Cross-Architecture Programming for Accelerated Compute; Freedom of Choice for Hardware

Fortran, OpenMP offload to GPU Demo

Supporting OpenMP Standard, Intel’s OMP RT Implementation

oneAPI

intel

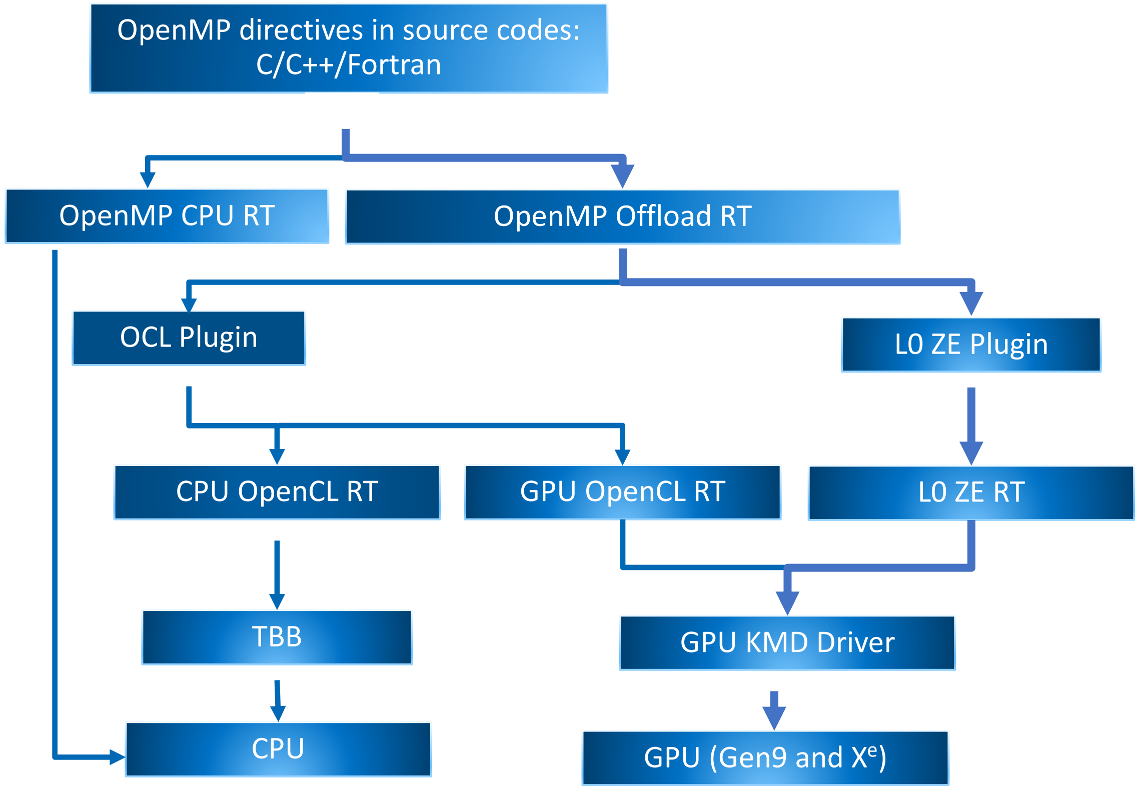
Introducing Intel® Fortran Compiler

Fortran Standard

- Fortran 77, 90, 95 and 2003 fully supported
- vast* support to Fortran 2008 and Fortran 2018
- vast* support to OpenMP 4.5, 5.0 and 5.1
- JIT and AOT compilation

OpenMP

- managed by a nonprofit consortium
- multi-platform API for shared-memory multiprocessing
- portable and scalable
- pragma-based implementation is easy to...
  - read
  - debug
  - incrementally port
  - toggle between serial and parallel versions
- support offloading code to "devices" (e.g. FPGAs, GPUs, etc)



* see more at https://www.intel.com/content/www/us/en/developer/articles/technical/fortran-language-and-openmp-features-in-ifx.html

GRILLIX & PARALLAX

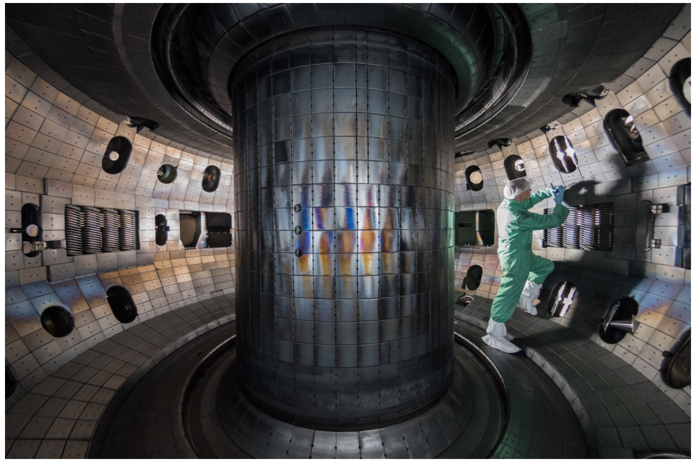
Both projects are funded by EUROFusion and developed in by the Max Planck Society, Germany. The development started in 2016 using Fortran 2008 and hybrid OpenMP+MPI parallelism.

PARALLAX

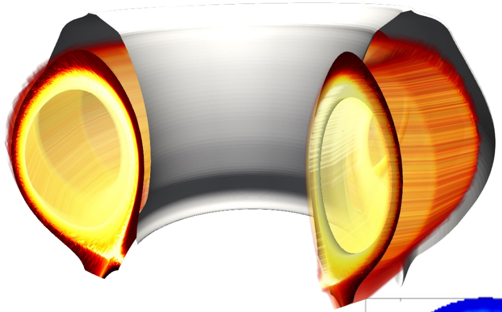
- flux-coordinate-independent (FCI) framework for use in other projects, such as GRILLIX and GENEX
- implements datatypes, numerical methods and provides interfaces for external libraries
- intent to simplify and unify workflow of plasma physics applications

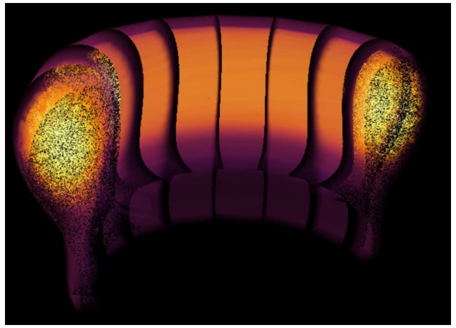
GRILLIX

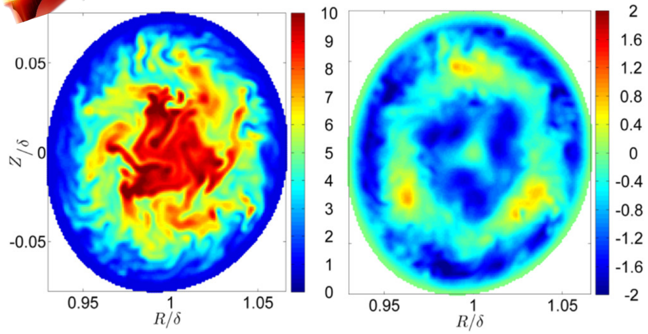
- 3d edge-fluid turbulence for Tokamak magnetic confinement devices (fusion reactor)
- flux-coordinate approach with toroidally staggered grids
- includes e.g. the global version of the drift reduced Braginskii model
- Further reading: https://aip.scitation.org/doi/10.1063/1.5089864



By Rswilcox - Own work, CC BY-SA 4.0,
https://commons.wikimedia.org/w/index.php?curid=73919081

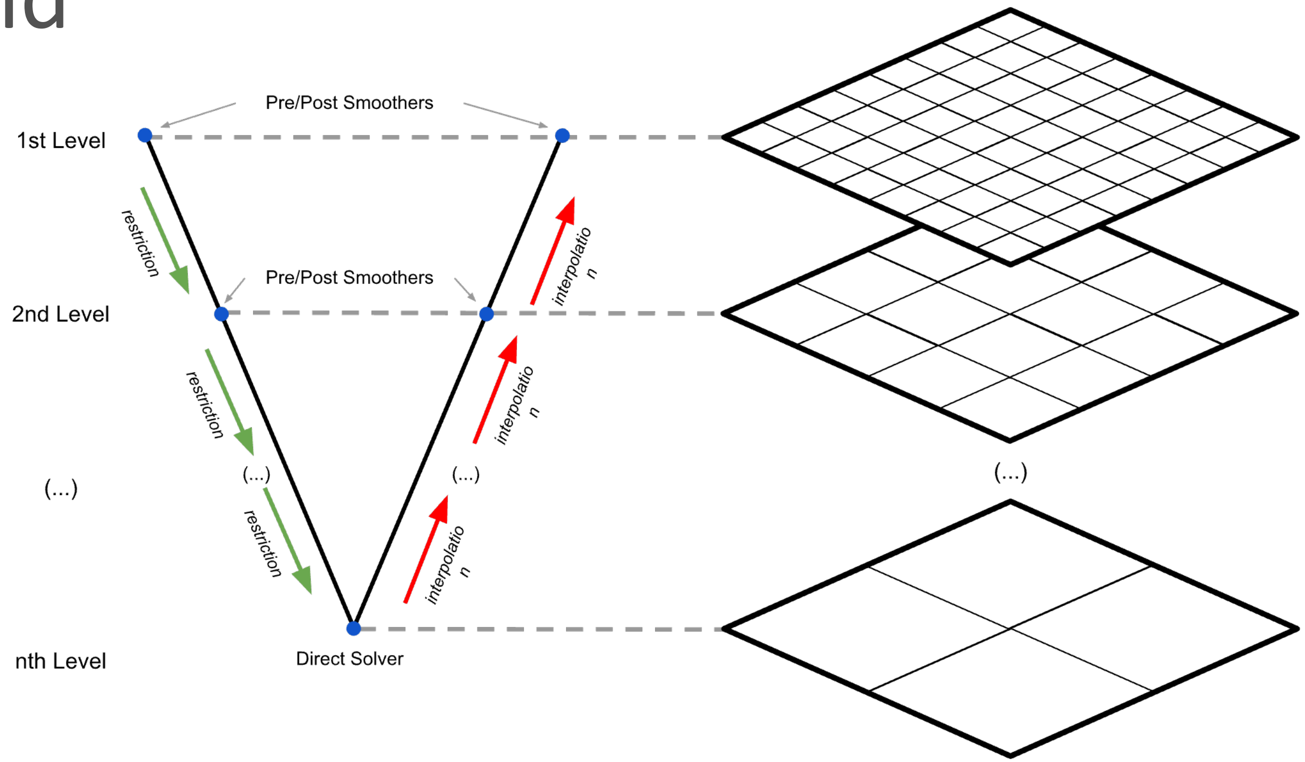






Steigmer et al.
https://iopscience.iop.org/article/10.1088/1361-6587/aaa373

GRILLIX/PARALLAX Multigrid Solver



- composed of several individual components
  - incremental migration possible!
- solver invokes different components in a hierarchical scheme
  - multiple calls -> most expensive part of the computation

GPU-Porting Workflow

- OpenMP allows gradual porting of individual components
- clean syntax – the **code looks simple!**
- performance must be measured "by kernel" until the full solver is completed
  - Special care to avoid copying unnecessary data to devices
- OpenMP is a portable standard – one code base for all backend
- Intel Fortran Compiler provides a backend for Intel GPUs

```fortran
if (first_call) then
   first_call = .false.
   !$omp target enter data map(to: ai, aj, aval, b, dinv, &
   !$omp    rbidx, nred, nblack, ndim)
end if

!$omp target data map(to: n_iter) map(tofrom: x)
do it = 1, n_iter
   !$omp target teams distribute parallel do private(ax, ll, ii, kk)
   do kk = lb_red, ub_red
      ll = rbidx(kk)
      ax = 0.0
      do ii = ai(ll), ai(ll+1)-1  ! always 5 iterations
         ax = ax + aval(ii)* x(aj(ii))
      end do
      x(ll) = x(ll) + (b(ll) - ax) * dinv(ll)
   Enddo

   !$omp target teams distribute parallel do private(ax, ll, ii, kk)
   do kk = lb_black, ub_black
      (...)
   Enddo

   !$omp target teams distribute parallel do private(ax, ll, ii, kk)
   do kk = lb_ghost, ub_ghost
      (...)
   enddo
(…)
enddo
```

Conclusions & Highlights

- Available in the Intel® oneAPI HPC Toolkit 2022.3.
- Production-ready for CPUs and Intel GPUs.
- Based on ifort frontend and runtime libraries and uses LLVM backend compiler technology.
- In addition to Fortran 2018, ifx also supports from FORTRAN 77 to Fortran 2008, all main versions of Fortran language standards.
- Supports OpenMP 4.5 and OpenMP 5.x directives and GPU offloading features.

This robust implementation provides Fortran programmers access to many capabilities of Intel Data Center GPUs right from their native language.