C2SM
Center for Climate
Systems Modeling

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

# OpenACC directives in COSMO

Xavier Lapillonne

Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

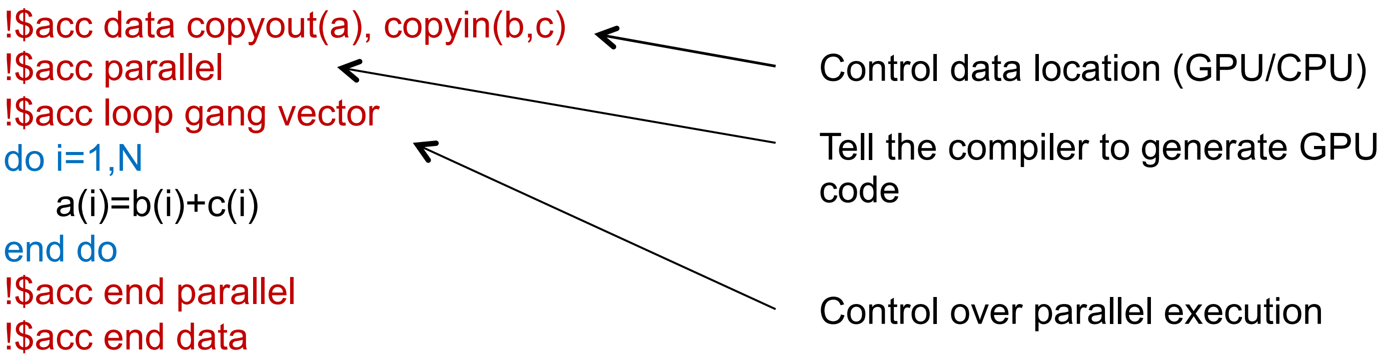
Eidgenössisches Departement des Innern EDI
**Bundesamt für Meteorologie und Klimatologie MeteoSchweiz**

HP2C

CSCS
Swiss National Supercomputing Centre

# OpenACC compiler directives

OpenACC : Open standard, supported by 3 compiler vendors PGI, Cray, Caps



```
!$acc data copyout(a), copyin(b,c)
!$acc parallel
!$acc loop gang vector
do i=1,N
    a(i)=b(i)+c(i)
end do
!$acc end parallel
!$acc end data
```

Control data location (GPU/CPU)

Tell the compiler to generate GPU code

Control over parallel execution

## Why using directives ?

| | GPU-Specific language (Ex: CUDA) | Compiler directives |
|---|---|---|
| Implementation, porting effort | Re-write | Incremental adaptation of existing code |
| Control over parallel execution | Yes | Yes |
| Control over memory hierarchy | Yes | No (automatic) |
| Software management | Often separate GPU and CPU code | Single source possible |
| Code maintenance/development | Requires understanding of GPU computing and new language | Retain original language, requires basic concept of GPU computing |
| Target Architecture | CUDA: Nvidia, OpenCL: multiple | Multiple |

# Porting strategy

- OpenACC used for Physics, data assimilation, other parts of the time loop
- Parallelization: horizontal direction, 1 thread per vertical column
- Most loop structures unchanged, one only adds directives : port large part of the code
- For time critical parts (mostly in the physics) additional optimizations[1]:
  - Loop restructuring
  - Removal of local automatic array
  - Scalar replacement
  - …

  Note : some of the GPU-optimized routines are slower on the CPU. For the moment we keep a separate GPU and CPU source for these parts.

# Directives in COSMO[2]

- 3000 !$acc directives
- 60 000 lines of code ported to GPU

[1] Lapillonne and Fuhrer, Parallel Processing Letters, under review

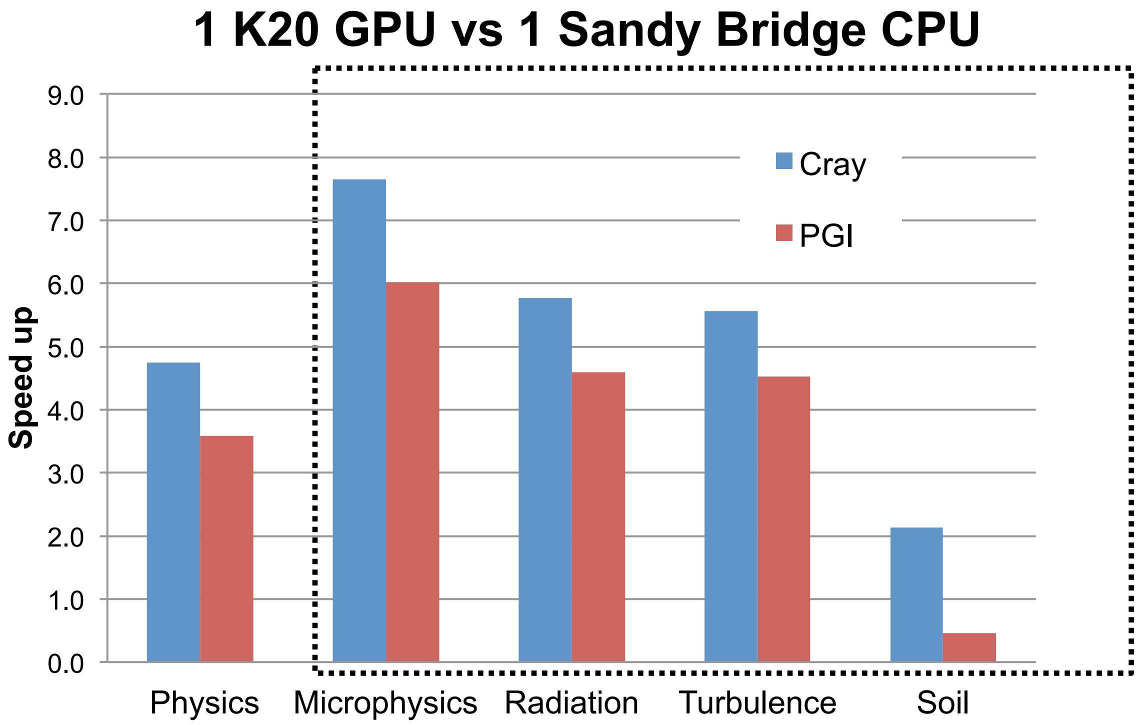
[2] thank to T. Diamanti, D. Leutwyler, C. Padrin , A. Roches, S. Schaffner.

# Current status of Physics

| Scheme | C-2 | C-7 | Status | Remark |
|---|---|---|---|---|
| microphysics | | | | |
| - hydci_pp (ice scheme) | | x | done | |
| - hydci_pp_gr (graupel) | x | | ready (standalone) | not ported yet in OPCODE |
| sub grid scale oro. (sso) | x | x | done | |
| radiation | x | x | done | |
| turbulence | x | x | done | only operational options |
| soil model | | | | |
| - terra_multlay | x | x | done | only operational options |
| - terra1 | | | - | |
| - terra2 | | | - | |
| - seaice | | | - | |
| - flake_interface | | | - | |
| convection | | | | |
| - conv_tiedtke | | x | work in progress | |
| - organize_conv_kainfri | | | | |
| - conv_shallow | x | | done | |
| Boundary exchange | x | x | done | |

# Performance results using directives : Physics

- Test domain 263x92x60, 1h simulation.
- Socket to Socket comparison : 8 core Intel Sandy Bridge CPU vs K20 GPU
- Using 2 different compilers PGI and Cray



- 90% of physics run time on original CPU code: microphysics, radiation and turbulence -> Higher optimization effort
- Overall speed up factor between 3.5x and 4.7x (depend also on input data)

# Summary / Discussion

- Large parts of COSMO were successfully ported using OpenACC
- For the physics, additional optimizations were possible and we obtain good performance on GPU : 3.5x speed up on a node to node comparison
- Compute intensive parts perform better -> new possibilities for the physics
- The physics is now using different source files for CPU and GPU. We still need to determine how much could be shared

# Next steps

- Implement missing parametrizations in OPCODE (gsp_gr and tiedke_conv)
- Explore how to have as much shared code between CPU and GPU code
- Merge OpenACC work in COSMO official code

# Thank you

# Performance results using directives : Physics

- Test domain 128x128x60, 1h simulation.
- Socket to Socket comparison : 8 core Intel Sandy Bridge CPU vs K20 GPU

| note : res. 128x128x60 time 1h run (=180 steps) | | | | | | |
|---|---|---|---|---|---|---|
| | Physics | Microphysics | Radiation | Turbulence | Soil | copy-block |
| Sandy bridge | 43.8 | 9.940 | 12.300 | 17.020 | 1.73 | |
| Sandy bridge lblock_phys=T nproma=16 | 56.6 | 8.700 | 14.140 | 20.110 | 1.9 | 7.6 |
| K20x (CCE) | 9.23 | 1.300 | 2.130 | 3.060 | 0.81 | 8.47 |
| K20 (PGI) | 12.2 | 1.650 | 2.680 | 3.760 | 3.76 | 11.34 |
| Speed up CCE | 4.745 | 7.646 | 5.775 | 5.562 | 2.136 | |
| Speed up PGI | 3.59 | 6.02 | 4.59 | 4.53 | 0.46 | |