CP2K : high-performance electronic structure calculations

Marcella Iannuzzi
Department of Chemistry, University of Zurich

http://www.cp2k.org

CP2K History

25. June 2001

CP2K source repository goes on line on berlios.de

Oct. 2011first official release CP2K 2.2

then on sourceforge.net

now active development takes place under GitHub

https://github.com/cp2k

20 years of open development

Origin from combining two codes:

Quickstep DFT Code, MPI Stuttgart (Lippert, Krack, Hutter)

Fist MD Code, Penn, Philadelphia (Mundy, Balasubramanian, Bagchi)

CP2K SOURCE CODE DEVELOPMENT

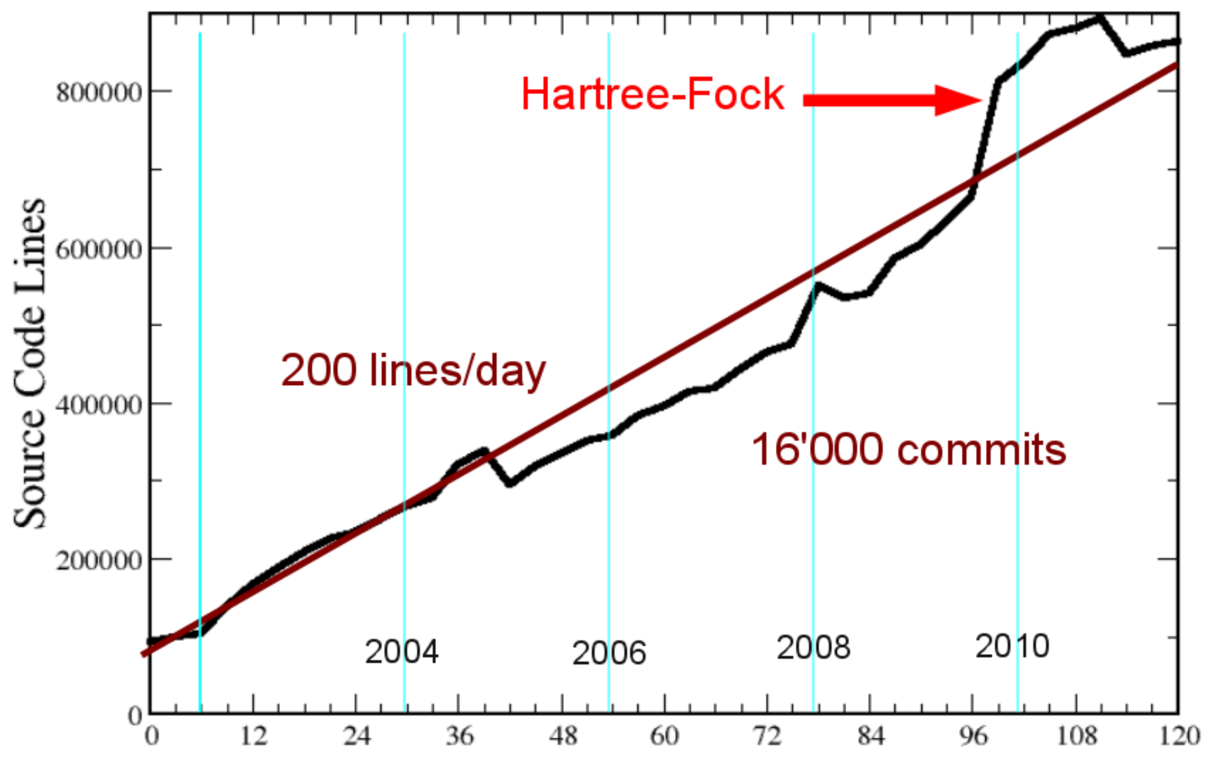
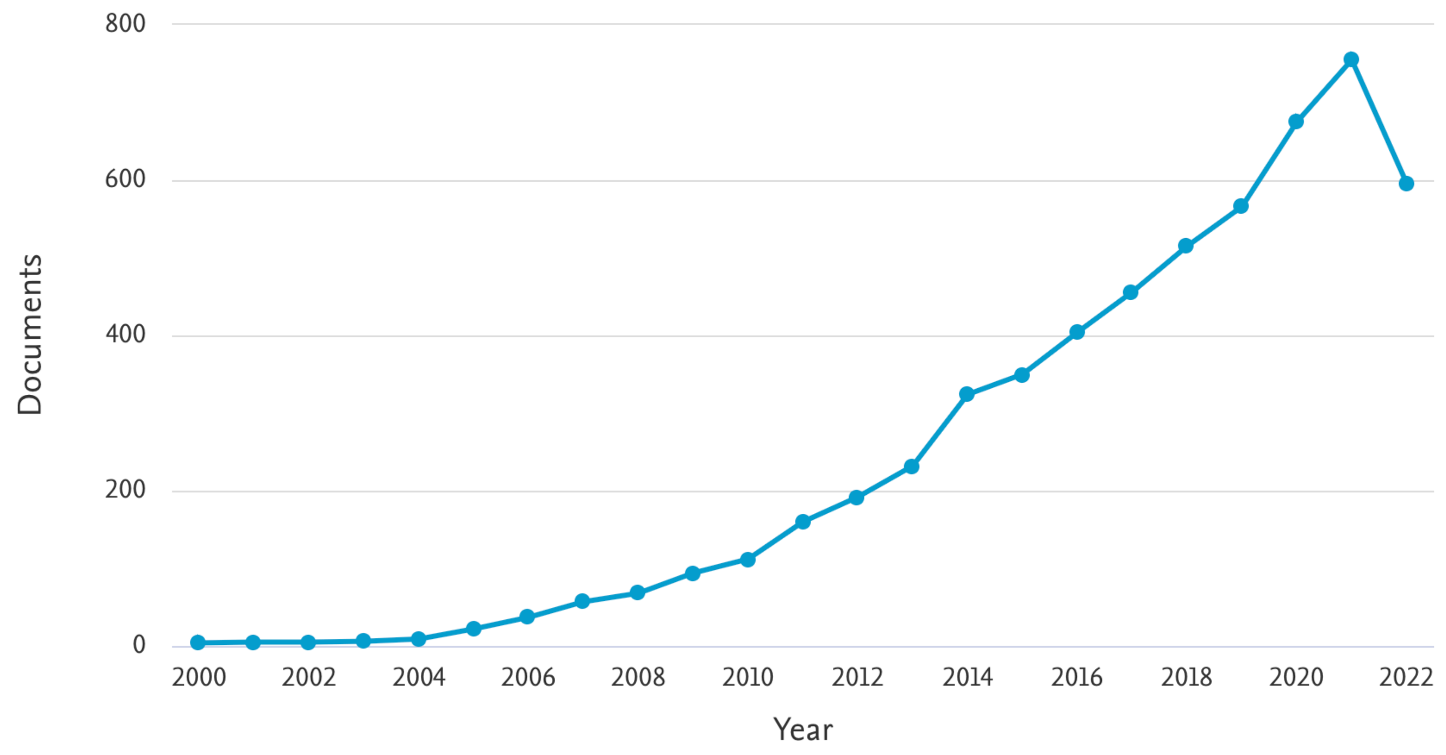


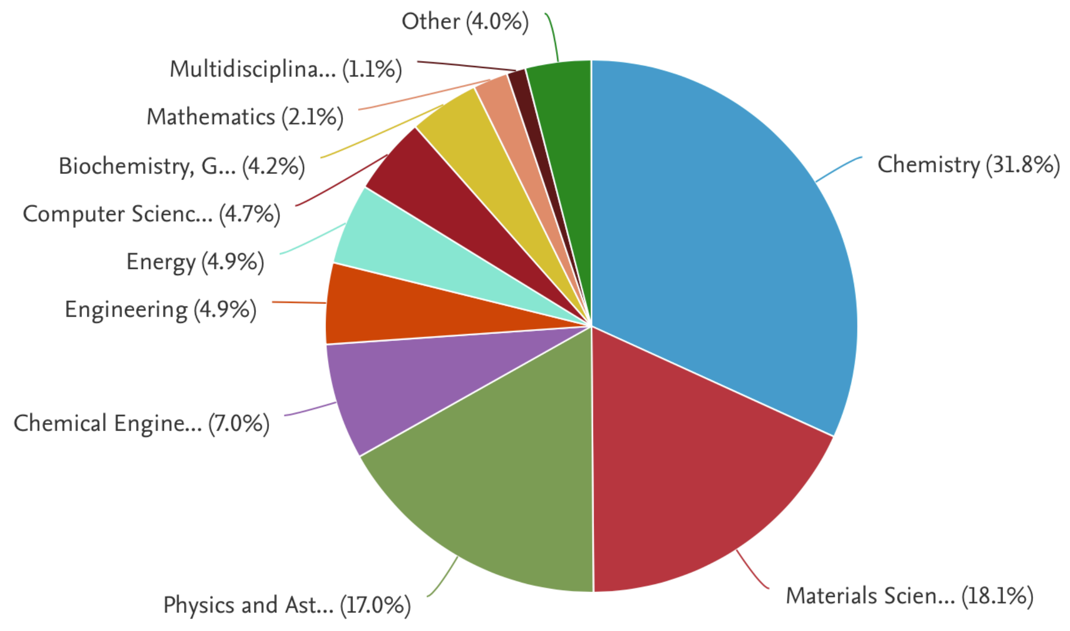
Image from Jürg Hutter

Citations to CP2K (SCOPUS)

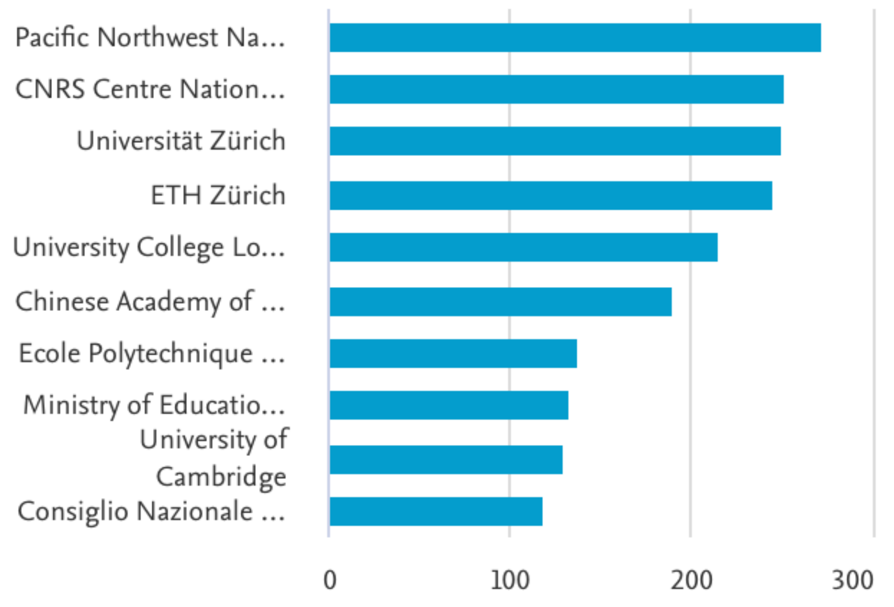
Documents by year



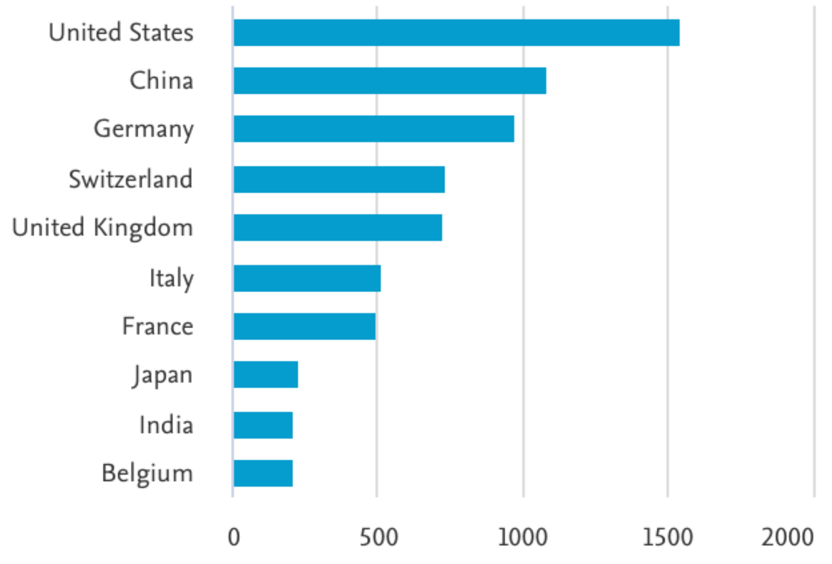
Documents by subject area



Documents by affiliation



Documents by country/territory



A General Atomistic Simulations

Extended condensed matter systems

- Force Methods: KS/OF DFT (vdw), Hybrid, MP2, RPA, GW, Classical Force Fields, QM/MM, DFTB, XTB, mixed

- Sampling Methods: GeoOpt, CellOpt, Molecular Dynamics, Ehrenfest MD, FES and PES tools (Metadynamics), Monte Carlo, PIMD

- Properties and spectroscopy: vibrational, IR,TDDFT, NMR, EPR, NEXAFS, Rama NEGF, n,…

Excellent parallel scalability
Flexibility in combining methods

Open Source

The source of CP2K is open and freely available for everybody under the GPL license.

Github Repository: github.com/cp2k
→ open development with continuous integration

Development Version	Released Version
▪ Most recent ▪ All new features ▪ Potentially unstable / buggy ▪ only available via Git	▪ Older ▪ Stable, no ongoing development ▪ All major functionality in good shape ▪ Only rare backports of bug fixes (as time permits)

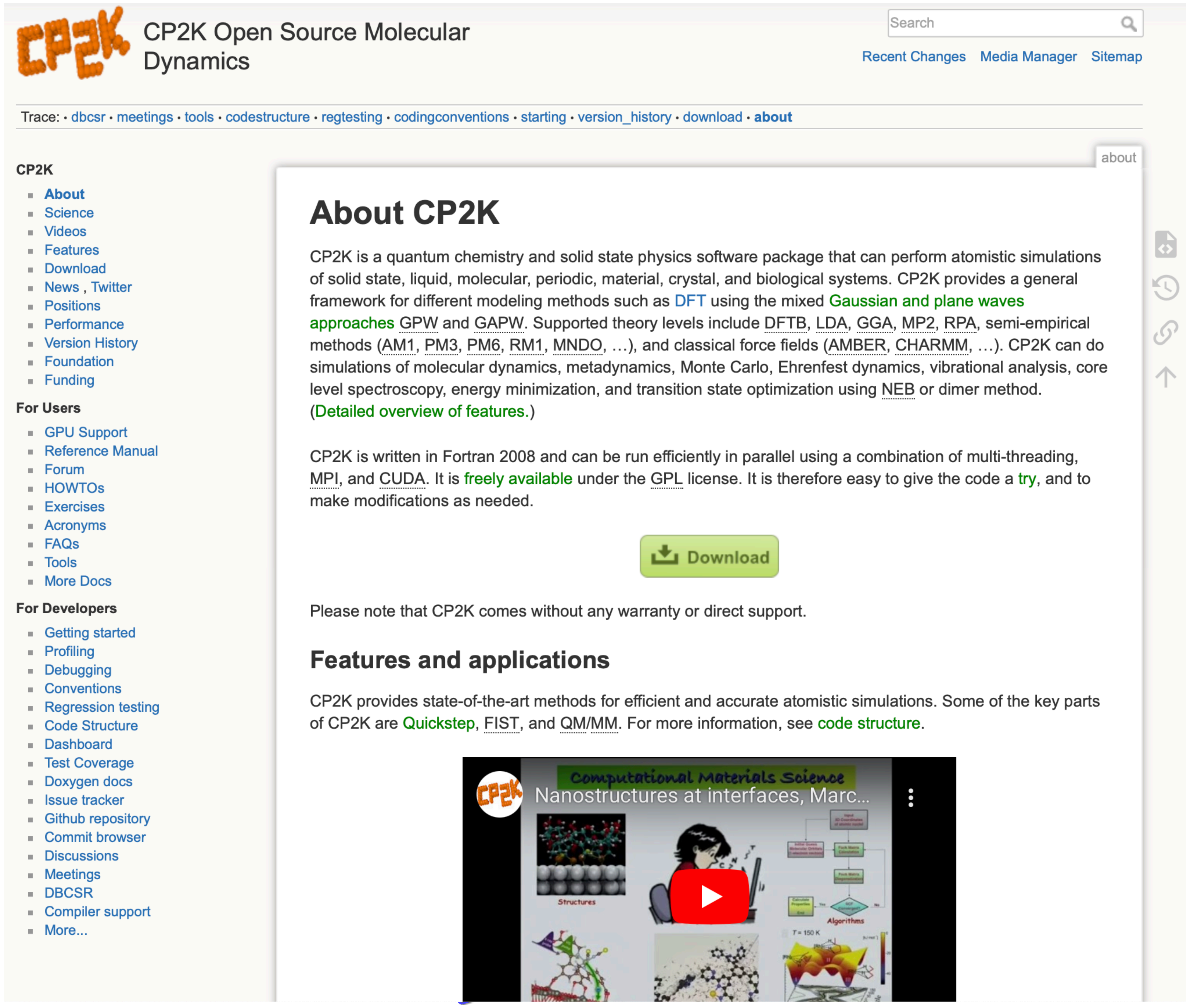
From official releases
→ tarballs (cp2k-X.Y.tar.bz2). Latest cp2k-2022.2.tar.bz2

From distributions

Features and Science

User information and material found on
→ www.cp2k.org

info for
Users
&
Developers


CP2K Open Source Molecular Dynamics

Search

Recent Changes Media Manager Sitemap

Trace: • dbcsr • meetings • tools • codestructure • regtesting • codingconventions • starting • version_history • download • about

CP2K

- About
- Science
- Videos
- Features
- Download
- News , Twitter
- Positions
- Performance
- Version History
- Foundation
- Funding

For Users

- GPU Support
- Reference Manual
- Forum
- HOWTOs
- Exercises
- Acronyms
- FAQs
- Tools
- More Docs

For Developers

- Getting started
- Profiling
- Debugging
- Conventions
- Regression testing
- Code Structure
- Dashboard
- Test Coverage
- Doxygen docs
- Issue tracker
- Github repository
- Commit browser
- Discussions
- Meetings
- DBCSR
- Compiler support
- More...

about

About CP2K

CP2K is a quantum chemistry and solid state physics software package that can perform atomistic simulations of solid state, liquid, molecular, periodic, material, crystal, and biological systems. CP2K provides a general framework for different modeling methods such as DFT using the mixed Gaussian and plane waves approaches GPW and GAPW. Supported theory levels include DFTB, LDA, GGA, MP2, RPA, semi-empirical methods (AM1, PM3, PM6, RM1, MNDO, …), and classical force fields (AMBER, CHARMM, …). CP2K can do simulations of molecular dynamics, metadynamics, Monte Carlo, Ehrenfest dynamics, vibrational analysis, core level spectroscopy, energy minimization, and transition state optimization using NEB or dimer method. (Detailed overview of features.)

CP2K is written in Fortran 2008 and can be run efficiently in parallel using a combination of multi-threading, MPI, and CUDA. It is freely available under the GPL license. It is therefore easy to give the code a try, and to make modifications as needed.

Download

Please note that CP2K comes without any warranty or direct support.

Features and applications

CP2K provides state-of-the-art methods for efficient and accurate atomistic simulations. Some of the key parts of CP2K are Quickstep, FIST, and QM/MM. For more information, see code structure.

Computational Materials Science
Nanostructures at interfaces, Marc...
Structures
Algorithms

Some CP2K code's Features

- Fortran2008, active development under GitHub
- Freely available, open source, General Public License
- Community Developers Platform (UZH, IBM Research, ETHZ, UPB, LLNL, PNNL, PSI, U Bochum, Aalto, Lincoln-UK, UGent, URegensburg, McGill-Can, HPE-CH, Intel-CH….)
- User community through Google groups
- MPI and OpenMP parallelisation, CUDA C extensions : porting on >100'000 cores and to GPUs
- Quality control: automatic regression and memory leak (>3000)
- External Library: Lapack/BLAS, ScaLapack/BLACS, MPI, OpenMP, FFTW, libint, libxc, LIBXSMM, ELPA, COSMA, LibVori, PEXSI, QUIP, spglib, SPLA…
- Internal library handling sparse matrices and tensors https://github.com/cp2k/dbcsr
- Integration: i-PI, PLUMED, OMEN, SIRIUS, PhonoPy, PyRetis, …

Preparation/Installation

The standard way to build CP2K is via the **toolchain script**

```
$ git clone --recursive https://github.com/cp2k/cp2k.git

$ cd cp2k/tools/toolchain

$ ./install_cp2k_toolchain.sh –help

$ ./install_cp2k_toolchain.sh –with-libxsmm=install –with-openblas=system –
with-fftw=system –enable-cuda
```

README

```
MPI is detected and it appears to be OpenMPI
nvcc not found, disabling CUDA by default
Compiling with 8 processes.
==================== Finding binutils from system paths ====================
[...]

==================== generating arch files ====================
arch files can be found in the /data/cp2k/tools/toolchain/install/arch
subdirectory Wrote /data/cp2k/tools/toolchain/install/arch/local.sopt
Wrote /data/cp2k/tools/toolchain/install/arch/local.sdbg
Wrote /data/cp2k/tools/toolchain/install/arch/local.ssmp
[...]
========================== usage =========================

cp /data/cp2k/tools/toolchain/install/arch/* to the cp2k/arch/ directory

source /data/cp2k/tools/toolchain/install/setup

make -j 8 ARCH=local VERSION="sopt sdbg ssmp popt pdbg psmp"
```

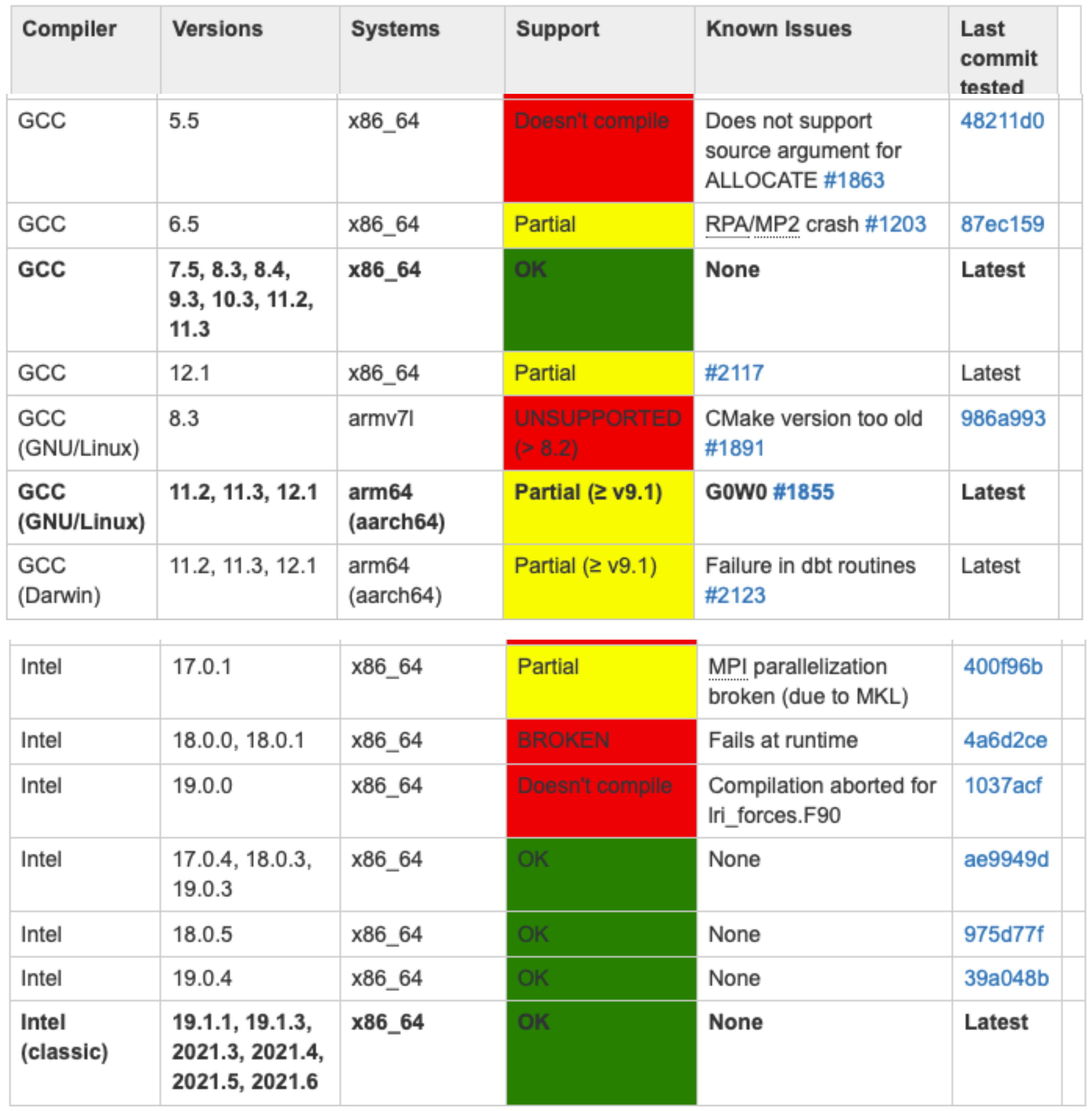
Default: Uses system compiler, linker and MPI

Builds and configures for: libxc, libint, libxsmm, ELPA, SIRIUS

Support for Linux & macOS

Compiler Support

CP2K adheres to the Fortran 2008 standard, not all compilers (or compiler versions) are able to build CP2K correctly. GCC is the most tested compiler. We test some Intel Compiler versions.



Compiler	Versions	Systems	Support	Known Issues	Last commit tested
GCC	5.5	x86_64	Doesn't compile	Does not support source argument for ALLOCATE #1863	48211d0
GCC	6.5	x86_64	Partial	RPA/MP2 crash #1203	87ec159
GCC	**7.5, 8.3, 8.4, 9.3, 10.3, 11.2, 11.3**	**x86_64**	**OK**	**None**	**Latest**
GCC	12.1	x86_64	Partial	#2117	Latest
GCC (GNU/Linux)	8.3	armv7l	UNSUPPORTED (> 8.2)	CMake version too old #1891	986a993
GCC (GNU/Linux)	**11.2, 11.3, 12.1**	**arm64 (aarch64)**	**Partial (≥ v9.1)**	**G0W0 #1855**	**Latest**
GCC (Darwin)	11.2, 11.3, 12.1	arm64 (aarch64)	Partial (≥ v9.1)	Failure in dbt routines #2123	Latest
Intel	17.0.1	x86_64	Partial	MPI parallelization broken (due to MKL)	400f96b
Intel	18.0.0, 18.0.1	x86_64	BROKEN	Fails at runtime	4a6d2ce
Intel	19.0.0	x86_64	Doesn't compile	Compilation aborted for lri_forces.F90	1037acf
Intel	17.0.4, 18.0.3, 19.0.3	x86_64	OK	None	ae9949d
Intel	18.0.5	x86_64	OK	None	975d77f
Intel	19.0.4	x86_64	OK	None	39a048b
Intel (classic)	**19.1.1, 19.1.3, 2021.3, 2021.4, 2021.5, 2021.6**	**x86_64**	**OK**	**None**	**Latest**

Uniform formatting by prettify script

make -j pretty

Write explicit code

Fight spaghetti code

Format and document

Write tests

Doxygen documentation

make doxify

Some External Libraries

Blas and Lapack vendor-provided libraries make significant difference (ACML,MKL,ESSL)

MPI and SCALAPACK optional for MPI builds, must match compiler (MPICH2, OpenMPI)

FFTW optional, to improve speed, FFTW3

LIBINT to include HF exchange, any should be compatible —enable-eri=1, -D__MAX_CONTR=4

LIBXSMM optional, to improve performance for matrix operations, deep learning primitives

LIBXC optional but more complete, version 5.1.0 or later, no fourth derivatives

ELPA optional improved performance for diagonalization, replaces SYEVD, optimised kernels

Sparse Matrix Library

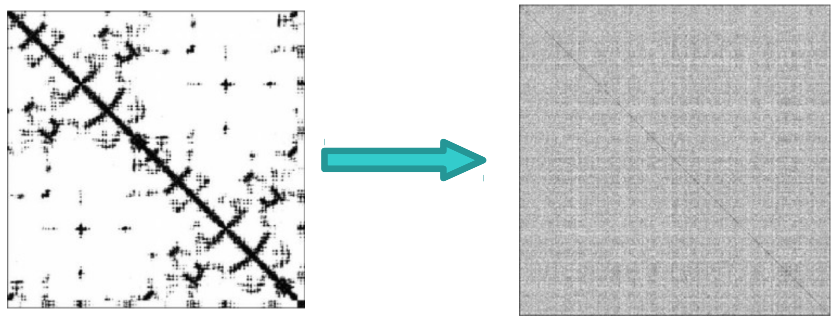
DBCSR: Distributed Blocked Compressed Sparse Row

standalone sparse matrix library designed to efficiently perform sparse matrix matrix multiplication, among other operations. It is MPI and OpenMP parallel, and can exploit accelerators.

https://github.com/cp2k/dbcsr

- For massively parallel architectures
- Optimised for 10000s of non-zeros per row (dense limit)
- Stored in block form : atoms or molecules
- Cannons algorithm: 2D layout (rows/columns) and 2D distribution of data
- Homogenised for load balance

https://dx.doi.org/10.1016%2Fj.parco.2014.03.012



given processor communicates only with nearest neighbours
transferred data decreases as number of processors increases

Regression Testing

do_regtest script

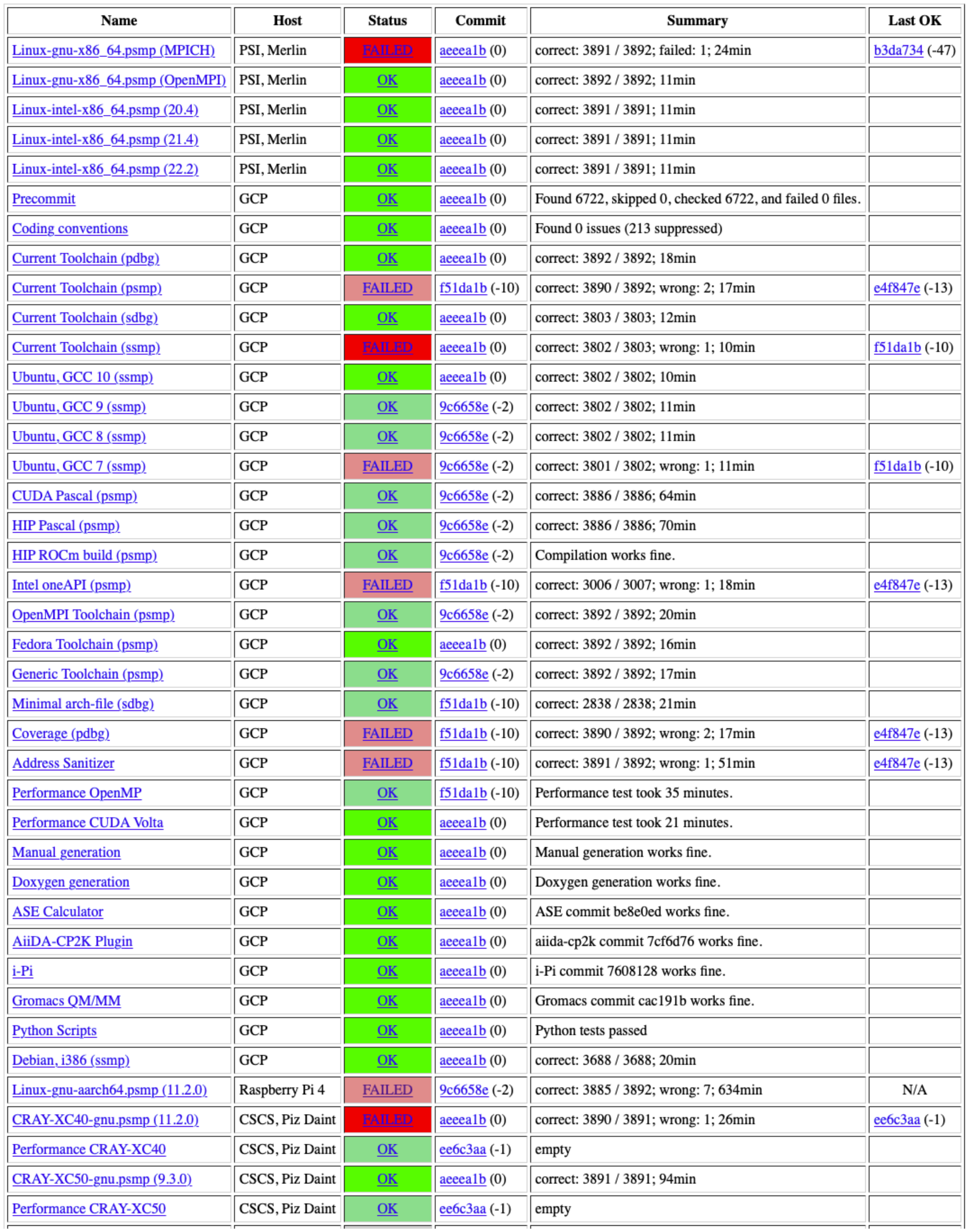
```
----------------------------- Settings ---------------------------------
MPI ranks:      2
OpenMP threads: 2
GPU devices:    1
Workers:        2
Timeout [s]:    400
Work base dir:  /scratch/snx3000/mkrack/rt/CRAY-XC50-gnu/cp2k/regtesting/
TEST-CRAY-XC50-gnu-psmp-2022-10-10_11-56-27
MPI exec:       ['srun', '--mem-per-cpu=1280']
Keepalive:      True
Debug:          False
ARCH:           CRAY-XC50-gnu
VERSION:        psmp
Flags:
omp,libint,fftw3,libxc,pexsi,elpa,elpa_nvidia_gpu,parallel,mpi3,scalapack,
cosma,xsmm,dbcsr_acc,max_contr=4,plumed2,spglib,sirius,check_diag,libvori,
libbqb,offload_cuda,no_offload_pw,spla_gemm_offloading,libvdwxc
------------------------------------------------------------------------
Skipping TMC/regtest_ana_on_the_fly because its requirements are not
satisfied.
Skipping Fist/regtest-quip because its requirements are not satisfied.
Launched 323 test directories and 2 worker...
```

```
 ------------------------------- Summary --------------------------------
 Number of FAILED  tests 0
 Number of WRONG   tests 0
 Number of CORRECT tests 3891
 Total number of   tests 3891

 Summary: correct: 3891 / 3891; 94min
 Status: OK

 *************************** Testing ended ******************************
```

Automatically skips unavailable features

Verification

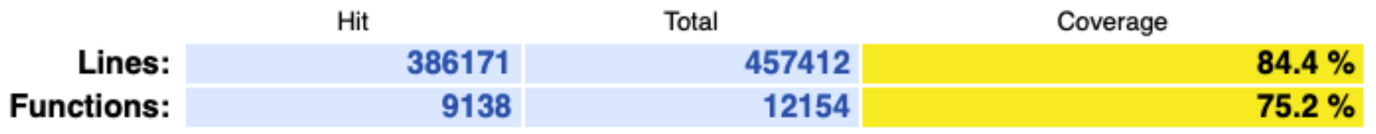
State of the latest version the DASHBOARD



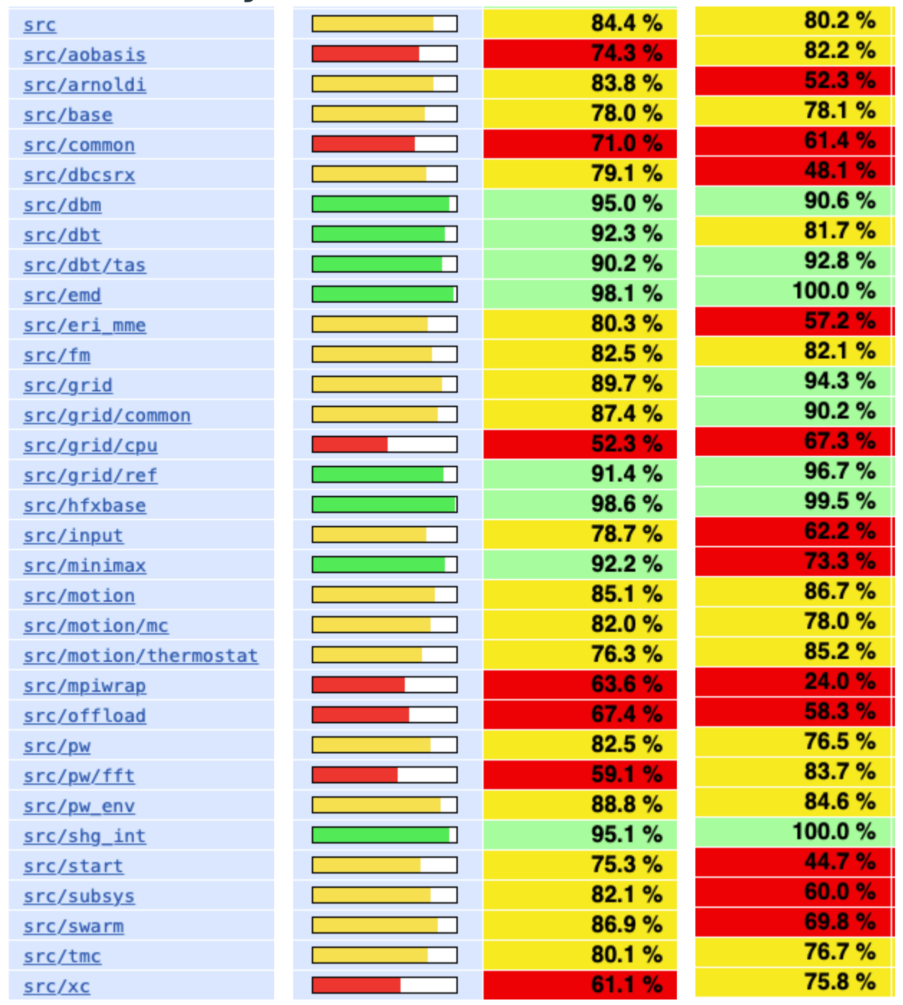
Name	Host	Status	Commit	Summary	Last OK
Linux-gnu-x86_64.psmp (MPICH)	PSI, Merlin	FAILED	aeeea1b (0)	correct: 3891 / 3892; failed: 1; 24min	b3da734 (-47)
Linux-gnu-x86_64.psmp (OpenMPI)	PSI, Merlin	OK	aeeea1b (0)	correct: 3892 / 3892; 11min	
Linux-intel-x86_64.psmp (20.4)	PSI, Merlin	OK	aeeea1b (0)	correct: 3891 / 3891; 11min	
Linux-intel-x86_64.psmp (21.4)	PSI, Merlin	OK	aeeea1b (0)	correct: 3891 / 3891; 11min	
Linux-intel-x86_64.psmp (22.2)	PSI, Merlin	OK	aeeea1b (0)	correct: 3891 / 3891; 11min	
Precommit	GCP	OK	aeeea1b (0)	Found 6722, skipped 0, checked 6722, and failed 0 files.	
Coding conventions	GCP	OK	aeeea1b (0)	Found 0 issues (213 suppressed)	
Current Toolchain (pdbg)	GCP	OK	aeeea1b (0)	correct: 3892 / 3892; 18min	
Current Toolchain (psmp)	GCP	FAILED	f51da1b (-10)	correct: 3890 / 3892; wrong: 2; 17min	e4f847e (-13)
Current Toolchain (sdbg)	GCP	OK	aeeea1b (0)	correct: 3803 / 3803; 12min	
Current Toolchain (ssmp)	GCP	FAILED	aeeea1b (0)	correct: 3802 / 3803; wrong: 1; 10min	f51da1b (-10)
Ubuntu, GCC 10 (ssmp)	GCP	OK	aeeea1b (0)	correct: 3802 / 3802; 10min	
Ubuntu, GCC 9 (ssmp)	GCP	OK	9c6658e (-2)	correct: 3802 / 3802; 11min	
Ubuntu, GCC 8 (ssmp)	GCP	OK	9c6658e (-2)	correct: 3802 / 3802; 11min	
Ubuntu, GCC 7 (ssmp)	GCP	FAILED	9c6658e (-2)	correct: 3801 / 3802; wrong: 1; 11min	f51da1b (-10)
CUDA Pascal (psmp)	GCP	OK	9c6658e (-2)	correct: 3886 / 3886; 64min	
HIP Pascal (psmp)	GCP	OK	9c6658e (-2)	correct: 3886 / 3886; 70min	
HIP ROCm build (psmp)	GCP	OK	9c6658e (-2)	Compilation works fine.	
Intel oneAPI (psmp)	GCP	FAILED	f51da1b (-10)	correct: 3006 / 3007; wrong: 1; 18min	e4f847e (-13)
OpenMPI Toolchain (psmp)	GCP	OK	9c6658e (-2)	correct: 3892 / 3892; 20min	
Fedora Toolchain (psmp)	GCP	OK	aeeea1b (0)	correct: 3892 / 3892; 16min	
Generic Toolchain (psmp)	GCP	OK	9c6658e (-2)	correct: 3892 / 3892; 17min	
Minimal arch-file (sdbg)	GCP	OK	f51da1b (-10)	correct: 2838 / 2838; 21min	
Coverage (pdbg)	GCP	FAILED	f51da1b (-10)	correct: 3890 / 3892; wrong: 2; 17min	e4f847e (-13)
Address Sanitizer	GCP	FAILED	f51da1b (-10)	correct: 3891 / 3892; wrong: 1; 51min	e4f847e (-13)
Performance OpenMP	GCP	OK	f51da1b (-10)	Performance test took 35 minutes.	
Performance CUDA Volta	GCP	OK	aeeea1b (0)	Performance test took 21 minutes.	
Manual generation	GCP	OK	aeeea1b (0)	Manual generation works fine.	
Doxygen generation	GCP	OK	aeeea1b (0)	Doxygen generation works fine.	
ASE Calculator	GCP	OK	aeeea1b (0)	ASE commit be8e0ed works fine.	
AiiDA-CP2K Plugin	GCP	OK	aeeea1b (0)	aiida-cp2k commit 7cf6d76 works fine.	
i-Pi	GCP	OK	aeeea1b (0)	i-Pi commit 7608128 works fine.	
Gromacs QM/MM	GCP	OK	aeeea1b (0)	Gromacs commit cac191b works fine.	
Python Scripts	GCP	OK	aeeea1b (0)	Python tests passed	
Debian, i386 (ssmp)	GCP	OK	aeeea1b (0)	correct: 3688 / 3688; 20min	
Linux-gnu-aarch64.psmp (11.2.0)	Raspberry Pi 4	FAILED	9c6658e (-2)	correct: 3885 / 3892; wrong: 7; 634min	N/A
CRAY-XC40-gnu.psmp (11.2.0)	CSCS, Piz Daint	FAILED	aeeea1b (0)	correct: 3890 / 3891; wrong: 1; 26min	ee6c3aa (-1)
Performance CRAY-XC40	CSCS, Piz Daint	OK	ee6c3aa (-1)	empty	
CRAY-XC50-gnu.psmp (9.3.0)	CSCS, Piz Daint	OK	aeeea1b (0)	correct: 3891 / 3891; 94min	
Performance CRAY-XC50	CSCS, Piz Daint	OK	ee6c3aa (-1)	empty	

Multiple platforms/architectures available, including full logs and their arch-files.

REGTESTING



	Hit	Total	Coverage
Lines:	386171	457412	84.4 %
Functions:	9138	12154	75.2 %



directory	line cov.	functions
src	84.4 %	80.2 %
src/aobasis	74.3 %	82.2 %
src/arnoldi	83.8 %	52.3 %
src/base	78.0 %	78.1 %
src/common	71.0 %	61.4 %
src/dbcsrx	79.1 %	48.1 %
src/dbm	95.0 %	90.6 %
src/dbt	92.3 %	81.7 %
src/dbt/tas	90.2 %	92.8 %
src/emd	98.1 %	100.0 %
src/eri_mme	80.3 %	57.2 %
src/fm	82.5 %	82.1 %
src/grid	89.7 %	94.3 %
src/grid/common	87.4 %	90.2 %
src/grid/cpu	52.3 %	67.3 %
src/grid/ref	91.4 %	96.7 %
src/hfxbase	98.6 %	99.5 %
src/input	78.7 %	62.2 %
src/minimax	92.2 %	73.3 %
src/motion	85.1 %	86.7 %
src/motion/mc	82.0 %	78.0 %
src/motion/thermostat	76.3 %	85.2 %
src/mpiwrap	63.6 %	24.0 %
src/offload	67.4 %	58.3 %
src/pw	82.5 %	76.5 %
src/pw/fft	59.1 %	83.7 %
src/pw_env	88.8 %	84.6 %
src/shg_int	95.1 %	100.0 %
src/start	75.3 %	44.7 %
src/subsys	82.1 %	60.0 %
src/swarm	86.9 %	69.8 %
src/tmc	80.1 %	76.7 %
src/xc	61.1 %	75.8 %

Profiling: Timing Report

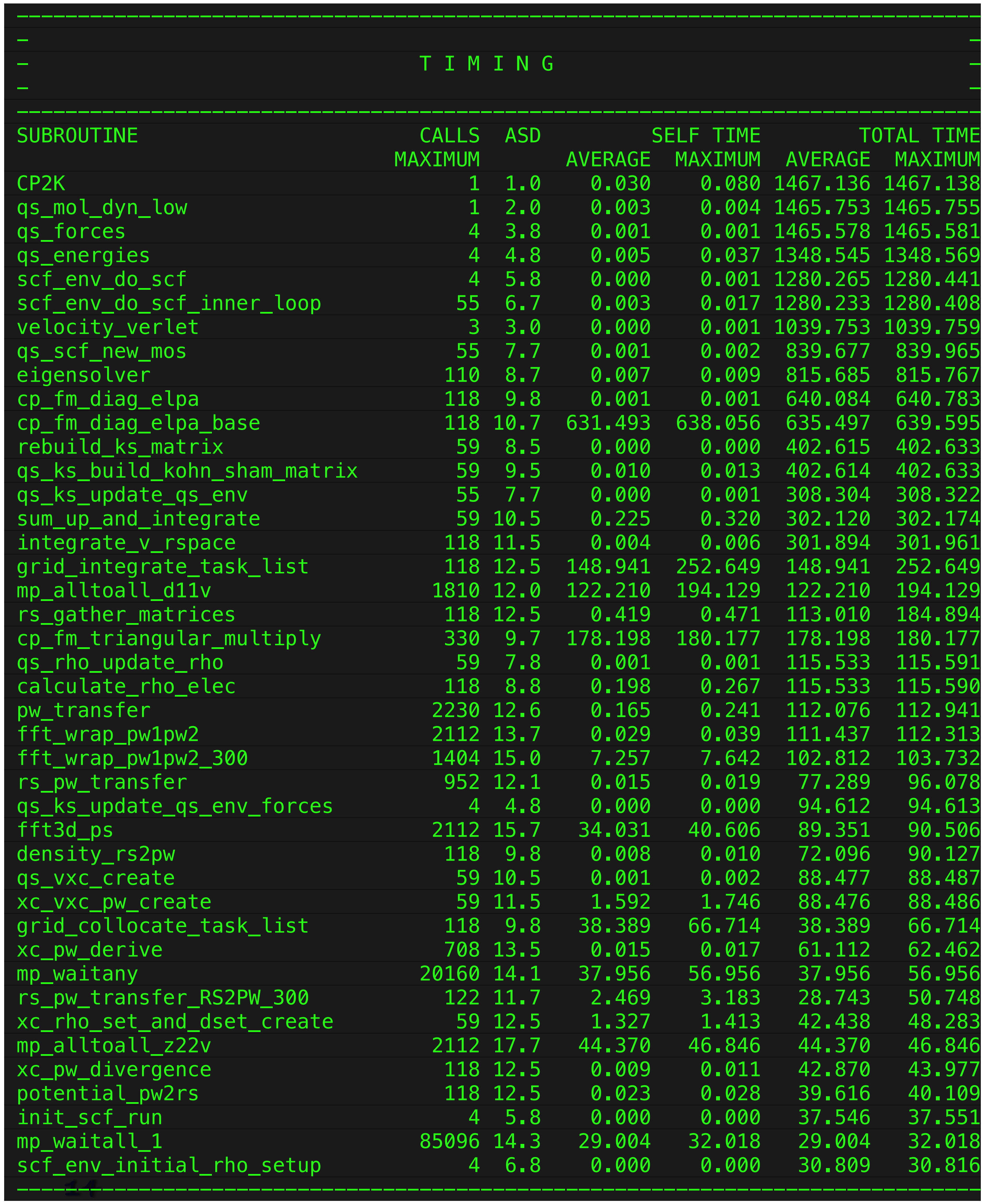
SUBROUTINE name contains method and step descriptors:

- pw Planewave
- fft Fast Fourier Transformation
- mp Message Passing (MPI)
- qs Quickstep
- scf Self-consistent field
- cp_fm Full matrix lin. alg. wrapper

ASD measure for how deeply nested a function is

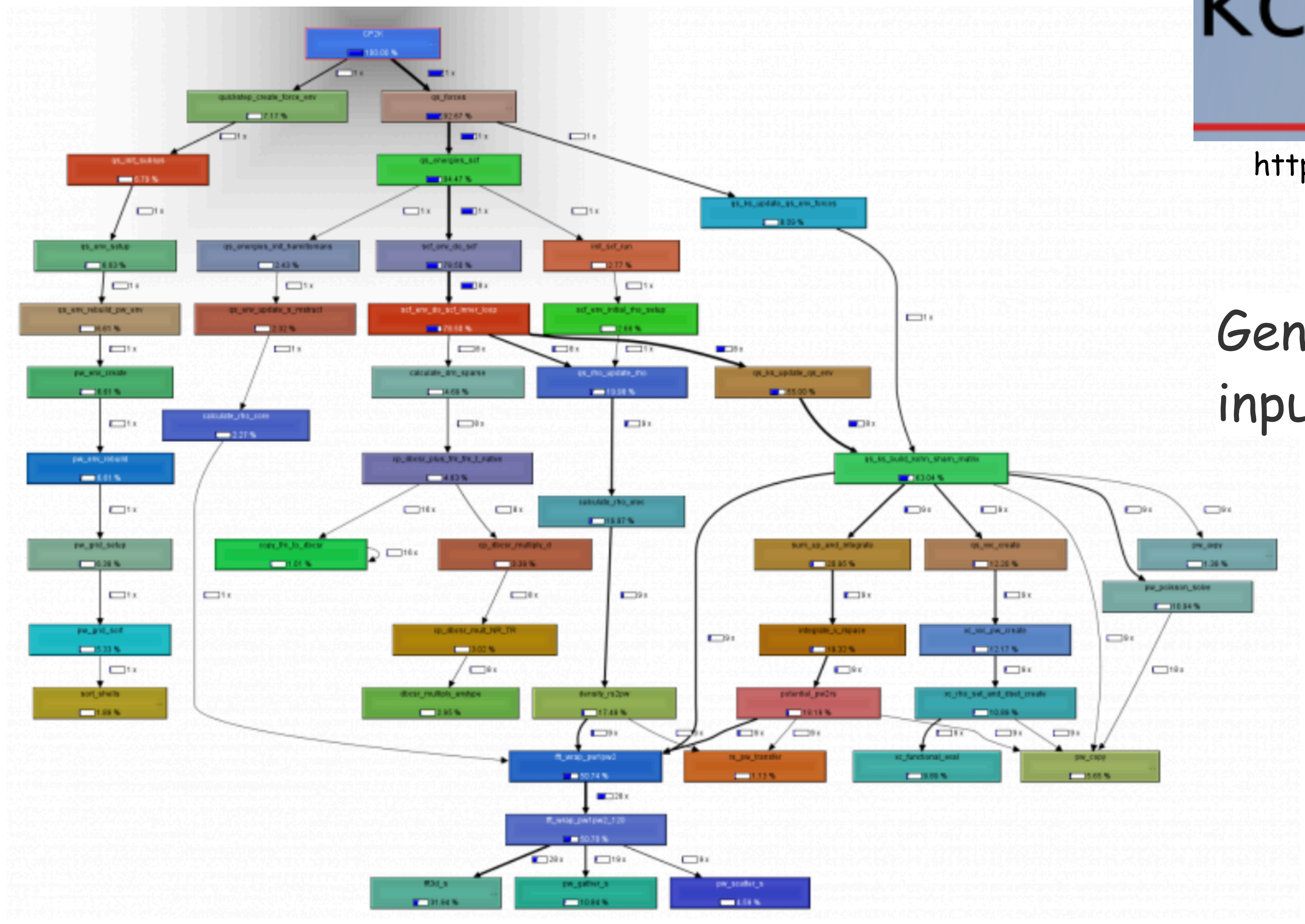
SELF TIME time spent in routine and non separately timed subroutines

```
CALL timeset(routineN,handle)
     ! Region to be timed
CALL timestop(handle)
```



```
--------------------------------------------------------------------------------
-                                                                              -
-                                T I M I N G                                   -
-                                                                              -
--------------------------------------------------------------------------------
SUBROUTINE                        CALLS  ASD         SELF TIME        TOTAL TIME
                                MAXIMUM       AVERAGE  MAXIMUM  AVERAGE  MAXIMUM
CP2K                                  1  1.0    0.030    0.080 1467.136 1467.138
qs_mol_dyn_low                        1  2.0    0.003    0.004 1465.753 1465.755
qs_forces                             4  3.8    0.001    0.001 1465.578 1465.581
qs_energies                           4  4.8    0.005    0.037 1348.545 1348.569
scf_env_do_scf                        4  5.8    0.000    0.001 1280.265 1280.441
scf_env_do_scf_inner_loop            55  6.7    0.003    0.017 1280.233 1280.408
velocity_verlet                       3  3.0    0.000    0.001 1039.753 1039.759
qs_scf_new_mos                       55  7.7    0.001    0.002  839.677  839.965
eigensolver                         110  8.7    0.007    0.009  815.685  815.767
cp_fm_diag_elpa                     118  9.8    0.001    0.001  640.084  640.783
cp_fm_diag_elpa_base                118 10.7  631.493  638.056  635.497  639.595
rebuild_ks_matrix                    59  8.5    0.000    0.000  402.615  402.633
qs_ks_build_kohn_sham_matrix         59  9.5    0.010    0.013  402.614  402.633
qs_ks_update_qs_env                  55  7.7    0.000    0.001  308.304  308.322
sum_up_and_integrate                 59 10.5    0.225    0.320  302.120  302.174
integrate_v_rspace                  118 11.5    0.004    0.006  301.894  301.961
grid_integrate_task_list            118 12.5  148.941  252.649  148.941  252.649
mp_alltoall_d11v                   1810 12.0  122.210  194.129  122.210  194.129
rs_gather_matrices                  118 12.5    0.419    0.471  113.010  184.894
cp_fm_triangular_multiply           330  9.7  178.198  180.177  178.198  180.177
qs_rho_update_rho                    59  7.8    0.001    0.001  115.533  115.591
calculate_rho_elec                  118  8.8    0.198    0.267  115.533  115.590
pw_transfer                        2230 12.6    0.165    0.241  112.076  112.941
fft_wrap_pw1pw2                    2112 13.7    0.029    0.039  111.437  112.313
fft_wrap_pw1pw2_300                1404 15.0    7.257    7.642  102.812  103.732
rs_pw_transfer                      952 12.1    0.015    0.019   77.289   96.078
qs_ks_update_qs_env_forces            4  4.8    0.000    0.000   94.612   94.613
fft3d_ps                           2112 15.7   34.031   40.606   89.351   90.506
density_rs2pw                       118  9.8    0.008    0.010   72.096   90.127
qs_vxc_create                        59 10.5    0.001    0.002   88.477   88.487
xc_vxc_pw_create                     59 11.5    1.592    1.746   88.476   88.486
grid_collocate_task_list            118  9.8   38.389   66.714   38.389   66.714
xc_pw_derive                        708 13.5    0.015    0.017   61.112   62.462
mp_waitany                        20160 14.1   37.956   56.956   37.956   56.956
rs_pw_transfer_RS2PW_300            122 11.7    2.469    3.183   28.743   50.748
xc_rho_set_and_dset_create           59 12.5    1.327    1.413   42.438   48.283
mp_alltoall_z22v                   2112 17.7   44.370   46.846   44.370   46.846
xc_pw_divergence                    118 12.5    0.009    0.011   42.870   43.977
potential_pw2rs                     118 12.5    0.023    0.028   39.616   40.109
init_scf_run                          4  5.8    0.000    0.000   37.546   37.551
mp_waitall_1                      85096 14.3   29.004   32.018   29.004   32.018
scf_env_initial_rho_setup             4  6.8    0.000    0.000   30.809   30.816
--------------------------------------------------------------------------------
```

Profiling: CALLGRAPH



kcachegrind

CallGraphViewer

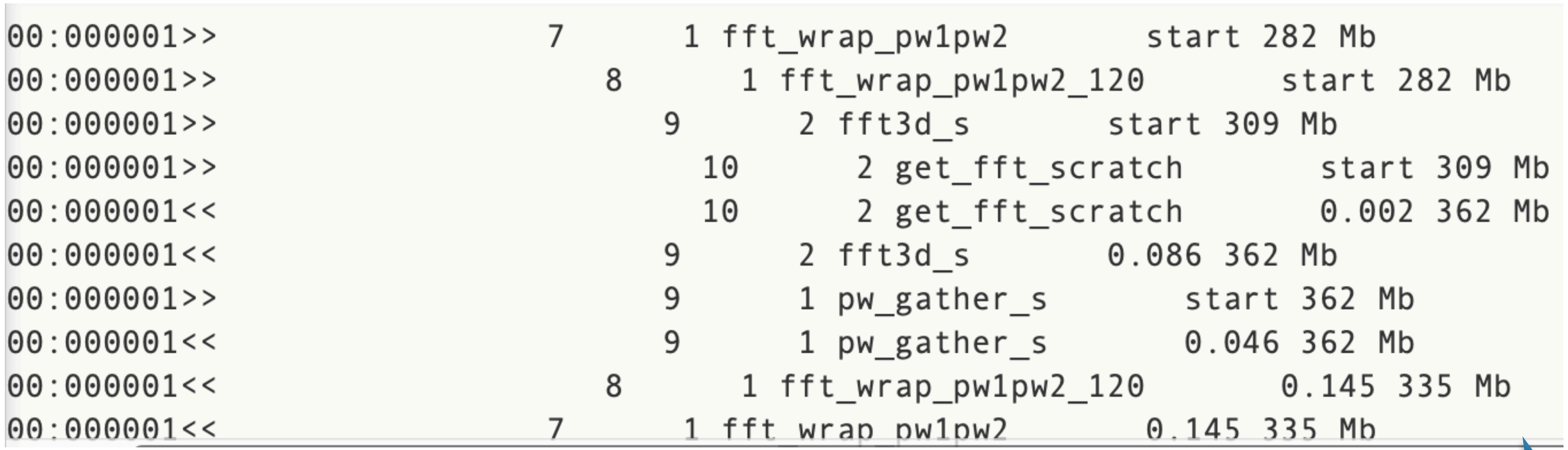
https://kcachegrind.sourceforge.net/

Generated by including the input keyword CALLGRAPH

```
valgrind --tool=callgrind ./cp2k.sopt -i test.inp -o test.out
```

CP2K Trace

Generated by including the input keyword **TRACE**



```
00:000001>>                    7       1 fft_wrap_pw1pw2       start 282 Mb
00:000001>>                       8       1 fft_wrap_pw1pw2_120       start 282 Mb
00:000001>>                          9       2 fft3d_s       start 309 Mb
00:000001>>                            10       2 get_fft_scratch       start 309 Mb
00:000001<<                            10       2 get_fft_scratch       0.002 362 Mb
00:000001<<                          9       2 fft3d_s       0.086 362 Mb
00:000001>>                          9       1 pw_gather_s       start 362 Mb
00:000001<<                          9       1 pw_gather_s       0.046 362 Mb
00:000001<<                       8       1 fft_wrap_pw1pw2_120       0.145 335 Mb
00:000001<<                    7       1 fft_wrap_pw1pw2       0.145 335 Mb
```

subroutine name

stack depth (properly aligned)

call count

either 'start' at entry of or a number at exit, which is the time

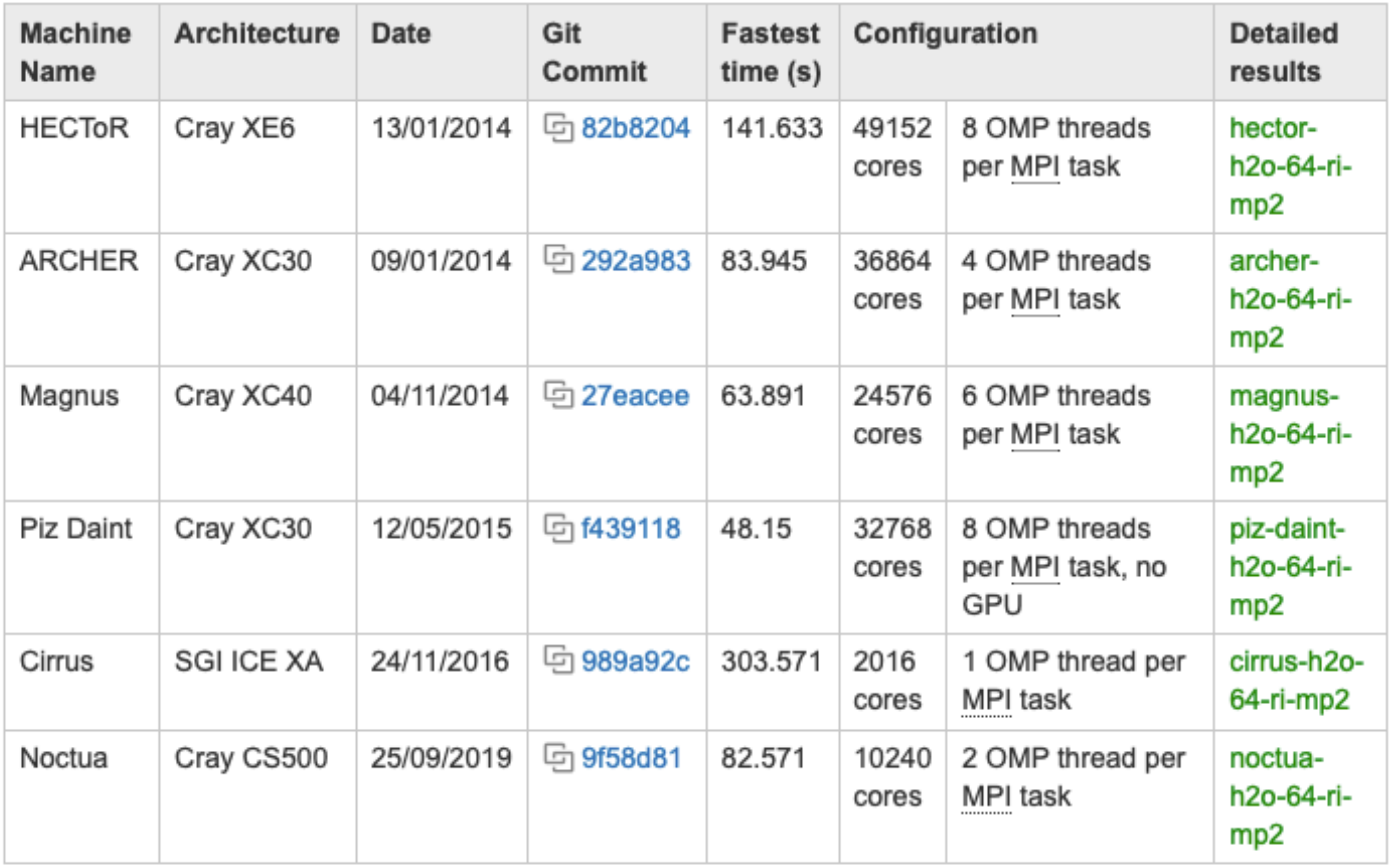
measure of the process memory used at the time of timeset/ timestop.

Useful as debugging tool, but verbose (TRACE_MAX, TRACE_ROUTINES)

Performance: CP2K Benchmark Suite

H2O-64 AIMD; 28000 atoms iron-silicate MMMD; 216 atoms LiH-HFX GAPW; 2048 H2O DFT-LS; H2O-64-RI-MP2

H2O-64-RI-MP2



Machine Name	Architecture	Date	Git Commit	Fastest time (s)	Configuration		Detailed results
HECToR	Cray XE6	13/01/2014	82b8204	141.633	49152 cores	8 OMP threads per MPI task	hector-h2o-64-ri-mp2
ARCHER	Cray XC30	09/01/2014	292a983	83.945	36864 cores	4 OMP threads per MPI task	archer-h2o-64-ri-mp2
Magnus	Cray XC40	04/11/2014	27eacee	63.891	24576 cores	6 OMP threads per MPI task	magnus-h2o-64-ri-mp2
Piz Daint	Cray XC30	12/05/2015	f439118	48.15	32768 cores	8 OMP threads per MPI task, no GPU	piz-daint-h2o-64-ri-mp2
Cirrus	SGI ICE XA	24/11/2016	989a92c	303.571	2016 cores	1 OMP thread per MPI task	cirrus-h2o-64-ri-mp2
Noctua	Cray CS500	25/09/2019	9f58d81	82.571	10240 cores	2 OMP thread per MPI task	noctua-h2o-64-ri-mp2

High-Performance Computing

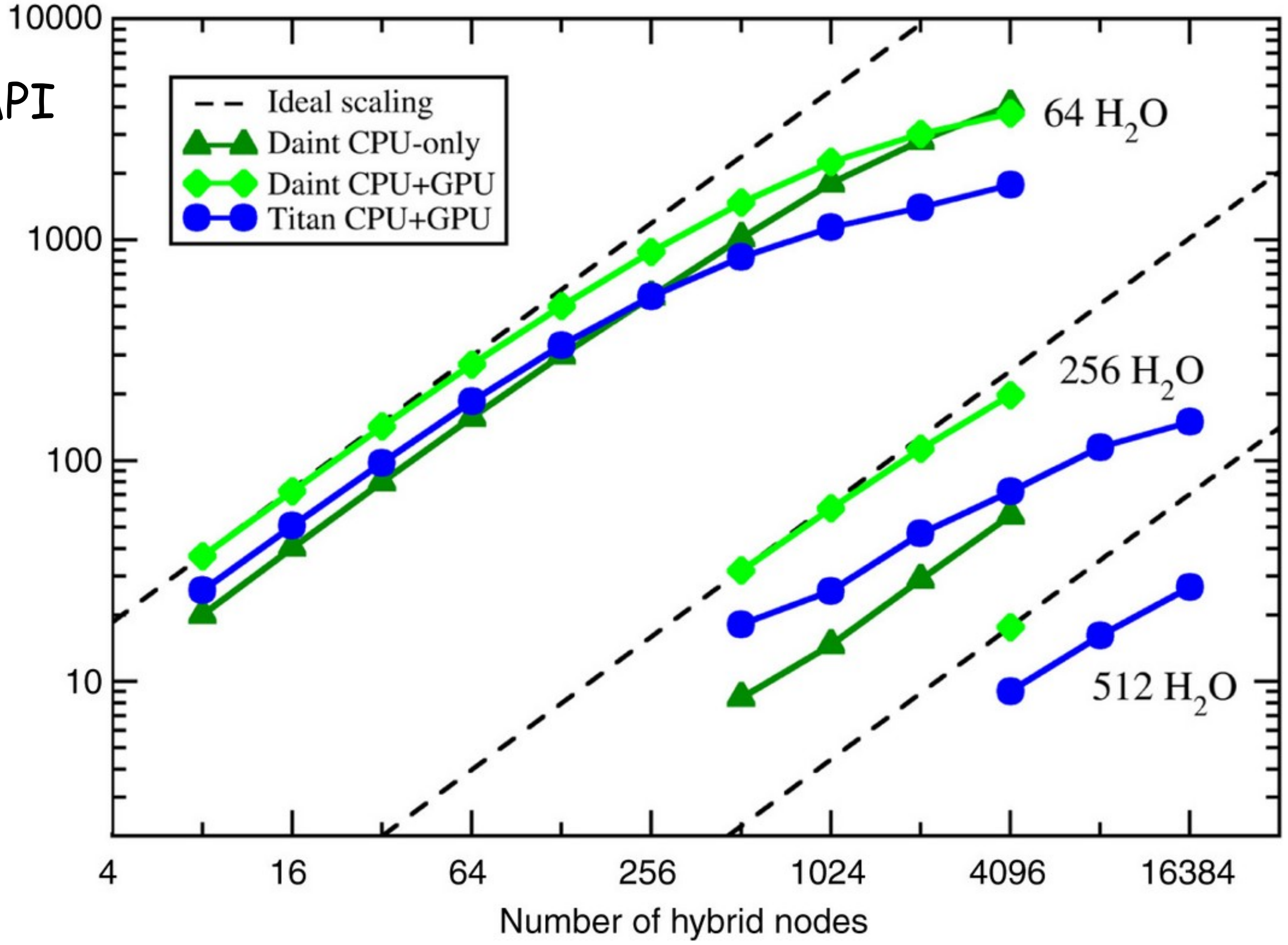
Massive Parallelization:

→ Coarsed grain distributed memory using MPI

→ Fine grain shared memory using OpenMP

→ Accelerator code for Nvidia/AMD/Intel

RPA



High-Performance Libraries:

→ DBCSR

→ Tensor DBCSR - sparse tensor contraction

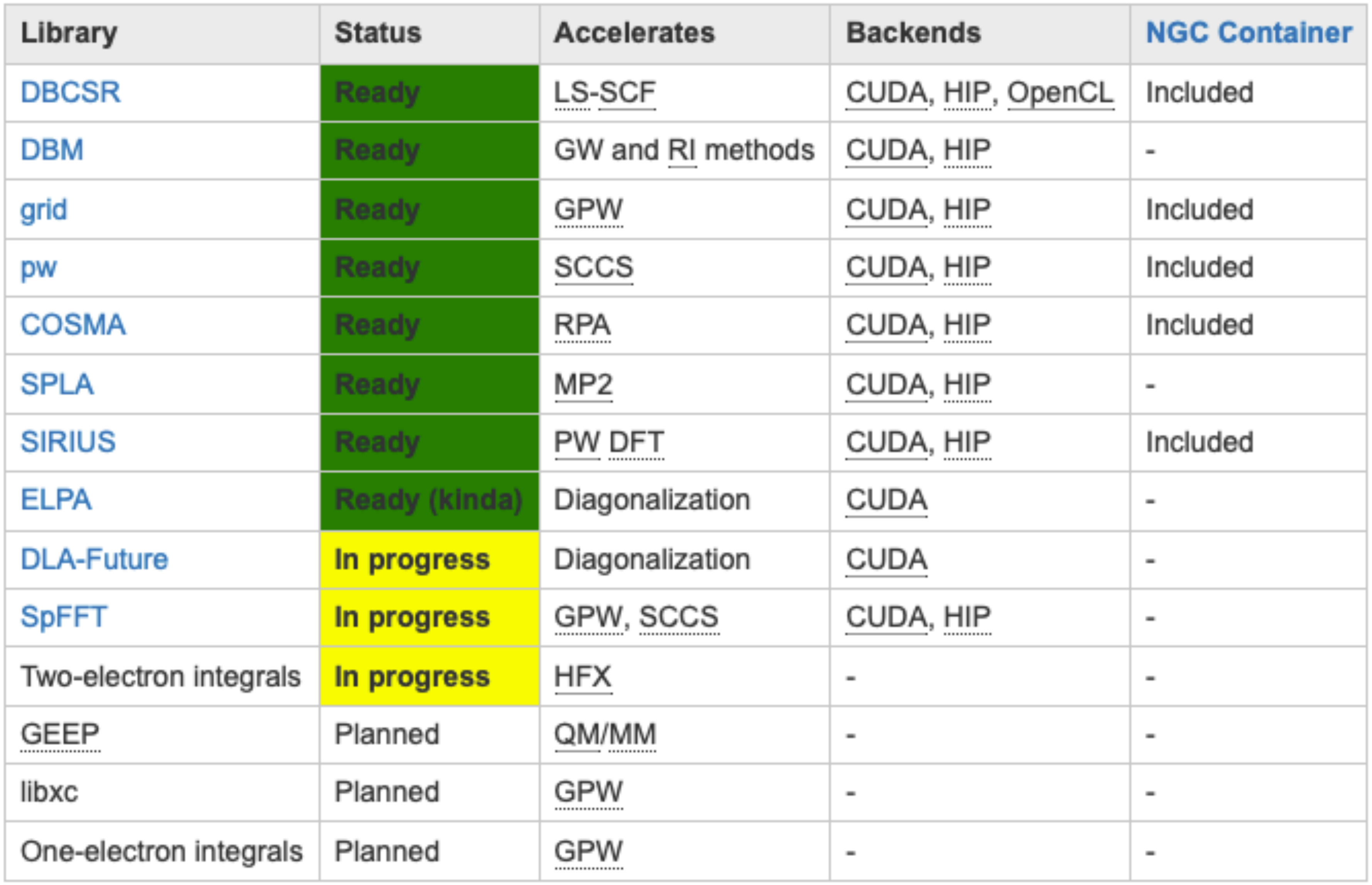
→ COSMA full matrix multiplication

→ Grid library: collocate/integrate Gaussian on grids

COSMA
Communication-Optimal
Matrix-Multiplication

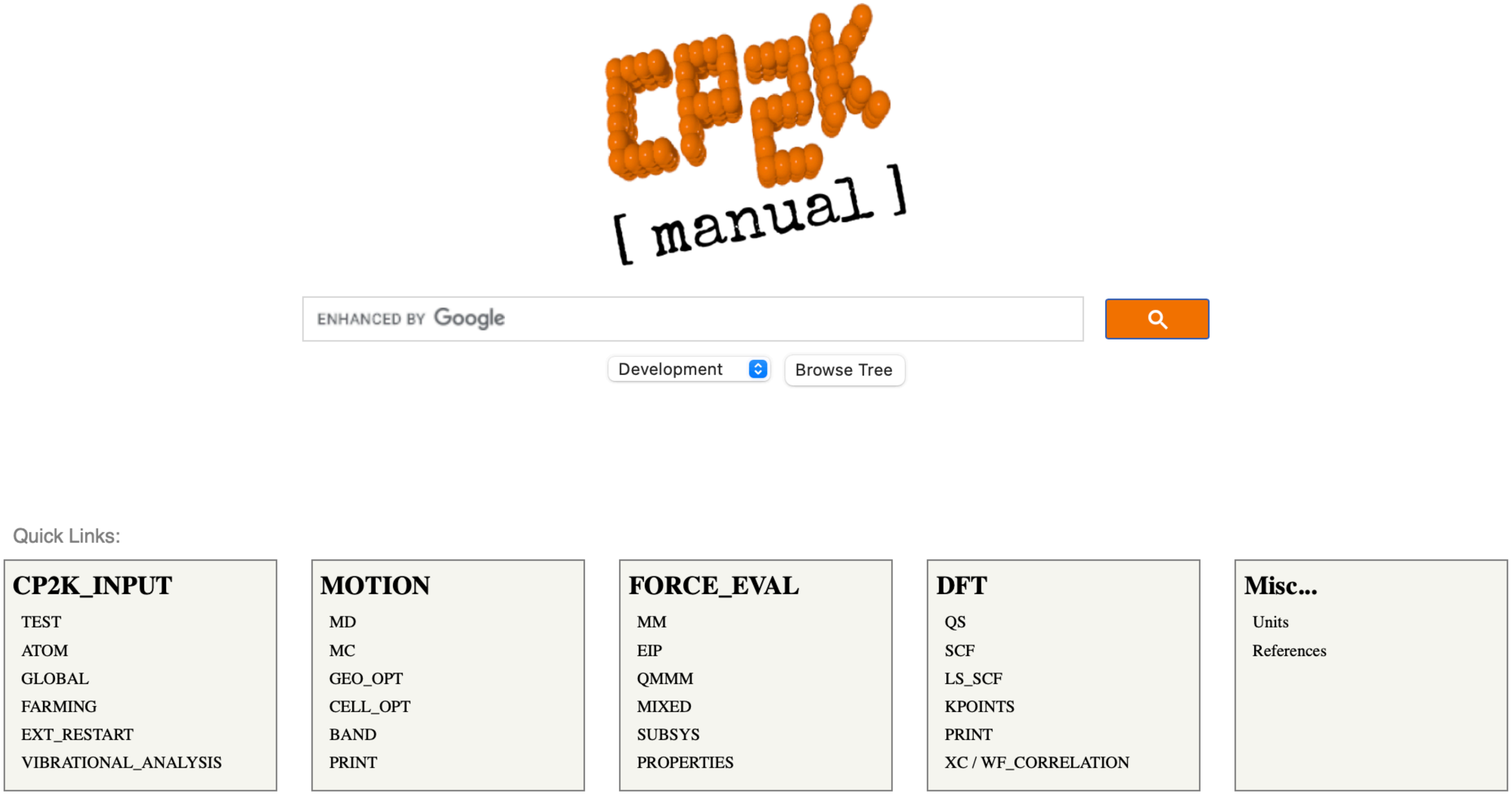
https://github.com/eth-cscs/COSMA

GPU Support



Library	Status	Accelerates	Backends	NGC Container
DBCSR	Ready	LS-SCF	CUDA, HIP, OpenCL	Included
DBM	Ready	GW and RI methods	CUDA, HIP	-
grid	Ready	GPW	CUDA, HIP	Included
pw	Ready	SCCS	CUDA, HIP	Included
COSMA	Ready	RPA	CUDA, HIP	Included
SPLA	Ready	MP2	CUDA, HIP	-
SIRIUS	Ready	PW DFT	CUDA, HIP	Included
ELPA	Ready (kinda)	Diagonalization	CUDA	-
DLA-Future	In progress	Diagonalization	CUDA	-
SpFFT	In progress	GPW, SCCS	CUDA, HIP	-
Two-electron integrals	In progress	HFX	-	-
GEEP	Planned	QM/MM	-	-
libxc	Planned	GPW	-	-
One-electron integrals	Planned	GPW	-	-

The Manual



Quick Links:

CP2K_INPUT	MOTION	FORCE_EVAL	DFT	Misc...
TEST	MD	MM	QS	Units
ATOM	MC	EIP	SCF	References
GLOBAL	GEO_OPT	QMMM	LS_SCF	
FARMING	CELL_OPT	MIXED	KPOINTS	
EXT_RESTART	BAND	SUBSYS	PRINT	
VIBRATIONAL_ANALYSIS	PRINT	PROPERTIES	XC / WF_CORRELATION	

Documents every possible CP2K input keyword

Mostly with helpful descriptions

cp2k.sopt --html-manual

Special Features

Interfaces: **i-Pi**, ASE, AiiDA, PyRetis, Travis, **PLUMED**, SIRIUS, PhonoPy, **OMEN**, GROMACS

Library: libcp2k

Farming: run many jobs in parallel

Atomic code: Pseudopotential generation

Automatic optimization of input parameters

Scripted input

Basis set optimization on molecules (MOLOPT)

Automatic numerical debugging of forces

CP2K+PLUMED

CP2K has an already built-in code for Free-energy calculations and Metadynamics

As alternative, the user can run free-energy calculations using PLUMED

PLUMED is already included in CP2K in version 2.7

CP2K uses PLUMED version 2.x

Download and install PLUMED

ModifyCP2K arch file

https://www.plumed.org

```
include /path/to/your/plumed2.0/installation/lib/plumed/src/lib/Plumed.inc
EXTERNAL_OBJECTS=$(PLUMED_STATIC_DEPENDENCIES)
```

`-D__PLUMED2` should be added to your DFLAGS and `-lz -ldl -lstdc++` to your LIBS variable

```
&METADYN
     USE_PLUMED .TRUE.
     PLUMED_INPUT_FILE ./filename.inp
&END METADYN
```

CP2K + i-Pi

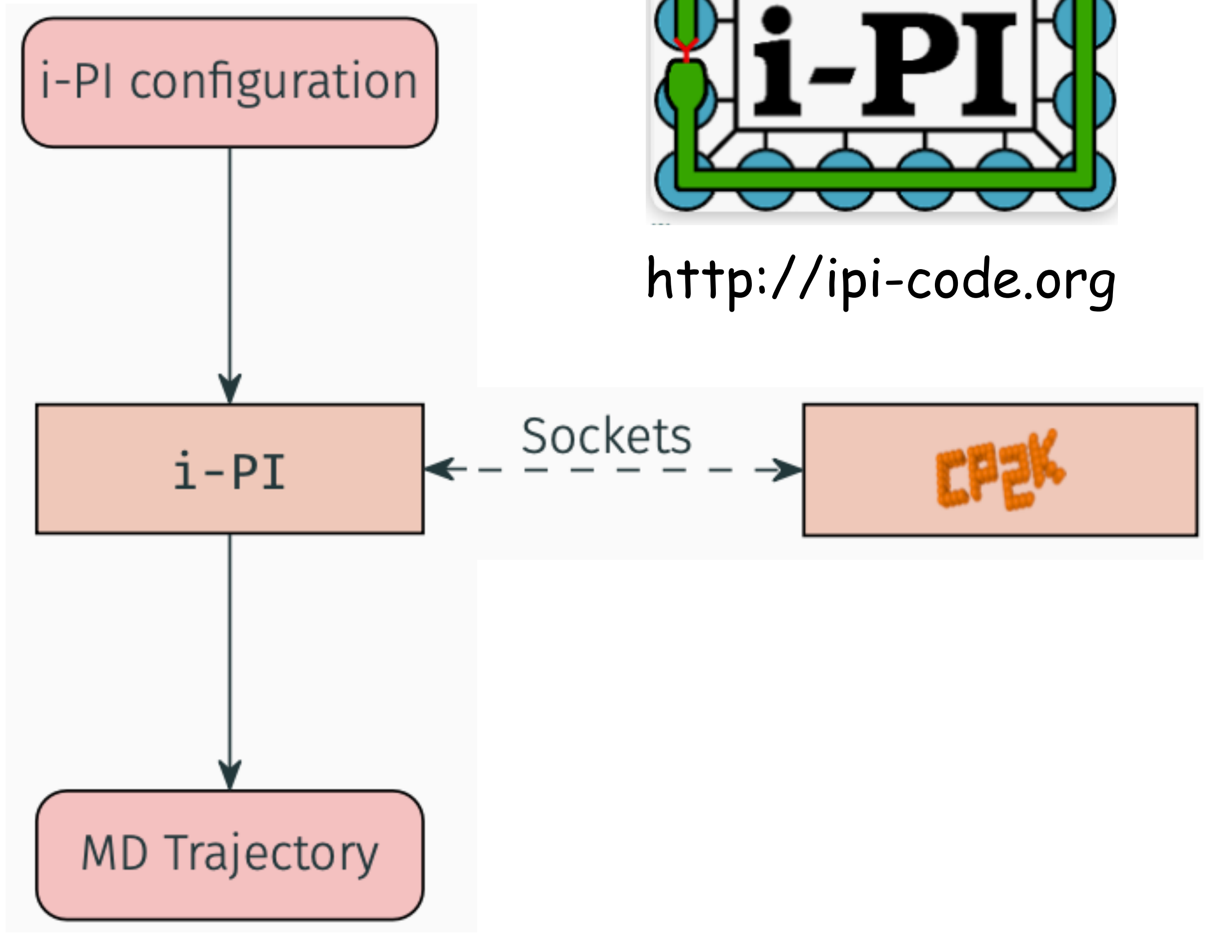
i-Pi is an interface for ab initio path integral MD

- Python based
- Focus on Path Integral Molecular Dynamics
- Communication with Force Engines via network sockets
- Many additional methods available

```
source ${PATH_TO_IPI}/env.sh
```

```
i-pi input.xml > log &
```

http://ipi-code.org



CP2K as the client code using an **internet/unix** domain socket on the host address "host_address" and on the port number "port" the following lines must be added to its input files:

```
&MOTION
...
   &DRIVER
      HOST host_address
      PORT port
      …
   &END DRIVER
   ...
&END MOTION
```

```
<ffsocket mode='unix' name='cp2k'>
    <address>host_address</address>
    <port>port</port>
    <latency>0.01</latency>
    <timeout>5000</timeout>
</ffsocket>
<system>
      <forces>
         <force forcefield='cp2k'>
         </force>
      </forces>
```

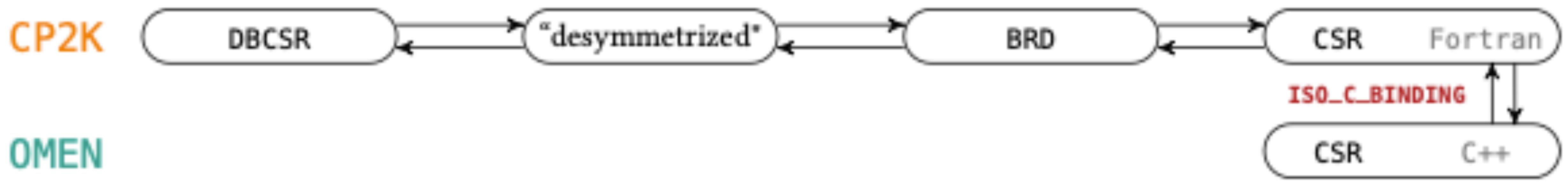
CP2K + OMEN

OMEN is a massively parallel, multi-dimensional, atomistic, and full-band simulation tool
to treat **electron, hole and phonon transport** in bulk, quantum well, and nanowire structures
NEGF and wave-function formalisms to solve the SE under **open boundary conditions**

OMEN https://github.com/saschabrueck/dft-transport

Target **libcp2k** : it can be used as a library

```
> make -j N ARCH=Linux-x86-64-gfortran VERSION=popt libcp2k
```



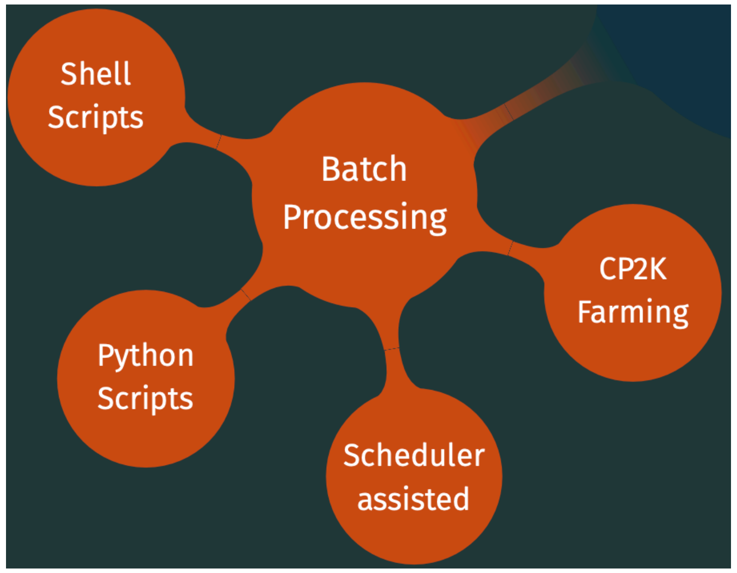
ISO C BINDING to call routines from C.

C function pointer passed to CP2K

OMEN takes S and H as input and outputs P

C pointer in SCF: calculation using S and H created from CP2K and P evaluated by OMEN

Run Automation: Farming



- Jobs are run inside the same CP2K process
- MPI gets initialized once reducing startup time
- Useful for many small jobs

```
&GLOBAL
    ROJECT OldMacDonald
    PROGRAM FARMING
     RUN_TYPE NONE
&END GLOBAL
&FARMING
   NGROUPS 2 ! number of parallel jobs
   MASTER_SLAVE ! for load balancing
   GROUP_SIZE 42 ! number of processors per group, default: 8
   &JOB
        JOB_ID 1 ! optional, required for dependencies
        DIRECTORY dir-1
        INPUT_FILE_NAME water.inp
        OUTPUT_FILE_NAME water.out
   &END JOB
   &JOB
      DEPENDENCIES 1
      DIRECTORY dir-2
      INPUT_FILE_NAME water.inp
      OUTPUT_FILE_NAME more_water.out
   &END JOB
[...]
&END FARMING
```

Workflows: AiiDA

- Python-based
- Strong focus on Data Provenance
- Database backend (PostgreSQL) + File Repository
- Advanced workflow engine on top of Python
- Plugin architecture:
  - CP2K Plugin
  - Gaussian Basis Set and Pseudopotential Plugin
  - more in the AiiDA Plugin Registry

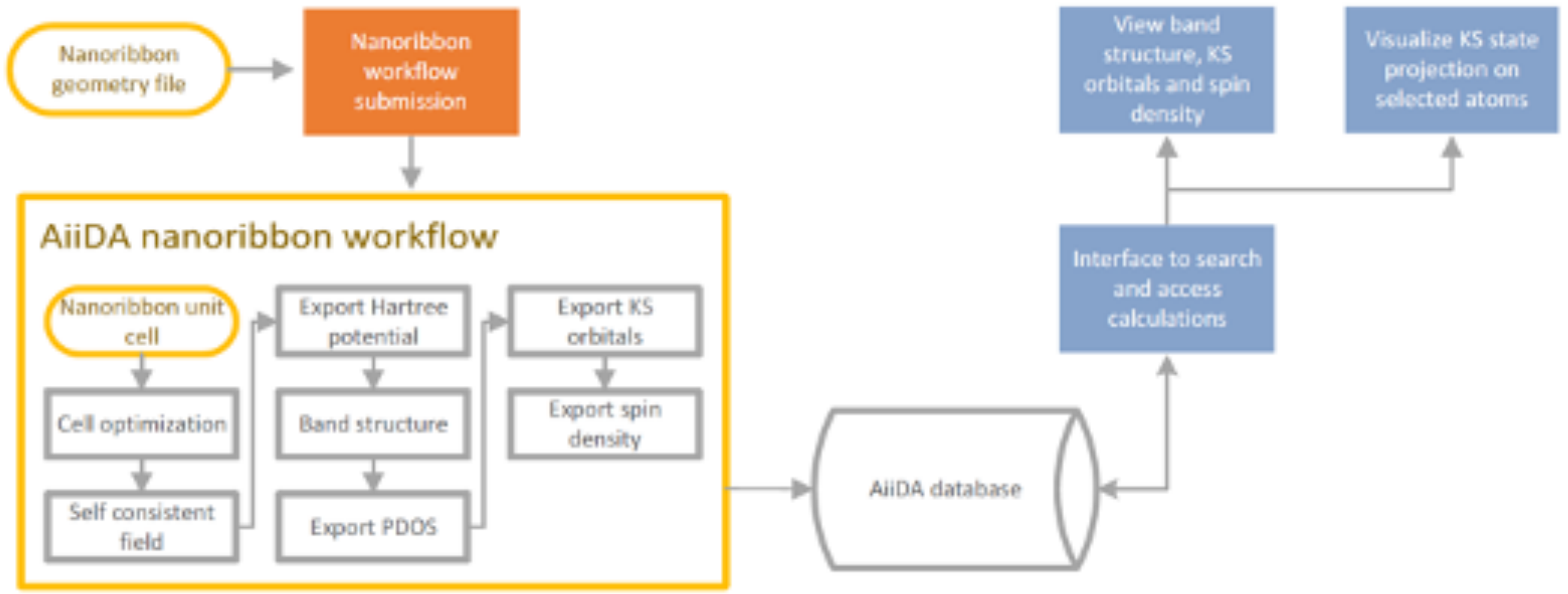
Jupyter Notebook integration

Integration with the MaterialsCloud Open Science platform

AiiDAlab

- can be run on a web platform & without computational or coding expertise
- perform HPC calculations automatically
- straightforward to develop (Python), share & publish (App Store)

Nanoribbons app
nanotech@surface
EMPA


Nanoribbon geometry file
Nanoribbon workflow submission
AiiDA nanoribbon workflow
Nanoribbon unit cell
Cell optimization
Self consistent field
Export Hartree potential
Band structure
Export PDOS
Export KS orbitals
Export spin density
AiiDA database
Interface to search and access calculations
View band structure, KS orbitals and spin density
Visualize KS state projection on selected atoms

CP2K Developers

https://www.cp2k.org ....Exercises,LectureSlides

https://manual.cp2k.org ....InputFilereference

<CP2K-SOURCE>/tests .... MinimalWorkingExamples

https://groups.google.com/group/cp2k . . . . Google Group/Forum

https://github.com/cp2k/cp2k/issues ....IssueTracker

https://github.com/orgs/cp2k/teams/cp2k-developers ....Discussions, meetings

Tiziano Müller Christian Plessl Ole Schütt Vedran Miletić
Anton Kozhevnikov Urban Borštnik Yannick Misteli Jiannan Liu
Edward Ditler Michael Banck M. H. Bani-Hashemian
Asdrubal Lozada-Blanco Jan Janssen Joost VandeVondele
ArnoP Pibemanden Alfio Lazzaro Patrick Seewald Hans Pabst
Andreas Glöß Ralph Koitz Iain Bethune Jan Wilhelm Abhishek Bagusetty
Christian S. Ahart Matthias Krack Michael Lass Eisuke Kawashima
Aliaksandr Yakutovich Matt Wattinks Juerg Hutter Sergey Chulkov
Xin Wu Christoph Schran sivkov Nico Holmberg Shoshana Jakobovits
Patrick Melix Samuel Andermatt Rangsiman Ketkaew Fabian Belleflamme
Augustin Bussy Fao LaN Simon Frasch Marko Kabic
Mathieu Taillefumier @DCM-Uni-Paderborn Anna Hehn Dmitry Ryndyk
Marcella Iannuzzi Vladimir Rybkinjr Dorothea Golze Herbert Forbert
Frederick Stein Holly Judge Jan Mattiat