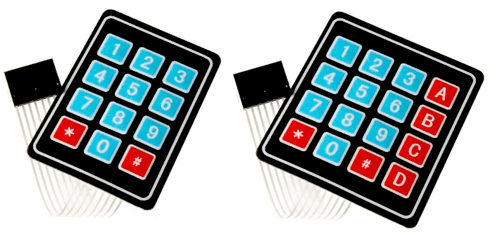
**TECLADO MATRICIAL**



Un teclado matricial es un dispositivo que agrupa varios pulsadores y permite controlarlos empleando un número de conductores inferior al que necesitaríamos al usarlos de forma individual. Podemos emplear estos teclados como un controlador para un autómata o un procesador como Arduino.

Estos dispositivos **agrupan los pulsadores en filas y columnas** formando una matriz, disposición que da lugar a su nombre. Es frecuente una disposición rectangular pura de NxM columnas, aunque otras disposiciones son igualmente posibles.

Los teclados matriciales son frecuentes en electrónica e informática. De hecho, los teclados de ordenador normales son teclados matriciales, siendo un buen ejemplo de teclado matricial con disposición no rectangular.

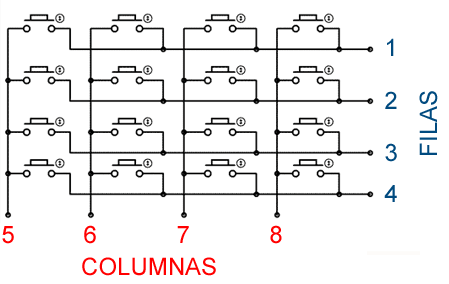
Una de las desventajas de usar un teclado matricial es que pueden causar problemas cuando se pulsa más de una tecla simultáneamente. Este es uno de los motivos por el que los teclados de ordenador usan una disposición no rectangular, agrupando ciertas teclas en circuitos diferentes (Ctrl, Alt, Shift…)

En el campo de la electrónica casera y Arduino, se venden múltiples modelos de teclados matriciales en **distintos soportes** (rígidos o flexibles) y con **distinto número de teclas**, siendo habituales configuraciones de 3×3, 3×4, y 4×4.

Podemos emplear teclados matriciales en nuestros proyectos de electrónica y robótica, por ejemplo, para cambiar el modo de funcionamiento de un montaje, para solicitar un password, como teclas de dirección para controlar un brazo robótico o un vehículo, o proporcionar instrucciones a un robot.

La siguiente imagen muestra, a modo de ejemplo, una disposición rectangular de 4×4, aunque el funcionamiento es análogo en otras disposiciones. Al detectar la pulsación en la columna X y la fila Y, sabremos que se ha pulsado la tecla (X,Y).

Internamente la disposición de los pulsadores es la siguiente, que es muy similar a la disposición que vimos al tratar sobre [displays LED matriciales](https://www.luisllamas.es/matriz-led-arduino-max7219/).



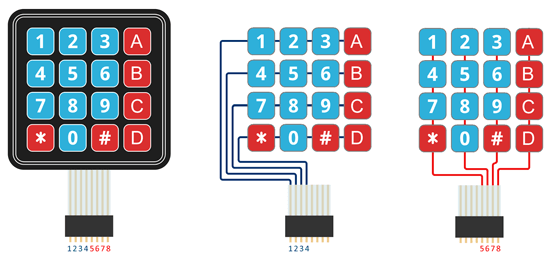
Para detectar la pulsación de una tecla actuaremos de forma similar a la [lectura simple de un pulsador](https://www.luisllamas.es/leer-un-pulsador-con-arduino/). En resumen, ponemos a tierra un extremo del pulsador, y el otro lo conectamos a una entrada digital con una resistencia de pull-up.

Para leer todas las teclas tendremos **que hacer un barrido por filas**. En primer lugar ponemos todas las filas a 5V, y definimos todas las columnas como entradas con resistencia de pull-up.

Progresivamente ponemos una fila a 0V, y leemos las entradas de la columna. Una vez realizada la lectura volvemos a ponerla a 5V, pasamos a la siguiente fila, y volvemos a realizar el progreso hasta recorrer todas las filas.

El papel de filas y columnas en la explicación es **intercambiable**, pudiendo realizar un barrido por las columnas, y lectura en las filas.

Para detectar NxM pulsadores necesitamos sólo N+M conductores. Por tanto ahorro de conductores es superior cuanto más grandes sean N y M, y más parecidos entre sí. (ej: 16 pulsadores en 2×8 necesitan 10 conductores, y en 4×4 sólo 8 conductores.)

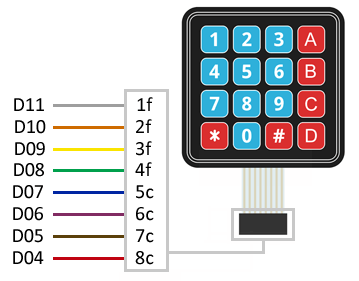


En consecuencia, los teclados de 1xM, Nx1 y 2×2 no suponen ningún ahorro en conductores aunque, independientemente, puede que os resulten de utilidad únicamente por agrupar en un mismo dispositivo la disposición que deseáis.

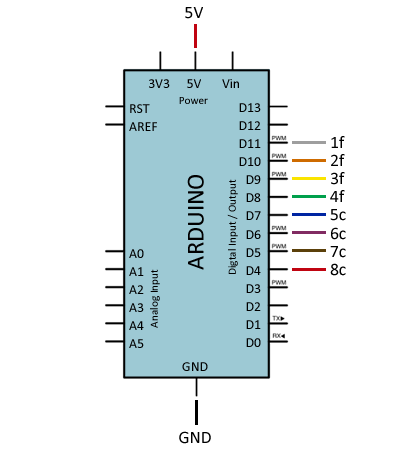
Como hemos comentado, la mayor desventaja de la disposición matricial es que pueden dar problemas al detectar la pulsación de múltiples teclas de forma simultánea.

**ESQUEMA DE CONEXIÓN**

El esquema de conexión es sencillo. Simplemente conectamos todos los pines a entradas digitales de Arduino. Por ejemplo, en el ejemplo de un teclado de 4×4 el esquema quedaría de la siguiente forma.



Que visto desde Arduino quedaría de la siguiente forma.



**EJEMPLOS DE CÓDIGO**

Existen varias librerías diseñaras para facilitar la lectura de teclados matriciales en Arduino. Por ejemplo, la librería Keypad, disponible en [este enlace](https://github.com/Chris--A/Keypad). La librería proporciona ejemplos de código, que resulta aconsejable revisar.

El siguiente ejemplo es una modificación a partir de los disponibles en la librería.

#include <Keypad.h>

const byte rowsCount = 4;

const byte columsCount = 4;

char keys[rowsCount][columsCount] = {

{ '1','2','3', 'A' },

{ '4','5','6', 'B' },

{ '7','8','9', 'C' },

{ '#','0','\*', 'D' }

};

const byte rowPins[rowsCount] = { 11, 10, 9, 8 };

const byte columnPins[columsCount] = { 7, 6, 5, 4 };

Keypad keypad = Keypad(makeKeymap(keys), rowPins, columnPins, rowsCount, columsCount);

void setup() {

Serial.begin(9600);

}

void loop() {

char key = keypad.getKey();

if (key) {

Serial.println(key);

}

}