

What's new in Linux 2.6?



What's new in Linux 2.6?

Michael Opdenacker

Free Electrons

<http://free-electrons.com>

Created with [OpenOffice.org](http://openoffice.org) 2.x

Thanks to Nicolas Rougier (Copyright 2003, <http://webloria.loria.fr/~rougier/>) for the Tux image



Rights to copy



Attribution – ShareAlike 2.5

You are free

- to copy, distribute, display, and perform the work
- to make derivative works
- to make commercial use of the work

Under the following conditions



Attribution. You must give the original author credit.



Share Alike. If you alter, transform, or build upon this work, you may distribute the resulting work only under a license identical to this one.

- For any reuse or distribution, you must make clear to others the license terms of this work.
- Any of these conditions can be waived if you get permission from the copyright holder.

Your fair use and other rights are in no way affected by the above.

License text: <http://creativecommons.org/licenses/by-sa/2.5/legalcode>

© Copyright 2004-2007

Free Electrons

feedback@free-electrons.com

Document sources, updates and translations:

<http://free-electrons.com/articles/linux26>

Corrections, suggestions, contributions and translations are welcome!



Best viewed with...

This document is best viewed with a recent PDF reader or with OpenOffice.org itself!

- ▶ Take advantage of internal or external hyperlinks. So, don't hesitate to click on them!
- ▶ Find pages quickly thanks to automatic search
- ▶ Use thumbnails to navigate in the document in a quick way

If you're reading a paper or HTML copy, you should get your copy in PDF or OpenOffice.org format on <http://free-electrons.com/articles/linux26!>



Contents

- ▶ Feature changes
- ▶ Changes for device driver writers
- ▶ Changes for system integrators
- ▶ Summary
- ▶ References



What's new in Linux 2.6?

Feature changes



Preemptible kernel option (1)

Linux 2.6 now features a preemptible option (`CONFIG_PREEMPT`)

- ▶ Original MontaVista Linux 2.4 patch now available in mainstream.
- ▶ Unlike in 2.4, kernel code can be interrupted at almost any time (except when spinlocks are held)
- ▶ This reduces latency for high priority processes.
- ▶ This replaces the Linux 2.4 “low latency” patches that added lots of scheduler calls throughout the kernel code.



Preemptible kernel option (2)

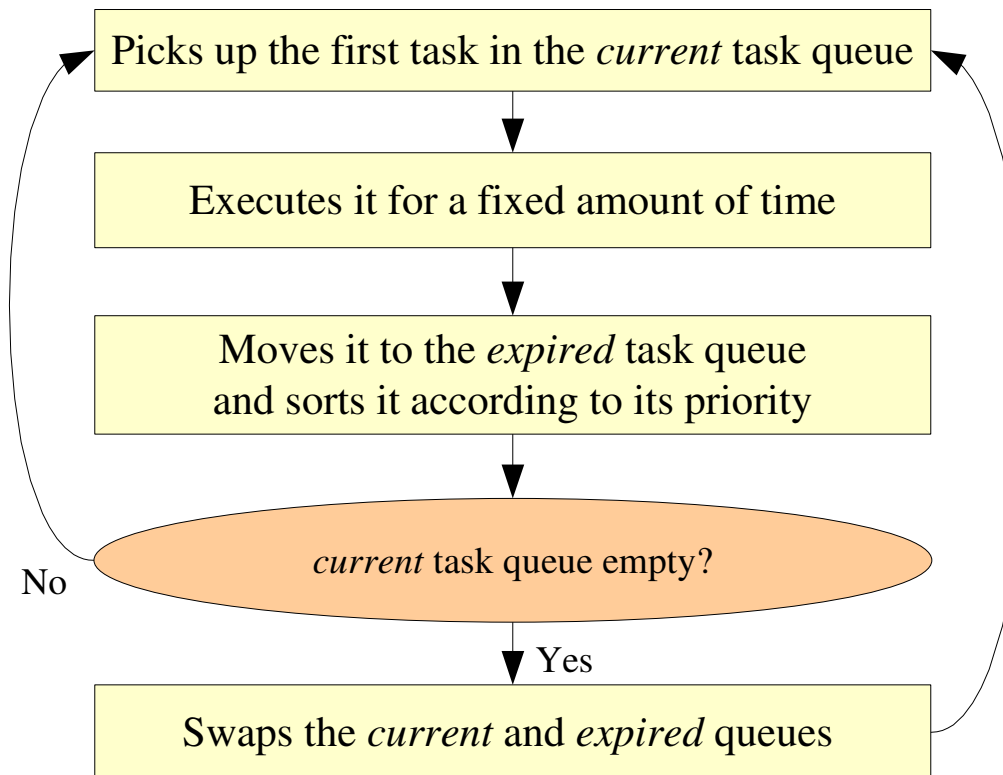
Implications for driver writers

- ▶ Anything can happen between 2 instructions
- ▶ Uniprocessor machines behave like SMP ones. No issue for clean drivers written with SMP in mind.
- ▶ Caution with per-CPU variables (not in use in device drivers so far)
See <http://lwn.net/Articles/22911/>



Improved scheduler

- ▶ In Linux 2.4, the scheduler had to look at every task to determine its relative importance. Pretty slow for numerous tasks!
- ▶ In Linux 2.6, scheduling is done in a constant amount of time



Other timing improvements (1)

▶ New synchronization primitives

- ▶ **Mutex:** function to ensure that a shared resource is used by only one task at a time. Time consuming system call when the resource is not used and the task doesn't need to be blocked.
- ▶ **Fast user space mutexes:** enables to check the resource availability from userspace before making a syscall to block the task. Also let applications prioritize access to shared resources amongst tasks.



Other timing improvements (2)

- ▶ Improvements in I/O subsystems and rewritten I/O scheduler
 - ▶ Makes sure no process gets stuck waiting for too long
- ▶ Can be built with no virtual memory system
 - ▶ Eliminates the overhead of page fault handling. More efficient than just having no swap partition.



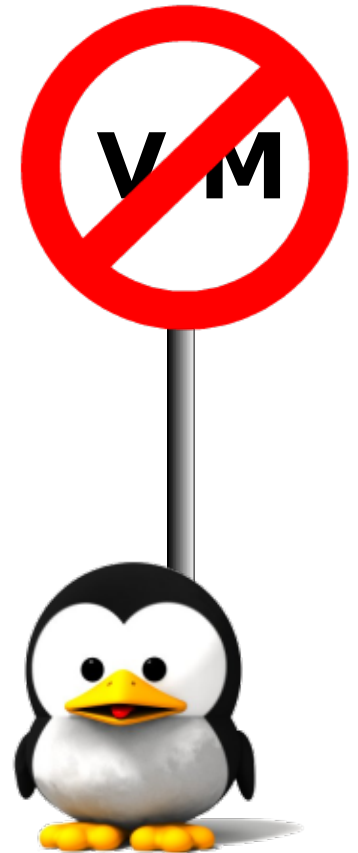
POSIX threads (pthreads)

- ▶ Linux 2.4: Native POSIX Thread Library (NPTL) only available as patches
- ▶ Linux 2.6: improved pthreads implementation, available in the standard kernel
- ▶ POSIX threads advantages over Linux threads
 - ▶ POSIX signals can't be lost and can carry information
 - ▶ Can exchange signals between threads, instead of just between processes.
 - ▶ Make it easier to set task priorities and schedule periodic tasks (useful for polling resources) with high precision (high resolution timers available)



Merging with uClinux

- ▶ Most of uClinux code now merged with the mainstream Linux tree.
- ▶ uClinux m68k tree now released in mainstream: `arch/m68knommu`
- ▶ Other supported architectures: Hitachi's H8/300 series, NEC v850 processor
- ▶ `arch/armnommu` not merged yet. Still available as a separate patch.
- ▶ Linux 2.6 can be built with no virtual memory



NUMA support

NUMA: Non Uniform Memory Access

- ▶ Support for high-end systems with multiple processors, with separate memory pools directly connected to each processor.
- ▶ Memory access much faster if done by the local processor
- ▶ New topology API for these systems in Linux 2.6
- ▶ The Linux scheduler can now use this topology information to run processes on the best processor.



Support for custom designs

- ▶ Linux 2.6 standardizes the support of *subarchitectures*, i.e. designs with the same processor but with different IRQ schemes or component differences.
- ▶ This avoids code duplication and enables the reuse of all the sources files that are in common.



Hyperthreading support

Hyperthreading: makes the processor appear as two processors. So far only implemented in Intel P4.

Linux 2.4: could overload 1 of the 2 virtual processors

Linux 2.6: the scheduler now knows how to optimize the load between the 2 processors.



Scalability improvements

- ▶ Support up to 64GB of RAM in paged mode on 32 bit x86 systems (Intel Physical Address Extension)
- ▶ Process rollover: 32000 to 1 billion
Improves application starting performance on very busy systems
- ▶ Number of groups and users: 65000 to 4 billion
- ▶ Improved 64 bit support on block devices. Up to 16 TB file systems.
- ▶ Supports 4000 disks on a system
- ▶ 256 to 4095 (2^{12}) major device types, 255 to 1048576 (2^{20}) minor device types.



Module subsystem

- ▶ Brand new module subsystem, much code moved into the kernel.
- ▶ Module files suffixed with `.ko` (kernel object) instead of `.o`
- ▶ Can disable module unloading (`CONFIG_MODULE_UNLOAD=n`)
- ▶ Modules can announce which hardware they support
- ▶ Module interface differences. Broken compatibility (see the programming section).
- ▶ `/etc/modules.conf` replaced by `/etc/modprobe.conf`!



Unified Device Model

- ▶ Linux 2.6 provides a unified interface to reference device buses, subdevices, power information.
- ▶ Information available through the new `/sys` mount point (replacing `/proc/sys`)
- ▶ Eliminates differences between legacy and hotplug devices (which simplifies their management)
- ▶ Enables efficient power management and better ACPI support.



External devices

USB

- ▶ USB 2.0 support
- ▶ USB OTG support introduced in Linux 2.6.9.
See <http://www.linux-usb.org/gadget/h2-otg.html>
- ▶ USB device support (“gadget” drivers)
The Linux machine (PDA, phone) behaves as a USB device.



Wireless devices

Wireless devices

- ▶ 802.11x

Single wireless subsystem and API. Make it possible to implement common userspace tools for all devices

- ▶ IrDA

Power management support

- ▶ Bluetooth

Official, improved Bluetooth support (already available in late 2.4 kernels)



Block devices

IDE

- ▶ Rewritten IDE driver
- ▶ Using IDE CD/RW drives no longer need SCSI emulation

Filesystems

- ▶ NTFS read and write (still experimental) support
- ▶ Windows Logical Disk Manager (new partitioning scheme introduced by Windows 2000)
- ▶ Improved FAT12 support (used in media players with flash storage)

And many more!



User interface devices (1)

Input devices

- ▶ Reworked and modularized human interface layer
- ▶ Support for headless systems (no mouse, no keyboard, no display)
- ▶ Touchscreen support
- ▶ Keyboards and mice only export one node in `/dev/input`, whatever the hardware and protocol



User interface devices (2)

Display

- ▶ Framebuffer console
 - ▶ 24 bpp boot logo
 - ▶ Support resizing and rotating (useful for PDAs)



Multimedia

Audio

- ▶ ALSA (Advanced Linux Sound Architecture) replaced OSS (Open Sound System)

Video

- ▶ Major upgrade of the Video4Linux subsystem and API. More hardware and features supported.
- ▶ Support for Digital Video Broadcasting



Networking

- ▶ IPsec protocols support: cryptographic security at network protocol level (transparent to applications)
- ▶ Multicast networking support

Network filesystems

- ▶ NFS v4 support
- ▶ NFS server: 64 times as many concurrent users and larger request queues
- ▶ Support for CIFS (SMB extension)
- ▶ InterMezzo filesystem support. Supports disconnected operation, working on locally cached files)



Misc (1)

- ▶ User Mode Linux (arch/uml)

Linux kernel running as a user process

Can run several virtual machines within the physical one

Limitation: only supports x86

- ▶ Laptops

- ▶ Laptop mode (since 2.6.6)

Delays writes to hard disk to keep it off as long as possible

- ▶ New edition of software suspend-to-disk



Misc (2)

- ▶ Configuration management
Option to make the kernel configuration file available through `/proc/config.gz` (`CONFIG_IKCONFIG=y`)
- ▶ Compressed kernel size
2.6.0 vs 2.4.0: +68%
2.6.8.1 vs 2.4.27: +15%



Removed in Linux 2.6

- ▶ Kernel built-in web server (kHTTPd)
All performance bottlenecks with Apache and others
have been resolved
- ▶ Deprecated features (such as `/dev/cua`)
- ▶ Old code



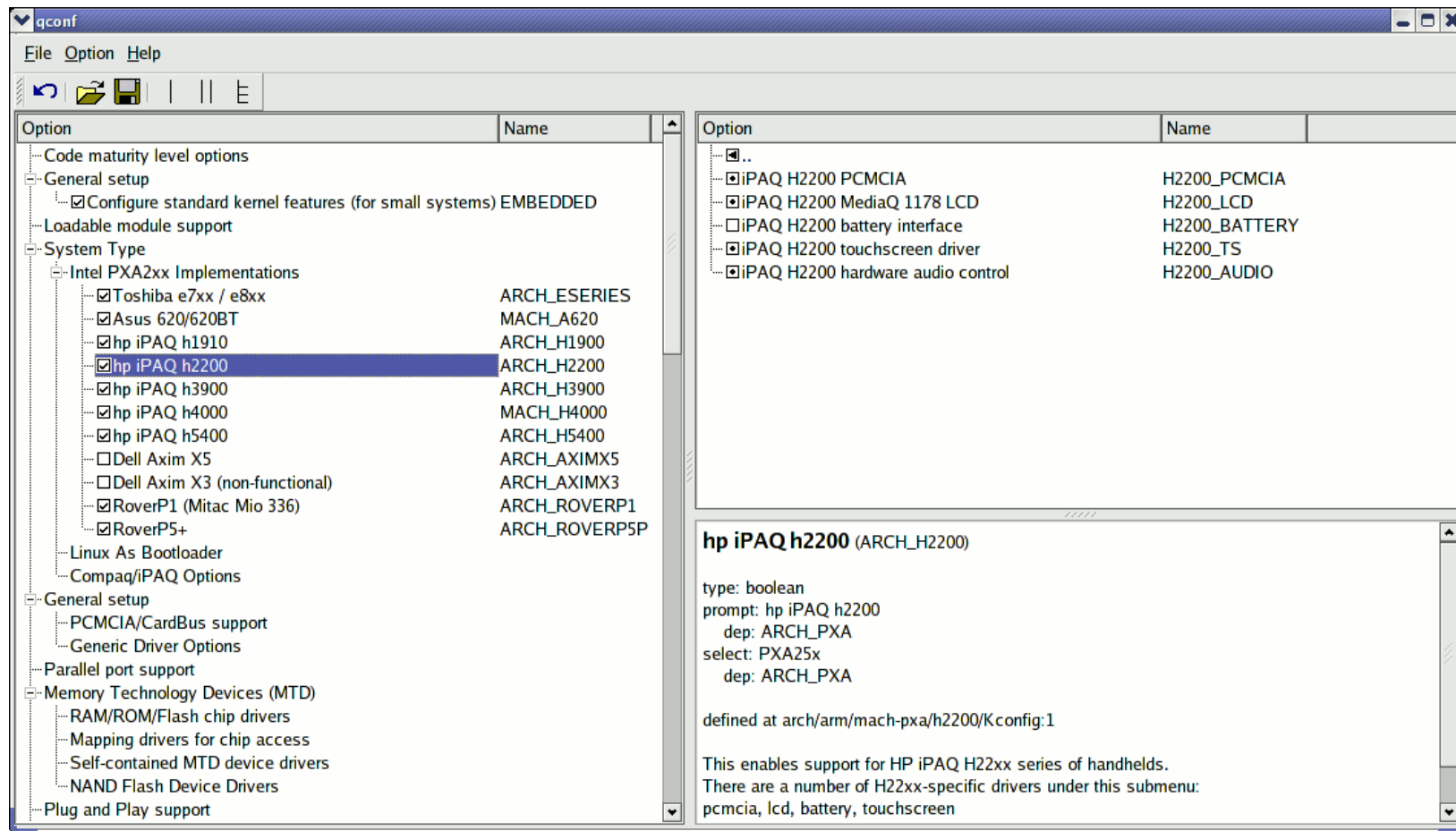
What's new in Linux 2.6?

Changes for device driver writers



Kernel configuration (1)

qconf: new qt based graphical configuration tool



Kernel configuration (2)

qconf makes kernel configuration easier

- ▶ Still run as `make xconfig`
- ▶ Much more convenient interface. Easier to browse sections and subsections. Automatically hides unneeded sections (which condition is not satisfied).
- ▶ Make sure you read `help -> introduction`: useful options! You can display configuration variable names, conditions.
- ▶ File browser: easier to load configuration files



Kernel building

- ▶ `make modules` no longer needed
Modules built by the `make` command.
- ▶ `make depends` (or `make dep`) no longer exists



Module writing (1)

- ▶ No longer need the below line:

```
#define MODULE
```

- ▶ Different way of counting module instances
MOD_INC_USE_COUNT is deprecated
Should be managed outside the module.
Otherwise, race condition risks.



Module writing (2)

- ▶ `MODULE_PARAM` no longer exists
Use `module_param(name, type, perm)` instead
Offers better type checking and worked for statically compiled modules too.
- ▶ Symbols are no longer exported by default
`EXPORT_NO_SYMBOLS` is redundant



Driver writing

▶ `devfs` obsolete

Time to get rid of all `devfs` related function calls, and use `udev` instead from userspace!



Module building

Building modules now more complicated

Each module requires objects from the kernel tree

- ▶ Build environment information (vermagic)
- ▶ Symbol version information

Consequences

- ▶ Need a configured and built kernel.
However, `/lib/module/<linux-version>/build` is sufficient if available. Just contains the needed sources.
- ▶ Need to use `kbuild`, the kernel building system. Can no longer use a plain `gcc` command!



What's new in Linux 2.6?

Changes for system integrators



Migrating an existing system to 2.6 (1)

- ▶ Upgrade your `.config`

You can run `make oldconfig` and check that all the settings you need are still there. Add missing ones or useful new ones with `make xconfig`.

- ▶ Update your `module-init-tools`:

`insmod`, `modprobe`, `rmmod`... (impacted by module file and kernel subsystem changes)

- ▶ Update all scripts running `insmod` with `.o` module files

- ▶ Convert `/etc/modules.conf` to `/etc/modprobe.conf`



Migrating an existing system to 2.6 (2)

- ▶ Add `/sys` to `/etc/fstab`
- ▶ Modify startup and halt scripts to mount and unmount `/sys` too.
- ▶ Implement all the above changes in your `initrd` too (if any)!
- ▶ Migrating from OSS to ALSA
Good idea to migrate to Linux 2.4 ALSA first, to reduce complexity.



Conclusion

- ▶ Lots of improvements
- ▶ But sometimes broken compatibility. Usefulness of breaking compatibility:
 - ▶ Avoids building unnecessary compatibility code, preventing things from being implemented in a simple way
 - ▶ Allows to update unsafe code
 - ▶ No restrictions on what can be done (except standards compliance)
 - ▶ Helps to find out which code is no longer needed (when nobody complains)



The word from Linus

“I am totally `_uninterested_` in a stable ABI for kernel modules, and in fact I'm actively against even `_trying_`. I want people to be very much aware of the fact that kernel internals do change, and that this will continue.”

linux-kernel mailing list, December 2003

ABI: Application Binary Interface

Low level interface which guarantees binary compatibility



References

- ▶ Wonderful world of Linux 2.6, by Joseph Pranevich
<http://kniggit.net/wwol26.html>
- ▶ Linux 2.6 driver porting, by Jonathan Corbet
<http://lwn.net/Articles/driver-porting/>
- ▶ Migrating to Linux 2.6, William von Hagen
A series of 6 articles covering all migration aspects
<http://linuxdevices.com/articles/AT3855888078.html>




Thanks

- ▶ Joseph Pranevich, for his very thorough review of Linux 2.6 changes:
<http://kniggit.net/wwol26.html>
- ▶ Jonathan Corbet, for his dedication to explaining kernel changes. See
<http://lwn.net/Articles/driver-porting/>
- ▶ To the OpenOffice.org project, for their presentation and word processor tools which satisfied all my needs.
- ▶ To the Handhelds.org community, for giving me so much help and so many opportunities to help.
- ▶ To the members of the whole Free Software and Open Source community, for sharing the best of themselves: their work, their knowledge, their friendship.
- ▶ To people who sent corrections and updates:
Matti Aaltonen, David Brownell





Related documents



Free Electrons

Embedded Freedom

HOME DEVELOPMENT SERVICES TRAINING DOCS COMMUNITY COMPANY BLOG

Recent blog posts

ELC Europe in Grenoble

Free Electrons at ELC

Linux kernel 2.6.29 - New features for embedded users

The Buildroot project begins a new life

FOSDEM 2009 videos

USB-Ethernet device for Linux

Program for Embedded Linux Conference 2009 announced

Public session changes


Real hardware in our training sessions

Call for presentations for the LSM embedded track

Docs

Most of the below documents are presentations used in our [training sessions](#), or in technical conferences.

License

 All our documents are available under the terms of the [Creative Commons Attribution-ShareAlike 3.0 license](#). This essentially means that you are free to download, distribute and even modify them, provided you mention us as the original authors and that you share these documents under the same conditions.

Linux kernel

- [Embedded Linux kernel and driver development](#)
- [New features in Linux 2.6](#) (since 2.6.10)
- [Kernel initialization](#)
- [Porting Linux to new hardware](#)
- [Power management in Linux](#)
- [Linux PCI drivers](#)
- [Block device drivers](#)
- [Linux USB drivers](#)
- [DMA](#)

Architecture specific documents

- [ARM Linux specifics](#)
- [Linux on TI OMAP processors](#)

Embedded Linux system development

- [Embedded Linux system development](#)
- [Real time in embedded Linux systems](#)
- [Block filesystems](#)
- [Flash filesystems](#)
- [Free software development tools](#)
- [The U-boot bootloader](#)
- [The GRUB bootloader](#)
- [The blob bootloader](#)
- [Hotplugging with udev](#)
- [Introduction to uClinux](#)
- [Java in embedded Linux](#)
- [Embedded Linux optimizations](#)
- [Audio in embedded Linux systems](#)
- [Multimedia in embedded Linux systems](#)
- [Embedded Linux From Scratch... in 40 minutes!](#)
- [Building embedded Linux systems with Buildroot](#)
- [Developing embedded distributions with OpenEmbedded](#)
- [The Scratchbox development environment](#)

Miscellaneous

- [Introduction to the Unix command line](#)
- [SSH](#)
- [Linux virtualization solutions](#) (with an embedded perspective)
- [Advantages of Free Software and Open Source in embedded systems](#)
- [Introduction to GNU/Linux and Free Software](#)

All our technical presentations
on <http://free-electrons.com/docs>

- ▶ Linux kernel
- ▶ Device drivers
- ▶ Architecture specifics
- ▶ Embedded Linux system development



How to help

You can help us to improve and maintain this document...

- ▶ By sending corrections, suggestions, contributions and translations
- ▶ By asking your organization to order development, consulting and training services performed by the authors of these documents (see <http://free-electrons.com/>).
- ▶ By sharing this document with your friends, colleagues and with the local Free Software community.
- ▶ By adding links on your website to our on-line materials, to increase their visibility in search engine results.

Linux kernel

- Linux device drivers
- Board support code
- Mainstreaming kernel code
- Kernel debugging

Embedded Linux Training

All materials released with a free license!

- Unix and GNU/Linux basics
- Linux kernel and drivers development
- Real-time Linux, uClinux
- Development and profiling tools
- Lightweight tools for embedded systems
- Root filesystem creation
- Audio and multimedia
- System optimization

Free Electrons

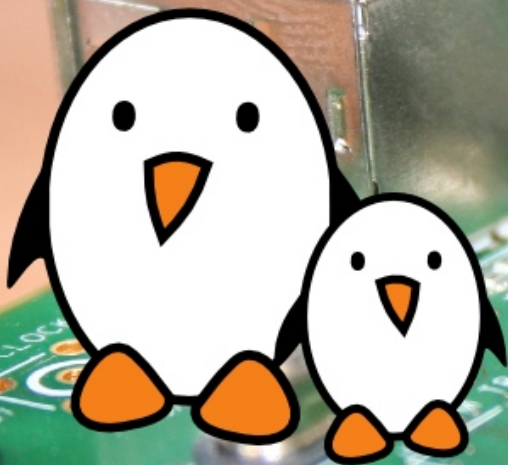
Our services

Custom Development

- System integration
- Embedded Linux demos and prototypes
- System optimization
- Application and interface development

Consulting and technical support

- Help in decision making
- System architecture
- System design and performance review
- Development tool and application support
- Investigating issues and fixing tool bugs



Free Electrons
Embedded Linux Experts

<http://free-electrons.com>