# 04-4 The Kernel

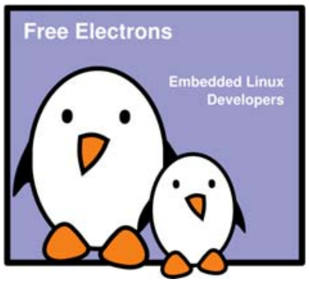
## It all started with…

```
From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)
Newsgroups: comp.os.minix
Subject: What would you like to see most in minix?
Summary: small poll for my new operating system
Message-ID: <1991Aug25.205708.9541@klaava.Helsinki.FI>
Date: 25 Aug 91 20:57:08 GMT
Organization: University of Helsinki

Hello everybody out there using minix –

I'm doing a (free) operating system (just a hobby, won't be big and professional
like gnu) for 386(486) AT clones. This has been brewing since april, and is
starting to get ready. I'd like any feedback on things people like/dislike in
minix, as my OS resembles it somewhat(same physical layout of the file-system (due
to practical reasons)among other things).

I've currently ported bash(1.08) and gcc(1.40), and things seem to work. This implies
that I'll get something practical within a few months, and I’d like to know
what features most people would want. Any suggestions are welcome, but I won’t
promise I’ll implement them :-)

        Linus (torvalds@kruuna.helsinki.fi)
```

## Free Electrons

# Linux kernel introduction

Michael Opdenacker
Thomas Petazzoni
**Free Electrons**



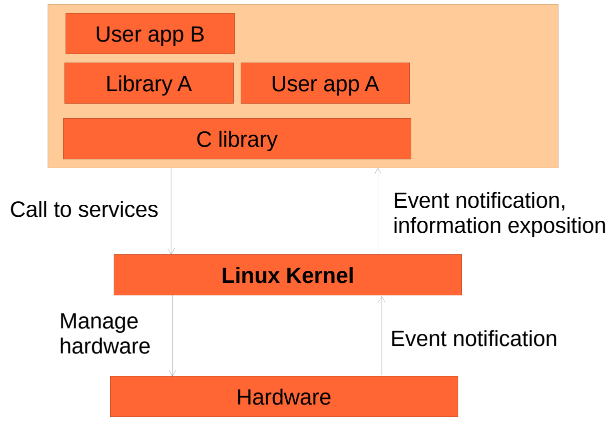
Latest update: 10/1/2013,
Document sources, updates and translations:
http://free-electrons.com/docs/kernel-intro
Corrections, suggestions, contributions and translations are welcome!

# Embedded Linux driver development

Kernel overview

Linux features

## Linux kernel in the system



## History

- The Linux kernel is one component of a system, which also requires libraries and applications to provide features to end users
- The Linux kernel was created as a hobby in 1991 by a Finnish student, Linus Torvalds
- Linux quickly started to be used as the kernel for free software operating systems
- Linus Torvalds has been able to create a large and dynamic developer and user community around Linux
- Nowadays, hundreds of people contribute to each kernel release, individuals or companies big and small

## Linux kernel key features

- Portability and hardware support. Runs on most architectures.
- Scalability
  Can run on super computers as well as on tiny devices
  (4 MB of RAM is enough).
- Compliance to standards and interoperability.
- Exhaustive networking support.
- Security
  It can't hide its flaws. Its code is reviewed by many experts.
- Stability and reliability.
- Modularity
  Can include only what a system needs even at run time.
- Easy to program
  You can learn from existing code. Many useful resources on the net.

## Supported hardware architectures

2.6.31 status

What's the current version?

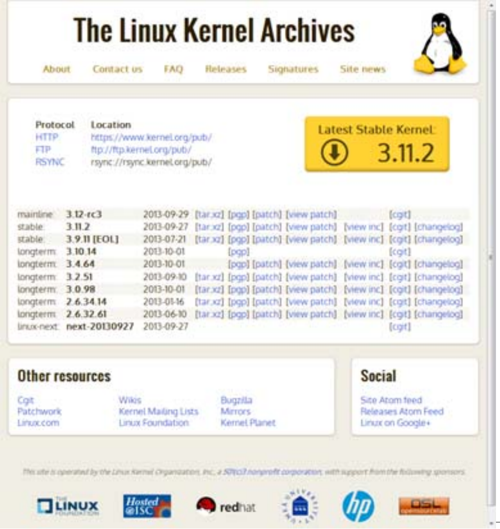
3.11.2

- See the .../arch/ directory in the kernel sources
- Minimum: 32 bit processors, with or without MMU, and gcc support
- 32 bit architectures (.../arch/ subdirectories)
  **arm**, avr32, blackfin, cris, frv, h8300, m32r, m68k, m68knommu, microblaze, mips, mn10300, parisc, s390, sparc, um, xtensa
- 64 bit architectures:
  alpha, ia64, sparc64

How did I find it?

kernel.org

- 32/64 bit architectures
  powerpc, x86, sh
- Find details in kernel sources: .../arch/<arch>/Kconfig or .../Documentation/<arch>/



# The Linux Kernel Archives

| Protocol | Location |
|---|---|
| HTTP | https://www.kernel.org/pub/ |
| FTP | ftp://ftp.kernel.org/pub/ |
| RSYNC | rsync://rsync.kernel.org/pub/ |

Latest Stable Kernel:
3.11.2

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| mainline: | 3.12-rc3 | 2013-09-29 | [tar.xz] | [pgp] | [patch] | [view patch] | | [cgit] | |
| stable: | 3.11.2 | 2013-09-27 | [tar.xz] | [pgp] | [patch] | [view patch] | [view inc] | [cgit] | [changelog] |
| stable: | 3.9.11 [EOL] | 2013-07-21 | [tar.xz] | [pgp] | [patch] | [view patch] | [view inc] | [cgit] | [changelog] |
| longterm: | 3.10.14 | 2013-10-01 | | [pgp] | | | | [cgit] | |
| longterm: | 3.4.64 | 2013-10-01 | | [pgp] | [patch] | [view patch] | [view inc] | [cgit] | [changelog] |
| longterm: | 3.2.51 | 2013-09-10 | [tar.xz] | [pgp] | [patch] | [view patch] | [view inc] | [cgit] | [changelog] |
| longterm: | 3.0.98 | 2013-10-01 | [tar.xz] | [pgp] | [patch] | [view patch] | [view inc] | [cgit] | [changelog] |
| longterm: | 2.6.34.14 | 2013-01-16 | [tar.xz] | [pgp] | [patch] | [view patch] | [view inc] | [cgit] | [changelog] |
| longterm: | 2.6.32.61 | 2013-06-10 | [tar.xz] | [pgp] | [patch] | [view patch] | [view inc] | [cgit] | [changelog] |
| linux-next: | next-20130927 | 2013-09-27 | | | | | | [cgit] | |

### Other resources

Cgit
Patchwork
Linux.com
Wikis
Kernel Mailing Lists
Linux Foundation
Bugzilla
Mirrors
Kernel Planet

### Social

Site Atom feed
Releases Atom Feed
Linux on Google+

This site is operated by the Linux Kernel Organization, Inc., a 501(c)3 nonprofit corporation, with support from the following sponsors.

## System calls

- The main interface between the kernel and userspace is the set of system calls
- About ~300 system calls that provides the main kernel services
- This interface is stable over time: only new system calls can be added by the kernel developers
- This system call interface is wrapped by the C library, and userspace applications usually never make a system call directly but rather use the corresponding C library function

## Virtual filesystems

- Linux makes system and kernel information available in user-space through virtual filesystems (virtual files not existing on any real storage). No need to know kernel programming to access such information!
- Mounting /proc:
  sudo mount -t proc none /proc
- Mounting /sys:
  sudo mount -t sysfs none /sys

Filesystem type

Raw device or filesystem image
In the case of virtual filesystems, any string is fine

Mount point

## /proc details

A few examples:

- /proc/cpuinfo: processor information
- /proc/meminfo: memory status
- /proc/version: kernel version and build information
- /proc/cmdline: kernel command line
- /proc/<pid>/environ: calling environment
- /proc/<pid>/cmdline: process command line

Lots of details about the /proc interface are available in Documentation/filesystems/proc.txt (some 1700 lines) in the kernel sources.

## ... and many more! See by yourself!

```
beagle$ ls -F /proc
1/    16/   36/   45/   75/        cpuinfo       kmsg          slabinfo
10/   17/   38/   46/   76/        crypto        kpagecount    softirqs
101/  18/   39/   5/    79/        device-tree/  kpageflags    stat
11/   19/   40/   53/   8/         devices       loadavg       swaps
12/   2/    41/   530/  80/        diskstats     locks         sys/
127/  20/   412/  531/  81/        dri/          meminfo       sysrq-trigger
129/  21/   418/  533/  87/        driver/       misc          sysvipc/
13/   24/   42/   563/  88/        execdomains   modules       timer_list
138/  243/  429/  564/  9/         fb            mounts@       timer_stats
139/  244/  430/  565/  asound/    filesystems   mtd           tty/
14/   245/  437/  567/  buddyinfo  fs/           net@          uptime
140/  261/  440/  57/   bus/       interrupts    pagetypeinfo  version
142/  268/  442/  6/    cgroups    iomem         partitions    vmallocinfo
144/  27/   443/  69/   cmdline    ioports       sched_debug   vmstat
145/  3/    445/  7/    config.gz  irq/          schedstat     zoneinfo
151/  320/  447/  73/   consoles   kallsyms      scsi/
152/  345/  449/  74/   cpu/       key-users     self@
```

## Embedded Linux usage

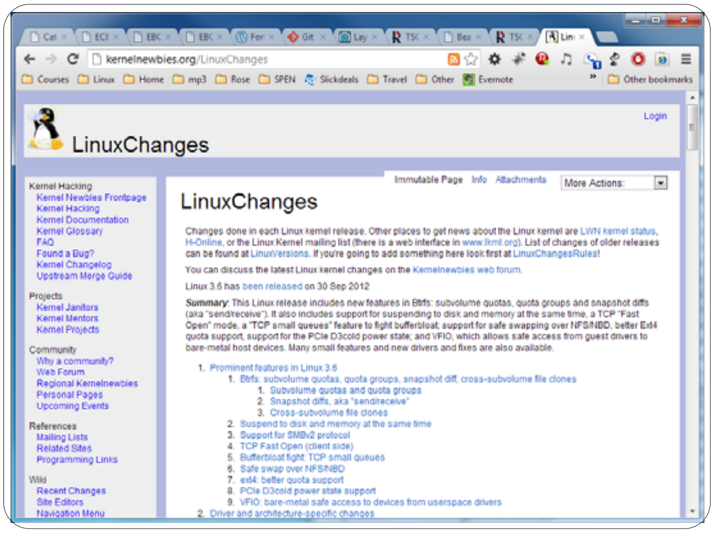
Kernel overview

Linux versioning scheme and development process

## What's new in each Linux release?

```
commit 3c92c2ba33cd7d666c5f83cc32aa590e794e91b0
Author: Andi Kleen <ak@suse.de>
Date:   Tue Oct 11 01:28:33 2005 +0200

    [PATCH] i386: Don't discard upper 32bits of HWCR on K8

    Need to use long long, not long when RMWing a MSR. I think
    it's harmless right now, but still should be better fixed
    if AMD adds any bits in the upper 32bit of HWCR.

    Bug was introduced with the TLB flush filter fix for i386

    Signed-off-by: Andi Kleen <ak@suse.de>
    Signed-off-by: Linus Torvalds <torvalds@osdl.org>
...
```

- The official list of changes for each Linux release is just a huge list of individual patches!
- Very difficult to find out the key changes and to get the global picture out of individual changes.
- Fortunately, a summary of key changes with enough details is available on http://wiki.kernelnewbies.org/LinuxChanges



LinuxChanges

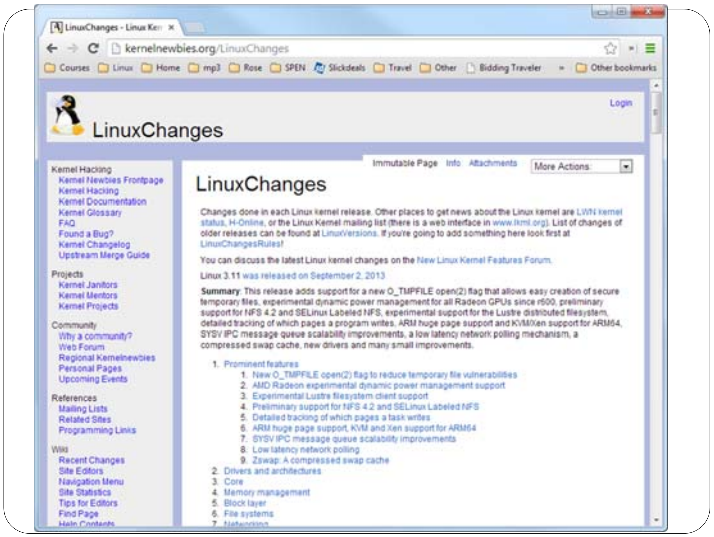
Changes done in each Linux kernel release. Other places to get news about the Linux kernel are LWN kernel status, H-Online, or the Linux Kernel mailing list (there is a web interface in www.lkml.org). List of changes of older releases can be found at LinuxVersions. If you're going to add something here look first at LinuxChangesRules!

You can discuss the latest Linux kernel changes on the Kernelnewbies web forum.

Linux 3.6 has been released on 30 Sep 2012

**Summary**: This Linux release includes new features in Btrfs: subvolume quotas, quota groups and snapshot diffs (aka "send/receive"). It also includes support for suspending to disk and memory at the same time, a TCP "Fast Open" mode, a "TCP small queues" feature to fight bufferbloat; support for safe swapping over NFS/NBD, better Ext4 quota support, support for the PCIe D3cold power state; and VFIO, which allows safe access from guest drivers to bare-metal host devices. Many small features and new drivers and fixes are also available.

1. Prominent features in Linux 3.6
   1. Btrfs: subvolume quotas, quota groups, snapshot diff, cross-subvolume file clones
      1. Subvolume quotas and quota groups
      2. Snapshot diffs, aka "send/receive"
      3. Cross-subvolume file clones
   2. Suspend to disk and memory at the same time
   3. Support for SMBv2 protocol
   4. TCP Fast Open (client side)
   5. Bufferbloat fight: TCP small queues
   6. Safe swap over NFS/NBD
   7. ext4: better quota support
   8. PCIe D3cold power state support
   9. VFIO: bare-metal safe access to devices from userspace drivers
2. Driver and architecture-specific changes



LinuxChanges

Changes done in each Linux kernel release. Other places to get news about the Linux kernel are LWN kernel status, H-Online, or the Linux Kernel mailing list (there is a web interface in www.lkml.org). List of changes of older releases can be found at LinuxVersions. If you're going to add something here look first at LinuxChangesRules!

You can discuss the latest Linux kernel changes on the New Linux Kernel Features Forum.

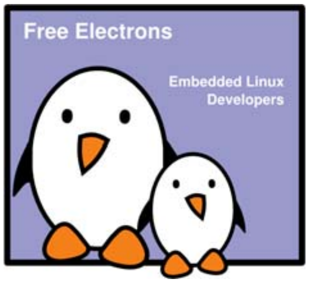
Linux 3.11 was released on September 2, 2013

**Summary**: This release adds support for a new O_TMPFILE open(2) flag that allows easy creation of secure temporary files, experimental dynamic power management for all Radeon GPUs since r600, preliminary support for NFS 4.2 and SELinux Labeled NFS, experimental support for the Lustre distributed filesystem, detailed tracking of which pages a program writes, ARM huge page support and KVM/Xen support for ARM64, SYSV IPC message queue scalability improvements, a low latency network polling mechanism, a compressed swap cache, new drivers and many small improvements.

1. Prominent features
   1. New O_TMPFILE open(2) flag to reduce temporary file vulnerabilities
   2. AMD Radeon experimental dynamic power management support
   3. Experimental Lustre filesystem client support
   4. Preliminary support for NFS 4.2 and SELinux Labeled NFS
   5. Detailed tracking of which pages a task writes
   6. ARM huge page support, KVM and Xen support for ARM64
   7. SYSV IPC message queue scalability improvements
   8. Low latency network polling
   9. Zswap: A compressed swap cache
2. Drivers and architectures
3. Core
4. Memory management
5. Block layer
6. File systems
7. Networking

## Embedded Linux kernel usage

# Embedded Linux kernel usage



Michael Opdenacker
Thomas Petazzoni
**Free Electrons**

Latest update: 10/1/2013,
Document sources, updates and translations:
http://free-electrons.com/docs/kernel-usage
Corrections, suggestions, contributions and translations are welcome!

# Embedded Linux usage

Compiling and booting Linux

Linux kernel sources

## Location of kernel sources

- The official version of the Linux kernel, as released by Linus Torvalds is available at http://www.kernel.org
  - This version follows the well-defined development model of the kernel
  - *However, it may not contain the latest development from a specific area,* due to the organization of the development model and because features in development might not be ready for mainline inclusion

Like omap

- Many kernel sub-communities maintain their own kernel, with usually newer but less stable features
  - Architecture communities (ARM, MIPS, PowerPC, etc.), device drivers communities (I2C, SPI, USB, PCI, network, etc.), other communities (real-time, etc.)
  - They generally don't release official versions, only development trees are available

## Linux kernel size (1)

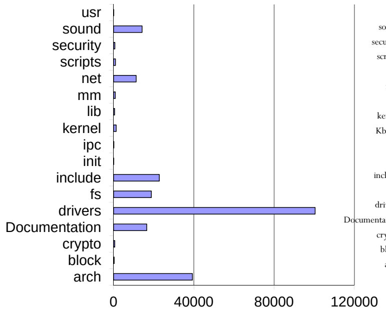
- Linux 2.6.31 sources:
  Raw size: 350 MB (30,900 files, approx 12,000,000 lines)
  gzip compressed tar archive: 75 MB
  bzip2 compressed tar archive: 59 MB (better)
  lzma compressed tar archive: 49 MB (best)
- Minimum Linux 2.6.29 compiled kernel size with CONFIG_EMBEDDED,
  for a kernel that boots a QEMU PC (IDE hard drive, ext2 filesystem, ELF executable support):
  532 KB (compressed), 1325 KB (raw)
- Why are these sources so big?
  Because they include thousands of device drivers, many network protocols, support many architectures and filesystems...
- The Linux core (scheduler, memory management...) is pretty small!

## Linux kernel size (1)

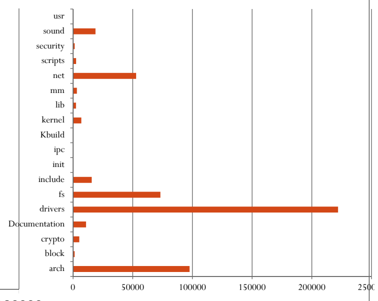
- Linux 2.6.31 sources:
  Raw size: 350 MB (30,900 files, approx 12,000,000 lines)
  gzip compressed tar archive: 75 MB
  bzip2 compressed tar archive: 59 MB (better)
  lzma compressed tar archive: 49 MB (best)
- Linux 2.6.32 sources:
- 1.3G
- Linux 3.0.9 sources:
- 1.6G
- Linux 3.2.18 sources:
- 721M, 48K files

## Linux kernel size (2)

Size of Linux source directories (KB)



Linux 2.6.17
Measured with:
du -s --apparent-size



Linux 2.6.29-r46
Measured with:
du -s --apparent-size

## Getting Linux sources

- Full tarballs
  - Contain the complete kernel sources
  - Long to download and uncompress, but must be done at least once
  - Example:
    https://www.kernel.org/pub/linux/kernel/v3.x/linux-3.11.2.tar.xz

## Getting Linux sources

- Incremental patches between versions
  - It assumes you already have a base version and you apply the correct patches in the right order
  - Quick to download and apply
  - Examples
    http://kernel.org/pub/linux/kernel/v2.6/patch-2.6.14.bz2 (2.6.13 to 2.6.14)
    http://kernel.org/pub/linux/kernel/v2.6/patch-2.6.14.7.bz2 (2.6.14 to 2.6.14.7)
- All previous kernel versions are available in http://kernel.org/pub/linux/kernel/

## Getting Linux sources

- `git clone git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux-2.6.git linux-2.6`

## Getting Linux sources

```
host$ git clone git://github.com/RobertCNelson/linux-dev.git
```

```
host$ cd linux-dev.git
```

```
host$ git checkout origin/am33x-v3.8 -b am33x-v3.8
```

```
host$ time git clone git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable.git (21 minutes)
```

## Top-Level Source Directory

```
host$ cd BeagleBoard/linux-dev/KERNEL/
host$ ls –F
arch/           Kbuild           REPORTING-BUGS
block/          Kconfig          samples/
COPYING         kernel/          scripts/
CREDITS         lib/             security/
crypto/         MAINTAINERS      sound/
Documentation/  Makefile         System.map
drivers/        mm/              tools/
firmware/       modules.builtin  usr/
fs/             modules.order    virt/
include/        Module.symvers   vmlinux*
init/           net/             vmlinux.o
ipc/            README
```

## Using the patch command

The patch command applies changes to files in the current directory:

- Making changes to existing files
- Creating or deleting files and directories

patch usage examples:

- patch -p<n> < diff_file
- cat diff_file | patch -p<n>
- bzcat diff_file.bz2 | patch -p<n>
- zcat diff_file.gz | patch -p<n>

n: number of directory levels to skip in the file paths

You can reverse a patch with the -R option

You can test a patch with the --dry-run option

## Embedded Linux usage

Compiling and booting Linux
Kernel configuration

## Kernel configuration

Defines what features to include in the kernel:

- Stored in the `.config` file at the root of kernel sources.
  - Simple text file
- Most useful commands to create this config file:
  `make [xconfig|gconfig|menuconfig|oldconfig]`
- To modify a kernel in a GNU/Linux distribution:
  the configuration files are usually released in /boot/, together with kernel images: /boot/config-2.6.17-11-generic

```
beagle$ ls –F /boot
uEnv.txt*  uImage@  uImage-3.2.25
```

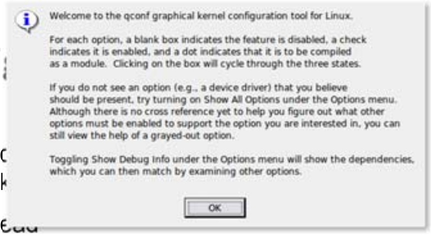
## make xconfi

`make xconfig`

- The most commo
  to configure the k
- Make sure you read
  help -> introduction: useful options!
- File browser: easier to load configuration files
- New search interface to look for parameters
- Required Debian / Ubuntu packages:

```
host$ sudo apt-get update
host$ sudo apt-get install libqt4-dev
```

libqt3-mt-dev, g++



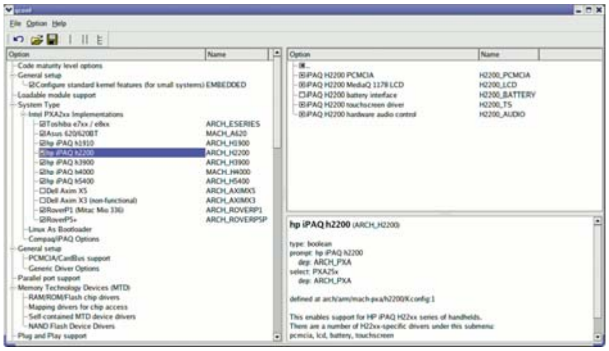
Welcome to the qconf graphical kernel configuration tool for Linux.

For each option, a blank box indicates the feature is disabled, a check indicates it is enabled, and a dot indicates that it is to be compiled as a module. Clicking on the box will cycle through the three states.

If you do not see an option (e.g., a device driver) that you believe should be present, try turning on Show All Options under the Options menu. Although there is no cross reference yet to help you figure out what other options must be enabled to support the option you are interested in, you can still view the help of a grayed-out option.

Toggling Show Debug Info under the Options menu will show the dependencies, which you can then match by examining other options.

## make xconfig screenshot

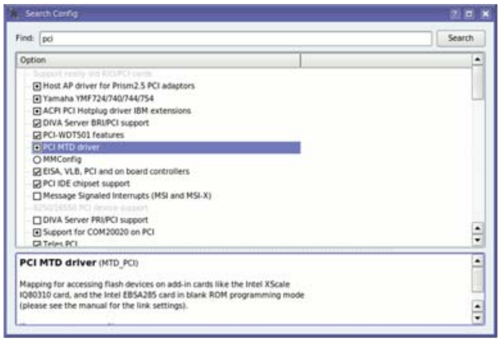


```
Option                                                       Name
-Code maturity level options
-General setup
  └☑Configure standard kernel features (for small systems)  EMBEDDED
-Loadable module support
-System Type
  -Intel PXA2xx Implementations
    -☑Toshiba e7xx / e8xx                                   ARCH_ESERIES
    -☑Asus 620/620BT                                        MACH_A620
    -☑hp iPAQ h1910                                         ARCH_H1900
    -☑hp iPAQ h2200                                         ARCH_H2200
    -☑hp iPAQ h3900                                         ARCH_H3900
    -☑hp iPAQ h4000                                         MACH_H4000
    -☑hp iPAQ h5400                                         ARCH_H5400
    -☐Dell Axim X5                                          ARCH_AXIMX5
    -☐Dell Axim X3 (non-functional)                         ARCH_AXIMX3
    -☑RoverP1 (Mitac Mio 336)                               ARCH_ROVERP1
    -☑RoverP5+                                              ARCH_ROVERP5P
  -Linux As Bootloader
  -Compaq/iPAQ Options
-General setup
  -PCMCIA/CardBus support
  -Generic Driver Options
-Parallel port support
-Memory Technology Devices (MTD)
  -RAM/ROM/Flash chip drivers
  -Mapping drivers for chip access
  -Self-contained MTD device drivers
  -NAND Flash Device Drivers
-Plug and Play support
```

```
Option                                       Name
-☒...
-☒iPAQ H2200 PCMCIA                          H2200_PCMCIA
-☒iPAQ H2200 MediaQ 1178 LCD                 H2200_LCD
-☐iPAQ H2200 battery interface               H2200_BATTERY
-☒iPAQ H2200 touchscreen driver              H2200_TS
-☒iPAQ H2200 hardware audio control          H2200_AUDIO
```

```
hp iPAQ h2200 (ARCH_H2200)

type: boolean
prompt: hp iPAQ h2200
   dep: ARCH_PXA
select: PXA25x
   dep: ARCH_PXA

defined at arch/arm/mach-pxa/h2200/Kconfig:1

This enables support for HP iPAQ H22xx series of handhelds.
There are a number of H22xx-specific drivers under this submenu:
pcmcia, lcd, battery, touchscreen
```

## make xconfig search interface



Find: pci

```
Option
 Support really old RIO/PCI cards
 ☒Host AP driver for Prism2.5 PCI adaptors
 ☒Yamaha YMF724/740/744/754
 ☒ACPI PCI Hotplug driver IBM extensions
 ☑DIVA Server BRI/PCI support
 ☑PCI-WDT501 features
 ☒PCI MTD driver
 ○MMConfig
 ☑EISA, VLB, PCI and on board controllers
 ☑PCI IDE chipset support
 ☐Message Signaled Interrupts (MSI and MSI-X)
 8250/16550 PCI device support
 ☐DIVA Server PRI/PCI support
 ☒Support for COM20020 on PCI
 ☑Teles PCI
```

**PCI MTD driver** (MTD_PCI)

Mapping for accessing flash devices on add-in cards like the Intel XScale IQ80310 card, and the Intel EBSA285 card in blank ROM programming mode (please see the manual for the link settings).

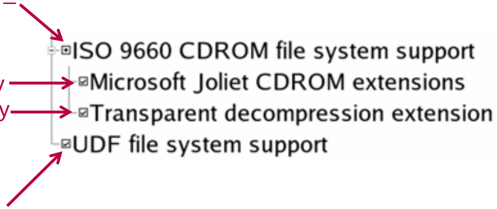
Looks for a keyword in the description string

Allows to select or unselect found parameters.

## Kernel configuration options

Compiled as a module (separate file)
CONFIG_ISO9660_FS=m

Driver options
CONFIG_JOLIET=y
CONFIG_ZISOFS=y



☒ISO 9660 CDROM file system support
☑Microsoft Joliet CDROM extensions
☑Transparent decompression extension
☑UDF file system support

Compiled statically into the kernel
CONFIG_UDF_FS=y

## Corresponding .config file excerpt

```
#
# CD-ROM/DVD Filesystems
#
CONFIG_ISO9660_FS=m
CONFIG_JOLIET=y
CONFIG_ZISOFS=y
CONFIG_UDF_FS=y
CONFIG_UDF_NLS=y

#
# DOS/FAT/NT Filesystems
#
# CONFIG_MSDOS_FS is not set
# CONFIG_VFAT_FS is not set
CONFIG_NTFS_FS=m
# CONFIG_NTFS_DEBUG is not set
CONFIG_NTFS_RW=y
```

Section name
(helps to locate settings in the interface)

All parameters are prefixed
with CONFIG_

## make gconfig

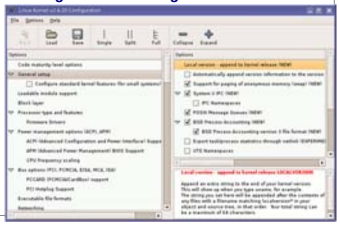
make gconfig

New GTK based graphical configuration interface. Functionality similar to that of make xconfig.

Just lacking a search functionality.

Required Debian packages:

```
host$ sudo apt-get install gtk+-2.0 glib-2.0 libglade2-dev
```



Code maturity level options
General setup
□ Configure standard kernel features (for small systems)
Loadable module support
Block layer
Processor type and features
Firmware Drivers
Power management options (ACPI, APM)
ACPI (Advanced Configuration and Power Interface) Support
APM (Advanced Power Management) BIOS Support
CPU Frequency scaling
Bus options (PCI, PCMCIA, EISA, MCA, ISA)
PCCARD (PCMCIA/CardBus) support
PCI Hotplug Support
Executable file formats
Networking

Local version - append to kernel release (NEW)
□ Automatically append version information to the version
☑ Support for paging of anonymous memory (swap) (NEW)
☑ System V IPC (NEW)
□ IPC Namespaces
☑ POSIX Message Queues (NEW)
☑ BSD Process Accounting (NEW)
☑ BSD Process Accounting version 3 file format (NEW)
□ Export task/process statistics through netlink (EXPERIMENTAL)
□ UTS Namespaces

Local version - append to kernel release (LOCALVERSION)

Append an extra string to the end of your kernel version. This will show up when you type uname, for example. The string you set here will be appended after the contents of any files with a filename matching localversion* in your object and source tree, in that order. Your total string can be a maximum of 64 characters.

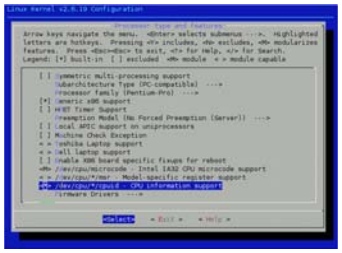
## make menuconfig

make menuconfig

Useful when no graphics are available. Pretty convenient too!

Same interface found in other tools: BusyBox, buildroot...

Required Debian packages: libncurses-dev



```
Linux Kernel v2.6.19 Configuration
                    Processor type and features
 Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted
 letters are hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes
 features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.
 Legend: [*] built-in  [ ] excluded  <M> module  < > module capable

     [ ] Symmetric multi-processing support
         Subarchitecture Type (PC-compatible)  --->
         Processor family (Pentium-Pro)  --->
     [*] Generic x86 support
     [ ] HPET Timer Support
         Preemption Model (No Forced Preemption (Server))  --->
     [ ] Local APIC support on uniprocessors
     [ ] Machine Check Exception
     < > Toshiba Laptop support
     < > Dell laptop support
     [ ] Enable X86 board specific fixups for reboot
     <M> /dev/cpu/microcode - Intel IA32 CPU microcode support
     < > /dev/cpu/*/msr - Model-specific register support
     <M> /dev/cpu/*/cpuid - CPU information support
         Firmware Drivers  --->

              <Select>    < Exit >    < Help >
```

## make oldconfig

make oldconfig

- Needed very often!
- Useful to upgrade a .config file from an earlier kernel release
- Issues warnings for configuration parameters that no longer exist in the new kernel.
- Asks for values for new parameters

If you edit a .config file by hand, it's strongly recommended to run make oldconfig afterwards!

## make allnoconfig

make allnoconfig

- Only sets strongly recommended settings to y.
- Sets all other settings to n.
- Very useful in embedded systems to select only the minimum required set of features and drivers.
- Much more convenient than unselecting hundreds of features one by one!

## Undoing configuration changes

A frequent problem:

- After changing several kernel configuration settings, your kernel no longer works.
- If you don't remember all the changes you made, you can get back to your previous configuration:
  > cp .config.old .config
- All the configuration interfaces of the kernel (xconfig, menuconfig, allnoconfig...) keep this .config.old backup copy.

```
host$ git diff .config
host$ git checkout .config
```

## make help

make help

- Lists all available make targets
- Useful to get a reminder, or to look for new or advanced options!

# Make help

```
make help
Cleaning targets:
  clean                                     - Remove most generated files but keep the config and
                  enough build support to build external modules
  mrproper              - Remove all generated files + config + various backup files
  distclean             - mrproper + remove editor backup and patch files

Configuration targets:
  config                - Update current config utilising a line-oriented program
  nconfig       - Update current config utilising a ncurses menu based program
  menuconfig            - Update current config utilising a menu based program
  xconfig               - Update current config utilising a QT based front-end
  gconfig               - Update current config utilising a GTK based front-end
  oldconfig             - Update current config utilising a provided .config as base
  localmodconfig  - Update current config disabling modules not loaded
  localyesconfig  - Update current config converting local mods to core
  silentoldconfig - Same as oldconfig, but quietly, additionally update deps
  defconfig             - New config with default from ARCH supplied defconfig
  savedefconfig   - Save current config as ./defconfig (minimal config)
  allnoconfig           - New config where all options are answered with no
  allyesconfig          - New config where all options are accepted with yes
  allmodconfig          - New config selecting modules when possible
  alldefconfig    - New config with all symbols set to default
  randconfig            - New config with random answer to all options
  listnewconfig   - List new options
  oldnoconfig     - Same as silentoldconfig but set new symbols to n (unset)

Other generic targets:
  all                                       - Build all targets marked with [*]
* vmlinux               - Build the bare kernel
* modules               - Build all modules
  modules_install - Install all modules to INSTALL_MOD_PATH (default: /)
  firmware_install- Install all firmware to INSTALL_FW_PATH
                  (default: $(INSTALL_MOD_PATH)/lib/firmware)
  dir/            - Build all files in dir and below
  dir/file.[ois]  - Build specified target only
```

# Make help

```
Configuration targets:
  config        - Update current config utilising a line-oriented program
  nconfig       - Update current config utilising a ncurses menu based program
  menuconfig    - Update current config utilising a menu based program
  xconfig       - Update current config utilising a QT based front-end
  gconfig       - Update current config utilising a GTK based front-end
  oldconfig     - Update current config utilising a provided .config as base
  localmodconfig  - Update current config disabling modules not loaded
  localyesconfig  - Update current config converting local mods to core
  silentoldconfig - Same as oldconfig, but quietly, additionally update deps
  defconfig       - New config with default from ARCH supplied defconfig
  savedefconfig   - Save current config as ./defconfig (minimal config)
  allnoconfig     - New config where all options are answered with no
  allyesconfig    - New config where all options are accepted with yes
  allmodconfig    - New config selecting modules when possible
  alldefconfig    - New config with all symbols set to default
  randconfig      - New config with random answer to all options
  listnewconfig   - List new options
  oldnoconfig     - Same as silentoldconfig but set new symbols to n (unset)
```

# Embedded Linux usage

Compiling and installing the kernel
for the host system

# Compiling and installing the kernel

Compiling step

- make

You can speed up compiling by running multiple compile jobs in parallel, especially if you have multiple CPU cores.

Example: make -j 4

# Kernel cleanup targets

- Clean-up generated files
(to force re-compiling drivers):
make clean
- Remove all generated files. Needed when switching from one architecture to another
**Caution: also removes your .config file!**
make mrproper
- Also remove editor backup and patch reject files:
(mainly to generate patches):
make distclean

# Generated files

Created when you run the make command. The kernel is in fact a single binary image, nothing more !

- …/vmlinux
Raw Linux kernel image, non compressed.
- …/arch/<arch>/boot/zImage (default image on arm)
zlib compressed kernel image
- …/arch/<arch>/boot/bzImage (default image on x86)
Also a zlib compressed kernel image.
Caution: bz means “big zipped” but not “bzip2 compressed”!

News: new compression formats are now available since 2.6.30: lzma and bzip2. Free Electrons also contributed lzo support (very fast decompression).

## Files created by make install

- /boot/vmlinuz-<version>
  Compressed kernel image. Same as the one in /arch/<arch>/boot
- /boot/System.map-<version>
  Stores kernel symbol addresses
- /boot/config-<version>
  Kernel configuration for this version

Don't Use

## Files created by make modules_install

/lib/modules/<version>/: Kernel modules + extras

- kernel/
  Module .ko (Kernel Object) files, in the same directory structure as in the sources.
- modules.alias
  Module aliases for module loading utilities. Example line:
  alias sound-service-?-0 snd_mixer_oss
- modules.dep
  Module dependencies
- modules.symbols
  Tells which module a given symbol belongs to.

Don't Use

All the files in this directory are text files.
Don't hesitate to have a look by yourself!

## The Details

To understand a system one must first understand it parts.

--Chris Hallinan

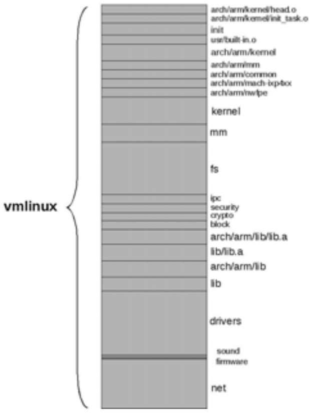
## Link Stage: vmlinux

```
$ arm-angstrom-linux-gnueabi-ld -
  EB -p --no-undefined -X -o
  vmlinux
-T arch/arm/kernel/vmlinux.lds \
arch/arm/kernel/head.o        \
arch/arm/kernel/init_task.o   \
init/built-in.o               \
--start-group                 \
usr/built-in.o                \
arch/arm/kernel/built-in.o    \
arch/arm/mm/built-in.o        \
arch/arm/common/built-in.o    \
arch/arm/mach-ixp4xx/built-in.o \
arch/arm/nwfpe/built-in.o     \
kernel/built-in.o             \
mm/built-in.o                 \
fs/built-in.o                 \
```

```
ipc/built-in.o         \
security/built-in.o    \
crypto/built-in.o      \
block/built-in.o       \
arch/arm/lib/lib.a     \
lib/lib.a              \
arch/arm/lib/built-in.o \
lib/built-in.o         \
drivers/built-in.o     \
sound/built-in.o       \
firmware/built-in.o    \
net/built-in.o         \
-end-group             \
.tmp_kallsyms2.o       \
```

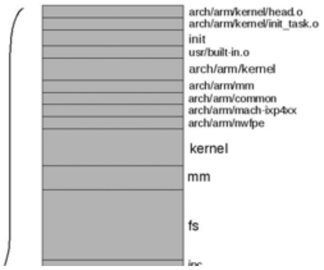
Look in **~/BeagleBoard/oe/build/tmp-angstrom_v2012_05-eglibc/sysroots/*x86_64-linux*/usr/bin/armv7a-angstrom-linux-gnueabi**

## vmlinux image components



## Compare the two

```
$ arm-linux-ld -EB -p --no-
  undefined -X -o vmlinux
-T arch/arm/kernel/vmlinux.lds \
arch/arm/kernel/head.o        \
arch/arm/kernel/init_task.o   \
init/built-in.o               \
--start-group                 \
usr/built-in.o                \
arch/arm/kernel/built-in.o    \
arch/arm/mm/built-in.o        \
arch/arm/common/built-in.o    \
arch/arm/mach-ixp4xx/built-in.o \
arch/arm/nwfpe/built-in.o     \
kernel/built-in.o             \
mm/built-in.o                 \
fs/built-in.o                 \
```



# vmlinux Image Components Description

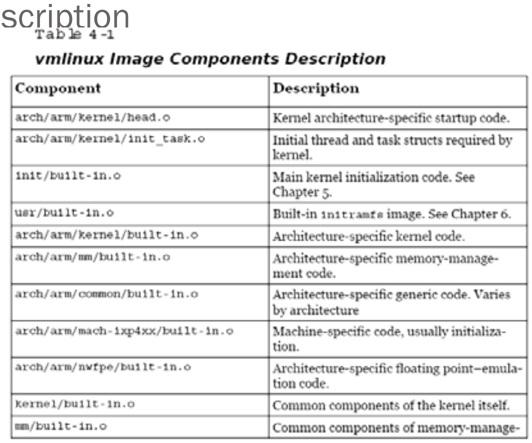


Table 4-1

***vmlinux Image Components Description***

| Component | Description |
|---|---|
| arch/arm/kernel/head.o | Kernel architecture-specific startup code. |
| arch/arm/kernel/init_task.o | Initial thread and task structs required by kernel. |
| init/built-in.o | Main kernel initialization code. See Chapter 5. |
| usr/built-in.o | Built-in initramfs image. See Chapter 6. |
| arch/arm/kernel/built-in.o | Architecture-specific kernel code. |
| arch/arm/mm/built-in.o | Architecture-specific memory-management code. |
| arch/arm/common/built-in.o | Architecture-specific generic code. Varies by architecture |
| arch/arm/mach-ixp4xx/built-in.o | Machine-specific code, usually initialization. |
| arch/arm/nwfpe/built-in.o | Architecture-specific floating point–emulation code. |
| kernel/built-in.o | Common components of the kernel itself. |
| mm/built-in.o | Common components of memory-manage- |