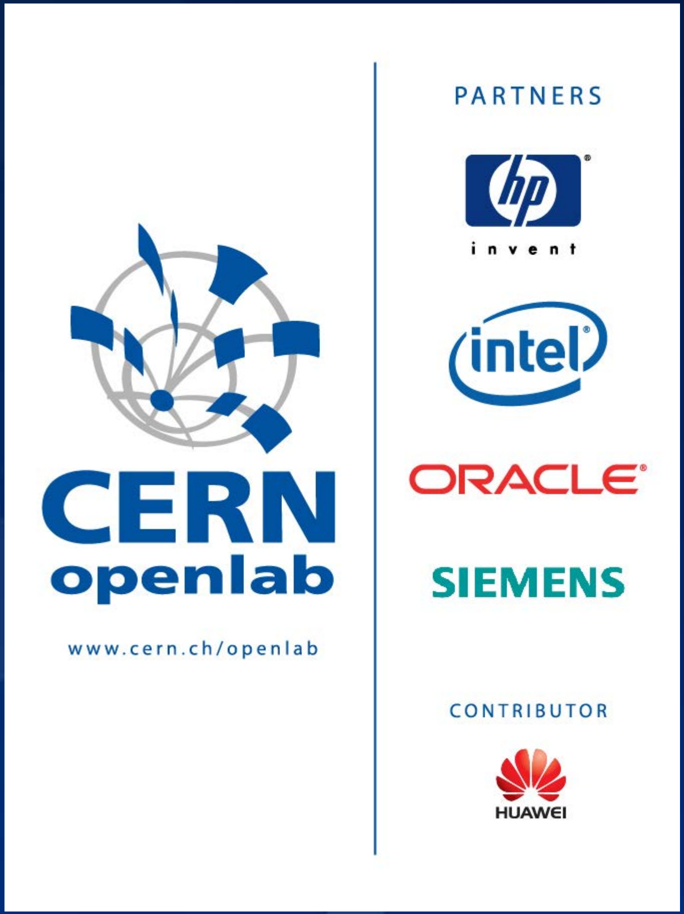
Practical results of the Intel MIC / Xeon Phi project at CERN openlab

2nd Annual Concurrency Forum Meeting
February 5th 2013

Andrzej Nowak, CERN openlab
Based on the work of Sverre Jarp, Alfio Lazzaro, Julien Leduc, Andrzej Nowak, Klaus-Dieter Oertel, Liviu Valsan

CERN openlab

- **CERN openlab is a framework for evaluating and integrating cutting-edge IT technologies or services in partnership with industry**
- **The Platform Competence Center (PCC) has worked closely with Intel for the past decade and focuses on:**
  - many-core scalability
  - performance tuning and optimization
  - benchmarking and thermal optimization
  - teaching



DISCLAIMER

Our tests are based on pre-production MIC / Xeon Phi hardware and software

Brief history

Early access

- Work since MIC alpha (under RS-NDA)
- ISA reviews in 2008

Results

- 3 benchmarks ported from Xeon and delivering results: ROOT, Geant4, ALICE HLT trackfitter

Expertise

- Understood and compared with Xeon
- **Post-launch dissemination**

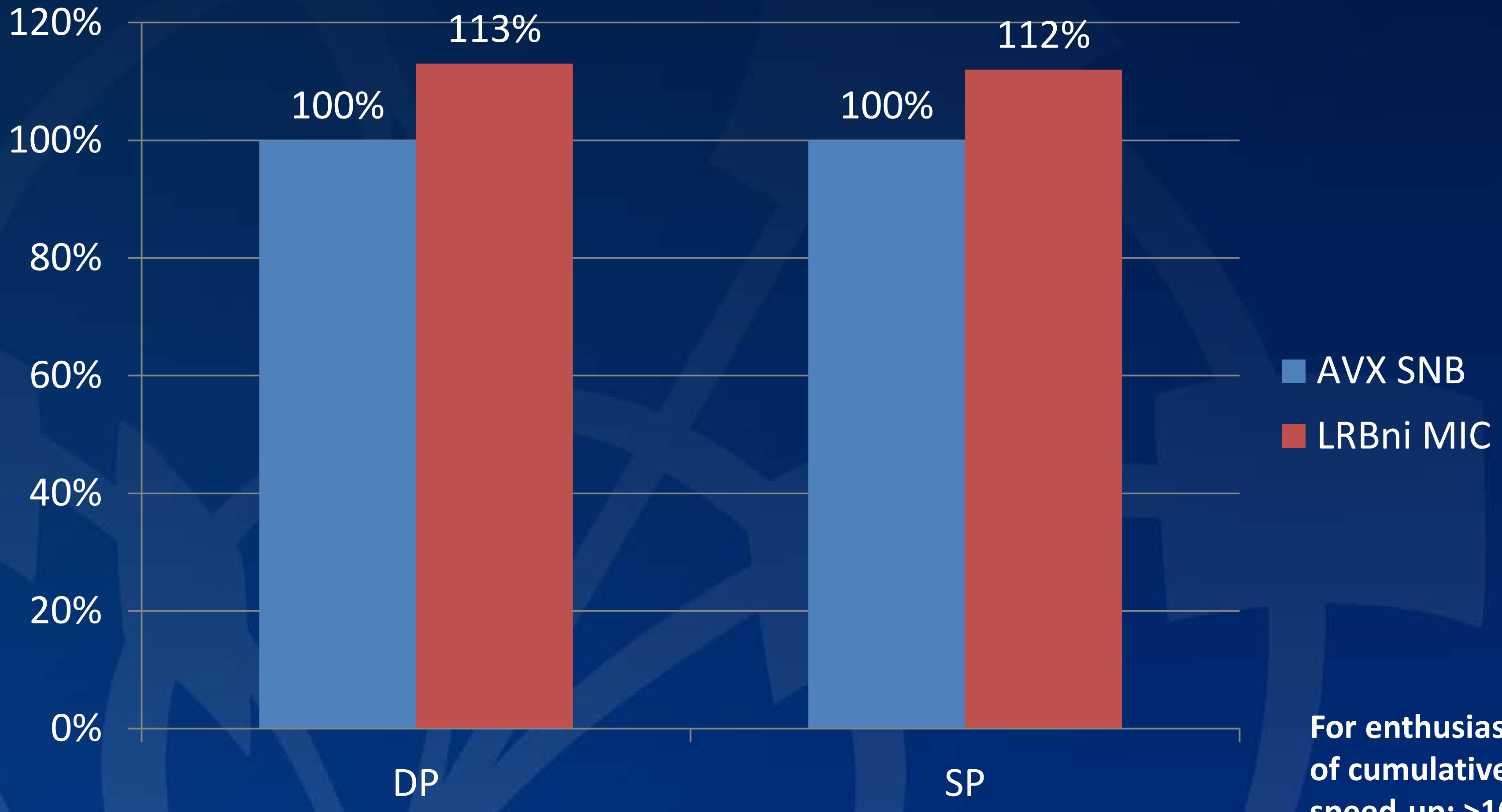
Ported applications

- **ALICE/CBM track fitter prototype**
  - Threaded
  - Explicitly vectorized with a VC-like technology
- **MLFit**
  - Threaded (pthreads, MPI, OpenMP, TBB)
  - Vectorized (Cilk+)
- **Early multi-threaded Geant4 prototype**
  - Threaded (pthreads)
  - No vectorization
- **Test hardware**
  - Pre-production Knights Corner – 61 cores @ 1.1 GHz
  - Sandy Bridge-EP – 16 cores @ 2.7 GHz, Turbo on
  - Frequency unscaled results reported (1:1 comparison)

Porting – how much work?

	LOC	1st port time	New ports	Tuning
TF	< 1’000	days	N/A	2 weeks
MLFit	3’000	< 1 day	< 1 day	weeks
MTG	2’000’000	1 month	< 1 day	< 1 week

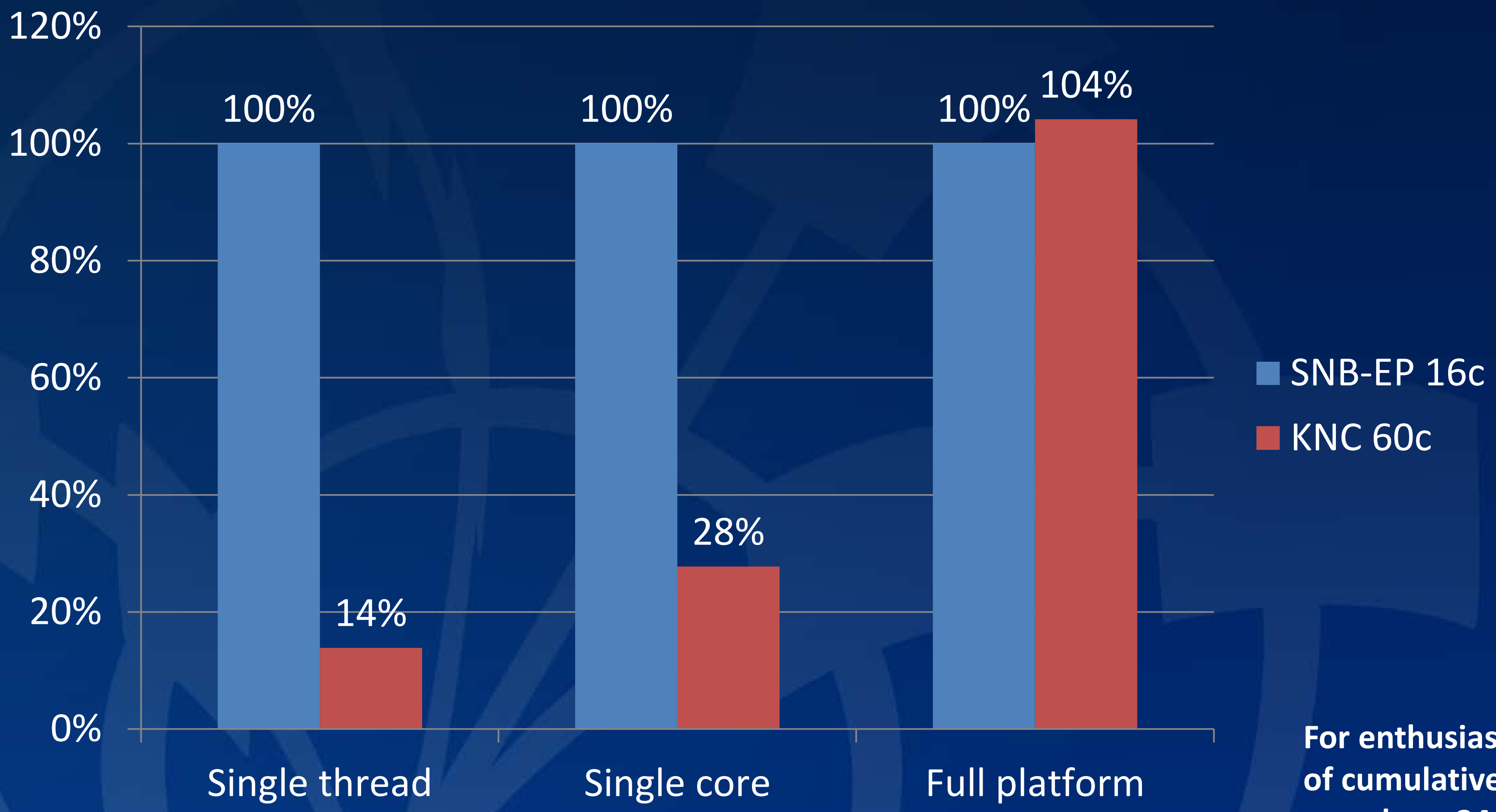
Track fitter throughput
(higher is better)



For enthusiasts of cumulative speed-up: >100x

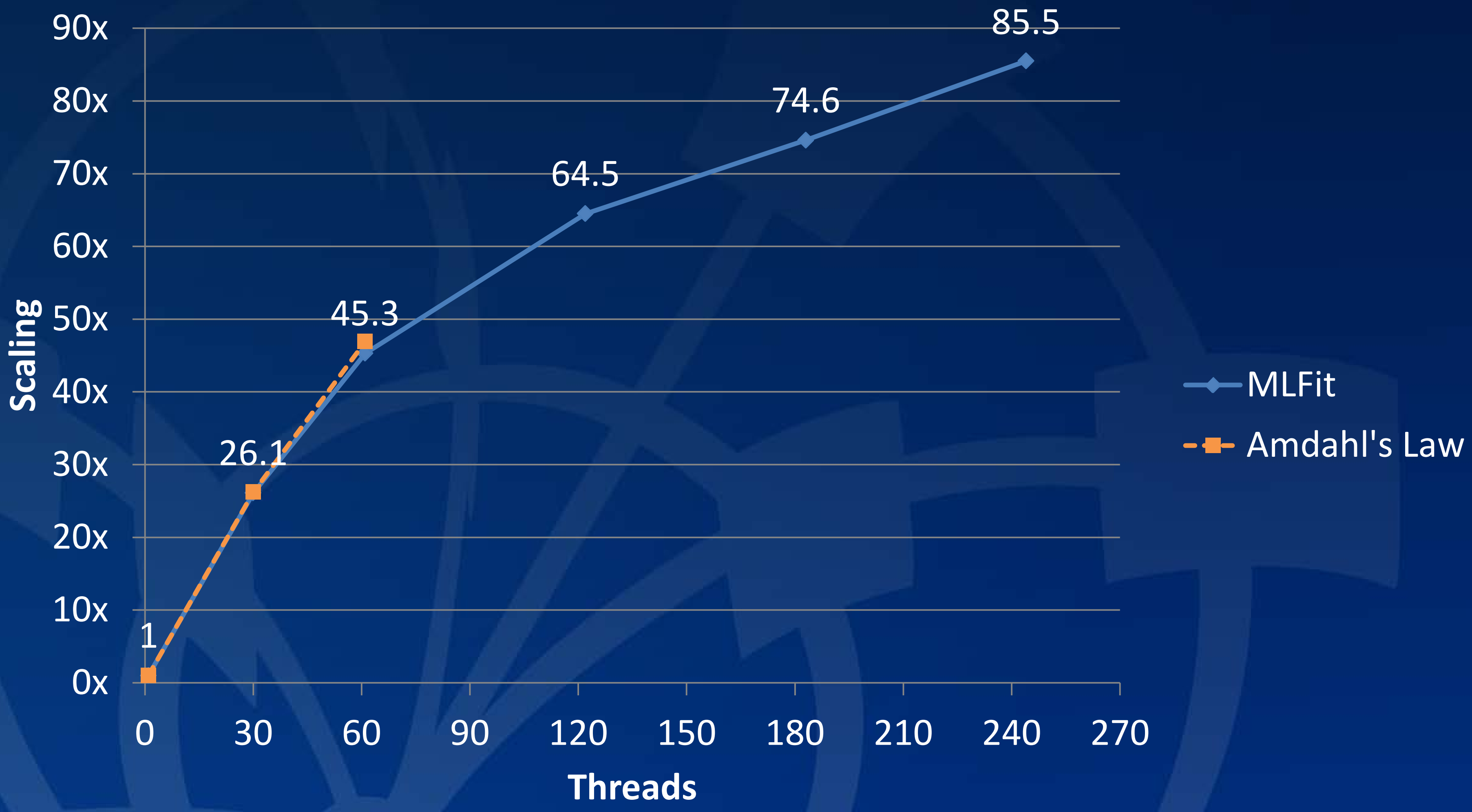
MLFit performance

OpenMP, no block splitting, higher is better

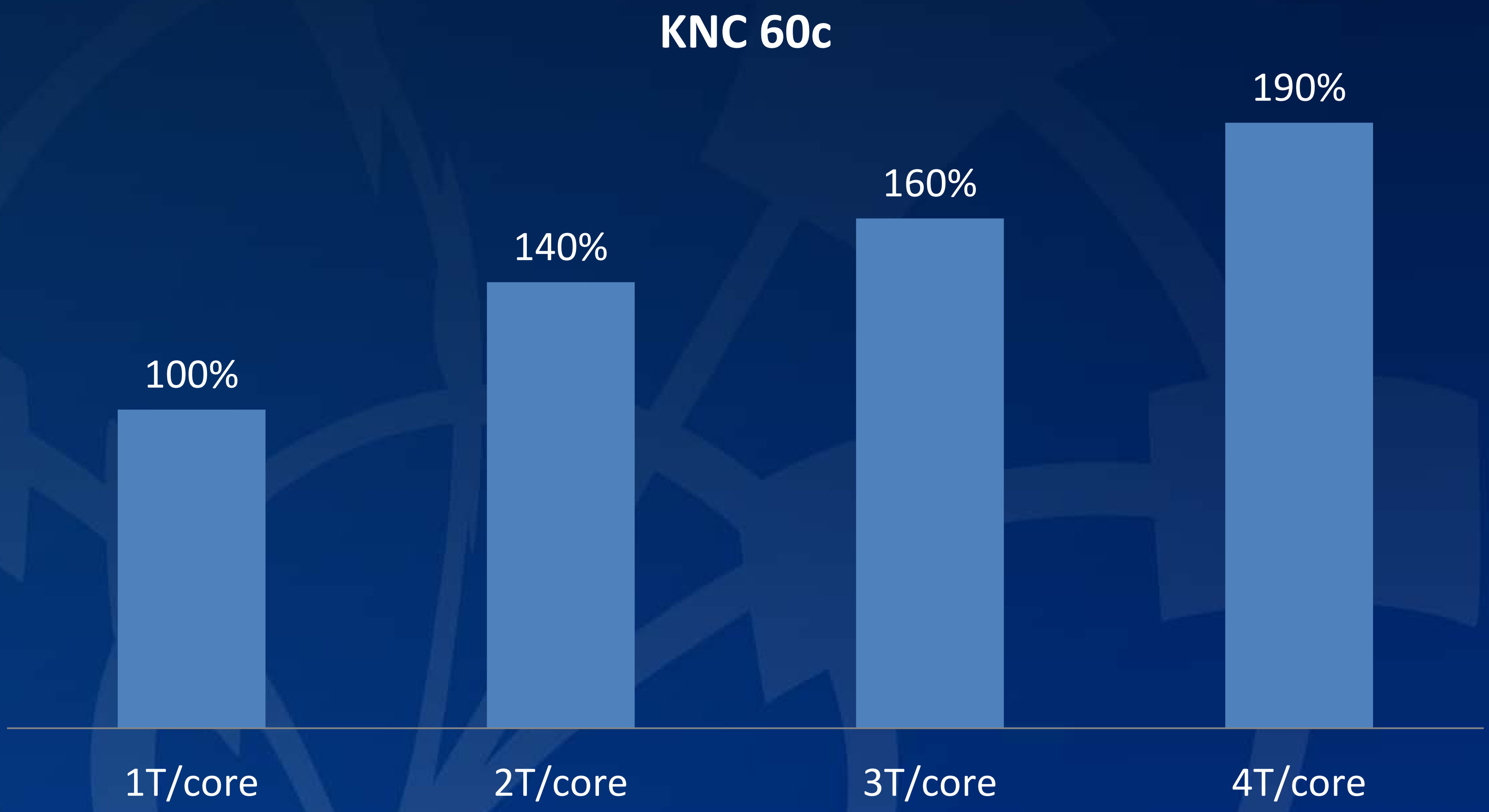


For enthusiasts of cumulative speed-up: 34x

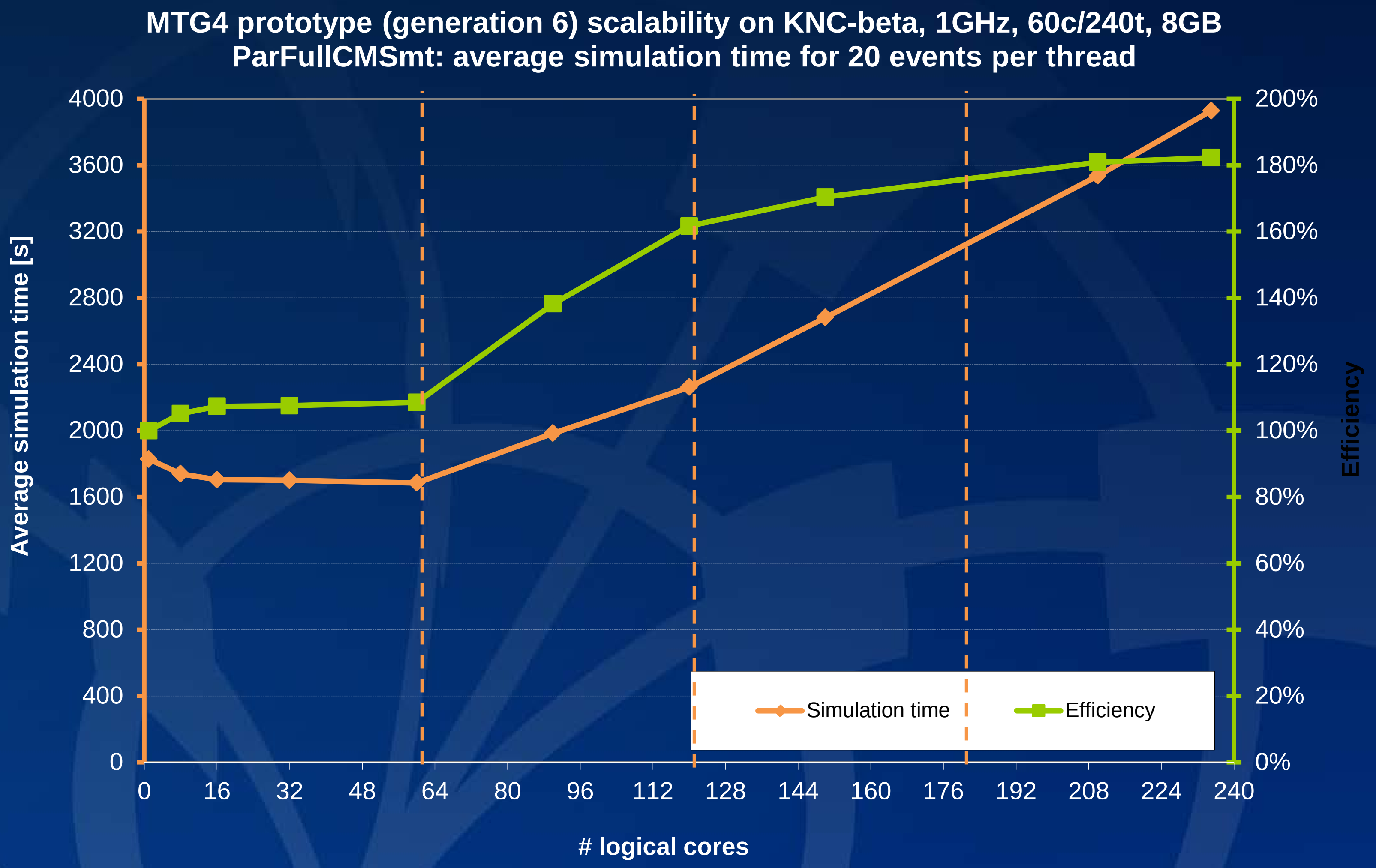
MLFit scaling
(OpenMP)



MLFit threading performance

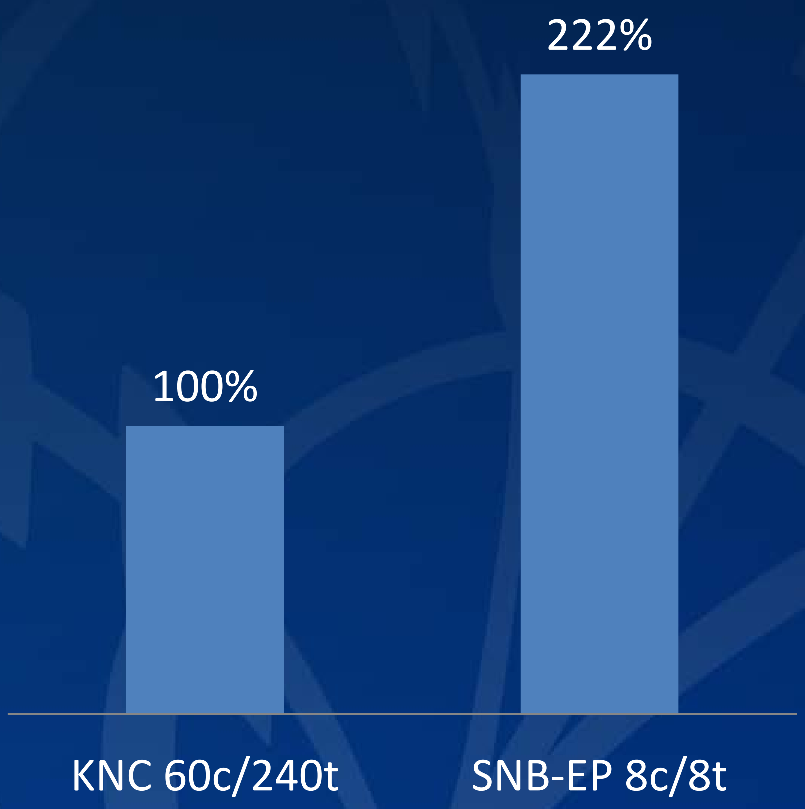


MTG4 scalability

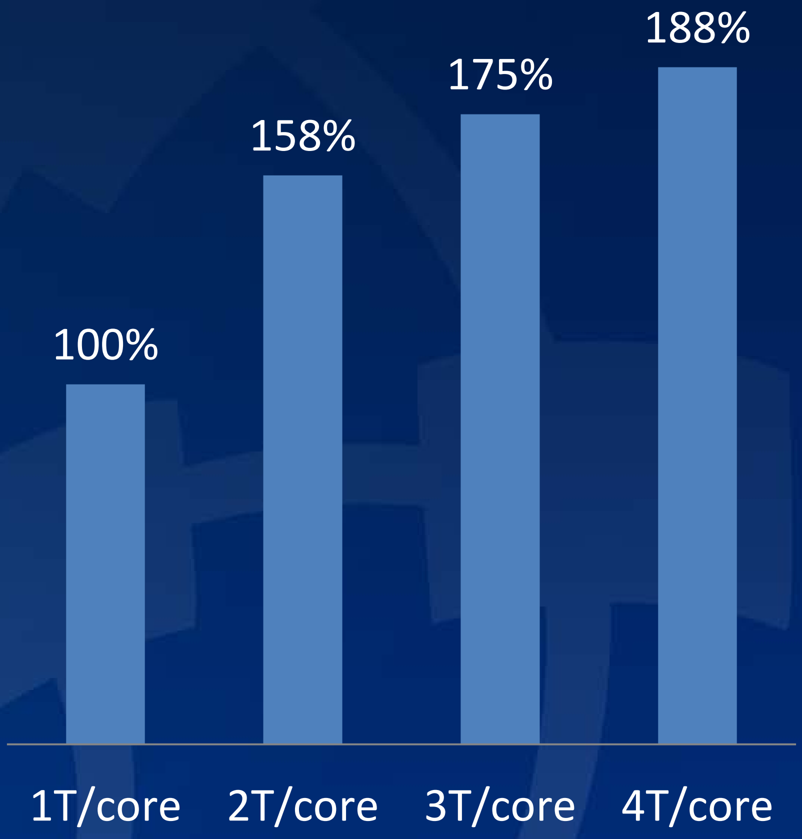


MTG4 performance (higher is better, no vectorization)

Full platform performance



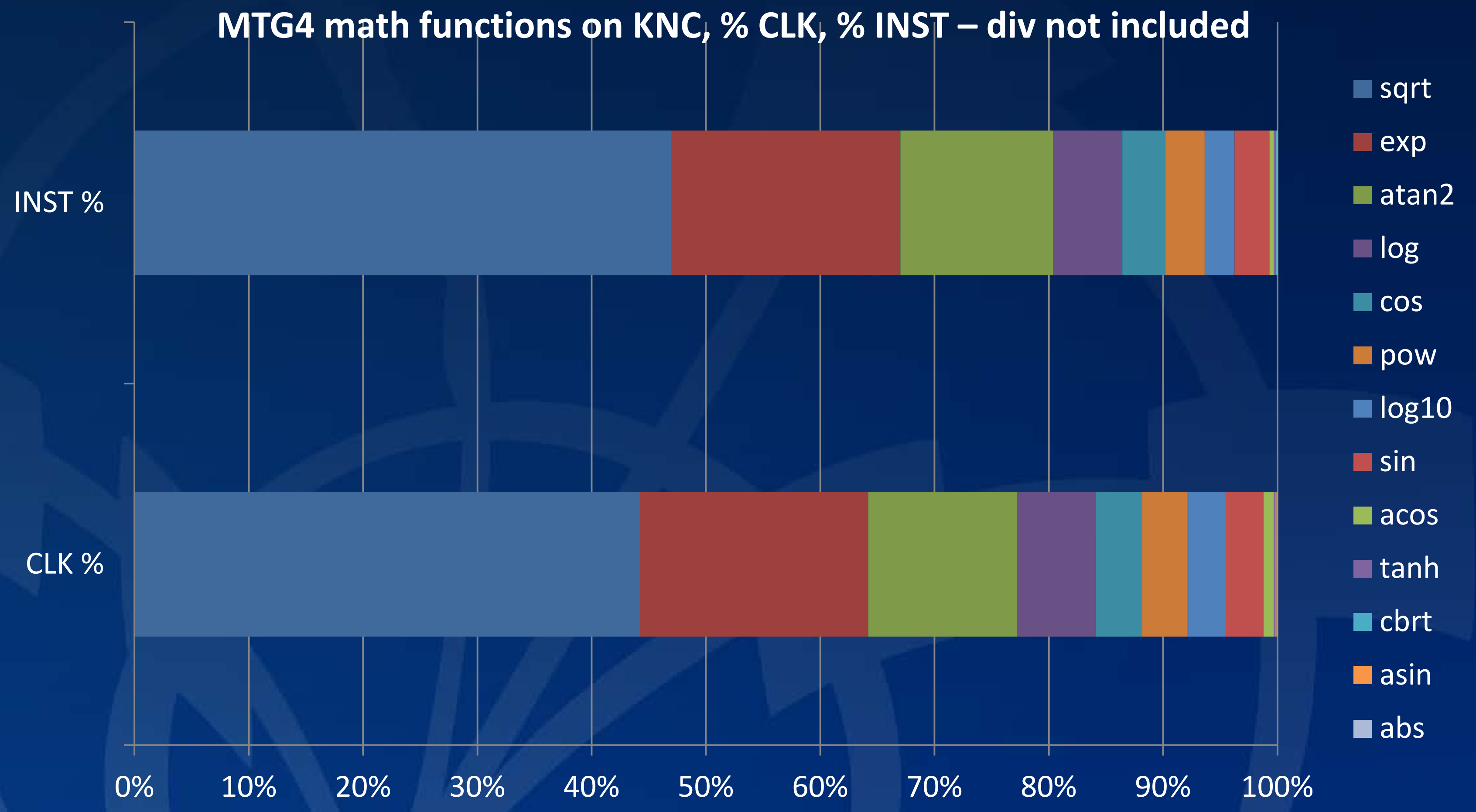
HW threading throughput



MTG4 – example profile

Function / Call Stack	CLK %	INST %
sqrt	14.35%	22.16%
exp	6.47%	9.47%
atan2	4.22%	6.31%
CLHEP::RanluxEngine::flat	3.24%	5.60%
G4ElasticHadrNucleusHE::HadronNucleusQ2_2	3.01%	2.41%
G4PhysicsVector::Value	2.76%	0.95%
log	2.22%	2.85%
G4VoxelNavigation::LevelLocate	2.05%	0.66%
G4VoxelNavigation::ComputeStep	1.64%	1.10%
G4ClassicalRK4::DumbStepper	1.59%	2.96%
G4SteppingManager::DefinePhysicalStepLength	1.54%	1.39%
G4Navigator::ComputeStep	1.40%	1.01%

MTG4 math


MTG4 math functions on KNC, % CLK, % INST – div not included
INST %
CLK %
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
sqrt
exp
atan2
log
cos
pow
log10
sin
acos
tanh
cbrt
asin
abs

Performance - summary

- Optimized applications surpass dual-socket Xeon performance
- Non-optimized performance reaches approximately a single Xeon socket
- Math function usage and performance are key vis a vis Xeon
- Compiler maturity still an open question

Multiple dimensions of performance

Dimension	Software
Nodes	MPI
Sockets	Threading and NUMA control
Cores	Threading
ILP	Efficient compilers and code
Pipelining	Efficient compilers
Vectors	Various options

MIC software - scenarios

Native mode

workload runs entirely on a MIC system (networked via PCIe)

MIC Architecture Progra
Compilers and Runtimes
Intel Xeon processor
Intel MIC architecture co-processor
Common with Inte
Languages
C, C++, Fortran co
Intel developer too and libraries
Coding and optimization techn
Ecosystem suppo
ates Need for Dual Programming Archite

Offload

MIC as an accelerator where host gets weak

Balanced

MIC and host work together

Cluster

application distributed across multiple MIC cards (possibly including host)

MIC evolution – implications for HEP

- Small core evolution
- Hybrid mixes (Xeon + MIC)
- ISA convergence
- Will I/O latency or BW be a constraint?
- Other applications:
  - HPC-like code (e.g. QCD, CFD)
  - Triggering
  - High data throughput (ICE-DIP)

MIC evolution: ICE-DIP

- FP7 project looking for (amongst other things) efficient methods of accelerator/co-processor use
- Focus on data taking past LS1
- Of particular interest
  - Getting data into the platform
  - Getting data into the accelerator/co-processor
  - Efficient processing
  - Efficient distribution of results

Are you interested?

- We are always looking for interesting prototypes from the physics community
- We recommend:
  - Data oriented design as opposed to only control flow oriented (OOO) – no vectorization, no fun
  - Porting to the Intel compiler on Xeon first
  - Thinking in multiple dimensions of performance – vectorize, thread, limit memory usage
  - Checking precision and math usage

THANK YOU

Q & A

CERN
openlab

Questions? Andrzej.Nowak@cern.ch