Application Accelerators: Deus ex machina?

SC06 International Workshop on Performance Analysis and Optimization of High-End Computing Systems

Jeffrey S. Vetter¹,², Sadaf Alam¹, Nikhil Bhatia¹, Jeremy Meredith¹, Philip Roth¹
Future Technologies Group @ ORNL

- Founded October 1, 2004

- Future Technologies Group
  - Sadaf Alam
  - Richard Barrett
  - Nikhil Bhatia
  - Tammy Darland
  - Jeff Kuehn
  - Collin McCurdy
  - Jeremy Meredith
  - Ken Roche
  - Philip Roth
  - Olaf Storaasli
  - Jeffrey Vetter
  - Weikuan Yu
  - Micah Beck, joint
  - David Bader, joint

- Sponsors and Collaborators
  - DARPA HPCS
  - DoE MICS
  - SciDAC PERI
  - SciDAC SDM
  - SciDAC PDSI
  - FAST-OS
  - DoD HPCMP
  - LDRD
  - Applications Teams
  - Cray Supercomputing Center for Excellence
  - Vendors
  - Many others

http://ft.ornl.gov/
Highlights

**Background and motivation**
- Current trends in architectures favor two strategies
  - Homogenous multicore
  - Application accelerators
- Correct architecture for an application can provide astounding results

**Challenges to adopting application accelerators**
- Performance prediction
- Productive software systems

**Solutions**
- Modeling assertions
- Multi-paradigm programming
The Drama

➔ Years of prosperity
  – Increasing large-scale parallelism
  – Increasing number of transistors
  – Increasing clock speed
  – Stable programming models and languages

➔ Notable constraints force a new utility function for architectures
  – Power
  – Heat / thermal envelope
  – Signaling
  – Packaging
  – Memory, I/O, interconnect latency and bandwidth
  – Instruction level parallelism
  – Market trends favor ‘good enough’ computing – *Economist*
Current Approaches to Continued Performance Improvement

» Chip Multiprocessors
  – Homogenous multicore
    • Intel
    • AMD
    • IBM
  – Heterogeneous Multicore
    • STI CELL

» Application accelerators to augment general purpose multi-cores
Results from Initial Multicores Provide Performance Boost

Quad-core results are forthcoming…
Quad Kilo-core chips are on the way!

- 4 core chips already listed on Dell’s website
- 8 core chips likely
- ??

- Rapport
  - Rapport currently offers a 256 core chip at 100 Mhz
  - Planning 1024 core chip in 2007 – Kilocore™
  - Targeted at mobile and other consumer applications
Enter Application Accelerators

Optional hardware installed to accelerate applications beyond the performance of the general purpose processor

<table>
<thead>
<tr>
<th></th>
<th>Intel Woodcrest Dual Core</th>
<th>NVIDIA Quadro FX 4500 GPU</th>
<th>NVIDIA GeForce 6600 GPU</th>
<th>IBM Cell Processor</th>
<th>ClearSpeed Avalon</th>
</tr>
</thead>
<tbody>
<tr>
<td>clock frequency</td>
<td>3.0 GHz</td>
<td>470 MHz</td>
<td>350 MHz</td>
<td>3.2 GHz</td>
<td>250 MHz</td>
</tr>
<tr>
<td>type</td>
<td>CPU</td>
<td>accelerator card</td>
<td>accelerator card</td>
<td>CPU</td>
<td>accelerator card</td>
</tr>
<tr>
<td>power usage</td>
<td>80 W</td>
<td>110 W</td>
<td>30 W</td>
<td>100 W</td>
<td>20 W</td>
</tr>
<tr>
<td>speed single / double precision</td>
<td>~48 GFLOPS / ~24 GFLOPS</td>
<td>180 GFLOPS / NA</td>
<td>20 GFLOPS / NA</td>
<td>~256 GFLOPS / ~25 GFLOPS</td>
<td>50 GFLOPS / 50 GFLOPS</td>
</tr>
<tr>
<td>typical size</td>
<td>CPU socket</td>
<td>PCIe / MXM¹ card</td>
<td>PCIe / MXM¹ card</td>
<td>CPU socket</td>
<td>PCI-X card</td>
</tr>
<tr>
<td>cooling</td>
<td>heatsink + fan</td>
<td>heatsink + fan</td>
<td>HS-only or HS+fan</td>
<td>heatsink + fan</td>
<td>HS-only</td>
</tr>
</tbody>
</table>

¹ MXM: Mezzanine Mezzanine Module
For Example ... Graphics Cards
For Example ... STI Cell
For Example ... ClearSpeed
For Example ... FPGAs
AMD and Intel are Embracing this Trend with the Torrenza and Geneseo
Experimental Computing Lab @ ORNL

- 144 processor Cray XD1, each containing dual 2.2GHz Opteron processors, w/ six nodes connected via HyperTransport Virtex-II Pro FPGA
- Numerous dual-core, quad-core servers
- Access to Cray XT3, Cray X1E, Cray MTA2
- Dual processor, 500 GB RAID files
- Various simulators and SDKs
- A 31-node LNXI cluster consisting of 32-bit Intel Xeon 2.6GHz processors, networked with 10/100 Ethernet, which serves as a Lustre and OpenSSI testbed.
- An Iwill H8501 server with 8 1.8GHz dual core Opteron processors with 32 GB of memory on a NUMA HT interconnect, configured as a 16-way SMP.

Accelerators

- A Cell Broadband Engine (CBE) blade system with dual 2.4GHz Cell processors, each with a 64-bit Power Architecture PPE core and eight SPE SIMD cores.
- 144-port Flextronics Infiniband switch connecting LNXI cluster (48 HCAs presently)
- An SRC-6C MAPstation Reconfigurable Computing Platform pairing dual 2.8GHz Xeon processors with the Xilinx Virtex-II FPGA connected via DIMM slots.
- Several variants of GPU accelerators, including NVIDIA NV3x, NV4x, and G7x based cards in dual-processor Opteron and Xeon machines.
- An ATI FireStream 1GB PCIe GPU-based stream computing card.
- An AGEIA PhysX P1 PCI 128MB GDDR3 physics accelerator board.
- Three Digilent Virtex-II Pro FPGA Development System boards, with a variety of I/O ports, including USB and Ethernet.
- A Nallatech XtremeDSP Development Kit with the Xilinx Virtex-II Pro FPGA and dual-channel high-performance ADCs and DACs.
- Two ClearSpeed Advance PCI-X boards, each capable of 100GF.
Architectures that Match Application Requirements can offer Impressive/Astounding Performance Benefits

- Numerous FPGA results on integer, logic, flop applications
  - 40x on Smith-Waterman
  - 10x speedup on MD
- Numerous results on GPU
  - Geo-registration offers 700x speedup over commodity processor
  - MD
  - GPGPU …
- HPCC RandomAccess on Cray X1E
  - 7 GUPS on 512 MSPs

### Molecular Dynamics

<table>
<thead>
<tr>
<th>System</th>
<th>Seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell PPE</td>
<td>0.425</td>
</tr>
<tr>
<td>MTA2 w/32 procs</td>
<td>~0.035</td>
</tr>
<tr>
<td>2.2GHz Opteron</td>
<td>0.125</td>
</tr>
<tr>
<td>Cell w/ 8 SPEs</td>
<td>0.013</td>
</tr>
<tr>
<td>GPU (7900GT)</td>
<td>0.012</td>
</tr>
</tbody>
</table>
Disruptive Technologies and the S-Curve

Déjà vu?
- Floating Point Systems accelerator (1970-80s)
- Weitek coprocessors (1980s)

Some differences …
- Flops are free
- Power and thermal envelopes are constraining designs
Challenges for Accelerators, Users, Organizations, (and multicores?)

- **Performance prediction**
  - Should I purchase an accelerator?
  - What will be the performance improvement on my application workload with the accelerator?
  - Is the accelerator working as we expect?
  - How can I optimize my application for the accelerator?

- **Productive software systems**
  - Do I have to rewrite my application for each accelerator?
  - How stable is the functionality across systems?
  - How stable is the performance across systems?
Performance Prediction
Symbolic Performance Models with MA

Modeling Assertion (MA) = Empirical data + Symbolic modeling

- Advantages over traditional modeling techniques
  - Modularity, portability, and extensibility
  - Parameterized, symbolic models are evaluated with Matlab and Octave

- Construct, validate, and project application requirements as a function of input parameters

Declare important application variables

Declare important application operations

Annotate code with MA API

Validate Modeling Assertions empirically at runtime

Terminate when model is representative and error level is acceptable

Incrementally refine model based on error rates by adding and modifying variable and operation declarations
MA Framework

MA API in C (for Fortran & C applications With MPI)

Classes of API calls currently implemented and tested

ma(f)_subroutine_start/end
ma(f)_loop_start/end
ma(f)_flop_start/stop
ma(f)_heap/stack_memory
ma(f)_mpi_xxxx
ma(f)_set/unset_tracing

Source code annotation

Runtime system generate trace files

Model validation

Control flow model

Symbolic model

Post-processing toolset (in Java)

main ()
{
    ...
    loop (NAME = conj_loop) (COUNT = niter)
    {
        loop (NAME = norm_loop) (COUNT = 12npcols)
        {
            mpi_irecv (NAME = nrecv) (SIZE = dp * 2);
            mpi_send (NAME = nsend) (SIZE = dp * 2);
        }
    }
}

send = niter*(12npcols*(dp*2)+12npcols*(dp)+cgitmax*(12npcols*(dp*na/num_proc_cols)+dp*na/num_proc_cols+12npcols*(dp)+12npcols*(dp)+12npcols*(dp)*12npcols*(dp)*na/num_proc_cols)+dp*na/num_proc_cols)+12npcols*(dp))
Example with MA Annotation

```fortran
! Example with MA Annotation
!
call maf_def_variable_int('na',na)
call maf_def_variable_int('nonzer',nonzer)
.....
call maf_def_variable_assign_int('num_proc_cols',
    ! '2^ceil(log(nprocs)/(2*log(2)))',num_proc_cols)
.....
call maf_loop_start('conj_loop','niter',niter)
do it = 1, niter.....
call maf_flop_start('flopzeta','4*na/num_proc_cols',
    ! 4*na/num_proc_cols)
do j=1, lastcol-firstcol+1
    norm_temp1(1) = norm_temp1(1) + x(j)*z(j)
    norm_temp1(2) = norm_temp1(2) + z(j)*z(j)
enddo
! MA MPI API call
call maf_flop_stop('flopzeta')
.....
call maf_loop_end('conj_loop',it-1)
.....
call maf_subroutine_start('conj_grad')
.....
call maf_mpi_irecv('l2rcv','dp*na/num_proc_cols',
    ! dp*na/num_proc_cols+12npcols*(dp)),) +
    ! 12npcols*(dp*na/num_proc_cols) +
    ! dp*na/num_proc_cols+12npcols*(dp))
call maf_subroutine_end('conj_grad')
```

Input parameters: na, nonzer, niter and nprocs

Derived parameters: nz, num_proc_cols, l2cpcols and dp (size of REAL)

End markers used for validation

Markup for subroutine invocation

MA MPI API call

Send operation count for floating-point operation count

Send a loop with loop count

Example Model Validation

**NAS CG**
Class S: na=1400, nonzer=7  
Class W: na=7000, nonzer=8  
Class A: na=14000, nonzer=11  
Class B: na=75000, nonzer=13  
Class C: na=150000, nonzer=15

**NAS SP**
Class S: problem_size=7  
Class W: problem_size=36  
Class A: problem_size=64  
Class B: problem_size=102  
Class C: problem_size=162

opq: ma_flop:7000:7000:0.0: PASS=50: FAIL=0  
cj_sumred: ma_loop:1:1:0.0: PASS=50: FAIL=0  
l4rcv: ma_mpi_irecv:8:8:0.0: PASS=50: FAIL=0  
l4snd: ma_mpi_send:8:8:0.0: PASS=50: FAIL=0  
sumred: ma_flop:1:1:0.0: PASS=50: FAIL=0  
floprhopq: ma_flop:21001:21001:0.0: PASS=50: FAIL=0  
cj_rho: ma_loop:1:1:0.0: PASS=50: FAIL=0  
l5rcv: ma_mpi_irecv:8:8:0.0: PASS=50: FAIL=0  
l5snd: ma_mpi_send:8:8:0.0: PASS=50: FAIL=0  
flopnzx: ma_flop_start:3503:4347:-0.194: PASS=0: FAIL=2
Modeling Assertions for Accelerators

- **MA framework provides critical information on computational intensity and data movement that is critical for mapping applications to accelerators**

- **MA is providing insight into DOE applications for acceleration**
  - Biomolecular application: AMBER
  - Climate Modeling: POP
Large Scale App Case Study: Parallel Ocean Program

- Part of the Community Climate System Model (CCSM)
- Main simulation phases:
  1. Baroclinic process
  2. Barotropic process
- Key input parameters:
  1. Grid points: imt_global, jmt_global, km and nt
  2. MPI processor grid: nproc_x * nproc_y
  3. imt and jmt grid points are distributed and mapped onto nproc_x and nproc_y MPI processor grid
- MA communication models are developed and validated for a standard input configuration called x1
  - imt_global=320
  - jmt_global=384
  - km=40
  - nt=2
- Additional features:
  - Time-step simulations
  - Dynamic feature: Convergence of the Conjugate Gradient calculations
  - SPMD programming model
Existing Performance Analysis of POP

Figure 3. Performance in seconds per model day as a function of processor count for the baroclinic section in the x1 configuration.

Figure 4. Performance in seconds per model day as a function of processor count for the barotropic solver in the x1 configuration.

Insight into the Barotropic Calculation using these MA Symbolic Models

Important applications features expressed symbolically: communication, computation, memory size, I/O, etc.
Evaluation of the MA Symbolic Models for POP

Effect of grid configuration on message volume

MPI Send Message Volume for the Two Calculation Phases: Barotropic and Baroclinic
Sensitivity Analysis: How do Input Grid Parameters Influence MPI Message Volume?

We increased POP grid parameters (imt_global & jmt_global), for a fix-sized MPI grid 32x32.

x1(imt_global) = 320
x2(imt_global) = 640
x3(imt_global) = 960
x1(jmt_global) = 384

Over 98% of total messages are sent in barotropic calculation phase.

Observations: The impact of imt_global and jmt_global is different on different MPI messages. Some messages are invariant to one or the other while some increase linearly with an increase in the input parameter value.
Mapping Amber to FPGAs

Mapping to FPGAs

Obtained 3x **application** speedup on FPGA using HLL on SRC 6C MapStation.

jac Amber8 benchmark:

List time (% of nonbond) = 4.72 (5.19)
Direct Ewald time = **70.82**
Recip Ewald time = 14.76
Total Ewald time (% of nonbond) = 86.23 (94.81)

FFT time (% of Recip) = 4.76 (32.24)
MPPS: Multi-Paradigm Programming System
Multi-Paradigm Computing Challenges

- Multi-Paradigm systems offer lots of performance potential, but...

- ...it is challenging to realize that potential
  - Different APIs, different tools, different assumptions!
  - Different ISAs, SDKs
  - Explicit data movement
  - Simplistic scheduling
  - Static binding to available resources
MPPS Basis: Multi-Paradigm Procedure Call (MPPC)

- Multi-Paradigm Procedure Calls
  - Adopt highly successful RPC approach
  - Open protocol for communication within infrastructure

- MPPC runtime system
  - Runtime agent to manage access to device
  - Directory service for dynamic discovery of devices and their status
  - Local service OS on devices (if possible)

- Support for defining adaptive policies for scheduling application requests onto computing devices
  - Simple policies built-in
  - Custom policies can be driven by automated administration and performance tools
Compiler Support for MPPS

- **Pragmas identify regions of code to accelerate**
  - Built on Open64
  - Similar to OpenMP analysis
    - #mpps accel

- **Extracts code for device service**
  - Device code compiled separately with device specific SDK

- **Replaces original code with MPPC call**
  - Marshals data; starts, waits on device
Summary

- Accelerators will continue to gain market share in one form or another
  - Expansion slots
  - On-chip accelerators which are used as necessary

- Performance modeling and analysis will become critical challenges for procurements, validation, and optimization
  - Modeling assertions

- Software systems that can mask the complexity will become much more important
  - Multi-paradigm Programming System
  - Automated generation of MPPC calls
This research was sponsored by the Office of Mathematical, Information, and Computational Sciences, Office of Science, U.S. Department of Energy under Contract No. DE-AC05-00OR22725 with UT-Batelle, LLC. Accordingly, the U.S. Government retains a non-exclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.

http://ft.ornl.gov

vetter@computer.org