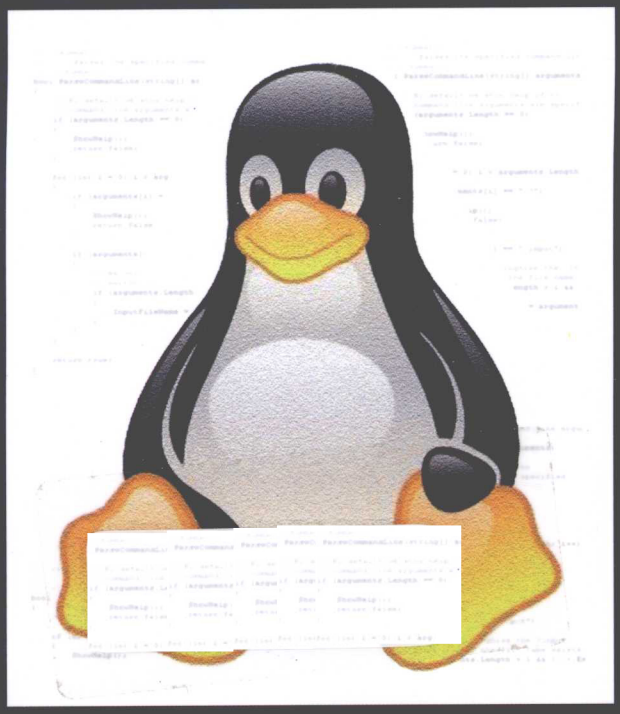
HZ BOOKS
华章科技

首本Linux内核API手册　多重索引方式方便速查

Linux内核API完全参考手册

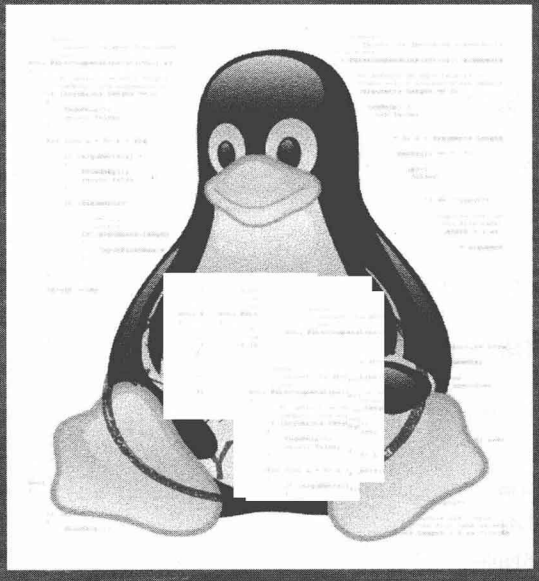
邱铁 周玉 邓莹莹 编著



机械工业出版社
China Machine Press

Linux内核API完全参考手册

邱铁 周玉 邓莹莹 编著



机械工业出版社
China Machine Press

Linux作为源码开放的操作系统已经广泛应用于计算机与嵌入式设备，因此学会Linux内核开发与编程显得越来越重要。本书以最新的Linux内核版本2.6.30为依据，对常用的内核API作了系统分析和归纳，设计了典型实例并对开发场景进行了详细讲解。本书中分析的内核API模块包括：内核模块机制API、进程管理内核API、进程调度内核API、中断机制内核API、内存管理内核API、内核定时机制API、内核同步机制API、文件系统内核API和设备驱动及设备管理API。

本书立足Linux内核API分析，深入实践，内容翔实，读者可以从低起点进行高效的内核分析与编程实践。本书可作为高等院校计算机、电子、信息类大学生及研究生进行Linux操作系统学习和编程的教材或参考书，也可作为Linux开发人员和广大Linux编程开发爱好者的参考用书。

图书在版编目（CIP）数据

Linux内核API完全参考手册/邱铁，周玉，邓莹莹编著. —北京：机械工业出版社，2011.1

ISBN 978-7-111-32357-0

Ⅰ.L…　Ⅱ.①邱…　②周…　③邓…　Ⅲ.Linux操作系统—技术手册　Ⅳ.TP316.89-62

中国版本图书馆CIP数据核字（2010）第211364号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）
责任编辑：秦　健
北京京北印刷有限公司印刷
2011年1月第1版第1次印刷
185mm×260mm · 43.5印张
标准书号：ISBN 978-7-111-32357-0
定价：79.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88378991；88361066
购书热线：（010）68326294；88379649；68995259
投稿热线：（010）88379604
读者信箱：hzjsj@hzbook.com

前　言

进入21世纪，IT技术以前所未有的速度向前发展。Linux作为源码开放的操作系统，在众多爱好者的共同努力下，不断成长并趋于完善。由于GNU计划所开发的各种组件和系统发行版所必备的软件可以运行于Linux内核之上，整个Linux内核符合通用公共许可证（General Public License, GNU），使得Linux在PC机、服务器以及嵌入式系统开发等领域得到了广泛应用。

编者在长期的Linux内核开发中发现，当前介绍内核API方面的书籍很少。目前市面上关于Linux内核编程开发方面的书可以分为三类：第一类，Linux内核分析，所分析的内核源码版本一般相对较早，而对于最新版本的内核源代码则很少提及；第二类，Linux编程，主要是以用户层面上的编程为主，一般涉及用户API；第三类，嵌入式Linux开发，相对于特定的硬件平台，只对所用到的特定内核API作简要说明。对于使用Linux内核进行编程开发，需要全面了解内核API，而目前市面上找不到一本能够全面介绍最新的Linux内核API的图书，这也正是本书写作的目的所在。

本书的编写工作从2009年6月开始，所有的内核API验证实例均基于最新的Linux内核源代码2.6.30版本，当时2.6.30内核刚刚发布。经过近15个月的源代码分析、编程实践与实例验证，对常用的内核API作了系统归纳，并编写了典型验证程序，做到了理论分析与实际编程的统一。本书分析的内核API模块包括：内核模块机制API、进程管理内核API、进程调度内核API、中断机制内核API、内存管理内核API、内核定时机制API、内核同步机制API、文件系统内核API和设备驱动及设备管理API。

在实例编写过程中，感谢陈潇参与了第8章部分实例的验证，开放源码项目组王伟、王亚坤和于龙等参与项目讨论并分析实例。另外，我们在编写过程中听取了同事、同行专家的意见和建议，并参阅了大量国内外文献的精华资料，特别是活跃在开放源码社区的Linux爱好者，向他们表示感谢。

由于Linux更新速度较快，再加上编者所具备知识的广度和深度所限，书中存在的错误与不当之处请各位同仁批评指正。对于书中的问题，读者可以发送到E-mail：openlinux2100@gmail.com，及时和作者交流，以便再版时更正与完善。

编者

2010年10月1日于大连

本书使用方法

Linux内核API分析必备知识

- 介绍了内核分析与开发的基础知识。

检索方法1

- 本书目录按照Linux功能模块大类分为9大模块；
- 每个模块内部的内核API函数按照模块名称顺序排列。

检索方法2

- 本书附录将所有常用内核API函数按照英文字母表顺序排列。

目　录

第 1 章　Linux内核API分析必备知识

从这里开始，我们来探索Linux内核2.6.30版本的内核API。内核API与用户API是具有本质区别的，因为它们所运行的系统模式是不同的。若要进行Linux内核源代码分析与内核API验证，需要具备一定的基础知识，掌握了这些基础知识后，才能在Linux内核源代码分析与内核API验证实例的理解中做到游刃有余。

Linux内核编程注意事项

Linux可以运行在两种模式下：用户模式（user mode）和内核模式（kernel mode）。当我们编写一个普通程序时，有时会包含stdlib.h文件，也就是说我们使用了C标准库，这是典型的用户模式编程，在这种情况下，用户模式的应用程序要链接标准C库。在内核模式下不存在libc库，也就没有这些函数供我们调用。

此外，在内核模式下编程还存在一些限制：

- 不能使用浮点运算。因为Linux内核在切换模式时不保存处理器的浮点状态。
- 不要让内核程序进行长时间等待。Linux操作系统本身是抢占式的，但是内核是非抢占内核，就是说用户空间的程序可以抢占运行，但是内核空间程序不可以。
- 尽可能保持代码的整洁性。内核调试不像调试应用程序那样方便，因此，在前期代码编写的过程中保持代码的整洁易懂，将大大方便后期的调试。
- 在内核模式下编程，系统内的所有资源都是由内核来统一调配的，并且数量有限，因此申请资源用完后一定要进行释放，避免出现死锁情况。
- Linux内核API有很多配对使用，例如，文件引用计数有加操作，也会有相应的减操作。如果在实验中进行了“引用计数”加操作，函数执行后未进行减操作还原，那么可能会出现系统崩溃。

本书中的所有内核API验证实例都是在Linux内核模式下进行编程与验证的。

本书中模块编译Makefile模板

在Linux 2.6内核中，模块的编译需要配置过的内核源代码；编译过程首先会到内核源码目录下读取顶层的Makefile文件，然后再返回模块源码所在目录；经过编译、链接后生成的内核模块文件的后缀为.ko。

2.6内核模块的Makefile模板：

```
ifneq ($(KERNELRELEASE),)
mymodule-objs:= mymodule 1.o  mymodule 2.o     #依赖关系
obj-m += mymodule.o            #编译、链接后将生成mymodule.o模块

else
PWD := $(shell pwd)
KVER := $(shell uname -r)
KDIR := /lib/modules/$(KVER)/build

all:
    $(MAKE) -C $(KDIR) M=$(PWD)   #此处将再次调用make
```

```
clean:
    rm -rf *.o *.mod.c *.ko *.symvers *.order *.markers *~
endif
```

当在命令行执行make命令时，将调用Makefile文件。KERNELRELEASE是在内核源码的顶层/usr/src/linux-2.6.30/Makefile文件中定义的一个变量，位置在第358行，如图1-1所示。在第一次读取执行此Makefile时，变量$（KERNELRELEASE）没有被设置，因此第一行ifneq的条件失败，从else后面开始执行，设置PWD、KVER和KDIR等变量。

```
350  KBUILD_CPPFLAGS := -D__KERNEL__
351
352  KBUILD_CFLAGS   := -Wall -Wundef -Wstrict-prototypes -Wno-trigraphs \
353                     -fno-strict-aliasing -fno-common \
354                     -Werror-implicit-function-declaration
355  KBUILD_AFLAGS   := -D__ASSEMBLY__
356
357  # Read KERNELRELEASE from include/config/kernel.release (if it exists)
358  KERNELRELEASE = $(shell cat include/config/kernel.release 2> /dev/null)
359  KERNELVERSION = $(VERSION).$(PATCHLEVEL).$(SUBLEVEL)$(EXTRAVERSION)
360
361  export VERSION PATCHLEVEL SUBLEVEL KERNELRELEASE KERNELVERSION
362  export ARCH SRCARCH CONFIG_SHELL HOSTCC HOSTCFLAGS CROSS_COMPILE AS LD CC
363  export CPP AR NM STRIP OBJCOPY OBJDUMP MAKE AWK GENKSYMS PERL UTS_MACHINE
364  export HOSTCXX HOSTCXXFLAGS LDFLAGS_MODULE CHECK CHECKFLAGS
365
```

图1-1　内核源码的顶层Makefile

当make到标号all时，-C$(KDIR)指明跳转到内核源码目录下读取那里的Makefile。M=$(PWD)，表明返回到当前目录继续读入、执行当前的Makefile，也就是第二次调用make。这时的$(KERNELRELEASE)已被定义，因此语句ifneq成功，make将继续读取紧接在ifneq后面的内容。ifneq的内容为kbuild语法的语句，指明模块源码中各文件之间的依赖关系和要生成的目标模块名称。

语句“mymodule-objs:= mymodule 1.o mymodule2.o”表示mymoudule.o由mymodule1.o与mymodule2.o链接生成。语句“obj-m:=mymodule.o”表示编译链接后将生成mymodule.ko模块，这个文件就是要插入内核的模块文件。

如果make的目标是clean，直接执行clean标号后的操作，也就清除*.o *.mod.c *.ko *.symvers *.order *.markers *~这些文件操作。执行完clean后面的rm命令后，整个make工作就结束了。

内核调试函数printk

本书中的模块代码中用到了内核调试函数printk( )，在用户空间里我们经常使用C语言函数printf来向标准输出终端流打印信息。printk( )是内核使用的函数，因为内核没有链接标准C函数库，其实printk( )接口和printf( )基本相似，printk( )函数能够在终端一次最多显示大小为1024个字节的字符串。printk( )函数执行时首先设法获取控制台信号量，然后将要输出的字符存储到控制台的日志缓冲区，再调用控制台驱动程序来刷新缓冲区。若printk( )无法获得控制台信号量，则只能把要输出的字符存储到日志缓冲区，并依赖拥有控制台信号量的进程来刷新这个缓冲区。printk( )函数会将数据存储到日志缓冲区，但是为了安全考虑，在这之前需要使用日志缓冲区锁，保证并发调用printk( )的安全性。

内核模式下系统信息输出函数printk( )与用户模式下的printf( )函数在输出内容上也是有区别的，主要是两点：一是内核在切换模式时不保存处理器的浮点状态，因此printk( )并不支持浮点数运算；二是printk( )可以指定一个记录级别，内核根据这个级别来判断是否在终端上打印消息，而printf( )函数则不需要。

printk( )的语法格式为：

```
printk(记录级别 "格式化输出信息");
```

或者是采用编号形式：

```
printk("<记录级别编号>格式化输出信息");
```

其中“记录级别”是在include/linux/kernel.h中的简单宏定义，其在内核源代码中的形式如图1-2所示的第91~98行。它们扩展开是如“<number>”这样的字符串，加入到printk( )函数要打印的消息的前面。

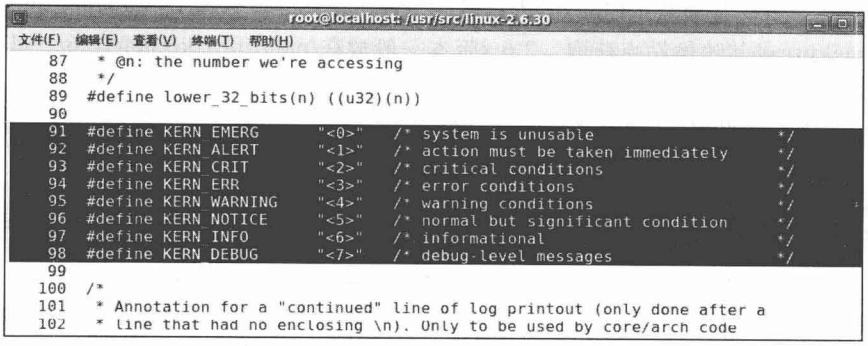


图1-2　记录级别在内核源代码中的宏定义

内核用这个指定的记录等级和当前终端的记录等级console_loglevel进行比较，从而决定是否向终端打印输出。表1-1给出了所有记录等级、字符串编号及说明。

表1-1　记录等级说明

记录等级	字符串代号	描　述
KERN_EMERG	“<0>”	这是一个最高优先输出的紧急事件消息，表示操作系统崩溃前会进行输出提示信息
KERN_ALERT	“<1>”	输出警告消息，通知需要采取措施
KERN_CRIT	“<2>”	这是一个临界情况，当发生严重的软件或硬件操作失败时，进行输出提示信息
KERN_ERR	“<3>”	当系统检测到发生一个错误时，会输出信息。设备驱动程序常用KERN_ERR来报告硬件的错误信息
KERN_WARNING	“<4>”	提示信息，一般用于提醒。常用在与系统安全相关的消息输出
KERN_NOTICE	“<5>”	这是一个普通的提示，系统输出时需要注意的情况信息
KERN_INFO	“<6>”	非正式的消息，可能无关紧要，例如驱动程序进行挂载时，一般打印硬件相关信息
KERN_DEBUG	“<7>”	这是用于程序开发与调试的信息，完成编码后，这类信息一般都要删除

内核将最重要的记录等级KERN_EMERG定为“<0>”，将无关紧要的记录等级“KERN_DEBUG”定为“<7>”。

例如：

```
printk("没有等级信息，则采用默认级别！\n");
printk(KERN_INFO "内核提示信息\n");
printk(KERN_DEBUG "内核调试信息\n");
```

如果没有特别指定一个记录等级，函数会选用默认的DEFAULT_MESSAGE_LOGLEVEL，现在默认等级是KERN_WARNING。由于这个默认值将来存在变化的可能性，所以还是应该为自定义的消息指定一个记录等级。

本章后面的实例中考虑到要让系统强制输出信息，因此使用了“<0>”级别，然后通过dmesg |

tail或者是dmesg -c来查看系统输出信息。

内核编译与定制

获得Linux内核与补丁

要编译Linux，首先是要获得Linux的内核源码（kernel source code）。最新的Linux官方源码可以从www.kernel.org或其映像站点获取，2.6.x版本一般放在/pub/linux/kernel/v2.6/，其在官方网站上的目录索引如图1-3所示。

将下载的内核源代码放在Linux系统目录文件夹/usr/src/中。本书用以下命令下载最新2.6.30内核源码包。

```
cd  /usr/src/
wget http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.30.tar.bz2
```

下载Linux 2.6.30内核补丁，其在官方网站上的目录索引如图1-4所示。

下载补丁的命令如下：

```
cd  /usr/src/
wget http://www.kernel.org/pub/linux/kernel/v2.6/patch-2.6.30.bz2
```

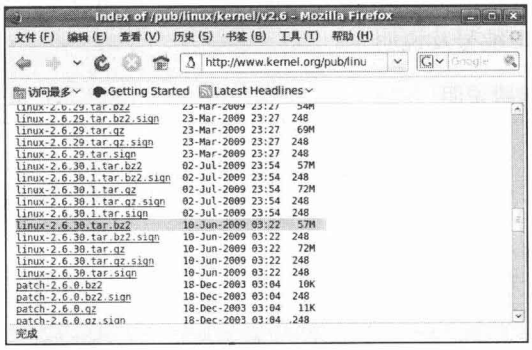


图1-3　Linux内核网页目录索引

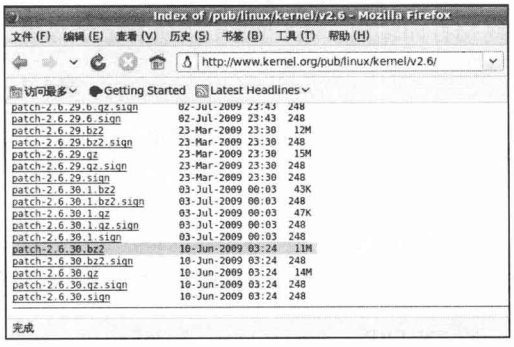


图1-4　Linux内核补丁网页目录索引

准备编译需要的工具

要想顺利完成内核编译，首先要检查或安装必要的工具：

1）安装gcc、make等编译工具：

```
apt-get install build-essential
```

2）安装make menuconfig时必需的库文件：NCurses (libncurses5-dev或ncurses-devel)，这是当make menuconfig时用作生成菜单窗口的程序库：

```
apt-get install libncurses-dev
apt-get install kernel-package
```

3）安装Linux系统生成kernel-image的一些配置文件和工具：

```
apt-get install fakeroot
apt-get install initramfs-tools, module-init-tools
```

4）在编译Linux内核时，通常还需要以下工具（这些工具一般是可选的）：

- GNU C++ Compiler (g++ 或 gcc-c++) - 编译make xconfig使用的Qt窗口时需要；
- Qt 3 (qt-devel 或 qt3-devel) - make xconfig时用作Qt窗口的程序库；

• GTK+ (gtk+-devel) - make gconfig时用作GTK+窗口的程序库；

• Glade (libglade2-devel) - 要编译make gconfig时的GTK+窗口时需要。

在Ubuntu系统中，我们可以使用下面的命令来获得相关的软件包：

```
apt-get update
apt-get install libncurses5-dev wget bzip2
```

解压内核

如果下载了GNU Zip格式的Linux核心源码压缩档（文件扩展名称为*.tar.gz），可以用指令“tar zxvf linux-版本编号.tar.gz”解压，例如：

```
tar  xzvf  linux-2.6.30.tar.gz
```

如果下载了BZip2的Linux核心源码压缩档（文件扩展名称为*.tar.bz2），可以用指令“tar jxvf linux-版本编号.tar.bz2”解压，本书介绍的是bz2的压缩包，如下所示：

```
tar  xjvf  linux-2.6.30.tar.bz2
```

把源码包解压到/usr/src中，通过运行解压命令后，发现/usr/src中多了一个linux-2.6.30文件夹，如图1-5所示。

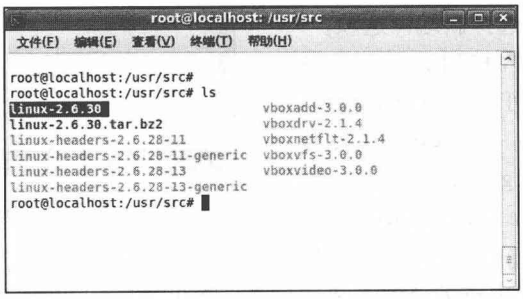


图1-5　内核解压后的文件夹

给内核打补丁

这一步在内核的编译过程中是可选的，如果你对内核有特殊的要求，可以将自己写的补丁打到内核中去。

对于本章中所下载的linux-2.6.30内核源码包在PC机上编译是不需要这一步骤的。如果读者有新的要求，可以写补丁包，例如，对当前的linux-2.6.30制作的补丁包（文件名为patch-2.6.30.bz2），可以使用patch命令给Linux内核源码打入补丁：

```
cd linux-2.6.30
bzcat ../patch-2.6.30.bz2 | patch -p1
```

由于我们是在PC机进行的Linux内核API验证，因此，这一步骤省略了。

设定编译选项

当编译Linux内核时，其中一个最重要的步骤就是定制新内核的配置选项，需要确定哪些是必选的，哪些是要编译成可加载模块（Loadable Modules）时进行动态加载，哪些不需要编译进新内核中。这些要根据使用的具体情况而定，定制的原则是：在满足功能需求的前提下，使新内核占用空间最少、耗费资源最少、运行速度最快。

在定制编译选项时，Linux系统提供了多个方法进行设定编译选项：

• config

- menuconfig
- xconfig
- gconfig
- oldconfig

也可以通过以下的命令来取得旧编译选项（注意，如果是初学者编译Linux内核，可以与旧的编译选项进行对比选择）：

```
cp /boot/config-`uname -r` .config
```

Make config为终端问答文字格式的编译选项，make menuconfig为菜单选项格式的配置界面如图1-6所示，在这里可以通过键盘来设置各个选项。

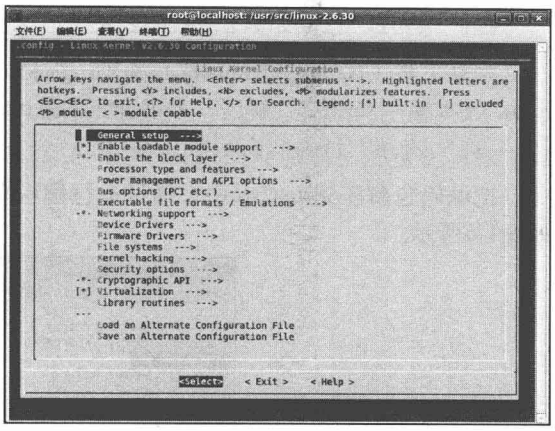


图1-6 make menuconfig编译选项界面

make xconfig为基于QT/Tcl的图形配置界面，如图1-7所示。make gconfig为基于GTK+的图形配置界面，如图1-8所示。这两个基于图形界面的编译选项均可通过鼠标操作编译选项，操作比较方便。

make oldconfig命令只选择新编译选项。一般情况下，当编译Linux内核时，执行make menuconfig弹出对话框，可以对内核的编译选项进行新的设定，并生成一个.config文件，make的时侯就是根据.config的设定值进行编译内核。如果再重新执行make menuconfig时，内核编译选项又重新回到了以前的默认值。当我们设定了内核编译选项之后，执行make oldconfig命令就可以保存前面设定好的内核编译配置选项。当下次再执行make menuconfig命令时，出现的设定就是前一次的设定内容。

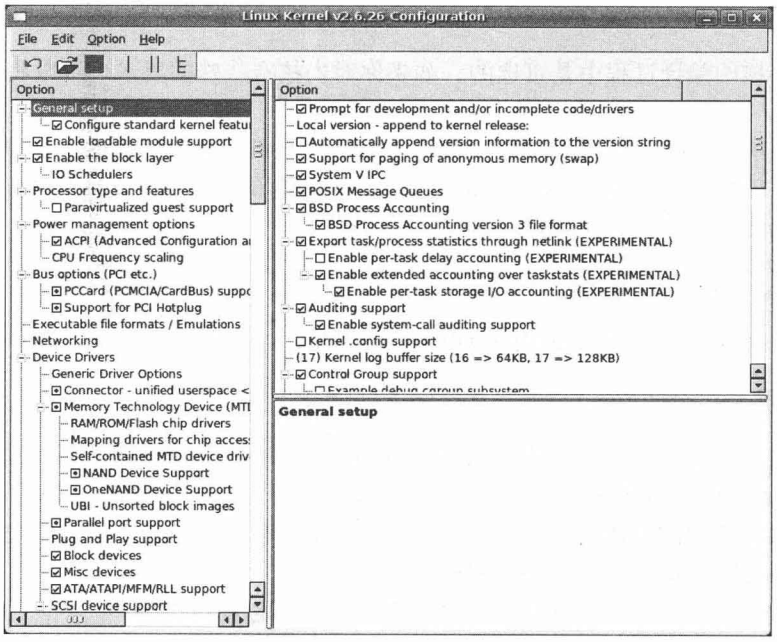


图1-7 make xconfig编译选项界面

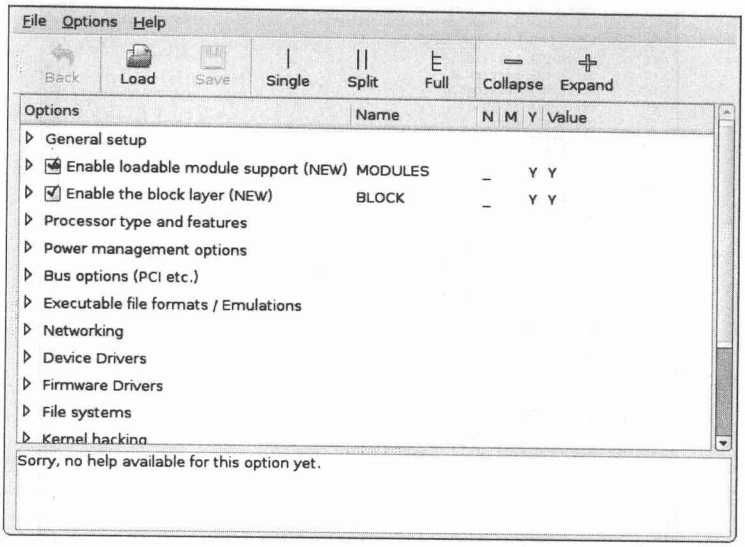


图1-8 make gconfig编译选项界面

注意 如果是在PC机下，没有特殊的要求，初学者编译内核，可以先选择“默认”的编译选项。一旦出现问题，便可以对照旧的编译配置文件，逐步查找并解决问题。

编译与安装内核

首先用make mrproper命令清除所有旧的配置和旧的编译目标等文件：

```
cd /usr/src/linux-2.6.30
make mrproper
```

接着执行命令make来编译内核，在默认情况下，make是一个顺序执行的工具。它按次序调用底层编译器来编译C/C++源文件。在某些情况下，有的源文件不需以其他源文件为基础即可编译，这时可以使用-j选项调用make来完成并行编译操作。make指令格式如下：

```
make -jn
```

n代表同时编译的进程，可以加快编译速度，n由用户计算机的配置与性能决定，当前的典型值为10。make编译内核过程如图1-9所示。

经过以上编译内核操作，将会在目录arch/x86/boot下生成名为“bzImage”的文件（如图1-10所示）即编译出来的新内核。为方便管理，需要把它移动至目录/boot中，并改名为“vmlinuz-核心版本”。为保存编译选项方便日后参考，同时也要把.config复制至/boot中，并改名为 “config-核心版本”。我们可以通过输入命令make install来完成这些步骤：

```
make install
```

接下来执行命令：

```
make modules
```

进行编译模块，如图1-11所示。

最后执行命令：

```
make modules_install
```

make modules_install是将内核模块安装到/lib/modules中，其执行过程如图1-12所示。