Introduction to Cell BE

Trond Hagen
SINTEF ICT, Department of Applied Mathematics

Winter School, Geilo, January 2008

Schedule, Friday

09:00 - 09:45	Introduction to Cell BE	Trond Hagen
10:00 - 10:45	Programming Cell BE	André Brodtkorb
11:00-12:00	"Birds of a feather" – parallel processing Summary and discussion	Johan Seland

Cell Broadband Engine

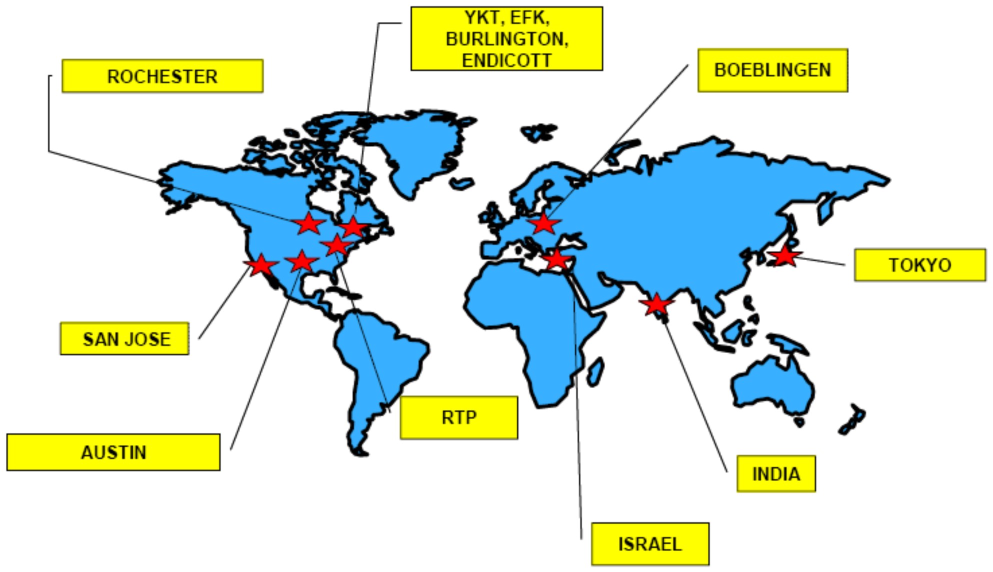
- Nine core heterogeneous architecture
  - One general-purpose core
    Power Processor Element (PPE)
  - Eight special accelerator cores
    Synergistic Processor Elements (SPE)

- The PlayStation 3 uses the Cell processor as its CPU

- The technology in the Cell is similar to that in a GPU, but the Cell is more general purpose so it can be used for a wider variety of tasks

Cell Broadband Engine History

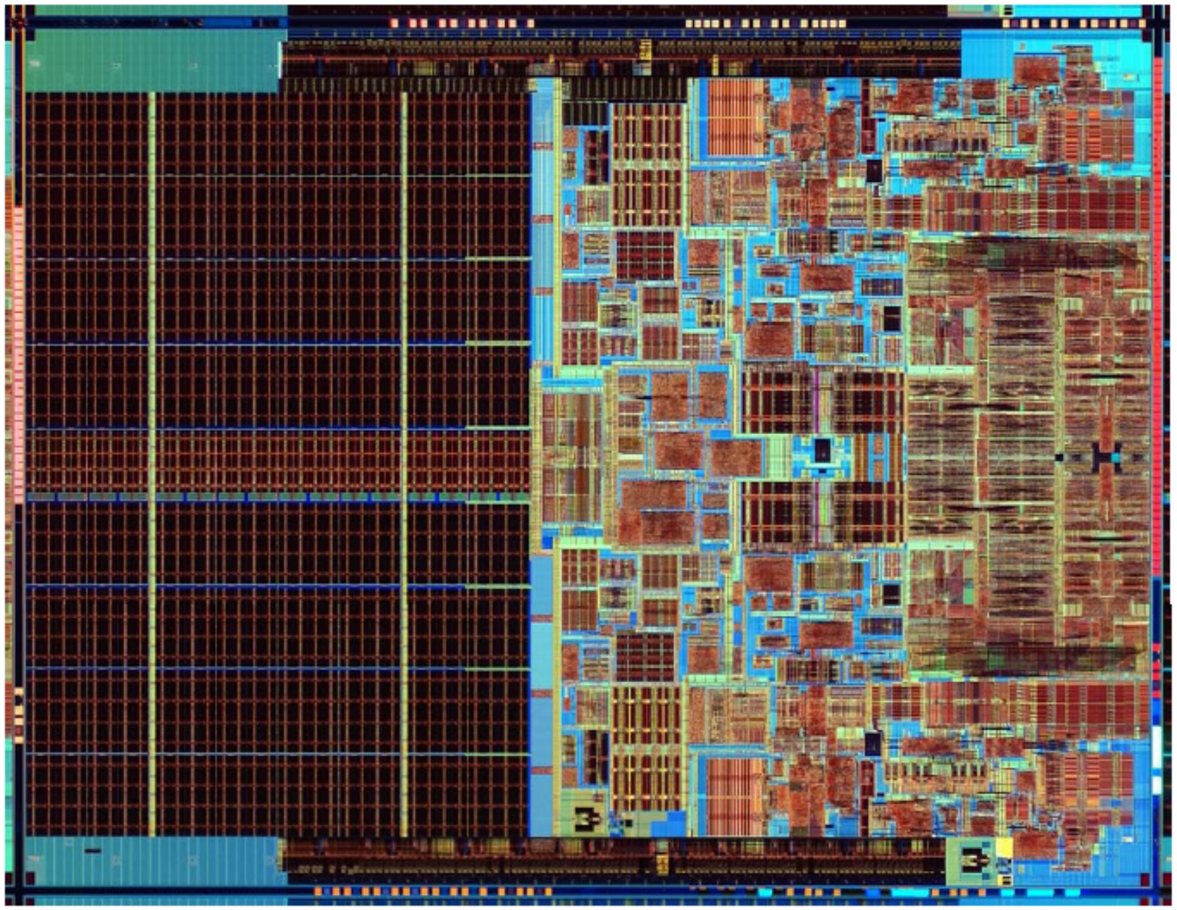
- In 2000, Sony, Toshiba, and IBM formed an alliance (STI) to design and manufacture the processor.
- In 2001 the design center opened. ~$400M investment, 5 years, 600 people
- In 2005 Sony confirmed that the Cell would be shipped in Playstation 3
- Later in 2005 IBM and Mercury CS announced a partnership agreement to build Cell-based computer systems.
- Alliance extended until 2011.



Motivation for Cell BE

- Increasing the frequency has several implications:
- Memory problem
  - Memory speeds have not increased as fast as core frequencies
- Instruction pipelines
  - Longer instruction pipelines allow for higher core frequencies, but results in high penalty when branch prediction fails or with cache misses
- Power consumption
  - When the frequency increases, the power consumption increases disproportionately

Intel Core 2 Duo

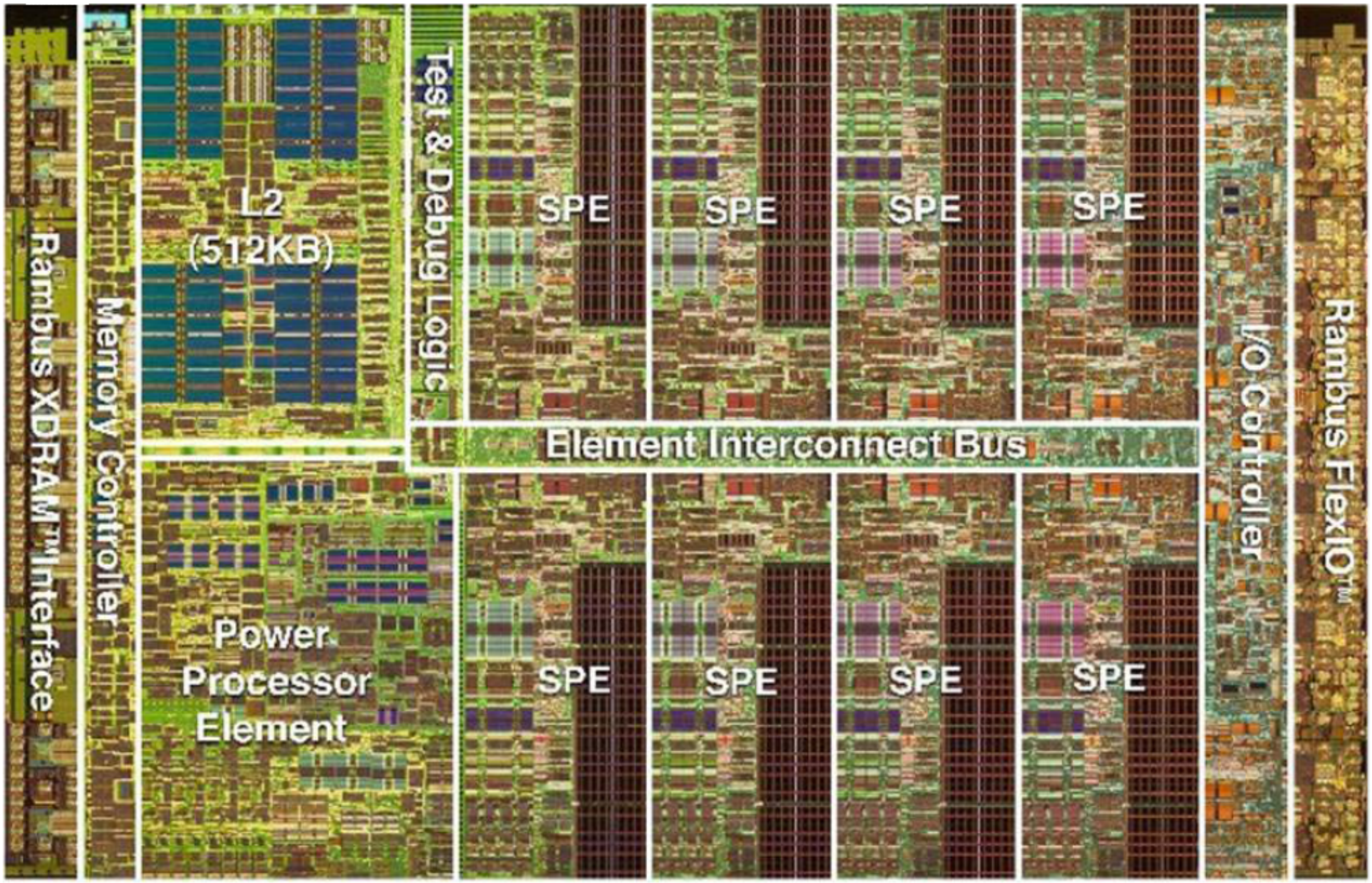


Two “fat” cores

Cell BE Solutions

- Increase concurrency
  - Multiple cores
  - SIMD / Vector operations in a core
  - Start memory movement early so that memory is available when needed
- Increased efficiency
  - Simple cores devote more resources to actual computation
  - Programmer managed memory is more efficient than dragging data through caches
  - Large register files give the compiler more flexibility and eliminate transistors needed for register renaming
  - Specialize processor cores for specific tasks

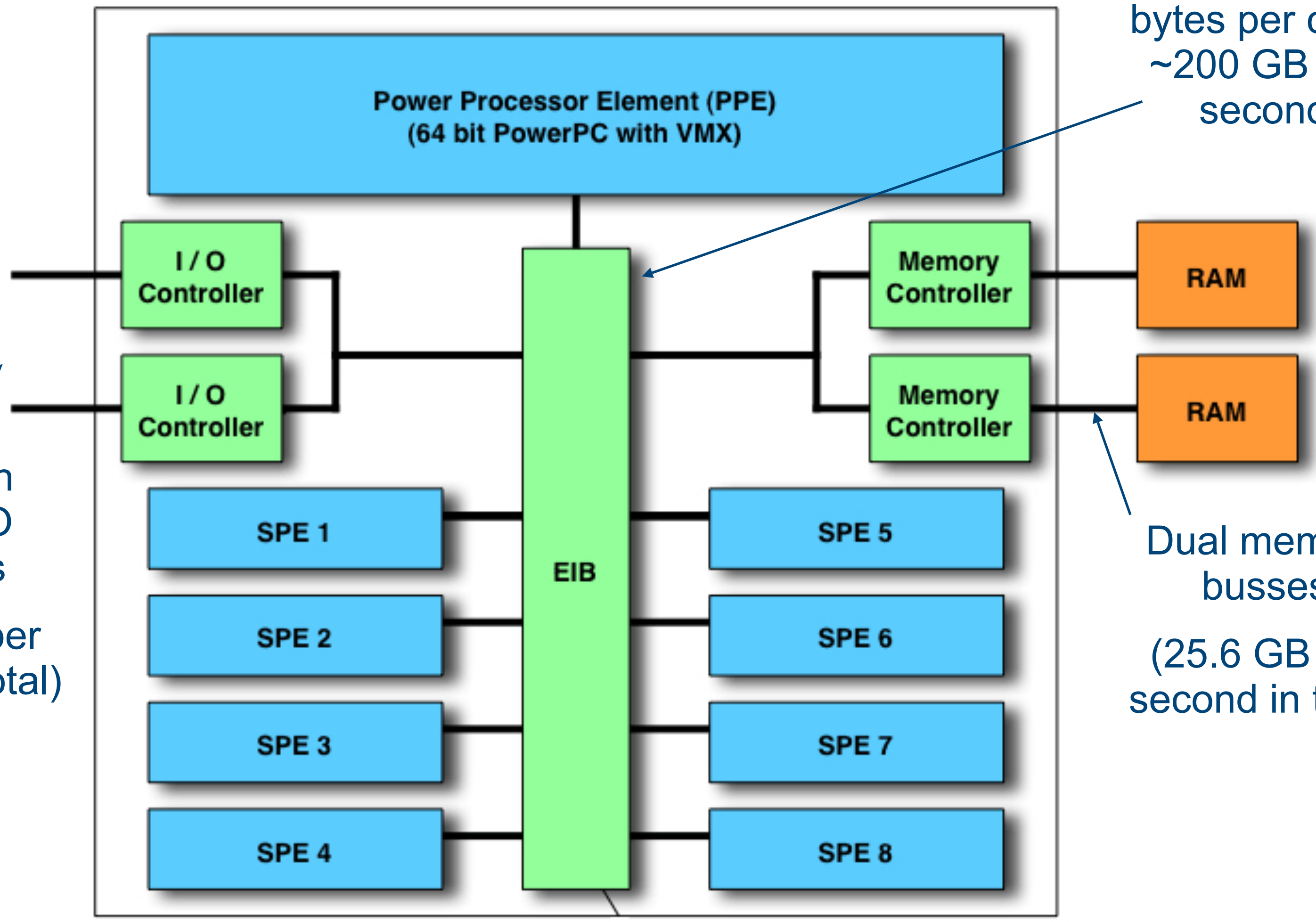
The Cell BE Die



One “fat” core and eight “thin” cores

Cell Architecture

EIB peak is 96 bytes per cycle ~200 GB per second



Dual High speed I/O channels

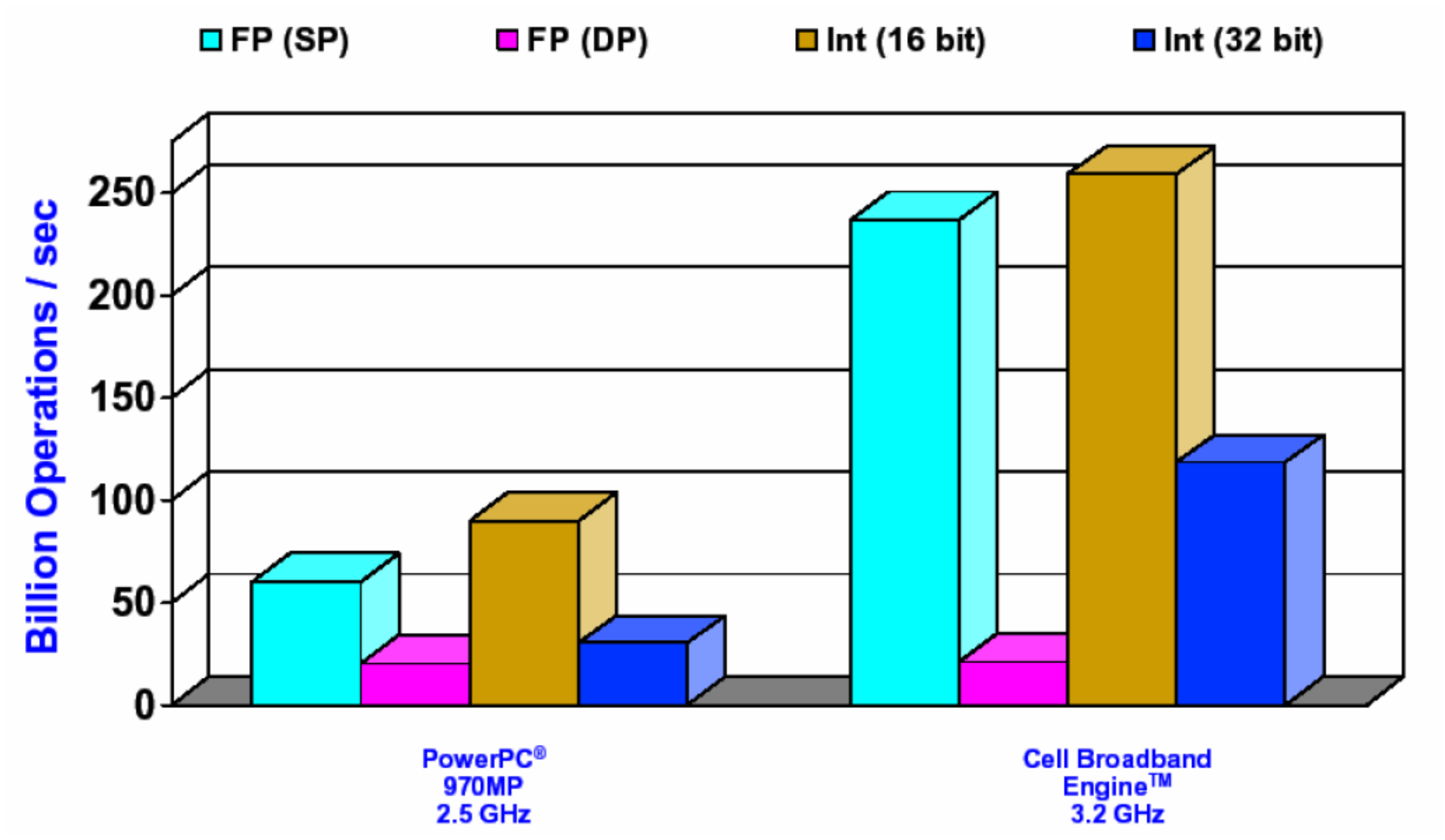
(76.8 GB per second in total)

Dual memory busses

(25.6 GB per second in total)

image courtesy of Nicholas Blachford

Theoretical Peak Performance



Key Performance Characteristics

- Cell BE performance is about an order of magnitude better compared to general-purpose processors for applications that can utilize the SIMD capability
- Each SPE is able to perform approximately the same as, or more than, a general-purpose processor with SIMD.

Power Consumption: CPU vs. Cell BE

- CPU – Intel Core 2 Duo “Conroe”
  - 65 W / 20 GFLOPS = **3.25** Watt / GFLOPS
- Cell BE
  - 70 W / 250 GFLOPS = **0.28** Watt / GFLOPS
- Each SPE
  - 5 W / 25.6 GFLOPS = **0.19** Watt / GFLOPS

The Power Processor Element (PPE)

- The PPE is a general purpose processor which is acting as a controller for the SPEs
- The operating system and most of the application runs on the PPE, but the highly computational intensive tasks are off-loaded to the SPEs
- The PPE is a 64 bit “Power Architecture” processor with 512 K cache
- Includes support for vector instructions (VMX, AltiVec)

Synergistic Processor Elements (SPEs)

- Each Cell contains 8 SPEs
- Composed of a Synergistic Processing Unit (SPU) and a Memory Flow Controller (MFC)
- Each SPE includes a 256 KB local store instead of cache for instructions and data
- The SPE contain 128 x 128 bit registers
- Vector processor capable of 4 x 32 bit operations per cycle
  - Programs need to be “vectorized” for maximum performance

SPE Local Store

- Cache problem:
  - If data being worked on is not present in the cache, the CPU stalls and has to wait for this data to be fetched. This stalls the CPU for hundreds of cycles

- SPEs lack a cache and instead use local store
  - By not using cache, a lot of the complexity with a cache is removed, and the calculations are made faster
  - 16 bytes can be moved to or from the local store per cycle giving 64 Gbytes per second
  - Cache can deliver similar or even faster data rates but only in very short bursts (a couple of hundred cycles).
  - The local store can deliver data at this rate continually for over ten thousand cycles without going to RAM

PPE vs. SPE

- Both PPE and SPE execute SIMD instructions
  - PPE processes SIMD operations in the VXU within its PPU
  - SPEs process SIMD operations in their SPU
- Both processors execute different instruction sets
- Programs written for the PPE and SPEs must be compiled by different compilers

Communication Between the PPE and SPEs

- PPE communicates with SPEs through memory-mapped I/O registers supported by the MFC of each SPE.
- Three primary communication mechanisms between the PPE and SPEs.
  - Mailboxes:
    - Queues for exchanging 32-bit messages.
    - Two mailboxes for sending messages from the SPE to the PPE (SPU Write Outbound Mailbox, SPU Write Outbound Interrupt Mailbox)
    - One mailbox for sending messages to the SPE (SPU Read Inbound Mailbox)
  - Signal Notification Registers:
    - Each SPE has two 32-bit signal-notification registers, each has a corresponding memory-mapped I/O register into which the signal-notification data is written by the sending processor.
    - They can be used by other SPEs, the PPE, or other devices to send information such as a buffer-completion synchronization flag, to an SPE
  - DMAs:
    - Transfer data between main storage and local store.

PPE and SPE MFC Command Differences

- Code running on the SPU issues an MFC command by executing a series of writes and/or reads using channel instructions.
- Code running on the PPE or other devices issues an MFC command by performing a series of stores and/or loads to memory-mapped I/O registers in the MFC.
- Data-transfer directions for MFC DMA commands is always referenced from the perspective of an SPE
  - get: transfer data into an SPE (from main storage to local store)
  - put: transfer data out of an SPE (from local store to main storage)

Element Interconnect Bus (EIB)

- The EIB is a communication bus which connects the PPE, the memory controller, the SPEs, and two off-chip I/O interfaces
- The EIB is presently implemented as a circular ring comprised of four 16B-wide unidirectional channels which counter-rotate in pairs
- Bandwith on the EIB is ~200 GB/s

Stream Processing – Decoding Digital TV

- A Cell processor can be set up to perform streaming operations in a sequence with one or more SPEs working on each step

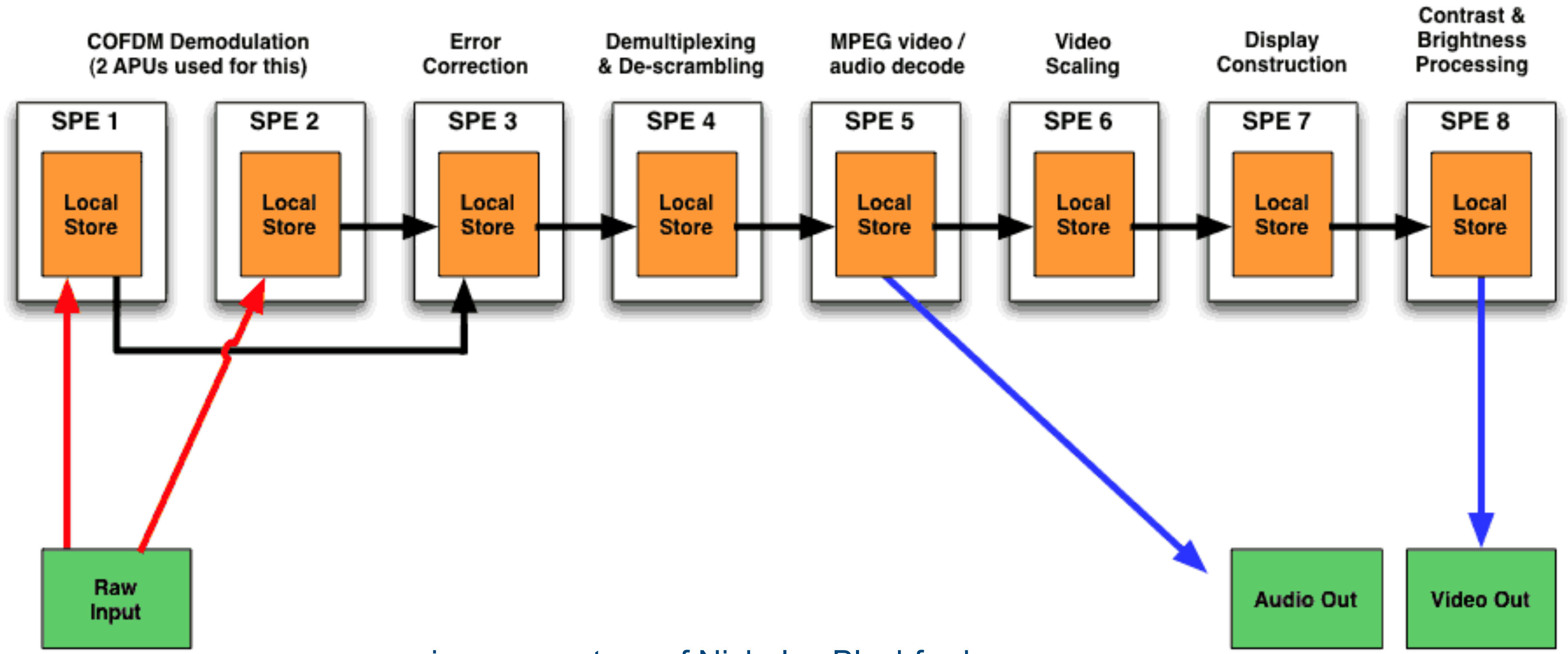
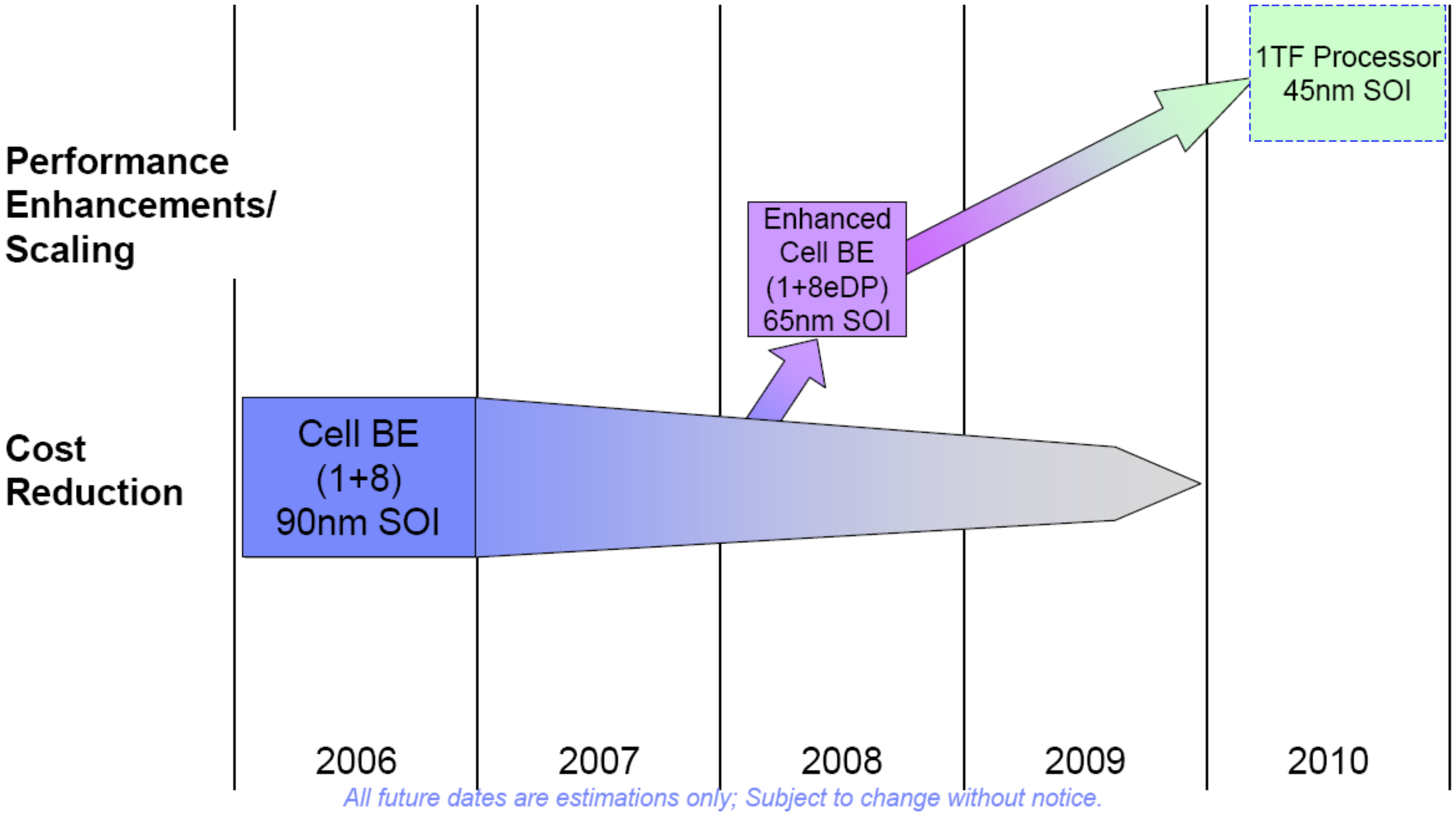

COFDM Demodulation (2 APUs used for this)
Error Correction
Demultiplexing & De-scrambling
MPEG video / audio decode
Video Scaling
Display Construction
Contrast & Brightness Processing
SPE 1
SPE 2
SPE 3
SPE 4
SPE 5
SPE 6
SPE 7
SPE 8
Local Store
Raw Input
Audio Out
Video Out

image courtesy of Nicholas Blachford

Cell BE Architecture Roadmap


Performance Enhancements/ Scaling
Cost Reduction
Cell BE (1+8) 90nm SOI
Enhanced Cell BE (1+8eDP) 65nm SOI
1TF Processor 45nm SOI
2006
2007
2008
2009
2010
All future dates are estimations only; Subject to change without notice.

PlayStation 3

- The PlayStation 3 is probably the easiest and cheapest way for programmers to get their hands on the Cell processor
- Create an Linux partition in addition to the game OS partition
- Install Linux and Cell BE SDK

Image courtesy of hardware.no

- Installing Linux and Cell BE SDK, walkthrough
  http://www-128.ibm.com/developerworks/library/pa-linuxps3-1/

PlayStation 3 (Cont’d)

- Only 6 SPEs are available on the PS3.
  - One SPE is disabled during the test process, to improve manufacturing yields, and one is reserved for the operating system.
- Only 256 MB system memory.
- The clock-frequency is 3.2 GHz, which gives a theoretical performance of ~160 GFLOPS (1 PPE + 6 SPEs).
- Graphics processing in the PS3 i handled by a NVIDIA RSX (G70-based) graphics card, but it is not possible to get access to the RSX using Linux.

PlayStation 3 Breaks World Record

- Folding@Home – distributed computing
  - Understand protein folding, misfolding, and related deseases
  - First project that surpassed one petaFLOPS

System	TFLOPS	Active processors	GFLOPS / processor
CPUs	245	224213	1,09
GPUs	38	647	59
Cell BE	825	33267	24,8

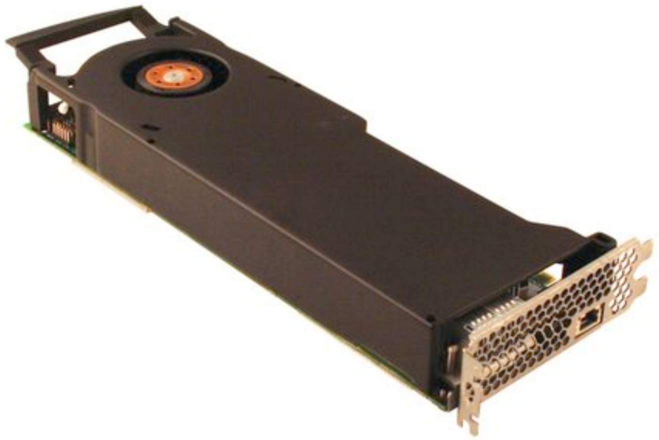
Note: GFLOPS / processor is not an accurate estimate of the performance. If users suspend the client they enlarge the time between getting the data and delivering the result, which reduces the FLOPS value

Ray-tracing using the Cell BE

- Real-time rendering of a complex landscape using only software rendering techniques
  - Linux based PlayStation 3
  - 3 million triangles
  - 1080p resolution
  - iRT (An iterative ray-tracer for the Cell processor)
  - Only using the Cell processor and not the NVIDIA RSX
- Video

Cell Processors in x86 Workstations

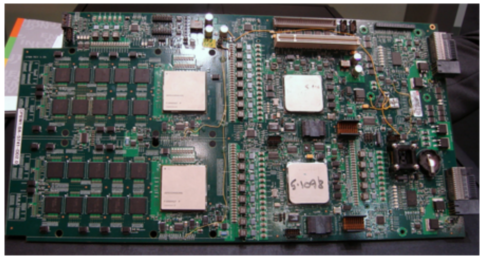
- Mercury Cell Accelerator board which can be plugged into any PCIe-enabled workstation
- 1 Cell processor running at 2.8 GHz
- MultiCore Plus SDK Software
  - Software for programming the Cell processor



Mercury Cell based PCIe board

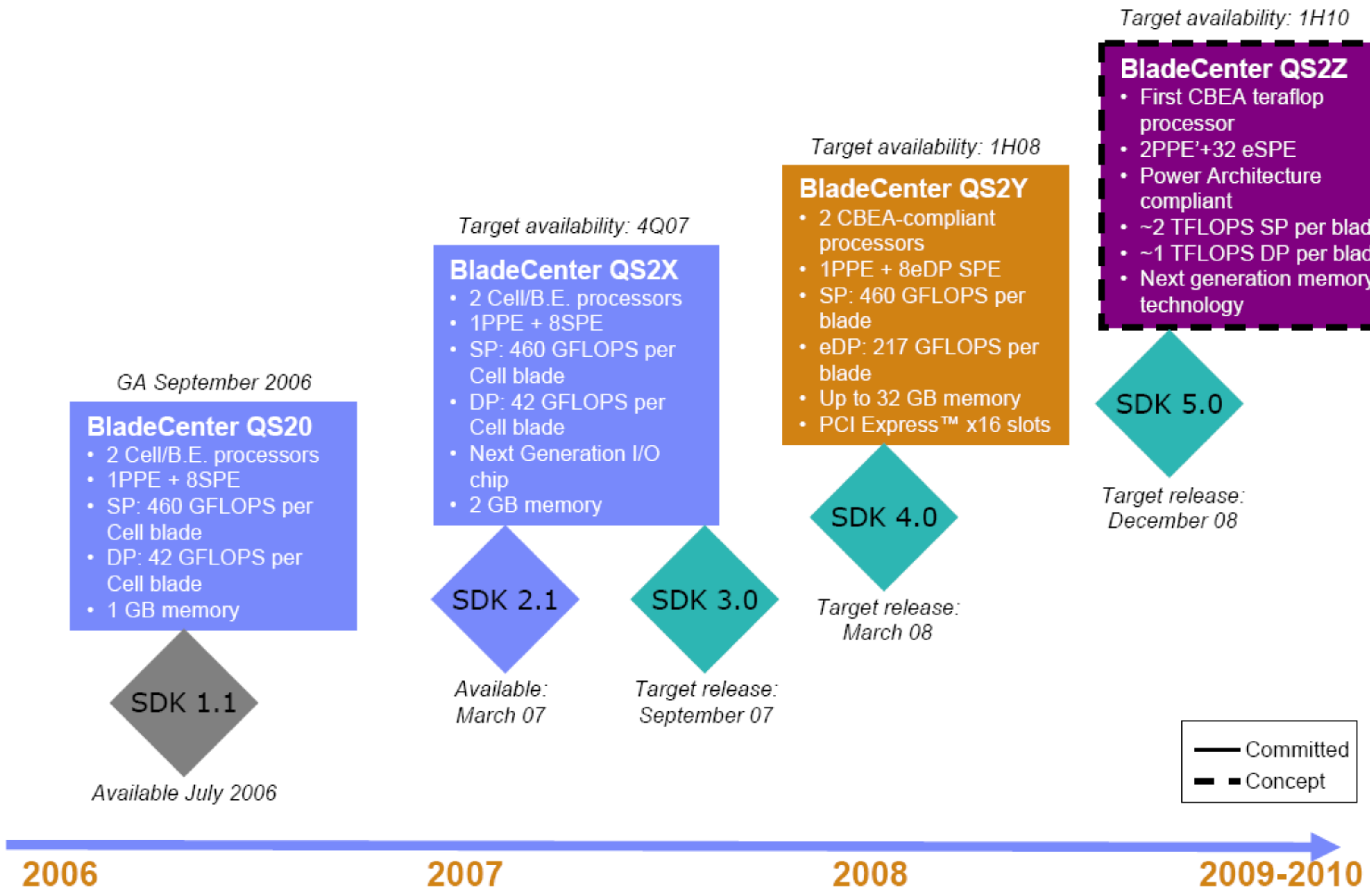
Cell BE Blade Server

- QS21 BladeCenter
  - Two Cell processors per blade
  - 3.2 GHz
  - 2 GB memory
  - ~500 GFLOPS
  - 16 SPEs


QS21 BladeCenter

BladeCenter H Chassis

Cell BE Architecture Blades



GA September 2006

BladeCenter QS20
- 2 Cell/B.E. processors
- 1PPE + 8SPE
- SP: 460 GFLOPS per Cell blade
- DP: 42 GFLOPS per Cell blade
- 1 GB memory

SDK 1.1

Available July 2006

Target availability: 4Q07

BladeCenter QS2X
- 2 Cell/B.E. processors
- 1PPE + 8SPE
- SP: 460 GFLOPS per Cell blade
- DP: 42 GFLOPS per Cell blade
- Next Generation I/O chip
- 2 GB memory

SDK 2.1

Available: March 07

SDK 3.0

Target release: September 07

Target availability: 1H08

BladeCenter QS2Y
- 2 CBEA-compliant processors
- 1PPE + 8eDP SPE
- SP: 460 GFLOPS per blade
- eDP: 217 GFLOPS per blade
- Up to 32 GB memory
- PCI Express™ x16 slots

SDK 4.0

Target release: March 08

Target availability: 1H10

BladeCenter QS2Z
- First CBEA teraflop processor
- 2PPE'+32 eSPE
- Power Architecture compliant
- ~2 TFLOPS SP per blade
- ~1 TFLOPS DP per blade
- Next generation memory technology

SDK 5.0

Target release: December 08

Roadrunner Project

- Next generation supercomputer to be built
- System is designed to deliver 1.6 petaFLOPS peak
- World's first TOP500 Linpack sustained 1.0 petaFLOPS system
- 16 000 AMD Opteron cores
- 16 000 Cell processors
  - The new Cell processor with >100 GFLOPS double precision performance
- Installation in 2008

Programming a Cell

- Not hard to program a Cell if you are familiar with multithreading, vector operations and DMA.
- Porting algorithms to Cell:
  - First step is to make an application run on the PPE.
  - Second step is to identify what should run on the SPEs
  - Third step is to vectorize the code for maximum performance

Conclusion

- Cell is a heterogeneous architecture which gives a vast performance boost over traditional CPUs
- Many people do not like change, to them Cell represents a threat. For others it represents an opportunity.
- Start rewriting your algorithms ☺

Thank You!