Elmer on Intel® Xeon-Phi

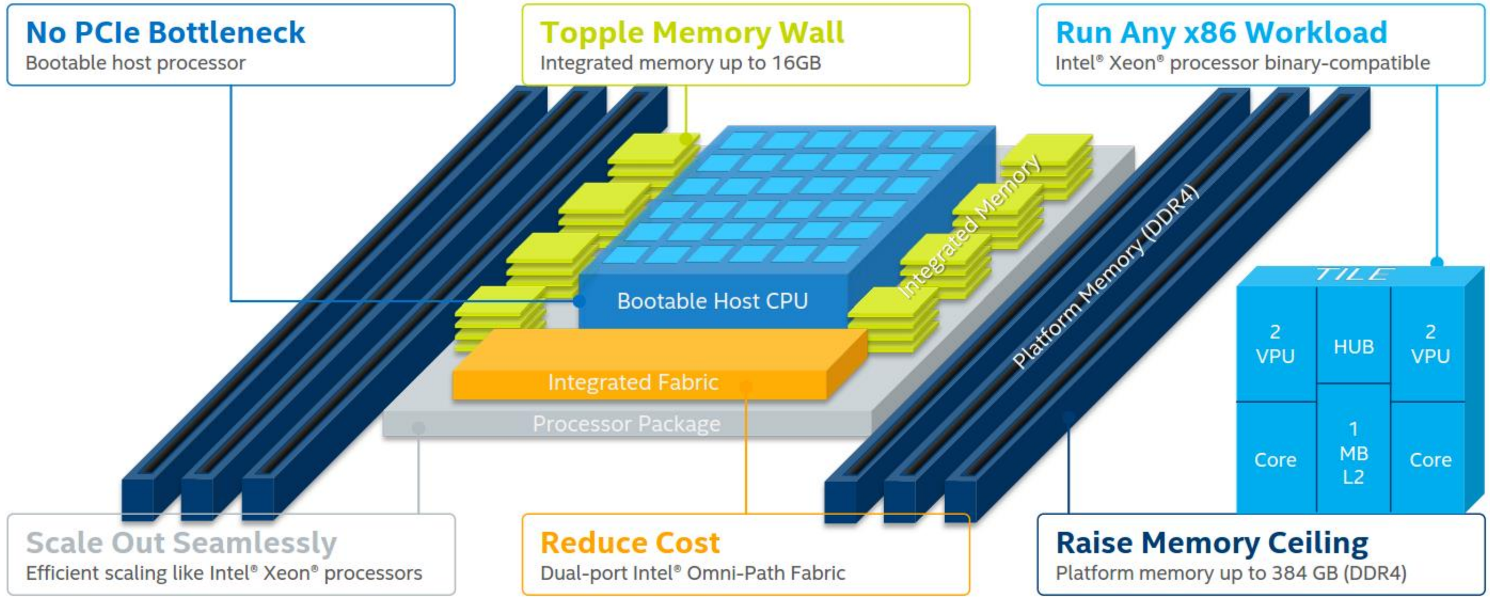
Mikko Byckling (Intel)
Juhani Kataja (CSC)

ElmerTeam
CSC – IT Center for Science

Porting Elmer to MIC

- Porting work started already Q2/12
- Focus to build ElmerSolver on a MIC
  - MIC = Many Integrated Cores
- Cooperation with Mikko Byckling (Intel) within *Intel Parallel Computing Center* (IPCC)

Porting Elmer to MIC



[1]Reduced cost based on Intel internal estimate comparing cost of discrete networking components with the integrated fabric solution

Porting Elmer to MIC

- Internally OpenMP threading supported by
  - Solver API routines related to element assembly
  - Element assembly loop of some solvers already implemented
  - Time integration routines
  - Sparse matrix vector products
- Library support for OpenMP exists in
  - External BLAS routines
  - External LAPACK routines
  - Direct solvers such as Cholmod, SPQR and Pardiso

Porting Elmer to MIC

- Perform disruptive changes if necessary
  - Maintain backwards compatibility
  - Build backwards compatible interfaces to new methods if necessary
- Optimization order
  - Vectorization
  - Threading
- Tools currently in use
  - Intel Vtune (to find hotspots and non-vectorizable parts of the code on the time critical path)
  - Intel Inspector XE (to find threading bugs)
- Targeting both Xeon and Xeon Phi

Porting Elmer to MIC

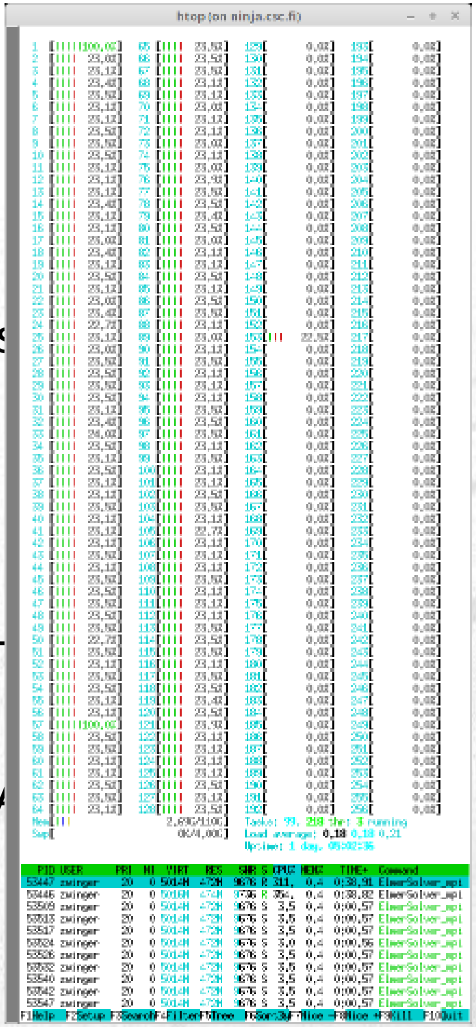
- Modern Fortran code with a modular structure
  - Initial focus on Finite element assembly
  - Improve the vectorization properties by changing the key data structures
  - Add OpenMP multithreading
- All ~50 solvers in Elmer need to be modified

= AHLOW!

```
!$omp parallel do private(Element,n,nd)
    DO t=1,active
       Element => GetActiveElement(t)
       n  = GetElementNOFNodes(Element)
       nd = GetElementNOFDOFs(Element)
       CALL LocalMatrix(Element, STIFF, FORCE, n,
nd)
       CALL DefaultUpdateEquations(STIFF,FORCE,&
                                   UElement=Element)
    END DO
!$omp end parallel do
```

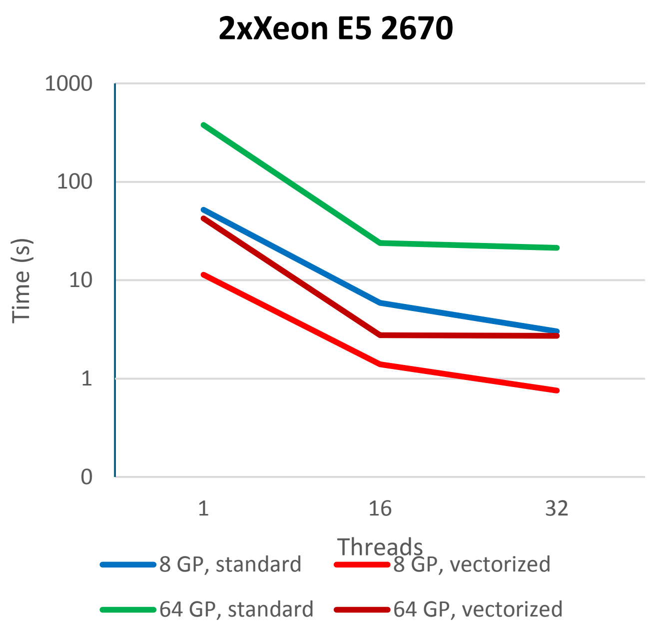
Porting Elmer to MIC

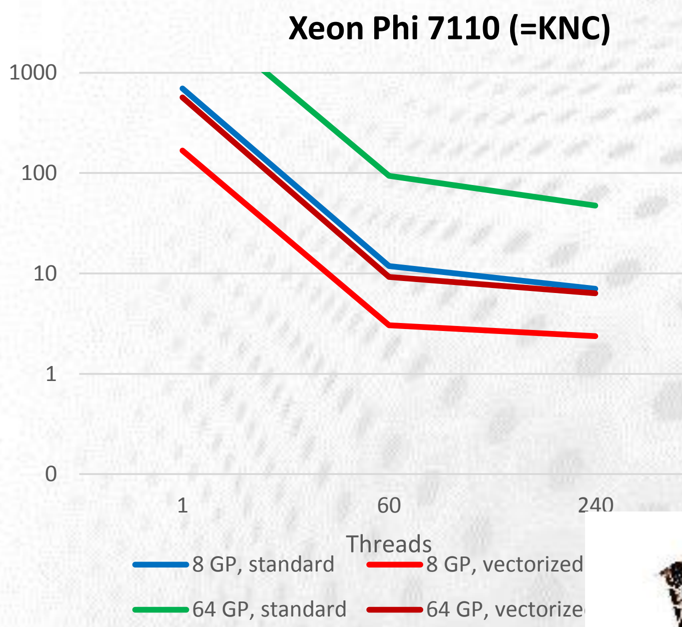
- Poisson (elliptic problem) solver
  - Large vectors (FEM Gauss points
  - Mesh colouring (avoid race conditions)
  - Tested on Xeon Phi developer N platform
    - Intel® Xeon Phi (TM) CPU 721 1.30GHz
    - 64 cores (256 HT 4x)
    - 96GB DDR4,16GB MCDRA
    - KNL (KNights Landing)



Porting Elmer to MIC

- Poisson model problem, 1M Hexahedral elements

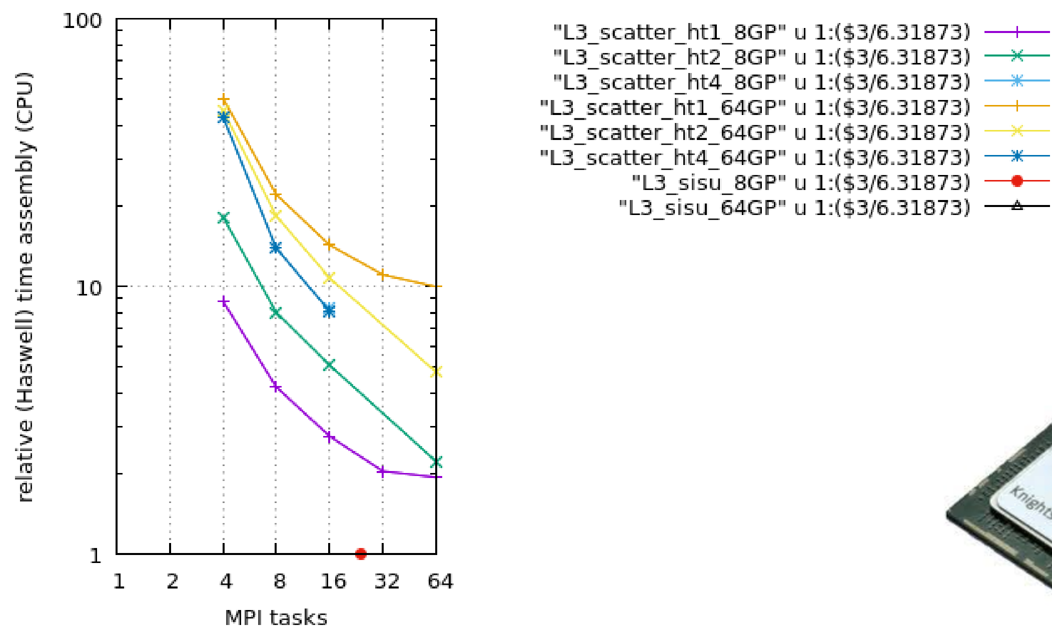




Porting Elmer to MIC

- Poisson model problem, 1M Hexahedral elements

2xXeon E5 2670 **Xeon Phi 7210 (=KNL)**



Porting Elmer to MIC

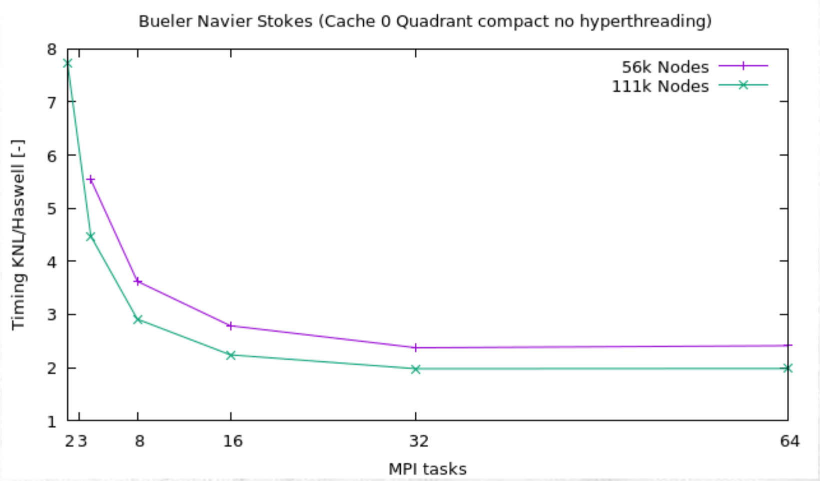
Activity supported by

eSTICC

norden

NordForsk

- Production solver used in Elmer/Ice
- Synthetic ice-sheet goemetry (Bueler-profile) with (Navier-)Stokes solver with non-linear rheology law
- Utilize (C)Pardiso
- Timing of linear system solve
- Compare with Haswell node 24 cores



Conclusions

- If you have a system based on MIC's, you can deploy Elmer/Ice with reasonable performance (similar between Xeon and Xeon Phi)
- Multi-threading (OpenMP) has been introduced to many solvers and will continue
- Assembly can utilize SIMD (=vector units) if we apply p-bubbles for stabilization
- Improvements have equally positive impact on traditional CPU's (Xeon Hasswel, Broadwell)