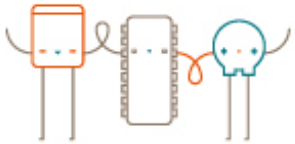


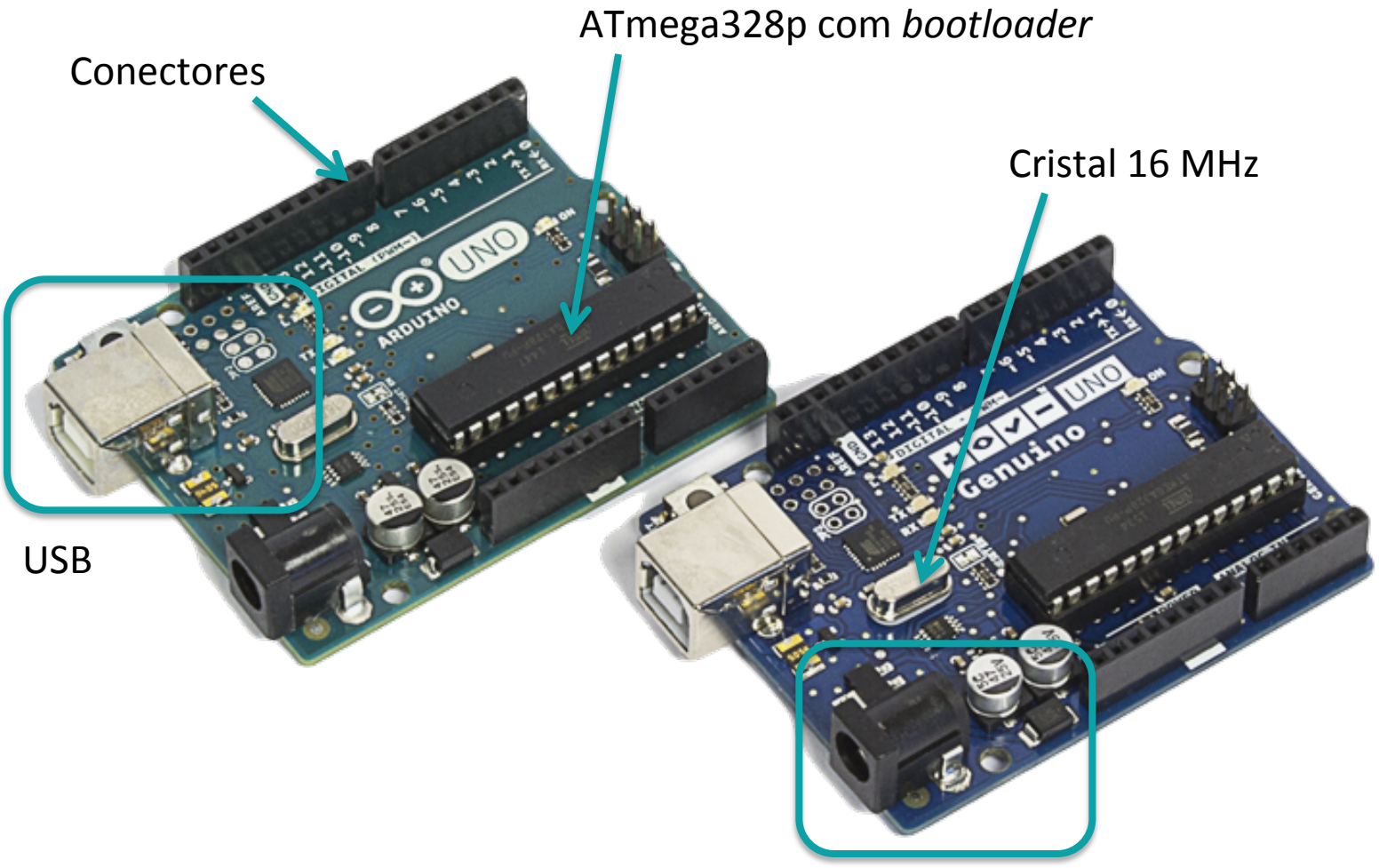
Genuino
DAY2016

Microcontroladores Avulso com o Arduino

Genuino
DAY2016

Carlos Rodrigues – Abril de 2016

Arduino UNO



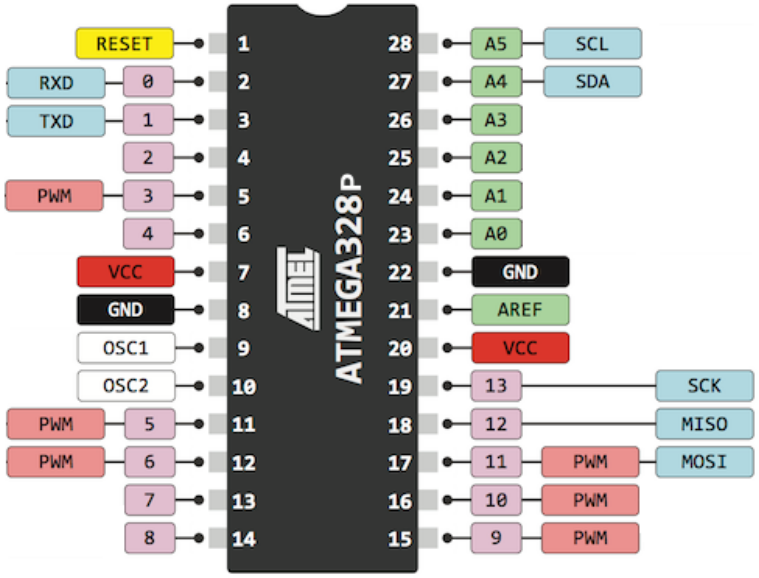
Genuino DAY2016

Microcontroladores ATmega

- Tem oscilador interno (1 ou 8 MHz)
- Requer alimentação regulada (3.3 ou 5V)
- Condensadores **recomendados** nos pinos de alimentação (para isolar ruído)

- Isto é **tudo** o que precisa para funcionar
- Consumo mais baixo do que a 16 MHz
- Sem *bootloader* arranca instantaneamente

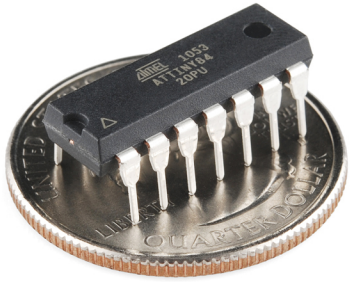
ATmega168
ATmega328p



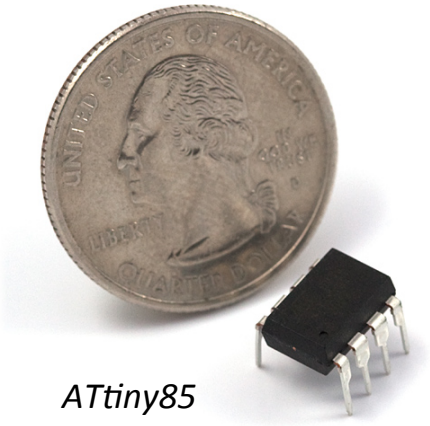
https://github.com/carlosefr/atmega

Genuino DAY2016

Microcontroladores ATtiny



ATtiny84



ATtiny85

- Têm osciladores internos (1 ou 8 MHz)
- Compatíveis com o Arduino IDE

- Não precisam de **nenhum** componente externo para funcionar (nem de alimentação regulada*)
- São mais **baratos** (1 ou 2€ cada)
- Consomem menos energia

- Têm menos memória Flash (8 KB)
- Têm menos memória SRAM (512 bytes)

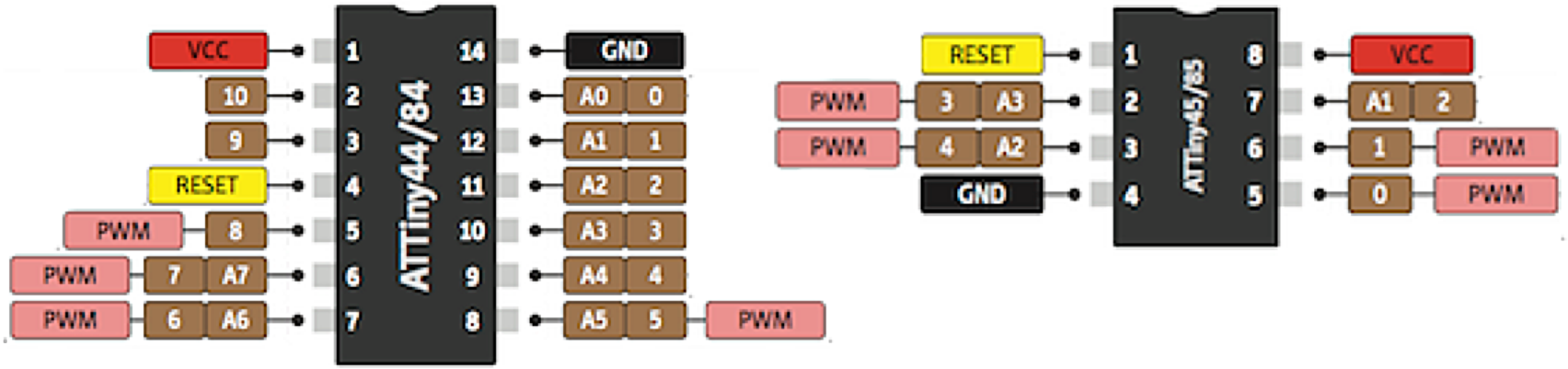
*Apenas em certas condições e **sempre** 5V ou menos!

http://highlowtech.org/?p=1695

Genuino DAY2016

Microcontroladores ATtiny

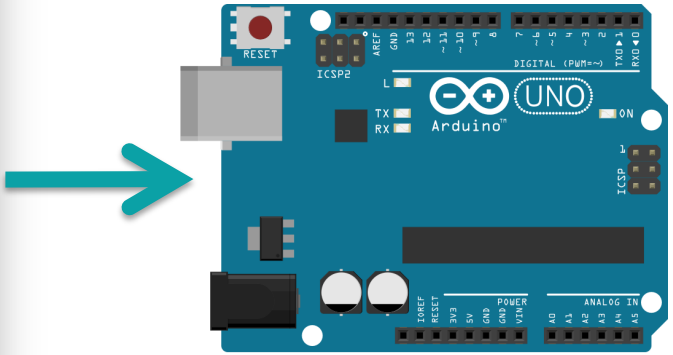
- Miniaturizar projectos mais simples
- Modularizar projectos mais complexos
- Alternativa para integração com outras plataformas (ex. Raspberry Pi)



http://highlowtech.org/?p=1695

Genuino DAY2016

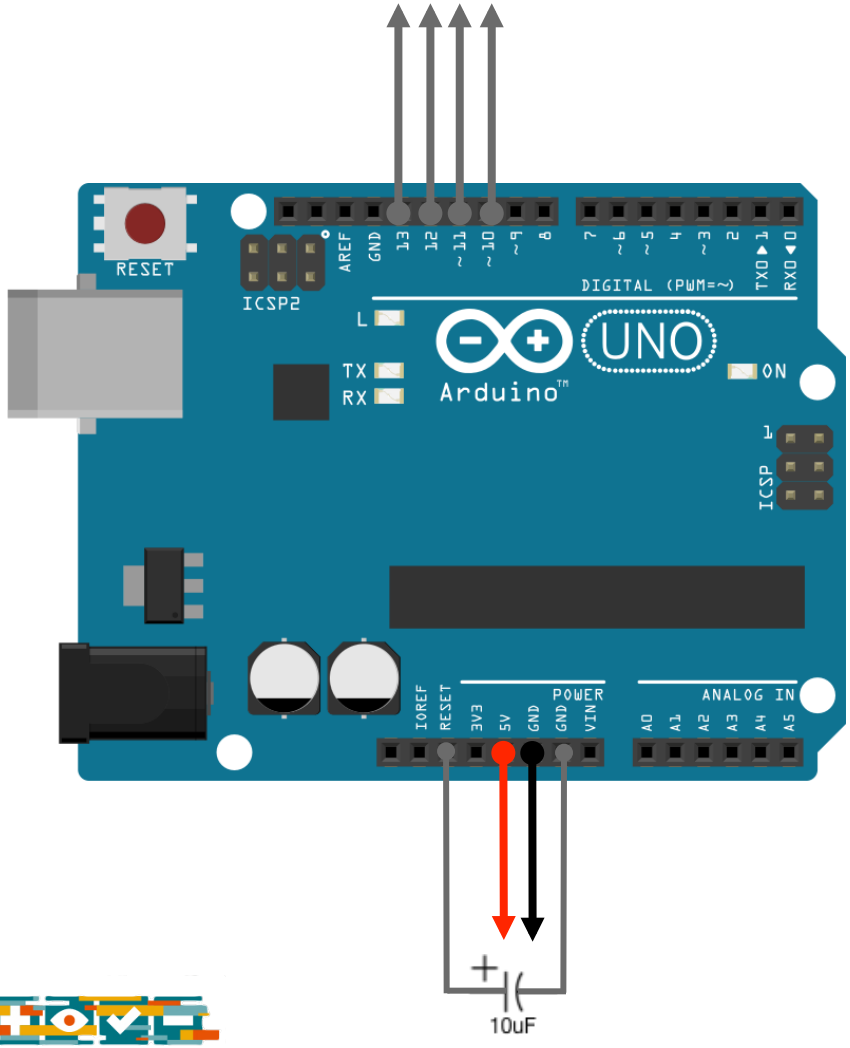
Arduino *as* ISP



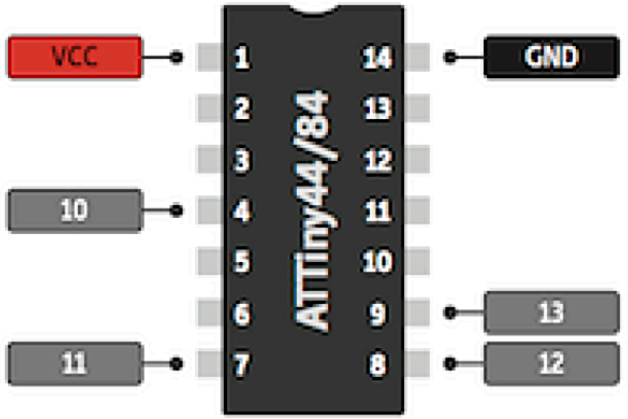
https://www.arduino.cc/en/Tutorial/ArduinoISP

Genuino DAY2016

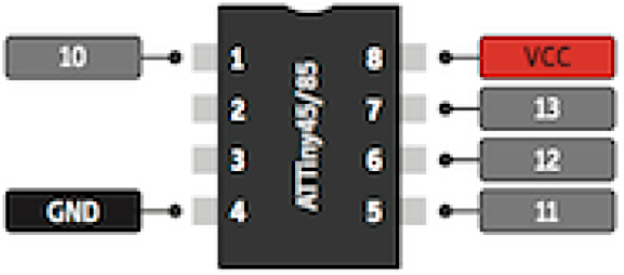
Arduino *as* ISP



ATtiny84



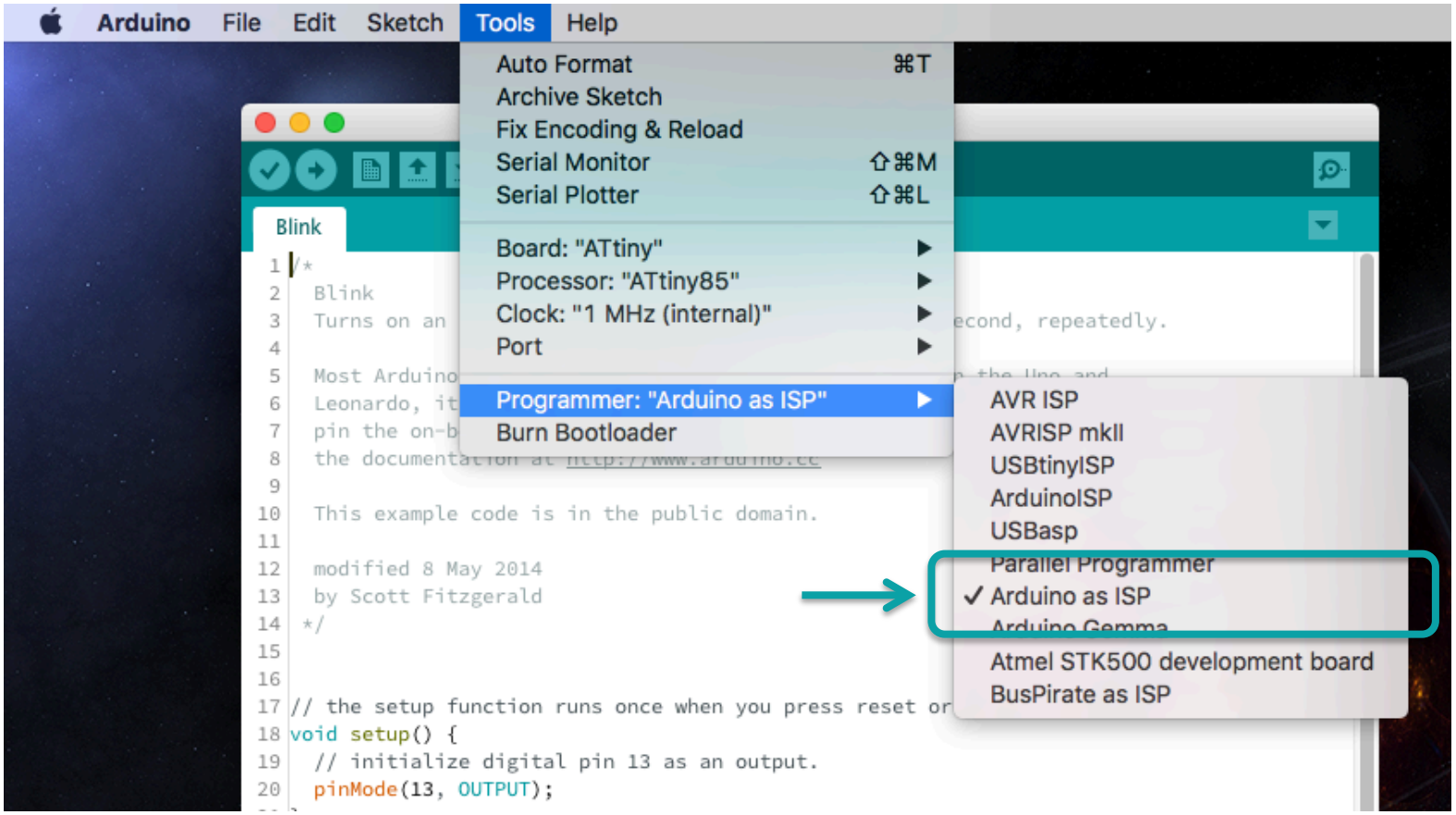
ATtiny85



https://www.arduino.cc/en/Tutorial/ArduinoISP

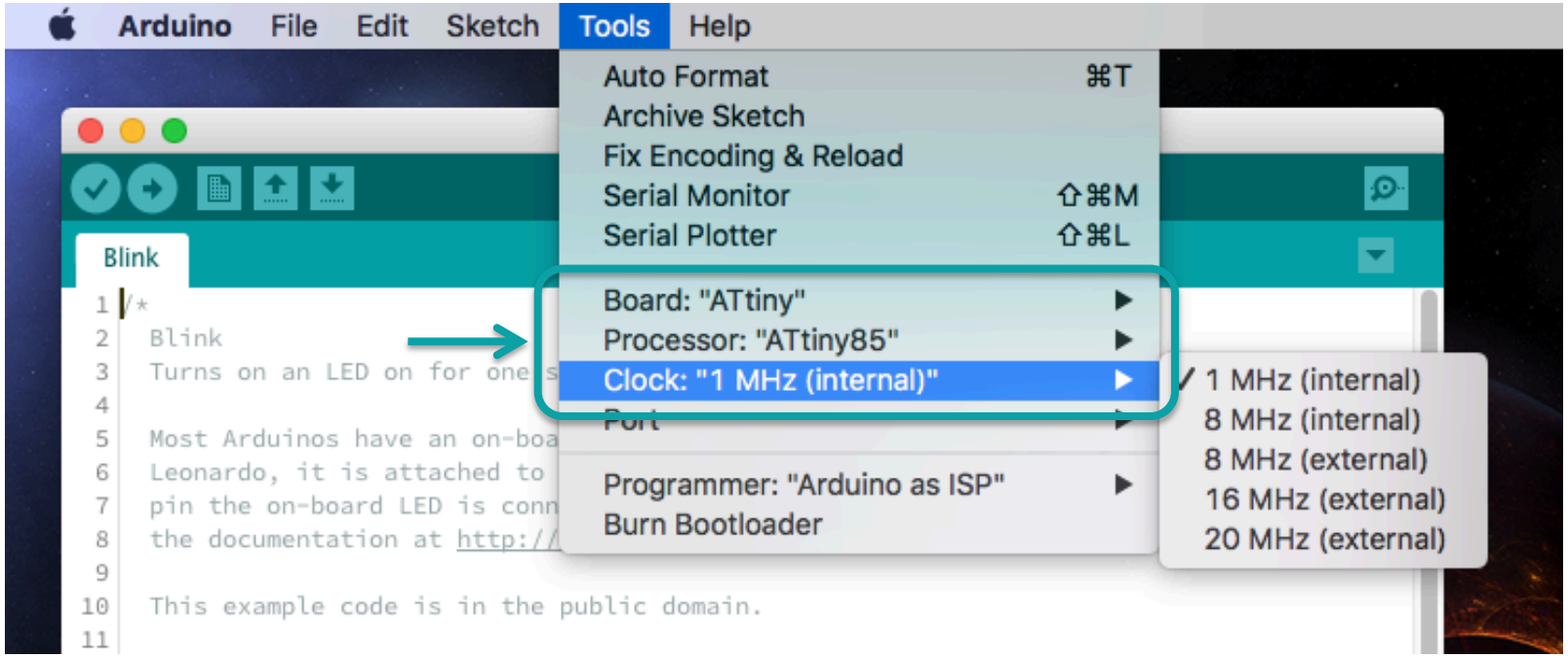
Genuino DAY2016

Arduino *as* ISP



https://www.arduino.cc/en/Tutorial/ArduinoISP

Configuração

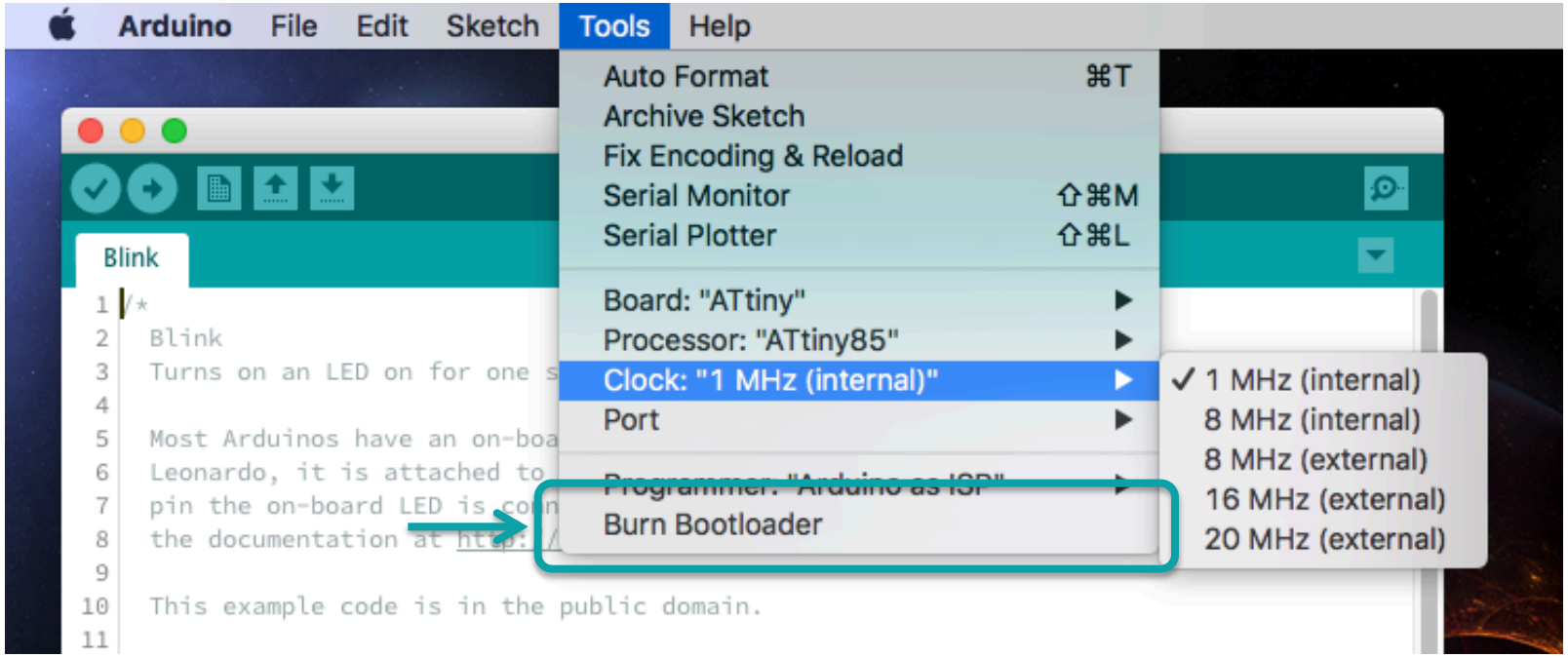


As opções de ***Clock*** afectam o consumo e algumas bibliotecas (ex. I^2C).

http://highlowtech.org/?p=1695

Genuino DAY2016

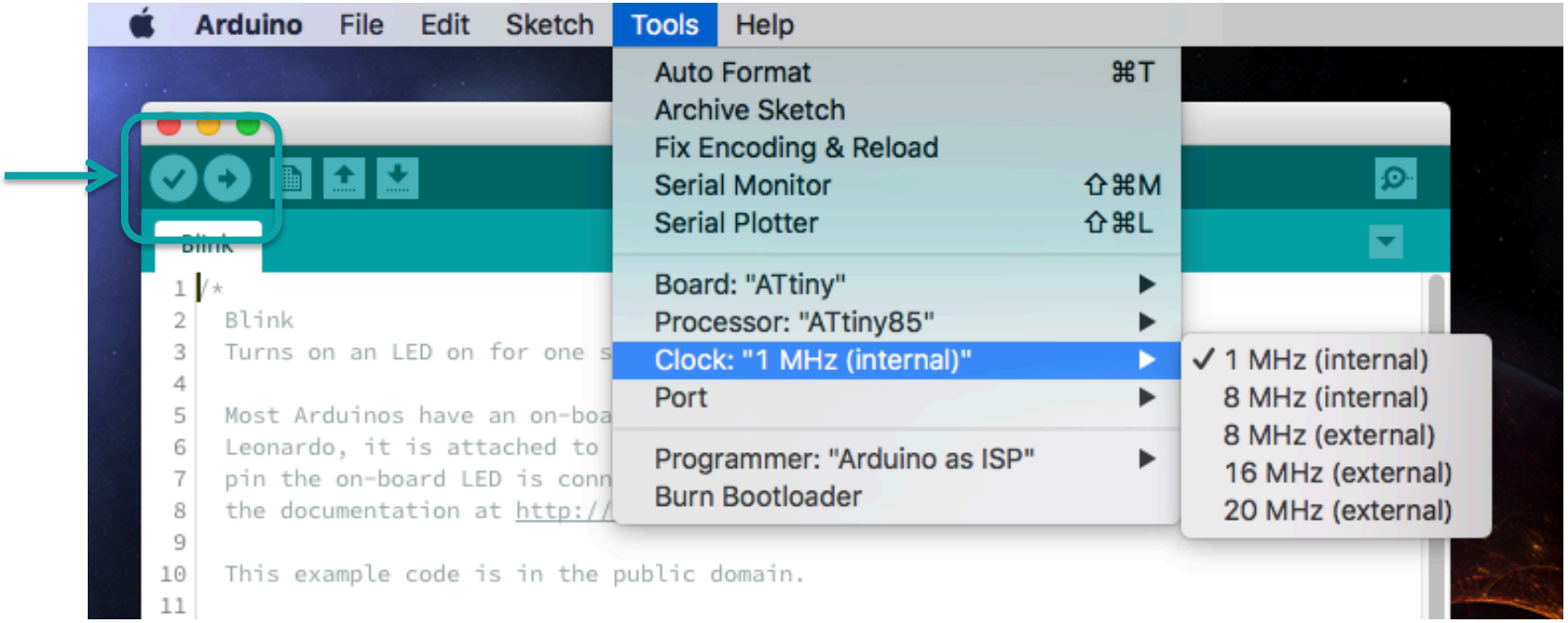
Configuração



Aqui o ***Burn Bootloader*** apenas configura os *fuses* do microcontrolador.

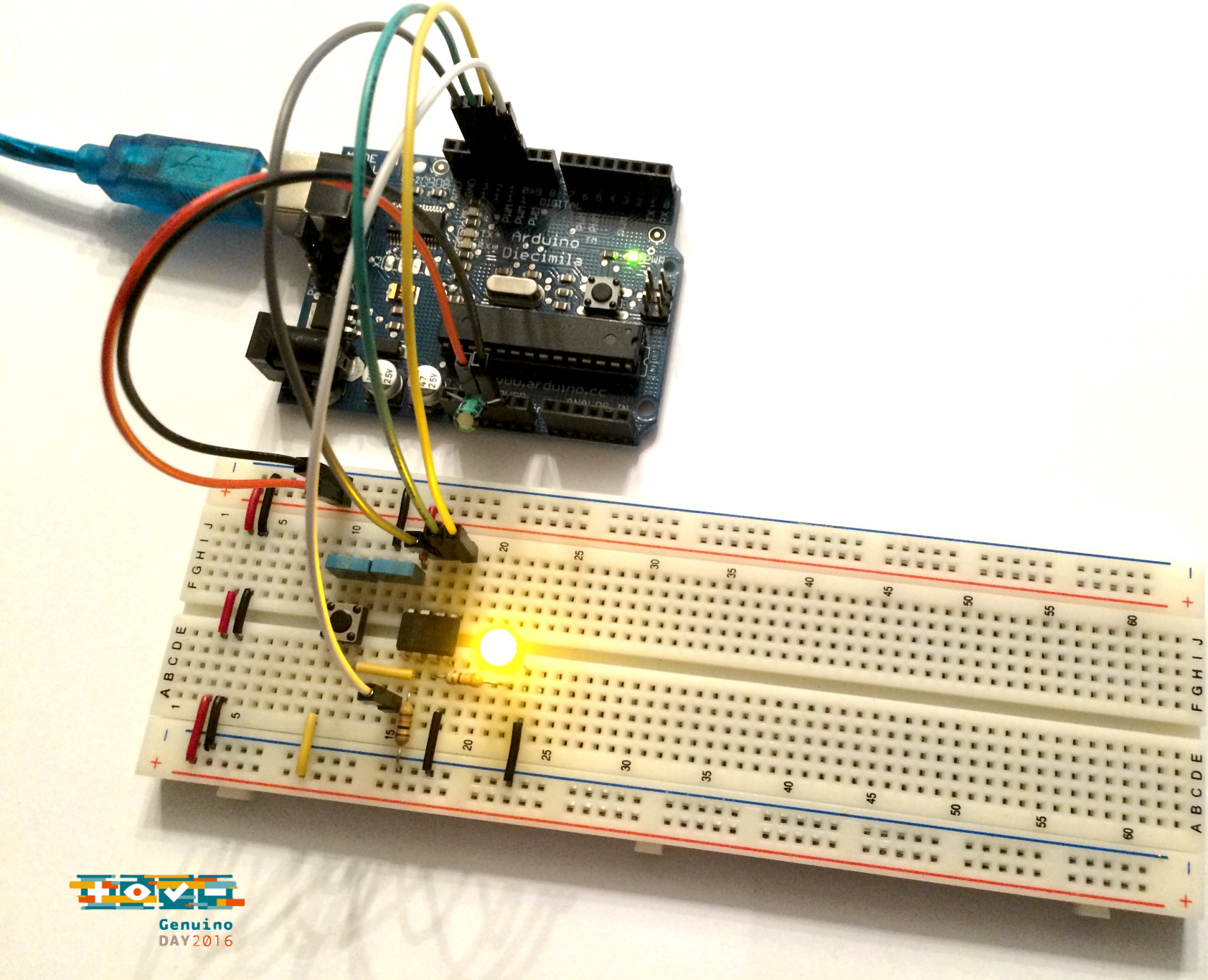
http://highlowtech.org/?p=1695

Programação



Pode usar-se o botão de ***Upload*** normal ou o ***Upload Using Programmer***.

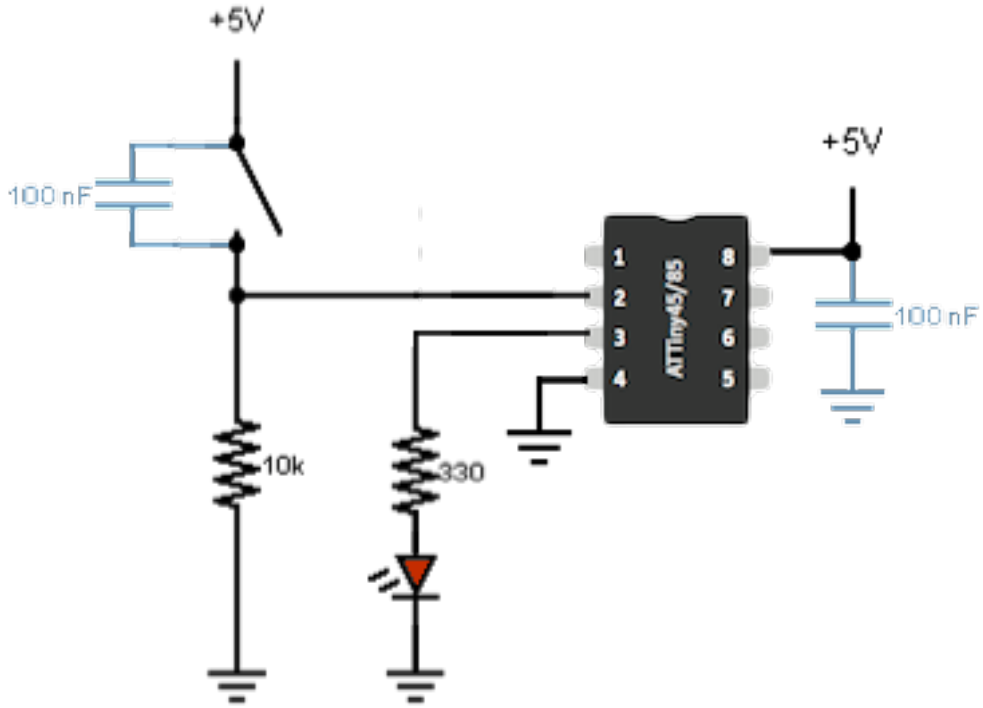
http://highlowtech.org/?p=1695

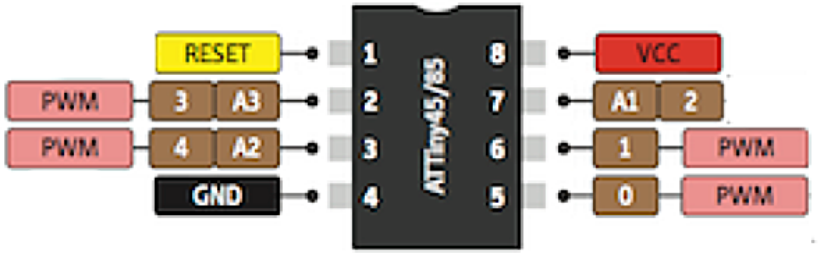


Genuino
DAY2016

Hello, World!

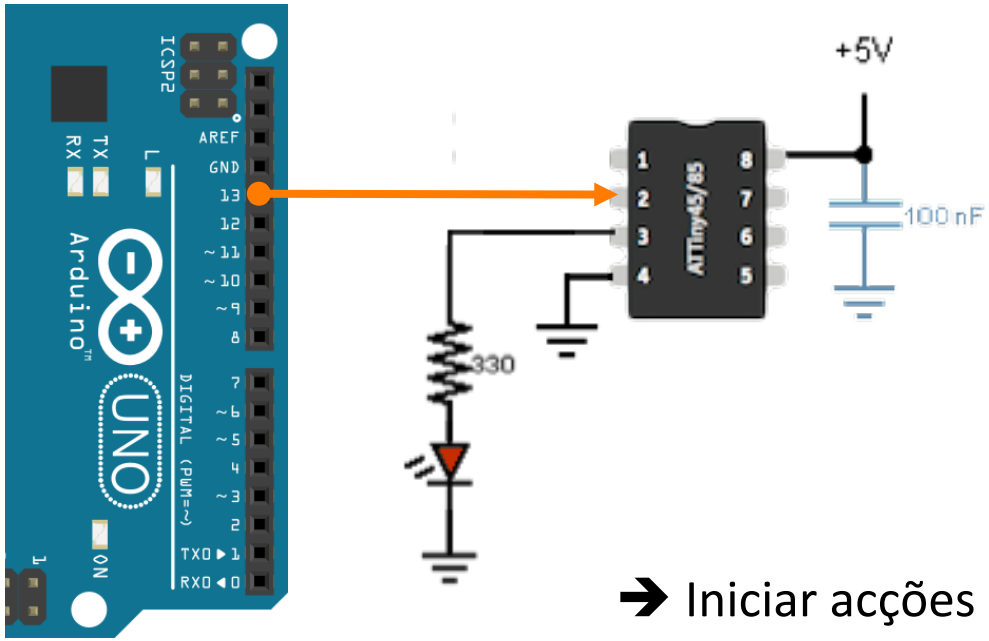
```
// O botão alterna o estado do LED...

const uint8_t LED_PIN = 4;
const uint8_t BUTTON_PIN = 3;

boolean led_state = LOW;
boolean last_button_state = LOW;

void setup() {
  pinMode(LED_PIN, OUTPUT);
  pinMode(BUTTON_PIN, INPUT);
}

void loop() {
  boolean button_state = digitalRead(BUTTON_PIN);

  if (button_state == HIGH && last_button_state == LOW) {
    led_state = !led_state;

    digitalWrite(LED_PIN, led_state);
  }

  last_button_state = button_state;
}
```





Genuino DAY2016

Comunicação por Sinalização



➔ Iniciar acções **bloqueantes** em *background*

Ex: Mover um motor *stepper* até ordem em contrário

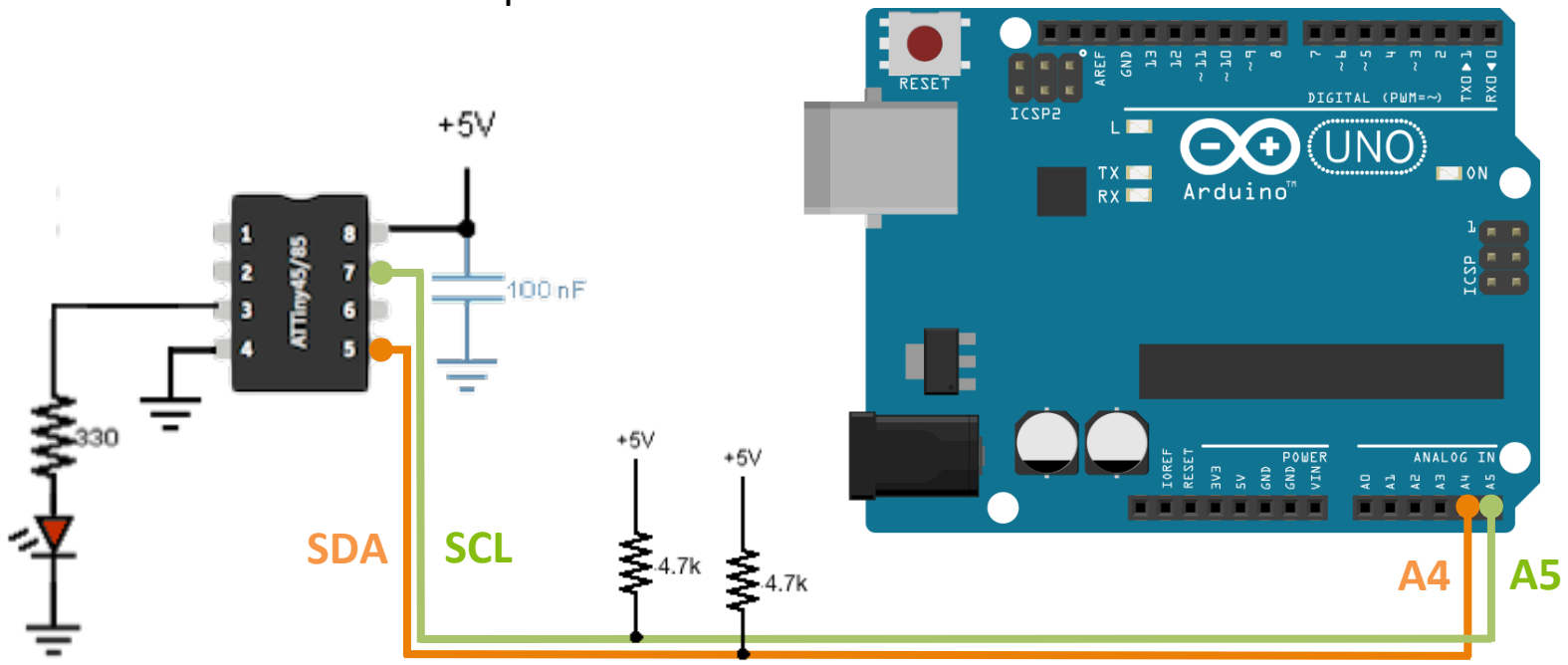
➔ Sinalizar a ocorrência de **eventos**

Ex: Processamento complexo de sensores

Genuino DAY2016

Comunicação por I²C

- Também conhecido como SMBus ou TWI
- Até **127 dispositivos** ligados em sequência
- Muitos sensores usam este protocolo



As resistências *pull-up* são **opcionais** quando o *master* é um Arduino.

Comunicação por I²C

```
// Master no Arduino...

#include <Wire.h>

const uint8_t SLAVE_ADDRESS = 0x03;

void setup() {
  Wire.begin();
}

void loop() {
  static uint8_t led_value = 1;
  static int8_t led_increment = 1;

  Wire.beginTransmission(SLAVE_ADDRESS);
  Wire.write(led_value);
  Wire.endTransmission();

  if (led_value == 0 || led_value == 255) {
    led_increment = -led_increment;
  }

  led_value += led_increment;
  delay(10);
}
```

```
// Slave no ATtiny...

#include <TinyWireS.h>

const uint8_t SLAVE_ADDRESS = 0x03;
const uint8_t LED_PIN = 4;

volatile uint8_t led_value = 0;

void i2c_incoming(uint8_t bytes) {
  // O master só envia 1 byte...
  led_value = TinyWireS.receive();
}

void setup() {
  TinyWireS.begin(SLAVE_ADDRESS);
  TinyWireS.onReceive(i2c_incoming);

  pinMode(LED_PIN, OUTPUT);
}

void loop() {
  analogWrite(LED_PIN, led_value);

  TinyWireS_stop_check();
  tws_delay(10);
}
```

https://github.com/rambo/TinyWire

Genuino DAY2016

Comunicação por I²C

```arduino
// Master no Arduino...

#include <Wire.h>

const uint8_t SLAVE_ADDRESS = 0x03;

void setup() {
  Wire.begin();
}

void loop() {
  static uint8_t led_value = 1;
  static int8_t led_increment = 1;

  Wire.beginTransmission(SLAVE_ADDRESS);
  Wire.write(led_value);
  Wire.endTransmission();

  if (led_value == 0 || led_value == 255) {
    led_increment = -led_increment;
  }

  led_value += led_increment;
  delay(10);
}
```

```arduino
// Slave no ATtiny...

#include <TinyWireS.h>

const uint8_t SLAVE_ADDRESS = 0x03;
const uint8_t LED_PIN = 4;

volatile uint8_t led_value = 0;

void i2c_incoming(uint8_t bytes) {
  // O master só envia 1 byte...
  led_value = TinyWireS.receive();
}

void setup() {
  TinyWireS.begin(SLAVE_ADDRESS);
  TinyWireS.onReceive(i2c_incoming);

  pinMode(LED_PIN, OUTPUT);
}

void loop() {
  analogWrite(LED_PIN, led_value);

  TinyWireS_stop_check();
  tws_delay(10);
}
```

https://github.com/rambo/TinyWire

Genuino DAY2016

Comunicação por I²C

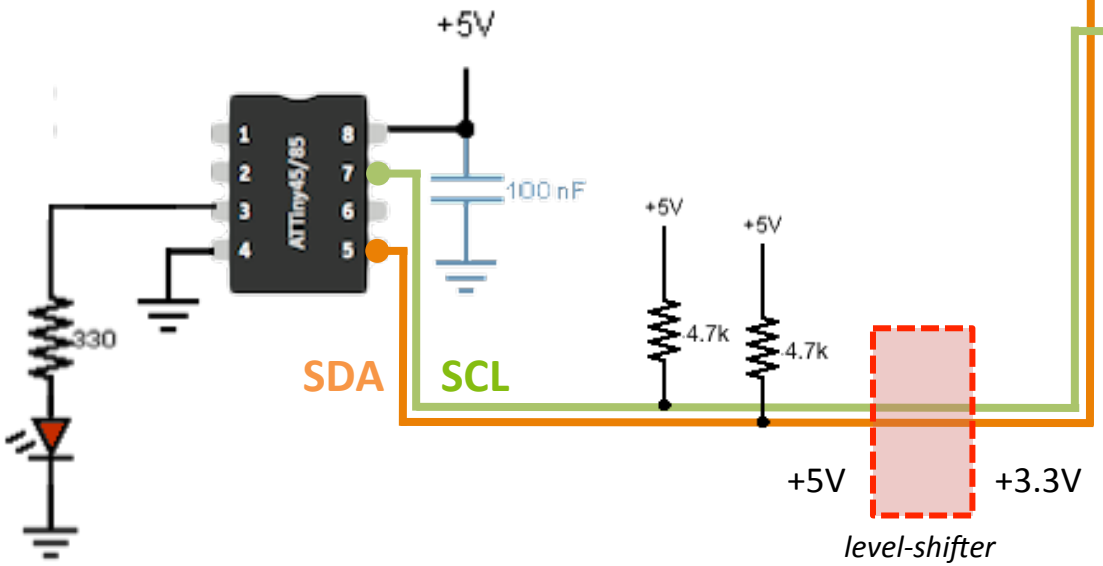
```ino
// Master no Arduino...

#include <Wire.h>

const uint8_t SLAVE_ADDRESS = 0x03;

void setup() {
  Wire.begin();
}

void loop() {
  static uint8_t led_value = 1;
  static int8_t led_increment = 1;

  Wire.beginTransmission(SLAVE_ADDRESS);
  Wire.write(led_value);
  Wire.endTransmission();

  if (led_value == 0 || led_value == 255) {
    led_increment = -led_increment;
  }

  led_value += led_increment;
  delay(10);
}
```

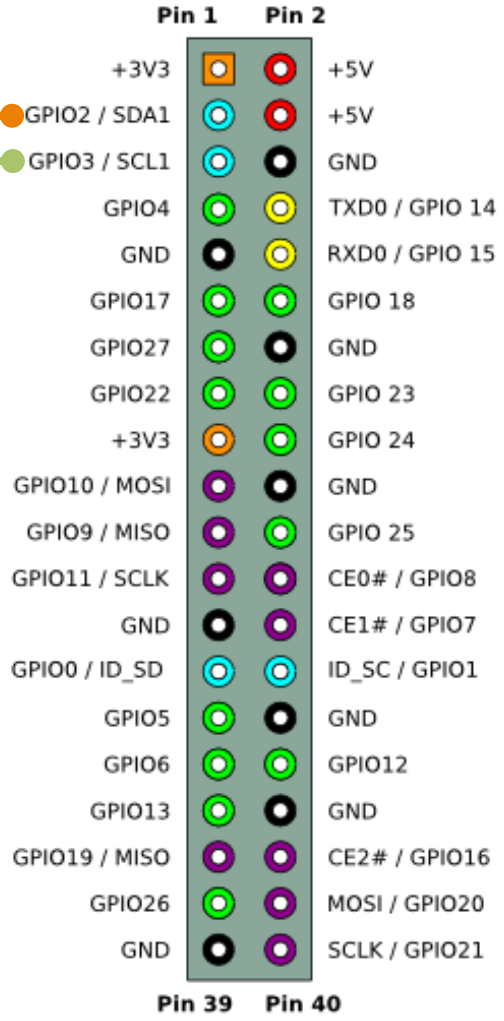
```ino
// Slave no ATtiny...

#include <TinyWireS.h>

const uint8_t SLAVE_ADDRESS = 0x03;
const uint8_t LED_PIN = 4;

volatile uint8_t led_value = 0;

void i2c_incoming(uint8_t bytes) {
  // O master só envia 1 byte...
  led_value = TinyWireS.receive();
}

void setup() {
  TinyWireS.begin(SLAVE_ADDRESS);
  TinyWireS.onReceive(i2c_incoming);

  pinMode(LED_PIN, OUTPUT);
}

void loop() {
  analogWrite(LED_PIN, led_value);

  TinyWireS_stop_check();
  tws_delay(10);
}
```

https://github.com/rambo/TinyWire

Genuino DAY2016

I²C com o Raspberry Pi





- O *level-shifter* (bidireccional) é **obrigatório**!
- O ATtiny tem de estar configurado para 8 MHz
- Pode ser necessário abrandar* o *bus* no Raspberry Pi

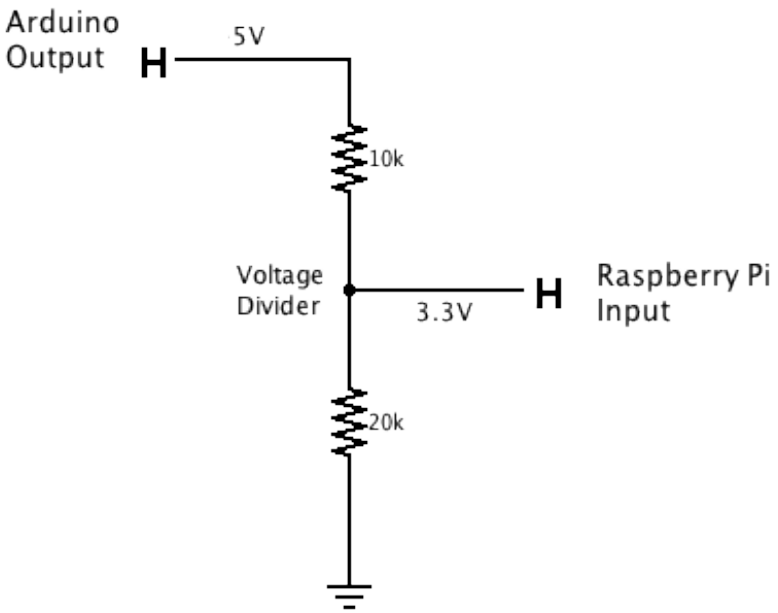
https://gist.github.com/carlosefr/5849220

* `echo "options i2c_bcm2708 baudrate=10000" > /etc/modprobe.d/i2c.conf`

Genuino DAY2016

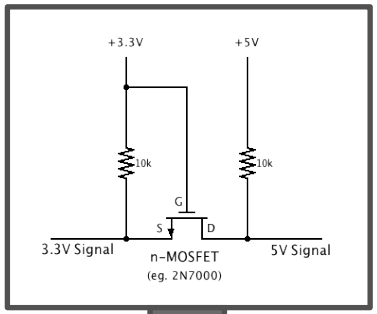
(Logic-Level Shifting)

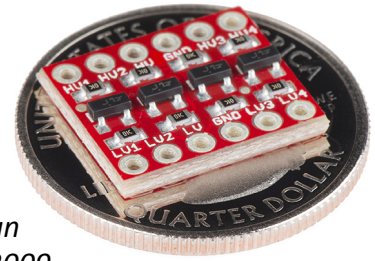
Unidireccional
(linhas de sinalização)



No sentido contrário, 3.3V é reconhecido como um HIGH lógico e não é necessário *level-shifting*.

Bidireccional
(linhas de dados)





Sparkfun
BOB-12009

Genuino
DAY2016

I²C com o Raspberry Pi

```python
# Master no Raspberry Pi...

import time
import smbus

SLAVE_ADDRESS = 0x03

led_value = 1
led_increment = 1

i2c = smbus.SMBus(1)

while True:
    time.sleep(0.01)

    try:
        i2c.write_byte(SLAVE_ADDRESS, led_value)
    except IOError as e:
        print("IOError: " + str(e))
        continue

    if led_value == 0 or led_value == 255:
        led_increment = -led_increment

    led_value += led_increment
```

```arduino
// Slave no ATtiny...

#include <TinyWireS.h>

const uint8_t SLAVE_ADDRESS = 0x03;
const uint8_t LED_PIN = 4;

volatile uint8_t led_value = 0;

void i2c_incoming(uint8_t bytes) {
  // O master só envia 1 byte...
  led_value = TinyWireS.receive();
}

void setup() {
  TinyWireS.begin(SLAVE_ADDRESS);
  TinyWireS.onReceive(i2c_incoming);

  pinMode(LED_PIN, OUTPUT);
}

void loop() {
  analogWrite(LED_PIN, led_value);

  TinyWireS_stop_check();
  tws_delay(10);
}
```

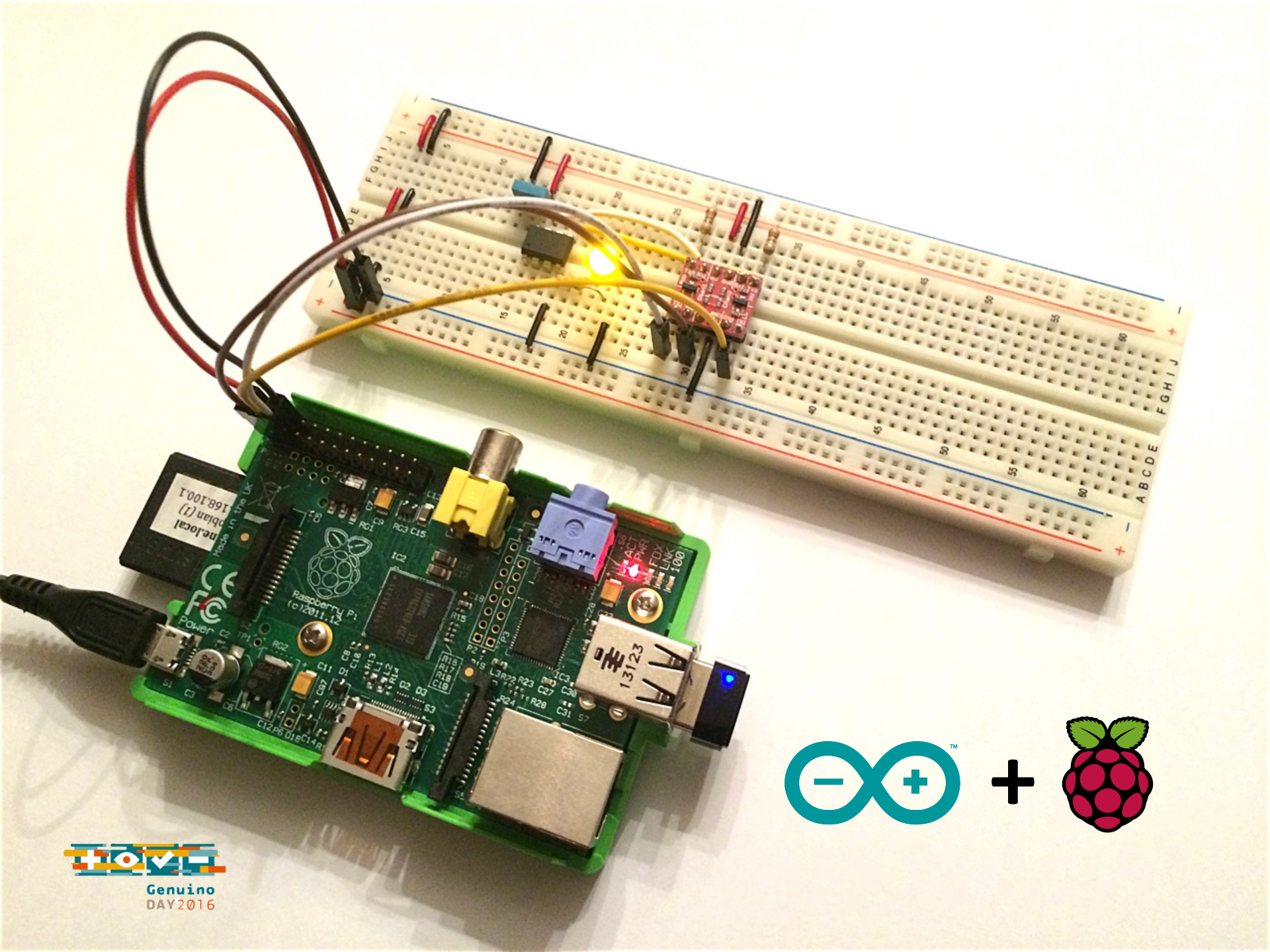
Genuino DAY2016

I²C com o Raspberry Pi

```python
# Master no Raspberry Pi...

import time
import smbus

SLAVE_ADDRESS = 0x03

led_value = 1
led_increment = 1

i2c = smbus.SMBus(1)

while True:
    time.sleep(0.01)

    try:
        i2c.write_byte(SLAVE_ADDRESS, led_value)
    except IOError as e:
        print("IOError: " + str(e))
        continue

    if led_value == 0 or led_value == 255:
        led_increment = -led_increment

    led_value += led_increment
```

```arduino
// Slave no ATtiny...

#include <TinyWireS.h>

const uint8_t SLAVE_ADDRESS = 0x03;
const uint8_t LED_PIN = 4;

volatile uint8_t led_value = 0;

void i2c_incoming(uint8_t bytes) {
  // O master só envia 1 byte...
  led_value = TinyWireS.receive();
}

void setup() {
  TinyWireS.begin(SLAVE_ADDRESS);
  TinyWireS.onReceive(i2c_incoming);

  pinMode(LED_PIN, OUTPUT);
}

void loop() {
  analogWrite(LED_PIN, led_value);

  TinyWireS_stop_check();
  tws_delay(10);
}
```

Genuino DAY2016


Genuino
DAY2016

Obrigado!

Questões?

PDF desta apresentação e código:
cloud.carlos-rodrigues.com/arduino-day/2016.zip

Carlos Rodrigues

cefrodrigues@gmail.com
twitter.com/carlosefr

Genuino DAY2016