UNIVERSITÉ
CÔTE D'AZUR

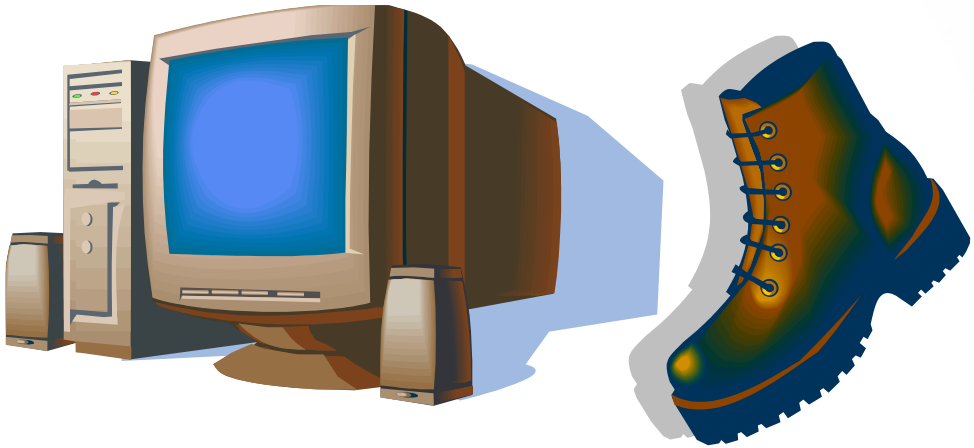
Introduction aux Systèmes et Logiciels Embarqués

Présentation: **Stéphane Lavirotte**
Auteurs: ... et al*

(*) Cours réalisé grâce aux documents de :
Stéphane Lavirotte, Michael Opdenacker (Bootlin),
François Mocq (Framboise314)

Mail: Stephane.Lavirotte@unice.fr
Web: http://stephane.lavirotte.com/
Université Nice Sophia Antipolis

UNIVERSITÉ
CÔTE D'AZUR



« Booter » un Système

Depuis le démarrage de la machine…

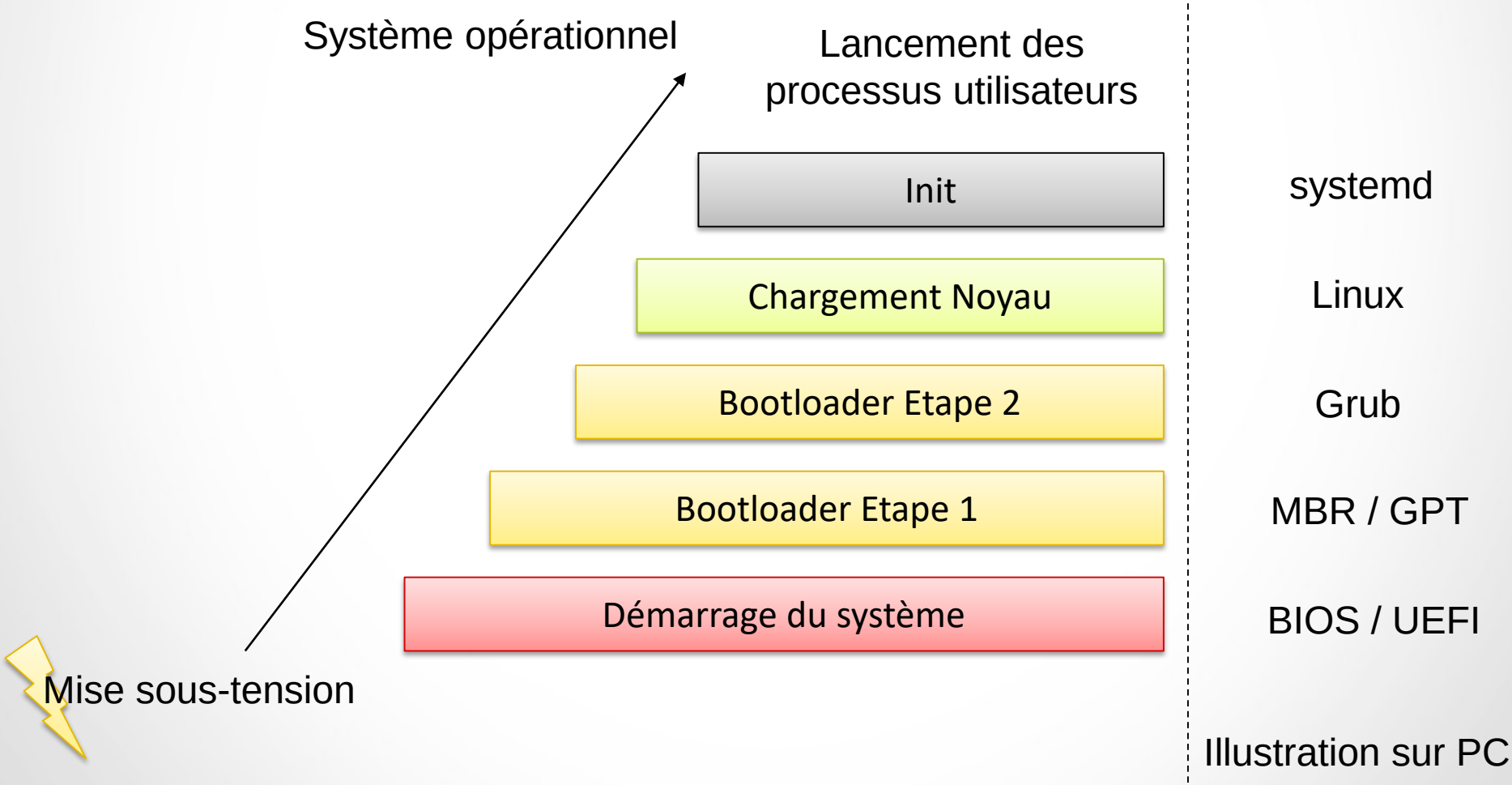
UNIVERSITÉ CÔTE D'AZUR

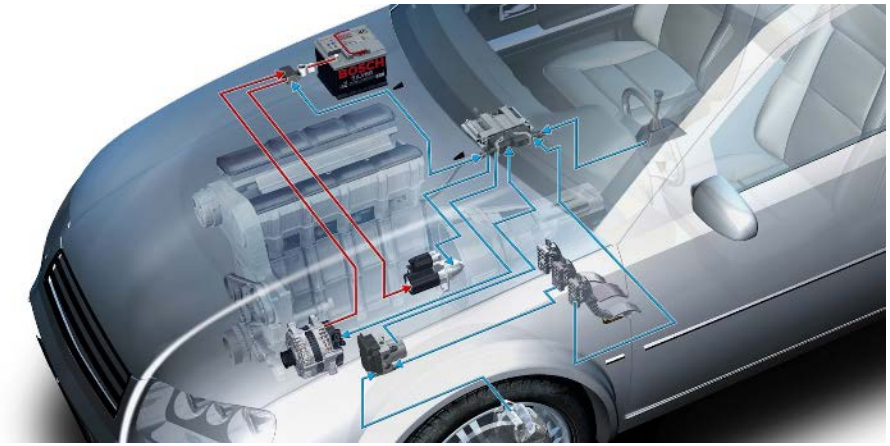
Étapes de Chargement du Système d'Exploitation

- ✓ Processus complexe
- ✓ Chargement successifs de plusieurs programmes
  - Charger le noyau est compliqué et ne peut être réalisé en une seul opération
    - Trop volumineux pour le loger en mémoire morte
    - Le programme de chargement du noyau est lui-même un programme qu'il faut charger avec un programme chargeur !
  - Le périphérique ou trouver le noyau est paramétrable
    - Disque dur puis CD seulement
    - Périphérique USB puis CD puis disque dur
    - Réseau
    - ...
- ✓ Le système de chargement est **dépendant** de la plateforme

UNIVERSITÉ CÔTE D'AZUR

Une vision macro





Démarrage d'un PC

Un processus complexe avec de nombreuses étapes

UNIVERSITÉ CÔTE D'AZUR

Du BIOS... à l'UEFI

- ✓ BIOS: Basic Input Output System
  - Microprogramme (*firmware*) sur une EEPROM de la carte mère
  - Effectue les opération élémentaires lors de la mise sous tension
  - Développé en Assembleur (évolue peu)
  - Un système efficace mais vieillissant
    - Exécuté en mode 16 bits et 1Mo de mémoire accessible

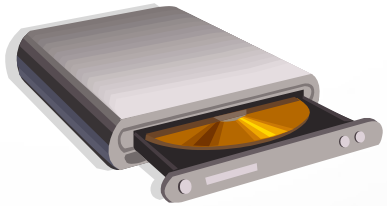
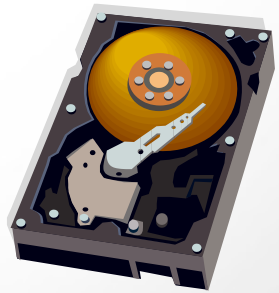
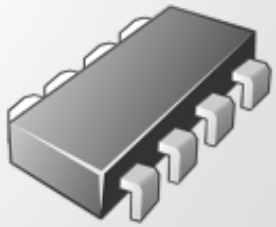
- ✓ UEFI: Unified Extensible Firmware Interface
  - Le successeur du BIOS
  - Développé en C: donne une souplesse pour les évolutions
  - Pris en charge par de nombreux systèmes
    - Linux depuis 2000, Mac OS X 10.4, Windows Vista SP1
  - http://www.uefi.org/

UNIVERSITÉ CÔTE D'AZUR

La Séquence de Boot : Exemple simple avec le BIOS

- ✓ Chaque processeur s'initialise
  - Self-test
  - Multiprocesseurs: élection d'un CPU leader
- ✓ CPU leader exécute les instructions en 0xffffffff0
- ✓ Instruction en 0xffffffff0
  - Saut vers le début du programme BIOS (Basic Input /Output System)
- ✓ BIOS: POST (Power On Self Test)
- ✓ BIOS: Choix d'un périphérique d'amorçage

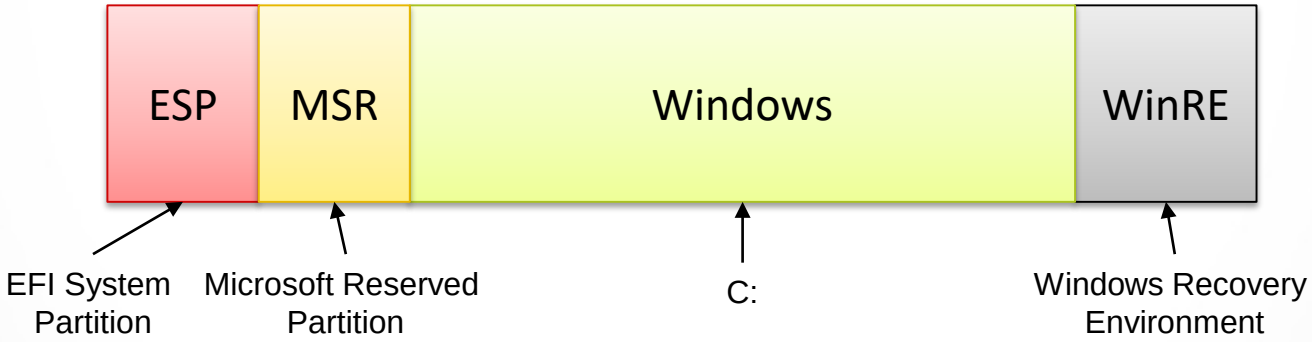


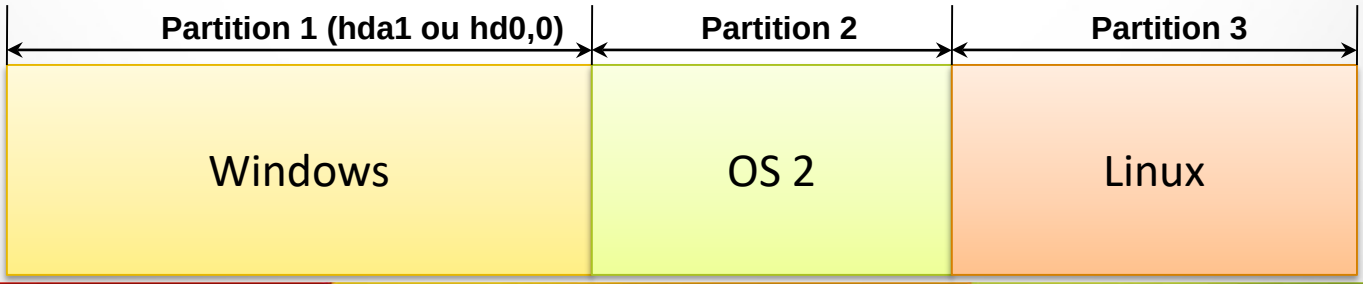
UNIVERSITÉ CÔTE D'AZUR

(Rappel sur) Le partitionnement

- ✓ Subdivision de l'espace de stockage en différent zone
  - Au moins une partition
  - **Nécessité d'une table de partition**
- ✓ Exemple de partitionnement Windows 10



- ✓ Exemple de partitionnement avec plusieurs OS



UNIVERSITÉ CÔTE D'AZUR

Table de Partitionnement

MBR

- ✓ Master Boot Record
  - Code d'amorçage (lancer le bootloader)
  - Table de partitionnement
  - Signature (0x55 AA)
- ✓ BIOS et UEFI
- ✓ Limitation
  - 4 partitions maximum
  - Taille partition < 2,2To

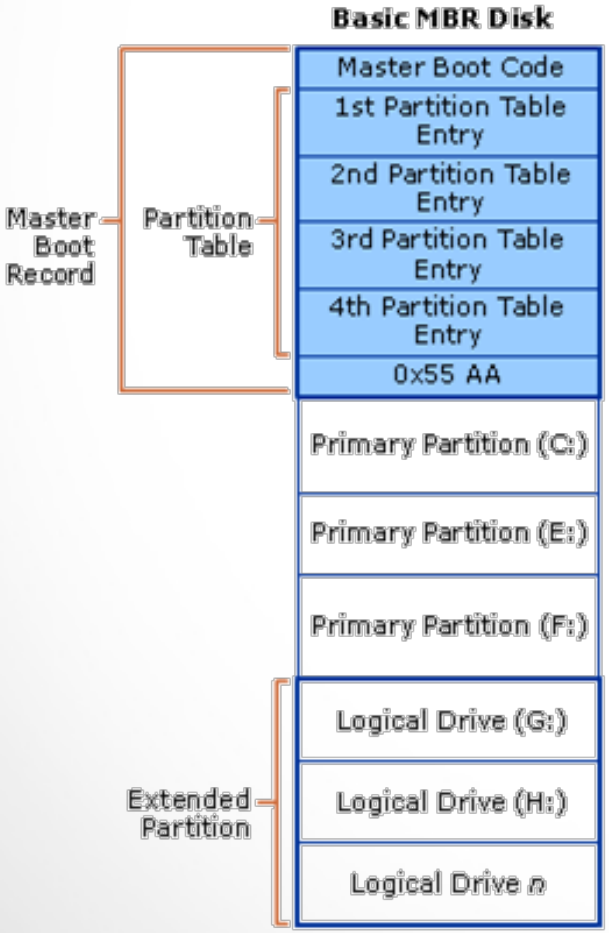
GPT

- ✓ GUID Partition Table
  - MBR protecteur
  - GPT primaire: entête
  - GPT secondaire: table partitions
- ✓ UEFI
- ✓ Limitation
  - 128 partitions
  - Taille partition < 256To
  - Pas pour Windows 32bits

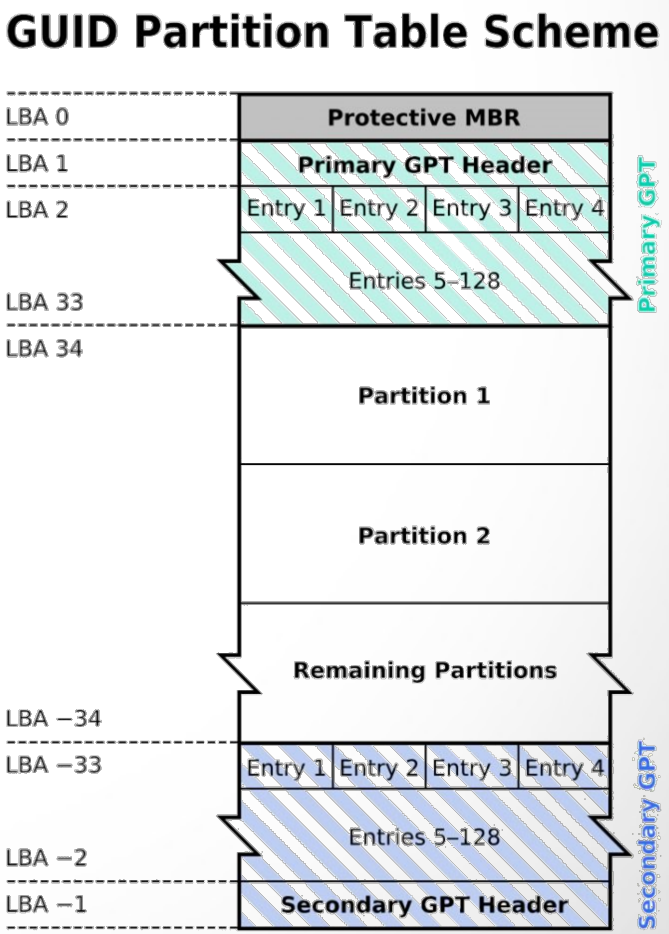
UNIVERSITÉ CÔTE D'AZUR

Table de Partitionnement

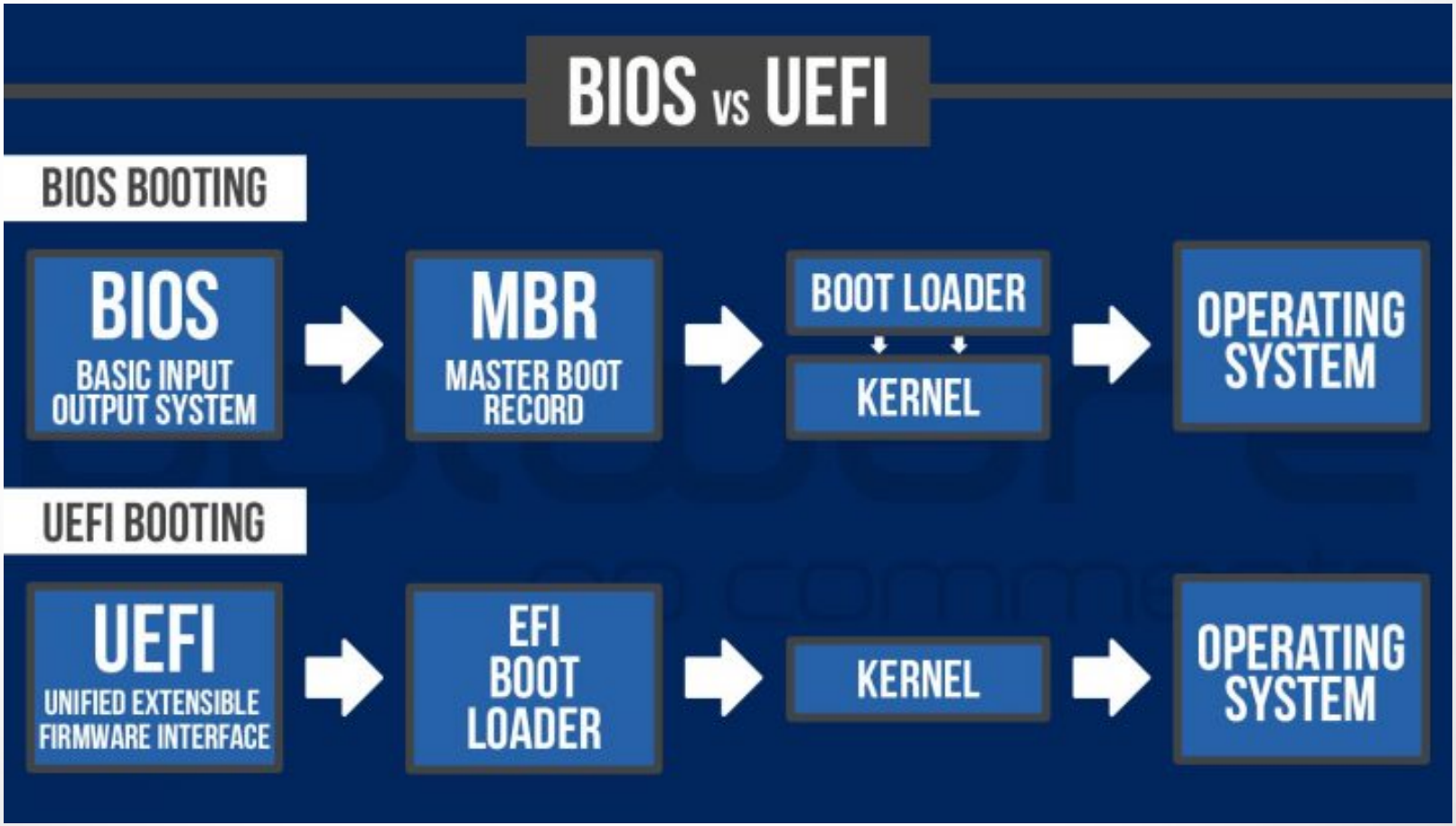
MBR



GPT



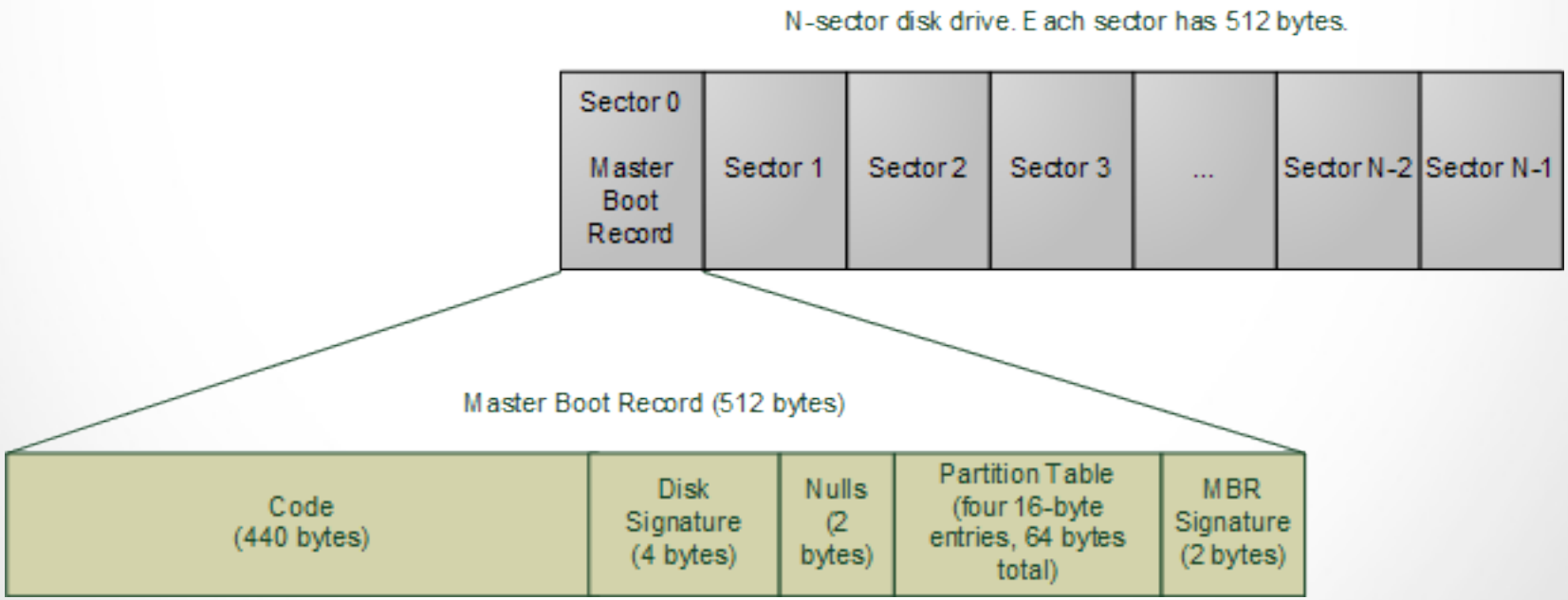
BIOS vs UEFI


BIOS vs UEFI
BIOS BOOTING
BIOS
BASIC INPUT OUTPUT SYSTEM
MBR
MASTER BOOT RECORD
BOOT LOADER
KERNEL
OPERATING SYSTEM
UEFI BOOTING
UEFI
UNIFIED EXTENSIBLE FIRMWARE INTERFACE
EFI BOOT LOADER
KERNEL
OPERATING SYSTEM

UNIVERSITÉ CÔTE D'AZUR

Exemple: Zoom sur le MBR

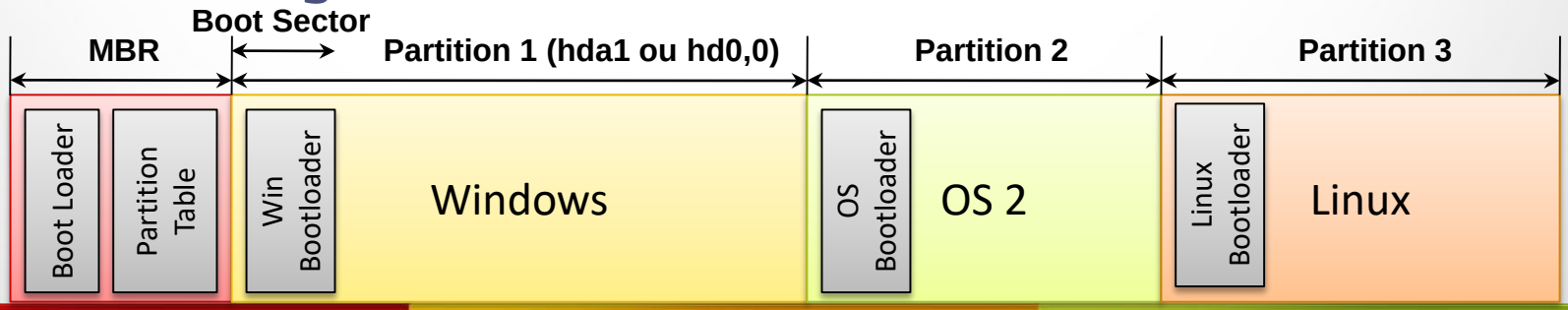
- ✓ Code sur 440 octets
- ✓ Ne peut être qu'un amorçage du bootloader complet (une redirection vers un programme plus complet)



UNIVERSITÉ CÔTE D'AZUR

La Séquence de Boot : Exemple du MBR

- ✓ BIOS: Charge le MBR (Master Boot Record) du périphérique de boot
  - 1er secteur (512 octets) du périphérique
- ✓ BIOS: inspecte le MBR:
  - Vérification (nombre magique, table des partitions)
  - Recherche le secteur de boot
  - Attention à l'écrasement si multi OS sur la machine
- ✓ BIOS: charge le secteur de boot
  - Charge le début du programme chargeur de Linux (Lilo, Grub) ou le chargeur d'un autre OS



UNIVERSITÉ CÔTE D'AZUR

La Séquence de Boot: Boot Loader

- ✓ Soit
  - Le programme du *boot sector* termine le chargement du *boot loader* (MBR)
  - Le boot sector déclenche le chargement du chargeur d'un autre OS qui chargera le premier secteur du Boot Loader
    - 1er secteur du Boot Loader installé dans le secteur de boot secondaire (en début de partition Linux)
    - On se retrouve alors dans la même situation, le Boot Loader termine son chargement
- ✓ Le Boot loader: charge le noyau et lance son exécution
  - Options possibles pour paramétrer le noyau

UNIVERSITÉ CÔTE D'AZUR

Chargeur de Noyau : LILO et GRUB

- ✓ LILO (LInux LOader)
  - Le chargeur « historique » (plus utilisé maintenant)
  - Outil ad hoc Linux (soit Linux soit Other)
  - Des limitations trop contraignantes
    - Premier secteur sur cylindre < 1024

- ✓ GRUB (GRand Unified Bootloader)
  - Le chargeur générique de GNU/FSF
  - Support natif pour de multiples OS (Linux, *BSD, ...)
  - Nombreuses améliorations (« micro shell », ...)

UNIVERSITÉ CÔTE D'AZUR

Configuration de GRUB

- ✓ **Fichier de configuration :**
  - `/boot/grub/grub.cfg`
  - **Fichier lisible assez facilement**
  - **Définition des différentes possibilités de démarrage**
  - **Passage de paramètres au noyau**

- ✓ **Identifiant des disques et partitions**
  - **Utilisation des UUID (nécessite un fichier `initrd` avec des scripts)**
  - **Ou utilisation de la dénomination `/dev/sda1` (depuis Grub 1.9x)**

UNIVERSITÉ CÔTE D'AZUR

Exemple de fichier `grub.cfg`

```
menuentry "Debian GNU/Linux" {
        set root=(hd0,3)
        linux /vmlinuz root=/dev/hda3 ro
        initrd /initrd.img
}
```

- ✓ *Les paramètres ont la signification suivante:*
  - `set root=(hd0,3)`
    - *utiliser la 3ème partition du premier disque*
  - `linux /vmlinuz root=/dev/hda3 ro`
    - *utiliser le noyau se trouvant en* `/vmlinuz` *avec les paramètres pour le lancement du noyau :* `root=/dev/hda3 ro`
    - *un UUID peut être utilisé pour identifier le périphérique spécial par bloc plutôt que par son nom de fichier* `root=UUID=81b289d5-…`
  - `initrd /initrd.img`
    - *utiliser l'image initrd/initramfs située en* `/initrd.img`

UNIVERSITÉ CÔTE D'AZUR

Installation de GRUB

- ✓ Connaitre le nom des périphériques
  - `blkid`
- ✓ Installer GRUB pour la première fois (ou pour tout réinstaller)
  - `grub-install /dev/sdX`
- ✓ Mettre à jour automatiquement le fichier `grub.conf` à partir des fichiers trouvés dans `/boot`
  - `update-grub`

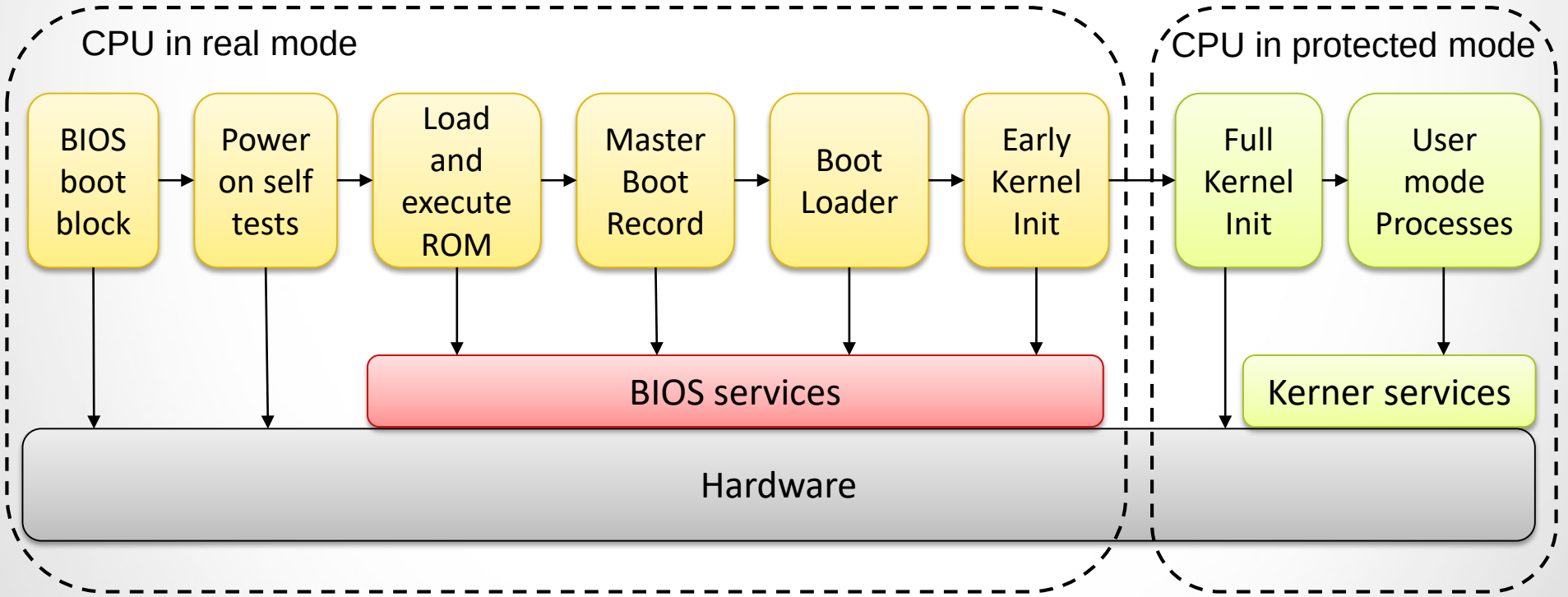
UNIVERSITÉ CÔTE D'AZUR

Paramétrage du Noyau au Démarrage

- ✓ Les paramètres sont passés par le bootloader
- ✓ Liste des paramètres du noyau dans le fichier
  - `/usr/src/linux/Documentation/kernel-parameters.txt`
- ✓ Exemple de paramètres:
  - Avec valeur:
    - `init=` : définit l'exécutable à lancer après avoir monté la racine
    - `root=` : définit la partition racine à monter via
      - UUID: identifiant unique d'une partition d'un disque dur
      - `/dev/...`: identification classique des périphériques sous Unix
    - `vga=` : donne le mode vidéo pour le démarrage
  - Sans valeur:
    - `ro(rw)` : définit que la partition racine doit être montée en lecture (ou en lecture écriture)
    - `quiet` : ne pas afficher les message de démarrage du noyau
    - `single`: démarrage en mode « single user »

UNIVERSITÉ CÔTE D'AZUR

Pour résumer: Les Grandes Etapes du Boot d'un PC

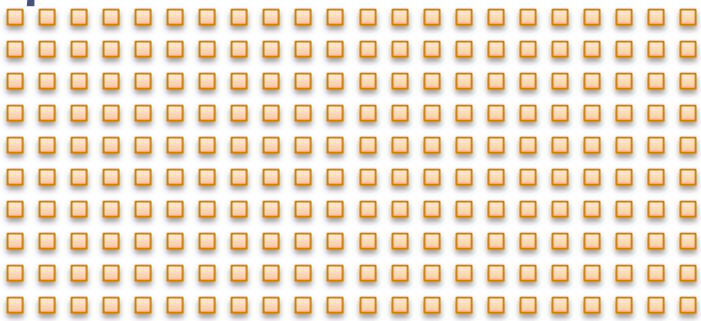


Démarrage d'une Raspberry Pi

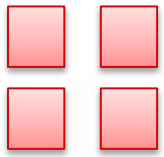
Un exemple de plateforme embarquée

GPU ?

- ✓ Graphics Processing Unit
  - Assure les fonctions de calcul pour l'affichage
  - Structure hautement parallèle
  - Mais cela reste un processeur (donc exécute du code)
- ✓ GPU
  - Milliers de cœurs pour traiter efficacement des tâches simultanées



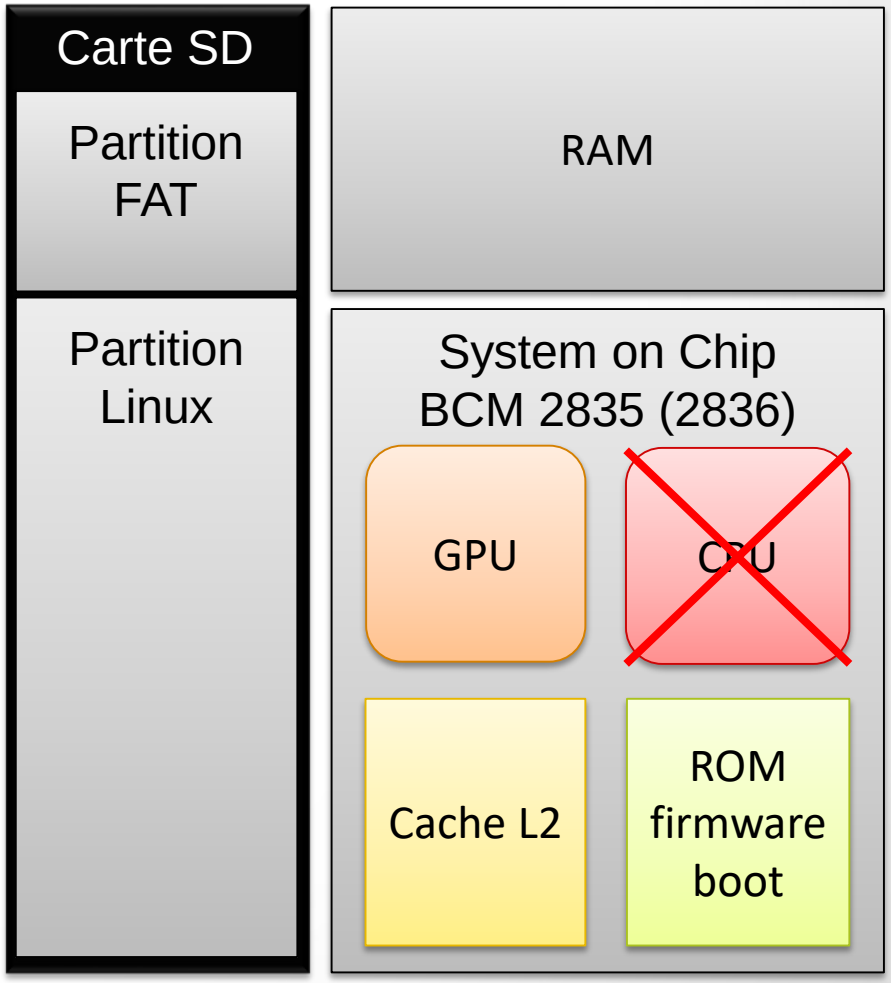
- ✓ CPU
  - Nombre restreint de cœurs optimisés pour un traitement en série



UNIVERSITÉ CÔTE D'AZUR

Etapes du Boot de la Rapberry Pi (1/5)

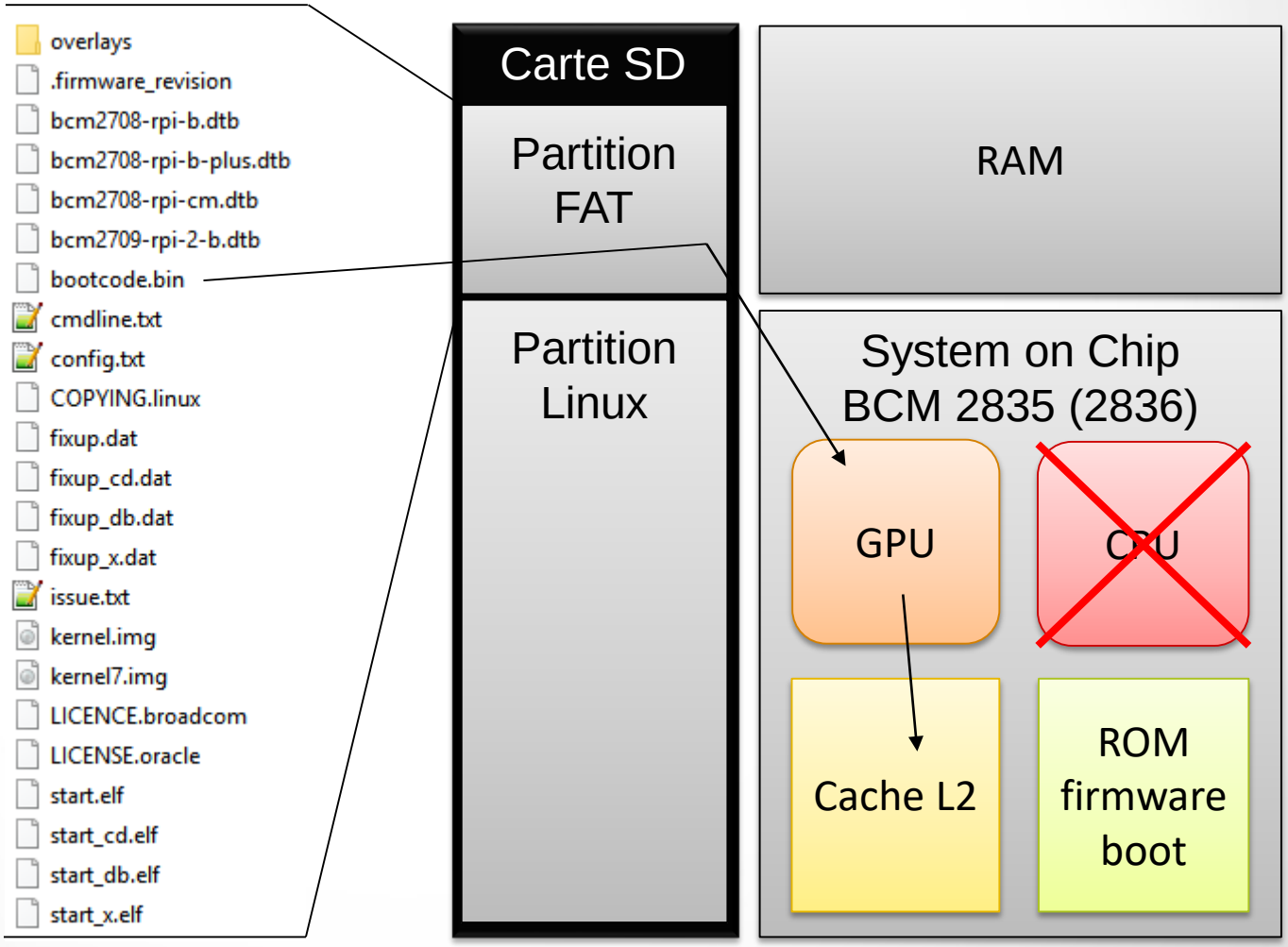
- ✓ Mise sous tension
- ✓ GPU activé, CPU en veille, carte SD désactivée
- ✓ Dans la ROM premier étage du bootloader
- ✓ Le GPU va exécuter le programme en ROM
- ✓ Permet d'accéder à la carte SD
- ✓ Accès à la première partition en FAT



UNIVERSITÉ CÔTE D'AZUR

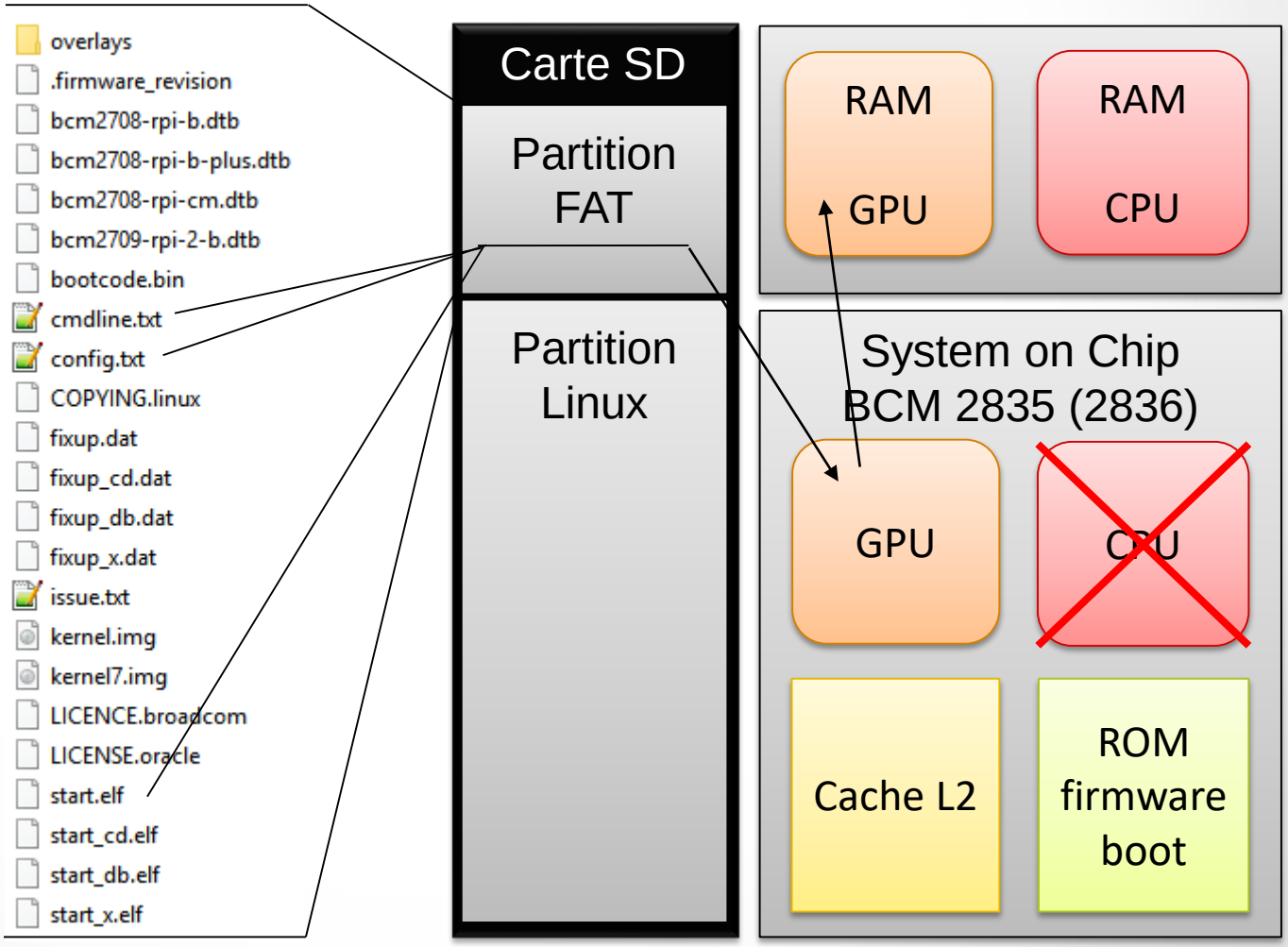
Etapes du Boot de la Rapberry Pi (2/5)

- ✓ Chargement de bootcode.bin dans le cache et exécution par le GPU
- ✓ Met en service la RAM en service



Etapes du Boot de la Rapberry Pi (3/5)

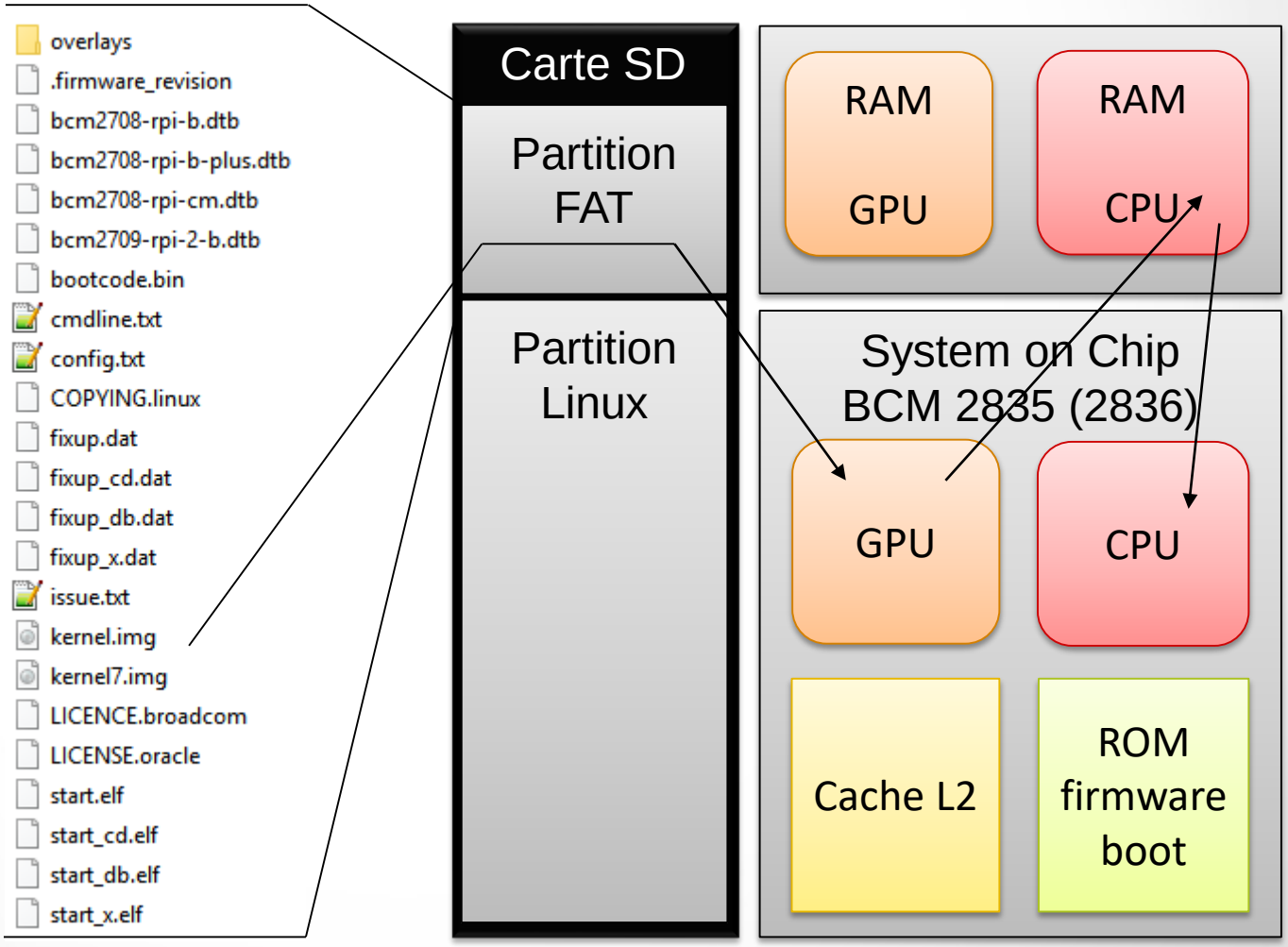
- ✓ Le GPU exécute start.elf
- ✓ Charge config.txt et cmdline.txt
- ✓ Séparation de la mémoire en 2 zones



UNIVERSITÉ CÔTE D'AZUR

Etapes du Boot de la Rapberry Pi (4/5)

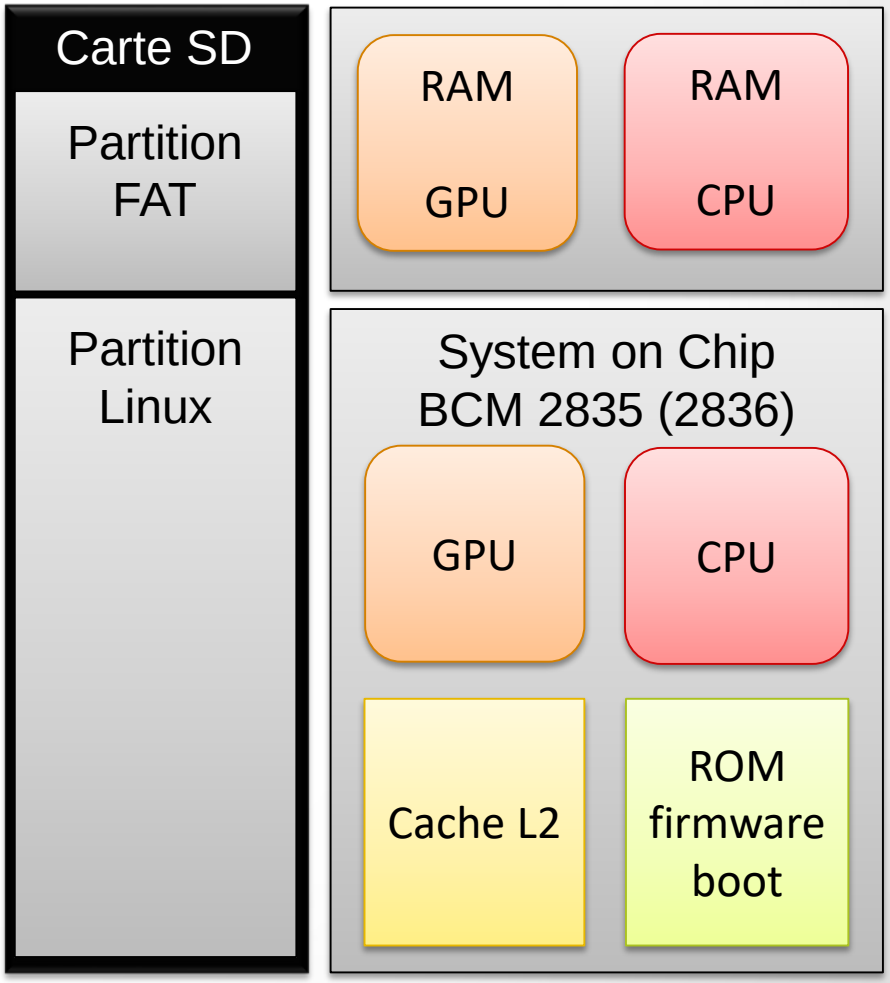
- ✓ Chargement du noyau
- ✓ (kernel.img sur Pi et kernel7.img sur Pi2)
- ✓ Exécution du noyau sur le CPU



UNIVERSITÉ CÔTE D'AZUR

Etapes du Boot de la Rapberry Pi (5/5)

- ✓ Exécution du noyau
- ✓ Montage de la partition racine (partition Linux)
- ✓ Fin de chargement du noyau
- ✓ Lancement du premier processus



UNIVERSITÉ CÔTE D'AZUR

Des Chargeurs pour l'Embarqué

- ✓ Chaque plateforme a ses spécificités
  - Chargeur adapté à la plateforme
- ✓ Uboot: Universal Bootloader
  - Le plus utilisé sur arm
  - http://uboot.sourceforge.net/
  - Matériel supporté: arm, ppc, mips, x86
- ✓ RedBoot: Chargeur de démarrage basé sur eCos de RedHat
  - http://sources.redhat.com/redboot/
  - Matériel supporté: x86, arm, ppc, mips, sh, m68k...

Démarrage des services

Systemd

(et rappel historique sur SysVinit)

UNIVERSITÉ CÔTE D'AZUR

Démarrage Premier Processus

- ✓ Le noyau est compressé!
  - Les 1ères instructions = programme de décompression
    - Décompresse le reste du noyau, puis lance l'exécution
- ✓ Le noyau commence son exécution
  - Initialise le noyau, détecte les périphériques, ...
  - Monte la racine en read-only
  - Lance le premier processus (processus de pid 1)
    - Le processus lancé par défaut est `/sbin/init`
    - Possibilité de spécifier un autre processus grâce `init=` lors du chargement du noyau
- ✓ Plusieurs systèmes possibles pour les scripts de service
  - System V init (1983) encore en fonction très récemment
  - systemd (2010) remplace System V init

UNIVERSITÉ
CÔTE D'AZUR

Après le lancement du premier processus

- ✓ Responsabilité de mettre en place le reste de l'espace utilisateur
  - Monter les systèmes de fichiers
  - Démarrer les services et démons
  - Charger les pilotes de périphériques manquants
  - ...

- ✓ Mais aussi:
  - Adoption des processus orphelin
    - On connait forcément ce processus qui a le PID 1 et c'est l'anc^tre de tous

UNIVERSITÉ
CÔTE D'AZUR

SysVinit

Le mécanisme historique

UNIVERSITÉ CÔTE D'AZUR

System V init ou « SysVinit »

- ✓ Le processus `init`
  - Lit le fichier `/etc/inittab`
  - Exécute les scripts rc (Run Control)
  - Se trouve dans un état (niveau) d'exécution: runlevel
- ✓ Sous Debian GNU/Linux
  - 0 : Arrêt du système (halt)
  - 1 : Mode utilisateur unique (single user mode)
  - 2-5 : Mode multi-utilisateurs (multi-user mode)
  - 6 : Redémarrer le système (reboot)
- ✓ On trouve la description dans `/etc/inittab`
- ✓ Les scripts pour le démarrage des services
  - Scripts Shell dans `/etc/init.d/`

UNIVERSITÉ CÔTE D'AZUR

Changement de runlevel

- ✓ Lorsque `init` change de niveau, des tâches sont déclenchées:
  - En fonction des règles dans le fichier `/etc/inittab`
  - En fonction des scripts de Run Control des répertoires `/etc/rcn.d`
    - `SXXyyyyy`: lors de l'entrée dans le niveau
    - `KXXyyyyy`: lors de la sortie du niveau
  - Exemples (liens vers les script dans `/etc/init.d/`)
    - `/etc/rc3.d/S02apache2`: démarre le serveur Web
    - `/etc/rc3.d/S03ssh`: démarre le serveur ssh
- ✓ Fichier de configuration `/etc/inittab`
  - Décrit les règles de fonctionnement de init
  - Contient des entrées de la forme:
    - `id:niveau:action:process`
    - `id` : identifiant (1- 4 caractères)
    - `niveau` : niveau d'exécution auquel la règle s'applique
    - `action` : action
    - `process` : le processus à exécuter

UNIVERSITÉ CÔTE D'AZUR

Définition de /etc/inittab

- ✓ Les actions possibles:
  - `respawn` : relance le processus
  - `once` : processus lancé une fois le niveau d'exécution atteint
  - `wait` : idem once + init attend la fin du processus
  - `boot` : exécuté lors du boot (runlevel ignoré)
  - `bootwait` : idem boot + attente terminaison
  - `off`: ne fait rien
  - `ondemande` : lorsque le niveau correspondant est atteint
  - `initdefault` : indique le niveau d'exécution par défaut
  - `sysinit` : exécuté durant le boot, mais avant boot et bootwait
  - `powerwait`, `powerfail`, `powerokwait`, `powerfailnow` : incident d'alimentation...
  - `ctrlaltdel` : action à exécuter lorsque init reçoit SIGINT (via ctrl+alt+del)
  - `kbrequest` : lorsqu'une combinaison spéciale est réalisée au clavier

UNIVERSITÉ CÔTE D'AZUR

Exemple de fichier /etc/inittab

- **Exemple** /etc/inittab:

```
# The default runlevel
id:2:initdefault:
# Boot-time system configuration/initialization
si::sysinit:/etc/init.d/rcS
# Define runlevels
l0:0:wait:/etc/init.d/rc 0
l1:1:wait:/etc/init.d/rc 1
...
# What to do when CTRL-ALT-DEL
ca:12345:ctrlaltdel:/sbin/shutdown –t1 -a -r now
#/sbin/getty invocations for the runlevel
1:2345:respawn:/sbin/getty 38400 tty1
2:23:respawn:/sbin/getty 38400 tty1
```

UNIVERSITÉ CÔTE D'AZUR

Conclusion sur SysVinit

- ✓ Mécanisme simple
  - Basé sur des fichiers textes pour le montage des partitions
  - Basé sur des scripts Shell pour le démarrage des services

- ✓ Inconvénients:
  - Outils hétérogène pour les différentes fonctionnalités gérées
    - Script Shell pour le démarrage d'un service
    - `/etc/fstab` pour la gestion du montage des partitions
  - Démarrage séquentiel des différents scripts
    - Démarrage (boot) de la machine lent et non optimisé

systemd

« system deamon »: Le démon du système

Gestionnaire de système et services

UNIVERSITÉ CÔTE D'AZUR

systemd

- ✓ Projet mené par Lennart Poettering en 2010
  - Développeur allemand, employé par la société Red Hat
- ✓ Adopté par la plupart des distributions

- ✓ Déplace le problème:
  - Du démarrage de tous les services (sysVinit)
  - A la gestion du système et de tous les services

- ✓ Organisation des processus en groupes (cgroups)
  - Supervision des services
  - Contrôle des services et de leur environnement
- ✓ Utilise une syntaxe déclarative pour gérer les unités

UNIVERSITÉ CÔTE D'AZUR

Unités de systemd

- ✓ Les tâches de systemd sont organisées en tant qu'unités de différents types
  - `.services`: Démons et Services (init.d)
  - `.socket`: Démons sur le démarrage de Sockets (inetd)
  - `.device`: Pilotes de Périphériques (udev)
  - `.mount`: Points de montage (fstab)
  - `.automount`: Périphériques amovibles (autofs)
  - `.swap`: Partitions et fichiers de swap (fstab)
  - `.target`: Groupe d'unités et point de synchronisation
  - `.path`: Changement de système de fichier (inotify)
  - `.timer`: Ordonnancement de services (crond)
  - `.snapshot`: Sauvegarde de l'état de tous les service
  - `.slice`: Limite les ressources (cgroup)
  - `.scope`: Regroupe des processus arbitraires (cgroup)

UNIVERSITÉ CÔTE D'AZUR

cgroups et targets

- ✓ Systemd place chaque service dans un groupe de contrôle (`cgroup`)
  - Isolation des processus et allocation des ressources en fonction des groups de contrôle
- ✓ Les cibles (`targets`) sont des groups d'unités
  - `graphical.target`: toutes les unités pour démarrer un poste de travail avec une interface utilisateur graphique
- ✓ Créé et gère les sockets pour la com entre les composants du système
  - Ex: créé d'abord `/dev/log` puis démarre le démon `syslog`:
    - les processus utilisant `syslog` peuvent démarrer en parallèle
    - les services peuvent être redémarrés sans perte de données (le noyau mettra en mémoire tampon pendant que le processus redémarre)

UNIVERSITÉ CÔTE D'AZUR

Exemple « service simple »

```
[Unit]
Description=Deluge Bittorrent Client Daemon
After=network-online.target

[Service]
Type=simple

User=deluge
Group=deluge
UMask=007

ExecStart=/usr/bin/deluged -d

Restart=on-failure

# Configures the time to wait before service is stopped forcefully.
TimeoutStopSec=300

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

UNIVERSITÉ CÔTE D'AZUR

Exemple de service « timer »

goodmorning.service

```
[Unit]
Description=good morning
RefuseManualStart=true
RefuseManualStop=true
ConditionACPower=true

[Service]
Type=oneshot
ExecStart=/bin/systemd-inhibit \
    --what=handle-lid-switch   \
    --why=goodmorning          \
    /bin/su mde -c             \
        "/usr/bin/mplayer
/home/mde/Downloads/song.mp3"
```

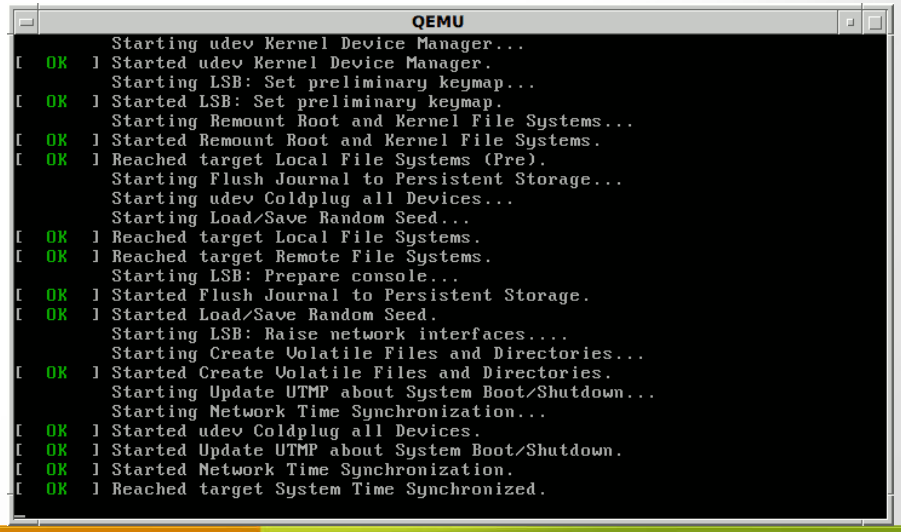
goodmorning.timer

```
[Unit]
Description=good morning

[Timer]
Unit=goodmorning.service
OnCalendar=Fri *-*-* 7:00
WakeSystem=true
Persistent=false

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

UNIVERSITÉ CÔTE D'AZUR

Administration systemd

- ✓ **Administration des services avec systemd**
  - **Commande** `systemctl` *`cmd unit`*
    - `start`, `stop`, `restart`: **démarre, arrête ou redémarre un service**
    - `status`: **fournit des informations sur l'état du service**
    - `enable`, `disable`: **active ou désactive le service au démarrage**
- ✓ **Lorsque** `systemd` **est lancé par le noyau:**
  - **Lit le fichier** `/etc/systemd/system.conf`
- ✓ **Power Management**
  - `systemctl reboot`
  - `systemctl poweroff`
  - `systemctl suspend`
  - `systemctl hibernate`
  - `systemctl hybrid-sleep`



UNIVERSITÉ CÔTE D'AZUR

Fichier par défaut et surcharge

- ✓ `/lib/systemd/system`:
  - Valeurs par défaut de systemd
  - Ne pas changer les fichiers dans ce dossier system
- ✓ `/etc/systemd/system`:
  - Modifications locales par surcharge et extension
  - Remplacement d'une unité dans `/lib` par une du même nom dans `/etc`
  - Ajouter des services au démarrage:
    - Lien symbolique dans `/etc/systemd/system/default.target.wants`
  - Masquer une unité avec un lien symbolique sur `/dev/null`

UNIVERSITÉ CÔTE D'AZUR

SysVinit ou Systemd ?

- ✓ Polémique et Débats virulents sur systemd
  - Fork de la Debian
- ✓ Avantages de systemd
  - Introduit dans de nombreuses distributions: Fedora (2011), RedHat Entreprise (2014), Debian et Ubuntu (2015)
  - Efficace car démarrage des processus en parallèle
  - Permet l'analyse du temps de démarrage
- ✓ Inconvénients de systemd
  - Changement profond et plus complexe que SysVinit
  - Système complexe avec de nombreuses commandes
  - Pas dans la philosophie KISS (Keep It Simple, Stupid)
- ✓ Et dans l'embarqué...
  - Souvent besoin d'une maîtrise fine du démarrage
  - Quoi qu'il arrive, être le plus efficace possible
  - ➢ Beaucoup écrivent un script Shell spécifique pour le démarrage

Informations complémentaires

Les évaluations...

UNIVERSITÉ CÔTE D'AZUR

Evaluation du Module

- ✓ Contrôle continu
  - Donc au fur et à mesure du module
- ✓ Ce que j'attends de vous:
  - **Apprendre** de nouvelles connaissances
  - **Mettre en œuvre** ses nouvelles connaissances
  - **Comprendre** ce que vous faites... pour...
  - **Analyser** un problème à l'aide de ces connaissances
- ✓ Plusieurs évaluations:
  - QCM rapides sur les connaissances
  - Rendre un ou plusieurs TD pour la mise en œuvre
  - Répondre à des questions de synthèse et d'analyse