



ARM体系结构之(1)

亚嵌教育成都中心(RE.ER嵌入式学院)

--- 有思想的培训者

--- <http://www.re-er.com.cn>



内容

- n 计算机体系结构基础
- n ARM架构
- n ARM处理器的工作状态
- n ARM体系结构的存储器格式
- n ARM处理器的工作模式
- n ARM体系结构的寄存器组织
- n ARM处理器的异常处理



内容

- n 计算机体系结构基础
- n ARM架构
- n ARM处理器的工作状态
- n ARM体系结构的存储器格式
- n ARM处理器的工作模式
- n ARM体系结构的寄存器组织
- n ARM处理器的异常处理

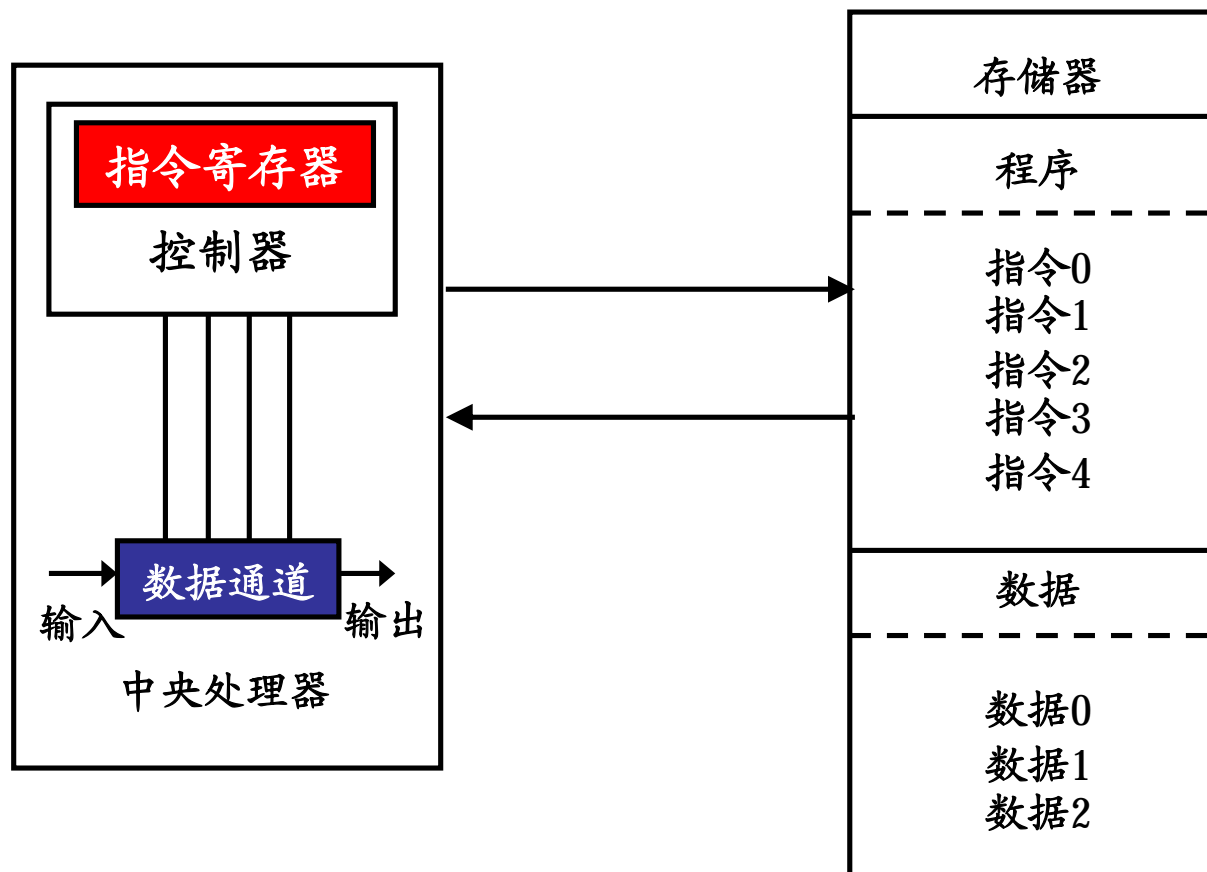


体系结构

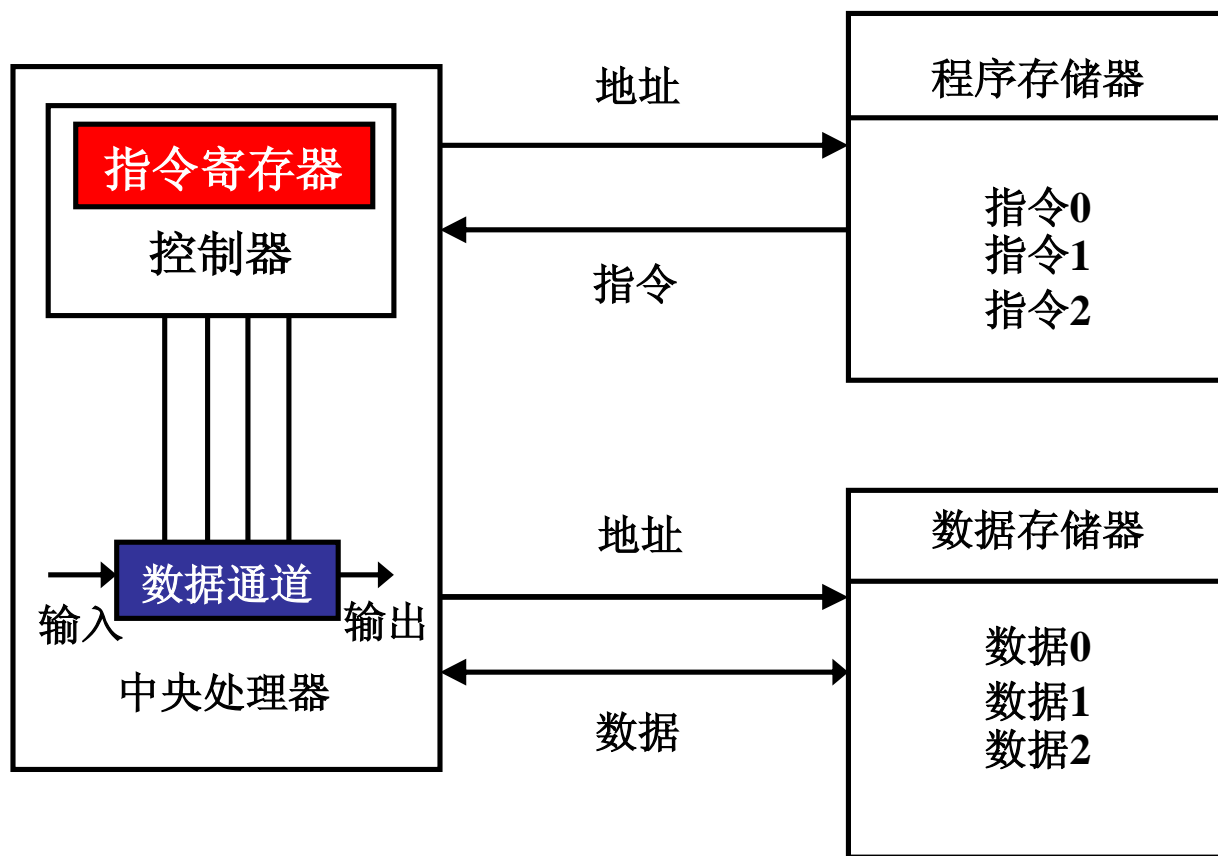
n 体系结构的发展历史

- Ø 冯诺伊曼之前的总线结构
- Ø 冯诺伊曼的观点
- Ø 冯诺伊曼结构（单一存储，统一编址）
- Ø 冯诺伊曼瓶颈（**DMA**传输：输入输出频繁且数据量特别大）
- Ø 哈佛结构
- Ø **ARM**体系中改进的哈佛结构

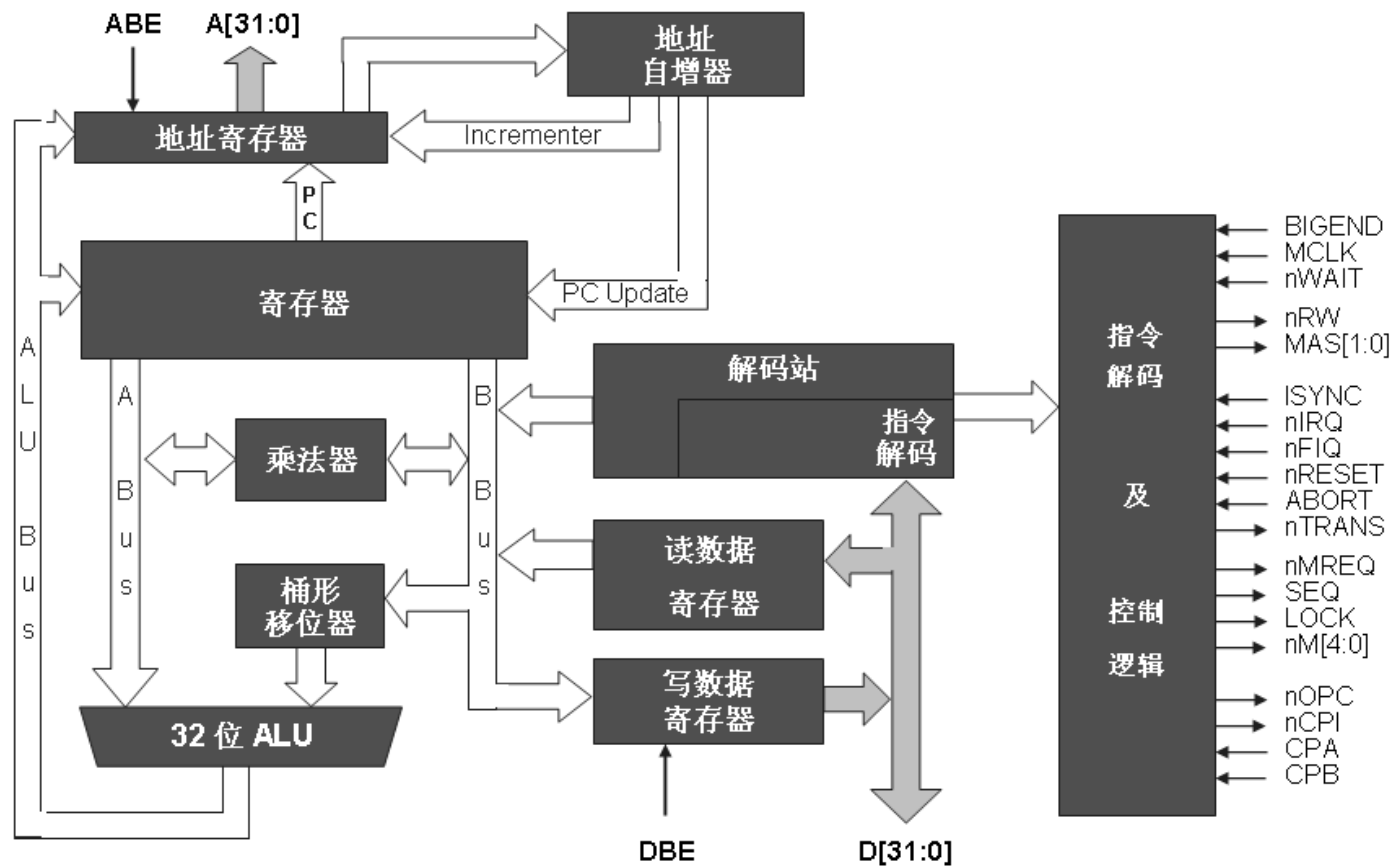
冯诺伊曼结构



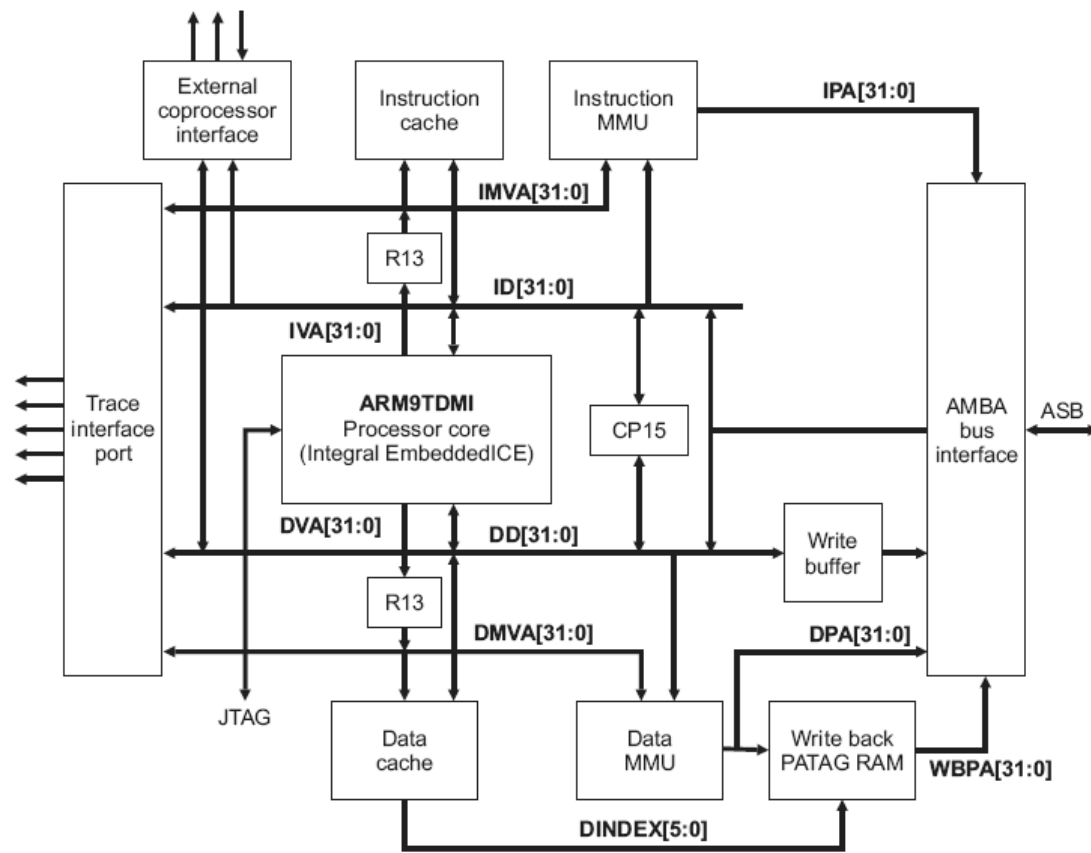
哈佛结构



ARM7



ARM9



ARM920T function block diagram



指令集（CISC vs RISC）

- n 指令集的发展历史

- o CISC（指令，微程序，微指令，片内，硬API）

- n 高级语言的编译得到了简化
 - n “直接执行”高级语言的计算机
 - n 卷积的指令级实现

- o 两个特点（长短不一，直接访存，寻址方式多样）



指令集（CISC vs RISC）

n CISC的弊端

- ∅ 流水线的建立（长短不一，访存方式复杂）
- ∅ 复杂指令的使用频度（80/20原则）
- ∅ 集成规模和成本（微指令的片内ROM）

机制 vs 策略

能不能减小指令的粒度？



指令集（CISC vs RISC）

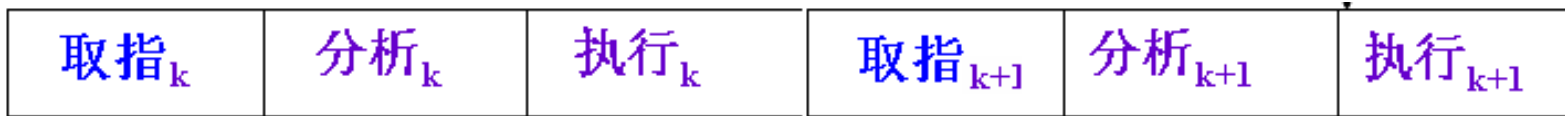
n RISC

- Ø 指令系统小（指令集规模，每条指令的复杂程度）
- Ø 配备专门的访存指令（寻址方式简单整齐）
- Ø 指令格式整齐划一（类似于微指令）
- Ø 寄存器数量多且通用



流水线技术

- n 多条指令执行方式
 - o 顺序
 - n 形式（第k和k+1条指令）



“ 执行时间 ” 与 “ 节拍 ”

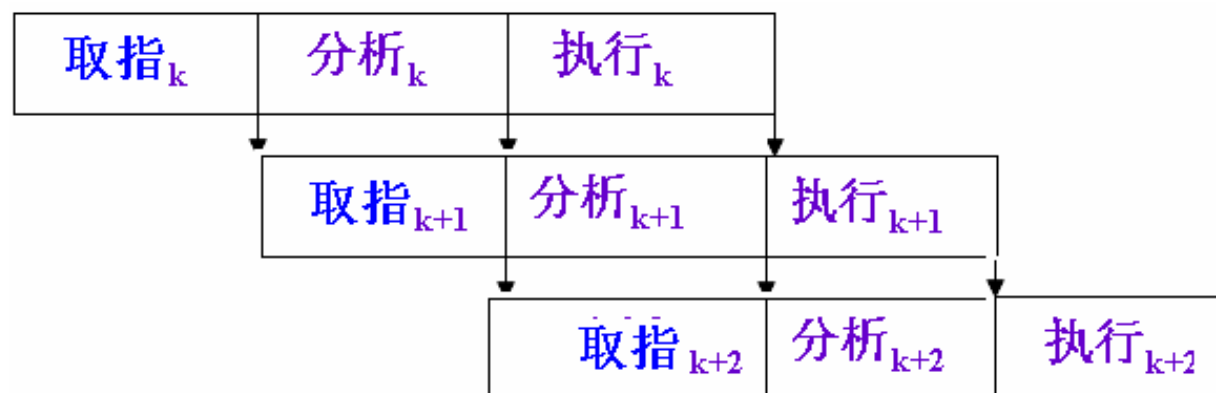
$$T = \sum_{i=1}^n (t_{\text{取}i} + t_{\text{分}i} + t_{\text{执}i})$$

流水线技术

n 多条指令执行方式

o 重叠

n 形式



n 时间

$$T = t_{\text{取}1} + \sum_{i=2}^n [\max\{t_{\text{分}i}, t_{\text{执}i-1}\}] + \sum_{i=2}^n [\max\{t_{\text{取}i}, t_{\text{分}i-1}\}] + t_{\text{执}n}$$

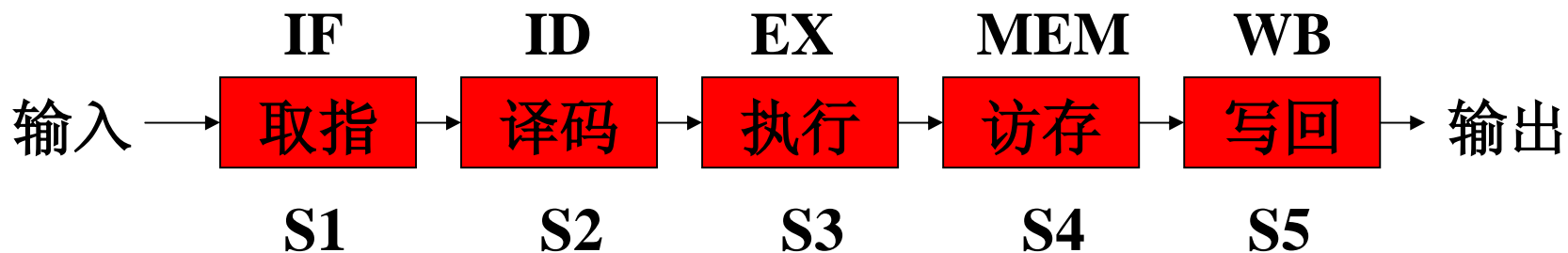
流水线技术

n 流水线

o 流水

把一个重复时序过程分成若干个子过程，每个子过程都可有效的在其专用功能段上和其它子过程同时执行的一种技术。

o 指令的流水线处理（连接图）



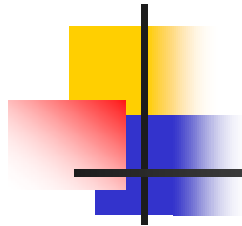


流水线技术

n 流水线

o 特点

- n 流水一定重叠，比重叠更苛刻。
- n 一条流水线通常有多个流水段组成。
- n 每段有专用功能部件，各部件顺序连接，不断流。
- n 流水线有建立时间、满载时间、排空时间，
- n 各段时间尽量短、一致；不一致时最慢子过程为瓶颈。



流水线——ARM7

ARM Thumb

PC

PC

Fetch

从存储器中读取指令

PC - 4

PC-2

Decode

解码指令

PC - 8

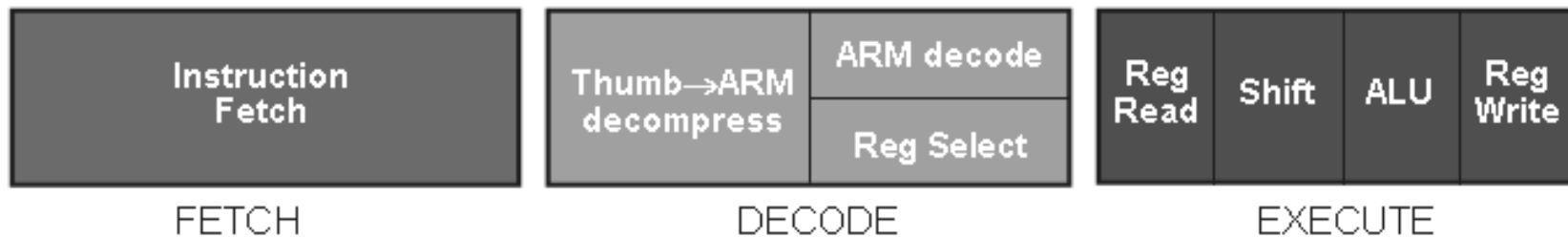
PC - 4

Execute

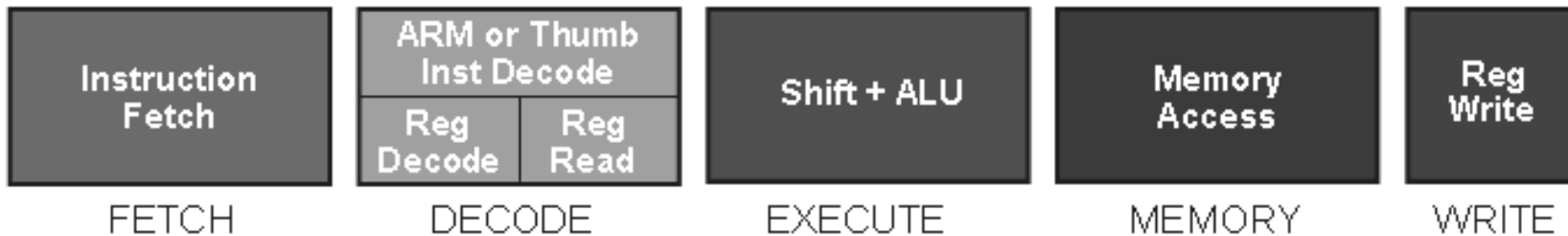
寄存器读（从寄存器Bank）
移位及ALU操作
寄存器写（到寄存器Bank）

流水线——ARM9

ARM7TDMI



ARM9TDMI





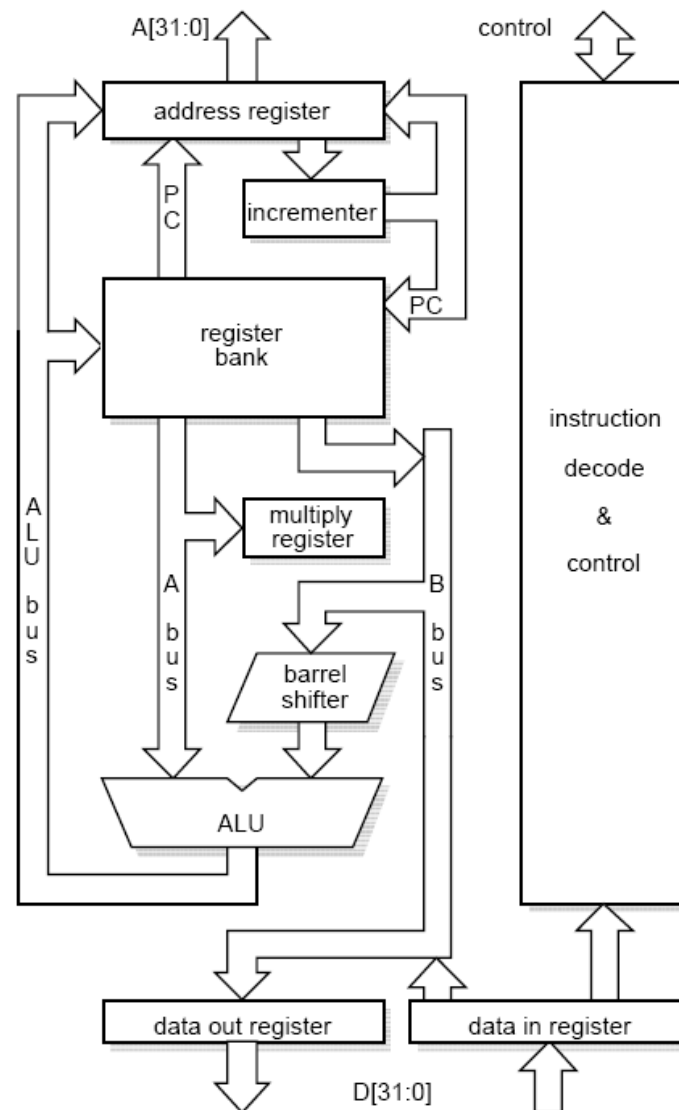
内容

- n 计算机体系结构基础
- n **ARM架构**
- n ARM处理器的工作状态
- n ARM体系结构的存储器格式
- n ARM处理器的工作模式
- n ARM体系结构的寄存器组织
- n ARM处理器的异常处理

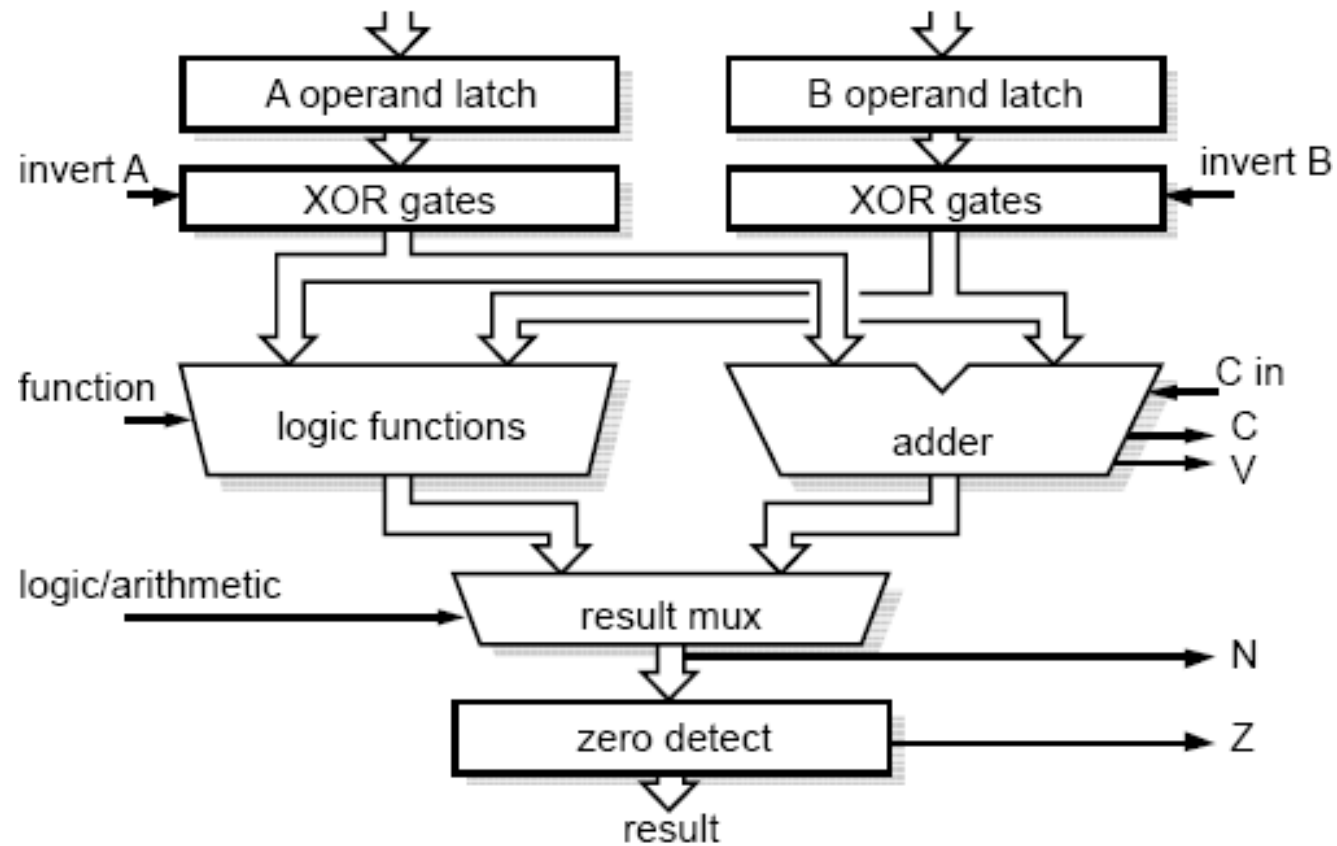
ARM架构

n 架构图

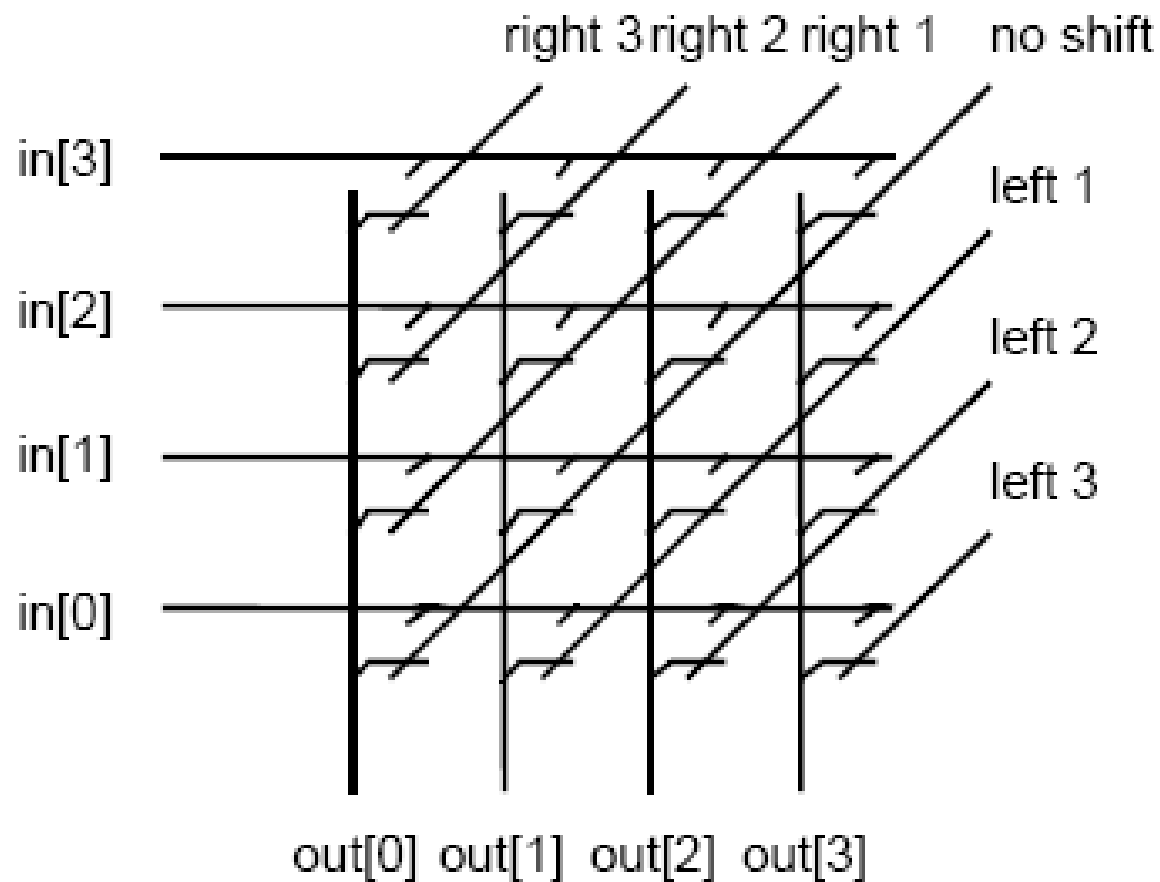
- ∅ 32bits ALU
- ∅ 31个32bits通用寄存器
- ∅ 6个状态寄存器
- ∅ 32X8bits乘法器
- ∅ 32X32桶形移位寄存器
- ∅ 指令译码器及控制器
- ∅ 指令流水线和数据/地址寄存器



ALU



桶形移位寄存器





内容

- n 计算机体系结构基础
- n ARM架构
- n **ARM处理器的工作状态**
- n ARM体系结构的存储器格式
- n ARM处理器的工作模式
- n ARM体系结构的寄存器组织
- n ARM处理器的异常处理



ARM处理器的工作状态

- n 从编程的角度看，ARM微处理器的工作状态一般有两种，并可在两种状态之间切换：
 - ø ARM状态，处理器执行32位的字对齐的ARM指令；
 - ø Thumb状态，处理器执行16位的、半字对齐的Thumb指令。



ARM处理器的工作状态

- n 在程序的执行过程中，微处理器可以随时在两种工作状态之间切换

- Ø 进入Thumb状态:

当操作数寄存器的状态位（位0）为1时，可以采用执行**BX**指令的方法，使微处理器从**ARM**状态切换到**Thumb**状态。此外，当处理器处于**Thumb**状态时发生异常（如**IRQ**、**FIQ**、**Undef**、**Abort**、**SWI**等），则异常处理返回时，自动切换到**Thumb**状态。

- Ø 进入ARM状态:

当操作数寄存器的状态位为0时，执行**BX**指令时可以使微处理器从**Thumb**状态切换到**ARM**状态。此外，在处理器进行异常处理时，把**PC**指针放入异常模式链接寄存器中，并从异常向量地址开始执行程序，也可以使处理器切换到**ARM**状态



内容

- n 计算机体系结构基础
- n ARM架构
- n ARM处理器的工作状态
- n **ARM体系结构的存储器格式**
- n ARM处理器的工作模式
- n ARM体系结构的寄存器组织
- n ARM处理器的异常处理



字、半字、字节

- n 字（Word）：在ARM体系结构中，字的长度为32位，而在8位/16位处理器体系结构中，字的长度一般为16位
- n 半字（Half-Word）：在ARM体系结构中，半字的长度为16位，与8位/16位处理器体系结构中字的长度一致。
- n 字节（Byte）：在ARM体系结构和8位/16位处理器体系结构中，字节的长度均为8位



大端格式和小端格式

大端格式和小端格式用于描述数据存储方法

- n 大端格式:

字数据的高字节存储在低地址中，
而字数据的低字节则存放在高地址中

- n 小端格式

低地址中存放的是字数据的低字节，高地址存放的是字数据的高字节



ARM体系结构的存储器格式

- n ARM体系结构将存储器看作是从零地址开始的字节的线性组合。从零字节到三字节放置第一个存储的字数据，从第四个字节到第七个字节放置第二个存储的字数据，依次排列。作为32位的微处理器，ARM体系结构所支持的最大寻址空间为4GB（2³²字节）



内容

- n 计算机体系结构基础
- n ARM架构
- n ARM处理器的工作状态
- n ARM体系结构的存储器格式
- n **ARM处理器的工作模式**
- n ARM体系结构的寄存器组织
- n ARM处理器的异常处理



处理器模式

- n 处理器模式决定了哪些寄存器是活动的以及对CPSR的访问权。处理器模式要么是特权模式，要么是非特权模式。特权模式允许对CPSR的完全读/写访问；与此相反，非特权模式只允许对CPSR的控制域进行读访问，但允许对条件标志的读/写访问。



处理器模式

- n ARM微处理器支持7种运行模式，分别为：
 - 用户模式（usr）：运行应用程序
 - 快速中断模式（fiq）：用于高速数据传输或通道处理
 - 外部中断模式（irq）：用于通用的中断处理
 - 管理模式（svc）：处理器复位以后进入，操作系统内核模式
 - 数据访问终止模式(abt)：访问存储器失败时进入。
 - 系统模式（sys）：特殊的用户模式，允许对cpsr的完全读写访问。
 - 未定义指令中止模式（und）：遇到未定义的指令或处理器不支持该指令时进入该模式。

- n 除用户模式外，其他6种为特权模式



内容

- n 计算机体系结构基础
- n ARM架构
- n ARM处理器的工作状态
- n ARM体系结构的存储器格式
- n ARM处理器的工作模式
- n **ARM体系结构的寄存器组织**
- n ARM处理器的异常处理







寄存器组织


- n ARM微处理器共有37个32位寄存器，其中31个为通用寄存器，6个为状态寄存器。
- n 但是这些寄存器不能被同时访问，具体哪些寄存器是可编程访问的，取决微处理器的工作状态及具体的运行模式。
 - o 在任何时候，通用寄存器R15~R0、一个或两个状态寄存器都是可访问的。这两个状态寄存器为cpsr和spsr(当前和备份的程序状态寄存器)
 - o 在非用户模式（特权模式）下，则可访问到特定模式分组寄存器

ARM状态下的通用寄存器与程序计数器











System & User	FIQ	Supervisor	About	IRQ	Undefined
R0	R0	R0	R0	R0	R0
R1	R1	R1	R1	R1	R1
R2	R2	R2	R2	R2	R2
R3	R3	R3	R3	R3	R3
R4	R4	R4	R4	R4	R4
R5	R5	R5	R5	R5	R5
R6	R6	R6	R6	R6	R6
R7	R7	R7	R7	R7	R7
R8	 R8_fiq	R8	R8	R8	R8
R9	 R9_fiq	R9	R9	R9	R9
R10	 R10_fiq	R10	R10	R10	R10
R11	 R11_fiq	R11	R11	R11	R11
R12	 R12_fiq	R12	R12	R12	R12
R13	 R13_fiq	 R13_svc	 R13_abt	 R13_irq	 R13_und
R14	 R14_fiq	 R14_svc	 R14_abt	 R14_irq	 R14_und
R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)

ARM状态下的程序状态寄存器

CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
	 SPSR_fiq	 SPSR_svc	 SPSR_abt	 SPSR_irq	 SPSR_und

 = 分组寄存器

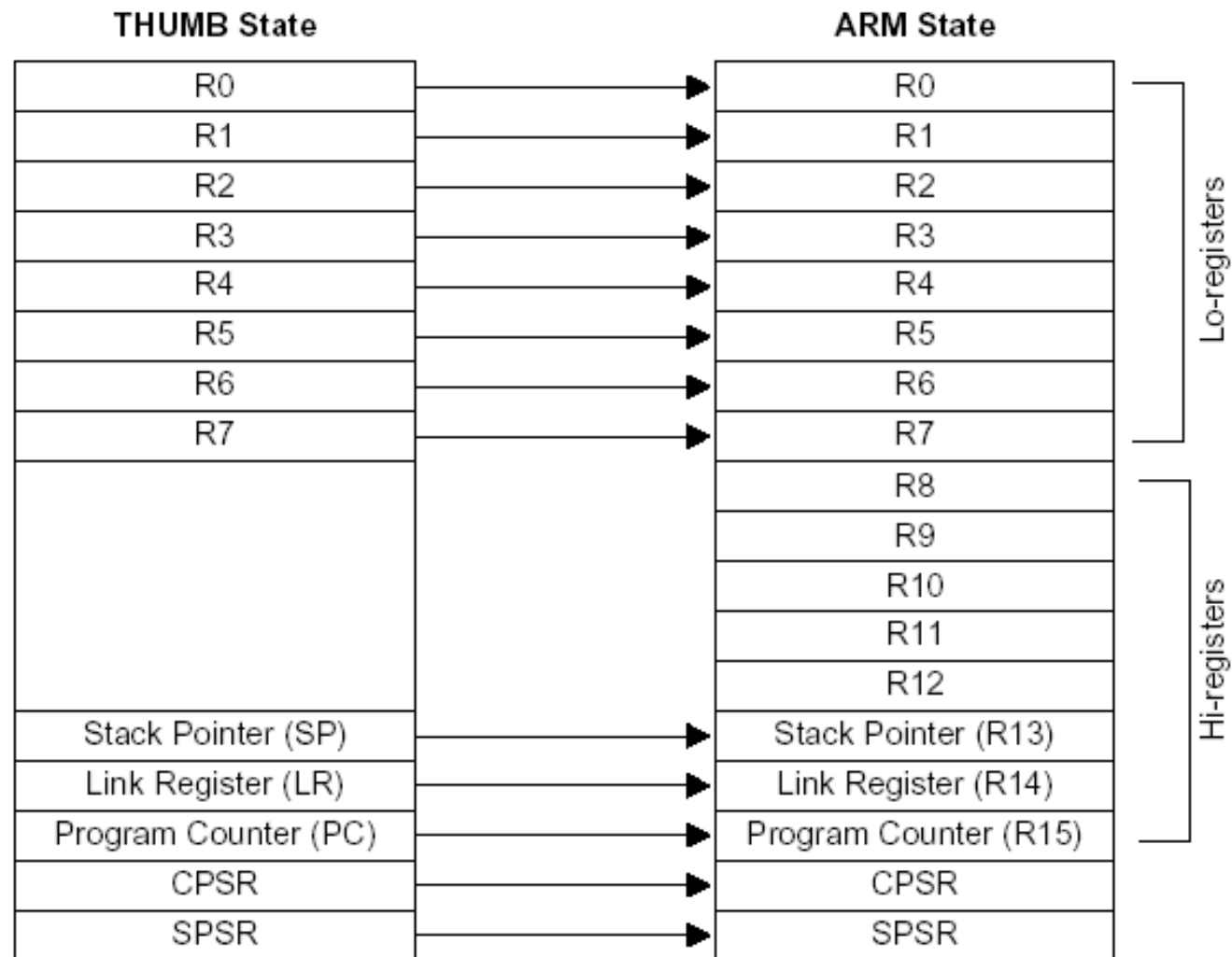
Thumb状态下的通用寄存器与程序计数器

System & User	FIQ	Supervisor	About	IRQ	Undefined
R0	R0	R0	R0	R0	R0
R1	R1	R1	R1	R1	R1
R2	R2	R2	R2	R2	R2
R3	R3	R3	R3	R3	R3
R4	R4	R4	R4	R4	R4
R5	R5	R5	R5	R5	R5
R6	R6	R6	R6	R6	R6
R7	R7	R7	R7	R7	R7
SP	 SP_fiq	 SP_svg	 SP_abt	 SP_irq	 SP_und
LR	 LR_fiq	 LR_svc	 LR_abt	 LR_irq	 LR_und
PC	PC	PC	PC	PC	PC

Thumb状态下的程序状态寄存器

CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
	 SPSR_fiq	 SPSR_svc	 SPSR_abt	 SPSR_irq	 SPSR_und

 = 分组寄存器





通用寄存器

- n 通用寄存器包括R0~R15，可以分为三类：
 - 未分组寄存器R0~R7；
 - 分组寄存器R8~R14
 - 程序计数器PC(R15)



1 未分组寄存器R0~R7

- n 在所有的运行模式下，未分组寄存器都指向同一个物理寄存器，他们未被系统用作特殊的用途



2 分组寄存器R8~R14

n 对于分组寄存器，他们每一次所访问的物理寄存器与处理器当前的运行模式有关。

o R8~R12:

每个寄存器对应两个不同的物理寄存器，当使用fiq模式时，访问寄存器R8_fiq~R12_fiq；当使用除fiq模式以外的其他模式时，访问寄存器R8_usr~R12_usr。

o R13、R14:

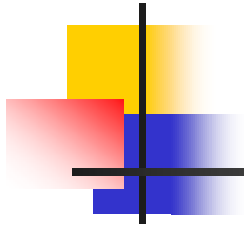
每个寄存器对应6个不同的物理寄存器，其中的一个为用户模式与系统模式共用，另外5个物理寄存器对应于其他5种不同的运行模式。

o 采用以下的记号来区分不同的物理寄存器:

R13_<mode>

R14_<mode>

其中，mode为以下几种模式之一：usr、fiq、irq、svc、abt、und。



n ARM处理器为特殊的任务或专门的功能指定了3个寄存器：
r13,r14和r15,他们通常被赋予不同的标识，以区别于其它寄存器。

- Ø R13: 堆栈指针(sp)，保存当前处理器模式的堆栈的栈顶
- Ø R14: 链接寄存器(lr)，保存调用子程序的返回地址
- Ø R15: 程序计数器(pc),其内容是处理器要取的下一条指令的地址。



寄存器R13 -堆栈指针(sp)

- n 由于处理器的每种运行模式均有自己独立的物理寄存器R13，在用户应用程序的初始化部分，一般都要初始化每种模式下的R13，使其指向该运行模式的栈空间，这样，当程序的运行进入异常模式时，可以将需要保护的寄存器放入R13所指向的堆栈，而当程序从异常模式返回时，则从对应的堆栈中恢复，采用这种方式可以保证异常发生后程序的正常执行。



寄存器R14-链接寄存器lr

- n 当执行BL子程序调用指令时，R14中得到R15（程序计数器PC）的备份。其他情况下，R14用作通用寄存器。与之类似，当发生中断或异常时，对应的分组寄存器R14_svc、R14_irq、R14_fiq、R14_abt和R14_und用来保存R15的返回值。



3 寄存器R15-程序计数器(PC)

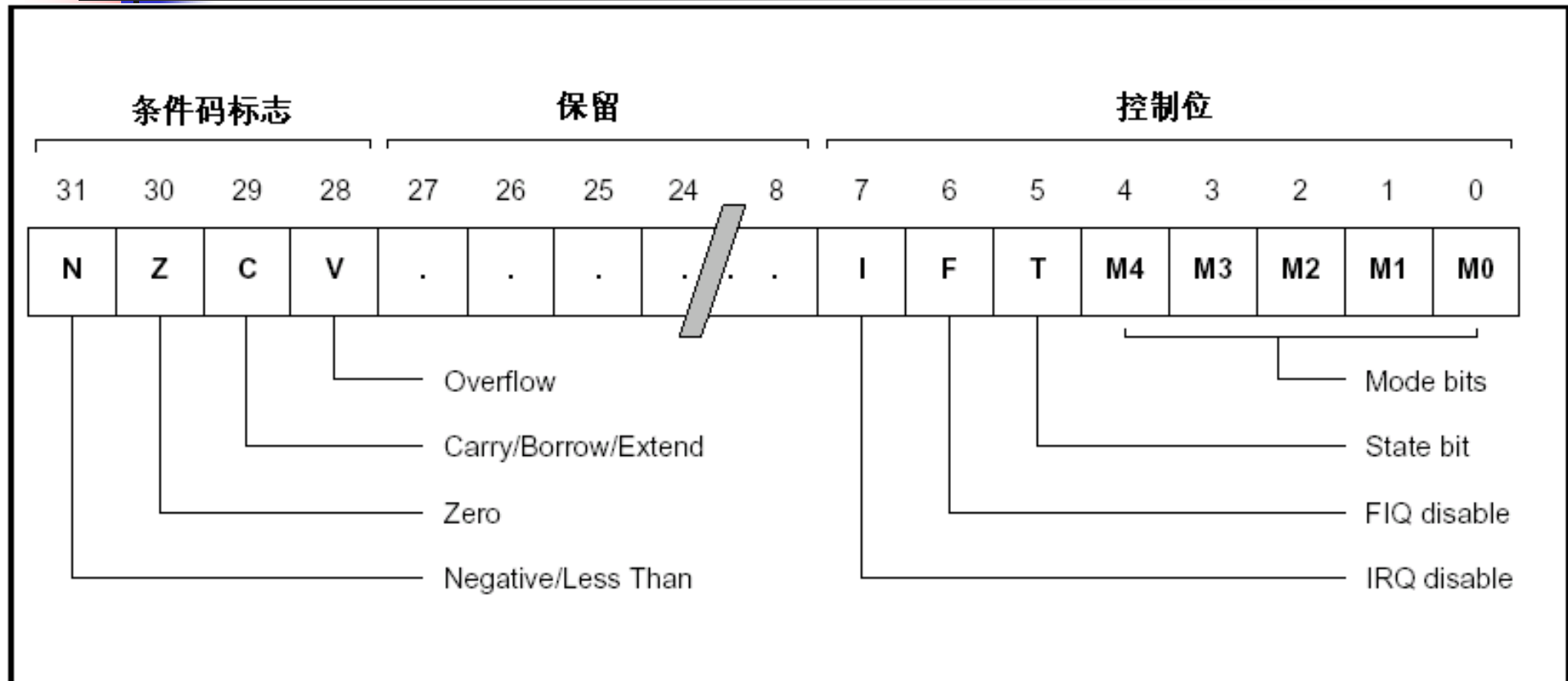
- n 寄存器R15用作程序计数器（PC）。在ARM状态下，位[1:0]为0，位[31:2]用于保存PC；在Thumb状态下，位[0]为0，位[31:1]用于保存PC；



程序状态寄存器

- n 寄存器R16用作CPSR(Current Program Status Register, 当前程序状态寄存器), CPSR可在任何运行模式下被访问, 它包括条件标志位、中断禁止位、当前处理器模式标志位, 以及其他一些相关的控制和状态位。
- n 每一种运行模式下又都有一个专用的物理状态寄存器, 称为SPSR (Saved Program Status Register, 备份的程序状态寄存器), 当异常发生时, SPSR用于保存CPSR的当前值, 从异常退出时则可由SPSR来恢复CPSR。备份的程序状态寄存器用来进行异常处理, 其功能包括:
 - 保存ALU中的当前操作信息
 - 控制允许和禁止中断
 - 设置处理器的运行模式
- n 由于用户模式和系统模式不属于异常模式, 他们没有SPSR, 当在这两种模式下访问SPSR, 结果是未知的。

程序状态寄存器



标志位	含义
N	当用两个补码表示的带符号数进行运算时， N=1 表示运算的结果为负数； N=0 表示运算的结果为正数或零
Z	Z=1 表示运算的结果为零； Z=0 表示运算的结果为非零；
C	可以有4种方法设置C的值： — 加法运算（包括比较指令 CMN ）：当运算结果产生了进位时（无符号数溢出）， C=1 ，否则 C=0 。 — 减法运算（包括比较指令 CMP ）：当运算时产生了借位（无符号数溢出）， C=0 ，否则 C=1 。 — 对于包含移位操作的非加/减运算指令， C 为移出值的最后一位。 — 对于其他的非加/减运算指令， C 的值通常不改变。
V	可以有2种方法设置V的值： — 对于加/减法运算指令，当操作数和运算结果为二进制的补码表示的带符号数时， V=1 表示符号位溢出。 — 对于其他的非加/减运算指令， C 的值通常不改变。
Q	在 ARM v5 及以上版本的E系列处理器中，用 Q 标志位指示增强的 DSP 运算指令是否发生了溢出。在其他版本的处理器中， Q 标志位无定义。



控制位

M[4: 0]	处理器模式	可访问的寄存器
0b10000	用户模式	PC, CPSR, R0-R14
0b10001	FIQ模式	PC, CPSR, SPSR_fiq, R14_fiq-R8_fiq, R7~R0
0b10010	IRQ模式	PC, CPSR, SPSR_irq, R14_irq, R13_irq, R12~R0
0b10011	管理模式	PC, CPSR, SPSR_svc, R14_svc, R13_svc, R12~R0,
0b10111	中止模式	PC, CPSR, SPSR_abt, R14_abt, R13_abt, R12~R0,
0b11011	未定义模式	PC, CPSR, SPSR_und, R14_und, R13_und, R12~R0,
0b11111	系统模式	PC, CPSR (ARM v4及以上版本), R14~R0



中断屏蔽

- n 中断屏蔽位用来禁止某些中断请求来中断处理器。**ARM**处理器内核有2个级别的中断请求—中断请求**IRQ**和快速中断请求**FIQ**
- n **Cpsr**有2个中断屏蔽位，位7和位6(或**I**和**F**)，他们分别控制**IRQ**和**FIQ**的中断屏蔽。**I**位置为1时，屏蔽**IRQ**, **F**位置为1时，屏蔽**FIQ**.



内容

- n 计算机体系结构基础
- n ARM架构
- n ARM处理器的工作状态
- n ARM体系结构的存储器格式
- n ARM处理器的工作模式
- n ARM体系结构的寄存器组织
- n **ARM处理器的异常处理**



异常 (Exceptions)

- n 当正常的程序执行流程发生暂时的停止时，称之为异常。
- n 在处理异常之前，当前处理器的状态必须保留，这样当异常处理完成之后，当前程序可以继续执行。
- n 处理器允许多个异常同时发生，它们将会按固定的优先级进行处理。



ARM所支持的异常类型

异常类型	具体含义
复位	当处理器的复位电平有效时，产生复位异常，程序跳转到复位异常处理程序处执行。
未定义指令	当ARM处理器或协处理器遇到不能处理的指令时，产生未定义指令异常。可使用该异常机制进行软件仿真。
软件中断	该异常由执行SWI指令产生，可用于用户模式下的程序调用特权操作指令。可使用该异常机制实现系统功能调用。
指令预取中止	若处理器预取指令的地址不存在，或该地址不允许当前指令访问，存储器会向处理器发出中止信号，但当预取的指令被执行时，才会产生指令预取中止异常。



ARM所支持的异常类型

异常类型	具体含义
数据中止	若处理器数据访问指令的地址不存在，或该地址不允许当前指令访问时，产生数据中止异常。
IRQ（外部中断请求）	当处理器的外部中断请求引脚有效，且CPSR中的I位为0时，产生IRQ异常。系统的外设可通过该异常请求中断服务。
FIQ（快速中断请求）	当处理器的快速中断请求引脚有效，且CPSR中的F位为0时，产生FIQ异常。



快速中断请求-FIQ

- n FIQ异常是为了支持数据传输或者通道处理而设计的。在ARM状态下，系统有足够的私有寄存器，从而可以避免对寄存器保存的需求，并减小了系统上下文切换的开销。
- n 若将CPSR的F位置为1，则会禁止FIQ中断，若将CPSR的F位清零，处理器会在指令执行时检查FIQ的输入。注意只有在特权模式下才能改变F位的状态。
- n 可由外部通过对处理器上的nFIQ引脚输入低电平产生FIQ。不管是在ARM状态还是在Thumb状态下进入FIQ模式，FIQ处理程序均会执行以下指令从FIQ模式返回：

SUBS PC,R14_fiq ,#4

该指令将寄存器R14_fiq的值减去4后，复制到程序计数器PC中，从而实现从异常处理程序中的返回，同时将SPSR_mode寄存器的内容复制到当前程序状态寄存器CPSR中。



外部中断请求-IRQ

- n IRQ异常属于正常的中断请求，可通过对处理器的nIRQ引脚输入低电平产生，IRQ的优先级低于FIQ，当程序执行进入FIQ异常时，IRQ可能被屏蔽。
- n 若将CPSR的I位置为1，则会禁止IRQ中断，若将CPSR的I位清零，处理器会在指令执行完之前检查IRQ的输入。注意只有在特权模式下才能改变I位的状态。
- n 不管是在ARM状态还是在Thumb状态下进入IRQ模式，IRQ处理程序均会执行以下指令从IRQ模式返回：

SUBS PC , R14_irq , #4

该指令将寄存器R14_irq的值减去4后，复制到程序计数器PC中，从而实现从异常处理程序中的返回，同时将SPSR_mode寄存器的内容复制到当前程序状态寄存器CPSR中。



ABORT（中止）

- n 产生中止异常意味着对存储器的访问失败。**ARM**微处理器在存储器访问周期内检查是否发生中止异常。
- n 中止异常包括两种类型：
 - 指令预取中止：发生在指令预取时。
 - 数据中止：发生在数据访问时。

当指令预取访问存储器失败时，存储器系统向**ARM**处理器发出存储器中止（**Abort**）信号，预取的指令被记为无效，但只有当处理器试图执行无效指令时，指令预取中止异常才会发生，如果指令未被执行，例如在指令流水线中发生了跳转，则预取指令中止不会发生。若数据中止发生，系统的响应与指令的类型有关。

- n 当确定了中止的原因后，**Abort**处理程序均会执行以下指令从中止模式返回，无论是在**ARM**状态还是**Thumb**状态：

SUBS PC, R14_abt, #4 ; 指令预取中止

SUBS PC, R14_abt, #8 ; 数据中止

以上指令恢复PC（从**R14_abt**）和**CPSR**（从**SPSR_abt**）的值，并重新执行中止的指令。



软件中断--SWI

- n 软件中断指令（**SWI**）用于进入管理模式，常用于请求执行特定的管理功能。
- n 软件中断处理程序执行以下指令从**SWI**模式返回，无论是在**ARM**状态还是**Thumb**状态：

MOV PC , R14_svc

以上指令恢复PC（从**R14_svc**）和CPSR（从**SPSR_svc**）的值，并返回到**SWI**的下一条指令。



未定义指令- Undefined Instruction

- n 当ARM处理器遇到不能处理的指令时，会产生未定义指令异常。采用这种机制，可以通过软件仿真扩展ARM或Thumb指令集。

- n 在仿真未定义指令后，处理器执行以下程序返回，无论是在ARM状态还是Thumb状态：

MOVS PC, R14_und

以上指令恢复PC（从R14_und）和CPSR（从SPSR_und）的值，并返回到未定义指令后的下一条指令。



对异常的响应

- n 当一个异常出现以后，**ARM**微处理器会执行以下几步操作：
 - o 将下一条指令的地址存入相应连接寄存器**LR**，以便程序在处理异常返回时能从正确的位置重新开始执行。
 - o 将**CPSR**复制到相应的**SPSR**中。
 - o 根据异常类型，强制设置**CPSR**的运行模式位。
 - o 强制**PC**从相关的异常向量地址取下一条指令执行，从而跳转到相应的异常处理程序处。



对异常的响应

R14_<Exception_Mode> = Return Link

SPSR_<Exception_Mode> = CPSR

CPSR[4:0] = Exception Mode Number

CPSR[5] = 0 ; 当运行于ARM工作状态时

If <Exception_Mode> == Reset or FIQ then

 ; 当响应FIQ异常时，禁止新的FIQ异常

 CPSR[6] = 1

 CPSR[7] = 1

PC = Exception Vector Address



从异常返回

- n 异常处理完毕之后，**ARM**微处理器会执行以下几步操作从异常返回：
 - o 将连接寄存器**LR**的值减去相应的偏移量后送到**PC**中。
 - o 将**SPSR**复制回**CPSR**中。
 - o 若在进入异常处理时设置了中断禁止位，要在此清除。

- n 可以认为应用程序总是从复位异常处理程序开始执行的，因此复位异常处理程序不需要返回。



异常向量 (Exception Vectors)

地 址	异 常	进入模式
0x0000,0000	复位	管理模式
0x0000,0004	未定义指令	未定义模式
0x0000,0008	软件中断	管理模式
0x0000,000C	中止（预取指令）	中止模式
0x0000,0010	中止（数据）	中止模式
0x0000,0014	保留	保留
0x0000,0018	IRQ	IRQ
0x0000,001C	FIQ	FIQ



异常优先级 (Exception Priorities)

- n 当多个异常同时发生时，系统根据固定的优先级决定异常的处理次序。

优先级	异 常
1 (最高)	复位
2	数据中止
3	FIQ
4	IRQ
5	预取指令中止
6 (最低)	未定义指令、SWI



应用程序中的异常处理

- n 当系统运行时，异常可能会随时发生，为保证在**ARM**处理器发生异常时不至于处于未知状态，在应用程序的设计中，首先要进行异常处理，采用的方式是在异常向量表中的特定位置放置一条跳转指令，跳转到异常处理程序，当**ARM**处理器发生异常时，程序计数器**PC**会被强制设置为对应的异常向量，从而跳转到异常处理程序，当异常处理完成以后，返回到主程序继续执行。



RE.ER嵌入式学院-服务品质保证

RE.ER嵌入式学院(亚嵌教育成都中心) ---

--- 有思想的培训者

--- 领先成都/西部第二家嵌入式培训机构三年

n 服务品质保证

- o 所有课程内容，保证完全掌握。如果存在没有掌握或消化不好的内容，免费学习，直到掌握为止；
- o 随时获得项目或实践机会；在获得额外资金收入外，还可不断积累项目经验；
- o 有机会享受每年至少12场相关技术的专题讲座；
- o 后续工作疑难问题帮助，辅助您在工作单位更快的发展。

亚嵌教育成都中心(RE.ER嵌入式学院)，有思想的培训者 --- <http://www.re-er.com.cn>



RE.ER嵌入式学院-课程介绍

- n 嵌入式Linux系统开发就业班(4个月)
- n 嵌入式系统开发就业班(6个月)
 - o 100%就业保证
 - o 适合人群：高校即将毕业学生。包括本科四年级学生、研究生三年级学生、专科三年级学生；
 - o 已经毕业但还没有找到工作的人员；
 - o 已经工作；但打算换到嵌入式或Linux行业的人员；比如之前从事过JAVA, .net, 单片机开发的人员；
- n 嵌入式系统开发高级(周末)班
 - o 100%学习效果保证
 - o 适合人群：想学习嵌入式Linux系统开发的在职人员
想学习嵌入式Linux系统开发的高校理工科学生
- n 嵌入式项目实训营
 - o 100%企业实际项目/产品开发
 - o 适合人群：想开发实际嵌入式/Linux项目或产品的人员
 - o 想积累嵌入式Linux项目或产品开发经验的人员；
 - o 想深入和牢固掌握嵌入式/Linux技术的人员；
- n

亚嵌教育成都中心(RE.ER嵌入式学院)，有思想的培训者 --- <http://www.re-er.com.cn>



RE.ER嵌入式学院

- ✦ 在成都及西部，RE.ER是第一个开设嵌入式LINUX课程的机构
- ✦ 在成都及西部，RE.ER是第一个开设嵌入式硬件课程的机构
- ✦ 在成都及西部，RE.ER是第一个开设FPGA课程的机构
- ✦ 在成都及西部，RE.ER是第一个开设音视频课程的机构
- ✦ 在成都及西部，RE.ER是第一个开设iphone开发课程的机构
- ✦ 在成都及西部，RE.ER是第一个以整个项目团队上课的机构
- ✦ 在成都及西部，RE.ER是第一个引入CMMI3 国际项目开发体系的机构
- ✦ 在成都及西部，RE.ER是第一个开设项目及产品开发班的机构
- ✦ 在成都及西部，RE.ER是第一个为高校进行嵌入式项目实训的机构