2 - Primeros pasos

En este apartado vamos a introducir varios conceptos que necesitamos tener claro para empezar a trabajar con nuestra Raspberry Pi

- 2.1 Sistemas operativos
- 2.2 Electrónica de Raspberry Pi 4 modelo B

2.1 Sistemas operativos

Como todo ordenador, **la Raspberry Pi necesita de un sistema operativo para funcionar**. Es bastante probable que si estás leyendo estos materiales desde un ordenador de sobremesa o portátil estés utilizando un buen sistema operativo como podría ser Vitalinux o cualquier otro derivado Linux como Ubuntu, Debian o Fedora. También pudiera ser estuvieses utilizando otro tipo de sistemas operativos propietarios como Windows o MacOs. Si estás utilizando una tableta o smartphone es bastante probable que esté utilizando Android o iOS. Todos ellos son sistemas operativos, pero ¿qué es un sistema operativo? sin complicarnos mucho podríamos decir que **un sistema operativo es el software que se ubica entre el hardware y las aplicaciones que tiene por encima**. Si quieres ampliar la información sobre qué son los sistemas operativos te dejo este enlace a materiales que preparé hace años para mi alumnado de FP básica.

Para trabajar con la Raspberry Pi nosotros/as vamos a trabajar con Raspberry Pi OS (antes llamado raspbian, seguro que en estos apuntes alguna vez me refiero al mismo de la manera antigua) que es un sistema operativo Linux que deriva a Debian optimizado para la Raspberry Pi por lo que si estás acostumbrado/a a trabajar con Vitalinux, Ubuntu o Debian no vas a notar ningún cambio.

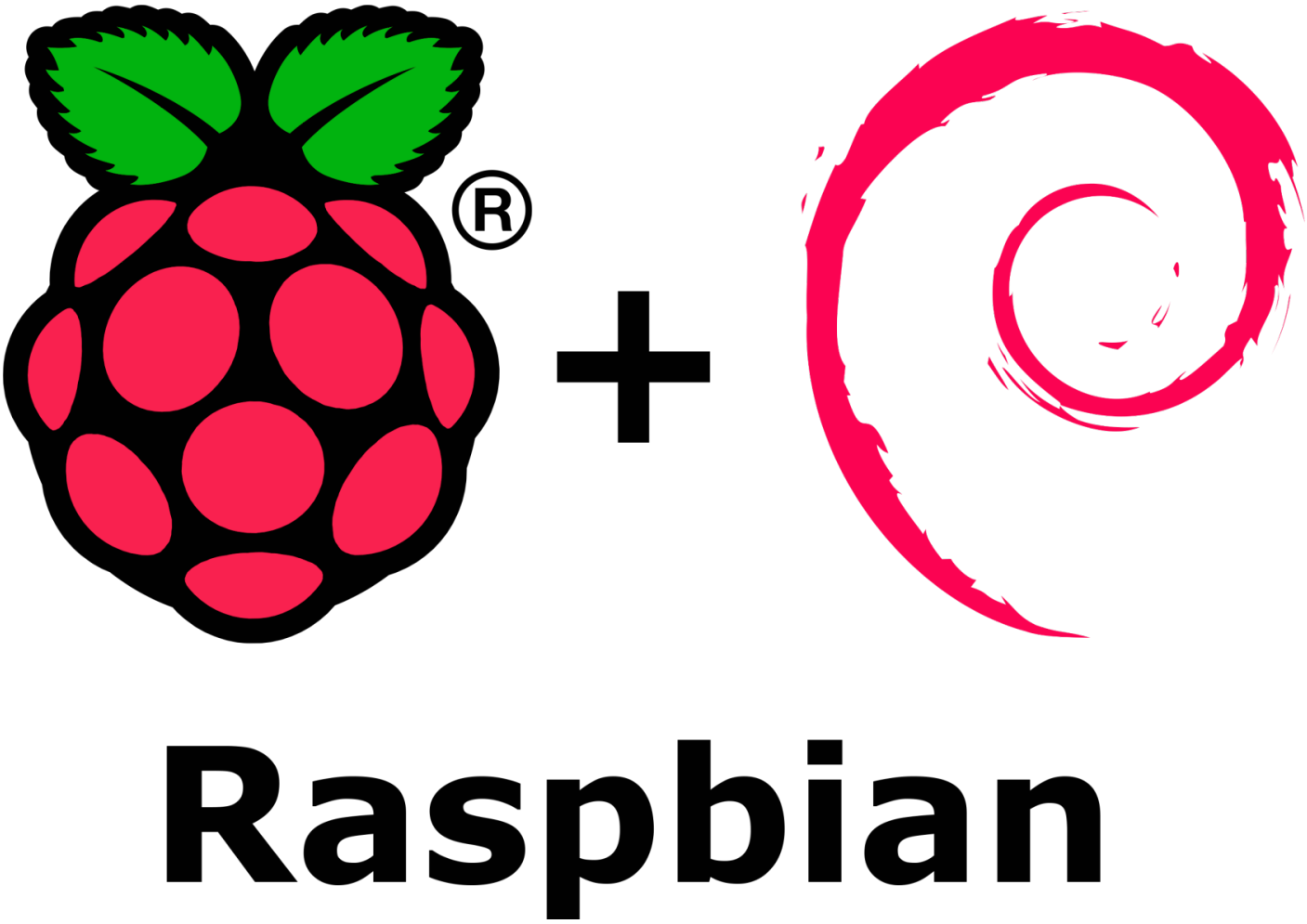


Imagen obtenida de https://pistachitos.com/guias/comenzando-con-raspbian/

Además de este sistema operativo en la Raspberry Pi puedes instalar otros como:

- Raspberry Pi OS (Raspbian)
- Windows 10 IoT Cores
- LibreELEC
- RetroPie
- OSMC
- Ubuntu Desktop
- Ubuntu Core
- Ubuntu Server
- RISC OS
- Kali Linux
- Arch Linux ARM
- FreeNAS
- CentOS
- Fedora
- SUSE Linux Enterprise Server
- LineageOS
- ...

Por si tienes curiosidad sobre como instalar Windows 10 IoT (Internet de las cosas) te dejo un par de enlaces con información. El 1º en castellano sobre W10 y el 2º en inglés sobre W11: https://www.ionos.es/digitalguide/servidores/know-how/sistemas-operativos-para-raspberry-pi/ y https://www.tomshardware.com/how-to/install-windows-11-raspberry-pi

Si en alguna ocasión has instalado algún sistema operativo en algún ordenador es bastante probable que hayas hecho uso de algún CD o DVD o de algún USB que hayas establecido como 1er elemento de arranque y, a partir de ahí, hayas seguido las instrucciones. En Raspberry Pi esta instalación la haremos haciendo uso de imágenes. Sigue leyendo para saber qué son y cómo las usaremos.

Imágenes

Si pensamos en una imagen quizás nos venga a la cabeza la representación gráfica de algo pero en este caso cuando utilizamos el término imagen lo que queremos indicar es la copia exacta de un disco. Esto nos permite utilizar una imagen existente de un sistema operativo y llevarla a nuestra Raspberry Pi de modo que será funcional. En este curso, en la mayoría de herramientas que utilicemos haremos uso de un sistema operativo y sobre él instalaremos diferentes servicios pero en alguna de las posibilidades que veamos también usaremos directamente una imagen que nos traerá ya todo configurado y listo para utilizar. Es el caso que veremos en el capítulo 3.4 BOBcera. Videoconsola retro.

Si queremos grabar una imagen en nuestra tarjeta SD únicamente debemos seguir los pasos que nos indican en este vídeo:

https://www.youtube.com/embed/ntaXWS8Lk34

En resumen:

1. De la web https://www.raspberrypi.com/software/ descarga el programa que corresponde a tu sistema operativo
2. Instala la aplicación "Raspberry Pi Imager" que acabas de descargar
3. Introduce la tarjeta micro SD en un lector de tu ordenador
4. Ejecuta la aplicación
5. Elige qué versión de sistema operativo quieres instalar. Por ejemplo la versión de escritorio (Desktop) si no estás acostumbrado/a a trabajar solo con comandos
6. Deberías preconfigurar algunas cuestiones como:
   1. nombre
   2. SSH, para conectarte posteriormente de modo remoto a la Raspberry Pi sin necesidad de conectarle teclado, ratón y monitor
   3. red WIFI
   4. locale (idioma)
7. Selecciona dónde está la tarjeta SD
8. Graba la imagen en la microSD
9. Cuando termine expúlsala del sistema operativo dónde hayas estado trabajado
10. Llévate la microSD a la Raspberry Pi y enciéndela. Debería arrancar el sistema operativo que has elegido.

> Si no pones un sistema operativo en la SD de tu Raspberry Pi no podrás usarla.

- CONECTANDO DESDE LA RED LOCAL
- CONECTANDO DE FORMA TEXTUAL CON SSH
- CAMBIAR USUARIO Y CONTRASEÑA
- APAGAR
- CONECTANDO DE FORMA GRÁFICA: VNC

Control de versiones. Git. Github.

En este apartado realmente os quiero hablar de Github pero para hablaros de GitHub os he de hablar de Git y, a su vez, para hablaros de Git os tengo que hablar sobre el control de versiones así que vamos a ver, en el orden lógico, unas pinceladas de estos elementos.

Control de versiones: El control de versiones nos permite ver los diferentes cambios que un documento ha sufrido con el tiempo viendo quién ha hecho qué cambio, cuándo lo ha hecho y por qué lo ha hecho. Para ampliar la información leer aquí.

Git: Es un software de control de versiones muy potente y muy sencillo de utilizar

Imagen obtenida de: https://commons.wikimedia.org/wiki/File:Git-logo.svg?uselang=es

Github: Es una plataforma web que actúa como repositorio de proyectos que utilizan en control de versiones git. De esta plataforma vamos a descargar varios de los proyectos que pondremos en marcha en los capítulos 3 y 4 y me parece adecuado que conozcas algo mas de la misma que limitarme a decirte mas adelante que ejecutes tal o cual comando que tiene como factor común una URL de github. Como curiosidad te diré que en github puedes encontrar parte del código que se desarrolla en CATEDU. Su respositorio es https://github.com/catedu

Imagen obtenida de https://github.com/logos

Docker. Docker-compose

Al igual que ocurría en el apartado anterior con github en el capítulo 3 y 4 haremos uso de docker y docker-compose y también me parece adecuado contarte qué son y cuales son ventajas en vez de que te limites a copias y pegar comandos.

Quizás hayas oído o utilizado en alguna ocasión máquinas virtuales con Virtual Box, vmware u otras herramientas que permiten su uso y gestión. De ser así habrás visto que sobre tu sistema operativo has creado un hardware virtual y en ese hardware virtual has instalado otro sistema operativo y los distintos programas que te ha interesado. A ese conjunto es a lo que se llama máquina virtual. Si en esa máquina, por ejemplo, borras todo no afecta a tu sistema operativo anfitrión, lo cual te permite una serie de ventajas como probar determinados sistemas y/o programas sin poner en riesgo tu equipo anfitrión. De igual modo, si tienes un determinado programa que tienes que utilizar y solo funciona en un sistema operativo que no es el que utilizas puedes crear una máquina virtual con ese programa y ejecutarlo. Las ventajas parecen claras. Ahora vamos con la gran desventaja: son muy pesadas, requieren de muchos recursos. Por ello surgen los contenedores y su máximo exponente es Docker.

Imagen obtenida de https://www.docker.com/company/newsroom/media-resources/

Docker y **Docker-compose** van a permitirnos levantar y tirar contenedores (dónde habrá distintos servicios) de un modo muy sencillo. Esto nos va a permitir que en nuestra Raspberry Pi podamos instalar diferentes utilidades sin que los requerimientos de unas afecten a los de otras. Son herramientas sencillas de utilizar y, en muchas ocasiones, copiaremos textos directamente de github que nos permitirán poner todo en marcha. De todos modos, para evitar el uso continuado del terminal que tanto nos gusta a los/as informáticos/as y tan poco al resto de usuarios/as en el capítulo 3 la primera herramienta que veremos será Portainer la cual nos permitirá gestionar los contenedores de un modo gráfico.

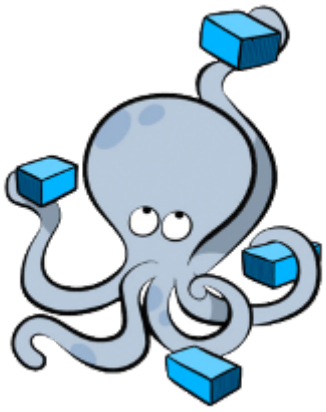


Imagen obtenida de https://github.com/docker/compose

Vamos a ver los comandos que deberemos ejecutar en la terminar de Raspberry Pi OS para instalar ambos servicios. Necesitarás conexión a internet en tu Raspberry Pi.

> Si no quieres instalar docker y docker-compose del modo que te cuento a continuación echa un vistazo al capítulo 3.3 Linux Media Delivery System (LMDS). Centro de descargas pues en el mismo, aunque harás uso del terminal, podrás instalar docker y docker-compose con un asistente.

Empecemos por instalar docker:

```
curl -fsSL https://get.docker.com -o get-docker.sh
sudo sh get-docker.sh
sudo usermod -aG docker ${USER}
sudo su - ${USER}
docker version
```

Tras ejecutar los comandos anteriores verás algo similar a:

```
pi@mediacenter:~ $ docker version
Client:
 Version:           20.10.5+dfsg1
 API version:       1.41
 Go version:        go1.15.15
 Git commit:        55c4c88
 Built:             Mon May 30 18:34:49 2022
 OS/Arch:           linux/arm64
 Context:           default
 Experimental:      true

Server:
 Engine:
  Version:          20.10.5+dfsg1
  API version:      1.41 (minimum version 1.12)
  Go version:       go1.15.15
  Git commit:       363e9a8
  Built:            Mon May 30 18:34:49 2022
  OS/Arch:          linux/arm64
  Experimental:     false
 containerd:
  Version:          1.4.13~ds1
  GitCommit:        1.4.13~ds1-1~deb11u3
 runc:
  Version:          1.0.0~rc93+ds1
  GitCommit:        1.0.0~rc93+ds1-5+deb11u2
 docker-init:
  Version:          0.19.0
  GitCommit:
```

Elaboración propia

> Ya no es necesario instalar docker-compose. Dejo a continuación TACHADO lo que había que hacer anteriormente por si alguien trabaja con versiones viejas de docker pero si acabáis de instalar docker según lo indicado anteriormente ya lo tenéis instalado y no deberíais ejecutar lo siguiente que que está tachado.

~~Ahora terminemos instalando docker-compose:~~

```
sudo apt-get install libffi-dev libssl-dev
sudo apt install python3-dev
sudo apt-get install -y python3 python3-pip
sudo pip3 install docker-compose
docker-compose version
```

~~Tras ejecutar el último comando verás algo similar a la siguiente imagen:~~

```
pi@mediacenter:~ $ docker-compose version
docker-compose version 1.25.0, build unknown
docker-py version: 4.1.0
CPython version: 3.9.2
OpenSSL version: OpenSSL 1.1.1n  15 Mar 2022
```

~~Elaboración propia~~

> Si en algún lugar de los apuntes aparece algún comando que sea `docker-compose up -d` sustituye el guión por un espacio dejándolo así `docker compose up -d`

Un concepto importante relativo a los contenedores es el de **volumen**. Simplificando, un volumen nos va a permitir compartir datos entre contenedores y/o la máquina anfitriona. Es decir, aunque un contenedor lo "tiremos" o borremos en la carpeta que hayamos establecido como volumen tendremos persistencia de datos.

2.2 Electrónica de Raspberry Pi 4 modelo B

La información que aquí detallo está obtenida principalmente de la página web oficial del producto https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/

A continuación dejo una imagen mas completa que las anteriores dónde se ubica para cada este modelo dónde está cada componente

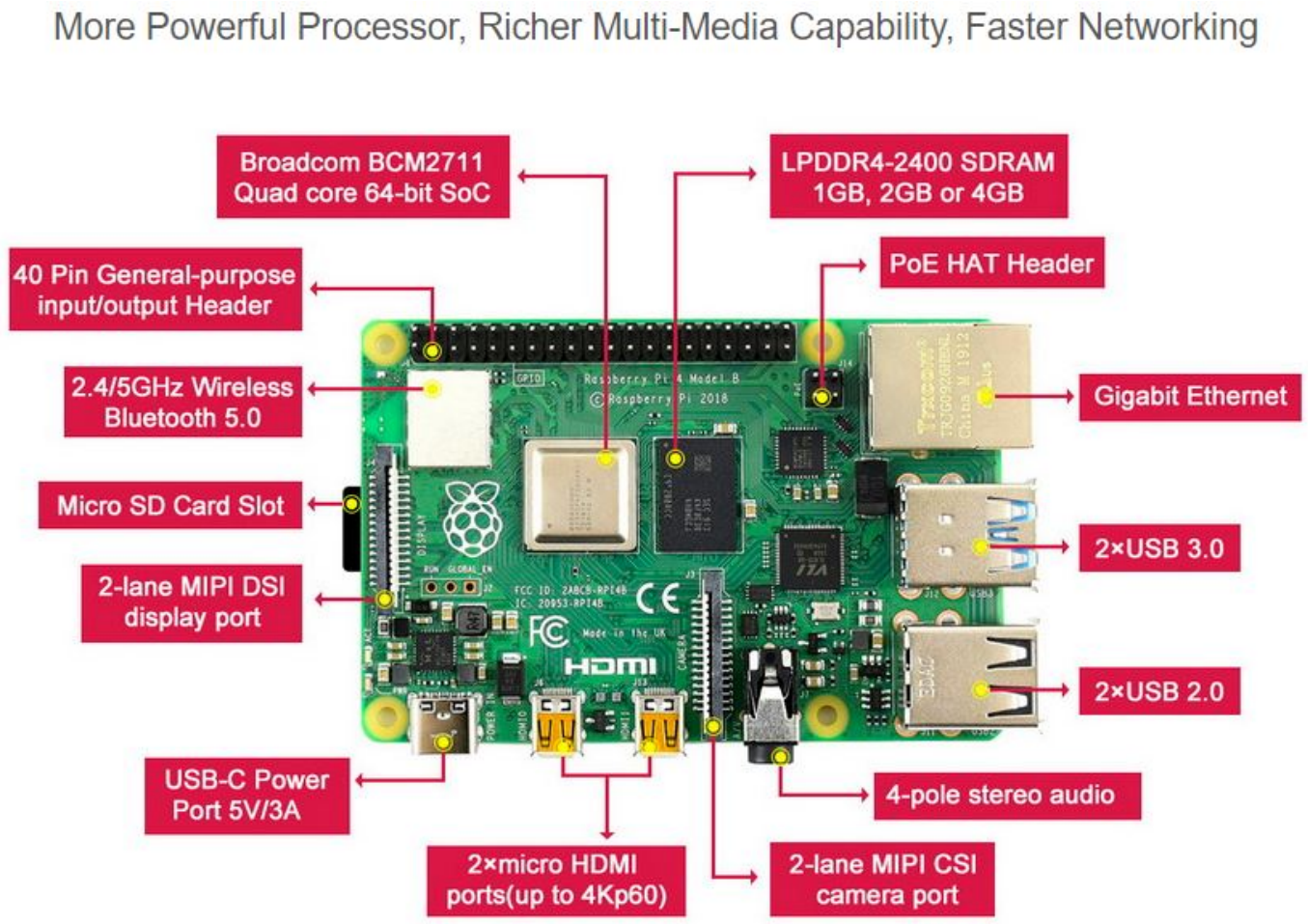


Imagen obtenida de https://cosmicteq.co.za/Raspberry-Pi-4-Model-B-1GB-p506861692

CPU, RAM, GPU

Este modelo de Raspberry cuenta con con chip Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz. En cuanto a la memoria RAM es de tipo LPDDR4-3200 SDRAM y puede ser adquirida con 2, 4 u 8 GB de memoria RAM (la web oficial indica que también existe un modelo de 1GB pero nunca lo he visto comercializado).

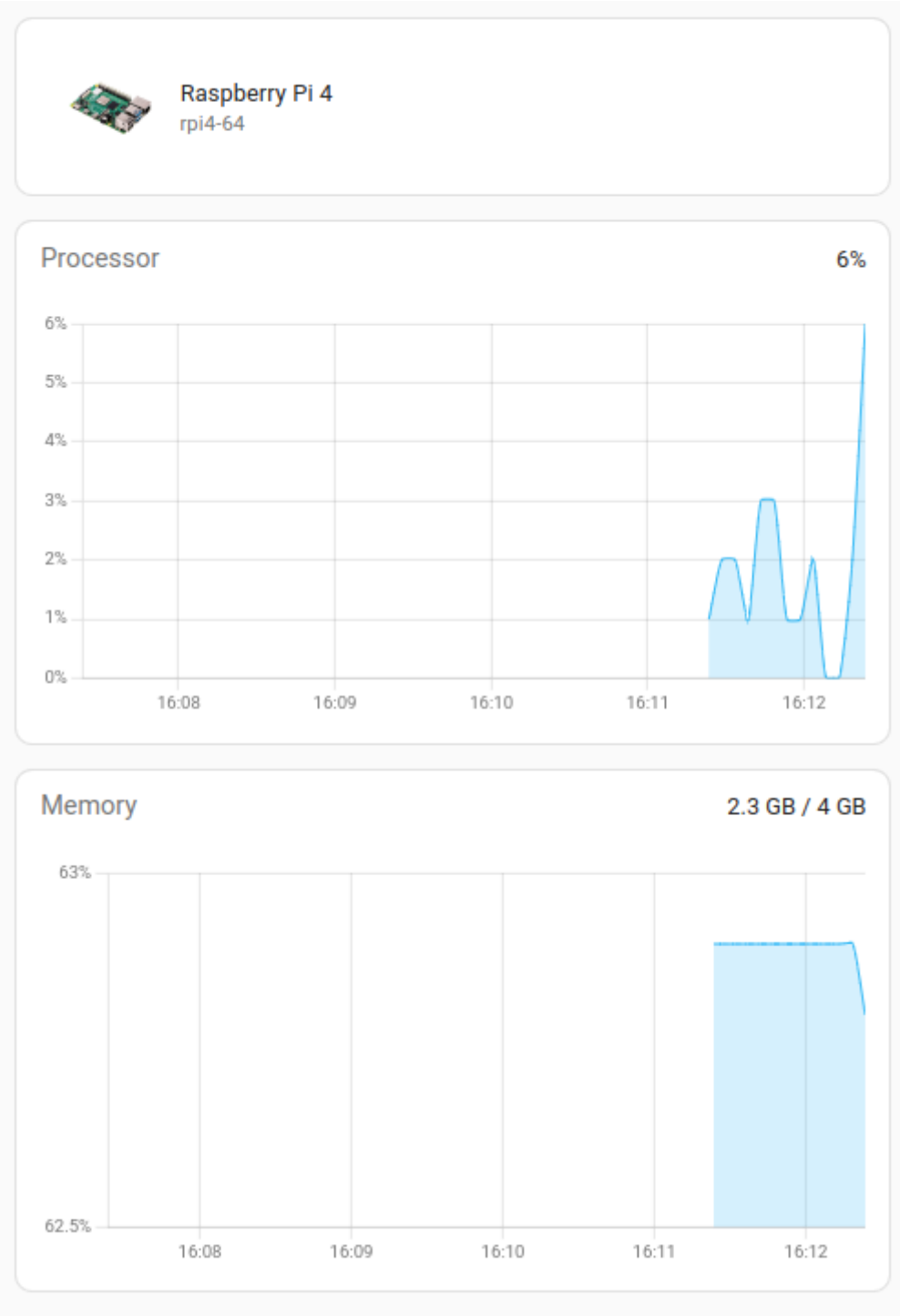
Por si sirve de referencia, en mi caso, los modelos de Raspberry Pi 4 modelo B de que dispongo son los de 4GB. Uno de los modelos es el que se encarga de toda la domótica y el otro modelo se encarga del centro de descargas y multimedia, bloquedo de publicidad, portainer y alguna otra cuestión. En ocasiones desconecto una de ellas y le pongo una tarjeta microSD que tengo específica con la BOBcera (juegos de mi juventud). A continución pongo unas captura de pantalla del consumo de recursos de estas Raspberry Pi para que se veais que están sobrada:

```
Archivo   Editar   Pestañas   Ayuda

  0[|||||                      10.3%]   Tasks: 176, 540 thr; 2 running
  1[||                          4.5%]   Load average: 2.05 2.20 2.24
  2[|                           3.2%]   Uptime: 6 days, 08:26:12
  3[|                           0.6%]
Mem[||||||||||||||||||||1.79G/3.71G]
Swp[||||||||||||||||||100.0M/100.0M]

    PID USER      PRI  NI  VIRT   RES   SHR S CPU%-MEM%   TIME+  Command
  56192 pi         20   0 2904M  268M 92016 R 20.9  7.1 26h11:58 /usr/lib/aarch6
 556742 pi         20   0  6784  4860  2780 R  3.9  0.1  0:01.54 htop
  56228 pi         21   1 2904M  268M 92016 S  3.3  7.1  1h53:17 /usr/lib/aarch6
   3709 root       20   0 2572M  147M 28728 S  0.7  3.9 40:10.80 /app/radarr/bin
   4648 pi         20   0  246M  112M  5488 S  0.7  3.0  2:13.98 /usr/bin/python
  11539 pi         20   0 6799M  838M  787M S  0.7 22.1  1h24:12 /usr/bin/python
  56227 pi         21   1 2904M  268M 92016 S  0.7  7.1  3:11.41 /usr/lib/aarch6
  56230 pi         20   0 2904M  268M 92016 S  0.7  7.1 22:15.46 /usr/lib/aarch6
  56290 pi         21   1 2904M  268M 92016 S  0.7  7.1 41:25.08 /usr/lib/aarch6
      1 root       20   0  161M  6116  3712 S  0.0  0.2  4:52.42 /sbin/init spla
    134 root       20   0 75740 25456 24880 S  0.0  0.7  3:29.47 /lib/systemd/sy
    174 root       20   0 22228  1340   700 S  0.0  0.0  0:02.82 /lib/systemd/sy
    390 systemd-t  20   0 88108  1732  1584 S  0.0  0.0  0:01.88 /lib/systemd/sy
    419 systemd-t  20   0 88108  1732  1584 S  0.0  0.0  0:00.01 /lib/systemd/sy
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Quit
```

Consumo de recursos media center. Elaboración propia



Consumo de recursos de Hassio. Elaboración propia

Si tu idea es usar la Raspberry Pi para jugar probablemente te lleves una decepción pues la potencia de su GPU no es suficiente para correr juegos modernos.

Audio

- 2 × micro-HDMI ports (up to 4kp60 supported): Por el conector HDMI además de audio tenemos vídeo.
- 4-pole stereo audio and composite video port

Vídeo

- 2 × micro-HDMI ports (up to 4kp60 supported)
- 2-lane MIPI DSI display port: Orientado a conectar una pantalla. Mas información aquí.
- 2-lane MIPI CSI camera port: Orientado a conectar una cámara. Mas información aquí.

Conectividad

En cuanto a su conectividad la Raspberry nos ofrece en este modelo la posibilidad de conectarnos a la red a través de WIFI, cable de red o Bluetooth. En concreto estas son las posibilidades que ofrece:

- 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless
- Bluetooth 5.0, BLE
- Gigabit Ethernet

> Si optas por instalar Raspberry Pi OS sin entorno gráfico deberás configurar la red desde el terminal. En este manual paso a paso te indican como hacerlo: https://bricolabs.cc/wiki/guias/raspberry_pi_-_instalacion_del_sistema_operativo_y_configuracion_de_red

Pinout

> El pinout que vamos a ver en este apartado es el del modelo de la Raspberry Pi 4 modelo B. Si no estás usando ese modelo asegúrate de cual es el pinout de tu dispositivo a fin de evitar conexiones incorrectas que dañen el dispositivo y/o los elementos que tengas conectados. No obstante la Raspberry Pi 4 tiene compatibilidad hacia atrás.

Información obtenida de https://www.raspberrypi.com/documentation/computers/raspberry-pi.html
Este modelo dispone de 40 pines.

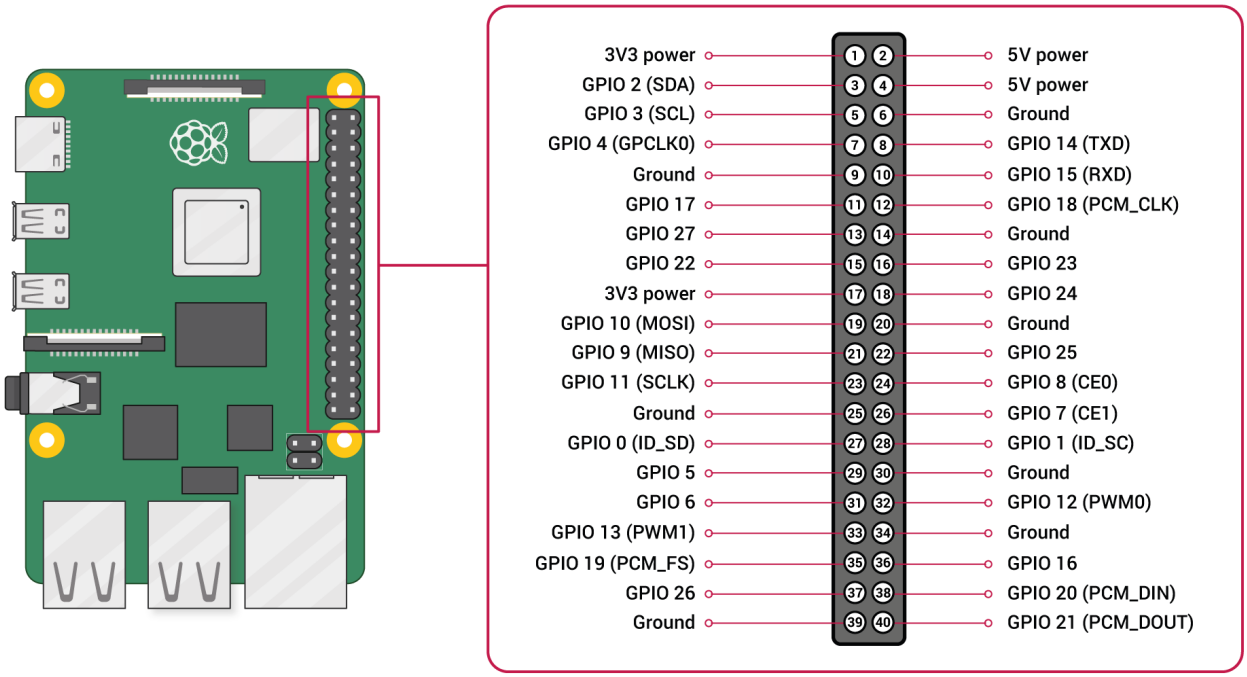


Imagen obtenida de https://www.raspberrypi.com/documentation/computers/raspberry-pi.html

Fíjate en la posición en la cual está colocada la Raspberry Pi y en como numera los pines (de izquierda a derecha y de arriba a abajo).

> Una cuestión importante es no confundir el nº de pin con el nº de GPIO

Si te fijas en el pin nº 7 corresponde al GPIO nº 4. Cuando en el capítulo 4, a través de Python, queramos interactuar con el pin nº7 deberemos referenciarlo como GPIO nº 4.

En relación a los voltajes fíjate que hay 2 pines que funcionan a 5V y otros 2 que funcionan a 3,3V. También hay varios marcados como tierra (ground)

Los pines GPIO se consideran activos (a 1, en alto) con 3,3V mientras que se consideran inactivos (a 0, en bajo) con 0 V.

Como curiosidad, si desde el terminal de tu sistema operativo raspbian ejecutas el comando `pinout` verás algo similar a:

```
pi@mediacenter:~ $ pinout
,--------------------------------.
| oooooooooooooooooooo J8   +======
| 1oooooooooooooooooooo  PoE |   Net
|  Wi                     lo +======
|  Fi   Pi Model 4B  V1.1 oo      |
|        ,----. +---+          +====
| |D|    |SoC | |RAM|          |USB3
| |S|    |    | |   |          +====
| |I|    `----' +---+             |
|                   |C|         +====
|                   |S|         |USB2
| pwr   |hd|   |hd| |I||A|      +====
`-| |---|m0|---|m1|----|V|-------'

Revision           : c03111
SoC                : BCM2711
RAM                : 4GB
Storage            : MicroSD
USB ports          : 4 (of which 2 USB3)
Ethernet ports     : 1 (1000Mbps max. speed)
Wi-fi              : True
Bluetooth          : True
Camera ports (CSI) : 1
Display ports (DSI): 1

J8:
   3V3  (1) (2)  5V
 GPIO2  (3) (4)  5V
 GPIO3  (5) (6)  GND
 GPIO4  (7) (8)  GPIO14
   GND  (9) (10) GPIO15
GPIO17 (11) (12) GPIO18
GPIO27 (13) (14) GND
GPIO22 (15) (16) GPIO23
   3V3 (17) (18) GPIO24
GPIO10 (19) (20) GND
 GPIO9 (21) (22) GPIO25
GPIO11 (23) (24) GPIO8
   GND (25) (26) GPIO7
 GPIO0 (27) (28) GPIO1
 GPIO5 (29) (30) GND
 GPIO6 (31) (32) GPIO12
GPIO13 (33) (34) GND
GPIO19 (35) (36) GPIO16
GPIO26 (37) (38) GPIO20
   GND (39) (40) GPIO21

POE:
TR01 (1) (2) TR00
TR03 (3) (4) TR02

For further information, please refer to https://pinout.xyz/
```

Elaboración propia