The Stampede is Coming: A New Petascale Resource for the Open Science Community

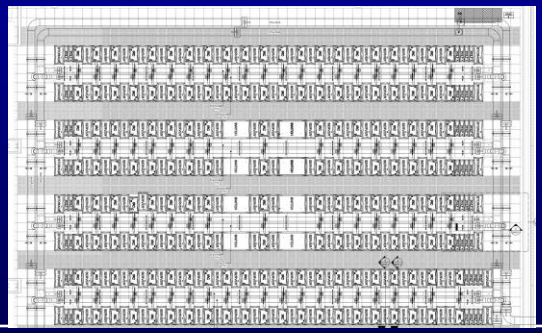
Dan Stanzione
Deputy Director
Texas Advanced Computing Center
dan@tacc.utexas.edu

The Texas Advanced Computing Center:
A Leader in High Performance Computing

***1,000,000x** performance increase in computing capability in 10 years.*

Ranger: 62,976 Processor Cores, 123TB RAM, 579 TeraFlops, Fastest Open Science Machine in the World, 2008

Lonestar: 23,000 processors, 44TB RAM, Shared Mem and GPU subsystems, #25 in the world 2011



Stampede: Coming soon…

TACC

Stampede

- A new NSF-funded XSEDE resource at TACC
- Funded two components of a new system:
  - Two+ petaflop, 100,000+ core Xeon E5 based Dell cluster to be the new workhorse of the NSF open science community (>3x Ranger)
  - Up to eight additional petaflops of Intel Many-Integrated-Core (MIC) Xeon Phi processors to provide a revolutionary innovative capability: a new generation of supercomputing.
  - Change the power/performance curves of supercomputing without changing the programming model (terribly much).

Stampede - Headlines

- Up to 10PF Peak performance in initial system (Early 2013)
  - 100,000+ conventional Intel processor cores
  - Almost 500,000 total cores with Intel Xeon Phi
- Upgrade with “Future Knight’s” in 2015
- 14PB+ disk
- 272TB+ RAM
- 56Gb FDR InfiniBand interconnect
- Nearly 200 racks of compute hardware (10,000 sq ft)
- **>SIX MEGAWATTS total power**

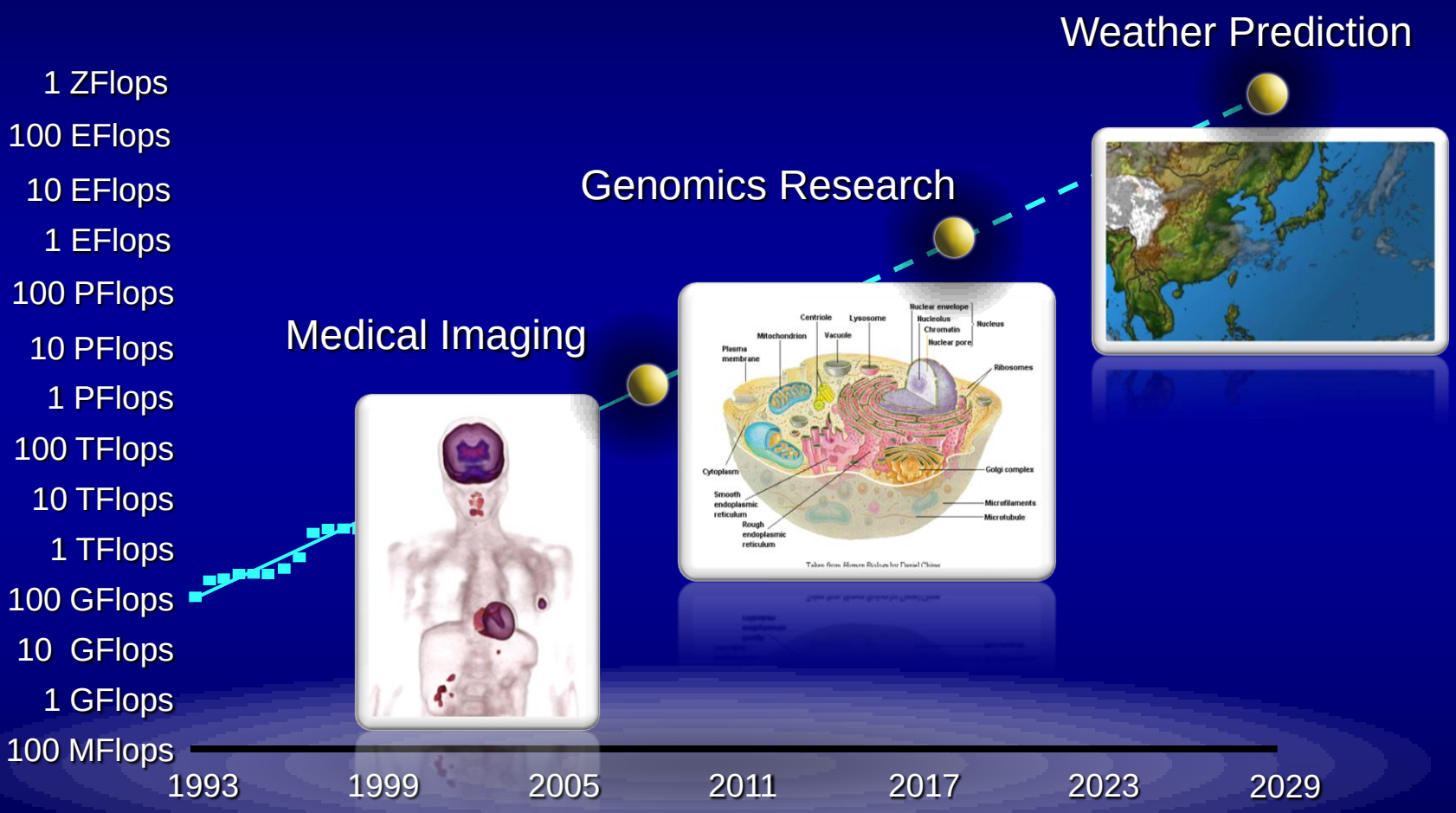
The Stampede Project

- A Complete Advanced Computing Ecosystem:
  - HPC Compute, storage, interconnect
  - Visualization subsystem (144 high end GPUs)
  - Large memory support (16 1TB nodes)
  - Integration with archive, and (coming soon) TACC global work filesystem and other data systems
  - People, training, and documentation to support computational science
- Hosted at an expanded building in TACC; massive power upgrades
  - 12MW new total datacenter power
  - Thermal energy storage to reduce operating costs

Contrast with Blue Waters

- Stampede is tasked with the “broad” instead of “deep” mission.
  - We estimate more than 1,000 projects will be supported on Stampede.
  - Blue Waters will handle a few of the very largest problems.
  - Stampede will run big problems too, but also lots and lots of small and midsize jobs.
- Smaller budget, shorter timescale.
- Stampede will be part of XSEDE

HPC Needs Decades of Moore's Law



Data Source: www.top500.org (Courtesy Intel)

TACC

An Inflection point (or two) in High Performance Computing

- Relatively “boring”, but rapidly improving architectures for the last 15 years.
- Performance rising much faster than Moore’s Law…
  - But power rising faster
  - And concurrency rising faster than that, with serial performance decreasing.
- Something had to give

Consider this Example

Homogeneous v. Heterogeneous

Acquisition costs and power consumption

- Homogeneous system: K computer in Japan
  - 10 PF unaccelerated
  - Rumored ~$1 billion acquisition costs
  - Large power bill (13MW) and foot print (800 racks)
- Heterogeneous system: Stampede at TACC
  - Near 10PF (2 SNB + up to 8 MIC)
  - Modest footprint: 8,000 ft^2, 5MW, 180 racks.
  - $27.5 million

Accelerated Computing for Exascale

- Exascale systems, predicted for 2018, would have required 500MW on the “old” curves.
- Something new was clearly needed.
- The “accelerated computing” movement was reborn (this happens periodically, starting with 387 math coprocessors).

Key aspects of acceleration

- We have lots of transistors… Moore's law is holding; this isn't necessarily the problem.
- We don't really need lower power per transistor, we need lower power per *operation*.
- How to do this?

The GPU Approach

- Don’t repeat the past mistakes of failed accelerators (i.e. have another market to drive volume and development).
- Streaming model
- Simpler processors, lots of concurrency (work by compilers or programmers doesn’t cost power at runtime).
- Very large die (tolerate failures to get yield).

More accelerators

- nVidia leads this charge in GPU, but there are many similar competing ideas:
  - ARM (Marvell/Calxeda) (simple cores, lots per die, low power per operation).
  - AMD Fusion (AMD cores plus GPU cores on same die).
  - FPGA (large arrays, lower power by special purpose circuit per application

Intel's MIC approach

- Since the days of RISC vs. CISC, Intel has mastered the art of figuring out what is important about a new processing technology, and saying "why can't we do this in x86?"
- The Intel Many Integrated Core (MIC) architecture is about large die, simpler circuit, much more parallelism, in the x86 line.

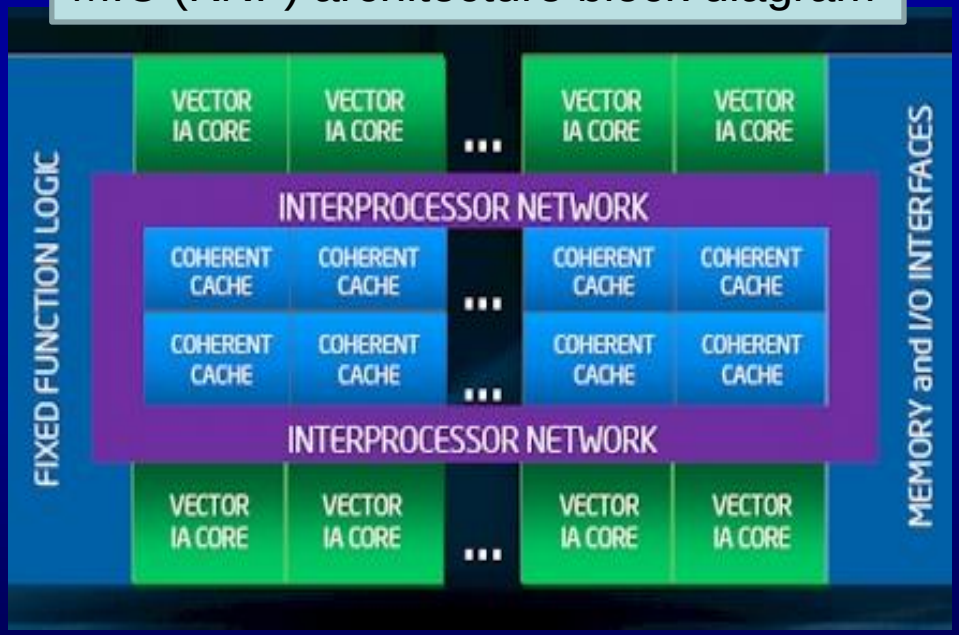
What is MIC?

- Basic Design Ideas
  - Leverage x86 architecture (CPU with many cores)
    - x86 cores that are simpler, but allow for more compute throughput
  - Leverage (Intel's) existing x86 programming models
  - Dedicate much of the silicon to floating point ops., keep some cache(s)
  - Keep cache-coherency protocol
  - Increase floating-point throughput per core
  - Implement as a separate device
  - Strip expensive features (out-of-order execution, branch prediction, etc.)
  - Widen SIMD registers for more throughput
  - Fast (GDDR5) memory on card

MIC Architecture

- Many cores on the die
- L1 and L2 cache
- Bidirectional ring network
- Memory and PCIe connection

MIC (KNF) architecture block diagram



Knights Ferry –SW Dev Platform

- Up to 32 cores
- 1-2 GB of GDDR5 RAM
- 512-bit wide SIMD registers
- L1/L2 caches
- Multiple threads (up to 4) per core
- Slow operation in double precision

Xeon PHI (was Knights Corner)

- First product
- Used in Stampede
- 50+ cores
- Increased amount of RAM
- Details are under NDA
- 22 nm technology

Stampede: Programming Models

- A ~2PF Xeon-only system (MPI, OpenMP)
- A ~8PF MIC-only system (MPI, OpenMP)
- A ~10PF heterogeneous system (MPI, OpenMP)
  - Run separate MPI tasks on Xeon vs. MIC, or use OpenMP extensions for offload for hybrid programs.

Stampede

- Base Cluster:
  - Intel Sandy Bridge processors
  - Dell dual-socket nodes w/32GB RAM
  - More than 6,000 nodes
  - More than 100,000 cores
- Co-Processors:
  - Intel “MIC” Many Integrated Core processors
  - Special release of “Knight’s Corner” (>50 cores)
  - “Knight’s Ferry” development platforms at TACC now
- Total Concurrency approaching 500,000 cores

Power/Physical

- Stampede will physically use 182 48U cabinets.
- Power density (after upgrade in 2015) will exceed 40kW per rack.
- Estimated 2015 peak power is 6.2MW.

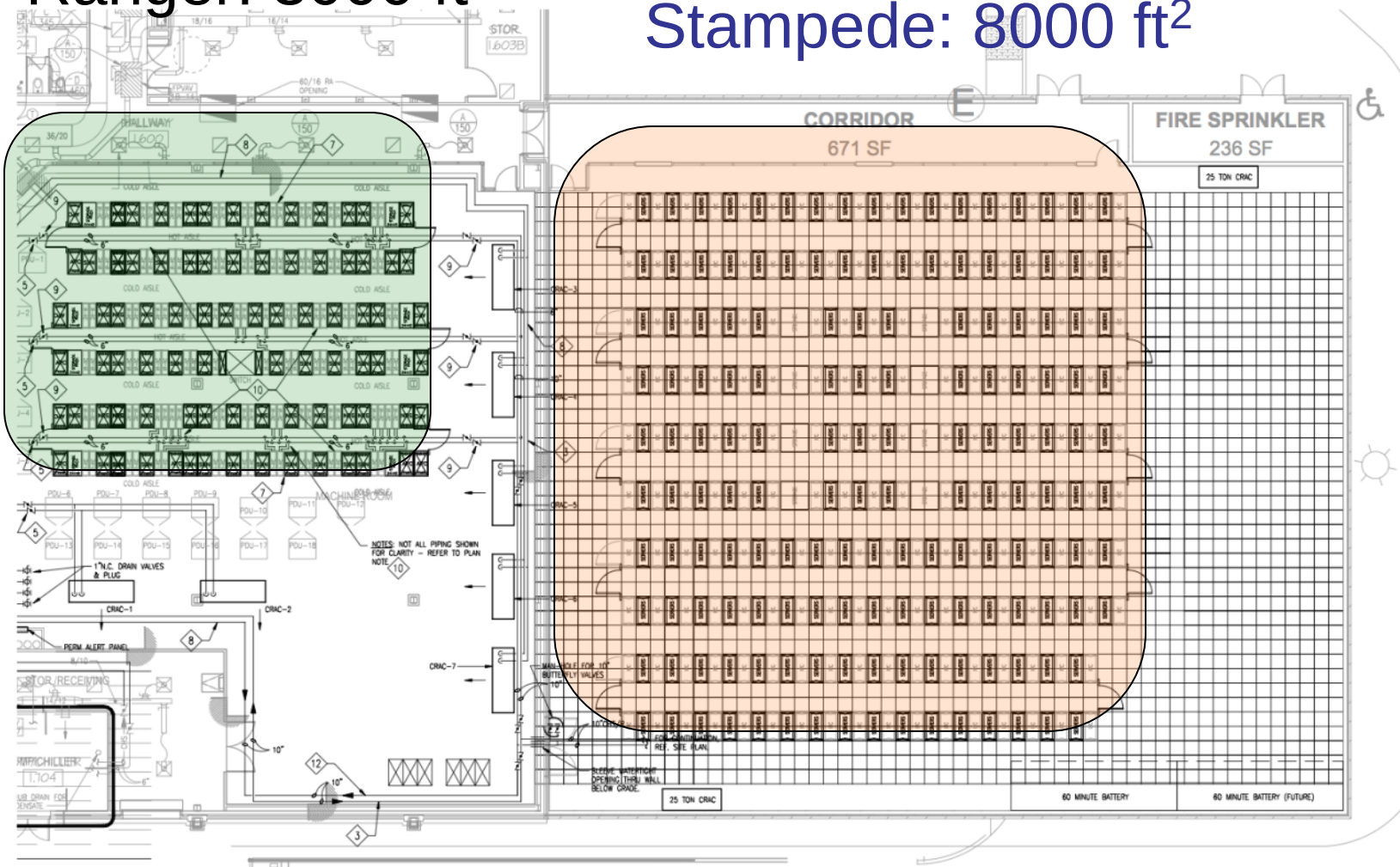
Key Datacenter Features

- Thermal energy storage to reduce peak power charges
- Hot aisle containment to boost efficiency (and simply provide enough cooling).
- Total IT power to 9.5MW, total power ~12MW.
- Expand experiments in mineral oil cooling.

Datacenter Expansion

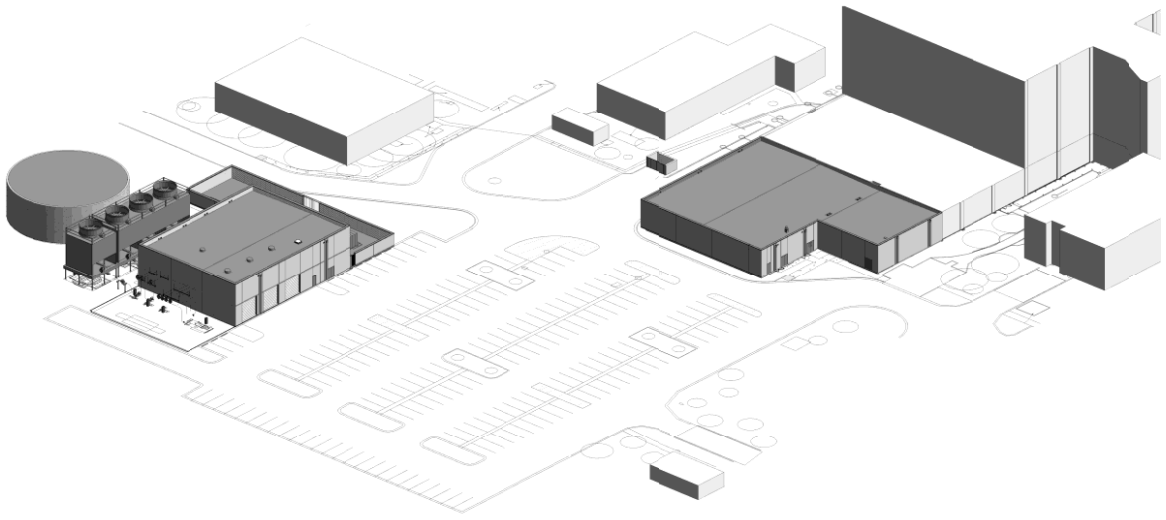
Ranger: 3000 ft^2

Stampede: 8000 ft^2



HIGH PERFORMANCE COMPUTING FACILITY EXPANSION

TEXAS ADVANCED COMPUTING FACILITY
THE UNIVERSITY OF TEXAS



95% DESIGN DEVELOPMENT / GMP PRICING SET

TACC

Stampede Datacenter –February 20th

TACC

Stampede Datacenter – March 22nd

TACC

Stampede Datacenter – May 16th

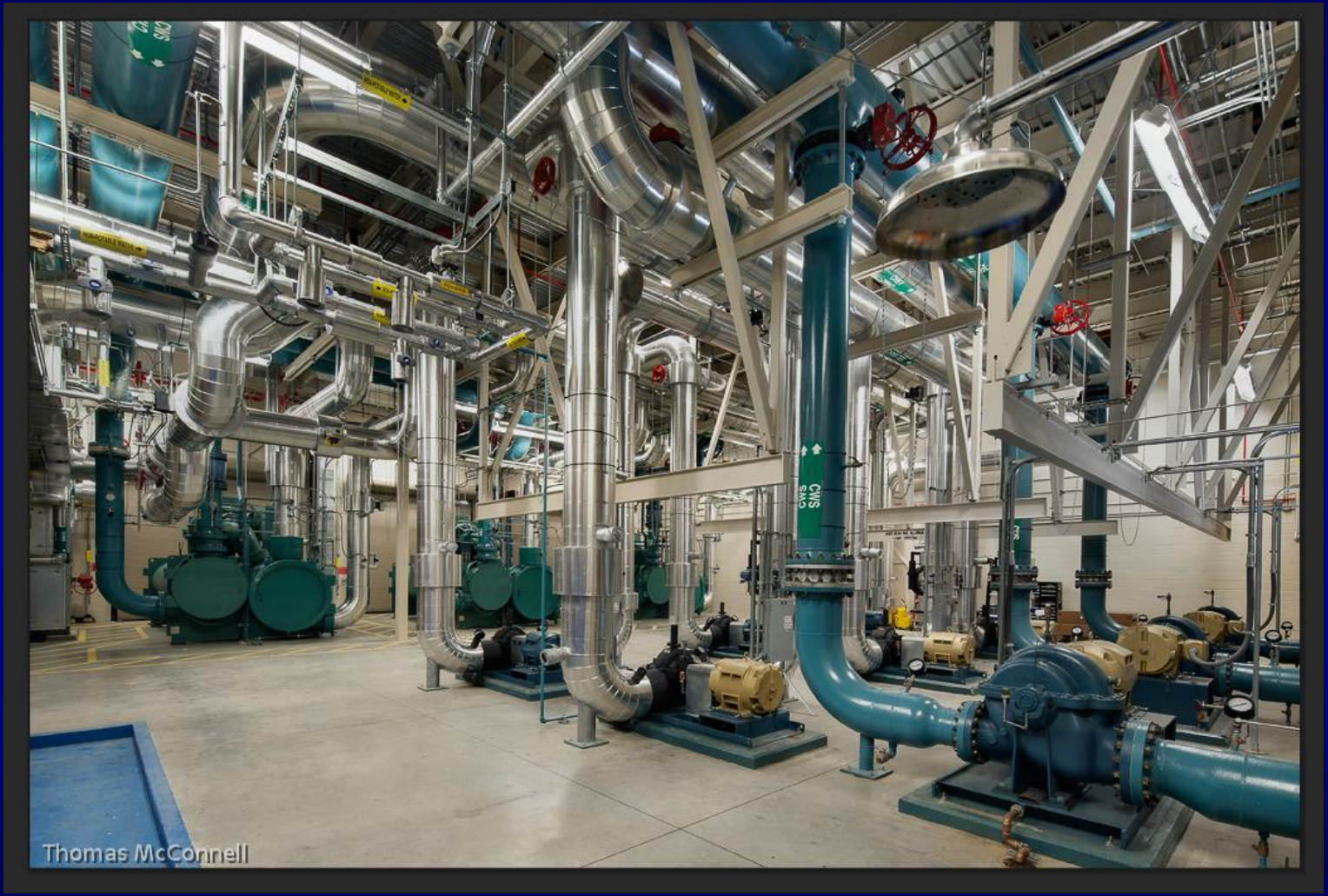
Stampede Datacenter – June 20th

Stampede Datacenter, ~September 10th

TACC

Stampede Datacenter, ~September 10th

Some utilities are involved



Actually, way more utility space than machine space

Turns out the utilities for the datacenter costs more, takes more time and more space than the computing systems

TACC

New Technologies/Milestones in the Stampede Project

- Density: surpasses 40KW/Cabinet
  - New Dell node designs to support multiple 300W expansion cards in single node ~1U
- Total system power past 5 *megawatts*
  - Thermal storage technology incorporated.
- Breakthrough price/performance and power/performance.
  - Inclusion of Intel MIC; but we must program it.
- Application concurrency past 1 *million* threads per application

What we at TACC like about MIC

- Intel's MIC is based on x86 technology
  - x86 cores w/ caches and cache coherency
  - SIMD instruction set
- Programming for MIC is similar to programming for CPUs
  - Familiar languages: C/C++ and Fortran
  - Familiar parallel programming models: OpenMP & MPI
  - MPI on host and on the coprocessor
  - Any code can run on MIC, not just kernels
- Optimizing for MIC is similar to optimizing for CPUs
  - **"Optimize once, run anywhere"**
  - Our early MIC porting efforts for codes "in the field" are frequently doubling performance on Sandy Bridge.

Will My Code Run on MIC?

- Yes
- That's the wrong question, it's:
  - Will your code run *best* on MIC?, or
  - Will you get great MIC performance without additional work?

Early MIC Programming Experiences at TACC

- Codes port easily
  - Minutes to days depending mostly on library dependencies
- Performance can require real work
  - While the sw environment continues to evolve
  - Getting codes to run *at all* is almost too easy; really need to put in the effort to get what you expect
- Scalability is pretty good
  - Multiple threads per core *really important*
  - Getting your code to vectorize *really important*

Adapting and Optimizing Code for MIC

- Hybrid Programming
  - MPI + Threads (OpenMP, etc.)

Today "Any" resource

- Optimize for higher thread performance
  - Minimize serial sections and synchronization
- Test different execution models
  - Symmetric vs. Offloaded
- Optimize for L1/L2 caches

Today/Soon Knights Ferry

- Test performance on MIC
- Optimize for specific architecture
- Start production

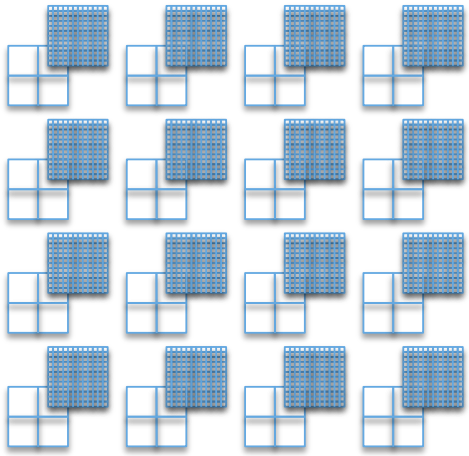
Stampede MIC 2013+

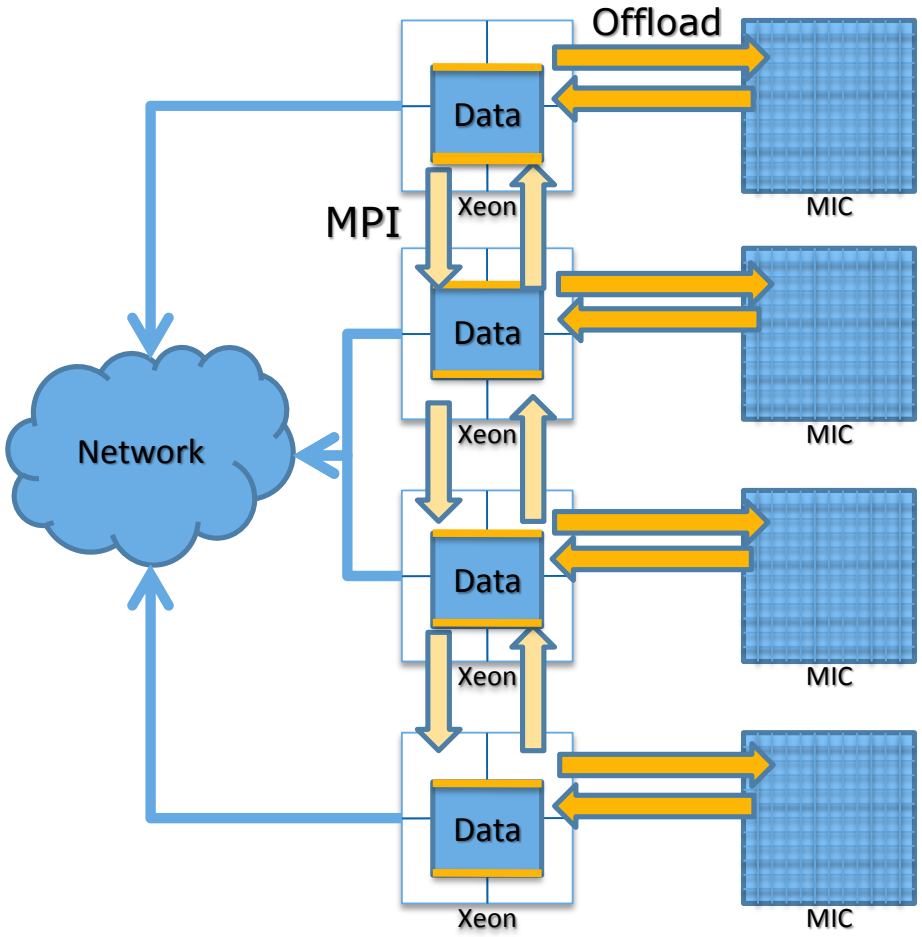
Stampede Programming
Five Possible Models

- Host-only
- Offload
- Symmetric
- Reverse Offload
- MIC Native

Programming Intel® MIC-based Systems
MPI+Offload

- MPI ranks on Intel® Xeon® processors (only)
- All messages into/out of processors
- Offload models used to accelerate MPI ranks
- Intel® Cilk™ Plus, OpenMP*, Intel® Threading Building Blocks, Pthreads* within Intel® MIC
- Homogenous network of hybrid nodes:





Offload Code Examples

- C/C++ Offload Pragma

```
#pragma offload target (mic)
#pragma omp parallel for reduction(+:pi)
 for (i=0; i<count; i++)  {
       float t = (float)((i+0.5)/count);
       pi += 4.0/(1.0+t*t);
 }
 pi /= count;
```

- Function Offload Example

```
#pragma offload target(mic)
      in(transa, transb, N, alpha, beta) \
      in(A:length(matrix_elements)) \
      in(B:length(matrix_elements)) \
      inout(C:length(matrix_elements))
        sgemm(&transa, &transb, &N, &N, &N,
&alpha, A, &N, B, &N, &beta, C, &N);
```

- Fortran Offload Directive

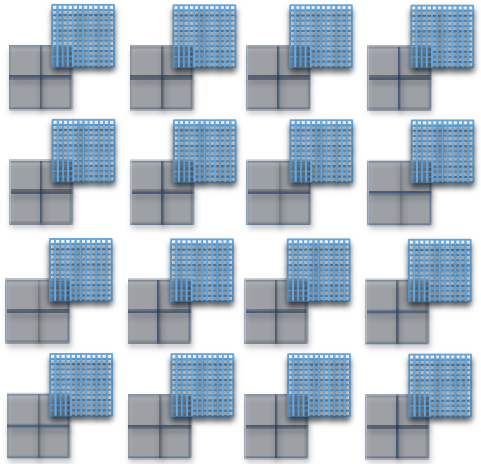
```
!dir$ omp offload target(mic)
!$omp parallel do
     do i=1,10
               A(i) = B(i) * C(i)
     enddo
```

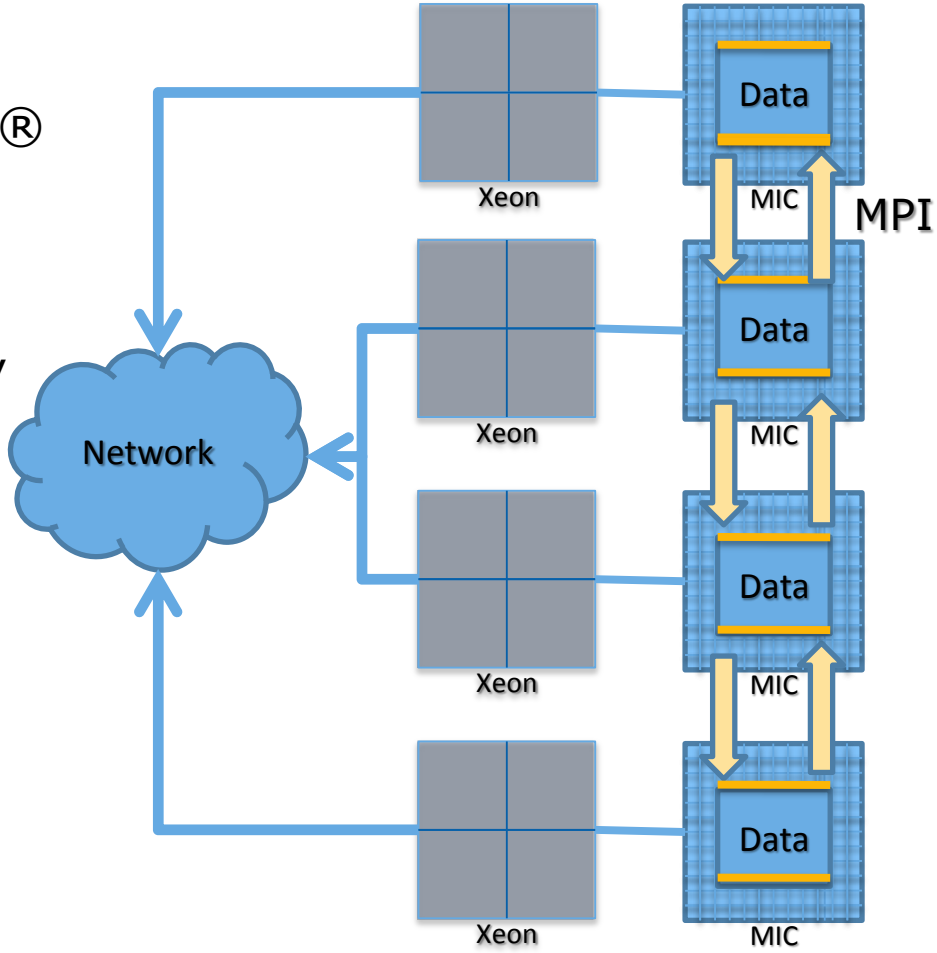
- C/C++ Language Extension

```
class _Cilk_Shared common  {
    int data1;
    char *data2;
    class common *next;
    void process();
};
_Cilk_Shared class common  obj1, obj2;
_Cilk_spawn   _Offload obj1.process();
_Cilk_spawn                obj2.process();
```

Programming Intel® MIC-based Systems
Many-core Hosted

- MPI ranks on Intel® MIC (only)
- All messages into/out of Intel® MIC
- Intel® Cilk™ Plus, OpenMP*, Intel® Threading Building Blocks, Pthreads used directly within MPI processes
- Programmed as homogenous network of many-core CPUs:

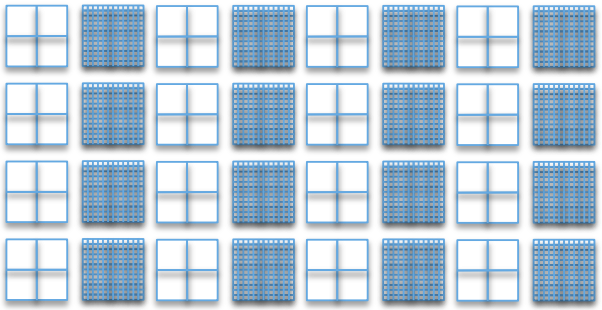


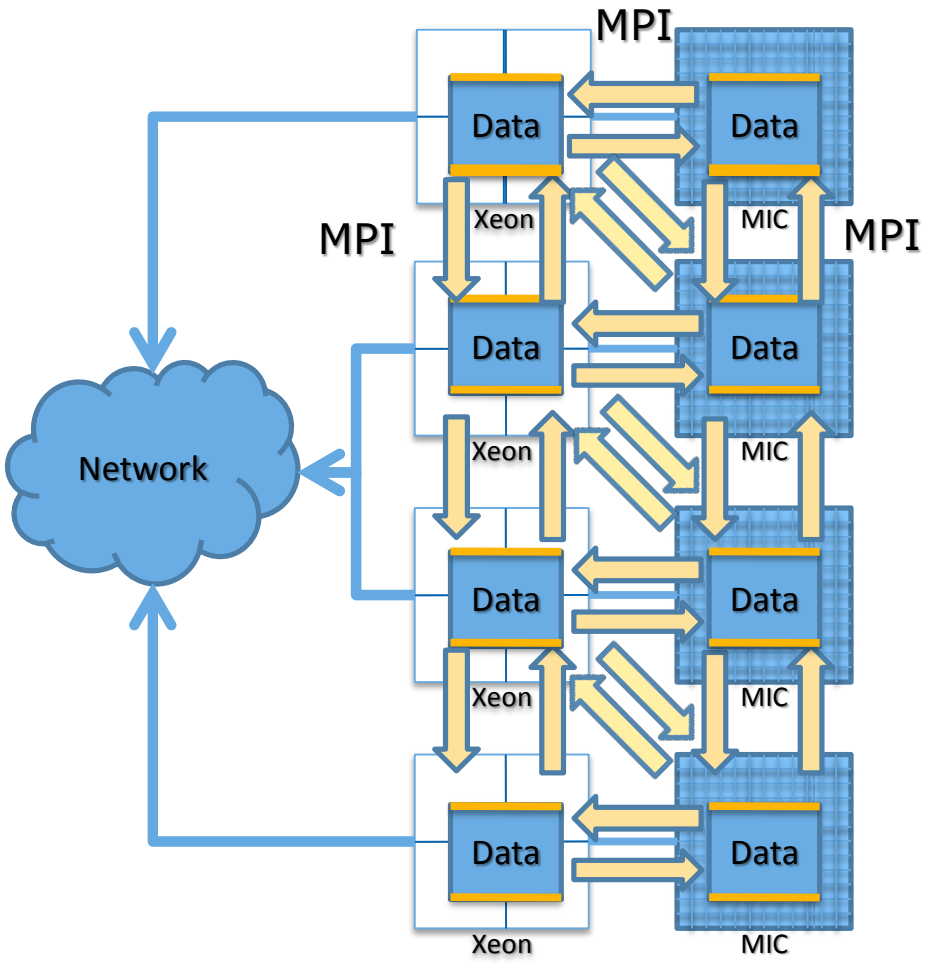


Programming Intel® MIC-based Systems
Symmetric

- MPI *ranks* on Intel® MIC and Intel® Xeon® processors
- Messages to/from any core
- Intel® Cilk™ Plus, OpenMP*, Intel® Threading Building Blocks, Pthreads* used directly within MPI processes

- Programmed as heterogeneous network of homogeneous nodes:





Programming Models

Ready to use on day one

- TBB's will be available to C++ programmers
- MKL will be available
  - *Automatic offloading* by compiler for some MKL features
  - (probably most transparent mode)
- Cilk Plus
  - Useful for task-parallel programing (add-on to OpenMP)
  - May become available for Fortran users as well
- OpenMP
  - TACC expects that OpenMP will be the most interesting programming model for our HPC users

MIC Programming with OpenMP

- MIC specific pragma precedes OpenMP pragma
  - Fortran: `!dir$ omp offload target(mic) <…>`
  - C: `#pragma offload target(mic) <…>`
- All data transfer is handled by the compiler (optional keywords provide user control)

- Options for:
  - Automatic data management
  - Manual data management
  - I/O from within offloaded region
    - Data can “stream” through the MIC; no need to leave MIC to fetch new data
    - Also very helpful when debugging (print statements)
  - Offloading a subroutine and using MKL

The Stampede is Coming

- Stampede has a *lot* of new technologies, and as such has a certain element of risk.
- However, as of now, we expect the early user/science period to begin in December.
- Full production by January 7th (Base system at least).
- You might hear some more about this machine at SC12 in November ☺.
- Thank to NSF, Intel, Dell

TACC

Roadmap

What comes next?

How to get prepared?

- Many HPC applications are pure-MPI codes
- Start thinking about upgrading to a hybrid scheme
- Adding OpenMP is a larger effort than adding MIC directives
- Special MIC/OpenMP considerations
  - Many more threads will be needed: 50+ cores (Production KNC) → 50+/100+/200+ threads
  - Good OpenMP scaling will be (much) more important
  - Vectorize, vectorize, vectorize
  - There is no unified last-level cache (LLC): Cache shared among cores, several MB on CPUs

Stay tuned
The coming years will be exciting …

Can I use Stampede?

- Yes!
- We expect to have 5,000-10,000 academic users next year.
- See Jeff's talk about XSEDE later today.
- Every academic researcher can apply for a startup account, and eventually larger.

At this point, there is a very slight chance I've left time for questions.

(Certainly, I've created enough confusion for them)

Thanks for listening!

dan@tacc.utexas.edu