第二章 ARM处理器编程模型与ARM指令集

- 了解ARM微处理器的工作状态
- 掌握ARM处理器的7种工作模式
- 熟悉ARM的存储器组织
- ARM的异常中断处理流程

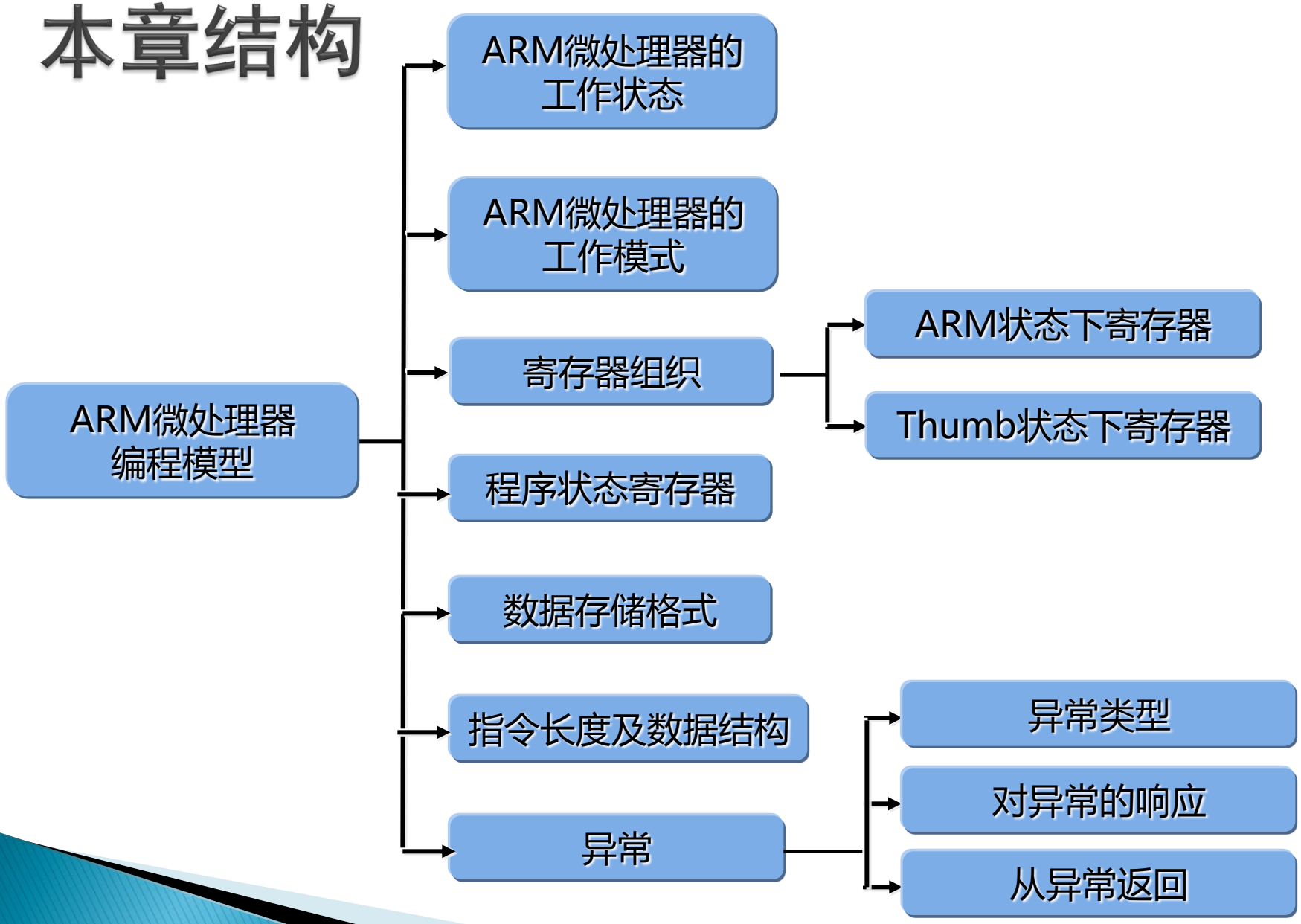
预习检查

- ARM微处理器有哪两种工作状态？
- ARM微处理器有哪几种工作模式？
- 程序状态寄存器有什么作用？
- ARM体系结构支持哪些异常？

本章目标

- 能够了解ARM微处理器的工作状态；
- 能够掌握ARM处理器的7种工作模式；
- 熟悉ARM的存储器组织；
- 能够了解ARM的异常中断处理。

本章结构


ARM微处理器编程模型
ARM微处理器的工作状态
ARM微处理器的工作模式
寄存器组织
ARM状态下寄存器
Thumb状态下寄存器
程序状态寄存器
数据存储格式
指令长度及数据结构
异常
异常类型
对异常的响应
从异常返回

理解CPU工作原理的关键概念

- 寄存器
- 译码单元与流水线
- 累加器与乘累加单元
- 寻址
- 异常、中断与处理器模式
- 虚拟内存
- 保护模式

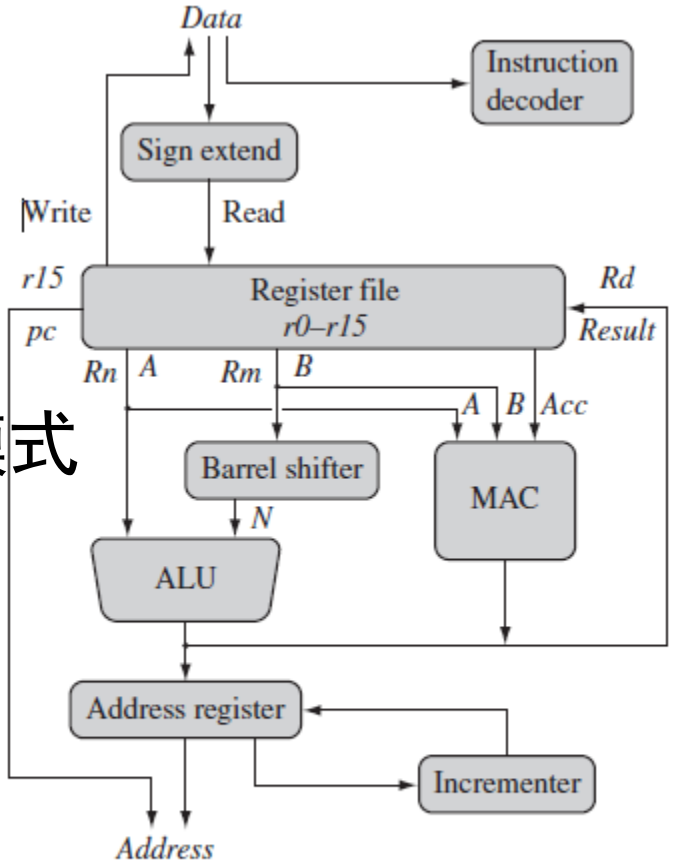
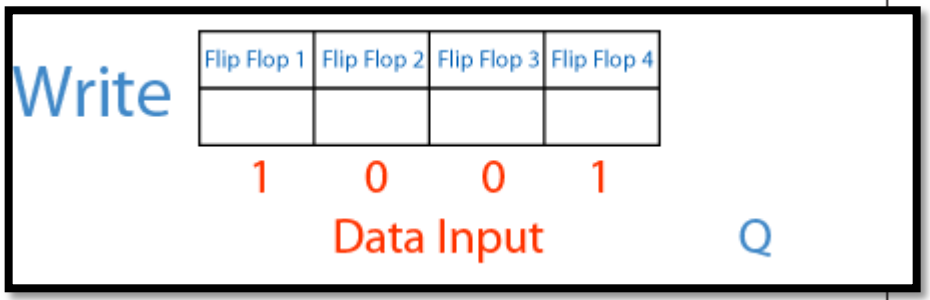


Figure 2.1 ARM core dataflow model.

寄存器

- 通用寄存器
- 桶状移位寄存器
- 地址寄存器



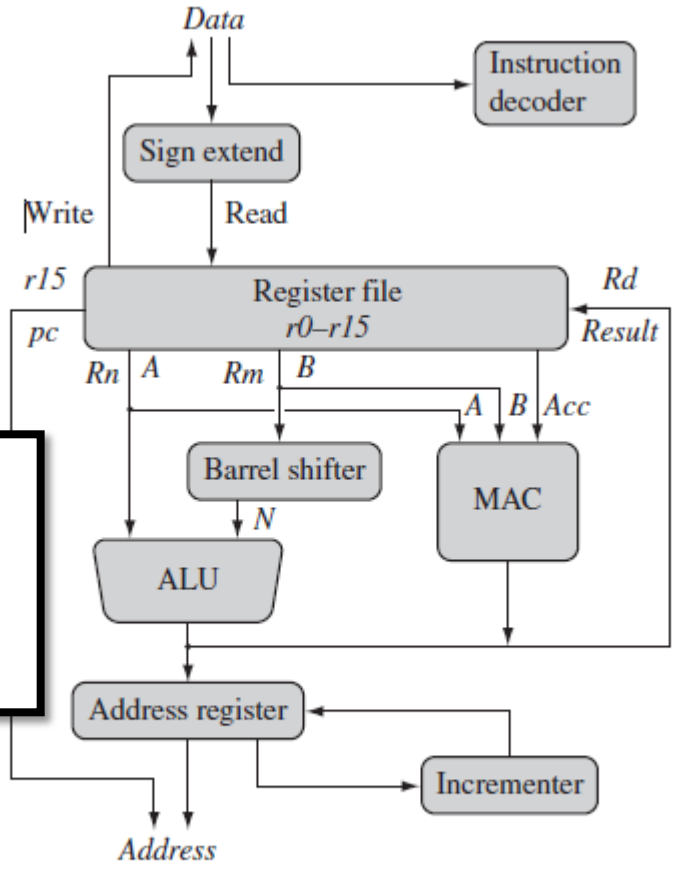
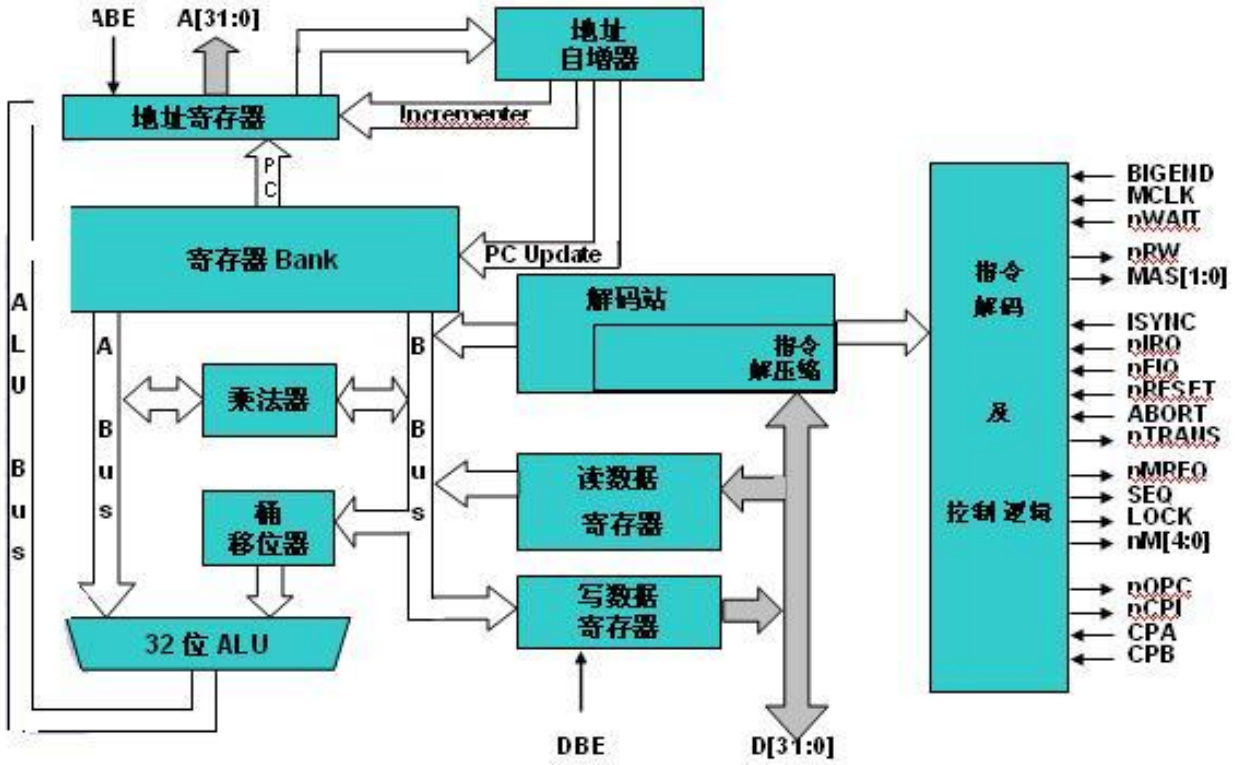


Figure 2.1 ARM core dataflow model.

译码单元与流水线



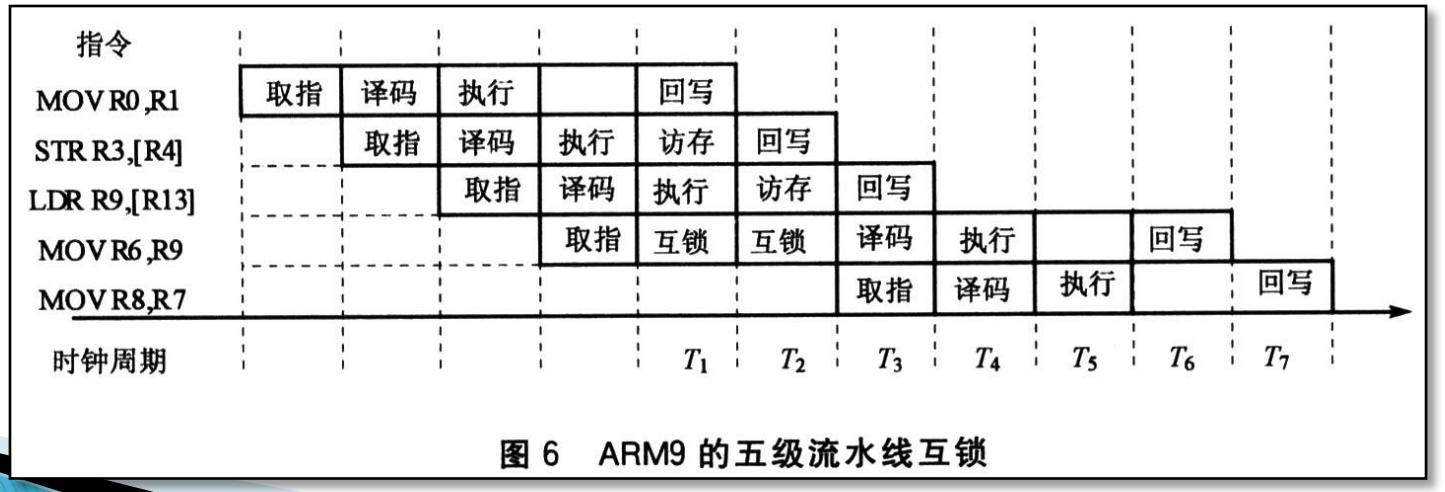
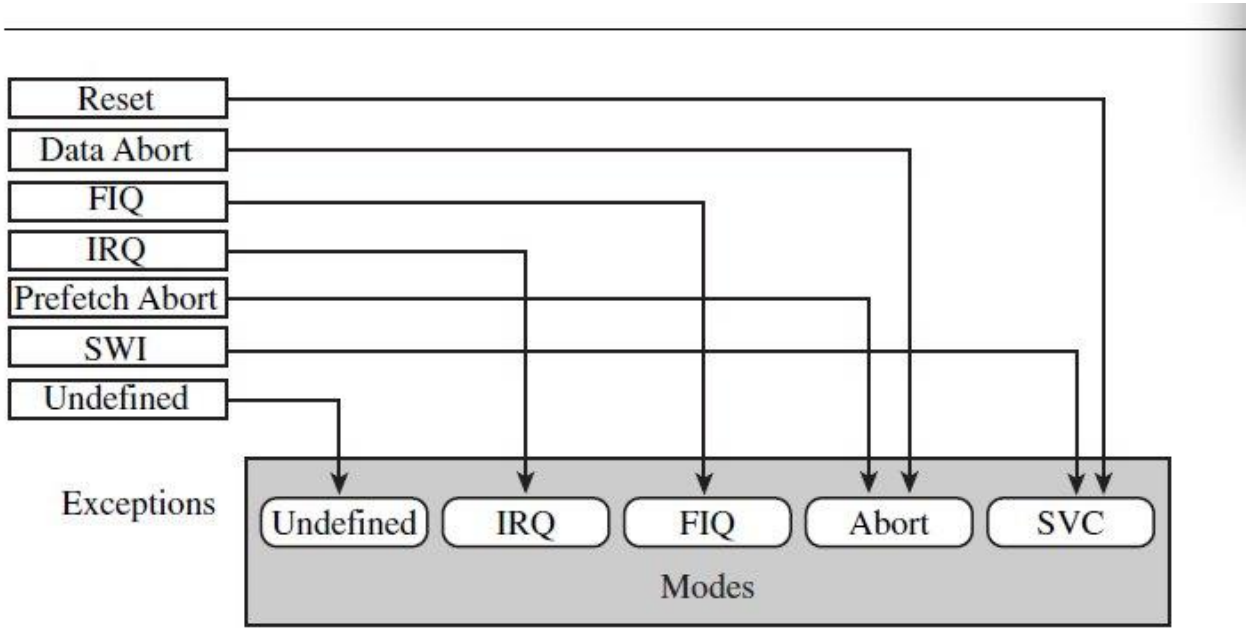


图 6　ARM9 的五级流水线互锁

异常、中断与ARM处理器模式

- CPU通过异常产生的中断进入相应的异常处理模式



Exceptions and associated modes.

虚拟内存

- 地址映射
- 分页

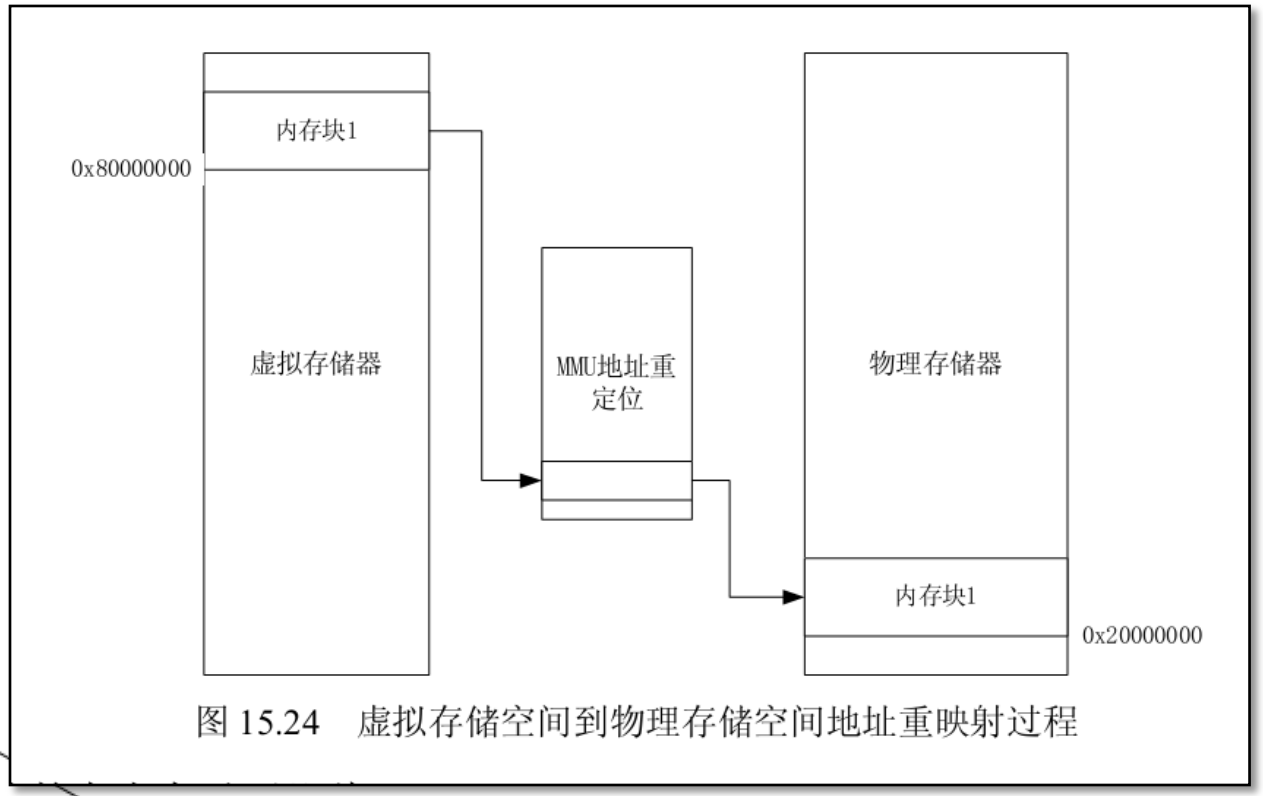
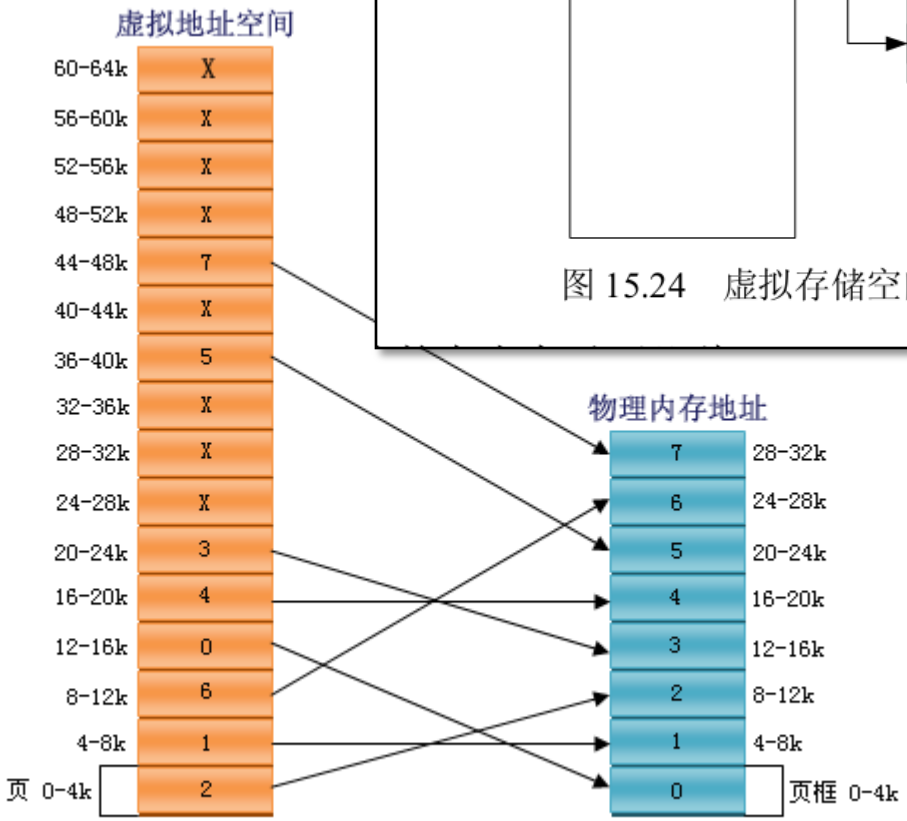


图 15.24　虚拟存储空间到物理存储空间地址重映射过程



页、页框映射关系图

ARM微处理器的工作状态

- 处理器状态:

  ARM9处理器内核使用V4T版本的ARM结构，具有两种操作状态：

  - ARM状态：32位，这种状态下执行的是字方式的ARM指令
  - Thumb状态：16位，这种状态下执行半字方式的ARM指令。

- 注意：两个状态之间的切换并不影响处理器模式或寄存器内容。

ARM微处理器的工作状态

- 处理器状态:

使用BX指令将ARM9内核的操作状态在ARM状态和Thumb状态之间进行切换，程序如下所示。

```
;从Arm状态切换到Thumb状态
   LDR    R0,=Lable+1
   BX     R0

;从Thumb状态切换到ARM状态
   LDR    R0,=Lable
   BX     R0
```

跳转地址标号

地址最低位为1，表示切换到Thumb状态

地址最低位为0，表示切换到ARM状态

寄存器工作模式

- 处理器7种工作模式：

处理器模式	说明	备注
用户 (usr)	正常程序执行模式	不能直接切换到其它模式
系统 (sys)	运行操作系统的特权任务	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)	操作系统保护模式	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ARM9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

寄存器工作模式

- 特权模式：

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)		
快中断 (fiq)		
中断 (irq)		
管理 (svc)		
中止 (abt)		
未定义 (und)		

除用户模式外，其它模式均为特权模式。ARM内部寄存器和一些片内外设在硬件设计上只允许（或者可选为只允许）特权模式下访问。此外，特权模式可以自由的切换处理器模式，而用户模式不能直接切换到别的模式。

寄存器工作模式

- 异常模式：

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)		
快中断 (fiq)		
中断 (irq)		
管理 (svc)		
中止 (abt)		
未定义 (und)		

这五种模式称为异常模式。它们除了可以通过程序切换进入外，也可以由特定的异常进入。当特定的异常出现时，处理器进入相应的模式。每种异常模式都有一些独立的寄存器，以避免异常退出时用户模式的状态不可靠。

寄存器工作模式

- 用户和系统模式：

处理器模式	说明	备注
用户 (usr)		
系统 (sys)		
快中断 (fiq)		
中断 (irq)		
管理 (svc)		
中止 (abt)		
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

这两种模式都不能由异常进入，而且它们使用完全相同的寄存器组。

系统模式是特权模式，不受用户模式的限制。操作系统在该模式下访问用户模式的寄存器就比较方便，而且操作系统的一些特权任务可以使用这个模式访问一些受控的资源。

阶段小结

- ARM9处理器支持的7种寄存器工作模式

ARM运行模式	描述
用户模式（usr）	正常的程序执行状态
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
特权模式（Supervisor，svc）	供操作系统使用的保护模式
数据访问终止模式(Abort，abt)	当数据或指令预取终止时进入该模式，可用于虚拟存储及存储保护
系统模式（System，sys）	运行具有特权的操作系统任务
未定义指令中止模式（Undefined，und）	当未定义的指令执行时进入该模式，可用于支持通过软件仿真硬件的协处理器

寄存器组织

- ARM9处理器内部寄存器组织：
  - 在ARM9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状态下的寄存器组织

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(V4)	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	R16(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

ARM状态下的寄存器组织

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						
		R8						R8_fiq
		R9						R9_fiq
		R10						R10_fiq
		R11						R11_fiq
		R12						R12_fiq
		svc			R13_abt	R13_und	R13_irq	R13_fiq
		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状态下的寄存器组织

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(V4)	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_irq	R14_fiq
	R15(PC)	R15						
状态寄存器	CPSR	CPSR						
	SPSR	无		SPSR_svc	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq

ARM状态下的寄存器组织

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)							
	R3(a4)							
	R4(v1)							
	R5(v2)							
	R6(v3)							
	R7(V4)							
	R8(V4)							R8_fiq
	R9(SB,v6)							R9_fiq
	R10(SL,v7)							R10_fiq
	R11(FP,v8)							R11_fiq
	R12(IP)							R12_fiq
	R13(SP)				abt	R13_und	R13_irq	R13_fiq
	R14(LR)				_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为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。

ARM状态下的寄存器组织

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
通和		R0						
		R1						
		R2						
		R3						
		R4						
		R5						
		R6						
		R7						
		R8						R8_fiq
		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。

ARM状态下的寄存器组织

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						
		R5						
		R6						
		R7						
		R8						R8_fiq
		R9						R9_fiq
		R10						R10_fiq
		R11						R11_fiq
		R12						R12_fiq
		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

ARM状态下的寄存器组织

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)							
	R1(a2)							
	R2(a3)							
	R3(a4)							
	R4(v1)							
	R5(v2)							
	R6(v3)							
	R7(V4)							
	R8(V4)	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。

ARM状态下的寄存器组织

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(V4)							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。一个用于用户和系统模式，其余5个分别用于5种异常模式。

2-3-1 ARM状态下的寄存器组织

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)							
	R3(a4)							
	R4(v1)							
	R5(v2)							
	R6(v3)							
	R7(V4)							
	R8(V4)							R8_fiq
	R9(SB,v6)							R9_fiq
	R10(SL,v7)							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。

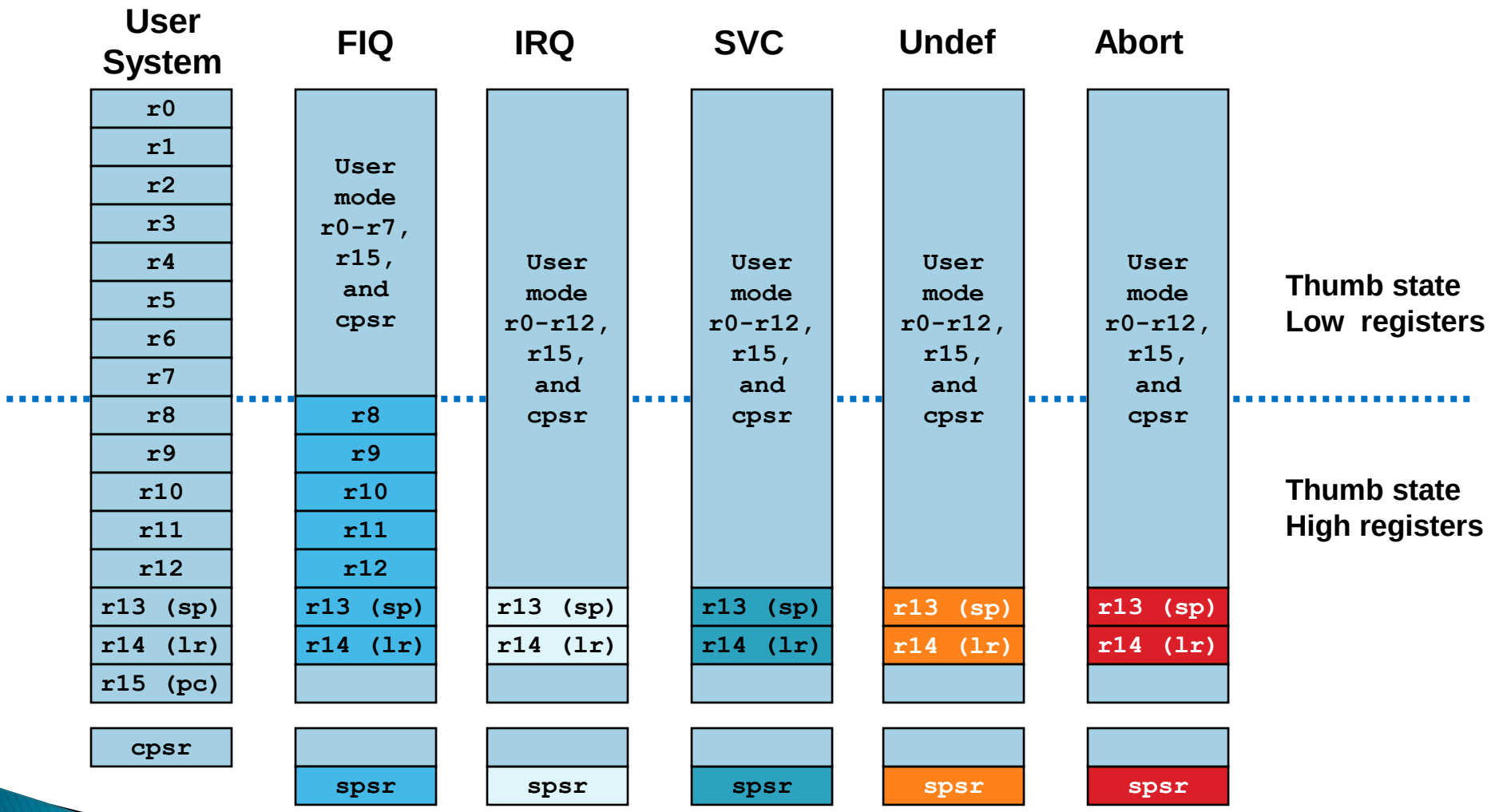
2-3-1 ARM状态下的寄存器组织

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(V4)							
	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

R14为链接寄存器（LR），在结构上有两个特殊功能：

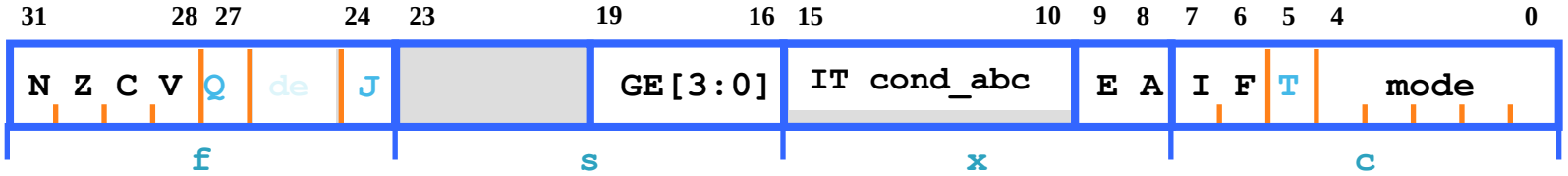
- 在每种模式下，模式自身的R14版本用于保存子程序返回地址；
- 当发生异常时，将R14对应的异常模式版本设置为异常返回地址（有些异常有一个小的固定偏移量）。

2-3-2 Thumb状态下的寄存器组织



Note: System mode uses the User mode register set

程序状态寄存器



- T Bit
  - T = 0: Processor in ARM state
  - T = 1: Processor in Thumb state
  - Introduced in Architecture 4T
- Mode bits
  - Specify the processor mode

- Condition code flags
  - N = **N**egative result from ALU
  - Z = **Z**ero result from ALU
  - C = ALU operation **C**arried out
  - V = ALU operation o**V**erflowed
- Sticky Overflow flag – Q flag
  - Architecture 5TE and later only
  - Indicates if saturation has occurred
- J bit
  - Architecture 5TEJ and later only
  - J = 1: Processor in Jazelle state
- Interrupt Disable bits.
  - I = 1: Disables the IRQ
  - F = 1: Disables the FIQ

阶段小结

- ARM9处理器内部寄存器简介：
  - 在ARM9处理器内部有37个用户可见的寄存器。
  - 31个通用32位寄存器；
  - 6个状态寄存器。

ARM体系结构的数据存储格式

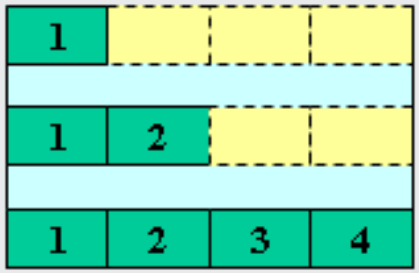
- 处理器用于存储数据的方式有两种，分别为大字节序格式和小字节序格式：
  - 大字节序格式：字数据的高字节存储在低地址中，而字数据的低字节则存放在高地址中。
  - 小字节序格式：字数据的高字节存储在高地址中，而字数据的低字节则存放在低地址中。

注意：ARM体系结构较新的版本对这两种数据存储方式都支持。某些较老的版本只支持小字节序存储方式，编程的时候需要注意。

指令长度及数据类型

- ARM9处理器指令长度：
  - 在ARM状态下，ARM微处理器的指令长度是32位；在Thumb状态下，指令长度为16位。
- ARM9处理器数据类型：
  - ARM微处理器中支持字节（8位）、半字（16位）、字（32位）三种数据类型，其中，字需要4字节对齐（地址的低两位为0）、半字需要2字节对齐（地址的最低位为0）。

- 字节 8位
- 半字 16位（必须分配为占用两个字节）
- 字 32为（必须分配为占用4各字节）



ARM微处理器的数据存储格式

- ARM9处理器支持的数据类型：

注意：

- V4版本之后的ARM结构都支持这3种结构（包括V4版本），而以前的版本只支持字节和字；
- 当数据类型定义为无符号型时，N位数据值使用正常的二进制格式表示范围为$0\sim2^{N-1}$的非负整数；
- 当数据类型定义为有符号型时，N位数据值使用2的补码格式表示范围为$-2^{N-1}\sim2^{N-1}-1$

ARM微处理器的数据存储格式

- ARM9处理器支持的数据类型：

注意：

- 所有数据操作，例如ADD，都以字为单位；
- 装载和保存指令可以对字节、半字和字进行操作，当装载字节或半字时自动实现零扩展或符号扩展；
- ARM指令的长度刚好是一个字（分配为占用4个字节），Thumb指令的长度刚好是半字（占用2个字节）。

异常

- 简介：
  - 只要正常的程序流被暂时中止，处理器将进入异常模式。例如响应一个来自外设的中断。在处理异常之前，ARM9内核保存当前的处理器状态（CPSR->SPSR），这样当处理程序结束时可以恢复执行原来的程序（SPSR->CPSR）。
  - 如果同时发生两个或更多异常，那么将按照固定的顺序来处理异常，详见教材“异常优先级”部分。

ARM所支持的异常

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。
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。

ARM所支持的异常

复位异常:

- 当nRESET信号被拉低时（一般外部复位引脚电平的变化和芯片的其它复位源会改变这个内核信号），ARM9处理器放弃正在执行的指令。
- 在复位后，除PC和CPSR之外的所有寄存器的值都不确定。

ARM所支持的异常

- 复位异常:

  当nRESET信号再次变为高电平时，ARM处理器执行下列操作：

  - 强制CPSR中的M[4:0]变为b10011（管理模式）；
  - 置位CPSR中的I和F位；
  - 清零CPSR中的T位；
  - 强制PC从地址0x00开始对下一条指令进行取指；
  - 返回到ARM状态并恢复执行 。

ARM所支持的异常

- 未定义的指令异常:
  - 当ARM9处理器遇到一条自己和系统内任何协处理器都无法处理的指令时，ARM9内核执行未定义指令陷阱。软件可使用这一机制通过模拟未定义的协处理器指令来扩展ARM指令集。

注：ARM9处理器完全遵循ARM结构V4T，可以捕获所有分类未被定义的指令位格式。

ARM所支持的异常

- 未定义的指令异常:

在模拟处理了失败的指令后，陷阱程序执行下面的指令：

```
MOVS    PC,R14_und
```

这个动作恢复了PC并返回到未定义指令之后的下一条指令。

ARM所支持的异常

- 软件中断异常:

使用软件中断(SWI)指令可以进入管理模式，通常用于请求一个特定的管理函数。SWI处理程序通过执行下面的指令返回：

```
MOVS    PC,R14_svc
```

这个动作恢复了PC并返回到SWI之后的指令。SWI处理程序读取操作码以提取SWI函数编号。

ARM所支持的异常

- 预取指中止异常:

当发生预取中止时，ARM9内核将预取的指令标记为无效，但在指令到达流水线的执行阶段时才进入异常。如果指令在流水线中因为发生分支而没有被执行，中止将不会发生。

在处理中止的原因之后，不管处于哪种处理器操作状态，处理程序都会执行下面的指令恢复PC和CPSR并重试被中止的指令：

```
SUBS    PC,R14_abt,#4
```

ARM所支持的异常

- 数据中止异常:
  - 当发生数据中止后，根据产生数据中止的指令类型作出不同的处理：
    - 数据转移指令（LDR、STR）回写到被修改的基址寄存器。中止处理程序必须注意这一点；
    - 交换指令（SWP）中止好像没有被执行过一样（中止必须发生在SWP指令进行读访问时）；

ARM所支持的异常

- 数据中止异常:
- 在修复产生中止的原因后，不管处于哪种处理器操作状态，处理程序都必须执行下面的返回指令，重试被中止的指令：

```
SUBS    PC,R14_abt,#8
```

ARM所支持的异常

- IRQ(Interrupt Request):
  - 中断请求（IRQ）异常是一个由nIRQ输入端的低电平所产生的正常中断（在具体的芯片中，nIRQ由片内外设拉低，nIRQ是内核的一个信号，对用户不可见）。IRQ的优先级低于FIQ。对于FIQ序列它是被屏蔽的。任何时候在一个特权模式下，都可通过置位CPSR中的I 位来禁止IRQ。
  - 不管异常入口是来自ARM状态还是Thumb状态，IRQ处理程序都会通过执行下面的指令从中断返回：

```
SUBS    PC,R14_irq,#4
```

ARM所支持的异常

- FIQ(Fast Interrupt Request):
  - 快速中断请求(FIQ)适用于对一个突发事件的快速响应，这得益于在ARM状态中，快中断模式有8个专用的寄存器可用来满足寄存器保护的需要（这可以加速上下文切换的速度）。
  - 不管异常入口是来自ARM状态还是Thumb状态，FIQ处理程序都会通过执行下面的指令从中断返回：

```
SUBS    PC,R14_irq,#4
```

  - 在一个特权模式中，可以通过置位CPSR中的F位来禁止FIQ异常。

对异常的响应(进入异常)

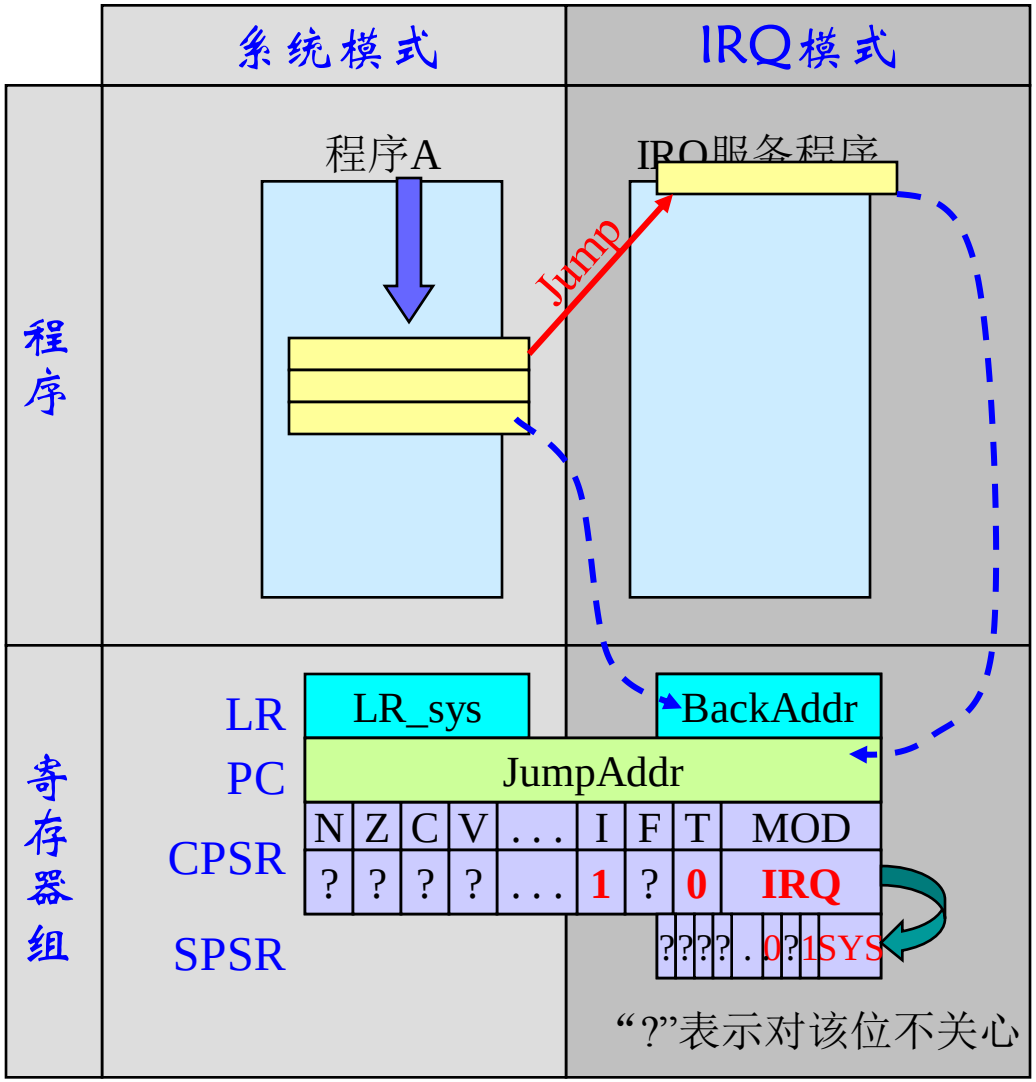
在异常发生后，ARM9内核会作以下工作:

- 在适当的LR中保存下一条指令的地址，当异常入口来自：
  - ARM状态，那么ARM9将当前指令地址加4或加8复制（取决于异常的类型）到LR中；
  - 为Thumb状态，那么ARM9将当前指令地址加2、4或加8 （取决于异常的类型）复制到LR中；异常处理器程序不必确定状态。
  - 如果同时发生两个或更多异常，那么将按照异常优先级的顺序来处理异常。
- 将CPSR复制到适当的SPSR中；
- 将CPSR模式位强制设置为与异常类型相对应的值；
- 强制PC从相关的异常向量处取指。
- ARM9内核在处理中断异常时置位中断禁止标志，这样可以防止不受控制的异常嵌套。

对异常的响应(进入异常)

进入异常过程：

- 1. 程序在系统模式下运行用户程序，假定当前处理器状态为Thumb状态、允许IRQ中断；
  - 将CPSR寄存器内容存入IRQ模式的SPSR寄存器
  - 置位I位（禁止IRQ中断）
  - 清零T位（进入ARM状态）
  - 设置MOD位，切换处理器模式至IRQ模式
  - 将下一条指令的地址存入IRQ模式的LR寄存器
  - 将跳转地址存入PC，实现跳转



从异常返回(退出异常)

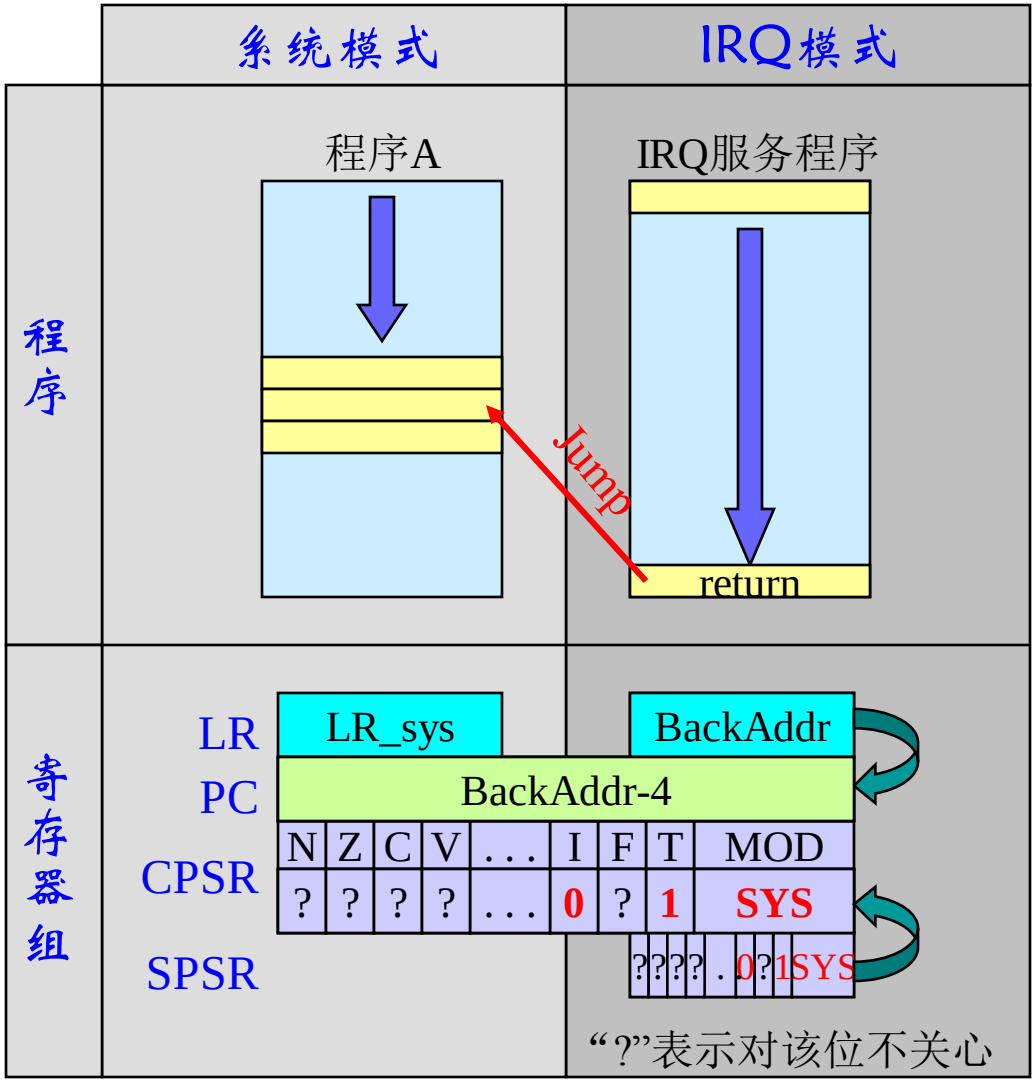
当异常结束时，异常处理程序必须:

- 将LR(R14)中的值减去偏移量后存入PC，偏移量根据异常的类型而有所不同;
- 将SPSR的值复制回CPSR;
- 清零在入口置位的中断禁止标志。

注意：恢复CPSR的动作会将T、F和I位自动恢复为异常发生前的值。

从异常返回(退出异常)

- 在异常处理结束后，异常处理程序完成以下动作：
  - 将SPSR寄存器的值复制回CPSR寄存器；
  - 将LR寄存的值减去一个常量后复制到PC寄存器，跳转到被中断的用户程序。



阶段小结

- ARM所支持的异常类型
- 对异常的响应
- 从异常返回

阶段练习

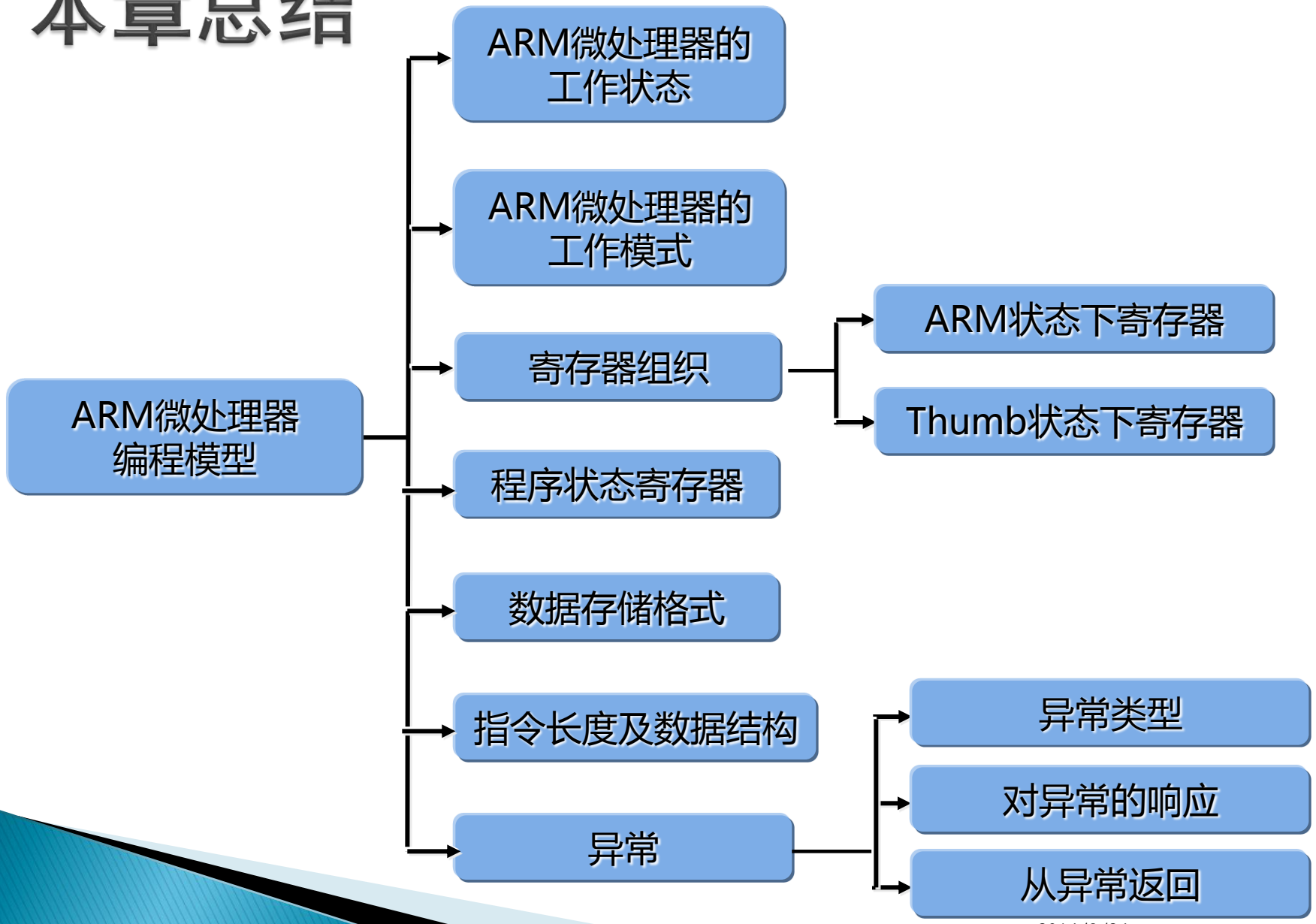
☺ 3. ARM状态下各组寄存器是如何组织的？

☺ 4. 程序状态寄存器各位的含义是什么？

☺ 5. ARM体系结构支持哪些异常？

☺ 6. ARM微处理器对异常是如何响应和返回的？

本章总结


ARM微处理器
编程模型
ARM微处理器的
工作状态
ARM微处理器的
工作模式
寄存器组织
ARM状态下寄存器
Thumb状态下寄存器
程序状态寄存器
数据存储格式
指令长度及数据结构
异常
异常类型
对异常的响应
从异常返回

阶段练习

- ☺ ARM哪两种状态？如何进行状态的切换？
- ☺ ARM处理器有那7种模式？系统复位的时候处于那种模式？
- ☺ 寄存器安什么标准分组的？R15,R14,R13的特殊用途是什么？
- ☺ 程序状态寄存器最低8位 的含义是什么？
- ☺ ARM体系结构支持哪些异常？与处理器的模式有何关系？
- ☺ ARM微处理器对异常是如何响应和返回的？