

请注意赛普拉斯已正式并入英飞凌科技公司。

此封面页之后的文件标注有“赛普拉斯”的文件即该产品为此公司最初开发的。请注意作为英飞凌产品组合的部分,英飞凌将继续为新的及现有客户提供该产品。

文件内容的连续性

事实是英飞凌提供如下产品作为英飞凌产品组合的部分不会带来对于此文件的任何变更。未来的变更将在恰当的时候发生,且任何变更将在历史页面记录。

订购零件编号的连续性

英飞凌继续支持现有零件编号的使用。下单时请继续使用数据表中的订购零件编号。

512-Mbit (64 Mbyte)/256-Mbit (32 Mbyte)/ 128-Mbit (16 Mbyte), 1.8 V/3.0 V, HyperFlash™ 产品系列

独有特性

HyperBus™ 低信号计数接口

- 3.0 V I/O, 11 个总线信号
 - 单端时钟
- 1.8 V I/O, 12 个总线信号
 - 差分时钟 (CK、CK#)
- 芯片选择 (CS#)
- 8 位数据总线 (DQ[7:0])
- 读写数据选通 (RWDS)
 - HyperFlash 存储器只将 RWDS 用作读取数据选通

高性能

- 高达 333 MB/s 的可持续读取吞吐量
- 双数据速率 (DDR): 每个时钟周期传输两字节的数据
- $V_{CC} = 1.8\text{ V}$ 时, 时钟频率可达 166 MHz (相应速率为 333 MB/s)
- $V_{CC} = 3.0\text{ V}$ 时, 时钟频率可达 100 MHz (相应速率为 200 MB/s)
- 96 ns 的初始随机读取访问时间
 - 初始随机访问的读取延迟: 5-16 个时钟周期
- 连续突发转换
- 可配置突发特性
 - 回卷突发长度:
 - 16 个字节 (8 个时钟周期)
 - 32 个字节 (16 个时钟周期)
 - 64 个字节 (32 个时钟周期)
 - 线性突发
 - 混合选项 — 一个回卷, 随后是线性突发
 - 每次数据转换中所选定的复位或线性突发类型
 - 可配置输出驱动强度
- 低功耗模式
 - 有效时钟将在读取期间停止操作, 其电流为 12 mA, 无需唤醒
 - 待机模式下的电流为 25 μA (典型值), 无需唤醒
 - 深度掉电模式下的电流为 8 μA (典型值)
 - 唤醒器件需要的时间: 300 μs

- 通过 INT# 输出生成外部中断
 - 繁忙 - 就绪状态转换
 - 通过 RSTO# 输出生成系统级上电复位
 - 用户可配置的 RSTO# 低电平周期
 - 512 字节的编程缓冲区
 - 以半页 (16 个字节) 的倍数, 最多 512 字节进行编程
 - 扇区擦除
 - 各扇区的空间统一为 256 kB
 - 可选的八个 4 kB 参数扇区 (总空间: 32 kB)
 - 高级扇区保护
 - 为每个扇区提供易失性和非易失性的保护方法
 - 每次可编程一个大小为 1024 字节的阵列
 - 工作温度
 - 工业级范围 (-40°C 到 85°C)
 - 扩展的工业级范围 (-40°C 到 $+105^{\circ}\text{C}$)
 - 扩展范围 (-40°C 到 $+125^{\circ}\text{C}$)
 - 获得了 ISO/TS16949 和 AEC Q100 标准认证
 - 耐久性
 - 所有扇区的擦写次数为 100000 个周期 (最小值)
 - 保持时间
 - 20 年的数据保持时间 (典型值)
 - 上电复位、擦除和编程的电流
 - 最大峰值电流 $\leq 100\text{ mA}$
 - 封装选项
 - 24 球 FBGA
- 文档编号: 001-99456 版本 *B

性能总结

读取访问时序	
$V_{CC}/V_{CCQ} = 1.8\text{ V}$ 时的时钟最大频率	166 MHz
$V_{CC}/V_{CCQ} = 3.0\text{ V}$ 时的最大时钟频率	100 MHz
最长访问时间, (t_{ACC})	96 ns
时钟频率为 166 MHz 时访问第一个字的最长 CS# 时间	118 ns

典型编程 / 擦除时间	
单字编程 (2 B = 16 b)	500 μs (~4 kB/s)
写入缓冲区编程 (512 B = 4096 b)	475 μs (~1 MB/s)
扇区擦除时间 (256 kB = 2 Mb)	930 ms (~282 kB/s)

典型的电流消耗	
突发读取 (频率为 166 MHz 时进行连续读取)	80 mA
上电复位	80 mA
扇区擦除电流	60 mA
写入缓冲区编程电流	60 mA
待机模式 (CS# = 高电平)	25 μA
深度掉电模式 (CS# = 高电平, 85°C)	30 μA (512 Mb)
	4 μA (其他所有容量)

目录

1. 概述	4	9. 数据完整性	56
2. 产品概述	4	9.1 擦写次数	56
3. 信号说明	6	9.2 数据保持时间	56
4. HyperBus 协议	7	9.3 器件 ID 和通用闪存接口 (ID-CFI) ASO 映射.....	57
4.1 指令 / 地址位分配	8	硬件接口	
4.2 读取操作.....	9	10. 接口状态	61
4.3 写入操作.....	12	10.1 硬件复位	61
软件接口		10.2 关闭电源 (硬件数据保护)	61
5. 地址空间映射	14	10.3 节能模式	61
5.1 闪存存储器阵列.....	15	11. 电气规范	63
5.2 设备 ID 和 CFI (ID-CFI) ASO	17	11.1 最大绝对额定值.....	63
6. 嵌入式操作	19	11.2 直流特性 (与 CMOS 兼容)	63
6.1 嵌入式算法控制器 (EAC)	19	11.3 加电和掉电.....	65
6.2 编程和擦除简介.....	19	12. 时序规范	69
6.3 数据保护.....	41	12.1 交流电特性.....	69
6.4 易失性与非易失性配置寄存器汇总	50	12.2 字突发数据加载的字编程	69
7. 软件接口参考	51	13. 物理接口	71
7.1 指令汇总.....	51	14. 订购信息	72
8. 嵌入式算法性能	55	14.1 订购器件编码	72
		14.2 有效组合	73
		15. 文档修订记录	74

1. 概述

Spansion® HyperFlash™ 系列产品是高速的 CMOS、MirrorBit® NOR 闪存器件，通过 HyperBus™ 低信号计数 DDR（双数据速率）接口可实现高速的读取吞吐量。在每一个时钟周期内，DDR 接口协议支持在 DQ 输入 / 输出信号传输两个数据字节。HyperFlash 的每一个读取或写入访问操作包含一系列的 16 位宽、一个时钟周期长的数据传输（位于内部 HyperFlash 核心中）以及两个相应的 8 位宽、半个时钟周期长的数据转换（位于 DQ 信号上）。

所有的数据和指令 / 地址都是按照 DDR 方式通过 8 位数据总线传输的。接收 DQ 信号上的指令 / 地址 / 数据信息时，HyperFlash 器件将使用时钟输入信号来捕获信息。读取写入数据选通（RWDS）是 HyperFlash 器件的输出信号，用于指示存储器中的数据被传输到主机。在读取操作的数据传输阶段，RWDS 将参照时钟的上升沿和下降沿。指令 / 地址 / 写入数据值的中心同各个时钟沿对齐的，而各个读取数据值同 RWDS 的转换边沿对齐。

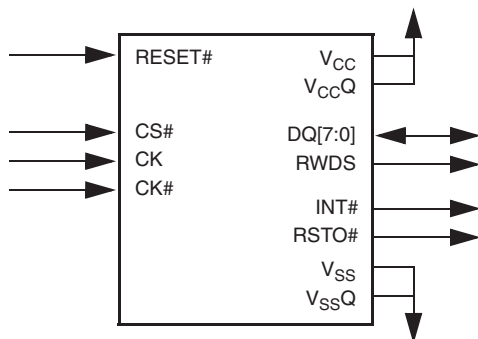
HyperFlash 器件的读和写操作都是按照突发序列实现的。读取操作可以被指定为使用回卷突发或线性突发。在回卷突发期间，访问操作将从所选定的位置开始，然后继续访问回卷突发序列组中的其他预设个数的位置。在线性突发期间，访问操作将从选定的位置开始，然后继续按序列地访问，直到读取操作被中止，CS# 返回高电平为止。写操作会传输一个或多个 16 位数值。

所有输入都与 LV-CMOS 兼容。这些器件使用 V_{CC} 和 V_{CCQ} 提供的 1.8 V 或 3.0 V（额定值）电源。更多有关 HyperBus 接口工作原理的信息，请参考 HyperBus 规格。本数据手册仅介绍了 HyperBus 接口操作中有关 HyperFlash 器件方面的内容。

2. 产品概述

HyperFlash 系列产品包括多种容量、1.8 V 或 3.0 V 的内核和 I/O、非易失性、同步闪存存储器器件。这些器件具有一个 8 位（一个字节）带宽的 DDR 数据总线，并且只使用了一个字宽（16 位数据）的地址边界。执行读取操作时，每一个时钟周期内将传输 16 位数据（在时钟上下沿各传输 8 位）。写作操作期间，在每一个时钟周期也输 16 位数据（在每个时钟边沿传输 8 位）。

图 2.1 HyperFlash 接口



HyperFlash 存储器使用了 HyperBus 低信号计数、高速的接口，并结合了芯片内执行（XIP）功能和数据存储闪存存储器的最佳特性。在拥有大容量、快速编程的数据存储闪存的基础上，该系列产品还能对 XIP 闪存进行快速的随机访问。

每个随机读取操作都会访问一个 32 字节长的数据；每一组对齐的数据被称为一页。每页中都包含一对 16 字节的阵列数据对齐组，该组被称为半页。所有半页都同 16 字节的地址边界对齐。每个读取操作需要两个时钟周期来定义目标行地址和突发类型，以及等于 t_{ACC} 的初始访问延迟时间。在初始延迟期间，第三个时钟周期指定了目标半页内的开始地址。初始数据值被读取后，在下一个时钟周期内可使用回卷或线性序列读取页中的其它数据。配置为线性突发模式时，当一个页面正向外输出数据时，器件将自动提取 MirrorBit 闪存存储器阵列中下一页中的数据。通过这种同时进行的突发输出和提取阵列中的数据，可以实现线性突发操作保持数据传输速率可达 333 MB/s（1 个字节（8 位数据总线）* 2（两个时钟沿上的数据）* 166 MHz = 333 MB/s）。

S26KL/S26KS 地址映射

类型	计数	地址	注意
字地址位于半页（16 个字节）内	8（字地址）	A2 – A0	16 个字节
字地址位于写缓冲区操作行（512 个字节）内	256（字地址）	A7 – A0	512 个字节
半页（16 个字节）位于擦除扇区（256 kB）内	8192（半页）	A16 – A3	
写缓冲区操作行（512 个字节）位于擦除扇区（256 kB）内	512（行）	A16 – A8	
擦除扇区（256 kB）的总数	256（512 Mb） 128（256 Mb） 64（128 Mb）	Amax – A17	

器件控制逻辑被分成两个并行的操作区：主机接口控制器（HIC）和嵌入式算法控制器（EAC）。其中，HIC 监控器件输入端的信号电平，并根据需要驱动输出，以完成器件与主机系统（HyperBus 主设备）之间的读取和写入数据传输。HIC 在读取传输时传递当前进入的地址映射的数据；将写入传输地址和数据信息输入到 EAC 指令存储器内；告知 EAC 电源转换和写入传输的情况。EAC 在写入传输后查看指令存储器中的合法指令序列，并执行相关的嵌入式算法。

更改存储阵列中的非易失性数据时，需要执行复杂的操作序列，这些操作被称为嵌入式算法（EA）。这些算法完全由设备内部的 EAC 管理。主算法执行主闪存阵列数据的编程和擦除。主机系统将指令代码写入到闪存器件地址空间内。EAC 接收指令，然后执行所有必要的步骤以完成该指令，并在 EA 执行期间提供状态信息。

每个存储器位的擦除状态为一个逻辑 1。编程操作会将逻辑 1（高电平）修改为逻辑 0（低电平）只能通过擦除操作才可以将逻辑 0 修改为逻辑 1。擦除操作必须在整个扇区（即 256 kbyte（或 4 kbyte 的参数扇区）执行。从 Spansion 出厂时，所有扇区均为擦除状态。

编程通过一个 512 字节的写入缓冲区完成。在开始编程操作前，可以向写入缓冲区内任意位置写入 1 到 256 个字。在闪存阵列内，将每个以 512 字节大小的对齐的数据组称为一行。编程操作将易失性写入缓冲区内的数据传输到非易失性存储阵列行中。该操作称为写入缓冲区编程。

在复位后或者使用写入缓冲区完成任意操作后，写入缓冲区内容将被置 1。对于写入到缓冲区指令未写入 0 的位置，在默认情况下仍为 1。在编程操作期间，写入缓冲区中的任何 1 都不会影响存储阵列中的数据。

除了 HyperBus 专用的各个必需的信号外，器件还包括一个复位（RESET#）输入、一个中断（INT#）输出和一个复位输出（RSTO#）。

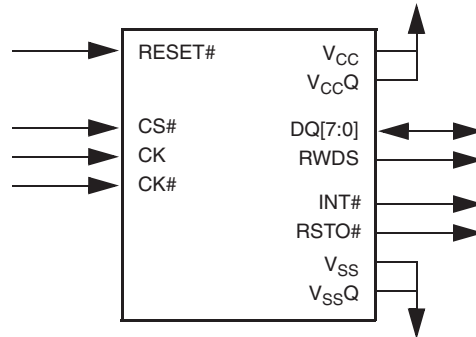
当 RESET# 输入从低电平转换为高电平时，器件将返回到默认状态（上电复位（POR）事件后的状态）。

INT# 输出可向 HyperBus 主设备提供中断，用于指示 HyperFlash 在编程或者擦除完成后状态从繁忙转换到就绪。

RSTO# 是一个开漏输出。它用于表示在器件内发生了一个上电复位（POR）事件，并可将其作为系统级复位信号。当内部加电复位事件完成时，用户定义的超时周期完成后，RSTO# 信号将从低阻态切换到高阻态。成功切换到高阻抗状态后，外部上拉电阻使 RSTO# 转为高电平，器件立即进入空闲状态。

3. 信号说明

图 3.1 HyperFlash 接口



信号描述

符号	类型	说明
CS#	输入	片选。当该信号从高电平转变为低电平时，将启动 HyperFlash 总线操作。当该信号从低电平变为高电平时，结束 HyperFlash 总线操作。
CK、CK#	输入	差分时钟。命令 - 地址 / 数据信息可以是输入也可以是输出，具体取决于 CK 和 CK# 信号之间的关系 CK# 仅适用于 1.8 V 的器件，在 3 V 器件上，可以悬空或连接到 CK。
RWDS	输出	读写数据选通。在读取操作时，输出数据是和 RWDS 边沿对齐的。
DQ[7..0]	输入 / 输出	数据输入 / 输出。在读写操作期间，通过这些 DQ 信号传输指令 - 地址 / 数据信息。
INT#	输出（开漏）	INT 输出。当该引脚处于低电平状态时，表明器件发生了一个内部事件。该引脚用于器件的系统级中断，以便表明已发生一个片上事件。INT# 是开漏输出。
RESET#	输入	硬件复位。当该信号处于低电平状态时，器件将进行自初始化并返回到空闲态。当 RESET# 处于低电平状态时，RWDS 和 DQ[7:0] 则将进入高阻态。RESET# 包括一个弱上拉电阻，如果 RESET# 保持未连接状态，它则将被上拉到高电平状态。
RSTO#	输出（开漏）	RSTO# 输出。RSTO# 是一个开漏输出，它用于表明在器件内发生加电复位（POR）事件。它还能够用作系统级复位信号。当内部加电复位事件完成时，用户定义的超时周期完成后，RSTO# 信号将从低阻态切换到高阻态。成功切换到高阻态后，外部上拉电阻将 RSTO# 置高，此时，器件会立即进入空闲态。
V _{CC}	电源	内核功耗。
V _{CCQ}	电源	输入 / 输出电源。
V _{SS}	电源	内核接地。
V _{SSQ}	电源	输入 / 输出 接地。

4. HyperBus 协议

所有总线数据操作都可分为读取或写入操作两类。一个总线数据操作从 **CS#** 为低电平、**CK** 为低电平且 **CK#** 为高电平时开始。在前三个时钟周期内，所执行的数据操作将以 **DDR** 的方式通过全部六个时钟沿被传入到 **HyperFlash** 器件内。在这三个时钟周期内，将传输指令 / 地址（**CA0**、**CA1**、**CA2**）信息的三个字来定义操作的下列特性：

- 读取操作还是写入操作。
- 对存储器阵列还是寄存器操作
 - 虽然 **HyperBus** 协议是提供给具有存储器和寄存器地址空间的从设备的，但本规格中所介绍的 **HyperFlash** 存储器并没有将存储器和寄存器区分为单独的地址空间。任意数据操作都特定针对单个地址空间，而不管操作指示的目标位置是在存储器空间还是寄存器空间内，该地址都不会发生改变。写入操作始终将操作地址和数据被放置到一个指令寄存器组中（缓冲区）。读取操作可读回存储器阵列或寄存器地址空间窗口中单个地址空间范围内由各指令的执行暂时覆盖的数据。采用寄存器空间覆盖层方法的单个地址范围与传统的并行 **NOR** 闪存存储器和擦除软件驱动器向后兼容。
- 读取操作使用的是线性突发序列还是回卷突发序列
 - 写入指令操作在每次操作传输单个字。字编程写入数据传输可通过一个高达 50 MHz 的线性突发实现。写入操作不支持回卷突发序列，并会忽略突发类型的指示。
- 目标半页地址（行地址和高位列地址）。
- 目标字（在半页内）地址（低位列地址）。

数据操作被定义后，会占用几个空闲时钟周期来满足传输数据前的初始读取访问延迟。一旦完成传输目标数据，**HyperBus** 主机会在 **CK** 为低电平和 **CK#** 为高电平时将 **CS#** 驱动为高电平，以表示操作完毕。每次可传输 16 位数据，其中前 8 位（15-8）在 **CK** 的上升沿上（写入数据或 **CA** 位）或 **RWDS** 沿上（读取数据）被传输，余下的 8 位（7-0）则在 **CK** 的下降沿上或 **RWDS** 沿上被传输。**CK** 为低电平、**CK#** 为高电平时，将 **CS#** 置高可随时停止对数据进行的读取或写入操作。读取数据与 **RWDS** 转换边沿对齐，写入数据中心与时钟沿对齐。

4.1 指令 / 地址位分配

指令 / 地址位分配

CA 位 #	位名	位功能
47	R/W#	将数据操作识别为读或写。 数值 1 表示读操作 数值 0 表示写操作 目标空间定义在 CA46 内。
46	目标	表示是对存储器或寄存器空间进行读或写操作。 数值 0 表示对存储器空间进行操作 数值 1 表示对寄存器空间进行操作 该寄存器空间供易失性存储器和外设设备使用。在读或写数据操作期间，HyperFlash 器件不会使用该特性，并且要将该位设置为 0。
45	突发类型	表示突发序列是线性的还是循环的。 数值 0 表示回卷突发 数值 1 表示线性突发
44-38 (512 Mb) 44-37 (256 Mb) 44-36 (128 Mb)	保留	保留用于将来进行地址扩展 应通过主机控制器应将各保留位的值设置为 0。
37-16 (512 Mb) 36-16 (256 Mb) 35-16 (128 Mb)	行与高位列地址	目标地址的半页部分。
15-3	保留	留给将来的列地址扩展使用 应通过主机控制器应将各保留位的值设置为 0。
2-0	低位列地址	目标地址的低位列地址：系统中的字地址位 (A2-A0) 选择半页内的起始字。

4.2 读取操作

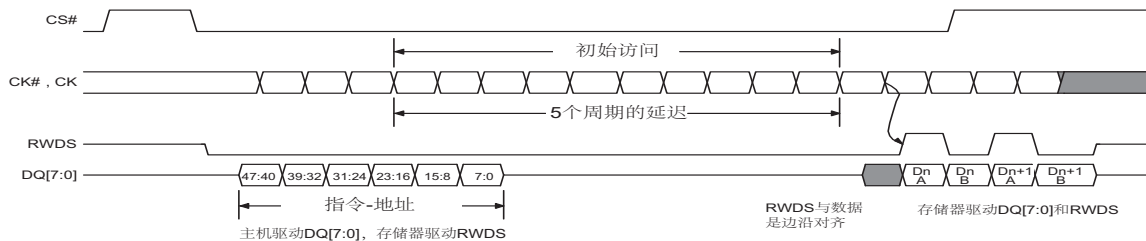
CA0 表示将执行一个读取操作，并指出它的突发类型（回卷或线性）。一旦半页地址被传输到 CA0 和 CA1 内，读操作便可以进行内部阵列访问。CA2 表示选定的半页内的目标字地址。然后，主机将根据配置寄存器中延迟计数设置所规定的周期数量继续计时。完成这些延迟周期后，存储器同时开始切换读 - 写数据选通（RWDS），并输出目标数据。新数据输出与 RWDS 转换是边沿对齐的。只要主机切换时钟（CK 和 CK#），便会继续输出数据。回卷突发持续在突发长度内进行循环，但线性突发将按序列输出数据，甚至会超出页界限。混合突发包含了一个初始回卷突发和在其后面的一个线性突发，如第 47 页上的 6.3.11 章节，混合突发所示。回卷突发可在主阵列，CFI 表和安全硅区域中执行。CK 为低电平、CK# 为高电平时，将 CS# 置高可随时停止读取操作。

在线性读取期间，如果超出了目标地址页和下一页的界限，则可能需要等待额外的延迟周期（第 11 页上的线性读操作跨越第一个页边界时（延迟计数为 11 个时钟周期）表和第 11 页上的线性读操作第一次跨越页边界时（延迟计数为 16 个时钟周期）表）。延迟周期的数量取决于目标页内的开使地址以及由配置寄存器指定的初始延迟周期数量。

当一个线性突发到达整个阵列的最后地址时，它将回卷到地址 0。

16 字节和 32 字节的回卷突发始终在页界限内完成，并不会引起由超出页界限引起的延迟。对于一个 64 字节的回卷突发读取操作，根据开使地址的具体情况（第 32 页上的 64 字节的回卷突发地址序列（延迟代码 = 16）表），当从目标地址跨越边界到下一页时，可能需要延迟周期。

图 4.1 读取操作



注意:

1. 必须将 CK 设置为低电平，并将 CK# 设置为高电平，这样才能启动操作。必须将 CS# 返回到高电平状态，才能启动一个新操作。
2. CA[23:16] 被捕捉后，将立即开始读取闪存阵列。
3. 读取延迟是由易失性配置寄存器（或非易失性配置寄存器）内的读取延迟值指定的。
4. 在该读取操作示例中，延迟计数值被设置为五个时钟周期。

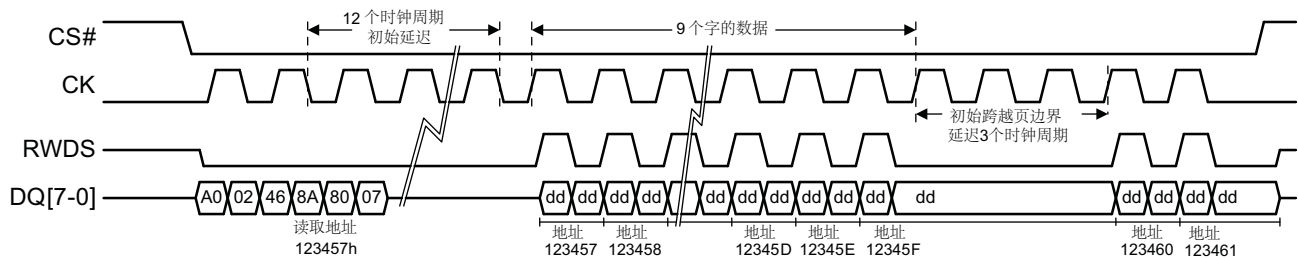
延迟代码选项相应的最大工作频率

延迟代码	延迟时钟	最大工作频率 (MHz)
0000	5	52
0001	6	62
0010	7	72
0011	8	83
0100	9	93
0101	10	104
0110	11	114
0111	12	125
1000	13	135
1001	14	145
1010	15	156
1011	16	166
1100	保留	无
1101	保留	无
1110	保留	无
1111	保留	无

注意:

1. 器件出厂时，默认的 NVCR 延迟被设置为 16 个时钟周期。
2. 装载到（非）易失性配置寄存器位 xVCR[7:4] 内的数值便为延迟代码。
3. 假设最大的操作频率适用的器件的 $t_{ACC} = 96 \text{ ns}$ 。

图 4.2 超出页界限的读取操作



注意:

1. 读取操作开始于器件地址 123457h。
2. 加载到配置寄存器内的延迟代码为 0111b，它会引起 12 个时钟周期的延迟。
3. 在这种情况下，超出页界限需要三个时钟周期（12 个时钟初始延迟 - 初始数据的 9 个时钟周期（字））
4. 未显示 CK#，但它是 CK 信号的补充内容。
5. CA45 = 1 表示线性读取突发。

线性读操作跨越第一个页边界时（延迟计数为 11 个时钟周期）

目标地址	时钟周期																													
	0	1	2	3	...	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30						
0	CA0	CA1	CA2	总线转换 + 初始延迟						D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17			
1										D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18			
2										D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19			
3										D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20			
4										D4	D6	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21			
5										D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22			
6										D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	X	D16	D17	D18	D19	D20	D21	D22			
7										D7	D8	D9	D10	D11	D12	D13	D14	D15	X	X	D16	D17	D18	D19	D20	D21	D22			
8										D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	D24	D25			
9										D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	D24	D25	D26			
10										D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	D24	D25	D26	D27			
11										D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	D24	D25	D26	D27	D28			
12										D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	D24	D25	D26	D27	D28	D29			
13										D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	D24	D25	D26	D27	D28	D29	D30			
14										D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	X	D24	D25	D26	D27	D28	D29	D30			
15										D15	D16	D17	D18	D19	D20	D21	D22	D23	X	X	D24	D25	D26	D27	D28	D29	D30			
16										D16	D17	D18	D19	D20	D21	D22	D23	D24	D25	D26	D27	D28	D29	D30	D31	D32	D33			
	—	—	1	2	...	11	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—				
	延迟计数																													

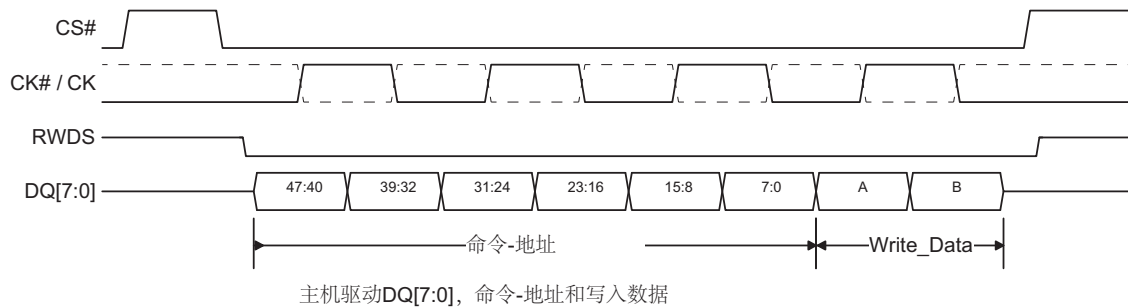
线性读操作第一次跨越页边界时（延迟计数为 16 个时钟周期）

目标地址	CS# 为低电平后的时钟周期																																
	0	1	2	3	...	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35									
0	CA0	CA1	CA2	总线转换 + 初始延迟			D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17									
1							D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	X	D16	D17									
2							D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	X	X	D16	D17									
3							D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	X	X	X	D16	D17									
4							D4	D6	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	X	X	X	X	D16	D17									
5							D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	X	X	X	X	X	D16	D17									
6							D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	X	X	X	X	X	X	D16	D17									
7							D7	D8	D9	D10	D11	D12	D13	D14	D15	X	X	X	X	X	X	X	D16	D17									
8							D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	D24	D25									
9							D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	X	D24	D25									
10							D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	X	X	D24	D25									
11							D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	X	X	X	D24	D25									
12							D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	X	X	X	X	D24	D25									
13							D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	X	X	X	X	X	D24	D25									
14							D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	X	X	X	X	X	X	D24	D25									
15							D15	D16	D17	D18	D19	D20	D21	D22	D23	X	X	X	X	X	X	X	D24	D25									
16							D16	D17	D18	D19	D20	D21	D22	D23	D24	D25	D26	D27	D28	D29	D30	D31	D32	D33									
	—	—	1	2	...	16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—									
	延迟计数																																

4.3 写入操作

写入操作的最前三个时钟周期用于提供表示操作特性的指令 / 地址信息。由于 HyperFlash 器件仅支持单个 16 位的写入操作或者连续的线性写入突发（仅在字编程指令操作期间进行数据加载时被支持），因此突发类型位 CA[45] 是‘无需关注’的。紧随 CA 信息，主机将数据传输到 DQ 总线上。数据的第一个字节（A）在时钟的上升沿被传输，第二个字节（B）在时钟的下降沿上被传输。写入数据中心是与 CK/CK# 沿对齐的。CK 为低电平、CK# 为高电平时，可随时通过将 CS# 置高来停止写入操作。

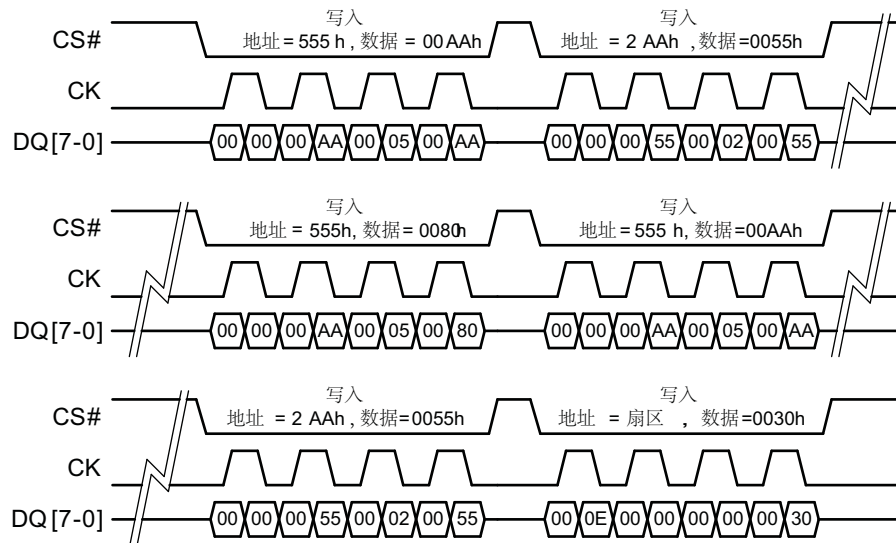
图 4.3 写入操作



注意：

1. 必须将 CK 设置为低电平，并将 CK# 设置为高电平，这样才能启动操作。必须将 CS# 返回到高电平状态，这样才能启动一个新的操作。
2. 一旦 CS# 为低电平状态，RWDS 将被驱动为低。
3. 写入操作被限为单字（16 位）传输或者一个线性写入突发（仅在字编程指令操作期间加载数据时被支持）。

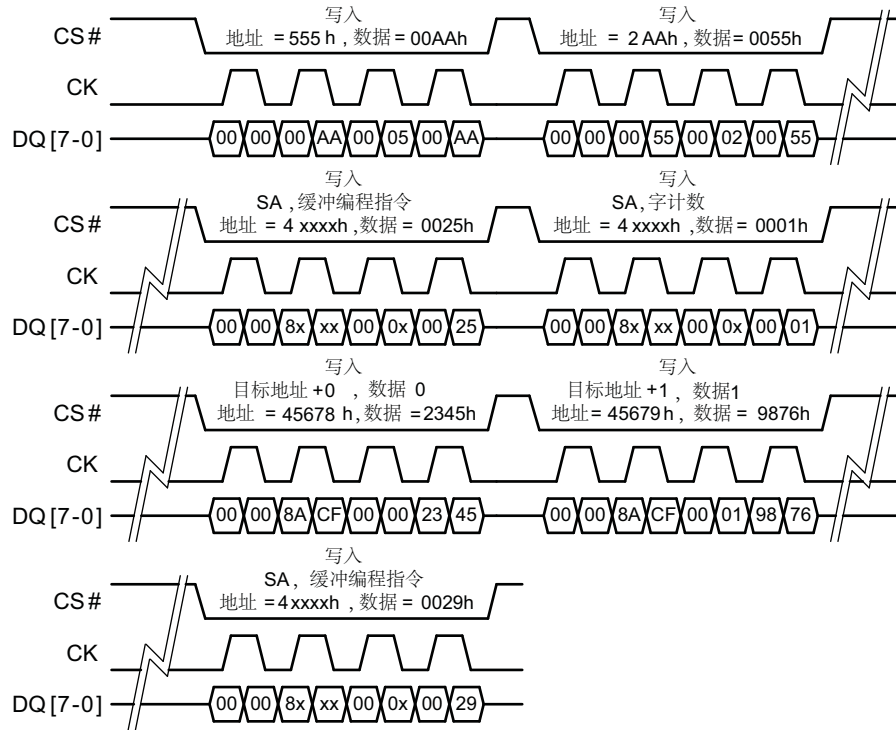
图 4.4 写入操作的实用示例：擦除操作指令序列



注意：

1. 请参考第 28 页上的图 6.4，扇区擦除操作，了解擦除操作指令的流程图。
2. 扇区的擦除操作从地址 0700000h 处开始。
3. 未显示 CK#，但它是 CK 信号的补充内容。
4. RWDS 不被显示，并不使用于写入操作期间。

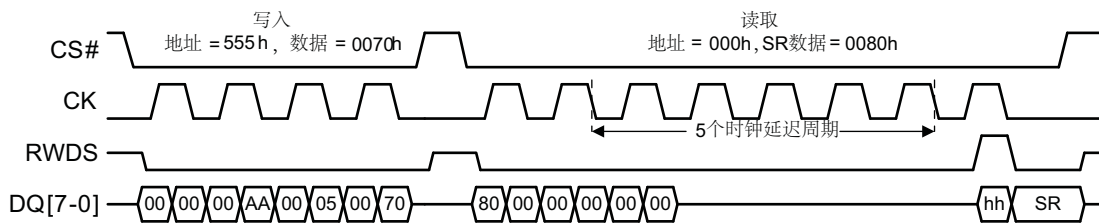
图 4.5 写入操作的实用示例：写入缓冲区编程指令序列



注意:

1. 请参考第 24 页上的图 6.2，使用状态寄存器验证写入缓冲区编程操作，了解写入缓冲区编程操作指令的流程图。
2. 分别将 2345h 和 9876h 编程到地址 45678h 和 45679h 上。
3. 未显示 CK#，但它是 CK 信号的补充内容。
4. RWDS 不被显示，并不使用于写入操作期间。

图 4.6 状态读取操作示例



注意:

1. 未显示 CK#，但它是 CK 信号的补充内容。

软件接口

5. 地址空间映射

虽然 HyperBus 协议是提供给具有存储器和寄存器地址空间的从设备的，但本规格中所介绍的 HyperFlash 存储器并没有将存储器和寄存器区分为单独的地址空间。任意数据操作都特定针对单个地址空间，不管 HyperBus 操作指示的目标位置是在选定器件的存储器空间还是寄存器空间内，该地址都不会发生改变。

写入操作始终将操作地址和数据放置到一个指令寄存器组中（缓冲区）。

读取操作可读回存储器阵列或寄存器地址空间窗口中单个地址空间范围内由各指令的执行暂时覆盖的数据。采用寄存器空间覆盖层方法的单个地址范围与传统的并行 NOR 闪存存储器和擦除软件驱动器向后兼容。

在闪存存储器器件的地址范围内，可能出现多个单独的地址空间。在任意给定时间，只有一个地址空间可见（进入）。

- 闪存存储器阵列：主非易失性存储阵列，用于存储可通过读取操作进行随机访问的数据。
- ID/CFI 区域：用于存储 Spansion 工厂预编程的器件特性信息。该区域包含器件标识（ID）和通用闪存接口（CFI）信息表。
- 安全硅区域（SSR）：1024 字节大小的一次性可编程非易失性存储阵列用于存储 Spansion 工厂预编程的永久性数据以及客户可编程的永久性数据。
- 持久保护位（PPB）：非易失性存储器阵列（每扇区一位）。编程后，每一位都会保护相关的扇区，以防止擦除和编程。
- PPB 锁定位：它是一个易失性寄存器位，用于使能或禁用对 PPB 位进行的编程和擦除。
- 密码：它是一个 OTP 非易失性阵列，可存储 64 位密码，用于允许在使用密码模式扇区保护时更改 PPB 锁定位的状态。
- 动态保护位（DYB）：它是易失性阵列（每扇区一位）。设置该位后，每一位都会保护相关的扇区，以防止进行擦除和编程。
- 状态或外设寄存器：寄存器访问，用于显示嵌入式算法状态并读取或写入其他寄存器。

闪存存储器阵列是主要同时也是默认的地址空间，但随时会被另一个地址空间覆盖掉。每个备用地址空间都被称为地址空间重叠（ASO）。

每个 ASO 将根据 ASO 进入指令来替换（覆盖）由进入 ASO 的指令所选定的扇区或者整个闪存器件空间。如果 ASO 仅覆盖了一个扇区，那么存储器阵列中的其他扇区仍保持为可读状态。没有被特定 ASO 地址映射定义的地址范围将留作将来使用。除非有另外说明，负责 ASO 地址映射之外的所有读取访问均返回无效（未定义）的数据。读取地址映射之外的位置取得的数据，无论是 1 还是 0，都未定义其含义。

通过各种地址映射模式，可以确定闪存器件地址空间在任何特定时刻的内容：

- 读取模式
- 状态寄存器（SR）模式
- 地址空间重叠（ASO）模式
- 外设寄存器模式

在读取模式下，主机系统存储控制器可能直接读取整个闪存阵列。存储器件的嵌入式算法控制器（EAC）在加电期间、硬件复位后、指令复位后、以及嵌入式算法（EA）挂起后会将器件置于读取模式。在读取模式下，设备可接受读取访问和指令。当 EA 挂起时，器件可在读取模式下接受部分指令。

在任一模式下，都可以发出状态寄存器读取命令，从而在设备地址空间中的每个字地址出现状态寄存器 ASO。在此状态寄存器 ASO 模式下，设备接口等待读取访问，写入访问被忽略。下一次设备读取访问将读取状态寄存器的内容并退出状态寄存器 ASO，然后返回到先前收到状态寄存器读取指令时的（调用）模式。

此外，读取或写入其他寄存器的指令也会同样地使用外设寄存器模式。在该模式下，寄存器会进入暂时 ASO 模式（在指令选定寄存器的读取或写入操作完成后自动退出）。读取或写入操作都是在寄存器访问指令序列中最后一个周期内实现。

在 EA 模式下，EAC 执行嵌入式算法，如编程或擦除非易失性存储阵列。在 EA 模式下，所有闪存存储器阵列都不可读。在 EA 模式下，只接受编程 / 擦除挂起指令或状态寄存器读取指令。所有其他指令均被忽略。因此，无法从 EA 模式进入其他 ASO。

在 ASO 模式下，进入其中一个剩余的重叠地址空间（覆盖主闪存阵列地址映射）。在任意特定时间内，只能进入一个 ASO。对器件执行的指令对当前进入的 ASO 产生影响。每个 ASO 都有自己特定的有效指令。它们被罗列在第 51 页上的指令定义表中每个 ASO 的相关章节内。

下列各 ASO 包含了非易失性数据，可通过编程将 1 改为 0：

- 安全硅区域
- ASP 配置寄存器（ASPR）
- 持久保护位（PPB）
- 密码
- 只有 PPB ASO 中包含非易失性数据，并可通过擦除该数据将 0 改为 1。

在进入其中一个非易失性 ASO 后发出编程或擦除指令时，EA 对该 ASO 执行操作。当 EA 处于活动状态时，该 ASO 不可读。EA 完成后，ASO 仍留在进入状态，但重新变成可读。在任意 ASO 的 EA 过程中，挂起和恢复指令无效。

可通过使用外设寄存器模式管理下列器件：上电复位定时器、中断配置寄存器、中断状态寄存器、易失性配置寄存器以及非易失性配置寄存器。

5.1 闪存存储器阵列

S26KL/S26KS 系列器件具有统一的扇区架构（扇区容量为 256 kB）。S26KL512S、S26KS512S 扇区和存储器地址映射表到 S26KL128S、S26KS128S 扇区和存储器地址映射表显示的是三种器件的扇区架构。

通过用户配置选项，可使用 8 个 4 kB 的参数扇区来覆盖第一个扇区（SA00）或最后一个扇区（SAmax）。参数扇区地址映射显示的是最低或最高扇区的分段情况，如主阵列扇区 0 被八个 4 kB 大小的参数扇区覆盖表和最后一个扇区被八个 4 kB 大小的参数扇区覆盖表所示。可通过使用针对相应参数扇区地址的标准擦除和编程指令序列，并按照正常的方式对参数扇区进行擦除和编程。注意：更小的参数扇区需要添加 A[16:11]，将其作为地址的一部分，用于确定擦除和编程指令序列时的目标参数扇区。可通过非易失性配置寄存器来配置第一个或最后一个统一的扇区，使其包含参数扇区。

注意：为了在单个页面上显示整个设备的扇区相关信息，将该表进行了简化。对于未明确列出的扇区以及它们的地址范围（如 SA001-SA510），其扇区起始地址和结束地址与该尺寸的所有其他扇区具有相同的组合。例如，所有 256 kB 扇区的组合均为 XX00000h-XX1FFFFh。

S26KL512S、S26KS512S 扇区和存储器地址映射

扇区容量（kbyte）	扇区数量	扇区范围	地址范围（16 位）	注意
256	256	SA00	0000000h – 001FFFFh	扇区起始地址
		:	:	—
		SA255	1FE0000h – 1FFFFFFh	扇区结束地址

S26KL256S、S26KS256S 扇区和存储器地址映射

扇区容量（kbyte）	扇区数量	扇区范围	地址范围（16 位）	注意
256	128	SA00	0000000h – 001FFFFh	扇区起始地址
		:	:	—
		SA127	0FE0000h – 0FFFFFFh	扇区结束地址

S26KL128S、S26KS128S 扇区和存储器地址映射

扇区容量（kbyte）	扇区数量	扇区范围	地址范围（16 位）	注意
256	64	SA00	0000000h – 001FFFFh	扇区起始地址
		:	:	—
		SA63	07E0000h – 07FFFFFFh	扇区结束地址

主阵列扇区 0 被八个 4 kB 大小的参数扇区覆盖

主阵列扇区容量	参数扇区号	地址容量	地址范围 (16 位)	注意
256 kB	0	4 kB	0000000h – 001FFFFh	参数扇区 0 的开头
	1	4 kB	0000800h – 0000FFFh	参数扇区 1
	2	4 kB	0001000h – 00017FFh	参数扇区 2
	3	4 kB	0001800h – 0001FFFh	参数扇区 3
	4	4 kB	0002000h – 00027FFh	参数扇区 4
	5	4 kB	0002800h – 0002FFFh	参数扇区 5
	6	4 kB	0003000h – 00037FFh	参数扇区 6
	7	4 kB	0003800h – 0003FFFh	参数扇区 7 的结束
	主阵列扇区 0 的显示部分	224 kB	0004000h – 001FFFFh	映射到主阵列扇区 0 中显示的部分

最后一个扇区被八个 4 kB 大小的参数扇区覆盖

主阵列扇区容量	参数扇区号	地址容量	地址范围 (16 位)	注意
256 kB	主阵列中最后扇区显示的部分	224 kB	1FF0000h – 1FFBFFFh	映射到主阵列最后扇区中显示的部分
	0	4 kB	1FFC000h – 1FFC7FFh	参数扇区 0 的开头
	1	4 kB	1FFC800h – 1FFCFFFh	参数扇区 1
	2	4 kB	1FFD000h – 1FFD7FFh	参数扇区 2
	3	4 kB	1FFD800h – 1FFDFFFh	参数扇区 3
	4	4 kB	1FFE000h – 1FFE7FFh	参数扇区 4
	5	4 kB	1FFE800h – 1FEFFFFh	参数扇区 5
	6	4 kB	1FFF000h – 1FFF7FFh	参数扇区 6
	7	4 kB	1FE0000h – 1FFFFFFh	参数扇区 7 的结束

注意:

- 将参数扇区映射到最高扇区内时，其上面序列地址位取决于器件的容量。最后一个扇区被八个 4 kB 大小的参数扇区覆盖表 使用了 512 Mbit 的器件地址映射。
- 最后扇区的扇区号分别为 255 (对应 512 Mb 的容量)、127 (对应 256 Mb 的容量) 和 63 (对应 128 Mb 的容量)。
- 擦除或编程某个参数扇区时，要在指令序列中指定的扇区地址为 A[max:11]。

5.2 设备 ID 和 CFI (ID-CFI) ASO

系统可以通过两种传统方法识别系统中已经安装的闪存类型。一种被传统定义为“自动选择”，现在指的是器件标识 (ID)。另一种方法称为通用闪存接口 (CFI)。

对于 ID，使用一条指令来启用一个地址空间重叠，最多可从这个空间内读取 16 字位置，用于从闪存中获取 JEDEC 制造商标识 (ID)、器件 ID 和一些配置及保护状态信息。系统可以使用制造商和器件 ID 为闪存设备选择相应的驱动程序软件。

CFI 也使用了一条指令来启用一个地址空间重叠，从该空间中读取关于闪存组织和操作的标准信息的详细表。通过该方法，在编写驱动程序软件时，便不用再熟知每种可能的存储设备细节。在编写驱动程序软件时，只需根据 CFI 表中的信息来调整驱动程序的运行方式即可，无需按照常规方式来处理多种不同的设备。

传统上，这两个地址空间为不同的重叠，分别使用单独的指令。不过，这两个地址空间的映射是非重叠的，因此可以组合成一个地址空间并一同出现在一个重叠中。进入“自动选择” (ID) 或 CFI 映射的任意一条传统指令都可以进入现在组合的 ID-CFI 地址映射。

ID-CFI 地址映射出现在 ID-CFI 进入指令使用的地址所选择的扇区中，并覆盖掉该扇区中的闪存阵列数据。进入 ID-CFI ASO 后，所有其他扇区的内容变成未定义。

ID-CFI 地址映射从所选扇区的 0 位置开始。对于从所定义 ID-CFI ASO 最大地址到所选扇区最大地址之间的位置，其数据未被定义。ID-CFI 进入指令使用与前一代存储器相同的地址和数据值分别用来读取 JEDEC 制造商 ID (自动选择) 和通用闪存接口 (CFI) 信息。

ID-CFI 地址映射概览

字地址	说明	读取 / 写入
(SA) + 0000h 到 000Fh	器件 ID (传统的“自动选择”值)	只读
(SA) + 0010h 到 0079h	CFI 数据结构	只读
(SA) + 007Ah 到 00FFh	未定义	只读

有关完整的地址映射信息，请参见第 57 页上的 ID (自动选择) 地址映射表。

5.2.1 器件 ID

美国电子工程设计发展联合会 (JEDEC) 标准 JEP106T 定义了兼容存储器的制造商 ID。通用行业用法定义了从存储器件读取制造商 ID 和器件特定 ID 的方法和格式。制造商和器件 ID 信息的主要为了使编程装置自动匹配器件和相应的编程算法。Spansion 在此 32 字节地址空间中增加了更多字段。

初始行业格式的结构适合任一内存数据总线宽度，如 x8、x16、x32。传统上 ID 代码值为字节宽度，但位于总线带宽地址边界。因此，器件地址输入递增时，将读取连续的字节、字或双字位置，并且 ID 代码始终位于数据总线上最低有效字节的位置。由于设备数据总线的带宽为字，因此每个代码字节分别位于每个字的低位字节。在初始行业格式中，高字节总是为 0。Spansion 修改了该格式，在地址空间的一些字中两个字节都得到了使用。有关器件 ID 地址映射的详细说明，请参见第 57 页上的 ID (自动选择) 地址映射表。

5.2.2 通用闪存接口（CFI）

JEDEC 通用闪存结构（CFI）规范（JESD68.01）定义了可从闪存设备读取的标准化数据结构，它允许厂商在整个设备系列中使用其指定的软件算法。数据结构包含系统配置信息，如各种电气和时序参数以及器件支持的特殊功能。这样，软件支持就变得与器件和器件 ID 无关，并且对整个闪存器件系列向前向后兼容。

系统可以从选定扇区中各个地址读取 CFI 信息，如第 57 页上的 9.3 章节，[器件 ID 和通用闪存接口（ID-CFI）ASO 映射](#)所示。

与设备 ID 信息相似，CFI 信息的结构也适合任一内存数据总线宽度，如 x8、x16、x32。代码值总是字节宽度，但位于数据总线带宽地址边界。因此，设备地址递增时，将读取连续的字节、字或双字位置，并且代码始终位于数据总线上最低有效字节的位置。由于数据总线为字宽度，因此每个代码字节分别位于每个字的低位字节，并且高位字节总为 0。

更多有关信息，请参考 *Spansion CFI 规格*，版本 1.4（或更高版本）以及 JEDEC 出版物 JEP137-A 和 JESD68.01。欲了解 JEDEC 标准，请访问该组织的网站：<http://www.jedec.org>。如要了解 Spansion CFI 规范，请访问 Spansion 公司网站：<http://www.spansion.com/Support/TechnicalDocuments/Pages/ApplicationNotes.aspx>（在发布本文档时提供），或者联系本公司网站上所提供的本地 Spansion 经销商。

6. 嵌入式操作

6.1 嵌入式算法控制器 (EAC)

EAC 从主机系统接收关于编程和擦除闪存阵列的指令，并执行更改非易失性存储器状态所需要的全部复杂操作。这样可以减少主机管理编程和擦除进程的负担。

EAC 操作有五种类别：

- 深度掉电模式
- 待机（读取模式）
- 地址空间切换
- 嵌入式算法（EA）
- 高级扇区保护（ASP）管理

6.1.1 深度掉电模式

在深度掉电（DPD）模式下，消耗的电流最低。器件处于空闲状态（并非处于 ASO 状态）时，必须进入 DPD 模式。此外，器件在擦除挂起或编程挂起状态时，也可以进入 DPD 模式。

6.1.2 EAC 待机

在待机模式下，电流消耗明显降低。当没有执行任何指令和嵌入式算法时，EAC 将进入待机模式。如果在嵌入式算法期间取消了选择器件（CE# = 高电平），器件仍会消耗工作电流，直至操作完成为止（I_{CC3}）。第 63 页上的 11.2 章节，*直流特性（与 CMOS 兼容）*中的 I_{CC4} 表明主机接口和 EAC 处于其待机状态时的待机电流规范。

6.1.3 地址空间切换

写入特定地址和数据序列（指令序列）可将存储器件地址空间从闪存存取阵列切换到其中某个地址空间重叠（ASO）。

嵌入式算法对当前活动（进入的）ASO 中的可见信息进行操作。在系统发出 ASO 退出指令、执行硬件复位前，或者在断开器件电源前，系统将一直能够存取 ASO。ASO 退出指令可将 ASO 切换回闪存存储阵列地址空间。进入特定 ASO 后接受的指令在指令定义表中 ASO 进入和退出指令之间被列出。有关所有指令序列的地址和数据要求，请参见第 51 页上的 7.1 章节，*指令汇总*。

6.1.4 嵌入式算法（EA）

更改存储阵列中的非易失性数据时，需要执行复杂的操作序列，这些操作被称为嵌入式算法（EA）。这些算法完全由器件内部的嵌入式算法控制器（EAC）管理。主算法执行对主阵列数据和 ASO 的编程和擦除操作。主机系统将指令代码写入到闪存器件地址空间内。EAC 接收指令用于执行所有必要的步骤以完成指令，并在 EA 执行期间提供状态信息。

6.2 编程和擦除简介

闪存数据位以大组（称为扇区）的形式被并行擦除。擦除操作将扇区中的每一个数据位都设置为逻辑 1 状态（高电平）。每个闪存数据位可以分别进行编程，使其从擦除 1 状态变为编程逻辑 0 状态（低电平）。为 0 的数据位不能通过编程返回 1。成功完成编程时，数据位仍然为 0。只有擦除操作才能将 0 变换为 1。对同一个字位置多次编程不同的 0 位，会导致原数据和正在编程的新数据进行逻辑 AND。

第 55 页上的 8. 章节，*嵌入式算法性能*介绍了编程和擦除时长。

编程和擦除操作可能被挂起。

- 擦除操作可能被挂起，以便能够在擦除操作期间编程或读取另一个扇区（不是在擦除扇区中）。在擦除挂起期间，不能启动其他擦除操作。
- 编程操作可能被挂起，以便能够读取另一个位置（并非位于正在编程的行中）。
- 在编程操作挂起期间，不能启动其他编程或擦除操作；在该时间内，编程或擦除指令将被忽略。
- 在完成嵌入的编程操作或读取访问后，可以恢复被挂起的擦除或编程操作。

- 编程和擦除操作可以根据需要被中断，但为了正常完成编程或擦除操作，恢复和下一次挂起指令之间的时长必须大于或等于第 55 页上的[嵌入式算法性能](#)中所介绍的 t_{PRS} 或 t_{ERS} 。
- 完成嵌入式算法（EA）时，EAC 返回至 EA 启动前的操作状态和地址空间（擦除挂起或 EAC 待机）。

系统可以通过读取状态寄存器（第 36 页上的[状态寄存器](#)）来确定编程或擦除操作的状态。

除了编程挂起和状态读取指令外，在嵌入式编程算法期间写入设备的所有指令都将被忽略。

除了擦除挂起和状态读取指令外，在嵌入式擦除算法期间写入器件的所有指令都将被忽略。

硬件复位会立即终止正在执行的所有编程 / 擦除操作，并在经过 t_{RPH} 时间后返回到读取模式。一旦器件返回待机状态，应重新启动被终止的操作，以保证数据的完整性。

出于性能和可靠性考虑，使用 16 字节地址范围在 16 字节半个页上进行内部编程。

第 63 页上的[直流特性](#)（与 CMOS 兼容）中的 I_{CC3} 显示了写入（嵌入式算法）操作的工作电流规范。

6.2.1 编程粒度

S26KL/S26KS 支持两种编程方法：字或写入缓冲区编程。

字编程检查指令中提供的数据字，在寻址的存储阵列字中编程 0，以匹配指令数据字中的 0。

写入缓冲区编程检查写入缓冲区，在寻址的存储阵列行中编程 0，以匹配写入缓冲区中的 0。写入缓冲区不需要全部填入数据。在一个编程操作中，可以尽可能少地编程，如单个位、多个位、单个字、多个字、半个页、多个半页或者整个缓冲器。使用写入缓冲区方法可以减轻主机系统在写入编程指令方面的负担，并能够减轻存储器件在编程操作方面的内部负担。与使用字编程指令对各个字进行编程相比，写入缓冲区编程更快、更有效率。

每半页可以使用任何一种方法进行编程。用不同方法编程的半个页可以在同一行内混合。

可以在一个半页中对字和写缓冲区进行多次编程，这样能够与传统软件兼容。但为了保证数据的完整性最佳，建议在同一个半页中只进行一次字编程和写入缓冲区编程。对于要求在同一半页中进行多次编程的应用，建议您添加系统软件错误检测与纠正功能，从而提高经过多次编程的半页中数据的完整性。

HyperFlash 芯片工艺后代产品可能不再支持在同半个页内进行多次编程，但并不包含该半个页的扇区中进行擦除。计划后代产品的软件移植能够：适用数据结构和数据管理方法，从而使每次擦除操作在每半页中只允许一次编程。

6.2.2 增量编程

同一字位置或半个页可以通过字或写入缓冲区编程方法进行多次编程，以增量方式将 1 改为 0。但是，增量编程对数据完整性产生影响，如第 21 页上的 6.2.1 章节，[编程粒度](#)所述。建议您添加系统软件错误检测与纠正功能，从而提高经过多次编程的半页中数据的完整性。

6.2.3 编程方法

6.2.3.1 字编程

字编程用于编程闪存存储阵列中任意位置上的单个字或多个字。

最小的字编程指令序列需要进行四个指令写入操作。为启动编程指令序列，先发出两个解锁指令写入操作（第一和第二操作），然后是编程设置指令（第三个操作）。接下来要写入编程地址和数据（第四个操作），以启动嵌入式编程算法。系统不需要提供进一步控制或时序。器件自动生成字编程脉冲，并内部检验编程的网格边距。当完成嵌入式编程算法时，EAC 随后会返回到它的待机模式。

前面所述的四个传输操作字编程指令序列均用于编程单个（16 位）字。利用突发写入功能，可以使用字编程序列写入多个连续字。单个字编程序列的解锁和编程指令序列相同。在数据 / 地址传输期间，一次确认 CS# 可加载多个连续数据值。出现的数据被编程到连续地址内，并从突发写入操作的指令地址阶段中所指定的目标地址开始写入。只要不超过对齐的 256 字地址边界，最多可编程 256 个字。在字编程序列期间使用突发写入加载多个字时，必须要降低时钟速率，如第 69 页上的图 12.1，[字编程指令期间多字加载的突发写入时序图](#)中所述。

系统可以通过读取状态寄存器（第 36 页上的[状态寄存器](#)）来确定编程操作的状态。

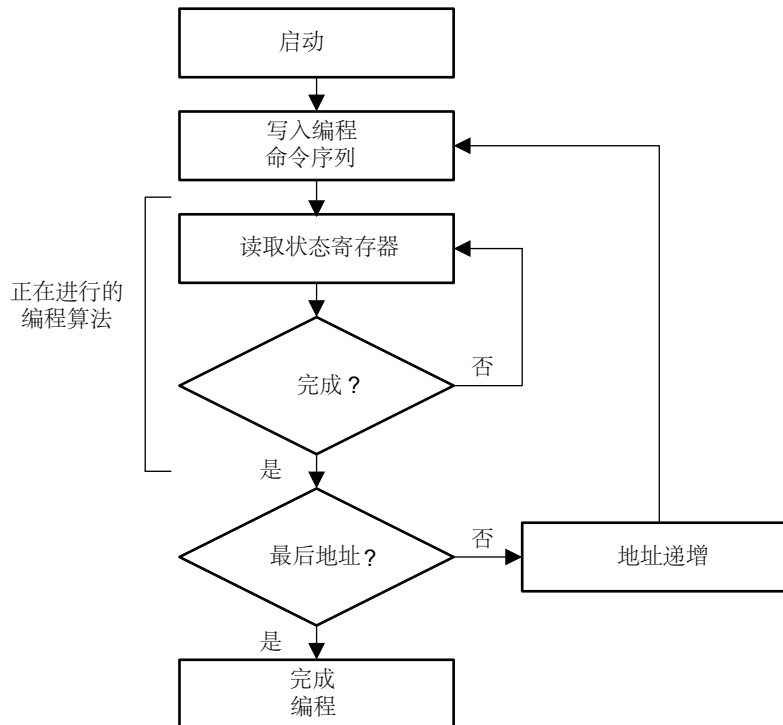
除了编程挂起和状态寄存器赌气指令外，在嵌入式编程算法期间写入器件的所有指令都将被忽略。

请注意，硬件复位（RESET# = V_{IL}）或断电可立即终止编程操作，并经过 t_{RPH} 时间后可使器件返回到读取模式。终止操作会使正在编程的区域处于中间状态，即包含无效或不稳定的数据。一旦器件完成硬件复位操作，使用相同的数据重新启动编程指令序列，从而完成编程操作，这样可确保数据被完全编程。但是，为了确保最佳的数据完整性，必须擦除或重新编程被终止的编程操作所在的扇区。

字编程指令也可以在进入 SSR ASO 后使用。

修改版的字编程指令没有解锁写入周期，从而能够在进入 ASP 配置寄存器（ASPR）、密码和 PPB ASO 后进行编程。进入 PPB 锁定和 DYB ASO 后，可以使用相同的指令更改易失性位。欲了解编程指令序列，请参见第 51 页上的[指令定义表](#)。

图 6.1 字编程操作



6.2.3.2 写入缓冲区编程

写入缓冲区用于在 512 字节边界（行）上排列的 512 字节地址范围内编程数据。因此，整个写入缓冲区编程操作必须与行边界对齐。不足 512 字节的编程操作可以在任意字边界上开始，但不能跨越行边界。在写入缓冲区编程操作开始时，缓冲器中所有位的位置全部为 1（FFFFh 字），因此未加载的位置仍保持现有数据。有关地质映射的信息，请参见第 4 页上的产品概述。

写入缓冲区编程在一个操作中最多可编程 512 字节。在每个写入缓冲区编程操作中，可以编程 1 位到 512 字节。强烈建议写入多个半页，每个半页只写入一次。为达到最佳性能，编程应在 512 字节边界对齐的全部 512 字节行上完成。

仅在闪存存储器阵列或 SSR ASO 中支持写入缓冲区编程。

写入缓冲区编程操作由前两个写入解锁周期来启动。随之是写入到缓冲器指令的第三个写入周期，其中包含要编程的扇区地址（SA）。接下来，系统写入字位置数量减 1。这会告知器件有多少写入缓冲区地址加载了数据以及何时编程缓冲器会发出确认指令。写入到缓冲器指令和写入字计数指令中的扇区地址必须匹配。要编程的扇区必须解锁（不受保护）。如果尝试编程一个锁定扇区，编程操作会被中止，同时状态寄存器会指示操作失败（参见第 37 页上的状态寄存器表）。

系统随后写入起始地址和数据字。这个起始地址是要编程的第一个地址和数据对，它用来选择写入缓冲区行上的起始字地址。扇区地址必须与通过写入到缓冲器指令进行编程的扇区地址匹配，否则操作将中止并返回初始状态。所有后续单字地址和数据对的写入操作必须是连续序列。所有写入缓冲区地址必须在同一行内。如果系统尝试加载此范围之外的数据，操作将中止并返回初始状态。

每个数据加载操作后，计数器递减。请注意，随着数据写入倒计时，每个写入操作都被认为数据正在加载到写入缓冲区。在写入缓冲区加载期间，不能执行任何指令。唯一可停止写入缓冲区加载的方式是向编程操作行之外的地址写入。这个无效地址将立即中止写入到缓冲器指令，并设置写入缓冲区中止状态位（SR[3]）。

一旦已加载指定数量的写入缓冲区地址，系统必须随后在扇区地址内写入编程缓冲器到闪存指令。器件随之开始忙碌。嵌入式编程算法自动编程数据，并验证数据是否为正确的数据组合。在这些操作期间，系统不需要提供任何控制或时序。如果加载的写入缓冲区位置的数量不正确，操作将中止并返回初始状态。由于在字计数结束时需要写入编程缓冲器到闪存指令，所以如果此时写入其他任何指令，编程操作会中止。

可以通过编程挂起指令来暂停写入缓冲区嵌入式编程操作。当嵌入式编程算法完成时，EAC 随后返回至编程操作启动时的 EAC 待机或擦除挂起待机状态。

系统可以通过读取状态寄存器来确定编程操作的状态（第 37 页上的状态寄存器表）。欲了解编程操作的流程图，请参见第 24 页上的图 6.2，使用状态寄存器验证写入缓冲区编程操作。

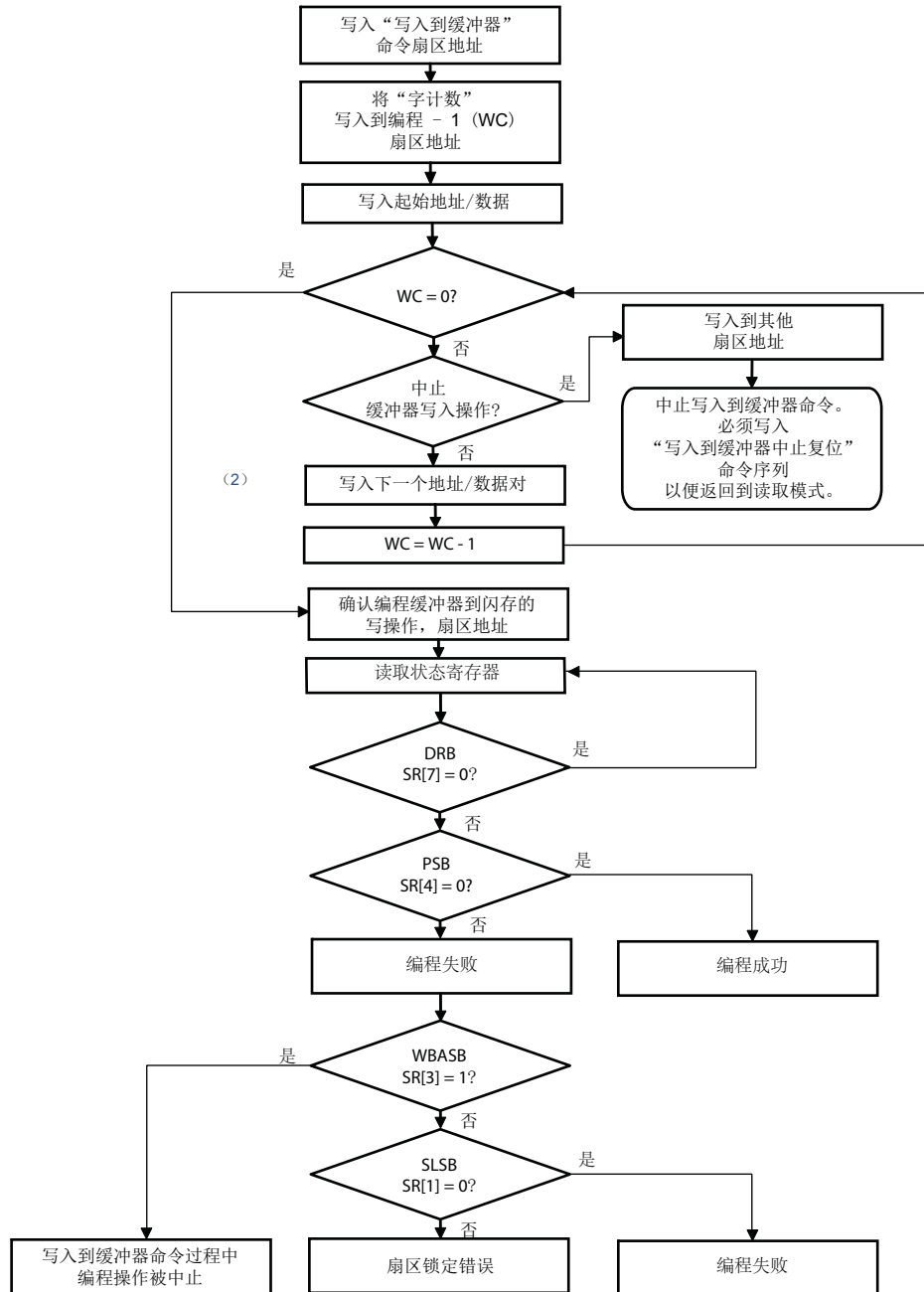
在下列情况下，写入缓冲区编程序列将中止：

- 加载的字计数值超过缓冲器容量（255）。
- 写入的地址超出在写入到缓冲器指令中提供的行。
- 加载完写入字计数的数据字数量后不发出写入缓冲区到闪存指令。

当导致写入缓冲区指令中止的任何条件发生时，指令操作将立即中止，并在状态寄存器的位置位 4 指明编程失败（PSB = 1），因为写入缓冲区中止位置位 3 被设置（WBASB = 1）。下一次成功的编程操作将清除失败状态，或者可以使用清除状态寄存器指令清除 PSB 状态位。

有两种方法可终止写入缓冲区编程序列：硬件复位或关开电源。不过，使用任一方法都可能造成正在编程的区域处于中间状态，即包含无效或不稳定的数据值。在此情况下，需要使用相同的数据对该区域进行重新编程或者擦除该区域，以确保正确编程数据值或适当擦除它们。为了确保最佳的数据完整性，必须擦除或重新编程被终止的编程操作所在的扇区。

图 6.2 使用状态寄存器验证写入缓冲区编程操作



注意：

- 欲了解写入缓冲区编程所需的指令序列，请参见第 51 页上的指令定义表。
- 当指定了扇区地址时，所选扇区中的任何地址均可接受。不过，为写入缓冲区地址位置加载数据时，所有地址必须在所选行范围内。

写缓冲区编程指令序列

序列	地址	数据	注释
发出解锁指令 1	555	AA	
发出解锁指令 2	2AA	55	
在扇区地址发出写缓冲区的指令	SA	0025h	
在扇区地址发出位置数量	SA	WC	WC = 要编程的字的数量 - 1
示例: WC = 0, 则需要编程一个字 WC = 1, 则需要编程两个字			
加载起始地址 / 数据对	起始地址	PD	选择某一行, 然后加载第一个地址 / 数据对。
加载下一个地址 / 数据对	WBL	PD	所有地址 必须 位于所选行的边界内, 并且要按序列连续加载。
加载最后一个地址 / 数据对	WBL	PD	所有地址 必须 位于所选行的边界内, 并且要按序列连续加载。
在扇区地址发出写缓冲区编程确认	SA	0029h	必须 在最后一个写入缓冲区位置被加载后执行该指令, 否则操作会中止。
器件进入忙碌状态。			

图标:

SA = 扇区地址 (非扇区地址位不用考虑。扇区内的地址足够。)

WBL = 写入缓冲区位置 (必须位于起始地址指定的行的边界内)

WC = 字计数

PD = 编程数据

6.2.4 编程挂起 / 编程恢复指令

通过使用编程关起指令, 系统可以中断嵌入式编程操作, 以便能够从非挂起的行中读取数据。当在某个编程过程中写入编程挂起指令时, 器件在 t_{PSL} (编程挂起延时) 时间内暂停编程操作并更新状态位。写入编程挂起指令时, 不用考虑地址。

编程操作挂起后, 系统可以读取任何非挂起的行中的阵列数据。当擦除操作被挂起时, 仍可以在编程期间发出编程挂起指令。在此情况下, 可以从擦除挂起或编程挂起之外的地址读取数据。

写入编程恢复指令后, 器件返回编程状态, 并且状态位被更新。系统可以通过读取状态寄存器来确定编程操作的状态。欲了解这些状态位的信息, 请参见第 36 页上的**状态寄存器**。

在编程挂起期间有效的存取和指令包括:

- 读取任何其他非擦除挂起的扇区
- 读取任何其他非编程挂起的行
- 状态读取指令
- 退出 ASO 或指令集退出
- 编程恢复指令
- 加载中断配置寄存器
- 加载中断状态寄存器

系统必须写入编程恢复指令, 才能退出编程挂起模式并继续编程操作。进一步写入的编程恢复指令被忽略。在器件恢复编程后, 方可写入另一个编程挂起指令。

编程操作可以根据需要经常中断, 但为了使编程操作正常完成, 恢复和下一次挂起指令之间的时长必须大于或等于第 19 页上的**嵌入式算法控制器 (EAC)**中所述的 t_{PRS} 。

进入 ASO 后, 不支持编程挂起和恢复。在编程挂起时, 不支持进入 ASO。

6.2.5 空白检查

空白检查指令用来确认所选的闪存存储器阵列扇区是否完全被擦除。空白检查指令不允许在空白检查期间读取阵列。若在执行此指令时读取阵列，会返回未知数据。

如要对一个扇区启动空白检查，可以在 EAC 处于待机状态时向该扇区中的地址 555h 写入 33h。

当器件正在编程、擦除或挂起时，不能写入空白检查指令。

通过读取状态寄存器来确认器件是否仍在忙碌，在完成时确认扇区是否空白。状态寄存器的第 7 位指示器件是否正在执行空白检查（与擦除操作类似）。如果扇区已擦除，状态寄存器的位 5 清除为 0；如果未擦除，设为 1。

一旦发现任何位没有擦除，器件将中止操作并报告结果。

空白检查完成后，EAC 返回待机状态。

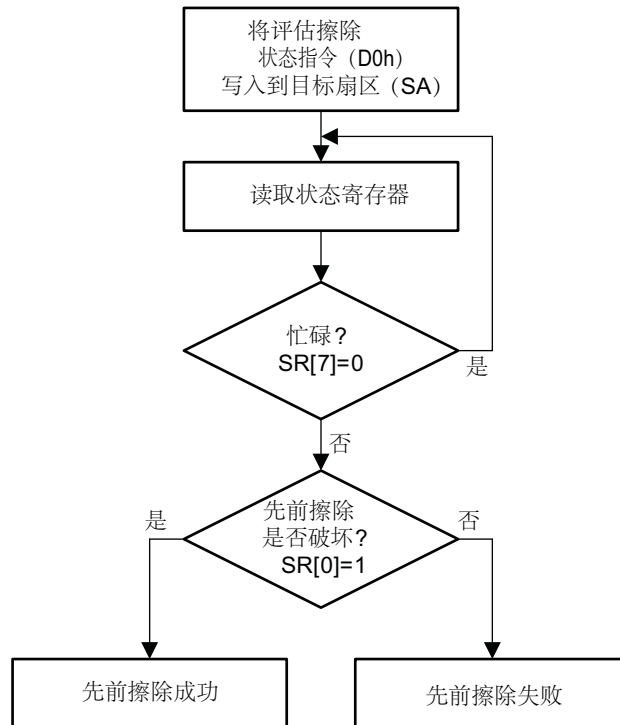
6.2.6 评估擦除状态

评估擦除状态（EES）指令用于验证对已指定地址的扇区进行的擦除操作是否完全成功。如果成功擦除了选定扇区，则状态寄存器中的扇区擦除状态位（SR[0]）被置 1；如果擦除操作未成功，则该位（SR[0]）被清 0。欲了解更多信息，请参见第 27 页上的图 6.3，[评估擦除状态的软件序列](#)。

使用 EES 指令可以检测擦除操作失败的原因，原因可能是：断电、复位或擦除操作过程失败。

EES 指令需要经过 t_{EES} 时间来完成并更新状态寄存器中扇区擦除状态位（SR[0]）。使用读取状态寄存器指令（70h）来读取状态寄存器中的器件就绪位（SR[7]），以便确定 EES 指令完成的时间。一旦状态寄存器中的器件就绪位指示器件返回到就绪（1）状态，则扇区擦除状态位（SR[0]）表明是否成功擦除了目标扇区。如果 SR[0] = 0，则表示扇区未被擦除，那么必须重新执行擦除操作，以确保该扇区中数据存储的可靠性。

图 6.3 评估擦除状态的软件序列



6.2.7 擦除方法

6.2.7.1 芯片擦除

芯片擦除功能擦除整个闪存存储器阵列。器件不要求系统在擦除之前进行预编程。嵌入式擦除算法在执行电擦除前，自动编程和验证整个存储器是否为全 0 数据组合。芯片擦除成功后，器件内的所有位置均包含 FFFFh。在这些操作期间，系统不需要提供任何控制或时序。先通过写入两个解锁周期启动芯片擦除指令序列，随后执行设置指令。另外两个解锁写入周期之后是芯片擦除指令，用于激活嵌入式擦除算法。

嵌入式擦除算法完成后，EAC 返回待机状态。请注意，正在执行嵌入式擦除操作时，系统不能从阵列中读取有效数据。系统可以通过读取状态寄存器来确定擦除操作的状态。欲了解这些状态位的信息，请参见第 36 页上的[状态寄存器](#)。芯片擦除操作开始后，只有状态读取、硬件复位或关断电源有效。所有其他指令均被忽略。不过，硬件复位或关断电源会立即终止擦除操作，并在经过 t_{RPH} 时间后返回读取模式。如果芯片擦除操作被终止，一旦器件返回待机状态，必须重新启动芯片擦除指令序列，以确保数据完整性。

受 ASP DYB 和 PPB 位保护的扇区不会被擦除。请参见第 51 页上的[软件接口参考](#)。如果一个扇区在芯片擦除期间受到保护，芯片擦除将跳过受保护的扇区，继续对下一个扇区执行擦除。如果对受保护扇区的擦除失败，状态寄存器擦除状态位和扇区锁定位将不设为 1。

6.2.7.2 扇区擦除

扇区擦除功能擦除存储阵列中的一个扇区。器件不要求系统在擦除之前进行预编程。嵌入式擦除算法在执行电擦除前，自动编程和验证整个扇区是否为全 0 数据组合。扇区擦除成功后，被擦除扇区内的所有位置均包含 FFFFh。在这些操作期间，系统不需要提供任何控制或时序。先通过写入两个解锁周期启动扇区擦除指令序列，随后执行设置指令。写入另外两个解锁周期后再写入要擦除的扇区地址，接着执行扇区擦除指令。

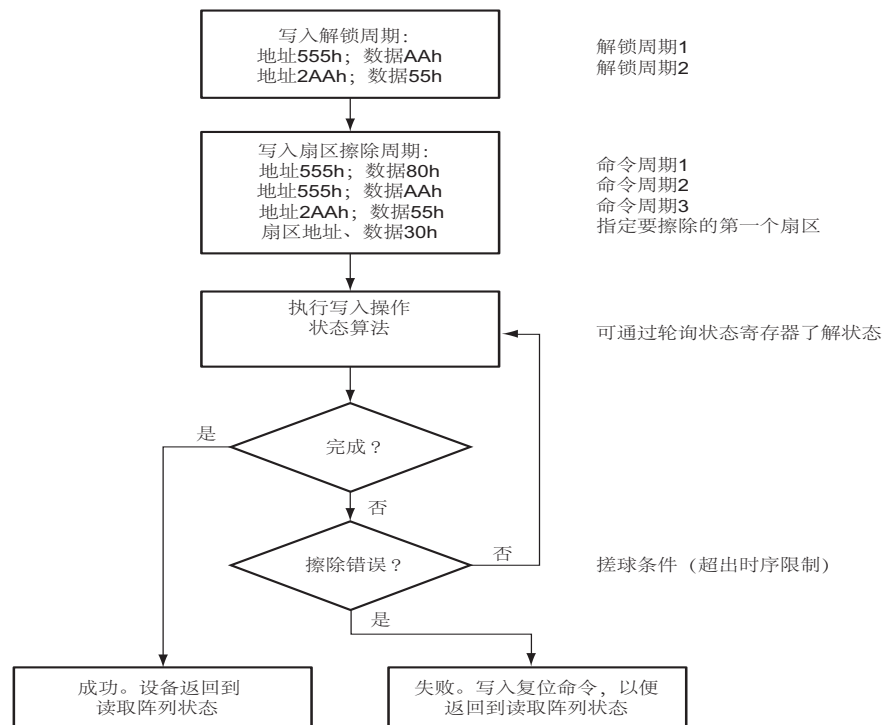
系统可以通过读取状态寄存器来确定擦除操作的状态。欲了解这些状态位的信息，请参见第 36 页上的 *状态寄存器*。

扇区擦除操作开始后，状态寄存器读取和擦除挂起指令均有效。嵌入式算法控制器会忽略其他所有指令。不过，硬件复位会立即终止擦除操作，并在经过 t_{RPH} 时间后返回读取模式。如果扇区擦除操作被终止，一旦器件执行了复位操作，必须重新启动扇区擦除指令序列，以确保数据完整性。

欲了解参数和时序框图的信息，请参见第 19 页上的 *嵌入式算法控制器 (EAC)*。

受 ASP DYB 和 PPB 位保护的扇区不会被擦除。请参见第 51 页上的 *软件接口参考*。如果尝试擦除一个锁定扇区，擦除操作会被中止，同时状态寄存器会指示操作失败（参见第 37 页上的 *状态寄存器表*）。

图 6.4 扇区擦除操作



6.2.8 擦除挂起 / 擦除恢复

通过使用擦除挂起指令，系统可以中断扇区擦除操作，然后对闪存存储器阵列中的数据进行读取或编程。此指令仅在扇区擦除或编程操作期间有效。如果在芯片擦除操作期间写入擦除挂起指令，该指令将被忽略。

如果在扇区擦除操作期间写入擦除挂起指令，器件需要最多 t_{ESL} （擦除挂起等待时间）时间来挂起擦除操作并更新状态位。

擦除操作被挂起后，该部分进入擦除挂起模式。系统可以从闪存存储器阵列读取数据或向其编程数据。读取擦除挂起扇区中的地址会生成不确定的数据。系统可通过读取状态寄存器来确定扇区是正在被擦除还是已挂起。欲了解这些状态位的信息，请参见第 36 页上的 [状态寄存器](#)。

擦除挂起的编程操作完成后，EAC 返回擦除挂起状态。系统可以通过读取状态寄存器来确定编程操作的状态，就像在执行标准编程操作一样。

如果在擦除挂起期间编程操作失败，状态寄存器清除或软件复位指令会使器件返回擦除挂起状态。在重新对擦除阵列编程之前，需要恢复并完成擦除操作。

在擦除挂起期间有效的存取和指令包括：

- 读取任何其他非挂起的扇区
- 编程任何其他非挂起的扇区
- 状态读取指令
- 退出 ASO 或指令集退出
- 擦除恢复指令
- SSR 进入
- SSR 读取
- SSR 编程

为恢复扇区擦除操作，系统必须写入擦除恢复指令。器件将返回擦除状态，状态位将更新。进一步写入的恢复指令被忽略。在芯片恢复擦除后，方可写入另一个擦除挂起指令。

请注意，当器件处于擦除挂起状态时，不能进入 DYB ASO。

6.2.9 非易失性配置寄存器与易失性配置寄存器

非易失性配置寄存器（NVCR）和易失性配置寄存器（VCR）用于定义 HyperFlash 总线的工作条件。可配置以下特性：

1. 回卷突发长度（16 字节、32 字节或 64 字节回卷突发）
 - a. 16 字节和 32 字节回卷突发要求进行延迟，64 字节循环触发则按照第 32 页上的 [64 字节的回卷突发地址序列（延迟代码 = 16）](#) 表所述的内容进行操作。
2. 读取延迟（5 到 16 个时钟周期，用于延迟初始读取操作）。
3. 输出驱动强度
4. 是否使用 4 kB 参数扇区，以及它们如何被映射到地址映射内。
5. SSR 冻结位用于锁定安全硅区域。
6. xVCR 冻结位用于锁定易失性配置寄存器和非易失性配置寄存器。

可以加载并读取 VCR 和 NVCR 的内容，如第 51 页上的 [指令定义表](#) 所述。在上电过程中或在发生硬件复位后，HyperFlash 器件使用 NVCR 的内容来定义总线特性。如果主机系统加载 VCR，则 VCR 的内容将定义总线特性（图 6.5）。预保持 NVCR 的默认设置，以便在引导操作期间该设置与主机控制器的设置对齐。在引导过程中经常使用优化设置来更行 VCR。加载完 VCR 后，总线特性源将从 NVCR（上电或硬件复位后）转移到 VCR。一旦加载好了 VCR，则只有发生上电或硬件复位时才能使总线特性返回到 NVCR 设置状态。解锁后，只要设别处于空闲状态，便可以随时替换 VCR。

NVCR 的擦除和重新编程次数是由 n_NVCR 规范定义（请参见第 56 页上的 [编程 / 擦除耐久性表](#)）的。为了确保在 NVCR 编程期间和编程后总线配置一致，需要在编程 NVCR 时使用 VCR 定义总线工作特性。

VCR 和 NVCR 配置寄存器的位分配

xVCR 位	功能	设置 (二进制)
xVCR.15	保留	1 — 保留 (默认)
xVCR14 – xVCR12	驱动强度	请参见 第 30 页上的 驱动强度代码表
xVCR.11	xVCR 冻结	0 — 锁定 VCR 或 NVCR (不能向 NVCR 进行任何编程 / 擦除操作, VCR 没有任何改变) 1 — 解锁 VCR 和 NVCR (出厂默认设置)
xVCR.10	SSR 冻结	0 — 锁定安全硅区域 (禁止编程) 1 — 解锁安全硅区域 (出厂默认设置)
xVCR.9 – xVCR.8	参数扇区映射	00 — 参数扇区和读取密码扇区被映射到的最低地址 01 — 参数扇区和读取密码扇区被映射到的最高地址 10 — 统一扇区和读取密码扇区被映射到最低地址 (出厂默认设置) 11 — 统一扇区和读取密码扇区被映射到最高地址
xVCR.7 – xVCR.4	读取延迟	0000 — 5 个时钟周期延迟 0001 — 6 个时钟周期延迟 0010 — 7 个时钟周期延迟 0011 — 8 个时钟周期延迟 0100 — 9 个时钟周期延迟 ... 1011 — 16 个时钟周期延迟 (出厂默认设置) 请参见 第 10 页上的 延迟代码选项相应的最大工作频率表
xVCR.3	保留	1 — 保留 (默认)
xVCR.2	保留	0 — 保留 (默认)
xVCR.1 – xVCR.0	突发长度	00 — 保留 01 — 64 个字节 10 — 16 个字节 11 — 32 个字节 (出厂默认设置)

注意:

1. 非易失性和易失性配置寄存器中的位放置情况完全相同。

驱动强度代码

xVCR14 – xVCR12	V _{CCQ} = 1.8 V 时的典型阻抗	V _{CCQ} = 3 V 时的典型阻抗	单位
000 (默认)	27	20	Ω
001	117	71	
010	68	40	
011	45	27	
100	34	20	
101	27	16	
110	24	14	
111	20	12	

注意:

1. V_{CCQ} 为额定值、温度为 25°C 的条件下所测量的典型阻抗。

```

graph TD
    NVCR[非易失性配置寄存器] --> CRMUX[CR选择复用器]
    VCR[易失性配置寄存器] --> CRMUX
    CRMUX -- "DIR=0=NVCR" --> DIR
    CRMUX -- "DIR=1=VCR" --> DIR
    DIR --> DIR_SIGNAL[DIR]
    DIR_SIGNAL --> SET_RESET[设置清除清除]
    SET_RESET --> DIR_SIGNAL
    SET_RESET --> CRMUX
    SET_RESET --> NVCR
    SET_RESET --> VCR
    SET_RESET --> DEV_CONFIG[设备配置]
    SET_RESET --> LOAD_VCR[加载易失性配置寄存器]
    SET_RESET --> POR[上电复位]
    SET_RESET --> HR[硬件复位]
  
```

非易失性配置寄存器

易失性配置寄存器

CR选择复用器
DIR=0=NVCR
DIR=1=VCR

DIR

设置清除清除

加载易失性配置寄存器

上电复位

硬件复位

设备配置

1. 软件复位不会更改 **CR** 选择复用器的状态。
2. 编程或擦除 **NVCR** 不会影响先前加载的 **VCR** 的内容。
3. 如果未加载 **VCR**，则编程 **NVCR** 会使 **VCR** 加载新的 **NVCR**。

上电或硬件复位后，VCR 和 NVCR 将立即冻结各位

NVCR 中第 11 位的值	VCR 中第 11 位的值	NVCR	VCR
1	1	可编程 / 可擦除	可设置 / 可清除
1	0	暂时锁定	暂时锁定
0	1	可编程 / 可擦除	可设置 / 可清除 (1)、(3)
0	0	永久锁定	永久锁定 (4)

1. 编程或擦除 NVCR 不起任何作用，直到发生下一个 POR 或硬件复位为止。
2. 加载 VCR 会立即改变状态。
3. 在 NVCR[11] = VCR[11] = 1 并且 NVCR[11] 位被编程后会进入这种状态。此状态只能维持到下一次 POR 或硬件复位为止。之后，NVCR[11] = VCR[11] = 0。
4. 在发生 POR 或硬件复位后，并且已经配置好了 NVCR[11] 的情况下，将进入这种状态。

突发序列示例

VCR/NVCR [1:0]	CA[45]	回卷边界 (字节)	起始地址 (十六进制格式)	地址序列 (十六进制格式) (字)
XX	1	线性	XXXXXX03	03、04、05、06、07、08、09、0A、0B、0C、0D、0E、0F、10、11、12、13、14、15、16、17、18、...
10	0	16	XXXXXX02	02、03、04、05、06、07、00、01、...
10	0	16	XXXXXX0C	0C、0D、0E、0F、08、09、0A、0B、...
11	0	32	XXXXXX0A	0A、0B、0C、0D、0E、0F、00、01、02、03、04、05、06、07、08、09、...
11	0	32	XXXXXX1E	1E、1F、10、11、12、13、14、15、16、17、18、19、1A、1B、1C、1D、...
01	0	64	XXXXXX03	03、04、05、06、07、08、09、0A、0B、0C、0D、0E、0F、10、11、12、13、14、15、16、17、18、19、1A、1B、1C、1D、1E、1F、00、01、02、...
01	0	64	XXXXXX2E	2E、2F、30、31、32、33、34、35、36、37、38、39、3A、3B、3C、3D、3E、3F、20、21、22、23、24、25、26、27、28、29、2A、2B、2C、2D、...

64 字节的回卷突发地址序列（延迟代码 = 16）

目标地址		时钟周期																																																								总线 转换 + 初始 延迟				CA0				CA1				CA2				延迟计数																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
		0	1	2	3	...	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
0	0							0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
1	1							1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	X	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
2	2							2	3	4	5	6	7	8	9	10	11	12	13	14	15	X	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
3	3							3	4	5	6	7	8	9	10	11	12	13	14	15	X	X	X	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
4	4							4	5	6	7	8	9	10	11	12	13	14	15	X	X	X	X	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
5	5							5	6	7	8	9	10	11	12	13	14	15	X	X	X	X	X	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
6	6							6	7	8	9	10	11	12	13	14	15	X	X	X	X	X	X	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
7	7							7	8	9	10	11	12	13	14	15	X	X	X	X	X	X	X	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
8	8							8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	0	1	2	3	4	5	6	7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

图标:

X: 标注 RWDS 不切换时总线上的空闲时间。 -: 表明 64 字节回卷突发已完成。

目标地址	时钟周期																																																										
	0	1	2	3	...	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52														
0							0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52
1							1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	
2							2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52		
3							3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52			
4							4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52				
5							5	6	7	8	9	10	11	12	13	14	15	X	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52				
6							6	7	8	9	10	11	12	13	14	15	X	X	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52				
7							7	8	9	10	11	12	13	14	15	X	X	X	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52				
8							8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52								
9							9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52									
10							10	11																																																			

X: 标注 RWDS 不切换时总线上的空闲时间。-: 表明 64 字节回卷突发已完成。

6.2.10 进入和退出 ASO

6.2.10.1 ID-CFI ASO

系统可以通过在读取模式期间发出 ID-CFI 进入指令序列来进入 ID-CFI ASO。此进入指令使用指令中的扇区地址（SA）来确定哪个扇区将被覆盖。欲了解详细信息，请参见第 51 页上的 [指令定义表](#)、第 17 页上的 [ID-CFI 地址映射概览表](#)、第 17 页上的 [5.2.1 章节](#)，[器件 ID](#) 以及第 18 页上的 [5.2.2 章节](#)，[通用闪存接口（CFI）](#)。

ID-CFI ASO 允许实现下列操作：

- 使用与进入指令中所用 SA 相同的 SA 来读取 ID-CFI ASO。
- ASO 退出。

下面是使用 CFI 进入和退出功能的 C 源代码示例。有关 Spansion 闪存存储器软件开发指南的通用信息，请参见 *Spansion 低级驱动器用户指南*。

```
/* Example: CFI Entry command */
*( (UINT16 *)base_addr + 0x55 ) = 0x0098; /* write CFI entry command */

/* Example: CFI Exit command */
*( (UINT16 *)base_addr + 0x00 ) = 0x00F0; /* write cfi exit command */
```

6.2.10.2 状态寄存器 ASO

状态寄存器读取指令发出时，寄存器捕获当前状态，并进入 ASO。状态寄存器 ASO 中的第一次读取访问使系统退出 ASO，并将其返回到状态寄存器读取指令发出时使用的地址空间映射。在读取状态并退出状态寄存器 ASO 之前，不应发送任何其他指令。状态寄存器的输出内容只是突发读取的第一个数据值。在随后时钟周期内，便不能确定输出的数据。

6.2.10.3 安全硅区域 ASO

系统可以通过在读取模式期间发出安全硅区域进入命令序列来存取安全硅区域。此进入指令使用指令中的扇区地址（SA）来确定哪个扇区将被覆盖。

安全硅区域 ASO 允许实现下列操作：

- 使用与进入命令中所用 SA 相同的 SA 读取安全硅区域。在 SSR 范围外的 SA 内执行的读取操作会返回不确定的数据。
- 读取安全硅区域范围外的某个 SA，会检索阵列数据。读取阵列不会从 SSR ASO 中退出。
- 使用字或写入缓冲器编程命令对用户安全硅区域进行编程。
- ASO 退出使用传统安全硅区域退出命令实现软件向后兼容。
- ASO 退出也可以使用 ASO 通用退出指令，以实现一致的退出方式。

6.2.10.4 ASP 配置寄存器（ASPR）ASO

系统可以通过在读取模式期间发出 ASP 配置寄存器进入指令序列来存取 ASP 配置寄存器。此进入指令不会使用进入指令中的扇区地址。ASP 配置寄存器所在的位置是器件地址空间中的字位置 0。器件地址空间中的所有其他位置均为未定义。

ASP 配置寄存器 ASO 允许实现以下操作：

- 使用器件地址 0 位置读取 ASP 配置寄存器。
- 使用修改版的字编程指令对用户 ASP 配置寄存器进行编程。
- ASO 退出使用传统指令集退出指令实现软件向后兼容。
- ASO 退出也可以使用 ASO 通用退出指令，以实现一致的退出方式。

6.2.10.5 密码 ASO

系统可以通过在读取模式期间发出密码进入指令序列来存取密码 ASO。此进入指令不会使用进入指令中的扇区地址。密码的位置是在器件地址空间中字位置 0 到 3。器件地址空间中的所有其他位置均为未定义。

密码 ASO 允许下列操作：

- 使用器件地址位置 0 到 3 读取密码。
- 使用修改版的字编程指令对密码进行编程。
- 使用密码解锁指令解锁 PPB 锁定位。
- ASO 退出使用传统指令集退出指令实现软件向后兼容。
- ASO 退出也可以使用 ASO 通用退出指令，以实现一致的退出方式。

6.2.10.6 PPB ASO

系统可以通过在读取模式期间发出 PPB 进入指令序列来存取 PPB ASO。此进入指令不会使用进入指令中的扇区地址。扇区的 PPB 位出现在扇区中所有字位置的位 0。

PPB ASO 允许下列操作：

- 在扇区中任意字的位 0 读取扇区的 PPB 保护状态。
- 使用修改版的字编程指令对 PPB 位进行编程。
- 使用 PPB 擦除指令擦除所有 PPB 位。
- ASO 退出使用传统指令集退出指令实现软件向后兼容。
- ASO 退出也可以使用 ASO 通用退出指令，以实现一致的退出方式。

6.2.10.7 PPB 锁定 ASO

系统可以通过在读取模式期间发出 PPB 锁定进入指令序列来存取 PPB 锁定 ASO。此进入指令不会使用进入指令中的扇区地址。全局 PPB 锁定位出现在器件中所有字位置的位 0。

PPB 锁定 ASO 允许下列操作：

- 在器件地址空间中任意字的位 0 读取 PPB 锁定保护状态。
- 使用修改版的字编程指令清楚 PPB 锁定位。
- ASO 退出使用传统指令集退出指令实现软件向后兼容。
- ASO 退出也可以使用 ASO 通用退出指令，以实现一致的退出方式。

6.2.10.8 动态保护位 (DYB) ASO

系统可以通过在读取模式期间发出 DYB 进入指令序列来存取 DYB ASO。此进入指令不会使用进入指令中的扇区地址。扇区的 DYB 位出现在扇区中所有字位置的位 0。

DYB ASO 允许下列操作：

- 在扇区中任意字的位 0 读取扇区的 DYB 保护状态。
- 使用修改版的字编程指令设置 DYB 位。
- 使用修改版的字编程指令清除 DYB 位。
- ASO 退出使用传统指令集退出指令实现软件向后兼容。
- ASO 退出也可以使用 ASO 通用退出指令，以实现一致的退出方式。

6.2.10.9 软件 (指令) 复位 /ASO 退出

软件复位是指令集的一部分 (参见第 51 页上的*指令定义表*)，也可使 EAC 返回待机状态，在下列情况下必须使用：

- 退出 ID/CFI 模式
- 发生超时时清除数据轮询的超时位 (DQ5)

软件复位不影响 EA 模式。编程或擦除已经开始后，在操作完成前，软件复位指令将一直被忽略。软件复位不影响输出；其主要作用是从 ASO 模式或者从失败的编程或擦除操作返回读取模式。

软件复位可以使未定义的状态 (因指令序列无效而造成) 返回读取模式。不过，从某些未定义状态返回正常操作可能需要硬件复位。

软件复位没有等待时间要求。软件复位指令在 t_{WPH} 期间执行。

6.2.11 状态寄存器

编程和擦除操作的状态由一个 16 位状态寄存器指示。发出状态寄存器读取指令后，将对状态寄存器的内容进行一次读取。状态寄存器的内容在器件地址空间的所有位置内均为别名（被覆盖）。覆盖对一次读取访问有效，尤其是发出状态寄存器读取指令后所进行的下一次读取访问。对状态寄存器进行一次访问后，将退出状态寄存器 ASO。

状态寄存器包含与最近完成的嵌入式算法（EA）结果相关的位 — 成功或失败：

- 擦除状态（位 5），
- 编程状态（位 4），
- 写入缓冲区异常终止（位 3），
- 扇区锁定状态（位 1），
- 扇区擦除状态位（位 0）。

以及，与正在执行的 EA 的当前状态相关的位：

- 器件繁忙（位 7），
- 擦除挂起（位 6），
- 编程挂起（位 2），

当前状态位表明 EA 状态：正在进行、挂起或已完成。

高 7 位（位 15:9）被保留。这些位的值不确定（高或低），在每次进行状态读取时发生变换。软件读取状态时，这些位将被忽略并被视为‘无需关注’。

清除状态寄存器指令和软件复位指令将结果相关的状态寄存器位（位 5、位 4、位 3、位 1 和位 0）清除为 0，但不影响到当前状态位。

状态寄存器

位编号	15:9	8	7	6	5	4	3	2	1	0
位说明	保留	保留	设备准备就绪位	擦除挂起状态位	擦除状态位	编程状态位	写入缓冲区异常终止状态位	编程挂起状态位	扇区锁定状态位	扇区擦除状态位
位名			DRB	ESSB	ESB	PSB	WBASB	PSSB	SLSB	ESTAT
复位状态	X	0	1	0	0	0	0	0	0	0
忙碌状态	无效	无效	0	无效	无效	无效	无效	无效	无效	无效
准备就绪状态	X		1	0 = 无擦除被挂起 1 = 擦除被挂起	0 = 擦除成功 1 = 擦除失败	0 = 编程成功 1 = 编程失败	0 = 编程未中止 1 = 编程在写入缓冲区指令期间被中止	0 = 没有编程被挂起 1 = 编程被挂起	0 = 扇区在操作期间未被锁定 1 = 扇区锁定错误	0 = 扇区擦除状态指令结果 = 上次擦除没有完全成功 1 = 扇区擦除状态指令结果 = 上次擦除已完全成功

注意:

1. 位 15 到 9 保留给将来使用，它们可能显示为 0 或 1。检查状态时，应该忽略（屏蔽）这些位。
2. 当器件中没有嵌入式算法正在执行时，位 7 为 1。
3. 仅在位 7 为 1 时，位 8 和位 6 到 0 才有效。
4. 冷复位或热复位可将所有位置于它们的复位状态。
5. 清除状态寄存器指令或软件复位指令均可将位 5、位 4、位 3、位 1 和位 0 清零。
6. 发出擦除挂起指令时，用户必须继续读取状态，直到 DRB 为 1 为止。
7. 擦除恢复指令可将 ESSB 清零。
8. ESB 反映最近擦除操作成功或失败。
9. PSB 反映最近编程操作成功或失败。
10. 在擦除挂起期间，对挂起扇区进行编程会导致编程失败并将编程状态位设置为 1。
11. 在擦除挂起期间，擦除操作会导致擦除失败并将擦除状态位设置为 1。
12. 在编程挂起期间，编程操作会导致编程失败并将编程状态位设置为 1。
13. 在编程挂起期间，擦除操作会导致擦除失败并将擦除状态位设置为 1。
14. 发出编程挂起指令时，用户必须继续读取状态，直到 DRB 为 1 为止。
15. 编程恢复指令可将 PSSB 清零。
16. SLSB 表明编程或擦除操作因目标存储器区域被锁定而失败。
17. SLSB 反映最近编程或擦除操作的状态。

6.2.12 错误类型和清除步骤

嵌入式操作状态方法报告的错误类型有三种。根据错误类型，报告的状态和清除错误状态的步骤会有所不同。下面是错误状态清除情况：

- 如果在错误发生之前器件进入了 ASO，它持续进入该状态，等待 ASO 读取或指令写入。
- 如果在错误发生之前一个擦除操作被挂起，器件返回擦除挂起状态，等待闪存阵列读取或指令写入。
- 否则，器件将处于待机状态，等待闪存阵列读取或指令写入。

6.2.12.1 嵌入式操作错误（与无效密码）

如果在嵌入式操作（编程、擦除、空白检查、或密码解锁）期间发生错误，嵌入式算法控制器（EAC）保持工作状态。状态寄存器通过有效状态位（SR[7] = 1）指示就绪状态，用于表明错误原因。嵌入式算法控制器一直工作，直至主机系统状态监控检测到错误状态并且错误状态被清除。

在嵌入式算法错误状态期间，状态寄存器的情况如下显示：

- SR[7] = 1；显示有效状态
- SR[6] = X；在 EA 错误期间可能是或不是擦除挂起
- 擦除或空白检查错误时 SR[5] = 1；其他情况 = 0
- 编程错误或密码无效时 SR[4] = 1；其他情况 = 0
- SR[3] = X；视为“无需关注”（屏蔽）
- SR[2] = 0；没有编程挂起
- SR[1] = 0
- SR[0] = X；视为“无需关注”（屏蔽）

当检测到嵌入式算法错误状态时，有必要清除错误状态，以便返回正常操作，为新读取或指令写入做好准备。错误状态可通过写入下列指令来清除：

- 复位指令
- 状态寄存器清除指令

在嵌入式算法错误状态期间可接受的指令包括：

- 状态寄存器读取
- 复位指令
- 状态寄存器清除指令

6.2.12.2 保护错误

如果嵌入式算法尝试更改受保护区域的数据（编程或擦除受保护的扇区或 OTP 区域），器件（EAC）会进入忙碌状态 20 到 100 μ s，然后返回正常操作。保护机制包括 DYB、PPB 和锁定。在忙碌状态过程中，状态寄存器通过无效状态位（SR[7] = 0）来表示非就绪状态。如果尝试编程或擦除一个锁定扇区，编程操作会被中止，同时状态寄存器会指示操作失败（参见第 37 页上的 [状态寄存器表](#)）。

在保护错误状态忙碌期间可接受的指令包括：

- 状态寄存器读取

忙碌期间结束时，器件返回正常操作，状态寄存器通过有效状态位显示就绪状态。器件已准备好对闪存阵列执行新读取或写入指令。

在经过保护错误状态忙碌时间后，状态寄存器情况如下显示：

- SR[7] = 1；显示有效状态
- SR[6] = X；在保护错误忙碌期间可能是或不是擦除挂起
- 擦除错误时 SR[5] = 1；其他情况 = 0
- 编程或密码解锁错误时 SR[4] = 1；其他情况 = 0
- SR[3] = X；视为“无需关注”（屏蔽）
- SR[2] = 0；没有编程挂起
- SR[1] = 1；因尝试更改受保护的位置而发生错误
- SR[0] = X；视为“无需关注”（屏蔽）

在经过保护错误状态忙碌时间后可接受的指令包括：

- 任何指令

在编程状态位被置位的情况下，再次进行编程会马上清除 SR[4]。在擦除状态位被设置的情况下，进一步进行擦除立即清除 SR[6]。

6.2.12.3 写入缓冲区中止

如果在执行写入到缓冲器指令期间发生错误，(EAC) 会保持忙碌状态。状态寄存器通过有效状态位来显示就绪状态。器件一直忙碌，直至主机系统状态监控检测到错误状态并且错误状态被清除为止。

在嵌入式算法错误状态期间，状态寄存器的情况如下显示：

- SR[7] = 1；显示有效状态
- SR[6] = X；在 WBA 错误状态期间可能是或不是擦除挂起
- SR[5] = 0；擦除成功
- SR[4] = 1；与编程相关的错误；其他情况 = 0
- SR[3] = 1；写入缓冲区异常终止
- SR[2] = 0；没有编程挂起
- SR[1] = 0；在操作期间扇区未锁定
- SR[0] = X；视为“无需关注”（屏蔽）

当检测到 WBA 错误状态时，有必要清除错误状态，以便返回正常操作，为新读取或指令写入做好准备。错误状态可通过写入下列指令来清除：

- 写入缓冲区异常终止复位指令
 - 清除状态寄存器并返回正常操作
- 状态寄存器清除指令

在嵌入式算法错误状态期间可接受的指令包括：

- 状态寄存器读取
 - 读取状态寄存器并返回 WBA 忙碌状态
- 写入缓冲区异常终止复位指令
- 状态寄存器清除指令

在实现嵌入式算法期间，执行与状态寄存器读取无关的读取操作会使 RWDS 被切换并返回不确定的数据。

6.3 数据保护

6.3.1 安全硅区域

每个器件都有一个 1024 字节大小的一次可编程安全硅区域（SSR）地址空间，该空间独立于闪存存储器阵列。SSR 分为 32 个可独立锁定的 32 字节对齐并且大小为 32 字节的区域。

在所述起始地址为 0 的 32 字节区域中：

- 低 16 个地址字节由 Spansion 通过一个 128 位随机数来编程。只有 Spansion 才能编程这些字节。尝试向这些位置写入 0 会导致操作失败，并生成一个编程状态错误（SR[4] = 1）。
- 后面的 4 个高地址字节（SSR 锁定字节）分别为每个 SSR 区域提供一个位，以便永久性阻止对每个区域进行的编程。从 Spansion 出厂时，这些字节都被擦除。编程某个 SSR 区域后，可通过锁定该区域来防止发生额外的编程。通过编程 SSR 锁定字节中的相关保护位，可以锁定所需区域。
- 最低地址范围后面的 12 个高字节被保留，以供将来使用（RFU）。主机系统可能对这些 RFU 字节中的各个位进行编程处理，但必须注意，新器件可能使用这些位来保护范围更大的 SSR 空间。从 Spansion 出厂时，这些字节都被擦除。

从 Spansion 出厂时，剩余的区域也被擦除。此外，还可以使用这些区域编程额外的永久数据。

欲了解 SSR 存储器空间的图形表示，请参见第 42 页上的图 6.6，[SSR 地址空间](#)。

SSR 存储器空间用于增强系统安全性。SSR 值（比如由 Spansion 编程的随机数）可用来连接闪存组件和系统 CPU/ASIC，以防止器件被替换。

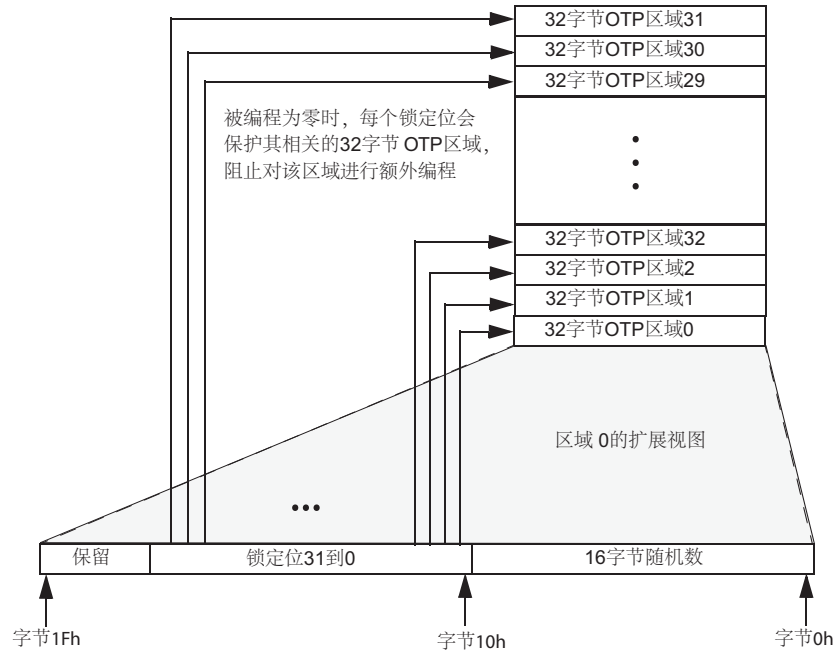
当配置寄存器 SSR 冻结（xVCR10）位被清零（或被编程为 0（若用 NVCR））时，它能阻止对整个 SSR 存储器空间进行编程。在上电系统正常操作剩余时间内，上述操作允许可靠引导代码控制 SSR 区域的编程，然后通过设置冻结位来阻止对 SSR 存储器空间进行额外的编程。

6.3.1.1 读取安全硅区域存储器空间

一旦使用 SSR 进入序列进入 SSR ASO，便立即读取 SSR 区域。SSR 被映射到 SSR 进入指令序列期间所指定的扇区内。如果执行的 SSR 读取操作属于 SSR 进入指令序列期间指定的扇区内，但不在有效的 8 kB SSR 地址范围内，那么该操作将返回不确定的数据。对 SSR ASO 未覆盖的扇区进行读取将检索阵列数据。SSR 退出序列使器件返回到阵列读取 ASO。

6.3.1.2 编程安全硅区域存储器空间

一旦使用 SSR 进入序列进入 SSR ASO，可立即编程 SSR 区域。SSR 编程指令协议与阵列普通编程协议相同。可多次向任意已给 SSR 地址发送 SSR 编程序列，但无法擦除该地址空间。第 42 页上的图 6.6，[SSR 地址空间](#)中显示的是 SSR 编程的有效地址范围。在有效 SSR 地址范围外编程 SSR 会忽略地址 A9 和更高地址，并转到读取有效 SSR 地址范围内。当冻结位为 0 时，对 SSR 进行编程会导致编程操作失败，但未指明失败情况。选用 ASP 保护模式不能保护 SSR 地址空间。冻结 SSR 位（xVCR.10）可能用于保护 SSR 地址空间。SSR 退出序列使器件返回到读取模式。

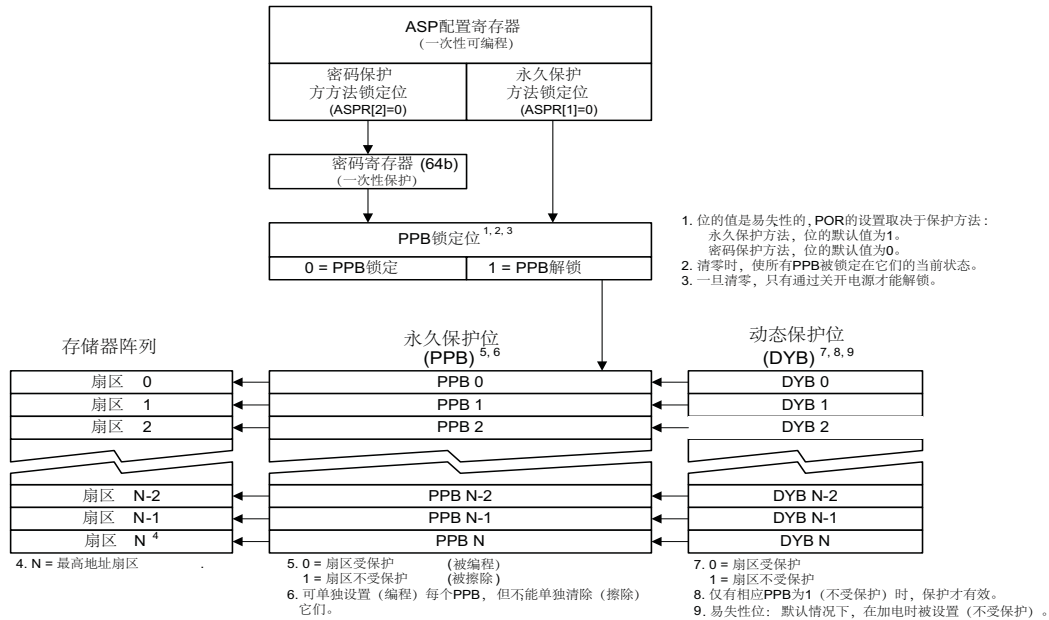
图 6.6 SSR 地址空间

SSR 地址映射

区域	字节地址范围（十六进制）	目录	初始供应状态（十六进制）
区域 0	0000	Spansion 编程随机数值的最低字节	Spansion 编程随机数值
	...	...	
	000F	Spansion 编程随机数值的最高字节	
	0010 - 0013	区域锁定位 字节 10 [位 0] = 0 时，它将锁定区域 0，阻止对该区域进行编程操作 ... 字节 13 [位 7] = 0 时，它将锁定区域 31，禁止向该区域进行编程操作	所有字节 = FF
	0014 - 001F	保留供将来使用	所有字节 = FF
区域 1	0020 - 003F	可供用户编程	所有字节 = FF
区域 2	0040 - 005F	可供用户编程	所有字节 = FF
...	...	可供用户编程	所有字节 = FF
区域 31	03E0 - 03FF	可供用户编程	所有字节 = FF

6.3.2 高级扇区保护（ASP）

高级扇区保护（ASP）是一组独立的硬件和软件方法，分别用于禁止或允许对任何或所有扇区执行编程或擦除操作。此部分介绍存储器阵列中所存储数据的各种保护方法。图 6.7 是这些方法的概述。

图 6.7 高级扇区保护概述



每个闪存存储器阵列扇区都有与其关联的一个非易失性（PPB）和一个易失性（DYB）保护位。无论哪一位是 0 时，扇区都受到保护，不能执行编程和擦除操作。

PPB 锁定为 0 时，不能执行编程和擦除操作。PPB 锁定状态可通过两种方法进行管理：持久保护和密码保护。

持久保护方法在 POR 或硬件复位期间将 PPB 锁定位设为 ‘1’，因此 PPB 位不会受到器件复位的保护。软件复位不会对 PPB 锁定位产生影响。有一个指令可将 PPB 锁定清除为 ‘0’，以保护 PPB。持久保护方法没有指令可设置 PPB 锁定，因此 PPB 锁定将一直为 0，直至下一次断电或硬件复位。持久保护方法允许引导代码通过编程或擦除 PPB 来更改扇区保护，然后通过清除 PPB 锁定在正常系统操作的剩余时间内保护 PPB，不使其更改。这有时称为引导代码控制的扇区保护。

密码方法在 POR 或硬件复位期间将 PPB 锁定清除为 0，以保护 PPB。对于密码方法，可以永久地编程并隐藏一个 64 位密码。通过一个指令来提供一个密码，将其与隐藏密码进行比较。如果密码匹配，则 PPB 锁定位设为 1，以取消 PPB 保护。通过一个指令来将 PPB 锁定清零。此方法需要使用一个密码来控制 PPB 保护。

PPB 锁定管理方法通过编程锁定寄存器中的 OTP 位来选择，以便永久性选择所使用的方法。

从 Spansion 出厂时，PPB 位被清除，因此所有闪存存储器阵列扇区不受保护。

6.3.3 PPB 锁定

持久保护锁定是一个易失性位，用于保护所有 PPB 位。清除为 0 时，它锁定所有 PPB；设成 1 时，允许更改 PPB。每个器件只有一个 PPB 锁定位。

PPB 锁定指令用于将该位清零。仅在所有 PPB 均配置为所需的设置后，才必须将 PPB 锁定位清零。

在持久保护模式下，PPB 锁定在 POR 或硬件复位期间设为 1。使用 PPB 锁定位清除序列清除后，没有软件指令序列能够设置 PPB 锁定位，只有再一次硬件复位或加电才能设置 PPB 锁定位。

在密码保护模式下，PPB 锁定在 POR 或硬件复位期间清除为 0。PPB 锁定位只能通过密码解锁指令序列设为 1。使用 PPB 锁定位清除序列会使 PPB 锁定位清零。

6.3.4 持久保护位（PPB）

持久保护位（PPB）位于一个单独的非易失性闪存阵列中。为每个扇区分配其中一个 PPB 位。当一个 PPB 位为 0 时，其关联扇区受到保护，不能执行编程和擦除操作。PPB 位可分别进行编程，但必须作为一个组进行清除。这与字相似，各个字可以在主阵列中分别编程，但整个扇区必须同时擦除。擦除前的预编程和验证由 EAC 执行。

编程一个 PPB 位需要典型的字编程时间。在编程或擦除 PPB 位期间，可通过访问状态寄存器来确定操作完成的时间。擦除所有 PPB 需要典型的扇区擦除时间。

PPB 锁定位为 0 时，PPB 编程或擦除指令不被执行，直至超时而不编程或擦除 PPB。如果在 PPB 锁定位为 0 时尝试编程或擦除一个 PPB 位，该操作会被中止，同时状态寄存器会指示操作失败（参见第 37 页上的 [状态寄存器表](#)）。

与特定扇区相关的 PPB 的保护状态，可通过在进入 PPB ASO 后写入一个 PPB 状态读取指令来检查。

6.3.5 动态保护位（DYB）

动态保护位是易失性的，对于每个扇区来说是唯一的，可分别进行修改。DYB 只控制那些 PPB 被清除的扇区的保护。通过发出 DYB 设置或清除指令序列，DYB 可设为 0 或清除为 1，从而相应地将每个扇区置于不受保护或受保护状态。此功能允许软件方便地保护扇区，以免意外更改；但并不禁止在需要更改时方便地去除保护。

DYB 可以在需要时随时设为 0 或清除为 1。

6.3.6 扇区保护状态汇总

每个扇区均可以处于下面一种保护状态：

- 解锁 – 扇区不受保护，保护状态可通过一个简单的指令进行更改。在开关电源或硬件复位后，保护状态默认为不受保护。
- 动态锁定 – 扇区受保护，保护状态可通过一个简单的指令进行更改。在开关电源或硬件复位后，保护状态不被保存。
- 持久锁定 – 扇区受保护，只有将 PPB 锁定位置设置为 1 时才能更改保护状态。保护状态是非易失性的，在开关电源或硬件复位后被保存。更改保护状态需要编程或擦除 PPB 位。

扇区保护状态

保护位值			扇区状态
PPB 锁定位置	PPB	DYB	
1	1	1	无保护 — PPB 和 DYB 可更改
1	1	0	保护 — PPB 和 DYB 可更改
1	0	1	保护 — PPB 和 DYB 可更改
1	0	0	保护 — PPB 和 DYB 可更改
0	1	1	保护 — PPB 不可更改，DYB 可更改
0	1	0	保护 — PPB 不可更改，DYB 可更改
0	0	1	保护 — PPB 不可更改，DYB 可更改
0	0	0	保护 — PPB 不可更改，DYB 可更改

6.3.7 ASP 配置寄存器

ASP 配置寄存器（ASPR）包含用于管理扇区安全性的非易失性 OTP 位。

ASP 配置寄存器

位	默认值	名称
15、14、13、12、10、9	1	保留
11	1	混合突发类型使能（不适用于 512 Mb 大小的器件） ASPR[11] = 0: 混合选项 — 一个回卷突发序列后接线性突发序列 ASPR[11] = 1: 传统式 — 纯回卷突发序列
8	0	保留
7	X	保留
6	1	保留
5	1	保留
4	1	保留
3	1	保留
2-1	1	持久 / 密码保护模式锁定位置 ASPR[2:1] = 00: 禁止 ASPR[2:1] = 01: 永久启用密码模式，并冻结 ASPR ASPR[2:1] = 10: 永久启用持久模式，并冻结 ASPR ASPR[2:1] = 11: 暂时启用持久模式（出厂默认设置）
0	1	保留

从工厂出厂时，所有器件默认使用持久保护方法，加电时所有扇区均不受保护。器件程序师或主机系统随后可以选择使用哪种扇区保护方法。通过对下面两个、一次性可编程、非易失性位中的一位进行编程，在相应模式下永久性锁定该部分：

- 持久保护模式锁定位置（ASPR[1]）
- 密码保护模式锁定位置（ASPR[2]）

如果同时对这两个锁定位置（ASPR[2] 和 ASPR[1]）进行编程，操作被中止，状态寄存器位 SR[4] 和 SR[1] 被设置，以表示操作失败。一旦编程了密码模式锁定位置，持久模式锁定位置永远被禁用，保护方案无法更改。类似的，如果编程了持久模式锁定位置，将永远禁用密码模式。成功编程 ASPR[2] 或 ASPR[1] 后，任何尝试编程 ASPR 的操作将被中止，状态寄存器位 SR[4] 和 SR[1] 被设置，以表示操作失败。

如果选择密码模式，必须在设置相应的锁定寄存器位之前编程密码。必须按序列以 0-1-2-3 序列编程四字密码，以别的序列编程密码会导致未定义的行为。对密码保护模式锁定位编程后，需要执行一次关电源、硬件复位或 PPB 锁定位设置指令将 PPB 锁定位设为 ‘0’，才能保护 PPB 阵列。

ASP 配置寄存器的编程时间与典型的字编程时间相同。在 ASP 配置寄存器编程 EA 的过程中，系统还可通过读取状态寄存器来确定 ASPR 编程状态。欲了解这些状态位的信息，请参见第 36 页上的 [状态寄存器](#)。

6.3.8 持久保护模式

持久保护方法在 POR 或硬件复位期间将 PPB 锁定位设为 ‘1’，因此 PPB 位不会受到器件复位的保护。有一个指令可将 PPB 锁定位清除为 ‘0’，以保护 PPB。持久保护方法没有指令可将 PPB 锁定位设为 ‘1’，因此 PPB 锁定位将一直为 ‘0’，直至下一次断电或硬件复位为止。

6.3.9 密码保护模式

密码保护模式通过在设置 PPB 锁定时要求 64 位密码，可比持久扇区保护模式提供更高级别的安全性。除了密码要求外，在加电和硬件复位后，PPB 锁定位还清除为 ‘0’，以确保在加电时提供保护。通过输入整个密码并成功执行密码解锁指令后，PPB 锁定位将设为 ‘1’，从而允许修改扇区 PPB。

密码保护注意事项：

- 密码编程指令只能编程 “0”。
- 从 Spansion 出厂时，密码全部是 1。它位于自己的存储空间中，可通过使用密码编程和密码读取指令进行访问。
- 所有 64 位密码组合均为有效密码。
- 编程和验证密码后，必须编程密码模式锁定位（为 ‘0’），以防止读取密码。
- 一旦编程了密码保护模式锁定位（为 ‘0’），即可防止在数据总线上读取 64 位密码和进一步编程密码。对密码区域的所有额外编程和读取指令均被禁止，这些指令被忽略。尝试编程受保护的密码将设置扇区锁定状态位（SR[1]）和编程状态位（SR[4]）。如果尝试进一步编程密码或密码保护模式锁定位，编程操作被中止，同时状态寄存器指示操作失败（参见第 37 页上的 [状态寄存器表](#)）。编程了密码保护模式锁定位后，无法验证密码内容。只能在选择密码保护模式前验证密码。
- 不能擦除密码模式锁定位。
- 要想启用解锁功能，只要以任何序列输入整个 64 位密码即可。如果密码解锁指令所提供的密码与隐藏内部密码不匹配，解锁操作失败，失败情况与编程受保护的扇区情况相似。状态寄存器返回就绪状态，编程状态位设为 ‘1’，以指示锁定扇区的编程操作失败。在此情况下，PPB 锁定位由于未提供有效密码而仍旧受到保护，因此不能更改其状态。
- 向器件提供有效的 64 位密码后，器件需要大约 $t_{PSWD} = 100 \mu s$ 时间来设置 PPB 锁定位。
- 每个 t_{PSWD} 时间只能接收一条密码解锁指令（参见第 55 页上的 [嵌入式算法特性表](#)）。这样，如果黑客试图实现所有 64 位组合以找到正确匹配密码，就需要不可思议的超长时间（5800 万年）。EA 状态检查方法可用于确定 EAC 何时准备好接受新的密码指令。
- 如果在设置密码模式锁定位后失去密码，无法清除 PPB 锁定位。

6.3.10 读取密码保护模式

读取密码模式可以取代默认的 [密码保护模式](#)。当用户将 ASPR[5] 编程为 ‘0’ 时，读取密码模式被启用，以代替默认 PPB 密码保护模式。只在编程完密码并将 ASPR[2] 设为 ‘0’ 后，读取密码模式才有效。

启用读取密码保护模式后，闪存存储阵列受保护，不能对其进行读取、编程和擦除。只有非易失性配置寄存器位 xVCR[9:8] 选定的最低或最高（256 kB）扇区地址范围保持可读取状态，直至成功执行密码解锁指令为止。对阵列中读取保护区域进行读取操作将返回到读取可读扇区。

在该模式下，PPB 锁定位用于控制地址的高序位。当 PPB 锁定位为 ‘1’ 时，地址中的位正常运行。当 PPB 锁定位为 ‘0’ 时，用于选择主阵列扇区地址范围的地址位均被强制设为 0（xVCR[9:8] = 00 或 10）或设为 1（xVCR[9:8] = 01 或 11）以选择最低或最高地址闪存存储器阵列地址范围，如下表所述。当 xVCR[9:8] = 00 或 10，阵列的底部（零地址）256 kB 是可读的。当 xVCR[9:8] = 01 或 11，阵列的顶部（最大地址）256 kB 是可读的。

与持久和密码保护模式相应的 ASP 配置寄存器选择

ASP 位	默认值	名称
2	1	持久 / 密码保护模式的锁定位 ASPR[2:1] = 00: 禁止 ASPR[2:1] = 01: 永久启用密码模式 ASPR[2:1] = 10: 永久启用持久模式 ASPR[2:1] = 11: 暂时启用持久模式（出厂默认设置）
1	1	

与启动模块地址范围相应的 xVCR 映射

xVCR 位	默认值	名称
xVCR.9 – xVCR.8	11	00 — 参数扇区和读取密码扇区被映射到最低地址 01 — 参数扇区和读取密码扇区被映射到最高地址 10 — 统一扇区和读取密码扇区被映射到最低地址 11 — 统一扇区和读取密码扇区被映射到最高地址

PPB 锁定位为 ‘0’ 时，PPB 位受保护，不能进行编程和擦除。PPB 锁定位为 ‘1’ 时，可编程或擦除 PPB 位。

与 PPB 密码保护模式相似，可以通过 POR 或硬件复位将 PPB 锁定位为 ‘0’。

读取密码保护注意事项：

- 当需要使用读取密码 OPN 选项时，用户可以将 ASPR[5] 位编程为 ‘0’，并根据需要使用 / 不使用读取密码。
- 读取密码方法中密码编程、读取和锁定指令序列的设置与 PPB 密码方法中的默认设置相同。
- 启用读取密码模式和密码保护模式（ASPR[2] 和 ASPR[5] 均被设为 ‘0’）时，所有地址被重定向到引导扇区，直至使用正确密码正常进入密码解锁序列为止。此时，读取密码模式被禁用，所有地址将选择正确的位置。
- 如果发生系统硬件复位，读取密码模式被重新启用。
- ASPR[5] 用于选用读取密码或 PPB 密码。如果 ASPR[5] 为 ‘0’，器件将使用读取密码模式。但是，仅在 ASPR[2] 为 ‘0’ 后读取密码才被启用。此时，只能在顶部或底部扇区范围内选择所有地址，直至通过正常解锁序列和正确密码解锁器件。ASPR[2] 为 ‘1’ 时，可正常选择地址。这样，用户可使用代码进行编程和测试，提供一个密码，然后通过设置 ASPR[2] 为 ‘0’ 进行锁定。
- 如果在使用读取密码保护方法期间发送一个读取密码指令序列，返回的结果是未定义的。只有通过硬件复位、POR 或 PPB 锁定位清除指令序列，才能使 PPB 锁定位的值返回 ‘0’。
- 当 PPB 锁定位为零时，在读取密码模式下只能执行 ID 读取指令、密码解锁指令和阵列读取操作。所有其他指令均被禁用，直至提供密码来启用整个器件读取以及指令正常操作为止。
- 读取密码保护模式有效（ASPR[5] = 0、ASPR[2] = 0、PPB 锁定位 = 0）时，可读取主阵列，但通过强制设置存储器扇区地址为 0 或 1 使得只有引导扇区可见。读取 DYB 或 PPB 地址空间返回未定义的数据。
- 读取密码保护模式有效时，不能编程存储器，也不能对寄存器进行写入操作。RESET 正常运行，通过复位模式位可修改总线协议。

6.3.11 混合突发

指明的另一种突发类型（包括一个回卷突发）后面是一个线性突发。256 Mbit 和容量更小的 HyperFlash 系列产品支持这种突发类型。512 Mbit 的 HyperFlash 不支持这种突发类型。

混合突发开始于目标地址回卷突发组长度内的一次循环处理，然后在超出初始回卷突发长度组尾端处切换到线性突发。16 字节和 32 字节回卷突发长度组支持混合突发，但 64 字节回卷突发长度组不支持这种突发类型。

ASP 配置寄存器的位分配（用于混合突发类型使能）

位	默认值	名称
11	1	混合突发类型使能 ASPR[11] = 0: 混合选项 — 一个回卷突发序列后接线性突发序列 ASPR[11] = 1: 传统式 — 纯回卷突发序列

32 字节和 16 字节混合突发读取的突发序列示例：

1. 32 字节示例（在 32 字节边界内进行循环处理，然后转换到线性突发）
 - a. 06-07-08-09-0A-0B-0C-0D-0E-0F-00-01-02-03-04-05-10-11
 - b. 0E-0F-00-01-02-03-04-05-06-07-08-09-0A-0B-0C-0D-10-11
2. 16 字节示例（在 16 字节边界内进行循环处理，然后转换到线性突发）
 - a. 06-07-00-01-02-03-04-05-08-09
 - b. 03-04-05-06-07-00-01-02-08-09

6.3.12 INT# 输出

INT# 引脚是一个开漏输出引脚，用来通知主机系统闪存器件中发生了某个事件。在下列情况下，用户可以选择从 INT# 输出引脚转换为有效（低电平）状态：

- 从繁忙状态转换为就绪状态

通过中断配置寄存器（ICR）控制操作，INT# 输出（通常为高电平）被启用。中断配置寄存器确定启用内部事件以触发 INT# 输出引脚上的状态转换（高电平转为低电平）的时间。中断状态寄存器指示最后一次清除 ISR 后所发生的被启用的内部事件。如果该引脚被启用，那么发生某个启用的事件时，INT# 输出引脚会从高电平转换为低电平。一旦主机识别 INT# 转换到低电平状态，可读取中断状态寄存器确定导致转换的内部事件。使用下面三种方法可以强制 INT# 输出返回到高阻抗状态（通过外部上拉电阻返回高电平状态）：

- 设置中断配置寄存器中的位 15 为 1 来禁用 INT# 输出。将 ICR[15] 置 1 后，中断状态寄存器将被清除。
- 将中断配置寄存器中适当的事件使能位设置为 1，可以禁用导致输出转换为低电平的事件通道。设置 ICR 中的相应位后，中断状态寄存器中的相关位被清除。
- 复位中断状态寄存器位中适当的位（通过写入 1）。该位指示会使输出转为低电平的内部事件。INT# 输出返回高电平前，必须复位中断状态寄存器中表示低电平并且在中断配置寄存器中已被使能的所有位。

硬件复位（RESET# = 低电平）或某个上电复位也会使 INT# 输出返回到默认（被禁用、高阻抗）状态。硬件复位和上电复位通过将中断配置寄存器恢复为默认（所有中断被禁用）状态来禁用所有中断。

中断配置寄存器

位	功能	类型	POR 默认状态	RESET# 默认状态	说明
15	INT# 输出使能	易失性， 读取 / 写入	1	1	1 = 禁用 INT# 输出（高电平或开漏） 0 = 使能 INT# 输出，内部事件将使该输出高电平转换为低电平
14	保留		1	1	保留
13-5	保留		1	1	保留供将来使用
4	就绪		1	1	1 = 就绪 / 繁忙状态下的转换不会引起 INT# 输出的电平发生转换 0 = 繁忙状态转换为就绪状态时，INT# 输出将从高电平转换为低电平
3	保留		1	1	保留以供将来使用
2	保留		1	1	保留以供将来使用
1	保留		1	1	保留以供将来使用
0	保留		1	1	保留以供将来使用

注意：

1. 上电复位（POR）和硬件复位都会使所有中断通道被禁用。

中断状态寄存器

位	功能	类型	POR 默认状态	RESET# 默认状态	说明
15-5	保留	易失性，读取 / 写入	1	1	保留供将来使用
4	就绪		1	1	1 = 繁忙状态没有转换为就绪状态 0 = 繁忙状态转换为就绪状态
3	保留		1	1	保留以供将来使用
2 (3)、(4)	POR 检测		0	1	1 = 没有发生 POR 0 = 发生了 POR
1	保留		1	1	保留以供将来使用
0	保留		1	1	保留以供将来使用

注意:

1. 硬件复位会使所有 ISR 位被置 1。
2. POR 使 ISR POR 检测位 (ISR[2]) 被清零，并使其他所有位被置 1。
3. ISR[2] 在 POR 期间被清除 (为 0)。只有发生硬件复位 (RESET# = 0) 或对 ISR 进行写操作才能设置该位 (为 1)。
4. INT# 输出状态不受 ISR[2] 的值的影响。
5. 对 ISR 进行写操作只能使各个位从 0 状态翻转到 1 状态。只有生成中断时才能将各个位从 1 翻转到 0。

6.4 易失性与非易失性配置寄存器汇总

6.4.1 非易失性配置寄存器

非易失性配置寄存器

符号	名称	宽度 (位)	NV 类型	默认值	参考资料
NVCR	非易失性配置寄存器	16	P/E	8EBBh	第 29 页上的 6.2.9 章节, 非易失性配置寄存器与易失性配置寄存器
	密码保护寄存器	64	OTP	FFFF FFFF FFFF FFFFh	第 46 页上的 6.3.9 章节, 密码保护模式
PPB	持久保护位	1 位 / 每个扇区	P/E	1	第 44 页上的 6.3.4 章节, 持久保护位 (PPB)
ASPR	ASP 配置寄存器	16	OTP	FEFFh	第 45 页上的 6.3.7 章节, ASP 配置寄存器
POR 时间	上电复位时间	16	OTP	FFFFh	第 66 页上的 11.3.1 章节, 加电 (冷) 复位 (POR)

6.4.2 易失性配置寄存器

易失性配置寄存器

符号	名称	宽度 (位)	默认值	参考资料
VCR	易失性配置寄存器	16	NVCR	第 29 页上的 6.2.9 章节, 非易失性配置寄存器与易失性配置寄存器
DYB	动态保护位	1 位 / 每个扇区	1	第 44 页上的 6.3.5 章节, 动态保护位 (DYB)
	PPB 锁定位	1	ASPR[2]	第 44 页上的 6.3.3 章节, PPB 锁定
ICR	中断配置寄存器	16	FFFFh	第 48 页上的 6.3.12 章节, INT# 输出

6.4.3 易失性结果与状态寄存器

易失性结果与状态寄存器

名称	宽度 (位)	默认值	参考资料
扇区锁定状态	4 位 / 每个扇区	无	请参考注意 (17), 以了解第 51 页上的 指令定义表
状态寄存器	16	xx80h	第 37 页上的 状态寄存器表
中断状态寄存器	16	FFFBh	第 49 页上的 中断状态寄存器表

7. 软件接口参考

7.1 指令汇总

指令定义 (表格 1/3)

指令序列	深度	总线周期 (注意 1-4)													
		第一个周期		第二个周期		第三个周期		第四个周期		第五个周期		第六个周期		第七个周期	
		地址	数据	地址	数据	地址	数据	地址	数据	地址	数据	地址	数据	地址	数据
读取 (5)	1	RA	RD												
复位 /ASO 退出 (6), (14)	1	XXX	F0												
状态寄存器读取 (16)	2	555	70	XXX	RD										
状态寄存器清除	1	555	71												
进入深度掉电模式	3	555	AA	2AA	55	XXX	B9								
编程上电复位定时器寄存器	4	555	AA	2AA	55	555	34	XXX	POR 时间						
读取上电复位定时器寄存器	4	555	AA	2AA	55	555	3C	XXX	RD POR 时间						
加载中断配置寄存器	4	555	AA	2AA	55	555	36	XXX	ICR						
读取中断配置寄存器	4	555	AA	2AA	55	555	C4	XXX	RD ICR						
加载中断状态寄存器	4	555	AA	2AA	55	555	37	XXX	ISR						
读取中断状态寄存器	4	555	AA	2AA	55	555	C5	XXX	RD ISR						
加载易失性配置寄存器	4	555	AA	2AA	55	555	38	XXX	VCR						
读取易失性配置寄存器	4	555	AA	2AA	55	555	C7	XXX	RD VCR						
编程非易失性配置寄存器	4	555	AA	2AA	55	555	39	XXX	NVCR						
擦除非易失性配置寄存器	3	555	AA	2AA	55	555	C8								
读取非易失性配置寄存器	4	555	AA	2AA	55	555	C6	XXX	RD NVCR						
字编程	4	555	AA	2AA	55	555	A0	PA	PD						
写入到缓冲区	6	555	AA	2AA	55	SA	25	SA	WC	WBL	PD	WBL	PD		
将缓冲区中的数据编程到闪存 (确认)	1	SA	29												
复位 “写缓冲区异常中止” 状态 (10)	3	555	AA	2AA	55	555	F0								
芯片擦除	6	555	AA	2AA	55	555	80	555	AA	2AA	55	555	10		
扇区擦除	6	555	AA	2AA	55	555	80	555	AA	2AA	55	SA	30		
空白检查	1	(SA) 555	33												
评估擦除状态	1	(SA) 555	D0												
擦除挂起	1	XXX	B0												
擦除恢复	1	XXX	30												
编程挂起	1	XXX	51												
编程恢复	1	XXX	50												

指令定义 (表格 2/3)

指令序列		周期	总线周期 (注意 1-4)													
			第一个周期		第二个周期		第三个周期		第四个周期		第五个周期		第六个周期		第七个周期	
			地址	数据	地址	数据	地址	数据	地址	数据	地址	数据	地址	数据	地址	数据
ID-CFI (自动选择) ASO	ID (自动选择) 进入	3	555	AA	2AA	55	(SA) 555	90								
	CFI 进入 (7)	3	(SA) 55	98												
	ID-CFI 读取	1	(SA) RA	RD												
	复位 /ASO 退出 (6)、(14)	1	XXX	F0 或 FF												
安全硅区域指令定义																
安全硅区域 (SSR) ASO	SSR 进入	3	555	AA	2AA	55	(SA) 555	88								
	读取 (5)	1	RA	(SA) RD												
	字编程	4	555	AA	2AA	55	555	A0	PA	PD						
	写入到缓冲区	6	555	AA	2AA	55	SA	25	SA	WC	WBL	PD	WBL	PD		
	将缓冲区中的数据编程到闪存 (确认)	1	SA	29												
	复位 “写缓冲区异常中止” 状态 (10)	3	555	AA	2AA	55	555	F0								
	SSR 退出 (10)	4	555	AA	2AA	55	555	90	XX	00h						
	复位 /ASO 退出 (6)、(14)	1	XXX	F0												
ASP 配置寄存器 (ASPR)	ASP 寄存器进入	3	555	AA	2AA	55	555	40								
	编程	2	XXX	A0	XXX	PD										
	ASPR 读取 (16)	1	0	RD												
	ASPR ASO 退出 (10)	2	XXX	90	XXX	0										
	复位 /ASO 退出 (6)、(14)	1	XXX	F0												
密码保护指令集定义																
密码 ASO 模式	密码 ASO 进入	3	555	AA	2AA	55	555	60								
	编程 (12)	2	XXX	A0	PWA _x	PWD _x										
	读取	4	0	PWD ₀	1	PWD ₁	2	PWD ₂	3	PWD ₃						
	解锁	7	0	25	0	3	0	PWD ₀	1	PWD ₁	2	PWD ₂	3	PWD ₃	0	29
	指令集退出 (11)、(14)	2	XXX	90	XXX	0										
	复位 /ASO 退出 (6)、(14)	1	XXX	F0												

指令定义（表格 3/3）

指令序列			周期	总线周期（注意 1-4）													
				第一个周期		第二个周期		第三个周期		第四个周期		第五个周期		第六个周期		第七个周期	
				地址	数据	地址	数据	地址	数据	地址	数据	地址	数据	地址	数据	地址	数据
非易失性扇区保护指令集定义																	
PPB（非易失性扇区保护）	PPB 进入	3	555	AA	2AA	55	555	C0									
	PPB 编程（15）	2	XXX	A0	SA	0											
	所有 PPB 擦除（15）	2	XXX	80	0	30											
	PPB 读取（15）、（16）	1	SA	RD（0）													
	SA 保护状态（16）、（17）	2	XXX	60	SA	RD											
	指令集退出（11）、（14）	2	XXX	90	XXX	0											
	复位 /ASO 退出（6）、（14）	1	XXX	F0													
全局非易失性扇区保护冻结指令集定义																	
PPB 锁定位	PPB 锁定进入	3	555	AA	2AA	55	555	50									
	PPB 锁定位清除	2	XXX	A0	XXX	0											
	PPB 锁定状态读取（16）	1	XXX	RD（0）													
	指令集退出（11）、（14）	2	XXX	90	XXX	0											
	复位 /ASO 退出（14）	1	XXX	F0													
易失性扇区保护指令集定义																	
DVB（易失性扇区保护） ASO	DVB ASO 进入	3	555	AA	2AA	55	555	E0									
	DVB 设置（15）	2	XXX	A0	SA	0											
	DVB 清除（15）	2	XXX	A0	SA	1											
	DVB 状态读取（16）	1	SA	RD（0）													
	SA 保护状态（15）、（16）、（17）	2	XXX	60	SA	RD											
	指令集退出（11）、（14）	2	XXX	90	XXX	0											
	复位 /ASO 退出（14）	1	XXX	F0													

指令定义注意：

X = 无需关注。

RA = 要读取的存储地址。

RD = 在读取操作期间从位置 RA 读取的数据。

PA = 要编程的存储位置的地址。

PD = 要在位置 PA 编程的数据。

 SA = 所选扇区的地址。256 kB 扇区的地址位 A_{MAX} - A17 以及 4 kB 参数扇区的地址位 A_{MAX} - A11 可以单独选择任何扇区。

WBL = 写入缓冲区位置。地址必须在同一行内。

WC = 字数等于要加载的写入缓冲区位置数量减 1。

PWAx = 密码地址 word0 = 00h、word1 = 01h、word2 = 02h 和 word3 = 03h。

PWDx = 密码数据 word0、word1、word2 和 word3。

注意：

- 所有值均为十六进制数值。所有地址都参考了 16 位字。
- 除了下列周期外，所有总线周期均为写入周期，包括：发生于读取周期期间的读取、ID/CFI 读取（制造商 ID/ 器件 ID）、指示位、安全硅区域读取、SSR 锁定读取以及状态寄存器读取的第二个周期。
- 指令序列中的数据位 DQ15-DQ8 均为“无需关注”（RD、PD、WC 和 PWD 除外）内容。
- 除非需要 SA 或 PA，在解锁和指令周期期间，地址位 A_{MAX}-A11 均为“无需关注”内容。（A_{MAX} 是最高地址引脚）。
- 读取阵列的数据时，不需要解锁或指令周期。
- 在器件处于 ID-CFI（自动选择）模式或者 DQ5 变为高电平（当器件正在提供状态数据时）的情况下，需要通过复位指令返回读取阵列数据的状态。
- 当器件准备好读取阵列数据或者器件处于 ID-CFI（自动选择）模式时，指令有效。
- 在擦除挂起模式下，系统可以对非擦除扇区进行读取和编程 / 暂停编程的操作，或者从该模式进入 ID-CFI ASO 模式。擦除挂起指令仅在扇区擦除操作期间有效。

9. 擦除恢复 / 编程恢复指令仅在擦除挂起 / 编程挂起模式下有效。
10. 检测到器件处于写入到缓冲器异常中止状态后，发出该指令序列以返回到读取模式。请注意，如果进行复位以退出 **ABORT** 状态，则需使用整个指令序列。
11. 退出指令会使器件返回到读取阵列状态。
12. 对于 **PWDx**，每个 “**A0**” 指令只能编程密码的一部分。密码的各个部分必须按连续序列进行编程（**PWD0** - **PWD3**）。
13. 所有 **ASP** 寄存器位都只能编程一次。编程状态 = 0，擦除状态 = 1。持久保护模式锁定位和密码保护模式锁定位不能同时被编程，否则将中止 **ASPR** 寄存器位编程操作，并使器件返回到读取模式。保留给将来使用的 **ASPR** 寄存器位的内容未经定义，它的值可能是 0，也可以是 1。
14. 如果发出任何进入指令，则必须发出退出指令，从而使器件返回到读取模式。
15. 位 0 = 0 表示受保护的状态；位 0 = 1 表示不受保护的状态。位 1 到 15 均为 1。用于 **DYB** 设置、**DYB** 清除或 **PPB** 编程指令的扇区地址可以是扇区内的任意位置；扇区地址的低位均 ‘无需关注’。
16. 读取状态寄存器、**DYB**、**PPB**、**SA** 保护、密码、**POR** 计数器、**ICR**、**ISR**、**VCR**、**NVCR**、**FIDR**、**ASPR**、**PPBL** 读取寄存器时得到的数据仅在器件输出第一个字时有效。如果 **CK/CK#** 在 **CS#** 保持低电平时继续切换状态，随后输出的数据均为未定义。
17. 读取 **SA** 保护状态时得到的数据将通过位 0-3 的值来表示指定的扇区是否是受保护。
18. 位 0 — 表示指定的扇区是否受保护（0 = 受保护，1 = 不受保护）。
19. 位 1 — 使用扇区的 **DYB** 位表示扇区是否受保护（0 = 受保护，1 = 不受保护）。
20. 位 2 — 使用扇区的 **PPB** 位表示扇区是否受保护（0 = 受保护，1 = 不受保护）。
21. 位 4-15 全部为 1。
22. 更小的参数扇区需要添加 **A[16:11]**，将其作为地址的一部分，用于确定擦除和编程指令序列时的目标参数扇区。
23. 使用 **ID**（自动选择）进入指令和 **CFI** 进入指令可以访问同一个 **ID/CFI** 数据集。使用 **ID** 或 **CFI** 进入指令序列后，可以访问 **ID/CFI** 数据集内的所有数据。
24. 仅适用于非 512 Mb 器件。

8. 嵌入式算法性能

嵌入式算法特性

参数	最小值	典型值 (1)	最大值 (2)	单位	注释
扇区擦除时间 (256 KB)	–	930	2900	ms	包括擦除之前的预编程时间 (4)
参数扇区擦除时间 (4 Kbyte)	–	240	725	ms	
芯片擦除时间 (128 Mb)		55	115	s	
芯片擦除时间 (256 Mb)	–	110	231	s	
芯片擦除时间 (512 Mb)	–	220	462	s	
单字编程时间	–	500	1260	µs	字编程指令序列
半页 (16 个字节) 缓冲的编程时间	–	270	1000	µs	缓冲编程指令序列
缓冲编程时间 (全部 512 个字节)	–	475	2000	µs	
擦除挂起 / 擦除恢复 (t_{ESL})	–		50	µs	
编程挂起 / 编程恢复 (t_{PSL})	–		50	µs	
从擦除恢复到下一次擦除挂起的时间 (t_{ERS})	–	100		µs	最少为 60 ns, 但 ≥ 擦除完成所需的典型时长
从编程恢复到下一次编程挂起的时间 (t_{PRS})	–	100		µs	最少为 60 ns, 但 ≥ 编程完成所需的典型的时间长度
空白检查 (256 kB 大小的扇区)	–	15	17	ms	
NOP (每行的编程操作数量)	–		256		工业级温度
	–		32		扩展的工业级温度 半页范围内, 每 8 个字 (16 个字节) 只能实现单个编程操作。
评估擦除状态时间 (t_{EES})	–	70	100	µs	
密码比较时间 (t_{PSWD})	80	100	120	µs	

注意:

- 在以下条件下得到典型的编程和擦除时间: 温度 = 25°C、 V_{CC} = 1.8 V 或 3.0 V、一万次循环、一个棋盘式数据组合。
- 在最坏条件下: 90°C、 V_{CC} = (1.70 V 或 2.7 V)、一万次循环、一个随机数据组合。
- 有效的写入缓冲区规格以 512 字节写入缓冲区操作为基础。
- 在嵌入式擦除算法的预编程步骤, 在扇区和芯片擦除前所有字均编程为 0000h。
- 系统级开销是为执行编程指令的总线循环序列所需要的时间。欲了解指令定义的信息, 请参见第 51 页上的指令定义表。

9. 数据完整性

9.1 擦写次数

编程 / 擦除耐久性

非易失性单元	温度范围	最小值	单位
任意扇区	工业级	100K	编程 - 擦除周期次数
	扩展的工业级	100K	
	扩展型	10K	
配置寄存器	工业级	100K	
	扩展的工业级	100K	
	扩展型	10K	

注意:

1. 数据收集循环限于 10 万次擦写周期。

9.2 数据保持时间

数据保持时间

参数	典型值	单位
擦写周期次数为 1K（或更小）后的数据保持时间（针对每次擦除后一次编程操作，每个半页）	20	年

9.3 器件 ID 和通用闪存接口 (ID-CFI) ASO 映射

ASO 的器件 ID 部分 (字位置 0h 到 0Fh) 提供了器件的制造商 ID、器件 ID 以及基本功能集信息。

欲了解更多信息, 请参见第 34 页上的 [ID-CFI ASO](#)。

ID (自动选择) 地址映射

字地址	数据	说明
(SA) + 0000h	0001h	制造商 ID
(SA) + 0001h	007Eh	器件 ID
(SA) + 0002h	保留	保留
(SA) + 0003h	保留	
(SA) + 0004h	保留	
(SA) + 0005h	保留	
(SA) + 0006h	保留	
(SA) + 0007h	保留	
(SA) + 0008h	保留	
(SA) + 0009h	保留	
(SA) + 000Ah	保留	
(SA) + 000Bh	保留	
(SA) + 000Ch	0005h	低软件位 位 0 — 状态寄存器支持 数值 1 表示支持状态寄存器 数值 0 表示不支持状态寄存器 位 1 — DQ 轮询支持 数值 1 表示支持 DQ 位轮询 数值 0 表示不支持 DQ 位轮询 位 3、位 2 — 指令集支持 数值 11 表示保留 数值 10 表示保留 数值 01 表示 HyperFlash 指令集 数值 00 表示传统指令集 位 4 至位 F — 保留位 = 0
(SA) + 000Dh	保留	高软件位
(SA) + 000Eh	0070h = 512 Mb (电压 = 1.8 V) 006Fh = 512 Mb (电压 = 3.0 V) 0072h = 256 Mb (电压 = 1.8 V) 0071h = 256 Mb (电压 = 3.0 V) 0074h = 128 Mb (电压 = 1.8 V) 0073h = 128 Mb (电压 = 1.8 V)	器件 ID
(SA) + 000Fh	0000h	器件 ID

CFI 查询识别字符串

字地址	数据	说明
(SA) + 0010h	0051h	查询唯一 ASCII 字符串 “QRY”
(SA) + 0011h	0052h	
(SA) + 0012h	0059h	
(SA) + 0013h	0002h	主要 OEM 指令集
(SA) + 0014h	0000h	
(SA) + 0015h	0040h	主扩展表地址
(SA) + 0016h	0000h	

CFI 查询识别字符串

字地址	数据	说明
(SA) + 0017h (SA) + 0018h	0000h 0000h	备用 OEM 指令集 (00h = 不存在)
(SA) + 0019h (SA) + 001Ah	0000h 0000h	备用 OEM 扩展表地址 (00h = 不存在)

CFI 系统接口字符串

字地址	数据	说明
(SA) + 001Bh	0017h (V _{CC} = 1.8 V) 0027h (V _{CC} = 3.0 V)	V _{CC} 最小值 (擦除 / 编程) (D7-D4: V, D3-D0: 100 mV)
(SA) + 001Ch	0019h (V _{CC} = 1.8 V) 0036h (V _{CC} = 3.0 V)	V _{CC} 最大值 (擦除 / 编程) (D7-D4: V, D3-D0: 100 mV)
(SA) + 001Dh	0000h	V _{PP} 最小电压 (00h = 不存在 V _{PP} 引脚)
(SA) + 001Eh	0000h	V _{PP} 最大电压 (00h = 不存在 V _{PP} 引脚)
(SA) + 001Fh	0009h	单字写入的典型超时为 2 ^N μs
(SA) + 0020h	0009h	最大多字节编程的 典型超时为 2 ^N μs (00h = 不支持)
(SA) + 0021h	000Ah	单个块擦除的典型超时时间为 2 ^N ms
(SA) + 0022h	0012h (512 Mb) 0011h (256 Mb) 0010h (128 Mb)	整个芯片擦除的典型超时时间为 2 ^N ms (00h = 不支持)
(SA) + 0023h	0002h	单字编程的最大超时为典型值的 2 ^N 倍
(SA) + 0024h	0002h	缓冲区写入的最大超时为典型值的 2 ^N 倍
(SA) + 0025h	0002h	单个块擦除的最大超时为典型值的 2 ^N 倍
(SA) + 0026h	0002h	整个芯片擦除的最大超时为典型值的 2 ^N 倍 (00h = 不支持)

CFI 器件几何定义

字地址	数据	说明
(SA) + 0027h	001Ah (512 Mb) 0019h (256 Mb) 0018h (128 Mb)	器件容量 = 2 ^N 个字节;
(SA) + 0028h	0000h	闪存器件接口描述 0 = 仅 x8, 1 = 仅 x16, 2 = x8/x16
(SA) + 0029h	0000h	
(SA) + 002Ah	0009h	多字节写入的最多字节数 = 2 ^N
(SA) + 002Bh	0000h	(00 = 不支持)
(SA) + 002Ch	0001h	器件内擦除块区域的数量 1 = 统一器件, 2 = 引导器件
(SA) + 002Dh	请参见说明部分内容	擦除块区域 1 信息 (参见 JEDEC JESD68-01 或 JEP137 规格) 00FFh, 0000h, 0000h, 0004h = 512 Mb (256 x 2 Mb 的块) 00FFh, 0000h, 0000h, 0004h = 512 Mb (256 x 2 Mb 的块) 00FFh, 0000h, 0000h, 0004h = 512 Mb (256 x 2 Mb 的块)
(SA) + 002Eh		
(SA) + 002Fh		
(SA) + 0030h		

CFI 器件几何定义

字地址	数据	说明
(SA) + 0031h	0000h	擦除块区域 2 信息 (参见 JEDEC JESD68-01 或 JEP137 规格)
(SA) + 0032h	0000h	
(SA) + 0033h	0000h	
(SA) + 0034h	0000h	
(SA) + 0035h	0000h	擦除块区域 3 信息 (参见 JEDEC JESD68-01 或 JEP137 规格)
(SA) + 0036h	0000h	
(SA) + 0037h	0000h	
(SA) + 0038h	0000h	
(SA) + 0039h	0000h	擦除块区域 4 信息 (参见 JEDEC JESD68-01 或 JEP137 规格)
(SA) + 003Ah	0000h	
(SA) + 003Bh	0000h	
(SA) + 003Ch	0000h	

CFI 主要厂商特定的扩展查询 (表格 1/2)

字地址	数据	说明
(SA) + 0040h	0050h	查询唯一 ASCII 字符串 “PRI”
(SA) + 0041h	0052h	
(SA) + 0042h	0049h	
(SA) + 0043h	0031h	主版本号, ASCII
(SA) + 0044h	0035h	次版本号, ASCII
(SA) + 0045h	001Ch	地址敏感解锁 (位 1-0) 00b = 需要, 01b = 不需要 工艺技术 (位 5-2) 0000b = 0.23 μm 浮门 0001b = 0.17 μm 浮门 0010b = 0.23 μm MirrorBit 0011b = 0.13 μm 浮门 0100b = 0.11 μm MirrorBit 0101b = 0.09 μm 浮门 0110b = 0.09 μm MirrorBit 0111b = 0.065 μm MirrorBit Eclipse 1000b = 0.065 μm MirrorBit 1001b = 0.045 μm MirrorBit
(SA) + 0046h	0002h	擦除挂起 0 = 不支持 1 = 只读 2 = 读取和写入
(SA) + 0047h	0001h	扇区保护 00 = 不支持 X = 最小组中的扇区数量
(SA) + 0048h	0000h	临时取消扇区保护 00 = 不支持 01 = 支持
(SA) + 0049h	0008h	扇区保护 / 不保护方案 04 = 高电压方法 05 = 软件指令锁定方法 08 = 高级扇区保护方法
(SA) + 004Ah	0000h	并发操作 00 = 不支持 X = Bank 数量

CFI 主要厂商特定的扩展查询 (表格 2/2)

字地址	数据	说明
(SA) + 004Bh	0001h	突发模式类型 00 = 不支持 01 = 支持
(SA) + 004Ch	0000h	页面读取模式类型 00 = 不支持 01 = 4 字页面 02 = 8 字页面 03 = 16 字页面
(SA) + 004Dh	0000h	ACC (加速) 最小供电电压 00 = 不支持 D7-D4: V D3-D0: 100 mV
(SA) + 004Eh	0000h	ACC (加速) 最大供电电压 00 = 不支持 D7-D4: V D3-D0: 100 mV
(SA) + 004Fh	0000h	WP# 保护 00h = 闪存设备无 WP 保护 (无引导) 01h = 顶部和底部的八个 8 kB 大小的扇区均受 WP 保护 (双引导) 02h = 底部引导设备, 有 WP 保护 (底部引导) 03h = 顶部引导设备, 有 WP 保护 (顶部引导) 04h = 统一、底部 WP 保护 (统一底部引导) 05h = 统一、顶部 WP 保护 (统一顶部引导) 06h = 所有扇区都受 WP 保护 07h = 统一、顶部或底部 WP 保护
(SA) + 0050h	0001h	编程挂起 00 = 不支持 01 = 支持
(SA) + 0051h	0000h	解锁省略 00 = 不支持 01 = 支持
(SA) + 0052h	000Ah	安全硅扇区 (用户 OTP 区域 = 1024 B) 的容量为 2^N (字节)
(SA) + 0053h	008Dh	软件特性 位 0: 状态寄存器轮询 (1 = 支持, 0 = 不支持) 位 1: DQ 轮询 (1 = 支持, 0 = 不支持) 位 2: 新编程挂起 / 恢复指令 (1 = 支持, 0 = 不支持) 位 3: 字编程 (1 = 支持, 0 = 不支持) 位 4: 位段编程 (1 = 支持, 0 = 不支持) 位 5: 自动检测编程 (1 = 支持, 0 = 不支持) 位 6: RFU (留给将来使用) 位 7: 每行多次写入 (1 = 支持, 0 = 不支持)
(SA) + 0054h	0005h	页面大小 = 2^N 个字节
(SA) + 0055h	0006h	最长的擦除挂起超时时间 < 2^N (μ s)
(SA) + 0056h	0006h	最长的编程挂起超时时间 < 2^N (μ s)
(SA) + 0057h 到 (SA) + 0077h	FFFFh	保留供将来使用
(SA) + 0078h	0006h	最长的嵌入式硬件复位超时时间 < 2^N (μ s) 通过复位引脚复位
(SA) + 0079h	0009h	最长的非嵌入式硬件复位超时时间 < 2^N (μ s) 上电复位

硬件接口

欲了解 HyperFlash 存储器产品中 HyperBus 硬件接口的总体概述，请参考 HyperBus 规范中的内容。下面部分将介绍硬件接口中涉及到 HyperFlash 器件的相关信息。

10. 接口状态

10.1 硬件复位

- 终止任何正在执行的操作
- 当 RESET# 处于低电平状态时，DQ[7:0] 进入高阻态。
- 退出所有 ASO
- 使所有输出三态化
- 复位状态寄存器
- 复位 EAC 使器件进入待机状态
- CE# 在复位操作期间 (t_{RPH}) 被忽略。
- 为符合复位电流规格 (I_{CC5})，CE# 必须保持高电平状态。

为确保数据完整性，一旦器件完成硬件复位过程，需要重新启动被中断的所有非易失性操作。

10.2 关闭电源（硬件数据保护）

当内核供电电压 (V_{CC}) 下降至低于锁定电压 (V_{LKO}) 时，存储器被视为断电。当 V_{CC} 低于 V_{LKO} 时，整个存储器阵列受保护，不能进行写入或擦除操作。这样可防止在电源转换期间存储内容发生篡改。在电源切换至断电期间， V_{CCQ} 应保持在不大于 V_{CC} 的状态。

如果 V_{CC} 降至 V_{RST} （最小值）以下，那么将 V_{RST} （最小值）上拉到 V_{CC} （最小值），器件进入上电复位接口状态，EAC 开始冷复位嵌入式算法。

10.3 节能模式

10.3.1 有效时钟停止

在进行数据读取操作传输期间，使用工作时钟停止模式会使器件接口的电流消耗下降到 I_{CC6} 。在读取操作期间进行数据输出时，如果 CK/CK# 在 $t_{ACC} + 30\text{ ns}$ 时间内保持稳定，则器件会自动进入该模式。在工作时钟停止模式中，输出数据被锁存，并且始终被驱动到数据总线上。第 63 页上的 11.2 章节，*直流特性（与 CMOS 兼容）* 中的 I_{CC6} 是指工作时钟停止模式的当前规范。

主机系统时钟停止暂停数据传输时，进入该模式有助于降低所消耗的电流。即使在这些延长的数据传输周期内 CE# 可能为低电平，存储器件主机接口经过 $t_{ACC} + 30\text{ ns}$ 时间后也会切换到工作时钟停止电流。这样，停止进行数据传输时，器件会进入低电流消耗模式。切换时钟重新启动数据传输时，会恢复有效的读取电流。

不过，EAC 的运行与主机接口的工作时钟停止模式无关，它会在嵌入式算法执行期间继续消耗电流。只有主机接口和 EAC 都处于各自的待机状态时，才能达到工作时钟停止电流等级。

10.3.2 深度掉电

在深度掉电（DPD）模式下，消耗的电流最低。器件处于空闲状态（并非处于 ASO 状态）时，必须进入 DPD 模式。此外，器件在擦除挂起或编程挂起状态时，也可以进入 DPD 模式。使用 DPD 进入指令序列进入 DPD 模式（参见第 51 页上的指令定义表）。

在任何读取或写入操作期间，通过确认 CS# 信号可以退出 DPD 模式（CS# 在 t_{DPDCSL} 时间内为低电平）。在 t_{DPDOUT} 期间，器件会忽略指令序列（不执行读取和写入操作），并在尝试进行读取时 RWDS 不被切换。使用‘虚拟’写入操作退出 DPD 模式是首选方法。

驱动 RESET# 输入为低电平（至少在 t_{RP} 时间）也可以使器件退出 DPD 模式。器件需要经过 t_{DPDOUT} 时间才能返回空闲状态。

退出 DPD 模式时，器件返回到上电复位后退出该模式的默认设置。

进入 / 退出 DPD 模式的时序

符号	参数	最小值	最大值	单位
t_{DPDIN}	从进入深度掉电模式（向寄存器 CR[15] 位写入零）到电流下降为 DPD 的时间	10	–	μs
t_{DPDCSL}	使器件退出 DPD 模式的 CS# 低电平时间	25	–	ns
t_{DPDOUT}	从深度掉电模式到闲置唤醒的时间	–	300	μs

图 10.1 深度掉电进入时序

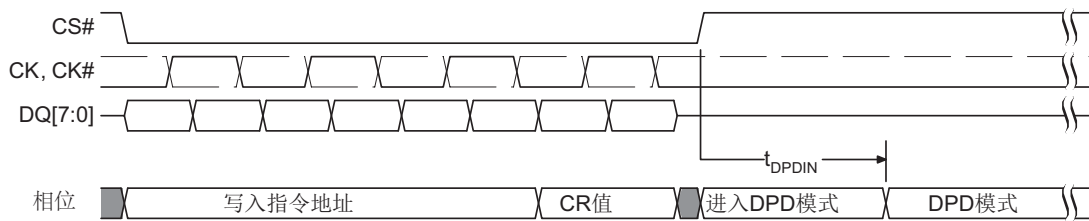


图 10.2 深度掉电 CS# 退出时序

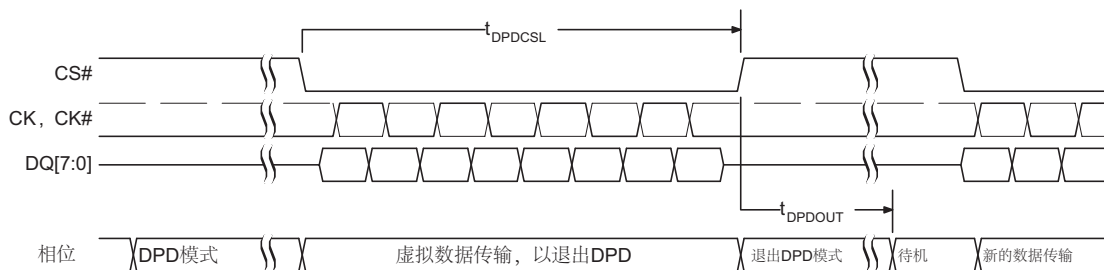
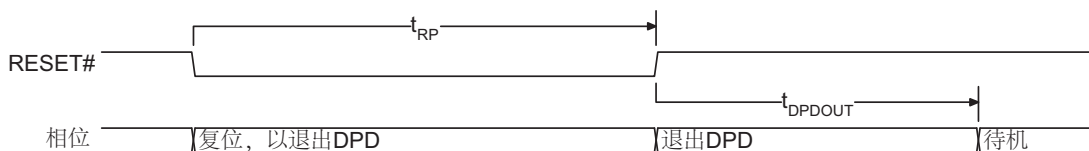


图 10.3 深度掉电 RESET# 退出时序



11. 电气规范

下面的内容介绍了电气规范中涉及到 HyperFlash 器件的相关信息。

11.1 最大绝对额定值

通电时的环境温度 -65°C 到 $+125^{\circ}\text{C}$

11.2 直流特性（与 CMOS 兼容）

直流特性（CMOS 兼容）

参数	说明	测试条件	最小值	典型值 (9)	最大值	单位
I_{CC1}	V_{CC} 有效读取电流 (只针对内核电流, IO 开关电流除外)	$CS\# = V_{IL}$, 166 MHz, $V_{CC} = 1.95\text{ V}$		130	180	mA
		$CS\# = V_{IL}$, 100 MHz, $V_{CC} = 3.6\text{ V}$		80	100	mA
I_{IO1}	V_{CC} 读取时有效电流	$CS\# = V_{IL}$, 166 MHz, $V_{CCQ} = 1.95\text{ V}$, $C_{LOAD} = 20\text{ pF}$		80	100	mA
		$CS\# = V_{IL}$, 100 MHz, $V_{CCQ} = 1.95\text{ V}$, $C_{LOAD} = 20\text{ pF}$		80	100	mA
I_{CC3P}	V_{CC} 有效编程电流 (1), (2)	$V_{CC} = V_{CC}$ 最大值		60	100	mA
I_{CC3E}	V_{CC} 有效擦除电流 (1), (2)	$V_{CC} = V_{CC}$ 最大值		60	100	mA
I_{CC4I}	工业级温度 (-40°C 到 $+85^{\circ}\text{C}$) 下的 V_{CC} 待机电流	$CS\#$, $RESET\# = V_{CC}$, $V_{CC} = V_{CC}$ 最大值		25	100	μA
I_{CC4IC}	在扩展的工业级 (汽车 — 车仓内) 温度 (40°C 到 $+105^{\circ}\text{C}$) 下 V_{CC} 待机电流	$CS\#$, $RESET\# = V_{CC}$, $V_{CC} = V_{CC}$ 最大值		25	300	μA
I_{CC4E}	扩展温度范围 (-40°C 到 $+125^{\circ}\text{C}$) 的 V_{CC} 待机电流	$CS\#$, $RESET\# = V_{CC}$, $V_{CC} = V_{CC}$ 最大值		25	300	μA
I_{CC5}	V_{CC} 复位电流 (5)	$CS\# = V_{IH}$, $RESET\# = V_{SS}$, $V_{CC} = V_{CC}$ 最大值		10	20	mA
I_{CC6}	工作时钟停止模式 (3)	$V_{IH} = V_{CC}$, $V_{IL} = V_{SS}$, $V_{CC} = 1.95\text{ V}$		6	12	mA
		$V_{IH} = V_{CC}$, $V_{IL} = V_{SS}$, $V_{CC} = 3.6\text{ V}$		6	12	mA
I_{CC7}	上电 (4) 期间的 V_{CC} 电流	$CS\# = X$, $V_{CC} = V_{CC}$ 最大值,		80	100	mA
I_{DPD}	深度掉电模式的电流, 512 Mb, 25°C	$CS\#$, $RESET\#$ $V_{CC} = V_{CC}$ 最大值		8	15	μA
	深度掉电模式的电流, 512 Mb, 25°C			30	50	μA
	深度掉电模式的电流, 512 Mb, 105°C			95	150	μA
	深度掉电模式下的电流 (所有其他容量), 25°C			3	6	μA
	深度掉电模式下的电流 (所有其他容量), 85°C			4	10	μA
	深度掉电模式下的电流 (所有其他容量), 105°C			5	15	μA
V_{OL}	输出低电压	$I_{OL} = 100\text{ }\mu\text{A}$ (DQ7-DQ0) $I_{OL} = 2\text{ mA}$ (INT# 和 RSTO#)			$0.15 \times V_{CCQ}$	V

注意:

- 在执行嵌入式算法期间, I_{CC} 有效。
- 并非 100% 经过了测试。
- 如果 $CK/CK\#$ 信号在 $t_{ACC} + 30\text{ ns}$ 时间内保持稳定, 则工作时钟停止模式会启用低功耗模式。
- $V_{CCQ} = 1.70\text{ V} \sim 1.95\text{ V}$ 或 $2.7\text{ V} \sim 3.6\text{ V}$ 。
- $V_{CC} = V_{CCQ} = 1.8\text{ V}$ 或 $V_{CC} = V_{CCQ} = 3.0\text{ V}$ 。
- 在加电期间, 有电流波峰需求, 要求系统能够提供该电流, 以保证该部分初始化的正确性。
- 如果复位开始时嵌入式操作正在执行中, 则电流消耗将保持在嵌入式操作规范, 直至嵌入式操作被复位停止。如果复位开始时没有执行任何嵌入式操作, 或者在嵌入式操作停止之后, 在 t_{RPH} 剩余期间消耗的电流为 I_{CC7} 。在 t_{RPH} 结束时, 器件进入待机模式, 直到进行下一个读取或写入操作为止。
- 推荐的 INT# 和 RSTO# 输出上拉电阻范围为 $5\text{ K}\Omega$ - $10\text{ K}\Omega$ 。
- 在下面条件下测得 I_{CC} 典型值: $t_{AI} = 25^{\circ}\text{C}$ 和 $V_{CC} = V_{CCQ} = 1.8\text{ V}$ 或 3.0 V (不适用于温度为 85°C 和 105°C 条件下的 I_{DPD})。

11.2.1 电容特性

1.8 V 电容特性

说明	参数	最小值	最大值	单位
输入电容 (CK、CK#、CS#)	CI	1.5	3.0	pF
输入电容差值 (CK、CK#)	CID		0.25	pF
输出电容 (RWDS)	CO	1.5	4.5	pF
I/O 引脚电容 (DQx)	CIO	1.5	4.5	pF
I/O 引脚电容差值 (DQx)	CIOD	—	0.5	pF
INT#、RSTO# 引脚电容	COP	—	8.0	pF

注意:

1. 这些值仅由设计保证，并仅进行过示例测试。
2. 接触电容值是通过 JEP147 过程获得的，该过程使用了向量网络分析仪进行电容测量。使用 V_{CC} 和 V_{CCQ} ，所有其他引脚（正在测试的引脚除外）均处于悬空状态。DQ 应处于高阻抗状态。
3. 请注意，要求 CK、CK#、RWDS 和 DQx 引脚的电容相同，从而保证系统中信号传输时间匹配。CS# 的电容没有上述引脚的电容重要，因为 CS# 变为有效状态（低电平）与数据出现在 DQ 总线上之间的时序不是至关重要。

3.0 V 电容特性

说明	参数	最小值	最大值	单位
输入电容 (CK、CS#)	CI	1.5	3.0	pF
输出电容 (RWDS)	CO	1.5	6.5	pF
I/O 引脚电容 (DQx)	CIO	1.5	6.5	pF
I/O 引脚电容差值 (DQx)	CIOD	—	0.5	pF
INT#、RSTO# 引脚电容	COP	—	8.0	pF

注意:

1. 这些值仅通过设计保证，并只经过示例测试。
2. 接触电容值是通过 JEP147 过程获得的，该过程使用了向量网络分析仪进行电容测量。使用 V_{CC} 和 V_{CCQ} ，所有其他引脚（正在测试的引脚除外）均处于悬空状态。DQ 应处于高阻抗状态。
3. 请注意，要求 CK、RWDS 和 DQx 引脚的电容相同，从而保证系统中信号传输时间匹配。CS# 的电容没有上述引脚的电容重要，因为 CS# 变为有效状态（低电平）与数据出现在 DQ 总线上之间的时序不很重要。

11.3 加电和掉电

当内核供电电压 (V_{CC}) 下降至低于 V_{CC} 锁定电压 (V_{LKO}) 时, 存储器被视为断电。当 V_{CC} 低于 V_{LKO} 时, 整个存储器阵列受到保护, 不能进行写入或擦除操作。这样可防止在电源转换期间存储内容发生篡改。电源转换降到 V_{SS} 以下, V_{CCQ} 需要保持在不高于 V_{CC} 的电压电平。

如果 V_{CC} 降至 V_{CC} 复位 (最小值) (V_{RST}) 以下, 那么将上述 V_{RST} 设置为 V_{CC} (最小值), 器件进入上电复位接口状态, EAC 开始冷复位嵌入式算法。

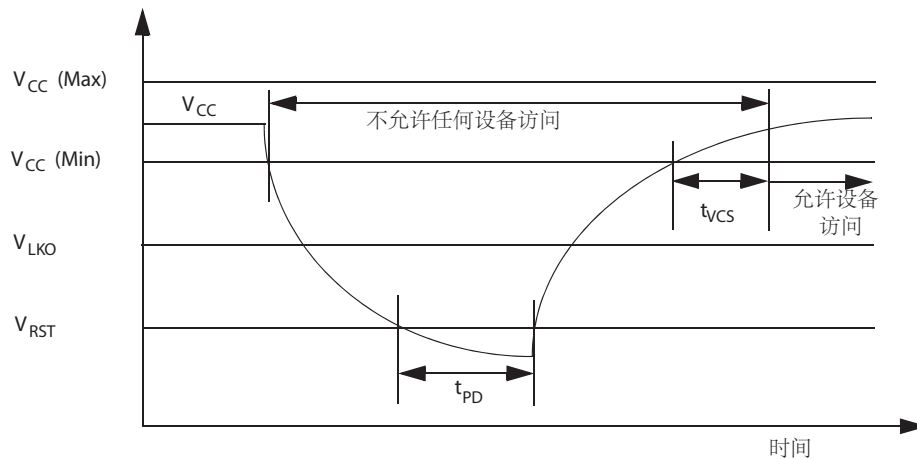
V_{CC} 不能小于 V_{CCQ} ($V_{CC} \geq V_{CCQ}$)。

V_{CC} 和 V_{CCQ} 增加到 V_{CC} 最小阈值以上, 并一直保持该状态, 经过 t_{VCS} 延迟时间后器件会忽略所有输入。在 t_{VCS} 期间, 器件会执行上电复位操作。

在掉电期间或当电压降至 V_{LKO} 以下, V_{CC} 电压必须保证低于 V_{RST} 的时间为 t_{PD} , 从而在这段时间内能够正常初始化。此时, V_{CC} 和 V_{CCQ} 再次上升到其工作范围。请参见第 65 页上的图 11.1, 掉电或电压下降。如果在电压下降过程中, V_{CC} 保持为高于 V_{LKO} 的状态, 那么当 V_{CC} 再次超过 V_{CC} 最小值时, 器件将被初始化, 并正常运行。如果该部分因初始化出错而被锁定, 可以使用软件复位来正确初始化该部分。

为稳定 V_{CC} 和 V_{CCQ} 电源, 必须采取正常的预防措施进行电源退耦。系统中的每个器件都应配备 V_{CC} 和 V_{CCQ} 电源, 由封装连接附近大小合适的电容器进行去耦 (此电容器通常为 $0.1 \mu F$)。

图 11.1 掉电或电压下降



1.8 V 上电 / 断电时的电压和时序

符号	参数	最小值	最大值	单位
V_{CC}	V_{CC} 电源	1.7	1.95	V
V_{LKO}	V_{CC} 截止电压, 低于该电压时将需要进行重新初始化操作	1.5	—	V
V_{RST}	保证初始化能够发生的 V_{CC} 低电压	0.5	—	V
t_{VCS}	从 V_{CC} 和 $V_{CCQ} \geq$ 最小值到开始第一次访问的时间 从 RESET# 低电平 - 高电平转换到开始第一次访问的时间 (V_{CC} 和 $V_{CCQ} \geq$ 最小值)	—	300	μs
t_{PD}	$V_{CC} \leq V_{RST}$ 的时间	10	—	μs

注意:

1. V_{CC} 的升降速率可能是非线性的。

3.0 V 上电 / 断电时的电压和时序

符号	参数	最小值	最大值	单位
V_{CC}	V_{CC} 电源	2.7	3.6	V
V_{LKO}	V_{CC} 截止电压, 低于该电压时将需要进行重初始化操作	2.4	—	V
V_{RST}	保证初始化能够发生的 V_{CC} 低电压	0.7	—	V
t_{VCS}	从 V_{CC} 和 $V_{CCQ} \geq$ 最小值到开始第一次访问的时间 从 $RESET\#$ 低电平 - 高电平转换到开始第一次访问的时间 (V_{CC} 和 $V_{CCQ} \geq$ 最小值)	—	300	μs
t_{PD}	$V_{CC} \leq V_{RST}$ 的时间	10	—	μs

注意:

1. V_{CC} 的升降速率可能是非线性的。

11.3.1 加电（冷）复位（POR）

第一次加电时, 电源电压低于 V_{LKO} , 然后逐渐上升, 以达到运行范围的最小值, 内部器件配置和冷复位操作被启动。在 POR 操作 (t_{VCS}) 期间和在由用户延长的 $RSTO\#$ 低电平状态时间内, $RESET\#$ 和 $CS\#$ 被忽略。器件处于 POR 状态或 $RSTO\#$ 为低电平时, 指令序列被锁定。该阶段期间: 不能选择器件、不接受任何指令、不驱动输出 ($RSTO\#$ 除外)。在此 POR 期间, $RESET\#$ 低电平是可选的。如果在 POR 期间将 $RESET\#$ 驱动为低电平, 它必须满足硬件复位参数 t_{RP} 和 t_{RPH} 。这样, POR 操作会在 t_{VCS} 和 t_{RPH} 结束后完成。如果 $RESET\#$ 在 t_{VCS} 期间为低电平, 它在 t_{VCS} 结束时可能保持该状态, 从而使器件保持硬件复位状态。如果 $RESET\#$ 在 t_{VCS} 结束时为高电平, 器件将进入待机状态。 $RSTO\#$ 低电平状态结束前, $CS\#$ 必须变为 V_{IH} 。

在冷复位期间, 器件消耗的电流为 I_{CC7} 。如果 $CS\#$ 在 t_{VCS} 期间一直为低电平, 器件在 t_{VCS} 时间内的消耗的电流可能超过典型 POR 电流, 但不会超过最大电流值。 $CS\#$ 的电平不会影响冷复位 EA。

如果 POR 在 t_{VCS} 结束时尚未正确完成, 以后转换到硬件复位状态时会导致转换到加电复位接口状态, 并会启动冷复位嵌入式算法。这样可确保器件能够完成冷复位, 即使系统一些部分的加电电压升高而导致 POR 不能正确启动或完成。

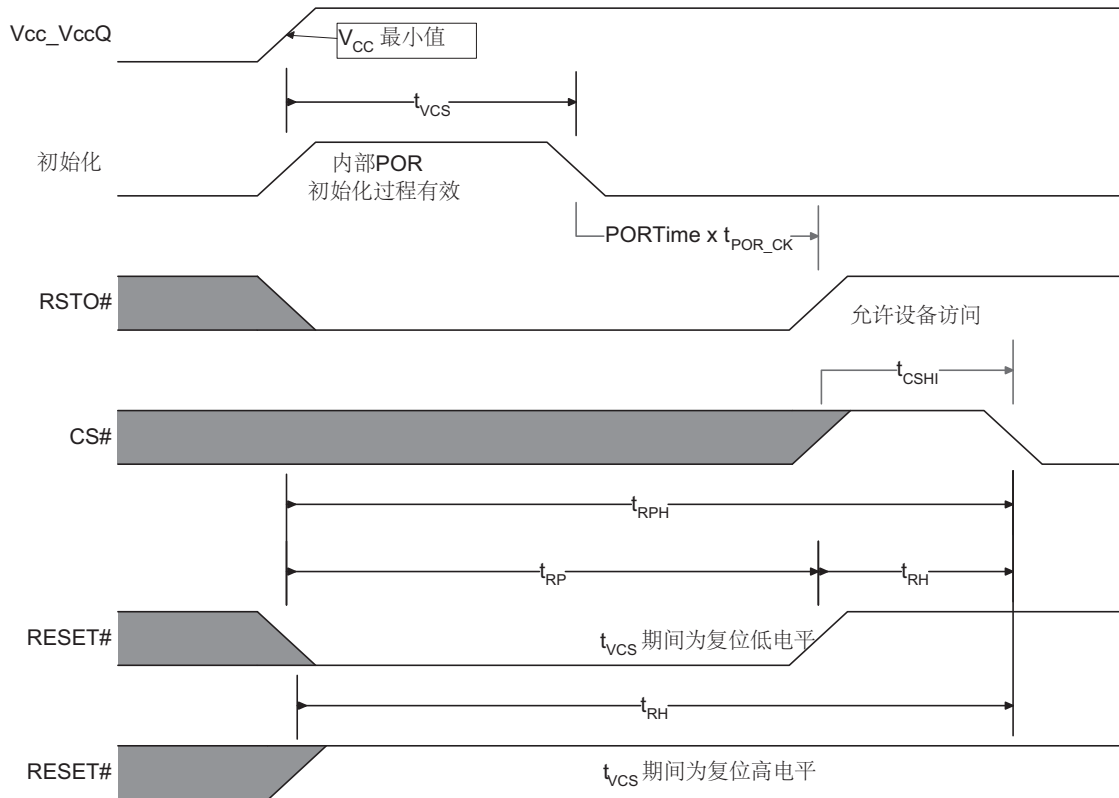
$RSTO\#$ 是一个开漏输出信号, 用于指示器件中发生 (POR) 事件的时间。该信号可作为系统级复位信号。内部 POR 事件完毕后, 如果用户定义的超时时间结束, $RSTO\#$ 信号将从低电平切换到阻抗高状态。成功切换到高阻抗状态后, 外部上拉电阻会使 $RSTO\#$ 变为高电平, 器件将立即进入空闲状态。 $RSTO\#$ 为低电平时, 器件不接受任何指令。

如果用户想要延长 $RSTO\#$ 时间, 使其超过 POR (t_{VCS}) 时间, 那么必须编程非易失性 PORTIME 寄存器。该寄存器的默认值 (FFFFh) 提供了零延时。 t_{VCS} 时间结束时, $RSTO\#$ 信号将返回到高阻抗状态。将编程到 16 位 PORTIME 寄存器中的值乘以 t_{POR_CK} (参见第 66 页上的用户 POR 扩展时钟时序表), 用于定义 $RSTO\#$ 脉冲超过 t_{VCS} 的时间。持续到 $RSTO\#$ 被确认的所编程扩展时间等于 PORTIME 寄存器中所编程的值加一个时钟周期。 PORTIME 寄存器是一个 OTP, 一旦被编程, 并不能尝试进行后续编程。

用户 POR 扩展时钟时序

参数	符号	最小值	最大值	单位
POR 延长时钟周期	t_{POR_CK}	25	42	μs

请注意, V_{CC} 小于 V_{CC} (最小值) 时, $RSTO\#$ 和 $INT\#$ 输出都是不定义的内容。当达到 V_{CC} (最小值) 时, $INT\#$ 输出将进入高阻抗状态。达到 V_{CC} (最小值) 后, 经过 t_{VCS} 以及用户定义的 POR 扩展时间后, $RSTO\#$ 输出将从低电平转换为高阻抗状态。

图 11.2 上电复位信号图

注意:

1. 要求 V_{CCQ} 和 V_{CC} 电压相同。
2. $PORTime$ 是用户编程的配置寄存器，满足 t_{VCS} 时间后确认 $RSTO\#$ 。第 50 页上的非易失性配置寄存器表中介绍了 $PORTime$ 。
3. $3 \cdot t_{POR_CK}$ 是用于生成 $RSTO\#$ 扩展时间的内部（片上）时钟周期。第 66 页上的用户 POR 扩展时钟时序表中对 t_{POR_CK} 进行了相关介绍。

11.3.2 硬件（暖）复位

RESET# 输入提供一种硬件复位方法，可使器件返回待机状态。RESET# 为低电平时，不能指令指令序列和读取操作。器件处于复位状态时，指令序列被锁定。

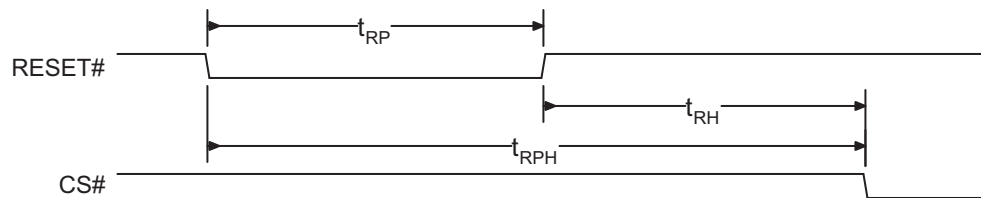
在硬件复位期间，器件消耗的电流为 I_{CC5} 。RESET# 保持为 V_{SS} 电压时，器件消耗的电流是 CMOS 待机电流 (I_{CC4})。如果 RESET# 保持为 V_{IL} 而不是 V_{SS} ，则待机电流更大。

硬件复位使非易失性配置寄存器 (NVCR) 对总线配置进行定义。请参见第 68 页上的图 11.3，硬件复位时序图。

器件完成 POR 并进入待机状态后，以后转换到硬件复位状态时将启动暖复位嵌入式算法。暖复位的时长比冷复位的时长小很多，它只需要几十 μ (t_{RPH}) 便能完成。在暖复位 EA 期间，所有正在执行的嵌入式算法都停止，EAC 返回其 POR 状态，而不会从非易失性存储器重新加载 EAC 算法。暖复位 EA 完成后，如果 RESET# 保持低电平，接口将保持在硬件复位状态。如果 RESET# 返回高电平，接口将转换到待机状态。如果 RESET# 在暖复位 EA 结束时为高电平，接口将直接转换到待机状态。

如果在 t_{VCS} 结束时尚未正确完成 POR，以后转换到硬件复位状态时将导致转换到加电复位接口状态，并会启动冷复位嵌入式算法。这样可确保器件能够完成冷复位，即使系统一些部分的加电电压升高导致 POR 不能正确启动或完成。

图 11.3 硬件复位时序图



上电复位参数

参数	说明	限制	时间	单位
t_{VCS}	从 V_{CC} 建立时间到开始第一次访问 (1) 的时间	最小值	300	μs
t_{RPH}	RESET# 为低到 CS# 为低的时间	最小值	30	μs
t_{RP}	RESET# 脉冲宽度	最小值	200	ns
t_{RH}	RESET# 为高电平到 CS# 为低电平的时间	最小值	150	ns
t_{PD}	$V_{CC} \leq V_{RST}$ 的时间	最小值	1	μs
t_{CSHI}	各操作之间的芯片选择为高电平的时间	最小值	6.0	ns

注意:

1. 在上电复位时间 (t_{VCS}) 内，不允许执行总线操作 (读和写操作)。
2. 时序是在下面时间内测得的：从 V_{CC} 达到 V_{CC} (最小值) 到复位时的 V_{IH} 和 CS# 上的 V_{IL} 。
3. RESET# 在 POR 期间为低电平是可选的。如果在 POR 期间确认 RESET， t_{RPH} 和 t_{VCS} 两者中的较长的时间确定 CS# 何时变为低电平。如果 RESET# 在 t_{VCS} 结束后变为低电平， t_{RPH} 从 t_{VCS} 结束时开始计算。CS# 变为低电平前，RESET 必须在 t_{RH} 时间内保持为高电平。
4. V_{CC} 的升降速率可能是非线性的。
5. $t_{RP} + t_{RH}$ 的总和不能小于 t_{RPH} 。

可以通过硬件复位退出 DPD 模式。此外，驱动 RESET# 输入为低电平 (至少在 t_{RP} 时间) 也可以使器件退出 DPD 模式。器件需要经过 t_{DPDOUT} 时间后才能返回空闲状态。退出 DPD 模式时，器件返回到上电复位后退出该模式的默认设置。请参见第 62 页上的 10.3.2 章节，深度掉电。

12. 时序规范

欲了解 HyperBus 接口通用时序规范的概述，请参见 HyperBus 规范。下面部分介绍了时序规范中有关 HyperFlash 器件的内容。

12.1 交流电特性

HyperFlash 1.8 V 的专用读取时序

参数	符号	166 MHz		单位
		最小值	最大值	
读取初始访问时间	t_{ACC}	—	96	ns
芯片选择有效到 RWDS 有效（低电平）的时间	t_{DSV}	—	8	ns

注意：

1. 采样值，未经过 100% 测试。

HyperFlash 3.0 V 的专用读取时序

参数	符号	100 MHz		单位
		最小值	最大值	
读取初始访问时间	t_{ACC}	—	96	ns

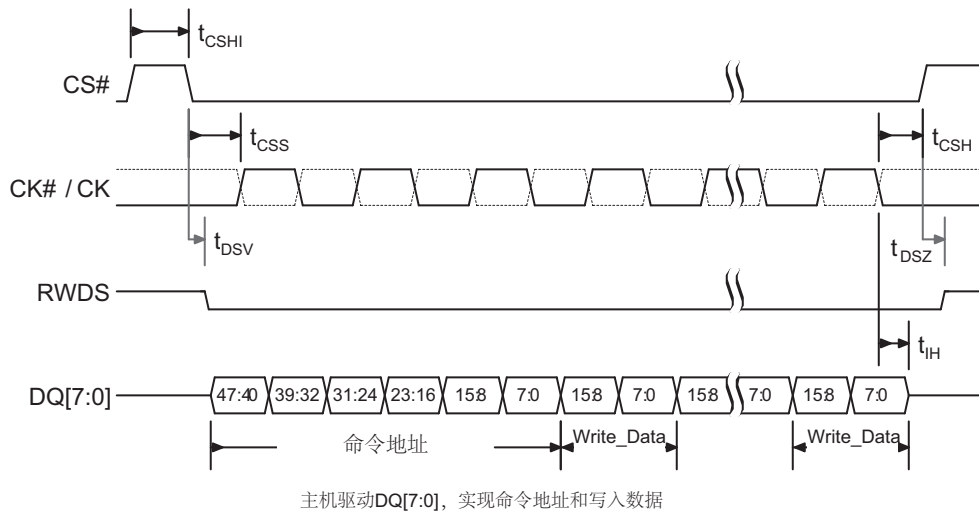
注意：

1. 采样值，未经过 100% 测试。

没有 HyperFlash 专用写入时序。

12.2 字突发数据加载的字编程

图 12.1 字编程指令期间多字加载的突发写入时序图



注意：

1. 必须将 CK 设置为低电平，并将 CK# 设置为高电平，这样才能启动操作。必须将 CS# 返回到高电平状态，才能启动一个新操作。
2. CS# 为低电平时，HyperFlash 存储器在写入期间驱动 RWDS 为低电平。
3. 在 ASO 状态中，不能执行突发写入操作。
4. 只有在字编程指令期间加载多自数据时，才能执行突发写入操作。
5. 只能执行线性突发写入操作。不支持回卷突发写入功能。
6. CK# 仅适用于 1.8 V 器件。3 V 器件使用一个单端时钟输入。

字编程指令期间多字加载的突发写入时序图

参数	符号	50 MHz (2)		单位
		最小值	最大值	
执行突发写入的操作频率			50	MHz
芯片选择建立到下一个时钟上升沿的时间	t_{CSS}	3	—	ns
芯片选择有效到 RWDS 有效（低电平）的时间	t_{DSV}	—	8	ns
输入建立时间	t_{IS}	1.0	—	ns
输入保持时间	t_{IH}	1.0	—	ns
时钟下降沿后的芯片选择保持时间	t_{CSH}	0	—	ns
芯片选择无效到 RWDS 为高阻态的时间	t_{DSZ}	—	6	ns
各操作之间的芯片选择为高电平的时间	t_{CSHI}	10.0	—	ns

注意：

1. 采样值，未经过 100% 测试。
2. 只有在字编程指令期间使用突发写操作时，才需要 50 MHz 的时序。

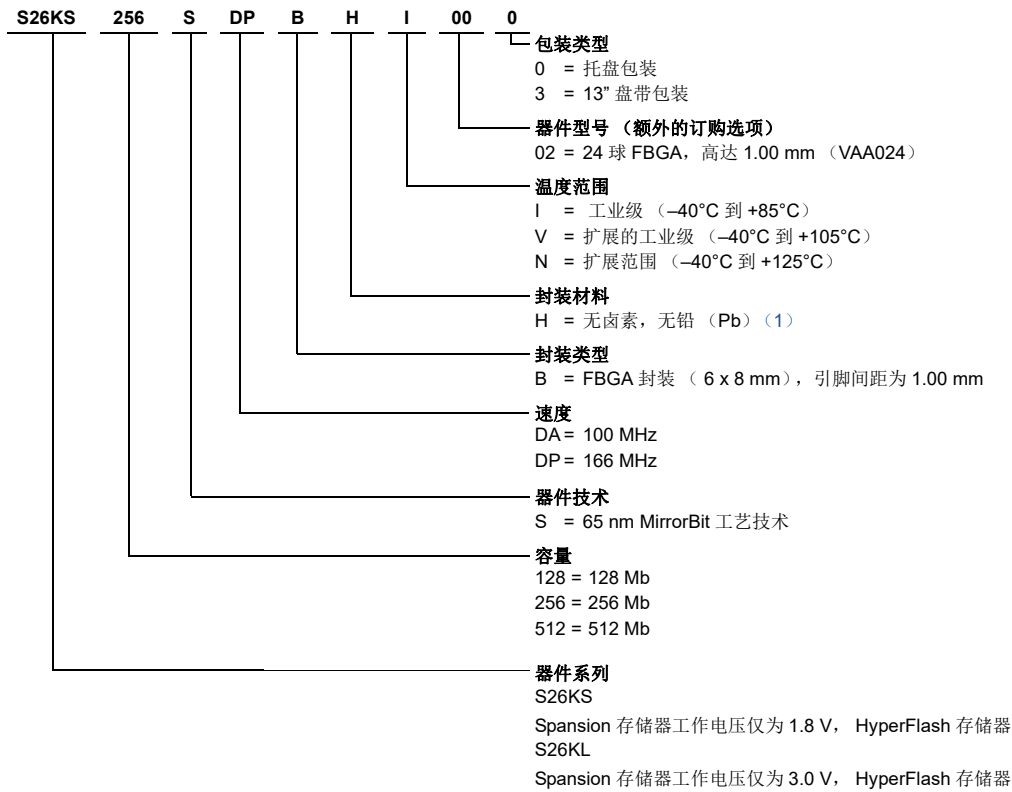
13. 物理接口

欲了解封装和 6 x 8 x 1 mm (VAA024) 物理封装图的信息，请参见 HyperBus 规范。

14. 订购信息

14.1 订购器件编码

下面介绍了订购器件编码的有效组合：



注意：

1. 无卤定义符合 IEC 61249-2-21 规范。

14.2 有效组合

有效组合表列出了计划批量供应的配置情况。该表会随着新组合的推出而不断更新。如要确认特定组合的可用性并了解最新推出的组合，请咨询您当地的销售代表。

有效组合

器件编号	速度选项	封装和材料	温度范围	型号	封装类型	订购器件编码 (x = 封装类型)	封装标识
S26KL512S	DA	BH	I、V	02	0、3	S26KL512SDABHI02x S26KL512SDABHV02x	6KL512SDAHI02 6KL512SDAHV02
S26KL256S	DA	BH	I、V	02	0、3	S26KL256SDABHI02x S26KL256SDABHV02x	6KL256SDAHI02 6KL256SDAHV02
S26KL128S	DA	BH	I、V	02	0、3	S26KL128SDABHI02x S26KL128SDABHV02x	6KL128SDAHI02 6KL128SDAHV02
S26KS512S	DP	BH	I、V	02	0、3	S26KS512SDPBHI02x S26KS512SDPBHV02x	6KS512SDPHI02 6KS512SDPHV02
S26KS256S	DP	BH	I、V	02	0、3	S26KS256SDPBHI02x S26KS256SDPBHV02x	6KS256SDPHI02 6KS256SDPHV02
S26KS128S	DP	BH	I、V	02	0、3	S26KS128SDPBHI02x S26KS128SDPBHV02x	6KS128SDPHI02 6KS128SDPHV02

注意：

1. FBGA 封装标识省略了订购器件编码前面的“S2”、封装类型特征以及包装类型特征。

15. 文档修订记录

文档历史页面

文档标题: S26KL512S/S26KS512S/S26KL256S/S26KS256S/S26KL128S/S26KS128S, 512-Mbit (64 Mbyte)/256-Mbit (32 Mbyte)/128-Mbit (16 Mbyte), 1.8 V/3.0 V, HyperFlash™ 产品系列 文档编号: 001-99456				
版本	ECN 编号	变更者	提交日期	变更说明
**	4889804	YOQI	08/24/2015	本文档版本号为 Rev**, 译自英文版 001-99198 Rev*C。
*A	4965361	BWHA	10/15/2015	将状态从高级更改为最终。
*B	6215260	PRIT	06/22/2018	更新的订购信息: 更新订购部件编号: 更新图中“包装材料”下的“H”对应的详细信息。 添加了注释“无卤定义符合 IEC 61249-2-21 规范。”并在“H”中提到了相同的注释。 更新为新模板。 完成日落审查。

销售、解决方案和法律信息

全球销售和 design 支持

赛普拉斯公司拥有一个由办事处、解决方案中心、厂商代表和经销商组成的全球性网络。要找到离您最近的办事处，请访问[赛普拉斯所在地](#)。

产品

ARM® Cortex® 微控制器	cypress.com/arm
汽车级产品	cypress.com/automotive
时钟与缓冲器	cypress.com/clocks
接口	cypress.com/interface
物联网	cypress.com/iot
微控制器	cypress.com/mcu
PSoC	cypress.com/psoc
电源管理 IC	cypress.com/pmic
触摸感应	cypress.com/touch
USB 控制器	cypress.com/usb
无线连接	cypress.com/wireless

PSoC® 解决方案

[PSoC 1](#) | [PSoC 3](#) | [PSoC 4](#) | [PSoC 5LP](#) | [PSoC 6 MCU](#)

赛普拉斯开发者社区

[论坛](#) | [WICED IoT 论坛](#) | [项目](#) | [视频](#) | [博客](#) | [培训](#) | [组件](#)

技术支持

cypress.com/support

© 赛普拉斯半导体公司，2015–2018。本文件是赛普拉斯半导体公司及其子公司，包括 Spansion LLC (“赛普拉斯”) 的财产。本文件，包括其包含或引用的任何软件或固件 (“软件”)，根据全球范围内的知识产权法律以及美国与其他国家签署条约由赛普拉斯所有。除非在本款中另有明确规定，赛普拉斯保留在该等法律和条约下的所有权利，且未就其专利、版权、商标或其他知识产权授予任何许可。如果软件并不附随有一份许可协议且贵方未以其他方式与赛普拉斯签署关于使用软件的书面协议，赛普拉斯特此授予贵方属人性质的、非独家且不可转让的如下许可 (无再许可) (1) 在赛普拉斯特软件著作权项下的下列许可 (一) 对以源代码形式提供的软件，仅出于在赛普拉斯硬件产品上使用之目的且仅在贵方集团内部修改和复制软件，和 (二) 仅限于在有关赛普拉斯硬件产品上使用之目的将软件以二进制代码形式的向外部最终用户提供 (无论直接提供或通过经销商和分销商间接提供)，和 (2) 在被软件 (由赛普拉斯公司提供，且未经修改) 侵犯的赛普拉斯专利的权利主张项下，仅出于在赛普拉斯硬件产品上使用之目的制造、使用、提供和进口软件的许可。禁止对软件的任何其他使用、复制、修改、翻译或汇编。

在适用法律允许的限度内，赛普拉斯未对本文件或任何软件作出任何明示或暗示的担保，包括但不限于关于适销性和特定用途的默示保证。没有任何电子设备是绝对安全的。因此，尽管赛普拉斯在其硬件和软件产品中采取了必要的安全措施，但是赛普拉斯并不承担任何由于使用赛普拉斯产品而引起的安全问题及安全漏洞的责任，例如未经授权的访问或使用赛普拉斯产品。此外，本材料中所介绍的赛普拉斯产品有可能存在设计缺陷或设计错误，从而导致产品的性能与公布的规格不一致。(如果发现此类问题，赛普拉斯会提供勘误表) 赛普拉斯保留更改本文件的权利，届时将不另行通知。在适用法律允许的限度内，赛普拉斯不对因应用或使用本文件所述任何产品或电路引起的任何后果负责。本文件，包括任何样本设计信息或程序代码信息，仅为供参考之目的提供。文件使用人应负责正确设计、计划和测试信息应用和由此生产的任何产品的功能和安全性。赛普拉斯产品不应被设计为、设定为或授权用作武器操作、武器系统、核设施、生命支持设备或系统、其他医疗设备或系统 (包括急救设备和手术植入物)、污染控制或有害物质管理系统中的关键部件，或产品植入之设备或系统故障可能导致人身伤害、死亡或财产损失其他用途 (“非预期用途”)。关键部件指，若该部件发生故障，经合理预期会导致设备或系统故障或会影响设备或系统安全性和有效性的部件。针对由赛普拉斯产品非预期用途产生或相关的任何主张、费用、损失和其他责任，赛普拉斯不承担全部或部分责任且贵方不应追究赛普拉斯之责任。贵方应赔偿赛普拉斯因赛普拉斯产品任何非预期用途产生或相关的所有索赔、费用、损失和其他责任，包括因人身伤害或死亡引起的主张，并使之免受损失。

赛普拉斯、赛普拉斯徽标、Spansion、Spansion 徽标，及上述项目的组合，WICED，及 PSoC、CapSense、EZ-USB、F-RAM 和 Traveo 应视为赛普拉斯在美国和其他国家的商标或注册商标。请访问 cypress.com 获取赛普拉斯商标的完整列表。其他名称和品牌可能由其各自所有者主张为该方财产。