The von Neumann Architecture and Alternatives

Matthias Fouquet-Lapar
Senior Principal Engineer
Application Engineering
mfl@sgi.com
The von Neumann bottleneck and Moore’s law
Physical semi-conductor limits

• Transistor Count is continuing to increase according to Moore’s Law for the foreseeable Future
• However, we hit physical limits with current semi-conductor technology in increasing clock-frequency - we have to extract the heat somehow from the chip
• So what can be done with all these transistors to increase performance (wall-clock time) if we can’t increase clock frequency?
Caches and Hyper-Threading

• In a von Neumann architecture larger and larger on and off caches are used to hide memory latencies
• This makes it easy for the programmer – but your effective silicon utilization will be relatively low (10% - 20%)
• Hyper-Threading allows additional hiding of memory references by switching to a different thread
Current Top-End Microprocessor Technology

- Xeon Quad-Core 268 million transistors
  4 MB L2 cache
- Tukwila Quad-Core 2-billion transistors
  30 MByte on-die cache
- Majority of the micro-processor surface (80 - 90%) is used for caches
Many / Multi-Core

- Additional transistors can be used to add cores on the same die
- Dual / Quad Cores are standard today, Many Cores will be available soon
- But (and you already guessed that there is no free lunch) there is **ONE** problem
The ONLY problem

- is your application / workflow, because it has to be parallelized

- and while you’re doing this, why not look at alternatives to get
  - higher performance
  - better silicon utilization
  - less power consumption
  - better price/performance
Application Specific Acceleration

- The way and degree of parallelization is Application specific and we feel that is important to have a set of Accelerator choices to build the Hybrid Compute Solution for your workflow
- SGI supports different Hybrid Acceleration Alternatives
SGI’s history and strategic directions in Application Specific Acceleration

- 25 years of Experience in Accelerating HPC Problems
- 10 years of Experience in Building Application Specific Accelerators for large scale Super-Computer Systems
  - 1984 first ASIC based IRIS workstation
  - 1998 TPU Product for MIPS based O2K/O3K Architectures
  - 2003 FPGA Product for ALTIX IA64 based Architectures
- 2008 Launch of the Accelerator Enabling Program
A Disruptive Technology 1984
Implement High-Performance Graphics in ASICs

- 1000/1200
  - 8MHz Motorola 68000 - PM1 (variant of Stanford UNiversity "SUN" board)
  - 3 -4 MB RAM Micro Memory Inc. Multibus No Hard Disk
  - Ethernet: Excelan EXOS/101
    - Graphics:
      - GF1 Frame Buffer
      - UC3 Update Controller
      - DC3 Display Controller
      - BP2 Bitplane
1998 : Tensor Processing Unit

An advanced form of Array Processing that is:

Hybrid Computational Model

Library-based Programming Model

Adaptive in execution

Slide 11
TPU: Application Specific Acceleration

- 10 Floating Point Functional Units (1.5 Gflops)
- 12 Application Specific Functional Units (3.1 Gops)
- 8 Mbytes SRAM Buffer Memory (6.4 GBytes/s)

"Deeply Pipelined & Heavily Chained Vector Operations"

Example Image Formation Application Acceleration
(4096 X 8192 32bit initial problem set)
TPU: Adaptive Processing (customer co-development)

Reconfigurable Data Paths

Programmable Sequence Controller

Data Dependent Dynamic Addressing

Host Memory

Data Source Logical Array

Data Destination Logical Array

Microcode Source Logical Array

XTalk DMA

DMA

Input/Output Buffer

Floating Point XBAR

DDA

DDA

Buffer Memory

Multiple Independent Arithmetic

Very Large Buffer Memory

Logical and Addressing Resources

Data Source

Logical Array

Data Destination

Logical Array

Microcode Source

Logical Array

Logic Array

Data

Source

Logical Array

Microcode

Source

Logical Array

DDA

DDA

Buffer Memory

Multiple Independent Arithmetic

Very Large Buffer Memory

Logical and Addressing Resources

Data Source

Logical Array

Data Destination

Logical Array

Microcode Source

Logical Array

DDA

DDA

Buffer Memory

Multiple Independent Arithmetic

Very Large Buffer Memory

Logical and Addressing Resources
SGI Tensor Processing Unit - Applications

- RADAR
- Medical Imaging (CT)
- Image Processing
  - Compression
  - Decompression
  - Filtering
- Largest Installation with 150 TPUs
- Continues to be sold and fully supported today
Accelerator Enabling Program

- FPGA Technology:
  - SGI RC100
  - XDI 2260i
- GPU & TESLA / NVIDIA
- ClearSpeed
- PACT XPP
- Many core CPUs
- IBM Cell
Accelerator Enabling Program

- Increased focus on work with ISVs and key customers providing the best accelerator choice for different scientific domains
- Application Engineering group with approx 30 domain experts will play an essential role
- Profiling existing applications and workflows to identify hot-spots – a very flat profile will be hard to accelerate
SGI Options for Compute Bound Problems

- Specialized Computing Elements
- Most of them come from the embedded space:
  - High Density
  - Very Low Power Consumption
  - Programmable in High Level Languages
- These attributes make them a very good fit for High Performance Computing
ALTIX 450/4700 ccNUMA System Infrastructure

CPU node

RASC™ (FPGAs)

Scalable GPUs

General Purpose I/O Interfaces

NUMAlink™ Interconnect Fabric

Slide 19
RASCAL: Fully Integrated Software Stack for Programmer Ease of Use

- Debugger (GDB)
- Performance Tools
- Download Utilities
- Application
- Abstraction Layer Library
- Algorithm Device Driver
- Download Driver
- COP (TIO, Algorithm FPGA, Memory, Download FPGA)

User Space
- Device Manager
- Hardware
- Linux® Kernel

Slide 21
Application using 1 FPGA:

```c
strcpy(ar.algorithm_id,"my_sort");
ar.num_devices = 1;
rasclib_resource_alloc(&ar,1);
```

Application using 70 FPGAs:

```c
strcpy(ar.algorithm_id,"my_sort");
ar.num_devices = 70;
rasclib_resource_alloc(&ar,1);
```
Demonstrated Sustained Bandwidth

RC100 System Bandwidth

Bandwidth (GB/s)

# of FPGAs
As well as running “off-the-shelf” Real-world Applications

- Using Mitrion Accelerated BLAST 1.0 without modifications
- Using RASCAL 2.1 without modifications
- Test case:
  - Searching the Unigene Human and Refseq Human databases (6,733,760 sequences; 4,089,004,795 total letters) with the Human Genome U133 Plus 2.0 Array probe set from Affymetrix (604,258 sequences; 15,106,450 total letters).
  - Representative of current top-end research in the pharmaceutical industry.
  - Wall-Clock Speedup of 188X compared to top-end Opteron cluster
World largest FPGA SSI Super Computer

- Setup in 2 days using standard SGI components
- Operationally without any changes to HW or SW
- 70 FPGAs (Virtex 4 LX200)
- 256 Gigabyte of common shared memory
- Successfully ran a bio-informatics benchmark.
- Running 70 Mitrionics BLAST-N execution streams speedup > 188x compared to 16 core Opteron cluster

- Aggregate I/O Bandwidth running 64 FPGAs from a single process: **120 Gbytes/sec**
Picture please 😊
XDI 2260i FSB socket

- Application FPGA
  - Altera Stratix III
  - SE 260
  - 8 MB SRAM

- Application FPGA
  - Altera Stratix III
  - SE 260
  - 8 MB SRAM

Bridge FPGA

- 9.6 GB/s

- Configuration

- 32 MB Configuration Flash

- Intel FSB
  - 8.5 GB/s
Intel Quick Assist

• Framework for X86 based accelerators
• Unified interface for different accelerator technologies
  – FPGAs
  – Fixed and Floating Point Accelerators
• It provides flexibility to use SW implementations of accelerated functions with an unified API regardless of the accelerator technique used
Intel Quick Assist AAL overview

Application

AAS (Accelerator Abstraction Services)

AIA (Accelerator Interface Adaptor)

Intel FSB (Front-Side Bus) AHM (Accelerator HW Module) Driver

Slide 30
NVIDIA – G80 GPU & TESLA
• **GPUs** are massively multithreaded many-core chips
  – NVIDIA Tesla products have up to 128 scalar processors
  – Over 12,000 concurrent threads in flight
  – Over 470 GFLOPS sustained performance

• **CUDA** is a scalable parallel programming model and a software environment for parallel computing
  – Minimal extensions to familiar C/C++ environment
  – Heterogeneous serial-parallel programming model

• NVIDIA’s **TESLA** GPU architecture accelerates CUDA
  – Expose the computational horsepower of NVIDIA GPUs
  – Enable general-purpose **GPU computing**

• CUDA also maps well to multi-core CPUs!
• Thread Execution Manager issues threads
• 12288 concurrent threads managed by HW
• 128 Thread Processors grouped into 16 Multiprocessors (SMs)
• 1.35 Ghz 128 Processors == 518 GFLOPS (SP)
• Parallel Data Cache (Shared Memory) enables thread cooperation
• Intel FSB based Accelerator
  – Plug in compatible into Intel XEON family, fits XEON socket
  – Intel QuickAssist support
• 48 FNA PAEs (Function Processing Array Elements)
  – 16-bit VLIW cores processing sequential algorithms
  – 84500 MIPS raw performance
  – Optimized for control flow dominated, irregular code
  – Chained operations (up to 4 levels) can be combined with predicated execution allowing for example to execute a if-then-else statement in one single cycle
PACT XPP Overview
From von-Neumann Instruction to Configuration/Data Flow

• 912 ALU Processing Array Element
  – 16-bit ALUs for dataflow processing
  – Logic operations, basic arithmetic operators (add/subtract)
  – „Special arithmetic“ including comparators and multipliers
  – reconfigurable compute fabric (reconfiguration through FNC PAE or event driven)
  – Supports partial configuration
  – 319 GOPS raw performance
  – Scaleable ALU architecture: 16-, 32-, 64-bit integer processing

• Fast program development and short turnaround cycles
  – ANSI C Programming and Debugging Environment
PACT XPP-3m XEON compatible Accelerator Module

- 4x XPP-3c DSP
- Fits into XEON 5000/7000 socket
- FSB Bandwidth 8 GBytes/s peak
- Module dimensions according to keep-out area
- Local memory eliminates system memory accesses
- XPP-3c–to–FSB Bridge interconnect via Proprietary Serial High-Speed Interface
  - 2.5GBytes/s per channel
  - 10GBytes/s total
- Power Dissipation 30W
PACT XPP FSB module
PACT – XPP Tool Chain

C-Based
- ANSI-C / C++ based Development System
- Cycle Accurate Simulator
- Debugging Environment

Libraries
- Intel IPP Functions
- DSP Library
- Imaging
- Video and Audio
- Crypto (DES, AES)

System Integration
- Intel AAL / QuickAssist
XPP-3 Benefits and Target Applications

Target Applications
- High-End DSP Processing
  - Video, Imaging, Biomedical, Recognition/Analysis
- Search and Pattern Matching
- Integer and Fixed Point Algebra

Benefits
- Software development flow (ANSI C / C++)
- Fast compile time and turnaround cycles
- Extensive Libraries
- Intel QuickAssist support
- Low Power Dissipation (30W including memory)
- Reduced system load
  - Local memory eliminates main memory accesses
- Price/Performance ratio
PACT XPP next generation

- Increasing core frequency from 350 Mhz to 1.2 Ghz
- Quadrupling amount of processing units (ALU-PAEs)
- Process shrink from 90nm to 45 nm
- Addition of Configurable Floating Point ALUs
  - 16 bit FP 240 GFLOPS
  - 32 bit FP 120 GFLOPS
  - 64 bit FP 60 GFLOPS
PACT Example: 2D Edge Detection

```c
#include "XPP.h"

main() {
    int v, h, inp;
    int p1[VERLEN][HORLEN]; int p2[VERLEN][HORLEN];
    int htmp, vtmp, sum;

    for(v=0; v<VERLEN; v++)
        for(h=0; h<HORLEN; h++) {
            XPP_getstream(1, 0, &inp);
            p1[v][h] = inp;
        }

    for(v=0; v<=VERLEN-3; v++)
        for(h=0; h<=HORLEN-3; h++) {
            htmp = (p1[v+2][h] - p1[v][h]) +
                    (p1[v+2][h+2] - p1[v][h+2]) +
                    2 * (p1[v+2][h+1] - p1[v][h+1]);
            if (htmp < 0) htmp = -htmp;

            vtmp = (p1[v][h+2] - p1[v][h]) +
                    (p1[v+2][h+2] - p1[v+2][h]) +
                    2 * (p1[v+1][h+2] - p1[v+1][h]);
            if (vtmp < 0) vtmp = -vtmp;
            sum = htmp + vtmp;
            if (sum > 255) sum = 255;
            p2[v+1][h+1] = sum;
        }

    for(v=1; v<=VERLEN-1; v++)
        for(h=1; h<=HORLEN-1; h++)
            XPP_putstream(4, 0, p2[v][h]);
```

Read input stream to Buffer

2D edge detection

Stream result from buffer to a port
PACT Example: 2D Edge Detection
Flow graph of module edge_m2

Result:
- 3 Modules:
- Edge_m1: loads RAMs
- Edge_m2: calculates
- Edge_m3: flushes RAMs
• **Multi-Threaded Array Processing:**
  - Designed for high performance, low power
  - Modular design
  - Programmed in high-level languages
  - Asynchronous, overlapped I/O

• **Scalable array of many Processor Elements (PEs):**
  - Coarse-grained data parallel processing
  - Supports redundancy and resiliency

• **Programmed in an extended version of C called C^n:**
  - Single “poly” data type modifier
  - Rich expressive semantics
ClearSpeed
Smart SIMD Processing Elements

- Multiple execution units
  - Floating point adder
  - Floating point multiplier
  - Fixed-point MAC $16 \times 16 \rightarrow 32 + 64$
  - Integer ALU with shifter
  - Load/store
- High-bandwidth, 5-port register file
- Fast inter-PE communication path (swazzle)
- Closely coupled SRAM for data
  - Keeping data close is key to low power
- High bandwidth per PE DMA (PIO)
- Per PE address generators
  - Complete pointer model, including parallel pointer chasing and vectors of addresses
  - Key for gather/scatter and sparse operations
- Platform design enables PE variants
  - E.g. memory size, data path widths etc.
ClearSpeed - The ClearConnect™ “Network on Chip” system bus

- Scalable, re-usable System on Chip (SoC) platform interconnect
- “Network-on-a-chip”
- System backbone
- Used to connect together all the major blocks on a chip:
  - Multiple MTAP smart SIMD cores
  - Multiple memory controllers
  - On-chip memories
  - System interfaces, e.g. PCI Express
- Blocks are connected into a unified memory architecture
- Distributed arbitration
- Scalable bandwidths
- Low power:
  - Power dissipation proportional to volume of data and distance travelled
The platform realized: The Clearspeed CSX700 Processor

- Includes dual MTAP cores:
  - 96 GFLOPS peak (32 & 64-bit)
  - 10-12W typical power consumption
    - 18W maximum power consumption
  - 250MHz clock speed
  - 192 Processing Elements (2x96)
  - 8 spare PEs for resiliency
  - ECC on all internal memories
- ClearConnect™ on-chip interconnect
- Dual integrated 64-bit DDR2 memory controllers with ECC
- Integrated PCI Express x16
- Integrated ClearConnect™ chip-to-chip bridge port (CCBR)
- 2x128 Kbytes of on-chip SRAM
- IBM 90nm process
- 266 million transistors
ClearSpeed software development environment

- $C^n$ optimising compiler
  - $C$ with **poly** extension for SIMD datatypes
  - Uses ACE CoSy compiler development system
- Assembler, linker
- Simulators
- Debuggers – csgdb
  - A port of the GNU debugger gdb for ClearSpeed’s hardware.
- Profiling – csprof
  - Heterogeneous visualization of an application’s performance while running on both a multi-core host and ClearSpeed’s hardware. Intimately integrates with the debugger.
- Libraries (BLAS, RNG, FFT, more..)
- High level APIs, streaming etc.
- Available for Windows and Linux (Red Hat and SLES)
  - Open source driver for Linux

Slide 49
ClearSpeed - Accelerated applications

- Molpro electronic molecular structure code
  - Collaboration with Dr Fred Manby’s group at Bristol University’s Centre for Computational Chemistry
- Sire QM/MM free energy code
  - Collaboration with Dr Christopher Woods
- BUDE molecular dynamics-based drug design
  - Collaboration with Dr Richard Sessions at Bristol University’s Department of Biochemistry
- Amber 9 implicit (production code)
  - Molecular dynamics simulation with implicit solvent
- Amber 9 explicit (demonstration)
  - Molecular dynamics simulation with explicit solvent
- Monte Carlo based financial codes
ClearSpeed - Amber 9 sander application acceleration

- Accelerated Amber 9 sander implicit shipping now (download from ClearSpeed website)
- Amber 9 sander explicit available in beta soon
- On CATS, both explicit and implicit should run as fast as 10 to 20 of the fastest contemporary x86 cores
  - Saving 90-95% of the simulation time -> more simulations
  - Alternatively, enabling larger, more accurate simulations
- While reducing power consumption by 66%
- And increasing server room capacity by 300%
ClearSpeed: Other acceleratable applications

- Computational Fluid Dynamics, Smooth Particle Hydrodynamics methods
  - Collaborations with Jamil Appa at BAE systems and Dr Graham Pullan at the Whittle Laboratory, Cambridge University
- Star-P from Interactive Supercomputing
- MATLAB and Mathematica when performing large matrix operations, such as solving systems of linear equations
ClearSpeed CATS
New Design Approach Delivers 1 TFLOP in 1U

- 1U standard server
- Intel 5365 3.0GHz
  - 2-socket, quad core
  - 0.096 DP TFLOPS peak
  - Approx. 600 watts
  - Approx. 3.5 TFLOPS peak in a 25 kW rack

- ClearSpeed Accelerated Terascale System
  - 24 CSX600 processors
  - ~1 DP TFLOPS peak
  - Approx. 600 watts
  - Approx. 19 TFLOPS peak in a 25 kW rack
  - 18 standard servers & 18 acceleration servers
Lessons learned - 1

- Not every Accelerator Technology is applicable to every HPC Problem – and it’s a moving target
- Not every HPC Application can be accelerated
- The Majority of Users is interested in complete Appliances/Applications
- “Ease-of-Use” is relative
- Standards starting to emerge now which will be key to broader acceptance
Lessons learned – 2

• Domain knowledge absolutely critical when working on acceleration
• Enabling accelerators for ISVs will be key
• Strategic partnerships are a great way to create synergies
• Think “out-of-the-line” to create unique solutions
Lessons learned - 3

- Keep in mind that a solution has to be price/performance competitive.
- There should be at least a 10-20X speedup compared to current top-end CPUs
- CPUs speeds (and multi-core architectures) continue to evolve
- Make sure your Algorithm Implementation scales across multiple Accelerators
- **Talk to the experts** – we are here to help you

Slide 56
A broader view of Hybrid Computing
Thank You