Use It or Lose It: 
Software Perspectives on Building SoC’s for HPC

John Leidel
Compiler/Tools Lead
Micron Advanced Memory Systems

SoC for HPC Workshop: PACT2015
<Begin Rant>

• Where are we now?
• Why are we here?
• Where do we go next?
Disclaimer

• This talk largely reflects my personal experience and empirical data gathered throughout the last decade in building custom ISA’s/system architectures
  – IEEE/ACM wouldn’t dare allow me publish this

• It does not reflect the views of my employer
Where are we now?
20 Years of Architecture Dev

...and how the software has responded

• Multicore/Hierarchical Die
  – Shift from pure frequency scaling to complexity scaling
  – Programmers responded with thread/task parallelism
    • OpenMP, Cilk++, Intel TBB, etc

• Architecture Complexity
  – Shift from condensed ISA’s to very complex, expressive arithmetic instructions
  – Programmers responded with intrinsics/libraries
    • Revisit vectorizing compilers (eg, fine grained dependence analysis)

• SoC Development
  – Bringing traditionally peripheral devices & interfaces onto the die
  – Programmers didn’t really do anything… (we couldn’t!)

...I’m deliberating leaving out heterogeneity

SoC for HPC Workshop: PACT2015
What do the apps tell us?

Total Instructions

gcc 4.8.4 –O3 –ffast-math –mavx2, etc
Caveat: Doesn’t include runtime code!

SoC for HPC Workshop: PACT2015
...it would be interesting to perform a cross correlation
Compiler (User) Optimized Uniqueness

...doesn’t reflect runtime code, stdlib, machine-optimized libs
What does this tell us?

• X86_64 is >600 instructions
  – I’m not picking on Intel/AMD…
• We’re only using a small percentage of these in HPC
  – < ~20%!
• Why are we paying the price for this in hardware and software?
• Why can’t we get more instructions to help solve more problems associated with the SoC aspects of the system!?
Where do we go next?
ISA & System Arch Codeesign

• We need to develop a more holistic view of the SoC and the system
• Develop ISA extensions and methodologies (optimizations) that exploit SoC “peripheral” devices
• Minimize the latency to perform out-of-core operations!
  – I/O
  – Communication
  – Atomicity
  – Barrier/Synchronization

SoC for HPC Workshop: PACT2015
“Go to the mattresses”

Runtimes

- Look to the runtime software for areas of interest for SoC optimization
- Non-SIMD delineation of parallelism often occurs here
  - OpenMP 4.0 has explicit SIMD!
- Initial stages of off-chip and off-node communication is also initiated here
- Complex memory operations often occur here (broadcast, AMO, Tag-bits, etc)
- *Compilers and tools are already aware of these interfaces!* 

*Examine how the application interacts with the parallel system!*

SoC for HPC Workshop: PACT2015
It sounds simple, but...

- The notion of touching large portions of the system architecture from the ISA is difficult to do (correctly)
  - **Area concerns:**
    - We’ve found (from experience) that with added complexity comes a wider data path
    - Promoting wide data paths at scale can be expensive!
  - **Security concerns:**
    - Utilizing peripheral devices with integral access to the MMU or other hardware components
  - **Reliability concerns:**
    - Utilizing peripheral devices that may be able to silently corrupt application state
    - DMA devices, MMU’s, expressive AMO’s
  - **Orthogonality:**
    - How simple can we keep the system? How different will the eventual ISA be?
    - EG, do we need to re-architect our entire software stack?

*SoC for HPC Workshop: PACT2015*
...but we also have some examples

- **Tera MTA**
  - Tag-bit memories (new ISA)

- **Cray T3D**
  - Remote memory operation support (Alpha)

- **IBM Cyclops64**
  - Software managed scratchpads (Power)

- **Convey MX-100**
  - Tag-bit memories & global addressing (new ISA+CNY Scalar)
Convey MX-100 Development

- Original machine/ISA development based upon a dataflow simulator
- Operations were abstracted to reflect a simple latency value
- Memory dependencies for sequential access and concurrency (communication) was a first order design primitive
- Analyzed benchmarks, applications, pathological kernels using parallel algorithms

Google Pagerank on 512 threads

Total cycle count to kernel completion
Memory utilization
Core utilization

SoC for HPC Workshop: PACT2015
CLR Barrier Scaling Against CHOMP OpenMP

~40% faster than OMP

Default CHOMP Personality, scaling to 3024 threads

Toward a Scalable Heterogeneous Runtime System for Convey MX Architecture, MTAAP 13’

SoC for HPC Workshop: PACT2015
Lets Continue the Discussion

jleidel@micron.com

john.leidel@ttu.edu
Backup
Unique SIMD Instructions

![Bar chart showing unique SIMD instructions for various benchmarks.](chart.png)