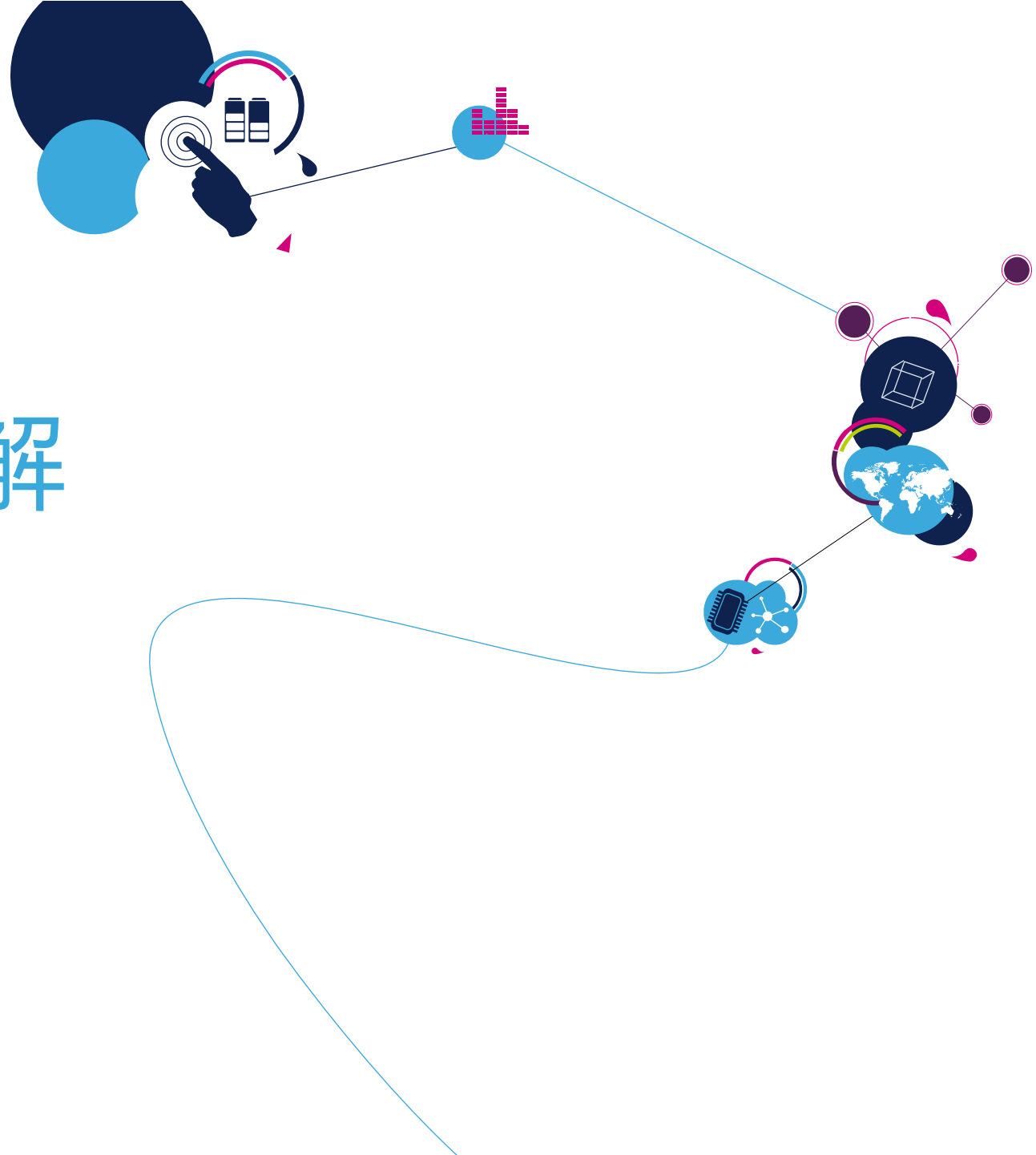


STM32H7技术详解

2019年8月27日

Beijing

life.augmented

Arm® Cortex®-M7 – 480 MHz

CORE, MEMORIES AND ACCELERATION

- Single-core Cortex-M7 480 MHz
- Dual-core Cortex-M7 480 MHz and Cortex-M4 240 MHz (STM32H7x5 and STM32H7x7 only)
- Flash and RAM acceleration
- SP-FPU and DP-FPU
- 4 x DMA

CONNECTIVITY

- 2 x USB2.0 OTG FS/HS
- 2 x SDMMC
- USART, UART, SPI, I²C
- 2 x CAN (1 x FD and 1 x TT)
- HDMI-CEC
- FMC, Dual-mode Quad-SPI
- Ethernet MAC IEEE1588
- Camera I/F
- Analog (comp, AOP)

AUDIO

- 3 x I²S + audio PLL
- 4 x SAI
- 2 x 12-bit DAC
- SPDIF-RX

GRAPHIC

- Chrom-ART Accelerator™

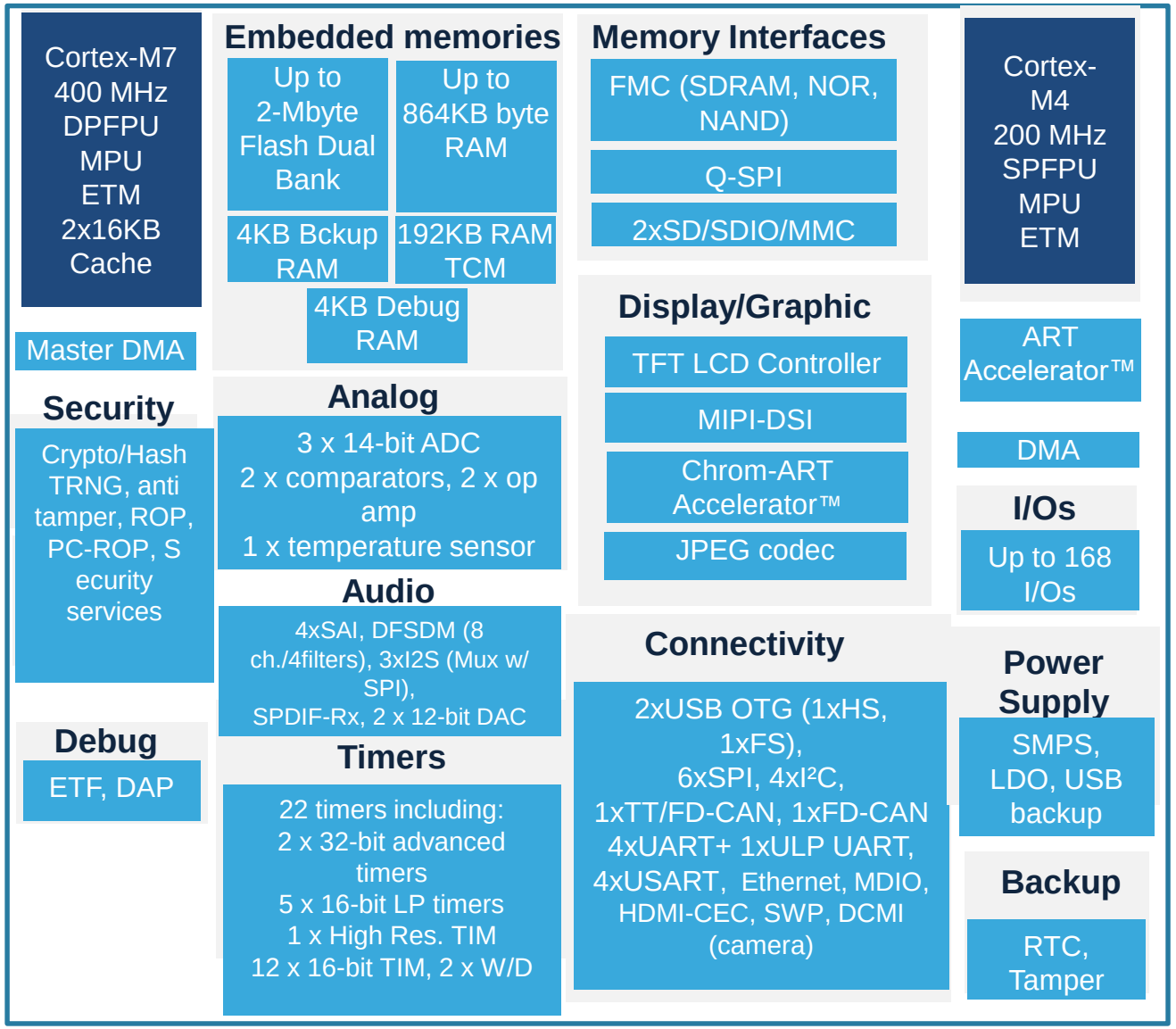
OTHER

- Crypto/Hash (except H742)[1]
- Security services (except H742)
- TRNG
- DFSDM
- 16- and 32-bit timers
- 3 x 16-bit ADC (up to 3.6 MSPS)
- Voltage range 1.62 to 3.6 V (except 100-pin package : 1.71 to 3.6 V)
- Multi-power domains

STM32 H7 Product line	f_{CPU} (MHz)	Dual-Bank Flash memory (bytes)	RAM (bytes)	Graphic	Power supply	Ambiant temperature range
Dual-core lines						
STM32H7x7	480 + 240	Up to 2 Mbytes	1 Mbyte (incl.128 Kbytes DTCM + 64 Kbytes ITCM + 64 Kbytes backup1) + 4 Kbytes backup2	TFT-LCD JPEG codec MIPI-DSI	DCDC + LDO	Standard 85 °C
STM32H7x5	480 + 240	Up to 2 Mbytes	1 Mbyte (incl.128 Kbytes DTCM + 64 Kbytes ITCM + 64 Kbytes backup1) + 4 Kbytes backup2	TFT-LCD JPEG codec	DCDC + LDO	Standard 85 °C (Opt. Industrial CPN 125 °C)[2]
Single-core lines						
STM32H7x3	480	Up to 2 Mbytes	1 Mbyte (incl.128 Kbytes DTCM + 64 Kbytes ITCM + 64 Kbytes backup1) + 4 Kbytes backup2	TFT-LCD JPEG codec	LDO	Standard 85 °C
STM32H742	480	Up to 2 Mbytes	692 K (incl.128 Kbytes DTCM + 64 Kbytes ITCM + 16 Kbytes backup1) + 4 Kbytes backup2	no	LDO	Standard 85 °C
Value line						
STM32H750	480	128 Kbytes	1 Mbyte (incl.128 Kbytes DTCM + 64 Kbytes ITCM + 64 Kbytes backup1) + 4 Kbytes backup2	TFT-LCD JPEG codec	LDO	Standard 85 °C

Notes :
1 : optional - dedicated CPN, STM32H750, STM32H753, STM32H755, STM32H757 for the Crypto Variants
2 : available Q4-2019 - Maximum extended temperature range: 125 °C ambiant / 140 °C junction. Dedicated part numbers

life.augmented


Cortex-M7
400 MHz
DPFPU
MPU
ETM
2x16KB
Cache
Master DMA
Security
Crypto/Hash
TRNG, anti
tamper, ROP,
PC-ROP, S
ecurity
services
Debug
ETF, DAP
Embedded memories
Up to
2-Mbyte
Flash Dual
Bank
Up to
864KB byte
RAM
4KB Bckup
RAM
192KB RAM
TCM
4KB Debug
RAM
Analog
3 x 14-bit ADC
2 x comparators, 2 x op
amp
1 x temperature sensor
Audio
4xSAI, DFSDM (8
ch./4filters), 3xI2S (Mux w/
SPI),
SPDIF-Rx, 2 x 12-bit DAC
Timers
22 timers including:
2 x 32-bit advanced
timers
5 x 16-bit LP timers
1 x High Res. TIM
12 x 16-bit TIM, 2 x W/D
Memory Interfaces
FMC (SDRAM, NOR,
NAND)
Q-SPI
2xSD/SDIO/MMC
Display/Graphic
TFT LCD Controller
MIPI-DSI
Chrom-ART
Accelerator™
JPEG codec
Connectivity
2xUSB OTG (1xHS,
1xFS),
6xSPI, 4xI²C,
1xTT/FD-CAN, 1xFD-CAN
4xUART+ 1xULP UART,
4xUSART, Ethernet, MDIO,
HDMI-CEC, SWP, DCMI
(camera)
Cortex-
M4
200 MHz
SPFPU
MPU
ETM
ART
Accelerator™
DMA
I/Os
Up to 168
I/Os
Power
Supply
SMPS,
LDO, USB
backup
Backup
RTC,
Tamper

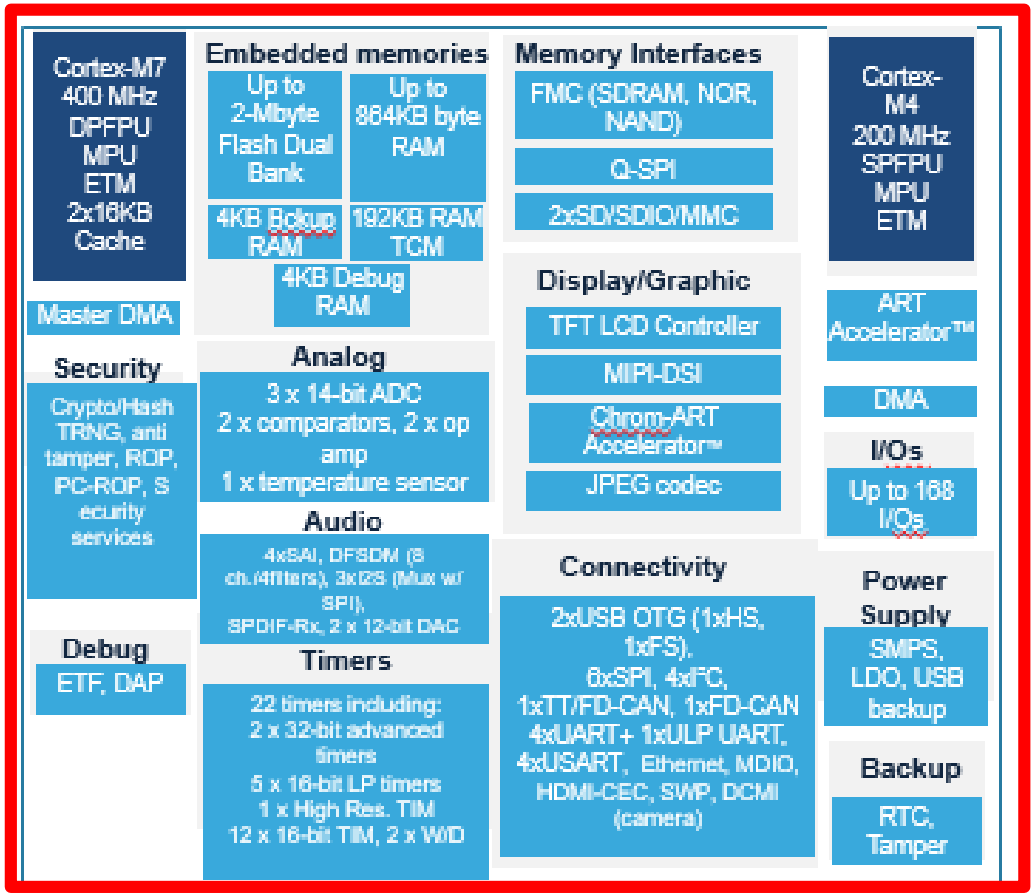
内容安排

- STM32H7 系统架构
  - 单核及双核STM32H7系统框图
  - AXI 总线及互联
  - ART 加速器
- STM32H7 存储器 结构
- STM32H7 Cache
- STM32H7 MPU
- STM32H7 启动方式
- *STM32H7-CPUs 概述*
  - *STM32H7 性能*
- STM32H7 双核优势和应用
- STM32H7 双核的同步和通信
  - IPC
  - *HSEM*
- STM32H7 电源管理
- STM32H7外设
  - DMA
  - 16位 ADC
  - LTDC MIPI 控制器
  - DLYB
  - FMC
- STM32H7目前的资源

目标

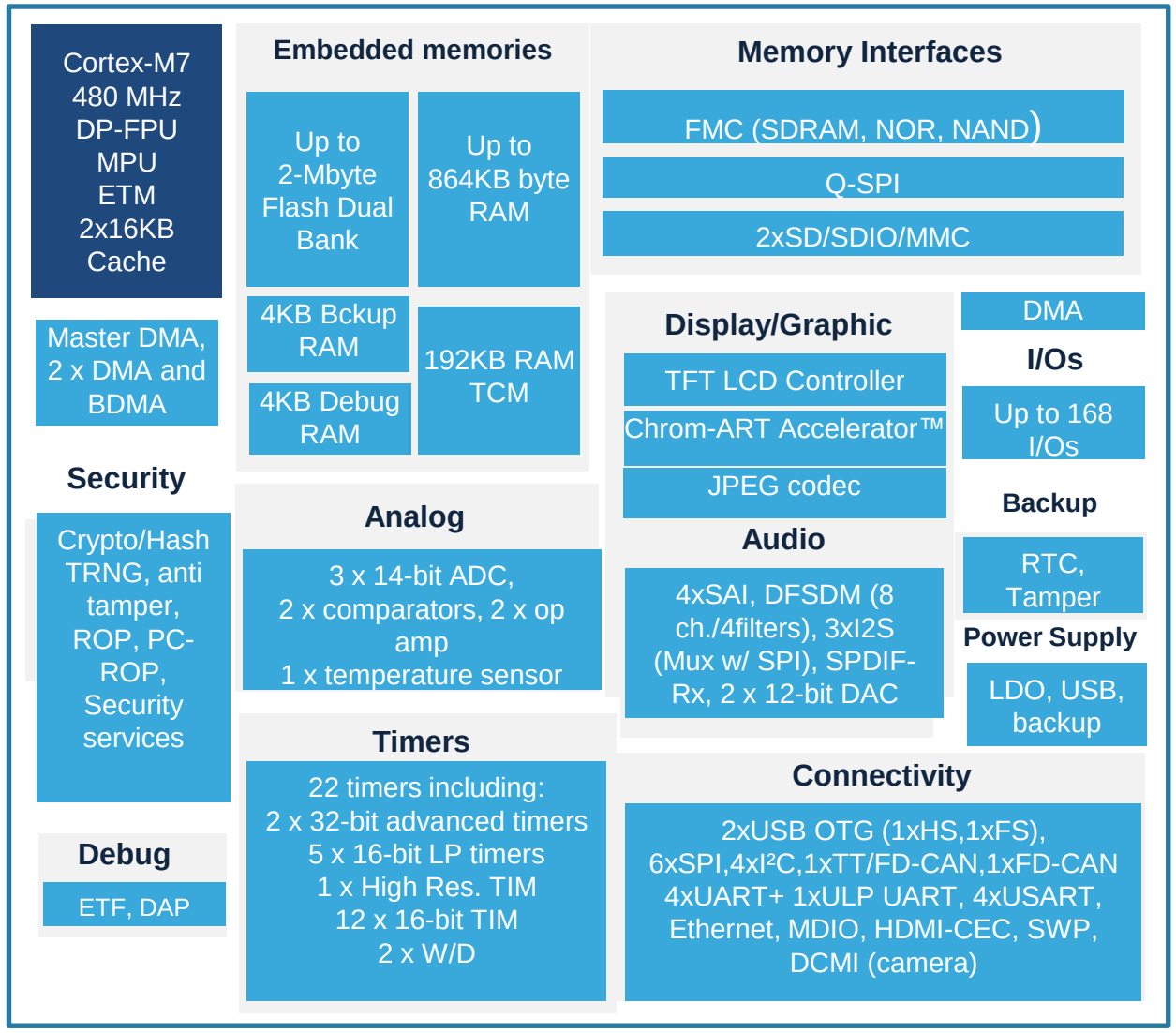
- 探讨STM32H7双核的系统架构，关键特点
- 结束的时候你将能够
  - 理解STM32H7单核及双核产品的特性，及H7的关键点
  - 安装开发工具，使用STM32Cube从头建立工程，进行H7高性能的demo，以及双核功能的调试

STM32H7 系统架构



STM32H7x3 单核 框架图

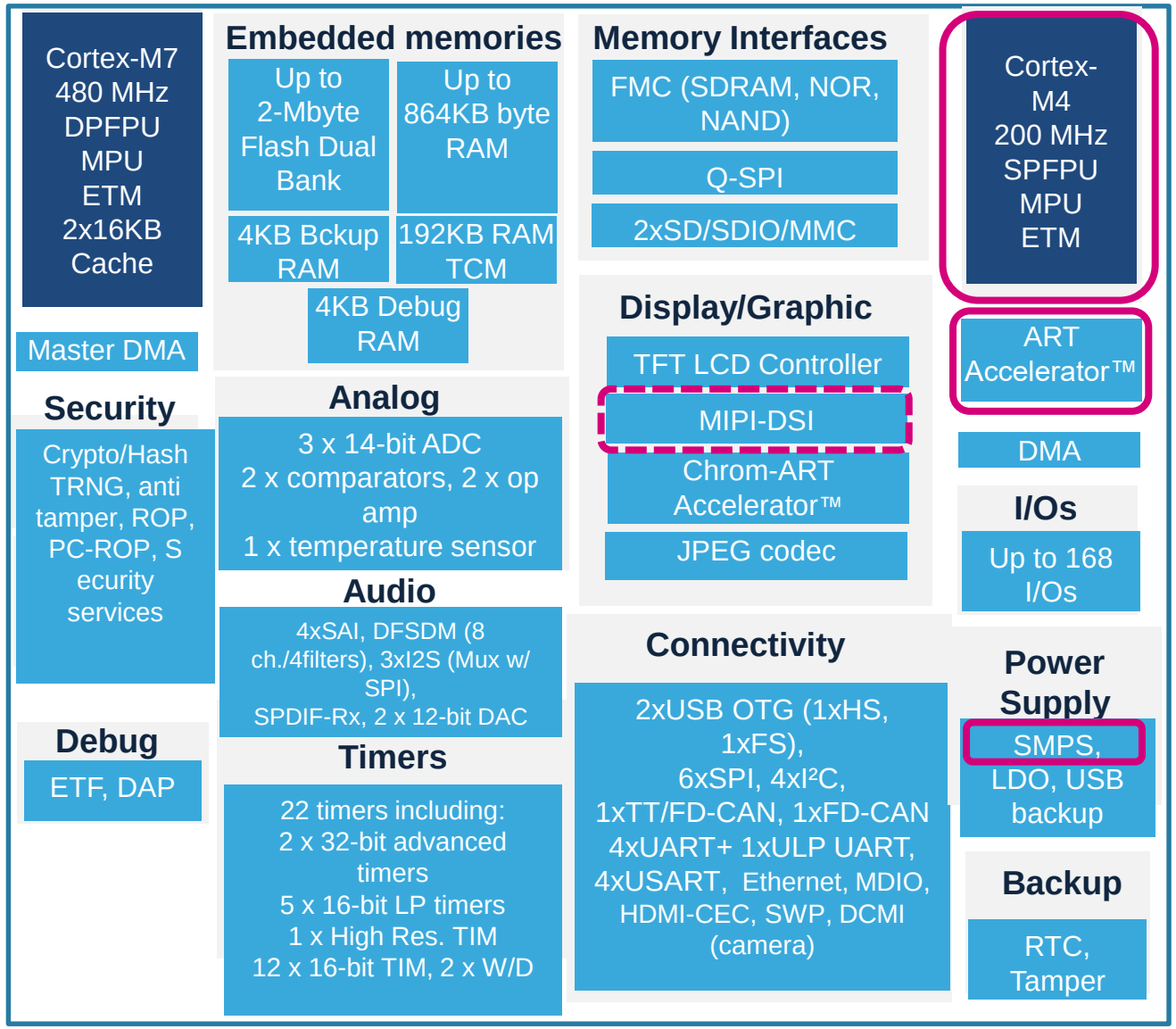
- H7长什么样子?
  - 40nm工艺
  - Cortex®-M7 内核，主频 480 MHz *V
    - 16 Kbytes I-$&D-$
    - 16 MPU 区域
    - 支持单精度和双精度 FPU
  - 总线上：3 个互联矩阵
  - 4 个DMA 控制器，减轻CPU负载
  - 高达35 通信外设
  - 11 个模拟外设
  - 高达 22个 timers 和 watchdogs
  - 加强的时钟管理功能
  - 跟STM32F7和STM32F4引脚pin-to-pin兼容



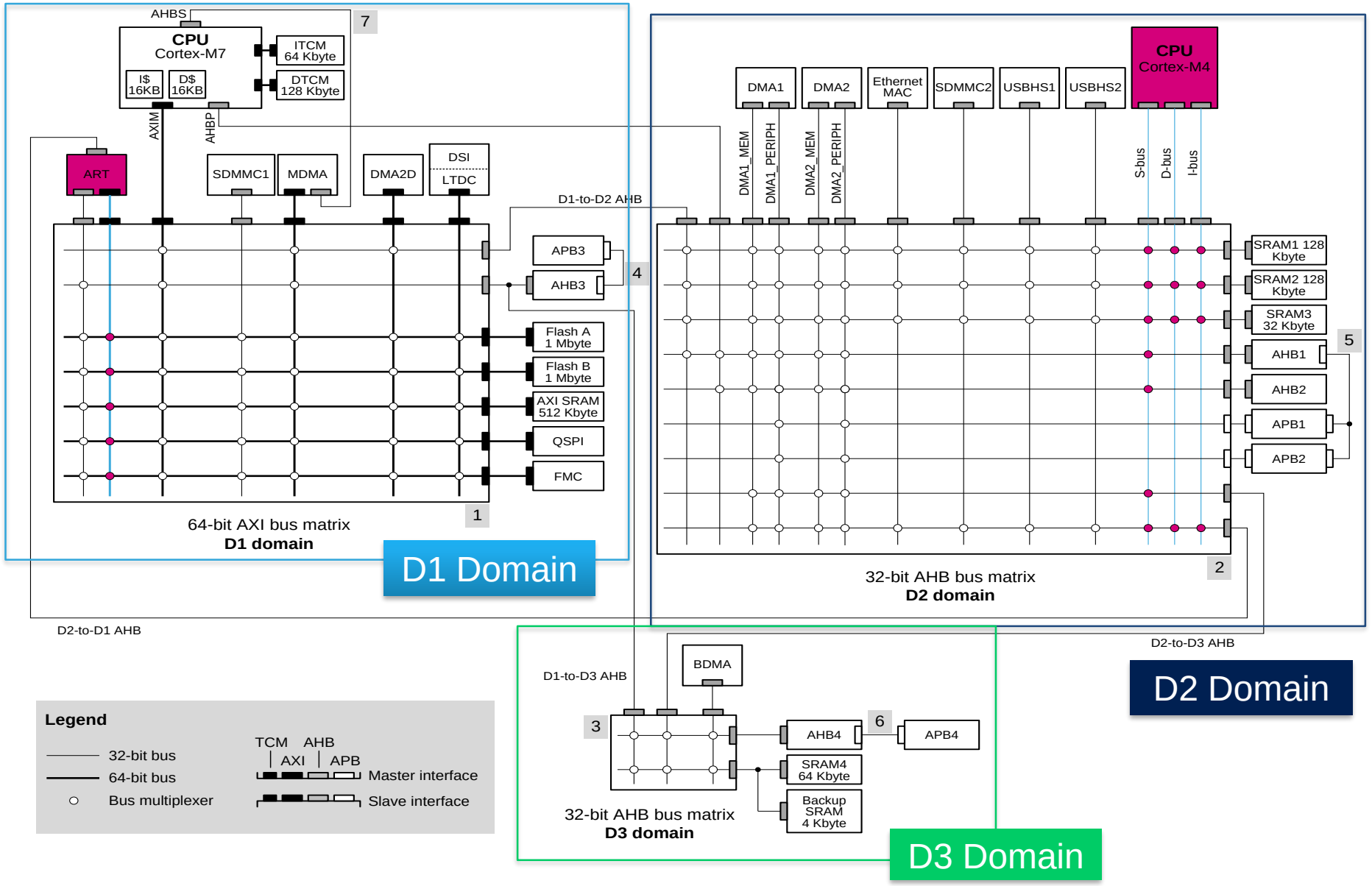
life.augmented

STM32H7x5/x7 双核 框架图

- 增加了什么?
  - Cortex®-M7 内核，主频 480 MHz
  - Cortex®-M4 内核，主频 240 MHz
  - 以及M4内核区域的ART加速器
  - 总线上：3 个互联矩阵
  - 图形上:
    - MIPI DSI host (只有在STM32H7x7 系列)
  - 供电上:
    - 内部的开关电源（SMSP）



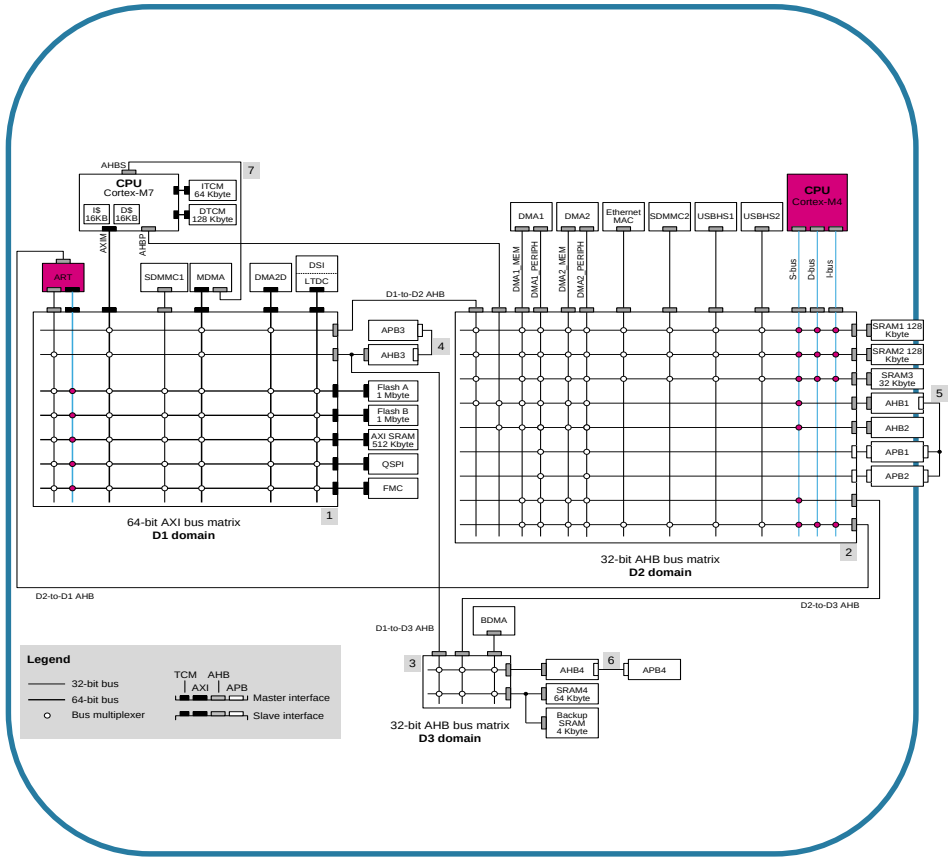
系统架构总览



life.augmented

概述

- 主设备和从设备之间的内部连接是通过：
  - 一个AXI总线矩阵，
  - 两个AHB总线矩阵，
  - 总线之间的桥；



应用好处

- 高效的系统和高速外围设备可以同时操作
- 公平仲裁与循环算法
- 使用QoS能力进行AXI仲裁

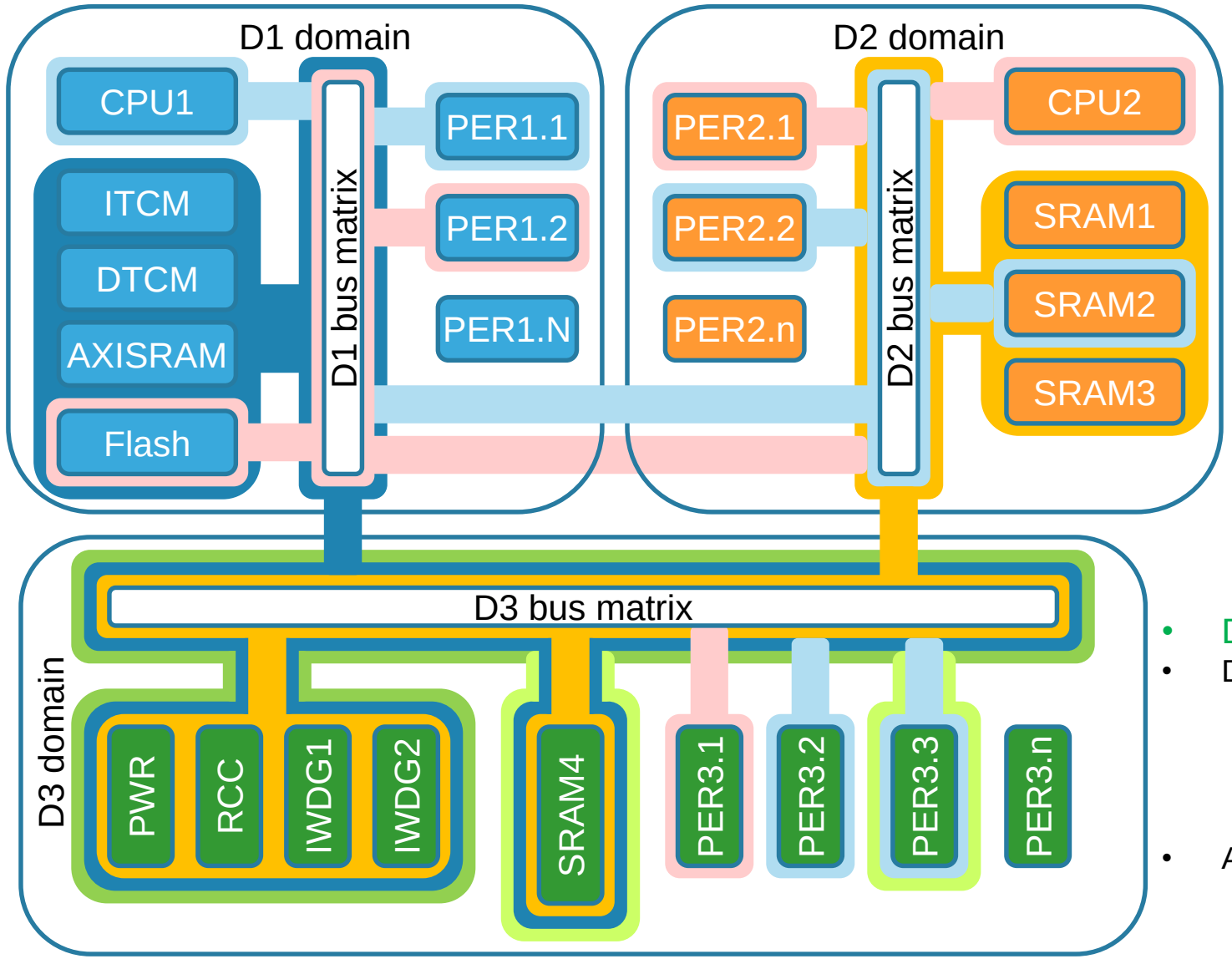
系统架构: 总线互联

互联性	20+ 个总线的主设备

• 总线矩阵
  • 在D1域的AXI 总线矩阵
  • 在D2和D3域的AHB总线矩阵
• Bus-to-bus 桥
  • 5 个 AHB-to-APB, 2 个 AXI-to-AHB
• 不同域之间的总线
  • D2-to-D1 AHB
  • D1-to-D2 AHB
  • D1-to-D3 AHB
  • D2-to-D3 AHB
	• CPU(s) 总线
	• 高性能外设
	  • SDMMC1, SDMMC2
	  • MDMA 控制器
	  • DMA1 和 DMA2 控制器
	  • BDMA 控制器
	  • Chrom-Art Accelerator™ (DMA2D)
	  • LCD-TFT 控制器 (LTDC)
	  • USBHS1 和 USBHS2 外设

系统架构的定义

- CPU1 子系统
- D1 Bus matrix
  - ITCM
  - DTCM
  - AXISRAM
  - Flash
- D3 Bus matrix
  - PWR
  - RCC
  - IWDG1
  - IWDG2
  - SRAM4
- Allocated peripherals
  - PER1.1
  - PER2.2
  - PER3.2
  - PER3.3
  - SRAM2

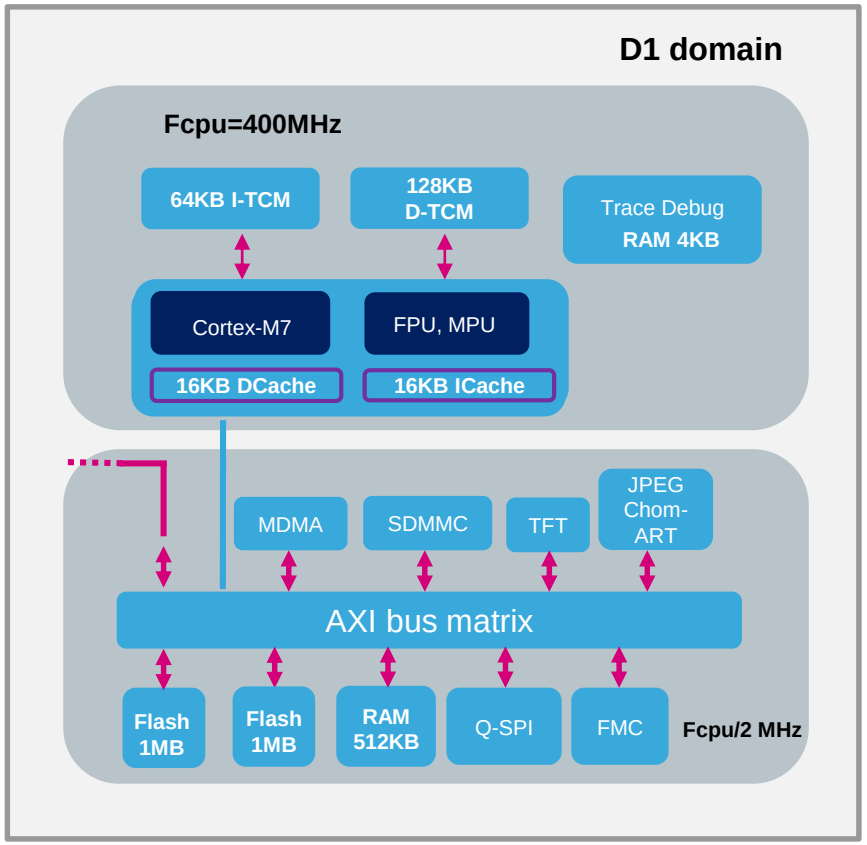


- CPU2 子系统
- D2 Bus matrix
  - SRAM1
  - SRAM2
  - SRAM3
- D3 Bus matrix
  - PWR
  - RCC
  - IWDG1
  - IWDG2
  - SRAM4
- Allocated peripherals
  - Flash
  - PER1.2
  - PER2.1
  - PER3.1

- D3 自主域
- D3 Bus matrix
  - PWR
  - RCC
  - IWDG1
  - IWDG2
- Allocated peripherals
  - SRAM4
  - PER3.3

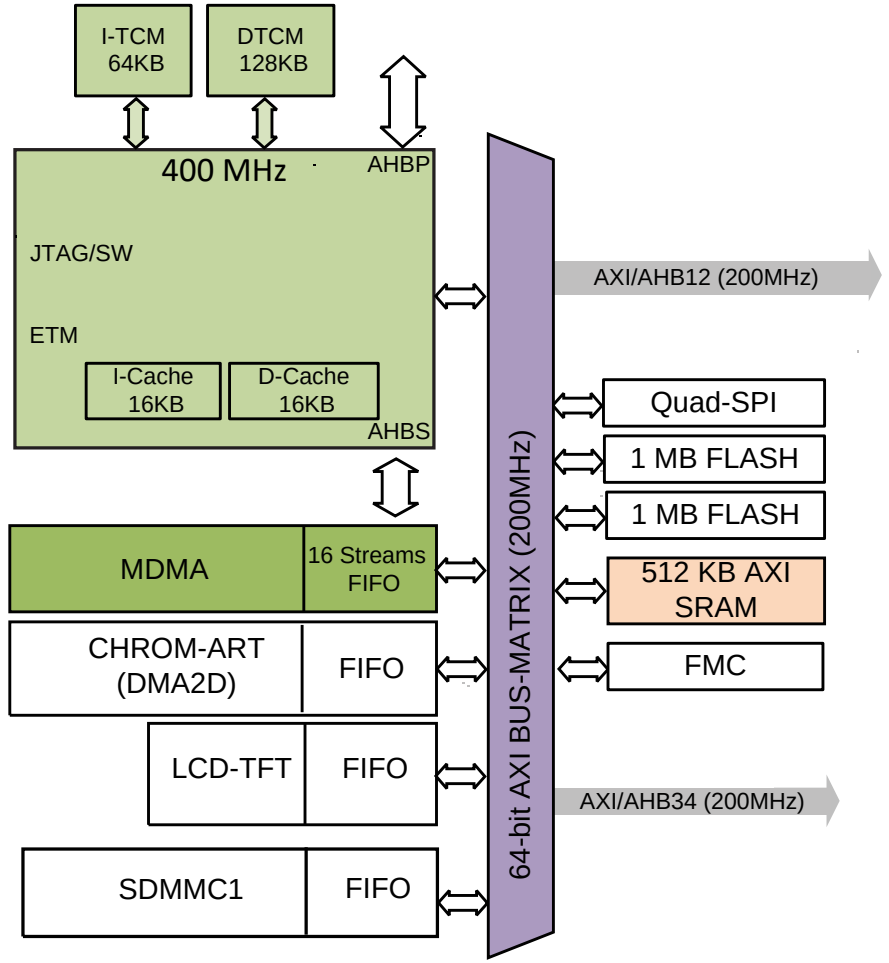
D1: 处理域

- D1域提供了MCU处理和图形的功能
- Cortex-M7 内核
  - 运行在480MHz
  - 16kB I-cache, 16kB D-cache
- AXI interconnect : 最大频率 240MHz
  - AHB 总线最大频率 240MHz
  - APB 总线最大频率 120MHz
- 图形之间
  - JPEG Codec
  - DMA2D
  - LTDC/DSI 控制器



D1: 处理域

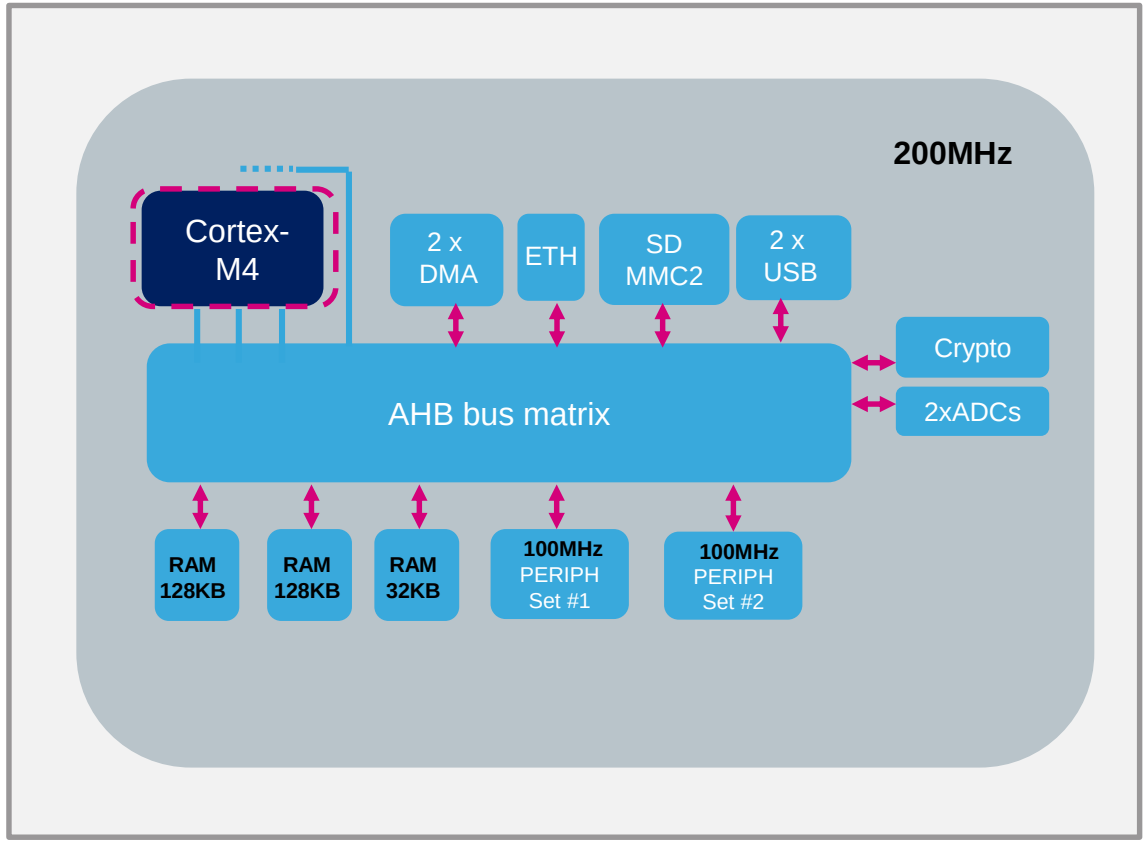
- Embedded Flash
  - 2x 1MB 带ECC的独立 Flash banks
    - Flash，每256位word加10位ECC校验位。
  - 带ECC的 SRAM (512kB + 128kB + 64kB)
    - SRAM，每32位 word加7位 ECC校验。
    - ITCM-RAM 和 AXI-SRAM，每64位 word 加8位 ECC校验。
  - ECC 一位检测一位纠错，或者两位检测
- 外部存储器
  - FMC (SRAM, NAND Flash, SDRAM, …)
  - QuadSPI
  - SD/MMC



life.augmented

D2: 外设域

- D2域提供了外设管理
- Cortex-M4 内核运行在 240MHz
- AHB 互联：
  - 最大频率 240MHz
  - APB 总线最大频率 120MHz
- 外设集
  - DMA1/2
  - 2xUSB, ETH
  - CAN, Timer, SPI, UART, …

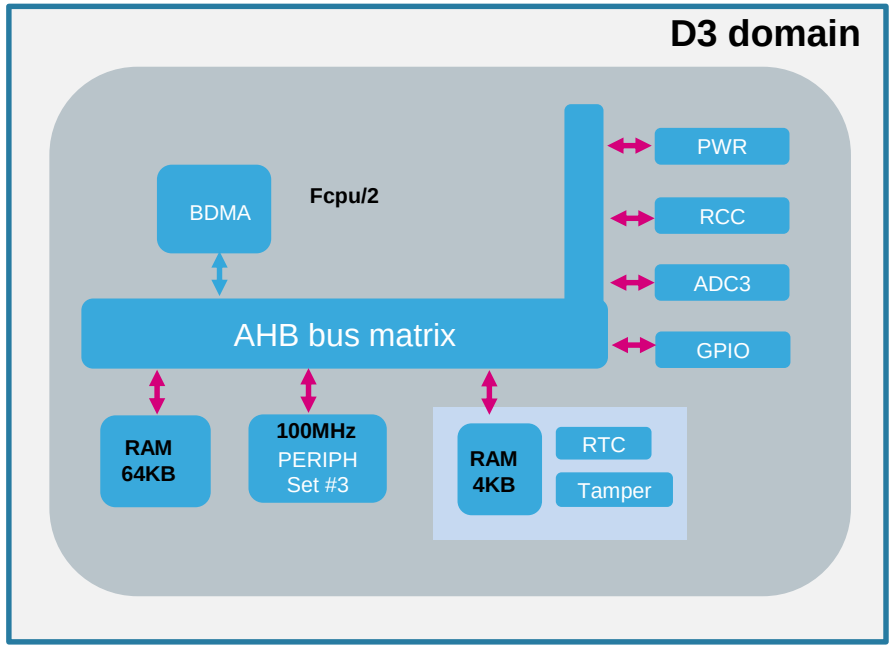


D2: 外设域

- 嵌入式存储器
  - 3个带 ECC的SRAM (128kB + 128kB + 32kB)
    - ECC on RAM is applied on each word of 32-bit (4 bytes).
- D2 SRAMs 作为Cortex-M4 内核应用**数据**和**代码**的优先存储区域。
- D2 SRAMs 可以作为外设在这个域的缓冲区，或者是本地 DMAs 数据输入/输出的缓冲区。本地 DMAs传输结束，数据可以通过MDMA 转移到D1处理域 。

D3: 低功耗域

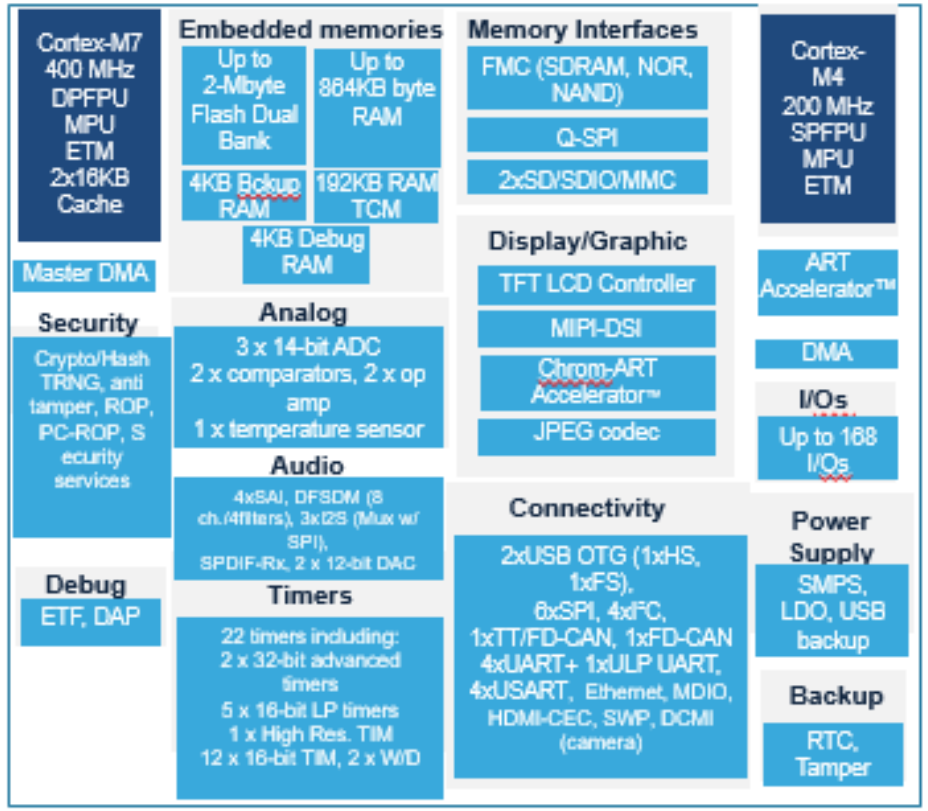
- D3提供系统管理和低功耗操作模式特性
- AHB 互联：
  - 最大频率 240MHz
  - APB 总线最大频率 120MHz
- 外设
  - BDMA
  - LPTIM, I2C, SPI, LPUART, SAI, ADC, ..
  - 系统外设: RCC, PWR, HSEM, SYSCFG, GPIOs.

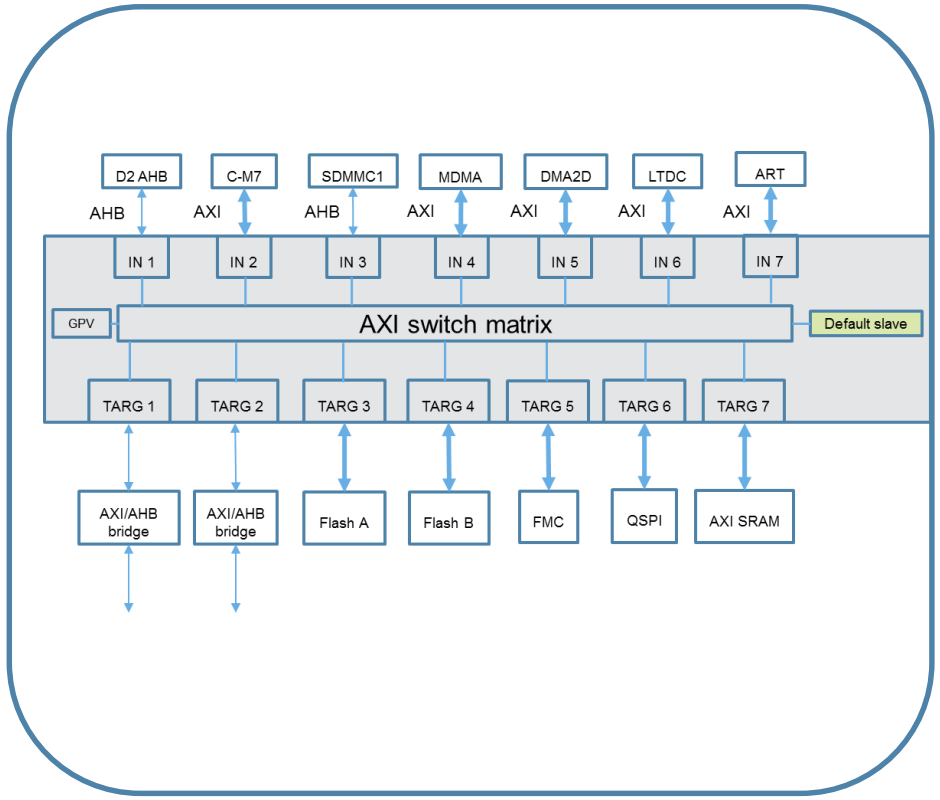


D3: 低功耗域

- 内部嵌入式存储器
  - SRAM4 with ECC (64 KB)
- D3 SRAM 可以用来保留数据，当D1和D2域进入DStandby模式

STM32H7 AXI 总线及互联



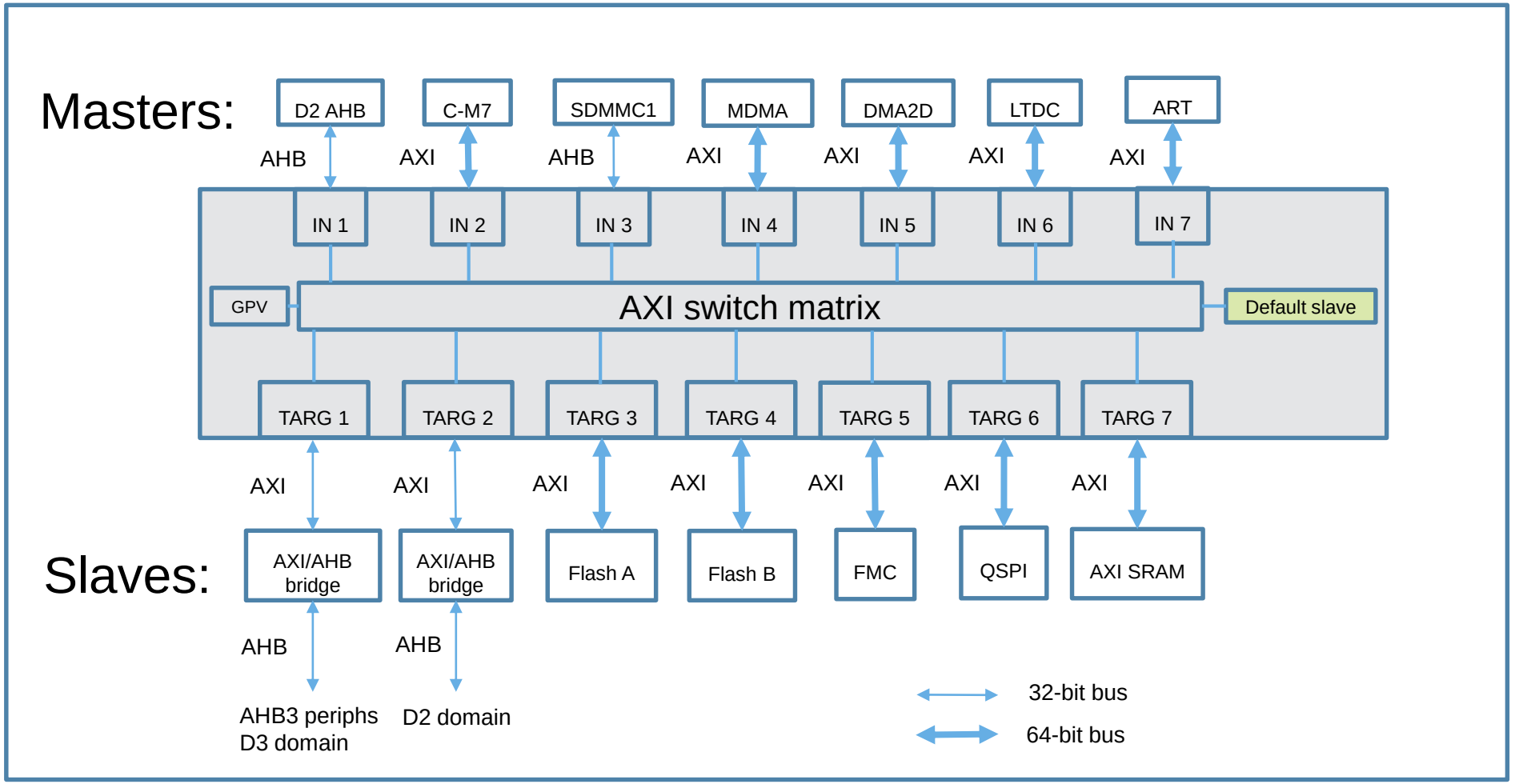


- STM32H7 AXI 互联特性
  - 64-bit AXI 总线开关矩阵
  - AHB/AXI 桥接功能
  - 可编程流量优先级管理
    - QoS 可以通过GPV 进行软件配置

应用优点

- 优化数据传输带宽
- 可配置的互连参数
- 提高实时任务的响应能力

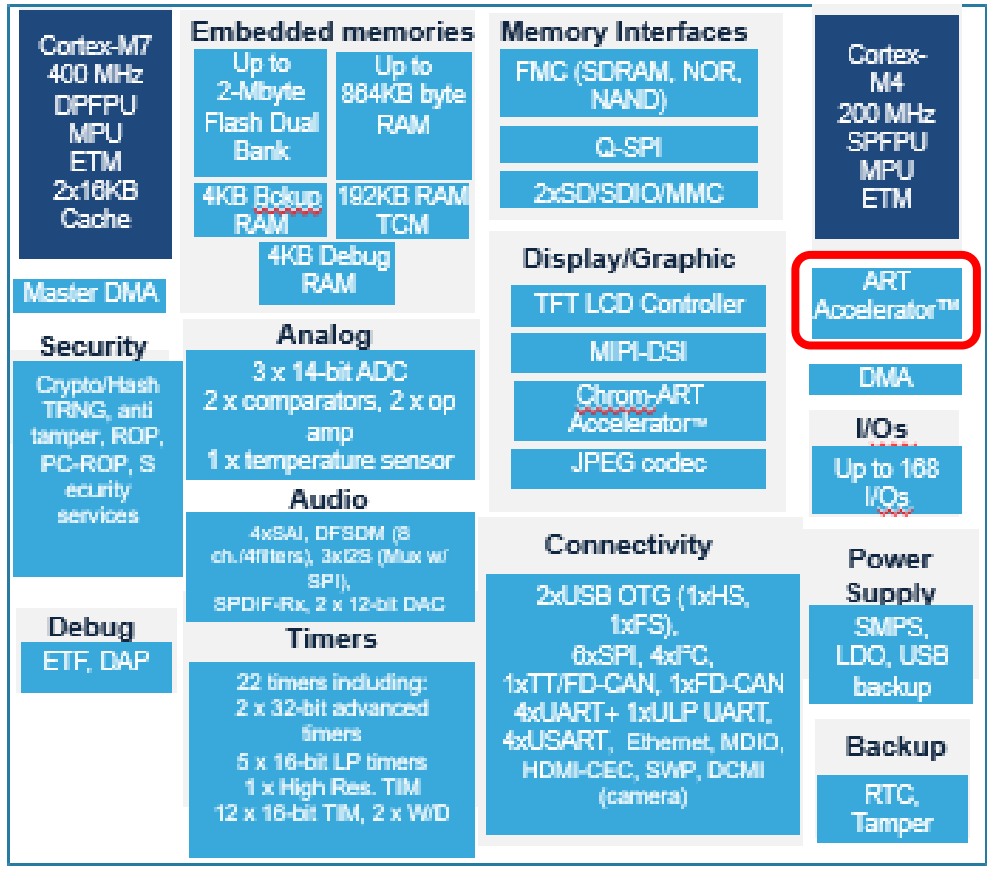
总线框图



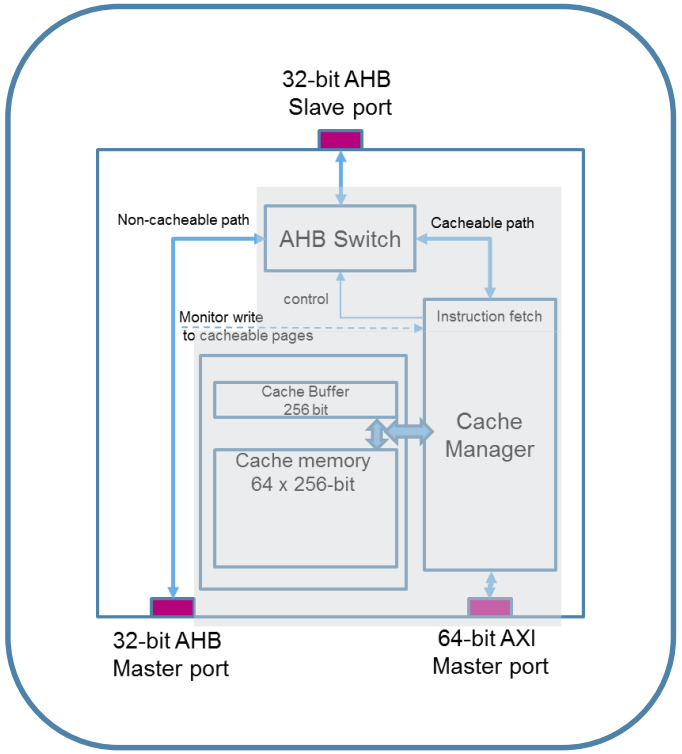
AXI总线架构关键特性

- 当两个AXI主设备同时尝试访问相同的AXI从设备，基于优先级仲裁
- AXI 读通道和写通道优先级可以独立配置
  - 配置的优先级值从0到15,
    - 值越高,优先级越高
  - 如果两个重合事务到达相同的AMIB,优先级高的事务比优先级低的事务先通过.
  - 如果两个事务有相同的QoS值,采用最近用过 (LRU)，它的优先级就更高的方案

STM32H7xx – ART



概览



- STM32H7 ART 特性
  - ART™(adaptive real-time) 加速器
  - 64 条 256 位的Cache， 加速Cortex-M4内核到D1域存储器的取指
  - 64位 AXI 主设备口选中代码到cache中

应用优点

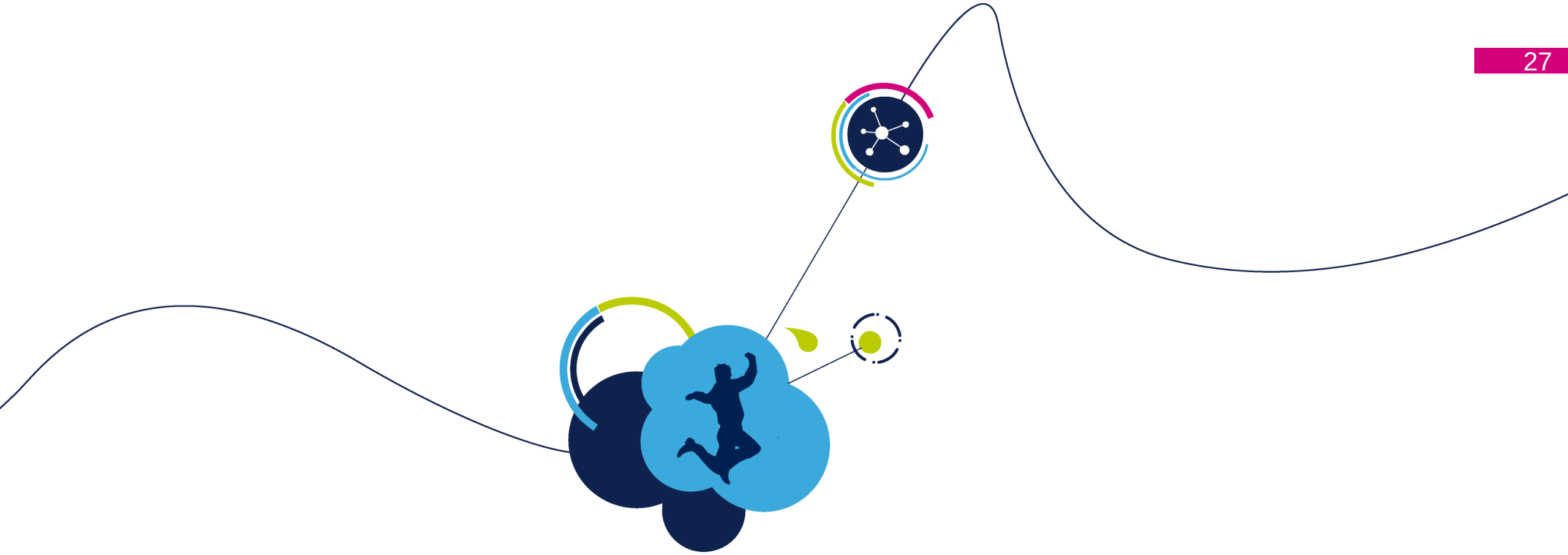
- Cortex-M4 内核可以及时的访问代码
- 避免因为Flash的等待周期造成延时等待

不同于之前的ART加速器

- ART™(adaptive real-time) 加速器模块可以加速Cortex-M4内核对以下这些存储器的取指：
  - 从D1域内部的存储器取指， 像Flash的两个bank， AXI 的SRAM
  - D1域的外部存储器，像QSPI Flash控制器 或者 FMC上外接的存储器

内容安排

- STM32H7 系统架构
  - 单核及双核STM32H7系统框图
  - AXI 总线及互联
  - ART 加速器
- **STM32H7 存储器 结构**
- STM32H7 Cache
- STM32H7 MPU
- STM32H7 启动方式
- *STM32H7-CPUs 概述*
  - *STM32H7 性能*
- STM32H7 双核优势和应用
- STM32H7 双核的同步和通信
  - IPC
  - *HSEM*
- STM32H7 电源管理
- STM32H7外设
  - DMA
  - 16位 ADC
  - LTDC MIPI 控制器
  - DLYB
  - FMC
- STM32H7目前的资源



STM32H7 存储器架构

STM32H7存储器

- STM32H7x5/ STM32H7x7 双核:
  - 嵌入式1M SRAM
    - 高达 864 Kbytes 的 System SRAM】
    - 128 Kbytes 数据 TCM RAM
    - 64 Kbytes 指令 TCM RAM
    - 4 Kbytes 的 backup SRAM
  - Flash
    - Flash控制器管理 从CPU AXI 到Flash 的访问。
      - 两个1 MB的内存块/bank，每个块/bank分为8个扇区/sector.

STM32H7架构总线

- **CPU AXIM总线**
  - 连接内部FLASH（取指或者数据存取）.
  - 连接内部SRAM（SRAM1，SRAM2)（取指或者数据存取）.
  - 连接外部存储器接口FMC （取指或者数据存取.
  - 连接外部存储器Quad SPI接口（取指或者数据存取).
- **ITCM 总线**
  - 对映射到TCM接口上的内部FLASH进行取指和数据读取。但只能对ITCM RAM进行取指操作。
- **DTCM 总线**
  - 内核通过DTCM总线对DTCM RAM进行取指或者数据存取操作
- **AHBS总线**
  - DMA通过该总线可以访问DTCM RAM，实现数据传输
- **AHBP总线**
  - 通过AHBP总线访问AHB1（包括APB）和AHB2上的外设

嵌入式SRAM

- 嵌入系统SRAM 分为5个块:
  - AXI SRAM (D1 域) 通过 D1 域 AXI 总线矩阵实现访问:
    - AXI SRAM (**512** KByte) 映射在地址0x2400 0000
    - 支持 bytes, half-words, full-words or double-words 访问
  - AHB SRAM (D2 域) 通过 D2 域 AHB 总线矩阵实现访问 :
    - AHB SRAM1 (**128** KByte) 映射在地址 0x3000 0000
    - AHB SRAM2 (**128** KByte) 映射在地址 0x3002 0000
    - AHB SRAM3 (**32** KByte) 映射在地址 0x3004 0000
    - 支持 bytes, half-words or full-words 访问
  - AHB SRAM (D3 域) 通过 D3域 AHB 总线矩阵实现访问 :
    - AHB SRAM4 (**128** KByte) 映射在地址 0x3800 0000， 并且大多系统主控制器可以访问
    - 支持 bytes, half-words or full-words 访问

嵌入 SRAM：D2 AHB SRAMs

- D2 域 AHB SRAMs 也可以重新映射到：
  - 地址 0x1000 0000 for AHB SRAM1
  - 地址 0x1002 0000 for AHB SRAM2
  - 地址 0x1004 0000 for AHB SRAM3
- 全部系统主设备都可以通过 D2 域 AHB 总线矩阵来访问 D2 域 AHB SRAMs

嵌入 SRAM : TCM RAM

- TCM SRAMs 是给 Cortex®-M7 专用的:
  - DTCM-RAM on TCM 接口 映射在地址 0x2000 0000， 并且只能通过下面两种方式访问:
    - Cortex®-M7,
    - MDMA 通过 Cortex®-M7 CPU AHBS 从总线.
  - ITCM-RAM on TCM 接口 映射在地址 0x0000 0000， 并且只能通过 Cortex®-M7访问.
- 这些内存可以运行在最大的CPU 时钟频率 (480MHz) @ **0**等待

内容安排

- STM32H7 系统架构
  - 单核及双核STM32H7系统框图
  - AXI 总线及互联
  - ART 加速器
- STM32H7 存储器 结构
- **STM32H7 Cache**
- STM32H7 MPU
- STM32H7 启动方式
- *STM32H7-CPUs 概述*
  - *STM32H7 性能*
- STM32H7 双核优势和应用
- STM32H7 双核的同步和通信
  - IPC
  - *HSEM*
- STM32H7 电源管理
- STM32H7外设
  - DMA
  - 16位 ADC
  - LTDC MIPI 控制器
  - DLYB
  - FMC
- STM32H7目前的资源

高速缓存(Cache)介绍

- 什么是高速缓存?
  - 高速存储器块，包含地址信息（通常称作TAG）和相关联的数据。
  - 目的是提高对存储器的平均访问速度
- 高速缓存的应用基于下面两个程序的局部性：
  - 空间局部性：如果一个存储器的位置被访问，那么将来它附近的位置也会被访问。比如顺序执行代码，或者使用一个数据结构
  - 时间局部性：被访问过一次的存储器位置，接下来会被多次引用。比如：循环
- 缓存行（cache line)
  - 逻辑上的一组存储器位置。内存交换数据的最小粒度
- 缓存命中（cache hit)，要访问的数据/指令已经存在缓存里
- 缓存缺失（cache miss)，要访问的数据/指令不在缓存里

高速缓存介绍

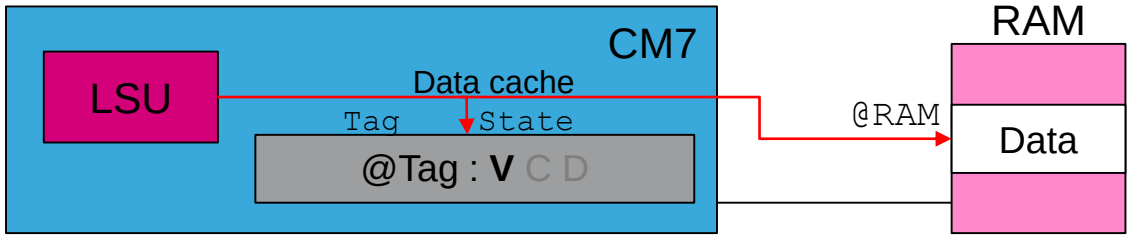
- 处理器需要访问某个可缓存的存储器位置时，先检查缓存内是否已经存在该位置的内容。
- 如果缓存命中，则直接从缓存读出。如果缓存缺失，则从存储器里读出，同时放入缓存。
- 缓存分配（Cache Allocation)
  - 缓存缺失（Cache miss）时，在缓存中发现一个位置，并把新的缓存数据存到这个位置。
  - 读分配（Read-allocate）：在进行读操作发生缓存缺失时，进行缓存分配。所有可缓存的存储器都是读分配。
  - 写分配（Write-allocate)：在进行写操作发生缓存缺失时，进行缓存分配。
- 高速缓存的应用带来一致性问题
  - 程序员不能控制对存储器的访问时机
  - 同一个数据被保存在多个物理位置

Cache术语

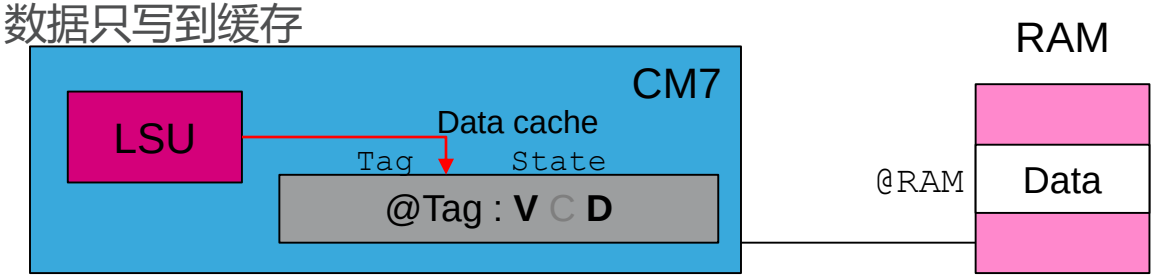
- 驱逐（Eviction）：
  - 从缓存中移除一个缓存行（cache line)，为新的数据腾位置的过程Write of an entire cache line on the bus
  - 发生于当一个“dirty”的缓存行被新的缓存行替代时
  - Dirty是标记那些需要写回到存储器中的缓存数据。
- 回写(Write-back):
  - 写数据时，只更新缓存，然后将缓存行标记为“dirty”
  - 当这个缓存行被新的缓存行替换时，再将数据写到存储器中
- 透写(Write through):
  - 写数据时同时更新缓存和二级存储
  - 缓存行不被标记为“dirty”

缓存策略

- 内部L1缓存策略描述:
  1. 透写 (读或者写分配)
     - 数据同时写到缓存和下一级存储器中



  2. 回写（读或者写分配）
     - 数据只写到缓存



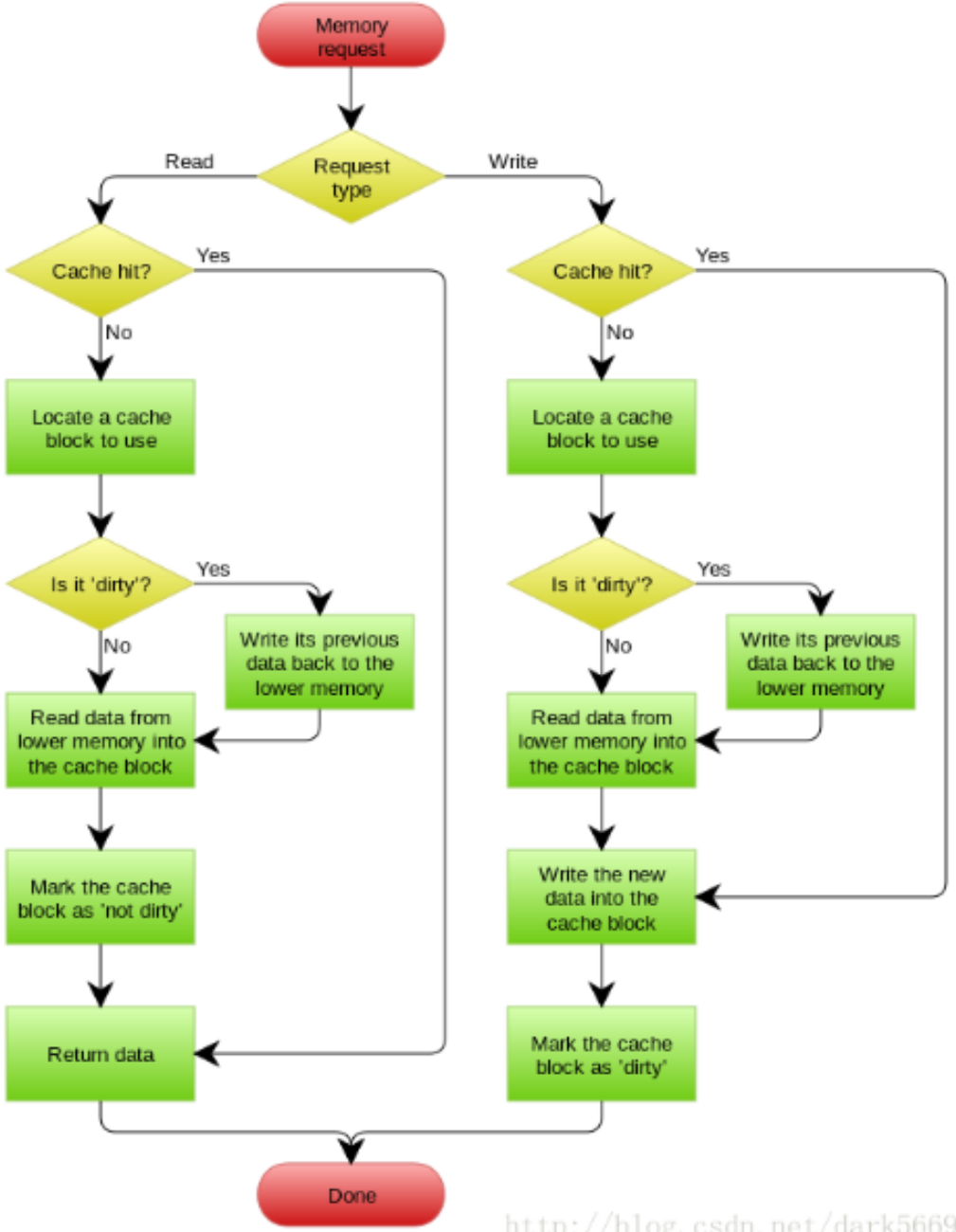
Cortex-M7高速缓存

- 采用哈佛结构，实现最优性能
- 指令缓存和数据缓存都是可选的，独立配置大小
- 只在AXIM接口有缓存 (TCMs和AHBP没有)
- 两路相联指令高速缓存, 四路相联数据高速缓存
- 对ARMv7E-M系统架构的要求
  - 增加Cache维护操作，增加新的相关寄存器.
  - 使用Cache会对系统软件有影响

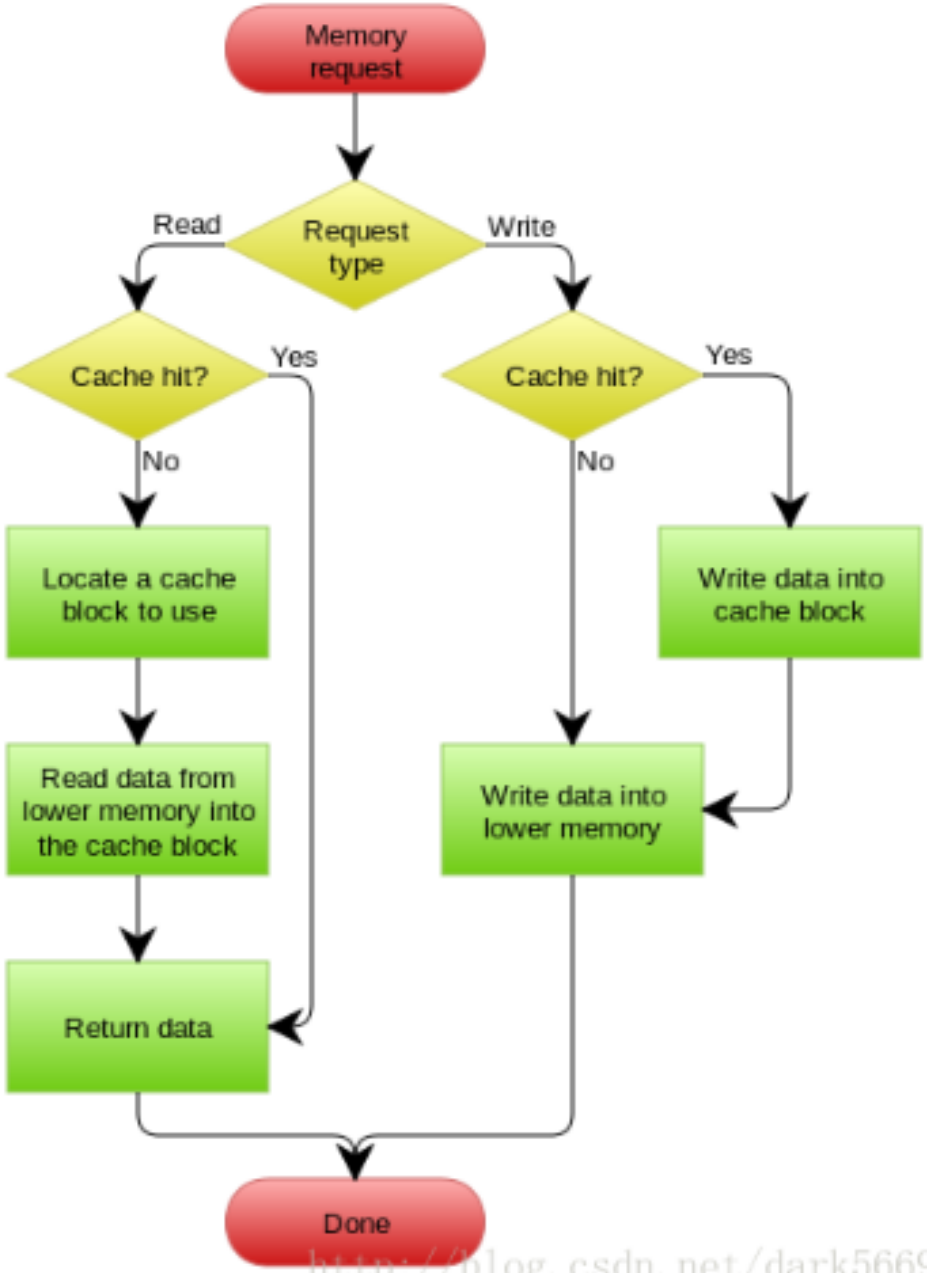
Cortex-M7高速缓存

- 全面支持下列缓存属性
  - 透写(Write Through), 没有写分配(write allocate)
  - WBRA:回写(Write-Back),没有写分配(WBRA)
  - WBWA:回写(Write-Back), 写分配(write allocate)
- 回写方式有利于优化性能
- 没有对“一致性” 的硬件支持
  - 共享内存不支持缓存，以保持和二级存储的一致性.

WB（write back）
+WA（write allocation）



WT （write through)
+WN(write no allocation)



Cache的一致性

- 没有对一致性的硬件支持
- 两种可选方案
  - 所有的共享存储器都定义为共享属性
    - 这些区域将默认不被缓存到D-Cache
    - 所有的操作都直接针对2级存储器（内部Flash，外部存储器），性能降低
    - 因为缓存对这些区域是透明的，写软件更容易
  - 通过软件进行cache的维护
    - Cortex-M7 的写操作要是全局可见的
      - 使用透写属性（通过MPU设置）
      - 使用SIWT@CACR (Shared = Write Through)
      - 通过指令清D-cache，然后所有更新位置禁止D-Cache操作
    - 其他主设备的写操作要对Cortex-M7可见
      - 比如作废Cortex-M7 Dache中数据.

缓存和存储器属性

- 存储器的属性由MPU进行设定
- 这些属性同样对AXIM/Cache接口上的存储器系统有影响
- 属性
  - 共享
    - 仅用于当多个master共享同一块存储空间时（一致性保证)
    - 默认**对数据的存取不在D-Cache里做缓存**
    - **对I-cache没有影响**– 共享的区域还是会被缓存
  - 分配策略
    - 写分配- 整体性能最佳
    - 读分配 – 具体例子（e.g memset()）
    - 透写 – 性能上比回写低，但对安全关键代码很有用
    - 对 I-Cache而言, 所有的分配策略都被当做读分配（没有写指令的操作)
  - 存储器类型
    - Normal – 可以被缓存（peripheral on cacheability/shared attributes)
    - DEV/SO – 仅用于Devices- 不能被缓存.

CMSIS cache 函数

Table 1. CMSIS cache functions

CMSIS function	Description
`void SCB_EnableICache (void)`	Invalidate and then enable the instruction cache
`void SCB_DisableICache (void)`	Disable the instruction cache and invalidate its contents
`void SCB_InvalidateICache (void)`	Invalidate the instruction cache
`void SCB_EnableDCache (void)`	Invalidate and then enable the data cache
`void SCB_DisableDCache (void)`	Disable the data cache and then clean and invalidate its contents
`void SCB_InvalidateDCache (void)`	Invalidate the data cache
`void SCB_CleanDCache (void)`	Clean the data cache
`void SCB_CleanInvalidateDCache (void)`	Clean and invalidate the data cache

AN 4839 Level 1 cache on STM32F7 Series

Cache相关函数及作用

- 这里以core_cm7.h里对cache封装的函数为例
  (C:\Program Files (x86)\IAR Systems\Embedded Workbench 7.2\arm\CMSIS\Include)

SCB_CleanDCache

Clean所有的cache-line，即将dirty的cache-line全部写到cache line对应的真实的物理地址中 所谓的drity属性，即写操作时，更新了相应的cache-line，但是没有更新到真实的物理地址， 而这个clean的动作，就是将cache中的内容更新到真实的物理地址中

SCB_CleanDCache_by_Addr

根据地址信息clean其对应的cache-line

Cache相关函数及作用

SCB_InvalidateDCache

无效D-cache，D-cache被invalidate之后，当有Host(如core，DMA等)读取数据时，会忽略相应的cache-line中的内容（因为被validate了），从真实的物理地址中去获取相应的数据

SCB_InvalidateDCache_by_Addr

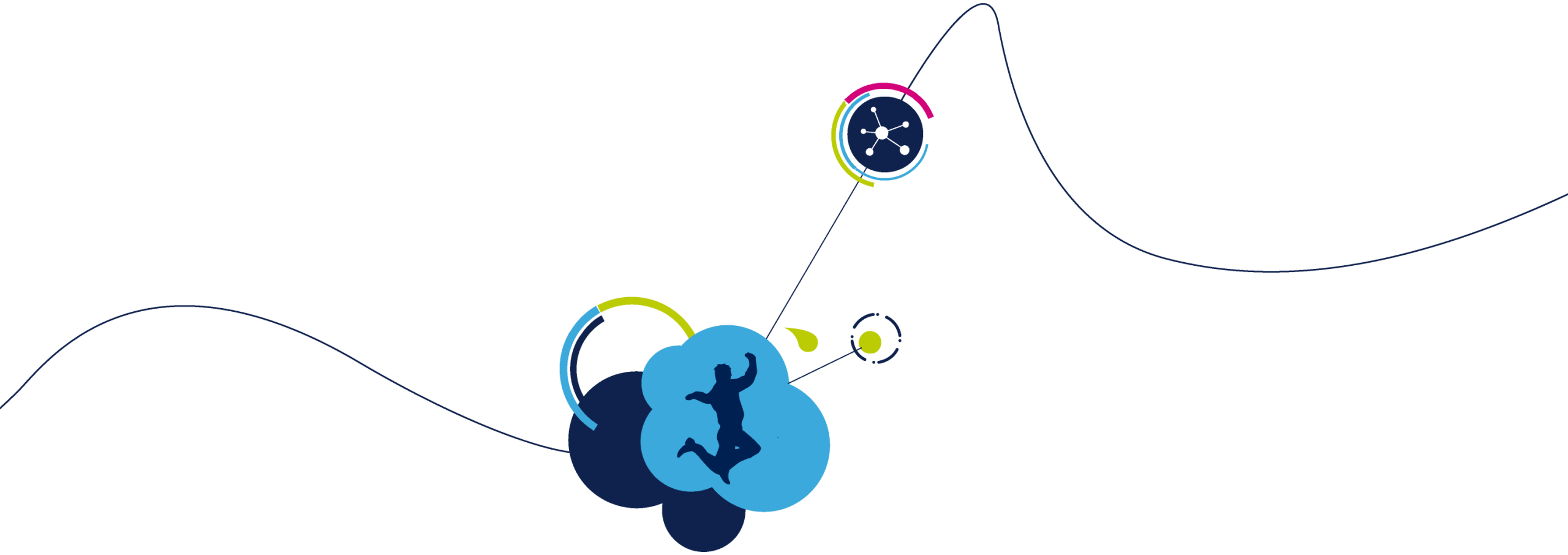
根据地址信息无效其对应的cache-line

存储器默认映射和属性

地址范围	名称	存储器类型	Cache	描述
0x0000 0000 0x1FFF FFFF	Code	Normal	WT	Typically ROM or flash memory. Memory required from address 0x0 to support the vector table for system boot code on reset
0x2000 0000 0x3FFF FFFF	SRAM	Normal	WBWA	SRAM region typically used for on-chip RAM
0x4000 0000 0x5FFF FFFF	Peripheral	Device	-	On chip peripheral address space
0x6000 0000 0x7FFF FFFF	RAM	Normal	WBWA	Memory with write-back, write allocate cache attribute for L2/L3 cache support
0x8000 0000 0x9FFF FFFF	RAM	Normal	WT	Memory with write-through cache attribute
0xA000 0000 0xBFFF FFFF	Device	Device, Shareable	-	Shared device space
0xC000 0000 0xDFFF FFFF	Device	Device, non-shareable	-	Non-shared device space
0xE000 0000 0xE00F FFFF	PPB	Strongly-Ordered	-	1MB region reserved as the PPB. This supports key resources, including the System Control Space and debug features.
0xE010 0000 0xFFFF FFFF	Vendor_SYS	Device	-	Vendor system region

初始化和使能L1 Caches

- 上电（power-on)复位时，在使能前，I-cache和D-cache必须全部被作废（invalidate)
  - Cache行的标签(Tag)中的valid位
  - 如果不这样做，可能会引发程序不可预测的行为
  - 注意：软复位时，如果复位前RAM里的值是确定可靠的，可以不用做这一步
- I-Cache:
  - 通过 ICIALLU 寄存器来作废整个I-Cache
- D-Cache:
  - 建议遍历整个D-Cache并作废
- 为了保证数据的一致性，必须在除能D-cache之前对其进行清理
  - 仅在使用**回写**策略时需要
  - 如果不这么做就可能会丢失数据

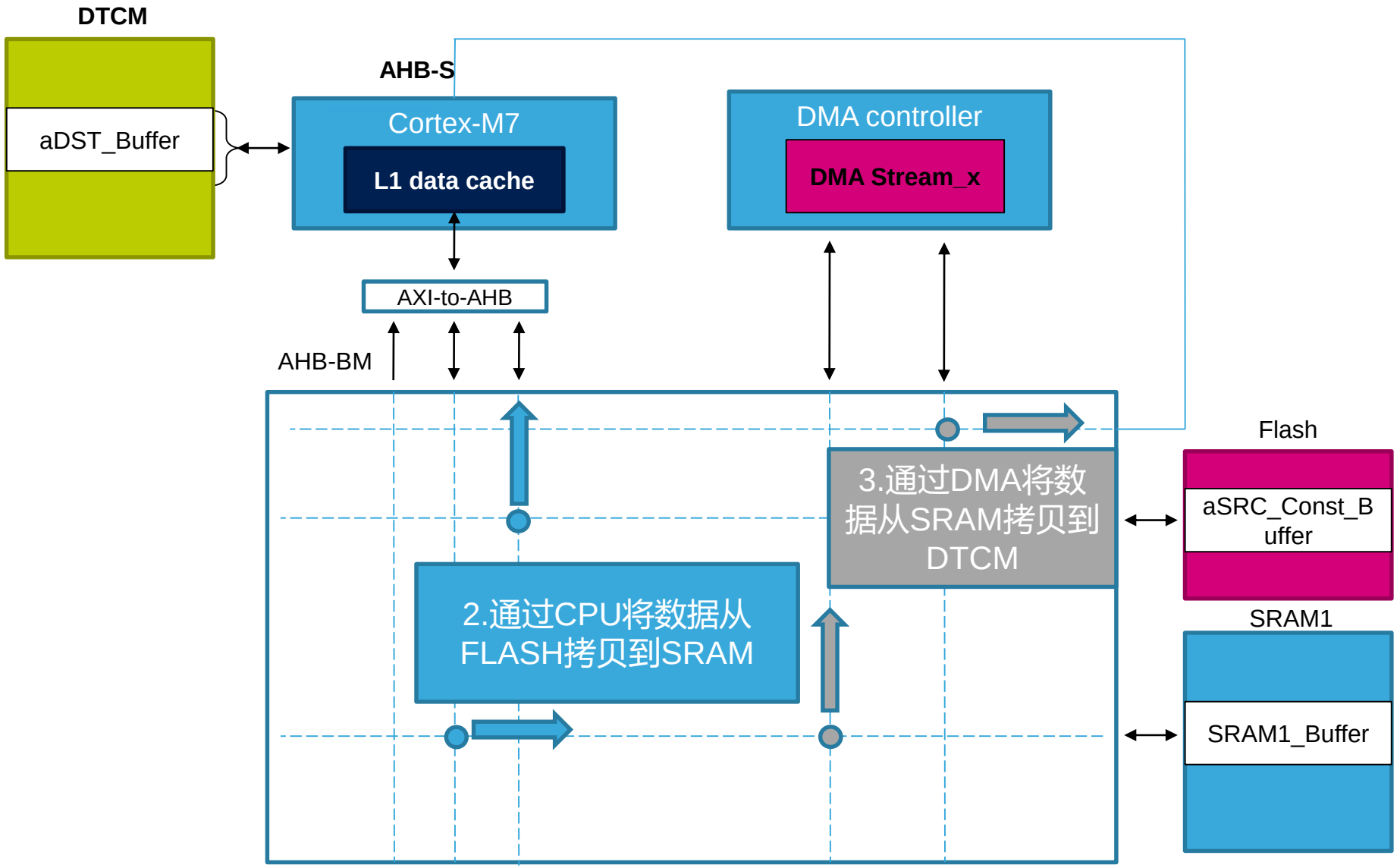


Cache一致性举例

程序描述

- 1.首先将地址0x20010000（SRAM1）处开始的128字节初始化为0x55.
- 2.将Flash中的128字节的常量数组aSRC_Const_Buffer拷贝到SRAM1地址0x20010000 (pBuffer).
- 3.配置并使能DMA，通过DMA将数据从SRAM1的地址0x20010000处拷贝到DTCM RAM中的数组 aDST_Buffer中.
  - DMA传输完成后，LED1 亮.
- 4.将Flash中的数组aSRC_Const_Buffer与DMA读出的数组aDST_Buffer进行比较
  - 如果不一样则LED2 亮.
  - 一样则LED3 亮

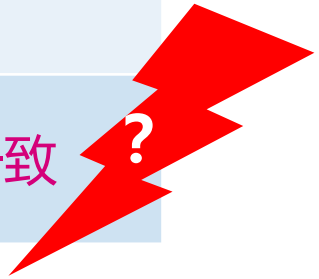
数据传输路径



Cortex-M7 数据一致性–Cache关 vs Cache开

实验1

Cache 关	Cache 开
默认关	/* Enable D-Cache */ SCB_EnableDCache();
LED1 &LED3 亮	LED1 &LED2 亮
Flash与DTCM的数据一致	Flash与DTCM的数据不一致



CMSIS定义了开关Cache的函数：
SCB_EnableDCache（），SCB_EnableICache （）

Cortex-M7 数据一致性——执行清Cache操作

实验2

- 使能D-cache

使能D-Cache

```
/* Fill 128 bytes with 0x55 pattern */
memset((uint8_t*)SRAM1_ADDRESS_START, 0x55, sizeof(aSRC_Const_Buffer));

/* Step-1 : Enable Data cache (uncomment following line to enable Data Cache */
/* Enable Data cache */
 SCB_EnableDCache();
```

- 在对可缓存区域进行写操作后，执行清Cache的操作。

2.通过CPU将数据从FLASH拷贝到SRAM

```
for (counter = 0; counter < (sizeof(aSRC_Const_Buffer)/4); counter++)
{
  *pBuffer++ = aSRC_Const_Buffer[counter];
}
```

执行清Cache操作

```
/* Step-2 : Perform cache clean operation by software (uncomment following line to
 * ensure coherency by software) */

/* Clean Data Cache */
SCB_CleanDCache();
```

- 所有的dirty line都被回写到SRAM1， 保证了cache和二级存储之间数据的一致性
- LED1 &LED3灯亮
- Flash与DTCM中的数据一致

Cortex-M7 数据一致性——透写属性

实验3

- 使能MPU，修改SRAM1的MPU属性从回写(write-back)改为透写（write-through）。

1. 在打开D-Cache之前把MPU配置好

2. MPU要配置为Write-Through透写模式

```
/* Fill 128 bytes with 0x55 pattern */
memset((uint8_t*)SRAM1_ADDRESS_START, 0x55, sizeof(aSRC_Const_Buffer));


/* Step-3 : Enable MPU and change SRAM region attribute (uncomment following line to
* set write-through policy on SRAM) */
 MPU_Config();

/* Step-1 : Enable Data cache (uncomment following line to enable Data Cache */
/* Enable Data cache */
 SCB_EnableDCache();

/* Step-4 : Force all CPU accesses to be write-through (uncomment following line to
* set all CPU accesses to cachable region as write-through) */

 /*__FORCE_WRITE_THROUGH(); */

for (counter = 0; counter < (sizeof(aSRC_Const_Buffer)/4); counter++)
{
      *pBuffer++ = aSRC_Const_Buffer[counter];
}
```

Cortex-M7 数据一致性——SIWT@CACR

实验4

- 使能MPU，SRAM1的MPU属性设置成回写

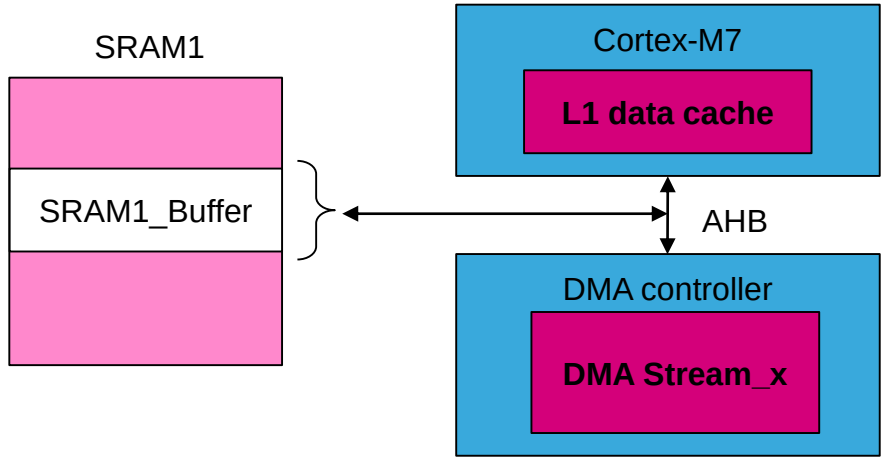
1. 在打开D-Cache之前把设置SIWT位

2.不管 MPU如何配置，都按照Write-Through透写模式对待

```
/* Fill 128 bytes with 0x55 pattern */
memset((uint8_t*)SRAM1_ADDRESS_START, 0x55, sizeof(aSRC_Const_Buffer));


/* Step-3 : Enable MPU and change SRAM region attribute */
 MPU_Config();


/* Step-4 : Force all CPU accesses to be write-through (uncomment following line to
* set all CPU accesses to cachable region as write-through) */

 __FORCE_WRITE_THROUGH();

/* Step-1 : Enable Data cache (uncomment following line to enable Data Cache */
/* Enable Data cache */
 SCB_EnableDCache();

for (counter = 0; counter < (sizeof(aSRC_Const_Buffer)/4); counter++)
{
      *pBuffer++ = aSRC_Const_Buffer[counter];
}
```

Cortex-M7与DMA之间的数据一致性

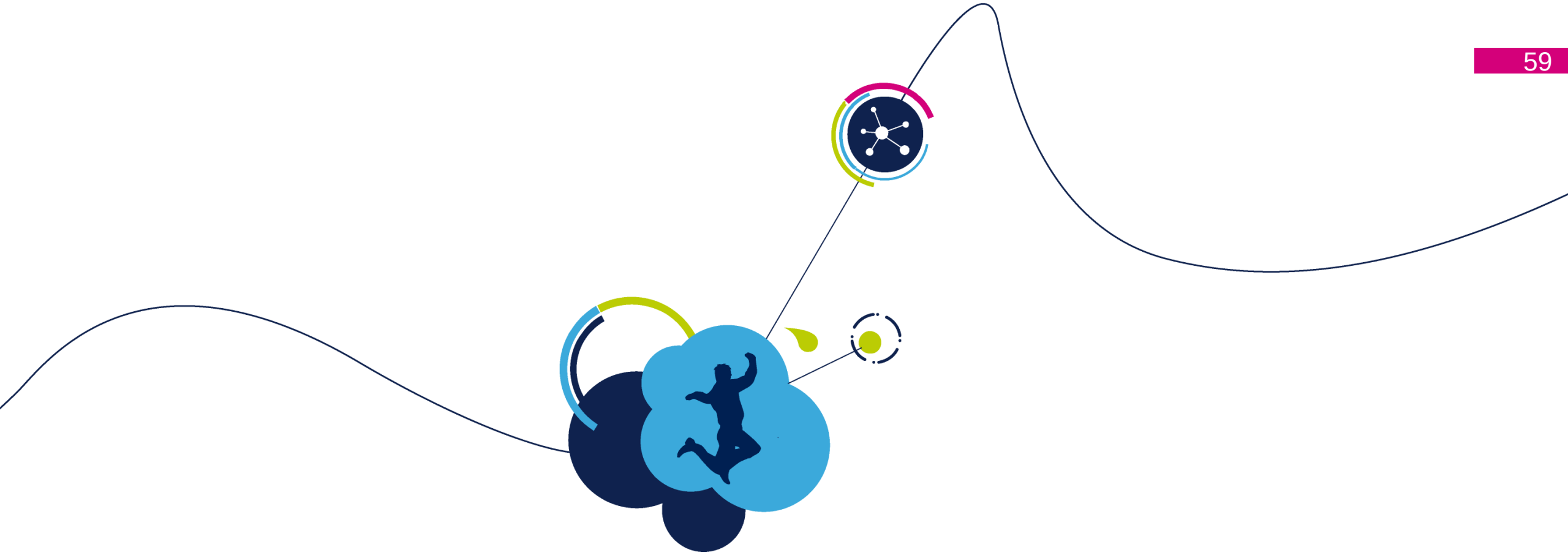
- 前面的例子中，Cortex-M7内核与DMA共享SRAM1中的一个数据buffer:



- 内核与DMA之间的数据一致性通过下面3种方法得到了保证:
  1. 通过置位SIWT@CACR 禁止对SRAM1进行cache
  2. 通过软件关闭D-Cache功能，或者及时进行清Cache的操作，将数据回写到存储器中
  3. 打开Cache功能，当把SRAM1设置为透写属性
  - 在这个例子中DMA的目标buffer不能被缓存（在DTCM中），所以没有数据一致性的问题。但数据不一致的问题同样有可能发生在DMA的目标buffer上，如果它是一个可被缓存的共享buffer的话.

内容安排

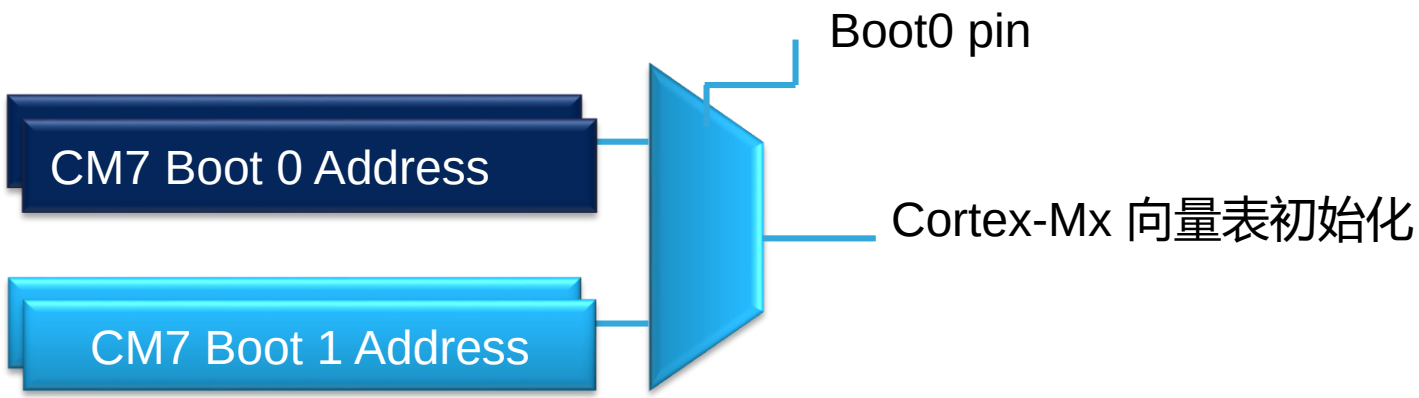
- STM32H7 系统架构
  - 单核及双核STM32H7系统框图
  - AXI 总线及互联
  - ART 加速器
- STM32H7 存储器 结构
- STM32H7 Cache
- STM32H7 MPU
- **STM32H7 启动方式**
- *STM32H7-CPUs 概述*
  - *STM32H7 性能*
- STM32H7 双核优势和应用
- STM32H7 双核的同步和通信
  - IPC
  - *HSEM*
- STM32H7 电源管理
- STM32H7外设
  - DMA
  - 16位 ADC
  - LTDC MIPI 控制器
  - DLYB
  - FMC
- STM32H7目前的资源



STM32H7 启动方式

Boot 方式

- boot 地址通过 option byte确定。 默认值如下:
  - 当Boot0=0时，CM7 从Flash memory 0x0800 0000 地址启动
  - 当Boot0=0时，CM4 从Flash memory 0x0810 0000 地址启动
  - 当Boot0=1时， 从System memory 或者 SRAM1 启动
- 复位后，在第4个系统时间的上升沿锁定引导 pin 的值



Boot 模式

- 如果程序引导内存地址超出内存映射区域，或者是保留区域，默认的boot地址是:
  - BOOT_CM7_ADD0: FLASH at 0x0800 0000
  - BOOT_CM4_ADD0: FLASH at 0x0810 0000
  - BOOT_ADD1_CM7: System Memory at 0x1FF0 0000
  - BOOT_ADD1 _CM4: SRAM1 at 0x1000 0000

- 当Flash level 2 保护使能时:
  - 只有从Flash引导是有效的
  - 如果程序boot的内存地址超出存储器范围或者 RAM地址， 默认的boot地址是:
    - Flash at address 0x0800 0000 for Cortex®-M7
    - Flash at address 0x0810 0000 for Cortex®-M4.

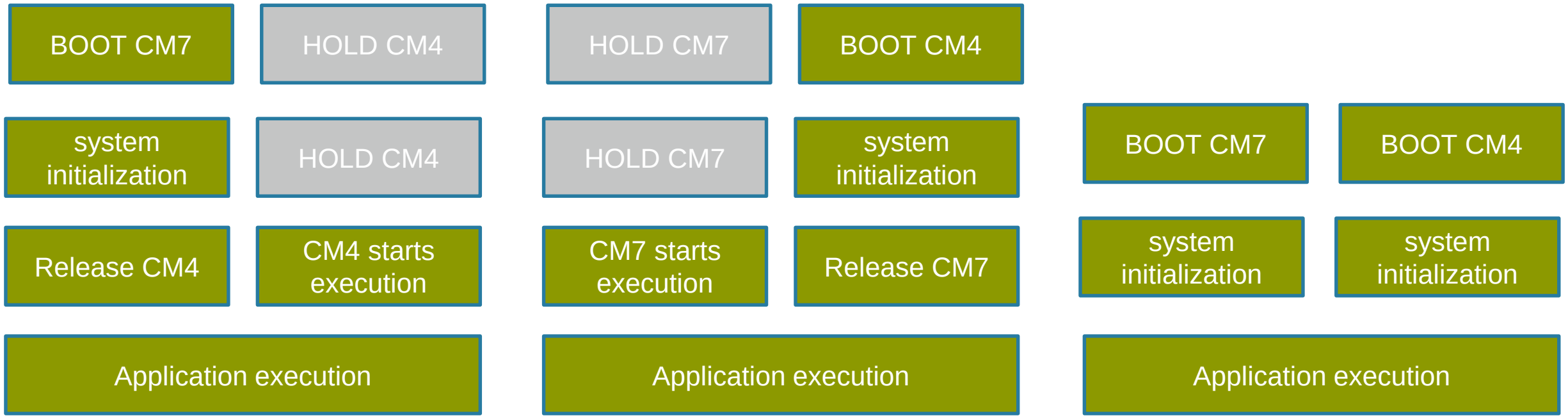
Boot 模式

- 在双核系列STM32H7x5 和 STM32H7x7, 两个内核可以单独 或者 同时启动 根据如下额option bytes配置 :

BCM7	BCM4	Boot order
0	0	Cortex®-M7 is booting and Cortex®-M4 clock is gated
0	1	Cortex®-M7 clock is gated and Cortex®-M4 is booting
1	0	Cortex®-M7 is booting and Cortex®-M4 clock is gated
1	1	Both Cortex®-M7 and Cortex®-M4 are booting

- 使能的CPU 被定义为主CPU, 它负责系统的初始化。
- 上电时可以做安全启动和初始化。

Boot 模式 : boot 顺序



BCM7=1 & BCM4=0

BCM7=0 & BCM4=1

BCM7=BCM4

life.augmented

内容安排

- STM32H7 系统架构
  - 单核及双核STM32H7系统框图
  - AXI 总线及互联
  - ART 加速器
- STM32H7 存储器 结构
- STM32H7 Cache
- STM32H7 MPU
- STM32H7 启动方式
- ***STM32H7-CPUs 概述***
  - ***STM32H7 性能***
- STM32H7 双核优势和应用
- STM32H7 双核的同步和通信
  - IPC
  - *HSEM*
- STM32H7 电源管理
- STM32H7外设
  - DMA
  - 16位 ADC
  - LTDC MIPI 控制器
  - DLYB
  - FMC
- STM32H7目前的资源

STM32H7 高性能实验

life.augmented

内容安排

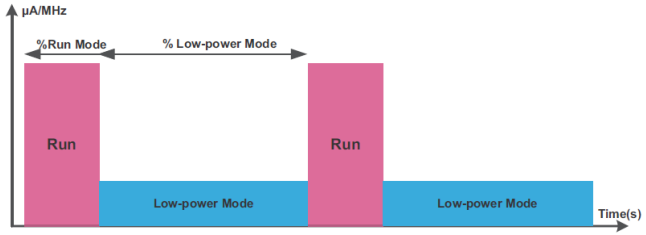
- STM32H7 系统架构
  - 单核及双核STM32H7系统框图
  - AXI 总线及互联
  - ART 加速器
- STM32H7 存储器 结构
- STM32H7 Cache
- STM32H7 MPU
- STM32H7 启动方式
- *STM32H7-CPUs 概述*
  - *STM32H7 性能*
- **STM32H7 双核优势和应用**
- STM32H7 双核的同步和通信
  - IPC
  - *HSEM*
- STM32H7 电源管理
- STM32H7外设
  - DMA
  - 16位 ADC
  - LTDC MIPI 控制器
  - DLYB
  - FMC
- STM32H7目前的资源

STM32H7 双核的优势和应用

life.augmented

为什么双核?

- 增强系统性能
  - 2x 处理器内核并行工作
  - 降低运行时间和平均功耗



- 提高系统的效率
  - CPUs之间工作的平衡
    - CPU1: GUI
    - CPU2: Connectivity

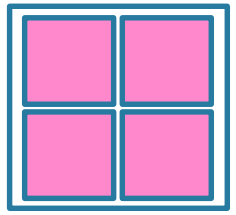
- 节省开发时间
  - 降低不同部门之间的依赖性

- 降低系统成本
  - 降低 BOM 成本，得益于 STM32H7可以完成更多task
    - 拓展了连接性
    - 加强了用户界面

多核处理器

Homogeneous
同构处理器

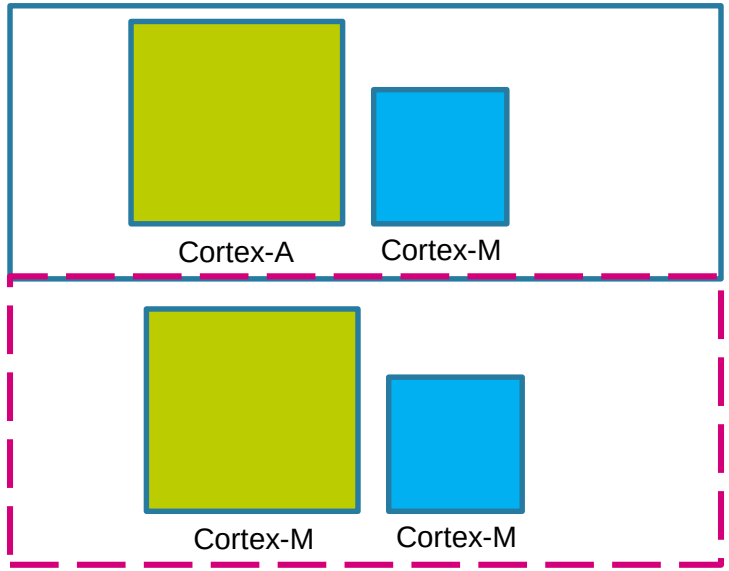
- 相同 ISA,
- 相同 micro-architecture
- 形同的 memory



多核

Heterogeneous
异构处理器

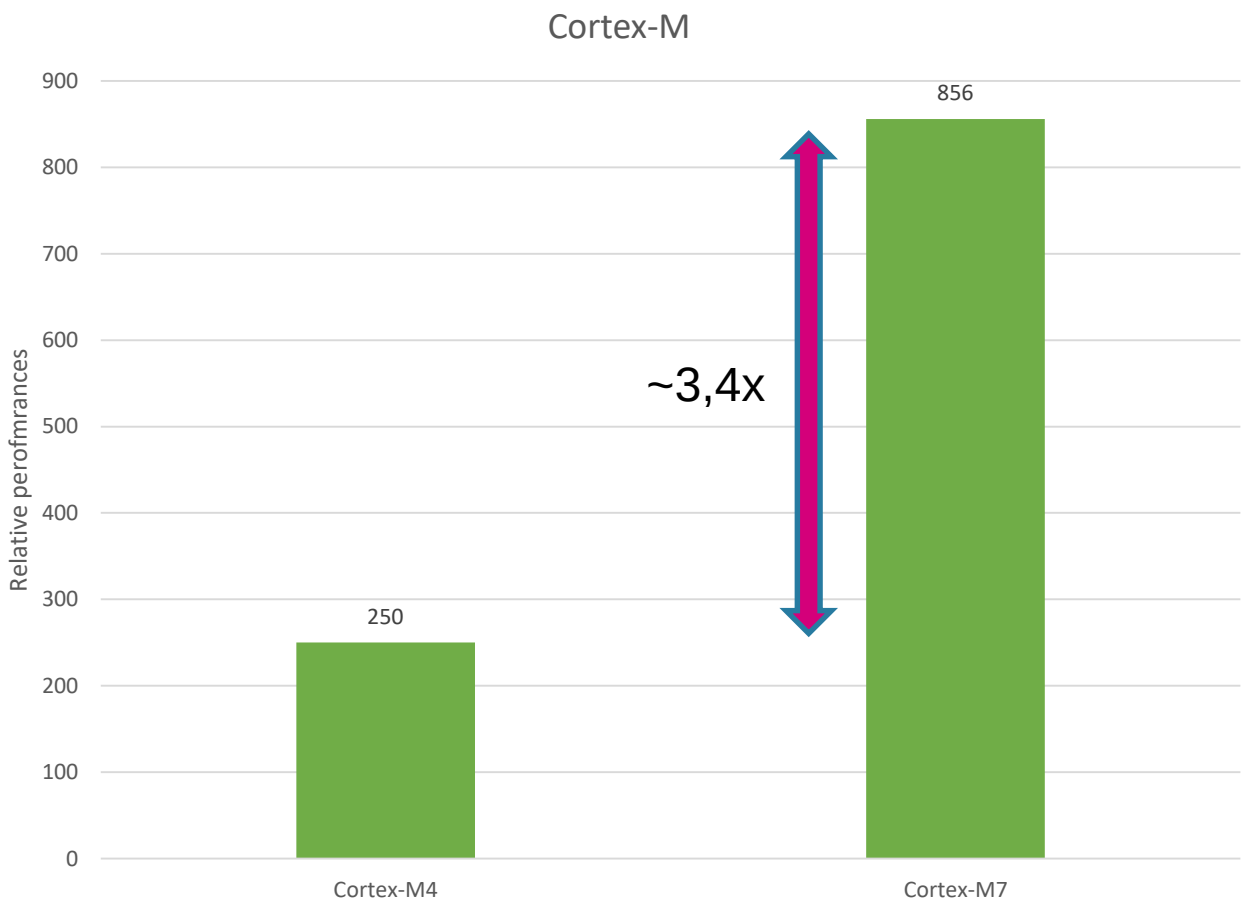
- 性能上的非对称
- 功能上的非对称



性能的非对称

Cortex-M7 vs. Cortex-M4

- 两个内核有不同的性能
  - Cortex-M7: 2.14 DMIPS/MHz
  - Cortex-M4: 1.25 DMIPS/MHz
- 两个内核频率上的不同 STM32H7x7/x5
  - Cortex-M7: up to 480 MHz
  - Cortex-M4: up to 240 MHz

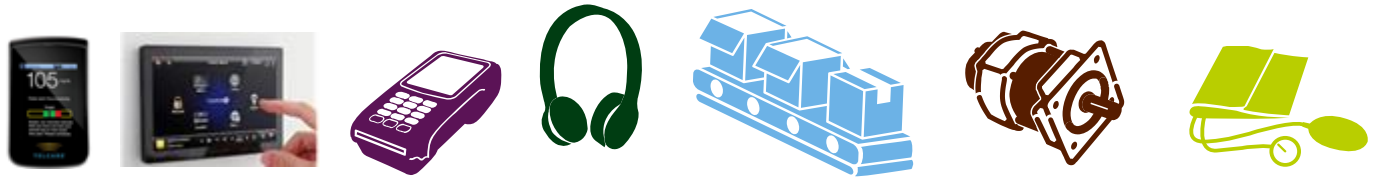


Cortex-M7+Cortex-M4 优势

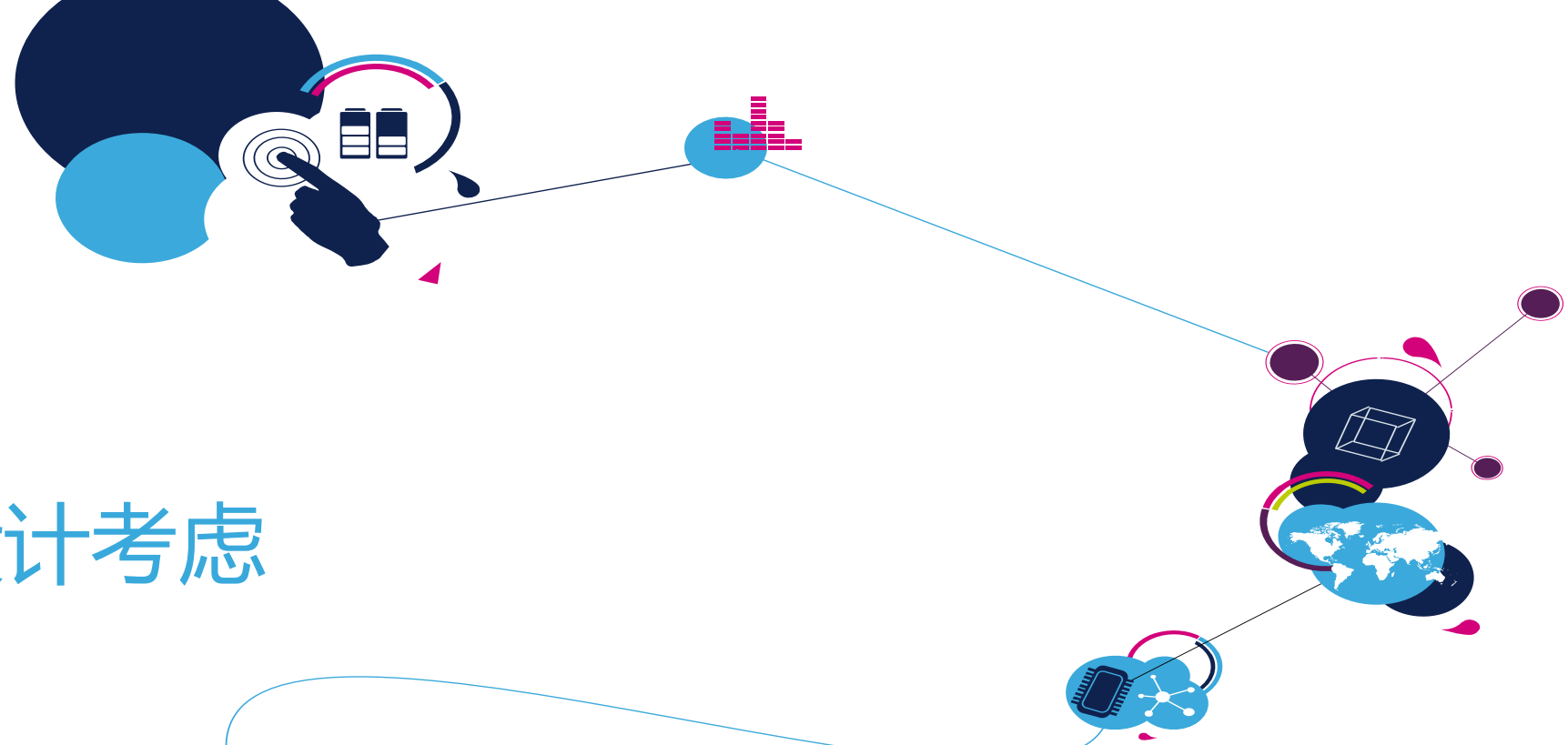
处理能力倍增

- 最大化性能
  - 2x SIMD 能力
  - 2x single cycle MAC 能力
  - 2x FPU 能力
- 实时处理能力增强
  - Tightly coupled memories
  - AXI-M interface 有 Cache memory
- 高效的负载均衡能力
  - 高效的外设管理能力
  - 更高的处理带宽
  - 增强了系统的响应能力

工作性能增强的例子



Core	消费类	工业	医疗
Cortex-M7	高级的用户界面, 高性能		
Cortex-M4	确定的传感器控制	实时控制和监控	实时性



双核应用的设计考虑

life.augmented

双核的系统设计考虑

- 如何给任务分配处理器和域?
- 如何分配不同的存储器映射?
- 如何分发中断?
- 如何处理不同内核之间的通信和同步?

双核应用设计

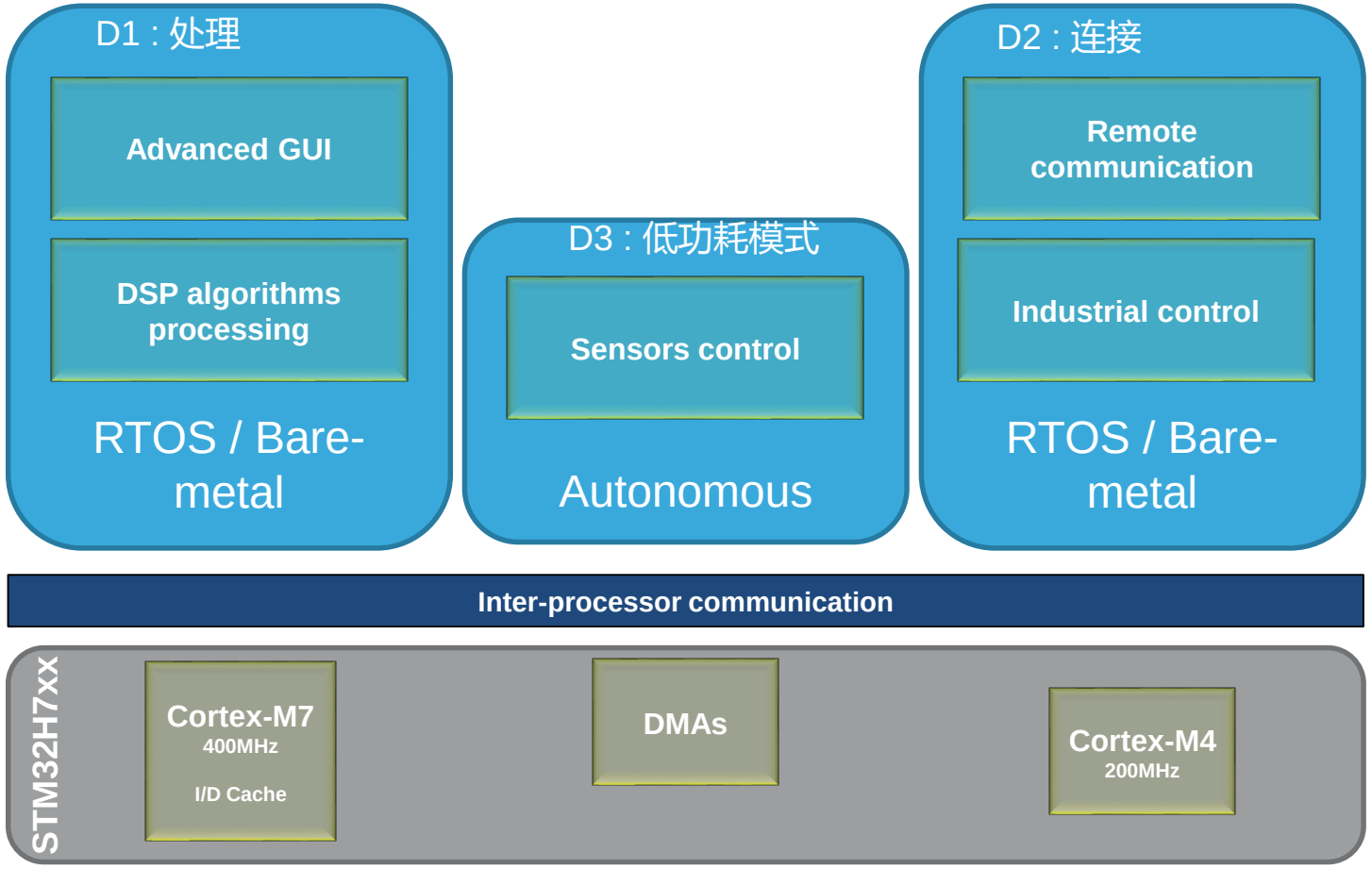
- 第1步：分析应用中的瓶颈在哪里
  分析整个应用的数据流，找出最需要优化的部分

第2步：设定需要满足的关键应用需求

- 性能
- 功耗

- 第3步: 区分出合适的并行任务
  - 完全独立的任务
  - 需要合作的任务: 可以划分成不同部分

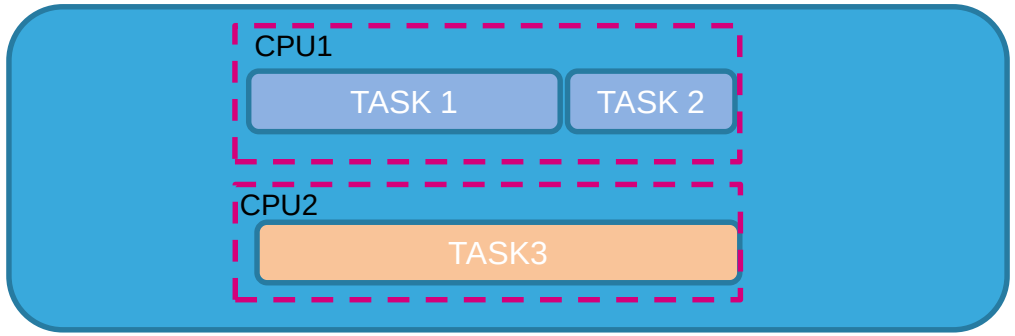
STM32H7 功能划分



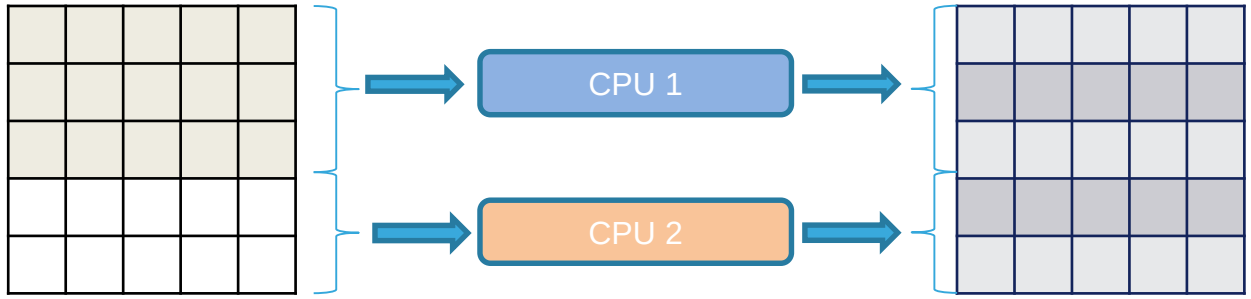
双核应用的设计

并行应用和依据性能的划分

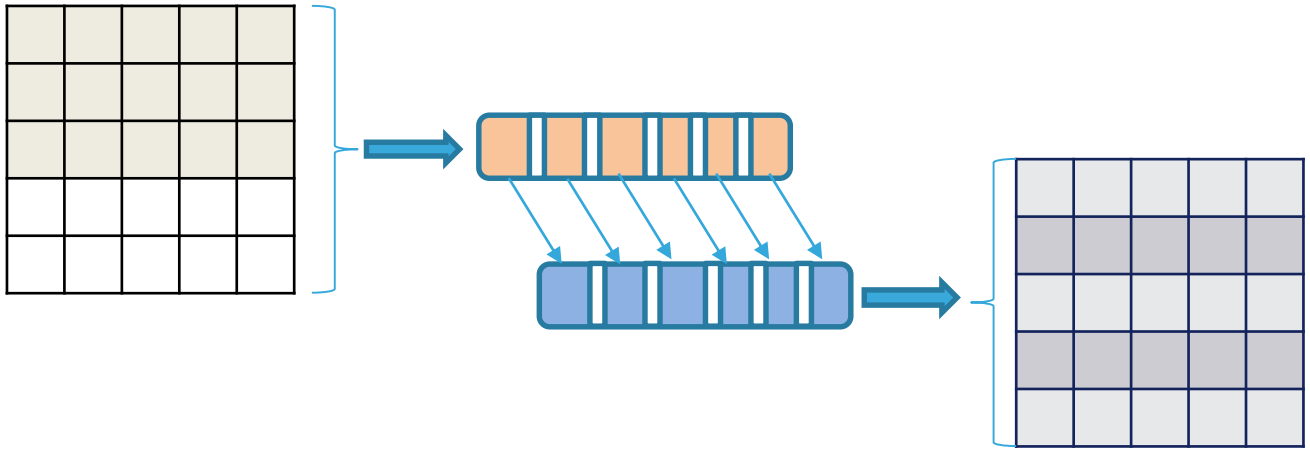
- 任务并行的划分



- 数据处理的划分



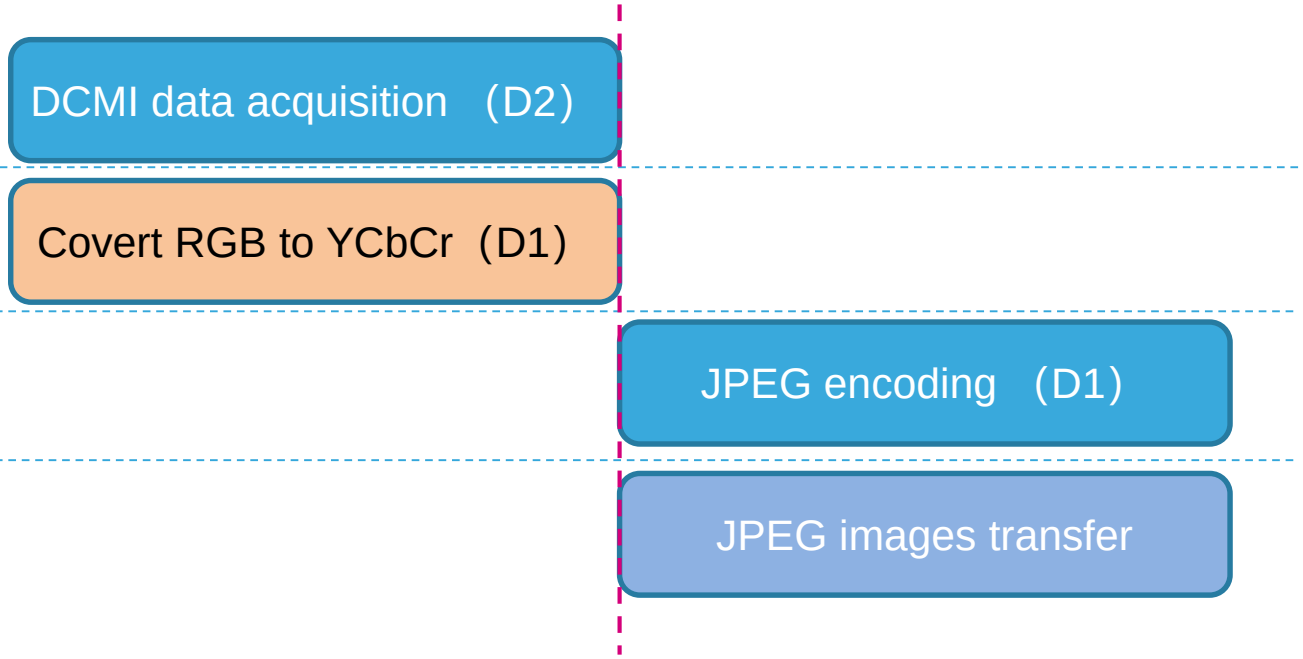
- 流水线划分



用户案例1

JPEG images 远程摄像头

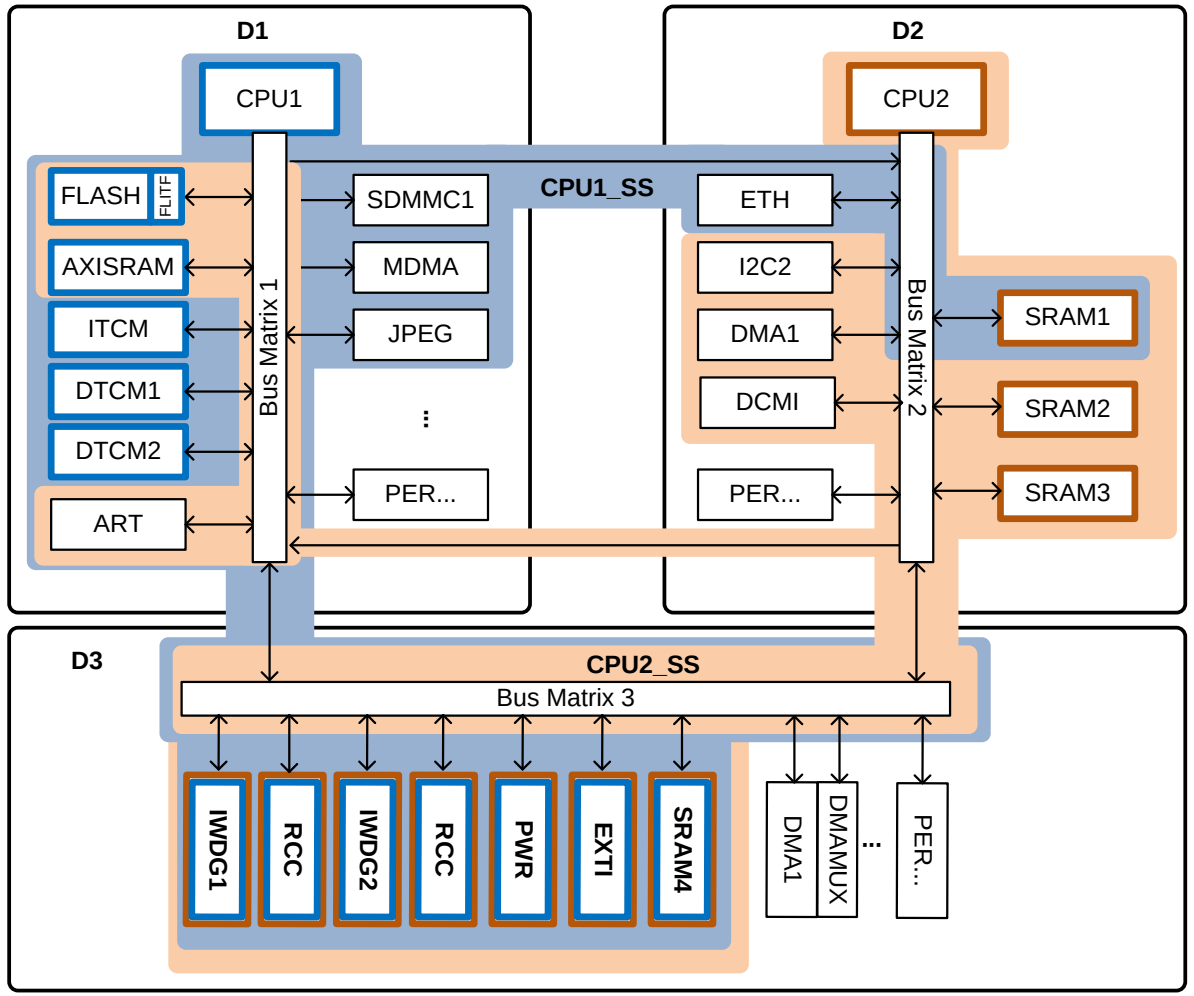
- 任务划分的例子：



- 外设资源:
  - Flash, AXISRAM, SRAMs, I2C, DCMI, DMA1, MDMA, Ethernet
- 任务的分布
  - **CPU2**:JPEG 图像传输
  - **CPU1**: 把RGB格式转换成 YCbCr 格式

用户案例1

CPUs 子系统



- D3域的RCC 使能CPU域的外设
  - 有利于降低整个系统的功耗
  - 被使能的外设也可以被第二个CPU访问
- 除了Flash, AXISRAM外， 根据不同的应用需求，来分配外设和存储器资源供哪个CPU使用:
  - JPEG HW codec, MDMA, DMAs, DCMI, …
- 对于多个CPU共享的SRAM1或者AXISRAM，需要软件保护来避免数据出错
  - MPU regions 可以用来隔离不同任务
    - 数据
    - 外设

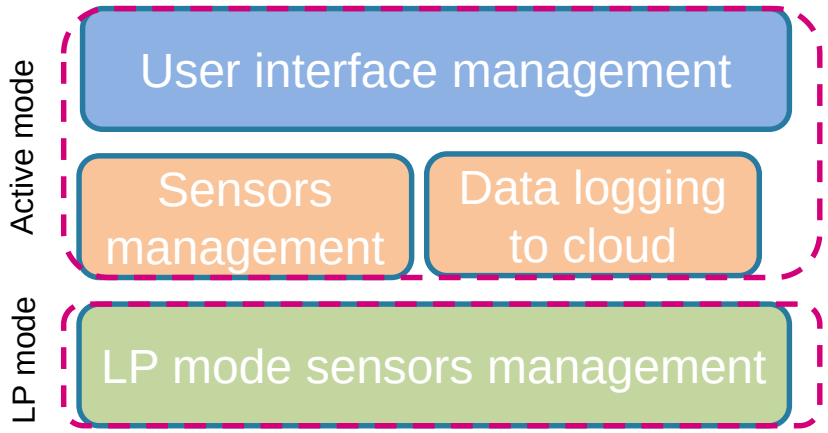
用户案例1

- 图像获取和预处理在CPU2的域里完成
  - 最优化的处理是数据直接放在CPU2域的SRAM中
- 数据的传输和编码通过 DMAs (DMA1/MDMA) 和 JPEG HW codec完成
  - DMA可以直接从 DCMI接口去数据到SRAM，降低CPU2的负载
  - MDMA 能够从 D2 SRAM中去数据，通过 JPEG HW codec来编码
- Cortex-M7 把编码完成后的图像流上传到网络上

用户案例2

传感器数据获取以及数据上传到云

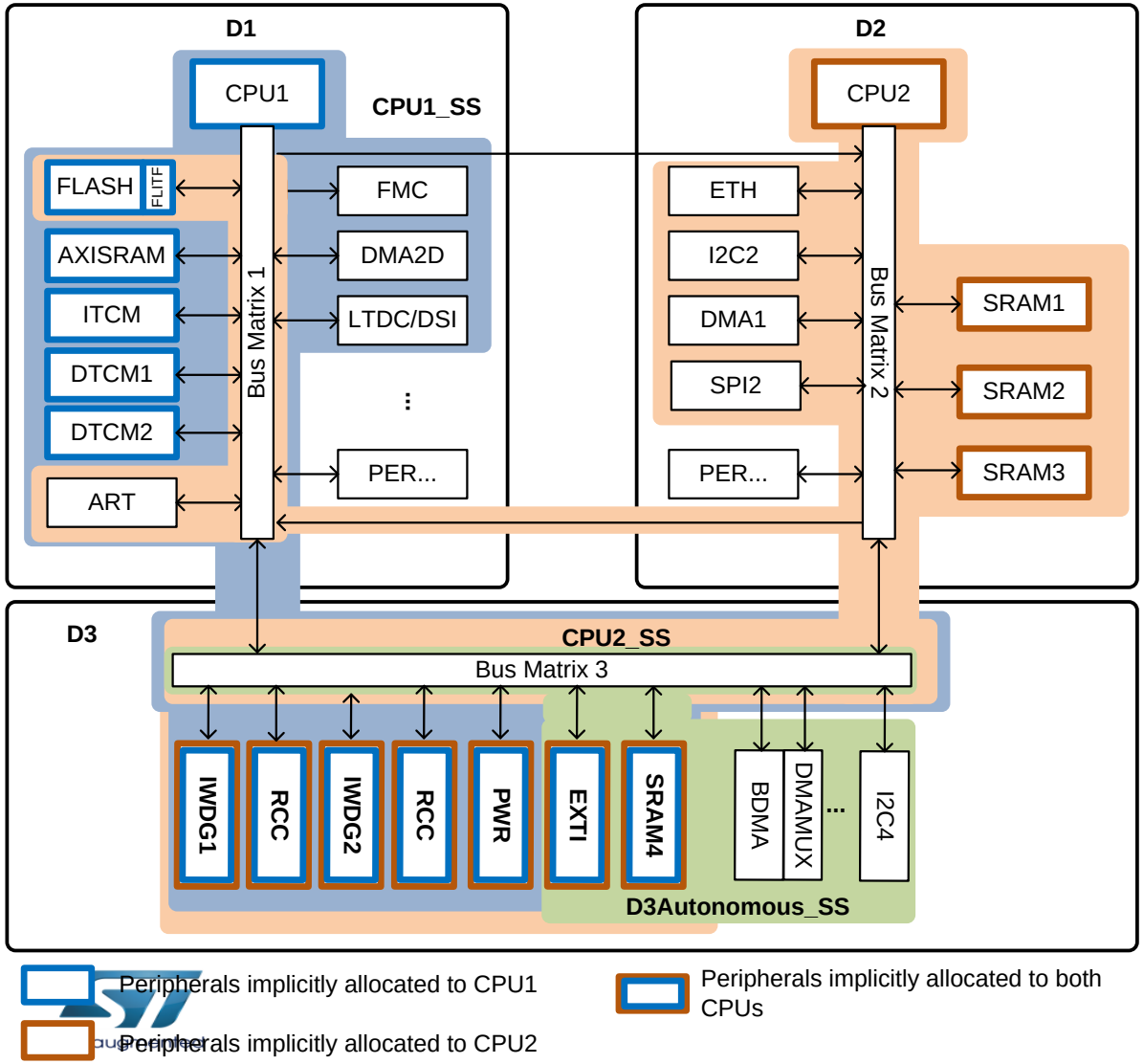
- 任务划分例子



- 外设资源:
  - Flash, AXISRAM, SRAMs, I2Cs, LTDC/DSI, DMA2D, DMA, BDMA
- 任务分配
  - **CPU1**: User interface 管理
  - **CPU2**: Sensors 管理 + Data 传到云端
  - **D3 autonomous**: LP mode sensors 管理

用户案例2

CPUs 子系统



- 用户界面的图形处理子系统分配给 CPU1
  - LTDC/DSI
  - DMA2D
  - FMC
- Ethernet, sensor 管理接口和 Flash 分配给 CPU2
- D3 自主模式 BDMA 和 sensor 管理接口 (SPI, I2C,…) 分配给D3域。
- 为了更好的节省功耗，分配给CPU2的Flash，如何不用的时候可以被释放，这样可以关断D1域的时钟和电源， 达到更好的功耗。

用户案例2

- CPU2 能够把CPU1从计算密集型的操作中解放出来，像加密、云通信
  - 把用户界面和耗时的任务之间隔离开来，保证系统能及时的响应

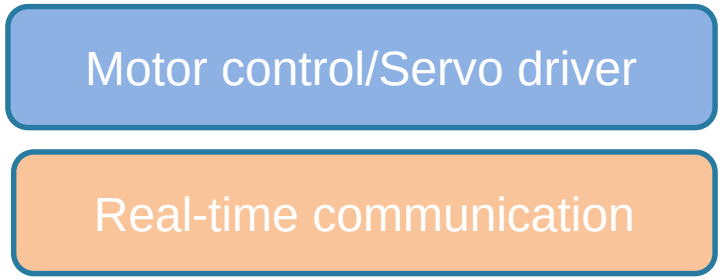
- 降低不同域之间的依赖性:
  - 通过独立的关掉不同域，来达到更低的功耗
    - CPU1主要使用D1 域的资源来实现用户界面管理，当CPU2进入低功耗模式时，D1域可以被关断。
    - 当Sensor在D3域的自主模式时，D1 和 D2 域可以同时被关断。

- 一个合适的数据映射，可以降低数据在不同存储器之间的搬移。 (例如 D3域的sensor数据可以直接使用SRAM4)

用户案例3

电机控制和实时通信

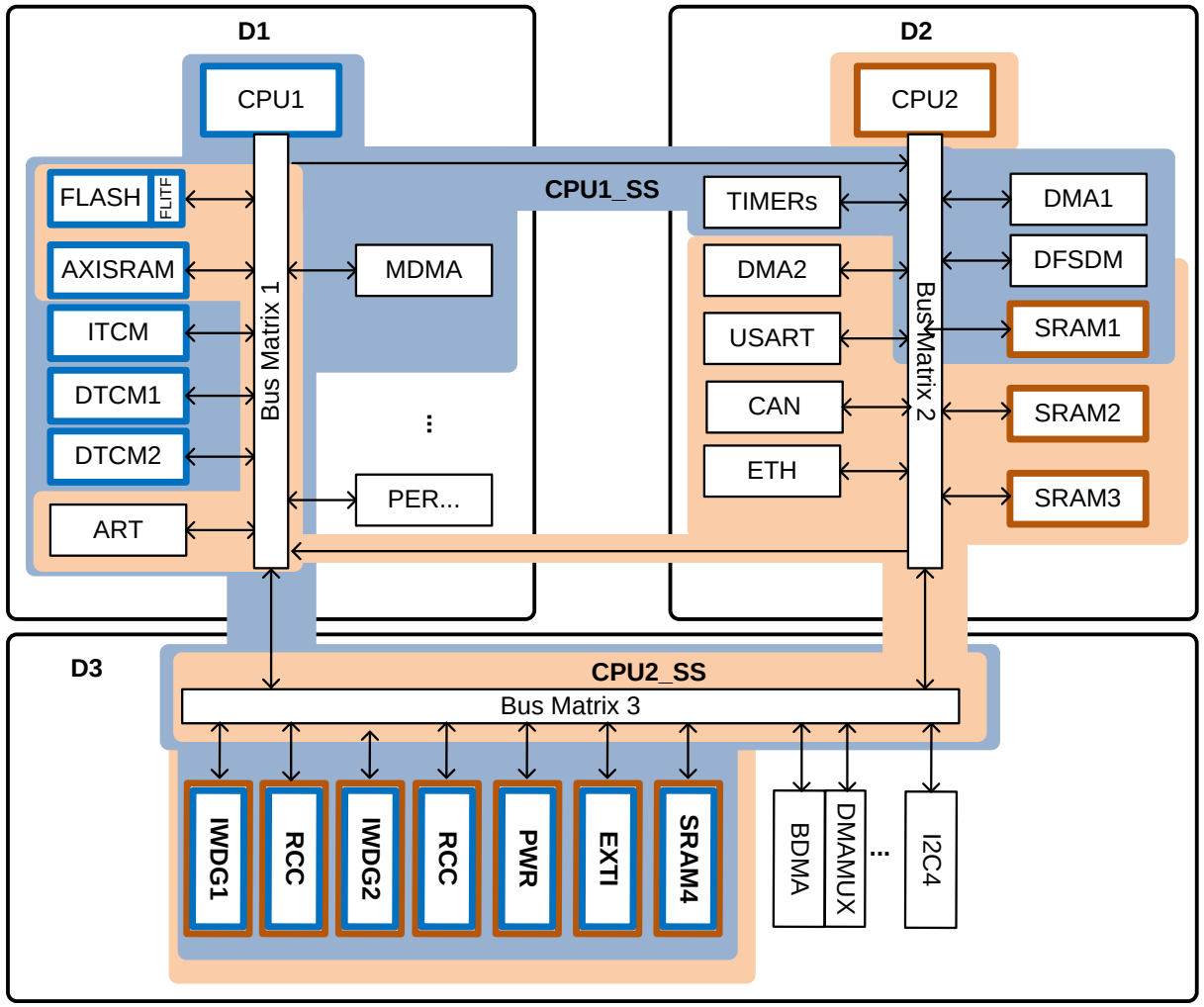
- 任务划分例子
  - 功能划分



- 外设资源:
  - Flash, AXISRAM, SRAMs, TIMERs, CAN, UART, ADCs/DFSDM, DMAs, Ethernet
- 外设和存储器分配
  - **CPU1**: 电机控制
    - Timers
    - ADCs/DFSDM
  - **CPU2**: 实时通信
    - Ethernet
    - CAN
    - USART

用户案例3

CPUs 子系统



Peripherals implicitly allocated to CPU1

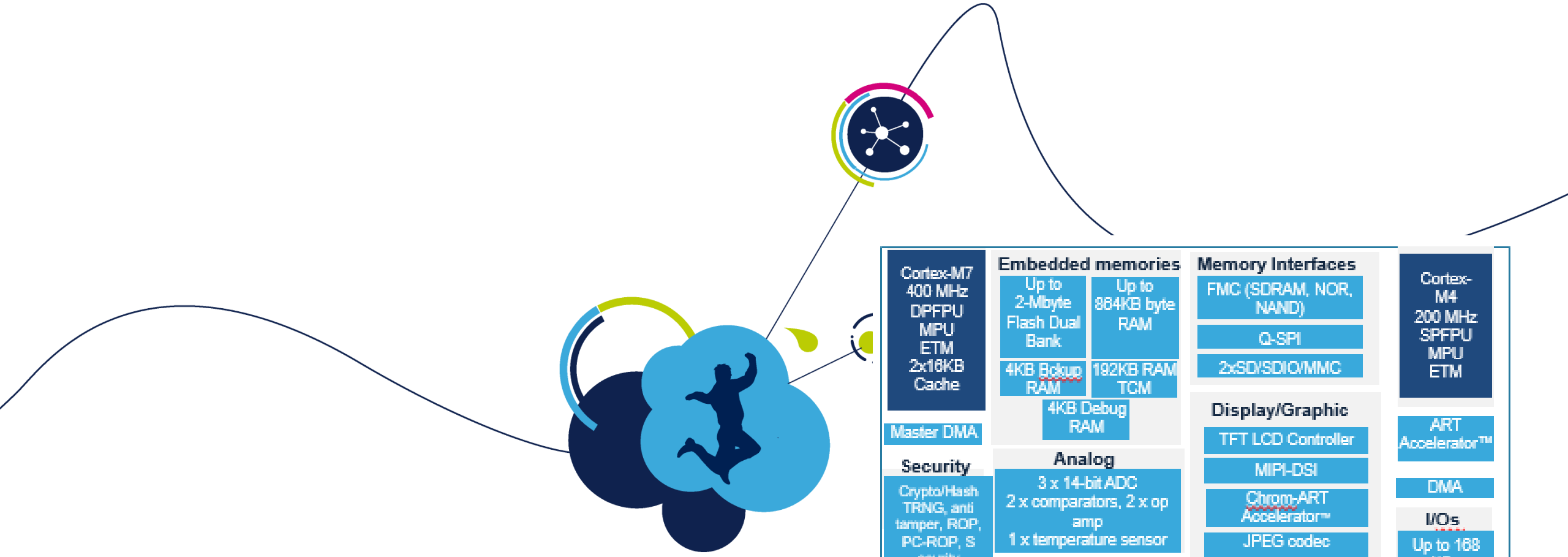
Peripherals implicitly allocated to CPU2

Peripherals implicitly allocated to both CPUs

- 电机控制子系统分配到 CPU1
  - DMA1
  - DFSDM
  - TIMERs
  - SRAM1
- 通信子系统分配到 CPU2
  - DMA2
  - USART
  - CAN
  - ETH
- 避免不同应用之间的资源共享，从而减少等待时间。
  - 例如： DMA1 分配给CPU1 使用，DMA2 分配给CPU2使用。

用户案例3

- 把实时性的任务和其它任务隔离开来，让他们独立的运行在不同的处理器内核上。
  - MPU 的配置可以让两个不同内核直接的数据避免冲突
- CPU2 用来处理通信相关的任务 (实时/标准):
  - Ethernet 协议栈
  - CAN 通信
- CPU1：电机控制/ 伺服驱动

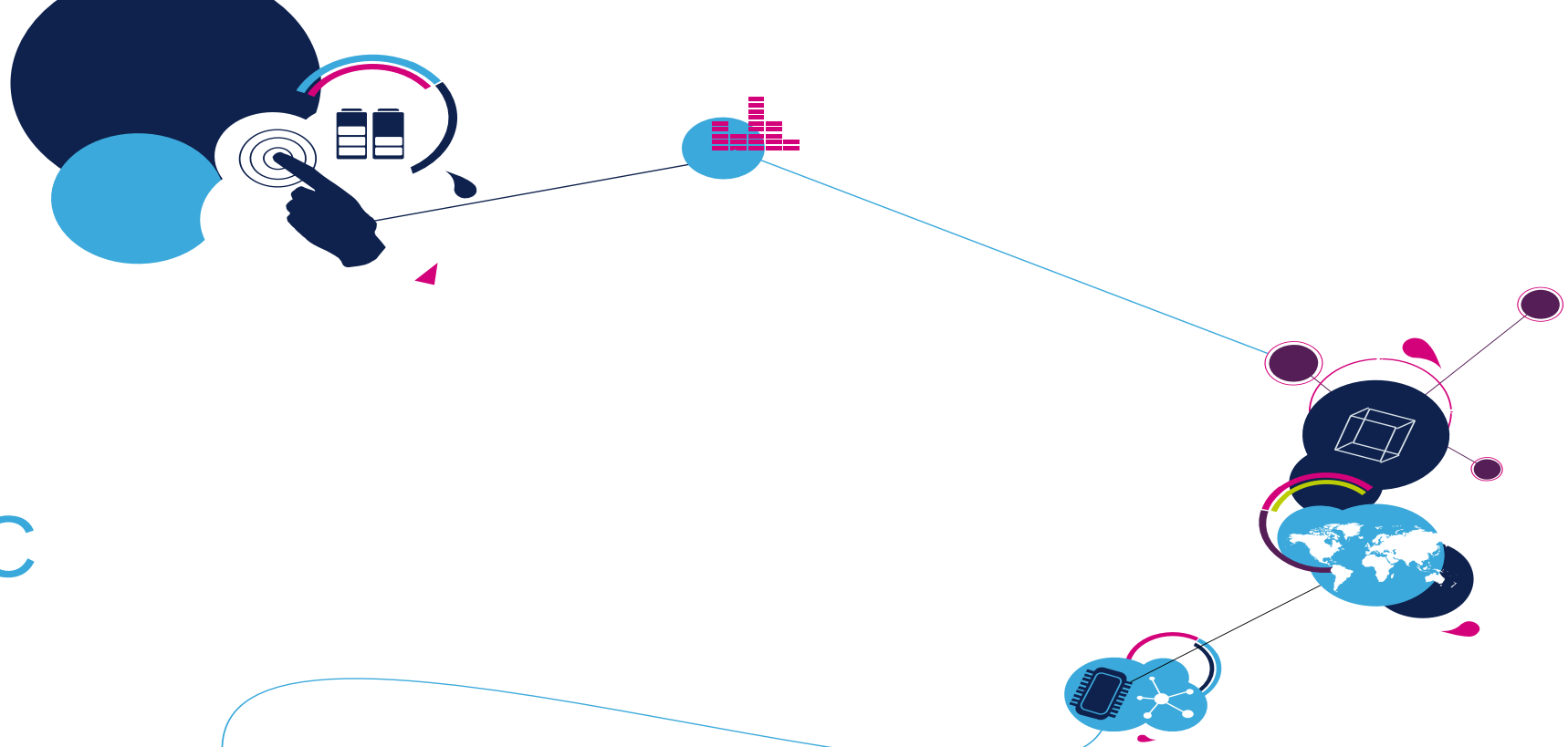


双核的同步和通信

life.augmented

内容安排

- STM32H7 系统架构
  - 单核及双核STM32H7系统框图
  - AXI 总线及互联
  - ART 加速器
- STM32H7 存储器 结构
- STM32H7 Cache
- STM32H7 MPU
- STM32H7 启动方式
- *STM32H7-CPUs 概述*
  - *STM32H7 性能*
- STM32H7 双核优势和应用
- **STM32H7 双核的同步和通信**
  - **IPC**
  - *HSEM*
- STM32H7 电源管理
- STM32H7外设
  - DMA
  - 16位 ADC
  - LTDC MIPI 控制器
  - DLYB
  - FMC
- STM32H7目前的资源



STM32H7 IPC

Inter-Processor Communication

life.augmented

为什么 Inter-procesors communication (IPC)?

- 运行在不同CPU内核上的应用可能是独立的，也可能是相互协作的
- 很多情况下，需要一种机制来实现：
  - 数据共享
  - 资源共享和同步
- IPC 是一种机制，用于不同内核时间的通信和数据交换
- 这种机制可以保证：
  - 安全的并行运算Safe parallel computation
  - 同步
  - 应用程序的模块化

IPC 机制

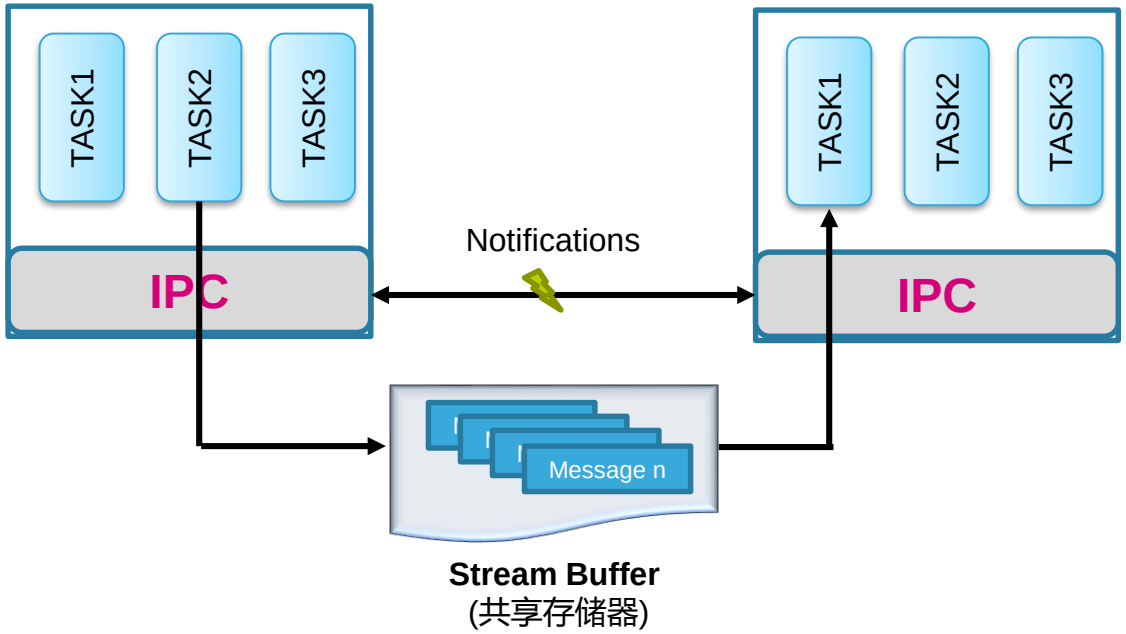
- 消息的传输
  - **Send** 消息到其他的处理器
  - 从其他的处理器 **Receive** 消息
- 共享存储器
  - 一个处理器和其他处理器共用一块公共的存储区域
- 处理器同步的实现方式
  - 使用信号量
  - 使用中断

STM32H7 IPC 解决办法

- STM32H7两种可以使用的解决方案
  - 使用设备资源建立通信/ 同步协议协议
  - 使用 FreeRTOS IPC 模块 (在STM32Cube H7 Firmware)
- IPC 模块可以用在 :
  - FreeRTOS OS
  - 或者 OS-less 的应用
- IPC 模块用 :
  - 用Stream buffers 在不同 CPUs之间数据传输
  - 使用内核共享的存储器区域来分配 Stream buffers

FreeRTOS IPC 模块

- IPC 模块是一整套 APIs， 用于不同应用任务之间交换数据
  - 每个Task有一个特定的 Stream buffer (一个发送，一个接收)



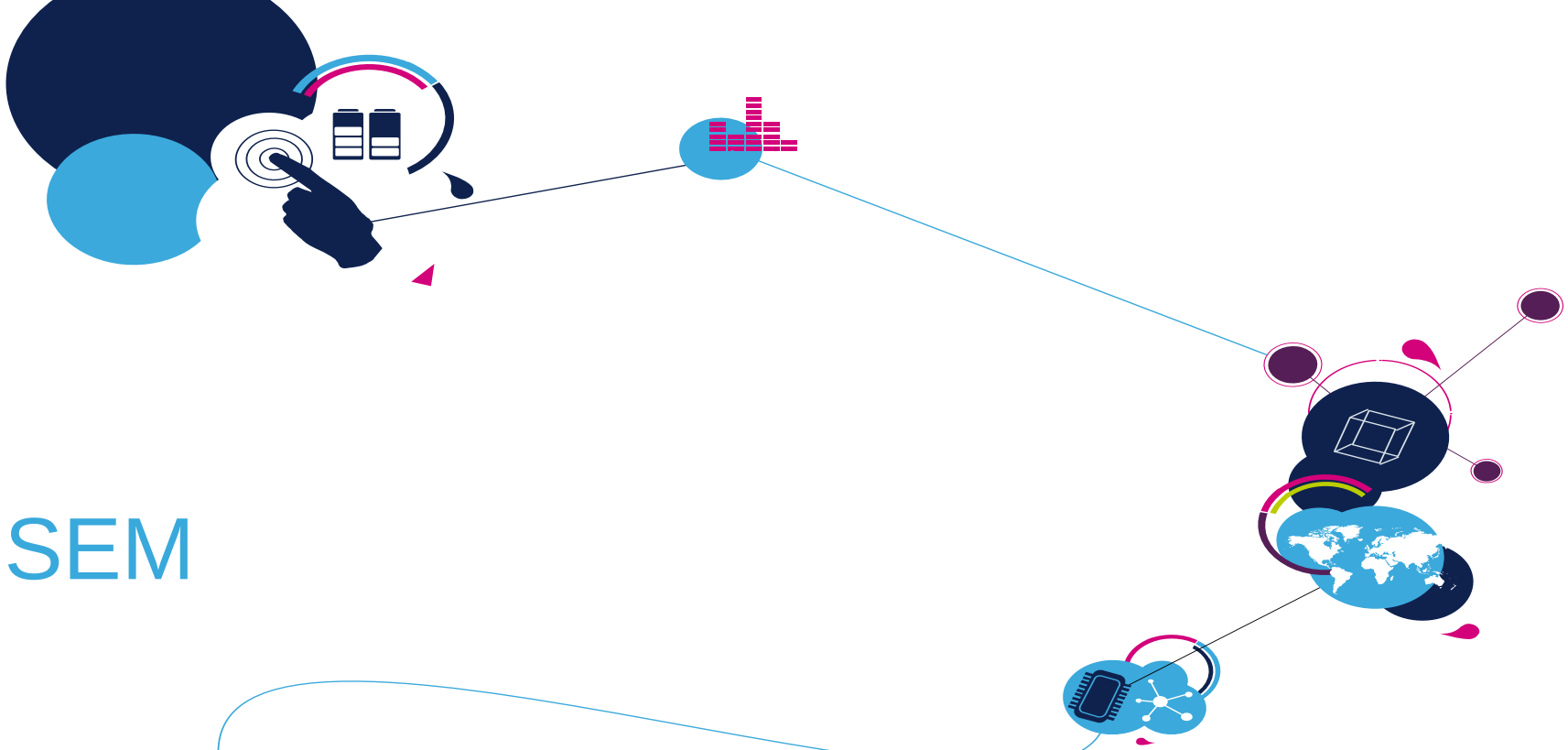
FreeRTOS IPC 模块

- 目前的IPC 模块定义了一个很友好的 APIs

功能	描述
pxStreamBufferCreateBlocking ()	Create The Stream Buffer located in a shared Memory
uxStreamBufferSend()	Send message to the other Core
uxStreamBufferReceive()	Receive message from the other Core

内容安排

- STM32H7 系统架构
  - 单核及双核STM32H7系统框图
  - AXI 总线及互联
  - ART 加速器
- STM32H7 存储器 结构
- STM32H7 Cache
- STM32H7 MPU
- STM32H7 启动方式
- *STM32H7-CPUs 概述*
  - *STM32H7 性能*
- STM32H7 双核优势和应用
- **STM32H7 双核的同步和通信**
  - IPC
  - ***HSEM → 动手实验***
- STM32H7 电源管理
- STM32H7外设
  - DMA
  - 16位 ADC
  - LTDC MIPI 控制器
  - DLYB
  - FMC
- STM32H7目前的资源



STM32H7 - HSEM

Hardware Semaphore

life.augmented

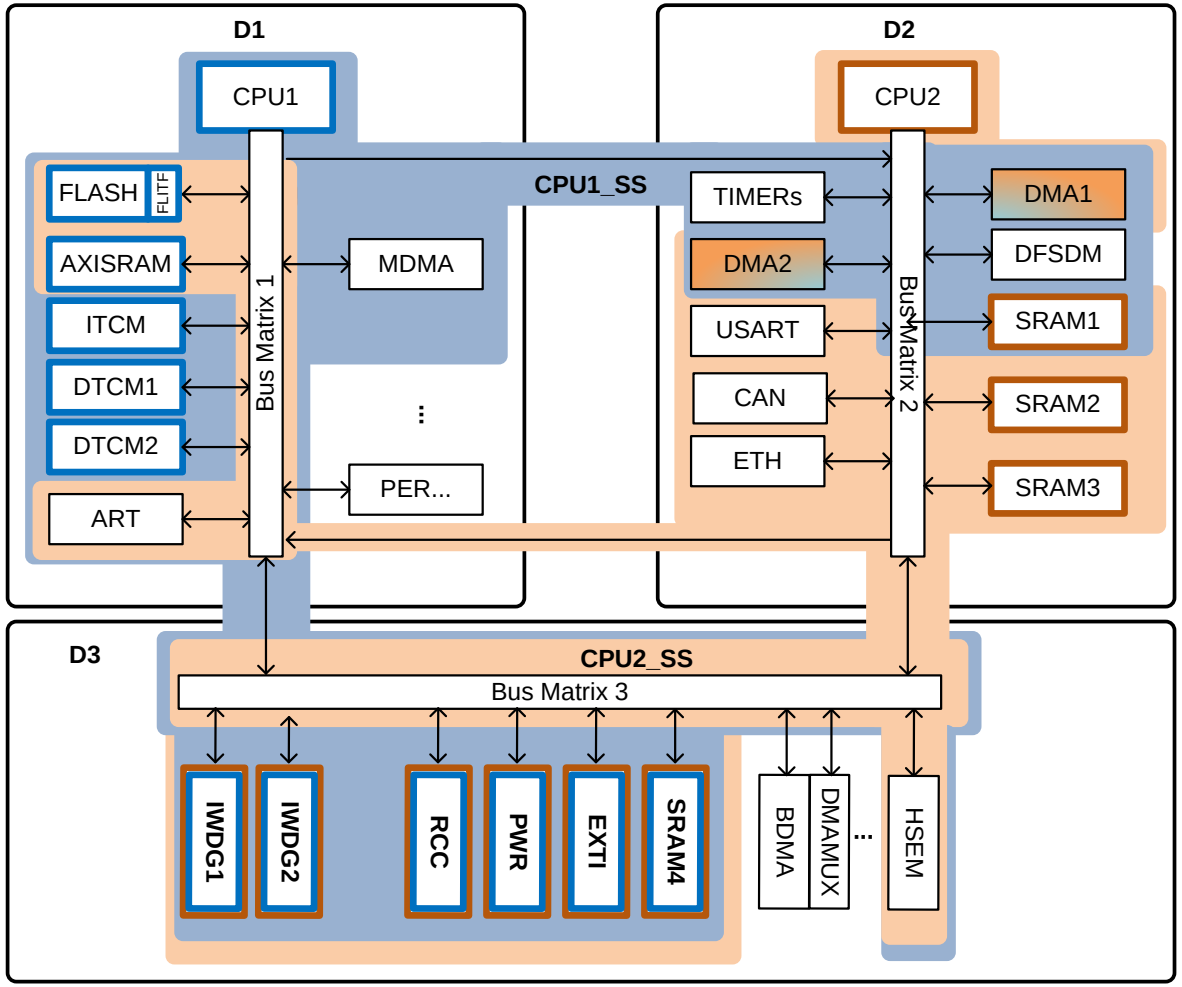
- 管理访问权限和同步
  - 不同的进程运行在相同的CPU上
  - 不同的CPU
- 32信号量
- 两种锁机制
  - 2-step write, read back lock
  - 1-step read lock
- 信号量释放中断生成

应用好处

- 防止共享资源访问冲突
- 确保在进程之间进行同步
- 没有阻塞信号处理

应用例子

电机控制和实时通信

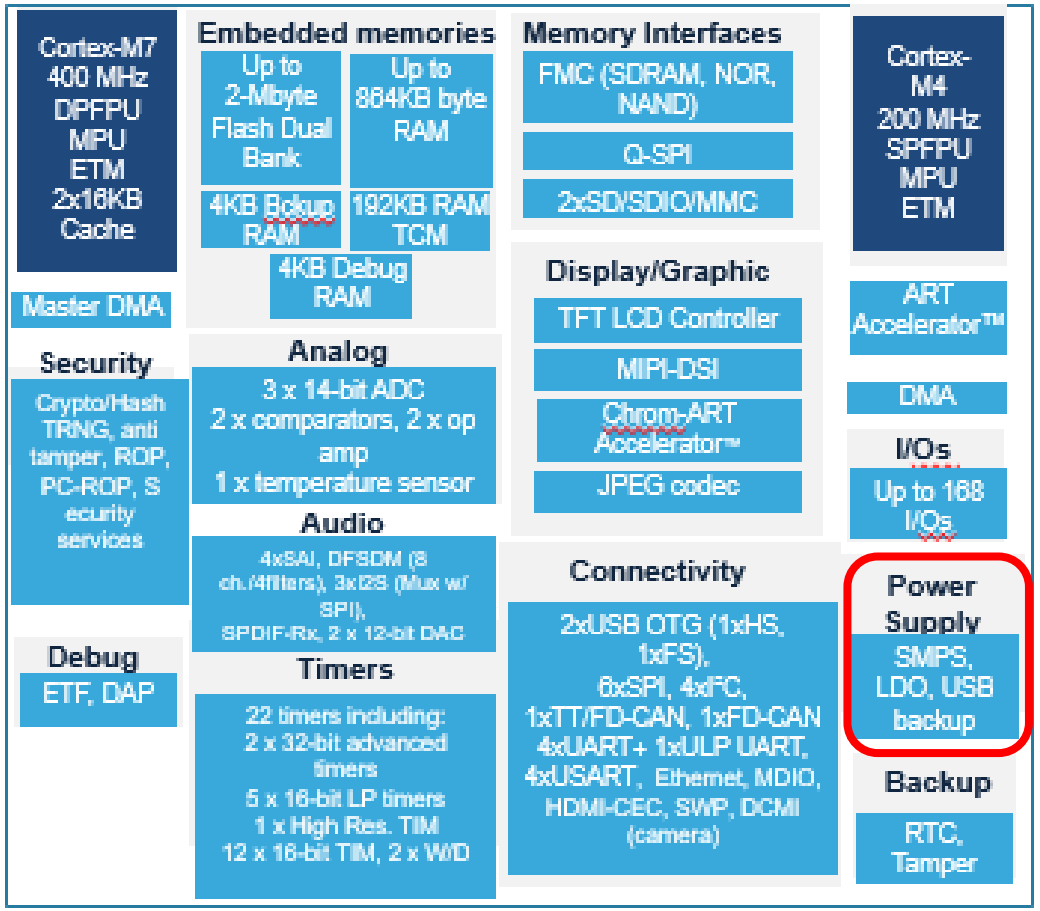


- HSEM can help to manage shared resources such as DMA controller,… when it is needed by the two applications on CPU1 and CPU2
- HSEM can be used on the same application running on CPU2 for task synchronization

内容安排

- STM32H7 系统架构
  - 单核及双核STM32H7系统框图
  - AXI 总线及互联
  - ART 加速器
- STM32H7 存储器 结构
- STM32H7 Cache
- STM32H7 MPU
- STM32H7 启动方式
- *STM32H7-CPUs 概述*
  - *STM32H7 性能*
- STM32H7 双核优势和应用
- STM32H7 双核的同步和通信
  - IPC
  - *HSEM → 动手实验*
- **STM32H7 电源管理**
- STM32H7外设
  - DMA
  - 16位 ADC
  - LTDC MIPI 控制器
  - DLYB
  - FMC
- STM32H7目前的资源

STM32H7 电源管理



Arm® Cortex®-M7 – 480 MHz

CORE, MEMORIES AND ACCELERATION

- Single-core Cortex-M7 480 MHz
- Dual-core Cortex-M7 480 MHz and Cortex-M4 240 MHz (STM32H7x5 and STM32H7x7 only)
- Flash and RAM acceleration
- SP-FPU and DP-FPU
- 4 x DMA

CONNECTIVITY

- 2 x USB2.0 OTG FS/HS
- 2 x SDMMC
- USART, UART, SPI, I²C
- 2 x CAN (1 x FD and 1 x TT)
- HDMI-CEC
- FMC, Dual-mode Quad-SPI
- Ethernet MAC IEEE1588
- Camera I/F
- Analog (comp, AOP)

AUDIO

- 3 x I²S + audio PLL
- 4 x SAI
- 2 x 12-bit DAC
- SPDIF-RX

GRAPHIC

- Chrom-ART Accelerator™

OTHER

- Crypto/Hash (except H742)[1]
- Security services (except H742)
- TRNG
- DFSDM
- 16- and 32-bit timers
- 3 x 16-bit ADC (up to 3.6 MSPS)
- Voltage range 1.62 to 3.6 V (except 100-pin package : 1.71 to 3.6 V)
- Multi-power domains

STM32 H7 Product line	f_{CPU} (MHz)	Dual-Bank Flash memory (bytes)	RAM (bytes)	Graphic	Power supply	Ambiant temperature range
Dual-core lines						
STM32H7x7	480 + 240	Up to 2 Mbytes	1 Mbyte (incl.128 Kbytes DTCM + 64 Kbytes ITCM + 64 Kbytes backup1) + 4 Kbytes backup2	TFT-LCD JPEG codec MIPI-DSI	DCDC + LDO	Standard 85 °C
STM32H7x5	480 + 240	Up to 2 Mbytes	1 Mbyte (incl.128 Kbytes DTCM + 64 Kbytes ITCM + 64 Kbytes backup1) + 4 Kbytes backup2	TFT-LCD JPEG codec	DCDC + LDO	Standard 85 °C (Opt. Industrial CPN 125 °C)[2]
Single-core lines						
STM32H7x3	480	Up to 2 Mbytes	1 Mbyte (incl.128 Kbytes DTCM + 64 Kbytes ITCM + 64 Kbytes backup1) + 4 Kbytes backup2	TFT-LCD JPEG codec	LDO	Standard 85 °C
STM32H742	480	Up to 2 Mbytes	692 K (incl.128 Kbytes DTCM + 64 Kbytes ITCM + 16 Kbytes backup1) + 4 Kbytes backup2	no	LDO	Standard 85 °C
Value line						
STM32H750	480	128 Kbytes	1 Mbyte (incl.128 Kbytes DTCM + 64 Kbytes ITCM + 64 Kbytes backup1) + 4 Kbytes backup2	TFT-LCD JPEG codec	LDO	Standard 85 °C

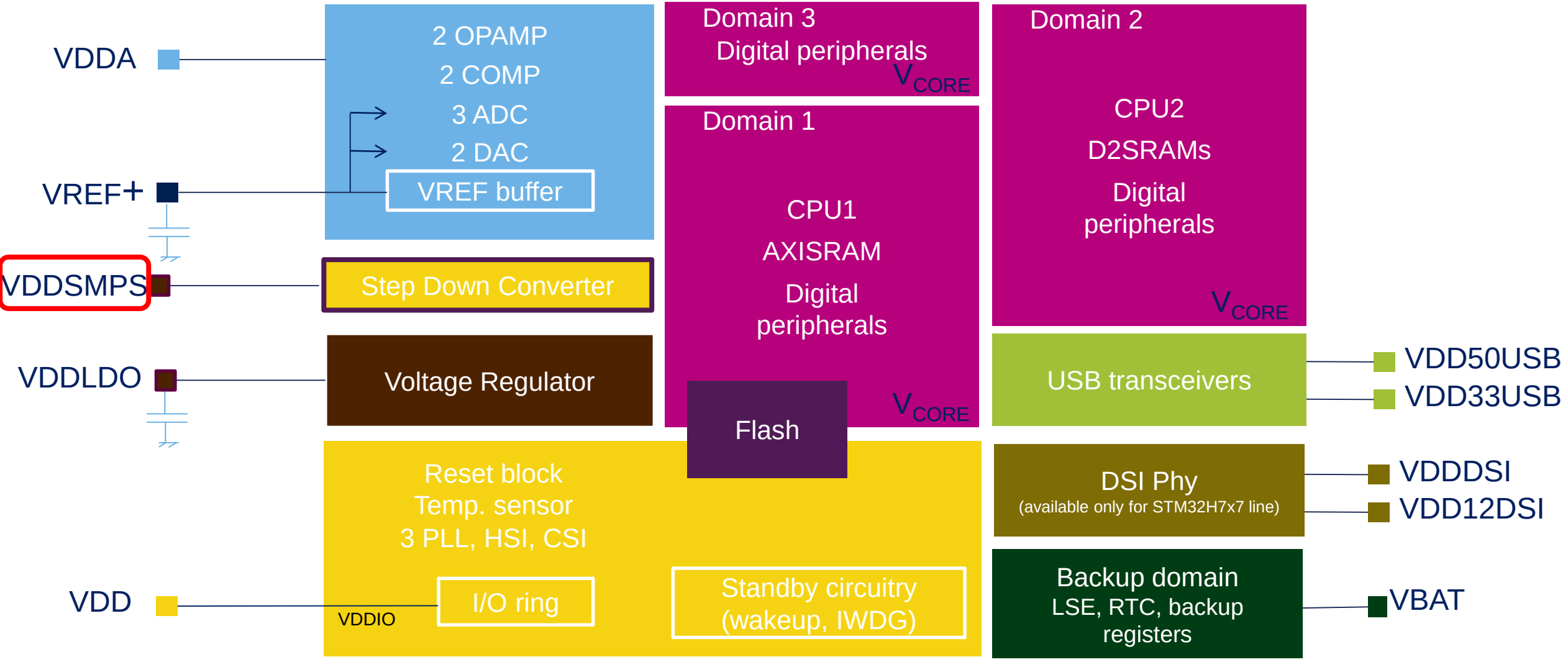
Notes :
1 : optional - dedicated CPN, STM32H750, STM32H753, STM32H755, STM32H757 for the Crypto Variants
2 : available Q4-2019 - Maximum extended temperature range: 125 °C ambiant / 140 °C junction. Dedicated part numbers

电源总述

- 提供电源管理和功耗控制
  - 不同的供电配置
  - 电压 动态调节（Scaling）
  - 不同域的功耗控制
  - 低功耗模式唤醒
- 5种低功耗模式，可快速唤醒
- VBAT backup 模式，RTC 和 backup 寄存器
- 独立的供电

应用优势

- 内部集成DC-DC电压转换器，降低系统功耗.
- 优化功耗：
  每个域的电压控制
  动态电压调整
- Autonomous D3 domain 操作



life.augmented

独立的电源供电优化电源和性能

- V_{DD} from 1.71 to 3.6 V (1.62 to 3.6V with external supply supervisor)
- V_{BAT} from 1.2 to 3.6V including the RTC and 128-byte backup registers
- **V_{DDSMPS} from 1.8(*) to 3.6 V (when $V_{DDSMPS} \leq 1.2$ V)**
- V_{DDLDO} from 1.62 to 3.6 V ($V_{DDLDO} \leq V_{DD}$)
- V_{DDA} from 1.62 to 3.6 V
  - 1.62 V min. when ADCs or COMPs are used
  - 1.8 V when DACs are used
  - 2.0 V when OPAMPs are used
  - 1.8 V min. when VREFBUF is used
- $V_{DD50USB}$ from 4.5 to 5.5 V for USB regulator
- $V_{DD33USB}$ from 3 to 3.6 V for USB transceivers
- V_{DDDSI} from 3 to 3.6 V for DSI regulator
- $V_{DD12DSI}$ from 1.2V for DSI PHY

life.augmented

(*) V_{DDSMPS} is limited to 2.2 to 3.6 V on revY Silicon

供电监管

供电监管实现动态功耗管理

- V_{DD}, & V_{DDSD} via POR/PDR, BOR (reset), and PVD (threshold interrupt on EXTI).
- V_{DDA} via AVD (threshold interrupt on EXTI)
- V_{BAT} via VBAT threshold (interrupt via Tamper)
- V_{CORE} Core domain supply, via level detector (reset).

  Core domain over voltage detection (forces SD to 1.0V, ACTVOSRDY signal invalid)
- V_{SW} Backup domain supply, via level detector (reset).
- V_{BKP} Backup domain RAM supply, via level detector (ready register bit BRRDY)
- $V_{DD33USB}$ USB regulated supply, via level detector (ready register bit USBRDY)
- V_{FBSMPS} SD regulated supply, via level detector (ready register bit SDEXTRDY) New
- V_{CAPDSI} DSI regulated supply, via level detector (ready register bit RRS) New

Step Down 电压转换器

- 用于转换VDD 给以下供电:
  - 直接给V_{CORE} 域供电
    - SD will follow the device operating modes (Run, Stop, Standby).
    - SD output level will be according the selected VOS and SVOS.
  - 给内核的电压调节器LDO供电
    - SD operating modes will follow the device operating modes (Run, Stop, Standby).
    - SD output level will be according the selected SDLEVEL (1.8 V or 2.5 V).
  - 提供外部供电
    - SD forced always on in High Performance operating mode.
    - SD output level will be according the selected SDLEVEL (1.8 V or 2.5 V).

- 操作模式
  - 高性能 (Run or External supply)
  - 低功耗 (Stop)
  - 关断 (Standby)

LDO 电压调节器

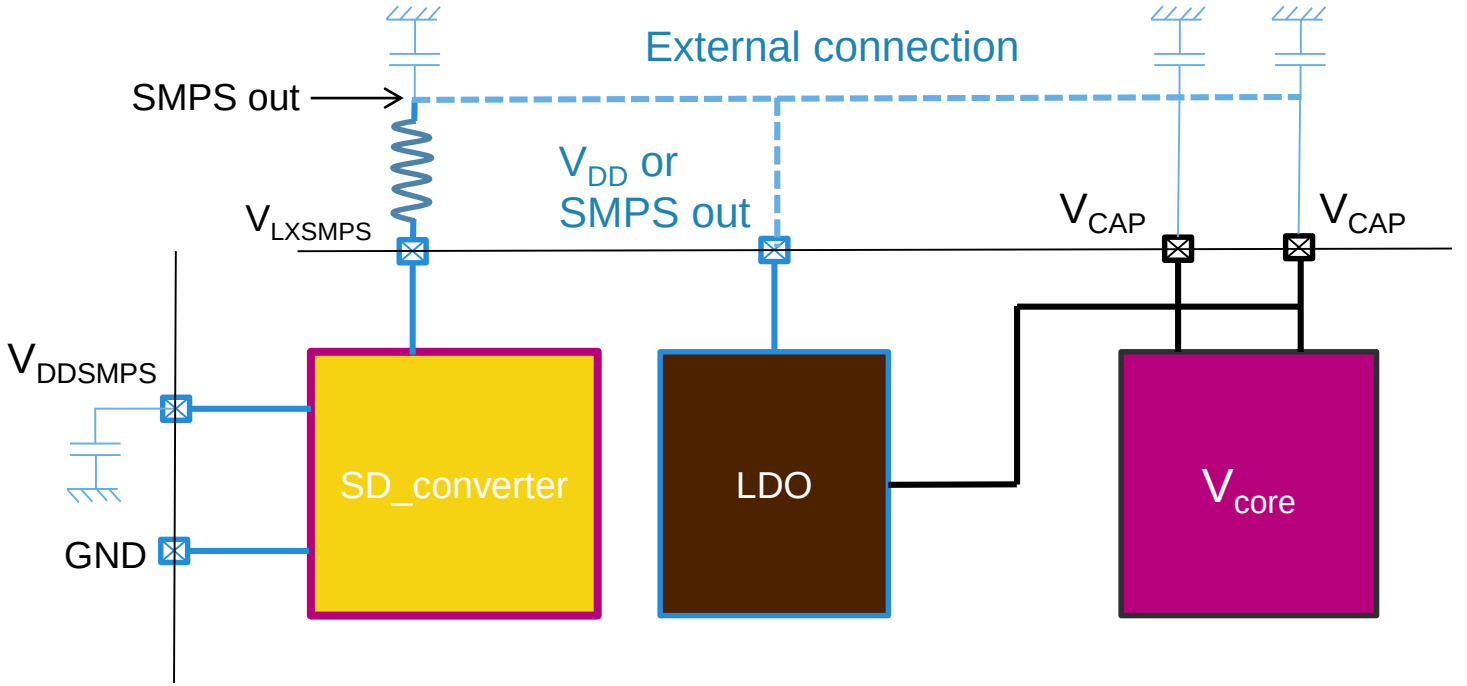
- 用于把V_{DDLDO} 的电压调节到 V_{CORE}的电压
  - Directly supply the V_{CORE} domains
    - LDO operating modes will follow the device operating modes (Run, Stop, Standby).
    - LDO output level will be according the selected VOS and SVOS.
  - Bypass
    - LDO is off, the Core domain is directly supplied on V_{CORE}.
- Operating modes
    - Main (Run)
    - Low Power (Stop)
    - Off (Standby)

Backup domain regulator

- Operating modes
  - On
  - Off
- Used to regulate the V_{SW} supply to the Backup RAM level
  - When V_{CORE} is present
    - Backup domain regulator is off. (Allows to save backup battery consumption)
    - Backup RAM is supplied from V_{CORE}.
  - When V_{CORE} is absent (STANDBY or VBAT mode) and backup regulator is enabled
    - Backup domain regulator is on.
    - Backup RAM is supplied from V_{BKUP}.
  - When V_{CORE} is absent (STANDBY or VBAT mode) and backup regulator is disabled
    - Backup domain regulator is off.
    - Backup RAM is powered down. (data lost)

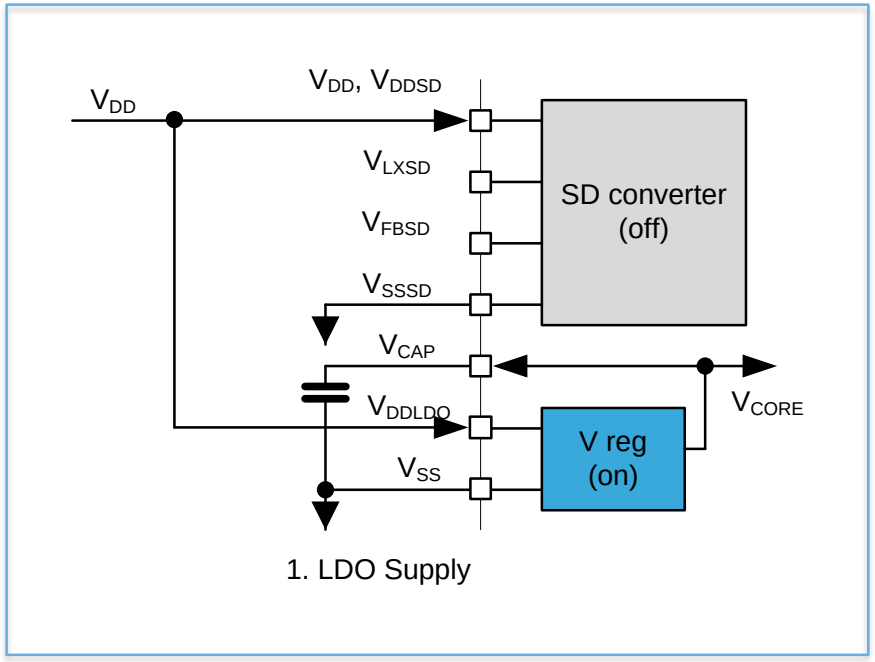
供电配置

- 两个电压调节器可以用来调节 V_{DD} 供给V_{CORE} 的电压
- 上电时默认的电压配置是：
  - SD converted 输出 = 1.2 V
  - LDO 输出 = 1.00 V
- 软件可以初始化电压的配置：
  - 先于任何的时钟配置和 SRAM访问
  - 电压配置是一次写入的
- 从Standby 模式退出时候，电压配置是继续保持的。



性能和功耗之间的灵活选择

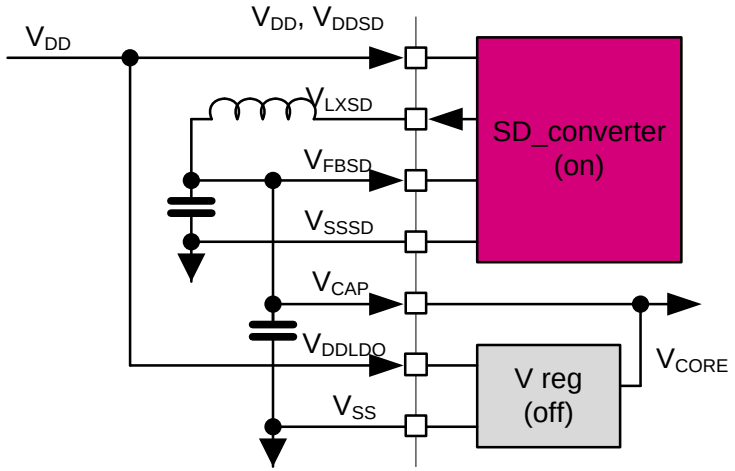
- 1. LDO 供电 (SD disabled)
  - LDO 由 VDD供电



1. LDO Supply

性能和功耗之间的灵活选择

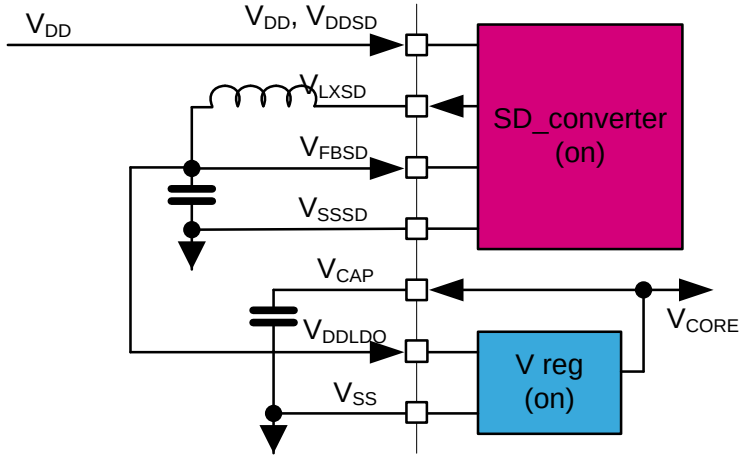
- 2. 直接 SD 供电 (LDO disabled)
  - Regulator follows device operation modes



2. Direct SD Supply

性能和功耗之间的灵活选择

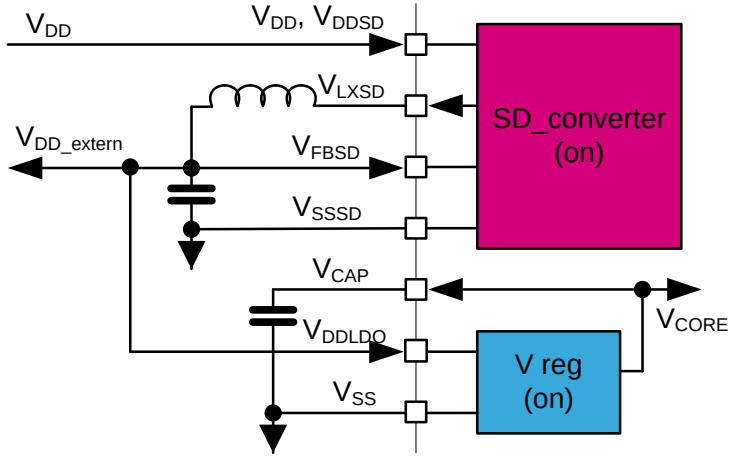
- 3. SD 给 LDO供电 (不给外部供电)
  - Both regulators follow device operation modes



3. SD supplies LDO (No External supply)

性能和功耗之间的灵活选择

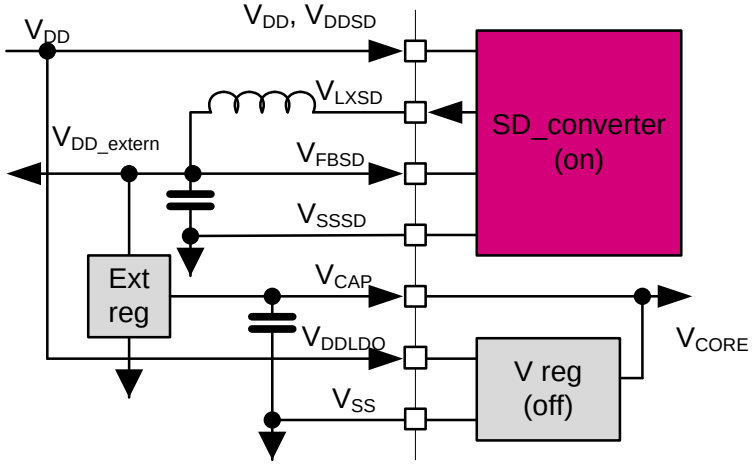
- 4. SD 给外部电路供电, 同时给LDO供电
  - SD 必须工作在高性能模式下
  - LDO follow device operation modes



4. External SD supply, supplies LDO

性能和功耗之间的灵活选择

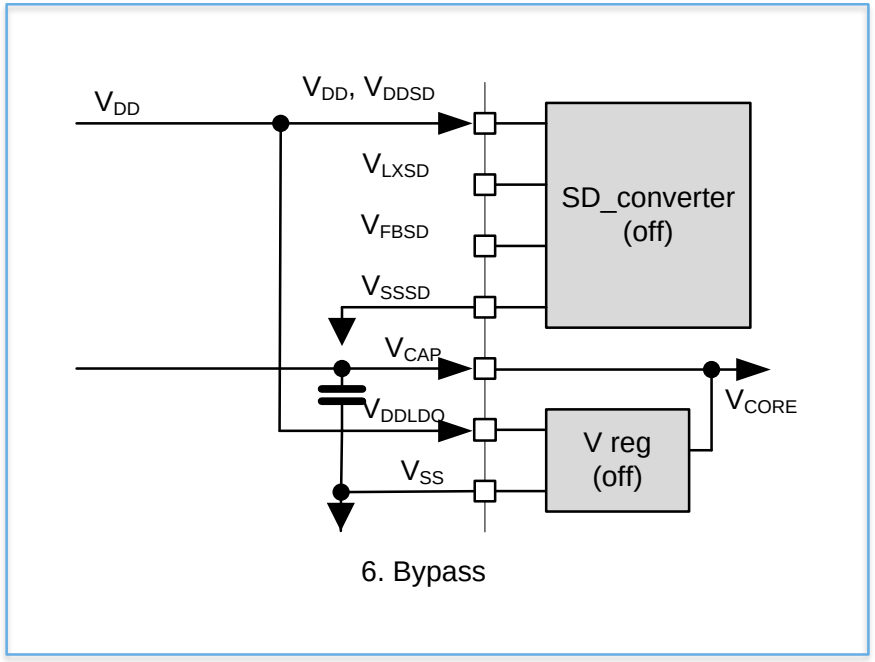
- 5. SD external supply & LDO bypass (LDO disabled)
  - SD 必须工作在高性能模式下



5. External SD Supply & Bypass

性能和功耗之间的灵活选择

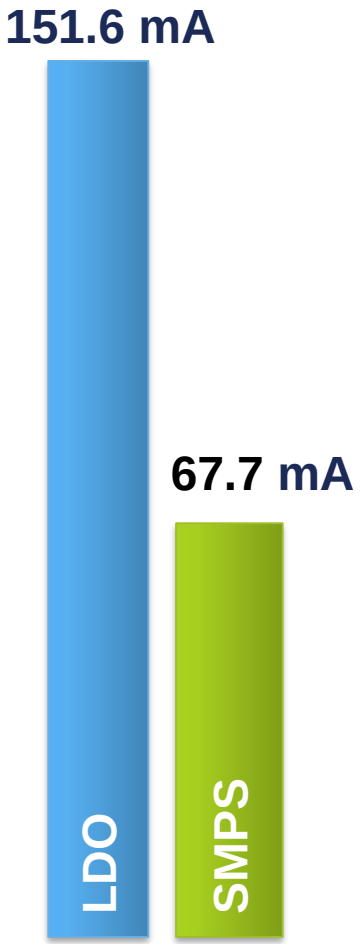
- 6. Bypass (SD & LDO disabled)



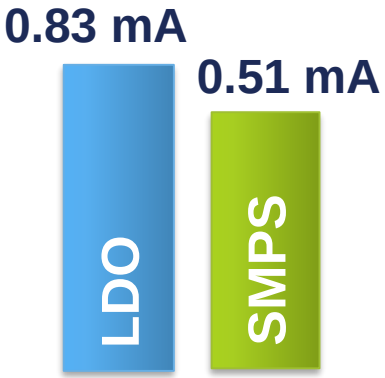
6. Bypass

功耗测量

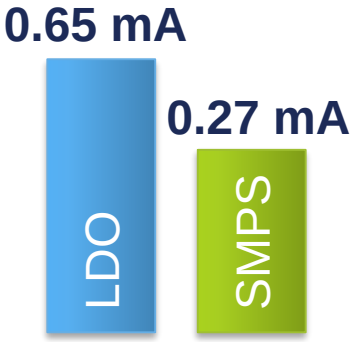
测量条件:
室温, VDD= 3.3 V
revY的芯片测量



Run Mode
CM7 400MHz exec from FLASH_A cache ON, CM4 200MHz exec from FLASH_B ART ON All periph disabled



Stop Mode
D1STANDBY, D2STANDBY, D3STOP VOS4



Stop mode
D1STANDBY, D2STANDBY, D3STOP VOS5

life.augmented

给模拟外设的独立电源参考电压

- V_{REF+} : ADCs 和 DACs的参考电压
  - 可以由外部的参考电压提供，或者通过内部的参考电压缓冲器。
  - V_{REF+} 引脚, 内部电压参考在TFBGA100的封装上不可以使用。这种封装上，这个引脚和 V_{DDA} 复用，用来连接外部参考电压。这时内部参考电压缓冲器不可以用。

动态的子系统配置，可以优化功耗

- 系统可以给子系统配置环境:
  - CPU1 (Cortex M7)
  - CPU2 (Cortex M4)
  - D3 autonomous
- 基于外设的分配，系统可以有不同的配置
  - 一个CPU 子系统可以由下面组成:
    - CPU 和 相关域的总线矩阵
    - D3 域的总线矩阵
    - 总线矩阵绑定的外设
    - 由相关域总线分配的外设 (PERxEN)
  - D3 域 自主模式由下面组成:
    - D3 域的总线矩阵
    - 自主模式相关分配的外设 (PERxAMEN)
      - Autonomous mode peripherals need also to be allocated by a CPU sub-system.

优化功耗，来自于：不同的 V_{CORE} 域，电压范围，独立的操作模式

- 电源控制 V_{CORE} 的电压和域的电压
  - V_{CORE} 域的划分
    - D1 domain – CPU1 (CM7), Flash, RAM, peripherals
    - D2 domain – CPU2 (CM4), RAM, peripherals
    - D3 domain – RAM, peripherals
  - 操作模式
    - CPU modes (CRun, CSleep, CStop)
    - Domain modes (DRun, DStop, DStandby)
    - System modes (Run, Stop, Standby)
  - 电压范围
    - RUN mode Voltage Scaling (VOS), provides 3 ranges.
    - STOP mode Voltage Scaling (SVOS), provide 3 ranges.

电压范围模式

性能和功耗之间的灵活配比

Voltage range	Mode	SYSCLK	CM7	CM4	D2/D3 domain	Stop mode Wakeup Peripheral
Range 1	**Run**	480 MHz max	480 MHz	240 MHz	240 MHz	Kernel clock on
Range 2	**Run**	360 MHz max	360 MHz	180 MHz	180 MHz	
Range 3	**Run**	240 MHz max	240 MHz	120 MHz	120 MHz	
	Stop	off	off	off	off	
Range 4	**Stop**					off
Range 5	**Stop**					

Range 1 is not available on industriel profile. Max frequency is 300MHz

RUN模式

- 每一个外设始终可以被配置为ON 或者 OFF
  - 重启时，所有的外设时钟都是 OFF, 除了 Flash 控制器的时钟
  - 在D1Run 模式下，TCM 和 AXISRAM 时钟总是 ON
  - 在D2 Run 模式下， D2 SRAMs 时钟 总是ON
- 当 Cortex-M4 从D2SRAMs或者D3SRAM运行时:
  - D1 域总是切换到 Power-down 模式
  - 中断向量需要重新映射到SRAM上!

CPU 操作模式

Mode	Description
CRun	CPU Active ➔ CPU, CPU-sub system bus matrix(s), CPU enabled peripheral clock(s) active.
CSleep	CPU Sleep ➔ CPU clock stopped, CPU-sub system bus matrix(s), CPU sleep enabled peripheral clock(s) active.
CStop	CPU Deepsleep ➔ CPU, CPU-sub system bus matrix(s), CPU peripheral clock(s) stopped.

- CPU 的操作模式直接由CPU控制
  - 进入 low power 模式
    - CSleep is entered by executing WFI/WFE or on return from ISR. (DEEPSLEEP = 0)
    - CStop is entered by executing WFI/WFE or on return from ISR. (DEEPSLEEP = 1)
  - 退出 low power 模式
    - When low power mode was entered by WFI or return from ISR
      - on any CPU interrupt with sufficient priority
    - When low power mode was entered by WFE
      - on CPU event
      - SVONPEND = 0 ➔ on interrupts with sufficient priority
      - SVONPEND = 1 ➔ on any interrupt

域的操作模式

Mode	Description
DRun	Domain bus matrix, clock active.
DStop	Domain bus matrix, Domain peripheral clock(s) stopped.
DStandby	Domain powered down.

- D1 & D2 域的操作模式由CPU分配不同域的外设来决定。
  - The domain is in:
    - DRun when a CPU with allocated domain peripherals is in CRun or CSleep
    - DStop when the CPU(s) with allocated domain peripherals is(are) in CStop and at least one domain PDDS_Dn bit select stop
    - DStandby when the CPU(s) with allocated domain peripherals is(are) in CStop and all domain PDDS_Dn bits select standby

系统的操作模式

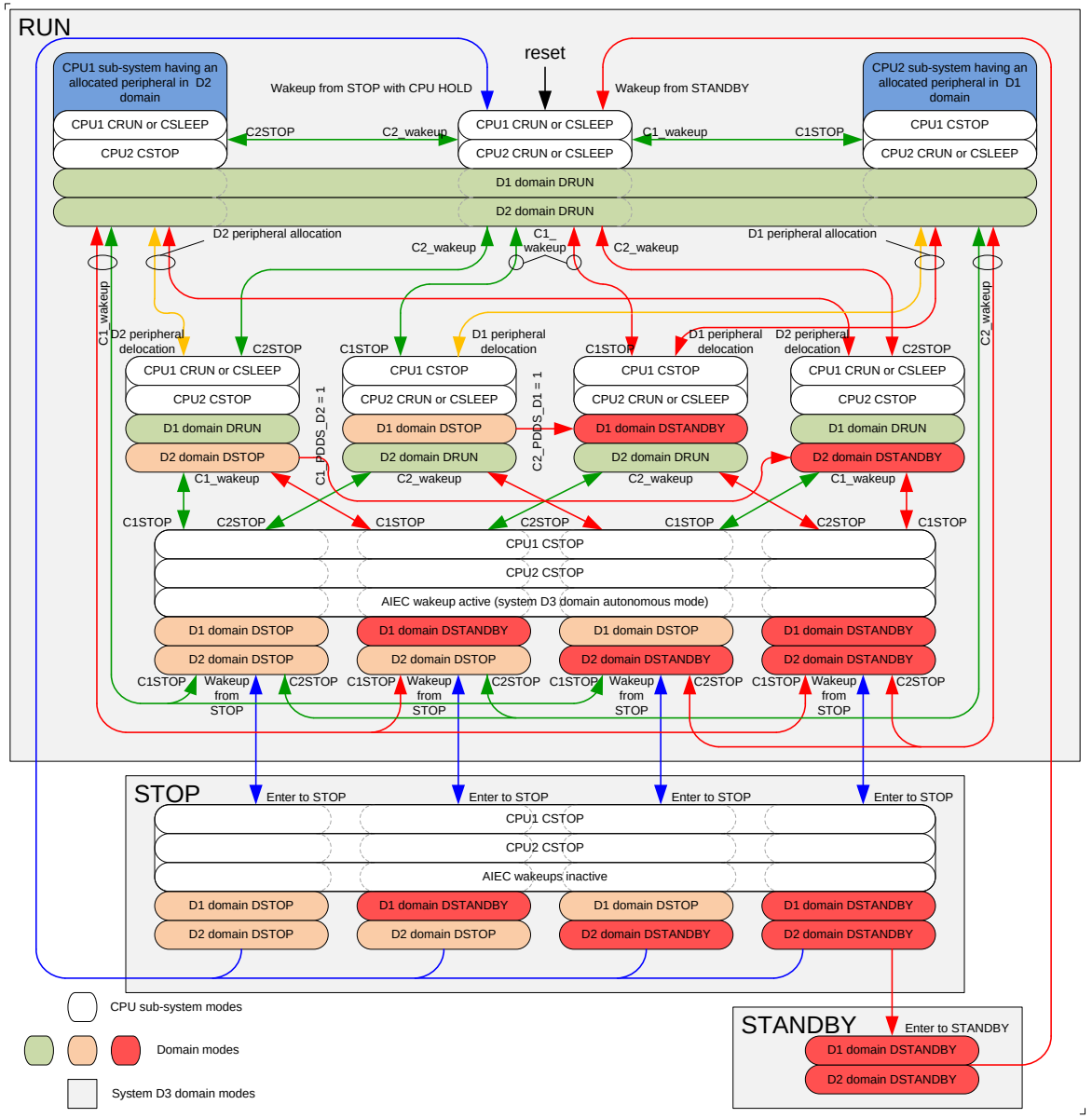
Mode	Description
Run	System clock is active and forwarded to the system.
Stop	System clock is stopped.
Standby	System is powered down. (Backup domain may be kept active)

- 系统的操作模式由CPUs 和wakeup 源来决定
  - Run when a CPU is in CRun or CSleep or a wakeup source is active.
  - Stop when the CPUs are in CStop and all wakeup sources are cleared and D3 domain is not forced in Run mode, and at least one PDDS_Dn bit select stop.
  - Standby when the CPUs are in CStop and all wakeup sources are cleared and D3 domain is not forced in Run mode, and all PDDS_Dn bits select standby.
- 系统支持D3 在自主运行模式下:
  - A peripheral in the D3 domain need to be enabled for Autonomous mode.
  - A CPU forces the D3 domain to stay in Run mode via its RUN_D3 register bit

系统功耗的控制状态

- 系统的功耗状态由 CPUs 和D3 域共同控制。

- 状态之间的转换需要由以下情况触发：
  - a CPU going to low power mode.
    - CPU, Domain and System modes
  - a peripheral allocation / de-location
    - Domain mode
  - a wakeup event
    - CPU, Domain and System modes
  - changing PDDS setting to STANDBY
    - Domain mode

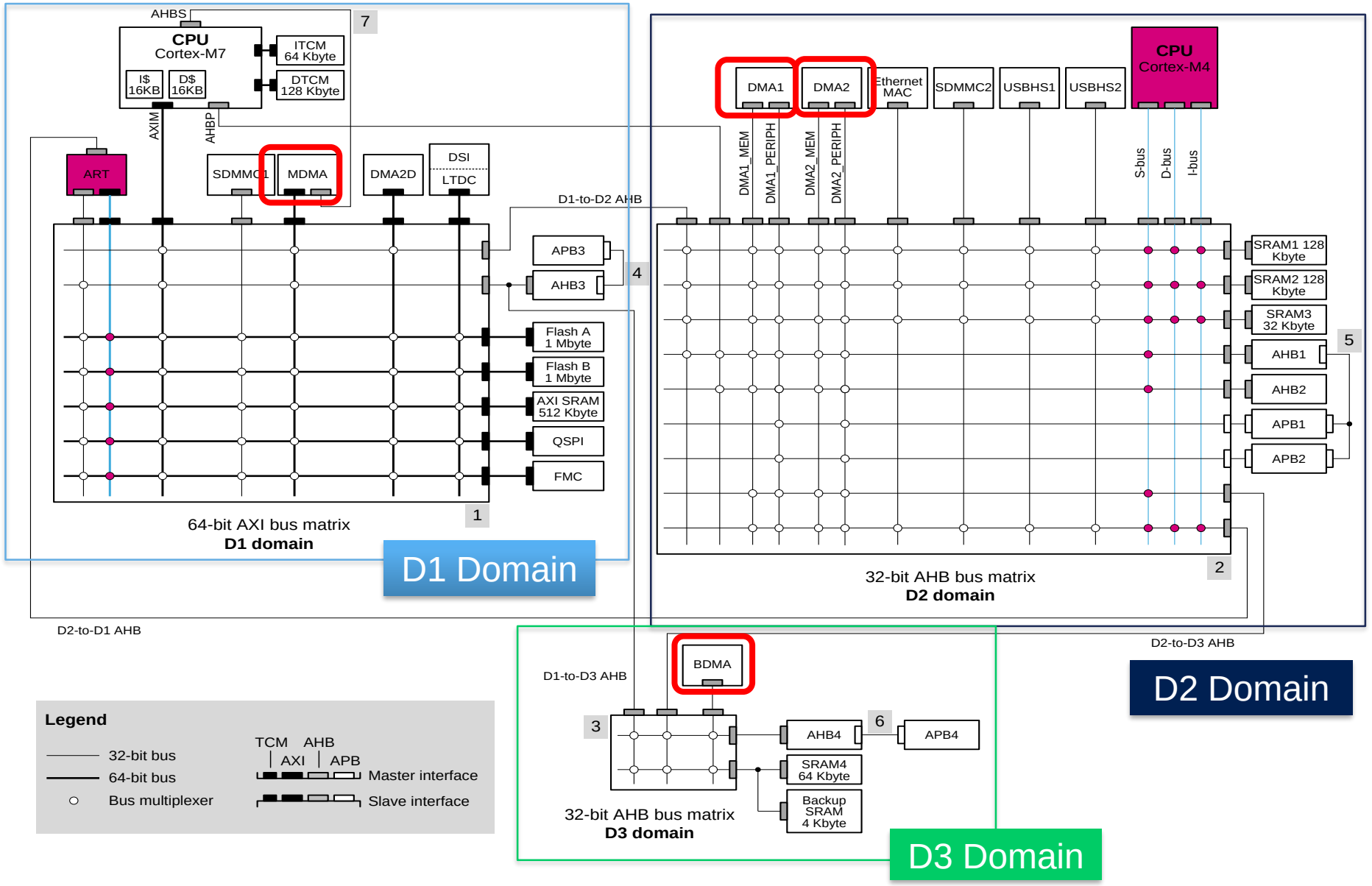


life.augmented

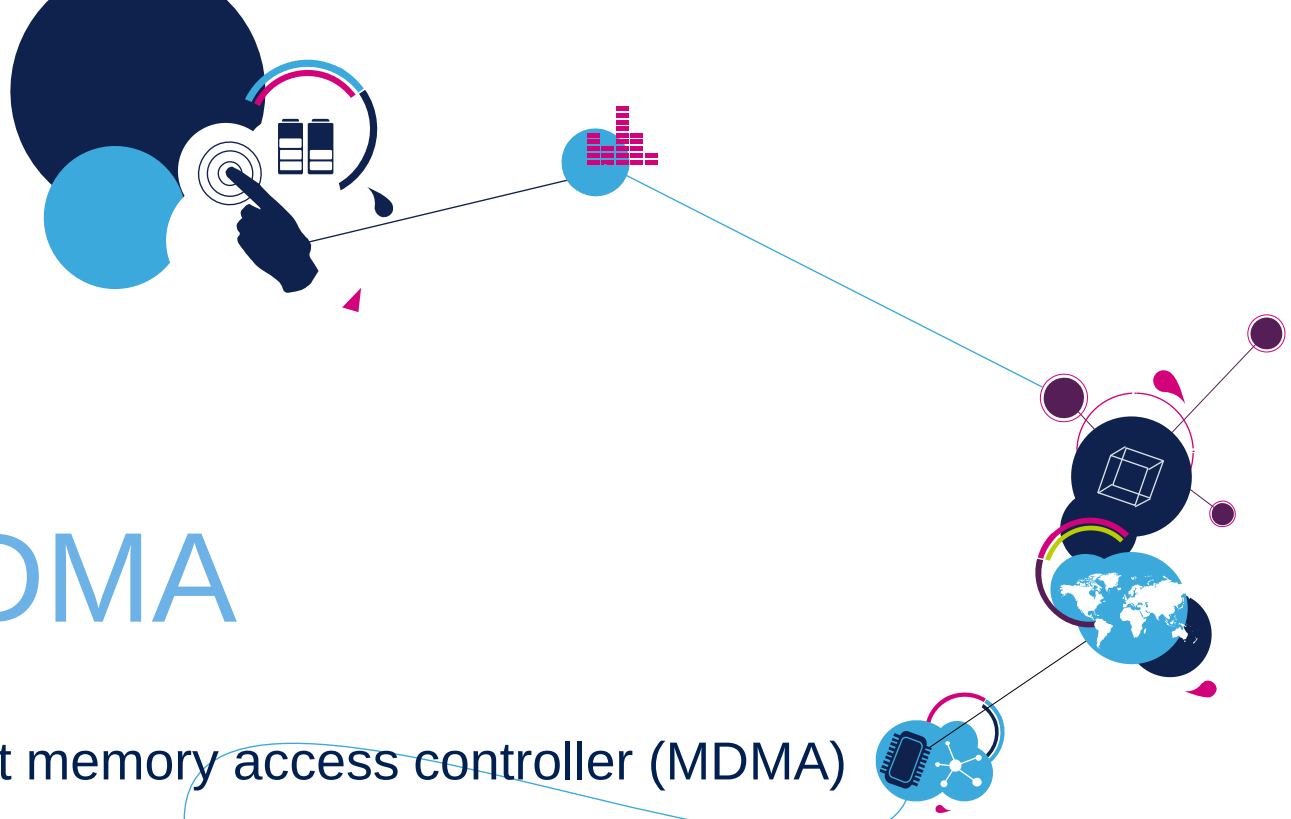
内容安排

- STM32H7 系统架构
  - 单核及双核STM32H7系统框图
  - AXI 总线及互联
  - ART 加速器
- STM32H7 存储器 结构
- STM32H7 Cache
- STM32H7 MPU
- STM32H7 启动方式
- *STM32H7-CPUs 概述*
  - *STM32H7 性能*
- STM32H7 双核优势和应用
- STM32H7 双核的同步和通信
  - IPC
  - *HSEM → 动手实验*
- STM32H7 电源管理
- **STM32H7外设**
  - **DMA**
  - 16位 ADC
  - LTDC MIPI 控制器
  - DLYB
  - FMC
- STM32H7目前的资源

DMA



life.augmented

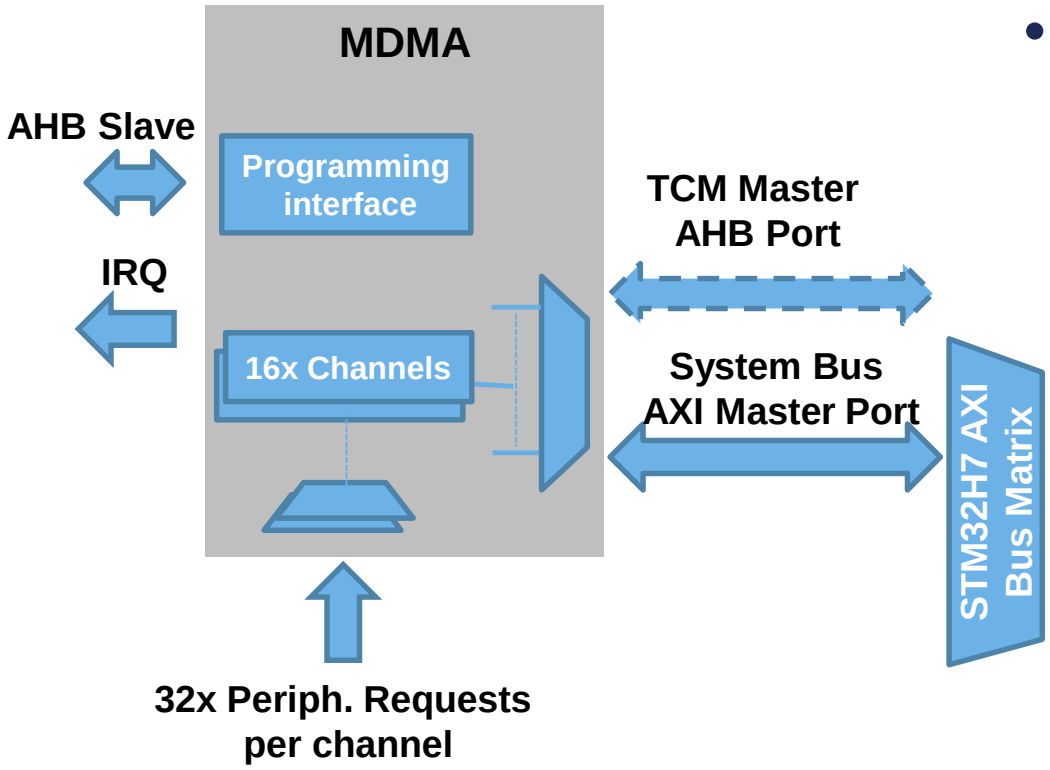


STM32H7 - MDMA

主直接内存存取控制器 Master direct memory access controller (MDMA)

版本 1

life.augmented



- STM32H7 MDMA 特征
  - 双 master 总线
    - 64-bit AXI and 32-bit AHB master port
  - 灵活的和独立的配置
  - 硬件和软件优先级管理
  - 可配置的数据传输模式
    - Peripheral-to-Memory, Memory-to-Peripheral, and Memory-to-Memory 模式

应用益处

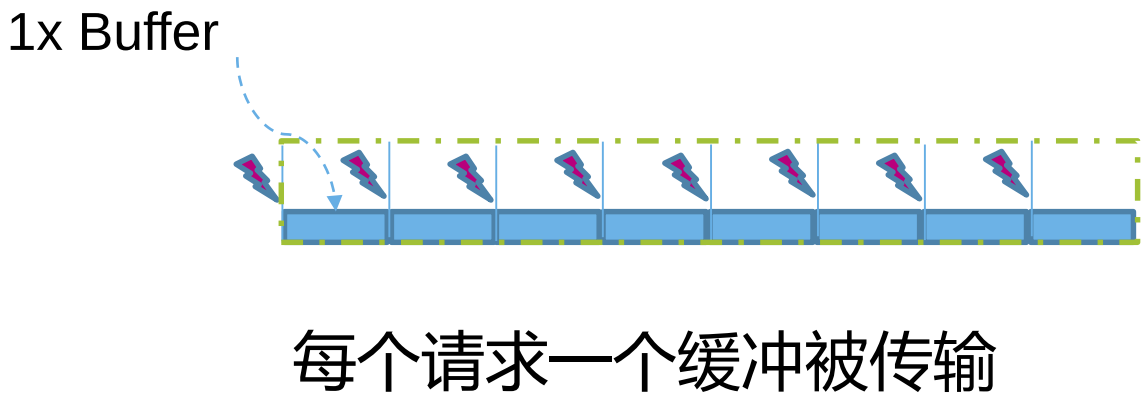
- DMA 支持块传输和链表
- 减少CPU 在数据传输管理中的负载
- 简单的集成

传输类型

- MDMA 支持单一或增量突发传输,
- 软件可配置的突发大小, 多达128 bytes
- 最大突发数据大小128 bytes. 对于数据量较大的突发长度限制在128级FIFO，将用于存储临时需要传送的数据
  - e.g. 16x64-bit or 32x32-bit
- 对于TCM内存访问，仅当增量和数据大小相同，且小于或等于32位时才允许突发访问.

MDMA 通道触发模式

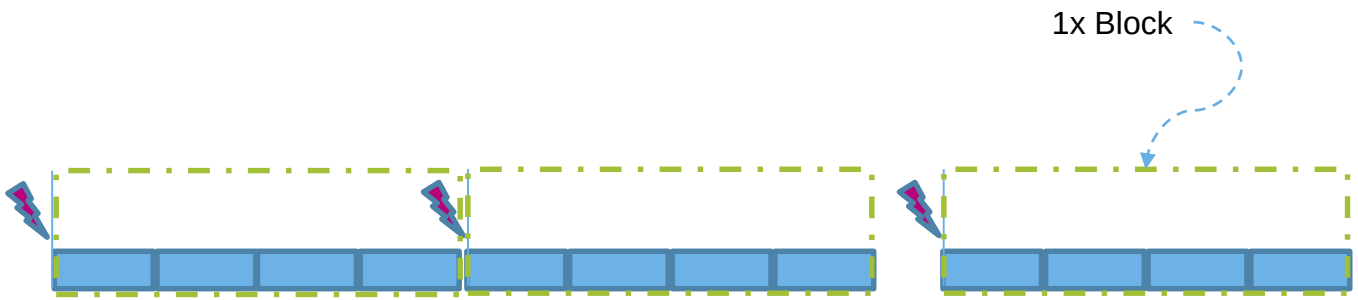
- 单个请求，数据的大小组成 传输，为下列中的一种
  1. 缓冲传输大小
     - TRGM=’00’



每个请求一个缓冲被传输

MDMA通道触发模式

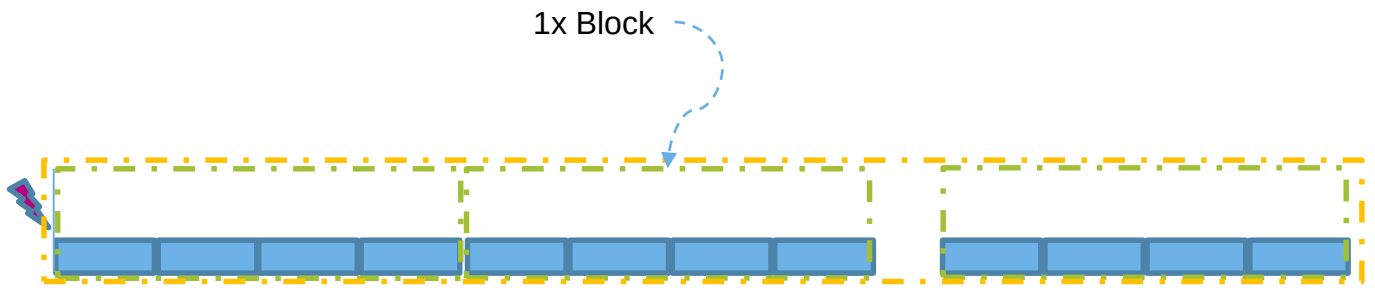
- 单个请求，数据的大小 组成 传输，为下列中的一种:
  1. 缓冲传输大小
     - TRGM=’00’
  2. 块的大小
     - TRGM=’01’



每次请求一个块被传输

MDMA通道触发模式

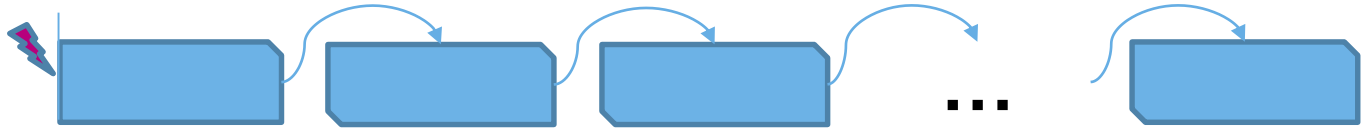
- 单个请求，数据的大小 组成 传输，为下列中的一种:
  1. 缓冲传输大小
     - TRGM=’00’
  2. 块的大小
     - TRGM=’01’
  3. 重复块
     - TRGM=’10’



每次请求几个块被传输

MDMA通道触发模式

- 单个请求，数据的大小 组成 传输，为下列中的一种:
  1. Buffer 传输大小
     - TRGM=’00’
  2. 块的大小
     - TRGM=’01’
  3. 重复块
     - TRGM=’10’
  4. 完成通道数据
     - TRGM=’11’



一个请求触发开始传输，直到通道链表指针为NULL

MDMA buffer 传输

- Buffer transfer 是在一个通道上，一个MDMA请求事件，传输的最小逻辑上传输的数据量。
  - 在当前通道上，需要传输的数据总量,是在 MDMA请求后由触发模式决定。
- 要传输的数据项的数目，它们的宽度（8位，16位，32位或64位）和用于数据传输的 burst 长度是软件可编程。
- 通过在给定掩码地址写掩码数据值，DMA /外设请求事件之后的服务请求被响应。
  - 主要用于清除请求事件源标志.

MDMA Block传输(1/2)

- 一个块是一个连续的数组,多达 64 Kbytes,由连续的 buffer transfers 传输完成
- 每个块的数据由 “start address” and “block length” 定义。
- 根据触发模型配置,一个整块传输可以由一个外设/ DMA请求触发。
  - 如果没有其他高优先级通道请求激活，将用相同的通道启动一个新的 buffer transfer

MDMA Block传输(2/2)

- 当一个块传输完成后,可以执行三个操作:
  - 块是一个重复的块传输的一部分: block length 重载和new block start address 的计算基于块重复地址更新寄存器中的信息。
  - 它是一个单一块或是重复的块传输的最后一块:下一个块信息从内存加载
  - 当前MDMA通道，最后一个块传输完成：这个 通道是禁用的 并且 没有进一步的 MDMA请求将被接受。

MDMA Block 重复模式

- 块重复模式允许 repeat a block transfer,用不同的源和目标的 “start addresses” 。
- 当重复块模式处于活动状态(重复计数非0),在当前的块传输完成，块的参数将被更新。
  - 重新装入数据字节的块数目到传输值 (BNDT)
  - 更新块的源和目的地址值，根据块的重复源 / 目的地址更新配置(BRSUM/BRDUM)
  - 重复计数器减 1
- 当重复块计数器达到0,最后一块将被视为一个单独的块传输.

MDMA Linked 链表模式

- 链接列表模式允许从 Channel Link Address Register (CLAR) 所给的地址， 加载新的 MDMA configuration
- 此操作后,通道准备接受新的请求,如前面定义的块/重复块模式,或继续传输，如果触发模式设置为 Complete channel data (TRGM=11)
- 当加载触发和总线选择寄存器(TBR)值时，自动触发源可能会改变
- 如果触发模式设置为TRGM=11 :触发模型和选用SW请求不能被改变

MDMA Linked链表模式

- 通道配置(Channel link address LAR)必须在AXI地址空间
- LAR value 必须用 Double Word address 对齐, i.e. LAR[2:0] = 0x0

Register (32-bit word)	Offset from Link Address register	Description
CTCR	0x00	Transfer Configuration register
CBNDTR	0x04	Block number of data register
CSAR	0x08	Source address register
CDAR	0x0C	Destination address register
CBRUR	0x10	Block Repeat address Update register
CLAR	0x14	**Link Address register:** Next descriptor
CTBR	0x18	Trigger and Bus selection Register
CMAR	0x1C	Mask address register
CMDR	0x20	Mask Data register

MDMA 用法实例

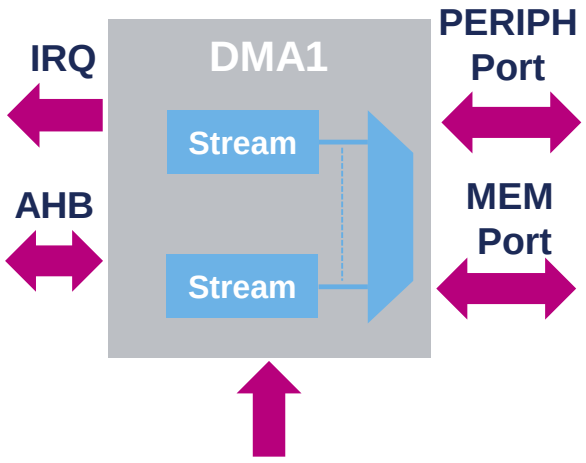
- 从其他 (DMA1 / DMA2和BDMA)收集数据和使数据在D1域的CPU可用(DTCM或AXI-SRAM)， MDMA 是有用的

- DMA链表，执行一组DMA传输，且不需要CPU干预
  - 可以用来准备数据给其它的 DMAs ， 然后设置DMA配置开始传输。
  - 它是用来支持分散/聚合。这意味着源和目标区域不需要占用连续的内存区域。源和目标数据区域是由一系列的链表描述符控制数据块的转移。

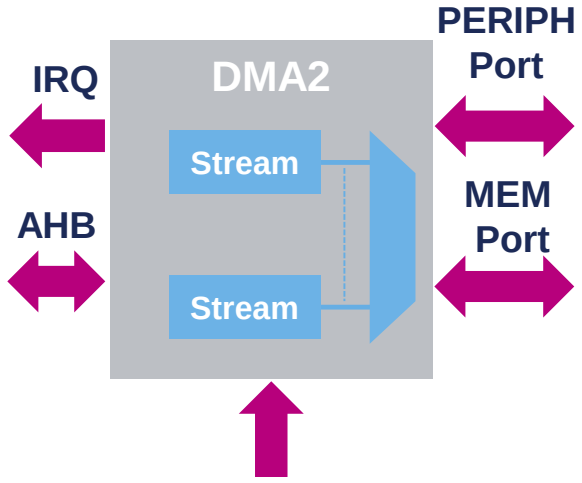
STM32H7 - DMA

直接内存存取控制器(DMA)

life.augmented



Up to 105 request per stream



Up to 105 request per stream

- DMA1/2 特征
  - 双 AHB master 总线
  - 灵活配置
  - 硬件软件优先级管理
  - 配置数据传输模式
    - Peripheral-to-Memory, Memory-to-Peripheral, and Memory-to-Memory modes

应用好处

- DMA 支持D2 域的 timers, ADC, 和 通讯外设
- 把CPU从数据传输管理中卸载
- 简单的集成

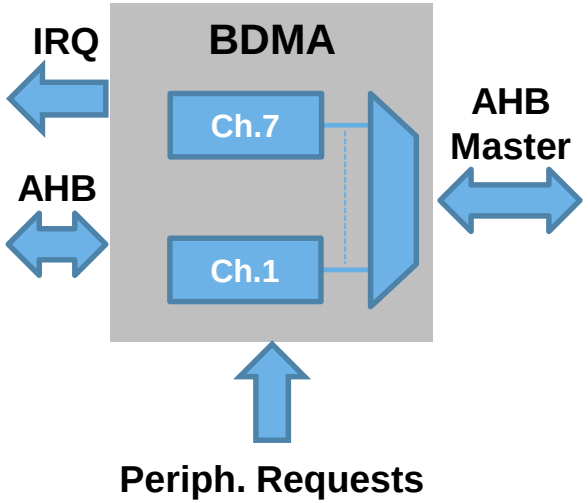
相关外设

- 参考和这个外设有关的这些外设培训:
  - DMAMUX (DMA request router)
  - MDMA (Master direct memory access controller)

STM32H7 - BDMA

基本的直接内存访问控制器 (BDMA)

life.augmented



- STM32H7 BDMA 特征
  - 灵活的配置
  - 硬件软件优先级管理
  - 可配置的数据传输模式
    - Peripheral-to-Memory, Memory-to-Peripheral, Peripheral-to-Peripheral, and Memory-to-Memory modes

应用的好处

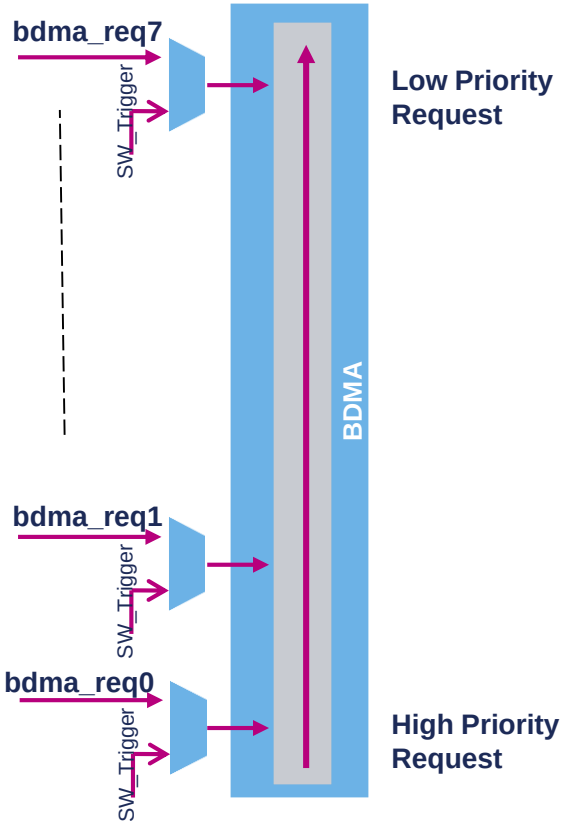
- DMA 支持 D3 域的外设
- 当 CPU 进入 CStop 并且 D1 或 D2 域处在 Dstandby 时，执行数据传输

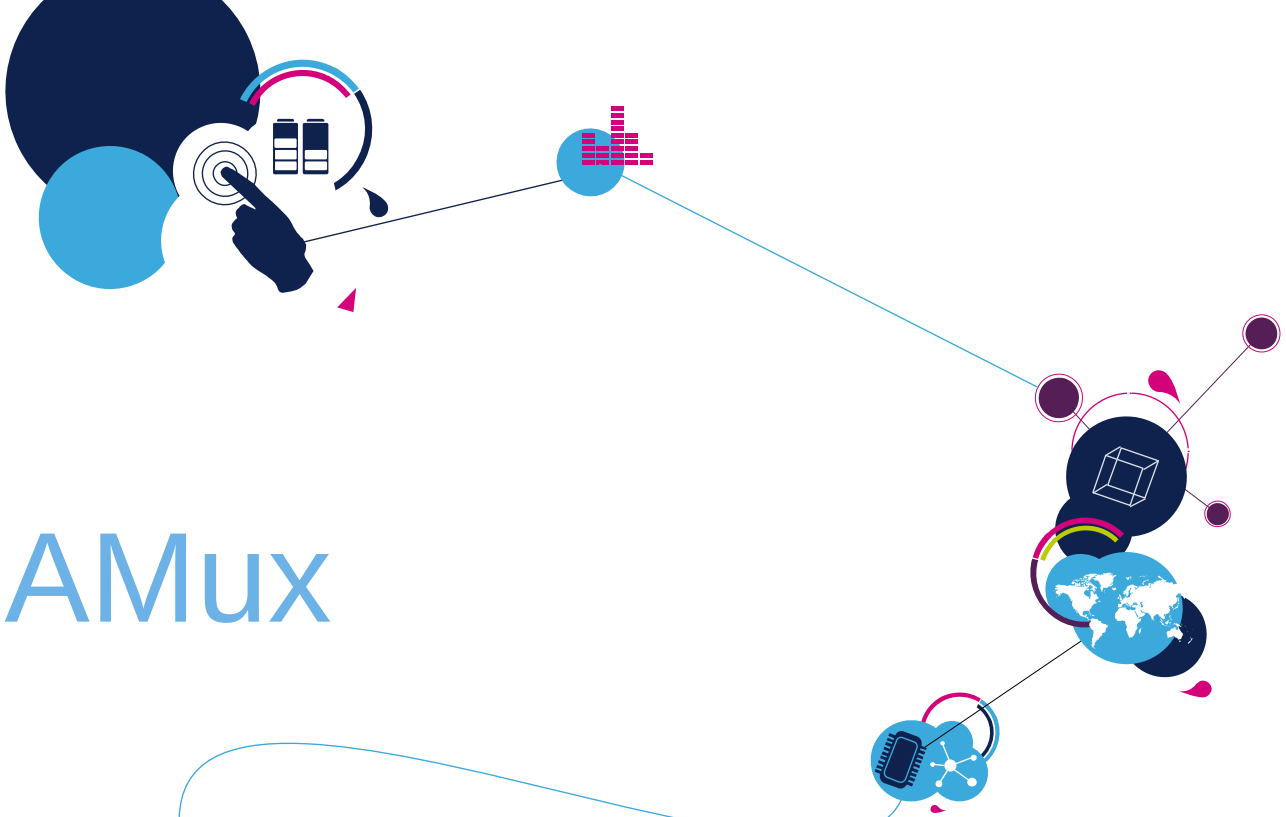
主要特征

- 7 个独立的可配置通道
  - 每个通道都可以 Hardware request or software trigger
  - 软件可编程的优先级与硬件优先级的平等
- 独立的和灵活的通道配置
  - 完全可编程的通道 (data format, increment type, address)
  - 独立的通道中断标志 (half transfer, transfer complete, transfer error, global flags)
  - 支持 circular buffer 管理.
- 在出现总线访问错误的情况下，错误通道自动禁止

BDMA requests 映射

- BDMA1 控制器提供 7 channels 的访问
  - **New**: 通过一个多路复用器来映射外设请求(Not OR gate)
  - 每个通道有独立软件触发



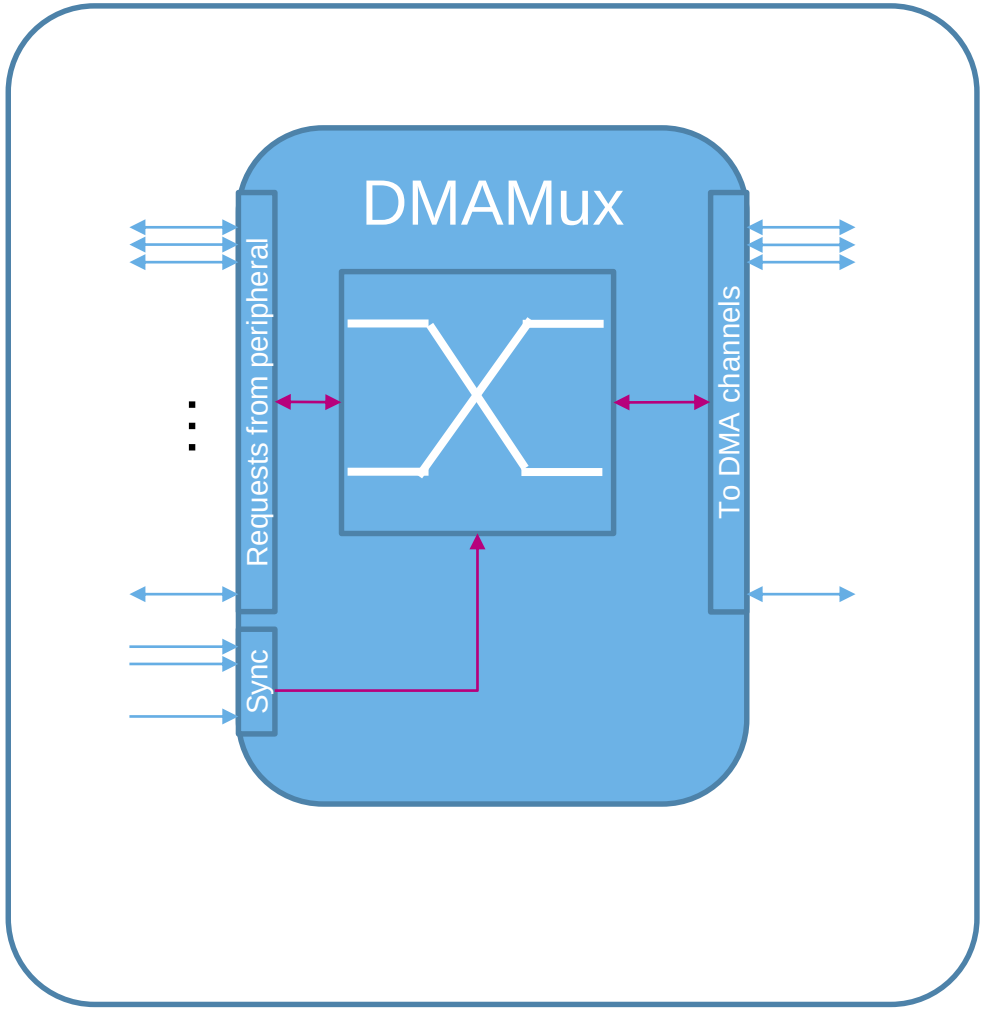


STM32H7- DMAMux

DMA 请求路由

life.augmented

概述



- DMA请求路由 (DMAMux) 管理:
  - 分配DMA请求线到外设
  - 在同步输入时，请求转发与事件同步
  - 请求链接使用DMA请求计数器，并且事件生成给DMA

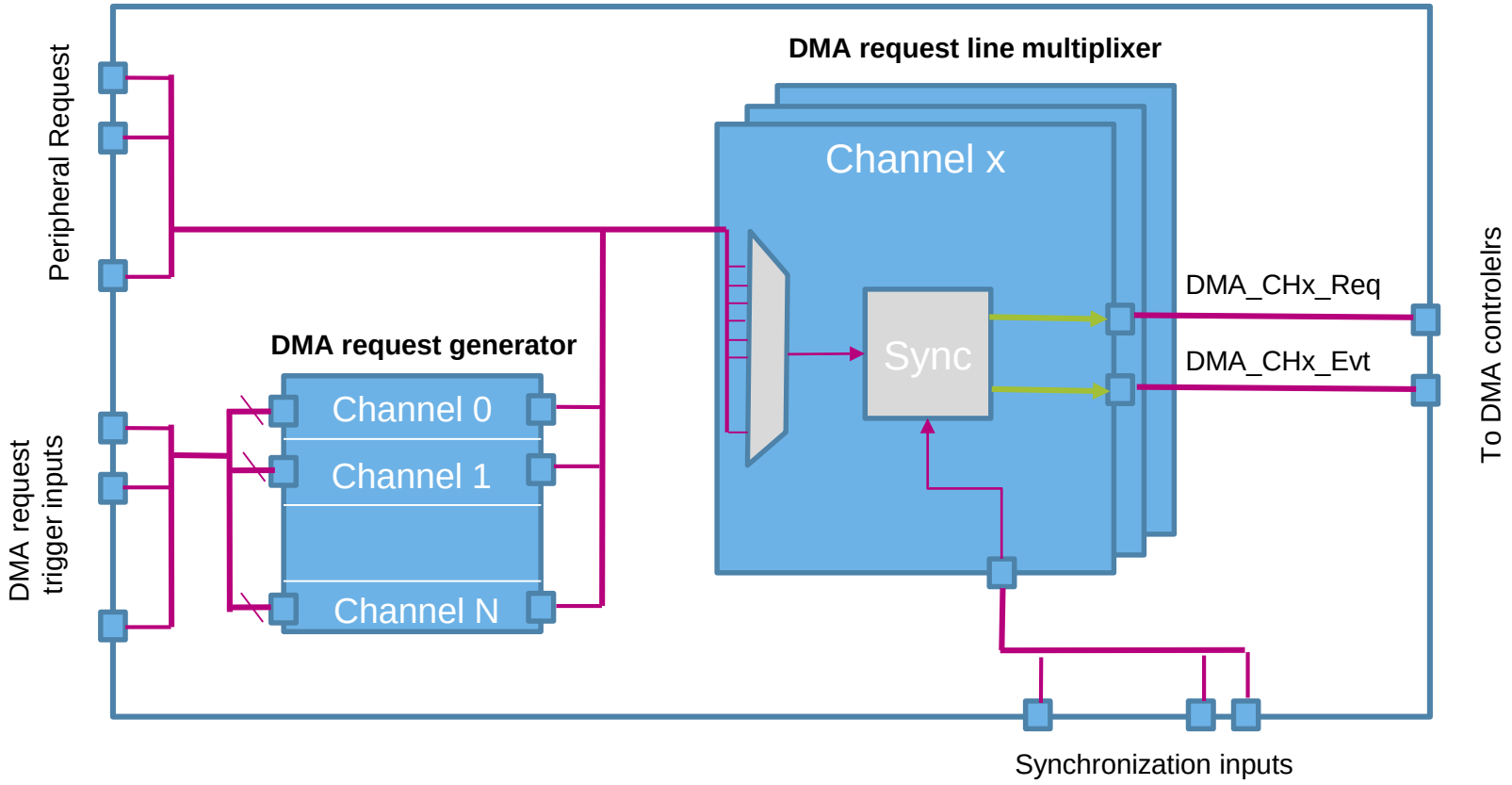
应用好处

- DMA请求映射的选择有高灵活性
- 外部和内部的DMA请求管理

主要特征

- DMA 请求线多路复用器
  - 灵活的映射外设请求到 DMA controller channel
  - 多达 107 个输入 DMA request lines
  - 同步操作模式,有单独的 enable bit
  - 多达16 个同步输入选择器
  - 选定同步输入时，有事件 overrun flag
- DMA 请求生成通道:
  - DMA 请求生成，在异步事件时
  - 多达 32 个 DMA 请求触发输入选择
  - Event overrun flag， 给选定的DMA请求触发输入

DMAMux 框图



内容安排

- STM32H7 系统架构
  - 单核及双核STM32H7系统框图
  - AXI 总线及互联
  - ART 加速器
- STM32H7 存储器 结构
- STM32H7 Cache
- STM32H7 MPU
- STM32H7 启动方式
- *STM32H7-CPUs 概述*
  - *STM32H7 性能*
- STM32H7 双核优势和应用
- STM32H7 双核的同步和通信
  - IPC
  - *HSEM → 动手实验*
- STM32H7 电源管理
- **STM32H7外设**
  - DMA
  - **16位 ADC**
  - LTDC MIPI 控制器
  - DLYB
  - FMC
- STM32H7目前的资源

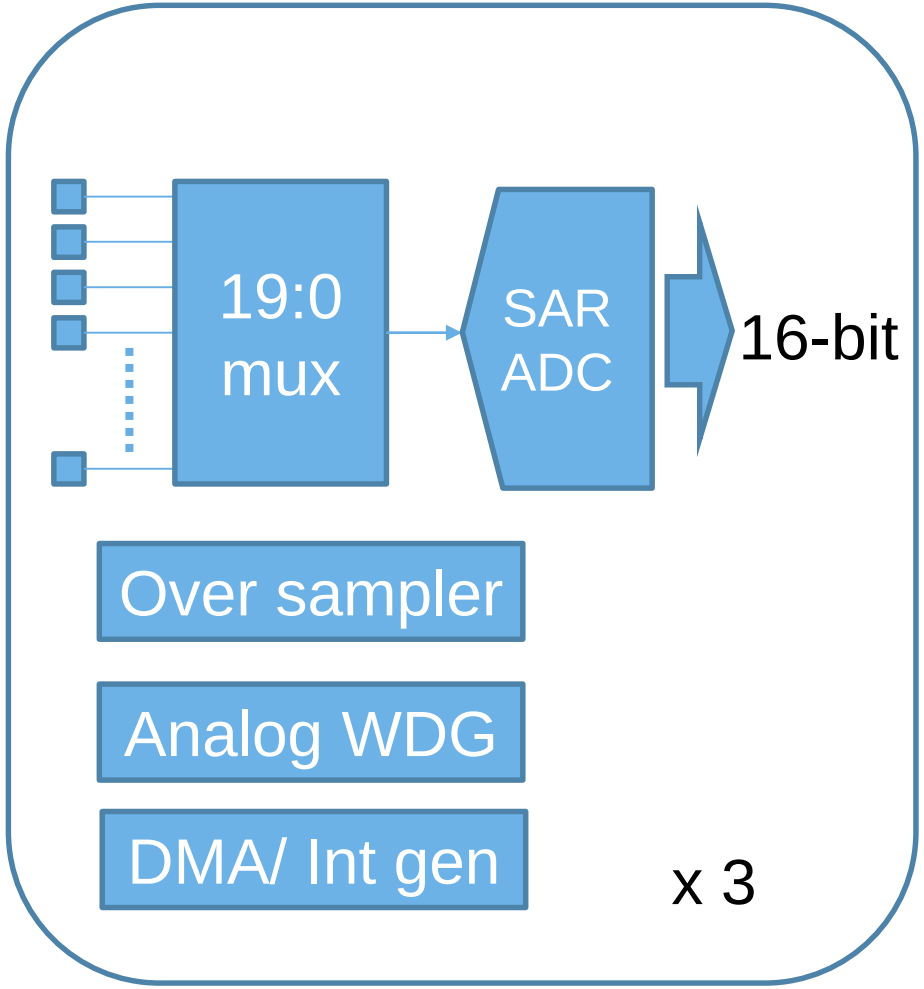
STM32H7 – ADC

Analog-to-Digital Converter

Revision 1

life.augmented

概述



- 提供模拟到数字的转换
  - 三个 ADCs 高达24个输入通道
  - 16位结构,高达21-bit过采样
  - 8 Msamples/s max. (14-bit)
  - 每个ADC有3个模拟看门狗。
  - DMA 请求生成
  - 中断生成

应用好处

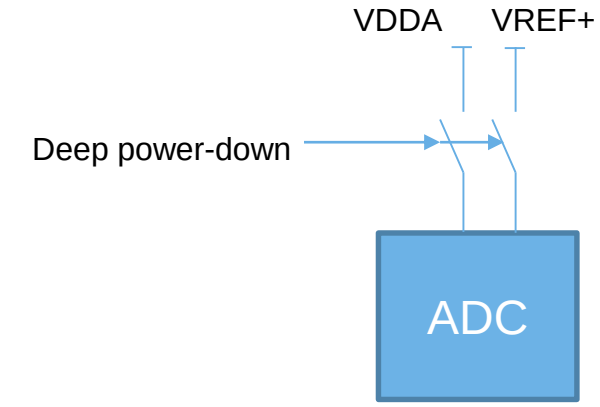
- 超低功耗: 210 µA @ 1 Msample/s
- 灵活的触发，数据管理，减轻CPU的负载

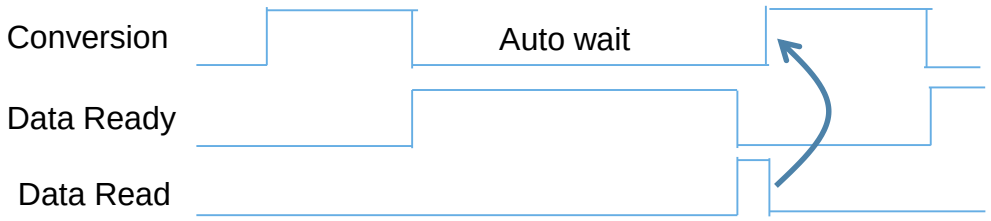
关健特色

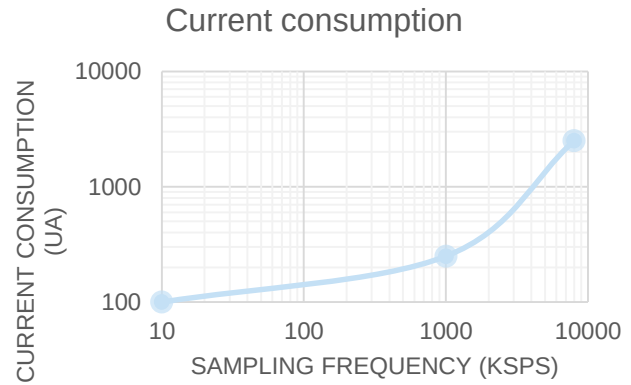
ADC 单元	3 模块
输入通道	高达24个外部通道(GPIOs)、单端/差分
技术	16-bit 逐次逼近
转换时间	125nS, 8 Msamples/s (when f_{ADC_CLK} = 72 MHz, 14bit)
功能模式	单次、连续扫描、不连续或注入
触发器	软件或外部触发(for Timers & IOs)
特殊功能	硬件过采样,模拟看门狗
数据处理	中断生成、DMA请求
低功耗模式	深度掉电,自动延迟、功耗取决于速度

几种低功耗功能实现

- 深度掉电模式
  - 内部供电ADC可以通过电源开关禁用,减少泄漏
- 自动延时转换
  - ADC可以自动等直到最后数据读取
- 功耗取决于采样时间
  - 2.5mA @ 8 Msamples/s,
  - 210 µA @ 1 Msample/s,
  - 100 µA @ 10 ksamples/s

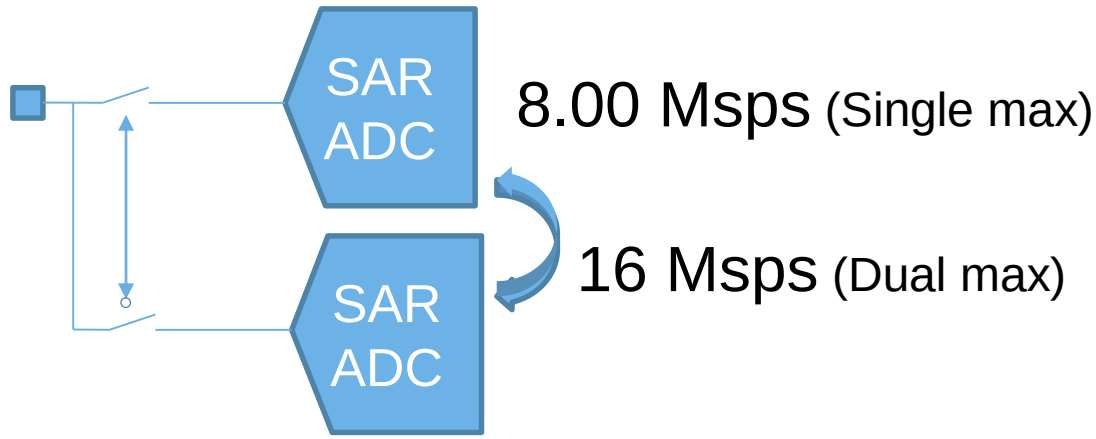


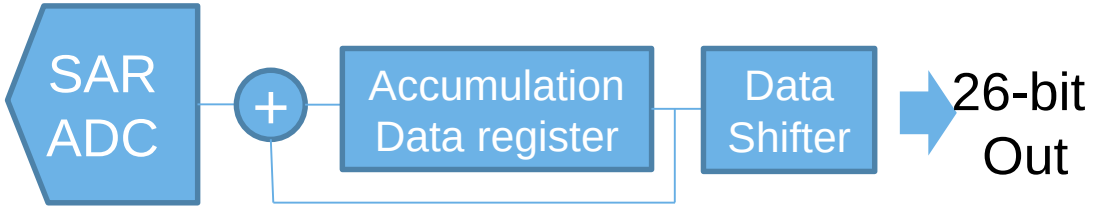


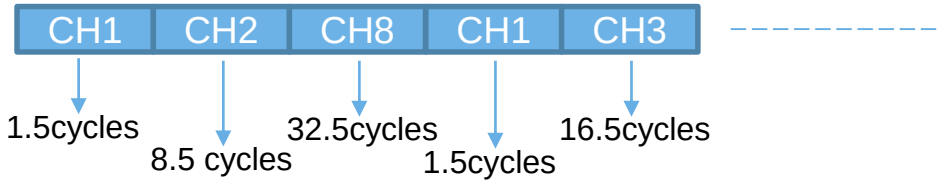


一些高性能功能实现

- 8.00 Msamples/s for 72 MHz ADC clock@14bit
- 交错模式可以支持 16 Msamples/s
- 硬件过采样
  - 累加器和位移器能输出26-bit数据，不需要CPU的支持
- 灵活的定序器
- 自动校准减少offset,更好的线性







ADC 转换速度

转换速度和精度相关

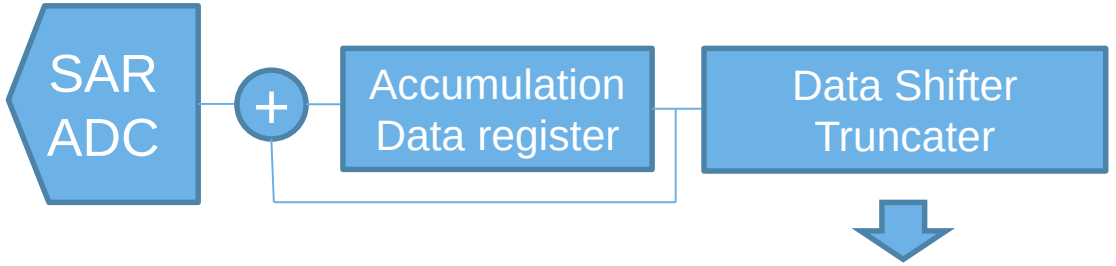
- ADC 需要最小 1.5$_{ADC_CLKs}$ 采样周期 和 7.5$_{ADC_CLKs}$ 转换(14 bits).
- 最大72 MHz时钟与9 cycles 的结果是 8 Msamples / s
- 更低精度，更高速度
  - 16-bit : 8.5$_{ADC_CLKs}$(+1.5) => 7.2 Msamples/s
  - 12-bit : 6.5$_{ADC_CLKs}$(+1.5) => 9 Msamples/s
  - 10-bit : 5.5$_{ADC_CLKs}$ (+1.5) => 10.2 Msamples/s
  - 8-bit : 4.5$_{ADC_CLKs}$ (+1.5) => 12 Msamples/s

Resolution	$t_{Conversion}$
16 bits	8.5 Cycles
14 bits	7.5 Cycles
12 bits	6.5 Cycles
10 bits	5.5 Cycles
8 bits	4.5 Cycles

硬件过采样

数据预处理减轻CPU的负载

- 可编程的过采样率:
  x2 to x1024
- 可编程数据移位和截断 Left shift of 0 to 15 bits, right shift of 0 to 11 bits.
- 高达32-bit 数据宽度
- 平均,降低数据率,信噪比提高
- 基本过滤



Oversampling ratio	Output resolution	Equivalent sampling frequency max
x1(none)	16 bits	7.2Msps
x16	18 bits	450 ksamples/s
x256	20 bits	28.1 ksamples/s
x1024	21 bits	7.0 ksamples/s

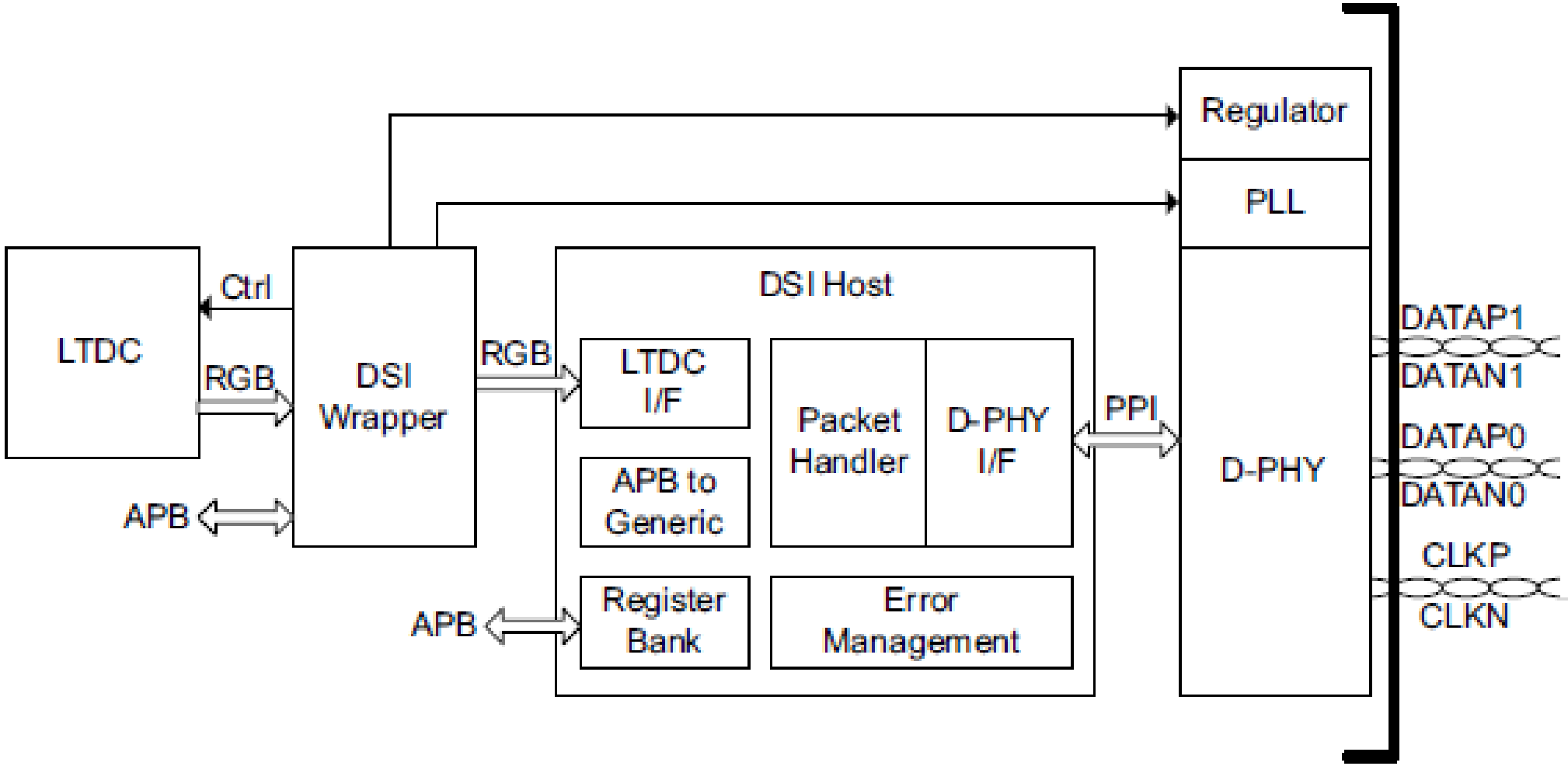
内容安排

- STM32H7 系统架构
  - 单核及双核STM32H7系统框图
  - AXI 总线及互联
  - ART 加速器
- STM32H7 存储器 结构
- STM32H7 Cache
- STM32H7 MPU
- STM32H7 启动方式
- *STM32H7-CPUs 概述*
  - *STM32H7 性能*
- STM32H7 双核优势和应用
- STM32H7 双核的同步和通信
  - IPC
  - *HSEM → 动手实验*
- STM32H7 电源管理
- **STM32H7外设**
  - DMA
  - 16位 ADC
  - **LTDC MIPI 控制器**
  - DLYB
  - FMC
- STM32H7目前的资源

STM32H7x7 Graphic line
DSI Host

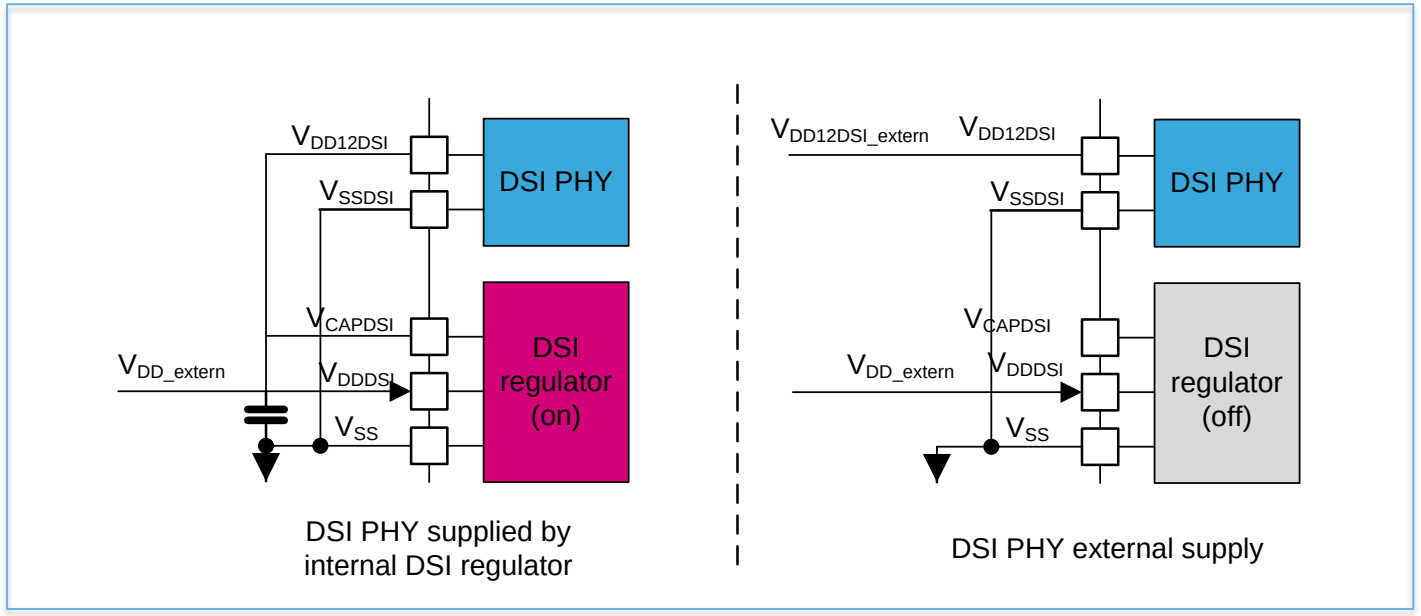
life.augmented

Display Serial Interface (DSI)



DSI 电压调节器

- DSI 接口通过专门的 VDD12DSI来供电。供电方式有两种，一种是内部集成的 DSI 电压调节器，一种是外部的DSI供电。



DSI 电压调节器

- 如果DSI 不适用的话：
  - VDDDSI 引脚或者连接到 VDD，或者 floating.
  - VCAPDSI 引脚必须连接到外部的 VDD12DSI，但是不需要外部的电容。
  - VSSDSI 引脚必须接地。

内容安排

- STM32H7 系统架构
  - 单核及双核STM32H7系统框图
  - AXI 总线及互联
  - ART 加速器
- STM32H7 存储器 结构
- STM32H7 Cache
- STM32H7 MPU
- STM32H7 启动方式
- *STM32H7-CPUs 概述*
  - *STM32H7 性能*
- STM32H7 双核优势和应用
- STM32H7 双核的同步和通信
  - IPC
  - *HSEM → 动手实验*
- STM32H7 电源管理
- **STM32H7外设**
  - DMA
  - 16位 ADC
  - LTDC MIPI 控制器
  - **DLYB**
  - FMC
- STM32H7目前的资源

STM32H7- DLYB

延迟块

life.augmented

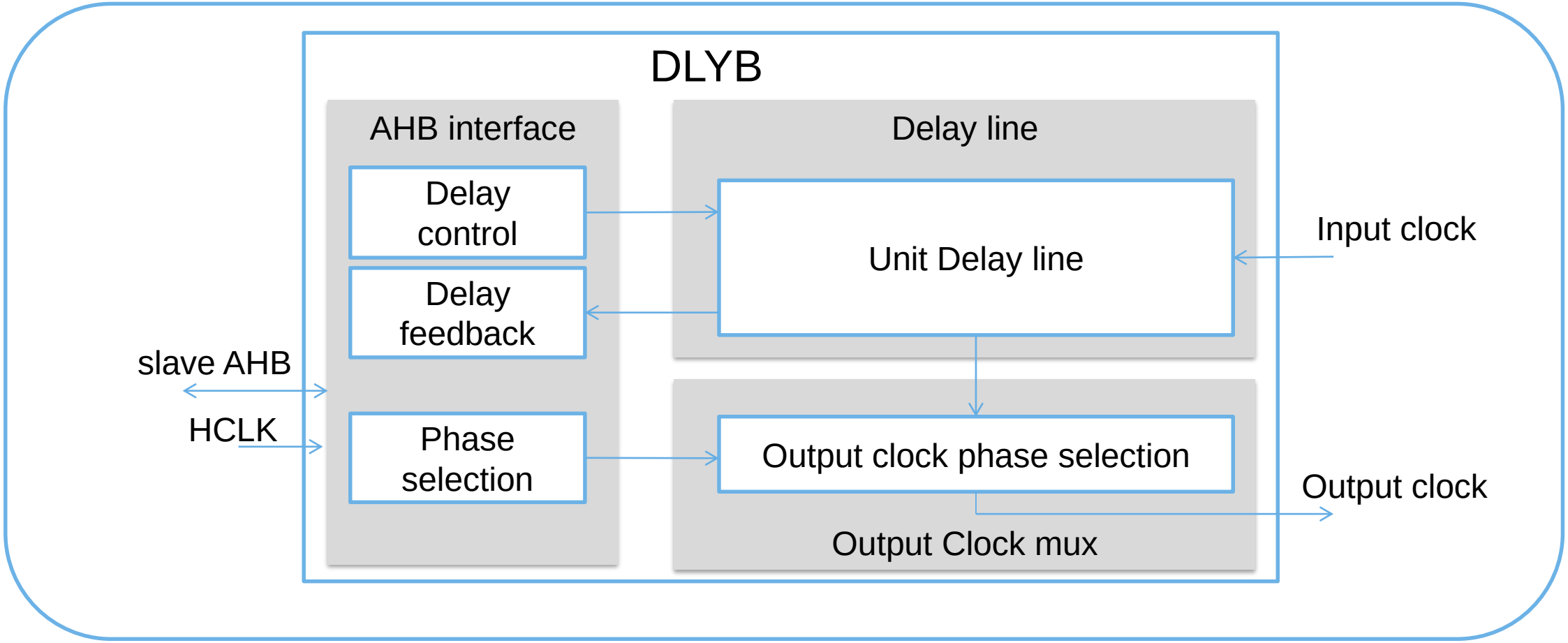
概述

- 为SDMMC和 QSPI 数据通信接口，产生一个延迟采样时钟
- 输入时钟频率范围 25 MHz to 208 MHz.
- 固件控制
- 没有电压和温度漂移补偿.

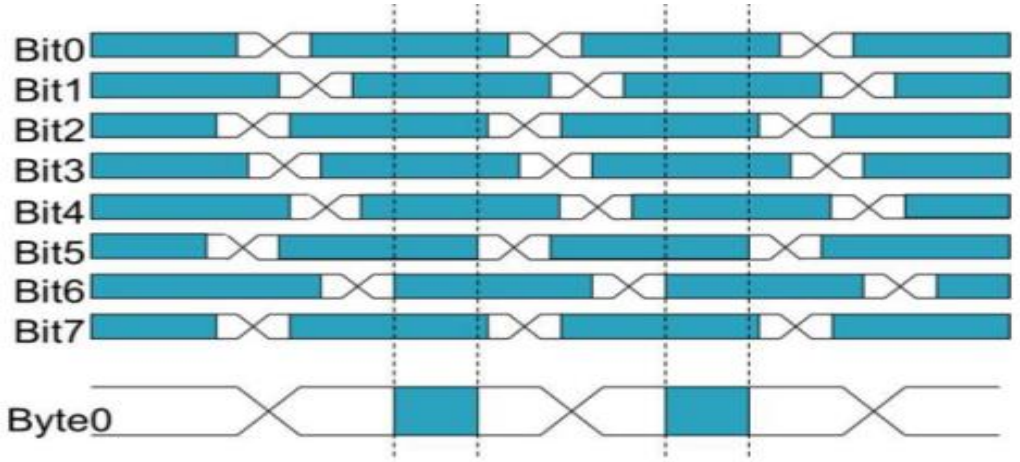
应用好处

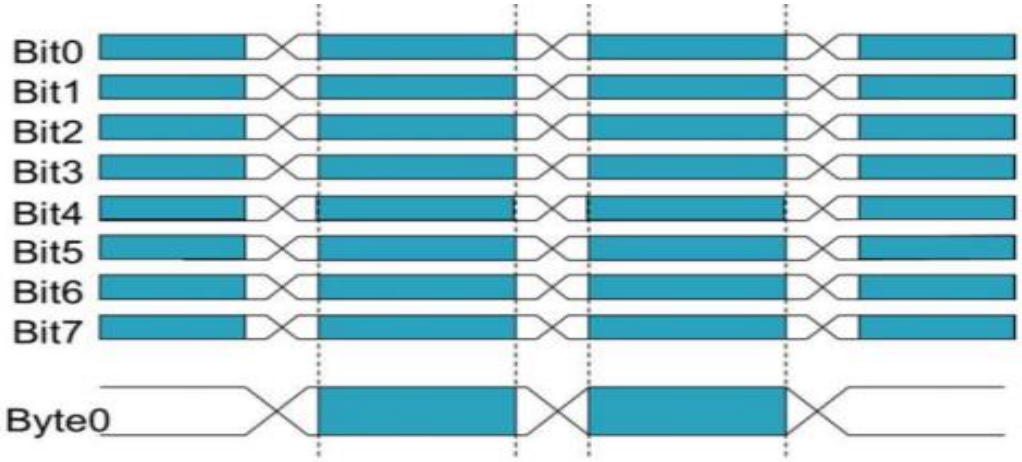
- 支持SDMMC卡可变延迟
- 提高SDMMC和QSPI的时序裕量

框图

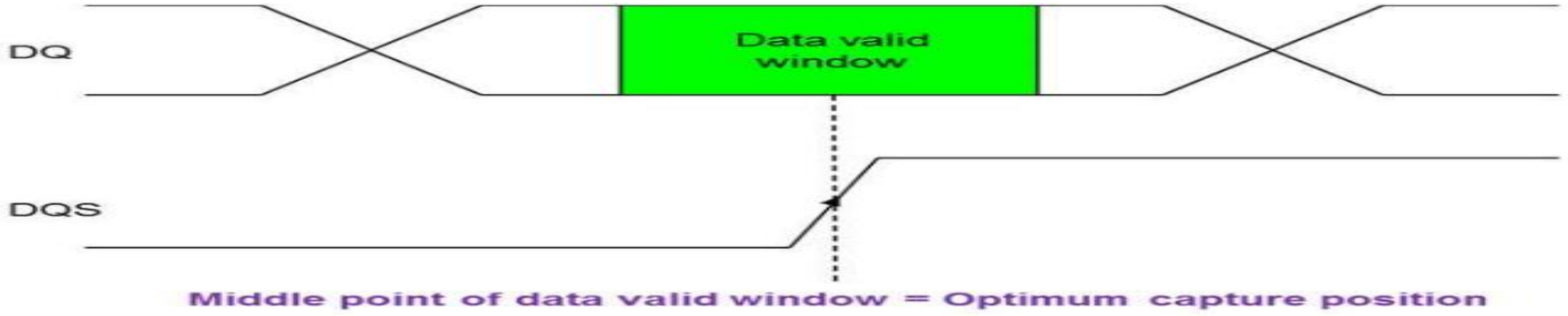

DLYB
AHB interface
Delay line
Delay control
Delay feedback
Unit Delay line
Input clock
slave AHB
HCLK
Phase selection
Output clock phase selection
Output clock
Output Clock mux

长度均衡





- 最终，同一组的信号必须匹配设置和持有时间



- 必须考虑整个信号通路:衬底,PCB & 走线的长度

内容安排

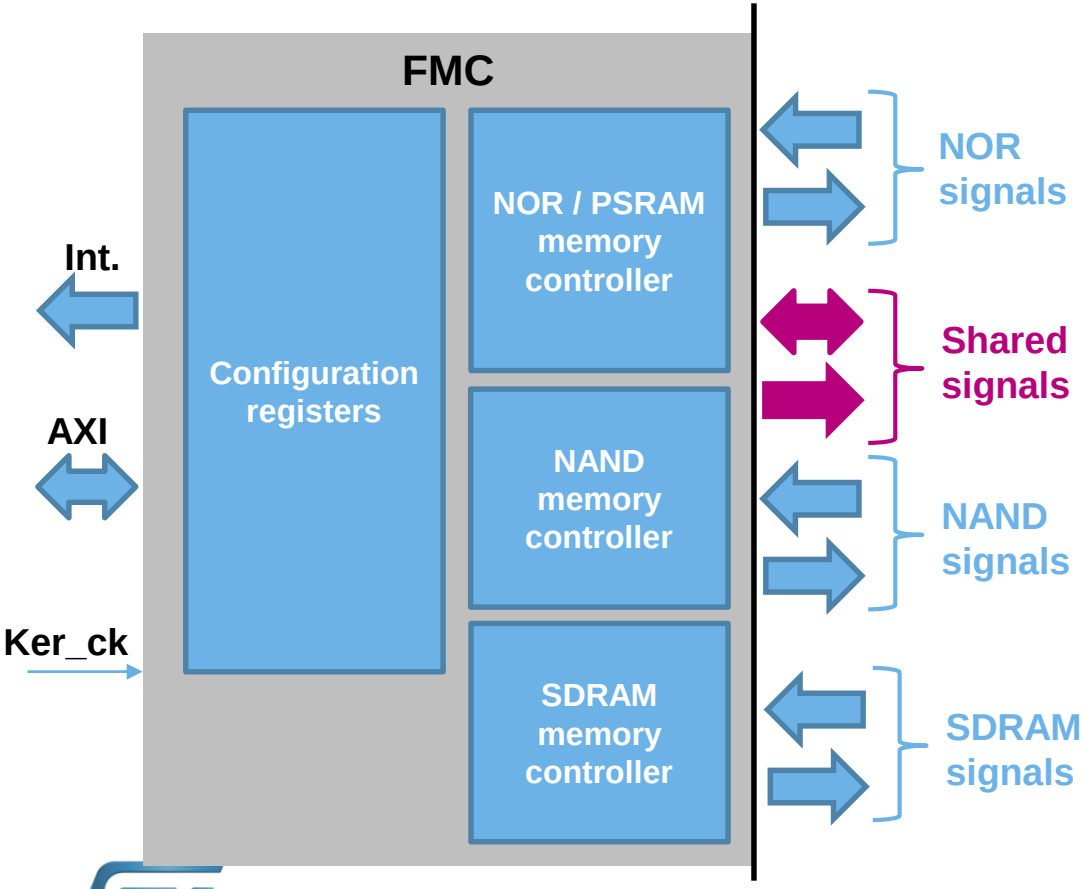
- STM32H7 系统架构
  - 单核及双核STM32H7系统框图
  - AXI 总线及互联
  - ART 加速器
- STM32H7 存储器 结构
- STM32H7 Cache
- STM32H7 MPU
- STM32H7 启动方式
- *STM32H7-CPUs 概述*
  - *STM32H7 性能*
- STM32H7 双核优势和应用
- STM32H7 双核的同步和通信
  - IPC
  - *HSEM → 动手实验*
- STM32H7 电源管理
- **STM32H7外设**
  - DMA
  - 16位 ADC
  - LTDC MIPI 控制器
  - DLYB
  - **FMC**
- STM32H7目前的资源

STM32H7 - FMC

灵活的存储控制器

life.augmented

概述



- FMC 支持外部内存通过下列方法
  - NOR Flash/PSRAM 控制器
  - NAND memory 控制器
  - SDRAM memory 控制器
  - 独立的内核时钟

应用好处

- RAM 扩展
- Flash memory 扩展
- 并行接口(Intel 8080 / Motorola 6800)

主要特点

- 完全独立 banks
  - 四个 banks 支持独立的外部 memories
  - 每个内存 bank 有独立的 Chip 选择
  - 每个内存 bank 有独立的 配置
- 配置灵活
  - FMC 外部访问频率高达 HCLK/2
  - 可编程的时序支持宽范围的设备
  - 8- ,16- or 32-bit 数据总线
  - 外部异步等待控制
  - 扩展模式（读时序和协议，不同于写时序）
  - 支持同步设备的突发模式访问(NOR Flash and PSRAM)

支持的设备

兼容各种各样的接口和存储器

- 静态 memory-mapped 设备包括
  - Static random access memory (SRAM)
  - Read-only memory (ROM)
  - NOR / OneNAND Flash memory
  - PSRAM
- NAND Flash memory
  - 包括 ECC 硬件检查高达 8 Kbytes of data read/written
  - 3 可能的中断源（level, rising edge and falling edge)
- SDRAM memory
  - 同步DRAM（SDRAM）接口内存映射

兼容各种各样的接口和**memories**

- 平行的液晶显示模块 (Parallel LCD modules)
  - Intel 8080 and Motorola 6800

低功耗模式

模式	说明
Run	Active.
Sleep	Active. Peripheral interrupts cause the device to exit Sleep mode.
(D1)Stop	Frozen. Peripheral registers content is kept.
(D1)Standby	Powered-down. The peripheral must be reinitialized after exiting domain and system Standby mode.

STM32H7 安全特性

life.augmented

Standard Flash protections

- Readout Device Protection (RDP)on Flash and Backup SRAM
  - Level 0:
    - No protection. Access to Flash and backup SRAM allowed whatever the boot configuration (Flash, RAM, Debug)
  - Level 1:
    - No access to Flash memory or backup SRAM can be performed while the debug feature is connected or while booting from RAM or system memory bootloader.
    - Option bytes can still be modified
    - Level regression is possible with flash mass erase.
  - Level 2:
    - All protections provided by Level 1 are active.
    - Booting from RAM or system memory bootloader is no more allowed.
    - JTAG, SWV (single-wire viewer), ETM, and boundary scan are disabled.
    - User option bytes can no longer be changed => Permanent protection

Standard Flash protections

- Write protection
  - Protect Flash memory against unwanted modifications of the non-volatile code and/or data
  - Any Flash sector can be independently write-protected or unprotected

- Proprietary Code Readout Protection (PCROP)
  - Code area defined as **execute-only** to protect code from dumping
  - One PCROP area per bank can be defined with a granularity of 256 bytes
  - No data access is allowed (code shall be compiled with correct option)
  - **Only the Cortex-M7 core is allowed to execute (fetch) code from the PCROP area. Cortex M4 has no access to PCROP area.**

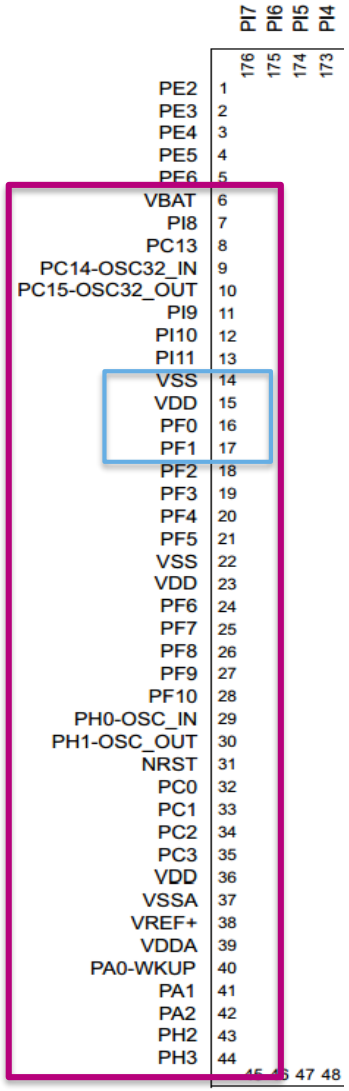
STM32H7 封装

life.augmented

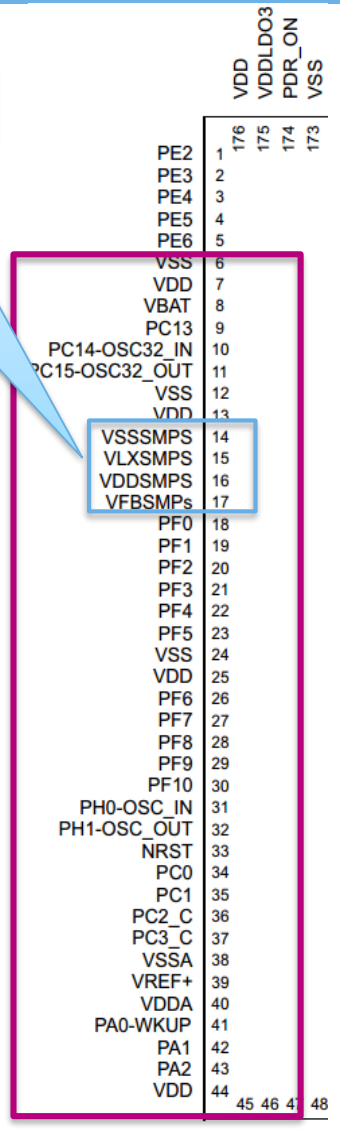
Pin package compatibility

Pin incompatibility between legacy and Industrial / Graphic lines (176 LQFP)

STM32H753xxx
STM32H743xxx
Legacy line



SMPS

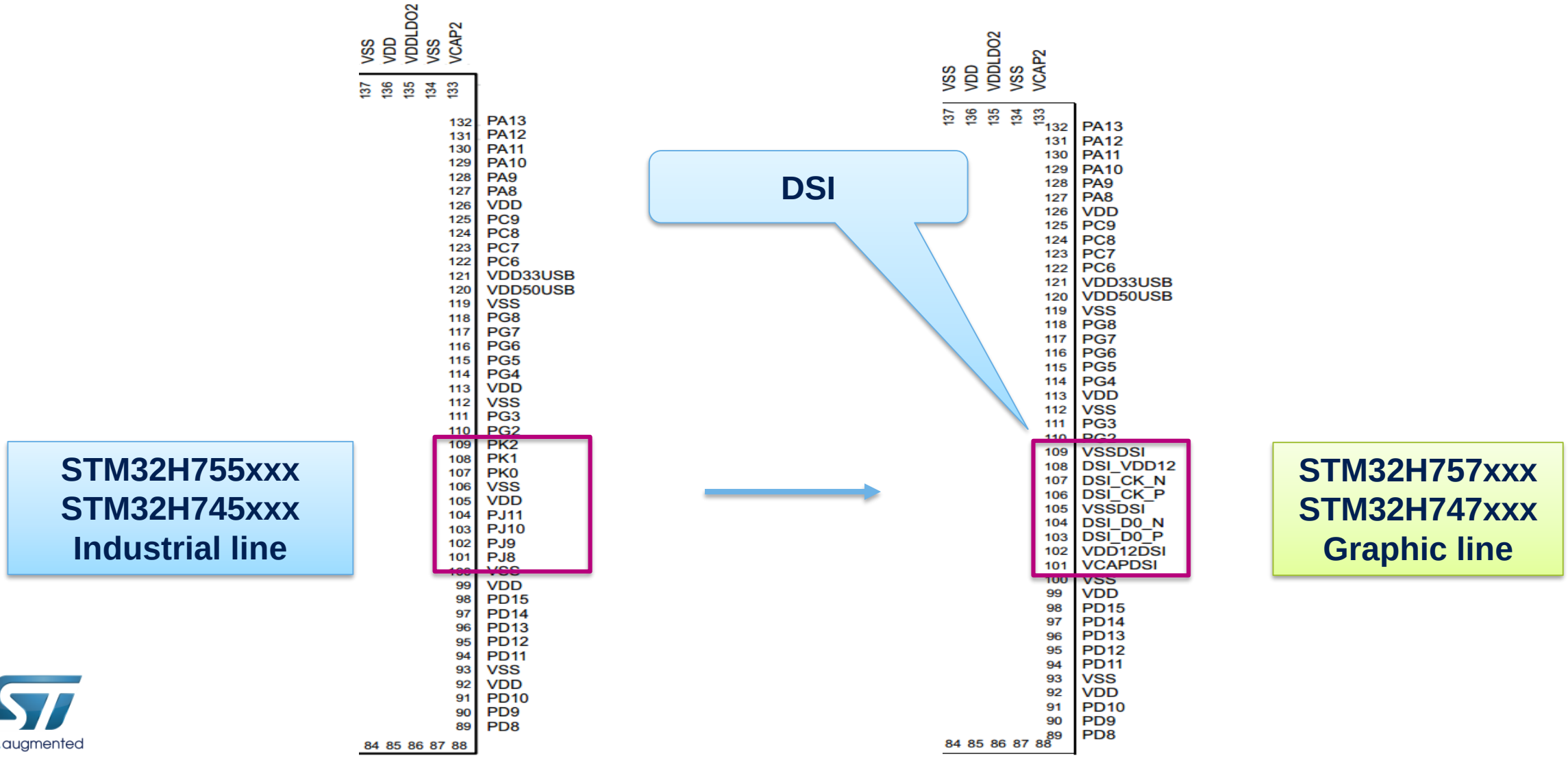


STM32H755xxx
STM32H745xxx
STM32H757xxx
STM32H747xxx
Industrial & Graphic line

life.augmented

Pin package compatibility

Pin incompatibility between Industrial and Graphic lines (176 LQFP)

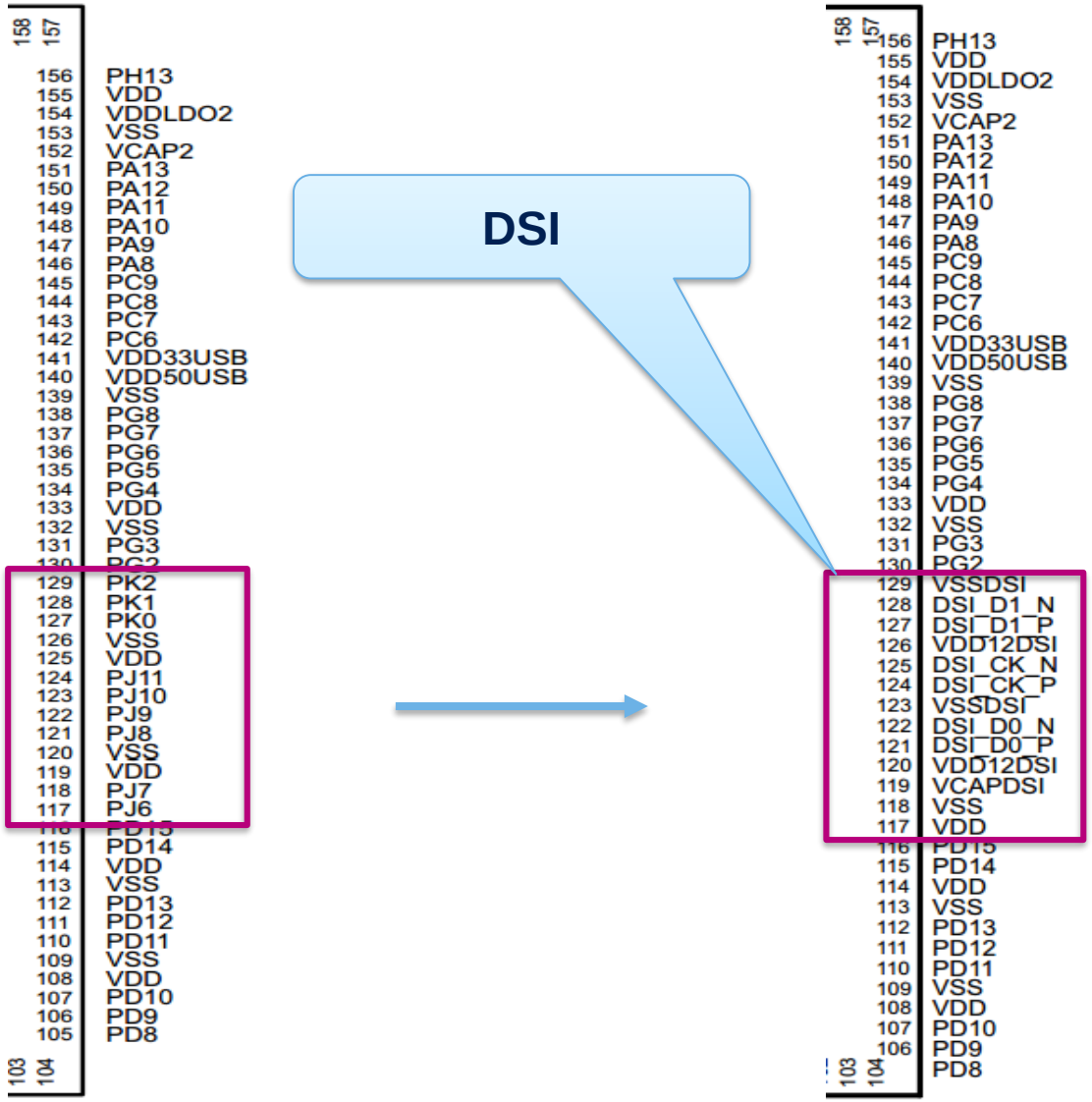


life.augmented

Pin package compatibility

Pin incompatibility between Industrial and Graphic lines (LQFP208)

STM32H755xxx
STM32H745xxx
Industrial line



STM32H757xxx
STM32H747xxx
Graphic line

life.augmented

WLCSP156 for Graphic line

- **WLCSP156** package for Graphical line only (STM32H7x7)

	13	12	11	10	9	8	7	6	5	4	3	2	1
A	DNC(1)	VDDLDO3	VCAP3	PB8	VDD	PB4	PG15	VDD	PD4	PD0	PA15	VDD LDO2	VSS
B	VBAT	PE4	VDD	PE0	VSS	PB5	PB3	VSS	PD3	PC12	VDD	VCAP2	PA12
C	PC14-OSC32_IN	PC15-OSC32_OUT	PE5	VSS	PB9	PB7	PB6	PD6	PC11	VSS	PA13	PA10	PA11
D	VDD	VSS SMPS	VSS	PE3	PE2	PE1	BOOT0	PD7	PC10	PA9	PA8	PC9	PC8
E	VDD SMPS	VLX SMPS	VFB SMPS	PF0	PC13	PE6	PDR_ON	PD2	PA14	PC7	VDD50 USB	VDD	VDD33 USB
F	PF3	PF2	PF4	PF5	PF1	PF11	PD5	PD1	PC6	PG4	VSS	PG8	PG5
G	VDD	VSS	PC0	PC1	PA6	PF12	PE10	PE11	PD8	PG3	PG2	DSI_ D1_P	DSI_ D1_N
H	PH1-OSC_OUT	PH0-OSC_IN	NRST	PA5	PB1	PF13	PE7	PB10	PB13	PD14	VSSDSI	DSI_ CK_P	DSI_ CK_N
J	VSSA	VREF+	VDDA	PA3	PA7	PF15	PE8	PE12	PB12	PD11	PD15	DSI_ D0_P	DSI_ D0_N
K	PC2_C	PC3_C	PA2	PA4	PB0	PF14	PE9	PE13	PB11	PD9	PD13	VDD	VCAP DSI
L	PA0-WKUP	PA1	VSS	PC5	VSS	PG0	VSS	PE15	VSS	VDDLDO1	PD10	PD12	VSS
M	VSS	VDD	PC4	PB2	VDD	PG1	VDD	PE14	VCAP1	VDD	PB14	PB15	VSS

For STM32H7x7 only

Thank you

/STM32 @ST_World st.com/e2e

life.augmented