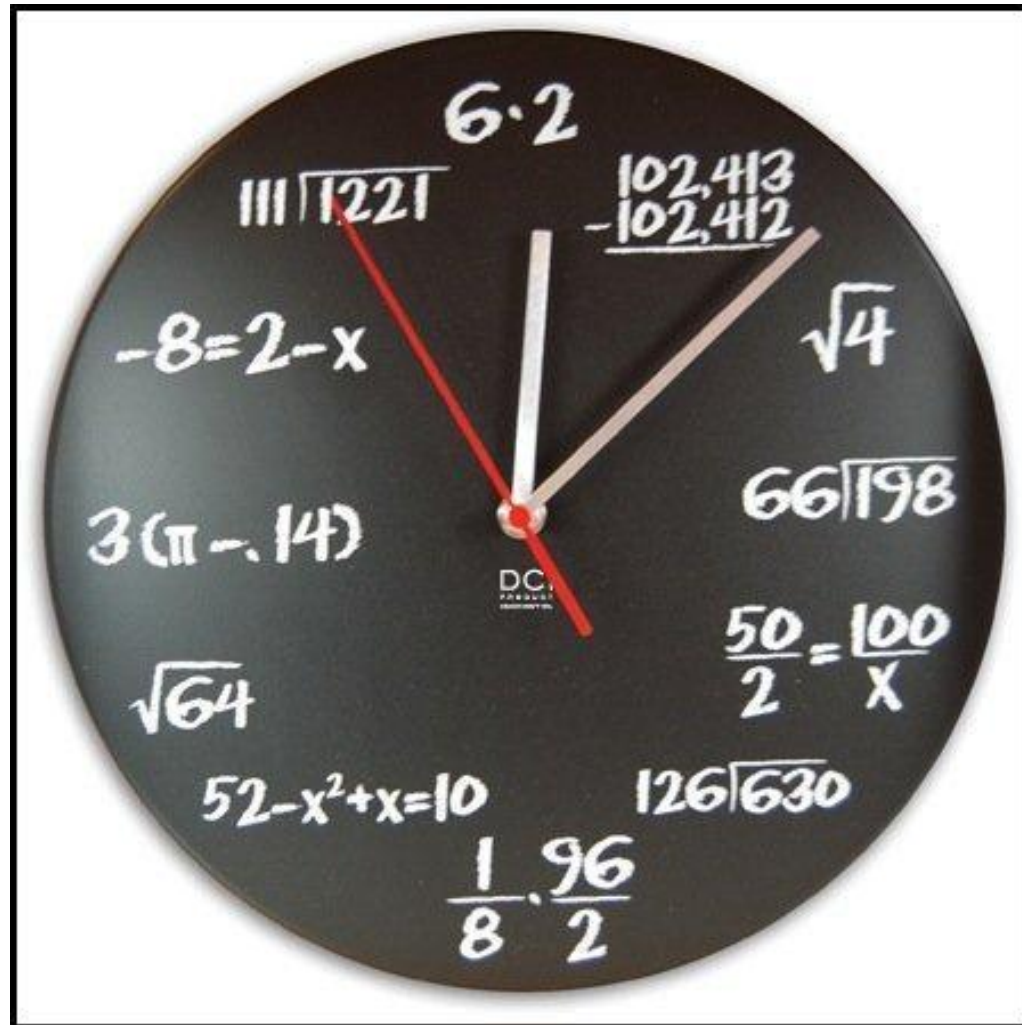
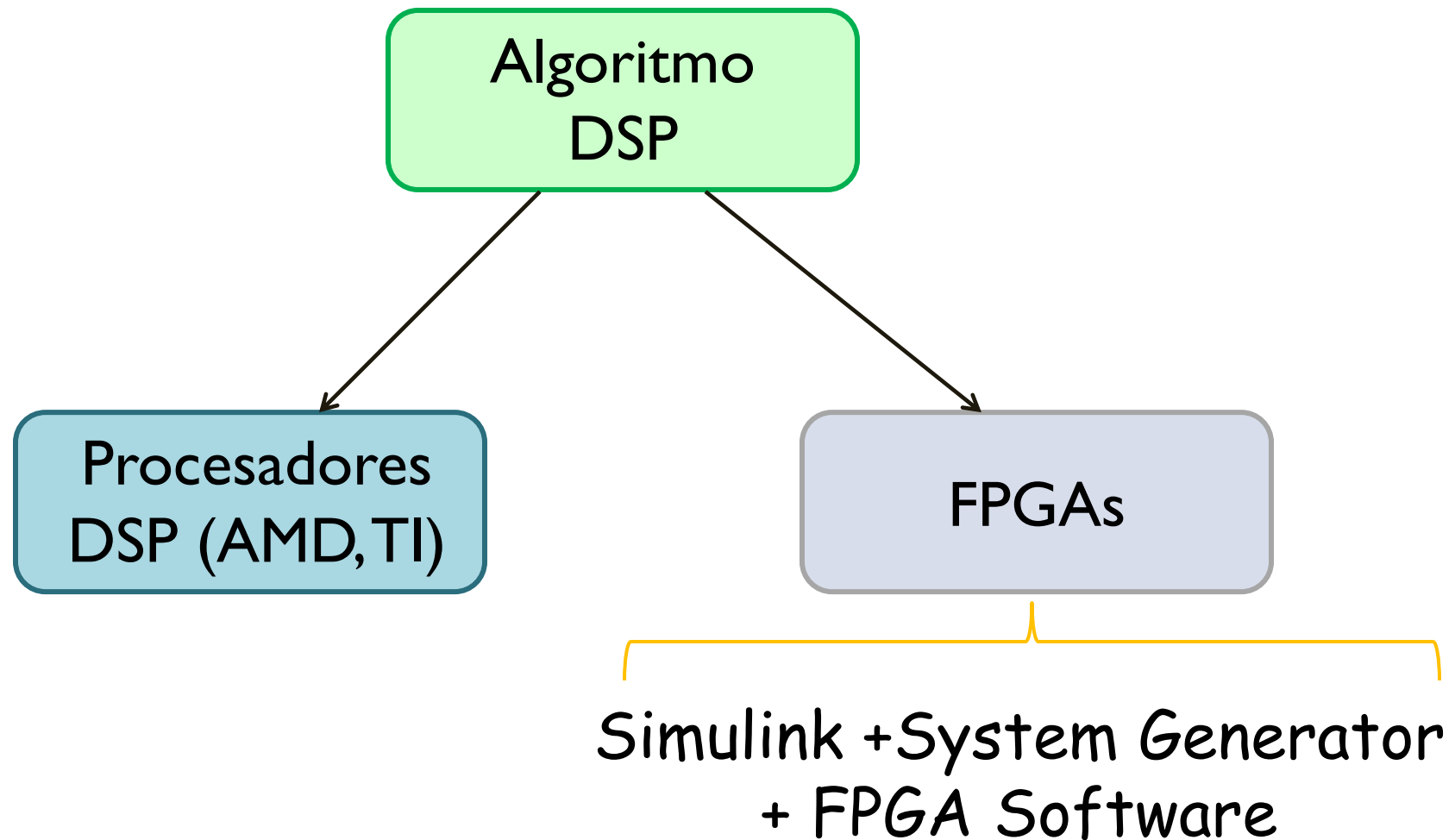


◉ **INTRODUCCIÓN A SIMULINK/SYSTEM GENERATOR (XILINX FPGA)**

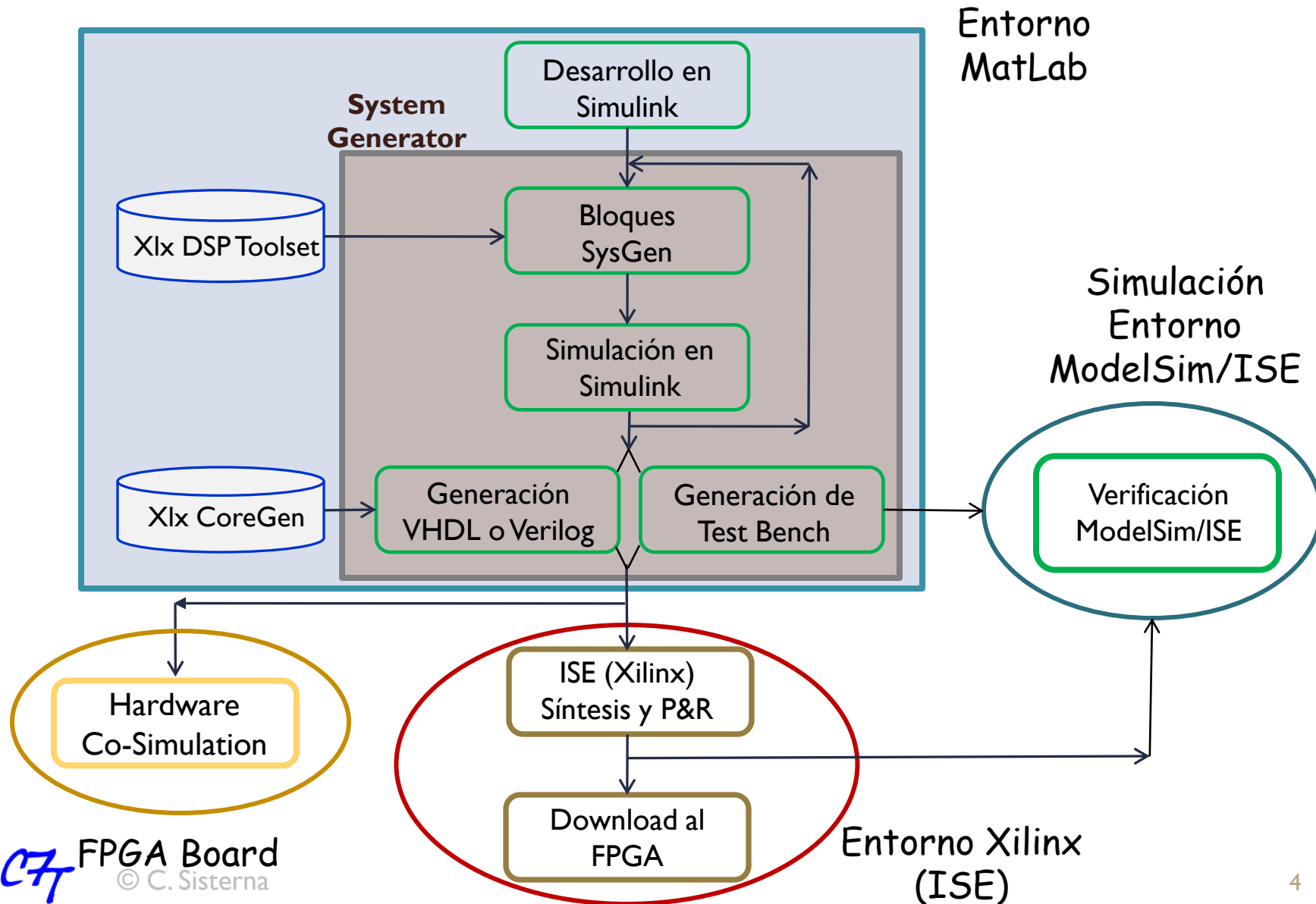
Algoritmos DSP



Implementación Algoritmos DSP



Flujo de Diseño



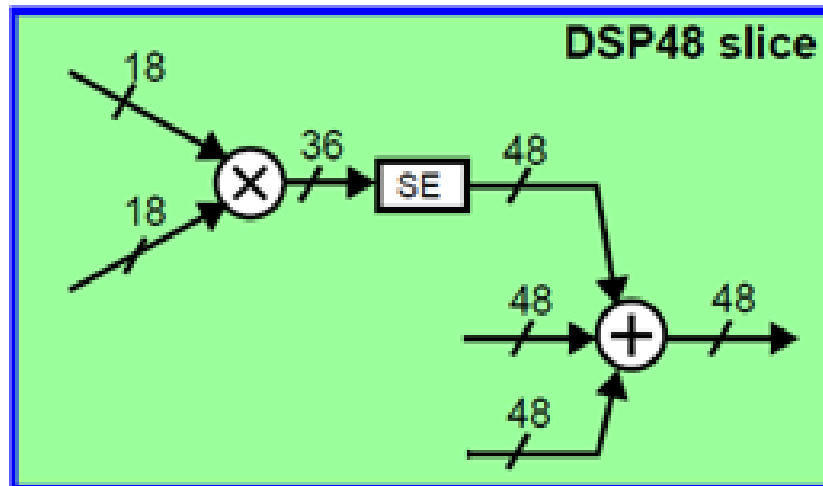
DSP Blocks

Xilinx FPGAs



Xilinx FPGA – Bloque DSP48

- Desde un punto de vista simple el DSP48 puede multiplicar dos números de 18 bits (18×18) y acumular el resultado en un ACC de 48 bits. Todo expresado en complemento a dos.

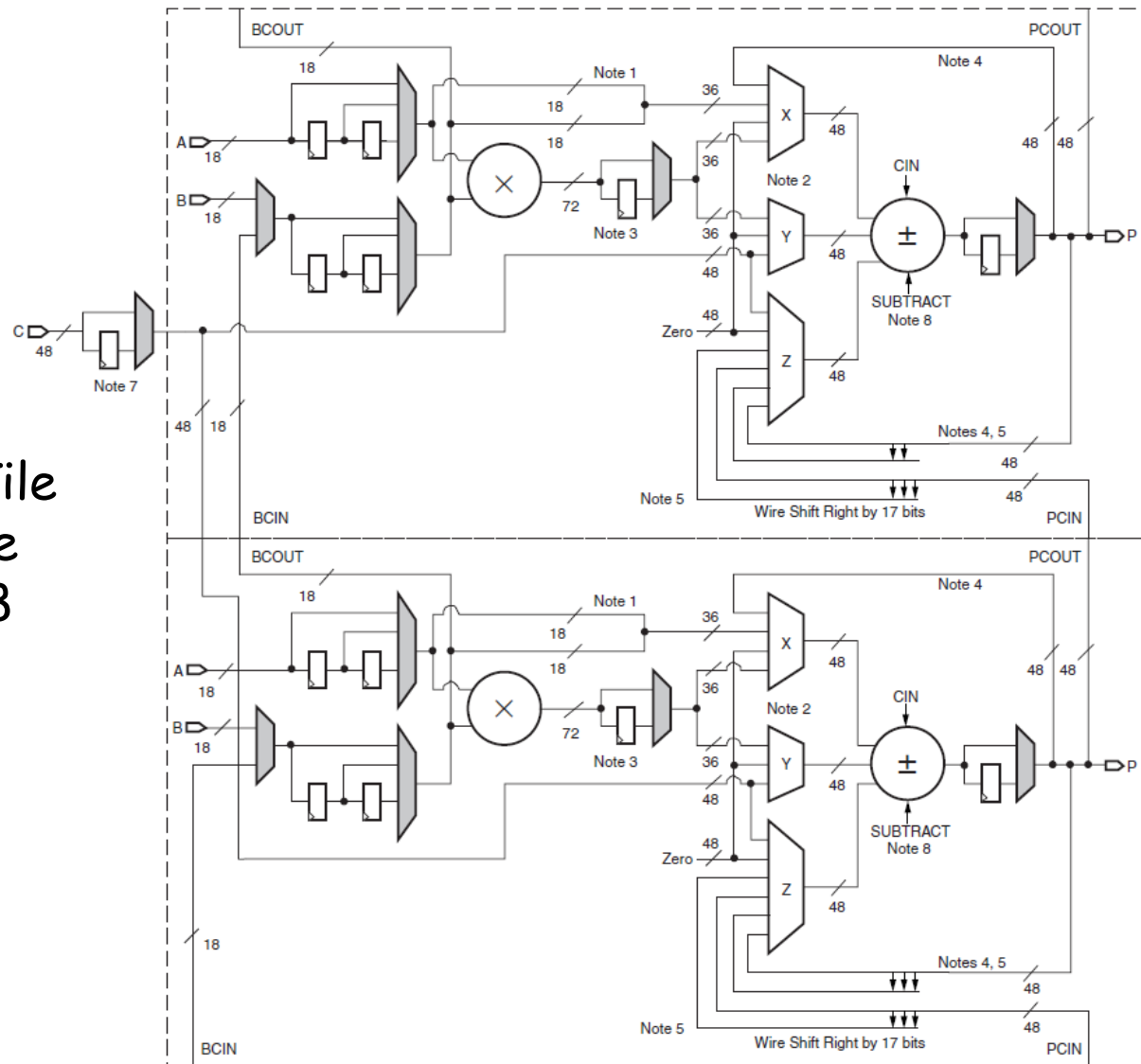


SE - Sign Extension

Nota: Dos DSP48 slices componen lo que se llama un Xtreme DSP Tile

Xilinx DSP48 Tile

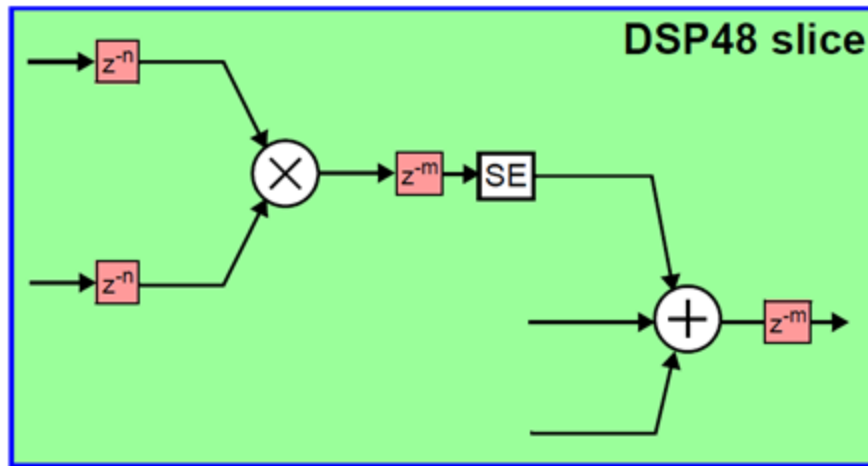
Un DSP48 Tile
consiste de
dos DSP48
Slices



ug073_c1_03_020405

Bloque DSP48 - Registros

Existen diferentes opciones del uso de los registros disponibles en el DSP48



Los registros z^{-n} pueden tener $n = 0, 1$ o 2 para las entradas del multiplicador.

Los registros z^{-m} pueden tener $m = 0$ o 1 .

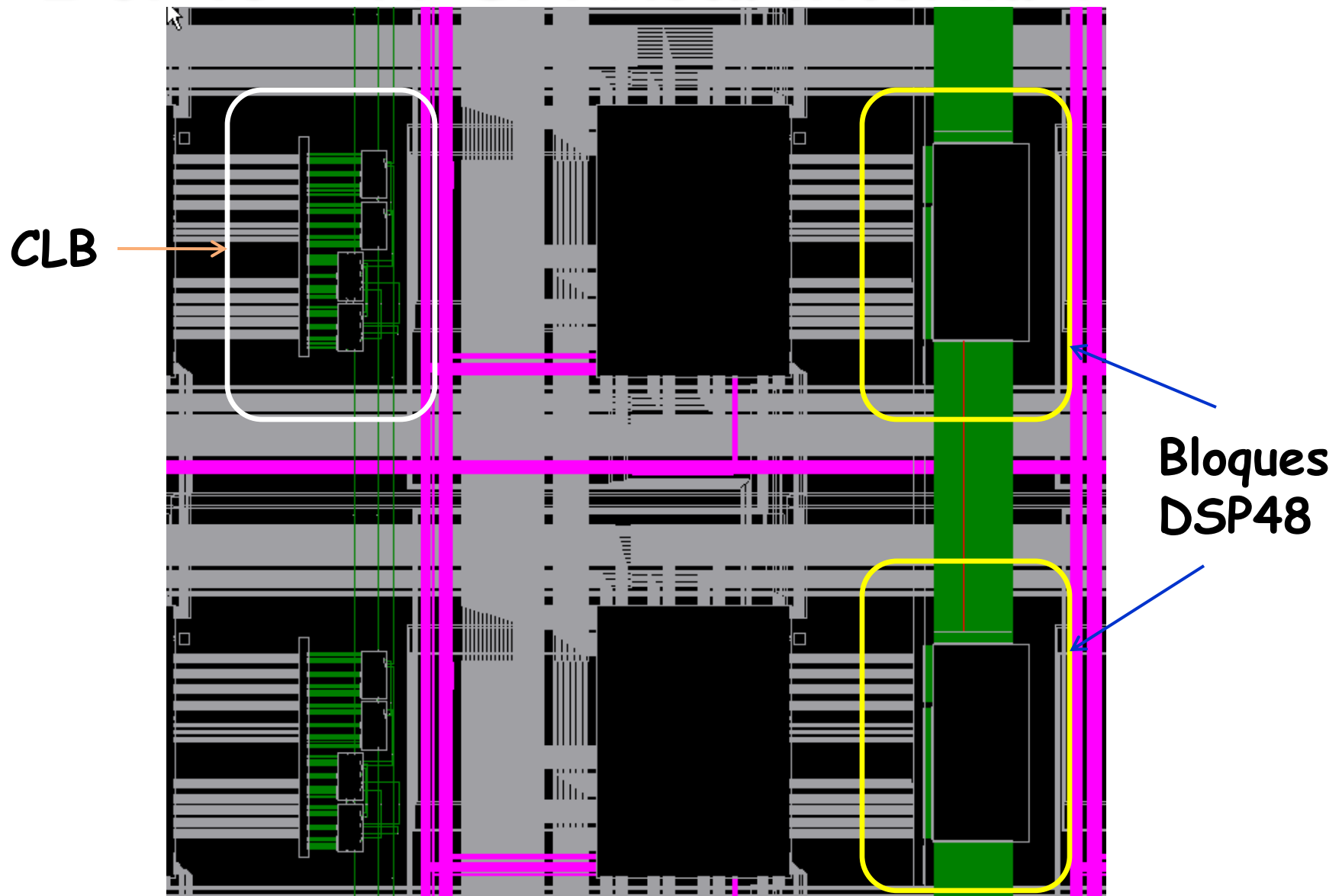
Xilinx FPGA – Bloque DSP48

- El bloque DSP48 es parte de los bloques prediseñados ASMBLs (Application Specific Modular Blocks).
- El bloque DSP48 casi no usa el ruteo del FPGA, solo entrada y salidas. Ello implica: bajo consumo de potencia, muy alta frecuencia de trabajo y una implementación en silicio muy eficiente.
- Bloques DSP48E pueden fácilmente conectarse con sus bloques DSP48 vecinos. Pudiendo formar una cascada de bloques DSP48.

C77



DSP48 – FPGA Vista Interna I



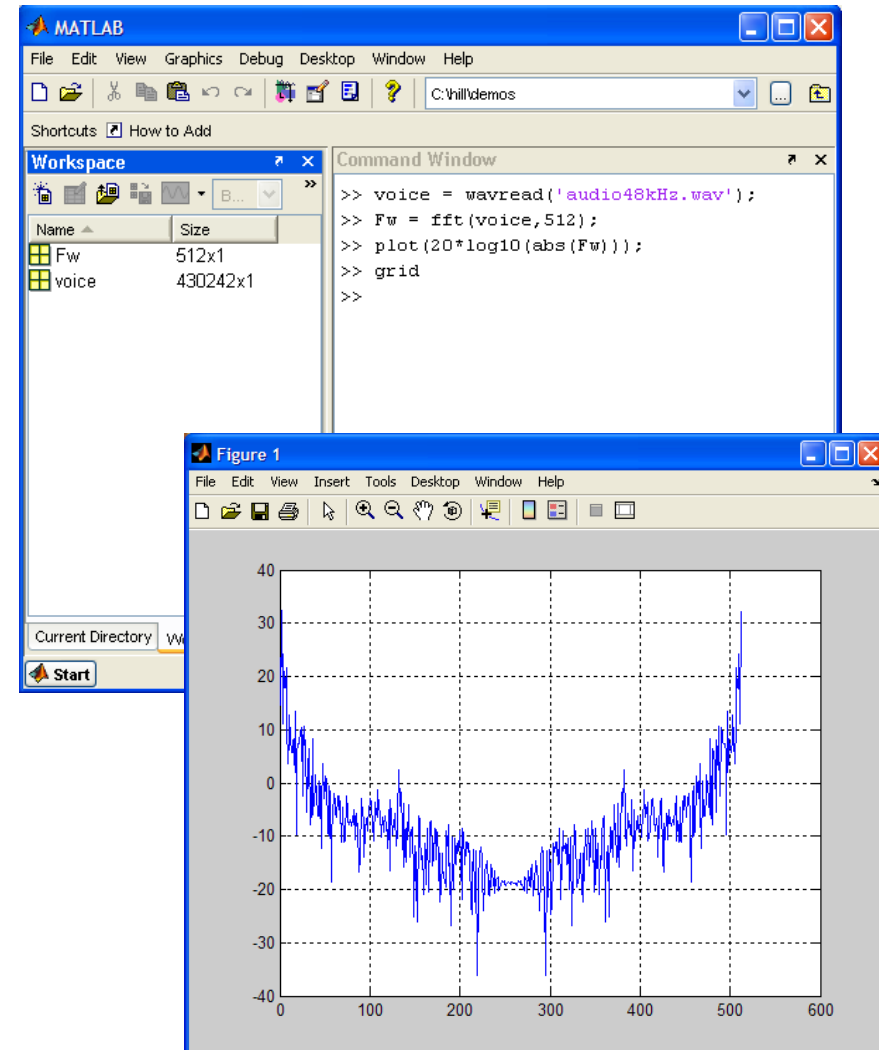
Introducción a MatLab – Simulink



Matlab

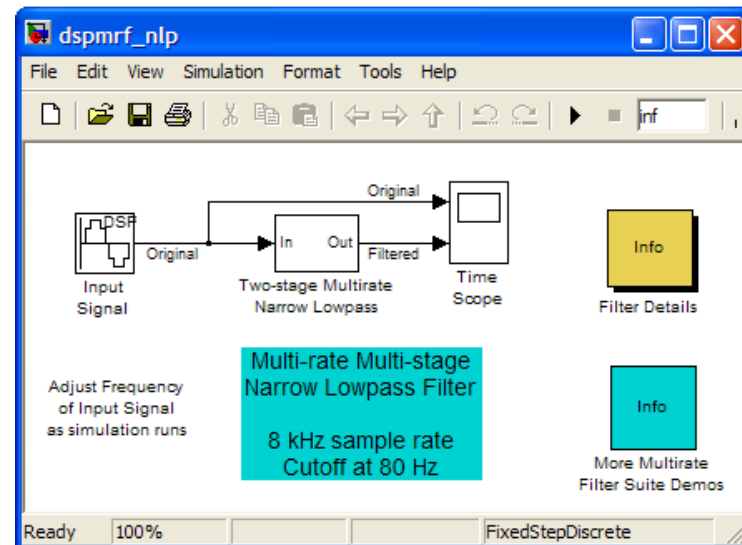


- Matlab provee un entorno de computación técnica que facilita la exploración de soluciones matemáticas a problemas de sistemas:
 - Extensas librerías de funciones matemáticas, procesamiento de señales, comunicaciones, etc.
 - Visualización: gran variedad de funciones para el ploteo y visualización de datos.



Simulink

- Simulink provee un entorno de diseño gráfico para el desarrollo y simulación de sistemas dinámicos:
 - Totalmente integrado a Matlab
 - Editor gráfico
 - Simulador disparado por evento
 - Extensa librería de funciones parametrizadas
 - Bloques: matemática, fuentes, destinos, comunicaciones, DSP, Sistemas de Control, etc.



Simulink

- Visual Data Flow.
- Alternativa al uso de lenguajes de programación ('C').
- Posibilita visualizar la naturaleza dinámica de un sistema.
- **Puede modelar concurrencia en un sistema:**
 - Ejecuta secciones de un sistema en paralelo.
 - Similar a la concurrencia en HDLs

Fuentes de Señales en Simulink

- Algunos de las fuentes de señales mas usadas en diseños DSP son:
 - White-Noise (band limited)
 - Constante
 - Rampa
 - Onda Senoidal
 - Función Escalón
 - Generador de Pulsos
 - Contador/Contador Libre
 - Datos binarios de un archivo
 - Datos proveniente del espacio MatLab

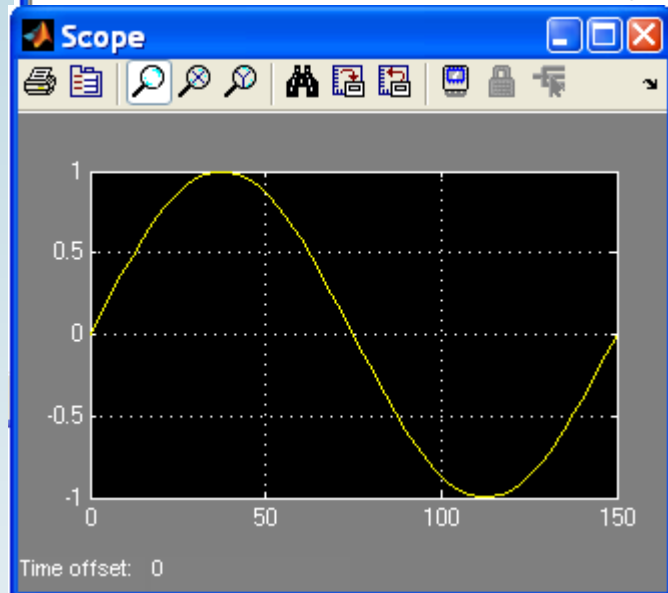
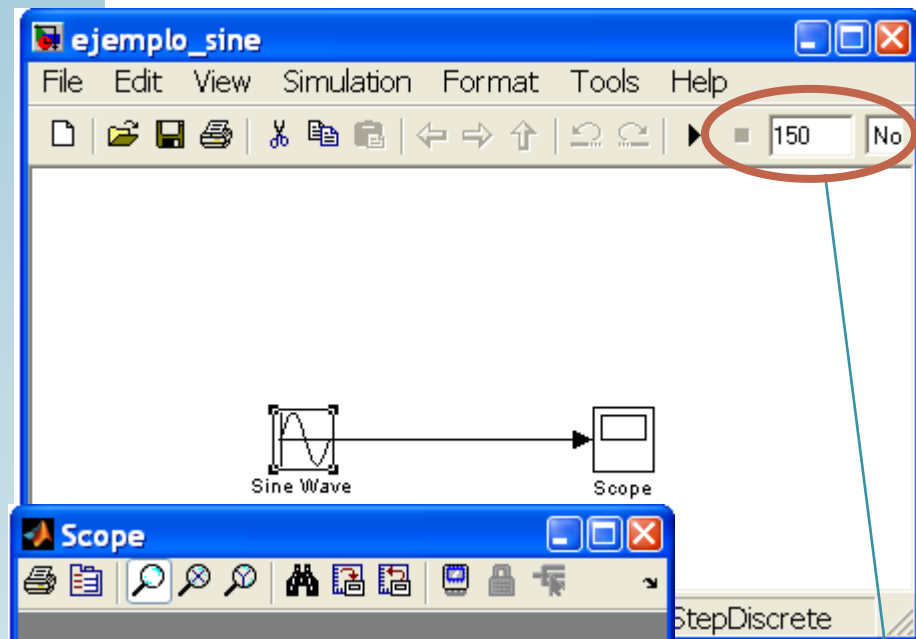
Periodo de Muestreo

- Cada señal usada en un diseño DSP en Simulink debe ser *muestreada* cada cierto tiempo llamado *periodo de muestreo*.
- Si bien el periodo de muestreo puede ser *arbitrario*, los diferentes bloques disponibles en Simulink usan un periodo de muestreo.
- Un periodo de muestreo de 1/1000 (0.001) indica que la función del bloque se ejecutará cada 0.001 segundo y producirá la respectiva salida.
- Recordar el Teorema de Nyquist:
 - $F_m \geq 2f_{\max}$

Periodo de Muestreo (cont.)

- El periodo de muestreo de un bloque tiene relación directa con la frecuencia del reloj del hardware que implementará el bloque
- El periodo de muestreo debe ser fijado sí o sí en:
 - Bloque denominado 'Gateway In'
 - Bloque sin entradas
- En los bloques que no es especificado, el periodo de muestreo es derivado del periodo de muestreo de las entradas al mismo.

Ejemplo Simulink I



Source Block Parameters: Sine Wave

Sine Wave

Output a sine wave:

$O(t) = \text{Amp} * \sin(\text{Freq} * t + \text{Phase}) + \text{Bias}$

Sine type determines the computational technique used. The parameters in the two types are related through:

Samples per period = $2 * \pi / (\text{Frequency} * \text{Sample time})$

Number of offset samples = $\text{Phase} * \text{Samples per period} / (2 * \pi)$

Use the sample-based sine type if numerical problems due to running for large times (e.g. overflow in absolute time) occur.

Parameters

Sine type: Time based

Time (t): Use simulation time

Amplitude: 1

Bias: 0

Frequency (rad/sec): $2 * \pi * (1/150)$

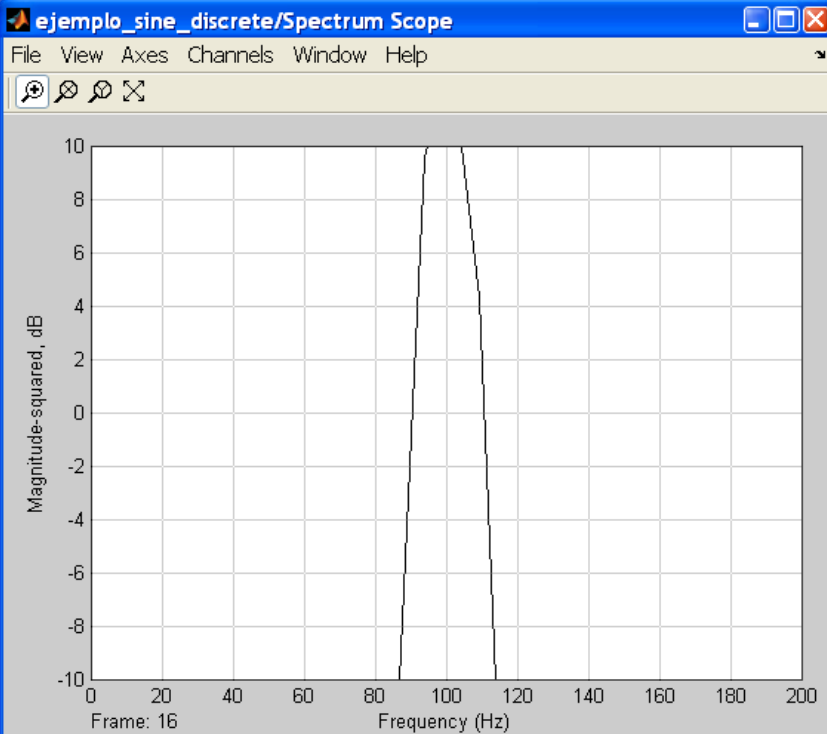
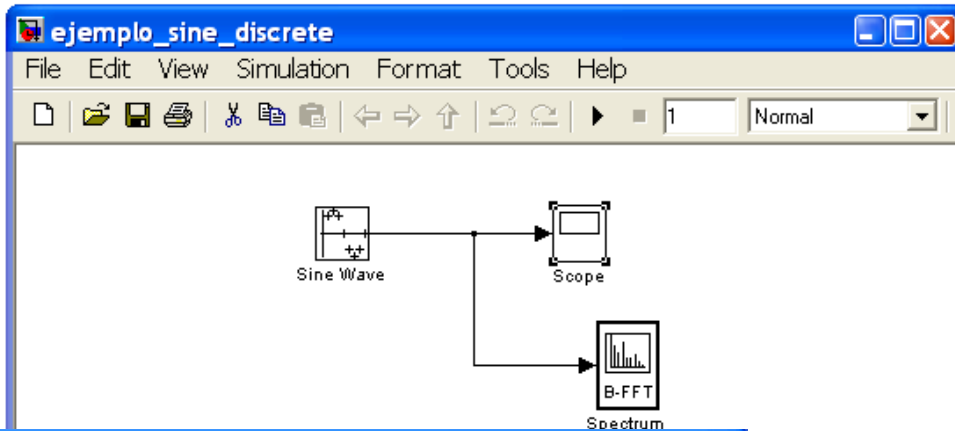
Phase (rad): 0

Sample time: 0

☒ Interpret vector parameters as 1-D

OK Cancel Help

Ejemplo Simulink 2



Source Block Parameters: Sine Wave

Sine Wave

Output a sine wave:

$$O(t) = \text{Amp} * \sin(\text{Freq} * t + \text{Phase}) + \text{Bias}$$

Sine type determines the computational technique used. The parameters in the two types are related through:

Samples per period = $2 * \pi / (\text{Frequency} * \text{Sample time})$

Number of offset samples = $\text{Phase} * \text{Samples per period} / (2 * \pi)$

Use the sample-based sine type if numerical problems due to running for large times (e.g. overflow in absolute time) occur.

Parameters

Sine type: **Sample based**

Time (t): **Use simulation time**

Amplitude: **1**

Bias: **0**

Samples per period: **10**

Number of offset samples: **0**

Sample time: **1/1000**

☒ Interpret vector parameters as 1-D

OK Cancel Help

100Hz



SYSTEM GENERATOR

System Generator

- Flujo de diseño integrado a Simulink, generando directamente el archivo de configuración del FPGA (.bit)
- Integra:
 - MatLab, Simulink
 - HDL Síntesis
 - Librerías de DSP
 - Herramientas de implementación del FPGA
 - Simulación en doble precisión y punto fijo
- Abstracción Aritmética
 - Punto-fijo arbitrario, incluyendo cuantización y overflow

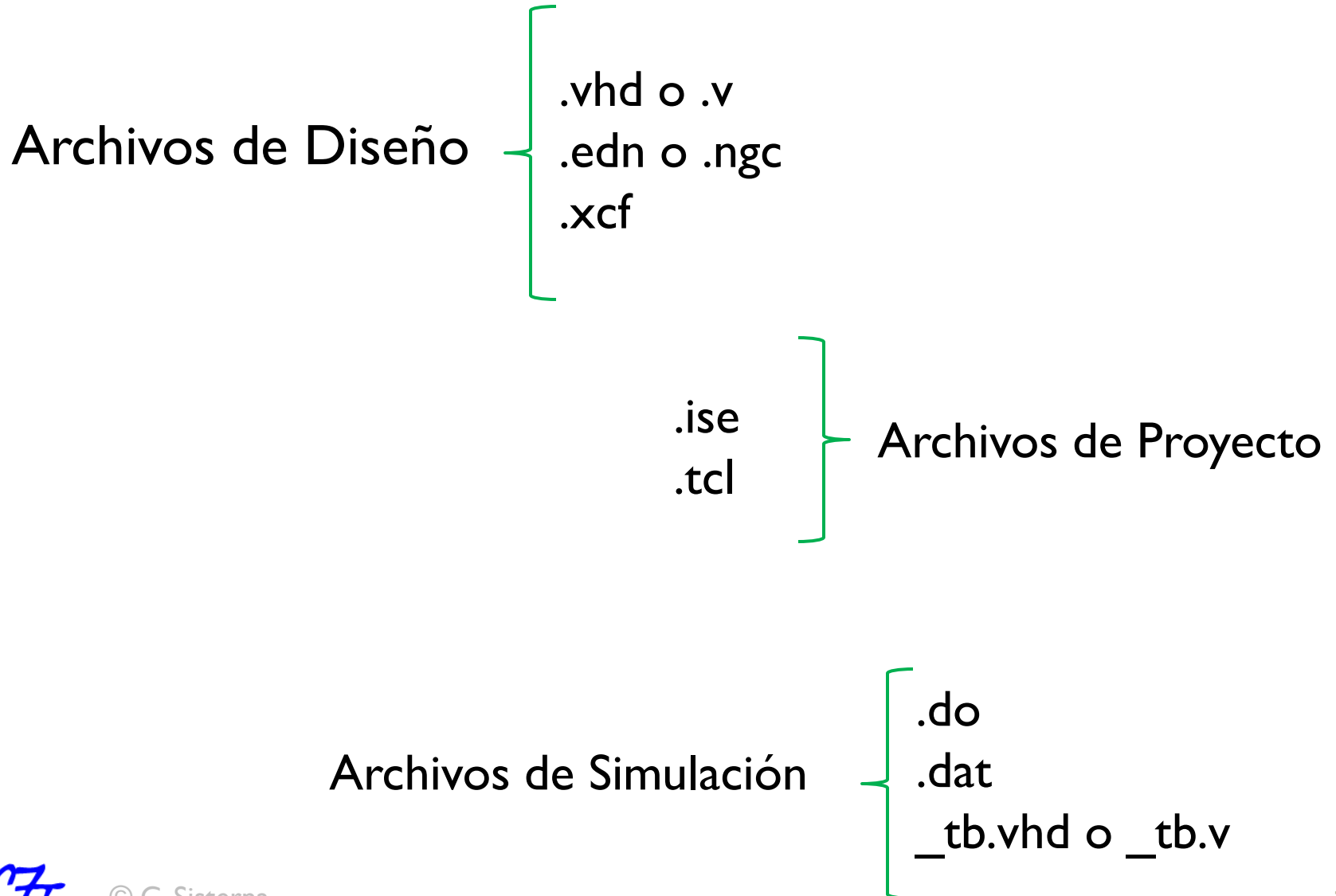
System Generator

- Es un 'Toolbox' plug-in al entorno Simulink.
- Provee:
 - Bloques de distintas funciones implementables en FPGA
 - Generación de archivos VHDL
 - Simulación
 - IP Cores (48, hasta ahora)
 - Test benches

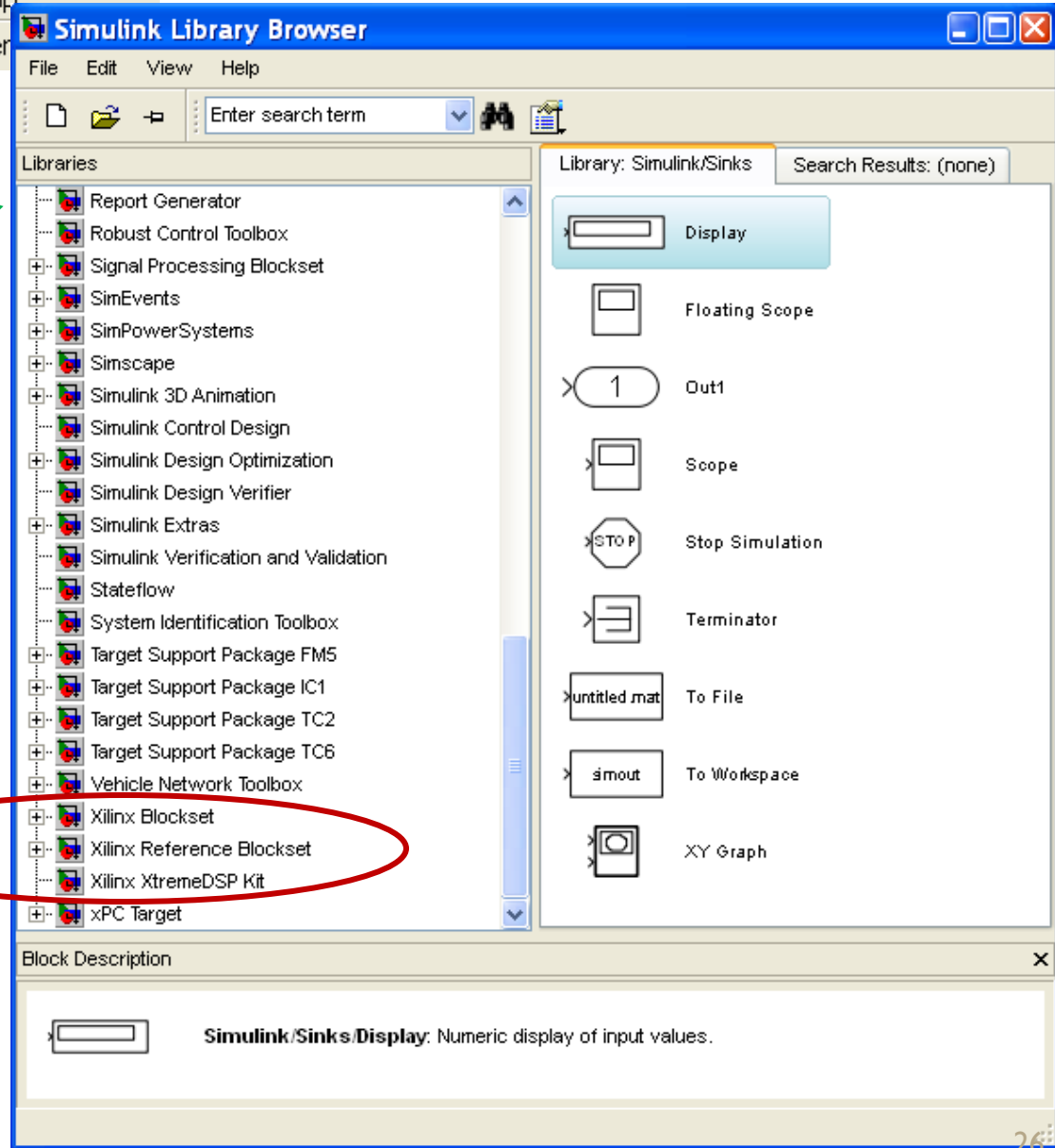
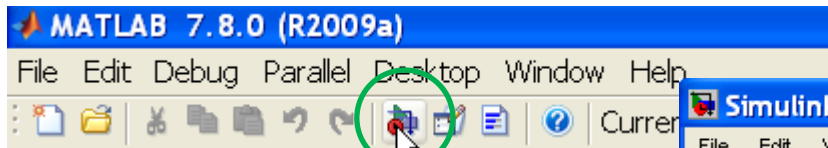
Generación de Archivos FPGA

- Generación del código VHDL y Verilog para:
 - Virtex-7, Virtex-6, Virtex-5, Virtex-4, Spartan-7, Spartan-6, Spartan-3E
- Mapeo y expansión del hardware respectivo a cada bloque
- Preservación de la jerarquía del modelo una vez generados los códigos VHDL y Verilog
- Utilización de IP cores de CoreGenerator
- Generación automática de:
 - Un proyecto ISE
 - Archivo de restricciones (constraint file *.ucf)
 - Archivo de simulación Test Bench y script *.do (ModelSim)

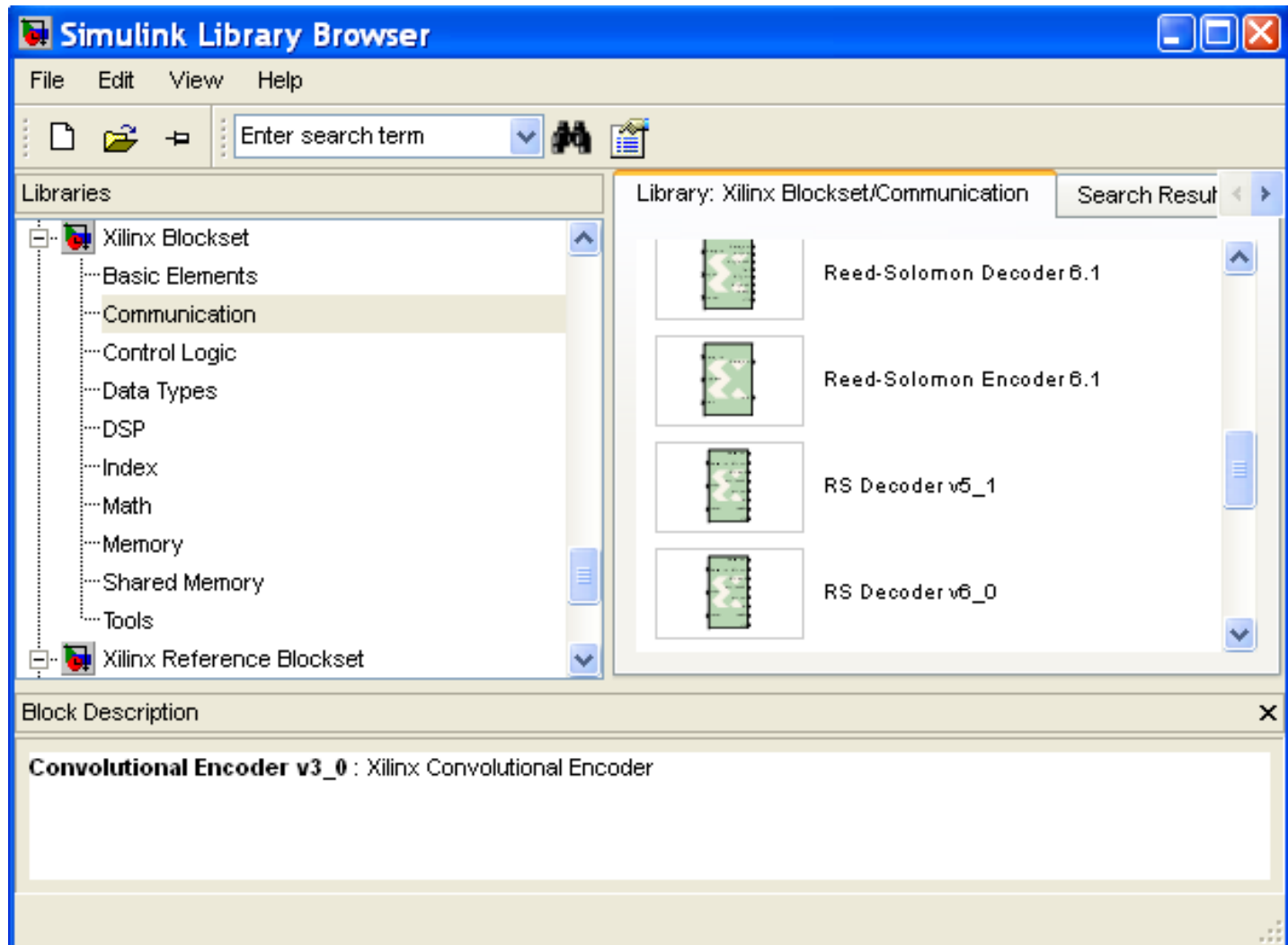
SysGen – Archivos Generados



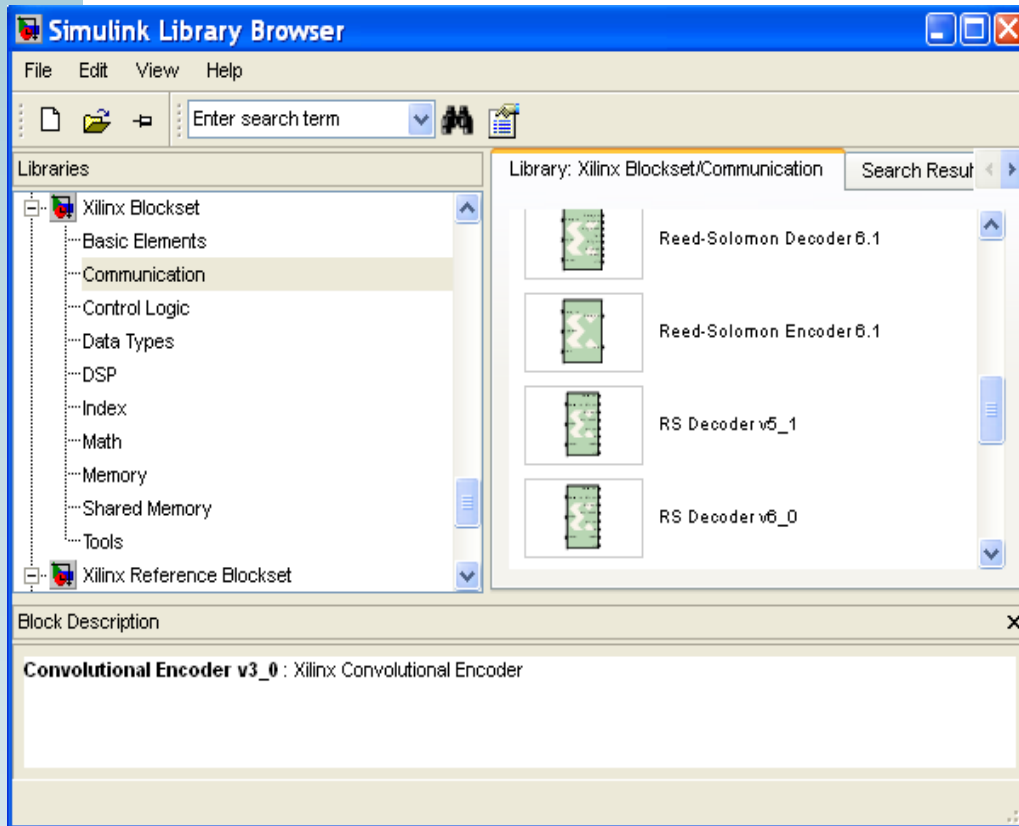
Simulink Librería



Librería de Xilinx

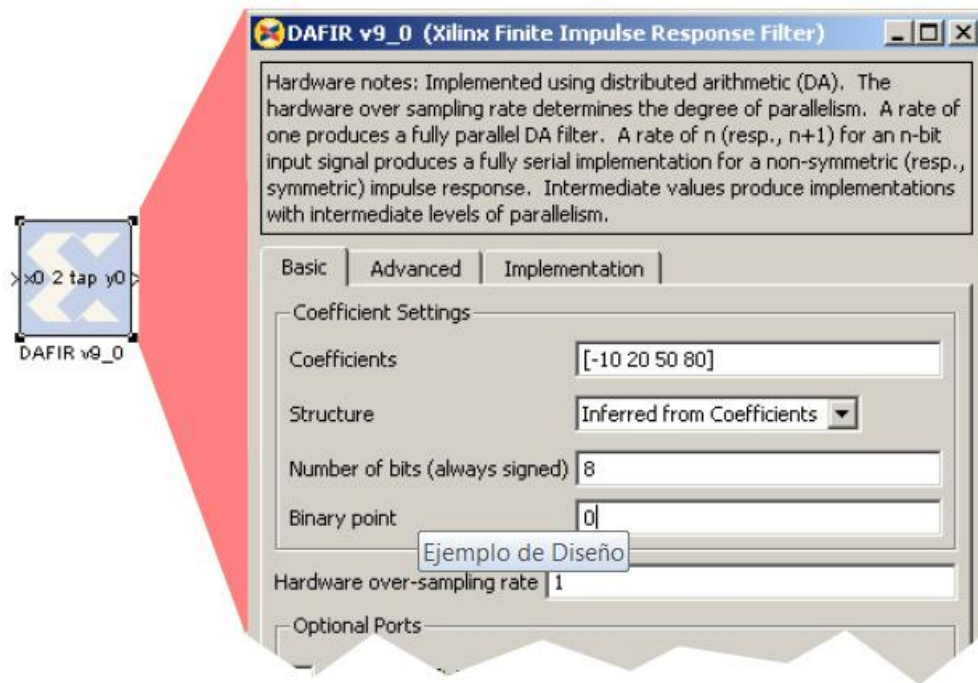


Librería de Xilinx

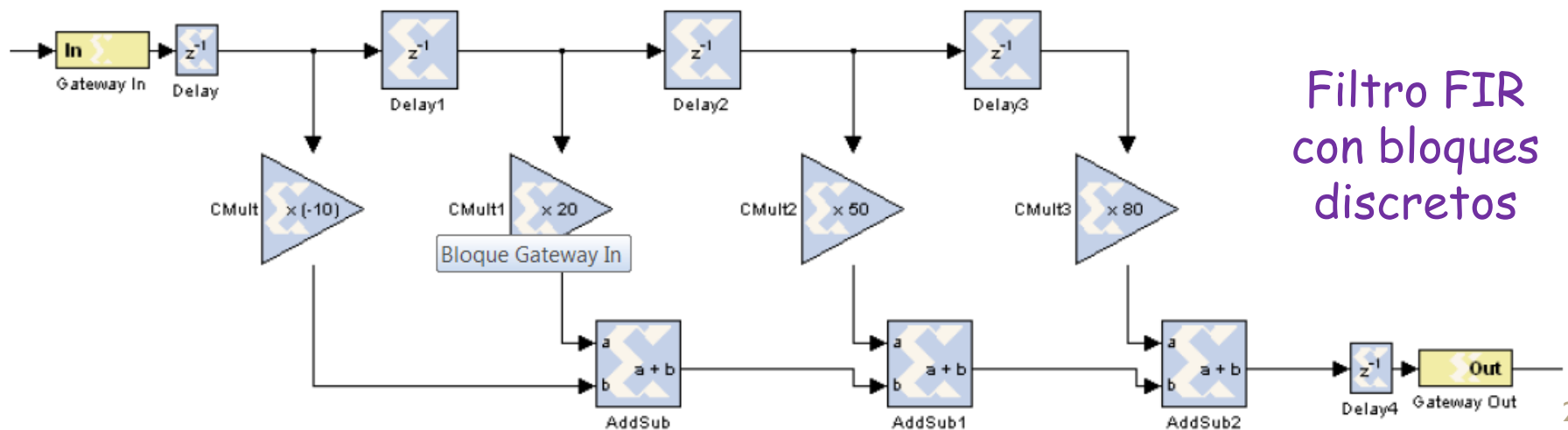


- Elementos básicos: contadores, retardos, lógica binaria, etc.
- Comunicaciones: ECC bloques, Viterbi, Reed Solomon, etc.
- Tipo de Datos: convertir, rotar, concatenar, etc.
- DSP: IIR, FIR, FFT, etc.
- Math: multiplicar, acumular, invertir, contadores, etc.
- Memoria, dual port RAM, single port RAM, ROM, FIFO.
- Tools: ModelSim, Estimador de Recursos, SysGen Token

Xilinx Librería



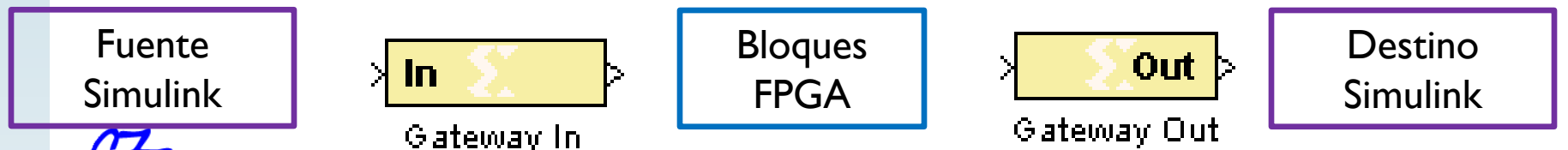
Bloque de Filtro FIR disponible en la librería



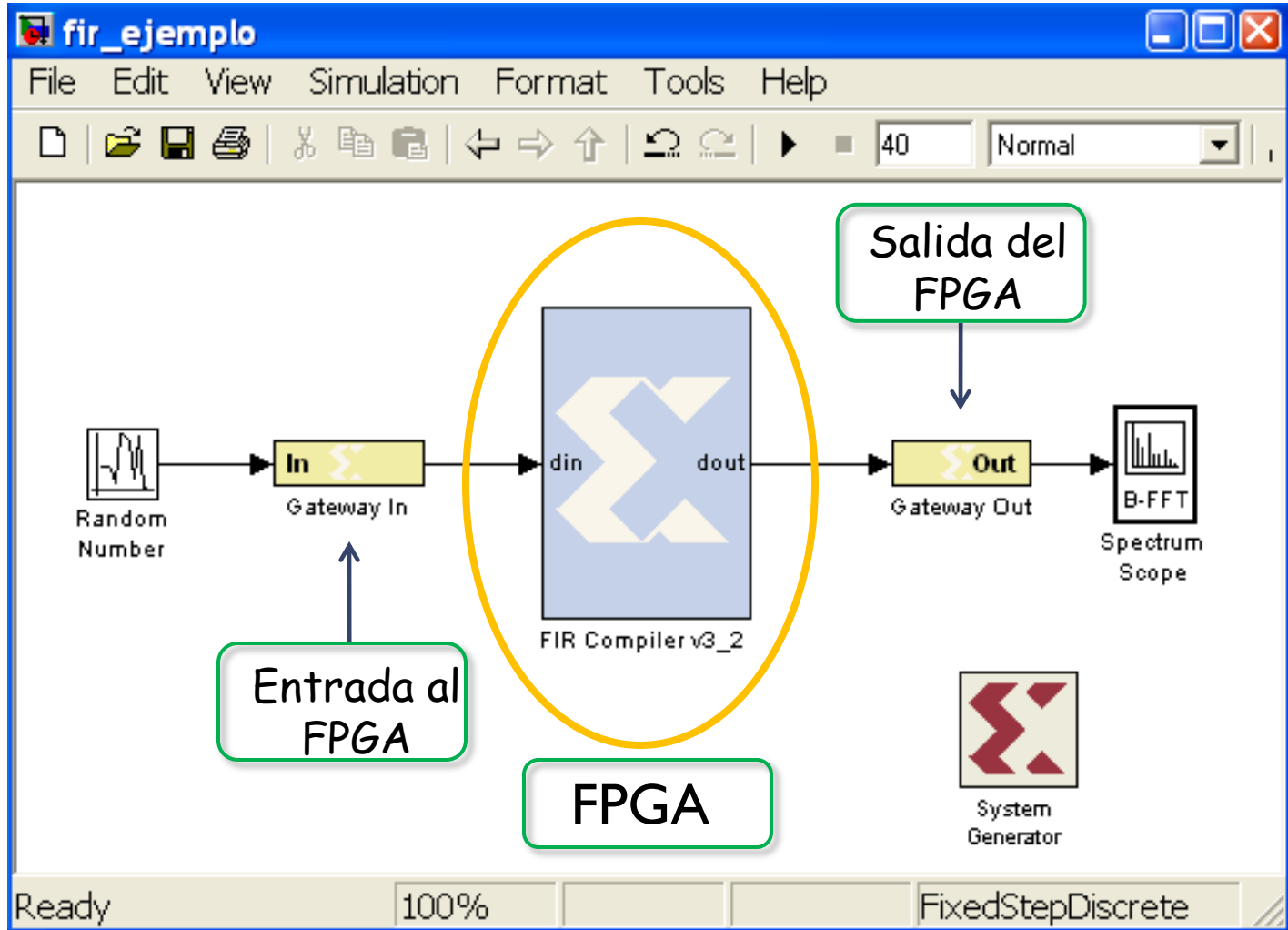
Filtro FIR con bloques discretos

Interface Simulink-FPGA

- Simulink usa 'double' para representar números en simulación. A double es un número en punto flotante en complemento a dos de 64 bits.
- TODOS los bloques de Xilinx usan representación en n-bits en punto fijo (complemento a dos es opcional).
- Por ello se usa un bloque para la conversión de datos desde Simulink a un bloque de Xilinx, llamado *Gateway In*. Y otro bloque para la conversión desde un bloque de Xilinx a Simulink, llamado *Gateway Out*.



Ejemplo de Diseño

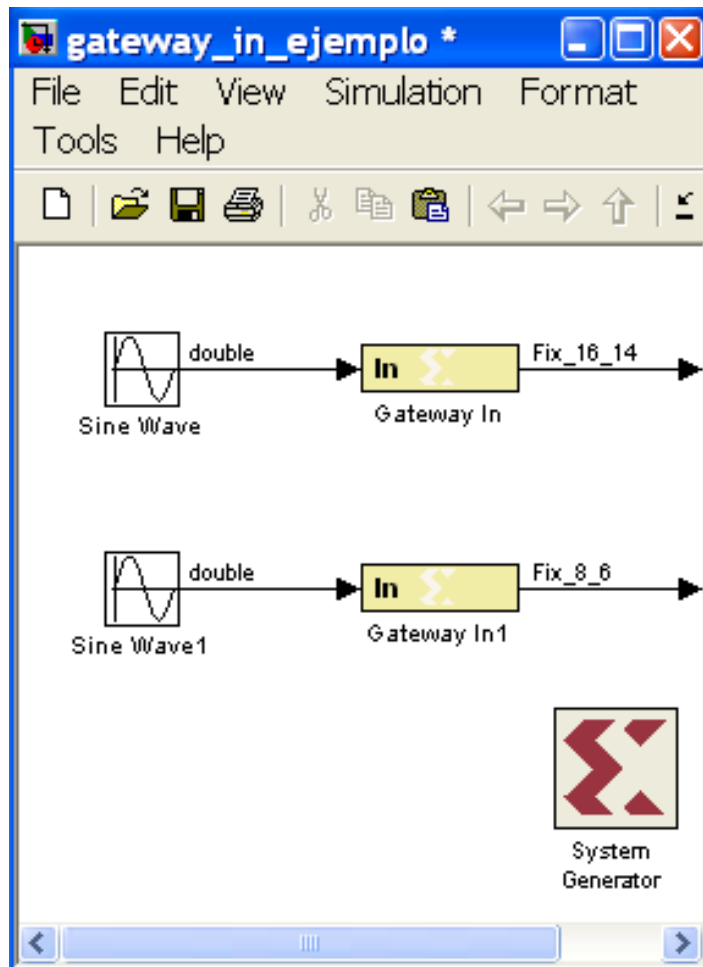


Bloque Gateway In



- Controla la conversión desde un número representado en double, entero o punto fijo a un número booleano de N-bits, que puede ser con signo (complemento a dos) o sin signo, con o sin punto fijo.
- Presenta la opción de manejar los extra bits durante la conversión (overflow).
- Este bloque define cuales van a ser las entradas del diseño codificado en HDL por SysGen.
- Define los estímulos en caso de que la opción de 'Create Testbench' haya sido seleccionada.
- Nombra los correspondientes puertos de entrada de la entidad (entity) generada por System Generator.

Bloque Gateway In



The screenshot shows the configuration dialog for the 'Gateway In1 (Xilinx Gateway In)' block. The dialog has two tabs: 'Basic' and 'Implementation'. The 'Basic' tab is selected.

Gateway in block. Converts inputs of type Simulink integer, double and fixed point to Xilinx fixed point type.

Hardware notes: In hardware these blocks become top level input ports.

Basic | Implementation

Output type:
☐ Boolean ☒ Signed (2's comp) ☐ Unsigned

Number of bits: 8

Binary point: 6

Quantization:
☐ Truncate ☒ Round (unbiased: +/- Inf)

Overflow:
☐ Wrap ☒ Saturate ☐ Flag as error

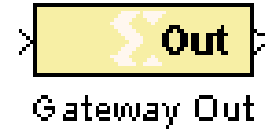
Sample period: 1

Simulation

☐ Override with doubles

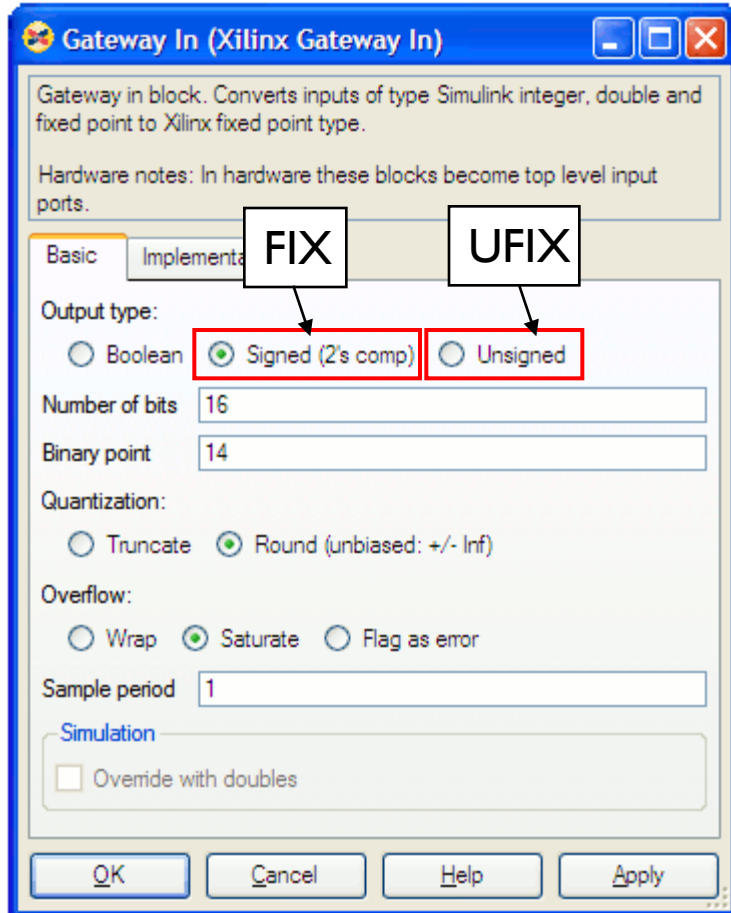
OK Cancel Help Apply

Bloque Gateway Out



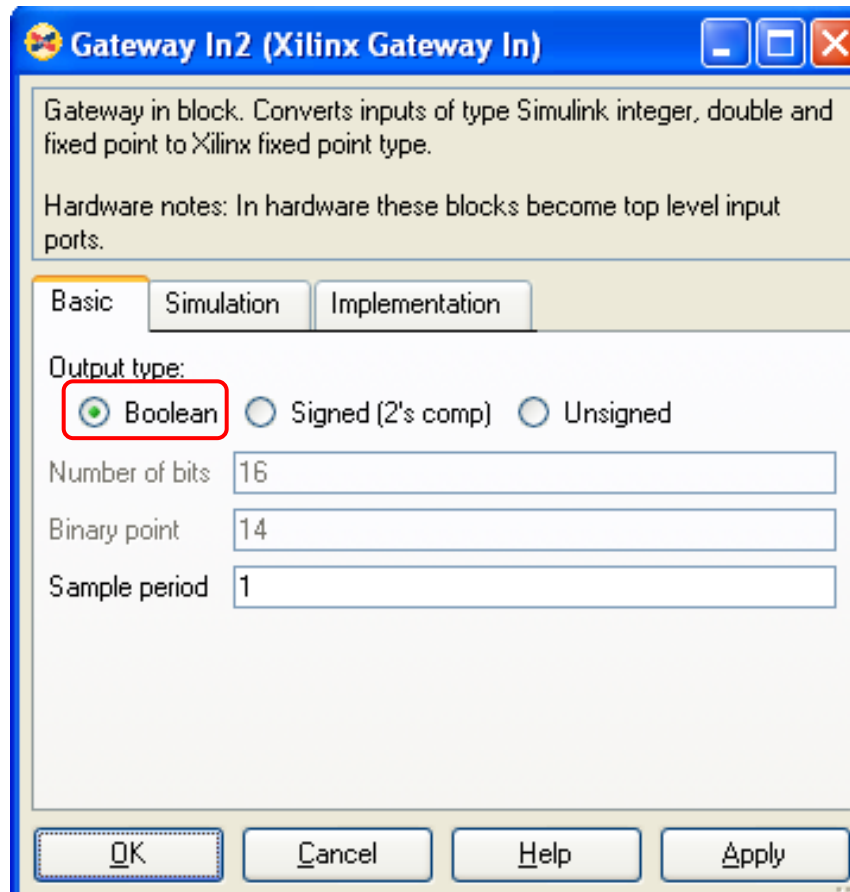
- Convierte los datos de punto fijo del sistema en el FPGA a punto flotante Simulink.
- Define cuales van a ser los puertos de salida del sistema generado por System Generator.
- Nombra los correspondientes puertos de salida de la entidad (entity) generada por System Generator.

SysGen – Tipo de Datos



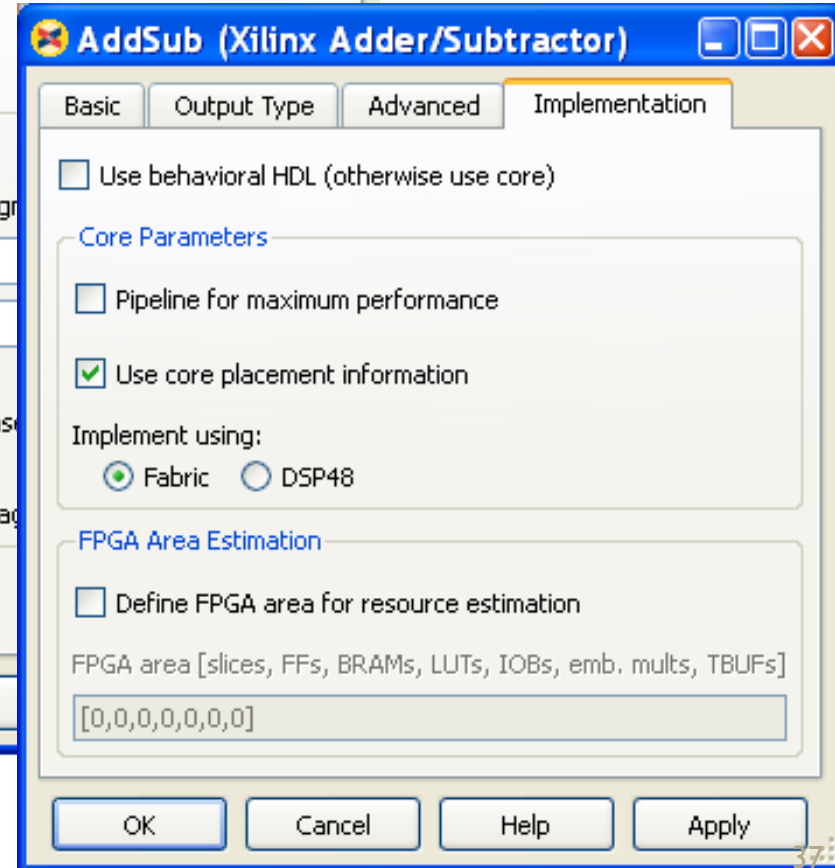
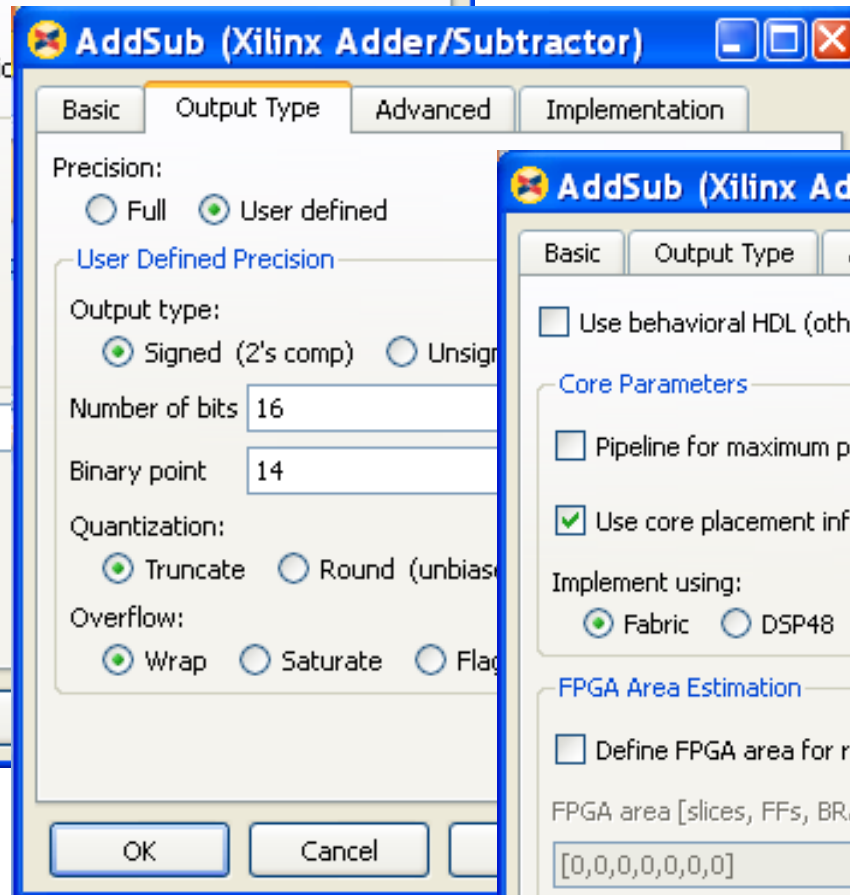
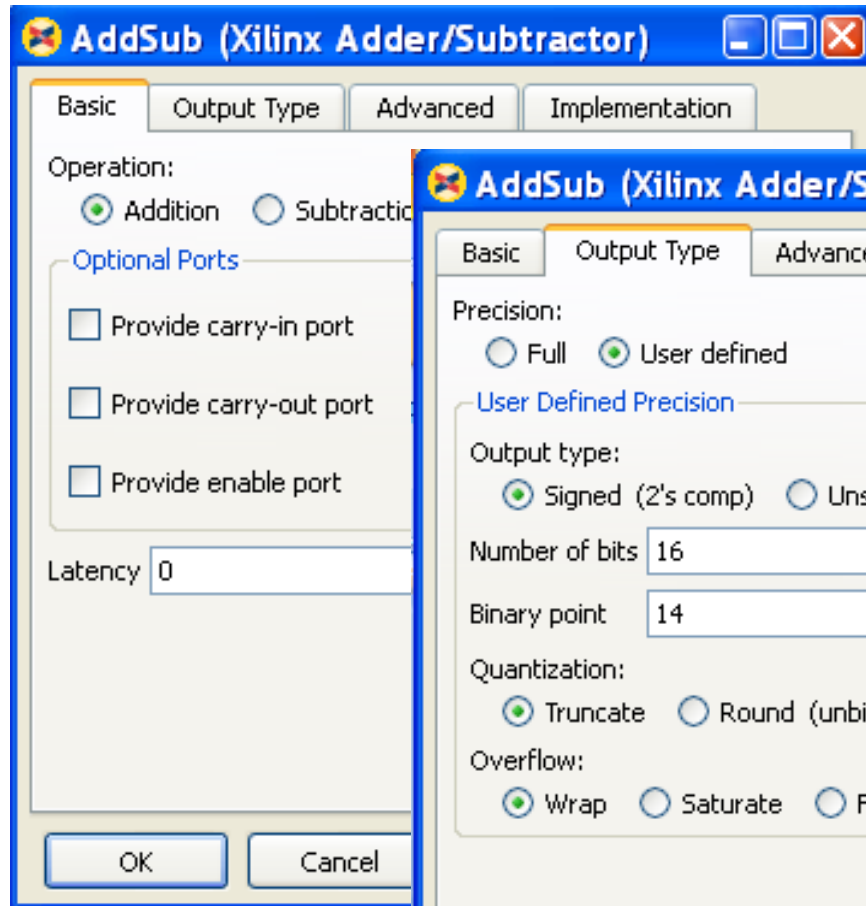
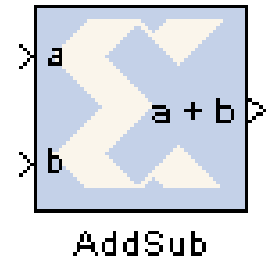
- **FIX**: tipo de dato fijo, es un número en punto fijo representado en complemento a dos.
- **UFIX**: es un número en punto fijo representado sin signo.

SysGen – Tipo de Datos

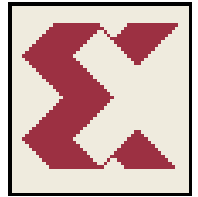


- **Boolean:** pueda tomar valores '1' o '0'. Se usa para controlar señales como LOAD, CS, RESET, etc.

SysGen – Parámetros de un Bloque



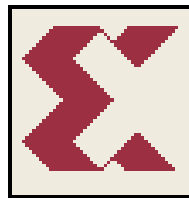
Bloque System Generator



System
Generator

- **Cada** diagrama usando bloques de Xilinx in Simulink requiere que el bloque System Generator sea colocado en el diagrama.
- El bloque **no** se conecta a nada. Pero controla el proceso de generación del código HDL y de la implementación del sistema.
- Parámetros del bloque SysGen determinan FPGA a usar, código HDL, frecuencia objetivo, directorio de trabajo, etc.

Bloque System Generator



System
Generator

System Generator: counter_enabled

Xilinx System Generator

Compilation :
 Settings...

Part :

Target Directory :

Synthesis Tool :

Hardware Description Language :

FPGA Clock Period (ns) :

Clock Pin Location :

☐ Create Testbench

☐ Import as Configurable Subsystem

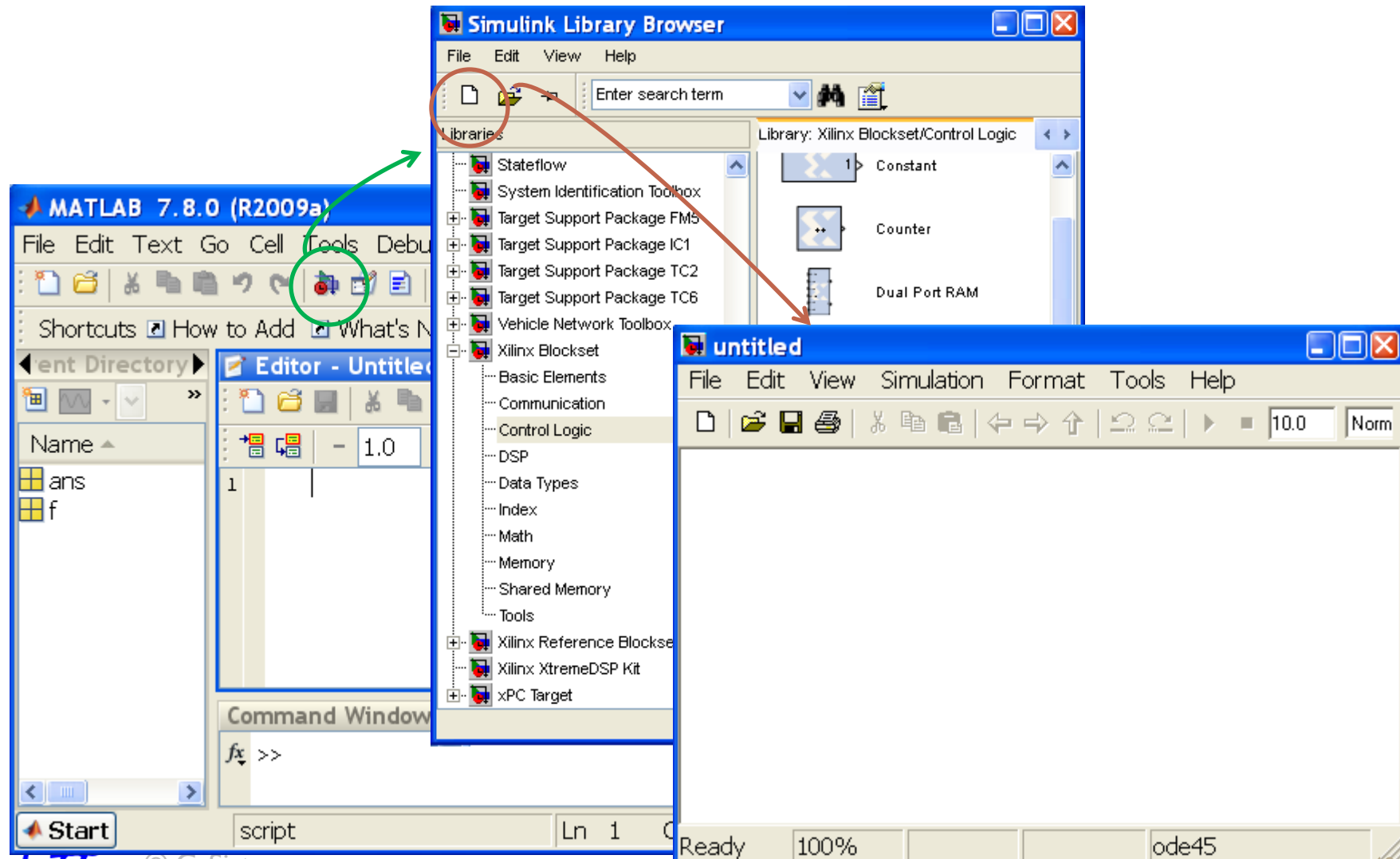
☐ Provide clock enable clear pin

Override with Doubles :

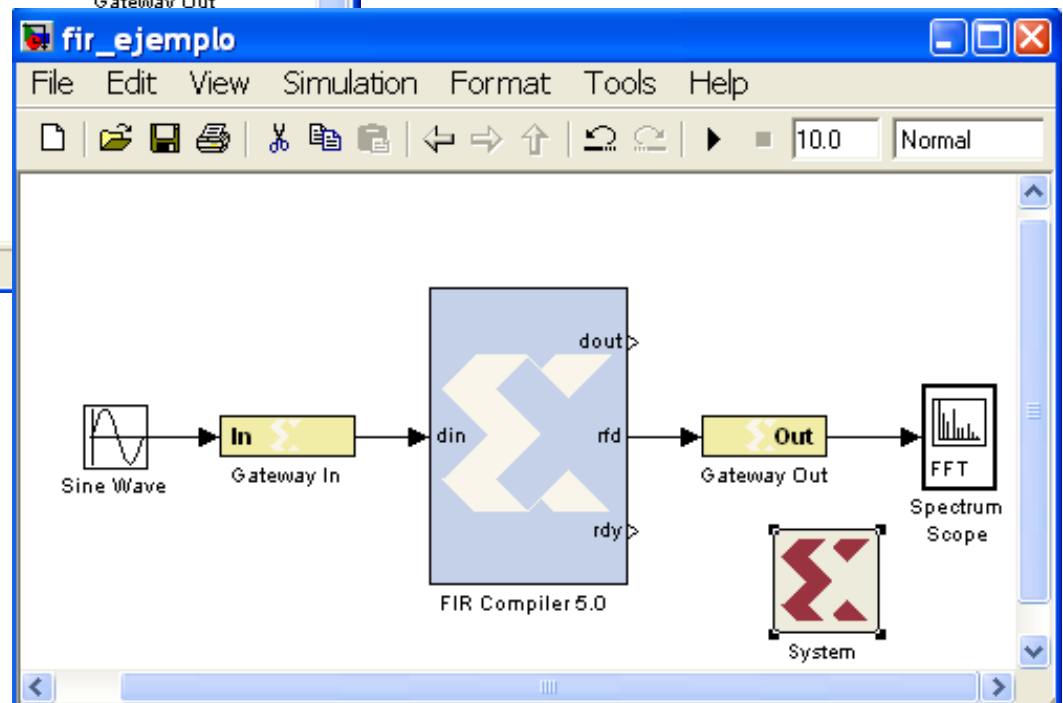
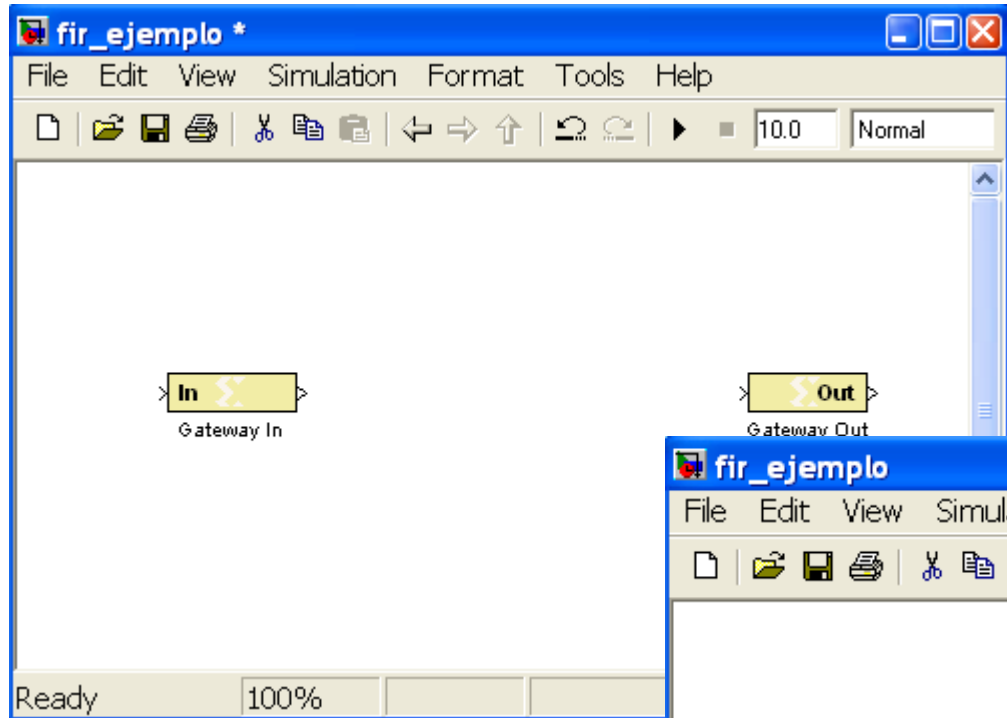
Simulink System Period (sec) :

Block Icon Display:

Creando un Diseño en Simulink

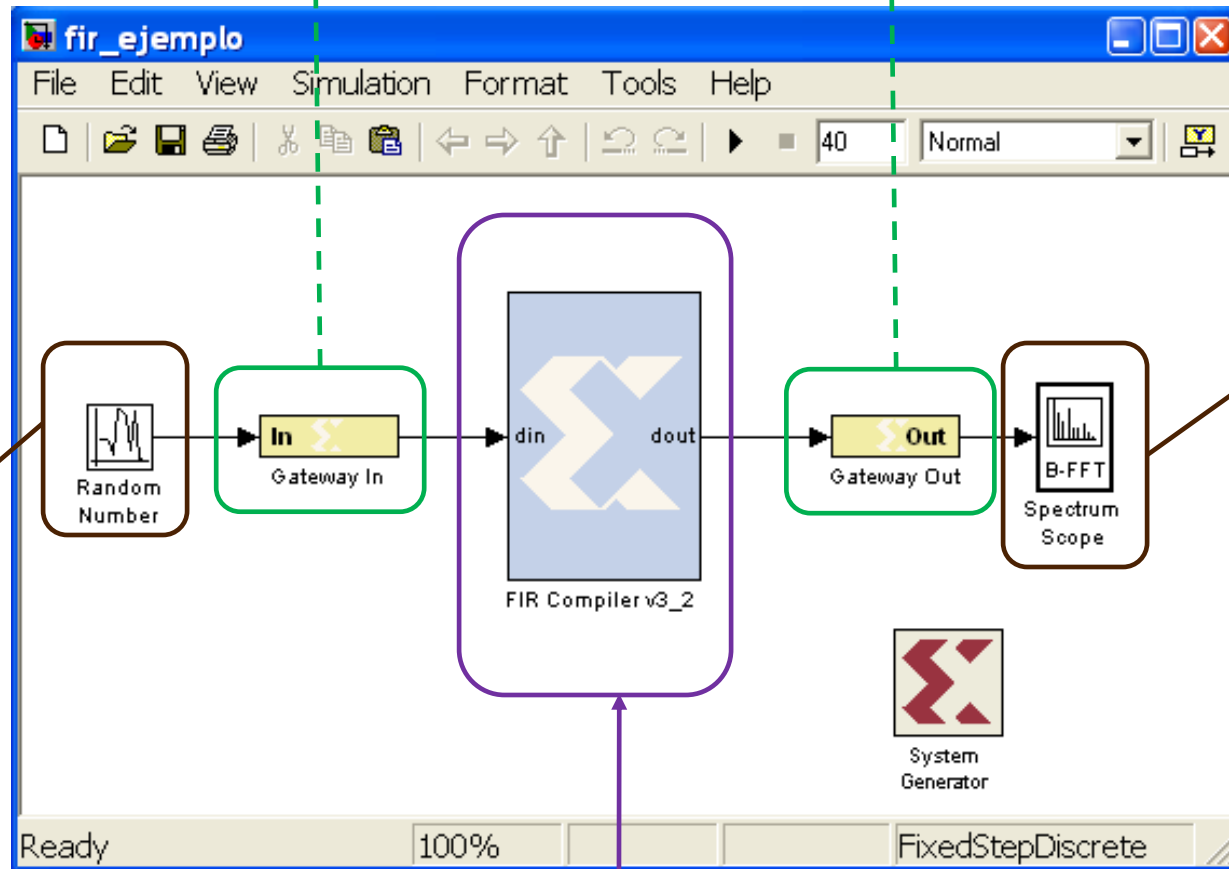


Usando Xilinx Blockset



Ejemplo de Diseño

Gateway In/OUT: interface
entre SysGen y Simulink

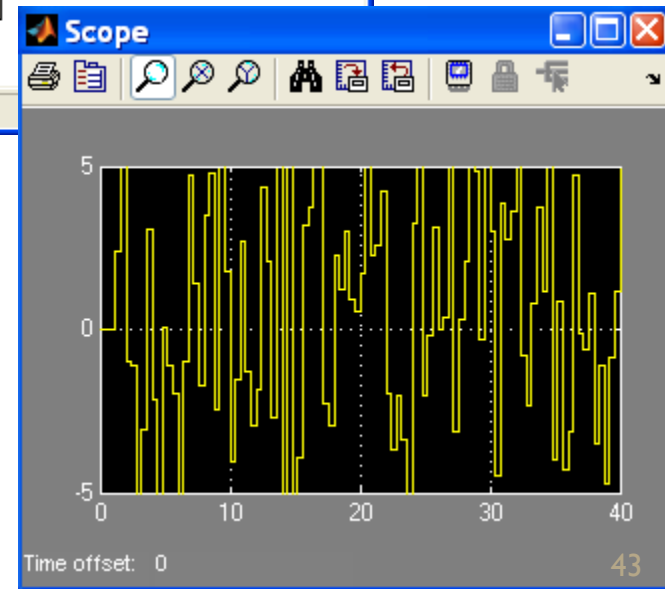
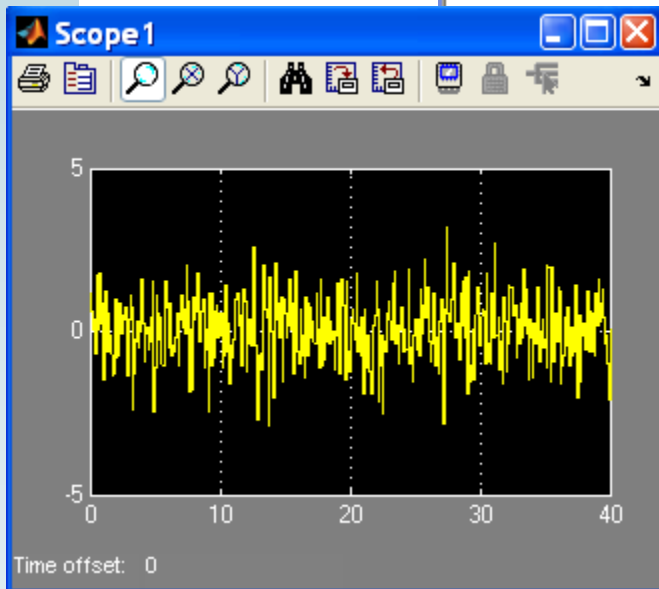
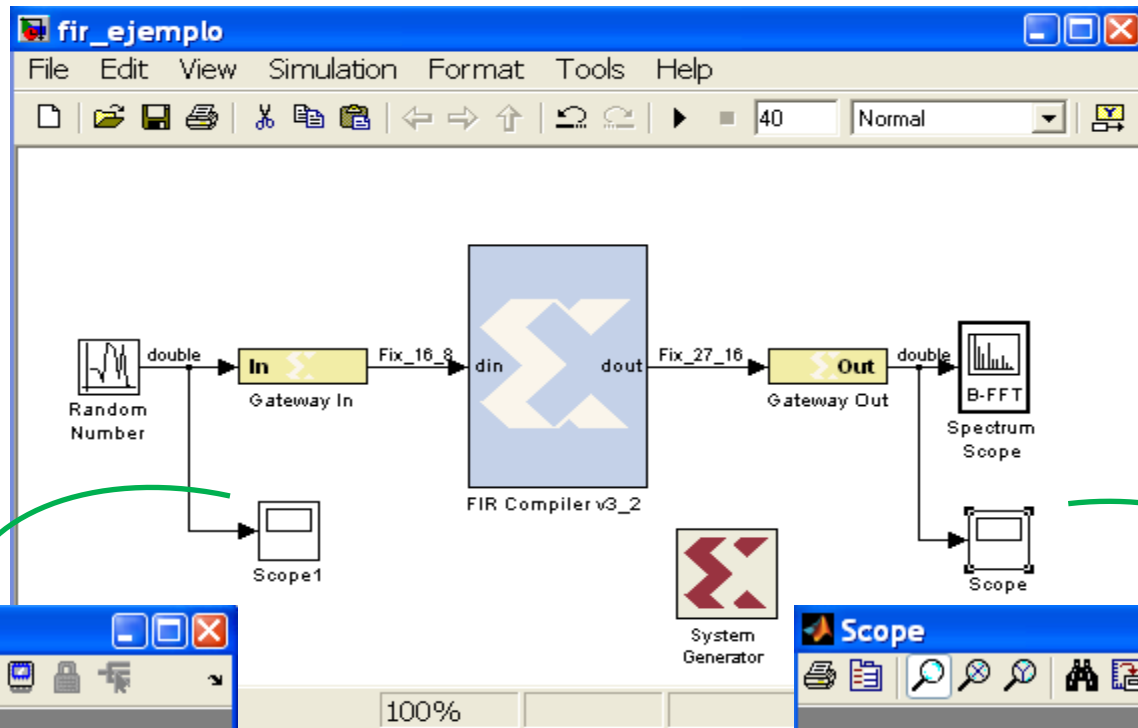


Simulink
Fuente de
Senal

Simulink
Graficos
de Senal

Lógica a implementar en el FPGA

Simulación en Simulink

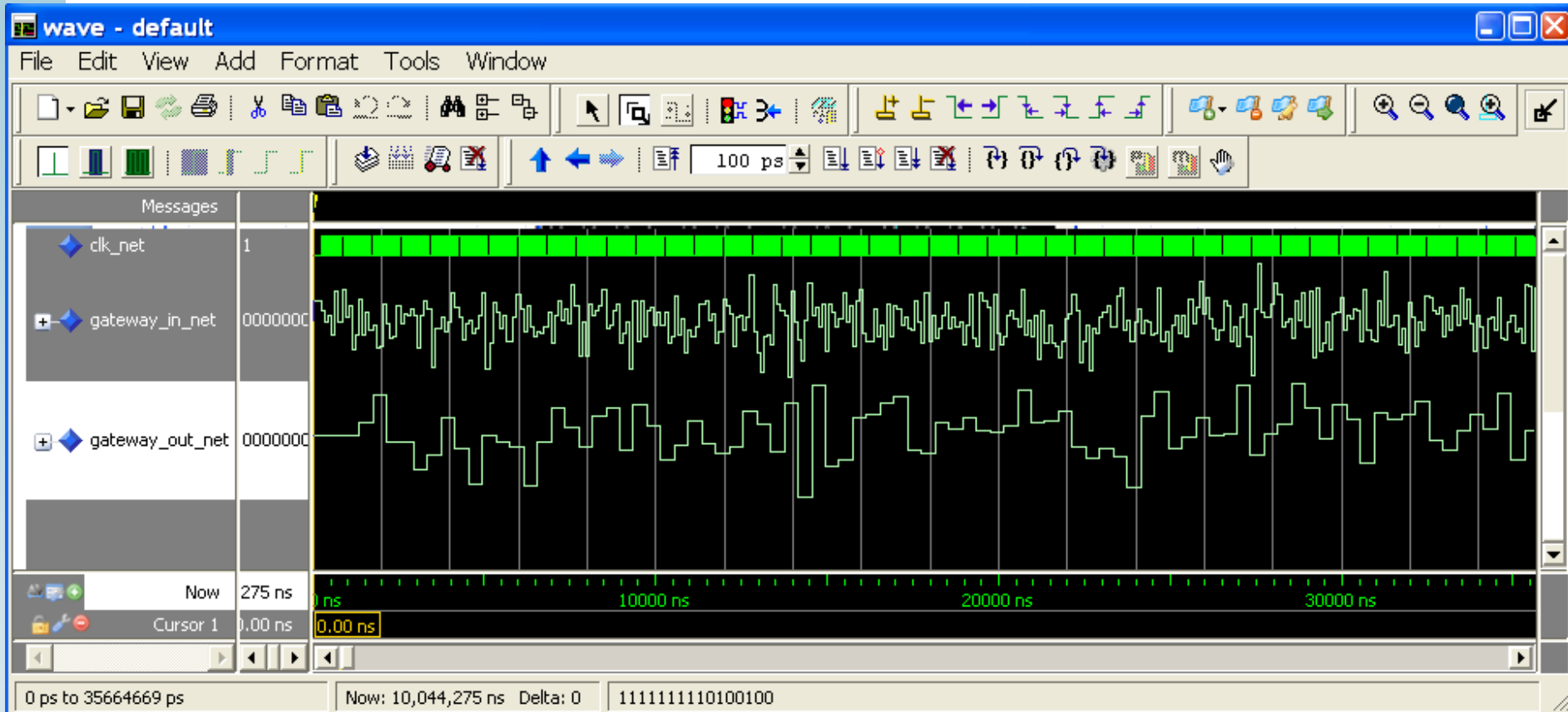


Proyecto FPGA – Xilinx ISE

The screenshot displays the Xilinx ISE environment. The title bar indicates the project path: `F:\A_ILM_09\CET 486\Ch11_DSP_VHDL\MatLab\netlist\fir_ejemplo_cw.ise - [fir_ejemplo_cw.vhd]`. The menu bar includes File, Edit, View, Project, Source, Process, Window, and Help. The toolbar contains various icons for file operations and simulation. The Sources window on the left shows the project hierarchy for 'Implementation', listing files like `xc4vsx35-10ff668`, `fir_ejemplo_cw - structural (fir_ejemplo_cw.vhd)`, and various components like `clk_probe`, `default_clock_driver_x0`, `xlpersistentdff`, `synth_reg`, and `delay_comp`. The main editor window displays the VHDL code for `fir_ejemplo_cw.vhd`.

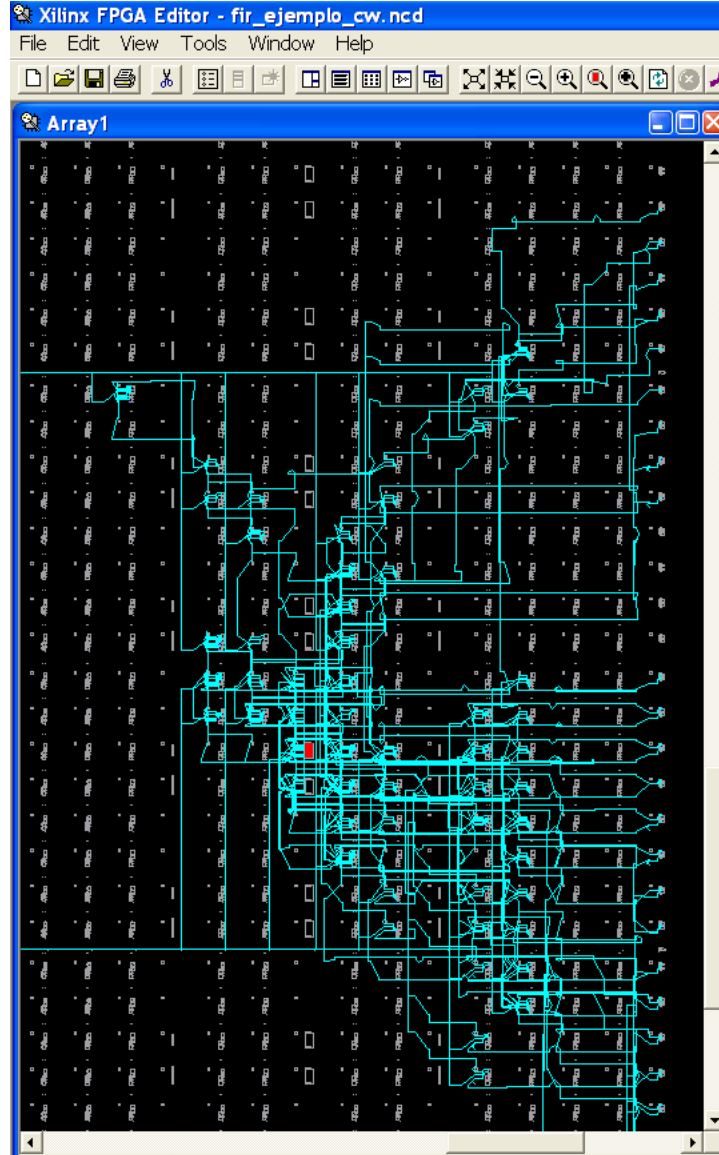
```
428 library IEEE;
429 use IEEE.std_logic_1164.all;
430 use work.conv_pkg.all;
431
432 entity fir_ejemplo_cw is
433     port (
434         ce: in std_logic := '1';
435         clk: in std_logic; -- clock period = 100.0 ns (10.0 Mhz)
436         gateway_in: in std_logic_vector(15 downto 0);
437         gateway_out: out std_logic_vector(26 downto 0)
438     );
439 end fir_ejemplo_cw;
440
441 architecture structural of fir_ejemplo_cw is
442     component xlpersistentdff
443     port (
444         clk: in std_logic;
445         d: in std_logic;
446         q: out std_logic
447     );
448 end component;
449 attribute syn_black_box: boolean;
450 attribute syn_black_box of xlpersistentdff: component is true;
```

Simulacion en Xilinx



C77

© C. Sisterna



Muchas Gracias !

C7 Technology

<http://www.c7t-hdl.com>

<http://hdl-fpga.blogspot.com.ar/>