USQCD Software

All Hands Meeting FNAL, May 1, 2014
Rich Brower Chair of Software Committee

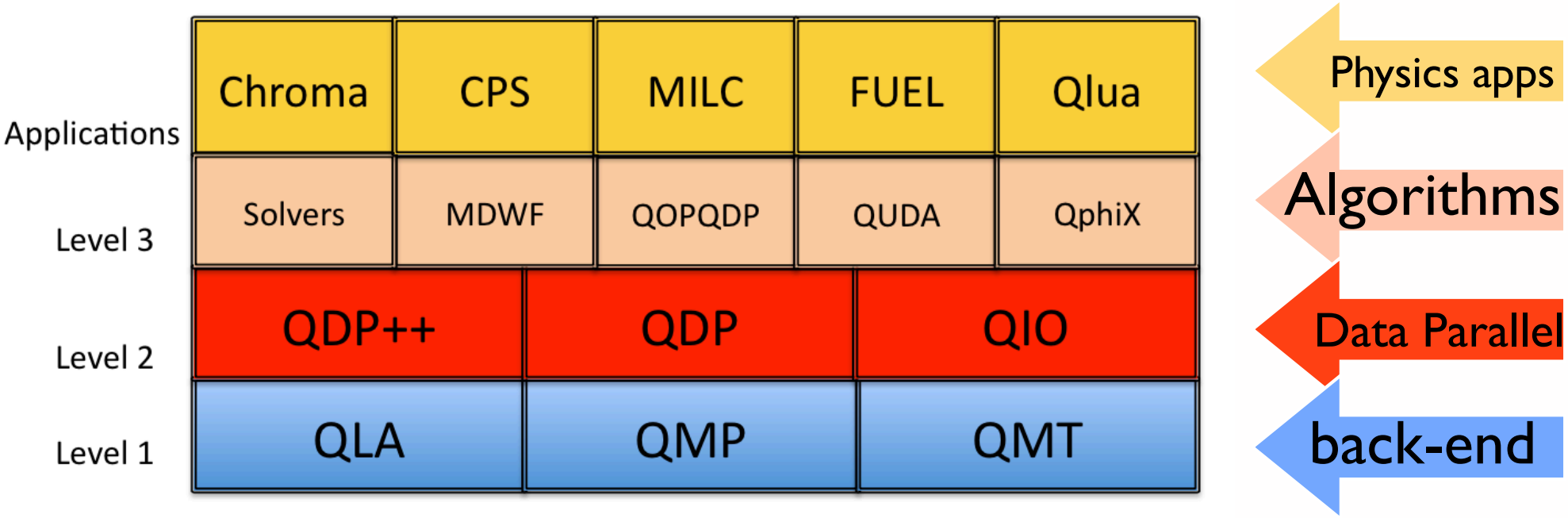
Not possible to summarize status in any detail of course.

- Recent document available on request
  - 2 year HEP SciDAC 3.5 proposal (Paul Mackenzie)
  - NP Physics Midterm Review (Frithjof Karsch)
  - CARR proposal (Balint Joo)

Major USQCD Participants

- ANL: James Osborn, Meifeng Lin, Heechang Na
- BNL: Frithjof Karsch, Chulwoo Jung, Hyung-Jin Kim,S. Syritsyn,Yu Maezawa
- Columbia: Robert Mawhinney, Hantao Yin
- FNAL: James Simone, Alexei Strelchenko, Don Holmgren, Paul Mackenzie
- JLab: Robert Edwards, Balint Joo, Jie Chen, Frank Winter, David Richards
- W&M/UNC: Kostas Orginos, Andreas Stathopoulos, Rob Fowler (SUPER)
- LLNL: Pavlos Vranas, Chris Schroeder, Rob Faulgot (FASTMath), Ron Soltz
- NVIDIA: Mike Clark, Ron Babich
- Arizona: Doug Toussaint, Alexei Bazavov
- Utah: Carleton DeTar, Justin Foley
- BU: Richard Brower, Michael Cheng, Oliver Witzel
- MIT: Pochinsky Andrew, John Negele,
- Syracuse: Simon Catterall, David Schaich
- Washington: Martin Savage, Emanuell Chang
- Many Others: Peter Boyle, Steve Gottlieb, George Fleming et al
- "Team of Rivals" (Many others in USQCD and Int'l Community volunteer to help!)

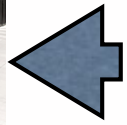
USQCD Software Stack



On line distribution: http://usqcd.jlab.org/usqcd-software/

Very successful but after 10+ Years it is showing it's age:

Top priority: Physics on existing Hardware



Applications	Chroma	CPS	MILC	FUEL	Qlua
Level 3	Solvers	MDWF	QOPQDP	QUDA	QphiX
Level 2	QDP++		QDP		QIO
Level 1	QLA		QMP		QMT

Argonne NATIONAL LABORATORY

IBM

OAK RIDGE National Laboratory

NVIDIA

TACC

intel

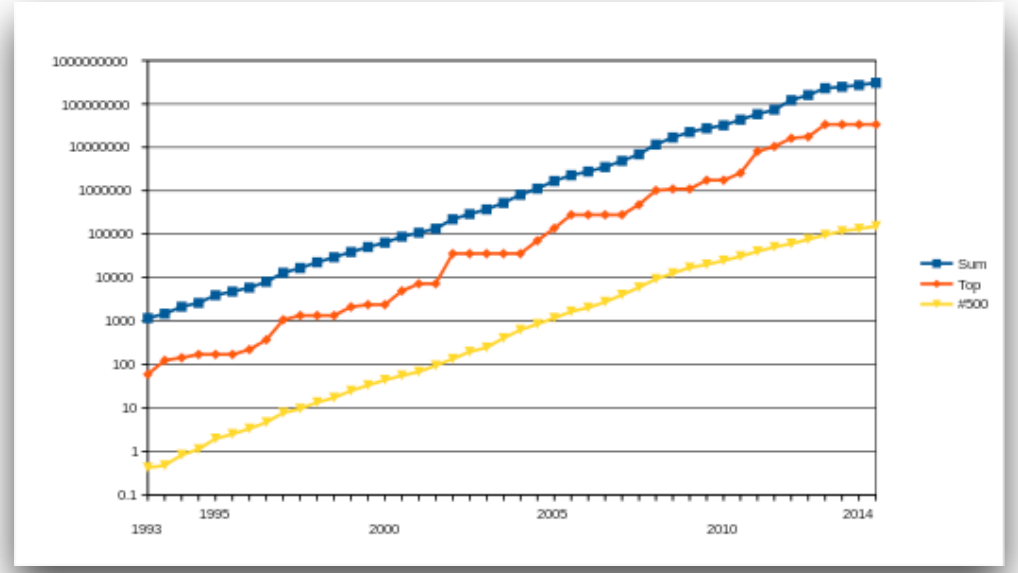
NCSA

CRAY X1

GOOD NEWS: Lattice Field Theory Coming of Age

K. Wilson: "Lecture at Lattice 1989 Capri"

"lattice gauge theory could also require a 10^8 increase in computer power AND spectacular algorithmic advances before useful interactions with experiment ..."



CM-2 100 Mflops (1989) 10^7 increase in 25 years *BF/Q 1 Pflops (2012)*

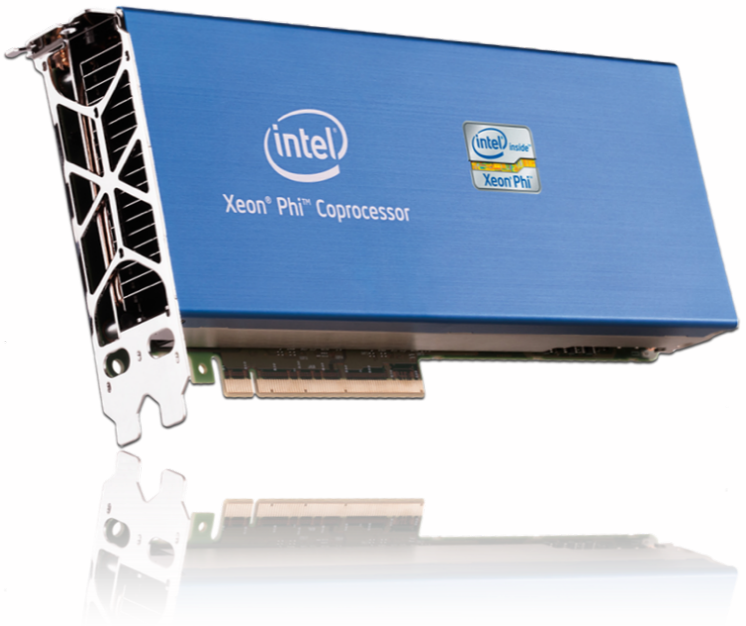
Future GPU/PHI architectures will soon get us there!

What about spectacular Algorithms/Software?

Next 2 year & beyond to SciDAC 4?

- Prepare for INTEL/CRAY CORAL
  - Strong collaboration with Intel Software Engineers: QphiX
  - 3 NESPAS for CORI at NERCS
- Prepare for IBM/NVIDA CORAL (Summit & Sierra)
  - Strong Collaboration with NVIDIA Software Engineers: QUDA
- Many New Algorithms on Drawing Board
  - Multi-grid for Staggered, introduce into HMC & fast Equilibration
  - Deflation et al for Disconnected Diagrams
  - Multi-quark and Excited State Sources.
  - Quantum Finite Element Methods (You got to be kidding?)
- Restructuring Data Parallel Back End
  - QDP-JIT (Chroma/JLab).
  - GridX (CPS/Edinburgh),
  - FUEL(MILC/ANL),
  - Qlua(MIT),

Multi-core Libraries

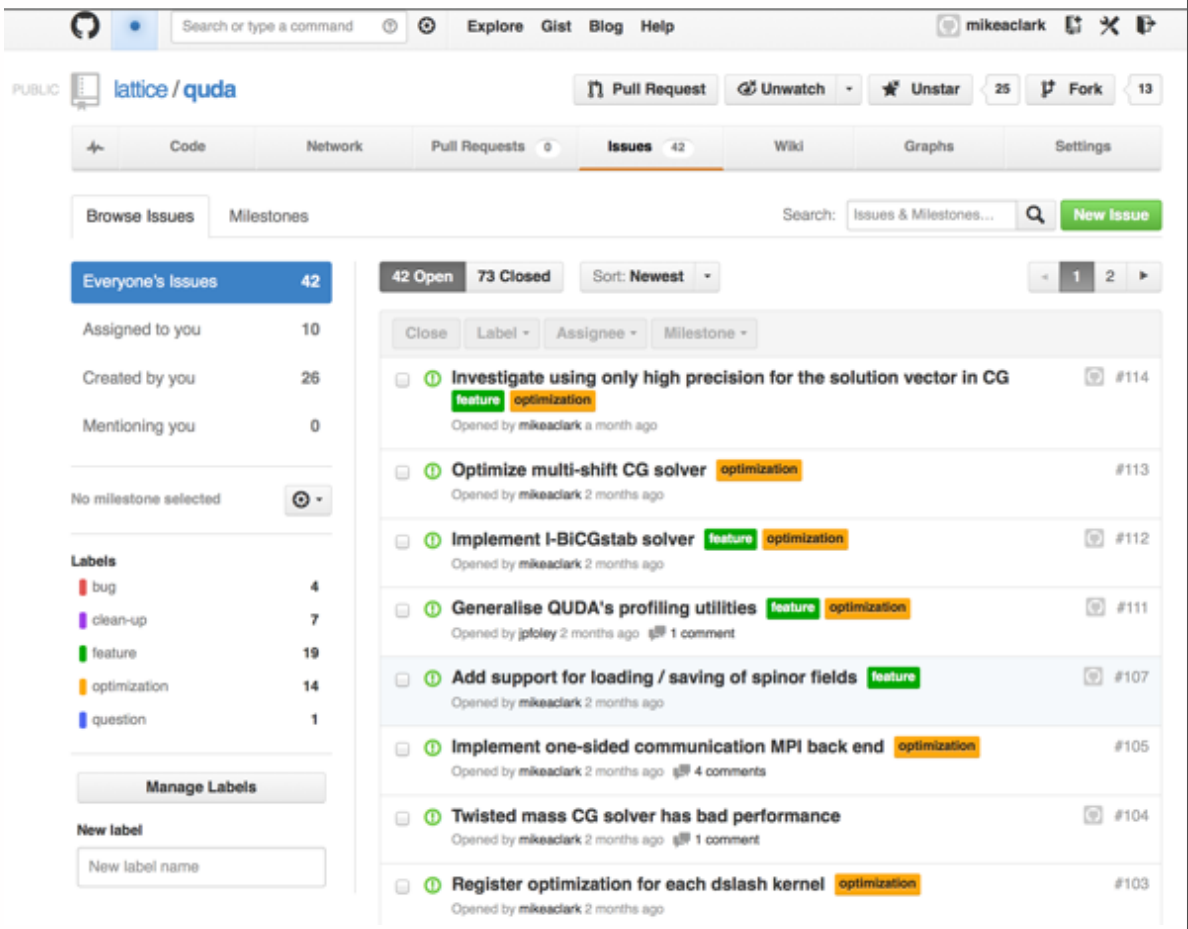


- The CORAL initiative in next two years will coincide with both NVIDIA/IBM and INTEL/CRAY rapidly evolving their architectures and programming environment with unified memory, higher bandwidth to memory and interconnect etc.

QUDA: NVIDIA GPU

- "QCD on CUDA" team – http://lattice.github.com/quda

- Ron Babich (BU-> NVIDIA)
- Kip Barros (BU ->LANL)
- Rich Brower (Boston University)
- Michael Cheng (Boston University)
- Mike Clark (BU-> NVIDIA)
- Justin Foley (University of Utah)
- Steve Gottlieb (Indiana University)
- Bálint Joó (Jlab)
- Claudio Rebbi (Boston University)
- Guochun Shi (NCSA -> Google)
- Alexei Strelchenko (Cyprus Inst.-> FNAL)
- Hyung-Jin Kim (BNL)
- Mathias Wagner (Bielefeld -> Indiana Univ)
- Frank Winter (UoE -> Jlab)



GPU code Development

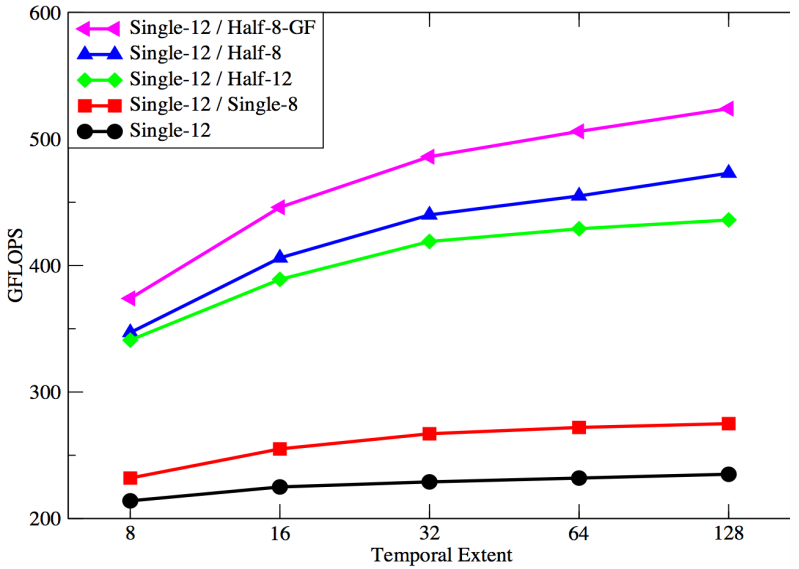
- **SU(3) matrices are all unitary complex matrices with det = 1**
  - **12-number parameterization: reconstruct full matrix on the fly in registers**

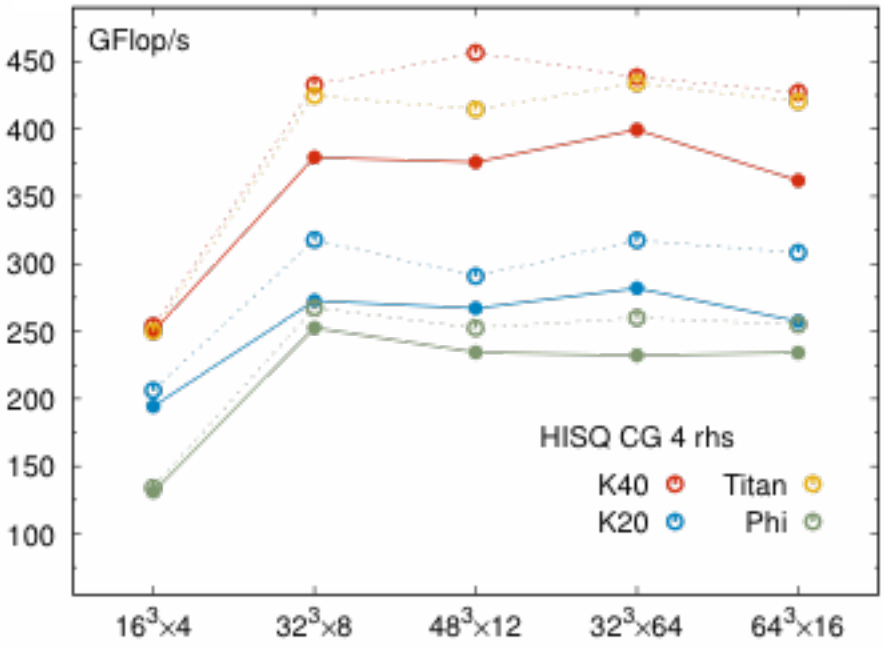
$$\begin{pmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{pmatrix} \rightarrow \begin{pmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{pmatrix} \quad \mathbf{c} = (\mathbf{a} \times \mathbf{b})^*$$

Group Manifold: $S_3 \times S_5$

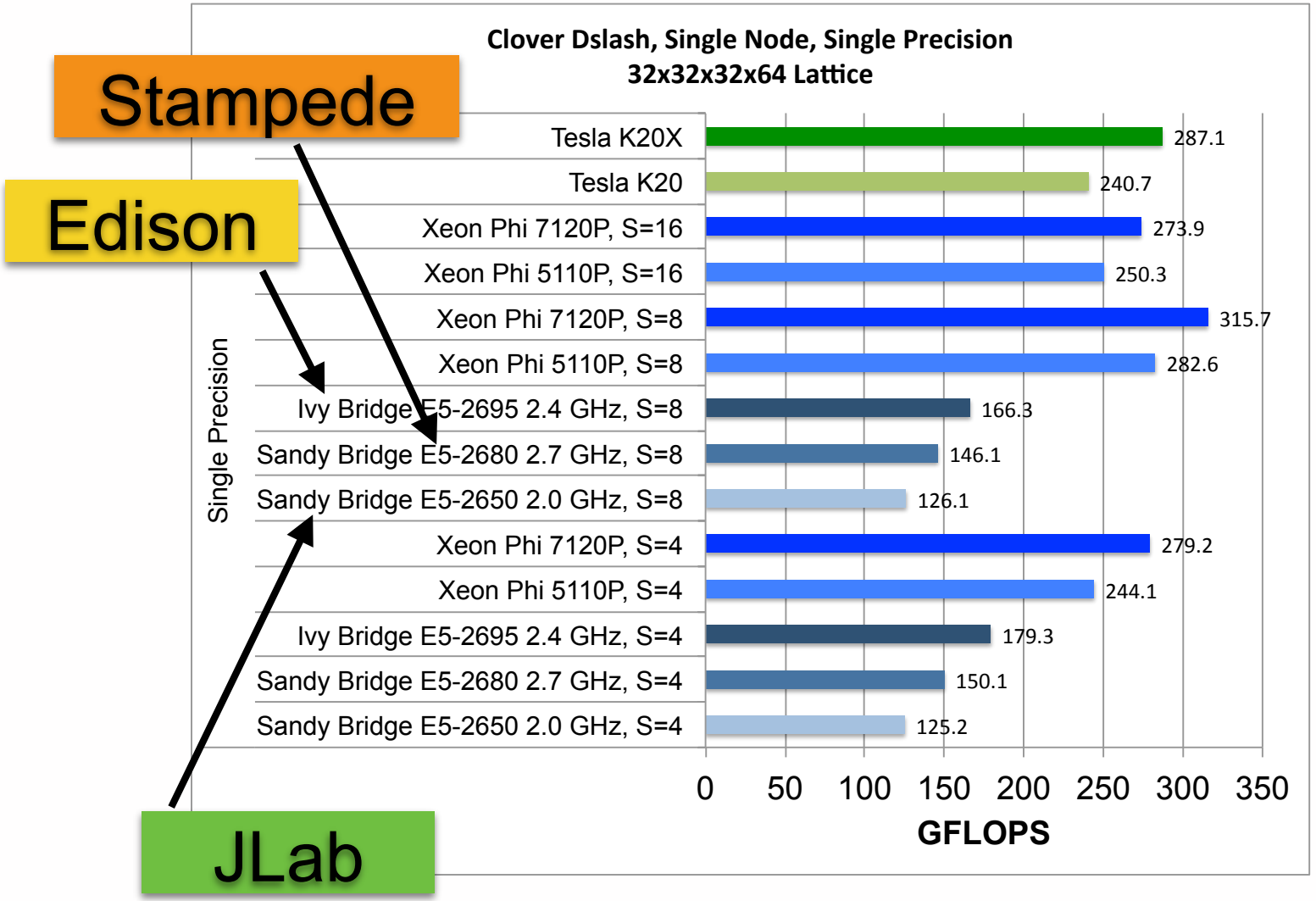
  - **Additional 384 flops per site**
  - **Also have an 8-number parameterization of SU(3) manifold (requires sin/cos and sqrt)**

- **Impose similarity transforms to increase sparsity**
- **Still memory bound - Can further reduce memory traffic by truncating the precision**
  - **Use 16-bit fixed-point representation**
  - **No loss in precision with mixed-precision solver**
  - **Almost a free lunch (small increase in iteration count)**



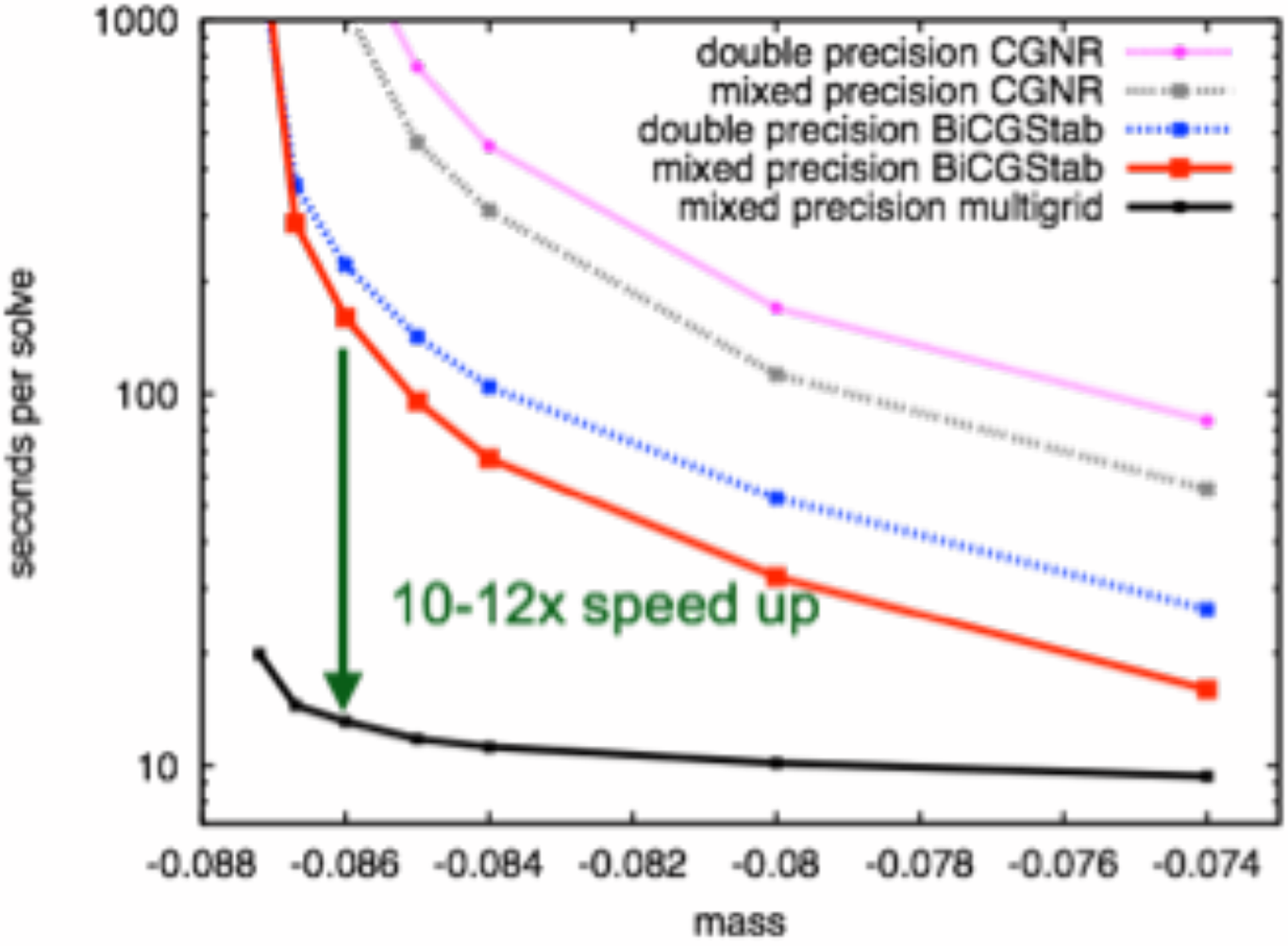


Xeon Phi and x86 Optimization



Performance of Clover-Dslash operator on a Xeon Phi Knight's Corner and other Xeon CPUs as well as NVIDIA Tesla GPUs in single precision using 2-row compression. Xeon Phi is competitive with GPUs. The performance gap between a dual socket Intel Xeon E5-2695 (Ivy Bridge) and the NVIDIA Tesla K20X in single precision is only a factor of 1.6x.

Multigrid (or Wilson Lattice Renormalization Group for Solvers)



20 Years of QCD MULTIGRID
In 2011 **Adaptive SA MG [3]** successfully extended the 1991 **Projective MG [2]** for algorithm to long distances.

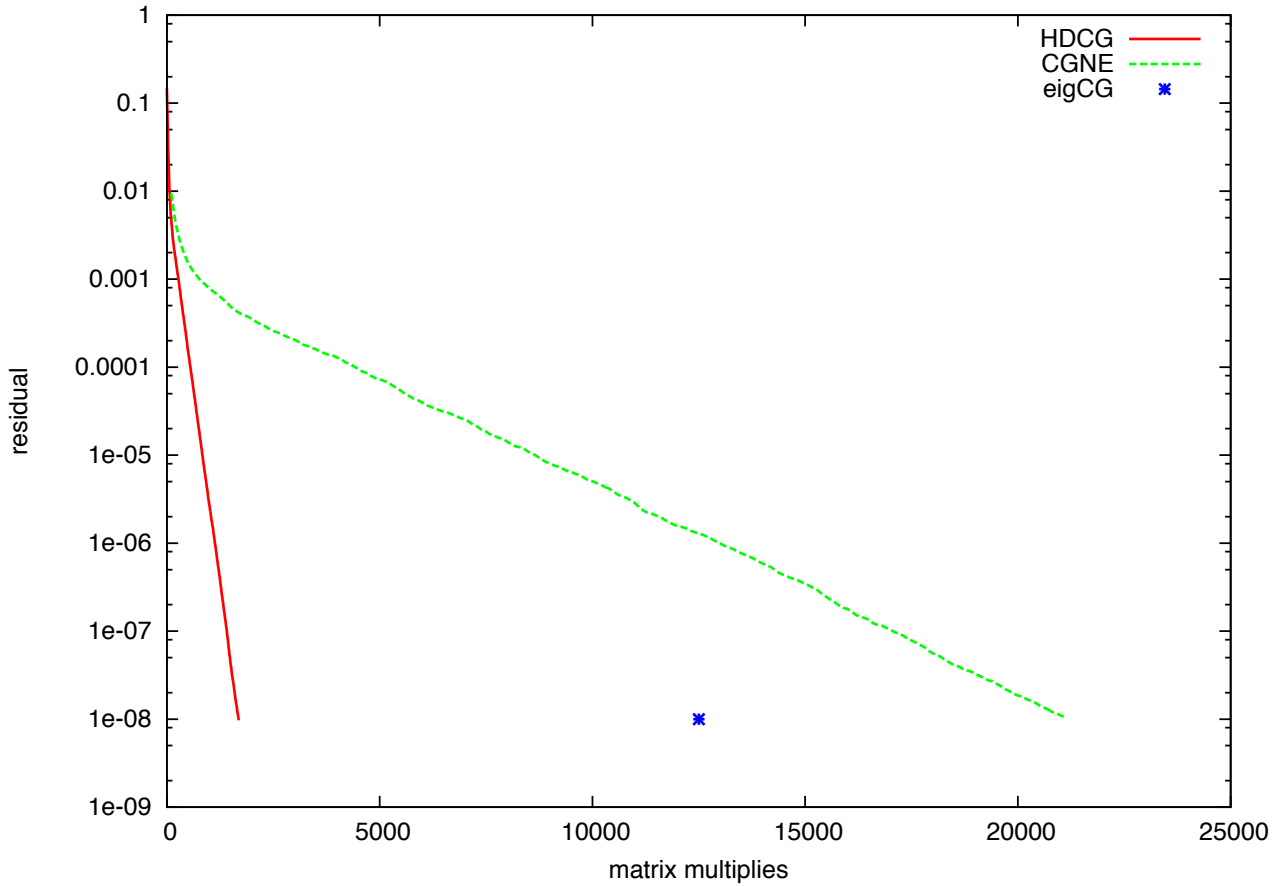
Performance on BG/Q [3]

Adaptive Smooth Aggregation Algebraic Multigrid

"Adaptive multigrid algorithm for the lattice Wilson-Dirac operator" R. Babich, J. Brannick, R. C. Brower, M. A. Clark, T. Manteuffel, S. McCormick, J. C. Osborn, and C. Rebbi, PRL. (2010).

BFM multigrid sector

- Newly developed (PAB) multigrid deflation algorithm gives 12x algorithm speedup after training
- Smoother uses a Chebyshev polynomial preconditioner
- *can project comms buffers in the polyprec to 8 bits without loss of convergence!*



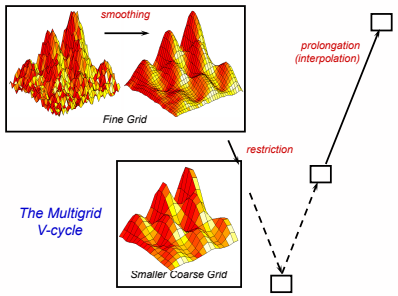
Multigrid for DW from Peter Boyle

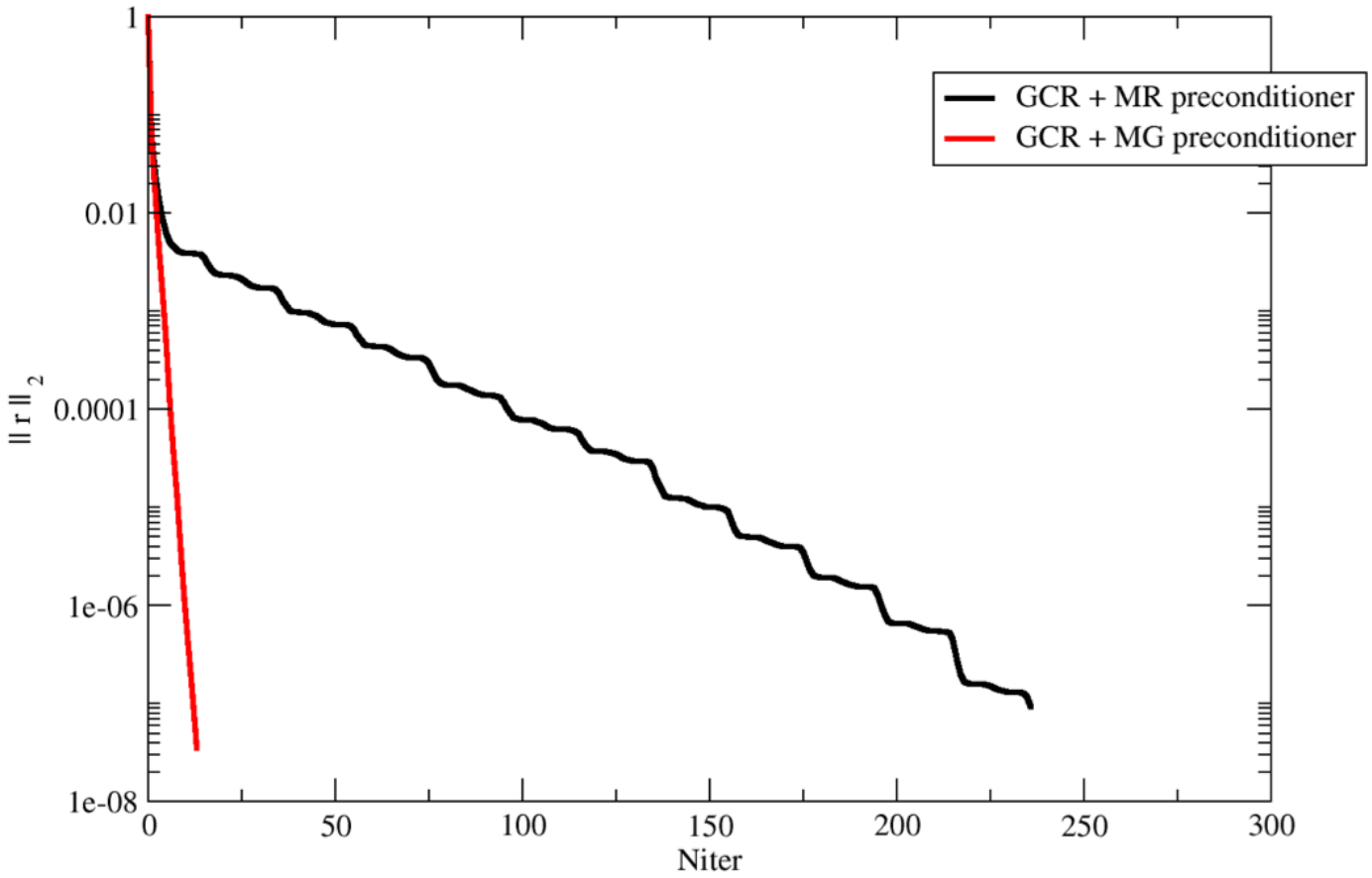
Wilson-clover:Multigrid on multi-GPU (then Phi)

Problem: Wilson MG for Light Quark beats QUDA CG solver GPUs!

Solution: Must put MG on GPU of course

+ =>

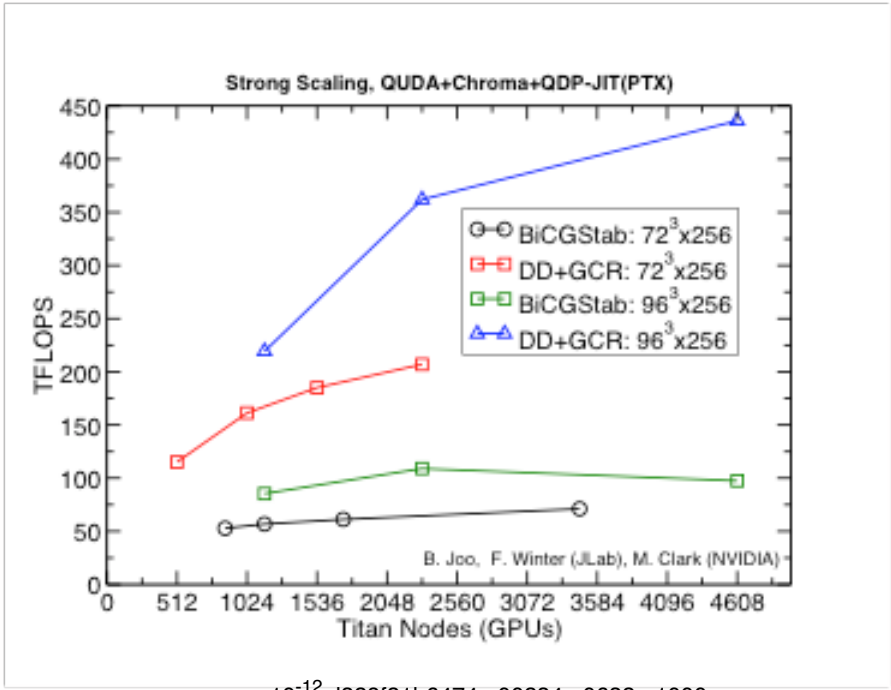


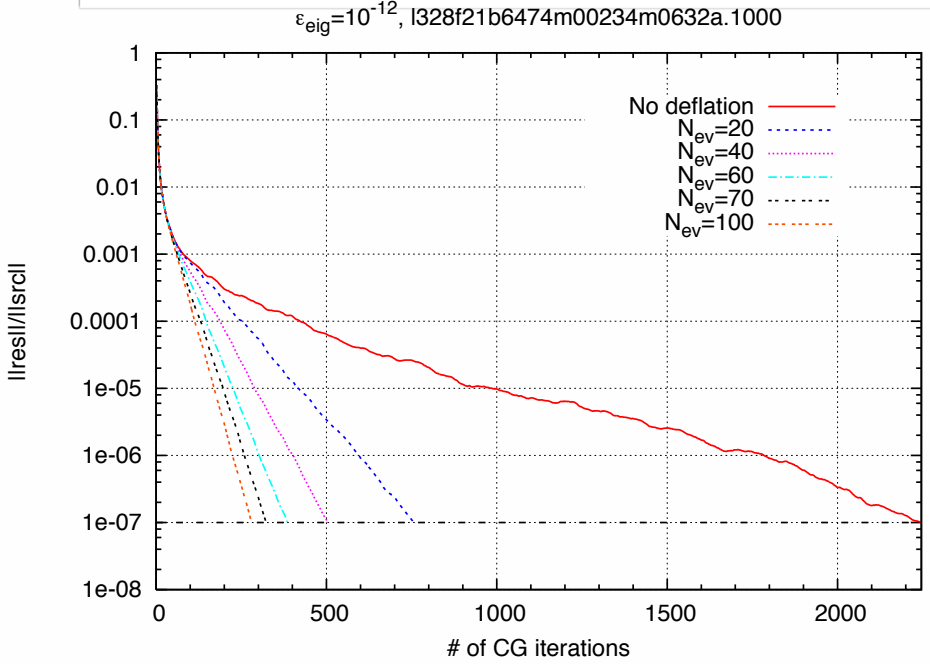


GPU + MG will reduce $ cost by O(100) : see Rich Brower Michael Cheng and Mike Clark, Lattice 2014

Domain Decomposition & Deflation

- DD+GCR solver in QUDA
  - GCR solver with Additive Schwarz domain decomposed preconditioner
  - no communications in preconditioner
  - extensive use of 16-bit precision
- 2011: 256 GPUs on Edge cluster
- 2012: 768 GPUs on TitanDev
- 2013: On BlueWaters
  - ran on up to 2304 nodes (24 cabinets)
  - FLOPs scaling up to 1152 nodes
- Titan results: work in progress





A Few "Back End" Slides

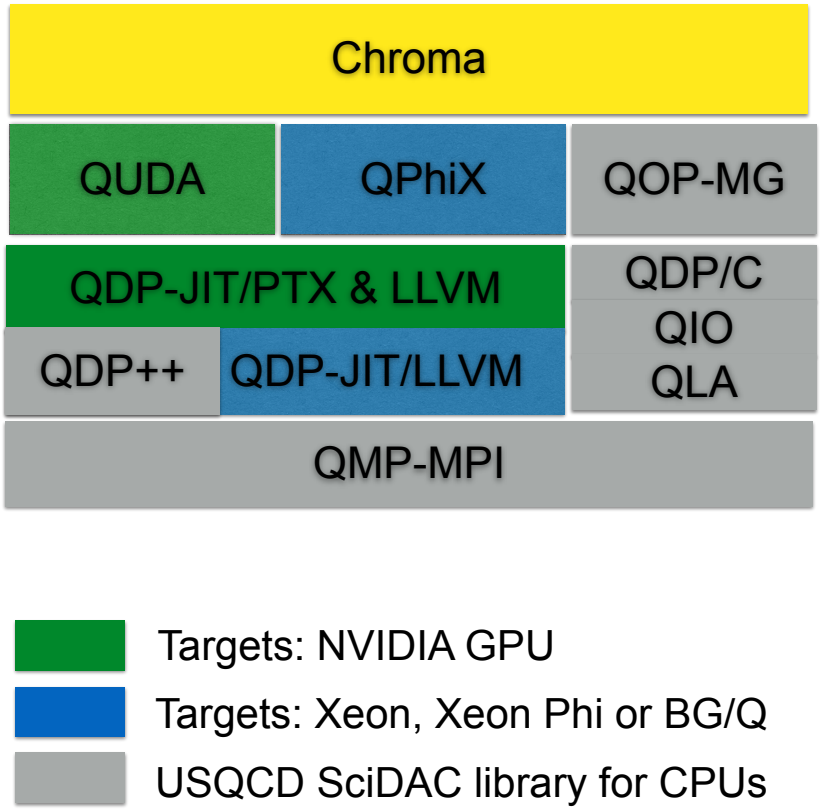
- New Data Parallel Foundation

MPI + OpenMP4 for PHI and GPUs?
+ Level 3 QUDA/QphiX Libraries?

Jlab: QCD-JIT Method

Software: Gauge Gen. & Propagators

- **Chroma**: application to do gauge generation and propagator inversions
- **QUDA**: GPU QCD Component (solvers) Library
- **QPhiX**: Xeon Phi, Xeon Solver Library
- **QDP++:** Data parallel productivity layer on which Chroma is based
- **QDP-JIT/PTX:** Reimpementation of QDP++ using JIT compilation of expression templates for GPUs
- **QDP-JIT/LLVM:** QDP-JIT but generating code via LLVM JIT framework
- **QOP-MG:** Multi-Grid solver based on QDP/C stack
- **QMP-MPI:** QCD message passing layer over MPI



Peter Boyle's GRID

See https://github.com/paboyle/Grid

Grid

Data parallel C++ mathematical object library

This library provides data parallel C++ container classes with internal memory layout that is transformed to map efficiently to SIMD architectures. CSHIFT facilities are provided, similar to HPF and cmfortran, and user control is given over the mapping of array indices to both MPI tasks and SIMD processing elements.

- Identically shaped arrays then be processed with perfect data parallelisation.
- Such identically shapped arrays are called conformable arrays.

The transformation is based on the observation that Cartesian array processing involves identical processing to be performed on different regions of the Cartesian array.

The library will both geometrically decompose into MPI tasks and across SIMD lanes. Local vector loops are parallelised with OpenMP pragmas.

Data parallel array operations can then be specified with a SINGLE data parallel paradigm, but optimally use MPI, OpenMP and SIMD parallelism under the hood. This is a significant simplification for most programmers.

see OpenMP4 http://openmp.org/wp/openmp-specifications/

QDP/C & QOPDP replacement (Osborn)

* starting with low level code that is fully vectorized and threaded

* using QMP and OpenMP

* have Asqtad solver and link fattening running, working on fermion force

* have plugged solver and fattening into QOPQDP so it can use the new code without need to change application that is already using QOPQDP

* there is some conversion overhead, will eventually drop existing QOPQDP interface and start calling new code directly

* long term goal is to develop high level code generator that generates the low level code, and work on targeting CPU and GPU architectures

Below are benchmark results in Gflops/node (1 BG/Q node = 204.8 Gflops peak) for the single mass solver. "L" is the effective local box size = V^0.25. It performs much better than the old code when the problem fits in cache. In single precision it gets up to 23% of peak (the old code got up to 13%). When it starts spilling to memory, the performance drops quite a bit, though it is still a bit faster than the old code. I think I can still delay the onset of this spilling to memory some (by a factor of 2 in volume).

512 nodes BG/Q single precision

L	old	new
8.49	11.6	27.7
10.09	21.5	37.5
11.31	24.2	43.6
12	25.7	46.5
13.45	20.2	21.3
16	16.3	19.8

512 nodes BG/Q double precision

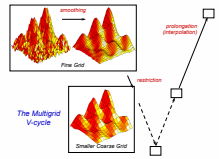
L	old	new
8.49	9.4	22.0
10.09	12.4	28.6
11.31	9.8	11.1
12	9.0	10.4

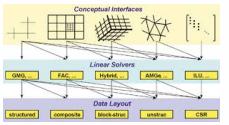
Qlua++ Agenda (MIT)

- Future generations of HPC hardware will be different from what the USQCD software stack was designed for. In the future we will face fat nodes with many cores and multitiered memory, and slow networks with low bandwidth and high latency per Flops.
- LQCD applications continues to evolve to complex bodies of codes with many non-obvious opportunities for parallelism. We can expect the epoch of yakuza programming to be close to its end. A systematic way to provide high performance and scalability of all LQCD codes is needed.
- For the front-end, the data parallel programming model is still useful.
- Back-ends need to be able (a) to rely on fixed semantics of the front-end, and (b) to be free to exploit available hardware.
- We need to pay close attention to memory management, out of order execution, just in time compilation, and other software techniques to exploit HPC hardware efficiently.
- A serious look at existing stable standards is warranted. MPI and HDF5 are examples of mature technologies that the LQCD community could benefit from.

FASTMath: Qlua+HYPRE

- QCD/Applied Math collaboration has long history: 8 QCDNA (Numerical Analysis) Workshops 1995-2014.
- Fast development framework is being constructed based on the combined strength of the FASTMath's HYPRE library at LLNL and the Qlua software at MIT.
- HYPER enhanced: Complex arithmetic and 4d and 5d hyper-cubic lattices.
- Qlua to HYPRE interface: to important Dirac Linear operators.
- Qlua is enhanced: 4d and 5d MG blocking and general "color" operators
- HYPRE: exploration of bootstrap algebraic multigrid (BAMG) algorithm for Dirac
- Goal to explore multi-scale for Wilson, Staggered and Dirac operators
- Test HYPRE methods at scale in Qlua and port into QUDA and QphiX libraries.





QUDA

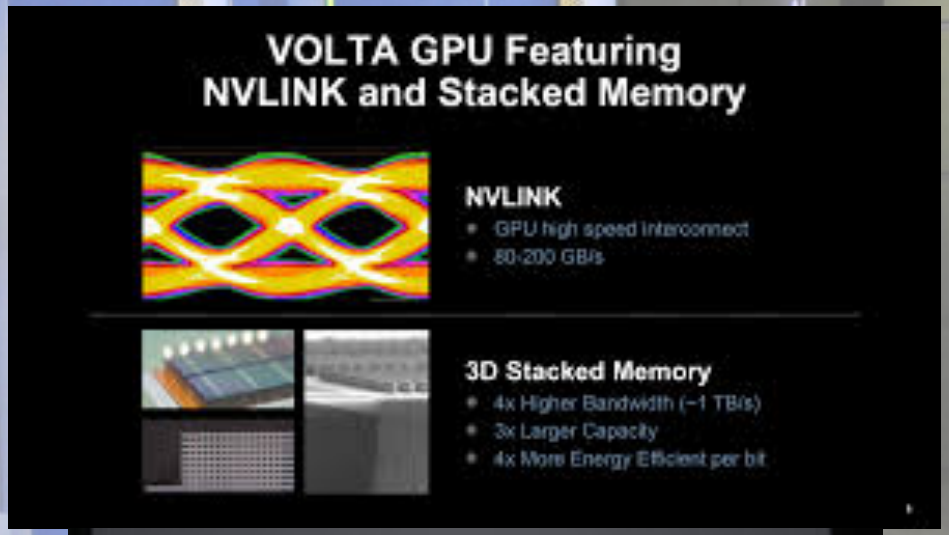
QphiX

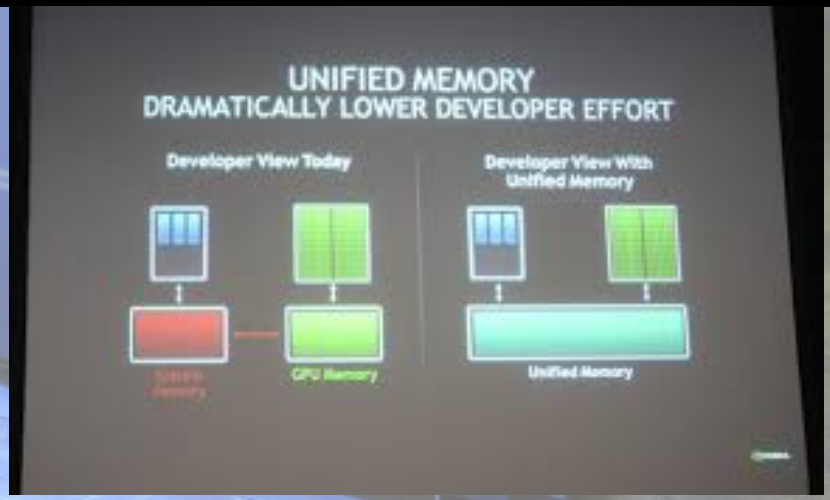
FUTURE: CORAL

OAK RIDGE National Laboratory

Argonne NATIONAL LABORATORY

Lawrence Livermore National Laboratory





HISTORY: WHERE AM I ?

(5+ YEARS AFTER BIRTH OF LATTICE QCD)

Courtesy Special Collections, UC Santa Cruz