C2SM Center for Climate Systems Modeling

ETH Eidgenössische Technische Hochschule Zürich Swiss Federal Institute of Technology Zurich

News from POMPA and HP2C

COSMO User Seminar 2013

Xavier Lapillonne and the HP2C team

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

Goals of the HP2C project

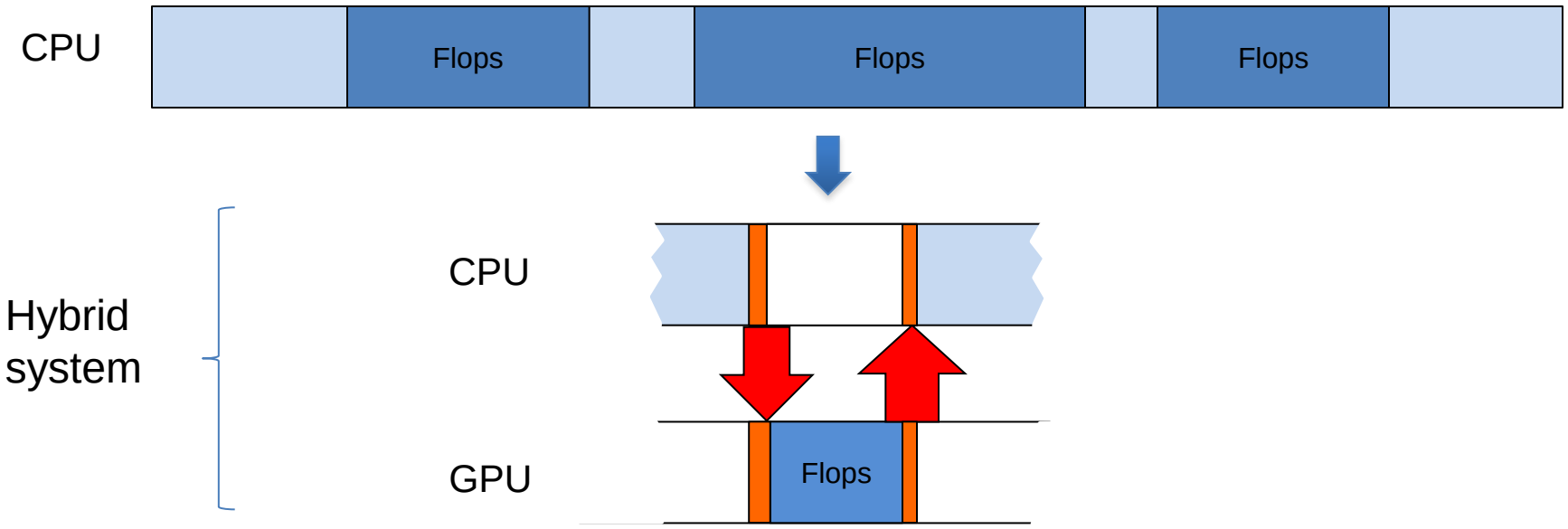
- How to write code that:
  - runs efficiently on different architectures
  - allow domain scientist to easily bring new developments
  - will still be a good fit in the future

- Target architectures :
  - Multicore CPU and GPUs

- Approaches :
  - New Library STELLA : dynamical core. Improved performance and CPU, enable GPU
  - Compiler directives (OpenACC): rest of the code, e.g. physics, assimilation … . Allow to run on GPU, no improvement on CPU.

Using GPUs : the accelerator approach

- CPU and GPU have different memories



- Most intensive parts are ported to GPU, data is copied back and forth between the GPU and the CPU between each accelerated part.

What does this mean for atmospheric model application ?

- Low FLOP count per load/store (stencil computation)
- Example with COSMO-2 (operational configuration at MeteoSwiss) :

*

Part	Time/Δt
Dynamics	**172 ms**
Physics	**36 ms**
Total	**253 ms**

vs

§

Transfer of ten prognostic variables

118 ms

CPU-GPU data transfer time is large with respect to computation time:

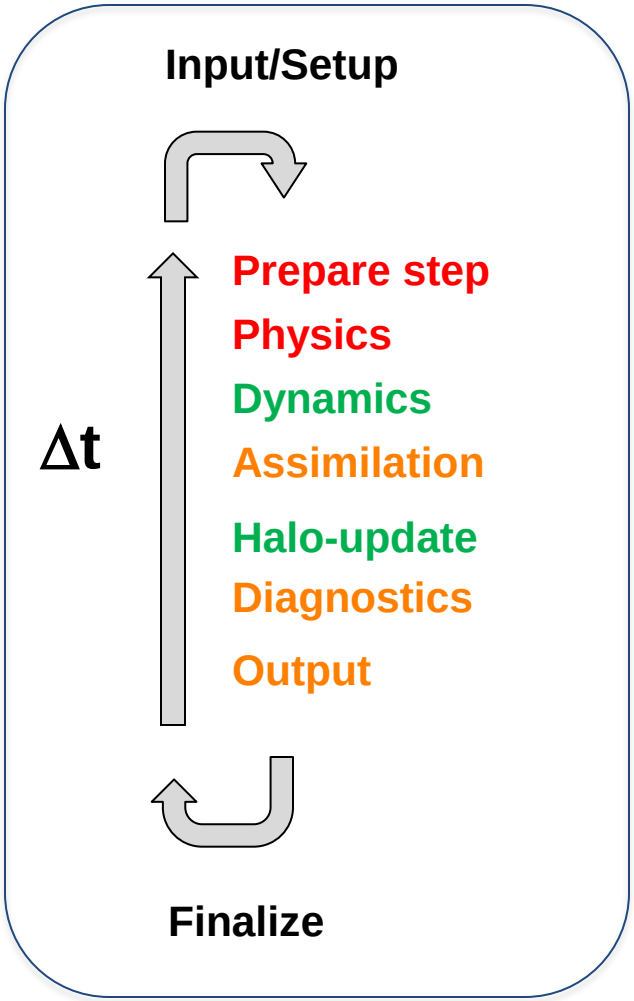
Accelerator approach might not be optimal

* CPU measurements: Cray XT6, Magny-Cours, 45 nodes, COSMO-2

§ GPU measurements: PCIe 8x, HP SL390, 2 GB/s one way, 8 sub-domains

COSMO on GPUs

- **Avoid CPU-GPU copies by executing all the time loop computation on GPU**



→ keep on CPU / copy to GPU

→ **OpenACC directives**
→ **OpenACC directives**
→ **STELLA**
→ **OpenACC directive (part on CPU)**
→ **GPU-GPU communication library (GCL)**
→ **mixed CPU and GPU (OpenACC)**
→ **mixed CPU and GPU (OpenACC)**

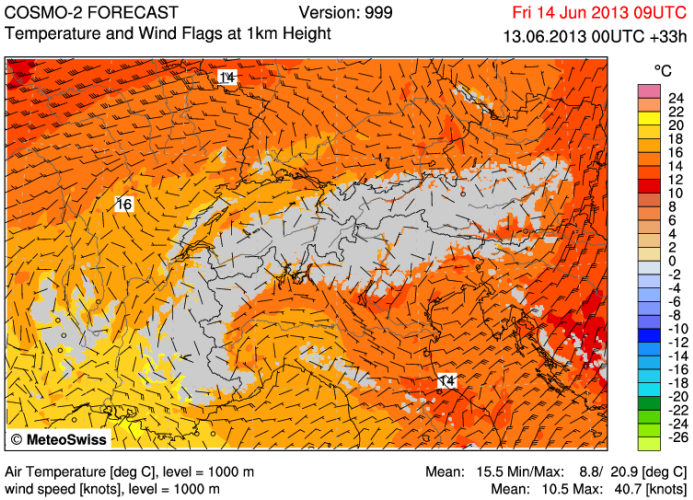
Current status of the physics porting

Scheme	C-2	C-7	Status
microphysics			
- hydci_pp (ice scheme)		x	done
- hydci_pp_gr (graupel)	x		done
sub grid scale oro. (sso)	x	x	done
radiation	x	x	done
turbulence	x	x	done
soil model			
- terra_multlay	x	x	done
- terra1			-
- terra2			-
- seaice			-
- flake_interface			-
convection			
- conv_tiedtke		x	ready
- organize_conv_kainfri			
- conv_shallow	x		done
Boundary exchange	x	x	done

thank to D. Leutwyler, C. Padrin , A. Roches, S. Schaffner.

It works !

- All component integrated together : Communication, Dynamics, Physics
- Run “operational” suite about one time a day on the OPCODE prototype system since June 2013



33h simulation on prototype OPCODE system (8 GPUs).

144 CPUs with 12 cores each (1728 cores)

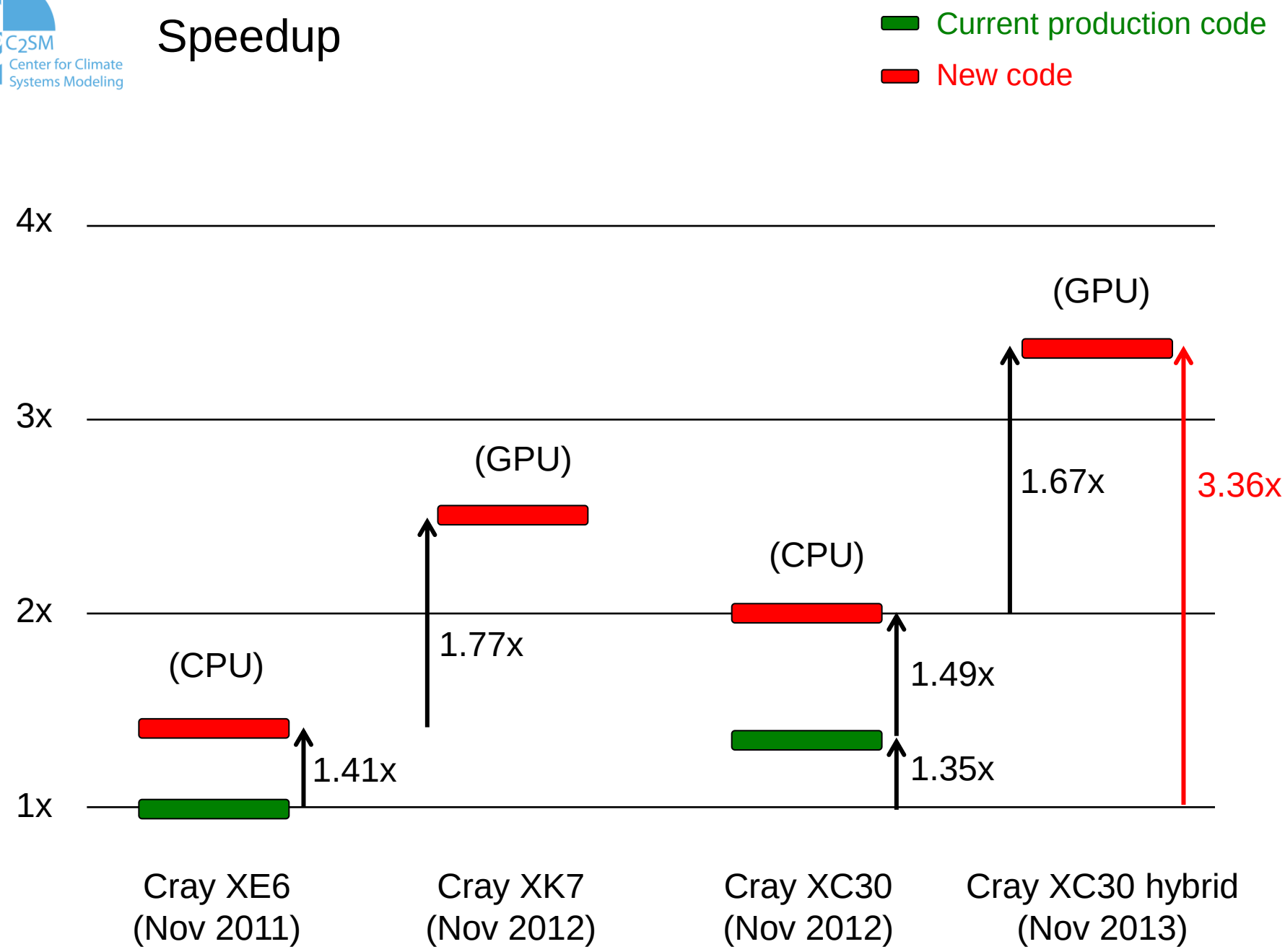
Multi GPU based hardware
1 node with 8 GPUs

1 cabinet Cray XE5

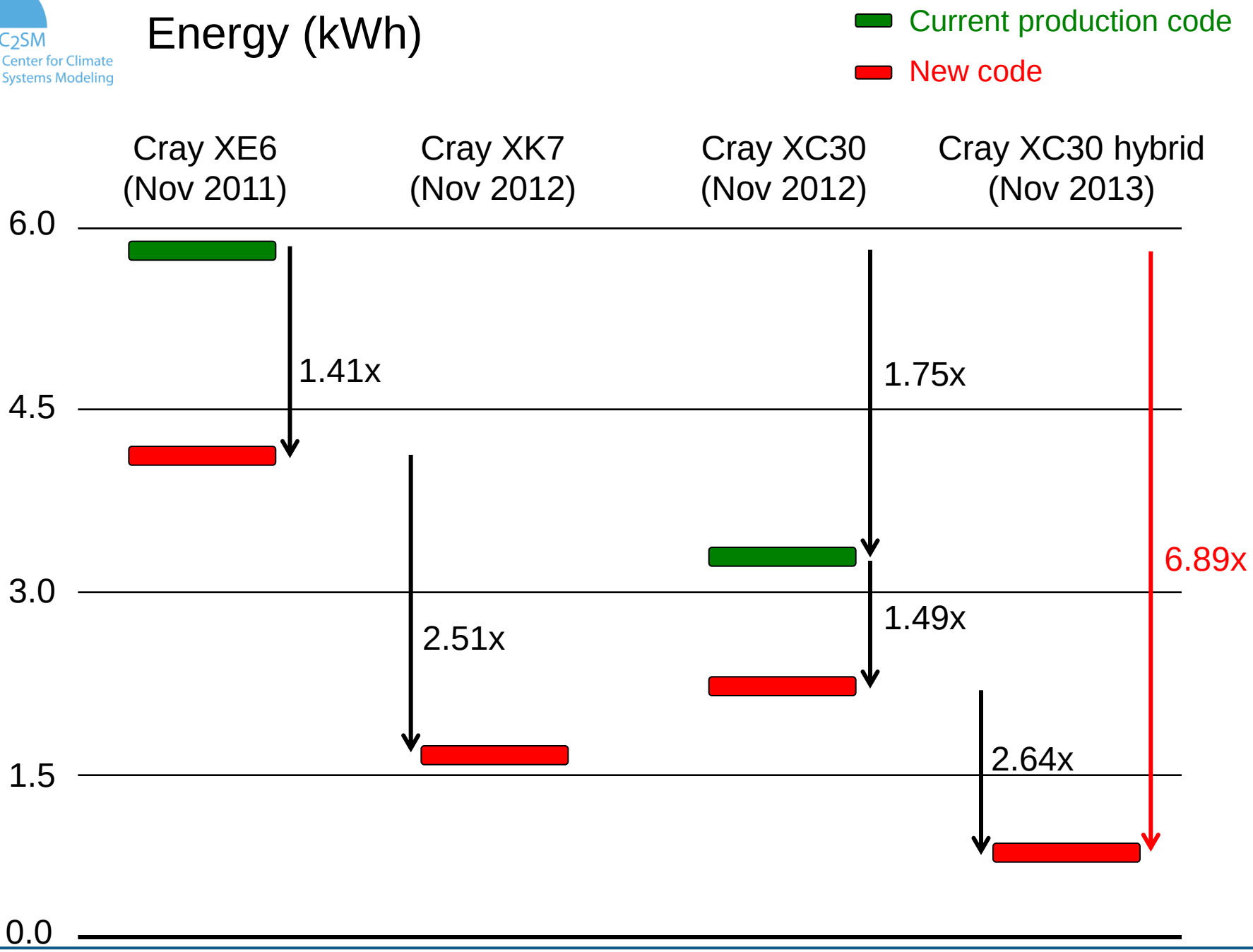
Comparing performance

- Depends on:
  - Hardware and use case
  - Considered architecture
- How to compare different systems ?
- Speed up : socket to socket, same problem size
- Energy cost :
  - Fill a cabinet with a COSMO-2 ensemble
  - Take as much members as possible to get optimal performance
  - Compare time and energy cost per member

Speedup


Current production code
New code
4x
3x
2x
1x
(CPU)
1.41x
(GPU)
1.77x
(CPU)
1.49x
1.35x
(GPU)
1.67x
3.36x
Cray XE6
(Nov 2011)
Cray XK7
(Nov 2012)
Cray XC30
(Nov 2012)
Cray XC30 hybrid
(Nov 2013)

Energy (kWh)


Current production code
New code
Cray XE6 (Nov 2011)
Cray XK7 (Nov 2012)
Cray XC30 (Nov 2012)
Cray XC30 hybrid (Nov 2013)
6.0
4.5
3.0
1.5
0.0
1.41x
2.51x
1.75x
1.49x
2.64x
6.89x

News and Next steps

- HP2C project ended in June 2013

- Revised priority project POMPA : Accepted by COSMO consortium
  -> bring back the development in COSMO trunk (i.e. COSMO 5.X)

- Planned developments this year:
  - Merge all changes with COSMO 5.0
  - Adapt physics for the ICON-COSMO shared library
  - Bring back work into the trunk
  - Add all missing parts required for climate and weather applications
  - Find best solution for efficient and maintainable code which performs well on CPU and GPU using OpenACC directives

C2SM
Center for Climate
Systems Modeling

Thank you

Directives / Compiler choices for OPCODE

```
!$acc data create(a,b,c)
!$acc update device(b,c)
!$acc parallel
  !$acc loop gang vector
  do i=1,N
   a(i)=b(i)+c(i)
  end do
!$acc end parallel
!$acc update host(a)
!$acc end data
```

Device memory management

Generates GPU kernel.
Map parallelism to architecture

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

- Solution retained for OPCODE (for physics, assimilation)
- Running and tested: Cray, PGI
- CAPS: not investigated yet

Arrays allocated on the GPU in the Fortran code can be passed to the C++ dycore using the host_data directive. No GPU-CPU transfer required.

➡ Being able to test code with different compilers is essential