**Software**

# Driver

Como estamos usando el driver de la marca adafruit modelo pca9685, la librería que hay que usar es: Adafruit\_PCA9685.

El enlace de la librería es el siguiente: <https://github.com/adafruit/Adafruit_Python_PCA9685>

# Giroscopio

Para el giroscopio mpu6050 la librería a utilizar será mpu6050-raspberrypi. El enlace de la librería es el siguiente: <https://github.com/Tijndagamer/mpu6050>

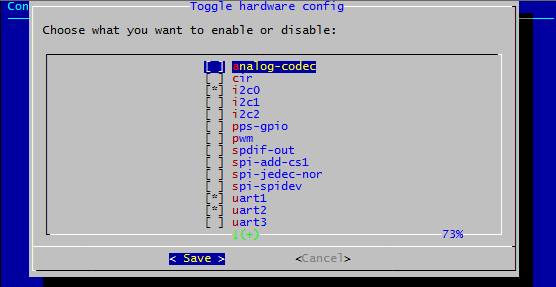
En versiones posteriores se comenzará a utilizar la librería creada por nosotros, que esta en el repositorio con el nombre de mpu6050

# Direcciones

Como vamos a usar diferentes dispositivos conectados al puerto I2c de nuestra placa, vamos a necesitar saber qué dirección se asigna a cada uno.

Antes de comprobar en qué dirección se encuentra el driver hay que activar el protocolo i2c en la placa ya que por defecto viene desactivada.

Comprobamos si está activado con armbian-config. Menú “system settings” -> “Hardware” -> ., en el caso de la Orange pi.



En este ejemplo se han activado i2c0 e i2c1, pero podemos activar i2c2 también.

Al tener que utilizar dos drivers, para el control de los servos, es necesario cambiar la dirección de cada uno con un soldador, de forma que no interfieran con el giroscopio, ni entre ellos.

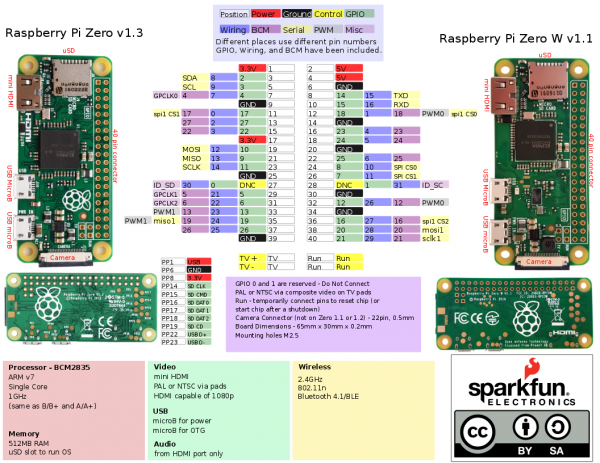
Además una vez hecho esto hay que actualizar el objeto robot en el código de forma que coincidan.

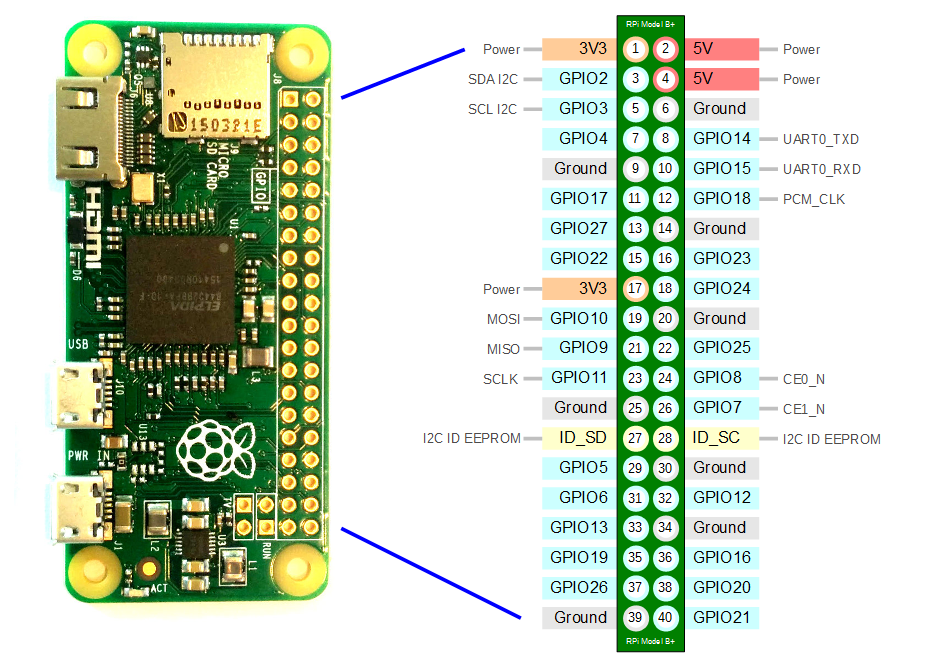
# Permisos

A no ser que ejecutemos nuestros programas desde el usuario root tendremos que darle permiso sobre los puertos i2c al resto de usuarios que no son root puesto que por defecto viene desactivado, una buena guía de cómo hacerlo es la siguiente <https://www.abelectronics.co.uk/kb/article/1061/i2c--smbus-and-armbian-linux>

Pines

El driver que hemos cogido no trabaja con el puerto sda\_0 ni scl\_0, si no que la librería está preparada para trabajar con el pin scl\_1 y sda\_1. Si no la librería no encuentra el driver de forma correcta





Función salida pwm

El driver escogido da un pulso se entre 450, que es su mínimo, mientras que 2100 parece ser un límite seguro para que los servos no se sobrecarguen. Por lo tanto, hay que hacer una función que permita recibir un ángulo deseado, y después pasarlo a un pulso pwm. E**l rango del pulso hay que comprobarlo experimentalmente.**

Raspberry pi

USUARIO : pi

CONTRASEÑA: icai2019

Para poder utilizar la raspberry pi hay que cargarle una imagen de un sistema operativo, en este caso el elegido es raspbian por el soporte que da la comunidad, para ello hay que descargarse dicha imagen en el siguiente enlace <https://www.raspberrypi.org/downloads/>

Una vez cargada la imagen de raspbian, será necesario conectarse a la placa via hdmi para poder configurar la conexión ssh.

Para establecer una ip fija, se sigue el siguiente tutorial:

<https://www.raspberrypi.org/forums/viewtopic.php?t=191140>

Una vez hecho esto hay que activar el ssh, en el menú de interfaces, así como el i2c, en el, mismo menú.

# Movimiento

Como la dirección de los servos se encuentra invertida, hay que restar el valor máximo de lo ángulos para poder mover dos servos al mismo sitio (si se manda una mano bajar, que la otra también baje), para ello hay que dividir los servos entre derecha e izquierda. Se tomará el lado derecho como referencia, por facilidad. Se pondrán en posiciones impares los servos de la izquierda, y en posiciones pares los servos de la derecha, de acuerdo con la siguiente tabla:

|  |  |
| --- | --- |
| Servo | Pin |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |
| 9 |  |
| 10 |  |
| 11 |  |
| 12 |  |
| 13 |  |
| 14 |  |
| 15 |  |
| 16 |  |
| 17 |  |
| 18 |  |
| 19 |  |
| 20 |  |

Dicha tabla hay que actualizarla en el fichero json

Simulación

Para realizar la simulación del humanoide, se va a realizar con el motor de simulación Bullet. Es gratis y bastante sencillo de utilizar.

Un tutorial de como utilizarlo es el siguiente: <https://docs.google.com/document/d/10sXEhzFRSnvFcl3XxNGhnD4N2SedqwdAvK3dsihxVUA/edit>

El workflow será el siguiente, primero una inteligencia artificial (modelo aún por determinar), realizará una “averiguación” de los motores a mover o pasos a realizar, después se pasarán dichos actos a la simulación, en pybullet. Una vez esto se devolverán las recompensas, y se volverá al inicio del programa.

<http://wiki.ros.org/urdf/Tutorials/Create%20your%20own%20urdf%20file>