

# QDP++ and Chroma

Robert Edwards  
Jefferson Lab

<http://www.lqcd.org>

<http://www.jlab.org/~edwards/qdp>

<http://www.jlab.org/~edwards/chroma>

# SciDAC Software Structure

Optimised Wilson Op for P4, Wilson and Staggered Ops for QCDOC

Level 3

Optimised Dirac Operators,  
Inverters

Level 2  
Exists in C,  
C++, scalar and  
MPP using  
QMP

Level 1

QDP (QCD Data Parallel)

Lattice Wide Operations,  
Data shifts

QIO

XML I/O  
LIME

QLA (QCD Linear Algebra)

QMP (QCD Message Passing)

exists, implemented in MPI, GM, gigE and  
QCDOC

# Data Parallel QDP C/C++ API

- Hides architecture and layout
- Operates on lattice fields across sites
- Linear algebra tailored for QCD
- Shifts and permutation maps across sites
- Reductions
- Subsets
- Entry/exit – attach to existing codes
- Implements SciDAC level 2.

# Data-parallel Operations

- *Unary and binary:*  
-a; a-b; ...
- *Unary functions:*  
 $\text{adj}(a)$ ,  $\cos(a)$ ,  $\sin(a)$ , ...
- *Random numbers:*  
// platform independent  
 $\text{random}(a)$ ,  $\text{gaussian}(a)$

- *Comparisons (booleans)*  
 $a \leq b$ , ...
- *Broadcasts:*  
 $a = 0$ , ...
- *Reductions:*  
 $\text{sum}(a)$ , ...

# QDP++ Type Structure

- Lattice Fields have various kinds of indices
  - Color:  $U^{ab}(x)$  Spin:  $\Gamma_{\alpha\beta}$  Mixed:  $\psi_\alpha^a(x), Q^{ab}_{\alpha\beta}(x)$
- Tensor Product of Indices forms Type:

	<i>Lattice</i>	<i>Color</i>	<i>Spin</i>	<i>Complexity</i>			
Gauge Fields:	Lattice	$\otimes$	Matrix( $N_c$ )	$\otimes$	Scalar	$\otimes$	Complex
Fermions:	Lattice	$\otimes$	Vector( $N_c$ )	$\otimes$	Vector( $N_s$ )	$\otimes$	Complex
Scalars:	Scalar	$\otimes$	Scalar	$\otimes$	Scalar	$\otimes$	Scalar
Propagators:	Lattice	$\otimes$	Matrix( $N_c$ )	$\otimes$	Matrix( $N_s$ )	$\otimes$	Complex
Gamma:	Scalar	$\otimes$	Scalar	$\otimes$	Matrix( $N_s$ )	$\otimes$	Complex

- QDP++ forms these types via nested C++ templating
- Formation of new types (eg: half fermion) possible

# QDP++ Expressions

- Can form expressions:

$$c_\alpha^i(x) = U_\mu^{ab}(x+\nu) b_\alpha^i(x) + 2 d_\alpha^i(x) \text{ for all } x$$

- QDP++ code:

```
multild<LatticeColorMatrix> U(Nd);  
LatticeFermion c, b, d;  
int nu, mu;  
  
c = shift(u[mu], FORWARD, nu)*b + 2*d;
```

- PETE- Portable Expression Template Engine
  - Temporaries eliminated, expressions optimised

# Linear Algebra Example

```
// Lattice operation  
A = adj(B) + 2 * C;
```

```
// Lattice temporaries  
t1 = 2 * C;  
t2 = adj(B);  
t3 = t2 + t1;  
A = t3;
```

```
// Merged Lattice loop  
for (i = ... ; ... ; ...) {  
    A[i] = adj(B[i]) + 2 * C[i];  
}
```

- Naïve ops involve lattice temps – inefficient
  - Eliminate lattice temps - PETE
  - Allows further combining of operations ( $adj(x)*y$ )
  - Overlap communications/computations
- 
- Remaining performance limitations:
    - Still have site temps
    - Copies through “=”
  - Full perf. – expressions at site level

# QDP++ Optimization

- Optimizations “under the hood”
  - Select numerically intensive operations through template specialization.
  - PETE recognises expression templates like:

$$z = a * x + y$$

from type information at compile time.

- Calls machine specific optimised routine (axpyz)
- Optimized routine can use assembler, reorganize loops etc.
- Optimized routines can be selected at configuration time,
- Unoptimized fallback routines exist for portability

# Chroma: A lattice QCD Library using QDP++ and QMP. Work in development

- A lattice QCD toolkit/library built on top of QDP++
- Library is a module – can be linked with other codes.
- Features:
  - Utility libraries (gluonic measure, smearing, etc.)
  - Fermion support (DWF, Overlap, Wilson, Asqtad)
  - Applications
    - Spectroscopy, Props & 3-pt funcs, eigenvalues
    - Not finished – heatbath, HMC
  - Optimization hooks – level 3 Wilson-Dslash for Pentium and now QCDOC
- Large commitment from UKQCD!

eg: McNeile: computes propagators with CPS, measure pions with Chroma all in same code

```

LatticeFermion psi, p, r;
Real c, cp, a, d;
for(int k = 1; k <= MaxCG; ++k)
{
    // c = | r[k-1] |**2
    c = cp;

    // a[k] := | r[k-1] |**2 / <M p[k], Mp[k] > ;
    // Mp = M(u) * p
    M(mp, p, PLUS); // Dslash

    // d = | mp | ** 2
    d = norm2(mp, s); ←

    a = c / d;

    // Psi[k] += a[k] p[k]
    psi[s] += a * p; ←

    // r[k] -= a[k] M^dag.M.p[k] ;
    M(mmp, mp, MINUS);
    r[s] -= a * mmp; ←

    cp = norm2(r, s);
    if ( cp <= rsd_sq ) return;

    // b[k+1] := |r[k]|**2 / |r[k-1]|**2
    b = cp / c;

    // p[k+1] := r[k] + b[k+1] p[k]
    p[s] = r + b*p; ←
}

```

**Norm squares**

**VAXPY  
operations**

## Performance Test Case - Wilson Conjugate Gradient

- In C++ significant room for perf. degradation
- Performance limitations in Lin. Alg. Ops (VAXPY) and norms
- Optimization:
  - Funcs return container holding function type and operands
  - At “=”, replace expression with optimized code by template *specialization*

# QCDOC Performance Benchmarks

QCDOC	Wilson				
350Mhz, 4 nodes	Dslash Mflops	$(a + b*D)*\psi$	CG		
$2^4$	279 [38.8%]	232 [32.2%] 216 [30%]	136 [19%] 124 [17%]	[Assembly] [C++ linalg]	
$4^4$	351 [48.8%]	324 [45%] 295 [41%]	283 [39%] 236 [33%]		
$4^2 \times 8^2$	353 [49%]	323 [45%] 294 [41%]	293 [41%] 243 [34%]		

- Assembly Wilson-dslash, optimized and non-opt. vaxpy/norms
- Optimized assembler routines by P. Boyle
- Percent peak lost outside dslash – reflects all overheads
- QDP++ overhead small ( $\sim 1\%$ ) compared to best code

# Pentium over Myrinet/GigE Performance Benchmarks

	Wilson	CG	DWF	
3D mesh, 2.6Ghz 256 nodes / 8 nodes GigE      GigE	Dslash Mflops/node		Dslash	CG
4^4/node	874	495 710	765	457 626
8^4	845	720 741	676	607 625
3D mesh, 2.0Ghz 128 nodes Myrinet				
4^4	1270	673	936	582
8^4	742	620	606	531

- SSE Wilson-dslash
- Myrinet nodes, 2.0Ghz, 400 Mhz (front-side bus)
- GigE nodes, 2.6 Ghz, 533 Mhz (front-side bus) ~ 200Gflops sustained

# QDP++ Status

- Version 1
  - Scalar and parallel versions
  - Optimizations for P4 clusters, QCDOC
  - Used in production of propagators now at JLab
  - QIO (File I/O) with XML manipulation
    - Supports both switch and grid-based machines today
    - Tested on QCDOC, default version on gigE and switches
  - Adopted by and support from UKQCD
  - Single out thanks to Balint Joo and Peter Boyle for their outstanding contributions
  - High efficiency achievable on QCDOC

# Future Work

- QMP
  - Further QMP/GigE perf. improvements.
- QDP++
  - Generalize comm. structure to parallel transporters – allows multi-dir. shifts.
  - Continue leverage off optimized routines
  - Increase extent of optimizations for new physics apps
- IO
  - Move to new Metadata Standard for gauge configs
  - On-going infrastructure devel. for cataloguing/delivery
- Chroma
  - Finish HMC implementations for various fermion actions (UKQCD - overlap)