# Game Processor Architectures

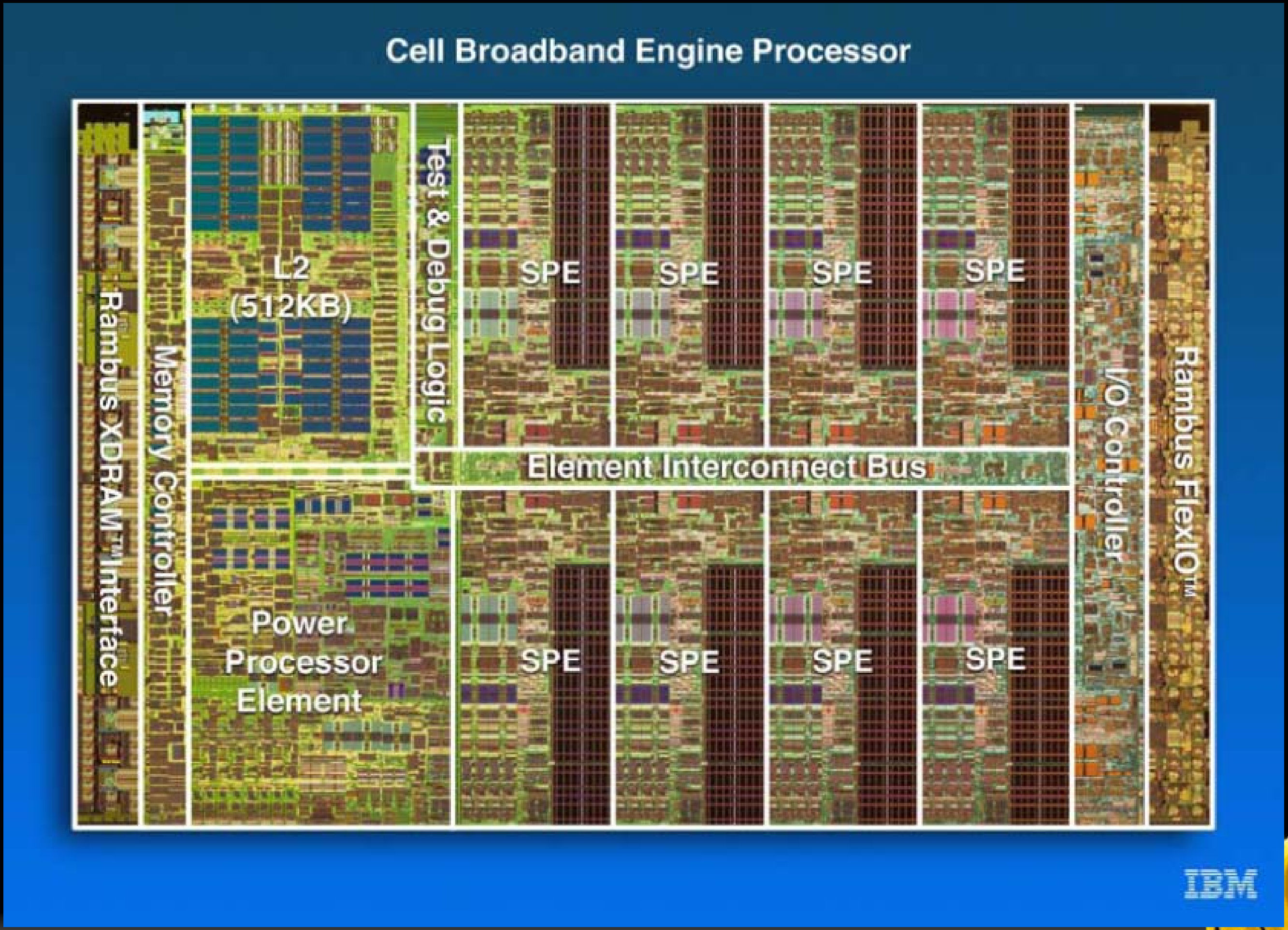
Dr. Mark Heinrich

COT 4810

UCF EECS

# PlayStation 3

# Cell Broadband Engine (CBE)



# Cell Origins and Acknowledgments

- Cell is the result of a partnership between Sony, Toshiba, and IBM
- Cell represents the work of more than 400 people starting in 2001
- More detailed papers on the Cell implementation and the SPE micro-architecture can be found in the ISSCC 2005 proceedings

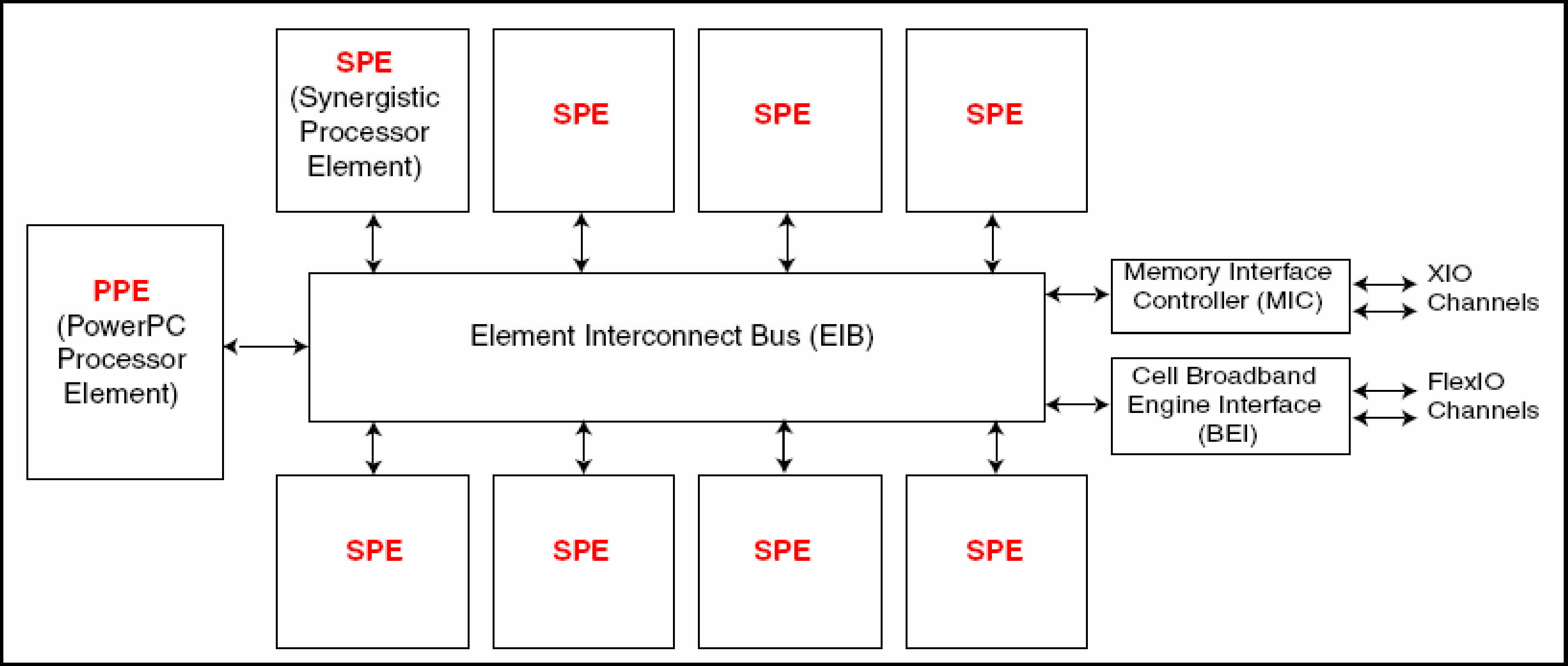
# CBE Architecture

- Effectively a 9-way multiprocessor
  - 8-way CMP plus one control processor, designed by IBM
- One main 64-bit PPE processor
  - Power Processor Element, 2 hardware threads
  - Good at control tasks, task switching, OS-level code
- 8 SPE processors
  - Synergistic Processor Element
  - Good at compute-intensive tasks
- Like SIMD multiprocessors of old…sort of

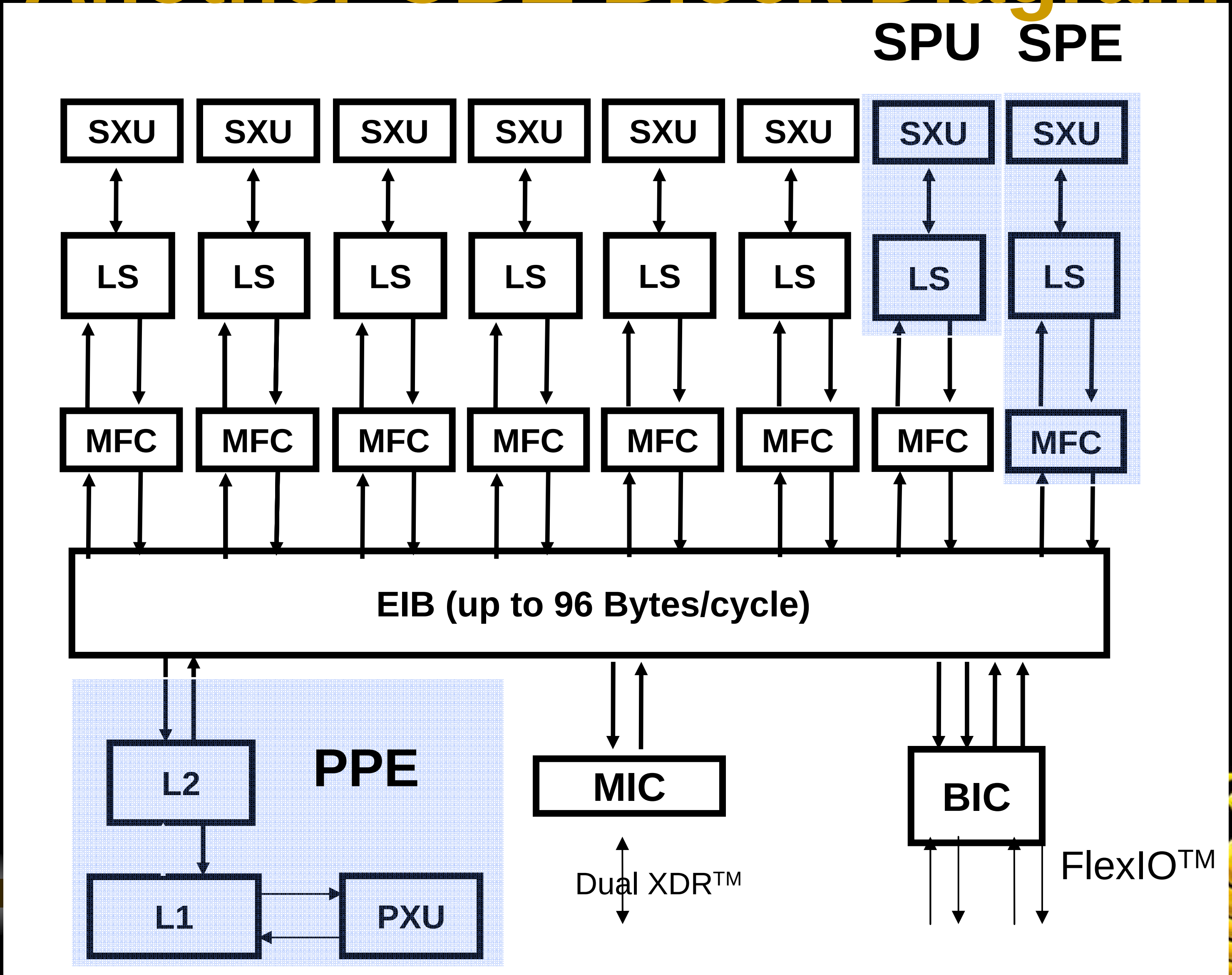
# Attributes of Cell

- Cell is Multi-Core
  - Contains 64-bit Power Architecture TM
  - Contains 8 Synergistic Processor Elements (SPE)
- Cell is a Flexible Architecture
  - Multi-OS support (including Linux) with Virtualization technology
  - Path for OS, legacy apps, and software development
- Cell is a Broadband Architecture
  - SPE is RISC architecture with SIMD organization and Local Store
  - 128+ concurrent transactions to memory per processor (16 per SPE)
- Cell is a Real-Time Architecture
  - Resource allocation (for Bandwidth Measurement)
  - Locking Caches (via Replacement Management Tables)

# CBE Block Diagram

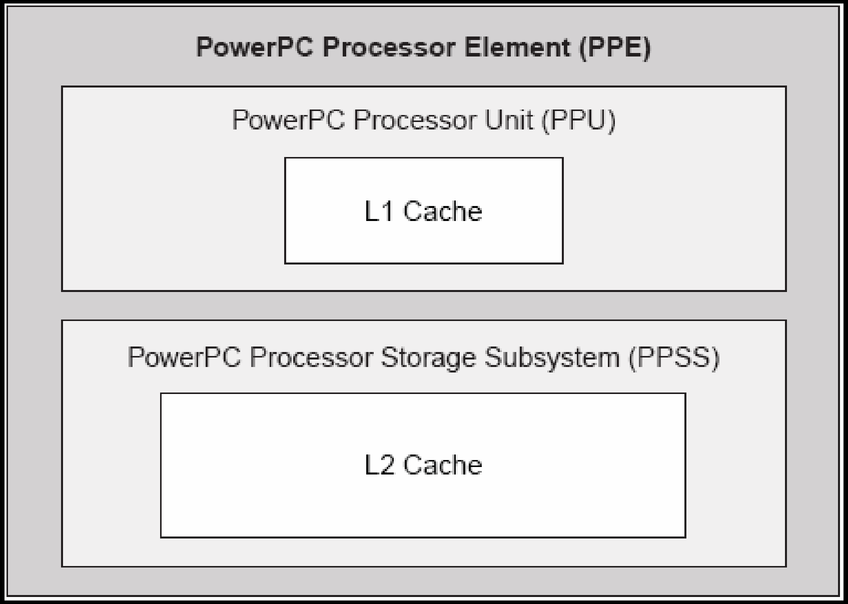

SPE
(Synergistic Processor Element)
SPE
SPE
SPE
PPE
(PowerPC Processor Element)
Element Interconnect Bus (EIB)
Memory Interface Controller (MIC)
XIO Channels
Cell Broadband Engine Interface (BEI)
FlexIO Channels
SPE
SPE
SPE
SPE

# Another CBE Block Diagram



# Power Processor Element

- 64-bit PowerPC Architecture
- In-order, 2-way hardware Multi-threaded RISC core
- Coherent Load/Store with 32KB I & D L1 and 512KB L2
- Tradition virtual memory subsystem
- Supports Vector/SIMD instruction set
- Runs OS, manages system resources etc



PowerPC Processor Element (PPE)

PowerPC Processor Unit (PPU)

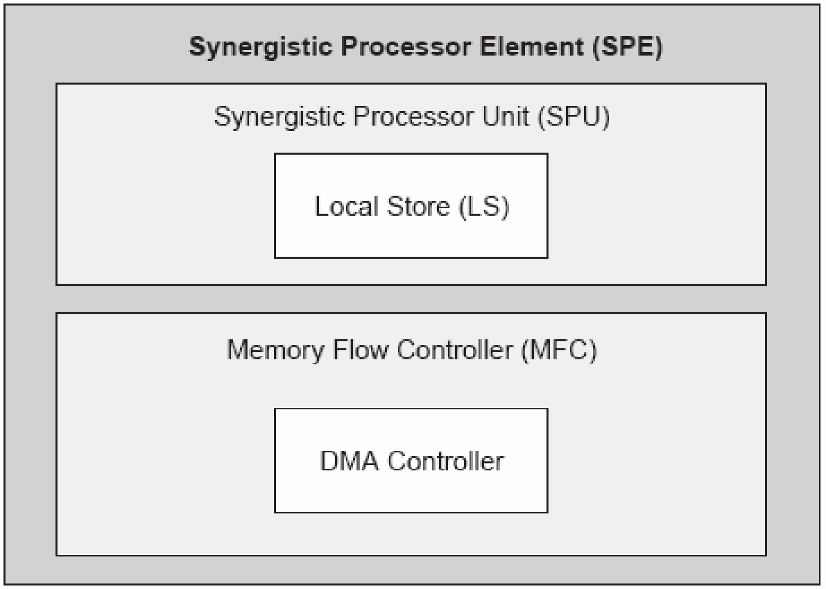
L1 Cache

PowerPC Processor Storage Subsystem (PPSS)

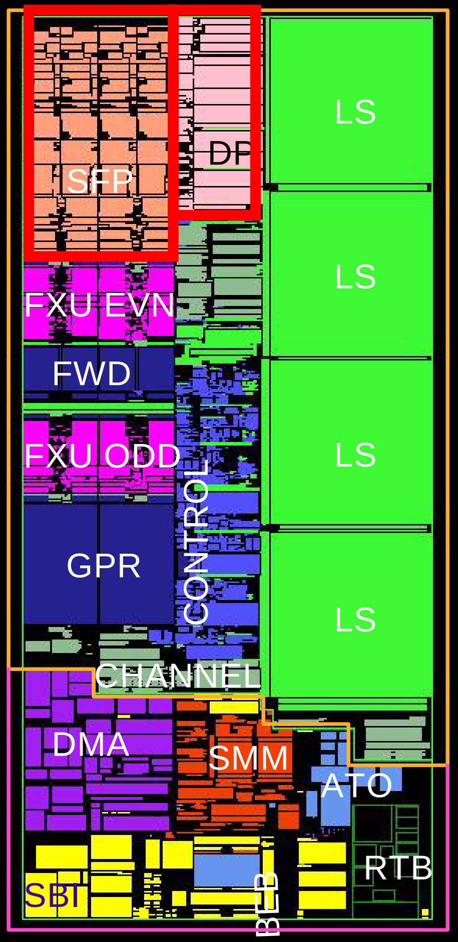
L2 Cache

# Synergistic Processor Element

- RISC core
- Dual issue, up to 16-way 128-bit SIMD
- 128-bit, 128 entry register file
- 256kb local store
- Vector/SIMD
- MFC controls DMAs to/from Local Store over EIB


Synergistic Processor Element (SPE)
Synergistic Processor Unit (SPU)
Local Store (LS)
Memory Flow Controller (MFC)
DMA Controller

# Synergistic Processor Element



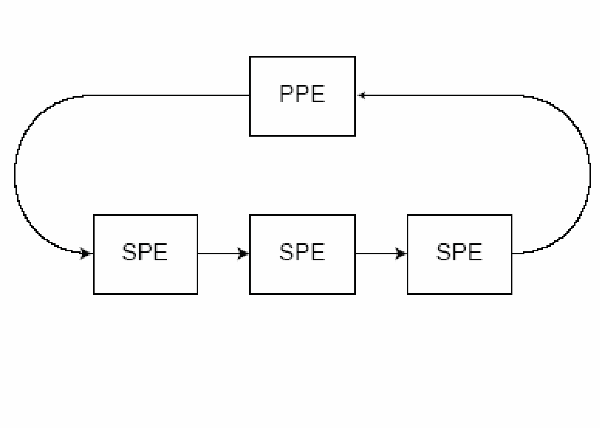
14.5mm$^2$ (90nm SOI)

- User-mode architecture
  - No translation/protection within SPU
  - DMA is full Power Arch protect/x-late
- Direct programmer control
  - DMA/DMA-list
  - Branch hint
- VMX-like SIMD dataflow
  - Broad set of operations
  - Graphics SP-Float
  - IEEE DP-Float (BlueGene-like)
- 256kB Local Store
  - Combined I & D
  - 16B/cycle L/S bandwidth
  - 128B/cycle DMA bandwidth

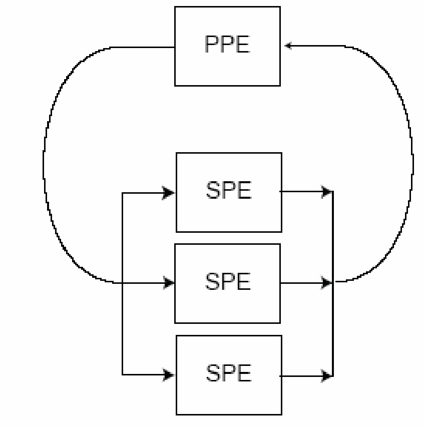
# DMA Transfers

- Primary method of transferring data to/from SPU's local store
- Maximum size 16KB
- Can be initiated by either PPE or SPE, but typically initiated by the SPE
- Offloads data transfer work to DMA controller
  - SPU continues with computation
- Double buffer for efficient use

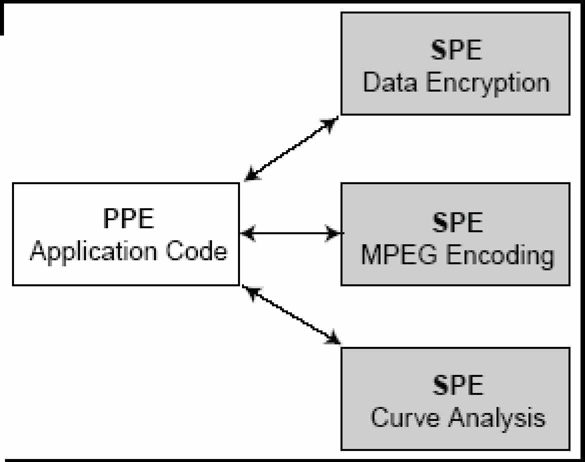
# Usage models



Multistage pipeline



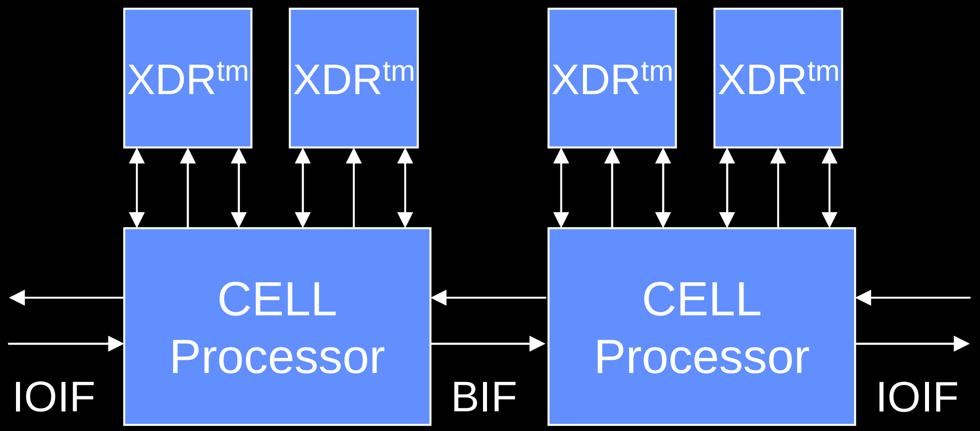
Parallel

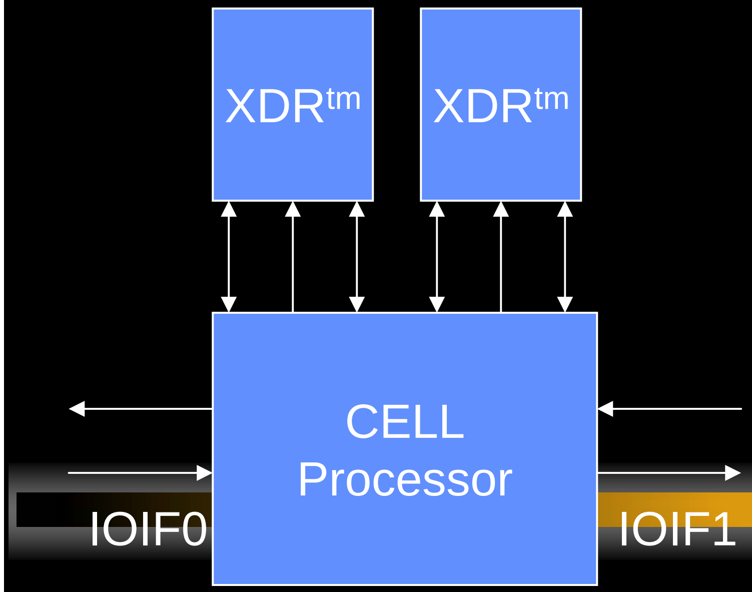


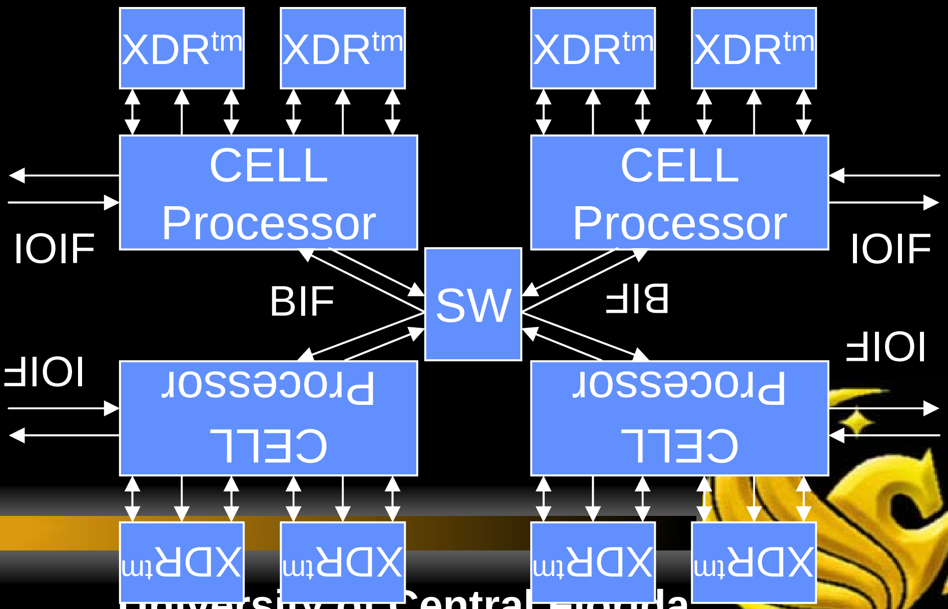
Services

# Cell Can Support Many Systems

- Game console systems
- Blade systems (QS20)
- HDTV
- Home media servers
- Supercomputers







# Programming the CBE

- C/C++ with PPU/SPU vector intrinsics
- SDK available from IBM
  - System Simulator
  - GNU/Toolchain
  - Documentation
  - Sample code
- More later

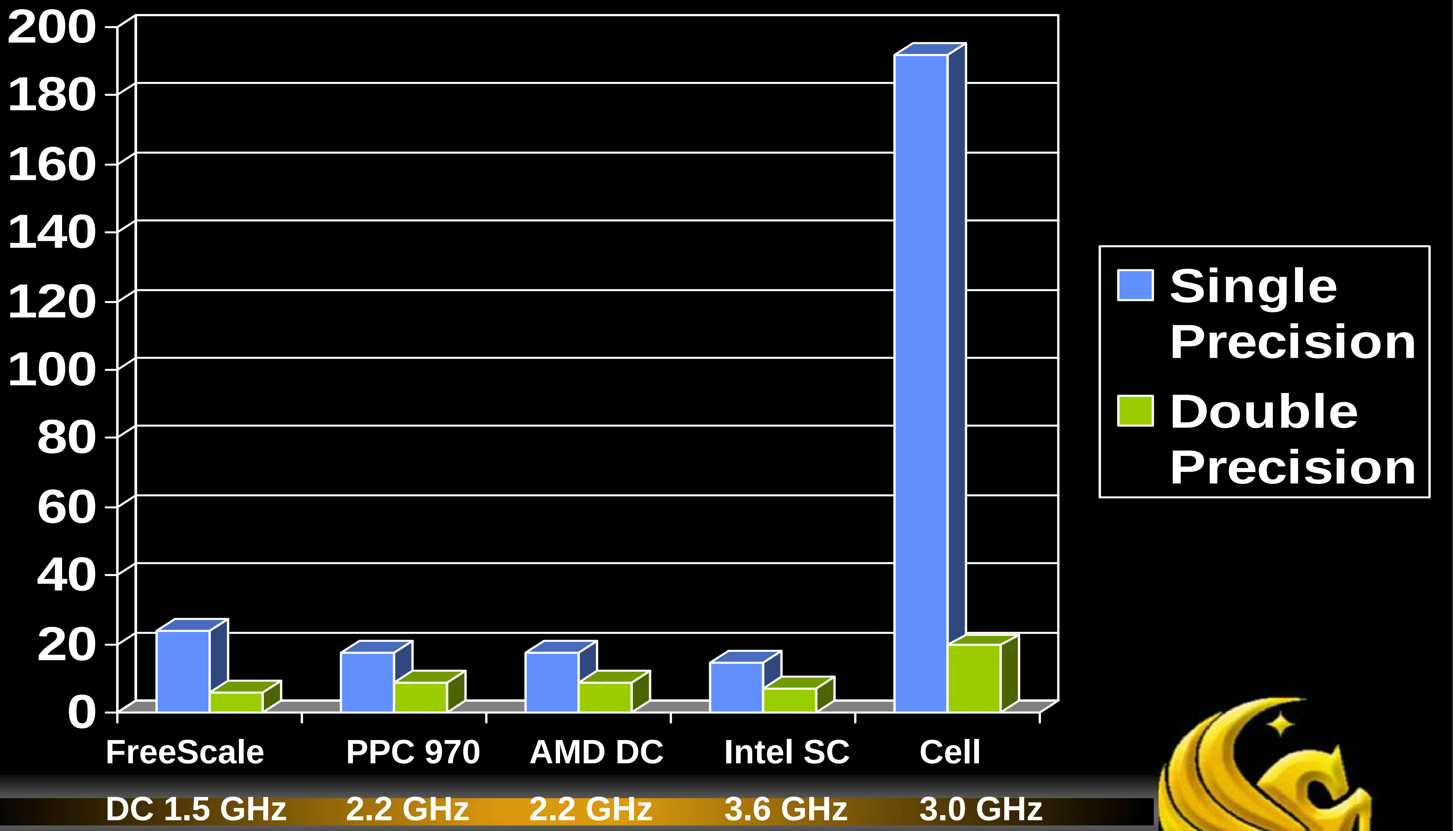
# Targetting the SPU

- Two Pipelines, dual-issue
  - Even (load & store)
  - Odd (execute)
- Design for maximum SIMD operation
- No SPU hardware branch prediction
  - Programmer/compiler specified branch hints
  - ~20 cycle penalty for misdirected branch hints
- Maximum use of register file
  - Loop unrolling

# Cell Characteristics

- Clock speed
  - > 3.2 GHz
- Peak performance (single precision)
  - > 204.8 GFLOPS
- Peak performance (double precision)
  - > 20.8 GFLOPS
- Area 221 mm$^2$
- Technology 90nm SOI
- Total # of transistors 234M (slightly less than Core 2 Duo)

# Peak GFLOPs (SPEs only)

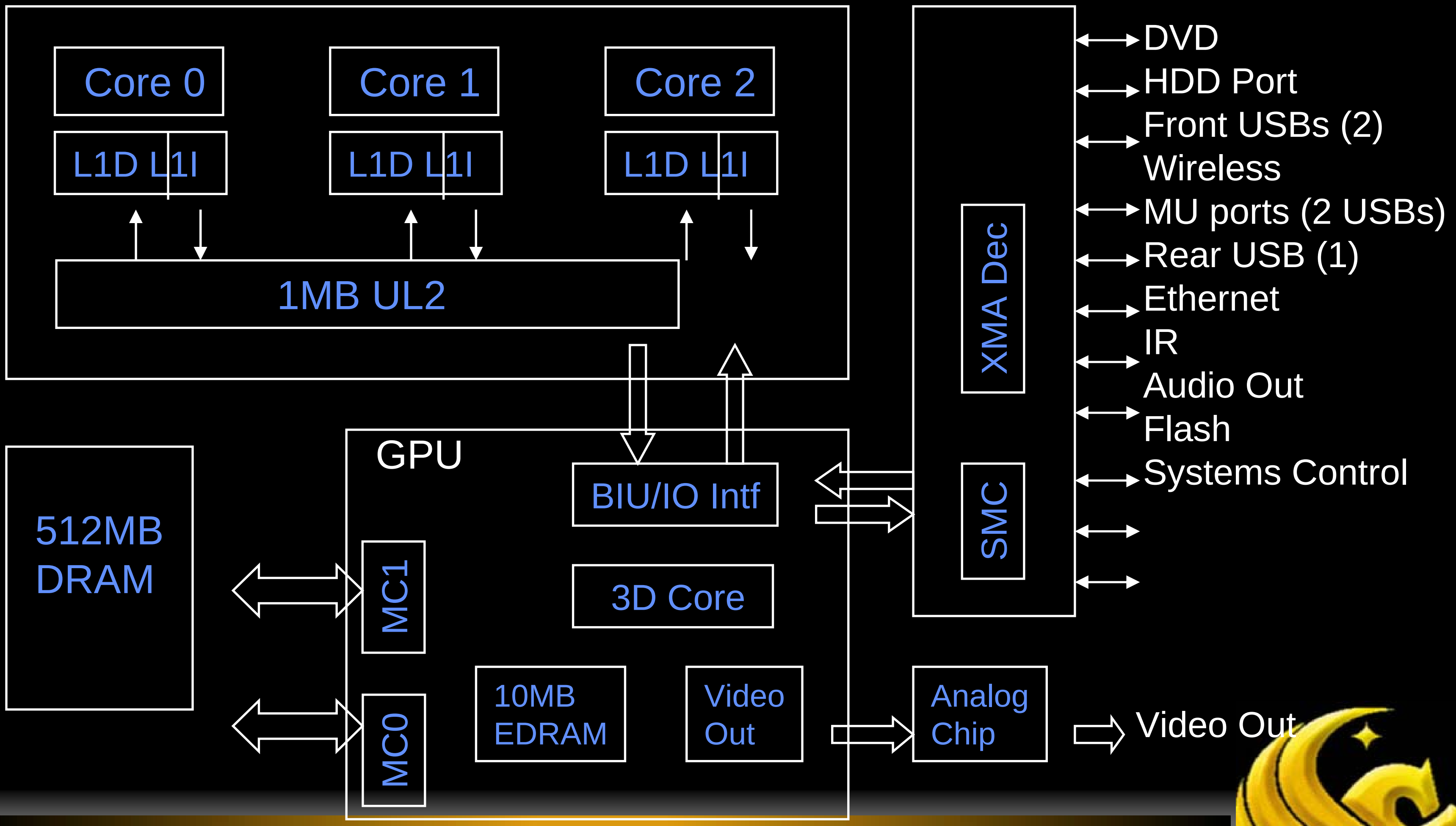


University of Central Florida

# Xbox360 – "Xenon" processor

- Provides game developers with a balanced, powerful platform
  - Three SMT processors, 32KB L1 D$ & I$, 1MB UL2 cache
  - 165M transistors total
  - 3.2 GHz Near-POWER ISA
  - 2-issue, 21-stage pipeline, with 128 128-bit registers
  - Weak branch prediction – supported by software hinting
  - In-order instructions
  - Narrow cores – 2 INT units, 2 128-bit VMX units, 1 of anything else
- An ATI-designed 500MZ GPU w/ 512MB of DDR3DRAM
  - 337M transistors, 10MB framebuffer
  - 48 pixel shader cores, each with 4 ALUs

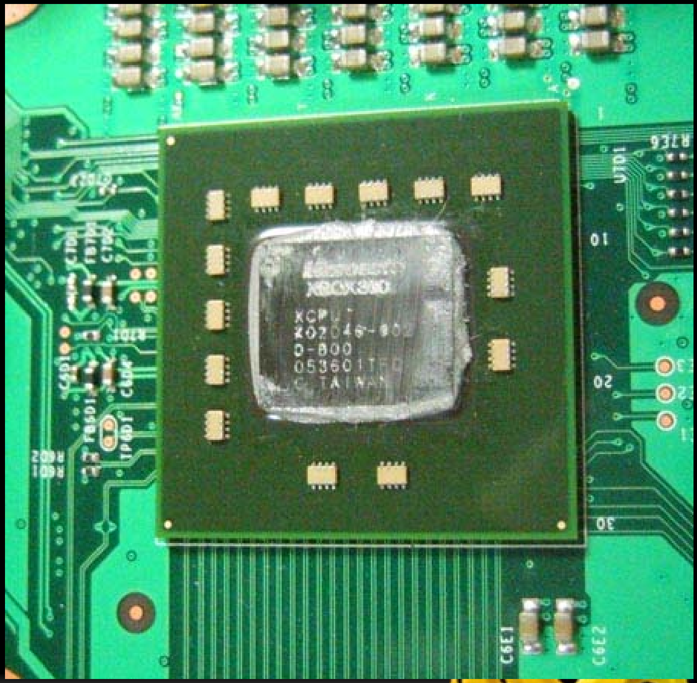
# Xenon Diagram



University of Central Florida

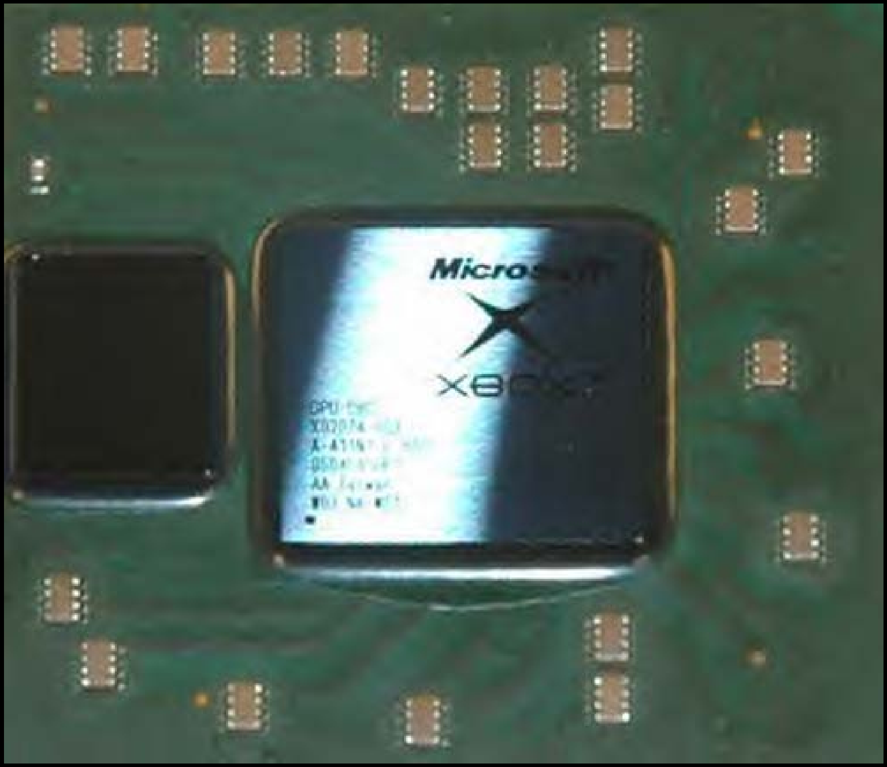
# Xbox 360 CPU

– Custom-designed IBM PowerPC-based CPU with 3 symmetrical cores running at 3.2Ghz each; 2 hardware threads per core and 6 hardware threads total



# Graphics Processor

- 500 MHz custom-designed chip
  - Developed by Microsoft and ATI
  - 48 parallel processing units
  - 10 MB of embedded DRAM
  - Unified Shader Architecture - One unit can execute both pixel and vertex shader instructions



# Memory/Hard Drive

- Total memory
  - 512MB GDDR3 RAM
- Hard drive
  - Detachable and upgradeable 20GB hard drive
  - Serial ATA interface for data and power connector
  - Same as any other SATA notebook HDD

# Hardware Abstraction Layer

- There is virtually no hardware abstraction layer on the Xbox 360.  Everything has direct access to the hardware
- This eliminates a lot of lagging and software overhead you could possibly see in a PC

# I/O Ports

- 3 USB 2.0 Ports
  - Used for controllers, removable storage, etc.
- 2 Memory Card Ports
  - Only accept Xbox 360 memory cards
- Support for up to 4 wireless controllers
- Ethernet port for internet connectivity
- Infrared sensor for remote

# Wireless Controllers

- There is a button on the front of the Xbox that works sort of like the “easy setup” button on a CISCO device
- Controllers also have a button in the middle that is used to sync up with the console (has many other applications also)

# The OS

- The Xbox 360 OS is a custom operating system which is extended from the Xbox 1 (Original Xbox)
- Xbox 1 said to have roots in Windows 2000
- The similarity between Xbox 1 and Xbox 360's operating system as well as XNA allows developers to easily move from one to the other
- The Xbox 360 OS is integrated into the system's hardware and other services in order to optimize not only the system but also the ease of development for it

# XNA

- XNA is the set of builder tools Microsoft has developed to help game developers design, develop, and manage their games. (more later)
- XNA framework is based on .NET Framework 2.0

# Possible Projects

- Develop a game on one of the development environments for one of the following game architectures
  - Xbox 360 – (IBM Xenon)
  - PlayStation 3 – (Cell Broadband Engine)
  - PlayStation Portable – MIPS R4400
- Will be difficult to run your game on the actual hardware
  - Xbox, can deploy to a 360 for a fee
  - PlayStation 3 boots Linux, but with no access to the video hardware
  - PSP has a good dev environment but no simulator

# Xbox 360 Resources

- Xbox 360 has the Microsoft XNA Game Studio Express
  - What does XNA stand for?
  - http://msdn.microsoft.com/directx/XNA/default.aspx
  - Can run same codes on Windows and the Xbox 360 with the XNA Framework
- To run on an Xbox 360 itself, you need an XNA Creators Club subscription purchased directly from the Xbox Live Marketplace. Two subscription options are available: $99 per year or $49 per four months

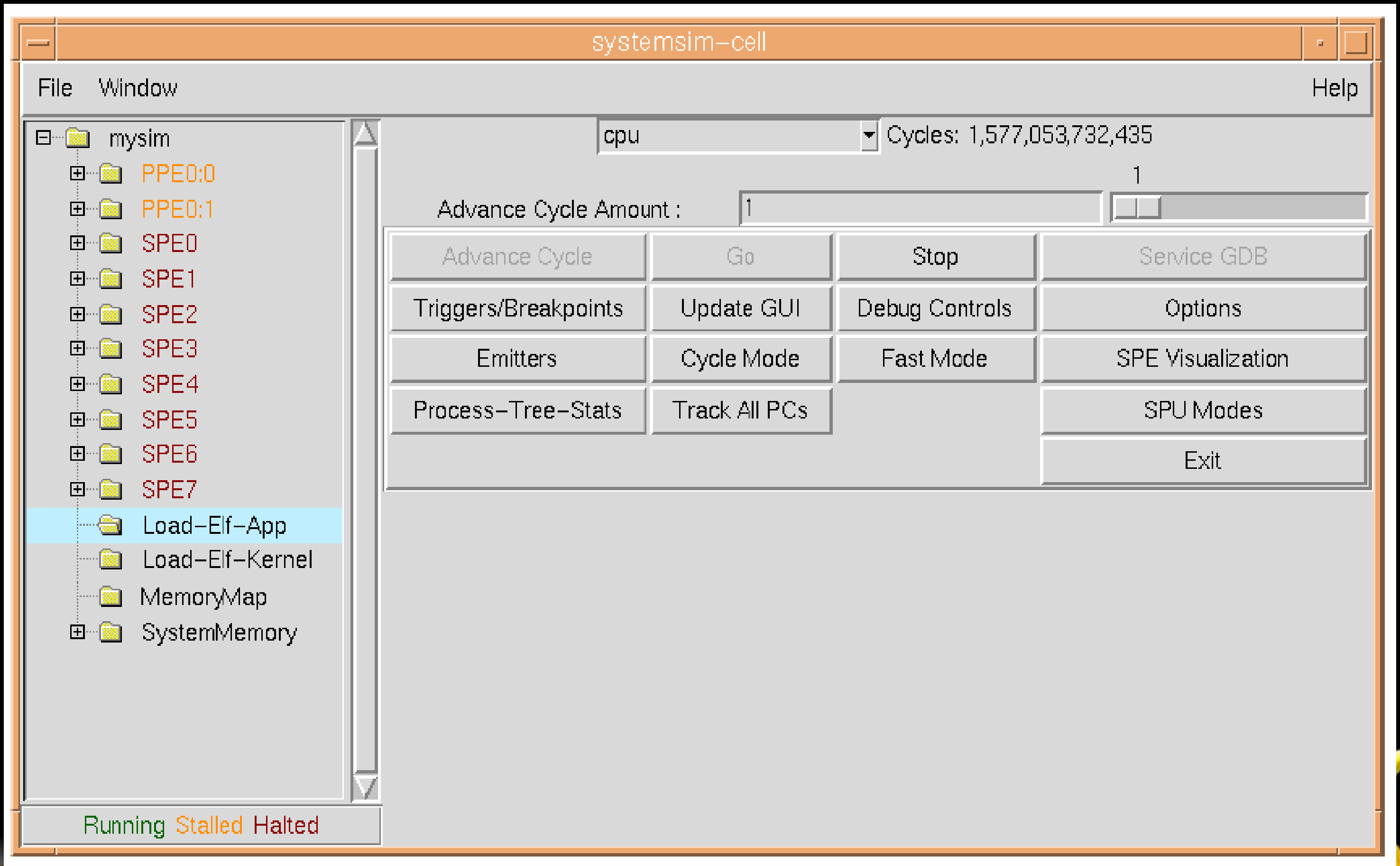
# Cell BE Resources

- IBM has released a full-system simulator for the Cell BE Processor
  - http://www.alphaworks.ibm.com/tech/cellsystemsim
  - Part of the Cell SDK: http://www.alphaworks.ibm.com/tech/cellsw
  - Can get games working in simulation
- Can also boot Linux on the PlayStation 3
  - Unfortunately can only run text-based apps or games as there is some dispute with Nvidia over drivers and access to the video hardware

# IBM Full System Simulator

- Current version 1.0.1
- Runs on x86 Linux (FC4), patched 2.6.15 kernel
  - May work on other flavours..
- Simulates entire CBE system
  - Cycle-accurate SPU simulation
  - Non-cycle-accurate PPE & MFC simulation
- Compile on sytemsim, run directly on hardware

# System simulator

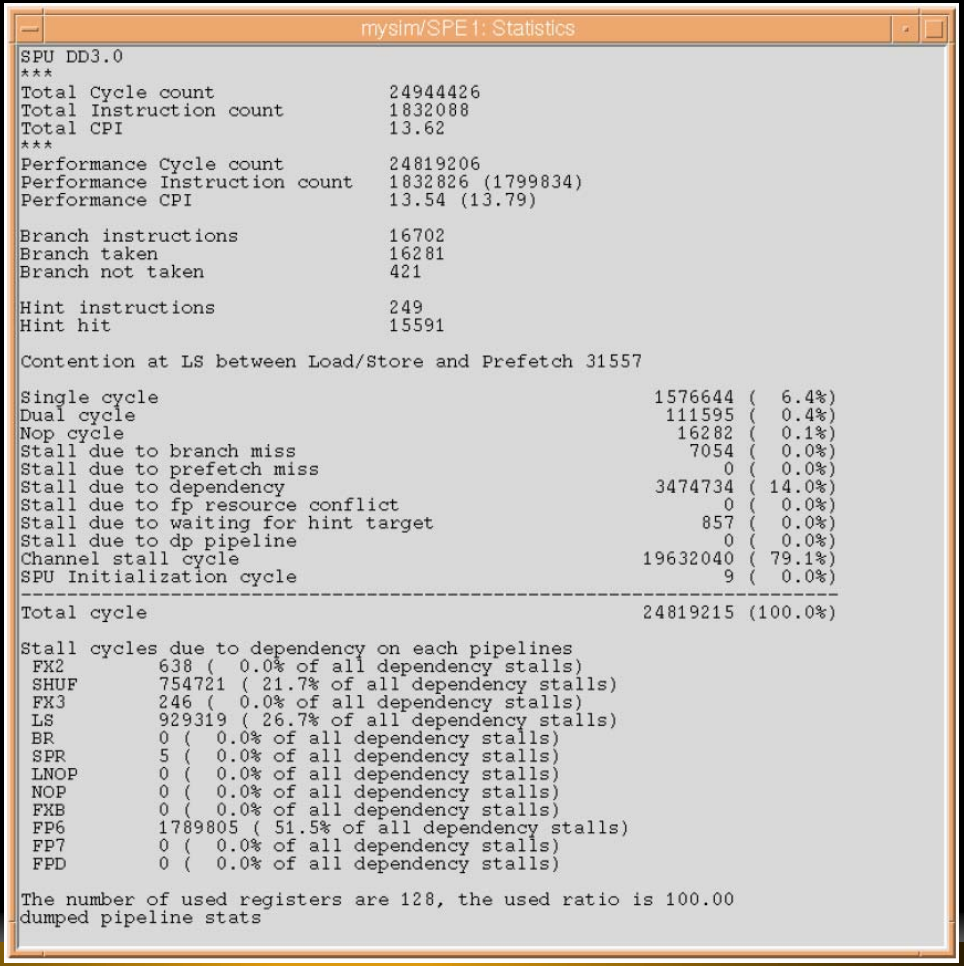


cpu

Cycles: 1,577,053,732,435

Advance Cycle Amount : 1

# System simulator



```
SPU DD3.0
***
Total Cycle count                24944426
Total Instruction count          1832088
Total CPI                        13.62
***
Performance Cycle count          24819206
Performance Instruction count    1832826 (1799834)
Performance CPI                  13.54 (13.79)

Branch instructions              16702
Branch taken                     16281
Branch not taken                 421

Hint instructions                249
Hint hit                         15591

Contention at LS between Load/Store and Prefetch 31557

Single cycle                                           1576644 (  6.4%)
Dual cycle                                              111595 (  0.4%)
Nop cycle                                                16282 (  0.1%)
Stall due to branch miss                                  7054 (  0.0%)
Stall due to prefetch miss                                   0 (  0.0%)
Stall due to dependency                                3474734 ( 14.0%)
Stall due to fp resource conflict                            0 (  0.0%)
Stall due to waiting for hint target                       857 (  0.0%)
Stall due to dp pipeline                                     0 (  0.0%)
Channel stall cycle                                   19632040 ( 79.1%)
SPU Initialization cycle                                     9 (  0.0%)
-----------------------------------------------------------------------
Total cycle                                           24819215 (100.0%)

Stall cycles due to dependency on each pipelines
 FX2        638 (  0.0% of all dependency stalls)
 SHUF       754721 ( 21.7% of all dependency stalls)
 FX3        246 (  0.0% of all dependency stalls)
 LS         929319 ( 26.7% of all dependency stalls)
 BR         0 (  0.0% of all dependency stalls)
 SPR        5 (  0.0% of all dependency stalls)
 LNOP       0 (  0.0% of all dependency stalls)
 NOP        0 (  0.0% of all dependency stalls)
 FXB        0 (  0.0% of all dependency stalls)
 FP6        1789805 ( 51.5% of all dependency stalls)
 FP7        0 (  0.0% of all dependency stalls)
 FPD        0 (  0.0% of all dependency stalls)

The number of used registers are 128, the used ratio is 100.00
dumped pipeline stats
```

# PSP Resources

- http://ps2dev.org/
- Lots of other tools for the PSP here
  - http://ps2dev.org/psp/Tools
  - But no simulator

# This Presentation

- Can be found online here:

  http://csl.cs.ucf.edu/~heinrich/GameProcessors.pdf