Hardware Libre:
Clasificación y desarrollo de hardware reconfigurable en entornos GNU/Linux

Iván González, Juan González, Francisco Gómez-Arribas

Escuela Politécnica Superior
Universidad Autónoma de Madrid

ÍNDICE

¿Qué es el hardware Libre?

- **Paralelismo con Software Libre**
- **Software Libre: Ofrece 4 libertades**
  - Libertad de uso
  - Libertad de compartir (distribuir)
  - Libertad de modificación (Fuentes)
  - Libertad de distribución de las modificaciones
- **Hardware libre**: Aspira a ofrecer esas mismas 4 libertades, pero aparecen problemas.

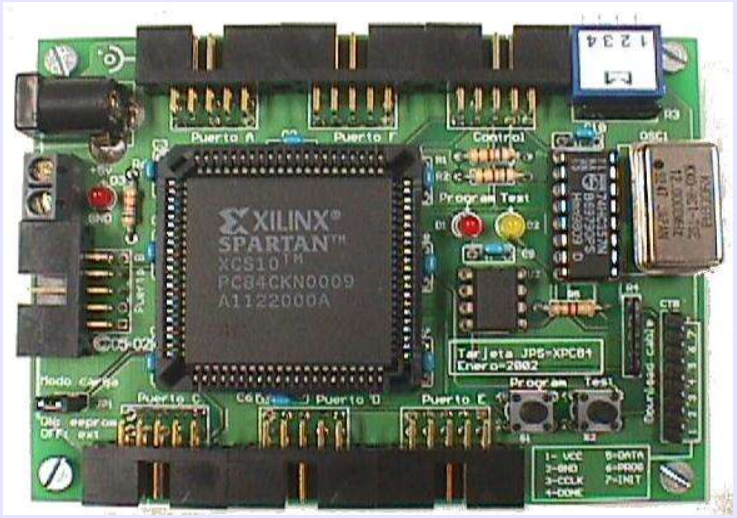
¿  ?

El objetivo del hardware libre es aplicar las mismas 4 libertades del software libre, en su propio campo

Clasificación del hardware

- **Según su naturaleza, encontramos dos grandes grupos:**

HARDWARE ESTÁTICO, conjunto de materiales de los sistemas electrónicos.



Existencia física

HARDWARE RECONFIGURABLE, el que viene descrito mediante lenguajes de descripción hardware (HDL)

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity cont8 is
  port (clk   : in std_logic; -- Reloj
        clear : in std_logic;
        q     : out std_logic_vector (7 downto 0)); --
Salida
end cont8;

architecture beh of cont8 is
signal cuenta : std_logic_vector (7 downto 0);
```

Es "código"

Siempre hay que especificar de qué tipo de hardware estamos hablando

ÍNDICE

Problemas del hardware libre

- Queremos aplicar las 4 libertades, pero surgen problemas:

1. **Un diseño físico es único.** Para compartir mi placa con otra persona, bien le dejo la mía o bien se la tiene que fabricar. La compartición tal cual la conocemos en el mundo del software no es posible.

2. **La compartición tiene asociado un coste**. Para compartir hardware libre hay que FABRICAR y comprar componentes. Además hay que verificar su correcto funcionamiento.

3. **Disponibilidad de los componentes**. ¿Están disponibles los chips?

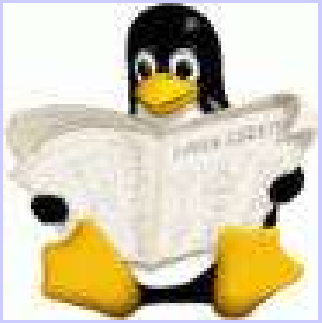
Problemas derivados de su Existencia Física

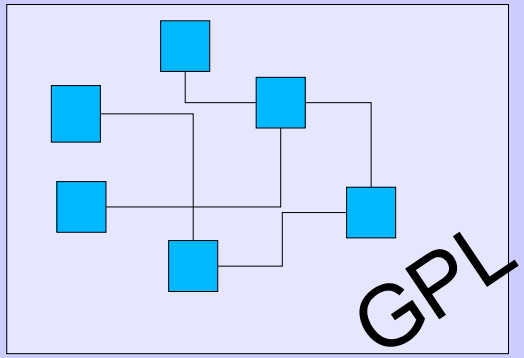
Definición de hardware libre (I)

- No hay una definición clara
- **Una propuesta:**

> El **hardware libre** (o abierto) ofrece las mismas 4 libertades que el software libre, pero aplicadas a los **PLANOS** del hardware.

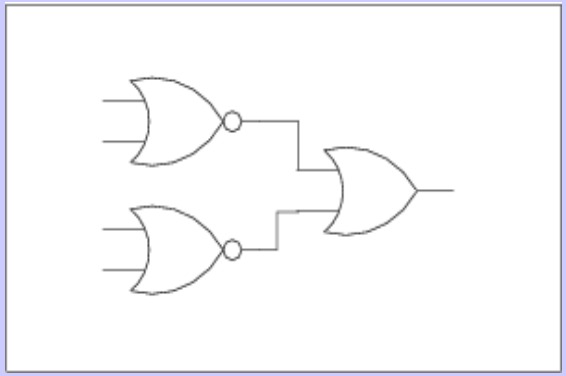
- En el software se habla de fuentes, en el hardware de planos
- **Los planos se pueden compartir igual que el software. Es la fabricación la que tiene un coste.**



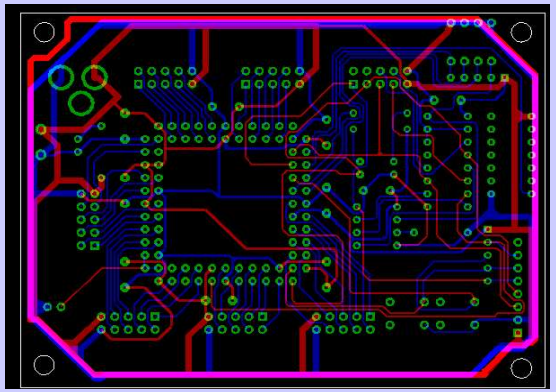


Tipos de planos en electrónica

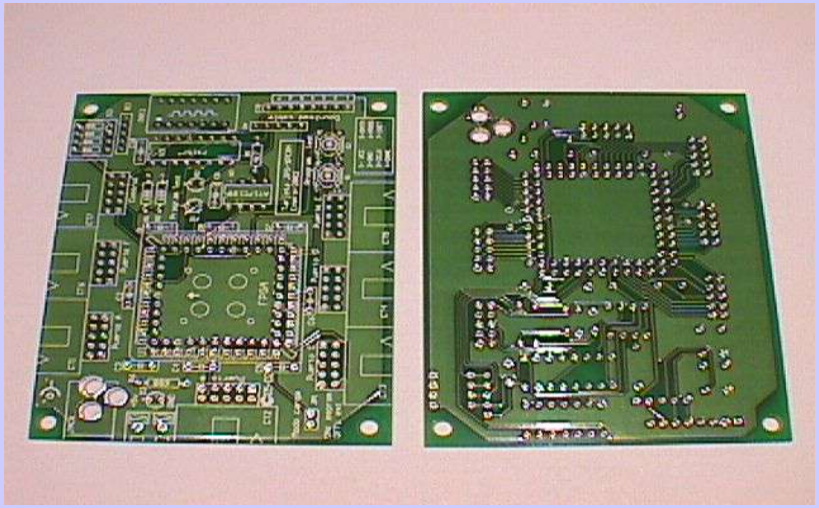
- Esquemático



- Circuito Impreso (PCB)



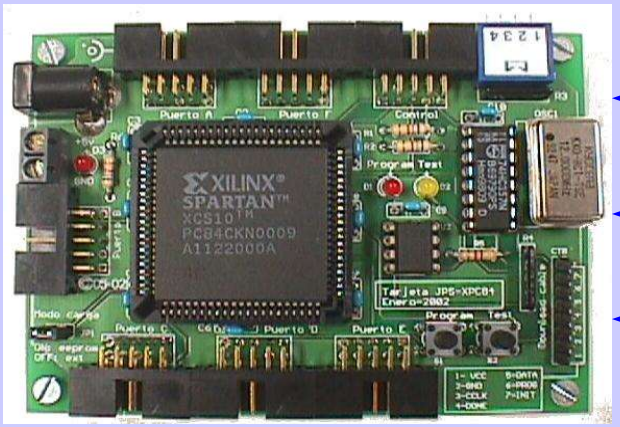
- Fichero para fabricación (GERBER)
  - Sólo en placas industriales

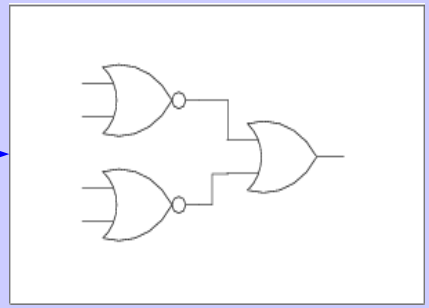


Definición hardware libre (II)

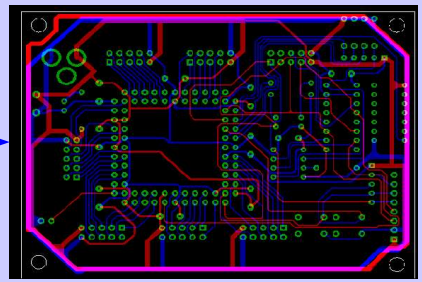
Un diseño se considera hardware libre si ofrece las 4 libertades del software libre en el **esquemático**, **PCB** y **fichero para fabricación**

Hardware libre

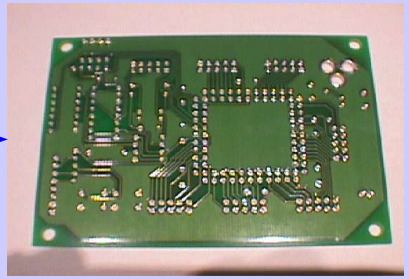




.sch



.pcb



gerber

Formato de los planos (I)

- **Fichero de fabricación:** GERBER, estándar industrial ✓
- **Esquemático y PCB:** Cada aplicación su propio formato ✗

No hay formato estándar

- **Lo ideal**: **Herramientas de desarrollo** (EDA) **Libres**
- **La realidad:** Software propietario, con formatos propietarios

El formato **impone restricciones** a la compartición de los planos

¿Es **hardware libre** si el formato de alguno de sus planos es propietario?

Formato de los planos (II)

- **Nuestra propuesta:**

Que sea el **autor el que decida**, con independencia de la aplicación empleada para su diseño

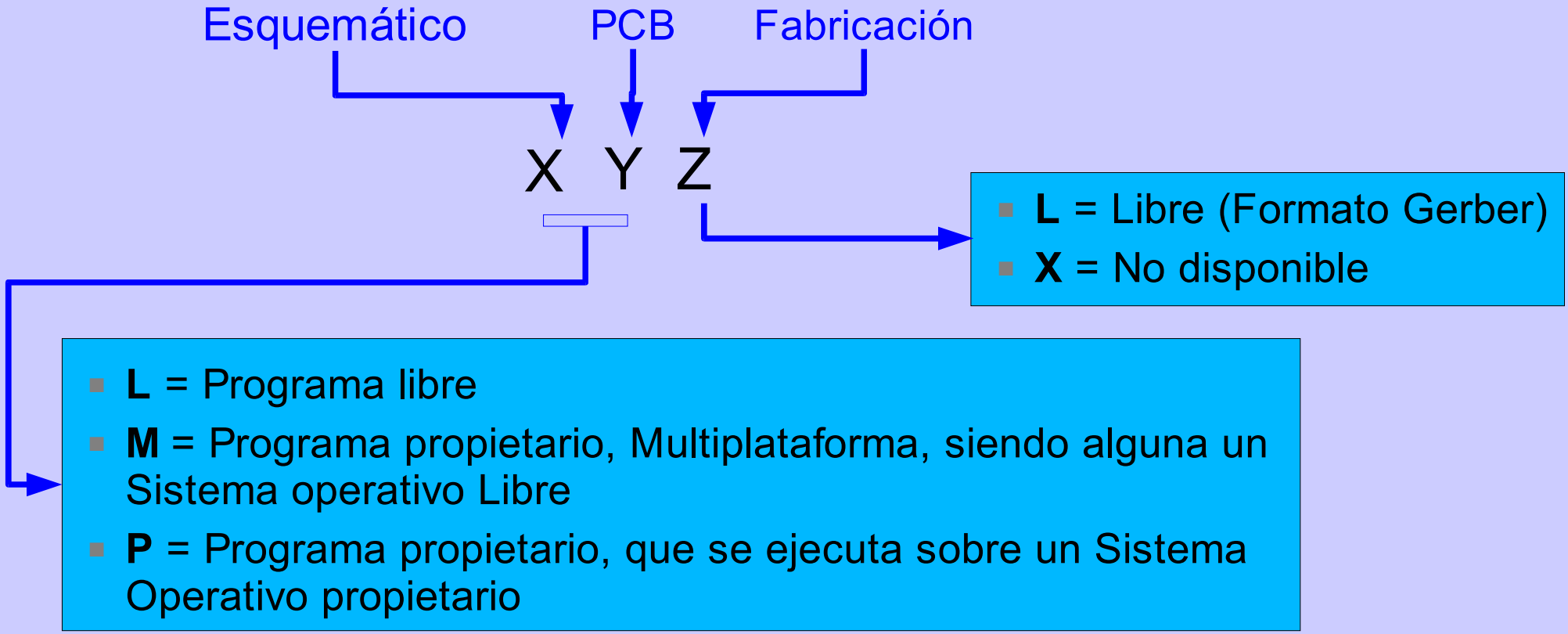
- **Herramientas de desarrollo Libres:** Proyecto gEDA.
Prometedor, pero en desarrollo

- **Software propietario todavía muy por delante,**
Orcad, Tango, Eagle, Protel,...

La aplicación utilizada impone restricciones a la compartición. En base a esas restricciones clasificaremos el hardware libre

Clasificación del hardware libre (I)

- **Clasificación de los diseños en función de los planos y el tipo de programas usados para generarlos**
  - Tres componentes
  - Cada una asociada a un tipo de plano



Clasificación del hardware libre (II)

- **Se pueden tener diseños LLL, PPL, PML...**
- **Existen tres tipos a destacar:**

- **Diseños LLL**
  - Ninguna restricción. Cualquiera los puede ver, modificar y fabricar. Diseños "Pura Sangre"

- **Diseños MML**
  - Lo más práctico para el diseñador, al día de hoy. Necesaria una licencia, pero al menos se puede usar un sistema operativo libre

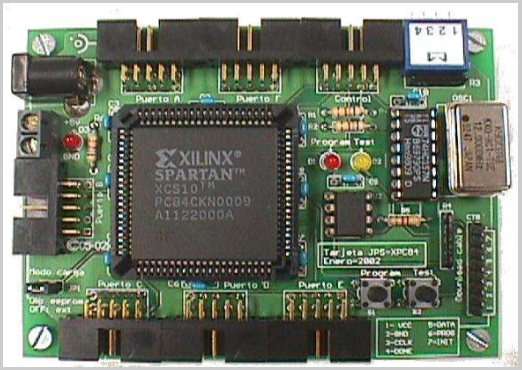
- **Diseños PPX**
  - Es lo más restrictivo. Necesarias licencias de las herraminetas y del sistema operativo. No hay fichero de fabricación

Clasificación del hardware libre (III)

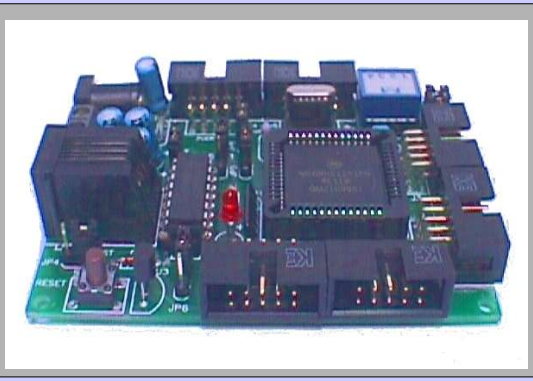
- **Lo ideal: Diseños LLL**
  - ¿Existe algún diseño LLL?

?

- **Lo práctico: Diseños MML**
  - Programa de diseño Eagle (CadSoft)
  - Descarga gratuita (Freeware)
  - Ejemplo: **Tarjeta JPS**

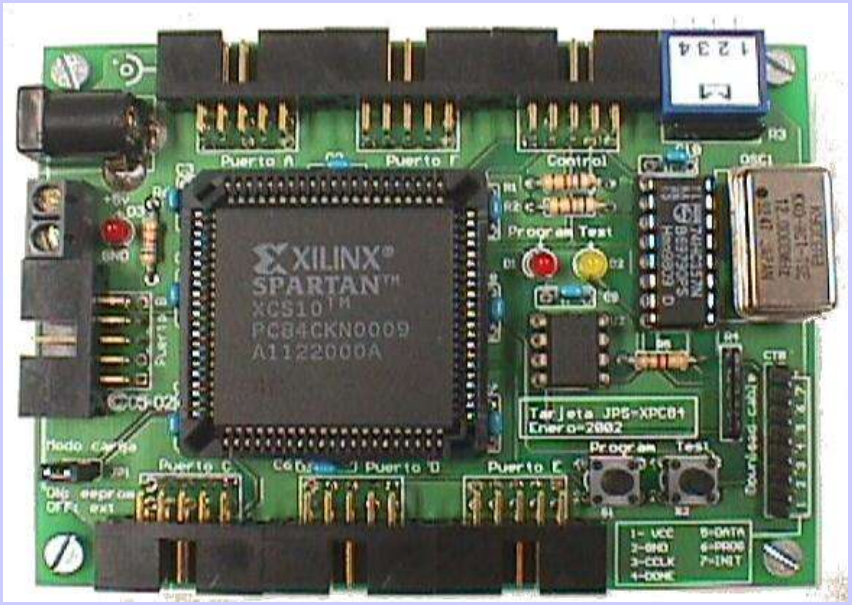


- **Lo más restrictivo: Diseños PPX**
  - Ej. **Tarjeta CT6811, Tarjeta CT293**
  - Orcad y Tango (Windows)



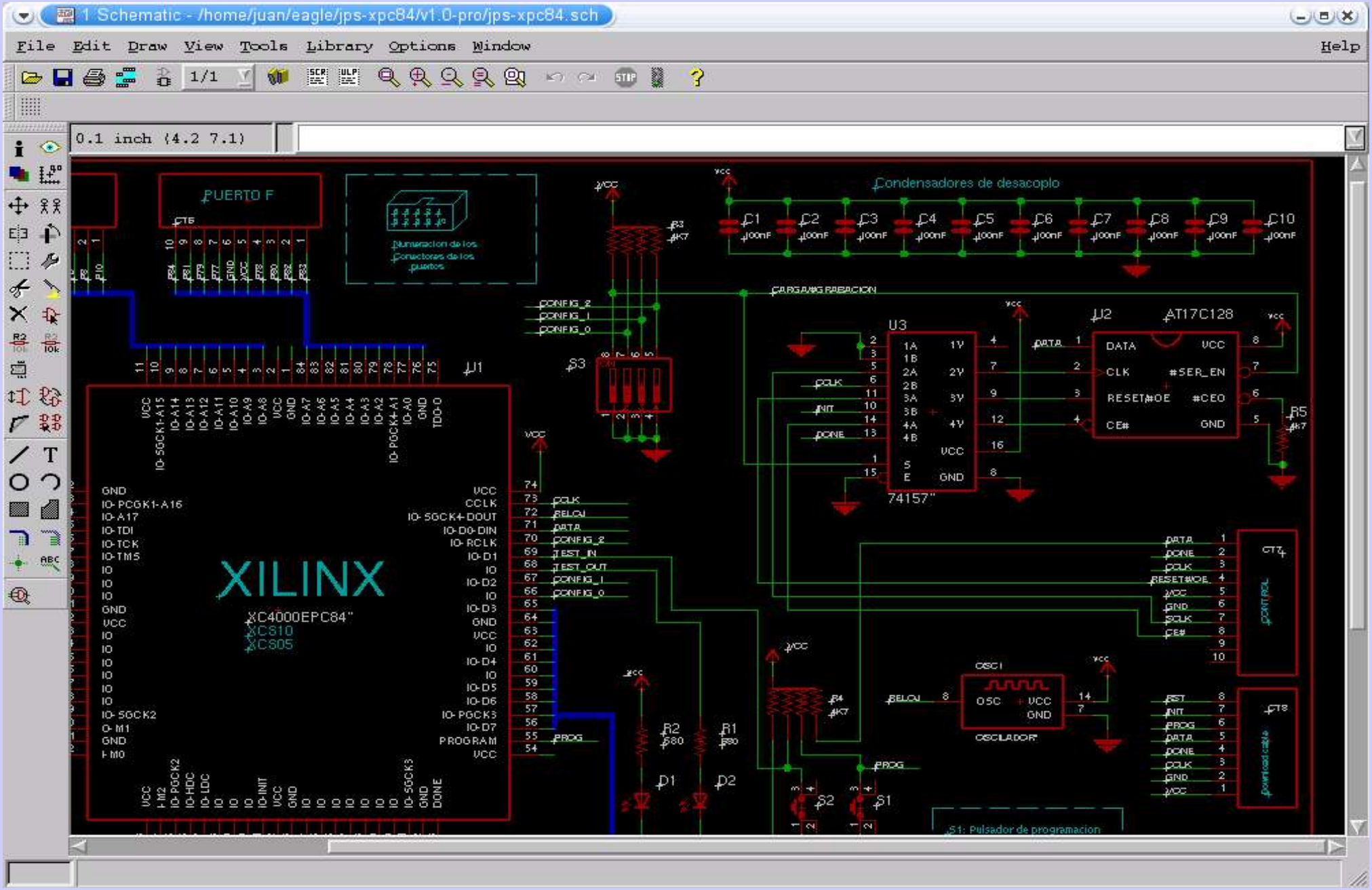
Un ejemplo: La Tarjeta JPS

- **Hardware libre. Tipo MML**.
  - Herramienta de diseño: **Eagle**
  - Distribución Linux: **Debian/Sarge**
- Cualquiera la puede fabricar
- Cualquiera la puede modificar
- Cualquier empresa la puede comercializar
- Cualquier Universidad la puede adaptar
- ...

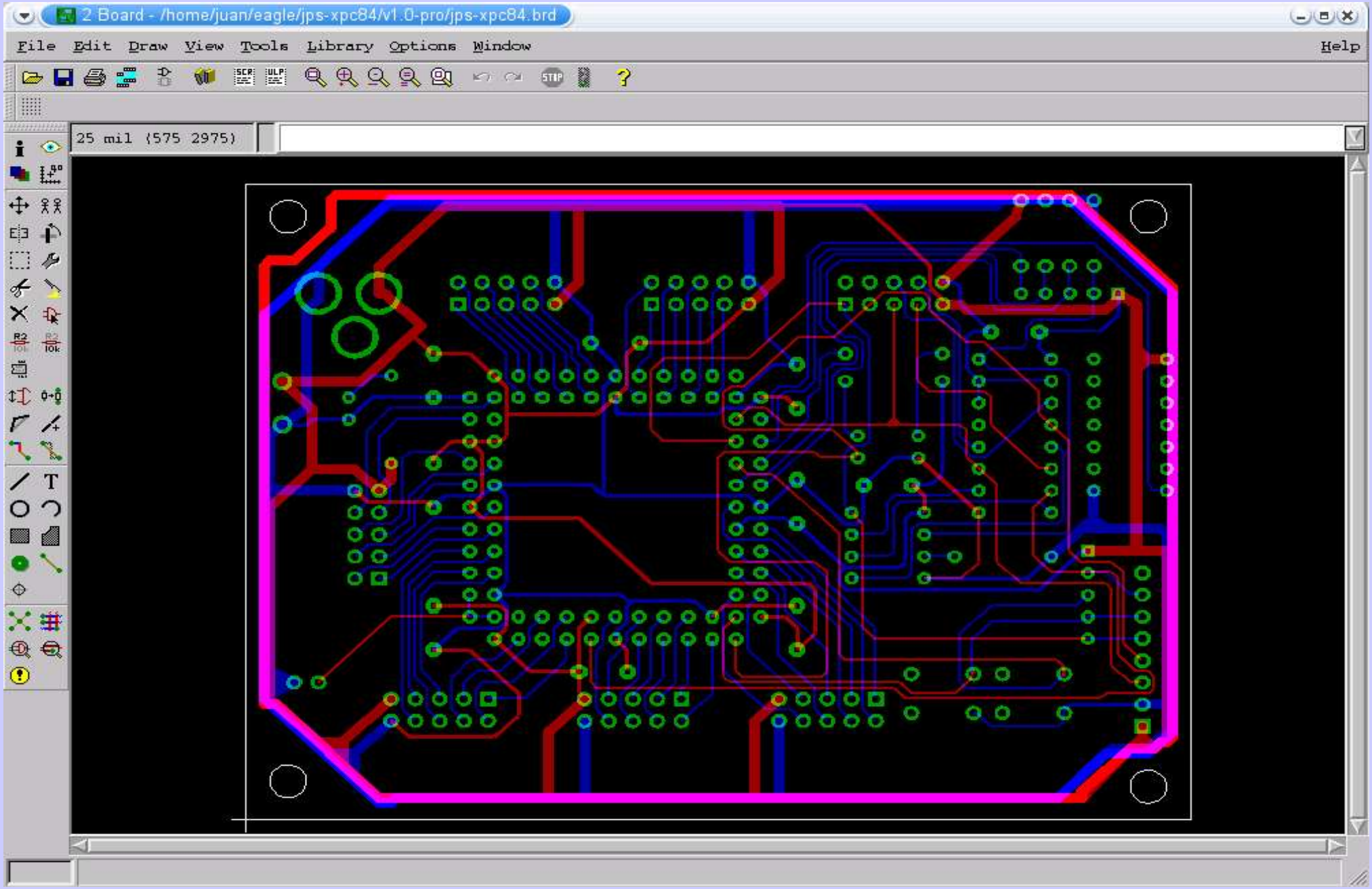


Presentada en Hispabot 2003

La aplicación EAGLE (No libre)



La aplicación EAGLE (No libre)



ÍNDICE

- **INTRODUCCIÓN**
- **PARTE I: Hardware estático**
- **PARTE II: Hardare reconfigurable**
- **APLICACIONES**
- **CONCLUSIONES**

Introducción

- **Hardware Reconfigurable:** Viene descrito mediante un lenguaje de descripción hardware (HDL) y se puede sintetizar en una FPGA
- **Los diseños son "Código Fuente"**
- **Para hacer que sean libres, sólo hay que aplicar la licencia GPL o similar.**

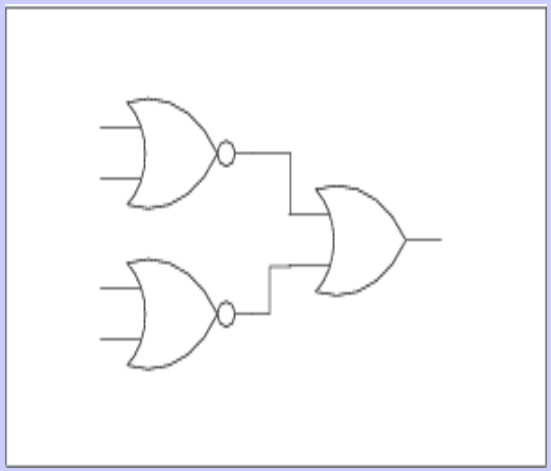
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity cont8 is
  port (clk   : in std_logic; -- Reloj
        clear : in std_logic;
        q     : out std_logic_vector (7 downto 0)); --Salida
end cont8;

architecture beh of cont8 is
signal cuenta : std_logic_vector (7 downto 0);
```

Lenguajes de descripción hardware (I)

- El hardware se puede describir utilizando un lenguaje HDL, como VHDL, Verilog, Handel C...
- Los diseños son ficheros de texto (“Código fuente”), que describen tanto la estructura del diseño como el comportamiento de las partes integrantes



mi_diseño.sch

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity cont8 is
  port (clk   : in std_logic; -- Reloj
        clear : in std_logic;
        q     : out std_logic_vector (7 downto 0)); --Salida
end cont8;

architecture beh of cont8 is
signal cuenta : std_logic_vector (7 downto 0);
```

mi_diseño.vhdl

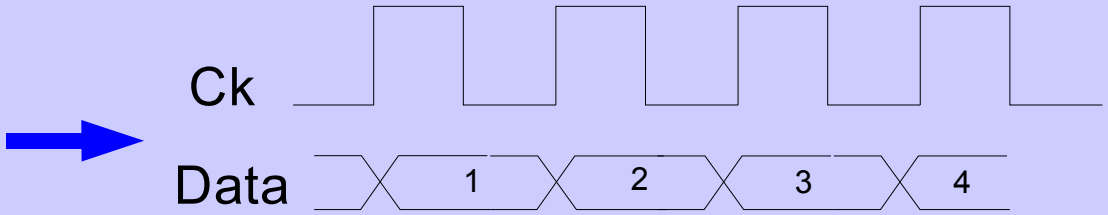
Lenguajes de descripción hardware (II)

- Se pueden realizar simulaciones

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity cont8 is
  port (clk   : in std_logic; -- Reloj
        clear : in std_logic;
        q     : out std_logic_vector (7 downto 0)); --Salida
end cont8;

architecture beh of cont8 is
signal cuenta : std_logic_vector (7 downto 0);
```



- Se pueden tener librerías de componentes, igual que con el software

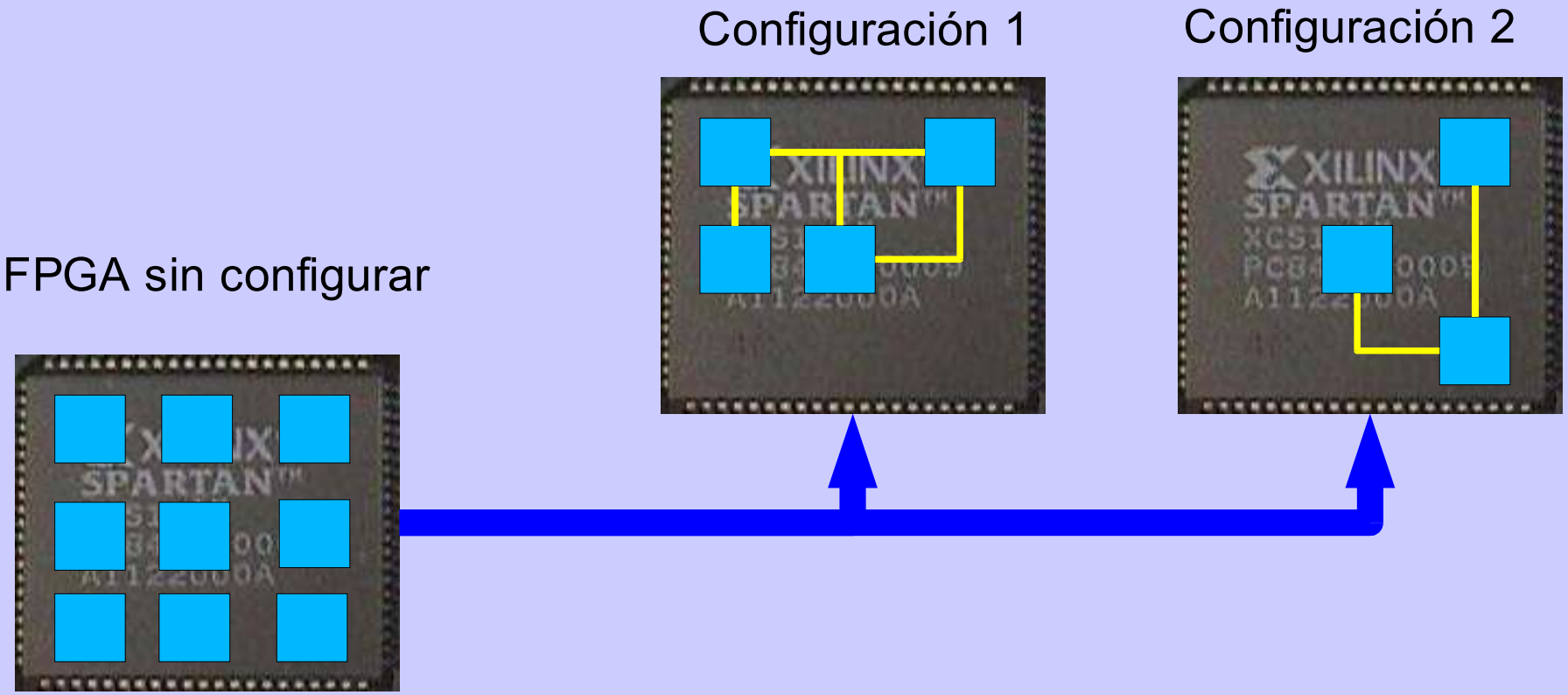
Mi_mux.vhdl

Mi_cpu.vhdl

Mi_dispositivo.vhdl

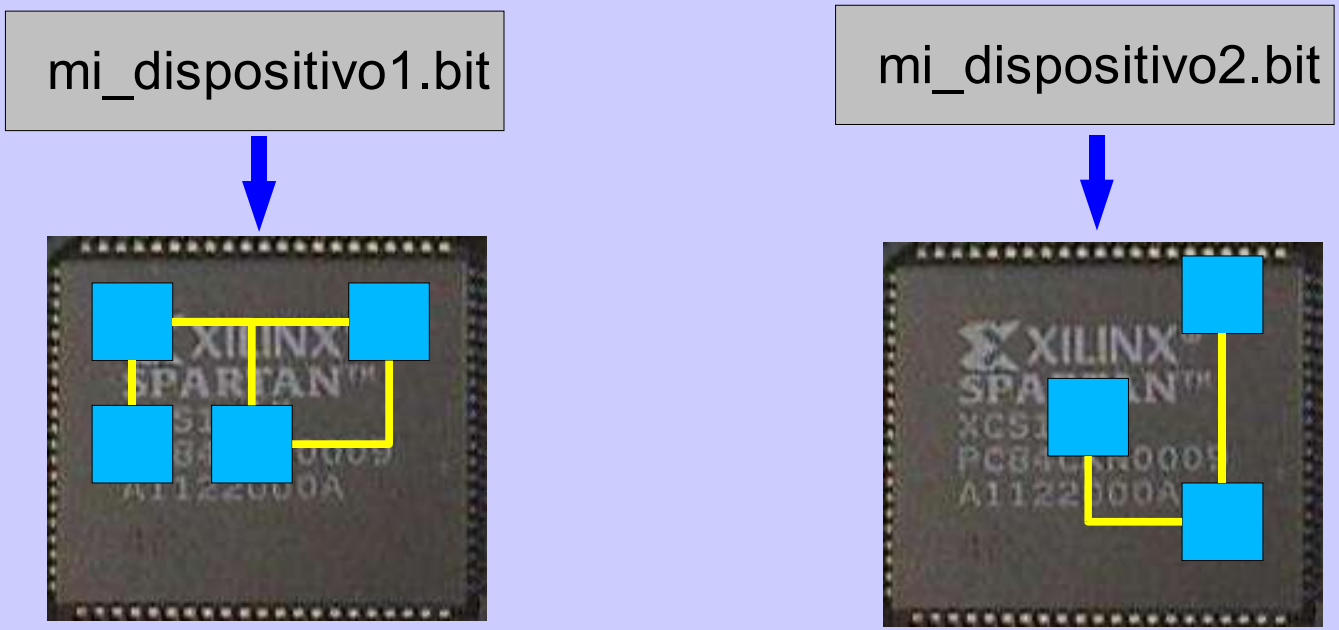
FPGAs (I)

- Dispositivos electrónicos que nos permiten implementar circuitos digitales
- **Compuestas por bloques iguales configurables** (CLBs) que se unen dinámicamente según se especifica en una memoria de configuración



FPGAs (II)

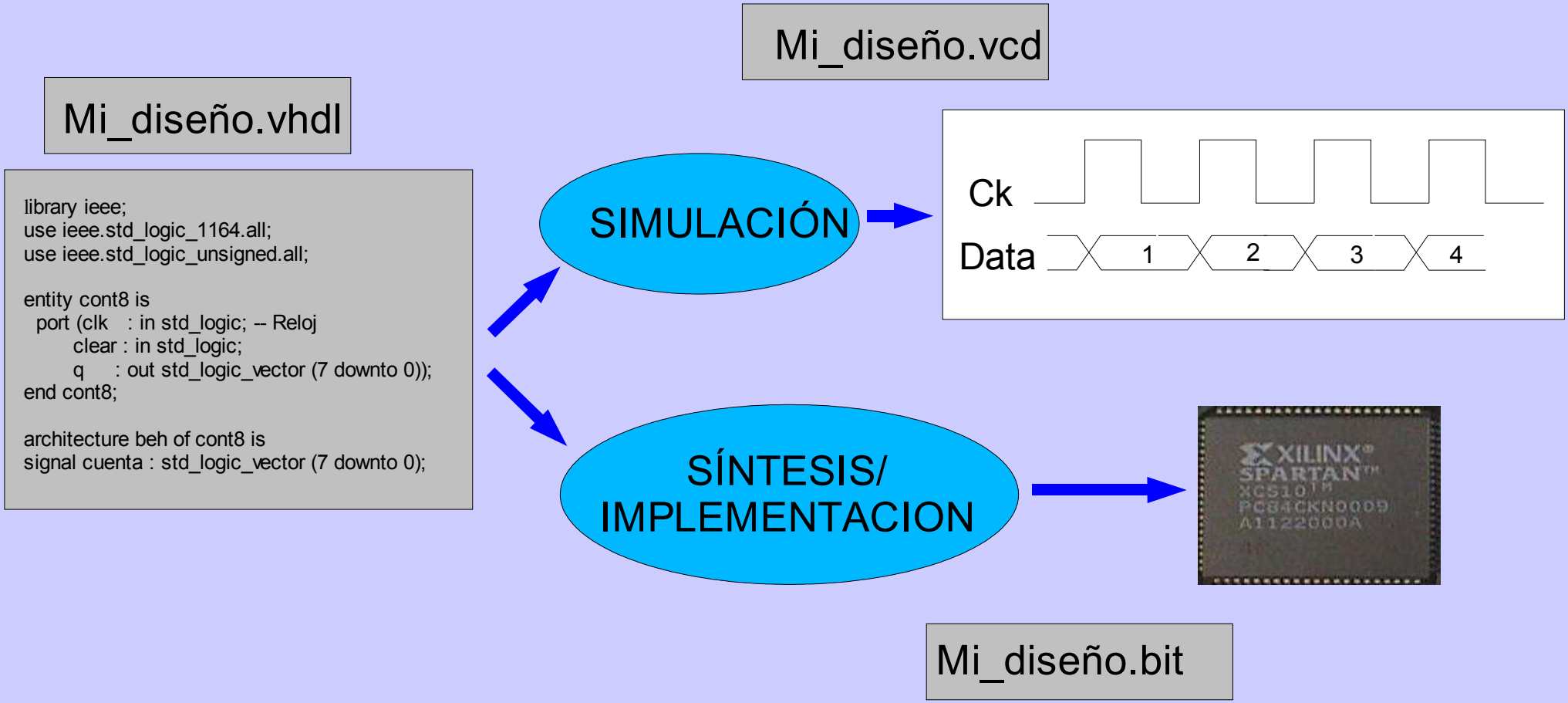
- Cambiando el contenido de la memoria de configuración, la FPGA se convierte en un Dispositivo u otro.
- El fichero que contiene la configuración se llama **Bitstream**



- **¡¡Dispositivos universales!!** Se pueden "convertir" en cualquier circuito digital, según la configuración que se le cargue

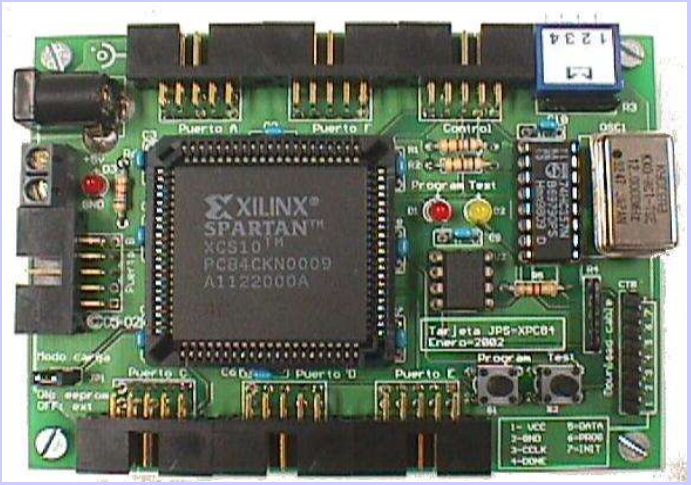
Hardware Reconfigurable (I)

- HDL+FPGA = Hardware reconfigurable
- ¡¡El hardware es ahora un software!!
- Hardware libre = Ficheros HDL con licencia GPL



Hardware Reconfigurable (II)

- El hardware reconfigurable se puede **compartir**
- Se pueden ofrecer las mismas **4 libertadas** a los diseños en HDL (Licencia GPL)
- Nuevas **comunidades Hardware** que comparten información (Ejemplo: OpenCores)
- **Repositorios con Hardware** para que cualquiera los use
- Necesaria una **plataforma** en la que descargar los diseños
  - Muchas entrenadoras en el mercado
  - La **JPS** es una de ellas, que además libre.

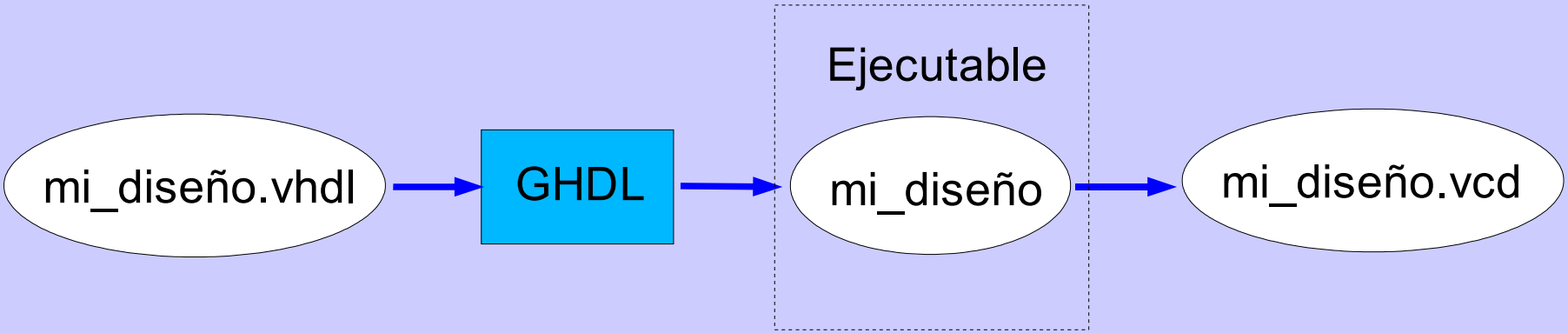


Hardware Reconfigurable

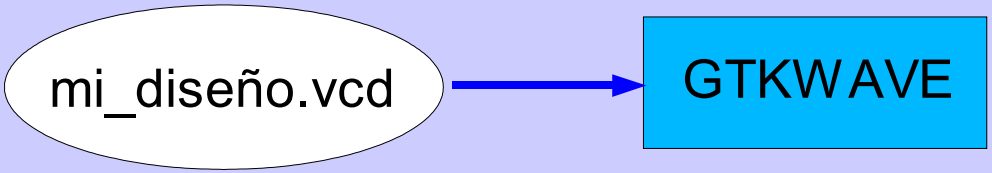
- **Simulación en GNU/Linux**
- **Síntesis/Implementación en GNU/Linux**

Simulación en GNU/LINUX

- Lenguaje empleado: **VHDL**
- Herramienta para Analizar/simular: **GHDL**
  - Proyecto prometedor
  - Basado en el GCC

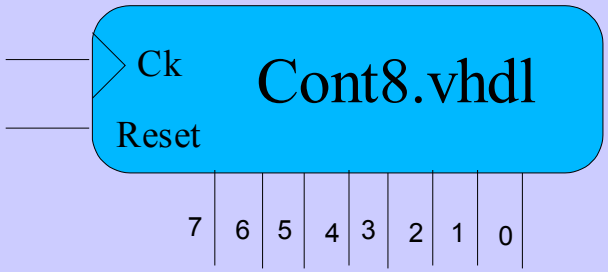


- Para visualizar el resultado de la simulación: **GTKWAVE**

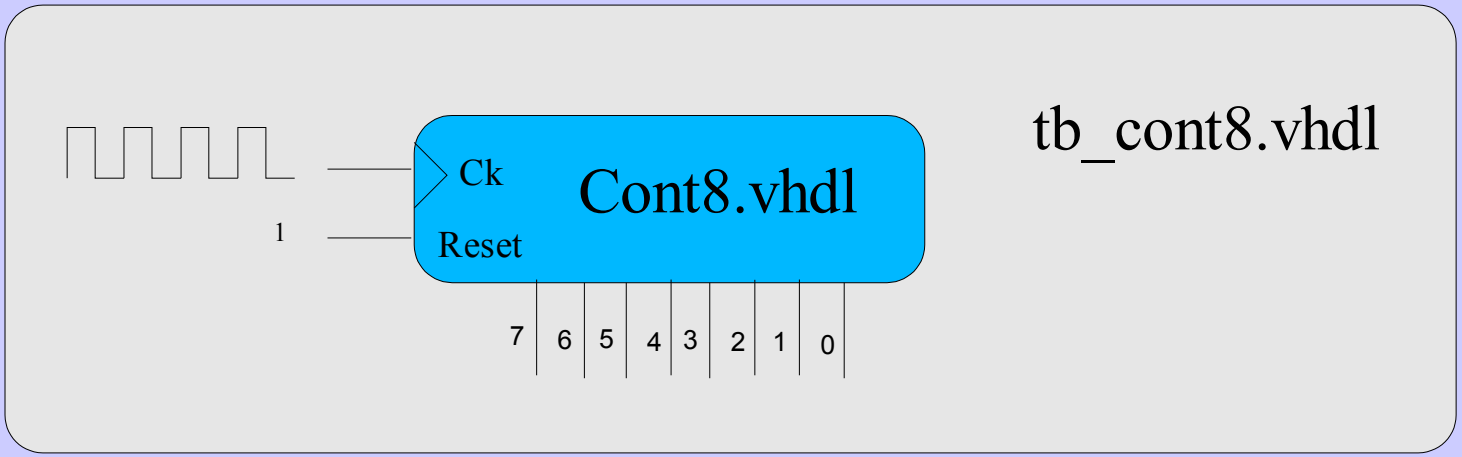


Demo de Simulación en GNU/LINUX (I)

- **Contador binario de 8 bits**



- Para probarlo hay que conectarlo a una señal de reloj y hacer un reset : **Banco de pruebas**, **tb_cont8.vhdl**



Demo de Simulación en GNU/LINUX (II)

cont8.vhdl

```
Library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity cont8 is
  port (clk   : in std_logic; -- Reloj
        clear : in std_logic;
        q     : out std_logic_vector (7 downto 0)); --Salida
end cont8;

architecture beh of cont8 is
signal cuenta : std_logic_vector (7 downto 0);

begin
  output: process(clk,clear)
  begin
    if (clear='0') then
      q<="00000000";
      cuenta<="00000000";
    elsif (clk'event and clk='1') then
      cuenta<=(cuenta+1);
      q<=cuenta;
    end if;
  end process;
end beh;
```

Demo de Simulación en GNU/LINUX (II)

$ ghdl -a –ieee=synopsys *.vhdl

Análisis

Cont8.o tb_cont8.o

$ ghdl -e –ieee=synopsys tb_cont8

Elaboración

tb_cont8 (Ejecutable)

$ ./**tb_cont8** –vcd=simulacion.vcd –stop-time=200ns

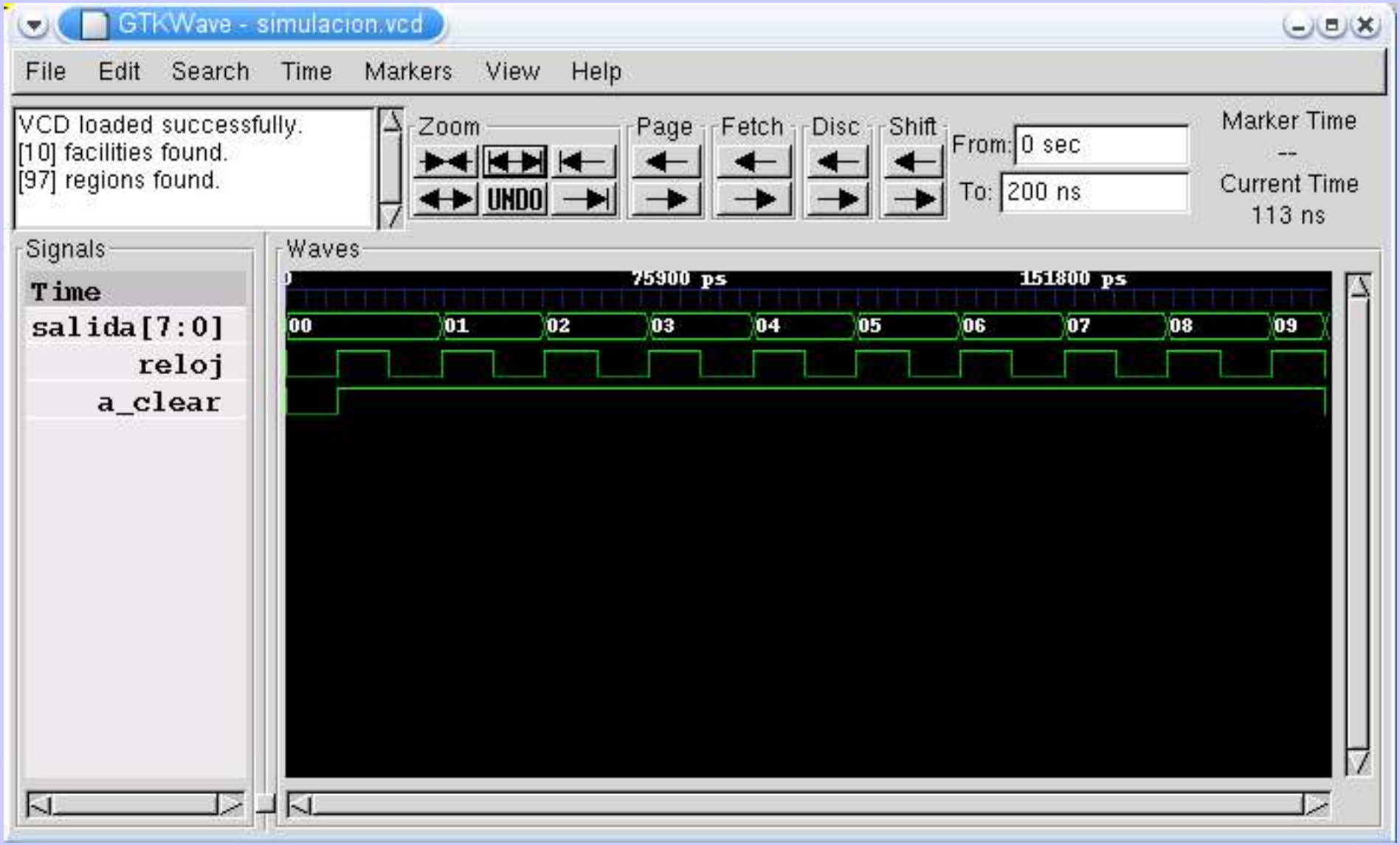
Simulación

Simulacion.vcd

$ gtkwave simulacion.vcd

Visualización

Visualización con GTKWAVE



Hardware Reconfigurable

- Simulación en GNU/Linux
- **Síntesis/Implementación en GNU/Linux**

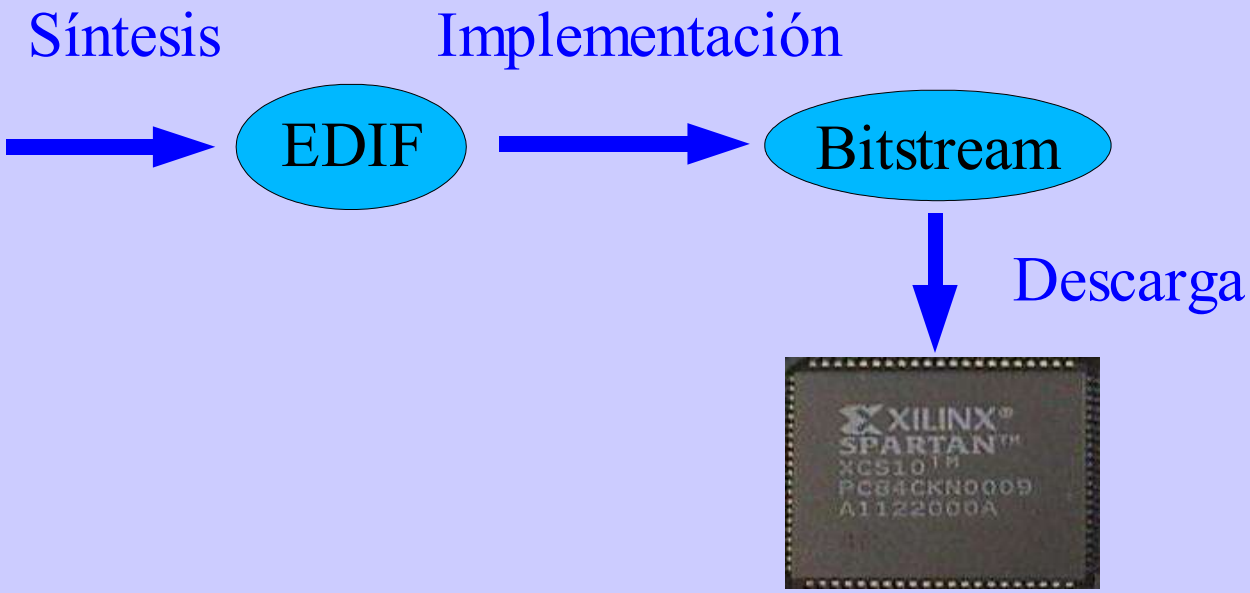
Introducción

- **Fase de síntesis**: A partir del código en VHDL se obtiene un fichero netlist en el formato EDIF
- **Fase de implementación**: A partir del netlist se genera el bitstream
- **Fase de descarga**: Finalmente se descarga el Bitstream en la FPGA

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity cont8 is
  port (clk   : in std_logic; -- Reloj
        clear : in std_logic;
        q     : out std_logic_vector (7 downto 0));
end cont8;

architecture beh of cont8 is
signal cuenta : std_logic_vector (7 downto 0);
```



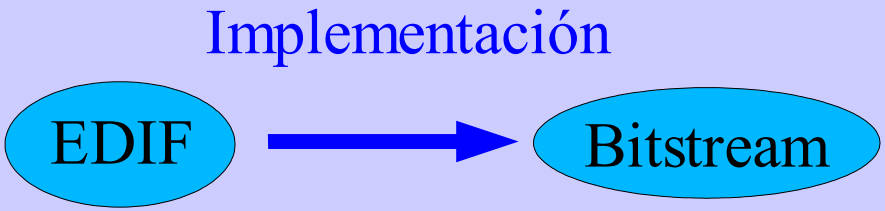
Síntesis en FPGA

- **No hay herramientas libres**, pero podría haberlas.
  - Se podrían realizar
  - Es dependiente de la tecnología, pero el fabricante aporta la información necesaria
  - El formato EDIF está documentado
- **Existen herramientas propietarias**, que corren bajo linux (Ej. Synplify Pro)
- **Existen herramientas que corren bien bajo WINE** (Ej. ISE)

Síntesis ✓

Implementación en FPGA

- **Se obtiene el *Bitstream* a partir del fichero EDIF**



- **La tecnología de los dispositivos FPGA se considera secreto industrial**
  - No es posible conocer toda la información de configuración

No es posible el diseño de herramientas libres

- **Herramientas**
  - Actualmente los fabricantes más importantes no tienen herramientas para Linux
  - Xilinx ha anunciado que estarán disponibles en 6 meses

Cerrando el ciclo de diseño de FPGAs en Linux

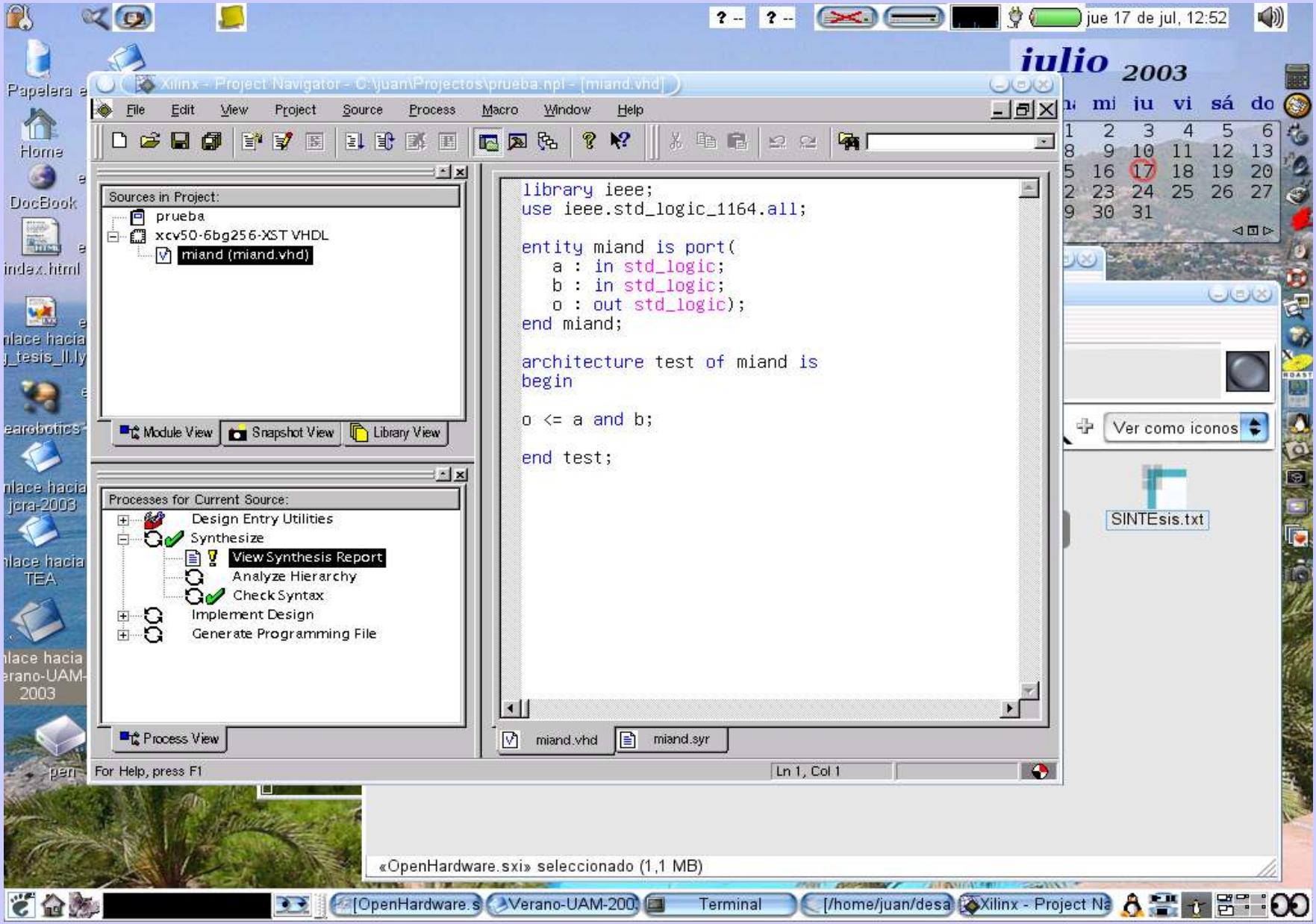
- **Existen herramientas de edición libres** ✓
- **Existen herramientas de simulación libres** ✓
- **Existen herramientas de síntesis pero no son libres** ✗
  - Al menos están disponibles para Linux
- **No hay herramientas libres para la implementación, ni puede haberlas** ✗

Solución que hemos empleado:

Simulación: GHDL + GTKWAVE
Síntesis/implementación: ISE 4.2 + WINE

Cerrando el ciclo de diseño de FPGAs en Linux

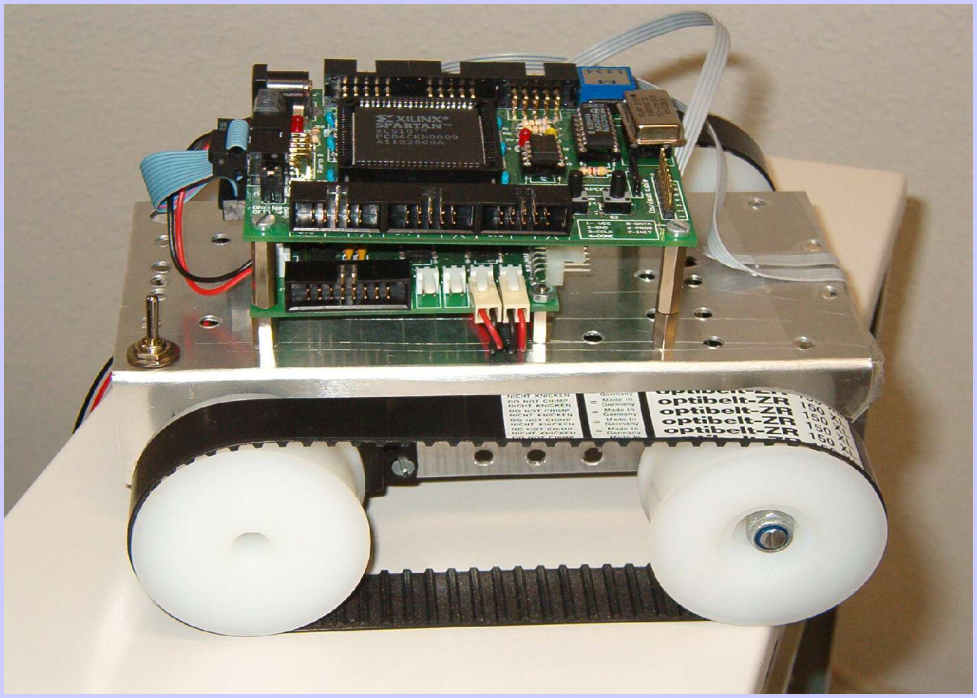
- Y para muestra un botón...

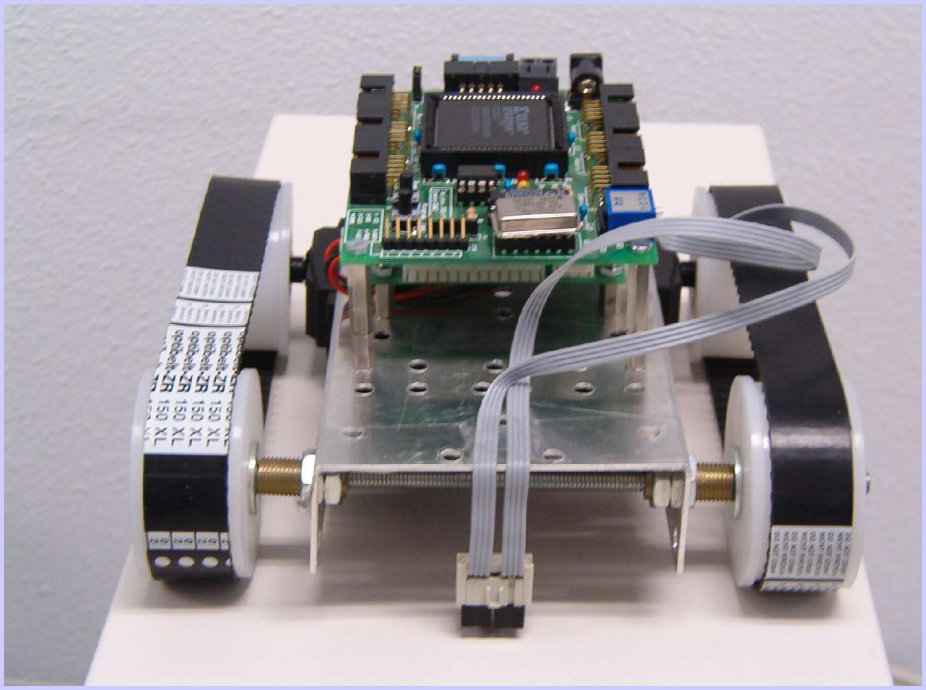


ÍNDICE

- **INTRODUCCIÓN**
- **PARTE I: Hardware estático**
- **PARTE II: Hardare reconfigurable**
- **APLICACIONES**
- **Conclusiones**

Aplicación I: Robot de Docencia

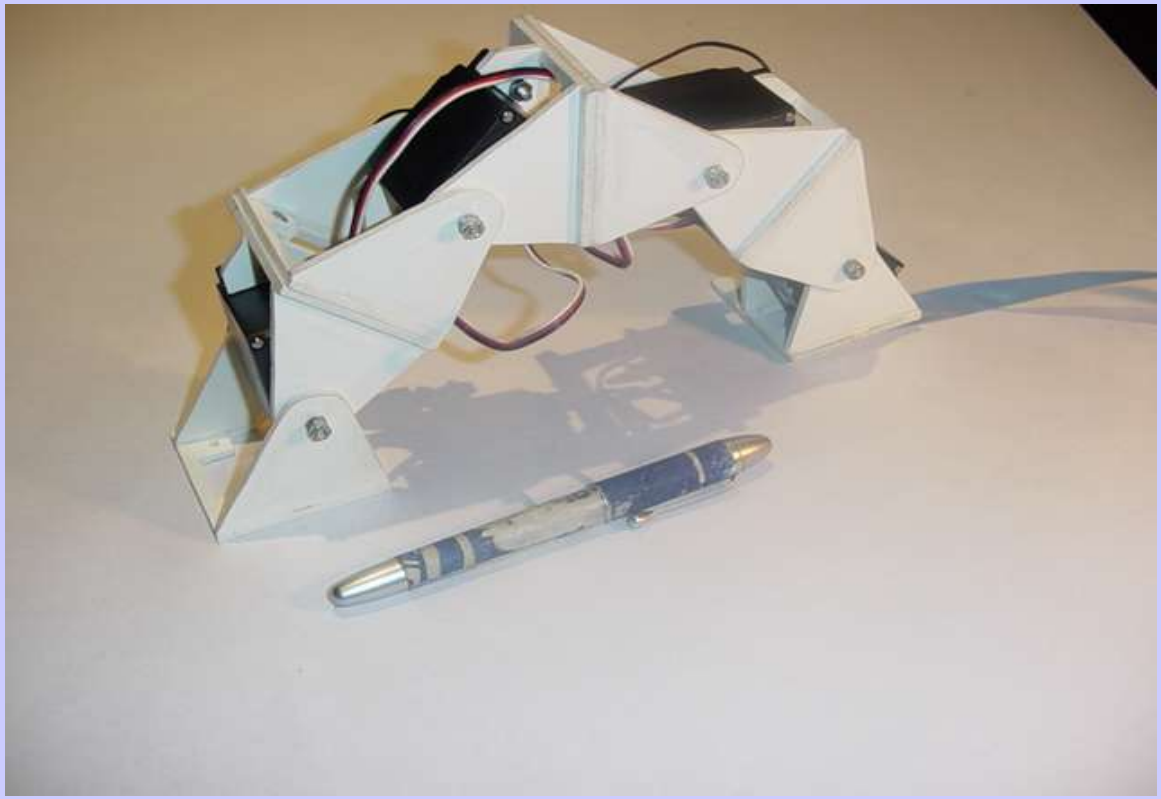




- Robot seguidor de línea
- CPU PandaBear en VHDL
  - Risc, 16 Bits
  - 8 instrucciones

- Ocupación: 147 CLBs (75%)
- Frecuencia 12MHZ

Aplicación II: Robot gusano



- Robot ápodo que se desplaza mediante ondas sinusoidales
- Controlador hardware en una FPGA. Presentado en el JCRA 2003.

ÍNDICE

Conclusiones (I)

- **Al hablar de hardware libre hay que distinguir entre hardware estático y hardware reconfigurable**
- **Hardware estático**
  - Propuesta una definición
  - Establecida clasificación según las restricciones impuestas por las aplicaciones de diseño
  - Es el autor es que decide la libertad, no la aplicación
- **Hardware reconfigurable**
  - Es libre si se aplica licencia GPL o similar
  - Herramientas libres para la simulacion: GHDL, GTKWAVE
  - No hay sintetizadores libres, pero podría haberlos
  - No puede haber un entorno completamente libre. Los detalles internos de las FPGAs son secreto industrial

Conclusiones (II)

- **Hemos cerrado el ciclo de diseño en una máquina GNU/Linux**
  - Tarjeta JPS
  - Entorno ISE 4.2 de Xilinx, usando Wine

Hardware Libre tiene mucho futuro, sobre todo el **hardware reconfigurable** por su similitud con el software libre. La limitación viene impuesta por las herramientas de los Fabricantes de FPGAs

El **hardware estático libre** está más limitado, y su difusión es mucho más lenta, debido a que hay que "Fabricar". Tiene especial interés en su aplicación docente, donde las universidades y centros de investigación los podrían fabricar

Conclusiones (III)

- **SOLUCIONES**:
  - Utilizar **Laboratorios Virtuales**, donde hay un servidor con el programa propietario que permite hacer la síntesis e implementación: GRANJAS DE SINTESIS
  - Proyectos Europeos a gran escala: ¿Open FPGA?

En cualquier caso, el hardware libre es un nuevo frente de batalla hacia una sociedad del conocimiento libre

Referencias:

- **Tarjeta JPS** :
  - http://www.iearobotics.com/personal/juan/doctorado/jps-xpc84/jps-xpc84.html
  - http://www.iearobotics.com, google interno: "Tarjeta JPS"
- **LABOWEB**: Laboratorio Virtual
  - http://www.ii.uam.es/~laboweb/
- **PROYECTO HARDWARE ABIERTO DE MICROBOTICA S.L:**
  - http://www.microbotica.com/ha.htm
- **TARJETA CT6811**:
  - http://www.iearobotics.com/proyectos/ct6811/ct6811.html
  - http://www.iearobotics.com, google interno: "Tarjeta CT6811"
- **OPENCORES**: Comunidad de hardware reconfigurable
  - http://www.opencores.org/

Hardware Libre:
Clasificación y desarrollo de hardware reconfigurable en entornos GNU/Linux

Juan González, Iván González, Francisco Gómez-Arribas

Escuela Politécnica Superior
Universidad Autónoma de Madrid