Flash fileSystems

Mg. Ing. Gonzalo E. Sanchez
MSE - 2020

Implementación de Sistemas Operativos II

Flash fileSystems

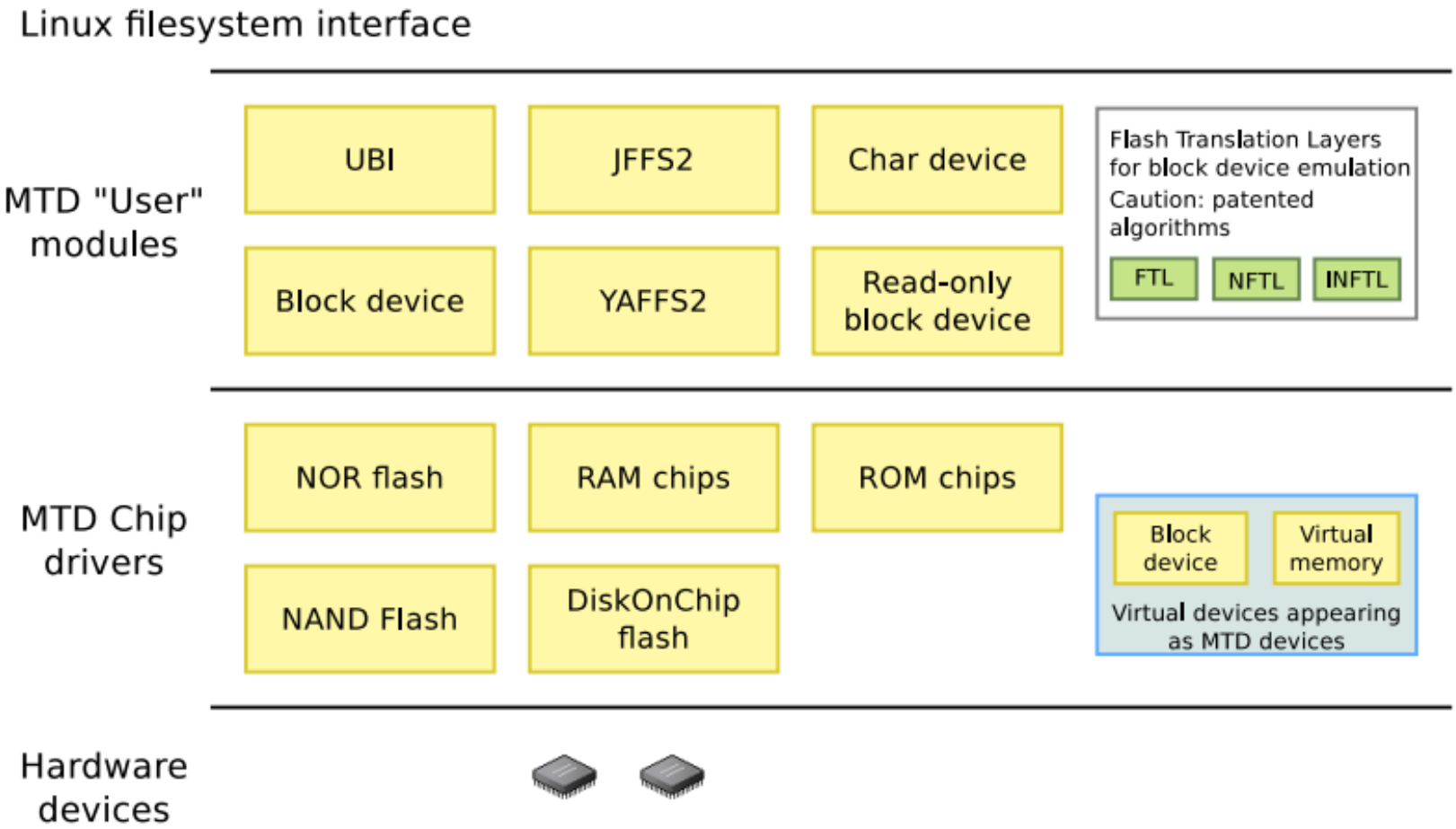
- Introducción

Introducción

Flash fileSystems

- Existe un subsistema llamado MTD.
- MTD es el acrónimo de Memory Technology Devices.
- Todos los dispositivos MTD son visibles en **/proc/mtd**.
- El driver **mtdchar** crea un char device para cada dispositivo MTD en el sistema.
- Lo normal es que este dispositivo sea nombrado **/dev/mtdX** o **/dev/mtdXro** utilizando el número major 90.
- Minor numbers pares para read-only, impares para read-write

Flash fileSystems



Flash fileSystems

- Además el driver **mtdchar** provee una función **ioctl()** para borrar y administrar la memoria flash.
- En particular este driver es usado por las utilidades **mtd-utils**.
- También se tiene el driver **mtdblock** que crea un block device para cada dispositivo MTD en el sistema.
- Nombrados **/dev/mtdblockX** utilizan el major number 31, y el minor es el número del dispositivo.
- Permite lectura/escritura a nivel bloque. Utilizado para montar filesystems.

Flash fileSystems

- Los dispositivos MTD normalmente están particionados.
- Permite el uso de distintas áreas de flash para distintos propósitos:
  - Read-only filesystem.
  - Área de bootloader.
  - Área de kernel.
  - Áreas de backup.
- A diferencia de los dispositivos block, las particiones para dispositivos MTD son descritas externamente.

Flash fileSystems

- Métodos de descripción de tabla de particiones para MTD:
  - Device Tree.
  - Definida en el kernel (si no hay soporte de DT).
  - Especificada a través de la línea de comandos del kernel.
- Cada partición se vuelve un dispositivo MTD separado.
  - No se numeran como un block device.
  - Ejemplo: **/dev/mtd1** puede ser la segunda partición del primer dispositivo flash, o la primera partición del segundo dispositivo.

Flash fileSystems

- En general el DT es el lugar estándar para especificar una partición MTD.

- Ejemplo: **arch/arm/boot/dts/omap3-igep0020.dts**

Flash fileSystems

```
        nand@0,0 {
                linux,mtd-name= "micron,mt29c4g96maz";
[...]
                partition@0 {
                        label = "SPL";
                        reg = <0 0x100000>;
                };
                partition@0x80000 {
                        label = "U-Boot";
                        reg = <0x100000 0x180000>;
                };
[...]
                partition@0x780000 {
                        label = "Filesystem";
                        reg = <0x680000 0x1f980000>;
                };
```

Flash fileSystems

- Para placas o plataformas que no utilizan DT, se pueden definir las particiones en el kernel.
- Esto ya no es comúnmente utilizado.
- Ejemplo: arch/arm/mach-omap2/board-igep0020.c

- **NOTA:** Este archivo fue removido en el kernel 3.13.

Flash fileSystems

```
static struct mtd_partition igep2_flash_partitions[] = {
    {
        .name   = "X-Loader",
        .offset = 0,
        .size   = 2 * (64*(2*2048))
    },
    {
        .name   = "U-Boot",
        .offset = MTDPART_OFS_APPEND,
        .size   = 6 * (64*(2*2048)),
    },
    [...]
    {
        .name   = "File System",
        .offset = MTDPART_OFS_APPEND,
        .size   = MTDPART_SIZ_FULL,
    },
};
```

Flash fileSystems

- Afortunadamente las particiones MTD pueden ser definidas a través de la linea de comandos del kernel.
- Se necesita el nombre del dispositivo MTD.
- Ejemplo:

```
NAND device: Manufacturer ID: 0x2c, Chip ID: 0xbc (Micron NAND 512MiB 1,8V 16-bit)
Creating 5 MTD partitions on "omap2-nand.0":
0x000000000000-0x000000080000 : "X-Loader"
0x000000080000-0x000000200000 : "U-Boot"
0x000000200000-0x000000280000 : "Environment"
0x000000280000-0x000000580000 : "Kernel"
0x000000580000-0x000020000000 : "File System"
```

Flash fileSystems

- Se utiliza el parámetro de booteo de kernel **mtdparts**.
- Ejemplo:
  mtdparts=omap2-nand.0:512k(X-Loader)ro,1536k(UBoot)ro,512k(Environment),4m(Kernel),16m(RootFS),-(Data)
- En el comando anterior se definieron 6 particiones:
  - 1rst stage para el bootloader (512 KB solo lectura)
  - U-Boot (1536 KB solo lectura) y Environment (512 KB).
  - Kernel (4 MB)
  - Root filesystem (16 MB) y Data Filesystem (restante).

Flash fileSystems

- **IMPORTANTE**: Los tamaños de las particiones deben ser múltiplos del tamaño de bloque para borrado.

- Cuando se hace un append **ro** esa partición es solo lectura (read-only).

- El guión **-** se utiliza para marcar que se utilice todo el espacio restante.

Flash fileSystems

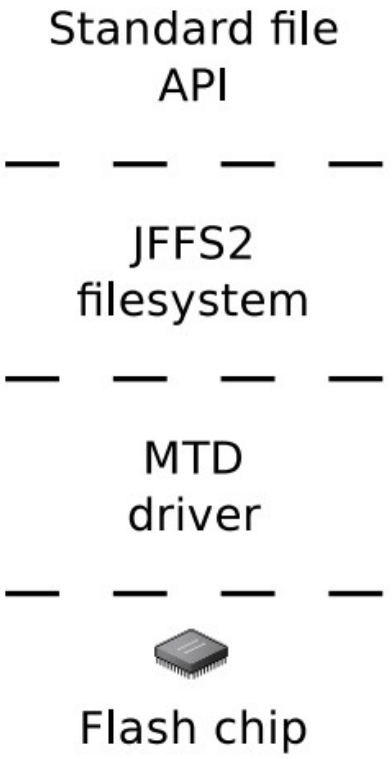
- **mtd-utils** contiene herramientas para manipular dispositivos MTD (paquete - también incluido en BusyBox*).
  - **mtdinfo** obtiene información detallada sobre un dispositivo MTD.
  - **flash_eraseall** para borrar completamente un dispositivo MTD.
  - **flashcp** para escribir sobre una flash NOR.
  - **nandwrite** para escribir sobre una flash NAND.
  - Herramientas UBI.
  - Herramientas para creacion de imagenes de flash filesystems: **mkfs.jffs2**, **mkfs.ubifs**

Flash fileSystems

- El filesystem **jffs2** es el estándar para MTD flash hoy en dia.
- Ventajas:
  - Funcionalidades como compresión on the fly ahorrando espacio y disminuyendo accesos I/O.
  - Confiable en power down.
  - Wear-level implementado (utilización de ciclos de escritura parejos en toda la flash).

Flash fileSystems

- Desventajas de **jffs2**:
  - No se escala satisfactoriamente.
  - El tiempo de montaje depende del tamaño del filesystem.
  - El kernel debe escanear todo el filesystem para determinar qué bloques pertenecen a qué archivo.
  - Es necesario utilizar CONFIG_JFFS2_SUMMARY en el kernel para que esta información de almacene en flash.
  - Esto hace que el tiempo de montaje sea drásticamente reducido.



Flash fileSystems

- Para utilizar **jffs2** en el target se necesita el paquete **mtd-utils** o una variante embebida en BusyBox.
- Borrar y formatear una partición jffs2:
  flash_eraseall -j /dev/mtd2
- Montar la partición:
  mount -t jffs2 /dev/mtdblock2 /mnt/flash
- Grabar una imagen jffs2:
  nandwrite -p /dev/mtd2 rootfs.jffs2

Flash fileSystems

- Para crear una imagen **jffs2** se utiliza **mkfs.jffs2**.
- Primero debe encontrarse el tamaño de bloque de borrado (esto es en el target):
  cat /proc/mtd.
- Luego se crea la imagen en el host:
  mkfs.jffs2 --pad --no-cleanmarkers --eraseblock=256 -d rootfs/ -o rootfs.jffs2
- Esto es teniendo en cuenta que el bloque de borrado sea de 256KB.

Flash fileSystems

- **mkfs.jffs2 --pad --no-cleanmarkers --eraseblock=256 -d rootfs/ -o rootfs.jffs2**
- La opción **--pad** justamente efectúa un padding sobre la imagen hasta el final del bloque de borrado.
- No importa que la imagen sea menor que el tamaño de la partición MTD.
- El filesystem jffs2 utiliza la partición completa de todas maneras.
- **--no-cleanmarkers** es solo para memorias NAND.

Flash fileSystems

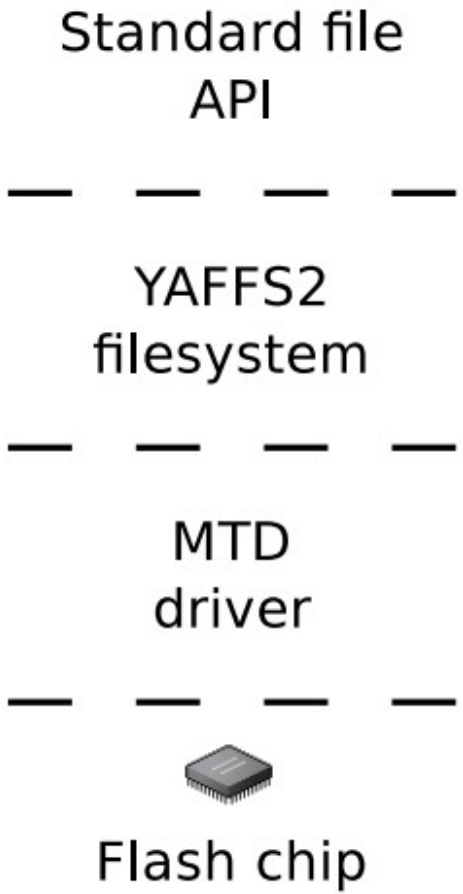
- Puede ser posible que no se desee tener el paquete mtd-utils en el target.
- Se puede crear una imagen JFFS2 en el HOST.
- En la línea de comandos de U-Boot, se descarga la imagen jffs2 a RAM a traves de tftp.
- Escribirla en flash mediante los mismos comandos antes mencionados.
- Limitación: Si la imagen es mas grande que el espacio en RAM debe hacerse en varios trozos.

Flash fileSystems

- Para bootear desde un root filesystem jffs2 se agrega a la linea de comandos del kernel: **root=/dev/mtdblock<x>, rootfstype=jffs2**
- El segundo argumento es necesario porque el kernel no puede autodetectar el tipo de filesystem dentro de **flash MTD**.
- Para particiones root en un **block device** (memoria SD, HDD), el argumento **rootfstype** puede reducir el tiempo de booteo.
- Esto es porque el kernel no debe probar bloque por

Flash fileSystems

- Otro tipo de filesystem es el **yaffs2**.
- En su mayoría soporte para NAND flash.
- No posee compresión.
- Resistente a fallas de alimentación
- Wear-level implementado.
- Tiempo de booteo pequeño.



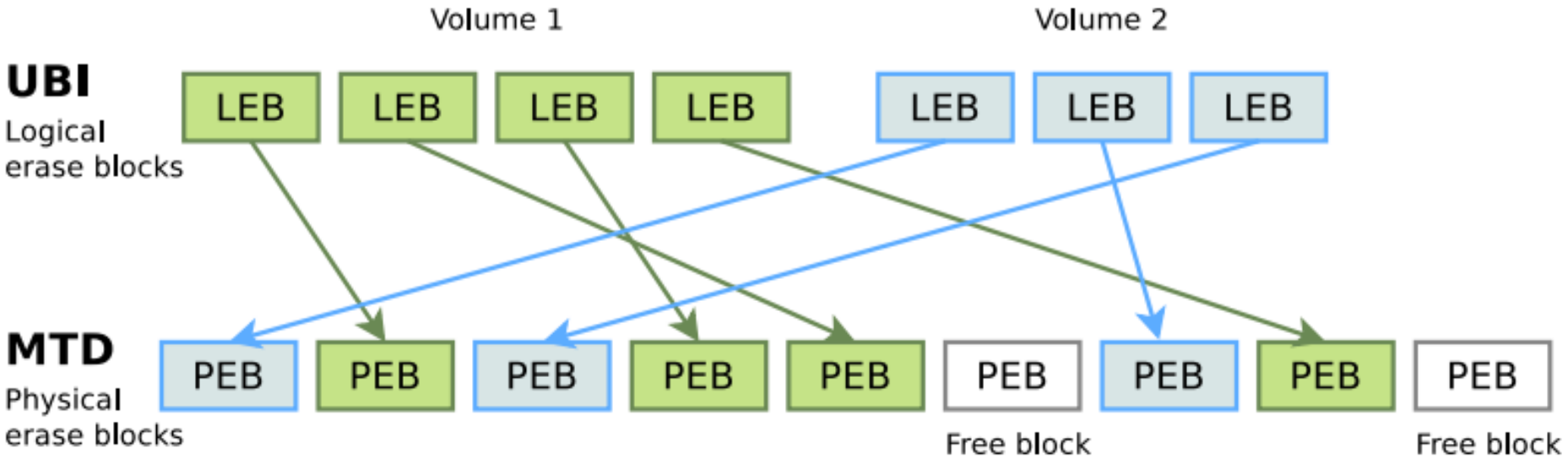
Flash fileSystems

- Existe una capa de software previa llamada UBI.
- Siglas de *Unsorted Block Images,* se utiliza como base para **ubifs**.
- Permite la creación de múltiples volúmenes lógicos y escrituras esparcidas a través de todos los bloques físicos.
- Se ocupa de administrar el borrado de bloques y hacer wear-leveling.

Flash fileSystems

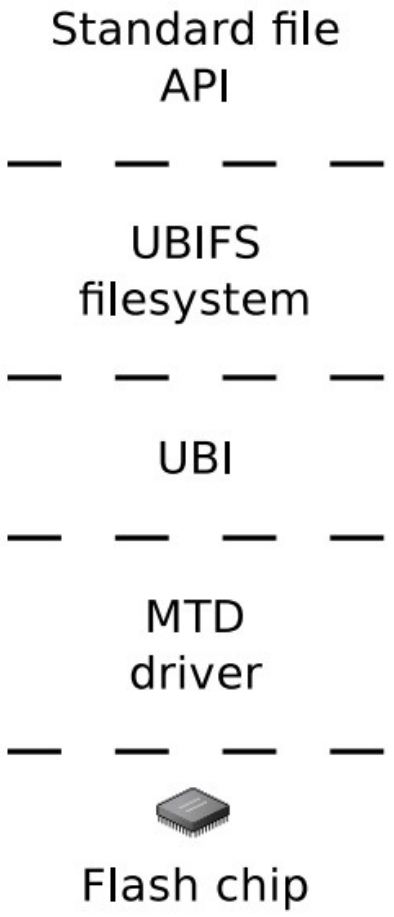
- Hace que los filesystems sean más fáciles de implementar.
- El wear-level puede trabajar sobre todo el almacenamiento, no solo sobre particiones individuales (gran ventaja).
- Los volúmenes pueden ser dinámicamente redimensionados o pueden ser configurados read-only (estáticos).

Flash fileSystems

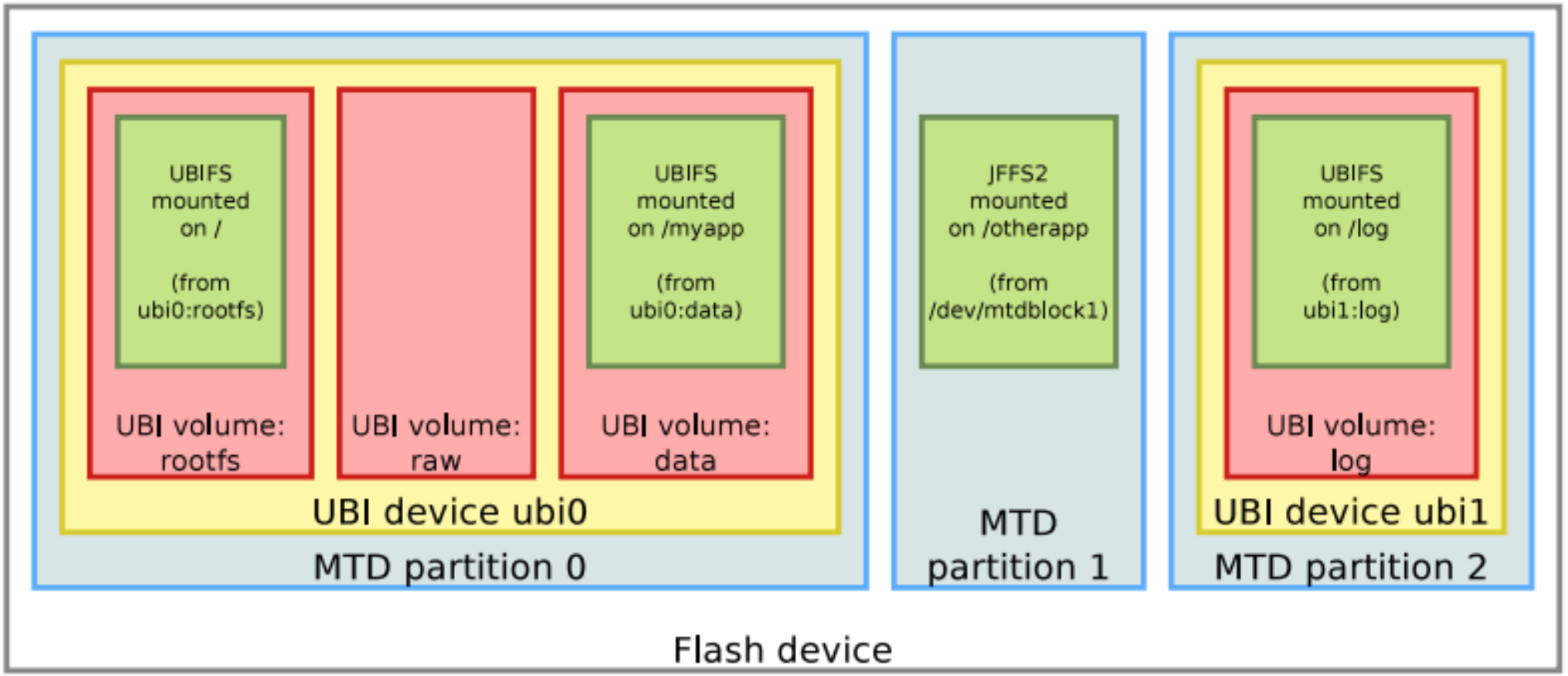


Flash fileSystems

- Sobre la capa **UBI** se implementa el filesystem **UBIFS**.
- Es la próxima generación de filesystem **jffs2** (mismos desarrolladores linux-mtd).
- Desventaja: tiene un overhead considerable de metadata sobre particiones pequeñas.



Flash fileSystems



Flash fileSystems

- Desempeño de filesystems.
- **jffs2:**
  - La peor performance.
  - Requiere CONFIG_SUMMARY para tener un tiempo de booteo aceptable.
- **yaffs2:**
  - Buena performance, pero no se encuentra en el mainline.

Flash fileSystems

- Desempeño de filesystems.
- **ubifs:**
  - La mejor solución y performance para particiones medianas y grandes.
  - Mucho overhead en metadata para particiones pequeñas.
- El unico caso donde jffs2 y yaffs2 todavia son útiles es para particiones muy pequeñas.

Flash fileSystems

- Un problema típico de almacenamiento en dispositivos block basados en flash es que se necesita una interface block.
- No hay forma de acceder a la interface flash de bajo nivel y utilizar un filesystem linux para hacer wear-leveling.
- No hay detalles sobre la capa FTL que usan (Flash Translation Layer).
- No se sabe acerca del algoritmo de wear-level.
- Por lo anterior, altamente recomendado que se limiten las escrituras en estos dispositivos.

Flash fileSystems

- No debe utilizarse almacenamiento flash como área swap.
- De todas maneras esto no es común en sistemas embebidos.
- Montar los filesystems como solo lectura, o utilizar read-only filesystems (squashFS) cuando sea posible.
- Mantener archivos volátiles en RAM (tmpfs).
- No utilizar opción sync en comando mount (escribe cambios en el momento).

Gracias.