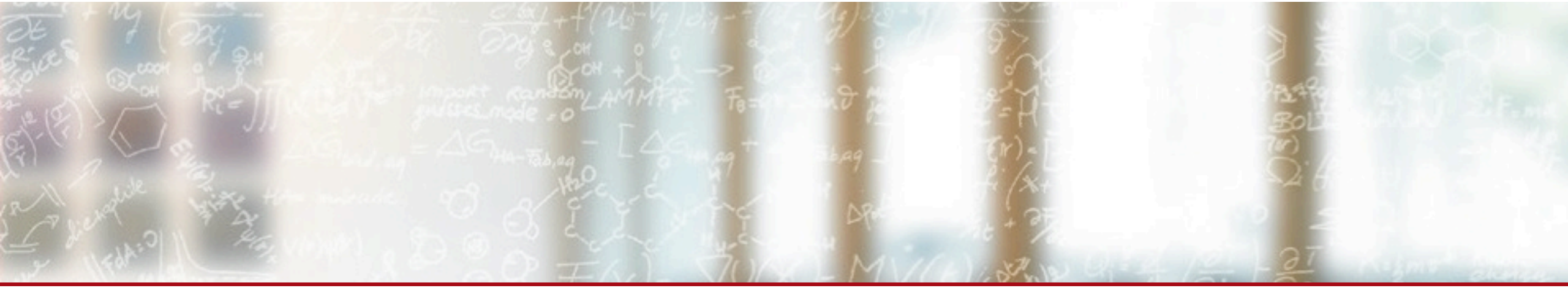
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich



How to build HPC software at CSCS: a practical introduction

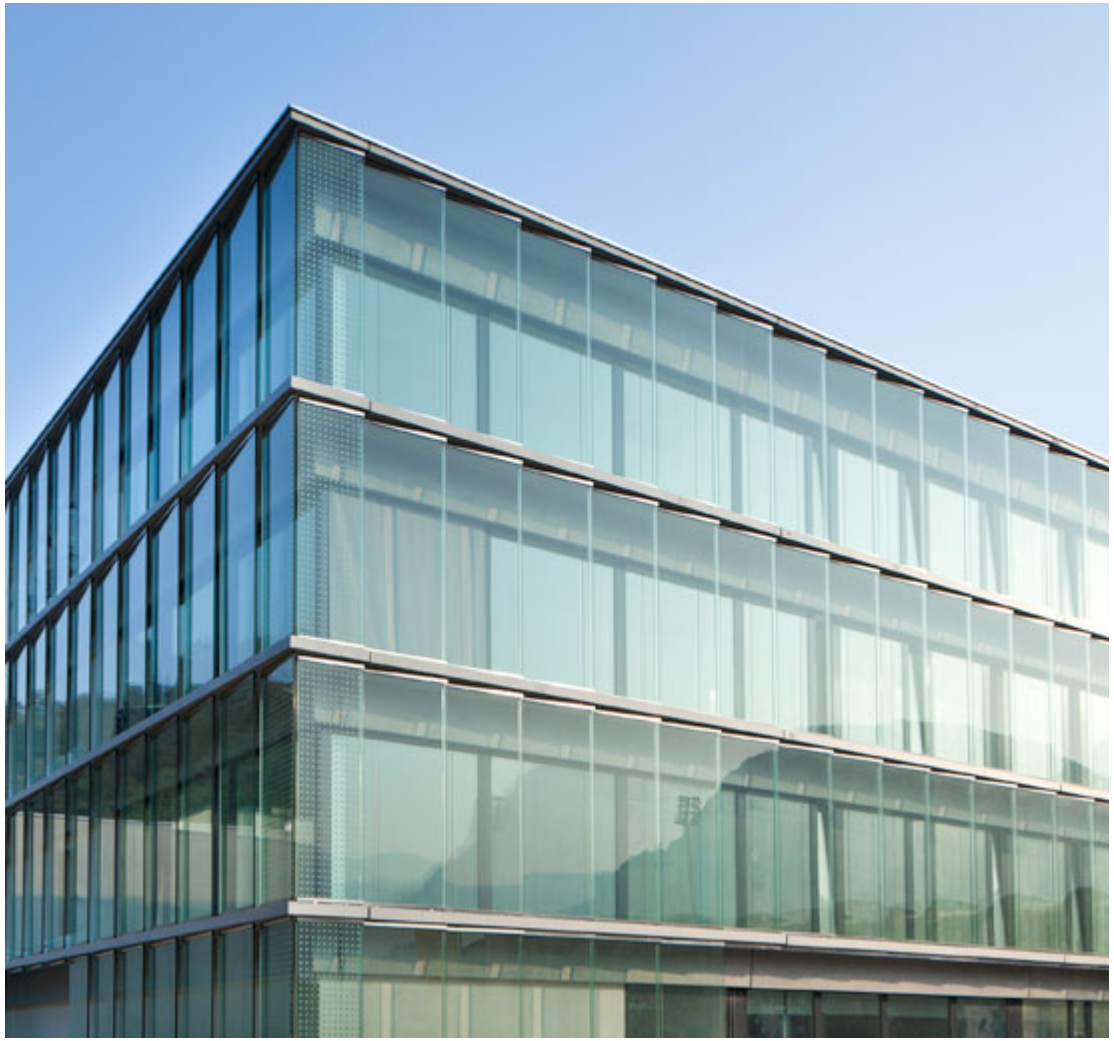
User Lab Day 2022

Alberto Invernizzi, Harmen Stoppels and Luca Marsella, CSCS

September 2nd 2022

Outline of the Presentation

- Cray Programming Environment
- Easybuild Framework
- Spack workflows
- Useful Links



CSCS office building in Lugano

Cray Programming Environment

Piz Daint Specifications

Model	Cray XC50/XC40
XC50 Compute Nodes (Intel Haswell processor)	Intel® Xeon® E5-2690 v3 @ 2.60GHz (12 cores, 64GB RAM) and NVIDIA® Tesla® P100 16GB
XC40 Compute Nodes (Intel Broadwell processor)	Intel® Xeon® E5-2695 v4 @ 2.10GHz (2 x 18 cores, 64/128 GB RAM)
Login Nodes	Intel® Xeon® E5-2650 v3 @ 2.30GHz (10 cores, 256 GB RAM)
Interconnect Configuration	Aries routing and communications ASIC Dragonfly network topology
Scratch capacity	Piz Daint scratch filesystem: 8.8 PB

File Systems:

- **/project** is mounted with **read-only** access on compute nodes

Cray Linux Environment 7.0 UP03

- Cray Linux Environment (CLE) is the operating system on Cray systems
- CLE 7.0 UP03 is based on the SUSE Linux Enterprise Server version 15
- CLE 7.0 UP03 software release is available on Piz Daint since Feb 2022
- HPE Cray Technical Documentation:
  - https://support.hpe.com

NVIDIA CUDA Toolkit v11

- Comprehensive development environment to build **GPU-accelerated applications**
  - **compiler** for NVIDIA GPUs
  - optimized **math libraries**
  - **debugging** and **performance** tools

- Features **programming guides**, user manuals, API reference and **online documentation** to get started quickly

- NVIDIA developer portal:
  https://developer.nvidia.com/cuda-zone

NVIDIA Tesla P100 GPU Accelerator

Cray XC Programming Environment

- Cray XC PE 21.09 includes the **Cray Developer Toolkit - CDT 21.09**
  - non-default Programming Environments can be accessed with **cdt** modules

- The following products have been updated within this release:
  - **Cray Compiling Environment - CCE**
    - cce 12.0.3, cray-mpich 7.7.18, cray-libsci 20.09.1
  - **Cray Performance Measurement & Analysis Tools - CPMAT**
    - perftools 21.09.0
  - **Cray Environment Setup and Compiling support - CENV**
    - modules 3.2.11.4 and craype 2.7.10
  - **Third party products**
    - gcc 10.3.0 and 11.2.0, cray-python 3.9.4.1, cray-R 4.1.1.0

Setting the Programming Environment

- CSCS systems use the modules framework
  - The modules manage applications and libraries path
  - You can check currently loaded modules with **module list**

- Some **modules** are already loaded at login
  - The default environment on Piz Daint is **PrgEnv-cray**
  - The default architecture is XC50 (Intel Haswell): **craype-haswell**
  - You can browse the available modules with **module avail**

- You must **adjust your target architecture**
    - **daint-gpu** targets the XC50 (Intel Haswell and P100 Tesla GPUS)
    - **daint-mc** targets the XC40 (Intel Broadwell multicore)
    - These modules update the **MODULEPATH**

Static vs Dynamic linking

- Binaries can be linked **statically** or **dynamically** to the libraries on the system:
  - Cray compiler wrappers (**cc**, **CC**, **ftn**) create dynamically-linked executables by default
  - Static linking: wrapper flag **-static** or **export CRAYPE_LINK_TYPE=static** before building

- **Dynamically linked** binaries can generally be used after a system library update

- **Statically linked binaries** using directly or indirectly the network interface libraries (uGNI/DMAPP) **must be recompiled** after an update:
  - This includes **applications using MPI or SHMEM libraries**, as well as the PGAS (Partitioned Global Address Space) languages such as **UPC, Fortran with Coarrays, and Chapel**

  - DMAPP (Distributed Shared Memory Application) and uGNI (user Generic Network Interface) are tied to specific kernel versions and no backward or forward compatibility is provided

Non-default Programming Environments with Cray Development Toolkit

- Use the command **module avail cdt** to get the list of available **cdt** modules
  - Loading a non default **cdt** module while building or at runtime requires prepending **CRAY_LD_LIBRARY_PATH** to **LD_LIBRARY_PATH**
- The environment variable CRAY_LD_LIBRARY_PATH holds every product library path in the current environment, updated when modules are loaded / unloaded
- In the example below, we link dynamically the library of **cray-libsci** provided by the non-default unsupported Programming Environment **cdt/20.08** (Aug 2020):

```
module load cdt/20.08
export LD_LIBRARY_PATH=$CRAY_LD_LIBRARY_PATH:$LD_LIBRARY_PATH
<build command> or <run command>
```

More info on the User Portal at https://user.cscs.ch/computing/compilation

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

EasyBuild Framework

What is EasyBuild?

- HPC software installation framework by UGhent (Belgium)
  - **automates** software builds, allowing to **reproduce** easily previous builds
  - recipes written in **Python** with **automatic dependency resolution**
- Key features
  - supports **co-existence of versions/builds** with dedicated installation prefix
  - allows **sharing** with the HPC community of EasyBuild users
  - code patching generating modulefiles and **retaining logs** of the build
- Advanced features
  - recipe file (know as **easyconfig**) used for build is archived
  - build entire software stack with a single command in **parallel**

More details on https://easybuild.readthedocs.io

EasyBuild on Piz Daint

- EasyBuild is available loading **EasyBuild-custom**
  - The architecture on Piz Daint defines **CRAY_CPU_TARGET**
    - **module load daint-gpu EasyBuild-custom**
  - The installation folder is set by **EASYBUILD_PREFIX**
    - Defaults to **$HOME/easybuild/daint/<haswell|broadwell>**
    - Adjust **MODULEPATH**: **module use $EASYBUILD_PREFIX/modules/all**

- Most useful EasyBuild options
  - **eb -S/--search <pattern> | grep Cray** Recipes matching Cray toolchains
  - **eb <name_version>.eb -r** Build a package resolving dependencies
  - **eb -h / -H** Short / full help message

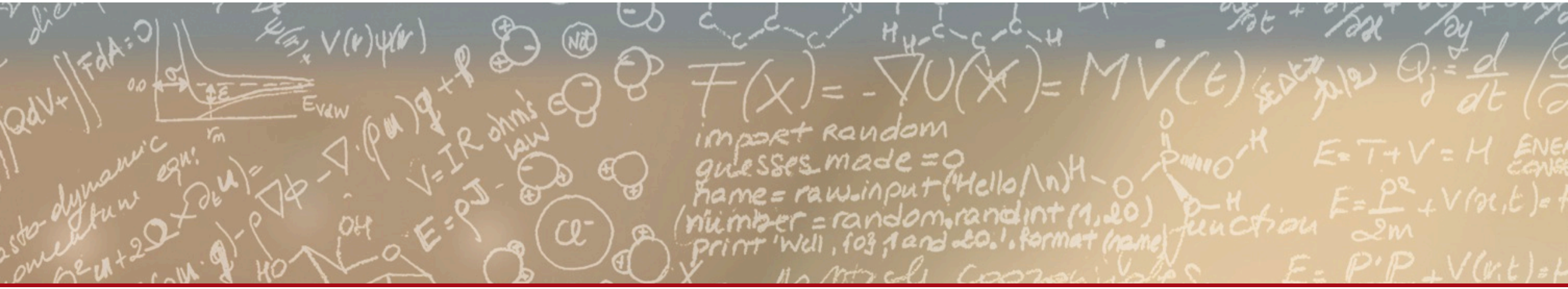
Customize existing recipes

- Extend or customize existing EasyBuild recipes
  - clone CSCS repository
    - **git clone https://github.com/eth-cscs/production.git**
- EasyBuild recipes are listed alphabetically under the folder
  - **production/easybuild/easyconfigs**
- Use your local repository defining **EB_CUSTOM_REPOSITORY**
  - **export EB_CUSTOM_REPOSITORY=<full_path>/production/easybuild**
- Load EasyBuild after selecting the target architecture
  - **module load daint-gpu EasyBuild-custom**

Useful Links

- CSCS User Portal:
  - http://user.cscs.ch
- HPE/Cray documentation:
  - https://support.hpe.com
- NVIDIA Documentation:
  - https://docs.nvidia.com
- Contact us:
  - https://support.cscs.ch

Piz Daint in the machine room at CSCS



Thank you for your kind attention