# Linux驱动\_MTD

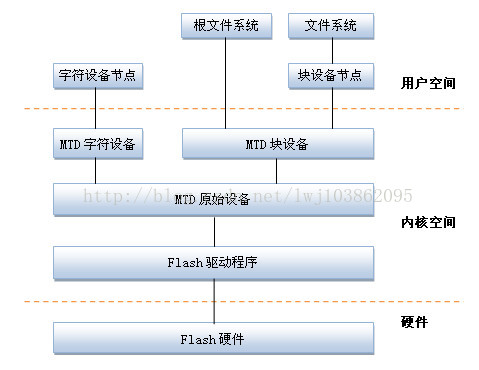
## 1. 基本知识

MTD 设备是象闪存芯片、小型闪存卡、记忆棒等之类的设备，它们在嵌入式设备中的使用正在不断增长。

MTD 驱动程序是在Linux下专门为嵌入式环境开发的新的一类驱动程序。相对于常规块设备驱动程序，使用 MTD 驱动程序的主要优点在于MTD驱动程序是专门为基于闪存的设备所设计的，所以它们通常有更好的支持、更好的管理和基于扇区的擦除和读写操作的更好的接口。

注意：SD/MMC卡、CF（Compact Flash）卡、USB Flash等并不是MTD设备，因为这些设备中已经有一个内置的Flash Translation Layer，这个layer处理erase、wear leveling事情了（这个TL应该是固件中支持的）。所以这些设备直接当做普通的Block Device使用。

MTD层为NOR FLASH和NAND FLASH设备提供统一接口。MTD将文件系统与底层FLASH存储器进行了隔离。



如上图所示，MTD设备通常可分为四层，从上到下依次是：设备节点、MTD设备层、MTD原始设备层、硬件驱动层。

**Flash硬件驱动层**：Flash硬件驱动层负责对Flash硬件的读、写和擦除操作。MTD设备的Nand Flash芯片的驱动则drivers/mtd/nand/子目录下,Nor Flash芯片驱动位于drivers/mtd/chips/子目录下。

**MTD原始设备层：**用于描述MTD原始设备的[**数据结构**](http://lib.csdn.net/base/datastructure)是mtd\_info，它定义了大量的关于MTD的数据和操作函数。其中**mtdcore.c**:  MTD原始设备接口相关实现，**mtdpart.c** :  MTD分区接口相关实现。

**MTD设备层：**基于MTD原始设备，linux系统可以定义出MTD的**块设备（主设备号31）**和**字符设备（设备号90）**。其中**mtdchar.c** :  MTD字符设备接口相关实现，**mtdblock.c** : MTD块设备接口相关实现。

**设备节点：**通过mknod在/dev子目录下建立**MTD块设备节点（主设备号为31）**和**MTD字符设备节点（主设备号为90）**。通过访问此设备节点即可访问MTD字符设备和块设备 。

## 2. nandflash驱动分析

分析驱动的套路都是差不多的，先找到nandflash对应的platform device和platform driver。

### 2.1 platform device

首先定义了nandflash的platform\_device结构体，指定了操作的nandflash的寄存器地址。

static struct resource s3c\_nand\_resource**[]** **=** **{**

**[**0**]** **=** **{**

**.**start **=** S3C2410\_PA\_NAND**,**

**.**end **=** S3C2410\_PA\_NAND **+** S3C24XX\_SZ\_NAND **-** 1**,**

**.**flags **=** IORESOURCE\_MEM**,**

**}**

**};**

struct platform\_device s3c\_device\_nand **=** **{**

**.**name **=** "s3c2410-nand"**,**

**.**id **=** **-**1**,**

**.**num\_resources **=** ARRAY\_SIZE**(**s3c\_nand\_resource**),**

**.**resource **=** s3c\_nand\_resource**,**

**};**

在s3c2440\_map\_io函数中，将s3c\_device\_nand的名称修改为s3c2440-nand。

s3c\_device\_nand**.**name **=** "s3c2440-nand"**;**

当然，仅有这些数据是完全不够的，还需要设置一些私有数据。

s3c\_device\_nand**.**dev**.**platform\_data **=** **&**smdk\_nand\_info**;**

static struct s3c2410\_platform\_nand smdk\_nand\_info **=** **{**

**.**tacls **=** 20**,**

**.**twrph0 **=** 60**,**

**.**twrph1 **=** 20**,**

**.**nr\_sets **=** ARRAY\_SIZE**(**smdk\_nand\_sets**),**

**.**sets **=** smdk\_nand\_sets**,**

**};**

其中：

TACLS：表示CLT/ALE的建立时间(setup time)。

TWRPH0：表示CLE/ALE的持续时间。

TWRPH1：表示CLE/ALE的维持时间(hold time)。

smdk\_nand\_sets为nandflash分区表。

static struct s3c2410\_nand\_set smdk\_nand\_sets**[]** **=** **{**

**[**0**]** **=** **{**

**.**name **=** "NAND"**,**

**.**nr\_chips **=** 1**,**

**.**nr\_partitions **=** ARRAY\_SIZE**(**smdk\_default\_nand\_part**),**

**.**partitions **=** smdk\_default\_nand\_part**,**

**},**

**};**

static struct mtd\_partition smdk\_default\_nand\_part**[]** **=** **{**

**[**0**]** **=** **{**

**.**name **=** "bootloader"**,**

**.**size **=** 0x00040000**,**

**.**offset **=** 0**,**

**},**

**[**1**]** **=** **{**

**.**name **=** "params"**,**

**.**offset **=** MTDPART\_OFS\_APPEND**,**

**.**size **=** 0x00020000**,**

**},**

**[**2**]** **=** **{**

**.**name **=** "kernel"**,**

**.**offset **=** MTDPART\_OFS\_APPEND**,**

**.**size **=** 0x00200000**,**

**},**

**[**3**]** **=** **{**

**.**name **=** "root"**,**

**.**offset **=** MTDPART\_OFS\_APPEND**,**

**.**size **=** MTDPART\_SIZ\_FULL**,**

**}**

**};**

最后调用platform\_device\_register注册到系统中。

### 2.2 platform driver

同样的，会注册s3c2440-nand对应的platform driver。

static struct platform\_driver s3c2440\_nand\_driver **=** **{**

**.**probe **=** s3c2440\_nand\_probe**,**

**.**remove **=** s3c2410\_nand\_remove**,**

**.**suspend **=** s3c24xx\_nand\_suspend**,**

**.**resume **=** s3c24xx\_nand\_resume**,**

**.**driver **=** **{**

**.**name **=** "s3c2440-nand"**,**

**.**owner **=** THIS\_MODULE**,**

**},**

**};**

### 2.3 s3c2440\_nand\_probe

static int s3c2440\_nand\_probe**(**struct platform\_device **\***dev**)**

**{**

**return** s3c24xx\_nand\_probe**(**dev**,** TYPE\_S3C2440**);**

**}**

static int s3c24xx\_nand\_probe**(**struct platform\_device **\***pdev**,**

enum s3c\_cpu\_type cpu\_type**)**

**{**

struct s3c2410\_platform\_nand **\***plat **=** to\_nand\_plat**(**pdev**);**

struct s3c2410\_nand\_info **\***info**;**

struct s3c2410\_nand\_mtd **\***nmtd**;**

struct s3c2410\_nand\_set **\***sets**;**

struct resource **\***res**;**

int err **=** 0**;**

int size**;**

int nr\_sets**;**

int setno**;**

//分配内存，用于存储s3c2410\_nand\_info

info **=** kmalloc**(sizeof(\***info**),** GFP\_KERNEL**);**

memzero**(**info**,** **sizeof(\***info**));**

//设置platform\_device的私有参数

//可以使用platform\_get\_drvdata得到私有参数

platform\_set\_drvdata**(**pdev**,** info**);**

//初始化自旋锁

spin\_lock\_init**(&**info**->**controller**.**lock**);**

//初始化等待队列

init\_waitqueue\_head**(&**info**->**controller**.**wq**);**

//打开nandflash时钟

info**->**clk **=** clk\_get**(&**pdev**->**dev**,** "nand"**);**

clk\_enable**(**info**->**clk**);**

//获取nandflash操作寄存器，并标记这块空间

res **=** pdev**->**resource**;**

size **=** res**->**end **-** res**->**start **+** 1**;**

info**->**area **=** request\_mem\_region**(**res**->**start**,** size**,** pdev**->**name**);**

//将device结构体赋给info

//device结构体用于描述设备相关的信息设备之间的层次关系，以及设备与总线、驱动的关系。

info**->**device **=** **&**pdev**->**dev**;**

//platform为platform device的platform\_data，也就是smdk\_nand\_info

info**->**platform **=** plat**;**

//将nandflash的寄存器地址ioremap

info**->**regs **=** ioremap**(**res**->**start**,** size**);**

//cpu\_type设置为TYPE\_S3C2440

info**->**cpu\_type **=** cpu\_type**;**

//设置TACLS/TWRPH0/TWRPH1寄存器并使能controller

err **=** s3c2410\_nand\_inithw**(**info**,** pdev**);**

//获取分区信息

sets **=** **(**plat **!=** **NULL)** **?** plat**->**sets **:** **NULL;**

nr\_sets **=** **(**plat **!=** **NULL)** **?** plat**->**nr\_sets **:** 1**;**

//分区的个数

info**->**mtd\_count **=** nr\_sets**;**

//分配分区个数\*s3c2410\_nand\_mtd结构体空间

size **=** nr\_sets **\*** **sizeof(\***info**->**mtds**);**

info**->**mtds **=** kmalloc**(**size**,** GFP\_KERNEL**);**

memzero**(**info**->**mtds**,** size**);**

//初始化每个mtd分区,并将mtd分区添加到系统中

nmtd **=** info**->**mtds**;**

**for** **(**setno **=** 0**;** setno **<** nr\_sets**;** setno**++,** nmtd**++)** **{**

s3c2410\_nand\_init\_chip**(**info**,** nmtd**,** sets**);**

nmtd**->**scan\_res **=** nand\_scan**(&**nmtd**->**mtd**,** **(**sets**)** **?** sets**->**nr\_chips **:** 1**);**

**if** **(**nmtd**->**scan\_res **==** 0**)** **{**

s3c2410\_nand\_add\_partition**(**info**,** nmtd**,** sets**);**

**}**

**if** **(**sets **!=** **NULL)**

sets**++;**

**}**

**}**

2.3.1

2.3.2

//增加新的分区 mtd\_utils 分区升级yaffs2 jffs2