Message passing on the QCDSP Supercomputer

Michael Creutz

Brookhaven Lab

Is the QCDSP machine general purpose?

Requires:

1. A compiler
   - Tartan C++ compiler from Texas Instruments
   - individual nodes $\sim$ “50 Mhz PC’s”
2. Inter-node communication

Riken/BNL/Columbia software environment:

- tuned for lattice gauge theory
- physics wedded with machine geometry
- excellent performance (30% of peak)
- inflexible: 4-d gauge fields, SU(3), ...
- closed

This project:

- simple general “mimd” communication
- favor flexibility over efficiency
- machine geometry invisible at top level
- pure C/C++, no assembly
- http://thy.phy.bnl.gov/~creutz/qcdsp/

Top level

typedef yourobject DATASIZE;

- i.e. SU(3) matrices for lattice gauge

include "include/communication.C"

- inserts communication routines

allocate(n);

- sets up space for n items
- spread over processors

Three basic functions:

- store(i, & item);
- fetch(i, & item);
- worksync();

Any processor can store or fetch any item from any other

All processors call worksync()

- messages passed
- pending stores/fetches completed
- multiple stores/fetches can be stacked
- synchronizes machine

Conveniences

Variants:

- add to stored value: add(i, & item);
- multiple fetch:
  fetch(int *i, DATASIZE **d, int count);
- overloadings: item=fetch(i);

Auxiliary functions:

- if (onnode(i)) // item i is local
- global sums: int, float, DATASIZE
- broadcast
- global and, global or
- global max, global min
- cmalloc: fast "chip" memory

System functions should eventually replace these

Test examples

1. Fast Fourier Transform
   - recursively subdivide lattice
   - each half to half the processors
   - when sub-lattice on 1 processor: standard fft
   - combine results: heavy communication
   - not very fast :(=

2. Pure gauge theory
   - 2/3 speed of RBC pure gauge code
   - arbitrary space time dimensions
   - arbitrary (even) sides
   - arbitrary SU(N)
   - fetch neighbors in parallel

3. Grassmann integration

- my lattice '98 poster
- distributed hash table
- each processor handles local table
- non-local communication in parallel
- runs well, memory limited

More on Hash Tables

Extend communications class

- class hindex
- class hvalue
- include hashcom.C

Store hvalues indexed by hindices

- distributed randomly over machine
- processor handles his part of the table
- final locations unknown and unneeded

Basic functions:

- hstore(hindex,hvalue);
- hvalue hfetch(hindex);
- worksync();

Fast parallel loops over table:

- operate on local elements
- send results to network
- nonlocal communication in parallel
- check messages to keep things moving

Middle level
what's happening

Message structure:

- destination
- source
- data element
- action: store, fetch, acknowledge, error, ...
- extra: index, ...

Toroidal 4-d architecture:

- send on positive X,Y,Z, or T wires
- receive from negative wires
- route not predetermined
- lookup table for good wires
- all wires can be active
- store & acknowledge circle machine

Internal functions

sendmessage()
- activate one free wire towards destination
- if none available, insert in queue

readmessage()
- check for finished wires
- forward messages not for here

handlemessages()
- call readmessage()
- perform requested actions
- sends acknowledgements
- check message queue

worksync()
- globally flag unfinished stores/fetches
- call handlemessages() until done

Bottom level
machine interface

Serial communication unit – SCU

- initialize registers for message size
- set receive address registers
- write to send address register starts transfer
- poll status register for progress:

  $$\text{if } ((0x1111\& \sim (*(SCU\_BASE + 0x40)))) \ \ldots$$
- all in C/C++, no assembly

worksync()

- use two (of three) global interrupt lines
- first flags unfinished stores/fetches
- second syncs when done

Summary

Simple top level interface to QCDSP

High level language

Rapid prototyping

Flexibility over efficiency

- factor of 2-3 loss?

Performance

- FFT: poor, excessive communication
- Lattice Gauge: acceptable, great flexibility
- Grassmann: excellent, memory issues