μClinux

源代码中 Make 文件完全解析

——基于ARM开发平台

田军营　韩建海　马志荣　编著

附光盘

机械工业出版社
CHINA MACHINE PRESS

μClinux
源代码中 Make 文件完全解析
——基于 ARM 开发平台

田军营　韩建海　马志荣　编著

机 械 工 业 出 版 社

在 Linux 或 μClinux 源代码中，Make 文件是一种重要的文件，它担当着编译生成系统目标代码的重任。本书立足于 μClinux 源代码，通过分析其中的 Make 文件，以及与这些 Make 文件密切相关的部分源代码，给出一种在源代码水平上理解系统目标代码生成过程和生成原理的方法。这不仅是一本关于 Make 文件编写、学习、组织、应用的书籍，也是一本基于 ARM 技术进行 μClinux 嵌入式系统开发的工程技术人员不可缺少的参考书，同时也是高等院校中有志于学习和应用 Linux 或 μClinux 师生们的良好教材。

图书在版编目（CIP）数据

μClinux 源代码中 Make 文件完全解析——基于 ARM 开发平台／田军营等编著. —北京：机械工业出版社，2005.6
ISBN 7-111-16528-4

Ⅰ. 基... Ⅱ. 田... Ⅲ. Linux 操作系统—机器代码程序—程序分析
Ⅳ. TP316.89

中国版本图书馆 CIP 数据核字（2005）第 044681 号

机械工业出版社（北京市百万庄大街 22 号　邮政编码 100037）
策　　划：胡毓坚
责任编辑：李利健
责任印制：洪汉军

北京原创阳光印业有限公司·新华书店北京发行所发行

2005 年 7 月第 1 版·第 1 次印刷
787mm×1092mm 1/16·24.75 印张·615 千字
0 001—5000 册
定价：42.00 元（含 1CD）

前　言

随着科技日新月异的发展，信息家电、手持设备、无线设备等个性化设备层出不穷，相应的硬件和软件也得到迅速发展，嵌入式系统变得越来越重要。

ARM 嵌入式处理器是一种高性能、低功耗的 RISC 芯片。该处理器具有 RISC 体系的一般特点，又具有自身的许多优点，使得它在无线设备、蓝牙技术、互联网、消费电子、汽车、海量存储设备、成像设备（打印机及相机）、安全产品等领域获得了广泛的应用。目前，基于 ARM 技术的处理器已经占据了 32 位 RISC 芯片 75% 以上的市场份额。可以说，ARM 技术几乎无处不在。ARM 技术具有很高的性能和功效，因而容易被厂商接受。同时，随着合作伙伴的增多，可获得更多的第三方工具、制造和软件支持，使整个系统成本降低，产品进入市场的时间加快，从而具有更大竞争力。

Linux 是一种可以运行于个人电脑上的、类 Unix 的操作系统。它于 1991 年产生，具有稳定、高效、易定制、易裁减、硬件支持广泛等特点，同时又是一个免费的开放源代码的操作系统。后 PC 时代的计算设备——嵌入式计算设备的发展，使得一些著名的公司选中了 Linux 操作系统作为开发嵌入式产品的工具。Linux 在嵌入式领域异军突起，成为目前最流行的嵌入式操作系统。

μClinux 是最流行的嵌入式 Linux 系统之一。在 μClinux 单词中，u 表示 Micro——小的意思，C 表示 Control——控制的意思，μClinux 就是 Micro-Control-Linux，字面上的理解就是“针对微控制领域而设计的 Linux 系统”。μClinux 包含丰富的功能：文件系统、各种驱动程序、通信模块、TCP/IP、PPP、HTTP、Web 服务器代码等。在 Internet 上流传的 μClinux 已经被移植到当前几乎所有的硬件平台上，功能与 PC 机上运行的 Linux 不相上下，其代码也十分复杂。由于 μClinux 的代码经过世界范围内优化，性能稳定、可靠、高效，模块代码均可以从 Internet 上获得并能进行移植，所以，μClinux 作为一体化平台，已成为越来越多嵌入式系统的开发利器，如该平台可广泛用于掌上电脑、GPS 接收机、POS 收银机、测试仪等领域。

Make 是一个程序系统，它通过描述程序中各个文件之间相互关系及其更新命令的 Make 文件，来自动决定该程序中哪些文件需要重新编译，并重新编译它们。Make 不仅可以处理 C 语言程序，而且还可以处理那些能用 Shell 命令运行其编译器的各种语言程序。Make 在实际应用中不仅仅限于程序，它更适用于那些由于一些文件变化导致另外一些文件必须更新的任务。

对于 μClinux 这样的操作系统，Make 文件是系统编译、生成的基础。分析系统源代码文件中所包含的 Make 文件，对理解系统的结构、代码组织、原理及工作都有重要的意义。在 μClinux 源代码中，Make 文件是非常多的，不同的 Make 文件与不同的系统部分相关。从整体上看，这些 Make 文件分为核心 Make 文件（或主 Make 文件）、子目录 Make 文件、规则 Make 文件（Rules. make）、核心配置文件四类。从布局上看，它们分别对应于 μClinux 系统的各个方面，如对应于不同的硬件体系，特别是不同的 CPU，又如对应于操作系统的某些具体模块的实现，像进程管理、文件管理、进程通信等部分。

为了便于理解和掌握 μClinux 源代码中的 Make 文件，根据 Make 文件在源代码中的作用

和地位，可将它们分为以下五部分：

- 核心 Make 文件(或主 Make 文件)；
- arch/ * /Makefile：结构 Make 文件；
- 子目录 Make 文件：大约有 300 个；
- Rules.make：所有子目录 Make 文件所遵循的公共规则；
- .config：核心配置文件。

它们的关系大致为：核心 Make 文件可以读从核心配置过程中得到的核心配置文件.config；根据核心配置文件，核心 Make 文件负责建立两个主要结果：vmlinux(常驻核心影像)和模块(任何模式文件)；这些结果通过递归进入核心源代码子目录树，并运行相关的子目录 Make 文件来完成；被访问的子目录 Make 文件同样依赖核心配置文件；结构 Make 文件在核心 Make 文件递归进入时为核心 Make 文件提供基于特殊结构的结构信息(如基于 CPU、硬件板卡等方面的硬件资源信息)；子目录 Make 文件在核心 Make 文件递归进入时为核心 Make 文件提供基于相关子目录内容的信息。所以，在 μClinux 源代码体系中，几乎每一个子目录均包含一个从核心 Make 文件传递下来的由执行指令组成的 Make 文件，这些 Make 文件就是子目录 Make 文件。在这个意义上，结构 Make 文件也是一种子目录 Make 文件。不过，它是一种具有特殊含义的子目录 Make 文件，在 μClinux 源代码体系中具有特殊地位和特殊作用，因此将它单独列出，单独分类，并称为结构 Make 文件。子目录 Make 文件使用核心配置文件信息以构造各种文件列表，而后执行包含公共规则的 Rules.make。Rules.make 定义了所有子目录 Make 文件都赖以遵守的公共规则。它通过一组变量确定一个公共内部界面，并在此基础上定义一组公共规则。

对掌握和研究 μClinux 源代码的用户进行考查，不难发现，使用 Make 文件的用户有四类：

- 建立核心的用户。这些用户输入诸如“make menuconfig”或“make bzImage”命令，通常不读或不编辑任何 Make 文件(或任何其他源代码文件)。
- 建立设备驱动程序、文件系统、网络协议的一般高级开发用户。这些用户需要掌握子目录 Make 文件，并利用它们开发子系统。为了有效地进行开发，他们需要较为全面地了解核心 Make 文件和 Rules.make 公共内部界面的相关细节。
- 建立诸如 spac 或 ia64 结构的结构高级开发用户。这些用户需要掌握结构 Make 文件，并利用它们开发结构子系统。为了有效地进行开发，他们需要较为全面地了解核心 Make 文件和 Rules.make 公共内部界面的相关细节。
- 建立核心系统本身的核心开发用户。这些用户需要掌握所有的 Make 文件内容。

因此，本书的内容对建立核心的用户、一般高级开发用户、结构高级开发用户、核心开发用户都有意义，同时，对刚开始学习用 GNU Make 编写实际应用 Make 文件的人员，和刚开始学习研究 μClinux 核心源代码中 Make 文件的人员也有不同程度的帮助，并为他们提供了一个可以参照的实际平台。

鉴于英文中无“μ”字母，源代码中“μClinux”的书写一般用“uClinux”代替。为使正文与代码的叙述一致，书中凡使用“μClinux”的地方均使用了“uClinux”。

本书的主要内容包括：第 1 章阐述嵌入式系统、ARM 技术、uClinux 的概念和特点；第 2 章首先通过一个 Make 文件的具体例子，说明 Make 文件的功能、建立方法和建立特点，而后给出编制 Make 文件的相关知识；第 3 章从 Linux 体系中的 Make 文件入手，阐明了这些 Make 文件

的相互关系以及这些关系的实现;第 4 章在阐明基于 ARM 的 uClinux 源代码中 Make 文件之间的相互关系的基础上,对这些 Make 文件进行了分类和说明;第 5～8 章分别给出对单个 Make 文件的功能分析和代码说明;第 9 章以系统配置为中心,分析了完成系统配置的源代码以及它们与相关的 Make 文件的关系,说明了实现系统配置的技巧;第 10 章给出了 uClinux 在自动建立系统源代码之间依赖上的代码分析。源代码系统之间的依赖是 uClinux 的 Make 文件系统功能实现的基础,它的自动实现对系统建立有着极其重要的意义。由于任何系统源代码在阅读时都非常难懂,因此,第 11 章给出了第 5～10 章中部分源代码的执行情景分析,这种分析对读者理解相关源代码的功能十分重要,并对读者阅读前面各章节的内容提供有益的帮助。

根据上述内容,阅读本书的读者可以按下列原则进行阅读:

- 对嵌入式系统、ARM 技术、uClinux 以及 Make 文件有较深入了解的读者,可以略去第 1 章和第 2 章的内容不读。
- 对 Make 文件只有一般了解的读者,建议在仔细读完第 2 章内容后,再进行本书后续内容的阅读。
- 阅读本书第 4～10 章的读者,最好根据本书提供的光盘信息进行相关练习,并作出适当的观察。无此条件的读者,可结合第 11 章的相关内容进行阅读。方法是先仔细阅读第 4～10 章的相关内容,有一定基础后,再对第 11 章的有关内容进行阅读,以期产生良好的阅读效果。
- 有一定嵌入式系统开发经验或正在从事相关开发的人员,阅读本书更为有利。
- 本书的内容可作为 Linux 源代码中相关部分的重要参考。对学习 Linux 也有意义。

本书附带的光盘提供以下内容:

① uClinux 源代码。

② uClibc 源代码。

③ arm-elf-gcc、arm-elf-ld、arm-elf-objdump 等 ARM 工具源代码和程序。

④ arm-elf-gcc、arm-elf-ld、arm-elf-objdump 等 ARM 工具说明书。

⑤ make 编译 uClinux 源代码后得到的中间结果或中间文件。

⑥ 部分练习用的实验材料。

由于编者水平有限,书中出现的错误和不妥之处,敬请读者批评、指正。

编者

目　录

第 1 章　基于 ARM 的 uClinux 简介

ARM 是一款基于 RISC 体系的处理器，也是一款开发嵌入式系统的主流芯片。ARM 芯片和嵌入式 Linux 在工业上的广泛应用，使得它们的结合更具生命力。本章从嵌入式系统出发，进一步阐明基于 ARM 的 uClinux 状况。

1.1　嵌入式系统

后 PC 时代有三大技术发明：嵌入式系统、Linux、ARM 技术。现在，它们已成功地走在一起，共同组织成由嵌入式 Linux 搭建的基于 ARM 的嵌入式系统。

1. 嵌入式系统的概念

嵌入式系统是什么呢？嵌入式系统一般指非 PC 系统，包括硬件和软件两部分。硬件部分包括处理器、存储器、外设器件、I/O 端口、图形控制器等。软件部分包括操作系统和应用程序。操作系统控制着应用程序编程和系统与硬件之间的交互；应用程序控制着系统的运作和行为。

嵌入式系统的核心是嵌入式处理器。嵌入式处理器一般具备四个特点：①对实时多任务有较强的支持能力。即，该处理器能完成多任务且有较短的中断响应时间，使内部的代码和实时内核的执行时间减少到最低限度。②具有功能较强的存储区保护功能。由于嵌入式系统软件结构的模块化，为避免软件模块之间出现错误的交叉和有利于软件诊断，需要设计强大的存储区保护功能。③可扩展的处理器结构。这种嵌入式处理器能满足应用的最高性能，并在最短的时间内开发出来。④嵌入式处理器功耗很低。如靠电池供电的嵌入式系统，功耗只有 mW 级，甚至 μW 级。

在当前数字信息技术和网络技术高速发展的后 PC 时代，嵌入式系统已经广泛地渗透到科学研究、工程设计、军事技术、各类产业和商业文化艺术以及人们的日常生活等各个方面。随着国内外各种嵌入式产品的进一步开发和推广，嵌入式技术和人们的生活越来越紧密地结合在一起。

2. 嵌入式系统的特点

嵌入式系统的运行环境和应用场合决定了嵌入式系统具有区别于其他系统的特点：

嵌入式系统具有嵌入式处理器。嵌入式处理器可以分为三类：嵌入式微处理器、嵌入式微控制器、嵌入式 DSP。嵌入式微处理器就是和通用计算机的微处理器对应的 CPU。在应用中，一般是将微处理器装配在专门设计的电路板上，在母板上只保留与嵌入式相关的功能，这样可以满足嵌入式系统体积小和功耗低的要求。目前的嵌入式微处理器主要包括：PowerPC、Motorola 68000、ARM 系列等等。嵌入式微控制器又称为单片机，它将 CPU、存储器（少量的 RAM、ROM 或两者都有）和其他外设封装在同一片集成电路里。常见的有 8051 单片机。嵌入式 DSP 专门用来对离散时间信号进行极快的处理计算，提高编译效率和执行速度。在数字滤波、FFT、谱分析、图像处理的分析等领域，DSP 正在大量进入嵌入式市场。

嵌入式系统具有微内核结构。操作系统分为内核层和应用层。内核层提供进程管理、文件系统、设备管理等基本功能。应用层基于内核层开发,同时立足于应用。嵌入式系统一般采用微内核结构。微内核只提供最基本的功能,大量的有价值的内容存在于应用层,以便用户裁减。

任务调度。嵌入式操作系统一般支持多任务。多任务靠CPU在多个任务之间切换、调度来实现。任务有优先级,不同的任务,优先级可能相同,也可能不同。任务调度有三种方式:可抢占式调度、不可抢占式调度和时间片轮转调度。可抢占式调度基于任务的优先级,当前正在运行的任务可以随时让位给优先级更高的处于就绪态的其他任务;不可抢占式调度是指一个任务一旦获得CPU,就独占CPU运行,除非由于某种原因,它决定放弃CPU的使用权;时间片轮转调度是指当两个或两个以上任务有同样的优先级,不同任务轮转地使用CPU,直到系统分配的CPU时间片用完。目前,大多数嵌入式操作系统对不同优先级的任务采用基于优先级的抢占式调度法,对相同优先级的任务则采用时间片轮转调度法。

硬实时和软实时。有些嵌入式系统是硬实时系统,有些嵌入式系统是软实时系统。硬实时系统对系统响应时间有严格要求,一旦系统响应时间不能满足,就可能引起系统崩溃或产生致命错误;软实时系统并不要求限定任务必须在一定的时间内完成,只要求任务运行得越快越好。

内存管理。绝大多数嵌入式系统采用实存储器管理策略。由于实存储器管理要求直接访问内存,同时,用户程序、内核及其他用户程序共存于一个地址空间中,所以,用户程序开发时要保证它们和内核之间互不侵犯地址空间,以免破坏内核正常工作,或导致用户程序运行异常。

内核加载方式。嵌入式操作系统内核可以在Flash上直接运行,也可以加载到内存中运行。Flash的运行方式是把内核的可执行映像烧写到Flash上,系统启动时从Flash的某个地址开始执行。内核加载方式是把内核的压缩文件存放在Flash上,系统启动时读取压缩文件在内存里解压,然后开始执行。这种方式相对复杂一些,但是运行速度可能更快,因为RAM的存取速率要比Flash高。

另外,嵌入式系统同通用型计算机系统相比,具有以下特点:

嵌入式CPU是面向特定应用的。嵌入式CPU与通用型计算机CPU的最大不同就是嵌入式CPU是为特定用户群设计的,它具有功耗低、体积小、集成度高等特点,能够把通用CPU中许多由板卡完成的任务集成在芯片内部,从而有利于嵌入式系统设计小型化,移动能力大大增强,跟网络的耦合也越来越紧密。

嵌入式系统是先进计算机技术、半导体技术和电子技术在各行业中具体应用后形成的产物。

嵌入式系统的硬件和软件需要高效率地设计,量体裁衣,去除冗余,力争在同样的硅片面积上实现更高的性能。

嵌入式系统和具体应用有机地结合在一起,它的升级换代也和具体产品同步进行,因而嵌入式产品一旦进入市场,就具有较长的生命周期。

为了提高执行速度和系统可靠性,嵌入式系统中的软件一般都固化在存储器芯片或单片机本身中,而不是存储于磁盘等载体中。

嵌入式系统本身不具备自举开发能力,即使设计完成以后,用户通常也不能对其中的程序

功能进行修改，必须有一套开发工具和环境才能进行开发。

3. **嵌入式处理器：ARM**

ARM(Advanced RISC Machines)既可以被认为是一个公司的名字，也可以被认为是对一类微处理器的通称，还可以被认为是一种技术的代名词。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。目前，采用 ARM 技术知识产权(IP)核的微处理器，即通常所说的 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 技术的微处理器的应用已经占据了 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方面。世界各大半导体生产商从 ARM 公司购买其设计的 ARM 微处理器核后，根据各自不同的应用领域，加入适当的外围电路，从而形成自己的 ARM 微处理器芯片进入市场。

到目前为止，ARM 微处理器及技术的应用几乎已经深入到各个领域：

工业控制领域：作为 32 位的 RISC 架构，基于 ARM 核的微控制器芯片不但占据了高端微控制器市场的大部分市场份额，同时也逐渐向低端微控制器应用领域扩展。ARM 微控制器的低功耗、高性价比，向传统的 8 位/16 位微控制器提出了挑战。

无线通信领域：目前已有超过 85% 的无线通信设备采用了 ARM 技术，ARM 以其高性能和低成本，在该领域的地位日益巩固。

网络应用：随着宽带技术的推广，采用 ARM 技术的 ADSL 芯片正逐步获得竞争优势。

消费类电子产品：ARM 技术在目前流行的数字音频播放器、数字机顶盒和游戏机中得到广泛采用。ARM 在语音及视频处理上进行了优化，并获得广泛支持，同时对 DSP 应用领域提出了挑战。

成像和安全产品：现在流行的数码相机和打印机，绝大多数采用 ARM 技术。手机中的 32 位 SIM 智能卡也采用了 ARM 技术。

除此以外，ARM 微处理器及技术还应用到了许多不同的领域，并会在将来取得更加广泛的应用。

ARM 微处理器一般具有如下特点：

体积小、低功耗、低成本、高性能；

支持 Thumb(16 位)/ARM(32 位)双指令集，能很好地兼容 8 位/16 位器件；

大量使用寄存器，指令执行速度更快；

大多数数据操作都在寄存器中完成；

寻址方式灵活简单，执行效率高；

指令长度固定。

目前，ARM 微处理器包括下面几个系列。它们除具有 ARM 体系结构的共同特点以外，每个系列的 ARM 微处理器都有各自的特点和应用领域。

ARM7 系列、ARM9 系列、ARM9E 系列、ARM10E 系列、SecurCore 系列、Inter 的 Xscale、Inter 的 StrongARM。其中，ARM7 系列、ARM9 系列、ARM9E 系列和 ARM10E 系列为 4 个通用处理器系列，它们分别提供一套相对独特的性能来满足不同应用领域的需求。SecurCore 系列专门为安全要求较高的应用而设计。详细叙述如下：

ARM7 系列微处理器为低功耗的 32 位 RISC 处理器，最适合对价位和功耗要求较高的消费者应用。ARM7 微处理器系列具有如下特点：

具有嵌入式 ICE－RT 逻辑，调试开发方便；

极低的功耗，适合对功耗要求较高的应用，如便携式产品；

能够提供 0.9MIPS/MHz 的 3 级流水线结构；

代码密度高并兼容 16 位的 Thumb 指令集；

对操作系统的支持广泛，包括 Windows CE、Linux、Palm OS 等；

指令系统与 ARM9 系列、ARM9E 系列和 ARM10E 系列兼容，便于用户的产品升级换代；

主频最高可达 130MIPS，高速的运算处理能力使其能胜任绝大多数的复杂应用。

ARM7 系列微处理器的主要应用领域为：工业控制、Internet 设备、网络和调制解调器设备、移动电话等多种多媒体和嵌入式应用。ARM7 系列微处理器包括如下几种类型：ARM7TDMI、ARM7TDMI-S、ARM720T、ARM7EJ。其中，ARM7TDMI 是目前使用最广泛的 32 位嵌入式 RISC 处理器，属低端 ARM 处理器核。TDMI 的基本含义为：

T：支持 16 位压缩指令集 Thumb；

D：支持片上 Debug；

M：内嵌硬件乘法器(Multiplier)；

I：嵌入式 ICE，支持片上断点和调试点。

ARM7 系列性能特征见表 1-1。

表 1-1　ARM7 系列性能特征表

性能特征 / 类型	Cache 大小（指令/数据）	紧密耦合存储器(TCM)	存储器管理	AHB 总线接口	Thumb	DSP	Jazelle
ARM7TDMI	无	无	无	有	有	无	无
ARM7TDMI-S	无	无	无	有	有	无	无
ARM7EJ	无	无	无	有	有	有	有
ARM720T	8KB	无	MMU	有	有	无	无

ARM9 系列微处理器在高性能和低功耗特性方面提供了最佳性能。它们具有以下特点：

5 级整数流水线，指令执行效率更高；

提供 1.1MIPS/MHz 的哈佛结构；

支持 32 位 ARM 指令集和 16 位 Thumb 指令集；

支持 32 位的高速 AMBA 总线接口；

全性能的 MMU，支持 Windows CE、Linux、Palm OS 等多种主流嵌入式操作系统；

MPU 支持实时操作系统；

支持数据 Cache 和指令 Cache，具有更高的指令和数据处理能力。

ARM9 系列微处理器主要应用于无线设备、仪器仪表、安全系统、机顶盒、高端打印机、数字照相机和数字摄像机等。ARM9 系列微处理器包括 ARM920T、ARM922T 和 ARM940T 三种类型，适用于不同的应用场合。ARM9 系列性能特征见表 1-2。

表 1-2 ARM9 系列性能特征表

性能特征 / 类型	Cache 大小（指令/数据）/KB	紧密耦合存储器(TCM)	存储器管理	AHB 总线接口	Thumb	DSP	Jazelle
ARM920T	16/16	无	MMU	有	有	无	无
ARM922T	8/8	无	MMU	有	有	无	无
ARM940T	4/4	无	MMU	有	有	无	无

ARM9E 系列微处理器为可综合处理器。它们使用单一处理器内核，减少了芯片面积和系统复杂程度，提供了增强的 DSP 处理能力，适合于同时使用 DSP 和微控制器的应用场合，也适合于 Java 应用系统。其主要特点如下：

支持 DSP 指令集，适合于需要高速数字信号处理的场合；

5 级整数流水线，指令执行效率更高；

支持 32 位 ARM 指令集和 16 位 Thumb 指令集；

支持 32 位的高速 AMBA 总线接口；

支持 VFP9 浮点处理协处理器；

全性能的 MMU，支持 Windows CE、Linux、Palm OS 等多种主流嵌入式操作系统；

MPU 支持实时操作系统；

支持数据 Cache 和指令 Cache，具有更高的指令和数据处理能力；

主频最高可达 300MIPS。

ARM9E 系列微处理器主要应用于下一代无线设备、数字消费品、成像设备、工业控制、存储设备和网络设备等领域。ARM9E 系列微处理器包括 ARM926EJ-S、ARM946EJ-S 和 ARM966EJ-S 三种类型，以适用于不同的应用场合。ARM9E 系列性能特征见表 1-3。

表 1-3 ARM9E 系列性能特征表

性能特征 / 类型	Cache 大小（指令/数据）	紧密耦合存储器(TCM)	存储器管理	AHB 总线接口	Thumb	DSP	Jazelle
ARM926EJ-S	4～128KB/4～128KB	有	MMU	双 AHB	有	有	有
ARM946EJ-S	4KB～1MB/4KB～1MB	有	MMU	AHB	有	有	无
ARM966EJ-S	无	有	无	AHB	有	有	无

ARM10E 系列微处理器采用了新体系结构和先进的节能方式，与同等的 ARM9 系列器件相比，在同样的时钟频率下，性能提高了近 50%，同时，功耗极低。其主要特点如下：

支持 DSP 指令集，适合于需要高速数字信号处理的场合；

6 级整数流水线，指令执行效率更高；

支持 32 位 ARM 指令集和 16 位 Thumb 指令集；

支持 32 位的高速 AMBA 总线接口；

支持 VFP10 浮点处理协处理器；

全性能的 MMU，支持 Windows CE、Linux、Palm OS 等多种主流嵌入式操作系统；

支持数据 Cache 和指令 Cache,具有更高的指令和数据处理能力;

主频最高可达 400MIPS;

内嵌并行读/写操作部件。

ARM10E 系列微处理器主要应用于下一代无线设备、数字消费品、成像设备、工业控制、通信和信息系统等领域。ARM10E 系列微处理器包括 ARM1020E、ARM1022E 和 ARM1026E 三种类型,以适用于不同的应用场合。ARM10E 系列性能特征见表 1-4。

表 1-4 ARM10E 系列性能特征表

性能特征 / 类型	Cache 大小(指令/数据)/KB	紧密耦合存储器(TCM)	存储器管理	AHB 总线接口	Thumb	DSP	Jazelle
ARM1020E	32/32	无	MMU	双 AHB	有	有	无
ARM1022E	16/16	无	MMU	双 AHB	有	有	无
ARM1026E	可变	有	MMU + MMU	双 AHB	有	有	有

SecurCore 系列微处理器提供了完善的 32 位 RISC 技术的安全解决方案,具有 ARM 体系结构低功耗、高性能等特点。概括起来,具有如下特点:

带有灵活的保护单元,以确保操作系统和应用数据的安全;

采用软内核技术,防止外部对其进行扫描探测;

可集成用户自己的安全特性和其他协处理器。

SecurCore 系列微处理器主要用于一些对安全性能要求较高的应用产品及应用系统,如电子商务、电子政务、电子银行业务、网络和认证系统等领域。SecurCore 系列微处理器包括 SecurCore SC100、SecurCore SC110、SecurCore SC200 和 SecurCore SC210 四种类型,以适用于不同的应用场合。SecurCore 系列性能特征见表 1-5。

表 1-5 ARM SecurCore 系列性能特征表

性能特征 / 类型	Cache 大小(指令/数据)	紧密耦合存储器(TCM)	存储器管理	AHB 总线接口	Thumb	DSP	Jazelle
SecurCore SC100	无	无	MMU	无	有	无	无
SecurCore SC110	无	无	MMU	无	有	无	无
SecurCore SC200	可选	无	MMU	无	有	有	有
SecurCore SC210	可选	无	MMU	无	有	有	有

Xscale 处理器基于 ARMv5TE 体系结构的解决方案,是一款全性能、高性价比、低功耗的处理器。它支持 16 位的 Thumb 指令和 DSP 指令集,已被使用在数字移动电话、个人数字助理和网络产品等场合。Xscale 处理器是 Inter 目前主要推广的一款 ARM 微处理器。

Inter StrongARM SA-1100 处理器是采用 ARM 体系结构、高度集成的 32 位 RISC 微处理器。它融合了 Inter 公司设计、处理技术,以及 ARM 体系结构的电源效率。在软件上采用既兼容 ARMv4 体系结构,又具有 Inter 技术优点的体系结构。基于 ARM 的 Inter 处理器性能特征见表 1-6。

表 1-6　基于 ARM 的 Inter 处理器性能特征表

性能特征 / 类型	Cache 大小（指令/数据）/KB	紧密耦合存储器(TCM)	存储器管理	AHB 总线接口	Thumb	DSP	Jazelle
XScale	32/32	无	MMU	N/A	有	有	无
StrongARM	16/8	无	MMU	N/A	无	无	无

4. 开发嵌入式系统的技术

嵌入式系统的开发技术有自己独特的特征，可以从下述两个方面进行描述。

（1）与嵌入式系统开发过程相关的技术

嵌入式系统开发与 Windows 系统开发有所不同。首先，嵌入式系统开发中使用的设备有明显的角色之分。执行编译、链接、定址过程的计算机叫宿主机；运行嵌入式软件的硬件平台叫目标机。应用程序在宿主机上通过交叉编译器、连接器等编译、链接、定址，转换为目标机的二进制代码程序。交叉编译器、连接器就是运行在一个计算机平台上的，为产生另一个平台代码的编译器、连接器。其次，目标机上应用程序的调试采用交叉调试器，使用宿主机-目标机的调试方式。宿主机和目标机之间用串口线或以太网线或 BDM 线相连。交叉调试有任务级、源码级和汇编级调试，调试时需将宿主机上的应用程序和操作系统内核下载到目标机的 RAM 中或直接烧录到目标机的 ROM 中。目标监控器是调试器对目标机上运行的应用程序进行控制的代理，事先被固化在目标机的 Flash、ROM 中。它在目标机上电后自动启动，并等待宿主机调试器发来的命令，配合调试器完成应用程序的下载、运行和基本的调试功能，将调试信息返回给宿主机。嵌入式目标系统生成与调试方式见图 1-1。

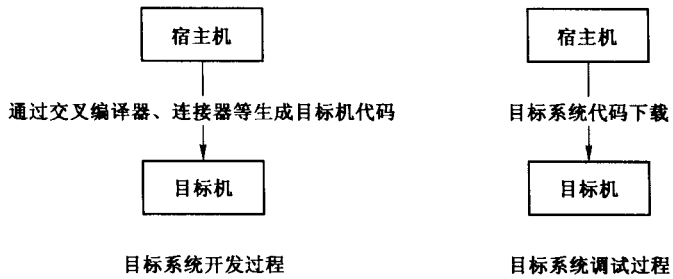

图 1-1　嵌入式目标系统生成和调试过程中宿主机和目标机的关系

（2）向嵌入式平台移植软件的相关技术

一般情况下，嵌入式应用程序都是先在宿主机上编写，经编译、连接生成目标机上二进制目标代码后，再移植到目标机上。在宿主机上编写软件时，必须注意软件的可移植性。具体操作中，需要选用具有较高移植性的编程语言，并尽量少调用操作系统函数，注意屏蔽不同硬件平台带来的字节顺序、字节对齐等问题。下面是一些在编制嵌入式应用程序时需要注意的问题。

字节顺序。内存中，多于一个字节的内存数据类型的数据存放需要满足一定的顺序。这种顺序通常有小端和大端两种。小端字节序指低字节数据存放在内存低地址处，高字节数据存放在内存高地址处；大端字节序是指高字节数据存放在低地址处，低字节数据存放在高地址处。

字节对齐。某些嵌入式处理器寻址方式决定了内存中占2字节的int16、uint16等类型数据只能存放在偶数内存地址处，占4字节的int32、uint32等类型数据只能存放在4的整数倍的内存地址处；占8字节的类型数据只能存放在8的整数倍的内存地址处，占1字节的类型数据可以存放在任意地址处。所以，不同的嵌入式处理器平台要求不同的编程方式。在针对通用平台的编程中，解决字节对齐方法有许多种，比如在GCC的项目管理文件Makefile中增加编译参数--pack-struct；增加组包函数和拆包函数等。组包函数和拆包函数解决字节对齐问题的原理是，通过在发数据包处增加组包函数，使数据按CPU的要求组织放在指定地址处，组成包发给下层，而后在收数据包处增加拆包函数，使收到的内存中数据再按要求还原。

位段。由于位段的空间分配方向因硬件平台的不同而不同，因此分配顺序的不同可导致数据存取的错误。解决这一问题的一种方法是采用条件编译方式，针对不同的平台定义顺序不同的位段；也可以在组包函数和拆包函数中加上位段处理。

代码优化。在嵌入式系统的开发中，代码优化可以减少代码尺寸，尽可能提高代码执行效率。代码优化的方法有：①提高代码的执行效率。比如程序经常使用switch-case语句，每一个由机器语言实现的测试和跳转仅仅是为了决定下一步要做什么，这就浪费了处理器时间。为了提高速度，可以把具体的情况按照它们发生的相对频率排序，把最可能发生的情况放在第一，最不可能发生的情况放在最后，就会减少平均的代码执行时间。再如，使用全局变量比向函数传递参数更有效率。使用全局变量可以除去函数调用前参数入栈和函数完成后参数出栈的操作，从而提高代码执行速度。②尽可能地减小代码的尺寸，并避免使用标准库例程。很多大的标准库例程总是设法处理所有可能的情况，从而占用了庞大内存空间，增加了代码尺寸，同时降低了代码执行效率。③避免内存泄漏。为使用户程序有足够的内存运行空间，必须让用户程序把申请到的内存在不用后及时返还系统，以确保有足够的内存空间供再次申请。如果程序没有及时释放所占用的内存，即存在内存泄漏，一些程序会因为没有足够的内存空间而无法运行。

5. 嵌入式系统的应用

嵌入式系统应用前景非常广阔。生活中的洗衣机、电冰箱、自行车、小汽车，甚至办公室里的远程会议系统等等，这些都可能是嵌入式系统。以蓝牙为代表的小范围无线接入协议的出现，使嵌入式无线电的概念悄然兴起。家中、办公室、公共场所里，人们可能使用数十片甚至更多的嵌入式无线电芯片，将一些电子信息设备或电气设备构成无线网络。在车上、旅途中，人们利用这样的嵌入式无线电芯片可以实现远程办公、远程遥控，真正实现把网络随身携带。下面是一些嵌入式系统的具体应用。

嵌入式移动数据库。移动数据库就是支持移动计算的数据库，它有两层含义：用户在移动的过程中可以联机访问数据库资源；用户可以带着数据库移动。典型的应用是在开着的救护车上查询最近的医院。

网络设备。这种设备包括路由器、交换机、Web server、网络接入盒等各种网络设备。比如，利用这些设备可以构建家居网络，在一个家中建立一个通信网络，为家庭信息提供必要的通路。在家庭网络操作系统的控制下，通过相应的硬件和执行机构，实现对所有家庭网络上的家电和设备的控制和监测。

嵌入式语音芯片。嵌入式语音芯片是通过采用语音识别和语音合成、语音学层次结构体系和文本处理模型等技术，以实现手持设备、智能家电等多个领域利用语音进行人性化的交互