

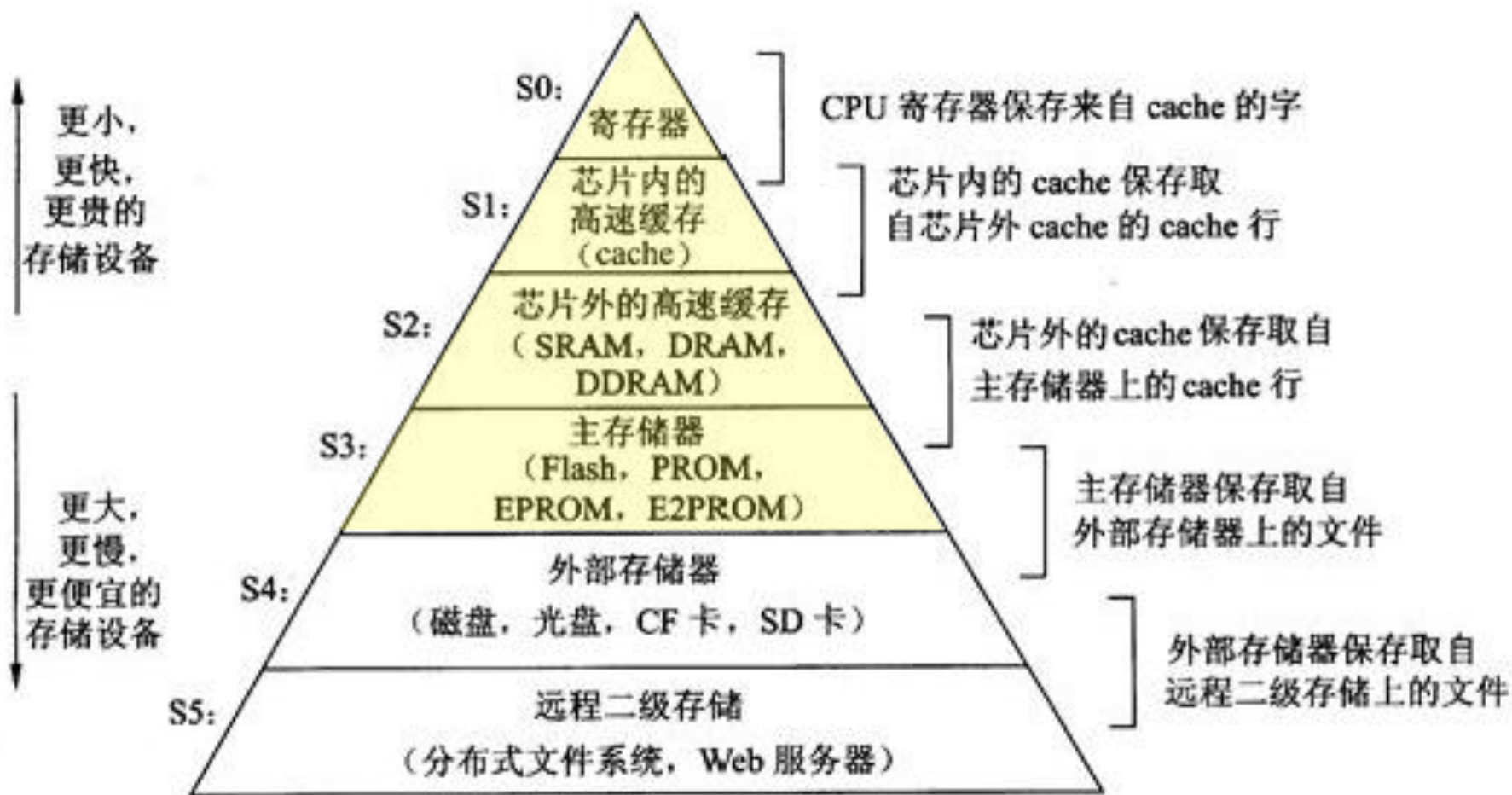
## 4. 存储器系统

## ➤ 4.1 存储器系统概述

- 存储器系统作为嵌入式系统中不可或缺的一部分，主要是用来**存放指令和数据**。
- 嵌入式系统的主存储器由于体系结构的限制，存在若干不足之处。
- **折中考虑数据访问需求和成本性能**，一般会构建**多级层次存储器**，以优化的控制调度算法、合理的成本、合理的性能构成可用和经济的存储器系统。

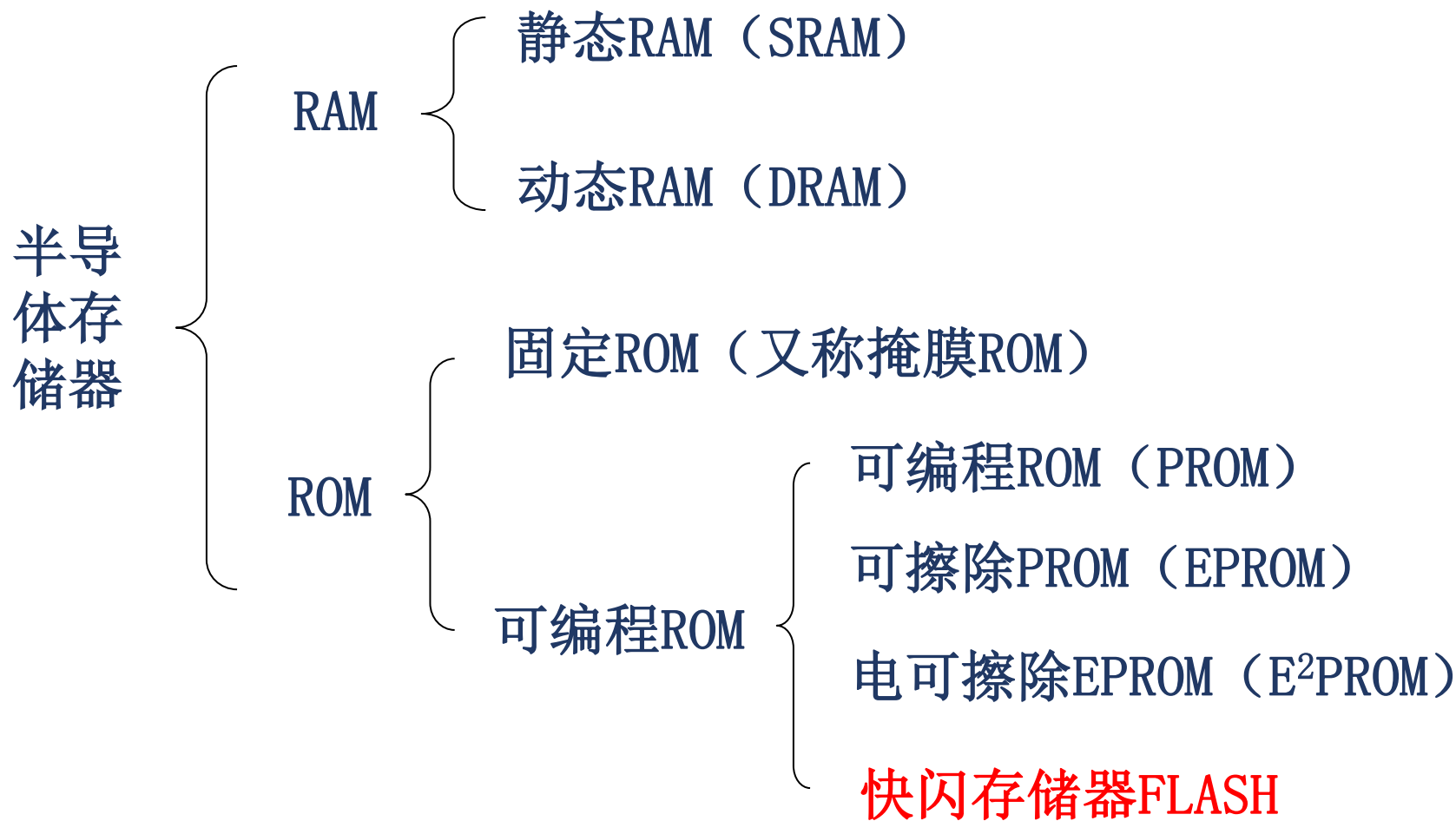
## 4.1 存储器系统概述

### 存储器的性能用途层次结构



## ➤ 4.2 存储器的分类和性能分析

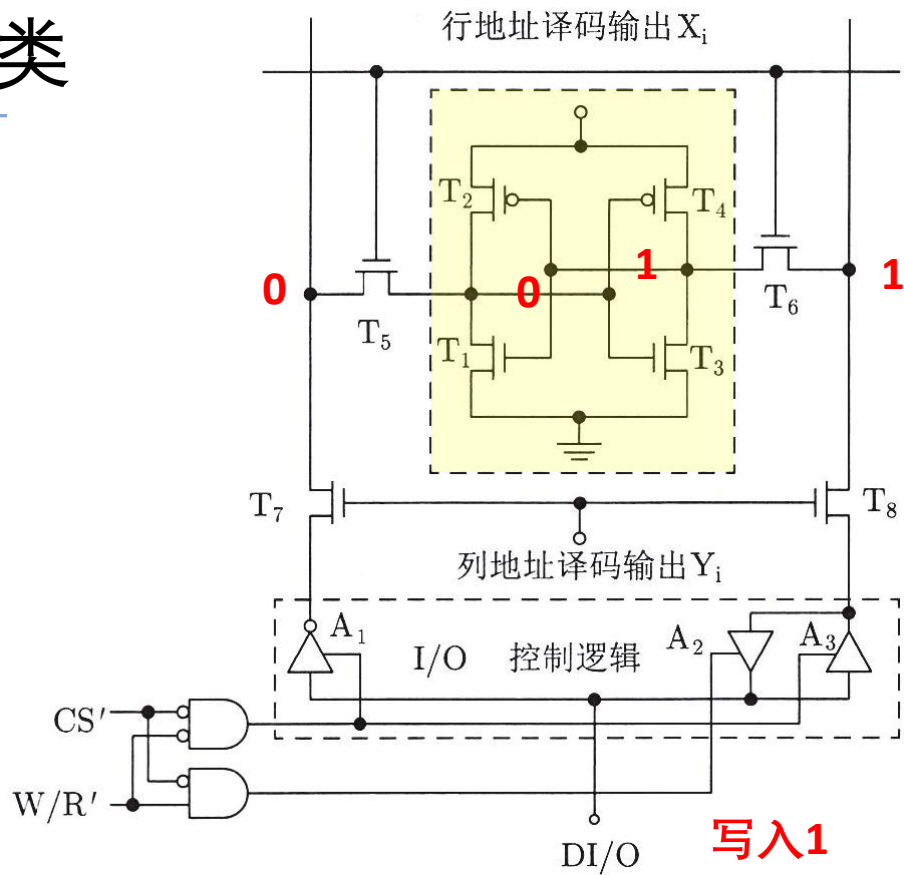
### 存储器分类



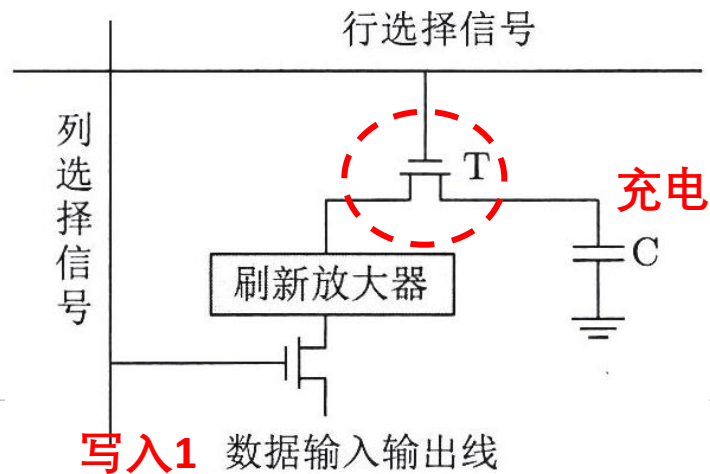
## 4.2 存储器的分类

### RAM

#### SRAM (Static RAM)

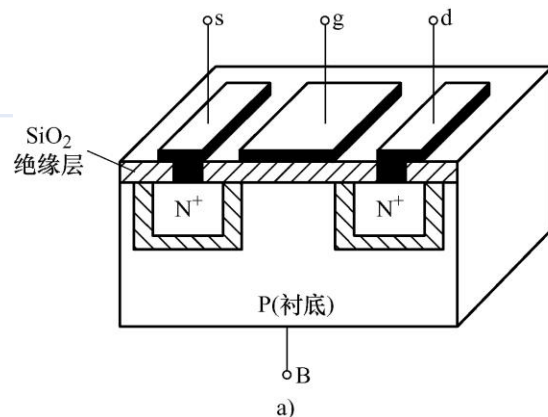


#### DRAM (Dynamic RAM)

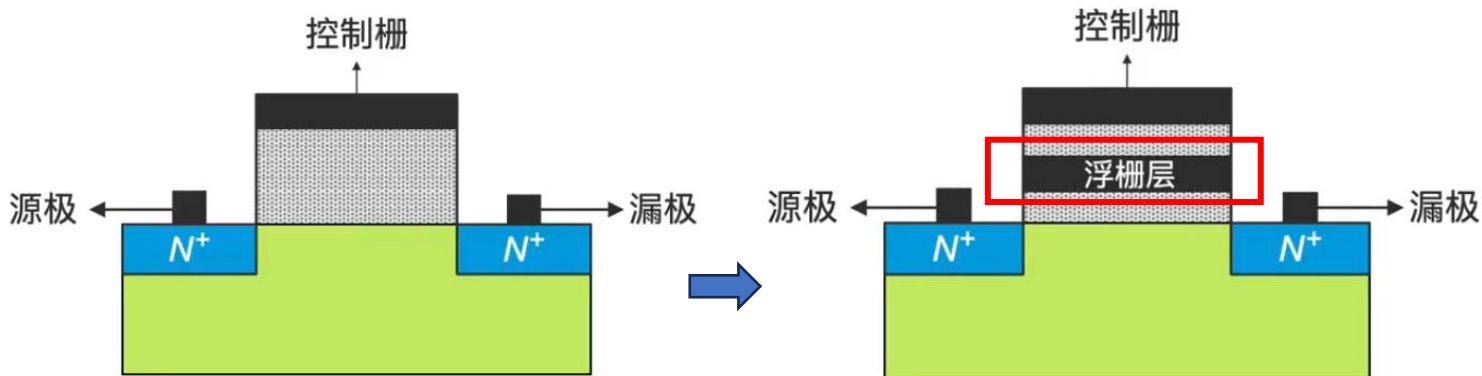


## 4.2 存储器的分类 - FLASH

### FLASH的工作原理

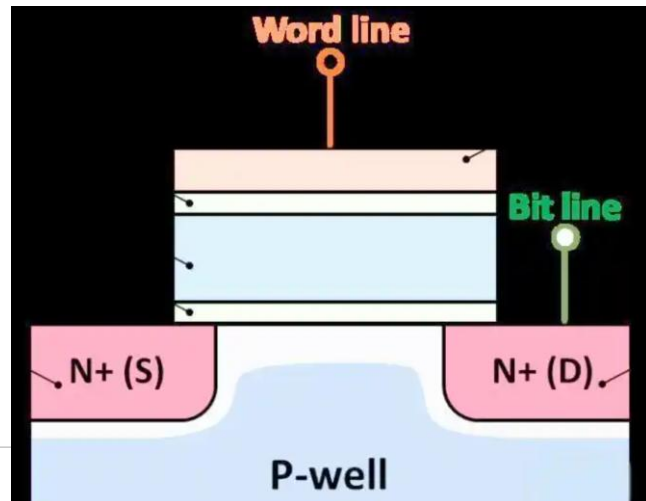


#### MOS管



闪存基本存储单元由浮栅MOS管构成，通过电荷存储实现数据保存。

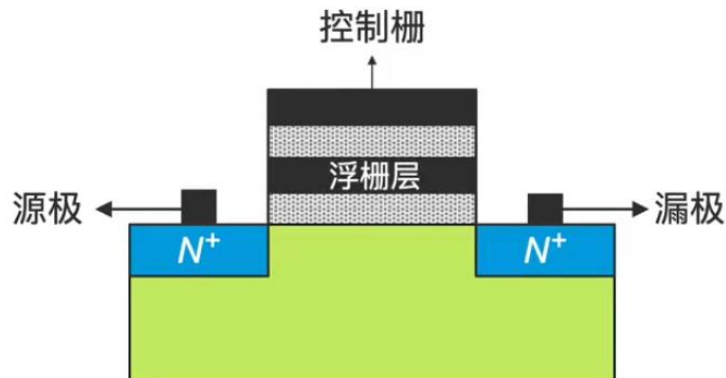
浮栅能够存储电荷，进而影响控制栅极的开关阈值电平 $V_{th}$ 。如果浮栅中没有电荷，那么该存储单元就被表示为“1”；而当浮栅中存储了负电荷或电子时，该单元则被表示为“0”。



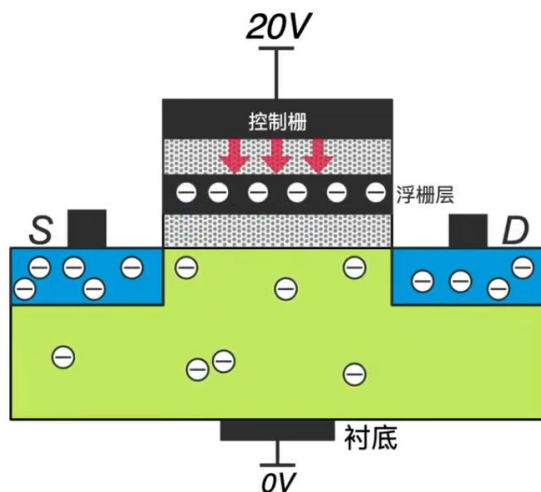
## 4.2 存储器的分类 - FLASH

### FLASH 工作原理

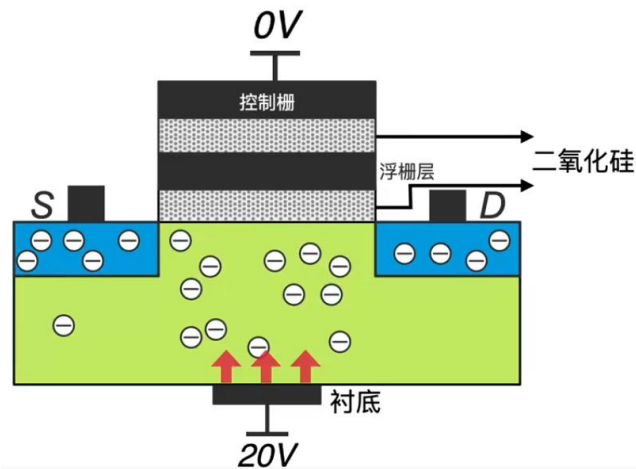
闪存基本存储单元  
由浮栅MOS管构成，  
通过电荷存储  
实现数据保存。



写入“0”和写入“1”



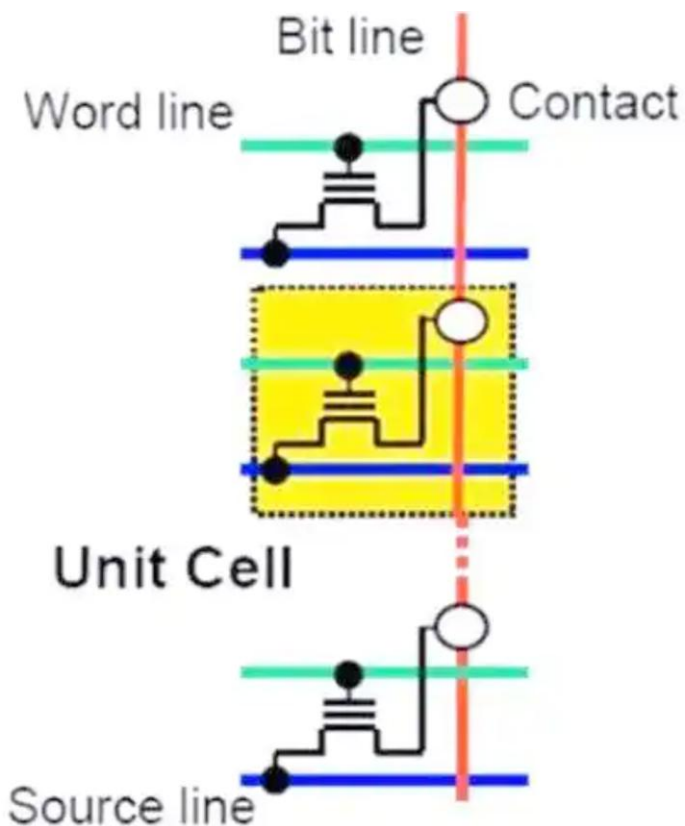
存入电荷表示逻辑“0”



吸出电荷表示逻辑“1”，即擦除

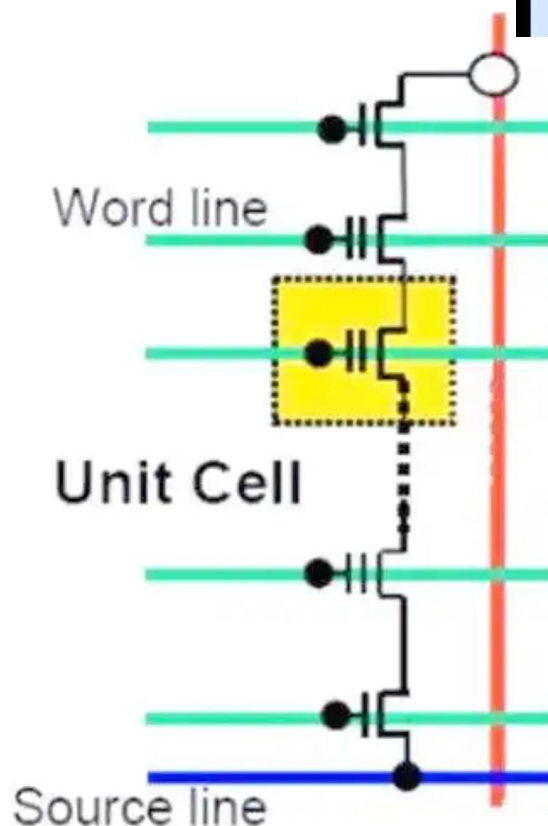
## 4.2 存储器的分类 - FLASH 2种FLASH类型

### NOR型

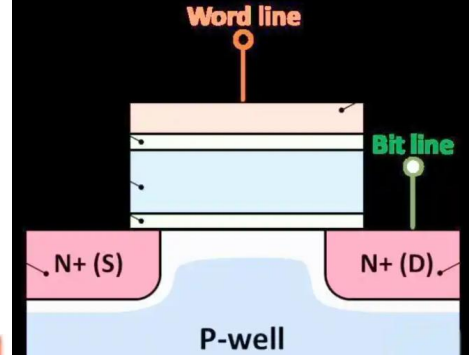


NOR型将位线直接连接到每个存储单元，使得每个单元都有独立的布线。尽管这种设计增加了布线的复杂性，但提供了随机读取每个单元的能力。

### NAND型



NAND型采用存储单元串联的方式连接，使得源极线可以被多个单元共享，减少了布线的数量。其特点是集成度高、容量大且价格低。



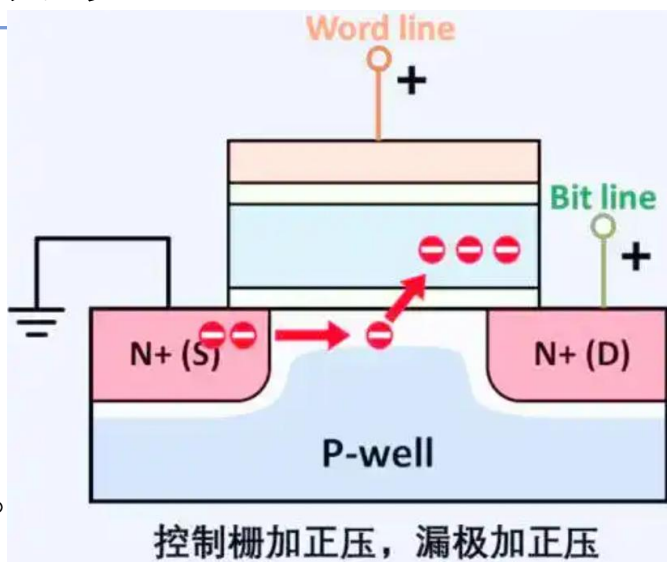
## 4.2 存储器的分类

### NOR型 FLASH

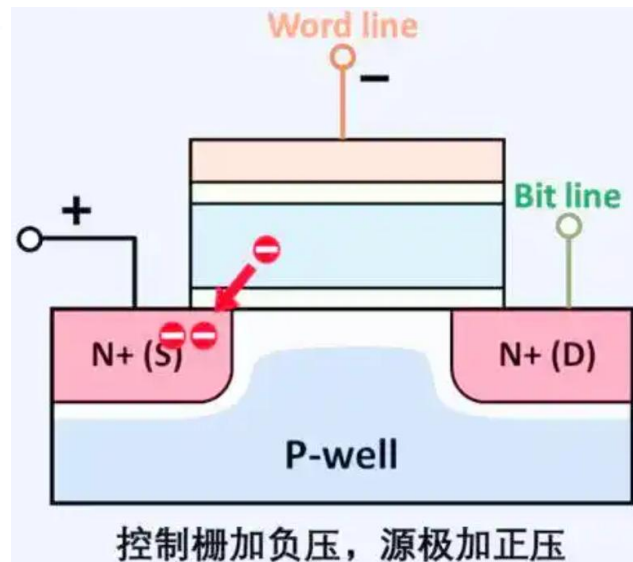
**写入和擦除：**通过控制栅极电压实现。向浮置栅极注入电荷，实现存储“0”的操作；通过抽出浮置栅极上的电荷来实现擦除，相当于存储“1”。

**读取：**在控制栅极施加正电压，通过检测源极和漏极之间的电流大小，判断存储的是“0”还是“1”。

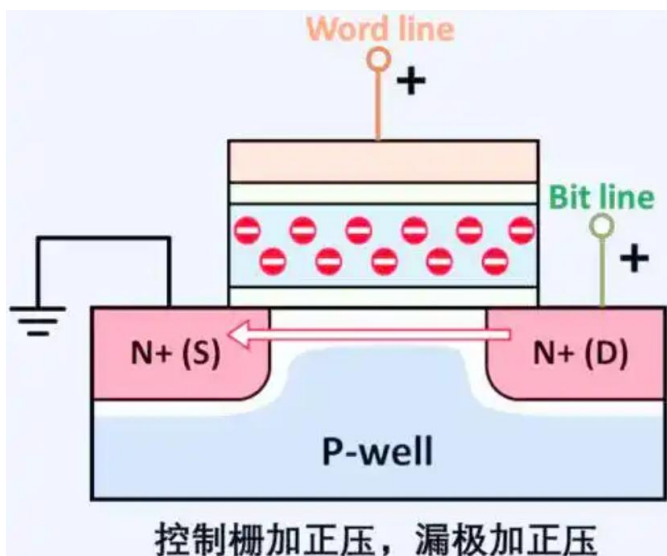
“0”的写入



“1”的写入(擦除)

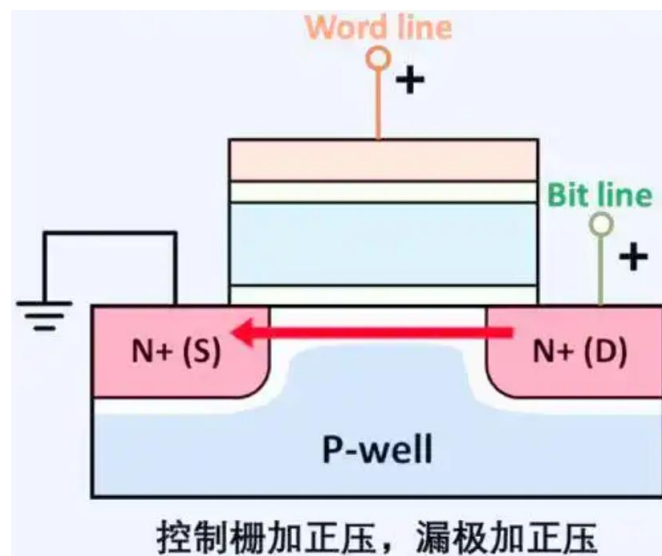


“0”的读取



由于栅极电荷排斥电子，源漏之间电流很小

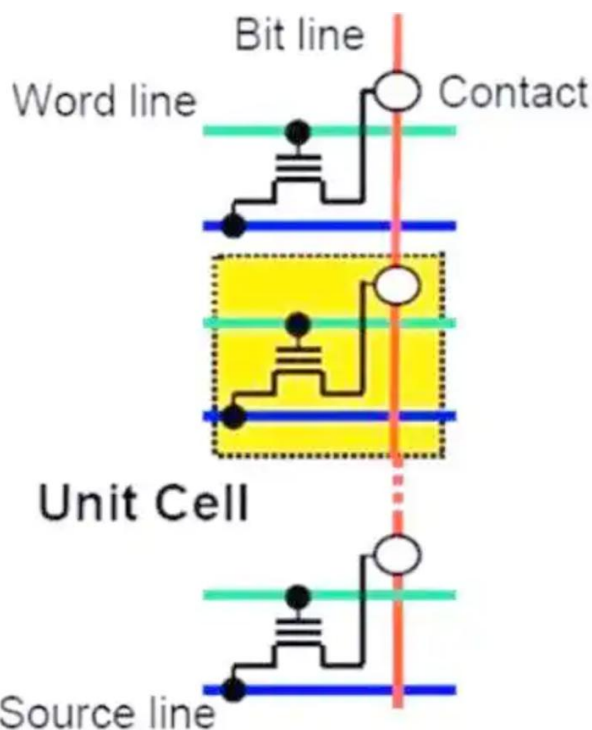
“1”的读取



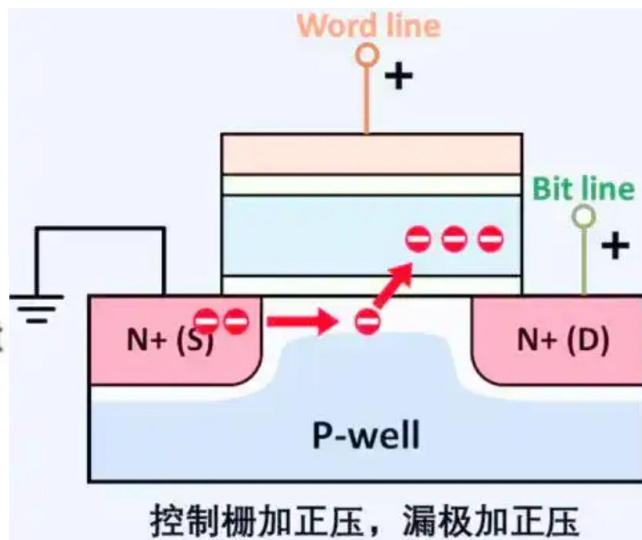
由于栅极没有电荷阻碍电子流动，源漏之间会有大电流通过

## 4.2 存储器的分类

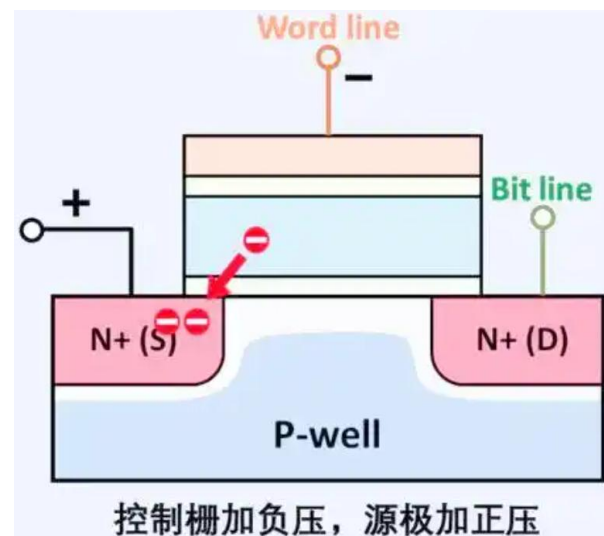
### NOR型



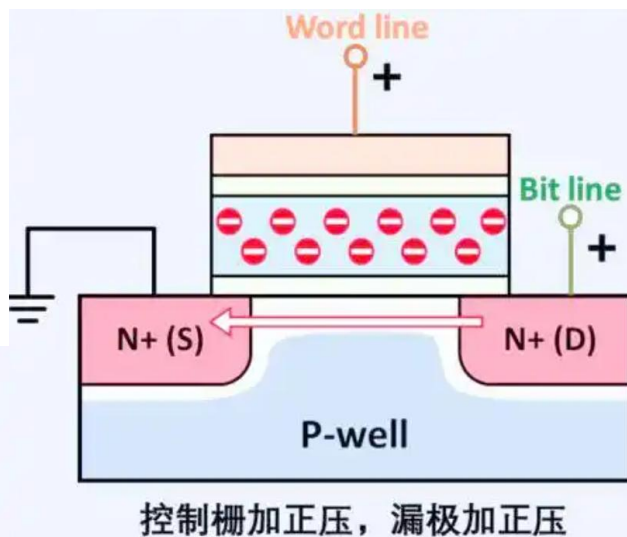
“0”的写入



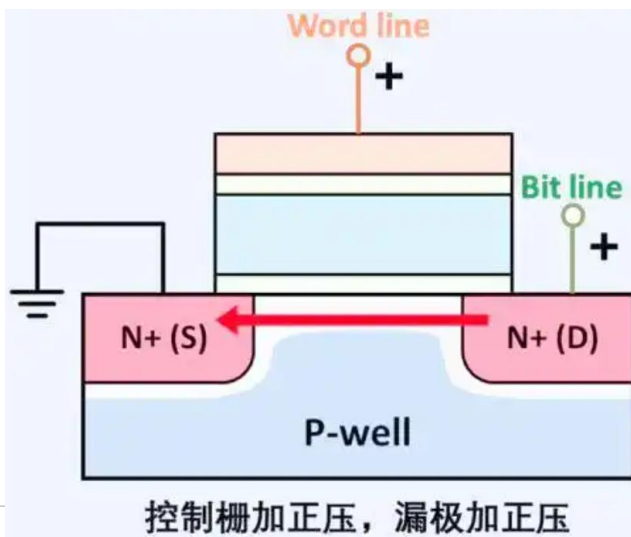
“1”的写入(擦除)



“0”的读取

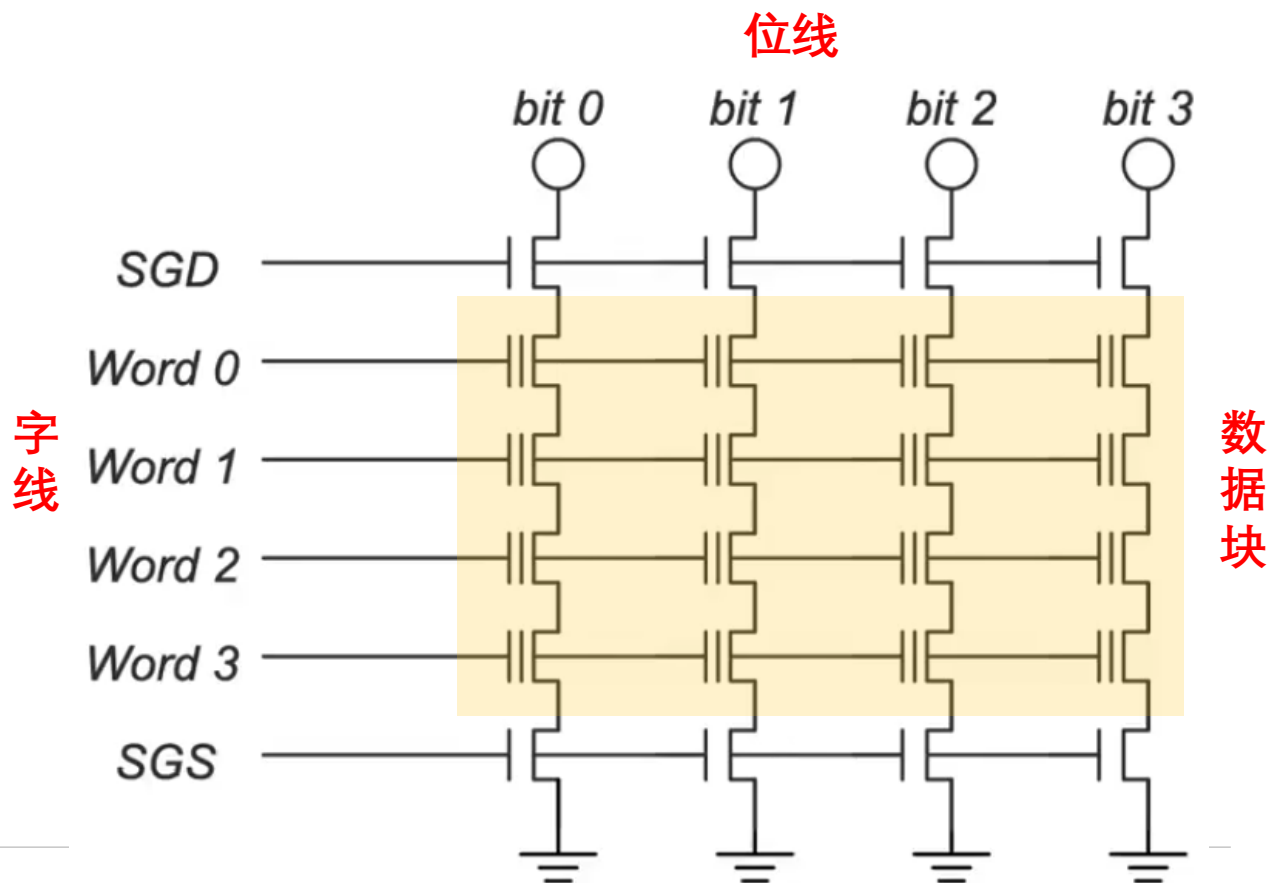
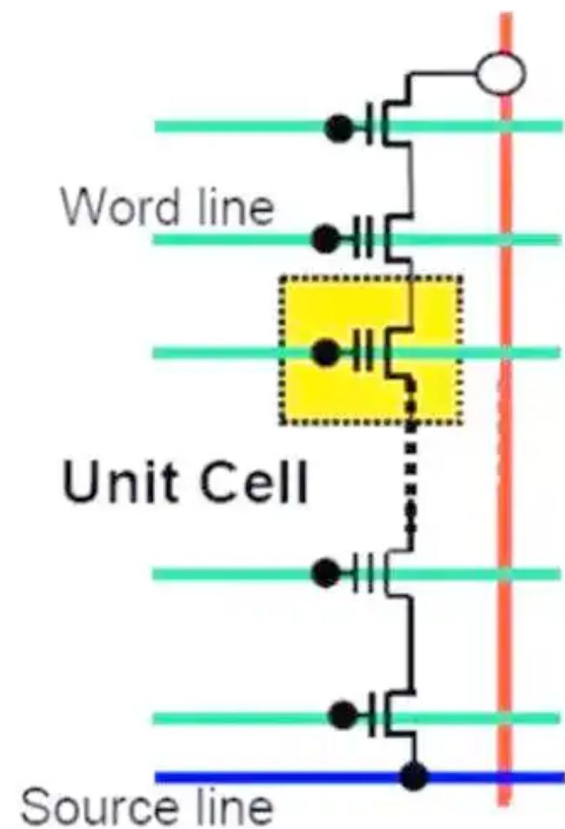
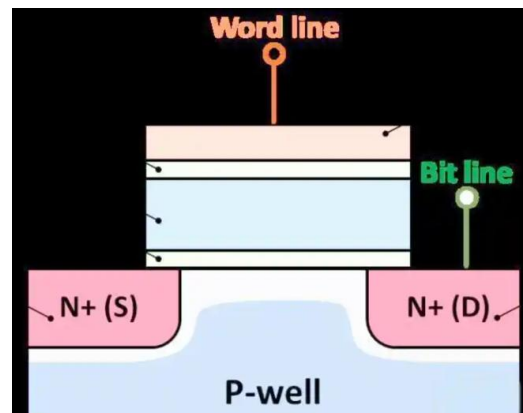


“1”的读取



## 4.2 存储器的分类

### NAND型

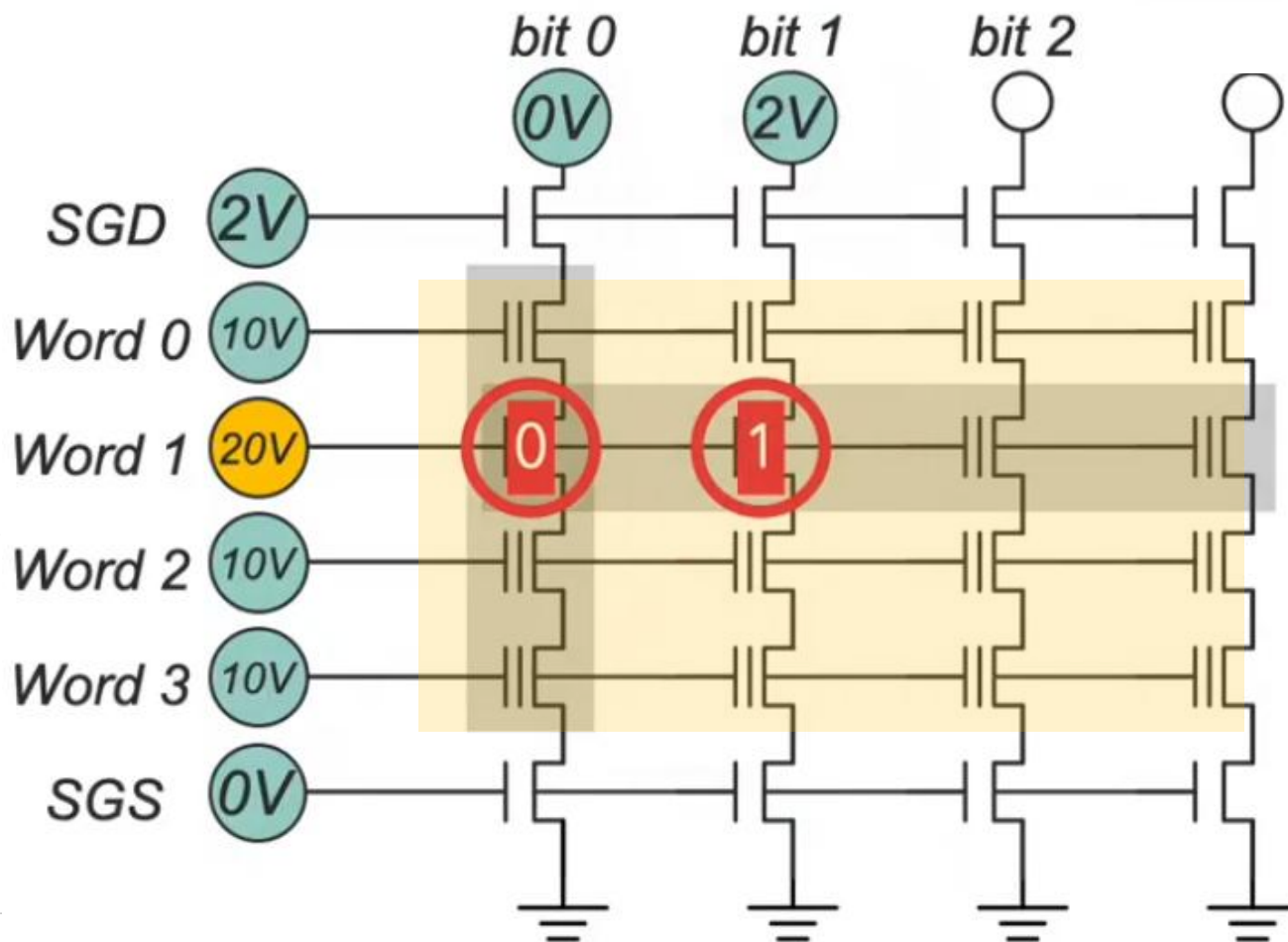


## 4.2 存储器的分类

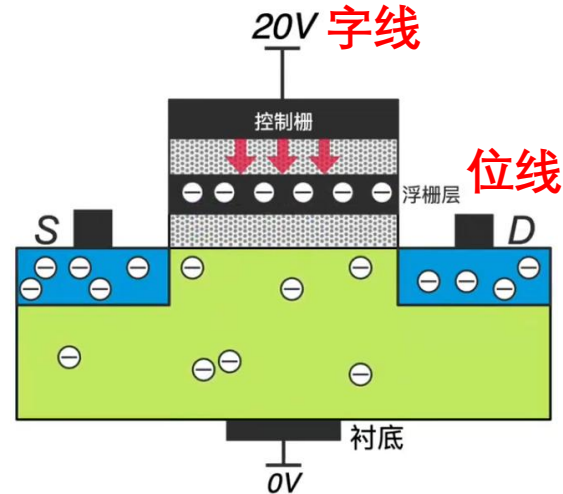
### NAND型工作原理

字线

字线给  
高电平

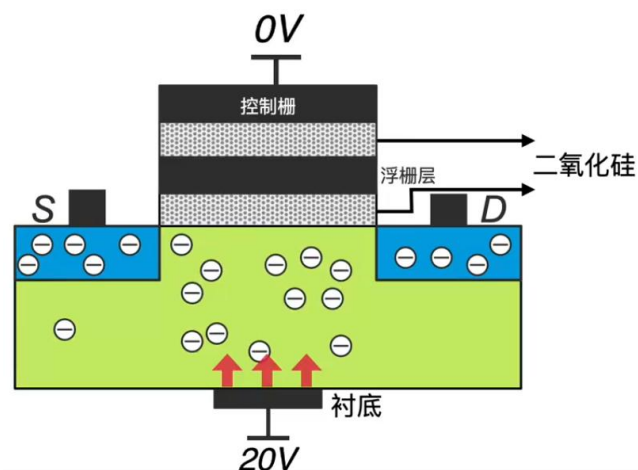


位线



## 4.2 存储器的分类

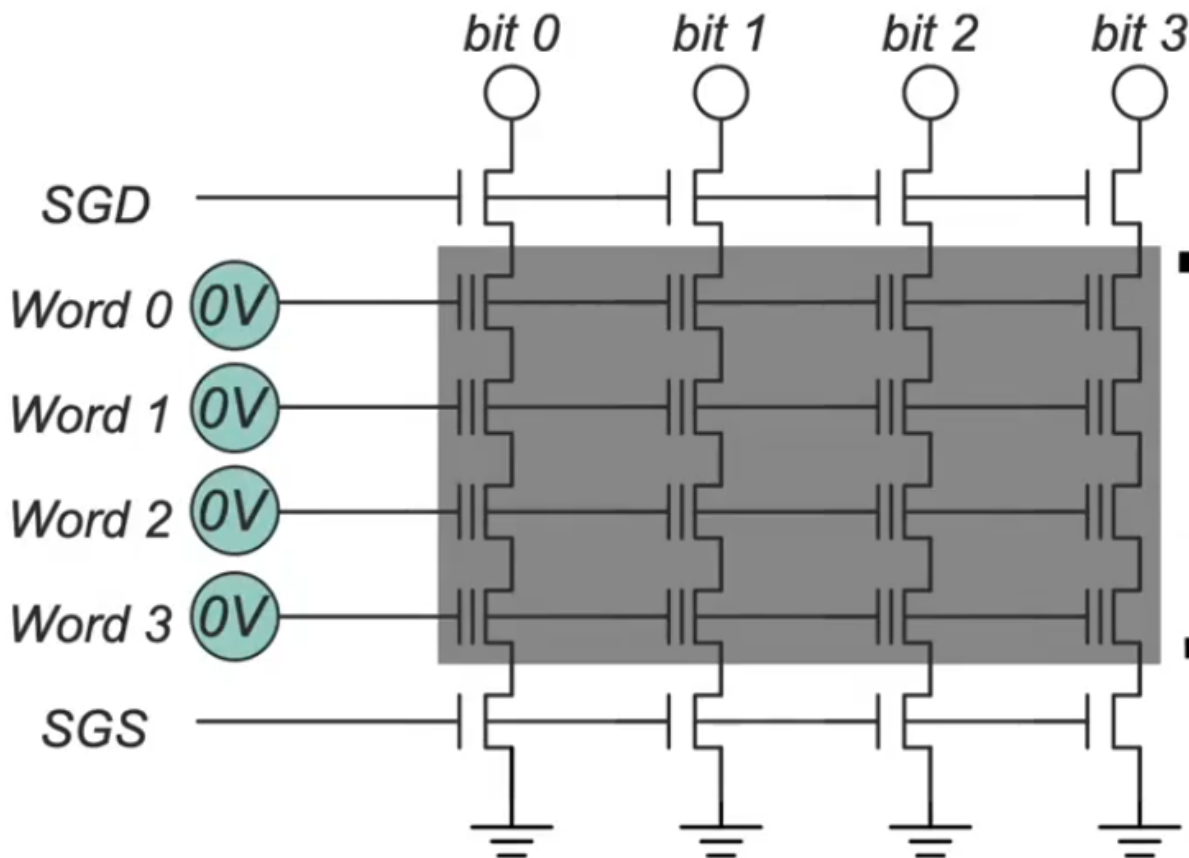
### NAND型工作原理



位线

字线

字线给  
高电平

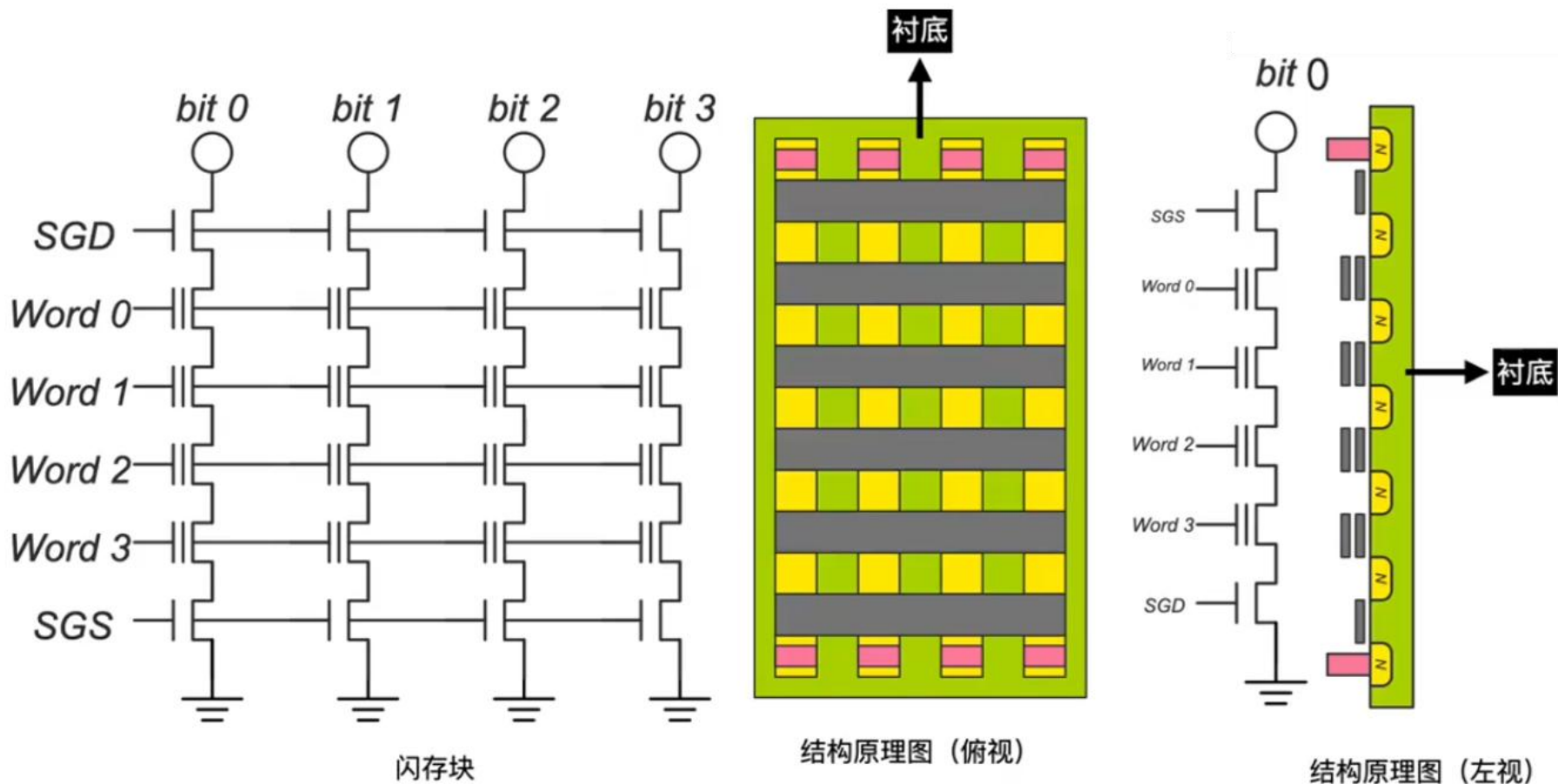


数据块  
擦除要  
整块擦  
除



## 4.2 存储器的分类

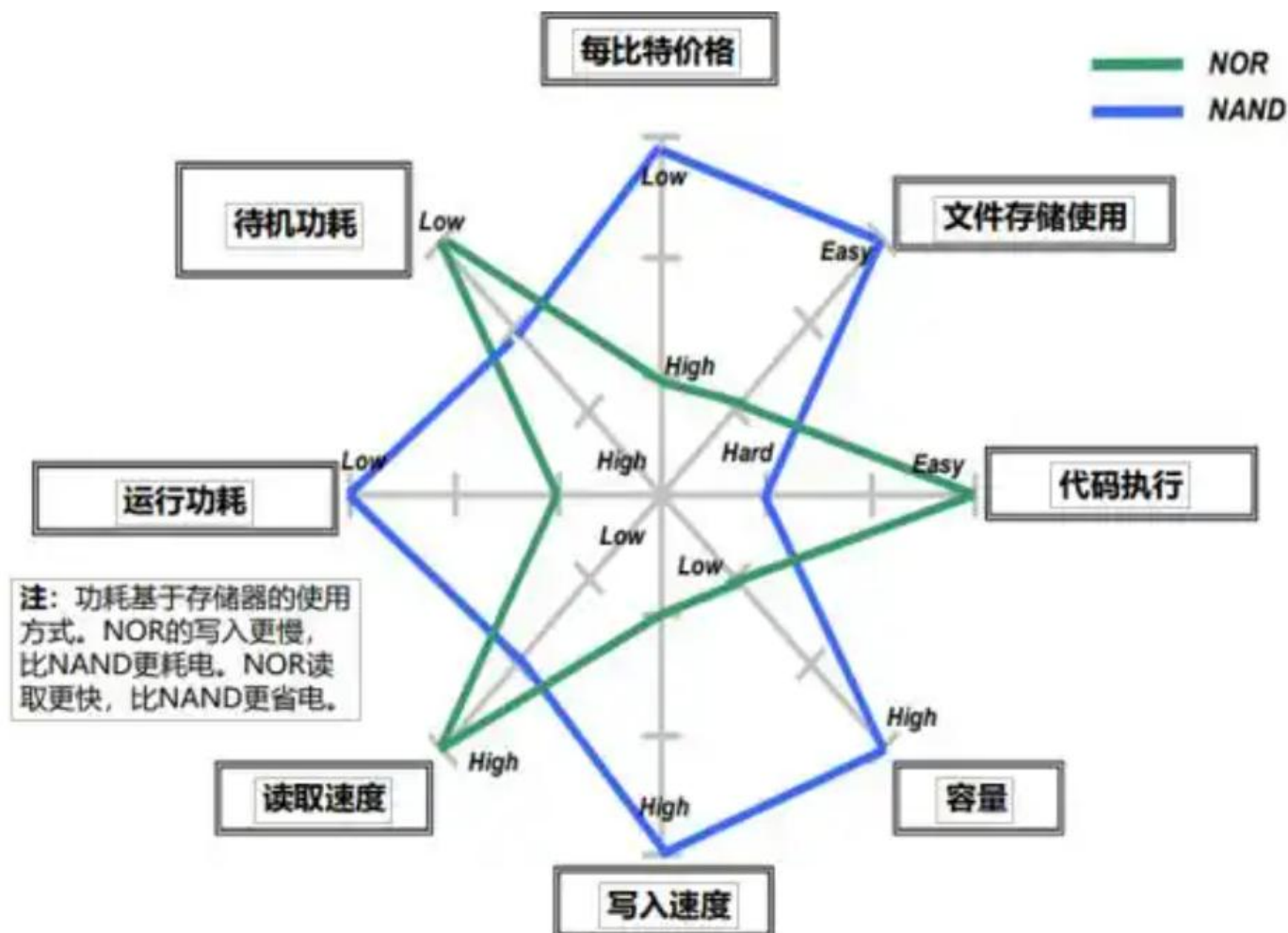
### NAND型工作原理



## 4.2 存储器的分类

### NAND与NOR FLASH的比较

东芝存储提供的图表总结了NAND和NOR闪存在多个关键设计特性上的差异。包括容量、读取速度、写入速度、功耗、每比特成本，以及在文件存储和代码存储方面的适用性。



## ➤ 4.2 存储器的分类

### NAND与NOR FLASH的比较

参 数	NOR 闪存	NAND 闪存
容量	中	大
程序可否直接运行	可以	不可以
擦除	慢	快
工作速度写	慢	快
读	快	快
擦除次数	10 000~100 000	100 000~1 000 000
擦除方式	FN 隧道穿越	FN 隧道穿越
编程方式	热电子注入	FN 隧道穿越
访问方式	随机访问	顺序访问
价格	高	很低
擦除单位	块	块
编程单位	字节	页
读取单位	字节	页
优势	随机访问	寿命长，成本低
用途	启动 ROM（BIOS 等）	存储装置

## ➤ 4.2 存储器的主要技术指标

### 主要技术指标

#### ➤ 容量

以字节为单位，包括：B、KB、MB、GB、TB。

#### ➤ 存取时间

CPU对内存进行读或写操作时所需要花费的时间。也就是从CPU发出读/写命令，到完成本命令为止的时间。一般DRAM为几十纳秒，SRAM为几纳秒。

#### ➤ 存储周期

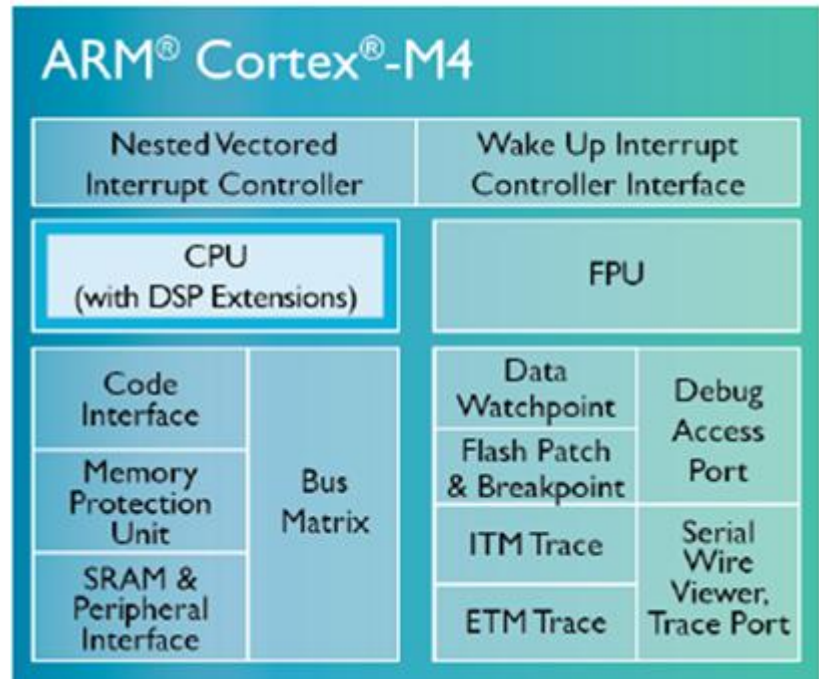
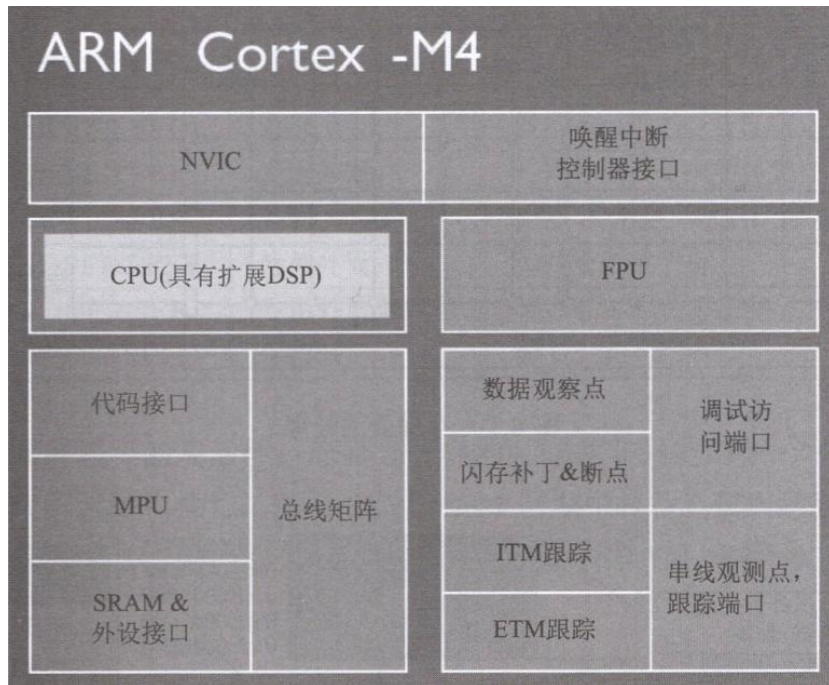
存储器从接收到读出命令到被读出信息稳定在输出端为止的时间间隔称为“取数时间”。两次独立的存取操作之间所需最短时间称为“存取周期”

#### ➤ 访问带宽

内存总线所能提供的数据传输能力，其大小可表示为：  
 $(\text{存储器位数}/8) \times \text{读取速度峰值}$

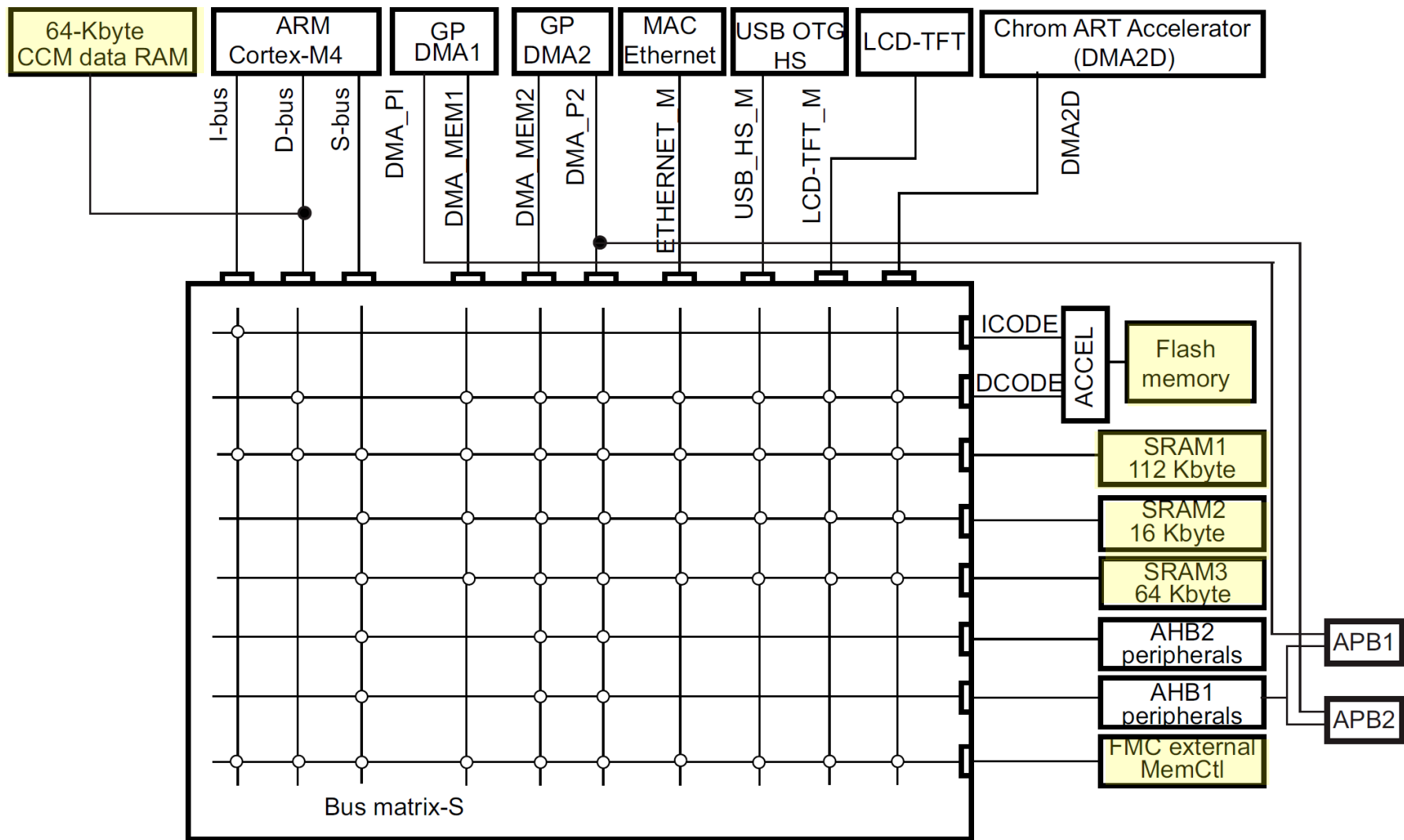
## ➤ 4.3 ARM存储器管理

### Cortex-M4芯核结构



## 4.3 ARM存储器管理

### 存储器挂载结构



## ➤ 4.3 ARM存储器管理 – 存储空间

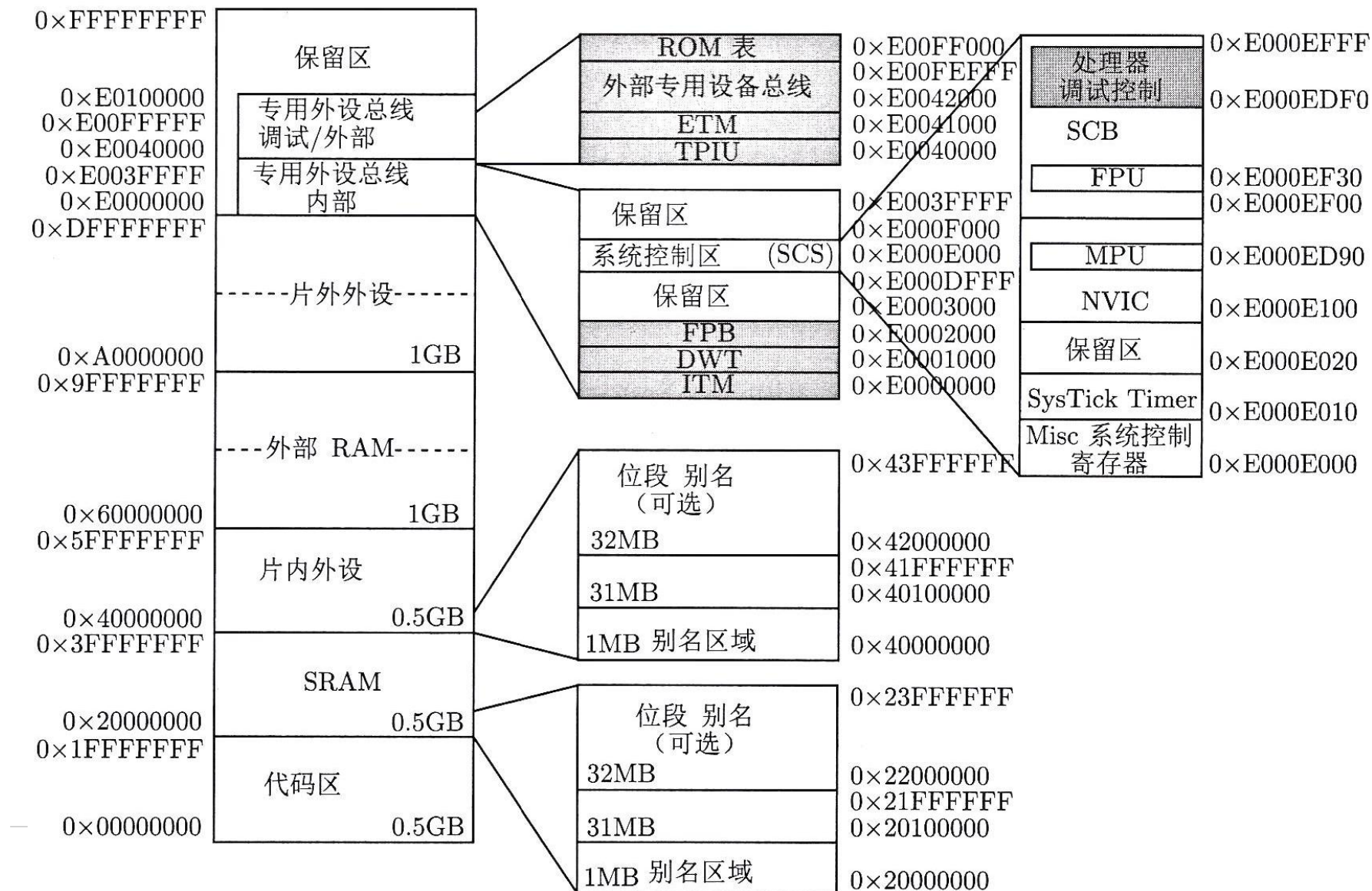
### 存储地址空间

- STM32F4的地址总线是32位，可以访问4GB的地址空间
- STM32F4将可访问的存储空间根据功能划分为若干个模块

地 址 范 围	存储功能	描 述
0x0000.0000 ~ 0x1FFF.FFFF	代码区、数据区	是 Flash 存储，通常用来存储程序代码，可以存储数据
0x2000.0000 ~ 0x3FFF.FFFF	SRAM	数据和代码的运行区域。断电后数据消失
0x4000.0000 ~ 0x5FFF.FFFF	片内外设	片内外设模块中的各类寄存器和缓存区域
0x6000.0000 ~ 0x9FFF.FFFF	外部 RAM	内部 SARM 不够用，外加的 RAM
0xA000.0000 ~ 0xDFFF.FFFF	片外外设	请查看芯片手册
0xE000.0000 ~ 0xE00F.FFFF	专用外设总线	请查看芯片手册
0xE010.0000 ~ 0xFFFF.FFFF	保留区	无

## 4.3 ARM存储器管理 – 存储空间

### STM32F4器件地址空间映射图



## ➤ 4.3 ARM存储器管理 – 存储分区

系统级 System level 512MB	0xFFFF FFFF 0xE000 0000
片外外设 External Device 1GB	0xDFFF FFFF 0xA000 0000
片外RAM External RAM 1GB	0x9FFF FFFF 0x6000 0000
片上外设 Peripheral 512MB	0x5FFF FFFF 0x4000 0000
片上SRAM SRAM 512MB	0x3FFF FFFF 0x2000 0000
代码区 Code 512MB	0x1FFF FFFF 0x0000 0000

### ➤ 0x00000000-0x07FFFFFFF

别名区域: 根据BOOT配置, 可映射到:  
Flash存储器(从0x08000000开始)

系统存储器(内置Bootloader)

SRAM(从0x20000000开始)

主要用途: CPU复位后首先从0x00000000读取初始栈指针和程序计数器, 这个重映射机制允许灵活选择启动设备

### ➤ 0x08000000-0x0FFFFFFF

Flash存储器区域: 内部Flash的实际物理地址

用途: 存储程序代码、常量数据和向量表

### ➤ 0x10000000-0x1FFFFFFF

片上存储器区域: 包含特殊功能内存  
包含:

CCM: 0x10000000开始

.....

## ➤ 4.3 ARM存储器管理 – 存储分区

系统级 System level 512MB	0xFFFF FFFF ----- 0xE000 0000 ----- 0xDFFF FFFF
片外外设 External Device 1GB	----- 0xA000 0000 ----- 0x9FFF FFFF
片外RAM External RAM 1GB	----- 0x6000 0000 ----- 0x5FFF FFFF
片上外设 Peripheral 512MB	0x4000 0000 ----- 0x3FFF FFFF
片上SRAM SRAM 512MB	0x2000 0000 ----- 0x1FFF FFFF
代码区 Code 512MB	----- 0x0000 0000 -----

### ➤ 片上SRAM区

主SRAM(SRAM1) 0x20000000开始

附加SRAM区域(SRAM2/3等): 紧接SRAM1后排列

备份SRAM: 某些STM32中包含电池供电的小型SRAM

用途: 存储全局变量、堆、栈和动态分配的数据

.....

### ➤ 片上外设区

**0x40000000-0x4FFFFFFF**

APB1/APB2/AHB1/AHB2外设区域: 所有外设寄存器均映射在此区域。

**APB1外设**(低速): 0x40000000-0x4000FFFF(USART2/3, SPI2/3, I2C1/2/3, 各种定时器等)

**APB2外设**(高速): 0x40010000-0x4001FFFF(USART1, SPI1, TIM1/8, ADC等)

**AHB1外设**: 0x40020000-0x4007FFFF(GPIO, DMA, RCC, Flash接口等)

**AHB2外设**: 0x40080000-0x400FFFFFFF(USB OTG, DCM1等)

**0x50000000-0x5FFFFFFF**

扩展外设区域: 主要用于更高级的STM32型号中的额外外设。

## 4.3 ARM存储器管理 – 存储分区

系统级 System level 512MB	0xFFFF FFFF 0xE000 0000
片外外设 External Device 1GB	0xDFFF FFFF 0xA000 0000
片外RAM External RAM 1GB	0x9FFF FFFF 0x6000 0000
片上外设 Peripheral 512MB	0x5FFF FFFF 0x4000 0000
片上SRAM SRAM 512MB	0x3FFF FFFF 0x2000 0000
代码区 Code 512MB	0x1FFF FFFF 0x0000 0000

### ➤ 片外存储区

**0x60000000-0x7FFFFFFF**

外部存储器(FMC/FSMC的Bank 1和2)区域

用途: 连接外部SRAM、NOR Flash和PSRAM

特点: 通过 FMC/FSMC进行管理

**0x80000000-0x8FFFFFFF**

外部存储器(FMC/FSMC的Bank 3)区域

用途: 主要用于连接外部NAND Flash

**0x90000000-0x9FFFFFFF**

外部存储器(FMC/FSMC的Bank 4)区域

用途: 在STM32F4/F7/H7系列中用于连接PC卡和CompactFlash设备

### ➤ 片外外设区

**0xA0000000-0xBFFFFFFF**

保留区域/特定于型号的扩展区域

**0xC0000000-0xCFFFFFFF**

FMC Bank 5和6区域

用途: 在支持SDRAM的STM32中连接外部SDRAM

**0xD0000000-0xDFFFFFFF**

保留区域/设备扩展空间

## 4.3 ARM存储器管理 – 存储分区

系统级 System level 512MB	0xFFFF FFFF 0xE000 0000
片外外设 External Device 1GB	0xDFFF FFFF 0xA000 0000
片外RAM External RAM 1GB	0x9FFF FFFF 0x6000 0000
片上外设 Peripheral 512MB	0x5FFF FFFF 0x4000 0000
片上SRAM SRAM 512MB	0x3FFF FFFF 0x2000 0000
代码区 Code 512MB	0x1FFF FFFF 0x0000 0000

### ➤ 系统存储区

#### 0xE0000000-0xE00FFFFF:

内核外设区域(PPB – Private Peripheral Bus)

包含:

SysTick系统定时器: 0xE000E010-0xE000E01F

NVIC中断控制器: 0xE000E100-0xE000E4EF

系统控制块(SCB): 0xE000ED00-0xE000ED8F

MPU内存保护单元: 0xE000ED90-0xE000EDBF

DBGMCU调试单元: 0xE000EDF0-0xE000EDFC

特点: ARM Cortex-M核心的标准外设, 所有STM32系列都类似。

#### 0xE0100000-0xFFFFFFFF

供应商特定区域/保留区域

包含:

厂商专用的调试接口

高级调试和跟踪功能

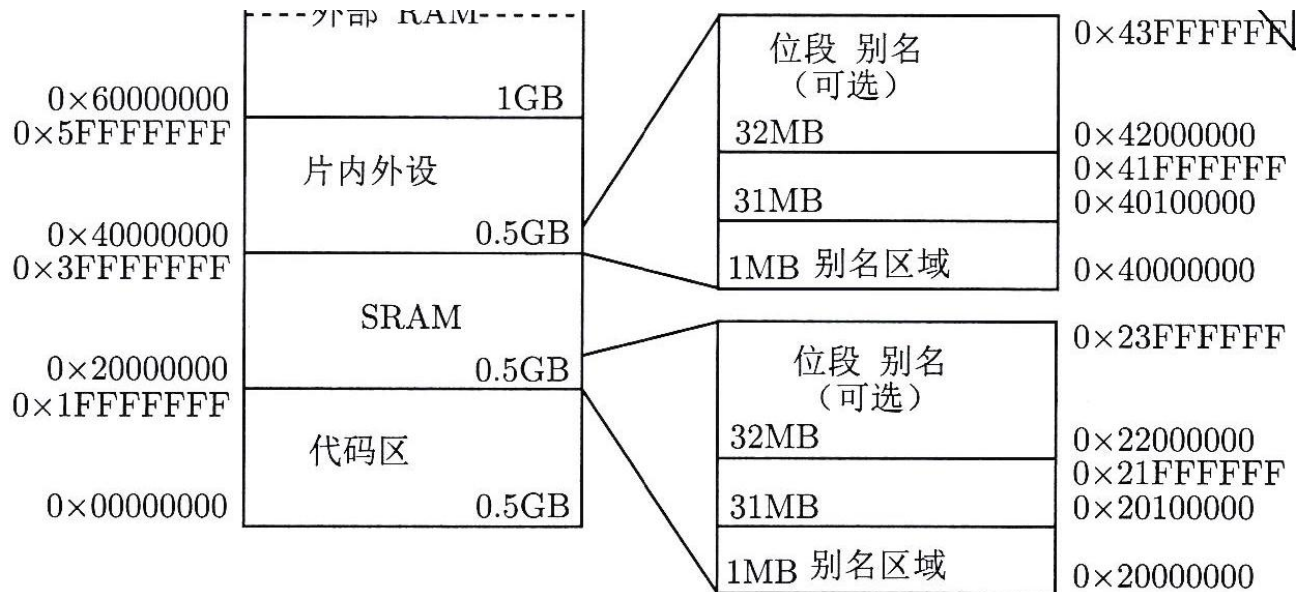
系统ROM表(0xE00FF000-0xE00FFFFF)

大部分空间保留给未来的ARM架构扩展

## 4.3 ARM存储器管理 – 存储分区

### 位带区域

位带映射允许对特定内存区域的单个位进行原子操作，避免了读-修改-写操作中可能的中断问题。



#### SRAM位带区域:

原始区域: 0x20000000-0x200FFFFFFF (1MB)

位带别名区域: 0x22000000-0x23FFFFFFF (32MB)

映射关系: 别名区域中的每个地址对应原始区域中的一个位

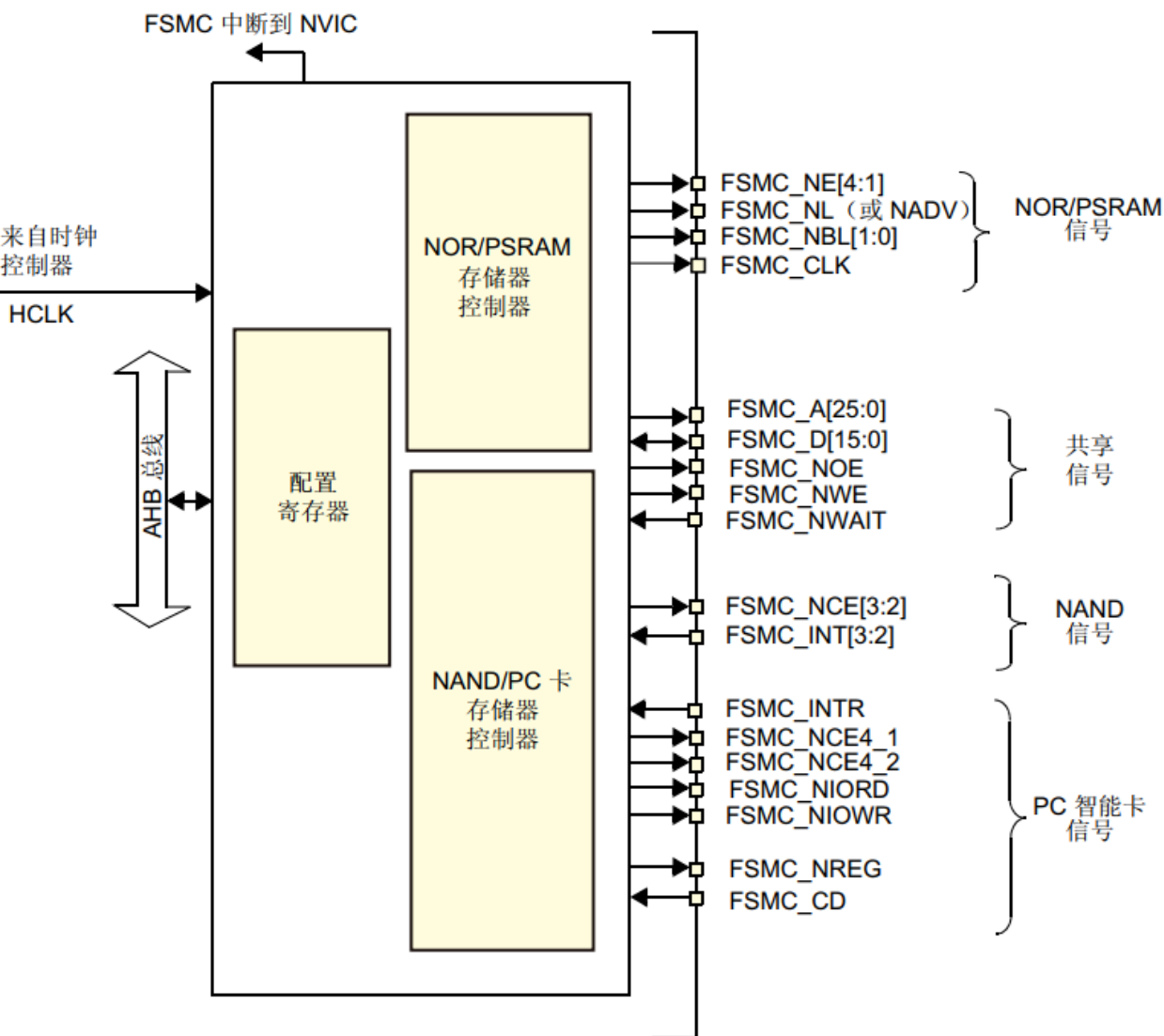
#### 外设位带区域:

原始区域: 0x40000000-0x400FFFFFFF (1MB)

位带别名区域: 0x42000000-0x43FFFFFFF (32MB)

## 4.3 ARM存储器管理 - FSMC

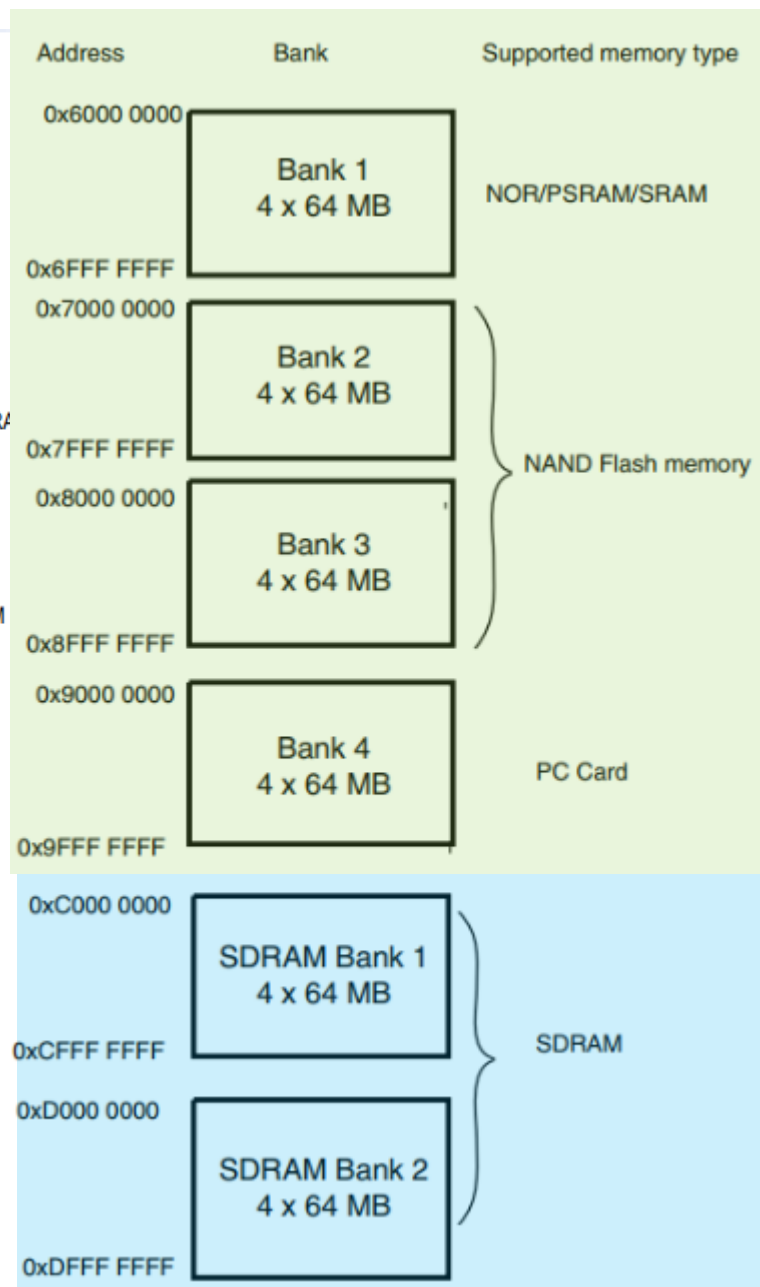
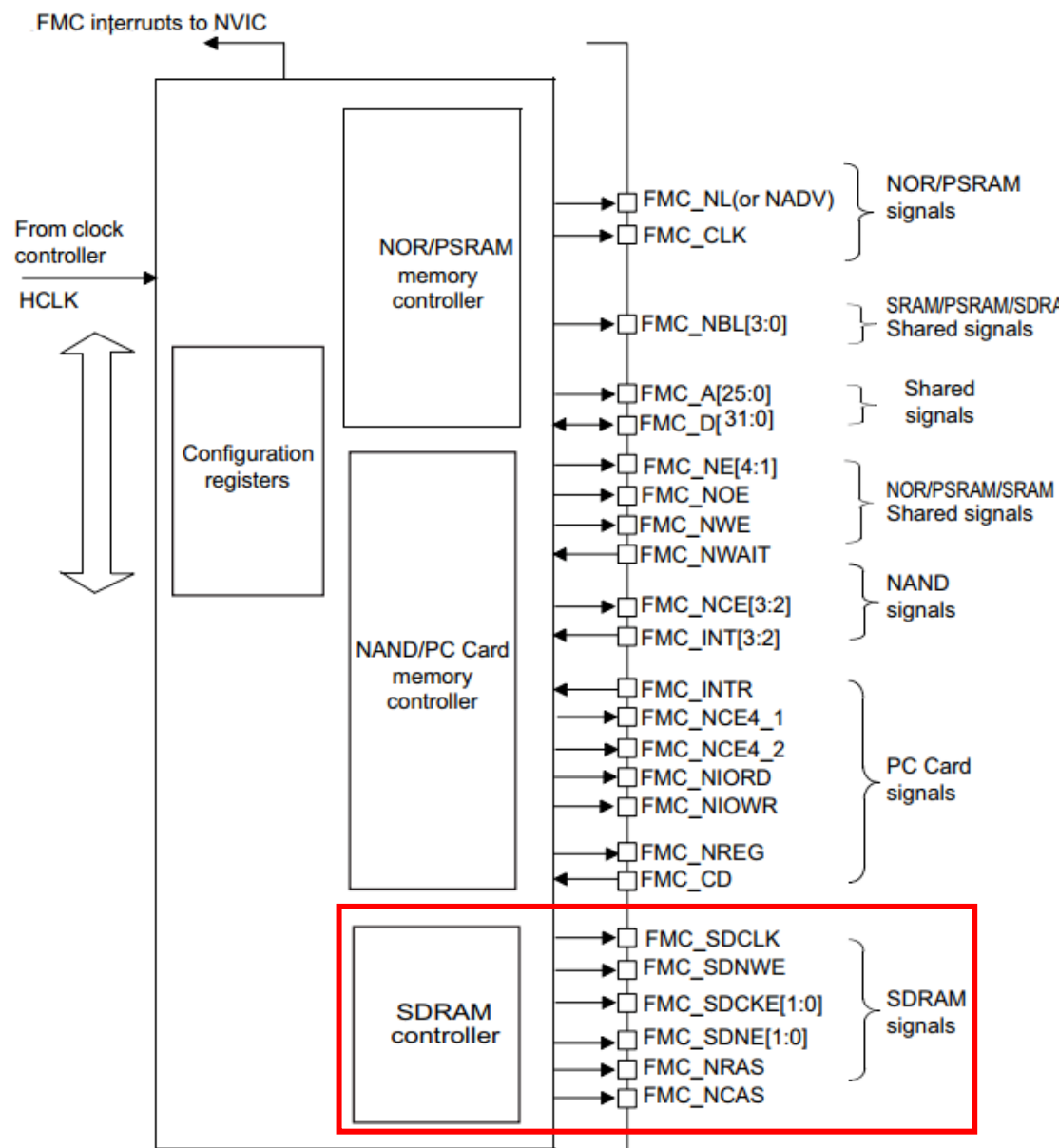
### FSMC (Flexible Static Memory Controller) 静态存储器控制



Address	Banks	Supported memory type
6000 0000h	Bank 1 4 × 64 MB	NOR / PSRAM
6FFF FFFh		
7000 0000h	Bank 2 4 × 64 MB	NAND Flash
7FFF FFFh		
8000 0000h	Bank 3 4 × 64 MB	NAND Flash
8FFF FFFh		
9000 0000h	Bank 4 4 × 64 MB	PC Card
9FFF FFFh		

## 4.3 ARM存储器管理 - FMC

### FMC (Flexible Memory Controller)



## ➤ 4.3 ARM存储器管理 – 存储格式

### 小端格式和大端格式

#### ➤ 小端格式

字长为2字节或更多字节时，字节在内存中的存放顺序为低字节存放在低地址上。**Cortex-M4默认采用的格式。** ➔ **低址低字节**

#### ➤ 大端格式

字长为2字节或更多字节时，字节在内存中的存放顺序为低字节存放在高地址上。  
➔ **低址高字节**

例：Cortex-M4的字长为32位，即4字节，存储一个32位数据：0x12345678到地址0x4FFF0000起始的地址空间：

**小端格式**

地址	数据
0x4FFF 0000	0x78
0x4FFF 0001	0x56
0x4FFF 0002	0x34
0x4FFF 0003	0x12

**大端格式**

地址	数据
0x4FFFF 0000	0x12
0x4FFFF 0001	0x34
0x4FFFF 0003	0x56
0x4FFFF 0003	0x78

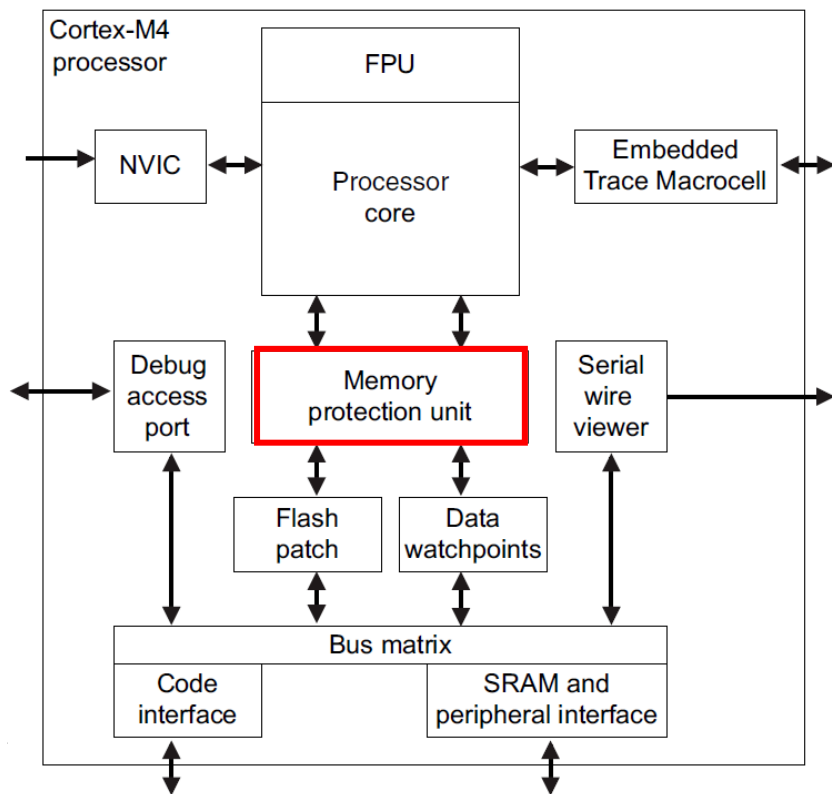
## ➤ 4.3 ARM存储器管理 – 启动配置

**STM32有三种启动模式，分别从不同地址启动**

启动模式 选择引脚		启动模式	说明
BOOT1	BOOT0		
X	0	主闪存存储器	启动地址：0x08000000，是STM32内置Flash。一般使用JTAG或SWD下载程序都是下载到此，重启后从此执行。
0	1	系统存储器	启动地址：0x1FFF0000，系统存储器ROM，其中程序由厂家设置BootLoader。厂家提供的BootLoader引导从串口下载程序到Flash中。
1	1	内置SRAM	启动地址：0x20000000，内置SRAM。一般用于调试。

## 4.4 ARM Cortex-M4 MPU存储保护

- MPU属于Cortex-M4内核的一个外设，它根据Cortex-M4可寻址空间对内存空间定义（分区、地址、大小、属性等）来限制CPU的访问行为，起到保护内存数据的作用。
- MPU位于CPU内核的内存接口各总线内部连接矩阵之间，监视着CPU的一切内存访问行为。若某个程序命令CPU去访问被MPU禁止访问的空间，MPU将会产生一个“内存管理故障”异常。



寄存器	说明
MPU_TYPE	只读，MPU类型
MPU_CTRL	使能MPU
MPU_RNR	分区编号索引
MPU_RBAR	分区基地址配置
MPU_RASR	分区属性配置