PRÁCTICA DE ARDUINO - CONTROL DE SERVO MOTOR

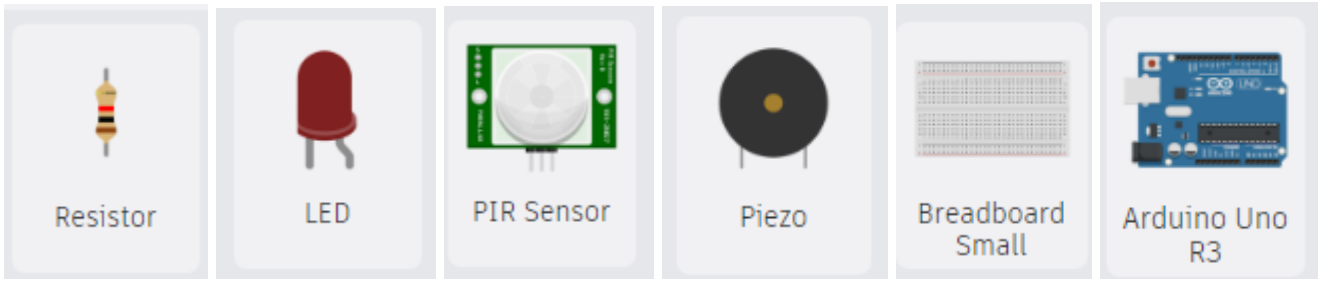
NIVEL: 1
DURACIÓN: 2 horas
OBJETIVO: Los estudiantes aprenderán a crear un sistema de alarma utilizando un sensor de movimiento PIR (Passive Infrared Sensor).
MATERIALES

- Placa Arduino Uno (o similar)
- Sensor PIR
- Buzzer
- Resistencia de 220 ohmios
- Protoboard
- Cables de conexión
- Computadora con el software Arduino IDE instalado
- ThinkerCad

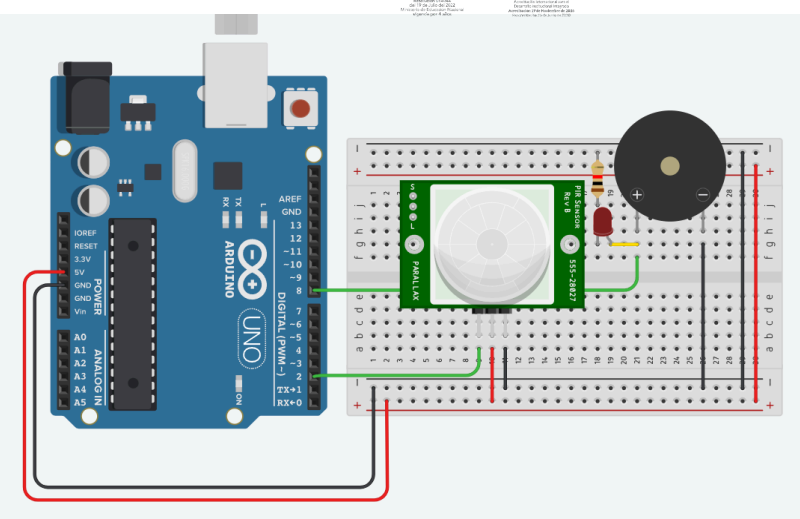
1. DESARROLLO DE LA PRÁCTICA

1.1 SIMULACIÓN EN THINKERCAD

1.1.1 COMPONENTES



1.1.2 MONTAJE



2.2 PROGRAMACIÓN EN ARDUINO IDE

```
// Definir los pines
const int pirPin = 2;// Pin donde se conecta la salida del sensor PIR
const int alarmaPin = 8; // Pin donde se conecta el buzzer y LED

int pirEstado = LOW;// Estado inicial del PIR

void setup() {
  // Configurar los pines
  pinMode(pirPin, INPUT); // Pin del PIR como entrada
  pinMode(alarmaPin, OUTPUT);// Pin del buzzer y LED como salida

  // Iniciar la comunicación serial
  Serial.begin(9600);
}

void loop() {
  // Leer el estado del PIR
  pirEstado = digitalRead(pirPin);

  // Si se detecta movimiento (PIR en HIGH)
  if (pirEstado == HIGH) {
    digitalWrite(alarmaPin, HIGH);// Activar la alarma
    Serial.println("Movimiento detectado! Alarma activada.");
  } else {
    digitalWrite(alarmaPin, LOW); // Apagar la alarma
    Serial.println("Sin movimiento.");
  }

  delay(100);// Pequeño retardo
}
```

Explicación

- El pin pirPin está conectado a la salida del sensor PIR. Usamos digitalRead() para leer si el PIR detecta movimiento (estado HIGH) o no (estado LOW).
- Si el PIR detecta movimiento (estado HIGH), activamos la alarma utilizando digitalWrite() para encender el buzzer o LED (dependiendo de lo que hayas conectado). Si no se detecta movimiento, apagamos la alarma.
- El código incluye la función Serial.println() para mostrar en el Monitor Serial si se ha detectado movimiento o no, lo que puede ayudar a monitorear el estado del sensor mientras se ejecuta el programa.
- El sensor PIR tiene dos estados posibles: HIGH (detecta movimiento) o LOW (no detecta movimiento).
- Si el valor de pirEstado es HIGH, significa que se ha detectado movimiento. En este caso, el código dentro del bloque if se ejecuta:
    - Se activa la alarma encendiendo el buzzer o el LED (digitalWrite(alarmaPin, HIGH)).
    - Se muestra el mensaje "Movimiento detectado! Alarma activada." en el Monitor Serial.
- Si la condición del if es falsa (es decir, no se detecta movimiento y pirEstado es LOW), el código dentro del bloque else se ejecuta. En este caso, el buzzer o LED se apaga (digitalWrite(alarmaPin, LOW)), y se muestra en el Monitor Serial el mensaje "Sin movimiento.".

2. ANÁLISIS DE RESULTADOS

- ¿Cómo funciona el sensor PIR para detectar movimiento?

El sensor PIR detecta el movimiento a través de la detección de cambios en la radiación infrarroja emitida por los objetos en su entorno. Cuando un objeto caliente (como una persona) pasa frente al sensor, este detecta la variación de los niveles de radiación IR y activa su señal de salida. El sensor distingue entre el fondo "estático" y el movimiento.

- ¿Qué sucede si hay movimientos muy rápidos o en ángulos difíciles de detectar?

El sensor PIR tiene un ángulo de detección limitado (generalmente alrededor de 120°) y una distancia máxima de detección que puede variar entre 5 y 10 metros, dependiendo del modelo. Movimientos muy rápidos o que ocurran fuera del rango de detección pueden no ser captados por el sensor. Para mejorar la cobertura, se podría agregar otro sensor PIR o ajustar la posición del sensor actual.

- ¿Qué podemos hacer si queremos que la alarma suene por un tiempo definido incluso después de que el movimiento ya no sea detectado?

Para hacer que la alarma suene durante un período de tiempo definido podemos usar un temporizador. Por ejemplo, una vez que se detecta movimiento, podríamos mantener la alarma activada durante 5 segundos antes de apagarla:

```
if (pirEstado == HIGH) {
  digitalWrite(alarmaPin, HIGH);  // Activar alarma
  delay(5000);  // Mantenerla encendida durante 5 segundos
  digitalWrite(alarmaPin, LOW);   // Apagar la alarma después de 5 segundos
}
```

- ¿Cómo podemos mejorar este sistema para incluir más funcionalidades?

- Se podría incluir un botón para desactivar manualmente la alarma.
- Mostrar el estado del sistema en una pantalla LCD, indicando cuándo la alarma está activada o desactivada.
- Conectar un módulo de grabación o una cámara para capturar imágenes o videos cuando se detecte movimiento.
- Usar un módulo WiFi (como el ESP8266) para enviar una notificación al teléfono móvil cuando se detecte movimiento, convirtiendo el sistema en una alarma inteligente.

3. CONCLUSIONES

a. El sensor PIR es una herramienta eficaz y fácil de usar para detectar movimiento en su entorno, utilizando la radiación infrarroja emitida por los objetos en movimiento. Esto lo hace ideal para proyectos de seguridad y automatización del hogar.

b. Este proyecto demuestra cómo se puede activar una alarma sencilla, como un buzzer o un LED, utilizando un sensor PIR y un Arduino. El uso de digitalRead() y digitalWrite() facilita la detección y la respuesta a la señal del sensor.

c. El sistema puede expandirse agregando características como temporizadores, botones de desactivación, más sensores o conectividad a una red para convertirlo en una alarma inteligente y avanzada.

d. Este tipo de sistema es aplicable en una variedad de entornos, como alarmas de seguridad en casas, oficinas o para encender luces automáticamente cuando alguien entra en una habitación.

e. Aunque es un sistema sencillo, la combinación de un sensor PIR con Arduino es muy escalable, permitiendo la integración con otros dispositivos y funciones para crear un sistema de alarma más robusto y personalizable.