

Taller de Arduino – Ejercicios

1. Escribir un programa que encienda un LED cuando un pulsador este presionado y lo apague si se libera. El pulsador izquierdo actúa sobre el LED rojo y el pulsador derecho actúa sobre el LED verde.
2. Modificar el programa anterior para que además haga parpadear el LED amarillo cada un segundo (1 seg. encendido y 1 seg. apagado).
(Nota: ver el ejemplo del IDE Arduino `BlinkWithoutDelay`)
3. Escribir un programa que haga parpadear un LED variando la frecuencia del parpadeo con el potenciómetro.
(Nota: ver los ejemplos `AnalogReadSerial`)
4. Escribir un programa que encienda el LED verde con diferente intensidad ajustable con el potenciómetro. Tener en cuenta que el valor leído desde el ADC es de 10 bits y el valor a escribir en la salida PWM es de 8 bits, por lo que se debe aplicar un factor de escala.
(Nota: ver los ejemplos `AnalogReadSerial` y `Fade`)
5. Escribir un programa que envíe por puerto serie (Monitor Serial) el estado de los pulsadores. Enviar de forma continua una línea de texto con el estado de los tres pulsadores, similar a:
111
110
011
000
111
6. Escribir un programa que cambie el estado de los LEDs al recibir una letra por puerto serie:
 - Al recibir la letra 'v' se debe conmutar el estado del LED verde
 - Al recibir la letra 'a' se debe conmutar el estado del LED amarillo
 - Al recibir la letra 'r' se debe conmutar el estado del LED rojo(Nota: utilizar la función `Serial.readBytes()`)

Entradas y salidas digitales

`pinMode(pin, mode)`

- `pin`: número de pin al cual se quiere configurar el modo
- `mode`: INPUT o OUTPUT

`digitalWrite(pin, value)`

`value = digitalRead(pin)`

- `pin`: número de pin
- `value`: valor del pin, HIGH o LOW

Funciones de temporización

`delay(ms)`

Pausa el programa un tiempo (en milisegundo) especificado por el parámetro.

- `ms`: unsigned long

`time = millis()`

Devuelve la cantidad de milisegundos desde el inicio del programa

- `time`: unsigned long

Entrada y salida analógica (salida PWM)

`value = analogRead(pin) // 0 - 1023`

`analogWrite(pin, value)`

- `value`: 0 a 255

Puerto de comunicación serie

`Serial.begin(speed)`

- `speed`: bits por segundo (long)

`Serial.println(val)`

`Serial.println(val, format)`

- `val`: valor a imprimir – cualquier tipo de dato
- `format`: especifica la base numérica (DEC, HEX, OCT, BIN)

`Serial.write(val)`

`Serial.write(str)`

`Serial.write(buf, len)`

- `val`: valor a enviar como único byte
- `str`: cadena a enviar como serie de bytes
- `len`: cantidad de bytes a enviar desde un arreglo

`Serial.readBytes(buffer, length)`

- `buffer`: donde se almacenaran los datos leídos (`char []` o `byte []`)
- `length`: cantidad de bytes a leer (int)