第一部分

入门指导

第 1 章

嵌入式和实时空间

如果你想周游世界并应邀去许多不同的地方演讲，那么写一个 UNIX 操作系统就可以了。

——Linus Torvalds

1.1 什么是嵌入式

在一个聚会上，当一个有吸引力的异性走近你，问你的工作时，你应该轻描淡写，说得越少越好，但最终谈话还是会落在你为嵌入式系统写软件这个事实上。在这位新相识开始扫视周围，找到一个律师或医生谈话前，你最好用最引人注意的解释说明到底什么是嵌入式系统。

我通常会说嵌入式系统就是一个内置计算机的设备，但是该设备的使用者不必知道或者特别在意那个计算机。它是隐蔽的。我通常以汽车里控制发动机的计算机为例来说明。你不觉得开车有什么不同，那是因为发动机是计算机控制的，而且汽车里还有很多计算机，有的控制防震刹车片，有的决定什么时候打开气囊，另外一些在交通拥堵的早晨给坐在车里的你提示信息和提供娱乐。事实上，今天普通的车都比 20 世纪 70 年代的登月飞行器有更强的处理能力。甚至你的手机也比登月飞行器处理能力强大。

你可以继续举出很多例子，嵌入式计算机的应用比个人电脑（PC）要多得多[⊖]。事实上，最近的市场数据显示，个人计算机使用的微处理器芯片只占每年市场份额的 2%。普通的房间即使没有个人电脑，也至少有几十个嵌入式计算机。

从编程的角度来说，嵌入式系统与传统的桌面应用有很多显著区别。比如，多数桌面应用处理的都是相对可预见的一组输入输出（I/O）设备——磁盘、图形显示、键盘、鼠标、声卡和网络接口。操作系统通常都可以很好地支持这些设备，应用程序员不需要特别关注这些。

⊖ 当前，多数人至少有模糊的嵌入式计算机的概念，因为他们至少都使用一个这样的计算机 —— 手机。

但与之相反，嵌入式系统通常比典型的桌面计算机包括种类更多的 I/O 设备。一个典型的系统可能包括各种用户 I/O，如开关、按钮和增加了各种显示方法的触摸屏。它可能有一个或多个通信通道，或者是异步串口、通用串行总线（USB）或网络端口。它可能通过模数转换器（A/D）和数模转换器（D/A）完成数据采集和控制。这些设备几乎都没有应用程序员所熟悉的操作系统的支持。因此，嵌入式系统程序员通常都需要直接处理硬件。

嵌入式设备通常受资源的严格限制。尽管一个典型的 PC 目前已发展到有 4GB 的 RAM 和几百 GB 的磁盘，但嵌入式设备通常只有几个 MB 的 RAM 和非易失存储器。这些都对编程者的创造力有很高要求。

1.2　什么是实时

实时的概念更难解释。实时的基本含义是我们期望计算机对它的环境即时响应。但是什么是“即时”呢？有人认为实时意味着真的很快，这并不完全正确。实时仅仅意味着在系统运行的环境里足够快。如果我们谈论的是控制汽车发动机的计算机，那就是快！这个计算机需要决定每次发动机转一圈时对燃料流量、点火时间的控制。

另一方面，考虑一个或多个计算机控制的化工厂，计算机系统负责控制过程和检测潜在的破坏性故障。因为化学过程的时间常数最少在几秒到几分钟之间，所以可以假设计算机系统可以有足够的时间处理任何故障以避免造成损失。

但是假设当故障发生时，计算机正在打印一份关于上周生产情况的长篇报告或正在处理工资单，那么它对潜在的紧急情况的反应有多快呢？

实时处理的本质不仅在于计算机对它的环境有足够快速的响应，而且在于足够快的可靠响应。发动机的控制计算机必须在发动机转动的每一圈都调整燃料流量和点火时间。如果有所延迟，发动机就不能正常工作。化工厂的控制器必须有足够的时间正确检测和响应异常情况以避免事故，如果不能，这个控制器就是无效的。

我觉得下面这句话说得很恰当：

> 实时系统处理的正确性不仅要求处理的逻辑正确，也要求在规定时间内有结果。如果系统的时间约束不能被满足，那么可以说系统就失效了。
>
> ——Donald Gillies，实时处理常见问题

所以实时编程的艺术就是在随机的异步事件中，设计能可靠地满足时间约束的系统。的确，这说起来容易做起来难，目前有很多研究实时系统原理的文献和开发工作。

1.3　为什么 Linux 适用

Linux 模仿 UNIX 的基本架构，在其基础上发展成为一个通用的操作系统。没有人认为

UNIX 适用于嵌入式或实时操作系统（RTOS）。它太大了，占用太多的资源，而且其调度原则是基于顺序而不是优先级。因此，简而言之，它与嵌入式操作系统的要求完全是背道而驰的。

但是 Linux 有 UNIX 系统的早期版本所欠缺的几样东西。它是免费的，而且你可以得到源代码，并且有一个拥有大量热情的 Linux 开发者和使用者的大社区。你在使用中所面临的问题总有人正遇到或曾经遇到过，所有这些都在网络上，窍门就是找到它们。

1.3.1 开源

Linux 是在自由软件基金会（Free Software Foundation，FSF）倡导的开源软件的宗旨下发展起来的。开源的理念非常简单，即软件应该免费获取以用于使用、修改和复制。这个理念在建立了以太网和万维网的技术文化下发展了 20 多年，近年来开始向商业领域发展。

有许多关于开源软件的误解。也许对它是什么的最好解释就是从它不是什么开始。

- **开源不是共享**。使用共享软件的前提是付给版权持有者一定的费用。共享软件通常以一定的免费形式发布，或者时间受限或者功能受限。要得到完整版，就必须付钱。但与之相反，开源代码可以免费获取，你没有任何为之付钱的义务。
- **开源不是公共**。根据定义，公共代码是没有版权的。开源代码的作者拥有版权，他在开源软件协议的框架下发布软件。版权持有者授权你可以在协议的框架内使用代码，如果你不遵守协议，版权持有者有权要求你停止使用代码。
- **开源不一定免费**。没有任何强制规定要求付款给开源软件代码，但不排除收取打包和发布费用。许多公司都在经营特定的打包 Linux 分发产品。

为什么要为免费得到的东西付钱？想必是因为任何东西只要存在，你就可以从供应商那里得到一些支持。当然，支持的质量很大程度上取决于供应商。

所以免费意味着可以自由使用代码但不意味着零花费。想想“免费演讲”和“免费啤酒”的区别你就明白了。

开源代码是：

- **服从开源协议框架**。在许多案例中都是 GNU 通用公共协议（GPL）[⊖]。
- **经过严格的同行评议**。作为一个开源程序员，你的代码就放在那里，任何人都可以看到。开源社区非常严格，开源代码会经历广泛的测试和同行评议。这是一个适者生存的竞争过程，最后只有最好的代码能够存在。“最好”是一个主观的术语。它可能是最好的技术解决方案但同时也可能可读性很差。
- **高度的颠覆性**。开源运动颠覆了主流模式，主流模式认为软件之类的知识产权应该受到保护，所以你可以通过这个赚钱。与之相反，开源的理念是软件应该免费提供给每个人，这样可以最大化地利于社会的发展。如果你可以通过这个赚钱，那挺好的，但

⊖ GNU 是一个自参照缩写，意思是“GNU 非 UNIX”。GNU 的项目由 Richard Stallman 在 1983 年发起，旨在打造一个“完整的兼容 UNIX 的系统”，完全由免费的软件组成。

这不是初衷。FSF 的创始人 Richard Stallman 一直大力倡导软件不应该属于某个人（见附录 C）。

因此当微软公司视开源为他们商业模式的一个严重威胁时，这就毫不奇怪了。微软的发言人一直将开源归为“非美式的”。而另一方面，许多开源软件的领头供应商一直让他们的程序员和工程师为开源社区添砖加瓦。这不仅仅是公益，更是一个很好的商业模式！

1.3.2　移植和定制

Linux 最初是为 Intel 的 x86 系列处理器开发的，而且多数正在进行的内核开发工作也还是针对 x86 系列的。尽管如此，Linux 内核的设计将有所不同，以前的代码是依赖于处理器的，且对于每个不同架构的处理器都需要改动，但开源代码则只要进行简单的重新编译就可以导入一个新处理器。因此，Linux 可以导入多数 32 位甚至 64 位的处理器，处理器架构包括：

- Motorola 68k 和它的很多变体
- Alpha
- PowerPC
- 高级 RISC 机器（ARM）
- Sparc
- MIPS

这只是很多流行的处理器中的一些。所以，无论嵌入式项目中要用什么样的 32 位架构，总有 Linux 可用，也总有很多的开发者在支持它。

一个典型的桌面 Linux 的安装运行需要 10 ～ 20GB 的磁盘空间，并需要 1GB 的 RAM 来较好地运行。但是嵌入式的目标板通常只有 64MB 或更少的 RAM 和也许是 128MB 的 flash ROM 存储器。幸运的是，Linux 是高度模块化的。10GB 的内容大都包括文档、桌面实用程序、游戏等选项，这些在嵌入式目标板中不是必要的。如果资源受限的话，生成一个功能完整的只占用不超过 2MB flash 存储器的 Linux 系统是不难的。

内核本身是高度可配置的，包括一些合理的用户工具，允许在应用中去掉某些不需要的内核功能。

1.4　哪里用嵌入式 Linux

任何地方都在用。2005 年 7 月，LinuxDevices.com 网站列出了超过 300 个运行 Linux 的商业产品。它们包括手机、个人数字助手（PDA）及通过路由器和网关使用的其他手持设备；精简客户端、多媒体设备、电视机顶盒、机器人和加固的 VMEbus [⊖]；适用于军事和控制应用的机箱。这些只是 LinuxDevices 网站的编辑们正好知道的产品。

⊖　即 VERSA Module Eurocard bus，是工业计算机打包和互联标准。

首个或许也是最知名的应用嵌入式 Linux 的家庭娱乐设备是 TiVo 个人录像机，当它在 2000 年首次出现时，就引发了电视业的革命。TiVo 基于 PowerPC 处理器，运行“ home grown ”嵌入式 Linux 移植版本，使用图形绘制芯片生成视频。

拥有一个运行 Linux 的设备的一半乐趣在于让它做比设计者原定目标更多的或不同的事情。许多网站和书都为 TiVo 扩展了更多功能。增加存储容量也许是最明显的拓展，其他流行的扩展包括显示天气、运动比分、股票价格和设置网络服务器。

嵌入式 Linux 的应用不局限于用户产品。它在售卖终端、视频监控系统、机器人甚至外太空都有应用。NASA 的 Gaddard 空间飞行中心开发了一个 Linux 版本叫 FlightLinux 以解决宇宙飞船上机载计算机的特殊问题。在国际空间站，基于 Linux 的设备用于控制称作 Automated Transfer Vehicle（ATV）的无人宇宙飞船的交会和对接。

一直以来，电信服务运营商都依靠专业的自有平台满足电信网络的有效性、可靠性、通话性能和服务响应时间等要求。现在运营商们开始接受“ 开放架构 ”，采用商业非定制硬件和软件以努力在保证服务性能的同时降低成本。

Linux 在开放的标准化网络设施的发展中扮演了很重要的角色。2002 年，开源开发实验室（OSDL）设立了一个工作组来定义 Carrier Grade Linux(CGL)，以努力达到电信行业的高实用性、服务能力和可测性要求。CGL 的目的是达到“ 5 个 9 ”的可靠性水平，意味着系统在 99.999% 的时间里是正常运转的，也就是说一年内只能有不到 5 分钟的故障时间。

1.5 开源协议

多数软件终端用户协议都明确限制了你只可以使用协议范围内的功能。典型的限制条件是不允许复制或重新发布。你通常会被警告不要试图对软件进行“ 逆向工程 ”。

相反，开源协议是只要你愿意，就允许使用、修改和复制授权的软件。和权利相伴的是义务。如果你修改并发布了一个开源协议内的软件，你就必须将修改后的源代码也纳入该框架。你的修改就成为“ 派生的工作 ”，也在该协议的范围内。这就允许其他使用者更好地理解软件，并按他们的意愿做出更多的修改。

开源协议也叫“ 公共版权 ”协议，使用版权的目的是表达这样一个观点 ：使用版权法来作为一种便利而不是限制获取软件知识产权的途径。尽管版权通常用于一个作者限制其他人复制、修改和发布一个产品，公共版权则明确允许在同样的协议框架下修改和再发布该产品。因此“ 公共版权 ”允许你从他人的工作中获益，但是你的任何改动均需要在同样的框架下发布。

毫无疑问，最知名的、应用最广泛的开源协议是 GNU 通用公共协议（GPL），它最初由 FSF 在 1989 年发布。Linux 内核是在 GPL 框架内的，但是有一个问题使 GPL 不能在一些商业场合使用。使用或参考其他在 GPL 架构下发布的软件而生成的软件，即使只是链接到一个

库，也被认为是派生产品，也必须受限于 GPL 架构而以源代码的形式提供。

为了解决这一问题，推动开源库的发展，FSF 推出了 GPL 库（LGPL）。区别是链接到 LGPL 框架库的程序不被认为是派生工作，尽管你仍然必须使代码能被库所用，但不要求你发布代码。

后来，LGPL 就因其提供给用户的自由度更小而被称为“更小的 GPL”。所以当 LGPL 让使用开源软件开发有所有权的产品成为可能时，FSF 就鼓励开发者将他们的库置于 GPL 的框架内，以达到最大限度的开源。

与之完全相反的是伯克利软件发布授权（BSD），它比 GPL 早了约 12 年。它建议但不要求对源代码的改动必须反馈给开发者社区，并且允许使用其他协议生成产品，包括个人协议。

也有不少其他协议介于这两者之间。比如在 1998 年开发的 Mozilla 公共协议（MPL），当时 Netscape 公司将他们的浏览器开源了，它包含比 BSD 协议更多的关于派生产品的要求，但是比 GPL 或 LGPL 少。开源促进会（Open Source Initiative）是一个非盈利性的组织，负责认证满足开源定义的协议，2011 年 12 月，在它的网站上列出了 78 个通过认证的协议。

多数在 GPL 的框架下发布的软件，包括 Linux 内核，都适用于协议的第 2 版，它在 1991 年发布，正好和 Linux 的发布在同一年。FSF 在 2007 年 6 月发布了 GPL 的第 3 版。发布第 3 版的目的之一是解决“TiVo 化（tivoization）”的问题，这是由 FSF 的创始人 Richard Stallman 发明的一个术语。结果表明 TiVo 只能运行有授权数字签名的代码。所以，即使 TiVo 使其源代码纳入 GPL 框架，修改的代码将仍然不能运行。

Stallman 认为这违背了 GPL 的精神。其他的开发者包括 Linus Torvalds 则将数字签名视为一个有用的安全工具而不想将其完全禁止。争论还在继续。但是在任何情况下，内核本身不可能很快过渡到第 3 版。

法律问题

开源的法律意义产生了相当多的 FUD[⊖]，尤其是 SCO 的关于 Linux 内核受 UNIX 代码破坏的言论。SCO 组织，正式名称是 Santa Cruz 公司，在 1996 年从 Novell 获得 UNIX 系统 V 源代码的版权，不过现在还有一些关于 SCO 到底从 Novell 购买到什么的争论。无论如何，SCO 声称 IBM 公司引入了部分 SCO 版权的 UNIX 代码到 Linux 内核中，因此要求 Linux 使用者支付协议费用。

SCO 的案子最终败诉了，该公司曾于 2007 年根据《破产法》第 11 章申请破产，但是突然间通过开源协议问题之争又赢得了很多钱。结果是嵌入式开发者都需要意识到开源和专有软件的协议问题。嵌入式软件，包括衍生的或从其他来源获得的元素，通常与操作系统密切相关。没人期望嵌入式工程师成为知识产权代理人，但无论如何有必要明白你所使用的和生成的软件的授权框架以保证一切运行顺利。

⊖　害怕、不确定和怀疑。

问题要从两方面看待。现在仍然有人努力去识别对 GPL 的侵权，目的不是挣钱，而是通过给侵权者施压让他们停止行动来维护 GPL 的完整性。特别地，GPL 反侵权项目曝光了十几个嵌入式 Linux 供应商，他们一直快速开发但又不遵守 GPL 规则。据 Harald Welte，GPL 反侵权项目的发起者说，常见的非法设备都是网络设备，像路由器、机顶盒和车载导航系统等。

开源协议专家 Bruce Perens 说，嵌入式开发者似乎都有这样一个想法：“这是嵌入式的，没有人能改变源代码，所以无需应用 GPL”。他们的确是这样做的。

现在我们有一些关于嵌入式实时空间及 Linux 怎样应用其中的想法，第 2 章将会描述怎样在一个工作站上安装 Linux。

1.6 资源

embedded.com——《Embedded Systems Design》杂志的网站。这个网站并不是专门针对 Linux 的，但是作为一个更通用的信息工具，对于解决嵌入式系统问题是很有用的。

fsf.org——自由软件基金会。

gpl-violations.org——通用公共授权侵权项目，宗旨是针对 GPL“提高对过去和现在的侵权的认识”，据网站所述，它“仍然只是一个人的努力”。

keegan.org/jeff/tivo/hackingtivo.html——许多个研究 TiVo 破解的网站之一。

kernel.org——Linux 内核档案。你可以在这里下载最新的内核版本和任何之前的版本。

网上的 Linux 资源是非常多的，这里列出了一些嵌入式开发者非常感兴趣的网站。

Linuxdevices.com——全面研究关于嵌入式 Linux 问题的新闻和门户网站。

opensource.org——开源促进会（OSI），一个非盈利的组织，“为了社区的利益，致力于管理和推动开源定义的发展”。OSI 认证满足他们对开源定义的软件协议。

osdl.org——开源发展实验室（OSDL），一个非盈利的协会，致力于加速 Linux 在公司和嵌入式领域的成长和应用。2005 年 9 月，OSDL 接管了嵌入式 Linux 协会的工作，该协会曾开发了针对嵌入式 Linux 的平台规范。

sourceforge.net——“世界上最大的开源开发网站”，对开源开发者提供包括项目托管和管理、版本控制、差错和问题跟踪、备份存档、通信协作等免费服务。

slashdot.org——口号是“网虫的重要新闻（News for nerds，stuff that matters）”，一个非常受欢迎的关于开源软件尤其是 Linux 的网上新闻论坛。

ucLinux.org——Linux/ 微控制器项目是将 Linux 连接到没有存储管理单元的系统的端口。