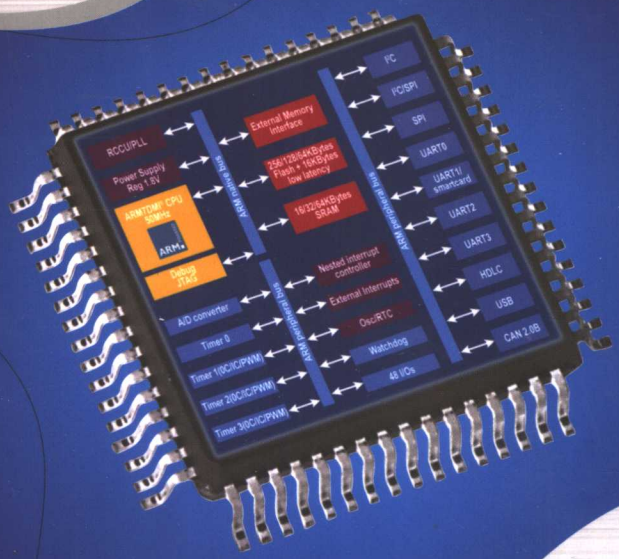

沈建华
姜　宁　编著

STR71x系列ARM微控制器原理与实践

CD-ROM INCLUDED

北京航空航天大学出版社

STR71x 系列
ARM 微控制器原理与实践

沈建华　姜　宁　编著

北京航空航天大学出版社

内 容 简 介

本书详细介绍了意法半导体(STMicroelectronics,简称 ST)STR710 系列 ARM 微控制器的原理、结构、资源和开发使用方法。全书共 6 章:第 1 章介绍了一些必需的 ARM 基础知识;第 2 章介绍了 ST ARM 产品系列;第 3 章详细介绍了 STR710 系列的片内资源和编程接口;第 4 章简单介绍了 ST ARM 的软硬件开发工具;第 5 章详细介绍了 STR710 的软件库函数和使用方法;第 6 章为 μC/OS-II 在 STR710 上的移植及使用(放入光盘)。本书所附光盘包含了 ARM 开发调试软件、开发工具介绍、STR710 开发板硬件资料、所有外设接口的例子程序(源码)、STR710 原版的数据手册、应用参考,以及第三方软件、工具等。

本书可作为 STR7 微控制器开发技术人员的参考手册,也可用于 ARM 相关应用、培训课程的参考书。

图书在版编目(CIP)数据

STR71x 系列 ARM 微控制器原理与实践/沈建华等编著.
北京:北京航空航天大学出版社,2006.9
ISBN 7-81077-794-7

Ⅰ.S… Ⅱ.沈… Ⅲ.微处理器,ARM
Ⅳ.TP332

中国版本图书馆 CIP 数据核字(2006)第 107052 号

© 2006,北京航空航天大学出版社,版权所有。
未经本书出版者书面许可,任何单位和个人不得以任何形式或手段复制或传播本书及其所附光盘内容。
侵权必究。

STR71x 系列 ARM 微控制器原理与实践
沈建华 姜 宁 编著
责任编辑 詹艳玲 王 松
北京航空航天大学出版社出版发行
北京市海淀区学院路 37 号(100083) 发行部电话:010-82317024 传真:010-82328026
http://www.buaapress.com.cn E-mail:bhpress@263.net
涿州市新华印刷有限公司印装 各地书店经销
*
开本:787×1092 1/16 印张:27.25 字数:698 千字
2006 年 9 月第 1 版 2006 年 9 月第 1 次印刷 印数:6 000 册
ISBN 7-81077-794-7 定价:42.00 元(含光盘 1 张)

前 言

近年来，基于 ARM 核的微控制器应用日趋普及，随着芯片价格的不断下降，其性价比已远远超过了许多传统的 8 位和 16 位单片机。使用 ARM 微控制器，具有架构统一、芯片选择范围广、开发工具一致、网上资源丰富、软件可复用等优势。

意法半导体(STMicroelectronics，简称 ST)是全球著名的 IC 设计制造商，其产品线非常完整，几乎包含了所有半导体产品。2005 年初我们与 ST 合作，开始使用、推广 STR7 系列 ARM 微控制器。在合作推广过程中，双方觉得有必要出版一本 STR71x 的中文技术参考书，以便于国内的嵌入式系统开发工程师使用 STR71x 系列 ARM 微控制器。

综观目前多种 ARM 微控制器，ST ARM 微控制器主要有以下特点和优势：

产品线广

有基于 ARM7TDMI 核的 STR71x 系列，基于 ARM7TDMI(-S)核的 STR75x 系列，以及基于 ARM966E 核的 STR91x 系列(世界上首款基于 ARM9 的 SoC MCU，此书出版时刚发布)。还有颇具特色的 STR73x 系列(ARM7TDMI 核)，它是单 5 V 供电的真正单片 ARM 微控制器，外设丰富、I/O 口线多(144 脚)，非常适合工业应用，以及对原来 5 V 系统的升级。用户可以根据各种应用需求，选择合适的 ST ARM 系列产品。

外设丰富

ST ARM 微控制器集成了常用的各种外设，通信接口特别丰富，如 4 个 UART 异步串口、BSPI、I^2C、Timer、Watch Dog、RTC、PLL、12 位 ADC、PWM、USB、CAN、HDLC、Smart Card、10/100M 网络等，弥补了其他许多 ARM 微控制器的不足，可以大大简化系统硬件设计，降低系统成本。

内带存储器容量大

STR7 系列内嵌(64～256 KB)＋16 KB Flash，16～64 KB SRAM，可以承载较大的软件，满足日趋复杂的嵌入式应用系统需求。另外，其分块 Flash 结构，可以在 Flash 中执行程序时，改写另一块 Flash 的内容，给用户带来很大的灵活性。

功耗控制灵活

STR7 系列比其他 ARM 微控制器具有更多的功耗控制方式，如 SLOW、WAIT、STOP 和 STANGBY 等，与合理的软件设计配合，可以满足一些对功耗要求较高的应用场合。

完整的软件库支持

这是与其他 ARM 微控制器相比，使用 ST 系列 ARM 微控制器最具优势之处。ST 提供了 STR 微控制器所有外设的软件库支持，使用户不必关心许多外设寄存器的具体定义，只要

调用相关的库函数，设置一些参数，就可以使用这些硬件外设资源。这极大地方便了用户使用，降低了开发人员的硬件技术要求，可以加快软件开发进程。其最新的 STR91x 系列（ARM966E 核），进一步提供了图形化的外设配置工具，体现了一种全新的微控制器使用、编程方法。

由于篇幅所限，本书在最后定稿时，裁减了部分内容，主要是第 3 章的 STR7 电气特性、第 4 章的开发工具使用方法和第 5 章的 STR7 开发板硬件介绍和第 6 章的 μC/OS-II 在 STR710 上的移植。为保证内容的完整性，便于读者了解相关内容，裁减的部分放在本书所附的光盘中。

本书由华东师范大学计算机系嵌入系统实验室负责编写，最后由我们和 ST 相关技术人员共同审核。其中第 2、3、5 章内容主要参考 ST 公司的 STR71x datasheet、STR71x Microcontroller Reference Manual、STR71x Software Library User Manual 等，最新的资料可以从 ST 的网站（mcu. st. com/mcu）下载。在此书的编写、审核过程中，我的研究生张群忠、吴红举、崔文华、杨海波、梁丹、陈海东、姜立娣、朱航宇等参与了部分资料整理、代码验证等工作；ST 技术经理梁平、市场经理曹锦东、技术工程师黄裕军、金尔雅等，在样片、资料、书稿审核、技术支持等方面做了很多工作；北航出版社胡晓柏编辑、深圳英蓓特信息技术有限公司徐光峰先生、上海沁科信息技术有限公司王永虹先生等也给予了很多关心和支持。在此，谨向他（她）们表示衷心的感谢。

由于时间仓促及水平所限，以及 ST 的技术文档本身也在不断修正，错误及不妥之处在所难免，欢迎各位读者批评指正。我也会在我们实验室的网站（www. emlab. net）上及时发布相关信息，欢迎访问并相互交流。

沈建华

2006 年 8 月于

华东师范大学

此为试读，需要完整PDF请访问：www.ertongbook.com

目 录

第 3 章 STR71x 系列微控制器的内部资源

第1章

ARM及ST微控制器概述

1.1 ARM处理器体系结构

1.1.1 ARM介绍

ARM(Advanced RISC Machines)公司是微处理器行业的一家知名企业，该公司设计了许多高性能、廉价、低功耗的RISC处理器IP核，及其相关技术和软件，并将相应的知识产权出售给世界各地的知名半导体、软件和OEM厂商。同时，ARM架构又是面向低成本预算设计的第一款RISC微处理器。

在许多成功的32位嵌入式系统中，ARM处理器都是其核心组成部分。从1985年第1个ARM1原型诞生至今，ARM经历了漫长的探索之路。到2000年年底，已经有超过10亿个ARM处理器被销售到了世界各地。当前，采用ARM技术IP核的微处理器遍及汽车、消费电子、成像、工业控制、海量存储、网络、安保和无线等各类产品市场。基于ARM技术的处理器占据了32位RISC芯片75%的市场份额。ARM公司已经成为了全球性的嵌入式RISC标准缔造者。ARM的成功建立在简单而又强大的原始设计上，随着技术的不断创新，这个设计也在不断改进。ARM的内核并不是单一的，而是一个遵循相同设计理念和使用相同指令集的内核系列。

最成功的ARM内核之一——ARM7TDMI，具有最高120 Dhrystone MIPS的性能、较高的代码密度和低功耗等特性，使它成为移动嵌入式设备的最佳选择。本书主要介绍的STR71x系列微控制器，就是基于ARM7TDMI内核设计的。

1.1.2 ARM体系结构版本

迄今为止，ARM体系结构共定义了6个版本，版本号分别为v1～v6。从版本v1到版本v6，ARM体系结构的指令集功能不断扩大。同时，各版本中还有一些变种，这些变种定义了该版本指令集中不同的功能。ARM系列处理器采用的实现技术各不相同，性能差别很大，应用场合也有所不同，但是只要它们支持相同的ARM体系结构版本，基于它们的应用软件就是兼容的。此外，ARM公司精心规划体系结构发展过程，使得在较早的架构版本上编写的代码也可以在后继版本上运行，实现向下兼容。

自从第1个ARM处理器于1985年问世以来，ARM体系结构一直在发展。表1.1列出了从最初的体系结构版本到现在的版本v6的逐步改进过程。

表 1.1 ARM 体系结构版本

版 本	对应的 ARM 内核实例	功能描述
ARMv1	ARM1	第一个 ARM 处理器； 26 位寻址
ARMv2	ARM2	32 位乘法器； 32 位协处理器支持
ARMv2a	ARM3	片上 cache； 原子交换指令； 协处理器 15 用于 cache 管理
ARMv3	ARM6 和 ARM7DI	32 位寻址； 独立的 cpsr 和 spsr； 新增 und 和 abt 模式； MMU 支持虚拟存储
ARMv3M	ARM7M	有符号和无符号长乘法指令
ARMv4	StrongARM	有符号和无符号半字与字节的 load-store 指令； 新增 sys 模式； 不再支持 26 位寻址方式
ARMv4T	ARM7TDMI 和 ARM9T	Thumb
ARMv5TE	ARM9E 和 ARM10E	ARMv4T 的超集； 增加 ARM 与 Thumb 之间切换的额外指令； 增强乘法指令； 额外的 DSP 类型指令； 快速乘累加
ARMv5TEJ	ARM7EJ 和 ARM926EJ.	JAVA 加速
ARMv6	ARM11	改进的多处理器指令； 边界不对齐和混合大小端数据的处理； 新的多媒体指令

1.1.3 ARM7TDMI 处理器内核

ARM 公司设计了许多处理器核，根据使用内核的不同划分为 ARM7、ARM9、ARM10 和 ARM11 等系列。后缀数字 7、9、10 和 11 表示不同的内核设计。数字的升序说明性能和复杂度的提高。其中，ARM7TDMI 是当前非常流行的一款 ARM 内核，已经被用在许多 32 位嵌入式微处理器上。ARM7TDMI 提供了非常好的性能功耗比，是第一个包括 Thumb 指令集、快速乘法指令和嵌入式 ICE 调试技术的内核，这使得 ARM7 成为嵌入式系统的理想选择。ARM7TDMI(-S)是 ARM7TDMI 的可综合版本。

1.1.3.1 内核应用、特点及框图

ARM7TDMI 内核是一个 32 位的嵌入式 RISC 处理器内核，它经过硬件宏单元优化，并充分考虑了性能、功耗和芯片面积等因素，达到最佳组合。采用 ARM7TDMI 内核，芯片生产厂

此为试读，需要完整PDF请访问：www.ertongbook.com

商可以方便地建立较小的芯片面积、较低的功耗和较高性能的嵌入式 SoC。

1. 应用领域

- 个人音频系统；
- 无线手持设备；
- 寻呼机；
- 喷墨打印机；
- 数码照相机。

2. 特　性

- 支持 32 位/16 位 RISC 体系结构(ARM v4T)；
- 高性能和高灵活度的 32 位 ARM 指令集；
- 高代码密度的 16 位 Thumb 指令集；
- 指令和数据共用 32 位总线；
- 3 级流水线；
- 32 位 ALU；
- 超小圆片面积，较低功耗；
- 完全静态操作；
- 支持协处理器接口；
- 支持扩展调试功能：
  — EmbeddedICE-RT 实时调试单元；
  — JTAG 接口单元；
  — 拥有接口与 Embedded Trace Macrocell(ETM)直接连接。

3. 优　势

- 支持 ARM 和 Thumb 指令混和编程，适用于对速度和代码密度有较高要求的嵌入式应用；
- 数据和代码共用数据总线简化了 SoC 集成过程；
- ARM7TDMI(-S)内核代码可以被 ARM9、ARM9E 和 ARM10 内核兼容；
- 较小的圆片面积降低了 SoC 面积、成本以及功耗；
- 静态、低功耗设计适用于电池供电器件；
- 使用协处理器可以将指令集进行扩展，以适应特殊应用；
- 拥有 EmbeddedICE-RT 实时调试单元和 ETM 单元，支持实时调试设备。

表 1.2 是 ARM 公司 2005 年提供的使用不同工艺实现的 ARM7TDMI 内核性能参数列表。

表 1.2　ARM7TDMI 内核性能参数

工　艺	0.13 μ 工艺处理器	0.18 μ 工艺处理器
带 cache 内核面积/mm²		
无 cache 内核面积/mm²	0.26	0.53
频率/MHz	133	100～80
带 cache 典型功耗/(mW/MHz)		
无 cache 典型功耗/(mW/MHz)	0.06	

图 1.1 是 ARM7TDMI(-S)的内核框图，ARM7TDMI(-S)的内核功能图如图 1.2 所示。

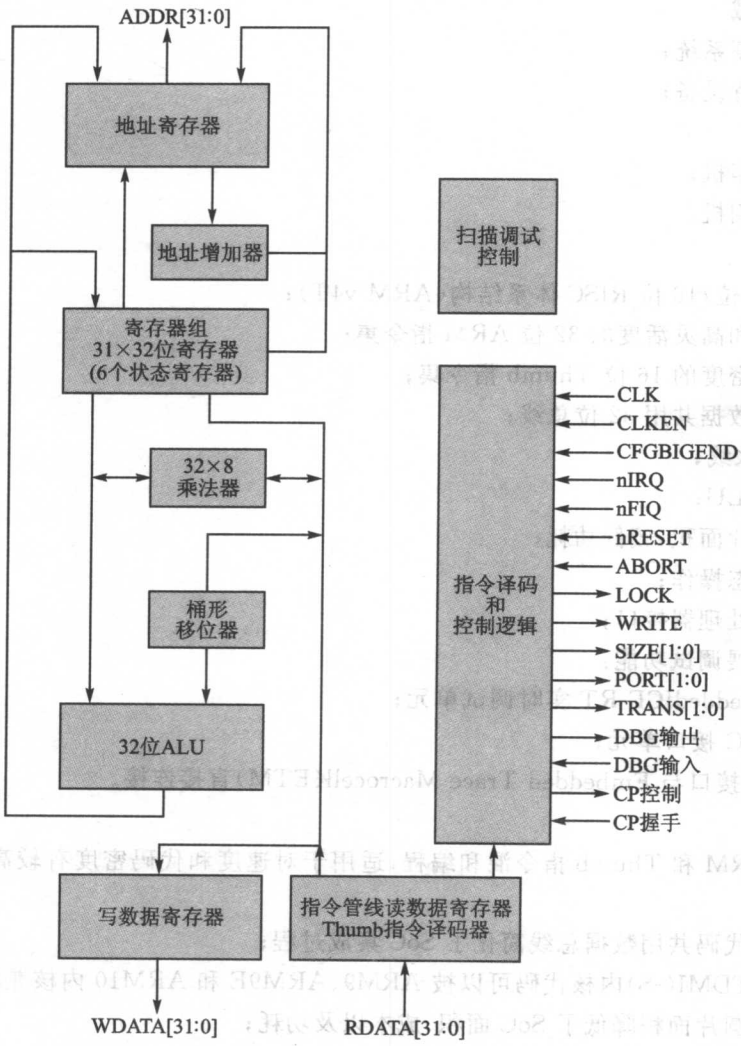

图 1.1　ARM7 内核框图

1.1.3.2　流水线(pipeline)

ARM7 内核是冯·诺伊曼体系结构，数据和指令使用同一条总线。ARM7 CPU 的核心是一条指令流水线，流水线用来处理从程序存储器中取出的指令。ARM7 使用三级流水线结构，如图 1.3 所示，执行 ARMv4 指令集(ISA)。

三级流水线结构是流水线技术中较简单的实现形式，并且不会出现多级流水线中会遇到的写前读(read-before-write)等数据相关问题。流水线的三级结构是由硬件实现的，在执行一条指令的同时，译码另一条指令并读取第三条指令。流水线技术大大提高了 CPU 的指令吞吐率，使得大多数 ARM 指令可以在一个时钟周期内完成。在执行无跳转的线性代码时，流水线的效率最高。当出现分支时，流水线将被清空，需要重新填满才能恢复到全速执行。后面我们将看到 ARM 指令集中许多有趣的特性，这些特性使得代码中小的跳转可以被平滑掉，从而

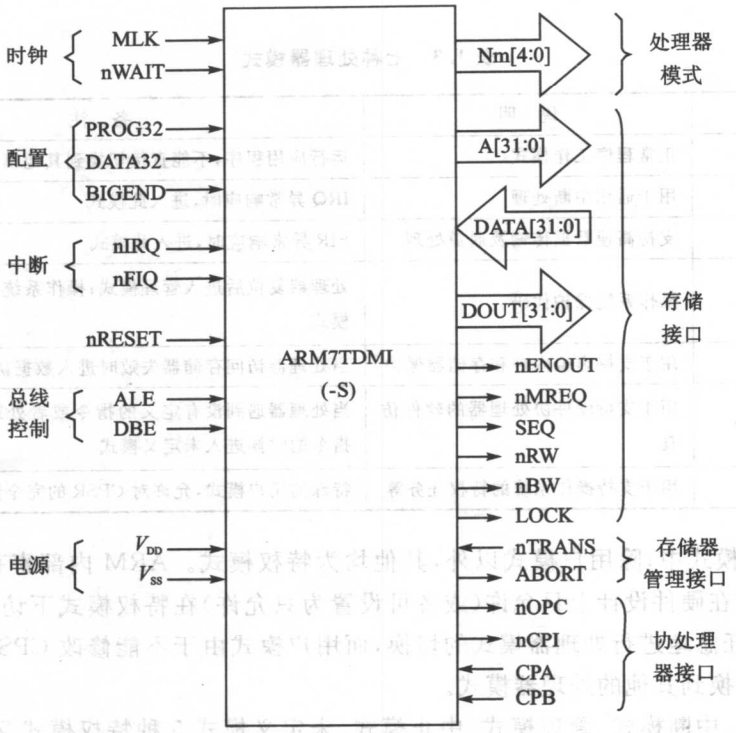

图1.2 ARM7内核功能图

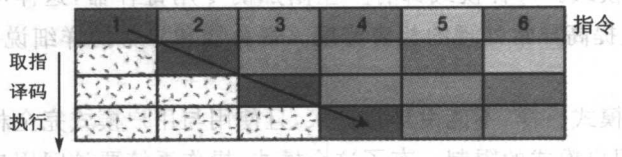

图1.3 ARM7三级流水线

使指令流可以更好地通过流水线。由于流水线是CPU的一部分,它对程序员是透明的。但必须记住:PC指向的是正在被执行指令的前8个字节位置,当计算与PC有关的地址时,这一情况需要注意。例如指令:

```
0x4000  LDR PC,[PC,#4]
```

该指令会把地址PC+4的内容载入PC,但由于PC指向的是正在执行指令的前8个字节,所以地址0x400C中的内容会被载入PC,而不是像用户可能以为的那样载入地址0x4004中的内容。

1.1.4 处理器模式

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。ARM体系结构共支持7种处理器模式:用户模式(user)、快中断模式(fiq)、中断模式(irq)、管理模式(svc)、中止模式(abt)、未定义模式(und)和系统模式(sys)。ARM7TDMI完全支持这7种模式,如

表 1.3 所列。

表 1.3 七种处理器模式

处理器模式	说 明	备 注
用户模式(user)	正常程序工作模式	运行应用程序,不能直接切换到其他模式
中断模式(irq)	用于通用中断处理	IRQ 异常响应时,进入此模式
快中断模式(fiq)	支持高速数据传输及通道处理	FIR 异常响应时,进入此模式
管理模式(svc)	操作系统保护代码	处理器复位后进入管理模式;操作系统内核也处于管理模式
中止模式(abt)	用于支持虚拟内存和存储器保护	当处理器访问存储器失败时进入数据访问中止模式
未定义模式(und)	用于支持硬件协处理器的软件仿真	当处理器遇到没有定义的指令或者处理器不支持该种指令的时候进入未定义模式
系统模式(sys)	用于支持操作系统的特权任务等	特殊的用户模式,允许对 CPSR 的完全读写访问

7 种处理器模式中,除用户模式以外,其他均为特权模式。ARM 内部寄存器(如 CPSR)和一些片内外设在硬件设计上只允许(或者可设置为只允许)在特权模式下访问。另外,在特权模式下,可以任意地进行处理器模式的切换,而用户模式由于不能修改 CPSR 寄存器的值,所以不能直接切换到其他的处理器模式。

快中断模式、中断模式、管理模式、中止模式、未定义模式 5 种特权模式又被称为异常模式。异常模式除了可以通过程序切换进入以外,也可以由特定的异常进入。当异常发生时,处理器进入相应的异常模式。每种模式都有一些附加的专用寄存器,这样可以避免异常退出时程序状态不可靠,并且提高异常处理的相应速度。寄存器和异常的详细说明见 1.1.6 和 1.1.7 小节。

系统模式与用户模式一样,不能由异常进入,且使用与用户模式完全相同的寄存器。然而它是特权模式,不受用户模式的限制。有了这个模式,操作系统要访问用户模式的寄存器就比较方便。同时,操作系统的一些特权任务可以采用这个模式,以访问一些受控的资源,而不必担心出现异常时处理器状态的变化。

1.1.5 处理器工作状态

ARM 处理器有 3 种工作状态:ARM 状态、Thumb 状态和 Jazelle 状态。3 种状态分别对应三种指令集:32 位的 ARM 指令集、16 位的 Thumb 指令集和 8 位的 Jazelle 指令集。只有进入特定的状态,相应的指令集才有效。ARM、Thumb 和 Jazelle 指令不能混合使用,除非通过特定的指令(如 BX)进行状态之间的切换。

CPSR 的 J(Jazelle)和 T(Thumb)位反映了程序的状态:当 T 位置位时,处理器处于 Thumb 状态;当 J 位置位时,处理器处于 Jazelle 状态;当 T 和 J 位都为 0 时,处理器处于 ARM 状态。

ARM7TDMI(-S)处理器内核使用 ARMv4T 结构实现,该结构只支持 ARM 和 Thumb 指令集。

有关 ARM 和 Thumb 指令集特征的详细描述如表 1.4 所列。

表 1.4　ARM 和 Thumb 指令集特征

指令集	ARM	Thumb
指令长度	32 位	16 位
内核指令	58 条	30 条
条件执行	大多数指令	只有分支指令
数据处理指令	访问桶形移位寄存器和 ALU	独立的桶形移位寄存器和 ALU 指令
程序状态寄存器	特权模式下可读/写	不能访问
寄存器使用	15 个通用寄存器＋pc	8 个通用寄存器＋7 个高寄存器＋pc

例 1.1 为使用 BX 指令在 ARM 和 Thumb 状态之间进行切换的程序范例。

[例 1.1]　处理器工作状态切换

```
;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
```

备注　1. 处理器工作状态的切换并不影响处理器模式和寄存器内容。

2. 由于 Thumb 状态下不能访问 CPSR，故所有的异常处理都在 ARM 状态中进行。如果在 Thumb 状态中发生了异常，那么处理器会切换到 ARM 状态，异常处理完毕后再切换回 Thumb 状态。

1.1.6　寄存器组织

ARM7 是 Load-Store(装载-存储)结构，如图 1.4 所示。所以为了执行任何数据处理指令，首先要把数据从存储器中装载到寄存器中，在数据处理指令执行结束后，如果需要，数据再被保存到存储器中。

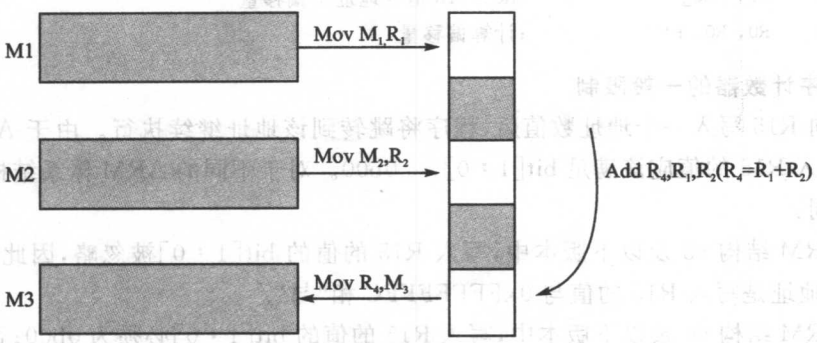

图 1.4　Load-Store(装载-存储)结构

1.1.6.1 通用寄存器

ARM 的中央寄存器集是 16 个用户寄存器 R0～R15。这些寄存器均是 32 位宽，R0～R12 没有其他特殊功能。寄存器 R13～R15 在内核中有特殊功能。R13 被用作栈指针(Stack Pointer，SP)。R14 为链接寄存器(Link Register，LR)，当调用一个函数时返回地址被自动保存到链接寄存器，并立即在函数返回时有效。这使得快速进入和返回“叶”函数(不调用其他函数的函数)成为可能。如果函数是分支的一部分(即该函数将调用另一个函数)，链接寄存器必须入栈(R13)。R15 为程序计数器(Program Counter，PC)。有趣的是，许多指令可以在 R13～R15 中执行，就好像它们是标准的用户寄存器。寄存器组结构如图 1.5 所示。

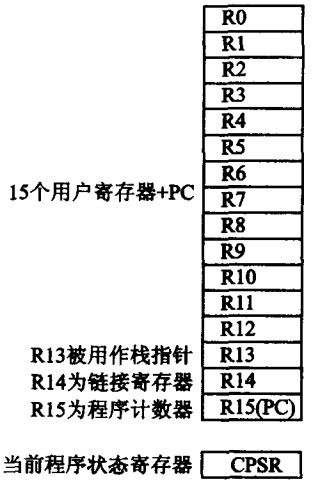

图 1.5 寄存器组

PC 虽然可以作为通用寄存器使用，但是有些指令在使用 PC 时有一些特殊限制。当违反了这些限制时，该指令的执行结果将是不可预测的。

1. 读程序计数器的一般限制

由于 ARM7TDMI 采用了 3 级流水线机制，当正确读取了 PC 的值时，该值为当前的指令地址加 8 个字节。也就是说，对于 ARM 指令集，PC 指向当前指令的下两条指令的地址。由于 ARM 指令是字对齐的，所以 PC 值的第 1 位和第 0 位总为 0。

当使用 STR 或者 STM 指令保存 R15 时，出现了例外。这些指令可以将 PC 保存为当前指令地址加 8 字节，也有可能保存为当前指令地址 12 字节。偏移量是 8 还是 12 因 ARM 处理器芯片的不同而不同，但对于某个具体的芯片它是个常量。所以在实际应用中，应尽量避免使用 STR 或者 STM 指令来读取 R15 的值，或者使用合适的指令序列来确定当前芯片的偏移量。例 1.2 中的代码可以用来确定 STR 指令读取 PC 时的偏移量。

[例 1.2] 获取芯片采用的地址偏移量

```
SUB    R1, PC, #4          ;R1 = 下面 STR 指令的地址
STR    PC, [R0]            ;保存 STR 指令地址 + 偏移量
LDR    R0, [R0]            ;R0 = STR 指令地址 + 偏移量
SUB    R0, R0, R1          ;计算偏移量
```

2. 写程序计数器的一般限制

当成功向 R15 写入一个地址数值后，程序将跳转到该地址继续执行。由于 ARM 指令是字对齐的，写入 R15 的值应该满足 bit[1：0] = 0b00。对于不同的 ARM 体系结构版本，具体要求有所不同：

- 在 ARM 结构 v3 及以下版本中，写入 R15 的值的 bit[1：0]被忽略，因此指令的实际目标地址是写入 R15 的值与 0xFFFFFFFC 相“与”。
- 在 ARM 结构 v4 及以下版本中，写入 R15 的值的 bit[1：0]必须为 0b00；否则，结果将不可预测。

1.1.6.2 当前程序状态寄存器

除了 16 个用户寄存器外还有一个 32 位宽的寄存器——当前程序状态寄存器(CPSR)，如

此为试读，需要完整PDF请访问：www.ertongbook.com