Cell Broadband Engine Overview

Course Code: L1T1H1-02
Cell Ecosystem Solutions Enablement

Class Objectives – Things you will learn

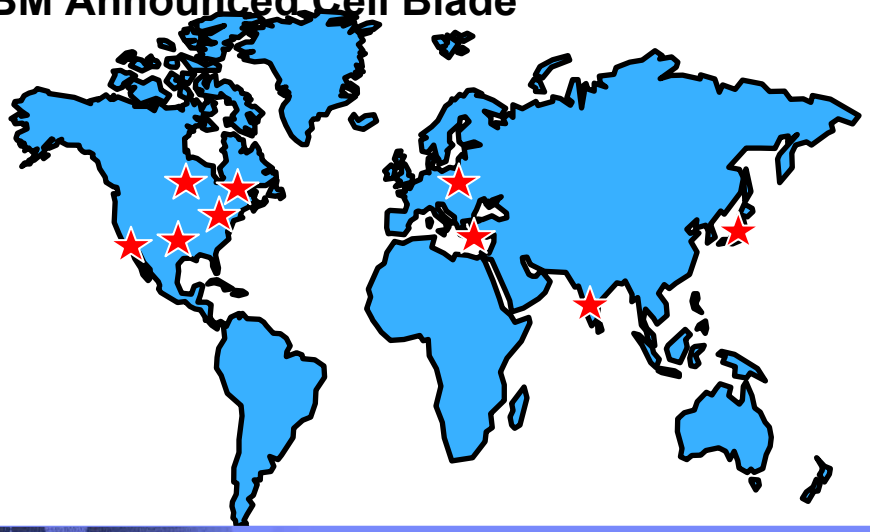
- **An overview of**
  - Cell history
  - Cell microprocessor highlights
  - Hardware architecture and components
  - Software development environment including standard interfaces, programming models, operating system runtime strategy, system simulator, development tools, …
  - Cell performance characteristics
  - Cell blade server
  - Cell application affinity and target opportunities

Class Agenda

- **Introduction to Cell**
- **Cell Hardware Overview**
  - Cell highlights
  - Cell processor
  - Cell processor components
- **Cell Software Overview**
  - Cell software environment
  - Application development overview
  - Cell programming overview
  - Cell software design considerations
  - Development tools
  - Cell system simulator
  - Optimized libraries
- **Cell performance characteristics**
- **Cell blade**
- **Cell application affinity**

Cell History

- **IBM, SCEI/Sony, Toshiba Alliance formed in 2000**
- **Design Center opened in March 2001**
  - Based in Austin, Texas
- **Single CellBE 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**



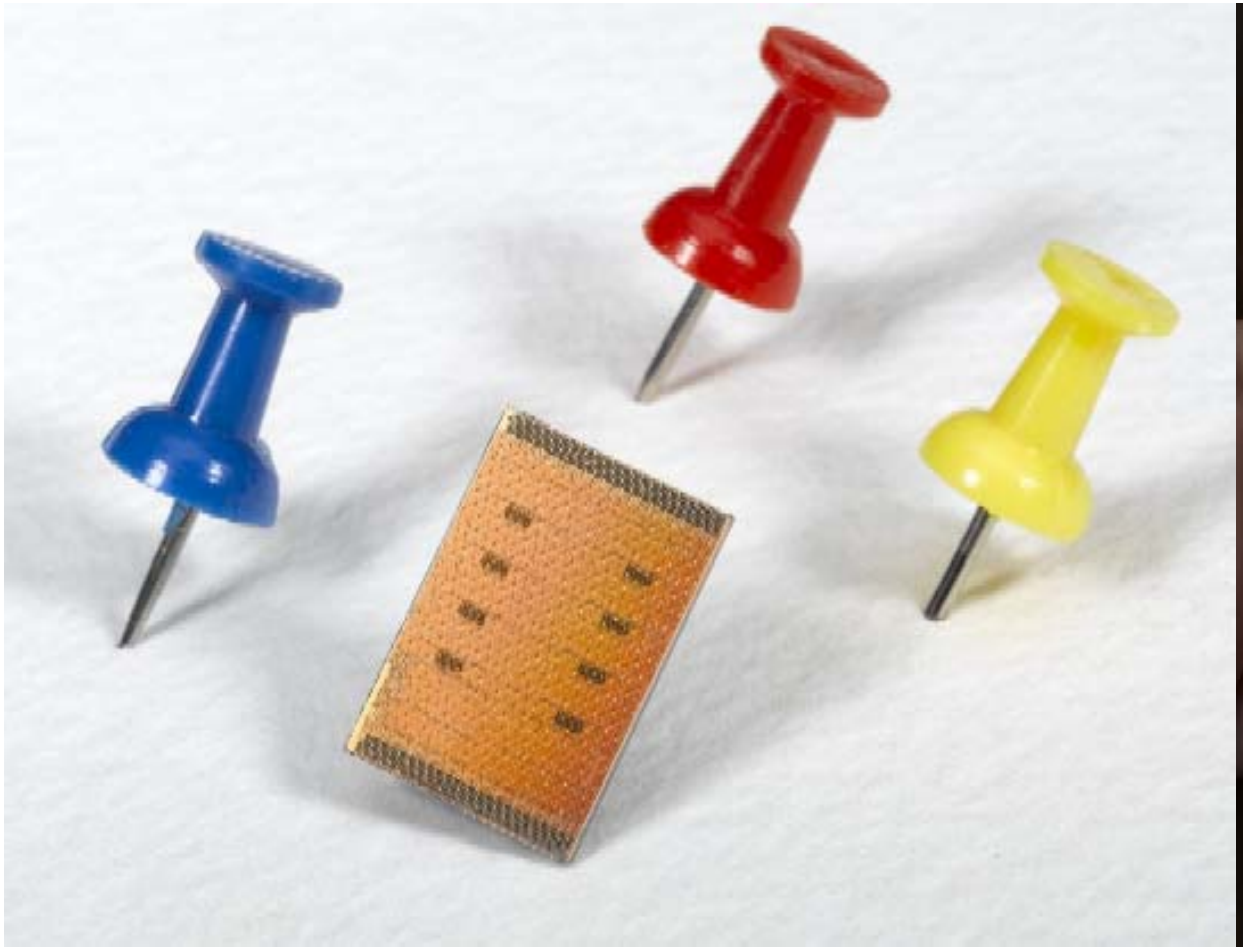
SONY COMPUTER ENTERTAINMENT

SONY

TOSHIBA

IBM

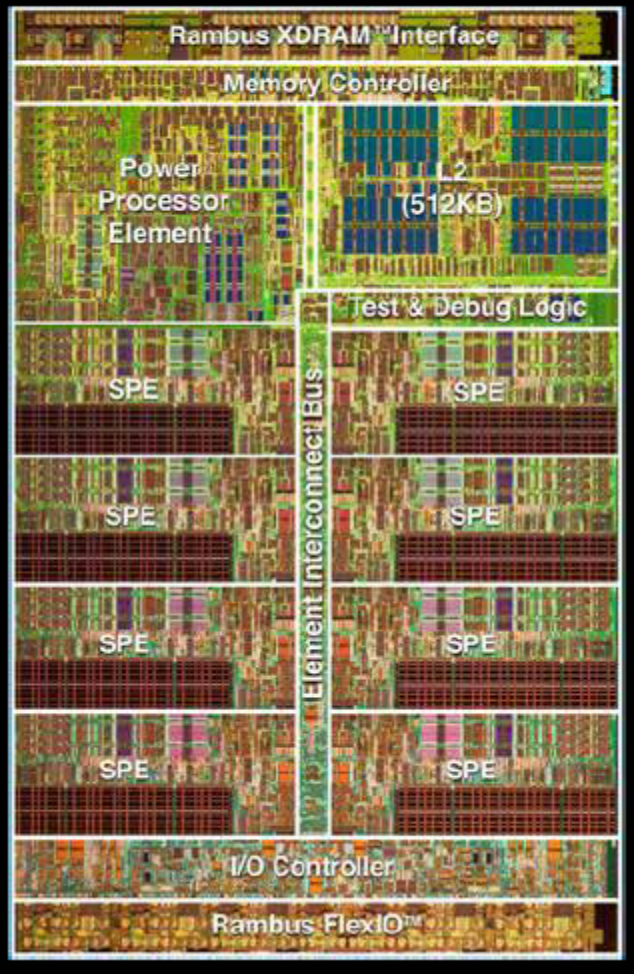
Cell



Cell Hardware Overview

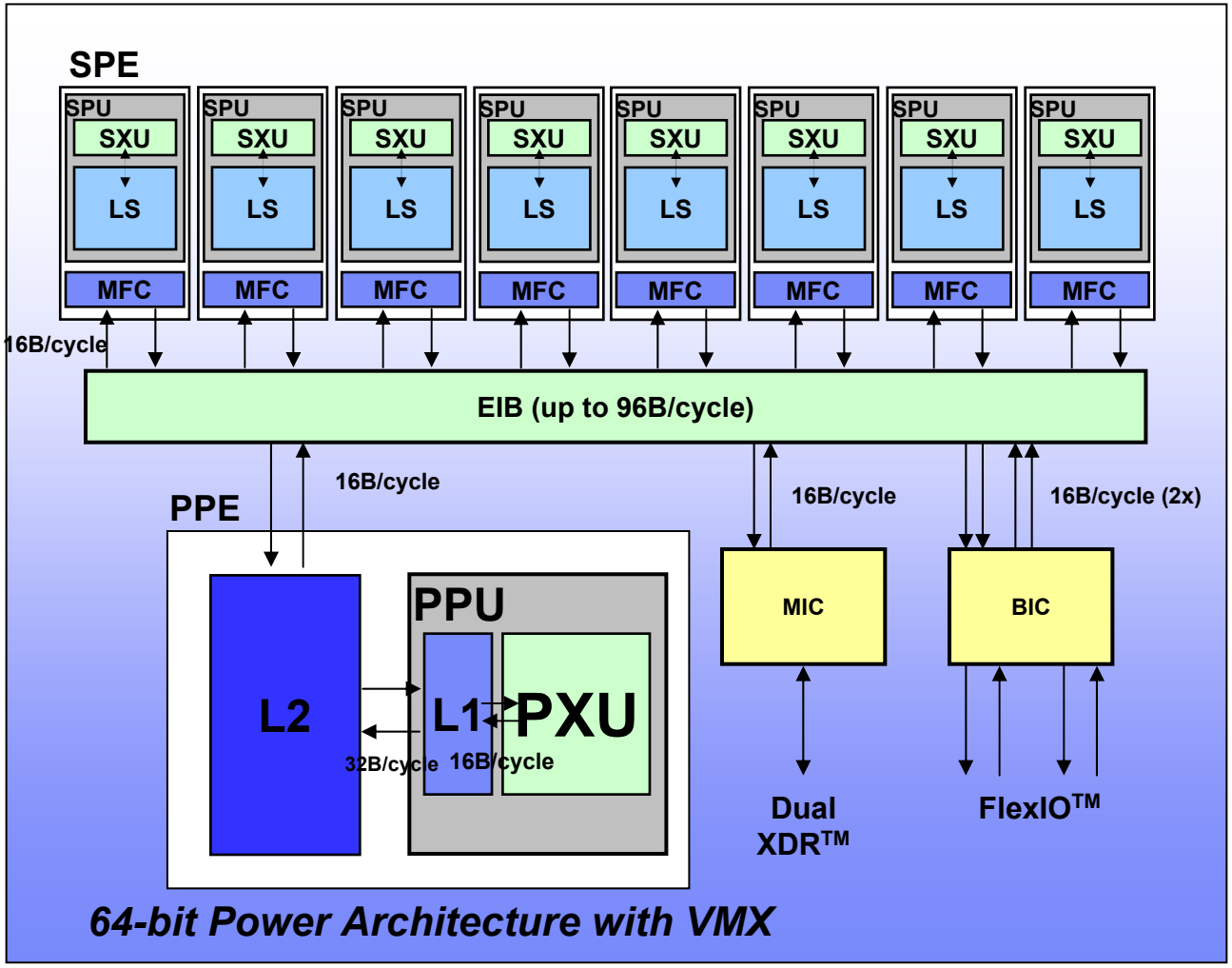
Highlights (3.2 GHz)

- 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)



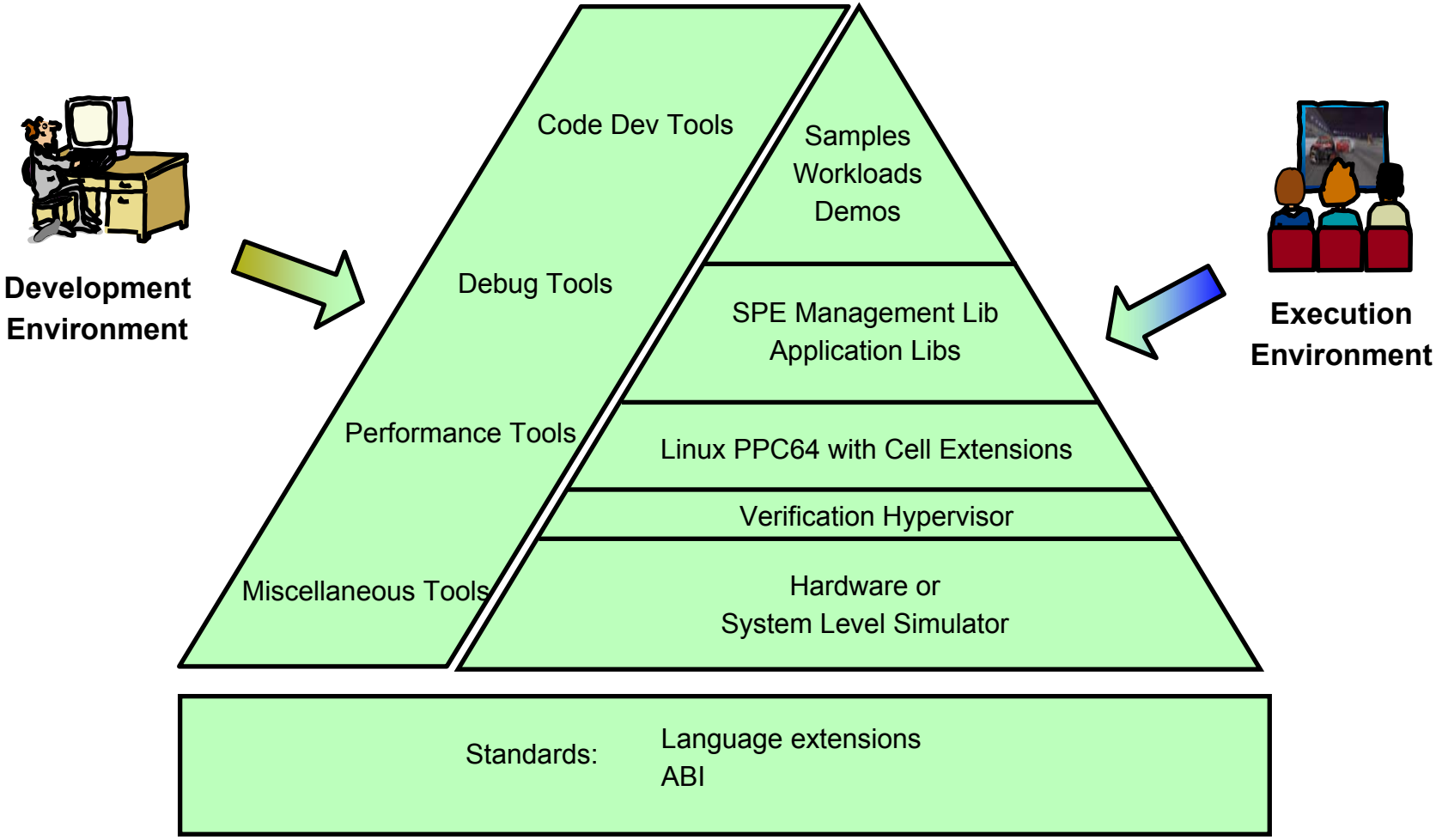
Cell Features

- **Heterogeneous multi-core 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



Software Overview

Cell Software Environment



Cell Standards

Application Binary Interface Specifications

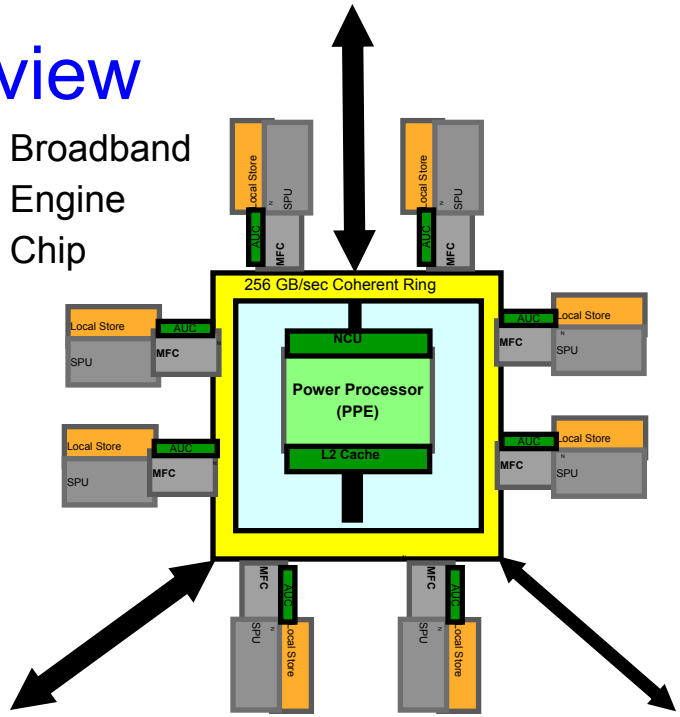
- Defines such things as data types, register usage, calling conventions, and object formats to ensure compatibility of code generators and portability of code.
  - SPE ABI
  - Linux Cell ABI

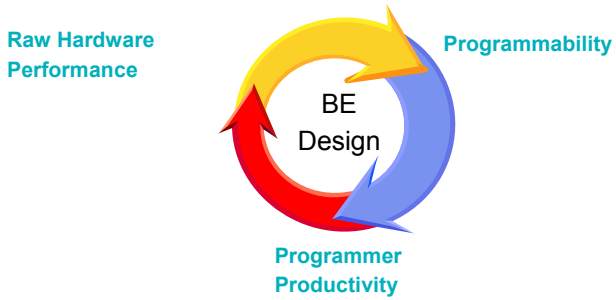
- **SPE C/C++ Language Extensions**
  - Defines standardized data types, compiler directives, and language intrinsics used to exploit SIMD capabilities in the core.
  - Data types and Intrinsics styled to be similar to Altivec/VMX.

- **SPE Assembly Language Specification**

Application Development Overview

- CellBE Programming Features
- Flexible Program Models
  - Application Accelerator Model
  - Function Offload Model
  - Computation Acceleration
  - Heterogeneous Multi-Threading

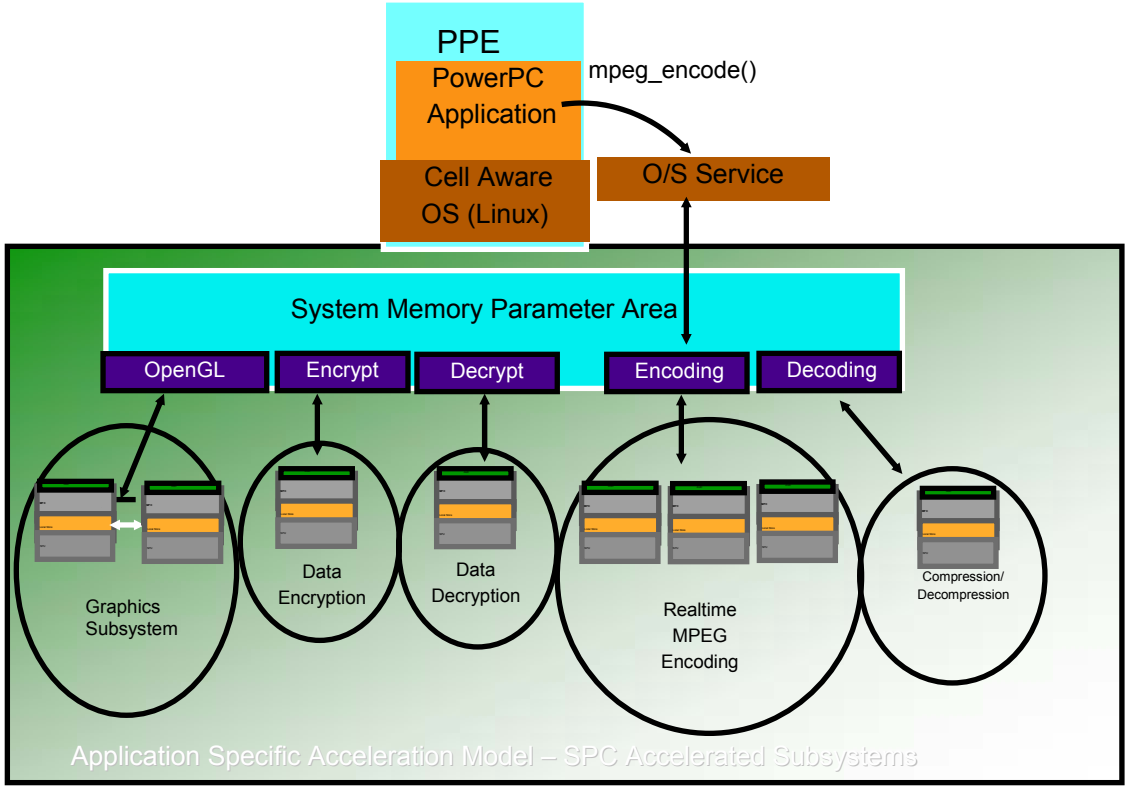




Programming Models

Application Specific Accelerators

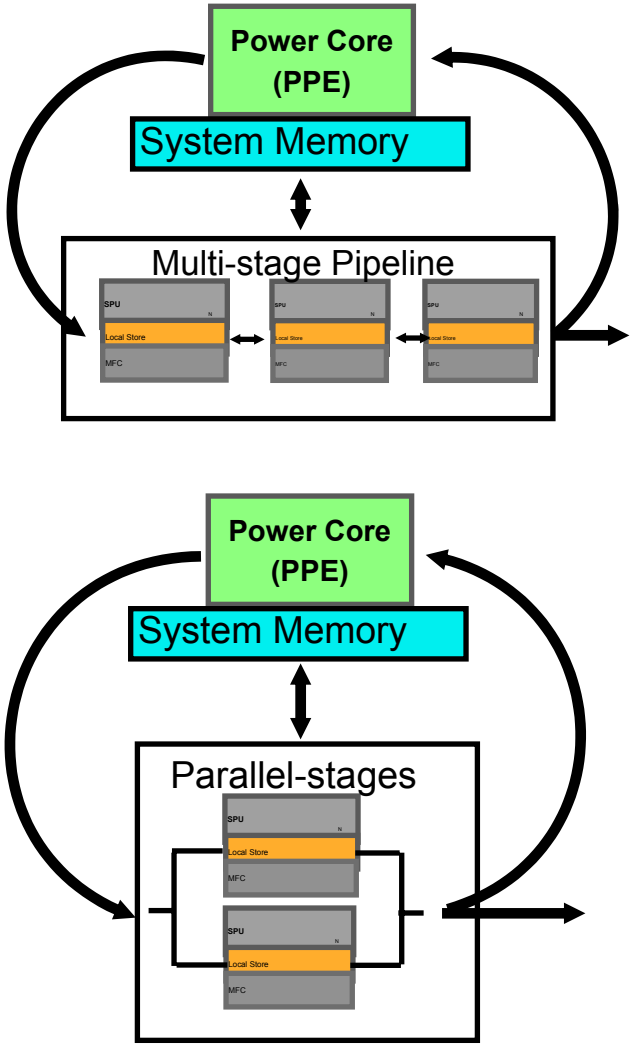
- Acceleration provided by OS or application libraries
- Application portability maintained with platform specific libraries



Subsystem Programming Model

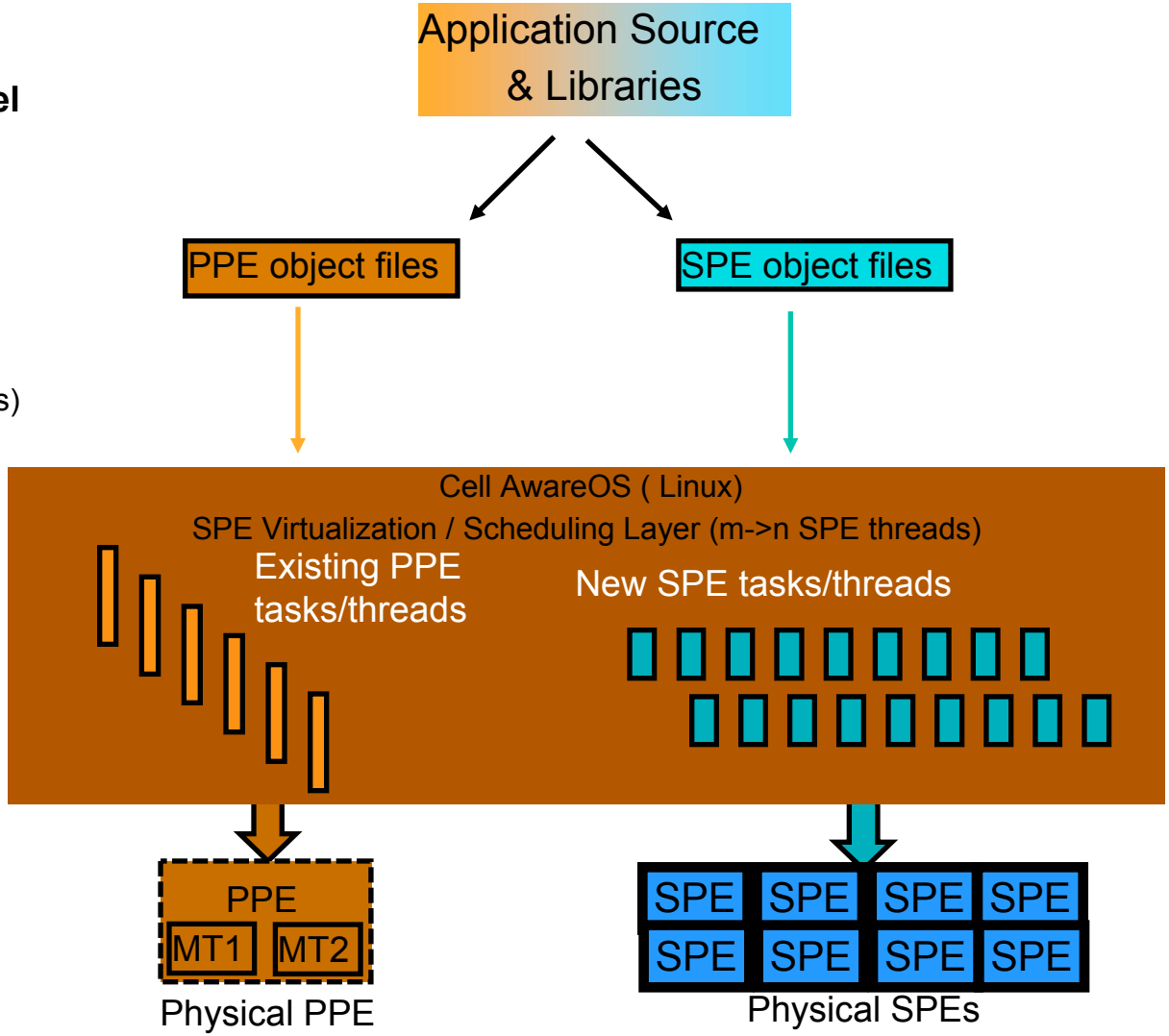
Function Offload

- Dedicated Function (problem/privileged subsystem)
  - Programmer writes/uses SPU "libraries"
    - ➔Graphics Pipeline
    - ➔Audio Processing
    - ➔MPEG Encoding/Decoding
    - ➔Encryption / Decryption
  - Main Application in PPE, invokes SPU bound services
    - ➔RPC Like Function Call
    - ➔I/O Device Like Interface (FIFO/ Command Queue)
  - 1 or more SPUs cooperating in subsystem
    - ➔Problem State (Application Allocated)
    - ➔Privileged State (OS Allocated)
  - Code-to-data or data-to-code pipelining possible
  - Very efficient in real-time data streaming applications



Operating System Runtime Strategy

- **Heterogeneous Multi-Threading Model**
  - PPE Threads, SPE Threads
  - SPE DMA EA = PPE Process EA Space
  - OS supports Create/Destroy SPE tasks
  - Atomic Update Primitives used for Mutex
  - SPE Context Fully Managed
    - Context Save/Restore for Debug
    - Virtualization Mode (indirect access)
    - Direct Access Mode (realtime)
  - OS assignment of SPE threads to SPEs
    - Programmer directed using affinity mask
  - SPE Compilers use SPE Management Lib.



CELL Software Design Considerations

- **Two Levels of Parallelism**
  - Regular vector data that is SIMD-able
  - Independent tasks that may be executed in parallel
- **Computational**
  - SIMD engines on 8 SPEs and 1 PPE
  - Parallel sequence to be distributed over 8 SPE / 1 PPE
  - 256KB local store per SPE usage (data + code)
- **Communicational**
  - 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 bottle-neck, load balance

Development Tools

Code Development Tools

- **GNU based binutils**
  - gas SPE assembler
  - gld SPE ELF object linker
    - gld extensions for embedding SPE object modules in PPE executables
  - misc bin utils (ar, nm, ...) targeting SPE modules
  - hosted on Linux IA32, Linux PowerPC
- **GNU based C/C++ compiler targeting SPE**
  - From STI Partner
  - retargeted compiler to SPE
  - Supports common SPE Language Extensions and ABI (ELF/Dwarf2) object output
- **Cell Broadband Engine Optimizing Compiler (IBM Proprietary)**
  - Based on the highly optimizing IBM XL C/C++ for PowerPC
  - IBM XL C retargeted to generate SPE assembler code (including vector intrinsics) - highly optimizing
  - Prototype – XL C Compiler supporting CellBE Programmer Productivity Aids
    - Single Source compilation using OpenMP directives (PPE and SPE object code generated)
    - Auto-Vectorization (auto-SIMD) for VMX and SPE
    - Auto-Parallelization across SPEs
    - Local Store software managed caching model
  - Hosted on Linux/x86, Linux on Power, and Windows/x86
  - An alpha version of IBM XL C for CBE hosted on Linux/x86 is available on IBM Alphaworks
    - C language support for PPE and SPE
    - C++ language support for PPE

Debug Tools

- **CellBE system simulator**
  - Executable availability on AlphaWorks
- **GNU gdb**
  - ptrace and spe_ptrace enabled
  - Multi-core Application source level debugger supporting PPE multithreading, SPE multithreading, interacting PPE and SPE threads
  - Three modes of debugging SPU threads
    - Attach to SPE thread
    - Launch mode – launch a new debug session for each SPE thread
    - Pass-thru mode – follow execution into SPE thread
- **RISCwatch**
  - Low level hardware (JTAG) debugger

Prototype Performance Tools

- **pmcount**
  - Tool to access to HW performance counters
- **Performance inspector**
  - Suite of GPL based performance analysis tools extended to support SPE threads
    - tprof – timer based analysis tool
    - ptt – per thread time
    - ai – above idle
    - post – report generator
    - a2n – address to name
- **Oprofile**
  - System level profiler

SPE Performance Tools

- **Static analysis (spexlc_timing)**
  - Annotates assembly source with instruction pipeline state
- **Dynamic analysis (CellBE System Simulator)**
  - Generates statistical data on SPE execution
    - Cycles, instructions, and CPI
    - Single/Dual issue rates
    - Stall statistics
    - Register usage
    - Instruction histogram

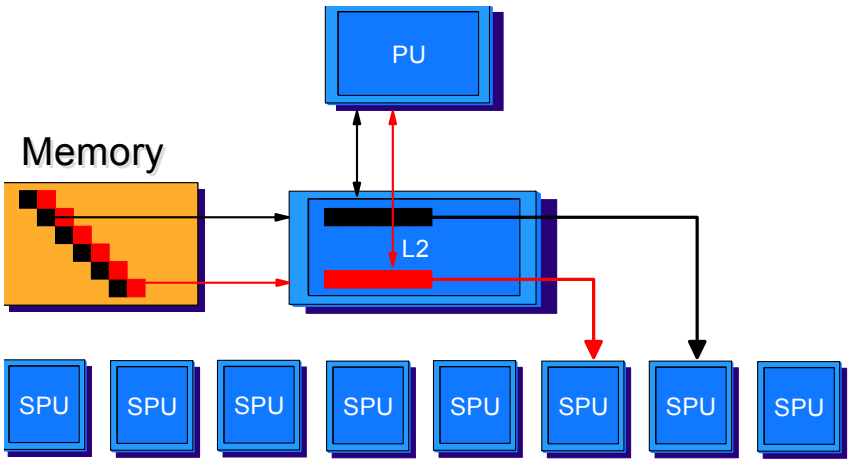
Cell Performance Characteristics

Why Cell processor is so fast?

Key Architectural Reasons

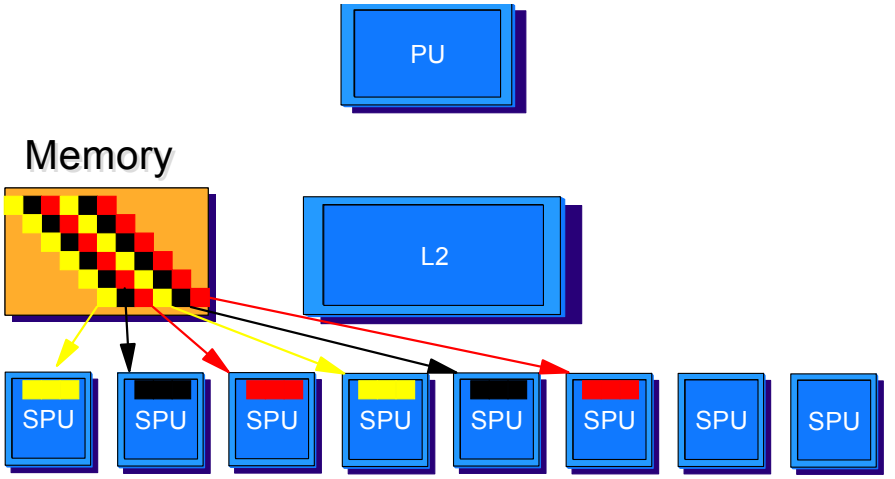
- Parallel processing inside chip
- Fully parallelized and concurrent operations
- Functional offloading
- High frequency design
- High bandwidth for memory and IO accesses
- Fine tuning for data transfer

Staging



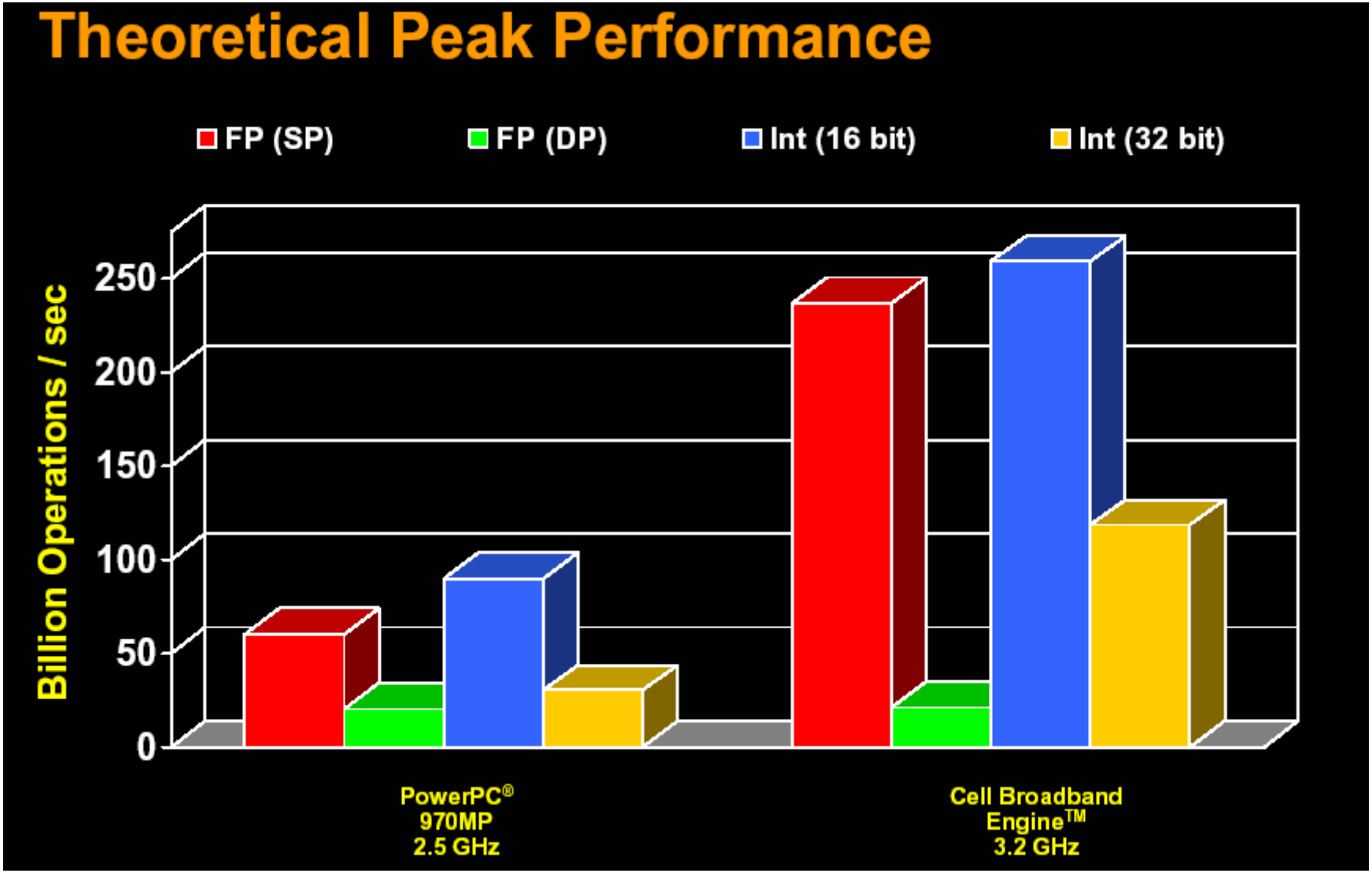
L2 - 4 outstanding loads + 2 prefetch

Data



SPU - 16 outstanding loads per SPU

Theoretical Peak Performance



Cell BE Performance Summary

Type	Algorithm	3.2 GHz GPP	3.2 GHz Cell	Cell Perf Advantage
HPC	Matrix Multiplication (S.P.)	24 Gflops (w/SIMD)	200 GFlops* (8SPEs)	8x
	Linpack (S.P.)	16 GFlops (w/SIMD)	156 GFlops* (8SPEs)	9x
	Linpack (D.P.): 1kx1k matrix	7.2 GFlops (IA32/SSE3)	9.67 GFLops* (8SPEs)	1.3x
graphics	Transform-light	170 MVPS (G5/VMX)	256 MVPS** (per SPE)	12x
	TRE	1 fps (G5/VMX)	30 fps* (Cell)	30x
security	AES encryp. 128-bit key	1.03 Gbps	2.06Gbps** (per SPE)	16x
	AES decryp. 128-bit key	1.04 Gbps	1.5Gbps** (per SPE)	11x
	TDES	0.12 Gbps	0.16 Gbps** (per SPE)	10x
	DES	0.43 Gbps	0.49 Gbps** (per SPE)	9x
	SHA-1	0.85 Gbps	1.98 Gbps** (per SPE)	18x
video processing	mpeg2 decoder (CIF)	----	1267 fps* (per SPE)	--
	mpeg2 decoder (SDTV)	354 fps (IA32)	365 fps** (per SPE)	8x
	mpeg2 decoder (HDTV)	----	73 fps* (per SPE)	--

Notes: * Hardware measurement ** Simulation results

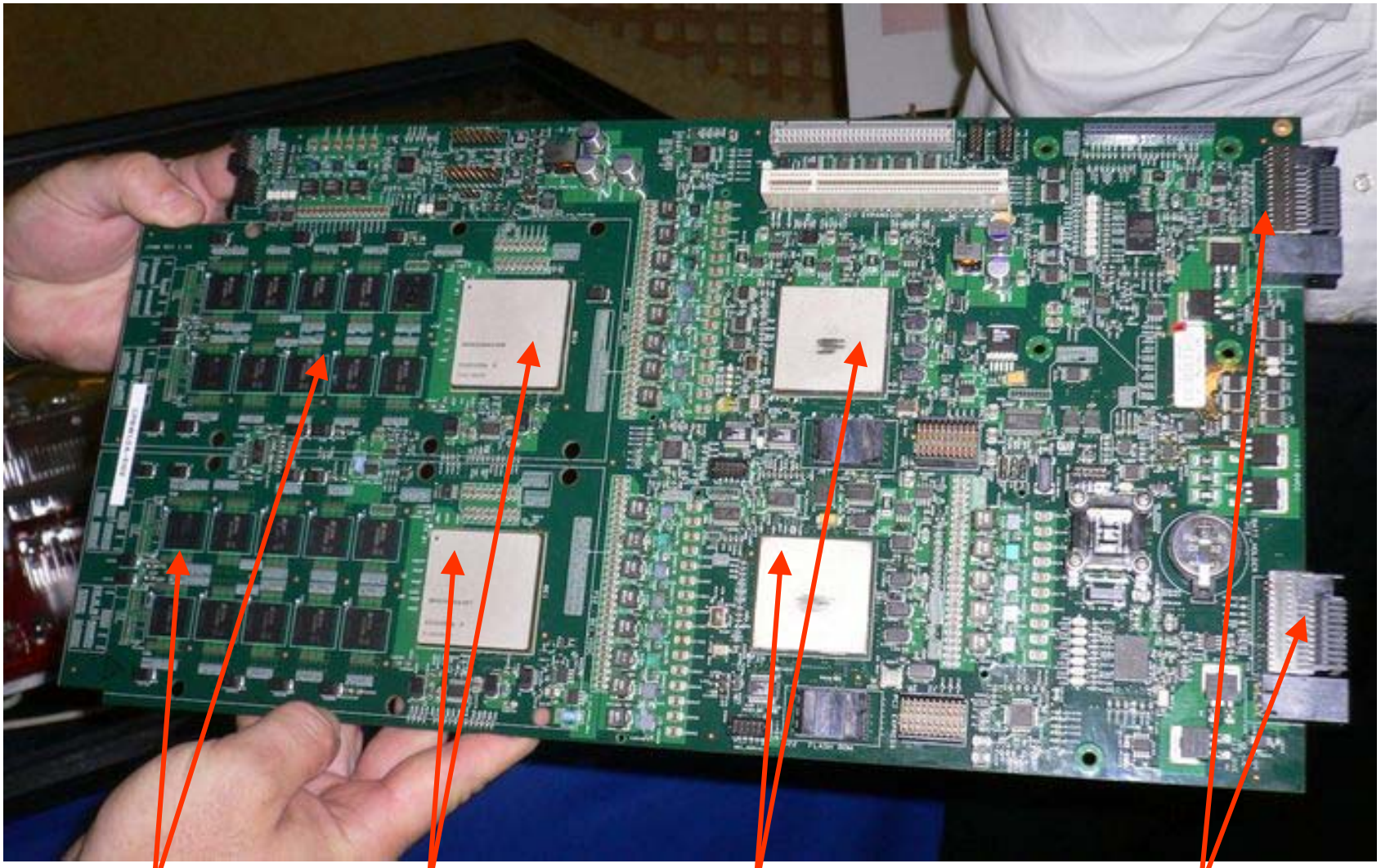
Source: Cell Broadband Engine Architecture and its first implementation – A performance view, http://www-128.ibm.com/developerworks/library/pa-cellperf/

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
  - its 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

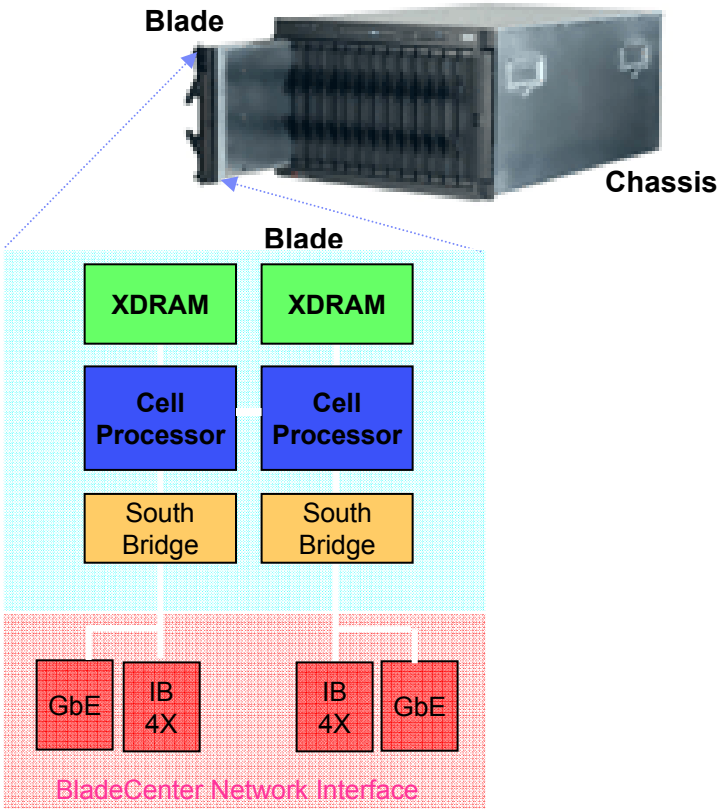
Cell Blade

The First Generation Cell Blade



Cell Blade Overview

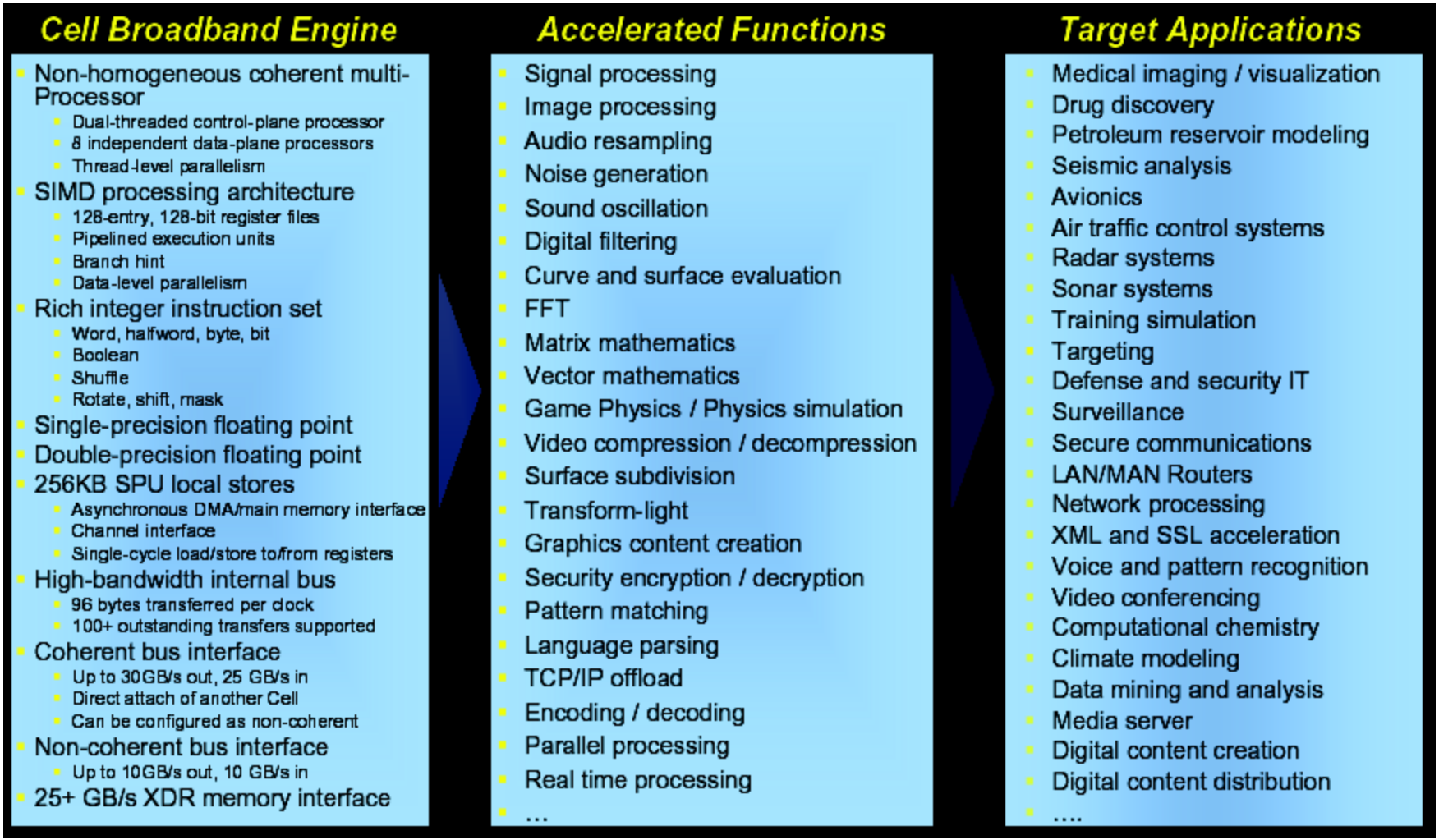
- **Blade**
  - Two Cell BE Processors
  - 1GB XDRAM
  - BladeCenter Interface (Based on IBM JS20)
- **Chassis**
  - Standard IBM BladeCenter form factor with:
    - 7 Blades (for 2 slots each) with full performance
    - 2 switches (1Gb Ethernet) with 4 external ports each
  - Updated Management Module Firmware.
  - External Infiniband Switches with optional FC ports.



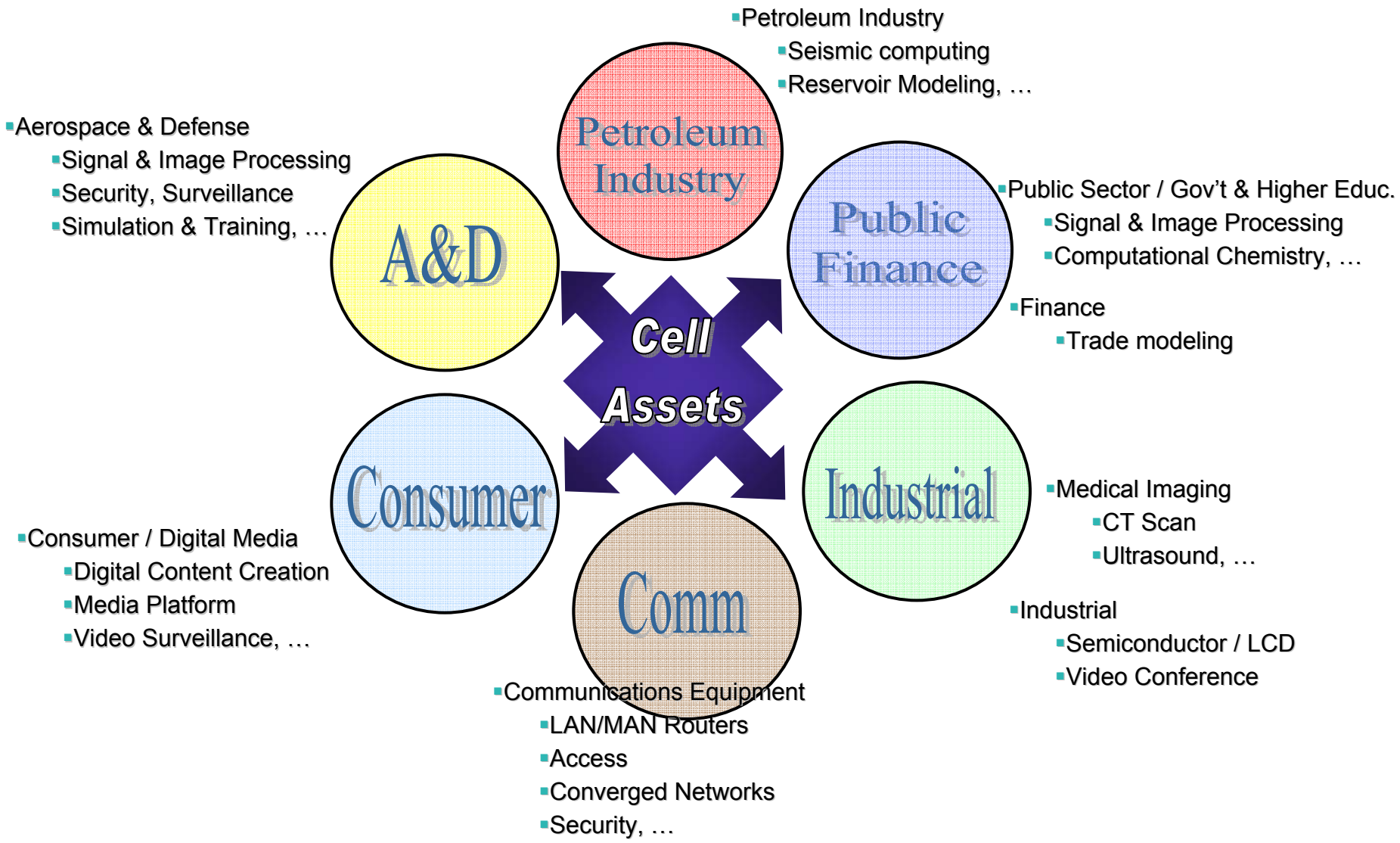
- **Typical Configuration (available today from E&TS)**
  - eServer 25U Rack
  - 7U Chassis with Cell BE Blades, OpenPower 710
  - Nortel GbE switch
  - GCC C/C++ (Barcelona) or XLC Compiler for Cell (alphaworks)
  - SDK Kit on http://www-128.ibm.com/developerworks/power/cell/

Cell Application Affinity

Cell Application Affinity



Target Opportunities for Cell Blade

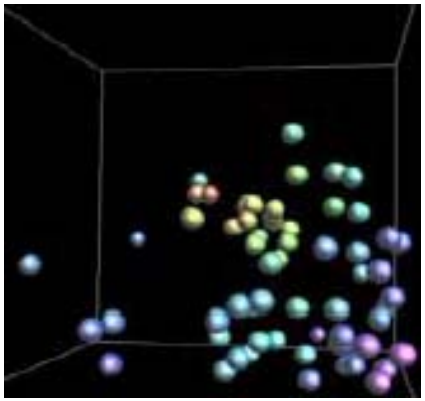


Samples / Workloads / Demos

- 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

Geometry Engine

Physics Simulation



Terrain Rendering Engine

Subdivision Surfaces

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**

IBM Microelectronics Division
1580 Route 52, Bldg. 504
Hopewell Junction, NY 12533-6351

The IBM home page is http://www.ibm.com
The IBM Microelectronics Division home page is http://www.chips.ibm.com