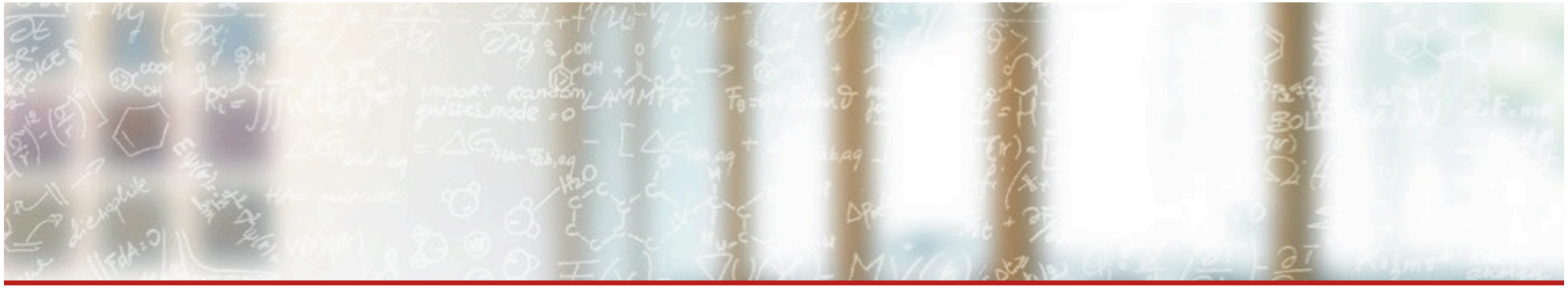
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

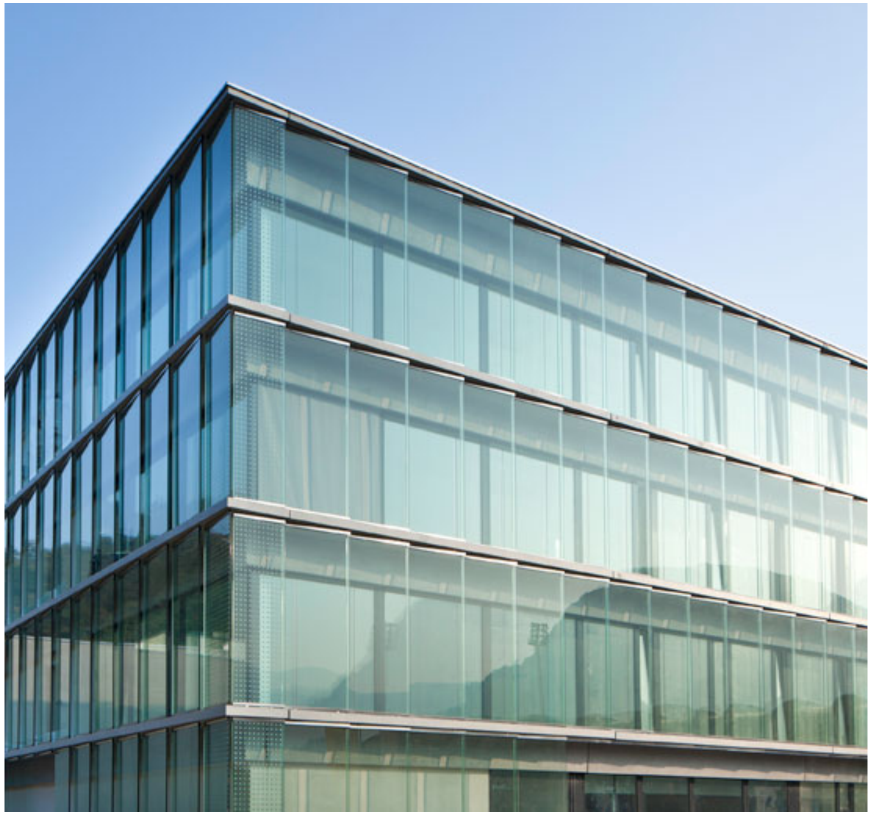


Approaching EasyBuild with an Intel toolchain

EasyBuild Workshop 8th-9th September

Luca Marsella, CSCS

Outline



- Motivation: why using EasyBuild?
- Get it started on your platform
- Building scientific applications
- Resolving the dependecies
- How to setup an Intel toolchain

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

Motivation: Why using EasyBuild?

Motivation: Why using EasyBuild?

EasyBuild:

- an open source python framework to **build** scientific software
- makes the maintenance of **multiple software** deployments easy
- **automatically** downloads, compiles and installs software packages
- takes care of **resolving dependencies** and **creating modulefiles** too
- successfully adopted by many HPC centres, on **Cray systems** as well
- **CSCS** has already experimented this tool **successfully** on few platforms

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

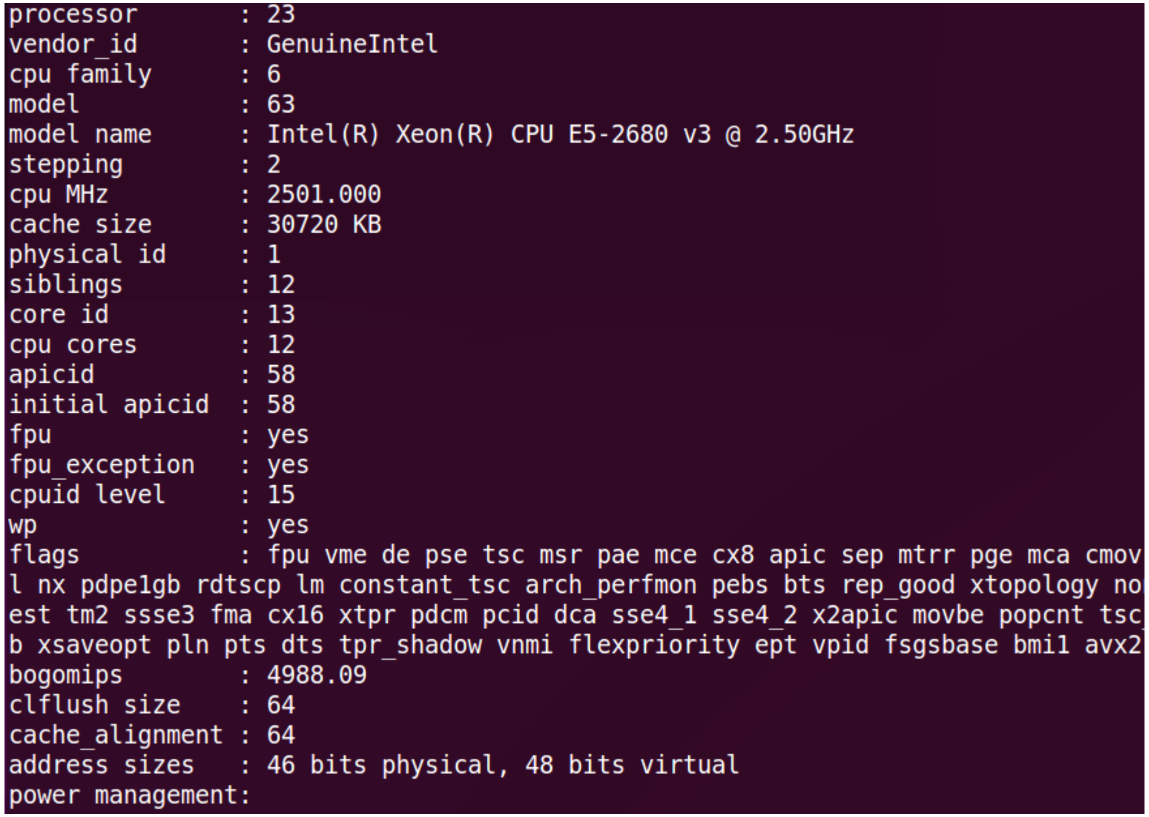
ETH zürich

Get it started on your platform

Start from scratch

Test cluster @ CSCS

2 x 12 core Intel Haswell CPU



```
processor       : 23
vendor_id       : GenuineIntel
cpu family      : 6
model           : 63
model name      : Intel(R) Xeon(R) CPU E5-2680 v3 @ 2.50GHz
stepping        : 2
cpu MHz         : 2501.000
cache size      : 30720 KB
physical id     : 1
siblings        : 12
core id         : 13
cpu cores       : 12
apicid          : 58
initial apicid  : 58
fpu             : yes
fpu_exception   : yes
cpuid level     : 15
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
l nx pdpe1gb rdtscp lm constant_tsc arch_perfmon pebs bts rep_good xtopology no
est tm2 ssse3 fma cx16 xtpr pdcm pcid dca sse4_1 sse4_2 x2apic movbe popcnt tsc
b xsaveopt pln pts dts tpr_shadow vnmi flexpriority ept vpid fsgsbase bmi1 avx2
bogomips        : 4988.09
clflush size    : 64
cache_alignment : 64
address sizes   : 46 bits physical, 48 bits virtual
power management:
```

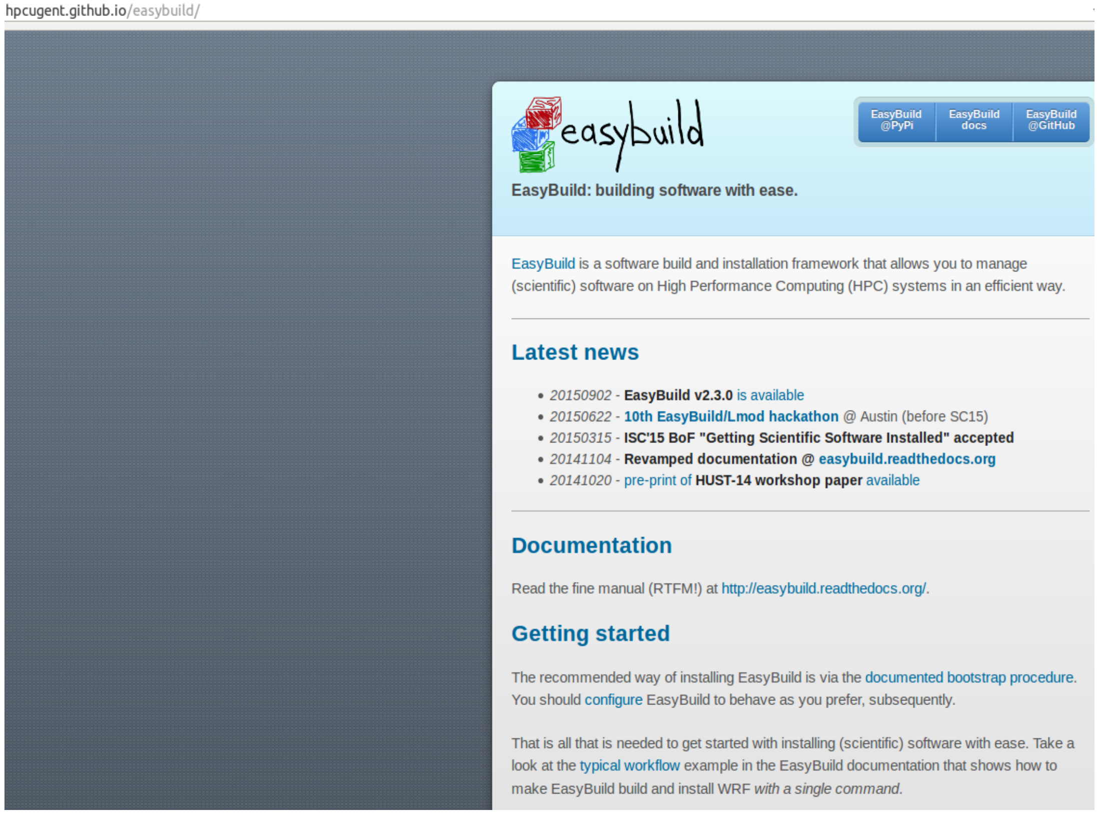
Easybuild @ GitHub : bootstrap!

Step by step:

1. download the bootstrap script: **bootstrap_eb.py**

2. **./bootstrap_eb.py $PREFIX**
   1. download easy_install
   2. download and install EasyBuild
   3. install EasyBuild in $PREFIX

3. EasyBuild module available in **$PREFIX/modules/all/EasyBuild**



EasyBuild: building software with ease.

EasyBuild is a software build and installation framework that allows you to manage (scientific) software on High Performance Computing (HPC) systems in an efficient way.

Latest news

- *20150902* - **EasyBuild v2.3.0** is available
- *20150622* - **10th EasyBuild/Lmod hackathon** @ Austin (before SC15)
- *20150315* - **ISC'15 BoF "Getting Scientific Software Installed" accepted**
- *20141104* - **Revamped documentation @ easybuild.readthedocs.org**
- *20141020* - pre-print of **HUST-14 workshop paper** available

Documentation

Read the fine manual (RTFM!) at http://easybuild.readthedocs.org/.

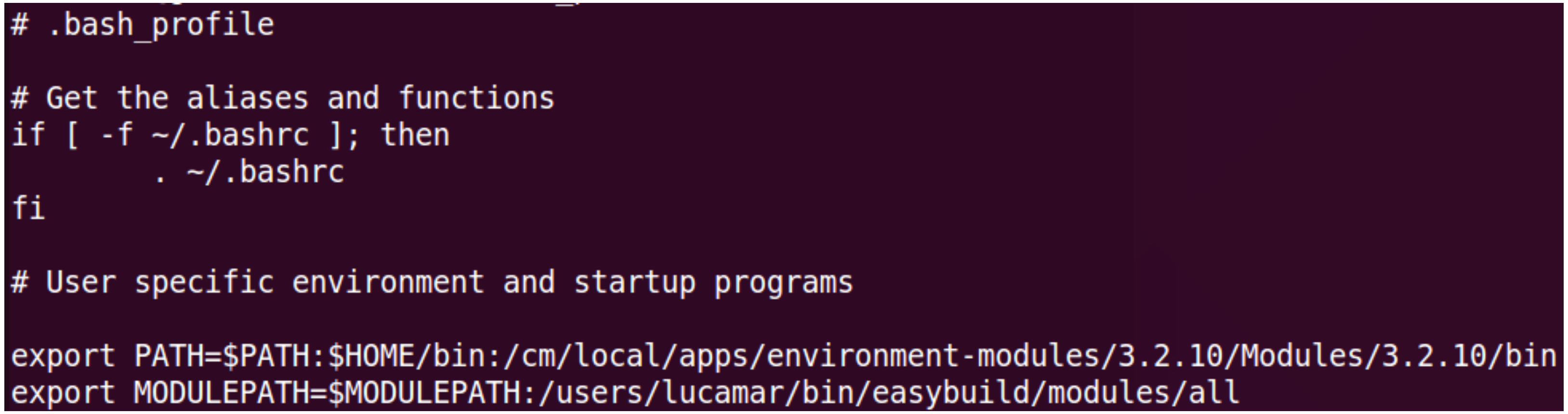
Getting started

The recommended way of installing EasyBuild is via the documented bootstrap procedure. You should configure EasyBuild to behave as you prefer, subsequently.

That is all that is needed to get started with installing (scientific) software with ease. Take a look at the typical workflow example in the EasyBuild documentation that shows how to make EasyBuild build and install WRF *with a single command.*

Requirements

- Linux (or OS X)
- Python version >= 2.6
- modules tool: e.g. Tcl modules
- module command/script (**modulecmd**, modulecmd.tcl or lmod) must be available via $PATH



```
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi

# User specific environment and startup programs

export PATH=$PATH:$HOME/bin:/cm/local/apps/environment-modules/3.2.10/Modules/3.2.10/bin
export MODULEPATH=$MODULEPATH:/users/lucamar/bin/easybuild/modules/all
```

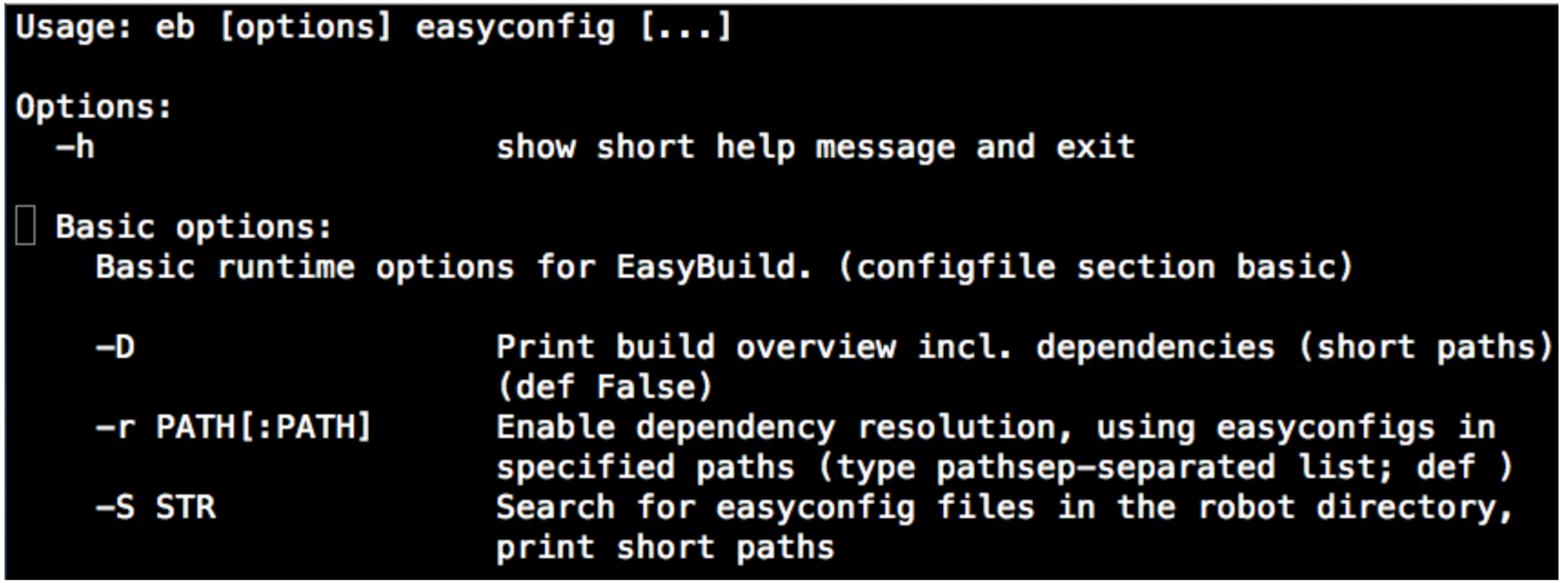
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

Building scientific applications

Search for packages and dependencies

Looking for software packages available in the repository: start with the help screen

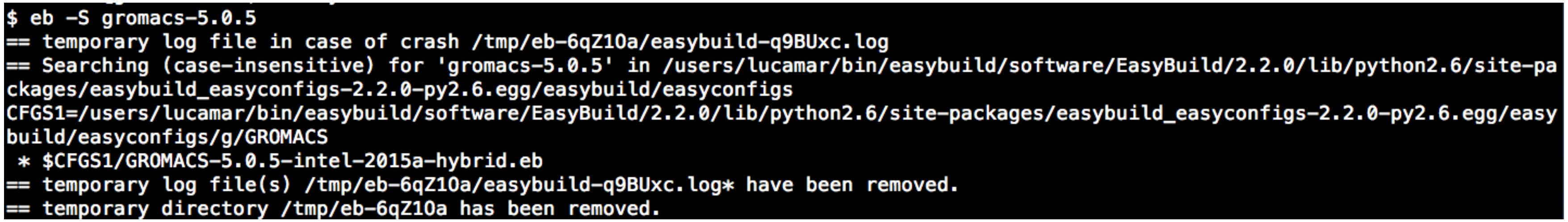


```
Usage: eb [options] easyconfig [...]

Options:
  -h                        show short help message and exit

□ Basic options:
    Basic runtime options for EasyBuild. (configfile section basic)

    -D                      Print build overview incl. dependencies (short paths)
                            (def False)
    -r PATH[:PATH]          Enable dependency resolution, using easyconfigs in
                            specified paths (type pathsep-separated list; def )
    -S STR                  Search for easyconfig files in the robot directory,
                            print short paths
```

Search for packages and dependencies

Example:

- search for the software Gromacs (version 5.0.5) using **eb –S**
- discover that Gromacs-5.0.5 can be installed with the toolchain intel-2015a
- have a look at the configuration file (easyconfig) provided
- check the dependencies needed in order to build the software



```
$ eb -S gromacs-5.0.5
== temporary log file in case of crash /tmp/eb-6qZ10a/easybuild-q9BUxc.log
== Searching (case-insensitive) for 'gromacs-5.0.5' in /users/lucamar/bin/easybuild/software/EasyBuild/2.2.0/lib/python2.6/site-pa
ckages/easybuild_easyconfigs-2.2.0-py2.6.egg/easybuild/easyconfigs
CFGS1=/users/lucamar/bin/easybuild/software/EasyBuild/2.2.0/lib/python2.6/site-packages/easybuild_easyconfigs-2.2.0-py2.6.egg/easy
build/easyconfigs/g/GROMACS
 * $CFGS1/GROMACS-5.0.5-intel-2015a-hybrid.eb
== temporary log file(s) /tmp/eb-6qZ10a/easybuild-q9BUxc.log* have been removed.
== temporary directory /tmp/eb-6qZ10a has been removed.
```

Resolving the dependencies

Gromacs-5.0.5 dependencies

EasyBuild flags -D and -r combined:

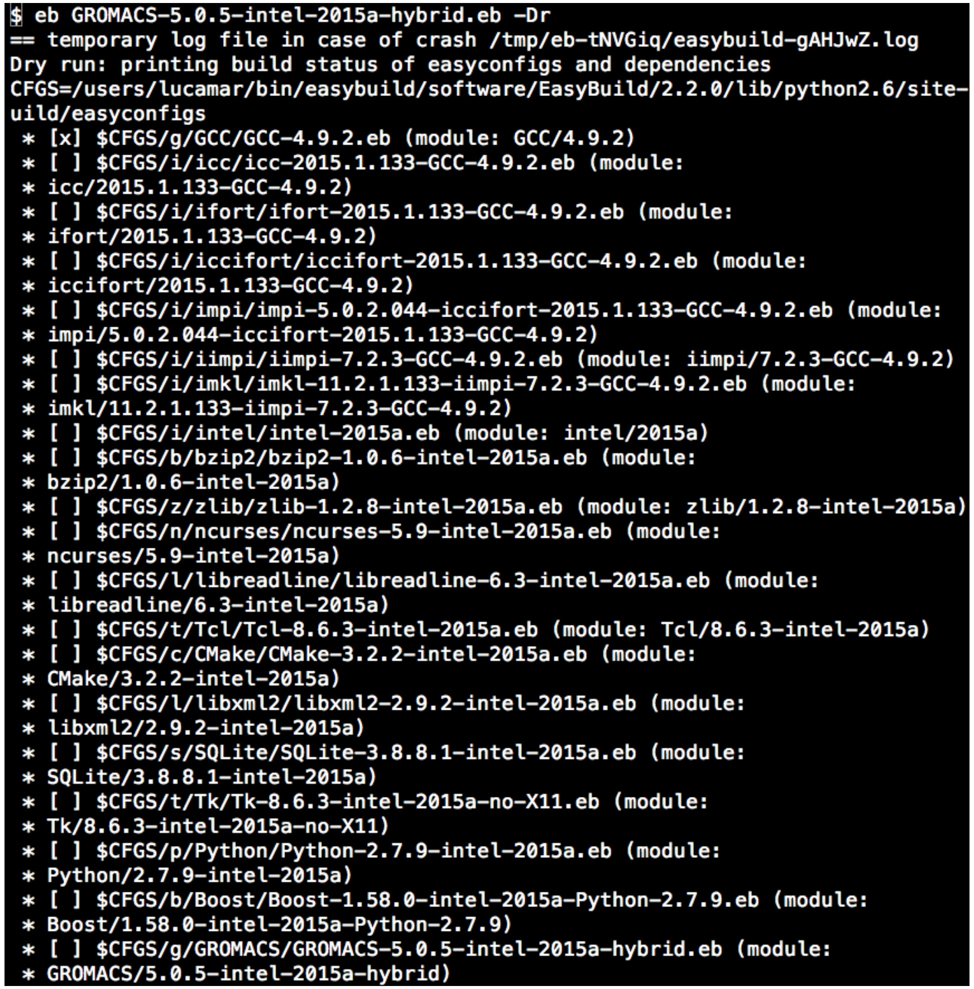
-D / --dry-run-short (print dependencies)

-r / --robot (resolve dependencies)

N.B:“Using --robot is particularly useful for software packages that have an **extensive list of dependencies**, or when reinstalling software using a different compiler toolchain”

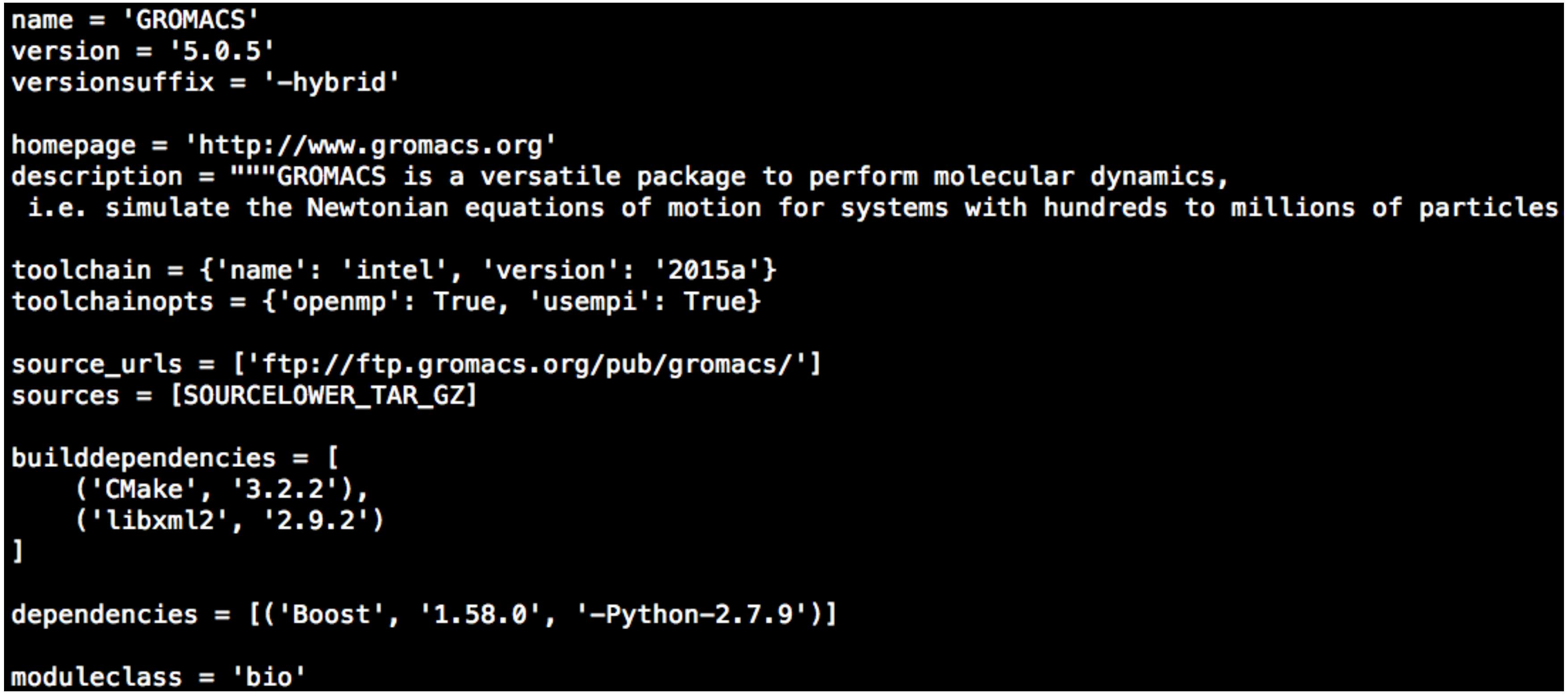
Build using a new toolchain version:
--try-toolchain with **--robot**

E.g.: **eb GROMACS-5.0.5-intel-2015a-hybrid.eb -r --try-toolchain-version=2015b**



```
$ eb GROMACS-5.0.5-intel-2015a-hybrid.eb -Dr
== temporary log file in case of crash /tmp/eb-tNVGiq/easybuild-gAHJwZ.log
Dry run: printing build status of easyconfigs and dependencies
CFGS=/users/lucamar/bin/easybuild/software/EasyBuild/2.2.0/lib/python2.6/site-
uild/easyconfigs
 * [x] $CFGS/g/GCC/GCC-4.9.2.eb (module: GCC/4.9.2)
 * [ ] $CFGS/i/icc/icc-2015.1.133-GCC-4.9.2.eb (module:
 * icc/2015.1.133-GCC-4.9.2)
 * [ ] $CFGS/i/ifort/ifort-2015.1.133-GCC-4.9.2.eb (module:
 * ifort/2015.1.133-GCC-4.9.2)
 * [ ] $CFGS/i/iccifort/iccifort-2015.1.133-GCC-4.9.2.eb (module:
 * iccifort/2015.1.133-GCC-4.9.2)
 * [ ] $CFGS/i/impi/impi-5.0.2.044-iccifort-2015.1.133-GCC-4.9.2.eb (module:
 * impi/5.0.2.044-iccifort-2015.1.133-GCC-4.9.2)
 * [ ] $CFGS/i/iimpi/iimpi-7.2.3-GCC-4.9.2.eb (module: iimpi/7.2.3-GCC-4.9.2)
 * [ ] $CFGS/i/imkl/imkl-11.2.1.133-iimpi-7.2.3-GCC-4.9.2.eb (module:
 * imkl/11.2.1.133-iimpi-7.2.3-GCC-4.9.2)
 * [ ] $CFGS/i/intel/intel-2015a.eb (module: intel/2015a)
 * [ ] $CFGS/b/bzip2/bzip2-1.0.6-intel-2015a.eb (module:
 * bzip2/1.0.6-intel-2015a)
 * [ ] $CFGS/z/zlib/zlib-1.2.8-intel-2015a.eb (module: zlib/1.2.8-intel-2015a)
 * [ ] $CFGS/n/ncurses/ncurses-5.9-intel-2015a.eb (module:
 * ncurses/5.9-intel-2015a)
 * [ ] $CFGS/l/libreadline/libreadline-6.3-intel-2015a.eb (module:
 * libreadline/6.3-intel-2015a)
 * [ ] $CFGS/t/Tcl/Tcl-8.6.3-intel-2015a.eb (module: Tcl/8.6.3-intel-2015a)
 * [ ] $CFGS/c/CMake/CMake-3.2.2-intel-2015a.eb (module:
 * CMake/3.2.2-intel-2015a)
 * [ ] $CFGS/l/libxml2/libxml2-2.9.2-intel-2015a.eb (module:
 * libxml2/2.9.2-intel-2015a)
 * [ ] $CFGS/s/SQLite/SQLite-3.8.8.1-intel-2015a.eb (module:
 * SQLite/3.8.8.1-intel-2015a)
 * [ ] $CFGS/t/Tk/Tk-8.6.3-intel-2015a-no-X11.eb (module:
 * Tk/8.6.3-intel-2015a-no-X11)
 * [ ] $CFGS/p/Python/Python-2.7.9-intel-2015a.eb (module:
 * Python/2.7.9-intel-2015a)
 * [ ] $CFGS/b/Boost/Boost-1.58.0-intel-2015a-Python-2.7.9.eb (module:
 * Boost/1.58.0-intel-2015a-Python-2.7.9)
 * [ ] $CFGS/g/GROMACS/GROMACS-5.0.5-intel-2015a-hybrid.eb (module:
 * GROMACS/5.0.5-intel-2015a-hybrid)
```

Easyconfig files



```
name = 'GROMACS'
version = '5.0.5'
versionsuffix = '-hybrid'

homepage = 'http://www.gromacs.org'
description = """GROMACS is a versatile package to perform molecular dynamics,
 i.e. simulate the Newtonian equations of motion for systems with hundreds to millions of particles

toolchain = {'name': 'intel', 'version': '2015a'}
toolchainopts = {'openmp': True, 'usempi': True}

source_urls = ['ftp://ftp.gromacs.org/pub/gromacs/']
sources = [SOURCELOWER_TAR_GZ]

builddependencies = [
    ('CMake', '3.2.2'),
    ('libxml2', '2.9.2')
]

dependencies = [('Boost', '1.58.0', '-Python-2.7.9')]

moduleclass = 'bio'
```

How to setup an Intel toolchain

List of toolchains

A number of toolchains is provided:
eb --list-toolchains

Brief mnemonic names:

- goolf

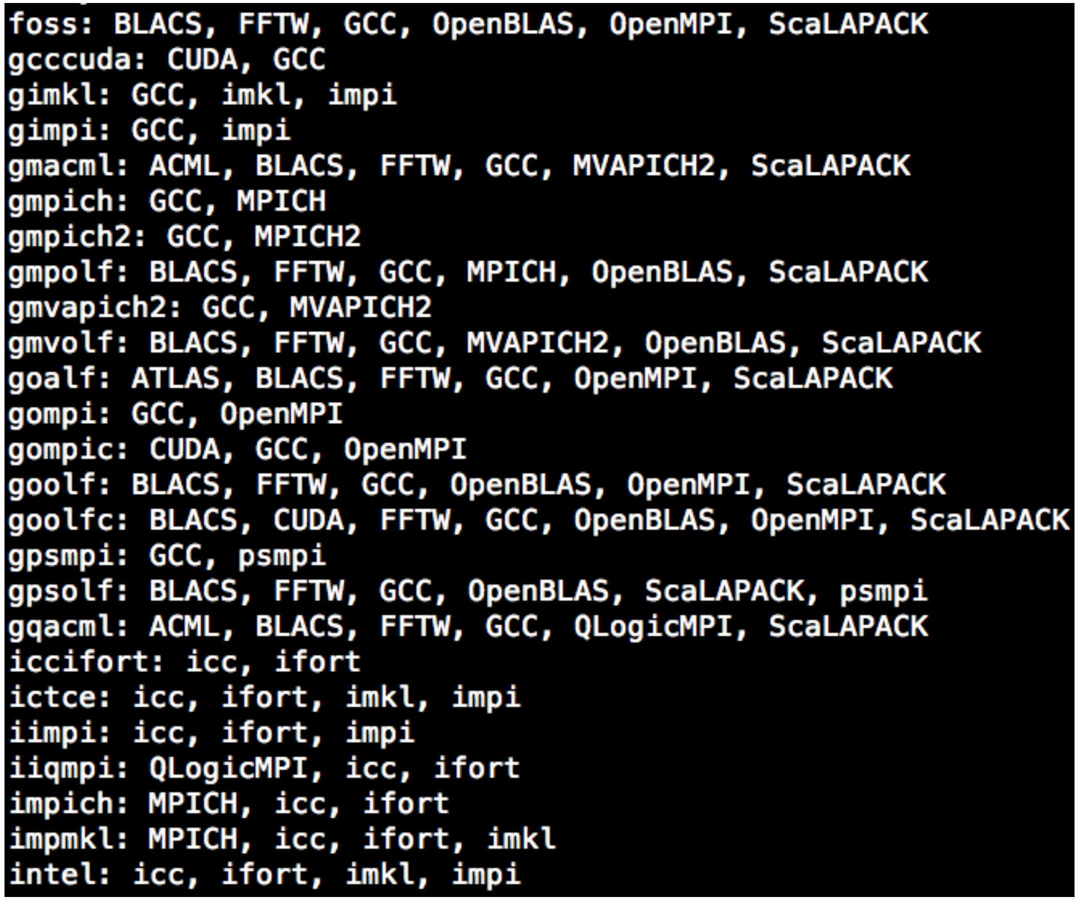
GCC, OpenMPI, OpenBLASLAPACK, FFTW and ScaLAPACK

- iimpi

icc/ifort, impi

Full list on EasyBuild documentation:
http://easybuild.readthedocs.org/en/latest/eb_list_toolchains.html#toolchains-table

Redundancies? E.g.: ictce and intel

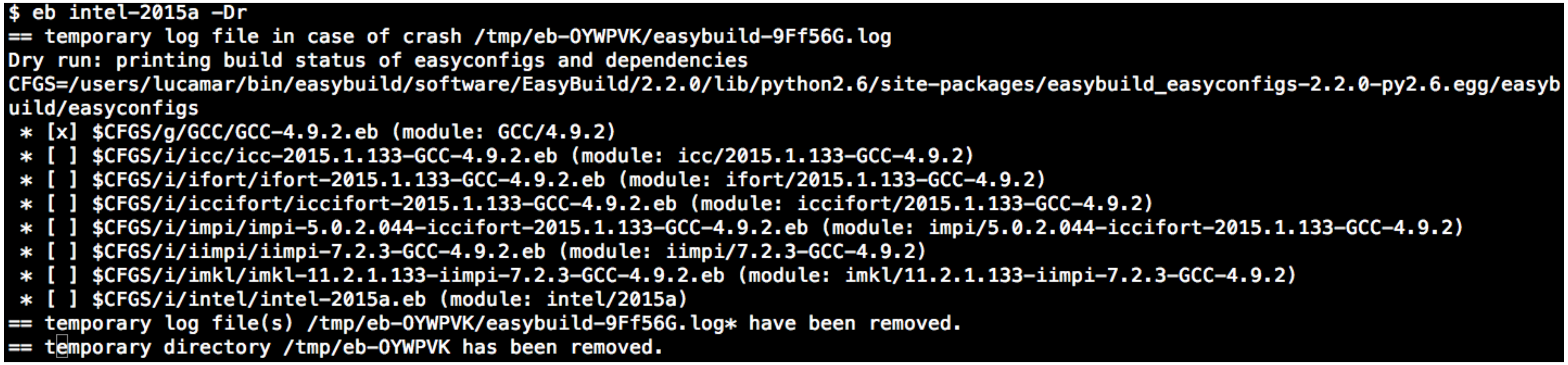


```
foss: BLACS, FFTW, GCC, OpenBLAS, OpenMPI, ScaLAPACK
gcccuda: CUDA, GCC
gimkl: GCC, imkl, impi
gimpi: GCC, impi
gmacml: ACML, BLACS, FFTW, GCC, MVAPICH2, ScaLAPACK
gmpich: GCC, MPICH
gmpich2: GCC, MPICH2
gmpolf: BLACS, FFTW, GCC, MPICH, OpenBLAS, ScaLAPACK
gmvapich2: GCC, MVAPICH2
gmvolf: BLACS, FFTW, GCC, MVAPICH2, OpenBLAS, ScaLAPACK
goalf: ATLAS, BLACS, FFTW, GCC, OpenMPI, ScaLAPACK
gompi: GCC, OpenMPI
gompic: CUDA, GCC, OpenMPI
goolf: BLACS, FFTW, GCC, OpenBLAS, OpenMPI, ScaLAPACK
goolfc: BLACS, CUDA, FFTW, GCC, OpenBLAS, OpenMPI, ScaLAPACK
gpsmpi: GCC, psmpi
gpsolf: BLACS, FFTW, GCC, OpenBLAS, ScaLAPACK, psmpi
gqacml: ACML, BLACS, FFTW, GCC, QLogicMPI, ScaLAPACK
iccifort: icc, ifort
ictce: icc, ifort, imkl, impi
iimpi: icc, ifort, impi
iiqmpi: QLogicMPI, icc, ifort
impich: MPICH, icc, ifort
impmkl: MPICH, icc, ifort, imkl
intel: icc, ifort, imkl, impi
```

Which dependencies are needed to build the toolchain?

Focus on a specific toolchain e.g. intel-2015a:

- **eb intel2015a -Dr** prints out the list of dependencies to build the full toolchain
- each dependency comes with an easyconfig file and can be resolved automatically

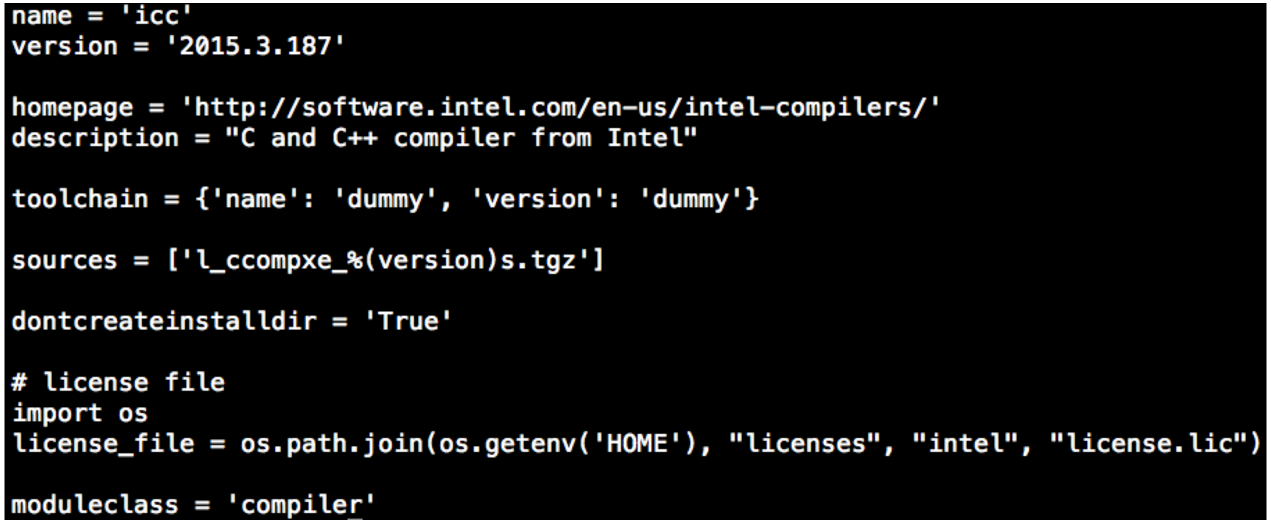


```
$ eb intel-2015a -Dr
== temporary log file in case of crash /tmp/eb-OYWPVK/easybuild-9Ff56G.log
Dry run: printing build status of easyconfigs and dependencies
CFGS=/users/lucamar/bin/easybuild/software/EasyBuild/2.2.0/lib/python2.6/site-packages/easybuild_easyconfigs-2.2.0-py2.6.egg/easyb
uild/easyconfigs
 * [x] $CFGS/g/GCC/GCC-4.9.2.eb (module: GCC/4.9.2)
 * [ ] $CFGS/i/icc/icc-2015.1.133-GCC-4.9.2.eb (module: icc/2015.1.133-GCC-4.9.2)
 * [ ] $CFGS/i/ifort/ifort-2015.1.133-GCC-4.9.2.eb (module: ifort/2015.1.133-GCC-4.9.2)
 * [ ] $CFGS/i/iccifort/iccifort-2015.1.133-GCC-4.9.2.eb (module: iccifort/2015.1.133-GCC-4.9.2)
 * [ ] $CFGS/i/impi/impi-5.0.2.044-iccifort-2015.1.133-GCC-4.9.2.eb (module: impi/5.0.2.044-iccifort-2015.1.133-GCC-4.9.2)
 * [ ] $CFGS/i/iimpi/iimpi-7.2.3-GCC-4.9.2.eb (module: iimpi/7.2.3-GCC-4.9.2)
 * [ ] $CFGS/i/imkl/imkl-11.2.1.133-iimpi-7.2.3-GCC-4.9.2.eb (module: imkl/11.2.1.133-iimpi-7.2.3-GCC-4.9.2)
 * [ ] $CFGS/i/intel/intel-2015a.eb (module: intel/2015a)
== temporary log file(s) /tmp/eb-OYWPVK/easybuild-9Ff56G.log* have been removed.
== temporary directory /tmp/eb-OYWPVK has been removed.
```

Intel packages and license files

Building Intel 2015.3.187:

- l_ccompxe_2015.3.187.tgz
- l_fcompxe_2015.3.187.tgz
- l_mkl_11.2.3.187.tgz
- l_mpi_p_5.0.3.048.tgz



```
name = 'icc'
version = '2015.3.187'

homepage = 'http://software.intel.com/en-us/intel-compilers/'
description = "C and C++ compiler from Intel"

toolchain = {'name': 'dummy', 'version': 'dummy'}

sources = ['l_ccompxe_%(version)s.tgz']

dontcreateinstalldir = 'True'

# license file
import os
license_file = os.path.join(os.getenv('HOME'), "licenses", "intel", "license.lic")

moduleclass = 'compiler'
```

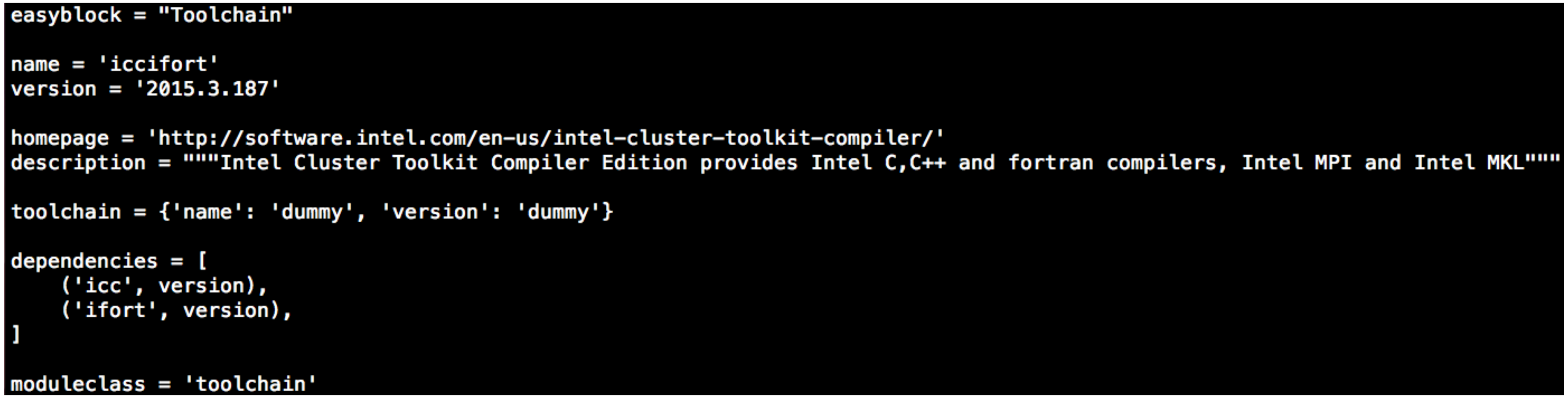
Quick and dirty solution for license path:

- export INTEL_LICENSE_FILE=/cm/shared/licenses/intel

Easyconfig and modulefiles wrappers

Some of the dependencies are wrappers i.e. contain already installed features:
- iccifort (icc and ifort)
- Iimpi (icc, ifort, intel-mpi)

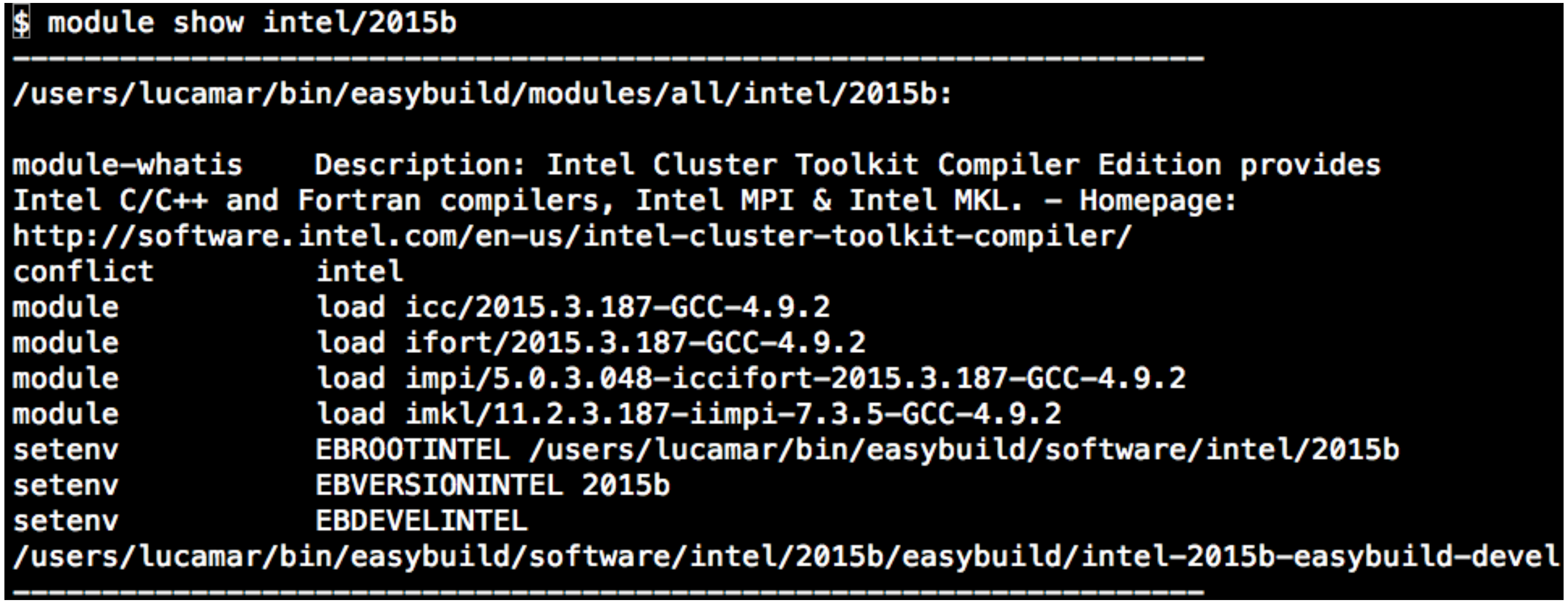
Their easyconfig files do not contain any source file, their dependency is resolved immediately



```
easyblock = "Toolchain"

name = 'iccifort'
version = '2015.3.187'

homepage = 'http://software.intel.com/en-us/intel-cluster-toolkit-compiler/'
description = """Intel Cluster Toolkit Compiler Edition provides Intel C,C++ and fortran compilers, Intel MPI and Intel MKL"""

toolchain = {'name': 'dummy', 'version': 'dummy'}

dependencies = [
    ('icc', version),
    ('ifort', version),
]

moduleclass = 'toolchain'
```

Conclusion

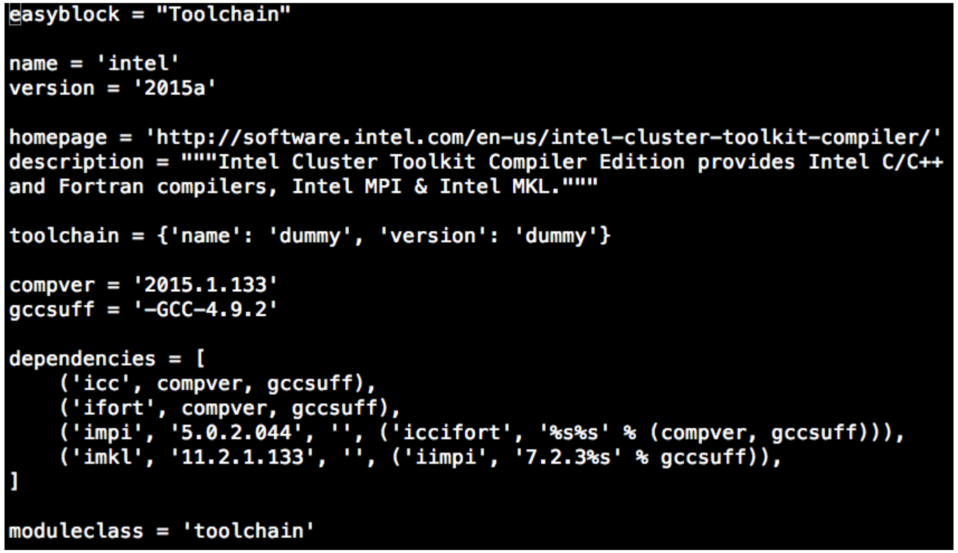
Editing few easyconfig files changing version numbers one easily builds a new version of Intel



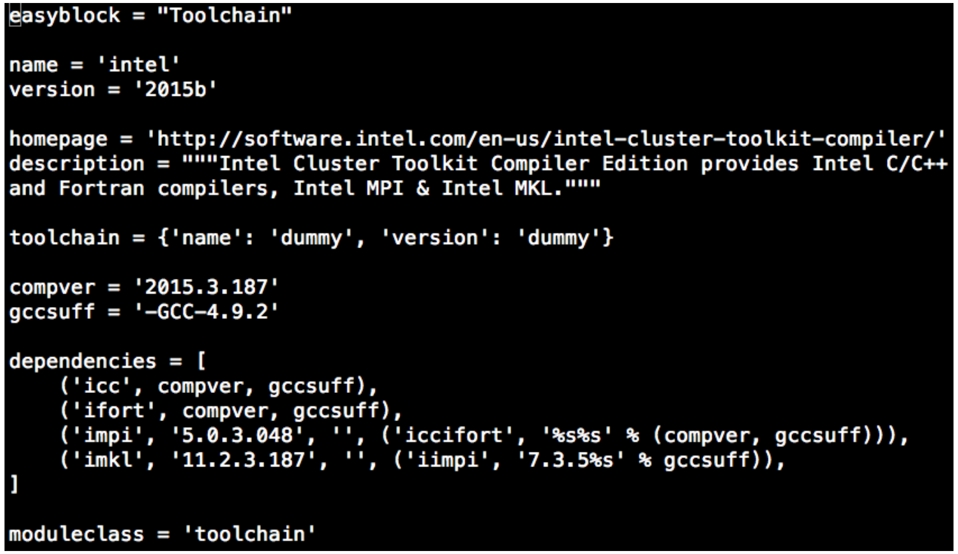
```
$ module show intel/2015b
-------------------------------------------------------------------
/users/lucamar/bin/easybuild/modules/all/intel/2015b:

module-whatis    Description: Intel Cluster Toolkit Compiler Edition provides
Intel C/C++ and Fortran compilers, Intel MPI & Intel MKL. - Homepage:
http://software.intel.com/en-us/intel-cluster-toolkit-compiler/
conflict         intel
module           load icc/2015.3.187-GCC-4.9.2
module           load ifort/2015.3.187-GCC-4.9.2
module           load impi/5.0.3.048-iccifort-2015.3.187-GCC-4.9.2
module           load imkl/11.2.3.187-iimpi-7.3.5-GCC-4.9.2
setenv           EBROOTINTEL /users/lucamar/bin/easybuild/software/intel/2015b
setenv           EBVERSIONINTEL 2015b
setenv           EBDEVELINTEL
/users/lucamar/bin/easybuild/software/intel/2015b/easybuild/intel-2015b-easybuild-devel
-------------------------------------------------------------------
```

Conclusion

Minimal difference between easyconfig files of the toolchains: little effort is required to adapt them



```
easyblock = "Toolchain"

name = 'intel'
version = '2015a'

homepage = 'http://software.intel.com/en-us/intel-cluster-toolkit-compiler/'
description = """Intel Cluster Toolkit Compiler Edition provides Intel C/C++
and Fortran compilers, Intel MPI & Intel MKL."""

toolchain = {'name': 'dummy', 'version': 'dummy'}

compver = '2015.1.133'
gccsuff = '-GCC-4.9.2'

dependencies = [
    ('icc', compver, gccsuff),
    ('ifort', compver, gccsuff),
    ('impi', '5.0.2.044', '', ('iccifort', '%s%s' % (compver, gccsuff))),
    ('imkl', '11.2.1.133', '', ('iimpi', '7.2.3%s' % gccsuff)),
]

moduleclass = 'toolchain'
```

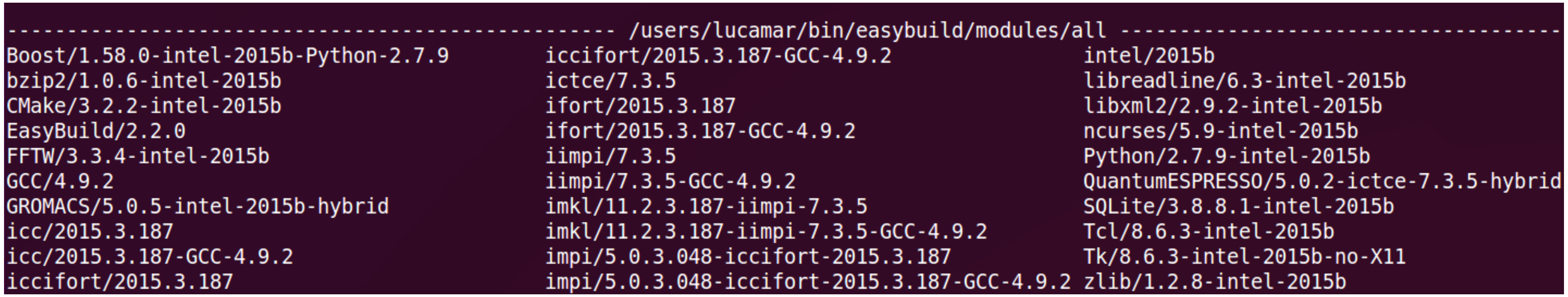


```
easyblock = "Toolchain"

name = 'intel'
version = '2015b'

homepage = 'http://software.intel.com/en-us/intel-cluster-toolkit-compiler/'
description = """Intel Cluster Toolkit Compiler Edition provides Intel C/C++
and Fortran compilers, Intel MPI & Intel MKL."""

toolchain = {'name': 'dummy', 'version': 'dummy'}

compver = '2015.3.187'
gccsuff = '-GCC-4.9.2'

dependencies = [
    ('icc', compver, gccsuff),
    ('ifort', compver, gccsuff),
    ('impi', '5.0.3.048', '', ('iccifort', '%s%s' % (compver, gccsuff))),
    ('imkl', '11.2.3.187', '', ('iimpi', '7.3.5%s' % gccsuff)),
]

moduleclass = 'toolchain'
```

Conclusion

Using the flag **--try-toolchain-version** one can easily build **libs** and **apps** with the new Intel:

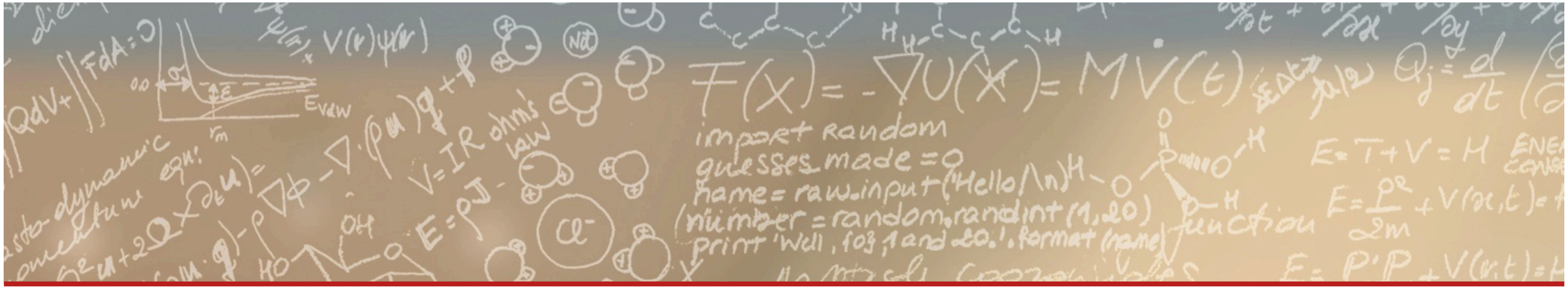
- Boost
- FFTW
- Gromacs
- QuantumEspresso



```
--------------------------------------------------- /users/lucamar/bin/easybuild/modules/all -------------------------------------
Boost/1.58.0-intel-2015b-Python-2.7.9        iccifort/2015.3.187-GCC-4.9.2                intel/2015b
bzip2/1.0.6-intel-2015b                      ictce/7.3.5                                  libreadline/6.3-intel-2015b
CMake/3.2.2-intel-2015b                      ifort/2015.3.187                             libxml2/2.9.2-intel-2015b
EasyBuild/2.2.0                              ifort/2015.3.187-GCC-4.9.2                   ncurses/5.9-intel-2015b
FFTW/3.3.4-intel-2015b                       iimpi/7.3.5                                  Python/2.7.9-intel-2015b
GCC/4.9.2                                    iimpi/7.3.5-GCC-4.9.2                        QuantumESPRESSO/5.0.2-ictce-7.3.5-hybrid
GROMACS/5.0.5-intel-2015b-hybrid             imkl/11.2.3.187-iimpi-7.3.5                  SQLite/3.8.8.1-intel-2015b
icc/2015.3.187                               imkl/11.2.3.187-iimpi-7.3.5-GCC-4.9.2        Tcl/8.6.3-intel-2015b
icc/2015.3.187-GCC-4.9.2                     impi/5.0.3.048-iccifort-2015.3.187           Tk/8.6.3-intel-2015b-no-X11
iccifort/2015.3.187                          impi/5.0.3.048-iccifort-2015.3.187-GCC-4.9.2 zlib/1.2.8-intel-2015b
```

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich


F(X) = -∇U(X) = MV(t)
import Random
guesses_made = 0
name = raw_input('Hello\n')
number = random.randint(1, 20)
print 'Well, {0}, 1 and 20.'.format(name)
E = T + V = H
V = IR ohm's law
E = ρJ
Evdw

Thank you for your attention