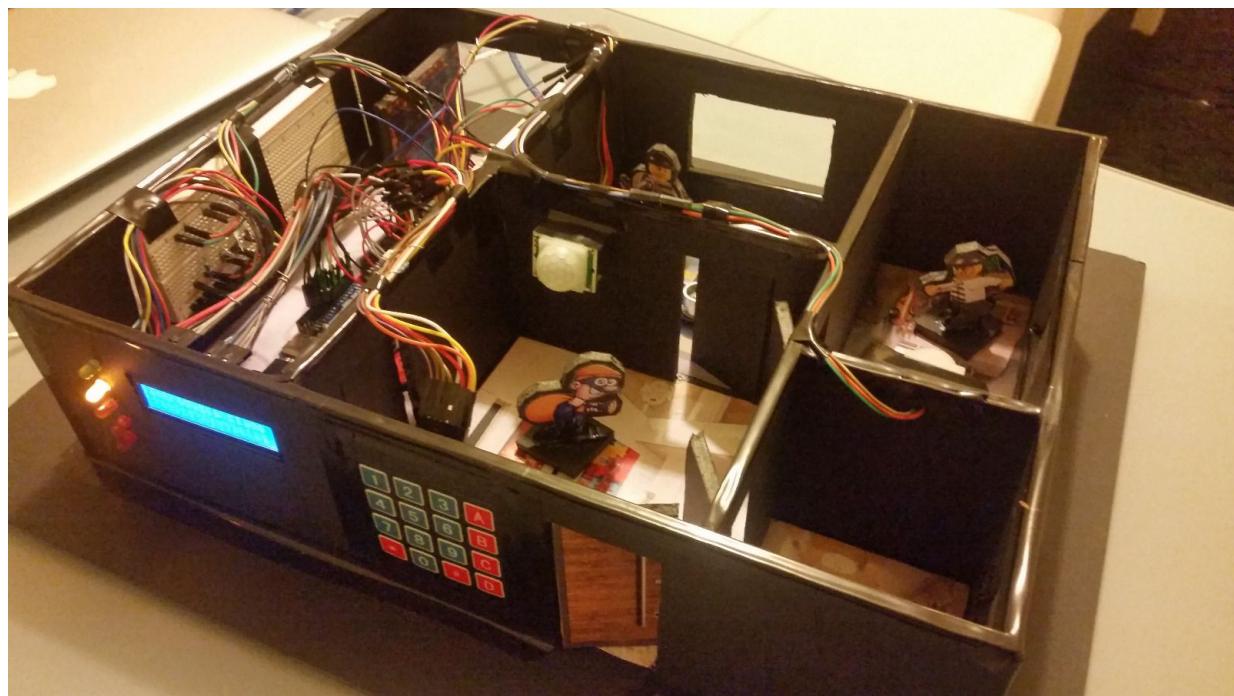


Sistema de detección de intrusos con alarma

Proyecto realizado para la asignatura:
“Didáctica de la informática y la tecnología II”



Realizado por:
Carlo Francesco Gandini Luca
Jennifer Hernández Bécares
Daniel Mayoral Sanz
Ana Isabel del Pino Calderón

Índice

1. Introducción	3
2. Descripción del proyecto	5
3. Tecnologías y material empleados	6
3.1. Material empleado (circuito)	6
3.2. Material empleado (maqueta)	6
4. Montaje del proyecto	7
4.1. Práctica 1: Sensor ultrasonidos	7
4.2. Práctica 2: Sensor PIR	12
4.3. Práctica 3: Integración sensores PIR y ultrasonidos	17
4.4. Práctica 4: Pantalla LCD y teclado 4x4	22
4.5. Práctica 5: Conexión entre los dos arduinos	29
5. Proyecto final. Sistema de detección de intrusos	33
5.1. Contexto	33
5.2. Diagramas de flujo	34
5.3. Construcción de la maqueta	37
5.3.1. Medidas	37
5.3.2. Descripción de la Práctica	37
5.3.3. Fotos del montaje	39
5.4. Simulación	42
5.5. Código	43
5.6. Resultado	44
5.7. Vídeo	46
6. Coste	47
7. Viabilidad y escalabilidad	48
8. Pruebas realizadas	49
9. Posibles extensiones	51
10. Conclusiones	52
11. Trabajo futuro	53
12. Referencias	53
Anexo 1	54
1. Código proyecto arduino 1	55
2. Código proyecto arduino 2	64

1. Introducción

¿Por qué estudiar y utilizar Arduino?

Arduino es una plataforma de software libre fácil de usar, que permite a los alumnos construir y programar sus propios proyectos. El kit ofrece un amplia variedad de componentes y sensores que permiten conectar el mundo físico con el virtual, desarrollando la concentración y las habilidades manuales. Con Arduino no solo se aprende a programar, sino que también anima a pensar creativamente, analizando situaciones y aplicando el pensamiento crítico y habilidades para resolver problemas reales.

La robótica educativa, dentro de los colegios, nos proporciona una forma creativa de utilizar y comprender las nuevas tecnologías, la posibilidad de implementar soluciones basadas en ingenio y destreza, dejando de ser solamente consumidores.

La plataforma de Arduino mantiene una posición muy cercana al mundo de la educación, y es de gran uso en los cursos de Secundaria y Bachillerato. Aprender es fácil y, como es habitual, la clave es dar con todas esas webs con recursos, guías y tutoriales para ir avanzando en el aprendizaje de esta plataforma de electrónica. Pero no se trata solo de un proyecto tecnológico, es mucho más, detrás de él existe una comunidad en movimiento y expansión, que incluye programadores, artistas, educadores, científicos, estudiantes entre muchos otros y que inspira la filosofía de la cultura libre, permitiendo compartir experiencias y resolver problemas vía online.

¿Cuales son los beneficios de la robótica en el aula?

El objetivo de este curso es proporcionar los conocimientos necesarios, utilizando principalmente ejemplos prácticos, para crear unidades didácticas en asignaturas de Secundaria en la que se enseñe Arduino. Durante el curso se aprenderá a utilizar una placa Arduino, el entorno de programación y un kit de desarrollo.

En educación pueden diferenciarse dos tipos de uso de la programación y la robótica como apoyo en clase: por un lado, la robótica y la programación educacional, consiste en un elemento físico o de programación que motiva a los estudiantes a construir, programar, razonar de manera lógica; por otro, la programación y la robótica como elemento social, utilizandolo como forma de juego.

En referencia al primer tipo de uso, el educacional, la tecnología de programación y robótica es especialmente beneficiosa en la enseñanza de asignaturas: ciencia, tecnología, ingeniería y matemáticas. Sin embargo, resulta desfavorable centrarse sólo en su uso como herramienta de enseñanza en estas materias y no ser también desplegadas para otras materias, que permiten utilizarlas como elemento de socialización.

Otro beneficio en el uso de estas herramientas sería, en el caso alumnos con necesidades especiales, tanto en las áreas cognitivas como psicosociales. La potencialidad de las propuestas educativas basadas en robots lo hacen especialmente útil en programas de refuerzo y de educación especial por su motivación.

Estamos convencidos de que los dispositivos tangibles aumentan el nivel de comprensión debido a que los estudiantes están manipulando las cosas en un mundo real. Sin embargo, podemos encontrar alumnos que prefieren trabajar con dispositivos no tangibles. Por tanto, lo que parece lógico es un enfoque híbrido entre robótica y programación, donde una fusión entre lo físico y lo virtual proporciona más flexibilidad a los docentes y a los estudiantes.

La principal suposición de este enfoque es que la interacción con robots puede reforzar los procesos educativos y los resultados, tales como el aprendizaje conceptual y el entrenamiento cognitivo, motivar a los estudiantes, apoyar la curiosidad y aumentar la conciencia sobre la robótica.

La metodología empleada pretende fomentar las capacidades del alumnado, estimulando la investigación, la observación y la imaginación, así como el hábito de trabajo individual y en equipo. Se procura fomentar la actitud de la curiosidad, así como humanísticos, mediante contenidos específicos, actividades y experiencias prácticas. Algunos de estos contenidos contribuyen a la consolidación de varias competencias.

La **idea** de este proyecto surgió gracias al profesor José Luis Vázquez en la asignatura de Fundamentos de Tecnología, ya que propuso un proyecto con nombre **Sistemas de detección de intrusiones**. A pesar de ello, el contenido del proyecto es completamente nuestro. Nos hemos basado principalmente en los tutoriales de cada uno de los elementos del tutorial de Elegoo y a partir de ahí hemos ido mejorando la idea e incorporando más cosas.

Después de escoger hacer este proyecto y avanzar bastante, hemos encontrado en youtube cosas parecidas a lo que queríamos realizar, como por ejemplo la siguiente:

https://www.youtube.com/watch?v=dRCnccv_dVE

La diferencia es que en ese proyecto incorporan cámaras en lugar de utilizar sensores PIR y de ultrasonidos. A pesar de ello, la idea de cómo disponer los elementos en la maqueta para el proyecto final la hemos sacado de ese vídeo.

2. Descripción del proyecto

Este proyecto tiene como objetivo realizar una alarma doméstica, un sistema de detección de intrusos que permita avisar al usuario de la intromisión de agentes externos en su domicilio.

La realización de dicho proyecto se ha llevado a cabo por un equipo compuesto por 4 personas, y se ha dividido en 5 prácticas:

- Práctica 1: Módulo sensor ultrasonidos
- Práctica 2: Módulo sensor PIR
- Práctica 3: Montaje de ambos sensores
- Práctica 4: Teclado matricial y pantalla LCD
- Práctica 5: Conexión de 2 Arduinos

El desarrollo de las prácticas y su unificación conforman el proyecto final, cumpliendo con el objetivo que se perseguía y permitiendo que los miembros del equipo trabajen de forma autónoma en cada parte.

Mediante la construcción de la maqueta de una casa en la que se ha instalado el sistema de alarma, aportamos una visión realista del funcionamiento y aplicación del proyecto.

El funcionamiento de la alarma doméstica permitirá configurar el dispositivo para tener realizar varias funciones:

1. Activar alarma
2. Activar ½ alarma
3. Cambiar contraseña

Más adelante se explicará en qué consiste cada una de las funciones.

3. Tecnologías y material empleados

A pesar de que más adelante se explicará qué se ha usado y cómo se ha usado todo el material, aquí listamos la lista de materiales que hemos empleado para llevar a cabo nuestro proyecto final a modo de resumen:

3.1. Material empleado (circuito)

- 2 Arduino UNO
- 2 placas breadboard
- Leds de colores: 3 rojos, 1 verde y 1 amarillo
- 1 buzzer
- 1 Keypad 4x4
- 1 LCD 16x2
- Resistencias de 220Ω
- 1 potenciómetro 250kΩ
- 1 Módulo de sensor ultrasonidos HC-SR04
- 2 sensores PIR HC-SR501
- Cables de conexión
- Cables de hembra a macho DuPont
- Alimentación: ordenador o powerbank con conexión para 2 puertos USB

3.2. Material empleado (maqueta)

- 2 Pliegos Cartón pluma, 594 x 420 mm de color negro.
- 1 Bote de pegamento o cola especial (fija espuma rígida, espuma de poliestireno, es flexible y resistente al envejecimiento) 50 ml
- Cola Vinilica 100gr o pegamento en barra (para pegar imágenes)
- 1 Lapicero HB
- 1 Goma de borrar
- 1 Tijera
- 1 Cutter
- 1 Escalímetro o cualquier herramienta de medida
- 1 Cinta autoadhesiva a elegir color
- Imágenes decorativas de cada ambiente (habitaciones de la casa)

4. Montaje del proyecto

4.1. Práctica 1: Sensor ultrasonidos



4.1.1. Descripción de la práctica : Módulo Sensor Ultrasonidos

El objetivo principal de esta práctica es aprender a utilizar el módulo sensor de ultrasonido. El sensor de ultrasonido es el responsable de medir distancias para evitar obstáculos.

En lugar de utilizar el sensor de ultrasonido de forma independiente, se han asociado dos led, uno de color rojo y otro de color verde; y un zumbador de sonido (más adelante buzzer). Cuando el sensor detecte movimiento, se encenderá el led rojo y el buzzer emitirá sonido. En caso de no detectar movimiento, permanecerá encendido el led de color verde.

4.1.2. Material utilizado

- 1 Arduino UNO
- 1 Placa breadboard
- 1 Módulo de sensor ultrasonidos HC-SR04
- 1 LED rojo
- 1 LED verde
- 1 Buzzer
- 2 Resistencias de 220Ω
- 9 Cables de hembra a macho DuPont

4.1.3. Descripción del Componente

El Módulo HC-SR04 es un sensor de distancias por ultrasonidos capaz de detectar objetos y calcular la distancia a la que se encuentra en un rango de 2 a 450 cm. El sensor funciona por ultrasonidos y contiene toda la electrónica encargada de hacer la medición. Su uso es tan sencillo como enviar el pulso de arranque y medir la anchura del pulso de retorno.

El sensor incorpora un par de transductores de ultrasonido que se utilizan de manera conjunta para determinar la distancia del sensor con un objeto colocado enfrente de este. La interfaz digital se logra mediante 2 pines digitales: el pin de trigger (disparo) y echo (eco).

1. El primero recibe un pulso de habilitación de parte del microcontrolador, mediante el cual se le indica al módulo que comience a realizar la medición de distancia.
2. El segundo Pin (echo) el sensor “muestra” al microcontrolador un pulso cuyo ancho es proporcional al tiempo que tarda el sonido en viajar del transductor al obstáculo y luego de vuelta al módulo.

4.1.4. Simulación

En este apartado podemos ver la simulación de la práctica en el programa Tinkercad.

En la imagen 1, Se aprecia que el ultrasonido está encendido y el led está en verde porque no detecta ningun obstaculo en una distancia mayor a 200 cm.

En la imagen 2. Se aprecia que el ultrasonido está encendido y el led está rojo porque detecta un obstáculo en una distancia igual o inferior a 200 cm. En este caso, el buzzer emite sonido para que se identifique la presencia de un obstáculo.

Imagen 1

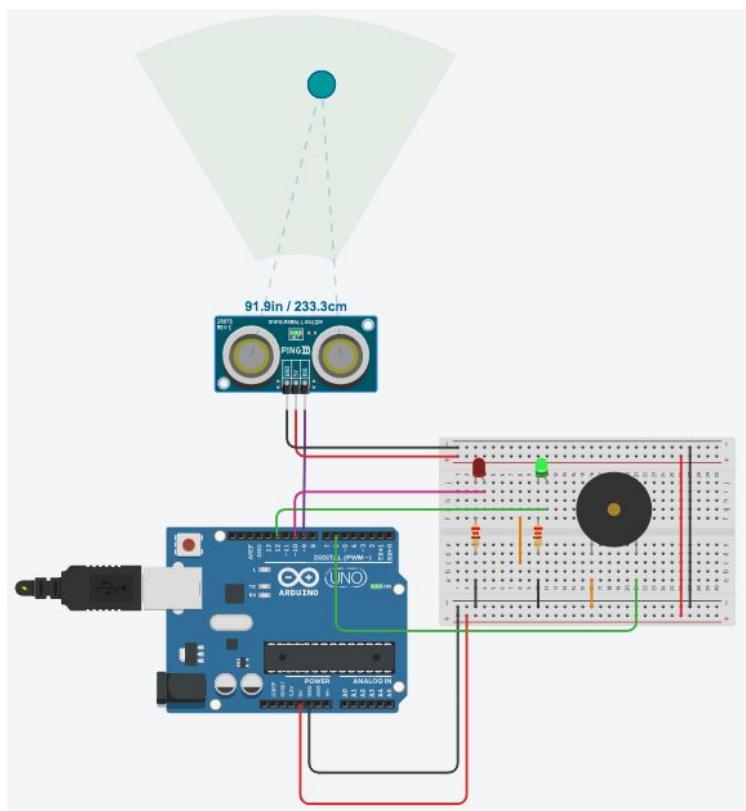
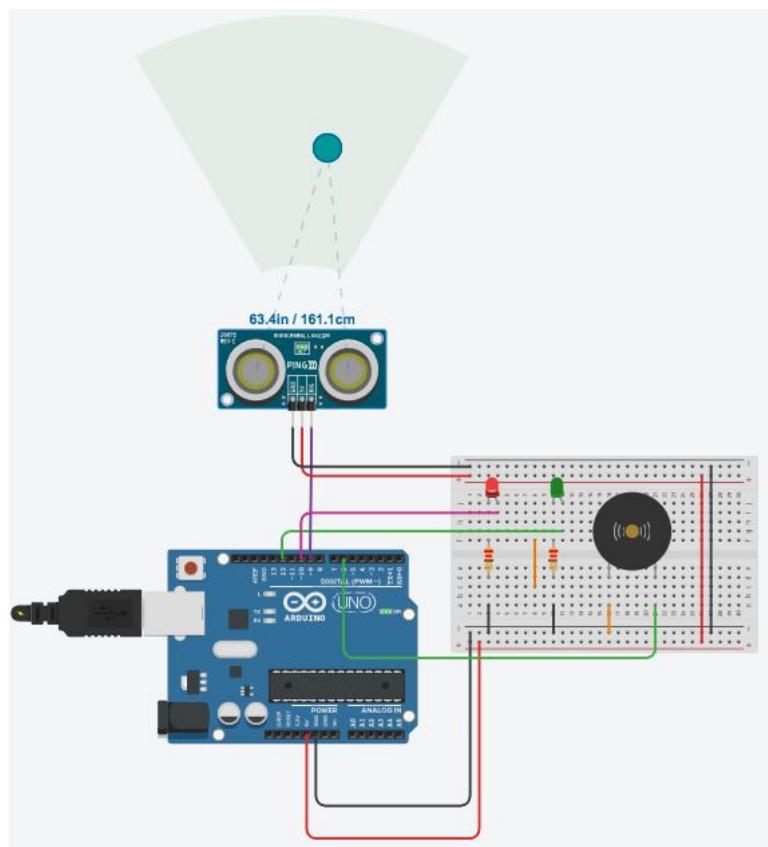


Imagen 2



4.1.5. Código comentado

Después de efectuar el cableado, se abre el programa “IDE de Arduino” y escribimos este código para que la placa Arduino R1 reconozca las diferentes partes de la práctica, e interactúen como se ha descrito anteriormente. Una vez escrito el código lo subimos a la placa y comprobamos que funciona.

```
int ledPinRojo = 10, ledPinVerde = 12;
int pinBuz = 6; // BUZZ
int ultrasonido = 9;

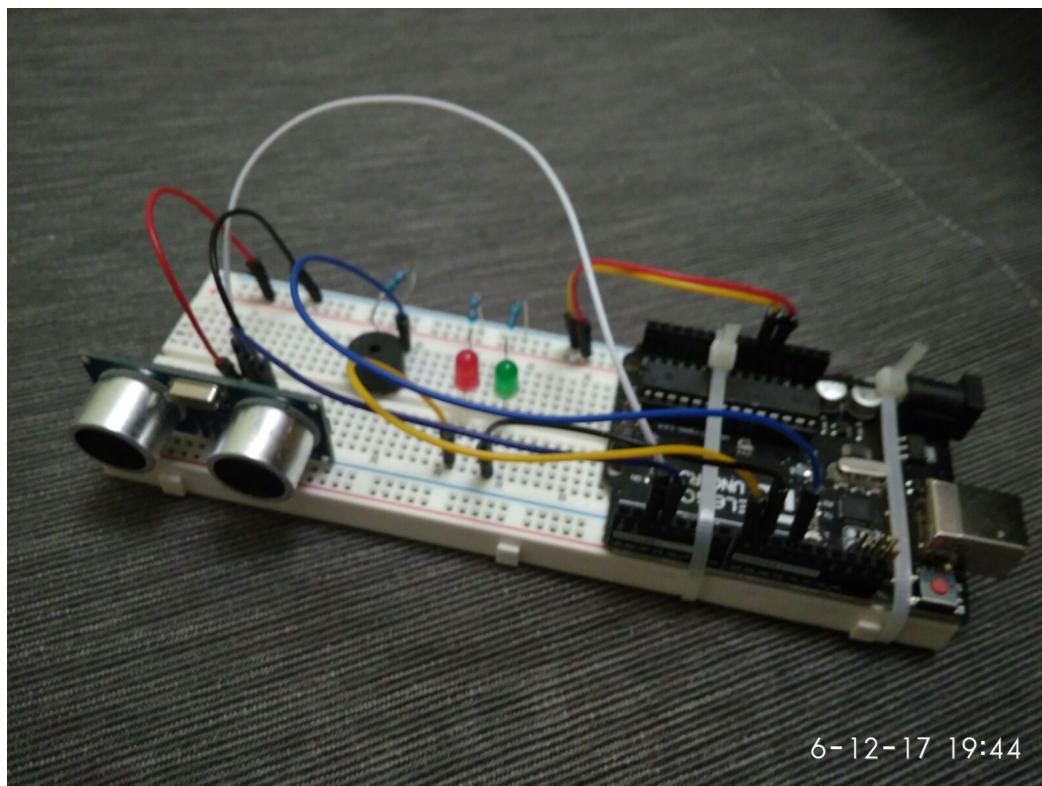
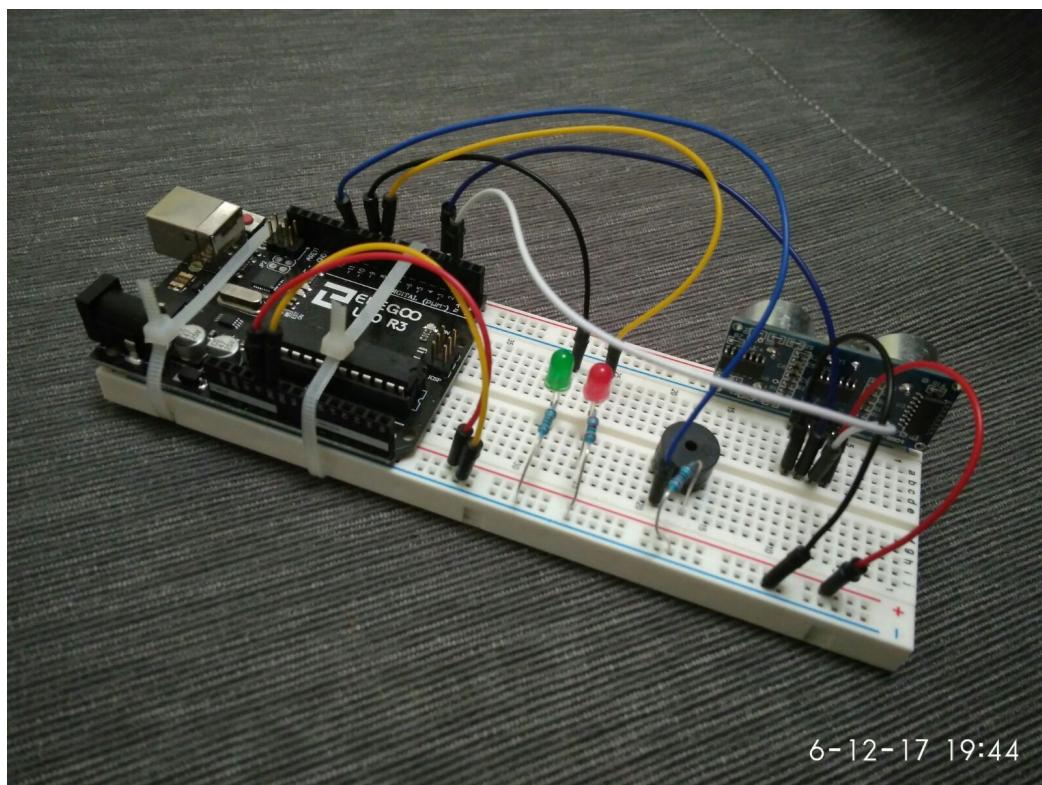
void setup() {
    Serial.begin(9600);
    pinMode(ledPinRojo, OUTPUT);
    pinMode(ledPinVerde, OUTPUT);
}

long readUltrasonicDistance(int pin) {
    pinMode(pin, OUTPUT); // Clear the trigger
    digitalWrite(pin, LOW);
    delayMicroseconds(2);
    digitalWrite(pin, HIGH);
    delayMicroseconds(10);
    digitalWrite(pin, LOW);
    pinMode(pin, INPUT);
    // Reads the pin, and returns the sound wave travel time in
    // microseconds
    return pulseIn(pin, HIGH);
}

void loop() {
    // Leemos la distancia del sensor ultrasonidos
    int cm = 0.01723 * readUltrasonicDistance(ultrasonido);

    // si la distancia es menor de 2 metros, encendemos el led
    // rojo y apagamos el verde. Ademas, hacemos sonar el
    // buzzer
    if (cm < 200) {
        tone(pinBuz, 350, 200);
        digitalWrite(ledPinRojo, HIGH);
        Serial.println("Motion detected!");
        digitalWrite(ledPinVerde, LOW);
    }
    else {
        // en caso contrario, apagamos el led rojo y encendemos
        // el verde
        digitalWrite(ledPinRojo, LOW);
        digitalWrite(ledPinVerde, HIGH);
    }
}
```

4.1.6. Fotos del montaje



4.2. Práctica 2: Sensor PIR



4.2.1. Descripción de la práctica: Sensor PIR

El objetivo principal de esta práctica es aprender a utilizar el módulo sensor PIR. El sensor PIR es el responsable de detectar el movimiento de un objeto.

En lugar de utilizar el sensor PIR de forma independiente, se han asociado dos led (uno de color rojo y otro de color verde) y un buzzer. Cuando el sensor detecte movimiento, se encenderá el led rojo y el buzzer emitirá sonido. En caso de no detectar movimiento, permanecerá encendido el led de color verde.

4.2.2. Material utilizado

- 1 Arduino UNO
- 1 Placa breadboard
- 1 Sensor PIR
- 1 LED rojos
- 1 LED verde
- 1 Buzzer
- 2 resistencias de 220Ω
- 11 Cables de hembra a macho DuPont

4.2.3. Descripción del Componente

El Módulo PIR HC-SR501, es un sensor electrónico pasivo que mide la luz infrarroja (IR) emitida por los objetos situados en su campo de visión. Se utilizan principalmente en los detectores de movimiento basados en PIR.

Todos los objetos con una temperatura por encima del cero absoluto emiten calor. Por lo general, esta radiación es invisible para el ojo humano, ya que irradia en longitudes de onda infrarrojas, pero puede ser detectado por dispositivos electrónicos diseñados para tal propósito.

El término pasivo, en este caso, se refiere al hecho de que los dispositivos PIR no generan o irradian cualquier energía para fines de detección. Trabajan en su totalidad para la detección de la energía emitida por otros objetos. Es importante tener en cuenta que los sensores PIR no detectan o miden "calor" sino que detectan la radiación infrarroja emitida por un objeto. Sin embargo, a pesar de ser diferentes, a menudo la radiación infrarroja está asociada a la temperatura del objeto.

El sensor incorpora tres pines de conexión +5V, OUT (3,3v) y GND, y dos resistencias variables de calibración (Ch1 y RL2).

1. Ch1: Esta resistencia permite establecer el tiempo que se va a mantener activa la salida del sensor. Una de las principales limitaciones de este módulo es que el tiempo mínimo que se puede establecer es de más o menos 3s. Si se cambia la resistencia por otra de 100K, puede bajar el tiempo mínimo a más o menos 0,5 s.
2. RL2: Esta resistencia variable permite establecer la distancia de detección, que puede variar entre 0-3m.

4.2.4. Simulación

En este apartado podemos ver la simulación de la práctica en el programa Tinkercad.

En la imagen 1, Se aprecia que el sensor PIR está encendido y el led está en verde porque no detecta ninguna temperatura.

En la imagen 2. Se aprecia que el sensor PIR está encendido y el led está rojo porque detecta una temperatura. En este caso, el buzzer emite sonido para que se identifique la presencia de un cuerpo.

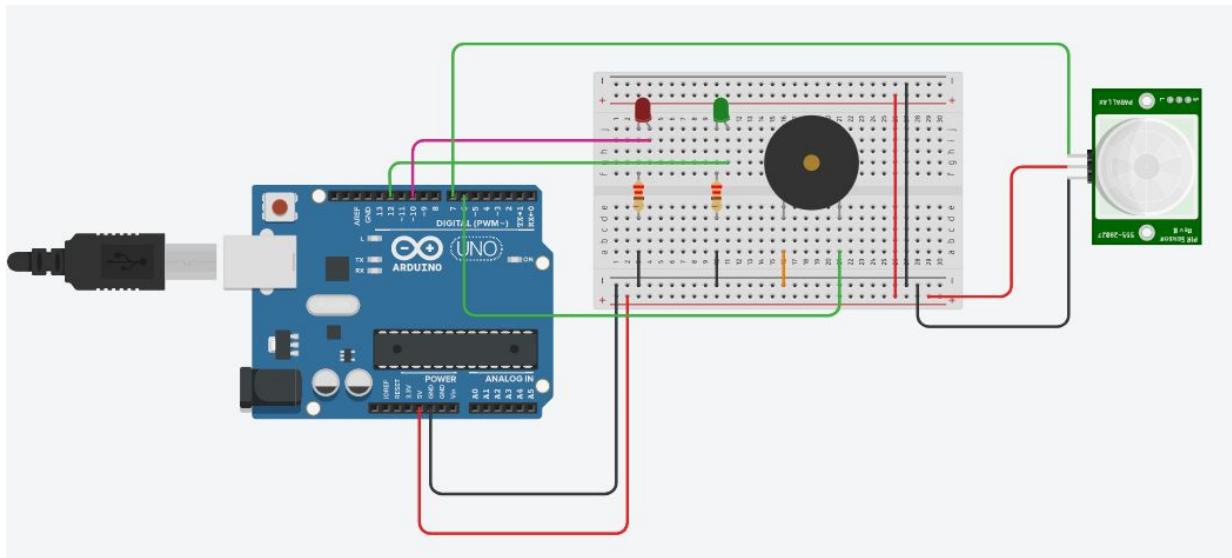


Imagen 1

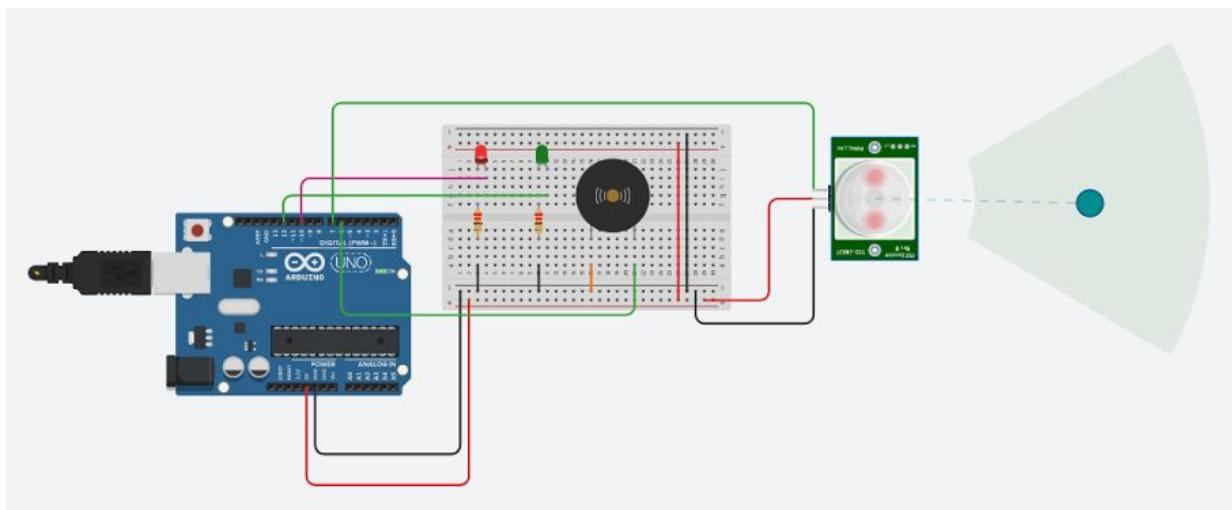


Imagen 2

4.2.5. Código comentado

Después de efectuar el cableado, se abre el programa “IDE de Arduino” y escribimos este código para que la placa Arduino R1 reconozca las diferentes partes de la práctica, e interactúen como se ha descrito anteriormente. Una vez escrito el código lo subimos a la placa y comprobamos que funciona.

```
int ledPinRojo = 10; // Led rojo
int ledPinVerde = 12; // LED verde
int pirPin = 7; // PIR
int pinBuz = 6; // Buzzer

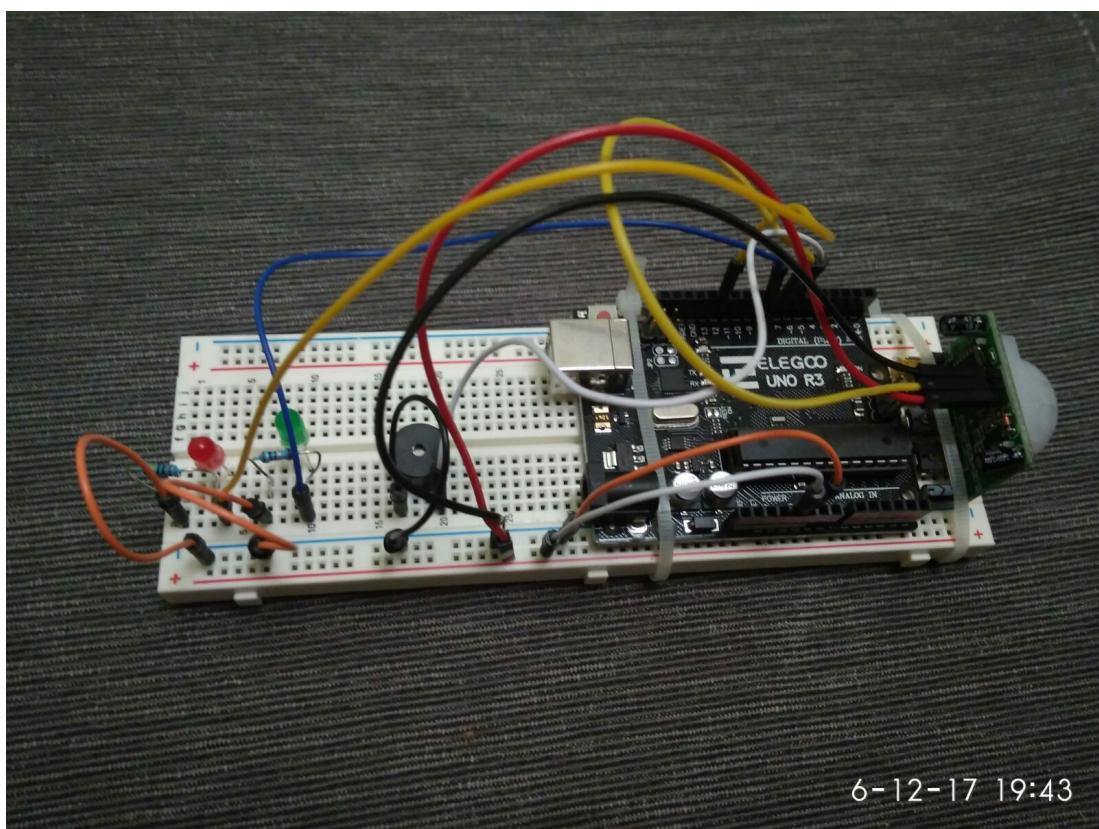
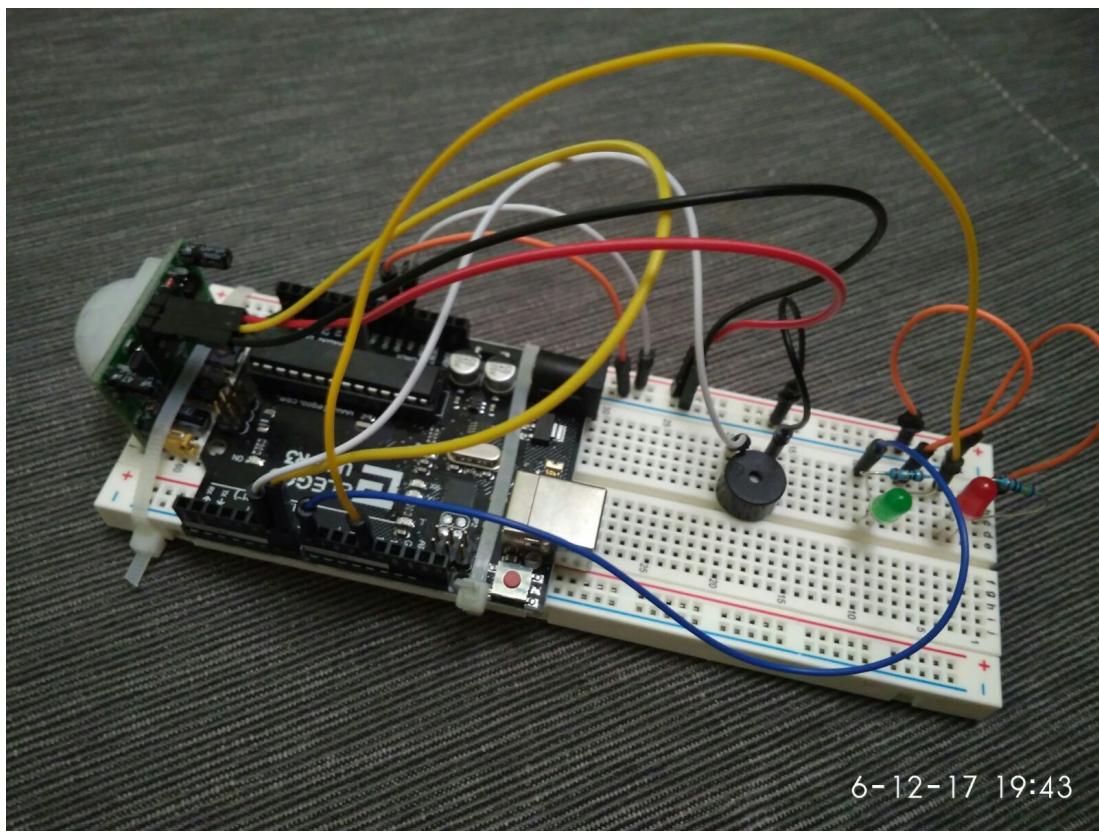
void setup()
{
    // Inicializamos el sensor y los leds
    Serial.begin(9600);

    pinMode(pirPin, INPUT_PULLUP);
    pinMode(ledPinRojo, OUTPUT);
    pinMode(ledPinVerde, OUTPUT);
}

void loop()
{
    // leemos los valores del sensor PIR
    int proximity = digitalRead(pirPin);
    delay(100);

    // Si se detecta movimiento, encendemos el led rojo
    // y el buzzer, y apagamos el verde
    if (proximity == HIGH)
    {
        tone(pinBuz, 350, 200);
        digitalWrite(ledPinRojo, HIGH);
        Serial.println("Motion detected!");
        digitalWrite(ledPinVerde, LOW);
    }
    else
    {
        // Si no se detecta movimiento, apagamos el led rojo
        // y encendemos el verde
        digitalWrite(ledPinRojo, LOW);
        digitalWrite(ledPinVerde, HIGH);
    }
}
```

4.2.6. Fotos del montaje



4.3. Práctica 3: Integración sensores PIR y ultrasonidos

4.3.1. Descripción de la práctica

El objetivo principal de esta práctica es integrar las dos prácticas anteriores. En lugar de utilizar los sensores de movimiento por separado, queremos hacer que cada uno de los sensores de movimiento tengan un led rojo asociado. Cuando un sensor detecta movimiento, se encenderá su propio led rojo, a modo de indicación para saber cuál es el sensor que está detectando movimiento. Por ello, tenemos 3 sensores y 3 leds rojos. Finalmente, dispondremos de un led verde, que se encenderá si ninguno de los tres sensores están detectando movimiento.

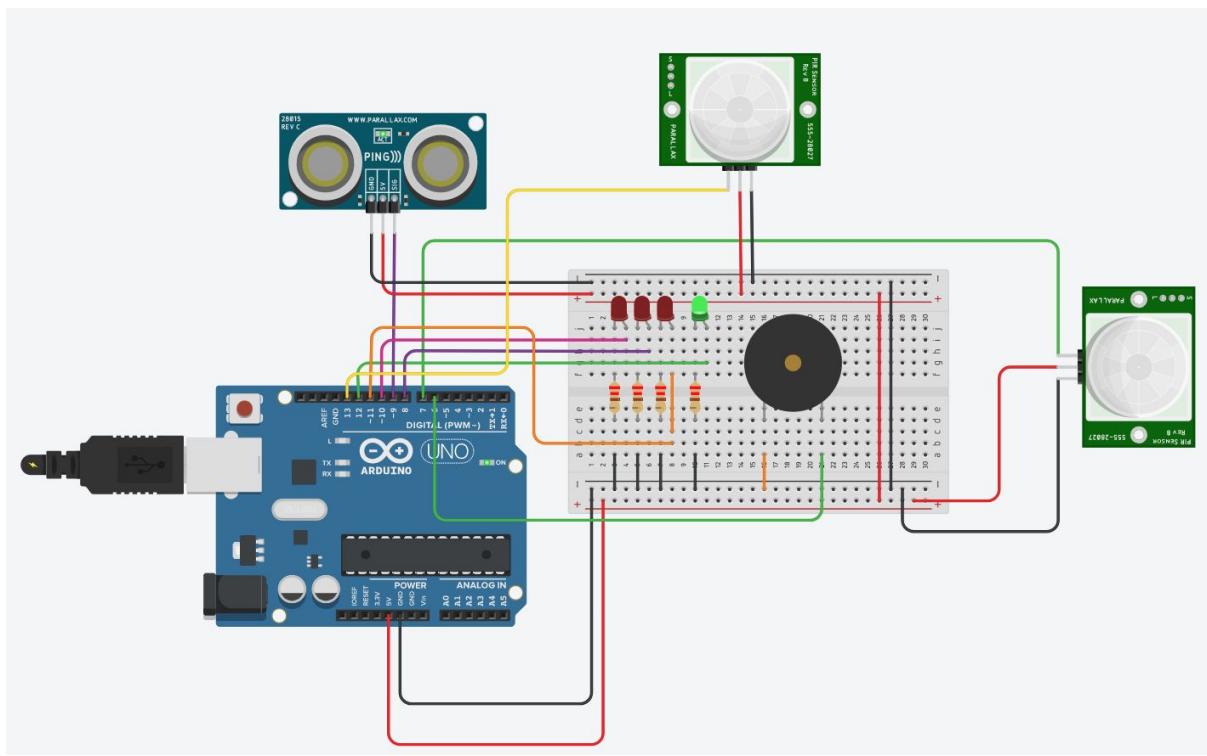
Esta práctica la hacemos con el objetivo de que luego en nuestro proyecto final seamos capaces de diferenciar en qué habitación de la casa están los ladrones.

4.3.2. Material utilizado

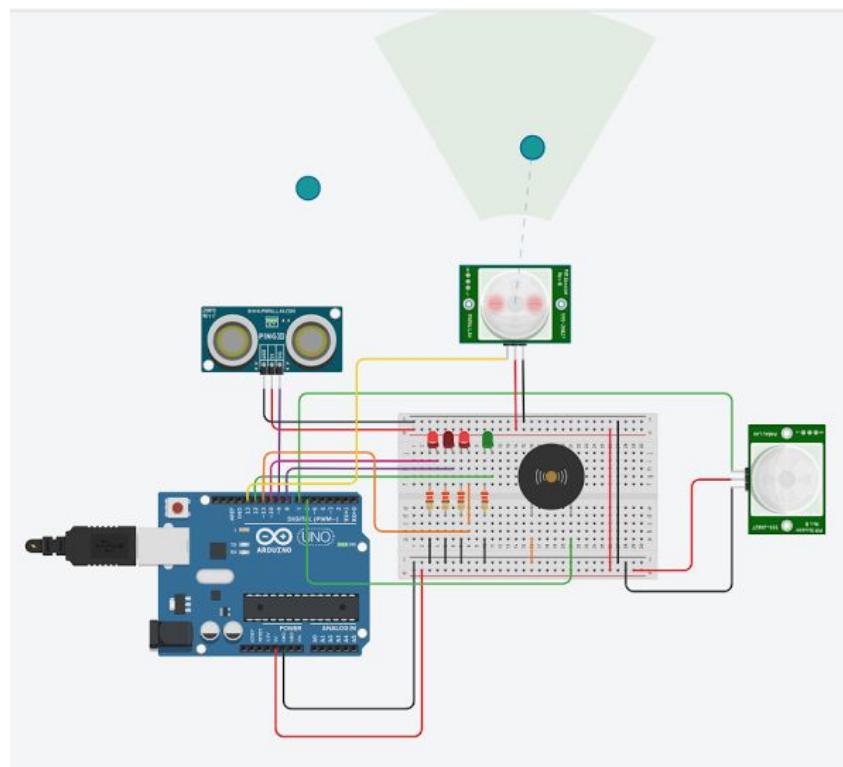
- 1 Arduino UNO
- Placa breadboard
- 2 sensores PIR
- 1 sensor de ultrasonidos
- 3 LEDs rojos
- 1 LED verde
- Buzzer
- 4 resistencias de 220Ω

4.3.3. Simulación

En este apartado podemos ver la simulación de la práctica en el programa Tinkercad. La primera imagen muestra la simulación con todos los sensores desactivados salvo el LED verde, que indica que no se están detectando intrusos en el sistema:



La siguiente imagen muestra la simulación cuando dos de los sensores están detectando intrusos. Como podemos ver, dos de los leds rojos están activados a modo indicativo:



NOTA IMPORTANTE: Una explicación completa del proyecto simulado en Tinkercad puede verse en <https://youtu.be/vTNzJ4K9xDk>

4.3.4. Código comentado

```
// Declaracion de variables auxiliares          1
int ledPinRojo = 8; // LEDs rojos            2
int ledPinRojo2 = 10;                         3
int ledPinRojo3 = 11;                         4
int ledPinVerde = 12; // LED verde           5
int pirPin = 7; // PIR                      6
int pirPin2 = 13;                           7
int pinBuz = 6; // BUZZ                     8
int ultrasonido = 9;                        9
// Fin de declaracion de variables auxiliares 10

void setup()                                11
{
    Serial.begin(9600);                      12
    // The PIR sensor's output signal is an open-collector, 13
    // so a pull-up resistor is required:                  14
    pinMode(pirPin, INPUT_PULLUP);             15
    pinMode(pirPin2, INPUT_PULLUP);            16
    // Inicializamos el sensor ultrasonidos           17
    pinMode(ultrasonido, INPUT);              18
    // Inicializamos los leds                         19
    pinMode(ledPinRojo, OUTPUT);               20
    pinMode(ledPinRojo2, OUTPUT);              21
    pinMode(ledPinRojo3, OUTPUT);              22
    pinMode(ledPinVerde, OUTPUT);              23
}
                                                24

// funcion para leer el sensor de ultrasonidos del pin 25
// que le indiquemos por parametro
long readUltrasonicDistance(int pin)        26
{
    pinMode(pin, OUTPUT); // Clear the trigger      27
    digitalWrite(pin, LOW);                         28
    delayMicroseconds(2);                         29
    // Sets the pin on HIGH state for 10 micro seconds 30
    digitalWrite(pin, HIGH);                       31
    delayMicroseconds(10);                        32
    digitalWrite(pin, LOW);                        33
    pinMode(pin, INPUT);                         34
    // Reads the pin, and returns the sound wave travel time in 35
    // microseconds
    return pulseIn(pin, HIGH);                   36
}

bool intrusos = false; // lo utilizamos para marcar el estado 37
```

```

void loop()
{
    bool desactivando = false; // utilizamos el valor
        desactivando como valor intermedio
    // hasta que no nos informen los dos sensores de que
        realmente no estan detectando
    // a nadie no podemos encender el led verde como que todo
        esta en orden. Tenemos
    // que esperar a que ambos comprueben que no hay nadie para
        encender el verde

    int proximity = digitalRead(pirPin); // proximidad del
        sensor pir 1
    delay(100);

    int proximity2 = digitalRead(pirPin2); // proximidad del
        sensor pir 2
    delay(100);

    int cm = 0.01723 * readUltrasonicDistance(ultrasonido); //
        calculamos la distancia
    // en cm al sensor de ultrasonidos
    Serial.println(cm);

    // Comprobamos si el sensor PIR 1 esta detectando algo
    if (proximity == HIGH)
    {
        tone(pinBuz, 350, 200);
        digitalWrite(ledPinRojo, HIGH);
        Serial.println("Motion detected!");
        digitalWrite(ledPinVerde, LOW);
        intrusos = true;
    }
    else
    {
        digitalWrite(ledPinRojo, LOW);
        desactivando = true;
    }

    // Comprobamos si el sensor PIR 2 esta detectando algo
    if (proximity2 == HIGH)
    {
        tone(pinBuz, 350, 200);
        digitalWrite(ledPinRojo3, HIGH);
    }
}

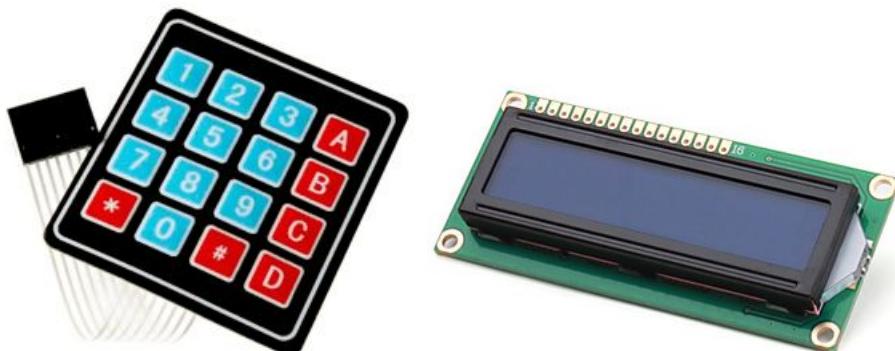
```

```
Serial.println("Motion detected!");
digitalWrite(ledPinVerde, LOW);
intrusos = true;
desactivando = false;
}
else
{
    digitalWrite(ledPinRojo3, LOW);
}

////// Comprobamos la distancia al ultrasonidos
// Si es menor de 2 metros, activamos la alerta roja!
if (cm < 200) {
    tone(pinBuz, 350, 200);
    digitalWrite(ledPinRojo2, HIGH);
    Serial.println("Motion detected!");
    digitalWrite(ledPinVerde, LOW);
    intrusos = true;
}
else
{
    digitalWrite(ledPinRojo2, LOW);
    if (desactivando == true) {
        intrusos = false;
    }
}

// Si no hemos detectado problemas en ningun sensor,
// activamos
// la luz verde que indica que no hay problemas
if (intrusos == false) {
    digitalWrite(ledPinVerde, HIGH);
}
}
```

4.4. Práctica 4: Pantalla LCD y teclado 4x4



4.4.1. Descripción de la práctica

El objetivo principal de esta práctica es aprender a utilizar periféricos de entrada y salida de datos. Para ello utilizaremos un teclado matricial como dispositivo de entrada, el cual permitirá la introducción de claves numéricas, y una pantalla LCD como dispositivo de salida, donde se mostrará al usuario una serie de instrucciones y los números que el usuario haya introducido.

Al inicio, tendremos programada una clave numérica, un led rojo encendido, y se mostrará por la pantalla LCD un mensaje el cual indique al usuario que el sistema se encuentra a la espera de que introduzca una clave. Cuando el usuario introduzca una clave numérica, la cual se mostrará por la pantalla, podrán ocurrir 2 estados:

- 1) Clave numérica incorrecta: Se mostrará un mensaje por la pantalla LCD que indique dicho estado y un buzzer producirá un sonido distintivo para este estado.
- 2) Clave numérica correcta: Se mostrará un mensaje por la pantalla LCD que indique dicho estado, se apagará el led rojo y se encenderá el led verde, y un buzzer producirá un sonido distintivo para este estado.

Esta práctica la hacemos con el objetivo de que luego en nuestro proyecto final seamos capaces de interactuar con el sistema de control de la alarma.

4.4.2. Material utilizado

- 1 Arduino UNO
- 1 Placa breadboard
- 1 Keypad 4x4
- 1 LCD 16x2
- 1 LED rojo
- 1 LED verde
- 1 Buzzer
- 1 Potenciómetro 250kΩ
- 4 Resistencias de 220Ω

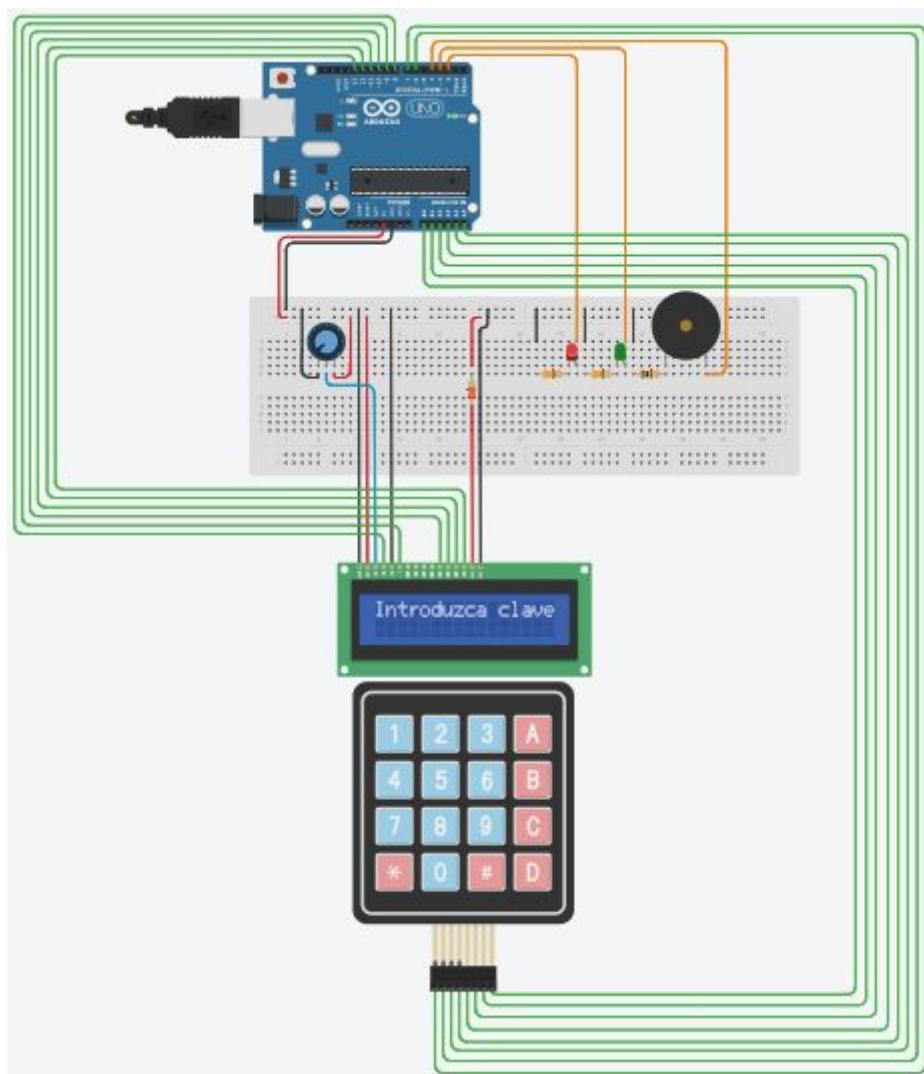
4.4.3. Descripción del Componente

Un teclado matricial es un dispositivo que agrupa los pulsadores en filas y columnas, formando una matriz, y permite controlarlos empleando un número de conductores inferior al que necesitaríamos al usarlos de forma individual.

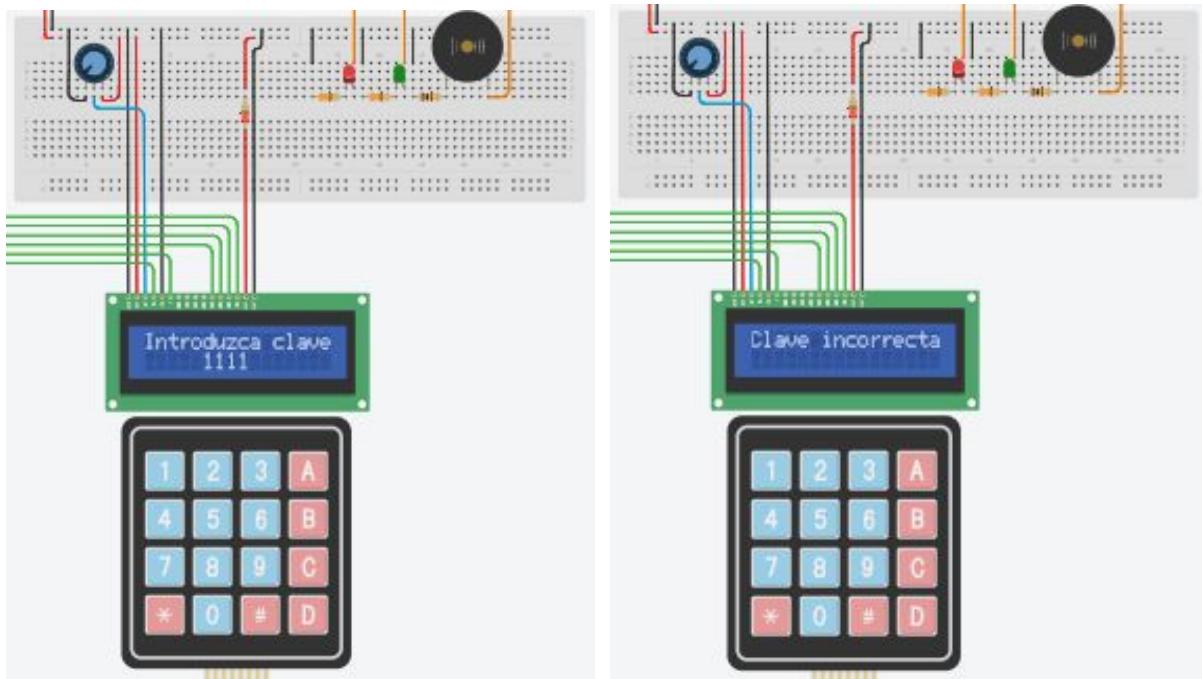
La pantalla LCD (Liquid Crystal Display) muestra información en una pantalla gracias a la iluminación del fondo. Para ajustar dicha iluminación suele ser necesario el uso de un potenciómetro que permite regular la resistencia entre la placa Arduino y el LCD. La pantalla que utilizaremos en esta práctica es de 16x2, es decir, tiene 2 filas de 16 caracteres cada una.

4.4.4. Simulación

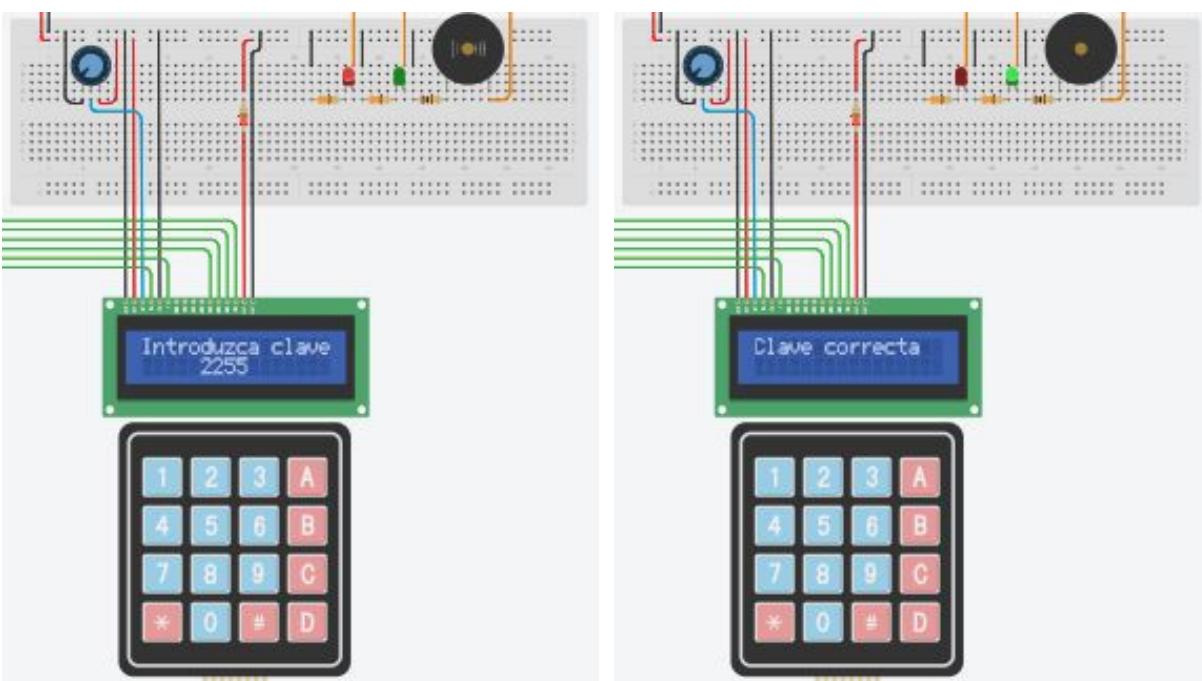
En este apartado podemos ver la simulación de la práctica en el programa Tinkercad. La primera imagen muestra la simulación del sistema a la espera de que el usuario introduzca una clave:



Estado 1 (Clave numérica incorrecta):



Estado 2 (Clave numérica correcta):



4.4.5. Código comentado

```
// Librerias necesarias
#include <LiquidCrystal.h>
#include <Keypad.h>

/*-----KEYPAD-----*/
const byte numRows= 4; // Numero de filas
const byte numCols= 4; // Numero de columnas
char keypressed;
char keymap[numRows][numCols]=
{
{'1', '2', '3', 'A'},
{'4', '5', '6', 'B'},
{'7', '8', '9', 'C'},
{'*', '0', '#', 'D'},
};

// Conexiones entre el teclado y los terminales de arduino
byte rowPins[numRows] = {7,6,A5,A4}; // Filas de 0 a 3
byte colPins[numCols] = {A3,A2,A1,A0}; // Columnas de 0 a 3

Keypad myKeypad= Keypad(makeKeymap(keymap), rowPins, colPins,
    numRows, numCols);

/*-----FIN KEYPAD-----*/

/*-----LCD-----*/
LiquidCrystal lcd(8, 9, 10, 11, 12, 13); // Pines (rs, en, d4
    , d5, d6, d7)

// Resolucion de la pantalla
int screenWidth = 16;
int screenHeight = 2;

/*-----FIN LCD-----*/

char codigoSecreto[4] = {'2','2','5','5'}; // Clave numerica

// Variables de comparacion entre clave introducida y
// programada
int posicion=0;
int clave=0;

int cursor=5; // Posicion de introduccion de datos en el LCD
```

```
// Declaracion de variables auxiliares          43
int ledRojo=2;                                44
int ledVerde=3;                                45
int buzzer=4;                                 46
                                                47
void setup()                                     48
{
// Inicializamos los leds y el buzzer          49
pinMode (ledRojo, OUTPUT);                      50
pinMode (ledVerde, OUTPUT);                     51
pinMode (buzzer, OUTPUT);                      52
                                                53
digitalWrite(ledRojo,HIGH); // Encendemos el LED rojo   54
digitalWrite(ledVerde, LOW); // Apagamos el LED verde  55
                                                56
// Inicializamos el LCD                         57
lcd.begin(screenWidth, screenHeight);           58
lcd.clear(); // Borramos                         59
lcd.setCursor(0,0); // Situamos el cursor en la linea 1 60
lcd.print("Introduzca clave"); // Escribimos        61
lcd.setCursor(cursor,1); // Situamos el cursor en la linea 2 62
}

void loop()                                      63
{
    char pulsacion = myKeypad.getKey(); // Leemos la pulsacion 64
    if (pulsacion != 0) // Si hemos pulsado          65
    {
        if (pulsacion != '#' && pulsacion != '*' && clave==0) // 66
            Y no ha sido # (almohadilla) ni * (asterisco)
        {
            lcd.print(pulsacion); // Escribimos la pulsacion 67
            cursor++; // Incrementamos la posicion del      68
                           cursor

            // Tono de pulsacion
            tone(buzzer,350);
            delay(200);
            noTone(buzzer);

            if (pulsacion == codigoSecreto[posicion]) // Si      69
                la pulsacion se corresponde con la clave       70
            {
                posicion++; // Incrementamos la posicion a comparar 71
            }
        }
    }
}
```

```

}

if (posicion == 4) // Si todas las pulsaciones
    coinciden con la clave
{
    lcd.clear(); // Borramos
    lcd.setCursor(0,0); // Situamos el cursor en
        la linea 1
    lcd.print("Clave correcta"); // Escribimos

    // Tono de clave correcta
    delay(200);
    tone(buzzer,500);
    delay(100);
    noTone(buzzer);
    tone(buzzer,600);
    delay(100);
    noTone(buzzer);
    tone(buzzer,800);
    delay(100);
    noTone(buzzer);

    clave=1; // Indicador de clave correcta

    digitalWrite(ledRojo,LOW); // Apagamos el LED rojo
    digitalWrite(ledVerde,HIGH); // Encendemos el LED
        verde
}
if(cursor>8) //Si hemos pulsado 4 numeros
{
    // Inicializamos las variables
    cursor=5;
    posicion=0;

    if(clave==0) // Y la clave no es correcta
    {
        lcd.clear(); // Borramos
        lcd.setCursor(0,0); // Situamos el cursor en la
            linea 1
        lcd.print("Clave incorrecta"); // Escribimos

        // Tono de clave incorrecta
        tone(buzzer,70,500);
        delay(250);
        noTone(buzzer);
        delay(1500);
    }
}

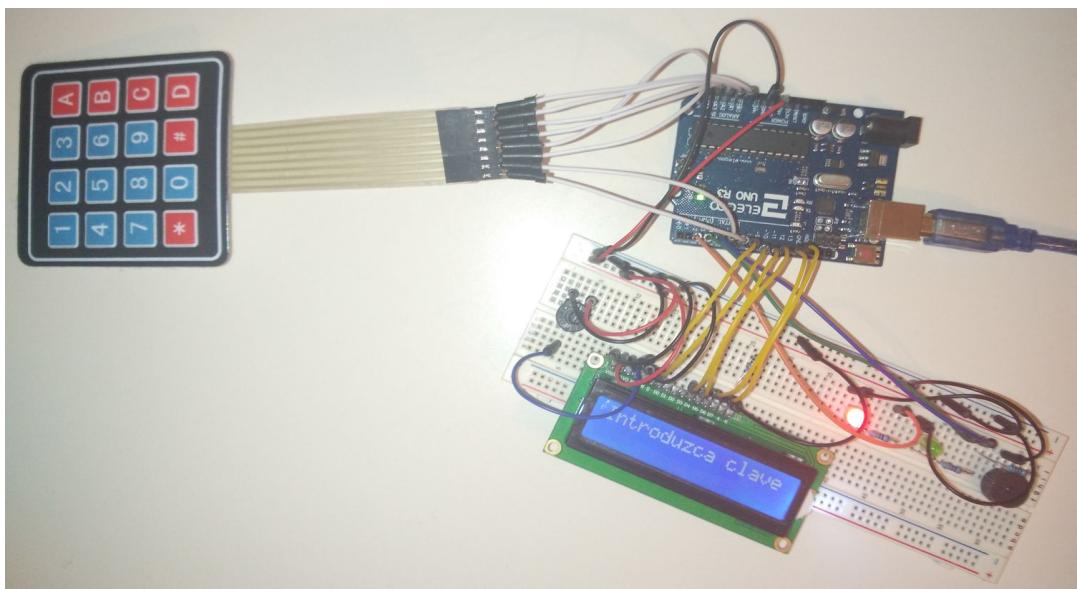
```

```

126     lcd.clear(); // Borramos
127     lcd.setCursor(0,0); // Situamos el cursor en la
128     linea 1
129     lcd.print("Introduzca clave"); // Escribimos
130     lcd.setCursor(5,1); // Situamos el cursor
131     en la linea 2
132   }
133 }
134
135 if (pulsacion == '*') // Si pulsamos * (asterisco)
136 {
137   // Inicializamos las variables
138   posicion = 0;
139   cursor = 5;
140   clave=0;
141   posicion=0;
142
143   lcd.clear(); // Borramos
144   lcd.setCursor(0,0); // Situamos el cursor en la linea 1
145   lcd.print("Introduzca clave"); // Escribimos
146   lcd.setCursor(5,1); // Situamos el cursor en la linea 2
147
148   digitalWrite(ledRojo,HIGH); // Encendemos el LED rojo
149   digitalWrite(ledVerde, LOW); // Apagamos el LED verde
150 }
151
}

```

4.4.6. Fotos del montaje



4.5. Práctica 5: Conexión entre los dos arduinos

4.5.1. Introducción

El motivo fundamental por el cual utilizamos dos arduinos parte de la necesidad de utilizar más pines para nuestro proyecto sin hacer cosas excesivamente complicadas.

Aunque la explicación completa de esto sea algo complicado, vamos a intentar reducirlo para que se entienda la forma de usarlo.

La idea básica es que exista un **protocolo de comunicación entre los dos arduinos** para que sean capaces de “entender el idioma del otro arduino”. ¿Qué significa esto?

1. **Cada arduino tiene que tener su propio código fuente** (archivo .ino distinto). Por tanto, habrá que tener dos instancias de Arduino IDE abiertas y cargar en cada Arduino su código correspondiente.

Nota: cada vez que se añade un nuevo Arduino en Tinkercad, aparece una nueva pestaña desplegable que te permite ver el código asociado al Arduino que acabamos de añadir.

2. El protocolo de comunicación implica que se va a decidir un “idioma” que entenderán los dos arduinos. Este idioma será como un **código** entre ellos para comunicar qué cosas están pasando en su placa. Por ejemplo, puede tener sentido que si tenemos conectado un sensor de ultrasonidos al Arduino número 1, quizás queramos comunicarle a un led que está conectado al Arduino número 2 que tiene que encenderse si se detecta presencia a menos de 1 metro. Se detalla la manera un poco más adelante.

Pasamos ahora a detallar las conexiones que habrá que realizar entre los dos arduinos para que todo lo anterior sea factible.

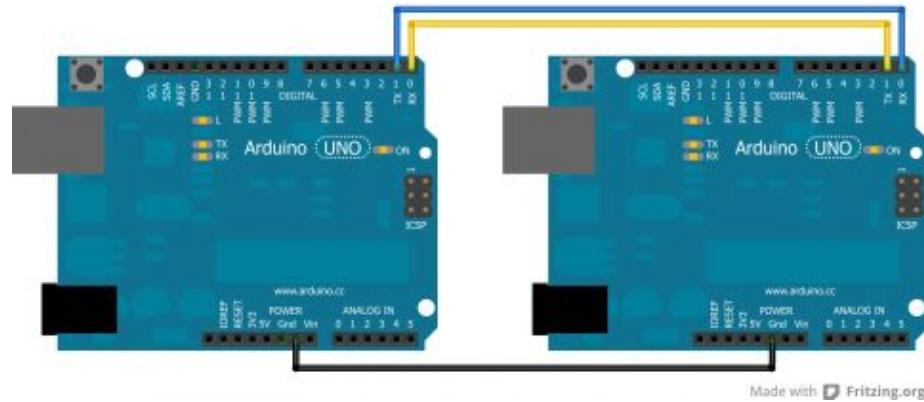
4.5.2. Detalles de la conexión entre los dos arduinos

Nosotros utilizamos la información de:

<http://robotic-controls.com/learn/arduino/arduino-arduino-serial-communication>

para aprender cómo comunicar nuestros arduinos entre ellos.

Suponemos que nos encontramos en la situación descrita en el punto 2 de la introducción del apartado anterior, donde queremos comunicar un arduino que tiene conectado un sensor de ultrasonidos con un arduino que tiene conectado un led, con la idea de que el led se encienda si se está detectando la presencia de alguien. Para ello, las conexiones que tenemos que realizar son las siguientes:



Es decir, tenemos que conectar en serie el pin tx del arduino 1 al pin rx del arduino 2, y el pin tx del arduino 2 al pin rx del arduino 1.

Adicionalmente, es **MUY importante** que tengan en común un **ground**, tal y como se muestra en la figura. De lo contrario, el proyecto no funcionará.

4.5.3. Ejemplo de uso

Mostramos aquí un ejemplo que ilustra cómo estamos utilizando esto en nuestro proyecto:

Arduino 1 (envía información):

```

int rx = 0;
int tx = 1;
void setup() {
  Serial.begin (112500);
  //initialize UART pins
  pinMode (rx, OUTPUT);
  pinMode (tx, INPUT);
}
void loop() {
  byte dataR = 0; // mandamos un 0 por el puerto serie al
    arduino receptor de informaci n
  Serial.print(dataR);
}
  
```

Arduino 2 (recibe información):

```
// receiver arduino
int rx = 0;
int tx = 1;
void setup() {
    Serial.begin (112500);
    //initialize UART pins
    pinMode (rx, OUTPUT);
    pinMode (tx, INPUT);
}
void loop() {
    if(Serial.available() > 0)
    {
        val = Serial.read(); //read the next byte
        if (val == '0') {
            // haz algo
        }
    }
}
```

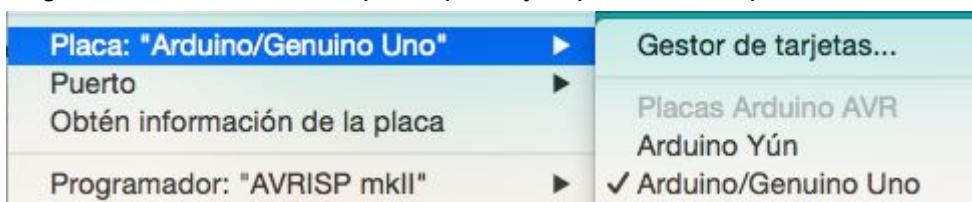
4.5.4. Guía para descargar el código en cada uno de los arduinos

Este apartado es especialmente importante, ya que nosotros tuvimos muchos problemas para conseguir cargar el código en los dos arduinos de forma correcta.

Los pasos a seguir son los siguientes:

Paso previo MUY importante: desconectar los cables que unen los pines tx y rx del Arduino. Si están conectados entre sí, el programa nos dará un error y no nos permitirá cargar el código en el Arduino. Una vez desconectados, podemos continuar:

1. Conectar el primero de los arduinos al puerto USB del ordenador.
2. Abrir Arduino IDE con el código del arduino correspondiente.
3. Escoger en Herramientas el tipo de placa y el puerto correspondiente a Arduino:



4. Pinchar en el botón de subir a la placa, que está en la parte superior de la pantalla:

5. Si hay algún error en el código, revisarlo y modificarlo para que el código sea correcto.
6. Si no hay ningún error, ya podemos desconectar nuestro Arduino del ordenador.
7. Conectar el segundo de los Arduinos. Si lo conectamos en el mismo puerto, no será necesario volver a repetir el punto 3. Esto es más fácil de hacer así que si conectamos

los dos arduinos a la vez, ya que al escoger el puerto sólo se puede escoger uno, y posiblemente no estaremos seguros de cuál es cuál.

8. Abrir en Arduino IDE el código correspondiente al segundo de los arduinos.
9. Volver a repetir los puntos 3 y 4 si fuera necesario.
10. Repetir los puntos 5 y 6.
11. Finalmente, volver a conectar correctamente los pines tx y rx de cada uno de los arduinos entre ellos.
12. Alimentar el circuito y ¡probar el resultado!

5. Proyecto final. Sistema de detección de intrusos

5.1. Contexto

Este conjunto de prácticas están pensadas para llevarse a cabo en la asignatura de TPR. Proyectos tecnológicos de cuarto de la E.S.O. La asignatura está diseñada para su oferta como optativa, y se rige por la ORDEN 2160/2016, de 29 de junio, de la Consejería de Educación, Juventud y Deporte, por la que se aprueban materias de libre configuración autonómica en la Comunidad de Madrid.

En particular, la asignatura de **Proyectos tecnológicos** tiene divididos sus contenidos en cuatro bloques: electromecánica, electrónica industrial, microcontroladores y automatismos. Los contenidos y actividades que se describen en esta memoria forman parte del bloque de microcontroladores (Arduino).

Los **objetivos** concretos de lo que sería la unidad didáctica relativa a Arduino son los siguientes:

- El alumno será capaz de comprender el funcionamiento básico de Arduino UNO.
- El alumno aprenderá a utilizar diferentes sensores para detectar información del entorno.
- El alumno podrá controlar los leds y un buzzer como indicadores de lo que está pasando en el circuito.
- El alumno comprenderá cómo las diferentes piezas interactúan entre ellas a través del código.
- El alumno será capaz de utilizar la pantalla de debug de Tinkercad y del programa Arduino UNO (serial monitor) para mostrar información sobre lo que está pasando en el circuito.

Ahora que ya entendemos el contexto del temario que hay que impartir, ¿por qué elegimos este proyecto en concreto y no otro? Existen varias razones:

- Auge de la domótica y el uso de microcontroladores.
- Las aplicaciones para dispositivos conectados a internet son amplias (IOT o Internet de las cosas).
- Una alarma es una aplicación real que llama la atención a los niños. Cuanta más utilidad vean en el trabajo que están realizando, más fácil será conseguir motivarlos e implicarlos en el proyecto que se está realizando.
- En tercero y cuarto de la ESO se enseña a usar actuadores y sensores con arduino. Un proyecto que es capaz de englobarlas todas ellas podría verse al final de cuarto de la ESO a modo de conclusión. Este proyecto podrían llevarlo a cabo en grupos de 4 alumnos, aunque habría que modificarlo ligeramente para que no haya que usar la pantalla LCD ni el teclado matricial, que son elementos que no se estudian en la ESO.

5.2. Diagramas de flujo

A continuación se muestran 4 diagramas de flujo de las 4 partes principales de la práctica: el bucle principal, el diagrama de detección cuando la alarma está activada, el diagrama de detección cuando la alarma está activada parcialmente (media alarma) y el diagrama correspondiente al cambio de contraseña.

Diagrama del bucle principal:

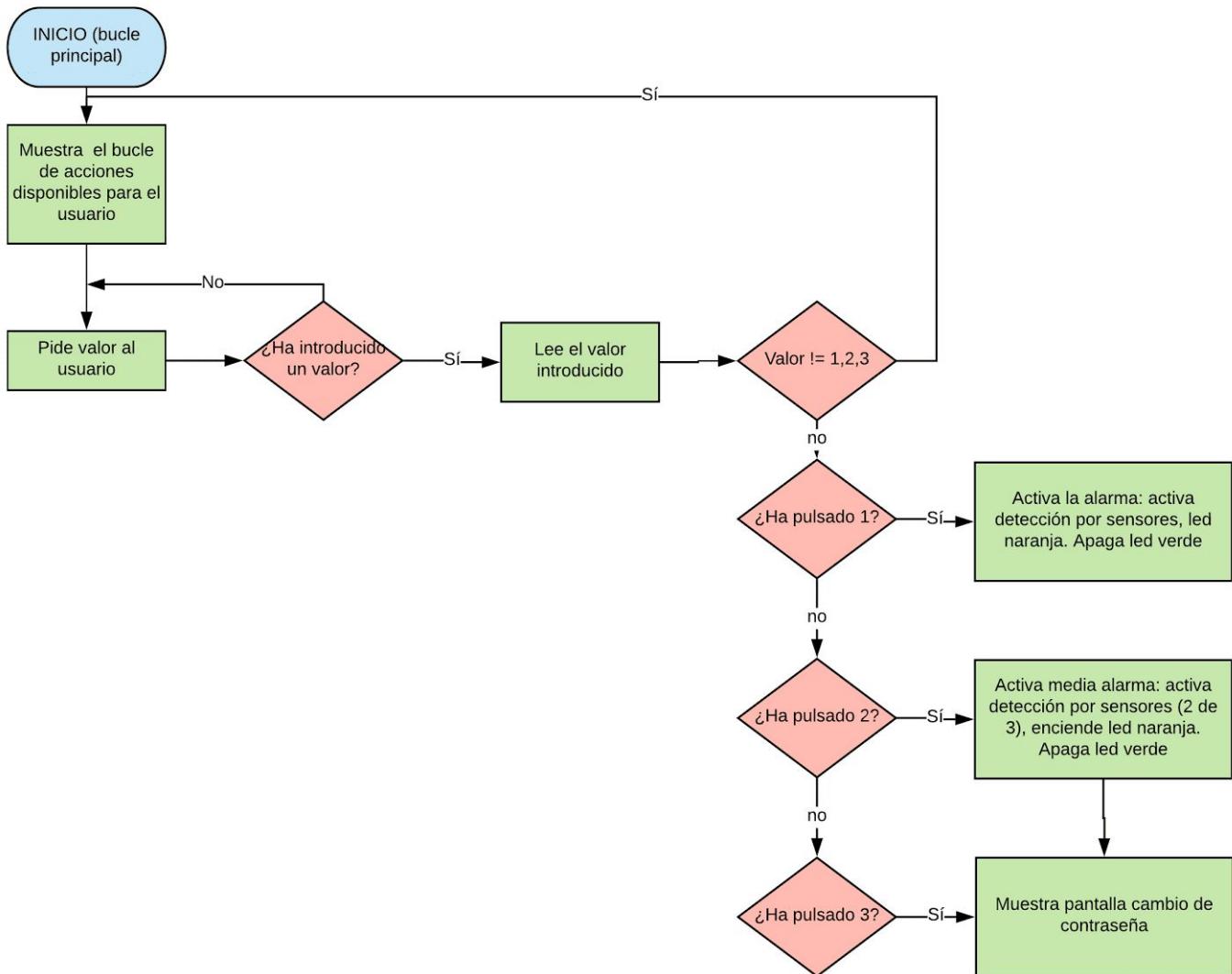


Diagrama de detección con alarma activada:

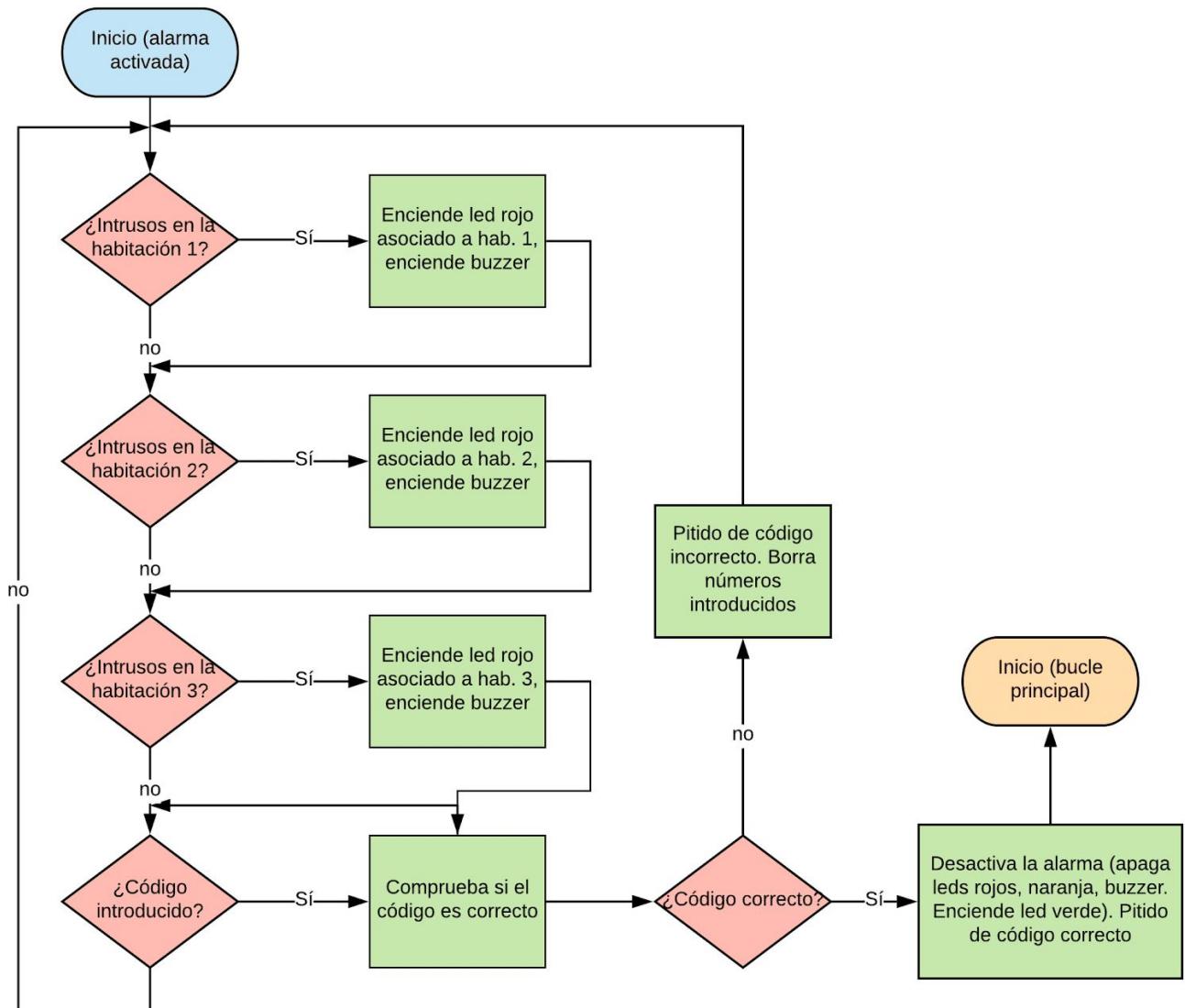


Diagrama de detección con alarma parcialmente activada:

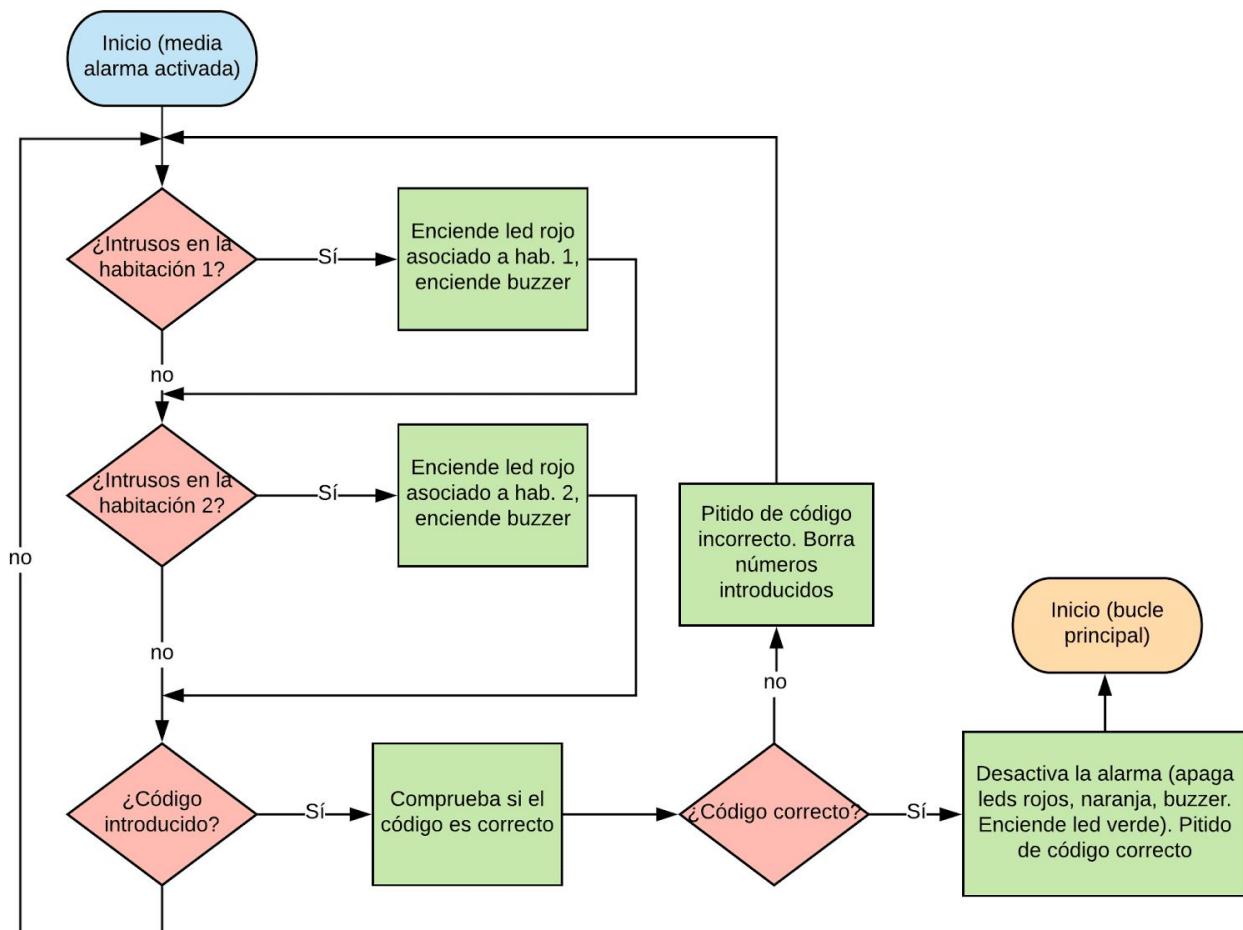
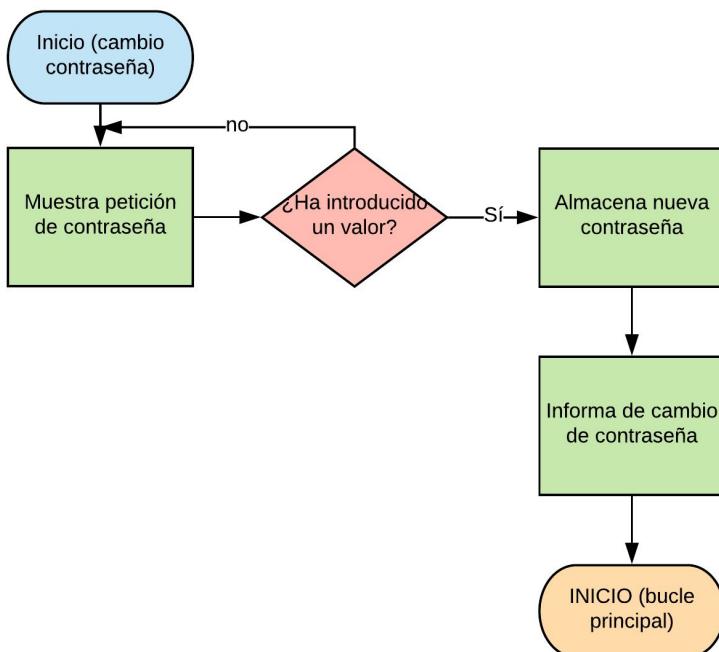


Diagrama de cambio de contraseña:



5.3. Construcción de la maqueta

Se diseña una vivienda unifamiliar de única planta, situada en parcela. Se considera un espacio reservado para la implantación de Arduino dentro de la vivienda, este espacio está ubicado en la zona de garaje o cuarto de calderas. Desde allí se distribuirá el cableado que suministra energía al resto de la vivienda.

5.3.1. Medidas

Escala utilizada: 1:250 cm

Perímetro total de la maqueta: 30 x 40 cm

Perímetro Total de la vivienda: 25 x 30 cm

Altura de vivienda: 10 cm

Grosor de las paredes: 0,50 cm (Grosor del Pliego)

Dimensiones de habitaciones:

- Salón: 16 x 16 cm
- Dormitorio: 9 x 20,5 cm
- Cocina: 16 x 14 cm
- Baño: 9 x 9,5 cm
- Garaje: 15 x 30 cm

5.3.2. Descripción de la Práctica

PASO 1 : Dibujo de Croquis de Planta

Material necesario:

- Un pliego de cartón pluma
- Un lapicero
- Una goma de borrar
- Un escalímetro

Proceso:

1. Sobre el pliego de cartón pluma, se dibuja el croquis en planta de la vivienda, dejando un margen alrededor de toda la vivienda, espacio que facilita el proceso de pegado.
2. Dibujado el perímetro de la vivienda, se pasa a distribuir el espacio interior, dividiéndolo en habitaciones
3. No olvidar tener en cuenta, el grosor de los tabiques y el muro perimetral.

PASO 2: Dibujo de Croquis de Paredes

Material necesario:

- Un pliego de cartón pluma
- Un lapicero
- Una goma de borrar
- Un escalímetro

Proceso:

1. En el otro pliego de cartón pluma, se diseñan las paredes y tabiques, donde se pondrán los huecos para ventanas y puertas.

PASO 3 : Corte de Piezas

Material necesario:

- Segundo pliego, donde se han diseñado las paredes y cutter.

Proceso:

1. Siguiendo el diseño de las paredes (dimensiones, puertas, ventanas, perforaciones para leds, aberturas para introducir la pantalla Lcd, etc)
2. Se procede al corte de las piezas.

PASO 4: Separación de ambientes.

Material necesario:

- Imágenes impresas
- Ordenador (Programa de Diseño)
- Tijeras de papel y cualquier otro material o manualidad

Proceso:

1. Con motivo de facilitar el proceso de maqueta, se ha optado por simplificar el diseño sin necesidad de utilizar o incorporar miniaturas. En este caso, se han buscado imágenes que describen las diferentes habitaciones de una casa. Aquí el proyecto se podría ampliar todo lo que las manualidades y la imaginación del alumno pueda llegar.
2. Seleccionadas las imágenes, se adaptan a la escala de la planta de vivienda (ver PASO 1), se imprimen y se recortan.

PASO 5: Proceso de Pegado

Material necesario:

- Pliego con el diseño en planta de la vivienda
- Imágenes impresas
- Cola vinílica
- Pegamento especial
- Cinta aislante

Proceso:

1. Se pegan las imágenes al espacio reservado a ello en el diseño en planta.
2. Una vez ambientado el espacio en habitaciones, se pegan las paredes de la vivienda con el pegamento especial.
3. Una vez secado el pegamento especial, se reforzará con cinta aislante aprovechando esta para corregir imperfecciones del corte de piezas.

PASO 6: Instalación de Arduino

Material necesario:

- La maqueta una vez terminada
- Los elementos del Kit Arduino mencionados en las prácticas anteriores.

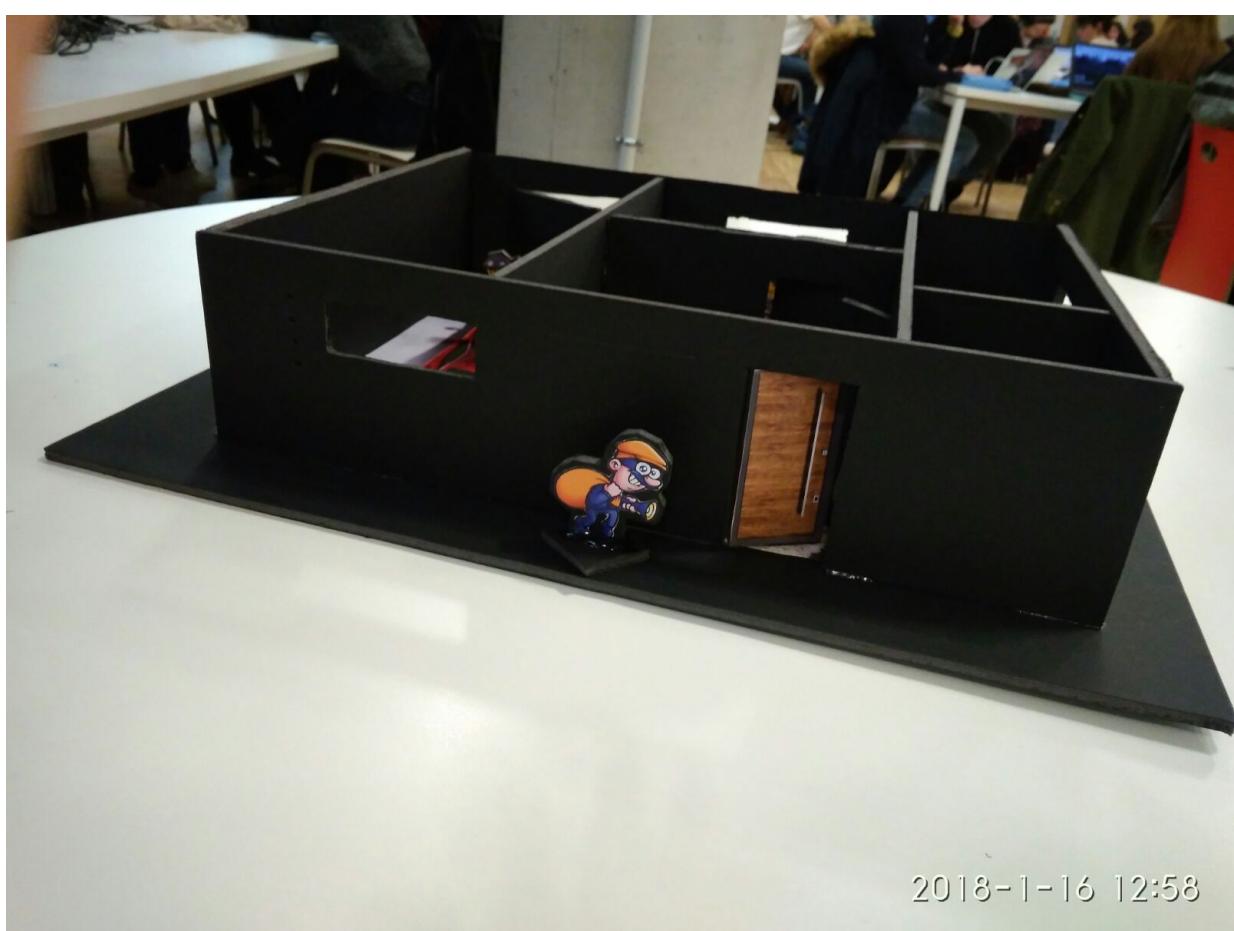
Proceso:

1. Se instalan dos placas Arduino y una Breadboard en el interior de la habitación destinada al garaje, y cuatro leds, un teclado y una pantalla LCD en la pared destinada a la fachada.

5.3.3. Fotos del montaje





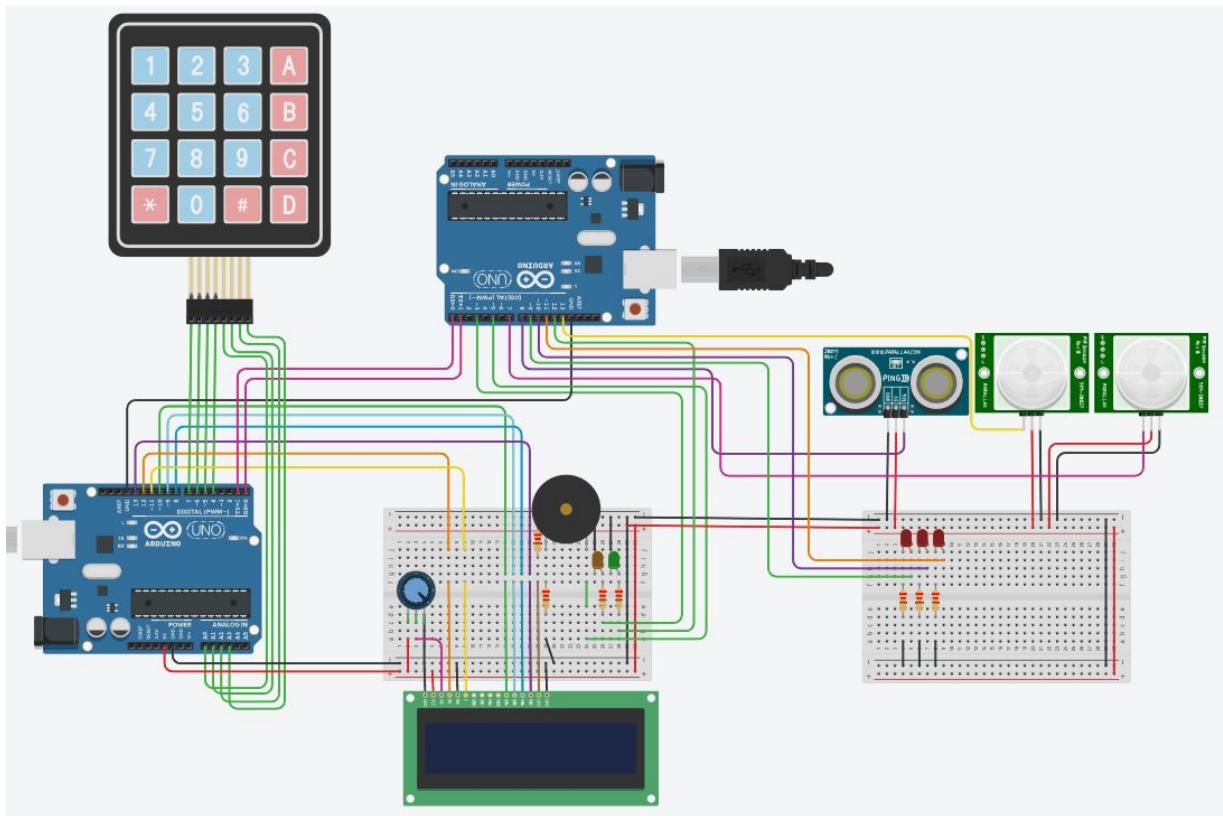


2018-1-16 12:58

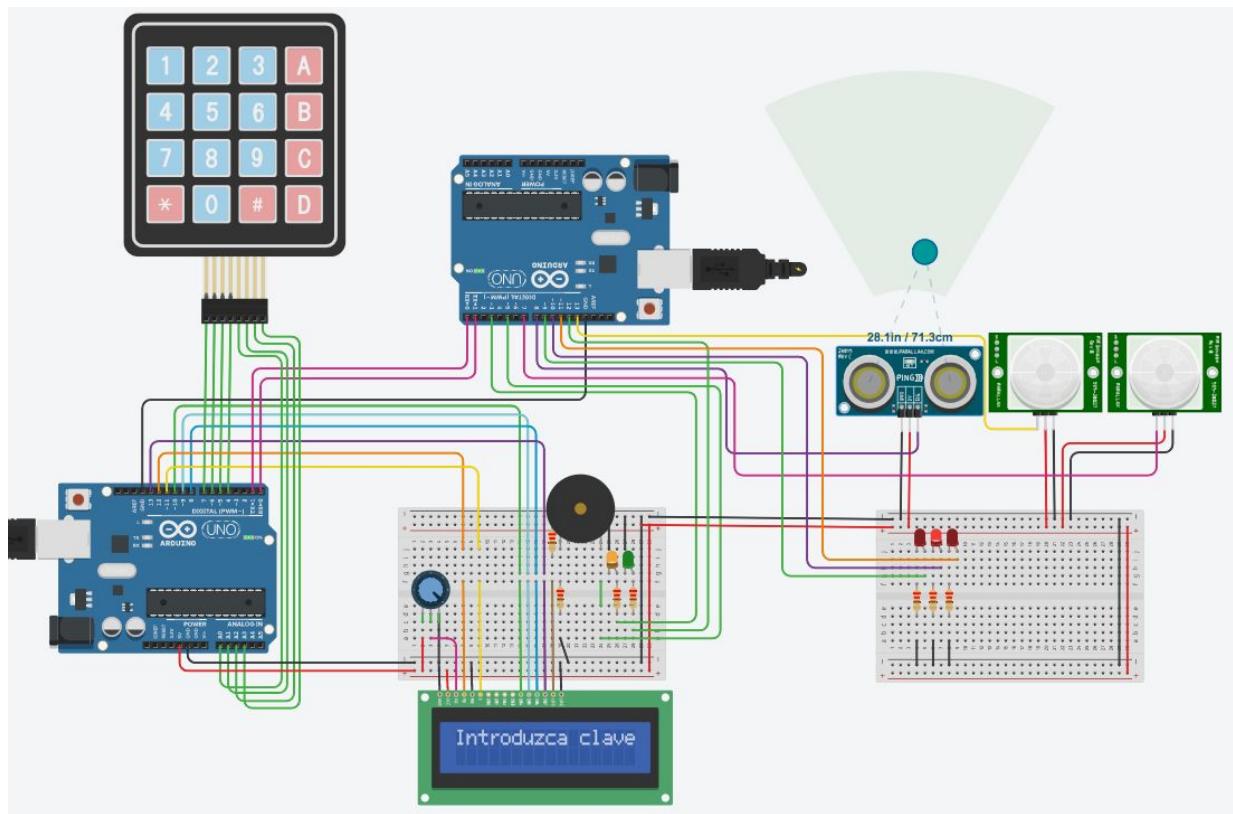


5.4. Simulación

A continuación mostramos las fotos del montaje completo en Tinkercad.
La primera foto muestra el montaje con el circuito apagado/desactivado:



En la siguiente imagen podemos ver que el led amarillo está encendido, ya que la alarma está activada, el sistema nos está pidiendo la contraseña y, además, se está detectando una presencia en la habitación “vigilada” por el sensor de ultrasonidos, por lo que el led rojo central está encendido también, alertándonos de los intrusos.

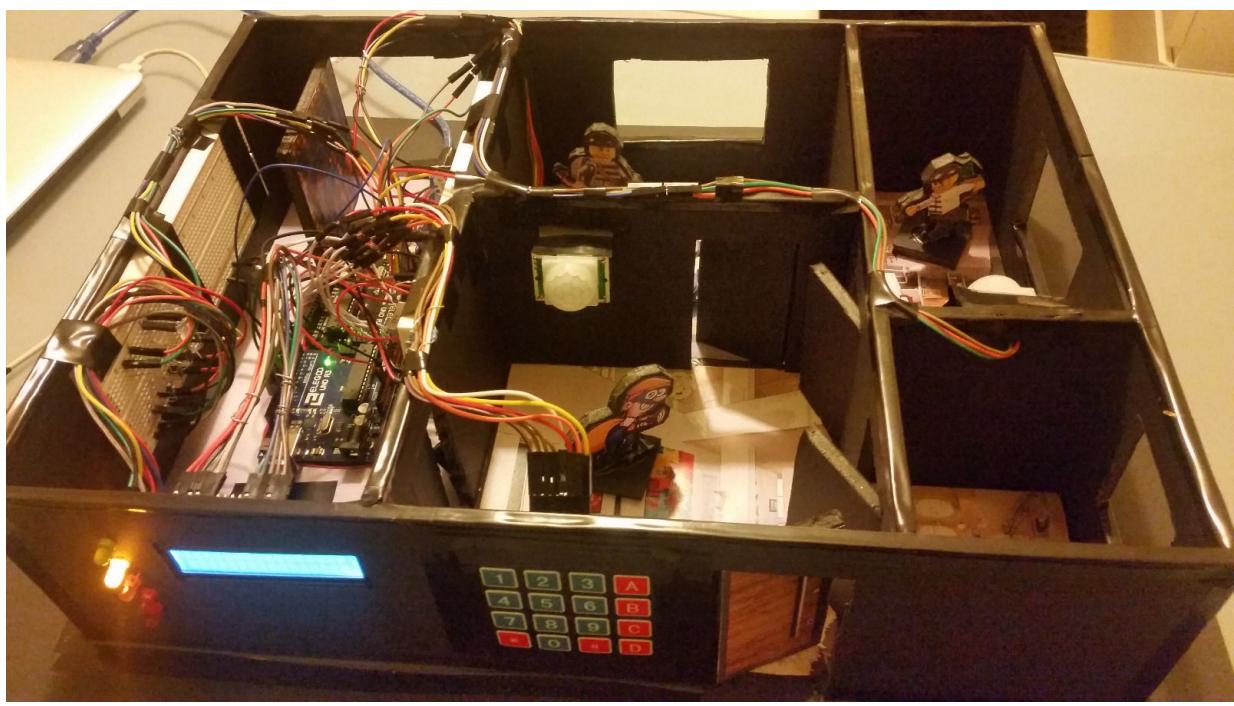


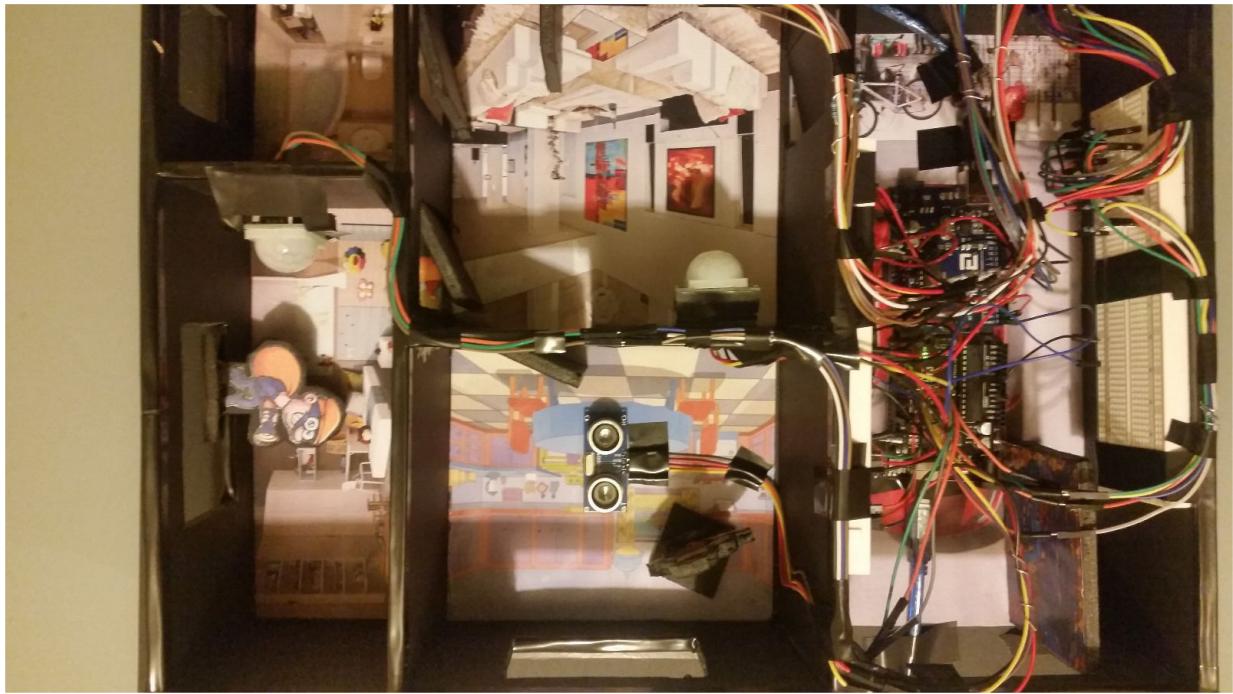
5.5. Código

El código correspondiente al arduino 1 y 2 del proyecto final se encuentra disponible en el **anexo 1**, debido a su gran longitud.

5.6. Resultado







5.7. Vídeo

Se han grabado dos vídeos. En primer lugar se grabó un vídeo con una maqueta básica para probar que la funcionalidad era correcta (<https://youtu.be/q3G0Ud6lFV8>). El vídeo realizado al acabar el proyecto, se encuentra en el siguiente enlace: <https://youtu.be/WVaGe2pM6V4>. La maqueta utilizada en este último vídeo ya está completa.

6. Coste

PRESUPUESTO

Nombre: TeprotectegemosdelMiedo (Empresa)
Dirección: C/ Maria Zambrano,
Teléfono: 653 912 590 / 91 234 56 78
E-mail: Nomasmiedo@ucm.es

Nombre: Guillermo Arduino (Cliente)
Dirección: C/ Didactica Informatica, 34
Teléfono: 634 567 890 / 91 234 56 78
E-mail: arduineandoESO@ucm.es

Nº	CONCEPTO	CANTIDAD	UNIDAD	€	TOTAL €
1	MATERIAL:				
	Kit Arduino Avanzado	1	Ud	49,99	49.99
	Pliego Cartón Pluma	2	Ud	5.45	10.90
	Bote Pegamento Especial	1	Ud	3.80	3.80
	Cola Vinilica	1	Ud	3.30	3.30
	Lapicero HB	1	Ud	0.45	0.45
	Goma de borrar	1	Ud	0.20	0.20
	Tijera	1	Ud	3.05	3.05
	Cutter	1	Ud	2.28	2.28
	Escalímetro	1	Ud	8.00	8.00
	Cinta autoadhesiva	1	Ud	0.50	0.50
	Cableado	30	ud	0.06	2.00
	Placa UNO R3+ Cable USB	1	ud	4.00	4.00
				TOTAL	88.47
2	OTROS CONCEPTOS:				
	Impresión Imágenes Color	1	Ud	0.50	0.50
	Mano de Obra	4	Ud	0.00	0.00
				TOTAL	0.50
				TOTAL	88.97

7. Viabilidad y escalabilidad

Los modelos de sistemas de detección de intrusos en el mercado son cada vez más abundantes, y su pretensión de abarcar varios aspectos de la domótica con el fin de hacer más fácil y segura la vida a los usuarios se encuentra en continua expansión.

Por ello, la adición de nuevas funciones al sistema es completamente factible. La dirección que parece tomar este área es la de diseñar sistemas centralizados que permitan al usuario controlar y automatizar numerosas facciones del hogar, tales como:

- Control de persianas y ventanas
- Automatizar el alumbrado
- Automatizar la calefacción
- Sistema contra incendios
- Automatización de electrodomésticos
- etc.

Por ello, podemos afirmar sin ninguna duda que la domótica, y el proyecto presente en este documento, es escalable. Es posible empezar con cuatro sensores y luego empezar a construir nuestra casa inteligente desde ahí. Las opciones de automatizar una vivienda son muy variadas.

La domótica doméstica cuenta también con un gran atractivo a su favor: el ahorro energético. La eficiencia energética es una de las grandes metas de la tecnología, y el uso optimizado de su consumición en el hogar se hace posible gracias a la domótica. Esto significa también para el usuario una reducción en el pago de su tarifa eléctrica y de gas.

8. Pruebas realizadas

Se han realizado múltiples pruebas para comprobar el funcionamiento del proyecto. Se listan a continuación:

Funcionamiento	Sí	No	¿Es correcto?
Se muestra el bucle de acciones acompañado por números y se permite elegir al usuario qué acción quiere realizar.	x		Sí
La opción 1 activa la detección en todos las habitaciones (alarma completa)	x		Sí
La opción 2 activa la detección parcial (media alarma)	x		Sí
La opción 3 permite cambiar la contraseña del sistema	x		Sí
La opción 4 permite poner la alarma por habitaciones		x	No. Sin implementar
La comunicación entre los dos arduinos funciona correctamente	x		Sí
Si la alarma está desactivada, ninguno de los leds rojos está encendido y el buzzer está desactivado	x		Sí
Si la alarma está activada: - Activado el led naranja para indicarlo - Led verde apagado - Se encienden los leds correspondientes cuando se detecta la presencia de intrusos en cada una de las habitaciones - Suena el buzzer en señal de alerta si hay intrusos	x x x x		Sí Sí Sí Sí
Si media alarma está activada: - Activado el led naranja para indicarlo - Led verde apagado - Se encienden los leds correspondientes cuando se detecta la presencia de intrusos en cada una de las habitaciones - El led rojo correspondiente a la habitación que está desactivada no se enciende si detecta intrusos - Suena el buzzer en señal de alerta si hay intrusos en las dos habitaciones restantes	x x x x x		Sí Sí Sí Sí Sí
¿Es posible el cambio de contraseña?	x		Sí

Si la contraseña introducida es correcta:	x		Sí
- Se vuelve al bucle que muestra las opciones	x		Sí
- Se enciende el led verde	x		Sí
- Se desactivan los leds rojos, el naranja y el buzzer	x		Sí
- Si se detecta presencia extraña, no ocurre nada	x		Sí
Si la contraseña introducida es incorrecta:	x		Sí
- Se borran los números introducidos y te vuelve a pedir la contraseña	x		Sí
- No se modifica el comportamiento del resto de elementos	x		Sí
Se permite modificar la contraseña satisfactoriamente:	x		Sí
- Al cambiar la contraseña, si se introduce la anterior, no lo detecta como válido	x		Sí
- Detecta como correcta la nueva contraseña a partir de ese momento	x		Sí
Se almacena la nueva contraseña	x	No, de momento no hemos conseguido que se almacene en la EEPROM la nueva contraseña	

9. Posibles extensiones

Creemos que este proyecto tiene bastantes extensiones posibles. Algunas de las que hemos considerado son las siguientes:

1. Aprovechando que nuestra maqueta tiene puertas y los usos que tienen las alarmas normalmente en chalets, etc., sería muy interesante añadir a nuestro proyecto tuviera interruptores reed. Los reed son interruptores eléctricos activados por un campo magnético. Los colocaríamos en puertas y ventanas para activar nuestra alarma si algún intruso accediera a la casa de alguna forma.
2. Tal y como está realizado el proyecto, si nuestros arduinos se quedaran sin alimentación la alarma fallaría, ya que no hay ningún sistema complementario que proporcione energía. Nos gustaría añadir una batería interna que entrara en funcionamiento cuando la alimentación principal fallara por cualquier motivo.
3. Sería interesante también contar con un sistema de protección dentro de la circuitería de la propia alarma, evitando así la manipulación no autorizada. De esta manera, si un intruso quisiese desactivarla (desenchufandola, cortando cables, etc.) la alarma se activará inmediatamente.
4. Adicionalmente, nos gustaría estudiar la posibilidad de que el usuario pueda desactivar la alarma de forma más sencilla, sin necesidad de introducir la clave. Para ello, habría dos opciones:
 - a. Crear una aplicación móvil que le permita activar la alarma y desactivarla de forma remota.
 - b. Utilizar un sistema de identificación por RFID (módulo lector RFID-RC522 RF) para desactivar la alarma.
5. Se podría añadir un apartado de configuración que permita programar la alarma para un calendario específico, de manera que ésta se conecte y desconecte en las fechas y horas indicadas por el usuario. Aplicando también un encendido y apagado de luces a determinadas horas para simular presencia durante un período en el que no estemos en casa.
6. Implementación de una parte de domótica donde, por ejemplo, las luces de la casa se encienden solas en la habitación en la que está el usuario si la alarma está desactivada pero le estamos detectando. Otra opción sería por ejemplo abrir la puerta del garaje o similares, para mejorar la experiencia del usuario.
7. Se podría mejorar y ampliar la maqueta comprando o fabricando mobiliario que personalice el ambiente, en este caso una vivienda unifamiliar

10. Conclusiones

Este proyecto ha sido fundamental para entender el funcionamiento de muchos de los componentes básicos que pueden utilizarse con un Arduino, y en especial aquellos que se imparten en la asignatura de tecnología en 3º y 4º de la ESO.

Integra, de forma que creemos será muy interesante para los alumnos, gran cantidad de componentes electrónicos.

Para llevar a cabo este proyecto en institutos, sería necesario adaptar ligeramente el proyecto. El problema es que es impensable utilizar el teclado matricial o la pantalla LCD, ya que son demasiado complicados de usar y de programar con niños. Sin embargo, es muy fácil reemplazarlo para que se pueda hacer el proyecto con ellos:

1. Por un lado, eliminaríamos completamente la pantalla LCD. Las opciones serán más simples:
 - a. La contraseña será siempre la que tengamos puesta por defecto en el código.
 - b. No habrá opción de activar la alarma por trozos, sino que solo podrán activar la alarma completa.
2. El teclado matricial se sustituirá. En lugar de escribir la contraseña en ese teclado, se reemplazará por botones, y cada uno de ellos corresponderá a uno de los números de la contraseña.
3. El feedback que recibe el usuario, en lugar de darse por la pantalla LCD, se hará utilizando los leds y el buzzer, con la posibilidad de incorporar algún elemento más que también ayude.

Sin embargo, este proyecto sería muy interesante para realizar en asignaturas como sistemas empotrados distribuidos (del máster de ingeniería informática) o similares, ya que el nivel se adecúa más a ese entorno que a una clase de instituto.

También sería posible llevar a cabo este proyecto como formación en microcontroladores en una empresa.

Por último, no hay que olvidar que este proyecto nos ayuda a comprender cómo funciona un sistema de detección de intrusos en la vida real. Esto es importante, ya que es un proyecto que, además de ayudarnos a comprender el funcionamiento básico de arduino y los componentes utilizados, nos ayuda a pensar en cómo la informática, los microcontroladores y el internet de las cosas pueden ser claves en nuestro día a día.

11. Trabajo futuro

A parte de las extensiones que se han mencionado previamente, consideramos que el trabajo que se podría llevar a cabo para mejorar o ampliar este proyecto es muy interesante.

Por un lado, habría que buscar una forma mejor de gestionar la cantidad de sensores que utilizamos (y, por tanto, el número de habitaciones cubiertas con nuestro sistema). Si llegáramos a incluir los interruptores reed en puertas y ventanas, nos quedaríamos sin pines en el arduino, por lo que habría que introducir **registros de desplazamiento** para solventar el problema.

Por otro lado, a pesar de que el código está preparado para que la opción 4 de configuración manual funcione (ya que hemos probado algo similar para activar solo media alarma y el funcionamiento sería parecido), no hemos terminado de programar cómo escogería el usuario qué habitaciones y sensores quiere que estén activadas. Esto no sería una tarea demasiado complicada, y como era similar a lo que ya habíamos hecho para la media alarma, hemos preferido centrarnos en la construcción de una maqueta apropiada para que el proyecto sea más educativo.

Finalmente, nos gustaría poder incorporar una parte domótica más real, como hemos mencionado en el apartado de extensiones. Aunque inicialmente nos planteamos que fuera parte de este proyecto, al final lo hemos descartado por falta de tiempo. El motivo es que hemos considerado más importante mejorar el código que teníamos, detallar el proceso completo que hemos seguido en la memoria para que el proyecto sea completamente reutilizable y hacer una maqueta para que al grabar el vídeo final no se vieran únicamente un conjunto de cables y no se entendiera nada.

12. Referencias

1. Comunicación serial entre Arduinos:
<http://robotic-controls.com/learn/arduino/arduino-arduino-serial-communication>
2. Elegoo. Elegoo Starter Kit para UNO
3. S. Fitzgerald and M. Shiloh, Arduino Projects Book
4. M. Margolis, Arduino Cookbook
5. D. Nedelkovski. Ultrasonic sensor hc-sr04 and arduino tutorial. Disponible en:
<http://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/>
6. Página de tutoriales Arduino: <https://www.arduino.cc/en/Tutorial/>
7. Inspiración para la maqueta: https://www.youtube.com/watch?v=dRCnccv_dVE
8. Youtube: tutoriales varios para aprender a utilizar alguno de los componentes electrónicos.

Anexo 1

En este apartado se muestra el código resultante que hace funcionar nuestro proyecto. Hay dos códigos diferentes, uno por cada uno de los Arduino UNO que estamos utilizando, ya que como hemos dicho previamente, cada uno de ellos necesita unas “instrucciones” específicas para funcionar y hacer su trabajo.

Nota: los códigos que se muestran en este apartado se corresponden con el montaje FÍSICO del circuito. No son exactamente iguales que los que utilizamos en Tinkercad, ya que el código del sensor ultrasonidos es ligeramente diferente. Sin embargo, el resto del código es exactamente igual en ambos. En caso de querer reproducir este proyecto en Tinkercad, será necesario volver a utilizar el sensor de ultrasonidos de la forma específica que se usa en esta plataforma.

1. Código proyecto arduino 1

```
/*This is the Sender arduino*/
#include <LiquidCrystal.h>
#include <Keypad.h>
#include <EEPROM.h>

int rx = 0;
int tx = 1;

/*-----KEYPAD
-----*/
const byte numRows= 4; //number of rows on the keypad
const byte numCols= 4; //number of columns on the keypad
char keypressed;
char keymap[numRows][numCols]=
{
{'1', '2', '3', 'A'},
{'4', '5', '6', 'B'},
{'7', '8', '9', 'C'},
{'*', '0', '#', 'D'},
};
//Code that shows the the keypad connections to the arduino
// terminals
byte rowPins[numRows] = {7,6,5,4}; //Rows 0 to 3
byte colPins[numCols] = {A0,A1,A2,A3}; //Columns 0 to 3
//initializes an instance of the Keypad class
Keypad myKeypad= Keypad(makeKeymap(keymap), rowPins, colPins,
    numRows, numCols);

/*----- FIN DEL KEYPAD
-----*/

// initialize the library with the numbers of the interface
// pins
LiquidCrystal lcd(12, 11, 10, 9, 8, 13);

// LCD Screen Resolution.
int screenWidth = 16;
int screenHeight = 2;

// the two lines
String lineOne, lineTwo;

// reference flags
int stringStart, stringStop, displayMode, i = 0;
int scrollCursor = screenWidth;
```

```

String linea1 = "1 Set alarm";                                41
String linea2 = "2 Set 1/2 alarm";                            42
String linea3 = "3 Change passwd";                           43
String linea4 = "4 Manual config";                          44
String linea1Act = "Alarm set";                             45
String linea2Act = "Enter passwd";                           46
bool alarmSet = false;                                     47
bool startAlarm = true;                                    48
                                         49
                                         50
// estados de la alarma                                     51
enum State { NONE, ALARM_SET, CHOSEN_ONE, CHOSEN_TWO,      52
    CHOSEN_THREE, CHOSEN_FOUR };
State estado;                                              53
                                         54
void changePass();                                         55
                                         56
void setup() {
    // set up the LCD's number of columns and rows:        57
    lcd.begin(screenWidth, screenHeight);                  58
                                         59
    Serial.begin (112500);                                 60
    //initialize UART pins                                61
    pinMode (rx, OUTPUT);                               62
    pinMode (tx, INPUT);                               63
                                         64
    lcd.clear();                                         65
    lcd.setCursor(0,0);       // situamos el cursor en la   66
        posicion 2 de la linea 0.                         67
    estado = NONE;                                       68
}
                                         69
                                         70
char codigoSecreto[4] = {'2','2','5','5'}; // Aqui va el   71
    codigo secreto
int cursor = 5;                                            72
int clave=0; // para el LCD                            73
int posicion=0; // necesaria para la clave            74
                                         75
                                         76
void changePass() {
    bool wrong = false;                                77
    int num1, num2, num3, num4;                      78
    int cont = 0;                                     79
                                         80
    lcd.clear();                                     81
                                         82

```

```

lcd.setCursor(0,0);
lcd.print("New passwd:");
lcd.setCursor(1,1);
lcd.print(">>> ");

while(cont < 4) {
    keypressed = myKeypad.getKey();

    if (keypressed != 0) //Si el valor es 0 es que no se ha
        pulsado ninguna tecla
    { // descartamos almohadilla y asterisco
        codigoSecreto[cont] = keypressed;
        lcd.print(keypressed);
        cont++;
    }
}

lcd.clear();
lcd.setCursor(0,0);
lcd.print("Passwd updated.");
delay(2000);

// lo guardamos en la eeprom para que la siguiente vez
empiece con ese
EEPROM.write(0, (int)codigoSecreto[0]);
EEPROM.write(1, (int)codigoSecreto[1]);
EEPROM.write(2, (int)codigoSecreto[2]);
EEPROM.write(3, (int)codigoSecreto[3]);
}

void compruebaNumYActua(int num) {
    if (keypressed == '1') {
        lcd.clear();
        lcd.setCursor(0,0);
        lcd.print("Alarm set!");
        estado = CHOSEN_ONE;

        alarmSet = true;
        startAlarm = true;

        byte dataR = 1; // lo que le mandamos al otro arduino
        Serial.print(dataR);
        delay(300);
        dataR = 2;
        Serial.print(dataR);
    }
}

```

```

    delay(300);                                126
                                                127
} else if (keypressed == '2') {
    lcd.clear();                               128
    lcd.setCursor(0,0);                      129
    lcd.print("Half alarm set!");           130
    estado = CHOSEN_TWO;                   131
                                            132
    alarmSet = true;                         133
    startAlarm = true;                       134
                                            135
    byte dataR = 5;                          136
    Serial.print(dataR);                    137
    delay(100);                            138
                                            139
} else if (keypressed == '3') {
    lcd.clear();                               140
    lcd.setCursor(0,0);                      141
    lcd.print("Changing pass!");            142
    changePass();                           143
    delay(100);                            144
    estado = CHOSEN_THREE;                 145
                                            146
} else if (keypressed == '4') {
    lcd.clear();                               147
    lcd.setCursor(0,0);                      148
    lcd.print("You pressed 4!");             149
    estado = CHOSEN_FOUR;                  150
}
}                                              151
                                                152
void clearBuffer() {                           153
    //clear out the serial buffer          154
                                            155
    byte w = 0;                             156
                                            157
    for (int i = 0; i < 10; i++)
    {
        while (Serial.available() > 0)
        {
            char k = Serial.read();
            w++;
            delay(1);
        }
        delay(1);
    }
}

```

```

    }

}

void setupTeclado()
{
    cursor = 5;
    clave = 0;
    posicion = 0;
    lcd.begin(16,2);
    lcd.setCursor(0,0);      // situamos el cursor el la
    // posicion 2 de la linea 0.
    lcd.print("Introduzca clave"); // escribimos en LCD
    lcd.setCursor(cursor,1); // cursor en la posicion de la
    // variable, linea 1
}

void loopTeclado()
{
    char pulsacion = myKeypad.getKey(); // leemos pulsacion
    int numPulsaciones = 0;
    if (pulsacion != 0) //Si el valor es 0 es que no se ha
    // pulsado ninguna tecla
    { // descartamos almohadilla y asterisco
        if (pulsacion != '#' && pulsacion != '*' && clave==0)
        {
            lcd.print(pulsacion); // imprimimos pulsacion
            numPulsaciones++;
            cursor++;
            delay(200);

            //---- Condicionales para comprobar la clave introducida
            -----
            // comparamos entrada con cada uno de los digitos, uno
            // a uno
            if (pulsacion == codigoSecreto[posicion]){
                posicion++; // aumentamos posicion si es correcto el
                // digito
            }

            if (posicion == 4)
            {
                // se han introducido los 4 correctamente
                byte dataX = 0; // lo que le mandamos al otro arduino
                Serial.print(dataX);
                delay(300);
            }
        }
    }
}

```

```

210
lcd.setCursor(0,0);           // situamos el cursor en la
211   pos 0 de la linea 0.
212 lcd.print("Clave correcta  ");          // escribimos
213   en LCD
214 delay(300);
215
216 dataX = 6; // lo que le mandamos al otro arduino
217 Serial.print(dataX);
218 delay(300);
219
220 // Jenny
221 //startAlarm = true;
222 alarmSet = false; // Disarmed!!!! lo quitamos!
223
224 clave=1; // indicamos que se ha introducido la clave
225
226 dataX = 0;
227 Serial.print(dataX);
228 delay(300);
229 }
230 if (numPulsaciones >=4 && posicion != 4) {
231   delay(200);
232   byte dataR = 3; // lo que le mandamos al otro arduino
233   Serial.print(dataR);
234   delay(200);
235 }
236 //--- En el caso de que este incompleta o no hayamos
237   acertado -----
238 if(cursor>8)           // comprobamos que no pase de la
239   cuarta posicion
240 {
241   cursor=5;           // lo volvemos a colocar al inicio
242   posicion=0;         // borramos clave introducida
243   lcd.setCursor(5,1);
244   lcd.print("      "); // borramos la clave de la
245   pantalla
246   lcd.setCursor(5,1);
247   if(clave==0)         // comprobamos que no hemos
248     acertado
249   {
250     delay(100);
251     // notificamos al otro arduino del error
252     byte dataR = 3; // lo que le mandamos al otro
253     arduino

```

```

        Serial.print(dataR);
        delay(100);

    }
}
}

//--- Condicionales para resetear clave introducida
-----
if (pulsacion == '*')
{ // asterisco para resetear el contador
    posicion = 0;
    cursor = 5;
    clave=0;
    posicion=0;
    lcd.setCursor(0,0); // situamos el cursor en la posicion
    2 de la linea 0.
    lcd.print("Introduzca clave"); // escribimos en LCD
    lcd.setCursor(5,1);
    lcd.print("      "); // borramos de la pantalla los numeros
    lcd.setCursor(5,1);
}
}

void loop() {

    int val1, val2, val3, val4;

    // leemos la clave que esta guardada
    val1 = EEPROM.read(0);
    val2 = EEPROM.read(1);
    val3 = EEPROM.read(2);
    val4 = EEPROM.read(3);

    if (val1 == 0 && val2 == 0 && val3 == 0 && val4 == 0) {
        // guardamos en la eeprom el codigo secreto inicial
        EEPROM.write(0, (int)codigoSecreto[0]);
        EEPROM.write(1, (int)codigoSecreto[1]);
        EEPROM.write(2, (int)codigoSecreto[2]);
        EEPROM.write(3, (int)codigoSecreto[3]);
    }

    // leemos la clave que esta guardada
    val1 = EEPROM.read(0);

```

```

val2 = EEPROM.read(1);                                291
val3 = EEPROM.read(2);                                292
val4 = EEPROM.read(3);                                293
                                                     294
if (!alarmSet) {                                     295
    setup();                                         296
    // 1. Activar alarma                            297
    // 2. Activar alarma (media alarma)             298
    // 3. Cambiar passwd                           299
    // 4. Configuracion manual (activar/desactivar aparatos
    de uno en uno)                                300
                                                     301
lcd.clear();                                         302
lcd.setCursor(0, 0);                                303
lcd.print(linea1);                                 304
lcd.setCursor(0, 1);                                305
lcd.print(linea2);                                 306
delay(1000);                                       307
lcd.clear();                                         308
lcd.setCursor(0, 0);                                309
lcd.print(linea2);                                 310
lcd.setCursor(0, 1);                                311
lcd.print(linea3);                                 312
delay(1000);                                       313
lcd.clear();                                         314
lcd.setCursor(0, 0);                                315
lcd.print(linea3);                                 316
lcd.setCursor(0, 1);                                317
lcd.print(linea4);                                 318
delay(1000);                                       319
                                                     320
                                                     321
lcd.clear();                                         322
lcd.setCursor(0,0);                                323
lcd.print("Chosen value: ");                      324
lcd.setCursor(0,1);                                325
lcd.print(">>> ");                            326
                                                     327
bool wrong = false;                                328
while(true) {                                     329
    keypressed = myKeypad.getKey();                330
                                                     331
    if (keypressed != 0) //Si el valor es 0 es que no se ha
        pulsado ninguna tecla                      332
    { // descartamos almohadilla y asterisco      333

```

```

    if (keypressed != '#' && keypressed != '*' &&
        keypressed <= '4') {
        lcd.print(keypressed);
        break;
    }
    if (keypressed > '4') {
        lcd.clear();
        lcd.setCursor(0,0);
        lcd.print("Wrong value!");
        lcd.setCursor(0,1);
        lcd.print(keypressed);
        lcd.setCursor(1,1);
        lcd.print(" not valid :(");
        wrong = true;
        break;
    }
}

delay(200);

// Si el numero es correcto, vemos que tenemos que hacer
// con el
if (!wrong) {
    compruebaNumYActua(keypressed);
}
delay(200);
}
else {
    // parte de la alarma */
    if (startAlarm == true) {
        setupTeclado();
        startAlarm = false;
    }
    loopTeclado();
    if (alarmSet == false) {
        delay(200);
    }
}
}

```

2. Código proyecto arduino 2

```
#include <EEPROM.h>                                1
int rx = 0;                                         2
int tx = 1;                                         3
int buzzer = 5;                                     4
// detectores                                     5
int ultrasonido = 8;                               6
// circuito fisico                                7
int echoPin = 8;                                    8
int trigPin = 6;                                   9
//                                         10
int pirPin = 7;                                    11
int pirPin2 = 13;                                 12
// leds                                         13
int ledNaranja = 3;                                14
int ledVerde = 12;                                 15
int ledPinRojoH1 = 9;                             16
int ledPinRojoH2 = 10;                            17
int ledPinRojoH3 = 11;                            18
byte val; // value read on from the serial port   19
byte dataR; //it contains the byte read from EEPROM 20
25
enum State { NONE, FULL_ALARM, HALF_ALARM, ALARM_SPLIT }; 21
State estado;                                         22
long readUltrasonicDistance(int pin);               23
void loopAlarmaCompleta();                         24
void enciendeMediaAlarma();                        25
void loopMediaAlarma();                           26
27
void setup()                                         27
{                                                 28
    Serial.begin (112500);                         29
    delay(500);                                    30
    pinMode(ledNaranja, OUTPUT);                  31
    pinMode(ledVerde, OUTPUT);                   32
    pinMode(buzzer, OUTPUT);                     33
    pinMode(ledPinRojoH1, OUTPUT);                34
35
36
37
38
39
40
41
42
43
44
45
```

```

pinMode(ledPinRojoH2, OUTPUT);                                46
pinMode(ledPinRojoH3, OUTPUT);                                47
pinMode(pirPin, INPUT_PULLUP);                               48
pinMode(pirPin2, INPUT_PULLUP);                               49
pinMode(ultrasonido, INPUT);                                 50
                                         51

// fisico
pinMode(trigPin, OUTPUT);                                  52
                                         53
                                         54

//initialize UART pins
pinMode(rx, OUTPUT);                                     55
pinMode(tx, INPUT);                                      56
estado = NONE;                                         57
}                                                       58
                                         59
                                         60

void apagaBuzzer() {
    noTone(buzzer);                                    61
}                                                       62
                                         63

void enciendeLedVerde() {
    digitalWrite(ledVerde, HIGH);                      64
}                                                       65
                                         66

void enciendeLedNaranja() {
    digitalWrite(ledNaranja, HIGH);                    67
}                                                       68
                                         69

void apagaLedVerde() {
    digitalWrite(ledVerde, LOW);                       70
}                                                       71
                                         72

void apagaLedNaranja() {
    digitalWrite(ledNaranja, LOW);                     73
}                                                       74
                                         75

void buzzer1() {
    tone(buzzer,350);                                76
}                                                       77
                                         78

void buzzerClaveCorrecta() {
    delay(200); // tono de clave correcta          79
    tone(buzzer,500);                                80
    delay(100);                                     81
    noTone(buzzer);                                82
    tone(buzzer,600);                                83
    delay(100);                                     84
    noTone(buzzer);                                85
    tone(buzzer,800);                                86
    delay(100);                                     87
    noTone(buzzer);                                88
}                                                       89
                                         90

```

```

void buzzerClaveIncorrecta() {
    delay(200); // tono de clave incorrecta
    tone(buzzer,300);
    delay(100);
    noTone(buzzer);
    tone(buzzer,100);
    delay(100);
    noTone(buzzer);
}

void buzzerAlarma() {
    tone(buzzer, 800); // play 400 Hz tone for 400 ms
    delay(200);
    tone(buzzer, 500); // play 800Hz tone for 400ms
    delay(200);
    noTone(buzzer);
}

bool intrusos = false;
bool alarmOn = false;
bool halfAlarmOn = false;

// los tres siguientes solo se activan cuando
// queremos encender cada hab por separado
bool hab1On = false;
bool hab2On = false;
bool hab3On = false;

void desactivaDetencion() {
    digitalWrite(ledPinRojoH1, LOW);
    digitalWrite(ledPinRojoH2, LOW);
    digitalWrite(ledPinRojoH3, LOW);
    apagaBuzzer();
    enciendeLedVerde();
    apagaLedNaranja();
    alarmOn = false;
    halfAlarmOn = false;
}

void loop()
{
    if(Serial.available() > 0)
    {

```

```

    val = Serial.read(); //read the next byte          136
    137
/*
0 - enciende led verde
1 - apaga led verde
2 - enciende led naranja
3 - apaga led naranja
4 - apaga buzzer
5 - enciende media alarma
6 - desactiva deteccion
7 - activa habitacion 1
8 - activa habitacion 2
9 - activa habitacion 3
10 - change pass (cambio de contraseña)
*/
if (val == '0') {
    desactivaDeteccion();
} else if (val == '1') {
    apagaLedVerde();
} else if (val == '2') {
    enciendeLedNaranja();
    alarmOn = true;
} else if (val == '3') {
    buzzerClaveIncorrecta();
} else if (val == '4') {
    apagaBuzzer();
} else if (val == '5') {
    enciendeMediaAlarma();
} else if (val == '6') {
    desactivaDeteccion();
    // tono de clave correcta?
    buzzerClaveCorrecta();
} else if (val == '7') {
    // activa hab 1
    hab1On = true;
} else if (val == '8') {
    // activa hab 2
    hab2On = true;
} else if (val == '9') {
    // activa hab 3
    hab3On = true;
}
} else {
    //buzzerClaveCorrecta();
}

```

```

        }

    if (alarmOn) {
        loopAlarmaCompleta();
    } else if (halfAlarmOn) {
        loopMediaAlarma();
    }

    delay(200);
}

long readUltrasonicDistance(int pin)
{
    // pinMode(trigPin, OUTPUT); // Clear the trigger
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    // Sets the pin on HIGH state for 10 micro seconds
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(pin, LOW);
    //pinMode(pin, INPUT);
    // Reads the pin, and returns the sound wave travel time in
    // microseconds
    //return pulseIn(pin, HIGH);
    return pulseIn(echoPin, HIGH);
}

void loopAlarmaCompleta()
{
    bool desactivando = false; // utilizamos el valor
    desactivando como valor intermedio
    // hasta que no nos informen los dos sensores de que
    // realmente no estan detectando
    // a nadie no podemos encender el led verde como que todo
    // est en orden. Tenemos
    // que esperar a que ambos comprueben que no hay nadie para
    // encender el verde

    int proximity = digitalRead(pirPin);
    delay(100);

    int proximity2 = digitalRead(pirPin2);
    delay(100);

    int cm = 0.01723 * readUltrasonicDistance(ultrasonido);
}

```

```

221 if (proximity == HIGH) // If the sensor's output goes low,
222     motion is detected
223 {
224     digitalWrite(ledPinRojoH1, HIGH);
225     intrusos = true;
226 }
227 else
228 {
229     digitalWrite(ledPinRojoH1, LOW);
230     desactivando = true;
231 }
232
233 if (proximity2 == HIGH) // If the sensor's output goes low,
234     motion is detected
235 {
236     digitalWrite(ledPinRojoH3, HIGH);
237     intrusos = true;
238     desactivando = false;
239 }
240 else
241 {
242     digitalWrite(ledPinRojoH3, LOW);
243 }
244
245 if (cm < 50) {
246
247     digitalWrite(ledPinRojoH2, HIGH);
248     intrusos = true;
249 }
250 else
251 {
252     digitalWrite(ledPinRojoH2, LOW);
253     if (desactivando == true) {
254         intrusos = false;
255     }
256
257     if (intrusos) {
258         buzzerAlarma();
259     }
260 }
261
262 void enciendeMediaAlarma() {
263

```

```

enciendeLedNaranja();
apagaLedVerde();
halfAlarmOn = true;
}

// loop media alarma
// equivale al plan nocturno, donde van a estar activados dos
// de los tres sensores
// no queremos que el de la habitaci n nos detecte
void loopMediaAlarma()
{
    bool desactivando = false; // utilizamos el valor
        desactivando como valor intermedio
    // hasta que no nos informen los dos sensores de que
        realmente no estan detectando
    // a nadie no podemos encender el led verde como que todo
        est en orden. Tenemos
    // que esperar a que ambos comprueben que no hay nadie para
        encender el verde

    // leemos el pir pin 1
    int proximity = digitalRead(pirPin);
    delay(100);

    // leemos el ultrasonido
    int cm = 0.01723 * readUltrasonicDistance(ultrasonido);

    if (proximity == HIGH) // If the sensor's output goes low,
        motion is detected
    {
        //tone(buzzer, 350, 200);
        digitalWrite(ledPinRojoH1, HIGH);
        intrusos = true;
    }
    else
    {
        digitalWrite(ledPinRojoH1, LOW);
        desactivando = true;
    }

    // comprobamos la distancia al ultrasonidos
    // si es menor de 200 cm, encendemos el led rojo de la
    // habitacion 2
    if (cm < 50) {
        // tone(buzzer, 350, 200);

```

```
    digitalWrite(ledPinRojoH2, HIGH);
    intrusos = true;
}
else
{
    digitalWrite(ledPinRojoH2, LOW);
    if (desactivando == true) {
        intrusos = false;
    }
}

if (intrusos) {
    buzzerAlarma();
}
}
```

303
304
305
306
307
308
309
310
311
312
313
314
315
316
317