This document describes m[g]flash support in linux.

Contents

1. Overview

Mflash and gflash are embedded flash drive. The only difference is mflash is MCP(Multi Chip Package) device. These two device operate exactly same way. So the rest mflash repersents mflash and gflash altogether.

Internally, mflash has nand flash and other hardware logics and supports 2 different operation (ATA, IO) modes. ATA mode doesn't need any new driver and currently works well under standard IDE subsystem. Actually it's one chip SSD. IO mode is ATA-like custom mode for the host that doesn't have IDE interface.

Followings are brief descriptions about IO mode.
A. IO mode based on ATA protocol and uses some custom command. (read confirm, write confirm)
B. IO mode uses SRAM bus interface.
C. IO mode supports 4kB boot area, so host can boot from mflash.

2. Reserved area configuration

If host boot from mflash, usually needs raw area for boot loader image. All of the mflash's block device operation will be taken this value as start offset. Note that boot loader's size of reserved area and kernel configuration value must be same.

3. Example of mflash platform driver registration

Working mflash is very straight forward. Adding platform device stuff to board configuration file is all. Here is some pseudo example.

```
static struct mg_drv_data mflash_drv_data = {
        /* If you want to polling driver set to 1 */
        .use_polling = 0,
        /* device attribution */
        .dev_attr = MG_BOOT_DEV
};

static struct resource mg_mflash_rsc[] = {
        /* Base address of mflash */
        [0] = {
                .start = 0x08000000,
                .end = 0x08000000 + SZ_64K - 1,
                .flags = IORESOURCE_MEM
        },
        /* mflash interrupt pin */
        [1] = {
                .start = IRQ_GPIO(84),
                .end = IRQ_GPIO(84),
                .flags = IORESOURCE_IRQ
        },
        /* mflash reset pin */
```

```
        [2] = {
                .start = 43,
                .end = 43,
                .name = MG_RST_PIN,
                .flags = IORESOURCE_IO
        },
        /* mflash reset-out pin
         * If you use mflash as storage device (i.e. other than MG_BOOT_DEV),
         * should assign this */
        [3] = {
                .start = 51,
                .end = 51,
                .name = MG_RSTOUT_PIN,
                .flags = IORESOURCE_IO
        }
};

static struct platform_device mflash_dev = {
        .name = MG_DEV_NAME,
        .id = -1,
        .dev = {
                .platform_data = &mflash_drv_data,
        },
        .num_resources = ARRAY_SIZE(mg_mflash_rsc),
        .resource = mg_mflash_rsc
};

platform_device_register(&mflash_dev);
```