


# Semana 3. Planificación de Procesos y Servicios Rpi.

SISTEMAS EMBEBIDOS   
(EMBEDED SYSTEMS)

Grado Dual en Industria Digital

Campus Vitoria

Curso 2020-2021

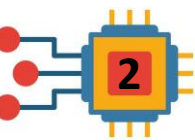


**Deusto**

Facultad de Ingeniería  
Ingeniaritza Fakultatea

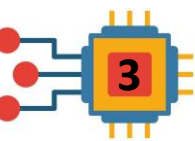
# Proceso de arranque

Que ocurre cuando se enciende la Rpi ??



# Proceso de arranque Raspberry Pi 4

1. BCM2711 SoC powers up
2. On board bootrom checks for bootloader recovery file (recovery.bin) on the SD card. If found, it executes it to flash the EEPROM and recovery.bin triggers a reset.
3. Otherwise, the bootrom loads the main bootloader from the EEPROM.
4. Bootloader checks it's inbuilt BOOT\_ORDER configuration item to determine what type of boot to do.
  - a) SD Card
  - b) Network
  - c) USB mass storage



# Proceso de arranque Raspberry Pi 4

- The Raspberry Pi uses a configuration file instead of the BIOS you would expect to find on a conventional PC.
- The system configuration parameters, which would traditionally be edited and stored using a BIOS, are stored instead in an optional text file named config.txt. This is read by the GPU before the ARM CPU and Linux are initialised.

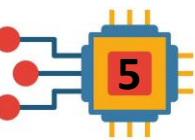
Detalles de config.txt:

<https://www.raspberrypi.org/documentation/configuration/config-txt/>

# Proceso de arranque

- You can display the currently-active configuration using  
`$ vcgencmd bootloader_config`
- To change these bootloader configuration items, you need to extract the configuration segment, make changes, re-insert it, then reprogram the EEPROM with the new bootloader.
- The Raspberry Pi will need to be rebooted for changes to take effect.

[https://www.raspberrypi.org/documentation/hardware/raspberrypi/bcm2711\\_bootloader\\_config.md](https://www.raspberrypi.org/documentation/hardware/raspberrypi/bcm2711_bootloader_config.md)



# Boot order fields

The BOOT\_ORDER property defines the sequence for the different boot modes. It is read right to left and up to 8 digits may be defined.

0x0 - NONE (stop with error pattern)

0x1 - SD CARD

0x2 - NETWORK

0x3 - USB device boot - Reserved - Compute Module only.

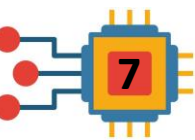
0x4 - USB mass storage boot (since 2020-09-03)

0xf - RESTART (loop) - start again with the first boot order field. (since 2020-09-03)

- Boot mode 0x0 will retry the SD boot if the SD card detect pin indicates that the card has been inserted or replaced.
- The default boot order is 0xf41 which means continuously try SD then USB mass storage.

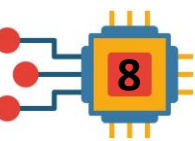
# RaspberryPi 4 boot EEPROM

<https://www.raspberrypi.org/documentation/hardware/raspberrypi/boot EEPROM.md>



# Servicios en Linux

- Systemd is an init system and system manager that is widely becoming the new standard for Linux machines.
- **systemctl** is a controlling interface and inspection tool for the widely-adopted init system and service manager systemd.





# Comandos Basicos de Systemcl

Acción	systemd	SysV
<b>Arrancar</b> un servicio	systemctl start foo	service foo start
<b>Detener</b> un servicio	systemctl stop foo	service foo stop
<b>Reiniciar</b> un servicio	systemctl restart foo	service foo restart
<b>Recargar el archivo de configuración</b> de un servicio (en systemd no todos los servicios lo soportan)	systemctl reload foo	service foo reload
<b>Rearrancar</b> un servicio que ya se encuentra en ejecución	systemctl condrestart foo	service foo condrestart
Mostrar el <b>estado</b> del servicio	systemctl status foo	service foo status
<b>Activar</b> un servicio para que sea ejecutado durante el arranque	systemctl enable foo	chkconfig foo on
<b>Desactivar</b> un servicio para que no sea ejecutado durante el arranque	systemctl disable foo	chkconfig foo off
Muestra el <b>estado</b> de un servicio durante el <b>arranque</b>	systemctl is-enable foo (1*)	chkconfig --list foo
<b>Crear o modificar el archivo de configuración</b> de un servicio	systemctl daemon-reload	chkconfig --add foo
<b>Listar los modos de ejecución</b> para los que un servicio está activado o desactivado	ls /etc/systemd/system/*.wants/foo.service	chkconfig

# Actividad

Dentro de `/etc/systemd/system` logeados como super usuario creamos un fichero llamado “myscrip.service” que llame a un shell script. Este script debe estar alojado en tu home dentro de un directorio llamado “myplanifiedscript ” y se tiene que llamar `savetime.sh`

El script `savetime.sh` debe guardar la hora actual en el fichero `/home/pi/myplanifiedscript/starttime.txt` (sobreescribiendo cualquier contenido anterior):

- Para probar si el servicio funciona correctamente `$sudo systemctl start myscript.service`
- Para pararlo, `$sudo systemctl stop myscript.service`
- Para habilitarlo al inicio, ejecutamos: `$sudo systemctl enable myscript.service`

# Comandos Basicos de Systemcl

Ejemplo del archivo: myscript.service

```
[Unit]
Description=My service
After=network.target

[Service]
ExecStart=/bin/bash savetime.sh
WorkingDirectory=/home/pi/myplanifiedscrip
t

StandardOutput=inherit
StandardError=inherit

Restart=always
User=pi

[Install]
WantedBy=multi-user.target
```

# Comandos Basicos de Systemcl

Ejemplo del archivo: savetime.sh

```
date >> starttime.txt
```

# Planificar procesos con crontab

```
$crontab -e
```

Seleccionar nano o vim según os sentáis más cómodos

```
$crontab -l
```

Lista la planificación actual

\* \* \* \* \*

Commands to be executed

Day of week (0-6) (Sunday = 0)

Month (1-12)

Day of Month (1-31)

Hour (0-24)

Minute (0-59)

# Planificar procesos con crontab



Keyword	Equivalent
@yearly	0 0 1 1 *
@daily	0 0 * * *
@hourly	0 * * * *
@reboot	Run at startup.

<https://crontab.guru/>

## Ejemplos

- Realizar backups 1 vez por semana
- Realizar backups al final del día
- Realizar un tarea en un intervalo de horas determinado en jornada laboral
- Ejecutar una tarea cada 5 minutos

# Actividad

Para la planificación de tareas se usan cron y crontab :

1. Comprobar si tenemos instalado el paquete gnome-schedule (pista: apt).

2. Entrar en el fichero de planificación de tareas con (ensayar su uso en:

<https://crontab.guru/>):

`crontab -e` #seleccionar nano como editor la primera vez

3. Añadir una línea para ejecutar una tarea todos los lunes a las 13:05: como:

```
05 13 * * 1 /bin/date >> /home/pi/disklog.txt ; /bin/df / >> /home/pi/disklog.txt
```

4. Añadir otra línea para ejecutar una tarea cada 5 minutos continuamente:

```
*/5 * * * * /bin/date >> /home/pi/disklog.txt ; /bin/df / >> /home/pi/disklog.txt
```

5. Comprobar que la tarea se ejecuta cada 5 minutos.