

# PROYECTO FINAL

SENSOR DE MOVIMIENTO

CON ESP32



David Del Valle Puado

2º STI

11/06/2021

<b>1 RESUMEN</b>	2
<b>2 JUSTIFICACIÓN</b>	2
¿Por qué no conectar Arduino Uno al módulo ESP8266?	3
¿Por qué no conectar Arduino Nano al módulo ESP8266?	3
Placa ESP32	4
¿Por qué elegir el módulo ESP32 frente al ESP8266?	5
Módulo de sensor de movimiento HC-SR501	6
HC-SR501 frente al sensor HC-SR04	12
Servomotor SG90	12
Telegram	14
Arduino-IDE	15
Fritzing	16
<b>3 OBJETIVOS</b>	18
Explicación	18
Nivel de domotización	20
Configuración de la placa ESP32 en Arduino-IDE	24
Montaje de los diversos componentes	26
<b>4 FUNDAMENTOS TEORICOS</b>	28
<b>5 DESCRIPCIÓN DEL DESARROLLO DEL PROYECTO</b>	29
Listado y obtención de materiales a utilizar:	29
Montaje en simulador para las primeras pruebas.	29
Pruebas y desarrollo en las siguientes placas	29
Desarrollo del código en Arduino-IDE (C ++)	29
Montaje físico y pruebas del código	29
Pruebas finales y retoques	29
Montaje de todo el conjunto	29
<b>6 TEMPORIZACION DEL PROYECTO</b>	29
<b>7 RESULTADOS Y ANALISIS</b>	32
<b>8 CONCLUSIONES</b>	35
<b>9 LINEAS DE INVESTIGACIÓN FUTURAS</b>	35
Creación de modelo con placa PCB	36
Desarrollo de una interfaz web y/o aplicación para dispositivos móviles	38
Implementación en el mercado.	38
<b>10 BIBLIOGRAFÍA</b>	44
<b>11 ANEXOS</b>	45

## 1 RESUMEN

El motivo principal para realizar este proyecto llegó con la idea de combinar las telecomunicaciones con la programación, además, también tenía la necesidad de poder montarlo físicamente en un espacio reducido.

Utilizar una placa Arduino, en este caso un *clon* fabricado por *Az-Delivery*, llegó con la idea de proporcionar un *proyecto* de artefacto domótico que se pudiera conectar a una aplicación móvil y que tuviera un coste total mucho menos elevado que algunos otros elementos del mercado.



### 1. ESP32

Concretamente la placa utilizada es el modelo ESP32, un potente módulo genérico de integración de Bluetooth, Bluetooth LE y Wifi, que permite una gran gama de aplicaciones.

Gracias a su conexión Wifi, podremos conectar el ESP32 a la aplicación *Telegram* para que nos envíe las alertas directamente a nuestro dispositivo móvil, reloj inteligente u ordenador mediante notificaciones en un chat creado por un *Bot* de la propia aplicación.

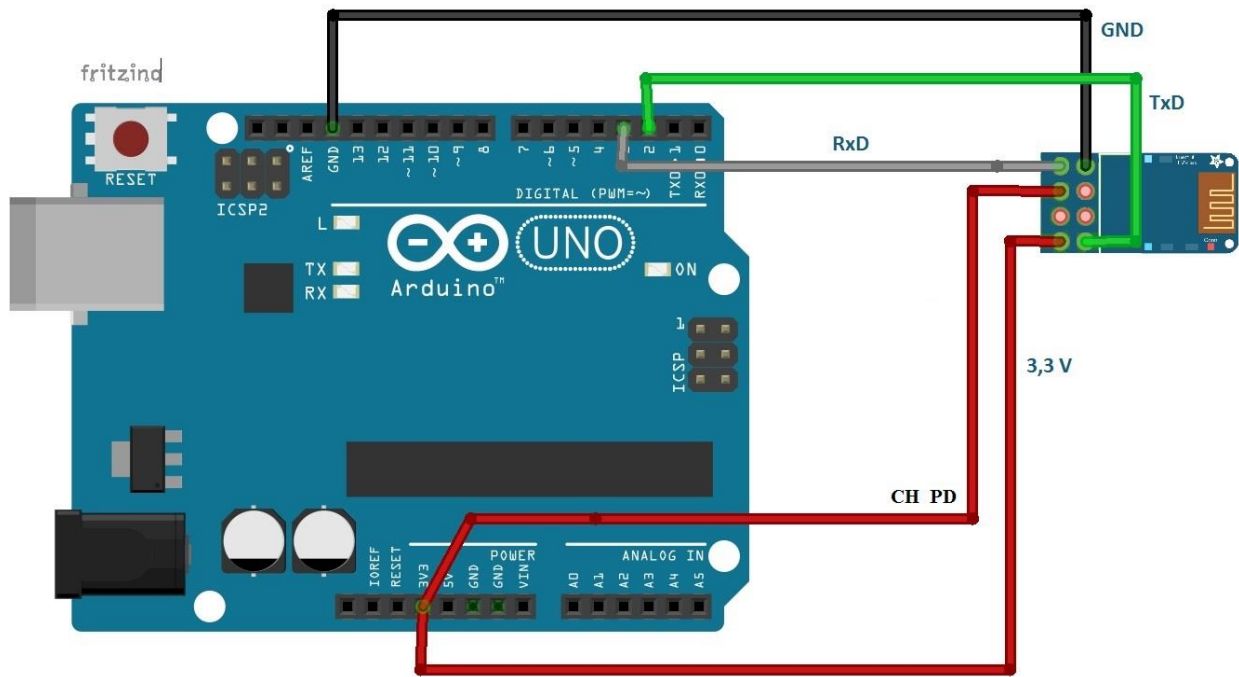
Me gustaría aclarar, que este sensor de movimiento solo estará en una fase beta al momento de su exposición y que se detallará en el [punto nueve](#) sus posibles desarrollos futuros con el fin de compactar, miniaturizar y abaratar el diseño lo máximo posible.

## 2 JUSTIFICACIÓN

Las razones para la realización de este proyecto es el estudio de la placa Arduino aplicada a las telecomunicaciones, dado que considero este microcontrolador como una placa con infinitos y útiles desarrollos en nuestro ámbito.

### ¿Por qué no conectar Arduino Uno al módulo ESP8266?

Arduino Uno es una placa de microcontrolador basada en ATmega328P. Tiene 14 pines de entrada /salida digital (de los cuales 6 se pueden usar como salidas PWM), 6 entradas analógicas, un resonador cerámico de 16 MHz (CSTCE16M0V53-R0), una conexión USB, un conector de alimentación, un encabezado ICSP y un botón de reinicio. Contiene todo lo necesario para soportar el microcontrolador; simplemente se debe conectar a un ordenador con un cable USB o encenderlo con un adaptador de corriente o una batería para comenzar.



### 2. Esquema de conexiones entre Arduino Uno y módulo Wifi

La placa Arduino Uno es la más genérica de todas y, a su vez, la más utilizada en los proyectos de electrónica con este sistema. Fue descartada por su tamaño más voluminoso, dado que buscamos ocupar el menor espacio posible, y su difícil integración con el módulo ESP8266.

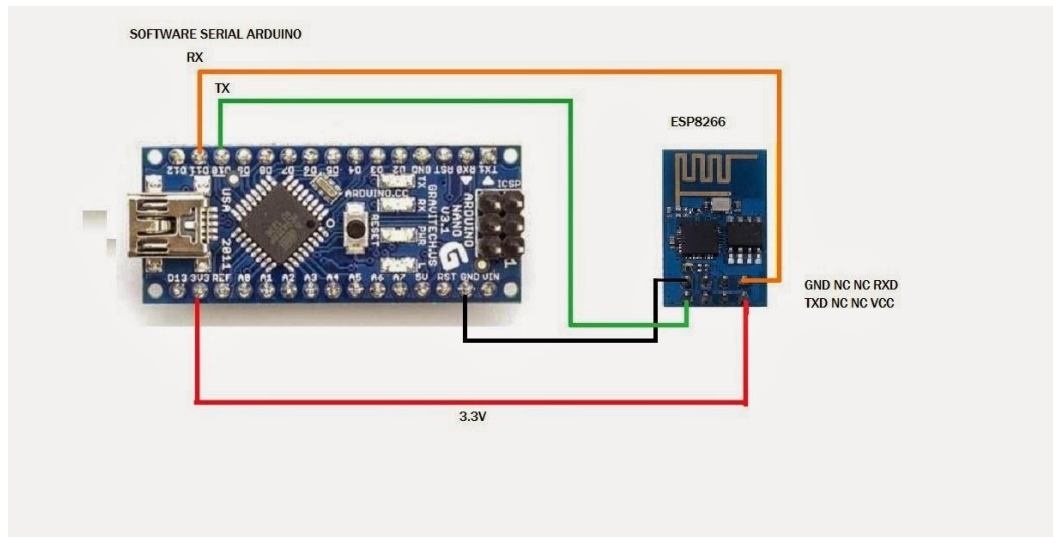
En la primera fase de este proyecto hice en un simulador (*Fritzing*) diferentes pruebas con esta placa y no me llegó a convencer su integración con el módulo Wifi, aunque funcionaba bien con el sensor de movimiento, pero no logré realizar el envío de datos a la plataforma.

### ¿Por qué no conectar Arduino Nano al módulo ESP8266?

Arduino Nano, es una versión reducida de Arduino UNO, eso minimiza la demanda de energía que consume y también hace que no se necesite tanto espacio para alojar la placa, por lo que es ideal para proyectos donde el tamaño sea importante, como es este caso.

Está basada en la ATmega328 e incorpora la ATmega328P, la misma que el Arduino Uno. La principal diferencia entre ellos es que el Arduino Uno se presenta en forma de PDIP (Plastic Dual-In-line Package) con 30 pines y la Arduino Nano está disponible en TQFP (Plastic quad flat pack) con 32 pines. No tiene una toma de alimentación de corriente directa como otras placas Arduino, sino que tiene un puerto mini-USB que se utiliza tanto para la programación como para la monitorización en serie. La característica fascinante de Nano es que elegirá la fuente de energía más fuerte con su diferencia de potencial, y la fuente de energía que selecciona el puente no es válida.

En el siguiente esquema se muestra la conexión del microcontrolador con el módulo Wifi.



3. Esquema de conexiones entre Arduino Nano y módulo Wifi

Dado que en este proyecto se busca la miniaturización y la simplicidad del diseño se ha evitado usar una mayor cantidad de módulos y de conexiones con todos los componentes. Además, en la placa usada (ESP32), el propio módulo Wifi viene integrado en ella, logrando eliminar un componente a conectar y ganando conexiones de los pines.

### Placa ESP32

Creado y desarrollado por *Espressif Systems*, el ESP32 tiene una serie de microcontroladores de bajo costo y de bajo consumo con sistema en chip con Wi-Fi y Bluetooth de modo dual integrados. Como una radio combinada Wi-Fi/Bluetooth de bajo costo, es popular no solo entre los aficionados sino también entre los desarrolladores de IoT. Su bajo consumo de energía, sus múltiples entornos de desarrollo de código abierto y sus bibliotecas la hacen perfectamente adecuada para desarrolladores de todo tipo.

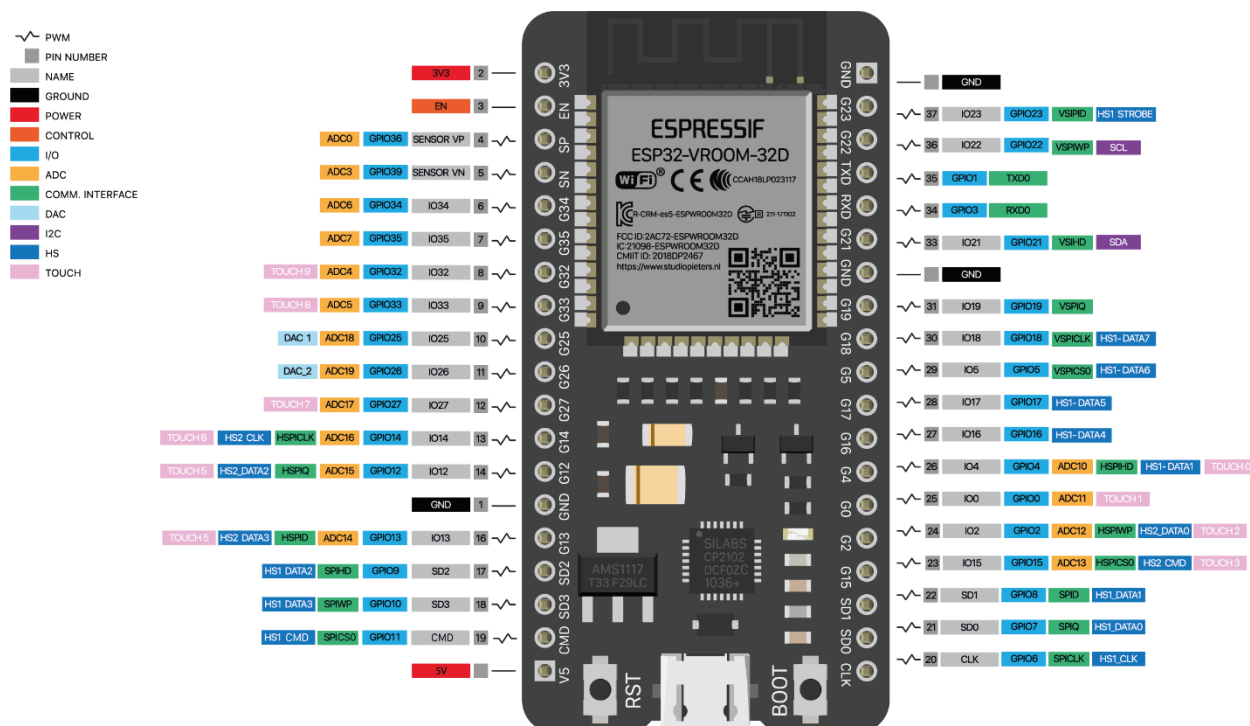


4. Microcontrolador ESP32 de Espressif Sytems

El módulo ESP32 es una solución de Wi-Fi /Bluetooth todo en uno, integrada y certificada. El procesador en realidad tiene dos núcleos de procesamiento cuyas frecuencias operativas pueden controlarse independientemente entre 80 MHz y 240 MHz.

El microcontrolador además de ser 10 veces más rápido, es de arquitectura de 32 bits y doble núcleo, por lo que la velocidad de procesamiento de datos es muy superior a la de un ATmega328P ([Arduino Uno](#) y [Arduino Nano](#)) Además, la ESP32 nos ofrece 18 pines analógicos, 3 veces más una placa Arduino Uno.

A continuación, se muestra el esquema del módulo ESP32:





### 5. Pinout de placa ESP32

Como podemos deducir, este microcontrolador tiene todo lo que buscamos para la correcta realización de este proyecto además de multiplicar por tres la velocidad de las placas Arduino Uno y Nano.

### ¿Por qué elegir el módulo ESP32 frente al ESP8266?

A simple vista parecen prácticamente iguales, pero tienen una gran cantidad de características distintivas que se exponen en la siguiente tabla:

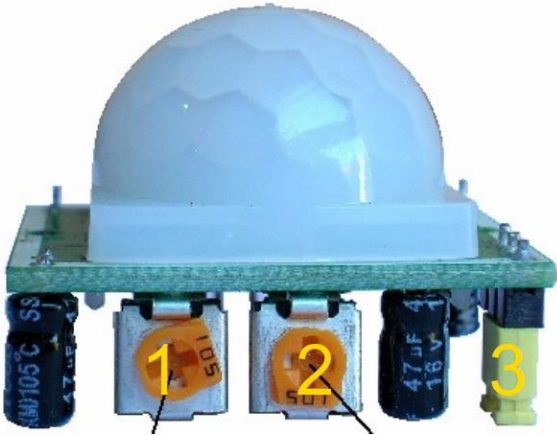
Característica	ESP32	ESP8266
Módulo		

CPU	Xtensa Dual-Core 32-bit LX6 con 600 DMIPS	Xtensa Single-core 32-bit L106
Velocidad del Wi-Fi	802.11n hasta 150 Mbps	Hasta 72,2 Mbps
GPIO	36	17
Bluetooth	Sí	No
DAC	Dos canales DAC de 8 bits	No
ADC	SAR de 12 bits	SAR de 10 bits
Canales ADC	8 canales	Un solo canal
Modos Wifi	Station/SoftAP/SoftAP+Station/P2P	Station/SoftAP/SoftAP+Station/P2P
Sensor táctil	Sí (8-Canales)	No
Sensor de temperatura	Sí	No
SRAM	520 kB (8 kB de SRAM en RTC)	Tamaño de la RAM < 50 kB
FLASH (Externo)	4Mbytes	4Mbytes
ROM	448 kB de ROM para el arranque y las funciones básicas	No hay ROM programable
Protocolos de red	IPv4, IPv6, SSL, TCP/UDP/HTTP/FTP/MQTT	IPv4, TCP/UDP/HTTP/MQTT
Rango de temperatura de funcionamiento	-40°C ~ +85°C	-40°C ~ 125°C
Tensión de funcionamiento	2.5V ~ 3.6V	2.5V ~ 3.6V
Corriente operativa	Promedio: 80 mA	Valor medio: 80 mA
Puerto	Micro-USB	Mini-USB
Precio	4€ - 10€	3€ - 6€

### Módulo de sensor de movimiento HC-SR501

El HC-SR501 es un tipo de sensor de movimiento, un sensor PIR (sensor infrarrojo pasivo) que consta de dos elementos separados. Por un lado, tiene un dispositivo emisor de la señal diferencial entre él y por otro, los sensores que activaran la señal de alarma.

El módulo es un sensor de bajo coste, pequeño y con una de las tecnologías más avanzadas de todos los sensores de movimiento actuales. Con sus dos potenciómetros y el jumper que integra se pueden modificar sus parámetros fácilmente, adaptándolas para todas las necesidades de sensibilidad y distancia, e incluso tiempo de activación y respuesta.



7. Control de sensibilidad del HC-SR501



6. Sensor HC-SR501 sin su cubierta

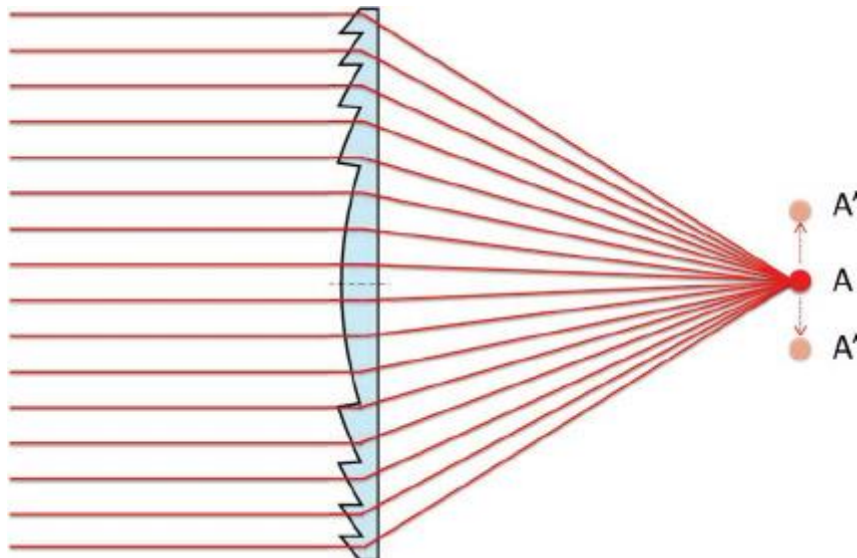
Esto se consigue mediante un circuito integrado BISS0001, que contiene amplificadores operacionales e interfaces electrónicas adicionales. Además de eso, el módulo permite realizar dos ajustes de sus funciones, uno es para la sensibilidad de la distancia de detección del PIR con unos potenciómetros. La otra función es la capacidad de detección automática de luz, aunque no está habilitada de fábrica

Esa última función se suele usar para algunos sistemas para que enciendan la luz de un sistema cuando se detecta movimiento, pero la iluminación ambiental no es elevada, es decir, cuando es de noche.

En el caso del HC-SR501, se tiene un rango de detección de movimiento con un alcance de 3 hasta 7 metros de distancia, y aperturas del PIR de hasta 90 y 110°.

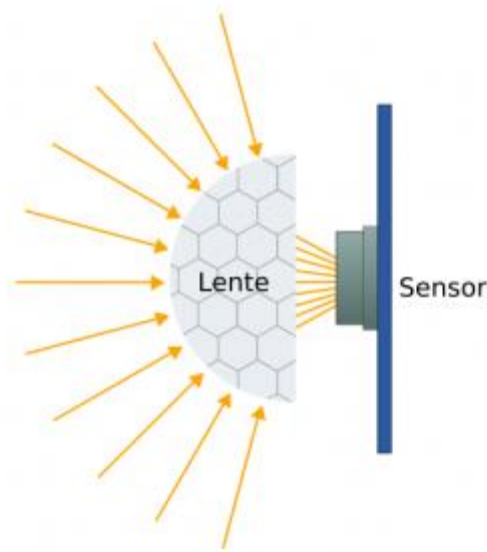
Como puedes ver, el sensor PIR está cubierto por una especie de cúpula blanca, eso es lo que se conoce como lente de Fresnel. Se llama así por el inventor y físico francés Augustin-Jean Fresnel. Gracias a él se permite construir lentes de gran apertura y corta distancia focal sin el peso y volumen del material que debería usarse con una lente convencional.





8. Funcionamiento de la lente

Gracias al diseño de esta lente inventada en 1822, se ha podido implementar en multitud de dispositivos, entre ellos el HC-SR501.



9. Lente implementada junto al sensor

A continuación, se listan las especificaciones técnicas del componente:

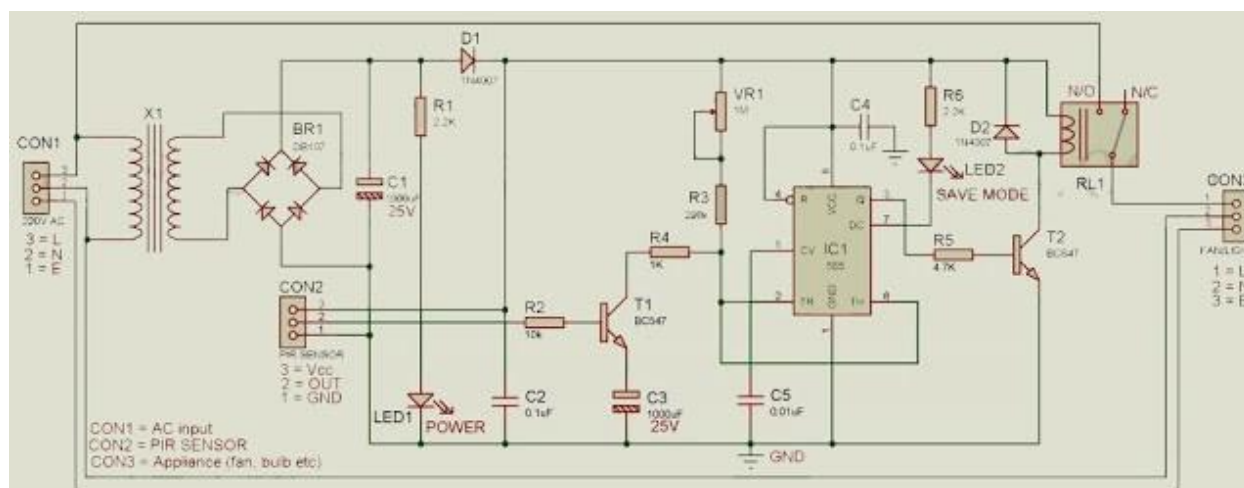
Componentes	PIR LH1778 y el controlador BISS0001
Voltaje	5-20 V(DC)
Nivel de las salidas	3.3 V
Tiempo de espera	5-300 segundos (ajustable)
Dimensiones	32mm*24mm
Angulo del sensor	110° ángulo cónico
Temperatura operacional	-15°/70°

Tamaño de la lente	23mm
--------------------	------

También podemos analizar el esquema interno de conexiones del sensor.

Podemos ver el diagrama del circuito que engloba varios componentes electrónicos como resistencias o un rectificador puente. El sensor de este circuito monitorea la radiación infrarroja continuamente y queda obstruido con la presencia de personas. En el momento que el sensor detecta algún tipo de presencia el patrón de radiación cambia y emite una señal alta de 3.3 V que se amplifica con un transistor (T1), después es entregada a un pin de activación (pin 2) y al pin de umbral (pin 6) del Timer IC 555.

El propio sensor PIR tiene un lapso de tiempo pequeño (esto se refiere a que nos otorga un monitoreo continuo) que se puede aumentar según nuestros requerimientos. Esto traducido al circuito sería que el temporizador IC actúa como resistencia (R3) y el potenciómetro (VR1) y el condensador (C3) actúan como circuito de retardo de tiempo. La salida del pin 3 del IC1 se alimenta al transistor (T2) que controla el relé (RL1).



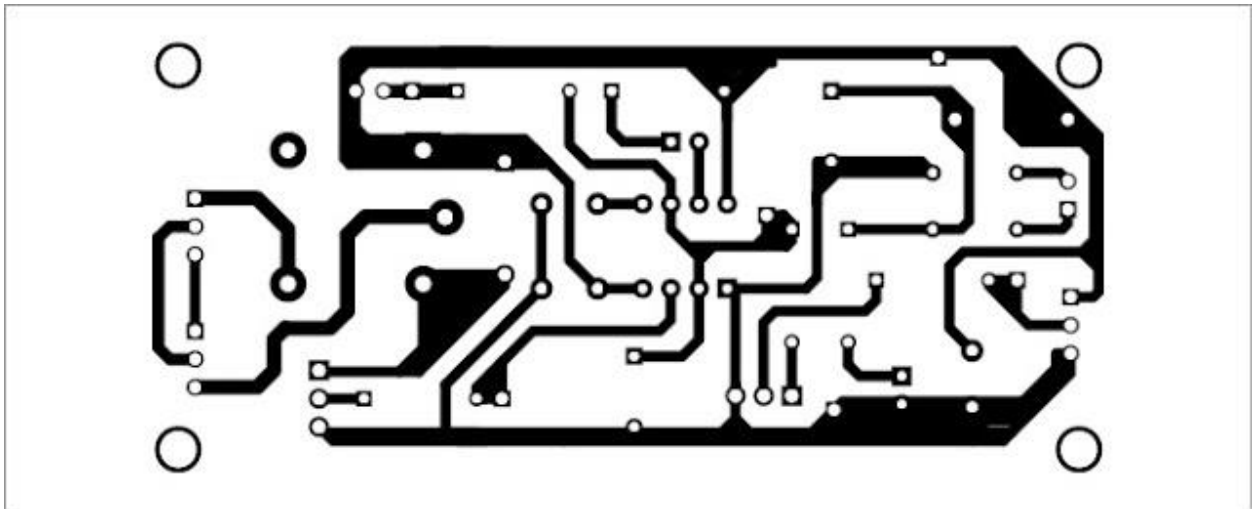
10. Esquema interno HC-SR501

En la primera fase, la resistencia variable (VR1) junto con la resistencia (R3) carga el condensador (C3) en el momento que el circuito se pone en funcionamiento. En ese punto, IC1 ofrece voltajes en los pines 2 y 6, que serían dos tercios del suministro del voltaje ( $\frac{2}{3} V_{cc}$  de IC). Debido a todo esto la salida del pin 3 sube y eso activa el relé del transistor (T2) para que el dispositivo empiece a funcionar.

Una vez el circuito se activa, el sensor PIR empieza a verificar cualquier cambio en el patrón de radiación. Cuando se encuentra algún bloqueo o movimiento en el pin de salida del sensor PIR sube, que son 3.3V aproximadamente para un periodo de tiempo definido previamente.

Esto se alimenta como entrada al transistor T1 y nos daría como resultado la descarga del condensador C3 a través de la resistencia R4. El ciclo continúa una vez que el voltaje del condensador cae por debajo de las dos terceras partes de la fuente de alimentación.

Las siguientes imágenes (11 y 12) representan el diseño del componente del lado de la soldadura y del lado del componente respectivamente.

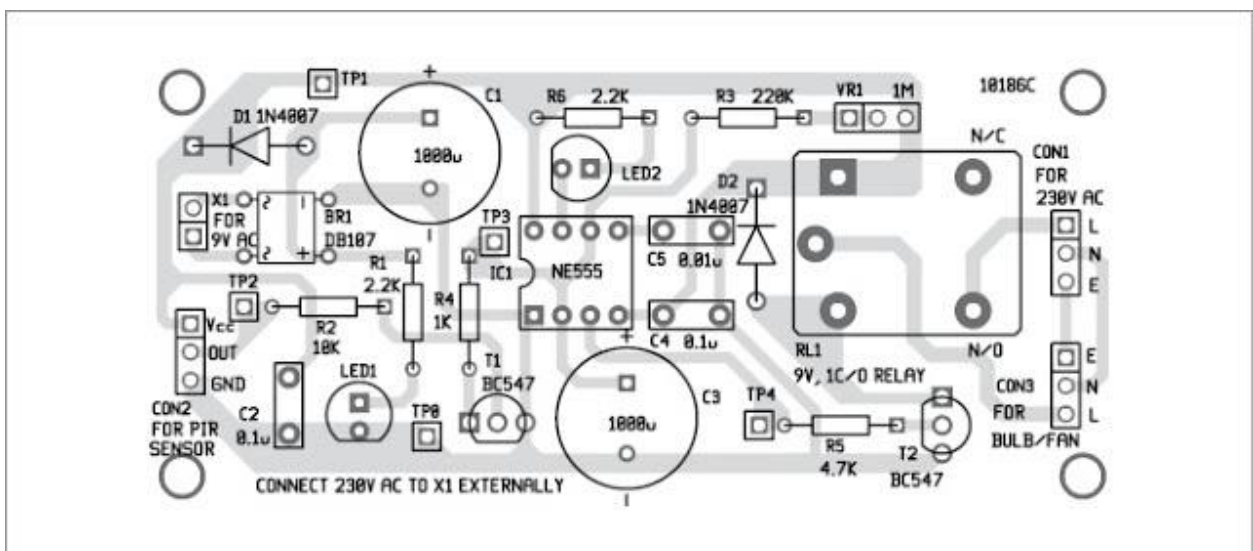


11. PCB del lado de soldadura del diagrama de circuito

Observamos que se mantiene la PCB en un espacio reducido para facilitar su incorporación en otros circuitos y/o sistemas

Para poder garantizar un mejor funcionamiento y detección de movimiento, el sensor debe revisarse antes de su puesta en marcha comprobando que todos los componentes estén funcionando. Después de esto, conectaremos los pines Vcc y GND para iniciar su puesta en marcha.

Los factores como la sensibilidad y los controles de tiempo se ajustaran acorde al proyecto aunque para obtener una máxima sensibilidad y señal de tiempo tendríamos que girar ambos preajustes en sentido horario. Hay que tener en cuenta que para mejorar la detección, la superficie de la lente del sensor PIR debe estar limpia.



12. PCB del lado de componentes del diagrama del circuito

A continuación, se listarán todos los componentes del sensor PIR:

Resistencias ( ± 5% de carbono)	
Nombre	Valor
R1, R6	22 KΩ
R2	10 KΩ
R3	220 KΩ
R4	1 KΩ
R5	4.7 KΩ
VR 1	1 MΩ POT

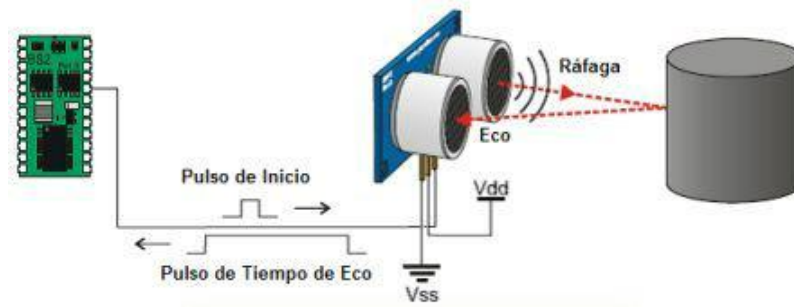
Condensadores		
Nombre	Valor	Tipo
C1, C3	1000 μF, 25 V	Electrolítico
C2, C4	0.1 μF	Disco de cerámica
C5	0.01 μF	Disco de cerámica

Semiconductores		
Nombre	Modelo	Tipo
IC1	NE555	Temporizador IC
T1, T2	BC547	Transistor NPN
D1, D2	1N4007	Diodo rectificador
BR1	DB107	Puente rectificador
Led 1, Led 2	5mm	Led de color

Otros componentes		
Nombre	Valor	Tipo
Modulo sensor PIR	N/A	N/A
X1	230 V AC Primario a 9V	3000 mA transformador secundario
RL1	9V, 1C / O relé	N/A
CON1-CON3	N/A	Conector de 3 pins

### HC-SR501 frente al sensor HC-SR04

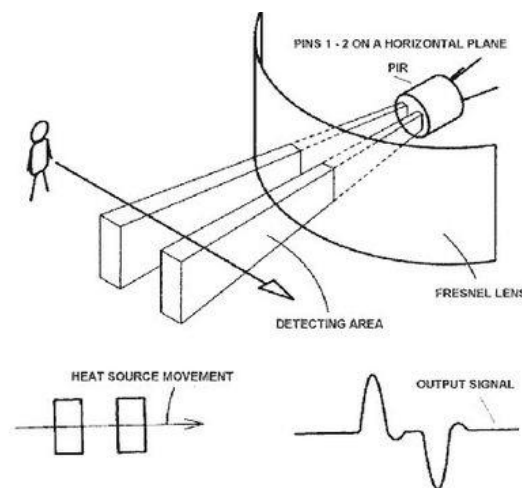
El sensor HC-SR04 es un sensor de distancia de bajo costo, su uso es muy frecuente en la robótica, utiliza transductores de ultrasonido para detectar objetos.



13. Funcionamiento del sensor HC-SR04

Su funcionamiento consiste en emitir un sonido ultrasónico por uno de sus transductores, y esperar que el sonido rebote de algún objeto presente, el eco es captado por el segundo transductor. La distancia es proporcional al tiempo que tarda en llegar el eco.

Por el contrario, y como ya hemos visto anteriormente, el HC-SR501 es un tipo de sensor pasivo de movimiento por infrarrojos.



14. Detección de movimiento HC-SR501

Sólo funciona cuando alguien se mueve en la franja que puede barrer su detector. Puede detectar movimiento de 3 hasta 7 metros de distancia.

### Servomotor SG90

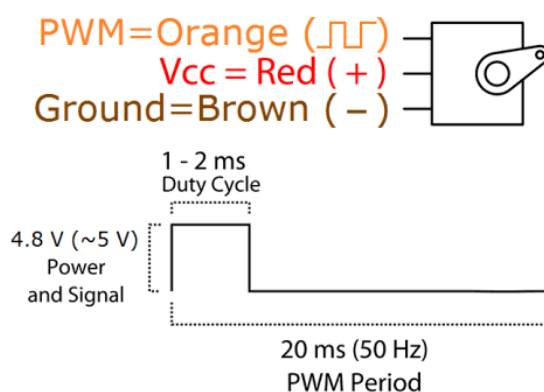
El servomotor SG90 llegó a ser incluido en este proyecto debido a un problema con el rango de cobertura del componente HC-SR501. Es un servo en miniatura de dimensiones reducidas, con un bajo consumo y un precio económico. La intención es usarlo para establecer un movimiento de 180° junto al sensor de movimiento.



15. Imagen del servomotor SG90

El SG90 ira conectado al microcontrolador ESP32, concretamente en el pin 9, para mantener todos los componentes en una misma placa de desarrollo y, así, conseguir un consumo más eficiente de energía junto con un tamaño más pequeño y una reducción de la cantidad de cables a usar.

En la siguiente foto podemos observar su esquema de conexiones, así como su tensión de alimentación y su frecuencia de trabajo.



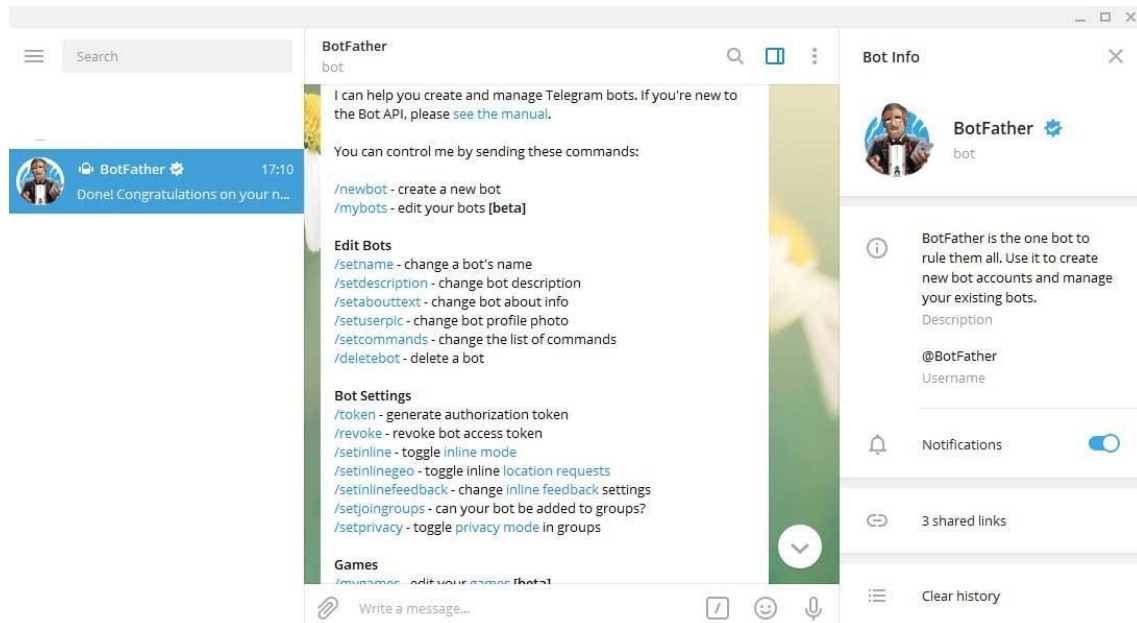
16. Pinout SG90

Las características técnicas de este módulo son las siguientes:

Dimensiones	22mm x 11,5mm x 27mm
Peso	9 gramos
Peso con cable y conector	10,6 gramos
Torque a 4.8 V	1.2 kg/cm
Voltaje de operación	4.0 a 7.2 volts
Velocidad de giro	120ms / 60 °
Compatibilidad	Tarjetas Arduino y microcontroladores a 5V
Precio	2.5€ - 5.5€

## Telegram

Telegram es una aplicación de mensajería donde no es necesario dar tu teléfono móvil (también funciona por un sistema de nombres de usuario); donde se pueden enviar mensajes de texto, voz e imágenes. Concretamente nos enfocaremos en su sistema de *bots* (chats automatizados) que usaremos para recibir las alertas de movimiento desde el ESP32.

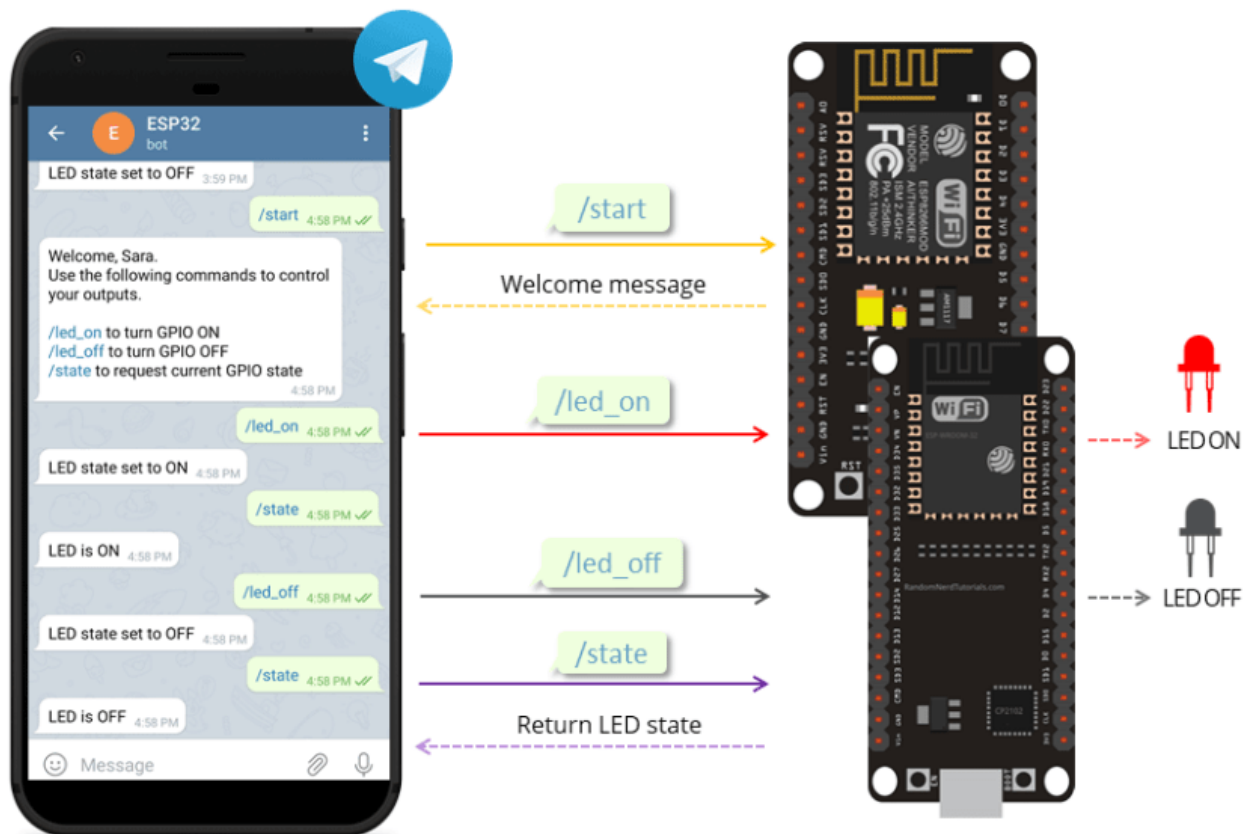


17. Ejemplo de un bot (Telegram)

Los *bots* son aplicaciones multiplataforma de terceros totalmente automatizadas, con una interfaz textual, integradas en el propio Telegram. Se controlan enviando mensajes con determinados comandos o instrucciones que hayan sido configurados en ellos. Normalmente, usando la instrucción */help* podremos obtener un listado de ellos.

Su configuración será descrita más adelante.





18. Comunicación entre Telegram y ESP32

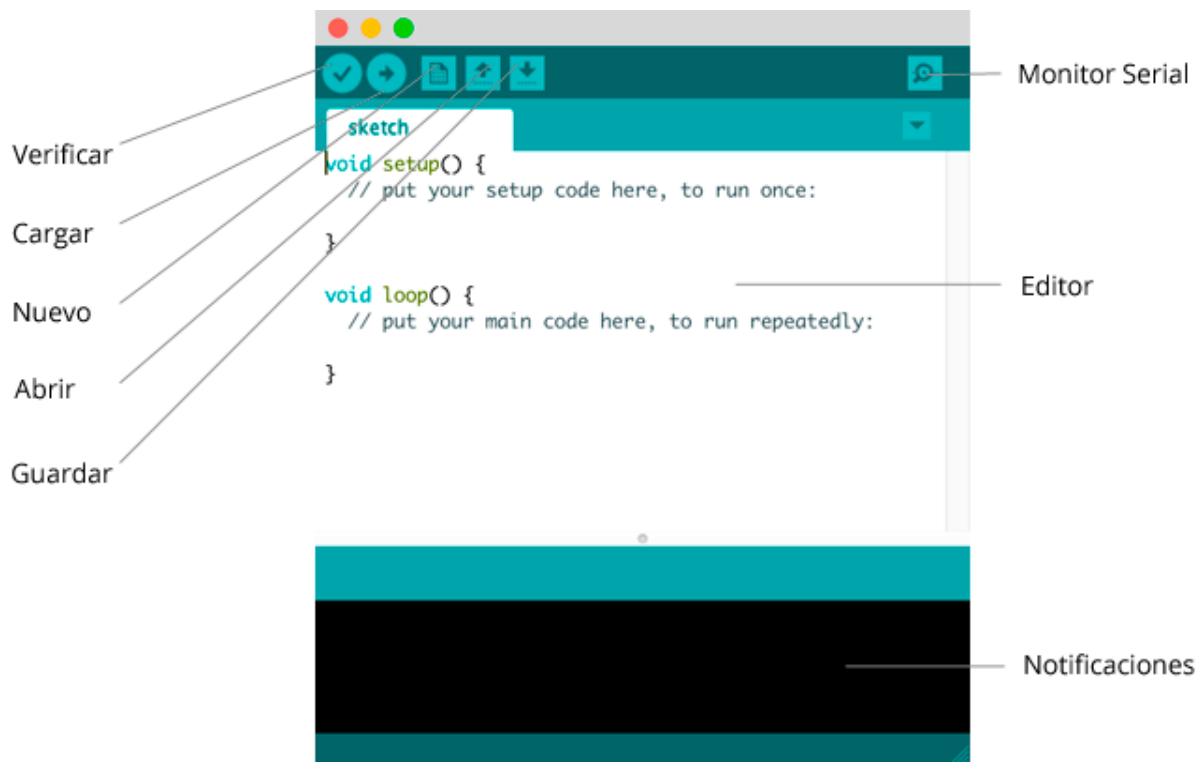
### Arduino-IDE

El entorno de desarrollo integrado (IDE) de Arduino es un programa multiplataforma compuesto por diferentes herramientas de programación. Generalmente se componen de:

- Editor de código.
- Compilador.
- Depurador.
- Constructor de interfaz gráfica.



En este caso concreto también se incorporan las herramientas para cargar el programa ya compilado en la memoria flash del hardware.



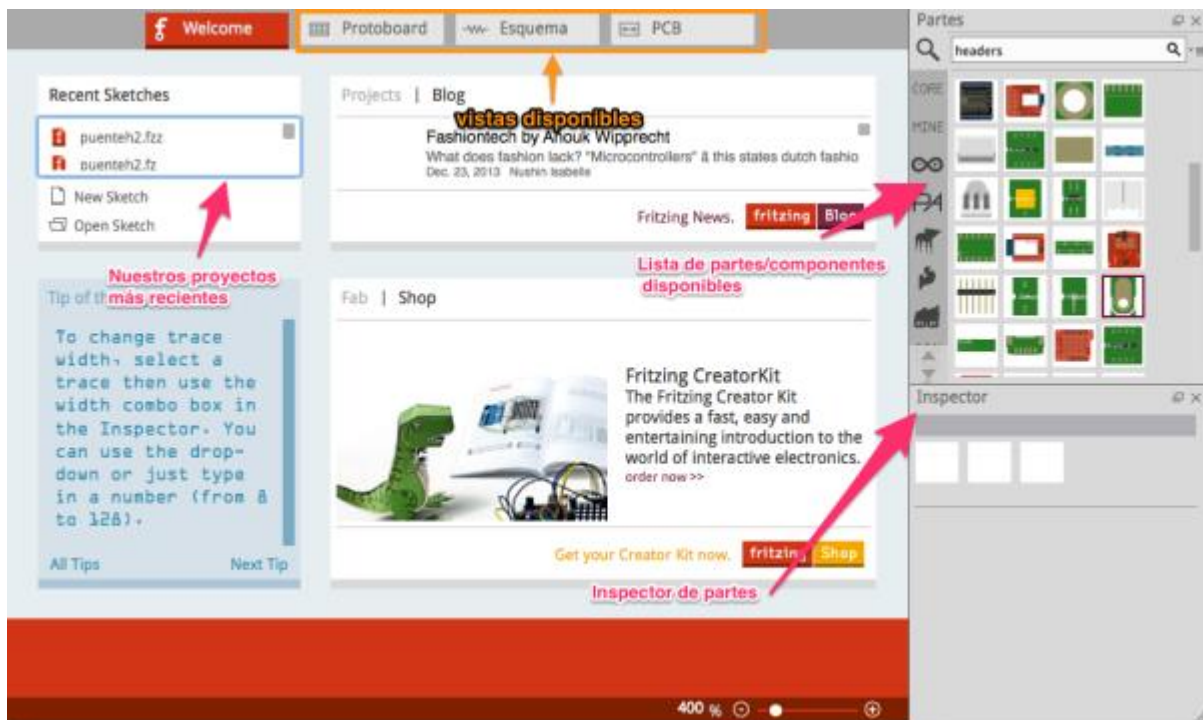
19. Arduino-IDE y sus funciones (Mac OS)

Su instalación es muy sencilla, descargamos el archivo ejecutable de la propia web de Arduino. Lo ejecutamos en nuestro ordenador e instalamos los componentes necesarios además de indicar el directorio donde lo queremos tener ubicado. La propia configuración de las placas y sus drivers será explicada más adelante.

## Fritzing

Creado como software libre (Open Source) bajo los principios de Processing y Arduino, Fritzing es un programa de diseño electrónico centrado en documentar prototipos basados en Arduino y

crear circuitos impresos para su posterior fabricación. Se puede instalar tanto en Windows, Linux o Mac; en mi caso el sistema operativo es Windows 10 (20H2).

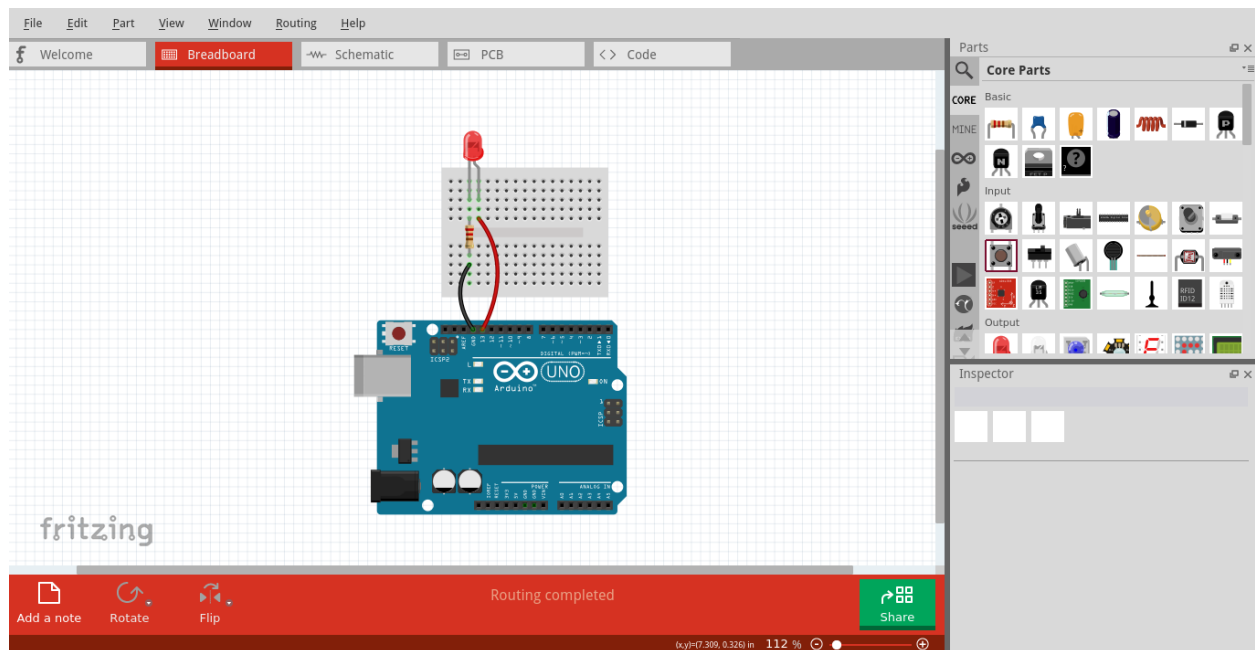


## 20. Página principal de Fritzing

Una vez abierto el programa podemos apreciar tres opciones de vista:

- Protoboard.
- Esquema.
- PCB.

La que más usaremos para las pruebas y planificación de este proyecto es la vista de protoboard, donde podemos agregar casi cualquier componente desde librerías dentro del propio programa o creadas por la comunidad que se pueden incorporar al mismo.



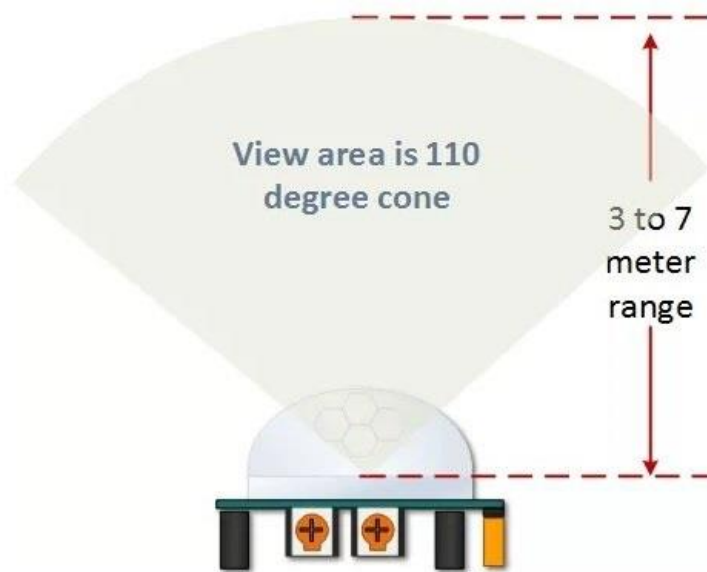
## 21. Vista de Protoboard

## 3 OBJETIVOS

### Explicación

En este proyecto pretendo lograr la creación de un sensor de movimiento conectado a la red mediante Wifi, que envíe alertas cuando se active el sensor a un dispositivo móvil, un ordenador o un reloj inteligente. También se ha de distinguir entre los falsos positivos, como las pisadas de un animal, y entre el positivo real, que será el movimiento de una persona promedio. Lo ideal sería colocar el sensor a la altura de la cintura, en el marco de una puerta o en alguna esquina superior de la sala donde queramos activar la detección de presencia para poder tener un correcto funcionamiento de la herramienta.

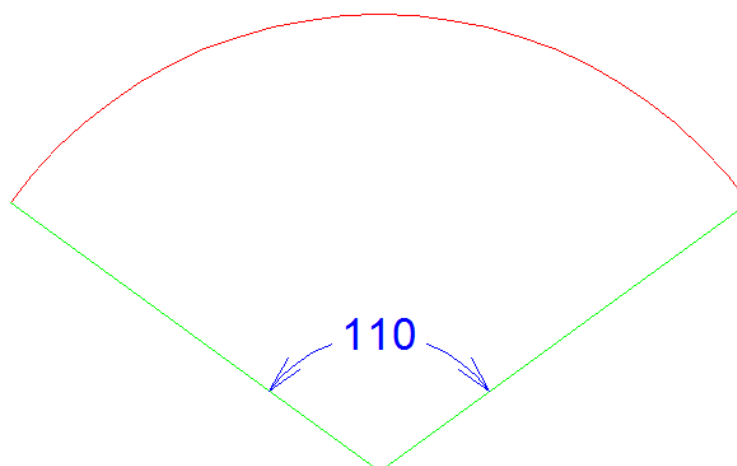
En la siguiente imagen observamos el rango que cubre nuestro sensor cuando el servomotor está en la posición de 180°:



#### 22. Rango de detección del HC-SR501

Si el sensor se instala en una zona fija nos cubrirá entre tres y siete metros de distancia con un ángulo efectivo de ciento diez grados. Lo ideal sería ponerle un servomotor con rotatorio para poder cubrir una mayor superficie con un solo sensor.

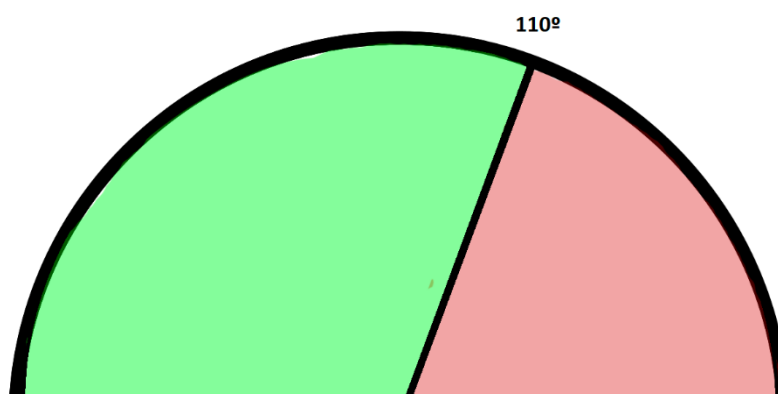
Incluir en el proyecto de detección de movimiento un pequeño servomotor llegó al encontrar un problema con el rango de cobertura del componente HC-SR501. La totalidad del ángulo cubierto por este sensor son  $110^\circ$  en una distancia de entre tres y siete metros, dependiendo del ajuste realizado, lo que no nos representa un mayor problema en estancias pequeñas pero en lugares más espaciosos esta angulosidad nos podría dar puntos ciegos que puedan ser usados para evadir al propio sensor. Aunque en entornos domésticos, sobre todo en interiores (exceptuando salones o estancias bastante espaciales), he observado después de diversas pruebas que colocando el sensor en una esquina de la habitación, manda alertas sin mayor problema.



23. Ángulo de 110º grados

Con todo esto dicho, se me ocurrió la posibilidad de rotar el HC-SR501 para cubrir una superficie de 180° y así evitar las zonas muertas del sensor en lo máximo posible. Aunque evitemos mayor cantidad de puntos ciegos, cuando el servomotor gire hasta 180° o esté en su posición 0° nos dejará un punto muerto de 70° durante unos segundos. Quizás esto último podría ser una falla de seguridad si el servomotor se llegara a parar o averiar en alguno de los extremos del recorrido.

En la siguiente imagen podemos apreciar con mayor claridad lo anteriormente citado:



24. Recorrido y punto ciego del servomotor

Observamos el arco superior en color azul, que haría referencia a la zona completa de detección del sensor. En la parte inferior izquierda podemos apreciar la zona en verde, que sería todo el rango efectivo de detección en ese momento (110°); y en la parte inferior derecha tenemos la zona roja, que nos indica el punto ciego (70°). Así observamos más gráficamente la desventaja del servomotor.

### Nivel de domotización

El nivel de domotización que añade el sensor de movimiento lo podemos calcular a partir de la Tabla de Niveles de Domotización, que es el nivel asignado a una instalación domótica como resultado entre la ponderación de diferentes dispositivos existentes y las aplicaciones domóticas

cubiertas. Se divide en tres bloques; la primera columna muestra las diferentes soluciones a cada aplicación domótica, que se reconocen entre ellas por su color, además, cada solución tiene una importancia relativa distinta que se calcula de forma automática en la segunda columna siguiendo la especificación EA 0026.

En la columna de valoración se reflejan las características de la instalación a evaluar, una vez cumplimentada, sumamos los puntos obtenidos. En función del resultado obtendremos uno de los siguientes niveles de domotización:

- Nivel 1: instalaciones con un nivel mínimo de dispositivos y/o aplicaciones domóticas. La suma de procesos ponderados debe ser por lo menos de trece puntos. Esto debe asegurarse con dispositivos repartidos entre tres aplicaciones distintas.
- Nivel 2: instalaciones con un nivel medio de dispositivos y/o aplicaciones domóticas, en este caso, la suma de puntos debe ser de treinta como mínimo y deben cubrirse al menos tres aplicaciones.
- Nivel 3: instalaciones con un nivel alto de dispositivos y/o aplicaciones domóticas donde la suma de puntos debe ser de cuarenta y cinco como mínimo, debiendo estar repartidos entre seis aplicaciones.

Estos dispositivos son los *necesarios*, que hacen posible la aplicación domótica.

En nuestro caso, usaremos la tabla para determinar la puntuación que nos da nuestro sensor de movimiento, marcando en la primera fila *1 por estancia*, pudiendo así hacernos una idea en caso de tener uno en cada habitación de nuestra vivienda.

Tabla de Niveles de Domotización	
Dispositivos	Nº de dispositivos o condición
Detectores de presencia	<input type="radio"/> Ninguno <input type="radio"/> 2 <input type="radio"/> 1 cada 20 m2 <input checked="" type="radio"/> 1 por estancia
Teclado codificado, llave electrónica, o equivalente.	<input checked="" type="radio"/> Ninguno <input type="radio"/> 1
Sirena interior	<input checked="" type="radio"/> No <input type="radio"/> Si
Contactos de ventana y/o impactos	<input checked="" type="radio"/> No <input type="radio"/> En puntos de fácil acceso <input type="radio"/> En todas las ventanas
Sistema de mantenimiento de alimentación en caso de fallo de suministro eléctrico	<input checked="" type="radio"/> No <input type="radio"/> Si
Módulo de habla/escucha, destinado a la escucha en caso de alarma* También se admite cualquier tipo de control que permita conocer si realmente existe un intruso (cámaras web...)	<input checked="" type="radio"/> No <input type="radio"/> Si
Sistema conectable con central de alarmas	<input checked="" type="radio"/> No <input type="radio"/> Si
<b>Suma Parcial Alarma de intrusión</b>	<input type="text" value="3"/>
Detectores de inundación necesarios en zonas húmedas (baños, cocina, lavadero, garaje)	<input checked="" type="radio"/> No <input type="radio"/> Los necesarios <sup>1)</sup>
Electro válvula de corte agua con instalación para "bypass" manual.	<input checked="" type="radio"/> No <input type="radio"/> Las necesarias <sup>1)</sup>
Detectores de concentraciones de gas butano y/o natural en zonas donde se prevea que habrá elementos que funcionen con gas	<input checked="" type="radio"/> No <input type="radio"/> Los necesarios <sup>1)</sup>
Electro válvula de corte gas con instalación para "bypass" manual	<input checked="" type="radio"/> No <input type="radio"/> Las necesarias <sup>1)</sup>
Detector de incendios	<input checked="" type="radio"/> No <input type="radio"/> 1 en cocina. <input type="radio"/> 1 cada 30 m2 <input type="radio"/> En todas las estancias
<b>Suma Parcial Alarmas técnicas</b>	<input type="text" value="0"/>

Simulación de presencia	<input checked="" type="radio"/> No <input type="radio"/> Relacionada con las persianas motorizadas o con puntos de luz. <input type="radio"/> Relacionada con persianas motorizadas y con puntos de luz
<b>Suma Parcial Simulación de presencia</b>	<input type="text" value="0"/>
Videoportero	<input checked="" type="radio"/> No <input type="radio"/> Si
<b>Suma Parcial Videoportero</b>	<input type="text" value="0"/>
Control de persianas	<input checked="" type="radio"/> No <input type="radio"/> Todas las de superficie superior a 2m2 <input type="radio"/> Todas
<b>Suma Parcial Control de persianas</b>	<input type="text" value="0"/>
Regulación luminica con control de escenas	<input checked="" type="radio"/> No <input type="radio"/> en dependencias dedicadas al ocio <input type="radio"/> En salón y dormitorios
En jardín o grandes terrazas mediante interruptor crepuscular o interruptor horario astronómico	<input checked="" type="radio"/> No <input type="radio"/> Si
Conexión/desconexión general de iluminación	<input checked="" type="radio"/> No <input type="radio"/> Un acceso <input type="radio"/> Todos los accesos
Control de puntos de luz y tomas de corriente más significativas	<input checked="" type="radio"/> No <input type="radio"/> 50% puntos luz <input type="radio"/> 80% puntos luz + 20% tomas corriente
<b>Suma Parcial Control de iluminación</b>	<input type="text" value="0"/>
Cronotermostato	<input checked="" type="radio"/> No <input type="radio"/> 1 en salón <input type="radio"/> zonificando la vivienda en un mínimo de dos zonas <input type="radio"/> Varios cronotermostatos, zonificando la vivienda por estancias
<b>Suma Parcial Control de clima</b>	<input type="text" value="0"/>



Posibilidad de realizar programaciones horarias sobre los equipos controlados	<input checked="" type="radio"/> No <input type="radio"/> Si
Gestor energético	<input checked="" type="radio"/> No <input type="radio"/> Si
<b>Suma Parcial Programaciones</b>	<input type="text" value="0"/>
Consola o equivalente	<input checked="" type="radio"/> No <input type="radio"/> Si
Control telefónico bidireccional	<input checked="" type="radio"/> No <input type="radio"/> Si <input type="radio"/> Interacción mediante SMS
Equipo para control a través de internet, WAP o equivalente	<input checked="" type="radio"/> No <input type="radio"/> Si
<b>Suma Parcial Interfaz usuario</b>	<input type="text" value="0"/>
Dispositivos conectables a empresas suministradoras a través de redes de comunicación	<input checked="" type="radio"/> 0 <input type="radio"/> 1 <input type="radio"/> 2 <input type="radio"/> 3 o más
<b>Suma Parcial Dispositivos conectables a empresas suministradoras</b>	<input type="text" value="0"/>

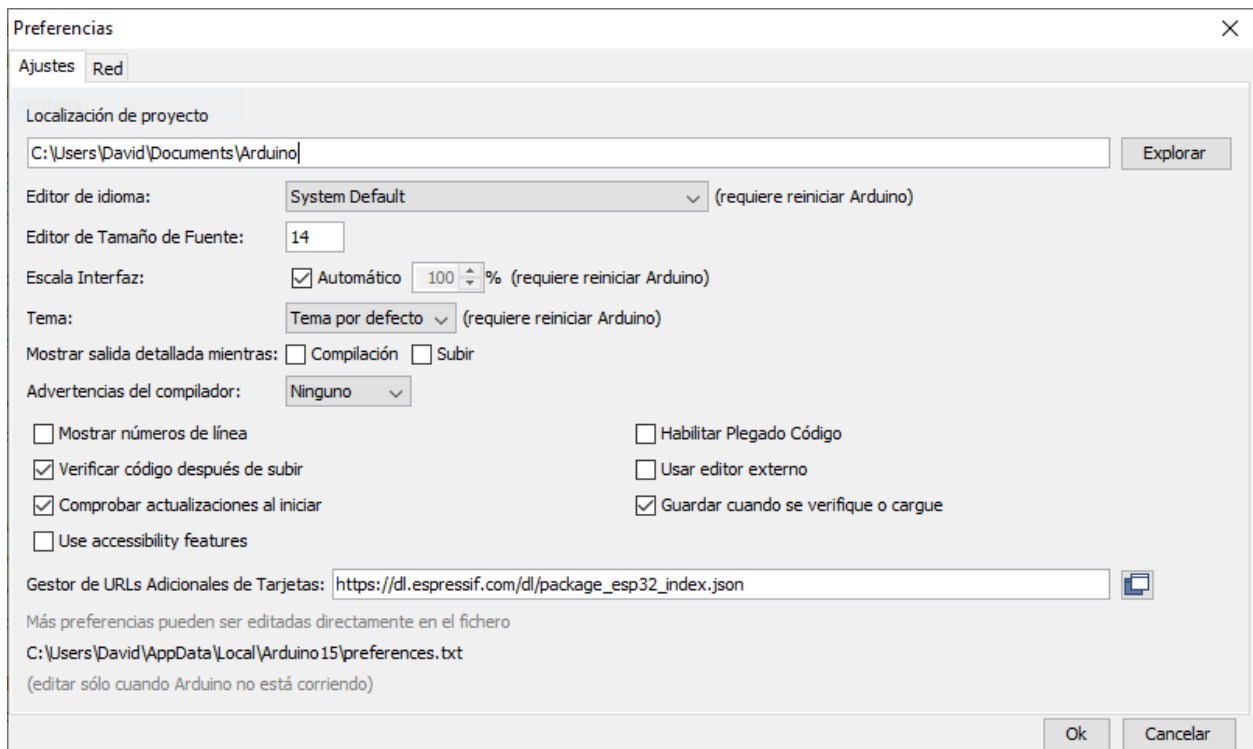
Tomas SAT y Tomas Multimedia	<input checked="" type="radio"/> No <input type="radio"/> 3 tomas satélite + 3 tomas multimedia <input type="radio"/> 3 tomas satélite +1 toma multimedia en todas las estancias, incluido terraza
Punto de acceso inalámbrico	<input checked="" type="radio"/> No <input type="radio"/> Wi-Fi
<b>Suma Parcial Red Multimedia</b>	<input type="text" value="0"/>
<b>SUMA TOTAL</b>	<input type="text" value="3"/>
<b>Número de aplicaciones domóticas cubiertas<sup>2)</sup></b>	

La suma parcial en alarma de intrusión nos da una puntuación de tres, por lo cual, si tuvieras un sensor por cada habitación la suma total de esto sería también de tres en la propia tabla de domótica. Con esto podemos deducir que este sensor por sí solo no nos aporta una gran cantidad de domotización en nuestra vivienda.

### Configuración de la placa ESP32 en Arduino-IDE

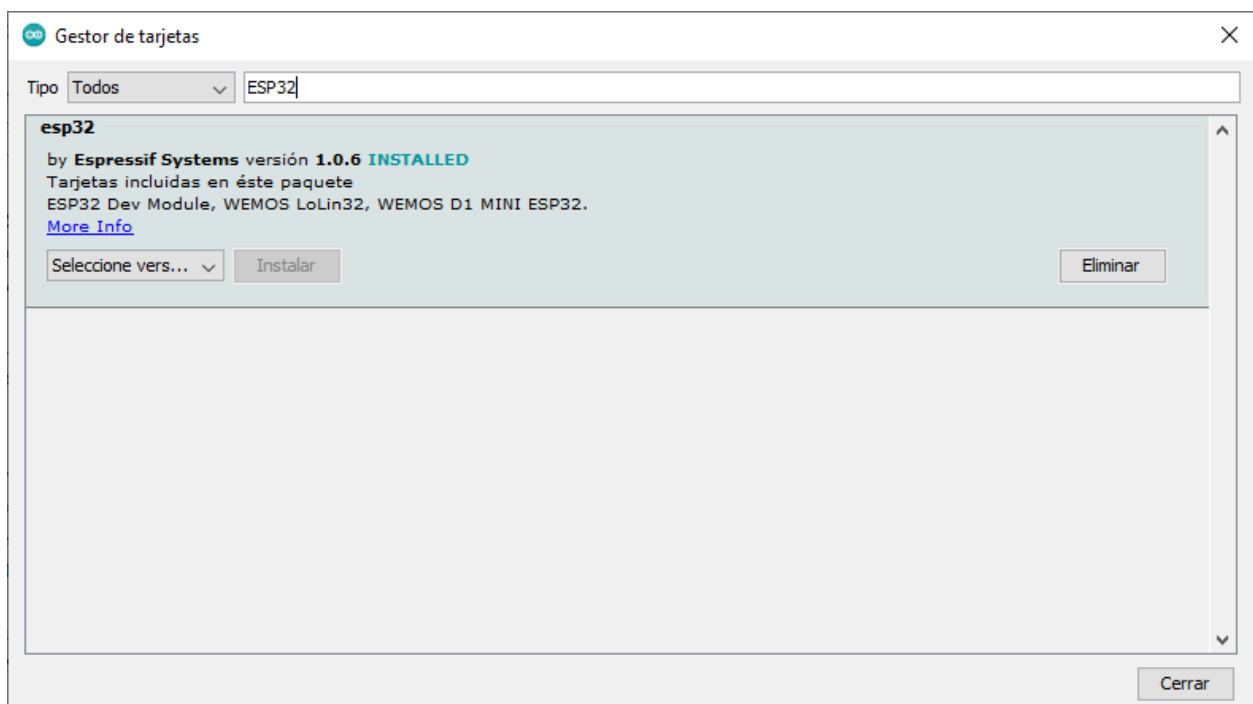
Iniciamos el procesador de código y nos dirigimos al menú “Archivo/Preferencias”. En el Gestor de URLs adicionales de Tarjetas copiamos el siguiente enlace:

[https://dl.espressif.com/dl/package\\_esp32\\_index.json](https://dl.espressif.com/dl/package_esp32_index.json)



### 25 Menú "preferencias" de Arduino-IDE

Después, nos dirigimos a “Herramientas/Placa” y buscamos la opción “Gestor de tarjetas”. Nos abre una ventana donde debemos buscar “ESP32”. La instalamos y esperamos a que finalice el proceso.



### 26. Librería necesaria para el funcionamiento de la placa ESP32

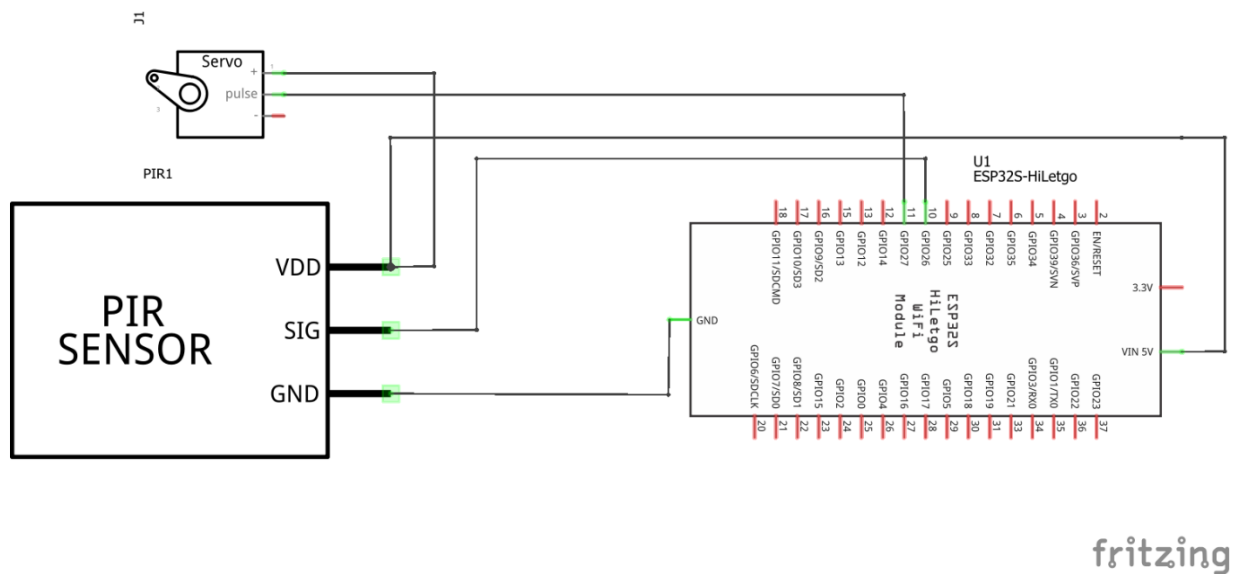
En ese mismo menú, buscamos ya nuestra placa, que estará nombrada como “ESP32 Dev Module”.

Una vez seleccionada, la placa es detectada y funciona perfectamente cuando le queremos subir algún código.

Es una configuración difícil y diferente, ya que no es una placa Arduino ni Genuino y necesitamos de librerías de la comunidad ESP32 para poder probar y cargar código desde este IDE.

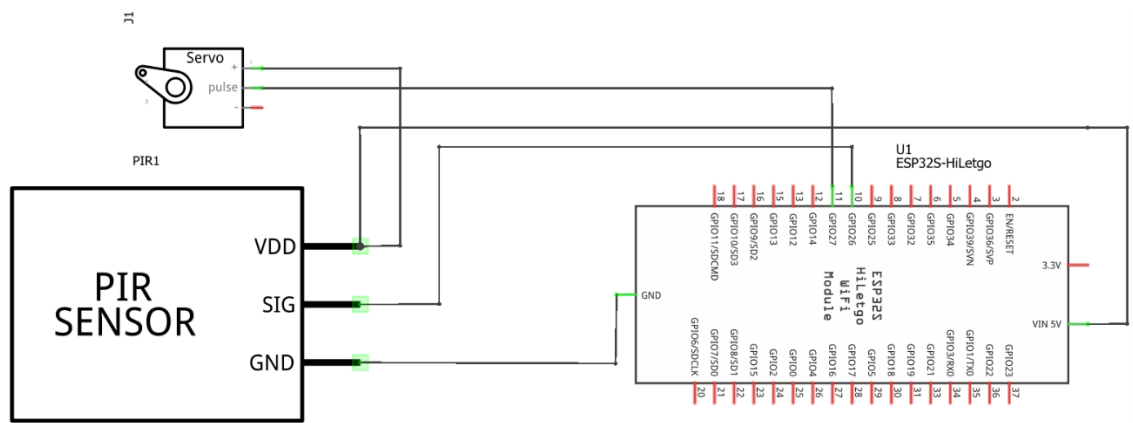
### Montaje de los diversos componentes

Para montar los diferentes componentes del proyecto usamos los pines 26 y 27. El pin 26 es usamos para alimentar enviar las ordenes órdenes al servomotor y el pin 27 es el que enviara las ordenes de detección del sensor PIR hasta la placa ESP32. En la siguiente imagen tenemos una vista esquemática de los componentes conectados entre sí:



27. Esquemático del conjunto

Antes de montarlo físicamente se han hecho varias pruebas en simulador (sin necesidad de protoboard) y se ha podido observar que le código del sensor PIR y el servomotor subido conjuntamente a la placa ESP32 ocupan casi la totalidad del espacio, por lo que se ha considerado la posibilidad de incluir un Arduino Nano para ejecutar el servomotor y de esa manera no cargar mucho la placa principal en este prototipo.

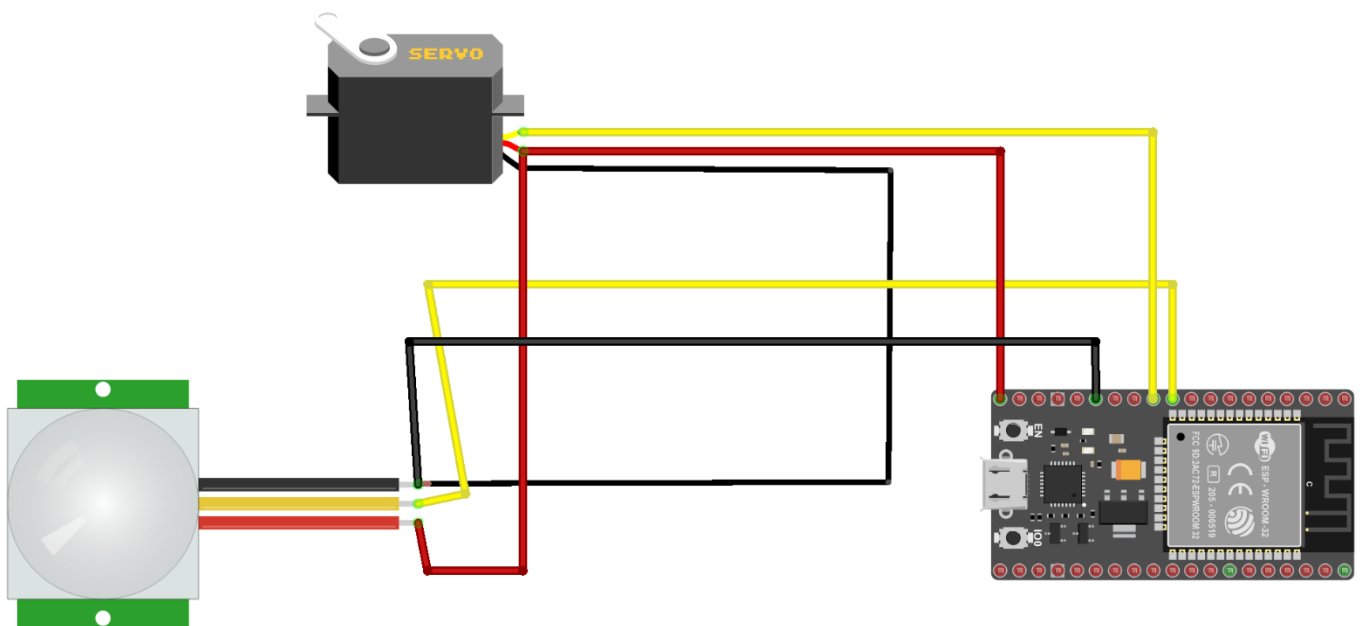


fritzing

### 28. Esquemático de los componentes

Cuando hice algunas pruebas con los componentes, intenté alimentar el sensor PIR y el servomotor con la salida de 3.3V, pero fue imposible dado que los dos componentes requieren una tensión de alimentación de 5V.

En la siguiente imagen tenemos como se vería el montaje del cableado (sin placa protoboard)



fritzing

### 29. Esquemático de los componentes

Por último, y gracias a que el programa Fritzing nos da esta utilidad se ha obtenido el diseño en PCB (Printed Circuit Board o placa de circuito impreso), que es una superficie no conductora donde podríamos implementar pistas o buses que sustituyeran al cableado y así tener un diseño mas simplificado y con un aspecto más profesional.



## 5 DESCRIPCIÓN DEL DESARROLLO DEL PROYECTO

La secuencia lógica para el logro del cometido es la siguiente:

### Listado y obtención de materiales a utilizar:

Empecé por lo básico, investigando que materiales serían necesarios para el correcto funcionamiento del propio proyecto. Investigue el mundo de Arduino y sus diversas placas, hasta dar con un módulo que sería ideal para este cometido. Después, busque diversos sensores de movimiento hasta dar con uno regulable y algo más avanzado para poder distinguir diferentes segmentos y no dar falsos positivos. Luego solo hacía falta un sitio para implementar las conexiones, un conjunto de cables y un ordenador para poder compilar el código.

### Montaje en simulador para las primeras pruebas.

Usando el simulador *Fritzing* y diversas librerías descargadas de internet, además de componentes creados por la comunidad, pude ver que placa y que sensores se podían ajustar más al desarrollo eficiente del trabajo.

### Pruebas y desarrollo en las siguientes placas:

i) Arduino Uno.

Descartada por tamaño excesivo.

ii) Arduino Nano.

Descartada por falta de módulo Wifi integrado.

iii) ESP32

Elegida para el desarrollo final por sus pequeñas dimensiones y su integración de un módulo dual Wifi/Bluetooth.

### Desarrollo del código en Arduino-IDE (C ++):

He usado este entorno de programación debido a que es bastante amigable, proveyendo de ejemplos y facilidad para añadir librerías y placas de desarrolladores externos. Aunque su desventaja principal es la velocidad, esto no me presenta mayor problema dado que el código no es demasiado extenso ni demasiado complejo.

También he experimentado problemas en un principio con el módulo de *Az-Delivery* ya que debemos configurar en programa y algunas librerías externas para su correcto funcionamiento. Todo esto será explicado con detalle a continuación:

### Montaje físico y pruebas del código:

#### Pruebas finales y retoques:

#### Montaje de todo el conjunto:

Una vez terminado todo el proceso de pruebas, procedemos al montaje final de todo el conjunto. No deberían apreciarse errores o fallos.

## 6 TEMPORIZACION DEL PROYECTO

1. Listado y obtención de materiales a utilizar:  
2 días
2. Montaje en simulador para las primeras pruebas.  
5 días

3. Pruebas y desarrollo en diversas placas  
6 días
4. Desarrollo del código en:
  - i) Arduino-IDE (C++)
 2 semanas y 2 días.
5. Montaje físico y pruebas del código.  
1 semana
6. Montaje físico y pruebas del código.  
1 semana
7. Pruebas finales.  
1 semana.
8. Montaje de todo el conjunto.  
3 días.
9. Desarrollo de los documentos explicativos y la presentación  
Durante todo el proyecto

TAREAS	FECHA DE INICIO	FECHA FINAL	DÍAS
Listado y obtención de materiales a utilizar	15-02-21	16-02-21	2
Montaje en simulador para las primeras pruebas	17-02-21	21-02-21	5
Pruebas y desarrollo en diversas placas	07-03-15	12-03-15	6
Desarrollo del código en Arduino-IDE	23-02-21	10-03-21	16
Montaje físico y pruebas del código	22-01-21	29-01-21	7
Pruebas finales	07-04-21	14-04-21	7
Montaje de todo el conjunto	23-04-21	28-04-21	6
Desarrollo de documentos explicativos	15-03-21	11-06-21	52

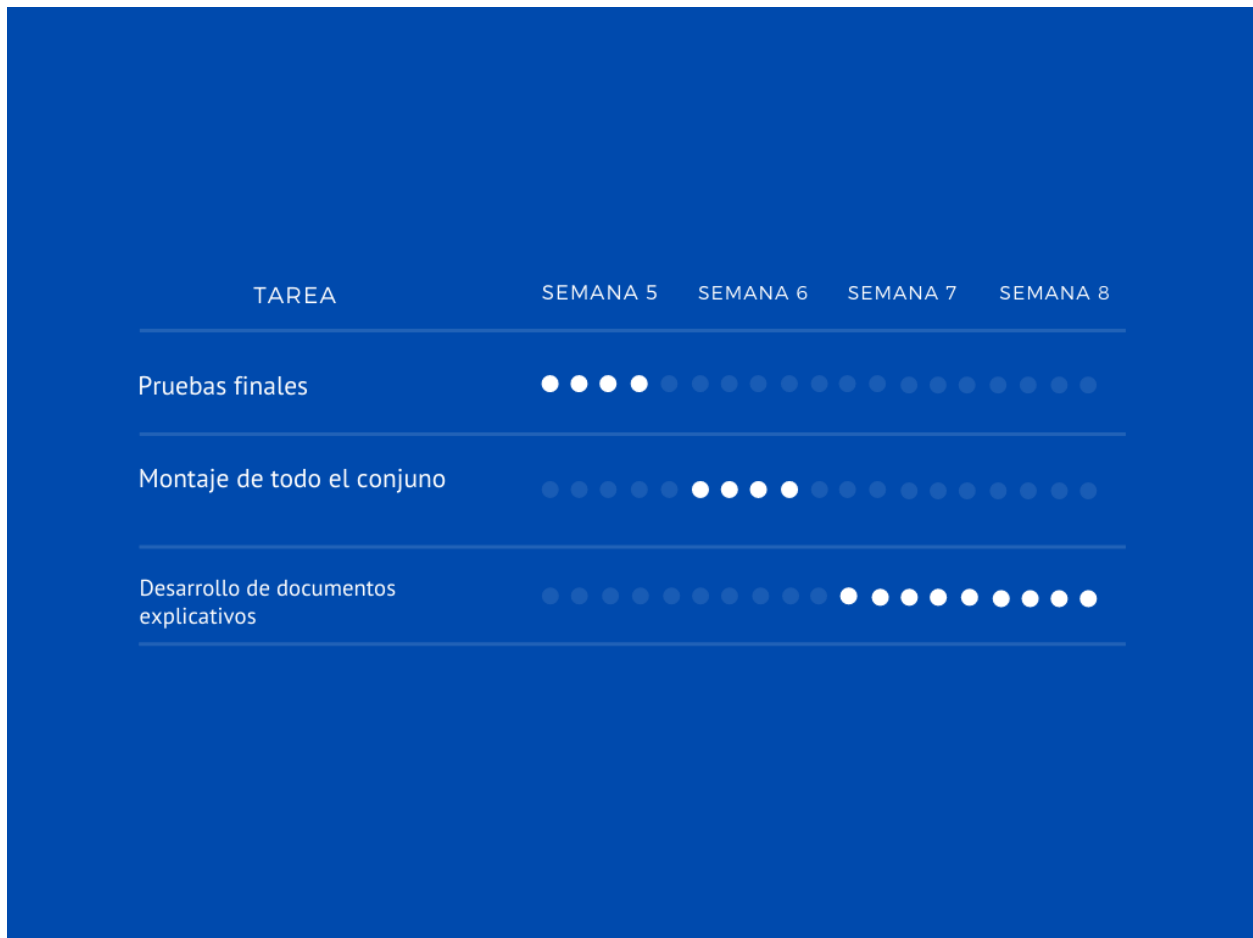
Se expondrá en un diagrama de gantt, que nos indica de una forma mas visual una vistad de las tareas programadas- Aquí se explican las fechas de inicio y finalizacion del proyecto, todas las tareas desarrolladas, una estimacion de cada una de ellas y como se superponen y/o si hay una relñacion entre ellas. Desglosamos las multiples tareas y lineas temporales en una vista general única.

Gracias a esto, podemos ver con claridad y tener una vista general simplificada de todo el desarrollo del proyecto, ademas de poder extraer datos sobre el rendimeiento del mismo.

Estas son las diferentes fases que ha atravesado el proyecto durante su realización:

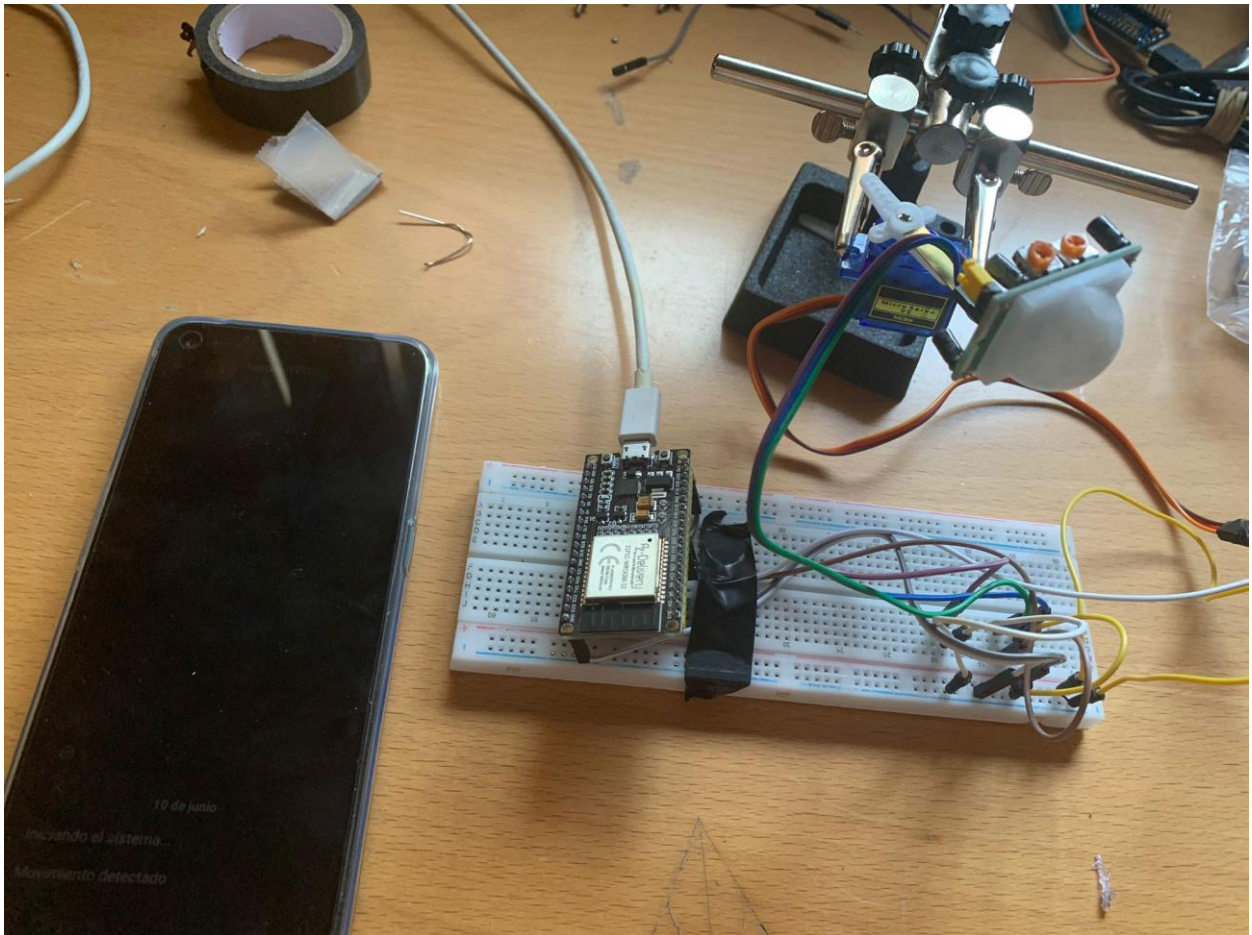






## 7 RESULTADOS Y ANALISIS

Los resultados obtenidos en este proyecto han sido muy favorables, debido a que el diseño ha funcionado a la casi a la perfección. En la siguiente imagen vemos el conjunto de todos los componentes montados y con las conexiones realizadas a través de una protoboard, evitando realizar soldaduras.



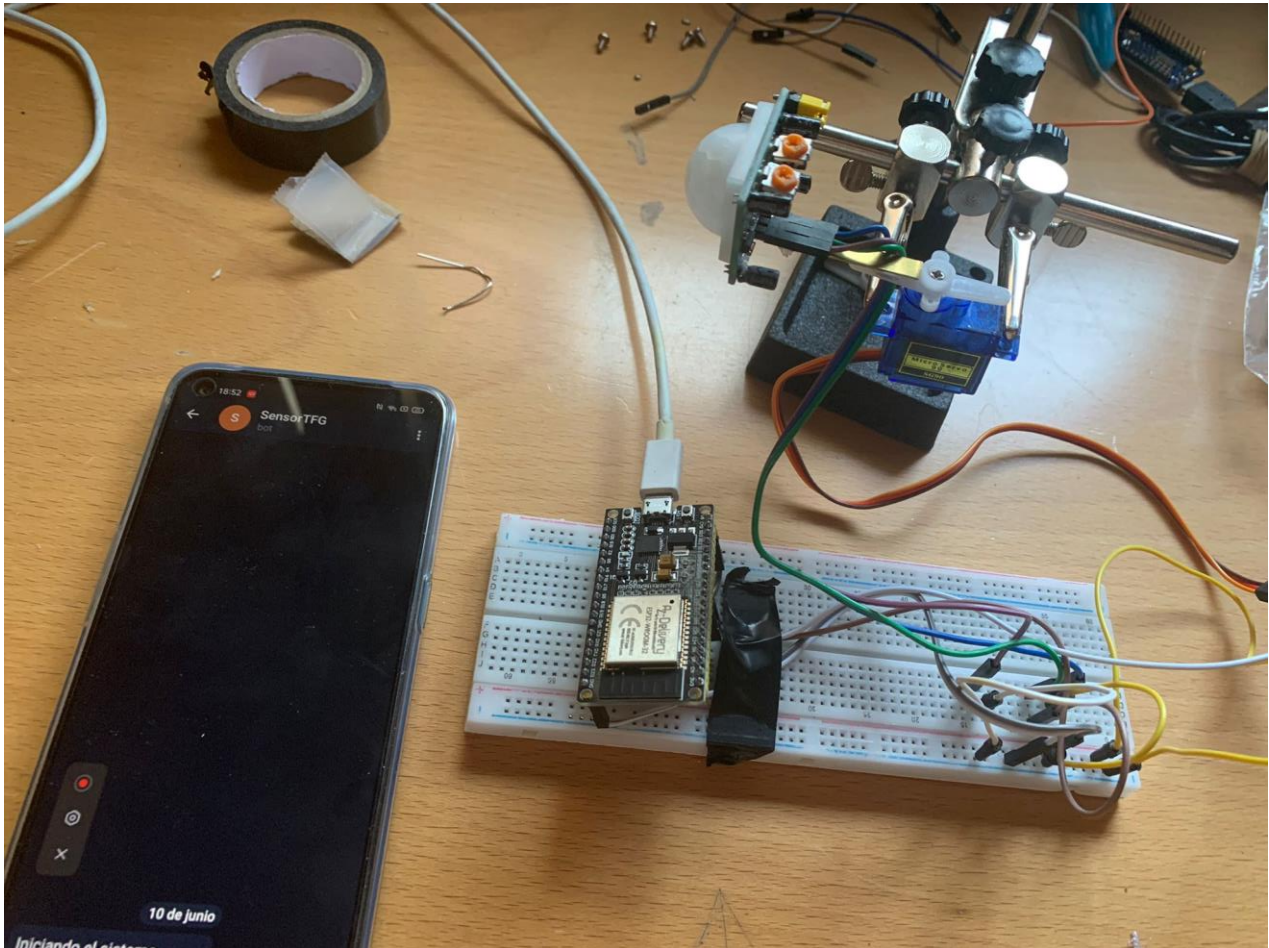
31. Imagen 1 del diseño montado

Podemos observar el conjunto montado en la placa protoboard.

El sensor PIR y el servomotor esta siendo sujetado por unos soportes dado que, por su peso, se caía hacia los lados y a habido que adaptar un trozo de metal al propio servo para que este mantenga en una posición correcta el sensor PIR.

Una vez conectado, el servomotor se activará y en la aplicación Telegram nos aparece un mensaje diciendo *Iniciando el sistema...* Segundos después, todo estará activado. Si el sensor PIR llegara a detectar algún tipo de movimiento el ESP32 nos enviara un mensaje a nuestro chat de *Movimiento detectado*

Se ha observado que el sensor, una vez detectado un objeto, no lo volverá a detectar hasta que este se cruce de nuevo con el sensor. Es decir, si el sensor PIR estuviera quieto le sería más difícil ejercer su función en este ámbito, de ahí que se haya incluido un servomotor para hacerlo girar.



32. Imagen 2 del diseño montado

He podido observar varios defectos del diseño, como la necesidad de un soporte para sujetar el servo y el sensor PIR, dado que por sí solo, este tiene desajustes en el peso y cae hacia cualquier lado al realizar el giro de 180°.

También, las conexiones con la protoboard no son muy seguras, dado que con algún movimiento brusco o el traslado del diseño los cables se pueden salir y entorpece el buen funcionamiento de los componentes. En las [líneas de investigación futuras](#) se tratara este apartado en más profundidad.

Las virtudes del diseño son la escasez de capital necesario para realizarlo, dado que por menos de veinte euros se pueden conseguir todos los componentes. Por otra parte, también hay una reducida cantidad de cables, dado que necesitamos únicamente seis para realizar las conexiones entre dispositivos y otros cuatro para las conexiones en la protoboard.

En cuanto a la conexión Wifi entre la placa y la aplicación Telegram; se conectan gracias a que la placa puede estar en modo *Wifi Station (Estación de Wifi)*, que es el modo por defecto de la placa donde se conecta a un punto de acceso. Para ello necesitamos añadir las librerías correspondientes. También se ha observado que el bot, al iniciar, se demora unos dos minutos hasta realizar toda la configuración principal.

Otro problema que podría surgir a largo plazo es el mantenimiento algunas librerías, dado que están creadas por la comunidad, depende de ellos un mantenimiento continuado o dejarlas de actualizar en cualquier momento. (*UniversalTelegramBot.h*, *ArduinoJson.h*).

La alimentación del conjunto se puede hacer mediante una conexión con cable USB directamente a un ordenador o con una toma de pared con entrada USB a la red eléctrica doméstica, lo cual facilita su funcionamiento en cualquier lugar sin necesidad de un equipo.

En el siguiente enlace podemos observar un video subido a la plataforma de YouTube donde demuestro el funcionamiento del proyecto:

<https://www.youtube.com/watch?v=Y8OCt28EY1s>

También todo este proyecto ha sido subido a la plataforma de GitHub se puede encontrar en el siguiente enlace:

<https://github.com/daviddvp/Proyecto-SensorPIR-ESP32>

## 8 CONCLUSIONES

Este proyecto sería de una gran utilidad en domótica o equipos de seguridad, pero el principal problema para su desarrollo a gran escala es la necesidad de conocer el entorno Arduino y de desarrollo. Se detectan grandes problemas cuando un usuario promedio debe proceder a su administración o reparación, e incluso al meter las credenciales de red necesitamos tener la interfaz Arduino para hacerlo.

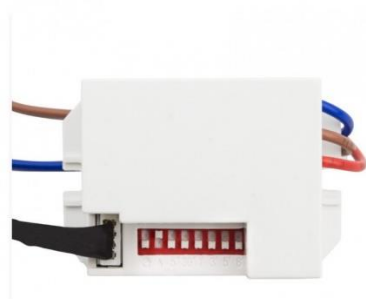
La configuración o administración de este sensor de movimiento si podría ser realizada por algún con conocimientos de Arduino

## 9 LINEAS DE INVESTIGACIÓN FUTURAS

Las líneas de investigación futuras del proyecto las dividiré en varios apartados

### **Miniaturización del proyecto.**

Para la miniaturización del proyecto eliminaría el uso de una placa protoboard y realizaría la conexión de los cables mediante soldadura a la propia placa y la los componentes. Los sensores utilizados no serían para prototipos Arduino, sino algo más genérico, como un *Detector de Movimiento PIR 120° Mini*.



34. Imagen 2 de sensor



33: Imagen 1 de sensor

Después, no usaríamos la placa completa del microcontrolador ESP32 sino únicamente el chip  
Sensor de movimiento



donde, en sus laterales, soldaríamos las conexiones, ya sea de manera tradicional o mediante micro soldaduras.



35. Chip ESP32

El conjunto se montaría alrededor de una carcasa protectora para evitar su deterioro frente a las inclemencias temporales. También se debería valorar el uso o no del servomotor.

### Creación de modelo con placa PCB

En este apartado se hace enfoque en la eliminación de los cables, dado que pueden desconectarse al no estar soldados o al realizar un movimiento brusco con el diseño.

Tomando como ejemplo la siguiente página nos basaremos en sus criterios para evaluar el posible desarrollo de nuestra placa PCB

#### [Fabricación de PCB - Prototipos de PCB de forma sencilla \(pcbway.es\)](https://pcbway.es/)

Para ellos necesitaríamos tener en cuenta las dimensiones, la cantidad de piezas, sus capas y el grosor.

Tipo de placa: ☐ Piezas sueltas ☒ Panel del cliente ☐ Panel creado por PCBWay

Tolerar X-out en el panel: ☒ Aceptar ☐ No aceptar

Diseños diferentes en el panel: 1 2 3 ☒ 4 5 6  p.e.

\* Dimensiones (paneles): 12000 X 12000 mm

\* Cantidad (paneles): 5 paneles

Capas: ☐ 1 Capa ☒ 2 Capas ☐ 4 Capas ☐ 6 Capas ☐ 8 Capas ☐ 10 Capas ☐ 12 Capas ☐ 14 Capas

Material: ☒ FR-4 ☐ Aluminio ☐ Rogers ☐ HDI (Vías enterradas / ligadas) ≥4 Capas

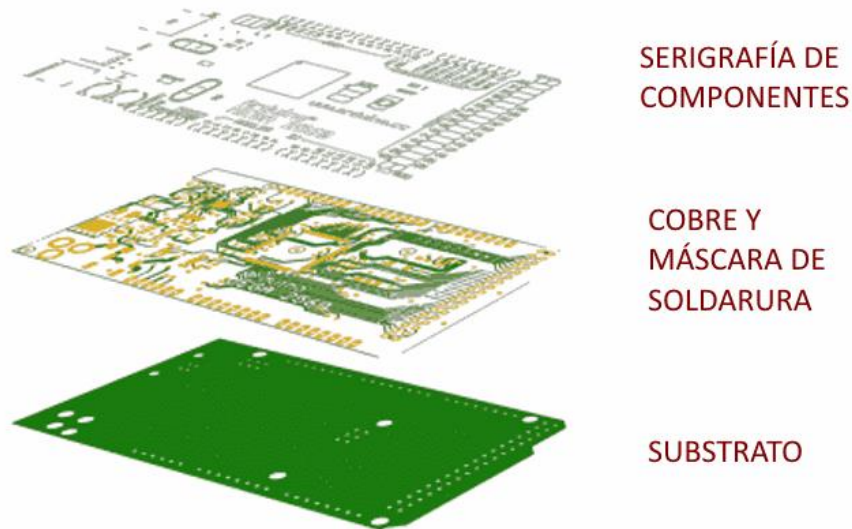
☐ Base de Cobre

\*El modelo de material se puede elegir a continuación. HDI está disponible para 4 capas o más.

### 36. Creación de placa PCB

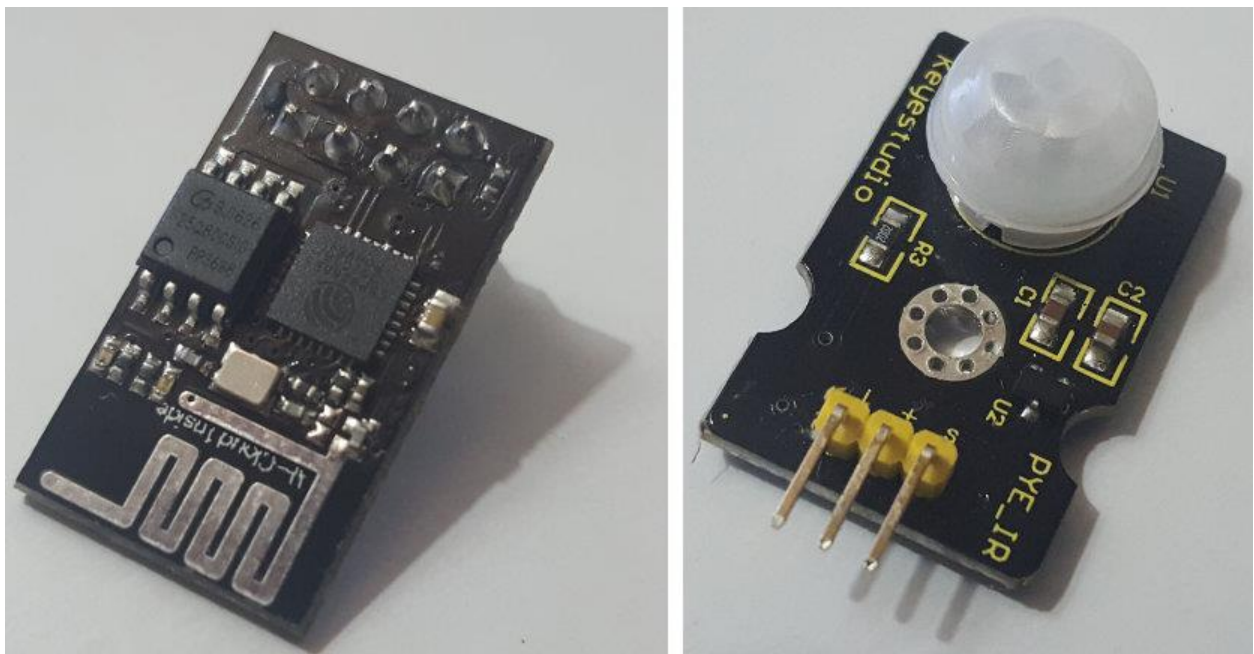
En la imagen anterior podemos ver la configuración básica del PCB. Elegimos dos capas para que no se nos crucen las conexiones de los cables como se ha visto en el [Esquemático de PCB](#).

En esta placa PCB integraremos únicamente el chip ESP32, sin los pines, dado que irían directamente soldados al propio chip, y el sensor PIR ya incluir en servomotor en la propia placa de circuito impreso sería físicamente incómodo.



### 37. Partes de una PCB

Para introducir el servomotor dentro del conjunto podríamos usar una placa como la siguiente junto con componentes SMD:



### 38. PCB con sensor PIR y ESP32 integrados

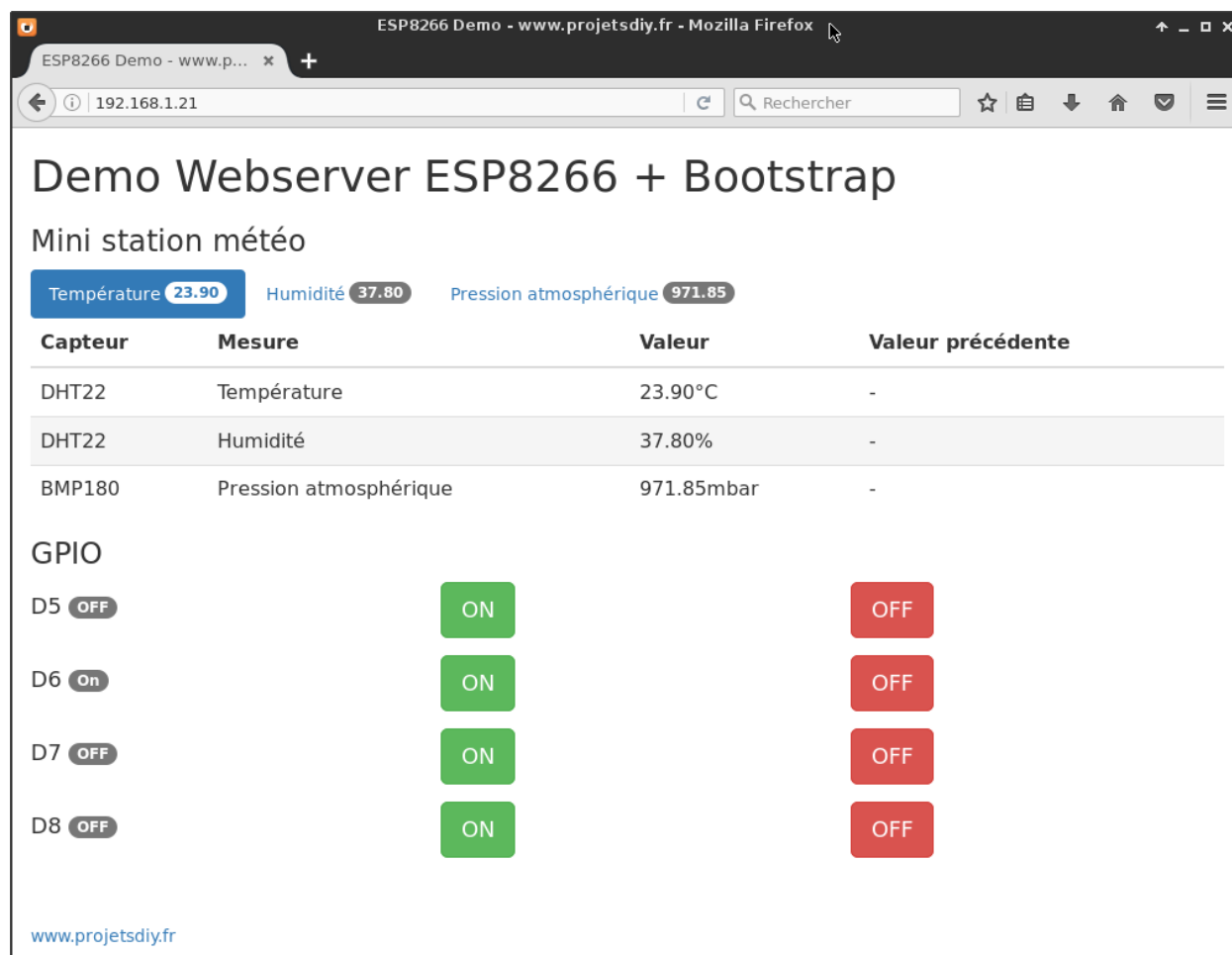
Conectaríamos el servomotor a uno de los pines y los de alimentación deberían ser soldados a la placa para posteriormente llevar únicamente dos a una batería para proporcionarle corriente a todos los componentes a la vez.

### Desarrollo de una interfaz web y/o aplicación para dispositivos móviles

El objetivo de esta línea de investigación sería el desarrollo de algún tipo de interfaz desde donde controlar el propio sensor, ya sea para controlar la activación o no de la rotación, o para recibir las alertas de detección de movimiento.

Esto daría una mayor facilidad para administrar las incidencias desde cualquier dispositivo y desde cualquier lugar con conexión a internet ya que esta interfaz web estaría en un servidor de internet accesible desde cualquier parte.

Obtendríamos un mayor control sobre el dispositivo y mayor facilidad para su configuración.



39. Ejemplo de interfaz web

### Implementación en el mercado.

En este apartado he pensado en su posible venta en el mercado de la domótica. La razón principal de todo esto viene a raíz de la idea de un desarrollo posterior de dos modelos, uno con servomotor para estancias más espaciales y otro sin servomotor para habitaciones más pequeñas.

He desarrollado dos tablas de costes dependiendo de cada producto, donde el más barato sería el que tiene ausencia de servomotor, obviamente por incluir un componente menos.

Sensor sin servomotor		
Componentes	Enlace	Precio
ESP32	<a href="https://n9.cl/0air">https://n9.cl/0air</a>	1.67 €
Sensor PIR 180 grados	<a href="https://n9.cl/4qjqo">https://n9.cl/4qjqo</a>	3.95 €
Cable con 4 salidas	<a href="https://n9.cl/ytdr4">https://n9.cl/ytdr4</a>	3.02 €
Caja PVC para los componentes	<a href="https://n9.cl/30nlx">https://n9.cl/30nlx</a>	1.61 €
Total		10.25 €

Sensor con servomotor		
Componentes	Enlace	Precio
ESP32	<a href="https://n9.cl/0air">https://n9.cl/0air</a>	1.67 €
Sensor PIR 180 grados	<a href="https://n9.cl/4qjqo">https://n9.cl/4qjqo</a>	3.95 €
Cable con 4 salidas	<a href="https://n9.cl/ytdr4">https://n9.cl/ytdr4</a>	3.02 €
Caja PVC para los componentes	<a href="https://n9.cl/30nlx">https://n9.cl/30nlx</a>	1.61 €
Servomotor de metal MG90S	<a href="https://n9.cl/2vyie">https://n9.cl/2vyie</a>	1.634€ (5 Ud. 8.17 €)
Total		11.89 €

Algunos posibles ejemplos de implementación en el mercado de un sensor de movimiento sería en un coche, donde podríamos ponerlo en la parte delantera o trasera para evitar atropellos o incidentes no deseados. También nos puede ayudar a detectar posiciones de objetos en el momento del aparcamiento





40. Ejemplo básico del funcionamiento en un coche

Otro lugar donde podría ubicarse sería en zonas comunes de edificios para domotizar la zona y, con la detección de presencia, encender luces con el paso de personas. Con este sistema, también se disuadiría a posibles intrusos dentro del edificio, dado que las luces se encenderían en cuanto estos entraran a la propiedad.



*41. Ejemplo de sensor PIR en zona común*

Por último, y dada la situación actual, otro uso de este sensor sería en dispensadores de gel hidroalcohólico. Evitaríamos el contacto con cualquier superficie y, con únicamente pasarla la mano, nos sería proporcionada una dosis del desinfectante. También, habría que modificar la programación del propio sensor para que se coordinara con un interruptor. En este caso, la comunicación por Wifi nos sería inservible y podríamos aplicar como placa un Arduino Nano. Podemos observar un prototipo muy básico en la siguiente fotografía:



42. Dispensador de gel con Arduino nano

El dispositivo mostrado no es de mi autoría, se puede encontrar paso por paso como construirlo en el siguiente [enlace](#).

Observamos un diseño final más comercial en esta imagen:



43. Ejemplo de dispensador comercial con sensor PIR

## 10 BIBLIOGRAFÍA

<https://www.arduino.cc/reference/en/>

<https://www.arduino.cc/en/Tutorial/HomePage>

<https://playground.arduino.cc>

<https://es.wikipedia.org/wiki/Fritzing>

<https://www.mpja.com/download/31227sc.pdf>

[https://www.canva.com/es\\_es/graficos/diagramas-gantt/](https://www.canva.com/es_es/graficos/diagramas-gantt/)

<https://aprendiendoarduino.wordpress.com/2016/12/11/ide-arduino/>

<https://www.arduino.cc/en/Guide/Windows>

<https://n9.cl/eshttps://n9.cl/es>

<http://www.cedom.es/sobre-domotica/tabla-de-niveles-para-evaluacion-de-instalaciones-domoticas#aqui>

<https://www.espressif.com/en/products/socs/esp32>

<https://programarfacil.com/esp8266/programar-esp32-ide-arduino/>

<https://www.profetolocka.com.ar/2020/07/09/programando-el-esp-32-con-el-arduino-ide/>

<https://hardwarehackingmx.wordpress.com/2014/01/04/fritzing-que-es-y-como-se-usa/>

[https://www.canva.com/design/play?category=tADWs1ZQ3GI&type=TACQ-ICLuV8&uid=72efa3d5-87a1-48ae-a368-a159658bf589&branch\\_match\\_id=929074866826789789](https://www.canva.com/design/play?category=tADWs1ZQ3GI&type=TACQ-ICLuV8&uid=72efa3d5-87a1-48ae-a368-a159658bf589&branch_match_id=929074866826789789)

[https://www.amazon.es/Internet-cosas-Arduino-Manual-práctico/dp/8428341869/ref=bx\\_49/260-7922287-6487626?pd\\_rd\\_w=j8BAg&pf\\_rd\\_p=c1b53c9a-c88e-47fb-820b-7d6665857066&pf\\_rd\\_r=VHW041P3Q8TKR0NZ2HJB&pd\\_rd\\_r=9db9c5db-4035-4472-afe3-a6dff4f54700&pd\\_rd\\_wg=O9qwM&pd\\_rd\\_i=8428341869&psc=1](https://www.amazon.es/Internet-cosas-Arduino-Manual-práctico/dp/8428341869/ref=bx_49/260-7922287-6487626?pd_rd_w=j8BAg&pf_rd_p=c1b53c9a-c88e-47fb-820b-7d6665857066&pf_rd_r=VHW041P3Q8TKR0NZ2HJB&pd_rd_r=9db9c5db-4035-4472-afe3-a6dff4f54700&pd_rd_wg=O9qwM&pd_rd_i=8428341869&psc=1)

[https://www.amazon.es/mundo-GENUINO-ARDUINO-Curso-práctico-formación/dp/8494345028/ref=sr\\_1\\_6?dchild=1&keywords=arduino&qid=1622746988&s=books&sr=1-6](https://www.amazon.es/mundo-GENUINO-ARDUINO-Curso-práctico-formación/dp/8494345028/ref=sr_1_6?dchild=1&keywords=arduino&qid=1622746988&s=books&sr=1-6)

[https://www.amazon.es/Robotica-Y-Domotica-Basica-Arduino/dp/8499646093/ref=sr\\_1\\_8?dchild=1&keywords=arduino&qid=1622746988&s=books&sr=1-8](https://www.amazon.es/Robotica-Y-Domotica-Basica-Arduino/dp/8499646093/ref=sr_1_8?dchild=1&keywords=arduino&qid=1622746988&s=books&sr=1-8)

[https://iberotecno.com/blog-supratecno/47-tutoriales-paso-a-paso/108-como-instalar-esp32-en-el-ide-de-arduino-tutorial#:~:text=A%20continuación%2C%20vamos%20al%20menu,resultado%20y%20presiona remos%20"Instalar".](https://iberotecno.com/blog-supratecno/47-tutoriales-paso-a-paso/108-como-instalar-esp32-en-el-ide-de-arduino-tutorial#:~:text=A%20continuación%2C%20vamos%20al%20menu,resultado%20y%20presiona remos%20)

<http://tutorialesdeelectronicabasica.blogspot.com/2020/03/diagrama-del-circuito-de-ahorro-de.html>

<https://moviltronics.com/dispensador-automatico-gel-anti-bacterial/>

<https://fritzing.org>

<https://github.com>

<https://es.stackoverflow.com>

<https://github.com/witnessmenow/Universal-Arduino-Telegram-Bot>

## 11 ANEXOS

A continuación, se explicarán los códigos aplicados a la placa y, por último, su configuración final.

Aquí se muestra el código que habilita la comunicación entre el ESP32 y la aplicación de mensajería Telegram.

Primero empezamos importando las librerías necesarias, que son *Wifi.h*, *ESP8266WiFi.h*, *WiFiClientSecure.h*, *UniversalTelegramBot.h* y *ArduinoJson.h*.

```
#include <WiFi.h>
```

```
#include <WiFiClientSecure.h>
```

```
#include <UniversalTelegramBot.h>
```

```
#include <ArduinoJson.h>
```

Introducimos nuestras credenciales de red

```
const char* ssid = "xxxx";
```

```
const char* password = "xxxx";
```

Iniciamos el bot de Telegram

```
#define BOTtoken "1864900471:AAFybjpgkbKMGTlinw_7bJua-1CD8pFeHqHw" // your  
Bot Token (Get from Botfather)
```

Definimos el ID del chat

```
#define CHAT_ID "987933826"
```

Indicamos la conexión con el bot de telegram

```
WiFiClientSecure client;
```

```
UniversalTelegramBot bot(BOTtoken, client);
```

```
const int motionSensor = 27;
```

Sensor de movimiento

```
bool motionDetected = false;
```

Indicamos imprimir un mensaje cuando se detecta movimiento

```
void IRAM_ATTR detectsMovement() {  
    motionDetected = true;  
}
```

```
void setup() {
```

Indicamos el modo INPUT\_PULLUP del sensor PIR

```
    pinMode(motionSensor, INPUT_PULLUP);
```

Indicamos el pin del sensor y el modo RISING del mismo

```
    attachInterrupt(digitalPinToInterrupt(motionSensor), detectsMovement, RISING);
```

Iniciamos la conexión Wifi de la placa

```
    Serial.print("Connecting Wifi: ");
```

```
    Serial.println(ssid);
```

Indicamos el modo por defecto del Wifi

```
    WiFi.mode(WIFI_STA);
```

```
    WiFi.begin(ssid, password);
```

Añadimos un certificado root a api.telegram.org

```
    client.setCACert(TELEGRAM_CERTIFICATE_ROOT);
```

Indicamos la conexión Wifi con una espera de 500 ms

```
    while (WiFi.status() != WL_CONNECTED) {  
        Serial.print(".");  
        delay(500);  
}
```

Definimos el contenido de los mensajes

```
    Serial.println("Buenos dias/tardes/noches");
```

```
    Serial.println("WiFi connected");
```

```
    Serial.print("IP address: ");
```

```
    Serial.println(WiFi.localIP());
```

```
bot.sendMessage(CHAT_ID, "Iniciando el sistema...", "");
}
```

Definimos el loop donde nos saltaran las notificaciones

```
void loop() {
  if(motionDetected){
    bot.sendMessage(CHAT_ID, "Movimiento detectado", "");
    Serial.println("Movimiento detectado");
    motionDetected = false;
  }
} }
```

Después de todo esto podemos ver el código que dará vida nuestro servomotor:

Incluimos la librería del servomotor Servo.h

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

Definimos el pin al que va conectado el servo

```
#define SERVO_PIN 9
```

Creamos el objeto Servo

```
Servo demoServo;
```

Definimos el ángulo del servomotor, que podrá variar desde 0° hasta 180°

```
int servoAngle = 0;
```

```
void setup()
```

```
{
```

Definimos el pin hacia el objeto para poder mandarle señal

```
demoServo.attach(SERVO_PIN);
}
```

```
void loop()
```

```
{
```

Incrementamos el ángulo del servomotor



```
for(servoAngle = 0; servoAngle <= 180; servoAngle  
{
```

Definimos el ángulo del servo y a donde se moverá

```
demoServo.write(servoAngle);
```

Esperamos 20 ms hasta que se mueve a la nueva posición

```
delay(20);  
}
```

Decrementamos en ángulo del servomotor

```
for (servoAngle = 180; servoAngle >= 0; servoAngle--)  
{
```

Definimos el ángulo del servomotor y su movimiento

```
demoServo.write(servoAngle);
```

Esperamos 20 ms

```
delay(20);  
}
```

Movemos el servomotor a 0°

```
demoServo.write(0);  
delay(5000);
```

Movemos el servomotor a 180°

```
demoServo.write(180);  
delay(5000);  
*/  
}
```

Una vez vistos los dos códigos por separado, los juntamos para hacerlos funcionar al mismo tiempo:

```
#include <WiFi.h>  
#include <WiFiClientSecure.h>  
#include <UniversalTelegramBot.h>  
#include <ArduinoJson.h>
```

Sensor de movimiento

```

#include <Servo.h>

const char* ssid = "xxxx";
const char* password = "xxxx";

#define SERVO_PIN

#define BOTtoken "1864900471:AAFybjpgkbKMGTlinw_7bJua-1CD8pFeHqHw" // your
Bot Token (Get from Botfather)

#define CHAT_ID "987933826"

Servo demoServo;    //create a servo object
WiFiClientSecure client;
UniversalTelegramBot bot(BOTtoken, client);

int servoAngle = 0;    //servo angle which can vary from 0 - 180
const int motionSensor = 27; // PIR Motion Sensor
bool motionDetected = false;

void IRAM_ATTR detectsMovement() {
    Serial.println("Movimiento detectado");
    motionDetected = true;
}

void setup() {
    demoServo.attach(SERVO_PIN); //attach the pin to the object so that we can send the
signal to it
    pinMode(motionSensor, INPUT_PULLUP);

    attachInterrupt(digitalPinToInterrupt(motionSensor), detectsMovement, RISING);

    Serial.print("Connecting Wifi: ");
    Serial.println(ssid);

```

```

WiFi.mode(WIFI_STA);
WiFi.begin(ssid, password);
client.setCACert(TELEGRAM_CERTIFICATE_ROOT);

while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
}

Serial.println("Buenos dias/tardes/noches");
Serial.println("WiFi connected");
Serial.print("IP address: ");
Serial.println(WiFi.localIP());

bot.sendMessage(CHAT_ID, "Iniciando el sistema...", "");
}

void loop() {
    if(motionDetected){
        bot.sendMessage(CHAT_ID, "Movimiento detectado", "");
        Serial.println("Movimiento detectado");
        motionDetected = false;
    }
    {
        for(servoAngle = 0; servoAngle <= 180; servoAngle++)
        {
            demoServo.write(servoAngle);
            delay(20);
        }
        for (servoAngle = 180; servoAngle >= 0; servoAngle--)

```

```
{  
  demoServo.write(servoAngle);  
  delay(20);  
}  
}
```

## 12 OTROS

Internet de las cosas (IoT) con Arduino. Manual práctico, Jesús Pizarro Peláez, ISBN 8428341869, Editorial Paraninfo.

Desarrollo de aplicaciones IoT en la nube para Arduino y ESP8266; Tomas Domínguez Mínguez, ISBN 8426728456, Editorial Marcombo.

El mundo GENUINO-ARDUINO. Curso práctico de formación; Oscar torrente Artero, ISBN 8494345028, Editorial RC Libros.

Robótica Y Domótica Básica Con Arduino; Pedro Porcuna López, ISBN 8499646093, Editorial RA-MA S.A.

Exploring Arduino: Tools and Techniques for Engineering; Jeremy Blum, ISBN 1118549368, Editorial Wiley.

Domótica e inmótica. Viviendas y Edificios Inteligentes. Cristóbal Romero Morales, ISBN 8499640176

Algunos enlaces del documento han sido acortados para mayor comodidad, exceptuando la bibliografía.