微机系统与接口

——嵌入式处理器

東南大學電氣工程學院
SCHOOL OF ELECTRICAL ENGINEERING, SEU

➢嵌入式系统的组成

嵌入式系统的硬件

核心芯片、存储器系统、外部接口

嵌入式系统的软件

嵌入式操作系统和应用软件

嵌入式系统的开发工具和开发系统

语言编译器、连接定位器、调试器

嵌入式处理器

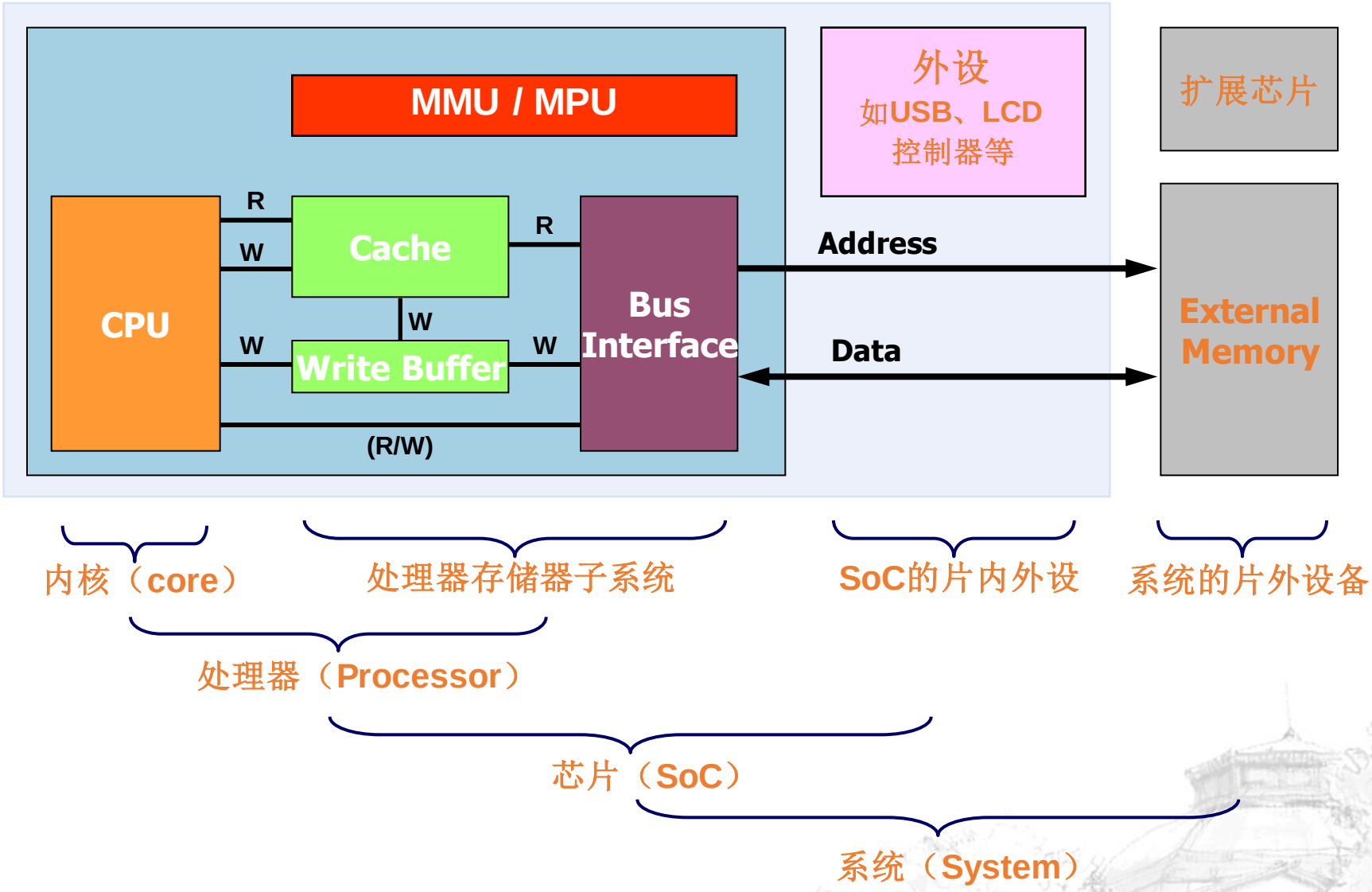
➢ 概述

嵌入式硬件系统基本组成

一个嵌入式系统产品包括硬件子系统和软件子系统，其中硬件子系统包括嵌入式处理器、总线、存储器、可编程输入输出接口即I/O以及外部设备。

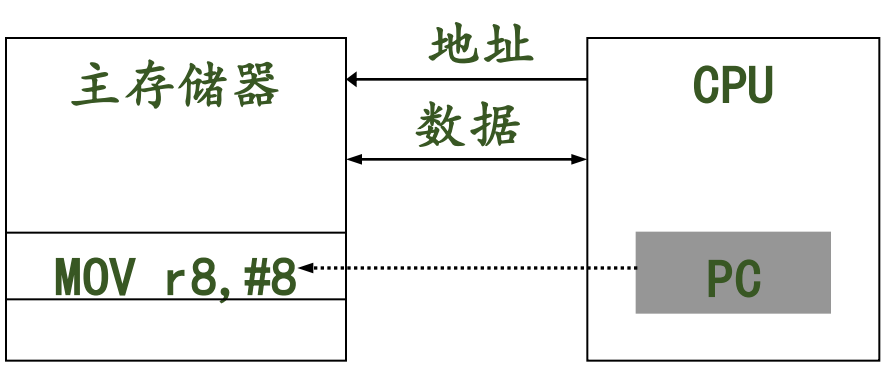
- 总线
- 存储器
- 输入/输出接口和设备

嵌入式系统

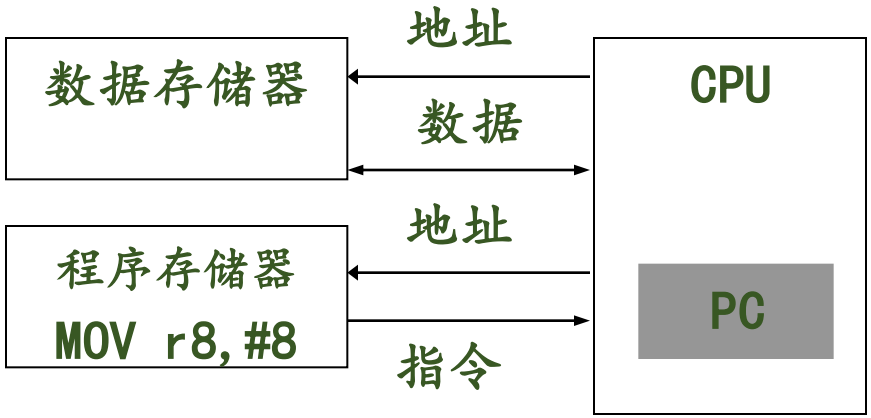

MMU / MPU
外设
如USB、LCD
控制器等
扩展芯片
CPU
R
W
Cache
R
W
W
W
Write Buffer
W
Bus
Interface
Address
Data
External
Memory
(R/W)
内核（core）
处理器存储器子系统
SoC的片内外设
系统的片外设备
处理器（Processor）
芯片（SoC）
系统（System）

嵌入式处理器

- 每个嵌入式系统至少包含一个嵌入式微处理器
- 嵌入式微处理器体系结构可采用冯•诺依曼（Von Neumann）结构或哈佛（Harvard）结构



冯•诺依曼结构



哈佛结构

总　线

- 嵌入式系统的总线一般集成在嵌入式处理器中。
- 从微处理器的角度来看，总线可分为片外总线(如：PCI、ISA等)和片内总线（如：AMBA、 AVALON、OCP、WISHBONE等）。
- 选择总线和选择嵌入式处理器密切相关，总线的种类随不同的微处理器的结构而不同。

存储器

- 嵌入式系统的存储器包括主存和外存。
- 大多数嵌入式系统的代码和数据都存储在处理器可直接访问的存储空间即主存中。
- 系统上电后在主存中的代码直接运行。主存储器的特点是速度快，一般采用ROM、EPROM、Nor Flash、SRAM、DRAM等存储器件。

存储器

- 目前有些嵌入式系统除了主存外，还有外存。外存是处理器不能直接访问的存储器，用来存放各种信息，相对主存而言具有价格低、容量大的特点。
- 在嵌入式系统中一般不采用硬盘而采用电子盘做外存，电子盘的主要种类有NandFlash、 SD （Secure Digital）卡、CompactFlash、SmartMedia、Memory Stick、MultiMediaCard、、 DOC（Disk On Chip）等。

输入/输出接口和设备

- 嵌入式系统的大多数输入/输出接口和部分设备已经集成在嵌入式微处理器中。
- 输入/输出接口主要有中断控制器、DMA、串行和并行接口等，设备主要有定时器（Timers）、计数器（counters）、看门狗（watchdog timers）、RTC、UARTs、PWM（Pulse width modulator）、AD/DA、显示器、键盘和网络等。

嵌入式处理器的指令集

嵌入式微处理器的指令系统可采用精简指令集系统RISC（Reduced Instruction Set Computer）或复杂指令集系统CISC（Complex Instruction Set Computer）

	CISC	RISC
价格	由硬件完成部分软件功能，硬件复杂性增加，芯片成本高	由软件完成部分硬件功能，软件复杂性增加，芯片成本低
性能	减少代码尺寸，增加指令的执行周期数	使用流水线降低指令的执行周期数，增加代码尺寸
指令集	大量的混杂型指令集，有简单快速的指令，也有复杂的多周期指令，符合HLL（high level language）	简单的单周期指令，在汇编指令方面有相应的CISC微代码指令
高级语言支持	硬件完成	软件完成
寻址模式	复杂的寻址模式，支持内存到内存寻址	简单的寻址模式，仅允许LOAD和STORE指令存取内存，其它所有的操作都基于寄存器到寄存器
控制单元	微码	直接执行
寄存器数目	寄存器较少	寄存器较多

嵌入式处理器的指令集

- 为满足应用领域的需要，嵌入式微处理器的指令集一般要针对特定领域的应用进行剪裁和扩充。
- 目前很多应用系统需要类似于DSP的数字处理功能。这些指令主要有：
  - ✓乘加(MAC)操作：它在一个周期中执行了一次乘法运算和一次加法运算。
  - ✓SIMD类操作：允许使用一条指令进行多个并行数据流的计算。
  - ✓零开销的循环指令：采用硬件方式减少了循环的开销。仅使用两条指令实现一个循环，一条是循环的开始并提供循环次数，另一条是循环体。
  - ✓多媒体加速指令：像素处理、多边形、3D操作等指令。

嵌入式处理器的性能

高端嵌入式处理器用于高强度计算的应用，使用不同的方法来达到更高的并行度

- ✓**单指令执行乘法操作：**通过加入额外的功能单元和扩展指令集，使许多操作能在一个单一的周期内并行执行。
- ✓**每个周期执行多条指令：**桌面和服务器的超标量处理器都支持单周期多条指令执行，在嵌入式领域通常使用VLIW(very large instruction word)来实现，这样只需较少的硬件，总体价格会更低些。例如TI的TMS320C6201芯片，通过使用VLIW方法，能在每个周期同时执行8条独立的32位指令。
- ✓**使用多处理器：**采用多处理器的方式满足应用系统的更高要求。一些嵌入式微处理器采用特殊的硬件支持多处理器。如TI的OMAP730包括了三个处理器核ARM9、ARM7、DSP。

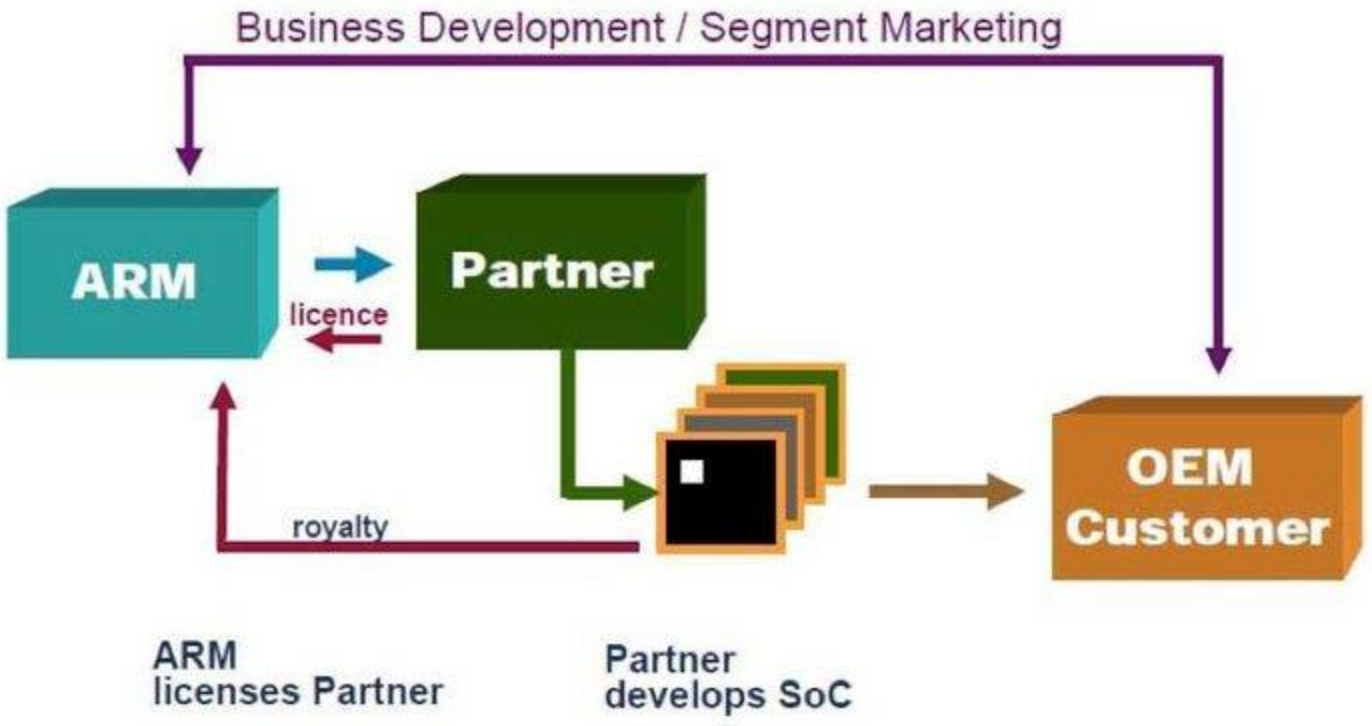
嵌入式处理器的功耗管理

- **提供功耗管理机制**
  - ✓**运行模式**（Running Mode）：处理器处于全速运行状态下。
  - ✓**待命模式**（Standby Mode）：处理器不执行指令，所有存储的信息是可用的，处理器能在几个周期内返回运行模式。
  - ✓**时钟关闭模式**（clock-off mode）：时钟完全停止，要退出这个模式系统需要重新启动。

- **影响功耗的其他因素还有总线（特别是总线转换器，可以采用特殊的技术使它的功耗最小）和存储器的大小（如果使用DRAM，它需要不断的刷新）。为了使功耗最小，总线和存储器要保持在应用系统可接受的最小规模。**

ARM系列

- 1991年，ARM公司成立于英国剑桥，主要出售芯片设计技术的授权。采用ARM技术知识产权（IP核）的微处理器，即我们通常所说的ARM 微处理器，已遍及工业控制、消费类电子产品、通信系统、网络系统、无线系统等各类产品市场，基于ARM技术的微处理器应用约占据了32位RISC微处理器75%以上的市场份额，ARM技术正在逐步渗入到我们生活的各个 高等三大
- 英国 手机和平板 销售实际的 授予IP许 ARM支付原 还提供了
- ARM ARM 式。套独 是一准的 RISC标 括 Int Qua 于软件系

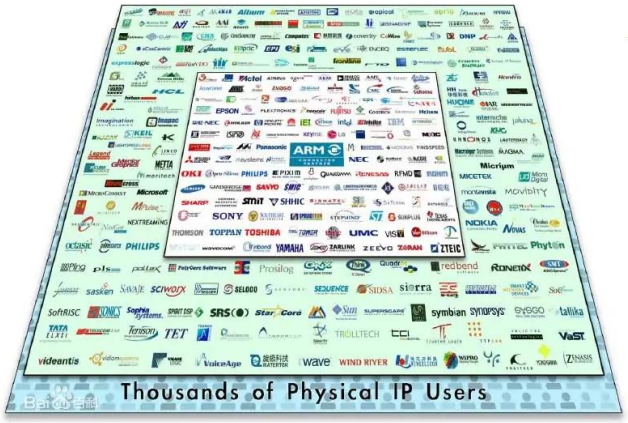


- 2016年7月18日，日本软银已经同意以234亿英镑（约合310亿美元）的价格收购英国芯片设计公司ARM。软银认为，凭借这笔收购，ARM将让软银成为下一个潜力巨大的科技市场（即物联网）的领导者。

资料源于：https://baike.baidu.com/item/ARM/5907#viewPageContent

ARM系列

ARM Ltd. headquarters



资料源于：https://baike.baidu.com/item/ARM/5907#viewPageContent

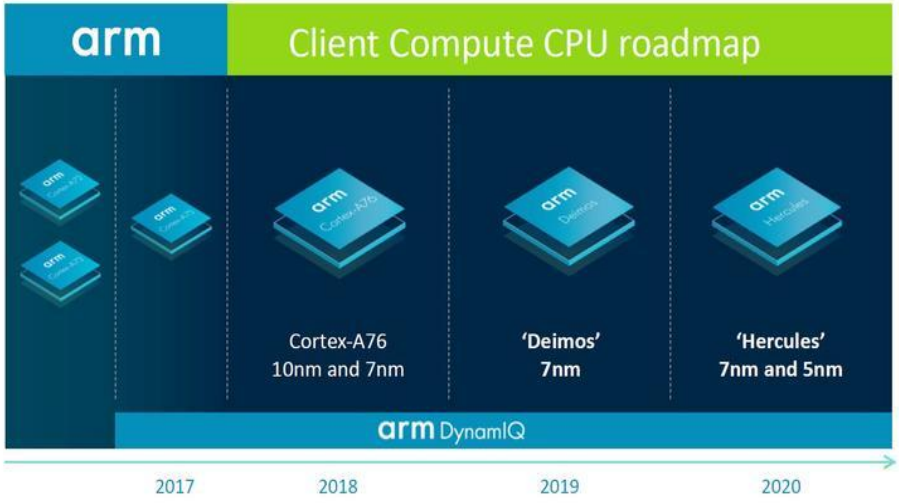
嵌入式处理器

➢ ARM处理器核简介

ARM公司开发了许多系列的ARM处理器核，目前最新的系列已经是ARM11、Cortex了。目前应用比较广泛的系列是：

ARM7	ARM9	ARM9E	ARM10
SecurCore	ARM11	Cortex	Xscale

6月初，Arm刚刚发布了全新Cortex A76架构，从设计的角度来看，它是一款完全重新打造的全新微架构，是“第二代奥斯汀家族”的领军者，代表了一个全新的开始。今天的路线图公开披露了Cortex A76之后的两代CPU架构代号——Deimos和Hercules，一如之前所言，这两款未来的架构都是基于新的Cortex A76微架构；在今年年底和未来几个月内，我们便可以在7nm的第一批商用设备上看到更多有关Cortex A76的信息。Deimos是其2019年的继任者，其目标是在7nm工艺上得到更广泛的应用，而Hercules被定位于2020年，将会是第一个针对5nm设计的架构。



嵌入式处理器

ARM处理器的版本及其对应的处理器家族

结构体系版本（Architecture）

- ARM v4T
- ARM v5TE
- ARM v6
- ARM v7

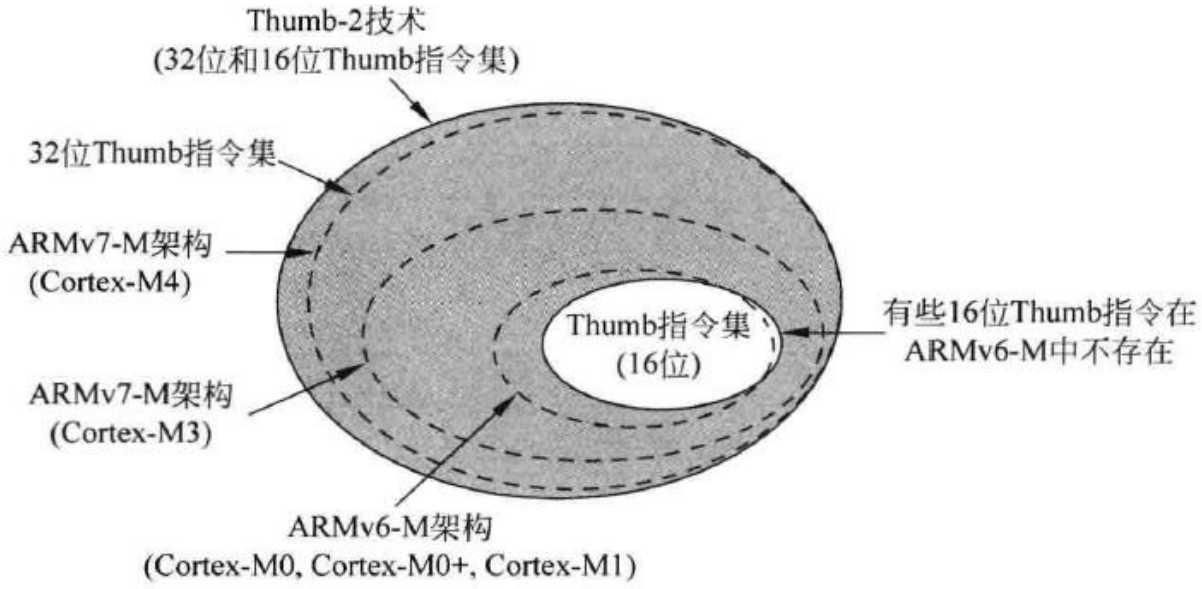
Processor Family

- ARM7
- ARM9
- ARM10
- ARM11
- ARM Cortex

按应用特征分类

- 应用处理器 Application Processor
  - 特征：MMU, Cache
  - 最快频率、最高性能、合理功耗
- 实时控制处理器 Real-time Controller
  - 特征：MPU, Cache
  - 实时响应、合理性能、较低功耗
- 微控制器 Micro-controller
  - 特征：no sub-memory system
  - 一般性能、最低成本、极低功耗

Thumb指令集和Cortex-M处理器实现的指令集间的差异



由于处理器支持Thumb-2 指令集中的16位和32位指令，因此无须在Thumb 状态(16位指令)和ARM状态(32位指令)间来回切换。

为什么 ARM 如此重要？

只要造手机，你就没法忽略这家芯片公司，这家来自于英国的技术公司，掌握着不少电子设备「大脑」中最核心的部分—CPU芯片里的指令集，它是一颗处理器所能执行的所有指令的总和，也是处理器运行的关键。而除了指令集，生产一颗 CPU 芯片自然也会有对应的硬件规格，它被称为「指令集架构」，而 ARM 带来的就是「ARM架构」了。

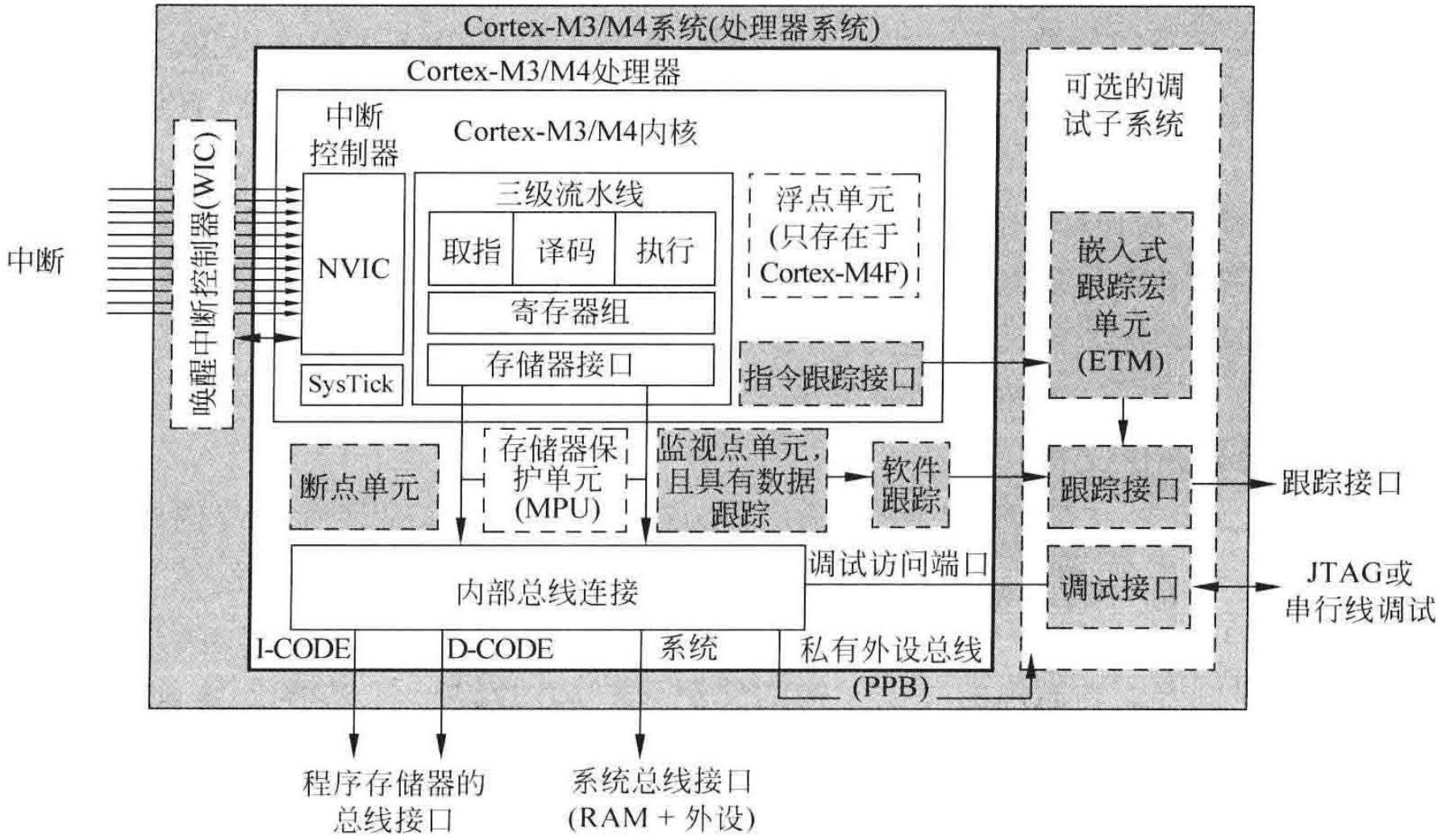
ARM架构特性盘点

ARM是32位RISC处理器

- ✓ 精简指令集
- ✓ Load/store 架构
- ✓ 多个32bit通用寄存器
- ✓ 多数指令为单周期执行指令
- ✓ 可并行执行的流水线结构
- ✓ 增强的指令
- ✓ 16bit Thumb 模式
- ✓ DSP 指令
- ✓ 条件执行指令
- ✓ 32 bit barrel shifter柱式位移器
- ✓ 特殊寄存器R13（堆栈指针），R14（链接），R15（PC）
- ✓ 任何对R15的操作可实现程序跳转

M4为扩展MCU应用范围而生

Cortex-M3/M4 处理器框图



M4处理器特性

ARM V7M 架构

- ✓ Thumb-2 技术
- ✓ SIMD 和DSP
- ✓ 单周期乘加指令（支持32 x 32 + 64 -> 64）
- ✓ 可选配的单精度浮点运算单元
- ✓ 集成可配置的可嵌套矢量中断控制器NVIC
- ✓ 兼容Cortex-M3

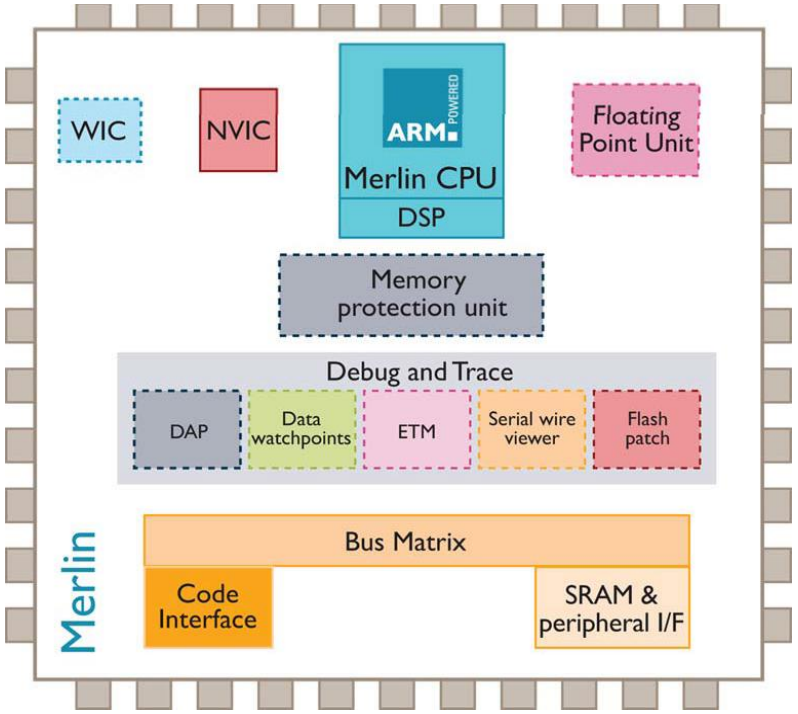
微内核架构

带分支预测的三级流水线

3套AHB-Lite 总线接口

可配置超低功耗

- 深度睡眠模式，中断可唤醒
- 浮点运算单元可单独关闭电源

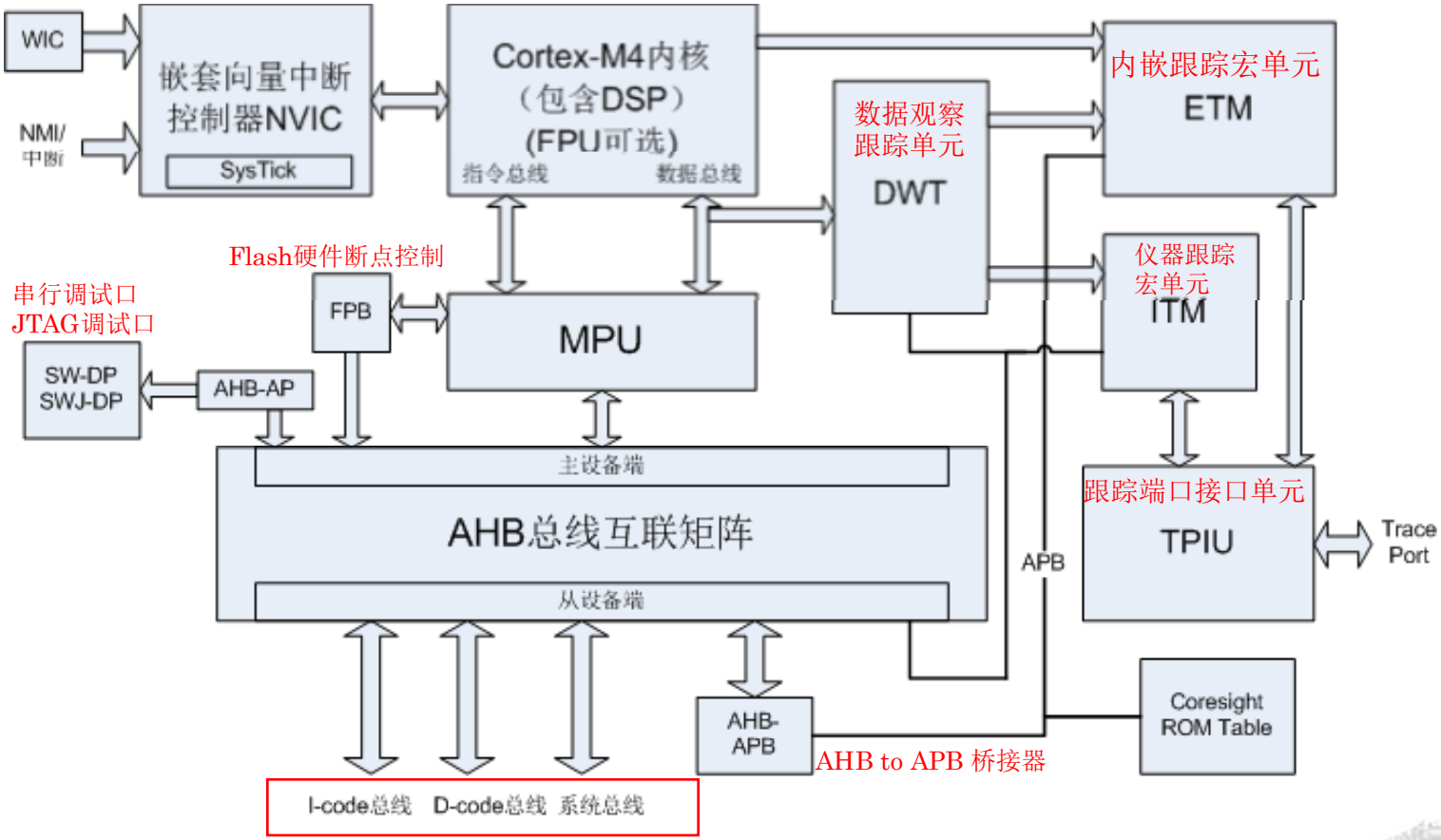


虚线框表示可选配部分

灵活配置

- 可配置中断控制器（1~240个中断源可配置，优先级可配置）
- 可选配的内存保护单元 - MPU（ Memory Protection Unit）
- 可选配的调试和跟踪模块

M4内核模块框图（一）



M4采用哈弗结构，为系统提供了3套总线，3套总线可以同时独立的发起总线传输读写操作

- I-code总线，用于访问代码空间的指令
- D-code总线，用于访问代码空间的数据
- System总线，访问其他系统空间

Cortex-M3和Cortex-M4处理器上的各种总线接口

- Cortex- M处理器的总线接口为32 位宽，且基于高级微控制器总线架构(AMBA) 标准，AMBA 中包含多个总线协议，任何芯片设计者都可以免费使用这些标准协议IP;
- Cortex-M3和Cortex-M4处理器主要使用的总线接口协议为AHB Lite（高级高性能总线），它用于程序存储器和系统总线接口;
- APB（高级外设总线）接口为处理器使用的另外一种总线协议，它通常用于基于ARM的微控制器的总线系统。

总线接口	描　　述
I-CODE	主要用于程序存储器，地址 0x0～0x1FFFFFFF 适用于取指和取向量操作，基于 AMBA 3.0 AHB Lite 总线协议
D-CODE	主要用于程序存储器，地址 0x0～0x1FFFFFFF 适用于数据和调试器访问操作，基于 AMBA 3.0 AHB Lite 总线协议
系统	主要用于 RAM 和外设，地址 0x20000000～0xFFFFFFFF(PPB 区域除外)适用于任何一种访问，基于 AMBA 3.0 AHB Lite 总线协议
PPB	外部私有外设总线(PPB)，用于地址 0xE0040000～0xE00FFFFF 范围内的私有调试部件，基于 AMBA 3.0 APB 协议
DAP	调试访问端口(DAP)接口，用于调试接口模块产生到包含系统存储器和调试部件在内的任意存储器位置的调试器访问。基于 ARM CoreSight 调试架构

M4内核模块框图（二）

➢ 核心处理器：中央内核（包含DSP），1.25 DMIPS/MHz，Thumb-2 ，单周期MAC，带可选配的单精度浮点运算单元

➢ NVIC 嵌套向量中断控制器 - 可配置的中断控制器

- 1:240 中断 -中断的具体路数由芯片厂商定义
- 采用向量中断的机制，自动取出对应的服务例程入口地址，无需软件判定
- 支持中断嵌套
- 1:255 优先级

NMI & SysTick

➢ SysTick定时器

- 倒计时定时器，用于在每隔一定的时间产生一个中断
- 系统睡眠模式下也可工作
- OS系统心跳定时

➢ Wake-up中断控制器

- 可配置
- 为低功耗模式提供唤醒功能
- 隔离不同的供电区域

M4内核模块框图（三）

- 内存保护单元MPU
  - ✓ 选配模块
  - ✓ 把内存分割成8个区域并进行保护
  - ✓ 非法访问将产生异常中断错误
- 总线互联矩阵BusMatrix
  - ✓ AHB互连的网络
  - ✓ 让数据在不同的总线之间并行传送 – 两个总线主机不访问同一块内存区域
  - ✓ 可支持bitbanding，实现按位操作一定的区域
- AHB to APB 桥接器
  - ✓ 从AHB总线转换到APB总线的桥接模块
  - ✓ APB总线用于访问系统上的私有慢速总线外设设备，通常为私有的调试模块
  - ✓ 芯片厂商可附加其他的外设设备

M4内核模块框图（四）- 可选配调试模块

➢ SW-DP/SWJ-DP串行调试口和支持JTAG的串行调试口

- 与AHB访问端口（AHB-AP）协同工作，将SW-DP的外部调试器命令转换成 内部总线命令
- 处理器核心没有JTAG扫描链，多数调试功能通过AHB访问来实现的
- SWJ-DP同时支持串行线协议和JTAG协议，而SW-DP只支持串行协议

➢ AHB-AP桥接

- 用于桥接SW-DP/SWJ-DP到AHB总线互联矩阵，从而发起AHB访问

➢ 内嵌跟踪宏单元ETM

- ETM与内核紧密耦合，可实现实时指令跟踪

➢ 数据观察跟踪单元DWT

- 可以设置数据观察点
- 当数据地址或数据的值匹配了观察点时，产生了一次匹配命中事件
- 匹配命中事件用于产生一个观察点事件，激活调试器以产生数据跟踪信息， 或让ETM联动

M4内核模块框图（五）- 可选配调试模块

➢ 仪器跟踪宏单元ITM

- 由软件驱动跟踪源，软件可以控制该模块直接把消息送给TPIU
- 输出的跟踪信息可以由软件设置，包括Printf类型的调试信息、操作系统及应 用程序的事件信息

➢ 跟踪端口的接口单元TPIU

- 用于和外部的跟踪硬件（如跟踪端口分析仪）交互

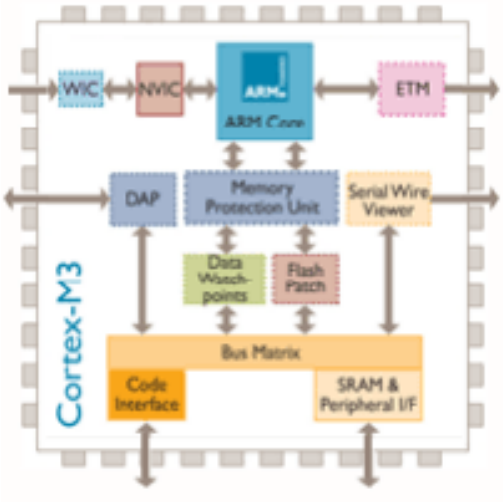
➢ Flash硬件断点控制FPB

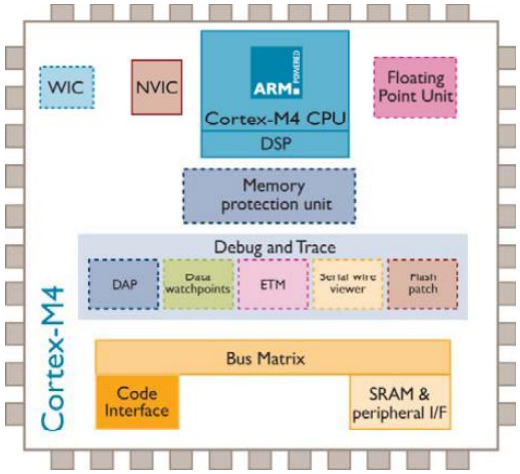
- FPB提供flash地址重载和断点功能
- 映射Flash上的指令地址，当内核访问该指令时，自动跳转到FPB指定的指令（通常为SRAM上的地址），实现硬件断点

➢ CoreSight ROM表

- 查找表，提供了系统包含哪些系统设备，调试组件，及寄存器的地址
- 方便芯片厂商根据自己的配置自定义添加模块
- 调试软件根据这些模块来自适应不同的芯片调试资源

Cortex-M4 vs Cortex-M3



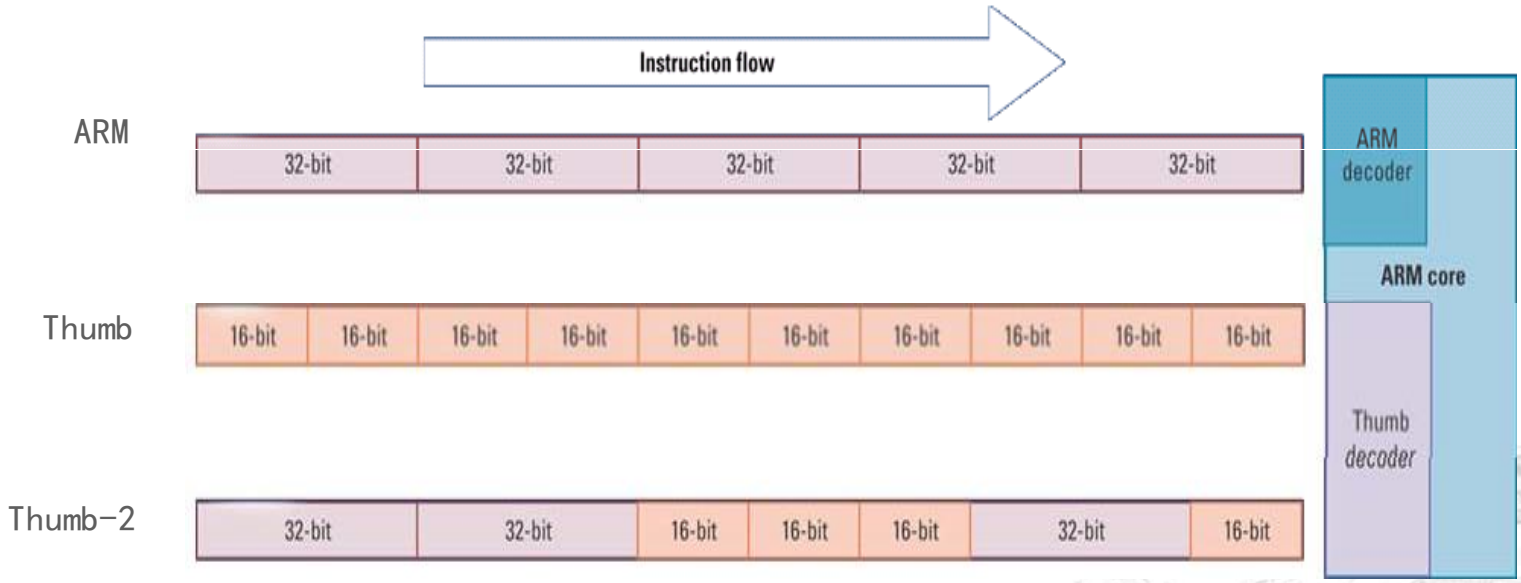


- Cortex-M4配备了单周期乘加（MAC）单元
- 优化了“单指令多数据指令”（SIMD）
- 优化了饱和算法指令
- 提供一个可选配的单精度浮点运算单元

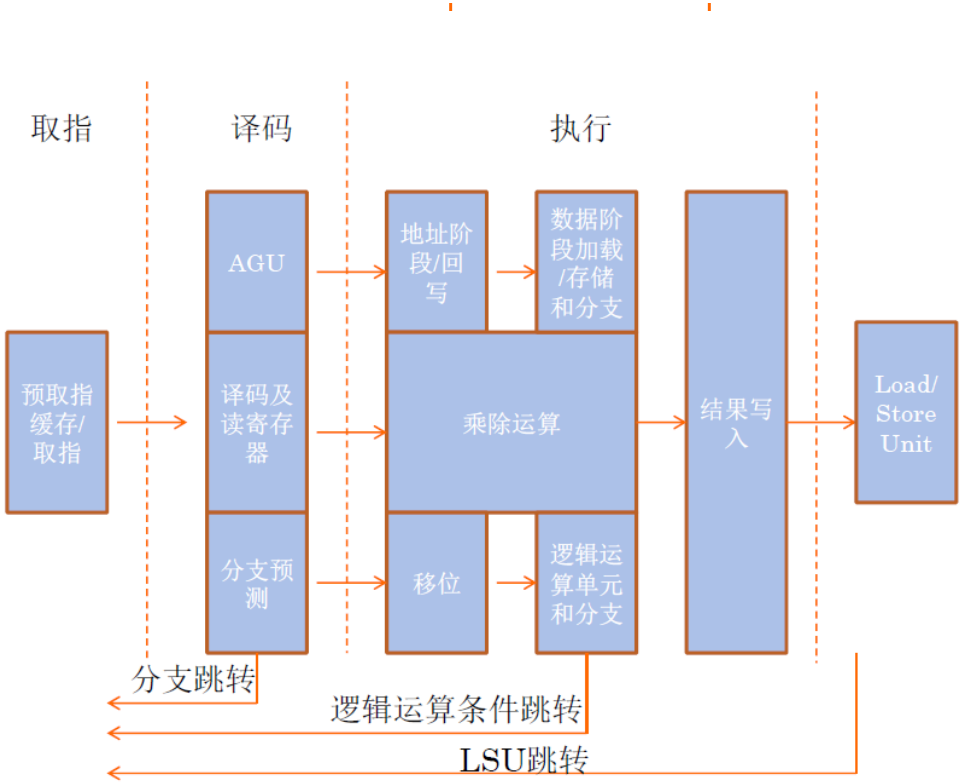
Thumb-2 技术

Thumb-2 指令集是ARMv7 架构的新技术

- 它兼容之前的Thumb指令集；
- 提供了一些新的16位Thumb指令以改善代码运行效率和流程；
- 提供了新的32位Thumb指令以改善性能和代码大小，一个32位指令可替代多个16位指令操作，32位指令与16位指令在同一处理器模式下解码执行，无需切换；
- Thumb-2指令集的设计是专门面向C语言的，且包括If/Then结构（预测接下来的四条语句的条件执行）、硬件除法以及本地位域操作。



Cortex-M4内核的流水线



Cortex-M4 为三级流水线架构

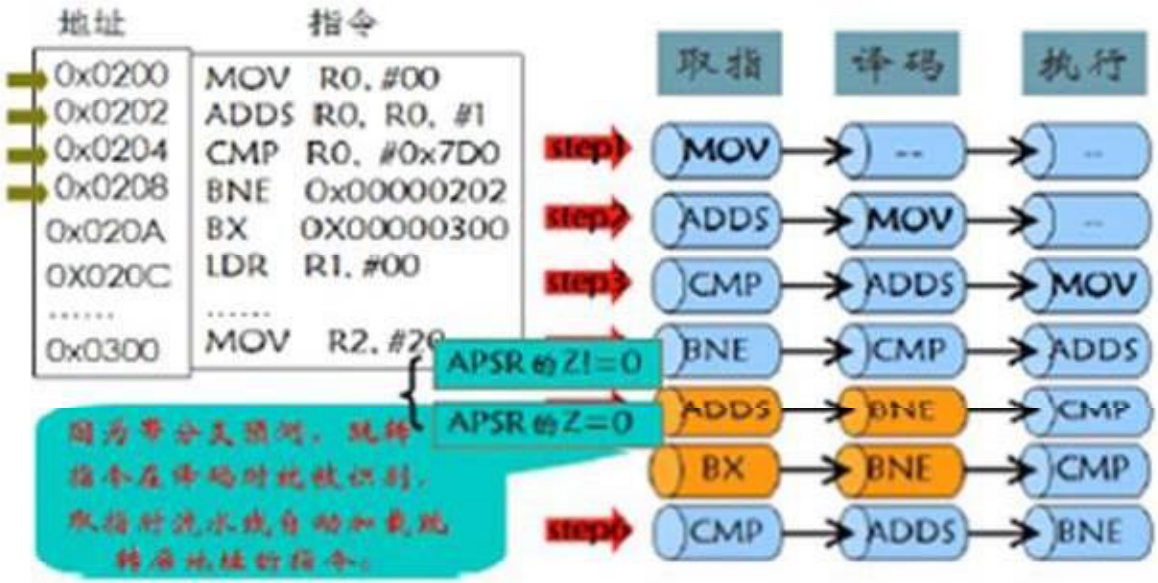
- **取指（Fetch）-** 用来计算下一个预取指令的地址，从指令 空间中取出指令，或者自动加载中断向量。此阶段还包含3 个长字的预取指缓冲区，用来做指令缓冲以及非对齐指令 的对齐（自动检测Thumb2指令）
- **译码（Decode）-** 解码指令，产生操作数的LSU（load/store Unit）地址，产生LR寄存器值
- **执行（Execute）-** 执行指令，产生LSU的回写执行结果， 执行乘除指令，以及进行逻辑运算并产生分支跳转

M4内核的流水线

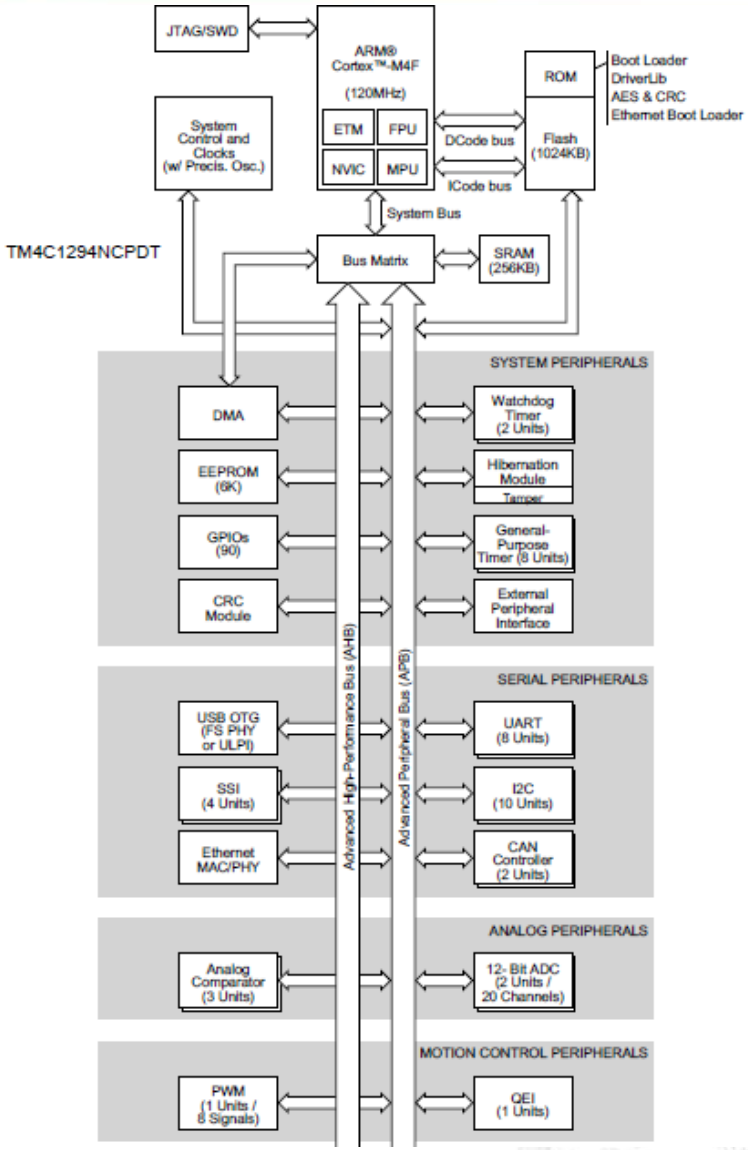
M4为32位系统，总线宽度为32位，因此一次可以取出32位的指令。如果代码都为16位的Thumb指令时，处理器会每隔一个周期做一次取指。如果缓冲区满，总线接取指则空闲下来。 当执行到跳转指令时，需要清洗流水线，处理器会不得不从跳转目的地重新取指。

M4内核的流水线－分支预测

- 一些算法需要对指令反复执行运算，简单的分支推测有利于减少因流水线清空所产生的开销
- 分支预测在指令在译码阶段就可以预测是否发生跳转，从而减少由于跳转分支打乱流水线导致的流水线气泡过大的问题



TI公司 Tiva TM4C1294NCPDT



➢处理器内核、功能和数据类型

- 嵌入式处理器内核

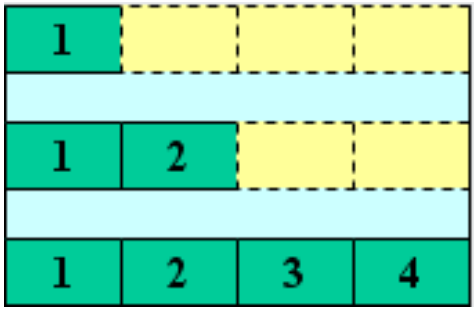
处理器内核（ARM）是一个计算机内核的设计架构，并不是一个芯片。由于这种开发方式和通用的总线架构的使用，每位芯片设计者都可以为ARM 处理器开发外设、存储器控制器以及片上存储器模块。这些设计通常被称作IP ，微控制器供应商可以在他们的产品中使用自己的外设设计或者其他公司的授权IP 。通过一种标准的总线协议，这些IP可以很轻松地被集成到一个大的设计中。

处理器内核按照体系结构（不涉及具体的处理器芯片），主要分为以下几类：

- MIPS
- ARM
- PowerPC
- 68K/COLDFIRE

➢处理器内核、功能和数据类型

- 嵌入式处理器支持的数据类型
  - 字节 8位
  - 半字 16位
  - 字 32位



➢处理器内核、功能和数据类型

- 嵌入式处理器支持的数据类型——大小端问题

例：int x, 地址是0x100，值是0x12345678

小端

0x103	12
0x102	34
0x101	56
0x100	78

大端

0x103	78
0x102	56
0x101	34
0x100	12

➢处理器状态、模式

- **处理器模式**

ARM体系结构支持7种处理器模式，分别为：

处理器模式	说明	备注
用户 (usr)	正常程序工作模式	不能直接切换到其它模式
系统 (sys)	用于支持操作系统的特权任务等	与用户模式类似，但具有可以直接切换到其它模式等特权
快中断 (fiq)	支持高速数据传输及通道处理	FIQ异常响应时进入此模式
中断 (irq)	用于通用中断处理	IRQ异常响应时进入此模式
管理 (svc)	操作系统保护代码	系统复位和软件中断响应时进入此模式
中止 (abt)	用于支持虚拟内存和/或存储器保护	在ARM7TDMI没有大用处
未定义 (und)	支持硬件协处理器的软件仿真	未定义指令异常响应时进入此模式

➢简介

在ARM处理器内部有37个用户可见的寄存器。

在不同的工作模式和处理器状态下，程序员可以访问的寄存器也不尽相同。

ARM状态各模式下的寄存器

寄存器类别	寄存器在汇编中的名称	各模式下实际访问的寄存器						
		用户	系统	管理	中止	未定义	中断	快中断
通用寄存器和程序计数器	R0(a1)	R0						
	R1(a2)	R1						
	R2(a3)	R2						
	R3(a4)	R3						
	R4(v1)	R4						
	R5(v2)	R5						
	R6(v3)	R6						
	R7(v4)	R7						
	R8(v5)	R8						R8_fiq
	R9(SB,v6)	R9						R9_fiq
	R10(SL,v7)	R10						R10_fiq
	R11(FP,v8)	R11						R11_fiq
	R12(IP)	R12						R12_fiq
	R13(SP)	R13		R13_svc	R13_abt	R13_und	R13_irq	R13_fiq
	R14(LR)	R14		R14_svc	R14_abt	R14_und	R14_irq	R14_fiq
	R15(PC)	R15						
状态寄存器	CPSR	CPSR						
	SPSR	无		SPSR_abt	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq

ARM状态各模式下的寄存器

寄存器类别	寄存器在汇编中的名称	各模式下实际访问的寄存器						
		用户	系统	管理	中止	未定义	中断	快中断
通用寄存器和程序计数器	R0(a1)	R0						
	R1(a2)	R1						
	R2(a3)	R2						
	R3(a4)	R3						
	R4(v1)	R4						
	R5(v2)	R5						
	R6	R6						
	R7	R7						
	R8	R8						R8_fiq
	R9	R9						R9_fiq
	R10	R10						R10_fiq
	R11	R11						R11_fiq
	R12(IP)	R12						R12_fiq
	R13(SP)	R13		R13_svc	R13_abt	R13_und	R13_irq	R13_fiq
	R14(LR)	R14		R14_svc	R14_abt	R14_und	R14_irq	R14_fiq
	R15(PC)	R15						
状态寄存器	CPSR	CPSR						
	SPSR	无		SPSR_abt	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq

所有的37个寄存器，分成两大类：

- 31个通用32位寄存器；
- 6个状态寄存器。

ARM状态各模式下可以访问的寄存器

寄存器类别	寄存器在汇编中的名称	各模式下实际访问的寄存器						
		用户	系统	管理	中止	未定义	中断	快中断
通用寄存器和程序计数器	R0(a1)	R0						
	R1(a2)	R1						
	R2(a3)	R2						
	R3(a4)	R3						
	R4(v1)	R4						
	R5(v2)	R5						
	R6(v3)	R6						
	R7(v4)	R7						
	R8(v5)	R8						R8_fiq
	R9(SB,v6)	R9						R9_fiq
	R10(SL,v7)	R10						R10_fiq
	R11(FP,v8)	R11						R11_fiq
	R12(IP)	R12						R12_fiq
	R13(SP)	R13		R13_svc	R13_abt	R13_und	R13_irq	R13_fiq
	R14(LR)	R14		R14_svc	R14_abt	R14_und	R14_irq	R14_fiq
	R15(PC)	R15						
状态寄存器	CPSR	CPSR						
	SPSR	无		SPSR_abt	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq

一般的通用寄存器

寄存器类别	寄存器在汇编中的名称	各模式下实际访问的寄存器						
		用户	系统	管理	中止	未定义	中断	快中断
通用寄存器程序	R0(a1)	R0						
	R1(a2)	R1						
		R2						
		R3						
		R4						
		R5						
		R6						
		R7						
		R8						R8_fiq
		R9						R9_fiq
		R10						R10_fiq
		R11						R11_fiq
	R12(IP)	R12						R12_fiq
	R13(SP)	R13		R13_svc	R13_abt	R13_und	R13_irq	R13_fiq
	R14(LR)	R14		R14_svc	R14_abt	R14_und	R14_irq	R14_fiq
	R15(PC)	R15						
状态寄存器	CPSR	CPSR						
	SPSR	无		SPSR_abt	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq

在汇编语言中,寄存器R0～R13是保存数据或地址值的通用寄存器。它们是完全通用的寄存器，不会被体系结构作为特殊用途，并且可用于任何使用通用寄存器的指令。

一般的通用寄存器

寄存器类别	寄存器在汇编中的名称	各模式下实际访问的寄存器						
		用户	系统	管理	中止	未定义	中断	快中断
通用寄存器	R0	R0						
	R1	R1						
	R2	R2						
	R3	R3						
	R4	R4						
	R5	R5						
	R6	R6						
	R7	R7						
	R8(v5)	R8						R8_fiq
	R9(SB,v6)	R9						R9_fiq
	R10(SL,v7)	R10						R10_fiq
	R11(FP,v8)	R11						R11_fiq
	R12(IP)	R12						R12_fiq
	R13(SP)	R13		R13_svc	R13_abt	R13_und	R13_irq	R13_fiq
	R14(LR)	R14		R14_svc	R14_abt	R14_und	R14_irq	R14_fiq
	R15(PC)	R15						
状态寄存器	CPSR	CPSR						
	SPSR	无		SPSR_abt	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq

其中R0～R7为未分组的寄存器，也就是说对于任何处理器模式，这些寄存器都对应于相同的32位物理寄存器。

一般的通用寄存器

寄存器类别	寄存器在汇编中的名称	各模式下实际访问的寄存器						
		用户	系统	管理	中止	未定义	中断	快中断
	R0(a1)	R0						
	R1(a2)	R1						
	R2(a3)	R2						
	R3(a4)	R3						
	R4(v1)	R4						
		R5						
		R6						
		R7						
		R8						R8_fiq
		R9						R9_fiq
		R10						R10_fiq
		R11						R11_fiq
		R12						R12_fiq
	R13(SP)	R13		R13_svc	R13_abt	R13_und	R13_irq	R13_fiq
	R14(LR)	R14		R14_svc	R14_abt	R14_und	R14_irq	R14_fiq
	R15(PC)	R15						
状态寄存器	CPSR	CPSR						
	SPSR	无		SPSR_abt	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq

寄存器R8～R14为分组寄存器。它们所对应的物理寄存器取决于当前的处理器模式，几乎所有允许使用通用寄存器的指令都允许使用分组寄存器

一般的通用寄存器

寄存器类别	寄存器在汇编中的名称	各模式下实际访问的寄存器						
		用户	系统	管理	中止	未定义	中断	快中断
通用寄存器和程序计数器	R0(a1)	R0						
	R1(a2)							
	R2(a3)							
	R3(a4)							
	R4(v1)							
	R5(v2)							
	R6(v3)							
	R7(v4)							
	R8(v5)	R8						R8_fiq
	R9(SB,v6)	R9						R9_fiq
	R10(SL,v7)	R10						R10_fiq
	R11(FP,v8)	R11						R11_fiq
	R12(IP)	R12						R12_fiq
	R13(SP)	R13		R13_svc	R13_abt	R13_und	R13_irq	R13_fiq
	R14(LR)	R14		R14_svc	R14_abt	R14_und	R14_irq	R14_fiq
	R15(PC)	R15						
状态寄存器	CPSR	CPSR						
	SPSR	无		SPSR_abt	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq

寄存器R8～R12有两个分组的物理寄存器。一个用于除FIQ模式之外的所有寄存器模式，另一个用于FIQ模式。这样在发生FIQ中断后，可以加速FIQ的处理速度。

一般的通用寄存器

寄存器类别	寄存器在汇编中的名称	各模式下实际访问的寄存器						
		用户	系统	管理	中止	未定义	中断	快中断
通用寄存器和程序计数器	R0(a1)	R0						
	R1(a2)	R1						
	R2(a3)	R2						
	R3(a4)	R3						
	R4(v1)	R4						
	R5(v2)	R5						
	R6(v3)	R6						
	R7(v4)							
	R8(v5)							R8_fiq
	R9(SB,v6)							R9_fiq
	R10(SL,v7)							R10_fiq
	R11(FP,v8)							R11_fiq
	R12(IP)	R12						R12_fiq
	R13(SP)	R13		R13_svc	R13_abt	R13_und	R13_irq	R13_fiq
	R14(LR)	R14		R14_svc	R14_abt	R14_und	R14_irq	R14_fiq
	R15(PC)	R15						
状态寄存器	CPSR	CPSR						
	SPSR	无		SPSR_abt	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq

寄存器R13、R14分别有6个分组的物理寄存器。1个用于用户和系统模式，其余5个分别用于5种异常模式。

堆栈指针寄存器R13（SP）

寄存器类别	寄存器在汇编中的名称	各模式下实际访问的寄存器						
		用户	系统	管理	中止	未定义	中断	快中断
通用寄存器和程序计数器	R0(a1)	R0						
	R1(a2)	R1						
	R2(a3)	R2						
	R3(a4)	R3						
	R4(v1)	R4						
	R5(v2)							
	R6(v3)							
	R7(v4)							
	R8(v5)							R8_fiq
	R9(SB,v6)							R9_fiq
	R10(SL,v7)							R10_fiq
	R11(FP,v8)							R11_fiq
	R12(IP)							R12_fiq
	R13(SP)	R13		R13_svc	R13_abt	R13_und	R13_irq	R13_fiq
	R14(LR)	R14		R14_svc	R14_abt	R14_und	R14_irq	R14_fiq
	R15(PC)	R15						
状态寄存器	CPSR	CPSR						
	SPSR	无		SPSR_abt	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq

寄存器R13常作为堆栈指针（SP）。在ARM指令集当中，没有以特殊方式使用R13的指令或其它功能，只是习惯上都这样使用。但是在Thumb指令集中存在使用R13的指令。

链接寄存器R14（LR）

寄存器类别	寄存器在汇编中的名称	各模式下实际访问的寄存器						
		用户	系统	管理	中止	未定义	中断	快中断
通用寄存器和程序计数器	R0(a1)	R0						
	R1(a2)	R1						
	R2(a3)	R2						
	R3(a4)							
	R4(v1)							
	R5(v2)							
	R6(v3)							
	R7(v4)							
	R8(v5)							
	R9(SB,v6)							
	R10(SL,v7)							
	R11(FP,v8)							
	R12(IP)							
	R13(SP)	R13		R13_svc	R13_abt	R13_und	R13_irq	R13_fiq
	R14(LR)	R14		R14_svc	R14_abt	R14_und	R14_irq	R14_fiq
	R15(PC)	R15						
状态寄存器	CPSR	CPSR						
	SPSR	无		SPSR_abt	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq

R14为链接寄存器（LR），在结构上有两个特殊功能：

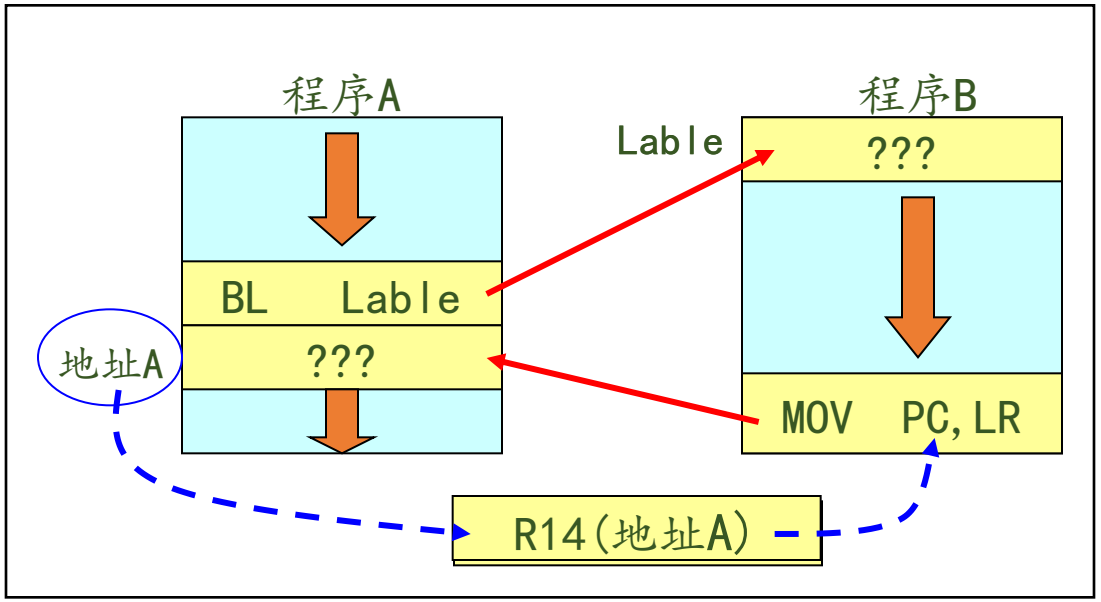
- 在每种模式下，模式自身的R14版本用于保存子程序返回地址；
- 当发生异常时，将R14对应的异常模式版本设置为异常返回地址（有些异常有一个小的固定偏移量）。

➢R14寄存器与子程序调用

操作流程

1. 程序A执行过程中调用程序B；

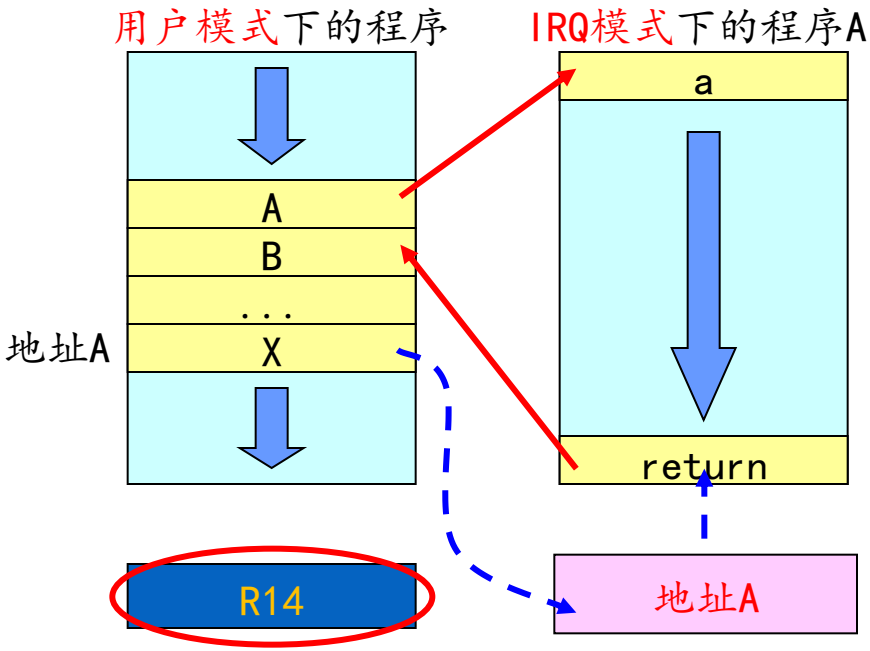
2. 程序跳转至标号Lable，执行程序B。同时硬件将“BL Lable”指令的下一条指令所在地址存入R14；



3. 程序B执行最后，将R14寄存器的内容放入PC，返回程序A；

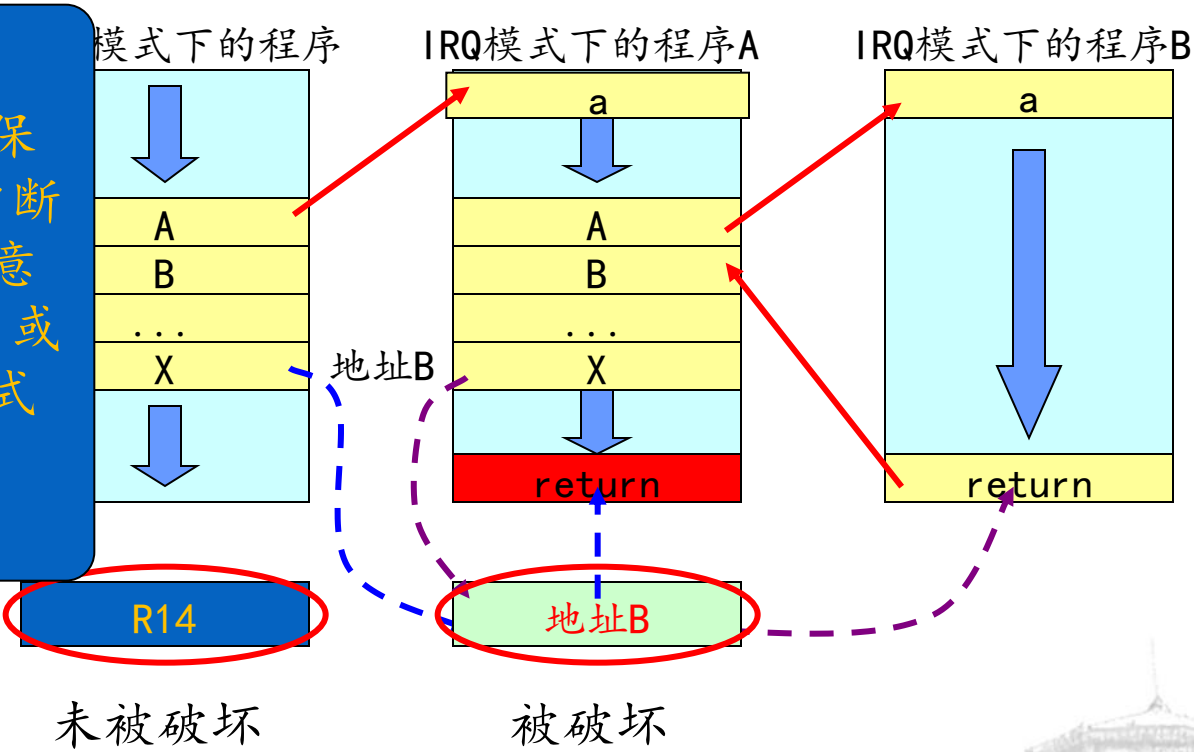
- R14寄存器注意要点

3. IRQ服务程序A执行完毕，将R14_irq寄存器的内容减去某个常量后存入PC，返回之前被中断的程序；



➢R14寄存器注意要点

解决办法是确保R14的对应版本在发生中断嵌套时不再保存任何有意义的值（将R14入栈），或者切换到其它处理器模式下。



程序计数器R15（PC）

寄存器类别	寄存器在汇编中的名称	各模式下实际访问的寄存器						
		用户	系统	管理	中止	未定义	中断	快中断
通用寄存器和程序计数器	R0(a1)	R0						
	R1(a2)	R1						
	R2(a3)	R2						
	R3(a4)	R3						
	R4(v1)							
	R5(v2)							
	R6(v3)							
	R7(v4)							
	R8(v5)							
	R9(SB,v6)							
	R10(SL,v7)							
	R11(FP,v8)							
	R12(IP)							
	R13(SP)							
	R14(LR)	R14		R14_svc	R14_abt	R14_und	R14_irq	R14_fiq
	R15(PC)	R15						
状态寄存器	CPSR	CPSR						
	SPSR	无		SPSR_abt	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq

寄存器R15为程序计数器（PC），它指向正在取指的地址。可以认为它是一个通用寄存器，但是对于它的使用有许多与指令相关的限制或特殊情况。如果R15使用的方式超出了这些限制，那么结果将是不可预测的。

➢Thumb状态下的寄存器

Thumb状态寄存器集是ARM状态集的子集，程序员可以直接访问的寄存器为：

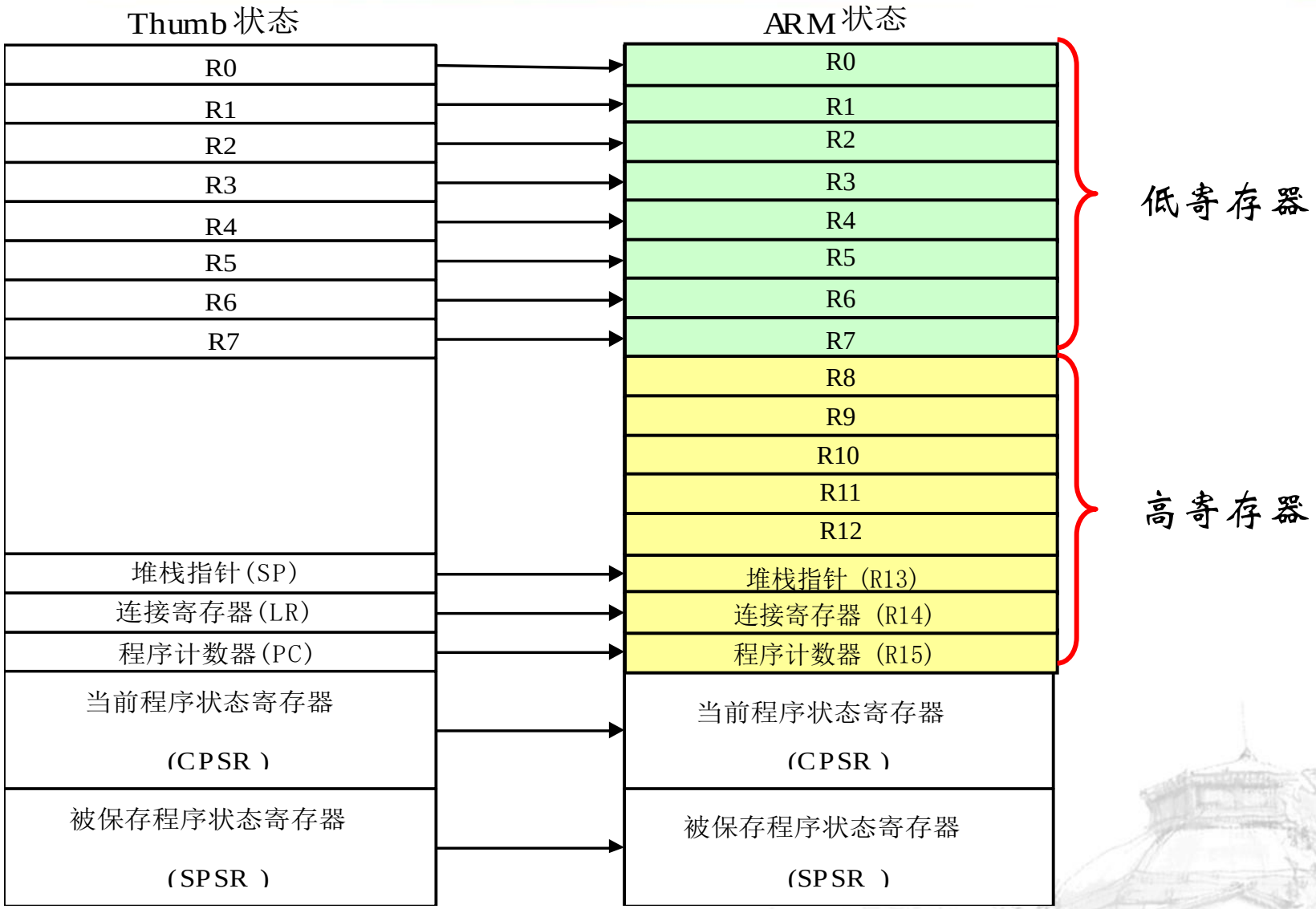
- 8个通用寄存器R0～R7；
- 程序计数器（PC）；
- 堆栈指针（SP）；
- 链接寄存器（LR）；
- 有条件访问程序状态寄存器（ CPSR）。

Thumb状态各模式下的寄存器

寄存器类别	寄存器在汇编中的名称	各模式下实际访问的寄存器						
		用户	系统	管理	中止	未定义	中断	快中断
通用寄存器和程序计数器	R0(a1)	R0						
	R1(a2)	R1						
	R2(a3)	R2						
	R3(a4)	R3						
	R4(v1)	R4						
	R5(v2)	R5						
	R6(v3)	R6						
	R7(v4,wr)	R7						
	SP	R13		R13_svc	R13_abt	R13_und	R13_irq	R13_fiq
	LR	R14		R14_svc	R14_abt	R14_und	R14_irq	R14_fiq
	PC	R15						
状态寄存器	CPSR	CPSR						

注意：括号内为ATPCS中寄存器的命名，可以使用RN汇编伪指令将寄存器定义多个名字。其中ADS1.2的汇编程序直接支持这些名称，但注意a1～a4，v1～v4必须用小写。

Thumb状态寄存器在Arm状态寄存器上的映射



嵌入式处理器——程序状态寄存器

➢简介

ARM7TDMI内核包含1个CPSR和5个供异常处理程序使用的SPSR。CPSR反映了当前处理器的状态，其包含：

- 4个条件代码标志（负(N)、零(Z)、进位(C)和溢出(V)）；
- 2个中断禁止位，分别控制一种类型的中断；
- 5个对当前处理器模式进行编码的位；
- 1个用于指示当前执行指令(ARM还是Thumb)的位。

嵌入式处理器——程序状态寄存器

➢简介