Porting new Code to RISC-V with Yocto/OpenEmbedded

Martin Maas (maas@cs.berkeley.edu)

1st RISC-V Workshop, January 15, 2015
Monterey, CA

Why we need a Linux Distribution

- To build an application for RISC-V, you need to:
  - Download and build the RISC-V toolchain + Linux
  - Download, patch and build application + dependencies
  - Create an image and run it in QEMU or on hardware
- Problems with this approach:
  - **Error-prone**: Easy to corrupt FS or get a step wrong
  - **Reproducibility**: Others can’t easily reuse your work
  - **Rigidity**: If a dependency changes, need to do it all over
- We need a Linux distribution!
  - Automatic **build process** with dependency tracking
  - Ability to distribute binary **packages and SDKs**

RISCV-POKY: A PORT OF THE YOCTO PROJECT

- We ported the **Yocto Project**
  - Official Linux Foundation Workgroup, supported by a large number of industry partners
  - Part I: **Collection of hundreds of recipes** (scripts that describe how to build packages for different platforms), shared with OpenEmbedded project
  - Part II: **Bitbake, a parallel build system** that takes recipes and fetches, patches, cross-compiles and produces packages (RPM/DEB), images, SDKs, etc.
- Focus on build process and customizability

GETTING STARTED WITH RISCV-POKY

- **Let's build a full Linux system** including the GCC toolchain, Linux, QEMU + a large set of packages (including `bash`, `ssh`, `python`, `perl`, `apt`, `wget`,...)
- **Step I**: Clone riscv-poky:

```
git clone git@github.com:ucb-bar/riscv-poky.git
```

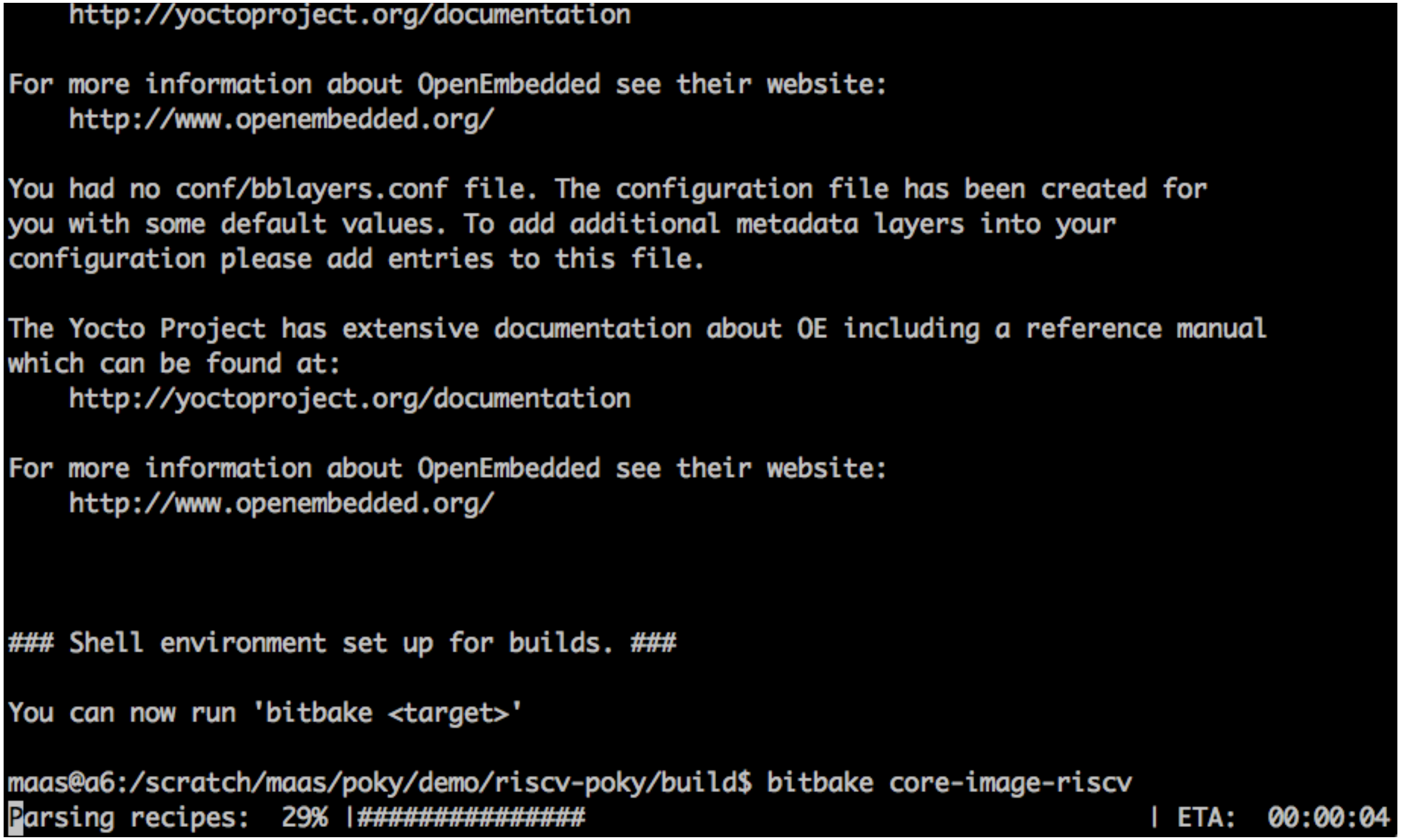
- **Step II**: Set up the build system:

```
source oe-init-build-env
```

- **Step III**: Build an image (may take hours!):

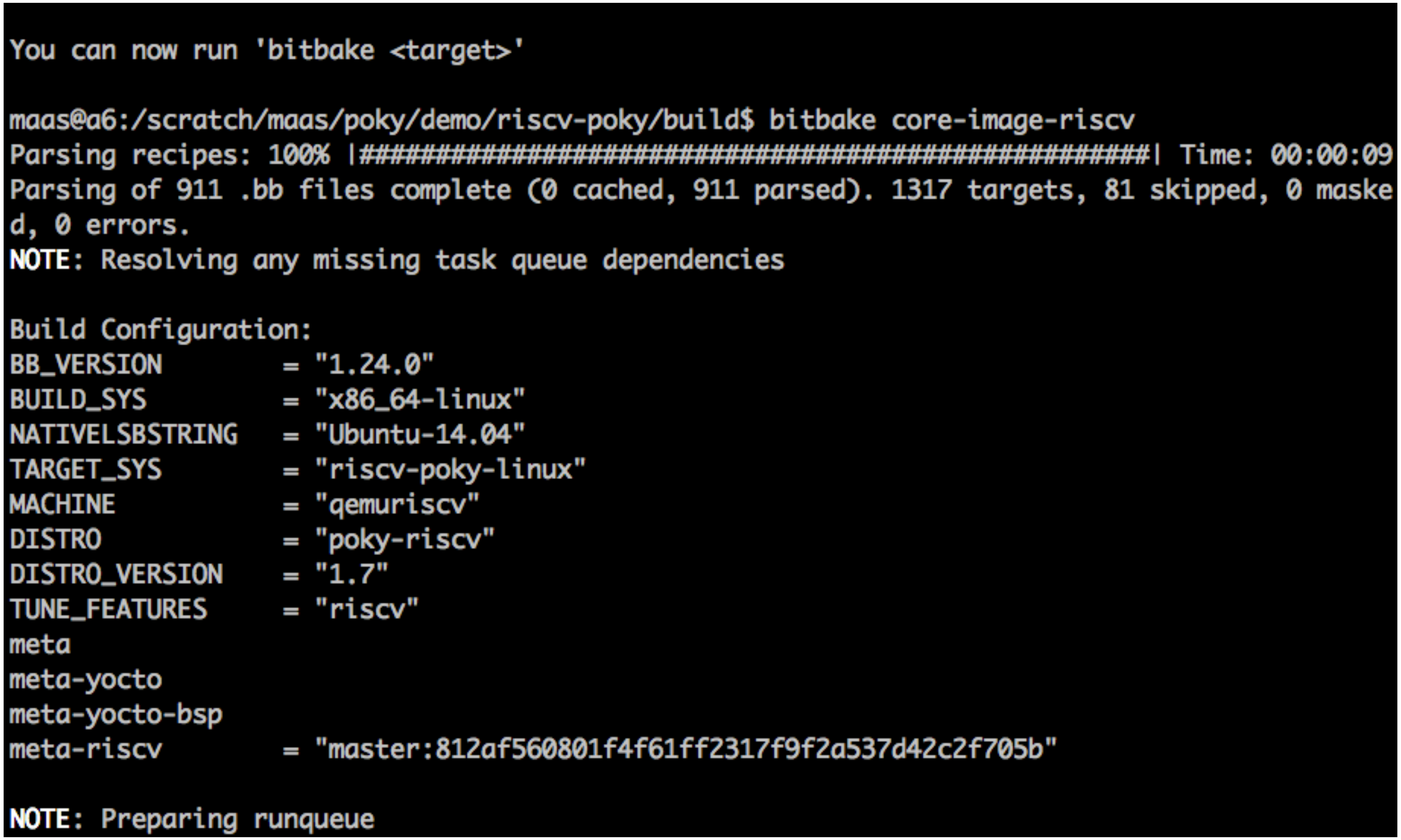
```
bitbake core-image-riscv
```

Build an Image (1/3)



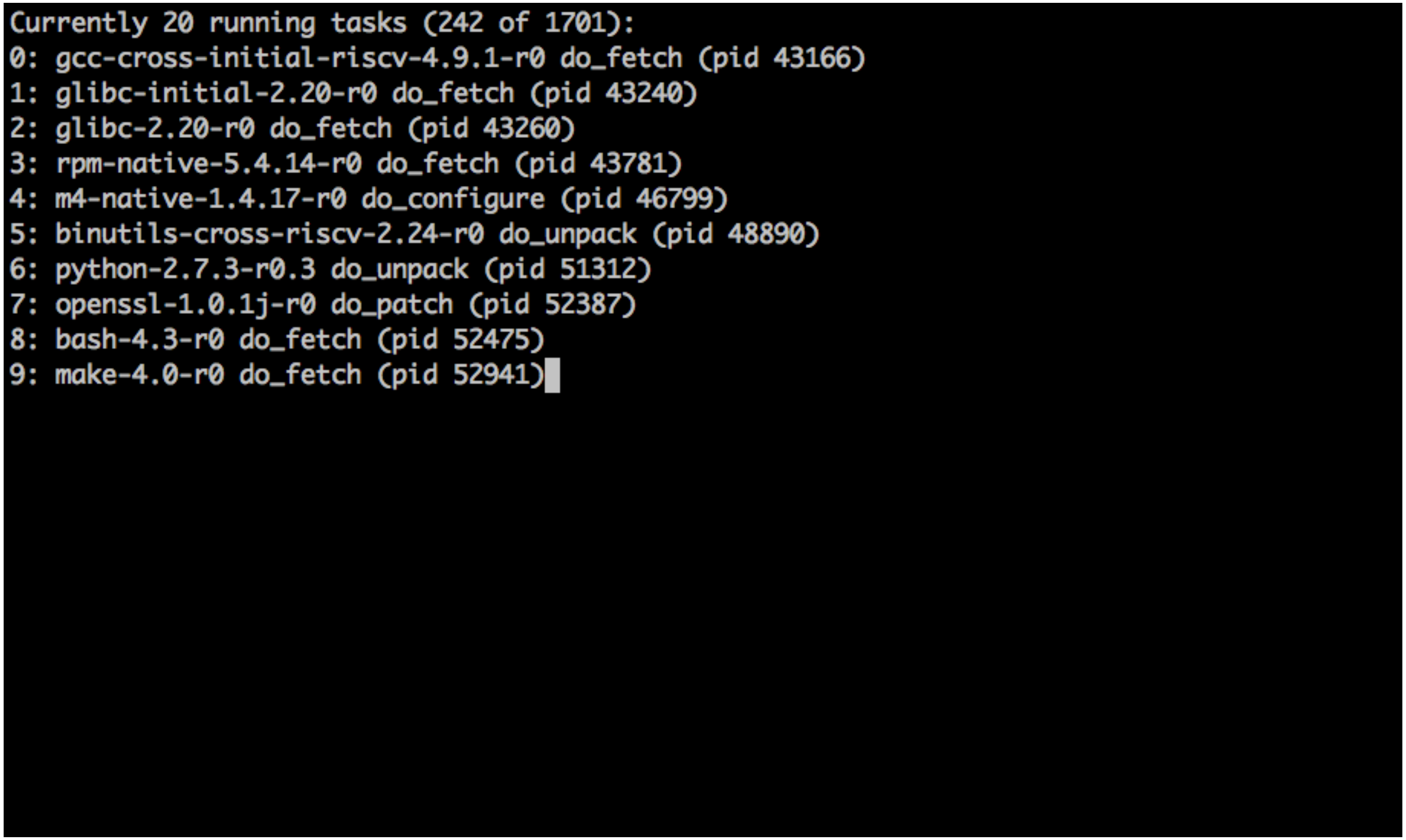
```
    http://yoctoproject.org/documentation

For more information about OpenEmbedded see their website:
    http://www.openembedded.org/

You had no conf/bblayers.conf file. The configuration file has been created for
you with some default values. To add additional metadata layers into your
configuration please add entries to this file.

The Yocto Project has extensive documentation about OE including a reference manual
which can be found at:
    http://yoctoproject.org/documentation

For more information about OpenEmbedded see their website:
    http://www.openembedded.org/



### Shell environment set up for builds. ###

You can now run 'bitbake <target>'

maas@a6:/scratch/maas/poky/demo/riscv-poky/build$ bitbake core-image-riscv
Parsing recipes:  29% |###############                                    | ETA:  00:00:04
```

Build an Image (2/3)



```
You can now run 'bitbake <target>'

maas@a6:/scratch/maas/poky/demo/riscv-poky/build$ bitbake core-image-riscv
Parsing recipes: 100% |#####################################################| Time: 00:00:09
Parsing of 911 .bb files complete (0 cached, 911 parsed). 1317 targets, 81 skipped, 0 maske
d, 0 errors.
NOTE: Resolving any missing task queue dependencies

Build Configuration:
BB_VERSION        = "1.24.0"
BUILD_SYS         = "x86_64-linux"
NATIVELSBSTRING   = "Ubuntu-14.04"
TARGET_SYS        = "riscv-poky-linux"
MACHINE           = "qemuriscv"
DISTRO            = "poky-riscv"
DISTRO_VERSION    = "1.7"
TUNE_FEATURES     = "riscv"
meta
meta-yocto
meta-yocto-bsp
meta-riscv        = "master:812af560801f4f61ff2317f9f2a537d42c2f705b"

NOTE: Preparing runqueue
```

BUILD AN IMAGE (3/3)

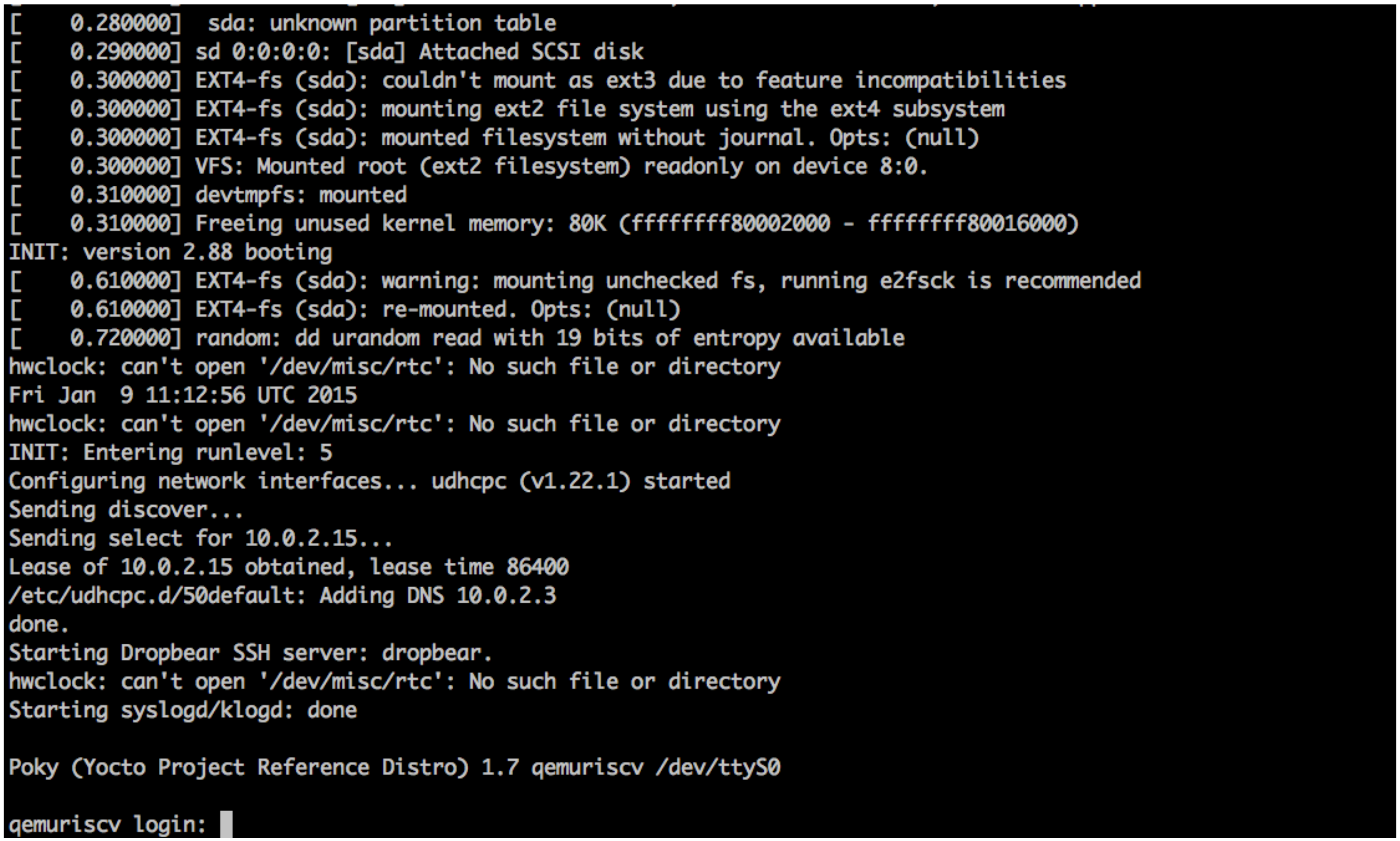


```
Currently 20 running tasks (242 of 1701):
0: gcc-cross-initial-riscv-4.9.1-r0 do_fetch (pid 43166)
1: glibc-initial-2.20-r0 do_fetch (pid 43240)
2: glibc-2.20-r0 do_fetch (pid 43260)
3: rpm-native-5.4.14-r0 do_fetch (pid 43781)
4: m4-native-1.4.17-r0 do_configure (pid 46799)
5: binutils-cross-riscv-2.24-r0 do_unpack (pid 48890)
6: python-2.7.3-r0.3 do_unpack (pid 51312)
7: openssl-1.0.1j-r0 do_patch (pid 52387)
8: bash-4.3-r0 do_fetch (pid 52475)
9: make-4.0-r0 do_fetch (pid 52941)
```

GETTING STARTED WITH RISCV-POKY

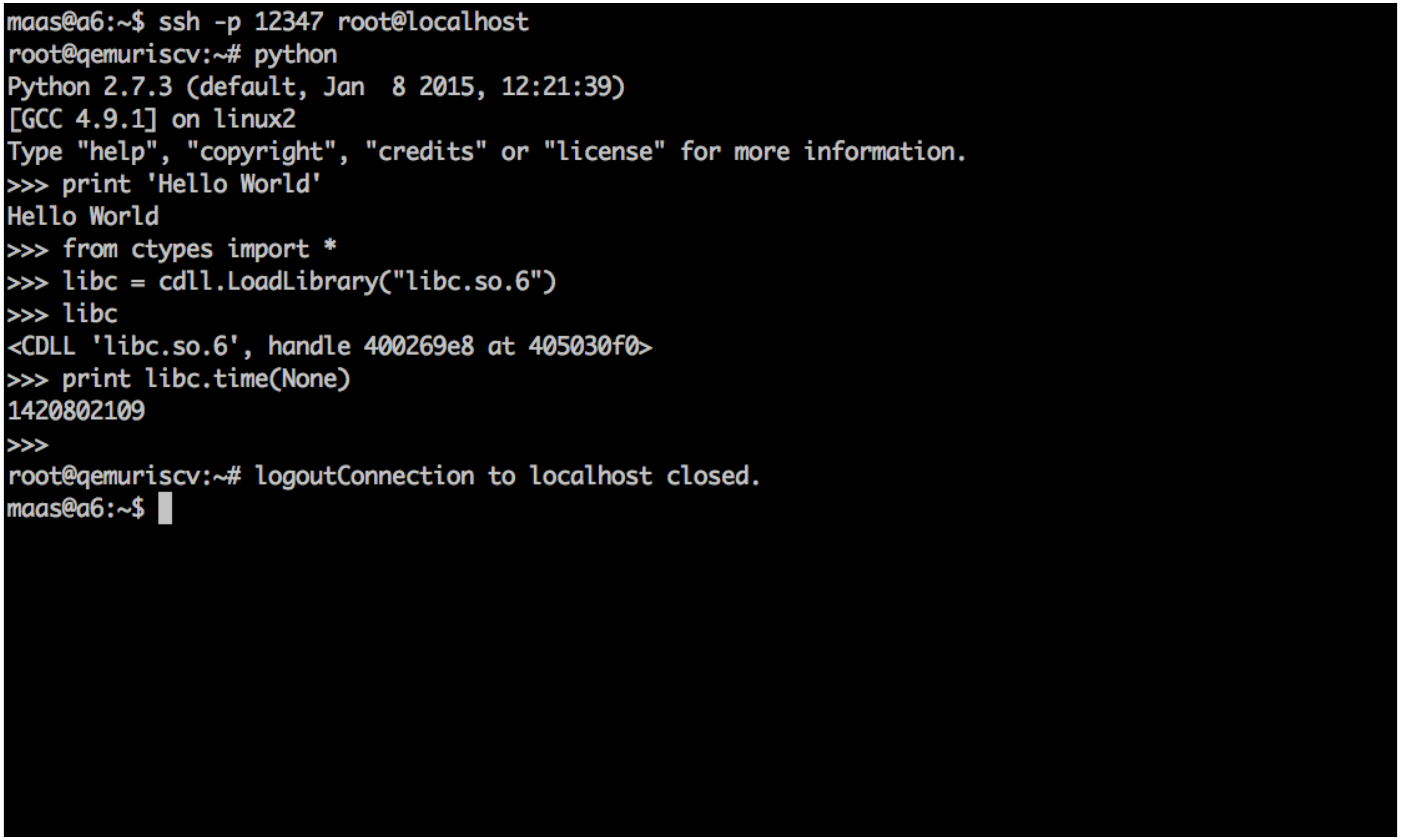
- **Let's build a full Linux system** including the GCC toolchain, Linux, QEMU + a large set of packages (including `bash`, `ssh`, `python`, `perl`, `apt`, `wget`,…)
- **Step I**: Clone riscv-poky:
  ```
  git clone git@github.com:ucb-bar/riscv-poky.git
  ```
- **Step II**: Set up the build system:
  ```
  source oe-init-build-env
  ```
- **Step III**: Build an image (may take hours!):
  ```
  bitbake core-image-riscv
  ```
- **Step IV**: Run in QEMU (and SSH into it):
  ```
  runqemu qemuriscv nographic slirp
  hostfwd="tcp::12347-:22”
  ```

Run in QEMU (1/2)



```
[    0.280000]  sda: unknown partition table
[    0.290000] sd 0:0:0:0: [sda] Attached SCSI disk
[    0.300000] EXT4-fs (sda): couldn't mount as ext3 due to feature incompatibilities
[    0.300000] EXT4-fs (sda): mounting ext2 file system using the ext4 subsystem
[    0.300000] EXT4-fs (sda): mounted filesystem without journal. Opts: (null)
[    0.300000] VFS: Mounted root (ext2 filesystem) readonly on device 8:0.
[    0.310000] devtmpfs: mounted
[    0.310000] Freeing unused kernel memory: 80K (ffffffff80002000 - ffffffff80016000)
INIT: version 2.88 booting
[    0.610000] EXT4-fs (sda): warning: mounting unchecked fs, running e2fsck is recommended
[    0.610000] EXT4-fs (sda): re-mounted. Opts: (null)
[    0.720000] random: dd urandom read with 19 bits of entropy available
hwclock: can't open '/dev/misc/rtc': No such file or directory
Fri Jan  9 11:12:56 UTC 2015
hwclock: can't open '/dev/misc/rtc': No such file or directory
INIT: Entering runlevel: 5
Configuring network interfaces... udhcpc (v1.22.1) started
Sending discover...
Sending select for 10.0.2.15...
Lease of 10.0.2.15 obtained, lease time 86400
/etc/udhcpc.d/50default: Adding DNS 10.0.2.3
done.
Starting Dropbear SSH server: dropbear.
hwclock: can't open '/dev/misc/rtc': No such file or directory
Starting syslogd/klogd: done

Poky (Yocto Project Reference Distro) 1.7 qemuriscv /dev/ttyS0

qemuriscv login: 
```

Run in QEMU (2/2)



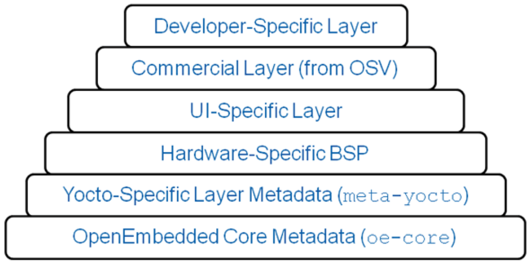
```
maas@a6:~$ ssh -p 12347 root@localhost
root@qemuriscv:~# python
Python 2.7.3 (default, Jan  8 2015, 12:21:39)
[GCC 4.9.1] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> print 'Hello World'
Hello World
>>> from ctypes import *
>>> libc = cdll.LoadLibrary("libc.so.6")
>>> libc
<CDLL 'libc.so.6', handle 400269e8 at 405030f0>
>>> print libc.time(None)
1420802109
>>>
root@qemuriscv:~# logoutConnection to localhost closed.
maas@a6:~$
```

DECIDING WHAT TO BUILD

- Decide what should go into the image:
  - Edit `meta-riscv/images/core-image-riscv.bb`
  - **Add packages** to `IMAGE_INSTALL` list, e.g.
    `IMAGE_INSTALL += “python python-ctypes”`
- Build packages for use with package-manager:
  - They’re already there: `build/tmp/deploy/rpm/riscv`
- Configure build by editing `conf/local.conf`
  - **Select init system**: We use SysV for now, systemd is available in Yocto
  - Switch **target machine** from `qemuriscv` and `riscv` machine to target real hardware instead of QEMU
  - Can use externally built toolchain

How to add your own Recipe

- Yocto is based on layers:
  - Recipes arranged in directory tree
  - Layers contain .bb files (recipes), .bbappend files (to adapt or extend lower layers)
  - We have a layer (`meta-riscv`) for anything RISC-V


Developer-Specific Layer
Commercial Layer (from OSV)
UI-Specific Layer
Hardware-Specific BSP
Yocto-Specific Layer Metadata (meta-yocto)
OpenEmbedded Core Metadata (oe-core)

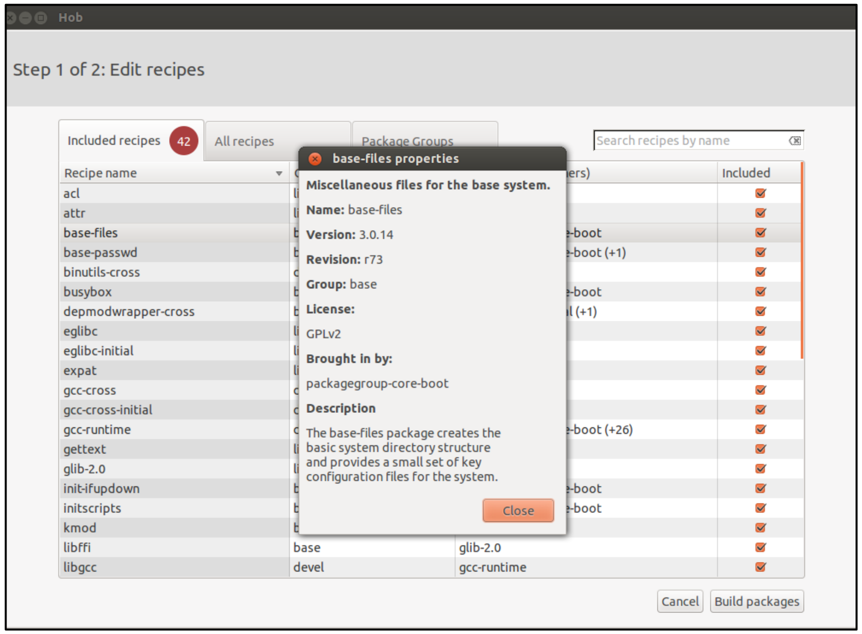
- If you want to add a recipe to RISC-V:
  - **Scenario I**: Recipe already exists in some layer. Add a .bbappend file, patches, etc. to `meta-riscv`
  - **Scenario II**: There is no recipe yet. Add your own recipe (i.e., source location, etc.) to `meta-riscv`

-- Most importantly: Submit a pull request! --

EXAMPLE: WGET_1.15.BB (EDITED)

```
SUMMARY = "Console URL download utility supporting HTTP, FTP, etc"
HOMEPAGE = "https://www.gnu.org/software/wget/"
SECTION = "console/network"
LICENSE = "GPLv3"
LIC_FILES_CHKSUM = "file://COPYING;md5=d32239bcb673463ab874e80d47fae504"
DEPENDS = "openssl zlib libpcre”

inherit autotools gettext texinfo update-alternatives

EXTRA_OECONF = "--enable-ipv6 --with-libssl-prefix=${STAGING_DIR_HOST} \
                --with-ssl=openssl --disable-rpath [...]”

ALTERNATIVE_${PN} = "wget"
ALTERNATIVE_PRIORITY = "100”

SRC_URI = "${GNU_MIRROR}/wget/wget-${PV}.tar.gz \
           file://fix_makefile.patch”

SRC_URI[md5sum] = "506df41295afc6486662cc47470b4618”
SRC_URI[sha256sum] = "52126be8cf1bddd7536886e74c053ad7d0ed2aa89b4b[...]fcd”
```

Some Additional Yocto Features



Hob

Step 1 of 2: Edit recipes

Included recipes 42 | All recipes | Package Groups | Search recipes by name

Recipe name	C	ers)	Included
acl	li		✓
attr	li		✓
base-files	b	-boot	✓
base-passwd	b	-boot (+1)	✓
binutils-cross	c		✓
busybox	b	-boot	✓
depmodwrapper-cross	b	l (+1)	✓
eglibc	li		✓
eglibc-initial	li		✓
expat	li		✓
gcc-cross	c		✓
gcc-cross-initial	c		✓
gcc-runtime	c	-boot (+26)	✓
gettext	li		✓
glib-2.0	li		✓
init-ifupdown	b	-boot	✓
initscripts	b	-boot	✓
kmod	b		✓
libffi	base	glib-2.0	✓
libgcc	devel	gcc-runtime	✓

base-files properties

Miscellaneous files for the base system.

Name: base-files

Version: 3.0.14

Revision: r73

Group: base

License:

GPLv2

Brought in by:

packagegroup-core-boot

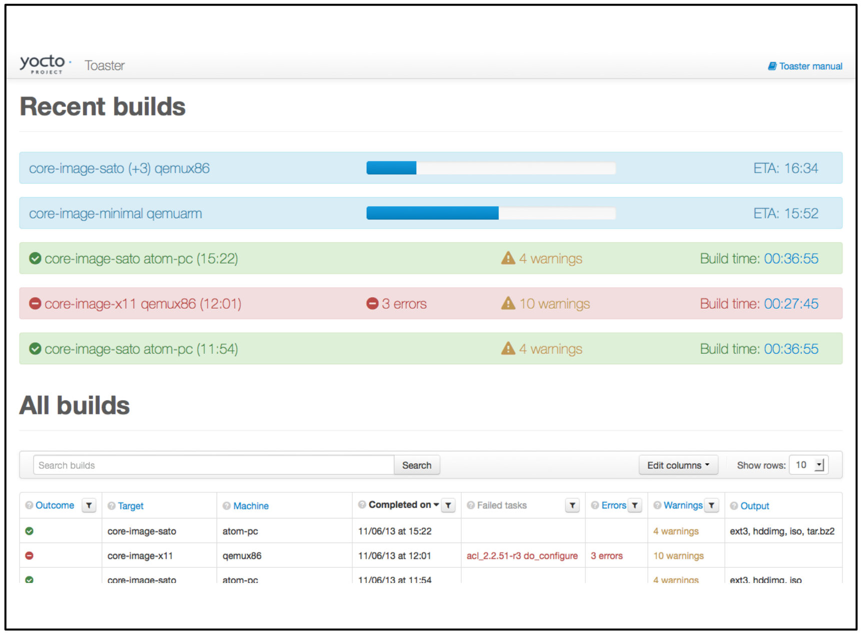
Description

The base-files package creates the basic system directory structure and provides a small set of key configuration files for the system.

Close

Cancel | Build packages

Hob: GUI to control Bitbake



yocto PROJECT · Toaster | Toaster manual

Recent builds

core-image-sato (+3) qemux86			ETA: 16:34
core-image-minimal qemuarm			ETA: 15:52
✓ core-image-sato atom-pc (15:22)		4 warnings	Build time: 00:36:55
● core-image-x11 qemux86 (12:01)	● 3 errors	10 warnings	Build time: 00:27:45
✓ core-image-sato atom-pc (11:54)		4 warnings	Build time: 00:36:55

All builds

Search builds | Search | Edit columns | Show rows: 10

Outcome	Target	Machine	Completed on	Failed tasks	Errors	Warnings	Output
✓	core-image-sato	atom-pc	11/06/13 at 15:22			4 warnings	ext3, hddimg, iso, tar.bz2
●	core-image-x11	qemux86	11/06/13 at 12:01	acl_2.2.51-r3 do_configure	3 errors	10 warnings	
✓	core-image-sato	atom-pc	11/06/13 at 11:54			4 warnings	ext3, hddimg, iso

Toaster: Build Server

Yocto provides a lot of industry-strength features:
QA, checking license files, central build repositories, etc.

What's Next?

- We also have a **Gentoo port** (by Palmer Dabbelt)
  - Bitbake is based on Portage (Gentoo build tool)
  - Will integrate submitted poky recipes into Gentoo
- **Add new packages:**
  libffi, riscv-llvm
- **Port more software:**
  Java, X Server, Gnome
- **Building and distributing binary SDKs**
  - Make it easier to get started with RISC-V
- **Official RISC-V package repository**
  - Distribute RISC-V Linux images with package manager

Fork me on GitHub

Clone riscv-poky today!

http://github.com/ucb-bar/riscv-poky