<https://abrazalaweb.net/2020/01/raspberry-pi-controlar-aire-acondicionado-con-infrarrojos/>

**Raspberry Pi: controlar Aire acondicionado con infrarrojos**

Raspbery Pi con una placa de expansión de Infrarrojos (IR expansion board), para controlar desde internet el aparato de aire acondicionado (en realidad la función de bomba de calor). En mi caso de momento sólo me interesa la función de encender y apagar.

Los pasos serán:

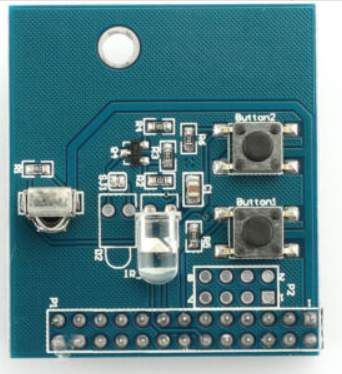
* Grabar las señales de infrarrojos que emite el mando a distancia (necesitaremos un receptor de infrarrojos en la RPi)
* Reproducir las señales que hemos grabado y comprobar que encienden/apagan el aire acondicionado
* Montar una sencilla web en la Raspberry Pi para controlar el aire acondicionado desde internet

Existe mucha información de esto en internet. He buscado mucho, y realizado muchas pruebas hasta conseguir el resultado final.

Vamos a ver:

1. [Hardware](https://abrazalaweb.net/2020/01/raspberry-pi-controlar-aire-acondicionado-con-infrarrojos/#hardware): Placa de expansión Infrarrojos para Raspberry Pi
2. [Software](https://abrazalaweb.net/2020/01/raspberry-pi-controlar-aire-acondicionado-con-infrarrojos/#software): Para capturar y reproducir las señales de infrarrojos del mando a distancia. **lirc** ( <http://www.lirc.org/> ) pero he encontrado demasiados problemas y ha sido imposible lograr el objetivo (es un software antiguo). La solución: **pigpio** ( <http://abyz.me.uk/rpi/pigpio/index.html> )
3. [Web](https://abrazalaweb.net/2020/01/raspberry-pi-controlar-aire-acondicionado-con-infrarrojos/#web)para control por internet
4. [Análisis](https://abrazalaweb.net/2020/01/raspberry-pi-controlar-aire-acondicionado-con-infrarrojos/#analisis) de los resultados de*Lirc vs Pigpio* al capturar las señales del mando a distancia

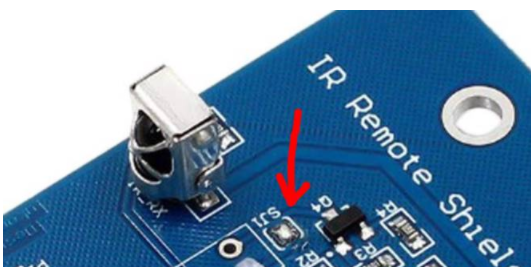
**Hardware: Expansión Infrarrojos**



Tiene un receptor y un emisor de infrarrojos (y dos pequeños botones, aunque no los uso para nada, en algunas páginas de venta se indica *Button1:GPIO 27, Button2:GPIO22, D1&D2:GPIO17, IR receiver: GPIO18.* )

Antes de empezar, hay que tener en cuenta que esa placa está preparada para tener dos emisores (hay que adquirir y soldar el segundo por separado). Lo que hay que tener en cuenta es que para que funcione con un emisor, hay que puentear el **«jumper» SJ1**, algunas de las placas que venden ya lo traen así de fábrica, la mía no, tuve que echar una pequeña gota de soldadura:

Jumper SJ1 «abierto» no va a funcionar con un sólo emisor



Alguna placas las venden con el SJ1 ya soldado

Una vez insertada la placa en la raspberry Pi, hay que modificar el archivo de configuración para cargar los módulos del kernel que permiten el uso del emisor y el receptor, en esta placa en concreto:

dtoverlay=gpio-ir,gpio\_pin=18

dtoverlay=gpio-ir-tx,gpio\_pin=17

## Software: LIRC (descartado), PIGPIO la solución

Puesto que mi Raspberry Pi tiene instalado Raspbian Stretch, actualizado, con kernel 4.19.X, mi punto de partida fue las instrucciones sobre lirc, en particular este: [[Stretch/Buster] Using LIRC with kernel 4.19.X and gpio-ir](https://www.raspberrypi.org/forums/viewtopic.php?f=28&t=235256).

Probé de muchas maneras, y aunque era capaz de capturar la señal del mando con el comando ***«mode2»***, me fue imposible enviarla con el comando **«irsend»** y el archivo de configuración que creaba con los datos capturados en «raw«. Comprobé el correcto funcionamiento de la placa con un archivo de configuración de un mando cualquiera descargado de la biblioteca que tiene **lirc** (con el truco de la cámara del móvil grabando el emisor, todo funcionaba).

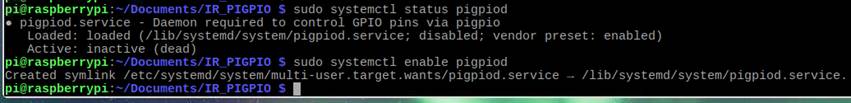
### PIGPIO

Al buscar simplemente cómo poder capturar una señal de infrarrojos y reproducirla, sin más, encontré lo que me proporcionó la guía para la solución: [Alternatives to LIRC for transmitting IR codes](https://raspberrypi.stackexchange.com/questions/75689/alternatives-to-lirc-for-transmitting-ir-codes). La documentación, ejemplos y scripts en Python están en <http://abyz.me.uk/rpi/pigpio/index.html>

Antes de nada hay que lanzar el daemon pgpiod (en mni caso en la raspberry pi ya estaba instalado, quizá se instaló durante todas mis pruebas y compilaciones de lirc…)

sudo pigpiod

Y, aprovechando para arrancarlo siempre en el boot:



Para ver cómo funcionan las cosas con pigpio, primero use la utilidad **piscope** (<http://abyz.me.uk/rpi/pigpio/piscope.html>), siguiendo las instrucciones, probando a capturar y reproducir señales de un mando a distancia de una TV,. Finalmente capturé la señal del mando a distancia del aire acondicionado, botón de encender, bastante compleja, y es porque manda información de la temperatura, el modo, etc:

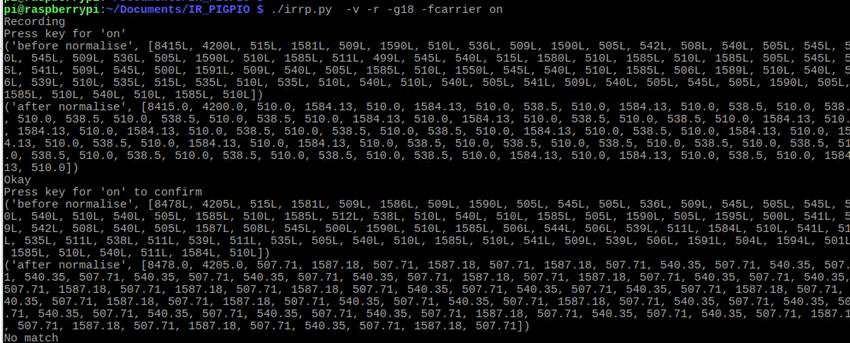
Señal del botón encender del mando a distancia del aire acondicionado

Una vez verificado que **pigpio** funciona, simplemente pruebo el script para grabar y reproducir las señales: [IR record and playback example](http://abyz.me.uk/rpi/pigpio/examples.html#Python_irrp_py) con el script**irr.py**.

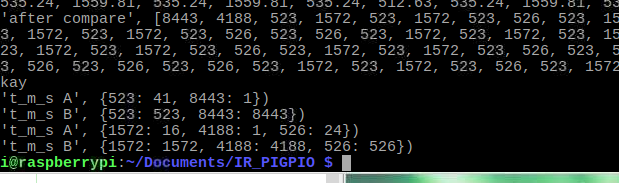
Lo ejecuto tal cual, apunto el mando del aire acondicionado, sólo voy a grabar un comando ‘on’, para ello uso (el pin GPIO18 es el del receptor), el archivo lo llamo carrier (luego le cambio el nombre)

$ ./irrp.py -v -r -g18 -fcarrier on

Hay que tener en cuenta que el script irrp.py requiere confirmación del botón pulsado, y que en el mando del aire acondicionado, al encender por primera vez, se enciende su pantalla. Al pulsar el botón por segunda vez, se apaga la pantalla y el aire acondicionado, y una nueva pulsación vuelve a encender. Por tanto hay que ser cuidadoso, ya que lo que se envía al encender no es lo mismo que al apagar. En la siguiente captura se comprueba el proceso, la segunda pulsación (que lo que hace es apagar) resulta en un «No Match», el script pide repetirlo, y la tercera pulsación (que vuelve a encender) ya coincide y guarda la configuración.

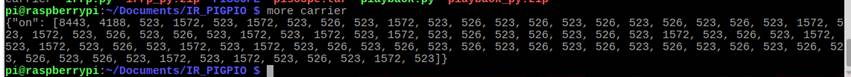


Proceso de captura del mando a distancia

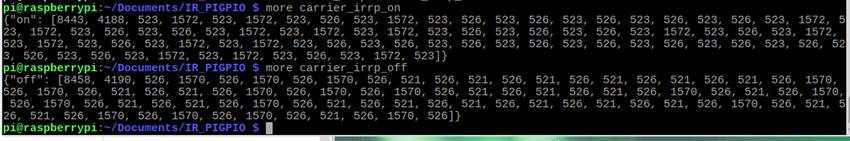


continuación y fin del proceso Ok

El archivo con las señales capturadas:



Le cambio el nombre y repito el proceso, esta vez para el apagado:

Archivos con las señales on / off , muy similares, salvo en algunos valores

Para enviar las señales a través del emisor de infrarrojos de la Raspberry Pi basta con usar los comandos;

pi@raspberrypi:~/$ ./irrp.py -p -g17 -fcarrier\_irrp\_off off

pi@raspberrypi:~/$ ./irrp.py -p -g17 -fcarrier\_irrp\_on on

Esto abre un mundo de posibilidades de jugar con la captura y reproducción de señales de infrarrojos con la RPi, muy interesante la los scripts para usar **piscope** playback piscope recordings <http://abyz.me.uk/rpi/pigpio/examples.html#Python_playback_py>

## Web para controlar el aire acondicionado desde internet

Una vez que podemos encender y apagar el aire acondicionado desde la RPi, montamos un servidor web en ella y una pequeña página web para acceder desde internet a nuestro aparato de aire acondicionado (para encenderlo antes de llegar a casa, apagarlo por si no nos acordamos si lo dejamos encendido, etc).

A continuación dos videos del control con la RPi funcionando, en el primero, con la cámara del móvil se percible el emisor de infrarrojos al pulsar el botón en la página web (en la segunda pulsación, como tengo un móvil grabando, el otro con la web, me resulta difícil darle al botón, pero se aprecia que el emisor infrarrojo se activa cuando la página web se recarga)

Yo uso lighttpd, y la página en php, llamando desde ella a unos simples scripts en bash.

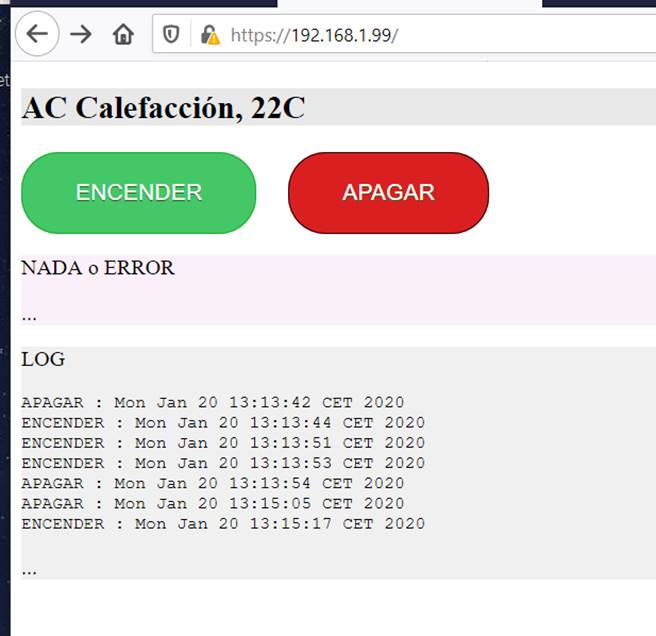
A continuación muestro todo (obviando y saltándome lo relacionado con la instalación y configuración del servidor web en la RPi).

Archivos en el servidor web, en */var/www/html* , propietario *www-data* y grupo *www-data*

* Página web principal, *php*, con un simple formulario y dos botones, y un log, que se escribe por los scripts en un archivo de texto
* script bash para el botón encender, *encender.sh*
* script bash para el botón apagar, *apagar.sh*
* archivo de configuración *carrier\_irrp\_o*n con los valores de la señal para encender
* archivo de configuración *carrier\_irrp\_off* con los valores de la señal para apagar
* script de pigpio ***irrp.py*** (el imprescindible, este no lo copio está en su web: <http://abyz.me.uk/rpi/pigpio/code/irrp_py.zip>)
* archivo de texto (*temp.txt*) donde se va escribiendo el log

Primero una pequeña captura de la página web, y a continuación el código de cada uno de los archivos indicados, todo en esta descarga:

[Archivos WebDescarga](https://abrazalaweb.net/myBlog/wp-content/uploads/2020/01/Web.zip)



Página web al cargar por primera vez

Código de la página web php

<?php

$respuesta="NADA o ERROR";

$res="...";

// activar desde el form

**if** (**isset**($\_POST['activar']) ) {

$res=exec("./encender.sh");

$respuesta="ENCENDIDO";

}

//desactivar desde el form, para reiniciar

**else** **if** (**isset**($\_POST['desactivar'])) {

$res=exec("./apagar.sh");

$respuesta="APAGADO";

}

?>

<!DOCTYPE html>

<**html**>

<**head**><**title**>...AC...</**title**>

<**style**>

.myButtonOn {

**background-color**:#44c767;

**border-radius**:28px;

**border**:1px solid #18ab29;

**display**:inline-block;

**cursor**:pointer;

**color**:#ffffff;

**font-family**:Arial;

**font-size**:17px;

**padding**:20px 40px;

**text-decoration**:none;

**text-shadow**:0px 1px 0px #2f6627;

**margin-right**: 20px;

}

.myButtonOn:**hover** {

**background-color**:#5cbf2a;

}

.myButtonOn:**active** {

**position**:relative;

**top**:1px;

}

.myButtonOff {

**background-color**:#d92020;

**border-radius**:28px;

**border**:1px solid #1c0201;

**display**:inline-block;

**cursor**:pointer;

**color**:#ffffff;

**font-family**:Arial;

**font-size**:17px;

**padding**:20px 40px;

**text-decoration**:none;

**text-shadow**:0px 1px 0px #2f6627;

}

.myButtonOff:**hover** {

**background-color**:#bd2c2a;

}

.myButtonOff:**active** {

**position**:relative;

**top**:1px;

}

</**style**>

</**head**>

<**body**>

<**h2** style="**background-color**:#E8E8E8">AC Calefacción, 22C</**h2**>

<**p**></**p**>

<**form** method="POST" action="index.php">

<**input** type="submit" class="myButtonOn" name="activar" value="ENCENDER">

<**input** type="submit" class="myButtonOff" name="desactivar" value="APAGAR">

</**form**>

<**div** style="**background-color**:#FAF0FA;">

<**p**><**p**><?php **echo** $respuesta ?></**p**></**p**>

<**p**><**p**><?php **echo** $res ?></**p**></**p**>

</**div**>

<**div** style="**background-color**:#F0F0F0;"<p>LOG </**p**>

<?php

$log=shell\_exec("tail -n 7 ./temp.txt");

**echo** "<pre>$log</pre>"

?>

<**p**>...</**p**>

</**div**>

</**body**>

</**html**>

*encender.sh*

#!/bin/bash

**.**/irrp.py -p -g17 -fcarrier\_irrp\_on on

fecha=$(date)

**echo** "ENCENDER : $fecha" >> temp**.**txt

apagar.sh

#!/bin/bash

**.**/irrp.py -p -g17 -fcarrier\_irrp\_off off

fecha=$(date)

**echo** "APAGAR : $fecha" >> temp**.**txt

archivos de configuración

carrier\_irrp\_on

{"on": [8443, 4188, 523, 1572, 523, 1572, 523, 526, 523, 1572, 523, 526, 523, 526, 523, 526, 523, 526, 523, 526, 523, 1572, 523, 1572, 523, 526, 523, 526, 523, 1572, 523, 1572, 523, 1572, 523, 526, 523, 526, 523, 526, 523, 1572, 523, 526, 523, 1572, 523, 1572, 523, 526, 523, 1572, 523, 1572, 523, 526, 523, 526, 523, 526, 523, 526, 523, 526, 523, 526, 523, 526, 523, 526, 523, 526, 523, 526, 523, 1572, 523, 1572, 523, 526, 523, 1572, 523]}

carrier\_irrp\_off

{"off": [8458, 4190, 526, 1570, 526, 1570, 526, 1570, 526, 521, 526, 521, 526, 521, 526, 521, 526, 521, 526, 521, 526, 1570, 526, 1570, 526, 521, 526, 521, 526, 1570, 526, 1570, 526, 1570, 526, 521, 526, 521, 526, 521, 526, 1570, 526, 521, 526, 1570, 526, 1570, 526, 521, 526, 521, 526, 1570, 526, 521, 526, 521, 526, 521, 526, 521, 526, 521, 526, 521, 526, 1570, 526, 521, 526, 521, 526, 1570, 526, 1570, 526, 1570, 526, 521, 526, 1570, 526]}

**Análisis de las señales capturadas con *lirc* y *pigpi*o**

Ahora que tenemos los valores de las señales válidos con Pigpio, podemos comparar con lo que capturé con lirc .

Parece ser que con lirc, el modo captura (comando **mode2**, con irrecord fue imposible) se capturaban demasiadas veces, 252 señales y un timeout cada 83 señales, (lo he puesto en tabla excel), con pigpio **irrp.py** 83 señales, o sea, la cosa parece bien

En las siguientes gráficas se ve que con los primeros 83 señales, lirc tiene un desfase al inicio y al fin.

La primera muestra la captura con pigpio para las señales on/off, la segunda lo mismo pero con lirc, que es parecida, pero no idéntica.

