

Update on code refactoring

Daan Degrauwe,
Philippe Marguinaud,
Ludovic Auger

Reminder: GPU porting strategy in ACCORD

Following (and relying heavily on!) developments by Meteo-France and ECMWF

3 key ingredients

- usage of “smart” (i.e. GPU-aware) structures to handle data transfers between CPU and GPU: Field API
- Using source-to-source translation tools to generate GPU-code from CPU-code
- Using highly-optimized platform-specific libraries when available (e.g. FFT, BLAS)

This strategy was chosen to achieve balance between

- Portability and performance on new hardware platforms
- Maintainability of the code
- Limiting impact on scientists

Building for GPUs

- Integration of fxtran source to source transformations in the build system
- Use a compiler wrapper:
 - Compile original source
 - Perform source-to-source transformation
 - Compile GPU code
 - Merge object files
- Directive based pre-processing
- Flexible and powerful mechanism: compatible with ial-bundle/cmake, but also e.g. with gmkpack

Porting to NVIDIA GPUs

- Very productive collaboration
- Good optimisation level on NVIDIA for ARPEGE
- Still develop & maintain on this platform

Porting to AMD GPUs

(Work in progress!)

- Introduce OpenMP target in the fxtran source-to-source translation tools
- Use Field API on AMD (ported by ECMWF, A. Nawab)
- Use AMD ROCM compiler instead of NVIDIA NVHPC compiler
- (Long) list of bugs submitted, still open issues, e.g. AMD (non-standard) linker is failing on MASTERODB

Recent progress on refactoring

- IO
- Setup
- ARPEGE and NH-LAM dynamics, including semi-Lagrangian, semi-implicit, lateral boundary conditions
- AROME and HARMONIE-AROME parameterizations

Very hard to progress :

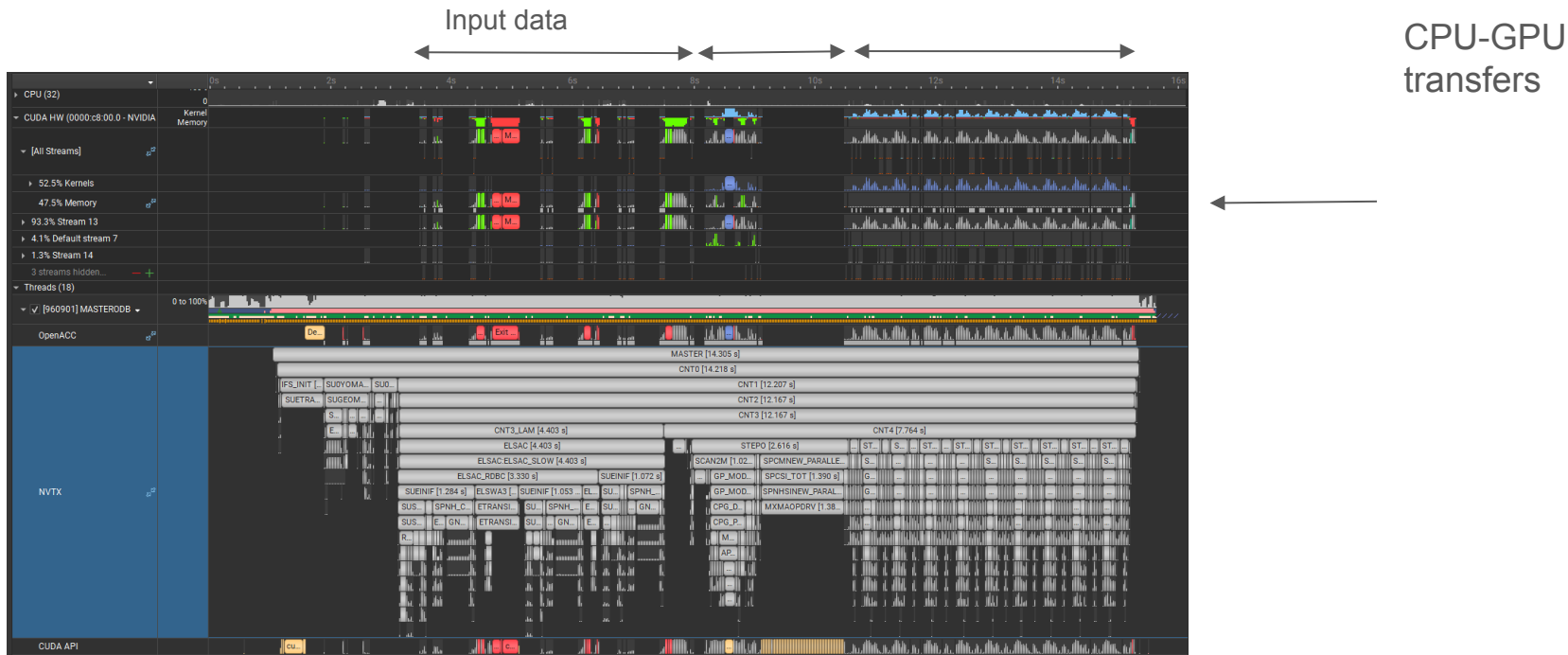
- without continuous integration
- without coordination with ECMWF

AROME

- Refactoring almost finished
 - Need little extra work on high level routines
 - Need a few adaptations on PHYEX code
- Already standalone test cases for shallow_mf, ice_ajust, turb; ported to NVIDIA
- Need cycle 50t2 and recent changes for rain_ice

ALARO

- Refactoring already finished before
- Milestone: full timestep running on GPU (i.e. without significant CPU-GPU transfers), but with quite some dirty hacks!



ALARO

- Refactoring already finished before
- Milestone: full timestep running on GPU (i.e. without significant CPU-GPU transfers), but with quite some dirty hacks!
- Consolidation in progress:
 - Port to CY50
 - Use ial-bundle/cmake build system with source-to-source translation integrated
 - Remove dirty tricks, mainly by using Field API at more places

SURFEX

- Not GPU friendly in its current shape, e.g.
 - Array syntax
 - No separation between IO, communications, memory management
- Being refactored:
 - Remove GPU-unfriendly aspects
 - Limited to NWP configurations
- Completion date unknown

More work in progress ...

- ECRAD
 - Loki port from ECMWF hopefully available soon
- fiat
 - GPU aware communications in MPL library
 - Dedicated memory for communications
- Ectrans
 - Existing interface (E)INV_TRANS/(E)DIR_TRANS uses raw Fortran arrays
 - Not compatible with Field API wrapping of GMV, GFL
 - Field API - based interface for ectrans under development

	ARPEGE	AROME	ALARO	Comments
Grid-point dynamics	Yes	Yes	Yes	Transform with fxtran
Spectral dynamics	Yes	Yes	Yes	Transform with fxtran
Physics	Yes	Test cases	Yes	Transform with fxtran
Surface	Old ISBA; SURFEX not ported	SURFEX not ported	Old ISBA; SURFEX not ported	Transform with fxtran (someday)
ECTRANS	Interface with Field API	Interface with Field API	Interface with Field API	Under development
ECRAD	Soon try ECMWF port	Soon try ECMWF port		Use ECMWF Ioki port
IO	Almost ready	Coupling not ready	Coupling not ready	Use IO server
Post-processing	No	No	No	Not started yet

Conclusions and outlook

- GPU adaptation progresses steadily
 - Targeting new architectures
 - More parts of the code refactored and ported
 - Steps towards consolidation

- Progress on AROME physics
- Continue on AMD
- Test and integrate ongoing developments: ECRAD, fiat/MPL, ectrans