señales. La escala de tiempos también se muestra.

Podemos volver a la ventana de esquemático haciendo clic en el icono SC de la ventana del simulador. La ventana de esquemático mostrará los niveles lógicos (0 o 1) de cada una de las señales añadidas a la ventana del visor de cronogramas. En la ventana de puntos de prueba SC podemos hacer clic en el icono **STEP** para avanzar a lo largo de la simulación. Los resultados aparecerán sobre el esquemático para las señales que hayamos añadido a la ventana del visor de cronogramas. Podemos añadir señales adicionales haciendo clic en el icono de punta de prueba en la ventana SC Probes. Hacer clic en las conexiones que queramos mostrar.

Ahora podemos verificar si el circuito funciona como se esperaba. Si tenemos problemas, comprobar la lógica del esquemático. Un problema común es que los símbolos no están conectados entre sí con hilos porque se han colocado uno al lado del otro. Una cosa que no se puede comprobar fácilmente viendo el esquemático.

Para limpiar el cronograma y comenzar la simulación de nuevo, ir a la opción del menú **WAVEFORM -> DELETE -> ALL WAVEFORMS WITH POWER ON**.

Tener en cuenta que la simulación funcional solo nos muestra como funciona la lógica, pero nada acerca de la temporización de las señales. Esto significa que no tenemos información de los retardos de las puertas (y por tanto frecuencia máxima de utilización), se pueden extraer riesgos o violaciones de tiempos de set-up y hold a partir de una simulación temporal. Una vez que hayamos compilado nuestro diseño para un dispositivo especifico, podremos realizar un análisis de tiempos.

### Mostrando señales y buses.

Si tenemos varias señales que mostrar, puede resultar más conveniente agruparlas formado un "Bus". Esto se puede realizar seleccionando primero las señales que queremos agrupar y yendo al menú **SIGNAL -> BUS -> COMBINE**. El valor de las señales del bus se muestra en notación hexadecimal (a no ser que se especifique otra). Podemos también deshacer un bus yendo al menú **SIGNAL -> BUS -> FLATTEN**. Un ejemplo de simulación con buses se muestra en la figura 3.11.

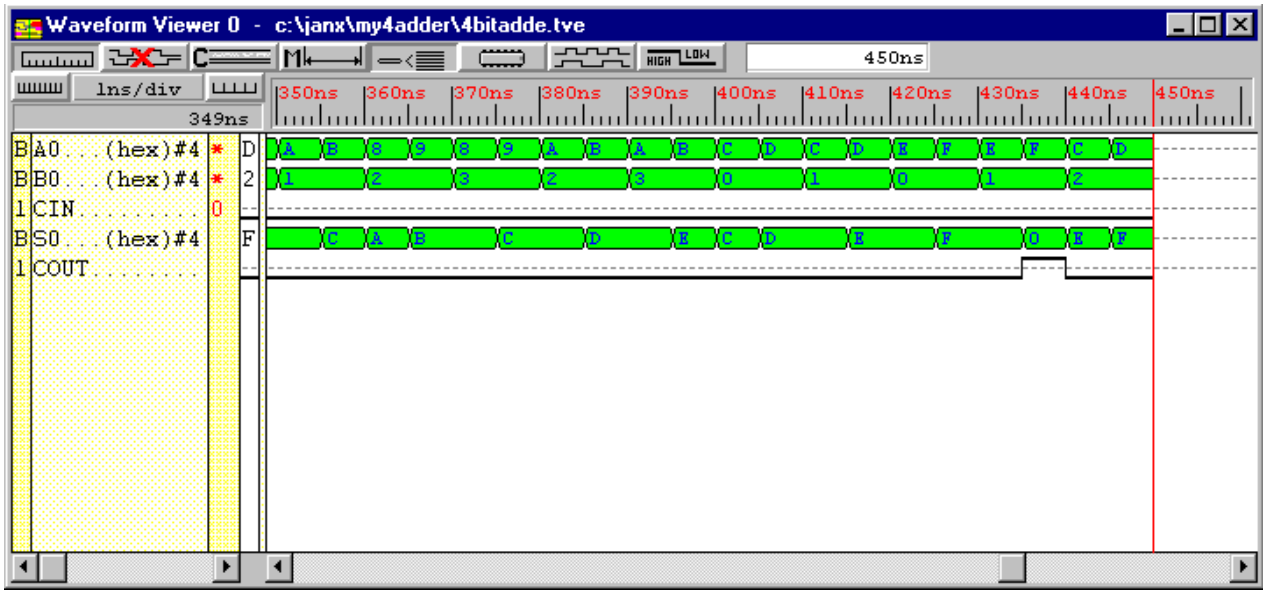



Figura 3.11.- Ventana del visor de cronograma en el que las señales se han combinado en buses.

Para ver las señales en un bus sin expandirlas, hacer clic en el icono BUS on/off (expandir) sobre la barra de menú superior del visor de cronogramas. El LSB de un bus se marca por un "*" y el MSB mediante un "$". Todas las demás señales se denotan por un signo "+". Si al bus se le da la vuelta (esto es, el LSB pasa a ser el MSB), podemos cambiar la dirección seleccionando el bus, haciendo clic con el botón derecho y seleccionando **BUS -> CHANGE DIRECTION**.

### Diseñando funciones de usuario.

Mencionamos antes que la tercera fila de los LEDs en la ventana de selección de estímulos se usa para asignar valores definidos por el usuario (o funciones lógicas) a una señal. Las funciones se pueden usar para asignar valores a una señal simple o a un bus. Esto se realiza haciendo clic en el botón Formula en la ventana de selección de estímulos. Esto hace aparecer la ventana "Set Formulas", como se observa en la figura 3.12.

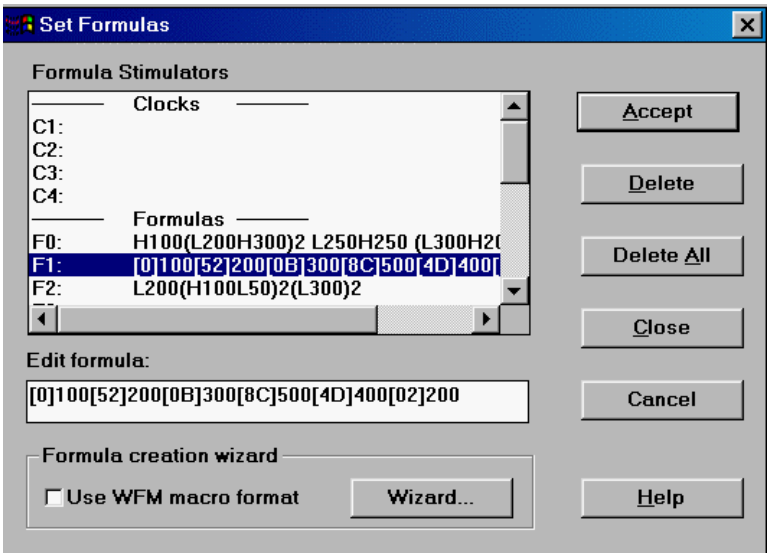



Figura 3.12.- Ventana de configuración de funciones usada para definir funciones lógicas.

Asignaremos una función lógica al LED F0.

- Hacer doble clic en F0: en la sección Formulas.
- En el campo Edit formulas escribir por ejemplo lo siguiente: H100(L200H300)2 L250H250 (L300H200)3. Esto define F0 como un estímulo que estará a nivel lógica alto durante 100ns, luego repite la secuencia (Bajo durante 200ns y Alto durante 300ns) dos veces, seguido por una señal a nivel lógico bajo de 250ns y alto de 250ns, etc.
- Hacer clic en el botón Accept.

Vamos a definir F1 para simular un bus.

- Hacer doble clic en F1.
- En el campo Edit formulas escribir por ejemplo lo siguiente: [0]100 [52]200 [0B]300 [8C]500 [4D]400 [02]200. El valor entre corchetes da el valor de la señal del bus en formato hexadecimal, mientras que el número que sigue al corchete especifica la duración. En el ejemplo anterior, la señal F1 tendrá un valor de 0(es decir, 0000 0000) durante 100 ns, un valor de $52_{(16}$ (es decir, 0101 0010), etc.
- Nota: el simulador ignora los espacios en las funciones.

Podemos usar también el asistente de creación de funciones para facilitarnos la creación de una función lógica. Seleccionar formula y hacer clic en el botón Wizard. Se abrirá la ventana del asistente de creación de funciones lógicas. Para asignar una función a una señal en particular, seleccionar la señal en la ventana del visor de cronogramas y luego hacer clic en el LED de la fila Form correspondiente.

### Como guardar los vectores de test y los resultados de la simulación.

Para guardar las selecciones de señal anteriores, ir al menú **FILE -> SAVE WAVEFORM**. Con esto se guardarán los vectores de test. También podemos guardar la simulación yendo a **FILE -> SAVE SIMULATION STATE**.

Cuando terminemos podemos guardar la simulación para usarla posteriormente o para imprimirla. Podemos especificar el tiempo de comienzo y de finalización del cronograma que queramos imprimir (esto nos ahorrará un montón de papel) yendo al menú FILE -> PRINT. En la ventana de impresión, rellenamos el Time Range (periodo de tiempo) del que queremos imprimir el cronograma (seleccionar el tiempo Start (comienzo) y End (Fin) del cronograma). No coger más de lo necesario. Rellenar también el periodo de tiempo por página. En general, el cronograma entra en una página.

### 3.3.2. Simulación Física.

La simulación física (timing simulation) nos ofrecerá información detallada sobre el tiempo que tarda una señal en pasar de una puerta a otra (retardo de puerta) y dará información de la respuesta del circuito en el peor caso. El retardo total de un circuito completo depende del número de puertas por las que circula la señal y de la forma que las puertas se han colocado en la FPGA o CPLD. De esta forma, la información de temporización solo se puede obtener tras la implementación del diseño (Place y Route) como se explica en la sección siguiente. Si no hemos hecho la implementación todavía, hacerla ahora. Cuando la implementación se termine, la información de tiempos se habrá generado y podrá ser usada por el simulador lógico.

La simulación de tiempos es muy similar a la simulación funcional descrita antes.

- En la ventana del gestor de proyectos, hacer clic en el botón VERIFICATION (primer icono en el botón). El simulador de tiempos se cargará y estará listo para usar.


- Seleccionar las señales en el visor de cronogramas y los estímulos, o cargar un cronograma creado anteriormente yendo al menú **FILE -> LOAD WAVEFORM**.
- Cuando todas las señales se hayan añadido y se hayan especificado los estímulos, **seleccionar TIMING de la lista desplegable** sobre la barra horizontal. Ahora podemos ejecutar la simulación haciendo clic en el icono de periodo de simulación corto o largo.
- En el caso en que hayamos usado Flip-Flops debemos activar el Global Reset. El reset global (GSR) se debe pulsar al comienzo de todas las simulaciones temporizadas. Esto pone a uno o a cero los flip-flops del esquemático. La señal de reset global no existe en el esquemático, pero existe en el dispositivo y en el netlist de la simulación temporizada. Para dispositivos XC4000, el nombre de conexión del reset global es GSR, y es activo a nivel alto. Para activar el reset global añadir la señal GSR al visor de cronogramas. En la ventana del generador de estímulos podemos asignar una función de usuario a uno de los LEDS Form, por ejemplo F2:H100L5000. Esto genera un pulso de reset a nivel alto durante 100ns y bajo durante 5000ns. Una forma alternativa es asignar uno de las teclas del teclado a la señal GSR y hacer el reset manualmente.

Podemos ahora ejecutar la simulación haciendo clic en el icono Step en la barra horizontal. Puede ser necesario ampliar el cronograma de forma que podamos ver los retardos de las puertas. Hacer clic en la escala de tiempos y arrastrar hacia una transición de reloj. Para medir el retardo, ir al menú **WAVEFORM -> MEASUREMENTS -> MEASUREMENTS ON**. Posicionar el cursor sobre el borde de la señal de interés para indicar el comienzo de la medida. Hacer clic en otra transición para completar la medida (ver figura adjunta). Tener cuidado que el periodo del

estímulo que estemos aplicando al circuito no sea mayor que el retardo máximo de la señal de salida. Por ejemplo, en la figura más abajo, la señal PUERTA debe permanecer constante durante al menos 5.3 ns, en otro caso la señal de salida (ZUMBADOR) no tendrá tiempo de permanecer en el valor correcto. Podemos cambiar el periodo (o frecuencia) de la señal de reloj yendo a **OPTION -> PREFERENCES -> SIMULATION**: seleccionar el valor correcto de B0: periodo de reloj.

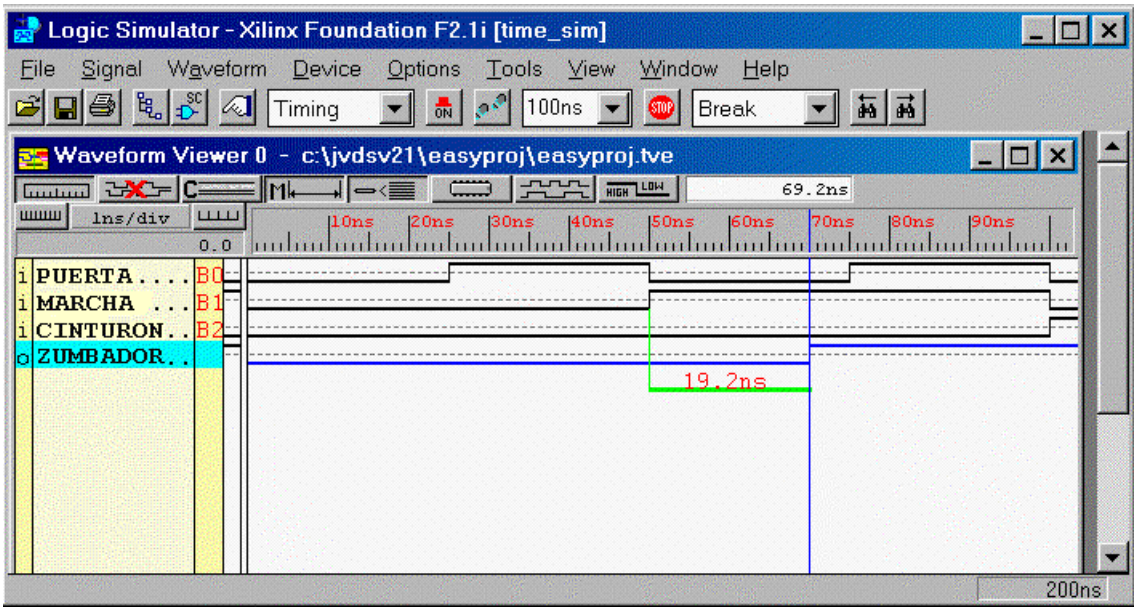



Figura 3.13.- Simulación temporizada, mostrando el retardo de la señal de salida (ZUMBADOR) en relación con la señal de entrada.

### Ficheros Script.

Una forma alternativa de definir las formas de onda y ejecutar el simulador es uar un fichero script. Para abrir el editor de script, ir a **TOOLS -> SCRIPT EDITOR**. Se abrirá una caja de dialógo. Podemos usar el asistente del editor de script que nos guiará a través de la creación del fichero script. Para más información de cómo crear un fichero script, hacer clic en el botón **HELP** o ir a **HELP -> HELP TOPICS** en la ventana del editor de script.

### Nota sobre señales indefinidas en una simulación.

Es posible que para un diseño complejo algunas de las señales se muestren como indefinidas (una caja gris en el visor de cronogramas o una X azul en el esquemático). Las señales que vienen de un circuito combinacional y se introducen en otro a menudo se colocan en el mismo CLB. Algunas de estas señales no son esenciales para el funcionamiento y el software de sintesis puede optimizarlas. Como resultado de este proceso, no se encuentran disponibles para la simulación temporizada. Podemos comprobar la lógica que ha sido eliminada yendo al informe de mapeo (ventana panel de la derecha – pestaña Report en la ventana del gestor de proyectos) tras realizar la implementación. Si queremos mantener la señal, podemos decirle al programa que lo haga. Hay dos posibilidades. Una el el fichero ucf, de la siguiente forma: net net_name keep; (en el que net_name es el nombre de la conexión que no queremos que se elimine). Otro método es colo el attributo “keep” en la conexión en el editor de esquemático.

- Hacer doble clic sobre la conexión.
- seleccionar “attributes”
- en “parameter name”, introducir “keep”
- seleccionar “add” (si queremos podemos mostrar los atributos)
- luego “ok”

## 3.4.- Realizando un diseño con ABEL – HDL.

Una forma alternativa al uso del editor de esquemáticos para introducir un diseño es usar una descripción del comportamiento del circuito. Las herramientas del entorno Xilinx Foundation F2.1 nos permiten introducir un diseño usando ABEL, VHDL o Verilog. Usaremos el lenguaje de descripción hardware ABEL (HDL) en una primera introducción. Si no se es muy familiar con ABEL, sería conveniente leer el capitulo de introducción a ABEL-HDL para una introducción rápida al lenguaje.

Podemos usar cualquier editor de texto para crear un fichero fuente ABEL mientras sigamos las convenciones y sintaxis del lenguaje ABEL como se describe en la introducción al lenguaje. Sin embargo, Foundation series nos proporciona una forma fácil de crear un fichero fuente: el asistente de diseño HDL. Este será el que usemos a continuación.

### Abrir un proyecto en el gestor de proyectos Foundation.

Si no estamos en el gestor de proyecto, abrirlo ahora. Vamos a crear un nuevo proyecto denominado AbelFac. Crearemos la misma función lógica que realizamos anteriormente en la sección de entrada de esquemáticos. Ir al menú **FILE -> NEW PROJECT**. En la ventana emergente, entrar el nombre, familia, tipo y velocidad del componente para el que se genera la lógica. En el caso que queramos añadir un fichero Abel al proyecto existente, podemos hacerlo creando una macro en Abel (ver la sección 3.8 sobre macros).

### Crear un diseño HDL y usar el asistente HDL.

En el gestor de diseño, hacer clic en el icono de editor HDL o ir al menú **APPLICATIONS -> HDL EDITOR**. Seleccionar HDL Design Wizard (Asistente para diseño HDL). En la ventana del asistente de diseño hacer clic en **NEXT** e ir a la ventana del asistente de lenguaje de diseño. Estaremos usando el lenguaje ABEL, luego seleccionar ABEL. Hacer clic en el botón **NEXT** con lo que se abrirá la ventana de nombre. Entrar el nombre del diseño. Vamos a llamarlo AbelFac. Los nombres no deben ser más largos de 8 caracteres. Si usamos nombres con más de 8 caracteres la síntesis dará errores. Sin embargo, el informe de errores no dirá lo que estuvo mal.

Hacer clic en **NEXT**. La ventana de puertos del asistente de diseño mostrará un símbolo a la izquierda. Seguir las instrucciones en esta ventana. Podemos crear aquí las patillas de entrada y de salida (ports). Hacer clic en el botón **NEW** y entrar el nombre (Name), seleccionar como Dirección (Direction) entrada o salida, dependiendo del tipo de puerto (port). Para los pines de salida vamos al botón AVANCED que abrirá la ventana de configuración avanzada. Seleccionar Combinatorial o Register, para un puerto combinacional o registrado (con flip flops). Para nuestro sencillo circuito, escogeremos combinacional. Cuando lo hayamos hecho, hacer clic en el botón FINISH. La ventana del editor HDL se abrirá en este momento.

### Crear el fichero fuente ABEL con el Editor HDL.

La ventana del editor HDL estará abierta con una plantilla que contiene el titulo, así como las declaraciones de patillas realizadas en el paso previo. Debemos verificar que están presentes todas las entradas y salidas. Luego tendremos que añadir la descripción lógica de nuestro circuito. Las descripciones lógicas pueden introducirse de varias maneras: Ecuaciones, Tablas de Verdad, y descripciones de Estados (para circuitos secuenciales). Usaremos una ecuación para definir la función lógica de nuestro circuito combinacional. Bajo la sección Equation (<<add your equations here>>) escribir la siguiente ecuación:

**ZUMBADOR = MARCHA & ( !PUERTA # !CINTURON);**

Recordar que en los nombres importan las mayúsculas. La figura siguiente muestra el fichero completo en ABEL. Podemos localizar los números de patilla en el código ABEL o podemos especificarlos más tarde usando un "fichero de restricciones de usuario" cuando compilemos el diseño (ver Editor de Restricciones en el capitulo de Implementación).

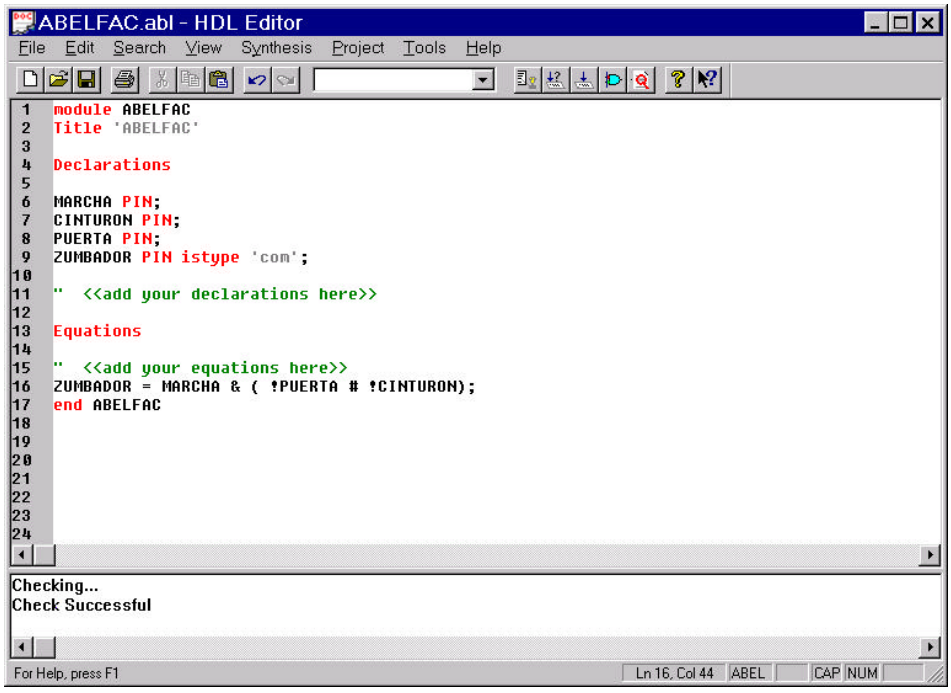



Figura 3.14.- Ventana del editor ABEL descibiendo el circuito de la Figura 3.1.

Para conocer algo más sobre la sintaxis de ABEL hacer clic en el icono del asistente de Lenguaje sobre la parte superior derecha de la barra de herramientas, o seleccionar **TOOLS -> LANGUAGE ASSISTANT**.

### Comprobar la sintaxis.

Vamos a asegurarnos que no hemos cometido errores de sintaxis. Ir al menú **SYNTHESIS -> CHECK SYNTAX**. Si hemos tenido éxito, surgirá una ventana que mostrará "Check Successful". También, el panel de la ventana inferior en el gestor de proyecto mostrará el estado del proceso o informará de cualquier error.

### Guardar el fichero

Hacer clic en el icono del disquete o ir a **FILE -> SAVE** para guardar el fichero fuente HDL. Podemos ahora salir de la ventana del editor HDL que nos devolverá a la ventana del gestor de proyectos.

### Añadir el diseño al proyecto.

Para hacer el fichero fuente HDL parte del proyecto prueba1, ir al menú **DOCUMENT -> ADD** en la ventana del gestor de proyecto. Mostrar ficheros de tipo: ABEL (*.ABL). Seleccionar el FacAbel.abl y hacer clic en OK. El fichero FacAbel.abl debe aparecer ahora en el directorio del proyecto sobre la ventana del gestor de proyectos (pestaña de ficheros).

En este momento podemos simular o compilar el circuito. Ver las secciones sobre simulación y compilación. Si vamos a simular el diseño, ir al menú **SYNTHESIS -> SYNTHESIZE** y luego hacer clic en el icono botón **SIMULATION** en la ventana del gestor de proyectos. Si sucede un error, comprobar que el nombre del fichero ABEL no tenga más de 8 caracteres.

## 3.5.- Editor de Diagramas de Estados.

Las maquinas de estados finitos se pueden especificar de varias formas. Una forma es usando HDL, como ABEL o VHDL. El **tutorial de ABEL** explica como usando las construcciones **STATE_DIAGRAM**, **IF-THEN-ELSE** o **TRUTH_TABLE** se pueden usar para especificar máquinas de estados finitas. Las herramientas Xilinx Foundation proporcionan un método alternativo para especificar una máquina de estado. Para usar el Editor de Estados, se debe haber instalado XABEL o X-VHDL (disponible en el PC del laboratorio; es también una parte de la Edición Estudiante de Xilinx). El Editor de Estados nos permite definir diagramas de estado usando una descripción gráfica y convertir la descripción gráfica en ABEL o VHDL.

Para mostrar el uso de un Editor de Estados, implementaremos un detector de secuencia. Implementaremos dos variaciones del mismo detector de secuencia, una implementada como máquina de Mealy y otra como máquina de Moore.

El detector de secuencia reconoce la secuencia siguiente “1011”. La máquina se mantendrá comprobando si la secuencia de entrada es correcta y no pasará al estado inicial tras reconocer la cadena correcta. En el caso que lo implementemos como máquina de Mealy, la salida estará asociada con transiciones como indica el siguiente diagrama de estado.

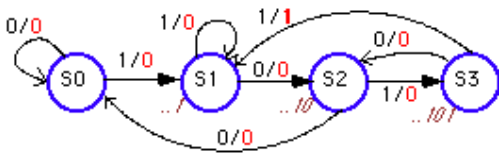



Figura 3.15.- Una máquina de Mealy para un detector de secuencia que detecta la secuencia 1011.

En los apartados siguientes explicaremos como definir el diagrama de estado anterior usando el Editor de Diagramas de Estados. Primero tendremos que crear un nuevo proyecto (o usar un proyecto existente).

### 3.5.1.- Crear una macro de maquina de estado.

En el gestor de proyecto, hacer clic en el botón Editor de Estados. Esto abrirá la ventana del editor de estados. Si queremos crear una nueva maquina de estado, seleccionar "Use the HDL Design Wizard" y hacer clic en OK.

En la ventana del asistente de diseño, podemos seleccionar ABEL o VHDL. Usaremos el lenguaje ABEL. Esto implica que el Editor de Estados convertirá la descripción gráfica en código ABEL (o VHDL). Especificar el nombre de fichero. Es buena práctica no usar nunca nombres más largos de 8 caracteres. Hacer clic en NEXT.

En la ventana Ports, definir las entradas: X, RST(para reset) y CLK (reloj). Definir también el port de salida Z. Hacer clic en el botón ADVANCED y seleccionar "combinacional" para la dirección de salida. Asumiremos que la salida del detector de secuencia viene de un circuito combinacional. Podemos hacer la máquina de estado síncrona seleccionando para la salida Z "registrada". Solo necesitamos definir una señal de reloj como uno de los ports de entrada. Nombrando uno de los ports de entrada CLK, la maquina de estado determina que CLK es la entrada de reloj. También proporcionaremos una señal de reset, RST, que nos permitirá poner el detector de secuencia a un estado conocido. Cuando todos los ports de entrada y salida se han definido hacer clic en NEXT.

En la ventana "Machines", seleccionar "one" y pulsar Finish. El nombre por defecto para el registro de estado es Sreg0. Esto abrirá la ventana del editor de estados en la que podemos describir gráficamente nuestro detector de secuencia, como muestra la figura 3.16. La parte superior de la ventana muestra los terminales de entrada y salida.

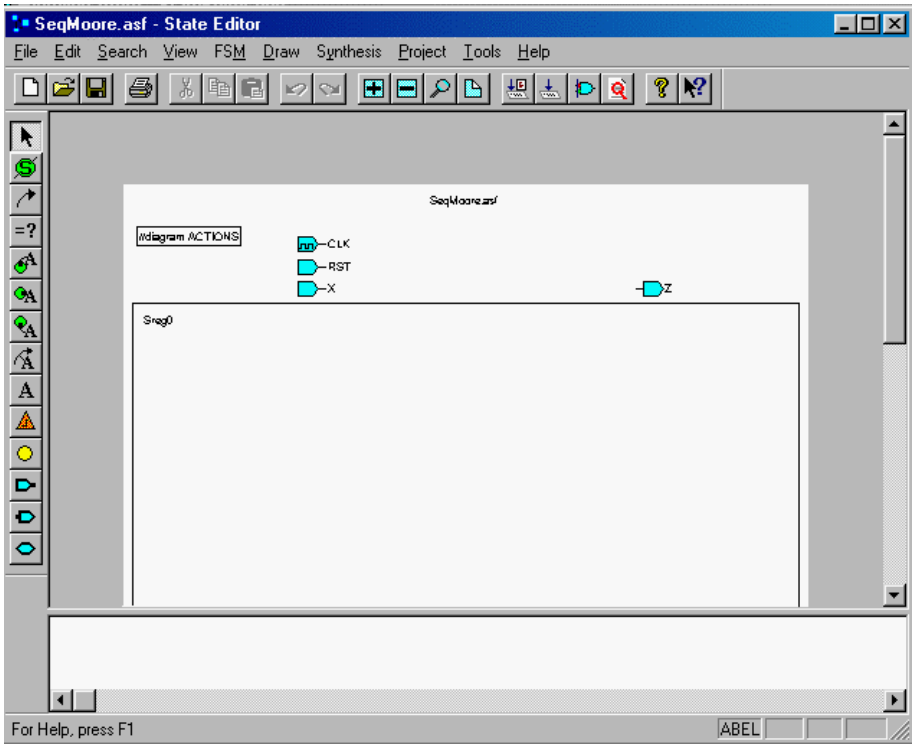



Figura 3.16.- Ventana del editor de estados usada para definir el diagrama de estado.

### 3.5.2.- Definiendo los estados.

En la ventana del editor de estados, hacer clic en el icono “S” en la barra de herramientas vertical izquierda o seleccionar FSM -> STATE del menú. Esto colocará una burbuja de estado en la ventana. Podemos cambiar su tamaño seleccionándola y arrastrando los cuadrados pequeños. Renombremos el nombre del estado haciendo clic sobre el nombre de estado en el centro de la burbuja. Hacer clic de nuevo para editar el nombre. Teclear S0 (o cualquier otro nombre que queramos darle al estado). Hacer esto para los otros tres estados como muestra la figura 3.17.

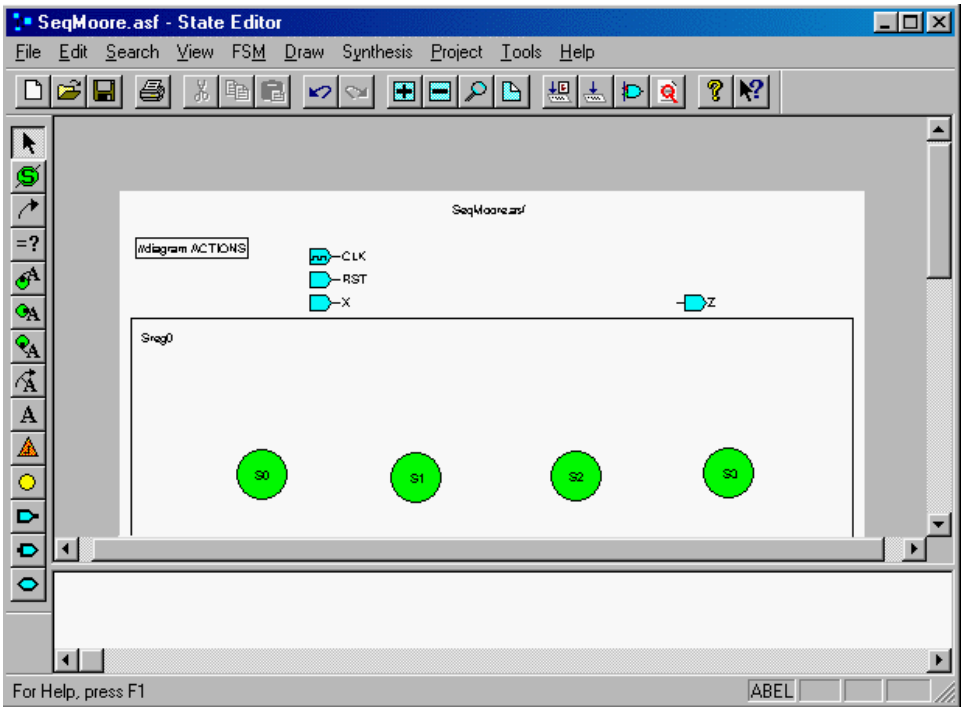



Figura 3.17.- Los cuatro estados del detector de secuencia.

Una máquina de estados debe tener un reset de forma que arranque en el estado correcto. Este reset no aparece en el esquemático pero la presencia del reset en el diagrama de estados dirige al compilador HDL a definir la codificación de estados de tal forma que la maquina comience en el estado correcto. Para añadir el reset al diagrama de estados, hacer clic en el icono reset (pequeño triángulo) en la barra de herramientas izquierda o seleccionar FSM -> RESET. Colocar el símbolo de reset sobre el diagrama. Queremos que la máquina comience en el estado S0. Esto se realiza haciendo clic dentro de la burbuja del estado S0. El reset puede ser síncrono o asíncrono. Haremos el reset asíncrono. Esto se hace con un doble clic sobre el símbolo de reset y seleccionando “ Asynchronous”.

### 3.5.3.- Definiendo las transiciones.

Un diagrama de estado queda definido al especificar bajo que condiciones de las señales de entrada la maquina pasa de un estado a otro. Estas transiciones vienen indicadas por flechas entre el estado y por el etiquetado de las condiciones de entrada que causan la transición como muestra la figura 3.18 para nuestro detector de secuencia. Definiremos primero las transiciones y luego las condiciones.

?? Hacer clic en el icono transición (icono con la flecha) en la barra de herramientas izquierda (o seleccionar **FSM -> TRANSITION**).

?? Colocar el cursor en el estado **S0** para comenzar la transición. Para crear un cambio de dirección en la flecha, colocar el cursor y hacer clic donde queramos que cambie de dirección. Luego, hacer clic dentro del estado **S1** para completar la transición. Hacer lo mismo para cada transición mostrada en la figura 3.15.

### 3.5.4.- Definiendo las condiciones.

El próximo paso es definir las condiciones asociadas con cada transición (flecha).

?? Hacer clic en el icono de condición (=?) o seleccionar FSM -> CONDITION.

?? Hacer clic en la flecha de transición, es decir, la transición entre el estado S0 y S1. Aparece una pequeña caja de texto.

?? Introducir la condición. Si usamos ABEL, necesitamos sintaxis ABEL para definir la condición. Para la transición de S0 a S1, teclear: X & !RST. La transición tendrá lugar cuando la señal sea “1”, es decir cuando X sea 1 y la señal reset esté negada.

?? Hacer lo mismo para todas las demás transiciones. Por ejemplo, la transición entre el símbolo reset y el estado S0 sucede bajo la condición: RST. De la misma forma, se mantiene en el estado S0 bajo la condición !X # RST (cuando X=0 o la señal RST se ha activado). Notar que hablando estrictamente no tendriamos que incluir la entrada RST en todas las transiciones de estados porque lo hemos incluido como un símbolo reset. Lo hemos incluido para demostrar condiciones de más de una entrada.

La figura 3.18 muestra el diagrama de estado con transiciones etiquetadas. Podemos seleccionar las condiciones y moverlas por el diagrama.

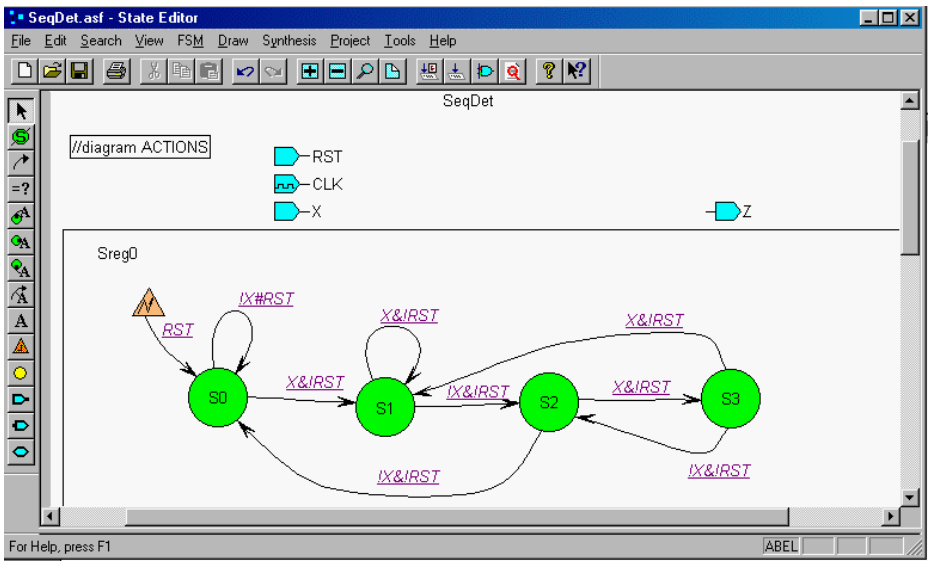



Figura 3.18.- Diagrama de estados del detector de secuencia con transiciones y condiciones.

### 3.5.5.- Definiendo las acciones.

Las acciones (señales de salida) se asocian con las transiciones como en el caso de la máquina de Mealy o bien con el estado como sucede con la máquina de Moore. En el ejemplo de nuestro detector de secuencia (máquina de Mealy) asociaremos acciones con transiciones.

?? Hacer clic en el icono de acción en transición de la barra de herramientas vertical o seleccionar **FSM -> ACTIONS -> TRANSITIONS**.
?? Hacer clic en la flecha de la transición. Aparece una caja de texto.
?? Teclear la acción deseada. Para la transición de S0 a S1, la salida Z pasará a valer 0. Esto viene especificado tecleando ^b0 en la caja de texto. Hacer lo mismo para todas las demás transiciones excepto para la transición de S3 a S1 que hace que la salida pase a valer 1: ^b1. En el caso que tengamos múltiples salidas necesitamos definir cada salida. Por ejemplo: OUT1=0;SUM=0; o en el caso de un bus: **OUTPUT=^b10**.
?? Podemos especificar la señal de reloj (en otro caso el editor de estados empleará la señal CLK definida en el asistente). Seleccionar **FSM -> MACHINE -> Sreg0**. Esto abrirá el diálogo de propiedades de la máquina de estados. Bajo General podemos seleccionar la señal de reloj y especificar si usamos el flanco ascendente o el descendente de reloj. Podemos usar esta ventana para especificar la señal reset también. Podemos especificar aquí el tipo de codificación: simbólica o codificada (binaria o definida por el usuario).

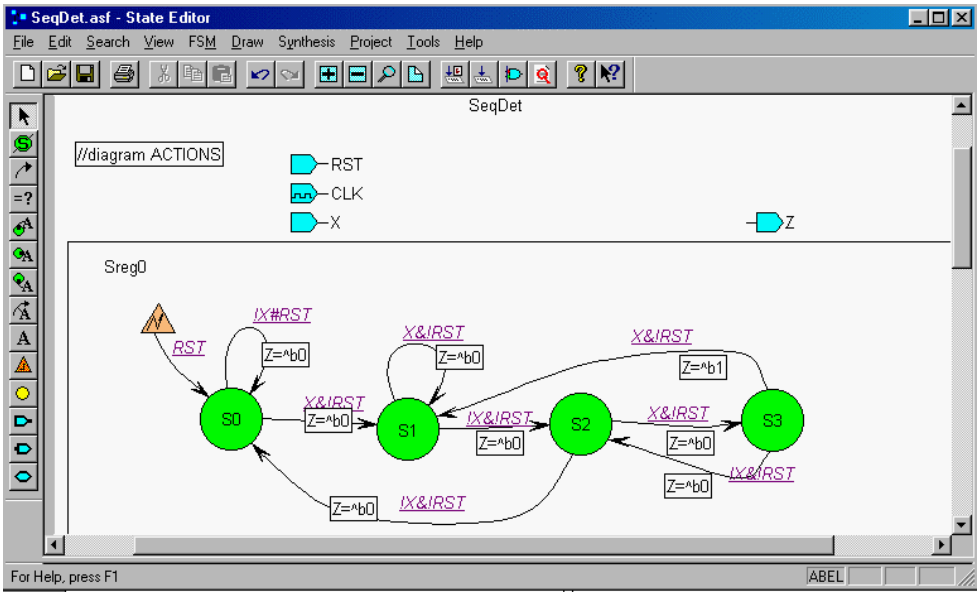



Figura 3.19.- Muestra del diagrama de estado completado.

### 3.5.6.- Generación del código HDL y compilación del diagrama de estado.

Guardaremos primero el diagrama de estados. Especificamos el tipo de codificación a usar para los estados (asignación de estados). Ir a **FSM -> MACHINE -> Sreg0**. Se abre el dialogo de propiedades, seleccionar **Symbolic encoding**.

En caso que queramos ver el código ABEL o VHDL, ir a **SYNTEHESIS -> HDL CODE GENERATION**. Se abrirá una ventana mostrando el código HDL. En este momento podemos crear una macro a partir del diagrama de estado yendo al menú **PROJECT –> CREATE MACRO**. Esto produce la síntesis del diagrama de estados y lo añade a la librería de símbolos.

Cuando abramos el editor de esquemáticos seremos capaces de colocar la macro desde la ventana SC Symbol. Tras colocar la macro podemos simularla para verificar que funciona correctamente.

En caso que haya errores, podemos ver el informe yendo a **SYNTHESIS -> VIEW REPORT**. También puede resultar instructivo comprobar el código HDL (si se es familiar con la forma de especificar un diagrama de estado en ABEL). Si no necesitamos una macro a partir del diagrama de estados, podemos ir al menú **SYNTHESIS -> SYNTHESIZE** para generar el netlist EDIF a partir del código ABEL. En caso de encontrar errores, consultaremos el informe (ir a **SYNTESIS -> REPORT**). También puede ayudar realizar una comprobación de síntesis sobre el código HDL generado como se explico antes. Por último seleccionar **PROJECT -> ADD TO PROJECT**.

### 3.5.7.- Simulación e implementación.

Una vez hayamos creado una macro o sintetizado el diagrama de estados podemos realizar una simulación. La simulación funcional de la máquina de Mealy del detector de secuencia se muestra en la figura 3.20. La salida es válida al final del tiempo que ocurre justo antes del flanco de subida de la señal de reloj. La salida se pone a 1 tras la secuencia 1011. Notar que la salida muestra algunos ruidos típicos de una máquina de Mealy asíncrona.

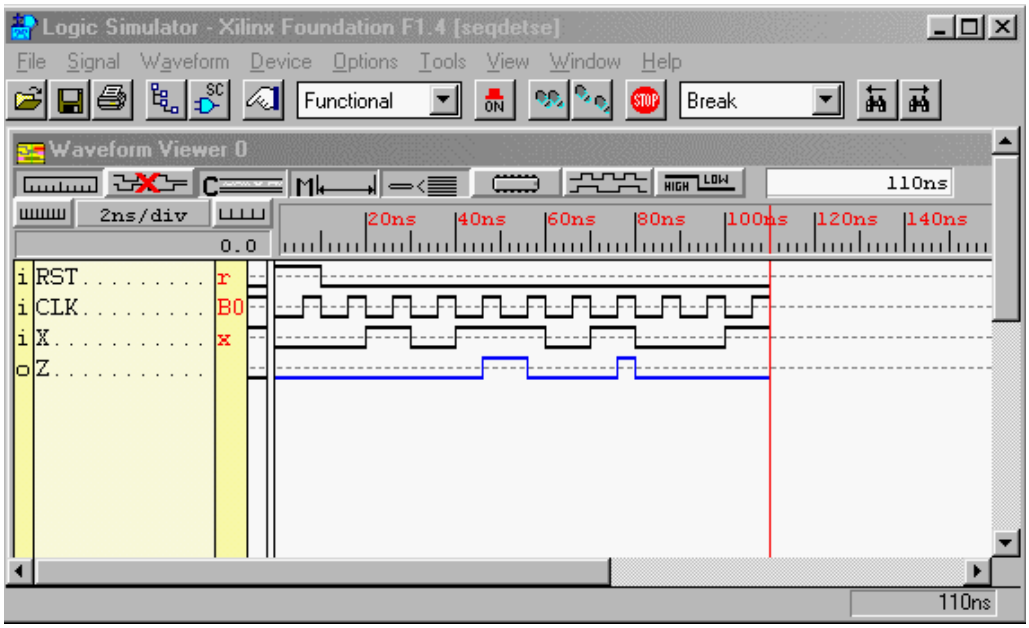



Figura 3.20.- Simulación funcional del detector de secuencia Mealy (secuencia 1011).

Cuando usamos para la codificación "Encoded (binaria)" en lugar de simbólica podemos seguir las transiciones de estado sobre el diagrama de estado durante la simulación. Esta es una buena posibilidad cuando comprobamos nuestro diagrama de estado. Arrancar el simulador. Abrir el diagrama de estado; esto se puede hacer usando el botón de Jerarquía en el editor de esquemáticos. Una vez abierta la ventana del editor de estados, ir al menú **TOOLS -> SIMULATION**. Esto activará la simulación gráfica. Lo mejor es tener ambas ventanas abiertas, el visor de cronogramas y el editor de estados de forma que se pueda seguir el estado de la simulación. La simulación gráfica no funciona con máquinas de Mealy.

Cuando el circuito funcione de forma correcta podemos implementarlo. Ir al gestor de proyectos y hacer clic en el botón Implementar.

### 3.5.8.- Ejemplo de máquina de Moore.

Si queremos eliminar los glitches en la salida de la máquina de Mealy, podemos implementar el detector de secuencia como máquina de Moore. El diagrama de estado viene dado por la figura 3.21. Tenemos que añadir un estado, S4, comparado con la máquina de Mealy.

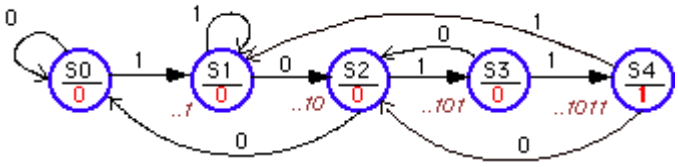



Figura 3.21.- Diagrama de estado del detector de secuencia para la secuencia 1011, implementado como máquina de Moore.

La correspondiente descripción del diagrama de estado se da en la figura 3.22. La diferencia con la máquina de Mealy, es que la acciones ahora no se asocian con un estado. Estas se pueden definir haciendo clic en el icono de Acción en Estado sobre la barra de herramientas vertical de la izquierda en la ventana del Editor de Estados (o seleccionando **FSM -> ACTIONS -> STATE**).

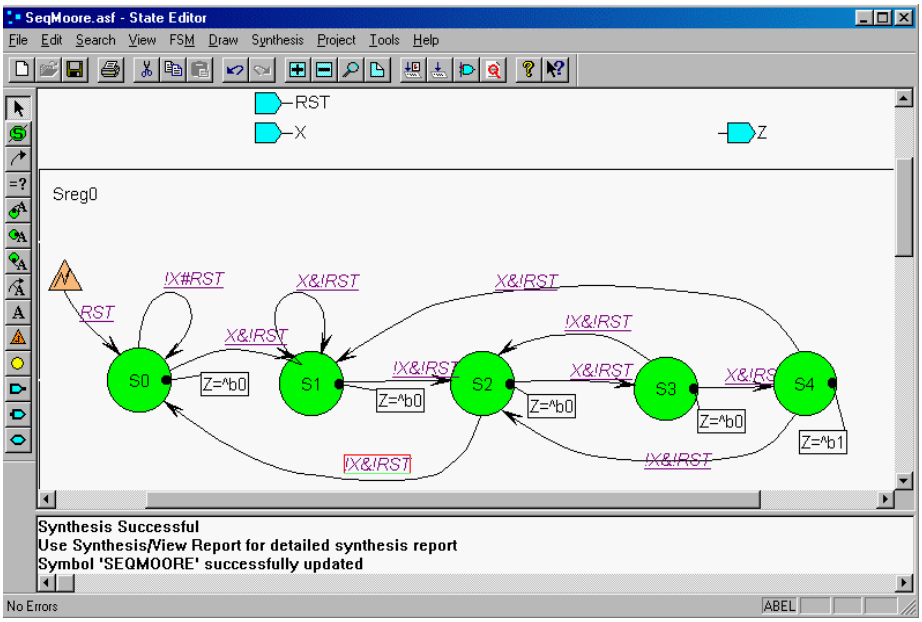



Figura 3.22.- Diagrama de estados completo del detector de secuencia implementado como máquina de Moore.

Para realizar una simulación gráfica usaremos la codificación de estados en lugar de la codificación simbólica. Seleccionar **FSM -> MACHINE -> Sreg0**. En la ventana de propiedades de la máquina seleccionar Encode: binary. Hacer clic en OK. Para ver el código ABEL, seleccionar **SYNTHESIS -> HDL CODE GENERATION**.

La simulación funcional de la máquina de Moore se muestra en la figura 3.23. La salida pasa a nivel alto tras la secuencia 1011. No se encuentran glitches en la salida si la comparamos con la máquina de Mealy (ver figura 3.20). La figura 3.24 muestra el simulador gráfico con el estado activo resaltado. Tener en cuenta que el editor gráfico solo puede usarse cuando el editor de Diagrama de Estado se abre desde el gestor de diseño o usando el icono de jerarquía en el esquemático.

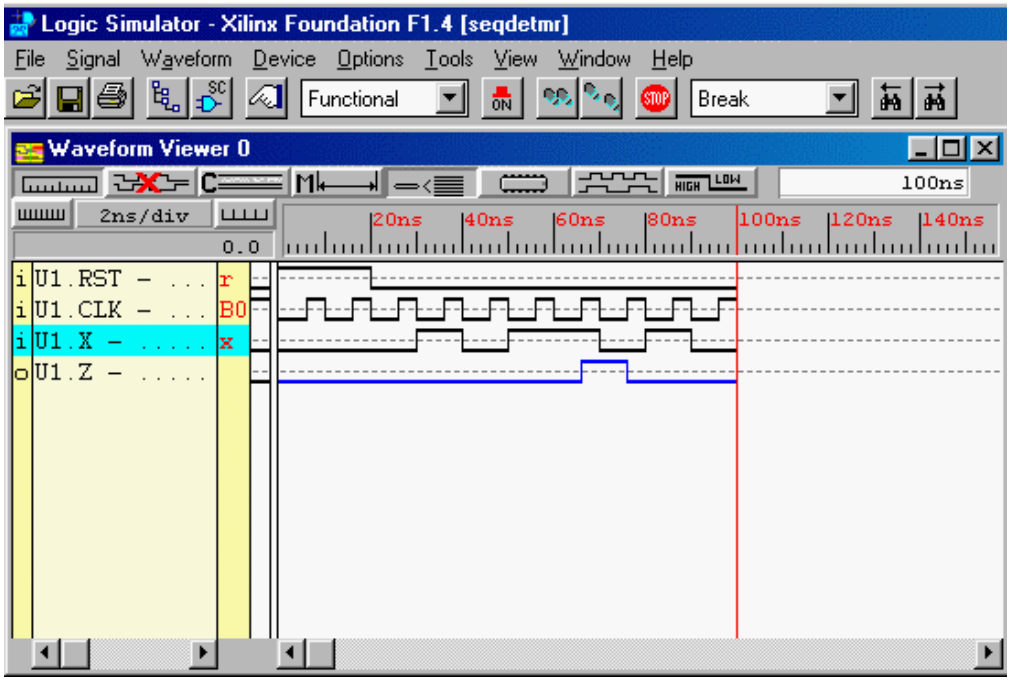



Figura 3.23.- Simulación funcional del detector de secuencia con máquina de Moore (1011).

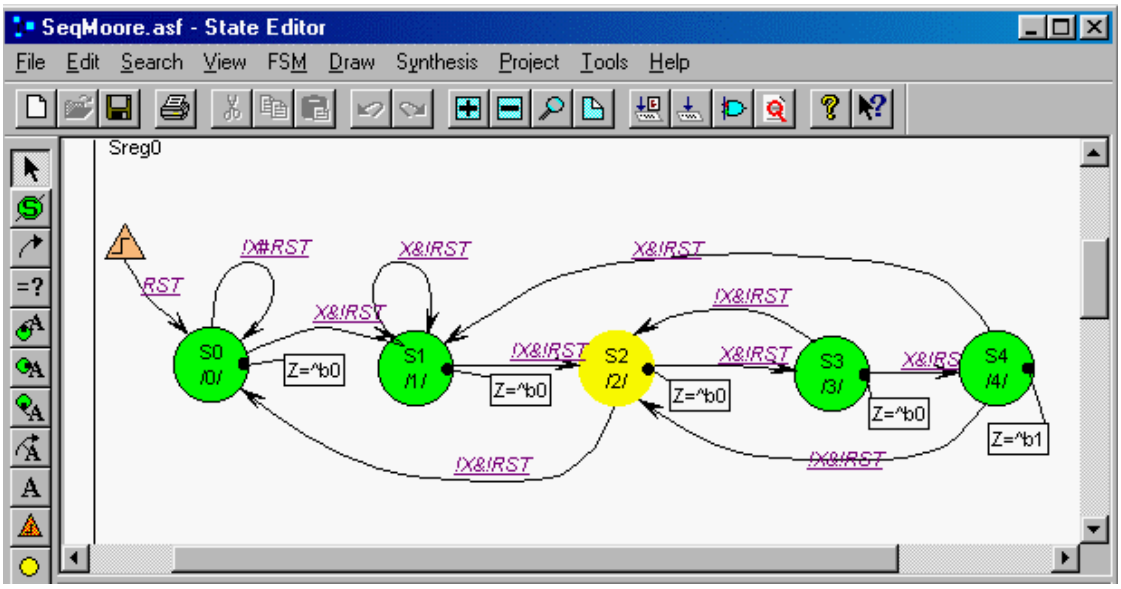



Figura 3.24.- Diagrama de estado durante la simulación mostrando el estado activo.

# 3.6.- Implementación del diseño.

Las herramientas de implementación traducirán el diseño (esquemático, HDL), sintetizarán el circuito para el dispositivo (place and route) y generarán un fichero (bitstream) que puede descargarse en el dispositivo.

## 3.6.1.- Implementación.

Volver al diagrama de flujo del proyecto en la ventana del gestor de proyecto y hacer clic en el botón **IMPLEMENTATION**. La ventana de implementación del diseño se abrirá, lo que nos permitirá especificar el dispositivo destino y su factor de velocidad. El dispositivo destino está ya especificado como 4010XLPC84 puesto que este fue el que seleccionamos cuando el proyecto fue creado. Si se necesita, podemos cambiar el proyecto destino. El nombre de Versión y Revisión se rellena automáticamente con rev1 y ver1. Podemos cambiar el nombre si queremos. Las opciones de implementación nos permiten especificar como se va a optimizar el diseño, mapeo, colocado, rutado y configurado. En general, las opciones por defecto deben ser bastante.

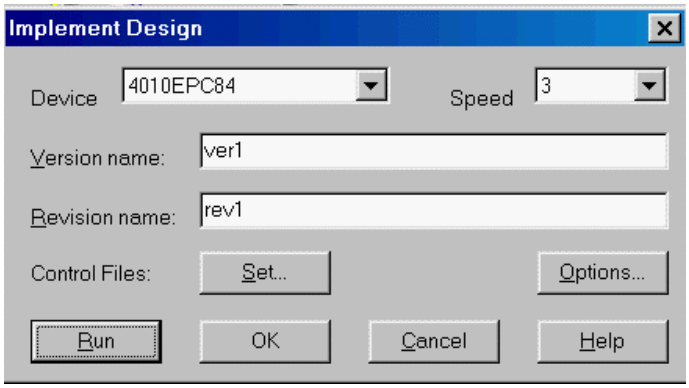



Figura 3.25.- Ventana de Implementación del diseño.

Hacer clic en el boton RUN. La ventana de flujo de sintesis se abrirá y mostrará el progreso a través de los diferentes pasos en el proceso de implementación. La figura siguiente muestra la ventana de flujo de sintesis.

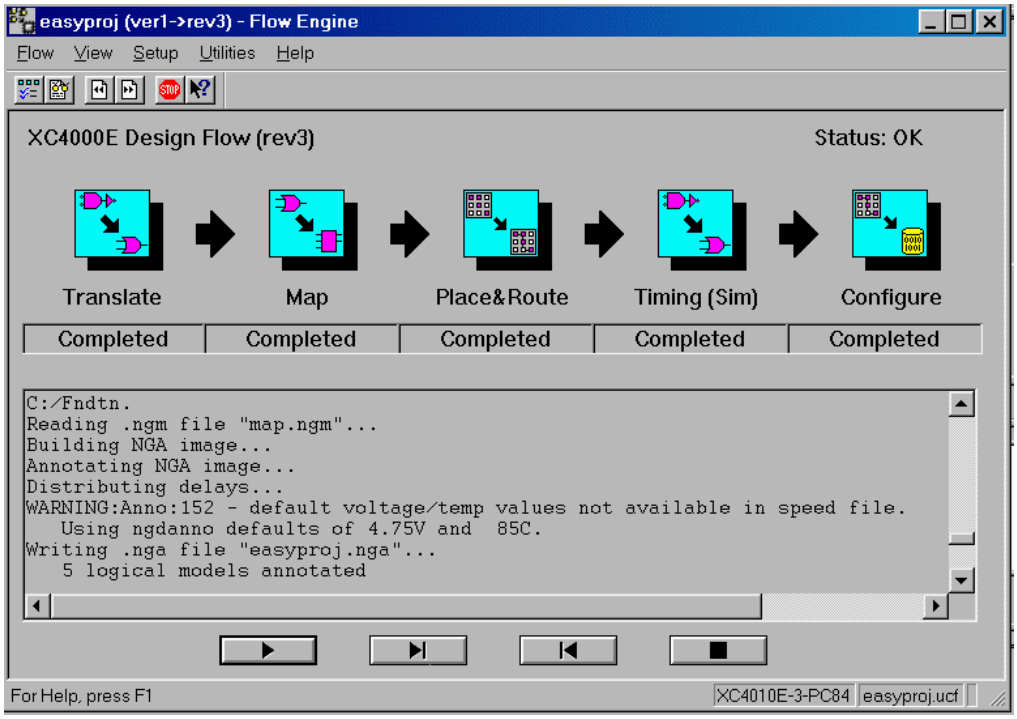



Figura 3.26.- Ventana de Flujo de sintesis.

El primer paso es la traducción del fichero de diseño (fichero EDIF) en un formato adecuado (fichero NGD – Base de datos Genérica Nativa), seguido por un proceso de mapeo del diseño al dispositivo destino especifico. Luego nos encontramos con la colocación y rutado. Luego viene la generación de información de temporización para que la use el simulador temporizado. El paso final es la generación del bitstream. Podemos seguir el estado de cada paso en la ventana de flujo de síntesis.

Cuando la implementación esté completa surgirá una pequeña ventana informándonos de que el flujo de síntesis se ha completado satisfactoriamente. Hacer clic en **OK**. En caso de **que haya errores debemos dirigirnos al fichero Log de implementación**, en el gestor de proyectos (hacer clic en la pestaña Reports sobre el panel de la ventana de la derecha).

### 3.6.2.- Ver los resultados de la implementación.

Podemos ver los informes de la ventana de flujo de síntesis. **Seleccionar UTILITIES -> REPORT BROWSER**. Alternativamente podemos hacer clic sobre el icono del Visor de Informes sobre la barra de herramientas. Esto abrirá la ventana del visor de informes. Veremos los informes de traducción, mapeo, place and route, informes de patillas y otros. Hacer clic en el informe de patillas para ver la asignación de patillas de entrada/salida realizada. Reconoceremos los mismos nombres que los que especificamos en el esquemático (Puerta, Marcha, Cinturon y Zumbador). Comprobar los números de patilla. Otro informe interesante es el informe de mapeo que nos informará si se ha eliminado lógica (como parte de la optimización) o se ha añadido. El informe place and route indica cuantos recursos del dispositivo se han empleado. También dará una estimación del retardo de interconexión promedio.

Podemos ver también los informes desde el gestor de proyectos. Los informes están asociados con cada versión. En el gestor de proyectos, hacer clic en la pestaña Versions en el panel

izquierdo. Seleccionar la versión y revisión de la que queremos ver los informes. Podemos hacer clic en el icono de Visor de Informes sobre la barra de herramientas superior. O podemos hacer clic también sobre la pestaña Reports en el panel derecho. Luego, hacer doble clic sobre los ficheros de Informes de implementación. Esto abrirá la ventana del visor de informes.

### 3.6.3.- Ficheros de restricciones.

#### Asignando patillas con el fichero de restricciones de usuario.

Hay dos tipos de restricciones: (1) localización y (2) temporización. En general las restricciones de localización nos permiten controlar el mapeo y posicionado de elementos lógicos en el dispositivo destino, como la disposición de los pads (patillas E/S). Las restricciones temporales informan al sistema que patillas son criticas y necesitan interconexiones cortas (líneas de alta velocidad). Nos concentraremos en las restricciones de localización. Podemos obtener más información en el menú **HELP -> FOUNDATION HELP CONTENTS**: Entering Constraints.

Si no asignamos números de patilla a las entradas y salidas de nuestro esquemático (o código HDL), debemos hacerlo ahora, antes de compilar el diseño. Si no asignamos las patillas, el compilador las asignara por nosotros. Cuando en el laboratorio usamos las placas XC40 debemos asignar las patillas nosotros para usar los conmutadores y LEDs de la placa de demostración.

Podemos asignar los números de patilla para la FPGA creando un fichero de restricciones de usuario (.ucf). Estos ficheros contienen información sobre localización de patillas y restricciones de tiempo. La forma más fácil de usar el Editor de Restricciones es con una interfaz de usuario gráfica que ejecutamos tras traducir el programa para crear nuevas restricciones en un fichero UCF. Notar que podemos acceder el editor de restricciones solo tras crear una versión del proyecto y de esta forma tras haber realizado la implementación al menos una vez.

Para abrir el editor de restricciones, ir a **TOOLS -> IMPLEMENTATION -> CONSTRAINT EDITOR** desde el gestor de proyectos. La ventana del editor se abrirá. Consiste en una ventana principal y tres ventanas con pestaña, denominadas Gobal, Ports y Advanced. Estamos interesados en la especificación de la disposición de los pads. Hacer clic en la pestaña Ports, lo que abrirá la ventana Ports como se muestra en la figura siguiente. En la parte superior de la ventana podemos ver una lista de nombres de port, que hemos especificado en el esquemático así como la disposición (si especificamos alguna). Para cambiar la disposición, escribir la (nueva) situación como P#. Para nuestro ejemplo, la posición de la patilla debe ser ZUMBADOR (P61), PUERTA (P19), MARCHA (P20) y CINTURON(P23). Cuando terminemos, ir a FILE -> SAVE y cerrar la ventana.

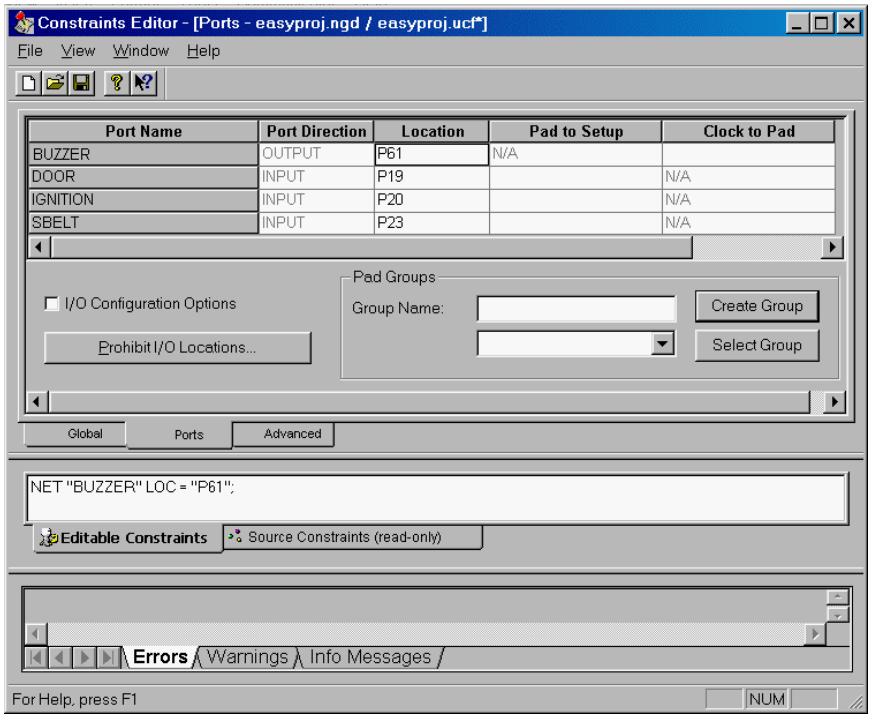



Figura 3.27.- Ventana del Editor de Restricciones.

Para que el sistema use el nuevo fichero de restricciones debemos volver a ejecutar la operación de traducción. Desde el gestor de proyectos, hacer clic sobre el botón implementación. Cuando la implementación sea correcta comprobar el informe de patillas para asegurarnos que las patillas se han colocado de forma correcta como se especificó en el nuevo fichero de restricciones de usuario.

Para más información sobre el editor de restricciones ir al menú **HELP -> HELP TOPICS** en la ventana del editor de restricciones. Esto abrirá la ventana de ayuda del editor de restricciones. Podemos ir también a la ayuda en línea seleccionando **HELP -> ONLINE DOCUMENTS**. Esto nos llevará a la pagina de ayuda de Xilinx.

### 3.6.4.- Creando una nueva versión de diseño o implementación.

Si queremos crear una nueva versión o nueva revisión, ir al gestor de proyectos. Seleccionar **PROJECT -> CREATE VERSION** o **CREATE REVISION**. Esto hará surgir la ventana de Creación de Versión o Revisión. Estas ventanas se parecen a la ventana de implementación discutidas anteriormente. Las versiones aparecerán en la ventana de la izquierda del gestor de proyectos, bajo la pestaña de versión.

### 3.6.5.- Usando el Planificador de conexionado.

Podemos usar el Editor de FPGA para ver como se han usado y colocado los recursos del dispositivo sobre la FPGA. Podemos usar también esta aplicación para colocar y rutar componentes críticos antes de ejecutar las herramientas de colocación y ruteo automáticas sobre el diseño. Para abrir el editor FPGA, ir al gestor de proyecto, y seleccionar **TOOLS -> IMPLEMENTATION -> FPGA EDITOR**.

El editor nos permite mostrar diferentes niveles de detalle sobre la FPGA como hilos cortos, hilos largos, cajas de conmutación, rutado entre componentes, componentes, etc. El nivel de detalle que se muestra se puede controlar haciendo clic en iconos de la barra de herramientas superior. La figura siguiente muestra la ventana del editor de FPGA con una vista del plano de la FPGA, alrededor del área donde se ha colocado el circuito lógico. Las tres cajas azules son los bloques de E/S de las señales PUERTA, MARCHA y CINTURON, y la raja es el bloque lógico. También se muestra la red de conmutación y el rutado de las señales con los bloques lógicos.

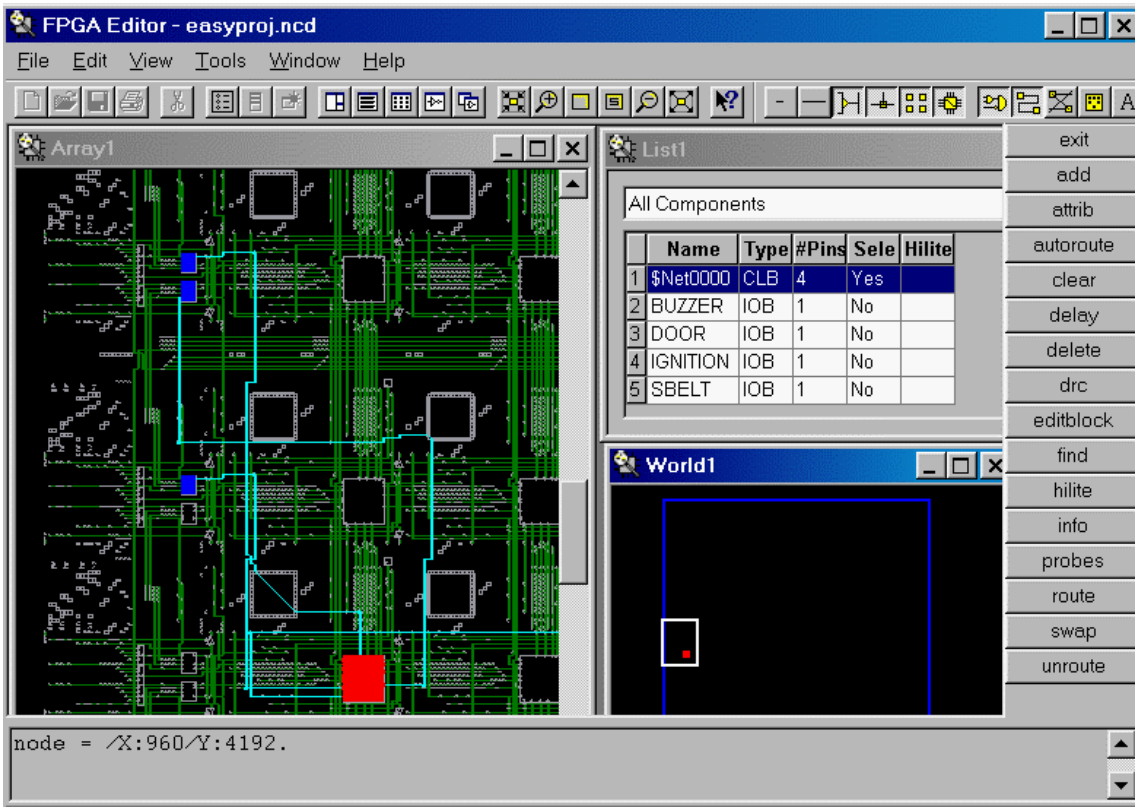



Figura 3.28.- Ventana del Editor de FPGA con la vista de planta del circuito lógico de la figura 3.1.

Para más información sobre el editor FPGA, seleccionar **HELP -> HELP TOPICS** desde la ventana del editor FPGA.

# 3.7.- Configuración del dispositivo (Programación).

Una vez que hemos creado correctamente el bitstream, podemos configurar la FPGA o CPLD. Tenemos varias posibilidades: usar la placa básica de montaje de la FPGA, la placa XC40 o la XC95 de CPLD. Las dos posibilidades que emplearemos serán:

- ? ?**Placa DEMO FPGA**, se emplea el software programador integrado en el sistema Foundation y un programador (Pport III, Xchecker o MultiLINX) conectado al puerto paralelo, puerto serie o puerto USB.
- ? ?**Placa XC40 Xess** o XC95, se emplea un programa proporcionado por el fabricante de la placa y ésta se conecta mediante un cable al puerto paralelo del PC.

## 3.7.1.- Usando la Placa de demostración con el cable Xchecker o programador paralelo.

Este cable se puede usar para programar la FPGA a través del puerto paralelo. Asegurarse que la placa de demostración está alimentada y el cable XChecker está conectado correctamente. Los conmutadores de configuración SW2 (1 a 3) deben conectarse todos a 1, como indica la figura 3.29.

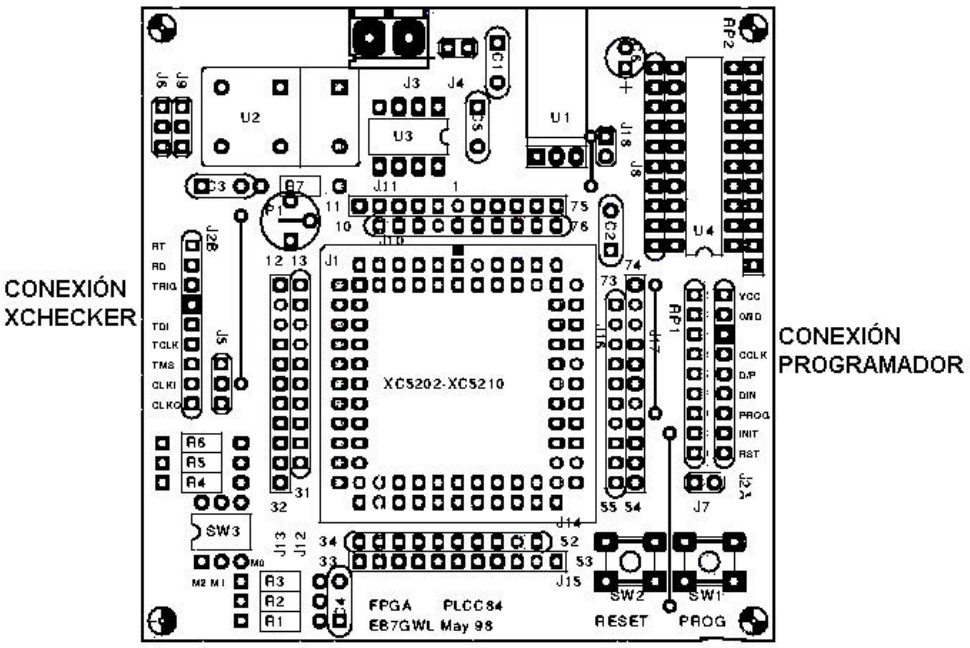



Figura 3.29.- Diagrama de la placa Demo FPGA.

A continuación se da una tabla con las conexiones de los dos visualizadores de 7 segmentos que dispone la placa demo.

| Visualizador Izquierdo | | Visualizador Derecho | |
|---|---|---|---|
| Segmento | Patilla | Segmento | Patilla |
| a | P39 | a | P49 |
| b | P38 | b | P48 |
| c | P36 | c | P47 |
| d | P35 | d | P46 |
| e | P29 | e | P45 |
| f | P40 | f | P50 |
| g | P44 | g | P51 |
| p.decimal | LDC (P37) | p.decimal | INIT (P48) |

Los puntos decimales de los visualizadores de 7 segmentos deben estar encendidos en este momento. A continuación seleccionar en la ventana del gestor de diseño el proyecto que queramos descargar y seleccionar la revisión adecuada (en el panel de la izquierda) como muestra la figura 3.30.

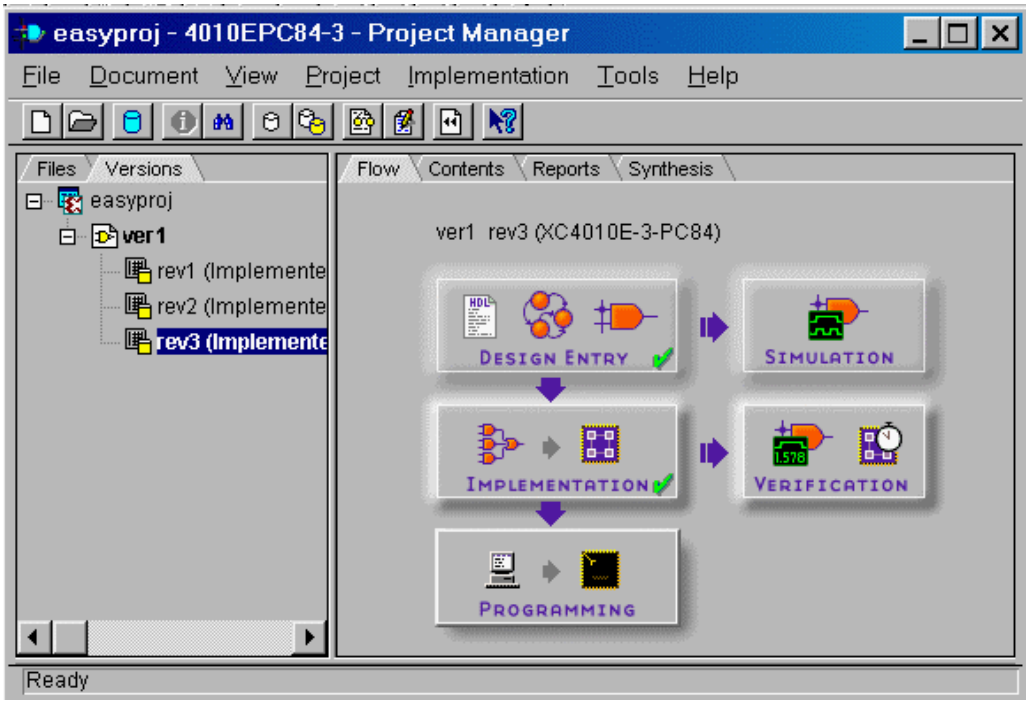



Figura 3.30.- Ventana del gestor de diseño.

Hacer clic en el botón de programación en el panel derecho del gestor de proyecto. Aparecerá una pequeña ventana de selección de programa: seleccionar el botón central, Hardware Debugger (combrobador del hardware). Se abrirá a continuación la ventana del programador, que se muestra en la figura 3.31.

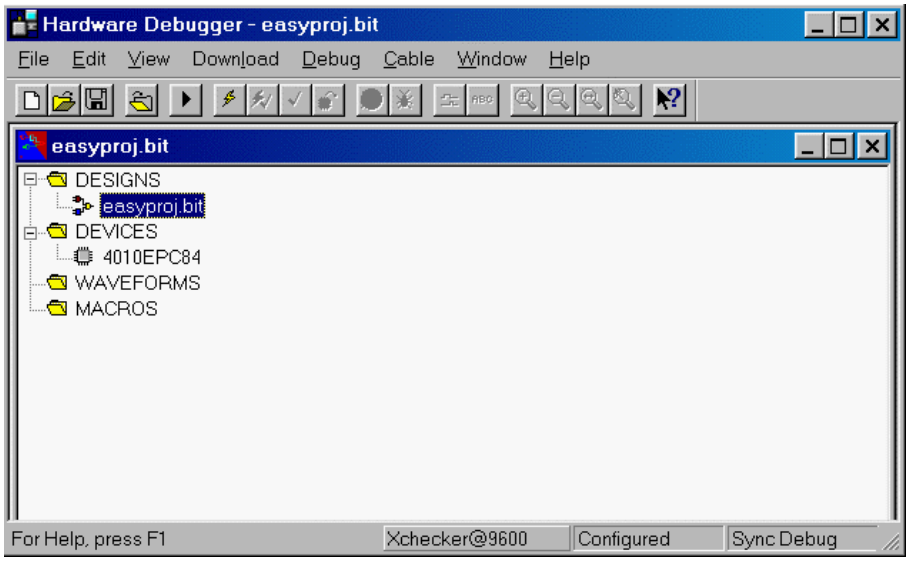


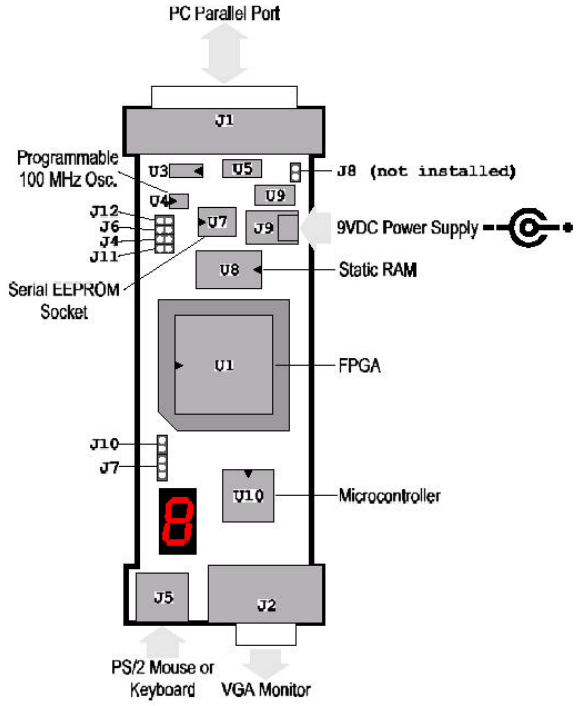

Figura 3.31.- Ventana del comprobador Hardware.

En la ventana de la figura, seleccionar el menú **DOWNLOAD -> DOWNLOAD DESIGN**. Comenzará el proceso de descarga. Durante la descarga el punto decimal del visualizador de 7 segmentos más a la derecha actúa como un indicador de error de programación. Durante la programación el punto decimal del visualizador de la derecha debe parpadear, mientras que el de la izquierda debe estar apagado. Cuando la programación ha sido correcta, los puntos decimales de ambos visualizadores quedan apagados. Si hay un error de programación, el punto decimal de la derecha queda encendido. El programador debe decir "DONE signal went high" y no dará errores si la programación fue correcta. Si sucede un error, comprobar el cable del Programador, la alimentación de la placa, y el estado de los interruptores. En el caso que haya un problema con el cable, ir a **CABLE -> COMMUNICATONS**. En la configuración de comunicaciones seleccionar: **PORT** (escoger LPT 1), Cable Type (Paralell).

Asegurarse también que el diseño ha sido compilado para el tipo correcto de dispositivo. Comprobar que el tipo de dispositivo (ir a la ventana del gestor de proyectos y seleccionar **FILE -> PROJECT TYPE**) se corresponde con el dispositivo de la placa. Si ha sido compilado para otro dispositivo que el XC4003EPC84, escoger el correcto y reimplementar el diseño.

### 3.7.2.- Usando la placa XC40 o XC95.

Estas placas disponen de varios recursos además de las FPGA XC4005XL-PC84 o XC4010XL-PC84 funcionando a 3.3V con tolerancia a 5V. Disponemos de un microprocesador 8031 que se puede comunicar con la FPGA , un generador de reloj programable, un conector VGA, un conector PS/" y 32Kbytes de SRAM de alta velocidad (15ns).




La FPGA y la RAM se pueden programar directamente con un cable conectado al puerto paralelo del PC sin necesidad de programador adicional.

La FPGA olvidará su programación al desconectar la alimentación. Si queremos que la mantenga habrá que programarla en una EEPROM que se inserta en el zócalo de 8 patillas. En la figura 3.31b podemos observar un diagrama de esta placa.

Para programar el dispositivo solo hay que dejar caer el fichero con la extensión bit generado por el proceso de implementación. Hecho esto tendrá lugar la programación inmediata del dispositivo, quedando los ficheros empleados en la lista con la posibilidad de poder recargar el diseño en el dispositivo pulsando el botón Reload.

Figura 3.31b. Diagrama de la placa XESS XC40 empleada en las prácticas.

Para realizar la programación de estas placas se emplea el programa gsxload. Una vez arrancado aparece la ventana de la figura 3.32.

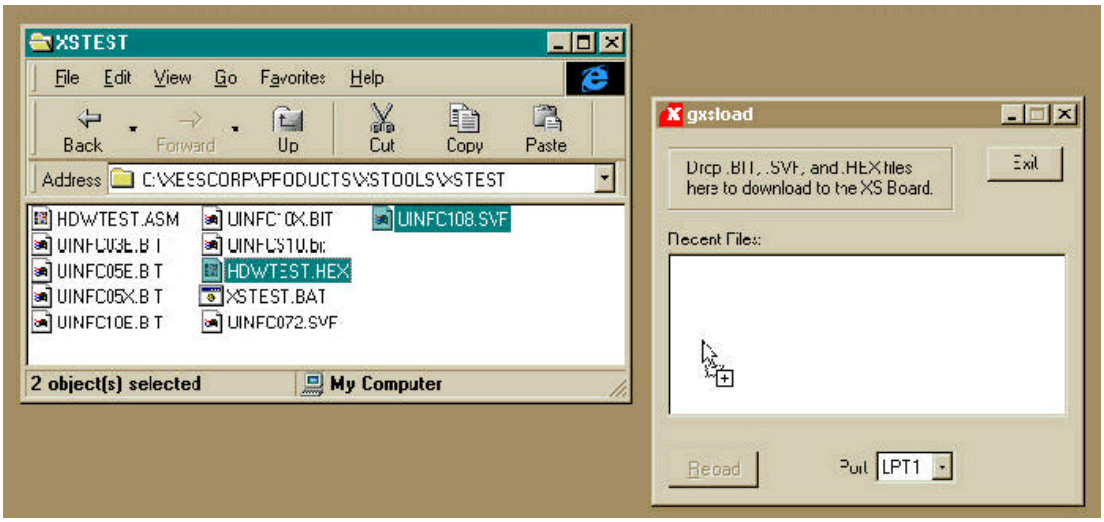



Figura 3.32.- Herramienta de programación de las placas XC40 y XC95.

Una vez programada la placa se tendrá acceso a 8 señales de entrada y 4 de salida mediante el software WinXSPort, además de poder encender un visualizador de 7 segmentos que dispone la placa, según la siguiente tabla para la placa XC40:

| Patilla | Etiqueta | Lógica | Bit en WinXSport |
|---|---|---|---|
| 34 (MD2) | LOC=MD2 | Normal | D7 |
| 32 (MD0) | LOC=MD0 | Normal | D6 |
| 49 | LOC=P49 | Normal | D5 |
| 48 | LOC=P48 | Normal | D4 |
| 47 | LOC=P47 | Normal | D3 |
| 46 | LOC=P46 | Normal | D2 |
| 45 | LOC=P45 | Negada | D1 |
| 44 | LOC=P44 | Negada | D0 |
| 75 (TDO) | LOC=TDO | Negada | S7 |
| 69 | LOC=P69 | Normal | S6 |
| 66 | LOC=P66 | Normal | S5 |
| 77 | LOC=P77 | Normal | S4 |
| 70 | LOC=P70 | Normal | S3 |
| 15 (TDI) | LOC=TDI | Normal | C3 |
| 17 (TMS) | LOC=TMS | Normal | C2 |
| 16 (TCK) | LOC=TCK | Negada | C1 |
| 19 | LOC=P19 | Normal | Segmento a |
| 23 | LOC=P23 | Normal | Segmento b |
| 26 | LOC=P26 | Normal | Segmento c |
| 25 | LOC=P25 | Normal | Segmento d |
| 24 | LOC=P24 | Normal | Segmento e |
| 18 | LOC=P18 | Normal | Segmento f |
| 20 | LOC=P20 | Normal | Segmento g |

Seleccionar del menú windows la opción setup y aparecerá la ventana de la figura 3.33.

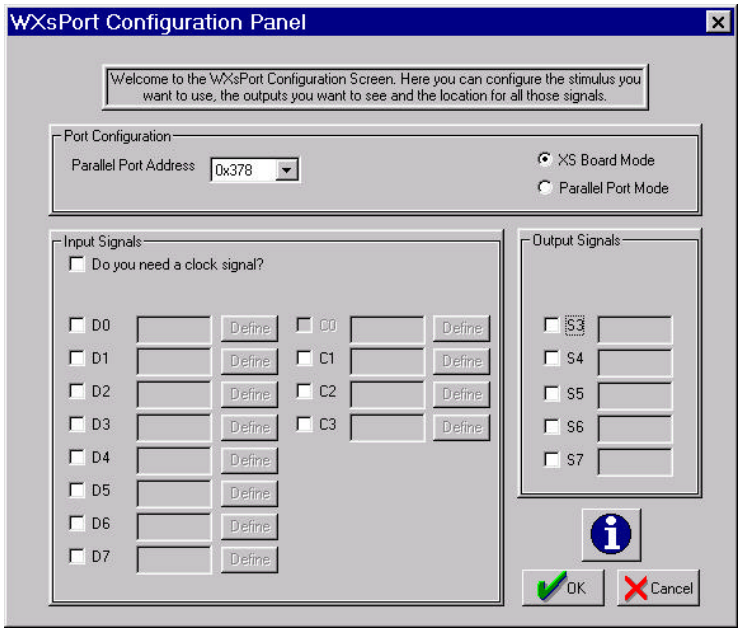



Figura 3.33.- Ventana de control de Entradas/Salidas de la FPGA.

La ventana contiene varios controles que realizan las siguientes funciones:

? ?La entrada **Parallel Port Adress** permite seleccionar el puerto paralelo donde esta conectada la placa, siempre usaremos 0x378.

? ?Tenemos ocho botones de entrada, desde D0 a D7, cada uno de los cuales está asociado con una entrada de la FPGA según la tabla anterior. Pulsando en define le damos un nombre a cada entrada de la FPGA que hayamos usado. Estos nombres aparecerán en el cronograma.

? ?Tenemos cinco botones de salida, desde C0 a C4, cada uno de los cuales está asociado con una salida de la FPGA según la tabla anterior.

? ?Si necesitamos una señal de reloj, está será D0 y se conectará al marcar la casilla "Do you need a clock signal?".

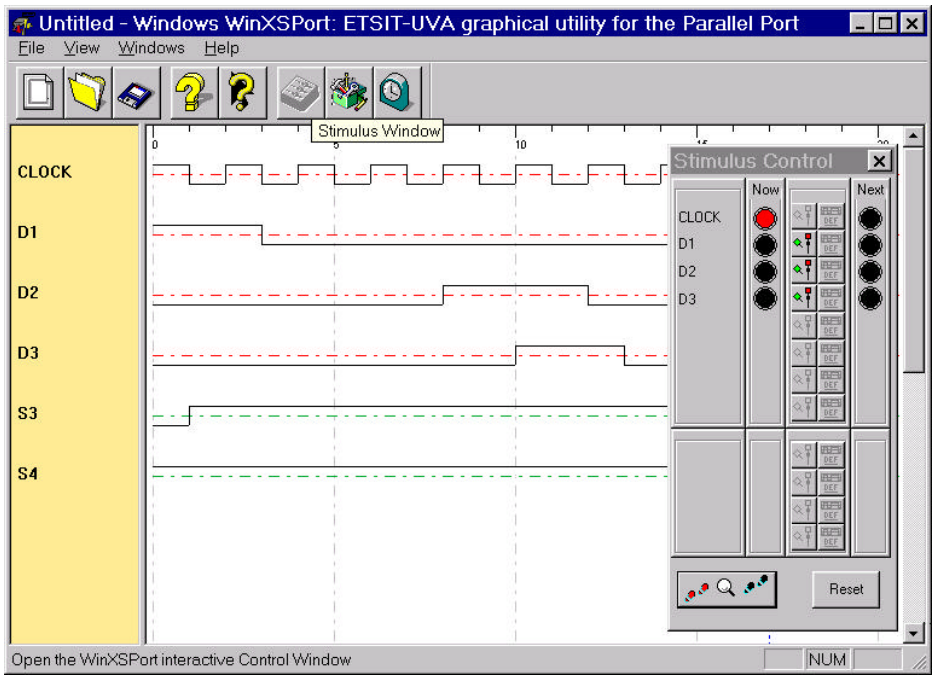



Figura 3.34.- Ventana del analizador lógico WinXSPort.

Emplear la ventana Stimulus Control para proporcionar señales de entrada a la placa y ver las salidas que presenta.

## 3.8.- Macros y Esquemáticos Jerárquicos.

Siempre es buena práctica mantener un diseño modular y jerárquico. Esto es muy importante para diseños de complejidad moderada y alta. A menudo, usaremos un circuito (consistente en diferentes puertas lógicas) una y otra vez. En lugar de dibujar estas puertas lógicas cada vez que las necesitemos, sería más eficiente convertirlas en una celda o macro con su propio símbolo lógico. Luego podremos usar esta macro cada vez que la necesitemos añadiendo a nuestro esquemático. También, el uso de macros mantiene el diseño más limpio y más fácil de entender. Como veremos a continuación, una macro no tiene que ser un esquemático, también puede venir definida por una descripción en lenguaje ABEL-HDL, VHDL o Verilog.

En este capitulo veremos brevemente como crear una macro a partir de un esquemático, y con ABEL-HDL, y como usar las macros para construir un esquemático de alto nivel. Por ejemplo, un comparador de 4 bits que compara dos palabras de 4 bits y da una salida a 1 si las dos palabras son idénticas, en otro caso un nivel lógico 0. Sin embargo, para mantener más general el diseño del comparador, las entradas X pueden ser seleccionadas de dos palabras diferentes (A o B) usando un multiplexor 2:1. Si las señales X (A o B) e Y son las mismas, el comparador genera un 1, en otro caso un 0 lógico. La figura 3.35 muestra el esquemático completo. Cada comparador compara un bit de cada palabra. Si los cuatro comparadores dan Zi igual a 1, los cuatro bits correspondientes de cada palabra son idénticos y la puerta AND dará un 1 lógico.

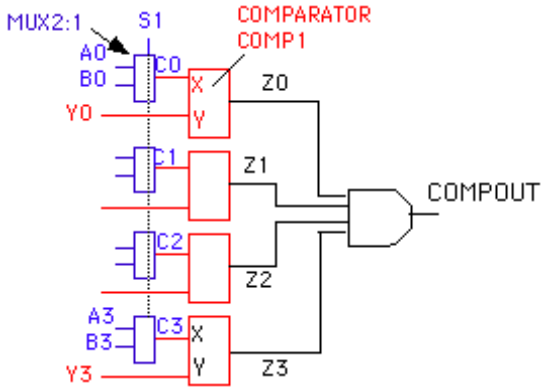



Figura 3.35.- Comparador de dos palabras de 4 bits.

Construiremos este circuito de forma modular, usando macros. Primero, crearemos una macro para un comparador de 1 bit usando la herramienta de entrada de esquemáticos. Llamaremos a esta macro "COMP1ME". Luego, construiremos una macro para el multiplexor 2:1 usando ABEL-HDL. Una vez se hayan creado estas dos macros construiremos el esquemático del circuito completo (esquemático de alto nivel).

Las macros definidas por el usuario se pueden crear de varias formas. Un método conveniente es usar el asistente de símbolos (Symbol Wizard) para crear macros basadas tanto en esquemáticos como en HDL. Con el asistente de símbolos creamos primero el símbolo y el esquemático o HDL subyacente después.

Una forma alternativa de crear macros es crear la fuente subyacente primero, bien sea el esquemático o el fichero HDL. Si el fichero fuente es un esquemático, ir al menú **HIERARCHY -> CREATE A MACRO SYMBOL** a partir del esquema actual en el editor de esquemáticos. En el caso que se trate de un fichero HDL, ir al menú **PROJECT -> CREATE MACRO** en el

Editor HDL. Si la macro es un diagrama de máquinas de estado, ir al menú **PROJECT -> CREATE MACRO**. Se creará un símbolo y se colocará en la librería del proyecto en los tres casos. Ilustraremos estos métodos con el siguiente ejemplo.

### 3.8.1.- Creación de una Macro para el comparador de 1 bit usando el editor de esquemáticos.

La expresión boleana para el comparador de 1 bit viene dada por:

$$Z = X.Y + X'.Y' = X \,\&\, Y \,\#\, !X \,\&\, !Y$$

El esquemático correspondiente se muestra en la figura 3.25. Observemos que el comparador de 1 bit es realmente una puerta XNOR. En principio, podemos usar la puerta XNOR para construir el comparador de 4 bits. Sin embargo, para ilustrar el uso de macros, lo haremos usando puertas AND e inversores como muestra la figura 3.36.

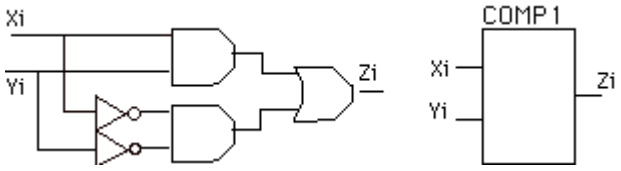



Figura 3.36.- Esquemático y símbolo del comparador de 1 bit.

Crearemos primero un nuevo proyecto, llamado "COMP4ME" (ir al ménu **FILE -> CREATE NEW PROJECT** en la ventana del gestor de proyectos). Rellenar el nombre de proyecto, directorio, tipo, familia, dispositivo y velocidad como lo hicimos en el capítulo 3.3.

### 3.8.2.- Usando el Asistente de Símbolos.

Abrir el Editor de Esquemáticos haciendo clic en el icono Schematic Entry en la ventana del gestor de proyecto. Usaremos el asistente de Símbolos, en la ventana del editor de esquemáticos, ir al menú **TOOLS -> SYMBOL WIZARD**. Un asistente nos guiará haciéndonos varias preguntas.

- Para el nombre del símbolo (Symbol Name), entrar el nombre que queremos darle al comparador de 1 bit. Llamémosle "COMP1ME".
- Seleccionar Schematic por Contents.
- Luego pulsar el botón NEXT. Rellenar el nombre de los puertos de entrada y salida. Para salida, pulsar el botón ADVANCED, y seleccionar "com" para indicar que la salida es de un circuito combinacional (si la salida proviene de un Flip-Flop deberíamos seleccionar "reg"). Cuando todas las entradas y salidas se han definido, pulsar el botón NEXT. Seguir las instrucciones.
- Cuando terminemos pulsar el botón FINISH.

El símbolo se creará a continuación y se añadirá a la librería del proyecto. También se creará un fichero de diseño que tendrá definido los nombres de puerto. El editor de esquemático se abrirá mostrando una página de esquemático vacía para la macro del comparador, excepto por las entradas y la salida.

Sobre el esquemático de la macro, podemos dibujar ahora las puertas lógicas (Figura 3.25) de la macro COMP1ME. La creación del esquemático es básicamente la misma que la explicada anteriormente. Sin embargo, hay dos diferencias.

1. No necesitamos añadir buffers de entrada o salida (IBUF o OBUF) puesto que las entradas y salidas de la celda no irán a las salidas de la FPGA o CPLD todavía.
2. No usamos IPADs o OPADs para los puertos de entrada y salida. Usamos terminales de E/S en su lugar para indicar la entradas y salidas. Realmente, si usamos el asistente, estos terminales ya se han añadido al esquemático. En el caso que creemos la macro a partir de un esquemático existente podemos añadir un terminal de E/S haciendo clic en el icono terminal E/S (conector jerárquico) a la izquierda de la barra de herramientas sobre la ventana SC Symbols. Se abrirá a continuación una ventana donde rellenaremos el nombre del terminal (p.ej. X, Y o Z) e indicaremos si se trata de entrada o salida. Tener cuidado de asignar la dirección correcta para entradas y salidas. La figura 3.37 muestra el esquemático completado del comparador de 1 bit.

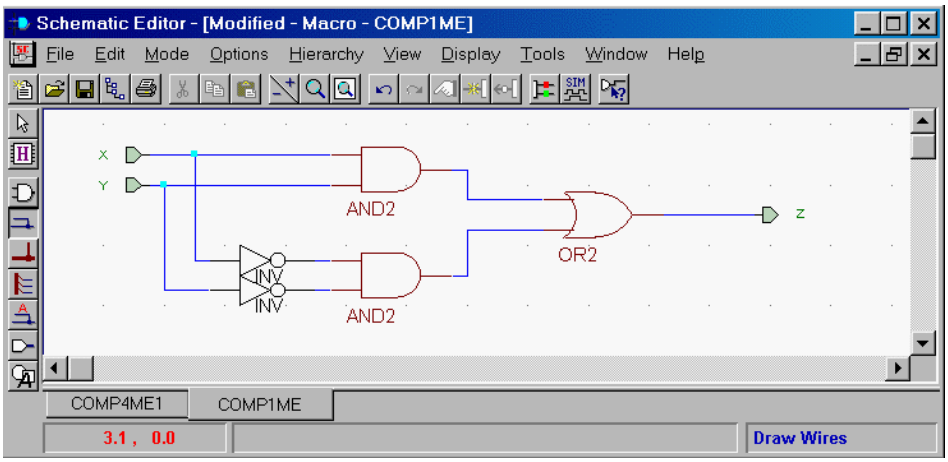



Figura 3.37.- Esquemático del comparador de 1 bit usado como macro.

Cuando terminemos guardar el esquemático. Para comprobar si la macro se ha creado, en el editor de esquemáticos ir a **FILE -> OPEN -> MACRO**. Debe aparecer la macro COMP1ME. En este momento podemos simular la macro.

Para simular la macro, podemos colocarla en el esquemático principal. Ir al simulador haciendo clic en el icono SIM en la barra de herramientas.

- En el visor de cronogramas, hacer clic en el icono Add Signal (o ir a SIGNALS -> ADD SIGNALS). En el panel central de la ventana de selección de componente, denominado Chip Selection veremos el nombre de la macro, COMP1ME. Hacer doble clic en él. En el panel derecho, Scan Hierarchy, se mostrarán todas las patillas de entrada y salida. Seleccionar estas patillas haciendo doble clic. Cuando este hecho, hacer clic en el botón CLOSE.
- Abrir la ventana del simulador y definir los estímulos para las patillas de entrada usando el contador binario. Ahora podemos realizar una simulación funcional y verificar su funcionamiento.

### 3.8.3.- Usando un esquemático existente como macro.

En caso que hayamos creado un esquemático como parte de otro proyecto y queramos usarlo como macro en un nuevo proyecto podemos hacer esto.

- Primero tendremos que añadir el antiguo esquemático al proyecto nuevo. En la ventana del gestor de proyectos, ir al menú **FILE -> ADD DOCUMENT**. Usar el panel para localizar el esquemático de interés. Cuando esté hecho, veremos que el esquemático se ha añadido al proyecto en la página de la izquierda (pestaña Files).
- Podemos necesitar modificar este esquemático, esto es cambiar los nombres de patillas, reemplazar PADS de entrada (IPADS) por terminales E/S y eliminar buffers (BUF). Recordar que para macros no hay que usar IPADs o OPADs (estos indican las patillas físicas del dispositivo) sino terminales E/S.
- Luego necesitamos crear una macro a partir del esquemático. Esto se realiza como sigue: ir al menú **HIERARCHY -> CREATE MACRO SYMBOL FROM CURRENT SHEET**. Se abrirá una ventana. Podemos cambiar el nombre de la macro, y añadir una breve descripción en la línea de comentario. También podemos comprobar las señales de entrada y salida. Hacer clic en OK.

### 3.8.4.- Crear un esquemático primero.

Una forma alternativa de crear la macro, es crear los esquemáticos primero y luego el símbolo. En este caso abriremos una nueva hoja y dibujaremos el esquemático. Para los puertos de entrada y salida necesitamos usar terminales E/S y no IPAD, OPAD, IBUF y OBUF como se explico antes. Cuando terminemos con el esquemático de la macro, necesitamos crear el símbolo. Esto se realiza de la misma forma que antes. Ir al menú **HIERARCHY -> CREATE MACRO SYMBOL FROM CURRENT SHEET**. Aparece la ventana de creación de símbolo. Para el nombre de símbolo, entrar el nombre que queramos darle. Lo llamaremos "COMP1ME". En el campo de comentario, teclear lo que queramos. Hacer clic en OK. El símbolo se añadirá a la lista de símbolo SC Symbols. Ahora podemos usar la macro creada (símbolo) como haríamos con cualquier otro símbolo. Podemos completar el esquemático de alto nivel en este momento, o crear otro macro.

### 3.8.5.- Crear una macro con ABEL HDL.

Aquí demostraremos como crear un nuevo símbolo (o macro) para un multiplexor 2:1 con ABEL. El símbolo para el multiplexor se muestra en la figura 3.38. El multiplexor selecciona entre dos palabras de cuatro bits: cuando S1 es 0, aparece la palabra A en la salida; cuando S1 es 1, la palabra B se conecta a la salida. Podemos considerar este multiplexor como cuádruple 2:1. Crearemos un una macro ABEL para este multiplexor denominada "MUX4X2ME".

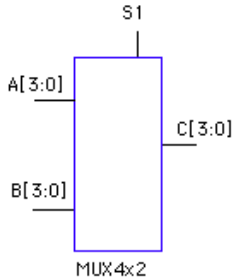



Figura 3.38.- Símbolo para el multiplexor cuadruple 2:1.

Hola como estarmo Asumamos que ya hemos creado un proyecto denominado COMP4ME del cual la macro para el MUX formará parte.

1- Abrir el proyecto: COMP4ME y abrir el editor de esquemático.
2- Crear el símbolo nuevo:
   - ? ? Seleccionar el menú TOOLS -> SYMBOL WIZARD y hacer clic en NEXT.
   - ? ? En la ventana de contenido del asistente, entrar el nombre del símbolo MUX4X2ME. El nombre de la macro ABEL no debe tener más de 8 caracteres. El proceso de síntesis dará un error sin decir que causo el error.
   - ? ? En la ventana contenidos, seleccionar el botón HDL y el botón lenguaje ABEL. Luego hacer clic en NEXT.
   - ? ? En la ventana de puertos del asistente, hacer clic en NEW para escribir el nombre de las entradas y las salidas. Entrar S1 en el campo de nombre y seleccionar Input en el campo direccion. Hacer clic en NEW y entrar A[3:0] en el campo nombre; seleccionar Input en la sección de dirección. Hacer lo mismo para las entradas B[3:0]. Luego, hacer clic en NEW de nuevo y entrar C[3:0] en el campo nombre y seleccionar Output para la dirección. Ir al botón ADVANCED y seleccionar puerto combinacional. Cuando terminemos de definir entradas y salidas, pulsar el botón NEXT.
   - ? ? Hacer clic en el botón FINISH. El nuevo símbolo se colocará en la librería del proyecto.

Podemos colocar ahora el MUX4X2ME en la nueva página del esquemático (COMP4ME). En caso de que necesitemos modificar el símbolo, como cambiar su forma, añadir, renombrar o eliminar patillas E/S, podemos hacerlo. En la ventana del editor de esquemáticos, seleccionar el menú **TOOLS -> SYMBOL EDITOR**. O también podemos hacer clic en el símbolo de macro del esquemático. Se abrirá la ventana de propiedades del símbolo. Hacer clic en Symbol Editor.

3- Crear una macro ABEL.
   - ? ? Tenemos que definir todavía la nueva macro para la que hemos creado el símbolo MUX4X2ME.
   - ? ? Tras colocar el símbolo en el esquemático hacer clic en el icono "H" (Jerarquia).
   - ? ? Luego, hacer doble clic sobre el símbolo MUX4X2ME. El editor ABEL abrirá su ventana de edición.
   - ? ? Notaremos que las patillas ya han sido declaradas. Asegurarse que las patillas de salida C3..C0 han sido declaradas como istype 'com' y no 'reg' porque la C es la salida de un circuito combinacional.
   - ? ? En la sección de ecuaciones, teclear la expresión para el multiplexor 2 a 1. La expresión de la salida del multiplexor es como sigue, en función de la tabla de verdad siguiente. Tener cuidado, los nombres de entrada y salida son sensibles a mayusculas.

**C = !S1 & A # S1 & B;**

| S1 | C |
|---|---|
| 0 | A |
| 1 | B |

Para simplificar la expresión podemos hacer uso de los conjuntos en la descripción ABEL (A=[A3..A0], etc). Cada miembro de un conjunto hereda las mismas operaciones lógicas. De esta forma la expresión de arriba es equivalente a las cuatro expresiones individuales:

```
C0 =!S1 & A0 # S1 & B0;
:
C1 =!S1 & A3 # S1 & B3;
```

La figura 3.39 muestra la ventana del editor ABEL. Notar que los conjuntos para A, B y C ya se encuentran definidos. Entras en la sección de ecuaciones la expresión para C.

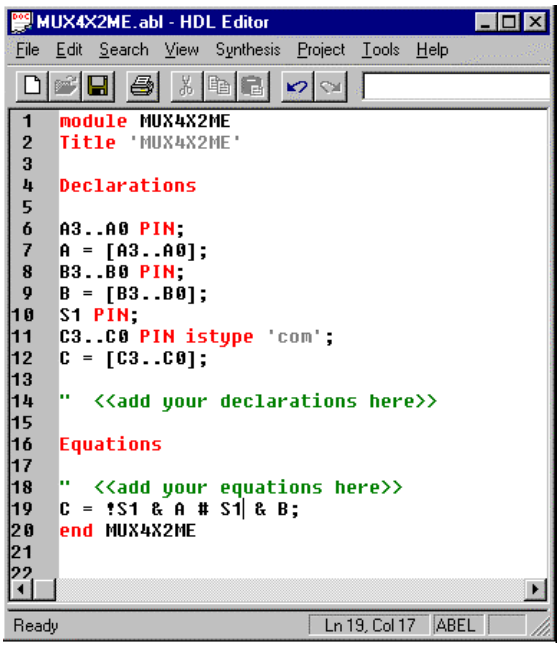



Figura 3.39.- Ventana del editor ABEL para el cuadruple multiplexor 2 a 1.

? ? Cuando este terminado, comprobar la sintaxis (ir al menú **SYNTHESIS -> CHECK SYNTAX**).

? ? Salir. Cuando se pida actualizar la macro, seleccionar YES. Ahora tendrá lugar la síntesis de la macro.

Hay una forma alternativa de crear una macro ABEL. En este caso no necesitamos abrir un esquemático. Asumamos que ya hemos creado un proyecto para el que queremos crear una macro.

? ? En la ventana del gestor de proyectos, hacer clic en el icono HDL. Escoger design Wizard.

? ? Seguir el mismo procedimiento explicado antes (Entrada de diseño con ABEL).

? ? Ir al menú **PROJECT -> Make MACRO**. Darle un nombre a la macro. Cuando se termine, el símbolo de la macro se añadirá a la librería de proyectos.

# 3.9.- Creación de un esquemático de alto nivel: comparador de 4 bits.

Ahora estamos listos para diseñar nuestro comparador de 4 bits usando las macros para el comparador de 1 bit y el cuadruple multiplexor 2 a 1. Ir a la ventana del capturados de esquematico y abrir una nueva hoja (a no ser que tengamos ya una abierta).

## 3.9.1.- Añadir los símbolos lógicos y macro.

Añadamos celdas de comparador de 1 bit haciendo clic sobre el icono de símbolo de la barra de herramientas de la izquierda. Teclear comp para ir al símbolo COMP1ME. Seleccionar COMP1ME y colocarlo en el esquemático. Luego, hacer clic sobre el símbolo para colocar multiples copias. Luego añadir el multiplexor cuadruple 2 a 1. Añadir luego la puerta AND de 4 entradas (AND4). Si el nombre de símbolo de COMP1 no se muestra y queremos verlo, seleccionar el símbolo y hacer doble clic. En la ventana de propiedades del símbolo pulsar la clave Attributes y seleccionar “Show Symbol Name”.

Asumamos que queremos implementar el comparador en una FPGA. Necesitaremos indicar las patillas de entrada y salida (a no ser que queramos hacer otra macro para el comparador de 4 bits; en este caso seguiriamos el mismo procedimiento que antes). Primero añadamos los buffers de entrada. Seleccionar IBUF4 en la ventana SC Symbol y colocarlo en el esquemático. Hacer esto para cada entrada A, B e Y (ver figura 3.29). Para la señal de entrada S1, colocar un IBUF simple. Añadir luego OBUF. Luego, colocar las patillas de entrada (IPAD4) y patilla de salida (OPAD) para indicar la conexión con la patilla real de la FPGA.

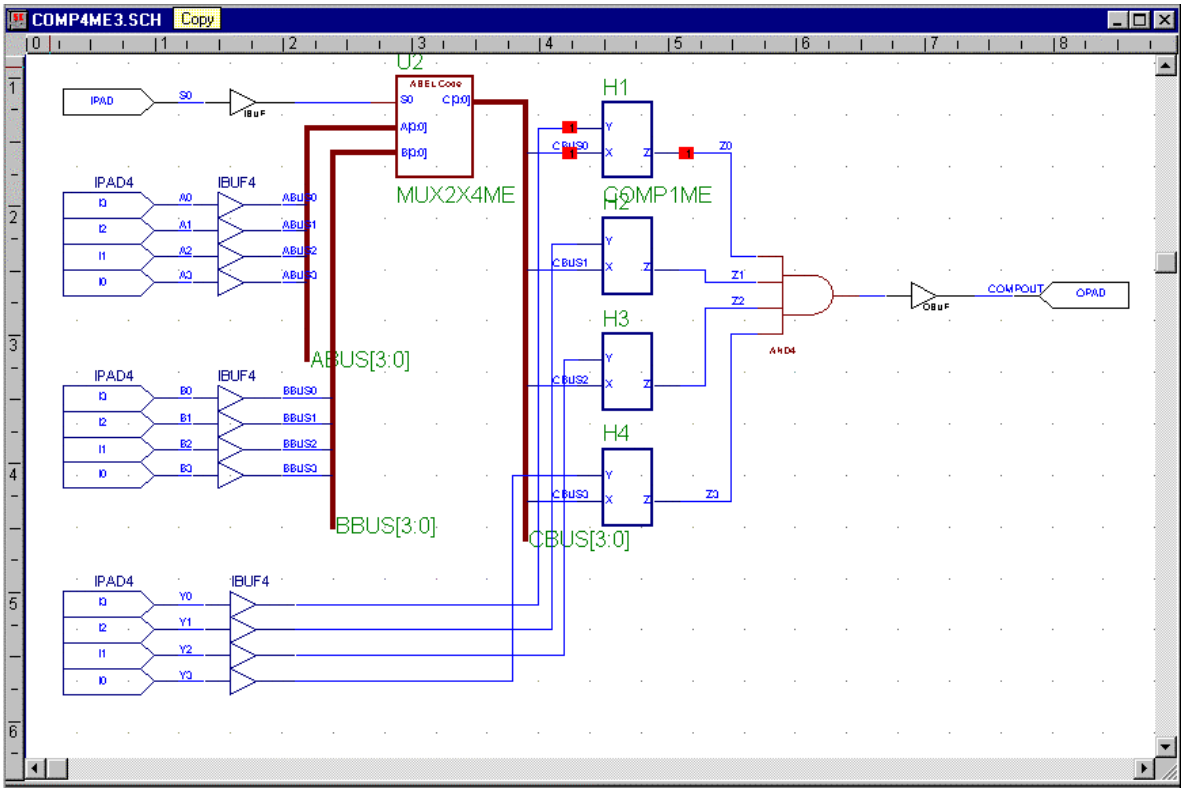



Figura 3.40.- Esquemático de alto nivel del comparador de 4 bits

### 3.9.2.- Añadir hilos y nombres de conexión.

Conectar los IPADs al IBUF y el OBUF al OPAD usando el comando de colocar hilo. Conectar también la salida de la puerta AND a la entrada del OBUF y la salida de los cuatro comparadores de 1 bit a las entradas de la puerta NAND de 4 entradas.

Ahora, queremos etiquetar la entrada, conexiones de salida y las salidas del comparador. Seleccionar el icono de nombres de net en la barra de herramientas izquierda. La ventana de nombre de net se mostrará. Entrar el nombre de net A0. Para acelerar el nombrado de conexiones, seleccionar el botón REPEAT. Cada vez que hagamos clic en una conexión, se colocará un nuevo nombre y el indice se incrementará en 1. Pulsar la tecla ESC para finalizar el comando. Hacer lo mismo con las entradas B y las salidas Z de los comparadores de 1 bit. No olvidar nombrar la conexión de entrada S1. A continuación, usaremos el comando de dibujo de buses para conectar las entradas.

### 3.9.4.- Dibujando los buses.

Para conectar las salidas de IBUF4 a las entradas del multiplexor A[3:0] y B[3:0] usaremos el comando dibujar bus. Hacer clic en el icono de dibujado de buses o ir al menú MODE -> DRAW BUSES. Hacer clic en la entrada del MUX A[3:0] y dibujar el bus como muestra la figura 3.29. Hacer clic para colocar un extremo del bus. Hacer clic con el botón derecho o doble clic para finalizar el bus. Aparecerá una ventana para editar etiqueta asociada al bus como muestra la figura 3.30. Si no aparece, hacer doble clic al final del bus. En la ventana entrar en el campo nombre: ABUS. Seleccionar un ancho de bus de 3 a 0. Este bus no es un bus de entrada ni de salida, por lo que el marcador de E/S debe decir ninguno (none). Hacer clic en OK. Hacer lo mismo para el bus B y el bus C y llamar a este bus BBUS y CBUS, respectivamente.

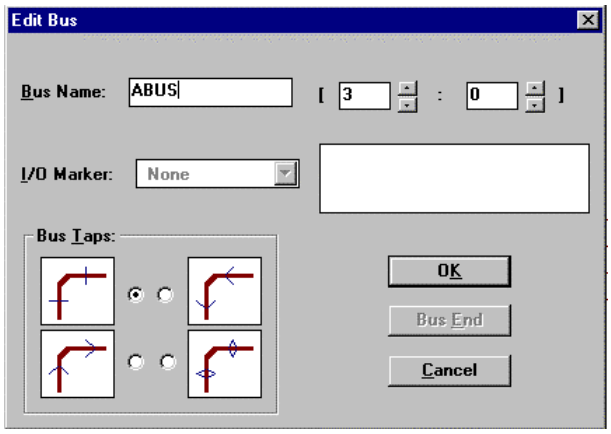



Figura 3.41.- Ventana de edición de Bus.

### 3.9.5.- Enlaces al bus.

Ahora podemos conectar las entradas de los comparadores de 1 bit a los buses. Esto se realiza haciendo clic en el icono de dibujo de conexiones al bus o yendo al menú **MODE -> DRAW BUS TAPS**. Hacer clic en el nombre de bus ABUS para seleccionar el bus al que queremos conectarnos. La linea de estado mostrará que la primera conexión al bus ABUS0 (normalmente

el miembro con menor indice) ha sido seleccionada para colocarla. Hacer clic con el cursor en la salida A0 del primer buffer de entrada (IBUF4) (bit menos significativo). Hacer clic en las salidas A1 del 2º, 3er y 4º buffer. Cuando terminemos pulsar la tecla ESC dos veces. Repetir el proceso para el BBUS y CBUS como se muestra en la figura 3.40.

El número de comienzo por defecto de la conexión es 0. Podemos usar las teclar arriba y abajo para seleccinar que bit del bus se enlazará. Si la tecla abajo se pulso la última, los numeros de conexión decrecerán.

Si queremos conectar un hilo simple a un bus, necesitamos dibujar un hilo que comience o termine en el bus. Luego necesitamos etiquetarlo con el nombre de un miembro del bus al que queremos conectarlo. Por ejemplo, queremos acceder al bit más significativo en el bus de conteo (CBUS). Necesitamos nombrar al hilo como CBUS3. Si no lo nombramos no habrá ninguna conexión (se generará un aviso) o si le damos un nombre diferente del bus, se producirá un error.

### 3.9.6.- Buses complejos.

Para simplificar un esquemático, podemos combinar buses y conexiones individuales en un bus complejo. Esto se hace nombrando al bus complejo con una etiqueta compuesta. Una etiqueta compuesta consiste en varios nombres de etiqueta (con o sin rangos), separadas por comas. Para introducir una etiqueta compuesta, usar el siguiente formato:

NOMBRE[F:L:I], NOMBRE[F:L]

donde F es el primer número, L es el último número, e I es el intervalo entre los números de un rango. Si no se especifica intervalo, el intervalo es 1. Como ejemplo podemos combinar los buses LED-OUT y las señales de error en un bus complejo con el nombre: LED_ONE[6:0], LED_TWO[6:0],ERROR, o podemos usar un número de hilos seleccionados: LED_ONE[3:0], LED_TWO[4:2],ERROR.

Es importante que cada uno de los segmentos del bus, como LED_ONE[6:0] hayan sido definidos con anterioridad. Para más información sobre buses, seleccionar en la ventana de entrada de esquemáticos **HELP -> SCHEMATIC EDITOR HELP CONTENTS**. En la ventana de ayuda seleccionar INDEX y teclear buses. La ayuda vendrá en Inglés.

### 3.9.7.- Terminales de E/S frente a PADS.

En caso que decidamos usar este esquemático como otra macro y no pretendamos implementar este esquemático particular en una FPGA o CPLD, no necesitaremos añadir PADS y BUFFERS. En este caso, cuando dibujemos los buses de entrada para conectar las entradas A y B, indicaremos en la ventana Add Terminal/Label que el bus es una entrada (o salida). Este nombre de bus será ahora el nombre de la entrada a la macro. Para crear una macro a partir del esquemático ir al menú **HIERARCHY -> CREATE MACRO SYMBOL FROM CURRENT SHEET**. Cuando el esquematico este terminado, guardarlo.

### 3.9.8.- Descendiendo por la jerarquia.

Para entrar en el esquemático de una macro, seleccionar el icono "H" (icono de jerarquia) y hacer doble clic sobre el símbolo de macro deseado. Para volver al esquemático de mayor nivel, hacer doble clic sobre el fondo del esquemático de menor nivel.

### 3.9.10.- Prueba de integridad y Netlist.

Cuando el esquemático se haya terminado podemos crear un netlist y realizar una prueba de integridad. Esto se realiza yendo al menú **OPTIONS -> CREATE NETLIST**. Cuando finalice, siempre es una buena idea comprobar que el esquemático no tiene errores de reglas electricas. Esto se realiza desde el menú **OPTIONS -> INTEGRITY TEST**. Podemos generar ahora un netlist EDIF yendo al menú **OPTIONS -> EXPORT NETLIST**.

### 3.9.11.- Simulación.

Podemos realizar ahora una simulación funcional del esquemático como se explico en la sección de simulación. La única diferencia es la presencia de buses. Para asignar señales a las entradas individuales del bus A y B, necesitamos expandir el bus en señales individuales. Esto se realiza seleccionando el bus A o B en la ventana de visor de cronograma y yendo al menú **SIGNAL -> BUS -> FLATTEN**. Podemos asignar ahora salidas del contador binario a la entradas individuales A y B. Podemos hacer clic en el icono de expansión de Bus en la barra de herramientas superior para comprimir o expandir el bus (tras crearlo primero), o ir al menú **SIGNAL -> BUS -> COMBINE** tras seleccionar las señales que queremos agrupar en un bus (en este caso será A0, A1, A2 y A3). Las señales en un bus comprimido se mostrarán en código hexadecimal (a no ser que se especifique otra).

## 3.10.- Errores más comunes.

Cuando tengamos un error, leer los mensajes de error cuidadosamente o consultar los informes para encontrar una explicación del error. A menudo encontraremos algunas sugerencias sobre lo que ha ido mal en estos informes. Para ayudarnos con la búsqueda de errores añadir puntos de prueba al esquemático de forma que podamos tracear y seguir los errores.

**NO USAR NOMBRES MÁS LARGOS DE 8 CARACTERES PARA PROYECTOS, MACROS O CARPETAS**. Xilinx no da un aviso cuando usamos nombres de más de 8 caracteres durante la síntesis (macros en ABEL), o la implementación sin explicar en que consiste el error.

La lista siguiente ofrece los errores más comunes encontrados en el laboratorio.

### 3.10.1.- Errores Generales.

1. **Nombres de ficheros en Xilinx**. No usar nombres de proyectos, macros o carpetas de más de 8 caracteres. Esto puede causar problemas inesperados como "Illegal Operation...Must import netlist first." Esto se aplica a todos los nombres que forma el camino entero. Para nombrar un fichero, el sistema usa el nombre de camino completo, ejemplo: c:\folder\<nombre de proyecto>. Ninguno de estos nombres debe se mayor de 8 caracteres.

2. **Fichero no encontrado**. Esto puede venir causado cuando copiamos nuestro proyecto a otra carpeta sin copiar el fichero .pdf. Es importante que siempre copiemos **la carpeta del proyecto Y el fichero .pdf**. Estos deben ir juntos. En caso que necesitemos una copia del proyecto se aconseja que no se copie usando el copiado y pegado del explorador de Windows, si no que lo copiemos desde Xilinx. En la ventana del gestor de proyecto, ir a FILE -> COPY PROJECT. Seleccionar la carpeta deseada y guardarla. Esto copiará ambos, carpeta de proyecto y fichero .pdf. Una forma incluso más apropiada es usar la opción ARCHIVE. Esto coloca la carpeta de proyecto y el fichero .pdf en un fichero zip que podemos copiar o mandarlo por e-mail facilmente.

### 3.10.2.- Errores con ABEL-HDL.

1. **No colocar las ecuaciones en la sección de ecuaciones**. Por ejemplo, colocar las ecuaciones tras la sección Truth_table provoca errores y ABEL no compilará.

2. **Definir la misma variable de salida dos veces**, una vez en la sección Equation y otra en la sección Truth_table. Esto confundirá al sistema.

3. **Olvidar definir un puerto de salida como "output"** en la opción de dirección o como salida registrada 'reg' en lugar de salida combinacional 'com'.

4. **El nombre de la macro ABEL tiene más de ocho caracteres.**

5. Tras modificar una macro que se usa en un esquemático de más alto nivel, necesitamos guardar y actualizar la macro. Lo mismo se aplica a una macro de esquemático que haya sido modificada. Asegurarse que sintetizamos el código y vamos al menú PROJECT-> UPDATE en Abel o a UPDATE simulator. Una forma segura de asegurarse que estamos usando los ficheros actualizados es salir del editor ABEL-HDL antes de abrir el simulador.

6. **Errores de sintaxis**. Estos pueden ser los más comunes y frustrantes. No escribir rapidamente y sin prestar atención el fichero fuente; es mucho mejor ir más despacio y ahorrarse después bastante tiempo al buscar los errores.

7. **Definiciones de conjuntos anidadas**. Por ejemplo, hemos definido un conjunto X=[X7..X0] y necesitamos usarlo como parte de otro conjunto, hacemos INPUT que contiene las Xs y otra variable Cin. Es un error definir INPUT=[Cin, X]. En vez de esta forma, podemos definirlo como INPUT = [Cin, X7 ..X0]. Este error no se avisa durante el chequeo de sintaxis si no que dará resultados de simulación inpredecibles.

8. **Uso incorrecto de parentesis**. Por ejemplo en los When-Then-Else de las sentencias If-Then-Else cuando usamos multiples ecuaciones tras la palabra clave THEN, tenemos que colocar estas ecuaciones entre parentesis. Por supuesto, asegurarse si usamos parentesis que cada parentesis abierto se empareja con otro cerrado.

9. Con las máquinas de estados finitas, debemos **definir las señales de reloj** de la máquina con ecuaciones (ejemplo: [Q1, Q0].clk = CLOCK;).

### 3.10.3.- Errores con el Editor de Esquemáticos.

1. **Si colocamos símbolos muy cercanos entre sí parece que están conectados**. Esto no es cierto. Debemos conectar los símbolos con un hilo (línea azul). Lo mejor es no colocar los símbolos demasiado cerca para que tengamos espacio de colocar un hilo entre ellos. El test de integridad captura este error. En la simulación es probable que se muestren señales que tengan una salida X o Z.

2. **Cuando nombramos una conexión** usando el comando "Name Wire", debemos hacer clic en la conexión (hilo) con cuidado de forma que el nombre se enganche a ese hilo. El nombre debe aparecer en azul. Si no lo conectamos a un hilo, el nombre aparecerá (en verde) pero no será un nombre de conexión.

3. **No nombrar diferentes conexiones con el mismo nombre**. Cada nombre de conexión tiene que tener un nombre único, a no ser que deban conectarse de forma eléctrica.

4. **Modificación de una macro o parte de un esquemático y no realizar la actualización de la macro** o esquemático. Esto puede provocar simulaciones incorrectas. Tras cambiar el esquemático, hacer un UPDATE SIMULATOR y exportar un netlist. También puede ayudar cerrar el esquemático tras las modificaciones para asegurarse que se actualiza.

5. **Cuando se usen buses, no tener la correspondencia correcta entre señales en dos buses** que se conecten (como conectar el bit más significativo al bit menos significativo). Tener cuidado en como se definen los buses y se conectan. Prestar especial atención a las señales cuando se conecten los buses.

6. **Volver a salvar el esquemático de mayor nivel con el nombre de una macro que ya exista**.

7. **Intentar mezclar ficheros de mayor nivel**. El esquemático de mayor nivel (o fichero ABEL) tiene normalmente el mismo nombre que nuestro proyecto. No cambiar esto. Un proyecto puede tener uno o más esquemáticos de mayor nivel o ficheros de diseño (ejemplo; PROJECT_NAME1.SCH, PROJET_NAME2.SCH). Sin embargo tienen que ser del mismo tipo, como un esquemático (.sch) o HDL (X-ABEL o VHDL) pero no ambos. Para comprobar cual es el fichero de mayor nivel del proyecto, mirar a la izquierda de la ventana del gestor de proyectos. Bajo el nombre del fichero .pdf del proyecto veremos listados ficheros .sch o ficheros .abl.

8. **Cuando se use una conexión a masa o a vcc, debemos usar los símbolos GND o VCC de la Librería de Símbolos**. No usar el símbolo GND que se muestra a la izquierda de la barra de herramientas (la segunda desde abajo) en el editor de esquemáticos.

9. **Cuando tengamos el error de "Bus has multiple drivers" durante la implementación**, esto puede indicar que tenemos más de un esquemático de mayor nivel. Por ejemplo, podemos crear una macro y guardarla en el esquemático. Podemos comprobar esto en la ventana del gestor de proyecto. Este segundo esquemático puede tener los mismos nombres para los buses que causarán un conflicto. Debemos eliminar cualquier esquemático que no forme parte del de mayor nivel.

10. **El esquemático muestra los símbolos lógicos en gris claro**. Aparece un mensaje diciendo "Automatic loading of project libraries disabled". Esto puede suceder cuando añadimos un esquemático a un proyecto nuevo. Para corregir este problema debemos añadir las librerías adecuadas al proyecto. Desde el gestor de proyecto seleccionar el menú "**File -> Project Libraries...**", seleccionar "xc4000xl" de "Attached Libraries" (o la familia FPGA que estemos usando). Seleccionar "ADD", "xc4000xl" debe aparecer ahora bajo "Project Libraries". Podemos necesitar también añadir la librería asociada con el esquemático que estamos añadiendo. Si el nombre del esquemático que estamos añadiendo es "FOO.SCH" debemos añadir la librería "FOO". Seleccionar el menú "**File -> Project Libraries**", seleccionar "FOO" de "Attached Libraries", seleccionar "ADD", "FOO" debe aparecer ahora bajo "Project Libraries". Volver a abrir el esquemático.

### 3.10.4.- Errores con el Simulador.

Los errores con el simulador a menudo son el resultado de errores en el esquemático o fichero fuente ABEL. Ver la sección 3.9.1 "Errores Comunes" con ABEL o esquemático.

1. **No actualizar el netlist o proyecto tras realizar una modificación al esquemático o fichero ABEL** antes de ejecutar la simulación. Esto puede provocar salidas en HiZ (Alta Impedancia) en el simulador. Una forma segura de asegurarse que estamos usando los ficheros actualizados es salir del editor (Esquemático o ABEL-HDL) tras realizar las

modificaciones. Salir también del simulador y volver a abrirlo tras realizar las modificaciones.

2. **Indicaciones incorrectas cuando usamos buses para mostrar las señales**. Esto puede ser el resultado de un bus al que se le ha cambiado la dirección, esto es, el bit más significativo se ha convertido en el menos significativo, etc. Esto se puede cambiar en el menú **SIGNAL->BUS->Change Direction**.

3. Cuando se abre la ventana de selección de componente (selección de señal), **no aparecen señales**. Esto puede ocurrir cuando acabamos de crear un nuevo proyecto y definimos un fichero fuente ABEL. Necesitamos añadir el fichero al proyecto. En la ventana del gestor de proyectos ir al menú **DOCUMENT->ADD**. Hojear hasta que encontremos el fichero que tiene el diseño que queremos simular. Volver al simulador y si es necesario rearrancar el simulador.

Un problema común es encontrar **señales indefinidas** que se resaltan en el simulador como líneas grises o como cajas azules con una X en el esquemático. Esto puede estar causado por una gran variedad de razones. Aquí tenemos unas cuantas:

- ✍✍**Hemos modificado el esquemático y no actualizado el netlist y simulador**. Puede ayudar el cerrar el editor de esquemáticos y la ventana del simulador primero y luego abrir de nuevo el simulador para asegurar que usa el último netlist.
- ✍✍Cuando **diferentes conexiones tienen el mismo nombre**. Por ejemplo, nombramos una conexión de entrada como ANAME y la conexión tras el buffer como ANAME. Aunque ambas conexiones ven la misma señal, son conexiones físicamente diferentes porque el buffer los separa. Las conexiones deben tener nombres únicos, a no ser que estén conectados eléctricamente.
- ✍✍**Durante la simulación física o temporal**. Las señales indefinidas pueden ser el resultado del programa de síntesis que a menudo optimiza señales (o lógica) eliminándolas. Ver también **señales indefinidas** en el apartado del simulador temporal.

### 3.10.5.- Errores con la Implementación.

1. **Error Signal: Net has multiple connections or multiple drivers** (Error en señal: la conexión tiene conexiones múltiples o fuentes múltiples) . Esto puede ser el resultado de dar el mismo nombre a diferentes conexiones que no deben estar conectadas, o darle a una conexión más de un nombre. Esto también se aplica a los buses. Para ayudarnos a encontrar que está conectado a una conexión particular podemos usar la ventana de consulta (ir a **MODE -> QUERY** o hacer clic en el icono de la derecha de la barra de herramientas superior) en el editor de esquemáticos. El cursor cambia a un símbolo de interrogación. Hacer clic en la conexión de interés y esta se mostrará en la ventana de consulta. Comprobar las conexiones.

2. **Una macro en el proyecto tiene el mismo nombre que el proyecto de mayor nivel** (o esquemático).

### 3.10.6.- Errores con el fichero de restricciones de usuario.

1. **Errores de sintaxis**. El fichero de restricciones de usuario es sensible a mayúsculas y minúsculas, ah es diferente de Ah. Asegurarse que los nombres de conexión (netnames) en el fichero ucf son exactamente los mismos que los del esquemático o fichero ABEL. No mezclar la letra "O" (o mayúscula) con el número "0" (cero).

2. **Olvidar colocar "P" enfrente del número de patilla**. El número de patilla debe venir precedido por una P como P19;

3. **Olvidar el punto y coma al final de una línea.**

4. **Los comentarios** deben ir precedidos por un símbolo # (almohadilla). No es necesario un punto y como tras la línea de comentario.

5. **Se genera un mensaje de error durante el "Mapping" (Implementación) relacionado con colocación de pines.** Comprobar que hemos especificado la referencia de componente correcta y número de patillas (estamos usando la familia XC4000XL con 84 patillas). Esto se puede hacer en la ventana del gestor de proyectos, haciendo clic en el icono de Información de Diseño, o desde la ventana del gestor de diseño ( ir a **DESIGN -> IMPLEMENT**); en la ventana de implementación hacer clic en el botón Part y seleccionar la referencia de dispositivo correcta).