Buildroot: what’s new?

Thomas Petazzoni
thomas.petazzoni@bootlin.com
Thomas Petazzoni

- CTO and Embedded Linux engineer at **Bootlin**
  - Embedded Linux specialists.
  - Development, consulting and training.
  - [http://bootlin.com](http://bootlin.com)

- **Contributions**
  - **Kernel support for the Marvell Armada**
    ARM SoCs from Marvell
  - Major contributor to **Buildroot**, an open-source, simple and fast embedded Linux build system

- **Toulouse**, south west of France
- Windsurfing, snowboarding
Poll time

- Who already knows about Buildroot?
Poll time

- Who already knows about Buildroot?
- Who is already using Buildroot?
Poll time

- Who already knows about Buildroot?
- Who is already using Buildroot?
- Who is using OpenEmbedded / Yocto Project?
Poll time

- Who already knows about Buildroot?
- Who is already using Buildroot?
- Who is using OpenEmbedded / Yocto Project?
- Who is using OpenWRT / LEDE?
Poll time

- Who already knows about Buildroot?
- Who is already using Buildroot?
- Who is using OpenEmbedded / Yocto Project?
- Who is using OpenWRT / LEDE?
- Who is using another build system?
Buildroot at a glance

▶ Is an **embedded Linux build system**, builds from source:
  ▶ cross-compilation toolchain
  ▶ root filesystem with many libraries/applications, cross-built
  ▶ kernel and bootloader images

▶ **Fast**, simple root filesystem in minutes
▶ **Easy** to use and understand: kconfig and make
▶ **Small** root filesystem, default 2 MB
▶ More than **2200 packages** available
▶ Generates filesystem images, not a distribution
▶ **Vendor neutral**
▶ Active community, stable releases every 3 months
▶ Started in 2001, oldest still maintained build system
▶ **http://buildroot.org**
What’s new?

- Last *What’s new* talk at the *Embedded Linux Conference 2014*, i.e. 3.5 years ago
- Lots of things have changed and improved in Buildroot since then, time for a new *What’s new* talk!
- Main topics discussed
  - Project activity
  - Release schedule and LTS
  - Architecture support
  - Toolchain support
  - Infrastructure improvements
  - Testing improvements
  - Misc
Project activity: per commits

Number of commits per release

Average Number of commits per release

- Kernel, drivers and embedded Linux - Development, consulting, training and support - https://bootlin.com
Project activity: contributors

Number of contributors per release

- Kernel, drivers and embedded Linux - Development, consulting, training and support - https://bootlin.com
Project activity: packages

Number of packages

0 1,000 2,000

2012.05 2012.08 2012.11 2013.02 2013.05 2013.08 2013.11 2014.02 2014.05 2014.08 2014.11 2015.02 2015.05 2015.08 2015.11 2016.02 2016.05 2016.08 2016.11 2017.02 2017.05 2017.08
Since 2009, releases every **three months**: YYYY.02, YYYY.05, YYYY.08, YYYY.11

- Never skipped a release or missed a release date!
Release schedule and LTS

Since 2009, releases every **three months**: YYYY.02, YYYY.05, YYYY.08, YYYY.11

- Never skipped a release or missed a release date!

Until now

- Point releases for the latest stable, but only until the next stable release
- Only option to get updates is to migrate to the next release

Since 2017.02:

- LTS release
- Every YYYY.02 release will be maintained for one year, with security, build and bug fixes
- Already did 6 point releases for 2017.02, from 2017.02.1 to 2017.02.6: April, May, June, July (x2), September.
- 526 commits, including 183 commits for security updates/fixes

Effort done by Peter Korsgaard - Kernel, drivers and embedded Linux - Development, consulting, training and support - [https://bootlin.com](https://bootlin.com)
Since 2009, releases every **three months**: YYYY.02, YYYY.05, YYYY.08, YYYY.11
- Never skipped a release or missed a release date!

Until now
- Point releases for the latest stable, but only until the next stable release
- Only option to get updates is to migrate to the next release

Since 2017.02: **LTS release**
- Every YYYY.02 release will be **maintained for one year**, with security, build and bug fixes
- Already did **6 point releases** for 2017.02, from 2017.02.1 to 2017.02.6: April, May, June, July (x2), September.
- 526 commits, including **183 commits for security** updates/fixes
- Effort done by Peter Korsgaard
Maintenance and physical meetings

- Used to have a single committer/project maintainer: Peter Korsgaard

- **Two additional committers** have been appointed in recent years:
  - Thomas Petazzoni (i.e, me)
  - Arnout Vandecappelle

- **Physical meetings**
  - One meeting before ELCE, was held last Saturday/Sunday
  - One meeting after FOSDEM, Brussels
  - One more private hackaton for the core team in the summer
Architectures

- Probably the build system with the largest number of architectures supported
Architectures

- Probably the build system with the largest number of architectures supported
  - ARC, ARM(eb,nommu), ARM64(eb), Blackfin, C-Sky, m68k, Microblaze(el), mips(64)(el), nios2, OpenRISC, PowerPC(64)(le), SuperH, Sparc(64), x86(64), Xtensa
Architectures

- Probably the build system with the largest number of architectures supported
  - ARC, ARM(eb,nommu), ARM64(eb), Blackfin, C-Sky, m68k, Microblaze(el), mips(64)(el), nios2, OpenRISC, PowerPC(64)(le), SuperH, Sparc(64), x86(64), Xtensa

- ARM Cortex M3/M4 noMMU support

- Merge of ARM/ARM64 options, to select ARM64 cores

- PowerPC64 little endian and big endian support, contributions from IBM

- MIPS improvements: MIPS32r6 and MIPS64r6 support, MIPS core selection, NaN/FP32 selection, contributions from Imagination Technologies

- OpenRISC, C-Sky, Sparc64 support

- Re-enabling of m68k both Coldfire (noMMU) and 68k (MMU)

- Blackfin and Microblaze improved with uClibc-ng support

- SH64 and AVR32 support removed
Toolchains

- Buildroot supports:
  - Building its own toolchain: so-called **internal toolchain** back-end
  - Using an existing pre-built toolchain: **external toolchain** back-end
Toolchains

- Buildroot supports:
  - Building its own toolchain: so-called **internal toolchain** back-end
  - Using an existing pre-built toolchain: **external toolchain** back-end

- Internal toolchain improvements
  - Support for **musl** C library added
  - Moved from uClibc to **uClibc-ng**
  - **Regular updates**: gcc up to 7.x (default is 6.x), binutils 2.29 (default 2.28), gdb 8.0 (default 7.12), glibc 2.26, uClibc-ng 1.0.26, musl 1.1.16.
  - **LTO** and **Fortran** support
  - Toolchain **wrapper** also used for the internal back-end: allows sanity checks
  - eglibc removed
Toolchains

- Buildroot supports:
  - Building its own toolchain: so-called **internal toolchain** back-end
  - Using an existing pre-built toolchain: **external toolchain** back-end

- Internal toolchain improvements
  - Support for **musl** C library added
  - Moved from uClibc to **uClibc-ng**
  - **Regular updates**: gcc up to 7.x (default is 6.x), binutils 2.29 (default 2.28), gdb 8.0 (default 7.12), glibc 2.26, uClibc-ng 1.0.26, musl 1.1.16.
  - **LTO** and **Fortran** support
  - Toolchain **wrapper** also used for the internal back-end: allows sanity checks
  - eglibc removed

- External toolchain improvements
  - Logic split in **multiple packages**, one per external toolchain family
  - Include/library paths **sanity checking** in the wrapper
  - Numerous **updates**: Linaro/Sourcery toolchains, new Imagination Technologies toolchains, removed old toolchains
• Side project, but Buildroot related
• Freely available **pre-built toolchains** for a wide range of architectures and configurations
• **34 different architecture/variants**
• glibc/uClibc-ng/musl, as available
• Two versions: stable and bleeding-edge
• Built by Buildroot, on Gitlab CI
• **Tested** by building a Linux kernel and minimal userspace, and if supported, booting under QEMU
• [http://toolchains.bootlin.com](http://toolchains.bootlin.com)
Infrastructure: relocatable SDK

- output/host contains
  - The native tools, including the cross-compiler
  - The toolchain *sysroot*, with all libraries and headers
- Can be used as an SDK
  - Allows application developers to build applications targeting the root filesystem without having to use Buildroot
- output/host is now **relocatable**, which makes it easier to use as an SDK
- `make sdk` prepares the SDK
  - Replaces absolute *RPATH* in native binaries by relative ones
  - Installs a `relocate-sdk.sh` script that users of the SDK must run to fix up the remaining absolute paths
- Related work:
  - output/host/usr/* moved to output/host/
  - *RPATH* in target binaries are now cleaned up
Infrastructure: hashes

- Each package now has a `<pkg>.hash` file that contains hashes
  - For the tarball being downloaded
  - For the patches being downloaded, if any
  - For the license files included in the upstream source code
- Tarball/patch hashes are checked when the package is extracted, i.e. at every build
- License files hashes are checked when generating the licensing report
  (make legal-info)
- Allows
  - check the integrity of what is downloaded,
  - ensure that tarballs stored locally have not been modified
  - detect if license terms are changed upstream
  - detect if upstream messes up and re-uploads a new (but different) tarball
- Almost all packages have a hash file now: 2166 packages out of 2232 packages

sha256 52426e75432e46996dc90f24fca027805a341c38fbbb022b60dc9acd2677ccf4 bind-9.11.1-P3.tar.gz
sha256 d3906dfe153e2c48440d3ca1d5319f5e89b4b820cfc5d0779c23d7ac2b175e9 COPYRIGHT
Packages include a description of the license and paths to license files

```
DBUS_LICENSE = AFL-2.1 or GPL-2.0+ (library, tools), GPL-2.0+ (tools)
DBUS_LICENSE_FILES = COPYING
```

Collected by `make legal-info`: source tarballs, patches, license files, manifests

Improvements

- **SPDX license codes** used to describe the licensing of all packages
- **Hashes** added for license files, in order to detect changes
- Storage of source code for binary artifacts such as pre-built toolchains, using `<pkg>_ACTUAL_SOURCE`
- Many **more packages have license details**: 2143 out of 2232 packages
Infrastructure: BR2_EXTERNAL

- BR2_EXTERNAL allows to implement packages, store *defconfigs* and other build-related files outside of the Buildroot tree
- Allows separating the upstream Buildroot from project/company-specific packages and data
- Simplified form of *layer* concept found in Yocto/OE/OpenWRT
- Available since 2014.02

**Improvements**

- Support for multiple BR2_EXTERNAL directories
- Support for implementing bootloader packages and filesystem image formats in BR2_EXTERNAL
Infrastructure: package infrastructures

- Base infrastructure: generic-package
- Specialized infrastructures for specific build systems: autotools-package, cmake-package, python-package
- Improvements
  - python-package extended to support Python 3.x
  - New perl-package infrastructure for Perl packages
  - New virtual-package infrastructure for virtual packages such as OpenGL, jpeg, udev
  - New waf-package infrastructure for Waf based packages
  - New rebar-package infrastructure for Erlang packages
  - New kconfig-package infrastructure, used by Linux, BusyBox, uClibc-ng, Barebox, U-Boot, etc.
  - New kernel-module infrastructure to help building kernel modules
Infrastructure: graphing

- Already existing:
  - `make graph-depends`, `make <pkg>-graph-depends`, to generate dependency graphs
  - `make graph-build`, graph of the build time per package

- Improvements
  - `make graph-size`, size of the filesystem, split by package
  - `make <pkg>-graph-rdepends`, graph of the reverse dependencies
Infrastructure: graphing

- **Already existing:**
  - `make graph-depends`, `make <pkg>-graph-depends`, to generate dependency graphs
  - `make graph-build`, graph of the build time per package

- **Improvements**
  - `make graph-size`, size of the filesystem, split by package
  - `make <pkg>-graph-rdepends`, graph of the reverse dependencies
Infrastructure: *skeleton* restructuring

- **Skeleton**: base of the root filesystem, main directory hierarchy and basic config files
- Initially a single set of files copied to `TARGET_DIR` at the beginning of the build
- Now:
  - `skeleton` is a virtual package, that depends on `skeleton-init-sysv`, `skeleton-init-systemd`, `skeleton-init-none` or `skeleton-custom`
  - Common base: `skeleton-init-common`
  - Core init scripts moved in `initscripts`
  - Allows to avoid SysV cruft in systemd systems and vice-versa
  - Allowed to implement read-only rootfs support with `systemd`
  - Support for merged `/usr`, used by `systemd` support
Support for generating filesystem images in a large number of formats

Improvements

- `ext2`, `ext3` and `ext4` images now generated by `mkfs.ext<X>` instead of `genext2fs`, to better support `ext3/ext4`
- Support for AXFS added
- ISO9660 support re-written, to support Grub2 and Isolinux as bootloaders, initramfs and pure ISO9660 scenarios
- Usage of `genimage` to generate complete SD card/MMC images in many `defconfigs`
- Ability to specify a custom script to run within the `fakeroot` environment when creating filesystem images
Reproducible builds

- Idea: get binary identical results for repeated builds of a given configuration
- Option `BR2_REPRODUCIBLE` added
- Various things already fixed:
  - Sets `SOURCE_DATE_EPOCH`, used by gcc and various packages
  - Date/time of files in the filesystem
  - Build date, user, host name in the Linux kernel build and BusyBox
  - Remove timestamps in Python `.pyc` files
  - ...
- A lot more remains to be done. Unfortunately, the developers who started this work are no longer active.
By far where most of the contributions go: updating existing packages and adding new packages

- Almost **1000 packages** added between 2014.05 and 2017.08

- Significant updates/additions
  - SELinux support
  - Qt 5.9 (including Qt WebEngine), Gtk 3.x, EFL updates
  - OpenCV 3.0
  - Kodi
  - Go, Mono
  - Python modules (many!), Perl modules, Erlang modules
  - Docker, aufs
  - System upgrade: SWupdate, RAUC
  - HW support: AMD Catalyst, Freescale i.MX, NVidia, TI
  - Apache, ClamAV, Dovecot, MariaDB, Nginx
  - Glib C++ stack: glibmm, atkmm, cairomm, gtkmm, etc.
  - ... and SuperTuxKart!
Run-time test infrastructure added in `support/testing/`

Test cases written in Python

Build a given Buildroot configuration, boot under QEMU, run commands and check results

Tests for filesystem images, packages, core functionality, init systems

```python
class TestDropbear(infra.basetest.BRTest):
    config = infra.basetest.BASIC_TOOLCHAIN_CONFIG + \
    """BR2_SYSTEM_DHCP="eth0"
BR2_PACKAGE_DROPBEAR=y
BR2_TARGET_ROOTFS_CPIO=y
# BR2_TARGET_ROOTFS_TAR is not set"
""

def test_run(self):
    img = os.path.join(self.builddir, "images", "rootfs.cpio")
    self.emulator.boot(arch="armv5",
                      kernel="builtin",
                      options=["-initrd", img,
                               "-net", "nic"])
    self.emulator.login("testpwd")
    cmd = "netstat -ltn 2>/dev/null | grep 0.0.0.0:22" \
    _, exit_code = self.emulator.run(cmd)
    self.assertEqual(exit_code, 0)
```
Testing: CI

- Already existing: http://autobuild.buildroot.org
  - Set of 50 architecture/toolchain configurations
  - Choose a random architecture/toolchain configuration, a random selection of packages, and build
  - Results reported on a Web page, e-mailed to the mailing list

- Improvements
  - All defconfigs are built on Gitlab CI
  - Run-time tests are executed on Gitlab CI
  - Preparation on autobuild.b.o to support testing multiples branches (master, next, LTS)
  - Notifications from autobuild.b.o sent to relevant developers
Tooling

- **DEVELOPERS file and associated get-developers tool**
  - Much like MAINTAINERS in the Linux kernel
  - Used when sending patches
  - Used to report build failures per package or per-architecture to the relevant developers

- check-package script to detect obvious mistakes in packages

- test-pkg to build test a package with a large number of architecture/toolchain configurations

- scanpypi script to generate Python packages
  - Connects to Pypi, analyzes the metadata, and produces a Buildroot package

---

**DEVELOPERS**

- N: Waldemar Brodkorb <wbx@openadk.org>
- F: arch/Config.in.bfin
- F: arch/Config.in.m68k
- F: arch/Config.in.or1k
- F: arch/Config.in.sparc
- F: package/glibc/
- F: package/mksh/
- F: package/uclibc/
- F: package/uclibc-ng-test/

**test-pkg**

- armv5-ctng-linux-gnueabi [ 1/49]: OK
- armv7-ctng-linux-gnueabihf [ 2/49]: OK
- br-aarch64-glibc [ 3/49]: OK
- br-armcortex-a32-glibc [ 4/49]: SKIPPED
- br-arm-generic [ 5/49]: OK
- br-arm-cortex-a9-glibc [ 6/49]: OK

...........

49 builds, 27 skipped, 0 build failed, 0 legal-info failed
Linux extensions infrastructure, to support building packages that need kernel patching: Xenomai, RTAI, specific drivers

Linux tools infrastructure, to build user-space tools part of the kernel tree: perf, gpio, iio, cpupower, tmon, self-tests

Complete revamp of the gettext handling, option BR2_SYSTEM_ENABLE_NLS to control native language support

Checks on the architecture of cross-compiled binaries, to detect packages that do not cross-compile to the correct architecture
Features on the radar

- **Git download cache**
  - Avoid re-cloning an entire Git repository every time the version/tag of a Git-fetched package is changed

- **Per-package out of tree build**
  - Avoids `rsync` when using `local` packages or `<pkg>_OVERRIDE_SRCDIR` and improves debugging experience
  - Avoids extracting the source code twice when building host and target variants

- **Top-level parallel build**
  - Building different packages in parallel
  - Requires per-package staging and host directories

- **Go and Meson package infrastructures**
Conclusion

- Active project
- LTS releases with security updates
- Relocatable SDK
- Rich and up-to-date package set
- Good and increasing testing effort
- Interesting new features on the roadmap
Questions? Suggestions? Comments?

Thomas Petazzoni
thomas.petazzoni@bootlin.com

Slides under CC-BY-SA 3.0