普华基础软件

普华基础软件培训

ORIENTAIS NvM

2021.4.12

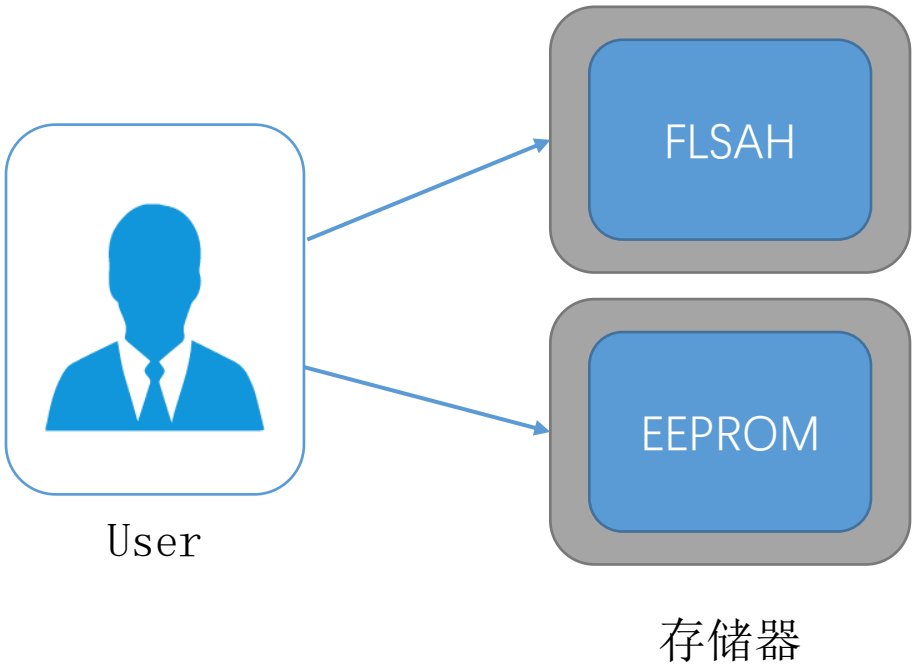
普华基础软件股份有限公司

iSOFT INFRASTRUCTURE SOFTWARE CO., LTD

NvM简介

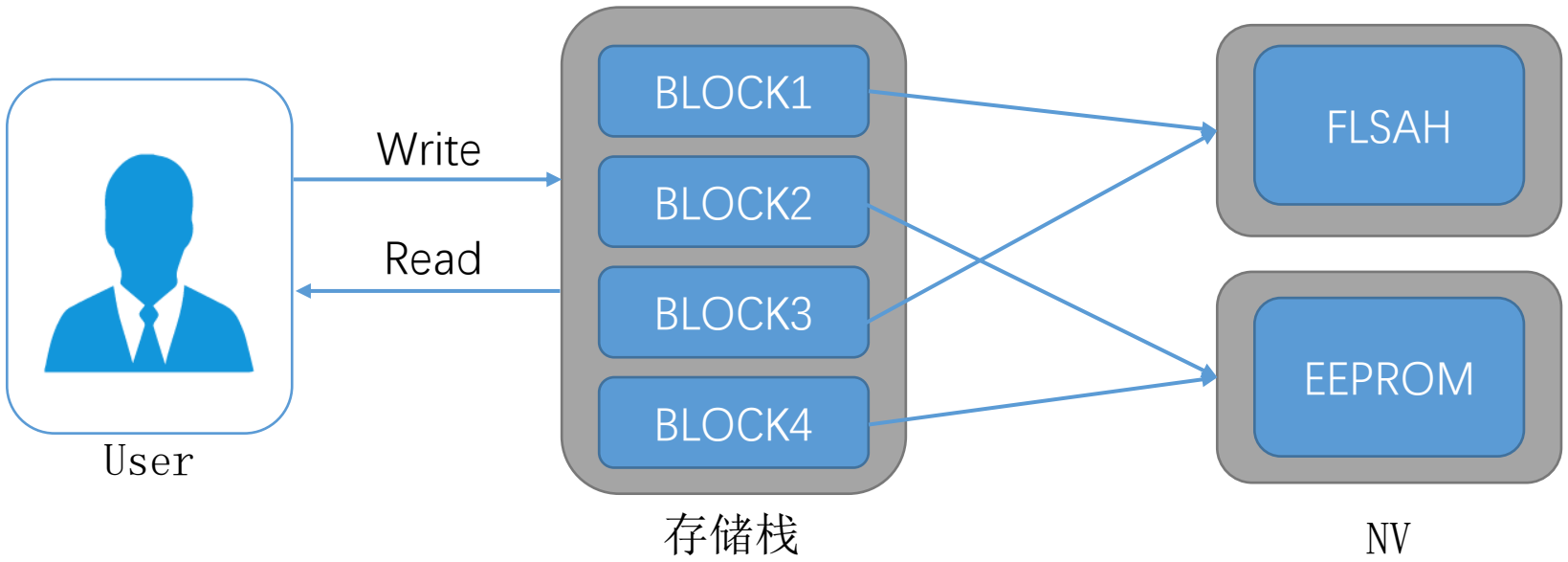
直接对存储器的访问需考虑

- 驱动程序
- 地址管理
- Flash寿命
- 大数据读写时的时间占用



NvM简介

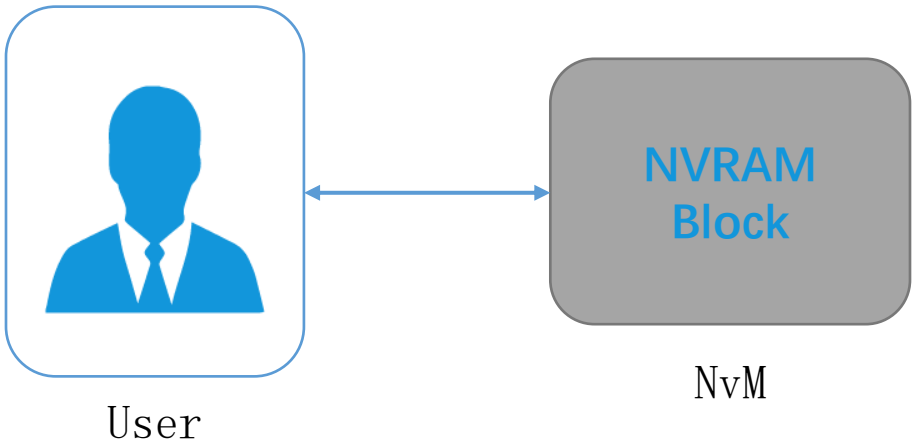
将存储的数据抽象为一个NVRAM Block来管理



NvM简介

存储管理的作用

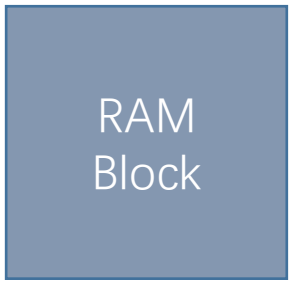
- 掉电后能保存NV数据
- 上电时能从非易失拷贝到RAM
- 应用程序访问是NV数据的拷贝
- 非易失数据能被随时读写
- 异步操作不阻塞其他应用程序
- 用户不关注底层的地址信息
- 支持用户回调接口
- 数据备份与恢复机制等



NvM简介

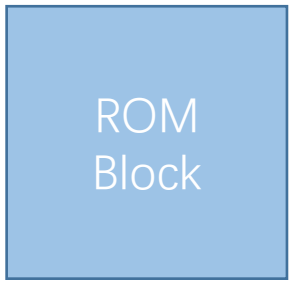
NVRAM Block的基本存储对象（Basic Storage Object）

存储NV data
位于非易失性存储中
Flsah、EEPROM



User data

位于RAM中



提供默认数据
位于ROM区
FLASH

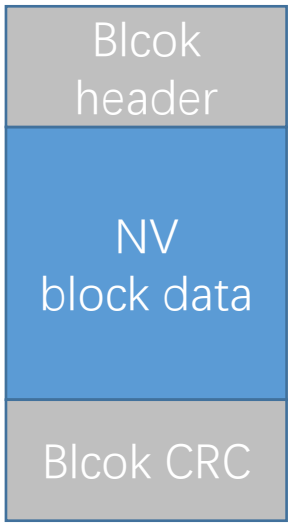
NVRAM状态信息

位于RAM存储中

NvM简介

基本存储对象-NV Block

- Header (可选) - Block ID Check
- NV data - 用户数据
- CRC值 （可选） - CRC比较



NV Block

NvM简介

基本存储对象-RAM Block

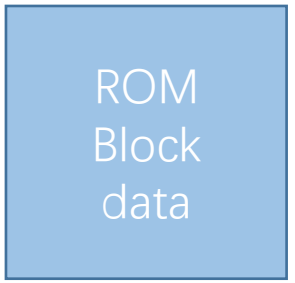
- Header（可选）
- RAM块数据应包含永久或临时分配的用户数据。
- RAM Block CRC值（可选）

RAM Block

NvM简介

基本存储对象-ROM Block

- ROM Block data -当NV block为空或损坏时提供默认数据



ROM Block

NvM简介

基本存储对象-Administrative Block

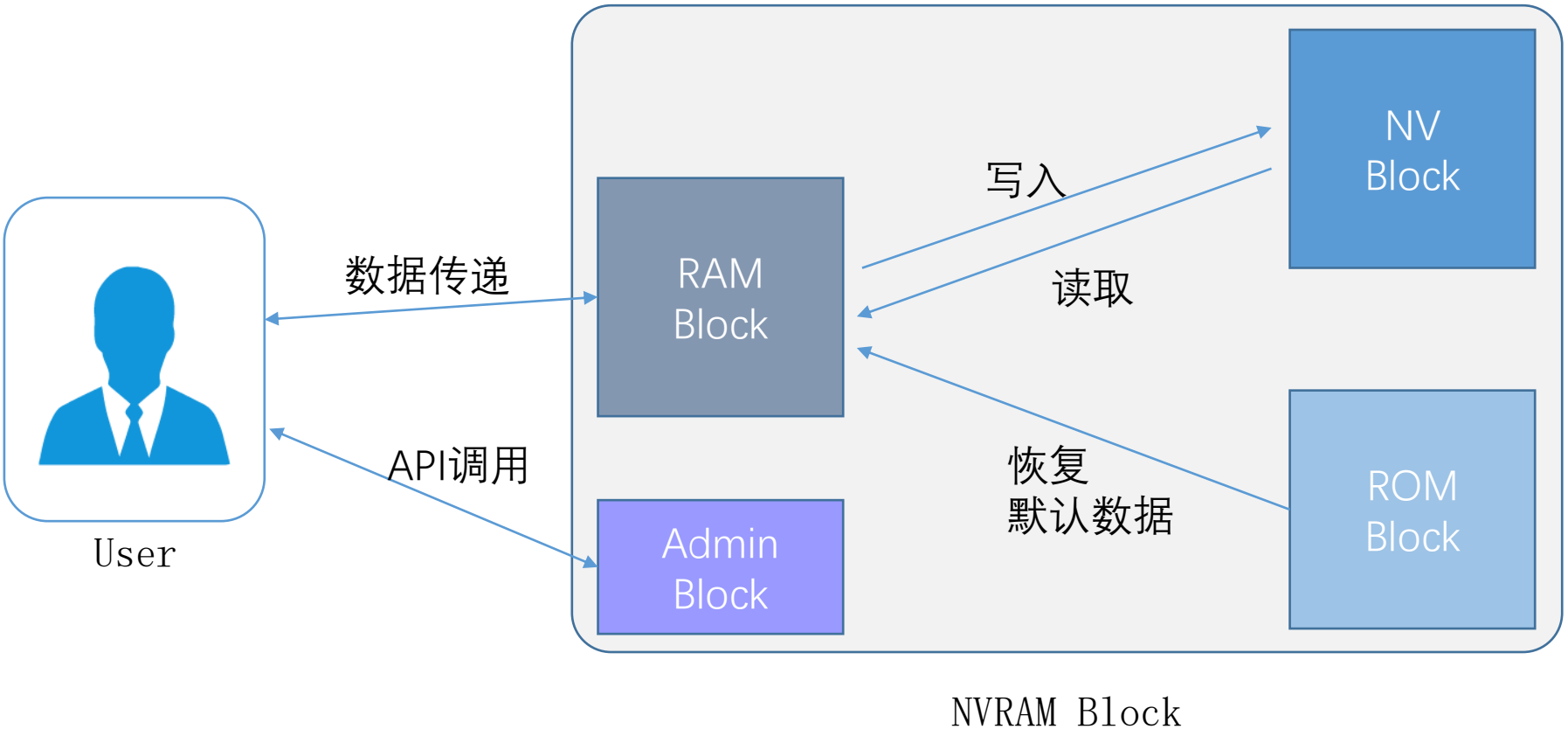
管理块数据包含属性、错误、状态信息包括：

- RAM block的有效/无效状态
- NV block Index
- 写保护状态
- Error/status值（如pending、OK、Not OK）等

Administrative Block

NvM简介

基本存储对象之间的关系示意图



NvM简介

NVRAM Block管理类型

NVRAM Block具有三种管理类型：

- NVM_BLOCK_NATIVE
- NVM_BLOCK_REDUNDANT
- NVM_BLOCK_DATASET

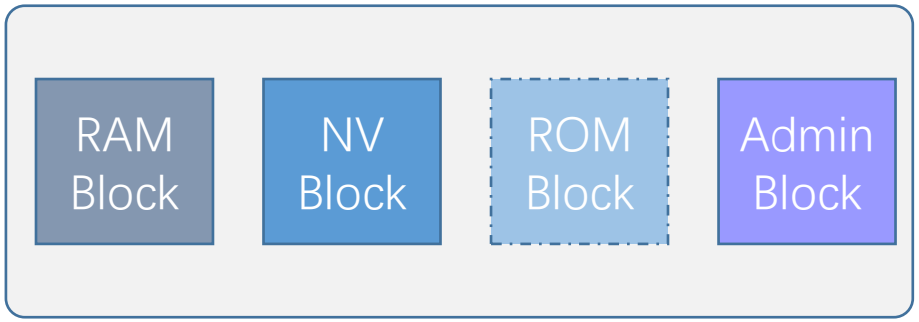
Management Type	NV Blocks	RAM Blocks	ROM Blocks	Administrative Blocks
NVM_BLOCK_NATIVE	1	1	0..1	1
NVM_BLOCK_REDUNDANT	2	1	0..1	1
NVM_BLOCK_DATASET	1..(m<256)	1	0..n	1

NvM简介

NVRAM Block管理类型 - NVM_BLOCK_NATIVE

NVM_BLOCK_NATIVE 表示一种最小开销的属性：

- NV Blocks: 1
- RAM Blocks: 1
- ROM Blocks: 0..1
- Administrative Blocks:1



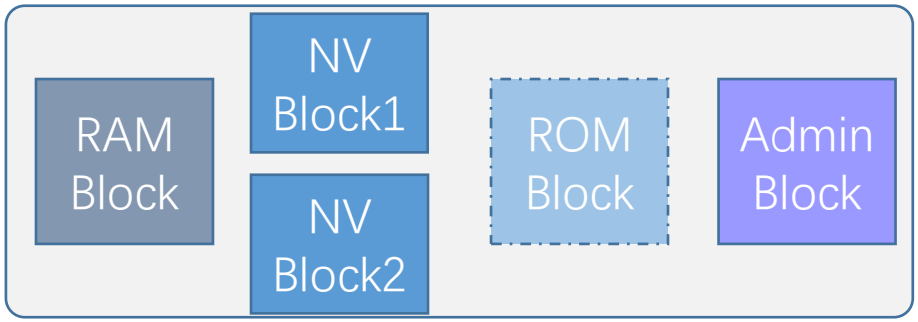
NATIVE Block

NvM简介

NVRAM Block管理类型 - NVM_BLOCK_REDUNDANT

NVM_BLOCK_REDUNDANT 表示一种冗余备份的属性：

- NV Blocks: 2
- RAM Blocks: 1
- ROM Blocks: 0..1
- Administrative Blocks:1



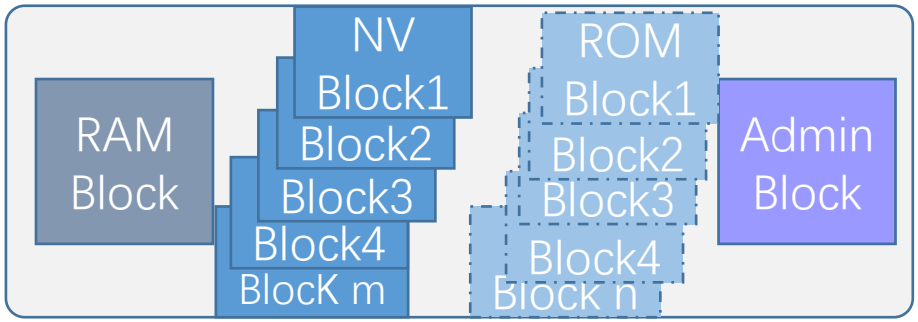
REDUNDANT Block

NvM简介

NVRAM Block管理类型 - NVM_BLOCK_DATASET

NVM_BLOCK_REDUNDANT 表示一种数据集的属性：

- NV Blocks: 1…(256)
- RAM Blocks: 1
- ROM Blocks: 0..n
- Administrative Blocks:1



REDUNDANT Block

NvM简介

NvM API配置等级

API Configuration class3有三种：

- Class3

  所有的API调用均可用,可使用功能最大化

- Class2

  可使用一组中级的API调用集

- Class1

  此配置等级只提供最低限度的API调用集，尤其为了适合非常有限的硬件的资源系统。这个API 调用集任何情况下都必须需要包含的

NvM简介

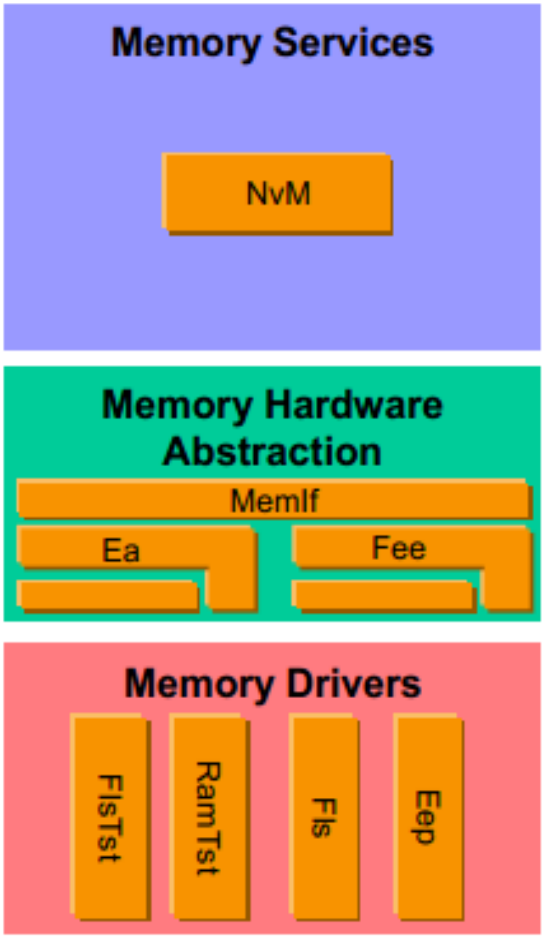
NvM API配置等级

API configuration class 3	*API configuration class 2*	*API configuration class 1*
Type 1: NvM_SetDataIndex (...) NvM_GetDataIndex (...) NvM_SetBlockProtection (...) NvM_GetErrorStatus(...) NvM_SetRamBlockStatus(...)	Type 1: NvM_SetDataIndex (...) NvM_GetDataIndex (...) NvM_GetErrorStatus(...) NvM_SetRamBlockStatus(...)	Type 1: NvM_GetErrorStatus(...) NvM_SetRamBlockStatus(...)
Type 2: NvM_ReadBlock(...) NvM_WriteBlock(...) NvM_RestoreBlockDefaults(...) NvM_EraseNvBlock(...) NvM_InvalidateNvBlock(...)	Type 2: NvM_ReadBlock(...) NvM_WriteBlock(...) NvM_RestoreBlockDefaults(...)	Type 2: --
Type 3: NvM_ReadAll(...) NvM_WriteAll(...) NvM_CancelWriteAll(...)	Type 3: NvM_ReadAll(...) NvM_WriteAll(...) NvM_CancelWriteAll(...)	Type 3: NvM_ReadAll(...) NvM_WriteAll(...) NvM_CancelWriteAll(...)
Type 4: NvM_Init(...)	Type 4: NvM_Init(...)	Type 4: NvM_Init(...)

NvM简介

存储栈各模块介绍

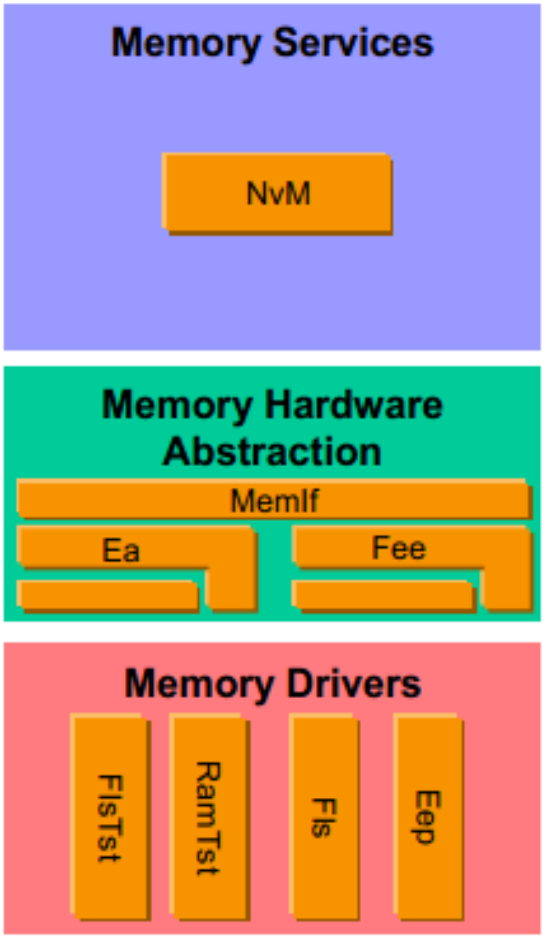
NvM	NVRAM Manager
MemIf	Memory Abstraction Interface
Ea	EEPROM Abstraction
Fee	Flash EEPROM Emulation
Eep	Internal/External EEPROM Driver
Fls	Internal/External Flash Driver



NvM简介

存储栈各模块介绍

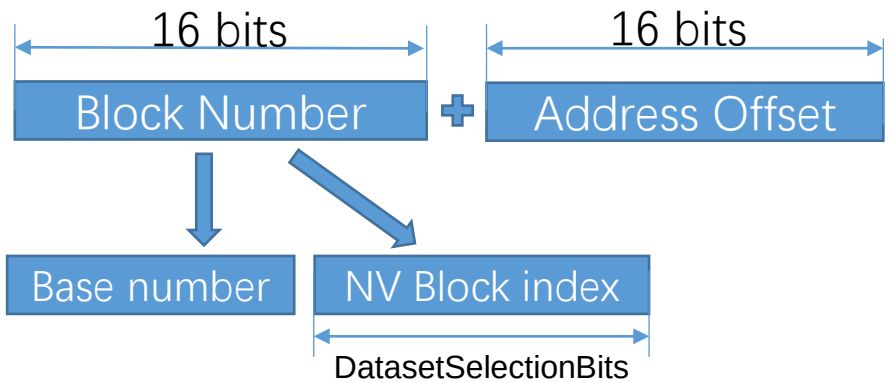
NvM	提供存储管理服务
MemIf	NvM访问不同存储模块的抽象层
Ea	基于EEP的NV Block管理
Fee	基于Flash的NV Block管理
Eep	内部或外部的EEP驱动程序
Fls	内部或外部的Flash驱动程序



NvM简介

NvM 寻址方式

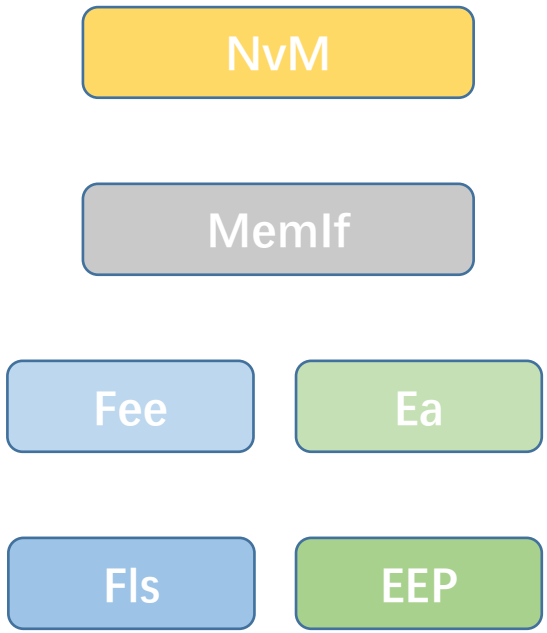
- MemIf、EA、Fee为NvM模块提供虚拟线性32位的地址空间
- 由16位逻block number和16位地址偏移组成
- NvM模块允许(理论上)最大65536个逻辑块
- 每个逻辑块(理论上)的最大大小为64kbytes
- 逻辑块号由NV block base number 和 NvMDatasetSelectionBits组成



NvM简介

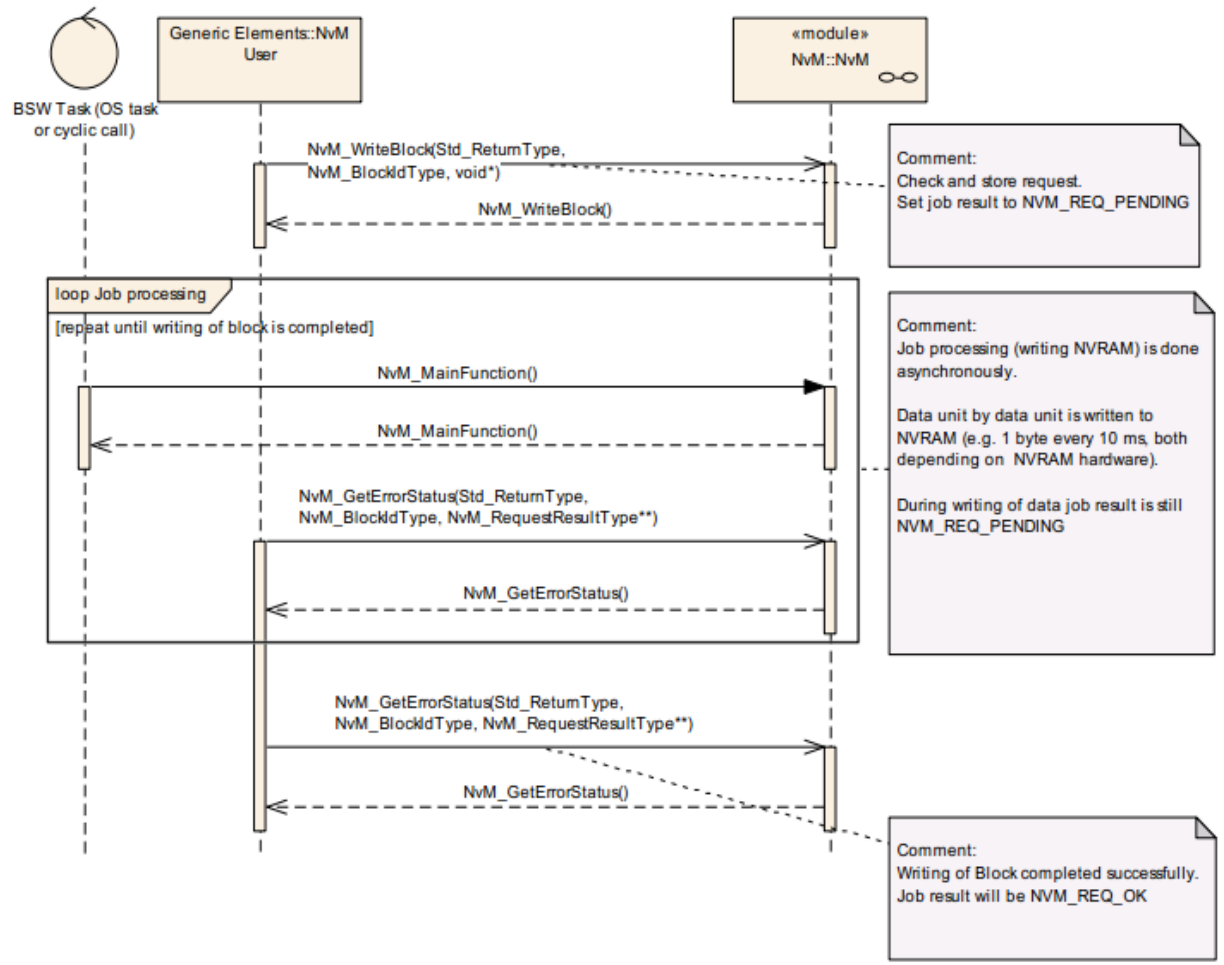
NvM 寻址方式

- NvM配置包含Base number信息
- MemIf配置DeviceIndex来选择相关的存储抽象模块（FEE/EA）
- Fee或Ea根据BlockNumber对NV block进行数据管理，并提供地址偏移信息。
- Fls和EEP仅实现设备的读写擦除的驱动



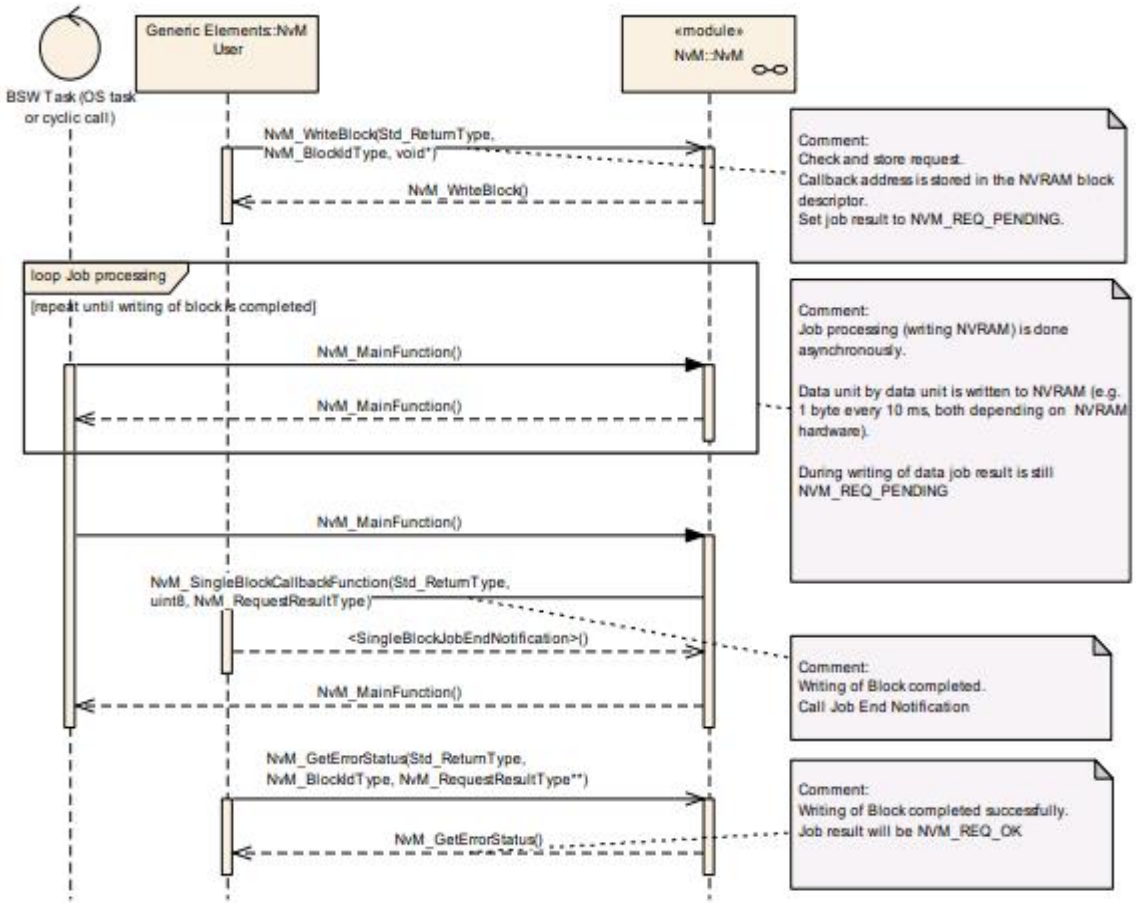
NvM简介

NvM 时序（异步调用-轮询方式）



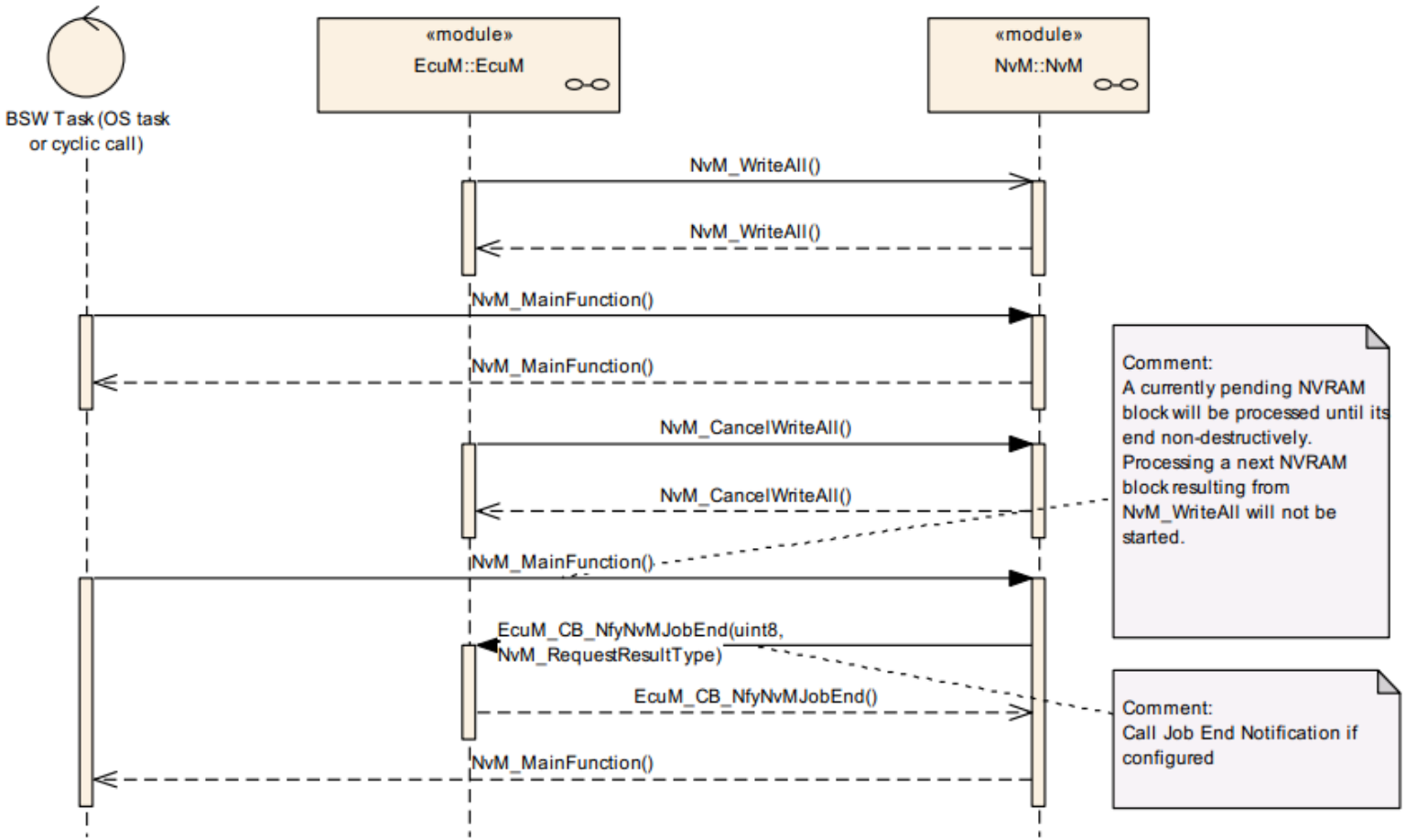
NvM简介

NvM 时序（异步调用–回调方式）



NvM简介

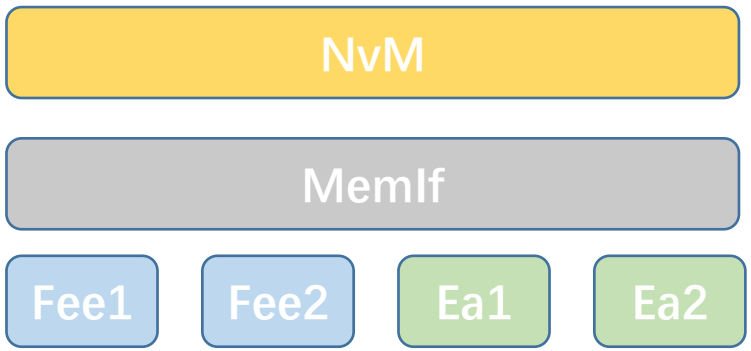
NvM 时序（异步调用–NvM回调）



MemIF简介

概述

- MemIf模块对FLASH或EEPROM存储器的抽象，属于硬件无关层
- MemIf模块允许NVM访问多个存储抽象模块（FEE/EA）
- MemIf模块对下层存储抽象模块（FEE/EA）的数量的抽象
- MemIf模块提供给上层在统一线性地址空间的虚拟分段



MemIF简介

模式选择

- **Fast Mode**
  - 在启动或关闭阶段，底层驱动可以转换到快模式，以允许快速写和擦除
  - 注意：快模式是否可用取决于驱动的实现和底层设备的能力。快模式是否使用取决于NVRAM管理器的配置和特定工程的需要
- **Slow Mode**
  - 在普通操作阶段，底层驱动使用慢模式，以减小底层驱动和通信媒体的运行时间和阻塞时间这方面的资源开销。
  - 注意：慢模式是否可用取决于驱动的实现和底层设备的能力。慢模式是否使用取决于NVRAM管理器的配置和特定工程的需要。

MemIF简介

概述

- MemIf模块所有API都映射到下层存储抽象模块（FEE/EA）
- MemIf模块通过“DeviceIndex”来选择相关的存储抽象模块（FEE/EA）
  - MemIf模块如果只配置了一个存储抽象模块，则通过宏定义来映射下层存储抽象模块（FEE/EA）
  - MemIf模块如果配置了多个存储抽象模块，则可以通过函数指针来映射，通过“DeviceIndex”来选择

Fee简介

概述

- Flash模拟EEPROM（FEE）是对设备特定地址结构和分段的抽象。它为上层提供虚拟的地址结构和分段，并提供无限的擦除周期。

Fee简介

概述

- （Logical）Block－逻辑块

  对模块用户可见，最小可写/可擦单元，

  由1个或多个虚拟页组成

- Virtual page－虚拟页

  由1个或多个物理页组成并且易于逻辑块处理和地址计算

- Virtual Address－虚拟地址

  由16位的块号和16位的偏移组成内部逻辑块

- Physical Address－物理地址

  地址信息在设备指定格式，依赖于驱动和设备

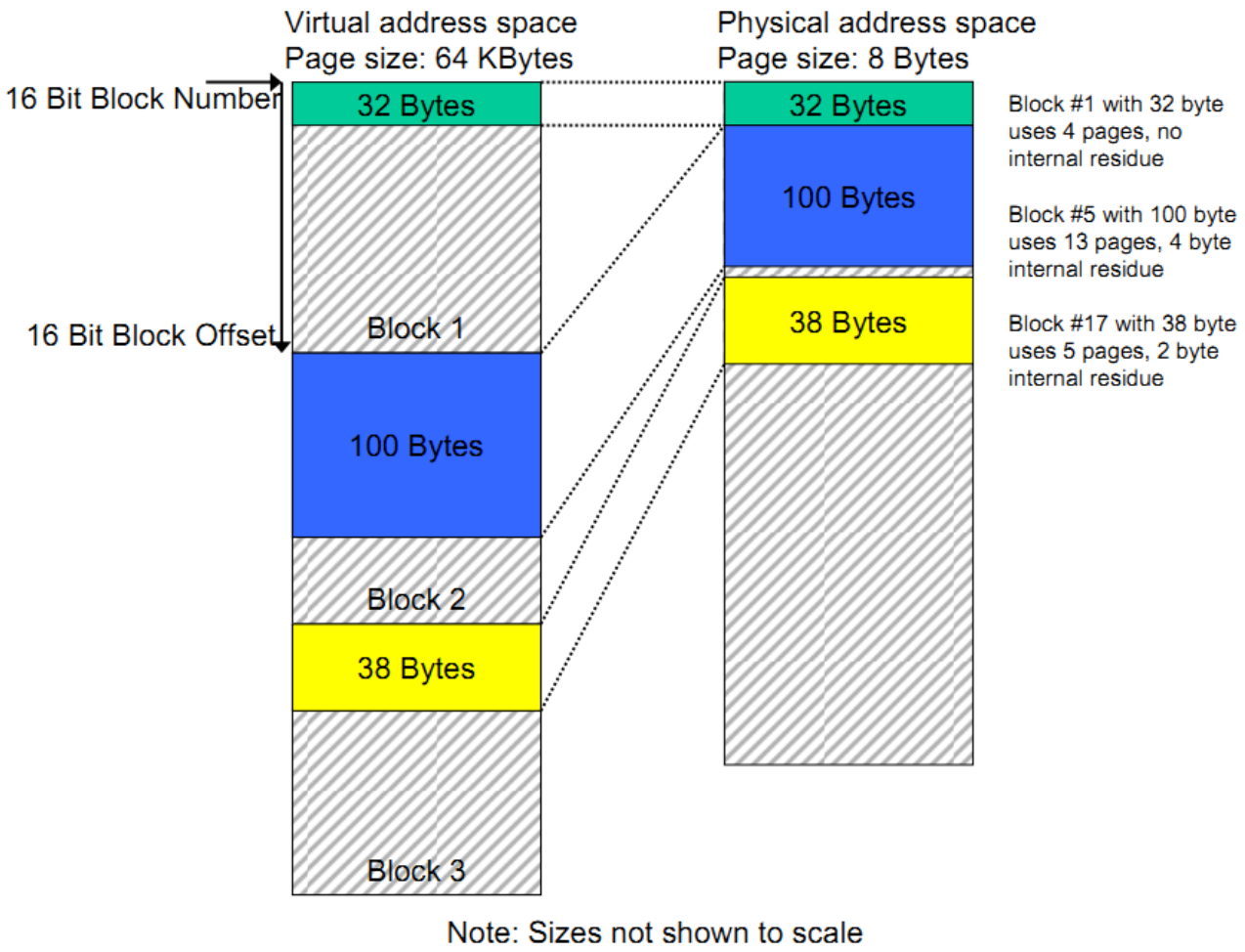
Fee简介

概述 -地址结构和分段

- FEE为上层提供32位虚拟线性地址空间和统一的分段结构:
- 16位块号：允许最大65536逻辑块
- 16位块偏移：允许每一个逻辑块最大64KB
- 通过配置FeeVirtualPageSize定义虚拟页
- 虚拟页的大小应该是物理页大小的整数倍
- 每个配置的逻辑块须为配置虚拟页大小的整数倍
- 逻辑块不允许与其它逻辑块相互重叠，也不许相互包含
- 逻辑块号的值不能配置为0x0000和0xFFFF

Fee简介

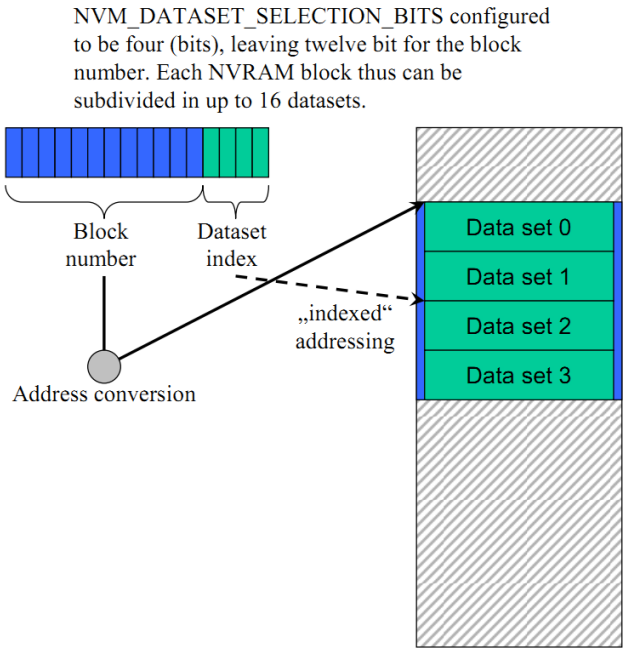
概述 -地址结构和分段



Fee简介

概述 -地址计算

- FEE模块函数应该合成16位的块号和16位的地址偏移来获得访问下层驱动所要用的物理flash地址，FEE模块和正确的地址格式使用取决于实现
- 16位块号位只能用于地址计算，并不能够表示特定的数据集或冗余副本



Fee简介

概述 –立即数

- 包含立即数的逻辑块须被立刻写，例如FEE模块至少应该保证写立即块时不需要再去擦除相应的存储区域（如：使用预擦除内存），并且写请求不能被现在正在运行模块的内部管理操作所延迟

Fee简介

概述 –管理块的一致性

- FEE模块须管理每个块的信息，从FEE模块的角度看该块是否“正确”。这些一致性信息只关心对块的内部操作，而不关心块的内容
- 当一个块的写操作已经开始，FEE模块须标记相应的块为“不一致”。当成功完成块的写操作时，这个块须重新标记为“一致”

Fls(Flash)简介

概述

- FLASH设备驱动提供读、写、擦除等服务，如果硬件支持可以配置写/读的保护设置和重置；
- 在ECU应用模式，只支持通过FEE来对FLS的操作，程序的烧写或BOOT不在此范围；
- FLASH设备驱动对内部FLASH设备直接访问；
- FLASH设备驱动对外部FLASH设备通过地址和数据总线连接（内存映射访问）；

Fls(Flash)简介

基本概念

- **Flash sector**

  Flash存储一次可擦除的最小单位

  Flash扇区的大小取决于硬件的支持

- **Flash page**

  Flash存储一次可烧写的最小单位

  Flash页的大小取决硬件的支持

- **Flash access code**

  在主处理函数中进行写/擦除FLS硬件的内部驱动程序

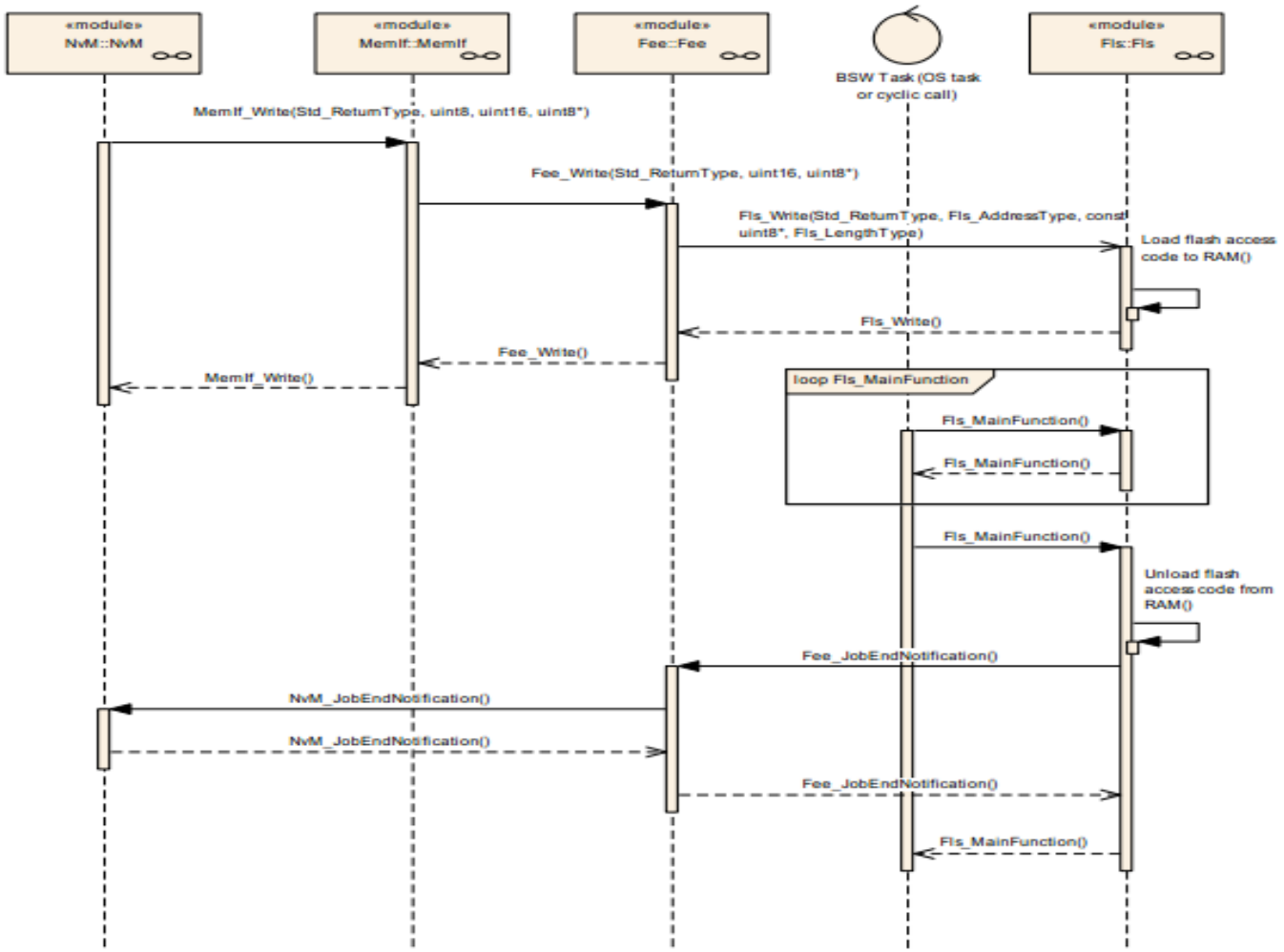
Fls(Flash)简介

功能

- FLS模块提供（读/写/擦除/比较）异步操作服务和（初始化/取消/获取状态/获取工作结果/设置 模式）同步操作服务
- FLS模块不提供缓冲数据，通过API传递指针
- FLS模块将所有的可用的flash内存空间整合成一个线性地址空间(通过FlsBaseAddress和FlsTotalSize参数来指示)
- FLS模块根据存储区域的物理结构，将“虚拟”地址（读、写、擦除和比较函数的地址参数和长度参数）映射到物理地址

Fls(Flash)简介

异步调用时序



EA简介

概述

- EEPROM Abstraction:EA是对设备特定地址结构和分段的抽象。它为上层提供虚拟的地址结构和分段，并提供无限的擦除周期
- EA模块只支持一次只处理一个任务，不提供队列方式处理未决任务

EA简介

概述

- （Logical）Block－逻辑块

  对模块用户可见，最小可写/可擦单元，

  由1个或多个虚拟页组成

- Virtual page－虚拟页

  由1个或多个物理页组成并且易于逻辑块处理和地址计算

- Virtual Address－虚拟地址

  由16位的块号和16位的偏移组成内部逻辑块

- Physical Address－物理地址

  地址信息在设备指定格式，依赖于驱动和设备

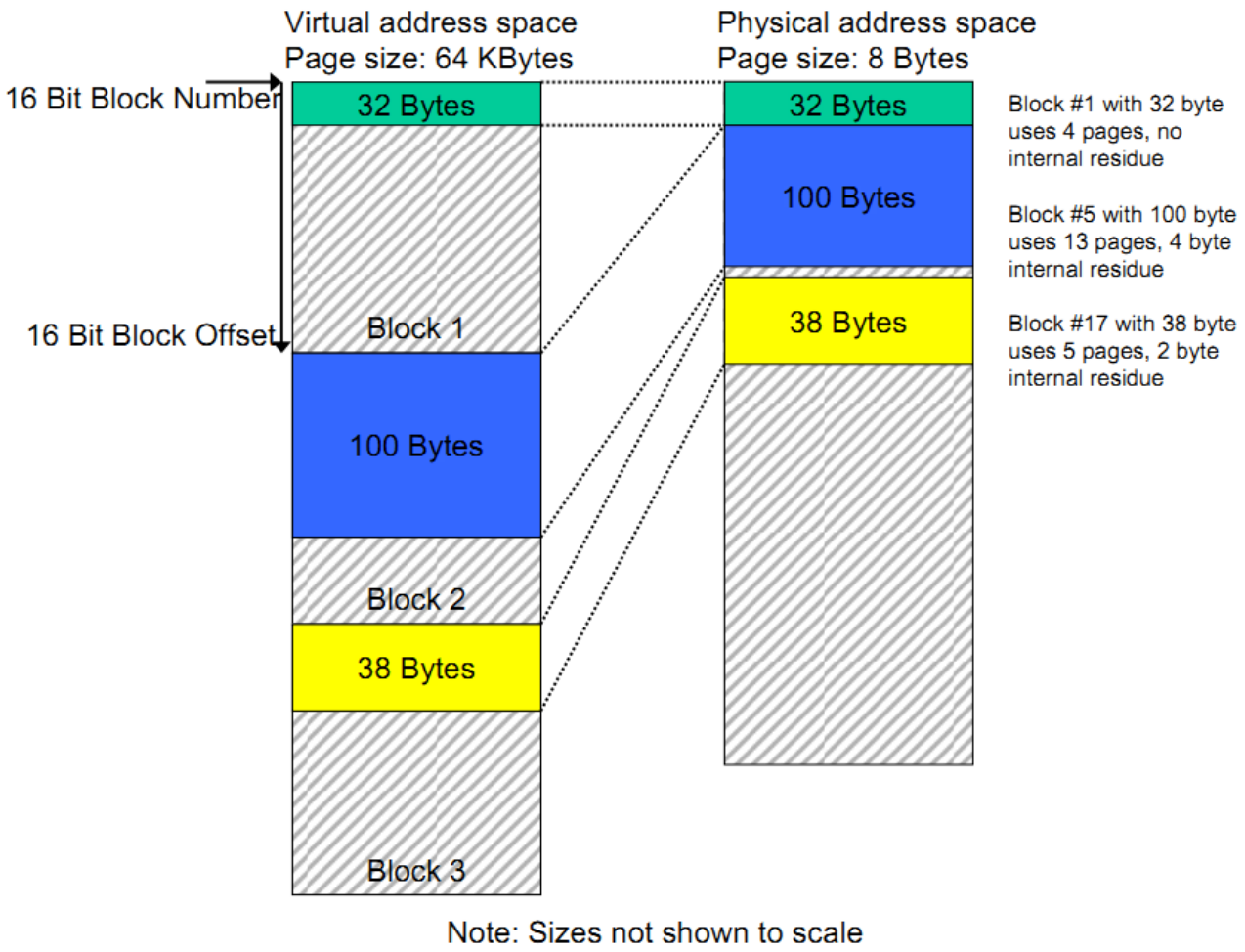
EA简介

概述 -地址结构和分段

- EA为上层提供32位虚拟线性地址空间和统一的分段结构:
- 16位块号：允许最大65536逻辑块
- 16位块偏移：允许每一个逻辑块最大64KB
- 通过配置EA_VIRTUAL_PAGE_SIZE定义虚拟页
- 虚拟页的大小应该是物理页大小的整数倍
- 每个配置的逻辑块须为配置虚拟页大小的整数倍
- 逻辑块不允许与其它逻辑块相互重叠，也不许相互包含
- 逻辑块号的值不能配置为0x0000和0xFFFF

EA简介

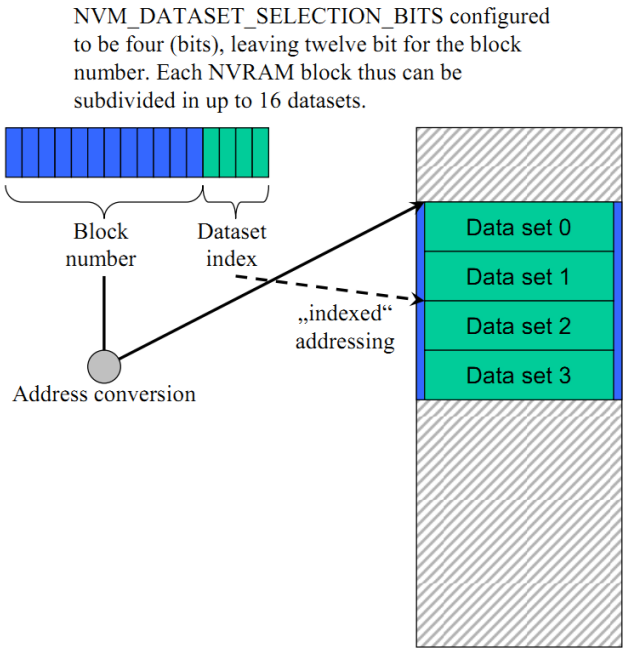
概述 -地址结构和分段



EA简介

概述 -地址计算

- FEE模块函数应该合成16位的块号和16位的地址偏移来获得访问下层驱动所要用的物理flash地址，FEE模块和正确的地址格式使用取决于实现
- 16位块号位只能用于地址计算，并不能够表示特定的数据集或冗余副本



EA简介

概述 –擦除周期

- EA模块的配置须在配置参数EaNumberOfWriteCycles中定义每个逻辑块所期望的擦/写周期
- 如果下层EEPROM设备或者设备驱动不提供配置每个物理存储单元的最小擦/写周期数，EA模块须提供机制来扩展写访问，使物理设备不会超载。这些须应用于EA模块的所有内部管理数据

Fee简介

概述 –立即数

- 包含立即数的逻辑块须被立刻写，例如FEE模块至少应该保证写立即块时不需要再去擦除相应的存储区域（如：使用预擦除内存），并且写请求不能被现在正在运行模块的内部管理操作所延迟

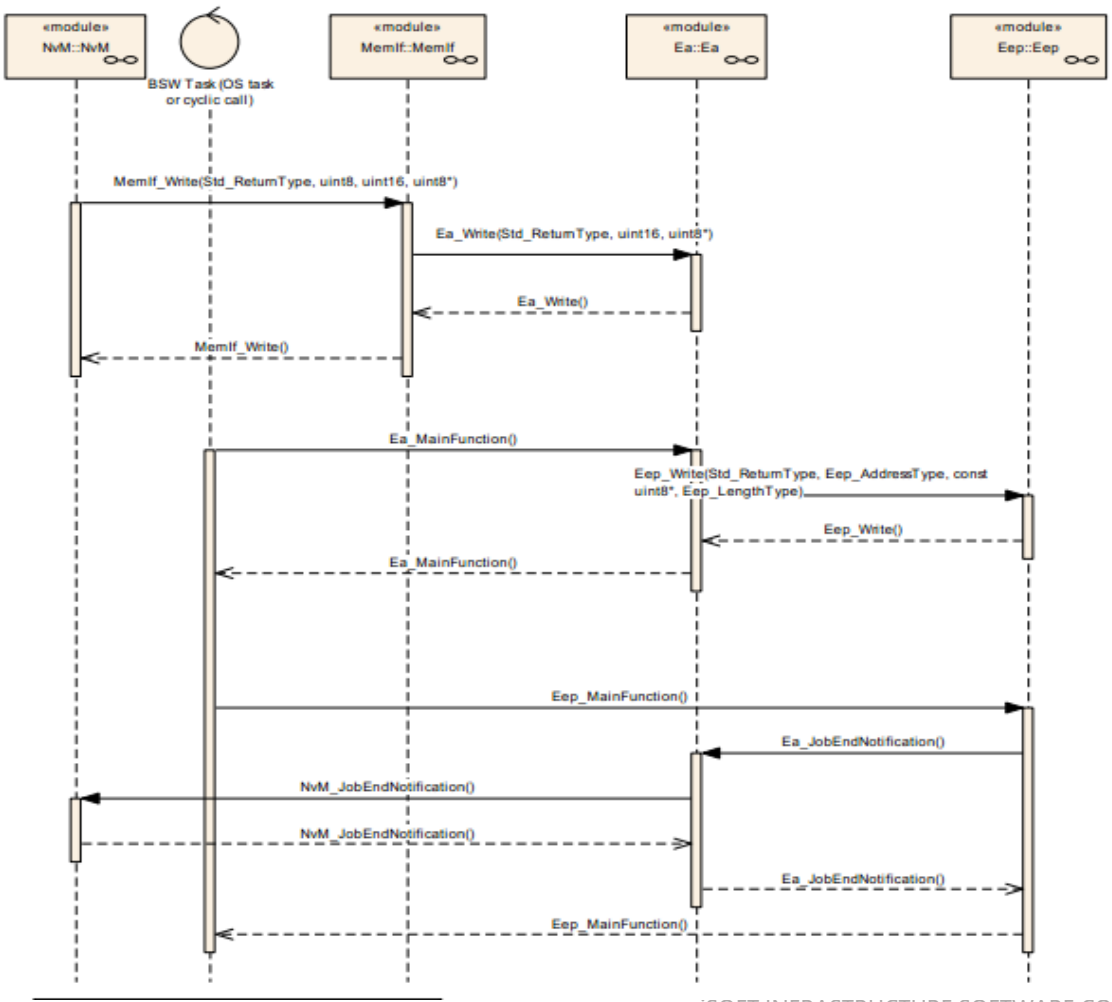
Fee简介

概述 –管理块的一致性

- EA模块须管理每个块的信息，从EA模块的角度看该块是否“正确”。这些一致性信息只关心对块的内部操作，而不关心块的内容
- 当一个块的写操作已经开始，EA模块须标记相应的块为“不一致”。当成功完成块的写操作时，这个块须重新标记为“一致”

EA简介

时序图（写数据）


«module»
NvM::NvM
BSW Task (OS task or cyclic call)
«module»
MemIf::MemIf
«module»
Ea::Ea
«module»
Eep::Eep
MemIf_Write(Std_ReturnType, uint8, uint16, uint8*)
Ea_Write(Std_ReturnType, uint16, uint8*)
Ea_Write()
MemIf_Write()
Ea_MainFunction()
Eep_Write(Std_ReturnType, Eep_AddressType, const uint8*, Eep_LengthType)
Eep_Write()
Ea_MainFunction()
Eep_MainFunction()
Ea_JobEndNotification()
NvM_JobEndNotification()
NvM_JobEndNotification()
Ea_JobEndNotification()
Eep_MainFunction()

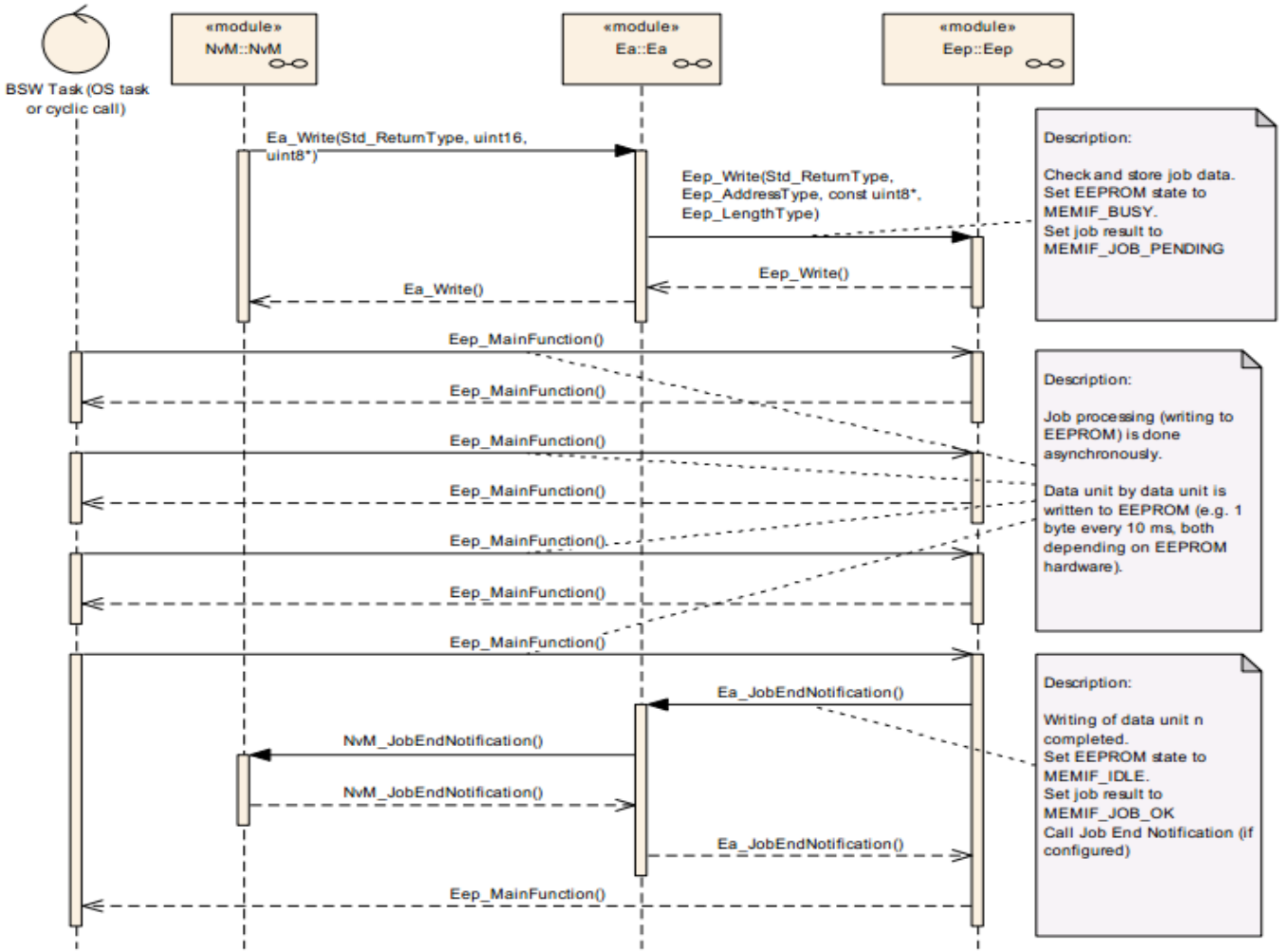
Eep(EEPROM)简介

概述

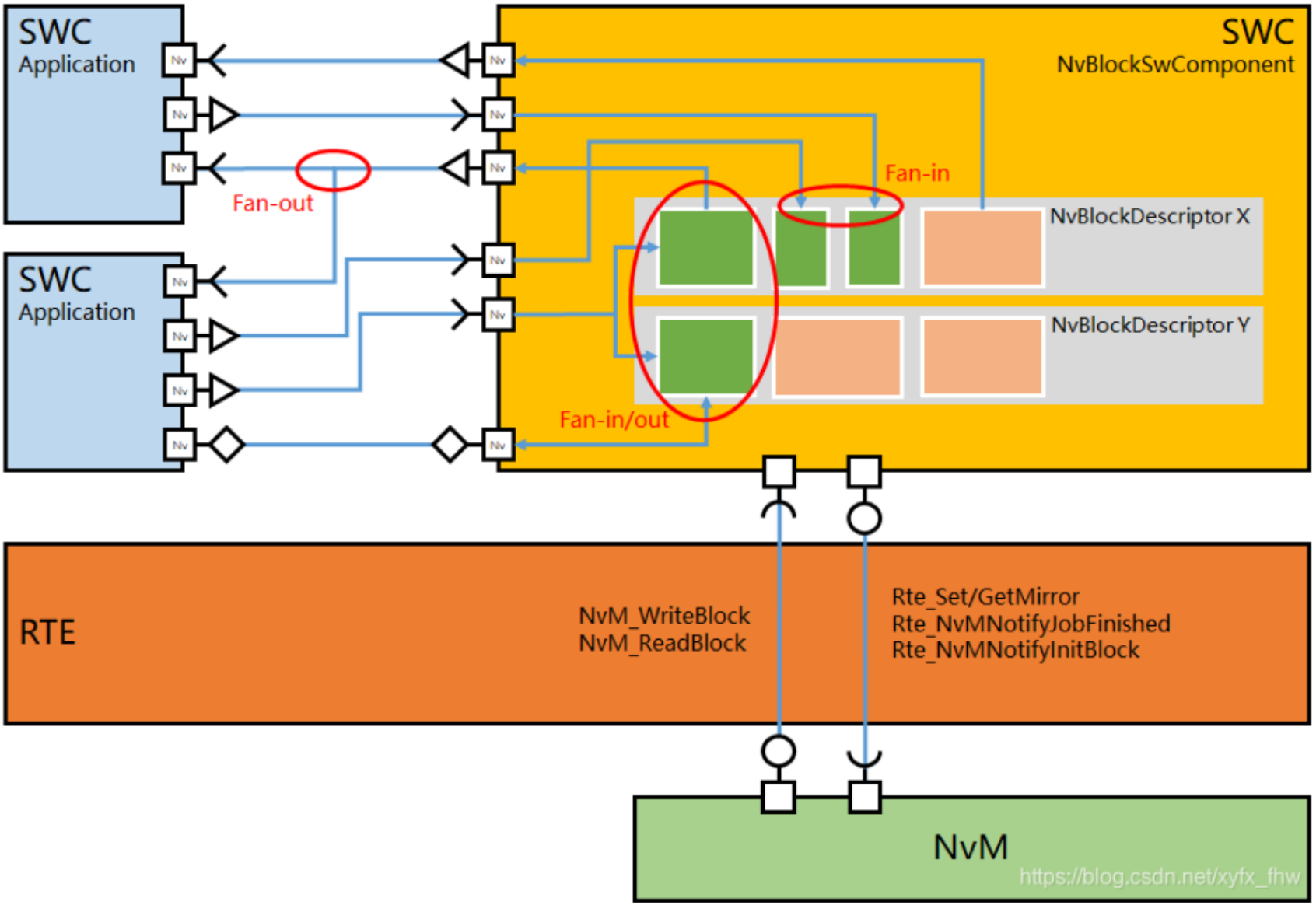
- EEP主要包含内部和外部EEPROM驱动，内部驱动可以直接访问微控器硬件，外部驱动通常使用SPI或IIC来访问EEPROM设备
- EEP提供了读，写，擦除服务。同样提供了比较EEPROM中数据块和内存(如RAM)中数据的服务
- EEP模块提供（读/写/擦除/比较）异步操作服务和（初始化/取消/获取状态/获取工作结果/设置 模式）同步操作服务

Eep(EEPROM)简介

异步时序图



NvM SWC交互



感谢关注！

责任　创新　卓越　共享

网址：www.i-soft.com.cn
信箱：Marketing@i-soft.com.cn
热线：400-650-9325