DSP应用丛书

TI-DSP

多核技术及实时软件开发

潘晔 廖昌俊 编著

中国工信出版集团

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

TI－DSP 多核技术及实时软件开发

潘　晔　廖昌俊　编著

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

内 容 简 介

本书从 DSP 软件开发的各个角度阐述了 TI 公司提供的 DSP 软件技术和开发工具，为 DSP 软件开发人员理清思路，以简化和加快 DSP 系统的软件开发。第 1 章首先从宏观上讨论了 DSP 嵌入式系统软件开发应注意的要素，然后简介了 TI 公司的 eXpressDSP 实时软件组件和开发工具。第 2 ~ 5 章分别从 DSP 可重用实时软件技术、多核嵌入式软件开发、优化的 DSP 库，以及 DSP 软件开发工具等几方面进行了详细介绍。

本书所涉及的材料，是截止到 2014 年的最新资料。结合编者的项目开发经验，增加了实现的例子，有利于读者理解和应用。本书可以作为电子信息类专业研究生和高年级本科生的参考书，对从事 DSP 技术研究和开发的科研人员和工程技术人员也很有参考价值。

图书在版编目（CIP）数据

TI - DSP 多核技术及实时软件开发/潘晔，廖昌俊编著 . —北京：电子工业出版社，2016. 1
（DSP 应用丛书）
ISBN 978-7-121-27635-4

Ⅰ. ①T… Ⅱ. ①潘… ②廖… Ⅲ. ①数字信号处理 - 程序设计 Ⅳ. ①TN911. 72

中国版本图书馆 CIP 数据核字（2015）第 281631 号

责任编辑：曲 昕 特约编辑：张传福
印 刷：北京季蜂印刷有限公司
装 订：北京季蜂印刷有限公司
出版发行：电子工业出版社
北京市海淀区万寿路 173 信箱 邮编 100036
开 本：787 × 1 092 1/16 印张：20. 25 字数：531 千字
版 次：2016 年 1 月第 1 版
印 次：2016 年 1 月第 1 次印刷
定 价：49. 80 元

凡所购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@ phei. com. cn，盗版侵权举报请发邮件至 dbqq@ phei. com. cn。

服务热线：（010）88258888。

序

自 20 世纪 70 年代末 80 年代初，DSP 处理器诞生以来，其发展与推广应用神速，在短短的 30 多年时间内，应用的领域和深度，令人叹为观止。

随着科学技术的进步，尤其是微电子和软件科学与技术的发展，微处理器的种类、型号与性能的发展，只能用“眼花缭乱”来形容。各类微处理器之间的技术融合、功能交叠，一方面使人们有了更多的选择余地，但也使得制订系统方案时的选择出现了方方面面的困难，主要是权衡利弊、取优舍劣、软硬件性能与开发难易程度的选择，性能价格比的考虑，以及发展前景的预测等。

微处理器（包括 DSP 处理器）硬件性能的极大改善，为软件的开发提供了很大的余地和空间；软件技术的进步也为微处理器软件的开发提供了极大的方便。特别值得一提的是编译器的优化，极大地提高了高级语言编译的效率，使其结果的优化程度，可以和直接用汇编语言程序的编译结果相媲美。也就是说，编程人员完全可以从烦琐的汇编语言编程工作中解放出来，使用自己熟悉的高级语言来编程，工作难度的降低和效率的提高，不言而喻。

为了推广自己公司的产品，各微处理器厂商还不断地推出、更新和优化自己系列处理器的开发工具和算法库，使应用系统的软件开发人员得以方便和高效地开展工作。

仍然存在不方便的是，不同厂商微处理器的硬件和软件系统，以及开发环境和工具，各不相同。因此，应用系统的开发人员，在系统开发之初，必须谨慎地选择所要使用的微处理器；不但是这一代产品，还得考虑后续的产品，因为改变所使用的微处理器，成本极为高昂，除了硬件、软件和开发环境的成本，还有开发所投入的人力成本，以及推迟新产品上市的机会成本，等等。

本书是针对德州仪器（TI）公司的 DSP 软件开发而编写的。如上所述，各家公司的硬件系统、软件系统和开发环境，各不相同。即便如此，基本的思路和方法还是一致的。有经验的开发人员都有这样的体会，真正熟悉了一家公司的东西，即便改用其他公司的产品上手也很容易，就是这个道理，即所谓“举一反三”。

参与本书编写的几位作者，都是电子科技大学的教师，是在 DSP 技术领域工作多年的资深教师和研究人员。他们有很好的“数字信号处理”的理论功底，熟悉 DSP 的硬件系统、软件系统和开发环境与工具，完成过多项包含 DSP 处理器的复杂系统的研制，因此，他们拥有完善的相关知识，积累了丰富的工作经验。

本书以实用为目的，基于作者的 DSP 工程开发经验，从 TI 公司纷繁复杂的文档中整理出有利于工程人员开发 DSP 系统的体系，为 DSP 软件开发人员理清思路。我相信，认真阅读和学习本书的读者，一定可以从中获得丰富的知识和体会，并在自己的学习和开发工作中，得益良多。

彭启琮

2015 年 11 月于

电子科技大学

前　言

随着数字信号处理（DSP）技术的发展，其应用无处不在。各种丰富多彩的多媒体智能终端带给人们方便快捷的应用体验，人们可以随时访问网络、处理音频和视频、规划交通路线等。除了上述消费类电子设备，工业控制、安防系统、通信系统、医疗设备、航天航空、军事装备等各方面都离不开 DSP。因此，DSP 软硬件开发以及系统集成等成为人们关注的问题。在通常情况下，开发一个 DSP 嵌入式系统，80% 的努力及 80% 的复杂度均取决于软件；如何提高 DSP 软件的开发速度、降低开发难度和成本至关重要。

目前，DSP 芯片的功能越来越复杂，多核片上系统（SOC）普遍应用，外设种类越来越多，大量新技术标准、新算法、新应用层出不穷。开发人员要花很长的时间来熟悉各种标准，而这些标准又在不停地改变。已有的设备和系统往往和特定的软硬件紧紧地联系在一起，很难升级和维护。开发人员常常面临不同方面的技术难题，还要重复开发类似的算法，既耗时又使成本增加。有些看似细节的问题，所涉及的处理方案可能影响整个系统，解决起来也较为复杂。用户所期待的是不用考虑产品所采用技术的不同，开发者也不希望陷入耗时费力的技术细节之中。因此，DSP 芯片的主要供应商（如 TI 公司）提供了一系列可重用的实时软件开发框架、组件、库，以及适应 SOC 的多核通信组件、编解码算法、网络开发包等。

而且，对于 DSP 工程师而言，选择一个优秀的软件开发工具将大大地加快整个开发的进度，成为帮助开发和调试的有利手段。Code Composer Studio（CCS）是 TI 公司嵌入式处理器系列的集成开发环境（IDE），也是目前使用最为广泛的 DSP 开发软件之一。CCS 以 Eclipse开源软件框架为基础。CCS 将 Eclipse 软件框架的优点和 TI 先进的嵌入式调试功能相结合，为嵌入式开发人员提供了功能丰富的开发环境。

由以上讨论可知，现代复杂的 DSP 嵌入式系统的开发已经不再是开发人员从头开始编写所有的软件，而是以成熟的框架和算法库为基础，充分利用开发工具，才能又快又好地完成；开发人员也不是独立完成整个系统，而是分工合作，可分成算法开发人员、系统集成开发人员，以及底层驱动开发人员等。

本书的目的就是从 DSP 软件开发的各个角度阐述 TI 公司提供的 DSP 软件和开发工具，为 DSP 软件开发人员理清思路，以简化和加快 DSP 系统的软件开发。本书系统地阐述了德州仪器（TI）公司的数字信号处理器（DSP）和多核片上系统（SOC）的相关软件技术，包括可重用的软件开发框架、实时操作系统内核、算法和多媒体库，以及适应 SOC 的多核通信组件，网络开发包等。全书分为五章，第 1 章讨论 DSP 嵌入式软件开发应注意的要素；第 2 章从 XDAIS 算法标准和三种参考编程框架等方面讨论 DSP 可重用实时软件技术；第 3 章从 DSP/BIOS 实时内核、网络开发包（NDK）、设备驱动包（DDK）和多核通信组件等方面讨论多核嵌入式软件开发；第 4 章讨论了优化的 DSP 库，包括算法库、数学库、图像处

理库以及音视频编解码；第 5 章介绍了 DSP 软件开发工具——Code Composer Studio（CCS）。

本书所涉及的材料，是截止到 2014 年的最新资料。在全面整理 TI 公司相关资料的基础上，结合编者的项目开发经验，增加了实现的例子，有利于读者理解和应用。

本书是在彭启琮教授的主导下，由潘晔和廖昌俊完成的。两位编著者均完成过大量的 DSP 软硬件工程项目，对 TI 公司的 DSP 软件和开发工具十分熟悉。其中潘晔编写了第 1、2 章和第 3 章的 3.1、3.2 节，并对全书统稿；廖昌俊编写了第 4、5 章和第 3 章的 3.3、3.4 节。管庆教授提供了第 5 章的实例和重要参考意见，教研室多位研究生同学参加的项目也作为本书的应用举例，在此表示感谢。

DSP 技术发展日新月异，应用广泛，新的软件技术和开发工具层出不穷。本书选择介绍的内容难免存在不当和错误，敬请读者反馈意见和批评指正。

编著者

2015 年 11 月于电子科技大学

目　录

第1章 绪 论

数字信号处理（DSP）的应用无处不在，包括消费类电子设备、工业控制、安防系统、通信系统、医疗设备、航天航空、军事装备等方方面面，而且 DSP 芯片以及多核片上系统（SOC）的功能越来越复杂，外设种类越来越多，大量新技术标准、新算法、新应用层出不穷。因此，开发一个 DSP 嵌入式系统，应以成熟的框架和算法库为基础，充分利用开发工具，才能降低开发难度和成本。

本章首先从宏观上讨论 DSP 嵌入式系统软件开发应注意的要素，然后简介 TI 公司的 eXpressDSP 实时软件组件和开发工具。

1.1 DSP 嵌入式软件开发要素

DSP 嵌入式系统的软件开发是一项极具挑战性的任务，需要考虑大量相互关联的要素，才能做出优化与权衡。只有事先做好规划，才能完成这一挑战；否则很容易造成开发时效率低下，甚至推倒重来。开发人员首先应明确一个基本问题：此项目的目标是什么？然而这一问题并不容易回答。明确了项目的目标意味着完全理解了影响此项目的各方面要素，最终产品就会在市场上取得成功。实际上，这些要素直接决定了开发人员的多数决策。例如，产品上市的时间和利润点决定了开发人员选用何种开发工具，是否将开源代码或商业软件集成到系统中，是否采用以及采用哪种操作系统等。下面将讨论以下几方面要素：操作系统、图形化与人机交互、安全性、开发工具与开源代码、代码结构、中间件和软件框架、多媒体编程、多处理器或多核 SOC 等。

1.1.1 操作系统

DSP 嵌入式系统软件开发的首要问题是考虑是否采用操作系统。有的系统很简单、功能有限，就不需要操作系统；有些微控制器也不采用操作系统，而是采用一些简单的编程接口。TI 公司为自己的 ARM 处理器 DSP 处理器提供了名为 StarterWare 的开发套件。StarterWare 是基于 C 语言的，不需要操作系统的支持；包括设备抽象层库以及外设编程的例子，例如网络、图形、USB 等。

如果需要操作系统，可选择的种类也很广泛。到底是选择购买 Windows CE，还是选择开源的 Linux，或者选择免版税但封闭的 Android？是选择高层操作系统（HLOS）还是实时操作系统（RTOS）？选择操作系统是首要任务，因为操作系统直接影响其他部分软件的开发。

1. 商业 OS、开源 OS 及免版税 OS

采用商业 OS 需要向所有者付版税，购买了商业 OS 可以得到该 OS 所有者公司的技术支持和培训，从而缩短学习路线，克服不可避免的困难，加快软件开发过程。而且，由于商业

OS 完全由其所有者公司控制，第三方工具和插件等通常是经过该公司评估和认证的，可以方便地集成到大型软件系统中。

开源 OS 可以免费获得，其中 Linux 是最著名的。开发者社区为开源 OS 提供支持和插件。如果开发人员采用了开源 OS，就需要自己负责集成、评估和测试操作系统以及第三方工具和插件，以保证所有软件能在一起正确工作。采用开源 OS 的好处是可以免费从开发者社区获得创新的软件模块，许多有创造力的开发者会定期向开发者社区贡献自己的成果。另外，许多操作系统厂商，如 MontaVista、Red Hat、RidgeRun、Timesys、Wind River 等开发了商业版的 Linux。

Android 是免版税 OS，可以从 Google 免费获得，但其内容和架构仍然是 Google 控制的，例如，Android 也基于 Linux，但 Google 决定集成哪些内容和插件以及多媒体处理软件等。Android 的封闭虽然限制了开发的灵活性，但简化和无缝集成了第三方代码，从而加快了系统开发过程。

2. RTOS 与 HLOS

选择操作系统需要考虑的另一个重要方面是待开发的应用系统是否是实时系统。RTOS 是专门为实时系统设计的，可以满足确定的响应时间。实际上，确定的响应时间对一些实时系统至关重要，甚至关乎生死。例如，汽车的刹车系统必须在踩下刹车时立即响应，否则就会有生命危险。

通用的 HLOS 一般把命令放到队列中，然后在适当的时候再响应；而 RTOS 必须能识别紧急命令并立即响应。正是由于 RTOS 主要关注响应的实时性，通常 RTOS 不能支持类似 Windows 系统的通用 HLOS 的广泛功能；RTOS 支持的一般局限于实时系统中最主要的时间敏感性任务。

通用 HLOS 的关注重点是便于使用，故 HLOS 比 RTOS 有覆盖面更宽和更深的抽象层，以隔离用户和硬件处理器或通信系统等。而且 HLOS 通常假定用户要处理很多任务，因此 HLOS 比 RTOS 功能复杂得多，代码量也大很多。反过来，这也使得 HLOS 不适合对响应时间有确定要求的实时处理应用。

1.1.2　图形化与人机交互

人机交互也是 DSP 嵌入式系统软件项目需要考虑的问题。在项目的最初设计阶段，就要考虑是否需要图形化用户接口（GUI）。人机接口（HMI）或 GUI 在消费电子系统中很流行。在嵌入式系统中用户也希望有条件地采用这类直观的接口，以方便用户，提高系统的易用性以及总体操作效率。而且，最终系统在市场上是否受欢迎也与广告描述的用户接口相关。

嵌入式系统的图形处理包括三个不同的任务：创建、构图和显示。首先是生成图形元素，比如窗口、图标、按钮等；其次是设计不同窗口、桌面和其他图形元素的构图，即这些元素在屏幕上看起来是什么样子以及当系统运行时它们如何改变；最后是按某种顺序在显示器上实际显示图形元素。

显然，当用户接口越来越复杂时，图形设计和处理也越来越复杂。软件开发人员为一个特定的嵌入式系统开发用户接口时会在很大程度上依赖系统的硬件能力。例如，大部分消费电子系统配有专门的图形或多媒体处理器，而无专门图形处理器的嵌入式系统可能需要硬件

加速器，才能分担系统主处理器的图形处理运算量，从而改善系统的图形响应性能。用户接口会影响用户对系统整体响应时间的感受，也就是说，一个反应迟钝的用户接口会给用户造成整个系统运行糟糕的印象。

对软件设计人员而言，一个需要考虑的关键问题是，一个特别的 HMI 或 GUI 对于系统在市场上的成功应用是否是绝对必要的。当然，用户接口对于许多消费电子设备或系统获得成功起了重要作用，但对于嵌入式系统就不是那么重要了。嵌入式系统的用户接口设计者需要在多方面进行权衡，分析各种图形和显示方案隐含的成本，确定满足系统成本、功能和性能的最优组合。另外需要考虑的问题包括：自行开发用户接口还是外聘专业的开发人员；需要多大的存储空间等。

现在已经有很多图形界面开发工具，包括开源的和商业套件。OpenGL 是一种应用广泛的跨平台图形框架，独立于任何 OS，由 khronos. org 提供支持。另外，有很多 Linux 平台的图形开发框架，如：Qt、X－11、GTK 和 DirectFB，每种框架都有多种实现子集，以及不同层次的硬件抽象。由于 Linux 本身不包括这些图形框架，因此对于没有经验的开发者，要在一种特定的嵌入式处理器上集成并运行 Linux 和这些图形框架是很耗费时间的。除了开源软件，一些商业 OS 也包括图形框架，如 WindowsCE、VxWorks、QNX 等；还有独立于 OS 的图形框架 Mentor Graphics' Inflexion，既可以运行于开源 OS 也可以运行于封闭 OS。

1.1.3 安全性

嵌入式系统如何考虑安全性？还是完全不需要关注安全方面？实际上，在面对黑客、病毒以及恶意程序时，电子设备是很脆弱的。在可预见的将来，安全性将是业界持续关注的主要问题。存储了用户私有信息的嵌入式系统不会对身份盗窃者、窜改者、恶意攻击以及匿名恶作剧免疫，而如医疗系统和紧急应答通信这些类型的嵌入式系统对个人和公众安全十分关键。因此，嵌入式开发人员必须在项目刚一开始就做好计划以保护系统，即通过设计适当的安全措施来保护操作和存储的信息。

对于保护嵌入式设备，理解黑客的目的是极重要的。许多设备远离安全设施，放在很容易受攻击的位置，更需要强健有效的安全措施。开发人员思考以下问题有助于集中注意力，部署和开发适当的方法为最关键的部分提供足够的保护。

- 什么是需要保护的？
- 谁是安全措施要对抗的？
- 如果试图保护的信息被泄露了，代价是什么？

当然，这些问题的答案会根据最终应用发生变化。例如，销售点终端要保护存储在终端上的以及通过连接到终端的通信信道发送的信用卡信息；媒体播放器要保护包含数字版权的内容的安全以免其被盗版；语音和数据通信终端要保护个人身份不被盗窃以免隐私被非法侵犯。深入回答这些问题会使开发人员的决定全面而有效，开发人员会找到什么数据必须保密以及哪些通信信道必须加密。系统中存在的第三方应用程序也必须被评估，必须充分讨论其可能的安全隐患。

既然不同的系统要求不同级别的安全性，在一些系统中经常采用的安全技术在另外的系统中却不采用。例如，在大众消费应用中，密码加速、安全启动和保护系统调试通道等构成了最低程度的安全保护，而在嵌入式应用中需要另外的安全措施来保护运行时环境以避免窜

改电压和时钟，以及保护各种通信信道。

1.1.4　开发工具

通常处理器供应商会提供一些开发工具和资源，包括：集成开发环境（IDE）、硬件评估模块/开发板（EVM）、软件开发包（SDK）等。这些工具为项目初期提供了测试平台，可以极大地减少开发时间。

IDE 在业界很流行，目前市场上有开源的、商业的以及处理器供应商提供的各种 IDE。IDE 将多种工具集成在同一环境中，可以简化嵌入式软件开发。在 IDE 中，多种开发工具通常共享相同的用户接口、命名方式、命令集以及其他总体操作等，数据在开发工具之间的搬移也较容易和直观。然而，并不是所有的 IDE 都是类似的，可能相当部分的 IDE 只有基本的代码生成工具，如汇编或编译器。而其他开发工具没有紧密地集成在一起。于是工具间的转换很费时，会影响整个软件开发过程的效率。

另外，IDE 的性能也很重要。某些基准程序可以作为这方面的比较基础，或者通过不同的 IDE 运行大量已有代码也可以进行 IDE 性能比较。而且，不同 IDE 提供的可视性区别也很大，良好的可视性能提高开发和调试代码的效率。

有些处理器供应商的 IDE 以开源的 Eclipse 为基础。Eclipse 可以支持多种编程语言，如 Java、Ada、C、C++、COBOL、Python、Ruby 等。Google 公司根据 Eclipse 开发了 Android 的开发工具，TI 公司在 Eclipse 的基础上开发了自己处理器的 IDE——CCS。于是采用 TI 处理器，基于 Android 平台的嵌入式系统可以得到 Google 和 TI 的双重支持。

多数 EVM 包括一块验证过的印刷电路板（PCB）和相关电子元件，以及少量软件资源，不过不同 EVM 支持的软件资源变化很大。支持广泛软件资源的 EVM 可以使开发人员更快地搭建好一个应用实例。更完整的 EVM 可以支持 IDE 中的 SDK、代码库，以及 Linux、Android 或 Windows CE 这类操作系统。

SDK 通常由芯片厂商免费提供，其基本组件通常为特定的代码生成器，但有的 SDK 包括了一个特定应用的全部软件参考设计。后面这种类型的 SDK 中，构成系统的大部分软件都已经准备好了，直接可以使用，例如：多媒体框架、编解码器、控制 USB 和网络等接口的输入输出代码。这样，软件开发人员可以集中精力实现系统的上层应用软件。然而，这类包含广泛软件的 SDK 的坏处是它会限制开发人员实现与众不同的创新功能。因为这类 SDK 的大部分软件为二进制目标码，开发人员无法访问底层源代码，从而很难进行个性化修改。反之，那些由源代码组成的 SDK 给开发者提供了修改和定制的机会。只要许可协议允许，开发者就可以按自己意愿进行修改。

SDK 除了要易于定制修改，还应提供结构化的、文档清楚的应用编程接口（API）。一个有效的 API 能简化编程，有利于软件构架的重新配置。API 还应为系统软件的每层提供钩子（HOOK），以便开发者能方便地将自己的软件模块加入到系统中，而不用重新编写整个新模块。这样可以显著减少软件开发时间，加快新产品的上市。

处理器供应商对开源软件的支持很重要。开发人员选择开源软件时要考虑是否能方便地移植到特定的处理器，若处理器供应商已经完成了移植更好。例如 TI 公司，已经将大量的开源中间件和工具移植到了自己的处理器上，以便开发人员直接使用。这些开源软件包括：

- OpenMax；

- GStreamer;
- Eclipse IDE;
- OpenGL;
- OpenCV;
- Yocto;
- Linaro。

TI 为嵌入式系统提供了移植好的 Android 软件库，包括操作系统、中间件以及其他工具，适用的处理器包括 TI 的 ARM、DSP + ARM、DSP、OMAP 以及数字媒体处理器等。详见 arowboat. org。

1.1.5 代码结构

代码结构对于嵌入式软件编程是很重要的。在开始设计软件时应考虑代码结构，这会影响将来代码的重用、个性化的设计，以及是否容易调试等。

1. 代码的重用

若在不同的应用中，代码、模块，甚至整个软件系统可重用，则软件开发的投资回报就会增加。为了取得最大的投资回报，制造商在制定产品线策略时，希望尽可能多的软件能在所有产品中重用。因此，代码重用成为了软件开发团队需要首先考虑的问题。

理论上，在一个产品线中，每一次提升系统能力常常意味着提高处理器性能，或者更换更强大的处理器。为了确保软件重用，软件团队必须考虑软件对产品线中不同处理器的兼容性，理想情况是为一种处理器开发的软件可以直接下载到产品线的其他系统中。因此，系统设计者必须仔细考虑各种处理器对软件的兼容性，以及软件对不同处理器的可移植性。

开发工具能帮助实现跨处理器平台的软件重用。例如，C 等高级语言编译器允许一定程度的代码重用。具有良好结构和完整文档的 API 也能保证软件在多种平台和处理器之间的扩展性。然而，有些 API 相对于它们的代码而言过于庞大臃肿，从而对系统性能造成了负面影响。

总之，当软件团队具有广泛的开发工具、软件组件、封装器，以及其他能力，就可以实现高层软件重用；即可以很容易地混合以前开发的代码、加入或裁减掉某些特性和功能，从而快速地实现待上市产品的特定需求。

2. 代码的定制

许多软件开发项目的一个重要目的是为最终产品提供独特的性能或与众不同的特性。这样，一旦产品上市，就能从竞争中脱颖而出，从而取得极大成功。软件开发团队在开发新的特别功能，并将他们与开源软件集成时，必须特别小心。除非将自己的软件系统严格组织，并与开源软件完全隔离，否则根据开源软件许可协议，这些新开发的独特软件也会被认为是开源的。因此，必须将这些代码贡献给开源社区。

一些技术公司已经采取了一些步骤来预防这类情况。例如，TI 公司提供的软件附有一份“软件清单”，详细解释和描述了这些代码的来源，以及相关的、需遵循的任何第三方或开源软件许可证。而且，TI 公司的开源软件审查委员会已经检查过了这些软件清单以及代码本身，以确保这些代码具有严格的组织和架构，从而保护系统厂商的利益。

3. 调试

几乎没有软件项目可以不经过调试就能发现和消除无法预料的错误和故障。事实上，如

果软件设计成便于调试，可以更快地实现全部功能。搭建软件系统的架构与各独立处理节点的组合时，可以考虑尽可能多地为调试提供代码可见性，以便于软件开发者孤立错误、快速修复有问题的代码段。

处理器供应商也会提供基于硬件仿真器的调试工具。例如，TI 提供全系列硬件仿真器，可以在处理器运行时查看相关的状态。而且，这些仿真功能是 TI 的 IDE——CCS 的组成部分，能方便用户在源代码中快速定位并修复故障。

1.1.6　中间件和软件框架

已经有大量的资源可以供嵌入式软件开发者使用。当然，选择特定操作系统和操作系统类型将影响贯穿项目始终的各种软件资源，包括中间件、软件框架，以及其他工具软件。例如，与类似 Windows CE 的商业操作系统相关的软件资源通常受操作系统供应商或授权的第三方厂商的限制；反之，选择类似 Linux 的开源操作系统的开发者必须从开源社区提供的广泛的开源中间件和工具中评估和选择自己需要的软件资源。而且，项目组可能不得不在自己的硬件上移植开源软件资源。不过 TI 等一些硬件厂商已经为用户完成了很多开源软件资源的移植。随着基于 Linux 的免版税 Android 系统的推出，多数开源软件资源提供了基于 Android 的版本。另外，TI 等芯片厂商以及其他技术公司也提供了专有的软件开发资源。

现在，软件开发人员使用开源中间件越来越普遍。有些中间件，如 GStreamer、OpenCV，以及 OpenMax 等已经在某种程度上遍布业界。开源的工具链，如 Yocto 和 Linaro 等也已崭露头角。

中间件是一组预先定义并完全开发的软件功能或任务，开发者可以选择其中的一些组件来部署自己的系统。软件框架为集成各子任务确定了一组规则，以便减少集成的时间和难度。典型的中间件是实现特定类型的处理任务。如 GStreamer 专注于多媒体处理，需要部署多媒体处理栈的开发团队可以从其中选择多种模块，如视频解码模块或音频处理插件。

TI 的微控制器和微处理器已经提供了许多移植了各种中间件的软件开发套件。这样可以使项目开发简化步骤，不用为硬件平台移植中间件，而是直接在最流行的中间件基础上集成自己的任务。

1.1.7　多媒体编程

一般而言，多媒体编程包括处理视频、音频、图像以及图形。一些主要考虑的问题包括支持何种编解码器、需求哪些 I/O 接口、系统实时性的要求、系统并发性的需求、要支持几个通道、比特率和帧率、显示分辨率、多媒体内容的质量、存储容量和带宽等，所有这些方面都会影响多媒体编程。

许多系统处理视频和音频内容并输出到音/视频端口。处理器厂商也许会提供音频和视频驱动，但嵌入式开发人员仍需要熟悉各种音/视频接口，以便出现问题时进行调试。音频接口有模拟和数字形式。对于模拟音频接口，系统中要包括模数转换器（ADC），数模转换器（DAC）或二者的结合（codec）。对于数字音频接口，通常为 SPDIF 和 HDMI 接口。模拟视频接口包括射频（RF）、复合视频、分离视频等。这些模拟视频流通过视频 ADC 构成的解码器转换成数字形式。视频编码器则将数字视频转换成模拟视频流供监视器显示。最常用

的数字视频接口为 HDMI、DVI 和 LCD。一些系统会与相机接口，包括 CMOS 传感器、智能传感器，或 USB 相机接口。这些接口有时会集成到一片 SOC 器件上。若没有集成到 SOC，系统开发者就必须选择不同的部件来实现要求的接口，此时必须确保所选的部件包括了合适的驱动。

许多嵌入式系统必须能处理压缩的音频和视频数据。这时，系统设计者和编程人员需要考虑如何实现压缩和解压。系统可能输入各种压缩的媒体数据，包括自己的文件系统从硬盘、USB 闪存或 SD 卡读取的压缩内容，从有线或无线网络下载的压缩比特流等。通常，系统软件必须能解析特定的数据头，以便获知相应的压缩算法。当前常用的主流媒体格式有：数据报流（TS）、程序流（PS）、AVI、MPEG－4、ASF 等。处理数据头的软件模块的主要功能之一是对音频和视频数据解复用，对比特流解复用后，将压缩数据的缓冲区送到解码器或解压器。表 1.1 列出了一些主流编解码器。

表 1.1 主流多媒体编解码器

媒体类型	编解码器
视频	H.264（AVC）、MPEG－4、MPEG－2、VC1、VP6/7/8、RealVideo
音频	AAC、MP3、WMA、MLP、AC3、Vorbis
图像	JPEG、PNG、GIF、TIFF
语音	G.7xx

1. 存储容量问题

目前许多嵌入式系统采用 DDR 存储器。为了降低系统成本，设计人员要在存储成本和容量之间进行折中。也就是说，设计者希望用最低成本的存储器提供能满足系统多媒体处理要求的足够带宽。

视频处理通常消耗大量的 DDR 容量，而音频处理消耗的相对少一点。将视频流传输到输入输出接口、解码、编码、后处理以及显示视频均会消耗大量存储器。编程人员应该意识到，应避免高清晰度视频的缓冲区拷贝，因为这种操作会超过系统中 DDR 的容量限制，造成低视频帧率或显示异常，从而给用户很差的使用感受。

2. 图形/视频混合

在许多应用中，图形和视频必须混合或组合以呈现给用户显示。一些嵌入式处理器包括了显示子系统，能直接用硬件进行这种混合。若没有显示子系统，需要用二维图形引擎来混合视频和图形。有的应用要实现三维（3D）视频，于是需要三维图形引擎。各种图形中间件也能帮助实现图形与视频的混合。

3. 响应时间

响应时间是指系统给出用户响应的时间。对最终用户而言，系统的响应时间至关重要。例如，如果视频会议的响应时间过长，图像或声音通道就有延迟，会对会议的参与者造成混乱。一种减少响应时间的方法是处理流水时减少缓冲区的大小，而且流水的每一步都要严格检查和优化以满足要求的响应时间。

4. 音视频同步

在音频/视频内容解码时，要求同步音频和视频流。例如，达到口型同步很常见，更复杂的同步要求系统的音频/视频解码与广播源等编码器同步。这种与源编码器同步的要求很常见，可以保证不跳帧或重复，从而减少运行时的比特流缓冲，避免溢出。

5. 质量

不同的应用要求不同程度的媒体质量。要达到高质量的音/视频，通常涉及软件算法或加速器，会消耗处理资源。因此，系统设计者必须考虑为多媒体处理任务分配足够的资源。而且，编程人员已经发现在代码的关键点设置内嵌测试能帮助调试质量问题。

6. 多媒体框架

OpenMax 和 GStreamer 都是免版税的跨平台多媒体中间件，软件开发人员可以快速将媒体处理模块插入到自己的系统中。这些多媒体模块包括音频、视频、静态图片以及其他类型的媒体处理例程。这些例程通常用 C 或 C ++ 等高级语言编写，包括为应用程序提供抽象层的编程接口。

OpenMax 由 Khronos 发起，赞助商有 Evans&Sutherland、ARM、TI、NVIDIA、SGI、Oracle/SUN 等。GStreamer 由 Oregon Graduate 于 1999 年发起，已经被 TI、Nokia 等公司采用。

1.1.8　多处理器或多核 SOC

多处理（MP）通常是指一种系统设计方法，特点是在多处理器或多核上运行多线程软件。当前构建计算或通信系统时，MP 流行的原因在于要克服单处理器结构的局限性。单处理器系统受限于处理器的速度，而多处理系统可以通过部署多处理器或处理引擎提高系统的吞吐量。除了性能的提高，与基于单个复杂、庞大的处理器系统相比，MP 结构还可以降低功耗；而功耗对于采用电池的移动设备而言十分重要。

由于 MP 的明显优势，很多嵌入式系统转向采用此技术。然而，如实时处理等嵌入式应用的典型特性可能无法在多数通用系统中找到。因此，嵌入式系统要特别考虑软件开发环境和工具是否支持嵌入式多处理。

自然地，MP 结构要求重点考虑处理器间的通信和协调以及组成多线程软件的各线程。在硬件方面，必须采用一种互联方法，而这决定了系统通信模式是采用消息传递机制还是共享内存机制。消息传递机制为处理单元配置私有存储空间，组成分布式多处理系统，数据采用消息包的形式传递。共享内存机制为所有的处理器设置一大块共享内存空间，数据通信就是将其放入共享内存，这种机制需要考虑缓存的一致性、存储单元的一致性以及同步原语。

从嵌入式软件开发的角度，MP 结构会涉及在处理器之间进行负载分配，如何协调系统中的处理器，如何将多线程软件分给各处理单元以及其他问题。幸运的是，有开发工具和工具集可以提供帮助。

1. OpenMP

OpenMP 是共享内存并行编程开发环境事实上的标准。TI 为它提供技术支持，而且已经移植到了许多的 TI 处理器上。OpenMP 是一个高层编程结构，以便显著简化开发多线程软件的许多问题。使用 OpenMP，用户为一个程序具体化并发策略时，只需按照编译器的指引，在高层代码标注，说明一段代码会被哪些线程运行，而编译器完成代码到处理单元的详细映射。OpenMP 可以运行于实时操作系统。

2. EZ tools

大量的嵌入式系统采用异构多处理结构，即一个或多个通用处理单元，以及若干数字信号处理器（DSP）。DSP 处理单元通常加速如编解码等处理密集性任务。TI 的 EZ tools 套件