Introduction to the Cell Processor

Michael Perrone (mpp@us.ibm.com)
Class Agenda

- Motivation for multicore chip design
- Cell basic design concept
- Cell hardware overview
  - Cell highlights
  - Cell processor
  - Cell processor components
- Cell performance characteristics
- Cell application affinity
- Cell software overview
  - Cell software environment
  - Development tools
  - Cell system simulator
  - Optimized libraries
- Cell software development considerations
- Cell blade
Where have all the gigahertz gone?
Technology Scaling – We’ve hit the wall

![Graph showing relative device performance over years. The x-axis represents years from 1988 to 2012, and the y-axis represents relative device performance. The graph includes lines for Conventional Bulk CMOS, SOI (silicon-on-insulator), High mobility, and Double-Gate technologies. The graph suggests a trend of increasing performance, with a question mark indicating a point where the trend may not continue as expected.]
Power Density – The fundamental problem

Source: Fred Pollack, Intel. New Microprocessor Challenges in the Coming Generations of CMOS Technologies, Micro32
What’s Causing The Problem?

Gate dielectric approaching a fundamental limit (a few atomic layers)

Dr. Michael Perrone, IBM.
Has This Ever Happened Before?

Steam Iron 5W/cm²
Has This Ever Happened Before?

Steam Iron
5W/cm²

Dr. Michael Perrone, IBM.
6.189 IAP 2007

Lecture 2

The Multicore Approach
Cell

Dr. Michael Perrone, IBM.
Cell History

- IBM, SCEI/Sony, Toshiba Alliance formed in 2000
- Design Center opened in March 2001
  - Based in Austin, Texas
- Single Cell BE operational Spring 2004
- 2-way SMP operational Summer 2004
- February 7, 2005: First technical disclosures
- October 6, 2005: Mercury Announces Cell Blade
- November 9, 2005: Open Source SDK & Simulator Published
- November 14, 2005: Mercury Announces Turismo Cell Offering
- February 8, 2006 IBM Announced Cell Blade
Lecture 2

Cell Basic Design Concept
Cell Basic Concept

- **Compatibility with 64b Power Architecture™**
  - Builds on and leverages IBM investment and community
- **Increased efficiency and performance**
  - **Attacks on the “Power Wall”**
    - Non Homogenous Coherent Multiprocessor
    - High design frequency @ a low operating voltage with advanced power management
  - **Attacks on the “Memory Wall”**
    - Streaming DMA architecture
    - 3-level Memory Model: Main Storage, Local Storage, Register Files
  - **Attacks on the “Frequency Wall”**
    - Highly optimized implementation
    - Large shared register files and software controlled branching to allow deeper pipelines
- **Interface between user and networked world**
  - Image rich information, virtual reality
  - Flexibility and security
- **Multi-OS support, including RTOS / non-RTOS**
  - Combine real-time and non-real time worlds
Cell Design Goals

- Cell is an accelerator extension to Power
  - Built on a Power ecosystem
  - Used best known system practices for processor design
- Sets a new performance standard
  - Exploits parallelism while achieving high frequency
  - Supercomputer attributes with extreme floating point capabilities
  - Sustains high memory bandwidth with smart DMA controllers
- Designed for natural human interaction
  - Photo-realistic effects
  - Predictable real-time response
  - Virtualized resources for concurrent activities
- Designed for flexibility
  - Wide variety of application domains
  - Highly abstracted to highly exploitable programming models
  - Reconfigurable I/O interfaces
  - Virtual trusted computing environment for security
Cell Synergy

- Cell is not a collection of different processors, but a synergistic whole
  - Operation paradigms, data formats and semantics consistent
  - Share address translation and memory protection model

- PPE for operating systems and program control

- SPE optimized for efficient data processing
  - SPEs share Cell system functions provided by Power Architecture
  - MFC implements interface to memory
    - Copy in/copy out to local storage

- PowerPC provides system functions
  - Virtualization
  - Address translation and protection
  - External exception handling

- EIB integrates system as data transport hub
6.189 IAP 2007

Lecture 2

Cell Hardware Components
Cell Chip

Highlights (3.2 GHz)

- 241M transistors
- 235mm²
- 9 cores, 10 threads
- >200 GFlops (SP)
- >20 GFlops (DP)
- Up to 25 GB/s memory B/W
- Up to 75 GB/s I/O B/W
- >300 GB/s EIB
- Top frequency >4GHz (observed in lab)
Cell Features

- **Heterogeneous multicore system architecture**
  - Power Processor Element for control tasks
  - Synergistic Processor Elements for data-intensive processing

- **Synergistic Processor Element (SPE) consists of**
  - Synergistic Processor Unit (SPU)
  - Synergistic Memory Flow Control (MFC)
    - Data movement and synchronization
    - Interface to high-performance Element Interconnect Bus

![Diagram of Cell Features]

64-bit Power Architecture with VMX

Dr. Michael Perrone, IBM.
Cell Processor Components (1)

- **Power Processor Element (PPE):**
  - General purpose, 64-bit RISC processor (PowerPC AS 2.0.2)
  - 2-Way hardware multithreaded
  - L1: 32KB I; 32KB D
  - L2: 512KB
  - Coherent load / store
  - VMX-32
  - Realtime Controls
    - Locking L2 Cache & TLB
    - Software / hardware managed TLB
    - Bandwidth / Resource Reservation
    - Mediated Interrupts

- **Element Interconnect Bus (EIB):**
  - Four 16 byte data rings supporting multiple simultaneous transfers per ring
  - 96Bytes/cycle peak bandwidth
  - Over 100 outstanding requests
Cell Processor Components (2)

● Synergistic Processor Element (SPE):
  ■ Provides the computational performance
  ■ Simple RISC User Mode Architecture
    ▪ Dual issue VMX-like
    ▪ Graphics SP-Float
    ▪ IEEE DP-Float
  ■ Dedicated resources: unified 128x128-bit RF, 256KB Local Store
  ■ Dedicated DMA engine: Up to 16 outstanding requests

● Memory Management & Mapping
  ■ SPE Local Store aliased into PPE system memory
  ■ MFC/MMU controls / protects SPE DMA accesses
    ▪ Compatible with PowerPC Virtual Memory Architecture
    ▪ SW controllable using PPE MMIO
  ■ DMA 1,2,4,8,16,128 -> 16Kbyte transfers for I/O access
  ■ Two queues for DMA commands: Proxy & SPU
Cell Processor Components (3)

- **Broadband Interface Controller (BIC):**
  - Provides a wide connection to external devices
  - Two configurable interfaces (60GB/s @ 5Gbps)
    - Configurable number of bytes
    - Coherent (BIF) and / or I/O (IOIFx) protocols
  - Supports two virtual channels per interface
  - Supports multiple system configurations

- **Broadband Interface Controller (BIC):**
  - Provides a wide connection to external devices
  - Two configurable interfaces (60GB/s @ 5Gbps)
    - Configurable number of bytes
    - Coherent (BIF) and / or I/O (IOIFx) protocols
  - Supports two virtual channels per interface
  - Supports multiple system configurations
Cell Processor Components (4)

- **Internal Interrupt Controller (IIC)**
  - Handles SPE Interrupts
  - Handles External Interrupts
    - From Coherent Interconnect
    - From IOIF0 or IOIF1
  - Interrupt Priority Level Control
  - Interrupt Generation Ports for IPI
  - Duplicated for each PPE hardware thread

- **I/O Bus Master Translation (IOT)**
  - Translates Bus Addresses to System Real Addresses
  - Two Level Translation
    - I/O Segments (256 MB)
    - I/O Pages (4K, 64K, 1M, 16M byte)
  - I/O Device Identifier per page for LPAR
  - IOST and IOPT Cache – hardware / software managed

---

Dr. Michael Perrone, IBM.
6.189 IAP 2007

Lecture 2

Cell Performance Characteristics
Why Cell Processor Is So Fast?

- Key Architectural Reasons
  - Parallel processing inside chip
  - Fully parallelized and concurrent operations
  - Functional offloading
  - High frequency design
  - High bandwidth for memory and IO accesses
  - Fine tuning for data transfer

Staging

Memory → L2 - 4 outstanding loads + 2 prefetch

Data

Memory → L2 → SPU - 16 outstanding loads per SPU

Dr. Michael Perrone, IBM.
Theoretical Peak Operations

![Bar Chart]

- **Freescale MPC8641D 1.5 GHz**
- **AMD Athlon™ 64 X2 2.4 GHz**
- **Intel Pentium D® 3.2 GHz**
- **PowerPC® 970MP 2.5 GHz**
- **Cell Broadband Engine™ 3.2 GHz**

- **FP (SP)**
- **FP (DP)**
- **Int (16 bit)**
- **Int (32 bit)**

Dr. Michael Perrone, IBM.
Cell BE Performance

- BE can outperform a P4/SSE2 at same clock rate by 3 to 18x (assuming linear scaling) in various types of application workloads

<table>
<thead>
<tr>
<th>Type</th>
<th>Algorithm</th>
<th>3 GHz GPP</th>
<th>3 GHz BE</th>
<th>BE Perf Advantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>HPC</td>
<td>Matrix Multiplication (S.P.)</td>
<td>25 Gflops</td>
<td>190 GFlops (8SPEs)</td>
<td>8x</td>
</tr>
<tr>
<td></td>
<td>Linpack (S.P.)</td>
<td>18 GFlops (IA32)</td>
<td>150 GFlops (BE)</td>
<td>8x</td>
</tr>
<tr>
<td></td>
<td>Linpack (D.P.)</td>
<td>6 GFlops (IA32)</td>
<td>12 GFlops (BE)</td>
<td>2x</td>
</tr>
<tr>
<td>bioinformatic</td>
<td>smith-waterman</td>
<td>570 Mcups (IA32)</td>
<td>420 Mcups (per SPE)</td>
<td>6x</td>
</tr>
<tr>
<td>graphics</td>
<td>transform-light</td>
<td>160 MVPS (G5/VMX)</td>
<td>240 MVPS (per SPE)</td>
<td>12x</td>
</tr>
<tr>
<td></td>
<td>TRE</td>
<td>1.6 fps (G5/VMX)</td>
<td>24 fps (BE)</td>
<td>15x</td>
</tr>
<tr>
<td>security</td>
<td>AES</td>
<td>1.1 Gbps (IA32)</td>
<td>2Gbps (per SPE)</td>
<td>14x</td>
</tr>
<tr>
<td></td>
<td>TDES</td>
<td>0.12 Gbps (IA32)</td>
<td>0.16 Gbps (per SPE)</td>
<td>10x</td>
</tr>
<tr>
<td></td>
<td>MD-5</td>
<td>2.68 Gbps (IA32)</td>
<td>2.3 Gbps (per SPE)</td>
<td>6x</td>
</tr>
<tr>
<td></td>
<td>SHA-1</td>
<td>0.85 Gbps (IA32)</td>
<td>1.98 Gbps (per SPE)</td>
<td>18x</td>
</tr>
<tr>
<td>communication</td>
<td>EEMBC</td>
<td>501 Telemark (1.4GHz mpc7447)</td>
<td>770 Telemark (per SPE)</td>
<td>12x</td>
</tr>
<tr>
<td>video</td>
<td>mpeg2 decoder (sdtv)</td>
<td>200 fps (IA32)</td>
<td>290 fps (per SPE)</td>
<td>12x</td>
</tr>
<tr>
<td>processing</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Key Performance Characteristics

- Cell's performance is about an order of magnitude better than GPP for media and other applications that can take advantage of its SIMD capability
  - Performance of its simple PPE is comparable to a traditional GPP performance
  - Each SPE is able to perform mostly the same as, or better than, a GPP with SIMD running at the same frequency
  - Key performance advantage comes from its 8 de-coupled SPE SIMD engines with dedicated resources including large register files and DMA channels

- Cell can cover a wide range of application space with its capabilities in
  - Floating point operations
  - Integer operations
  - Data streaming / throughput support
  - Real-time support

- Cell microarchitecture features are exposed to not only its compilers but also its applications
  - Performance gains from tuning compilers and applications can be significant
  - Tools/simulators are provided to assist in performance optimization efforts
Lecture 2

Cell Application Affinity
## Cell Application Affinity – Target Applications

### Cell Broadband Engine
- Non-homogeneous coherent multi-Processor
  - Dual-threaded control-plane processor
  - 8 independent data-plane processors
  - Thread-level parallelism
- SIMD processing architecture
  - 128-entry, 128-bit register files
  - Pipelined execution units
  - Branch hint
  - Data-level parallelism
- Rich integer instruction set
  - Word, halfword, byte, bit
  - Boolean
  - Shuffle
  - Rotate, shift, mask
- Single-precision floating point
- Double-precision floating point
- 256KB SPU local stores
  - Asynchronous DMA/main memory interface
  - Channel interface
  - Single-cycle load/store to/from registers
- High-bandwidth internal bus
  - 96 bytes transferred per clock
  - 100+ outstanding transfers supported
- Coherent bus interface
  - Up to 30GB/s out, 25 GB/s in
  - Direct attach of another Cell
  - Can be configured as non-coherent
- Non-coherent bus interface
  - Up to 10GB/s out, 10 GB/s in
  - 25+ GB/s XDR memory interface

### Accelerated Functions
- Signal processing
- Image processing
- Audio resampling
- Noise generation
- Sound oscillation
- Digital filtering
- Curve and surface evaluation
- FFT
- Matrix mathematics
- Vector mathematics
- Game Physics / Physics simulation
- Video compression / decompression
- Surface subdivision
- Transform-light
- Graphics content creation
- Security encryption / decryption
- Pattern matching
- Language parsing
- TCP/IP offload
- Encoding / decoding
- Parallel processing
- Real time processing
- ...

### Target Applications
- Medical imaging / visualization
- Drug discovery
- Petroleum reservoir modeling
- Seismic analysis
- Avionics
- Air traffic control systems
- Radar systems
- Sonar systems
- Training simulation
- Targeting
- Defense and security IT
- Surveillance
- Secure communications
- LAN/MAN Routers
- Network processing
- XML and SSL acceleration
- Voice and pattern recognition
- Video conferencing
- Computational chemistry
- Climate modeling
- Data mining and analysis
- Media server
- Digital content creation
- Digital content distribution
- ...
Cell Application Affinity – Target Industry Sectors

- **Aerospace & Defense**
  - Signal & Image Processing
  - Security, Surveillance
  - Simulation & Training, …

- **Petroleum Industry**
  - Seismic computing
  - Reservoir Modeling, …

- **Consumer / Digital Media**
  - Digital Content Creation
  - Media Platform
  - Video Surveillance, …

- **Communications Equipment**
  - LAN/MAN Routers
  - Access
  - Converged Networks
  - Security, …

- **Public Sector / Gov’t & Higher Educ.**
  - Signal & Image Processing
  - Computational Chemistry, …

- **Finance**
  - Trade modeling

- **Medical Imaging**
  - CT Scan
  - Ultrasound, …

- **Industrial**
  - Semiconductor / LCD
  - Video Conference
6.189 IAP 2007

Lecture 2

Cell Software Environment
CBE Standards

- **Application Binary Interface Specifications**
  - Defines such things as data types, register usage, calling conventions, and object formats to ensure compatibility of code generators and portability of code
    - SPE ABI
    - Linux for CBE Reference Implementation ABI

- **SPE C/C++ Language Extensions**
  - Defines standardized data types, compiler directives, and language intrinsics used to exploit SIMD capabilities in the core
  - Data types and Intrinsics styled to be similar to Altivec/VMX

- **SPE Assembly Language Specification**
System Level Simulator

- Cell BE – full system simulator
  - Uni-Cell and multi-Cell simulation
  - User Interfaces – TCL and GUI
  - Cycle accurate SPU simulation (pipeline mode)
  - Emitter facility for tracing and viewing simulation events
SW Stack in Simulation

Application Source Code

Programming Tools

Programming Model | OpenMP | MPI

Compilers

Executables

Runtime and libraries

System Software: Hypervisor, Linux/PPC or K42

CellSim: Simulation of hardware

Traces

Dr. Michael Perrone, IBM.
Cell Simulator Debugging Environment
Linux on CBE

- Provided as patched to the 2.6.15 PPC64 Kernel
  - Added heterogeneous lwp/thread model
    - SPE thread API created (similar to pthreads library)
    - User mode direct and indirect SPE access models
    - Full pre-emptive SPE context management
    - spe_ptrace() added for gdb support
    - spe_schedule() for thread to physical SPE assignment
      - currently FIFO – run to completion
  - SPE threads share address space with parent PPE process (through DMA)
    - Demand paging for SPE accesses
    - Shared hardware page table with PPE
  - PPE proxy thread allocated for each SPE thread to:
    - Provide a single namespace for both PPE and SPE threads
    - Assist in SPE initiated C99 and POSIX-1 library services
  - SPE Error, Event and Signal handling directed to parent PPE thread
  - SPE elf objects wrapped into PPE shared objects with extended gld
  - All patches for Cell in architecture dependent layer (subtree of PPC64)
CBE Extensions to Linux

**Programming Models Offered:**
- RPC, Device Subsystem, Direct/Indirect Access
- Heterogenous Threads -- Single SPU, SPU Groups, Shared Memory

**SPE Management Runtime Library**
- (32-bit)
- Standard PPC32 elf interp

**SPE Object Loader Services**
- Standard PPC64 elf interp

**32-bit GNU Libs (glibc, etc)**
- ILP32 Processes

**64-bit GNU Libs (glibc)**
- LP64 Processes

**64-bit Linux Kernel**
- Cell BE Architecture Specific Code
- Multi-large page, SPE event & fault handling, IIC & IOMMU support

**Cell Reference System Hardware**

**Cell32 Workloads**

**Cell64 Workloads**

**PPC32 Apps.**

**PPC64 Apps.**

**Dr. Michael Perrone, IBM.**
SPE Management Library

- **SPEs are exposed as threads**
  - SPE thread model interface is similar to POSIX threads.
  - SPE thread consists of the local store, register file, program counter, and MFC-DMA queue
  - Associated with a single Linux task
  - Features include:
    - Threads - create, groups, wait, kill, set affinity, set context
    - Thread Queries - get local store pointer, get problem state area pointer, get affinity, get context
    - Groups - create, set group defaults, destroy, memory map/unmap, madvise
    - Group Queries - get priority, get policy, get threads, get max threads per group, get events
    - SPE image files - opening and closing

- **SPE Executable**
  - Standalone SPE program managed by a PPE executive
  - Executive responsible for loading and executing SPE program
    - It also services assisted requests for I/O (eg, fopen, fwrite, fprintf) and memory requests (eg, mmap, shmat, ...)

Dr. Michael Perrone, IBM.
Optimized SPE and Multimedia Extension Libraries

- Standard SPE C library subset
  - optimized SPE C99 functions including stdlib c lib, math and etc.
  - subset of POSIX.1 Functions – PPE assisted
- Audio resample - resampling audio signals
- FFT - 1D and 2D fft functions
- gmath - mathematic functions optimized for gaming environment
- image - convolution functions
- intrinsics - generic intrinsic conversion functions
- large-matrix - functions performing large matrix operations
- matrix - basic matrix operations
- mpm - multi-precision math functions
- noise - noise generation functions
- oscillator - basic sound generation functions
- sim – simulator only function including print, profile checkpoint, socket I/O, etc …
- surface - a set of bezier curve and surface functions
- sync - synchronization library
- vector - vector operation functions
Sample Source

- cesof - the samples for the CBE embedded SPU object format usage
- spu_clean - cleans SPU register and local store
- spu_entry - sample SPU entry function (crt0)
- spu_interrupt - SPU first level interrupt handler sample
- spulet - direct invocation of a spu program from Linux shell
- sync
- simpleDMA / DMA
- tutorial - example source code from the tutorial
- SDK test suite
Workloads

- FFT16M – optimized 16 M point complex FFT
- Oscillator - audio signal generator
- Matrix Multiply – matrix multiplication workload
- VSE_subdiv - variable sharpness subdivision algorithm
Bring up Workloads / Demos

- Numerous code samples provided to demonstrate system design constructs
- Complex workloads and demos used to evaluate and demonstrate system performance

- Terrain Rendering Engine
- Subdivision Surfaces
- Physics Simulation
- Geometry Engine
Code Development Tools

● GNU based binutils
  ■ From Sony Computer Entertainment
  ■ gas SPE assembler
  ■ gld SPE ELF object linker
    – ppu-embedspu script for embedding SPE object modules in PPE executables
  ■ Miscellaneous bin utils (ar, nm, ...) targeting SPE modules

● GNU based C/C++ compiler targeting SPE
  ■ From Sony Computer Entertainment
  ■ Retargeted compiler to SPE
  ■ Supports common SPE Language Extensions and ABI (ELF/Dwarf2)

● Cell Broadband Engine Optimizing Compiler (executable)
  ■ IBM XLC C/C++ for PowerPC (Tobey)
  ■ IBM XLC C retargeted to SPE assembler (including vector intrinsics)
    – Highly optimizing
  ■ Prototype CBE Programmer Productivity Aids
    – Auto-Vectorization (auto-SIMD) for SPE and PPE Multimedia Extension code
  ■ Timing Analysis Tool
Bringup Debug Tools

- GNU gdb
  - Multicore Application source level debugger supporting
    - PPE multithreading
    - SPE multithreading
    - Interacting PPE and SPE threads
  - Three modes of debugging SPU threads
    - Standalone SPE debugging
    - Attach to SPE thread
      - Thread ID output when SPU_DEBUG_START=1
SPE Performance Tools (executables)

- **Static analysis (spu_timing)**
  - Annotates assembly source with instruction pipeline state

- **Dynamic analysis (CBE System Simulator)**
  - Generates statistical data on SPE execution
    - Cycles, instructions, and CPI
    - Single/Dual issue rates
    - Stall statistics
    - Register usage
    - Instruction histogram
Miscellaneous Tools – IDL Compiler

Written by programmer

Development Environment

PPE application

.idl

SPE function

IDL Compiler

PPE Compiler

spe_stub.c

stub.h

spe_stub.c

PPE Compiler

SPE Compiler

Generated by IDL Compiler

Call @ run-time

SPE Compiler

SPE binary

PPE binary

Generated by IDL Compiler

Call @ run-time
6.189 IAP 2007

Lecture 2

Cell Software Development Considerations
CELL Software Design Considerations

- Four Levels of Parallelism
  - Blade Level: Two Cell processors per blade
  - Chip Level: 9 cores run independent tasks
  - Instruction level: Dual issue pipelines on each SPE
  - Register level: Native SIMD on SPE and PPE VMX
- 256KB local store per SPE: data + code + stack
- Communication
  - DMA and Bus bandwidth
    - DMA granularity – 128 bytes
    - DMA bandwidth among LS and System memory
  - Traffic control
    - Exploit computational complexity and data locality to lower data traffic requirement
  - Shared memory / Message passing abstraction overhead
  - Synchronization
  - DMA latency handling
Typical CELL Software Development Flow

- Algorithm complexity study
- Data layout/locality and Data flow analysis
- Experimental partitioning and mapping of the algorithm and program structure to the architecture
- Develop PPE Control, PPE Scalar code
- Develop PPE Control, partitioned SPE scalar code
  - Communication, synchronization, latency handling
- Transform SPE scalar code to SPE SIMD code
- Re-balance the computation / data movement
- Other optimization considerations
  - PPE SIMD, system bottleneck, load balance
The First Generation Cell Blade

1GB XDR Memory  Cell Processors  IO Controllers  IBM Blade Center interface

Dr. Michael Perrone, IBM.
Cell Blade Overview

- **Blade**
  - Two Cell BE Processors
  - 1GB XDRAM
  - BladeCenter Interface (Based on IBM JS20)

- **Chassis**
  - Standard IBM BladeCenter form factor with:
    - 7 Blades (for 2 slots each) with full performance
    - 2 switches (1Gb Ethernet) with 4 external ports each
  - Updated Management Module Firmware.
  - External Infiniband Switches with optional FC ports

- **Typical Configuration (available today from E&TS)**
  - eServer 25U Rack
  - 7U Chassis with Cell BE Blades, OpenPower 710
  - Nortel GbE switch
  - GCC C/C++ (Barcelona) or XLC Compiler for Cell (alphaworks)
  - SDK Kit on http://www-128.ibm.com/developerworks/power/cell/
Summary

- Cell ushers in a new era of leading edge processors optimized for digital media and entertainment
- Desire for realism is driving a convergence between supercomputing and entertainment
- New levels of performance and power efficiency beyond what is achieved by PC processors
- Responsiveness to the human user and the network are key drivers for Cell
- Cell will enable entirely new classes of applications, even beyond those we contemplate today
Special Notices

© Copyright International Business Machines Corporation 2006
All Rights Reserved

This document was developed for IBM offerings in the United States as of the date of publication. IBM may not make these offerings available in other countries, and the information is subject to change without notice. Consult your local IBM business contact for information on the IBM offerings available in your area. In no event will IBM be liable for damages arising directly or indirectly from any use of the information contained in this document.

Information in this document concerning non-IBM products was obtained from the suppliers of these products or other public sources. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. Send license inquiries, in writing, to IBM Director of Licensing, IBM Corporation, New Castle Drive, Armonk, NY 10504-1785 USA.

All statements regarding IBM future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. The information contained in this document has not been submitted to any formal IBM test and is provided "AS IS" with no warranties or guarantees either expressed or implied.

All examples cited or described in this document are presented as illustrations of the manner in which some IBM products can be used and the results that may be achieved. Actual environmental costs and performance characteristics will vary depending on individual client configurations and conditions.

IBM Global Financing offerings are provided through IBM Credit Corporation in the United States and other IBM subsidiaries and divisions worldwide to qualified commercial and government clients. Rates are based on a client's credit rating, financing terms, offering type, equipment type and options, and may vary by country. Other restrictions may apply. Rates and offerings are subject to change, extension or withdrawal without notice.

IBM is not responsible for printing errors in this document that result in pricing or information inaccuracies. All prices shown are IBM's United States suggested list prices and are subject to change without notice; reseller prices may vary.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply. Many of the features described in this document are operating system dependent and may not be available on Linux. For more information, please check: http://www.ibm.com/systems/p/software/whitepapers/linux_overview.html

Any performance data contained in this document was determined in a controlled environment. Actual results may vary significantly and are dependent on many factors including system hardware configuration and software design and configuration. Some measurements quoted in this document may have been made on development-level systems. There is no guarantee these measurements will be the same on generally-available systems. Some measurements quoted in this document may have been estimated through extrapolation. Users of this document should verify the applicable data for their specific environment.
Special Notices (Cont.) -- Trademarks


A full list of U.S. trademarks owned by IBM may be found at: http://www.ibm.com/legal/copytrade.shtml.

Cell Broadband Engine and Cell Broadband Engine Architecture are trademarks of Sony Computer Entertainment, Inc. in the United States, other countries, or both.
Rambus is a registered trademark of Rambus, Inc.
XDR and FlexIO are trademarks of Rambus, Inc.
UNIX is a registered trademark in the United States, other countries or both.
Linux is a trademark of Linus Torvalds in the United States, other countries or both.
Fedora is a trademark of Redhat, Inc.
Microsoft, Windows, Windows NT and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries or both.
Intel, Intel Xeon, Itanium and Pentium are trademarks or registered trademarks of Intel Corporation in the United States and/or other countries.
AMD Opteron is a trademark of Advanced Micro Devices, Inc.
Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States and/or other countries.
TPC-C and TPC-H are trademarks of the Transaction Performance Processing Council (TPPC).
SPECint, SPECfp, SPECjbb, SPECweb, SPECjAppServer, SPEC OMP, SPECviewperf, SPECapc, SPECchpc, SPECjvm, SPECmail, SPECimap and SPECsfs are trademarks of the Standard Performance Evaluation Corp (SPEC).
AltiVec ™ is a trademark of Freescale Semiconductor, Inc.
PCI-X and PCI Express are registered trademarks of PCI SIG.
InfiniBand™ is a trademark the InfiniBand® Trade Association
Other company, product and service names may be trademarks or service marks of others.

Revised July 23, 2006
SPE Highlights

- **RISC like organization**
  - 32 bit fixed instructions
  - Clean design – unified Register file

- **User-mode architecture**
  - No translation/protection within SPU
  - DMA is full Power Arch protect/x-late

- **VMX-like SIMD dataflow**
  - Broad set of operations (8 / 16 / 32 Byte)
  - Graphics SP-Float
  - IEEE DP-Float

- **Unified register file**
  - 128 entry x 128 bit

- **256KB Local Store**
  - Combined I & D
  - 16B/cycle L/S bandwidth
  - 128B/cycle DMA bandwidth

14.5mm² (90nm SOI)
What is a Synergistic Processor? (and why is it efficient?)

- Local Store “is” large 2nd level register file / private instruction store instead of cache
  - Asynchronous transfer (DMA) to shared memory
  - Frontal attack on the Memory Wall
- Media Unit turned into a Processor
  - Unified (large) Register File
  - 128 entry x 128 bit
- Media & Compute optimized
  - One context
  - SIMD architecture
SPU Details

- **Synergistic Processor Element (SPE)**
- **User-mode architecture**
  - No translation/protection within SPE
  - DMA is full PowerPC protect/xlate
- **Direct programmer control**
  - DMA/DMA-list
  - Branch hint
- **VMX-like SIMD dataflow**
  - Graphics SP-Float
  - No saturate arith, some byte
  - IEEE DP-Float (BlueGene-like)
- **Unified register file**
  - 128 entry x 128 bit
- **256KB Local Store**
  - Combined I & D
  - 16B/cycle L/S bandwidth
  - 128B/cycle DMA bandwidth
- **Memory Flow Control (MFC)**

**SPU Units**

- **Simple (FXU even)**
  - Add/Compare
  - Rotate
  - Logical, Count Leading Zero
- **Permute (FXU odd)**
  - Permute
  - Table-lookup
- **FPU (Single / Double Precision)**
- **Control (SCN)**
  - Dual Issue, Load/Store, ECC Handling
- **Channel (SSC)** – Interface to MFC
- **Register File (GPR/FWD)**

**SPU Latencies**

- Simple fixed point - 2 cycles*
- Complex fixed point - 4 cycles*
- Load - 6 cycles*
- Single-precision (ER) float - 6 cycles*
- Integer multiply - 7 cycles*
- Branch miss (no penalty for correct hint) - 20 cycles
- DP (IEEE) float (partially pipelined) - 13 cycles*
- Enqueue DMA Command - 20 cycles*
SPE Block Diagram

- Floating-Point Unit
- Fixed-Point Unit
- Permute Unit
- Load-Store Unit
- Branch Unit
- Channel Unit
- Result Forwarding and Staging
- Register File
- Instruction Issue Unit / Instruction Line Buffer
- Local Store (256kB) Single Port SRAM
- DMA Unit

On-Chip Coherent Bus
- 8 Byte/Cycle → 16 Byte/Cycle → 64 Byte/Cycle → 128 Byte/Cycle

Dr. Michael Perrone, IBM.
SXU Pipeline

IF Instruction Fetch
IB Instruction Buffer
ID Instruction Decode
IS Instruction Issue
RF Register File Access
EX Execution
WB Write Back

Branch Instruction

Permute Instruction

Load/Store Instruction

Fixed Point Instruction

Floating Point Instruction

Dr. Michael Perrone, IBM.
MFC Detail

- Memory Flow Control System
- DMA Unit
  - LS <-> LS, LS<> Sys Memory, LS<> I/O Transfers
  - 8 PPE-side Command Queue entries
  - 16 SPU-side Command Queue entries
- MMU similar to PowerPC MMU
  - 8 SLBs, 256 TLBs
  - 4K, 64K, 1M, 16M page sizes
  - Software/HW page table walk
  - PT/SLB misses interrupt PPE
- Atomic Cache Facility
  - 4 cache lines for atomic updates
  - 2 cache lines for cast out/MMU reload
- Up to 16 outstanding DMA requests in BIU
- Resource / Bandwidth Management Tables
  - Token Based Bus Access Management
  - TLB Locking

- Isolation Mode Support (Security Feature)
- Hardware enforced “isolation”
  - SPU and Local Store not visible (bus or jtag)
  - Small LS “untrusted area” for communication area
- Secure Boot
  - Chip Specific Key
  - Decrypt/Authenticate Boot code
- “Secure Vault” – Runtime Isolation Support
  - Isolate Load Feature
  - Isolate Exit Feature

Legend:
- Data Bus
- Snoop Bus
- Control Bus
- Xlate Ld/St
- MMIO
Per SPE Resources (PPE Side)

Problem State

- 8 Entry MFC Command Queue Interface
- DMA Command and Queue Status
- DMA Tag Status Query Mask
- DMA Tag Status
- 32 bit Mailbox Status and Data from SPU
- 32 bit Mailbox Status and Data to SPU
- 4 deep FIFO
- Signal Notification 1
- Signal Notification 2
- SPU Run Control
- SPU Next Program Counter
- SPU Execution Status

Privileged 1 State (OS)

- SPU Privileged Control
- SPU Channel Counter Initialize
- SPU Channel Data Initialize
- SPU Signal Notification Control
- SPU Decrementer Status & Control
- MFC DMA Control
- MFC Context Save / Restore Registers
- SLB Management Registers

Privileged 2 State (OS or Hypervisor)

- SPU Master Run Control
- SPU ID
- SPU ECC Control
- SPU ECC Status
- SPU ECC Address
- SPU 32-bit PU Interrupt Mailbox
- MFC Interrupt Mask
- MFC Interrupt Status
- MFC DMA Privileged Control
- MFC Command Error Register
- MFC Command Translation Fault Register
- MFC SDR (PT Anchor)
- MFC ACCR (Address Compare)
- MFC DSSR (DSI Status)
- MFC DAR (DSI Address)
- MFC LPID (logical partition ID)
- MFC TLB Management Registers

Dr. Michael Perrone, IBM.
Per SPE Resources (SPU Side)

SPU Direct Access Resources

- 128 - 128 bit GPRs
- External Event Status (Channel 0)
  - Decrementer Event
  - Tag Status Update Event
  - DMA Queue Vacancy Event
  - SPU Incoming Mailbox Event
  - Signal 1 Notification Event
  - Signal 2 Notification Event
  - Reservation Lost Event
- External Event Mask (Channel 1)
- External Event Acknowledgement (Channel 2)
- Signal Notification 1 (Channel 3)
- Signal Notification 2 (Channel 4)
- Set Decrementer Count (Channel 7)
- Read Decrementer Count (Channel 8)
- 16 Entry MFC Command Queue Interface (Channels 16-21)
- DMA Tag Group Query Mask (Channel 22)
- Request Tag Status Update (Channel 23)
  - Immediate
  - Conditional - ALL
  - Conditional - ANY
- Read DMA Tag Group Status (Channel 24)
- DMA List Stall and Notify Tag Status (Channel 25)
- DMA List Stall and Notify Tag Acknowledgement (Channel 26)
- Lock Line Command Status (Channel 27)
- Outgoing Mailbox to PU (Channel 28)
- Incoming Mailbox from PU (Channel 29)
- Outgoing Interrupt Mailbox to PU (Channel 30)

SPU Indirect Access Resources
(via EA Addressed DMA)

- System Memory
- Memory Mapped I/O
- This SPU Local Store
- Other SPU Local Store
- Other SPU Signal Registers
- Atomic Update (Cacheable Memory)
Memory Flow Controller Commands

DMA Commands

**Put** - Transfer from Local Store to EA space
**Puts** - Transfer and Start SPU execution
**Putr** - Put Result - (Arch. Scarf into L2)
**Putl** - Put using DMA List in Local Store
**Putrl** - Put Result using DMA List in LS (Arch)

**Get** - Transfer from EA Space to Local Store
**Gets** - Transfer and Start SPU execution
**Getl** - Get using DMA List in Local Store

**Sndsig** - Send Signal to SPU

Command Modifiers: \(<f,b>\)
- \(f\): Embedded Tag Specific Fence
  - Command will not start until all previous commands in same tag group have completed
- \(b\): Embedded Tag Specific Barrier
  - Command and all subsequent commands in same tag group will not start until previous commands in same tag group have completed

SL1 Cache Management Commands

**sdcrt** - Data cache region touch (DMA Get hint)
**sdcrtst** - Data cache region touch for store (DMA Put hint)
**sdcrz** - Data cache region zero
**sdcrs** - Data cache region store
**sdcrf** - Data cache region flush

Command Parameters

- **LSA** - Local Store Address (32 bit)
- **EA** - Effective Address (32 or 64 bit)
- **TS** - Transfer Size (16 bytes to 16K bytes)
- **LS** - DMA List Size (8 bytes to 16 K bytes)
- **TG** - Tag Group (5 bit)
- **CL** - Cache Management / Bandwidth Class

Synchronization Commands

**Lockline** (Atomic Update) Commands:
- **getllar** - DMA 128 bytes from EA to LS and set Reservation
- **putllc** - Conditionally DMA 128 bytes from LS to EA
- **putlluc** - Unconditionally DMA 128 bytes from LS to EA
- **barrier** - all previous commands complete before subsequent commands are started
- **mfcsync** - Results of all previous commands in Tag group are remotely visible
- **mfceieio** - Results of all preceding Puts commands in same group visible with respect to succeeding Get commands
SPE Structure

● Scalar processing supported on data-parallel substrate
  ■ All instructions are data parallel and operate on vectors of elements
  ■ Scalar operation defined by instruction use, not opcode
    – Vector instruction form used to perform operation

● Preferred slot paradigm
  ■ Scalar arguments to instructions found in “preferred slot”
  ■ Computation can be performed in any slot
Register Scalar Data Layout

- **Preferred slot in bytes 0-3**
  - By convention for procedure interfaces
  - Used by instructions expecting scalar data
    - Addresses, branch conditions, generate controls for insert
Element Interconnect Bus

- **EIB data ring for internal communication**
  - Four 16 byte data rings, supporting multiple transfers
  - 96B/cycle peak bandwidth
  - Over 100 outstanding requests
Element Interconnect Bus – Command Topology

- “Address Concentrator” tree structure minimizes wiring resources
- Single serial command reflection point (AC0)
- Address collision detection and prevention
- Fully pipelined
- Content-aware round robin arbitration
- Credit-based flow control
Element Interconnect Bus – Data Topology

- Four 16B data rings connecting 12 bus elements
  - Two clockwise / Two counter-clockwise
- Physically overlaps all processor elements
- Central arbiter supports up to three concurrent transfers per data ring
  - Two stage, dual round robin arbiter
- Each element port simultaneously supports 16B in and 16B out data path
  - Ring topology is transparent to element data interface

![Diagram of Element Interconnect Bus – Data Topology]
Internal Bandwidth Capability

- Each EIB Bus data port supports 25.6GBytes/sec* in each direction.
- The EIB Command Bus streams commands fast enough to support 102.4 GB/sec for coherent commands, and 204.8 GB/sec for non-coherent commands.
- The EIB data rings can sustain 204.8GB/sec for certain workloads, with transient rates as high as 307.2GB/sec between bus units.

Despite all that available bandwidth…

* The above numbers assume a 3.2GHz core frequency – internal bandwidth scales with core frequency.
Example of Eight Concurrent Transactions