

## [linux 内核] 2.6 内核的配置与编译

首先对内核进行菜单配置，

代码：

```
1.  cd /usr/src/linux
2.  make menuconfig
```

[复制代码](#)

代码成熟度选项，

代码：

```
1.  Code maturity level options --->
2.  [*] Prompt for development and/or
    incomplete code/drivers
3.  [*] Select only drivers expected to
    compile cleanly
```

[复制代码](#)

打开使用开发中、不完全的代码/驱动会让内核配置多出很多选项，由于我们需要使用一些正在开发中的功能，因此必需打开这一选项。

通用设置选项

代码：

```
1.  General setup --->
2.  () Local version - append to kernel
    release
3.  [*] Support for paging of anonymous
    memory (swap)
4.  [*] System V IPC
5.  [*] POSIX Message Queues
```

```

6.      [*] BSD Process Accounting
7.      [*]   BSD Process Accounting version 3
           file format
8.      [*] Sysctl support
9.      [ ] Auditing support
10.     (15) Kernel log buffer size (16 => 64KB,
           17 => 128KB)
11.     [*] Support for hot-pluggable devices
12.     [*] Kernel Userspace Events
13.     [*] Kernel .config support
14.     [*]   Enable access to .config through
           /proc/config.gz
15.     [*] Configure standard kernel features
           (for small systems) --->
16.     --- Configure standard kernel features
           (for small systems)
17.     [ ]   Load all symbols for
           debugging/kksymoops
18.     [*]   Enable futex support
19.     [*]   Enable eventpoll support
20.     [*]   Optimize for size
21.     [*]   Use full shmem filesystem
22.     (0)   Function alignment
23.     (0)   Label alignment
24.     (0)   Loop alignment
25.     (0)   Jump alignment

```

[复制代码](#)

Local version - append to kernel release: 这里填入的是 64 字符以内的字符串，你在这里填上的字符串可以用 `uname -a` 命令看到。

Support for paging of anonymous memory (swap): 这

是使用交换分区或者交换文件来做为虚拟内存的，当然要选上了。

System V IPC: 表示系统 5 的 Inter Process Communication, 它用于处理器在程序之间同步和交换信息, 如果不选这项, 很多程序运行不起来的。

POSIX Message Queues: 这是 POSIX 的消息队列, 它同样是一种 IPC。建议你最好将它选上。

BSD Process Accounting: 这是允许用户进程访问内核将账户信息写入文件中的。这通常被认为是个好主意, 建议你最好将它选上。

Sysctl support: 这个选项能不重新编译内核修改内核的某些参数和变量, 如果你也选择了支持/proc, 将能从/proc/sys 存取可以影响内核的参数或变量。建议你最好将它选上。

Auditing support: 审记支持, 用于和内核的某些子模块同时工作, 例如 SELinux。只有选择此项及它的子项, 才能调用有关审记的系统调用。

Kernel log buffer size: 内核日志缓存的大小, 12 => 4 KB, 13 => 8 KB, 14 => 16 KB 单处理器, 15 => 32 KB 多处理器, 16 => 64 KB for x86 NUMAQ or IA-64, 17 => 128 KB for S/390。

Support for hot-pluggable devices: 是否支持热插拔的选项, 肯定要选上。不然 USB、PCMCIA 等这些设备都用不了。

Kernel Userspace Events: 内核中分为系统区和用户区, 这里系统区和用户区进行通讯的一种方式, 选上。

Kernel .config support: 将.config 配置信息保存在内核中，选上它及它的子项使得其它用户能从/proc 中得到内核的配置。还记得另一篇贴子我是如何取得启动光盘的内核配置信息，并在此基础上配置新的内核吗？

Configure standard kernel features (for small systems): 这是为了编译某些特殊的内核使用的，通常你可以不选择这一选项，你也不用对它下面的子项操心了。

Load all symbols for debugging/kksymoops: 是否装载所有的调试符号表信息，如果你不需要对内核调试，不需要选择此项。

Enable futex support: 不选这个内核不一定能正确的运行使用 glibc 的程序，当然要选上。

Enable eventpoll support: 不选这个内核将不支持事件轮循的系统调用，最好选上。

Optimize for size: 这个选项使 gcc 使用-Os 的参数而不是-O2 的参数来优化编译，以获得更小尺寸的内核，建议选上。

Use full shmem filesystem: 除非你在很少的内存且不使用交换内存时，才不要选择这项。

后面的这四项都是在编译时内存中的对齐方式，0 表示编译器的默认方式。使用内存对齐能提高程序的运行速度，但是会增加程序对内存的使用量。

内核也是一组程序呀。

可加载模块，

代码：

1. Loadable module support --->
2. ☒ Enable loadable module support
3. ☒ Module unloading
4. ☐ Forced module unloading
5. ☒ Module versioning support  
(EXPERIMENTAL)
6. ☐ Source checksum for all modules
7. ☒ Automatic kernel module loading

[复制代码](#)

#### Enable loadable module

support, 很多人喜欢将全部功能、硬件支持一股脑的编进内核, 而不是使用模块的方式。这样做非常不好 (个人觉得)。

其实我也做过嵌入式的开发, 在针对特定硬件的平台下尽可能将内核编小, 将始终是支持模块加载的。例如我们开发的防火墙就是做为内核的模块被加载的。使用模块支持, 你的系统能具有更好的可扩充性。还有一个原因就是自己编写的功能模块、设备驱动模块 (假设编写的质量不高) 以模块方式工作引起

#### Kernel

Panic 的机率要远远低于不支持模块全部编进内核的方式。讲了这么多, 终于可以理直气壮的选上这一功能了。

Module unloading, 不选这个功能, 加载的模块就不能卸载。没什么需要多解释的, 建议最好选上。

Forced module unloading, 这个选项能强行卸载模块, 即使内核认为这样并不安全, 也就是说你可以把正在使用中的模块卸载掉。如果你不是内核开发人员或者骨灰级的玩家, 不要选择这个选项。

#### Module versioning support

(EXPERIMENTAL), 这个功能可以让你使用其它版本的内核模块, 由于我自己写一些模块, 所以我会用到这个选

项，因为内核更新太快了，我的头文件更新根本赶不上内核的更新。还有，虽然我在 Gentoo 下开发，但实际真实环境用的却是从 kernel.org 下载的内核。虽然我选择了这个选项，不过建议你不要选择这个选项。

Source checksum for all modules，这个功能是为了防止更改了内核模块的代码但忘记更改版本号而造成版本冲突。我估计现在没有哪家公司在开发中还没使用版本控制工具，所以不需要这项了。如果你不是自己写内核模块，那就更不需要这一选项了。

Automatic kernel module loading，这个选项能让内核自动的加载部份模块，建议你最好选上。举个例子说明一下，如模块 eth1394 依赖于模块 ieee1394。如果选择了这个选项，可以直接加载模块 eth1394；如果没有选择这个选项，必需先加载模块 ieee1394，再加载模块 eth1394，否则将出错。

处理器内型及特性，

代码：

```
1. Processor type and features --->
2. Subarchitecture Type (PC-compatible)
3. ---> Processor family
(Pentium-4/Celeron(P4-based)/Pentium-4
M/Xeon) --->
4. [ ] Generic x86 support
5. [*] HPET Timer Support
6. [*] Symmetric multi-processing support
7. (2) Maximum number of CPUs (2-255)
8. [*] SMT (Hyperthreading) scheduler
support
```

```

9.      [ ] Preemptible Kernel

10.     [ ] Machine Check Exception

11.     <M> Toshiba Laptop support

12.     <M> Dell laptop support

13.     < > /dev/cpu/microcode - Intel IA32 CPU
          microcode support

14.     < > /dev/cpu/*/msr - Model-specific
          register support

15.     < > /dev/cpu/*/cpuid - CPU information
          support

16.     Firmware Drivers --->

17.     < > BIOS Enhanced Disk Drive calls
          determine boot disk (EXPERIMENTAL) High
          Memory Support (4GB) --->

18.     [ ] Allocate 3rd-level pagetables from
          highmem

19.     [ ] Math emulation

20.     [*] MTRR (Memory Type Range Register)
          support

21.     [ ] Boot from EFI support (EXPERIMENTAL)

22.     [*] Enable kernel irq balancing

23.     [ ] Use register arguments
          (EXPERIMENTAL)

```

[复制代码](#)

Subarchitecture Type, 这没什么好说的, 如果用 PC 机的话都选这个。

Processor family, 这也没什么好说的, 选择你机器对应的处理器即可。

Generic x86 support, 这一选项针对 x86 系列的 CPU 使用更多的常规优化。如果你在上面一项选的是 i386、i586

之类的才选这个。

HPET Timer Support, HPET 是替代 8254 芯片的下一代时钟处理器。这里你可以安全的选上这一选项。如果硬件不支持的话, 将仍使用 8254 时钟处理器。

Symmetric multi-processing support, 对称多处理器支持, 在单 CPU 的机器上, 不选这个选项会更快一些。由于超线程技术, 看起来是两颗 CPU, 因此要选上这个选项。

Maximum number of CPUs (2-255), 支持的最大 CPU 数。

SMT (Hyperthreading) scheduler support, 超线程支持, 如果你的 CPU 是 P4 超线程的, 应该选上这一选项。

Preemptible Kernel, 这个选项能使应用程序即使内核在高负载时也很可靠, 建议最好选上。

Machine Check Exception, 这个选项能让 CPU 检测到系统故障时通知内核, 一般我用组装的台式机选这项。本本嘛, 我感觉还是非常可靠的, 所以就不选它了。

Toshiba Laptop support, Dell laptop support, 这两项都是对本本的支持, 其实编译内核的原则应该是让内核能在特定的环境下运行, 由于我编译的内核可能公司的其它人也会使用, 所以我尽可能的不针对特定的硬件。将对特定的硬件支持编译成模块。

/dev/cpu/microcode - Intel IA32 CPU microcode support, 这个选项是让你使用不随 Linux 内核发行的 IA32 microcode, 但是你必需有 IA32 microcode 的二进制文件。

/dev/cpu/\*/msr - Model-specific register support,



这个选项能让特权 CPU 访问 x86 的 MSR 寄存器。由于超线程并不是真正的多处理器环境，所以不要选择这个。

`/dev/cpu/*/cpuid` - CPU information support, 这个选项能从 `/dev/cpu/x/cpuid` 获得 CPU 的唯一标识符。

BIOS Enhanced Disk Drive calls determine boot disk, 台式机的有些 BIOS 支持从某块特定的硬盘启动, 由于笔记本只能装一块硬盘, 所以就不选择这项了。如果你的 BIOS 不支持这个功能而你选上的话, 有可能无法启动。

High Memory Support (4GB), 4GB 的内存支持, 已经足够了。

Allocate 3rd-level pagetables from highmem, 除非你真的有几 G 的内存, 选择这个是没有意义的。

Math emulation, 估计现在没人有 386 或 486SX 的处理器了吧, 那就不要选这个。

MTRR (Memory Type Range Register) support, 这个选项必需要选上。

Boot from EFI support (EXPERIMENTAL), 由于我使用的是 GRUB, 所以选上这个也没什么用, 如果你打算使用 EFI 的功能, 你可以到 <http://elilo.sourceforge.net> 看看。

Enable kernel irq balancing, 选上这个选项能让内核进行 IRQ 均衡。


Use register arguments (EXPERIMENTAL), 使用 `-mregparm=3` 参数编译内核, 将前 3 个参数以寄存器方式进行参数调用。GCC 的版本必需大于等于 3.0。

收藏 分享 评分

回复 引用

订阅 TOP

沙发

发表于 2006-9-23 16:55 | 只看该

作者

最新主题与激烈辩论主题，5 分钟前由系统自动更新！

电源管理，

代码：



厉焯



开源将军



UID

21336

帖子

195

积分

453

现金

3368 圆

魅力

160 点

阅读权限

100

注册时间

2006-7-19

```
1.   Power management options (ACPI,
      APM)  --->
2.   [*] Power Management support
3.   [ ] Power Management Debug Support
4.   [ ] Software Suspend
      (EXPERIMENTAL)
5.   ACPI (Advanced Configuration
      and Power Interface) Support  --->
6.   APM (Advanced Power
      Management) BIOS Support  --->
7.   CPU Frequency scaling  --->
```

复制代码

Power Management support，电源管理没什么好说的，不想浪费电就选上。如果不选你可以跳过这份。

Power Management Debug Support，电源管理的调试信息支持，如果不是要调试内核有关电源管理部份，请不要选择这项。

Software Suspend

(EXPERIMENTAL)，休眠到硬盘。也就是将内存写入交换分区中，下次启动可以通过参数 `resume=/dev/swappartition`（例如：`resume=/dev/hda6`）来恢复上次机器运行的状态。这项功能对于系统引导时启动许多服务的机器来说很有用，可以节约启动时间。这项功能根据自己的需要选择吧，如果你选择这项功能，记得恢复休眠后重做交换分区。

代码：

1. ACPI (Advanced Configuration and Power Interface) Support --->
2. ☒ ACPI Support
3. ☐ Sleep States (EXPERIMENTAL)
4. ☒ AC Adapter
5. ☒ Battery
6. ☒ Button
7. ☒ Video
8. ☒ Fan
9. ☒ Processor
10. ☒ Thermal Zone
11. ☒ ASUS/Medion Laptop Extras
12. ☒ IBM ThinkPad Laptop Extras
13. ☒ Toshiba Laptop Extras
14. ☒ (0) Disable ACPI for systems before Jan 1st this year
15. ☐ Debug Statements
16. ☐ Power Management Timer Support

[复制代码](#)

ACPI Support，这是一种电源管理方式，你可以看看你的 BIOS 是否支持。如果支持的话建议你选上这项。

## Sleep States

(EXPERIMENTAL)，这项功能可以让系统进入休眠状态（不是休眠到硬盘）。休眠是指系统仍然通着电，只是进入最大幅度的省电状态；而休眠到硬盘是指系统已经断电。不过如果你不是驱动程序的电源管理部份的开发人员，建议你最好不要选择这项。相信未来 linux 下的驱动对电源支持的功能会越来越好，  
或者也搞个硬件兼容列表，到时就可以放心的使用这项功能了。

AC Adapter，检测是电源供电还是电池供电，通常只对本本有用。

Battery，通过/proc/acpi/battery 得到电池的信息，通常这也是针对笔记本的。

Button，捕获 Power、Sleep、Lid（我也不知道这是什么按钮）等按钮是否按下，并做相应的动作。

Video，集成在板上的显卡的 ACPI 支持，对有些板卡可能不起作用。

Fan，风扇的支持。这一点很明显，不选这项我的本本的风扇一直在转，选上以后风扇只是间断的转转。

Processor，当机器负荷轻时节省处理器的用电，处理器可是电脑中的第一用电大户（可能老式的 CRT 显示器和它有一比）。

Thermal Zone，这个我也不太清楚是什么，只是据说大部份的台式机和笔记本都支持，不选还可能把处理器烧掉。如果你不会让模块正常工作，还是把它编进内核吧，怪吓人的。

ASUS/Medion Laptop Extras、IBM ThinkPad Laptop Extras、Toshiba Laptop Extras，这三种本本的扩展支持。你的内核如果只是自己用，选个该选的就行了。

(0) Disable ACPI for systems before Jan 1st this year，输入四位数的年份，在该年的 1 月 1 日前不使用 ACPI 的功能。0 表示一直使用。

Debug Statements，详细的 ACPI 调试信息，不搞开发就别选。

Power Management Timer Support，我的本本支持 HPET（要是忘了是什么，再看看前面），所以不选它。要是你的机器不支持，应该把它选上。

代码：

```
1.  APM (Advanced Power Management) BIOS
    Support --->
2.  <M> APM (Advanced Power
    Management) BIOS support
3.  [ ] Ignore USER SUSPEND
4.  [*] Enable PM at boot time
5.  [ ] Make CPU idle calls when idle
6.  [ ] Enable console blanking using
    APM
7.  [ ] RTC stores time in GMT
8.  [ ] Allow interrupts during APM
    BIOS calls
9.  [*] Use real mode APM BIOS call
    to power off
```

[复制代码](#)

APM (Advanced Power Management) BIOS support，

高级电源管理的支持，一般来说笔记本应该选上，台式机可以不选。

Ignore USER SUSPEND，只有 NEC Versa M 系列的笔记本需要选择这一项。

Enable PM at boot time，启动时支持电源管理，选上这个选项能让系统自动的进行电源管理，除非在启动时死机，才不要选这项。

Make CPU Idle calls when idle，系统空闲时调用空闲指令。只有老式的 CPU 才用这项。其实调用空闲指令还是让 CPU 执行了一条指令。这个选项在内核循环中调用空闲指令。

Enable console blanking using APM，支持关闭监视器。据说这项功能对所有的笔记本都无效。如果你都按我的建议配置，系统是能自动休眠的（使用 ACPI）。你也不用担心你的显示器一直亮着的。

RTC stores time in GMT，按 Unix 的标准，硬件的时钟应该设为格林威治时间。还是那句老话，因为我还要用 Windows，所以硬件时钟设成了本地时间，当然就不要选这项了。

Allow interrupts during APM BIOS calls，允许 APM 的 BIOS 调用时中断。多数的机器不需要这项，Thinkpad 的一些新机器需要这项。如果休眠时挂机（包括睡下去就醒不来），再把这项选上。

Use real mode APM BIOS call to power off，建议最好选上此项，保证软件关机。如果你有兴趣可以试试你的机器不选这项能不能正常的软件关机（多数机器不能）。

补充一点，如果既选择了 ACPI 又选择了 APM，先加

载的将被使用。

通过这么多的例子，大家应该可以看出来在 menuconfig 中，圆括号内是参数，可以选择某一选项或者输入具体的参数。方括号只能选择 “Y” 或 “N” ，尖括号除了选择 “Y” 和 “N” 还可以选择 “M” 。“Y” 表示将该选项包括在内核中，menuconfig 中以 “\*” 表示。“N” 表示不使用此选项的功能，“M” 表示将此选项的功能编译成模块。

ACPI 是为了取代 APM 而设计的，因此尽量使用 ACPI 的功能，实在不行再加载 apm 模块。

代码：

1. [\*] CPU Frequency scaling
2. [ ] Enable CPUFreq debugging
3. < > /proc/cpufreq interface  
(deprecated)
4. Default CPUFreq governor  
(performance) --->
5. --- 'performance' governor
6. <M> 'powersave' governor
7. <M> 'userspace' governor for  
userspace frequency scaling
8. [ ] /proc/sys/cpu/ interface  
(2.4. / OLD)
9. <M> 'ondemand' cpufreq policy  
governor
10. <\*> CPU frequency table helpers
11. <M> ACPI Processor P-States driver
12. <M> AMD Mobile K6-2/K6-3 PowerNow!
13. <M> AMD Mobile Athlon/Duron

```

PowerNow!
14.  <M> AMD Opteron/Athlon64 PowerNow!
15.  <M> Cyrix MediaGX/NatSemi Geode
      Suspend Modulation
16.  <M> Intel Enhanced SpeedStep
17.  [ ] Use ACPI tables to decode valid
      frequency/voltage pairs---
18.  Built-in tables for Baniass CPUs
19.  <M> Intel Speedstep on ICH-M chipsets
      (ioport interface)
20.  <M> Intel SpeedStep on 440BX/ZX/MX
      chipsets (SMI interface)
21.  <M> Intel Pentium 4 clock modulation
22.  < > nvidia nForce2 FSB changing
23.  <M> Transmeta LongRun
24.  <M> VIA Cyrix III Longhaul--- shared
      options
25.  [ ]
      /proc/acpi/processor/./performance
      interface (deprecated)
26.  [ ] Relaxed speedstep capability
      checks

```

[复制代码](#)

CPU Frequency scaling, 这一选项允许改变 CPU 的主频, 使 CPU 在低负荷或使用电池时降低主频, 达到省电的目的。

Enable CPUfreq debugging, 是否允许调试 CPU 改变主频的功能, 如果要调试, 还需要在启动时加上参数。cpu freq.debug=

1: 变频技术的内核调试



## 2: 变频技术的驱动调试

## 4: 变频技术的调节器调试

/proc/cpufreq interface (deprecated) , 是否允许/proc/cpufreq 来调节主频, 建议使用默认的 sysfs 来调节。

Default CPUFreq governor (performance) --->, 默认的主频调节, 圆括号内的是你选择的结果, 这里表示以性能为主。

'powersave' governor, 最大限度的节约电能调节器。

'userspace' governor for userspace frequency scaling, 用户自定义调节器。

/proc/sys/cpu/ interface (2.4. / OLD), 兼容 2.4 内核的用户调节器。

'ondemand' cpufreq policy governor, 自动调节主频。

CPU frequency table helpers, 多数的 CPU 需要这一项来调节主频。

ACPI Processor P-States driver, 报告处理器的状态。

AMD Mobile K6-2/K6-3 PowerNow!, AMD 移动版 K6 处理器的变频驱动。

AMD Mobile Athlon/Duron PowerNow!, AMD 移动版毒龙、雷乌的变频驱动。

AMD Opteron/Athlon64 PowerNow!, AMD64 处理器的变频驱动。

Cyrix MediaGX/NatSemi Geode Suspend Modulation, Cyrix 处理器的变频驱动。

Intel Enhanced SpeedStep, Intel 的变频技术支持。

Use ACPI tables to decode valid frequency/voltage pairs, 使用 BIOS 中的主频 / 电压参数。

--- Built-in tables for Banias CPUs, 迅驰一代的主频 / 电压参数。

Intel Speedstep on ICH-M chipsets (ioport interface) , Intel ICH-M 南桥芯片组的支持。

Intel SpeedStep on 440BX/ZX/MX chipsets (SMI interface), Intel 440BX/ZX/MX 南桥芯片级的支持。

Intel Pentium 4 clock modulation, P4 处理器的时钟模块支持。

nVidia nForce2 FSB changing, nVidia nForce2 的支持。

Transmeta LongRun, Transmeta 处理器的支持。

VIA Cyrix III Longhaul, VIA Cyrix 处理器的支持。

/proc/acpi/processor/../performance

interface (deprecated), 从  
/proc/acpi/processor/../performance 获得 CPU  
的变频信息。

Relaxed speedstep capability checks, 不全面检测 Intel Speedstep, 有的系统虽然支持 Speedstep 技术, 却无法通过全面的检测。

总线类型,

1.    ☒ PCI support
2.            PCI access mode (Any)    --->
3.    ☐ Message Signaled Interrupts (MSI and MSI-X)
4.    ☐ Legacy /proc/pci interface
5.    ☐ PCI device name database
6.    ☒ ISA support
7.    ☒ EISA support
8.    ☒ Vesa Local Bus priming
9.    ☒ Generic PCI/EISA bridge
10.   ☒ EISA virtual root device
11.   ☐ EISA device name database
12.   ☐ MCA support
13.   < > NatSemi SCx200 support
14.           PCCARD (PCMCIA/CardBus)  
                 support    --->
15.           PCI Hotplug Support    --->

[复制代码](#)

PCI support, 没有人不知道这是什么总线类型吧, 实在不知道就去 google 查吧, 这个当然要选上。

PCI access mode (Any), 强烈建议选 Any, 系统将优先使用 MMConfig, 然后使用 BIOS, 最后使用

Direct 检测 PCI 设备。

Message Signaled Interrupts (MSI and MSI-X), 建议你不要选择这项,设备将使用默认的 IRQ 中断。如果选择这项,允许设备通过 PCI 总线写入内存堆栈产生一个中断。

Legacy /proc/pci interface, 是否使用/proc/pci 目录下的信息文件来描述 PCI 设备的信息。现在的系统多数都使用 lspci 工具来得到这样的信息。

PCI device name database, 如果你不打算使用 lspci 工具,就把这项和上面的一项选上。lspci 和 hotplug 都不需要内核中的设备信息库了。

ISA support, 是否使用工业总线。如果你没有老式的 ISA 设备,可以不选这项。现在基本上都没有 ISA 的设备了。不过需要注意的是如果你做嵌入式系统的开发,一些 PC104 的总线可能会桥接到 EISA 或者是。VESA 总线上。

EISA support, 扩展工业总线。

Vesa Local Bus priming, VESA 总线,也是扩展工业总线的一种。我的老 486DX66 的机器上的显卡就是这种总线,块板上大概还有 2 个 EISA 插槽各 3 个 ISA 插槽。

PCI/EISA bridge, PCI、EISA 两种总线的桥。

EISA virtual root device, EISA 总线的虚拟根设备。

EISA device name database, 内核中的 EISA 设备信息库。

MCA support，微通道总线。IBM 的台式机和笔记本上可能会有这种总线，包括它的 p 系列、e 系列、z 系列机器上都用到了这种总线。

NatSemi SCx200 support，这个我不知道是什么东西，看帮助是松下的一种半导体处理器的驱动。

总之，只要你的主板没有 ISA 插槽，而且你也不是搞嵌入式开发，工业自动化控制的。不要选“ISA support”就是了，如今的 ISA 设备在 x86 体系上基本是见不到了。不过自己制板的话，还是 ISA 的板子最好做。

[Linux 通用技术](#) | [Linux 发行版技术](#)  
[34 省 Linux 人](#) | [大学生:Linux 校友](#)  
[回复](#) [引用](#)

TOP

厉焯



开源将军



UID

21336

帖子

195

积分

453

现金

3368 圆

魅力

160 点

藤椅

 发表于 2006-9-23 16:58 | 只看该作者

2.6 内核的配置与编译(10)

1. PCCARD (PCMCIA/CardBus) support --->
2. <M> PCCard (PCMCIA/CardBus) support
3. [ ] Enable PCCARD debugging
4. [ ] Enable obsolete PCCARD code
5. <M> 16-bit PCMCIA support
6. [\*] 32-bit CardBus support
7. --- PC-card bridges
8. <M> CardBus yenta-compatible bridge support
9. <M> Cirrus PD6729 compatible bridge support
10. <M> i82092 compatible bridge support
11. <M> i82365 compatible bridge support
12. <M> Databook TCIC host bridge support

[复制代码](#)

阅读权限

100

注册时间

2006-7-19

PCCard (PCMCIA/CardBus) support, 一般只有笔记本电脑上才会有 PCMCIA 插槽, 如果你是台式机的话, 可以不选这一项, 然后跳过这一部份。

Enable PCCARD debugging, 通常不需要选择调试 PCMCIA 设备, 除非你是设备驱动的开发人员。

Enable obsolete PCCARD code, 老式的 PCMCIA 设备只持。现在很少有这样的设备了, 除非你买这样的设备时带了张 Linux 的驱动光盘才需要选上。而且估计你也只能在二手市场上买到这样的设备。

16-bit PCMCIA support, 16 位的 PCMCIA 总线支持。

32-bit CardBus support, 32 位的 PCMCIA 总线支持, 通常也叫 PCMCIA II 总线。

下面的是不同产家的 PCMCIA 芯片的驱动支持, 如果你知道你的本本用的是什么芯片组的话, 可以只选它而不选其它的。要是你不知道可以象我一样的全部选上, 然后用 modprobe 一种一种的试。最后我终于知道我的 HP zv5028 的本本用的是 yenta-compatible 的芯片组了。

- ```
1.    --- PC-card bridges
2.    <M> CardBus yenta-compatible bridge support
3.    <M> Cirrus PD6729 compatible bridge support
4.    <M> i82092 compatible bridge support
5.    <M> i82365 compatible bridge support
6.    <M> Databook TCIC host bridge support
```

[复制代码](#)

## 2.6 内核的配置与编译(11)

- ```
1.    PCI Hotplug Support --->
2.    <M> Support for PCI Hotplug (EXPERIMENTAL)
3.    < >   Fake PCI Hotplug driver (NEW)
4.    < >   Compaq PCI Hotplug driver (NEW)
5.    < >   IBM PCI Hotplug driver (NEW)
6.    < >   ACPI PCI Hotplug driver (NEW)
```

- 7.        ☐ CompactPCI Hotplug driver (NEW)
- 8.        ☐ PCI Express Hotplug driver (NEW)
- 9.        ☐ SHPC PCI Hotplug driver (NEW)

[复制代码](#)

Support for PCI Hotplug (EXPERIMENTAL)，一般来讲只有服务器上会有热插拔的设备，如果你使用的是台式机，你可以不选择此项并跳过这一部份。（其实我也没有选这一项，只是为了讲解的方便而选上的。）

Fake PCI Hotplug driver (NEW)，选上这一选项能让你的机器模拟 PCI 热插拔。注意，它并不是真正意义上的热插拔，绝对不允许带电插拔设备除非你的主板上集成了 PCI 热插拔芯片并且你的 PCI 设备本身支持热插拔。

Compaq PCI Hotplug driver (NEW)，Compaq 服务器上的热插拔芯片组的支持。

IBM PCI Hotplug driver (NEW)，IBM 服务器上的热插拔芯片组的支持。

ACPI PCI Hotplug driver (NEW)，PCI 热插拔设备是否支持 ACPI 电源管理（一般来说都是支持的）。

CompactPCI Hotplug driver (NEW)，精简 PCI 总线的热插拔设备的支持，通常在嵌入式系统中会用到精简 PCI 总线。

PCI Express Hotplug driver (NEW)，PCI 加速总线的热插拔设备的支持。现在 PCI Express 总线的显卡挺火的。但用于服务器上的 PCI 加速总线的设备我还没见过。（我是井底之蛙）

SHPC PCI Hotplug driver (NEW)，SHPC 热插拔控制芯片的支持。

## 2.6 内核的配置与编译(12)

可执行文件格式，

- 1.        ☐ Kernel support for ELF binaries
- 2.        ☐ Kernel support for a.out and ECOFF binaries

### 3. `<*>` Kernel support for MISC binaries

[复制代码](#)

Kernel support for ELF binaries, ELF 是开放平台下最常用的二进制文件, 它支持不同的硬件平台。

Kernel support for a.out and ECOFF binaries, 这是早期 UNIX 系统的可执行文件格式, 目前已经被 ELF 格式取代。

Kernel support for MISC binaries, 此选项允许插入二进制的封装层到内核中, 当使用 Java、.NET、Python、Lisp 等语言编写的程序时非常有用。

接下来应该讲硬件设备部份, 但考虑到硬件部份是针对具体硬件的, 大数 Linux 玩家都是硬件的 DIYer。因此对这一部份应该很熟悉。硬件设备部份将放到最后讲, 下一篇将讲文件系统部份。

文件系统,

1. `<*>` Second extended fs support
2. `[*]` Ext2 extended attributes
3. `[*]` Ext2 POSIX Access Control Lists
4. `[*]` Ext2 Security Labels
5. `<*>` Ext3 journalling file system support
6. `[*]` Ext3 extended attributes
7. `[*]` Ext3 POSIX Access Control Lists
8. `[*]` Ext3 Security Labels
9. `[ ]` JBD (ext3) debugging support
10. `<*>` Reiserfs support
11. `[ ]` Enable reiserfs debug mode
12. `[ ]` stats in /proc/fs/reiserfs
13. `[*]` ReiserFS extended attributes
14. `[*]` ReiserFS POSIX Access Control Lists
15. `[*]` ReiserFS Security Labels
16. JFS filesystem support



```

17.  [*]  JFS POSIX Access Control Lists
      18.  [ ]  JFS debugging
      19.  [ ]  JFS statistics
      20.  XFS filesystem support
21.  [*]  Realtime support (EXPERIMENTAL)
      22.  [*]  Quota support
      23.  [*]  Security Label support
      24.  [*]  POSIX ACL support
      25.  < > Minix fs support
      26.  < > ROM file system support
      27.  [*] Quota support
      28.  < > Old quota format support
      29.  Quota format v2 support
      30.  [*] Dnotify support
      31.  < > Kernel automounter support
32.  < > Kernel automounter version 4 support (also supports v3)
      33.  CD-ROM/DVD Filesystems --->
      34.  DOS/FAT/NT Filesystems --->
      35.  Pseudo filesystems --->
      36.  Miscellaneous filesystems --->
      37.  Network File Systems --->
      38.  Partition Types --->
      39.  Native Language Support --->

```

[复制代码](#)

有人说在编译内核时应该将/boot 分区和/分区的文件系统编译进内核，其它的可以编译成模块。对，但不确切。让我们来一起了解一下 linux 系统的启动顺序。在内核被加载后，如果 initrd 参数传入了内核，内核会去调用指定的文件。当然，initrd 和 System.map 通常都是/boot 下。但是同样可以用 initrd=(hd1,2)/initrd.img 这样的方式指定。内核启动完成后将调用/sbin/init，（如果是链接要保证目标文件能被内核加载）。不同的系统的启动脚本可能不太一样，这里不详细介绍。启动脚本向内核加载模块时可能用/sbin/modprobe 或

/sbin/insmod，由此看来/sbin 的文件系统是要内核支持的。编译的内核模块一般在/lib/modules/的版本目录下，所以/lib/modules 的文件系统是要内核支持的。一旦其它文件系统的模块能加载，系统就能向正常的访问内核中的文件系统一样访问模块支持的文件系统了。由于启动脚本、fstab 自动加载等文件一般在/etc 目录下，因此/etc 的文件系统是要内核支持的。

这里概要的介绍了保证系统正常启动的几个关键点，可能我反而把它讲复杂了。如果你能理解上面的这段话，你应该能清楚的知道哪些文件系统是要编译进内核的，哪些是可以编译成模块的。如果你

不太理解上面的这段话，下篇贴子我将详细介绍每个选项及几种常用的文件系统。当然这里面包含了我的偏见，如果你觉得我的说法不准确，有误导看官的地方，请

一定指出来。我在此先表示多谢了。

## 2.6 内核的配置与编译(14)

1. <\*> Second extended fs support
2. [\*] Ext2 extended attributes
3. [\*] Ext2 POSIX Access Control Lists
4. [\*] Ext2 Security Labels
5. <\*> Ext3 journalling file system support
6. [\*] Ext3 extended attributes
7. [\*] Ext3 POSIX Access Control Lists
8. [\*] Ext3 Security Labels
9. [ ] JBD (ext3) debugging support
10. <\*> Reiserfs support
11. [ ] Enable reiserfs debug mode
12. [ ] Stats in /proc/fs/reiserfs
13. [\*] ReiserFS extended attributes
14. [\*] ReiserFS POSIX Access Control Lists
15. [\*] ReiserFS Security Labels
16. JFS filesystem support
17. [\*] JFS POSIX Access Control Lists

```

18. [ ] JFS debugging
19. [ ] JFS statistics
20. XFS filesystem support
21. [*] Realtime support (EXPERIMENTAL)
22. [*] Quota support
23. [*] Security Label support
24. [*] POSIX ACL support
25. < > Minix fs support
26. < > ROM file system support
27. [*] Quota support
28. < > old quota format support
29. Quota format v2 support
30. [*] Dnotify support
31. < > Kernel automounter support
32. < > Kernel automounter version 4 support (also supports v3)
33. CD-ROM/DVD Filesystems --->
34. DOS/FAT/NT Filesystems --->
35. Pseudo filesystems --->
36. Miscellaneous filesystems --->
37. Network File Systems --->
38. Partition Types --->
39. Native Language Support --->

```

[复制代码](#)

Second extended fs support, 标准的 Linux 文件系统, 建议将这种文件系统编译进内核。

Ext2 extended attributes, Ext2 文件系统的结点名称、属性的扩展支持。

Ext2 POSIX Access Control Lists, POSIX 系统的访问权限列表支持。也就是 Owner/Group/Others 的 Read/Write/Execute 权限。请参考 Unix 标准文件系统权限。

Ext2 Security Labels, 扩展的安全标签, 例如 SELinux 之类的安全系统会使用到这样的扩展安全属性。

Ext3 journalling file system support, 如果你熟悉 Redhat Linux, 你一定会习惯 Ext3 文件系统。

Ext3 extended attributes, Ext3 文件系统的结点名称、属性的扩展支持。

Ext3 POSIX Access Control Lists, POSIX 系统的访问权限列表支持。

Ext3 Security Labels, 扩展的安全标签支持。

JBD (ext3) debugging support, Ext3 的调试。除非你是文件系统的开发者, 否则不要选上这一项。

Reiserfs support, 如果你熟悉 Suse Linux, 你一定会习惯 Reiserfs 文件系统。

Enable reiserfs debug mode, Reiserfs 的调试。除非你是文件系统的开发者, 否则不要选上这一项。

Stats in /proc/fs/reiserfs, 在 /proc/fs/reiserfs 文件中显示 Reiserfs 文件系统的状态。一般来说不需要选择这一项。

ReiserFS extended attributes, Reiserfs, 文件系统的结点名称、属性的扩展支持。

ReiserFS POSIX Access Control Lists, POSIX 系统的访问权限列表支持。

ReiserFS Security Labels, 扩展的安全标签支持。

JFS filesystem support, JFS 是 IBM 公司设计用于 AIX 系统上的文件系统。后来这一文件系统也能应用于 Linux 系统。

JFS POSIX Access Control Lists, POSIX 系统的访问权限列表支持。

JFS debugging, JFS 的调试。除非你是文件系统的开发者, 否则不要选上这一项。

JFS statistics, 在`/proc/fs/jfs`文件中显示 Reiserfs 文件系统的状态。一般来说不需要选择这一项。

XFS filesystem support, XFS 是 SGI 公司为其图形工作站设计的一种文件系统, 后来这一文件系统也能应用于 Linux 系统。

Realtime support (EXPERIMENTAL), 实时卷的支持, 能大幅提高大文件的读写速度。不过并不太安全, 建议暂时不要选择这一选项。

Quota support, XFS 文件系统的配额支持。

Security Label support, 扩展的安全标签支持。

POSIX ACL support, POSIX 系统的访问权限列表支持。

Minix fs support, Minix 可能是最早的 Linux 系统所使用的文件系统。后来被 Ext2 文件系统所取代。

ROM file system support, 内存文件系统的支持。除非你是嵌入式系统的开发者, 明确知道你要干什么, 否则不要选这一项。

Quota support, 配额支持。也就是说限制某个用户或者某组用户的磁盘占用空间。

Old quota format support, 旧版本的配额支持。

Quota format v2 support, 新版本(第二版)的配额支持。

Dnotify support, 基于目录的文件变化的通知机制。

Kernel automounter support, 内核自动加载远程文件系统的支持。

Kernel automounter version 4 support (also supports v3), 新的(第四版)的内核自动加载远程文件系统的支持, 也支持第三版。