Semi-inified caches

Nathalie Drach, André Seznec

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

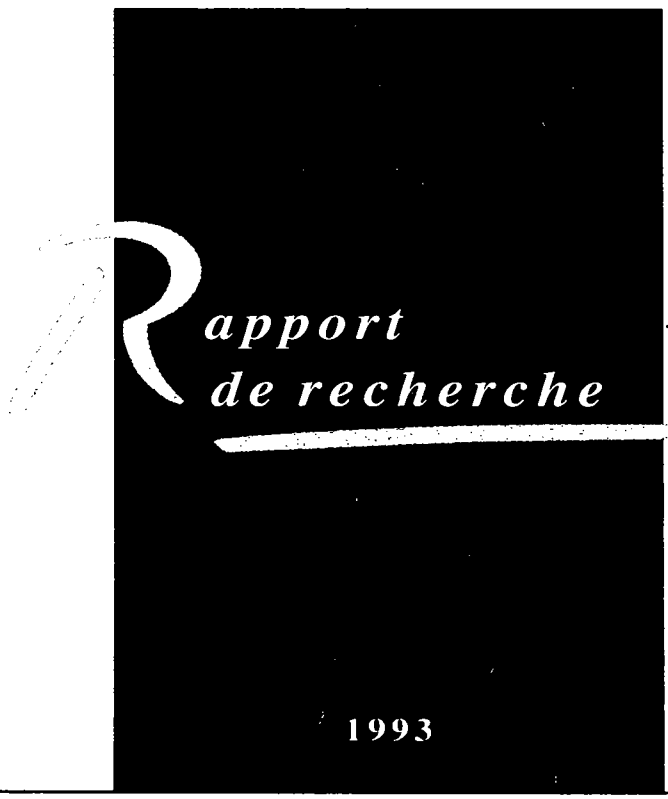
Semi-unified caches

Nathalie DRACH
André SEZNEC

N° 1841
Janvier 1993

PROGRAMME 1

Architectures parallèles,
Bases de données,
Réseaux et Systèmes distribués



IRISA

INSTITUT DE RECHERCHE EN INFORMATIQUE
ET SYSTEMES ALEATOIRES

Campus Universitaire de Beaulieu
35042 - RENNES CEDEX FRANCE
Tél : 99 84 71 00 - Télex : UNIRISA 950 473 F
Télécopie : 99 38 38 32

Semi-unified Caches

Nathalie Drach, André Seznec
Programme 1
Projet CALCPAR *

Publication Interne n° 696 - Janvier 1993 - 18 pages

Abstract

Since the gap between main memory access time and processor cycle time is continuously increasing, processor performance dramatically depends on the behavior of caches, and particularly on the behavior of small on-chip caches.

In this paper, we present a new organization for on-chip caches: the *semi-unified* cache organization. In most microprocessors, two physically split caches are used for respectively storing data and instructions. The purpose of the *semi-unified* cache organization is to use the data cache (resp. instruction cache) as an on-chip second-level cache for instructions (resp. data). Thus the associativity degree of both on-chip caches is artificially increased, and the cache spaces respectively devoted to instructions and data are dynamically adjusted. The *off-chip* miss ratio of a *semi-unified* cache built with two direct-mapped caches of size S is equal to the miss ratio of a unified two-way set associative cache of size 2S; yet, the hit time of this *semi-unified* cache is equal to the hit time of a direct-mapped cache; moreover both instructions and data may be accessed in parallel as for the split data/instruction cache organization.

Since on-chip miss penalty is lower than off-chip miss penalty, trace driven simulations show that using a direct-mapped *semi-unified* cache organization leads to higher overall system performance than using usual split instruction/data cache organization.

Caches Semi-unifiés

Résumé

Depuis ces dernières années, le temps de cycle du processeur a diminué beaucoup plus vite que le temps d'accès à la mémoire principale. Cette tendance a augmenté l'importance des caches.

Dans ce rapport, nous présentons une nouvelle organisation pour les caches situés sur la même puce que le processeur: l'organisation de caches *semi-unifiés*. Les deux caches de cette organisation sont séparés physiquement, mais le cache instructions (resp. données) est utilisé comme cache secondaire pour les données (resp. instructions). Ainsi le degré d'associativité pour les caches *semi-unifiés* est artificiellement augmenté et l'espace de stockage se répartit dynamiquement entre instructions et données. Pour deux caches *semi-unifiés* à correspondance directe de taille S, le taux d'échec nécessitant une requête à l'exterieur de la puce est égal au taux d'échec pour un cache unifié de taille 2S associatif par ensemble de deux blocs. De plus le temps d'accès en cas de succès pour les caches *semi-unifiés* est égal à celui concernant les caches à correspondance directe. Les instructions et les données peuvent être accédées en parallèle comme pour l'organisation de caches instructions et données séparées.

Les simulations réalisées montrent que l'organisation de caches semi-unifiés à correspondance directe donne de meillleures performances que les organisations de caches instructions et données séparées ou unifiées.

*This work was partially supported by CNRS (PRC-ANM)

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE (U.R.A 227) UNIVERSITE DE RENNES I INSA DE RENNES
INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE (UNITE DE RECHERCHE DE RENNES)

1 Introduction

Since the gap between main memory access time and processor cycle time is continuously increasing, processor performance dramatically depends on the behavior of caches, and particularly on the behavior of small on-chip caches.

Most of the recently introduced microprocessors have relatively small on-chip caches [2, 8, 11, 17]. High clock frequency and/or parallel instruction issuing are used; yet main memory access time remained approximately constant for ten years (around 250 ns) and relative miss penalties become very high even when second-level caches are used. Table 1 shows the miss penalties converted in Instruction Delays for some of the recently introduced microprocessors; in table 1, the miss penalty when a second-level cache is used is assumed to be 60 ns[1].

microprocessor	Frequency	Instruction per cycles	Miss penalty with 2nd level cache	Miss penalty without 2nd level cache
DEC 21064	200 Mhz	2	24	100
MIPS R4000	100 Mhz	1	6	25
TI SuperSparc	50 Mhz	3	9	37
Power 601	50 Mhz	2	6	25

Table 1 : Miss penalties converted in Instruction Delays

As cache misses induce long pipeline stall, reducing the miss ratio has become an important challenge for microprocessors designers [9]. In this paper, we present a new organization for on-chip caches: the *semi-unified* cache organization. First, in section 2, the respective advantages of using direct-mapped caches and using set-associative caches are recalled. We also recall why using unified caches may be prefered, by some way, to using split instruction/data caches and vice-versa.

The *semi-unified* cache organization is presented in section 3. In most microprocessors, two physically split caches are used for respectively storing data and instructions. The purpose of the *semi-unified* cache organization is to use the data cache (resp. instruction cache) as an on-chip second-level cache for instructions (resp. data). Thus the associativity degree of both on-chip caches is artificially increased, and the on-chip cache spaces respectively devoted to instructions and data are dynamically adjusted. The *off-chip* miss ratio of a semi-unified cache built with two direct-mapped caches of size S is equal to the miss ratio of a unified two-way set associative cache of size 2S. A direct-mapped semi-unified cache organization results in a substantial memory traffic reduction over a direct-mapped split cache organization. Yet, the hit time of this semi-unified cache is equal to the hit time of a direct-mapped cache; moreover both instructions and data may be accessed in parallel as when using a split instruction/data cache organization.

Trace driven simulations presented in section 4 show that the performance of a microprocessor system may dramatically be improved when a semi-unified cache is used in place of a split instruction/data cache.

[1] Access time to the first word of a line in a second level cache is around 60 ns (assuming 12-15ns static memory chips): a throughput of one word per 20 ns may then be obtained; this 60 ns delay corresponds to three delays:

- stalling the execution on the miss detection and presenting the address to the external cache
- access to the external cache
- load of missing line and finally restarting the execution.

2 Cache organizations

2.1 Direct mapped caches or set-associative caches?

In the last few years, the academic community currently admitted that the direct-mapped organization is the best cache organization [5], while the industry seemed divided: direct mapped caches are implemented in MIPS R4000, DEC 21064, HP-PA 7xxx, while set-associative caches have been used in IBM Power [7], Intel i860 [10], Texas Instruments SuperSparc [17] and Motorola 88110. In this paper, we shall adopt the same definitions as Hill in [5]:

Definition 2.1 *The cache hit time is the delay, as seen by the processor, required by the memory system to service a memory reference on a hit.*

Definition 2.2 *The cache access time is the average latency, as seen by the processor, required by the memory system to service a memory reference.*

2.1.1 Direct-mapped caches: better hit time

In [5], Hill argued that, in most microprocessor designs, the cache hit time determines the clock of the system: the hit time on most microprocessors is a single cycle.

A cache read on a direct-mapped cache may be decomposed into two consecutive steps:

1. Read the word and associated tags in cache
2. Check the tags against the address of the data

While a cache read on a n-way set-associative cache consists in three consecutive steps:

1. Read a set of n words and associated tags
2. n parallel tag checks against the address of the data
3. Selection of the correct word in the set

This extra step induces a higher hit time of a set-associative cache than on a direct-mapped cache, but differences between these hit times may not be very significant [2].

Optimistic execution

When using a direct-mapped cache, data (or instructions) flowing out from the cache may be directly used after step 1, because checking the validity of the data word may be executed in parallel with other pipeline activities. The current pipeline cycle is canceled if the data (or instructions) is found to be unvalid.

We shall refer to this technic as *optimistic execution*.

Using optimistic execution on a direct-mapped cache allows to obtain a cache hit time significantly lower than on a classical set-associative cache (15-30% are reported). [3]

2.1.2 Hardware complexity

More hardware logic is needed for implementing a set-associative cache than for implementing a direct-mapped cache: more comparators, more connections, etc. For microprocessors using primary *external caches*, this is a major drawback: due to pin-out limitations, most of this extra glue logic has to be added outside the chip.

But for on-chip caches, the extra hardware cost of a two or four way set-associative cache over a direct-mapped cache is quite limited.

[2] 2% was reported by Hill [5]

[3] Notice that optimistic execution is also possible with a set-associative cache [1] but at a significantly higher implementation cost.

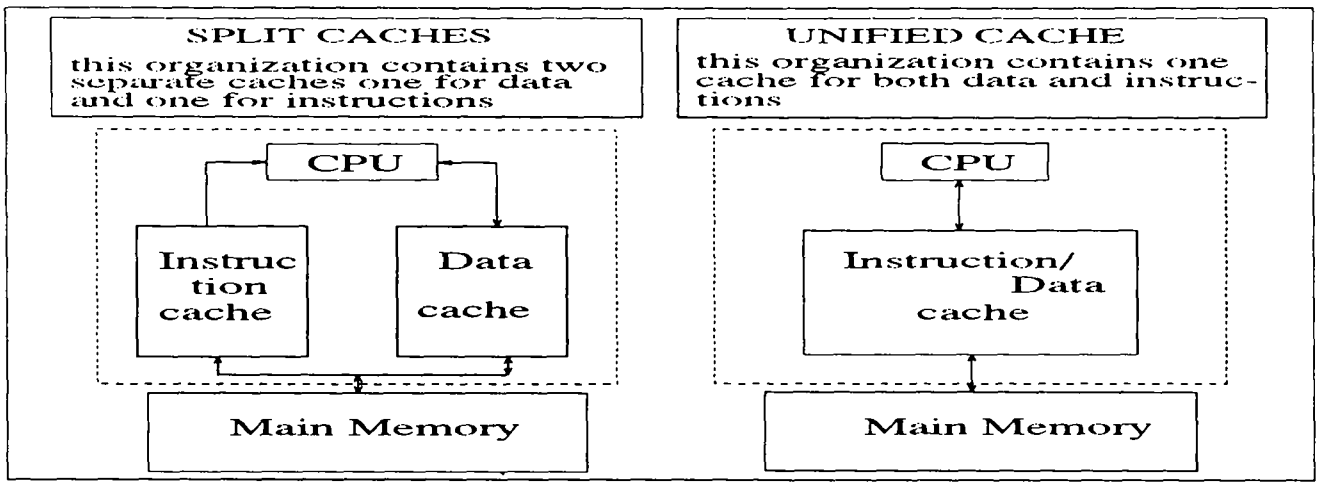


Figure 1 : Split caches and unified cache

2.1.3 Set-associative caches: higher hit ratio

The main argument for using a set-associative cache rather than a direct-mapped cache is its better hit ratio: in [4, 6], the authors reported that using a two-way set-associative cache in place of a direct-mapped cache removes about 30% of the misses. Surprisingly this ratio is approximately independent of the cache size.

Since on-chip caches are relatively small and therefore exhibit relatively large miss ratio, the performance benefit in using set-associative caches may be relatively high.

2.1.4 Synthesis

Performance of a system depends on both cycle time and the average number of cycles needed per instruction (CPI) [3]. The cache hit time generally determines the clock cycle. A direct-mapped cache may allow to use a high clock frequency, while using some degree of associativity on caches will help to achieve low CPI.

2.2 Unified cache or split caches

On some microprocessors, a unified cache for both instructions and data is used; this cache organization is used on many microprocessor systems which have external caches and for most second-level caches. Yet, generally, when on-chip caches are provided, a split cache organization is used (figure 1); data and instructions lie on two distinct caches.

2.2.1 Split caches: parallel access

When a split cache organization is used, both the instruction cache and the data cache may be accessed at the same time. Thus a load/store may be executed and an instruction or a group of instructions may be read in parallel.

On a unified cache, an instruction or group of instructions cannot be read at the same time when a load/store is executed : for the same total miss number, the CPI is higher when using a unified cache than when using a split cache organization.

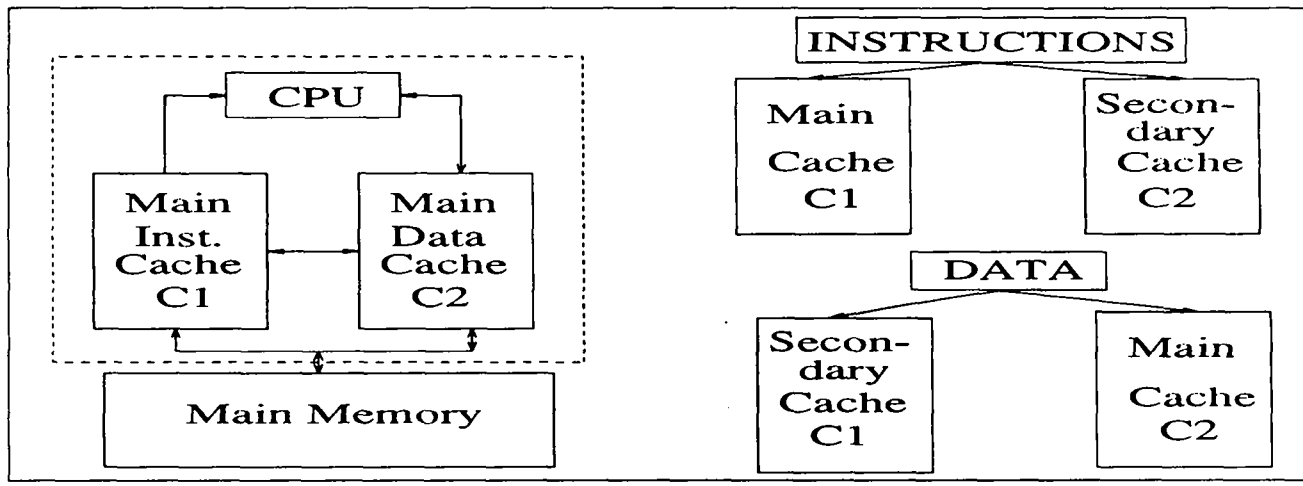


Figure 2 : Semi-unified caches

2.2.2 Unified cache: dynamic instruction/data spaces balancing

On a split cache organization, the spaces respectively devoted to instructions and data are constant; for on-chip caches, it is determined when the processor is designed.

Yet, on a unified cache, the cache spaces respectively devoted to instructions and data are dynamically adjusted on demand at execution time; therefore the cache space occupied by the instructions may be very small in some part of an application (e.g. a loop nest) or be predominant in another application or even in a different part of the same application.

3 Semi-unified caches

We introduce the semi-unified cache organization illustrated in figure 2.

Definition 3.1 *A semi-unified cache organization consists of two distinct instruction/data caches where the instruction cache (resp. data cache) is used as the secondary cache for data (resp. instructions).*

This cache organization has been defined for on-chip caches. The cache C1 (resp. C2) is the main cache for instructions (resp. data) and the secondary cache for data (resp. instructions). For sake of simplicity, we shall now consider that the semi-unified cache lies on the microprocessor chip, while higher level caches are external.

3.1 Sequencing a request on a semi-unified cache

Let us denote the instruction cache by C1 and the data cache by C2. Cache C1 is the secondary data cache and C2 the secondary instruction cache (figure 2).

Let us now describe the sequencing of an instruction request to address A (see figure 3):

1. the request is presented to cache C1;
2. on a miss in cache C1, the request is presented to cache C2. In the following paragraphs, this first-level cache miss is called an *on-chip miss*:

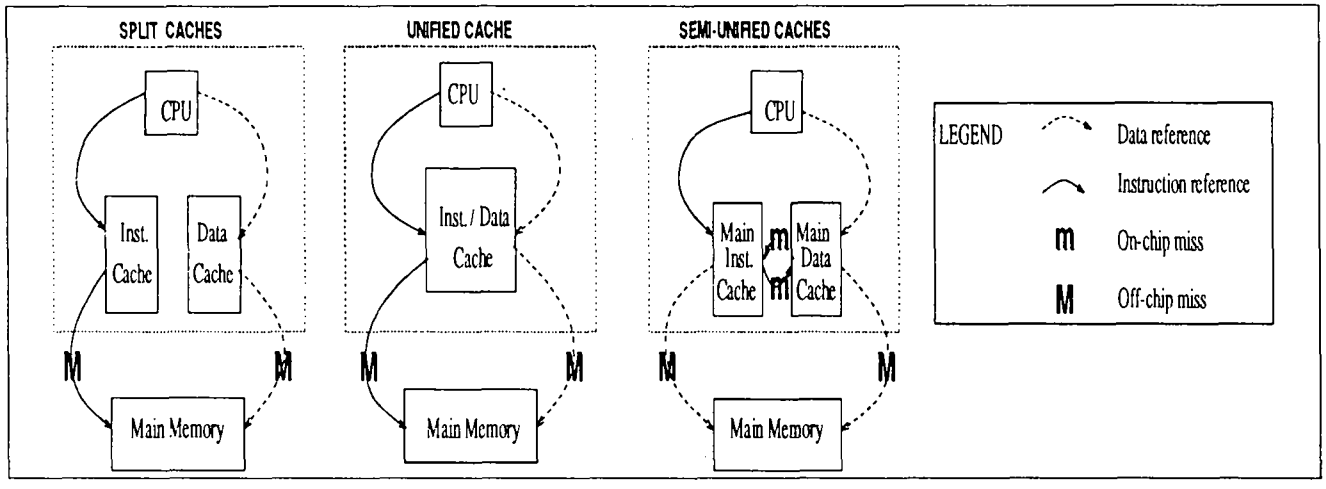


Figure 3 : On-chip and Off-chip misses

- On a hit in cache C2, the requested line is brought back to cache C1: a line must be rejected from cache C1, the replacement of this line is discussed in the next section. In general, the lines are swapped between cache C1 and cache C2.
- On a miss in cache C2, the request is presented to an external cache or the main memory. We shall call this second-level cache miss, an *off-chip miss*.

3.2 Replacement strategy for direct-mapped semi-unified caches

The semi-unified cache organization may be used with different basic cache structures for cache C1 and cache C2.

We now focus our attention on a semi-unified cache built with two direct-mapped caches of equal sizes. We shall refer to this organization as *a direct-mapped semi-unified cache.*

Off-chip misses E.g. let us consider an off-chip instruction miss:

The requested instruction line L must be stored in physical cache line $L(A)$ (where A is the instruction request address) of C1, then the content $L1$ of line L(A) must be rejected from cache C1.

But this rejected line $L1$ may be stored in cache C2 in physical cache line $L(A)$.

Finally the rejected line from the whole on-chip direct-mapped semi-unified cache must be chosen among a set of two lines:

1. $L1$ the content of the targeted physical line $L(A)$ in cache C1.
2. $L2$ the content of physical line $L(A)$ in cache C2: if $L2$ is rejected, then $L1$ is moved from cache C1 to cache C2.

The strategy for chosing the rejected line may be random or LRU as on a classical two-way set-associative caches.

On-chip misses Let us consider an on-chip instruction miss which hits on the secondary on-chip cache. The requested line L lies in line $L(A)$ of cache C2; this line must be brought back to line $L(A)$ in cache C1. The content L1 of line $L(A)$ in cache C1 must be removed from C1, but since line $L(A)$ in cache C2 is now empty, it may be stored in this location (L1 and L2 are swapped between cache C1 and cache C2): no external traffic is induced, global content of the on-chip cache has not changed.

3.3 Assets and drawbacks of semi-unified caches

The direct-mapped semi-unified cache organization built with two direct-mapped caches of equal sizes S have most of the advantages of both two-way set-associative cache organization and direct-mapped cache organization; it has also the assets of both unified cache organization and split cache organization:

- The cache hit time on a direct-mapped semi-unified cache is the same as when using split direct-mapped instruction/data caches of equal sizes S.

  But the off-chip miss ratio of the direct-mapped semi-unified cache is the same as that of a unified two-way set-associative cache of size 2S.

- Parallel access to instructions and data is provided; yet the on-chip cache spaces respectively devoted to instructions and data are dynamically adjusted as on a unified two-way set-associative cache.

Implementing such a semi-unified cache organization induces some additional hardware costs over an usual split cache organization: essentially, a data path is needed for swapping the lines between cache C1 and cache C2. Notice that an on-chip miss induces a miss penalty even when the request hits on the secondary on-chip cache.

In the next section, we use trace driven simulations to compare performance of microprocessors using a direct-mapped semi-unified cache organization or using classical cache organizations.

4 Performance evaluation

Trace driven simulations have been used to evaluate the performance benefit that may be expected from using a semi-unified cache organization in place of a classical cache organization.

In this section, we consider a split cache organization built with two direct-mapped caches of equal sizes S, a semi-unified cache organization built with two direct-mapped caches of equal sizes S and a unified cache organization built with one two-way set-associative cache of size 2S.

4.1 Simulation model

For sake of simplicity in our simulation, we suppose that on a split or semi-unified cache organizations, a new instruction is issued on each cycle and that on a unified cache organization, a new request is issued on each cycle. And we assume a write back strategy with a write buffer of two cache lines and a LRU replacement for direct-mapped semi-unified and unified two-way set-associative caches.

A cycle-by-cycle simulation was performed.

- **Servicing off-chip misses**: on an off-chip miss, *latency* cycles are needed for getting back the first 64 bits word from the memory or the off-chip extra-level[4] cache; then 4 cycles (resp. 2) more are needed for obtaining each extra 64 bits word in the line from the memory (resp. the off-chip extra-level cache). We also assume that the first word coming back from memory or off-chip extra-level cache is the missing word and that the execution may then be restarted as soon as this word is available.

[4] In case of a semi-unified cache organization, this off-chip extra level corresponds to the third cache level. It corresponds to the second cache level for split or unified cache organizations.

- **Write back**: when no off-chip extra-level cache is used, we assume that the write back of a line busies the memory for the same time as the service of a miss; this approximatively corresponds to the behavior of dynamic RAM.
  When an off-chip extra-level cache is used, we assume that the writes may be pipelined with other accesses: the write back of a line busies the off-chip cache for $2 * LineSize/64$ cycles.
- When off-chip extra-level cache is used, second-level misses are not considered.

Miss penalty

The miss penalty for an on-chip miss which hits on the second-level on-chip may be quite low: from 2 to 4 cycles depending on the pipeline for accessing the cache. As said in the introduction, the off-chip miss penalty is higher, particularly when no external cache is implemented. For example, the miss penalty on a MIPS R4000 system [11] with a second level cache corresponds to approximately 6 cycles, but is in the range of 25-30 cycles when no second-level cache is used. All simulation results reported in this paper were obtained assuming an on-chip miss penalty of 3 cycles, while several off-chip miss penalties were simulated.

Performance metric

As pointed out in the previous section, off-chip misses on a direct-mapped semi-unified cache are exactly the same as on a unified two-way set-associative cache with corresponding sizes.

In order to compare the performance of systems using semi-unified caches with classical systems, we need a more accurate metric than off-chip miss ratio. As a metric, we use the average number of clock cycles per instruction (CPI) [3].

Traces

We have simulated a single process and traces include user mode only.

The first set was composed with the three traces from the Hennessy-Patterson software obtained from the DLX simulator [3] (gcc, TeX and Spice).

Seven other traces were generated using the SparcSim simulator facility [16]; this set was composed with:

- RESEAU: the simulator of a particular interconnection network
- POISSON : a Poisson solver
- STRASSEN: a matrix-matrix multiply using the Strassen algorithm
- LINPACK: part of the LINPACK benchmark
- CACHE : the cache simulator itself
- CPTC : a Pascal-to-C translator
- MM : a sparse matrix vector multiply.

4.2 Miss ratio

Figure 4 illustrates on-chip miss ratio per instruction and off-chip miss ratio per instruction for split and semi-unified caches for each benchmark. The number of off-chip misses per instruction is lower for semi-unified caches than for split caches. For example with two on-chip 4 Kbytes caches, the off-chip miss ratio for semi-unified caches is between 20 % (CC1) to 80 % (POISSON) below the off-chip miss ratio of split caches.

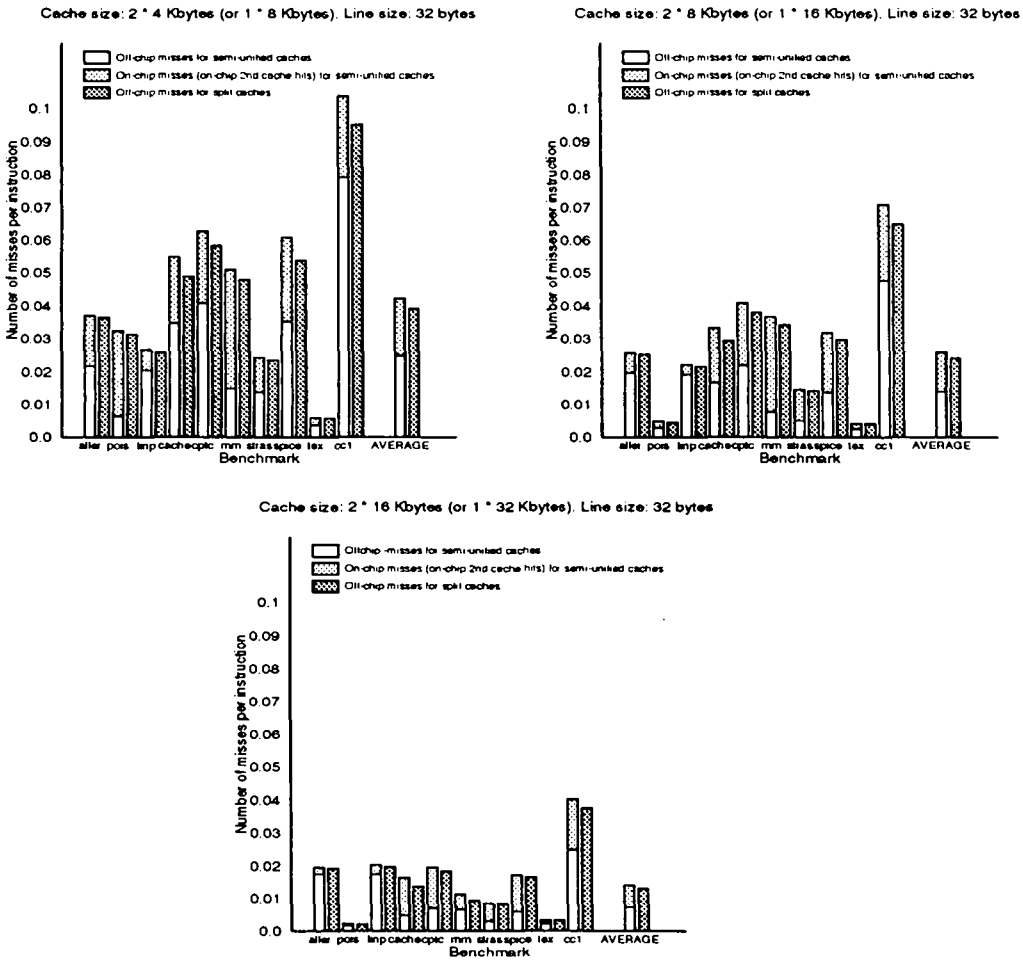


Figure 4 : On-chip and off-chip misses (line size = 32 bytes)

The total first level miss ratio on a semi-unified cache organization is generally slightly higher than the miss ratio on a split direct-mapped cache organization. But as the on-chip miss penalty is lower than the off-chip miss penalty, this may result in lower CPI as it will be shown below.

Notice also that only off-chip misses result in effective off-chip accesses. Using a semi-unified cache organization in place of a classical split cache organization significantly reduces the memory traffic; this may be very interesting in order to build low-end single-bus shared memory multiprocessor system.

4.3 Simulation results

By the end of 1992, on-chip cache sizes vary between 12 and 36 Kbytes [2, 8, 11, 17]. 8, 16 and 32 Kbytes on-chip caches were simulated with 16, 32 and 64 bytes lines [15].

Figures 5, 6, 7, 8 show CPI performance as a function of:

- cache organization: split, unified and semi-unified caches
- benchmark
- on-chip cache size: 8, 16 and 32 Kbytes
- line size: 16, 32 and 64 bytes

- off-chip penalty: 4, 5, 7 CPU cycles with external second-level caches and 14, 24, 34, 44, 54 CPU cycles with memory only

CPI semi-unified caches / CPI split caches	aller	poisson	linpack	cache	cptc	
off-chip penalty = 4 CPU cycles	0.99	0.99	1.00	0.96	0.95	
Off-chip penalty = 24 CPU cycles	0.89	0.97	0.99	0.78	0.76	
CPI semi-unified caches / CPI split caches	mm	strassen	spice	tex	cc1	**average**
Off-chip penalty = 4 CPU cycles	0.93	0.98	0.95	0.99	0.96	0.97
Off-chip penalty = 24 CPU cycles	0.65	0.84	0.75	0.96	0.82	0.84

Table 2 : CPI for a semi-unified cache organization over a split cache organization

Figures 5 and 6 show the CPI for different off-chip penalties: 4, 5, 7 CPU cycles with off-chip secondary cache and 14, 24, 34, 44, 54 CPU cycles without off-chip secondary cache (32 bytes line and two 8Kbytes caches or one 16 Kbytes cache). CPIs for direct-mapped semi-unified cache organization are generally better than for direct-mapped split and unified two-way set-associative cache organizations. This performance advantage is quite insignificant when a very fast external second-level cache is used, but becomes very important when the off-chip latency increases. The table 2 illustrates the CPI improvement for a semi-unified cache organization over a split cache organization considering two on-chip 8 Kbytes caches with 32 bytes line size. For instance, considering a 24 CPU cycles off-chip penalty, using a semi-unified cache organization in place of a split cache organization leads to a 34 % lower CPI for the MM benchmark and to a 15 % lower CPI in average on our set of benchmarks.

Thus the semi-unified cache organization seems to be a very good tradeoff.

As the gap between CPU cycle and off-chip miss penalty will continue to increase, these figures clearly indicate that implementing an on-chip direct-mapped semi-unified cache organization in a microprocessor is worthwhile for microprocessors dedicated to highend systems as well as for microprocessors dedicated to medium end and low end systems.

5 Conclusion

As the integration density increases, primary caches may be integrated on the microprocessor chip. Yet, on-chip cache sizes remain limited, therefore the available space must be cautiously used.

Microprocessor clock cycle, and therefore theoretical peak performance, is mostly determined by the cache hit time. Better cache hit time is possible if direct-mapped caches are used. Yet, a higher degree of associativity would reduce the miss ratio and finally the overall pipeline stall delays for servicing the misses.

In this paper, we have proposed the semi-unified cache organization. A semi-unified cache consists of a split instruction/data cache organization where the data cache (resp. instruction cache) is used as a second-level cache for instructions (resp. data).

As it may hit on the on-chip data cache (resp. instruction cache), a first-level instruction (resp. data) miss does not necessarily lead to an *off-chip* transaction. The *off-chip* miss ratio on a semi-unified cache built with two direct-mapped caches of size S, is equal to the miss ratio on a unified two-way set associative cache of size 2S. Yet, the hit time of this semi-unified cache is equal to the hit time of a direct-mapped cache of size S. Moreover, both instructions and data may be accessed in parallel as for a split instruction/data cache organization.

Using a semi-unified cache organization in place of a usual split instruction/data cache organization slightly increases the first-level cache miss ratio and significantly decreases the off-chip miss ratio. Yet, as the

penalty of an on-chip miss is lower than the penalty of an off-chip miss, the overall system performance is increased as shown by simulation results in section 4.

Since the gap between off-chip miss penalty and on-chip internal delays will continue to grow, using a direct-mapped semi-unified cache organization is very attractive because it allows fast clock cycle and optimizes the off-chip miss ratio.

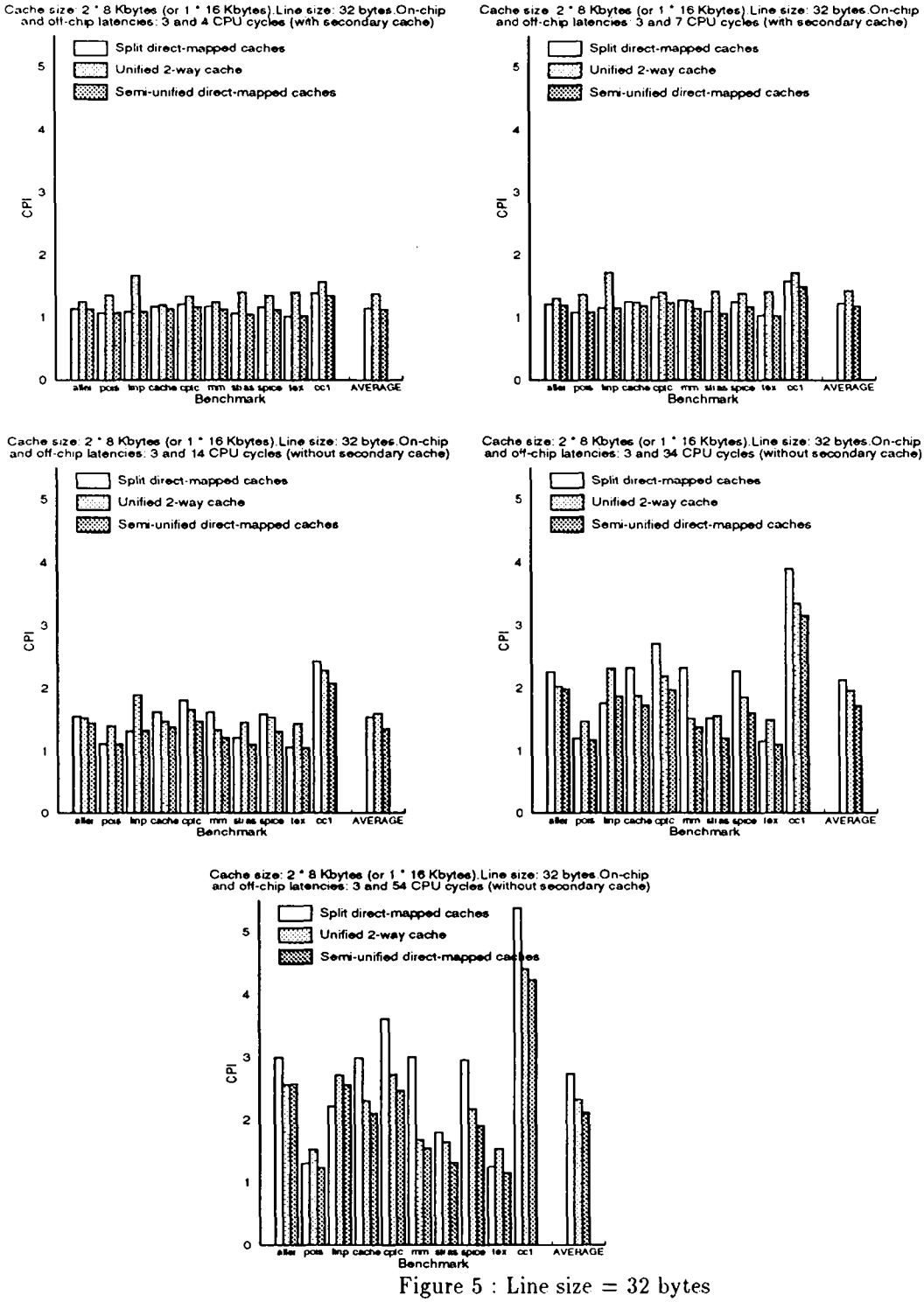


Figure 5 : Line size = 32 bytes

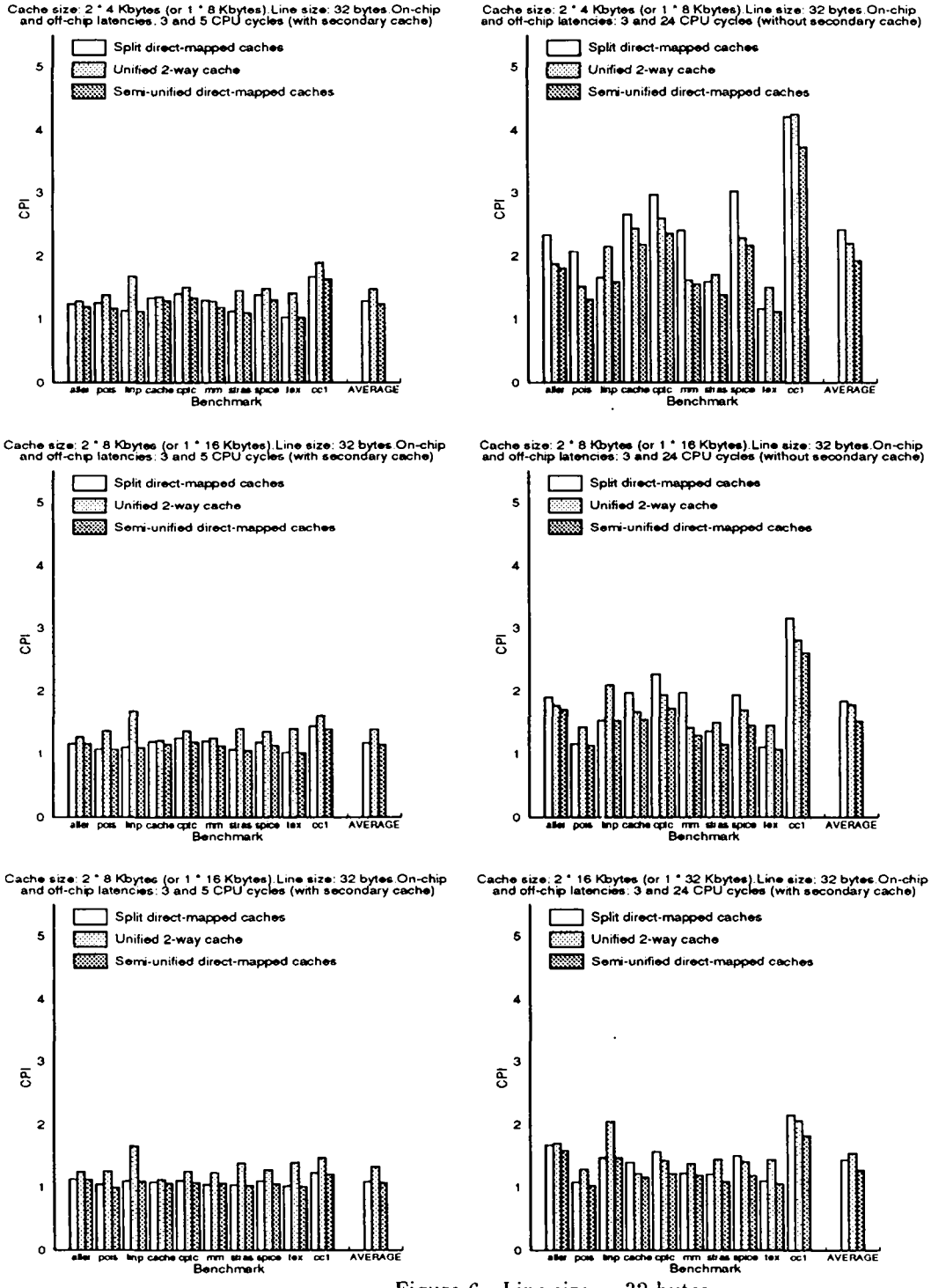


Figure 6 : Line size = 32 bytes

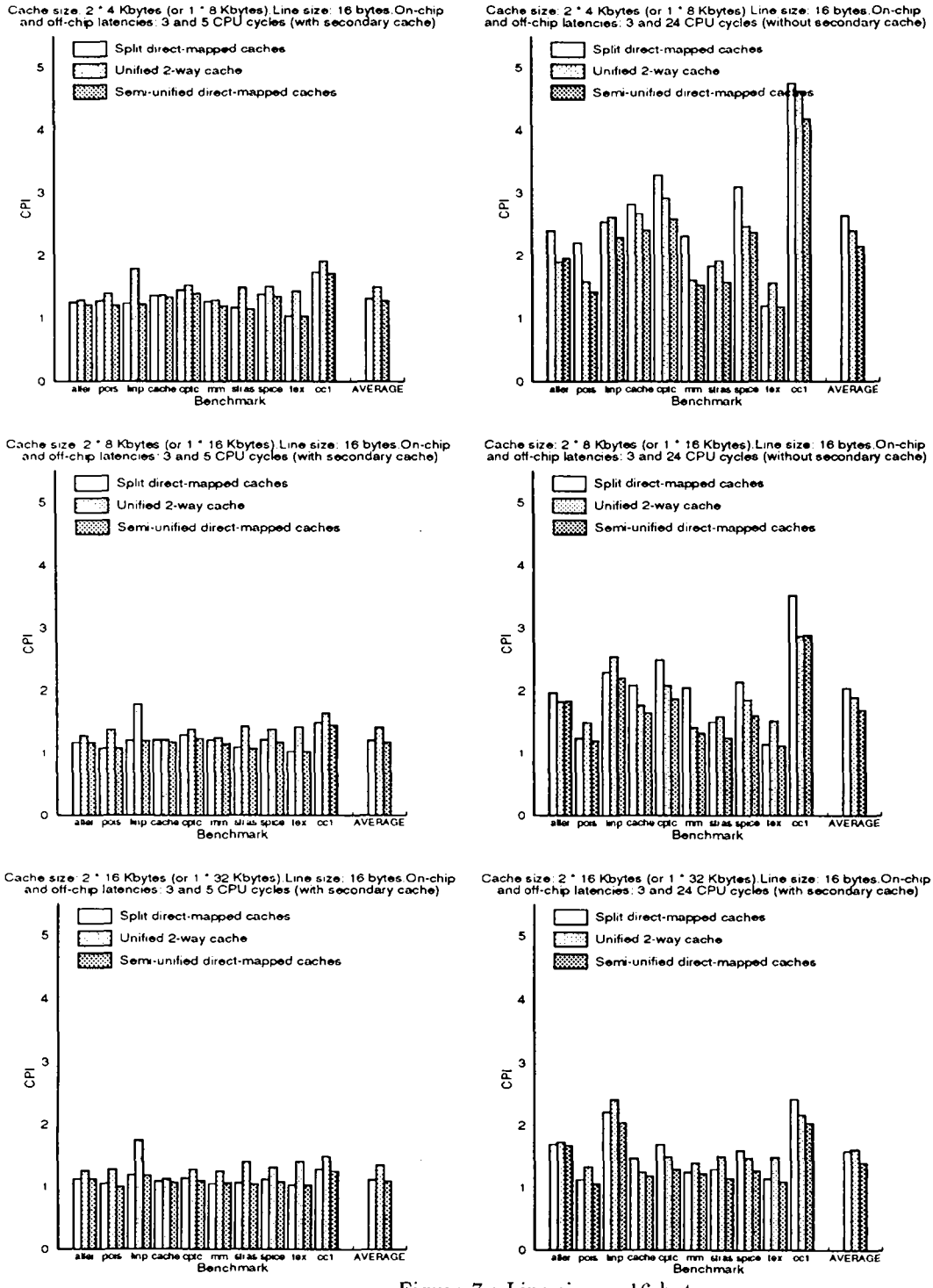


Figure 7 : Line size = 16 bytes

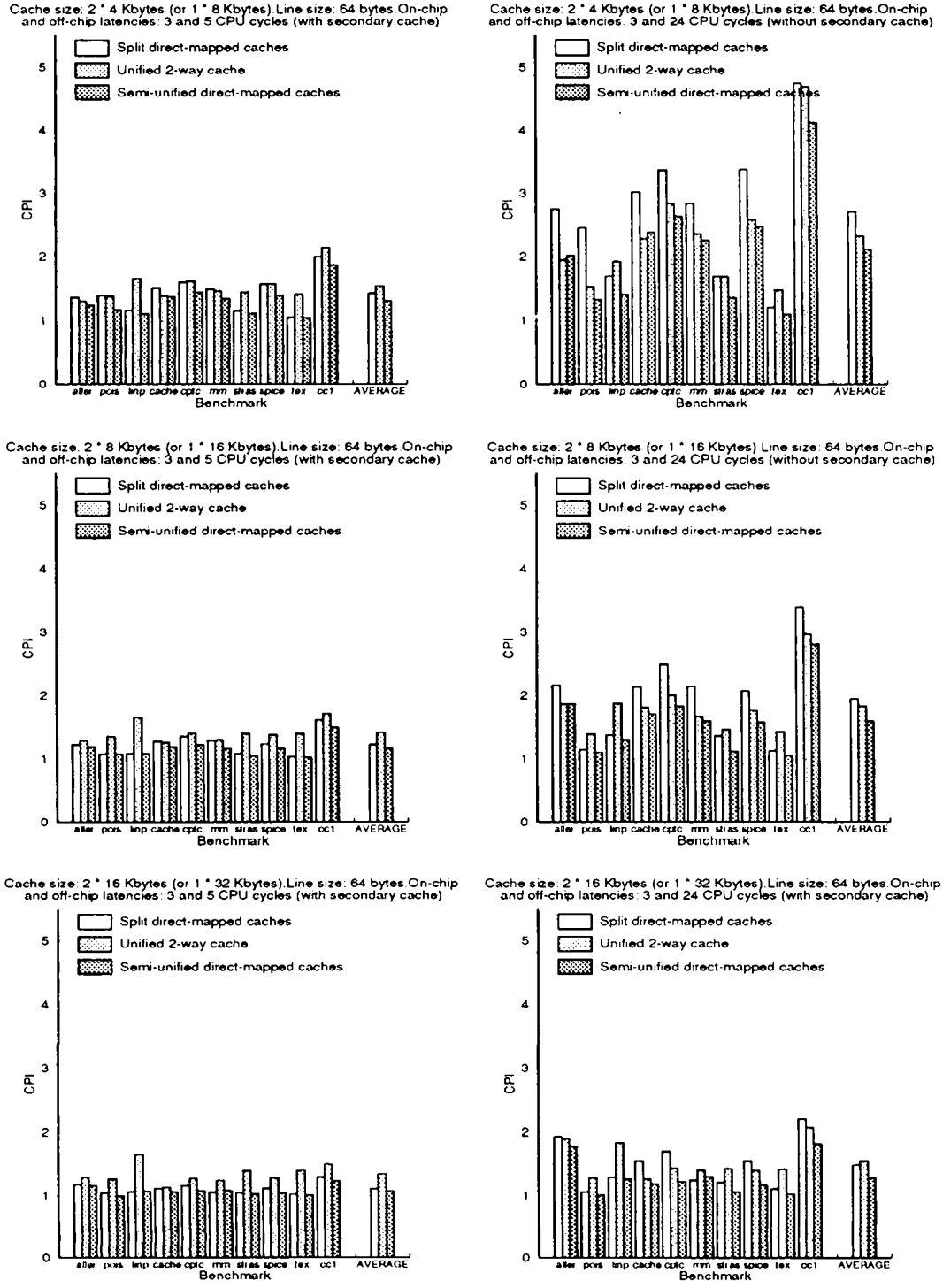


Figure 8 : Line size = 64 bytes

Bibliographie

[1] J.H. Chang, H. Chao, and K. So, "Cache Design of A Sub-Micro CMOS System/370", pp208-213, Proceedings of the 14th International Symposium on Computer Architecture (IEEE-ACM), May 1987

[2] *DECChip 21064-AA RISC Microprocessor, Preliminary Data Sheet* Digital Equipment Corporation, 1992

[3] J.L. Hennessy, D.A. Patterson, "Computer Architecture a Quantitative Approach", Morgan Kaufmann Publishers, Inc. 1990

[4] M.D. Hill, "Aspects of Cache Memory and Instruction Buffer Performance", Ph.D Thesis, University of Berkeley, 1987

[5] M.D. Hill, " A Case for Direct-mapped Caches", IEEE Computer, Dec. 1988

[6] M.D.Hill, A.J. Smith, "Evaluating Associativity in CPU Caches", IEEE Transactions on Computers, Dec. 1989

[7] *IBM RISC system/6000 technology* IBM Corporation

[8] *i860: 64-bit microprocessor hardware reference manual* Intel, 1990

[9] N.P. Jouppi, "Improving Direct-Mapped Cache Performance by the Addition of a Small Fully-Associative Cache and Prefetch Buffers", Proceedings of the 17^{th} International Symposium on Computer Architecture, June 1990

[10] L. Kohn, S.W. Fu, "A 1,000,000 Transistor Microprocessor", IEEE Micro, feb. 1989

[11] G. Kane, J. Heinrich, "MIPS RISC Architecture", Prentice-Hall, 1992

[12] K. Olukotun, T. Mudge, and R. Brown, "Performance Optimization of Pipelined Primary Caches", Proceedings of 19^{th} International Symposium on Computer Architecture , May 1992

[13] S.A. Przysbylski, "Performance-Directed Memory Hierarchy Design", PhD Thesis, Stanford University, 1988

[14] A.J. Smith, "A Comparative Study of Set Associative Memory Mapping Algorithms and Their Use for Cache and Main Memory", IEEE Transactions on Sofware Engineering, March 1978

[15] A.J. Smith, "Line (block) Size Choice for CPU Cache Memories", IEEE Transactions on Computers, Sept. 1987

[16] *SparcSim Manual* SUN Inc, Dec. 1989

[17] *TMS390Z55 Cache Controller, Data Sheet* Texas Instrument, 1992

LISTE DES DERNIERES PUBLICATIONS INTERNES IRISA PARUES EN 1992

PI 686 FROM EQUATIONS TO HARDWARE. TOWARDS THE SYSTEMATIC MAPPING OF ALGORITHMS ONTO PARALLEL ARCHITECTURES
François CHAROT, Patrice FRISON, Eric GAUTRIN, Dominique LAVENIER, Patrice QUINTON, Charles WAGNER
Octobre 1992, 18 pages.

PI 687 THE COMPILATION OF PROLOG and its Execution with MALI
Pascal BRISSET, Olivier RIDOUX
Novembre 1992, 90 pages.

PI 688 GENERALISATION DE L'ANALYSE FACTORIELLE MULTIPLE A L'ETUDE DES TABLEAUX DE FREQUENCE ET COMPARAISON AVEC L'ANALYSE CANONIQUE DES CORRESPONDANCES
Lila ABDESSEMED, Brigitte ESCOFIER
Novembre 1992, 16 pages.

PI 689 OVERVIEW OF THE KOAN PROGRAMMING ENVIRONMENT FOR THE iPSC/2 AND PERFORMANCE EVALUATION OF THE BECAUSE TEST PROGRAM 2.5.1
François BODIN, Thierry PRIOL
Décembre 1992, 16 pages.

PI 690 CONCURRENT PROGRAMMING NOTATIONS IN THE OBJECT-ORIENTED LANGUAGE ARCHE
Marc BENVENISTE, Valérie ISSARNY
Décembre 1992, 34 pages.

PI 691 COMMUNICATION EFFICIENT DISTRIBUTED SHARED MEMORIES
Masaaki MIZUNO, Michel RAYNAL, Gurdip SINGH, Mitchell L. NEILSEN
Décembre 1992, 24 pages.

PI 692 ETUDE COMPAREE DES ARCHITECTURES DES MICROPROCESSEURS MIPS R4000, DEC 21064 ET T.I. SUPERSPARC
André SEZNEC, Anne-Marie KERMARREC, Thierry VAULEON
Décembre 1992, 106 pages.

PI 693 ETUDE DE L'UTILISATION D'ARCHITECTURES PARALLELES A MEMOIRE DISTRIBUEE POUR LA REALISATION DE COMMUTATEURS DE RESEAUX HAUT DEBIT
Jean-Marc JEZEQUEL, Claude JARD, Thierry ESTIMBRE
Décembre 1992, 40 pages.

Unité de Recherche INRIA Rennes
IRISA, Campus Universitaire de Beaulieu 35042 RENNES Cedex (France)

Unité de Recherche INRIA Lorraine Technopôle de Nancy-Brabois - Campus Scientifique
615, rue du Jardin Botanique - B.P. 101 - 54602 VILLERS LES NANCY Cedex (France)
Unité de Recherche INRIA Rhône-Alpes 46, avenue Félix Viallet - 38031 GRENOBLE Cedex (France)
Unité de Recherche INRIA Rocquencourt Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 LE CHESNAY Cedex (France)
Unité de Recherche INRIA Sophia Antipolis 2004, route des Lucioles - B.P. 93 - 06902 SOPHIA ANTIPOLIS Cedex (France)

EDITEUR
INRIA - Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 LE CHESNAY Cedex (France)

ISSN 0249 - 6399