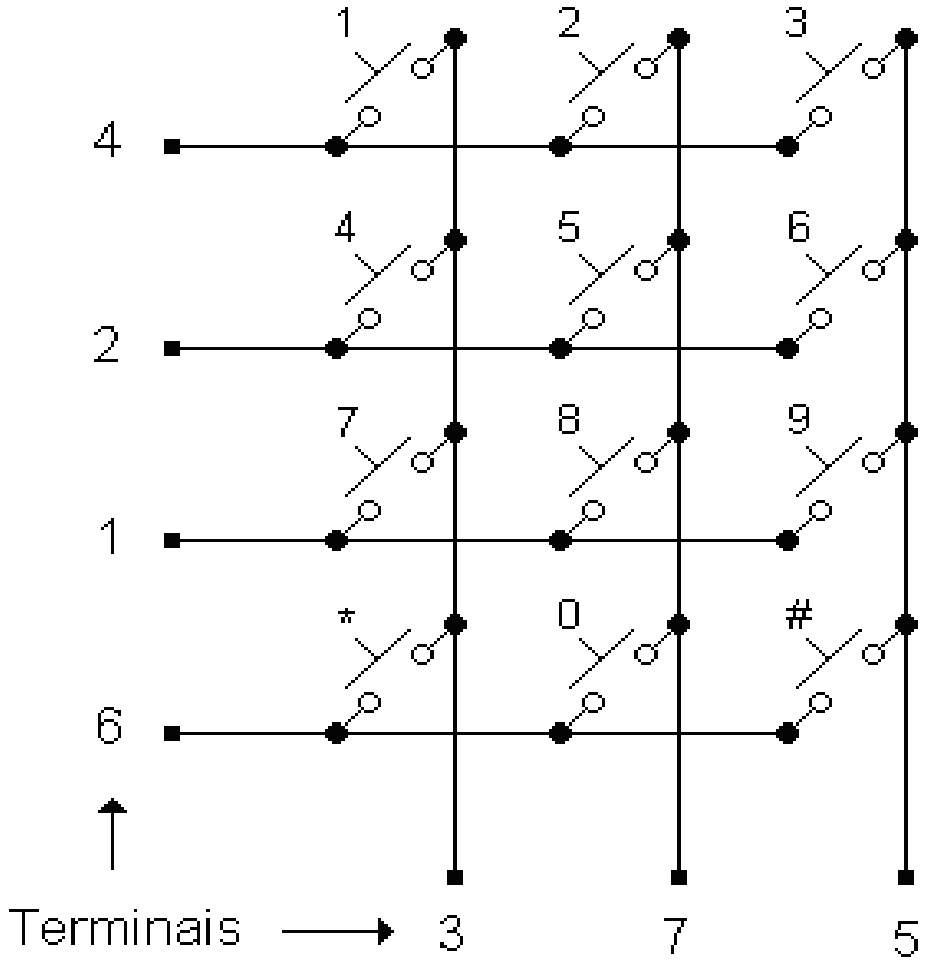
Teclado Matricial, Memoria EEPROM y Generación De Señal PWM

Ángelo Joseph Soto Vergel

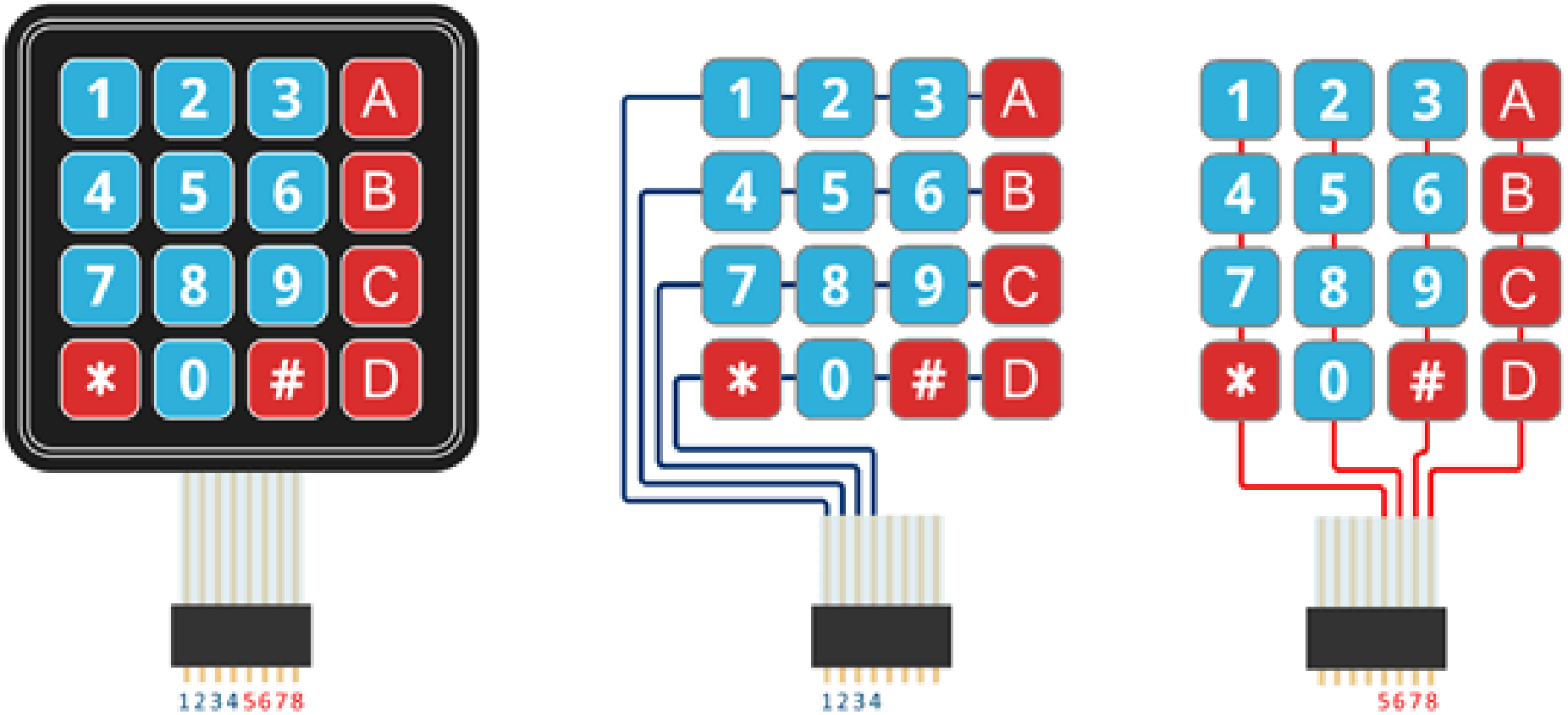
Angelo Joseph
Soto Vergel

Teclado Matricial



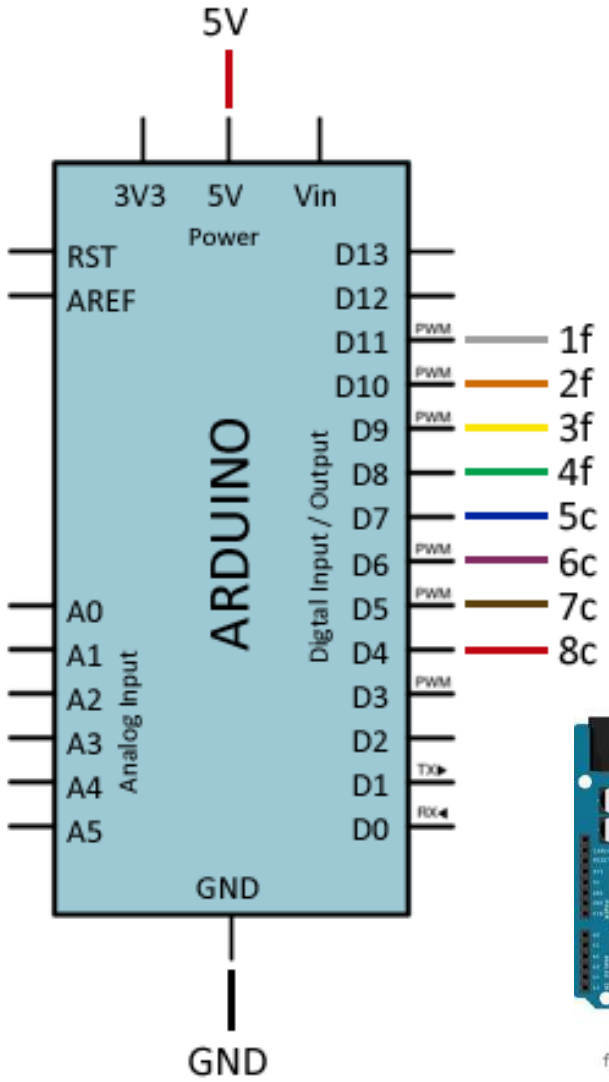
Angelo Joseph
Soto Vergel

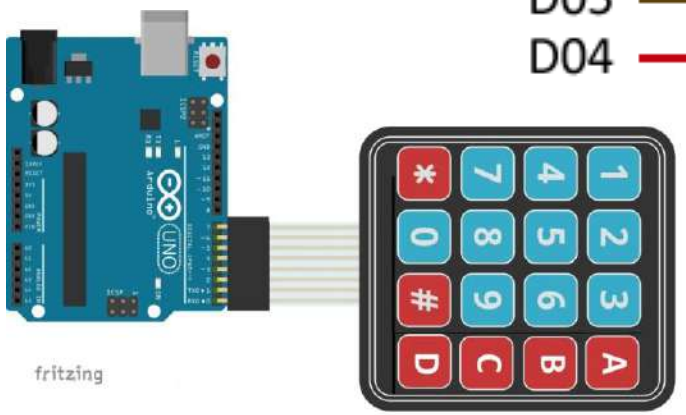
Teclado Matricial

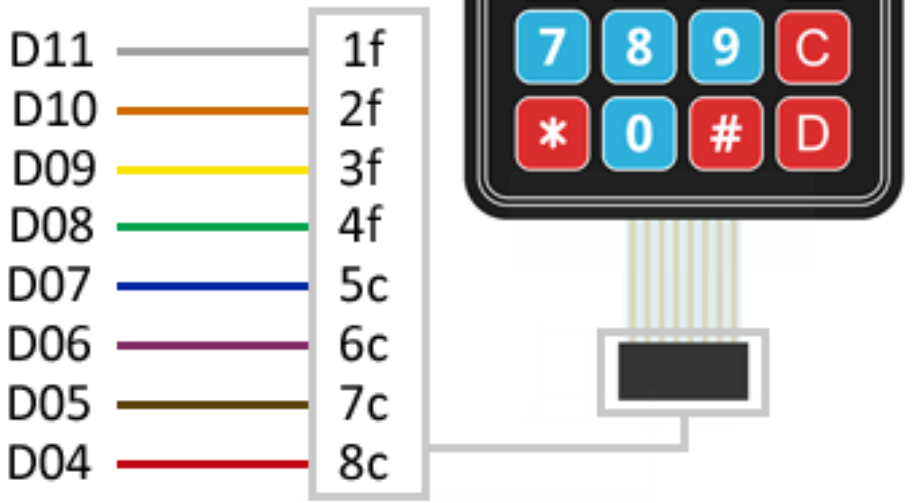


Angelo Joseph
Soto Vergel

Esquema De Conexión



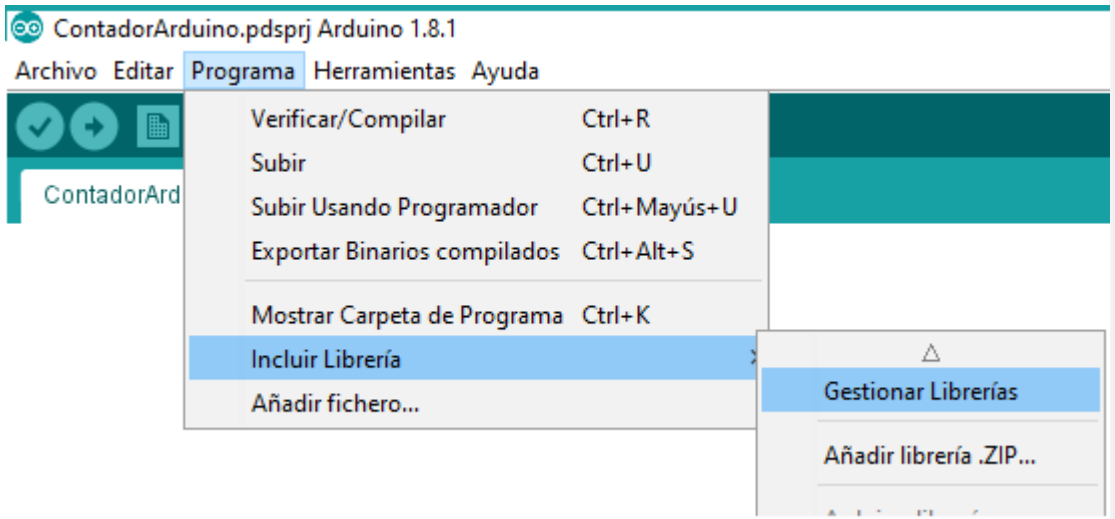


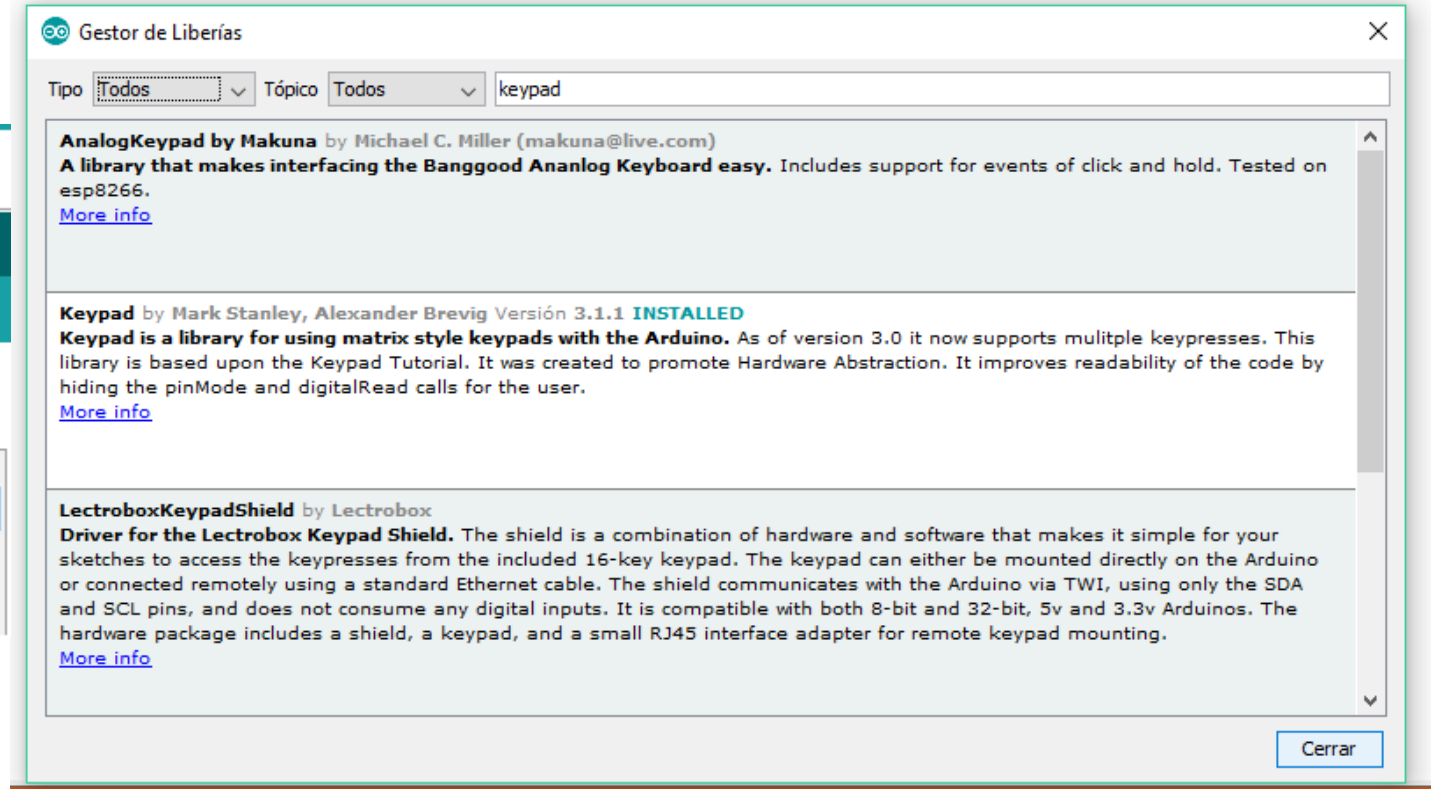


Angelo Joseph
Soto Vergel

Instalación De La Librería Para El Manejo Del Teclado Matricial En Arduino

(Librería Keypad.h)





Angelo Joseph
Soto Vergel

Manejo Del Teclado Matricial En Arduino

```
#include <Keypad.h>
const byte rows = 4; //four rows
const byte cols = 3; //three columns
char keys[rows][cols] = {
  {'1','2','3'},
  {'4','5','6'},
  {'7','8','9'},
  {'#','0','*'}
};
const byte rowPins[rows] = {5, 4, 3, 2}; //connect to the row pinouts of the keypad
const byte colPins[cols] = {8, 7, 6}; //connect to the column pinouts of the keypad
Keypad kp = Keypad(makeKeymap(keys), rowPins, colPins, rows, cols);
```

Instrucciones necesaria para el manejo del teclado matricial

Angelo Joseph
Soto Vergel

Funciones Para El Manejo Del Teclado Matricial

(Librería Keypad.h)

- begin(makeKeymap(userKeymap))

Inicializa el mapa de teclado interno para ser igual a userKeymap.

- waitForKey()

Esta función esperará siempre hasta que alguien presione una tecla. Advertencia: Bloquea todos los demás códigos hasta que se presiona una tecla. Eso significa que no hay LED parpadeante, no hay actualizaciones de pantalla LCD, no hay nada con la excepción de las rutinas de interrupción.

- getKey()

Devuelve la clave que se presiona, si la hay. Esta función no bloquea.

- isPressed(char keyChar)

Devuelve true si la clave keyChar es presionada.

Angelo Joseph
Soto Vergel

Funciones Para El Manejo Del Teclado Matricial

- getState()

Devuelve el estado actual de cualquiera de las teclas. Los cuatro estados son IDLE, PRESSED, RELEASED y HOLD.

- keyStateChange()

Permite saber cuando la clave ha cambiado de un estado a otro. Por ejemplo, en lugar de simplemente probar una clave válida, puede probar cuando se pulsa una tecla.

- setHoldTime(unsigned int time)

Establecer la cantidad de milisegundos que el usuario tendrá que mantener pulsado un botón hasta que se active el estado HOLD.

Angelo Joseph
Soto Vergel

Funciones Para El Manejo Del Teclado Matricial

- setDebounceTime(unsigned int time)

Establece la cantidad de milisegundos que el teclado esperará hasta que acepte un nuevo keypress/keyEvent. Este es el método de "rebote de tiempo".

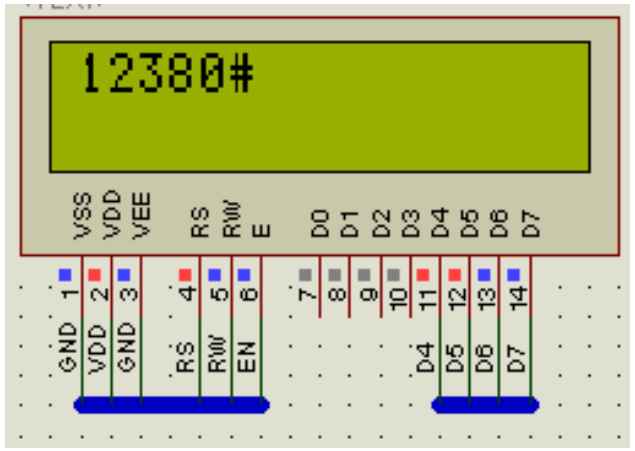
- addEventListener(keypadEvent)

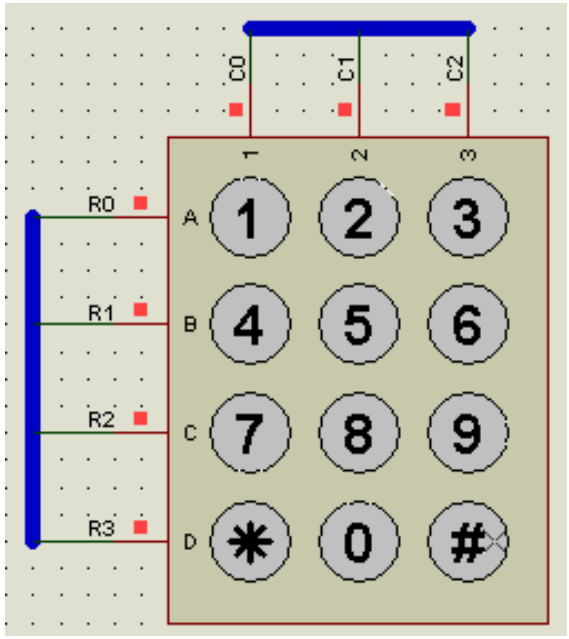
Activar un evento si se utiliza el teclado.

Angelo Joseph
Soto Vergel

Ejemplo Teclado

Introducir datos por el teclado y visualizarlos en el LCD. Cuando se pulsa la tecla '*' borrar el LCD.





Angelo Joseph
Soto Vergel

Código

Angelo Joseph
Soto Vergel

Simulación En Proteus

Angelo Joseph
Soto Vergel

Memoria EEPROM

En Arduino existen tres tipos de memoria:

- **FLASH**, no volátil, donde se graba el sketch (incluido el bootloader).
- **SRAM** (*Static Random Access Memory*), volátil, donde se almacenan las variables durante el funcionamiento.
- **EEPROM** (*Electrically-Erasable Programmable Read-Only Memory*), no volátil, que puede usarse para almacenar información entre reinicios.

Angelo Joseph
Soto Vergel

Capacidad De La Memoria EEPROM De Arduino

Placa Arduino	Memoria EEPROM (KB)
UNO	1
DUE	-
Leonardo	1
Mega	4
Micro	1
Mini	1
Nano	1

Placa Arduino	Memoria EEPROM (KB)
Ethernet	1
Esplora	1
Bluetooth	1
Fio	1
Pro Mini	1
Lilypad	1

Angelo Joseph
Soto Vergel

Formas De Usar La EEMPROM De Arduino

En la memoria de Arduino se debe trabajar por direcciones y byte a byte tanto para leer como para escribir. Esto significa que para una memoria de 1kB se tendrá desde la dirección 0 hasta la 1023 y se podrá utilizar valores de 0 a 255. En caso de que se quiera guardar valores mayores se tendrá que dividirlos por bytes. Entonces es posible:

Angelo Joseph
Soto Vergel

Formas De Usar La EEMPROM De Arduino

- **Utilizando un solo byte**. Leer y escribir un solo byte.
- **Utilizando dos bytes.** El software de Arduino tiene instrucciones pensadas para obtener el byte más significativo y el menos significativo, para separa el dato en dos bytes y almacenarlos de forma independiente.
- **Utilizando más bytes.** Después de separar una variable en bytes para almacenarla en la memoria EEPROM de Arduino surge el problema de volverla a componer para recuperar la información. Así, por ejemplo, si se tiene una variable de tipo *long* separada en cuatro bytes dentro de la EEPROM, al leer esos datos lo que se tiene es cuatro bytes, no un *long*.

Angelo Joseph
Soto Vergel

Funciones Para El Manejo De La EEMPROM

(Librería EEPROM.h)

- read(adress) //Lee un byte de la dirección "*adress*".
- write(adress,value) //Escribe un byte "*value*" en la dirección "*adress*". Una escritura EEPROM tarda 3,3 ms en completarse.
- put(adress,data) //Escribe cualquier tipo de dato en la EEPROM.
- get(adress,data) //Lee cualquier tipo de dato de la EEPROM.
- update(adress,value) //Escribe un byte en la EEPROM. El valor sólo se escribe si difiere del ya guardado en la misma dirección.

Angelo Joseph
Soto Vergel

Ejemplo EEPROM

Diseñar un sistema para el control de acceso; a través de un teclado de 3x4 introducir una clave de 3 dígitos que cuando sea correcta abra una puerta (con un pulso a un relé) y lo indique a una pantalla LCD. Guardar la clave de acceso en la memoria EEPROM.

Angelo Joseph
Soto Vergel

Código

Angelo Joseph
Soto Vergel

Simulación En Proteus

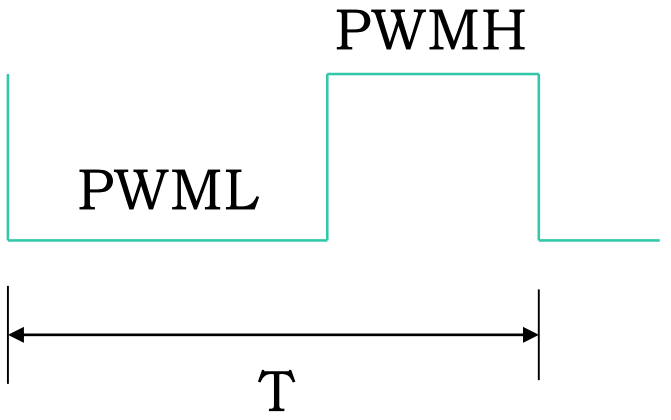
Angelo Joseph
Soto Vergel

PWM (Pulse Width Modulation)

Es una técnica en la que se modifica el ciclo de trabajo de una señal periódica (una sinusoidal o una cuadrada, por ejemplo), ya sea para transmitir información a través de un canal de comunicaciones o para controlar la cantidad de energía que se envía a una carga.

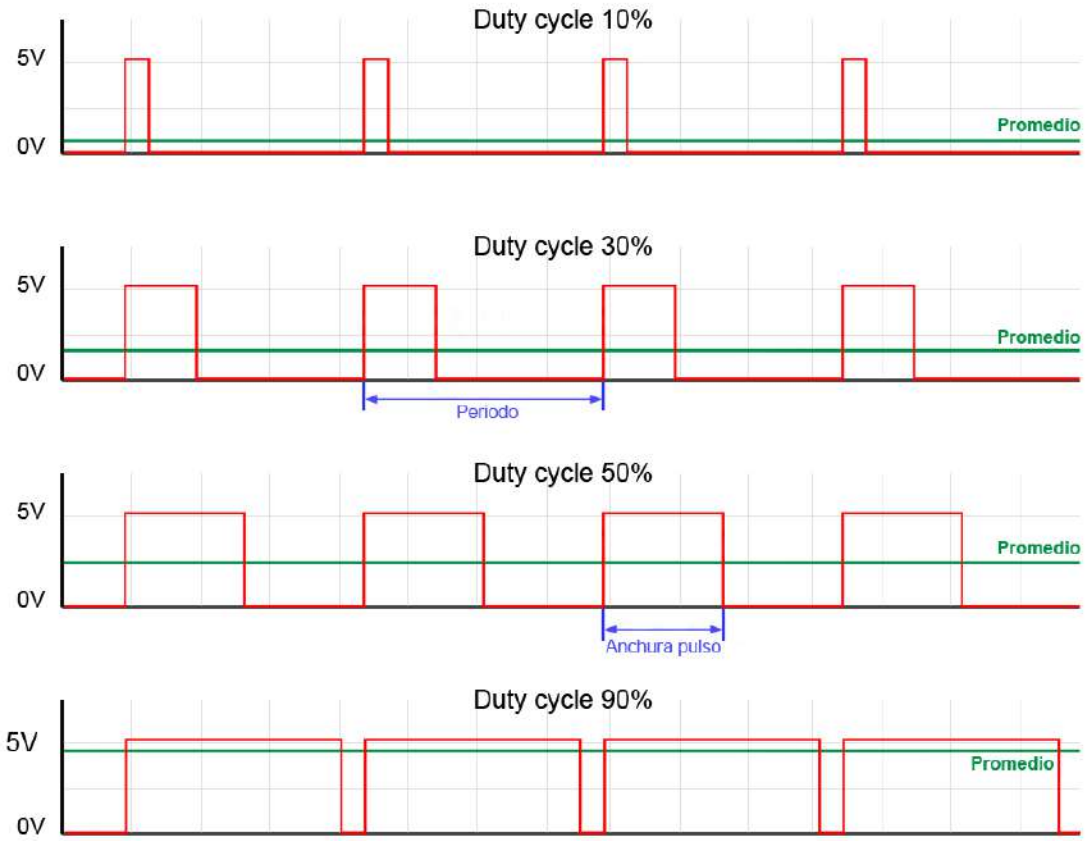
$$D = \frac{PWMH}{T}$$

D es el ciclo de trabajo.
$PWMH$ es el tiempo en que la función es positiva.
T es el periodo de la función.



Angelo Joseph
Soto Vergel

PWM En Arduino



Una llamada a analogWrite() se encuentra en una escala de 0 a 255, de modo que para un valor de 255 solicita un ciclo de trabajo del 100% (siempre activado) y para un valor de 127 es un ciclo de trabajo del 50%. También es posible generar un PWM usando los pines digitales.

$$V_{medio} = (V_{cc+} - V_{cc-}) \cdot \frac{DutyCycle}{100}$$

$$DutyCycle = 100 \cdot \frac{V_{medio}}{(V_{cc+} - V_{cc-})}$$

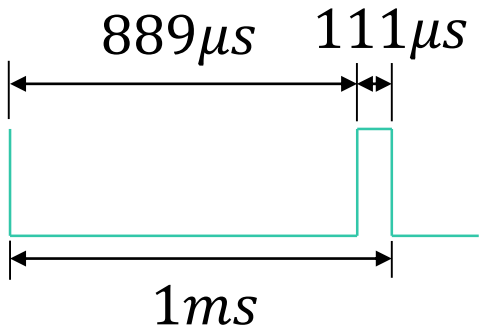
Angelo Joseph
Soto Vergel

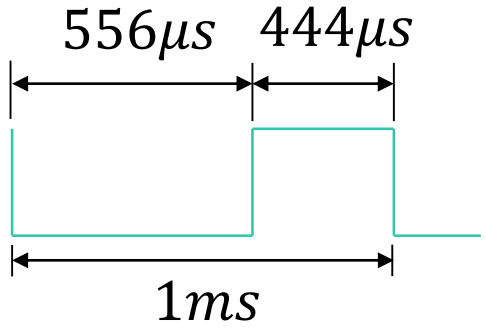
Ejemplo PWM

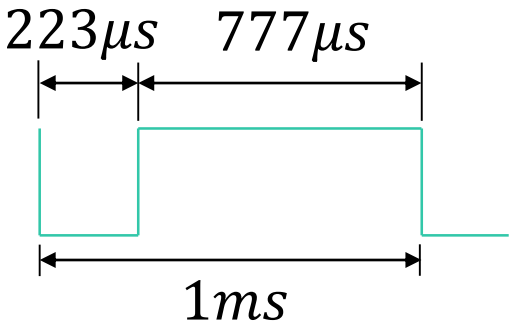
A través de un teclado, introducir los datos de velocidad de un motor y generar una señal modulada en ancho de pulso proporcional al dato de la velocidad.

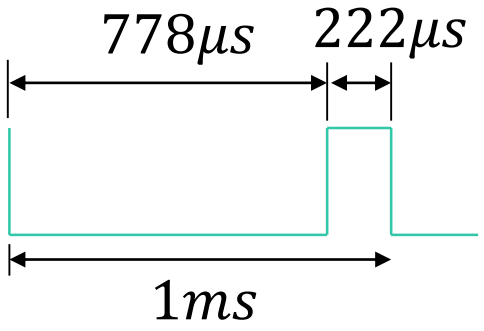
Angelo Joseph
Soto Vergel

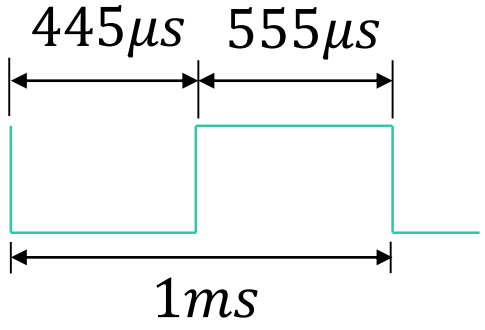
9 Velocidades

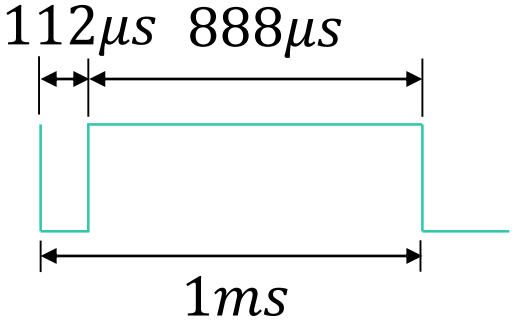


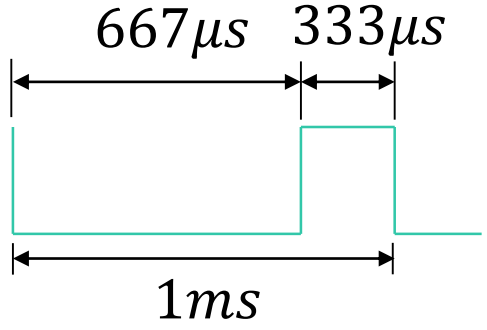


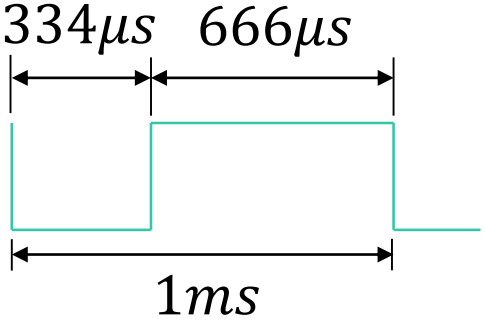


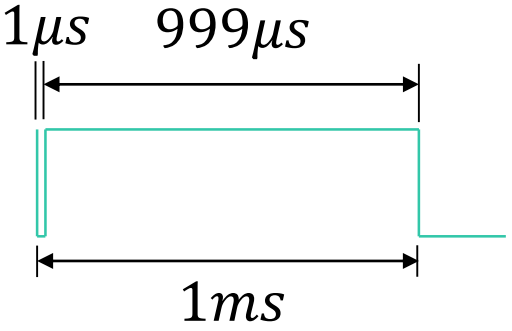












Angelo Joseph
Soto Vergel

Código

Angelo Joseph
Soto Vergel

Simulación En Proteus

Angelo Joseph
Soto Vergel

Ejercicio

Realice una aplicación con un teclado matricial y un LCD de doble línea que permita escribir texto en el LCD de la misma forma como se escribía texto en los celulares de segunda generación (también debe tener la opción de escribir números). Cuando se escriba "clc" se debe borra el display. Cuando llegue al final de la primera línea, debe seguir escribiendo en la segunda línea. Cuando llegue al final de la segunda línea debe sobreescribir lo que esté en la primera línea.