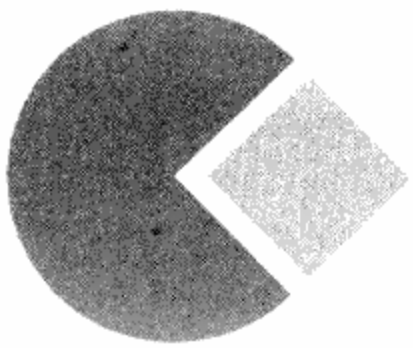


第 8 章　NAND Flash 控制器

本章目标

- 了解 NAND Flash 芯片的接口
- 掌握通过 NAND Flash 控制器访问 NAND Flash 的方法

8.1　NAND Flash 介绍和 NAND Flash 控制器使用

NAND Flash 在嵌入式系统中的地位与 PC 上的硬盘类似，用于保存系统运行所必需的操作系统、应用程序、用户数据、运行过程中产生的各类数据。与内存掉电后数据丢失不同，NAND Flash 中的数据在掉电后仍可永久保存。

8.1.1　Flash 介绍

常用的 Flash 类型有 NOR Flash 和 NAND Flash 两种。NOR Flash 由 Intel 公司在 1988 年发明，以替代当时在市场上占据主要地位的 EPROM 和 E^2PROM。NAND Flash 由 Toshiba 公司在 1989 年发明。两者的主要差别如表 8.1 所示。

表 8.1　NOR/NAND Flash 的差别

		NOR	NAND
容量		1MB～32MB	16MB～512MB
XIP		Yes	No
性能	擦除	非常慢（5s）	快（3ms）
	写	慢	快
	读	快	快
可靠性		比较高，位反转的比例小于 NAND Flash 的 10%	比较低，位反转比较常见，必需有校验措施，比如TNR必须有坏块管理措施
可擦除次数		10000～100000	100000～1000000
生命周期		低于 NAND Flash 的 10%	是 NOR Flash 的 10 倍以上

续表

	NOR	NAND
接口	与 RAM 接口相同	I/O 接口
访问方法	随机访问	顺序访问
易用性	容易	复杂
主要用途	常用于保存代码和关键数据	用于保存数据
价格	高	低

NOR Flash 支持 XIP，即代码可以直接在 NOR Flash 上执行，无需复制到内存中。这是由于 NOR Flash 的接口与 RAM 完全相同，可以随机访问任意地址的数据。在 NOR Flash 上进行读操作的效率非常高，但是擦除和写操作的效率很低；另外，NOR Flash 的容量一般比较小。NAND Flash 进行擦除和写操作的效率更高，并且容量更大。一般而言，NOR Flash 用于存储程序，NAND Flash 用于存储数据。基于 NAND Flash 的设备通常也要搭配 NOR Flash 以存储程序。

Flash 存储器件由擦除单元（也称为块）组成，当要写某个块时，需要确保这个块已经被擦除。NOR Flash 的块大小范围为 64kB～128kB；NAND Flash 的块大小范围为 8kB～64kB，擦/写一个 NOR Flash 块需 4s，而擦/写一个 NAND Flash 块仅需 2ms。NOR Flash 的块太大，不仅增加了擦写时间，对于给定的写操作，NOR Flash 也需要更多的擦除操作——特别是小文件，比如一个文件只有 1kB，但是为了保存它却需要擦除大小为 64kB～128kB 的 NOR Flash 块。

NOR Flash 的接口与 RAM 完全相同，可以随意访问任意地址的数据。而 NAND Flash 的接口仅仅包含几个 I/O 引脚，需要串行地访问。NAND Flash 一般以 512 字节为单位进行读写。这使得 NOR Flash 适合于运行程序，而 NAND Flash 更适合于存储数据。

容量相同的情况下，NAND Flash 的体积更小，对于空间有严格要求的系统，NAND Flash 可以节省更多空间。市场上 NOR Flash 的容量通常为 1MB～4MB（也有 32MB 的 NOR Flash），NAND Flash 的容量为 8MB～512MB。容量的差别也使得 NOR Flash 多用于存储程序，NAND Flash 多用于存储数据。

对于 Flash 存储器件的可靠性需要考虑 3 点：位反转、坏块和可擦除次数。所有 Flash 器件都遭遇位反转的问题：由于 Flash 固有的电器特性，在读写数据过程中，偶然会产生一位或几位数据错误（这种概率很低），而 NAND Flash 出现的概率远大于 NOR Flash。当位反转发生在关键的代码、数据上时，有可能导致系统崩溃。当仅仅是报告位反转，重新读取即可；如果确实发生了位反转，则必须有相应的错误检测/恢复措施。在 NAND Flash 上发生位反转的概率更高，推荐使用 EDC/ECC 进行错误检测和恢复。NAND Flash 上面会有坏块随机分布，在使用前需要将坏块扫描出来，确保不再使用它们，否则会使产品含有严重的故障。NAND Flash 每块的可擦除次数通常在 100000 次左右，是 NOR Flash 的 10 倍。另外，因为 NAND Flash 的块大小通常是 NOR Flash 的 1/8，所以 NAND Flash 的寿命远远超过 NOR Flash。

嵌入式 Linux 对 NOR、NAND Flash 的软件支持都很成熟。在 NOR Flash 上常用 jffs2 文件系统，而在 NAND Flash 上常用 yaffs 文件系统。在更底层，有 MTD 驱动程序实现对它们的读、写、擦除操作，它也实现了 EDC/ECC 校验。

8.1.2 NAND Flash 的物理结构

以 NAND Flash K9F1208U0M 为例，K9F1208U0M 是三星公司生产的容量为 64MB 的 NAND Flash，常用于手持设备等消费电子产品。它的封装及外部引脚如图 8.1 所示。

引脚名称	引脚功能
$I/O_0 \sim I/O_7$	数据输入/输出
CLE	命令锁存使能
ALE	地址锁存使能
$\overline{CE}$	芯片使能
$\overline{RE}$	读使能
$\overline{WE}$	写使能
$\overline{WP}$	写保护
$R/\overline{B}$	就绪/忙输出信号
Vcc	电源
Vss	地
N.C	不接

图 8.1 NAND Flash 封装及引脚

K9F1208U0M 的功能结构图如图 8.2 所示。

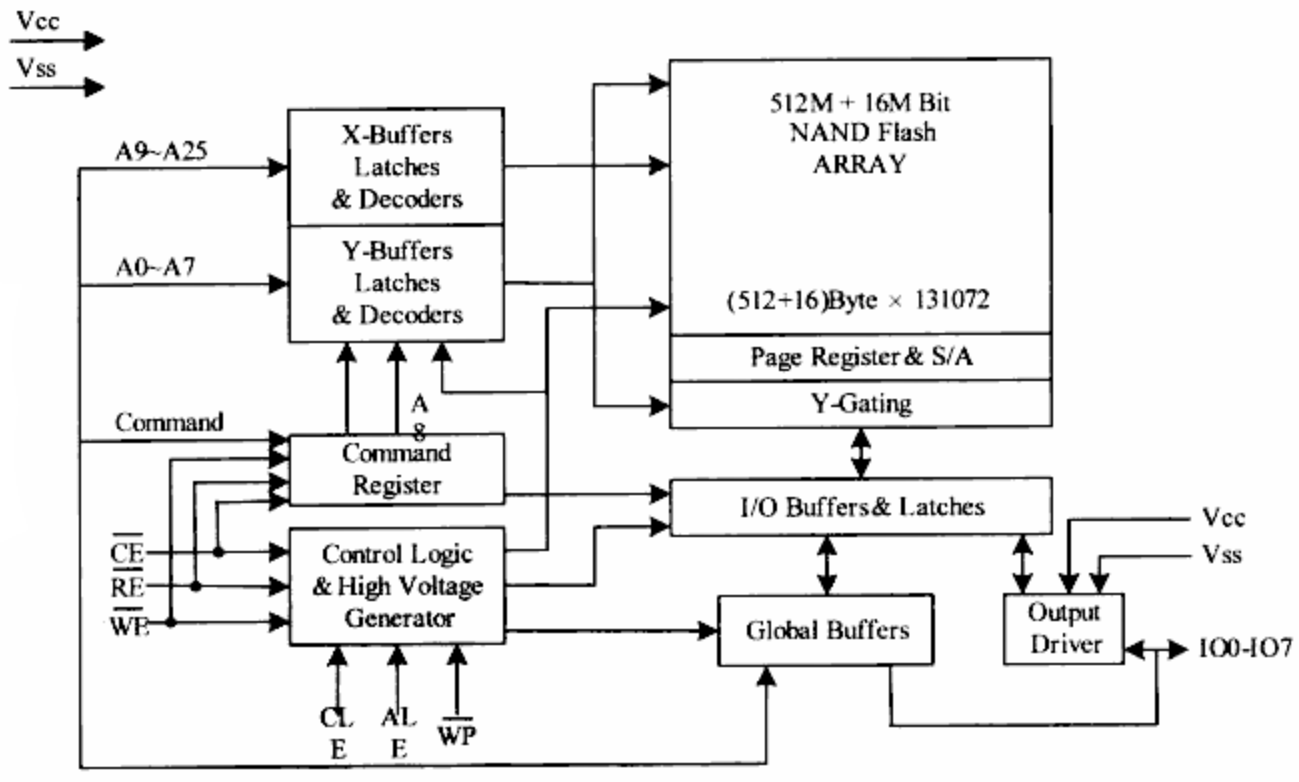


图 8.2 NAND Flash K9F1208U0M 功能结构

K9F1208U0M 的内部结构分为 10 个功能部件。

① X-Buffers Latche & Decoders：用于行地址。

② Y-Buffers Latche & Decoders：用于列地址。

③ Command Register：用于命令字。

④ Control Logic & High Voltage Generator：控制逻辑及产生 Flash 所需高压。

⑤ Nand Flash Array：存储部件。

⑥ Page Register & S/A：页寄存器，当读、写某页时，会将数据先读入/写入此寄存器，

大小为 528 字节。

⑦ Y-Gating。

⑧ I/O Buffers & Latches。

⑨ Global Buffers。

⑩ Output Driver。

NAND Flash 存储单元组织结构如图 8.3 所示。

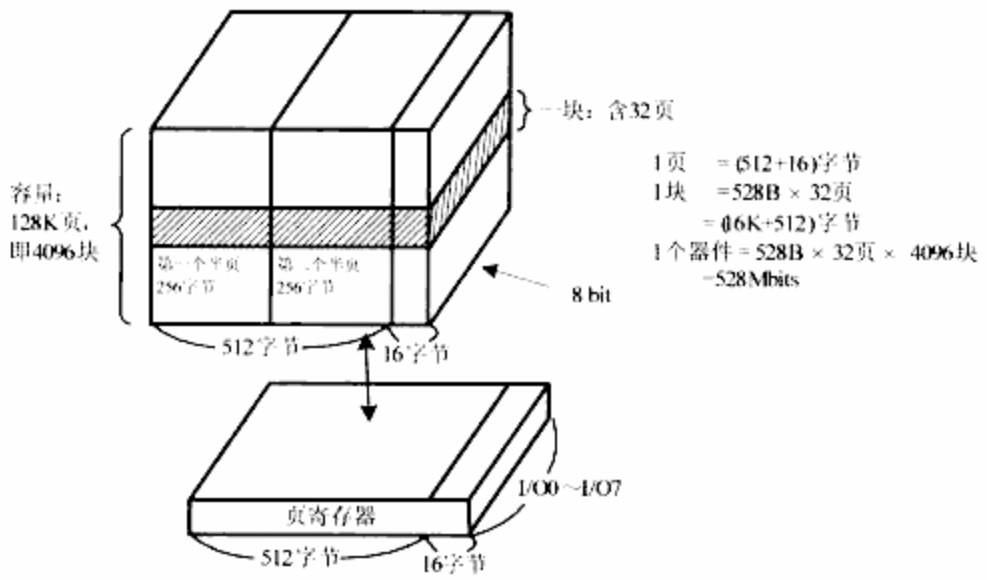


图 8.3　NAND Flash K9F1208U0M 存储单元组织结构

K9F1208U0M 容量为 528Mbit，分为 131072 行（页）、528 列；每一页大小为 512 字节，外加 16 字节的额外空间，这 16 字节额外空间的列地址为 512～527。

命令、地址、数据都通过 8 个 I/O 口输入/输出，这种形式减少了芯片的引脚个数，并使得系统很容易升级到更大的容量。写入命令、地址或数据时，都需要将 WE#、CE#信号同时拉低。数据在 WE#信号的上升沿被 NAND Flash 锁存；命令锁存信号 CLE、地址锁存信号 ALE 用来分辨、锁存命令或地址。K9F1208U0M 的 64MB 存储空间需要 26 位地址，因此以字节为单位访问 Flash 时需要 4 个地址序列：列地址、行地址的低位部分、行地址的高位部分。读/写页在发出命令后，需要 4 个地址序列，而擦除块在发出擦除命令后仅需要 3 个地址序列。

8.1.3　NAND Flash 访问方法

1．硬件连接

NAND Flash 与 S3C2410/S3C2440 的硬件连接如图 8.4 所示。

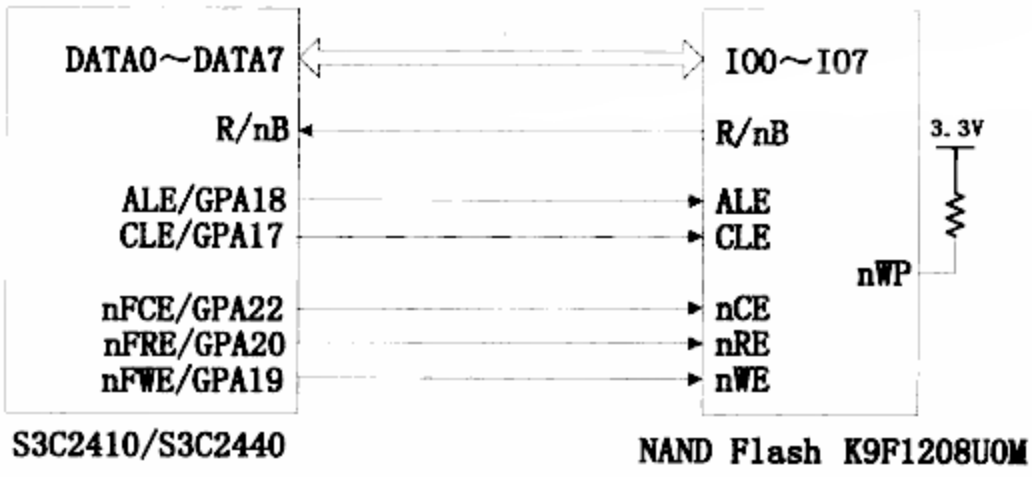


图 8.4　NAND Flash 与 S3C2410/S3C2440 的硬件连线

NAND Flash 与 S3C2410/SC32440 的连线比较少：8 个 I/O 引脚（IO0～IO7）、5 个使能信号（nWE、ALE、CLE、nCE、nRE）、1 个状态引脚（RDY/B）、1 个写保护引脚（nWP）。地址、数据和命令都是在这些使能信号的配合下，通过 8 个 I/O 引脚传输。写地址、数据、命令时，nCE、nWE 信号必须为低电平，它们在 nWE 信号的上升沿被锁存。命令锁存使能信号 CLE 和地址锁存信号 ALE 用来区分 I/O 引脚上传输的是命令还是地址。

2. 命令字及操作方法

操作 NAND Flash 时，先传输命令，然后传输地址，最后读/写数据，期间要检查 Flash 的状态。对于 K9F1208U0M，它的容量为 64MB，需要一个 26 位的地址。发出命令后，后面要紧跟着 4 个地址序列。比如读 Flash 时，发出读命令和 4 个地址序列后，后续的读操作就可以得到这个地址及其后续地址的数据。相应的命令字和地址序列如表 8.2、8.3 所示。

表 8.2　　NAND Flash K9F1208U0M 的命令字

命　令	第 1 个访问周期	第 2 个访问周期	第 3 个访问周期
Read 1（读）	00h/01h	-	-
Read 2（读）	50h	-	-
Read ID（读芯片 ID）	90h	-	-
Page Program（写页）	80h	10h	-
Block Erase（擦除块）	60h	D0h	-
Read Status（读状态）	70h	-	-
Read Multi-Plane Status（读多层的状态）	71h	-	-
Reset（复位）	FFh	-	-
Page Program（Dummy）	80h	11h	-
Copy-Back Program（True）	00h	8Ah	10h
Copy-Back Program（Dummy）	03h	8Ah	11h
Multi-Plane Block Erase	60h-60h	D0h	-

表 8.3　　NAND Flash K9F1208U0M 的地址序列

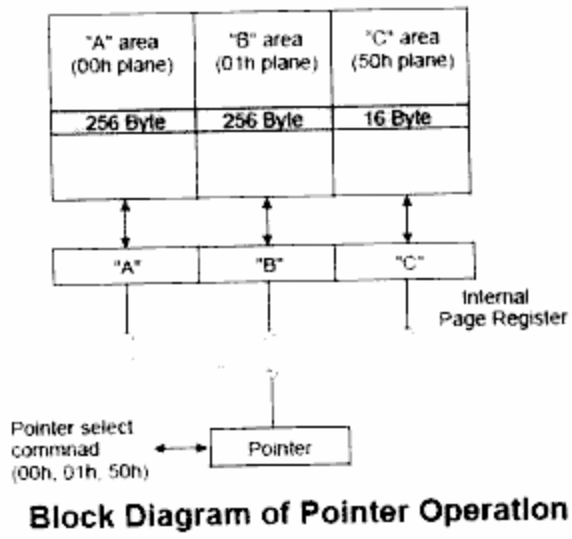
	I/O 0	I/O 1	I/O 2	I/O 3	I/O 4	I/O 5	I/O 6	I/O 7	备　注
第 1 个地址序列	A0	A1	A2	A3	A4	A5	A6	A7	列地址
第 2 个地址序列	A9	A10	A11	A12	A13	A14	A15	A16	行地址 即：页地址
第 3 个地址序列	A17	A18	A19	A20	A21	A22	A23	A24	
第 4 个地址序列	A25	L	L	L	L	L	L	L	

注：① K9F1208U0M 一页大小为 512 字节，分为两部分：上半部、下半部。
② 列地址用来在半页（256 字节）中寻址。
③ 当发出读命令 00h 时，表示列地址将在上半部寻址。
当发出读命令 01h 时，表示列地址将在下半部寻址。
④ A8 被读命令 00h 设为低电平，被 01h 设为高电平。
⑤ L 表示低电平。

K9F1208U0M 一页大小为 528 字节，而列地址 A0～A7 可以寻址的范围是 256 字节，所以必须辅以其他手段才能完全寻址这 528 字节。将一页分为 A、B、C 三个区：A 区为 0～255 字节，B 区为 256～511 字节，C 区为 512～527 字节。访问某页时，需要选定特定的区，这称为“使地址指针指向特定的区”。这通过 3 个命令来实现：命令 00h 让地址指针指向 A 区、命令 01h 让地址指针指向 B 区、命令 50h 让地址指针指向 C 区。命令 00h 和 50h 会使得访问 Flash 的地址指针一直从 A 区或 C 区开始，除非发出了其他的修改地址指针的命令。命令 01h 的效果只能维持一次，当前的读、写、擦除、复位或者上电操作完成后，地址指针重新指向 A 区。写 A 区或 C 区的数据时，必须在发出命令 80h 之前发出命令 00h 或者 50h；写 B 区的数据时，发出命令 01h 后必须紧接着就发出命令 80h。图 8.5 形象地表示了 K9F1208U0M 的这个特性。

Destination of the pointer

Command	Pointer position	Area
00h	0 ~ 255 byte	1st half array(A)
01h	256 ~ 511 byte	2nd half array(B)
50h	512 ~ 527 byte	spare array(C)



Block Diagram of Pointer Operation

图 8.5　SAMSUNG NAND Flash 地址指针操作

下面逐个讲解表 8.2 的命令字。

（1）Read 1：命令字为 00h 或 01h。

如图 8.5 所示，发出命令 00h 或 01h 后，就选定了读操作是从 A 区还是 B 区开始。从表 8.3 可知，列地址 A0～A7 可以寻址的范围是 256 字节，命令 00h 和 01h 使得可以在 512 字节大小的页内任意寻址——这相当于 A8 被命令 00h 设为 0，而被命令 01h 设为 1。

发出命令字后，依据表 8.3 发出 4 个地址序列，然后就可以检测 R/nB 引脚以确定 Flash 是否准备好。如果准备好了，就可以发起读操作依次读入数据。

（2）Read 2：命令字为 50h。

与 Read 1 命令类似，不过读取的是 C 区数据，操作序列为：发出命令字 50h、发出 4 个地址序列、等待 R/nB 引脚为高，最后读取数据。不同的是，地址序列中 A0～A3 用于设定 C 区（大小为 16 字节）要读取的起始地址，A4～A7 被忽略。

（3）Read ID：命令字为 90h。

发出命令字 90h，发出 4 个地址序列（都设为 0），然后就可以连续读入 5 个数据，分别表示：厂商代码（对于 SAMSUNG 公司为 Ech）、设备代码（对于 K9F1208U0M 为 76h）、保留的字节（对于 K9F1208U0M 为 A5h）、多层操作代码（C0h 表示支持多层操作）。

（4）Reset：命令字为 FFh。

发出命令字 FFh 即可复位 NAND Flash 芯片。如果芯片正处于读、写、擦除状态，复位命令会终止这些命令。

（5）Page Program(True)：命令字分两阶段，80h 和 10h。

它的操作序列如图 8.6 所示。

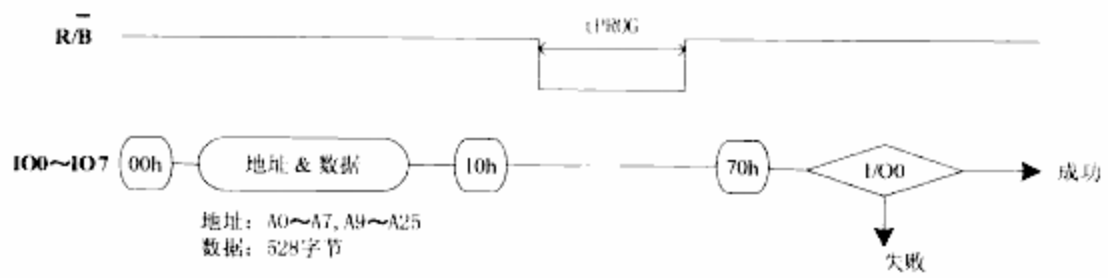


图 8.6 NAND Flash 写操作

NAND Flash 的写操作一般是以页为单位的，但是可以只写一页中的一部分。发出命令字 80h 后，紧接着是 4 个地址序列，然后向 Flash 发送数据（最大可以达到 528 字节），然后发出命令字 10h 启动写操作，此时 Flash 内部会自动完成写、校验操作。一旦发出命令字 10h 后，就可以通过读状态命令 70h 获知当前写操作是否完成、是否成功。

（6）Page Program(Dummy)：命令字分两阶段，80h 和 11h。

NAND Flash K9F1208U0M 分为 4 个 128Mbit 的存储层（plane），每个存储层包含 1024 个 block 和 528 字节的寄存器。这使得可以同时写多个页(page)或者同时擦除多个块(block)。块的地址经过精心安排，可以在 4 个连续的块内同时进行写或者擦除操作。

如图 8.7 所示为 K9F1208U0M 的块组织图。

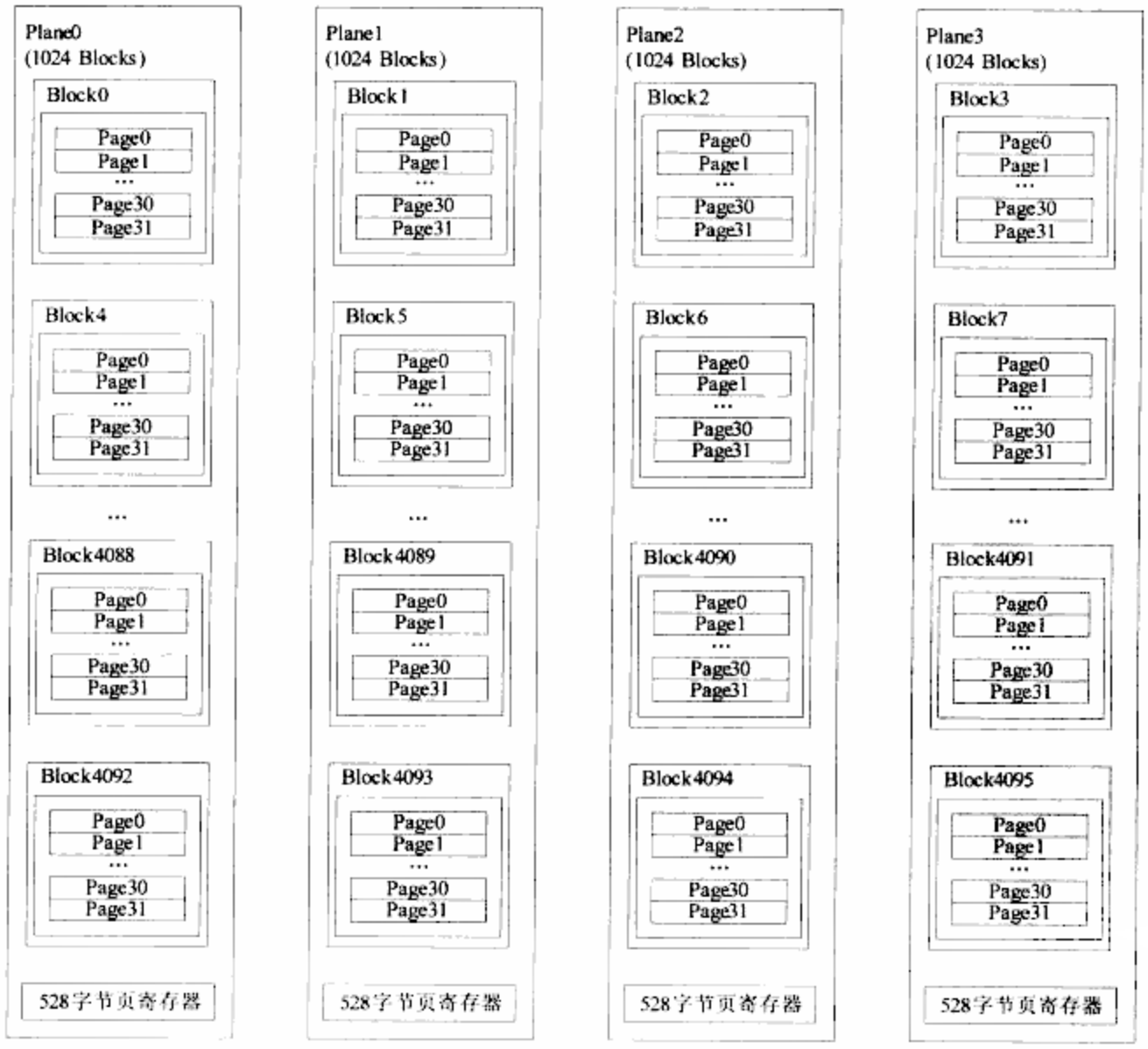


图 8.7 K9F1208U0M 的 block 组织结构

命令 Page Program(Dummy)正是在这种结构下对命令 Page Program(True)的扩展，后者仅能对一页进行写操作，前者可以同时写 4 页。命令 Page Program(Dummy)的操作序列如图 8.8 所示。

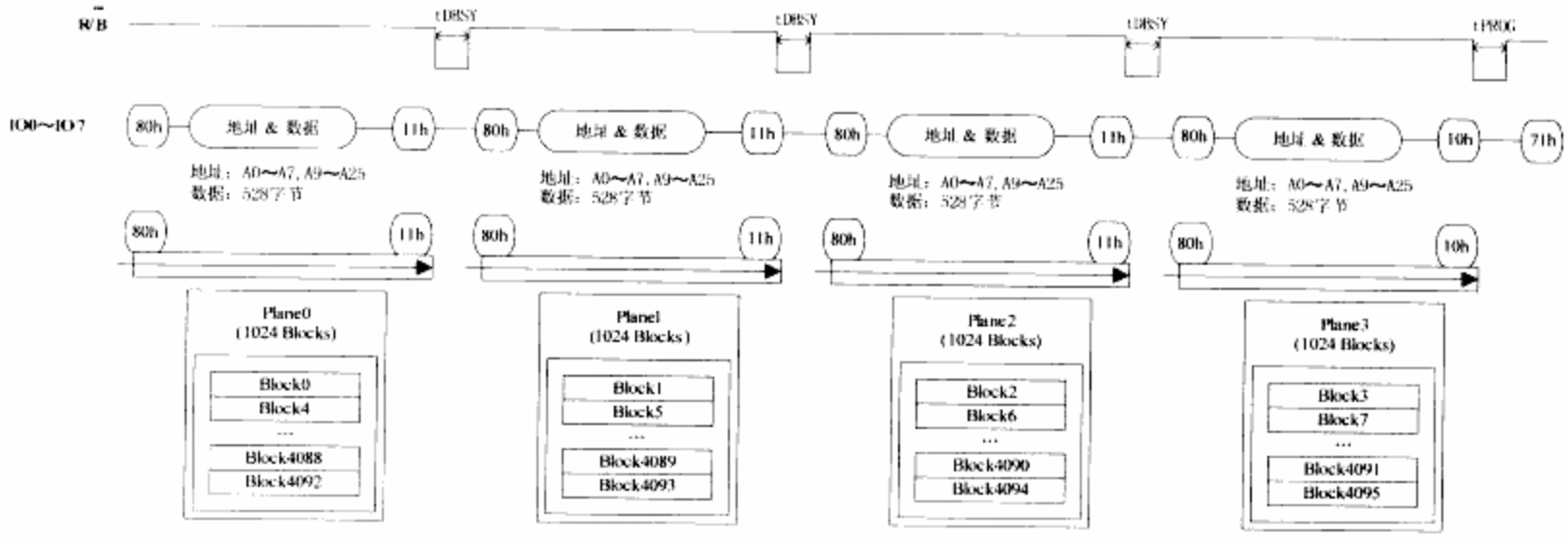


图 8.8　Four-Plane Page Program

发出命令字 80h、4 个地址序列及最多 528 字节的数据之后，发出命令字 11h（11h 称为“Dummy Page Program command”，相对地，10h 称为“True Page Program Command”）；接着对相邻层（plane）上的页进行同样的操作；仅在第 4 页的最后使用 10h 替代 11h，这样即可启动 Flash 内部的写操作。此时可以通过命令 71h 获知这些写操作是否完成、是否成功。

（7）Copy-Back Program(True)：命令字分 3 阶段，00h、8Ah、10h。

此命令用于将一页复制到同一层（plane）内的另一页，它省略了读出源数据、将数据重新载入 Flash，这使得效率大为提高。此命令有两个限制：源页、目的页必须在同一个层（plane）中，并且源地址、目的地址的 A14、A15 必须相同。

操作序列如图 8.9 所示。

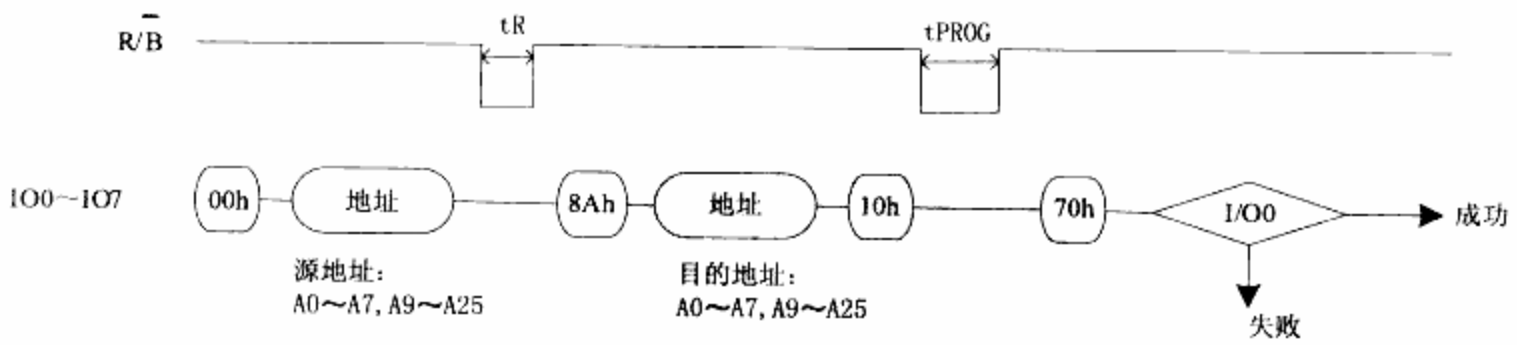


图 8.9　One Page Copy-Back program Operation

首先发出命令 Read 1(00h)、4 个源地址序列，此时源页的 528 字节数据很快就被全部读入内部寄存器中；接着发出命令字 8Ah(Page-Copy Data-input command)，随之发出 4 个目的地址序列；最后发出命令字 10h 启动对目的页的写操作。此后可以使用命令 70h 来查看此操作是否完成、是否成功。

（8）Copy-Back Program(Dummy)：命令字分 3 阶段，03h、8Ah、11h。

与命令 Page Program(Dummy)类似，Copy-Back Program(Dummy)可以同时启动对多达 4

个连续 plane 内的 Copy-Back Program 操作。操作序列如图 8.10 所示。

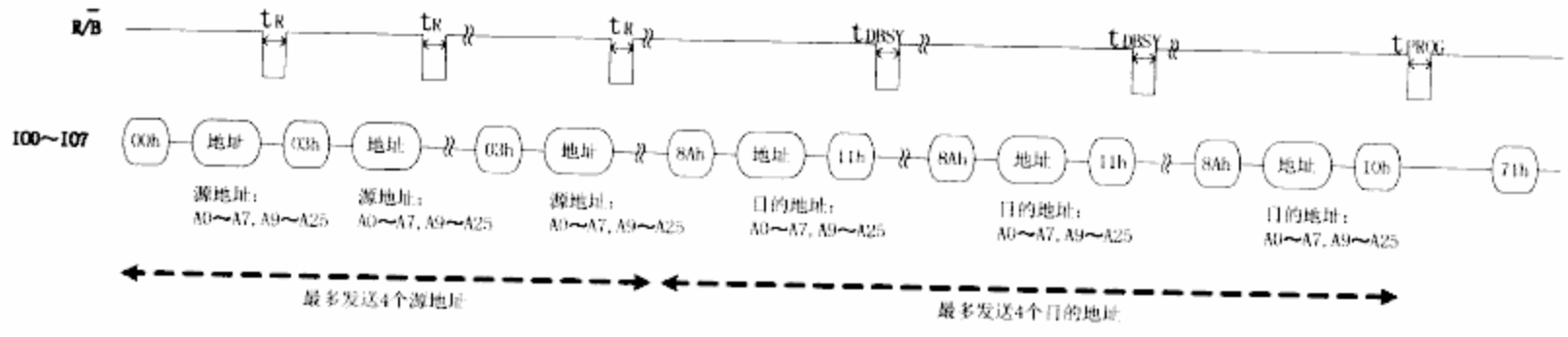


图 8.10 Four-Plane Copy-Back Page Program

从图 8.10 可知，首先发出命令字 00h、源页地址，这使得源页的 528 字节数据被读入所在 plane 的寄存器；对于随后的其他 plane 的源页，发出命令字 03h 和相应的源页地址将数据读入该 plane 的寄存器；按照前述说明读出最多 4 页的数据到寄存器后，发出命令字 8Ah、目的地址、命令字 11h，在发出最后一页的地址后，用 10h 代替 11h 以启动写操作。

（9）Block Erase：命令字分 3 阶段，60h、D0h。

此命令用于擦除 NAND Flash 块（block，大小为 16KB）。发出命令字 60h 之后，发出 block 地址——仅需要 3 个地址序列（请参考表 8.3，仅需要发出 2、3、4 cycle 所示地址），并且 A9～A13 被忽略。操作序列如图 8.11 所示。

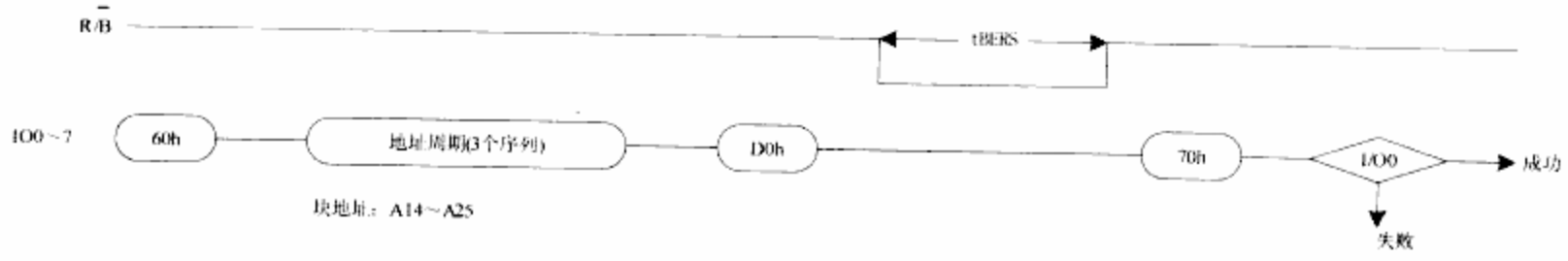


图 8.11 Block Erase Operation

（10）Multi-Plane Block Erase：60h----60h D0h

此命令用于同时擦除不同 plane 中的块。发出命令字 60h 之后，紧接着发出 block 地址序列，如此最多可以发出 4 个 block 地址，最后发出命令字 D0h 启动擦除操作。操作序列如图 8.12 所示。

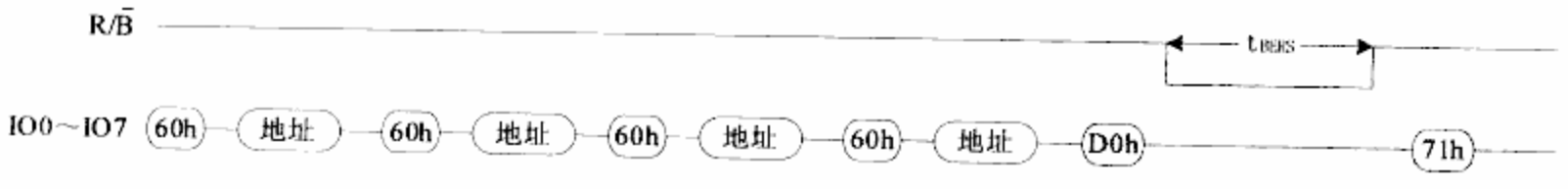


图 8.12 Four-Plane Block Erase Operation

（11）读状态命令有以下两种：

① Read Status：命令字为 70h。

② Read Multi-Plane Status：命令字为 71h。

Flash 中有状态寄存器，发出命令字 70h 或者 71h 之后，启动读操作即可读入此寄存器。状态寄存器中各位的含义如表 8.4 所示。

表 8.4　NAND Flash 读状态寄存器时各数据位的定义

I/O 引脚	所标识的状态	命令 70h 对应的定义	命令 71h 对应的定义
I/O0	总标记：成功/失败	成功：0，失败：1	成功：0，失败：1
I/O1	Plane0 的标记：成功/失败	忽略	成功：0，失败：1
I/O2	Plane1 的标记：成功/失败	忽略	成功：0，失败：1
I/O3	Plane2 的标记：成功/失败	忽略	成功：0，失败：1
I/O4	Plane3 的标记：成功/失败	忽略	成功：0，失败：1
I/O5	保留	忽略	忽略
I/O6	设备状态	忙：0，就绪：1	成功：0，失败：1
I/O7	写保护状态	保护：0，没有保护：1	保护：0，没有保护：1

注：① I/O0 是所有 Plane 的“总标记”，只要有一个 Plane 的操作是失败的，I/O0 就会被设为“失败”。
② I/O0～I/O4 引脚只标记它对应的 Plane。

8.1.4　S3C2410/S3C2440 NAND Flash 控制器介绍

NAND Flash 控制器提供几个寄存器来简化对 NAND Flash 的操作。比如要发出读命令时，只需要往 NFCMD 寄存器中写入 0 即可，NAND Flash 控制器会自动发出各种控制信号。

1．操作方法概述

访问 NAND Flash 时需要先发出命令，然后发出地址序列，最后读/写数据；需要使用各个使能信号来分辨是命令、地址还是数据。S3C2410 的 NAND Flash 控制器提供了 NFCONF、NFCMD、NFADDR、NFDATA、NFSTAT 和 NFECC 等 6 个寄存器来简化这些操作。S3C2440 的 NAND Flash 控制器则提供了 NFCONF、NFCONT、NFCMMD、NFADDR、NFDATA、NFSTAT 和其他与 ECC 有关的寄存器。对 NAND Flash 控制器的操作，S3C2410 和 S3C2440 有一点小差别：有些寄存器的地址不一样，有些寄存器的内容不一样，这在实例程序中会体现出来。

NAND Flash 的读写操作次序如下。

① 设置 NFCONF（对于 S3C2440，还要设置 NFCONT)寄存器，配置 NAND Flash。

② 向 NFCMD 寄存器写入命令，这些命令字可参考表 8.2。

③ 向 NFADDR 寄存器写入地址。

④ 读/写数据：通过寄存器 NFSTAT 检测 NAND Flash 的状态，在启动某个操作后，应该检测 R/nB 信号以确定该操作是否完成、是否成功。

2．寄存器介绍

下面讲解这些寄存器的功能及具体用法。

（1）NFCONF：NAND Flash 配置寄存器。

这个寄存器在 S3C2410、S3C2440 上功能有所不同。

① S3C2410 的 NFCONF 寄存器。

被用来使能/禁止 NAND Flash 控制器、使能/禁止控制引脚信号 nFCE、初始化 ECC、设置 NAND Flash 的时序参数等。

TACLS、TWRPH0 和 TWRPH1 这 3 个参数控制的是 NAND Flash 信号线 CLE/ALE 与写控制信号 nWE 的时序关系，如图 8.13 所示。

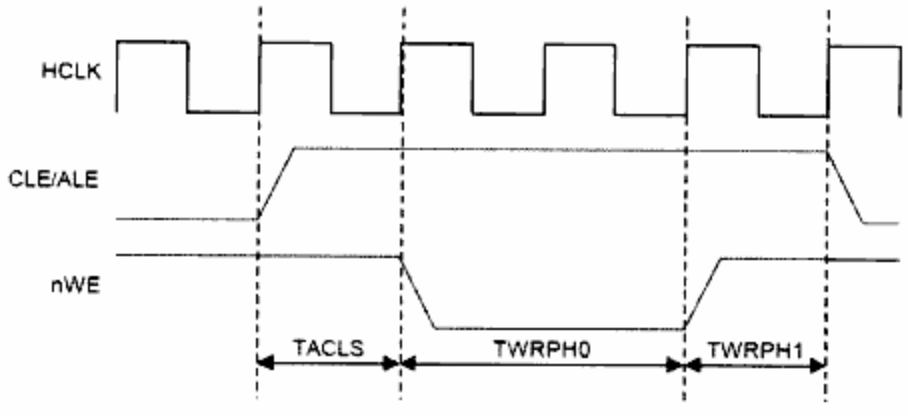


图 8.13 NAND FLASH MEMORY TIMING（TACLS = 0, TWRPH0 = 1, TWRPH1 = 0)

② S3C2440 的 NFCONF 寄存器。

被用来设置 NAND Flash 的时序参数 TACLS、TWRPH0、TWRPH1，设置数据位宽；还有一些只读位，用来指示是否支持其他大小的页（比如一页大小为 256/512/1024/2048 字节）。

它没有实现 S3C2410 的 NFCONF 寄存器的控制功能，这些功能在 S3C2440 的 NFCONT 寄存器里实现。

（2）NFCONT：NAND Flash 控制寄存器，S4C2410 没有这个寄存器。

被用来使能/禁止 NAND Flash 控制器、使能/禁止控制引脚信号 nFCE、初始化 ECC。它还有其他功能，在一般的应用中用不到，比如锁定 NAND Flash。

（3）NFCMD：NAND Flash 命令寄存器。

对于不同型号的 Flash，操作命令一般不一样。对于本板使用的 K9F1208U0M，请参考表 8.2。

（4）NFADDR：NAND Flash 地址寄存器。

当写这个寄存器时，它将对 Flash 发出地址信号。

（5）NFDATA：NAND Flash 数据寄存器。

只用到低 8 位，读、写此寄存器将启动对 NAND Flash 的读数据、写数据操作。

（6）NFSTAT：NAND Flash 状态寄存器。

只用到位 0，0:busy，1:ready。

8.2 NAND Flash 控制器操作实例：读 Flash

本实例讲述如何读取 NAND Flash，擦除、写 Flash 的操作与读 Flash 类似，读者可以自行编写程序。

8.2.1 读 NAND Flash 的步骤

下面讲述如何从 NAND Flash 中读出数据，假设读地址为 addr。

1. 设置 NFCONF（对于 S3C2440，还要设置 NFCONT）

（1）对于 S3C2410。

在本章实例中设为 0x9830——使能 NAND Flash 控制器、初始化 ECC、NAND Flash 片选信号 nFCE = 1（inactive，真正使用时再让它等于 0），设置 TACLS = 0，TWRPH0 = 3，TWRPH1 = 0。这些时序参数的含义为：TACLS = 1 个 HCLK 时钟，TWRPH0 = 4 个 HCLK 时钟，TWRPH1 = 1 个 HCLK 时钟。

K9F1208U0M 的时间特性如下：

```
CLE setup Time = 0 ns, CLE Hold Time = 10 ns,
ALE setup Time = 0 ns, ALE Hold Time = 10 ns,
WE Pulse Width = 25 ns
```

参考图 8.13，可以计算：即使在 HCLK=100MHz 的情况下，TACLS+TWRPH0+TWRPH1=6/100 μS = 60ns，也是可以满足 NAND Flash K9F1208U0M 的时序要求的。

（2）对于 S3C2440。

时间参数也设为：TACLS=0，TWRPH0=3，TWRPH1=0。NFCONF 寄存器的值如下：

```
NFCONF = 0x300
```

NFCONT 寄存器的取值如下，表示使能 NAND Flash 控制器、禁止控制引脚信号 nFCE、初始化 ECC。

```
NFCONT = (1<<4)|(1<<1)|(1<<0)
```

2．在第一次操作 NAND Flash 前，通常复位一下 NAND Flash

（1）对于 S3C2410。

```
NFCONF &= ~(1<<11) (发出片选信号)
NFCMD   = 0xff   (reset 命令)
```

然后循环查询 NFSTAT 位 0，直到它等于 1。

最后禁止片选信号，在实际使用 NAND Flash 时再使能。

```
NFCONF |= (1<<11) (禁止 NAND Flash)
```

（2）对于 S3C2440。

```
NFCONT &= ~(1<<1) (发出片选信号)
NFCMD   = 0xff   (reset 命令)
```

然后循环查询 NFSTAT 位 0，直到它等于 1。

最后禁止片选信号，在实际使用 NAND Flash 时再使能。

```
NFCONT |= 0x2    (禁止 NAND Flash)
```

3．发出读命令

先使能 NAND Flash，然后发出读命令。

（1）对于 S3C2410。

```
NFCONF &= ~(1<<11) (发出片选信号)
NFCMD   = 0 (读命令)
```

（2）对于 S3C2440。

```
NFCONT &= ~(1<<1) (发出片选信号)
NFCMD   = 0 (读命令)
```

4. 发出地址信号

这步请注意，表 8.3 列出了在地址操作的 4 个步骤对应的地址线，没用到 A8（它由读命令设置，当读命令为 0 时，A8=0；当读命令为 1 时，A8=1），如下所示：

```
NFADDR = addr & 0xff
NFADDR = (addr>>9) & 0xff       (左移 9 位，不是 8 位)
NFADDR = (addr>>17) & 0xff      (左移 17 位，不是 16 位)
NFADDR = (addr>>25) & 0xff      (左移 25 位，不是 24 位)
```

5. 循环查询 NFSTAT 位 0，直到它等于 1，这时可以读取数据了

6. 连续读 NFDATA 寄存器 512 次，得到一页数据（512 字节）

循环执行第 3、4、5、6 这 4 个步骤，直到读出所要求的所有数据。

7. 最后，禁止 NAND Flash 的片选信号

（1）对于 S3C2410。

```
NFCONF |= (1<<11)
```

（2）对于 S3C2440。

```
NFCONT |= (1<<1)
```

8.2.2 代码详解

实验代码在/work/hardware/nand 目录下，源文件为 head.S、init.c 和 main.c。本实例的目的是把一部分代码存放在 NAND Flash 地址 4096 之后，当程序启动后通过 NAND Flash 控制器将它们读出来、执行。以前的代码都小于 4096 字节，开发板启动后它们被自动复制进“Steppingstone”中。

连接脚本 nand.lds 把它们分为两部分，nand.lds 代码如下：

```
01 SECTIONS {
02   firtst     0x00000000 : { head.o init.o nand.o}
03   second 0x30000000 : AT(4096) { main.o }
```

```
04 }
05
```

第 2 行表示 head.o、init.o、nand.o 这 3 个文件的运行地址为 0，它们在生成的映象文件中的偏移地址也为 0（从 0 开始存放）。

第 3 行表示 main.o 的运行地址为 0x30000000，它在生成的映象文件中的偏移地址为 4096。

head.S 调用 init.c 中的函数来关 WATCH DOG、初始化 SDRAM；调用 nand.c 中的函数来初始化 NAND Flash，然后将 main.c 中的代码从 NAND Flash 地址 4096 开始处复制到 SDRAM 中；最后跳到 main.c 中的 main 函数继续执行。

由于 S3C2410、S3C2440 的 NAND Flash 控制器并非完全一样，这个程序要既能处理 S3C2410，也能处理 S3C2440，首先需要分辨出是 S3C2410 还是 S3C2440，然后使用不同的函数进行处理。读取 GSTATUS1 寄存器，如果它的值为 0x32410000 或 0x32410002，就表示处理器是 S3C2410，否则是 S3C2440。

nand.c 向外引出两个函数：用来初始化 NAND Flash 的 nand_init 函数、用来将数据从 NAND Flash 读到 SDRAM 的 nand_read 函数。

1. nand_init 函数分析

代码如下：

```
252 /* 初始化 NAND Flash */
253 void nand_init(void)
254 {
255 #define TACLS   0
256 #define TWRPH0  3
257 #define TWRPH1  0
258
259     /* 判断是 S3C2410 还是 S3C2440 */
260     if ((GSTATUS1 == 0x32410000) || (GSTATUS1 == 0x32410002))
261     {
262         nand_chip.nand_reset            = s3c2410_nand_reset;
263         nand_chip.wait_idle             = s3c2410_wait_idle;
264         nand_chip.nand_select_chip      = s3c2410_nand_select_chip;
265         nand_chip.nand_deselect_chip    = s3c2410_nand_deselect_chip;
266         nand_chip.write_cmd             = s3c2410_write_cmd;
267         nand_chip.write_addr            = s3c2410_write_addr;
268         nand_chip.read_data             = s3c2410_read_data;
269
270         /* 使能 NAND Flash 控制器，初始化 ECC，禁止片选，设置时序 */
271         s3c2410nand->NFCONF = (1<<15)|(1<<12)|(1<<11)|(TACLS<<8)|(TWRPH0
```

```
<<4)|(TWRPH1<<0);
    272     }
    273     else
    274     {
    275         nand_chip.nand_reset            = s3c2440_nand_reset;
    276         nand_chip.wait_idle             = s3c2440_wait_idle;
    277         nand_chip.nand_select_chip      = s3c2440_nand_select_chip;
    278         nand_chip.nand_deselect_chip    = s3c2440_nand_deselect_chip;
    279         nand_chip.write_cmd             = s3c2440_write_cmd;
    280         nand_chip.write_addr            = s3c2440_write_addr;
    281         nand_chip.read_data             = s3c2440_read_data;
    282
    283         /* 设置时序 */
    284         s3c2440nand->NFCONF = (TACLS<<12)|(TWRPH0<<8)|(TWRPH1<<4);
    285         /* 使能 NAND Flash 控制器，初始化 ECC，禁止片选 */
    286         s3c2440nand->NFCONT = (1<<4)|(1<<1)|(1<<0);
    287     }
    288
    289     /* 复位 NAND Flash */
    290     nand_reset();
    291 }
```

第 260 行读取 GSTATUS1 寄存器来判断为 S3C2410 还是 S3C2440，然后分别处理：S3C2410、S3C2440 的 NAND Flash 控制器中，有一些寄存器的功能是相同的，但是它们的地址不一样；有一些寄存器的功能已经发生变化。所以使用两套函数来进行处理。

第 262～268 行设置 S3C2410 的 NAND Flash 处理函数，第 275～281 行设置 S3C2440 的 NAND Flash 处理函数，把这些函数赋给 nand_chip 结构，以后通过这个结构来调用。这只是一个编程技巧，代码的关键还是 s3c2410_nand_reset、s3c2440_nand_reset 等函数。

如果处理器是 S3C2410，则调用第 271 行的代码设置 NFCONF 寄存器：使能 NAND Flash 控制器，初始化 ECC，禁止片选，设置时序。如果处理器是 S3C2440，则使用第 284、286 两行代码来进行相同的设置（涉及两个寄存器：NFCONF 和 NFCONT）。

最后，第 290 行调用 nand_reset 函数复位 NAND Flash。在第一次使用之前通常复位一下。

其中涉及的各个函数都只有几行，主要是读写寄存器。

2. nand_read 函数分析

它的原型如下，表示从 NAND Flash 位置 start_addr 开始，将数据复制到 SDRAM 地址 buf 处，共复制 size 字节。

```
void nand_read(unsigned char *buf, unsigned long start_addr, int size)
```

代码如下：

```
297 /* 读函数 */
298 void nand_read(unsigned char *buf, unsigned long start_addr, int size)
299 {
300     int i, j;
301
302     if ((start_addr & NAND_BLOCK_MASK) || (size & NAND_BLOCK_MASK)) {
303         return ;    /* 地址或长度不对齐 */
304     }
305
306     /* 选中芯片 */
307     nand_select_chip();
308
309     for(i=start_addr; i < (start_addr + size);) {
310       /* 发出 READ0 命令 */
311       write_cmd(0);
312
313       /* Write Address */
314       write_addr(i);
315       wait_idle();
316
317       for(j=0; j < NAND_SECTOR_SIZE; j++, i++) {
318           *buf = read_data();
319           buf++;
320       }
321     }
322
323     /* 取消片选信号 */
324     nand_deselect_chip();
325
326     return ;
327 }
```

可以看到，读 NAND Flash 的操作分为 6 步。

① 选择芯片（第 307 行）。

② 发出读命令（第 311 行）。

③ 发出地址（第 314 行）。

④ 等待数据就绪（第 315 行）。

⑤ 读取数据（第 318 行）。

⑥ 结束后，取消片选信号（第 324 行）。

流程如图 8.14 所示。

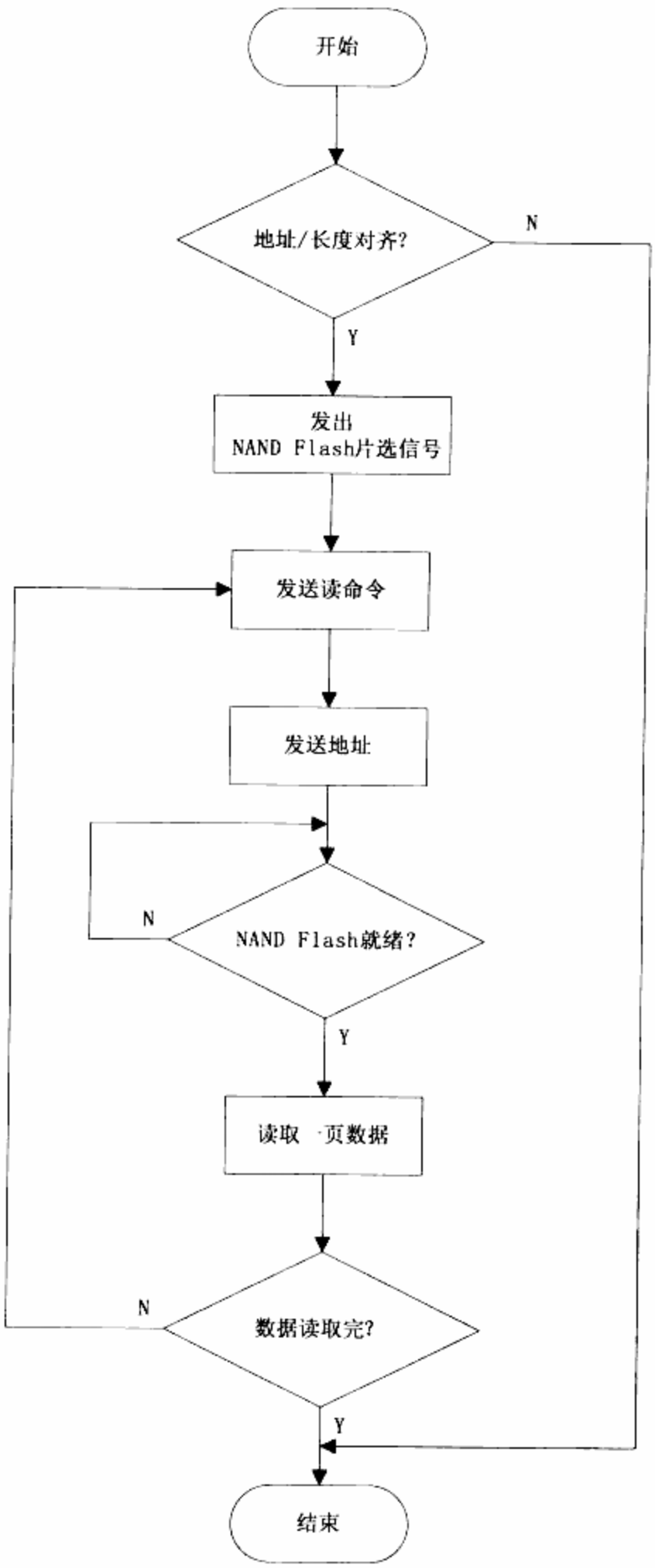
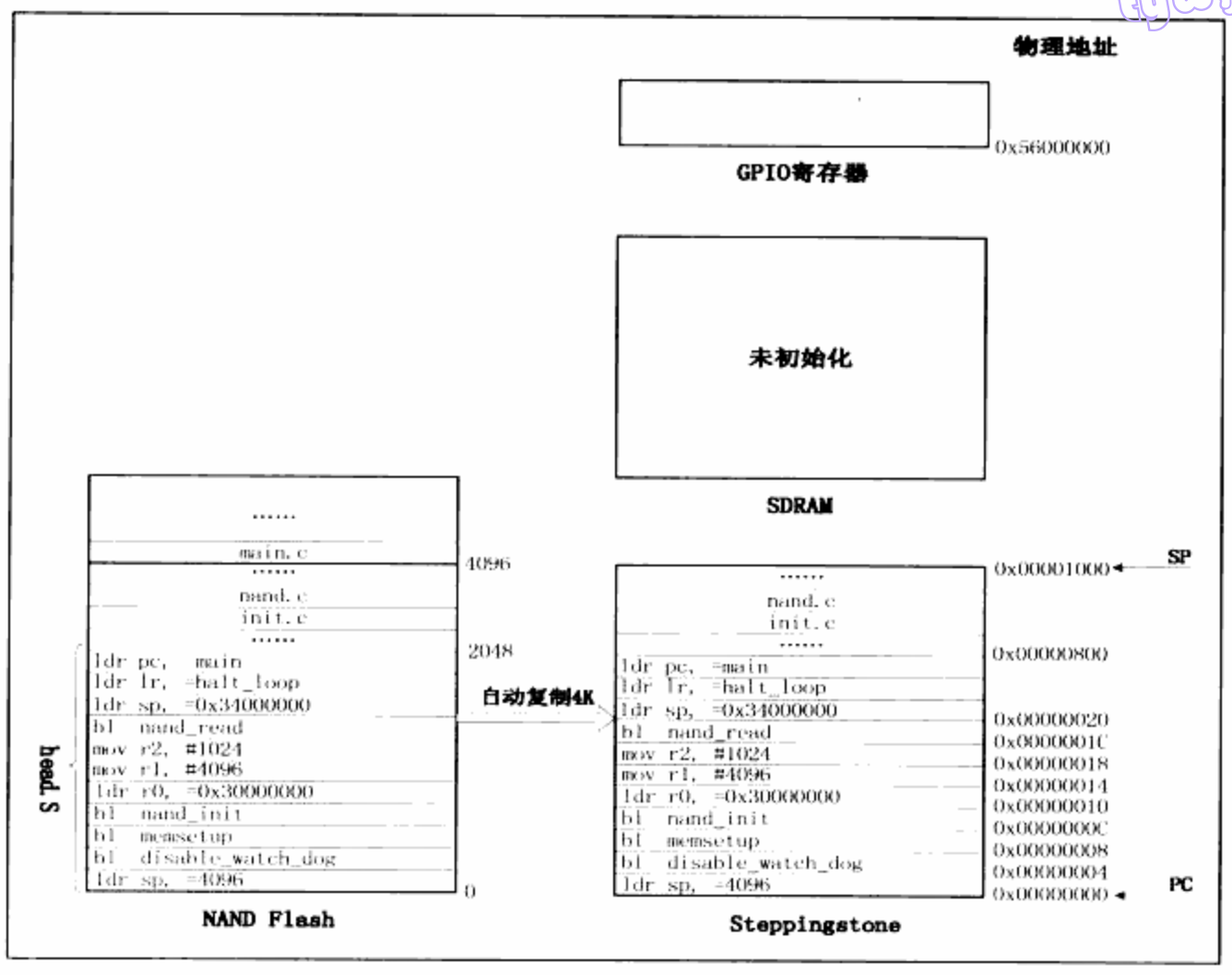
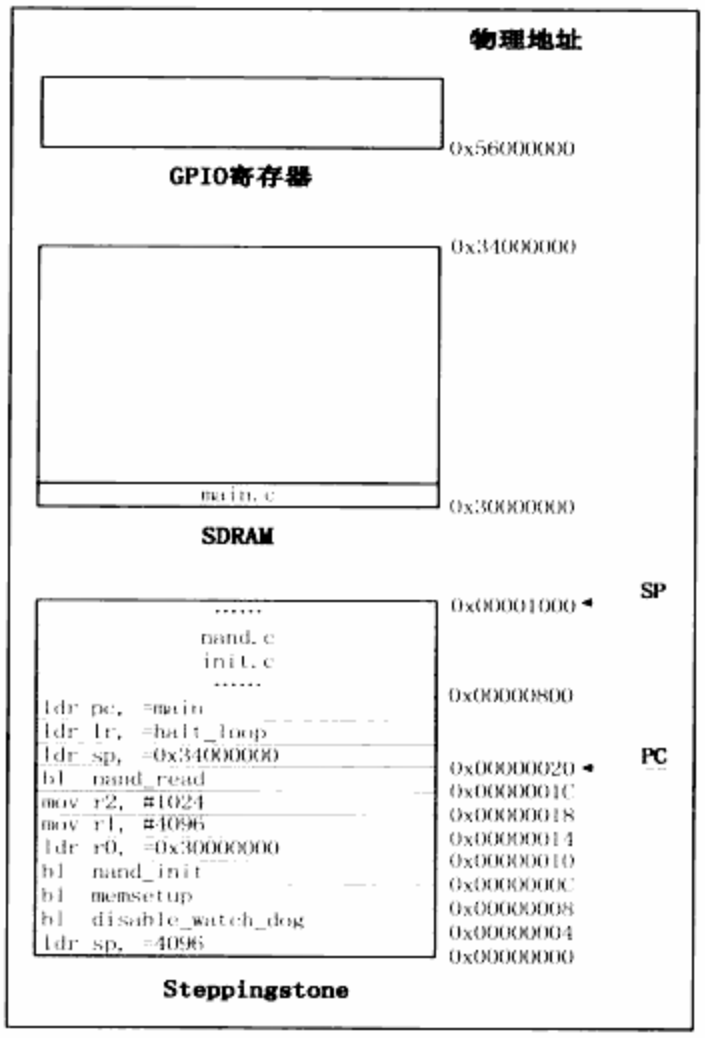


图 8.14 NAND Flash 读操作流程

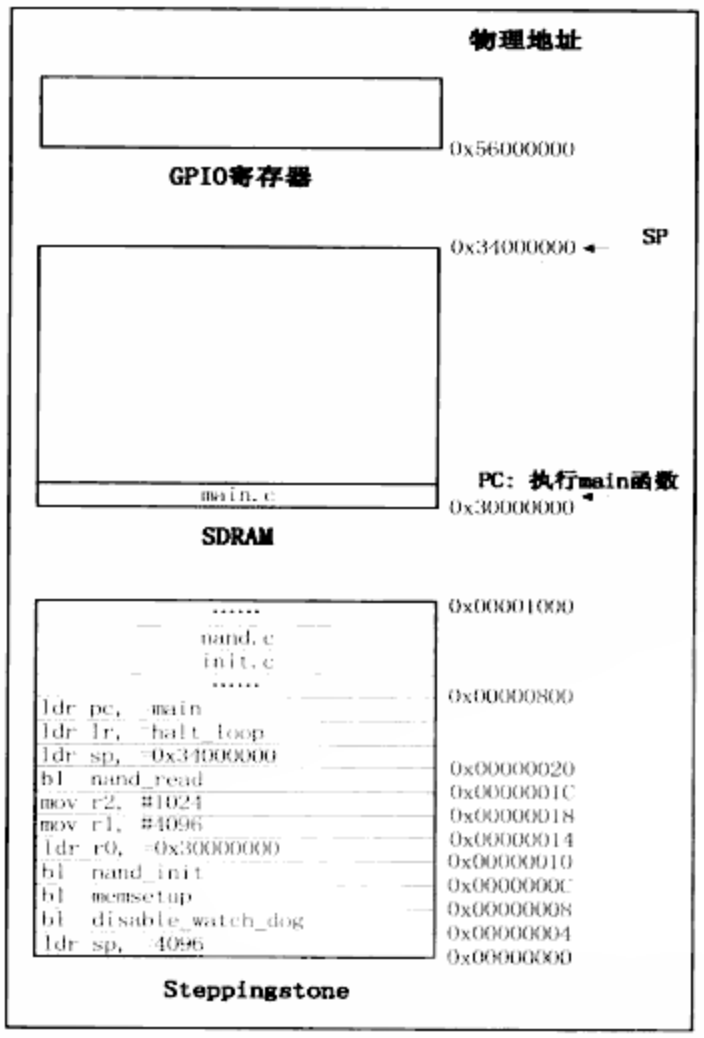
为了更形象地了解程序执行时代码复制、程序执行位置，请参考图 8.15。



1. 从NAND Flash启动



2. nand_read复制main.c的代码到SDRAM中



3. 执行SDRAM中的main函数

图 8.15　从 NAND Flash 复制代码到 SDRAM 并执行的过程