Práctica 1. Introducción al diseño con Xilinx ISE.

1. Introducción.

Esta práctica sirve de introducción a la descripción de sistemas mediante el lenguaje VHDL. Para ello, emplearemos el simulador de VHDL Modelsim que acompaña las herramientas del entorno de diseño de Xilinx. Este simulador admite el mismo conjunto de VHDL capaz de sintetizarse en los dispositivos programables Xilinx dado que incorpora librerías específicas para estos dispositivos. Pero a pesar de ello, es un simulador de uso muy extendido dado que a través de las librerías es posible un uso más general.

A primera vista, Modelsim resulta un programa no demasiado intuitivo, pero si se siguen los pasos apropiados, se puede realizar la simulación de forma sencilla. Esta complejidad viene dada por la posibilidad de controlar el simulador mediante comandos, y su capacidad para poder simular todo tipo de características VHDL, incluso aquellas no sintetizables; además de admitir descripciones en Verilog.

Generalmente, el uso de ModelSim se realiza desde el entorno Xilinx, ya que es posible lanzar la simulación ModelSim del código VHDL que se está diseñando en Xilinx ISE de forma automática. Así, ModelSim se ejecuta automáticamente desde Xilinx ISE.

Si se realizan modificaciones en el código VHDL, es necesario cerrar ModelSim, guardar las modificaciones realizadas, y ejecutarlo de nuevo desde ISE. Un proyecto crea un directorio donde se introducen todos los ficheros relacionados, y uno o varios subdirectorios con las librerías creadas (por defecto siempre se crea la librería **work**).

2. Objetivo de la práctica.

En esta práctica se realiza una aproximación al manejo del entorno de diseño de FPGA y CPLD de Xilinx, denominado Xilinx ISE.

Mediante la realización de varios elementos funcionales sencillos que se describen a continuación, realizaremos el proceso completo de simulación e implementación de un diseño, para así conocer el proceso de diseño que se debe seguir en la mayor parte de las sesiones de prácticas de esta asignatura.

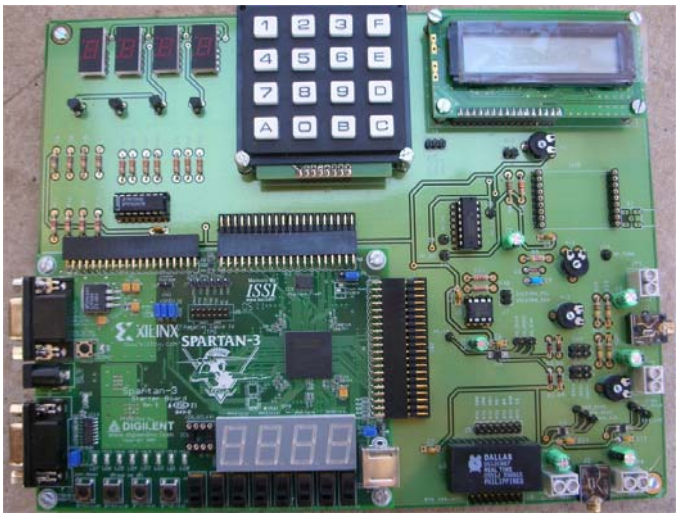
3. Descripción de la placa de programación.

Esta placa esta formada realmente por dos placas independientes:

1. Una placa comercial DIGILENT con una FPGA Xilinx Spartan 3 XC3s200FT256-4 y algunos periféricos integrados. Esta placa, posee LED's (8), Display de 7 segmentos (4), puerto PS2, puerto serie RS232, 2 módulos de 256x16 de SRAM (1MByte), 2 Mbit de PROM para almacenamiento permanente de la programación de la FPGA, 4 pulsadores, 8 microinterruptores, reloj de 50MHz y 3 conectores de expansión. Consultar **S3BOARD-rm.pdf** para información detallada de la placa.
2. Por otro lado, una placa de mayores dimensiones donde se encuentran más periféricos:
    a. Cuatro display de 7 segmentos controlados a través de un mismo bus BCD y habilitación individual que incluye el decodificador BCD a 7 segmentos 74LS47, con lo que se requieren 8 líneas para su control (4 líneas para el código BCD y 4 líneas de habilitación).
    b. Un conversor analógico/digital serie de 12 bits y dos canales National Semiconductor ADC122S021 [National Semiconductor Corp. *ADC122S021 Data Sheet*. USA. 2005], 200ksps donde únicamente son necesarias 4 líneas de conexión con la FPGA.
    c. Dos conversores digital/analógico Burr-Brown DAC7513 de 12 bits [Texas Instruments. *Burr-Brown DAC7513 Data Sheet*. USA. 2003], cada uno con 3 líneas de conexión con la FPGA.
    d. Un teclado hexadecimal pasivo con 4 líneas de entrada y 4 de salida para realizar lectura de tecla.
    e. Una memoria I^2C EEPROM serie de 8 bits Microchip 24LC08B [Microchip Technology Inc. *24AA08/24LC08B Data Sheet*. USA. 2002] que precisa de dos líneas de comunicación para lectura/escritura.
    f. Un display LCD 16x2 con controlador estándar HD44780 [Hitachi. *HD44780U LCD Controller*. Japan. 1999].
    g. Un reloj de tiempo real no volátil Dallas DS12C887 [Dallas Semiconductor. *Real Time Clock DS12C887 Data Sheet*. USA. 2000].

h. Un zócalo para dispositivo de comunicación inalámbrica (por el momento no conectado).

Respecto de los conversores D/A y A/D, es posible conectarlos a un conector de clavija de 3,5mm para desarrollar aplicaciones de audio (se permite el control de dos canales para permitir procesado estéreo de la señal), siendo posible conectar cualquier dispositivo de entrada (reproductor MP3, ordenador PC, walkman, etc.) y salida (auriculares, altavoces autoamplificados, etc.), aunque es posible elegir si la entrada/salida proviene de la clavija, o bien de un conector de tornillo o un potenciómetro incorporado en la placa (el potenciómetro permite probar si se leen valores correctos de ADC sin necesidad de conectar señal externa). Como regla general, se incluyen puntas de prueba sobre las señales en todos los componentes, para poder depurar con analizador lógico u osciloscopio las señales de comunicación entre la FPGA y cada uno de los periféricos.



INFORMACIÓN DEL PATILLAJE DE CONEXIÓN ENTRE LA FPGA Y LOS PERIFÉRICOS.

El patillaje de los conectores externos es el siguiente:

Table 13-3: **Pinout for A2 Expansion Connector**

Schematic Name	FPGA Pin	Connector		FPGA Pin	Schematic Name
GND		1	2		VU (+5V)
V_{CCO} (+3.3V)	V_{CCO} (all banks)	3	4	(E6)	**BCD_1**
Teclado: Y4	(D5)	5	6	(C5)	**BCD_2**
Teclado: Y3	(D6)	7	8	(C6)	**BCD_3**
Teclado: Y2	(E7)	9	10	(C7)	**BCD_4**
Teclado: Y1	(D7)	11	12	(C8)	**D1_EN**
Teclado: X4	(D8)	13	14	(C9)	**D2_EN**
Teclado: X3	(D10)	15	16	(A3)	**D3_EN**
Teclado: X2	(B4)	17	18	(A4)	**D4_EN**
Teclado: X1	(B5)	19	20	(A5)	**LCD_D0**
PA-IO18	(B6)	21	22	(B7)	**LCD_D1**
MA2-DB1	(A7)	23	24	(B8)	**LCD_D2**
MA2-DB3	(A8)	25	26	(A9)	**LCD_D3**
MA2-DB5	(B10)	27	28	(A10)	**LCD_D4**
LCD: RS	(B11)	29	30	(B12)	**LCD_D5**
LCD: R/W	(A12)	31	32	(B13)	**LCD_D6**
LCD: E	(A13)	33	34	(B14)	**LCD_D7**
MA2-INT/GCK4	(D9) Oscillator socket	35	36	(B3) FPGA PROG_B	PROG-B
DONE	(R14) FPGA DONE	37	38	(N9) FPGA INIT_B	INIT
CCLK	(T15) FPGA CCLK	39	40	(M11)	DIN

Table 13-4: **Pinout for B1 Expansion Connector**

Schematic Name	FPGA Pin	Connector		FPGA Pin	Schematic Name
GND		1	2		VU (+5V)
V_{CCO} (+3.3V)	V_{CCO} (all banks)	3	4	(C10)	**RTC_D0**
RF_CD	(T3) FPGA RD_WR_B config	5	6	(E10)	**RTC_D1**
RF_OUT	(N11) FPGA D1 config	7	8	(C11)	**RTC_D2**
RF_IN	(P10) FPGA D2 config	9	10	(D11)	**RTC_D3**
RF_TXEN	(R10) FPGA D3 config	11	12	(C12)	**RTC_D4**
24LC08B: SCL	(T7) FPGA D4 config	13	14	(D12)	**RTC_D5**
24LC08B: SDA	(R7) FPGA D5 config	15	16	(E11)	**RTC_D6**
AD_CS	(N6) FPGA D6 config	17	18	(B16)	**RTC_D7**
AD_SCLK	(M6) FPGA D7 config	19	20	(R3) FPGA CS_B config	**RTC_SQW**
AD_DOUT	(C15)	21	22	(C16)	**RTC_IRQ**
AD_DIN	(D15)	23	24	(D16)	**RTC_RST**
DA1_SYNC	(E15)	25	26	(E16)	**RTC_RD**
DA1_SCLK	(F15)	27	28	(G15)	**RTC_WR**
DA1_DIN	(G16)	29	30	(H15)	**RTC_ADR**
DA2_SYNC	(H16)	31	32	(J16)	**RTC_CS**
DA2_SCLK	(K16)	33	34	(K15)	MB1-RESET
DA2_DIN	(L15)	35	36	(B3) FPGA PROG_B	PROG-B
DONE	(R14) FPGA DONE	37	38	(N9) FPGA INIT_B	INIT
CCLK	(T15) FPGA CCLK Connects to (A14) via	39	40	(M11)	DIN

NOTA: CONSULTAR DATA-SHEET DE CADA UNO DE LOS COMPONENTES Y REVISAR ESQUEMÁTICO DE PLACA DE EXPANSIÓN PARA CONOCER SIGNIFICADO DE CADA PATILLA.

En esta práctica no emplearemos estos dispositivos periféricos, pero dado que sí lo haremos en futuras sesiones, se proporciona ahora toda la información para tenerla como referencia, por lo que será interesante tener a mano esta información para el desarrollo de futuras prácticas.

4. Flujo de Diseño con Xilinx ISE.

Objetivos

- Tras completar esta práctica podrás desarrollar programas sencillos para realizar una programación de una FPGA, incluyendo la simulación

---

Paso 1

Ejecuta ISE Project Navigator y crea un Nuevo proyecto.

❶ Selecciona **Inicio → Programas → FPGA → Xilinx ISE 8.2i → Project Navigator**

Si aparece algún mensaje de actualización WebUpdate, puedes descartar la acción y continuar.

❷ Una vez en Project Navigator, selecciona **File → New Project**

Aparece el asistente para Nuevo proyecto (**Figura 1-1**)

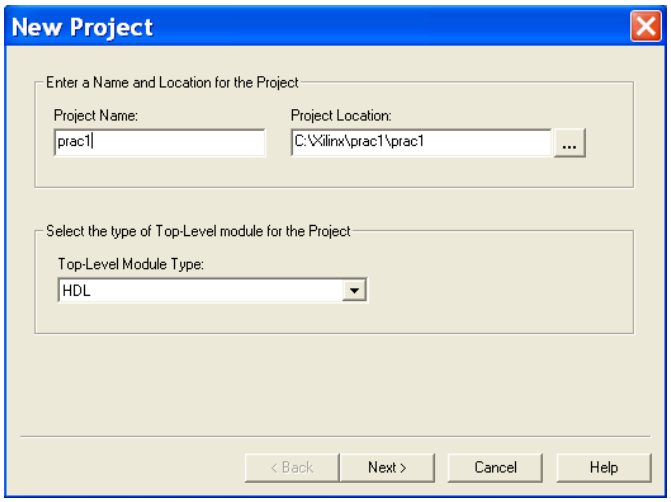


Figura 1-1. Asistente de Nuevo Proyecto

❸ Para elegir la carpeta destino, pulsa el botón "**...**" y una vez seleccionado el destino, pulsa <OK>
- En nuestro caso, elegiremos: *c:\xilinx\dcse* (si la carpeta "dcse" no está creada, crearla previamente)

❹ Asigna un nombre al proyecto, en este caso, Lab1.

❺ Pulsa **Next**

Aparece la ventana de selección del dispositivo que vamos a emplear (**Figura 1-2**)

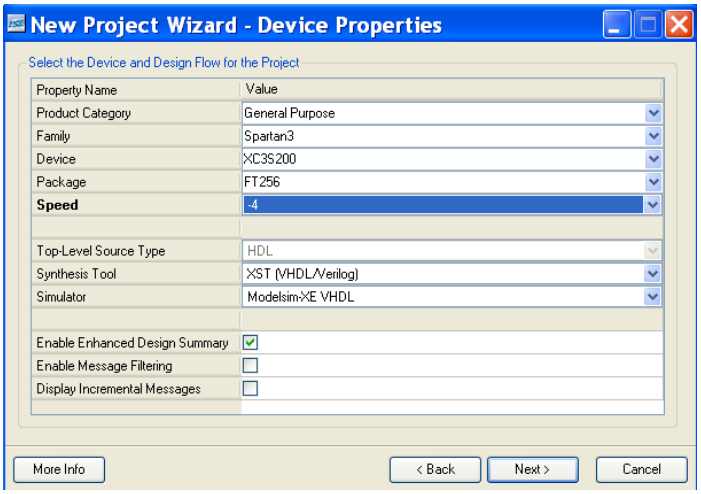


Figura 1-2. Selección del dispositivo y tipo de diseño (XST-VHDL) e ISE Simulator

❻ Dada la placa que disponemos, elegiremos como opciones de dispositivo las siguientes:

- ✓ Device Family: **Spartan3**
- ✓ Device: **xc3s200**
- ✓ Package: **ft256**
- ✓ Speed Grade: **–4**
- ✓ Synthesis Tool: **XST (VHDL/Verilog)**
- ✓ Simulator: **ModelSim XE Simulator**

Al pulsar Next, aparece una nueva ventana para crear una nueva fuente de diseño (Create New Source, **Figura 1-3**). Este diálogo permite crear un "esqueleto" de un código VHDL para posteriormente diseñarlo, apareciendo una serie de ventanas que facilitan la labor de crear este tipo de diseños. Si los ficheros VHDL ya existen (como es nuestro caso en esta primera práctica), no es necesario ejecutar crear ningún código nuevo.

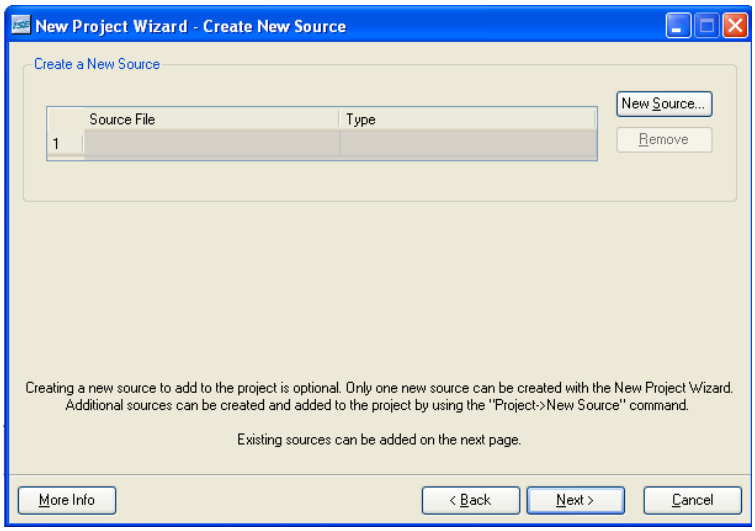


Figura 1-3. Create New Source

❼ Click **Next**

En la ventana "Add Existing Sources" es donde se incorporarán al diseño los ficheros de código VHDL que ya tengamos disponibles. (**Figura 1-4**).

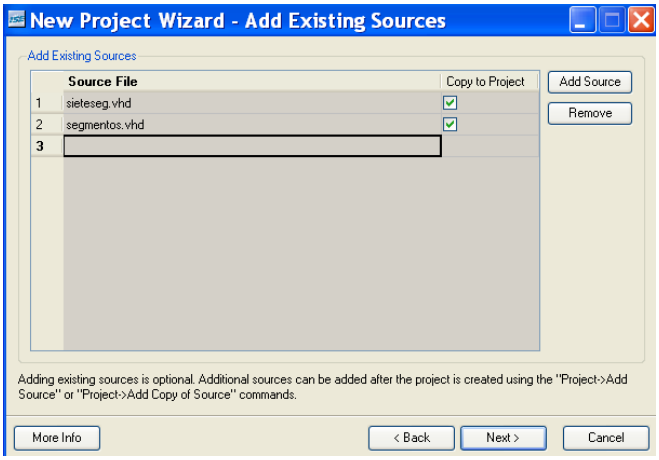


Figura 1-4. Add Existing Sources

Añadir ficheros de diseño a nuestro proyecto Paso 2

Los pasos a seguir son los siguientes:

❶ Haz Click en **Add Source** y selecciona los ficheros ***segmentos.vhd*** y ***sieteseg.vhd*** que previamente debes haberte descargado del Aula Virtual de la UV.

❷ Haz click en **Open,** para cada uno de ellos.

❸ Haz Click en **<Next>** dejando marcada la opción de que el fichero se copie al directorio del proyecto. Click en **Finish**.

❹ Click <**OK**>

❼ Con esto, ya tendremos el proyecto definido, con las fuentes de código de nuestro diseño.

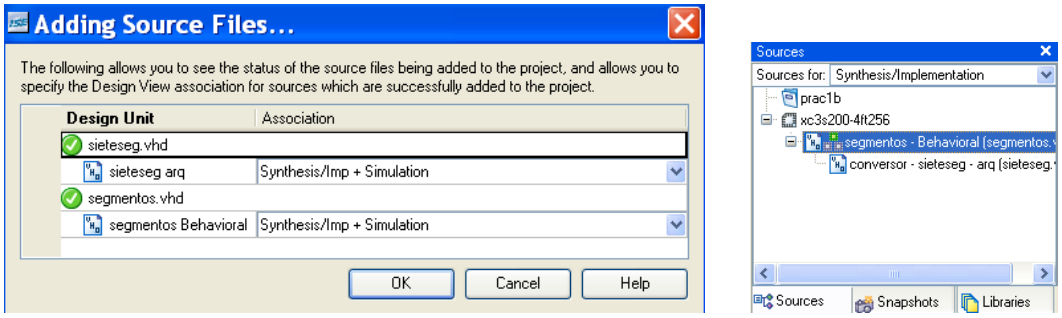


Figura 1-5. Vista resumida y jerárquica de las Fuentes contenidas en el diseño

Actividad:

Abre el código de los ficheros .vhd el diseño e intenta averiguar lo que hace el sistema, describiendo cada una de las partes del programa principal **segmentos.vhd**, así como el secundario (**sieteseg.vhd**).

Simular el Diseño — Paso 3

Añade el fichero de Test ***testseg.vhd*** y ejecuta la simulación comportamental (**behavioral**), para ello, es necesario: añadir este fichero al proyecto a través de la opción descrita seguidamente:

❶ Selecciona **Project → Add Copy of Source** y busca el fichero ***testseg.vhd*** en la carpeta donde lo hayas guardado

❷ Una vez seleccionado, haz click en <Open>

❸ Elige **Simulation Only** y click en <OK> para añadir el fichero al proyecto.

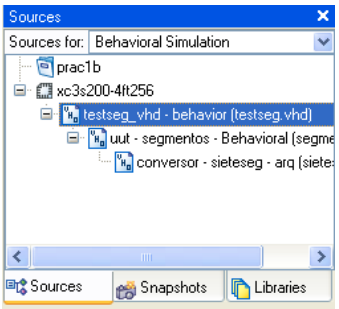


Figura 1-6. Vista jerárquica del diseño en modo simulación

❹ Dentro de esa ventana, marca el fichero de test (testseg.vhd) y en la ventana de procesos, expande la opción **ModelSim Simulator**, haz clic con el botón derecho en **Simulate Behavioral Model**, y elige **Properties**.

❺ Introduce el valor de 10000 para **Simulation Run Time** y haz clic en <OK>

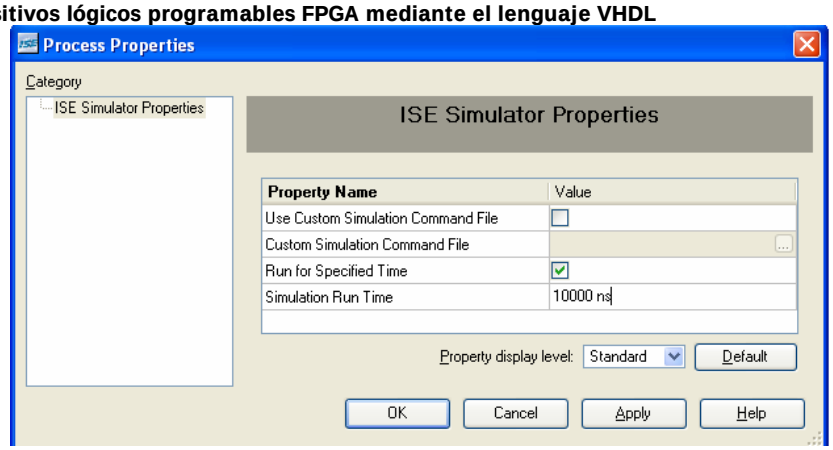
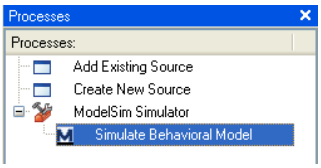


Figura 1-7. Propiedades de la simulación



❻ Haz doble-click en **Simulate Behavioral Model** para lanzar la simulación:

Por defecto, se simula un determinado tramo de tiempo, si se desea simular un tiempo mayor, es necesario elegir ese tramo de tiempo en el desplegable que por defecto marca "100 ps", y pulsar el botón inmediatamente a su derecha. Si se pulsa repetidamente, en cada pulsación se verá cómo se simula un tramo de tiempo.

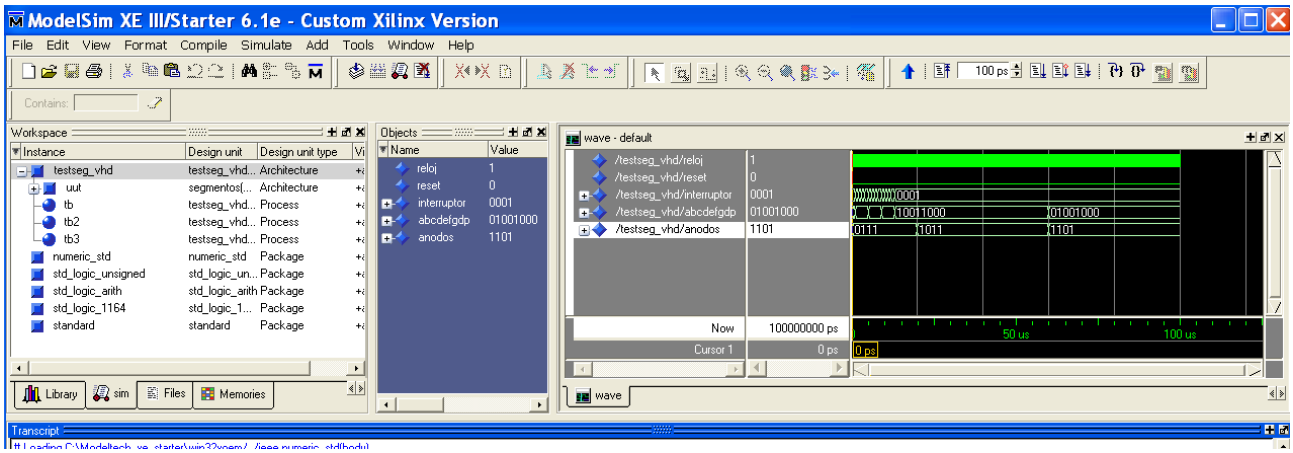


Figura 1-8. Simulador ModelSim HDL

Actividad:

Comprueba que los estímulos de entrada aplicados al diseño se corresponden con el fichero de test realizado.

Comprueba que los resultados de salida son los esperados.

Si deseo visualizar otras señales en la ventana **Wave**, ¿cómo las añado?

Asignación de patillas — Paso 4

Antes de la implementación, se debe realizar la asignación de patillas para asegurarse de que cada una de las señales externas de la FPGA queda conectada al dispositivo electrónico que hemos elegido. En este caso, las conexiones van dirigidas al reloj externo, los pulsadores e interruptores, y los display de 7 segmentos.

❶ La información de la asignación de patillas se debe realizar con la opción de ***User Constraints →Assign Package Pins*** en la ventana de procesos. Nos preguntará que si queremos que esta asignación de patillas se incorpore al proyecto, a lo que diremos que sí.

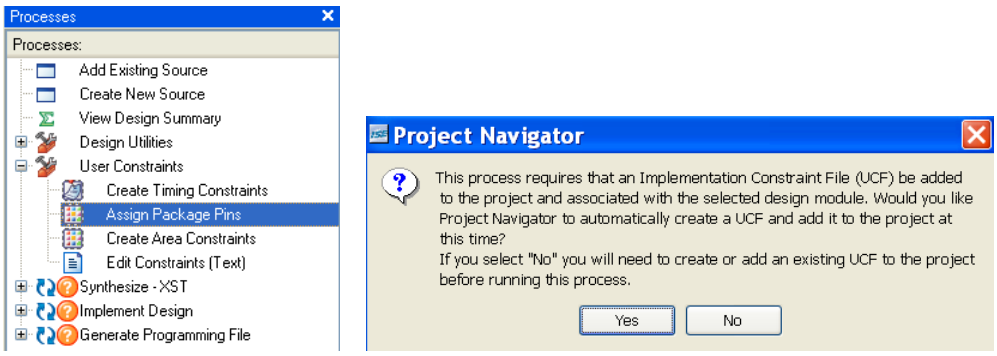
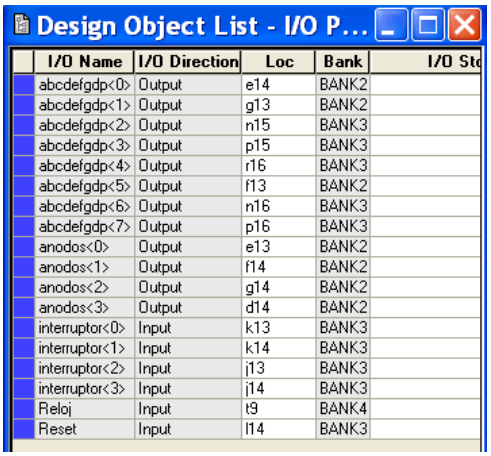


Figura 1-9. Selección de la opción para asignación de patillas.

❷ Nos aparece una ventana con el editor de patillaje PACE; en la columna ***Loc*** procederemos a rellenar el patillaje de cada una de las señales del diseño según la figura siguiente:



I/O Name	I/O Direction	Loc	Bank	I/O St
abcdefgdp<0>	Output	e14	BANK2	
abcdefgdp<1>	Output	g13	BANK2	
abcdefgdp<2>	Output	n15	BANK3	
abcdefgdp<3>	Output	p15	BANK3	
abcdefgdp<4>	Output	r16	BANK3	
abcdefgdp<5>	Output	f13	BANK2	
abcdefgdp<6>	Output	n16	BANK3	
abcdefgdp<7>	Output	p16	BANK3	
anodos<0>	Output	e13	BANK2	
anodos<1>	Output	f14	BANK2	
anodos<2>	Output	g14	BANK2	
anodos<3>	Output	d14	BANK2	
interruptor<0>	Input	k13	BANK3	
interruptor<1>	Input	k14	BANK3	
interruptor<2>	Input	j13	BANK3	
interruptor<3>	Input	j14	BANK3	
Reloj	Input	t9	BANK4	
Reset	Input	l14	BANK3	

❸ Una vez hecho esto, se guarda el fichero, y automáticamente se añadirá dentro de la ventana de fuentes como un elemento de diseño más, asegurando que las entradas y salidas están ubicadas donde se ha especificado aquí.

Implementación del Diseño — Paso 5

Durante la implementación, se generan diversos informes (reports), que proporcionan información sobre cómo se ha desarrollado el proceso de implementación. Para realizar este proceso seguiremos los siguientes pasos:

❶ En la ventana de Fuentes (**Sources**) seleccionamos **Sources for: Synthesis/Implementation** y el fichero de máximo nivel jerárquico ***segmentos.vhd***

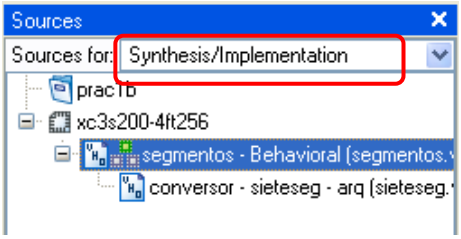


Figura 1-10. Ventana de Fuentes para proceder a la implementación.

❷ En la ventana de Procesos, hacemos doble-click en **Implement Design** (**Figura 1-11**)

Notice that the tools run all of the processes required to implement the design. In this case, the tools run Synthesis before going into Implementation.

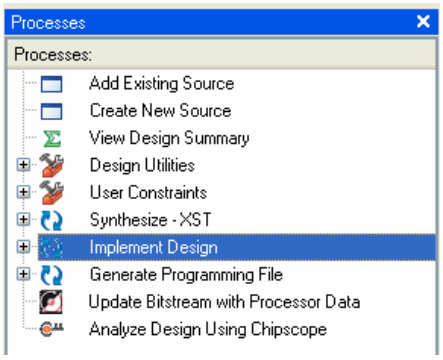


Figura 1-11. Ventana de Procesos que aparece al marcar la fuente.

❸ Mientras la implementación se lleva a cabo, puedes pulsar en el símbolo + al lado de **Implement Design** para expandir el proceso de implementación y ver el progreso. Para este diseño, puede que haya alguna etapa con avisos, pero se pueden ignorar de forma segura ya que no afectan el resultado final.

Cada vez que una etapa finalice, aparece alguno de los siguientes símbolos:

✓ Marca de verificación si el proceso se ha ejecutado con éxito.

! Exclamación para un resultado con avisos (warnings)

✕ Un aspa para el caso de existir errores.

❹ Los mensajes de error y aviso se pueden leer en la consola inferior del entorno de diseño.

❺ Cuando la implementación está completada, puedes revisar el resultado, y por ejemplo, en **Design Summary** puedes ver el nivel de ocupación que el tiene el diseño con respecto al total de recursos disponibles en el dispositivo seleccionado (**Figura 1-12**).

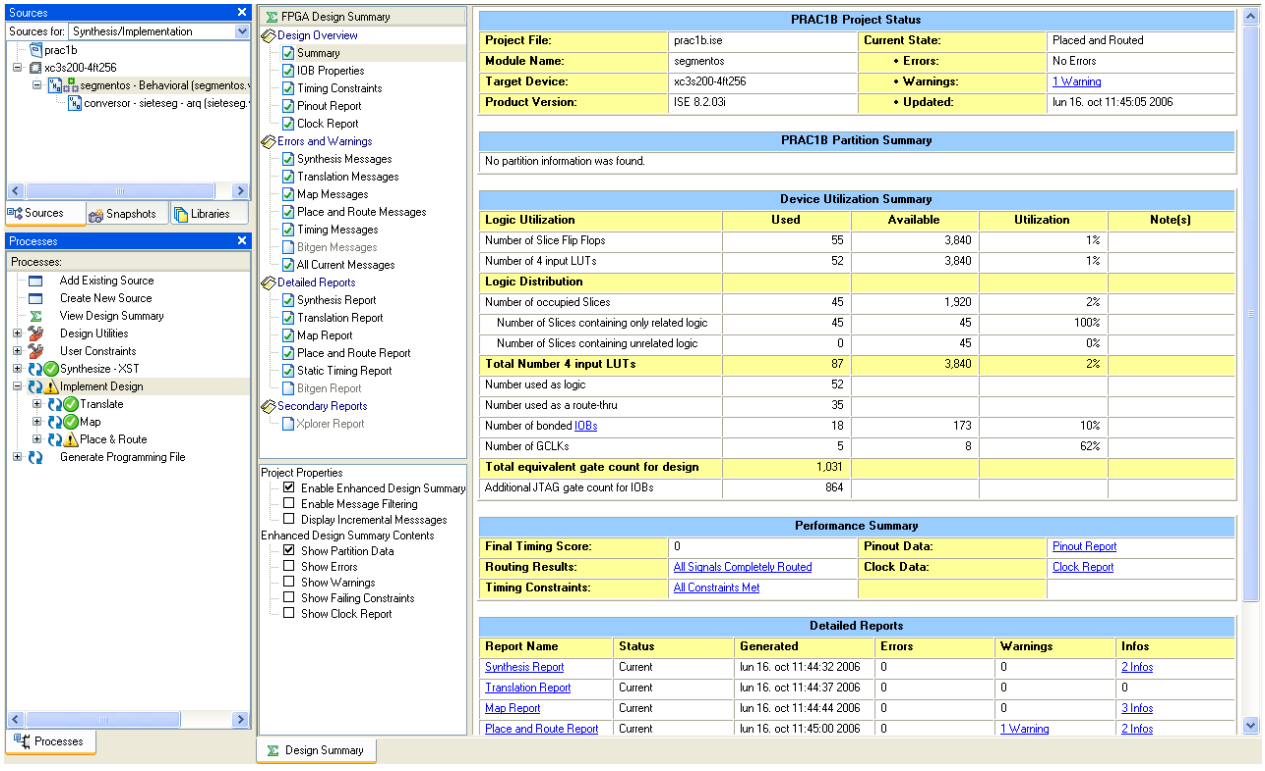


Figura 1-12. Design Summary

Programación de la FPGA. Paso 6

Para programar la FPGA, en la ventana de procesos se debe ejecutar la opción **"Generate Programming File"** (si queremos evitar un aviso a la hora de implementación, podemos hacer clic en el botón derecho, ir a **Properties**, y en la categoría **Startup options → FPGA startup clock**, seleccionar **JTAG clock).**

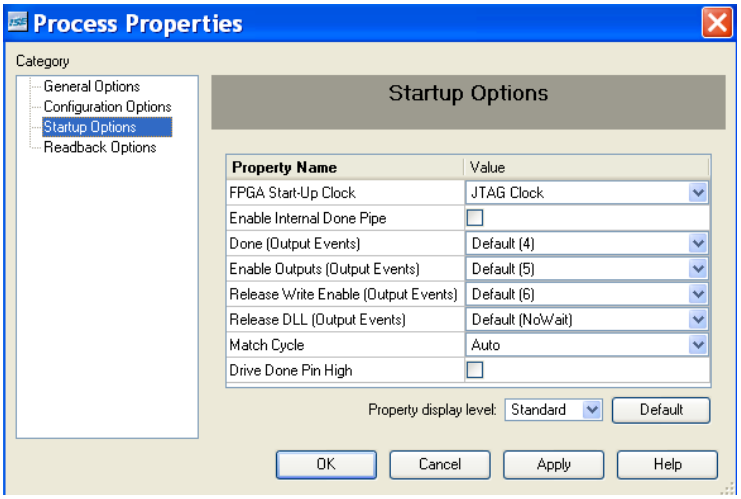


Figura 1-13. Propiedades de Generate Programming File.

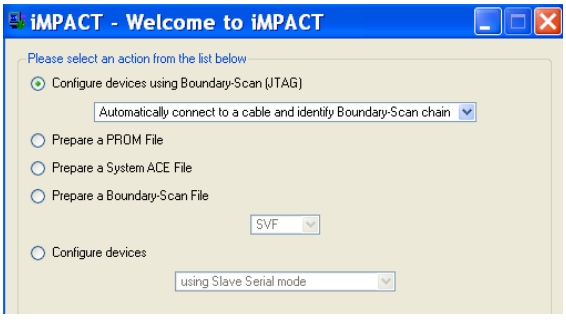
Posteriormente, ejecutar la opción **Configure Device (iMPACT)** para arrancar el programa de descarga a la FPGA, donde generaremos el fichero binario para descargar en la FPGA a través del puerto JTAG de la placa Digilent.

❶ Asegúrate que el cable de configuración JTAG está conectado a la placa y al Puerto paralelo del PC.

❷ Conecta la alimentación de la placa DIGILENT

❸ Haz doble click en **Configure Device (iMPACT)**.

❹ La siguiente ventana aparece:



❺ Deja seleccionada la misma opción de la figura, verificando que ésta se corresponde con **Boundary-Scan Mode**, click <Finish>

❻ Click <OK> cuando aparezca una ventana informando de que se han detectado dos dispositivos en la cadena JTAG.

❼ Asigna el ficheros ***segmentos.bit*** al dispositivo xc3s200 (primer dispositivo en la cadena JTAG) y **bypass** el segundo dispositivo.

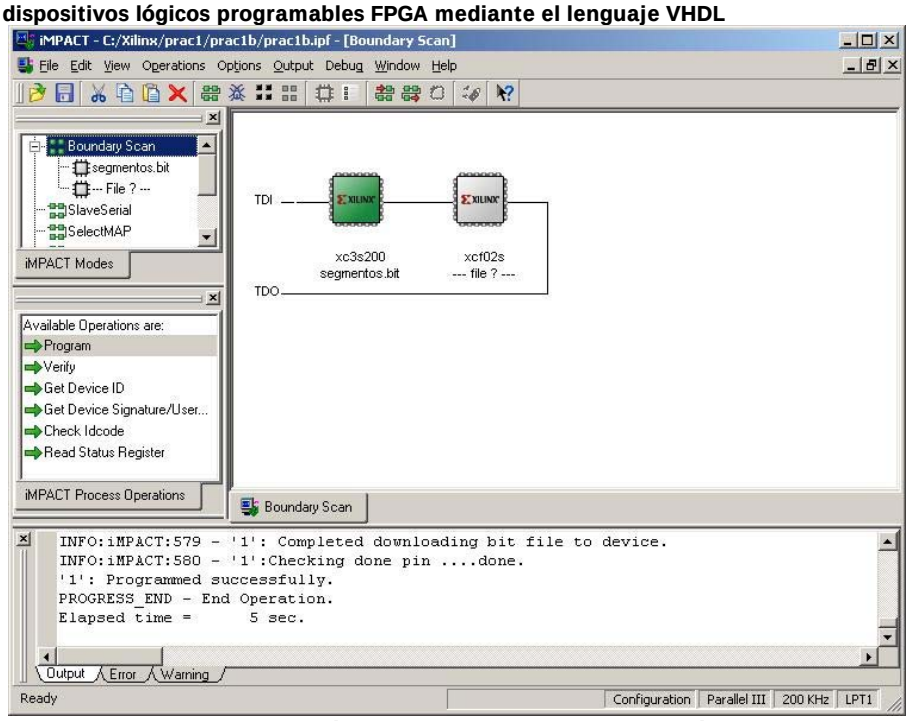


Figura 1-14. Asignación del fichero de programación a la FPGA

❽ Haz click con el botón derecho sobre el dispositivo FPGA xc3s200 y ejecuta **Program,** aparece una ventana donde dejamos todas las opciones por defecto (importante **NO** activar la verificación).

❾ Una vez finalizado el proceso, la FPGA queda programada, y ya se puede comprobar si el diseño funciona de la forma en la que habíamos predicho. Pulsa sobre los interruptores para comprobar toda la funcionalidad del diseño

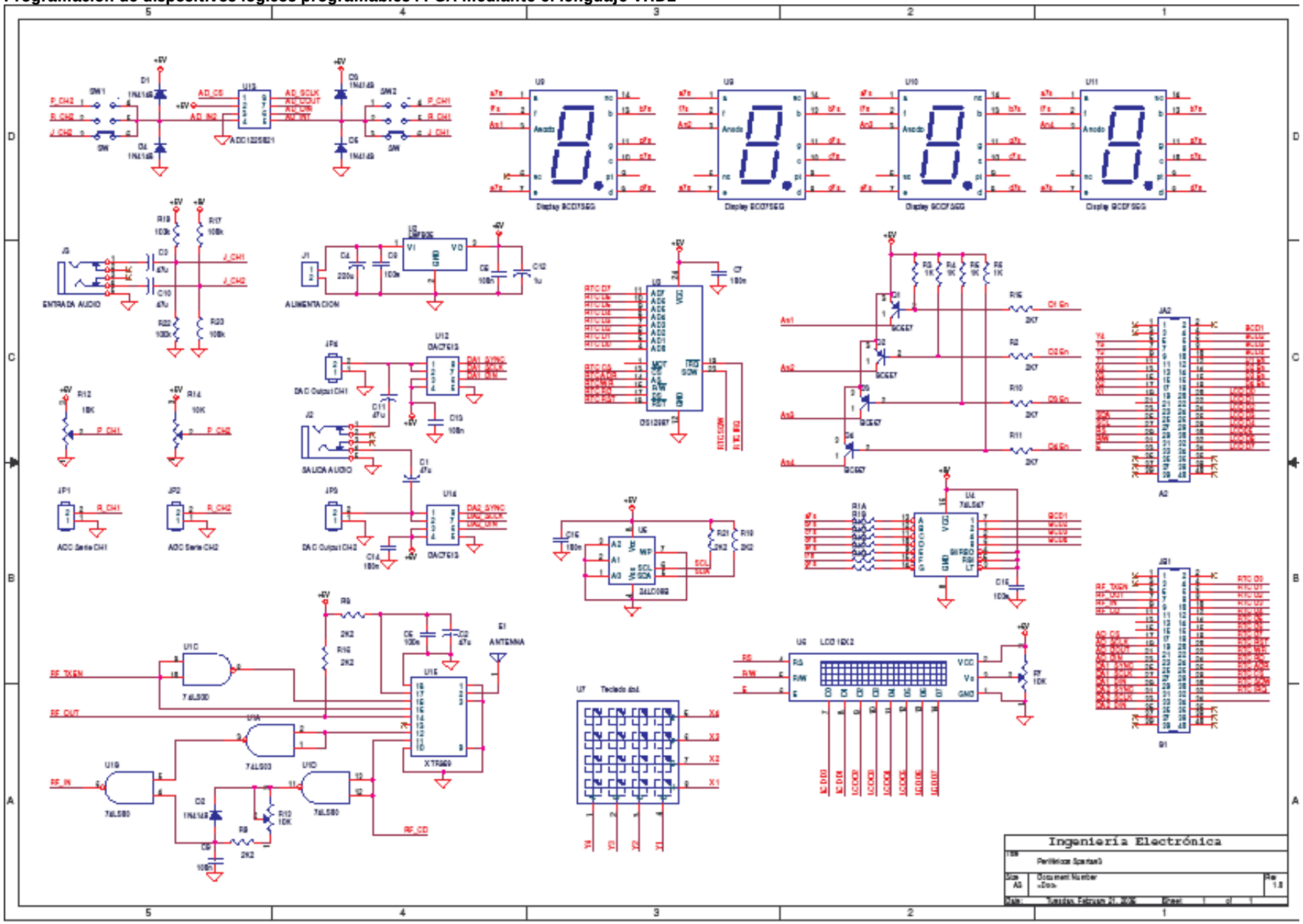
Actividad:

Como podrás ver, los display sólo se encienden alternativamente uno cada vez. ¿Sería posible hacer que todos se vieran a la vez, cada uno mostrando su información correspondiente. Además, la visualización no es correcta, intenta averiguar dónde está el error dentro del diseño realizado.

La acción de los interruptores, ¿Qué efecto causa?, ¿es ese el efecto que debiera causar?. Si el funcionamiento no es el que debiera, ¿qué solución planteas?

¿Por qué crees que es necesario utilizar la señal **relojint** si ya disponemos de un reloj externo (la señal **reloj**)?

Modifica la asignación de patillas y/o el diseño VHDL para que en lugar de los display 7 segmentos de la placa DIGILENT, el resultado se visualice por los display 7 segmentos de la placa de expansión descrita en el inicio de este guión de prácticas.


Ingeniería Electrónica