Requirements for Scalable Application Specific Processing in Commercial HPEC

Steve Miller
Chief Engineer
## The 3 Single-Paradigm Architectures

<table>
<thead>
<tr>
<th>Scalar</th>
<th>Vector</th>
<th>App-Specific</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel Itanium</td>
<td>Cray X1</td>
<td>Graphics - GPU</td>
</tr>
<tr>
<td>SGI MIPS</td>
<td>NEC SX</td>
<td>Signals - DSP</td>
</tr>
<tr>
<td>IBM Power</td>
<td></td>
<td>Prog’ble - FPGA</td>
</tr>
<tr>
<td>Sun SPARC</td>
<td></td>
<td>Other ASICs</td>
</tr>
<tr>
<td>HP PA</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Scalar**: Intel Itanium, SGI MIPS, IBM Power, Sun SPARC, HP PA

**Vector**: Cray X1, NEC SX

**App-Specific**: Graphics - GPU, Signals - DSP, Prog’ble - FPGA, Other ASICs
Architectural Challenges

- **Hardware**
  - Bandwidth to/from System
  - Scalability

- **Software**
  - Compilers/Languages
  - Debuggers
  - APIs
Multi-Paradigm Computing *UltraViolet*

Terascale to Petascale Data Set: Bring Function to Data

**Scalable Shared Memory**
- Globally addressable
- Thousands of ports
- Flat & high bandwidth
- Flexible & configurable

reiScale to Petascale Data Set: Bring Function to Data

**Scalable Shared Memory**
- Globally addressable
- Thousands of ports
- Flat & high bandwidth
- Flexible & configurable

**Compute**
- Vector
- FPGA
- DSP

**Graphics**
- Reconfigurable
- IO

**FPGA**
- DSP
- Graphics

**IO**
- Reconfigurable
Software

• Provide for HDL modules
  Integrated environment with debugger
  Highest performance

• Leverage 3rd Party Std Language Tools
  Celoxia, Impulse Acceleration, Mitrion, Mentor Graphics

• Developed an FPGA aware version of GDB
  Capable of debugging the FPGA and System Software
  Capable of multiple CPUs and multiple FPGAs

• Developed RASC Abstraction Layer (RASCAL)
Software Overview

- Debugger (GDB)
- Application
- Abstraction Layer Library
- Algorithm Device Driver
- COP (TIO, Algorithm FPGA, Memory, Download FPGA)
- Download Utilities
- Device Manager
- Download Driver
- Linux Kernel
- Hardware
- User Space
The Abstraction Layer’s algorithm API mirrors the COP API with a few additions that enable wide scaling, and deep scaling.
Hardware

• Direct Connection to NUMAlink4
  6.4GB/s/connection

• Fast System Level Reprogramming of FPGA

• Atomic Memory Operations
  Same set as System CPUs

• Hardware Barriers

• Configurations to 8191 NUMA/FPGA connections
MOATB Block Diagram

NUMAlink  12.8 GB/s
SSP   6.4 GB/s
QDR SRAM  9.6 GB/s
3 reads @  1.6GB/s
3 writes @  1.6GB/s