Introduction

Communications in the Mutiprocessor System on Chip CELL

Paul Amblard, (Ludovic Demontes)

Lab. TIMA

Grenoble, France

Paul.Amblard@imag.fr

Ludovic.Demontes@imag.fr

CELL : presentation

- Multi-processor architecture
- Definition, Design, Fabrication : IBM, Sony, Toshiba
- Aims of version 1 : Playstation3, HD TV
- Associating many MPSOC CELLs : intensive computation
- Pictures from publications
- References : IBM web site, patents, papers ($>$5 K pages)

IBM/Sony/Toshiba

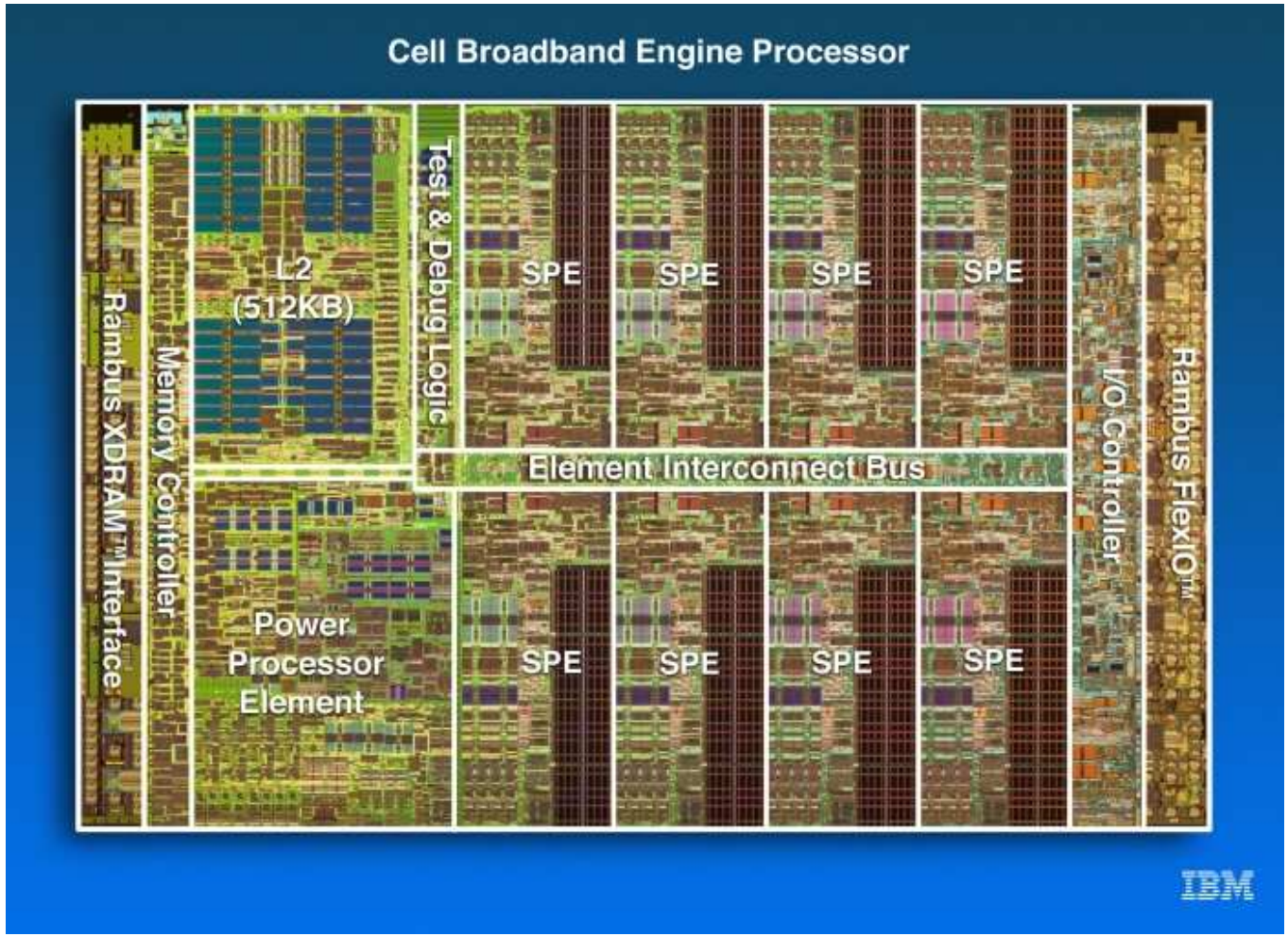


Figure 1: Photo of a Cell

SLS-TIMA : presentation

- From
  - a library of (hardware /software) blocks
  - an "application" program, splitted in fixed tasks
  - a description of a hardware architecture
- To
  - generation of hardware interfaces (?)
  - generation of software interfaces integrated in a "custom" O.S.
  - generation of simulation interfaces
  - simulation, exploration of hardware architectureS

Outline

- Presentation of architecture
  - hardware architecture
  - memory architecture
  - floating point
  - software architecture
  - communication architecture
- Hardware interfaces and communication primitives
- Programming

Hardware architecture

- One processor Power PC + cache L1 + cache L2
- Set of SPE (synergistic processing elements) (typ 8)
  - processor : instruction set different from PowPC
  - SPproc processor : many registers (128 X 128bits)
  - no cache (not yet)
  - local memory (Local Storage)
  - a "memory flow controller" (MFC) (super DMA including MMU)
- Input/Output controller
- Memory controller (for external Mem)
- Interconnection bus (high bandwidth, many simultaneous exchanges)

IBM/Sony/Toshiba

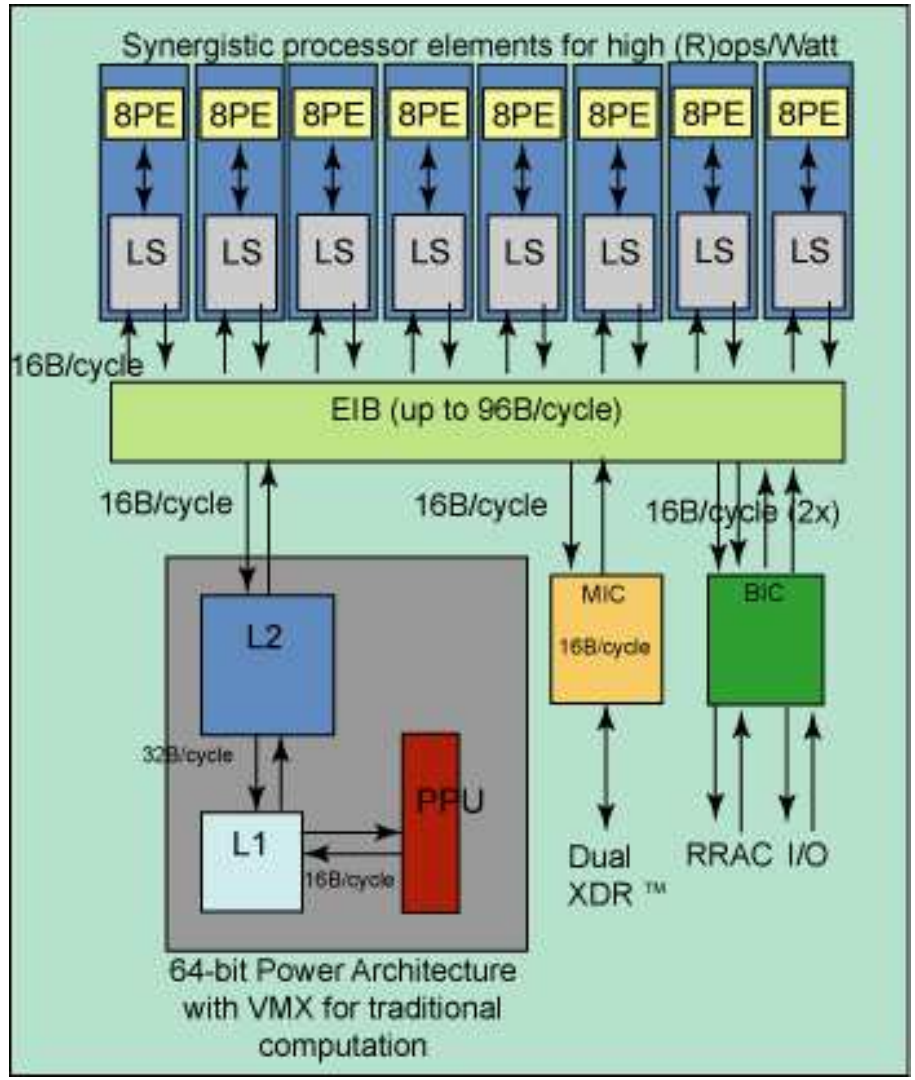


Figure 2: Global view

Memory architecture

- Organization of addresses
  - effective addresses (in PPC, in I/O, in DMA)
  - virtual address (obtained from EA by segmentation)
  - real address (obtained fom VA by pagination)
- Where do these addresses point to ?
  - physical external memory + (disk by translation)
  - local storages of SPEs
  - control registers of MFCs or I/O controllers
- Mechanisms
  - 2 levels of cache in PowPC (managed by hard)
  - "virtual" memory in MMU (PowPC AND MFCs)
  - local storages of SPEs

Memory architecture

- The SPE processor (SPproc) accesses its Local Storage by a physical address
- Small differences in translation by PowPC's MMU or MFCs' MMU (Controllability, size of pages..
- Translations can be inhibited
- Local Storage of SPE_i belongs to MFC_j-DMA addressable space

IBM/Sony/Toshiba

Figure 2-4. Storage Domains

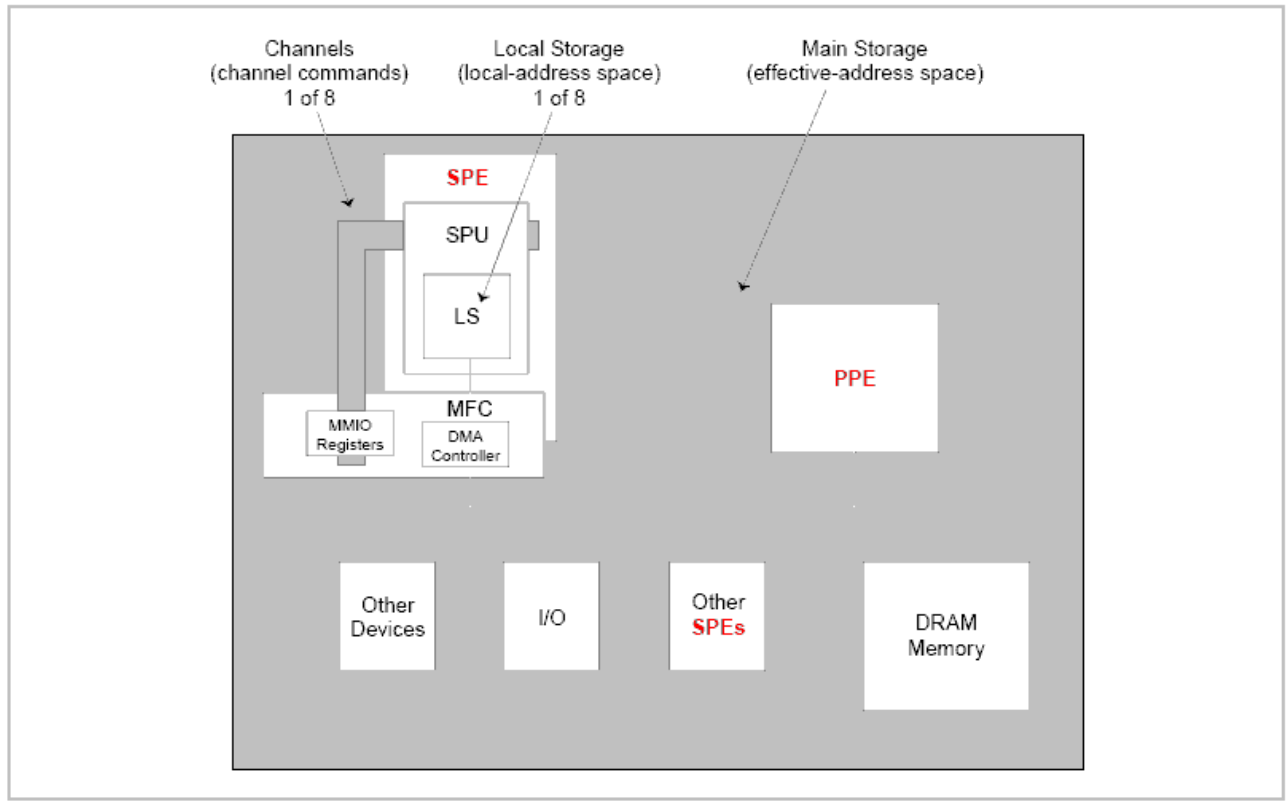


Figure 3: Addresses domains in a cell

IBM/Sony/Toshiba

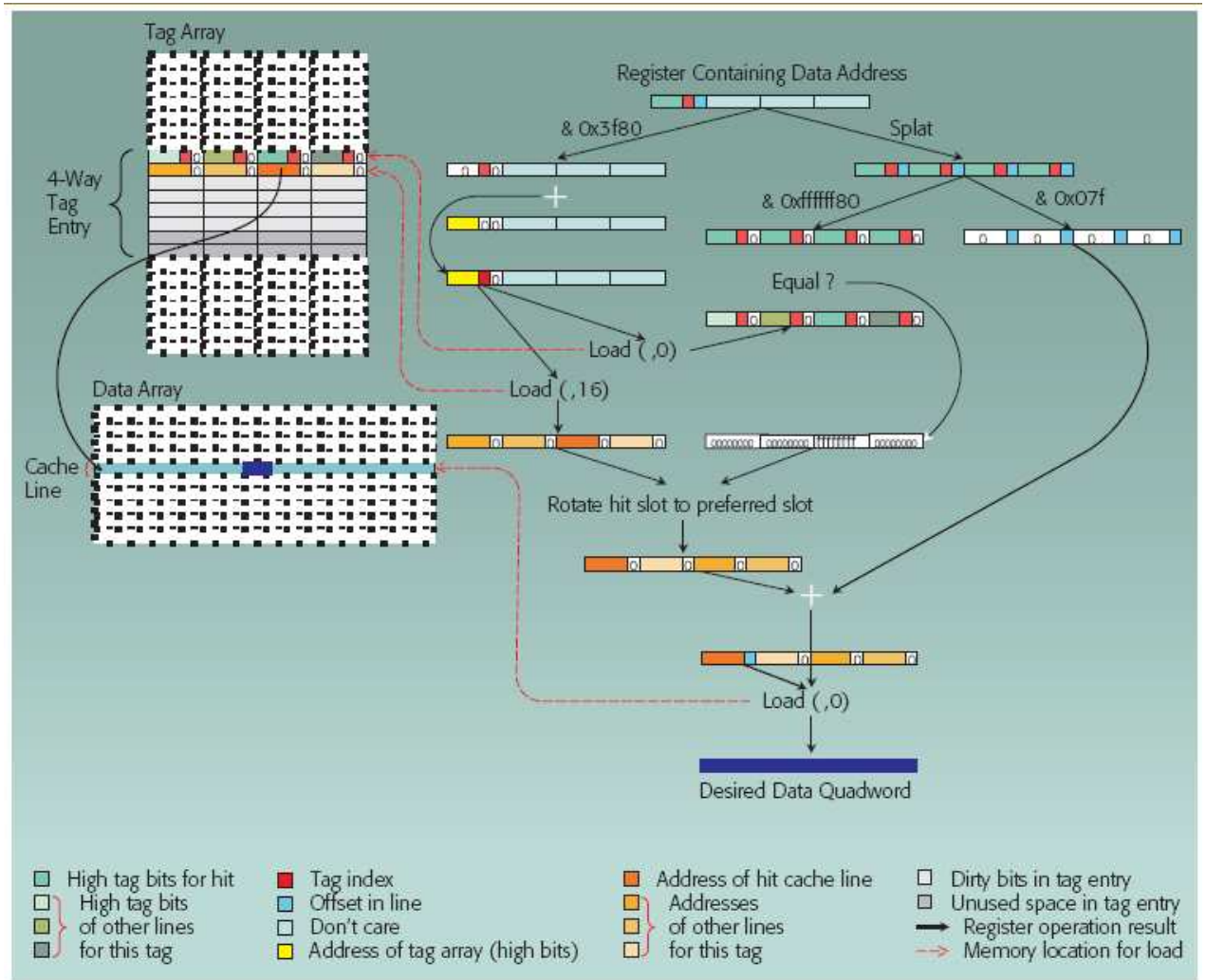


Figure 4: Cache mechanism (?)

Floating point computation

- Standard IEEE format in Power PC
- Two formats in SPE
- Idea : games, pictures, video streams are rounding-(by-truncation)-tolerant
  - Short format : fast (6 clock cycles), not IEEE compliant
  - Floating (and integer) : 4 simultaneous operations in a SPproc (in Single Precision); basic operation : A * B + C
  - Long format (64 b): slow (10 cycles), IEEE compliant

IBM/Sony/Toshiba

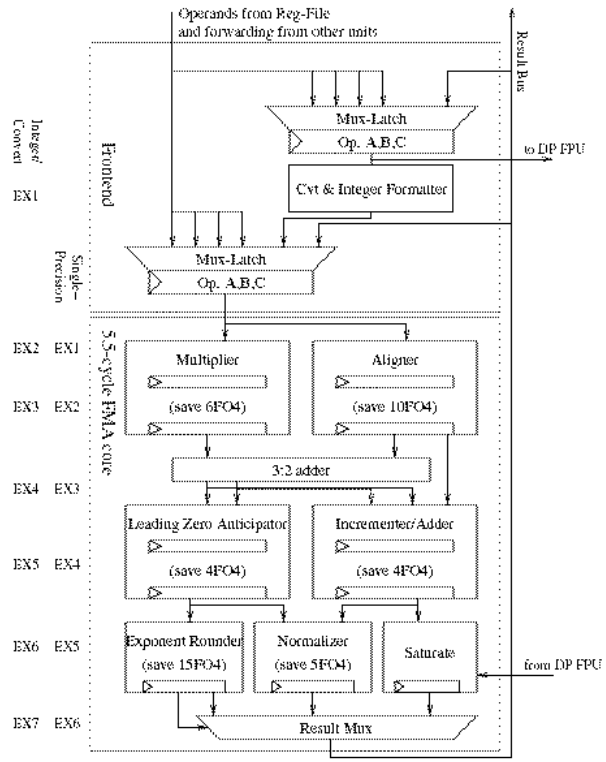


Figure 5: Floating 1 (Single Precision)

IBM/Sony/Toshiba

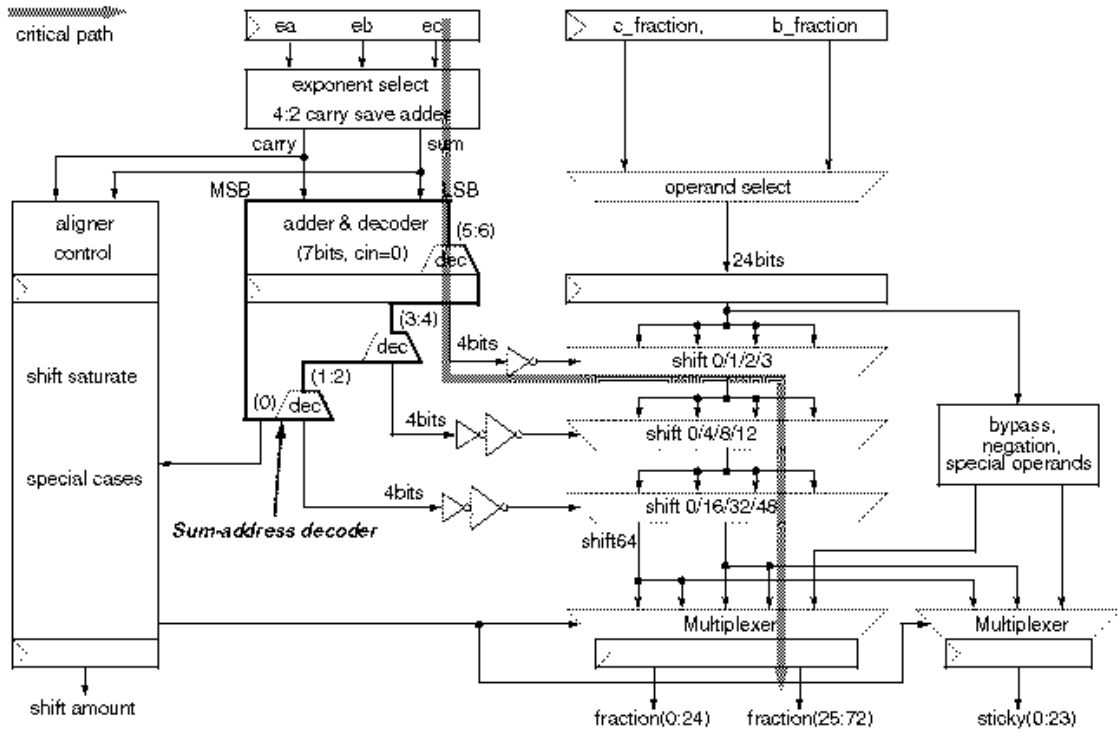


Figure 3. SPfpu Aligner. The timing critical path is marked in grey.

Figure 6: Floating 3

IBM/Sony/Toshiba

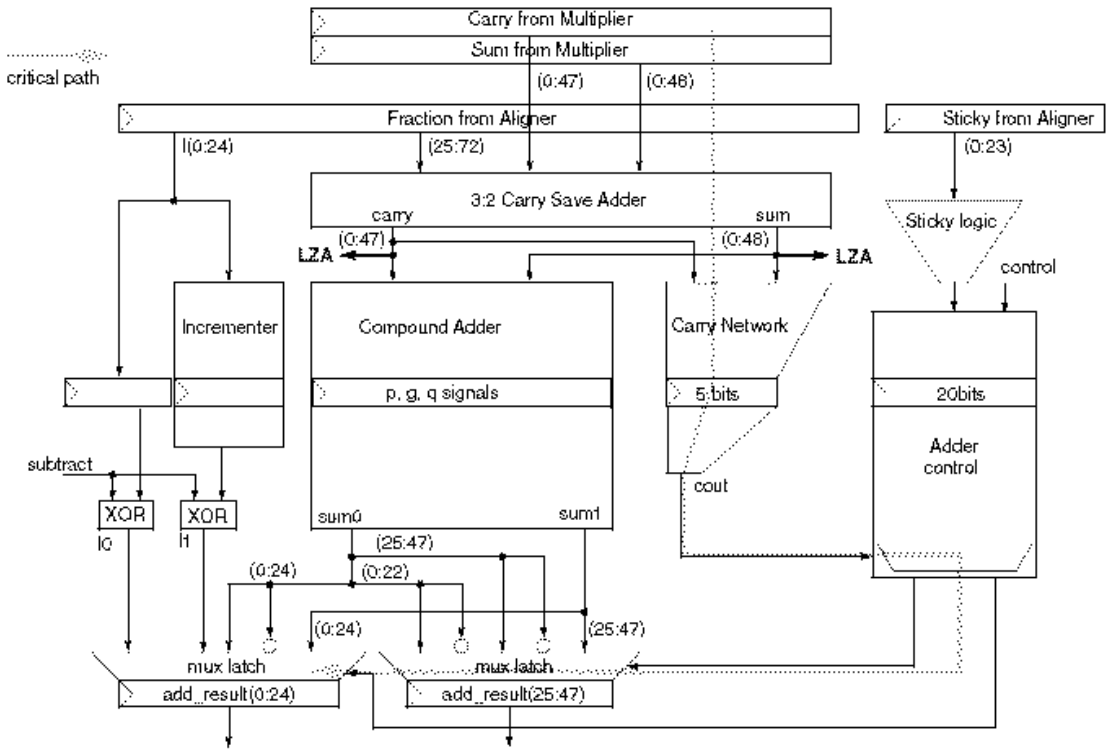


Figure 4. SPfpu Fraction Adder. The timing critical path is marked in grey.

Figure 7: Floating 2

Software architecture in brief

- The PowPC
  - runs the OS of the global machine (typ Linux)
  - manages tasks coordination (sending, receiving acknowledge..) for the SPEs
- SPEs run long intensive repetitive computations (int, float, vectors, simd) for games, Digital tv codec

Software architecture

- SPE tasks may change when time flows
- Split the applications in 10 tasks *for any given instant*
- Manage the flow of application parts (code)
- Manage the flow of data (2 families of view) (Mixing ?)
  - SPE1 computes task1 in its LS, transmits results to LS of SPE2, SPE2 computes task2 in its LS, transmits results to LS of SPE3, ..
  - SPE1 executes task1 on data in its LS, then task2, ...
- Manage the flow of control messages (synchro,..)
- SIMDization of code (not here)
- Dynamic allocation of processor to tasks : not in cell, not in SLS

Communications architecture

- The basic mechanism : channel between a SPproc and its associated MFC
- Who does initiate communication of MFC_i ?
  - PowPC, I/O controller or SPE_j (j different from i)
  - SPprocessor_i
- The three basic communications (using channels)
  - mailboxes
  - DMA transfer
  - signal notification

IBM/Sony/Toshiba

Figure 6-1. Typical MFC Block Diagram

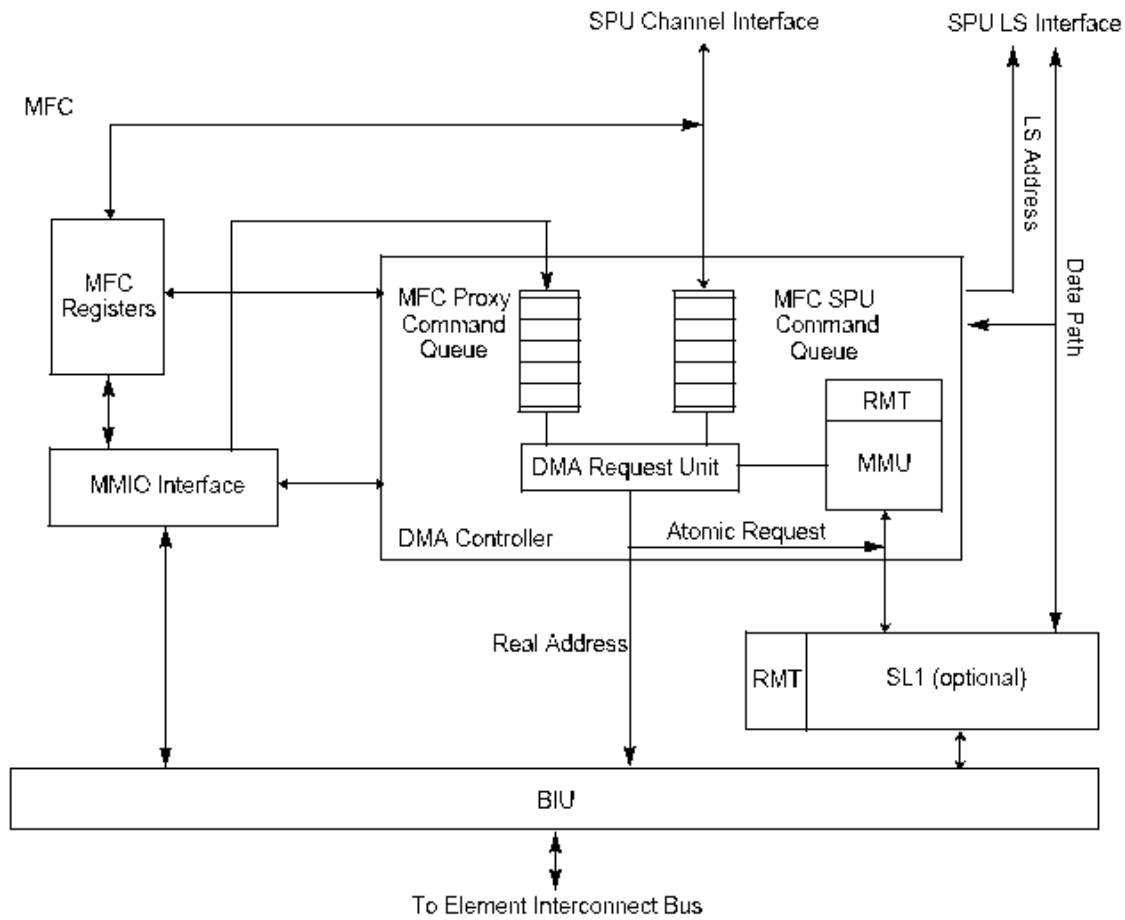


Figure 8: Memory Flow Controller

Communications architecture

- Many possibilities of synchronization
  (How to make sure that messages have been received)
- Interruptions (PowPC and SPproc)

Outline

- High speed presentation of architecture
- Hardware interfaces and communication primitives
  - : The basic mechanism : channels
  - : Mailboxes
  - : DMA Transfer
  - : Signal notification

Communications : channels

- Channels : part of the MFC
- Channel : basic communication between MFC and (its SPproc or PowPC)
- One Channel = (Direction, blocking, data fifo, count)
  - direction is from/towards the SPproc
  - blocking (or not) given by a bit in the SPproc status-word
  - data fifo
  - count in the fifo (free for SPproc-write, occupied for SPproc-read)
- Dedicated instructions in the SPproc (read channel$_x$, write channel$_y$, read channel count$_z$)
- 32 Dedicated channels between an SPproc and its MFC

Channels : behaviour

- Input Channel
  - PowPC writes into the channel, occupied-count ++
    - overwrites the fifo-end if already full
  - SPproc-read :
    - non blocking : SPproc gets the word, occupied-count - -
    - blocking and count = 0 : SPproc stalls
    - blocking and count not = 0 : SPproc gets a word, occupied-count - -
- Output channel
  - SPproc-write :
    - non blocking : SPproc puts the word, free-count - -
    - blocking and count = 0 : SPproc stalls
    - blocking and count not = 0 : SPproc puts the word, free-count - -
  - PowPC reads from the channel, free-count++

Channels

- A count modification can trigger an Interrupt or an Event (to SPproc or to PowPC)
- The PowPC can change the blocking bit of a given SPproc
- The PowPC does not stall
- A channel data word is read/written by PowPC (or other) as a control word of MFC, belonging to the address space
  (remember : translations managed by soft on PowPC)

Channels use (examples)

- Signal notification : output channel 3
- DMA command
  - address of the block in Local Storage : channel 16
  - size of the block to be transferred : channel 19
- Mailbox facilities
  - output mailbox : channel 28
  - input mailbox : channel 29

Mailboxes

- Blocking channel (mandatory)
- Output : SPproc makes `writechannel`, PowPC reads at an address mapped to this word
- (light variation : one output mailbox is read only if PowPC is in privileged mode)
- Typical use :
  - SPproc computes
  - DMA sends results into Memory (mapped to Central memory or Local Storage of an other SPE)
  - DMA signals end of transfer to SPproc
  - SPproc sends a mail to (PowPC or other SPE)
- $\Longleftarrow$ Take care of synchro !!

DMA transfers

- The MFC has an input channel (from its SPproc) with a queue of "commands" (See old litterature at ibm about channel processors in ibm 360...)
- These commands may have an identifier : group of commands
- Transfer or list of transfers
- Info about transfer in channels (non blocking, size 1)
  - Size of the block
  - Address in Local Storage
  - Effective Address in "Memory Space", translated "on-the-fly"
- These three words are written by SPproc (channels) or PowPC (control registers)

DMA transfers

- A DMA transfer can be executed while the SPproc "computes"
- A DMA transfer can be executed while the PowPC "computes"
- Several DMA transfers can be executed simultaneously
  - 4 "concentric" rings
  - One ring = many slices
- (Obvious problems of coherence)

IBM/Sony/Toshiba

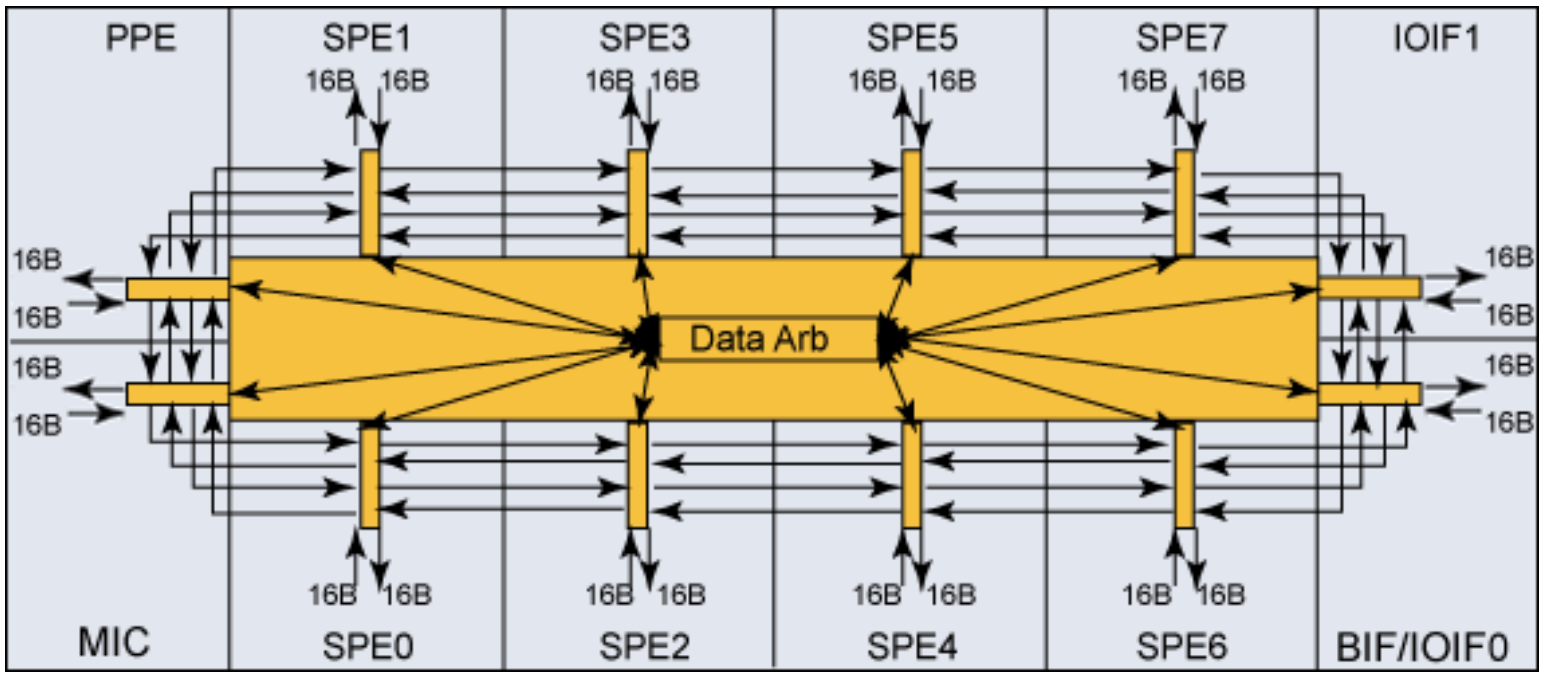


Figure 9: Internal bus for communications

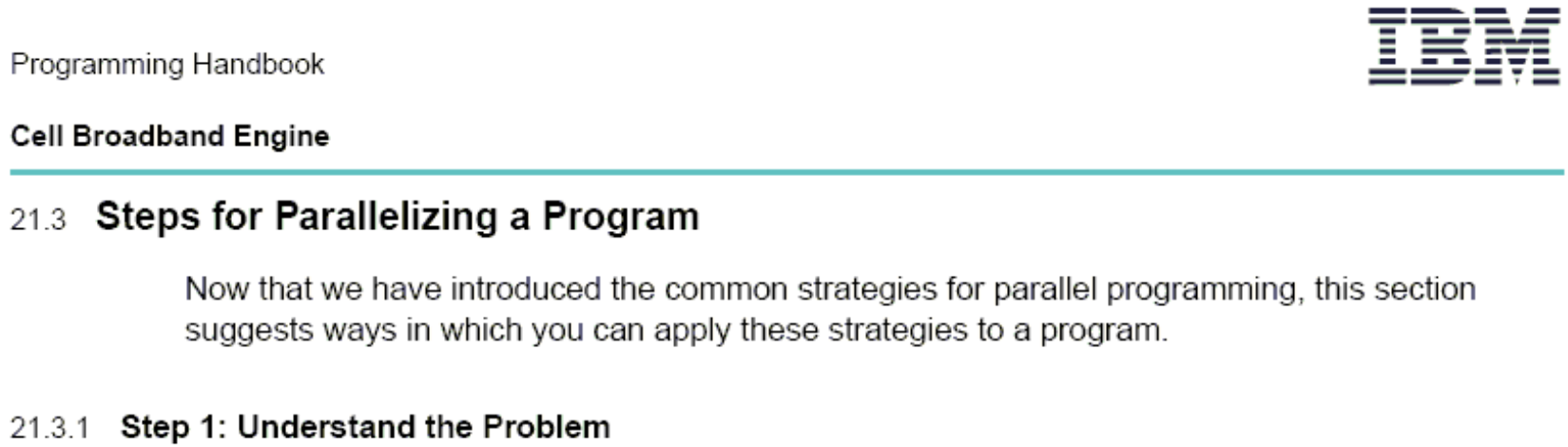
Signal notification

- Input blocking channels, size 1
- SPproc reads the data $\Longrightarrow$ atomic reset the word (ack)
- PowPC (or ctrl IO) writes the word : two modes
  - direct write (normal, One-to-one signaling)
  - OR-Accumulation in the word (Many-to-one signaling)

Outline

- Presentation of architecture
- Hardware interfaces and communication primitives
- Programming
  - Simulator : available on IBM Site, needs a PC with Fedora Core
  - Machines
  - Playstation 3 + Hardware hacking
  - SIMDization ??????????????????

En guise de conclusion !



Programming Handbook

IBM

Cell Broadband Engine

21.3 **Steps for Parallelizing a Program**

Now that we have introduced the common strategies for parallel programming, this section suggests ways in which you can apply these strategies to a program.

21.3.1 **Step 1: Understand the Problem**

Figure 10: Tout ce que vous vouliez savoir sur la programmation parallèle, ... et que vous n'aviez jamais osé demander