表 8-5 SLC 的 ECC 奇偶校验码分配表（2048 字节）

	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0
MECCn_0	～P64	～P64’	～P32	～P32’	～P16	～P16’	～P8	～P8’
MECCn_1 MECCn_2	～P1024	～P1024’	～P512	～P512’	～P256	～P256’	～P128	～P128’
	～P4	～P4’	～P2	～P2’	～P1	～P1’	～P2048	～P2048’
MECCn_3	1	1	1	1	～P8192	～P8192’	～P4096	～P4096’

（2）4 字节 SLC 的 ECC 奇偶校验码分配表，如表 8-6 所示。

表 8-6 SLC 的 ECC 奇偶校验码分配表（4 字节）

SECCn_0	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0
SECCn_1	～P2	～P2’	～P1	～P1’	～P16	～P16’	～P8	～P8’
	1	1	1	1	1	1	～P4	～P4’

下面是 NAND FLASH 控制器 ECC 错误校验的一个操作实例，其具体代码实现如下：

```
// NAND_CheckECCError 函数：主要功能实现 ECC 错误校验。
// 输入: Controller - Nand 控制器端口数
// 输出: ECC 错误类型
NAND_eERROR NAND_CheckECCError(u32 Controller)
{
    u32 uEccError0;
    //u32 uEccError1;
    NAND_eERROR eError;
    eError = eNAND_NoError;
    if((NAND_Inform[Controller].uNandType == NAND_Normal8bit) ||
    (NAND_Inform[Controller].uNandType == NAND_Advanced8bit) )
    {
        uEccError0 = Inp32(&NAND(Controller)->rNFECCERR0);
        switch (uEccError0 & 0x03)
```

```
    {
        case 0x00 :     eError = eNAND_NoError;
                    break;
        case 0x01 :     eError = eNAND_1bitEccError;
                    break;
        case 0x02 :     eError = eNAND_MultiError;
                    break;
        case 0x03 :     eError = eNAND_EccAreaError;
                    break;
    }

    if(NAND_Inform[Controller].uPerformanceCheck == 0)
    {
        switch ((uEccError0 & 0x0C)>>2)
        {
            case 0x00 :     eError |= eNAND_NoError;
                        break;
            case 0x01 :     eError |= eNAND_Spare1bitEccError;
                        break;
            case 0x02 :     eError |= eNAND_SpareMultiError;
                        break;
            case 0x03 :     eError |= eNAND_SpareEccAreaError;
                        break;
        }
    }
}
else if(NAND_Inform[Controller].uNandType == NAND_MLC8bit)
{
    uEccError0 = Inp32(&NAND(Controller)->rNFECCERR0);
```

```
        //uEccError1 = Inp32(&NAND(Controller)->rNFECCERR1);
        switch ((uEccError0>>26) & 0x07)
        {
            case 0x00 :     eError = eNAND_NoError;
                        break;
            case 0x01 :     eError = eNAND_1bitEccError;
                        break;
            case 0x02 :     eError = eNAND_2bitEccError;
                        break;
            case 0x03 :     eError = eNAND_3bitEccError;
                        break;
            case 0x04 :     eError = eNAND_4bitEccError;
                        break;
            case 0x05 :     eError = eNAND_UncorrectableError;
                        break;
        }
    }
    return eError;
}
```

1．ECC 模块特性

ECC 的产生是通过 ECC 锁定（MainECCLock，SpareECCLock）位的控制寄存器来控制的。当 ECCLock 为低时，ECC 校验码通过 H/W ECC 模块产生。

2．SLC ECC 寄存器配置

以下各表显示 SLC ECC 从外部 NAND FLASH 存储器的备用区中读取值的配置。比较 ECC 奇偶校验码是通过 H/W 模块来产生的，从存储器读取 ECC 的格式是非常重要的。

注：MLC ECC 译码方式不同于 SLC ECC。

8 位 NAND FLASH 存储器接口，如表 8-7 所示。

表 8-7 8 位 NAND FLASH 存储器接口

寄存器	位[31:24]	位[23:16]	位[15:8]	位[7:0]
NFMECCD0	4th ECC for I/O[7:0]	3rd ECC for I/O[7:0]	2nd ECC for I/O[7:0]	1st ECC for I/O[7:0]
NFMECCD1	没有使用			

寄存器	位[31:24]	位[23:16]	位[15:8]	位[7:0]
NFSECCD	没有使用	2nd ECC 用于 I/O[7:0]	1st ECC 用于 I/O[7:0]	

3．SLC ECC 设计向导

（1）在使用 SLC ECC 软件模式时，复位 ECCType 为‘0’（SLC ECC 使能）。当 MainECCLock (NFCON[7]) 和 SpareECCLock(NFCON[6])开启（‘0’）时，ECC 模块生成的 ECC 奇偶校验码用于所有数据的读/写。在读数据或写数据之前，必须复位 ECC 的值，方法是把 InitMECC (NFCONT[5])和 InitSECC (NFCON[4])的位设为‘1’，同时把 MainECCLock (NFCONT[7])的位设为‘0’（开启）。

ECC 奇偶校验码的产生与否主要是通过 MainECCLock (NFCONT[7]) 和 SpareECCLock(NFCONT[6])位来控制的。

（2）无论是读数据还是写数据，ECC 模块都产生 ECC 奇偶校验码来记录 NFMECC0/1。

（3）在完成读或写页（不包括备用区数据）后，设置 MainECCLock 位为‘1’（锁定）。ECC 奇偶校验码是锁定的，同时 ECC 状态寄存器的值将不会被改变。

（4）要产生备用区 ECC 奇偶校验码，SpareECCLock (NFCONT[6]) 位需要清零‘0’（开启）。

（5）无论是读数据还是写数据，备用区 ECC 模块都产生 ECC 奇偶校验码来记录 NFSECC。

（6）在完成读或写备用区后，设置 SpareECCLock 位为‘1’（锁定）。ECC 奇偶校验码是锁定的，同时 ECC 状态寄存器的值将不会被改变。

（7）可以使用这些值来记录备用区或检查误码。

（8）例如,检查误码的主要数据区在页的读操作，在主数据区产生 ECC 校验码后，必须移动 ECC 奇偶校验码（存储到备用区）到 NFMECCD0 和 NFMECCD1 寄存器中。这时，NFECCERR0 和 NFECCERR1 将产生有效的错误状态值。

注：NFSECCD 是 ECC 备用区（通常情况下，用户将产生的 ECC 值从主数据区写入到备用区中，其值与

NFMECC0/1 一样），它是从主数据区产生的。

4．MLC ECC 设计向导（编码）

（1）在使用 MLC ECC 软件模式时，设置 MsgLength 为‘0’（512 字节信息长度），同时设置 ECCType 为‘0’（使能 MLC ECC）。ECC 模块生成的 ECC 奇偶校验码用于 512 字节写数据。所以，必须复位 ECC 的值，方法是在写数据之前，把 InitMECC (NFCONT[5])的位写‘1’，同时把 MainECCLock (NFCONT[7])的位清零‘0’（开启）。

ECC 奇偶校验码的产生与否主要是通过 MainECCLock (NFCONT[7])位来控制的。

（2）无论何时进行写数据，MLC ECC 模块都会在内部产生 ECC 奇偶校验码。

（3）在完成 512 字节的写数据后（不包括备用区数据），奇偶校验码将自动更新 NFMECC0, NFMECC1 寄存器。如果使用 512 字节的 NAND FLASH 存储器，可以编程序将这些值到备用区。然而，如果使用 NAND FLASH 存储器超过 512 字节页以上，这时将不能直接编程。在这种情况下，必须复制这些奇偶校验码到其他的存储器中，比如 DRAM。之后写入所有主数据，可以将写入 ECC 的值复制到备用区。奇偶校验码有自我纠正信息，包括它本身。

（4）要产生备用区 ECC 奇偶校验码，需要设置 MsgLength 为 1（24 字节信息长度），同时设置 ECCType 为‘1’（使能 MLC ECC）。ECC 模块生成的 ECC 奇偶校验码用于 24 字节写数据。所以，你必须复位 ECC 的值，方法是在写数据之前，把 InitMECC (NFCONT[5])的位写‘1’，同时把 MainECCLock (NFCONT[7])的位清零‘0’（开启）。

ECC 奇偶校验码的产生与否主要是通过 MainECCLock (NFCONT[7])位来控制的。

（5）无论何时进行写数据，MLC ECC 模块都会在内部产生 ECC 奇偶校验码。

（6）在完成 24 字节元数据或额外数据写入后，奇偶校验码将自动更新 NFMECC0, NFMECC1 寄存器。可以编程将这些奇偶校验码移到备用区。奇偶校验码有自我纠正信息，包括它本身。

5．MLC ECC 设计向导（译码）

（1）在四个 512 字节主区域中，备用区是由四个 7 字节的奇偶区域，24 字节的元数据以及 7 字节用于这个元数据的奇偶区组成的。

（2）在使用 MLC ECC 软件模式时，设置 MsgLength 为 0（512 字节信息长度），同时设置 ECCType 为‘1’（使能 MLC ECC）。ECC 模块生成的 ECC 奇偶校验码用于 512 字节读数据。所以，你必须复位 ECC 的值，方法是在读数据之前，把 InitMECC (NFCONT[5])的位写‘1’，同时把 MainECCLock (NFCONT[7])的位

清零‘0’（开启）。

ECC 奇偶校验码的产生与否主要是通过 MainECCLock (NFCONT[7]) 和 SpareECCLock(NFCONT[6])位来控制的。

（3）无论何时进行写数据，MLC ECC 模块都会在内部产生 ECC 奇偶校验码。

（4）在完成 512 字节读数据（不包括备用区数据）后，需要读奇偶校验码。MLC ECC 模块需要奇偶校验码检测是否有错误位。所以在读 512 字节后，读 ECC 奇偶校验码是有必要的。一旦 ECC 奇偶校验码被读，MLC ECC 引擎开始在内部搜索任何错误。MLC ECC 的错误搜索引擎可以在最小为 155 周期内找到任何错误。在这段时间内，可以继续从外部 NAND FLASH 存储器中读主数据。ECCDecDone(NFSTAT[6])可以用于检测 ECC 译码是否被完成。

（5）当 ECCDecDone (NFSTAT[6])设置为‘1’时，NFECCERR0 显示是否有错误位存在。如果有任何一种错误存在的话，则可以通过参照 NFECCERR0/1 和 NFMLCBITPT 寄存器来进行修正。

（6）如果有很多主数据要读，继续第 2 步。

（7）元数据错误检测，设置 MsgLength 为 1（24 字节信息长度），同时设置 ECCType 为‘1’（使能 MLC ECC）。ECC 模块生成的 ECC 奇偶校验码用于 24 字节读数据。所以，必须复位 ECC 的值，方法是在读数据之前，把 InitMECC (NFCONT[5])的位写‘1’，同时把 MainECCLock (NFCONT[7])的位清零‘0’（开启）。

ECC 奇偶校验码的产生与否主要是通过 MainECCLock (NFCONT[7])位来控制的。

（8）无论何时进行写数据，MLC ECC 模块都会在内部产生 ECC 奇偶校验码。

（9）在完成 24 字节读数据后，需要读奇偶校验码。MLC ECC 模块需要奇偶校验码检测是否有错误位。所以在读 24 字节后，读 ECC 奇偶校验码是有必要的。一旦 ECC 奇偶校验码被读，MLC ECC 引擎开始在内部搜索任何错误。MLC ECC 的错误搜索引擎可以在最小为 155 周期内找到任何错误。在这段时间内，可以继续从外部 NAND FLASH 存储器中读主数据。ECCDecDone(NFSTAT[6])可以用于检测 ECC 译码是否被完成。

（10）当 ECCDecDone (NFSTAT[6])设置为‘1’时，NFECCERR0 显示是否有错误位存在。如果有任何一种错误存在的话，则可以通过参照 NFECCERR0/1 和 NFMLCBITPT 寄存器来进行修正。

6．NAND FLASH 存储器映射

关于 NAND FLASH 存储器的映射，可以参看图 8-5 所示。

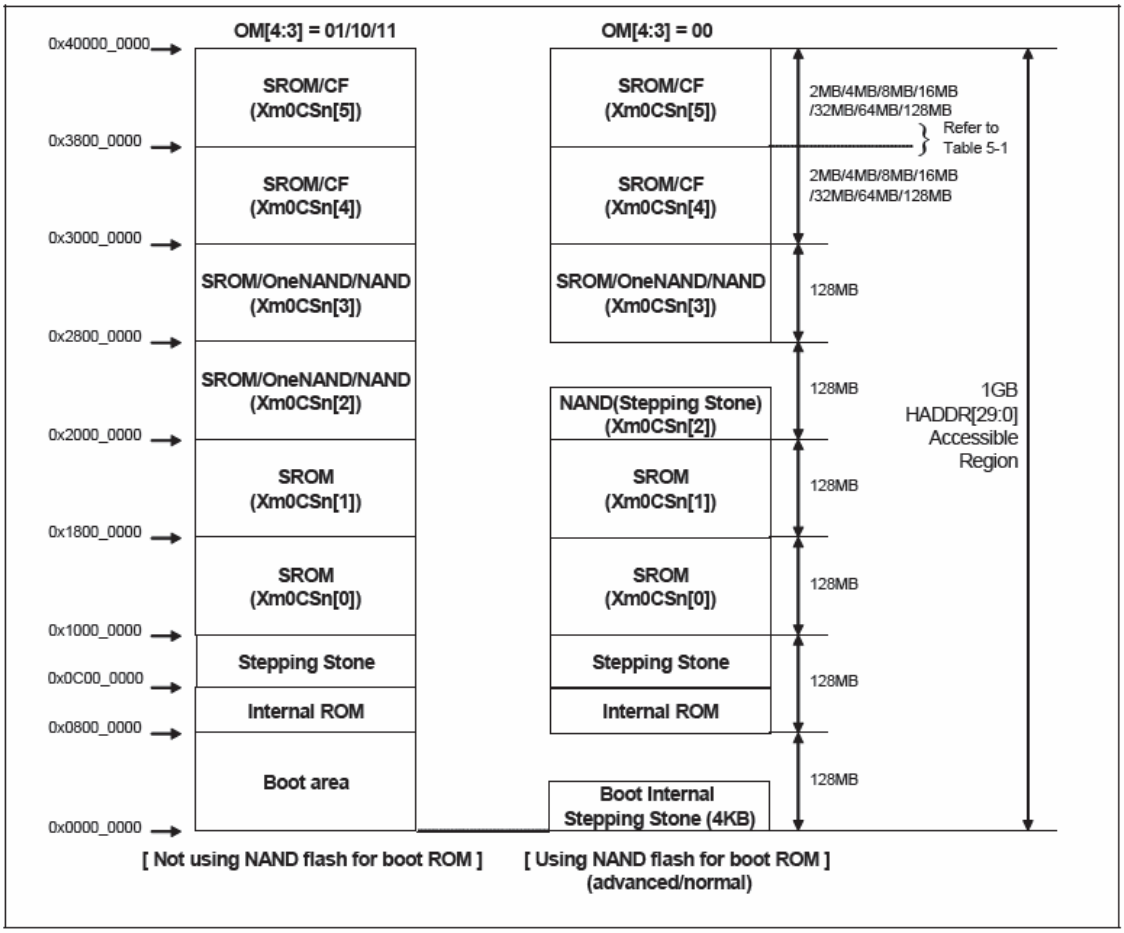

图 8-5　NAND FLASH 存储器映射结构图

7．NAND FLASH 存储器结构

以下是 NAND FLASH 存储器的结构图，如图 8-6 所示。

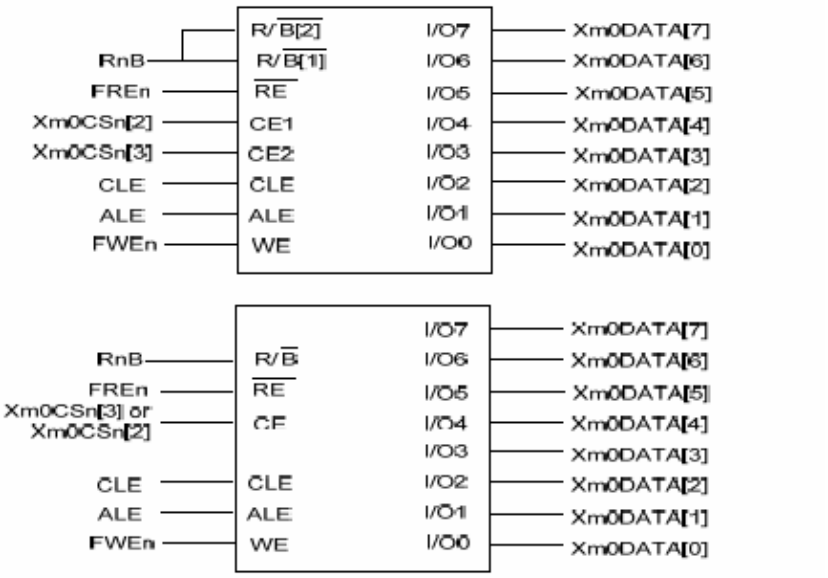

图 8-6　一个 8 位 NAND FLASH 存储器接口框图

注：NAND 控制器可以支持控制两个 NAND FLASH 存储器。

如表 8-8 所示，显示了 NAND FLASH 的两个控制器。

表 8-8 两个 NAND FLASH 控制器

	NAND	BOOT
Xm0CSn[2]	NAND 控制器 CS0	可配置
Xm0CSn[3]	NAND 控制器 CS1	可配置

如果想要从 NAND 启动，则必须使用 Xm0CSn[2]进行导入。

8.3 NAND FLASH 特殊控制寄存器

下面主要针对 NAND FLASH 控制器中的寄存器做详细的介绍，能够让读者对其中的每个寄存器的功能有所了解。

8.3.1 NAND FLASH 控制寄存器列表

NAND FLASH 控制寄存器中的不同类别寄存器定义，如表 8-9 所示。

表 8-9 NAND FLASH 控制寄存器列表

地址	读/写	复位值	名称	描述
Base + 0x00	读/写	0x0000100X	NFCONF	配置寄存器
Base + 0x04	读/写	0x000100C6	NFCONT	控制寄存器
Base + 0x08	读/写	0x00	NFCMMD	命令寄存器
Base + 0x0c	读/写	0x0000XX00	NFADDR	地址寄存器
Base + 0x10	读/写	0xXXXX	NFDATA	数据寄存器
Base + 0x14	读/写	0x00000000	NFMECCD0	第一个和第二个主 ECC 数据寄存器
Base + 0x18	读/写	0x00000000	NFMECCD1	第三个和第四个主 ECC 数据寄存器
Base + 0x1c	读/写	0x00000000	NFSECCD	备用 ECC 读寄存器
Base + 0x20	读/写	0x000000	NFSBLK	可编程开始块地址寄存器
Base + 0x24	读/写	0x000000	NFEBLK	可编程结束块地址寄存器
Base + 0x28	读/写	0x0080001D	NFSTAT	NAND 状态寄存器
Base + 0x2C	读	0x007FFFFA	NFECCERR0	ECC 错误状态 0 寄存器
Base + 0x30	读	0x007FFFFA	NFECCERR1	ECC 错误状态 1 寄存器
Base + 0x34	读	0xXXXXXX	NFMECC0	生成 ECC 状态 0 寄存器
Base + 0x38	读	0xXXXXXX	NFMECC1	生成 ECC 状态 1 寄存器
Base + 0x3C	读	0xXXXXXX	NFSECC	生成备用区 ECC 状态寄存器
Base + 0x40	读	0x00000000	NFMLCBITPT	4 位 ECC 错误位模式寄存器
* Base = 0x7020_0000				

8.3.2. NAND FLASH 配置寄存器

寄存器	地址	读/写	描述	复位值
NFCONF	0x70200000	读/写	NAND FLASH 配置寄存器	0x0000100X

NFCONF	位	描述	初始状态
NANDBoot	[31]	只读。显示是否使用 NAND 启动 1=NAND Flash 存储器启动	0
ECCClkCon	[30]	时钟控制 4 位 ECC 引擎 0：当系统时钟大于 66MHz 时，推荐使用 1：当系统时钟小于 66MHz 时，推荐使用	0
Reserved	[29:26]	保留	0000
MsgLength	[25]	信息（数据）长度为 4 位的 ECC（用于 MLC NAND） 0：512 字节为主数据区　1：24 字节为元数据	0
ECCType	[24]	ECC 类型选择： 0：SLC（1 位修正）ECC　1：MLC（4 位修正）ECC	0
Reserved	[15]	保留	0
TACLS	[14:12]	CLE & ALE 持续时间设置值 (0～7) 持续时间 = HCLK × TACLS	001
Reserved	[11]	保留	0
TWRPH0	[10:8]	TWRPH0 持续时间设置值(0～7) 持续时间= HCLK × (TWRPH0 + 1)	000
Reserved	[7]	保留	0
TWRPH1	[6:4]	TWRPH1 持续时间设置值(0～7) 持续时间= HCLK × (TWRPH1 + 1)	000
AdvFlash	[3]	预先 NAND Flash 存储器用于自启动 0：支持 512 字节/页 NAND Flash 存储器 1：支持 2048 字节/页 NAND Flash 存储器	H/W Set