



SciDAC Software C Language Interface

Carleton DeTar
All-Hands Meeting FNAL
February 21, 2003



Software Hierarchy

Level 3: Full inverters, etc.

Level 2: QDP data parallel (C, C++)

Level 1: QLA (single node linear algebra) (C)
QMP (node-to-node messaging)

<http://www.lqcd.org>

Message Passing QMP

Philosophy:

QCD-extended subset of MPI functionality

Minimal copying / DMA where possible

Channel-oriented / asynchronous communication

Multidirectional sends/receives for QCDOC

Grid and switch model for node layout

Implemented on GM and MPI.

gigE under development

QMP Simple Example

```
char *buf;
QMP_msgmem_t mm;
QMP_msghandle_t mh;

buf = (char *)QMP_allocate_aligned_memory(size);
mm = QMP_declare_msgmem(buf, size);
mh = QMP_declare_send_to(mm, node, 0);
QMP_start(mh);

...

QMP_wait(mh);
```

Receiving node does the same steps except

```
mh = QMP_declare_receive_from(mm, node, 0);
```

Can repeat start/wait.

QMP Capabilities

- # SMP and single-processor addressing
 - # Send strided blocks
 - # Multidirectional sends
 - # Broadcasts
 - # Global sums, reductions, and max, min
 - # Barrier
-

Linear Algebra QLA

- # Scalar processor operation
 - # Operates on singles and arrays
 - # Rich indirect addressing for arrays
 - # Linear algebra for QCD ($\sim 24,000$)
 - # Random numbers
 - # Complex arithmetic macros (~ 100)
 - # Generic accessor macros for types
-

Linear Algebra example

```
QLA_ColorMatrix u[n];  
QLA_ColorVector c[n];  
QLA_ColorVector * b[n];
```

```
QLA_V_veq_M_times_pV(c,u,b,n);  
does c[i] = u[i] * (*b[i]) for i = 0..n-1
```

Data Parallel QDP/C

- # Hides architecture and layout
 - # Operates on lattice fields across sites
 - # Linear algebra tailored for QCD
 - # Shifts and permutation maps across sites
 - # Reductions
 - # Subsets
-

Lattice fields

- # Nd spacetime dimensions; D, F precision
 - # QDP_ColorMatrix ($N_c = 2, 3, N$)
 - # QDP_Integer
 - # QDP_Real
 - # QDP_Complex
 - # QDP_HalfFermion
 - # QDP_DiracFermion
 - # QDP_ColorVector
 - # QDP_DiracPropagator
 - # QDP_RandomState
-

Linear Algebra example

$$c(r) = U_x(r)b(r + \hat{x}) \quad \forall r \in \text{even}$$

```
QDP_ColorMatrix *ux = QDP_create_M();
```

```
QDP_ColorVector *b = QDP_create_V();
```

```
QDP_ColorVector *c = QDP_create_V();
```

```
QDP_V_eq_G_times_sV(c, ux, b, QDP_neighbor[0],  
                    QDP_forward, QDP_even)
```

```
eq -> eq, peq(+=), meq(-=), eqm(=-)
```

Shift and Map examples

```
QDP_Shift knight[4][4];  
int d[4] = {0,1,2,0};  
knight[2][3] = QDP_create_shift(d);
```

```
QDP_Shift mirror[4];  
int mu = 1;  
mirror[mu] =  
    QDP_create_map(reflect, &mu, sizeof(mu));
```

where `reflect(x, mu)` maps x_μ to $L_\mu - 1 - x_\mu$

Subset and Reduction examples

$$\forall_t \quad p(t) = \sum_{\vec{r}} a^*(\vec{r}, t) \cdot b(\vec{r}, t)$$

```
QDP_Subset ts[nt];
```

```
ts = QDP_create_subset(timeslice, nt);
```

```
where t = timeslice(r)
```

```
QLA_Complex p[nt];
```

```
QDP_DiracPropagator a, b;
```

```
QDP_c_eq_D_dot_D_multi(p, a, b, ts);
```

Temporary data removal

```
QDP_ColorMatrix *a;  
QLA_ColorMatrix *r;  
r = QDP_expose_M(a);
```

```
...
```

```
QDP_reset_M(a);
```

```
-----
```

```
QLA_ColorMatrix r[sites_on_node];
```

```
QDP_extract_M(r,a,QDP_odd);
```

```
...
```

```
QDP_insert_M(a,r,QDP_odd);
```

QDP/C Implementation

- # Built on QLA/C and QMP/C
- # Layout flexibility
- # Deferred resolution of shifts

Documentation and downloads

Release and documentation

<http://www.lqcd.org>