第一章　走进 Linux

对经常使用计算机的人来说 ，时常会感到操作系统是一个神奇、神秘而又几乎无所不能的“ 上帝 ”。一打开计算机，我们首先看到的是操作系统，所有软件的运行都离不开它，它给我们带来一个个的惊喜，但有时也带来烦恼和不安。

实际上，很多人都有这样强烈的愿望，即“ 上帝 ”到底是怎样操纵这一切的？UNIX 操作系统曾经敞开 UNIX 操作系统的胸怀，让我们窥视到它的内在机制，但它毕竟属于“ 贵族 ”阶层，我们大多数人并不能使用上它。

Windows 以平民的身份来到我们中间，我们欢呼它的友好和平易近人，正因为 Windows，才使得计算机走进我们寻常百姓家，使得计算机普及成为现实。但 Windows 有时也“ 伤风感冒 ”，我们想找到原因，以便对症下药。可是，Windows 的窗户并没有打开，我们无法透过窗户看看 Windows 的内部世界到底是什么样的，这让我们困惑，尤其让喜欢追根寻源的人们感到失望。

Linux 带着一股清新的风翩翩而来，它并不成熟，也不完美，甚至自身有很多缺点，可 Internet 的龙卷风把它吹遍世界，世界各地的计算机爱好者狂热地喜欢上 Linux。Linux 不再是一个孤单的个体，而成为软件发展史上的“ 自由女神 ”，很多的计算机高手和计算机爱好者为之倾其了极大的热情。它在迅速地成长，短短几年功夫，从一个摇摇晃晃的婴儿成长为脚步稳健的少年。这一切都源于什么？那就是 Linux 的创始人 Linus Torvalds 把 Linux 适时地放入到了 GNU 公共许可证下。

1.1　GNU 与 Linux 的成长

GNU 是自由软件之父 Richard Stallman 在 1984 年组织开发的一个完全基于自由软件的软件体系，与此相应的有一份通用公共许可证（General Public License，简称 GPL）。Linux 以及与它有关的大量软件是在 GPL 的推动下开发和发布的。

自由软件之父 Stallman 像一个神态庄严的传教士一样喋喋不休地到处传播自由软件的福音，阐述他创立 GNU 的梦想：“ 自由的思想，而不是免费的午餐 ”。这位自由软件的“ 顶级神甫 ”为自己的梦想付出了大半生的努力，他不但自己创作了许多自由软件如 GCC 和 GDB，在他的倡导下，目前人们熟悉的一些软件如 BIND、Perl、Apache、TCP/IP 等都成了自由软件的经典之作。

如果说 Stallman 创立并推动了自由软件的发展，那么，Linus 毫不犹豫奉献给 GNU 的 Linux，则把自由软件的发展带入到一个全新的境界。

实际上，Linus 是一个理想主义者，但他又脚踏实地。当 Linux 的第一个“ 产品 ”版本

Linux 1.0 问世的时候，是按完全自由扩散版权进行扩散的。他要求 Linux 内核的所有源代码必须公开，而且任何人均不得从 Linux 交易中获利。他这种纯粹的自由软件的理想实际上妨碍了 Linux 的扩散和发展,因为这限制了 Linux 以磁盘拷贝或者 CD-ROM 等媒体形式发行的可能，也限制了一些商业公司参与 Linux 的进一步开发并提供技术支持的良好愿望。于是 Linus 决定转向 GPL 版权，这一版权除了规定自由软件的各项许可权之外，还允许用户出售自己的程序拷贝。

这一版权上的转变对 Linux 的进一步发展可谓至关重要。从此以后，便有很多家技术力量雄厚又善于市场运作的商业软件公司，加入到了原先完全由业余爱好者和网络黑客所参与的这场自由软件运动，开发出了多种 Linux 的发行版本，磨光了自由软件许多不平的棱角，增加了更易于用户使用的图形用户界面和众多的软件开发工具，这极大地拓展了 Linux 的全球用户基础。

Linux 内核的功能以及它和 GPL 的结合，使许多软件开发人员相信这是有前途的项目，开始参加内核的开发工作。并将 GNU 项目的 C 库、gcc、Emacs、bash 等很快移植到 Linux 内核上来。可以说，Linux 项目一开始就和 GNU 项目紧密结合在一起，系统的许多重要组成部分直接来自GNU项目。Linux操作系统的另一些重要组成部分则来自加利福尼亚大学Berkeley分校的 BSD UNIX 和麻省理工学院的 X Window 系统项目。这些都是经过长期考验的成果。

正是 Linux 内核与 GNU 项目、BSD UNIX 以及 MIT 的 X11 的结合，才使整个 Linux 操作系统得以很快形成，而且建立在稳固的基础上。

当 Linux 走向成熟时，一些人开始建立软件包来简化新用户安装和使用 Linux。这些软件包称为 Linux 发布或 Linux 发行版本。发行 Linux 不是某个个人或组织的事。任何人都可以将 Linux 内核和操作系统其他组成部分组合在一起进行发布。在早期众多的 Linux 发行版本中，最有影响的是 Slackware 发布。当时它是最容易安装的 Linux 发行版本，在推广 Linux 的应用中，起了很大的作用。Linux 文档项目（LDP）是围绕 Slackware 发布写成的。目前，RedHat 发行版本的安装更容易，应用软件更多，已成为最流行的 Linux 发行版本；而 Caldera 则致力于 Linux 的商业应用，它的发展速度也很快。这两个发行版本也有相应的成套资料。在中文的 Linux 发行版本方面，国内已经有众多的 Linux 厂商，如红旗 Linux，BluePoint Linux，中软 Linux 等。每种发行版本有各自的优点和弱点，但它们使用的内核和开发工具则是一致的。

1.2 Linux 的开发模式和运作机制

自由软件的出现，改变了传统的以公司为主体的封闭的软件开发模式。采用了开放和协作的开发模式，无偿提供源代码，允许任何人取得、修改和重新发布自由软件的源代码。这种开发模式激发了世界各地的软件开发人员的积极性和创造热情。大量软件开发人员投入到自由软件的开发中。软件开发人员的集体智慧得到充分发挥，大大减少了不必要的重复劳动，并使自由软件的脆弱点能够及时发现和克服。任何一家公司都不可能投入如此强大的人力去开发和检验商品化软件。这种开发模式使自由软件具有强大的生命力。

商业 UNIX 开发过程中，整个系统的开发有严格的质量保证措施、完整的文挡、完善的

源代码、全面的测试报告及相应的解决方案。开发者不能随意增加程序的特性和修改代码的关键部分，如果要修改代码，他们得将其写入错误报告中才能使其有效，并随后接收源代码控制系统的检查，如果发现修改不合适，修改也可能作废。每个开发者设计系统代码的一个或几个部分，开发者只有在程序检查过程中才能更改相应的代码。质量保证部门在内部对新的操作系统进行严格的回归测试，并报告发现的问题，开发者则有责任解决所报告的问题。质量保证部门采用复杂的统计分析系统以确保在下次发行时有百分之几的程序错误已修改。

总之，商业 UNIX 开发过程使得其代码非常复杂，公司为了保证下次操作系统的修订质量，得收集和统计分析操作系统的性能。开发商业 UNIX 是一个很大的工程，常常大到有数以百计的编程者、测试员、文档员以及系统管理员参与。

对于 Linux，你可将整个组织开发的概念、源代码控制系统、结构化的错误报告、统计分析等通通扔到一边去。

Linux 最初是由一群来自世界各地的自愿者通过 Internet 共同进行开发的。通过互联网和其他途径，任何人都有机会辅助开发和调试 Linux 的内核、链接新的软件、编写文档或帮助新用户。实际上，并没有单独的组织负责开发此系统，Linux 团体大部分通过邮递清单和 USENET 的消息组进行通信。许多协定已跳过开发过程，如果你想将自己的代码包括进“ 正式 ”内核，只需给 Linus Torvalds 发一个邮件，他就会进行测试并将其包括进内核（只要代码不使内核崩溃并且不与整个系统设计相悖，Linus 都很乐意将其包括进去）。

Linux 系统本身采用彻底开放、注重特性的方法进行设计。一般规律是大约隔几个月就发行一个 Linux 内核的新版本。当然发行周期还依赖于其他一些因素，如排除的程序故障数、用户测试预发行版的返回数以及 Linux 的工作量等。

可以说在两次发行期间，并不是每个故障都已排除，每个问题都已得到了解决。只要系统不出现很挑剔或明显的故障，就认为比较稳定，可以推出新版本。Linux 开发的动力不在于追求完美、无故障，而是要开发 UNIX 的免费实现。

如果你想把新的特性或应用软件增加到系统上，就得经过一个“ 初始 ”阶段。所谓“ 初始 ”阶段，就是一个由一些想对新代码挑出问题的用户不断进行测试的阶段。由于 Linux 团体大多在 Internet 上，“ 初始 ”软件通常安装在一个或多个 LinuxFTP 上，并且在 LinuxUSENET 消息组上张贴一张如何获取和测试其代码的消息，从而使得下载和测试“ 初始 ”软件的用户可以将结果、故障或问题等邮件告之作者。

初始代码中的问题解决后，代码就进入“ 第二 ”阶段：工作稳定但还不完全（即能够工作，但可能还不具备所有特性）。当然，它也可能进入“ 最后 ”阶段，即软件已完备并且可以使用。对于内核代码，一旦它完备，开发者就可让 Linus 将其包括进标准内核内，或者作为内核的可增加选项。

注意，这些仅是达成协定，并未形成规则。很多人对他们的软件不必发行“ 初始 ”或测试版充满信心，因此发行哪个版本是根据开发者的决定而定的。

你可能对一群自愿者居然能编写、调试出完整的 UNIX 系统惊讶不已。整个 Linux 内核通过拼凑而成，没有采用专利的源代码，大量工作都由自愿者完成，他们将 GNU 下的免费软件移植到 Linux 系统下，同时开发出库、文件系统以及通用的设备硬件驱动程序等。

实际上，Linus 率领的分布在世界各地的 Linux 内核开发队伍仍然在高速向前推进。当前推出的稳定的 Linux 内核的 2.4.x 版本充分显示了 Linux 开发队伍的非凡的创造能力以及

协作开发模式的价值。

1.3 走进 Linux 内核

如果说 CPU 是计算机硬件的心脏的话，那么，操作系统的内核则是整个计算机系统的心脏，或者说，是最高管理机构。Linux 的内核包含些什么？简单地说，它包含五大部分内容：进程调度、内存管理、进程间通信、虚拟文件系统及网络接口这五部分，我们也称为五个子系统。在走进 Linux 内核前，读者可能想知道，它到底有什么特点呢？

1.3.1 Linux 内核的特征

Linux 是个人计算机和工作站上的类 UNIX 操作系统。但是，它绝不是简化的 UNIX。相反，Linux 是强有力和具有创新意义的类 UNIX 操作系统。它不仅继承了 UNIX 的特征，而且在许多方面超过了 UNIX。作为类 UNIX 操作系统，Linux 内核具有下列基本特征。

(1) Linux 内核的组织形式为整体式结构。也就是说整个 Linux 内核由很多过程组成，每个过程可以独立编译，然后用连接程序将其连接在一起成为一个单独的目标程序。从信息隐藏的观点看，它没有任何程度的隐藏 —— 每个过程都对其他过程可见。这种结构的最大特点是内部结构简单，子系统间易于访问，因此内核的工作效率较高。另外，基于过程的结构也有助于不同的人参与不同过程的开发，从这个角度来说，Linux 内核又是开放式的结构，它允许任何人对其进行修正、改进和完善。

(2) Linux 的进程调度方式简单而有效。可以说 Linux 在追求效率方面孜孜不倦，体现在调度方式上也是别具一格。对于用户进程，Linux 采用简单的动态优先级调度方式；对于内核中的例程（如设备驱动程序、中断服务程序等）则采用了一种独特的机制 —— 软中断机制，这种机制保证了内核例程的高效运行。

(3) Linux 支持内核线程（或称守护进程）。内核线程是在后台运行而又无终端或登录 shell 和它结合在一起的进程。有许多标准的内核线程，其中有一些周期地运行来完成特定的任务（如 swapd），而其余一些则连续地运行，等待处理某些特定的事件（如 inetd 和 lpd）。内核线程可以说是用户进程，但和一般的用户进程又有不同，它像内核一样不被换出，因此运行效率较高。

(4) Linux 支持多种平台的虚拟内存管理。内存管理是和硬件平台密切相关的部分，为了支持不同的硬件平台而又保证虚拟存储管理技术的通用性，Linux 的虚拟内存管理为不同的硬件平台提供了统一的接口，因此把 Linux 内核移植到一个新的硬件平台并不是一件很困难的事。

(5) Linux 内核另一个独具特色的部分是虚拟文件系统（VFS Virtul File Systen）。虚拟文件系统不仅为多种逻辑文件系统（如 ext2,fat 等）提供了统一的接口，而且为各种硬件设备（作为一种特殊文件）也提供了统一接口。

(6) Linux 的模块机制使得内核保持独立而又易于扩充。模块机制可以使内核很容易地增加一个新的模块（如一个新的设备驱动程序），而无需重新编译内核；同时，模块机制还

可以把一个模块按需添加到内核或从内核中卸下，这使得我们可以按需要定制自己的内核。

（7）增加系统调用以满足特殊的需求。一般来说，系统调用是操作系统的设计者提供给用户使用内核功能的接口，但 Linux 开放的源代码也允许你设计自己的系统调用，然后把它加入到内核。

（8）网络部分面向对象的设计思想使得 Linux 内核支持多种协议、多种网卡驱动程序变得容易。

1.3.2　Linux 内核版本的变化

自从 1991 年 9 月 17 日，Linus Torvals 正式宣布 Linux 的第一个正式版本——0.02 版本，到现在，Linux 的内核版本发生了一系列的变化，新旧版本之间的时间间隔为几个月甚至几个星期，关于这一变化的非常详细的资料请看站点 http://ps.cus.umist.ac.uk/~rhw/kernel. versions.html 的内容。

我们把内核版本之间内容较大的变化分为三个阶段，第一阶段为 0.02～0.99.15j，第二阶段为 1.0～1.2.x，第三阶段为 1.2.x～2.x.x。一般来说，一个软件要到理论上已经完备或者已经没有毛病时才给予 1.0 版本的版本号，而 Linux 2.0 以后的版本比起 1.2.x 版本有了较大幅度的变化，请看站点 http://www.linuxhq.com/的内容。

从 Linux 诞生开始，Linux 内核就从来没有停止过升级，从 Linus 第一次发布的 0.02 版本到 1999 年具有里程碑意义的 2.2 版本，一直到现在的 2.4 版本，都凝聚了 Linux 内核开发人员大量辛苦的劳动。目前 Linux 在各种工作平台上，包括企业服务器和个人电脑上的广泛应用，使得 Linux 成为了 Windows 的强劲对手。

本书所分析的 Linux 内核版本是 2.4 版的 2.4.16 版。那么 Linux 2.4 版具有什么样的特点呢，我们可以用四个字来概括，那就是“广、新、快、小”。

1．广泛的支持

- 处理器芯片的广泛支持：Linux 2.4 提供了大量的处理器芯片的支持。原先的 Linux 就可以支持多种处理器体系结构，如 Intel x86、Motorola/IBM PowerPC、Compaq（DEC）Alpha 等，现在还增加了对 IA 64、S/390、SuperH 这 3 种体系结构的处理器的支持。对 Intel 的 x86 系列来说，AMD 和 Cyrix 公司的系列处理器产品也是使用 x86 指令的，同样也能获得很好的支持。
- 对 ISA 即插即用设备的支持 过去在 Linux 核心开发小组里面存在有两种不同的观点，一种是支持对 ISA 即插即用，另外一种持反对意见，认为对即插即用的支持简直是多余的。因此过去在 Linux 里对即插即用设置的通用做法只能是利用用户级的工具（如 isapnp tools），手动配置即插即用设备。现在的内核则有所不同了，在内核级实现了对即插即用的管理。我们可以看到系统会在启动的时候自动完成对即插即用设备的检测和自动配置，比如说，我们可以从一个即插即用的 IDE 控制器上启动系统。
- 广泛的文件系统支持：很少有一个操作系统能支持这么多种文件系统。Linux 使用的是 VFS（虚拟文件系统）的技术，提供了对多种文件系统的支持。从 Linux 1.x 到 Linux 2.2，Linux 已经可以支持多种文件系统。如 Windows 9x 的 VFAT、DOS 的 FAT、Mac OS 的 HFS、OS/2

的 HPFS、Windows NT 的 NTFS（NTFS 的支持还处于测试阶段）等；当然还包括 Linux 自己使用的高性能的 Ext2 文件系统。新版本的 Linux 新增支持现在的 DVD 使用的 UDF 文件系统和 SGI 的 IRIX 系统上的 XFS 文件系统。

在 Windows 里面使用 SMB 协议来实现“网上邻居”的共享访问，Linux 2.4 的内核里会让您自己选择是否从 Windows 98/NT 下载驱动器，还可以自动检测远端的系统类型，使得 Linux 在 Windows 环境的局域网里工作得更好。

对 NFS（网络文件系统）来说，Linux 2.4 版本支持最近发布的 NFS v3 版本的网络文件系统。

- 对软猫的支持：软猫实际上也被称为 WinModem，就是因为现有的这种软猫的驱动都是由为 Windows 开发的软件来完成的。这种 Modem 和一般 Modem 的处理方法不同，它的 DSP 处理并不是在硬件层次上完成的，而是使用软件通过 CPU 来实现的，因此无法在现有的 Linux 中配置这种 Modem 上网。现在的 Linux 内核里已经开始了这方面的支持。

2．新思路

- 新型的设备管理方法：Linux 2.4 引入了 I2O（Intelligent Input/Output）的设备驱动管理方法。它的做法是，将驱动程序分成了两个部分：一个是在操作系统模块的部分，另外一个是在硬件模块的部分。操作系统模块的部分是独立的，硬件模块的部分是依赖于硬件结构的。这种新型的管理方法使得 Linux 2.4 可以更好地支持大部分的 ISA 和 PCI 设备。
- 对 USB 总线的支持：近年来，USB（通用串口总线）的技术是计算机界振奋人心的事情之一，现在已经出现了大量的使用这种接口的设备，如键盘、鼠标、音箱、Modem 等。使用 USB 接口使得计算机外设的安装和使用变得更为简单，自然成为了一种潮流。现在的 Linux 也可以很好地支持这种总线接口的设备。
- 新型的二进制执行代码类型（Binary Types）：Linux 是第一个在内核级提供内建 Java 解释器的支持，从而进行 Java 代码的执行的操作系统之一。这在 Linux 2.2 版本里已经实现了。Linux 2.4 版本又做了改进，将这种支持的方法改为对“Misc”二进制类型的支持。通过使用这种类型的二进制代码类型，用户甚至可以利用 DOSEMU（MS DOS 模拟器）或者 WINE（MS Windows 模拟器）来运行在 DOS/Windows 下的.exe 或.com 的程序。同样用户也可以自己配置出 Java 字节码运行类型。
- 内核级的 Web 服务器：这种 Web 服务器和所谓的 Apache 用户层上的 Web 服务器并不冲突。对 HTTP 请求首先由内核级的 Web 服务器进行处理，如果不能处理就将请求提交给 Apache 用户级 Web 服务器来处理。像这样的构思和实现在网络操作系统中实属一绝。

3．高性能

- 对虚拟文件系统（VFS）的修改：Linux 2.4 版本的文件系统修改了 VFS 中的错误，尤其是在文件的缓存管理上。过去的文件系统的高速缓存管理是建立在复杂的双缓冲池（dual-buffer pool）上的，这种方法导致连开发人员都不知道什么时候将双缓冲池进行同步。这种处理方法并没有给文件处理带来好处，反而增加了内存的使用。因为要处理双缓冲系统的同步，使得系统的处理速度降低。现在开发人员修改这段代码，使用了简单有效的单缓冲系统，提高了文件系统的处理效率

- 对高端服务器的支持：Linux 2.4 版本的内核可以支持在 SMP（对称多处理器系统）下的多个 IO-APIC（输入输出的高级可编程中断控制器），提高了对高端服务器的支持效率。

Linux 2.4 版本可以支持多达 10 个 IDE 控制器。过去的 Linux 版本只能支持最多 4 个 ID 控制器。一些强大的企业级 Web 服务器正需要这样的硬件支持。

Linux 2.4 版本可以支持 Intel P6 以上芯片的 MTRR（内存类型范围寄存器），对非 Intel 的如 Cyrix 6x86、6x86MX、MII 的 ARR（地址范围寄存器）也能有很好的支持，这使一些高带宽的设备的运行性能得到了提高。

现在的内核可以支持多达 42 亿个用户。在 Intel 架构上可以支持到多达 4GB 的内存。并且现在的内核还可以支持多达 16 块以太网卡，同时支持最大容量为 2GB 的文件。

这些性能都使得 Linux 对高端设备的支持能力得到了提高。

- 对高速网的支持：Linux 2.4 版本支持 ATM 网络适配器等高速网络设备，为进一步的网络发展做好了准备。对低端用户来说，Linux 提供的 PPP 层和 ISDN 层的结合，提供了在并口线上的 PPP 和在以太网上的 PPP 支持。

4．小内核

- 内核本来就很小：Linux 的整个内核源代码大概需要占用 20 多 MB 的硬盘空间，但是编译出来的二进制代码只占用 600KB 左右的空间，完全可以放在一张软盘上，随时可以使用这张软盘启动系统。
- 对内存的需求很小：大家比较关心的一个问题是 Linux 现在需要多少内存才能正常工作。我们知道，大部分的操作系统在升级的同时，对硬件的需求也在不断提高，尤其是对内存的需求方面，很大层次上影响了系统的性能。不过 Linux 和其他操作系统不同，Linux 可以进行个性化的定制，用户完全可以根据自己的系统配置来生成自己需要的操作系统内核，也可以根据需要启动或关闭一些系统服务，这样可以减少系统对资源的占用，提高系统的运行效率。

Linux 内核发展到现在已经相当庞大，要想在一段时间内搞清所有的内容几乎是不可能的，因此，本书对内核的分析也集中在几个主要部分的主要内容上，其运行的平台也只选择了 i386 的单 CPU，在一些特殊情况下，我们也会讨论 SMP（对称多处理机）的情况。

走进 Linux 不是一件容易的事，但走出来同样不容易。阅读 Linux 源代码如同阅读一篇优美的作品，会深深地吸引着你，既可以满足你好奇的愿望，也可以检验你挑战困难的勇气。

1.4　分析 Linux 内核的意义

Linux 开放的源代码为我国软件产业的发展和腾飞提供了前所未有的机遇，这体现在以下几个方面。

1.4.1 开发适合自己的操作系统

因为操作系统是所有软件赖以生存的基础，因此，我们强烈地需要拥有自己的操作系统，这不仅对我们国家的民族软件发展有极大的好处，而且对国家的安全和国防事业都至关重要。但是如果象日本那样搞自己的一套体系结构（PC98），不与国际标准兼容，结果会严重阻碍软件业的发展，那也是死路一条。但是国产操作系统没有任何市场，而 Windows 又几乎处于垄断地位，面对这种局面，出路何在？Linux 的出现正符合我们所有的要求，因为源代码公开，我们可以立即加入开发，不仅开发速度大大快于任何商业操作系统，并且可以保证操作系统中不存在任何黑洞和隐蔽的问题，永远不会受制于人。因为 Linux 是国际化的，我们也不必考虑兼容性问题，永远不会同国际脱轨。因此 Linux 对于我们来说，是实现民族软件腾飞的一个难得的机遇。

实际上，操作系统的发展必将出现基于某一标准的百花齐放的局面，定制适合自己的操作系统也将不仅仅是梦想。但是，开发一个操作系统不是一件容易的事，甚至分析一个现有的操作系统也并不简单，而 Linux 作为分析实例是比较合适的。因为 Linux 的开放、众多人的参与以及 Linux 社区的互助都为 Linux 的学习和普及提供了良好的外部环境。

1．开发嵌入式操作系统

Linux 为嵌入操作系统提供了一个极有吸引力的选择，它和 UNIX 相似，是以内核为基础的、完全内存保护、多任务多进程的操作系统。支持广泛的计算机硬件，包括 X86、Alpha、Sparc、MIPS、PPC、ARM、NEC、MOTOROLA 等现有的大部分芯片。程序源码全部公开，任何人可以修改并在 GNU 通用公共许可证（GNU General Public License）下发行，这样，开发人员可以对操作系统进行定制，再也不必担心像 Windows 操作系统中“后门”的威胁。同时由于有 GPL 的控制，大家开发的东西大都相互兼容，不会走向分裂之路。Linux 用户遇到问题时可以通过 Internet 向网上成千上万的 Linux 开发者请教，这使最困难的问题也有办法解决。

正是嵌入式操作系统的特殊要求为 Linux 在嵌入式系统中的发展提供了广阔的空间，使得 Linux 成为嵌入式操作系统中的新贵。在应用上，嵌入式 Linux 可应用于信息家电（机顶盒、数字电视）、多媒体手机、工业、商业控制（智能工控设备、POS/ATM 机）、电子商务平台，甚至军事应用等。

2．开发实时操作系统

在实时 Linux 出现之前，在为实时应用选择系统平台的时候，人们大抵只有两种选择，要么使用 DOS 并自己编写所有必要的驱动程序，要么就得购买专用的实时系统。前者不仅费时费力，其性能也难以令人满意。而后者性能虽佳，其价格却高得让人难以接受。

实时 Linux 的出现解决了这一问题，它为实时应用领域的研究与开发提供了一个物美价廉的完备的操作系统平台。凭着自身的技术特色，借助于 Linux 的强大功能，实时 Linux 下开发出的实时应用有着不俗的表现。

1.4.2　开发高水平软件

自由软件联盟及“中国自由软件库”就已涵盖了操作系统、开发语言、视窗系统、数据库、网络、文字处理、排版及多媒体等各个领域，还有 VCD 解压源程序、路由器源程序等。利用自由软件让个人计算机带十几个硬盘实现阵列技术，及其亚微米超大规模集成电路 CAD 系统，可直接输出生产线控制数据等，这能让我们学到最先进的软件开发规范和开发技术，Linux 内核的许多面向通信的底层代码对开发我国自己的信息安全产品极有参考价值。

实际上，目前 Linux 的源代码中包含了世界各地几百名计算机高手的作品，分析这些源代码对于我们掌握核心技术会起到事半功倍的作用，尤其是各种驱动程序的编写，对于我们把软硬件结合起来发展民族信息产业至关重要。要改变目前我国软件开发在低层次上的重复过程，必须掌握操作系统的核心技术。

只要站在“巨人”的肩上，认真钻研，就一定能吃透它，利用它，研制出自己的解压芯片、路由器、磁盘阵列产品，开发出高级的 CAD 系统等，打破国外的技术封锁，振兴我国电子工业。

1.4.3　有助于计算机科学的教学和科研

对于从事计算机科学教学和科研的人来说，Linux 具有更多一层的意义。一般市场上出售的 UNIX，除了价格之外，还不提供其核心程序的源代码。这样，若想了解 UNIX 的内核，或在内核程序上作一些改进就很困难，更谈不上作为操作系统教学和科研的平台了，而 Linux 提供了从内核到上层的所有软件的全部源程序代码。在易于获得源代码的条件下，如果能对源代码的组织结构、实现原理及实现机制进行较详细的描述，那么对很多人深入了解源程序将有很大帮助。

实际上，Linux 也很适合教学用操作系统，一般的操作系统教材只讲操作系统的实现原理，学生既觉得抽象又感觉不到操作系统的重要价值。尽管有些书也是以 UNIX 为实例，但学生很难接触到 UNIX 操作系统，这对学生真正深入了解操作系统造成了困难。

国外很多大学已经把 Linux 作为教学用操作系统，我们认为这主要是因为：①Linux 平台易于建立；②Linux 内核源代码易于获得；③Linux 结构简单、清晰；④Linux 的实现采用了大量的数据结构，可以锻炼学生的抽象能力和知识应用能力。

可以说，Linux 内核源代码的开放乃至自由联盟各种应用程序源代码的开放，为我们的软件教学提供了活教材，我们的学生可以在这种“自由”文化的氛围下，学习并掌握软件开发的核心技术，我们就有希望在 21 世纪不仅拥有中文 Linux 操作系统，而且拥有适合中国国情的大量而优秀的 Linux 应用软件。

1.5　Linux 内核结构

Linux 是一个庞大、高效而复杂的操作系统，虽然它的开发起始于 Linus Torvalds 一

个人，但随着时间的推移，越来越多的人加入了 Linux 的开发和对它的不断完善。如何从整体上把握 Linux 内核的体系结构，对于 Linux 的开发者和分析者都至关重要。

1.5.1 Linux 内核在整个操作系统中的位置

Linux 的内核不是孤立的,必须把它放在整个系统中去研究,如图 1.1 所示,显示了 Linux 内核在整个操作系统的位置。

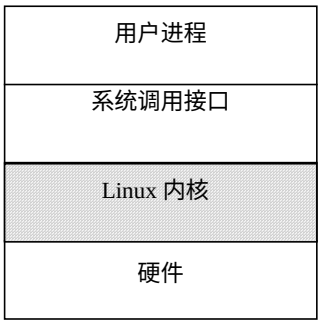

图 1.1 Linux 内核在整个操系统中的位置

由图 1.1 可以看出，Linux 操作系统由 4 个部分组成。

1．用户进程

用户应用程序是运行在 Linux 操作系统最高层的一个庞大的软件集合。当一个用户程序在操作系统之上运行时，它成为操作系统中的一个进程。

2．系统调用接口

在应用程序中，可通过系统调用来调用操作系统内核中特定的过程，以实现特定的服务。例如，在程序中安排一条创建进程的系统调用，则操作系统内核便会为之创建一个新进程。

系统调用本身也是由若干条指令构成的过程。但它与一般的过程不同，主要区别是：系统调用是运行在内核态（或叫系统态），而一般过程是运行在用户态。在 Linux 中，系统调用是内核代码的一部分。

3．Linux 内核

这是本书要讨论的重点。内核是操作系统的灵魂，它负责管理磁盘上的文件、内存，负责启动并运行程序，负责从网络上接收和发送数据包等。简言之，内核实际是抽象的资源操作到具体硬件操作细节之间的接口。

4．硬件

这个子系统包括了 Linux 安装时需要的所有可能的物理设备。例如，CPU、 内存、硬盘、网络硬件等。

上面的这种划分把整个 Linux 操作系统分为 4 个层次。把用户进程也纳入操作系统的范围内是因为用户进程的运行和操作系统密切相关，而系统调用接口可以说是操作系统内核的扩充，硬件则是操作系统内核赖以生存的物质条件。这 4 个层次的依赖关系表现为：上层依赖下层。

1.5.2　Linux 内核的作用

从程序员的角度来讲，操作系统的内核提供了一个与计算机硬件等价的扩展或虚拟的计算平台。它抽象了许多硬件细节，程序可以以某种统一的方式进行数据处理，而程序员则可以避开许多硬件细节。从另一个角度讲，普通用户则把操作系统看成是一个资源管理者，在它的帮助下，用户可以以某种易于理解的方式组织自己的数据，完成自己的工作并和其他人共享资源。

Linux 以统一的方式支持多任务，而这种方式对用户进程是透明的，每一个进程运行起来就好像只有它一个进程在计算机上运行一样，独占内存和其他的硬件资源，而实际上，内核在并发地运行几个进程，并且能够让几个进程公平合理地使用硬件资源，也能使各进程之间互不干扰安全地运行。

1.5.3　Linux 内核的抽象结构

Linux 内核由 5 个主要的子系统组成，如图 1.2 所示。

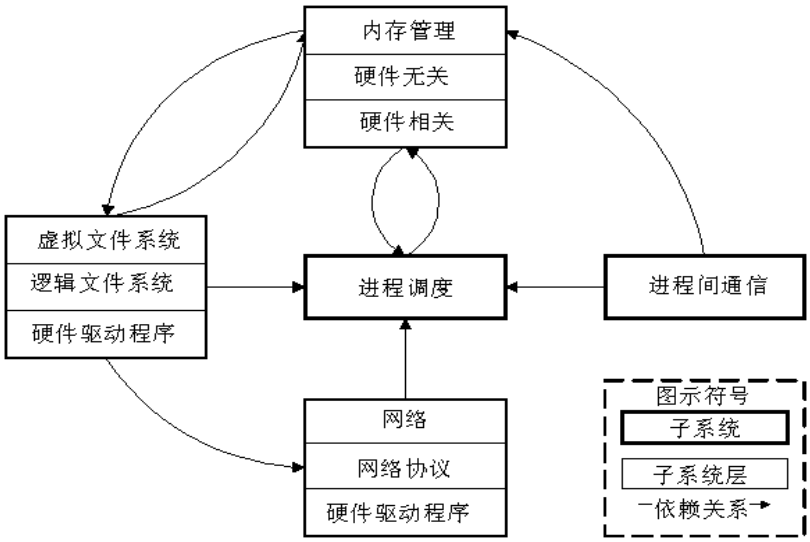

图 1.2　Linux 内核子系统及其之间的关系

(1) 进程调度（SCHED）控制着进程对 CPU 的访问。当需要选择下一个进程运行时，由调度程序选择最值得运行的进程。可运行进程实际是仅等待 CPU 资源的进程，如果某个进程在等待其他资源，则该进程是不可运行进程。Linux 使用了比较简单的基于优先级的进程调度算法选择新的进程。

(2) 内存管理（MM）允许多个进程安全地共享主内存区域。Linux 的内存管理支持虚拟内存，即在计算机中运行的程序，其代码、数据和堆栈的总量可以超过实际内存的大小，操

作系统只将当前使用的程序块保留在内存中，其余的程序块则保留在磁盘上。必要时，操作系统负责在磁盘和内存之间交换程序块。

内存管理从逻辑上可以分为硬件无关的部分和硬件相关的部分。硬件无关的部分提供了进程的映射和虚拟内存的对换；硬件相关的部分为内存管理硬件提供了虚拟接口。

(3) 虚拟文件系统 (Virtul File System，VFS) 隐藏了各种不同硬件的具体细节，为所有设备提供了统一的接口，VFS 还支持多达数十种不同的文件系统，这也是 Linux 较有特色的一部分。

虚拟文件系统可分为逻辑文件系统和设备驱动程序。逻辑文件系统指 Linux 所支持的文件系统，如 ext2，fat 等，设备驱动程序指为每一种硬件控制器所编写的设备驱动程序模块。

(4) 网络接口 (NET) 提供了对各种网络标准协议的存取和各种网络硬件的支持。网络接口可分为网络协议和网络驱动程序两部分。网络协议部分负责实现每一种可能的网络传输协议，网络设备驱动程序负责与硬件设备进行通信，每一种可能的硬件设备都有相应的设备驱动程序。

(5) 进程间通信 (IPC) 支持进程间各种通信机制。从图 1.2 所示可以看出，处于中心位置的是进程调度，所有其他的子系统都依赖于它，因为每个子系统都需要挂起或恢复进程。一般情况下，当一个进程等待硬件操作完成时，它被挂起；当操作真正完成时，进程被恢复执行。例如，当一个进程通过网络发送一条消息时，网络接口需要挂起发送进程，直到硬件成功地完成消息的发送，当消息被发送出去以后，网络接口给进程返回一个代码，表示操作的成功或失败。其他子系统 (内存管理，虚拟文件系统及进程间通信) 以相似的理由依赖于进程调度。

各个子系统之间的依赖关系如下。

- 进程调度与内存管理之间的关系：这两个子系统互相依赖。在多道程序环境下，程序要运行必须为之创建进程，而创建进程的第一件事，就是要将程序和数据装入内存。
- 进程间通信与内存管理的关系：进程间通信子系统要依赖内存管理支持共享内存通信机制，这种机制允许两个进程除了拥有自己的私有内存，还可存取共同的内存区域。
- 虚拟文件系统与网络接口之间的关系：虚拟文件系统利用网络接口支持网络文件系统 (NFS)，也利用内存管理支持 RAMDISK 设备。
- 内存管理与虚拟文件系统之间的关系：内存管理利用虚拟文件系统支持交换，交换进程 (swapd) 定期地由调度程序调度，这也是内存管理依赖于进程调度的唯一原因。当一个进程存取的内存映射被换出时，内存管理向文件系统发出请求，同时，挂起当前正在运行的进程。

除了如图 1.2 所示的依赖关系以外，内核中的所有子系统还要依赖一些共同的资源，但在图中并没有显示出来。这些资源包括所有子系统都用到的过程，例如分配和释放内存空间的过程，打印警告或错误信息的过程，还有系统的调试例程等。

1.6 Linux 内核源代码

为了深入地了解 Linux 的实现机制，还必须阅读 Linux 的内核源代码，下面是对有关源

代码的介绍。

1.6.1　多版本的内核源代码

对不同的内核版本，系统调用一般是相同的。新版本也许可以增加一个新的系统调用，但旧的系统调用将依然不变，这对于保持向后兼容是非常必要的——一个新的内核版本不能打破常规的过程。在大多数情况下，设备文件将仍然相同，而另一方面，版本之间的内部接口有所变化。

Linux 内核源代码有一个简单的数字系统，任何偶数内核（如 2.0.30）是一个稳定的版本，而奇数内核（如 2.1.42）是正在发展中的内核。本书是基于稳定的 2.4.16 源代码的。发展中的内核总是有最新的特点，支持最新的设备，尽管它们还不稳定，也许不是你所想要的，但它们是发展最新而又稳定的内核的基础。

目前，较新而又稳定的内核版本是 2.2.x 和 2.4.x，因为版本之间稍有差别，因此，如果你想让一个新驱动程序模块既支持 2.2.x，也支持 2.4.x，就需要根据内核版本对模块进行条件编译。

对内核源代码的修改是以补丁文件的形式发布的。patch 实用程序用来对内核源文件进行一系列的修订，例如，如果你有 2.4.9 内核源代码，而想移到 2.4.16，你可以获得 2.4.16 的补丁文件，应用 patch 来修订 2.4.9 源文件。例如：

```
$ cd /usr/src/linux
$ patch -p1 < patch-2.4.16
```

1.6.2　Linux 内核源代码的结构

Linux 内核源代码位于/usr/src/linux 目录下，其结构分布如图 1.3 所示，每一个目录或子目录可以看作一个模块，其目录之间的连线表示“子目录或子模块”的关系。下面是对每一个目录的简单描述。

include/目录包含了建立内核代码时所需的大部分包含文件，这个模块利用其他模块重建内核。

init/ 子目录包含了内核的初始化代码，这是内核开始工作的起点。

arch/子目录包含了所有硬件结构特定的内核代码,如图 1.3 所示,arch/子目录下有 i386 和 alpha 模块等。

drivers/ 目录包含了内核中所有的设备驱动程序，如块设备，scsi 设备驱动程序等。

fs/ 目录包含了所有文件系统的代码，如：ext2，vfat 模块的代码等。

net/ 目录包含了内核的连网代码。

mm/ 目录包含了所有的内存管理代码。

ipc/ 目录包含了进程间通信的代码。

kernel/　目录包含了主内核代码。

图 1.3 显示了 8 个目录，即 init、kernel、mm、ipc、drivers、fs、arch 及 net 的包含文件都在“include/”目录下。在 Linux 内核中包含了 drivers、fs、arch 及 net 模块，

这就使得 Linux 内核既不是一个层次式结构，也不是一个微内核结构，而是一个“整体式”结构。因为系统调用可以直接调用内核层，因此，该结构使得整个系统具有较高的性能，其缺点是内核修改起来比较困难，除非遵循严格的规则和编码标准。

在图 1.3 中所示的模块结构，代表了一种工作分配单元。利用这种结构，我们期望 Linus Torvalds 能维护和增强内核的核心服务，即 init/、kernel/、mm/及 ipc/，其他的模块 drivers、fs、arch 及 net 也可以作为工作单元，例如，可以分配一组人对块文件系统进行维护和进一步地开发，而另一组人对 scsi 文件系统进行完善。图 1.3 所示类似于 Linux 的自愿者开发队伍一起工作来增强和扩展整个系统的框架。

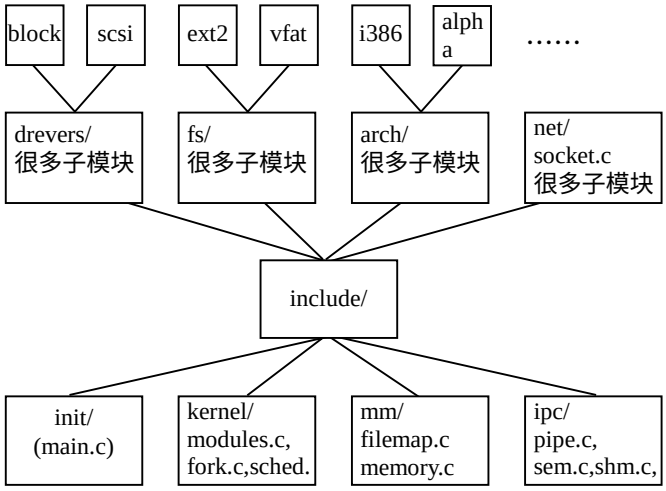

图 1.3　Linux 源代码的分布结构

1.6.3　从何处开始阅读源代码

像 Linux 内核这样庞大而复杂的程序看起来确实让人望而生畏，它像一个很大的球，没有起点和终点。在读源代码的过程中，你会遇到这样的情况，当读到内核的某一部分时又会涉及到其他更多的文件，当返回到原来的地方想继续往下读时，又忘了原来读的内容。在 Internet 上，很多人为此付出了很大的努力，制作出了源代码导航器，这为源代码阅读提供了很好的条件，下载站点为http://lxr.linux.no/source。下面给出阅读源代码的一些线索。

1．系统的启动和初始化

在基于 Intel 的系统上，当 loadlin.exe 或 LILO 把内核装入到内存并把控制权传递给内核时，内核开始启动。关于这一部分，看 arch/i386/kernel/head.S，head.S 进行特定结构的设置，然后跳转到 init/main.c 的 main（）例程。

2．内存管理

内存管理的代码主要在/mm，但特定结构的代码在 arch/*/mm。缺页中断处理的代码在

mm/memory.c ，而内存映射和页高速缓存器的代码在 mm/filemap.c。缓冲器高速缓存是在 mm/buffer.c 中实现，而交换高速缓存是在 mm/swap_state.c 和 mm/swapfile.c 中实现。

3．内核

内核中，特定结构的代码在 arch/*/kernel，调度程序在 kernel/sched.c，fork 的代码在 kernel/fork.c，task_struct 数据结构在 include/linux/sched.h 中。

4．PCI

PCI 伪驱动程序在 drivers/pci/pci.c ，其定义在 include/linux/pci.h。每一种结构都有一些特定的 PCI BIOS 代码，Intel 的在 arch/alpha/kernel/bios32.c。

5．进程间通信

所有 System V IPC 对象权限都包含在 ipc_perm 数据结构中，这可以在 include/linux/ipc.h 中找到 System V 消息是在 ipc/msg.c 中实现，共享内存在 ipc/shm.c 中，信号量在 ipc/sem.c 中，管道在 ipc/pipe.c 中实现。

6．中断处理

内核的中断处理代码是几乎所有的微处理器所特有的。中断处理代码在 arch/i386/kernel/irq.c 中，其定义在 include/asm-i386/irq.h 中。

7．设备驱动程序

Linux 内核源代码的很多行是设备驱动程序。Linux 设备驱动程序的所有源代码都保存在/driver，根据类型可进一步划分为：

/block

块设备驱动程序如 ide（在 ide.c）。如果想看包含文件系统的所有设备是如何被初始化的，应当看 drivers/block/genhd.c 中的 device_setup（），device_setup（）不仅初始化了硬盘，当一个网络安装 nfs 文件系统时，它也初始化网络。块设备包含了基于 IDE 和 SCSI 的设备。

/char

这是看字符设备（如 tty，串口及鼠标等）驱动程序的地方。

/cdrom

Linux 的所有 CDROM 代码都在这里，如在这儿可以找到 Soundblaster CDROM 的驱动程序。注意 ide CD 的驱动程序是 ide-cd.c，放在 drivers/block；SCSI CD 的驱动程序是 scsi.c，放在 drivers/scsi。

/pci

这是 PCI 伪驱动程序的源代码，在这里可以看到 PCI 子系统是如何被映射和初始化的。

/scsi

在这里可以找到所有的 SCSI 代码及 Linux 所支持的 scsi 设备的所有设备驱动程序。

/net

在这里可以找到网络设备驱动程序，如 DECChip 21040 PCI 以太网驱动程序在 tulip.c

中。

/sound

这是所有声卡驱动程序的所在地。

8．文件系统

EXT2 文件系统的源代码全部在 fs/ext2/ 目录下，而其数据结构的定义在 include/linux/ ext2_fs.h,ext2_fs_i.h 及 ext2_fs_sb.h 中。虚拟文件系统的数据结构在 include/linux/fs.h 中描述，而代码是在 fs/*中。缓冲区高速缓存与更新内核的守护进程的实现是在 fs/buffer.c 中。

9．网络

网络代码保存在/net 中，大部分的 include 文件在 include/net 下，BSD 套节口代码在 net/socket.c 中，IP 第 4 版本的套节口代码在 net/ipv4/af_inet.c。一般的协议支持代码（包括 sk_buff 处理例程）在 net/core 下，TCP/IP 联网代码在 net/ipv4 下，网络设备驱动程序在/drivers/net 下。

10．模块

内核模块的代码部分在内核中，部分在模块包中，前者全部在 kernel/modules.c 中，而数据结构和内核守护进程 kerneld 的信息分别在 include/linux/module.h 和 include/linux/kerneld.h 中。如果想看 ELF 目标文件的结构，它位于 include/linux/elf.h 中。

1.7 Linux 内核源代码分析工具

凡是尝试做过内核分析的人都知道，Linux 的内核组织结构虽然非常有条理，但是，它毕竟是众人合作的结果，在阅读代码的时候要将各个部分结合起来，确实是件非常困难的事情。因为在内核中的代码层次结构肯定分多个层次，那么对一个函数的分析，肯定会涉及到多个函数，而对每一个函数都可能有多层的调用，一层层下来，直接在代码文件中查找肯定会让你失去耐心和兴趣的。最后，很多人只能望洋兴叹，或者只是找一本书来看看基本原理，却不敢去说自己真正看过内核源代码。任何一本原理书只能是你阅读源代码的导航器，绝不能代替源代码的阅读。

俗话说：“工欲善其事，必先利其器”。面对 Linux 这样庞大的源代码，必须有相应工具的支持才能使分析有效地进行下去。在此介绍两种源代码的分析工具，希望能对感兴趣的读者有所帮助。

1.7.1　Linux 超文本交叉代码检索工具

Linux 超文本交叉代码检索工具 LXR（Linux Cross Reference），是由挪威奥斯陆大学数学系 Arne Georg Gleditsch 和 Per Kristian Gjermshus 编写的。这个工具实际上运行在 Linux 或者 UNIX 平台下，通过对源代码中的所有符号建立索引，从而可以方便地检索任何一个符号，包括函数、外部变量、文件名、宏定义等。不仅仅是针对 Linux 源代码，对于 C 语言的其他大型的项目，都可以建立其 LXR 站点，以便开发者查询代码，以及后继开发者学习代码。

目前的 LXR 是专门为 Linux 下面的 Apache 服务器设计的，通过运行 perl 脚本，检索在安装时根据需要建立的源代码索引文件，将数据发送到网络客户端的 Web 浏览器上。任何一种平台上的 Web 浏览器都可以访问，这就方便了习惯在 Windows 平台下工作的用户。

LXR 的英文网站为 http://lxr.linux.no/，在中国 Linux 论坛 http://www.linuxforum.net上有其镜像。

读者如果想建立自己的LXR网站,则可直接通过http://lxr.linux.no/lxr-0.3.tar.gz，下载 LXR 的 tarball 形式的安装包。另外，因为 LXR 使用 glimpse 作为整个项目中文本的搜索工具，因此还需要下载 glimpse，位置在 http://glimpse.cs.arizona.edu ，下载 glimpse-4.12.6.bin.Linux- 2.2.5-22-i686.tar.gz，也可以使用更新的版本下载以后按照说明进行安装和配置，就可以建立自己的 LXR 网站。如果你上网很方便，就可以直接从 http://lxr.linux.no/网站查询所需要的各种源代码信息。

1.7.2　Windows 平台下的源代码阅读工具（Source Insight）

为了方便地学习 Linux 源程序，我们不妨回到我们熟悉的 Windows 环境下。但是在 Windows 平台上，使用一些常见的集成开发环境，效果也不是很理想，比如难以将所有的文件加进去，查找速度缓慢，对于非 Windows 平台的函数不能彩色显示。在 Windows 平台下有一个强大的源代码编辑器，它的卓越性能使得学习 Linux 内核源代码的难度大大降低，这便是 Source Insight 3.0，它是一个 Windows 平台下的共享软件，可以从 http://www.sourceinsight.com/上下载30天试用版本。由于Source Insight是一个Windows平台的应用软件，所以首先要通过相应手段把 Linux 系统上的程序源代码移到 Windows 平台下,这一点可以通过在 linux 平台上将/usr/src 目录下的文件拷贝到 Windows 平台的分区上，或者从网上或光盘中直接拷贝文件到 Windows 平台的分区上。

这个软件的安装非常简单，双击安装文件名，然后按提示进行即可。安装完成后，就可启动该程序。这个软件使用起来也非常简单：先选择 Project 菜单下的 new，新建一个工程，输入工程名，接着要求你把欲读的源代码加入（可以加入整个目录）后，该软件就分析你所加的源代码。分析完后，就可以进行阅读了。对于打开的阅读文件，如果想看某一变量的定义，先把光标定位于该变量，然后单击工具条上的相应选项，该变量的定义就显示出来。对于函数的定义与实现也可以同样操作。