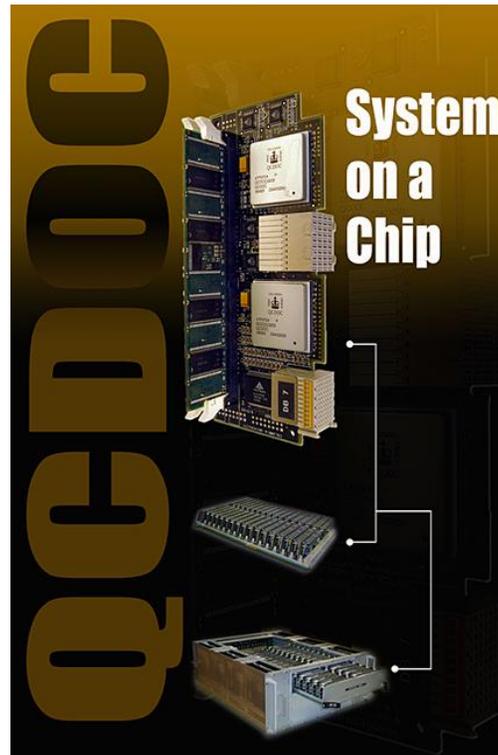


QCDOC Project, Lattice 2004



QCDOC Collaboration

Columbia: N. Christ, C. Cristian, C. Kim, L. Levkova
X. Liao, G. Liu, S. Li, H.W. Lin,
R.D. Mawhinney, A. Yamaguchi

UKQCD: P. Boyle, B. Joo

Riken-Brookhaven: S. Ohta (KEK), T. Wettig (Regensburg)

SciDac: C. Jung, K. Petrov

IBM Research: A. Gara, B. Nathanson, M. Liu



Introduction

Full QCD with chiral Dirac operator and light quark masses

Decorrelation in Monte-Carlo time evades trivial parallelisation

Optimise Dirac applications per second, not just Teraflop/s.

- Apply many FPU's in parallel to QCD
- Challenge is to provide the data fast enough
- Cost \Rightarrow many microprocessors, independent dram systems
- Off-node data references require message passing
- Both DRAM *and* interconnect constitute memory system

Very large machines \Rightarrow small local sub-lattice

Off-node data references common and very frequent

Interconnect as aggressive as a local DRAM system

Problem characteristics

Krylov solvers dominate: Dirac matrix multiply + global summation

Dirac application:

- Predictable memory access patterns
- Communications are nearest neighbour on Mesh Network
Dirac application scales trivially at fixed local sublattice
- Communications repetitive
- Communicate in multiple directions simultaneously

Exploit simplifications: special purpose machines



QCDOC Arcitecture

- 6 dimensional torus
- Allows partitioning
- QCD maximally spread out in four/five dimensions
- Up to 20k nodes \Rightarrow small subvolume per node
- fast on-chip memory
- high performance nearest neighbour communication
- Frequent global summation : Hardware assist
- Communication performance \simeq memory performance

Serial Communications Unit

- Communication is over the 2.6GB/s on chip bus
Communication hardware at same level as memory system
- Block-strided DMA operations
- 16 deep DMA instruction memory for each wire
Exploit repetitive nature of QCD codes
- **Low DMA latency $O(550)ns$**
- Single operation can start all wires
Effective latency $\simeq \frac{550}{8}ns$
- 50 MB/s per link \times 24 links
- 1.2GB/s aggregate
- **Concurrently runs all links efficiently**
- Programmable store and forward routes
- **Hardware** assist for **global summation** and broadcast



Prefetching Edram Controller

- Multiple ports
- Efficient **concurrent access** for Comms and Compute
- Processor Data Bus (PDB)
Peak 8GB/s Read + 8GB/s Write
- Processor local bus (PLB)
Communications access (2.6GB/s Read + 2.6GB/s Write)
- DDR ↔ Edram DMA 2.6GB/s
- Each port linearly **prefetches two streams**
- Double write buffers enable **efficient non-linear writes**
- Scatter dont Gather!!!

```
Psi [ shift[i] ] = U[i] * Chi[i];
```

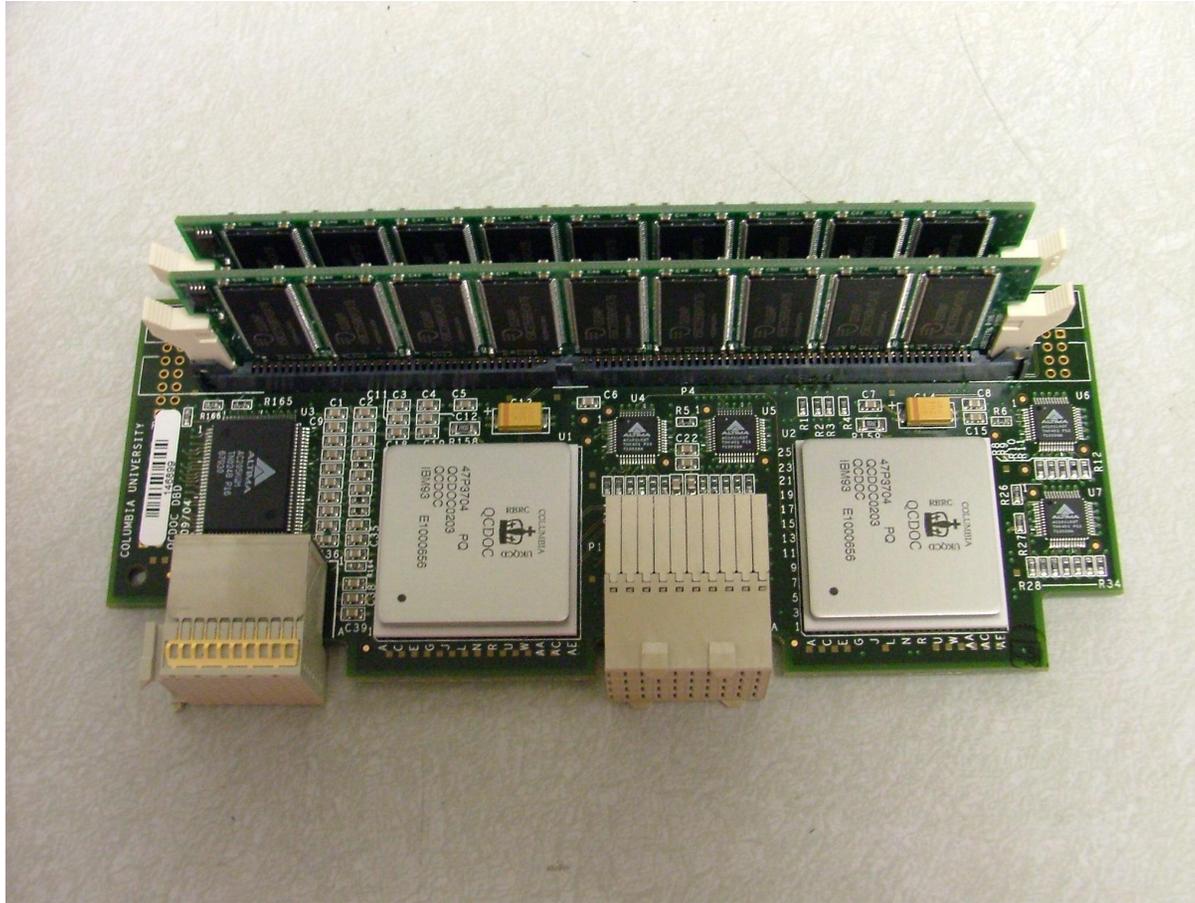


QCDOC Asic, 1 Gflop/s



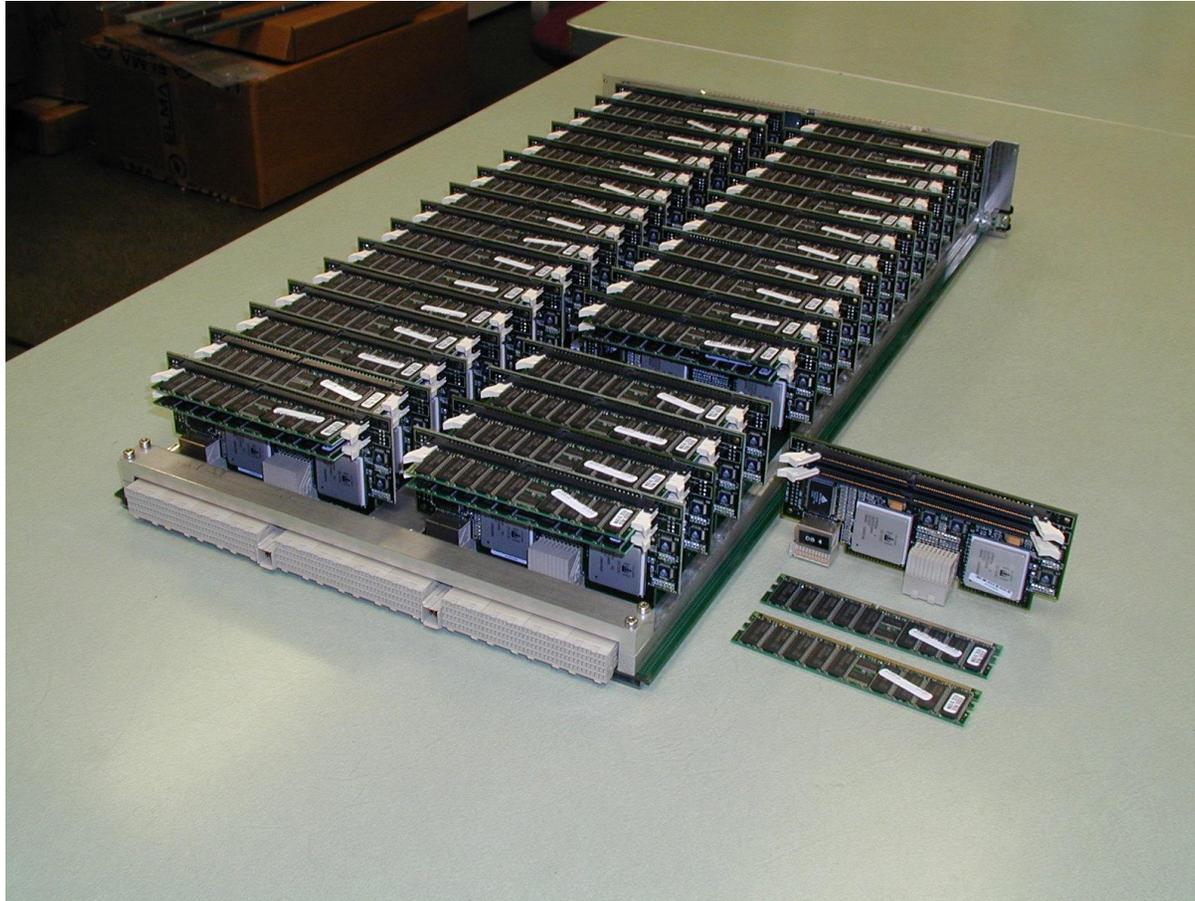
QCDOC Daughterboard 2 Gflop/s

Two independent compute nodes + Ethernet system

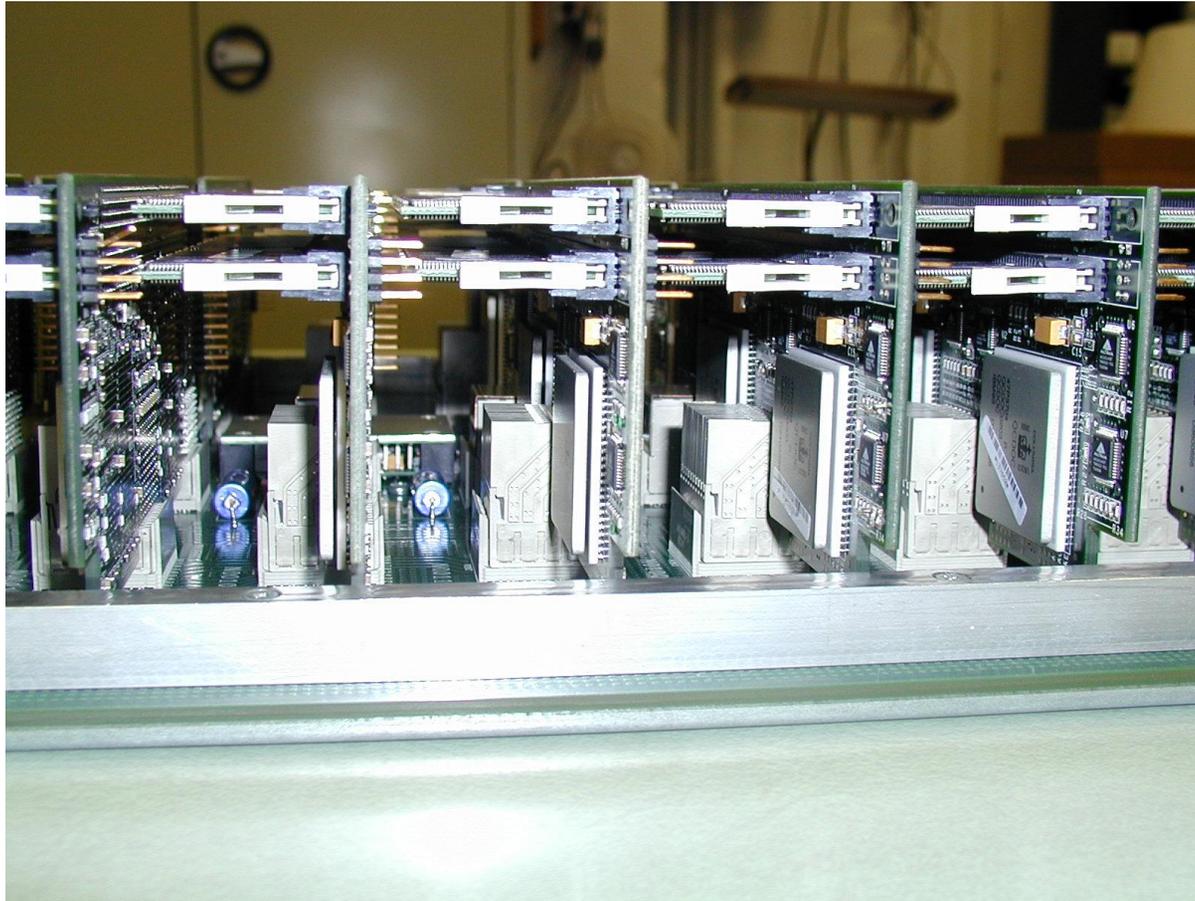


QCDOC Motherboard

64 Gflop/s, 32 Daughterboards, 64 nodes, very dense
768 Gigabit/s internode, 800 Mbit/s Ethernet.



QCDOC Motherboard



Single slot backplane 128 node system
 2×64 Gflop/s



Air Cooled Crate, 0.5 Tflop/s



Air cooled crate cabling



Water cooled crate, 1 TFlop/s, 1024 Nodes





Timeline

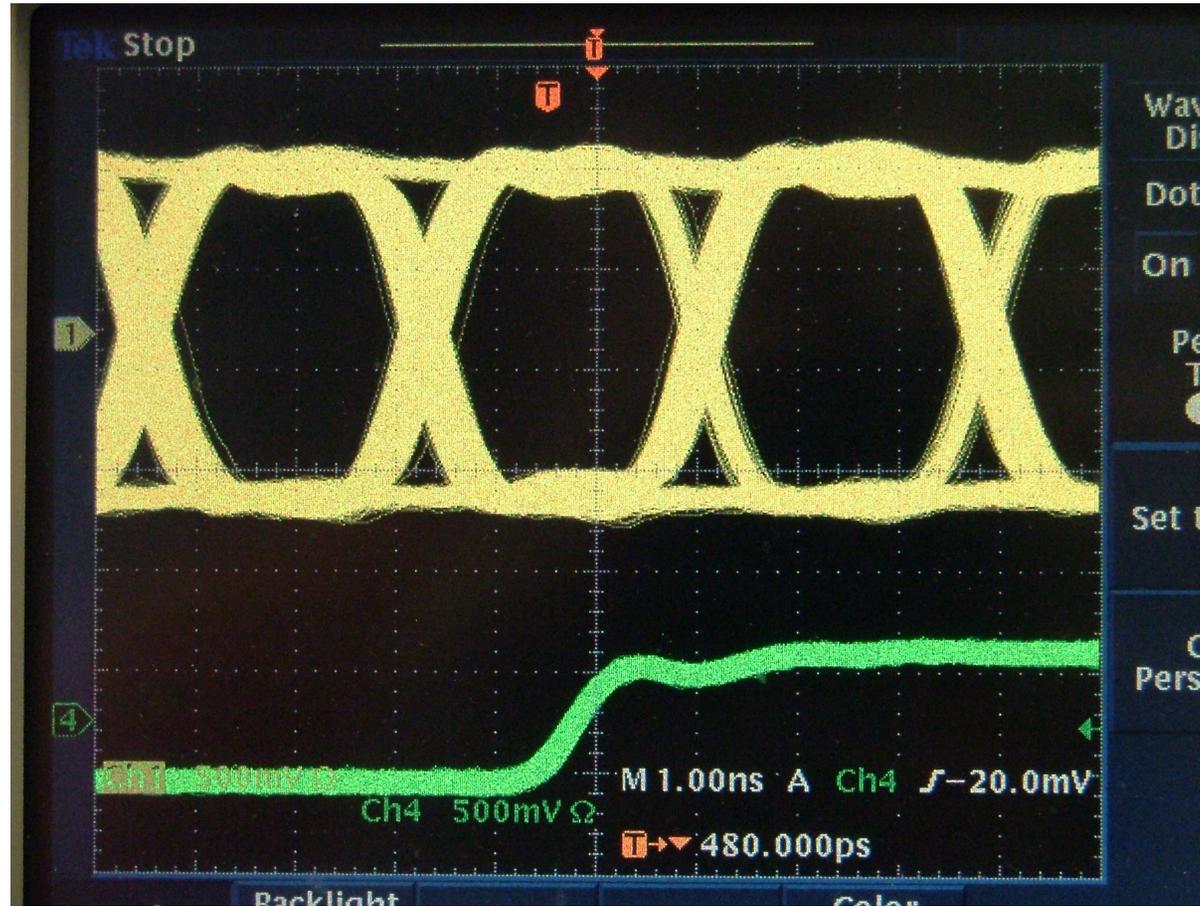
- September 2000 - March 2003 design & verification effort
- Asic tape out: 09:48am, April 8, 2003
- July 2003 single Daughterboard test-jigs
- September 2003 motherboards & single slot backplanes
- November 2003 SSBP's debugged
- November 2003 128 node machine debugged
- April 2004 512 node air cooled crate
- June 2004 512 node debugged
- June 2004 1024 node Water cooled cabinet
- July 2004 2048 node machine
- September 2004 12k node UKQCD machine
- September 2004 12k node RBRC machine
- March 2005 12k node DOE machine

All this works!

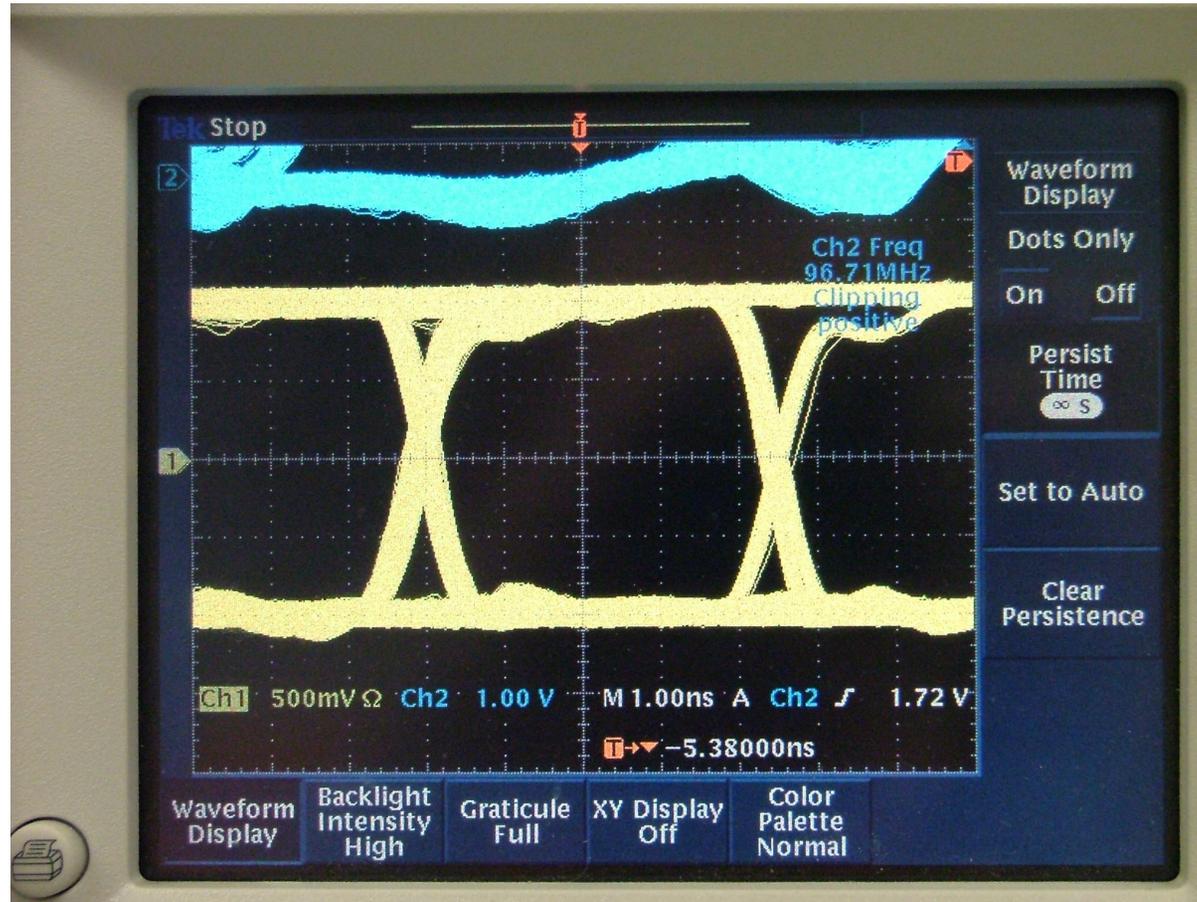
Signals & Data Eyes

- Sample signals for long time period
- Superimpose all transitions
- Jitter, inter-symbol interference, reflections broaden the pattern.
- Diagnostic for high speed signals
- 500Mbit SCU, 333MHz DDR, 125Mbit Ethernet

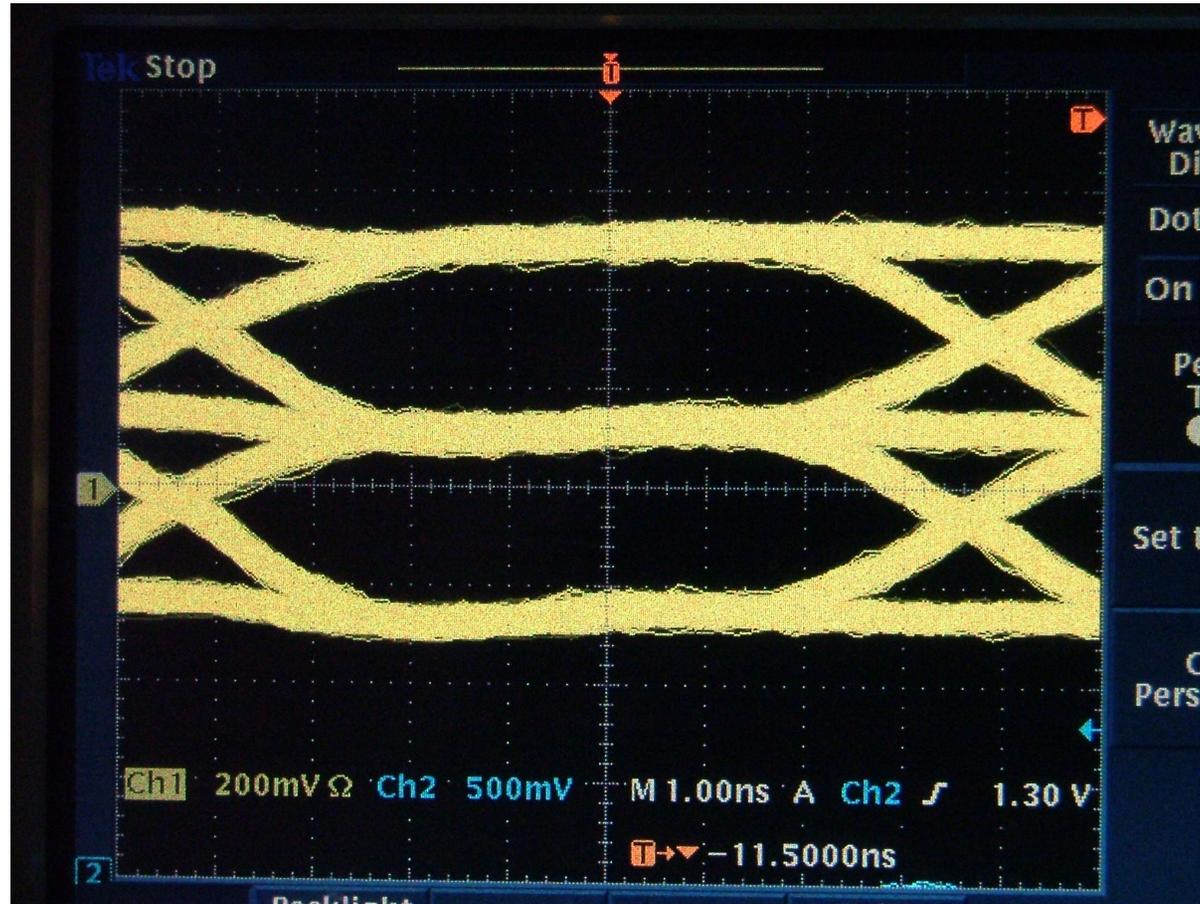
Serial Communications Data Eye



DDR memory data eye (write path)



Ethernet Data Eye (three level signal)



Testing status

- **No known bugs** on QCDOC Asic to date.
- **Used remotely** by SciDac/UKQCD experts (victims?) since November 2003

Date	Nodes	Frequency	Test
11/2003	64	450	Wilson Spectrum 10 Configs
11/2003	128	450	Wilson HMC 1000 traj, 2 days
11/2003	64	450	Wilson HMC 1000 traj, 5 days
11/2003	64	450	Naive Staggered Evolution
6/2004	64	360	AsqTad Evolution, big memory
6/2004	64	360	RHMC AsqTad
6/2004	64	360	File system stress tests
6/2004	512	360	Wilson HMC 1000 × 2 traj, 8h
6/2004	512	360	Clover HMC, 52h

Clock frequency reduced to aid hardware debug
 Working to increase clock frequency (unbuffered DIMM timing problematic).
 Many subtle problems have been fixed already, including

- Unbuffered dimms at 360MHz

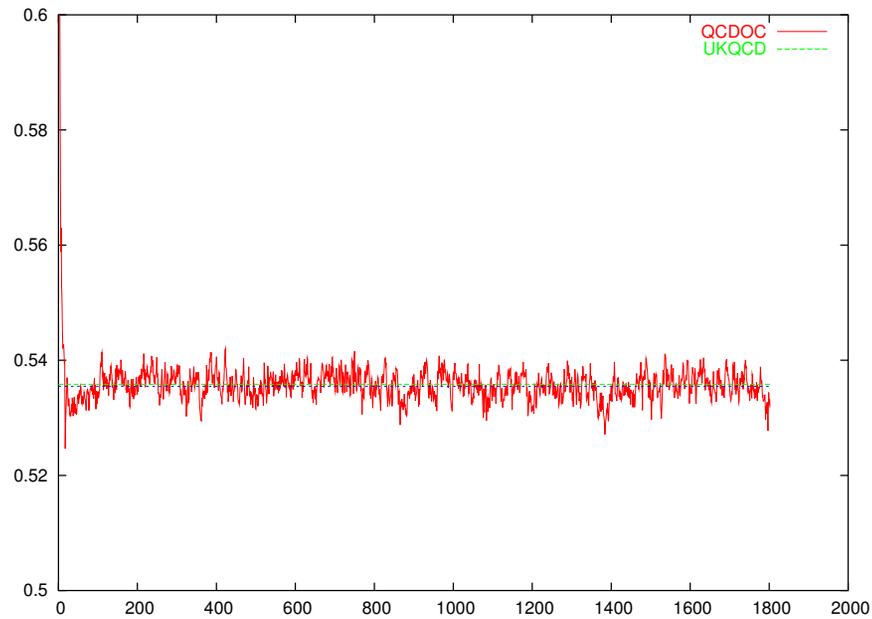


Dynamical clover plaquette log (8 hours)

$$\beta = 5.2, C_{SW} = 2.0171, \kappa = 0.1345$$

$$\text{UKQCD: Plaquette} = 0.53565(18)$$

$$\text{QCDOC: Plaquette} = 0.53569(17)$$



Status Summary

- No known QCDOC Asic bugs.
- Single motherboards have been stable since November 2003
- Small scale community service since October 2003
- 512 node machine stable since 12 June 2003
- 512 node Clover HMC achieves 47% peak, 64 bit arithmetic
- 1024 node machine assembled, under shakeout
- Large machines in September
- Clock speed still not known (420-450MHz best estimate)

Normal people can use it!

Programming environment

- Two standard “C” and “C++” compile tools: **GCC, XLC**
- POSIX libc on a custom kernel
- **Standard** “Unix-like” **environment** on custom O/S
- NO process control and inter-process communication.
- **File I/O fully supported from every node**
- Library for internode message passing (**QMP**, SCU)
- Memory allocation extensions for Edram
- **Standard compliant**: benefits everyone
documentation is *already* written & code is portable



Many code bases have been run on QCDOC

- Columbia Physics System
- MILC
- Chroma/QDP++

Optimised code is available and used:

Kernels	Codes
BAGEL Wilson, Clover, DWF	CPS, Chroma/QDP++
AsqTad assembler	CPS, MILC
BAGEL Linalg	CPS, Chroma/QDP++



Performance of QCDOC – April 29,2004

Results are shown as a percentage of peak speed. The Wilson and ASQTAD optimized results (presented at the March 2004 All Hands Meeting at BNL) were run at 450Mhz (900 MFlops/node peak speed). All other results from 128 node machines were run at 420 MHz on newly built QCDOC nodes, which are not yet stable at 450 MHz. These benchmarks represent our current status. They should be regarded as preliminary, since we expect further improvements in performance.

	Sites/node	Optimized Code			MILC Code			MILC Code			MILC/QDP
Precision		Double			Single			Double			Single
Application		D	D	CG	CG			CG			CG
Machine size		Sim	128	128	128			16			16
Memory used					DDR	CG temps EDRAM	CG temps+lat EDRAM ¹	DDR	CG temps EDRAM ¹	CG temps+lat EDRAM ¹	
Wilson ⁴	2 ⁴	47%	44%	32%							
	4 ⁴		43%	38%							
	6 ⁴		44%	39%							
	8 ⁴		29%								
ASQTAD ²	2 ⁴		22%	19%							2%
	4 ⁴	43%	42%	40%	7.6%	8.5%	9.5%	5.9%	6.5%	7.5%	15%
	6 ⁴		35%	33%	9.9%	11.5%	13.6%	7.6%	8.6%		19%
	8 ⁴		29%	28%	10.5%	12.3%		8.1%			
Clover ⁴	2 ⁴			31%							
	4 ⁴			47%							
DWF ⁴	2 ⁴ × 4			32%							
	4 ⁴ × 4			42%							
Application		ASQTAD force			ASQTAD force ³			ASQTAD force ³			
Machine size		Sim	128	128	128			16			
Memory used					DDR	temps EDRAM	temps+lat EDRAM ¹	DDR	temps EDRAM ¹	temps+lat EDRAM ¹	
ASQTAD ²	4 ⁴				7.8%		10.0%	5.5%	5.5%	6.1%	
	6 ⁴				8.2%		10.8%	6.1%	6.1%		
	8 ⁴				8.2%			6.0%			

1. Current MILC ASQTAD memory allocation code does not allow single precision, 8⁴ runs with the lattice and all temporaries in the on-chip memory (EDRAM). For double precision, the limit appears for 6⁴ lattices. No performance is quoted for these cases. Modifying the code could allow some use of the EDRAM.
2. For the MILC ASQTAD code, XLC has been used for the CG results. It offers a 10% speed-up relative to gcc, which is about 1% of peak here. Larger differences between XLC and gcc for MILC that were seen earlier appear to have been due to inadvertently toggling the “native double flag” of the MILC code when switching between the gcc and XLC compilers. With the native double flag off, neither gcc nor XLC compiles MILC code well.
3. For the MILC ASQTAD force term, the native double flag has no effect. It appears this part of MILC code was never optimized for this flag. The MILC ASQTAD force term code does use inlining functionality provided by Carleton DeTar. The performance is quoted for the contribution of the fermions to the force term for the gauge fields.
4. For many volumes, the optimized Wilson, Clover and DWF CG results are expected to increase by up to 5% of peak by further eliminating pipeline stalls in the Wilson kernels and improving receive buffer handling. In addition, for Clover fermions increased overlap of communications and computation is possible. For DWF, the 5-d part of the operator is not optimized on all volumes.

QCDOC Software

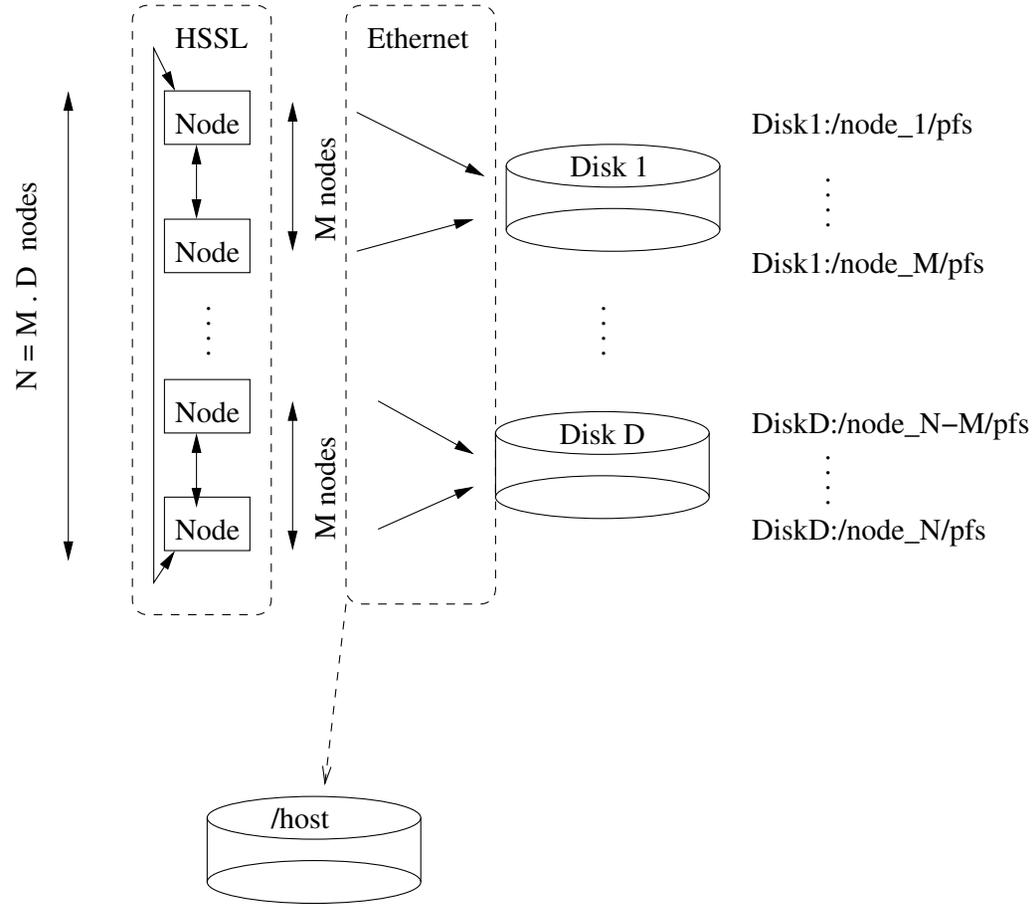
Boot kernel

- JTAG download of boot kernel to I-cache & MMU, memory bring up.
- Tests & Diagnoses hardware
- Boot kernel used to “bootstrap” run kernel into memory via Ethernet

Run kernel

- Custom interrupt stack and Ethernet driver
- RPC Server runs as kernel process
- Runs **application process** without task switching
no scheduling → no waiting
- **Services** Comms and I/O **system calls**
- **Memory protection** but not translation
zero copy DMA on simple hardware
eliminates TLB miss **overhead**.
- Traps and **reports** many hardware and programming **errors**
- Custom NFS client

NAS-RAID based parallel disk system



Host Operating System



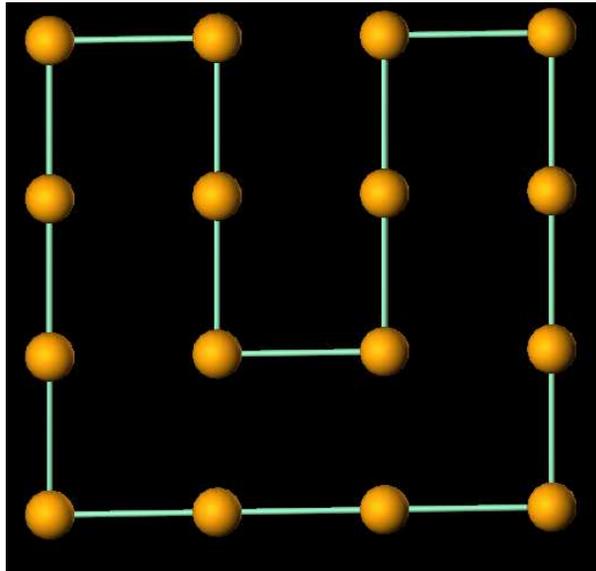
Qdaemon and qcsh.

- Bandwidth: **multi-GigE** from SMP host.
- \Rightarrow Agressively multi-threaded qdaemon
- **Reliable UDP** communication
many I/O threads. retransmit, acks etc...
- Threads for each user and each partition
- **Relies on UNIX** protection **mechanisms**.

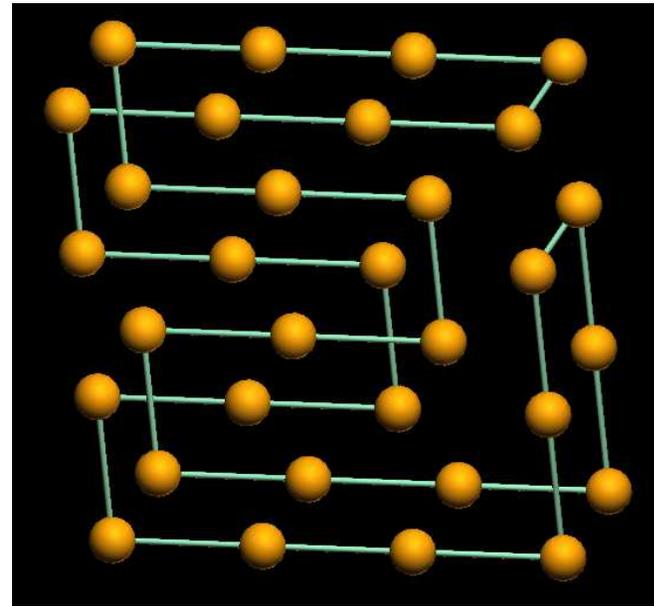
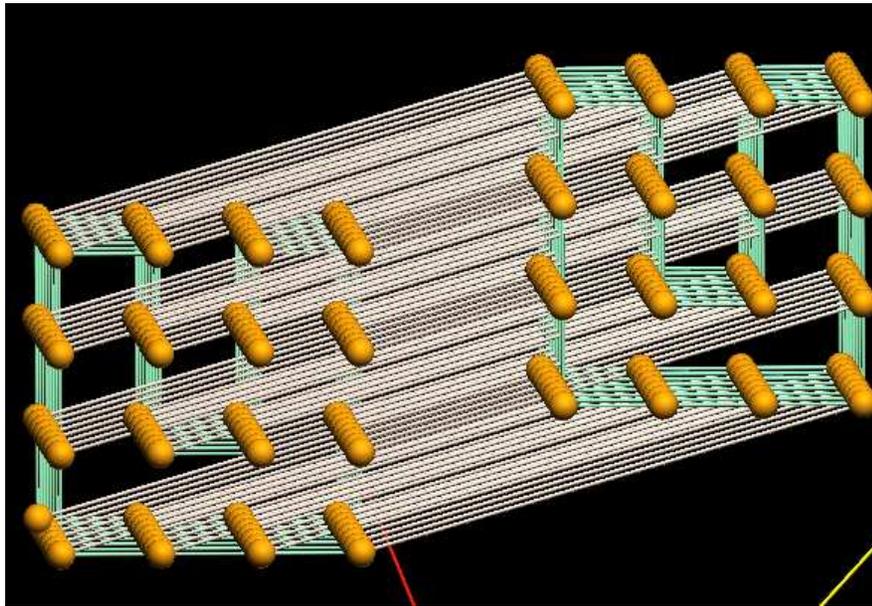
Partitioning

- Machine is a $2k \times 2l \times 2m \times 2 \times 2 \times 2$ hyper-torus.
- Partitions are 6d hyper-rectangular slices of machine
- Locally redefine Application to Machine axis map
- Present 1,2,3,4,5 or 6 dim torus to applications
- Details hidden from user

Remapped slice of machine: lower dimensional torus:



- Fold any two dimensions independent of others
- Can iterate to fold multiple dimensions together



Summary

- 512 node machine running (+ 256 nodes in smaller machines)
- 1024 node machine assembled and in shakeout
- Large machines this fall
- Very efficient, very scalable

Future

Industry trends are playing in our favour

- Move to high speed serial communications
RapidIO, Serial-ATA, PCI-Express
Reuse logical protocols?
- Move integrated memory controllers/SoC
- Move to many cores/FPU's on a single chip
- Refocussing on lower power (Pentium-4 Prescott farce)

Special purpose machine integration will retain density and cost advantages.

Special purpose machines will retain lower latency communications through and careful hardware design and lean software.

Can we reduce the development time?