

Ingeniería Informática

Sistemas Electrónicos Digitales (SED)

Profesor: José Miguel Delgado Hernández
Ingeniero de Telecomunicación
E-mail: jdelher@ull.es

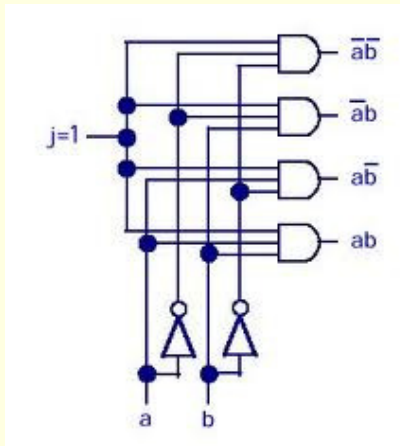
Práctica 2

Funciones booleanas elementales descritas con VHDL

Objetivos:

- Estudio de **funciones booleanas** elementales
- Familiarización con el entorno de desarrollo **ISE Design Suite**
- Conceptos básicos lenguaje de descripción hardware: **VHDL**

Práctica 2: Resumen



Funciones booleanas

```

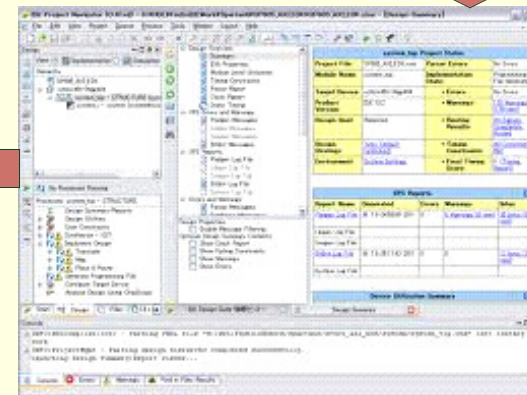
1 library IEEE;
2 use IEEE.std_logic_1164.all;
3
4 entity MUX4x2UH is
5     port (
6         A: in STD_LOGIC_VECTOR (3 downto 0);
7         B: in STD_LOGIC_VECTOR (3 downto 0);
8         C: out STD_LOGIC_VECTOR (3 downto 0);
9         S1: in STD_LOGIC
10    );
11 end MUX4x2UH;
12
13 architecture MUX4x2UH_arch of MUX4x2UH is
14 begin
15     -- <<enter your statements here>>
16     C(0) <= ((not S1) and A(0)) or (S1 and B(0));
17     C(1) <= ((not S1) and A(1)) or (S1 and B(1));
18     C(2) <= ((not S1) and A(2)) or (S1 and B(2));
19     C(3) <= ((not S1) and A(3)) or (S1 and B(3));
20

```

Lenguaje VHDL



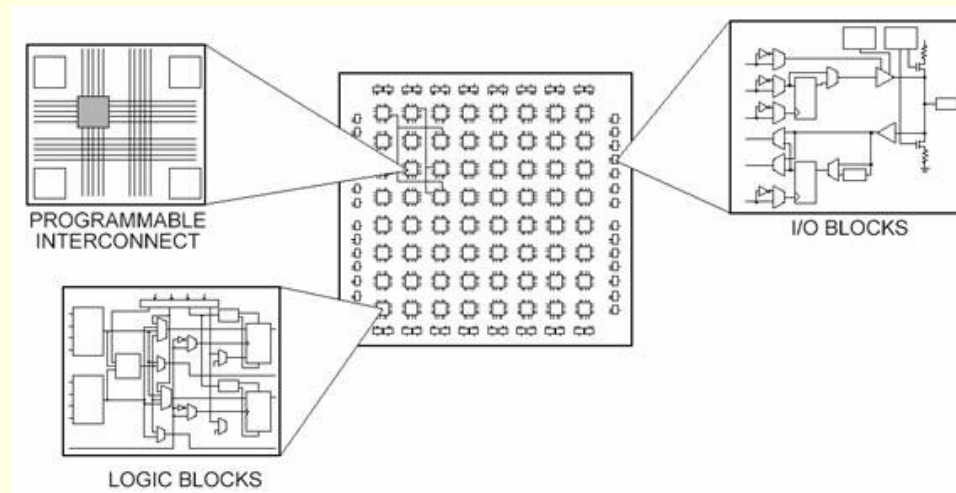
FPGA



ISE: Entorno de desarrollo

- ❑ FPGA = *Field-Programmable Gate Array*.
- ❑ Componente estándar (re)programable por el usuario. Esto implica:
 - ❑ Interconexión (re)programable.
 - ❑ Función lógica (re)programables.
 - ❑ E/S (re)programable.
- ❑ Inventado y patentado por S. Wahlstrom en 1967: una idea demasiado adelantada respecto a la tecnología disponible.

Cada chip de FPGA está hecho de un número limitado de recursos **predefinidos con interconexiones programables** para implementar un circuito digital reconfigurable y **bloques de E/S** para permitir que los circuitos tengan acceso al mundo exterior.

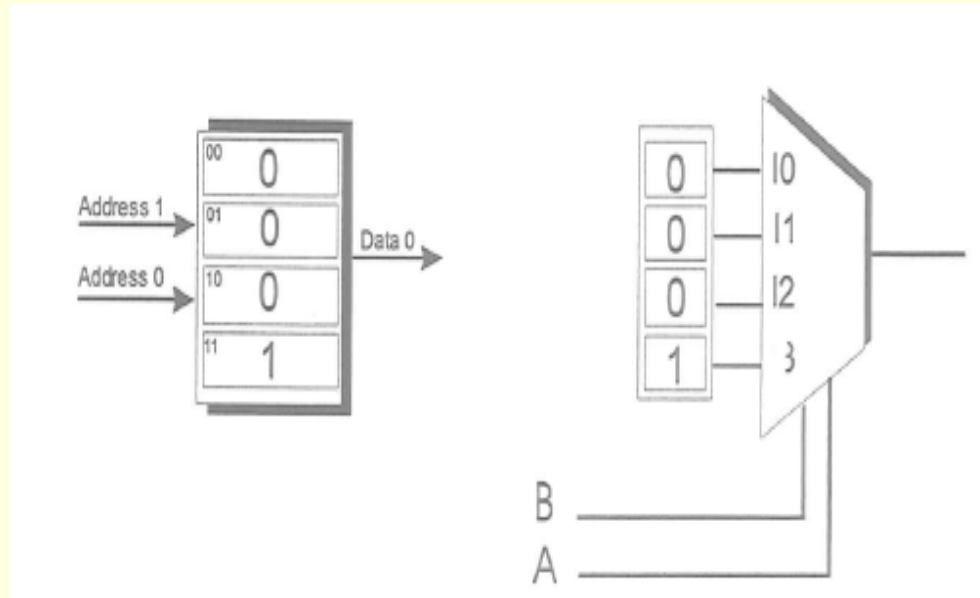
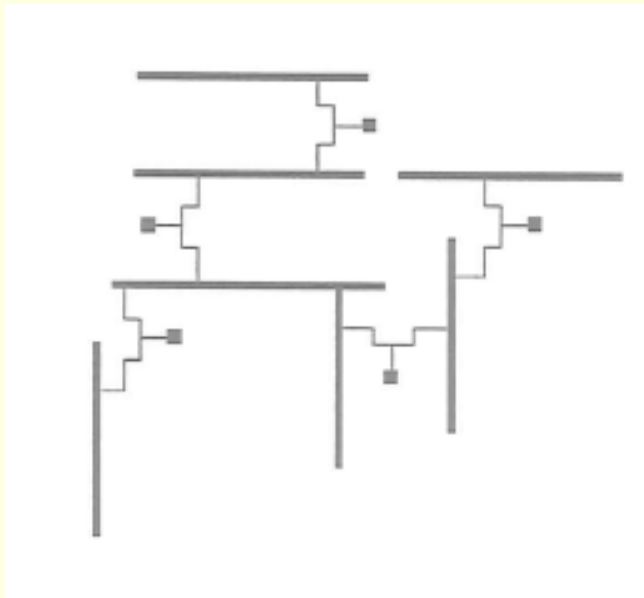


FPGA: Componentes básicos

- Funciones lógicas reconfigurables
- Interconexión reconfigurable
- “Patatas” E/S reconfigurables
- Memoria para almacenar configuración
- Circuito de control para configuración
- Bloques dedicados: μ P, multiplicadores, memorias, ...

Usuario

Configuración



CLB: Configurable Logic Block

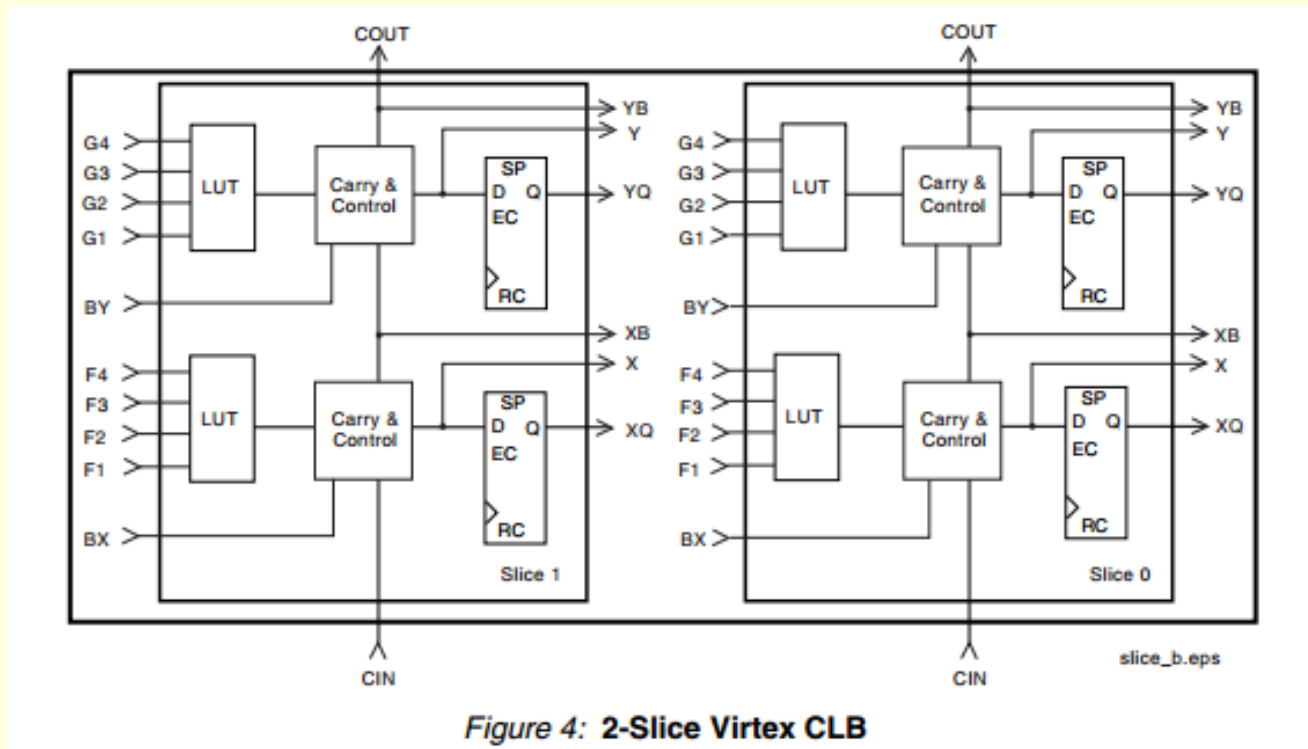
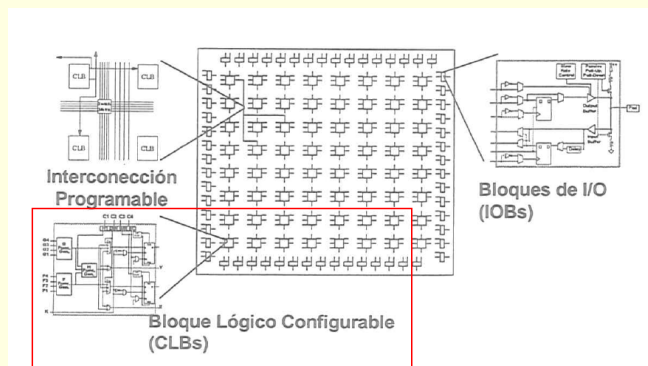
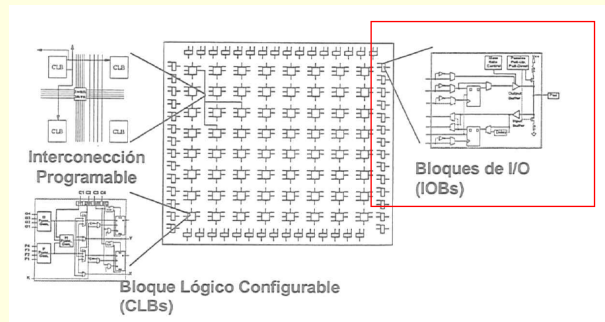
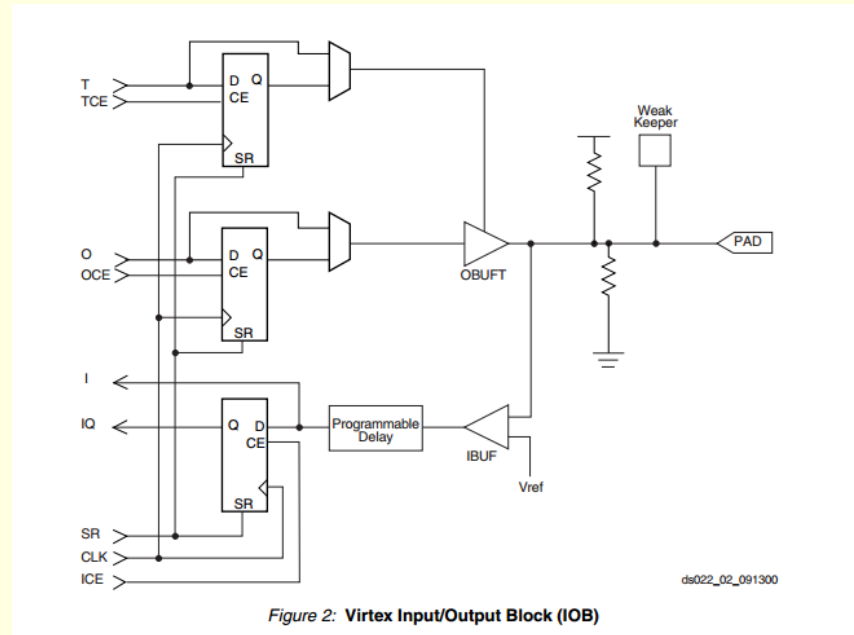


Figure 4: 2-Slice Virtex CLB



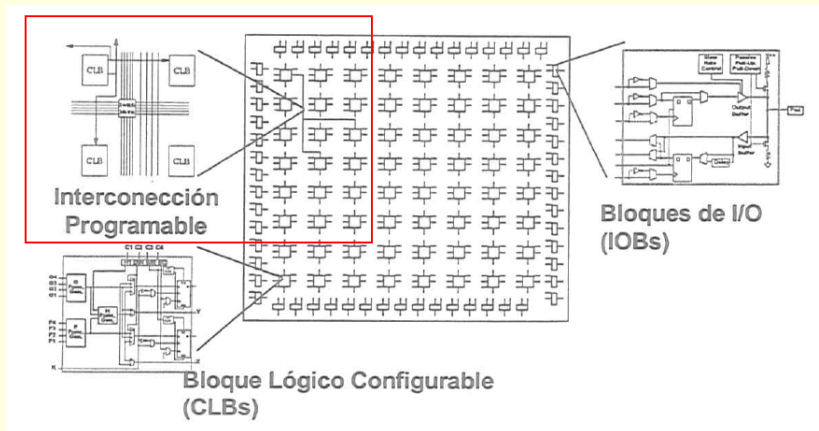
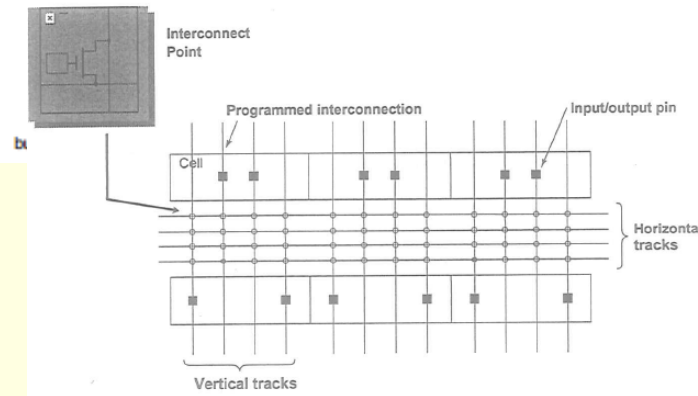
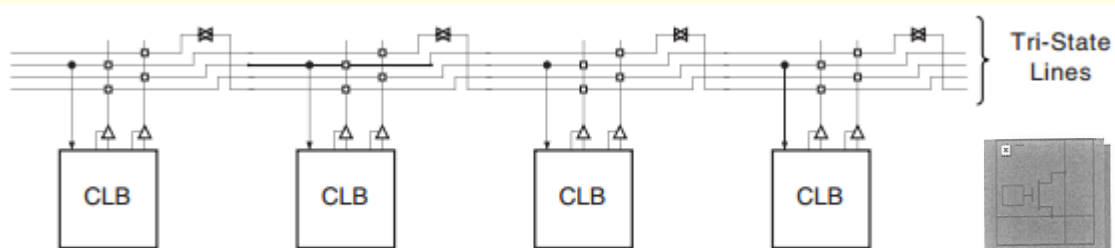
- Slices: Dos slices por cada CLB
- Basados en LookUpTables (LUT)
- Necesita un reloj (Clock)
- Multiplexores (MUX) y FlipFlop (FF)

I/OB: Input-Output Block



- Flip-Flop (FF)
- Necesita un reloj (Clock)
- Electrónica para salida/entrada
- Retardos programables

Programmable Routing Matrix



- Locales
- De propósito general
- I/O
- Dedicadas
- Globales
- Distribución de los relojes

Aplicaciones FPGA

- Automóvil, aeroespacial, defensa, industrial, audio, procesamiento de video e imagen, centro de datos, computación alto rendimiento, medicina, comunicaciones, seguridad, pruebas y medidas
- Cámaras digitales, smart phones, impresoras multifunción, TV 3D

Consumer Electronics

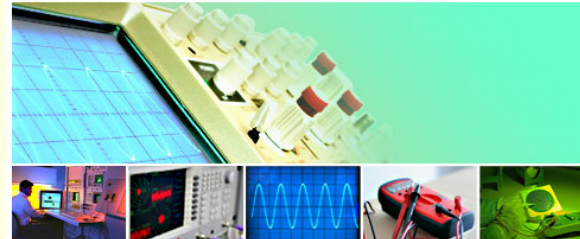


Aerospace and Defense



Test and Measurement

Home > End Markets > Test & Measurement



<http://www.altera.com/>



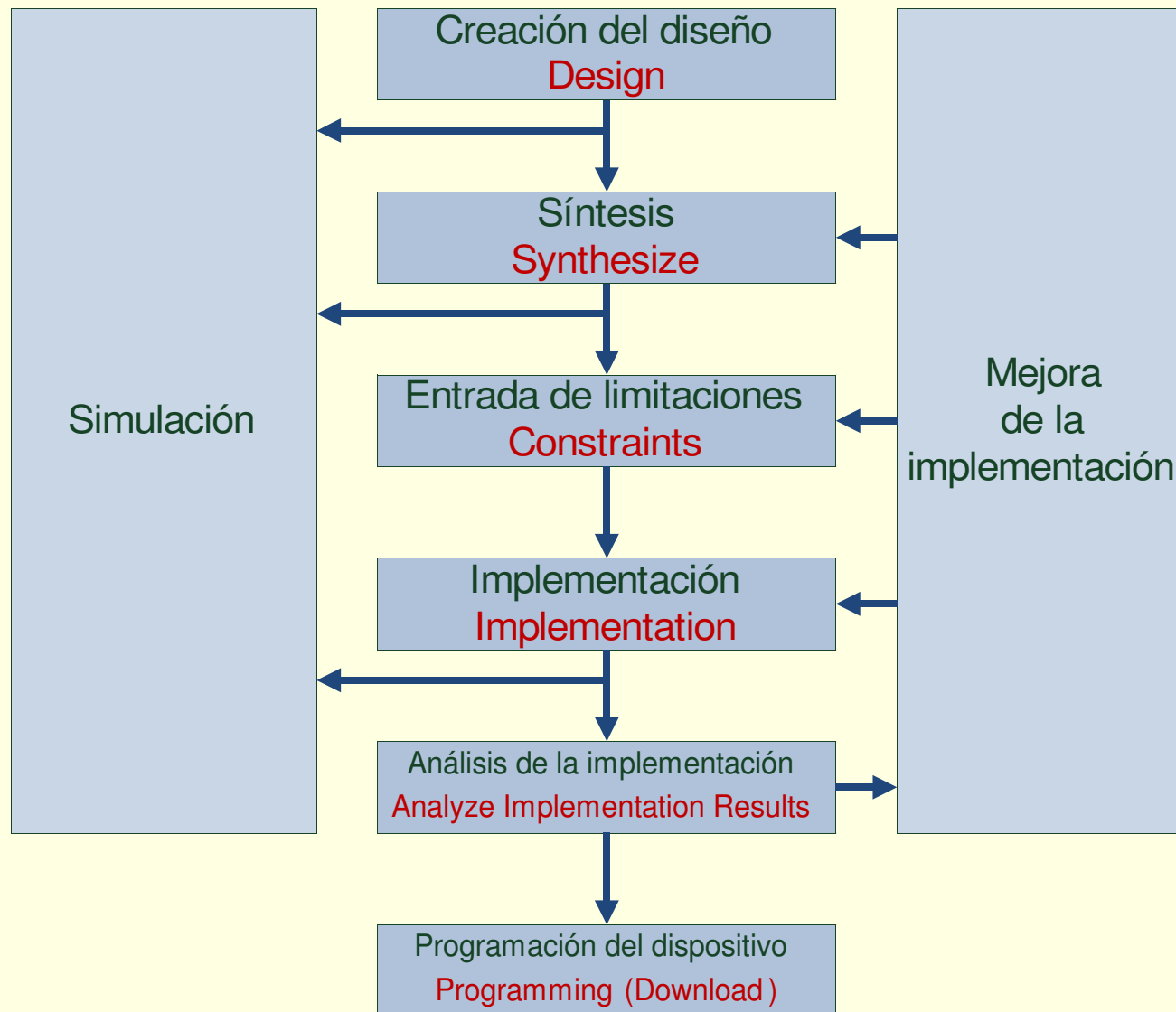
<http://www.xilinx.com/company/gettingstarted/index.htm>

Práctica 2: ¿Qué es ISE?

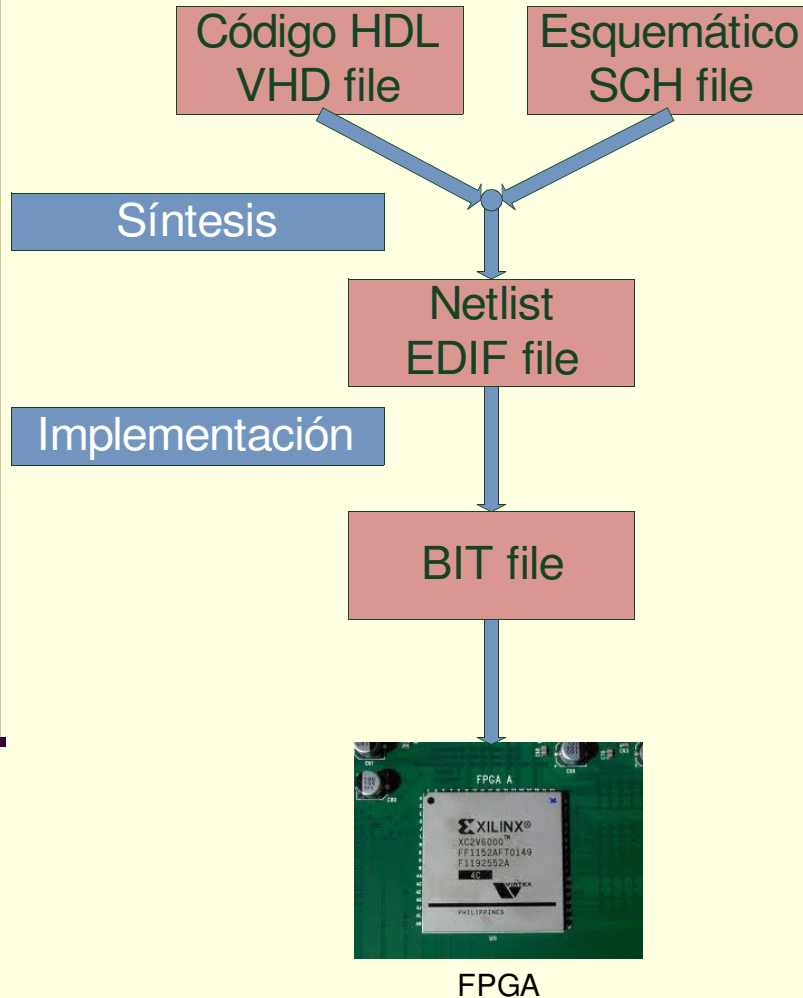
- **Integrated Software Environment**
- Es una interfaz gráfica para el diseño y la implementación de proyectos basados en FPGAs de Xilinx®.
- **Diseño**: Herramientas para la descripción del hardware: VHDL, IP cores, Schematic, State diagrams, Testbenches, Constraints files.
- **Implementación**: Synthesize, Translate, Map, Place&Route

http://www.xilinx.com/support/documentation/sw_manuals/xilinx12_4/isehelp_start.htm

Práctica 2: Proceso del diseño ISE

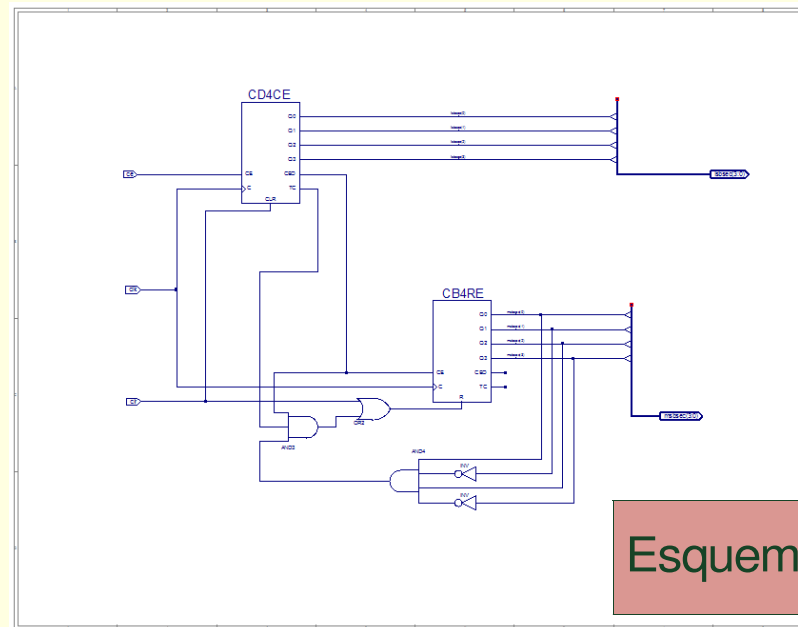


Descripción del proceso de diseño



```
11 USE ieee.std_logic_1164.all;
12
13 ENTITY STMACH_V IS
14     PORT (CLK,DCM_lock,reset,strtstop: IN std_logic;
15           clken,rst : OUT std_logic);
16 END;
17
18 ARCHITECTURE BEHAVIOR OF STMACH_V IS
19     SIGNAL sreg : std_logic_vector (2 DOWNTO 0);
20     SIGNAL next_sreg : std_logic_vector (2 DOWNTO 0);
21     CONSTANT clear : std_logic_vector (2 DOWNTO 0) := "100";
22     CONSTANT counting : std_logic_vector (2 DOWNTO 0) := "011";
23     CONSTANT start : std_logic_vector (2 DOWNTO 0) := "001";
24     CONSTANT stop : std_logic_vector (2 DOWNTO 0) := "111";
25     CONSTANT stopped : std_logic_vector (2 DOWNTO 0) := "101";
26     CONSTANT zero : std_logic_vector (2 DOWNTO 0) := "000";
27
28     SIGNAL next_clken : std_logic;
29 BEGIN
30     PROCESS (CLK, DCM_lock, reset, next_sreg, next_clken)
31     BEGIN
32         IF ( reset='1' ) OR ( DCM_lock='0' ) THEN
33             sreg <= clear;
34             clken <= '0';
35         ELSIF CLK='1' AND CLK'event THEN
36             sreg <= next_sreg;
37             clken <= next_clken;
38         END IF;
39     END PROCESS;
```

Código HDL



Esquemático

Práctica 2: Project Navigator

1. Toolbar

2. Panel: Diseño

- Vistas:

- Implementation
- Simulation

- Jerarquía

- Procesos

- Informes/Resumen
- Utilidades de diseño
- Limitaciones usuario
- Síntesis
- Implementación
- Generar Bitstream
- Configurar dispositivo

3. Espacio de trabajo

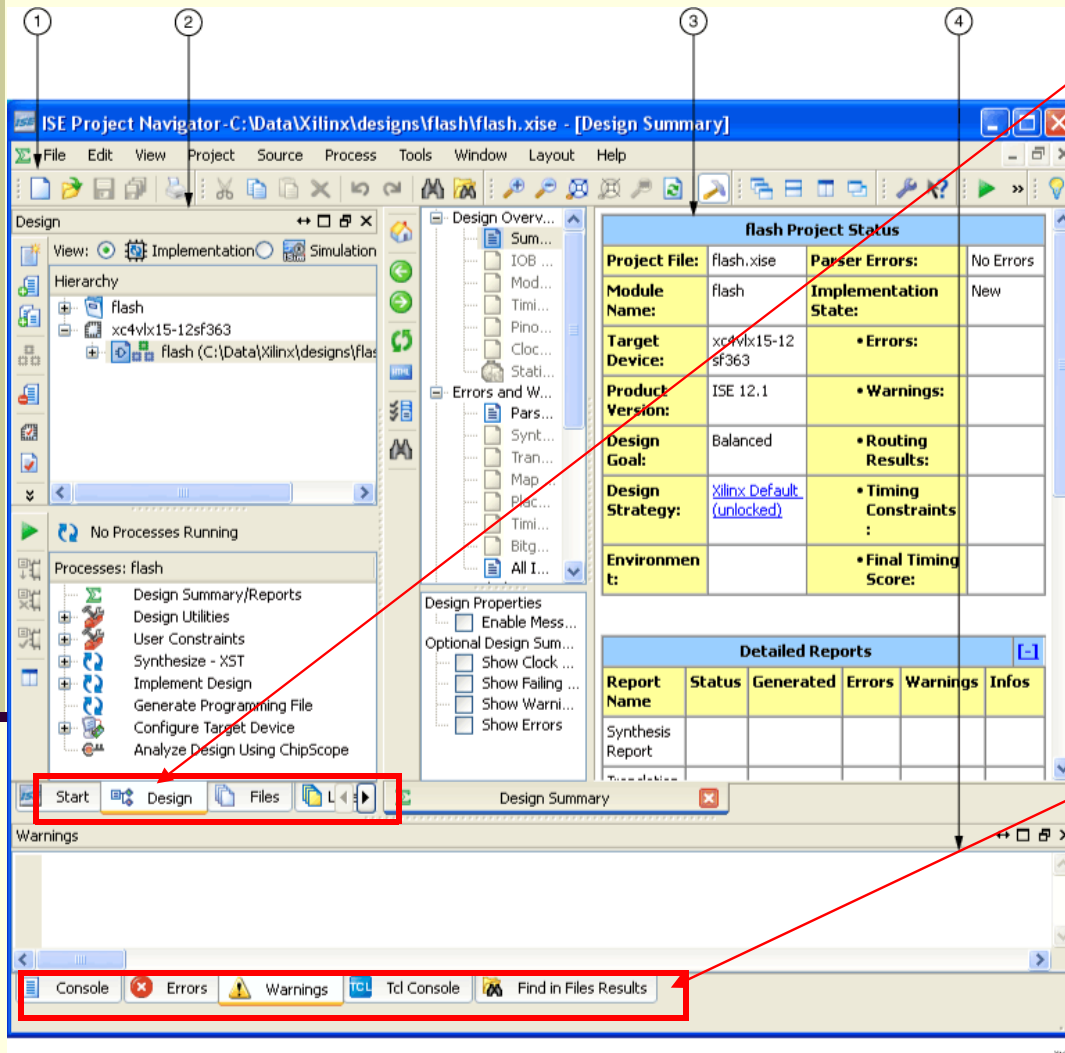
4. Ventana de mensajes

- Consola

- Errores

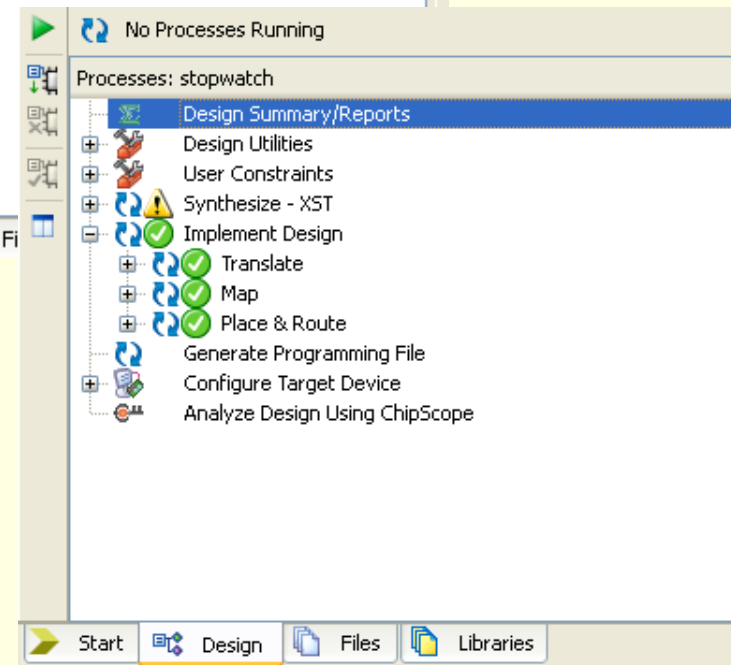
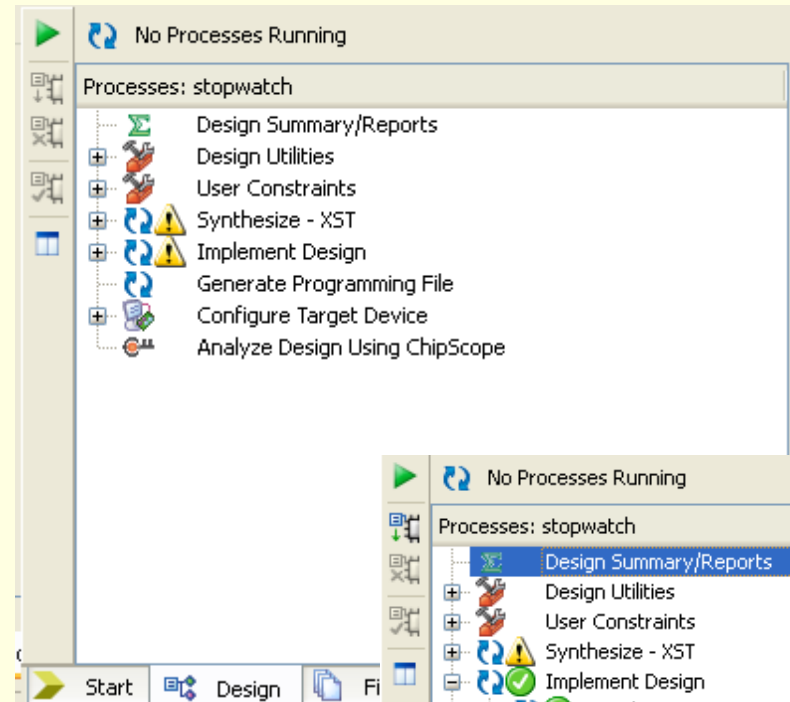
- Avisos

- Búsqueda en ficheros







Procesos

- Informes/Resumen
- Utilidades de diseño
- Limitaciones usuario
- Síntesis
- Implementación
- Generar Bitstream
- Configurar dispositivo



Sin marca, el proceso no se ha ejecutado

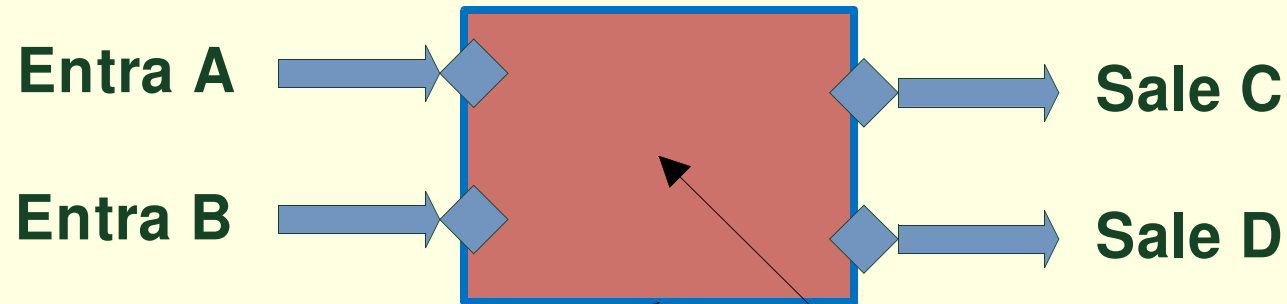
-  **El proceso se ha completado satisfactoriamente**
-  **El proceso se ha completado con avisos**
-  **El proceso no se ha completado . Hay errores**
-  **El proceso o alguno de los procesos dependientes está caducado**

VHDL: Conceptos iniciales

- Lenguaje de Descripción Hardware
- Sirven para:
 - Expresar ideas que modelan circuitos
 - Simular el circuito
 - Crear el circuito
 - Hacer bancos de pruebas
 - Documentar
- ¡Ojo! No es un lenguaje de programación software

VHDL: La entidad y la arquitectura

Es una abstracción: Caja negra



Entradas

Salidas

ENTITY

Describe la E /S del diseño

ARCHITECTURE

Describe el contenido del diseño

VHDL: Puertos de una entidad (PORTS)

- PORTS: Son los canales de comunicación
- Tienen un **nombre** que debe ser único dentro de la entidad.
- Se define el **modo** o dirección del flujo de datos: entrada, salida, bidireccional, buffer.
- Se define el **tipo** para indicar los valores que puede tomar.
- Son una clase especial de señal a la que se le añade el modo: IN, OUT, BUFFER, INOUT

NOMBRE

```
20  LIBRARY IEEE;  
21  USE IEEE.STD_LOGIC_1164.ALL;  
22  
23  ENTITY caja IS PORT (  
24      A : in  STD_LOGIC;  
25      B : in  STD_LOGIC;  
26      C : out  STD_LOGIC;  
27      D : out  STD_LOGIC);  
28  END caja;
```

MODO

TIPO

VHDL: Estructura de un diseño

```
20  LIBRARY IEEE;
21  USE IEEE.STD_LOGIC_1164.ALL;
22
23  ENTITY caja IS PORT (
24
25
26
27
28  END caja;
29
30  ARCHITECTURE SED of caja IS
31
32
33
34  BEGIN
35
36
37  END SED;
```

Declaración de los puertos

Parte declarativa de la
arquitectura

Cuerpo de la arquitectura

Nombre de la
entidad

Nombre de la
arquitectura

- Los tipos de datos predefinidos son:
 - Escalares: integer, real, enumerated, physical
 - Compuestos: array, record
 - Punteros: access
 - Archivos: file
- Tipos básicos predefinidos IEEE-1076:
 - **BIT**: valor '0' o valor '1'. Modela señales digitales
 - **BIT_VECTOR**: es un array unidimensional de bits. Modela buses
 - **INTEGER**: tipo entero
 - **BOOLEAN**: 'TRUE' o 'FALSE'
 - **REAL**: tipo para números en coma flotante
 - **ENUMERATED**: Conjunto de valores definidos por el usuario.
Ejemplo: TYPE color IS (blanco, rojo, azul, negro)

➤ IEEE.standard_logic_1164

- **U**: No inicializado, valor por defecto
- **X**: Desconocido forzado. salida con múltiples fuentes en corto
- **0**: Cero forzado. Salida de una puerta con nivel lógico bajo
- **1**: Uno forzado. Salida de una puerta con nivel lógico alto
- **Z**: Alta impedancia
- **W**: Desconocido débil
- **L**: Cero débil
- **H**: Uno débil
- **-**: No importa, se usa como comodín para la síntesis

- El objeto básico es la señal, lo utilizamos para modelar hilos del circuito.
- Contiene también información del tiempo en el que toma el valor.
- Se declaran en la arquitectura, antes del BEGIN.
- Pueden tener un valor inicial (no en síntesis) **`:=`**
- Para asignar valores se utiliza **`<=`**

Señales
declaradas antes de BEGIN

```
30 ARCHITECTURE SED of caja IS
31
32   SIGNAL sA : STD_LOGIC;
33   SIGNAL sB : BOOLEAN := TRUE;
34
35 BEGIN
36
37   sA <= '1';
38
39 END SED;
```

Señal iniciada

Se le asigna valor
alto a la señal sA

VHDL: Constantes y variables

- Se declaran antes del BEGIN
- Las constantes pueden ser de cualquier tipo
- Las variables:
 - Almacenan valores, pero no tienen información temporal
 - Solamente son visibles en interior de un proceso y no en toda la arquitectura.
 - Los valores son asignados por medio de **`:=`**

CONSTANTES
declaradas antes de BEGIN

VARIABLE
declarada antes de BEGIN
en un PROCESO

Asignación de un
VARIABLE

```
30 ARCHITECTURE SED of caja IS
31
32     CONSTANT alto      : STD_LOGIC := '1';
33     CONSTANT canales   : INTEGER   := 3;
34     CONSTANT tiempo    : TIME      := 15 ns;
35
36 BEGIN
37
38     PROCESS (abre, cierra)
39
40         VARIABLE humedad : INTEGER;
41
42         BEGIN
43
44             humedad := 30;
45
46         END PROCESS;
47
48 END SED;
```