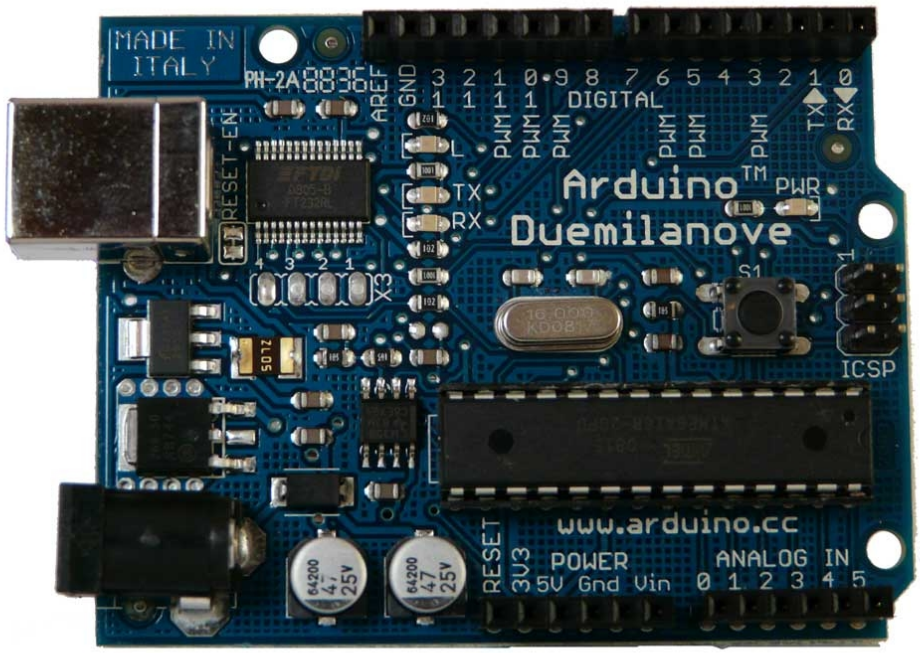
PRACTICAS CON ARDUINO



1º DE BACHILLERATO

PROYECTO INTEGRADO: TALLER DE ROBÓTICA

PRÁCTICA Nº 1: PARPADEO DE UN LED

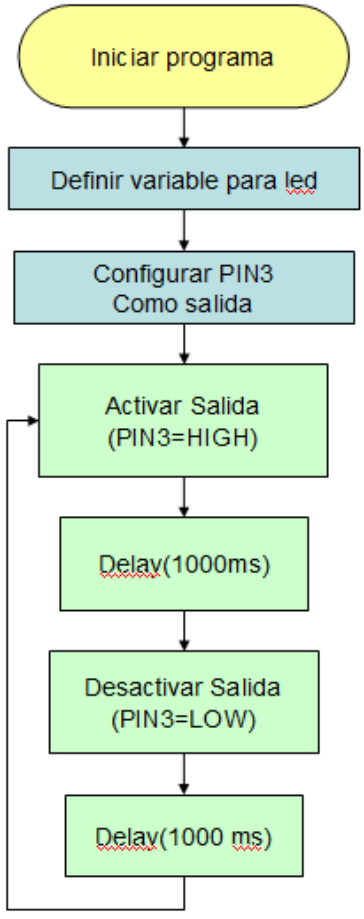
Materiales:
Placa Protoboard
Microcontroladora Arduino
Cable de conexión
1 LED
1 resistencia 120 Ω
Cables

Objetivo
Con esta práctica se pretende explicar al alumno el esquema de programación de la microcontroladora, el montaje de circuitos sencillos y la estructura de la placa Arduino, el programa es sencillo y consiste en hacer parpadear un LED con una frecuencia de 1 segundo
El alumno debe aprender:

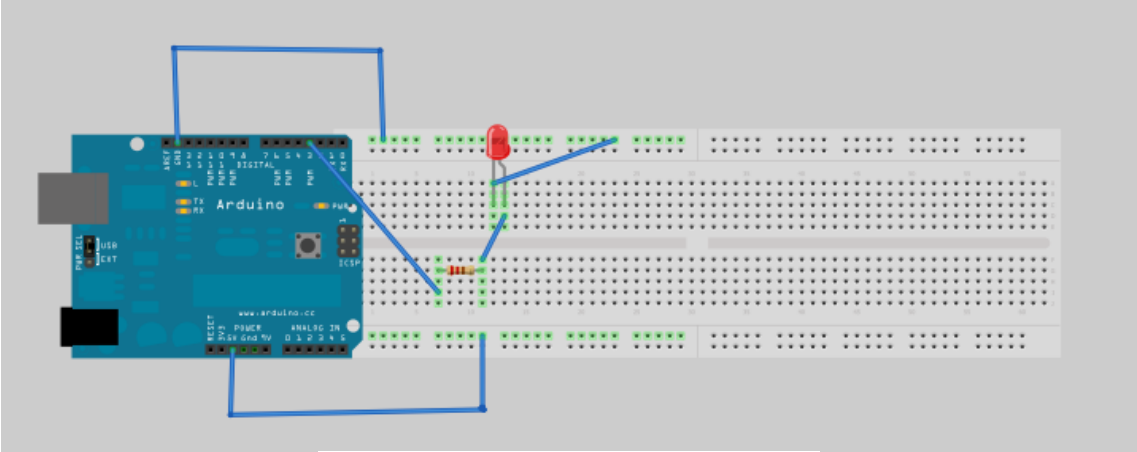
1. Definir variables *int*
2. Inicializar las variables de salida *pinMode(variable,OUTPUT);*
3. Hacer bucles para que se repita la secuencia *void loop()*
4. Función para hacer que funcione el LED *digitalWrite(variable,HIGH);*
5. Función para hacer que no funcione el LED *digitalWrite(variable,LOW);*
6. Función para mantener una variable en un estado durante un tiempo *delay(tiempo en ms)*

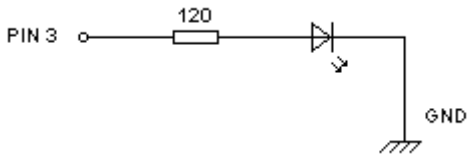
Diagrama de flujo

Parpadeo de un led con una frecuencia de 1 segundo



Montaje





Programa:

```
/*
  Parpadeo:  Enciende un LED un segundo, entonces se apaga 1 segundo y así
sucesivamente. El LED está conectado al Pin 3 y lleva  en serie una resistencia de
protección cuyo valor podrá estar entre 120 Ω y 220 Ω.
 */

int ledPin =  3;    // LED conectado a  pin 3

void setup()
{
  // inicializa el pin 3 como salida
  pinMode(ledPin, OUTPUT);
}

// ahora hacemos un bucle

void loop()
{
  digitalWrite(ledPin, HIGH);   // enciende el LED
  delay(1000);                  // espera un segundo
  digitalWrite(ledPin, LOW);    // apaga el LED
  delay(1000);                  // espera un segundo
}
```

Actividades:

1. Con el mismo montaje varia el tiempo de encendido y apagado del LED, ahora debe ser de 2 segundos
2. Con el mismo montaje, el LED debe estar encendido 1 segundo y apagado 3 segundos.
3. Cambia el pin donde conectamos el LED al pin 6, modifica el programa para que funcione adecuadamente.
4. Añade en serie un potenciómetro ¿Qué ocurre cuando giras el potenciómetro?

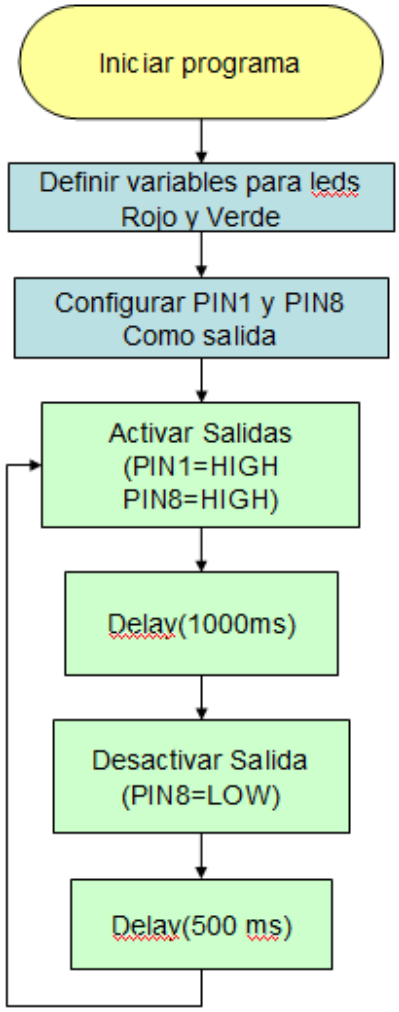
PRÁCTICA Nº 2: PARPADEO DE VARIOS LED

Materiales:
Placa protoboard
Microcontroladora Arduino
Cable de conexión
1 LED Rojo
1 LED Verde
2 Resistencias de 120 Ω
Cables

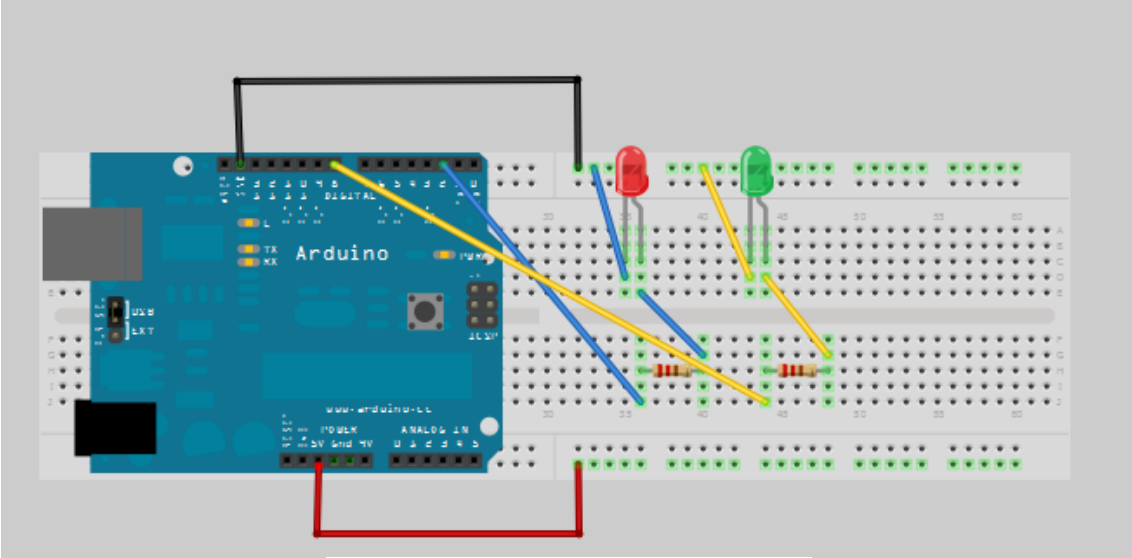
Objetivo
Ahora tenemos 2 LEDs. Queremos que el LED rojo esté siempre encendido y que el amarillo esté 1 segundo encendido y medio segundo apagado.
Se complica el montaje y se debe introducir una nueva variable.

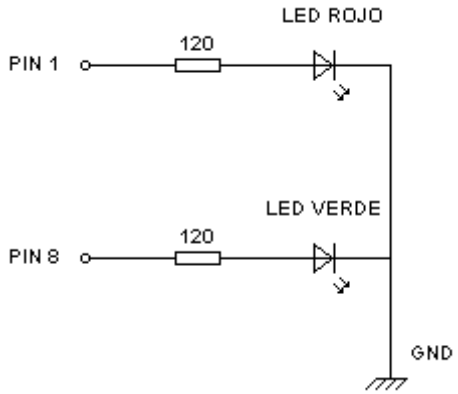
Diagrama de flujo

Un led siempre encendido y otro led se enciende 1 segundo y se apaga 0,5 segundos.



Montaje





Programa

```
/*
  Parpadeo. Practica 2
  Un led siempre encendido, rojo
  Un segundo led verde se enciende 1 seg y se apaga 0,5 segundos
  */

int ledrojo =  1;    // LED rojo conectado al  pin 1
int ledverde=8;     // LED verde conectado al pin 8

void setup()
{
  //ledrojo y ledverde son variables de salida
  pinMode(ledrojo, OUTPUT);
  pinMode(ledverde,OUTPUT);
}

void loop()
{
  digitalWrite(ledrojo, HIGH);   // enciende el led rojo
  digitalWrite(ledverde,HIGH);   //enciende el led verde
  delay(1000);                   // espera un segundo
  digitalWrite(ledverde, LOW);    // apaga el led verde
  delay(500);                  // espera medio segundo
}
```

Actividades

1.- Cambia el programa para que se mantenga encendido el LED verde y parpadea el rojo.

2.- Cambia el programa para que el LED rojo esté encendido 1,5 segundos y apagado 0,8 segundos.

3.- Cambia el programa para que cuando el LED rojo este encendido el LED verde esté apagado y cuando el LED rojo esté apagado el LED verde esté encendido. Frecuencia de 1 segundo.

4.- Monta un LED amarillo y modifica el programa para que se enciendan los LED con la siguiente secuencia Verde-Amarillo-Rojo

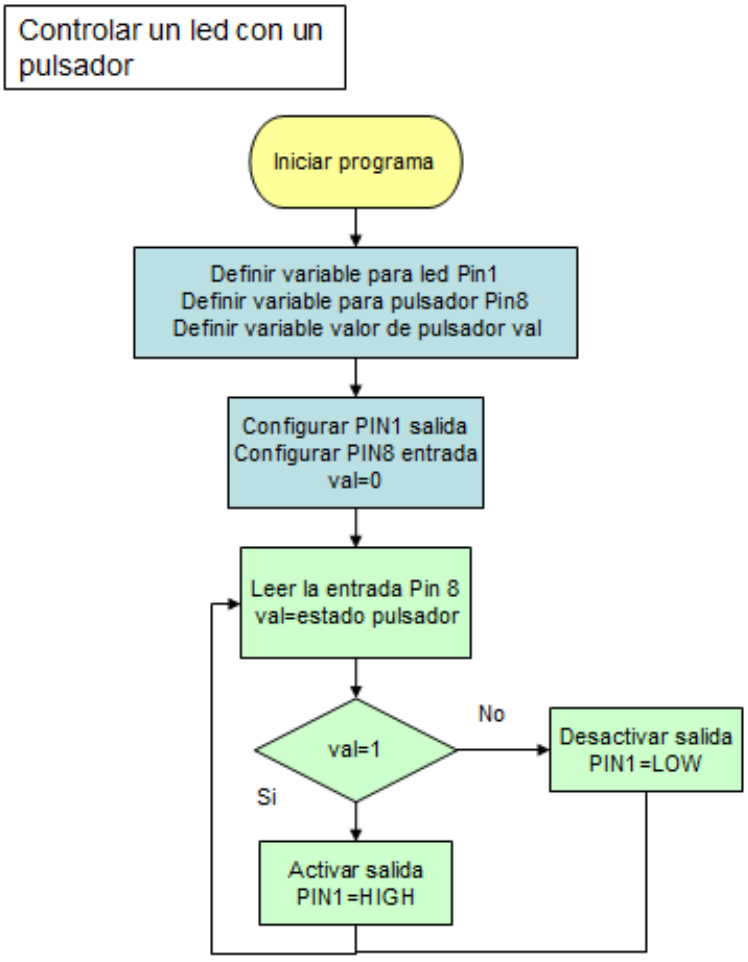
5.- Los LED deben encenderse con la secuencia siguiente: Verde-Amarillo-Rojo esperar 2 segundos todos apagados Rojo-Amarillo-Verde esperar 2 segundos todos encendidos.

PRÁCTICA Nº 3: CONTROL DE UN LED CON UN PULSADOR

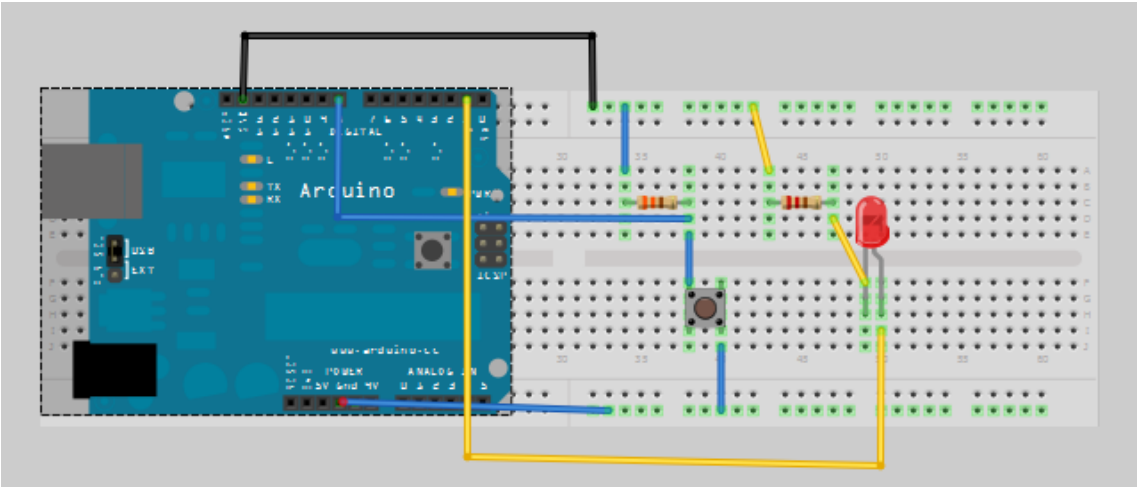
Materiales:
Placa protoboard
Microcontroladora Arduino
Cable de conexión
1 LED Rojo
1 Pulsador
1 Resistencia de 120 Ω
1 Resistencia de 150 kΩ
Cables

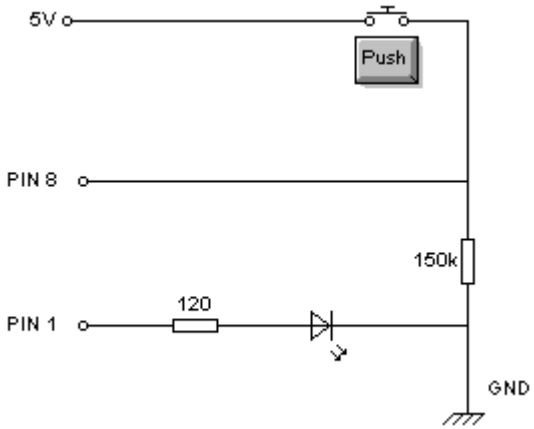
Objetivo
El LED rojo se debe encender cuando accionamos el pulsador.
En esta práctica introducimos los siguientes conceptos:
- Variable *val (donde se almacena el estado del pulsador)*
- Inicializar una variable de entrada de información *pinMode(variable,INPUT);*
- Función que nos dice el estado del interruptor *digitalRead(variable);*
- Función que permite actuar de una forma cuando el interruptor está cerrado y de otra forma cuando el interruptor está abierto *if (val==HIGH){ } else{}*

Diagrama de flujo



Montaje





Programa

```
/*El objetivo de este programa es controlar el encendido y apagado de un led
*Si el pulsador está pulsado se enciende el LED si no está apagado */

int pulsador=8;
int led=1;
int val=0;

void setup(){
  pinMode(pulsador,INPUT);
  pinMode(led,OUTPUT);
}

void loop(){
  val=digitalRead(pulsador);
  if(val==HIGH){
    digitalWrite(led,HIGH);
  }
  else{
    digitalWrite(led,LOW);
  }
}
```

Actividades

1.- Cambia el programa para que cuando el pulsador está pulsado el LED se apaga y cuando está sin pulsar el LED está encendido

2.- Pon un LED verde y varia el programa para que cuando acciones el pulsador el LED rojo esté encendido y el LED verde apagado, al dejar de pulsar el LED rojo se apaga y se enciende el LED verde

3.- Al presionar un pulsador 1 debe encenderse el LED rojo y al presionar el pulsador 2 debe apagarse el LED rojo y encenderse el verde.

4.- Hacer un programa con tres LED que se enciendan Verde-Amarillo-Rojo y cuando pulsemos el pulsador que cambie la secuencia a Amarillo-Rojo-Amarillo-Verde

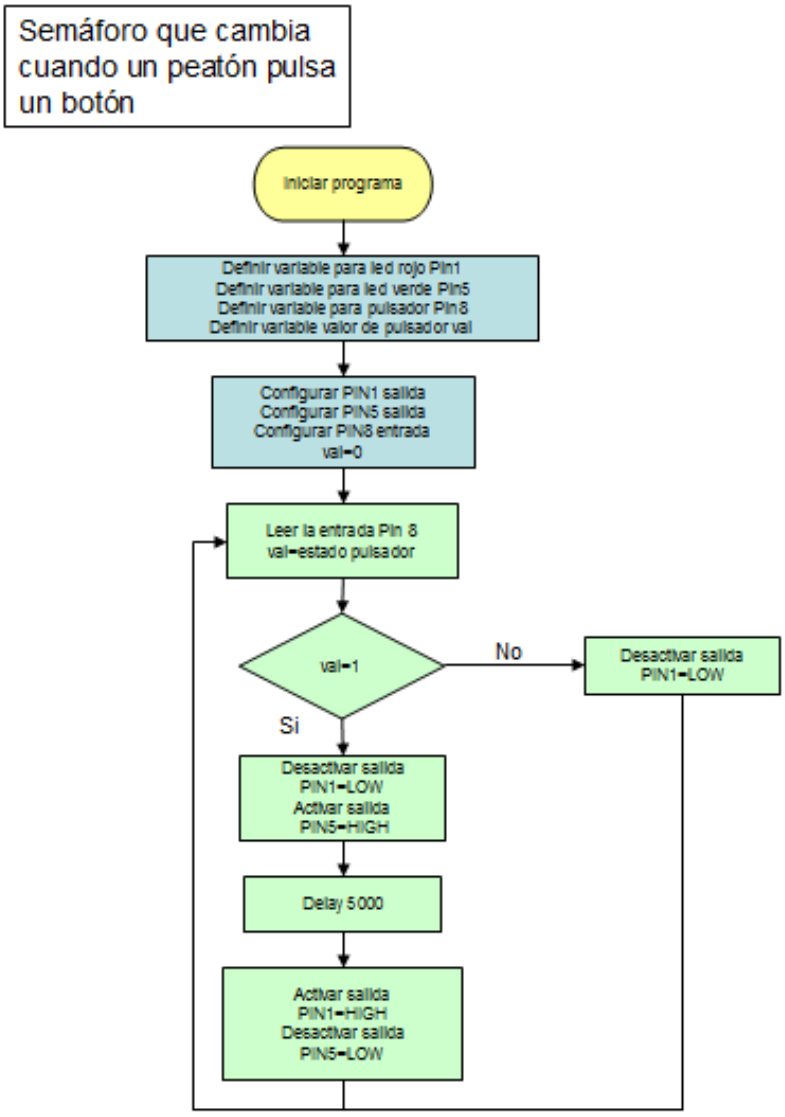
5.- Cuando pulsemos el pulsador la secuencia de dos LED Verde y Rojo va mas rápido

PRÁCTICA 4. SEMÁFORO PARA PEATONES

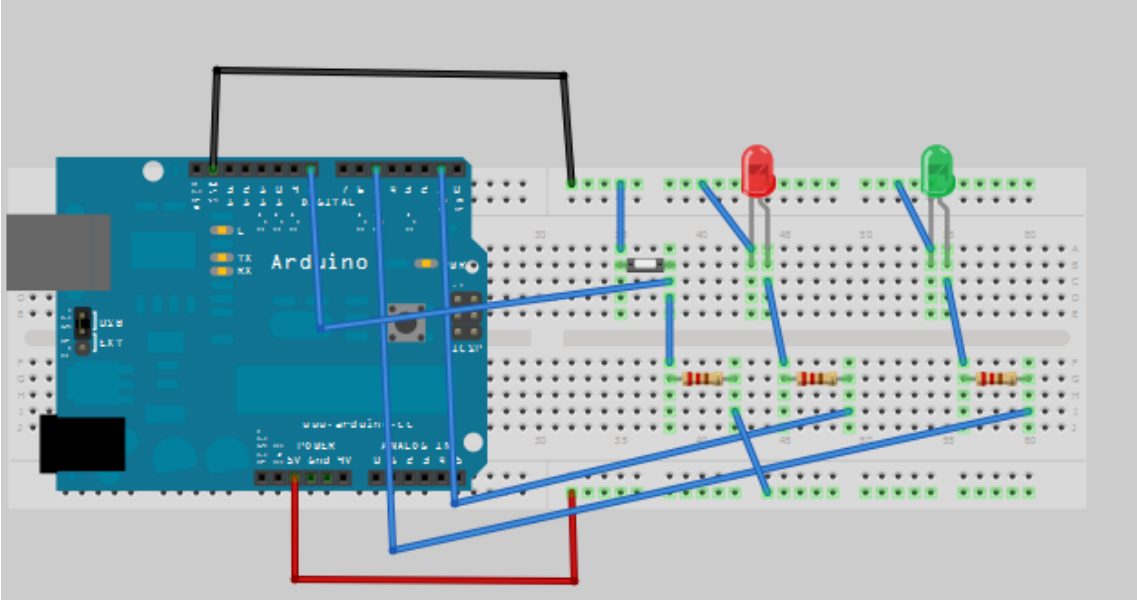
Materiales:
Placa protoboard
Microcontroladora Arduino
Cable de conexión
1 LED Rojo
1 LED Verde
1 Pulsador
2 Resistencias de 120 Ω
1 Resistencia de 150 kΩ
Cables

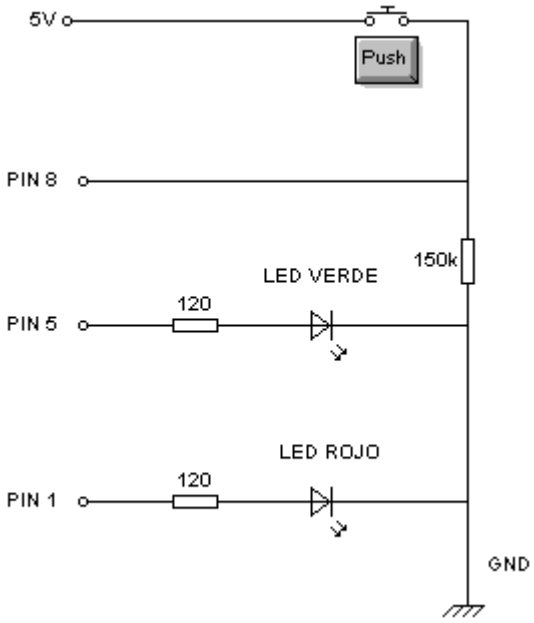
Objetivo
El LED rojo se debe encuentra encendido indicando que el peatón no puede pasar, cuando el peatón llega al semáforo pulsa el botón el led rojo se apaga y se enciende el verde durante cinco segundos.

Diagrama de flujo



Montaje


Arduino


5V
Push
PIN 8
150k
LED VERDE
120
PIN 5
LED ROJO
120
PIN 1
GND

Programa

```
/* El semaforo está rojo para el peatón
*Cuando el peatón pulsa el botón el LED rojo se apaga y se enciende el verde
*El LEd verde se  mantiene encendido 5 segundos y luego se enciende el rojo
*/

int interruptor=8;
int ledrojo=1;
int ledverde=5;
int val=0;

void setup(){
  pinMode(interruptor,INPUT);
  pinMode(ledrojo,OUTPUT);
  pinMode(ledverde,OUTPUT);
}

void loop(){
  val=digitalRead(interruptor);
  if(val==HIGH){
    digitalWrite(ledrojo,LOW);
    digitalWrite(ledverde,HIGH);
    delay(5000);
    digitalWrite(ledverde,LOW),
    digitalWrite(ledrojo,HIGH);
  }
  else{
    digitalWrite(ledrojo,HIGH);
  }
}
```

Actividades

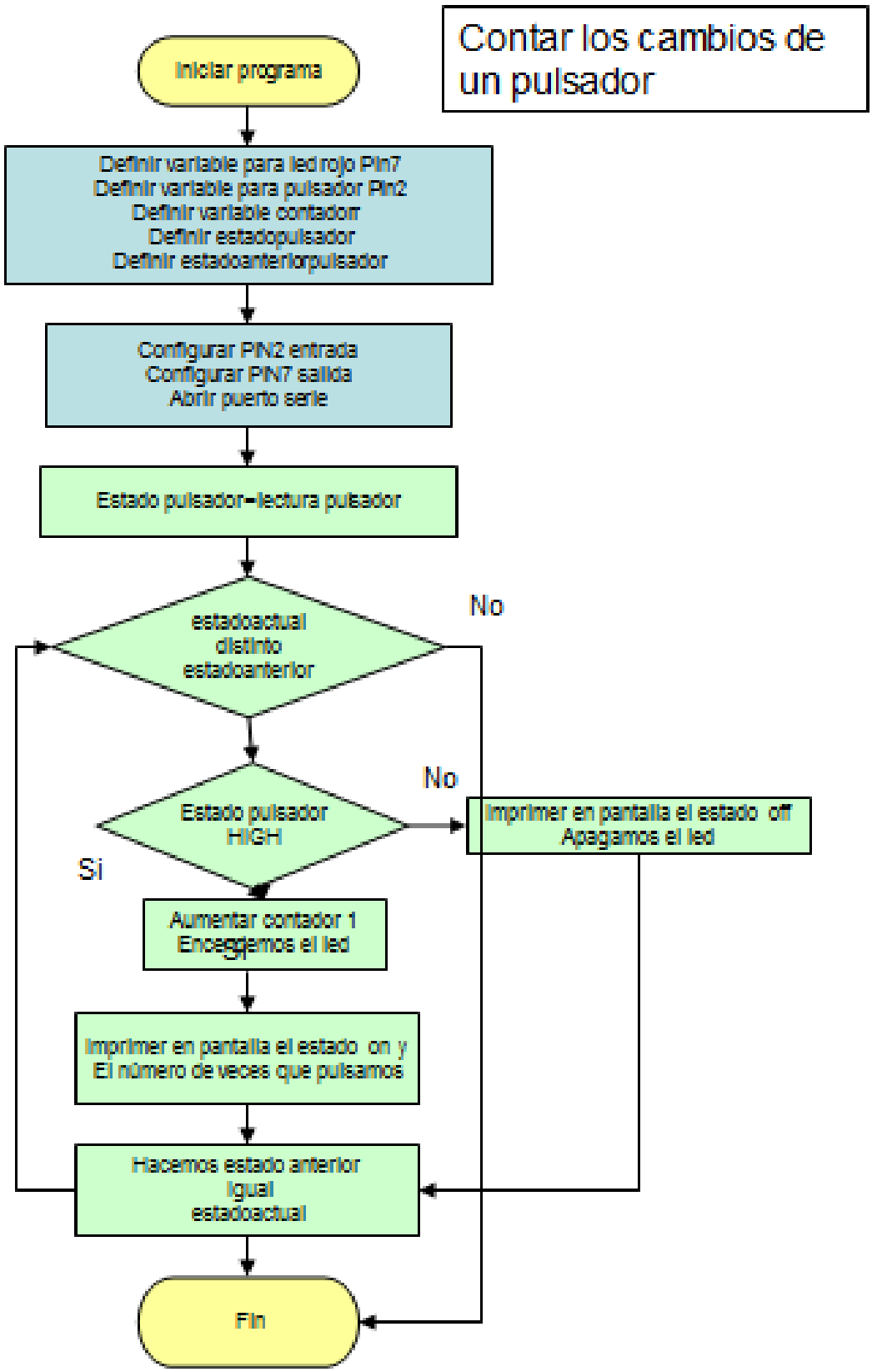
1.- Cambia el programa para que cuando pulsamos el interruptor el LED rojo tarde un segundo en apagarse para evitar que los coches tengan que dar un frenazo

PRÁCTICA Nº 5: CONTANDO EVENTOS.
CONTAR LAS VECES QUE PULSAMOS UN PULSADOR

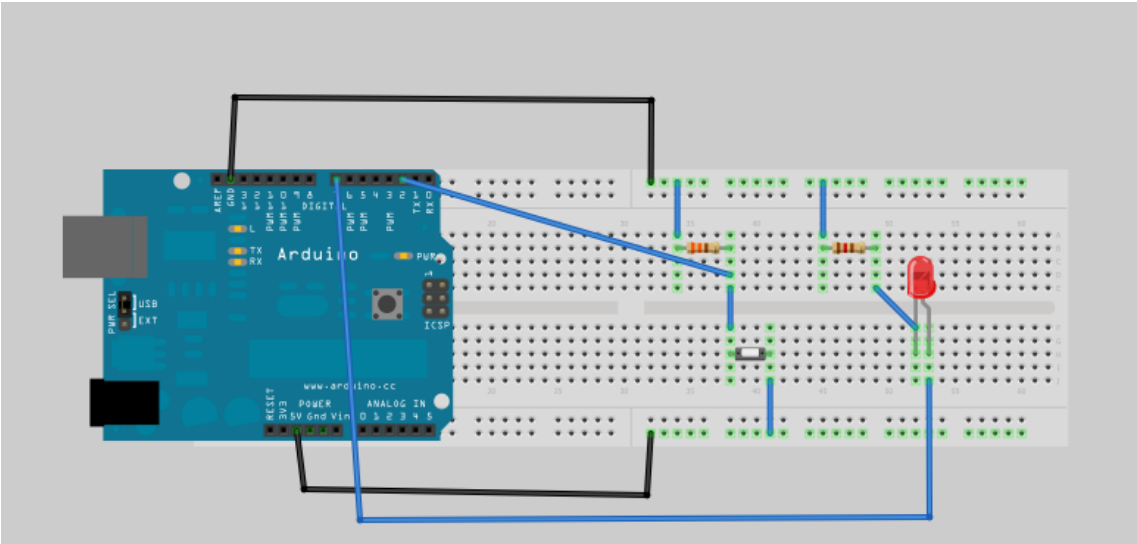
Materiales:
Placa protoboard
Microcontroladora Arduino
Cable de conexión
1 LED Rojo
1 Pulsador
1 Resistencias de 120 Ω
1 Resistencia de 150 kΩ
Cables

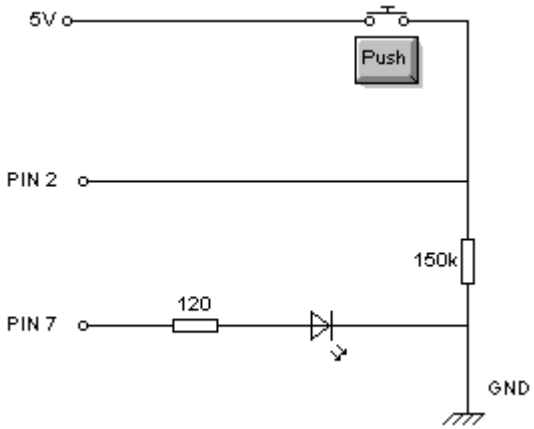
Objetivo
Cada vez que accionamos el pulsador, el LED rojo se enciende y además aparece en pantalla el número de veces que pulsamos. Esto nos serviría para contar eventos, por ejemplo, número de veces que se abre una puerta, personas que entran en un local, etc.
En esta práctica introducimos los siguientes conceptos:
· Serial.begin(9600) abre el puerto serie
· ++ aumenta uno; != distinto; == igual
· Serial.println() imprime datos con salto de línea
· Serial.print() imprime datos

Diagrama de flujo



Montaje





Programa

```
/*
  Este programa nos permite conocer las veces que cambiamos una entrada de
un estado a otro.
  Encendido o apagado
  */


int  pulsadorPin = 2;    // el pin donde hemos conectado el pulsador
int ledPin = 7;       // el pin donde hemos conectado el led
int contador = 0;   // cuenta el numero de veces que pulsamos
int estadopulsador = 0;         // estado actual del pulsador
int estadoanteriorpulsador = 0;     // estado anterior del pulsador

void setup() {
   pinMode(pulsadorPin, INPUT); // pulsador es una entrada
   pinMode(ledPin, OUTPUT);//el LED es una salida
  Serial.begin(9600);//abre puerto serie
}


void loop() {
  // read the pushbutton input pin:
  estadopulsador = digitalRead(pulsadorPin);//nos dice si el pulsador está
pulsado o no

    if (estadopulsador != estadoanteriorpulsador) {//compara el estado actual con
el anterior
    // si el estado cambia aumenta el contador
    if (estadopulsador == HIGH) {
          contador++;//contador aumenta
          digitalWrite(ledPin,HIGH);//enciende el led
      Serial.println("on"); //imprime datos al puerto serie con retorno de carro
      Serial.print("numero de veces que pulsamos:  "); //imprime puerto serie sin
retorno de carro
      Serial.println(contador, DEC);
    }
    else {
          digitalWrite(ledPin,LOW);   //apaga el led
          Serial.println("off");
    }
    estadoanteriorpulsador = estadopulsador;//salva el estado actual como
último estado
  }

}
```

Actividades

1.- Cambiar el programa para que el LED se encienda cuando se pulsa cuatro veces

2.- ¿Cómo podemos conseguir que cuente hasta 10 y se reinicialice el programa?

Nota: la actividad nº 2 es lo suficientemente complicada como para considerarla un reto. Así pues:

- Si consigue solucionarla una sola persona, tendrá 1,5 puntos más en la evaluación.
- Si consiguen solucionarla **2 personas del mismo grupo**, cada una de ellas tendrá **1 punto** más en la evaluación.
- Si consiguen solucionarla **3 personas del mismo grupo**, cada una de ellas tendrá **0,8 puntos** más en la evaluación.
- Si consiguen solucionarla 2 o más personas de distinto grupo, tan sólo os daré 0,1 puntos más en la evaluación a quienes lo averigüen: recordad que cada grupo es un equipo de investigadores, y que hay que tratar de impedir el espionaje industrial...

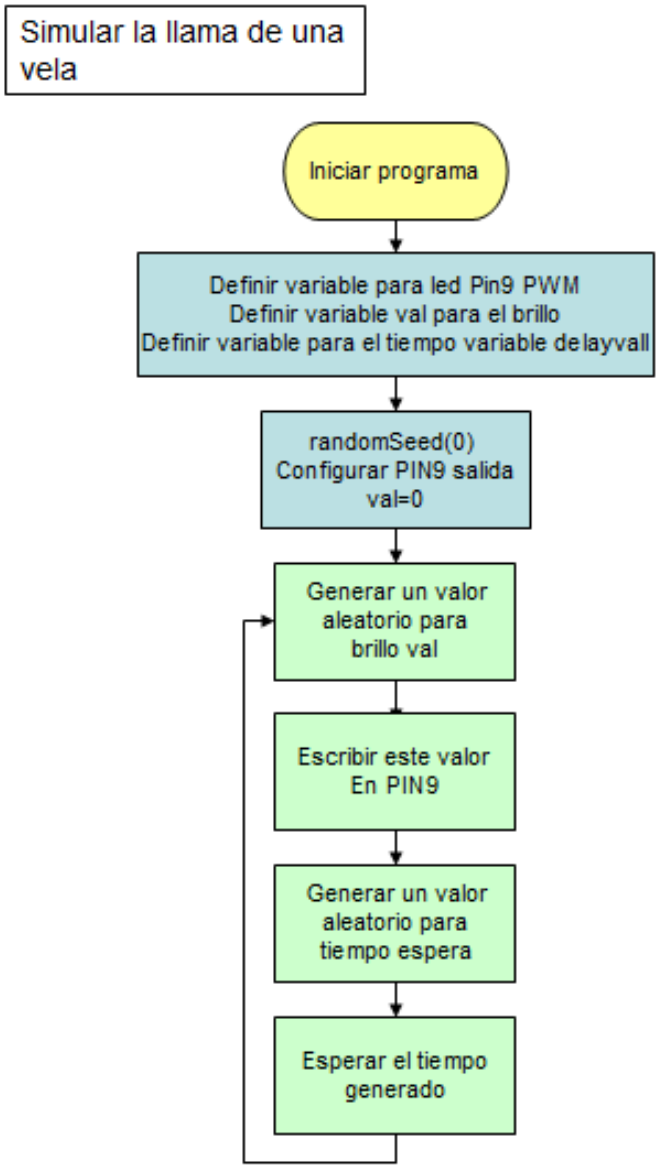
Tenéis de tiempo para pensarlo desde el momento en el que leáis estas líneas hasta la semana siguiente a aquella en la que se plantee en clase realizar esta práctica.

PRACTICA 6. SIMULAR LUZ DE UNA VELA

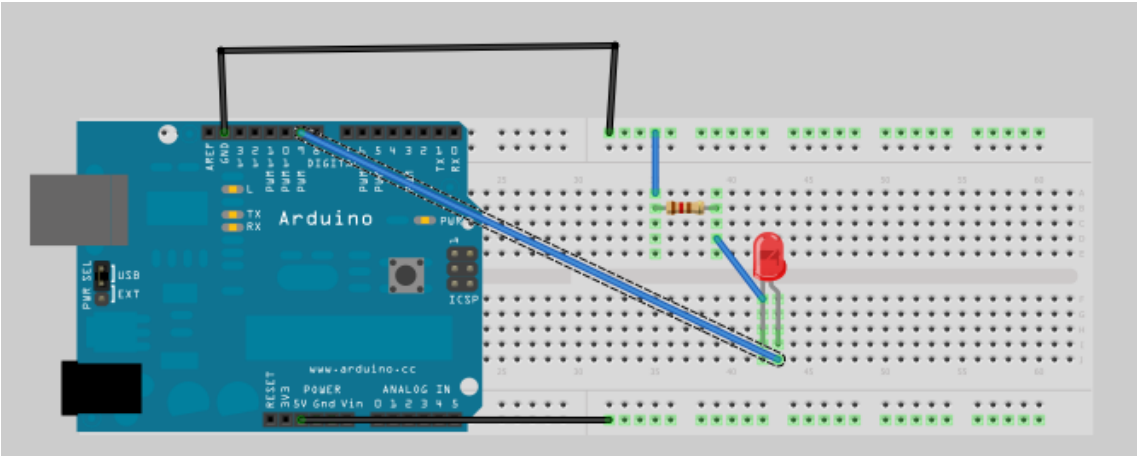
Materiales:
Placa protoboard
Microcontroladora Arduino
Cable de conexión
1 LED Rojo
1 Resistencia de 120 Ω
Cables

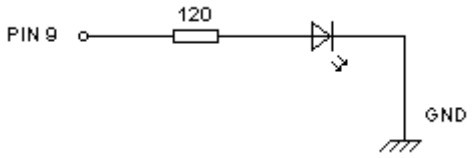
Objetivo
Se trata de simular el movimiento de la llama de una vela. Introducimos las entradas PWM e introducimos también una instrucción para generar un número aleatorio.
randomSeed(0) Inicializa los números aleatorios a cero
random(100,255) genera un número aleatorio entre 100 y 255

Diagrama de flujo



Montaje





Programa

```
/*
* Para simular la luz de la vela necesitamos:
* Saca por una de las salidas del puerto PWM un valor aleatorio que activa un LED
*/

int ledPin = 9; // selecciona el puerto PWM
int val = 0; // define y pone a cero la variable "brillo"
int delayval = 0; // define el intervalo de cambio de valor de salida

void setup() {
randomSeed(0); // inicializa el generador de números aleatorios
pinMode(ledPin, OUTPUT); // declara el pin de SALIDA pin 9
}

void loop() {
val = random(100,255); // genera un número aleatorio entre 100 y 255 que asigna a la variable val
analogWrite(ledPin, val); // envía ese valor a la salida pin 9
delayval = random(50,150); // genera un numero aleatorio entre 50 y 150 y lo asigna a la variable de temporización
delay(delayval); // espera un tiempo delayval medido en milisegundos
}
```