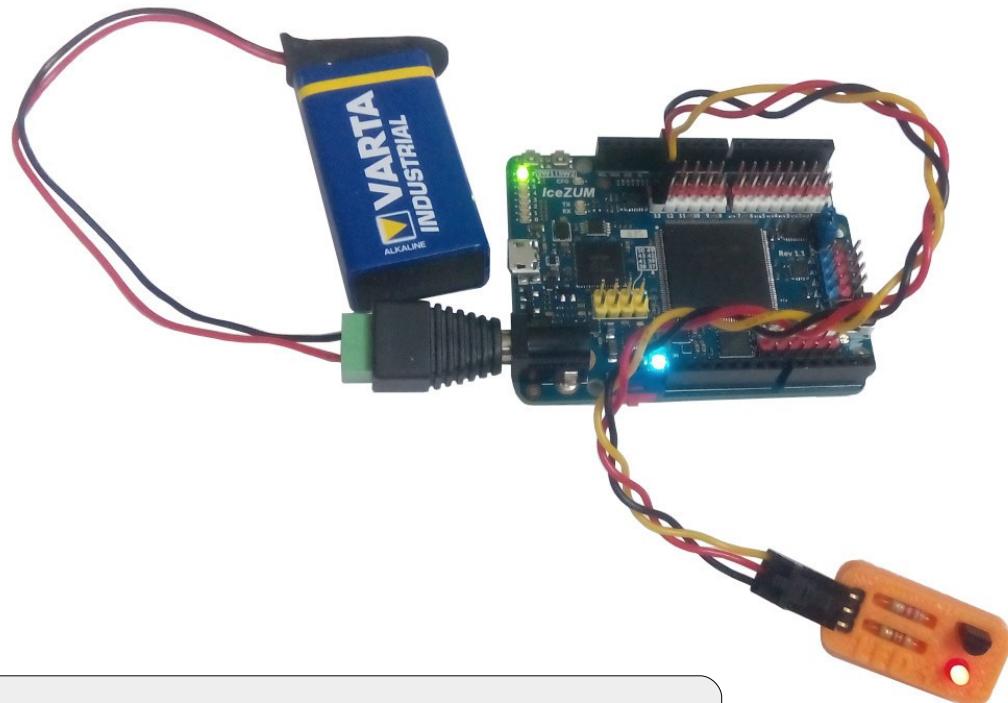


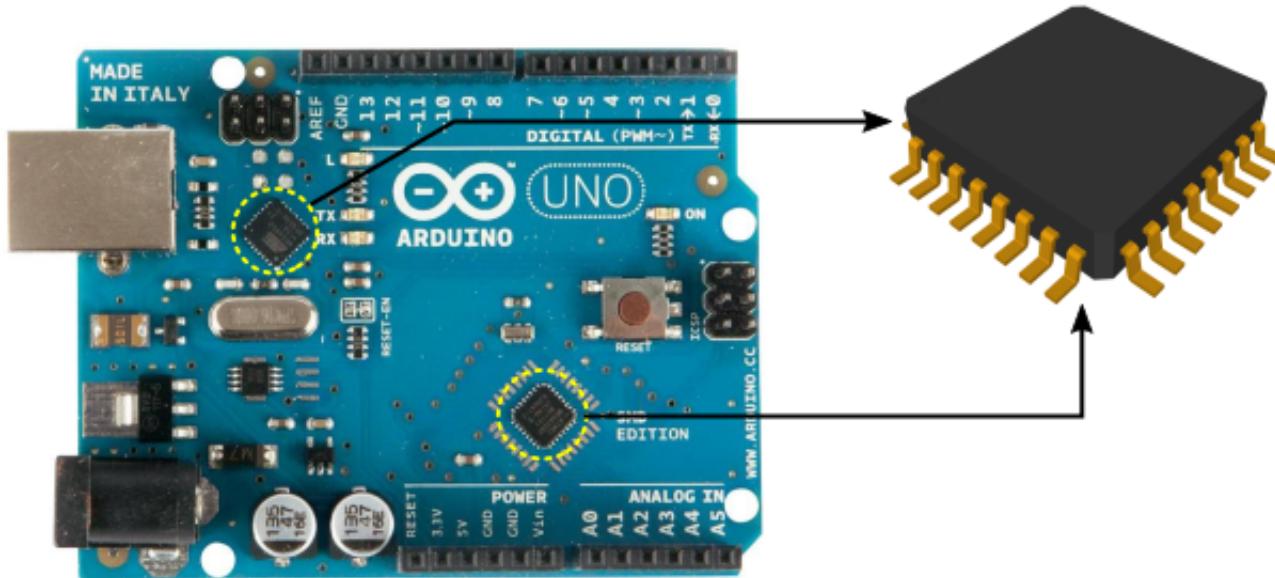
# FPGAs libres



Juan González Gómez (Obijuan)

<https://github.com/Obijuan>

# Chips digitales



- El gran invento del siglo XX
- Están por todos lados
- Muy baratos
- Los compramos y los usamos

- Son **cajas negras**
- No los podemos estudiar
- No los podemos modificar
- No los podemos compartir

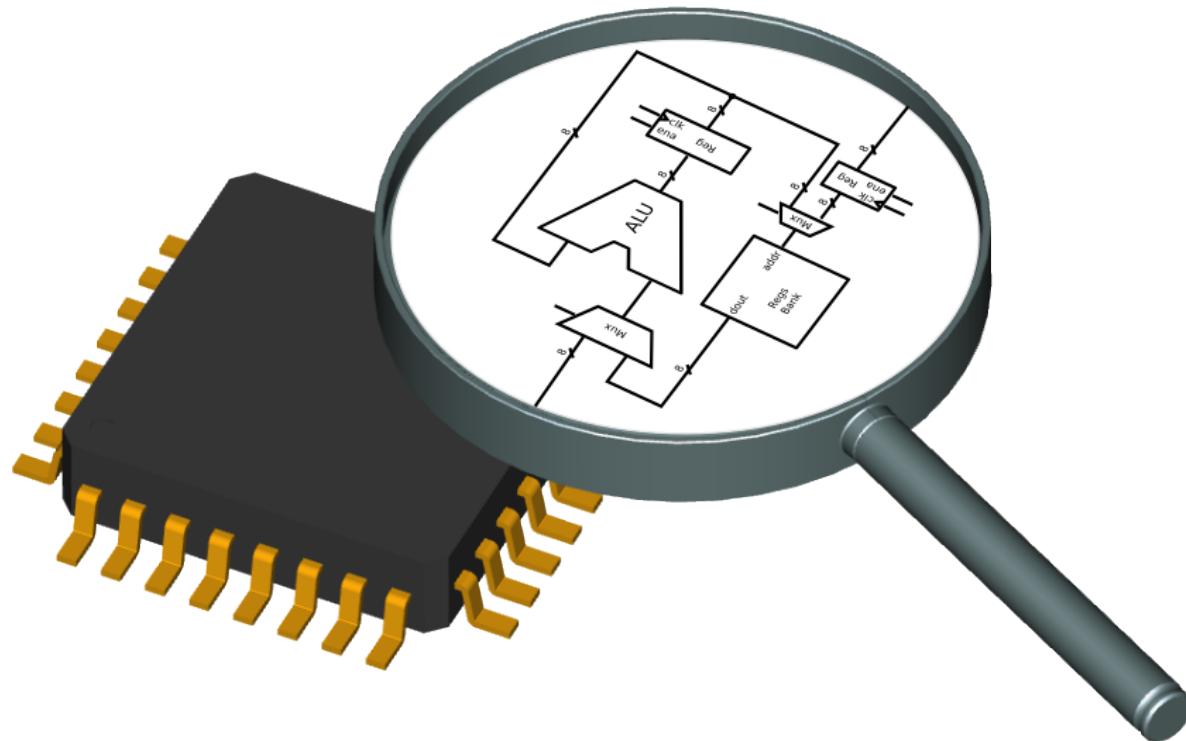
# FPGAs

¿Te imaginas hacer tus propios chips?  
¿Quieres aprender cómo son sus tripas?  
¿Cómo se diseñan?



Una primera aproximación es usar **FPGAs**

# Viaje al interior de los chips digitales

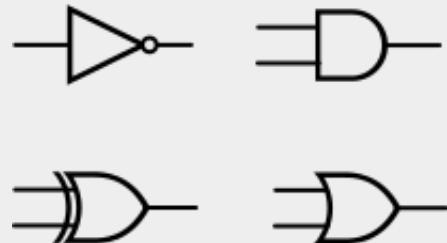


- Nivel de electrónica digital
- Información: Sólo 1s y 0s (Bits)
- Función: **Manipular, almacenar y transportar bits**

# Elementos en circuitos digitales

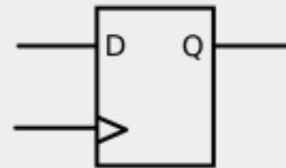
## Puertas lógicas

Manipulación bits



## Biestables

Almacenamiento bits



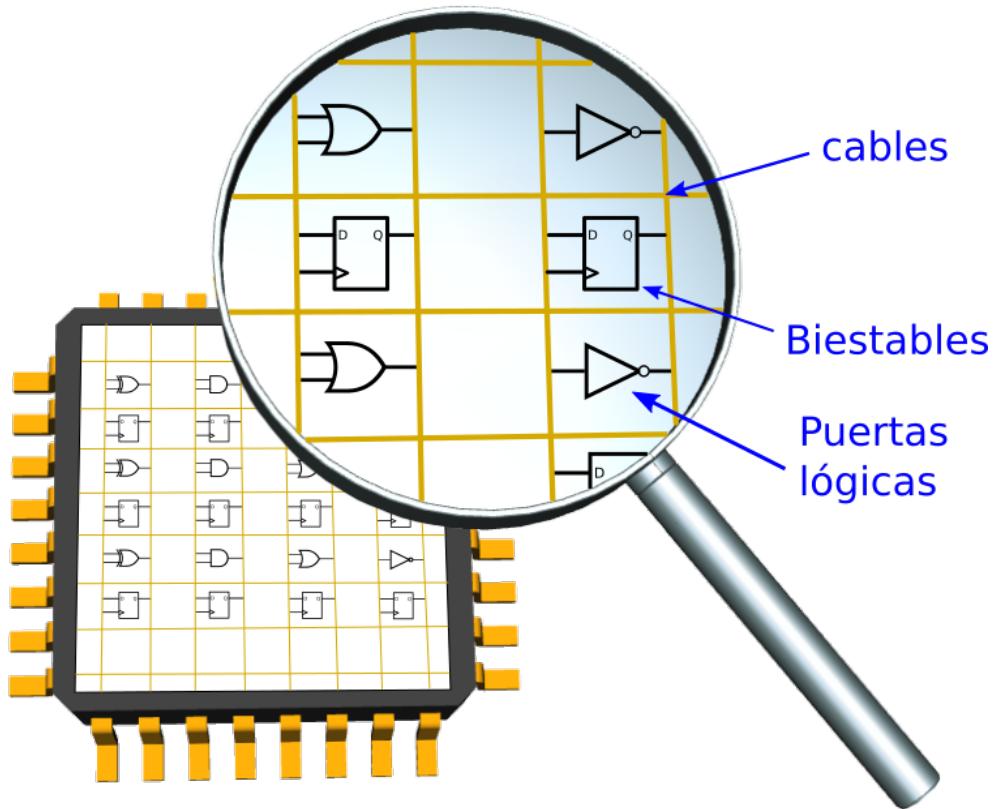
## Cables

Transporte bits



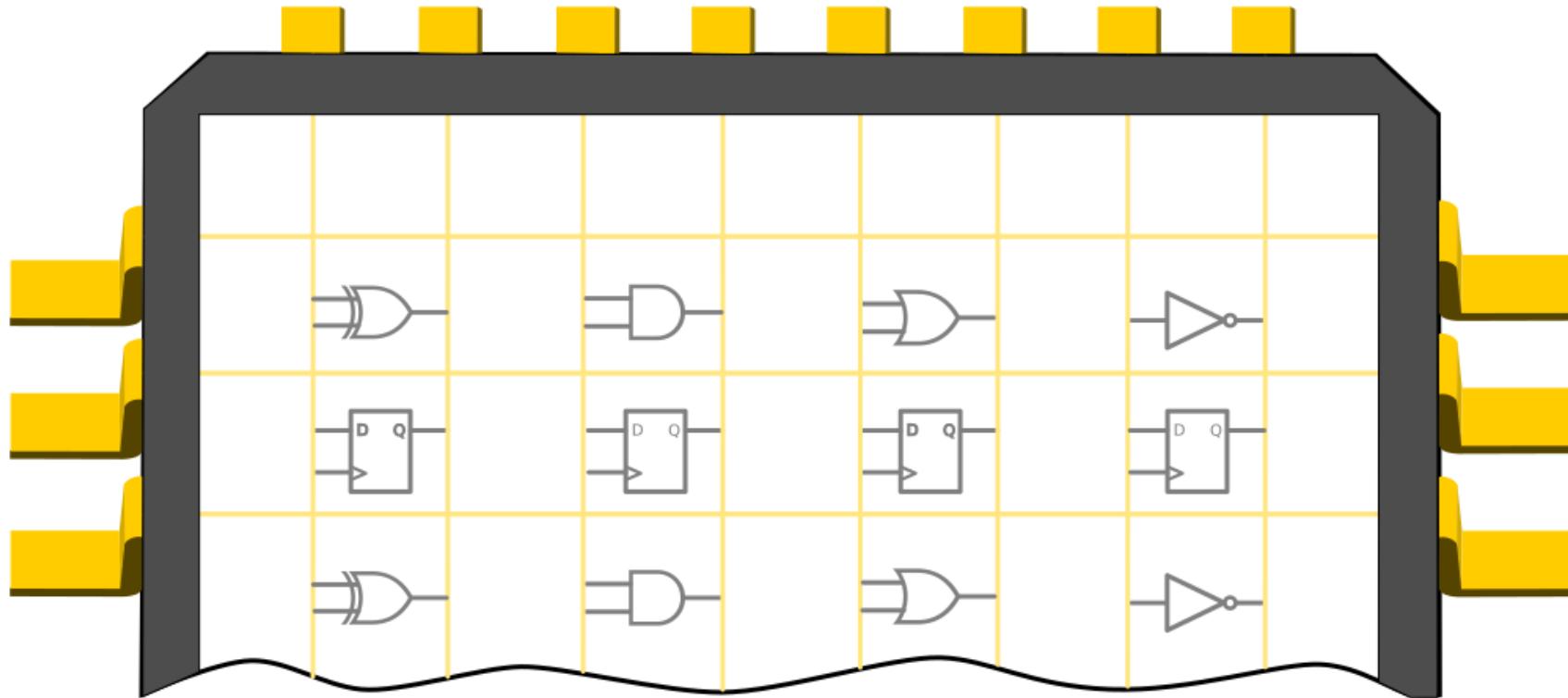
Cualquier circuito digital, por muy complejo que sea, se descompone en estos 3 tipos de **componentes elementales**

# Tecnología FPGA



**FPGA:** Chip “en blanco” que contiene una matriz con los 3 componentes básicos: puertas lógicas, biestables y cables

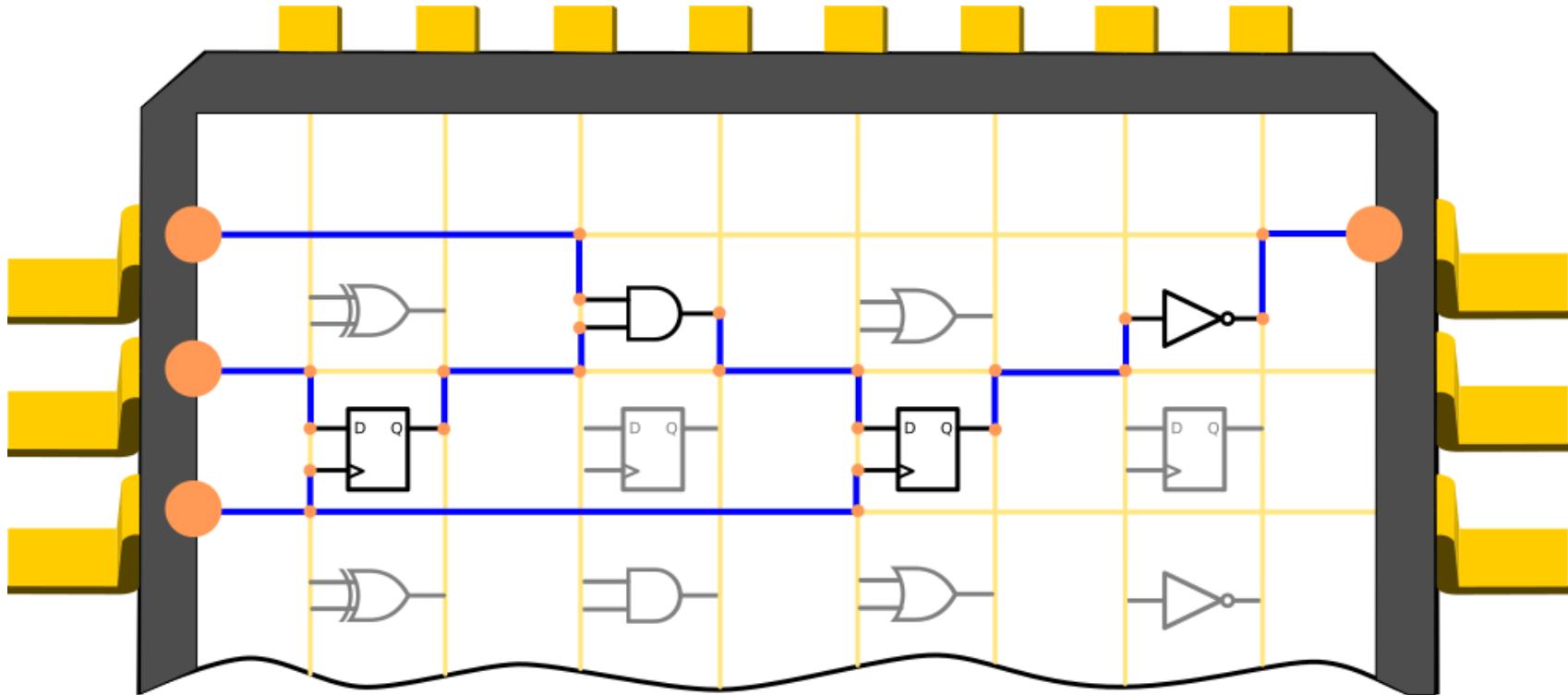
# Configuración



Las uniones entre cables son **configurables**

FPGA no configurada

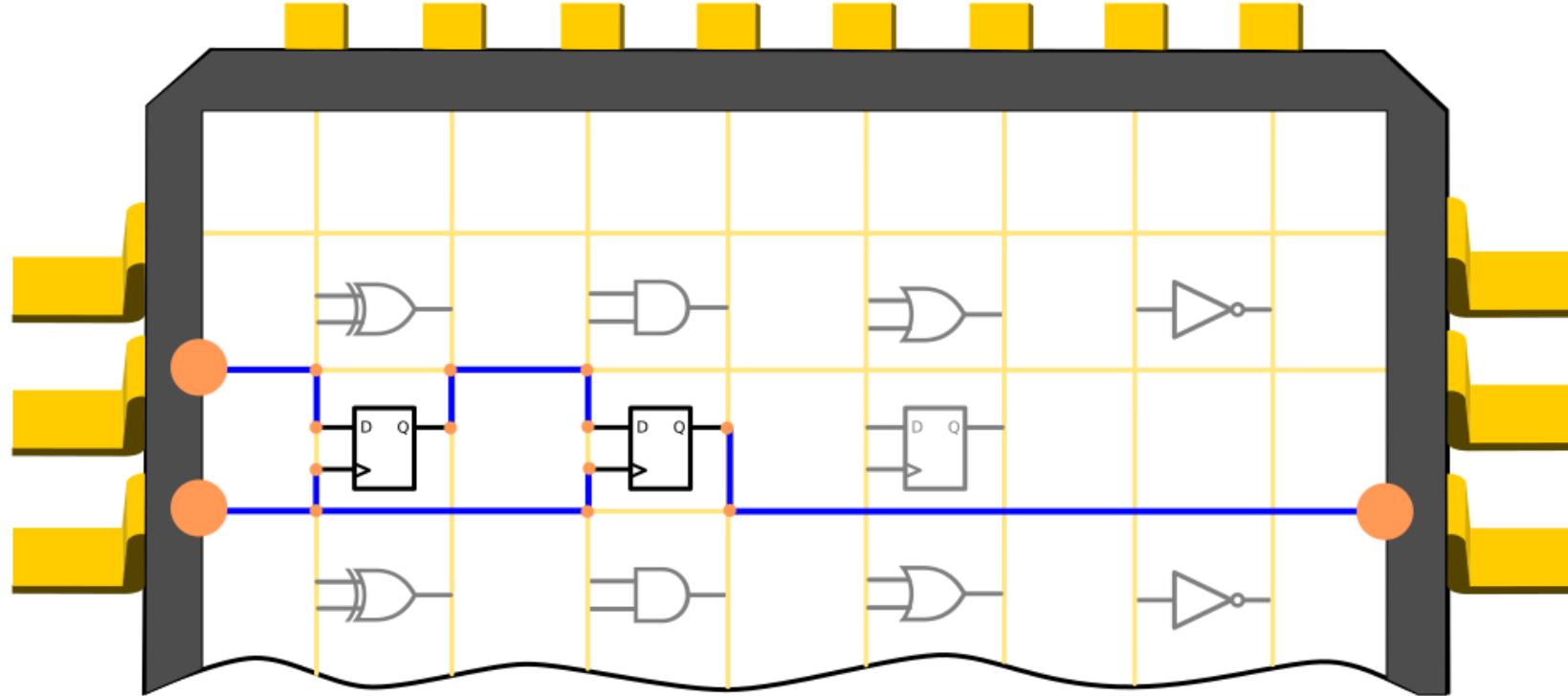
# Configuración



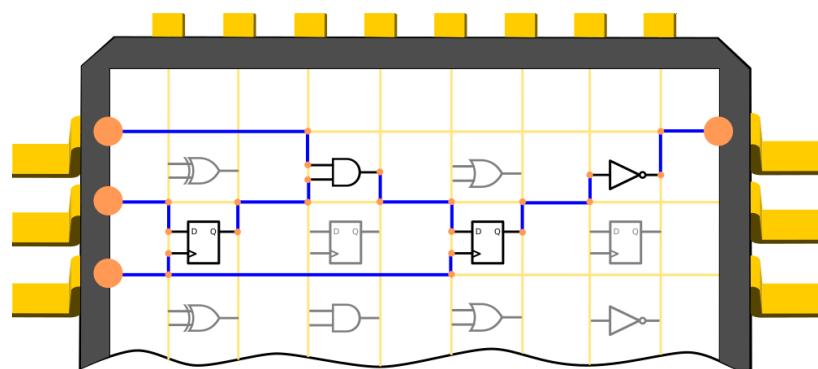
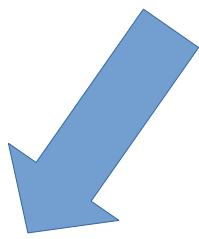
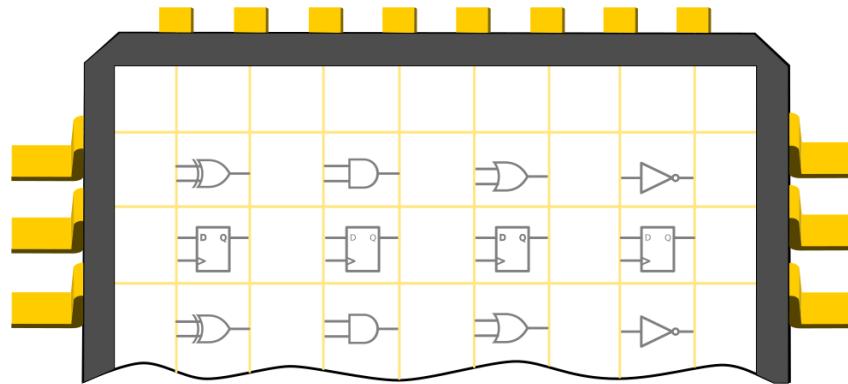
**Circuito creado** configurando las uniones entre los elementos básicos de la FPGA

FPGA configurada

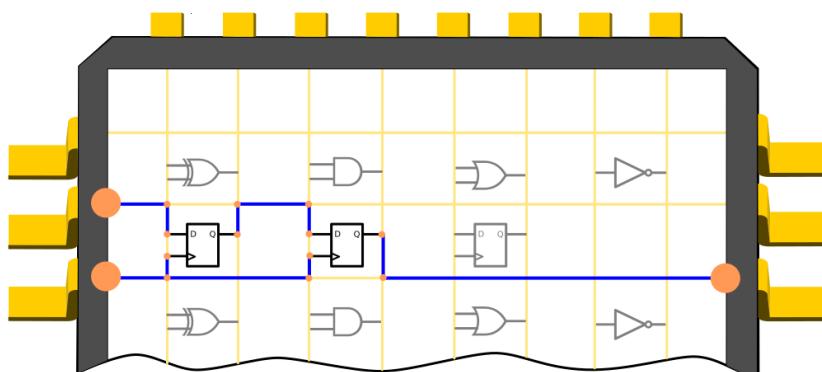
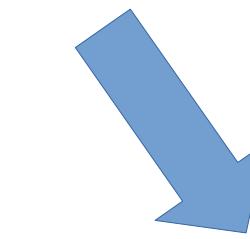
# Re-configuración



**¡Sólo con cambiar las uniones, aparece  
otro circuito diferente!**



**Circuito 1**

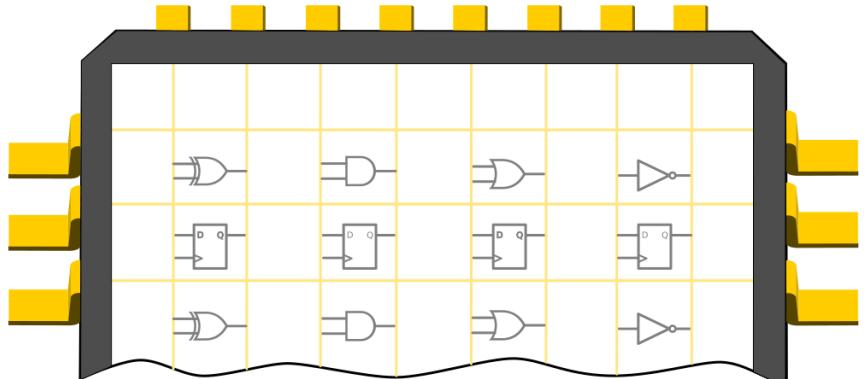


**Circuito 2**

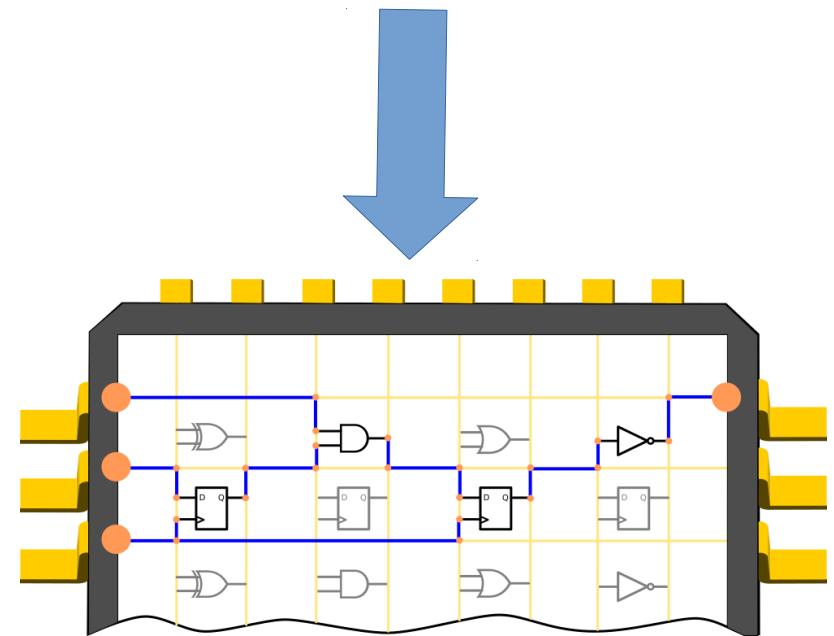
**¡FPGAs = Impresoras 3D de circuitos digitales!**

# Bitstream

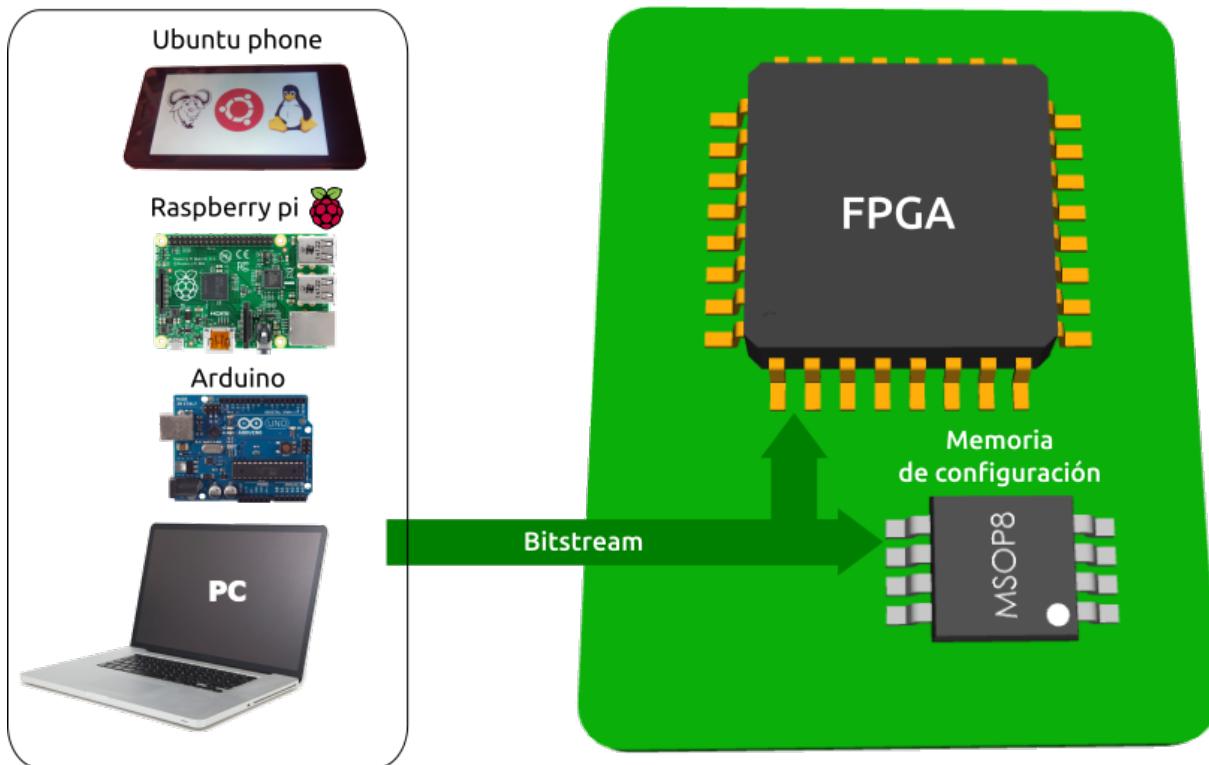
Bitstream  
...1001111100010011001010101.....



- La configuración se hace cargando un *bitstream* en la FPGA
- Estos bits determinan qué cables se conectan y cuales no

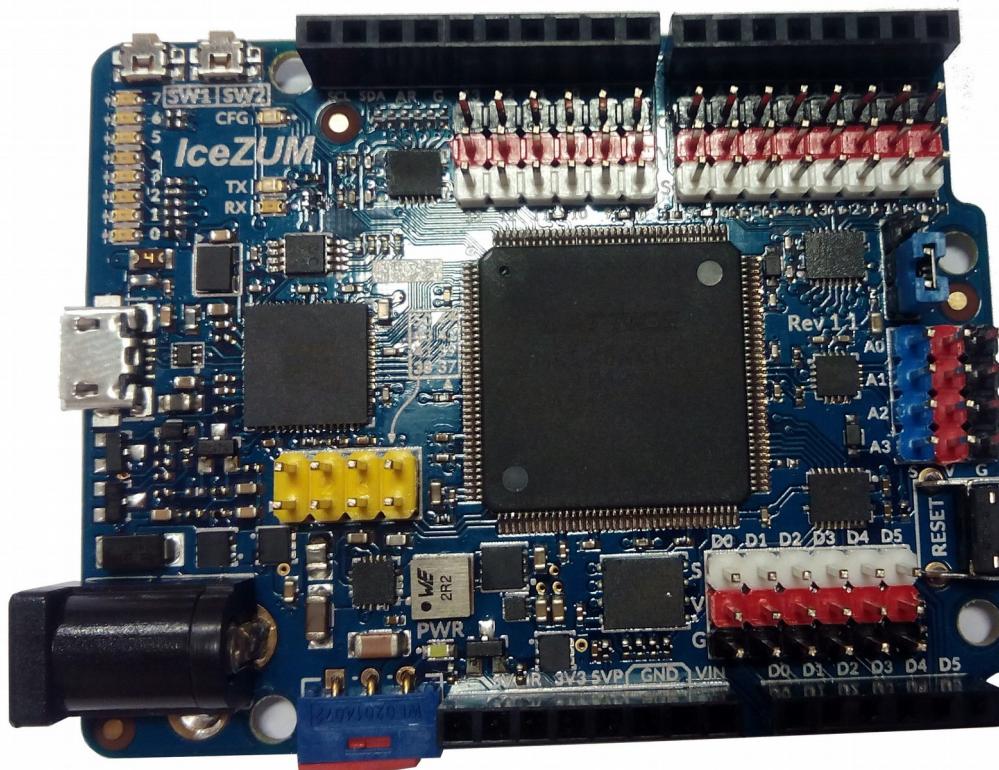


# Memoria de configuración



- **FPGA volátiles:** pierden su configuración al quitar alimentación
- El *bitstream* se guarda en una **memoria externa**: memoria de configuración
- Al arrancar la FPGA se carga con el *bitstream* de la memoria de configuración
- Desde un ordenador externo se carga el *bitstream* en la memoria de config.

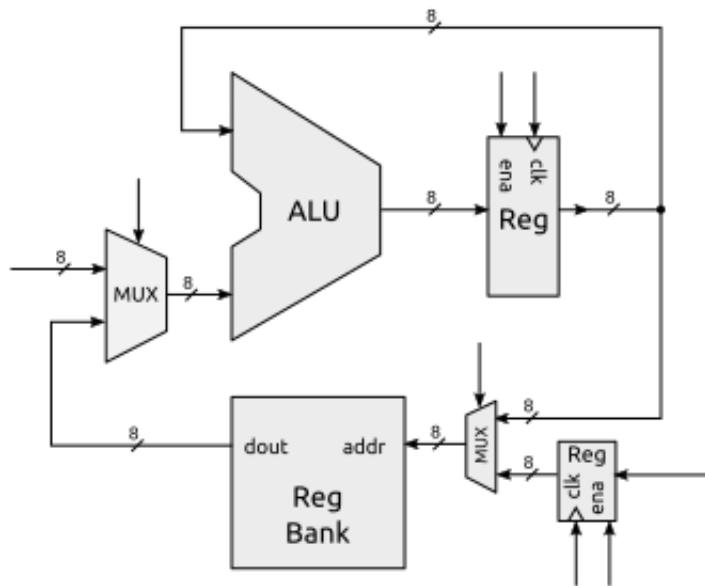
# Demo 1: Contador



Tarjeta Icezum Alhambra

<https://github.com/FPGAwars/icezum/wiki>

# Diseñando circuitos digitales



```
module simplez #(
    parameter BAUD = `B115200,
    parameter WAIT_DELAY = `T_200ms,
    parameter ROMFILE = "prog.list",
    parameter DEBUG_LEDS = 0
)(
    input wire clk,
    input wire rstn_ini,
    output wire [3:0] leds,
    output wire stop,
    output wire tx,
    input wire rx
);

reg [DW-1: 0] alu_out;
reg flag_z;

always @(*) begin
    if (alu_op2)
        alu_out = alu_in;
    else if (alu_clr)
        alu_out = 0;
    //-- Suma de operador 1 + operador 2
    else if (alu_add)
        alu_out = reg_a + alu_in;
    else if (alu_dec)
```

HDL

Los circuitos digitales modernos se diseñan usando **lenguajes de Descripción Hardware (HDL)**

# Descripción en lenguajes HDL

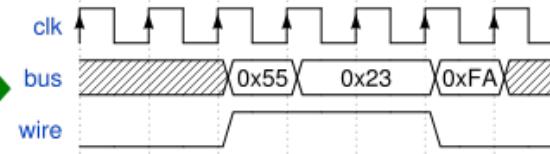
## Fichero HDL

```
module simplez #((
    parameter BAUD = `B115200,
    parameter WAIT_DELAY = `T_200ms,
    parameter ROMFILE = "prog.list",
    parameter DEBUG_LEDS = 0
)
(
    input wire clk,
    input wire rstn_ini,
    output wire [3:0] leds,
    output wire stop,
    output wire tx,
    input wire rx
);

reg [DW-1: 0] alu_out;
reg flag_2;

always @(*) begin
    if (alu_op2)
        alu_out = alu_in;
    else if (alu_clr)
        alu_out = 0;
    //-- Suma de operador 1 + operador 2
    else if (alu_add)
        alu_out = reg_a + alu_in;
    else if (alu_dec)
```

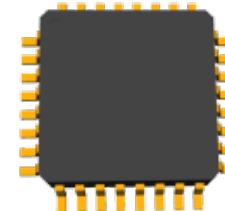
## Simulación



## Bitstream (FPGA)

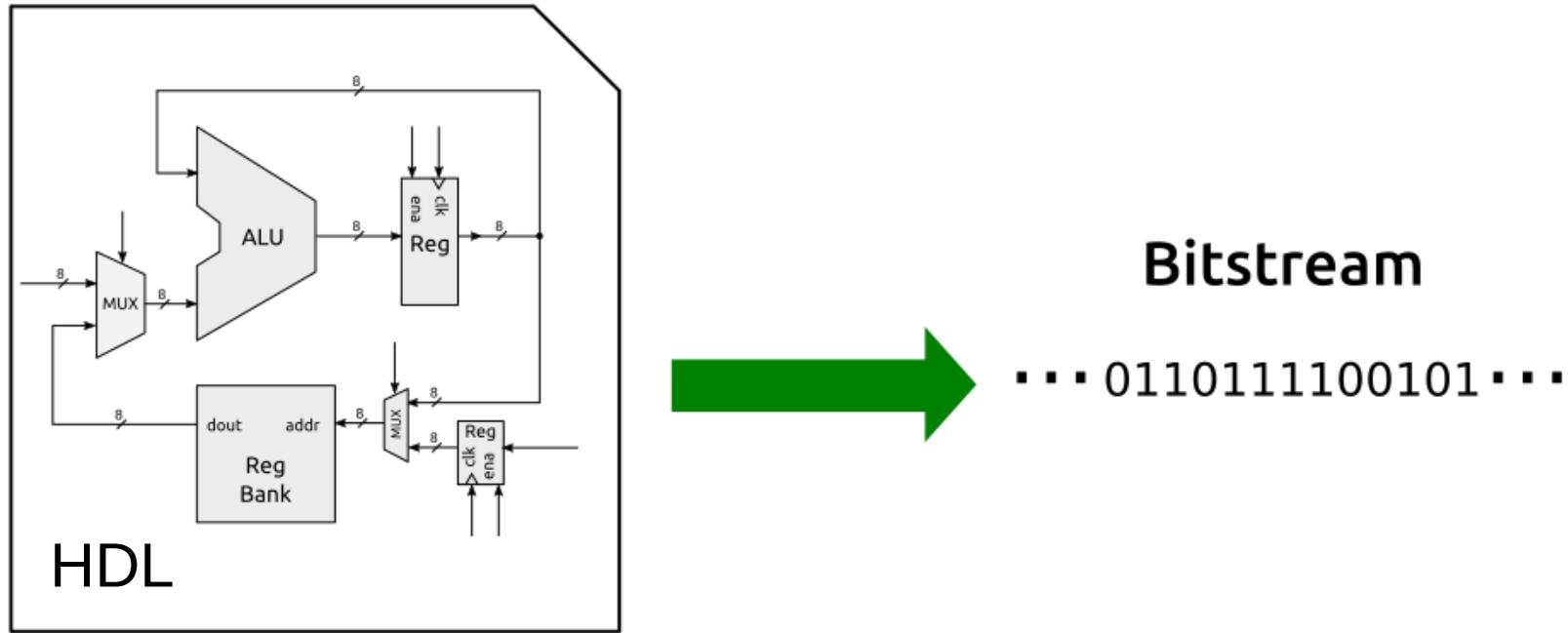
... 0110111100101 ...

## Fabricación chips



Desde una descripción en HDL podemos **simular el circuito**, generar el **bitstream** para FPGAs o **fabricar el circuito integrado**

# El hardware es software



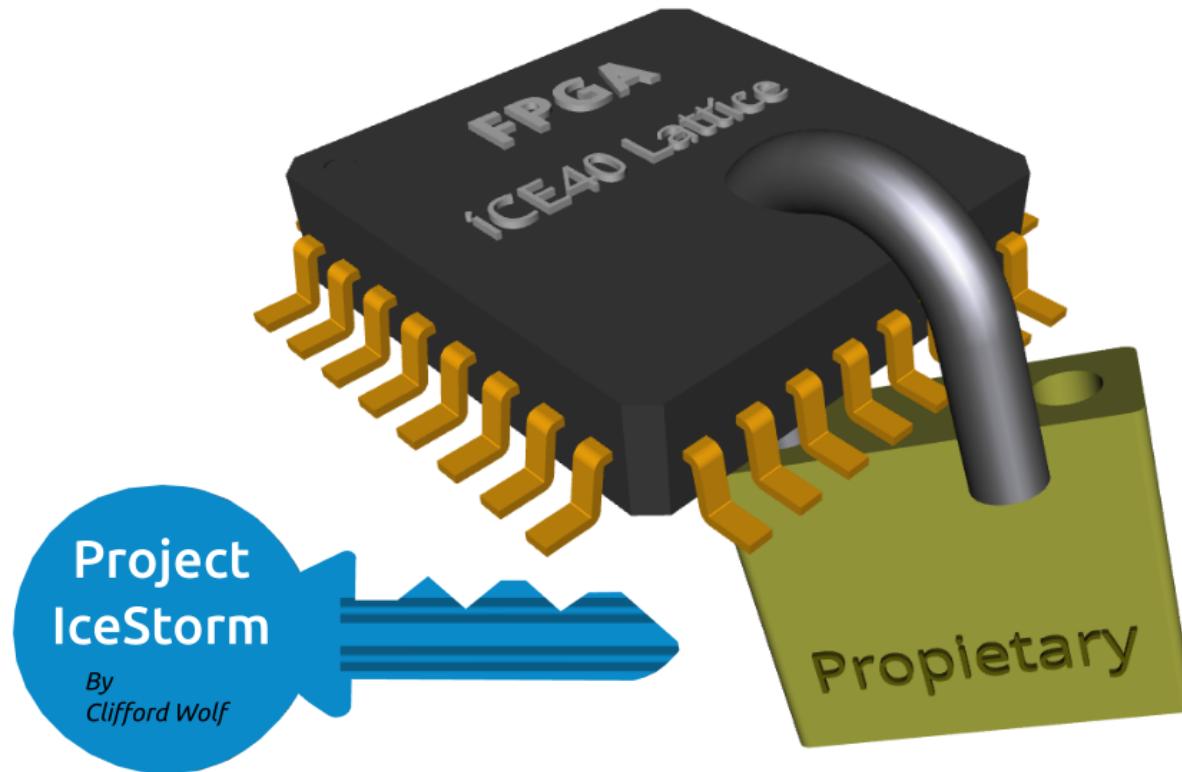
- El hardware libre es igual al software libre
- Muy fácil de compartir
- Telecopias del hardware
- Desarrollo de hardware en comunidad

# FPGAs: Sólo personal autorizado



- Sólo el fabricante conoce los detalles internos
- Sólo se puede usar lo que el fabricante haya previsto
- Atados de por vida al fabricante

# FPGAs libres: El renacimiento



- Proyecto Icestorm (Mayo, 2015)
- La primera *toolchain* que permiten pasar de Verilog al bitstream usando sólo Herramientas libres

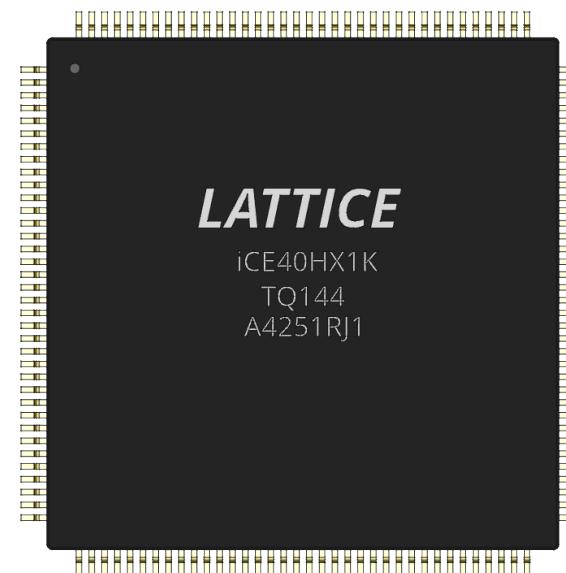
# FPGAs libres

- Definición:

Denominamos **FPGAs libres** a aquellas FPGAs que disponen de una **toolchain totalmente libre**

- **FPGAs libres actualmente:**

- Familia **Lattice iCE40**
- Sólo Lenguaje Verilog



<http://www.latticesemi.com/Products/FPGAandCPLD/iCE40.aspx>

# Proyecto Icestorm

- Herramientas programadas en C/C++
- Bajar del repo y compilar
- Línea de comandos
- Se usan típicamente junto con make
- (bajo nivel)

<http://www.clifford.at/icestorm/>

<https://github.com/cliffordwolf/icestorm>



# APIO

- Autor: **Jesús Arroyo**
- Multiplataforma (Linux, Mac, Windows, Raspberry)
- Línea de comandos
- Programado en python
- Multiplaca: icestick, icezum, icoboard, go-board

Comandos



APIO

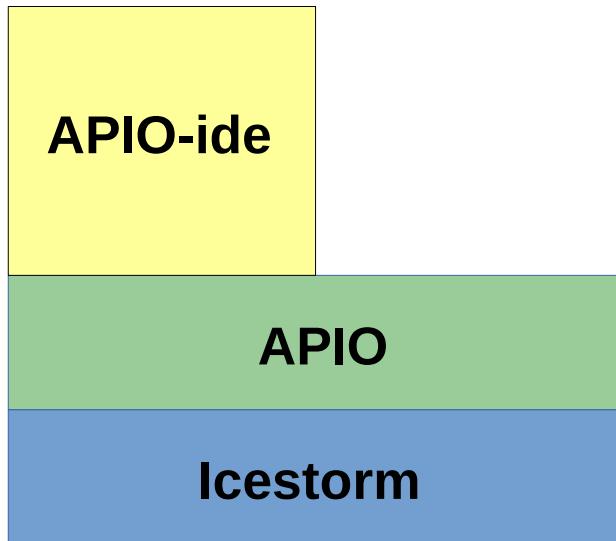
Icestorm

<https://github.com/FPGAwars/apio>

Demo



# APIO-ide



- **Autores:** Jesus Arroyo/Obijuan
- Plug-in para Atom
- No línea de comandos
- Llama a apio
- Aplicable a otros IDEs/editores
- Descripción en Verilog

<https://github.com/FPGAwars/api-ide>

# APIO ide

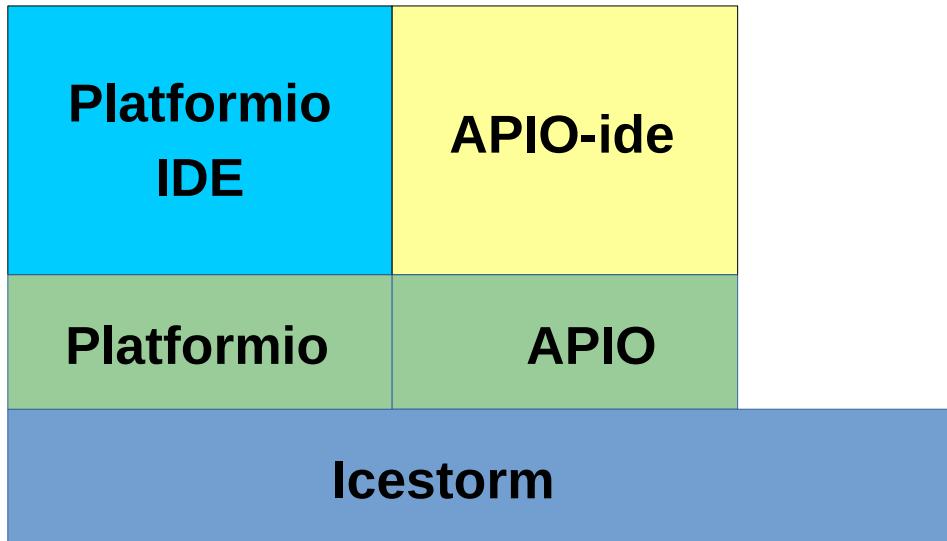
The screenshot shows the APIO ide interface. On the left is a sidebar with various icons for file operations like open, save, and build. The main area displays a file tree for a project named "contador-8-bits-apio". The "main.v" file is selected and shown in the code editor. The code is a Verilog module named "top" that defines a 8-bit counter with a prescaler of 20. It uses a register "counter" and an always block to increment it on each rising edge of the CLK input. The output of the counter is assigned to 8 LEDs (LED0 to LED7). The code editor has syntax highlighting and line numbers. At the bottom, there are tabs for "File 0", "Project 0", and "main.v", along with status indicators for build, issues, and encoding.

```
//---  
// Contador de 8 bits con prescaler  
// Parametro N: Numero de bits para usar en el prescaler  
// Board: icezum  
`default_nettype none  
  
module top(input wire CLK,  
            output wire LED0,  
            output wire LED1,  
            output wire LED2,  
            output wire LED3,  
            output wire LED4,  
            output wire LED5,  
            output wire LED6,  
            output wire LED7);  
  
    parameter N = 20;  
  
    reg [N-1 + 8:0] counter = 0;  
  
    always @ (posedge CLK) begin  
        counter <= counter + 1;  
    end  
  
    // Sacar los 8 bits mas significativos como salida del contador por los leds  
    assign {LED7, LED6, LED5, LED4, LED3, LED2, LED1, LED0} = counter[7+N:N];  
  
endmodule
```

Demo



# Platformio

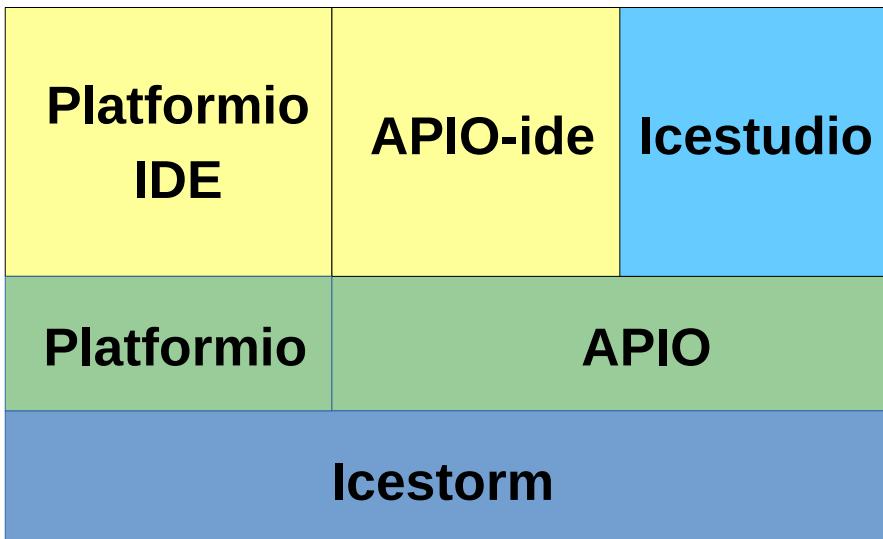


- **Autores:** Jesus Arroyo / Obijuan
- FPGAs libres en Platformio
- Linea de comandos
- IDE basado en Atom
- Aplicable a otros IDEs/editores

[http://platformio.org/platforms/lattice\\_ice40](http://platformio.org/platforms/lattice_ice40)



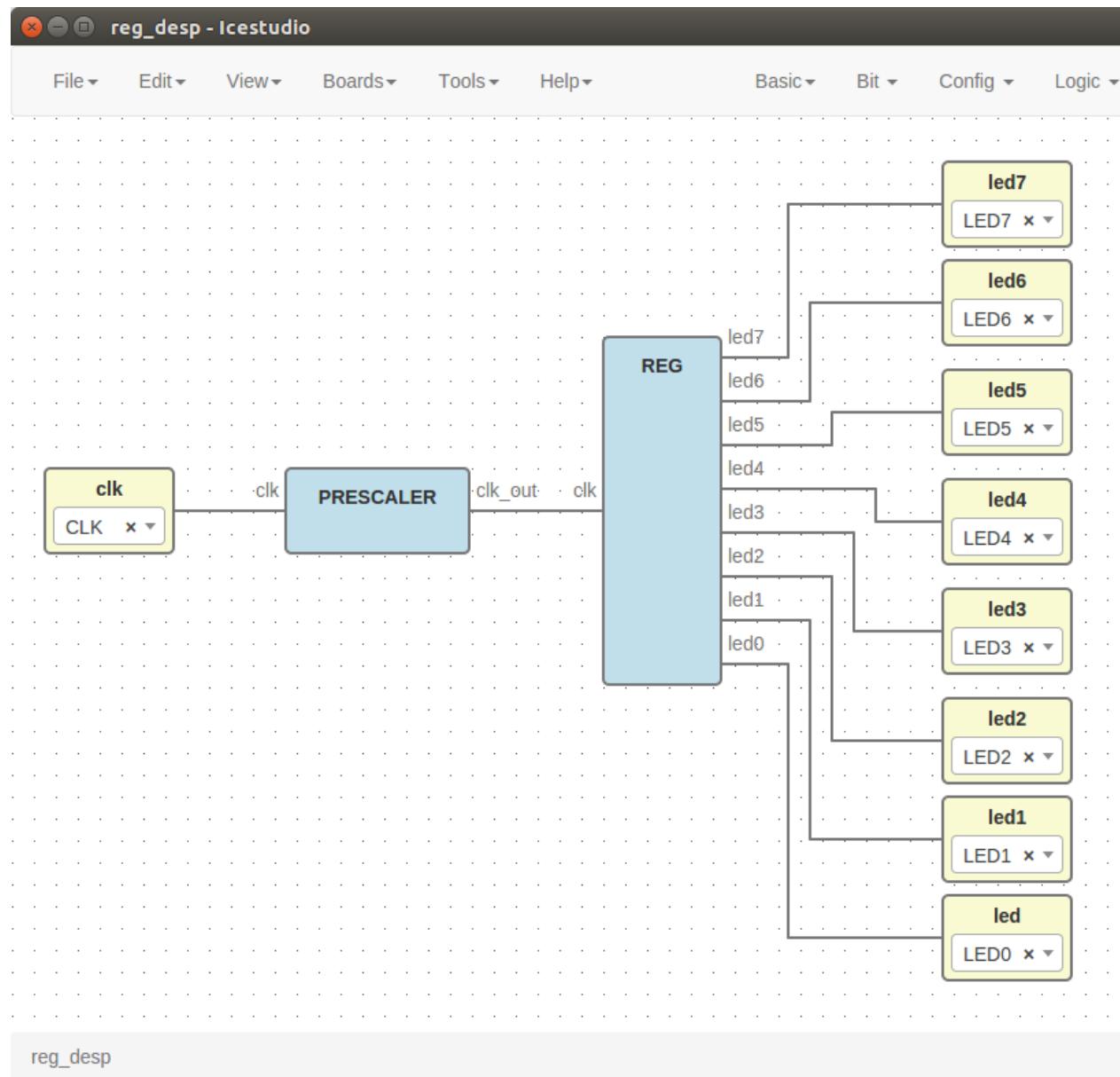
# icestudio



- Autor: **Jesús Arroyo**
- Electrónica digital para todos
- Sin conocimientos de verilog
- Herramienta visual
- Traduce a verilog

<https://github.com/FPGAwars/icestudio>

# Icestudio: Demo

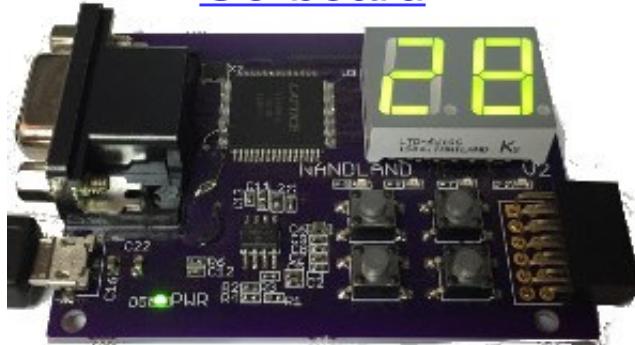


# Tarjetas entrenadoras con FPGAs libres

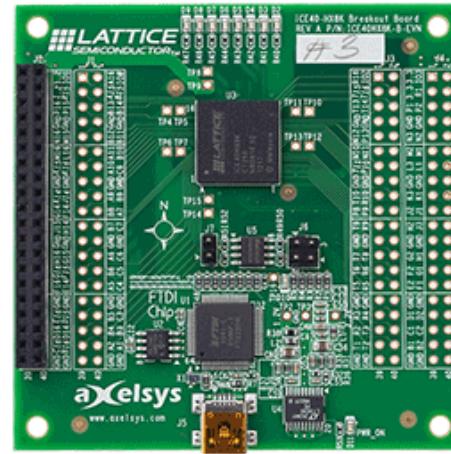
[Icestick](#)



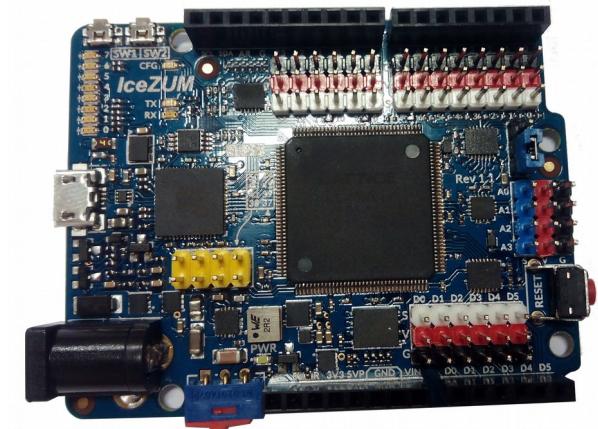
[Go-board](#)



[iCE40-HX8K Breakout Board](#)



[Icezum Alhambra](#)



- Conexión directa al PC (USB)
- Soportadas por Apio/Icestudio

# Tarjetas entrenadoras FPGA

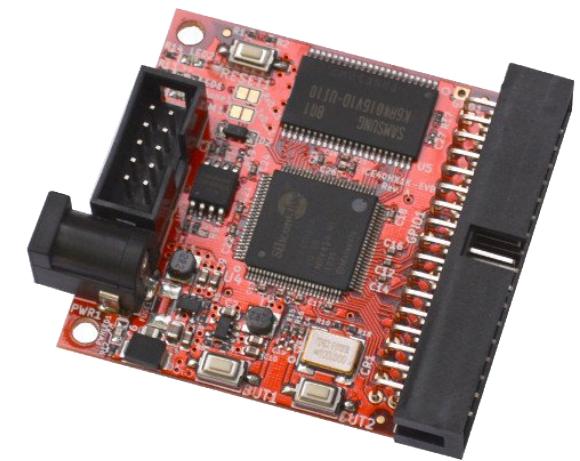
icoboard



Mystorm



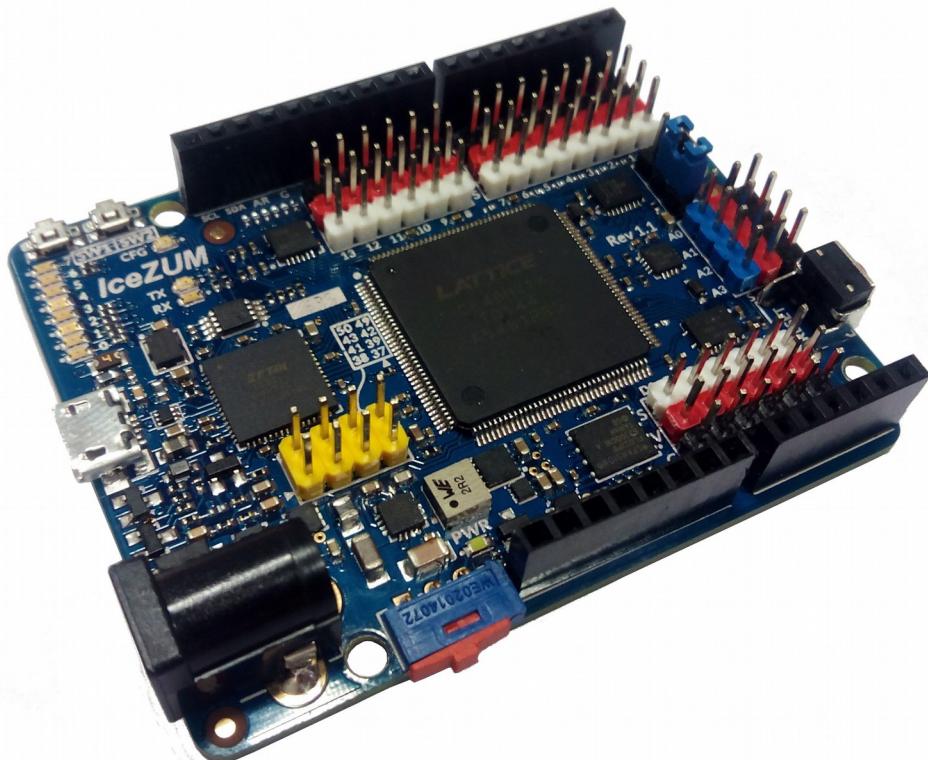
iCE40HX1K-EVB



- Conexión a Raspberry PI
- Soportada por Apio/Icestudio

NO Soportadas por Apio/Icestudio

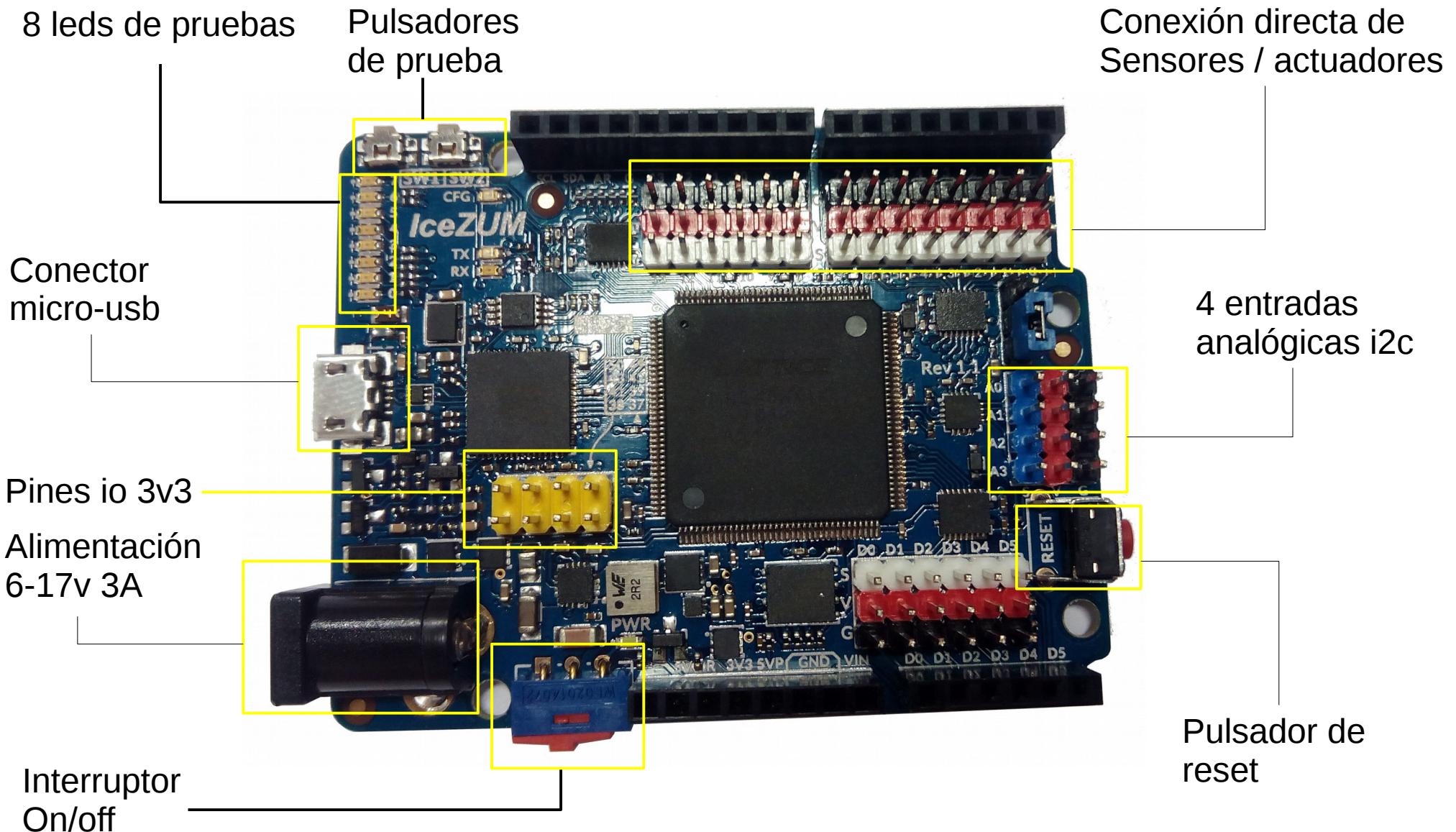
# Icezum Alhambra v1.1



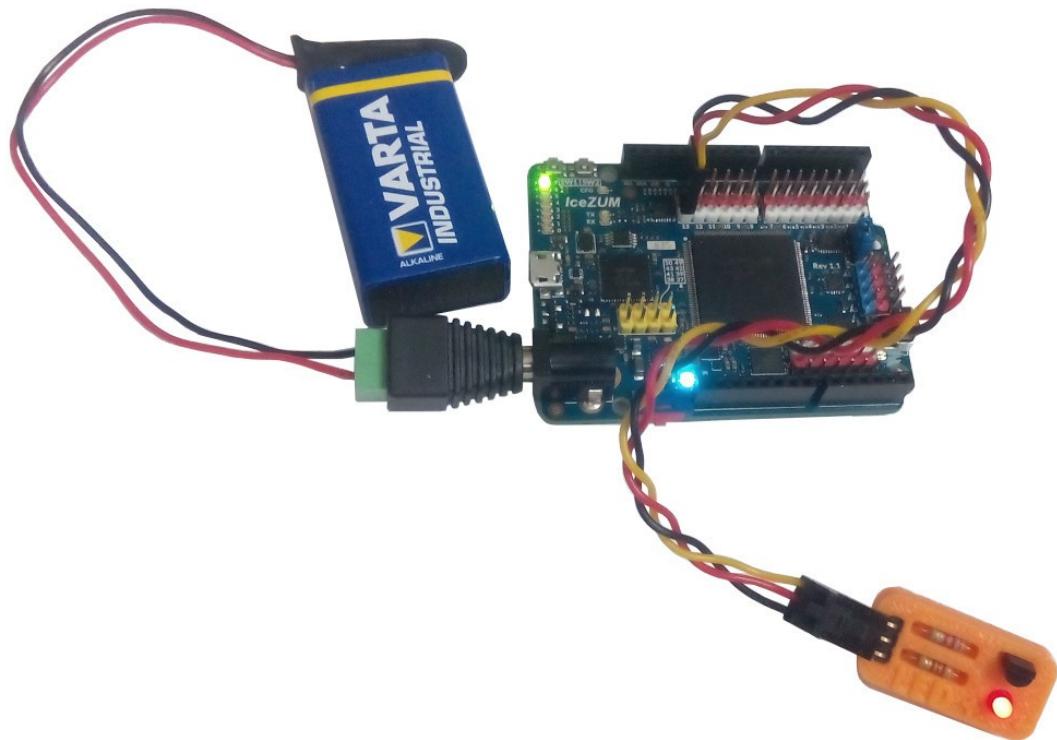
- Autor: **Eladio Delgado**
- Diseñada en Pinos del Valle (Granada)
- Arduino de las FPGAs
- Compatible Arduino
- Fácil conexión de circuitos externos/sensores/servos
- Reutilización de los shields de arduino
- 20 entradas/salidas de 5v
- 3A corriente de entrada
- Perfecta para hacer robots

<https://github.com/FPGAwars/icezum/wiki>

# Icezum Alhambra v1.1



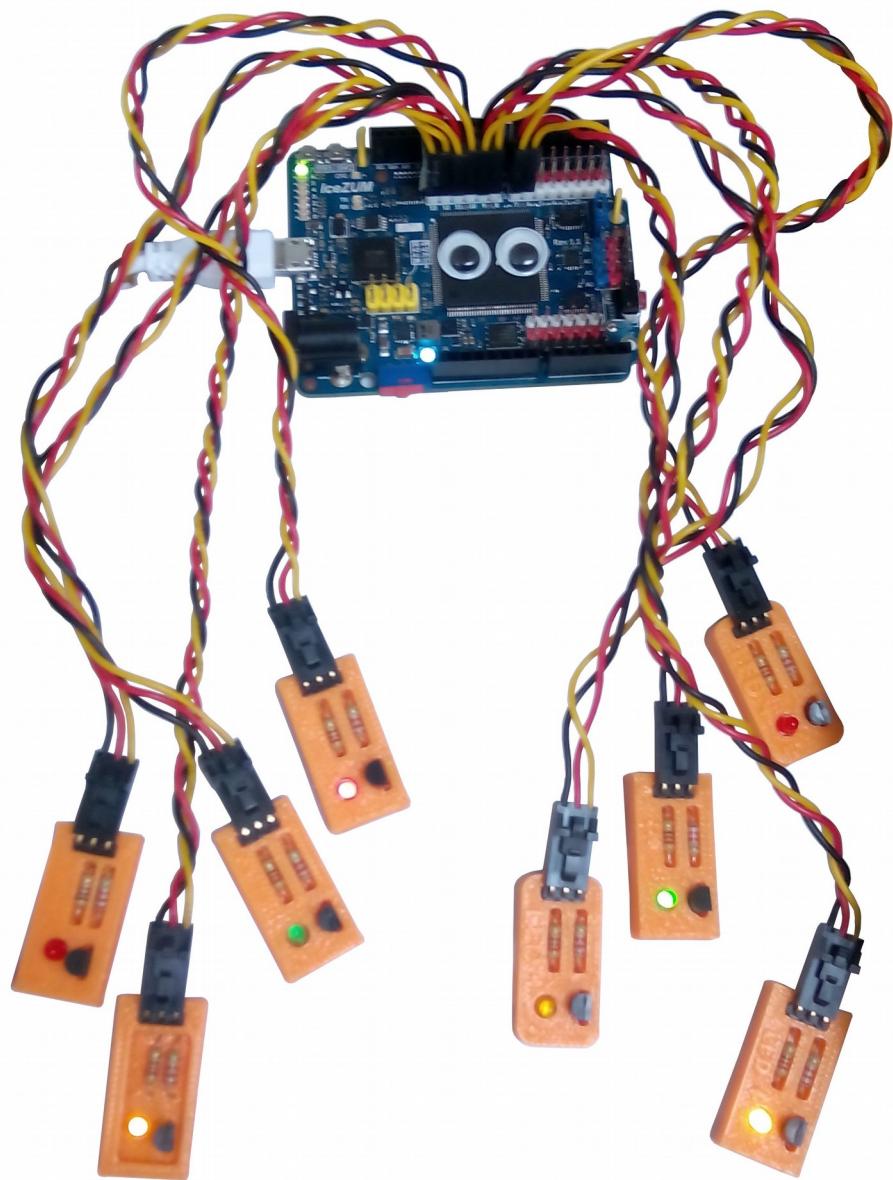
# Alhambra-led



- Autor: **Obijuan**
- Periférico “Hola mundo”
- 1 led + 1 transistor
- ¡PCB Imprimible!

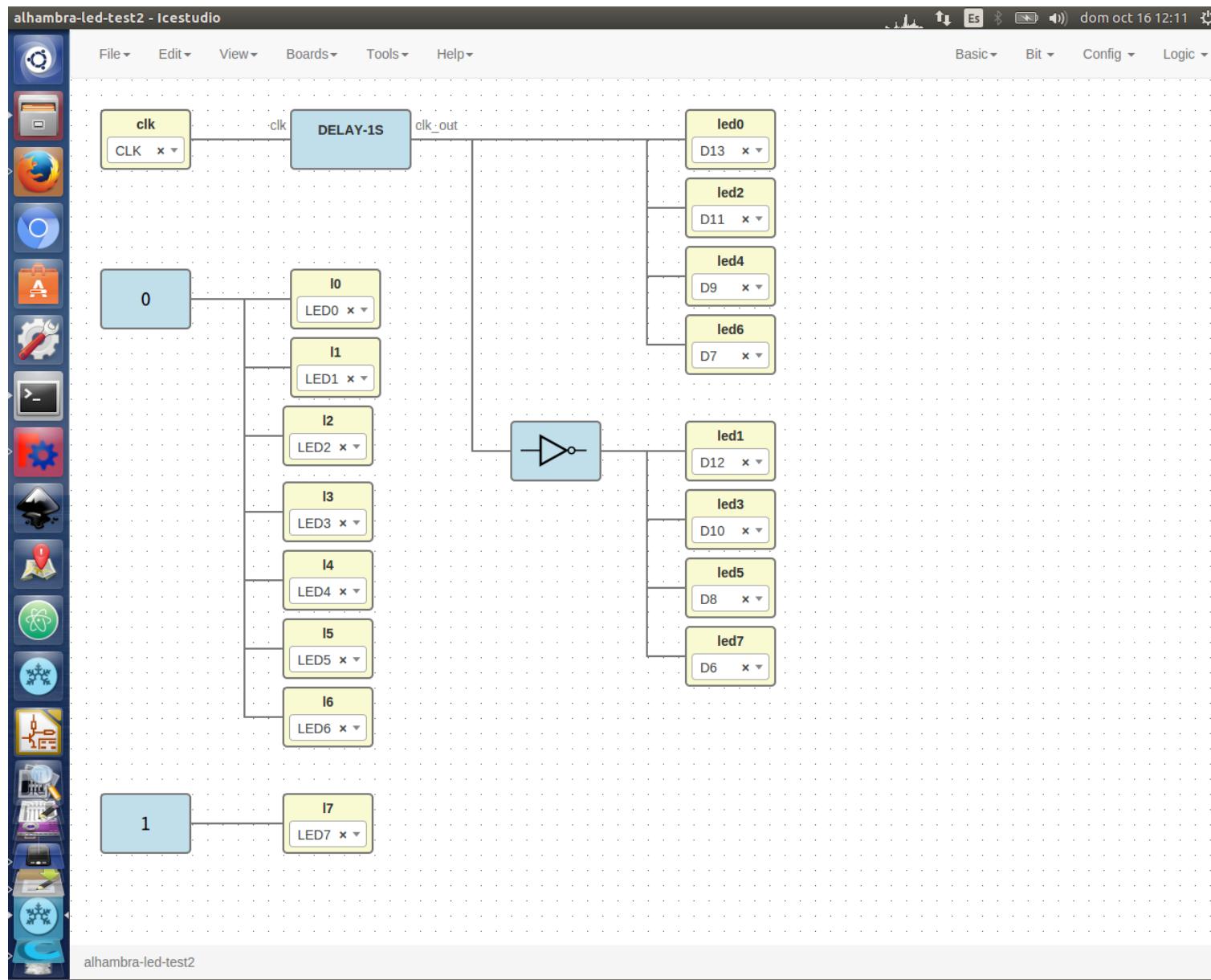
<https://github.com/FPGAwars/alhambra-led/wiki>

# Rasty



- FPGA Fun!
- 8 Alhambra-leds
- Alimentación: power bank

# Circuito de prueba de Rasty

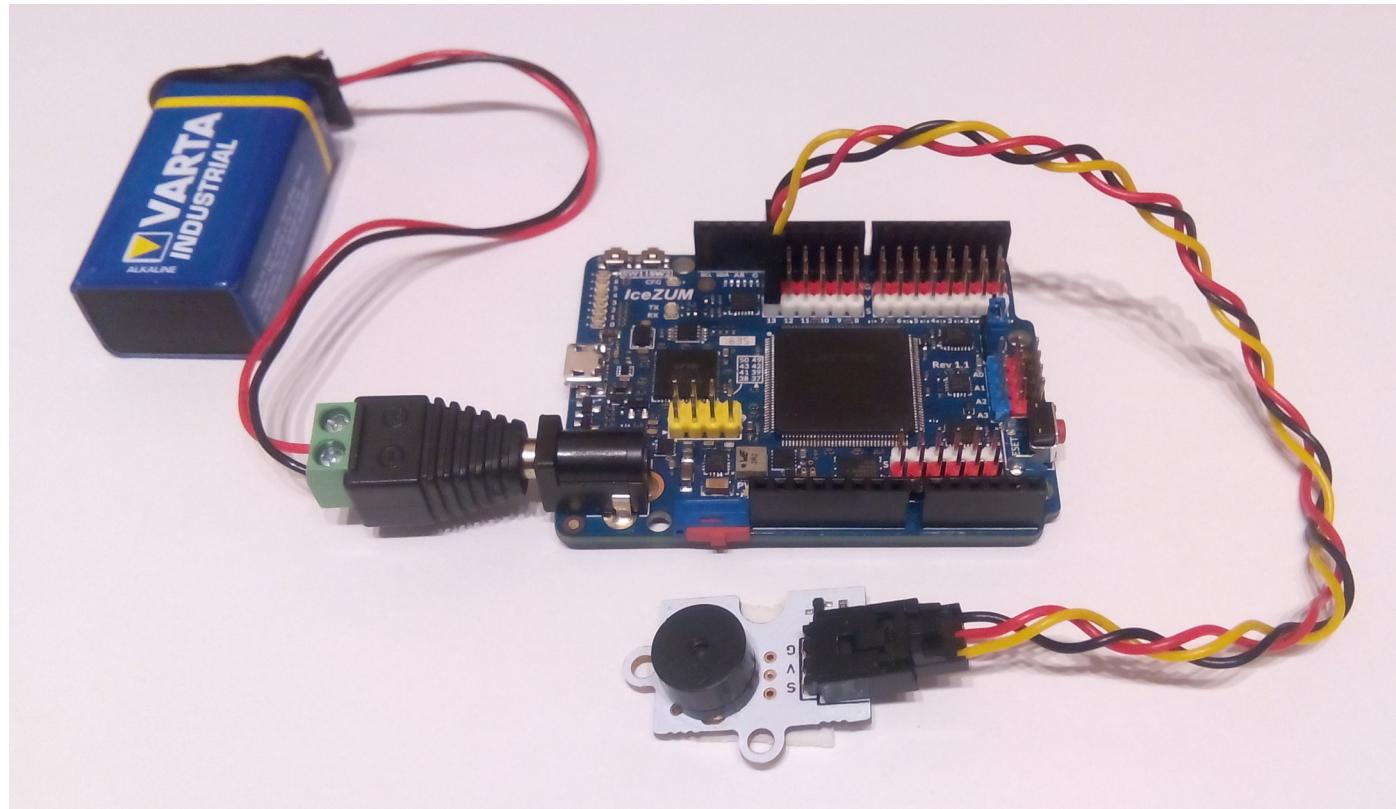


# Larby: Robot modular



- Servos conectados directamente a Icezum Alhambra
- Configuración mínima pitch-pitch
- Módulo impresos en 3D

# Sonidos simples con Zumbador



- Conexión a una placa con Zumbador
- Melodías básica
- ¡Fácil meter varios canales!

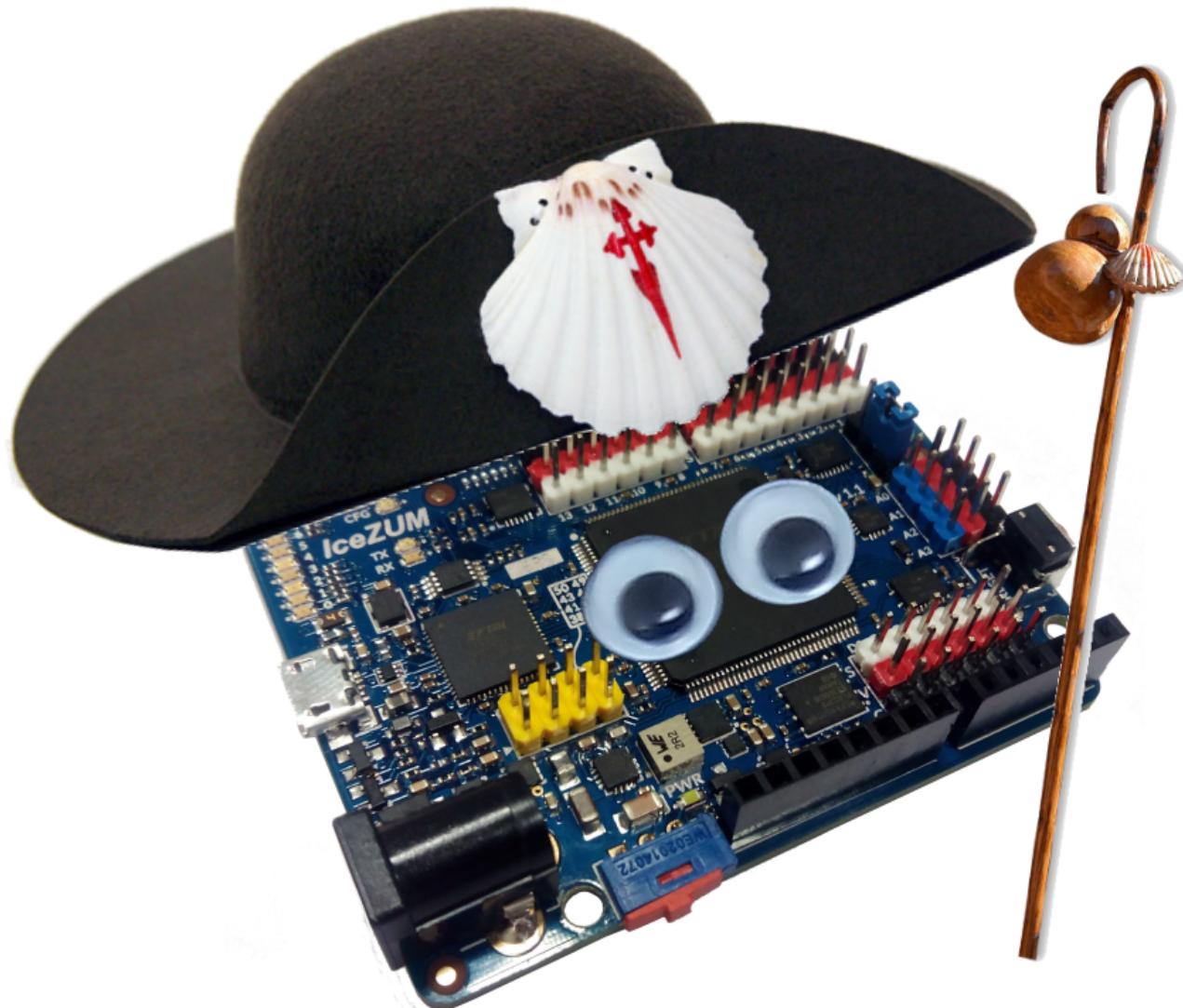
# Comunidad FPGAwars



- Comunidad para **compartir conocimiento** relacionado con **FPGAs libres**
- Es el **clonewars** de las FPGAs, pero en modesto :-)
- Idioma: Castellano
- 185 miembros
- Cualquier pregunta / comentario / sugerencia → Correo a la lista :-)

<http://fpgawars.github.io/>

# Icezum Alhambra peregrina



- A.k.a **CalmaSAV**
- Empieza:  
[Rinconingenieril.es](http://Rinconingenieril.es)
- Ámbito: Toda España
- Coordinado desde la lista  
de FPGAwars

# Me molan las FPGAs libres... ¿Por dónde empiezo?

**Paso 1:** Consigue una placa con FPGA libre

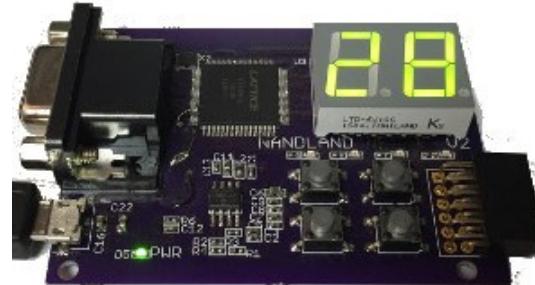
Icestick



iCE40-HX8K Breakout Board

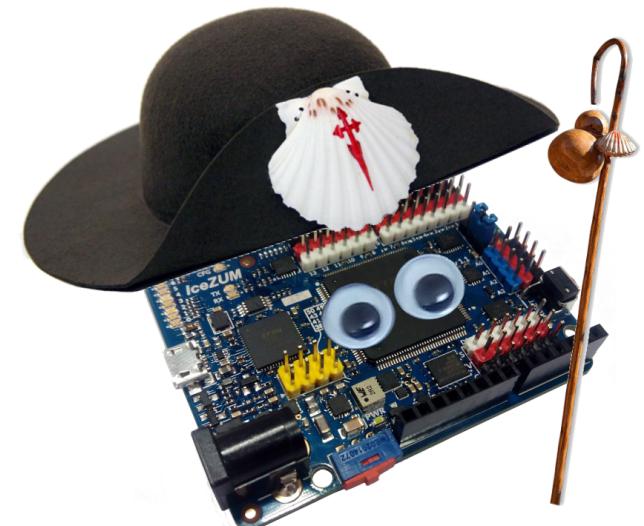
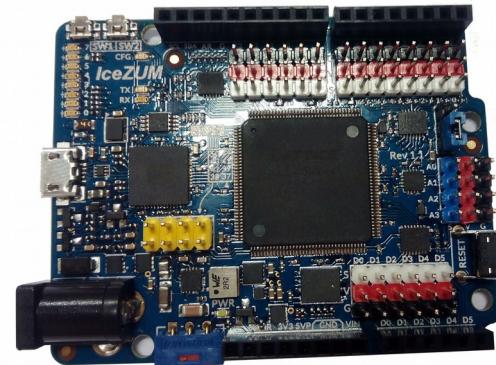


Go-board



Icezum Alhambra  
Peregrina

Icezum Alhambra

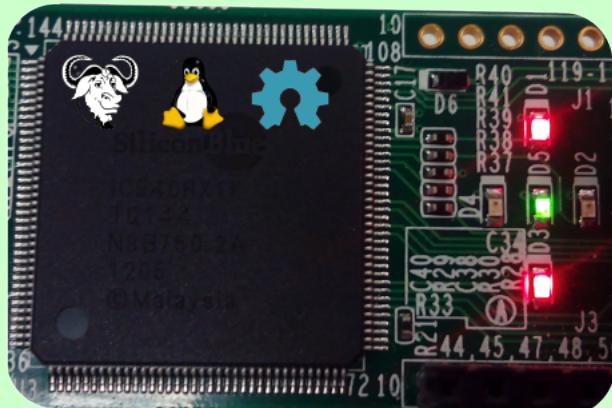


# Me molan las FPGAs libres... ¿Por dónde empiezo?

**Paso 2:** Instálate Apio / Icestudio

**Paso 3:** Empieza a aprender verilog

[Tutorial: Diseño Digital para FPGAs, con herramientas libres](#)



# Me molan las FPGAs libres... ¿Por dónde empiezo?

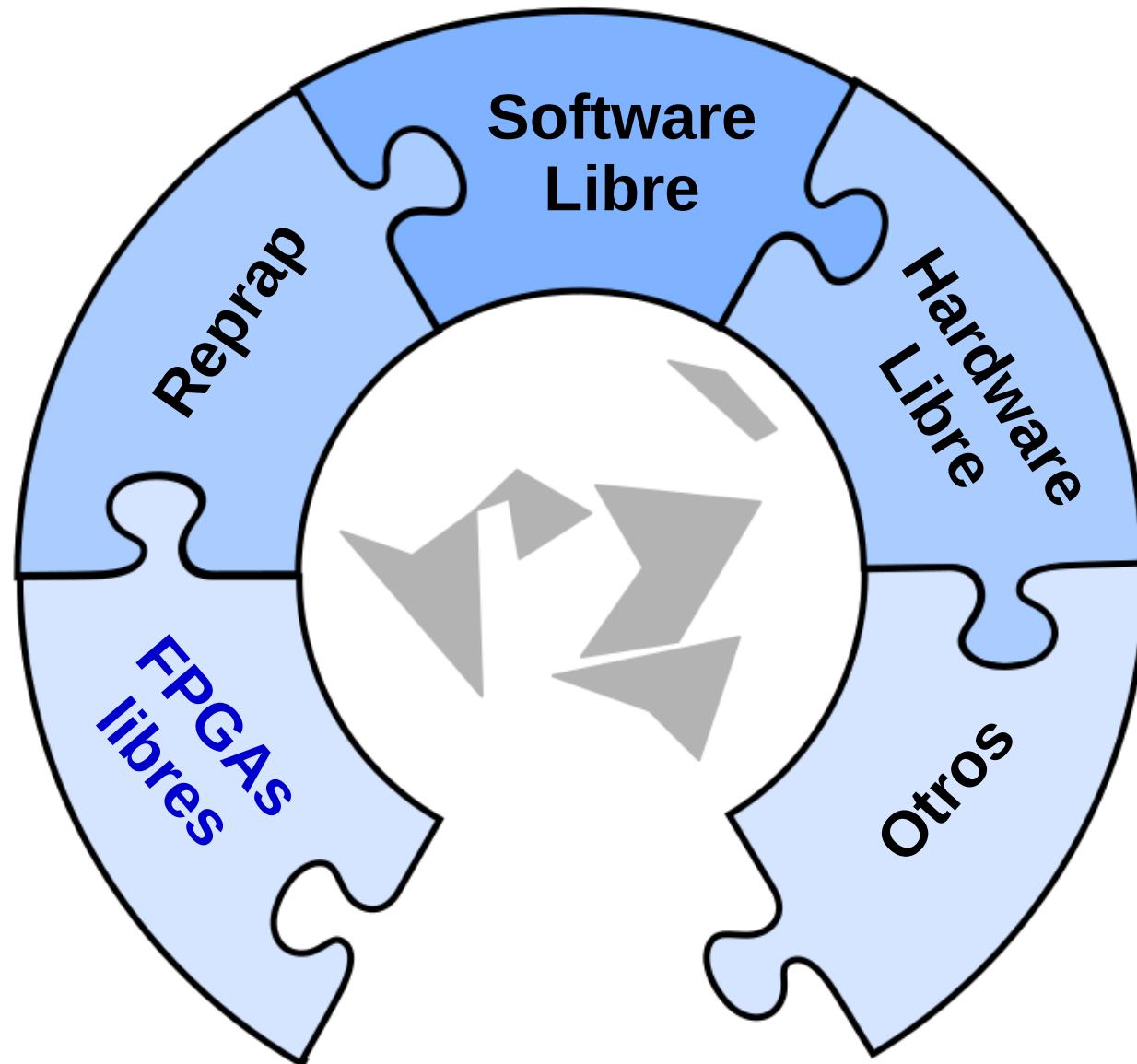
**Paso 4:** Apúntate al grupo de FPGAwars y haz preguntas

- Las FPGAs libres evolucionan muy rápido y en seguida la información se queda obsoleta
- En FPGAwars es donde se cuece lo último de lo último

**Paso 5:** Haz tus propios proyectos con FPGAs libres

**Paso 6:** ¡Comparte tu proyecto con la comunidad! :-)

# FPGAs en Patrimonio Tecnológico de la humanidad



# FPGAs libres



Juan González Gómez (Obijuan)

<https://github.com/Obijuan>