El kit de aprendizaje STEM Hub, es la herramienta definitiva para que cualquiera pueda aprender a programar.

Ya sea que estés en la escuela primaria, en la universidad, seas padre, maestro o abuelo, puedes aprender a escribir código en el lenguaje Arduino. No se requiere experiencia alguna.

STEM Hub integra 8 módulos de entrada comunes y 8 módulos de salida

- Módulo de entrada:
    - Botón
    - Potenciómetro
    - Potenciómetro deslizante
    - Sensor ultrasónico
    - Sensor PIR (infrarrojo piroeléctrico del cuerpo humano)
    - Sensor de luz
    - Sensor de sonido
    - Sensor de temperatura
- Módulo de salida:
    - Diodo emisor de luz
    - Servo
    - Motor paso a paso
    - Motor DC con hélice
    - Pantalla LCD
    - Buzzer
    - Altavoz
    - LED RGB

Cada módulo viene precableado con las líneas de alimentación necesarias y solo requiere conectar los terminales de señal, por lo que puede dedicar un mínimo de tiempo a conectar circuitos y más tiempo al aprendizaje real. Es tan fácil como aprender el abecedario.

Este es el regalo perfecto para los más pequeños de la familia para que se interesen por la programación y la electrónica. Da el primer paso hacia una carrera en informática.

El tutorial se adhiere al principio de progreso gradual, y la mayoría de los módulos se pueden aprender conectándolos directamente entre sí sin necesidad de programación. Luego, mediante el control del programa, se pueden realizar escenas que pueden verse en todas partes en la vida.

Tutorial

Diodo emisor de luz. LED

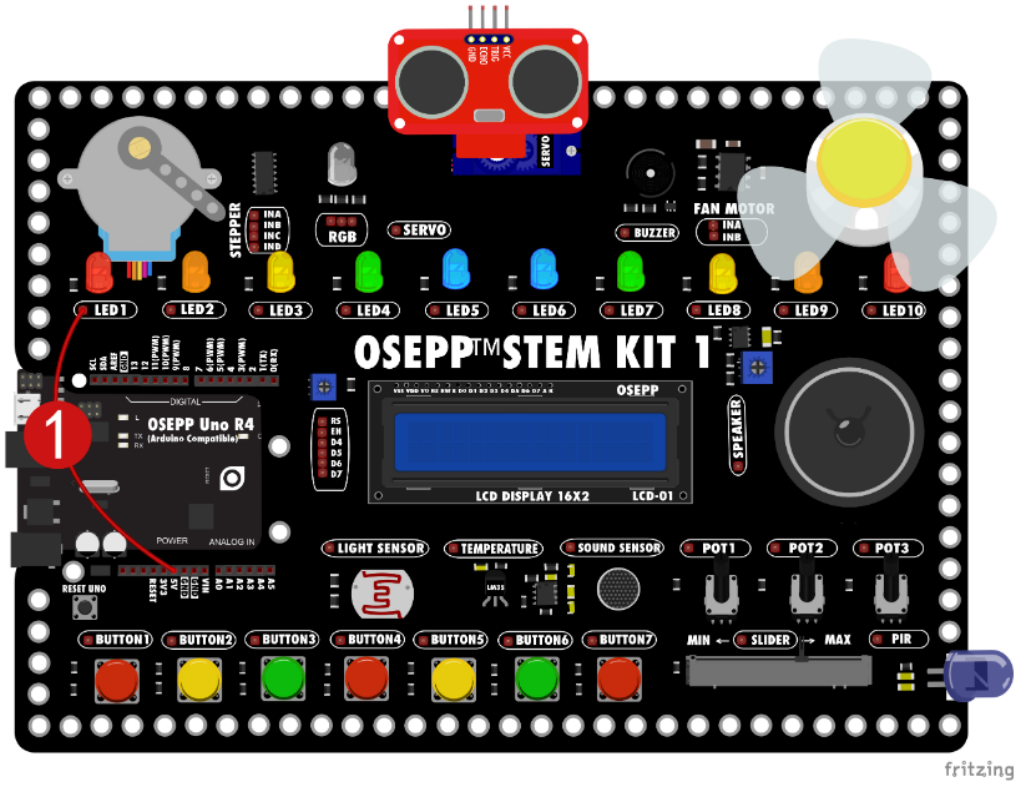
1. Encender un LED

El diodo emisor de luz, abreviado como LED, es un componente electrónico semiconductor que puede emitir luz y es un componente que convierte la energía eléctrica en energía luminosa. Los diodos emisores de luz se han utilizado ampliamente en iluminación LED interior y exterior, pantallas LED, semáforos, luces de automóviles, retroiluminación de pantallas, iluminación, comunicaciones de fibra óptica y más. Los diodos emisores de luz tienen ventajas que las fuentes de luz tradicionales no tienen, como alta eficiencia, larga vida útil, no se rompen fácilmente, tienen una velocidad de respuesta rápida y una alta confiabilidad.

¿Cómo encender un LED?

Si queremos que el LED de la placa de aprendizaje STEM KIT se encienda, solo necesitamos conectar el terminal del LED al puerto positivo de la fuente de alimentación de 5V con un cable y luego conectar la placa de aprendizaje STEM KIT a la fuente de alimentación. El LED se iluminará.

Como se muestra en la figura a continuación, utilice un cable Dupont para conectar `LED1` el terminal al pin `5V` de la placa de control UNO. Después de enchufar el conector USB y encender el STEM KIT, se encenderá el LED.



Resultados de la operación

En el experimento, cuando el LED se conecta a un voltaje de 5V, el LED se encenderá.

Análisis

Un LED rojo típico solo requiere un voltaje directo de 1,5V. Si el voltaje en el ánodo no es 1,5V mayor que el del cátodo, no circulará corriente a través del LED y este no emitirá luz. Cuando el voltaje en el ánodo es 1,5V mayor que el del cátodo, el LED se enciende, pero básicamente se convierte en un cortocircuito (la corriente será muy grande). Por lo tanto, se debe utilizar una resistencia para limitar la corriente, de lo contrario, el LED podría quemarse. Generalmente, la corriente continua máxima del LED es de 25mA. Dado que el LED de la placa de aprendizaje STEM KIT está conectado a una resistencia limitadora de corriente, conectar el cable directamente a 5V no dañará el LED.

2. LED controlado por Arduino

Instalación del software de programación

Antes de comenzar el experimento, es necesario dedicar algún tiempo a instalar el software de programación y comprender sus operaciones básicas.

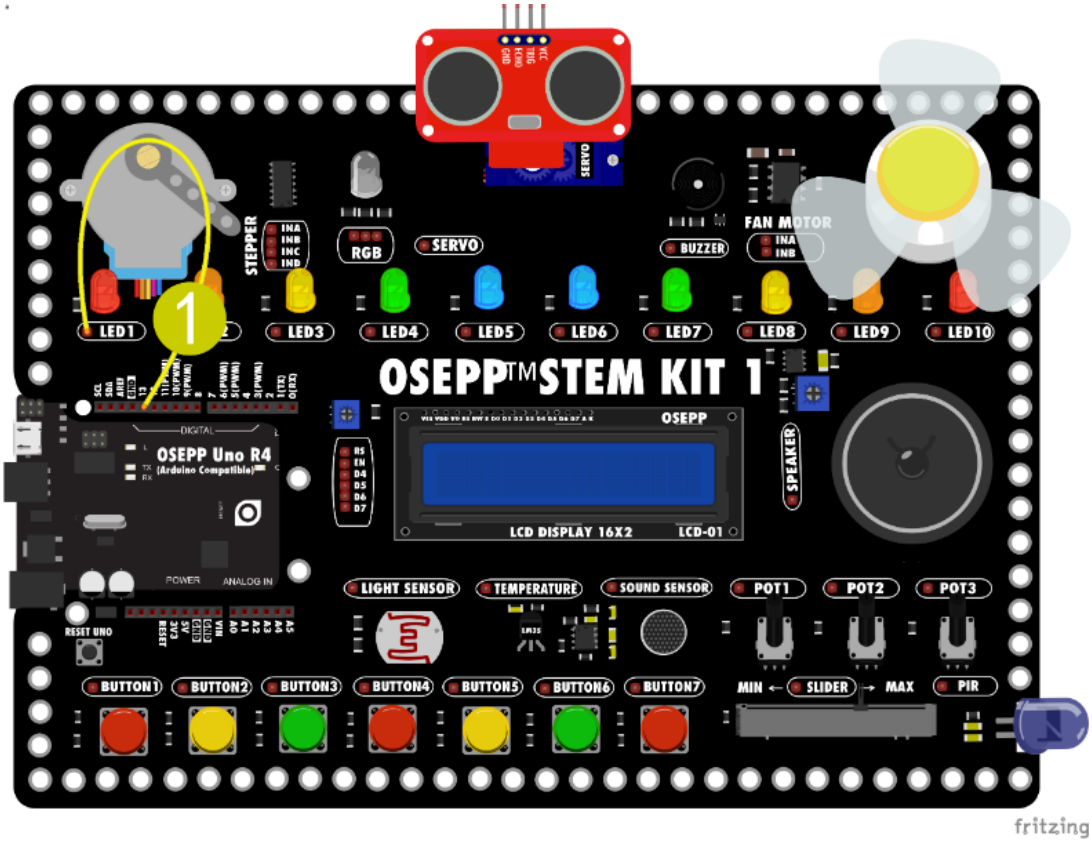
Si tiene experiencia en programación y prefiere utilizar software de programación basado en texto, consulte el siguiente enlace para instalar el IDE de Arduino:

/tutorial/robopro/instalación_ide_arduino

Los pines digitales de Arduino pueden tener salida 5V alta o baja 0V. En el último experimento, si el terminal del LED está conectado al polo negativo de la fuente de alimentación GND, no fluirá corriente a través del LED y no emitirá luz. Al controlar el pin de Arduino para emitir un nivel alto o un nivel bajo a través del programa, se puede controlar que el LED se encienda y apague, lo que equivale a conectar el terminal del LED al polo positivo o al polo negativo de la fuente de alimentación a través del programa.

Conexión

1. Conecte el terminal del LED1 al pin No. 13. de la placa UNO



2. Un extremo del USB está conectado a la computadora y el otro extremo está conectado a la interfaz de la placa base de UNO.

Construcción de programas

Antes de comenzar a programar, primero entendamos el programa Arduino. El programa Arduino más básico consta de dos partes setup y loop.

Encendido: el código de void setup(). Esta función setup se ejecuta solo una vez cuando se enciende o se reinicia el equipo. Se utiliza para inicializar variables, configurar el tipo de entrada/salida de los pines y configurar el puerto serial.

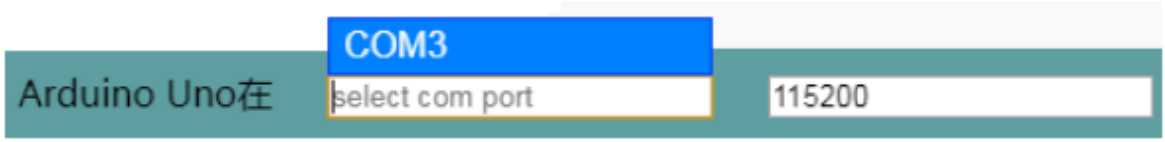
Ejecución repetida: el código de void loop(). La función loop se ejecutará repetidamente y, normalmente, la funcionalidad del programa se implementará aquí.

Programa Arduino

```
void setup()
{
    //led1
    pinMode(13, OUTPUT); //定义13号引脚为输出模式
}
void loop()
{
    digitalWrite(13, HIGH); //13号引脚输出高电平
    delay(1000); //延时1000毫秒
    digitalWrite(13, LOW); //13号引脚输出低电平
    delay(1000); ////延时1000毫秒
}
```

Subir programa

Seleccione el número de puerto en la parte inferior de la ventana del software actual.



Nota: es posible que COM3 no se muestre en su computadora; pueden aparecer otros números.

Luego haga clic en  el botón de carga en la esquina superior derecha para cargar el programa y aparecerá una ventana.

Cuando las palabras aparecen al final de la ventana`...Thank you.`, significa que el programa ha sido cargado y escrito en la placa de control UNO. Como se muestra a continuación:

```
上传中

avrdude: AVR device initialized and ready to accept instructions
avrdude: Device signature = 0x1e950f (probably m328p)
avrdude: reading input file "C:\Users\vector\AppData\Local\Temp\osepp-
R3AAIk/buildpath/sketch.ino.hex"
avrdude: writing flash (724 bytes):
avrdude: 724 bytes of flash written
avrdude: verifying flash memory against
C:\Users\vector\AppData\Local\Temp\osepp-R3AAIk/buildpath/sketch.ino.hex:
avrdude: load data flash data from input file
C:\Users\vector\AppData\Local\Temp\osepp-R3AAIk/buildpath/sketch.ino.hex:
avrdude: input file C:\Users\vector\AppData\Local\Temp\osepp-
R3AAIk/buildpath/sketch.ino.hex contains 724 bytes
avrdude: reading on-chip flash data:
avrdude: verifying ...
avrdude: 724 bytes of flash verified

avrdude done. Thank you.
```

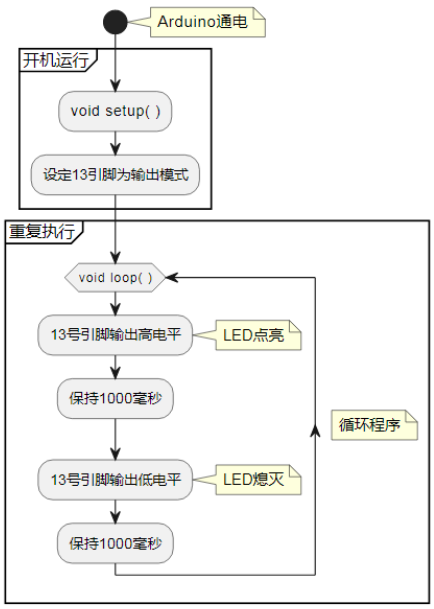
Puede cerrar esta ventana después de cargar correctamente.

Resultados de la operación

En este momento, el LED de la placa base se encenderá durante un segundo y luego se apagará durante un segundo según la configuración de nuestro programa, repitiendo el ciclo continuamente.

Análisis

El panel de control se ejecuta según el programa y el proceso de operación es el siguiente:



Cuando el pin digital de Arduino está configurado en modo de salida `OUTPUT`, solo puede emitir dos estados: nivel alto `HIGH` o nivel bajo `LOW`. Esta salida a menudo se denomina **salida digital** (a veces llamada binaria para los dos estados).
A estos estados a menudo se les denomina alto `HIGH` y bajo `LOW`. Un nivel alto `HIGH` significa "¡hay voltaje aquí!", y un nivel bajo `LOW` significa "¡no hay voltaje en este pin!".

Cuando se utiliza `digitalWrite()`el comando establece el pin `OUTPUT` en un nivel alto `HIGH`, es equivalente a conectar el pin al polo positivo de la fuente de alimentación dentro del chip. Mida el voltaje entre el pin y el terminal negativo de la fuente de alimentación, y el voltímetro muestra 5V.

Cuando se configura el pin `OUTPUT` en un nivel bajo `LOW`, el pin equivale a estar conectado al polo negativo de la fuente de alimentación. Cuando se mide el pin y el polo negativo de la fuente de alimentación nuevamente, el voltímetro no tiene salida. Al medir el voltaje entre el polo positivo de la fuente de alimentación y el pin, el voltímetro muestra 5V.

Botón pulsador

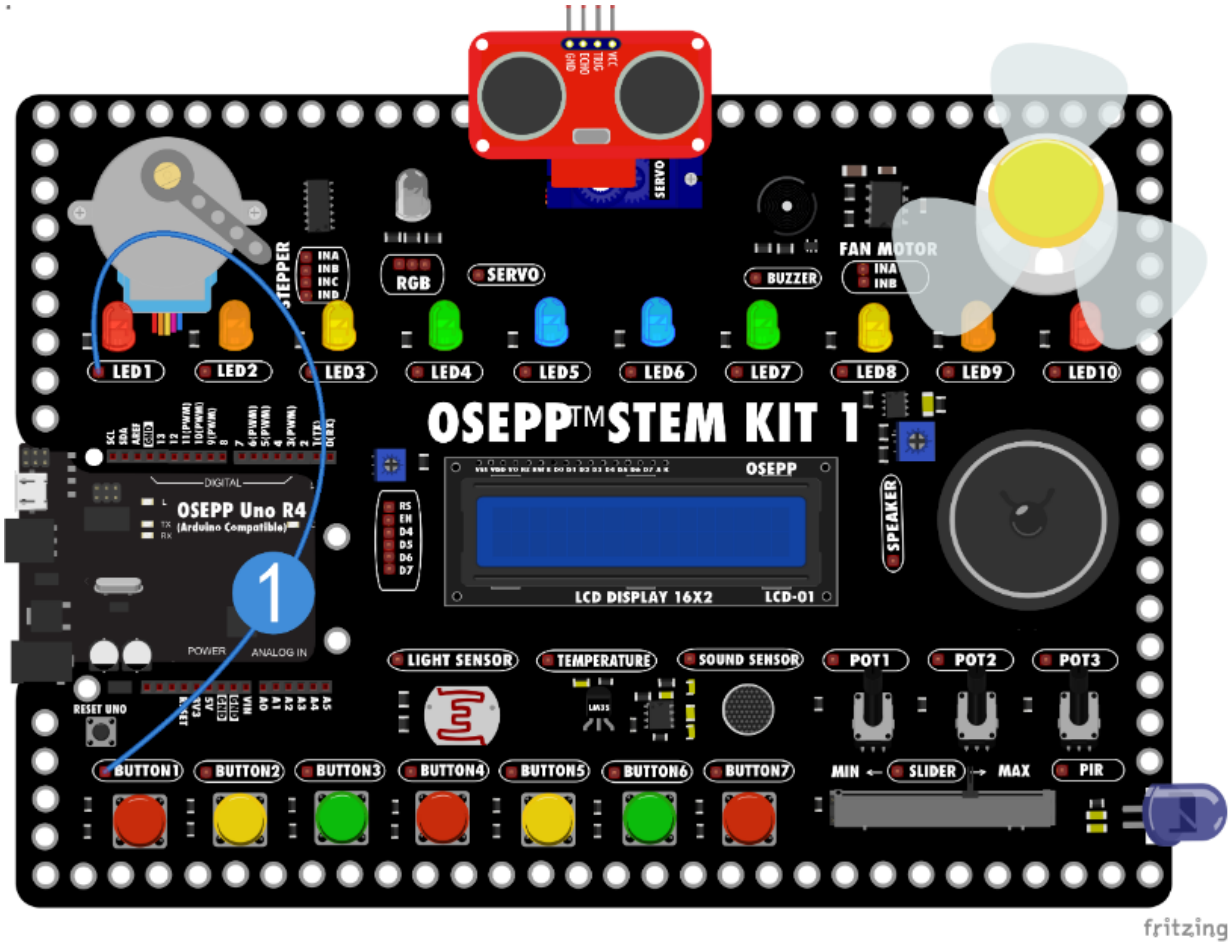
1. Conectar un botón a un LED

Un interruptor pulsador es un interruptor electrónico cuya estructura interna se basa en el cambio de fuerza de un resorte metálico para lograr el encendido y el apagado. Un interruptor de botón pulsador es sólo un tipo de interruptor. Los interruptores están presentes en todas partes en nuestra vida diaria. Cada luz de nuestra casa se controla con uno o dos interruptores, y cada aparato eléctrico tiene un interruptor para controlar el encendido y apagado.

Hay siete botones en la placa de aprendizaje STEM KIT. El siguiente experimento muestra cómo utilizar los botones de la placa de aprendizaje para controlar el LED.

Conexión

Conecte los terminales del pulsador `Button1` a los terminales del `LED1`.



Resultados de la operación

El LED de la placa de aprendizaje STEM KIT se enciende cuando está encendida. Cuando se presiona el botón `Button1`, el LED se apaga y, cuando se suelta, se enciende.

Análisis

Este parece ser el efecto opuesto al que conocemos habitualmente, que es iluminarse al presionarlo y apagarse al soltarlo. Esto se debe al diseño del circuito de la placa de aprendizaje STEM KIT. Para garantizar un uso seguro y confiable, la placa de desarrollo y los componentes que contiene no se dañarán incluso si los cables se conectan incorrectamente, por lo que el interruptor de la placa de aprendizaje adopta este diseño de circuito.

Cuando conectamos el botón al el LED y encendemos la placa de aprendizaje, la corriente fluirá a través del LED. Cuando se presiona el botón, la corriente fluye directamente al terminal negativo a través del botón y el LED no tiene corriente, por lo que no se encenderá.

2. Botón conectado a Arduino para controlar un LED

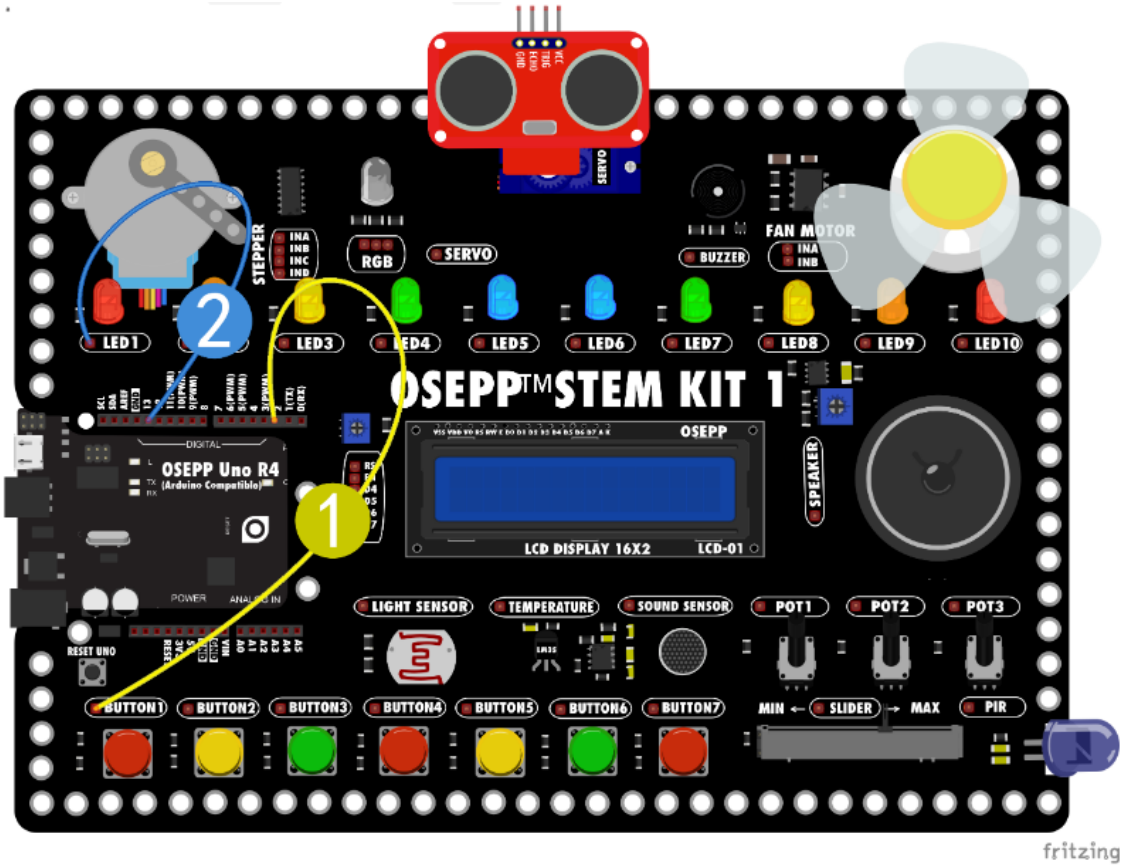
Podemos conectar un LED a Arduino, y también podemos conectar un interruptor a Arduino. El LED es un dispositivo de salida que puede indicar el estado de la salida. El interruptor es un dispositivo de entrada y Arduino puede determinar el estado del interruptor a través del programa.

Este experimento utiliza un programa para leer el estado del botón y luego controlar el estado de salida del LED.

Conexión

1. Conecte el pulsador Button1 al pin 2 de la placa UNO. Los pines 0 y 1 se utilizan para la comunicación con la computadora y la descarga de programas. Por lo general, no se utilizan en la aplicación.

2. Conecte el pin del LED1 al pin 13 de la placa UNO.



Construcción de programas

Programa Arduino

```
void setup()
{
    //button1
    pinMode(2, INPUT); //定义2号引脚为输入模式
    //led1
    pinMode(13, OUTPUT); //定义13号引脚为输出模式
}

void loop()
{
    digitalWrite(13, digitalRead(2) == LOW); //开关按下时LED点亮
}
```

Resultados de la operación

Cuando se presiona el botón Button1, el LED se enciende y cuando se suelta, el LED se apaga.

Análisis

- Estado de entrada del pin digital:

Cuando un pin digital de Arduino se configura como entrada `INPUT`, solo puede leer dos estados: `HIGH` ALTO o `LOW` BAJO.

La función `digitalRead()` se utiliza para monitorear el voltaje del pin de entrada `INPUT` y regresará `HIGH` si el voltaje es alto y si es bajo regresará `LOW`. Pero en la práctica, en términos generales, un voltaje superior a la mitad del voltaje de alimentación del chip se considera un nivel alto, mientras que un voltaje inferior a este se considera un nivel bajo.

Si el pin no está conectado (flotante), `digitalRead()`el valor devuelto no está definido y puede ser positivo `HIGH` o negativo `LOW`.

Hay otro modo para definir la entrada:

Código:

```
void setup()
{
    pinMode(3, INTPUT_PULLUP) //定义3号引脚为带上拉电阻的输入模式
}
```

3. Botón conectado a Arduino para controlar un LED. Interruptor con retardo

El interruptor de retardo de tiempo es un nuevo tipo de interruptor electrónico automático de retardo de tiempo desarrollado para ahorrar recursos eléctricos. Ahorra energía y es conveniente. Se utiliza principalmente en escaleras, baños y otros lugares. A continuación haremos un cambio de retardo de tiempo.

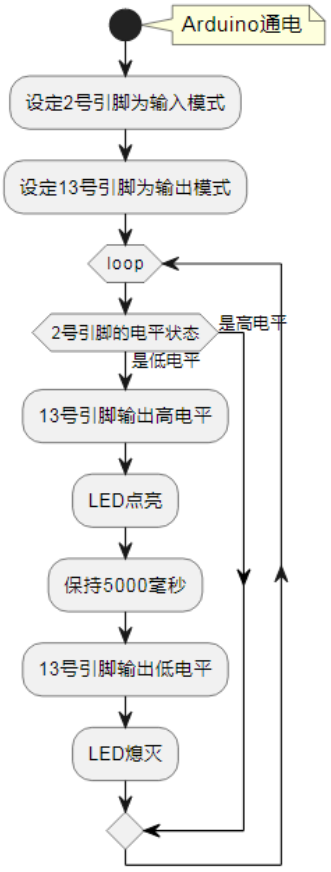
Este experimento se implementa a través del programa: cuando se presiona el botón `Button1`, el LED se enciende y se apaga después de 5 segundos.

Conexión

Continúe utilizando el circuito del experimento anterior con sólo unos pocos cambios en el programa.

Construcción de programas

Primero entendamos el flujo de trabajo del programa Arduino:



El interruptor tiene dos estados: presionado y no presionado. Cuando se presiona, es un nivel bajo, y cuando no se presiona, es un nivel alto (El interruptor en la placa de aprendizaje está conectado a una resistencia pull-up, y el nivel predeterminado es alto cuando el interruptor no está presionado).

El programa puede decidir si encender el LED evaluando los dos estados del interruptor.

El interruptor está conectado al pin 2. Cuando se presiona el interruptor, el pin 2 está en un nivel bajo. En este momento, el programa se ejecuta para encender el LED, es decir, el pin 13 emite un nivel alto.

Después de encenderse, el programa se detiene durante 5 segundos, luego el pin 13 emite un nivel bajo y el LED se apaga.

Si no se presiona el interruptor, el pin 2 siempre está alto. El programa omitirá esta sección de código directamente.

Según el diagrama de flujo, finalmente obtenemos el programa.

Programa Arduino.

```
void setup() {
    // button1
    pinMode(2, INPUT);  //定义2号引脚为输入模式;
    // led1
    pinMode(13, OUTPUT);  //定义13号引脚为输出模式;
}
void loop() {
    if (digitalRead(2) == LOW) {  //如果开关被按下
        digitalWrite(13, HIGH);   // 13号引脚输出高电平
        delay(5000);              //延时5000毫秒
        digitalWrite(13, LOW);    // 13号引脚转换为低电平
    }
}
```

Resultados de la operación

Si se presiona el interruptor, el LED se iluminará durante 5 segundos y luego se apagará hasta la próxima vez que se presione el interruptor.

Análisis

Sentencia `if`, si... ejecutar...

Si se presiona el botón pulsador, se encenderá el LED. El encendido del LED significa que se ejecuta la acción.

Una sentencia `if` es una parte de un lenguaje de programación que determina si se cumple una condición dada. Determina si se debe ejecutar el programa siguiente en función del resultado de la determinación (verdadero o falso). Si la condición es verdadera, se ejecuta; de lo contrario, se omite esta parte.

Potenciómetro

Un potenciómetro es una resistencia variable. Generalmente está compuesto por una resistencia y un sistema giratorio o deslizante, es decir, un contacto móvil se mueve sobre la resistencia para obtener una salida de tensión parcial.

Los potenciómetros son ampliamente utilizados y se utilizan en muchos productos, como el ajuste de volumen en audio, el ajuste de velocidad en ventiladores, el ajuste de brillo en luces, etc. La función del potenciómetro es ajustar el voltaje y la corriente.

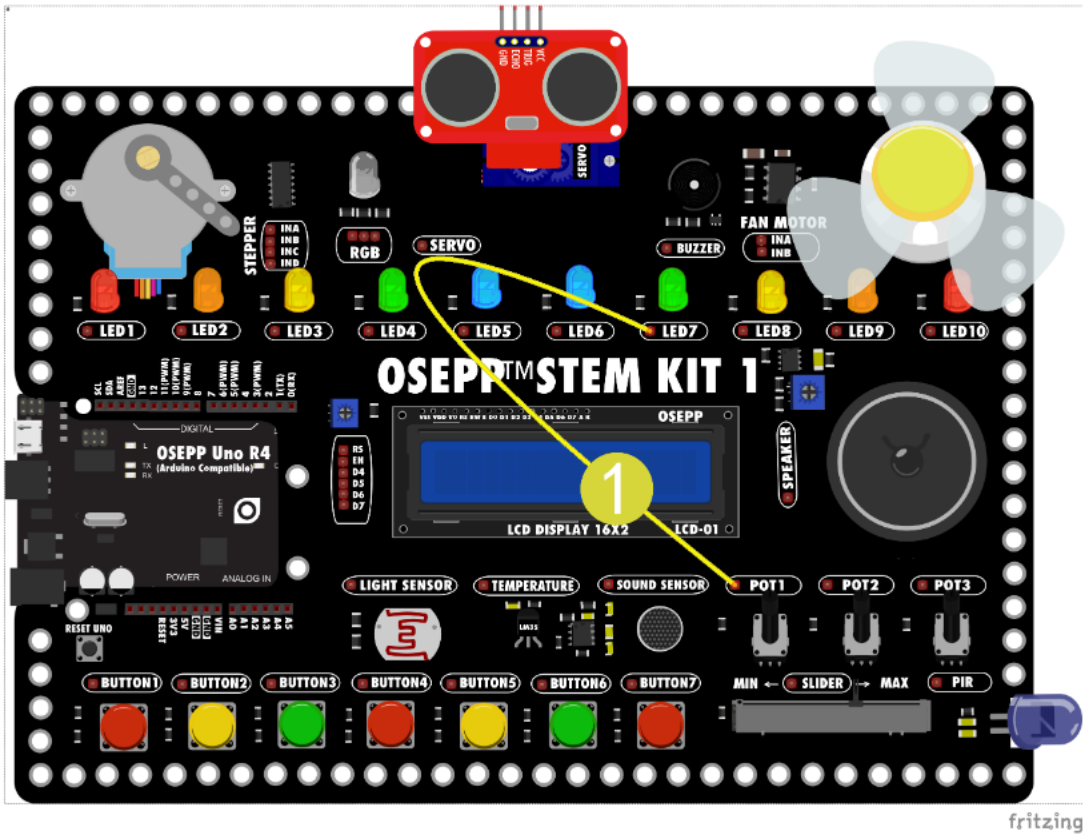
Características estructurales del potenciómetro: el cuerpo de la resistencia del potenciómetro tiene dos extremos fijos. Al ajustar manualmente el eje giratorio o la manija deslizante para cambiar la posición del contacto móvil en el cuerpo de la resistencia, se cambia el valor de la resistencia entre el contacto móvil y cualquier extremo fijo, modificando así el voltaje y la corriente.

1. Conexión del potenciómetro

En este experimento, el potenciómetro se conecta directamente al LED y luego se gira el potenciómetro, de modo que las características del potenciómetro se puedan comprender intuitivamente.

Conexión

Utilice un cable Dupont hembra a hembra para conectar los terminales del LED `LED7` a los terminales del potenciómetro `POT1`.



Resultados de la operación

A medida que gira el potenciómetro `POT1`, el brillo del LED cambia.

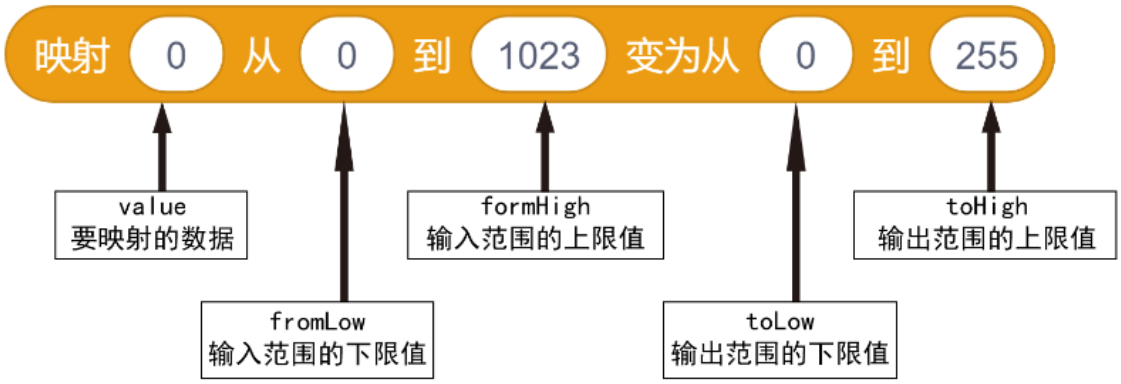
A través de este experimento, aprendimos sobre los potenciómetros. Un potenciómetro es simplemente una resistencia variable. Al ajustar el potenciómetro se modifica la resistencia del circuito, lo que modifica el voltaje y la corriente que se cargan en el extremo de carga.

2. Potenciómetro conectado a Arduino. Mapeo de brillo

En este experimento, se conecta un potenciómetro a Arduino para ajustar el brillo de un LED mediante **el mapeo de una función**

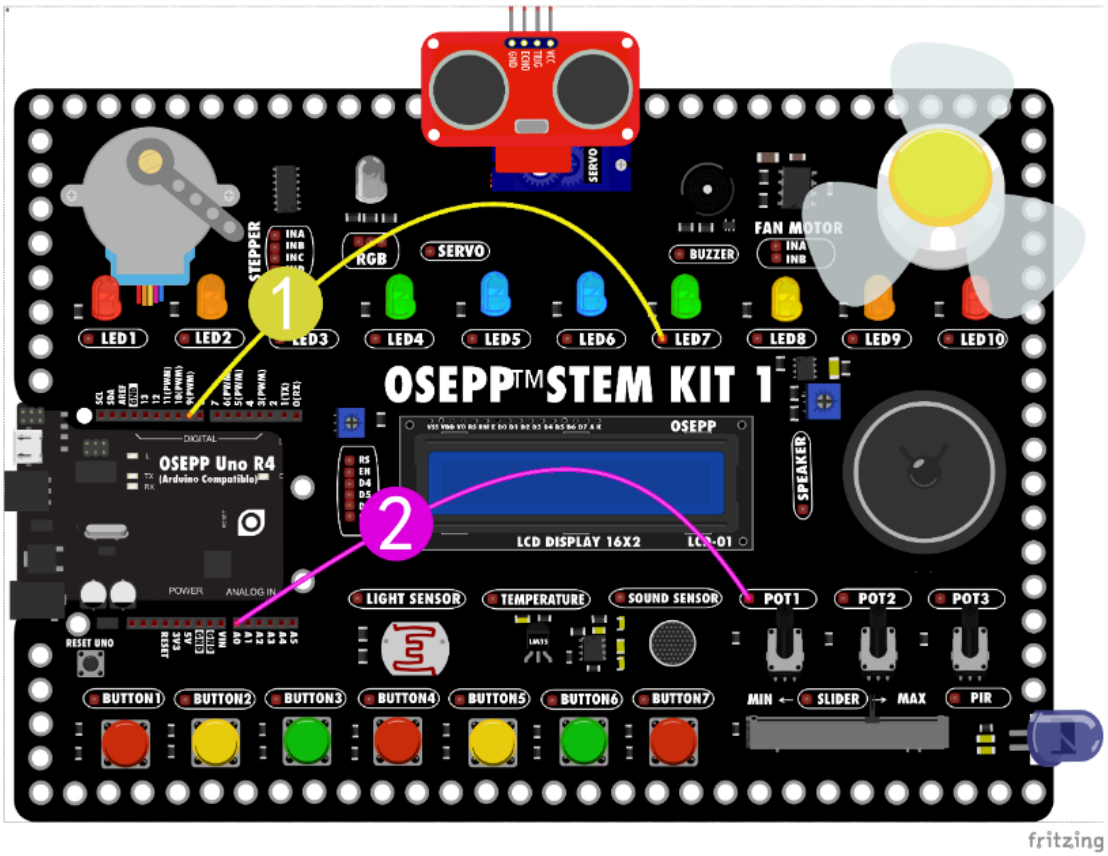
Bloques de funciones de mapeo

Función de mapeo: `map()`
Descripción: Mapea datos de un rango a otro.
Prototipo de función: `map(value, fromLow, fromHigh, toLow, toHigh)`
icono de bloque de construcción:



Conexión

1. Conectar el LED7 al pin 9 de la placa UNO.
2. Conectar el potenciómetro POT1 al pin **A0** de la placa UNO.



Construcción de programa

Programa Arduino

```
void setup()
{
    //potentiometer1
    pinMode(A0, INPUT); //定义A0引脚为输入模式
    //led1
    pinMode(9, OUTPUT); //9
}

void loop()
{
    digitalWrite(9, map(analogRead(A0), 0, 1023, 0, 255)); //把A0的值0-1023映射到0-255
}
```

Resultados de la operación

Girando el potenciómetro, cambiará el brillo del LED. Cuando se gira el potenciómetro hacia la izquierda, el LED cambia de brillante a opaco, y cuando se gira hacia la derecha, el LED cambia de opaco a brillante.

Análisis

- Pines analógicos

Pines analógicos: `A0-A7` con el convertidor analógico a digital (ADC) integrado de Arduino, la cantidad analógica en un pin `A0-A7` puede informar un valor entre `0-1023`, que se asigna a un rango de `0 a VCC` (el voltaje positivo en la fuente de alimentación, el cual es `5V` aquí). Los valores recopilados de `0-5V` en `A0-A7` se pueden enviar como `0-1023` al programa para su procesamiento.

Función `analogRead()`

`analogRead(pin)`se utiliza para leer el valor de voltaje de la magnitud analógica del pin analógico `A0-A7`. El parámetro `pin` representa el pin para el que se obtendrá el valor de voltaje analógico. La función devuelve un número entero entre `0-1023`.

`0V` obtiene el valor de `0`, `2.5V` obtiene el valor de `512`, `5V` obtiene el valor de `1023`.

- Pines PWM

Pin PWM: pin digital que suele estar marcado con # o *. La modulación por ancho de pulso es una técnica que utiliza la salida digital de un microprocesador para controlar circuitos analógicos. El nivel de salida es `0~255`.

El Arduino Uno tiene seis pines utilizados para PWM (pines digitales `3`、`5`、`6`、`9`、`10`、`11`).

- Análisis de aplicaciones

El nivel de salida del pin PWM es `0-255` y el valor de entrada en `A0` es `0-1023`.
Queremos mapear los valores `0-1023` para `0-255`.

La relación entre entrada y salida también se puede controlar cambiando **la relación de mapeo**. Por ejemplo, si quiere que el LED alcance el valor más brillante cuando el potenciómetro solo está girado hasta la mitad, entonces `0-1023` cámbielo a `0-512`. Si solo desea que el LED alcance la mitad del brillo, puede cambiar `0-255` a lo siguiente `0-128`.

Impresión del puerto serial de Arduino

Arduino tiene un hardware llamado puerto serial, el cual puede utilizar los pines número `0` (recibir) y el número `1` (enviar) para transmitir información de una manera específica. Los ordenadores suelen utilizar interfaces USB, por lo que también hay un chip en el Arduino Uno que convierte el puerto serie en una interfaz USB. De esta forma, el controlador puede enviar información al puerto serie, que luego se convierte en una señal USB y luego el ordenador recibe la información y la muestra.

El puerto serie también necesita ser configurado antes de su uso, principalmente la velocidad, llamada tasa de baudios. La directiva de configuración es `Serial.begin(speed)`,speed es la velocidad que se debe establecer, generalmente se establece en 115200.

La instrucción para que el puerto serial envíe datos es `Serial.print(val)`, val es la información a transmitir, que puede ser texto o datos.

`Serial.println(val)`, agregue un símbolo de salto de línea después del mensaje y los mensajes subsiguientes se mostrarán en una nueva línea. Se puede alternar entre imprimir/imprimir y envolver mediante la casilla de verificación.

Conexión

Conecte el potenciómetro deslizante `slider1` a los pines de `A0` de la placa UNO. El potenciómetro deslizante también es un tipo de potenciómetro.

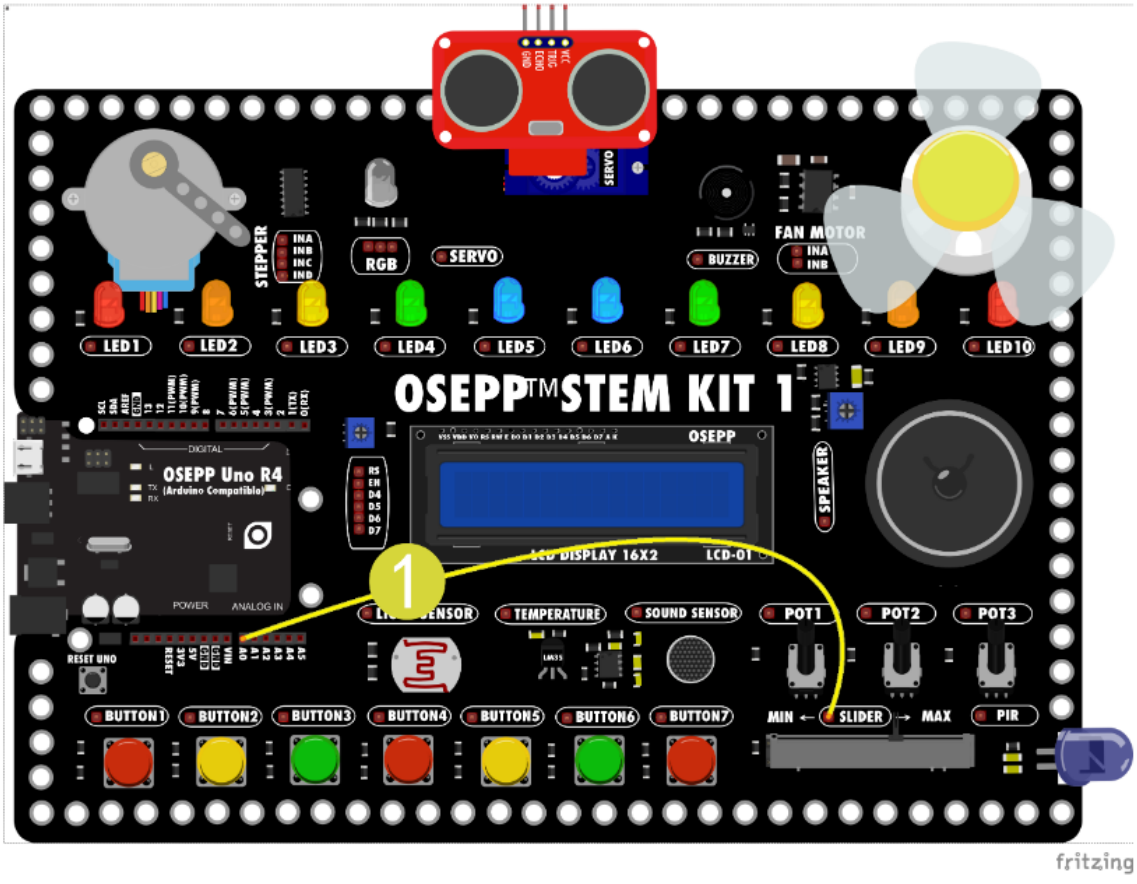
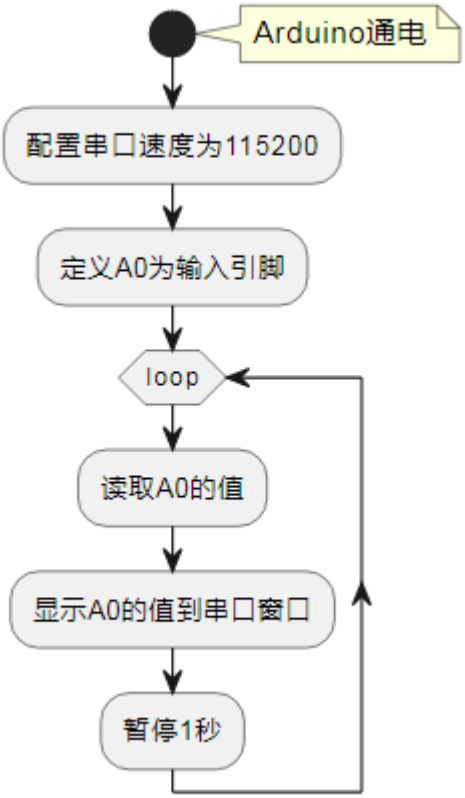


Diagrama de flujo de impresión del puerto serie



Construcción de programa

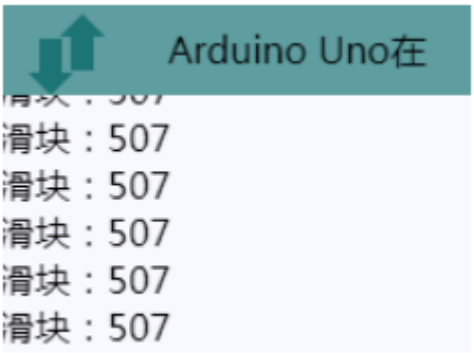
Programa Arduino

```
void setup()
{
    //slider1
    pinMode(A0, INPUT); //定义A0为输入模式
    Serial.begin(115200); //设置串口波特率
}

void loop()
{
    Serial.print("滑块: ");          //打印指令的文字
    Serial.println(analogRead(A0)); //打印A0的值并换行
    delay(1000);                    //延时1000毫秒
}
```

Resultados de la operación

Después de cargar el programa, haga clic en el icono de abajo. 

Cuando el icono cambie a  , podrá ver los datos de salida de impresión en la ventana siguiente.



Análisis

La impresión en serie (Monitor Serial) le permite informar resultados de su microcontrolador. Usando el monitor serial puedes obtener información sobre el estado de los sensores y entender qué está pasando en el circuito y el código mientras se ejecuta.

LCD

La pantalla de cristal líquido LCD1602 es un módulo de pantalla de caracteres ampliamente utilizado. Consiste en una pantalla de cristal líquido de caracteres, un circuito principal de control y su circuito de control extendido, así como una pequeña cantidad de resistencias, condensadores y piezas estructurales ensambladas en una placa PCB.
El módulo LCD de caracteres es un LCD de matriz de puntos especialmente utilizado para mostrar letras, números y símbolos, y los módulos más utilizados son 16×1, 16×2, 20×2 y 40×2.

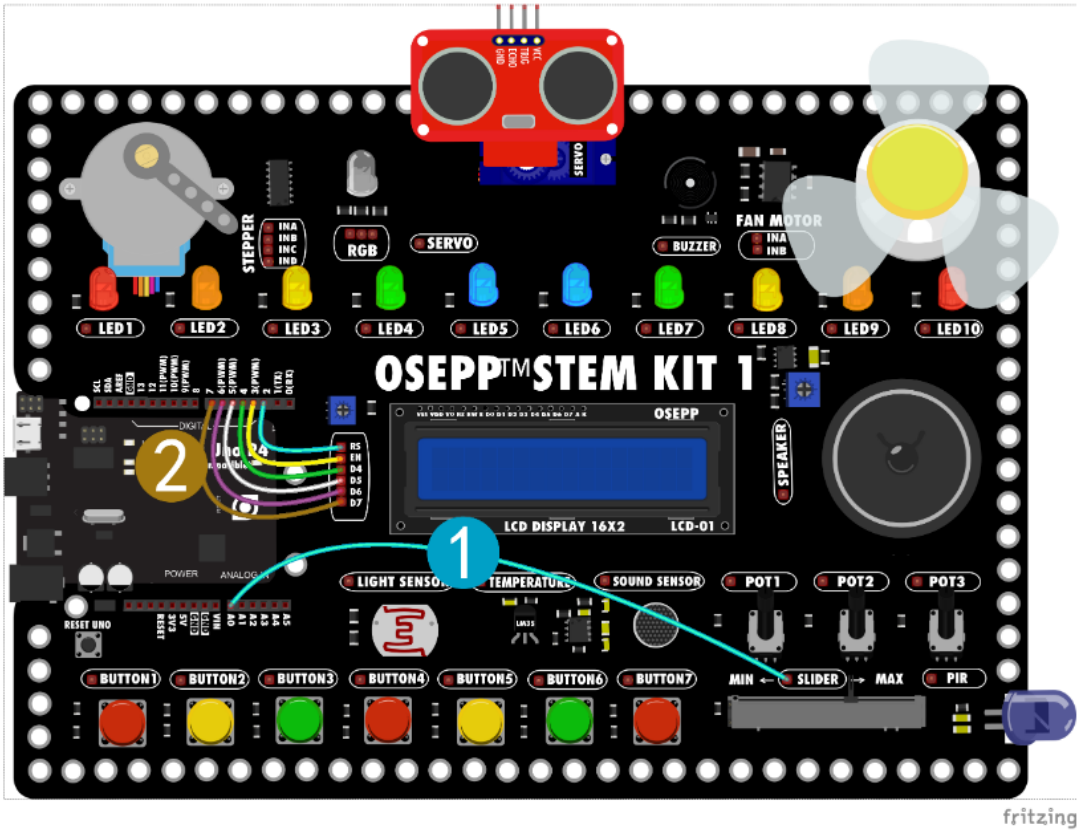
1. Conexión de la LCD

La pantalla de cristal líquido LCD1602 utilizada en la placa de aprendizaje STEM KIT puede mostrar 16x2 caracteres. (Nota: un carácter chino ocupará 2 anchos de carácter, pero para mostrar caracteres chinos se requiere una biblioteca de fuentes. Sin soporte de fuentes, solo se puede mostrar en inglés).

Si necesitamos saber el valor del potenciómetro, podemos usar la pantalla LCD para mostrarlo. Con una sencilla configuración, la pantalla LCD puede mostrar caracteres.

Conexión

1. Conecte el potenciómetro deslizante `slider1` al pin `A0` de la placa UNO.
2. Las interfaces del LCD `RS-D7` van conectadas a los pines `2~7` de la placa UNO respectivamente.



Construcción de programas

Programa Arduino

```
#include <LiquidCrystal.h>

LiquidCrystal lcd1(2, 3, 4, 5, 6, 7); //定义LCD的引脚

void setup()
{
    lcd1.begin(16, 2); //LCD初始化
    //slider1
    pinMode(A0, INPUT); //定义滑块电位器连接的引脚为输入模式
}

void loop()
{
    lcd1.clear();               //清屏
    lcd1.setCursor(0, 0);       //设置光标位置在0行、0列（左上角）
    lcd1.print(analogRead(A0)); //显示滑块电位器当前值
    delay(1000);                //延时1000毫秒
}
```

Resultados de la operación

El valor del potenciómetro deslizante se mostrará en la pantalla LCD y el valor está entre `0-1023`. Ajuste el potenciómetro deslizante y el valor cambiará en consecuencia. Agregamos un retraso de 1 segundo al final del programa. El programa se pausará durante este retraso de 1 segundo, por lo que la respuesta se sentirá muy lenta cuando la ajustemos. Si la pantalla no es clara, puede utilizar un destornillador para ajustar el potenciómetro junto al terminal LCD hasta que los caracteres de la pantalla sean claros.

Análisis

En este momento, la pantalla LCD actualizará el valor de voltaje del potenciómetro deslizante una vez por segundo. Vemos que en la pantalla LCD se muestran los números de `0-1023`, lo cual significa que Arduino mapea los valores de voltaje de `0-5V` leídos desde `A0` para llevarlos al display. Si desea visualizar el voltaje, debe calcularlo mediante el programa. A continuación, se indican dos métodos de cálculo:

Método 1:

- Use el valor de `1023` para expresar `5000mV`. Cada unidad es representa `5000/1023mV` (es decir, `4.89mV`)
- El valor de `analogRead(A0)` multiplicado por cada unidad (es decir, `4.89mV`) es el valor real en milivoltios. La división entera por `1000` sin el punto decimal es el voltio entero y el resto calculado por división de módulo es la parte `mV`.

Programa Arduino

```
#include <LiquidCrystal.h>

LiquidCrystal lcd1(2, 3, 4, 5, 6, 7); //定义LCD的引脚

void setup()
{
    lcd1.begin(16, 2); //LCD初始化
    //slider1
    pinMode(A0, INPUT); //定义滑块电位器连接的引脚为输入模式
}

void loop()
{
    lcd1.clear();                                       //清屏
    lcd1.setCursor(0, 0);                               //显示光标定位
    lcd1.print((analogRead(A0) * (5000 / 1023)) / 1000); //计算整数部分
    lcd1.print(".");                                    //添加一个小数点
    lcd1.print((analogRead(A0) * (5000 / 1023)) % 1000); //取模除计算小数部分
    delay(1000);                                        //延时1000毫秒
}
```

En este momento, la pantalla LCD muestra el valor de voltaje real del control deslizante.

Método 2:

- asigne directamente el valor de `analogRead(A0)` a milivoltios. Con la división entera por `1000` y descartando el punto decimal se obtiene el voltio completo. El resto calculado por división de módulo es la parte `mV`.

Programa Arduino

```
#include <LiquidCrystal.h>

LiquidCrystal lcd1(2, 3, 4, 5, 6, 7); //定义LCD的引脚

void setup()
{
    lcd1.begin(16, 2); //LCD初始化
    //slider1
    pinMode(A0, INPUT); //定义滑块电位器连接的引脚为输入模式
}

void loop()
{
    lcd1.clear();                                          //清屏
    lcd1.setCursor(0, 0);                                  //显示光标定位
    lcd1.print(map(analogRead(A0), 0, 1023, 0, 5000) / 1000); //计算整数部分
    lcd1.print(".");                                       //添加一个小数点
    lcd1.print(map(analogRead(A0), 0, 1023, 0, 5000) % 1000); //用模除计算小数部分
    delay(1000);                                           //延时1000毫秒
}
```

Cuando no se establecen variables especiales, los programas Arduino operan con números enteros. Para calcular la parte decimal utilizamos **módulo (%)**, el resultado del módulo es el resto de un número dividido por otro número.

Matemáticas con Arduino

- Sumar dos números: $1+2=3$
- Restar el segundo número del primer número: $2-1=1$
- Multiplicar dos números: $1*2=2$
- División entera, dividir el primer número por el segundo número: $2/1=2$
- División módulo, cuando el primer número se divide por el segundo número: $5/3=1...2$ El resultado es igual a 2

Sensor de luz. Fotorresistencia

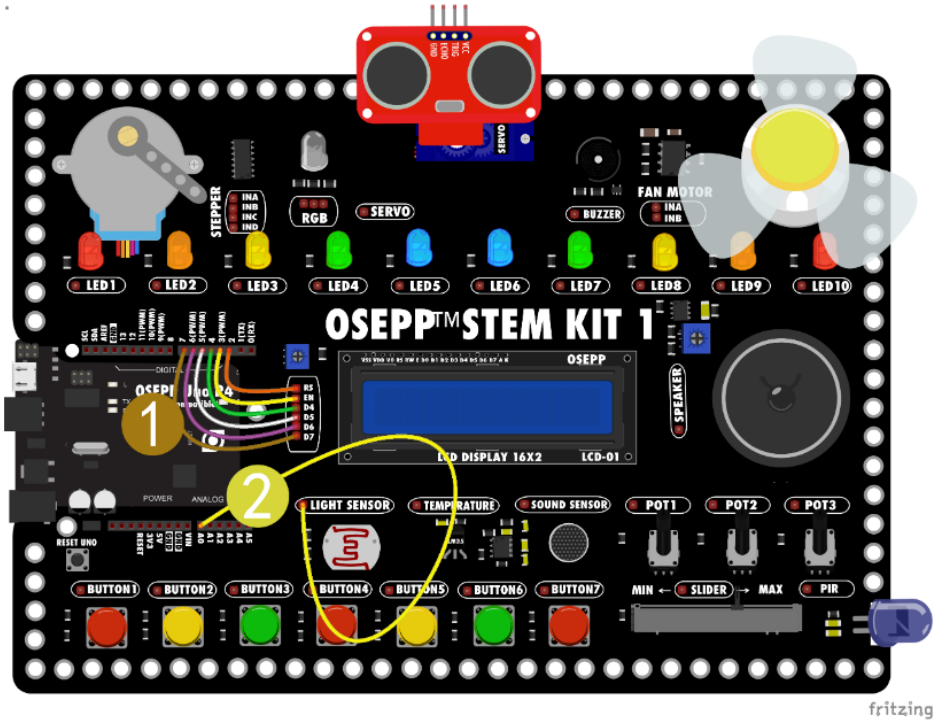
Los sensores de luz funcionan según el principio del efecto fotoeléctrico. El llamado efecto fotoeléctrico se refiere al fenómeno por el cual ciertas sustancias especiales pueden convertir la energía luminosa en energía eléctrica después de absorber luz. El efecto fotoeléctrico se puede dividir en dos tipos: efecto fotoeléctrico externo y efecto fotoeléctrico interno. El efecto fotoeléctrico externo se refiere al hecho de que, cuando se expone a la luz, los electrones pueden emitirse desde el interior de una sustancia hacia el exterior para generar fuerza eléctrica. Los fototubos y los tubos fotomultiplicadores son componentes fabricados en base al efecto fotoeléctrico externo. De la misma manera, en el interior de una sustancia se produce el efecto fotoeléctrico interno: cuando la luz incide sobre la sustancia, su resistividad interna cambia, modificando así la fuerza electromotriz. Los componentes fotoeléctricos como fotorresistores y fotocélulas se fabrican basándose en el efecto fotoeléctrico interno.

1. Mostrar el valor del sensor de luz en la LCD

El sensor de luz en la placa de aprendizaje es un fotorresistor. Cuando la luz incide sobre el sensor, la resistencia del sensor cambia y el voltaje cargado en el sensor también cambia en consecuencia y se refleja en los terminales. Utilizaremos un programa para leer este voltaje cambiante y mostrarlo en la pantalla LCD.

Conexión

1. Las interfaces del LCD `RS-D7` van conectadas a los pines `2~7` de la placa UNO respectivamente.
2. El terminal del sensor de luz `Light Sensor` se conectan al pin `A0` de la placa UNO.



Construcción de programa

Programa Arduino

```
#include <LiquidCrystal.h>

LiquidCrystal lcd1(2, 3, 4, 5, 6, 7); //定义LCD引脚

void setup()
{
  lcd1.begin(16, 2); //LCD初始化
  //light1
  pinMode(A0, INPUT); //定义光线传感器引脚A0为输入模式
}

void loop()
{
  lcd1.clear();               //清屏
  lcd1.setCursor(0, 0);       //显示光标定位
  lcd1.print(analogRead(A0)); //显示光线传感器的值
  delay(1000);                //延时1000毫秒
}
```

Resultados de la operación

En este momento, si bloquea el sensor de luz, verá que el valor que se muestra en la pantalla LCD cambia. Cuanto más débil sea la luz, menor será el valor; cuanto más fuerte sea la luz, mayor será el valor.

Análisis

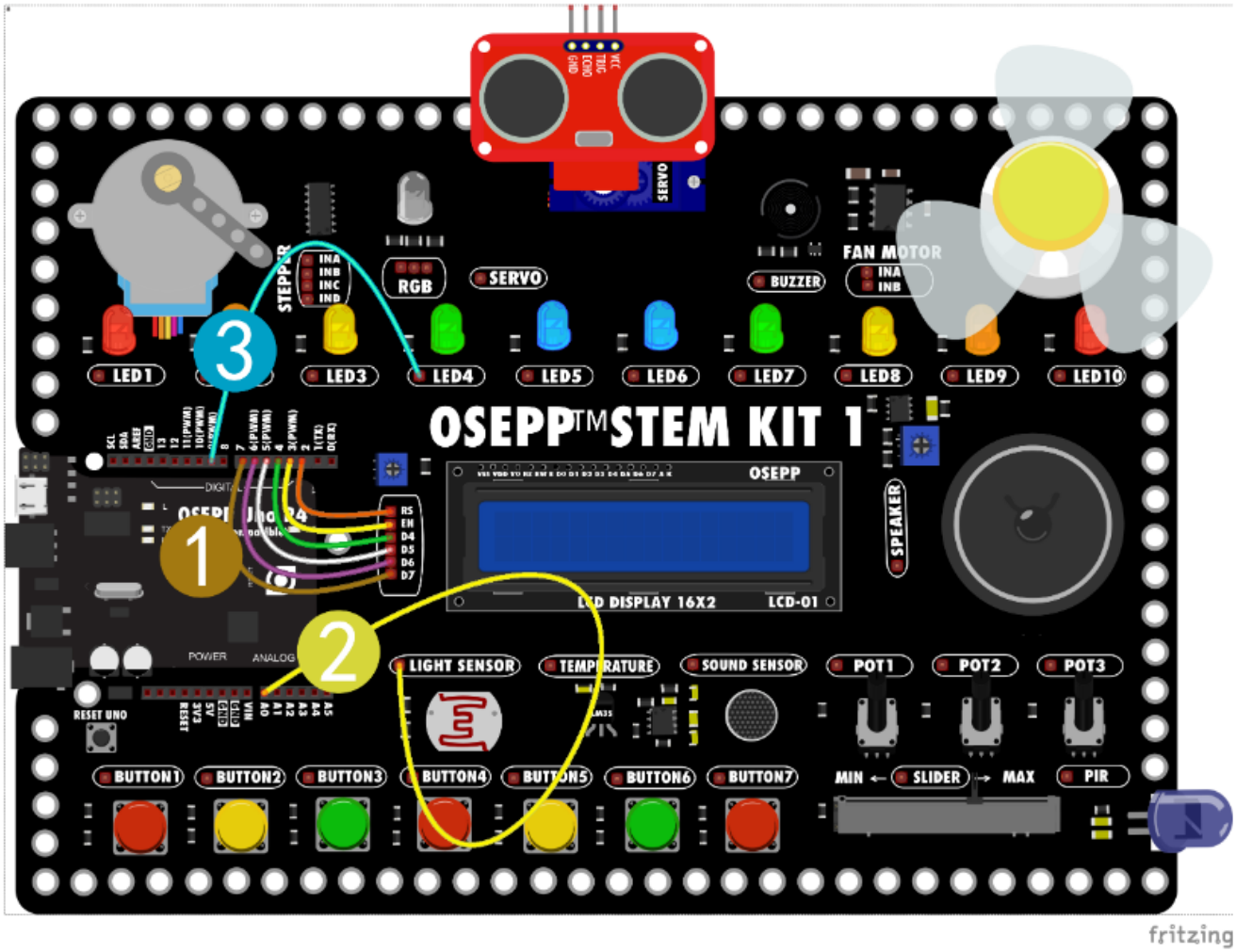
Un fotorresistor es una resistencia cuya resistencia disminuye a medida que aumenta la intensidad de la luz. Según su efecto fotoeléctrico interno, cuanto más intensa sea la luz, menor será la resistencia.

2. Sensor de luz: Control del brillo LED

El sensor de luz puede detectar cambios en la luz y reflejar los cambios en la luz a través de cambios en el voltaje. Luego podemos usar este cambio para crear un sensor de luz ambiental. Si la luz es más tenue, el LED será más brillante. Si hay suficiente luz, el LED no brillará.

Conexión

1. Las interfaces del LCD `RS-D7` están conectadas a los pines `2~7` de la placa UNO.
2. Conecte el terminal del sensor de luz `Light Sensor` al pin `A0` de la placa UNO.
3. Conecte el terminal del `LED4` al pin `9` de la placa UNO.



Construcción de programa

Programa Arduino

```
#include <LiquidCrystal.h>

LiquidCrystal lcd1(2, 3, 4, 5, 6, 7); //定义LCD引脚

void setup()
{
  lcd1.begin(16, 2); //LCD初始化
  //led1
  pinMode(9, OUTPUT); //定义LED引脚
  //light1
  pinMode(A0, INPUT); //定义光线传感器引脚
}

void loop()
{
  analogWrite(9, map(constrain(analogRead(A0), 100, 450), 100, 450, 255, 0));
  //把光线传感器当前读取在一个范围，且把这个范围映射到255-0来控制LED
  lcd1.clear();               //清屏
  lcd1.setCursor(0, 0);       //显示光标定位
  lcd1.print(analogRead(A0)); //显示光线传感器的值
  delay(1000);                //延时1000毫秒
}
```

Resultados de la operación

Cuando se bloquea la luz que llega al sensor, el LED se vuelve más brillante y, cuando se elimina la obstrucción, el LED se vuelve más tenue. Los valores de salida asignados aquí son llenados con `255-0` de grande a pequeño. Por lo tanto, cuando la luz es más débil, la luz del LED es más brillante y cuando la luz es más fuerte, la luz LED es más oscura.

Análisis

En el programa, limitamos los valores del sensor de luz a mínimo `100` y máximo `450` para evitar que el programa falle cuando los cambios de luz exceden el rango de valores de mapeo. Por supuesto, puedes cambiar este valor. Cuando la luz es más oscura, la pantalla LCD muestra el valor mínimo y cuando la luz es más brillante, la pantalla LCD muestra el valor máximo.

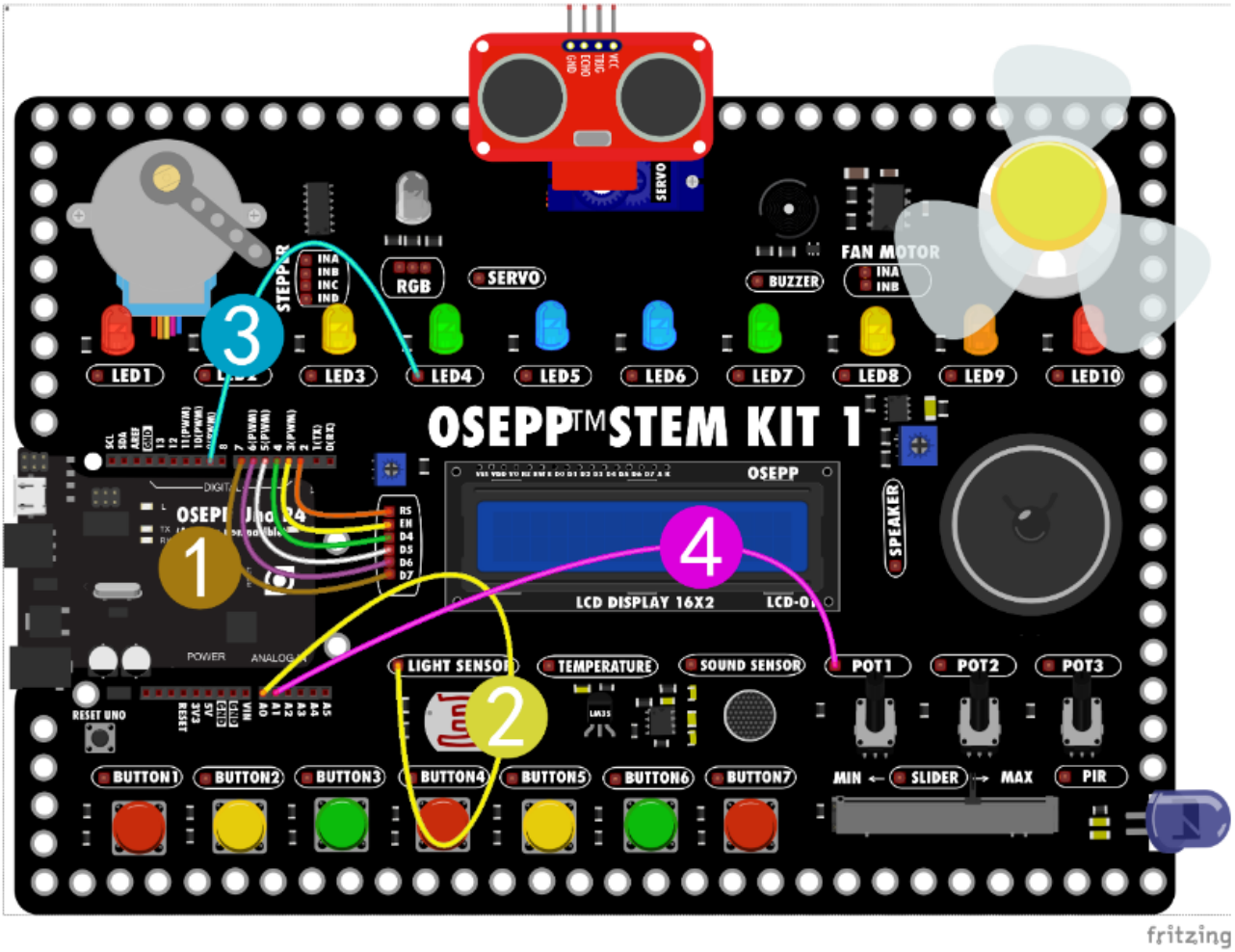
3. Lámpara LED con sensor y control de luz

Cuando cae la noche y la luz se atenúa, necesitamos encender las luces. ¿Podemos fabricar un interruptor más inteligente que encienda automáticamente la luz cuando oscurezca sin que tengamos que encenderlo manualmente? Si tenemos un sensor de luz, podemos hacer un interruptor de luz que detecte automáticamente la luz ambiental. La luz no se encenderá cuando haya mucha luz durante el día y se encenderá automáticamente cuando oscurezca por la noche.

Conexión

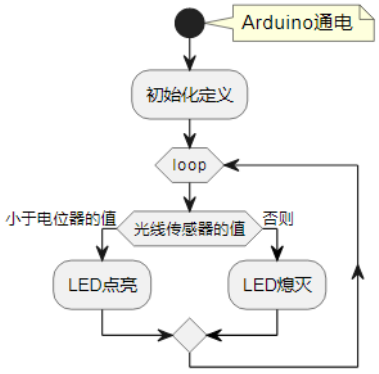
Esperamos que el brillo de la luz ambiental que hace que el LED se encienda en cada escena sea diferente, por eso agregamos un potenciómetro para ajustar el umbral del sensor de luz. Puede ajustar el valor del potenciómetro y compararlo con el valor del sensor de luz. Cuando la luz cambia, si el valor del sensor de luz es menor que el valor actual del potenciómetro, el LED activará el programa de iluminación.

1. Las interfaces del LCD RS-D7están conectadas a los pines 2~7 de la placa UNO.
2. Conecte el terminal del sensor de luz Light Sensor al pin A0 de la placa UNO.
3. Conecte el terminal del LED4 al pin 9 de la placa UNO.
4. Conecte el potenciómetro POT1 al pin A1 de la placa UNO.



Construcción de programa

Primero, entendamos el proceso.



Si el valor del sensor de luz (A0) es menor que el valor del potenciómetro (A1), el pin 9 emite un nivel alto y enciende el LED. De lo contrario, el pin 9 emite un nivel bajo y el LED se apaga.

Programa Arduino

```
#include <LiquidCrystal.h>

LiquidCrystal lcd1(2, 3, 4, 5, 6, 7); //定义LCD引脚

void setup()
{
  //led1
  pinMode(9, OUTPUT); //定义LED
  //light1
  pinMode(A0, INPUT); //定义光线传感器
  //potentiometer1
  pinMode(A1, INPUT); //定义电位器接到的引脚
  lcd1.begin(16, 2);  //LCD初始化
}

void loop()
{
  if (analogRead(A0) < analogRead(A1)) //比较两个值，如果光线传感器的值小于电位器的值
  {
    digitalWrite(9, HIGH); //点亮LED
  }
  else //否则
  {
    digitalWrite(9, LOW); //LED熄灭
  }
  lcd1.clear();               //清屏
  lcd1.setCursor(0, 0);       //显示光标定位
  lcd1.print("L:");           //显示字符，字符里的，要几半角
  lcd1.print(analogRead(A0)); //显示光线传感器的值
  lcd1.setCursor(0, 1);       //显示光标定位
  lcd1.print("P:");           //显示字符
  lcd1.print(analogRead(A1)); //显示电位器的值
  delay(1000);                //延时1000毫秒
}
```

Resultados de la operación

La primera línea de la pantalla LCD `L:xxx` muestra el valor actual del sensor de luz y la segunda línea `P:xxx` muestra el valor del potenciómetro, que es el umbral de activación que debemos establecer. Cuando la luz ambiental se oscurece, el valor del sensor se hará más pequeño y, mientras el valor del sensor sea menor que el valor establecido, el LED se iluminará. De lo contrario, el LED está apagado. Ajustando el valor del potenciómetro, puedes establecer la intensidad de luz con la que se activa el LED.

Análisis

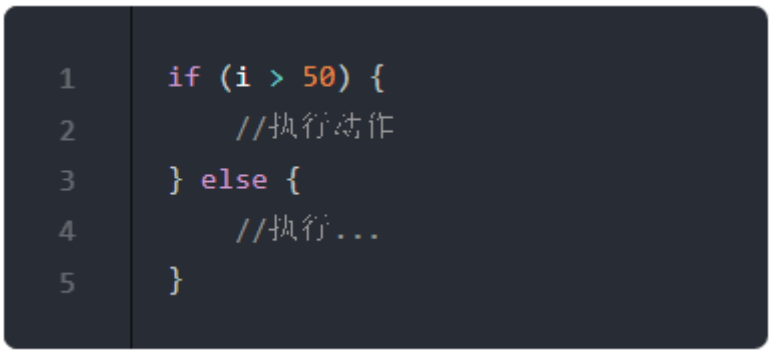
- Operaciones de comparación

La comparación es una operación matemática que se utiliza a menudo en programas.

Nombre del operador	Abreviaturas de operadores	Descripción	Ejemplo
Igual	==	Comprueba si los valores de dos operandos son iguales o no, si es así entonces la condición se vuelve verdadera.	(2 == 3) no es verdadero
No es igual a	!=	Comprueba si los valores de dos operandos son iguales o no, si los valores no son iguales entonces la condición se vuelve verdadera.	(2 != 3) es verdadero
Menor que	<	Comprueba si el valor del operando izquierdo es menor que el valor del operando derecho; si es así, la condición se vuelve verdadera.	(2 < 3) es verdadero
Mayor que	>	Comprueba si el valor del operando izquierdo es mayor que el valor del operando derecho; si es así, la condición se vuelve verdadera.	(2 > 3) no es cierto
Menor o igual a	<=	Comprueba si el valor del operando izquierdo es menor o igual que el valor del operando derecho; si es así, la condición se vuelve verdadera.	(2 <= 3) es verdadero
Mayor o igual a	>=	Comprueba si el valor del operando izquierdo es mayor o igual que el valor del operando derecho; si es así, la condición se vuelve verdadera.	(2 >= 3) no es verdadero

- Declaraciones condicionales

Si la condición es verdadera, entonces se realiza la acción A, de lo contrario, se realiza la acción B.



Este programa prueba si `i` es mayor que 50. Si es así, el programa realiza una acción específica. En otras palabras, las declaraciones entre llaves se ejecutarán si las declaraciones entre paréntesis son verdaderas. De lo contrario, el programa omite el código. (`else` Significa que cuando no se cumple la condición anterior, se ejecuta el código...)

LED RGB

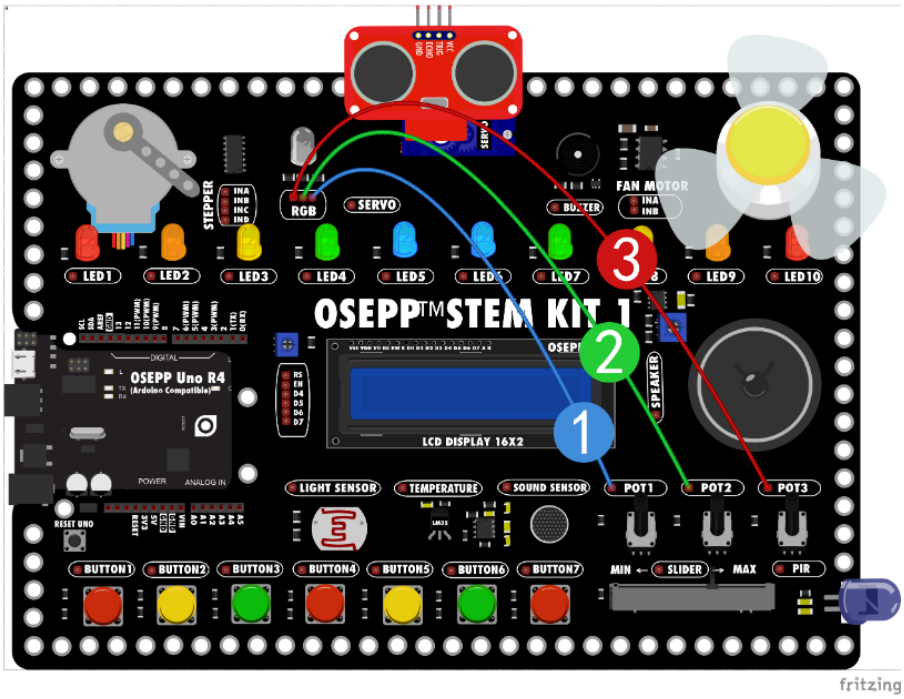
El módulo de color RGB, también conocido como módulo Tricolor o módulo de color rojo, verde y azul, es un LED de color aditivo que agrega los tres colores primarios rojo, verde y azul en diferentes proporciones para sintetizar varios colores de luz. Las luces LED de tres colores primarios RGB están compuestas por tres perlas de lámpara independientes: rojo, verde y azul. Hay cuatro pines comunes, un terminal común y tres terminales de control de color. Cualquier combinación de los tres colores puede producir otros colores. Por ejemplo, si el rojo y el verde están encendidos al mismo tiempo y el azul no está encendido, es amarillo; si el verde y el azul están encendidos al mismo tiempo y el rojo no está encendido, es cian; si el rojo y el azul están encendidos al mismo tiempo y el verde no está encendido, es magenta; si los tres colores están encendidos, se produce blanco.

1. LED RGB: control por potenciómetro

Hay 4 pines en el LED RGB, y el de la placa de aprendizaje es un LED RGB de cátodo común. Es como tener tres LED, rojo, verde y azul, y conectar sus cátodos entre sí y guiarlos hacia afuera con un solo cable. Luego, siempre que se introduzca un nivel alto en los tres terminales del ánodo, el diodo emisor de luz RGB se iluminará. El cátodo se ha conectado a la tierra de alimentación en la placa de aprendizaje `GND`, por lo que solo necesitamos conectar los tres terminales del ánodo.

Conexión

1. Conecte el terminal del potenciómetro `POT1` al terminal `B` del LED RGB
2. Conecte el terminal del potenciómetro `POT2` al terminal `G` del LED RGB
3. Conecte el terminal del potenciómetro `POT3` al terminal `R` del LED RGB



Resultados de la operación

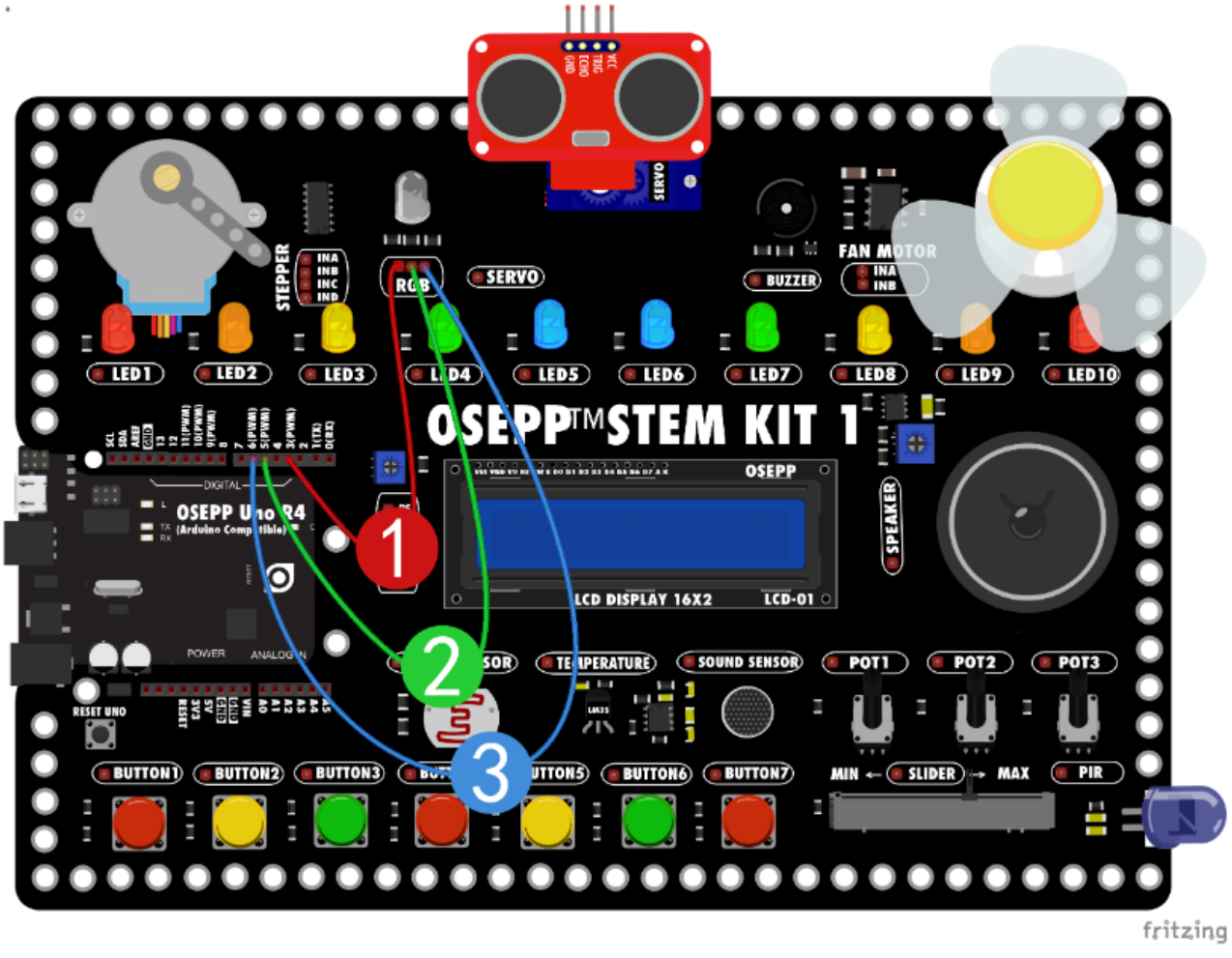
En este momento, puedes ver el cambio de color ajustando el potenciómetro. Al controlar el brillo de los LED individuales, puedes mezclar casi cualquier color que desees.

2. Control de LED RGB con Arduino

El efecto completo de mezcla de los tres colores primarios (rojo/verde/azul) se puede ajustar a través de la entrada de voltaje `PWM` en los tres pines R/G/B. Se pueden lograr efectos de iluminación geniales usando el control del programa Arduino.

Conexión

1. Conectar la terminal R del LED RGB al pin 3 de la placa UNO.
2. Conectar la terminal G del LED RGB al pin 5 de la placa UNO.
3. Conectar la terminal B del LED RGB al pin 6 de la placa UNO.



Construcción de programas

- Programa Arduino

```
void setup()
{
    //rgb1
    pinMode(3, OUTPUT); //定义3号引脚为输出模式
    pinMode(5, OUTPUT); //定义5号引脚为输出模式
    pinMode(6, OUTPUT); //定义6号引脚为输出模式
}

void loop()
{
    analogWrite(3, 255); //3号引脚输出红色全亮
    analogWrite(5, 0);   //5号引脚输出绿色0
    analogWrite(6, 0);   //6号引脚输出蓝色0
}
```

La tabla muestra los valores de los tres canales RGB. A medida que los completamos, podemos ver que los diodos emisores de luz RGB muestran los colores en la tabla. Siempre que se ingresen valores diferentes en los tres colores de RGB, el diodo emisor de luz RGB puede mostrar diferentes colores. Luego cambiamos este valor a un número aleatorio y el LED RGB puede mostrar colores aleatoriamente.

A continuación se muestra una tabla de datos de visualización de color, y puede ingresar números en el programa para controlar el color. Utilice Arduino para controlar el color de salida que necesita.

名称	颜色	色光		
		R	G	B
红色		255	0	0
黄色		255	255	0
绿色		0	255	0
青色		0	255	255
蓝色		0	0	255
品红色		255	0	255
栗色		128	0	0
橄榄色		128	128	0
深绿色		0	128	0
蓝绿色		0	128	128
深蓝色		0	0	128
紫色		128	0	128
白色		255	255	255
银色		192	192	192
灰色		128	128	128
黑色		0	0	0

- Programa Arduino

```
int R = 0; //定义整型变量R
int G = 0; //定义整型变量G
int B = 0; //定义整型变量B

void setup()
{
    //rgb1
    pinMode(3, OUTPUT); //定义3号引脚为输出模式
    pinMode(5, OUTPUT); //定义5号引脚为输出模式
    pinMode(6, OUTPUT); //定义6号引脚为输出模式
}

void loop()
{
    R = random(0, 256); //R等于0-256之间的随机数
    G = random(0, 256); //G等于0-256之间的随机数
    B = random(0, 256); //B等于0-256之间的随机数
    analogWrite(3, R);  //3号引脚输出R的随机数
    analogWrite(5, G);  //3号引脚输出G的随机数
    analogWrite(6, B);  //3号引脚输出B的随机数
    delay(500);         //延时500毫秒
}
```

Resultados de la operación

Después de cargar el programa en la placa UNO, el LED RGB cambiará de color aleatoriamente. El color cambia cada 0,5 segundos. Puedes cambiar el valor de retardo. Cuanto menor sea el valor, más rápido será el cambio.

Análisis

- Variables enteras: `int`

Una variable entera (int) es un tipo de variable que se puede utilizar para almacenar números entre -32768~+32767. En Arduino, el entero (int) es el tipo de variable comúnmente más utilizado. Una variable le dice al programa el tipo y rango del valor.

Hay variables enteras, variables enteras largas, variables enteras sin signo, variables enteras largas sin signo, variables de caracteres, variables de bytes y variables de punto flotante.

- Funciones aleatorias: `random()`

La función aleatoria `random(min, max)`genera un valor aleatorio.

Parámetros

- `min`: El límite inferior del número aleatorio generado (incluido este valor).
- `max`: El límite superior del número aleatorio generado (excluyendo este valor).
- Valor de retorno: Un valor aleatorio entre los valores mínimo `min` y máximo. `max -1`

Altavoz

Al altavoz también se le llama parlante. Es un transductor o componente electrónico que convierte señales electrónicas en sonido, y puede estar compuesto por uno o más grupos de audio. Todos los sonidos de los televisores y las grabadoras provienen de altavoces.

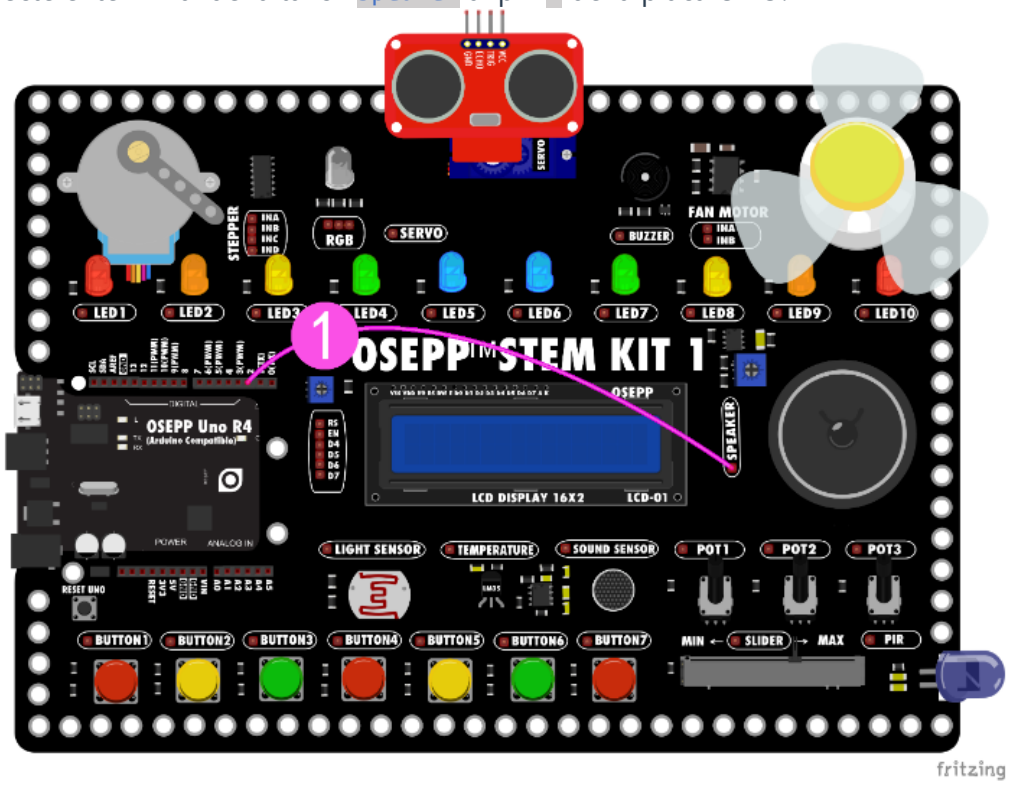
El altavoz está compuesto por un electroimán, una bobina y un diafragma de altavoz. Los altavoces convierten la frecuencia de la corriente eléctrica en sonido. Según los principios de la física, cuando la corriente pasa a través de una bobina y genera un campo electromagnético, la dirección del campo magnético sigue la regla de la mano derecha. Supongamos que el altavoz reproduce una melodía en C con una frecuencia de 256 Hz, es decir, vibra 256 veces por segundo. El altavoz emite una corriente alterna de 256 Hz y la corriente cambia 256 veces por segundo, emitiendo una frecuencia de melodía en C. El altavoz produce sonido cuando la bobina eléctrica vibra junto con la membrana del altavoz, empujando el aire circundante a vibrar. La frecuencia de las ondas sonoras audibles para el oído humano generalmente está entre 20 Hz y 20.000 Hz, por lo que los hablantes generales establecerán su frecuencia operativa dentro de este rango.

1. Activación del altavoz

Cómo hacer un sonido desde el altavoz. La frecuencia de funcionamiento del altavoz está entre 20-20KHz, por lo que siempre que Arduino emita la frecuencia dentro de esta banda de frecuencia, el altavoz puede emitir sonido.

Conexión

Conecte el terminal del altavoz `SpeaKer` al pin `2` de la placa UNO.



Construcción de programa

Programa Arduino

```
void setup()
{
    //speaker1
    pinMode(2, OUTPUT); //定义2号引脚为输出模式
}

void loop()
{
    tone(2, 131); //2号引脚输出131频率的波形
    delay(500);   //延时500毫秒
    noTone(2);    //停止声音
    delay(3000);  //延时3000毫秒
}
```

Resultados de la operación

Después de cargar el código, el altavoz emitirá un sonido `c3音调` durante 0,5 segundos y luego hará una pausa de 3 segundos.

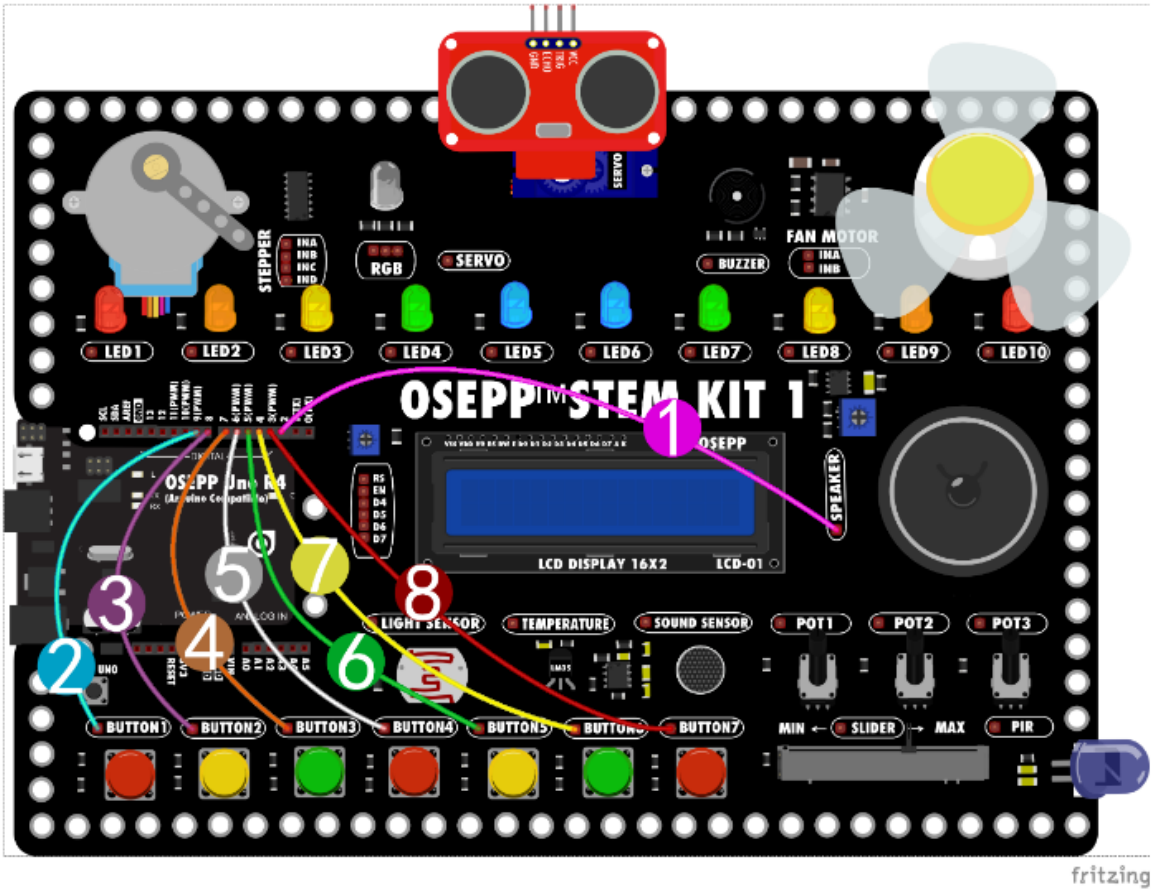
2. Teclado electrónico con altavoz

El experimento anterior hace que el altavoz emita un solo sonido, lo cual resulta demasiado aburrido. A continuación, queremos que el altavoz emita diferentes tonos. Y usa cada interruptor para controlar un tono, como un teclado electrónico.

Conexión

Conecte los cables de acuerdo con la siguiente tabla.

Número de serie	Terminales componentes	Número de pin placa UNO
1	Altavoz	2
2	Botón 1	9
3	Botón 2	8
4	Botón 3	7
5	Botón 4	6
6	Botón 5	5
7	Botón 6	4
8	Botón 7	3



Construcción de programas

- Programa Arduino

```
void setup()
{
  //speaker1
  pinMode(2, OUTPUT); //定义2号引脚为输出模式
  //button1
  pinMode(9, INPUT); //定义9号引脚为输入模式
  //button2
  pinMode(8, INPUT); //定义8号引脚为输入模式
  //button3
  pinMode(7, INPUT); //定义7号引脚为输入模式
  //button4
  pinMode(6, INPUT); //定义6号引脚为输入模式
  //button5
  pinMode(5, INPUT); //定义5号引脚为输入模式
  //button6
  pinMode(4, INPUT); //定义4号引脚为输入模式
  //button7
  pinMode(3, INPUT); //定义3号引脚为输入模式
}

```

```
void loop()
{
  if (digitalRead(9) == LOW) //如果9号引脚为低电平时
  {
    tone(2, 131); //2号引脚输出131频率的声音
  }
  else if (digitalRead(8) == LOW) //如果8号引脚为低电平时
  {
    tone(2, 147); //2号引脚输出147频率的声音
  }
  else if (digitalRead(7) == LOW) //如果7号引脚为低电平时
  {
    tone(2, 165); //2号引脚输出165频率的声音
  }
  else if (digitalRead(6) == LOW) //如果6号引脚为低电平时
  {
    tone(2, 175); //2号引脚输出175频率的声音
  }
  else if (digitalRead(5) == LOW) //如果5号引脚为低电平时
  {
    tone(2, 196); //2号引脚输出196频率的声音
  }
  else if (digitalRead(4) == LOW) //如果4号引脚为低电平时
  {
    tone(2, 220); //2号引脚输出220频率的声音
  }
  else if (digitalRead(3) == LOW) //如果3号引脚为低电平时
  {
    tone(2, 247); //2号引脚输出247频率的声音
  }
  else //否则
  {
    noTone(2); //停止声音
  }
}
```

Resultados de la operación

Solo hay 7 botones en el tablero de aprendizaje, por lo que configuraremos los primeros 7 tonos. Desde el botón 1 al 7, son Do-Re-Mi-Fa-Sol-La-Si.

Encuentra una canción sencilla y ve si puedes tocarla. `1-2-3-1`, `1-2-3-1`, `3-4-5` `3-4-5...`

Una pieza musical se compone de varias notas y cada nota corresponde a una frecuencia. La función `tone()`puede generar una señal PWM de frecuencia fija para impulsar el altavoz y hacer que emita sonido. La duración y el tono del sonido se pueden controlar mediante parámetros.

Hay dos formas de definir la duración del sonido. La primera es definir la duración del sonido a través de los parámetros de la función `tone()` y la otra es utilizar la función `noTone()` para detener el sonido.

Si no define la duración del sonido al usar la función `tone()`, entonces, a menos que detenga el sonido a través de la función `noTone()`, Arduino continuará generando señales de sonido a través de la función `tone()`.

Análisis

El Arduino sólo puede producir un sonido a la vez. Si un determinado pin de Arduino está generando una señal de sonido a través de una función `tone()`, entonces no es posible hacer que Arduino use otro pin para generar sonido a través de la función `tone()`.

- función `tone()`:

Código:

```
tone(pin, frequency)
tone(pin, frequency, duration)
```

Parámetros:

- **pin**: pin de sonido (este pin debe estar conectado al altavoz)
- **frecuencia**: frecuencia del sonido (unidad: Hz) – tipo entero sin signo
- **duración**: duración del sonido (unidad: microsegundos, este parámetro es opcional) – tipo entero largo sin signo.

Esta función no tiene valor de retorno.

Buzzer

Un buzzer o zumbador es un componente electrónico generador de sonido. Es un resonador electrónico con una estructura integrada, que se utiliza ampliamente como dispositivo generador de sonido en productos electrónicos como computadoras, impresoras, fotocopiadoras, alarmas, juguetes electrónicos, equipos electrónicos automotrices, teléfonos, temporizadores, etc.

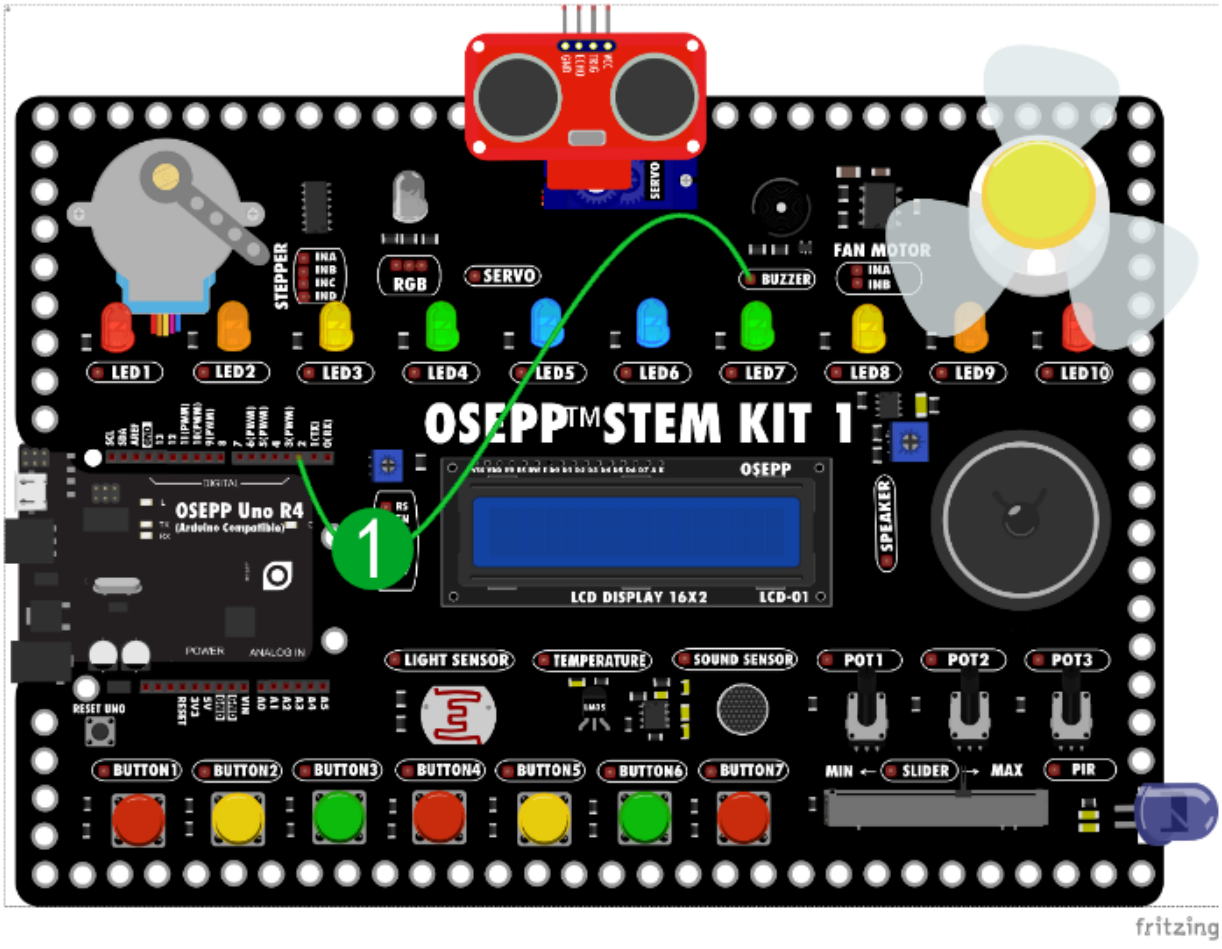
Los zumbadores se dividen en dos tipos: zumbadores activos y zumbadores pasivos. Los zumbadores activos tienen una fuente de oscilación interna, por lo que emitirán un sonido siempre que se encienda la alimentación.
El zumbador pasivo no tiene una fuente de oscilación interna y no puede ser accionado por una señal de DC para producir un sonido. Debe ser accionado por una señal de pulso con forma de onda de 2K a 5K.

1. Activación del buzzer

El tablero de aprendizaje STEM KIT utiliza un zumbador activo. El zumbador activo emitirá un sonido una vez que se encienda. Si lo conectamos a Arduino, emitirá un sonido siempre que la salida de nivel alto esté configurada en el programa.

Conexión

Conecte el terminal del zumbador `BUZZER` al pin `2` de la placa UNO.



Construcción de programa

- Programa Arduino

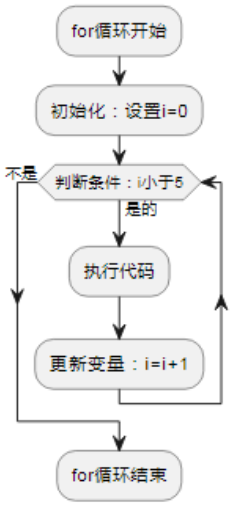
```
int i = 0; //定义一个整型变量

void setup()
{
  //buzzer1
  pinMode(2, OUTPUT); //定义蜂鸣器引脚
}

void loop()
{
  for (i = 0; i < 5; i++) //循环5次
  {
    digitalWrite(2, HIGH); //蜂鸣器发声
    delay(20);             //延时20毫秒
    digitalWrite(2, LOW);  //蜂鸣器停声
    delay(500);            //延时500毫秒
  }
  delay(5000); //延时5秒
}

```

Resultados de la operación

El zumbador suena una vez cada medio segundo durante 5 veces consecutivas, luego hace una pausa de 5 segundos antes de ingresar nuevamente al ciclo.

Análisis

- Repetición de declaraciones utilizando un contador



El bucle `for` consta de tres partes: inicialización, prueba condicional e iteración (es decir, la declaración que se ejecuta al final de cada proceso del bucle), cada parte está separada por un punto y coma.
Código: `for (i = 0; i < 5; i++)`

Código solución: `int i = 0;` Inicialice la variable i en 0, `i < 5;` pruebe la variable para ver si es menor que 5, `i++` e incremente i.

PIR

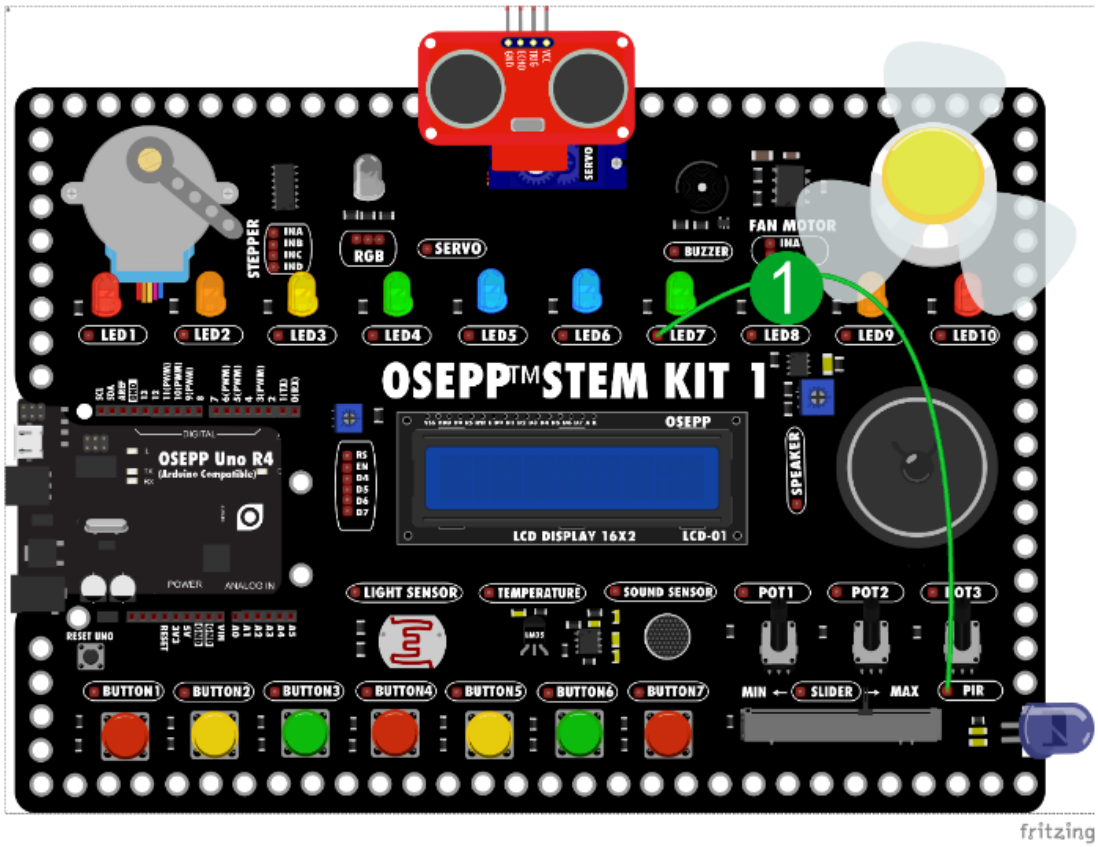
Todos los objetos calentados generarán rayos infrarrojos, y las longitudes de onda de los rayos infrarrojos irradiados son diferentes a diferentes temperaturas. Debido a que la temperatura del cuerpo humano se mantiene constante a 37 grados, emitirá rayos infrarrojos lejanos de 10um. El PIR se basa en esta característica para detectar cuerpos humanos. El principio es el siguiente: cuando los rayos infrarrojos de 10um emitidos por el cuerpo humano se enfocan sobre la superficie del sensor PIR a través de la lente Fresnel en el extremo frontal del PIR, el propio PIR tiene un filtro que solo permite la entrada de rayos infrarrojos lejanos de 8 a 14um. Cuando el sensor dentro del PIR recibe los rayos infrarrojos de 10nm, generará un movimiento de carga en el cristal piroeléctrico, lo que impulsará al MOS interno para generar una señal sinusoidal. Después de que esta señal sea amplificada por el amplificador operacional en el extremo posterior del PIR, la señal que exceda o sea inferior a un cierto valor de voltaje se puede extraer a través del comparador. Esta señal es la señal de pulso emitida por el PIR, que se puede conectar a la MCU u otros circuitos para lograr varias formas de control o alarma.

1. Sensor PIR conectado a un LED

El STEM KIT cuenta con un sensor PIR que, cuando detecta un cuerpo humano, genera un nivel alto en la terminal. Vuelve a un nivel bajo después de 1,5 segundos y el siguiente segundo es un tiempo de silencio, durante el cual no se activará nuevamente incluso si alguien está activo. Conectándolo a un LED, podemos ver fácilmente cómo funciona.

Conexión

Conecte el terminal del sensor `PIR` al terminal del `LED7` y encienda la alimentación. El PIR entra en el estado de autoprueba, que dura unos 3-5 segundos. Una vez completada la autoprueba, entra en el estado de prueba. Cuando el `PIR` detecta radiación infrarroja del cuerpo humano, se emitirá un nivel alto en el terminal durante 1 segundo, seguido de un período de silencio de 1 segundo. El `PIR` no realiza ninguna operación durante el período de silencio. El terminal emite un nivel bajo y luego ingresa nuevamente al estado de detección.

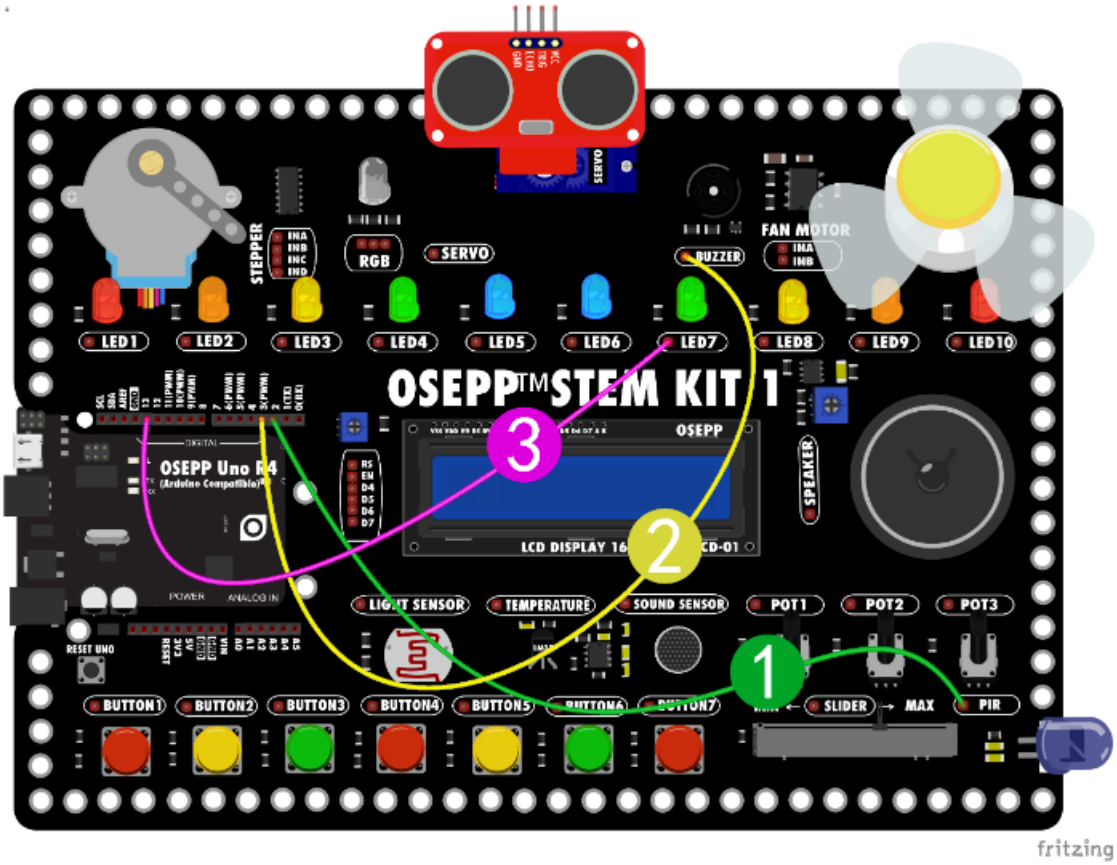


2. Timbre de bienvenida con sensor PIR

Cuando entramos en algunas tiendas, ¿escuchamos un mensaje de bienvenida, no? Se trata del uso de sensores `PIR` para detectar nuestro cuerpo humano y luego, a través del procesamiento de un programa, emite un sonido de bienvenida. También podemos experimentar con el uso de PIR para hacer un timbre.

Conexión

1. Conectar el terminal `PIR` al pin `2` de la placa UNO.
2. Conectar el terminal del zumbador `Buzzer` al pin `3` la placa UNO
3. Conectar `LED7` al pin `13` de la placa UNO.



Construcción de programa

- Programa Arduino

```
void setup()
{
    //led1
    pinMode(13, OUTPUT); //定义13号引脚为输出模式
    //pir1
    pinMode(2, INPUT); //定义2号引脚为输入模式
    //buzzer1
    pinMode(3, OUTPUT); //定义3号引脚为输出模式
}

void loop()
{
    digitalWrite(13, digitalRead(2)); //2号引脚高电平时13号输出高电平
    digitalWrite(3, digitalRead(2));  //2号引脚高电平时3号输出高电平
}
```

Resultados de la operación

Cuando el PIR detecta la fuente de calor infrarrojo del cuerpo humano, activará un nivel alto a través del circuito interno, encendiendo así el LED y haciendo sonar el zumbador. El módulo PIR de la placa de aprendizaje tiene un circuito de retardo incorporado, por lo que habrá un efecto de retardo incluso si no se agrega ningún código `delay()` aquí.

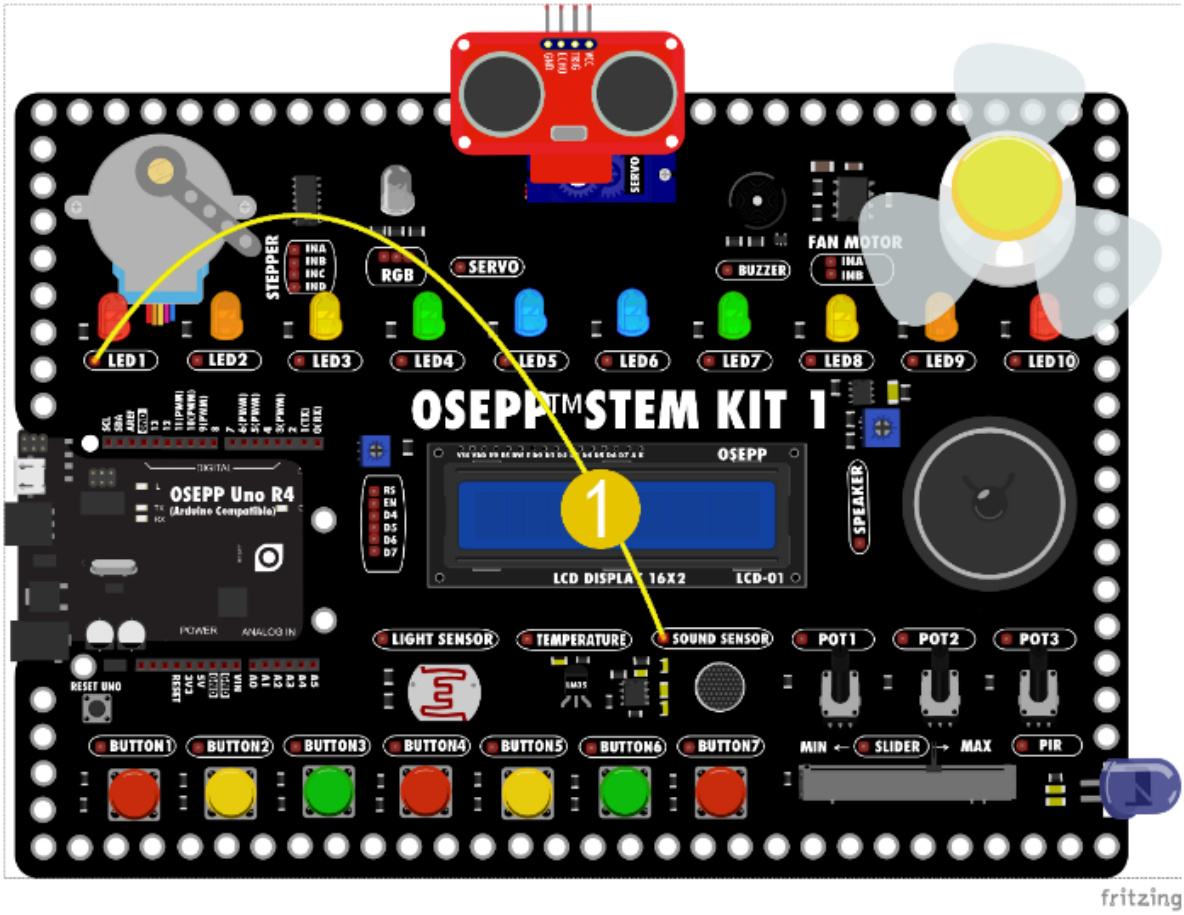
Sensor de sonido. Micrófono

Un micrófono también es un colector de sonido, es decir, un componente que convierte los sonidos recogidos del entorno circundante en una señal eléctrica.
El micrófono convierte las señales de sonido recogidas del mundo exterior en señales eléctricas.

1. Micrófono conectado a un LED

Conexión

Conecte el terminal del micrófono `Sound Sensor` al terminal `LED1`.



Resultado

En este momento, hable al micrófono y verás que el LED parpadea.

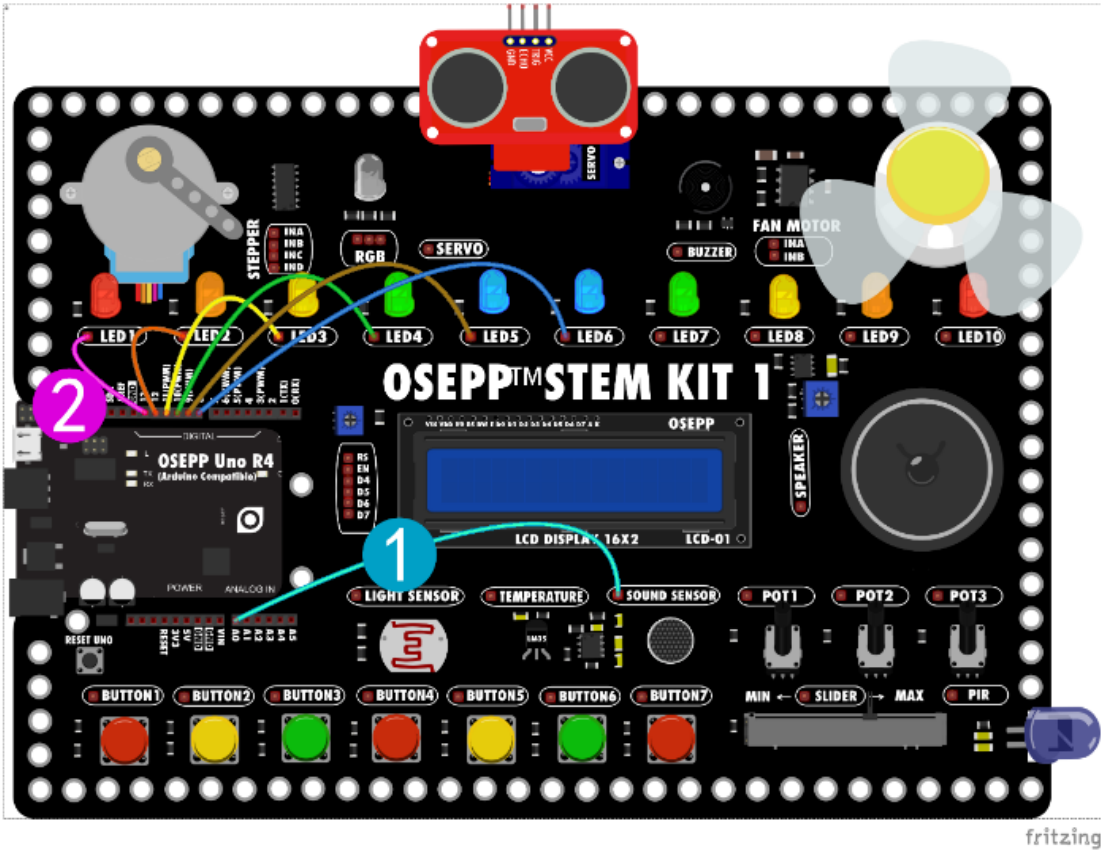
Análisis

Un micrófono es un dispositivo de conversión electroacústica que convierte los sonidos recogidos del mundo exterior en señales eléctricas. El micrófono recibe sonidos externos, y las señales eléctricas generadas son convertidas por el pin `A0` de Arduino y son los valores cambiantes `0-1023` en el programa.

2. Medidor de decibelios con micrófono

Conexión

1. Conecte el micrófono `Sound Sensor` al pin `A0` de la placa UNO.
2. Conecte los pines `LED1-LED6` a los pines 13-8 de la placa UNO.



Construcción de programa

- Programa Arduino

```
void setup()
{
    //soundSensor1
    pinMode(A0, INPUT); //定义A0为输入模式
    //led1
    pinMode(13, OUTPUT); //定义13号引脚为输出模式
    //led2
    pinMode(12, OUTPUT); //定义12号引脚为输出模式
    //led3
    pinMode(11, OUTPUT); //定义11号引脚为输出模式
    //led4
    pinMode(10, OUTPUT); //定义10号引脚为输出模式
    //led5
    pinMode(9, OUTPUT); //定义9号引脚为输出模式
    //led6
    pinMode(8, OUTPUT); //定义8号引脚为输出模式
}

```

```
void loop()
{
    digitalWrite(13, analogRead(A0) > 100); //A0大于100时13号引脚输出高电平
    digitalWrite(12, analogRead(A0) > 200); //A0大于200时12号引脚输出高电平
    digitalWrite(11, analogRead(A0) > 300); //A0大于300时11号引脚输出高电平
    digitalWrite(10, analogRead(A0) > 400); //A0大于400时10号引脚输出高电平
    digitalWrite(9, analogRead(A0) > 500);  //A0大于500时9号引脚输出高电平
    digitalWrite(8, analogRead(A0) > 600);  //A0大于600时8号引脚输出高电平
}
```

Resultados de la operación

Cuando hables a través del micrófono, verás que el LED se ilumina y, cuanto más fuerte sea la voz, más brillante será el LED. El código anterior utiliza una operación de comparación y se emitirá un nivel alto si es mayor que un valor determinado.

3. Lámpara con sensor de luz y sensor de sonido

Cuando caminábamos por el pasillo y aplaudíamos, las luces del pasillo se encendían automáticamente. Las luces del pasillo controladas por voz se controlaban mediante un micrófono. También podemos combinar el sensor de luz anterior para hacer una lámpara. El LED solo se encenderá cuando no haya suficiente luz ni sonido y se apagará después de un retraso de 5 segundos.

Conexión

1. Conectar el LED6 al pin 13 de la placa UNO.
2. Conectar el micrófono Sound Sensor al pin A0 de la placa UNO.
3. Conectar el sensor de luz Light Sensor al pin A1 de la placa UNO.

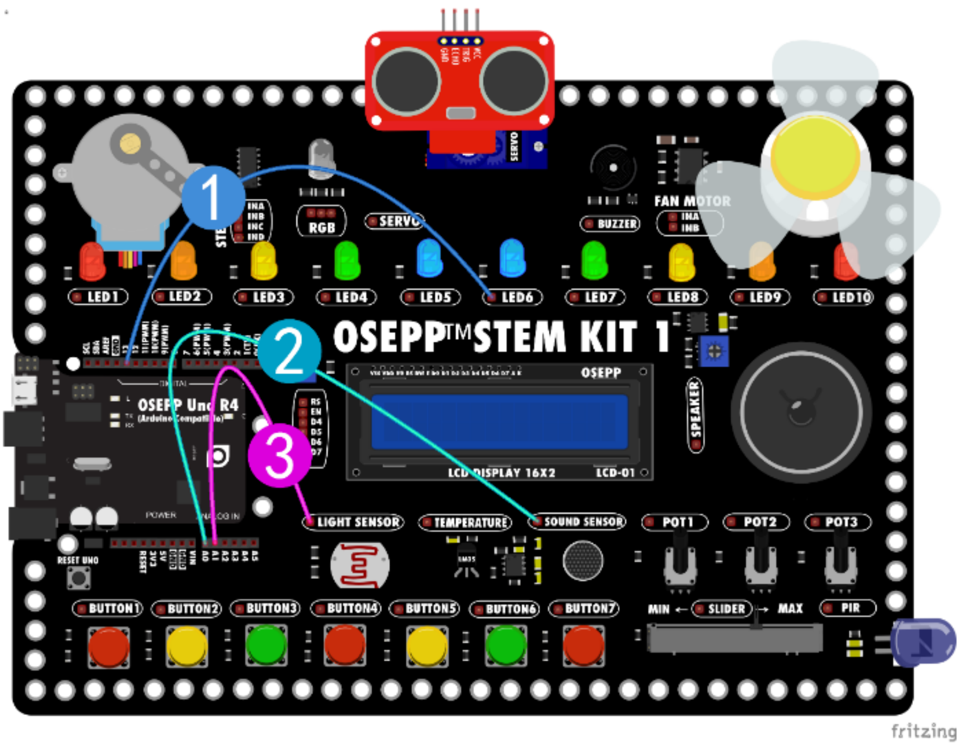
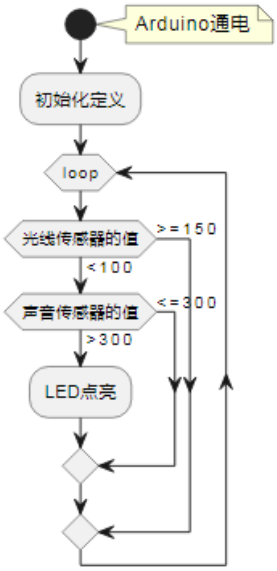


Diagrama de flujo



El valor del programa se puede configurar de acuerdo con la situación real. Cuanto menor sea el valor del sensor de luz, más oscuro será el entorno necesario. Cuanto mayor sea el valor del sensor de sonido, más fuerte será el sonido necesario para activarlo. También puede realizar las modificaciones adecuadas para ver el efecto real.

Construcción de programas

- Programa Arduino

```
void setup()
{
    //soundSensor1
    pinMode(A0, INPUT); //定义A0为输入模式
    //led1
    pinMode(13, OUTPUT); //定义13号引脚为输出模式
    //light1
    pinMode(A1, INPUT); //定义A0为输入模式
}

void loop()
{
    if (analogRead(A1) < 150 && analogRead(A0) > 300) //如果A1小于150与A0大于300，表示此
    {
        digitalWrite(13, HIGH); //13号引脚输出高电平
        delay(5000);            //延时5000毫秒
    }
    else
    {
        digitalWrite(13, LOW); //13号引脚输出低电平
        delay(100);            //延时100毫秒
    }
}
```

Resultados de la operación

Cuando la luz se atenúa, si el valor del sensor de luz es menor que `150` y el valor del sensor de sonido es mayor que `300`, se ejecutará el siguiente programa. El LED se enciende y se apaga después de un retraso de 5 segundos, de lo contrario, el LED permanece en un estado de nivel bajo.

Análisis

- Operadores lógicos

Se utiliza un nuevo operador en el código `(&&)`, y otros operadores comúnmente utilizados incluyendo `(||)` y `(!)`.

El código utiliza && para representar una conexión en serie. Si tanto la primera como la segunda condición son verdaderas, devuelve un valor verdadero (`true`). Es decir, se ejecutará sólo cuando se cumplan todas las condiciones al mismo tiempo.

Código:
`if(a&&b)` solo puede ser verdadero si `a` y `b` son ambos verdaderos, de lo contrario, falso.

El código utiliza || para representar una relación paralela. Si solo una de la primera y de la segunda condición es verdadera, devuelve un valor verdadero (`true`). Es decir, si se cumple una de las condiciones dadas, se ejecutará.

Código:
`if(a||b)` solo si `a` y `b` son verdaderos al tiempo, el resultado es verdadero, el resto son falsos.

El código utiliza ! para indicar que si la condición es "falsa", se devuelve un valor "verdadero" (`true`). Es decir, si no se cumplen las condiciones dadas, se ejecutará.

Código:
`if(!a)` si `a` es verdadero, entonces `!a` es falso; si `a` es falso, entonces `!a` es verdadero.

Sensor de temperatura

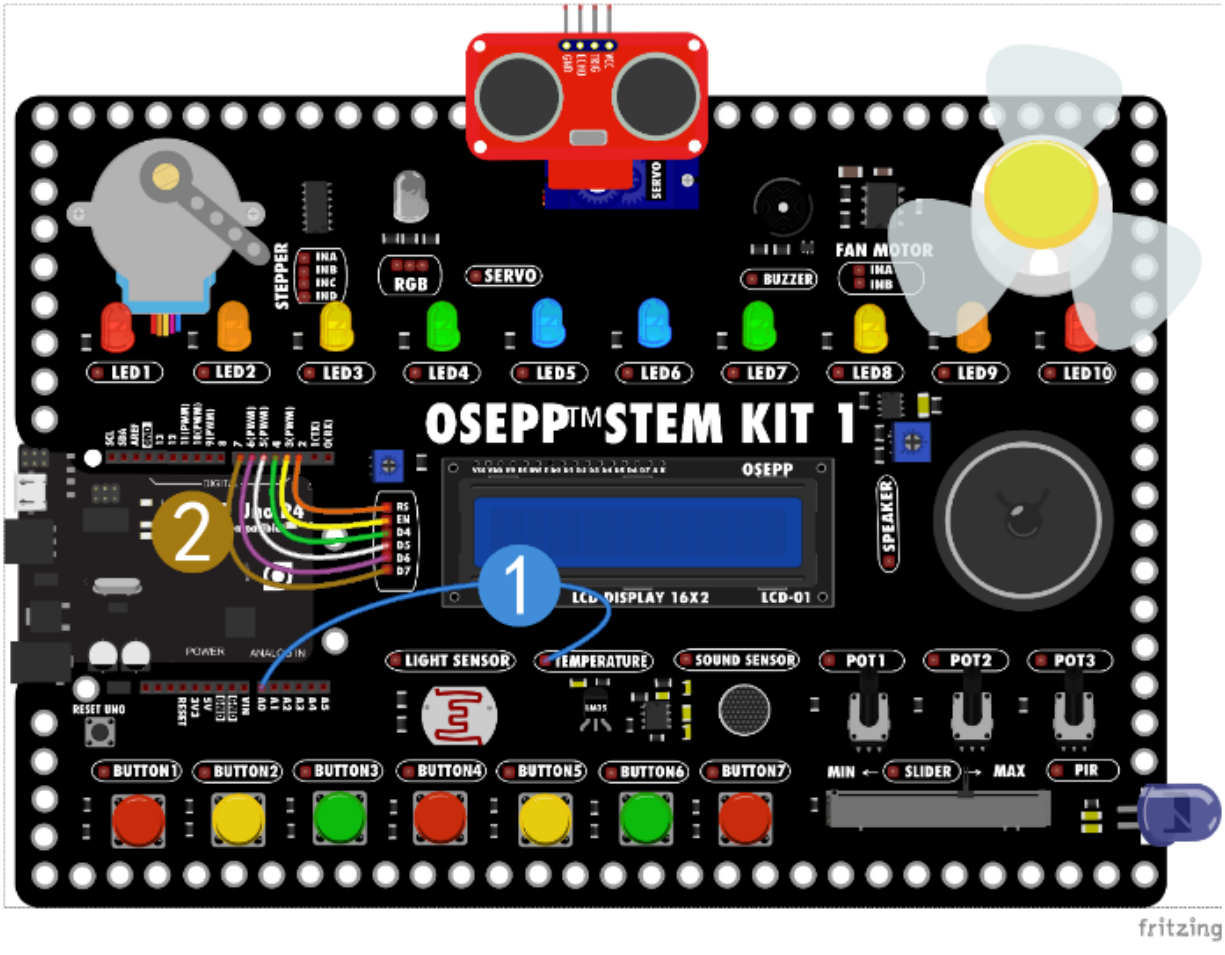
Los sensores de temperatura generalmente se refieren a componentes que convierten la temperatura en datos electrónicos. Las propiedades de muchos materiales y componentes cambian con la temperatura, por lo que hay muchos materiales que pueden usarse como sensores de temperatura. Los parámetros físicos de los sensores de que cambian con la temperatura incluyen: expansión, resistencia, capacitancia, fuerza electromotriz, propiedades magnéticas, frecuencia, propiedades ópticas, ruido térmico, etc.

1. Sensor de temperatura y LCD

La placa de aprendizaje utiliza el sensor de temperatura LM35. Si se aplica un voltaje al sensor de temperatura, cuando la temperatura externa cambia, el LM35 emitirá un voltaje cambiante. La fuente de alimentación se ha conectado dentro de la placa de aprendizaje, y solo necesitamos conectar los pines del sensor a los pines de la placa UNO

Conexión

1. El terminal del sensor de temperatura `Temperature` va conectado al pin `A0` de la placa UNO.
2. Las interfaces del LCD `RS-D7` van conectadas a los pines `2~7` de la placa UNO respectivamente.



Construcción de programa

- Programa Arduino

```
#include <LiquidCrystal.h>

LiquidCrystal lcd1(2, 3, 4, 5, 6, 7); //定义LCD引脚

void setup()
{
    //lm35_1
    pinMode(A0, INPUT); //定义A0为输入模式
    lcd1.begin(16, 2);  //LCD初始化
}

void loop()
{
    lcd1.clear();                           //清屏
    lcd1.setCursor(0, 0);                   //显示光标定位
    lcd1.print(analogRead(A0) * 0.48828125); //计算温度并显示
    delay(1000);                            //延时1000毫秒
}
```

Resultados de la operación

En este momento, podrá ver la temperatura ambiente actual en la pantalla LCD en grados Celsius. Al presionar el sensor de temperatura con la mano, el valor de temperatura del sensor de temperatura que se muestra en la pantalla LCD cambiará.

Motor eléctrico

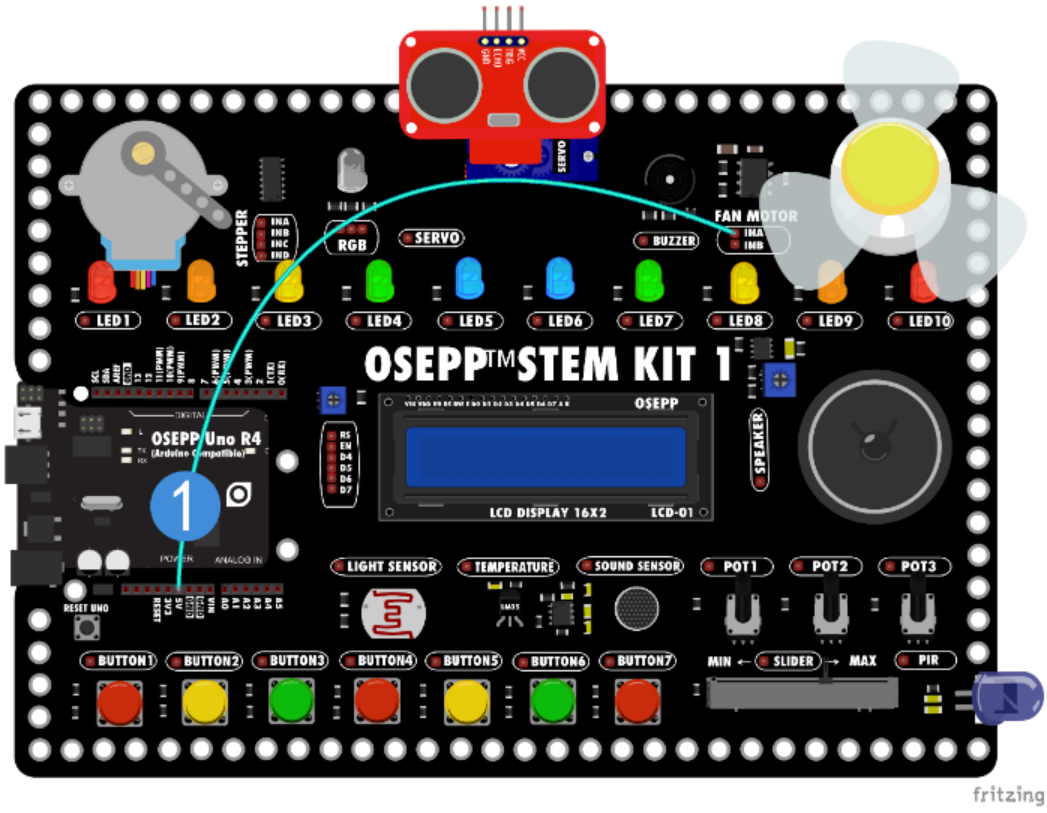
Un motor eléctrico, también conocido como motor o motor eléctrico, es un dispositivo eléctrico que convierte energía eléctrica en energía mecánica y luego puede utilizar la energía mecánica para generar energía cinética para impulsar otros dispositivos. El principio de rotación del motor se basa en la regla de la mano izquierda de John Ambrose Fleming. Cuando se coloca un cable en un campo magnético y fluye corriente a través de él, el cable cortará las líneas del campo magnético y hará que el cable se mueva. La corriente eléctrica entra en la bobina para generar un campo magnético. El dispositivo utiliza el efecto magnético de la corriente para hacer que el electroimán gire continuamente dentro de un imán fijo, que puede convertir la energía eléctrica en energía mecánica. El principio de un motor de corriente continua que genera energía al interactuar con un imán permanente o un campo magnético generado por otro conjunto de bobinas es que el estator permanece estacionario y el rotor se mueve en la dirección de la fuerza generada por la interacción.

1. Motor con hélice pequeño

Un motor es un componente que convierte la energía eléctrica en energía cinética. Siempre que le suministremos electricidad, el motor comenzará a girar.

Conexión

Conecte el pin de voltaje `5V` de la placa UNO al terminal del motor eléctrico (motor ventilador) `INA`.



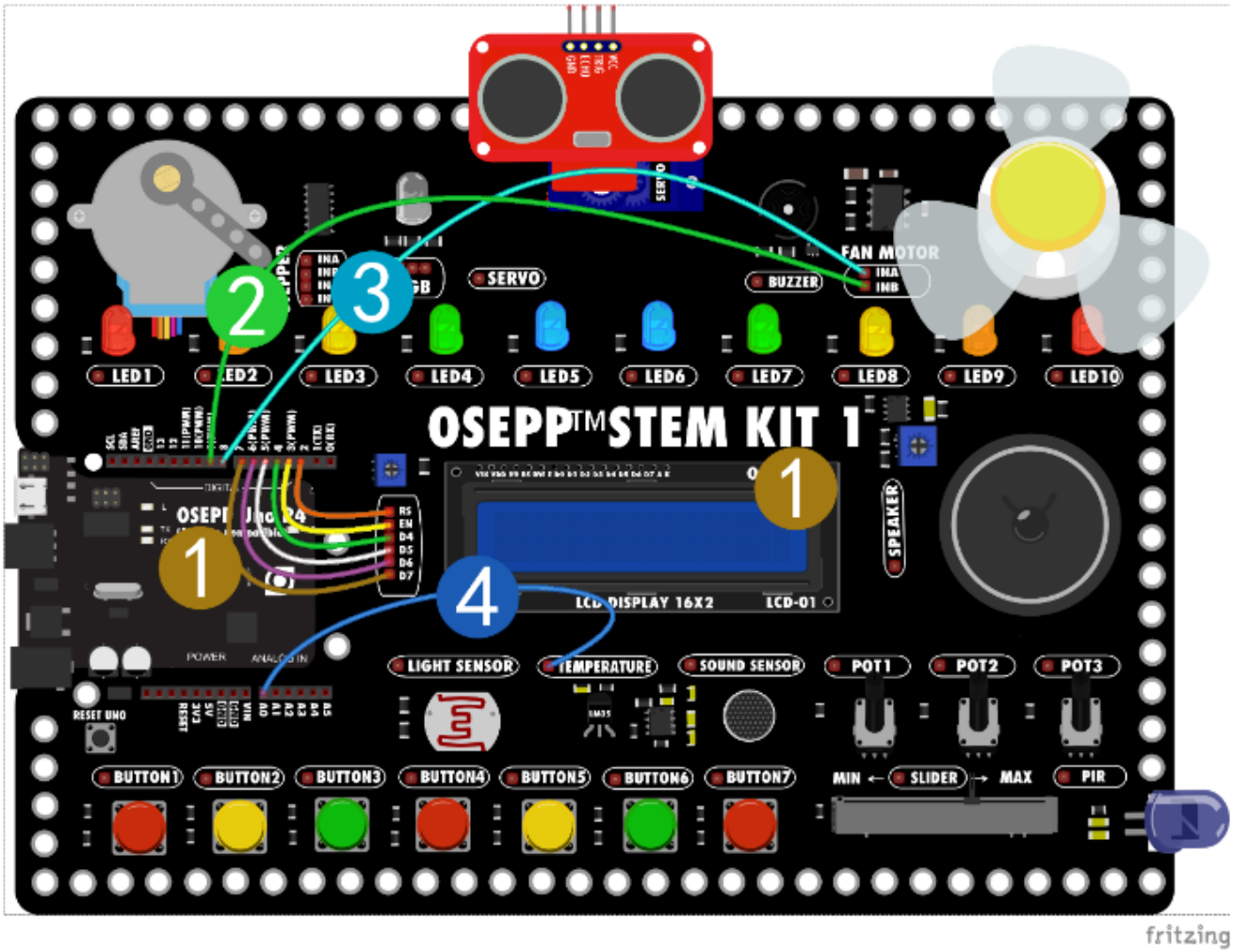
Ahora puedes ver que el motor está girando. Si se conecta a `INB`, el motor girará en la dirección opuesta a la anterior.

Ventilador con sensor de temperatura

Combina el sensor de temperatura que aprendiste antes para hacer un pequeño ventilador con sensor de temperatura. Cuando la temperatura alcance el umbral establecido, el pequeño ventilador comenzará a girar. Cuando la temperatura sea inferior a un valor determinado, el ventilador se detendrá.

Conexión

1. Las interfaces del LCD `RS-D7` van conectadas a los pines `2~7` de la placa UNO respectivamente.
2. Conecte el terminal del motor eléctrico (Fan motor) `INA` al pin `8` de la placa UNO.
3. Conecte el terminal del motor eléctrico (Fan motor) `INB` al pin `9` de la placa UNO.
4. El terminal del sensor de temperatura va conectado al pin `A0` de la placa UNO.



Construcción de programa

- Programa Arduino

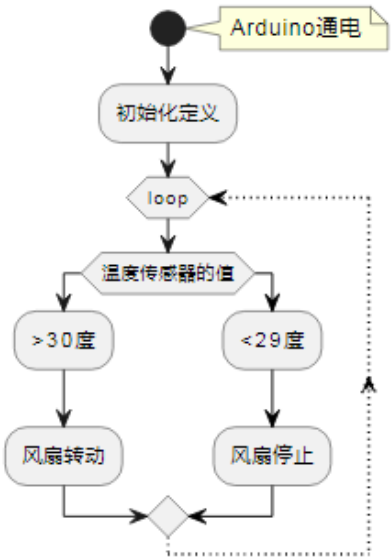
```
#include <LiquidCrystal.h>

LiquidCrystal lcd1(2, 3, 4, 5, 6, 7); //定义LCD引脚

void setup()
{
    lcd1.begin(16, 2);
    //lm35_1
    pinMode(A0, INPUT); //定义A0为输入模式
    //fan1
    pinMode(8, OUTPUT);  //定义 3引脚为输出模式
    pinMode(9, OUTPUT);  //定义9 3引脚为输出模式
}

void loop()
{
    if (analogRead(A0) * 0.48828125 > 30) //如果温度值大于30度
    {
        analogWrite(8, 255); //3引脚输出pwm为255
        analogWrite(9,  0);  //5引脚输出pwm为0
    }
    else if (analogRead(A0) * 0.48828125 < 29) //如果温度值小于29度
    {
        analogWrite(8, 0);  //2引脚输出pwm为0
        analogWrite(9, 0);  //5引脚输出pwm为0
    }
    lcd1.clear();                           //清屏
    lcd1.setCursor(0, 0);                   //显示光标定位
    lcd1.print(analogRead(A0) * 0.48828125); //显示温度
    delay(3000);                            //延时1000毫秒
}
```

Resultados de la operación

Si la temperatura ambiente experimental actual es superior a 30°C, el pequeño ventilador comenzará a girar. Si es menor a 29°C dejará de girar. Puede modificar estos dos valores según la temperatura ambiente actual que se muestra en la pantalla, para que el ventilador pueda arrancar y detenerse cuando el valor cambie.

Análisis

Diagrama de flujo



Si la temperatura detectada por el sensor de temperatura es mayor a 30 grados, el ventilador comenzará a girar. El código juzga que `>30` es mayor que 30 grados. Cuando la temperatura es exactamente de 30 grados, el ventilador no gira. Tiene que estar a 31 grados para que arranque.

Si se desea rotar cuando se alcanza 30°C se debe utilizar el código `if(analogRead(A0) * 0.48828125 >= 30)`. Simplemente cambie el signo mayor que por un signo mayor o igual a. Lo mismo se aplica a los valores menores de 29°C.

2. Ventilador con regulación de velocidad PWM

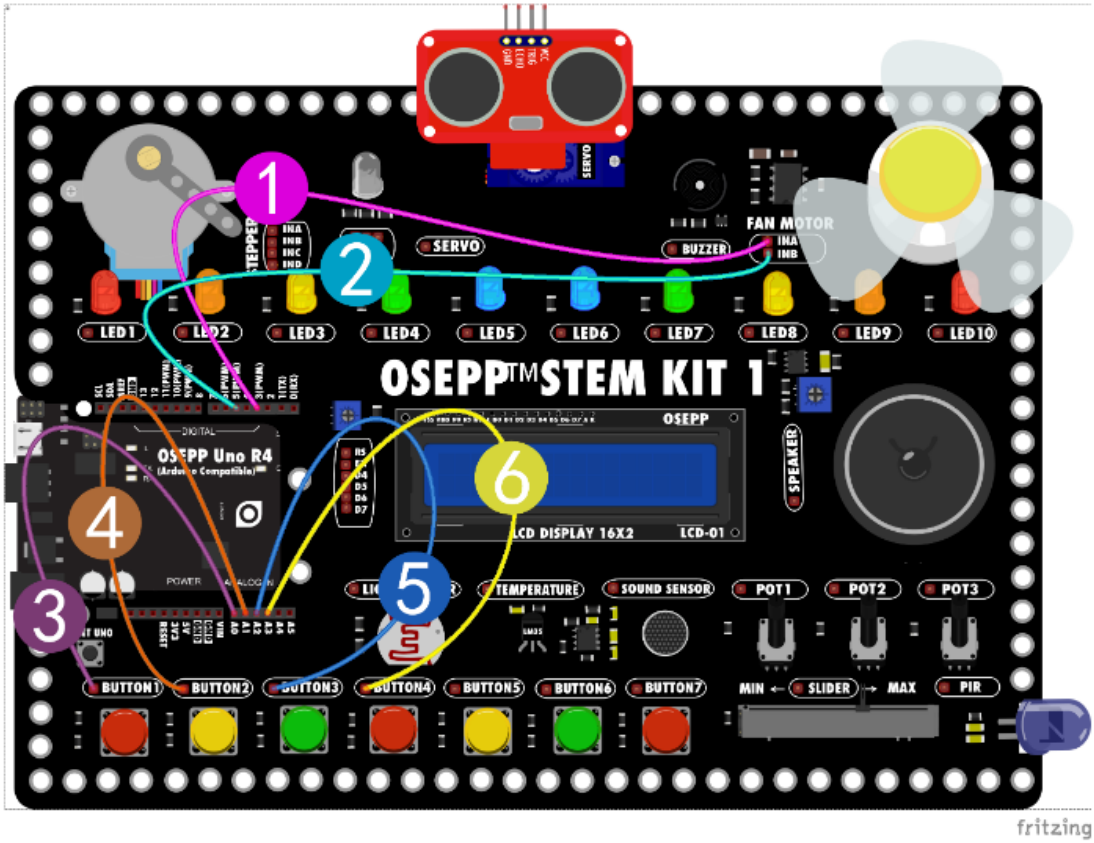
Cuando suministramos energía al motor, el motor girará, y cuando desconectamos la energía, se detendrá. Si seguimos acortando el intervalo de tiempo entre el encendido y el apagado, ¿el motor girará más rápido cuando el intervalo sea más corto y más lento cuando el intervalo sea más largo? Este principio se utiliza para realizar la regulación `PWM` de la velocidad del motor.

La modulación por ancho de pulso (Pulse Width Modulation, abreviado como `PWM`), es una tecnología que convierte señales analógicas en pulsos. Generalmente, el período del pulso después de la conversión es fijo, pero el período de trabajo del pulso cambiará según el tamaño de la señal analógica.

Conexión

Conecte los cables de acuerdo con la siguiente tabla.

Número de serie	Terminales componentes	Número pin placa UNO
1	Motor Fan INA	3
2	Motor Fan INB	5
3	Botón 1	A0
4	Botón 2	A1
5	Botón 3	A2
6	Botón 4	A3



Construcción de programa

- Programa Arduino

```
void setup()
{
    //button1
    pinMode(A0, INPUT); //定义开关1
    //button2
    pinMode(A1, INPUT); //定义开关2
    //button3
    pinMode(A2, INPUT); //定义开关3
    //button4
    pinMode(A3, INPUT); //定义开关4
    //fan1
    pinMode(3, OUTPUT); //定义3号引脚为输出模式
    pinMode(5, OUTPUT); //定义5号引脚为输出模式
}

void loop()
{
    if (digitalRead(A0) == LOW) //如果开关1按下
    {
        analogWrite(3, 50); //3号引脚输出pwm为50
        analogWrite(5, 0);  //5号引脚输出pwm为0
    }
```

```
    else if (digitalRead(A1) == LOW) //如果开关2按下
    {
        analogWrite(3, 150); //3号引脚输出pwm为150
        analogWrite(5, 0);   //5号引脚输出pwm为0
    }
    else if (digitalRead(A2) == LOW) //如果开关3按下
    {
        analogWrite(3, 0);   //3号引脚输出pwm为0
        analogWrite(5, 150); //5号引脚输出pwm为150
    }
    else if (digitalRead(A3) == LOW) //如果开关4按下
    {
        analogWrite(3, 0);  //3号引脚输出pwm为0
        analogWrite(5, 50); //5号引脚输出pwm为50
    }
    else
    {
        analogWrite(3, 0); //马达停止转动
        analogWrite(5, 0);
    }
}
```

Resultados de la operación

Cuando se presiona el interruptor `Button1`, el ventilador gira hacia adelante a baja velocidad y cuando se presiona el interruptor `Button2`, el ventilador gira hacia adelante a alta velocidad. Cuando se presiona el interruptor `Button3`, el ventilador gira en sentido inverso a baja velocidad y cuando se presiona el interruptor `Button4`, el ventilador gira en sentido inverso a alta velocidad. La velocidad se controla mediante el modo de salida `PWM`, pero la dirección se logra utilizando una señal de salida en un pin diferente.

Análisis

En los circuitos analógicos, el valor de las señales analógicas puede cambiar de forma continua sin prácticamente restricciones en cuanto a tiempo y amplitud. Básicamente, pueden adoptar cualquier valor real, y la entrada y la salida también cambian de forma lineal. Por lo tanto, en los circuitos analógicos, el voltaje y la corriente se pueden utilizar directamente para controlar objetos, como el control del interruptor de volumen en electrodomésticos, el control de brillo de lámparas LED halógenas, etc. Sin embargo, los circuitos analógicos tienen muchos problemas: por ejemplo, las señales de control tienden a desviarse con el tiempo y son difíciles de ajustar, consumen mucha energía, son susceptibles al ruido y a la interferencia ambiental, etc.

A diferencia de los circuitos analógicos, los circuitos digitales toman valores dentro de un rango predeterminado. En un momento dado, su salida solo puede estar en los estados ON y OFF, por lo que la tensión o la corriente se cargan en la carga analógica en una secuencia de pulsos repetitivos de manera on/off.

La tecnología PWM es un método de codificación digital de niveles de señales analógicas. Codifica el nivel de una señal analógica modulando el ciclo de trabajo de una

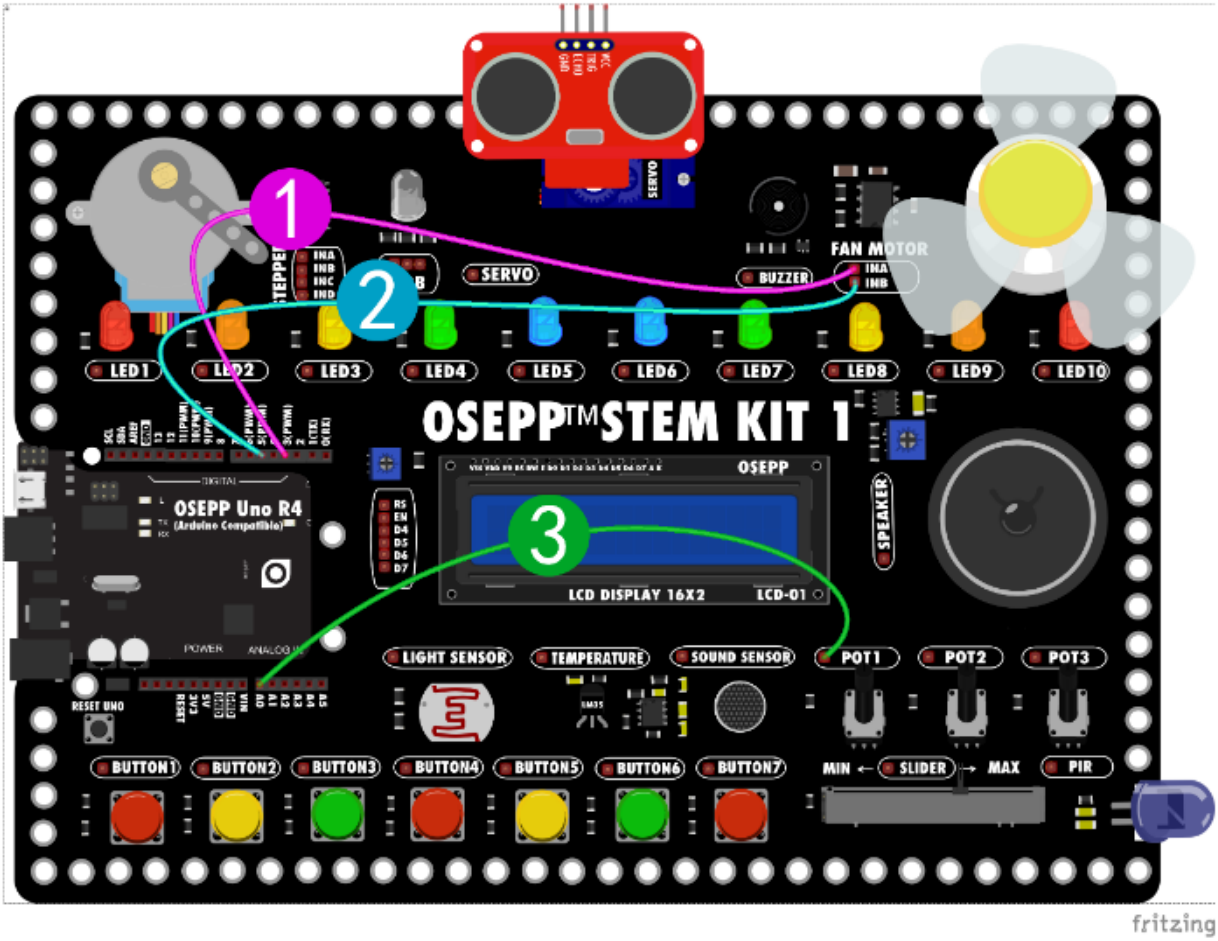
onda cuadrada mediante un contador de alta resolución (frecuencia de modulación). Su mayor ventaja es que todas las señales del procesador al objeto controlado están en formato digital y no hay necesidad de conversión de digital a analógico; y la capacidad de resistir el ruido también se mejora enormemente (el ruido solo puede tener un impacto sustancial en las señales digitales cuando es lo suficientemente fuerte como para cambiar el valor lógico). Esta es también la razón principal por la que PWM se usa ampliamente en industrias de transmisión de señales como las comunicaciones.
Actualmente, muchos microcontroladores (MCU) incluyen módulos controladores PWM.

3. Ventilador con control de velocidad continuo

Utilice el potenciómetro para ajustar la velocidad del motor. La velocidad infinitamente variable en realidad divide la velocidad del ventilador en 255 niveles, similares a 255 engranajes, de modo que no podemos sentir la existencia de los engranajes.

Conexión

1. El terminal del motor eléctrico (Fan Motor) INA se conecta al pin 3 de la placa UNO.
2. El terminal del motor eléctrico (Fan Motor) INB se conecta al pin 5 de la placa UNO.
3. Potenciómetro POT1 va conectado al pin A0 de la placa UNO.



Construcción de programa

- Programa Arduino

```
void setup()
{
    //potentiometer1
    pinMode(A0, INPUT); //定义A0为输入模式
    //fan1
    pinMode(3, OUTPUT); //定义3号引脚为输出模式
    pinMode(5, OUTPUT); //定义5号引脚为输出模式
}

void loop()
{
    analogWrite(3, map(analogRead(A0), 0, 1023, 0, 255)); //3号引脚输出pwm为A0读取到值
    analogWrite(5, 0);                                    //5号引脚输出pwm为0
    delay(1);                                             //延时1毫秒
}
```

Resultados de la operación

Al girar el potenciómetro también cambiará la velocidad del ventilador. A medida que el potenciómetro gira en el sentido de las agujas del reloj desde el extremo izquierdo, la velocidad del ventilador aumentará lentamente. La velocidad del ventilador aumenta y disminuye con una transición muy suave.

Análisis

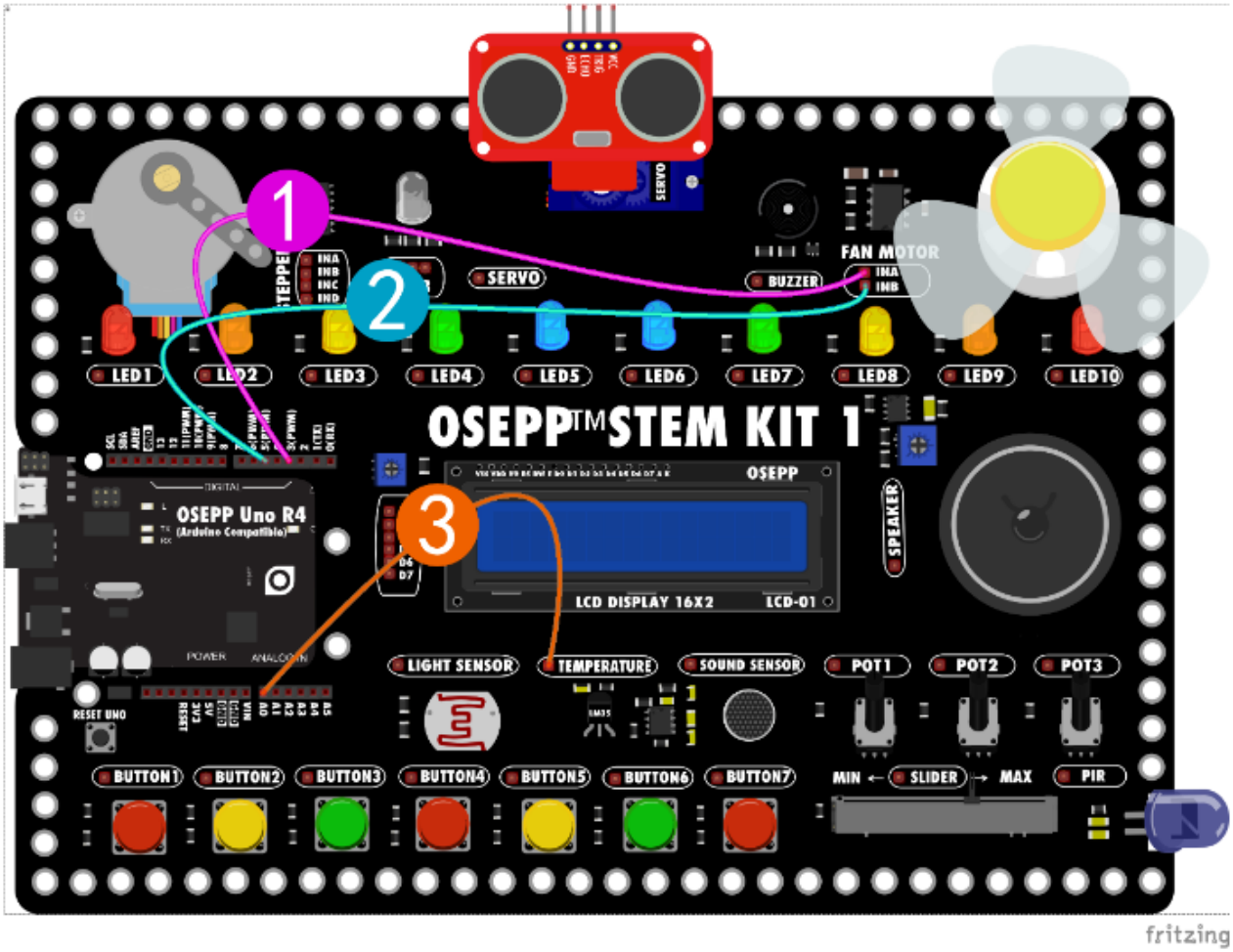
En el programa, Arduino primero leerá el valor del potenciómetro, el valor mínimo es `0` y el valor máximo es `1023`. Luego, este valor se asigna a `0-255` la señal `PWM` de salida para activar el ventilador.

4. Ventilador con control por temperatura

Combinado con un sensor de temperatura, la temperatura se puede utilizar para controlar el interruptor del motor del ventilador. Cuando la temperatura es baja, la velocidad del ventilador es baja, y cuando la temperatura es alta, la velocidad del ventilador es rápida. Cuando la temperatura desciende por debajo de cierto nivel, el ventilador se detiene. Esto crea un ventilador con regulación de velocidad y control de temperatura que se enciende y se apaga automáticamente.

Conexión

1. El terminal del motor eléctrico (Fan Motor) `INA` se conecta al pin `3` de la placa UNO.
2. El terminal del motor eléctrico (Fan Motor) `INB` se conecta al pin `5` de la placa UNO.
3. El sensor de temperatura `Temperature se` conecta al pin `A0` de la placa UNO.



Construcción de programas

- Programa Arduino

```
void setup()
{
    //lm35_1
    pinMode(A0, INPUT); //定义A0为输入模式
    //fan1
    pinMode(3, OUTPUT); //定义3号引脚为输出模式
    pinMode(5, OUTPUT); //定义5号引脚为输出模式
    Serial.begin(115200); //设置串口波特率
}

void loop()
{
    if (analogRead(A0) * 0.48828125 < 26) //如果温度小于26度
    {
        analogWrite(3, 0); //风扇等不转动
        analogWrite(5, 0);
    }
    else
    {
        analogWrite(3, map(analogRead(A0) * 0.48828125, 26, 33, 50, 255));
        //温度在26与33之间映射 风扇转速在20-100%
        analogWrite(5, 0);
    }
    Serial.println((analogRead(A0) * 0.48828125)); //串口显示温度
    delay(500);                                    //延时500毫秒
}
```

Resultados de la operación

Se establece un valor de temperatura 26 en el experimento. Cuando la temperatura del sensor de temperatura supera este valor, el ventilador comienza a girar. Cuando la temperatura es 26°C, la velocidad es aproximadamente 20%, y cuando la temperatura alcanza 33°C, la velocidad alcanza 100%. El valor que se establece aquí de 26 debe ajustarse de acuerdo con el cambio de la temperatura ambiente en ese momento. Se ha agregado la impresión del puerto serial al programa y la visualización de la temperatura se puede ver en el puerto serial. Al configurar el valor de la temperatura, consulte el cambio del valor que se muestra en el puerto serial.

Motor paso a paso

Un motor paso a paso, también conocido como motor de pulso, es un motor de inducción que implica muchos conocimientos en mecánica, motores, electrónica y computadoras. Como actuador, el motor paso a paso es uno de los productos clave de la mecatrónica y se utiliza ampliamente en diversos sistemas de control de automatización.

¿Qué es un motor paso a paso?

Un motor paso a paso es un actuador que convierte pulsos eléctricos en desplazamiento angular. Cuando el controlador paso a paso recibe una señal de pulso, impulsa el motor paso a paso para que gire un ángulo fijo (y ángulo de paso) en la dirección establecida. Puede controlar el desplazamiento angular controlando el número de pulsos para lograr un posicionamiento preciso. Al mismo tiempo, puede controlar la velocidad y la aceleración del motor controlando la frecuencia del pulso, logrando así el propósito de la regulación de la velocidad.

Características principales

El motor paso a paso puede lograr de forma sencilla un posicionamiento de alta precisión y detener la pieza de trabajo en la posición de destino con alta precisión simplemente operando la señal de pulso. El motor paso a paso se coloca en unidades de ángulos de paso básicos.

1. Control de motor paso a paso con Arduino

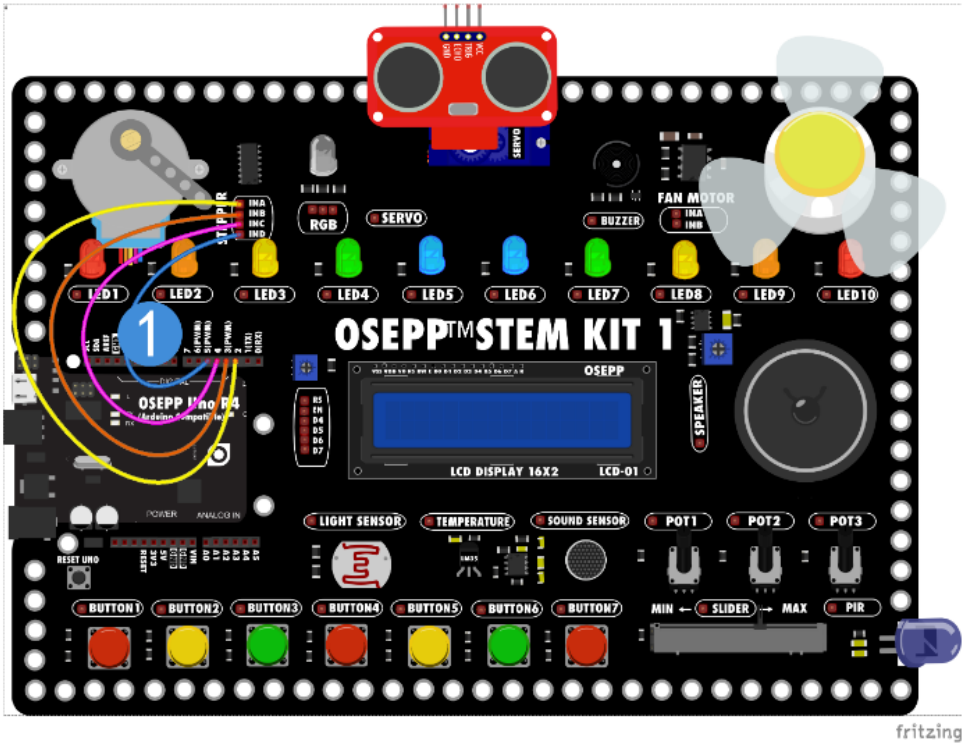
Los parámetros del motor paso a paso 28BYJ-48 en el Kit STEM son: Secuencia de señal de control de 4 pasos: 11,25 grados/paso, 32 pasos para una rotación. En el modo de 4 pasos, una rotación tomará: 32 (pasos/rotación) X 64 (relación de engranaje) = 2048 pasos.

El motor paso a paso utilizado en el experimento realiza `2048` pasos en una vuelta y se requiere `2048` señales de pulso para que el motor paso a paso realice un círculo. Luego, al dividir `2048` por `60` puedes crear un efecto similar al de un segundero.

Conexión

Los pines 2-5 de la placa UNO se conectan a las terminales INA-IND del motor paso a paso.

Pin placa UNO	Pin motor paso a paso
2	INA
3	INB
4	INC
5	IND



Construcción de programa

- Programa Arduino

```
#include <Stepper.h>

Stepper stepper1(2048, 2, 4, 3, 5); //定义步进电机的步数和引脚

void setup()
{
    stepper1.setSpeed(10); //定义每分钟转速
}

void loop()
{
    stepper1.step(-2048 / 60); //每次执行步数
    delay(1000);               //延时1000毫秒
}
```

Resultados de la operación

Después de cargar el código, el motor paso a paso gira como el segundero de un reloj.

Análisis

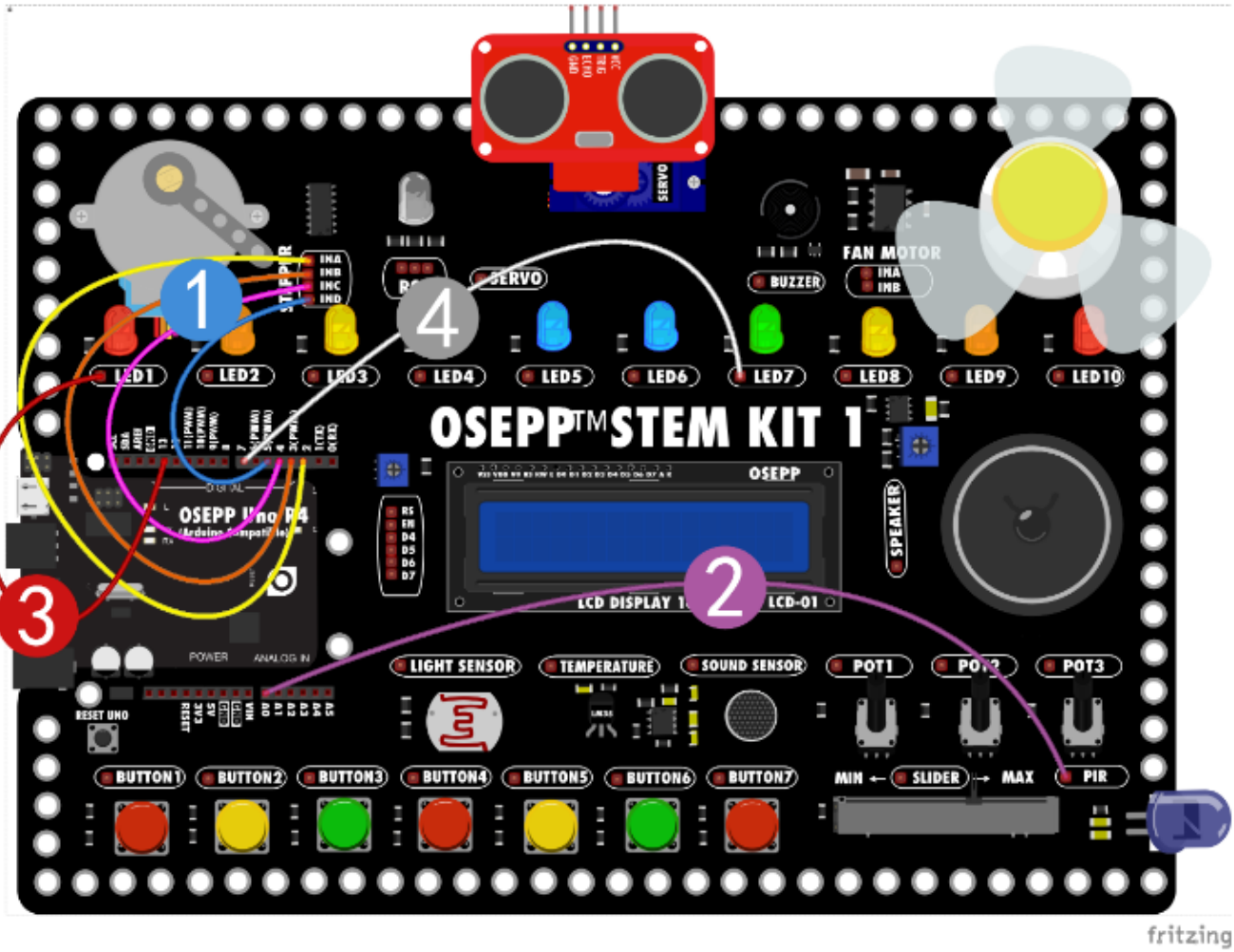
Como no se puede dividir 2048 entre 60 de manera exacta, habrá un error de un segundo cada pocas vueltas. 2048/60=34...82, disminuye su velocidad casi un segundo cada 4 vueltas.

2. Puerta automática con sensor

¿Qué escenarios se utilizan cuando se combinan motores paso a paso y PIR? Una puerta que se abre automáticamente. Si alguien se acerca a la puerta, el PIR lo detecta y abre la puerta. La puerta se cerrará automáticamente cuando alguien pase. La puerta se iluminará en verde cuando esté abierta y en rojo cuando esté cerrada.

Conexión

1. Los pines `2-5` van conectados a los pines `INA-IND` del motor paso a paso.
2. Conecte el pin PIR al pin A0 de la placa UNO.
3. Conecte el LED1 al pin 13 de la placa UNO.
4. Conectar el LED7 al pin 7 de la placa UNO.



Construcción de programas

```
#include <Stepper.h>

Stepper stepper1(2048, 2, 4, 3, 5); //定义步进电机的步数和引脚

void setup()
{
    stepper1.setSpeed(10); //定义每分钟转速
    //pir1
    pinMode(A0, INPUT); //定义PIR引脚
    //led1
    pinMode(13, OUTPUT); //定义LED1引脚
    //led2
    pinMode(7, OUTPUT); //定义LED2引脚
}

void loop()
{
    if (digitalRead(A0)) //读取PIR的值
    {
        digitalWrite(13, LOW); //红色熄灭
        digitalWrite(7, HIGH); //绿色点亮
        delay(1000);           //延时1000毫秒
        stepper1.step(1024);   //步进电机，正向转动1024步
        delay(2000);           //延时2000毫秒
        stepper1.step(-1024);  //步进电机，反向转动1024步
        delay(1000);           //延时1000毫秒
    }
    else
    {
        digitalWrite(13, HIGH); //红色点亮
        digitalWrite(7, LOW);   //绿色熄灭
        delay(1000);            //延时1000毫秒
    }
}
```

Resultados de la operación

Cuando el PIR detecta la fuente de calor infrarrojo del cuerpo humano, se activa un nivel alto. El LED verde se enciende, el motor paso a paso gira medio círculo en dirección hacia adelante, espera 2 segundos y luego gira medio círculo en dirección inversa. Luego el LED rojo se enciende y el LED verde se apaga.

Servo

Un servo es un motor de posición (ángulo), adecuado para sistemas de control que requieren que el ángulo se cambie y mantenga continuamente. Se ha utilizado ampliamente en juguetes de control remoto de alta gama, como aviones, modelos de submarinos y robots de control remoto.

El servo es principalmente adecuado para sistemas de control que requieren que los ángulos se cambien y mantengan constantemente, como los brazos y las piernas de los robots humanoides y el control de dirección de modelos de automóviles y modelos de aviones. La señal de control del servo es en realidad una señal de modulación de ancho de pulso (señal PWM), que puede ser generada por dispositivos FP-GA, circuitos analógicos o microcontroladores.

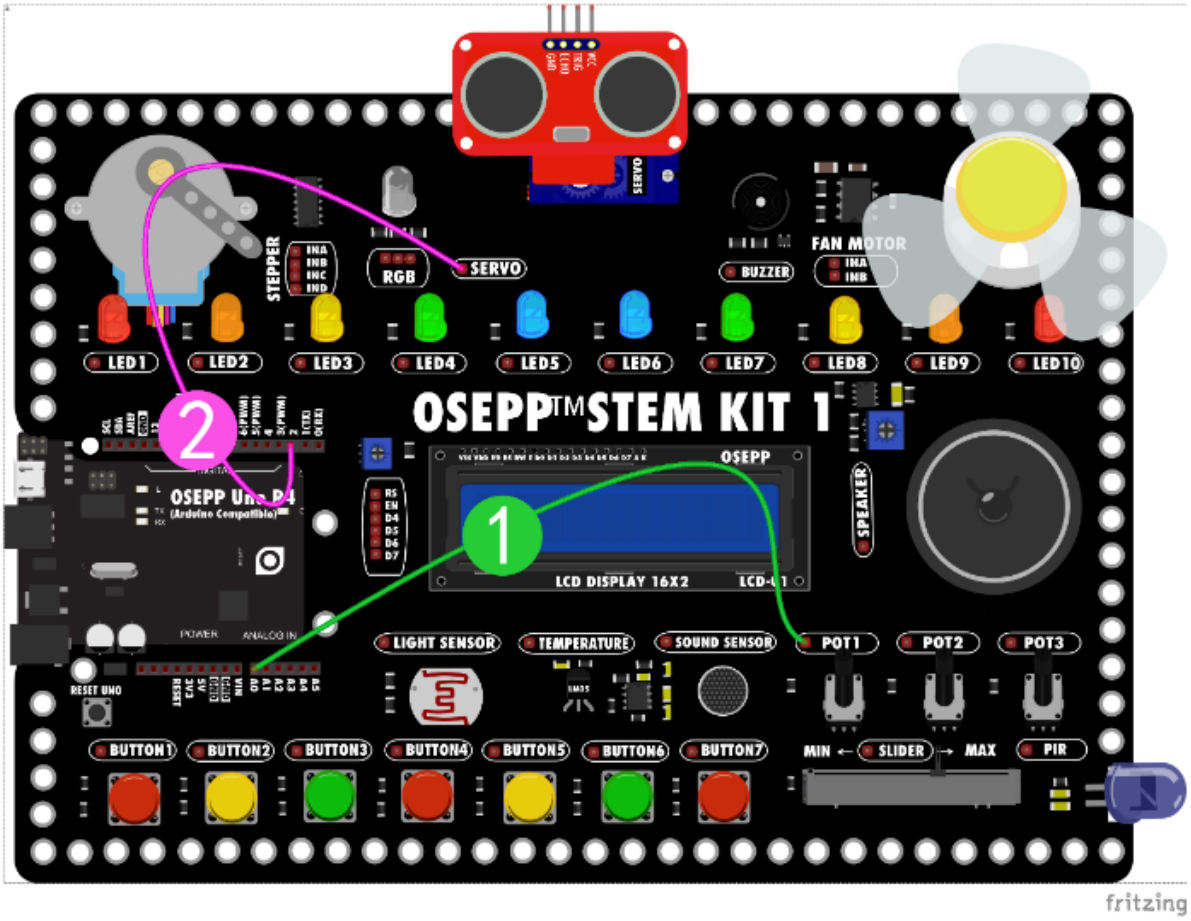
En pocas palabras, un servo es una unidad servo que integra un motor de DC, un controlador de motor, un reductor, etc., y está empaquetado en una carcasa que es fácil de instalar. El servo puede girar cualquier ángulo entre 0 y 180 grados según sus instrucciones y luego detenerse con precisión. Un sistema de motor que puede rotar un ángulo determinado con relativa precisión utilizando una señal de entrada simple. El servo está equipado con un potenciómetro (u otro sensor de ángulo) para detectar el ángulo de rotación del eje de salida. La placa de control puede controlar y mantener el ángulo del eje de salida con mayor precisión en función de la información del potenciómetro.

1. Servo controlado por potenciómetro

Este experimento utiliza un potenciómetro para controlar el ángulo del servo, por lo que debes conectar el servo y el potenciómetro al mismo tiempo.

Conexión

1. El potenciómetro POT1 estará conectado al pin `A0` de la placa UNO.
2. El terminal del servomotor Servo estará conectado al pin 3 de la placa UNO.



Construcción de programa

Código Arduino

```
#include <Servo.h>

Servo servo1;

void setup()
{
    servo1.attach(3); //定义舵机引脚
    //potentiometer1
    pinMode(A0, INPUT); //定义电位器引脚
}

void loop()
{
    servo1.write(map(analogRead(A0), 0, 1023, 0, 180)); //电位器的值映射到舵机输出
}
```

Resultados de la operación

Al girar el potenciómetro, el servo también girará. Cuando el potenciómetro se gira al mínimo a la izquierda, el servo está en la posición `0度`, y cuando el potenciómetro se gira al máximo a la derecha, el servo está en la posición `180度`.

Análisis

El servo solo puede funcionar en el rango de `0 a 180` grados, por lo que el rango de entrada del potenciómetro `0-1023` debe asignarse a `0-180`.

Módulo ultrasónico

El módulo de medición de distancia ultrasónica HC-SR04 puede proporcionar una función de detección de distancia sin contacto de 2cm a 400cm y la precisión de medición puede alcanzar hasta 3mm. El módulo incluye un transmisor, un receptor y un circuito de control ultrasónicos. Se utiliza a menudo en la medición de distancias y en la dirección de coches inteligentes o en algunos proyectos. La medición de distancia del automóvil inteligente puede detectar obstáculos que se encuentren más adelante a tiempo, lo que le permite girar a tiempo para evitarlos.

El principio de medición de distancia ultrasónica es utilizar la velocidad de propagación conocida de las ondas ultrasónicas en el aire para medir el tiempo que tarda la onda de sonido en reflejarse en un obstáculo después de ser transmitida, y calcular la distancia real desde el punto de transmisión hasta el obstáculo basándose en la diferencia de tiempo entre la transmisión y la recepción. En primer lugar, el transmisor ultrasónico emite ondas ultrasónicas en una dirección determinada y comienza a cronometrar al mismo tiempo que la emisión. Las ondas ultrasónicas se propagan en el aire y regresan inmediatamente si encuentran un obstáculo en el camino. El receptor ultrasónico deja de cronometrar inmediatamente cuando recibe la onda reflejada. La velocidad de propagación de las ondas ultrasónicas en el aire es C=340m/s. Según el tiempo T segundos registrado por el cronómetro, se puede calcular la distancia L entre el punto de emisión y el obstáculo, es decir:

$$L= C \times T/2$$

Esto se llama medición de diferencia horaria.

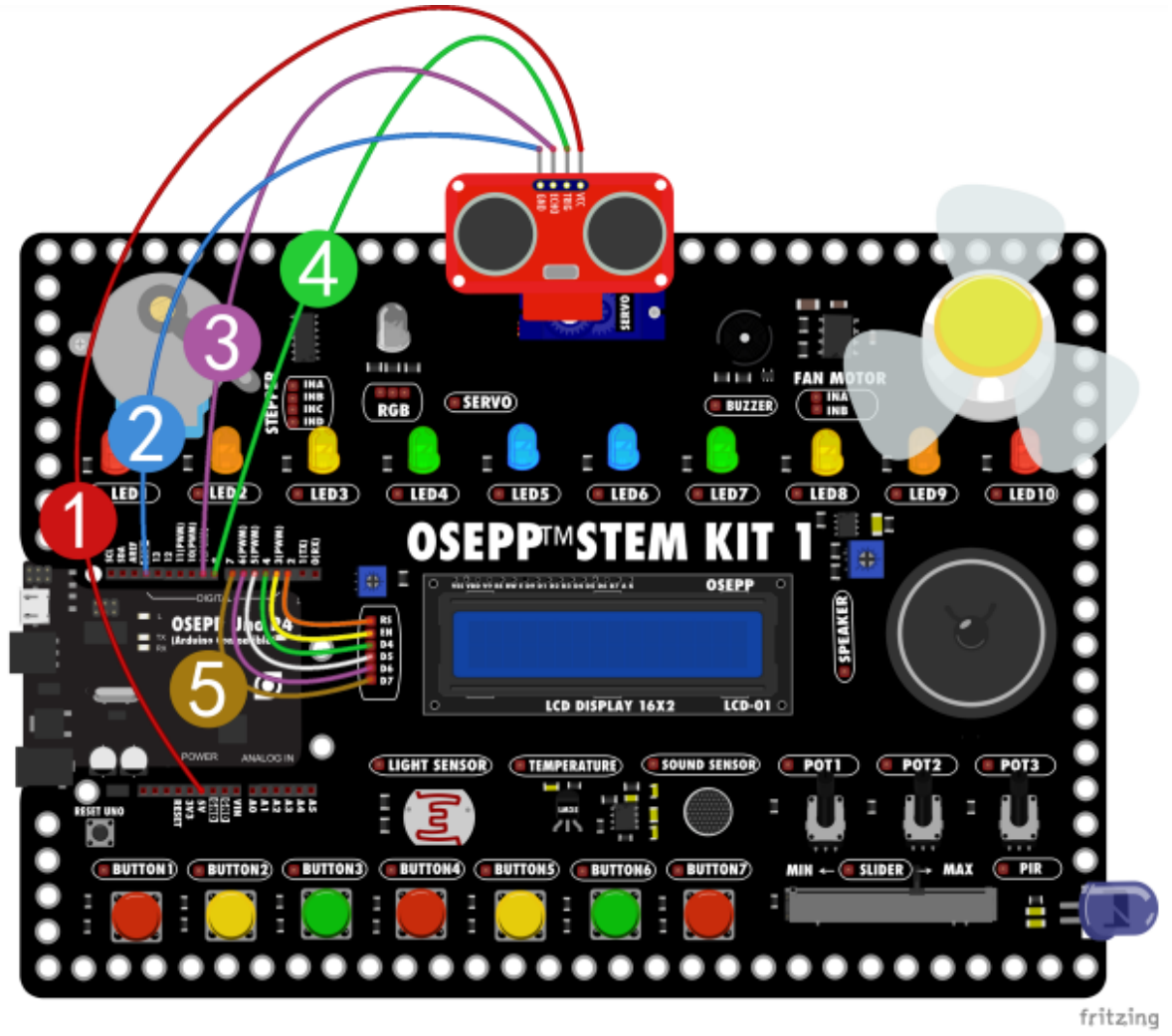
La frecuencia del sonido que la gente puede oír es de 20Hz a 2KHz, que son ondas sonoras audibles. Los sonidos que se encuentran fuera de este rango de frecuencia, por debajo de los 20Hz, se denominan ondas sonoras de baja frecuencia y los sonidos por encima de los 20KHz se denominan ondas ultrasónicas. El rango de frecuencia del habla en general es de 10Hz a 8KHz. Las ondas ultrasónicas tienen buena direccionalidad, un fuerte poder de penetración, son fáciles de obtener con energía sonora relativamente concentrada y pueden propagarse a grandes distancias en el agua. Las ondas ultrasónicas reciben ese nombre porque su límite de frecuencia inferior es aproximadamente igual al límite superior de la audición humana.

1. Medición de distancia por ultrasonido

Utilice la placa Uno para controlar el sensor ultrasónico para medir la distancia del obstáculo que se encuentra frente a usted y mostrarla en la pantalla LCD. Para ello, es necesario conectar tanto la pantalla LCD como el sensor ultrasónico a la placa UNO.

Conexión

1. El pin VCC del módulo ultrasónico va conectado al pin 5V de la placa UNO.
2. El pin GND del módulo ultrasónico va conectado al pin GND de la placa UNO.
3. El pin ECH0 del módulo ultrasónico va conectado al pin 9 de la placa UNO.
4. El pin TRIG del módulo ultrasónico va conectado al pin 8 de la placa UNO.
5. Las interfaces RS-D7 del LCD van conectadas a los pines 2~7 de la placa UNO respectivamente.



Construcción de programa

Código Arduino

```
#include <oseppRobot.h>
#include <LiquidCrystal.h>

OseppUltrasonic ultrasonic1(8, 9); //定义超声波引脚
LiquidCrystal lcd1(2, 3, 4, 5, 6, 7); //定义LCD引脚

void setup()
{
    lcd1.begin(16, 2); //LCD初始化
}

void loop()
{
    lcd1.clear();                   //清屏
    lcd1.setCursor(0, 0);           //显示光标定位
    lcd1.print(ultrasonic1.ping()); //显示距离
    lcd1.print("mm");               //单位
    delay(1000);                    //延时
}
```

Resultados de la operación

Una vez cargado el código correctamente, si hay un objeto frente al módulo ultrasónico y la distancia es entre `2-4000mm`. La pantalla LCD mostrará la distancia entre el objeto y el módulo ultrasónico. Puede cambiar la distancia entre el objeto y el módulo ultrasónico para ver si la pantalla LCD cambia.

Análisis

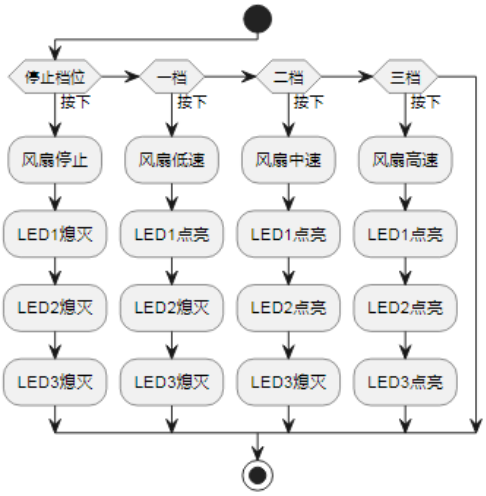
1. Utilice el puerto IO TRIG para activar el rango y proporcionar una señal de nivel alto de al menos 10us.
2. El módulo envía automáticamente 8 ondas cuadradas de 40kHz y detecta automáticamente si hay una señal de retorno.
3. Cuando se recibe una señal de retorno, se emite un nivel alto a través del puerto IO ECHO. La duración del nivel alto es el tiempo transcurrido desde la emisión hasta el retorno de la onda ultrasónica.

   Distancia de prueba= (Tiempo de alto nivel * Velocidad del sonido (340m/s))/2

Aplicación integral 1

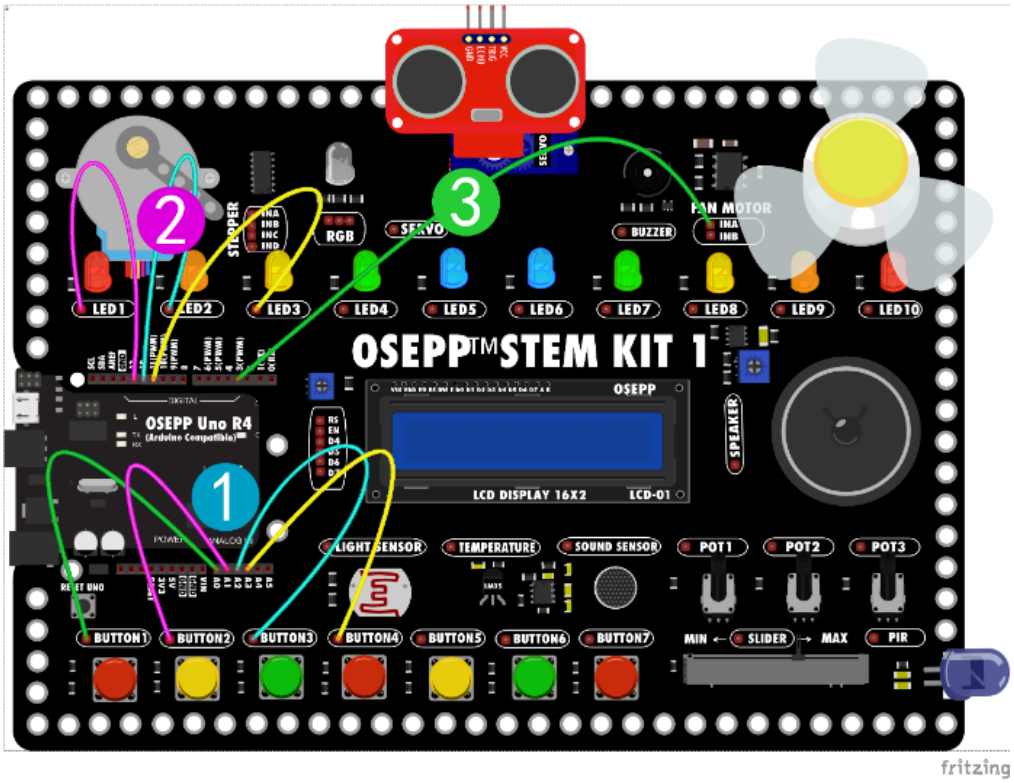
Pequeño ventilador con control de marcha

Un pequeño ventilador con control de tres marchas. Hay una marcha de parada y las tres restantes son de baja velocidad, velocidad media y alta velocidad.



Conexión

1. Los interruptores `Button1-Button4` están conectados a los pines `A0-A4` de la placa UNO respectivamente.
2. Conecte los `LED1-LED3` a los pines `13-11` de la placa UNO respectivamente.
3. Conecte el pin `INA` del motor ventilador al pin `3` de la placa UNO.



Construcción de programa

- Programa Arduino

```
void setup()
{
    //button1
    pinMode(A0, INPUT); //定义开关1
    //button2
    pinMode(A1, INPUT); //定义开关2
    //button3
    pinMode(A2, INPUT); //定义开关3
    //button4
    pinMode(A3, INPUT); //定义开关4
    //led1
    pinMode(13, OUTPUT); //定义LED1
    //led2
    pinMode(12, OUTPUT); //定义LED2
    //led3
    pinMode(11, OUTPUT); //定义LED3
    //fan1
    pinMode(3, OUTPUT); //定义风扇引脚
    pinMode(5, OUTPUT);
}

void loop()
{
    if (digitalRead(A0) == LOW) //如果开关1按下
    {
        analogWrite(3, 0); //风扇停止运行
        analogWrite(5, 0);
        digitalWrite(13, LOW); //LED1熄灭
        digitalWrite(12, LOW); //LED2熄灭
        digitalWrite(11, LOW); //LED3熄灭
    }
    else if (digitalRead(A1) == LOW) //如果开关2按下
    {
        analogWrite(3, 50); //风扇减速运行
        analogWrite(5, 0);
        digitalWrite(13, HIGH); //LED1点亮
        digitalWrite(12, LOW);  //LED2熄灭
        digitalWrite(11, LOW);  //LED3熄灭
    }
```

```
40      else if (digitalRead(A2) == LOW) //如果开关3按下
41      {
42          analogWrite(3, 120); //风扇中速运行
43 44       analogWrite(5, 0);
45          digitalWrite(13, HIGH); //LED1点亮
46          digitalWrite(12, HIGH); //LED1点亮
47          digitalWrite(11, LOW);  //LED3熄火
48      }
49      else if (digitalRead(A3) == LOW) //如果开关4按下
50      {
51          analogWrite(3, 255); //风扇高速运行
52          analogWrite(5, 0);
53          digitalWrite(13, HIGH); //LED1点亮
54          digitalWrite(12, HIGH); //LED1点亮
55          digitalWrite(11, HIGH); //LED1点亮
56      }
    }
```

Resultados de la operación

Cuando se presiona el `Button2` el ventilador se encenderá a baja velocidad y se encenderá el 1 LED al mismo tiempo, al presionar el `Button3`, el ventilador se encenderá a velocidad media y se encenderán 2 LED al tiempo, al presionar el `Button4` la velocidad del ventilador aumenta y se encenderán 3 LED al mismo tiempo. Cuando se presiona el `Button1`, el ventilador se detiene y los LED se apagan.

Aplicación integral 2

Lámpara de ritmo musical

Cuando suene la música, el LED del tablero de aprendizaje parpadeará junto con la música. ¿No es interesante?

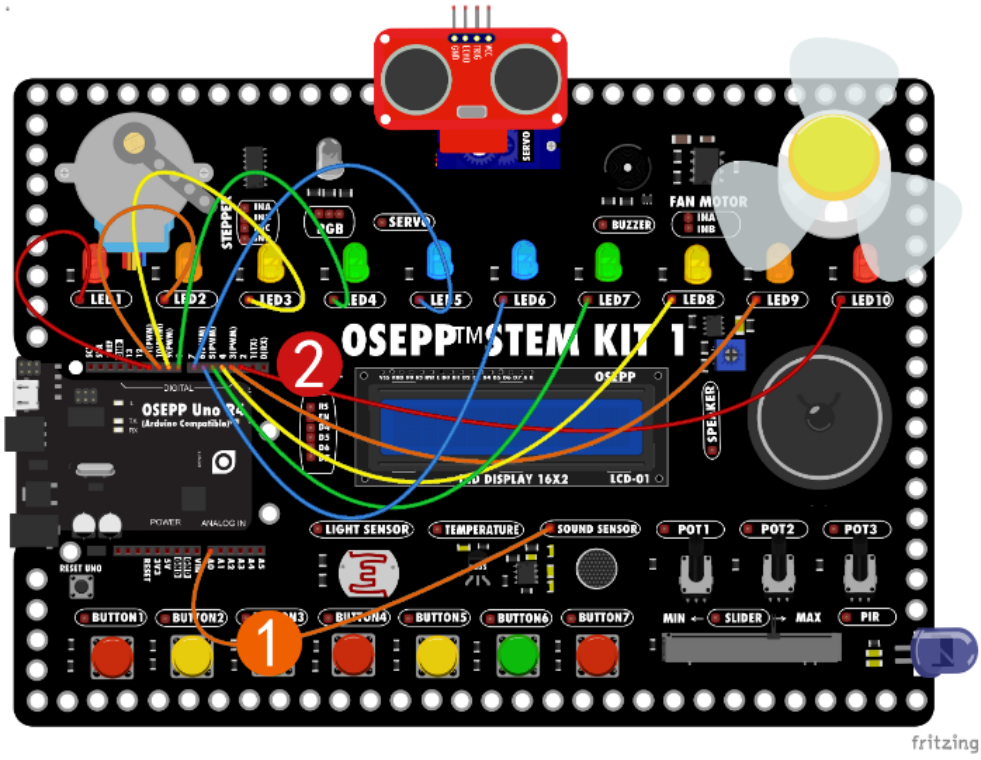
Podemos utilizar un micrófono para recoger señales de sonido y luego procesarlas a través de un programa. Deje que el LED del tablero de aprendizaje se ilumine con el ritmo de la música.

Conexión

1. Micrófono conectado al pin `A0`

2. Conexión LED, el LED1 a la izquierda corresponde al pin 11 y el LED10 a la derecha corresponde al pin 2

Número de serie	Terminales LED	Pin de la placa UNO
1	LED1	11
2	LED2	10
3	LED3	9
4	LED4	8
5	LED5	7
6	LED6	6
7	LED7	5
8	LED8	4
9	LED9	3
10	LED10	2



Construcción de programa

- Programa Arduino

```
int i = 0;       //LED引脚变量
float sound = 0; //计算映射值变量
int LEDs = 0;    //映射值变量
float ad = 0;    //麦克风电压值

void soudLED() //计算映射值子程序
{
  ad = analogRead(A0); //读取麦克风口电位
  if (ad < sound)      //如果口电位小于上次
  {
    sound = sound * 0.999 + ad * 0.001;
    // 一阶低通滤波器算法，也就是声音变小时，LED响应慢一点。
  }
  else
  {
    sound = sound * 0.95 + ad * 0.05;
    // 一阶低通滤波器算法，减慢LED的响应速度。但是比上面声音变小时要快一点。
  }
  LEDs = map(sound, 20, 200, 0, 4); //映射计算口读的值，映射值
}

void setup()
{
  //led1
  pinMode(11, OUTPUT); //定义LED引脚模式
  //led2
  pinMode(10, OUTPUT);
  //led3
  pinMode(9, OUTPUT);
  //led4
  pinMode(8, OUTPUT);
  //led5
  pinMode(7, OUTPUT);
  //led6
  pinMode(6, OUTPUT);
  //led7
  pinMode(5, OUTPUT);
  //led8
  pinMode(4, OUTPUT);
  //led9
  pinMode(3, OUTPUT);
  //led10
  pinMode(2, OUTPUT);
  //soundSensor1
  pinMode(A0, INPUT); //定义麦克风引脚模式
}

```

```
void loop()
{
  soudLED();              //计算映射位于积分，利声麦克风的位计算出来0-4之间的数
  for (i = 0; i < 5; i++) //循环i
  {
    if (LEDs > i) //如果麦克风映射位大于i
    {
      digitalWrite(i + 7, HIGH); //点亮7-11号引脚的LED
      digitalWrite(6 - i, HIGH); //点亮6-2号引脚的LED
    }
    else //否则
    {
      digitalWrite(i + 7, LOW); //熄灭7-11号引脚的LED
      digitalWrite(6 - i, LOW); //熄灭6-2号引脚的LED
    }
  }
}
```

Resultados de la operación

Después de cargar el código, usa tu teléfono para reproducir música junto al micrófono y el LED del tablero de aprendizaje parpadeará al ritmo de la música.

Análisis

Puedes intentar cambiar el valor en el código a 20.200. 20 corresponde al LED de punto de inicio, que puede ajustar la sensibilidad del disparador. 200 corresponde a la sensibilidad de todo el rango. Cuando el sonido sea fuerte, aumente el valor. O si la cantidad de luces LED es pequeña cuando el sonido es bajo, cambie este valor a un valor más pequeño.

Hay 5 LED en un lado, por lo que los valores detrás de ellos son 0 ~ 4, que son 5 números.

```
LEDs = map(sound, 20, 200, 0, 4);
```

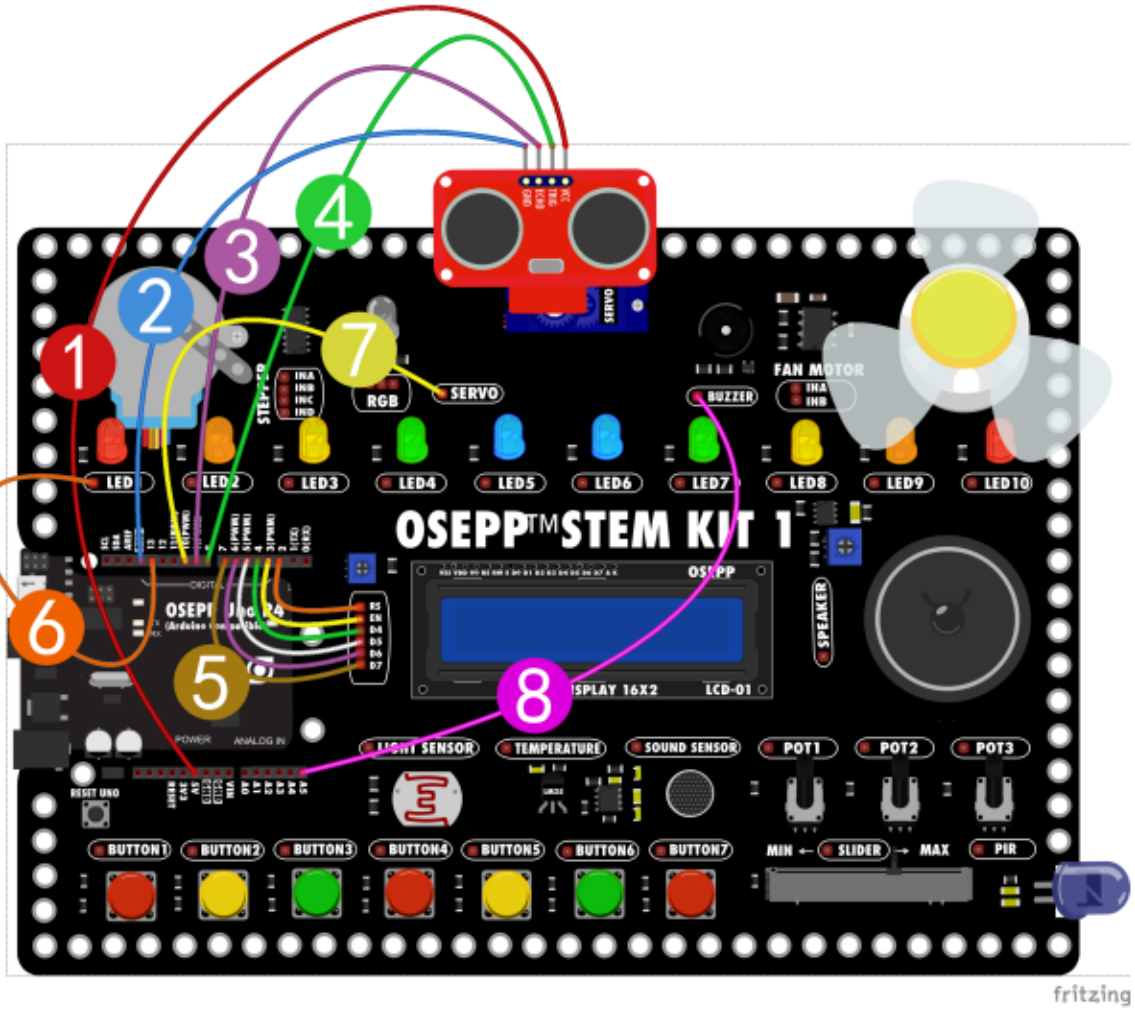
Aplicación integral 3

Radar de crucero ultrasónico

La antena del radar del vehículo gira y escanea de un lado a otro. ¿No te parece de alta tecnología? También podemos hacer uno, pero aquí utilizamos ecografía. Utilice el servo para realizar la navegación y el ultrasonido para escanear objetos. Cuando el escáner ultrasónico detecta un obstáculo, se detendrá y mostrará la distancia del obstáculo.

Conexión

1. El pin VCC del módulo ultrasónico va conectado al pin 5V de la placa UNO.
2. El pin GND del módulo ultrasónico va conectado al pin GND de la placa UNO.
3. El pin ECH0 del módulo ultrasónico va conectado al pin 9 de la placa UNO.
4. El pin TRIG del módulo ultrasónico va conectado al pin 8 de la placa UNO.
5. Las interfaces RS-D7 del LCD van conectadas a los pines 2~7 de la placa UNO respectivamente.
6. Conectar el LED1 al pin 13 de la placa UNO.
7. El pin del Servo va conectado al pin 10 de la placa UNO.
8. El pin del Buzzer va conectado al pin A5 de la placa UNO.



Construcción de programas

- Programa Arduino

```
#include <LiquidCrystal.h>
#include <Servo.h>
#include <oseppRobot.h>

LiquidCrystal lcd1(2, 3, 4, 5, 6, 7); //定义LCD引脚
Servo servo1;
OseppUltrasonic ultrasonic1(8, 9); //定义超声波模块引脚
int i = 0;                          //定义变量i

void saomiao() //子程序，舵机扫描
{
    servo1.write(i); //舵机旋转到变量i
    delay(500);      //延时500毫秒
}

void xianshi() //显示子程序
{
    lcd1.clear();                                //LCD清屏
    lcd1.setCursor(0, 0);                        //LCD光标定位
    lcd1.print(ultrasonic1.ping());              //LCD显示当前距离
    lcd1.print("mm");                            //距离单位
    lcd1.setCursor(0, 1);                        //LCD光标定位
    lcd1.print(i);                               //LCD显示当前舵机角度
    digitalWrite(13, ultrasonic1.ping() < 400); //距离小于400，LED点亮
}

void baojin() //报警声音子程序
{
    digitalWrite(A5, HIGH); //蜂鸣器输出声音
    delay(20);              //延时20毫秒
    digitalWrite(A5, LOW);  //蜂鸣器停止输出
}

void setup()
{
    lcd1.begin(16, 2); //LCD初始化
    //buzzer1
    pinMode(A5, OUTPUT);
    servo1.attach(10); //定义蜂鸣器引脚
    //led1
    pinMode(13, OUTPUT); //定义LED1引脚
}

```

```
void loop()
{
    while (i < 180) //如果变量i小于180
    {
        saomiao();                      //执行扫描子程序
        xianshi();                      //执行显示子程序
        if (ultrasonic1.ping() < 400) //如果距离小于400
        {
            baojin(); //执行报警程序
        }
        else
        {
            i += 10; //否则变量i加10
        }
    }
    while (i > 0) //如果变量i大于0
    {
        saomiao();                      //执行扫描子程序
        xianshi();                      //执行显示子程序
        if (ultrasonic1.ping() < 400) //如果距离小于400
        {
            baojin(); //执行报警程序
        }
        else
        {
            i -= 10; //否则变量i减去10
        }
    }
}
```

Resultados de la operación

Una vez iniciado el programa, el módulo ultrasónico instalado en el servo navegará y escaneará entre `0-180` grados. Si hay un obstáculo dentro de este rango y la distancia del obstáculo es menor a `400mm`, el servo se detendrá, el LED se iluminará y el zumbador hará sonar una alarma. Cuando se elimina el obstáculo, el LED se apaga, el sonido de la alarma se levanta y el escaneo de crucero continúa.