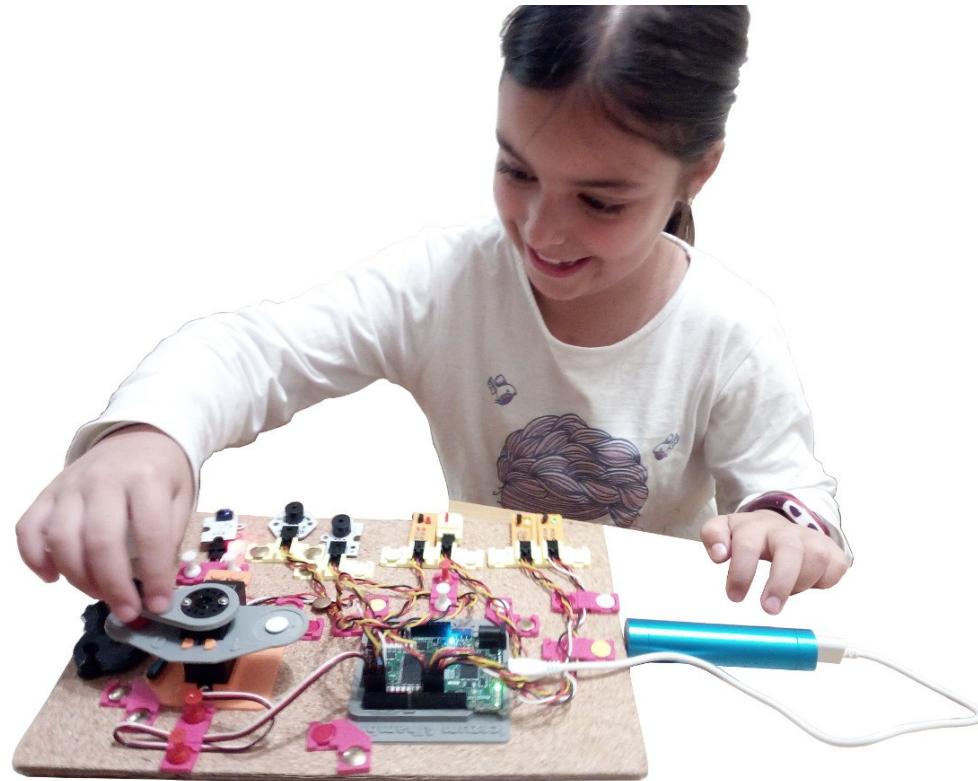


# Electrónica digital para todos con FPGAs Libres

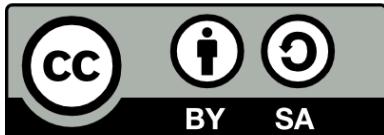


AYUNTAMIENTO DE  
CAMARGO

IBEROBOTICS  
ROBOTS PERSONALES Y DE SERVICIOS



Juan González Gómez (Obijuan)  
<https://github.com/Obijuan>



Curso de Verano UC: Robótica, Arduino y Hardware Libre

Centro Municipal de Empresas. Pol. Industrial de Trascueto, s/n. Revilla de Camargo

[https://github.com/Obijuan/myslides/wiki/2017\\_07\\_06:-UC:-Electrónica-digital-para-todos-con-FPGAs-libres](https://github.com/Obijuan/myslides/wiki/2017_07_06:-UC:-Electrónica-digital-para-todos-con-FPGAs-libres)

***La era digital: Todo es número***

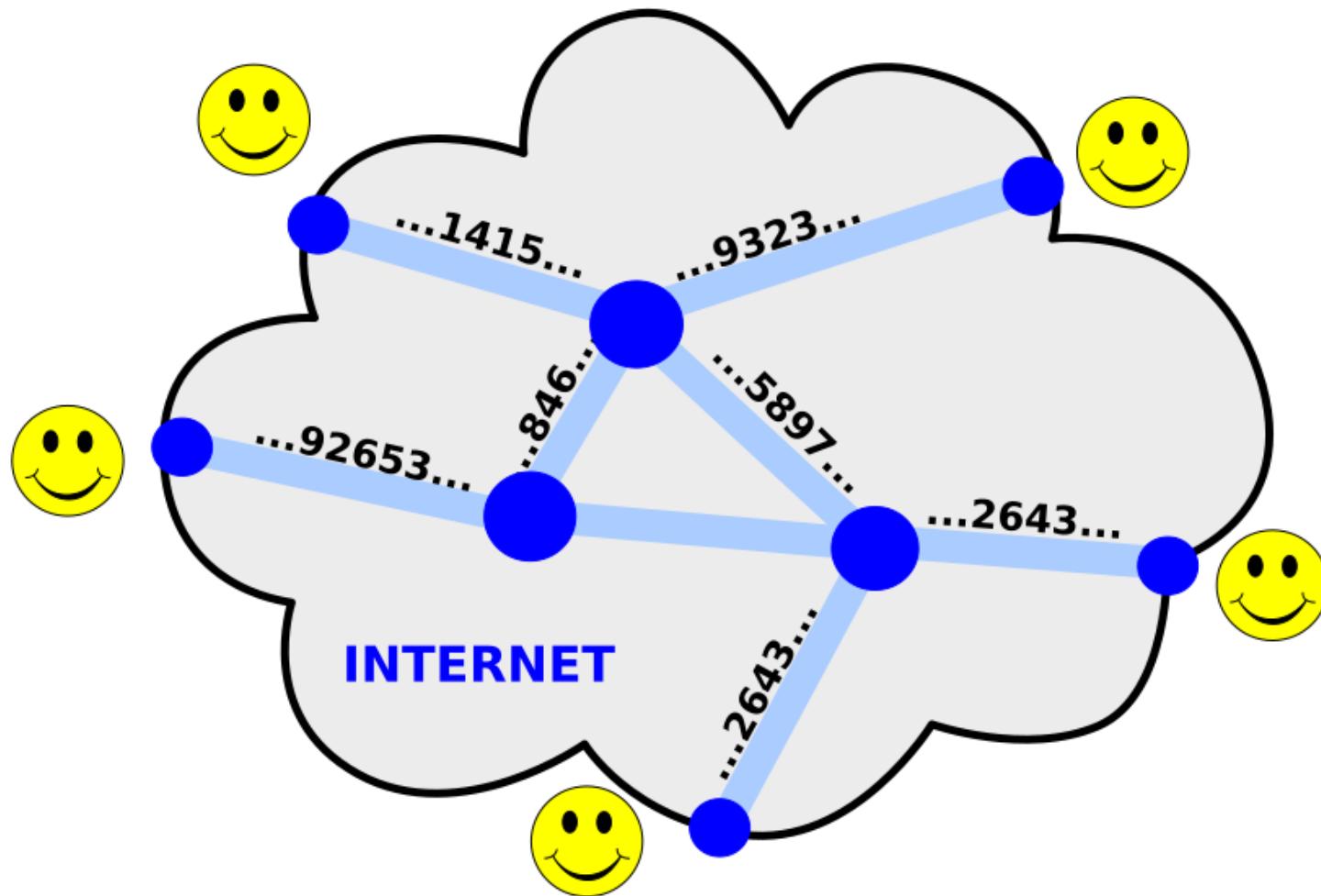
# Mundo digital



1415926535 8979323846 2643383279  
5028841971 6939937510 5820974944  
5923078164 0628620899 8628034825  
3421170679 8214808651 3282306647  
7245870066 0631558817 4881520920  
9628292540 9171536436 7892590360  
0113305305 4882046652 1384146951  
9415116094 3305727036 5759591953  
0921861173 8193261179 3105118548  
0744623799 6274956735 1885752724  
8912279381 8301194912 9833673362  
4406566430 8602139494 6395224737  
1907021798 6094370277 0539217176  
2931767523 8467481846 7669405132

**TODO SON NÚMEROS**

# Internet

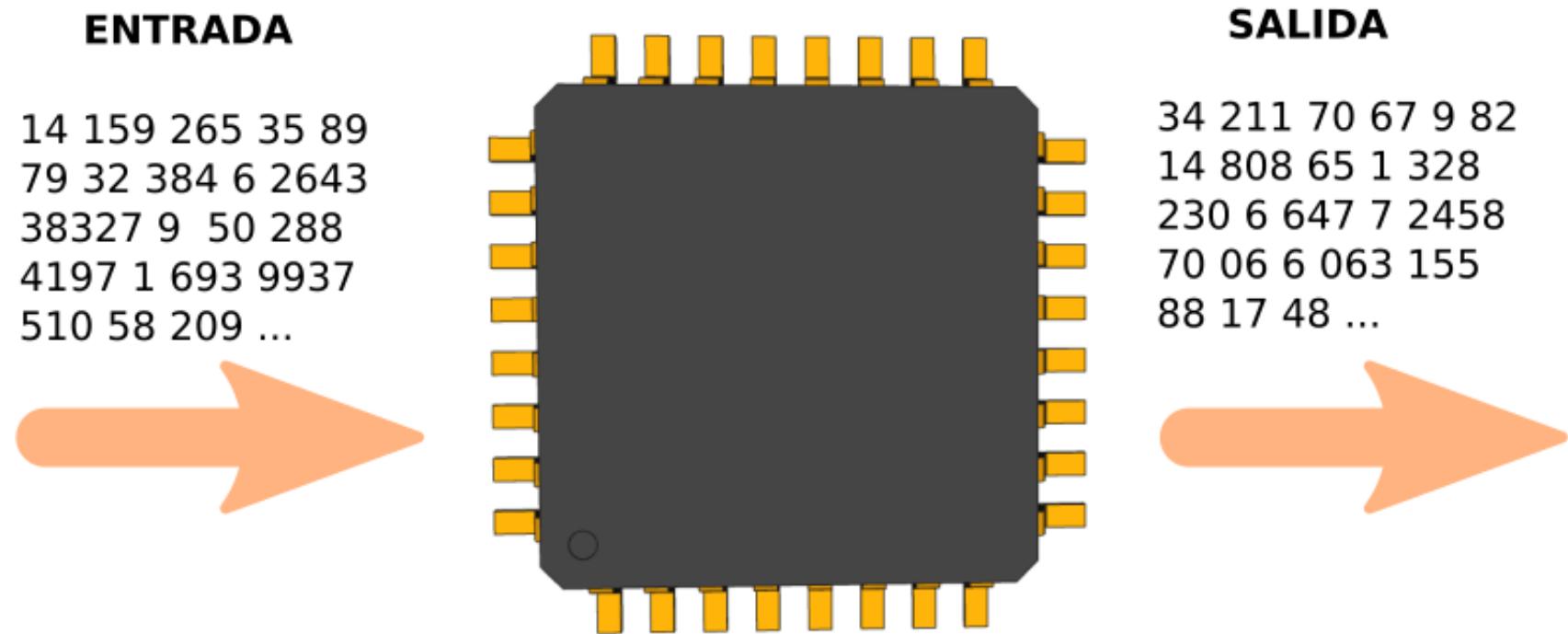


*Transporte, almacenamiento y manipulación  
de NÚMEROS*

# Los pitagóricos estarían orgullosos



# Electrónica digital (I)



*Circuitos electrónicos que manipulan,  
transportan y almacenan NÚMEROS*

# Bits

14159265358979323846264  
33832795028841971693993  
75105820974944592307816



Números en decimal

0001 0100 0001 0101 1001 0010 0110 0101 0011  
0101 1000 1001 0111 1001 0101 0000 0010 1000  
1000 0100 0001 0001 1001 0111 0001 0110 0111

Digitos Binarios

***TODO se reduce a 1s y 0s***

0 1

Ser o no ser

Par o impar

Verdadero o falso

Todo o nada

Yin y Yang

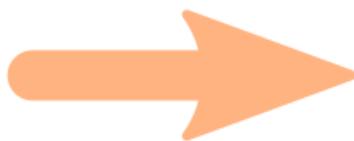
Culo o codo...

*En el interior de los microchips*

# Productos electrónicos y circuitos

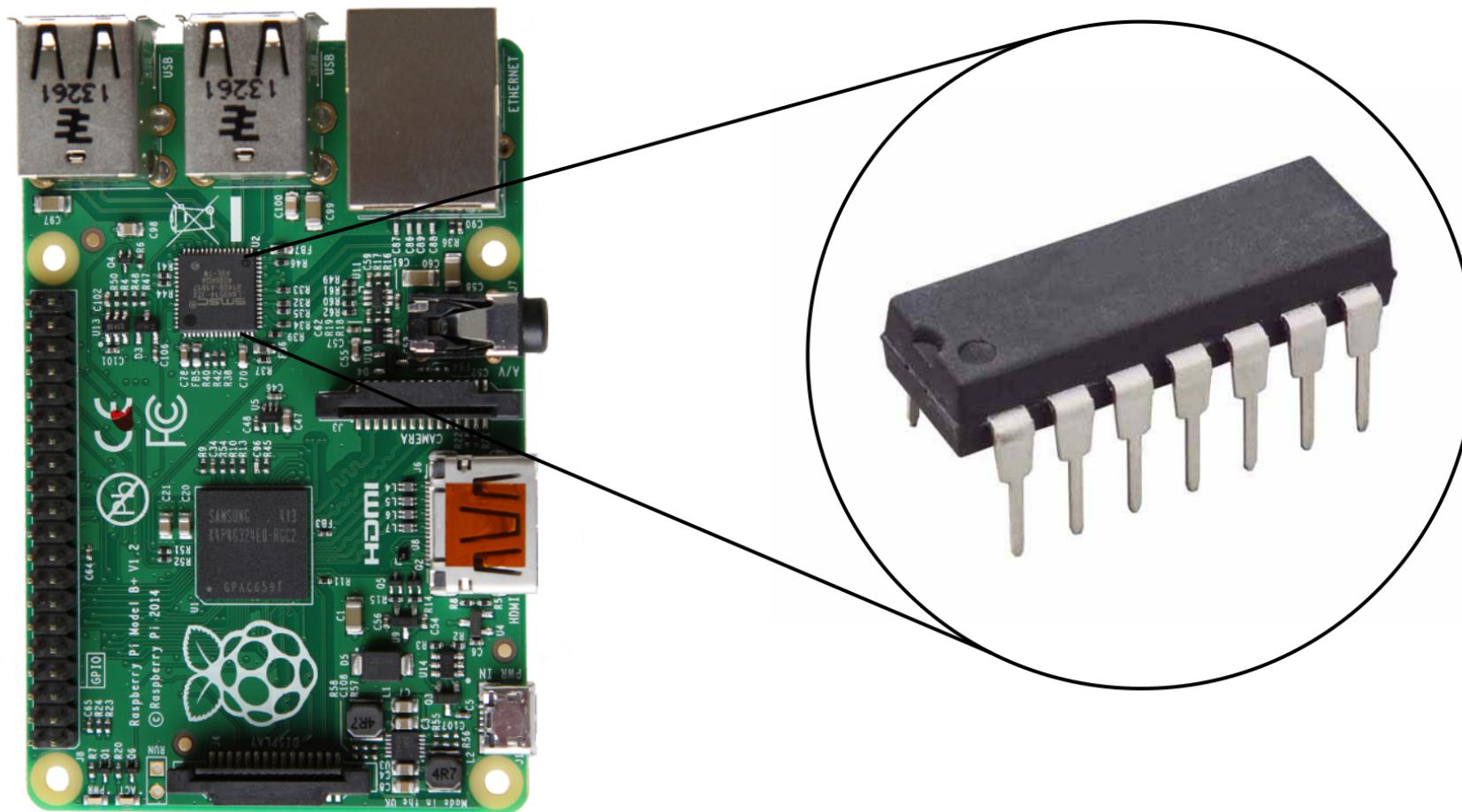


Producto Electrónico



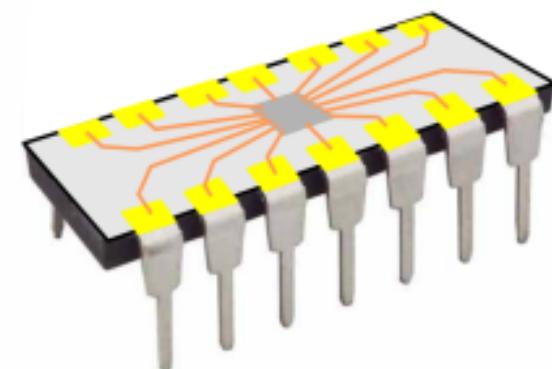
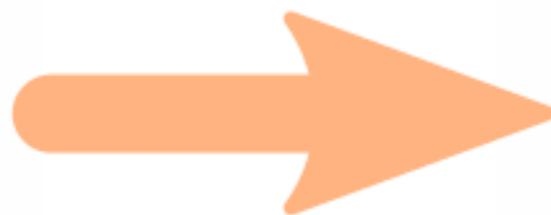
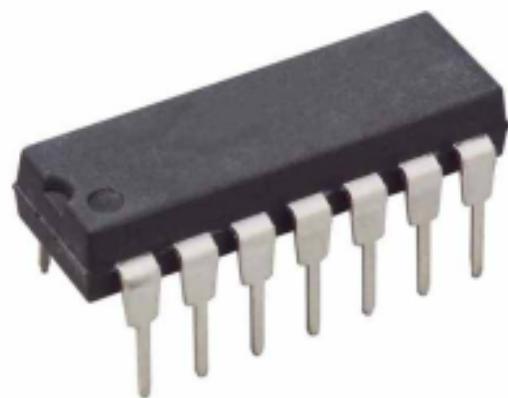
Circuito electrónico

# PCBs y Circuitos integrados



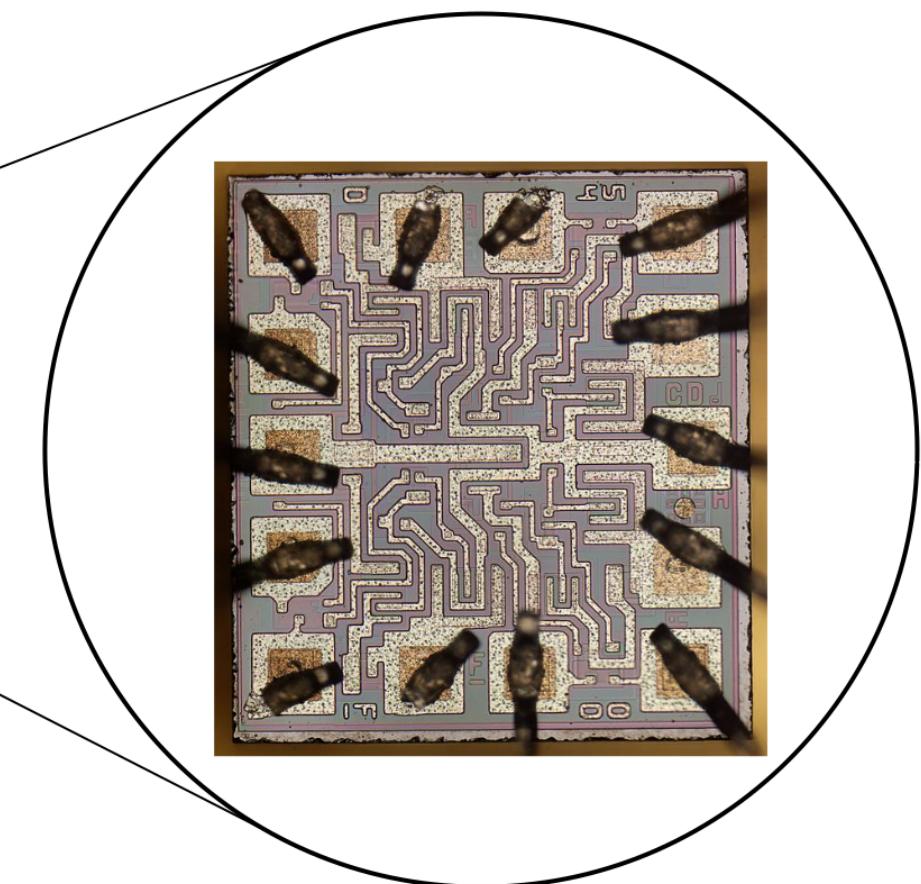
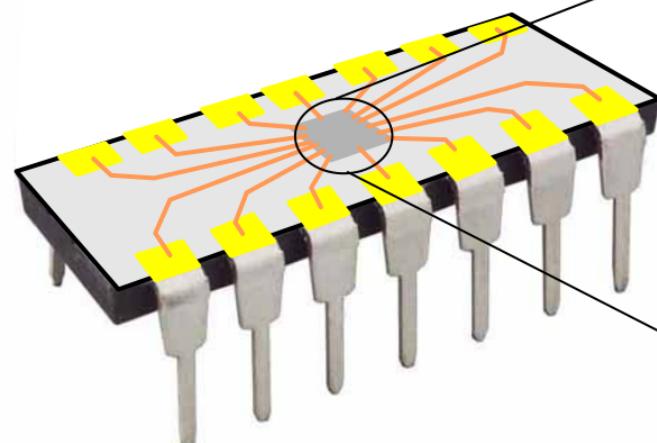
# Encapsulado y dado

*Demo: Mostrar chips  
Dip y eeprom*



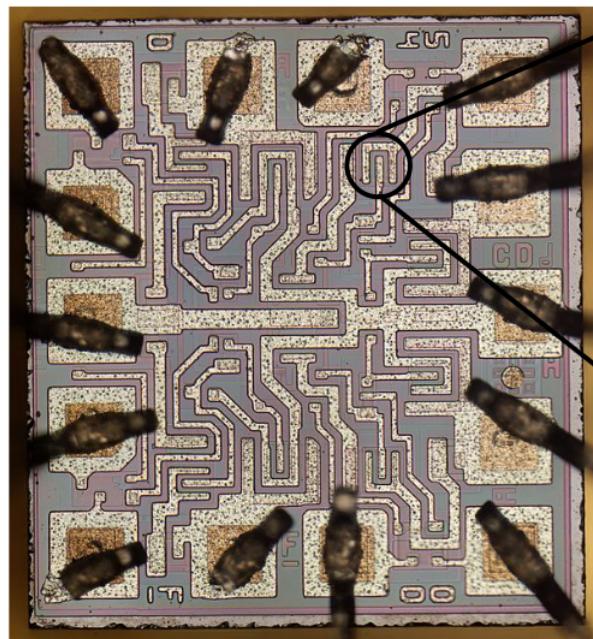
# Dado de silicio

*Demo:  
Proyecto 54/74*

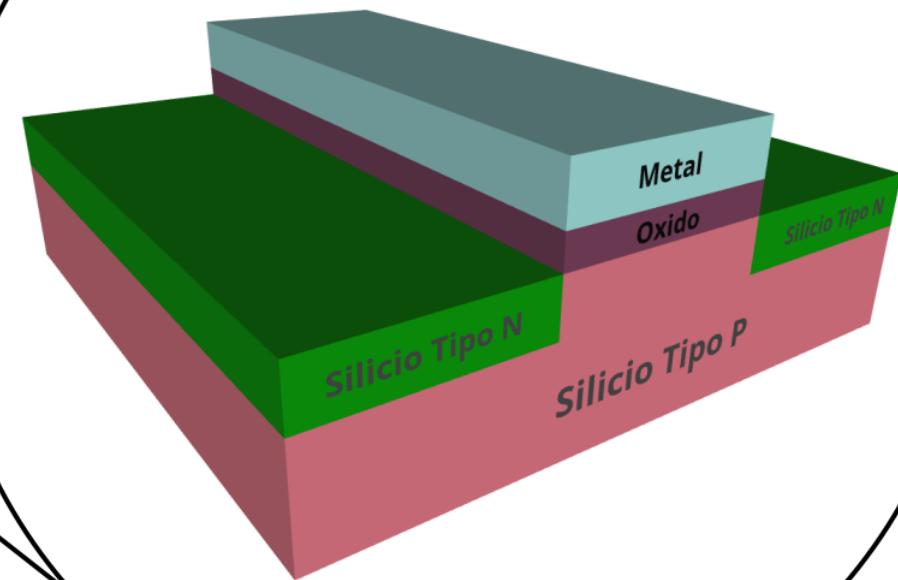


# Semiconductores

Dado

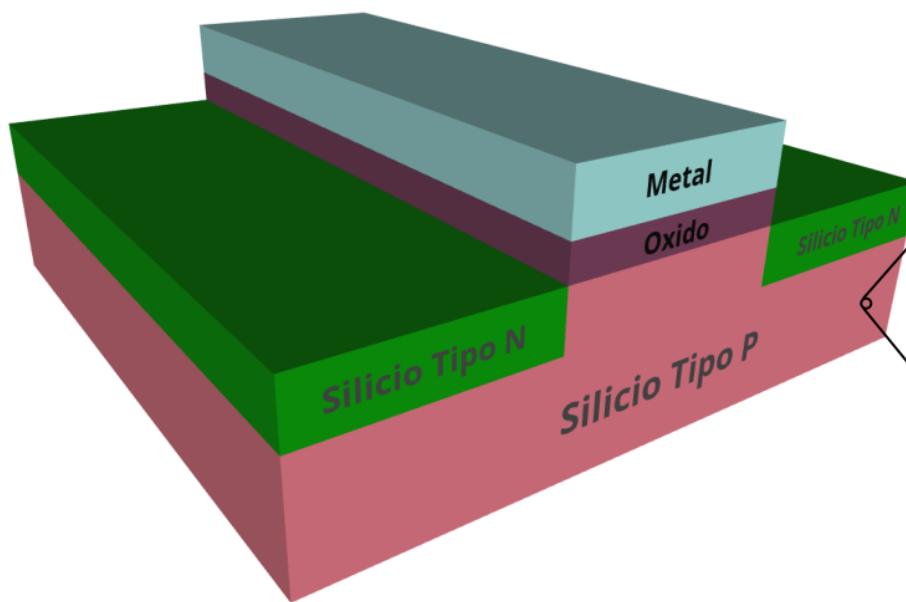


Transistor CMOS

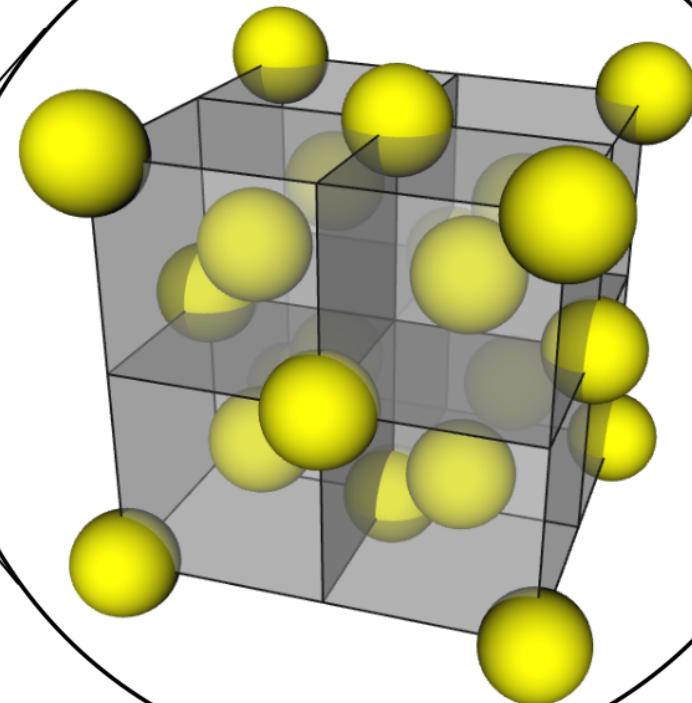


# Cristal de silicio

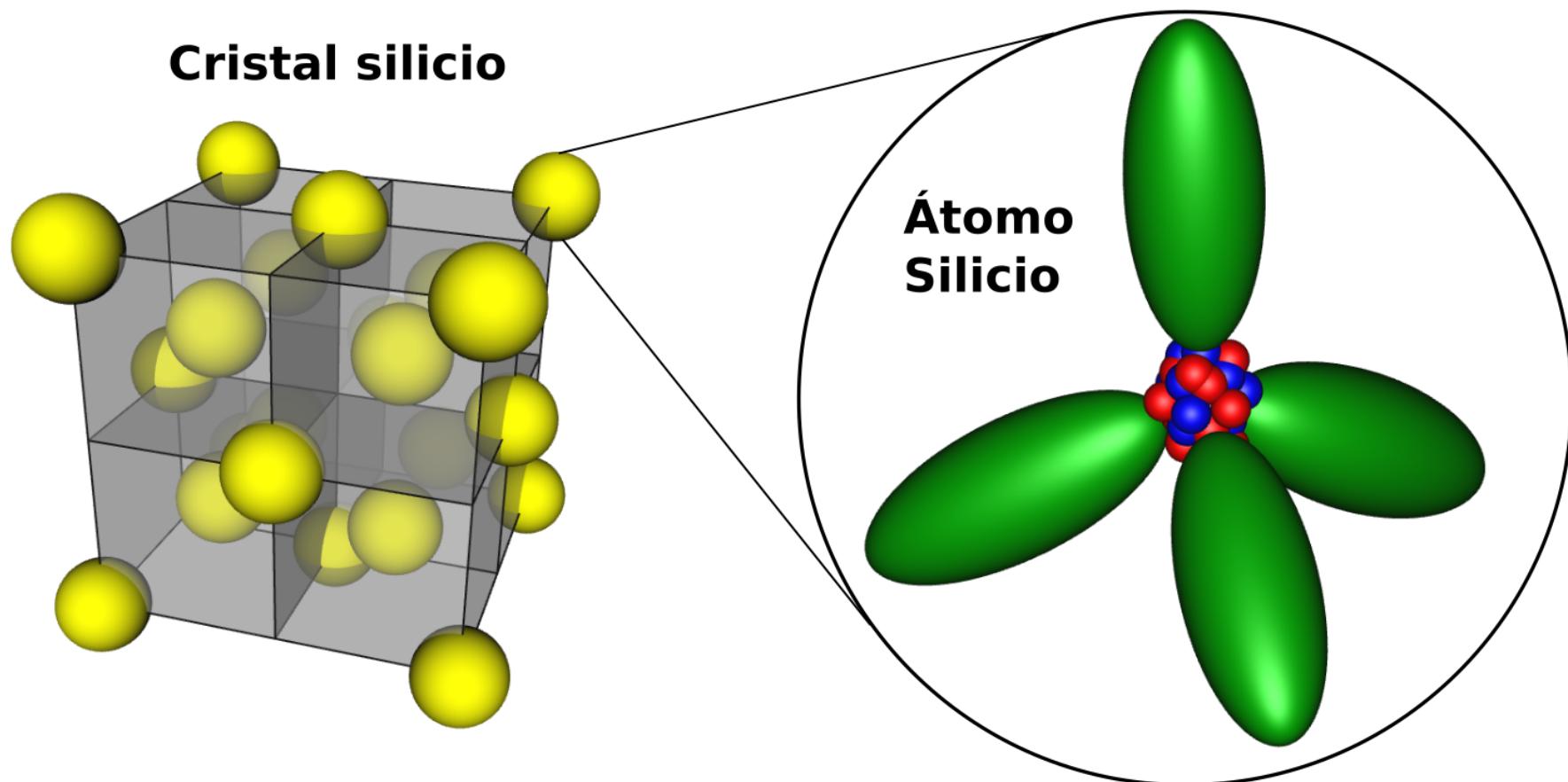
**Transistor CMOS**



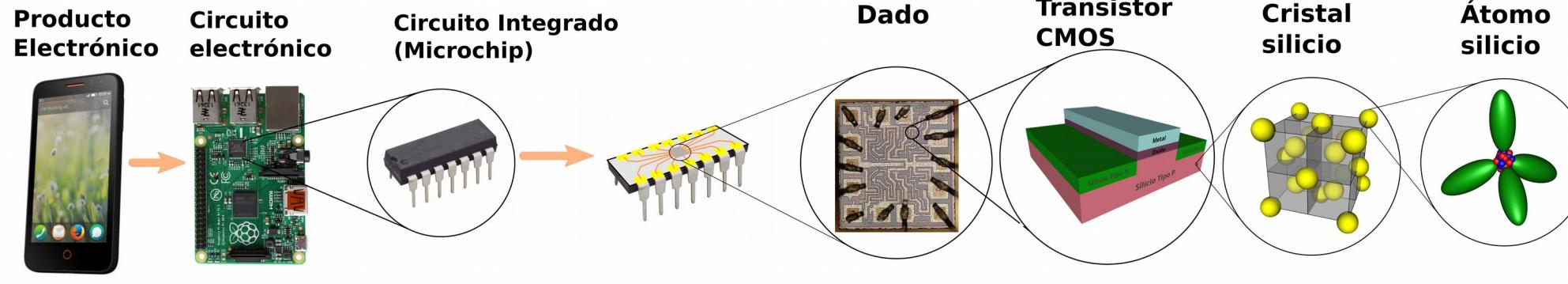
**Cristal silicio**



# Átomos :-)

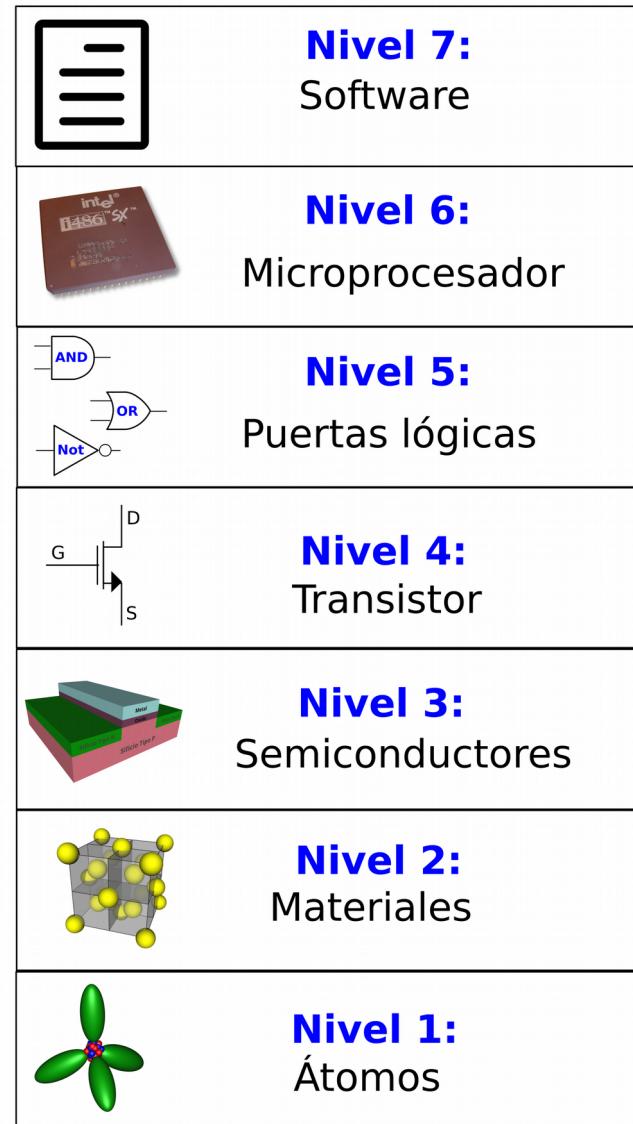


# Del producto al átomo

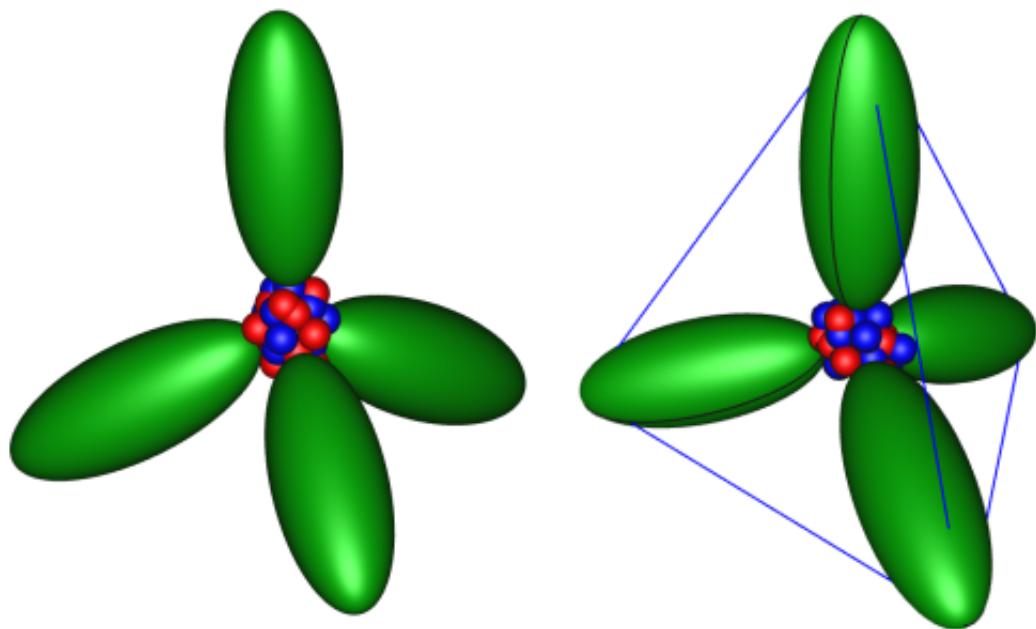


## *Niveles de descripción*

# Agrupación en niveles



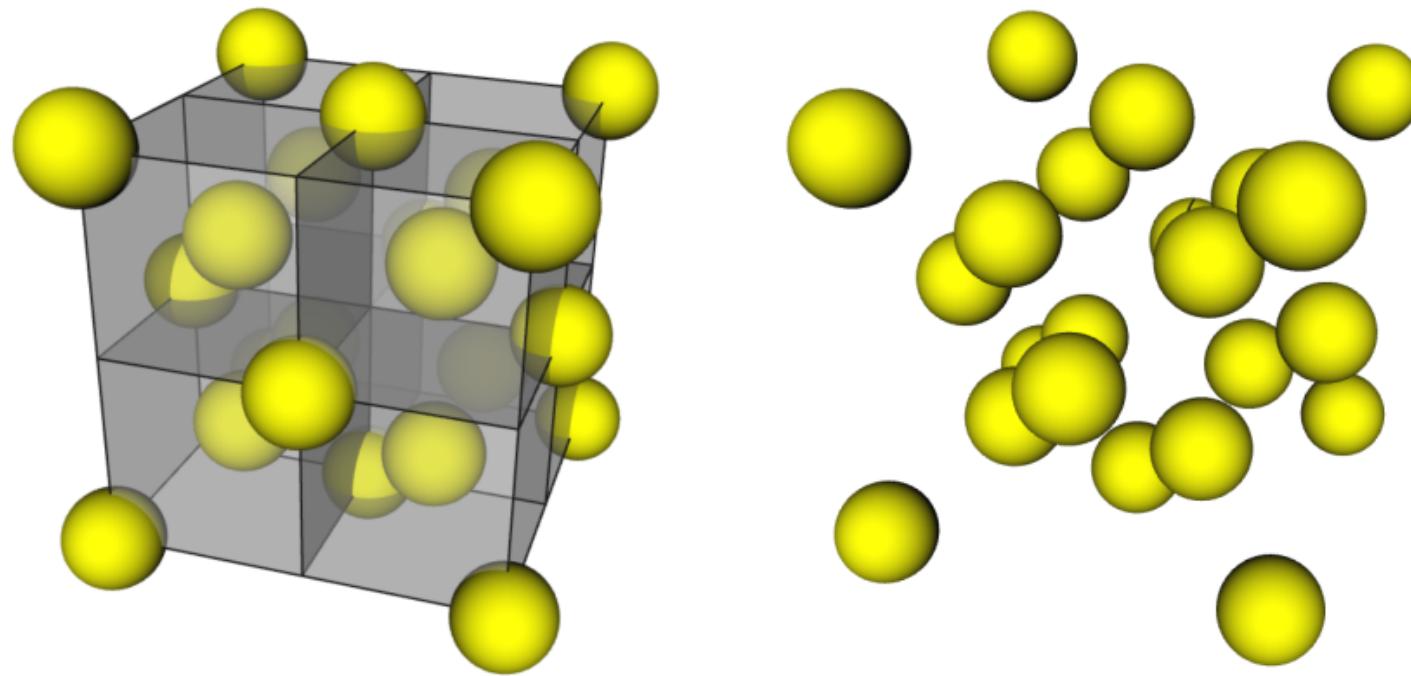
# Nivel 1: Átomos



*Átomo de silicio*

*Física, Química*

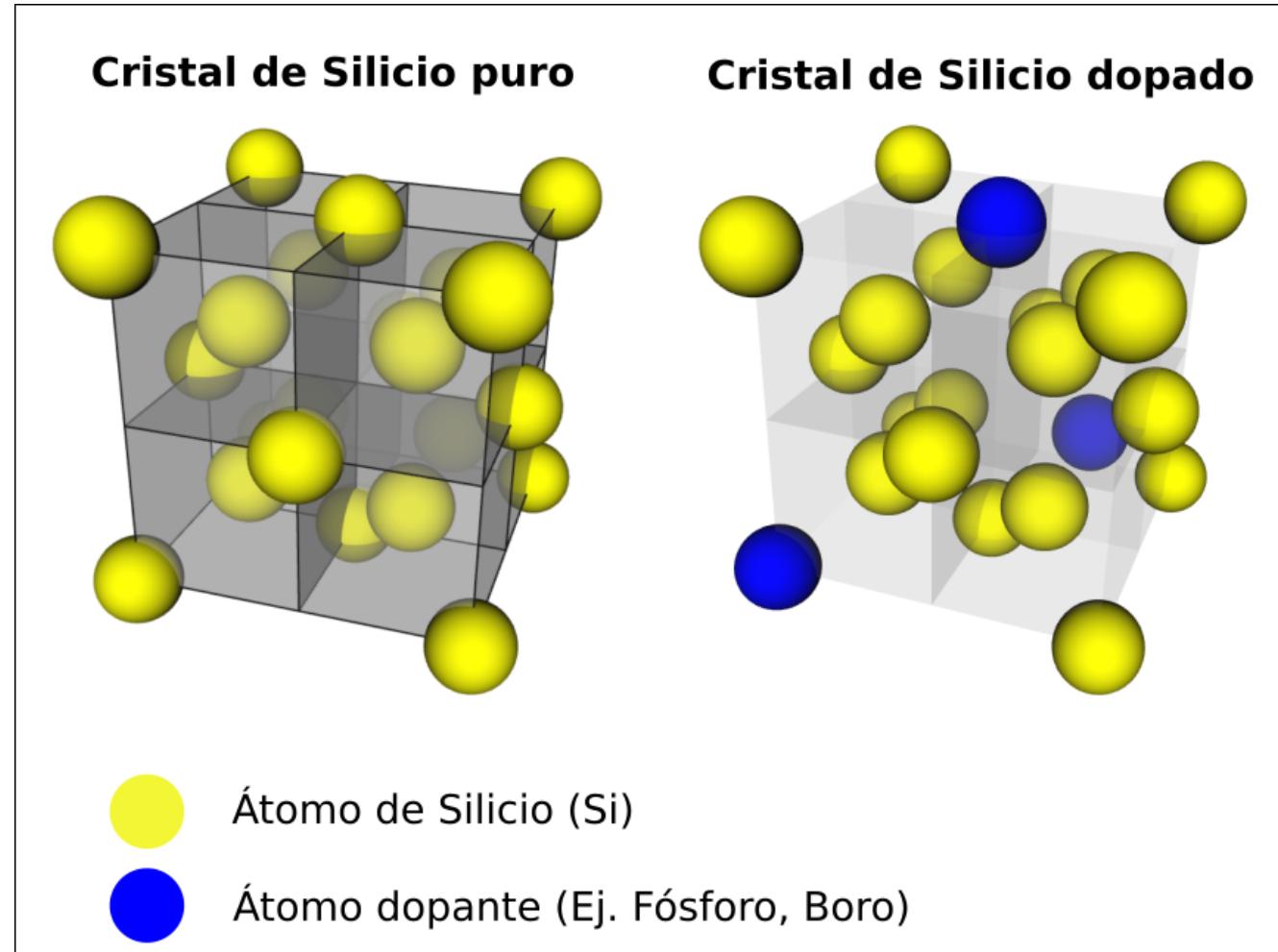
# Nivel 2: Materiales



*Cristal de silicio*

*Física del estado sólido*

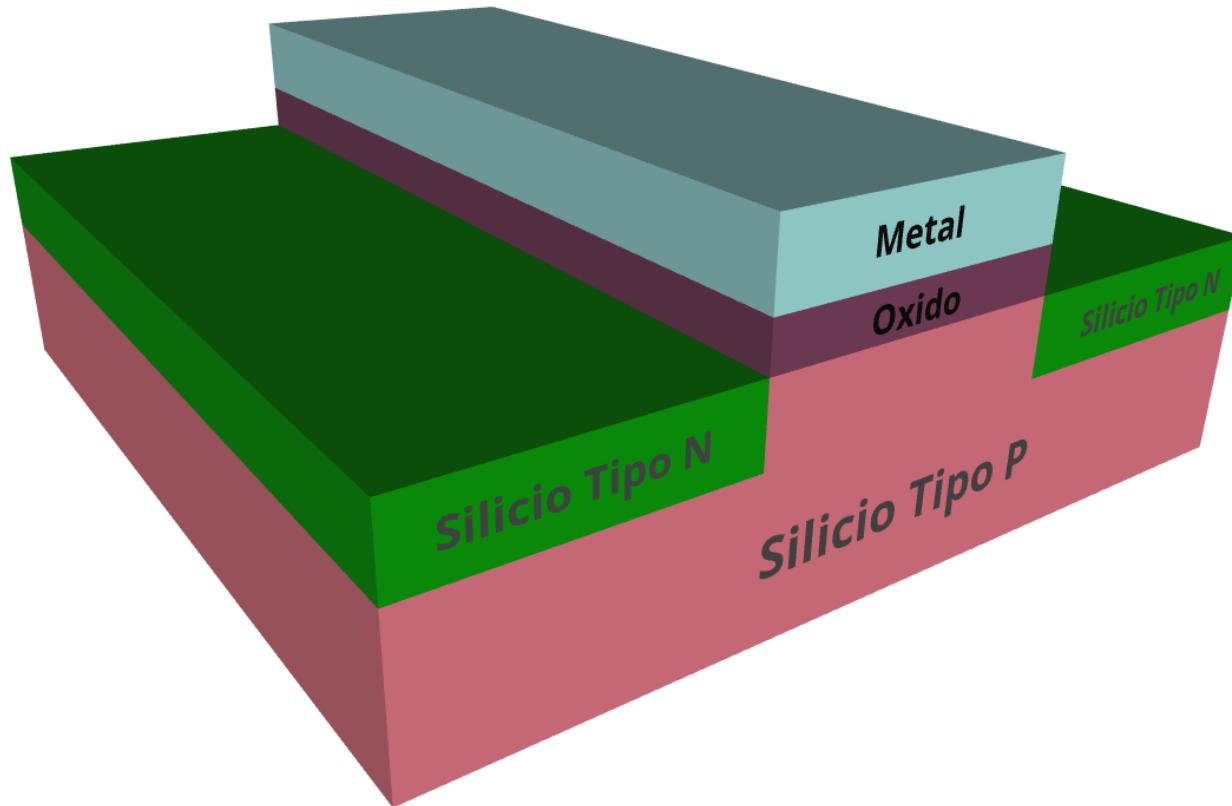
# Nivel 3: Semiconductores



*Uniones PN*

*Electrónica de dispositivos*

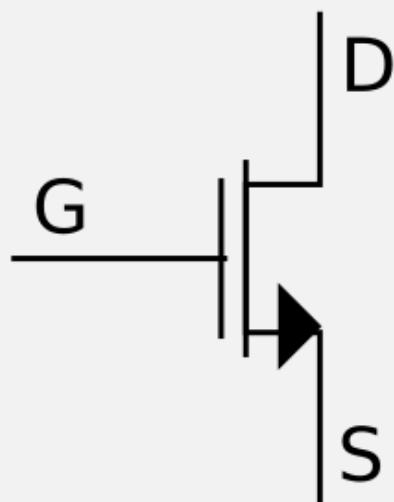
# Nivel 3: Semiconductores



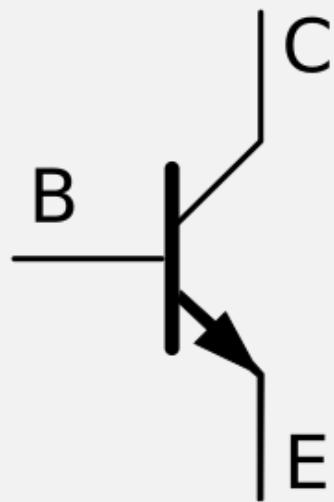
*Transistor  
MOSFET*

# Nivel 4: Transistor

**Transistor  
MOSFET**



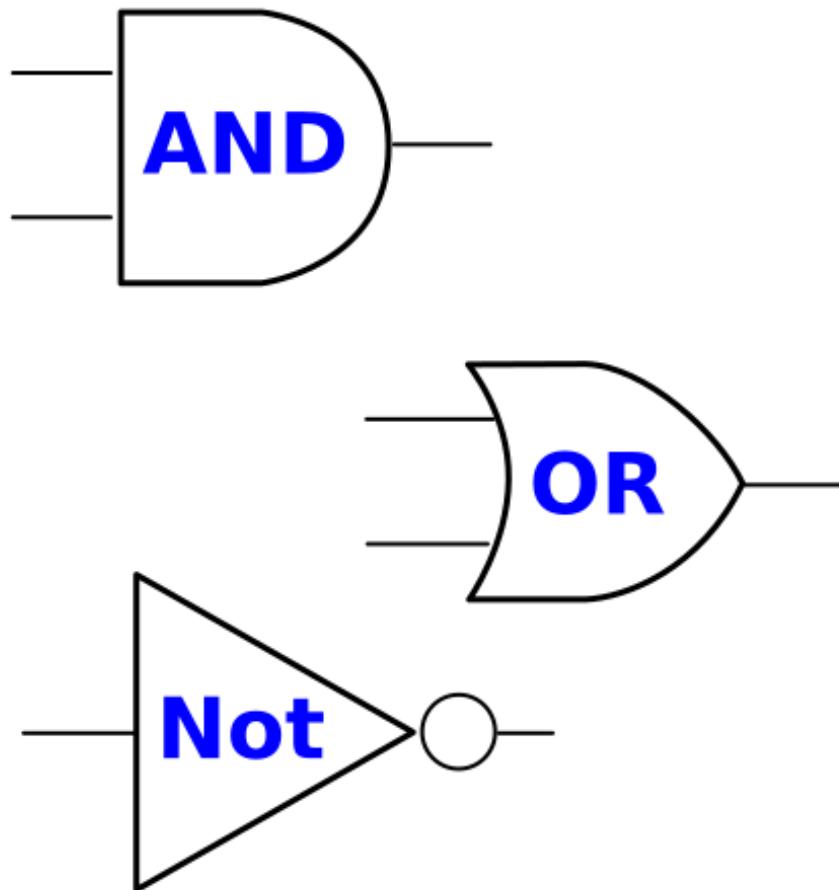
**Transistor  
Bipolar (NPN)**



*Transistor  
Microelectrónica*

# Nivel 5: Puertas lógicas

*Puertas Lógicas  
Electrónica  
Digital*

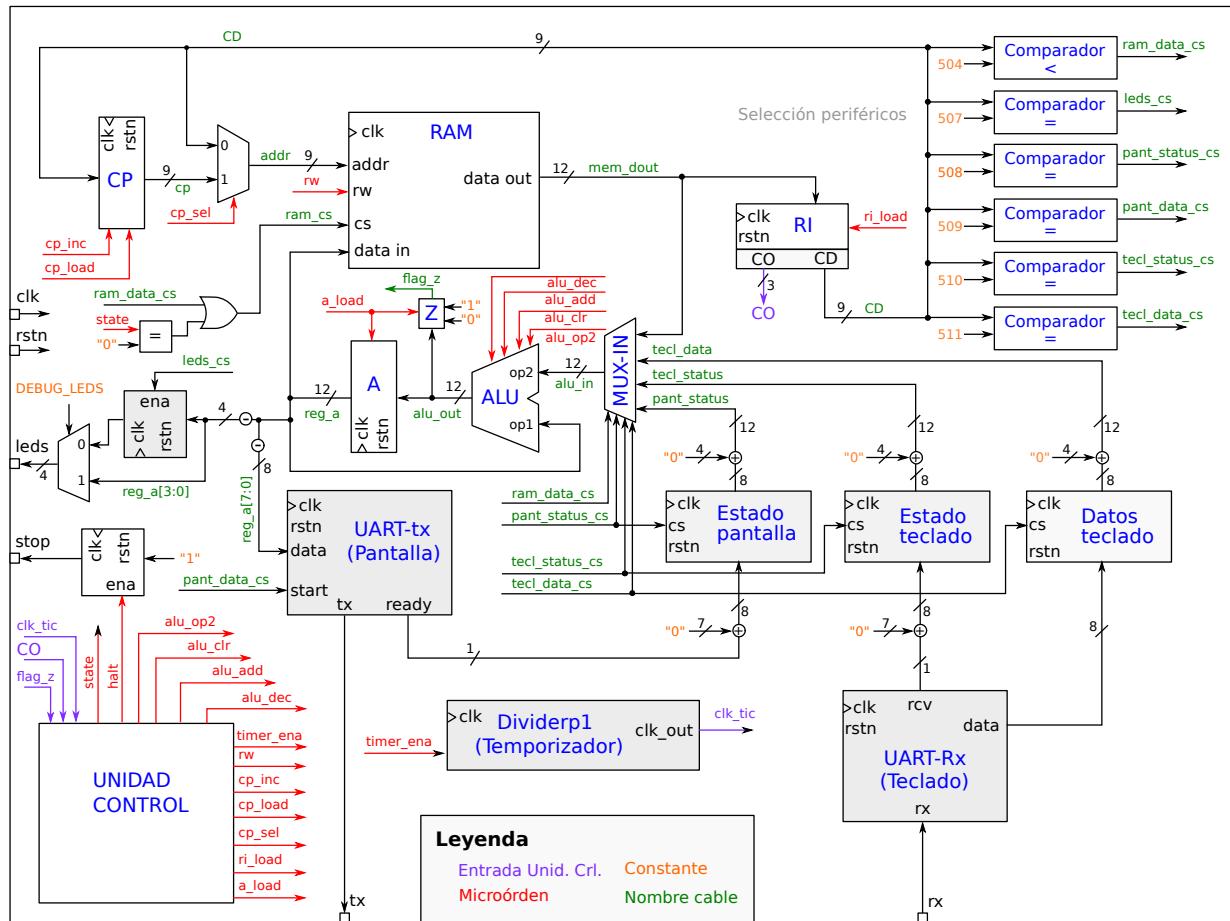


*¡Aparecen los bits!*

**0** = 0 voltios  
**1** = 5 voltios

# Nivel 6: Microprocesador

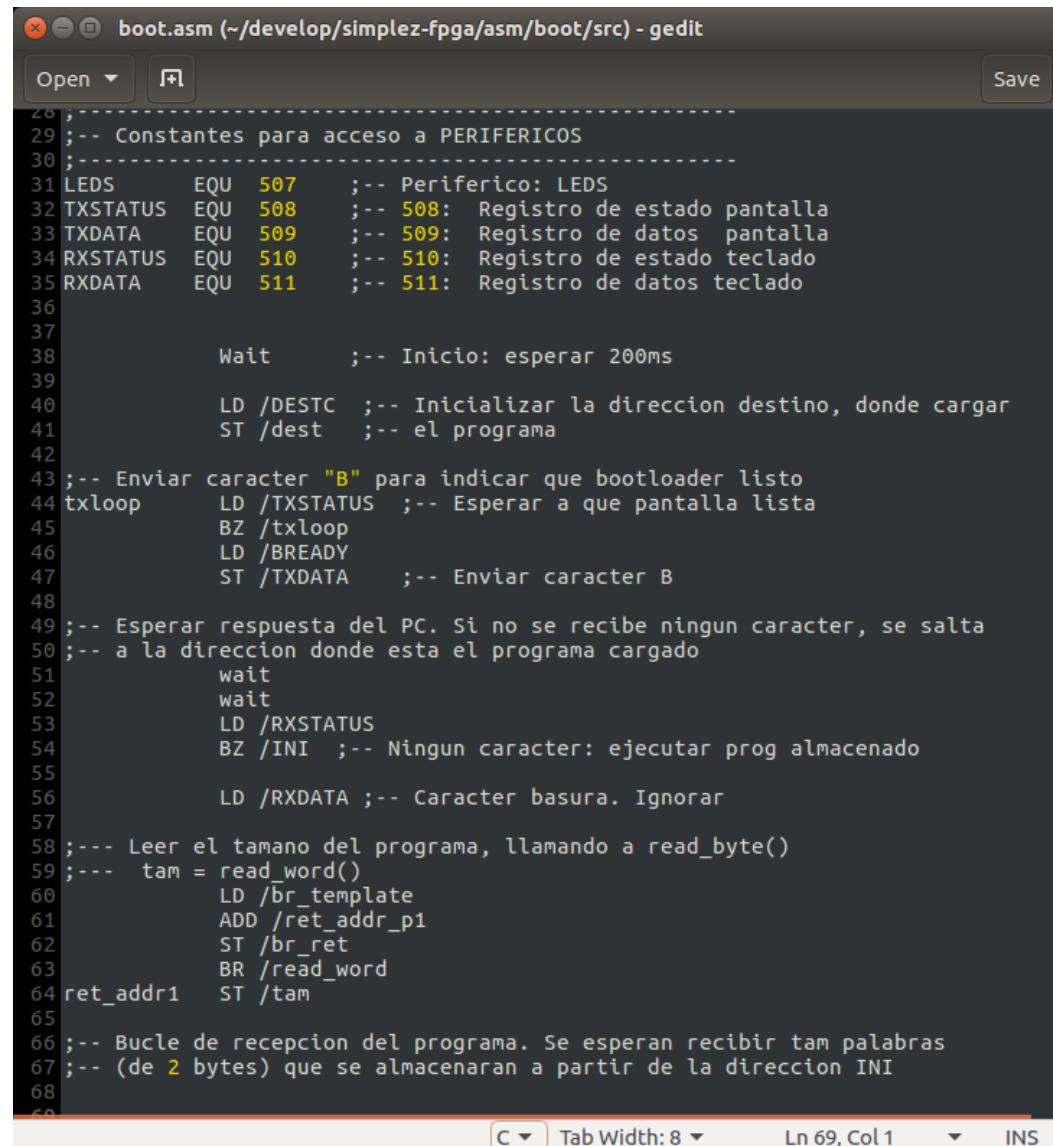
*Microprocesador  
Arquitectura de  
Ordenadores*



*Máquina universal!*

*Máquina reprogramable*

# Nivel 7: Software



```
boot.asm (~/develop/simplez-fpga/asm/boot/src) - gedit
Open Save
28;
29;-- Constantes para acceso a PERIFERICOS
30;
31LEDs EQU 507 ;-- Periferico: LEDs
32TXSTATUS EQU 508 ;-- 508: Registro de estado pantalla
33TXDATA EQU 509 ;-- 509: Registro de datos pantalla
34RXSTATUS EQU 510 ;-- 510: Registro de estado teclado
35RXDATA EQU 511 ;-- 511: Registro de datos teclado
36
37
38 Wait ;-- Inicio: esperar 200ms
39
40 LD /DESTC ;-- Inicializar la direccion destino, donde cargar
41 ST /dest ;-- el programa
42
43;-- Enviar caracter "B" para indicar que bootloader listo
44txloop LD /TXSTATUS ;-- Esperar a que pantalla lista
45 BZ /txloop
46 LD /BREADY
47 ST /TXDATA ;-- Enviar caracter B
48
49;-- Esperar respuesta del PC. Si no se recibe ningun caracter, se salta
50;-- a la direccion donde esta el programa cargado
51 wait
52 wait
53 LD /RXSTATUS
54 BZ /INI ;-- Ningun caracter: ejecutar prog almacenado
55
56 LD /RXDATA ;-- Caracter basura. Ignorar
57
58;-- Leer el tamano del programa, llamando a read_byte()
59;-- tam = read_word()
60 LD /br_template
61 ADD /ret_addr_p1
62 ST /br_ret
63 BR /read_word
64 ret_addr1 ST /tam
65
66;-- Bucle de recepcion del programa. Se esperan recibir tam palabras
67;-- (de 2 bytes) que se almacenaran a partir de la direccion INI
68
```

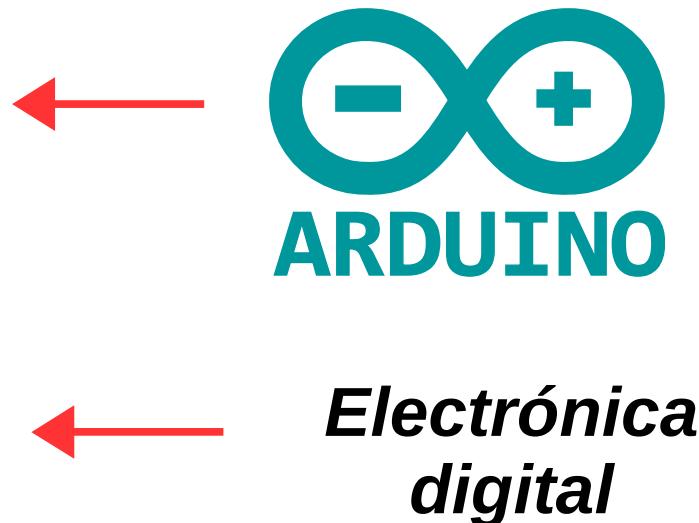
*Software  
Programación*

*Código  
Algoritmos*

*Muchos niveles de software*

# Arduino y Electrónica digital

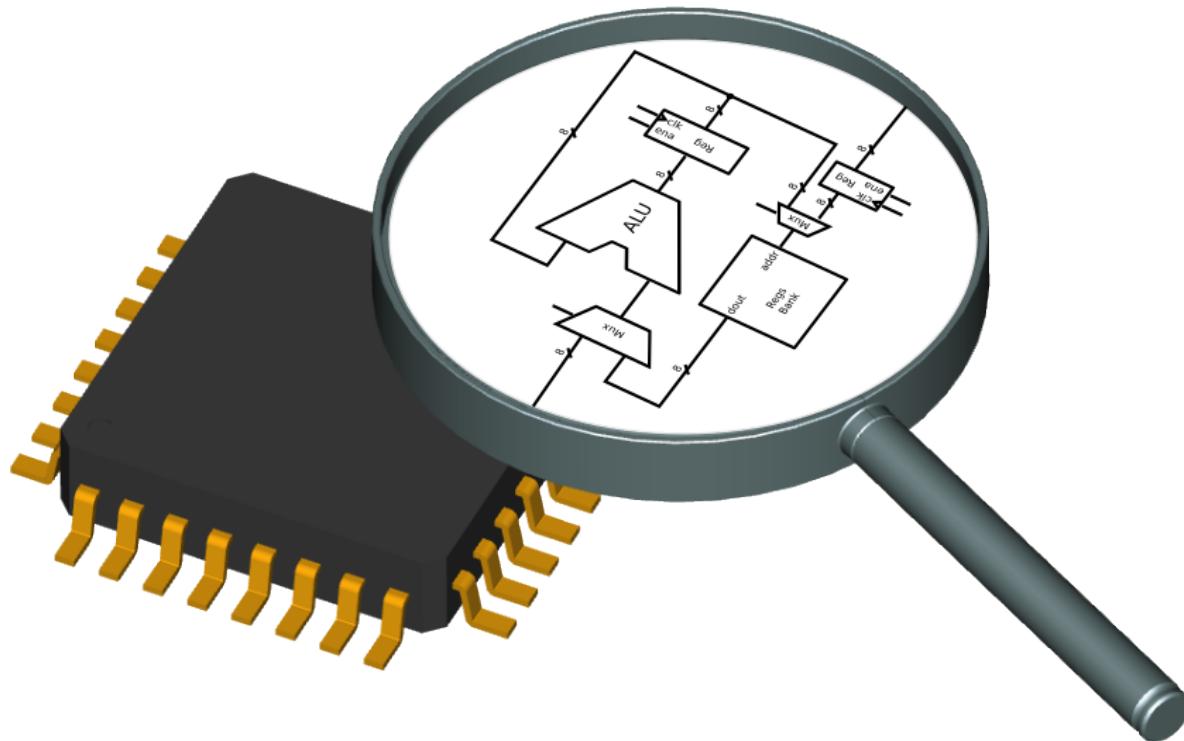
	<b>Nivel 7:</b> Software
	<b>Nivel 6:</b> Microprocesador
	<b>Nivel 5:</b> Puertas lógicas
	<b>Nivel 4:</b> Transistor
	<b>Nivel 3:</b> Semiconductores
	<b>Nivel 2:</b> Materiales
	<b>Nivel 1:</b> Átomos



**Electrónica  
digital**

# *Electrónica Digital y FPGAs*

# Diseño de microchips

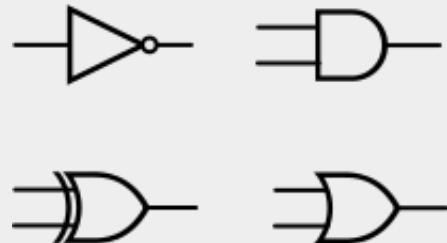


- Nivel de electrónica digital
- Información: Sólo 1s y 0s (Bits)
- Función: **Manipular, almacenar y transportar bits**

# Elementos en circuitos digitales

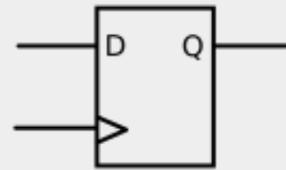
## Puertas lógicas

Manipulación bits



## Biestables

Almacenamiento bits



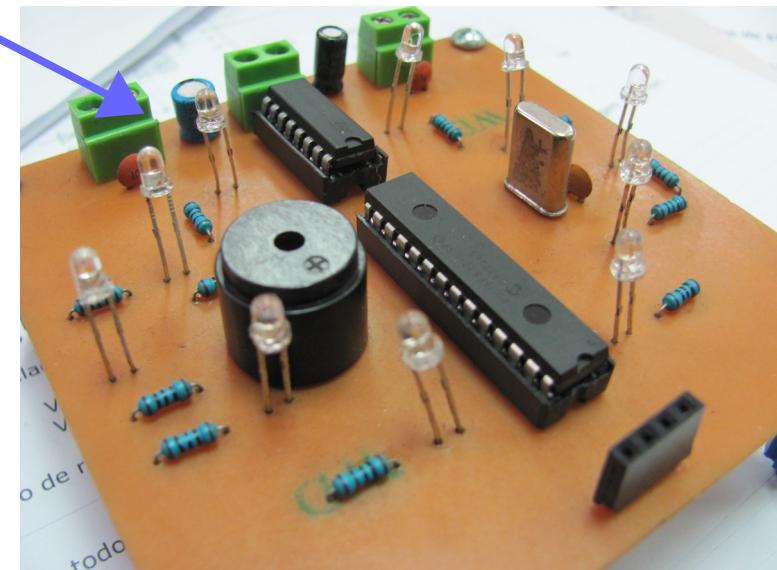
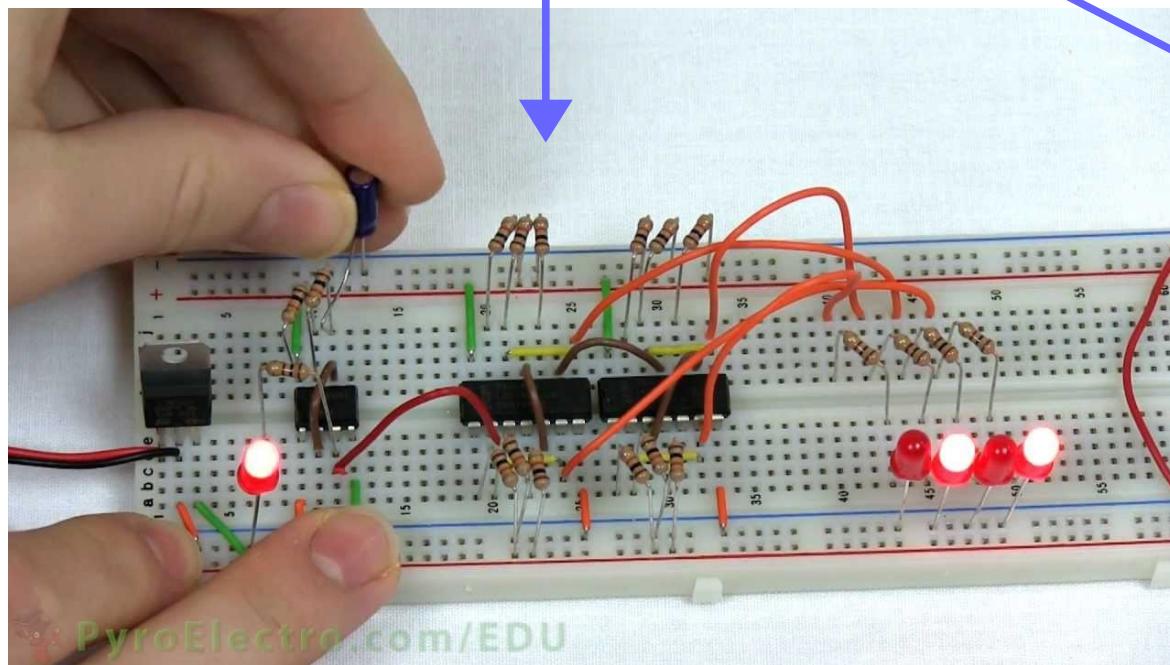
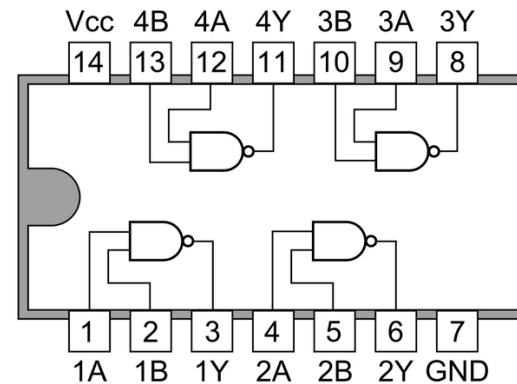
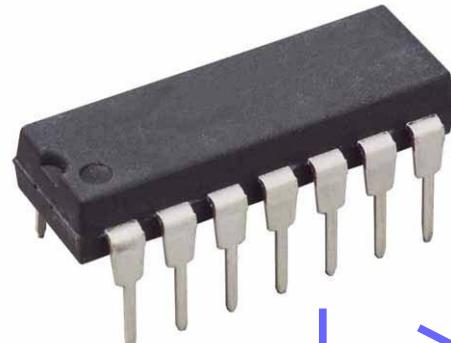
## Cables

Transporte bits

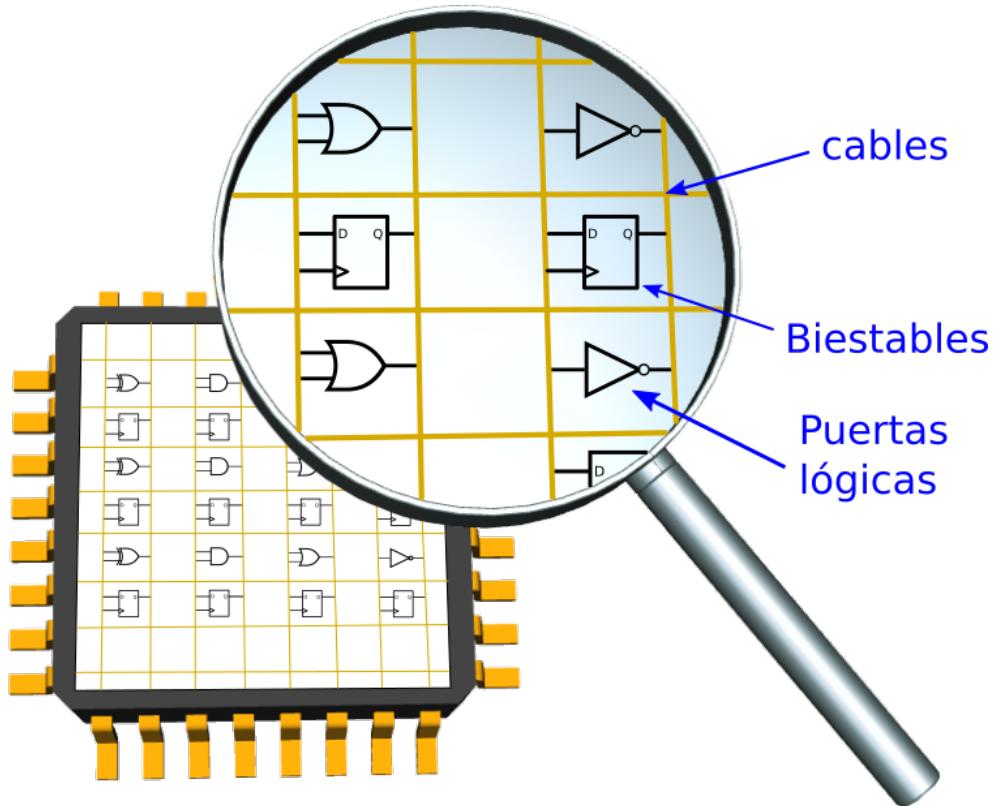


Cualquier circuito digital, por muy complejo que sea, se descompone en estos 3 tipos de **componentes elementales**

# ¿Cómo se hacen los circuitos digitales?

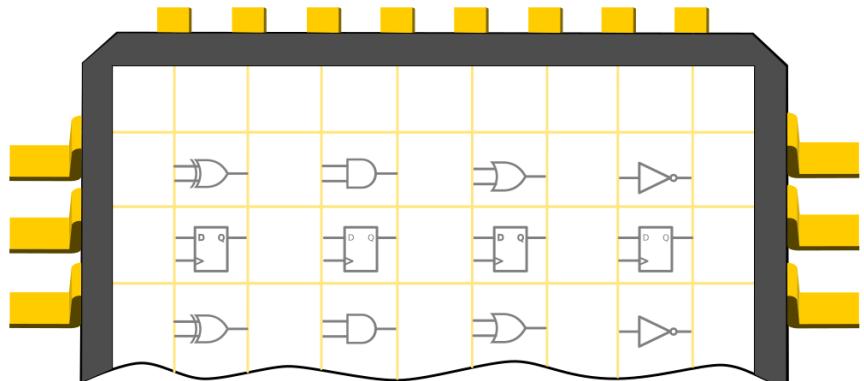


# Tecnología FPGA

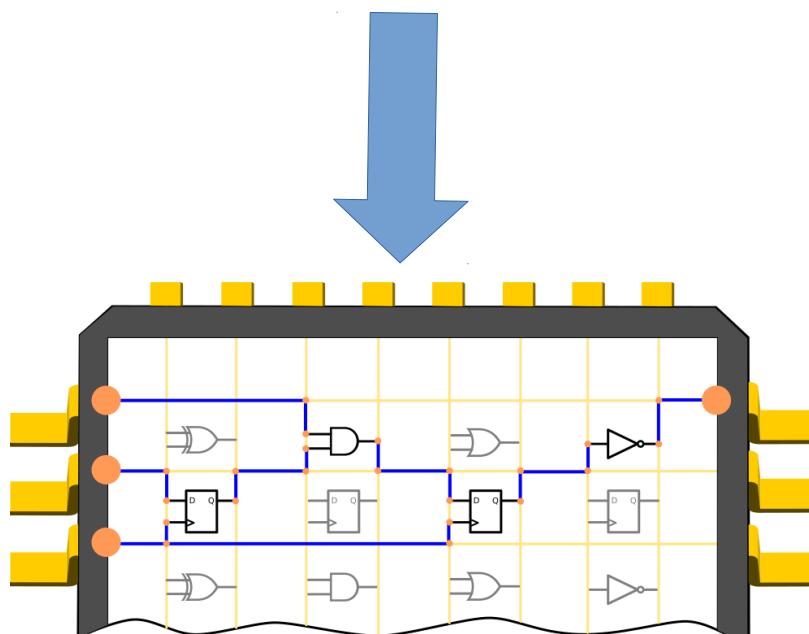


**FPGA:** Chip “en blanco” que contiene una matriz con los 3 componentes básicos: puertas lógicas, biestables y cables

# Electrónica digital con FPGAs

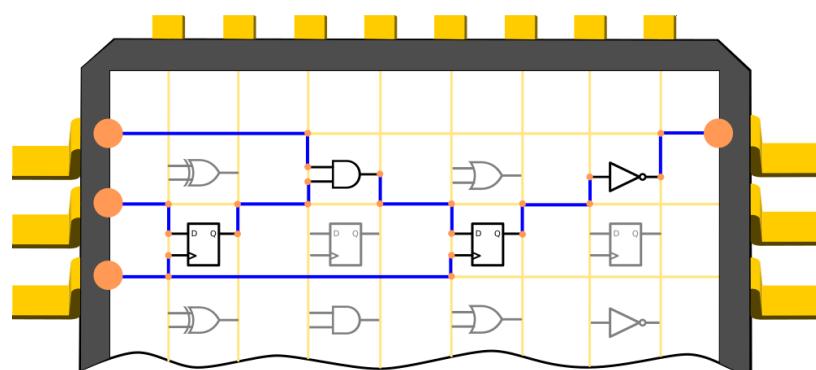
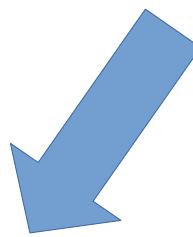
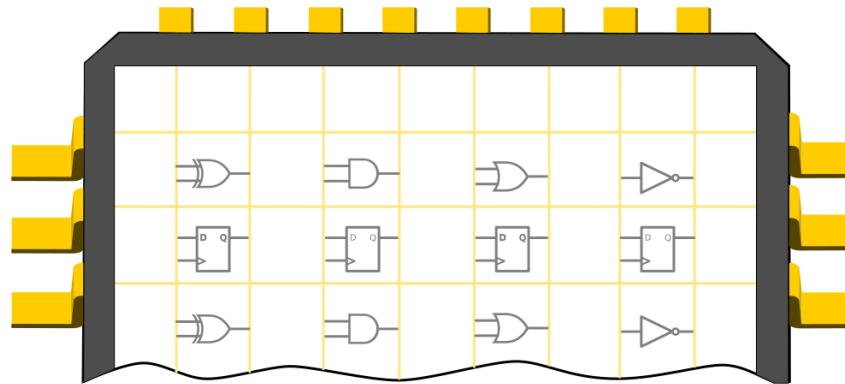


FPGA en Blanco

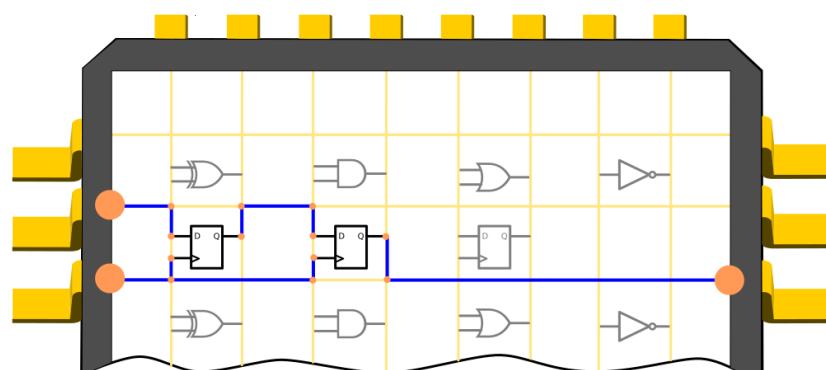
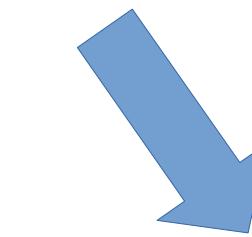


FPGA configurada

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



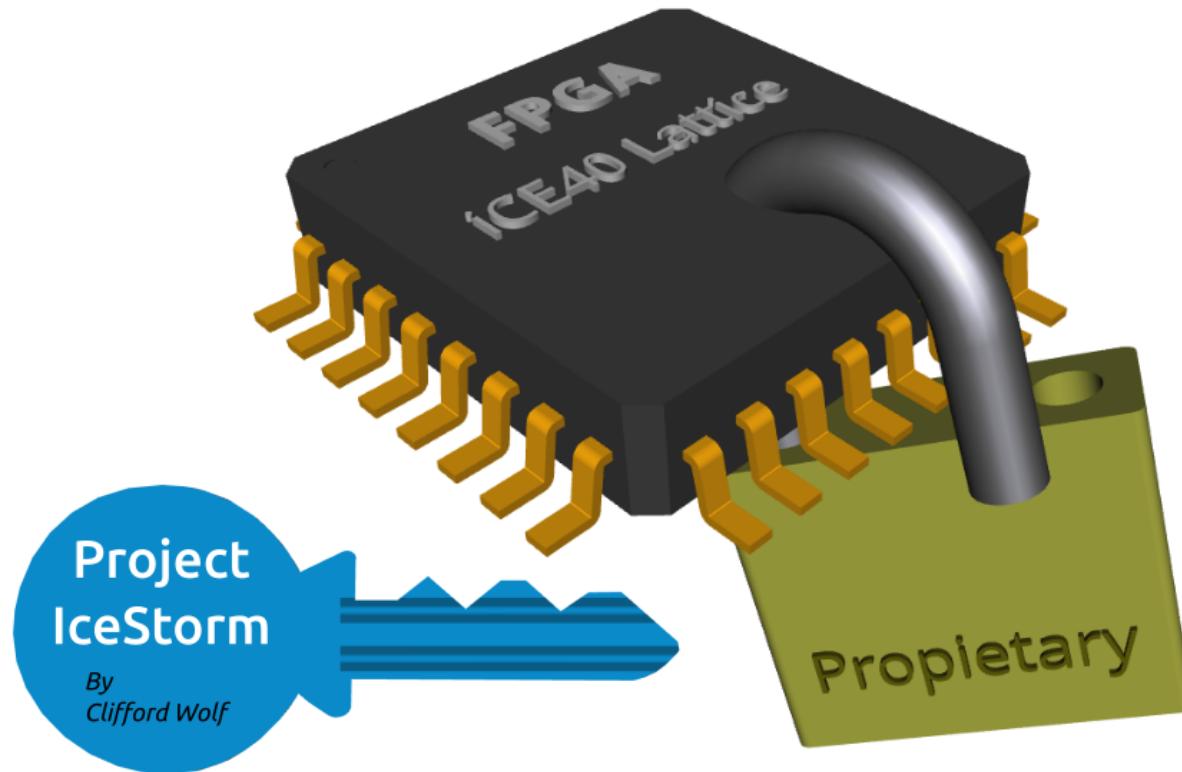
**Circuito 1**



**Circuito 2**

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

# FPGAs libres: El renacimiento



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

# Comunidad FPGAwars



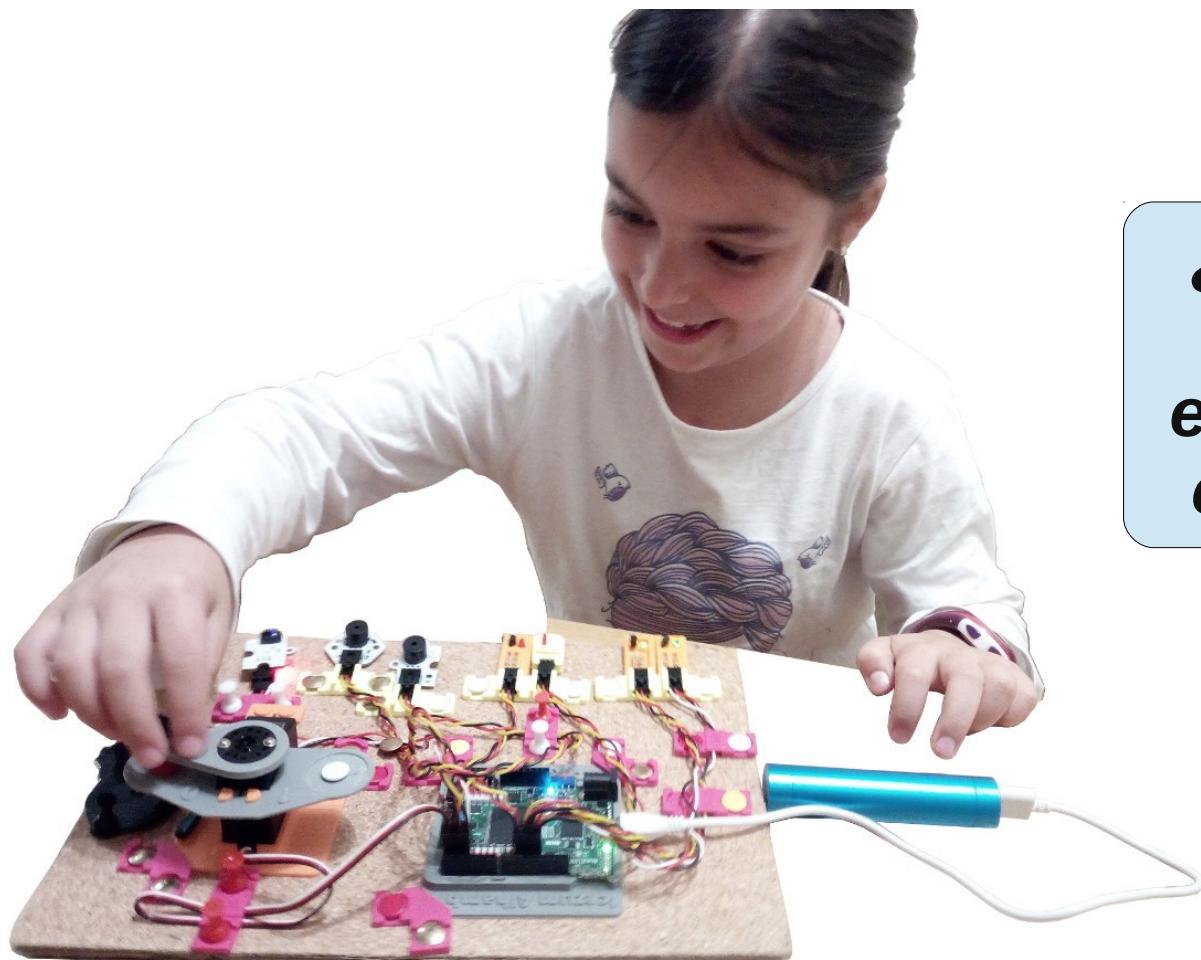
- Comunidad para **compartir conocimiento** relacionado con **FPGAs libres**
- Es el **clonewars** de las FPGAs, pero en modesto :-)
- Idioma: Castellano
- 424 miembros
- Cualquier pregunta / comentario / sugerencia → Correo a la lista :-)

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

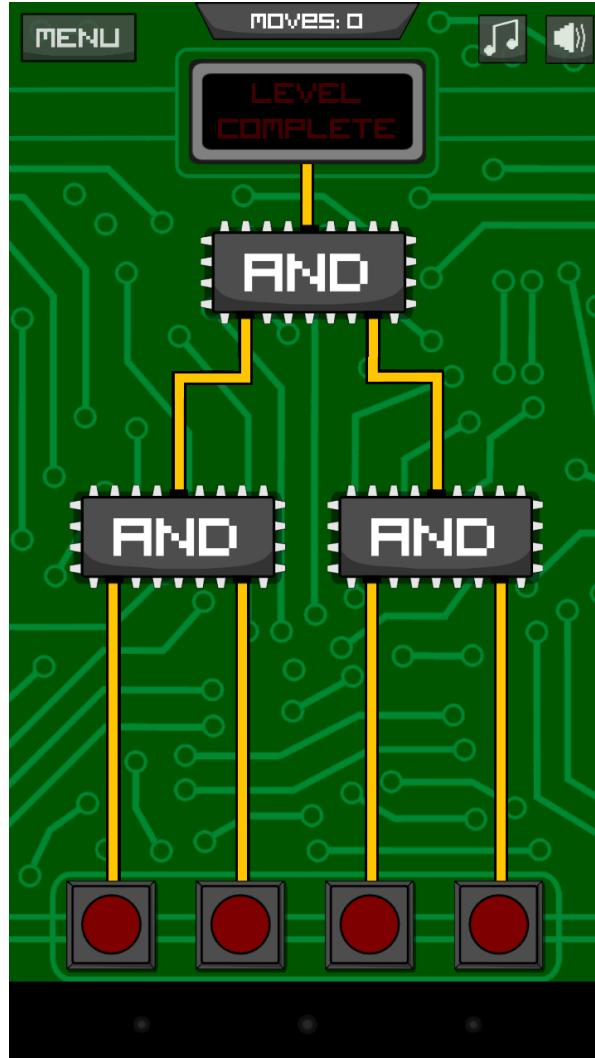
***Electrónica Digital fácil, intuitiva y divertida***

# Motivación

## Electrónica digital accesible

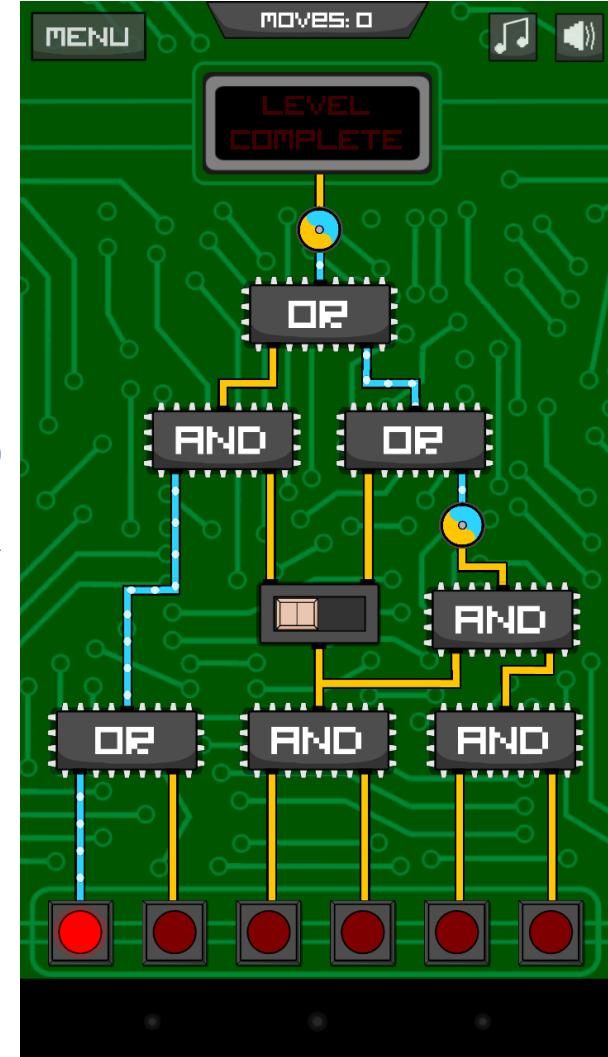


*¿Cómo podrían los  
niños y los no  
electrónicos diseñar  
circuitos digitales?*



APP:  
Circuit  
Scramble

Demo



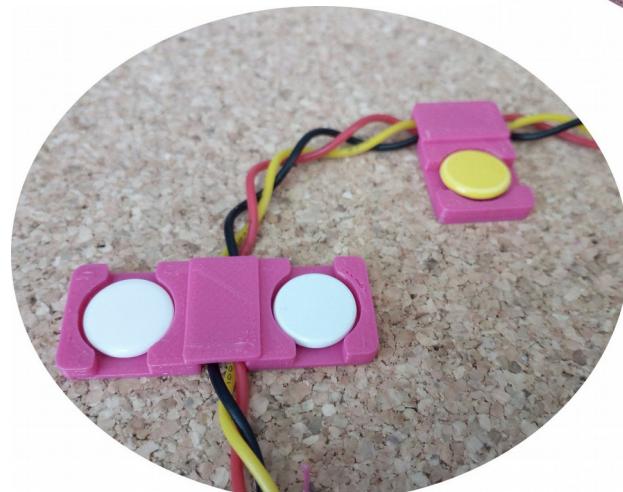
La electrónica digital es intuitiva y...  
¡Divertida!

# Elementos

**Panel de corcho (28x19cm)**



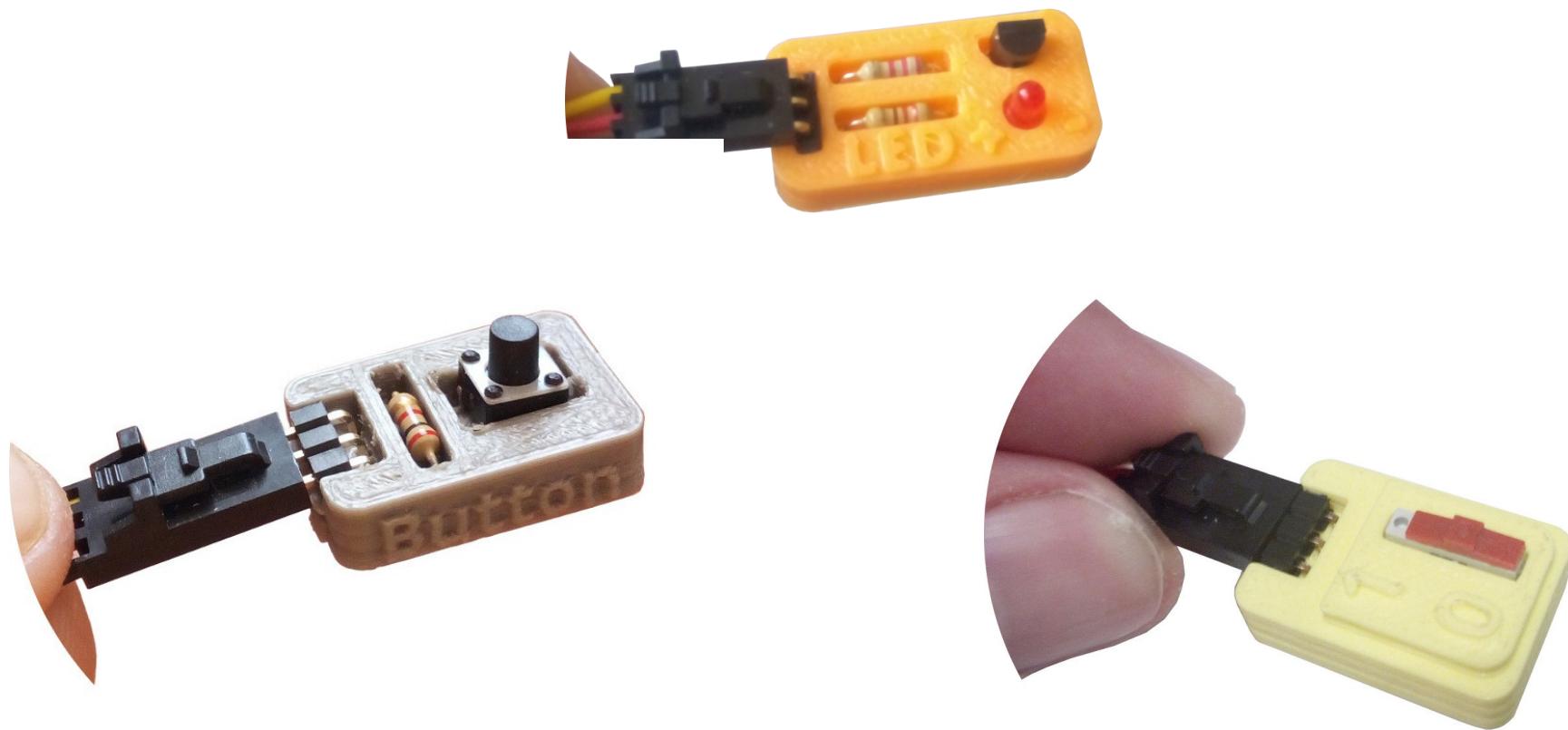
**Piezas Impresas en 3D para fijar elementos al corcho**



<https://github.com/Obijuan/3D-parts/wiki>

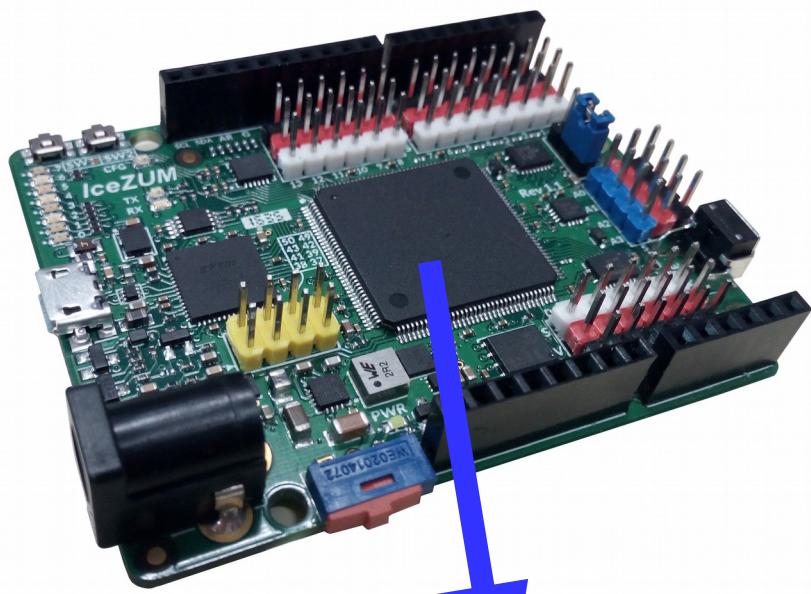
# Periféricos

**PCBprints:** Mini-circuitos impresos en 3D



# Icezum Alhambra v1.1

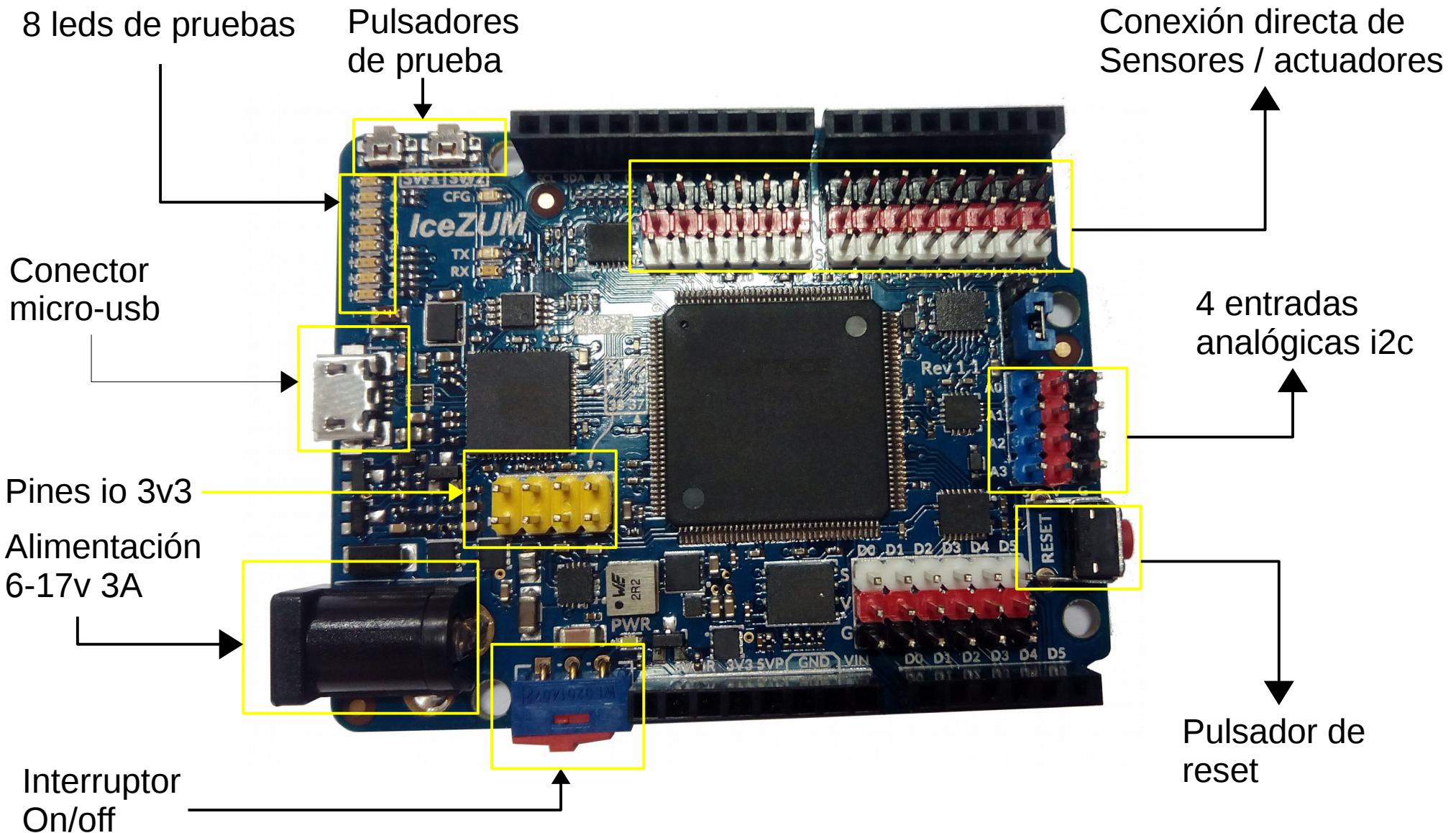
<https://github.com/FPGAwars/icezum/wiki>



**FPGA Libre**

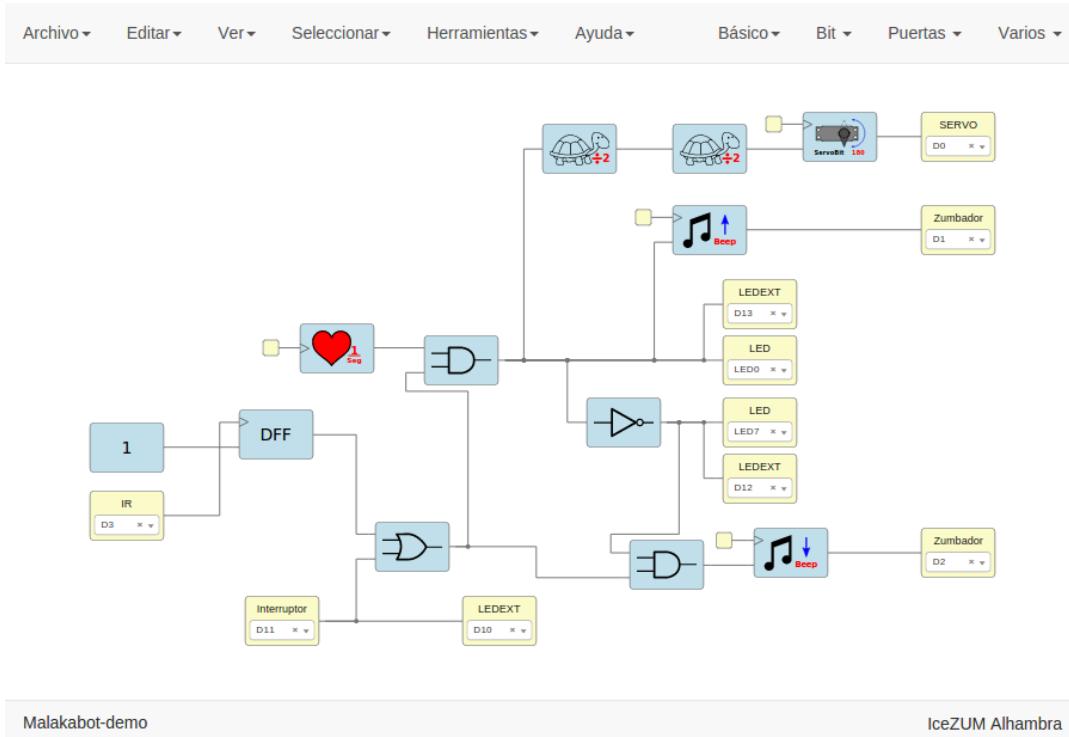
- Autor: **Eladio Delgado**
- Diseñada en Pinos del Valle (Granada)
- Arduino de las **FPGAs**
- Compatible Arduino
- Fácil conexión de circuitos externos/sensores/servos
- Reutilización de los shields de arduino
- 20 entradas/salidas de 5v
- 3A corriente de entrada
- Perfecta para hacer robots

# Icezum Alhambra v1.1





# icesstudio



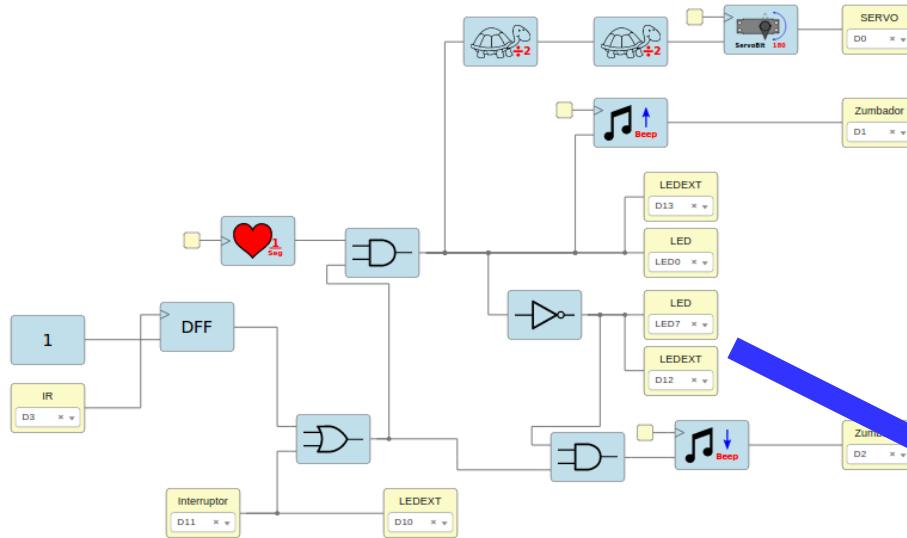
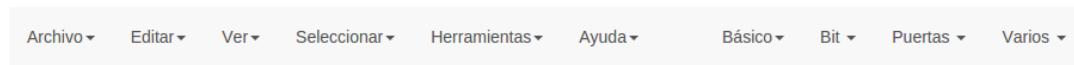
Malakabot-demo

IceZUM Alhambra

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

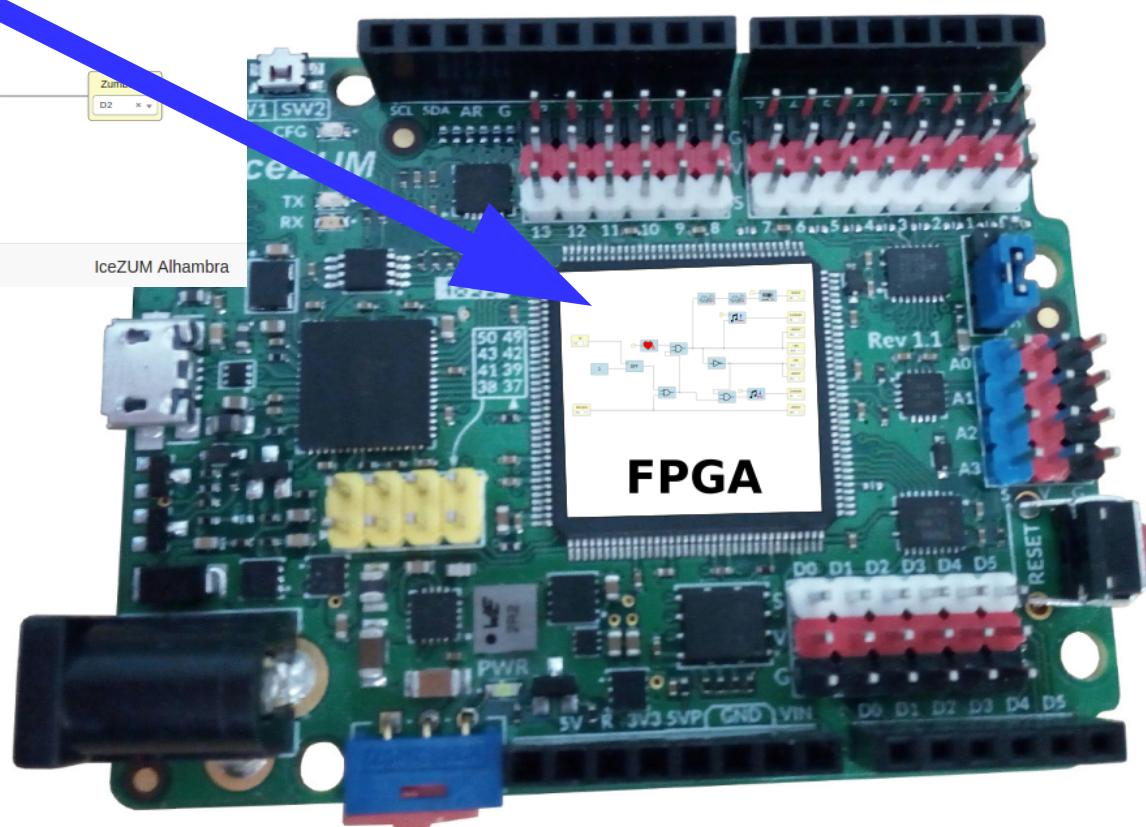
- Autor: **Jesús Arroyo**
- Electrónica digital para todos
- Herramienta visual
- Traduce a verilog

# La magia de las FPGAs

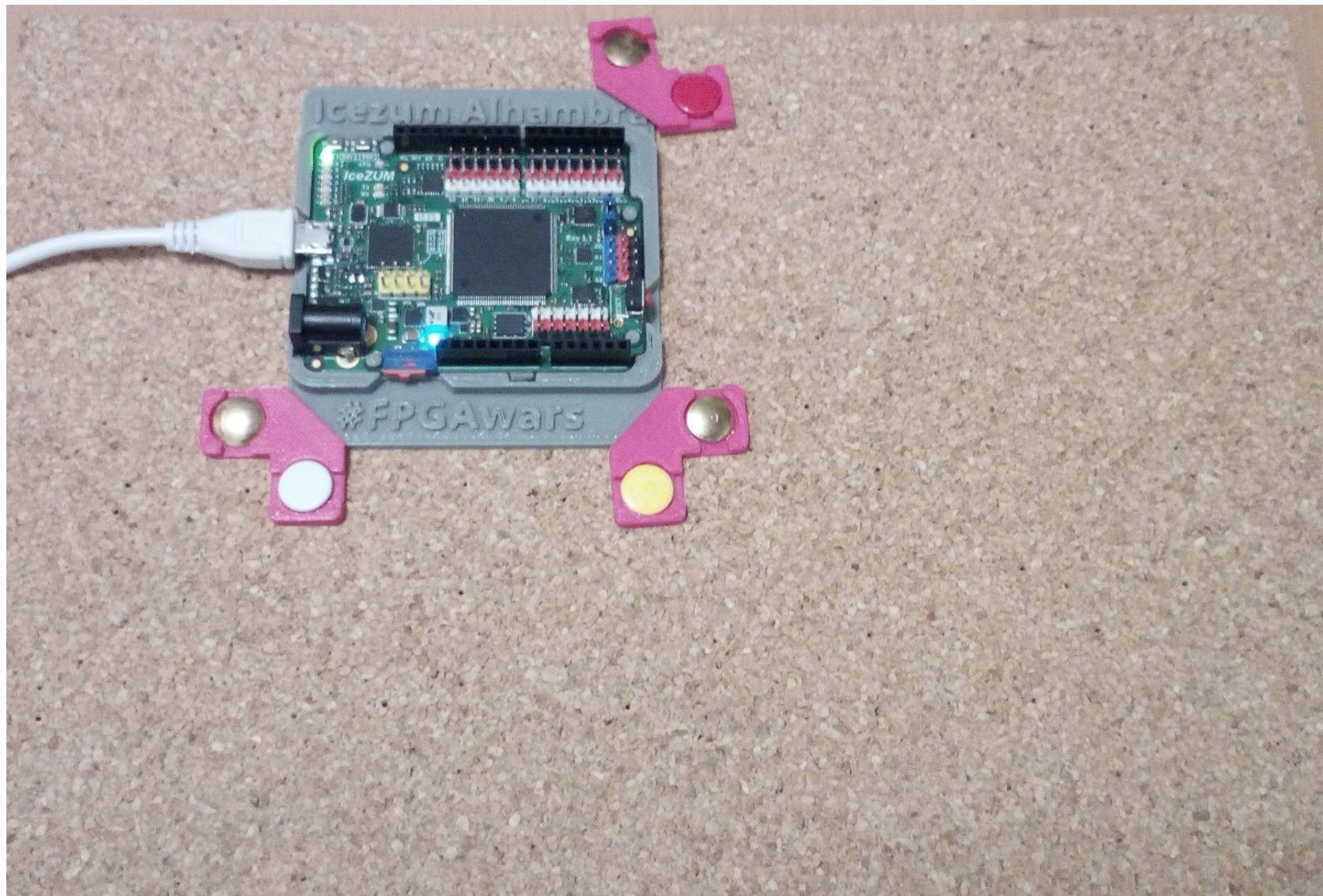


Malakabot-demo

IceZUM Alhambra



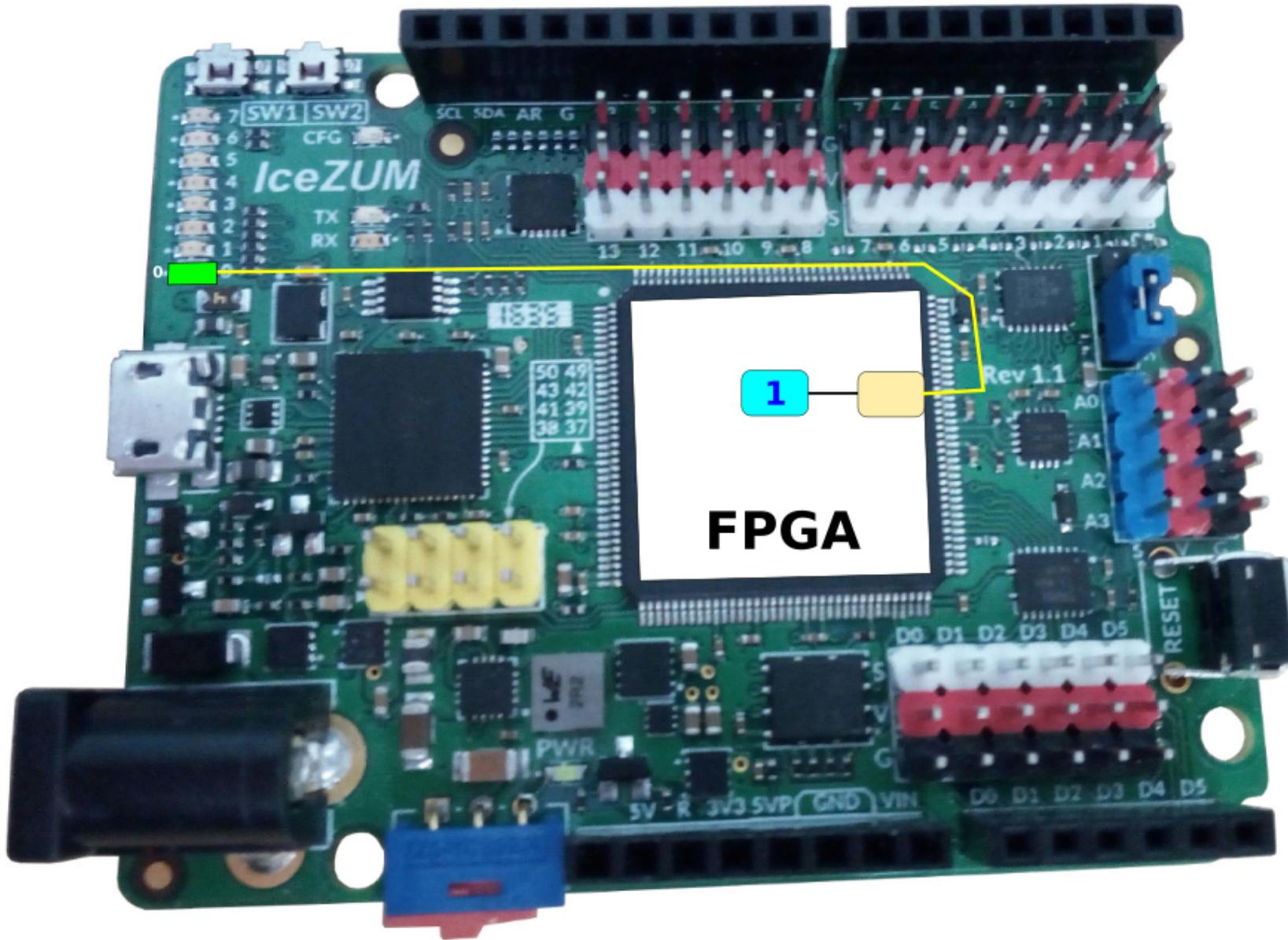
# ¡Empezamos!



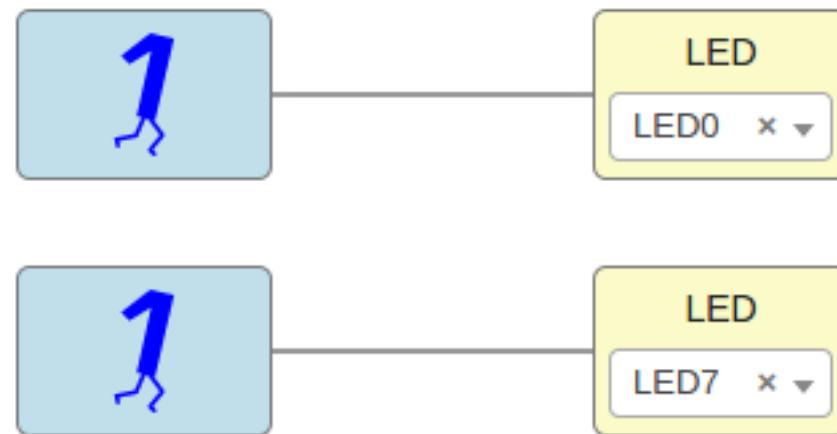
# Ejemplo 1: Hola Mundo



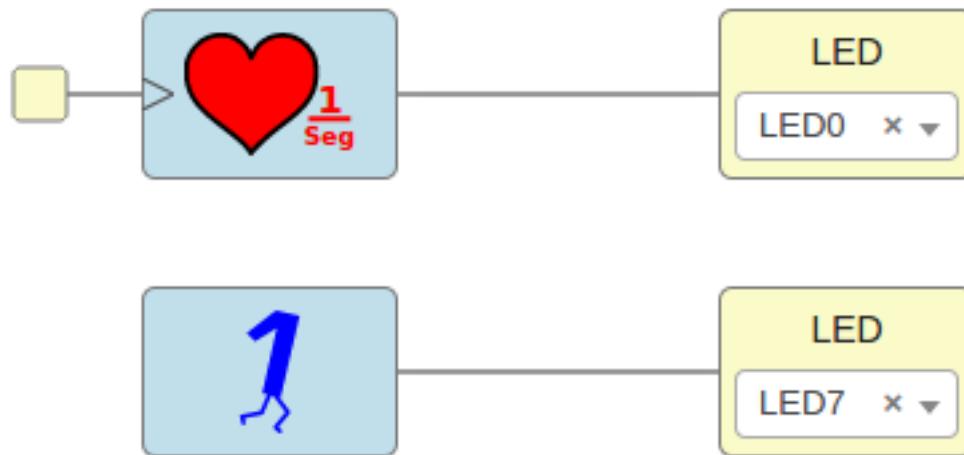
# Hola mundo: Implementación física



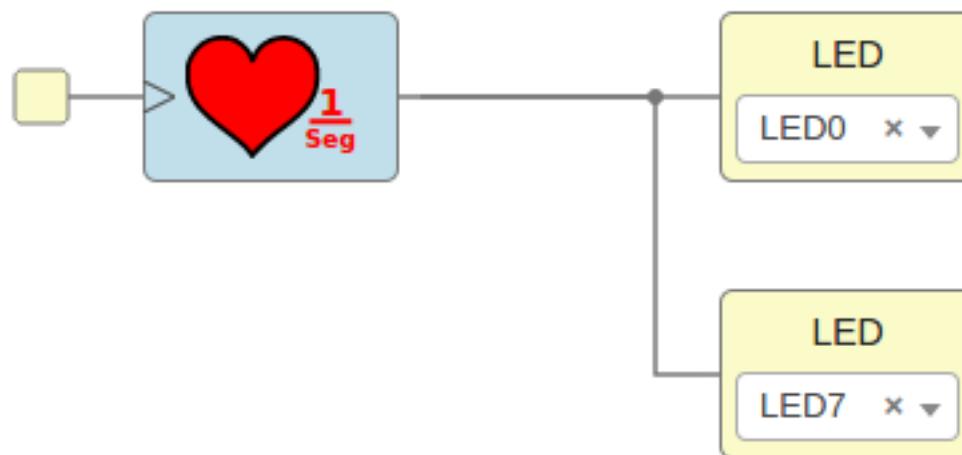
# Ejemplo 2: Dos leds en paralelo



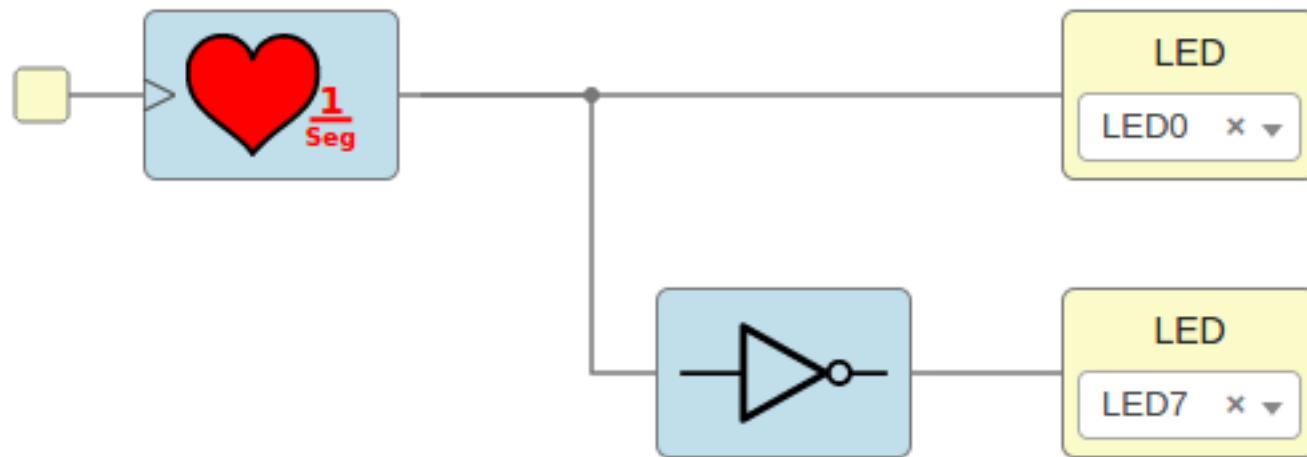
# Ejemplo 3: Led pulsante



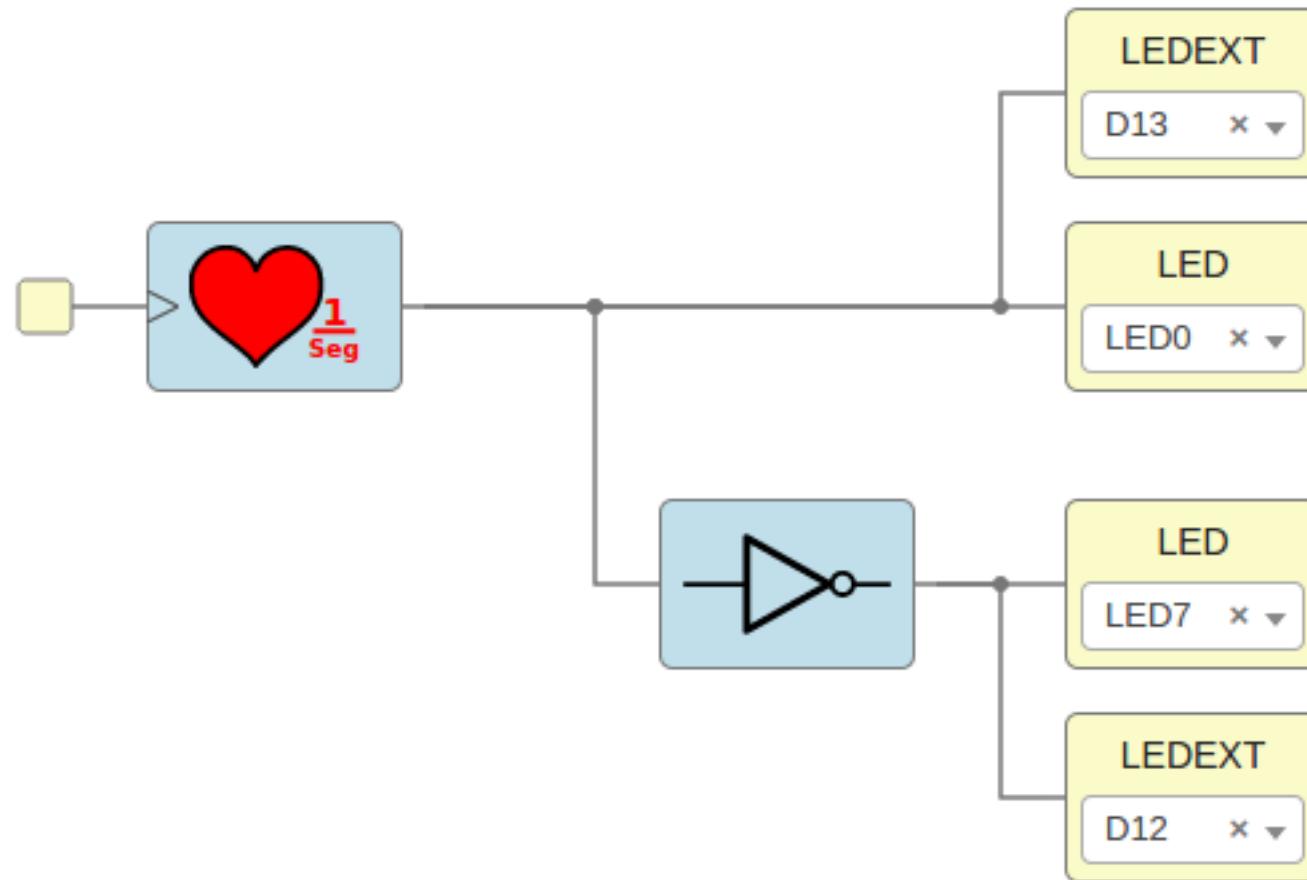
# Ejemplo 4: Leds pulsantes Mismo ritmo



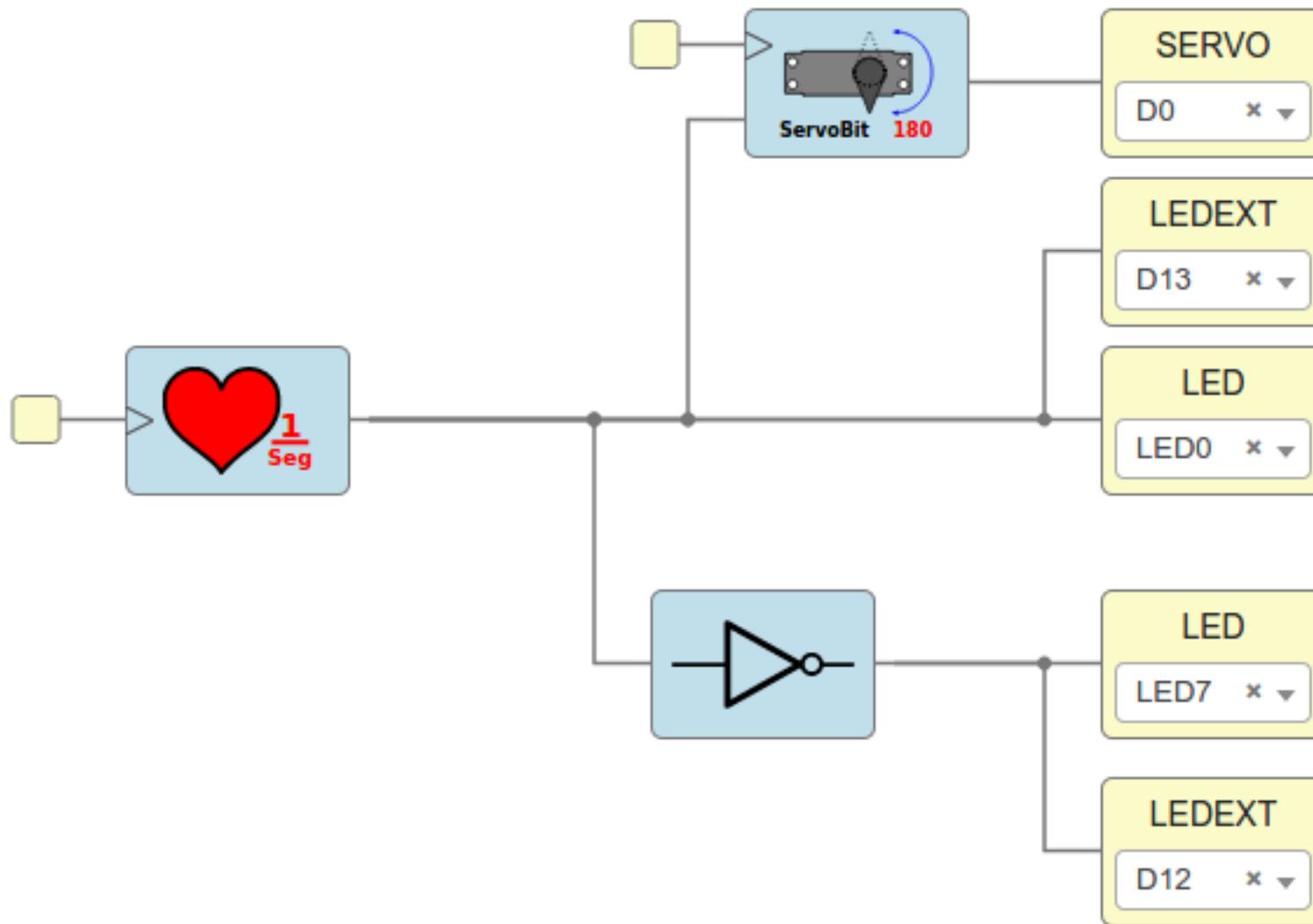
# Ejemplo 5: Leds alternativos



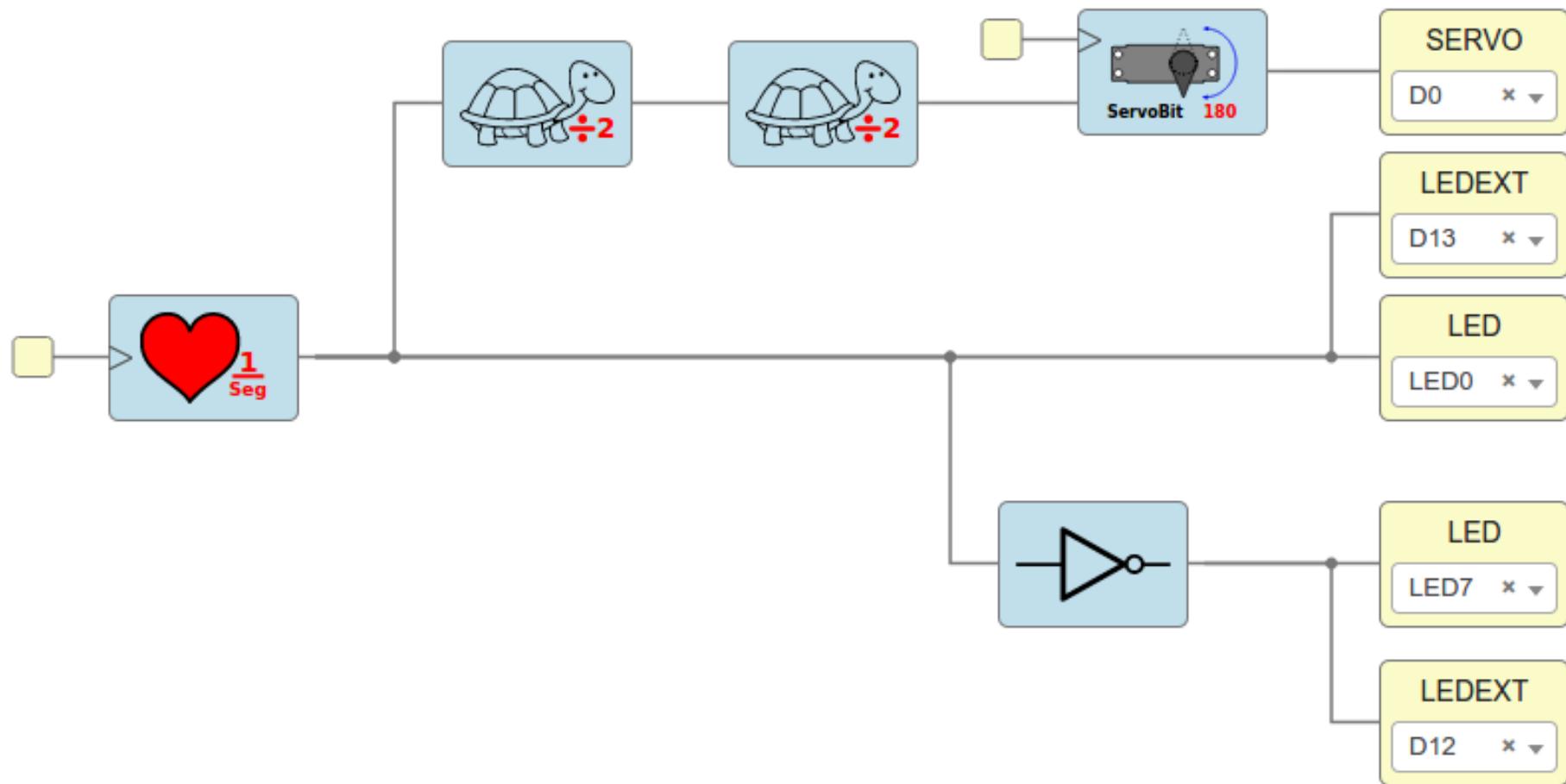
# Ejemplo 6: Leds externos



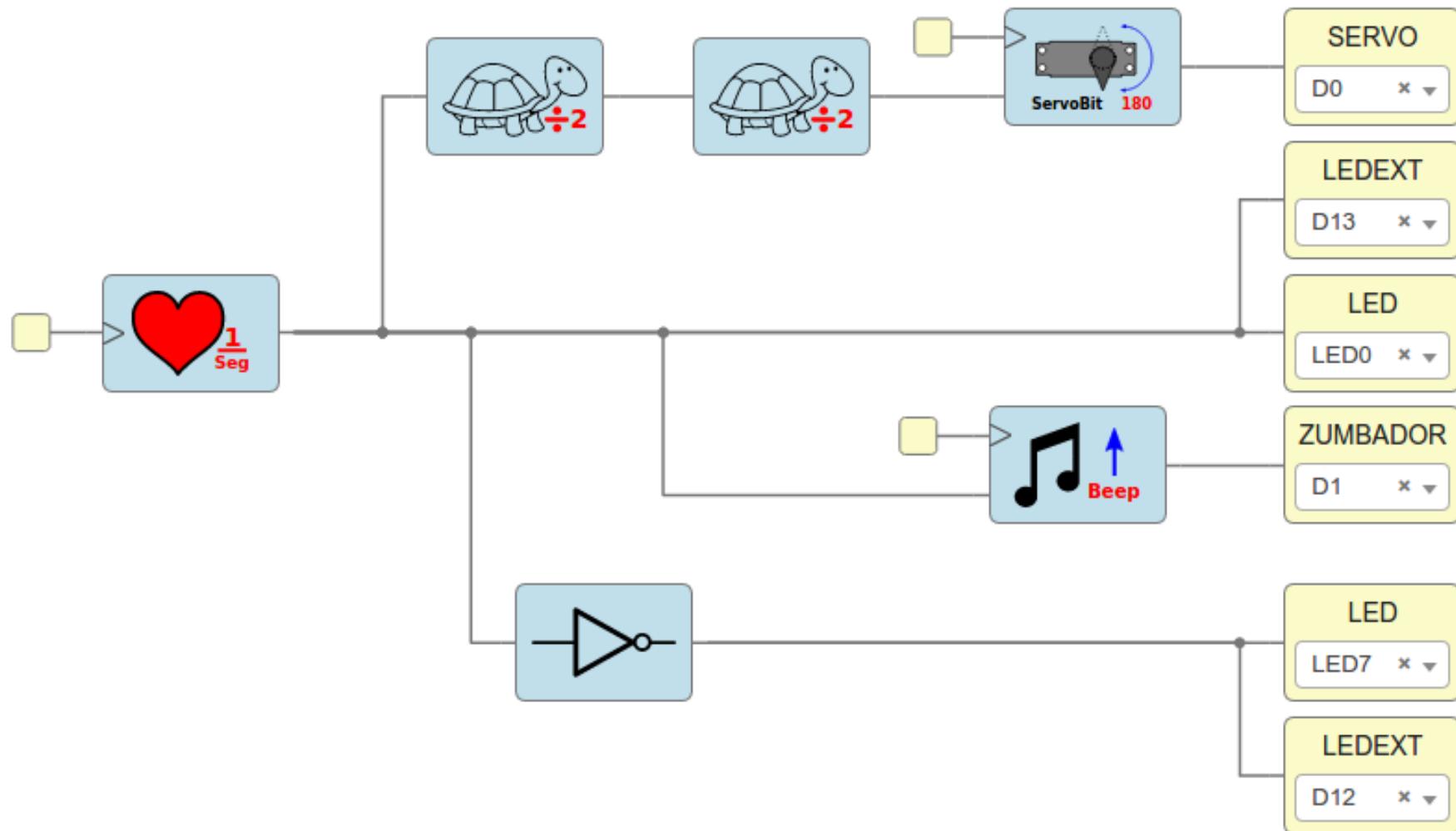
# Ejemplo 7: Servo binario



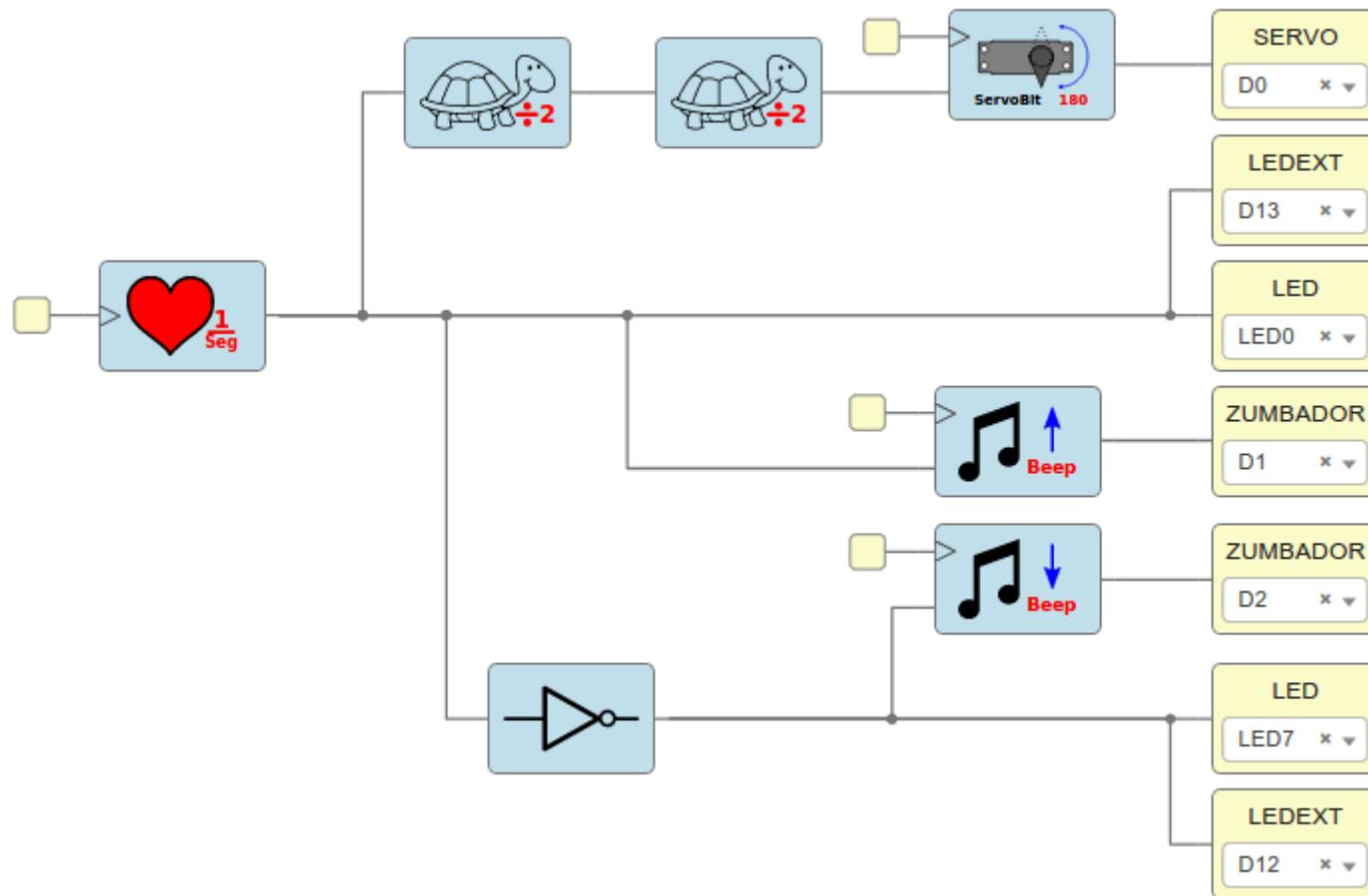
# Ejemplo 8: Bajando el ritmo



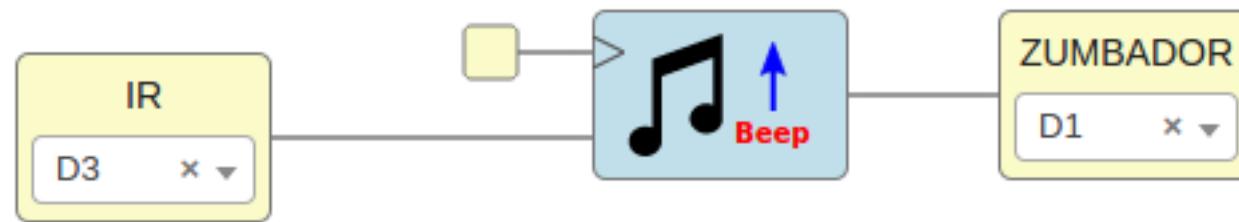
# Ejemplo 9: Zumbador



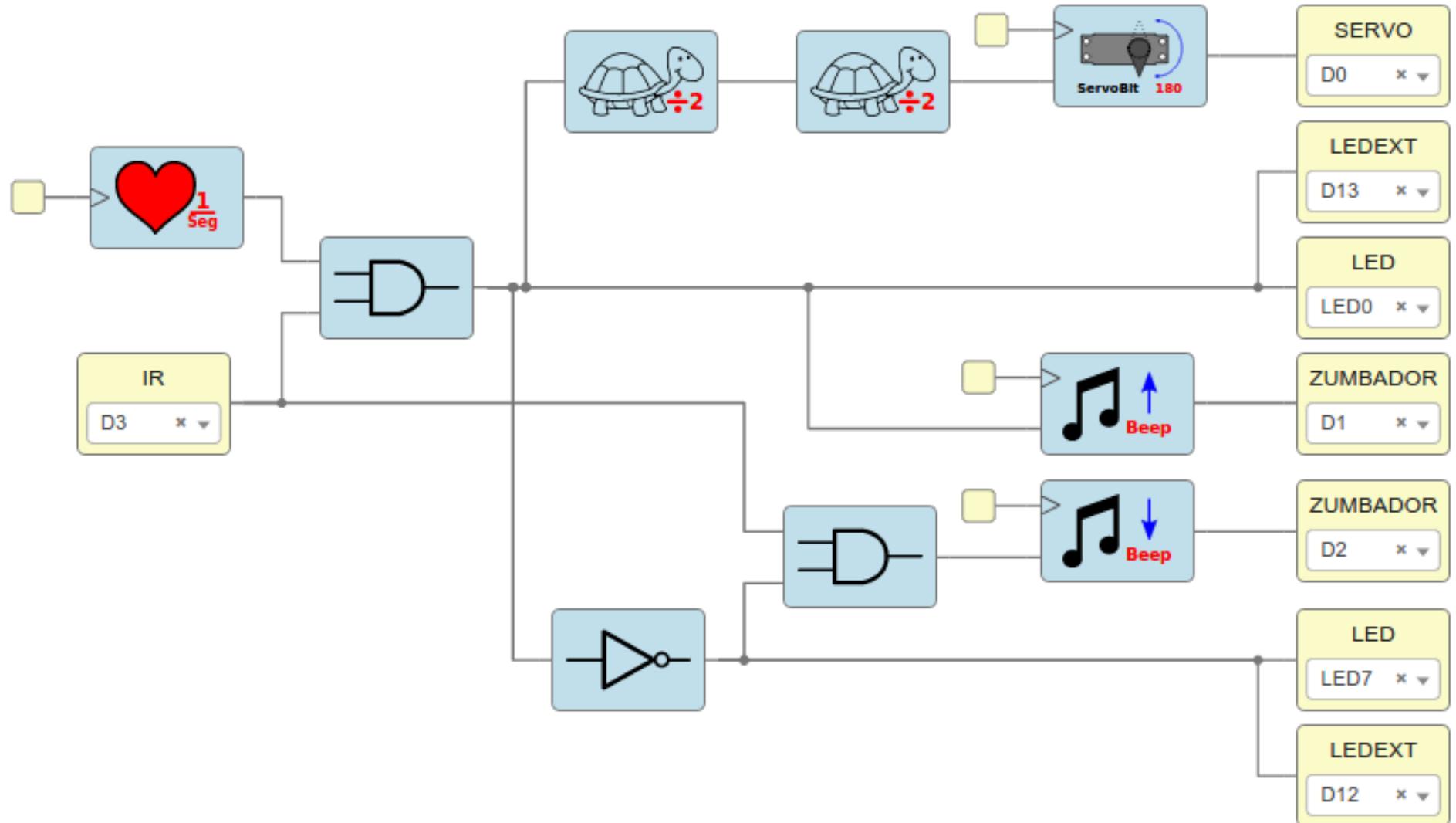
# Ejemplo 10: Sirena



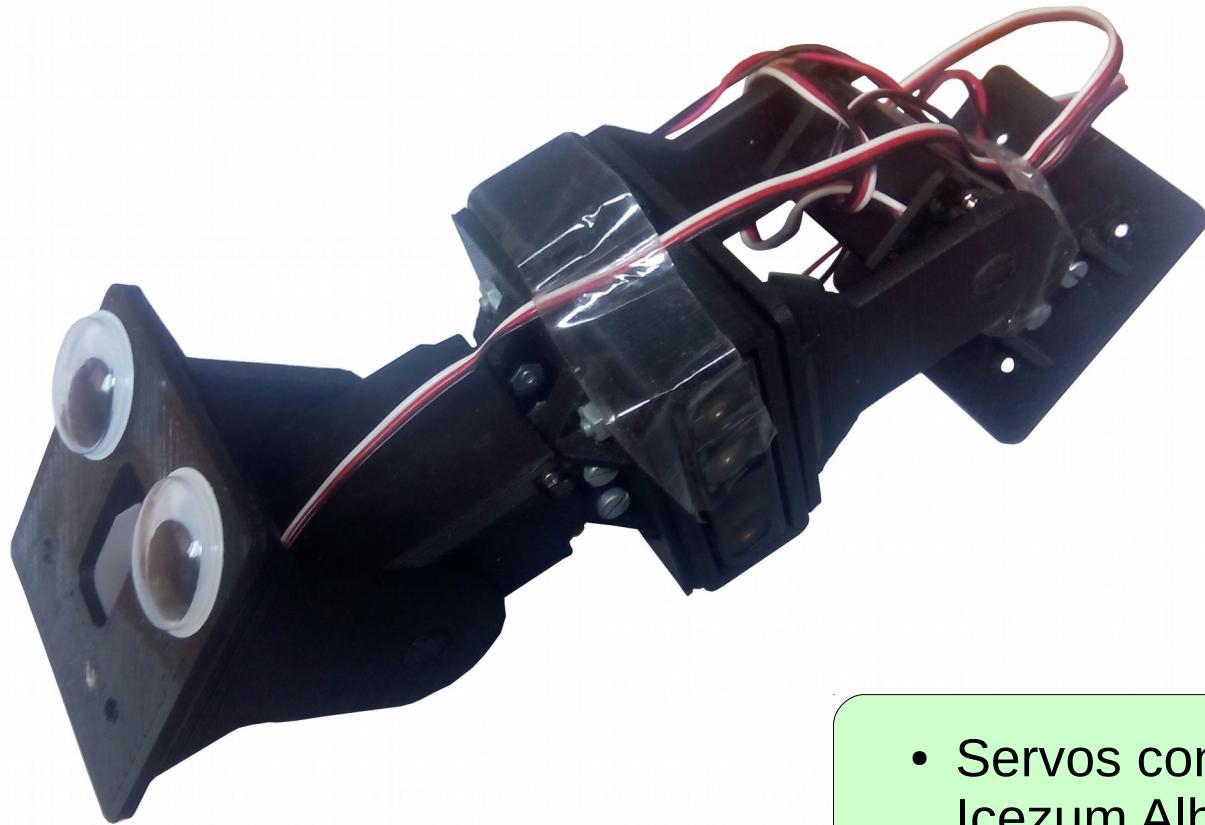
# Ejemplo 11: Test IR



# Ejemplo 12: Alarma v1.0



# Larby: Robot modular



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

# Lattuino

[https://github.com/INTI-CMNB/Lattuino\\_IP\\_Core](https://github.com/INTI-CMNB/Lattuino_IP_Core)



Lattuino\_Counter | Arduino 1.8.2

File Edit Sketch Tools Help

Lattuino\_Counter

```
// Lattuino Stick
// 4 bit counter

#define D1 14
#define D2 0
#define D3 1
#define D4 2
#define D5 3

#define DELAY 8

byte counter = 0;
int ledPin[] = {D4,D3,D2,D1};


```

Done uploading.

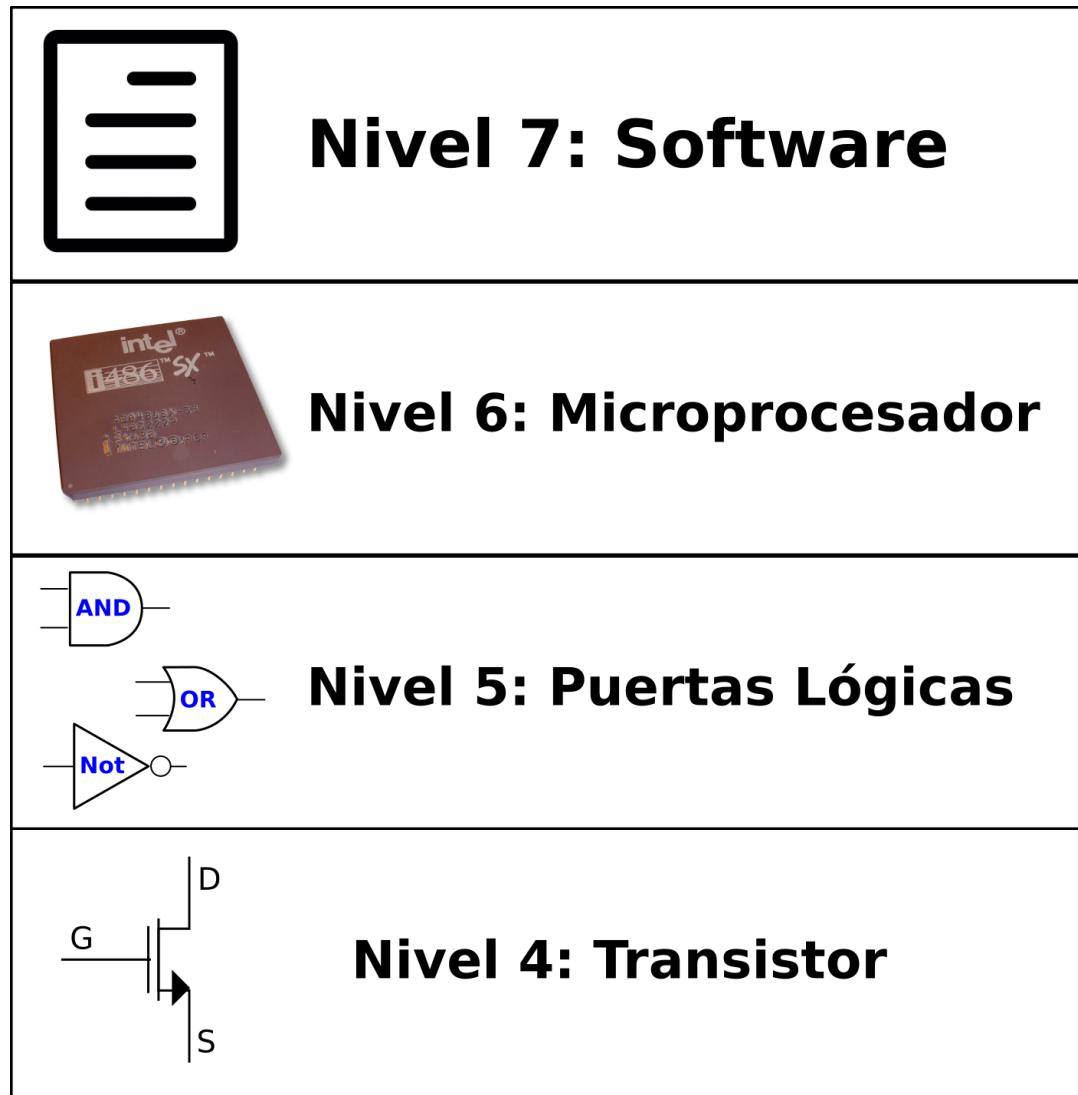
Sketch uses 496 bytes (35%) of program storage space. Maximum j  
Global variables use 10 bytes (7%) of dynamic memory, leaving I

28 Lattuino Stick (2k) on /dev/ttyUSB1

- Autor: **Salvador Tropea**
- Core de Arduino para FPGA
- Lattice Ice40 (1k, 4k, 8k)
- **VHDL**
- Herramientas privativas
- Migrando a **herramientas libres**



# Lattuino (II)

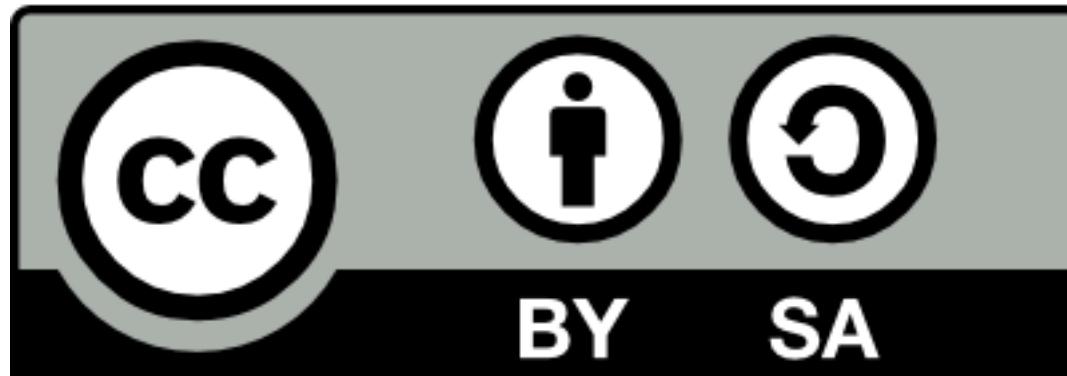
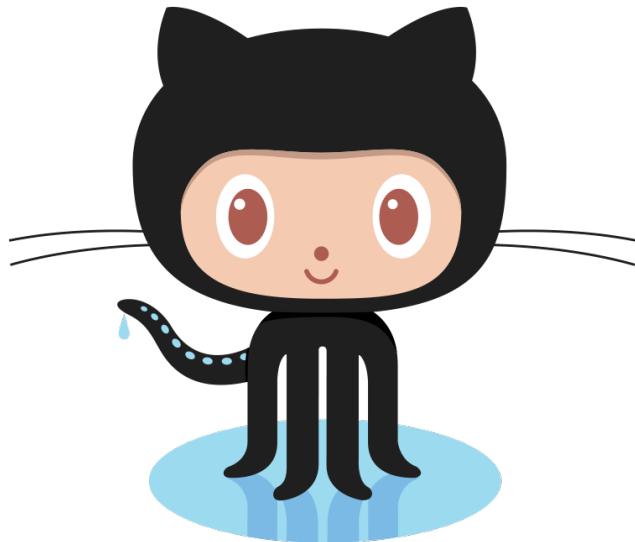


*Lattuino*



*FPGAs*

# ¡Comparte con la comunidad!



# ¡Que las FPGAs libres os acompañen!

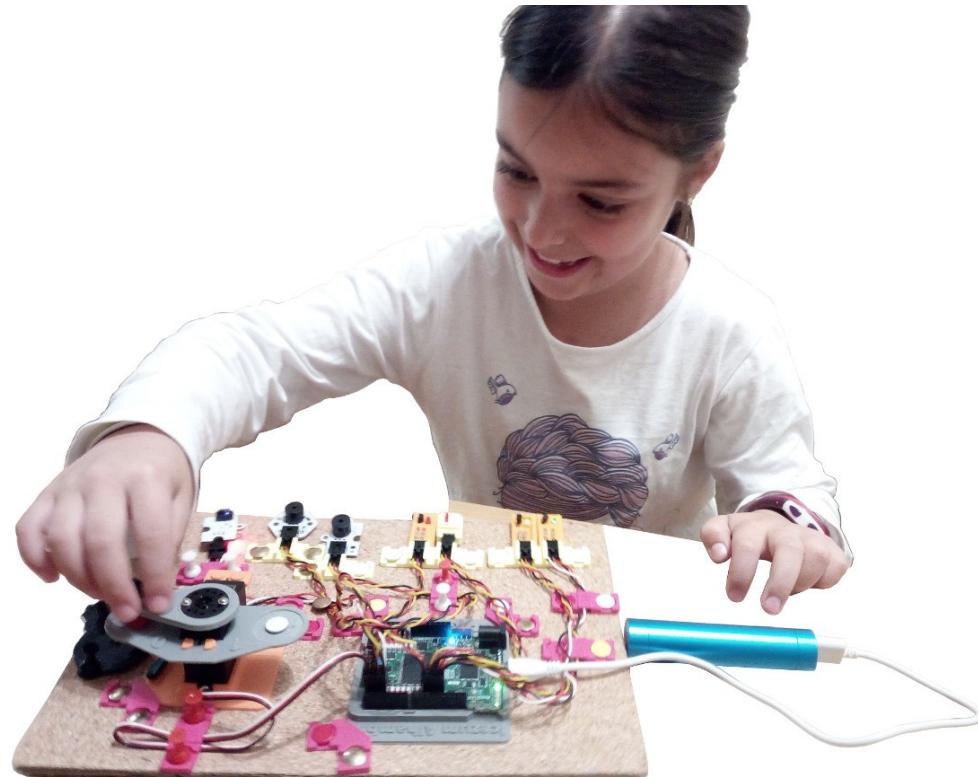


# Electrónica digital para todos con FPGAs Libres

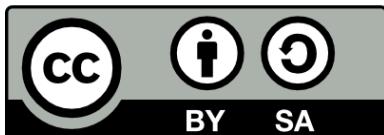


AYUNTAMIENTO DE  
CAMARGO

IBEROBOTICS  
ROBOTS PERSONALES Y DE SERVICIOS



Juan González Gómez (Obijuan)  
<https://github.com/Obijuan>



Curso de Verano UC: Robótica, Arduino y Hardware Libre

Centro Municipal de Empresas. Pol. Industrial de Trascueto, s/n. Revilla de Camargo

[https://github.com/Obijuan/myslides/wiki/2017\\_07\\_06:-UC:-Electrónica-digital-para-todos-con-FPGAs-libres](https://github.com/Obijuan/myslides/wiki/2017_07_06:-UC:-Electrónica-digital-para-todos-con-FPGAs-libres)