Workshop Arduino Bàsics

Jorge Pérez

@akirasan

Agenda:

Horario: 19:00 a 20:30

Martes 7 noviembre
Martes 14 noviembre
Martes 21 noviembre
Martes 28 noviembre

Martes 7:

INTRODUCCIÓN
RETO 0. [redacted]

Martes 14:

INTRODUCCIÓN RETO 1
RETO 1. [redacted]

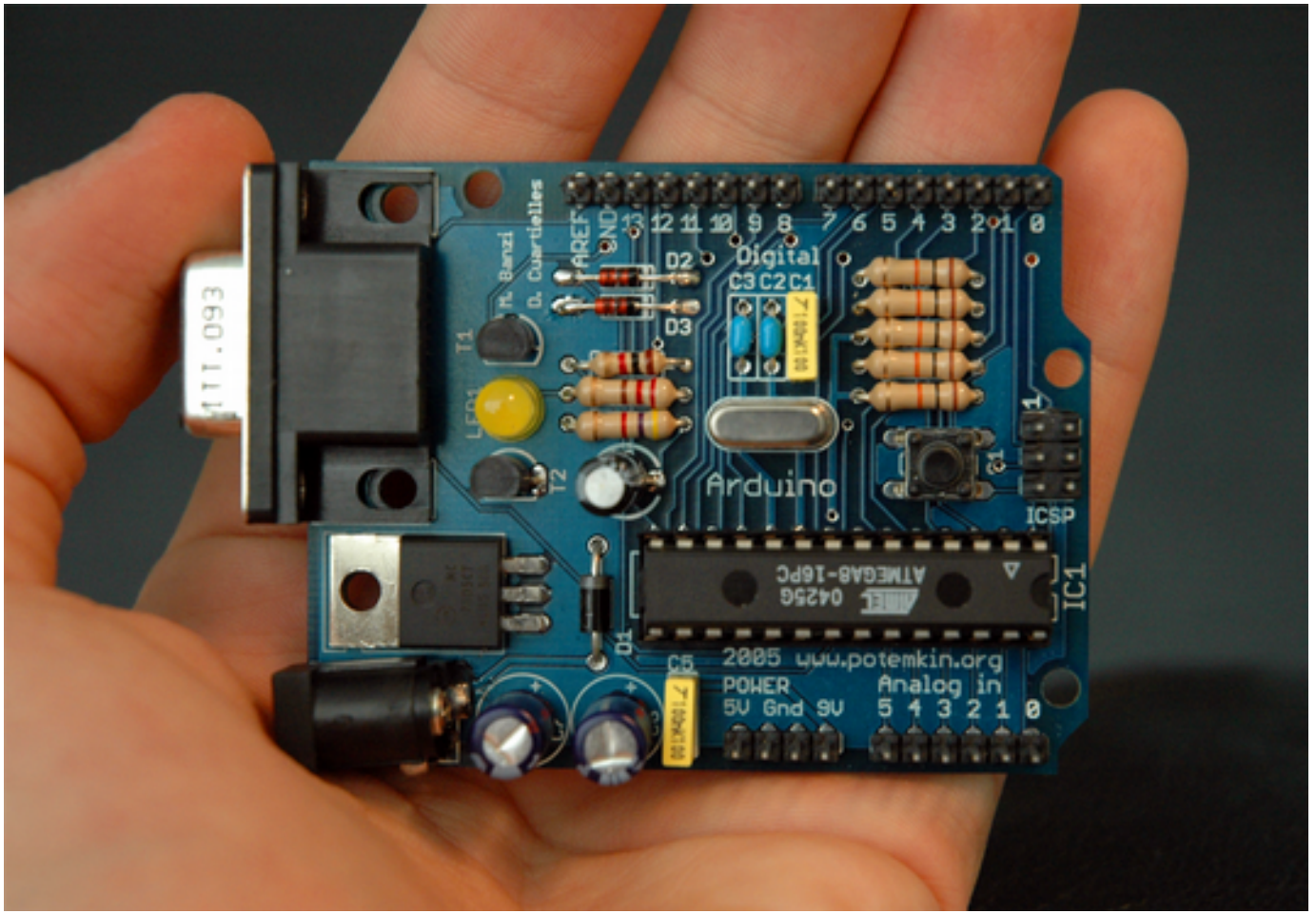
Martes 21:

FIN RETO 1
INTRODURCCIÓN RETO 2. [redacted]

Martes 28:

FIN RETO 2

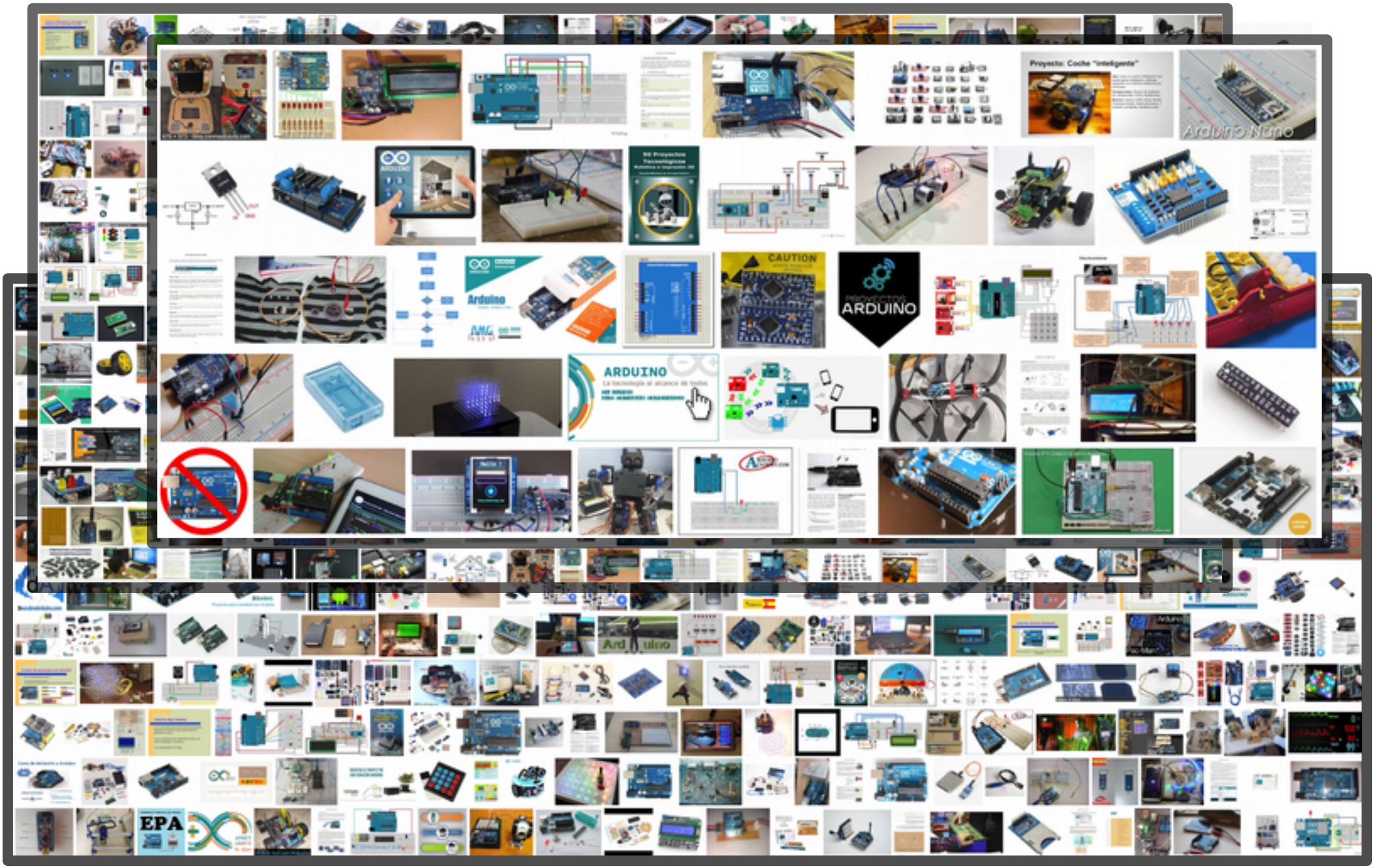
¿Qué es Arduino?: Hardware + Software



open source

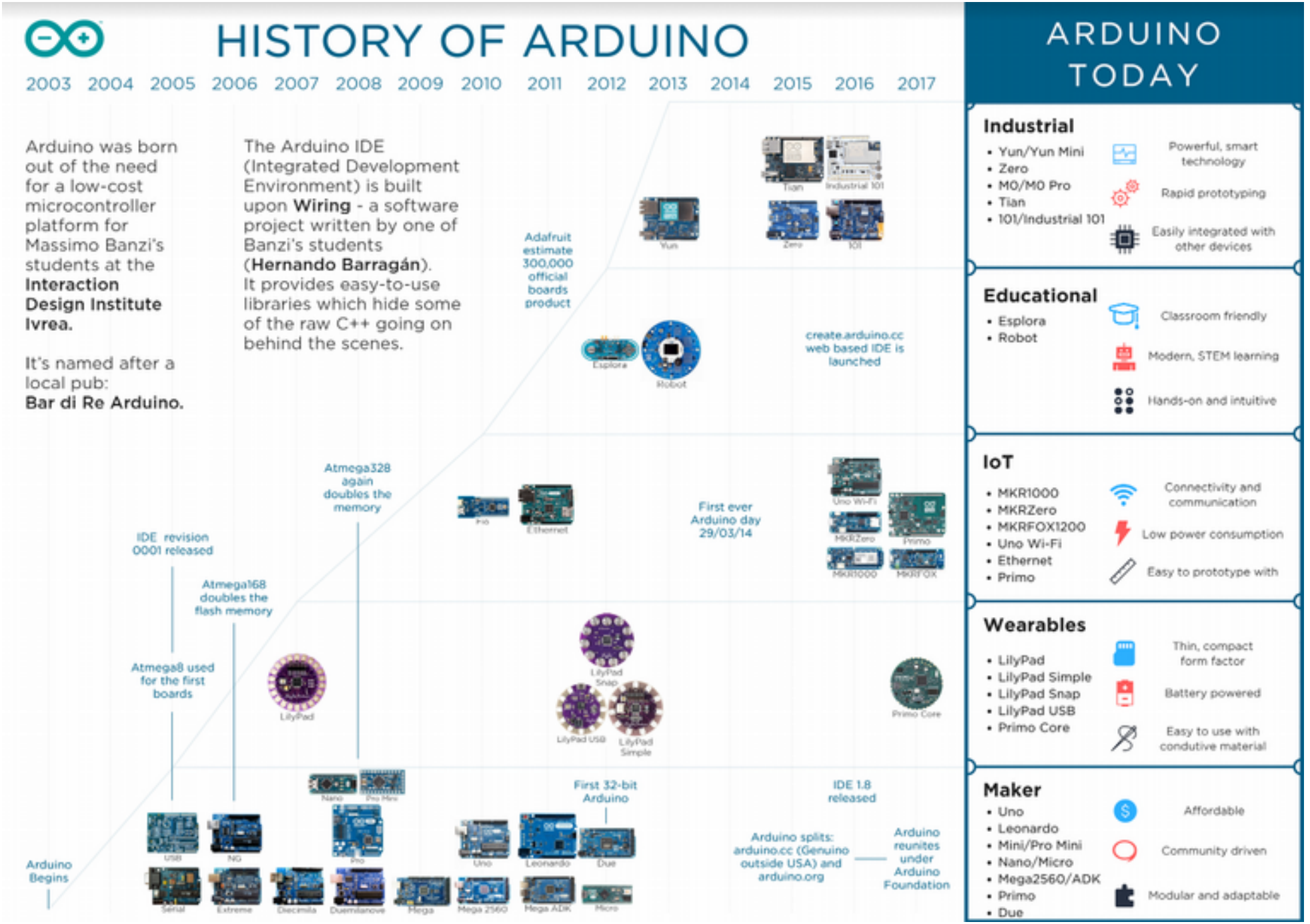
open source hardware

¿Qué es Arduino?: Google



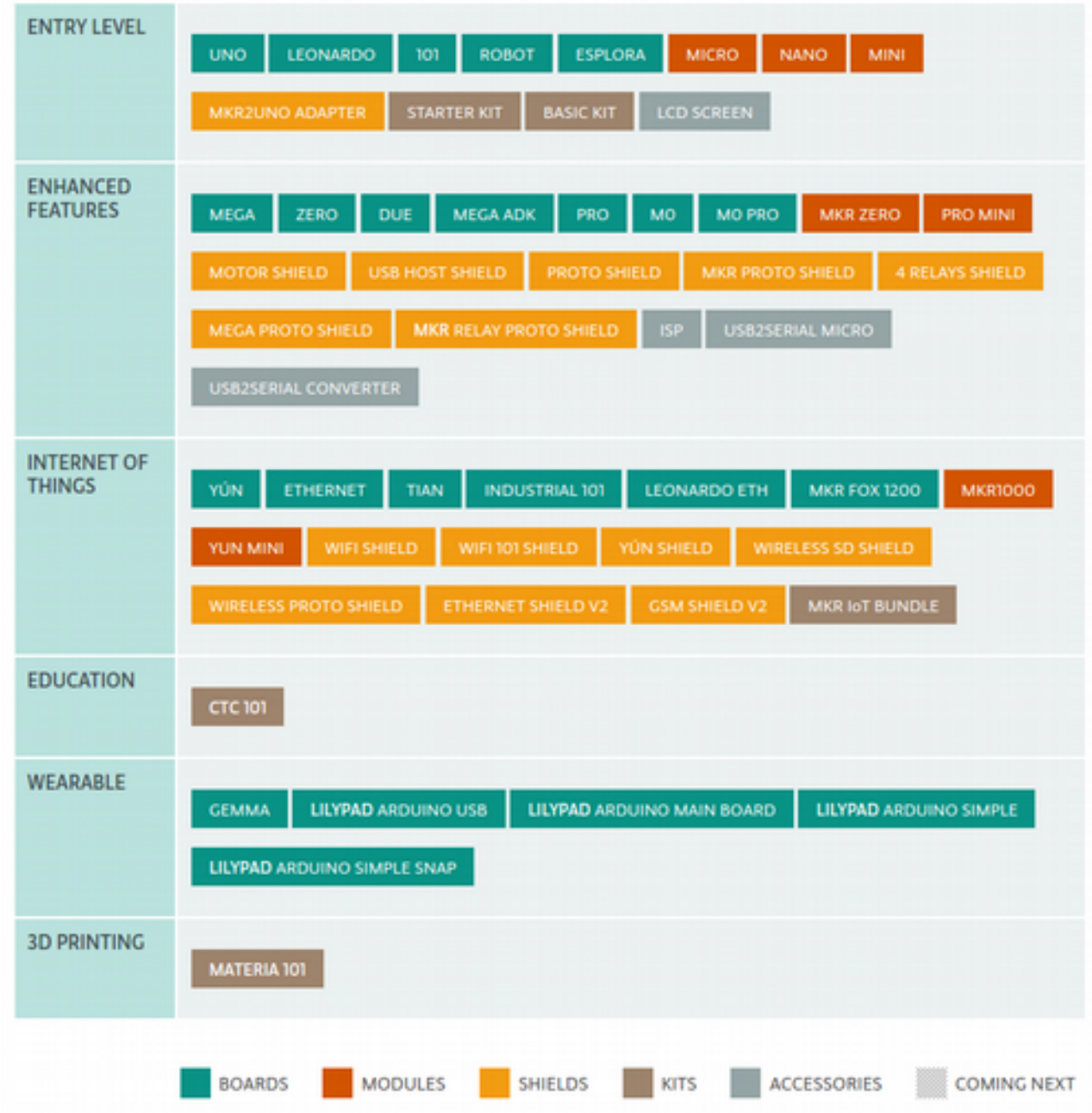
LOS ORIGENES: Los padres de la criatura

LOS ORIGENES: Evolución constante

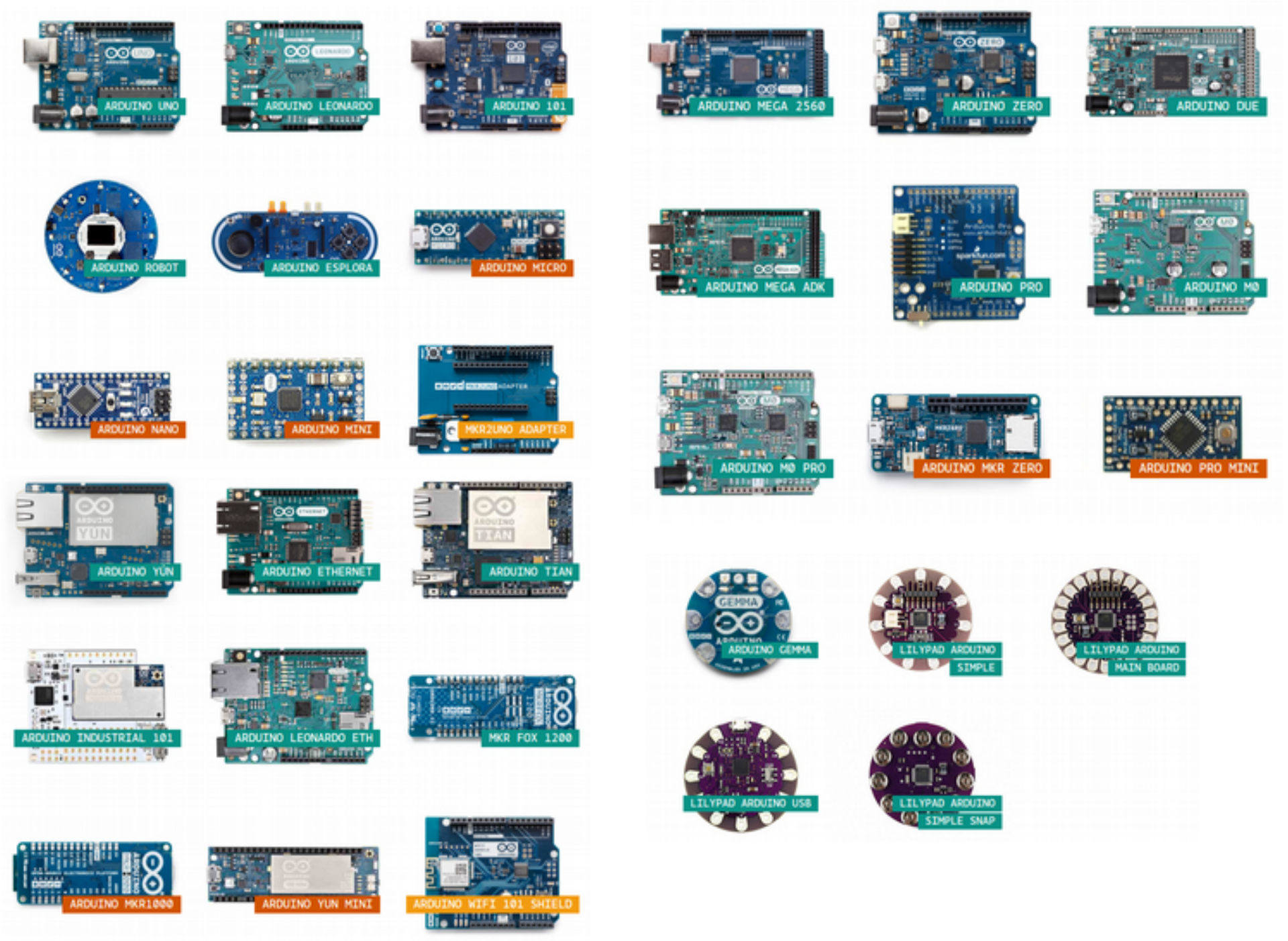


https://core-electronics.com.au/tutorials/history-of-arduino.html

La Familia: Sabores para todos



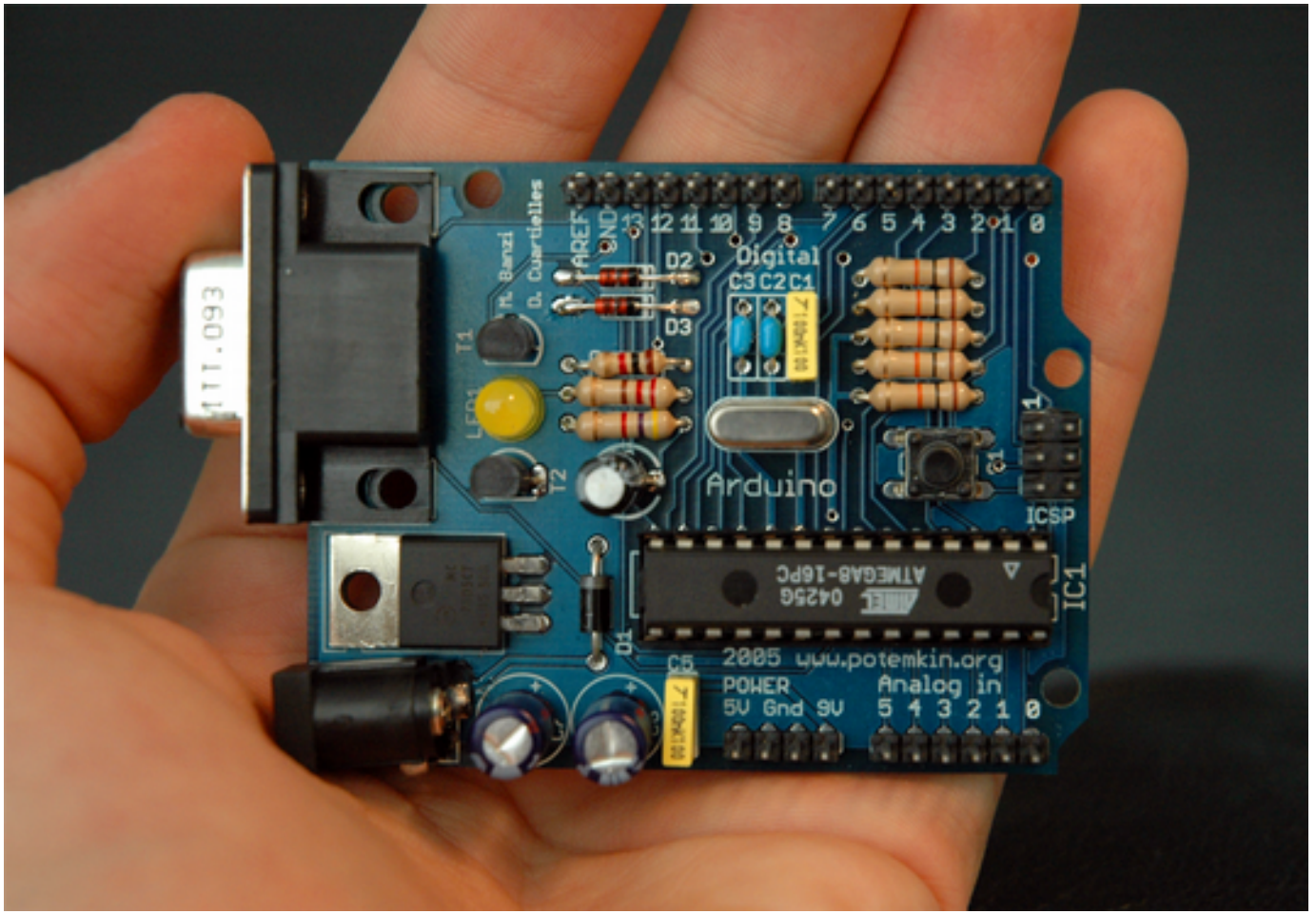
La Familia: Sabores para todos



LOS ORIGENES: Arduino The Documentary

https://vimeo.com/18390711

Hardware/Software: Vayamos por partes

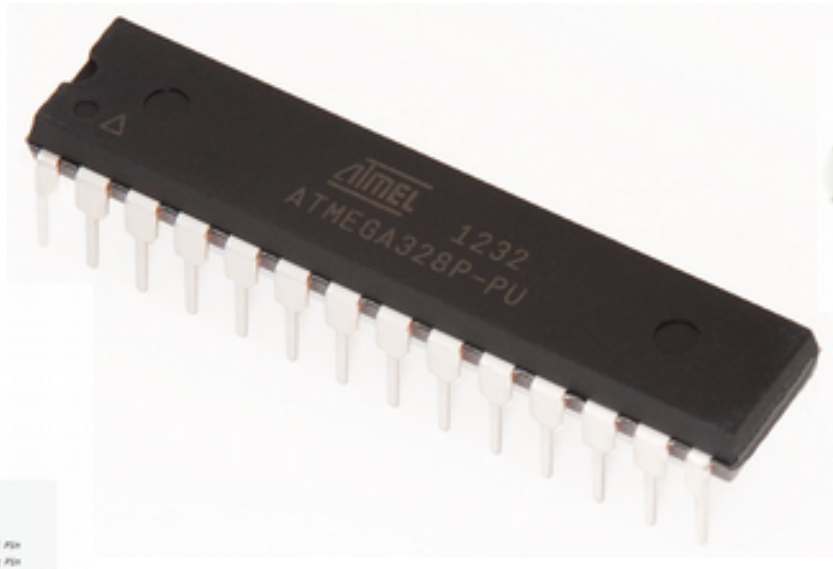


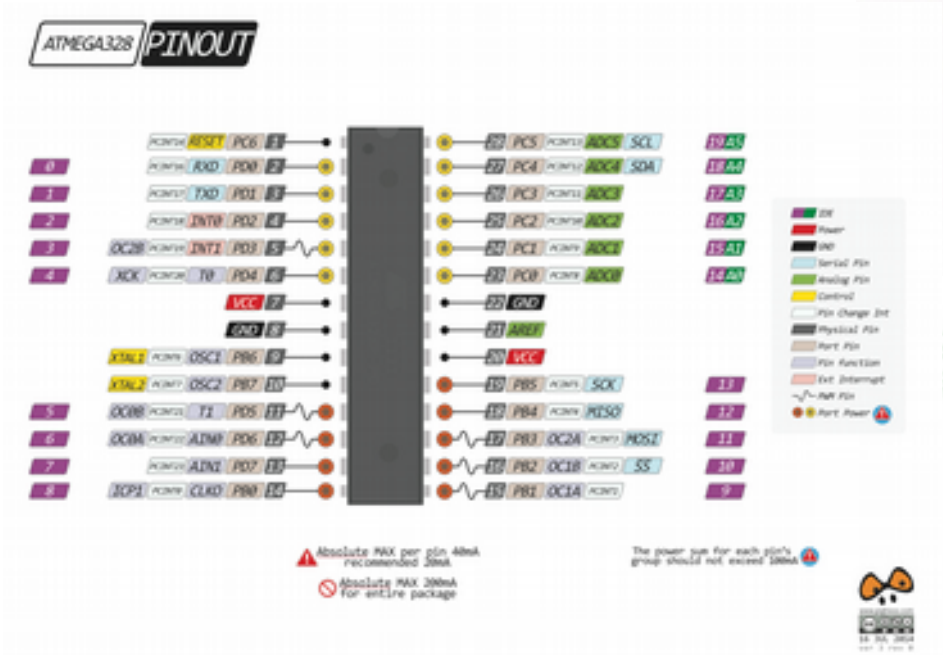
open source

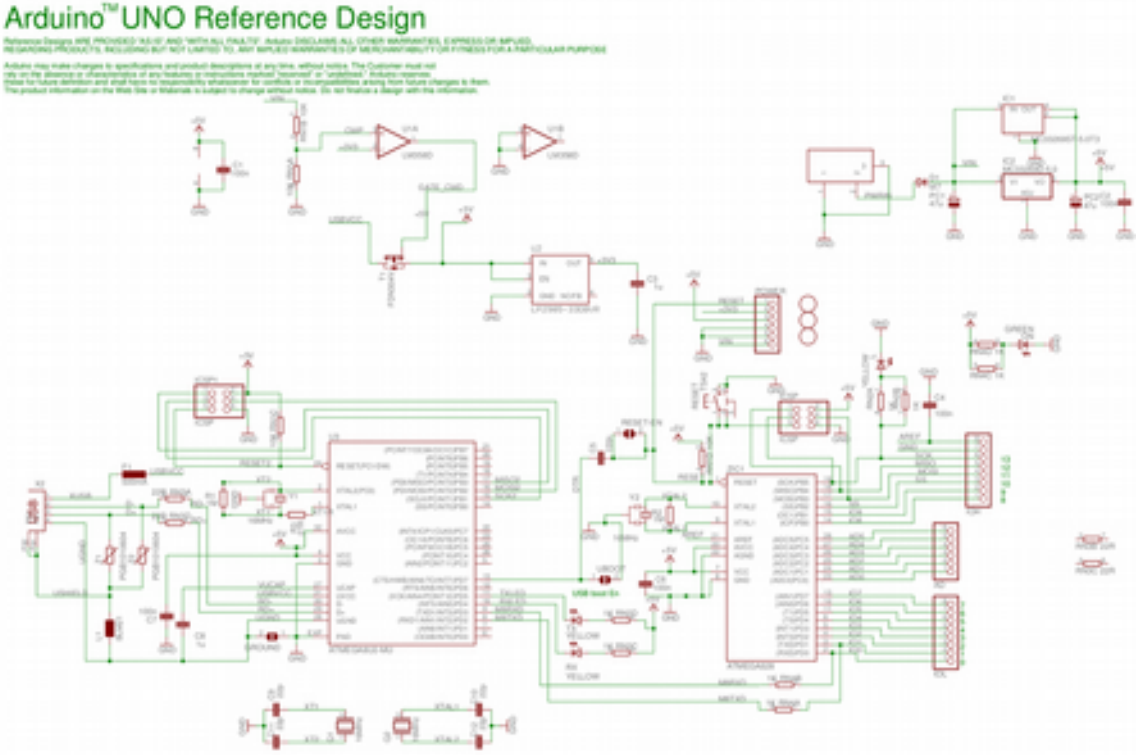
open source hardware

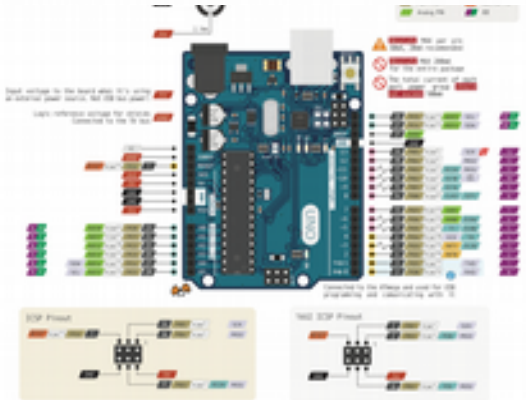
Hardware

Hardware: El microprocesador y algo mas...

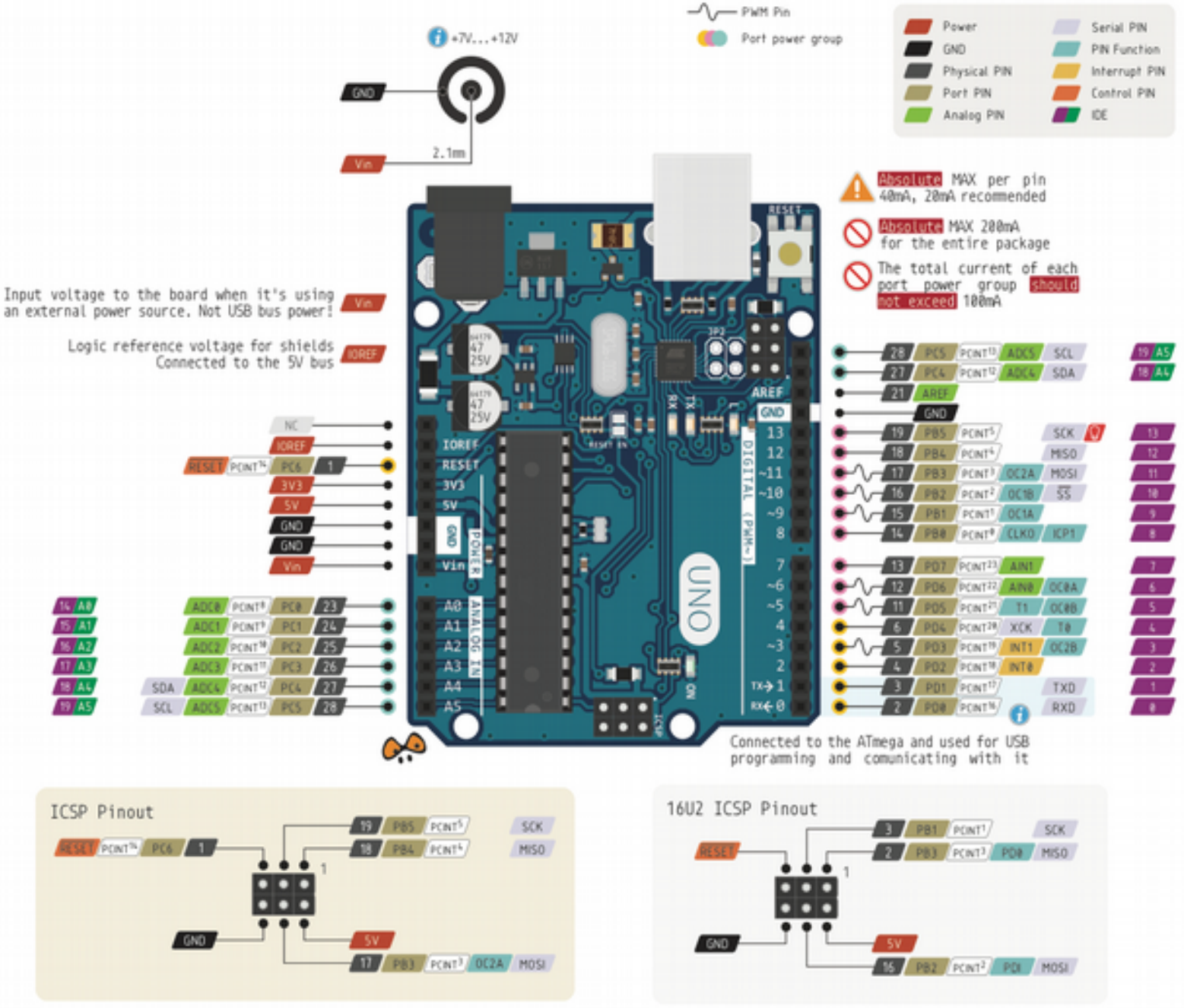








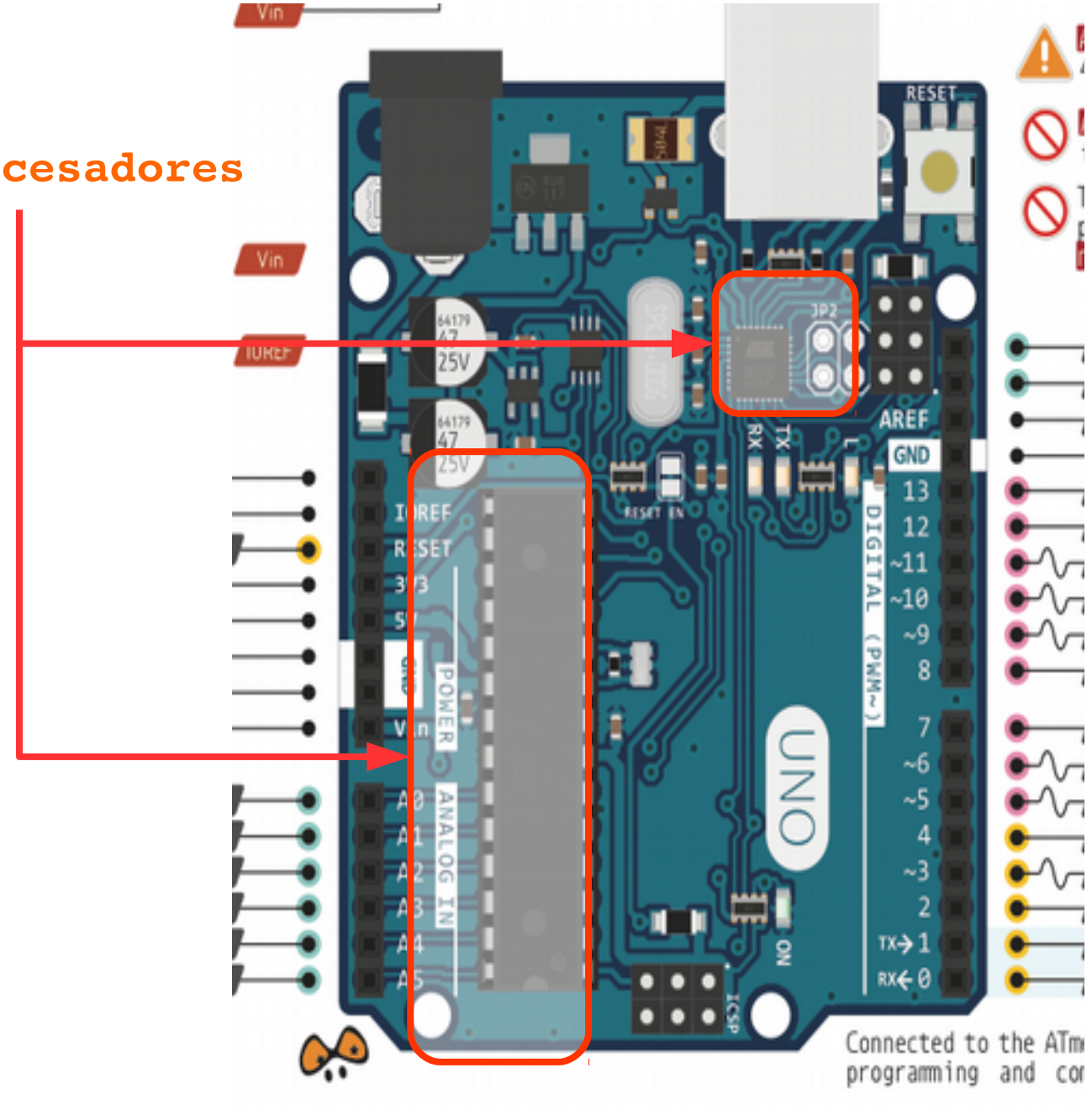
Hardware: El microprocesador y algo mas...



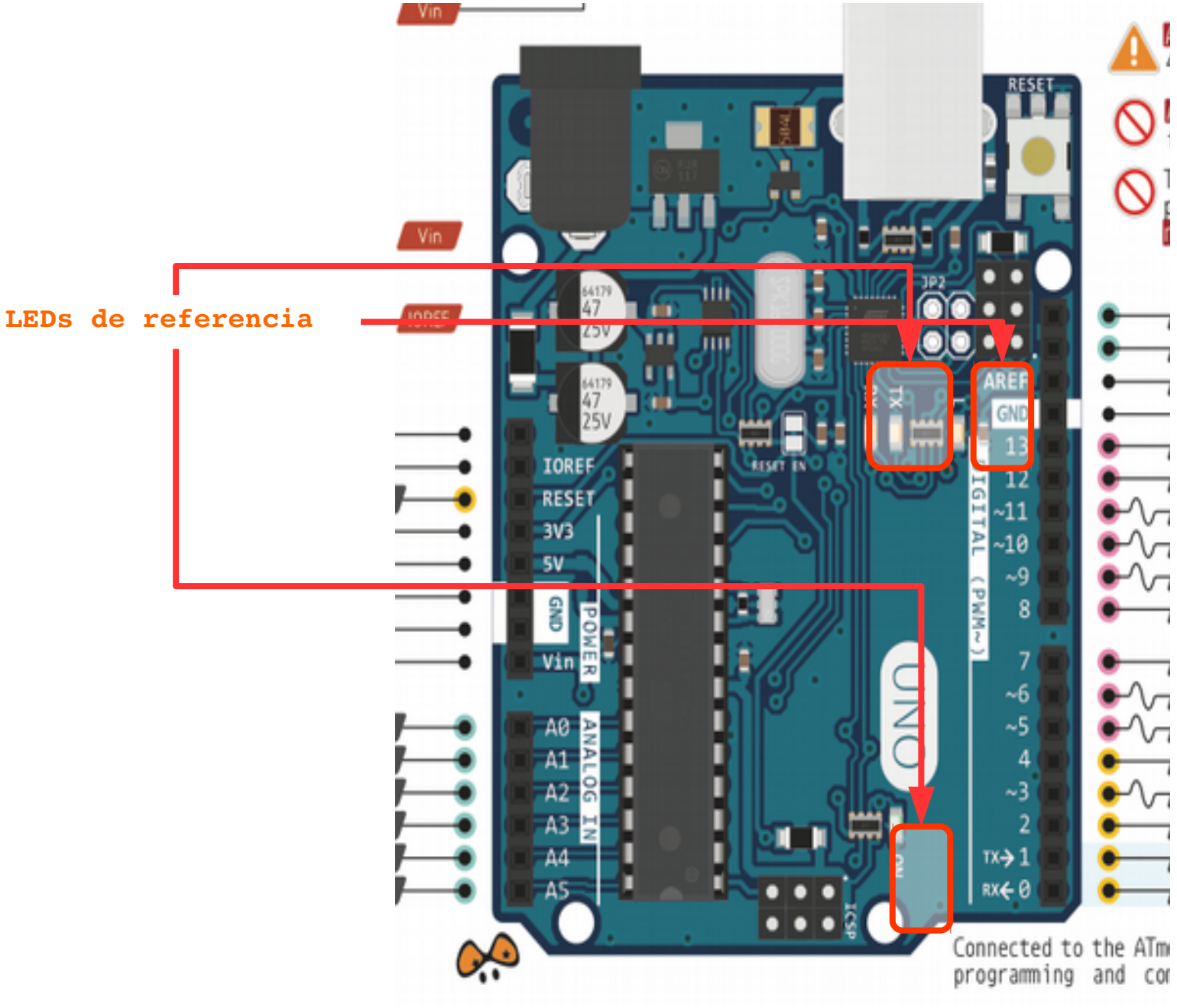
http://www.pighixxx.com

Hardware: El microprocesador y algo mas...

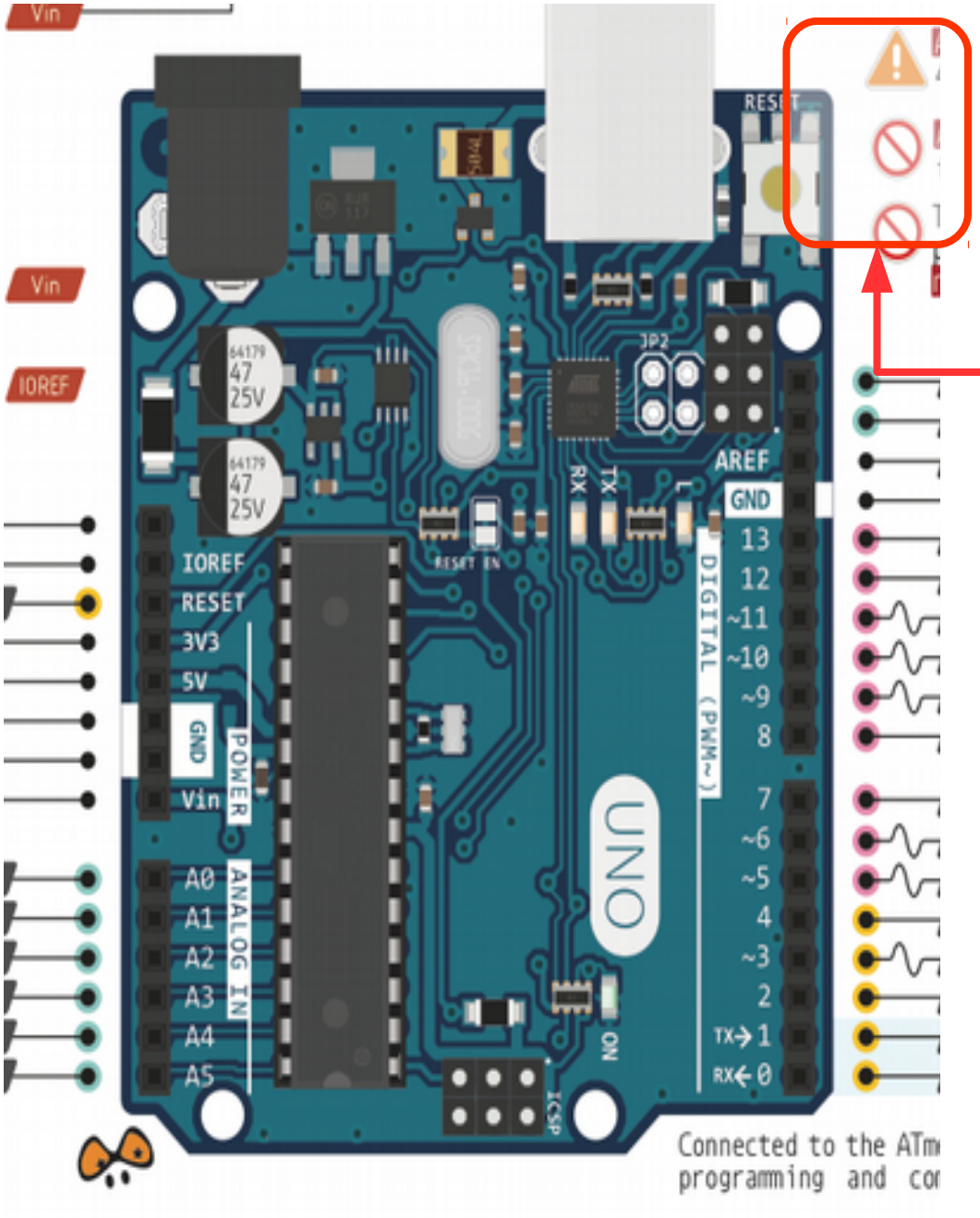
microprocesadores



Hardware: El microprocesador y algo mas...



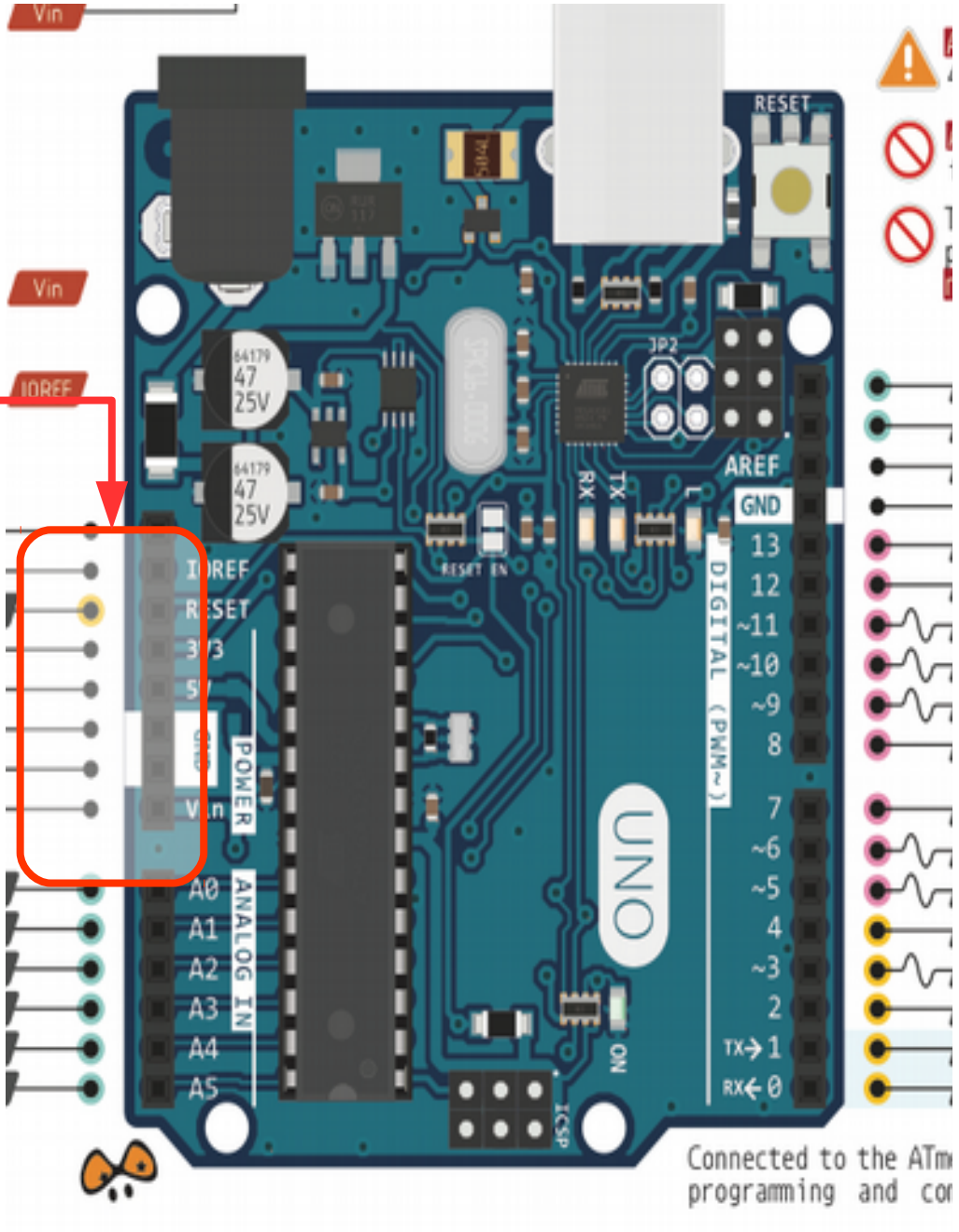
Hardware: El microprocesador y algo mas...



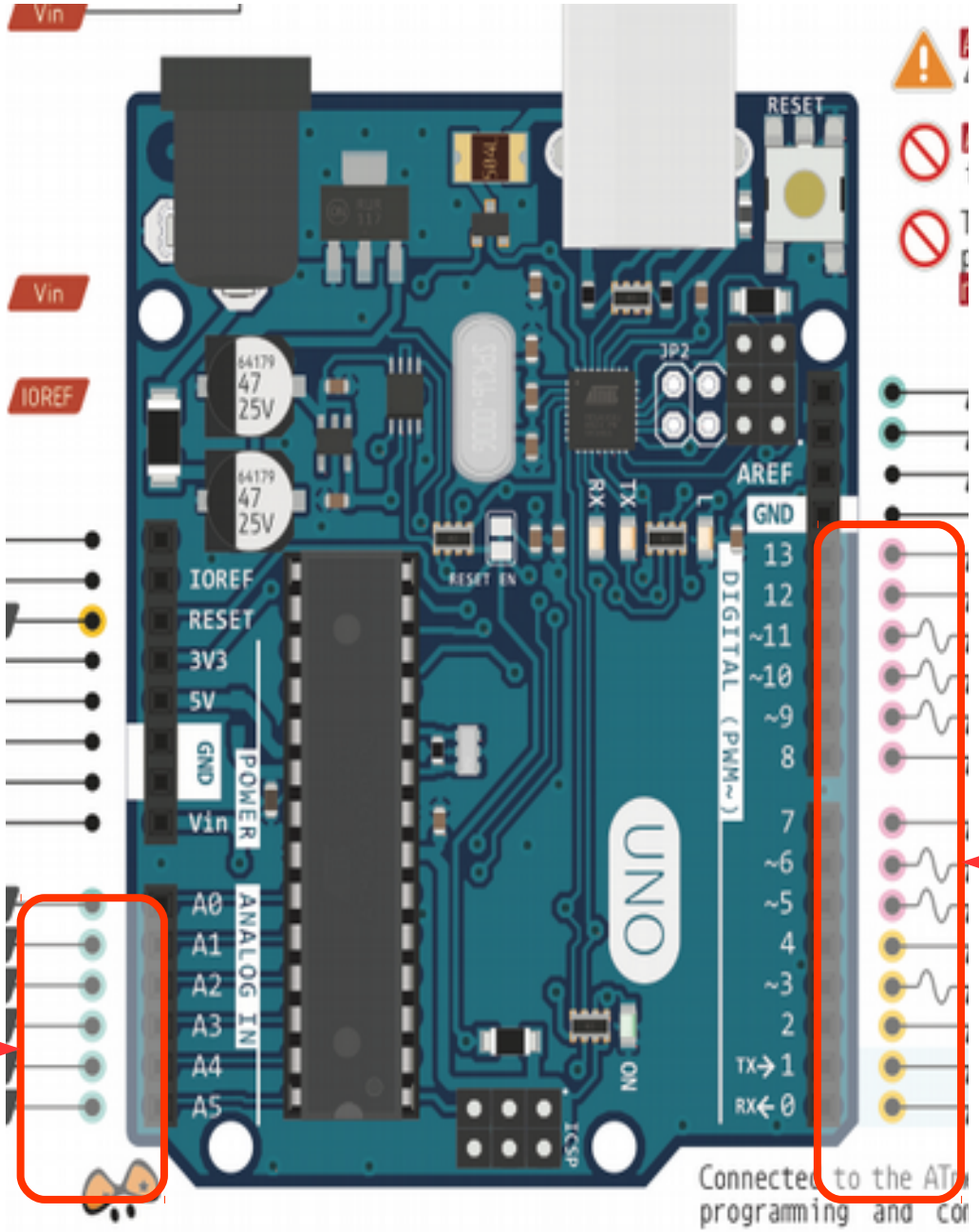
Botón de RESET

Hardware: El microprocesador y algo mas...

Pines de voltaje



Hardware: El microprocesador y algo mas...

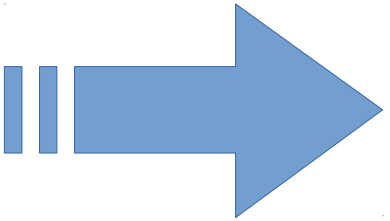


14 pines digitales:
8 digital
6 PWM ~ (salida)

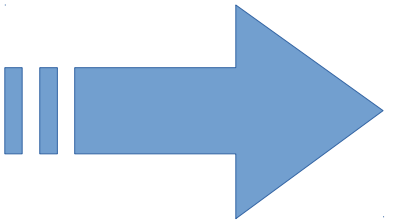
6 pines analógicos

Software

Software: Antepasados comunes del lenguaje



https://processing.org/



http://wiring.org.co/

Software: IDE. El entorno para humanos

IDE: Integrated Development Environment

LENGUAJE HUMANO. Estructura básica de UN PROGRAMA

```
Librerías

Varibles Globales
Constantes
Funciones propias
```

```
void setup() {
   // código de configuración
}
```

```
void loop() {
   // código que ejecuta
    // “para siempre”
}
```

COMPILADOR. El traductor: HUMANO vs MÁQUINA

compilador

Archivo Editar Programa Herramientas Ayuda

ejemplo_hola_mundo_led §

```
/*

  Ripolab Hacklab 2017

  Ejemplo HOLA MUNDO --- LED

*/

int PinLED = 13;

void setup() {
  pinMode(PinLED, OUTPUT);
}


void loop() {

  // LED activado
  digitalWrite(PinLED, HIGH);
  delay(1000);

  // LED apagado
  digitalWrite(PinLED, LOW);
  delay(1000);

}
```

ejemplo_hola_mundo_led.ino.hex

```
:100000000C945C000C946E000C946E000C946E00CA
:100010000C946E000C946E000C946E000C946E00A8
:100020000C946E000C946E000C946E000C946E0098
:100030000C946E000C946E000C946E000C946E0088
:10004000C419010C946E000C946E000C946E00CC
:100050000C946E000C946E000C946E000C946E0068
:100060000C946E000C946E0000000080002010069
:1000700000030407000000000000000000000072
:1000800025002800 2B0000000002400270 02A0083
:1000900004040404040404020202020203032E
:1000A000030303030102040810204080010204 0836
:1000B000102001020408102011241FBECFEFD8E049
:1000C000DEBFCDBF21E0A0E0B1E001C01D92A930AC
:1000D000B207E1F70E9463010C94D4010C94000074
:1000E00090E0FC01E859FF4F3491FC01EC55FF4FC3
:1000F0002491FC01E057FF4FE491EE2309F43BC04B
:10010000332339F1333091F038F43130A9F0323003
:1001100001F584B58F7D12C0373091F03830A1F0F1
:10012000 3430B9F4809180008F7D03C080918000CD
:10013000 8F77809380000DC084B58F7784BD09C010
:1001400008091B0008F7703C08091B0008F7D809345
:10015000B000F0E0EE0FFF1FE458FF4FA591B491FF
```

HUMANO vs MÁQUINA

Language Reference

Arduino programs can be divided in three main parts: *structure*, *values* (variables and constants), and *functions*.

Structure

- setup()
- loop()

Control Structures

- if
- if...else
- for
- switch case
- while
- do... while
- break
- continue
- return
- goto

Further Syntax

- ; (semicolon)
- {} (curly braces)
- // (single line comment)
- /* */ (multi-line comment)
- #define
- #include

Arithmetic Operators

- = (assignment operator)
- + (addition)
- - (subtraction)
- * (multiplication)
- / (division)
- % (modulo)

Comparison Operators

- == (equal to)
- != (not equal to)
- < (less than)
- > (greater than)
- <= (less than or equal to)
- >= (greater than or equal to)

Boolean Operators

- && (and)
- || (or)
- ! (not)

Pointer Access Operators

- * dereference operator
- & reference operator

Variables

Constants

- HIGH | LOW
- INPUT | OUTPUT | INPUT_PULLUP
- LED_BUILTIN
- true | false
- integer constants
- floating point constants

Data Types

- void
- boolean
- char
- unsigned char
- byte
- int
- unsigned int
- word
- long
- unsigned long
- short
- float
- double
- string - char array
- String - object
- array

Conversion

- char()
- byte()
- int()
- word()
- long()
- float()

Variable Scope & Qualifiers

- variable scope
- static
- volatile
- const

Utilities

- sizeof()
- PROGMEM

Functions

Digital I/O

- pinMode()
- digitalWrite()
- digitalRead()

Analog I/O

- analogReference()
- analogRead()
- analogWrite() - *PWM*

Due & Zero only

- analogReadResolution()
- analogWriteResolution()

Advanced I/O

- tone()
- noTone()
- shiftOut()
- shiftIn()
- pulseIn()

Time

- millis()
- micros()
- delay()
- delayMicroseconds()

Math

- min()
- max()
- abs()
- constrain()
- map()
- pow()
- sqrt()

Trigonometry

- sin()
- cos()
- tan()

Characters

- isAlphaNumeric()
- isAlpha()
- isAscii()
- isWhitespace()
- isControl()
- isDigit()
- isGraph()
- isLowerCase()
- isPrintable()
- isPunct()
- isSpace()
- isUpperCase()
- isHexadecimalDigit()

Random Numbers

- randomSeed()
- random()

Bits and Bytes

- lowByte()
- highByte()
- bitRead()
- bitWrite()
- bitSet()
- bitClear()
- bit()

External Interrupts

- attachInterrupt()
- detachInterrupt()

Interrupts

- interrupts()
- noInterrupts()

Communication

- Serial
- Stream

USB (32u4 based boards and Due/Zero only)

- Keyboard
- Mouse

Looking for something else?

See the libraries page for interfacing with particular types of hardware. Try the list of community-contributed code. The

- == (equal to)
- != (not equal to)
- < (less than)
- > (greater than)
- <= (less than or equal to)
- >= (greater than or equal to)

Boolean Operators

- && (and)
- || (or)
- ! (not)

Pointer Access Operators

- * dereference operator
- & reference operator

Bitwise Operators

- & (bitwise and)
- | (bitwise or)
- ^ (bitwise xor)
- ~ (bitwise not)
- << (bitshift left)
- >> (bitshift right)

Compound Operators

- ++ (increment)
- -- (decrement)
- += (compound addition)
- -= (compound subtraction)
- *= (compound multiplication)
- /= (compound division)
- %= (compound modulo)
- &= (compound bitwise and)
- |= (compound bitwise or)

- long()
- float()

Variable Scope & Qualifiers

- variable scope
- static
- volatile
- const

Utilities

- sizeof()
- PROGMEM

- map()
- pow()
- sqrt()

Trigonometry

- sin()
- cos()
- tan()

Characters

- isAlphaNumeric()
- isAlpha()
- isAscii()
- isWhitespace()
- isControl()
- isDigit()
- isGraph()
- isLowerCase()
- isPrintable()
- isPunct()
- isSpace()
- isUpperCase()
- isHexadecimalDigit()

Random Numbers

- randomSeed()
- random()

Bits and Bytes

- lowByte()
- highByte()
- bitRead()
- bitWrite()
- bitSet()
- bitClear()
- bit()

External Interrupts

- attachInterrupt()
- detachInterrupt()

Interrupts

- interrupts()
- noInterrupts()

Communication

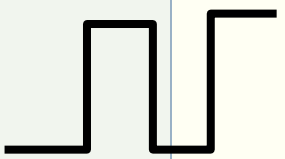
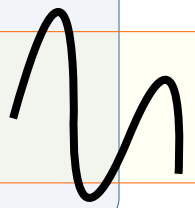
- Serial
- Stream

USB (32u4 based boards and Due/Zero only)

- Keyboard
- Mouse

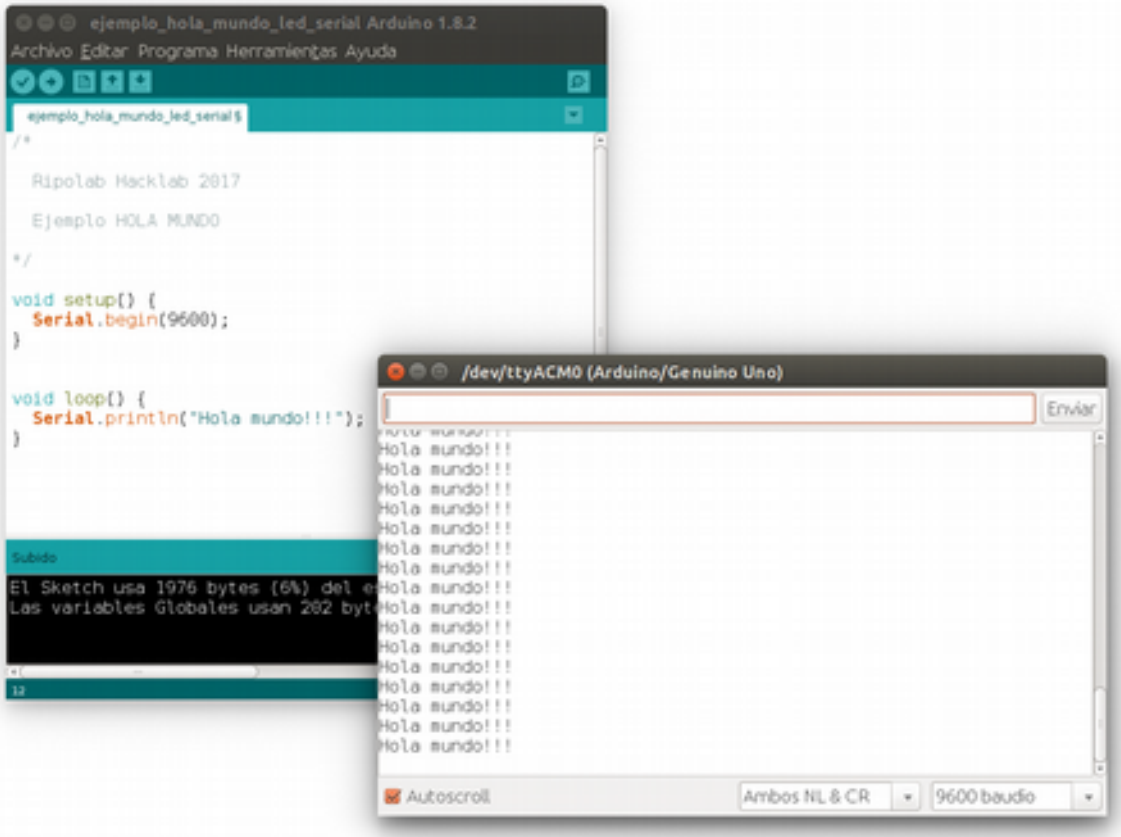
https://www.arduino.cc/en/Reference/HomePage

Lenguaje Básico. Hablar con las máquinas

ESCRIBIR / HABLAR	LEER / ESCUCHAR
`pinMode(#pin,OUTPUT)`	`pinMode(#pin,INPUT)`
`digitalWrite(#pin,HIGH)` `digitalWrite(#pin,LOW)`	`digitalRead(#pin)` 
`analogWrite(#pin,#valor)`	`analogRead(#pin)` 

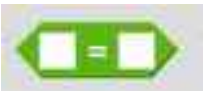
Lenguaje Básico. ¿Y la pantalla?

```
void setup() {
    Serial.begin(9600);
}

void loop() {
    Serial.println("Hola mundo!!!");
}
```



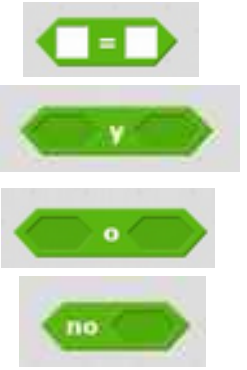
Lenguaje Básico. Repetir y repetir

```
for ([valor inicial]; [condición]; [incremento/decremento])
{
   // código que se repite
}
```



Lenguaje Básico. Tomar decisiones

```
if ([condición])
{
    // código que se ejecuta si
    // se cumple la condición_1
}
```



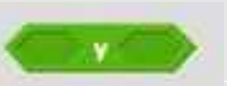
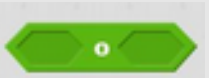

```
if ([condición_1])
{
    // código que se ejecuta si se
    // cumple la condición_1
}

else

{
    // código que se ejecuta si
    // NO se cumple la condición_1
}
```

Lenguaje Básico. Espera un momento...

```
delay(tiempo milisegundos);
```

```
delayMicroseconds(tiempo microisegundos);
```

Lenguaje Básico. Condiciones y booleanos

Igual	==	
No Igual	!=	
Menor que	<	
Mayor que	>	
Menor igual que	<=	
Mayor igual que	=>	
Y	&&	
O	||	
NO	!	

Al cacharreo!!!

#RETO 0