Lames QS20 au CINES

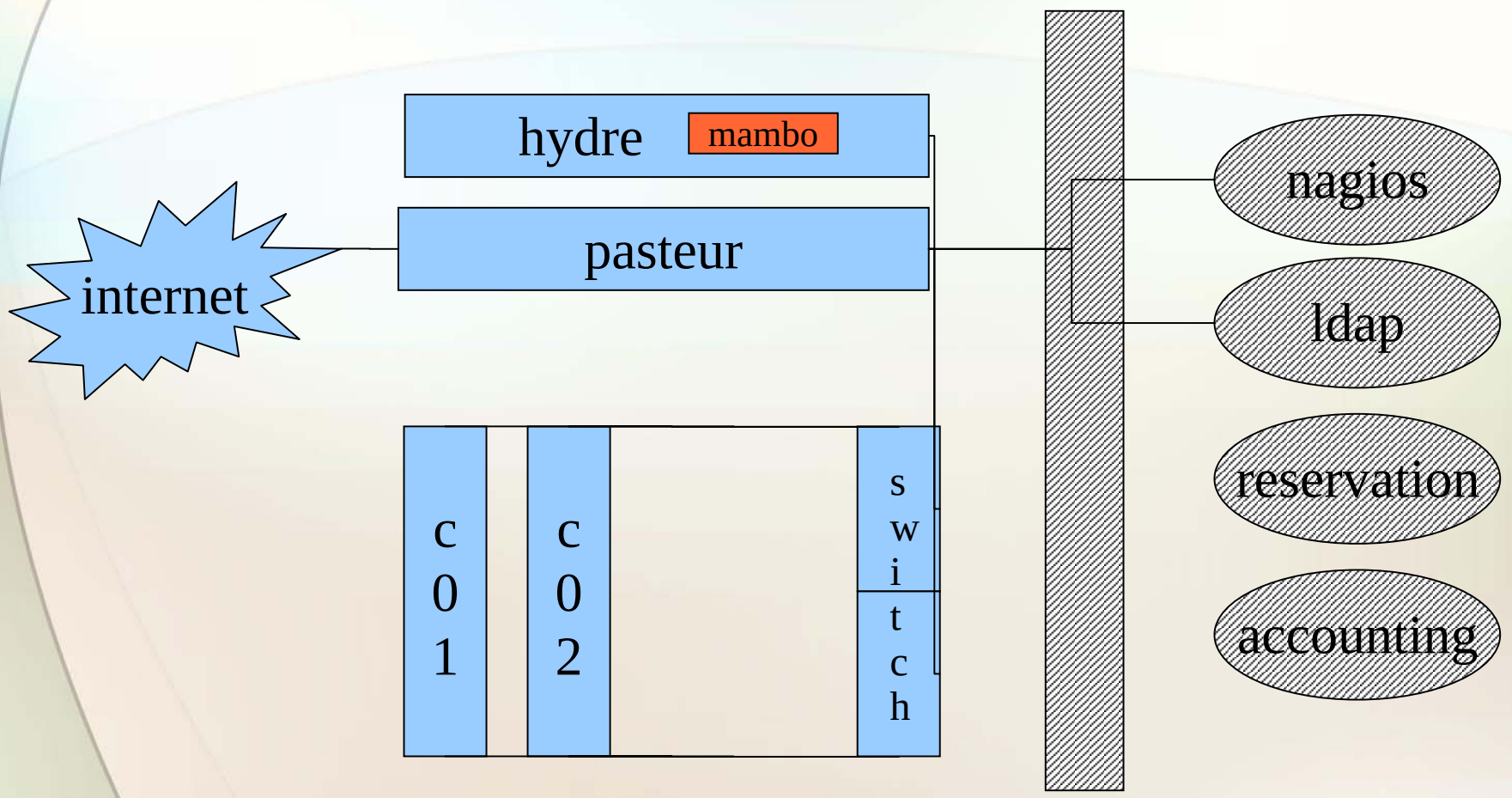
B. Cirou et G. Gil

Intégration et utilisation

Présentation

- Intégration au CINES
  - Matériel
  - Services
- Utilisation
  - soumission de Job
  - Utilisateurs
  - Programmation
- Futur

Intégration au CINES



Matériel au CINES

- noeud de connection (pasteur)
  - x86_64 (Xeon 2.33GHz)
- noeud simulateur (hydre)
  - x86_64 (Xeon 3GHz)
- 1 IBM Blade Center M8677-3RG
  - 2 Blades IBM QS20 (Cell 3.2GHz)
  - 2 switch Giga-ethernet Nortel

Matériel: Différences QS20 / PS3

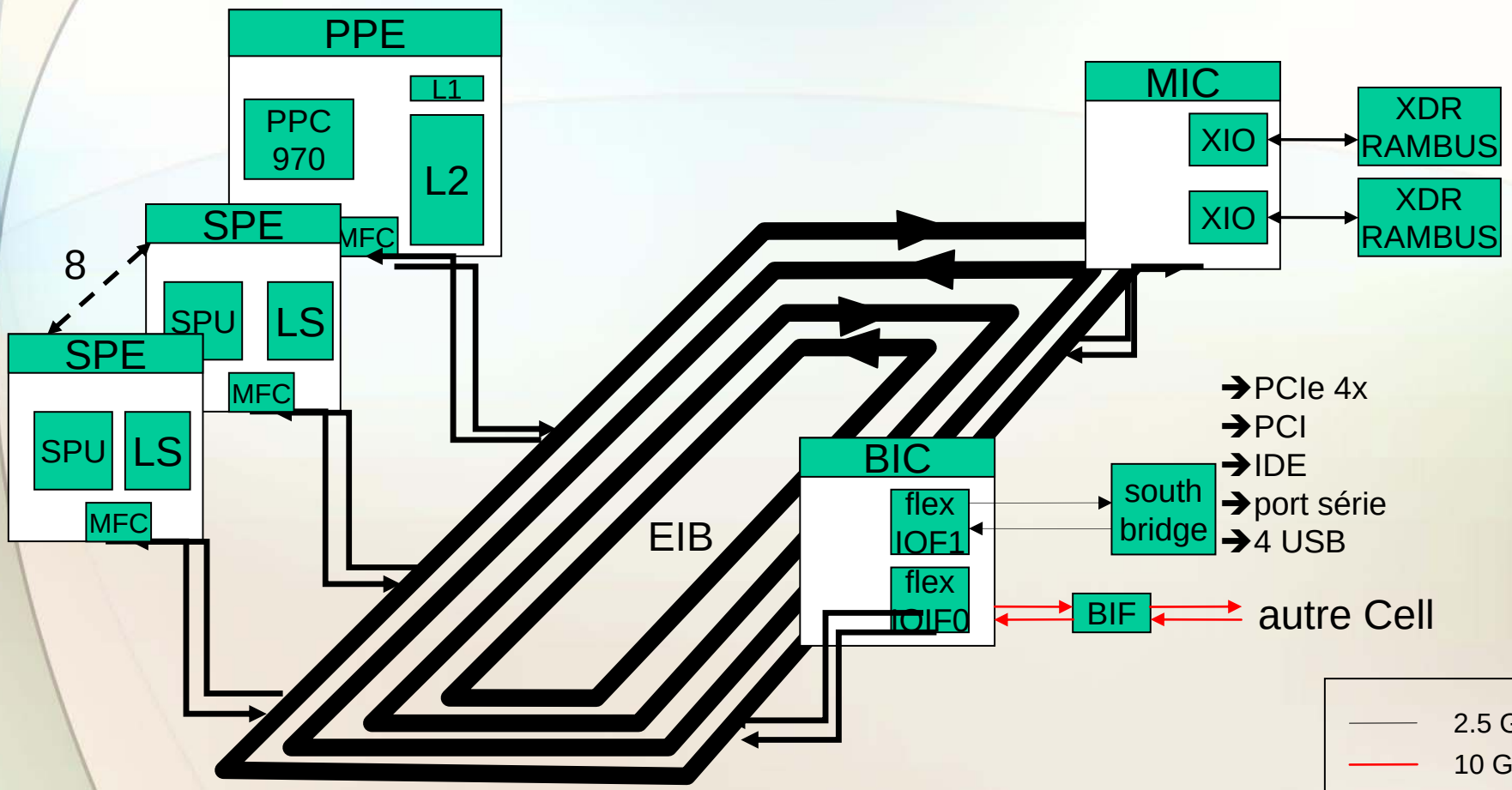
- SMP 2-way avec 2 Cell
  - 16 SPE accessibles par 1 processus
- 2x512MB de Rambus en accès NUMA
- 40GB hard disk
- 1 port miniCOM
- 1 slot PCI-express
- 2 Gb ethernet
- No sound
- No RSX Nvidia chipset
- No DVD/Blue-Ray
- No USB No SD/MMC Reader

Matériel: Cell@3.2GHz

- 1 PPE (64 bits)
  - PowerPC 970 (2-way Simultaneous Multi-Threading)
  - Vector Multimedia eXtension (VMX Altivec)
    - 32 registres 16 Bytes
  - L1 32kB instr. + 32kB data
  - L2 512 kB
  - Rambus 512 MB
- 8 SPE (32 bits)
  - Vectoriel
    - 128 registres de 16 Bytes
  - SRAM 256 kB
    - Latence écriture 4 cycles, lecture 6 cycles
  - Instructions SP non-conforme IEEE 754
  - Instructions DP non « fully-pipelined »
  - Sans prédiction de branchement

  ➔ moins de transistors, plus de GHz et plus de puces par wafer

Matériel: architecture du Cell

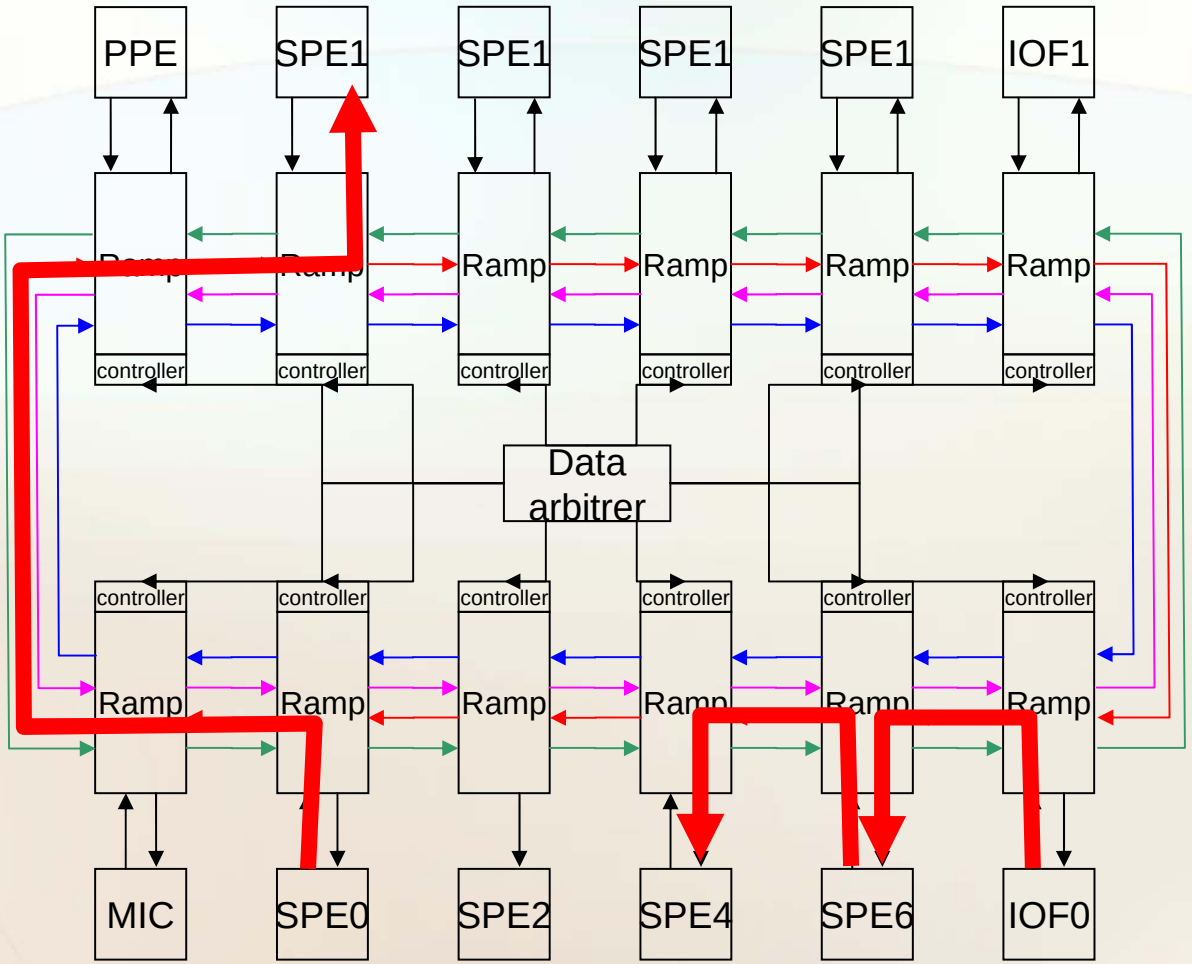


BIC: Broadband Interface Controller
BIF: Broadband interface
EIB: Element Interconnect Bus
IOIF: I/O interface

LS: Local Store
MIC: Memory Interface Controller
MFC: Memory Flow Controller
PPE: PowerPC Processing Element

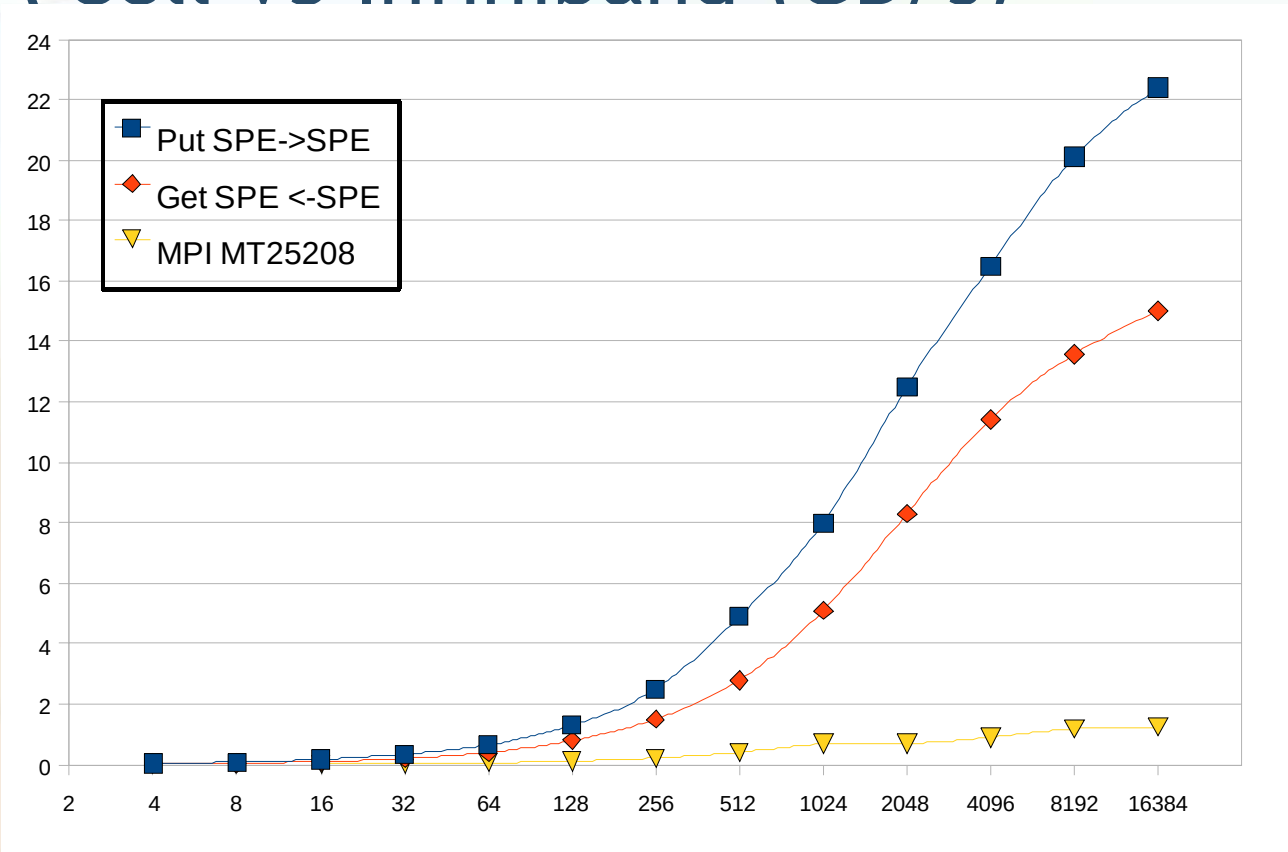
SPE: Synergetic Processing Element
SPU: Synergetic Processing Unit
XDR: eXtended Data Rate
XIO: XDR I/O

Matériel: architecture de l'EIB



Matériel: Bande passante

- DMA Cell VS Infiniband (GB/s)



Services

- Frontal Fedora 7 (Pasteur)
  - System imager 3.8 (Pb: install from scratch)
  - NFS 140 GB Raid 1
    - export /home and /opt/QS20
  - Interactive/batch session SLURM 1.2.1
  - Cell SDK 3.0
- Simulateur Fedora 7 (Hydre)
  - Cell SDK 3.0 + Simulateur
- Noeuds de calcul Fedora 7 (c01 et c02)
  - noeud de calcul (réservation SLURM)
  - Cell SDK 3.0
- Monitoring Nagios de l'ensemble

Services: calcul et SDK

- bibliothèques de calcul SDK 3.0 exploitant SPU
  - libblas (sous-ensemble)
    - subroutines pour LU ou Cholesky
  - libfft minimale
    - fft_1d_r2 (radix 2), fft2d
  - libmass
    - trigo, exp, log, sqrt
- bibliothèque Mercury (non disponible au CINES)
  - SAL

Utilisation: soumission de jobs

- Interactive
  - cirou@pasteur:~> srun -u bash -i
- Batch MPI
  - cirou@pasteur:~> srun -b jobfile.cmd

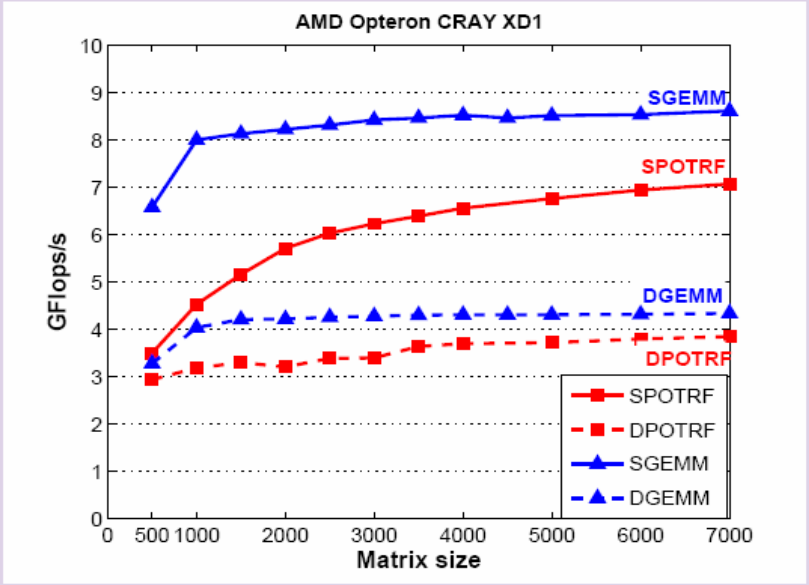
```
#SLURM -N 2 -n 2
#SLURM --time=30
echo c01-1 > /home/cirou/mpd.hosts
echo c02-1 >> /home/cirou/mpd.hosts
/opt/QS20/mpich2/bin/mpdboot --rsh=/usr/bin/rsh  -n 2 --chkup --
   verbose --ifhn=c01-1 -f /home/cirou/mpd.hosts
/opt/QS20/mpich2/bin/mpirun -np 2 hostname


/opt/QS20/mpich2/bin/mpdallexit
```

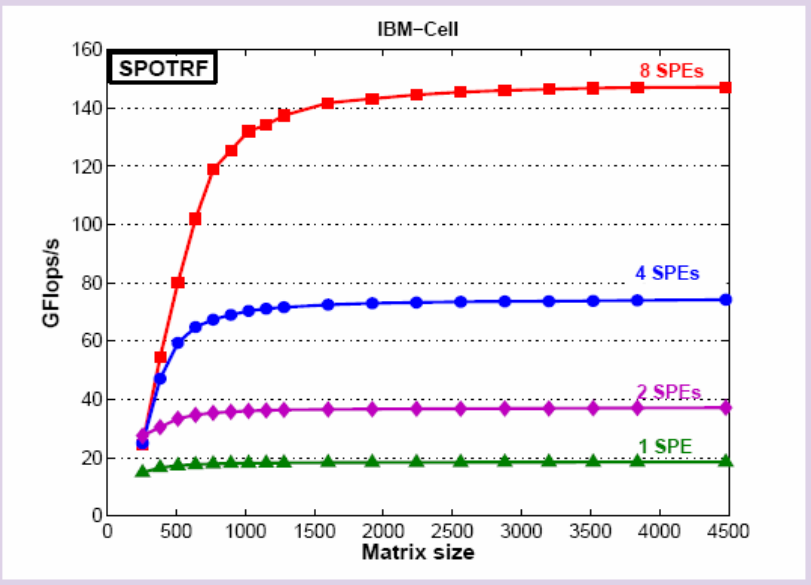
Utilisation: exemples

- A. Haidar (Doctorant de Luc Giraud, CERFACS)
  - pré-conditionneur simple précision
    - solveur Cholesky itératif écrit par J. Kurzak U-Tennessee (équipe de Dongarra)

CRAY XD1



IBM Cell



Utilisation: exemples

- J.L. Lamotte (MdC, Lip6)
  - mul, add, div double et quad précision

Fonction	Temps de calcul	Performance
Add_ds_ds_vect	50 cycles / 2	128 MFLOPs
Add_ds_ds_2vect	64 cycles / 4	200 MFLOPs
Add_ds_ds_4vect	72 cycles / 8	355 MFLOPs
Mul_ds_ds_vect	49 cycles / 2	130 MFLOPs
Mul_ds_ds_2vect	60 cycles / 4	213 MFLOPs
Mul_ds_ds_4vect	63 cycles / 8	406 MFLOPs
Div_ds_ds_2vect	111 cycles / 4	115 MFLOPs
Div_ds_ds_4vect	125 cycles / 8	204 MFLOPs
Add_qs_qs_4vect	449 cycles / 4	28.5 MFLOPs
Mul_qs_qs_4vect	583 cycles / 4	21.9 MFLOPs
Div_qs_qs_4vect	2667 cycles / 4	4.79 MFLOPs

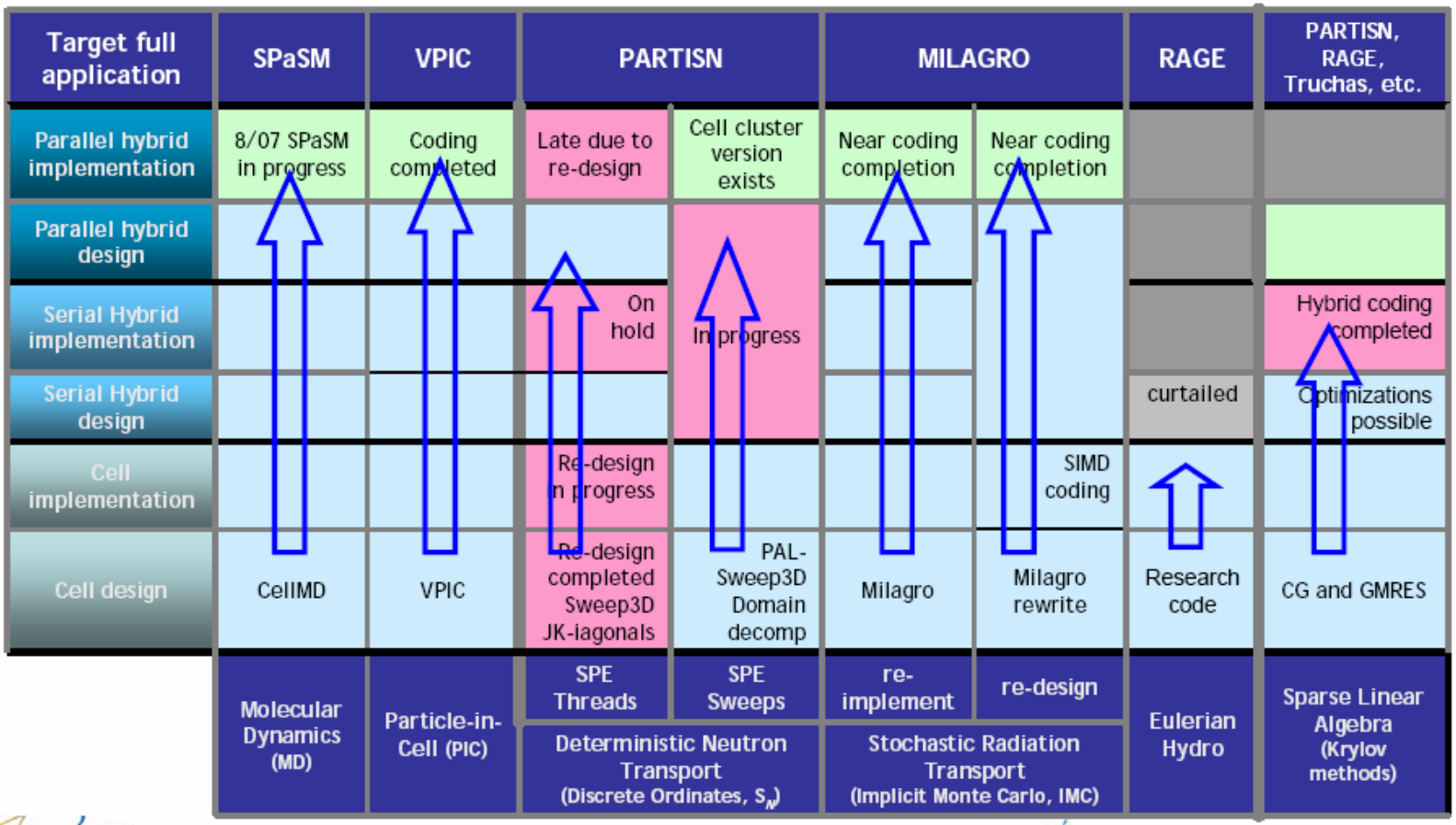
Utilisation: exemples

- P.Y. Aquilanti (stagiaire de Laurent Derrien,TOTAL)
  - Équation des ondes (differences finies)
- J. Dubois (stagiaire de S. Petiton, LIFL)
  - Algo Arnoldi

Utilisation: contacts

- Mr Tarek (CR CNRS, U-Nancy)
  - CellMD adapté par Gianni De Fabritiis (U. Barcelona) PS3grid.net
- Mr Boyrie (U. Montpellier 2) équipe CTTM (Chimie Théorique)
- Mr Dumas (Centre de Biochimie Structurale, Montpellier)
  - Superflip+lib FFTW. Calcul des structures cristalines à partir de données de diffraction.
- Mr Valiron (DR CNRS, Laboratoire d'Astrophysique de l'Observatoire de Grenoble)
  - MOLSCAT (NASA) Collision Molecule-Molecule/Atom
- Mr Fleurat-Lessard (ENS Lyon Chimie)
  - recompilation CPMD, Gaussian
- F. Bodin (Irisa)
  - QCD optimisé Cell (U-Milan). Physique des hautes énergies

Utilisation: Los Alamos



Los Alamos NATIONAL LABORATORY EST.1943

Operated by Los Alamos National Security, LLC for NNSA

WEAPONS SCIENCE & ENGINEERING CAPABILITY REVIEW

NNSA

UNCLASSIFIED

Utilisation: programmation

- C
  - bas niveau: LibSPE 2 + intrinsics SIMD (altivec)
  - bibliothèque pour programmes réguliers: ALF (Abstraction Library Framework)
  - directives pragma: CellSS
  - DaCS (Data Communication and Synchronization)
- OpenMP
  - octopiler
- C++
  - Rapidmind
- Mercury
  - MCF (multi core framework)
- Sony GameOS ($10000)
  - www.tmstation.scei.co.jp/ps3/form_e.php

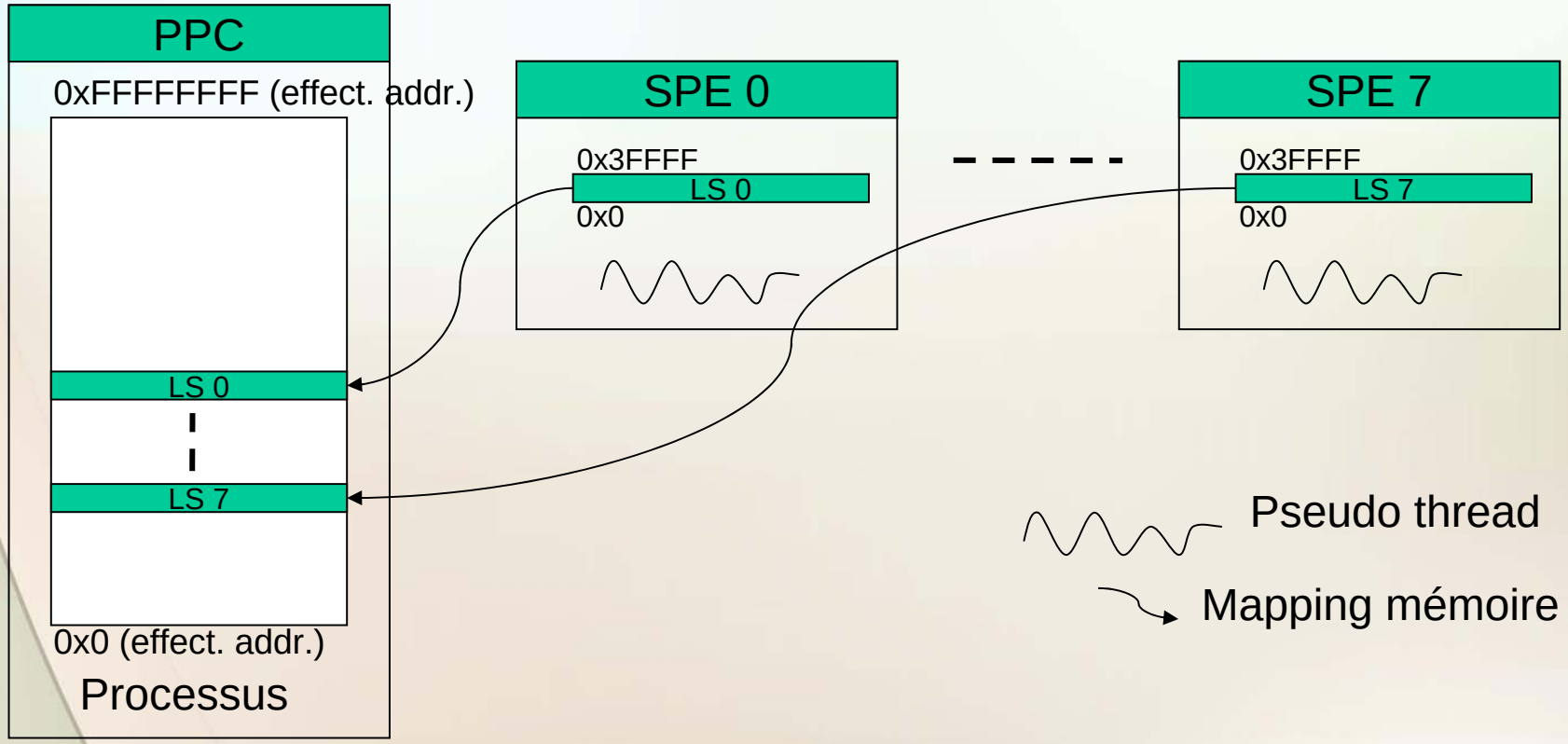
Utilisation: programmation

- Absence d'espace d'adressage unique
- Local Store du SPE != mémoire cache
  - transferts de données explicites (via DMA)
    - finesse des transferts DMA
      - Tailles 1,2,4 ou 8
      - Tailles multiples de 16 < 16384 (x128 pour performances)
    - Obligation alignement x16 (x128 pour performances)
  - adresse d'origine des données non stockées
  - obligation d'alignement des load/store (x16)

    ➔Existence instruction shuffle

Utilisation: programmation

Espaces d'adressages



Utilisation: programmation

Code PPC « toto.c »

```
1   #include <libspe.h>
2   #include <libmisc.h>
3   extern spe_program_handle_t mon_spu_symb;
4   int main(int argc,char**argv)¶
5   {
•     spe_gid_t gid;
•     spe_program_handle_t handle;
•     void *arg = malloc_align(64*1024,7);
•     void *env = NULL;
•     int dest 0 = 0, dest2 = 2;
•     unsigned long mask_dest = 1 << dest0 | 1
    << dest2;
•     int flags = 0;
•     int policy = SCHED_OTHER, int priority=0,
•     int spe_events = 0;
•     spe_gid_t gid = spe_create_group (policy,
    priority, spe_events);
```

```
1       speid_t id = spe_create_thread
        (gid,&mon_spu_binaire, &handle,argp,
        envp, mask_dest, flags);
2       int status;
3       int options = 0;
•       spe_wait (id, &status, options);
1       }
```

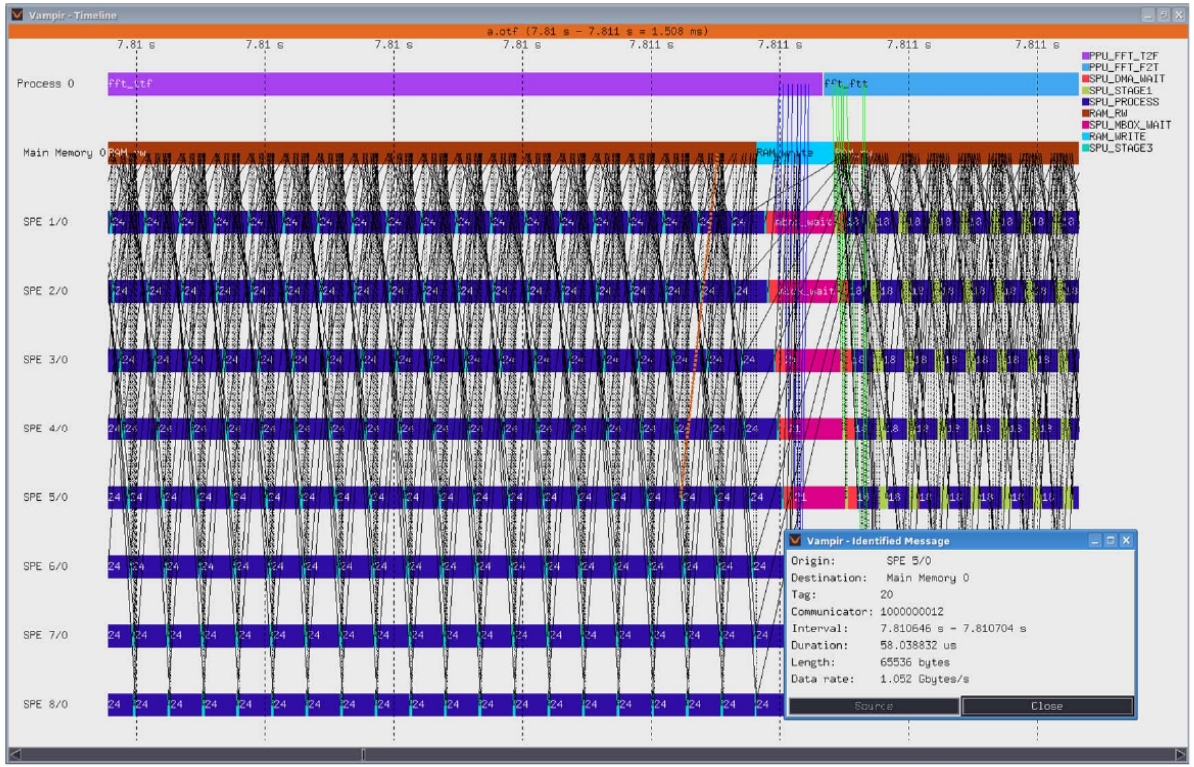
Code SPU « tutu.c »

```
1       #include <libmisc.h>
2       char buff[64*1024]
        __attribute__(aligned(128));
3       int main(uint64_t id, uint64_t arg,
        uint64_t env)¶
4       {
•         size_t n = 32768;
•         uint32_t local = (uint32_t) &buff[0];
•         size_t sz = copy_to_ls( (local, arg,
        n);
•         return 0;
1       }
```

Code SPU

Utilisation: traces

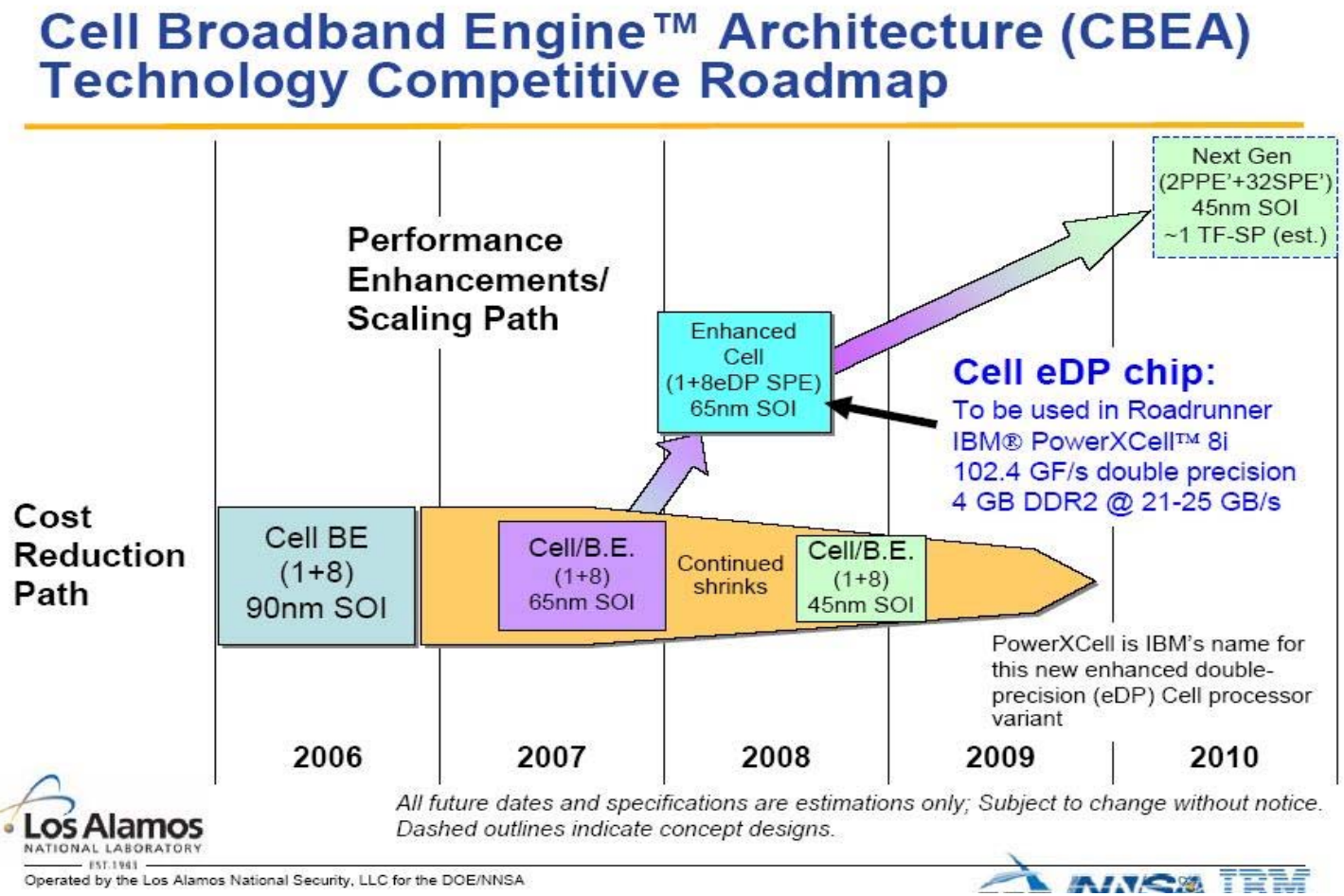
Trace Example: FFT Workload (IBM SDK), 16 MB page size



Synchronization Point, 1.5 ms Window, 50.3 GFLOPS

Futur

- Remplacement QS20 (QS21?)
- Participation à PS3Grid.net



Références

- IBM
  - SDK
    - ibm.com/developerworks/power/cell
    - bsc.org.es/projects/deepcomputing/linuxoncell/
  - Forum
    - ibm.com/developerworks/forums/forum.jspa?forumID=739
  - Journal
    - research.ibm.com/journal/rd/515/tocpdf.html
- Autres
  - mc.com
  - developer.rapidmind.net
  - ps3coderz.com
  - cellperformance.com
  - ps3grid.net

Ouverture de compte

- Tel: 04 67 14 14 80
- Formulaire http://dari.cines.fr

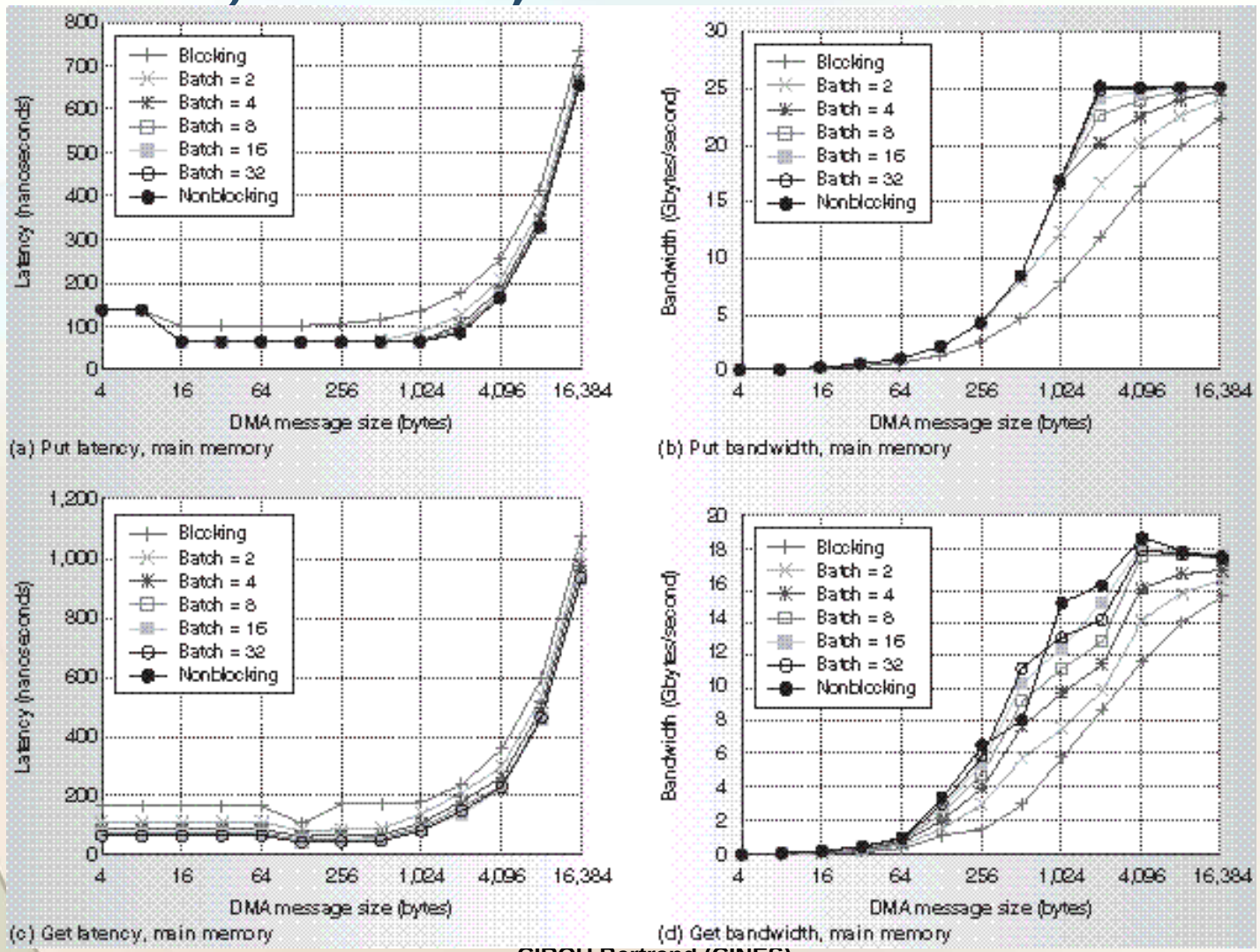
Linux: programmation des DMA

- DMA non-bloquants par défaut
- mfc_put (
  - addr32 LS
  - addr64 remote
  - size
  - tag
  - transfer id
  - replacement id)
- Symmétrique mfc_get(…)
- Variantes
  - liste de transferts DMA (suffixe « l »)
  - barrière (suffixe « b »)
  - ordre (suffixe « f »)
- Attente terminaison
  - mfc_read_tag_status (suffixe « all », « any »)

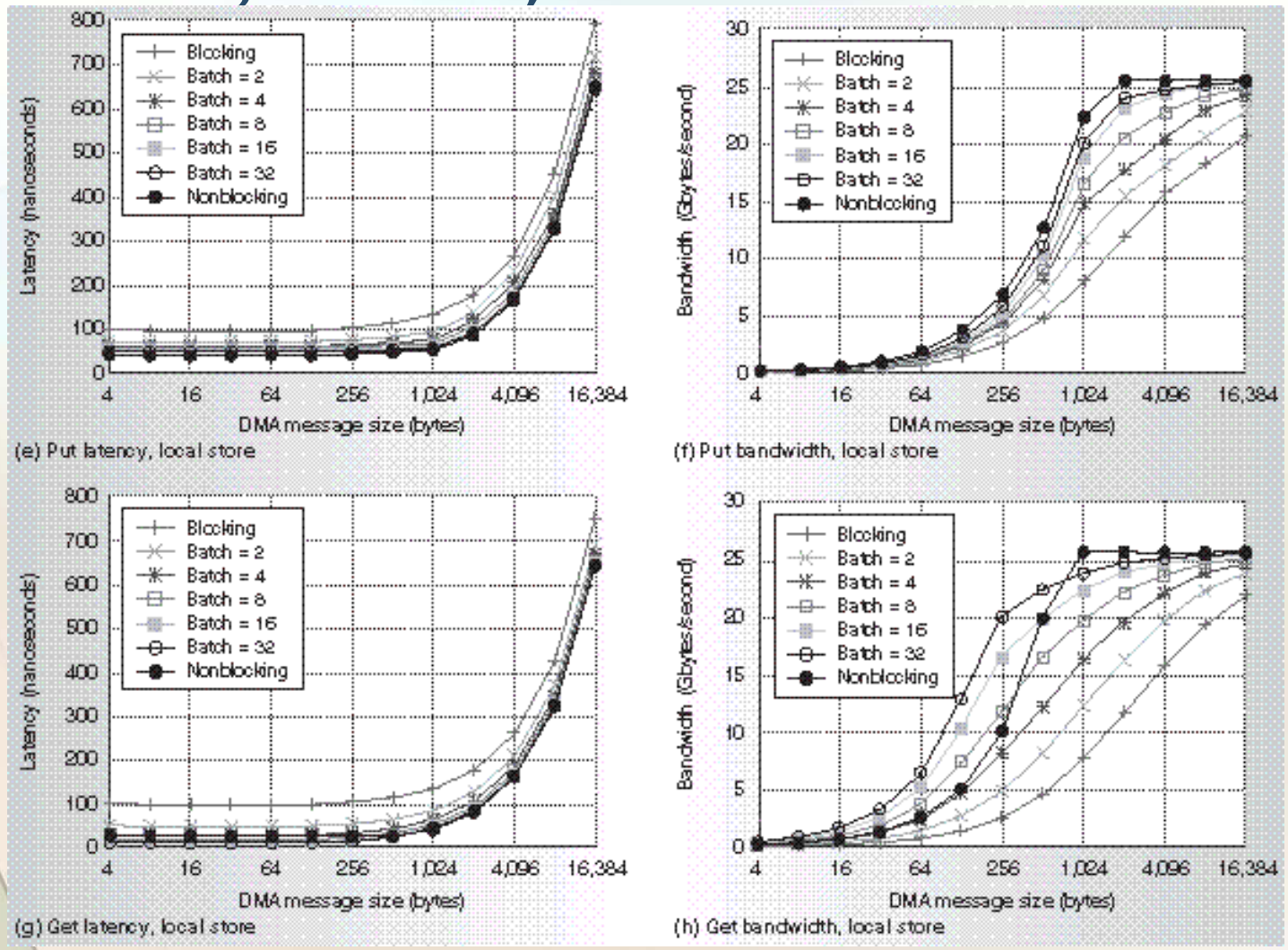
Linux: programmation mono Cell

- Synchronisations
  - Atomiques
    - mfc_getllar, mfc_putllc, mfc_putqlluc . . .
  - Mailboxes
    - int spu_read_out_mbox(speid)
    - err spu_write_in_mbox(speid,int)
  - Signaux
    - mfc_sndsig, . . .

Put/Get Cell➔Rambus (Austin, Watson, PNNL)



Put/Get Cell➔Cell (Austin, Watson, PNNL)



Programmation cluster de Cell

- Portage d'applications sans effort
  - Recompilation sur PPC 970
    - Libmpi (infiniband?)
    - Libblas SMP? permettant l'utilisation transparente des 8 SPE

  ➔grain de calcul sur 9x (~128ko) sans comm. extra-nœud

  ➔parallélisme moindre à taille de problème identique

  ➔augmenter la taille des problèmes

  ➔mais Rambus 2x512Mo par nœud

  ➔petites tailles de problèmes et faibles performances
- Solutions
  - micro-MPI IBM? sur les SPE (accès au réseau ?)
  - Repenser le parallélisme de l'application (avec efforts)