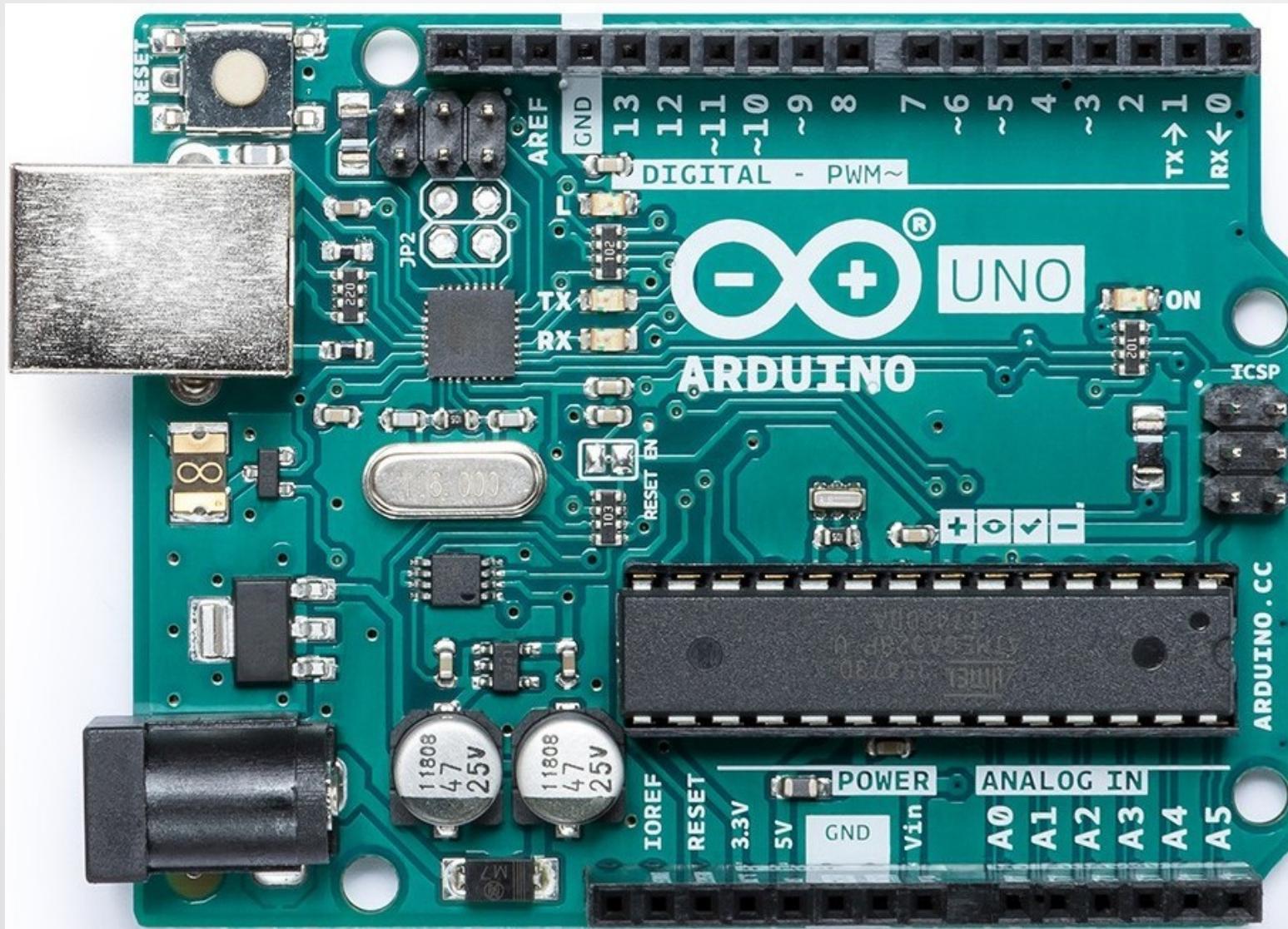
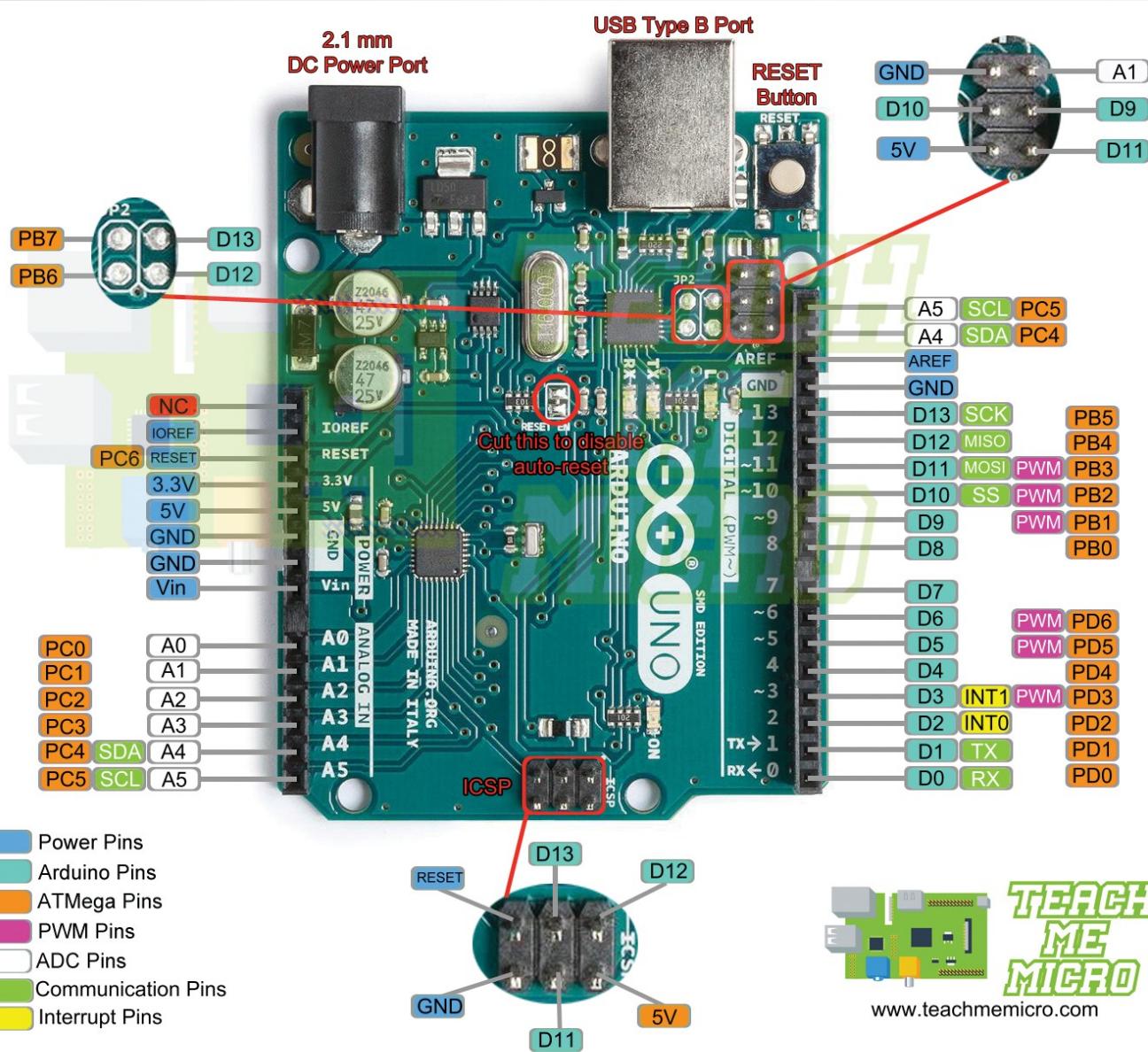


Entradas e saídas dixitais



Introducción – 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

Sáidas PWM

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

Comunic. Serie TX/RX

- Pins 0 e 1

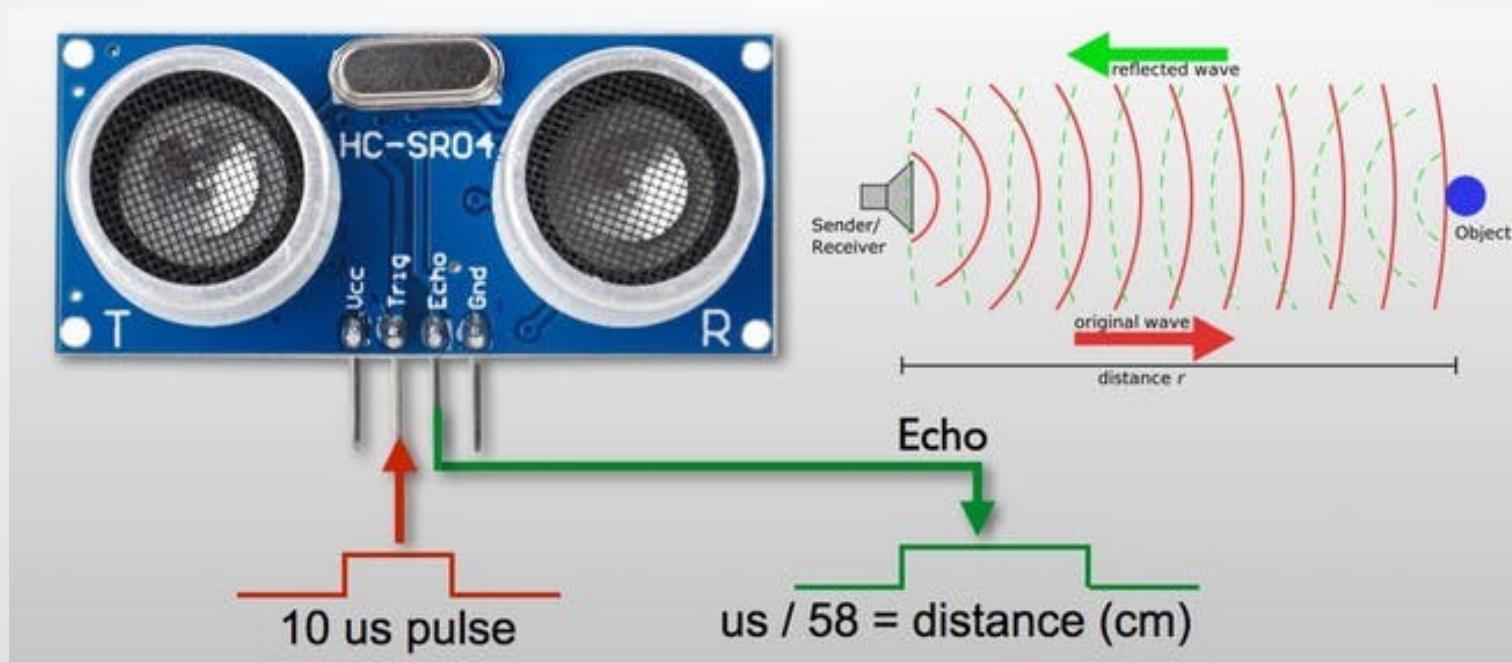
ICSP

- 6 pins para comunicarse directamente co proc.
Atmega328

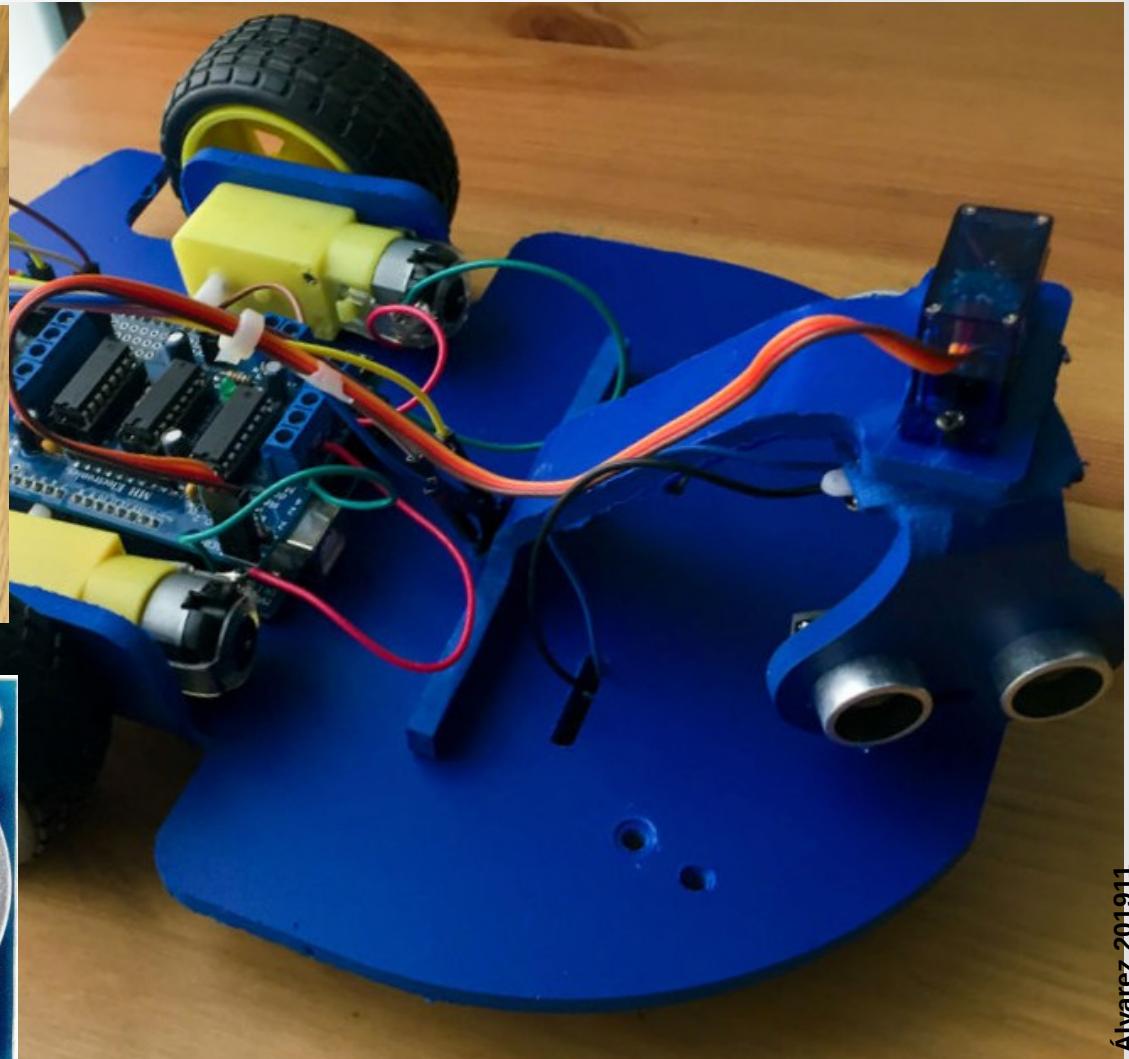
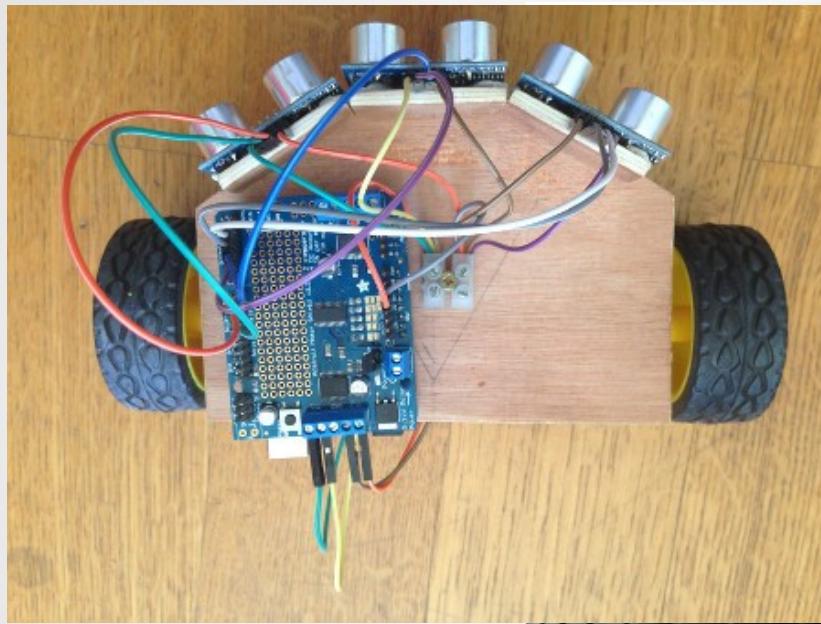
- 6 pins para programar o USB

E/S dixitais – Sensor ultrasóns HC-SR04

- O sensor HC-SR04 é un dos máis empregados para medir distancias. É un sensor dixital moi sinxelo e barato, que da resultados bastante precisos.
- Consiste nun emisor de ultrasóns combinado na mesma placa cun receptor (esquerda e dereita respectivamente). Funciona emitindo e recibindo pulsos de son (40 kHz). O tempo entre estes pulsos permítenos calcular a distancia ao obxecto.



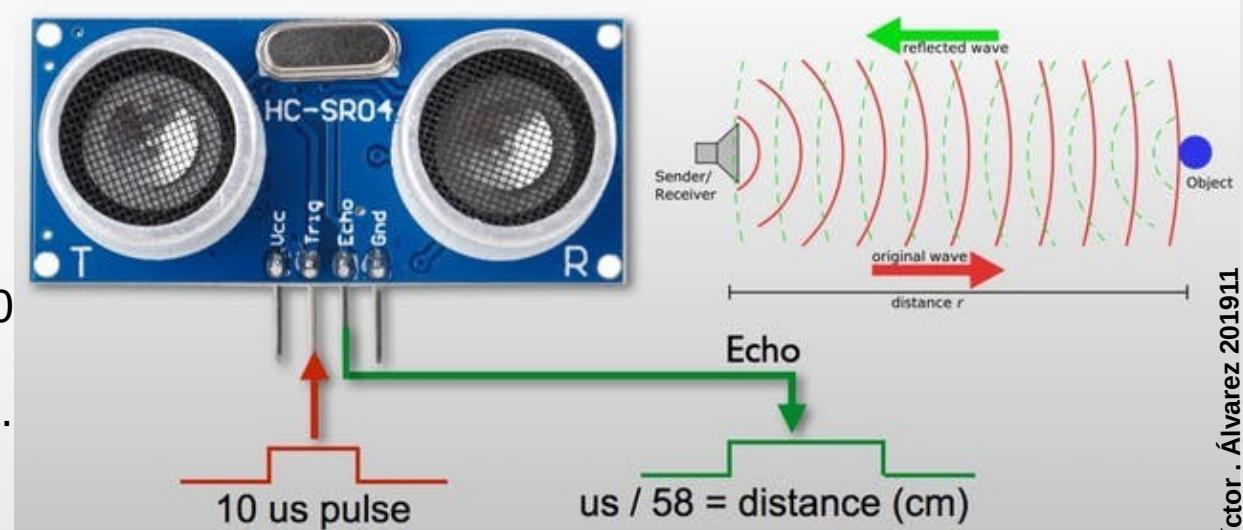
E/S dixitais – Sensor ultrasóns HC-SR04



E/S dixitais – Sensor ultrasóns HC-SR04

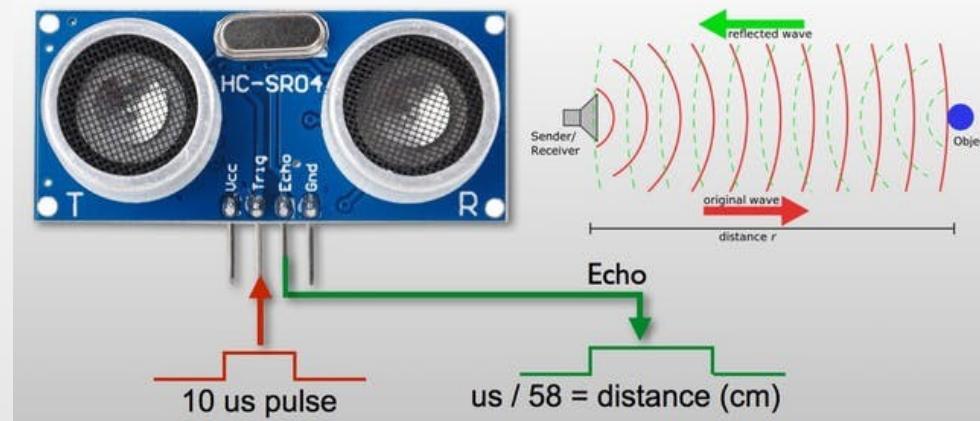
- O sensor HC-SR04 proporciona medidas de distancia entre 2 cm e 4 m, cunha precisión de 3 mm. O módulo inclúe un transmisor, un receptor e o circuíto de control.
- O principio básico de funcionamento é:
 - usar o pin de disparo (trigger) para emitir un pulso (HIGH) de como mínimo 10 μ s
 - o módulo envía automaticamente 8 pulsos de 40 kHz e detecta (echo) en que momento está de volta
 - o tempo que tardou en voltar é o tempo de ida e volta do pulso, polo que recibimos 2x a distancia que queremos medir
 - asumimos que a velocidade do son é de 340 m/s ou 0.034 cm/ μ s
- O obxecto debe estar a unha distancia angular mínima de 15°, aínda que as medicións más precisas son para ángulos de ~30°
- Recoméndase agardar uns 50 ms entre cada medida, aínda que se pode baixar ate 20 ms.

Datasheet HC-SR04

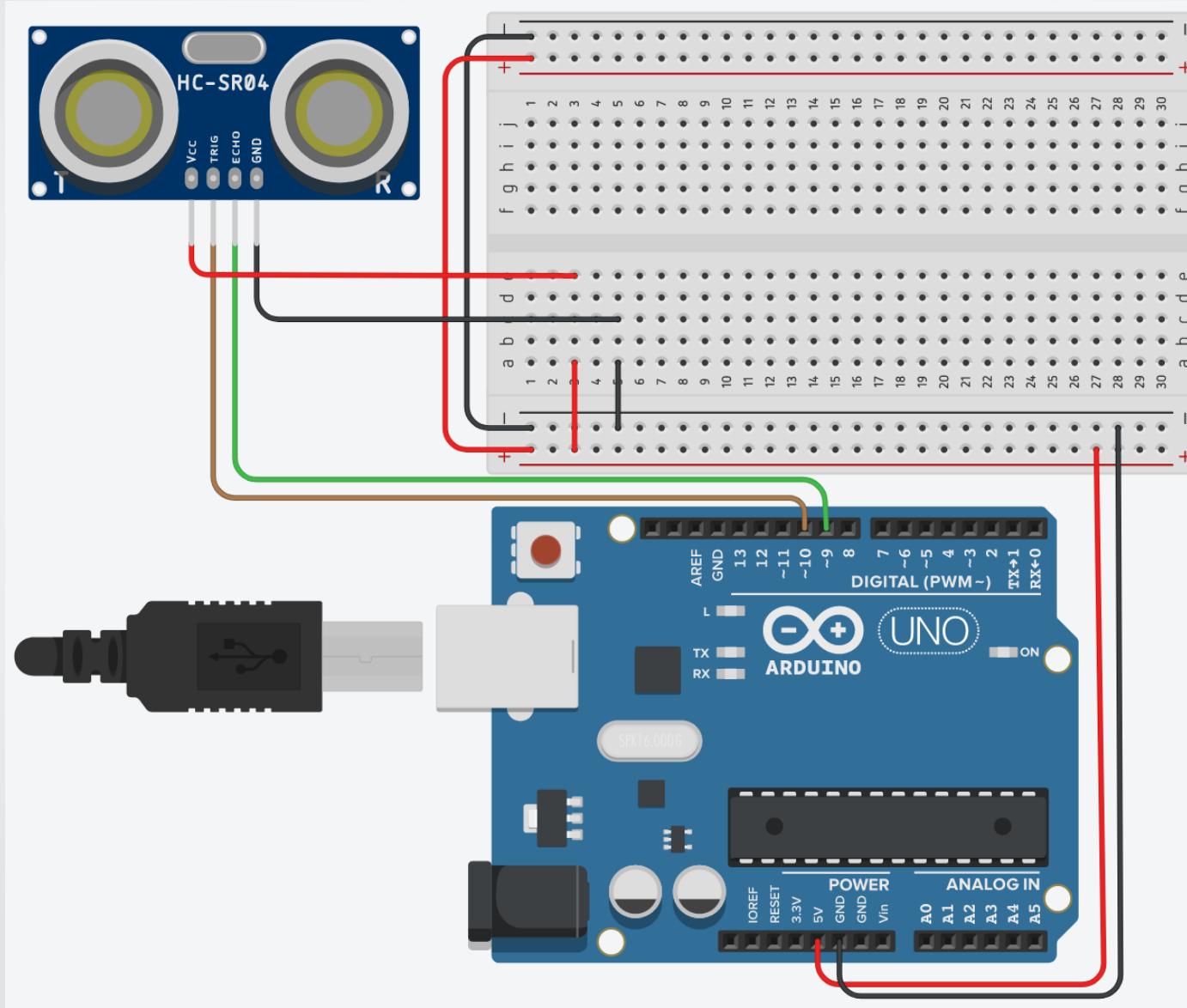


E/S dixitais – Sensor ultrasóns HC-SR04

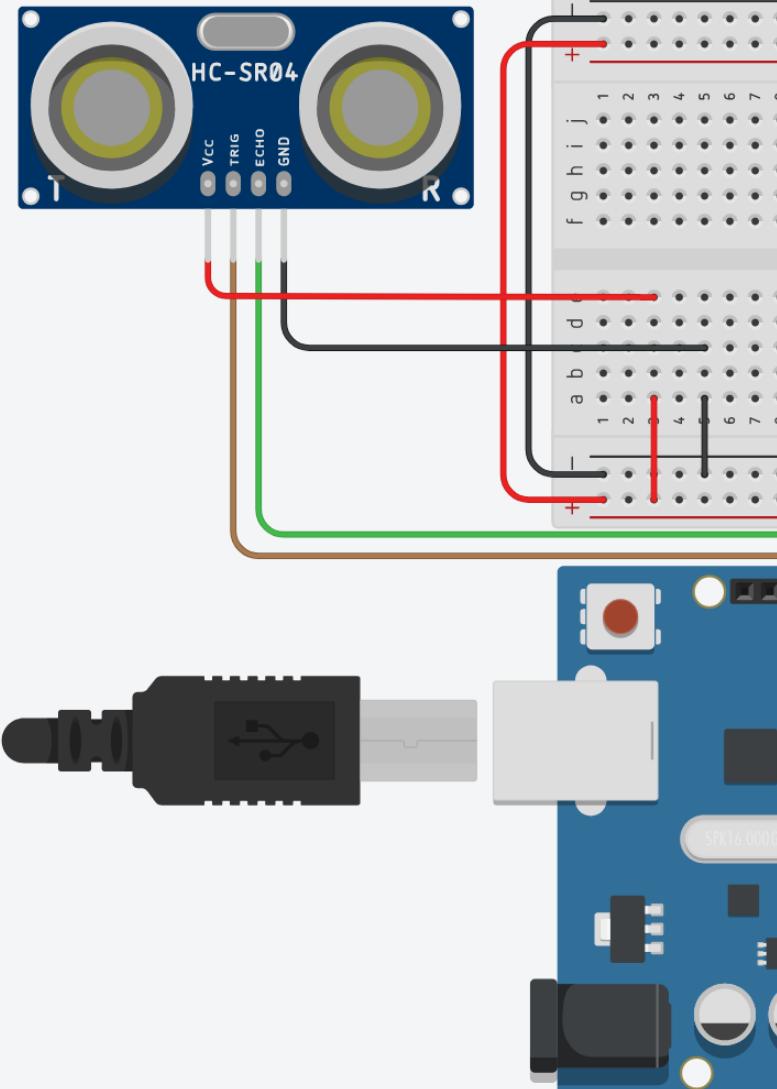
- Imos simular o sensor de aparcamento dun coche, para o que realizaremos unha montaxe e o sketch que a controla, de maneira que:
 - (a) detecte a distancia á que o coche se acerca ao obstáculo,
 - (b) cando se supere unha distancia mínima entre coche e obstáculo, se encendea un piloto no salpicadero,
 - (c) o conductor poda regular previamente cal é a distancia mínima admisible antes de que se encenda o piloto.
- Imos desenvolver o sketch por partes segundo o esquema anterior. Realizarremos primeiro a parte de medición, e despois incorporaremos o piloto empregando un LED e a regulación empregando un potenciómetro.



E/S dixitais – Sensor ultrasóns HC-SR04



E/S dixitais – Sensor ultrasóns HC-SR04



sensor.distancia.ultrasons.a.manini | Arduino 1.8.10

```
/*
 * Script para medir a distancia
 * a un obxecto, empregando un
 * sensor de ultrasóns HC-SR04
 */

#define TRIGGER 9
#define ECHO 10
//Definimos macro para calcular distancia
#define calcDist(t) t*0.017

float tempo, distancia;
float tInicial, tFinal;
int echo;
String mensaxe = "";
int espera = 200;

void setup() {
    Serial.begin(9600);
    pinMode(TRIGGER, OUTPUT);
    pinMode(ECHO, INPUT);
}

void loop() {
    //Enviamos un pulso co HC-SR04
    digitalWrite(TRIGGER, LOW);
    delayMicroseconds(5);
    digitalWrite(TRIGGER, HIGH);
    delayMicroseconds(10);
    digitalWrite(TRIGGER, LOW);
    echo = digitalRead(ECHO);
    tInicial = micros();
    while(echo == LOW) {
        echo = digitalRead(ECHO);
    }
    while(echo == HIGH) {
        echo = digitalRead(ECHO);
        tFinal = micros();
    }
    //A resta dos dous tempos danos
    //a duración do pulso desde que
    //se empeza a enviar até que se
}
```

Compilación Terminada

O rascunho usa 6132 bytes (19%) do espacio de almacenamiento
Variábeis globais usan 230 bytes (11%) de memoria dinâmica

51

sensor.distancia.ultrasons.a.manini | Arduino 1.8.10

```
sensor.distancia.ultrasons.a.manini
digitalWrite(TRIGGER, LOW);
echo = digitalRead(ECHO);
tInicial = micros();
while(echo == LOW) {
    echo = digitalRead(ECHO);
}
while(echo == HIGH) {
    echo = digitalRead(ECHO);
    tFinal = micros();
}
//A resta dos dous tempos danos
//a duración do pulso desde que
//se empeza a enviar até que se
//deixa de recibir
tempo = tFinal - tInicial;
tempo /= 2;

//Calculamos a distancia en cm
distancia = calcDist(tempo);

//Escribimos no porto serie os resultados
mensaxe = "Distancia: " + String(distancia) + " cm";
Serial.println(mensaxe);
delay(espera);
}

/*
 * Función para calcular a distancia en cm.
 *
 * A distancia medida é o doble da obtida
 * polo sensor (ida e volta).
 * As unidades serán en cm/us, a velocidade
 * do son é 340 m/s =
 * = 340 m/s * (1 s)/(1000000 us) * (100 cm)/(1 m) =
 * = 340 / 10000 cm/us = 0.034 cm/us
 */
float calcularDistancia(float t) {
    // v = 2*dist / tempo ==> dist = tempo*v/2
    // dist = tempo*0.034/2 = tempo*0.017 [cm]
    return t*0.017;
}

Compilación Terminada
```

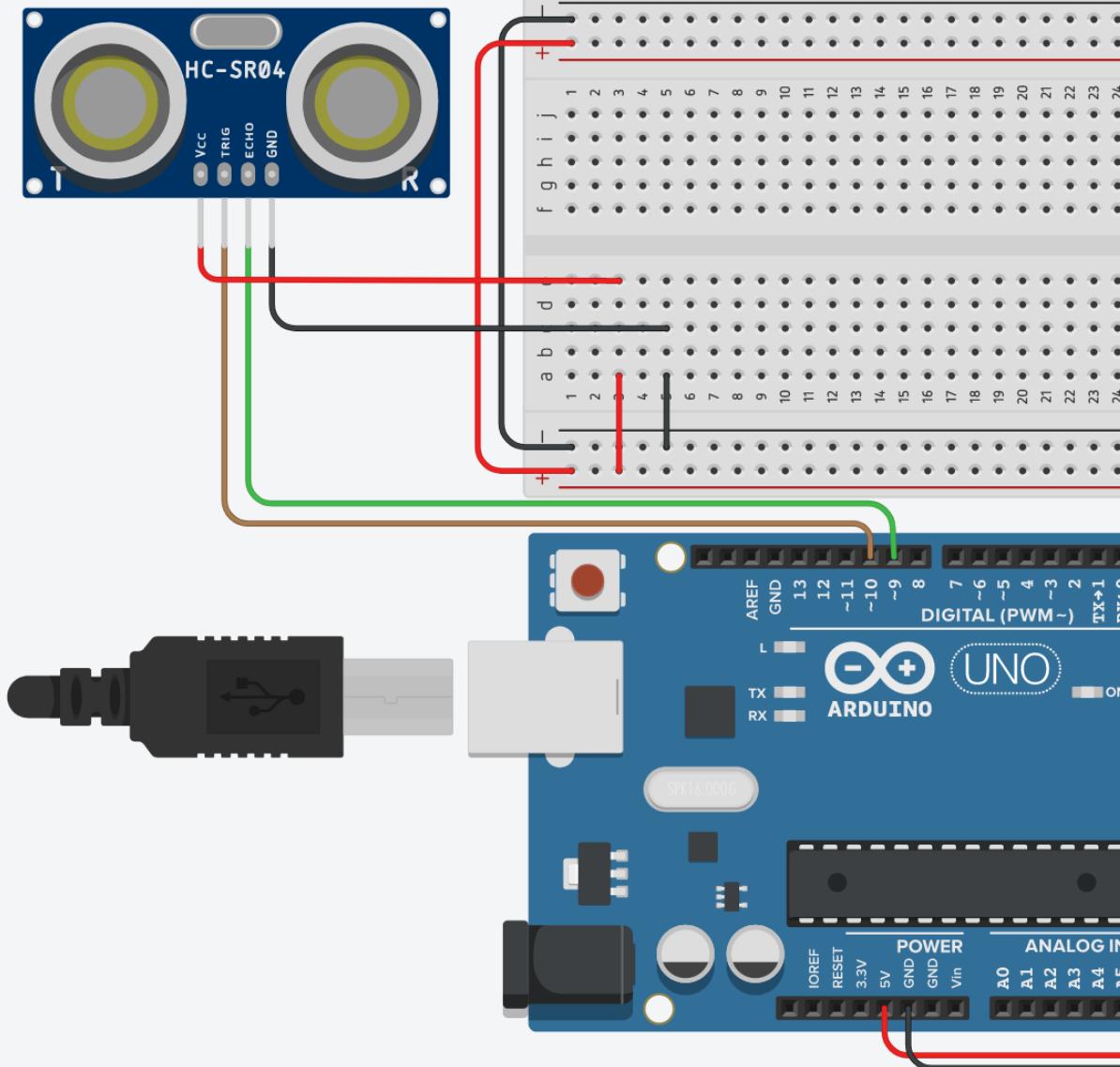
O rascunho usa 6132 bytes (19%) do espacio de almacenamiento
Variábeis globais usan 230 bytes (11%) de memoria dinâmica

51

Arduino/Genuino Uno em /dev/cu.wchusbserial1410

Víctor . Álvarez 201911

E/S dixitais – Sensor ultrasóns HC-SR04



```
/*
 * Script para medir a distancia
 * a un obxecto, empregando un
 * sensor de ultrasóns HC-SR04
 */

#define TRIGGER 9
#define ECHO 10
//Definimos macro para calcular distancia
#define calcDist(t) t*0.017 //en cm

float tempo, distancia;
String mensaxe = "";
int espera = 200;

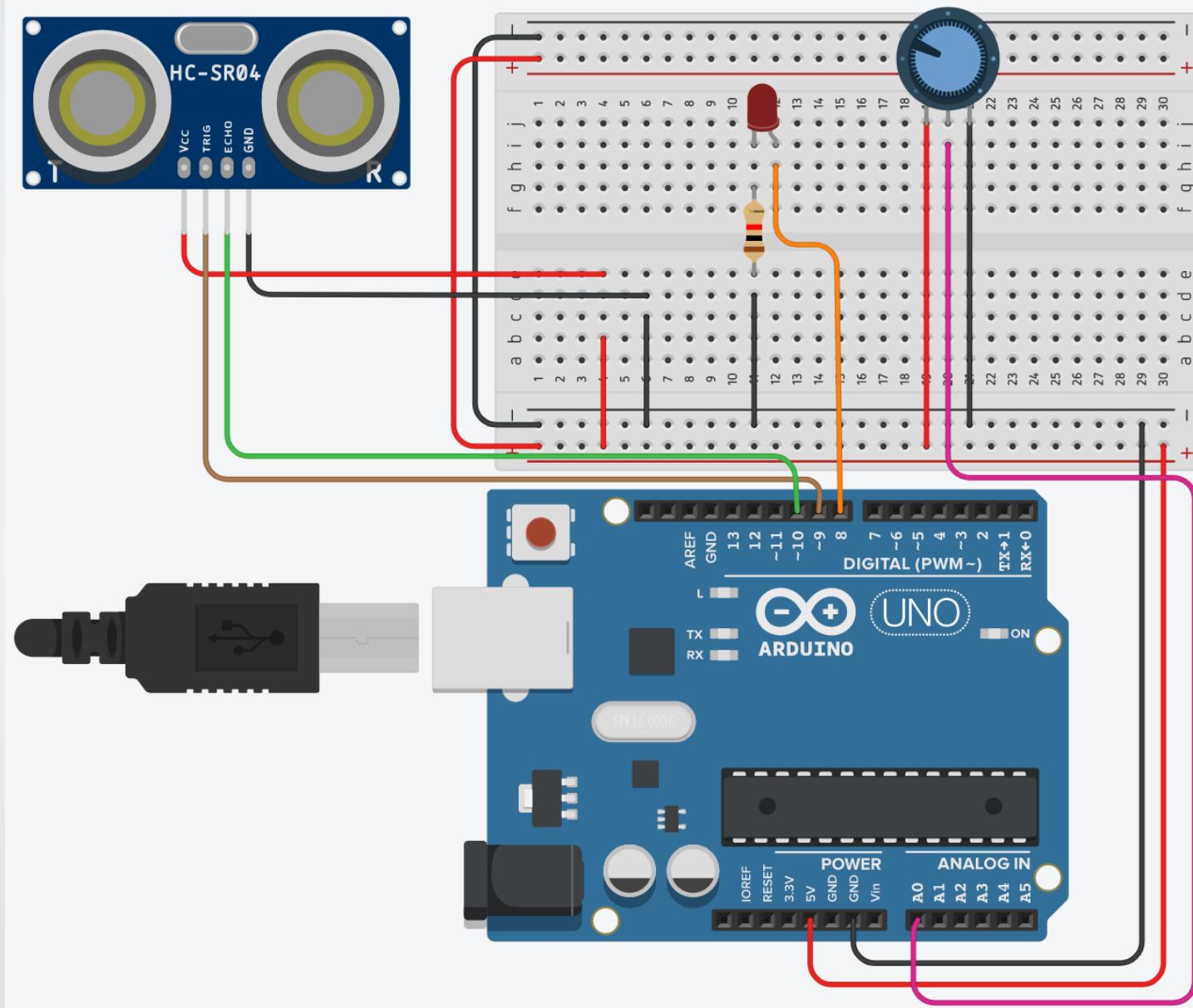
void setup() {
    Serial.begin(9600);
    pinMode(TRIGGER, OUTPUT);
    pinMode(ECHO, INPUT);
}

void loop() {
    //Enviamos un pulso co HC-SR04
    digitalWrite(TRIGGER, LOW);
    delayMicroseconds(5);
    digitalWrite(TRIGGER, HIGH);
    delayMicroseconds(25);
    digitalWrite(TRIGGER, LOW);
    tempo = pulseIn(ECHO, HIGH);

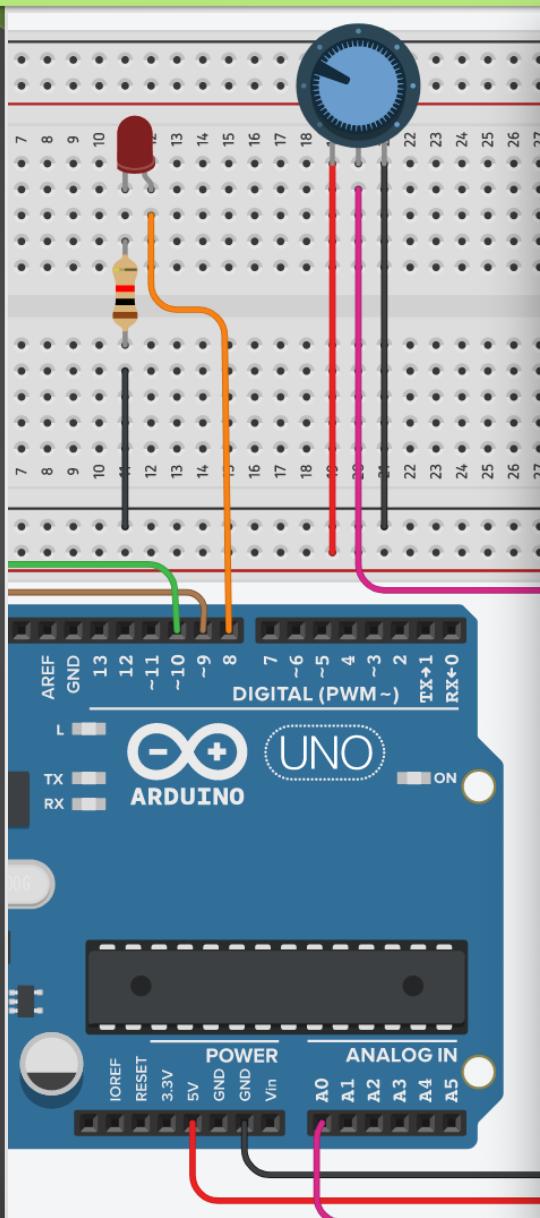
    //Calculamos a distancia en cm
    distancia = calcDist(tempo);

    //Escribimos no porto serie os resultados
    mensaxe = "Distancia: " + String(distancia) + " cm.";
    Serial.println(mensaxe);
    delay(espera);
}
```

E/S dixitais – Sensor ultrasóns HC-SR04



E/S dixitais – Sensor ultrasóns HC-SR04



```
sensor.distancia.ultrasons.con.potenciometro | Arduino 1.8.10
sensor.distancia.ultrasons.con.potenciometro

/*
 * Script para medir a distancia
 * a un obxecto, empregando un
 * sensor de ultrasóns HC-SR04.
 * A partir dunha distancia mínima
 * encéndese un LED; esta distancia
 * pódese regular cun potenciómetro
 */

#define TRIGGER 9
#define ECHO 10
//Definimos macro para calcular distancia
#define calcDist(t) t*0.017 //en cm

#define POTENCIOMETRO A0
#define LED 8

#define DIST_MIN 200

float tempo, distancia;
int valorPot, limite;
String mensaxe = "";
int espera = 400;

void setup() {
  Serial.begin(9600);
  pinMode(TRIGGER, OUTPUT);
  pinMode(ECHO, INPUT);
  pinMode(LED, OUTPUT);
}

void loop() {
  //Regulamos a distancia mínima para
  //que acenda o LED, co potenciómetro
  valorPot = analogRead(POTENCIOMETRO);
  limite = map(valorPot, 0, 1023, 0, DIST_MIN);

  //Enviamos un pulso co HC-SR04
  digitalWrite(TRIGGER, LOW);
  delayMicroseconds(5);
  digitalWrite(TRIGGER, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIGGER, LOW);
  tempo = pulseIn(ECHO, HIGH);

  //Calculamos a distancia en cm
  distancia = calcDist(tempo);

  //Encendemos LED se se supera a dist mínima
  if(distancia < limite) digitalWrite(LED, HIGH);
  else digitalWrite(LED, LOW);

  //Escribimos no porto serie os resultados
  mensaxe = "Distancia: " + String(distancia) + " cm";
  mensaxe += "\tLímite: " + String(límite) + " cm";
  Serial.println(mensaxe);
  delay(espera);
}
```

```
sensor.distancia.ultrasons.con.potenciometro | Arduino 1.8.10
sensor.distancia.ultrasons.con.potenciometro

#define DIST_MIN 200

float tempo, distancia;
int valorPot, limite;
String mensaxe = "";
int espera = 400;

void setup() {
  Serial.begin(9600);
  pinMode(TRIGGER, OUTPUT);
  pinMode(ECHO, INPUT);
  pinMode(LED, OUTPUT);
}

void loop() {
  //Regulamos a distancia mínima para
  //que acenda o LED, co potenciómetro
  valorPot = analogRead(POTENCIOMETRO);
  limite = map(valorPot, 0, 1023, 0, DIST_MIN);

  //Enviamos un pulso co HC-SR04
  digitalWrite(TRIGGER, LOW);
  delayMicroseconds(5);
  digitalWrite(TRIGGER, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIGGER, LOW);
  tempo = pulseIn(ECHO, HIGH);

  //Calculamos a distancia en cm
  distancia = calcDist(tempo);

  //Encendemos LED se se supera a dist mínima
  if(distancia < limite) digitalWrite(LED, HIGH);
  else digitalWrite(LED, LOW);

  //Escribimos no porto serie os resultados
  mensaxe = "Distancia: " + String(distancia) + " cm";
  mensaxe += "\tLímite: " + String(límite) + " cm";
  Serial.println(mensaxe);
  delay(espera);
}
```

E/S dixitais – Sensor ultrasóns HC-SR04

- Outra opción é empregar algunha das librerías que nos proporcione o fabricante ou a comunidade maker.
- Para o HC-SR04 existen varias. Imos refacer o script anterior para empregar a librería `<NewPing.h>` de Tim Eckel. Pódese consultar a documentación en:
<https://playground.arduino.cc/Code/NewPing/>
- Esta librería está programada como POO e proporciona un obxecto que nos aporta varios métodos (funcións) interesantes, p.ex.:
 - `sonar.ping_cm()` ou `sonar.ping_in()` - devolven a distancia en cm ou pulgadas respectivamente
 - `sonar.ping_median(15)` – envía 15 pings (ou o que poñamos), descarta os fóra de rango e devolve a mediana en μs
 - `sonar.ping_convert_cm(tempo)` ou `sonar.ping_convert_in(tempo)` – converte un tempo en μs a distancia en cm ou pulgadas
- Pódese instalar no IDE de Arduino seguindo Script > Incluir biblioteca

E/S dixitais – Sensor ultrasóns HC-SR04

The image shows two side-by-side screenshots of the Arduino IDE. Both windows have the title bar "sensor.distancia.ultrasons.con.potenciometro.e.libreria.newping | Ar...".

Left Window (Code View):

```
/*
 * Script para medir a distancia
 * a un obxecto, empregando un
 * sensor de ultrasóns HC-SR04.
 * A partir dunha distancia mínima
 * encéndese un LED; esta distancia
 * pódese regular cun potenciómetro.
 *
 * Inclúe a librería NewPing para
 * facilitar a codificación e lectura.
 */
#include <NewPing.h>

#define TRIGGER 9
#define ECHO 10

#define POTENCIOMETRO A0
#define LED 8

#define DIST_MIN 20
#define DIST_MAX 200

unsigned int |distancia;
int valorPot, limite;
String mensaxe = "";
int espera = 400;

NewPing sensor(TRIGGER, ECHO, DIST_MAX);

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

void loop() {
  distancia = sensor.ping_cm();

  //Regulamos a distancia mínima para
  //que acenda o LED, co potenciómetro
  valorPot = analogRead(POTENCIOMETRO);
  limite = map(valorPot, 0, 1023, 0, DIST_MIN);

  //Encendemos LED se se supera a dist mínima
  if(distancia < limite) digitalWrite(LED, HIGH);
  else digitalWrite(LED, LOW);

  //Escribimos no porto serie os resultados
  mensaxe = "Distancia: " + String(distancia) + " cm";
  mensaxe += "\tLímite: " + String(limite) + " cm";
  Serial.println(mensaxe);
  delay(espera);
}
```

Right Window (Code View):

```
#include <NewPing.h>

#define TRIGGER 9
#define ECHO 10

#define POTENCIOMETRO A0
#define LED 8

#define DIST_MIN 20
#define DIST_MAX 200

unsigned int |distancia;
int valorPot, limite;
String mensaxe = "";
int espera = 400;

NewPing sensor(TRIGGER, ECHO, DIST_MAX);

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

void loop() {
  distancia = sensor.ping_cm();

  //Regulamos a distancia mínima para
  //que acenda o LED, co potenciómetro
  valorPot = analogRead(POTENCIOMETRO);
  limite = map(valorPot, 0, 1023, 0, DIST_MIN);

  //Encendemos LED se se supera a dist mínima
  if(distancia < limite) digitalWrite(LED, HIGH);
  else digitalWrite(LED, LOW);

  //Escribimos no porto serie os resultados
  mensaxe = "Distancia: " + String(distancia) + " cm";
  mensaxe += "\tLímite: " + String(limite) + " cm";
  Serial.println(mensaxe);
  delay(espera);
}
```

Bottom Status Bar:

Carregamento completo

O rascunho usa 4722 bytes (14%) do espaço de armazéno rascunho usa 4722 bytes (14%) do espaço de armazenamento do programVariábeis globais usam 256 bytes (12%) de memória Variáveis globais usam 256 bytes (12%) de memória dinâmica, restando 1

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

E/S dixitais – outros sensores dixitais

- Existe unha gran variedade de sensores dixitais que se poden acoplar a unha placa Arduino:
 - sensores magnéticos como os que se montan nos detectores de apertura de portas ou fiestras, para alarmas:
https://en.wikipedia.org/wiki/Reed_relay
<https://www.luisllamas.es/esar-un-interruptor-magnetico-con-arduino-magnetic-reed/>
 - sensores de inclinación:
<https://www.arduino.cc/en/Main/TutorialTiltSensor>
 - sensores de presencia PIR:
<https://www.instructables.com/id/PIR-Motion-Sensor-Tutorial/>
<https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/how-pirs-work>
- Realiza a montaxe dun PIR na placa de entrenamento e un script que o controle, de xeito que ao detectar presenza encenda un LED.
- Investiga na documentación adxunta e propón unha posible solución.

Entradas dixitais

- Nesta unidade aprendemos a:
 - Montar e manexar un sensor de distancia por ultrasóns HC-SR04,
 - usar a medida do sensor para dar avisos luminosos e incluso regular o umbral de salto do aviso (potenciómetro)
 - Empregamos as funcións `pulseIn()` para medir a anchura temporal dun pulso, así como `delayMicroseconds()`
 - Comparar as nosas solucións con outras proporcionadas pola librería de Arduino
 - Instalar e usar outras librerías proporcionadas pola comunidade ou as casas comerciais
 - Usar a saída serie para mostrar mensaxes do script