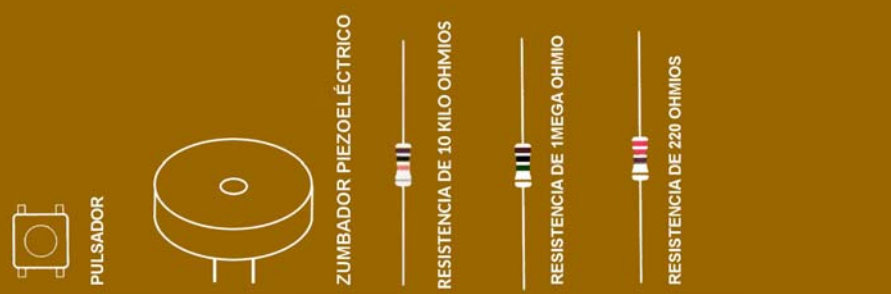
07



INGREDIENTES

TECLADO MUSICAL

CON POCAS RESISTENCIAS Y PULSADORES VAMOS A CONTRUIR UN PEQUEÑO TECLADO MÚSICAL

Descubra: circuito mixto con resistencias, matrices

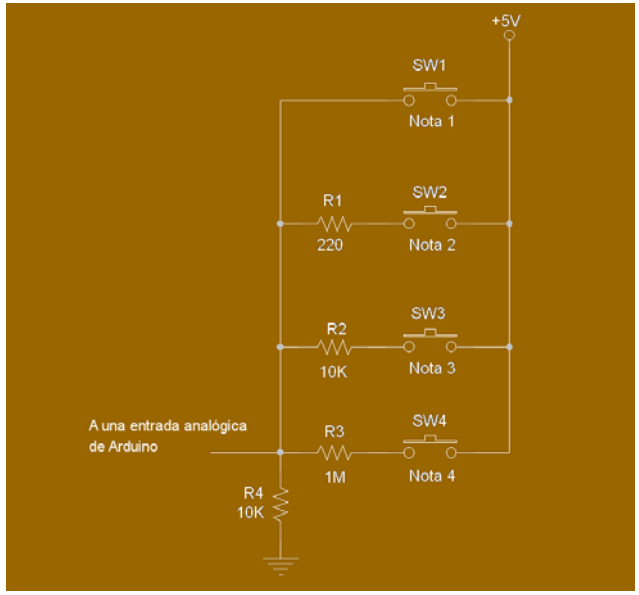
Tiempo: **45 MINUTOS**

Nivel: **medio-alto**

Proyectos en los que se basa: **1,2,3,4,6**

Al utilizar en este proyecto varias resistencias y pulsadores conectados a una entrada analógica de Arduino para generar diferentes tonos, se está construyendo algo que se conoce con el nombre de circuito mixto con resistencias.

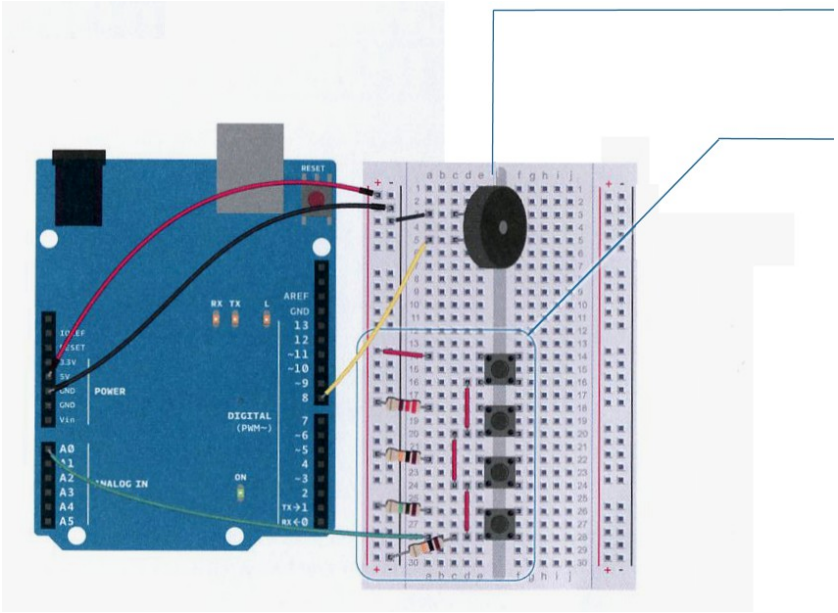
Esta es una forma de leer un número de pulsadores usando una entrada analógica. Es una técnica útil sino entiende como funcionan las entradas digitales. Se conectan un número determinado de pulsadores en paralelo y conectados a la entrada analógica A0 de Arduino. Cada uno de estos pulsadores se conectan al positivo de la alimentación a través de una resistencia. Cuando se presiona un pulsador, aparece una tensión en el terminal de entrada analógico A0 de Arduino, y según que pulsador se presione esta tensión será diferente. Si se presionan dos pulsadores al mismo tiempo se consigue en la entrada analógica un tensión que será proporcional al valor de la resistencias en paralelo de los dos pulsadores presionados. Al final se trata de varios divisores de tensión conectados en paralelo los cuales se activan cada vez que se presione un pulsador.



Circuito mixto con resistencias como entrada analógica.

Figura 1

MONTANDO EL CIRCUITO



La disposición de resistencias y los pulsadores conectados a esta entrada analógica se conoce con el nombre de circuito mixto

Figura 2

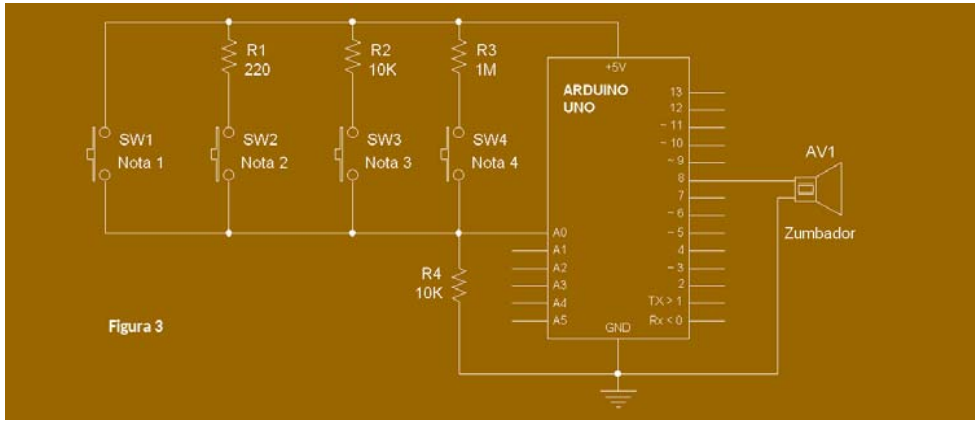
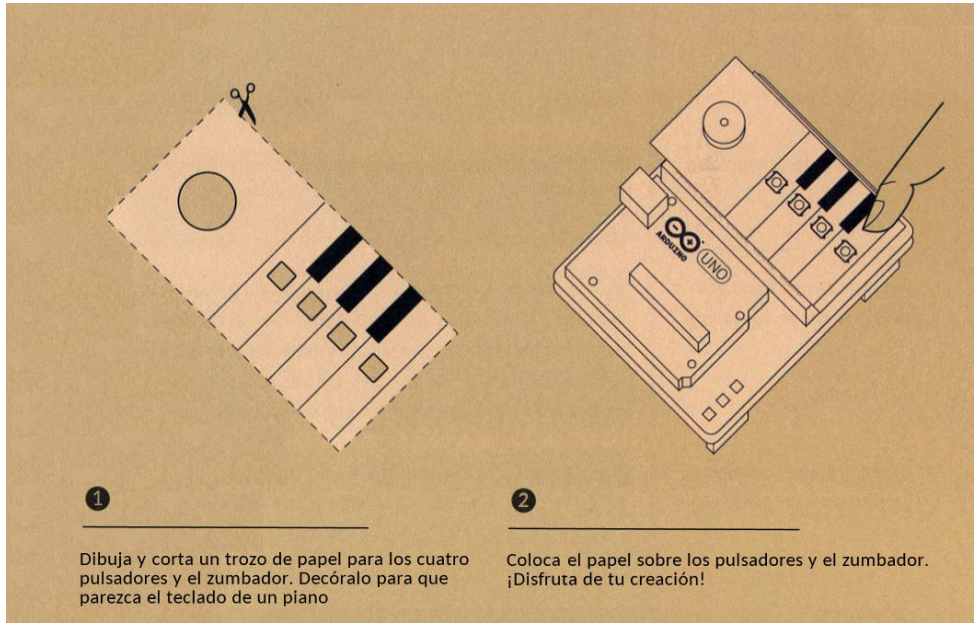


Figura 3

1. Conectar el cable de alimentación y la masa a la placa de pruebas como se hizo en proyectos anteriores. Conectar un pin del zumbador a masa. Conectar el otro pin al terminal 8 (salida digital) de Arduino.

2. Conectar los pulsadores en la placa de prueba tal y como se muestra en esta ilustración. La disposición de las resistencias y los pulsadores conectados a esta entrada analógica se conoce con el nombre de circuito mixto. Conectar el primer pulsador directamente a la alimentación positiva. Conectar el segundo, el tercero y el cuarto pulsador a la alimentación positiva a través de una resistencia de 220 ohmios, 10 kilo ohmios y 1 mega ohmio respectivamente. Conectar todas las salidas de los pulsadores entre sí, en un solo punto de unión. Conectar este punto de unión a masa a través de una resistencia de 10Kilo ohmios, y también conectarla a la entrada analógica 0 de Arduino. Cada pulsador junto con la resistencias se comportan como un divisor de tensión en el momento de que un pulsador se presiona.

Piensa en como mejorar este proyecto añadiendo un dibujo que imite un teclado. Mientras que los sintetizadores analógicos antiguos no tenían un diseño atractivo, este teclado es elegante y además digital. Preparar un pequeño trozo de cartón que se pueda cortar para acomodar los pulsadores y el zumbador. Colocar una etiqueta a cada tecla para saber que nota suena en el momento de pulsarla.



1. Dibuja y corta un trozo de papel para los cuatro pulsadores y el zumbador. Decóralo para que parezca el teclado de un piano

2. Coloca el papel sobre los pulsadores y el zumbador. ¡Disfruta de tu creación!

EL CÓDIGO

La matriz

En este programa, será necesario memorizar las frecuencias de las notas musicales que se van a oír cuando se presione un pulsador. Se van a utilizar las frecuencias medias que se corresponden con la cuarta octava de las notas musicales de un piano, C4, D4, E4 y F4, cuyas frecuencias son 262, 294, 330 y 349 ciclos por segundo o hercios. Para poder hacer todo esto se utiliza una nueva clase de variable llamada matriz.
La matriz es una forma de almacenar diferentes valores que están relacionados unos con otros, como las frecuencias de una escala musical, usando un solo nombre. El uso de una matriz permite acceder a la información que ella almacena de una forma rápida y eficiente. Para declarar una matriz se hace igual que con una variable, pero escribiendo a continuación del nombre corchetes cuadrados: []. Después del signo igual la información se puede colocar de varias formas, entre llaves para definir el valor de las variables o sin ellas si solo se define un valor. En la primera línea de la ventana de la derecha se declara una matriz de seis valores llamada botones (int botones[6];).
En la segunda línea ademas de definir una matriz con una sola variable también se indica el valor que va a contener (int botones[0] = 2;).
Para leer o cambiar los valores de las variables de una matriz, se accede individualmente a cada una de estas variables usando el nombre de la matriz y a continuación el número de posición que ocupa dentro de esta matriz. La posición se refiere al orden que ocupa esta variable cuando la matriz es creada. La primera variable ocupa la posición 0, la segunda la posición 1, la tercera la posición 2 y así sucesivamente. Por ejemplo, en la matriz *int notas[] = {262,294,330,349}*, el valor de la variable notas[2] vale 330.

Crear una matriz para guardar las frecuencias

Configurar una matriz para guardar las cuatro notas musicales usando las frecuencias que se muestran en el primer párrafo de esta hoja. Se crea como una matriz global al declararla antes de la función **setup()**.

Establecer la comunicación serie con el ordenador

Dentro de la función **setup(),** se escribe la instrucción que permite realizar la comunicación con el ordenador.

Leer el valor de la entrada analógica para enviarla al monitor serie

En el **loop()**, declarar una variable para guardar el valor que se obtiene al leer el pin de la entrada analógica A0 de Arduino (int ValorTeclaPulsada). Como cada pulsador tiene en serie una resistencia diferente conectada al positivo de la alimentación, en cada uno de ellos aparecerá una tensión diferente cuando estén presionados, por ejemplo, si se presiona el pulsador que tiene la resistencia de 10K en serie (pulsador de la Nota 3) en la entrada A0 aparecerán 2.4V (para +VCC = 4.8V), por tanto el valor que lee Arduino es 512. Para ver estos valores en la pantalla del ordenador se añade la línea "**Serial.println(ValorTeclaPulsada);**".

Utilizar la instrucción if()..else para determinar que nota deberá de sonar

Usando la instrucción **if()...else**, se puede asignar cada uno de los valores que se obtienen al presionar los pulsadores a un tono diferente, por ejemplo, el pulsador Nota 3 con 330Hz, para un valor de 512. Como las resistencias tienen una tolerancia, al presionar el pulsador de la Nota 3 Arduino puede leer un valor diferente a 512, por tanto se establecen unos márgenes (de 505 a 515) para saber que este pulsador ha sido presionado. Después de montar el circuito si alguno de los pulsadores no funciona usar la información del monitor serie para saber cual es el valor que lee Arduino y así ampliar el margen para este pulsador.


```
int botones[6];
// Crear una matriz para 6 variables de tipo entero

int botones[0] = 2;
// Dar a la primera variable de la matriz el valor de 2
```

```
1  int notas[ ] = {262,294,330,349};
```

```
2  void setup() {
3    Serial.begin(9600);
4  }
```

```
5  void loop() {
6    int ValorTeclaPulsada = analogRead(A0);
7    Serial.println(ValorTeclaPulsada);
```

```
8    if(ValorTeclaPulsada == 1023){
9      tone(8, notas[0]);
10   }
```

Reproducir las notas que se correspondan con el valor analógico de la entrada

Después de la función **if()**, se llama a la función **tone()**. El programa se dirige a la matriz para determinar que frecuencia se debe de reproducir. Si la lectura realizada en la entrada analógica A0 coincide con uno de los valores de las instrucciones if, se le indica a Arduino que reproduzca un tono. Es posible que el circuito produzca ruido, ya que los valores pueden fluctuar un poco mientras se presiona un pulsador. Para corregir esta variación, es una buena idea disponer de un rango de valores para verificar que el valor de la lectura coincide con alguno de los valores dentro de este rango. Se utiliza el operador lógico AND (función de multiplicación), representado por "&&", dentro de múltiples instrucciones **if()...else** para comprobar la condición anterior.

Si presiona el primer pulsador (SW1 - Nota 1), la nota [0] (262 Hz) se escuchará a través del zumbador. Si presiona el segundo pulsador (SW2 – Nota 2) se oirá la nota[1] (294 Hz), y si se presiona el tercer pulsador (SW3 – Nota 3) se oirá la tercera nota[2] (330 Hz). Ahora es cuando se puede ver lo útil y manejable que es usar una matriz.

Parar de toca un tono cuando no se presiona nada

Solo una frecuencia se puede reproducir a la vez, así que si se presionan dos pulsadores al mismo tiempo solo se oirá un sonido, aquel que tenga la resistencia más baja de los dos pulsadores presionados.

Para silenciar el zumbador cuando no se presiona un pulsador se utiliza la función **noTone()**, el cual necesita el número del terminal en donde dejar de reproducir el sonido.

COMO SE UTILIZA

Si los valores de las resistencias del circuito montado son los mismos que los mostrados en este proyecto, se deberá oír un sonido a través del zumbador cuando se presione un pulsador. Si no es así, mirar a través del monitor serie el valor que aparece al presionar un pulsador para verificar que se encuentran dentro de los rangos establecidos en las instrucciones **if()...else.** Si se oye un sonido intermitente, incrementar un poco el rango de valores de la instrucción cuyo valor se encuentra cerca del valor mostrado al presionar el pulsador que produce este sonido intermitente.

Presionar varios pulsadores al mismo tiempo, para ver que tipo de valor aparece en el monitor serie. Usar estos nuevos valores para producir mas sonidos. Experimentar con diferentes frecuencias para aumentar el número de tonos que puede reproducir este teclado. Es posible ver cuales son las frecuencias de las notas musicales en esta página: *arduino.cc/frequencies*

Si se reemplaza el circuito mixto de resistencias por sensores analógicos, ¿será posible usar la información que se obtiene de estos sensores para crear un instrumento musical más dinámico? Se podrían usar estos valores para cambiar la duración de una nota o, como el proyecto del Theremin, crear una gama de sonidos que varían con la iluminación.


```
11  else if(ValorTeclaPulsada >= 990 && ValorTeclaPulsada <= 1010){
12    tone(8, notas[1]);
13    }
14   else if(ValorTeclaPulsada >= 505 && ValorTeclaPulsada <= 515){
15    tone(8, notas[2]);
16    }
17   else if(ValorTeclaPulsada >= 5 && ValorTeclaPulsada <= 10){
18    tone(8, notas[3]);
19    }
```

```
20   else{
21    noTone(8);
22   }
23 }
```

La función **tone()** es útil para generar tonos, pero tiene algunas limitaciones. Solo puede crear formas de onda cuadrada, no ondas con formas senoidales o triangulares. Las ondas cuadradas en absoluto parecen ondas. Como se pudo ver en la figura 1 del proyecto número 6 (página 71), la onda cuadrada esta formada por una serie de pulsos en estado bajo (0V) y en estado alto (+5V).

Ahora es posible formar una banda para usar este teclado, pero habrá que tener en mente algunas cosas: solo un tono se puede reproducir a la vez y la función **tone()** interferirá con la instrucción **analogwrite()** sobre los pins 3 y 11, en caso de que se quiera usar junto con estos pins para aumentar el número de sonidos que este teclado pueda generar.

Las matrices son útiles para guardar el mismo tipo de información junta; se accede a la información de una matriz por los números índice los cuales hacen referencia a las variables individuales que contiene. Los montajes mixtos con resistencias son una forma fácil de obtener un mayor número de entradas analógicas al conectarlos a una entrada analógica dentro de un sistema.