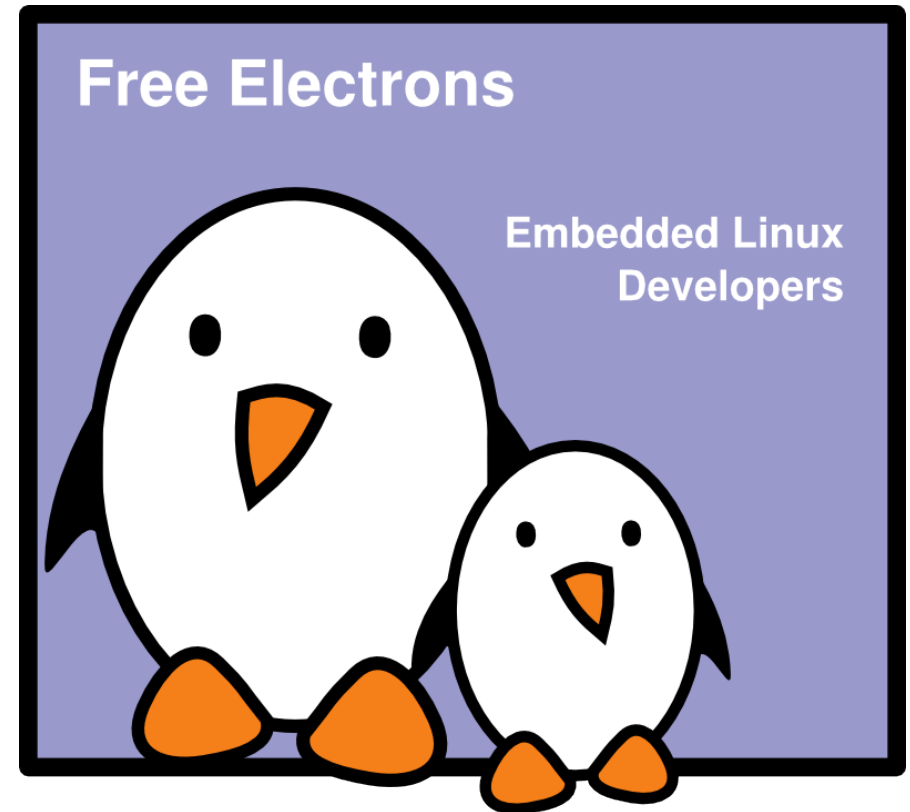




## The blob bootloader

Thomas Petazzoni  
**Free Electrons**





# Rights to copy

© Copyright 2008-2009, Free Electrons  
[feedback@free-electrons.com](mailto:feedback@free-electrons.com)

Document sources, updates and translations:  
<http://free-electrons.com/docs/blob>

Corrections, suggestions, contributions and translations are welcome!

Latest update: Sep 15, 2009



**Attribution – ShareAlike 3.0**

**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/3.0/legalcode>



# Blob bootloader

- ▶ Blob, Boot Loader Object, bootloader for StrongARM based platforms  
<http://www.lartmaker.nl/lartware/blob/>
- ▶ Written by Jan-Derk Bakker and Erik Mouw, GPL-licensed
- ▶ Simple bootloader: < 15k lines of code, including the code for all supported boards
- ▶ Inactive mainline
  - ▶ Latest stable release in august 2001
  - ▶ Latest development version in january 2002
  - ▶ Last serious activity in the CVS in 2004, few commits in 2006



# Supported boards

- ▶ Few boards supported in the released version
  - ▶ Assabet
  - ▶ Assabet+Neponset
  - ▶ LART
  - ▶ iPaq
  - ▶ System3
  - ▶ Badge4
  - ▶ NESAs
  - ▶ PLEB
- ▶ Shannon (aka Tuxscreen)
- ▶ Vercel UD-1
- ▶ Jornada 720
- ▶ Support for other boards can be added fairly easily
- ▶ Non-mainlined ports
  - ▶ Intel Lubbock
  - ▶ Motorola Linux phones, with download over USB
  - ▶ Others ?



# Features

- ▶ Initialize hardware
- ▶ Download a kernel and ramdisk through the serial line
- ▶ Write kernel and ramdisk to Flash
- ▶ Boot the kernel
- ▶ Determine the memory layout
- ▶ Give a command line to the kernel



# Using Blob

- ▶ On boot, by default, Blob initializes the hardware, loads and boots the kernel.
- ▶ Messages are available through the serial port, 9600 bauds, 8 bits, no parity, 1 stop bit

```
blob version 2.0.3
[...]
Memory Map:
  0x08000000 @ 0xC0000000 (8MB)
[...]
  0x08000000 @ 0xC9000000 (8MB)
Loading blob from flash . done
Loading kernel from flash ..... done
Loading ramdisk from flash ..... done
Autoboot in progress, press any key to stop ...
Starting kernel ...
Uncompressing Linux...done.
```

- ▶ To access the command line, press any key during the ten seconds break before kernel boot



# Using Blob : Commands

- ▶ `boot [kernel options]`  
Boot the kernel
- ▶ `download { blob | kernel | ramdisk }`  
Download blob, kernel or ramdisk from serial to RAM
- ▶ `flash { blob | kernel | ramdisk }`  
Write blob, kernel or ramdisk from RAM to Flash
- ▶ `Reblob`  
Restart blob
- ▶ `Reboot`  
Restart the system
- ▶ `reload { blob | kernel | ramdisk }`  
Reload the blob, kernel or ramdisk from Flash to RAM
- ▶ `reset`  
Reset the terminal
- ▶ `speed`  
Set the download speed
- ▶ `Status`  
Get status information



# Using Blob: download (1)

- ▶ Downloading blob, the kernel or ramdisk is done through the same serial line as the console
- ▶ When starting the download command, the speed is set to 115200 bauds and blob waits for an uuencoded version of blob, kernel or ramdisk

- ▶ On blob side

```
blob> download kernel
```

```
Switching to 115200 baud
```

```
You have 60 seconds to switch your terminal emulator  
to the same speed and start downloading. After that  
blob will switch back to 9600 baud.
```





# Using Blob: download (2)

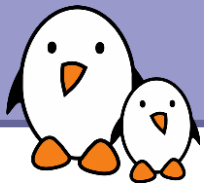
- ▶ On the host side, change the speed to 115200 bauds and either:
  - ▶ Upload the file with Minicom using the ASCII method
  - ▶ In another shell, run  
`uuencode zImage zImage > /dev/ttySx`
- ▶ And the end of the upload, on blob side :  
(Please switch your terminal emulator back to 9600 baud)  
`Received 65536 (0x00010000) bytes.`
- ▶ On the host side, switch back the speed to 9600 bauds



# Compiling Blob

- ▶ Requires a cross-compiler and a configured Linux kernel source tree
- ▶

```
export CC=/path/to/arm-linux-gcc
export OBJCOPY=/path/to/arm-linux-objcopy
./configure
  --with-linux-prefix=/path/to/linux/sources/
  --with-board=boardname
arm-unknown-linux-gnu
make
```
- ▶ Two versions
  - ▶ `src/blob`, the bootloader that can directly be booted
  - ▶ `src/blob-chain`, the bootloader that can be loaded from another bootloader



# 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

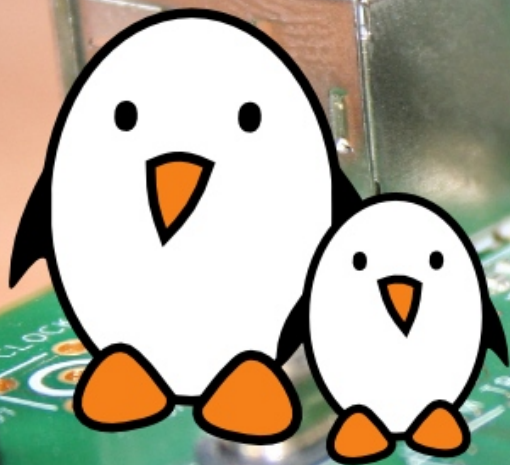
## Our services

### 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>