从 ARM 体系看嵌入式处理器的发展

From ARM Architecture to See the Developments of the Embedded Processor

清华大学 何荣森 何希顺 张跃 (北京 100084)

摘 要：文章介绍了 ARM 体系的发展历史，它的指令集特点，程序模型和利用 ARM 体系处理器的软件开发和硬件调试过程。同时从 ARM 体系，我们也可以看到 RISC 在嵌入式处理器领域的优势所在，以及它们将来必然在 SOC(系统芯片)中获得广泛应用。

关键词：ARM，ARM 体系，ARM 指令集，SOC，IP 核

1 ARM 体系的发展历史

CISC 体系由于指令集庞大，指令长度不固定，指令执行周期有长有短，使指令译码和流水线的实现在硬件上非常复杂，给芯片的设计开发和成本的降低带来了极大困难。

针对这些明显的弱点，美国加州大学伯克利分校的 Patterson 教授领导的研究生团队设计和实现了“伯克利 RISC I”处理器，他们在此基础之上又发展了后来 SUN 公司的 SPARC 系列 RISC 处理器，并使得采用该处理器的 SUN 工作站名振一时。与此同时，斯坦福大学也在 RISC 研究领域取得了重大进展，开发并产业化了 MIPS 系列 RISC 处理器（后来被 SGI 公司收购，并广泛用于 SGI 的图形工作站）。被这两种 RISC 处理器所取得的巨大成功所鼓舞，英国的 Acorn 计算机公司在 1983 到 1985 年之间也开发出了第一代 ARM RISC 处理器，当时 ARM 还只是 Acorn RISC Machine 的缩写。之后于 1990 年，公司又把名字改为简单的 ARM limited，并且是 Advanced RISC Machine(先进 RISC 机器)的缩写。经过这么多年的发展，ARM 已经形成了如下系列(表 1)的大家族。

表 1

Core	Architecture
ARM1	v1
ARM2	v2
ARM2aS, ARM3	v2a
ARM6, ARM600, ARM610	V3
ARM7, ARM700, ARM710	V3
ARM7TDMI, ARM710T, ARM720T, ARM740T	V4T
StrongARM, ARM8, ARM810	V4
ARM9TDMI, ARM920T, ARM940T	V4T
ARM9E－S	V5TE
ARM10TDMI, ARM1020E	V5TE

表 1 中的核心 Core 包括处理器核 Processor Core(ARM7TDMI, ARM8, StrongARM, ARM9TDMI & ARM10TDMI)和 CPU Core(其它)。CPU Core 实际上是在各种相应的处理器核的基础之上，集成和优化了 Cache 和 MMU 内存管理单元（Memory management unit）后形成的，它能以单独的 CPU 芯片提供给用户更高的性能。而各种处理器核（Processor Core）则可用软 IP 模块，或者硬 IP 模块的方式嵌入到各种用户不同的应用之中，形成相应的 SOC（System on a Chip）系统芯片，这不仅节省了功耗和成本，还最大限度的节省了用户的开发时间。另外，特别值得一提的是 StrongARM，它原来是 DEC 公司和 ARM 公司合作开发的。1998 年，Intel 收购了 DEC 的半导体部门。此后，Intel 又开发和生产出 SA110，SA1100，SA1110 等一系列 StrongARM 高性能嵌入式处理器。特别是 SA1110，本身就是一个非常典型的 SOC 芯片，已经把液晶控制器、外设接口（USB、IrDA、UART、PCMCIA）、音频编解码器 Codec 等和 CPU 集成到了同一个芯片内，可以很方便地嵌入于各种掌上设备。

收稿日期：2001－06－08

2 ARM 体系的指令集

一个 CPU 的指令集是硬件和软件之间的一个重要的分水岭，根据分层的思想，指令集向上要有力地支持编译器，向下要方便硬件的设计实现。ARM 是典型的 RISC 体系，根据 RISC 的设计思想，其指令集的设计应该尽可能地简单，和 CISC 体系相比，它可以通过一系列简单的指令来实现复杂指令的功能。

(1) ARM 的指令集包括 6 种典型的指令：

① 分支指令：如 B、BL 等。

② 数据处理指令：如 ADD、SUB、AND 等。

③ 状态寄存器转移指令 如 MRS、MSR 等。

④ LOAD－STORE 数据移动指令 如 LDR 等。

⑤ 协处理器指令 如 LDC、STC 等。

⑥ 异常处理指令 如 SWI 等。

(2) ARM 指令集是一个非常优秀的指令集，它有以下特点：

① 所有 ARM 指令都是 32 位定长，在内存中以 4 字节边界保存（地址最后两位为 0），这样方便译码电路和流水线的实现。当然 ARM 内核一般也支持另外一种 16 位的指令集 Thumb（感兴趣的读者可以参考 ARM 手册，本文不作详述），Thumb 指令集可以看作是 ARM 指令集的一种压缩形式，它在处理器中仍然要扩展为标准的 32 位 ARM 指令来运行。用户采用 16 位 Thumb 指令集最大的好处就是可以获得更高的代码密度和降低功耗，当然它是被 ARM 的开发工具所完全支持的。

② LOAD—STORE 架构。由于 ARM 指令集属于 RISC 体系，RISC 体系的特征就是：一般指令只能把内部寄存器和立即数作为操作数，只有 LOAD—STORE 类型的数据移动指令才可以访问内存，在内存和寄存器之间转移数据。

③ 由于硬件上有 barrel（桶型）移位器，所以 ARM 可以在一个指令中用一个指令周期完成一个移位操作和一个 ALU（算术逻辑）操作。

④ 所有指令都可以条件执行，这是由其指令格式决定的，如表 2 所示。

表 2

31　cond　28	27　0

任何指令的头 4 位都是条件指示位，根据 CPSR（当前程序状态寄存器，见后面表 4）中的 N（负指示），Z（零指示），C（进位指示），V（溢出指示）决定该指令是否执行。这样可以方便高级语言的编译器设计，很容易实现分支和循环。

⑤ 有功能很强的一次加载和存储（LOAD－SRORE）多个寄存器的指令：LDM 和 STM。这样，当发生过程调用或中断处理时，可以只用一条指令就能把当前多个寄存器的内容保护到内存堆栈中。

从以上特点可以看出，ARM 指令集可以很方便地实现汇编语言设计，同时也强有力支持了高级语言（如 C 语言）编译器的开发。

3　ARM 体系的程序模型

在 ARM 的指令集之上，如果应用简单，用户可以只需要编一些单个的控制程序就能满足要求，这就类似单片机的开发。但是如果应用很复杂，就需要加载嵌入式操作系统（如 WinCE，嵌入式 Linux），然后在嵌入式操作系统之上开发用户的应用程序，通过嵌入式操作系统来完成用户进程的调度和对低层硬件的控制。同时，嵌入式应用的环境往往要优先保证实时性。因此，ARM 体系对操作系统和实时性的保证都提供了强有力的支持，这从 ARM 的程序模型可以看出。

（1）处理器状态

ARM 体系支持 7 种处理器模式（Processor mode）如下所示：

① 用户模式（User: usr）：CPSR 中的 M[4: 0] = 0b10000（见后面表 4），这是用户程序运行的正常模式，在这种模式下，程序不能访问一些受保护的资源，以利于操作系统控制系统资源的使用。其它的 6 种模式则称为特权模式（Privileged mode）。

② 系统模式（System: sys）：CPSR 中的 M[4: 0] = 0b11111，这种模式只有 ARMv4 及其之上的版本才能支持，主要用于对操作系统的支持和对系统资源的管理。

③ 快速中断模式（FIQ: fiq）：CPSR 中的 M[4: 0] = 0b10001，快速响应用户中断，支持高速数据传输，以满足少数需要极高实时性的请求。

④ 中断模式（IRQ: irq）：CPSR 中的 M[4: 0] = 0b10010，用于一般的中断处理。可以对应一般的多个中断源。

⑤ 监督模式（Supervisor: svc）：CPSR 中的 M[4: 0] = 0b10011，用于操作系统的保护模式。

⑥ 中止模式（Abort: abt）：CPSR 中的 M[4: 0] = 0b10111，用于对虚拟内存的实施和保护。

⑦ 未定义模式（Undefined: und）：CPSR 中的 M[4: 0] = 0b11011，支持用软件仿真硬件的协处理器。

第③～⑦的五种模式则称为异常模式，这些模式的进入需要特定的异常发生。

（2）寄存器分配

对于 RISC 体系，由于采用了精简指令集和控制单元的硬连线设计，所以芯片上可以有更多的面积分配给寄存器，以加快程序运行。针对以上 7 种处理器模式，ARM 对其内部的寄存器进行了划分，如表 3 所示。

ARM 的寄存器都是 32 位的，从表 3 可以看出系统模式和用户模式使用的寄存器组是相同的，而且 R0～R14 在任何模式下都是可见的，但是其它 5 种异常方式则各添加了一些寄存器（用下标标明），以当异常发生时保护异常发生前用户的状态。

这里需要对当前和过去程序状态寄存器（CPSR 和 SPSR）作一些说明，其格式如表 4 所示。

其中各位如下：

N：结果为负指示。

表 3

	特权模式(Privileged mode)					
		异常模式(Exception mode)				
User	System	Supervisor	Abort	Undefined	IRQ	FIQ
R0 ~ R7	R0 ~ R7	R0 ~ R7	R0 ~ R7	R0 ~ R7	R0 ~ R7	R0 ~ R7
R8 ~ R12	R8 ~ R12	R8 ~ R12	R8 ~ R12	R8 ~ R12	R8 ~ R12	R8_fip ~ R12_fip
R13	R13	R13_svc	R13_abt	R13_und	R13_irq	R13_fiq
R14	R14	R14_svc	R14_abt	R14_und	R14_irq	R14_fiq
PC	PC	PC	PC	PC	PC	PC
CPSR	CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
		SPSR_svc	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq

Z：结果为 0 指示。

C：进位位指示。

V：溢出指示。

I：当设置为 1 时，禁止 IRQ 中断。

F：当设置为 1 时，禁止 FIQ 中断。

T：当设置为 1 时，表示执行 ARM 指令 ;设置为 0 时，表示执行 Thumb 指令。

M[4: 0]：当前处理器运行模式指示，参见前面的 7 种模式。

表 4

31	30	29	28	26 ~ 8	7	6	5	4 ~ 0
N	Z	C	V	无用	I	F	T	M4 ~ M0

(3) 异常处理

当在程序的运行之中，发生的外部（如中断请求）或内部（如未定义指令）事件需要处理器立即进行处理，就产生了一个异常。ARM 支持 7 种异常如表 5 所示。

表 5

异常方式	处理器模式	向量地址
重启(Reset)	SVC	0x00000000
未定义指令 (Undefined instruction)	UND	0x00000004
软件中断(Software interrupt)	SVC	0x00000008
预取指令错误(Prefetch abort)	Abort	0x0000000C
取数据错误(Data abort)	Abort	0x00000010
正常中断(IRQ)	IRQ	0x00000018
快速中断(FIQ)	FIQ	0x0000001C

当发生异常时，首先要保存当前程序的返回地址和 CPSR，再执行其它相应措施，然后进入到相应的异常向量地址，一般来说在异常向量地址是一个跳转指令，使程序进入相应的异常处理过程。其处理如下：

```
R14_ <exception_mode>  = return link
SPSR_ <exception_mode>  = CPSR
CPSR[4: 0] = exception mode number
CPSR[5] = 0        /* 执行 ARM 指令 */
If  <exception_mode>  = = Reset or FIQ then
CPSR[5] = 0CPSR[5] = 0 /* 屏蔽 IRQ 和 FIQ */
PC = 异常向量地址
```

当异常的处理过程返回时，只需把 SPSR 和 R14 的内容分别置回 CPSR 和 PC 了。

(4) 输入输出系统(I/O system)

ARM 对输入输出的外部设备是作为内存映象形式处理的，它们的寄存器地址和内存一样是统一编址的，访问这些外部设备可以采用读取和设置这些地址，就和访问内存一样。唯一的区别就是 I/O 寄存器是“读敏感”的，每次读的值都可能不一样。因此，它们不能被缓冲和缓存。

4 ARM 体系的软件开发工具和调试步骤

作为主要面向嵌入式应用的 ARM 体系，没有良好的软件开发支持是不可想象的，ARM 公司和许多第三方都为此提供了性能优越的软件开发工具。下面先就整个软件的开发和硬件调试流程作一个简单介绍，如图 1 所示。

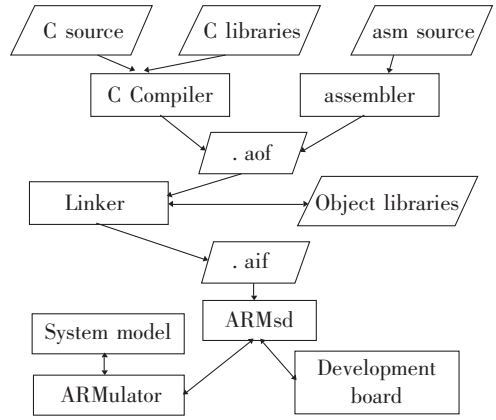

图 1

首先，由于 ARM 广泛用于嵌入式系统之中，所以开发者一般都采用基于 Windows PC 或者 UNIX 工作站的交叉开发系统（产生另外一种不同架构的代码）。首先，C 或者汇编源程序被编译为 ARM 目标

代码形式文件(. aof : ARM object format),然后经过“连接”就可以建立 ARM 映像形式文件(. aif : ARM image format)。通过 ARM 符号调试器 (ARMsd: ARM symbolic debugger) 就可以把 ARM 映像形式文件通过 RS232 串口或者 JTAG 接口下载到用户的目标开发板上运行。当然,开发者也可以先在自己的主机上用 ARMulator(ARM emulator) 先仿真运行,看程序运行是否正确。

下面就以 ARM 公司给 Windows 台式机提供的开发环境为例来稍作说明,该软件全名“ARM 软件开发工具包”(ARM Software Development Toolkit)。它主要包括两个部分:第一是 ARM Project Manager 项目管理器,用于 C 和汇编源程序的管理和编译。第二是 ARM Debugger for windows 调试器用于仿真和在线调试功能,相当于 ARMsd。

5　嵌入式处理器的发展前景

从 ARM 体系可看到当前嵌入式处理器的发展趋势,同时也可作为我国集成电路发展的参考。

(1) 首先,嵌入式处理器领域普遍要求设计周期短,低功耗,性能优越。CISC 体系很难满足这些要求,如最新的英特尔 P4 处理器,内部就有 4200 万个晶体管,使得设计难度和测试都很难把握 (在 Intel 的历史上,曾因质量问题,数次回收芯片),而且功耗高达 55 瓦,根本不可能用于嵌入式领域。而 CISC 的缺点正好是 RISC 的优点(这也难怪 ARM 最初选准了 RISC 作为突破口),相对而言,RISC 的设计周期和设计难度要小一个数量级,而且所用的管子数和芯片功耗也要少很多。所以在嵌入式处理器领域,RISC 体系占了压倒性优势也就不足为奇。

(2) 当前,以软硬件协同设计、具有知识产权的内核 (IP 核) 复用和超深亚微米技术为支撑的 SOC (System On Chip 系统芯片) 已成为集成电路发展的一大趋势。由于信息处理的复杂度,使得 SOC 芯片必须具备强大的数据处理能力,嵌入式 CPU 或 DSP 的使用将成为 SOC 的一个重要标志。事实上,一个 SOC 芯片上集成有一个或多个的微处理器,在今天的集成电路设计中已不少见。但是,这也给 SOC 的设计带来了极大难度。因此,采用第三方的 IP 核成了 SOC 设计的必然,这不仅可以减小自己的开发风险,更重要的是缩短了 SOC 芯片的上市周期,以应对激烈的市场竞争。ARM 体系也正是基于这一点,为用户提供了各种成熟的通用 RISC 处理器核,把它们作为 IP 核 (硬 IP 核或者软 IP 核) 嵌入到用户的各种 SOC 应用之中,如 GSM 移动通信处理芯片、蓝牙基带处理芯片等,从而使自己在嵌入式处理器领域占有重要的一席之地。

(3) 当前,我国的集成电路工业正处于一个大发展的时期,北京和上海都计划和正在建立几十条集成电路生产线。然而,对处理器来说,在台式机领域,我们想和 Intel 和 AMD 抗衡还不可能;只有在嵌入式领域,由于其应用的广泛性和特殊性,还可以大有一番作为。众所周知,集成电路设计,制造和封装是相对独立的。ARM 就是一家典型的集成电路设计公司。对集成电路设计业来说,它没有厂房和设备折旧的问题,其最大的成本就是设计人员的人力成本。因此,就发展我国的集成电路产业来说,光有生产线是不行的,还必须成立一批高水平的集成电路设计公司。而且,针对嵌入式领域的广大市场,我们的设计公司是否可以尝试充分利用我国的人才优势,开发出一些具有自主知识产权的 RISC 处理器 IP 核,并力争形成有一定市场规模的应用,这样我们国家才能在嵌入式处理器领域有一番作为。

参考文献

[1] David A Patterson, John L Hennessy. Computer Organization and Design: the hardware/software interface. San Francisco, Morgan Kaufmann, 1998.

[2] David Seal. ARM Architecture Reference Manual. Pearson Education limited, 2001.

[3] Steve furber. ARM System – On – Chip architecture(second edition). Pearson Education limited, 2001.

HE Rong-sen, HE Xi-shun, ZHANG Yue
(TsingHua university, Beijing 100084)

Abstract: In this paper, the history of the ARM Architecture and its characteristics are introduced in details, such as the instruction set, program model, the process of software development and hardware debugging. From the ARM Architecture, we could see that RISC has so many merits in embedded systems. And in the new trend, embedded processors will be widely used in SOC (System On Chip).

Key words: ARM, ARM architecture, ARM instruction set, SOC, IP Core

何荣森　1976 年生,硕士。研究方向为蓝牙通信技术、各种嵌入式系统的设计和开发等。
何希顺　1965 年生,博士。主要从事数字信号处理、掌上电脑以及嵌入式系统的研究和开发。