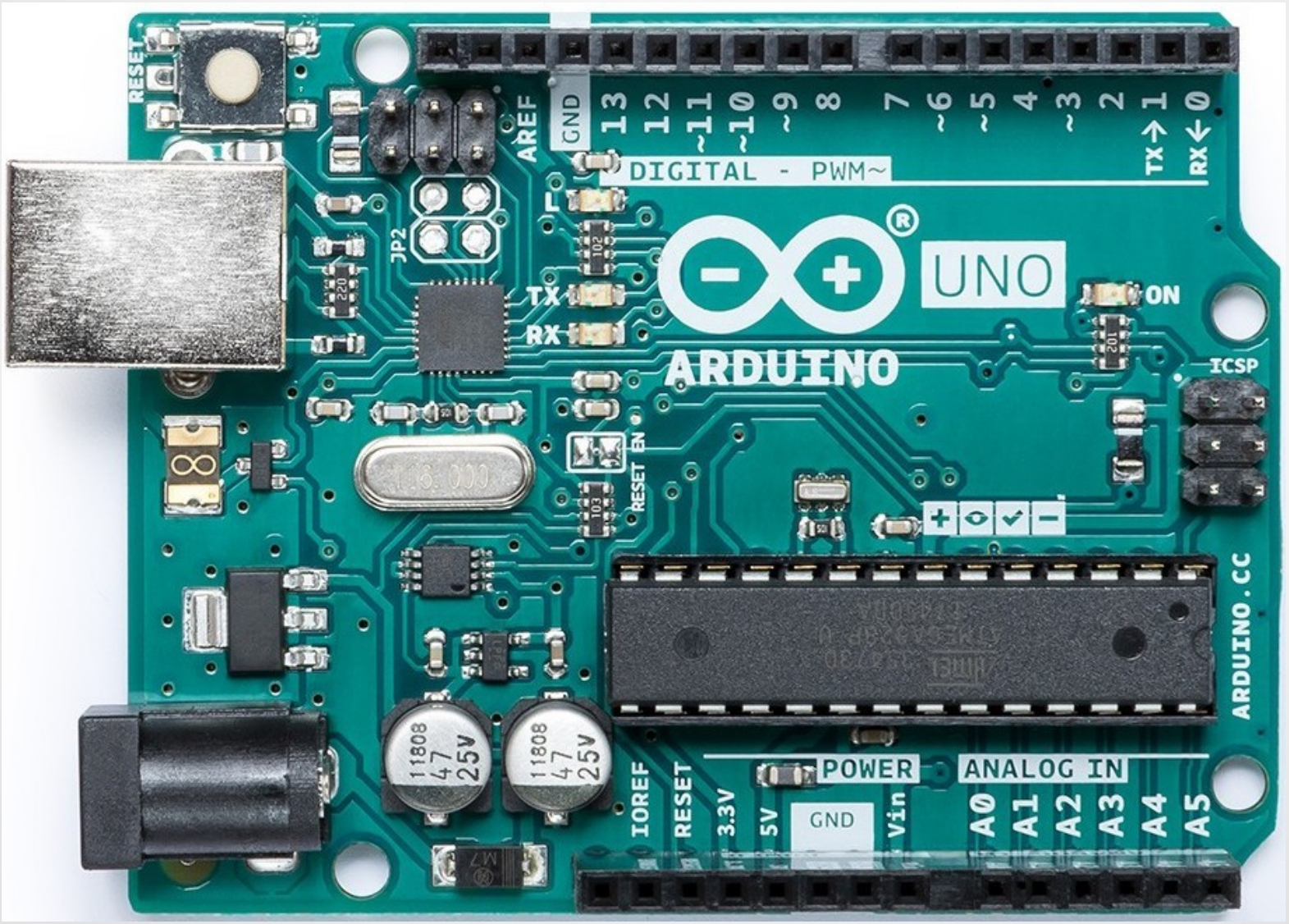
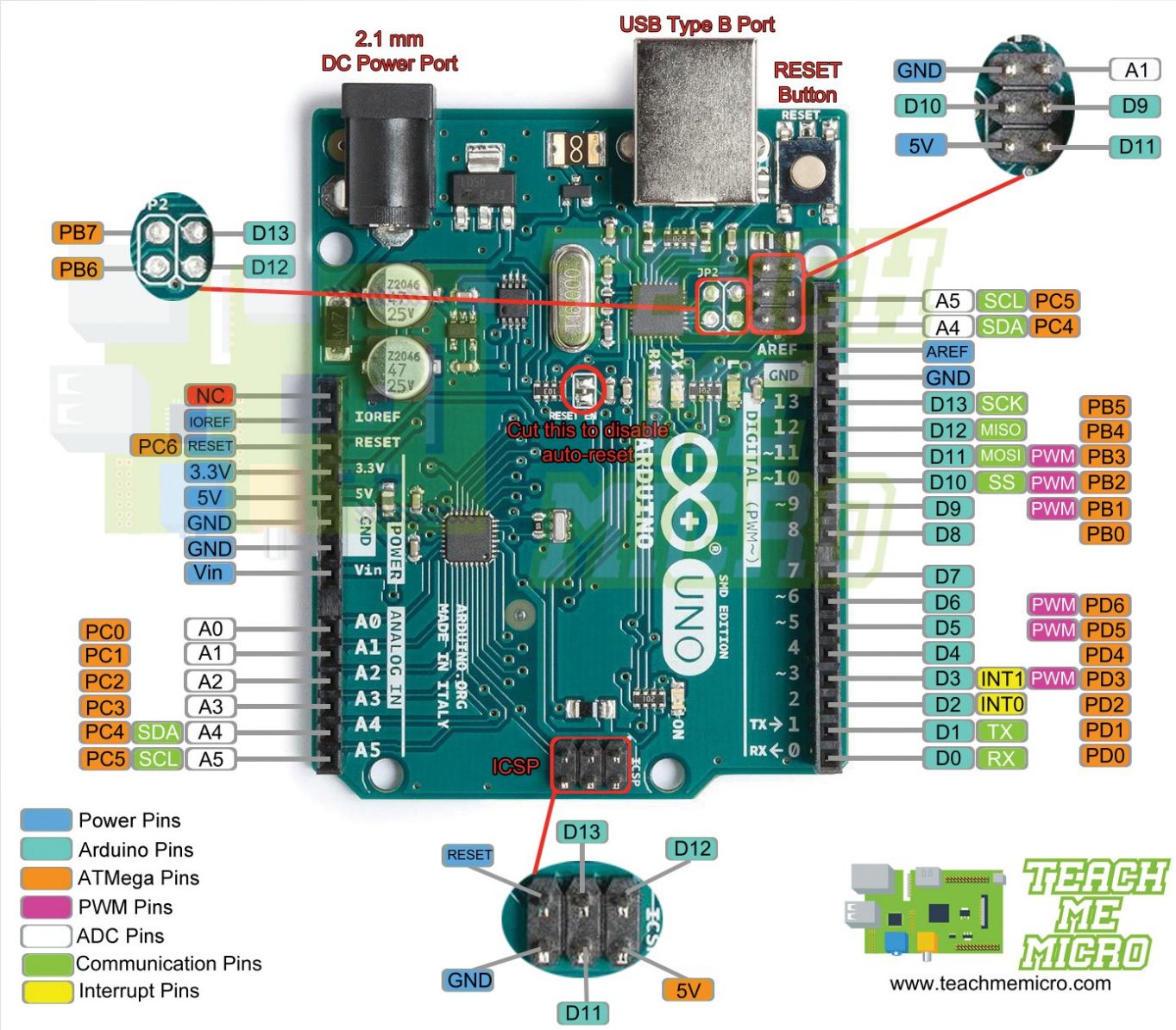
Entradas e saídas analóxicas



Víctor . Álvarez 201911

Introdución – pinout Arduino



E/S

Dixitais:

- Pins 0 a 13
  0 a 5V, 20 mA
  Low: 0 a 2V
  High: 3 a 5V

Entradas Analóxicas

- Pins A0 – A5
  0 a 5V, 20 mA prec 1024
  0 a 3V, 50 mA prec 1024

Saídas PWM

- ~ Pins 3, 5, 6, 9, 10, 11
  0 a 5 V, 20 mA prec 256

Comunic. Serie TX/RX

- Pins 0 e 1

ICSP

- 6 pins para comunicarse directamente co proc. Atmega328
- 6 pins para programar o USB

E/S analóxicas – Theremin

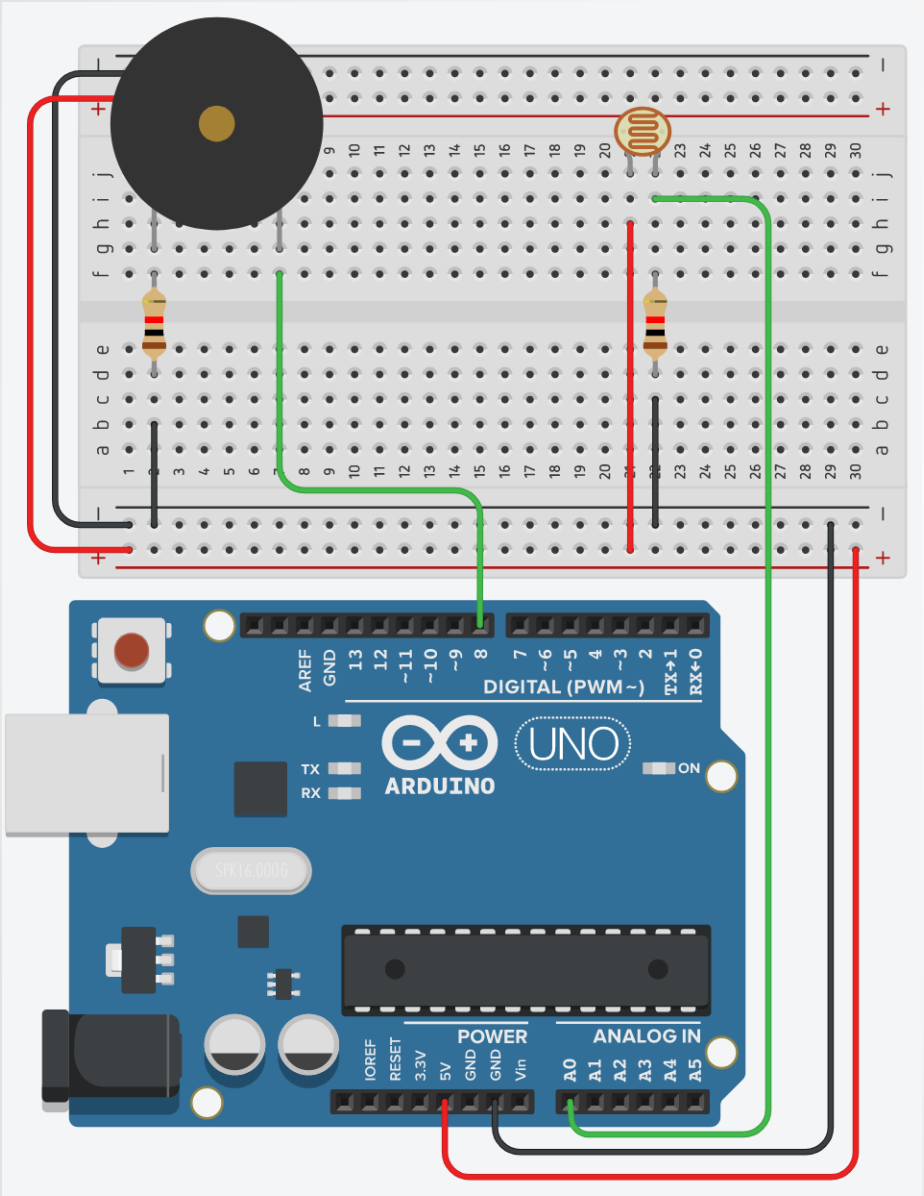
- O Theremin é un instrumento musical inventado por Leon Theremin en 1919, un dos primeiros instrumentos electrónicos. Consiste en dúas antenas dispostas perpendicularmente, unha vertial recta que controla o tono do son e outra circular horizontal, que controla o volumen do son. O concertista non toca fisicamente o instrumento.

https://en.wikipedia.org/wiki/Theremin

- Imos emular unha parte do mesmo cunha LDR e un zumbador piezoeléctrico. A LDR detectará a posición das mans e o zumbador emitirá un son.

Víctor . Álvarez 201911

E/S analóxicas – Theremin



theremin | Arduino 1.8.10

theremin

```
/*
 * Script que simula un Theremin.
 * Mediante unha LDR e un zumbador
 * detectamos a presencia dunha man
 * e mudamos a frecuencia emitida
 * polo zumbador.
 */

#define LDR A0          //Pin analóxico da LDR
#define BUZZER 8        //Pin dixital do zumbador

#define SILENCE 800     //Sen son ao superar umbral
#define FEC_MIN 500     //Frecuencias min e max do
#define FEC_MAX 1500    //zumbador

int valorLDR, frecuencia;
String mensaxe = "";
int demora = 200;

void setup() {
  Serial.begin(9600);
  pinMode(BUZZER, OUTPUT);
}

void loop() {
  valorLDR = analogRead(LDR);
  if(valorLDR <= SILENCE) {
    frecuencia = map(valorLDR, 0, SILENCE, FEC_MIN, FEC_MAX);
    tone(BUZZER, frecuencia);
    mensaxe = "Valor LDR: " + String(valorLDR) + "\t";
    mensaxe += "Frecuencia: " + String(frecuencia) + "\n";
    Serial.print(mensaxe);
  }
  else {
    noTone(BUZZER);
  }
  //delay(demora);
}
```

Guardado com Sucesso.

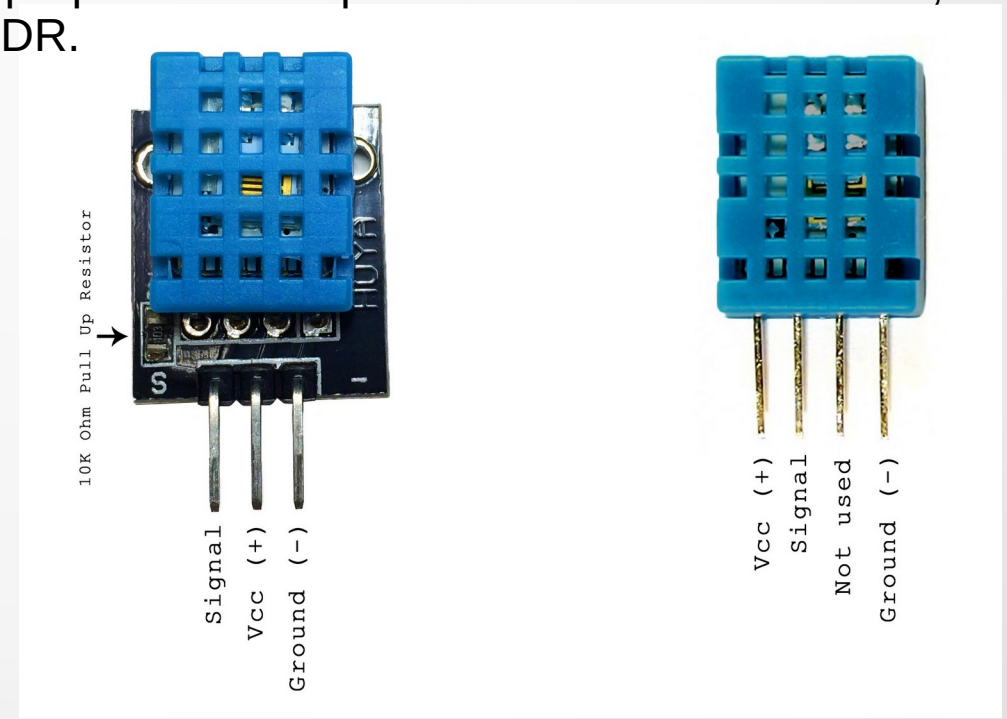
O rascunho usa 5258 bytes (16%) do espaço de armazenamento do program
Variáveis globais usam 251 bytes (12%) de memória dinâmica, restando

26 Arduino/Genuino Uno em /dev/cu.wchusbserial1410

Víctor . Álvarez 201911

E/S analóxicas – Theremin

- Podemos xogar cos parámetros SILENCE, FREC_MIN e FREC_MAX para modificar a frecuencia do son que nos da o zumbador.
- Tamén podemos axustar a tensión medida no punto medio do divisor de tensión entre a LDR e a resistencia de 47 kΩ, para que mapeala aos valores 0 a 1023.
- Podemos acoplar outros sensores analóxicos para mudar o ton do zumbador. Por exemplo, imos usar o DHT11, que nos proporciona temperatura e humidade relativa, para usalo como entrada en lugar da LDR.
- O DHT11 é un sensor que incorpora dous sensores analóxicos, un de humidade e unha NTC. No módulo que adoita aparecer nos kits de Arduino, tamén incorpora o circuíto de control para enviar os valores por un único pin. Isto facilita moito a programación, porque podemos usar diversas librerías dispoñibles na rede ou a través do xestor de librerías do IDE.



Datasheet DHT11 Datasheet modulo DHT11

E/S analóxicas – DHT11

- Tamén existe un sensor análogo, o DHT22. Nos seguintes enlaces podes encontrar informacion sobre características técnicas dos dous, así como a maneira de empregalos.

https://lastminuteengineers.com/dht11-dht22-arduino-tutorial/

http://www.circuitbasics.com/how-to-set-up-the-dht11-humidity-sensor-on-an-arduino/

- Nós imos empregar a librería <DHT.h> de Adafruit, para obter valores. Para instalar a librería, imos como sempre ao xestor de librerías do IDE de Arduino.
- Podemos acoplar outros sensores analóxicos para mudar o ton do zumbador. Por exemplo, imos usar o DHT11, que nos proporciona temperatura e humidade relativa, para usalo como entrada en lugar da LDR.

Datasheet DHT11 Datasheet modulo DHT11

E/S analóxicas – DHT11

```
/*
 * Script para medir temperatura
 * e humidade empregando o sensor
 * analóxico DHT11
 */
#include <DHT.h>

#define PIN_DHT A0

int tempo = 6000;
  DHT sensor(PIN_DHT, DHT11);  //Declaramos obxecto dht
float humidade, temperatura;

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

void loop() {
  humidade = sensor.readHumidity();
  temperatura = sensor.readTemperature();

  //Detección de erros
  if(isnan(humidade) || isnan(temperatura)) {
    Serial.println("Fallo na lectura do DHT11");
    return;
  }
  Serial.print("Humidade:\t");
  Serial.print(humidade);
  Serial.print(" %\t\t");
  Serial.print("Temperatura:\t");
  Serial.print(temperatura);
  Serial.print(" °C");
  Serial.print("\n");
  delay(tempo);
}
```

- Como se pode facer para sustituir a LDR do Theremin por este sensor?
- Que magnitude é máis axeitada: temperatura ou humidade relativa?

Datasheet DHT11

Datasheet modulo DHT11

Víctor . Álvarez 201911

E/S analóxicas – Detector de presenza

- Imos facer unha montaxe, de xeito que empregando un detector de presenza PIR (Passive Infra Red Sensor), se encenda unha lámpara dun pasillo ou dunha habitación.
- O PIR é un sensor de infravermellos que se activa ao detectar unha fonte de calor en movemento, dando nese caso un valor HIGH no pin de datos. Nalgúns modelos podemos modificar a sensibilidade (potenciómetro dereito) e máis o tempo de retardo na detección (potenciómetro esquerdo).
- Para alimentar unha lámpara, non podemos empregar a placa Arduino (lembra non superar os 20 mA nos pins), para iso imos botar man dun shield de relés ou ben relés individuais conectados á placa. Os que se usan habitualmente prescinden de bobina e teñen internamente un optoacoplador, co que se reduce moito as necesidades de alimentación cando o relé está activo.

Víctor . Álvarez 201911

E/S analóxicas – Detector de presenza

- Realiza unha montaxe e un script, de xeito que un PIR, xunto cun detector de luminosidade (LDR) e un pulsador, controlen o encendido dunha lámpara.
  - unha vez acesa, a luz debe agardar un tempo predeterminado para que calquera dos ‘sensores’ a apaguen,
  - se hai luz ambiental suficiente, a lámpara non encende aínda que o PIR o ordene,
  - o pulsador conmuta o estado da lámpara con independencia do estado do PIR e da luminosidade ambiental
- Podemos engadir a posibilidade de que o usuario final regule o tempo de encendido empregando un potenciómetro.

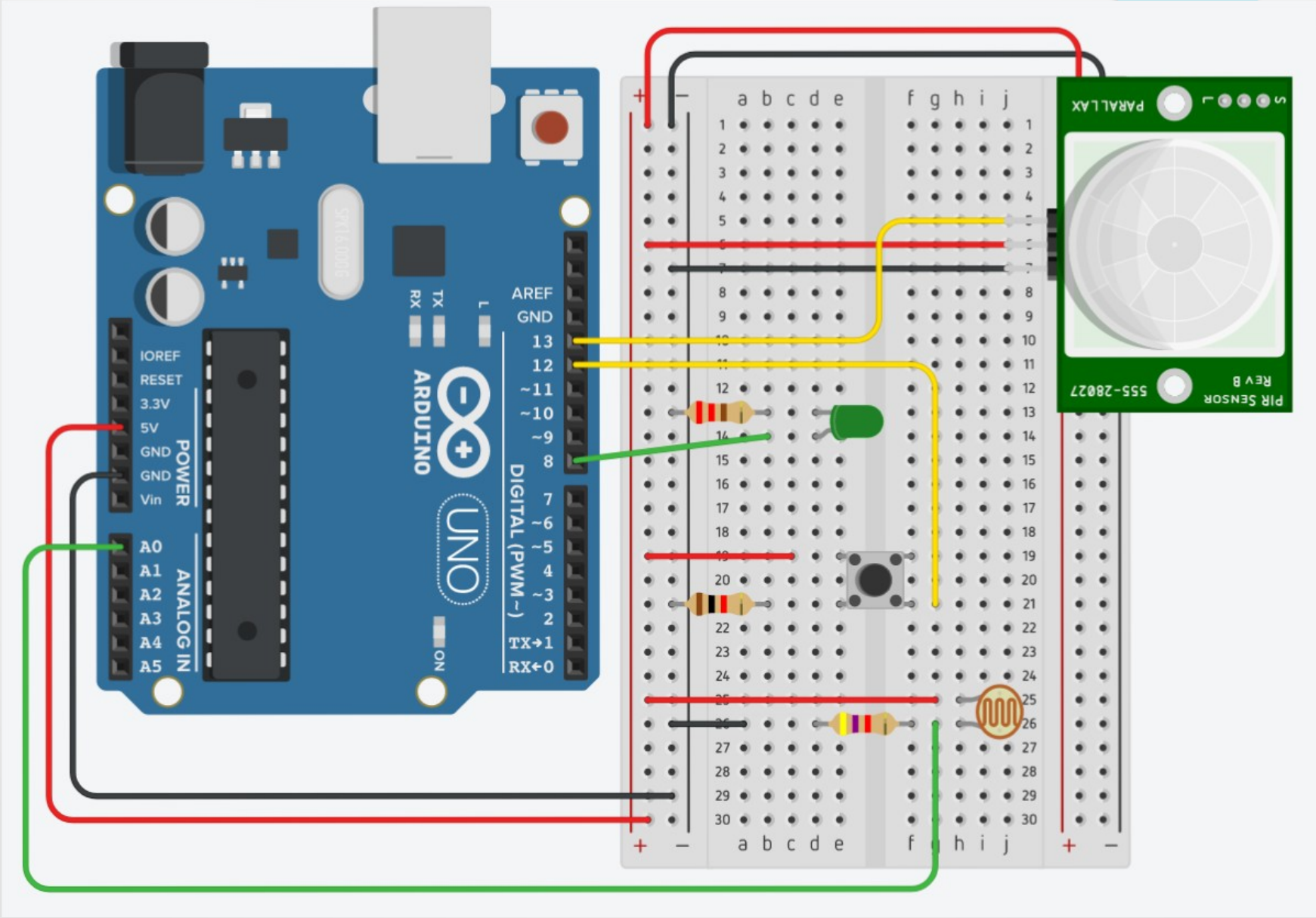
Víctor . Álvarez 201911

E/S analóxicas – Detector de presenza

- Da descrición anterior pódese concluír que:
  - (a) ao premer o pulsador, a lámpara muda de estado,
  - (b) con independencia do pulsador, a lámpara encende ao darse á vez PIR e LDR activas
  - (c) conclusión: L = B + PIR*LDR
- Mellor solucionar os problemas de estados antes de pelexarse cos condicionais na programación.

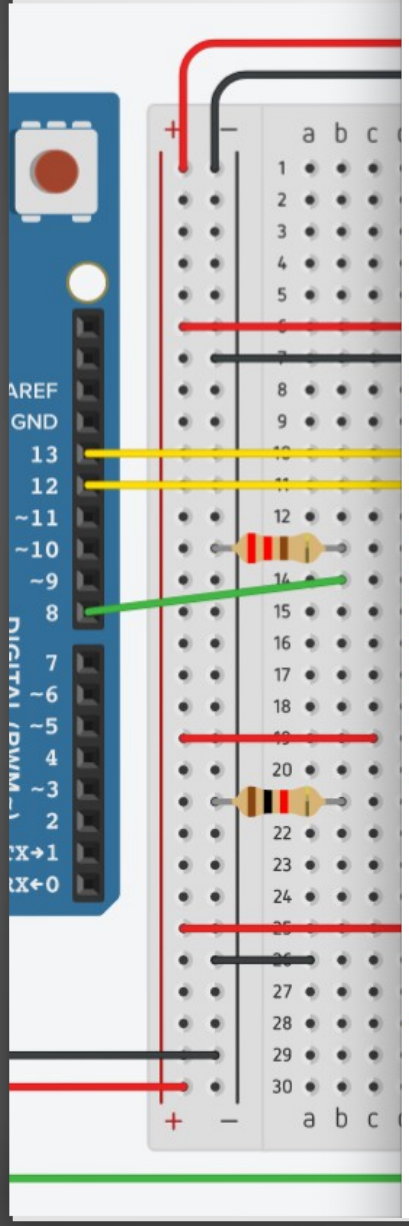
Víctor . Álvarez 201911

E/S analóxicas – Detector de presenza



Víctor . Álvarez 201911

E/S analóxicas – Detector de presenza



```
/*
 * Script que controla unha lámpara mediante
 * un PIR, unha LDR e un pulsador.
 * O encendido da lámpara faise cun relay,
 * que se simula co ON/OFF dun LED.
 */


#define PIR 13
#define BUTTON 12
#define LDR A0
#define RELAY 8

bool pir = false;
bool button = false;
int ldr = false;
bool relay = false;
bool lampara = false;
int valorLDR = 0;
int umbralLDR = 600; //Oscuridade
unsigned long int demoraLampara = 8000;
unsigned long int tInicial = 0;
bool tempoCumplido = true;
String mensaxe = "";
int demora = 500;

void setup() {
  pinMode(PIR, INPUT);
  pinMode(BUTTON, INPUT);
  pinMode(LDR, INPUT); //Non é preciso declaralo
  pinMode(RELAY, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  valorLDR = constrain(analogRead(LDR), 30, 900);
  valorLDR = map(valorLDR, 30, 900, 1023, 0);
  ldr = (valorLDR >= umbralLDR);
  pir = digitalRead(PIR);
  button = digitalRead(BUTTON);
  lampara = button || pir && ldr;
```

```
unsigned long int tInicial = 0;
bool tempoCumplido = true;
String mensaxe = "";
int demora = 500;

void setup() {
  pinMode(PIR, INPUT);
  pinMode(BUTTON, INPUT);
  pinMode(LDR, INPUT); //Non é preciso declaralo
  pinMode(RELAY, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  valorLDR = constrain(analogRead(LDR), 30, 900);
  valorLDR = map(valorLDR, 30, 900, 1023, 0);
  ldr = (valorLDR >= umbralLDR);
  pir = digitalRead(PIR);
  button = digitalRead(BUTTON);
  lampara = button || pir && ldr;
  //Se rele non activo AND hai que activar lampara AND
  //pasou suficiente tempo ==> actualizamos
  if(!relay && lampara && tempoCumplido) {
    relay = lampara;
    digitalWrite(RELAY, HIGH);
    tInicial = millis();
  }
  //Se se
  else if(tempoCumplido) {
    relay = false;  //Apagamos lampara
    digitalWrite(RELAY, LOW);
    tInicial = 0;   //Reiniciamos tempo actual
    delay(demora);  //Demoramos nova deteccion
  }
  mensaxe = "Lampara: " + String(lampara) + "\tBoton: " + String(but
  mensaxe += "\tRele: " + String(relay) + "\t\tPIR: " + String(pir);
  mensaxe += "\t\tLDR: " + String(ldr) + "\tDT: " + String(tempoCump
  Serial.println(mensaxe);
  tempoCumplido = millis() - tInicial >= demoraLampara ;
  delay(demora);
}
```

Víctor . Álvarez 201911

E/S analóxicas

- Nesta unidade aprendemos a:
  - Combinar entradas relativas a sensores analóxicos para producir saídas PWM,
  - combinar varios sensores para controlar un indicador sonoro ou LEDs
  - Usar as funcións tone() e noTone() da librería estándard de Arduino como saídas analóxicas (PWM) para un zumbador.
  - Usar un sensor combinado de temperatura e humidade relativa (DHT11), a través dunha librería dispoñible na rede.
  - Usar sensor dixital PIR e combinalo con senseores analóxicos para activar un relay
  - Buscar e instalar librerías a través do xestor dispoñible no IDE de Arduino.
  - Usar funcións da saída serie para mostrar mensaxes do script.

Víctor . Álvarez 201911