

How to support UBIFS through MTD

Stable: 11.02.2019 - 12:21 / Revision: 17.01.2019 - 17:44

Contents

1 Purpose	1
2 Overview	1
3 Kernel configuration	1
4 Using a UBIFS partition as root file system	2
5 Mounting a UBIFS partition	2
6 Create a default file system on a MTD partition	3
7 References	5

1 Purpose

The purpose of this article is to introduce the UBIFS file system:

- General information
- Main components
- How to use UBIFS

2 Overview

UBI/UBIFS^[1] is designed to work on top of raw Flash memories (NOR/NAND). It does not work on top of block devices (MMC/SD cards).

3 subsystems are involved with UBIFS:

- The UBIFS file system, which works on top of UBI volumes. Please refer to the MTD UBIFS documentation ^[2].
- The UBI subsystem, which works on top of MTD devices. This subsystem is a wear-leveling and volume management system for raw Flash memories. Please refer to the MTD UBI documentation ^[3].
- The MTD subsystem, which provides uniform interfaces to access raw Flash memories. Please refer to the [MTD framework](#).

3 Kernel configuration

UBIFS is activated by default in ST deliveries. Nevertheless, if a specific configuration is needed, this section indicates how UBIFS can be activated/deactivated in the kernel.

Activate UBIFS in the kernel configuration with the Linux Menuconfig tool: [Menuconfig or how to configure kernel](#).

```
[*] Device Drivers --->
    <*> Memory Technology Device (MTD) support --->
        <*> Enable UBI- Unsorted block images --->
File systems --->
    [*] Miscellaneous filesystems --->
        <*> UBIFS file system support
```

4 Using a UBIFS partition as root file system

Assuming the UBIFS image is already flashed to the raw Flash memory, the user has to provide:

- The boot arguments to attach the UBI device (using `ubi.mtd=X`), where X is the MTD device which hosts the root file system volume, named "rootfs".
- The file system type (using `rootfstype=ubifs`).
- The UBI volume that has to be mounted (using `root=ubiX_T` or `root=ubiX:NAME`), where X is the UBI device number, Y is the UBI volume number and NAME is the UBI volume name.

Please refer to the [NAND memory mapping](#) to check the "rootfs" location in ST deliveries.

The following is an example of the kernel boot arguments assuming that the rootfs has been flashed on MTD partition 3 (`/dev/mtd3`) and this MTD partition has an UBI volume named "rootfs". In this case, the kernel command-line^[4] will be:

```
ubi.mtd=3 root=ubi0:rootfs rootfstype=ubifs rootwait rw console=ttySTM0,115200
```

5 Mounting a UBIFS partition

Assuming that the "userfs" volume has been flashed on MTD partition 3 (`/dev/mtd3`), below steps show how to mount this volume.

Please refer to the [NAND memory mapping](#) to check the "userfs" location in ST deliveries.

- Attach mtd3 to UBI.

```
Board $> ubiattach /dev/ubi_ctrl -m 3
ubi0: attaching mtd3
ubi0: scanning is finished
ubi0: attached mtd3 (name "UBI", size 1018 MiB)
ubi0: PEB size: 262144 bytes (256 KiB), LEB size: 253952 bytes
ubi0: min./max. I/O unit sizes: 4096/4096, sub-page size 4096
ubi0: VID header offset: 4096 (aligned 4096), data offset: 8192
ubi0: good PEBs: 4068, bad PEBs: 4, corrupted PEBs: 0
ubi0: user volume: 4, internal volumes: 1, max. volumes count: 128
ubi0: max/mean erase counter: 2/0, WL threshold: 4096, image sequence number: 980643132
ubi0: available PEBs: 0, total reserved PEBs: 4068, PEBs reserved for bad PEB handling: 7
ubi0: background thread "ubi_bgt0d" started, PID 1138
UBI device number 0, total 4068 LEBs (1033076736 bytes, 985.2 MiB), available 0 LEBs (0 b
```

- Mount "userfs" volume.

```
Board $> mount -t ubifs ubi0:userfs /media
UBIFS (ubi0:3): background thread "ubifs_bgt0_3" started, PID 648
UBIFS (ubi0:3): UBIFS: mounted UBI device 0, volume 3, name "userfs"
UBIFS (ubi0:3): LEB size: 253952 bytes (248 KiB), min./max. I/O unit sizes: 4096 bytes/4096 bytes
UBIFS (ubi0:3): FS size: 138911744 bytes (132 MiB, 547 LEBs), journal size 9404416 bytes
UBIFS (ubi0:3): reserved for root: 0 bytes (0 KiB)
UBIFS (ubi0:3): media format: w4/r0 (latest is w5/r0), UUID 6A93917C-50A2-4498-B372-6F864
```

- Check "userfs" volume is mounted.

```
Board $> mount | grep ubifs
ubi0:userfs on /media type ubifs (rw,relatime,assert=read-only,ubi=0,vol=3)
```

6 Create a default file system on a MTD partition

- Format a MTD partition (mtd3 will be used in this example).

```
Board $> ubiformat /dev/mtd3
ubiformat: mtd3 (nand), size 1067450368 bytes (1018.0 MiB), 4072 eraseblocks of 262144 bytes
libscan: scanning eraseblock 4071 -- 100 % complete
ubiformat: 4068 eraseblocks have valid erase counter, mean value is 1
ubiformat: 4 bad eraseblocks found, numbers: 4068, 4069, 4070, 4071
ubiformat: formatting eraseblock 4071 -- 100 % complete
```

- Attach mtd3 to UBI subsystem.

```
Board $> ubiattach /dev/ubi_ctrl -m 3
ubi0: attaching mtd3
ubi0: scanning is finished
ubi0: attached mtd3 (name "UBI", size 1018 MiB)
ubi0: PEB size: 262144 bytes (256 KiB), LEB size: 253952 bytes
ubi0: min./max. I/O unit sizes: 4096/4096, sub-page size 4096
ubi0: VID header offset: 4096 (aligned 4096), data offset: 8192
ubi0: good PEBs: 4068, bad PEBs: 4, corrupted PEBs: 0
ubi0: user volume: 0, internal volumes: 1, max. volumes count: 128
ubi0: max/mean erase counter: 4/2, WL threshold: 4096, image sequence number: 1744321690
ubi0: available PEBs: 3988, total reserved PEBs: 80, PEBs reserved for bad PEB handling: 0
ubi0: background thread "ubi_bgt0d" started, PID 613
UBI device number 0, total 4068 LEBs (1033076736 bytes, 985.2 MiB), available 3988 LEBs (1012760576 bytes, 965.8 MiB)
```

- Check the UBI status.

```
Board $> ubinfo --all
UBI version: 1
Count of UBI devices: 1
UBI control device major/minor: 10:57
Present UBI devices: ubi0

ubi0
Volumes count: 0
Logical eraseblock size: 253952 bytes, 248.0 KiB
Total amount of logical eraseblocks: 4068 (1033076736 bytes, 985.2 MiB)
Amount of available logical eraseblocks: 3988 (1012760576 bytes, 965.8 MiB)
```

```
Maximum count of volumes          128
Count of bad physical eraseblocks: 4
Count of reserved physical eraseblocks: 76
Current maximum erase counter value: 6
Minimum input/output unit size:    4096 bytes
Character device major/minor:      240:0
```

- Create a UBI volume named "testfs" related to our partition.

```
Board $> ubimkvol /dev/ubi0 -N testfs -m
Set volume size to 1012760576
Volume ID 0, size 3988 LEBs (1012760576 bytes, 965.8 MiB), LEB size 253952 bytes (248.0 KiB)
```

- Mount "testfs".

```
Board $> mount -t ubifs ubi0:testfs /media
UBIFS (ubi0:0): default file-system created
UBIFS (ubi0:0): background thread "ubifs_bgt0_0" started, PID 763
UBIFS (ubi0:0): UBIFS: mounted UBI device 0, volume 0, name "testfs"
UBIFS (ubi0:0): LEB size: 253952 bytes (248 KiB), min./max. I/O unit sizes: 4096 bytes/4096 bytes
UBIFS (ubi0:0): FS size: 1009205248 bytes (962 MiB, 3974 LEBs), journal size 33521664 bytes (32.5 MiB)
UBIFS (ubi0:0): reserved for root: 4952683 bytes (4836 KiB)
UBIFS (ubi0:0): media format: w5/r0 (latest is w5/r0), UUID 66AD8600-A3F4-4EF7-9821-0FE87...
```

- Check that the file system is empty.

```
Board $> ls -l /media
total 0
```

- Create a random data file.

```
Board $> dd if=/dev/urandom of=/tmp/random.hex bs=1M count=10
10+0 records in
10+0 records out
10485760 bytes (10 MB, 10 MiB) copied, 0.667978 s, 15.7 MB/s
Board $> sync
```

- Copy the random data file in /media.

```
Board $> cp /tmp/random.hex /media/
Board $> sync
```

- Un-mount /media.

```
Board $> umount /media
UBIFS (ubi0:0): un-mount UBI device 0
UBIFS (ubi0:0): background thread "ubifs_bgt0_0" stops
```

- Mount "testfs".

```
Board $> mount -t ubifs ubi0:testfs /media
UBIFS (ubi0:0): background thread "ubifs_bgt0_0" started, PID 800
UBIFS (ubi0:0): UBIFS: mounted UBI device 0, volume 0, name "testfs"
UBIFS (ubi0:0): LEB size: 253952 bytes (248 KiB), min./max. I/O unit sizes: 4096 bytes/40
UBIFS (ubi0:0): FS size: 1009205248 bytes (962 MiB, 3974 LEBs), journal size 33521664 byt
UBIFS (ubi0:0): reserved for root: 4952683 bytes (4836 KiB)
UBIFS (ubi0:0): media format: w5/r0 (latest is w5/r0), UUID 66AD8600-A3F4-4EF7-9821-0FE87
```

- Check that the random data file created is identical in /tmp and /media.

```
Board $> md5sum /tmp/random.hex /media/random.hex
a7e4a0b65560e917805944ca04fc9695 /tmp/random.hex
a7e4a0b65560e917805944ca04fc9695 /media/random.hex
```

- Un-mount /media.

```
Board $> umount /media
UBIFS (ubi0:0): un-mount UBI device 0
UBIFS (ubi0:0): background thread "ubifs_bgt0_0" stops
```

- Detach mtd3.

```
Board $> ubidetach -m 3
ubi0: detaching mtd3
ubi0: mtd3 is detached
```

7 References

Please refer to the following links for full description:

1. [↑ UBIFS](#)
2. [↑ MTD UBIFS](#)
3. [↑ MTD UBI](#)
4. [↑ The kernel's command-line parameters](#)