

君正[®]

SPI NAND 添加参数说明文档

Date: Feb 2023



北京君正集成电路股份有限公司
Ingenic Semiconductor Co., Ltd.

Copyright © 2005-2023 Ingenic Semiconductor Co. Ltd. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Ingenic Semiconductor Co. Ltd.

Trademarks and Permissions



、Ingenic and Ingenic icons are trademarks of Ingenic Semiconductor Co.Ltd. All other trademarks and trade names mentioned in this document are the property of their respective holders.

Disclaimer

All the deliverables and data in this folder serve only as a reference for customer development. Please read through this disclaimer carefully before you use the deliverables and data in this folder. You may use the deliverables in this folder or not. However, by using the deliverables and data in this folder, you agree to accept all the content in this disclaimer unconditional and irrevocable. If you do not find the content in this disclaimer reasonable, you shall not use the deliverables and data in this folder. The deliverables and data in this folder are provided "AS IS" without representations, guarantees or warranties of any kind (either express or implied). To the maximum extent permitted by law, Ingenic Semiconductor Co., Ltd (Ingenic) provides the deliverables and data in this folder without implied representations, guarantees or warranties, including but not limited to implied representations, guarantees and warranties of merchantability, non-infringement, or fitness for a particular purpose. Deviation of the data provided in this folder may exist under different test environments.

Ingenic takes no liability or legal responsibility for any design and development error, incident, negligence, infringement, and loss (including but not limited to any direct, indirect, consequential, or incidental loss) caused by the use of data in this folder. Users shall be responsible for all risks and consequences caused by the use of data in this folder.

Ingenic Semiconductor Co., Ltd.

Add: Ingenic Headquarters, East Bldg. 14, Courtyard #10, Xibeiwang East Road, Haidian Dist., Beijing, China

Tel: **(86-10)56345000**

Fax: **(86-10)56345001**

<http://www.ingenic.cn>

目录

一、SPL 添加 device	4
二、Uboot 添加 device	5
三、Kernel 添加 device	错误! 未定义书签。

版本历史

日期	版本	描述
2023-02-01	2.0	发布第二版
2018-03-21	1.0	发布第一版

概述

支持新设备需要将 SFC NAND 参数添加到三处 SPL、UBOOT 和 KERNEL.

代码位置	参数目录
SPL	u-boot/tools/ingenic-tools/nand_device/
UBOOT	1) X1000 系列 NAND 参数代码路径: u-boot/drivers/mtd/devices/jz_sfc/nand_device/ 2) X2000 系列 NAND 参数代码路径: u-boot/drivers/mtd/devices/jz_sfc_v2/nand_device/
KERNEL	1) X1000 系列 NAND 参数代码路径: kernel/drivers/mtd/devices/ingenic_sfc_v1/nand_device/ 2) X2000 系列 NAND 参数代码路径: kernel/module_drivers/drivers/mtd/devices/ingenic_sfc_v2/nand_device/

1. SPL 添加新参数

步骤	描述
1	如果在参数目录下没有相应厂商文件时，拷贝已支持厂商.c 文件重命名为新厂商名称 例如： 1) cd cp tools/ingenic-tools/nand_device/ 2) cp foresee_nand.c new_nand.c 3) 替换 new_nand.c 文件中 FS/fs/foresee 厂商名称字段
2	参阅 FLASH 手册修改参数。

参数文件解释，以 u-boot/tools/ingenic-tools/nand_device/foresee_nand.c 为例：

```
#include <stdio.h>
```

```
#include "nand_common.h"
```

```
#define FS_MID
```

```
0xCD
```

► 解释：

厂商 ID 为 0xCD，设备 ID 为 0xA1

```
#define FS_NAND_DEVICD_COUNT 1
```

► 解释:

支持设备 ID 数量

```
static unsigned char fs_xaw[] = {0x4, 0x7};
```

► 解释:

ECC 错误状态值定义

```
static struct device_struct device[] = {
    DEVICE_STRUCT(0xA1, 2048, 2, 4, 3, 2, fs_xaw),
```

► 解释:

```
#define DEVICE_STRUCT(id, pagesize, addrlen, bit, bitcounts, eccstatcount, err) {\
    .device_id = id,           \    FLASH 设备 ID
    .page_size = pagesize,     \    FLASH 页大小
    .addr_len = addrlen,       \    单线或四线读地址长度
    .ecc_bit = bit,            \    状态寄存器中 ECC 最低位数
    .bit_counts = bitcounts,    \    状态寄存器中 ECC 所占位数
    .eccstat_count = eccstatcount, \    ECC 错误状态值数量
    .eccerrstatus = err,       \    ECC 错误状态值内存地址
}
```

```
};
```

```
static struct nand_desc fs_nand = {
    .id_manufactory = FS_MID,
    .device_counts = FS_NAND_DEVICD_COUNT,
    .device = device,
};
```

```
int foresee_nand_register_func(void) {
    return nand_register(&fs_nand);
}
```

2. UBOOT 添加新参数

步骤	描述
1	<p>如果在参数目录下没有相应厂商文件时，拷贝已支持厂商.c 文件重命名为新厂商名称，并修改参数目录下的 Makefile，添加新厂商名.o 文件。</p> <p>例如:</p> <ol style="list-style-type: none"> 1) cd u-boot/drivers/mtd/devices/jz_sfc_v2/nand_device/ 2) cp foresee_nand.c new_nand.c

	3) 替换 new_nand.c 文件中 FS/fs/foresee 厂商名称字段 4) vi ../Makefile 5) 添加 COBJS-\$(CONFIG_MTD_SFCNAND) +=nand_device/new_nand.o
2	参阅 FLASH 手册修改参数。

参数文件解释，以 u-boot/drivers/mtd/devices/jz_sfc_v2/nand_device/foresee_nand.c 为例：

```
#include <errno.h>
```

```
#include <malloc.h>
```

```
#include <linux/mtd/partitions.h>
```

```
#include "../jz_sfc_common.h"
```

```
#include "nand_common.h"
```

```
#define FS_DEVICES_NUM 1
```

► 解释：

支持设备 ID 数量

```
#define TSETUP 5
```

```
#define THOLD 5
```

```
#define TSHSL_R 20
```

```
#define TSHSL_W 20
```

```
#define TRD 180
```

```
#define TPP 400
```

```
#define TBE 3
```

```
static struct jz_sfc_nand_device *fs_nand;
```

```
static struct jz_sfc_nand_base_param fs_param[FS_DEVICES_NUM] = {
```

```
[0] = {
```

```
/*FS35ND01G*/
```

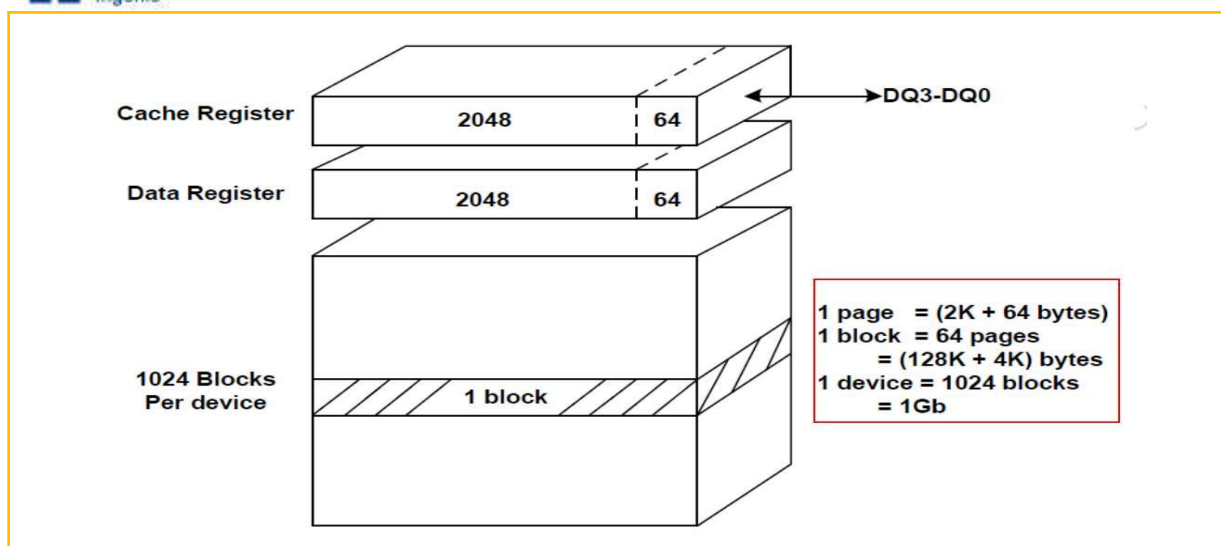
```
.pagesize = 2 * 1024,
```

```
.blocksize = 2 * 1024 * 64,
```

```
.oobsize = 64,
```

```
.flashsize = 2 * 1024 * 64 * 1024,
```

► 解释：



pagesize	页大小
blocksize	块大小
oobsize	每个页 OOB 区域大小
flashsize	FLASH 总大小

.tSETUP = TSETUP,
.tHOLD = THOLD,
.tSHSL_R = TSHSL_R,
.tSHSL_W = TSHSL_W,

► 解释:

片选时序参数:

tSLCH	CS# Active Setup Time	5		ns
tCHSH	CS# Active Hold Time	5		ns
tSHCH	CS# Not Active Setup Time	5		ns
tCHSL	CS# Not Active Hold Time	5		ns
tSHSL/tCS	CS# High Time	20		ns

tSETUP(tSLCH): CS# Active Setup Time

tHOLD (tCHSH): CS# Active Hold Time

tSHSL_R: CS #Deselect Time (for Array Read -> Array Read)

tSHSL_W: CS #Deselect Time (for Erase, Program or Read Status Registers ->Read Status Registers)

.tRD = TRD,
.tPP = TPP,
.tBE = TBE,

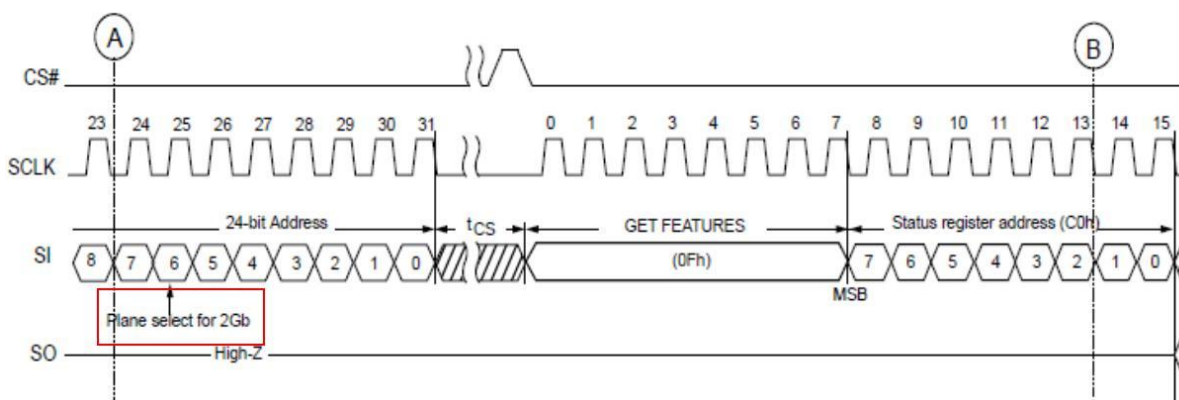
► 解释:

SYMBOL	PARAMETER	SPEC			UNIT
		MIN	TYP	MAX	
tRST	CS# High to Next Command After Reset(FFh)			500	μs
tRD	Page Read From Array		120	450	μs
tPROG	Page Program		430	800	μs
tERS	Block Erase		2	10	ms
NOP	Number of partial page program			1	time

tRD	读包含 OOB 的一页数据最大用时
tPP	写包含 OOB 的一页数据最大用时
tBE	擦除一个块数据最大用时

.plane_select = 0,

► 解释:



读写时序中包含片选信号时 plane_select=1, 否则为 0

注意: X1000 平台没有该参数, 不支持此操作

.ecc_max = 0x4,

► 解释:

ECCS2 ECCS1 ECCS0	ECC Status	<p>ECCS provides ECC Status as follows:</p> <p>000b = No bit errors were detected during the previous read algorithm.</p> <p>001b = Bit errors(=1) were detected and corrected.</p> <p>010b = Bit errors(=2) were detected and corrected.</p> <p>011b = Bit errors(=3) were detected and corrected.</p> <p>100b = Bit errors(=4) were detected and corrected.</p> <p>111b = Internal error was detected and the data not promised correctly.</p> <p>Others = Reserved.</p> <p>Bit errors might not be detected and corrected if their number exceedsthe tolerance. Therefore, block data should be refreshed when ECCStatus is equal to 100b.</p> <p>ECCS is set to 000b either following a RESET, or at the beginning ofthe READ. It is then updated after the device completes a valid READoperation.</p> <p>ECCS is invalid if internal ECC is disabled (via a SET FEATUREScommand to reset ECC_EN).</p>
-------------------------	------------	---

ECC 状态中描述最多可纠 Bits 数量

注意: 有些 FLASH 没有内部 ECC, 需要外部控制器支持, 我们不支持

.need_quad = 1,

► 解释:

激活四线功能时置 1，否则为 0

```

    },

};

static struct device_id_struct device_id[FS_DEVICES_NUM] = {
    DEVICE_ID_STRUCT(0xA1, "FS35ND01G", &fs_param[0]),

```

► 解释:

```

#define DEVICE_ID_STRUCT(id, name_string, parameter)    {\
    .id_device = id,          \    FLASH 设备 ID
    .name = name_string,      \    FLASH 设备名称
    .param = parameter,       \}   参数内存地址
};

```

```

static cdt_params_t *fs_get_cdt_params(struct sfc_flash *flash, uint8_t device_id)
{
    CDT_PARAMS_INIT(fs_nand->cdt_params);

    switch(device_id) {
        case 0xA1:
            break;

```

► 解释:

单线和四线读命令时序地址长度为 2 时 break，为 3 时地址字节数重新赋值，如下：

```

xx_nand->cdt_params.standard_r.addr_nbyte = 3;
xx_nand->cdt_params.quad_r.addr_nbyte = 3;

```

```

        default:
            pr_err("device_id err, please check your device id: device_id = 0x%02x\n",
device_id);
            return NULL;
        }

    return &fs_nand->cdt_params;
}

```

```

static inline int deal_ecc_status(struct sfc_flash *flash, uint8_t device_id, uint8_t
ecc_status)
{
    int ret = 0;

    switch(device_id) {

```

```

case 0xA1:
    switch((ret = ((ecc_status >> 4) & 0x7))) {
        case 0x0 ... 0x4:
            break;
        default:
            ret = -EBADMSG;
    }

```

► 解释:

case 0x0 ... 0x4: ECC 状态值在 0 到 4 之间时正常返回 ECC 状态值

default: 其他值时返回 -EBADMSG, 该页数据错误位数超过 4, 不可纠

```

        default:
            printf("device_id err, it maybe don't support this device, check your device id:
device_id = 0x%02x\n", device_id);
            ret = -EIO;    //notice!!!

    }
    return ret;

}

```

```

static int fs_nand_init(void) {

```

```

    fs_nand = kzalloc(sizeof(*fs_nand), GFP_KERNEL);
    if(!fs_nand) {
        pr_err("alloc fs_nand struct fail\n");
        return -ENOMEM;
    }

```

```

    fs_nand->id_manufactory = 0xCD;

```

► 解释:

MID is Manufacture ID (CDh for FSNAND), DID is Device ID (A1h for current device)

厂商 ID 为 0xCD, 设备 ID 为 0xA1

```

    fs_nand->id_device_list = device_id;
    fs_nand->id_device_count = FS_DEVICES_NUM;

```

```

    fs_nand->ops.get_cdt_params = fs_get_cdt_params;

```

► 解释:

单线和四线读时序地址长度不等于 2 时在此文件中 xx_get_cdt_params 函数中重新赋值

```

    fs_nand->ops.deal_ecc_status = deal_ecc_status;

```

► 解释:

ECC 状态处理函数地址

```
/* use private get feature interface, please define it in this document */
fs_nand->ops.get_feature = NULL;
```

```
return jz_sfc_nand_register(fs_nand);
}
```

```
SPINAND_MODULE_INIT(fs_nand_init);
```

3. KERNEL 添加新参数

步骤	描述
1	如果在参数目录下没有相应厂商文件时，拷贝刚在 UBOOT 中添加的 new_nand.c 到 KERNEL 参数目录下。 例如： 1) cd module_drivers/drivers/mtd/devices/ingenic_sfc_v2/nand_device/ 2) cp ~/SDK/u-boot/drivers/mtd/devices/jz_sfc_v2/nand_devcie/new_nand.c . 3) vi Makefile 4) 在 obj-y += dosilicon_nand.o foresee_nand.o... 后面添加 new_nand.o
2	对比其他参数文件修改 new_nand.c 例如: meld foresee_nand.c new_nand.c