A Selective Filter-Bank TLB System

Jung-Hoon Lee
CS, Yonsei University
134, Shinchon-dong,
Seoul, 120-749, Korea
+82-2-2123-2718
ljh@yonsei.ac.kr

Gi-Ho Park
System LSI Business,
Samsung Electronics
Co., Suwon, Korea
+82-31-279-7744
gmean@samsung.co.kr

Sung-Bae Park
System LSI Business,
Samsung Electronics
Co., Suwon, Korea
+82-31-279-7744
scpu1977@samsung.co.kr

Shin-Dug Kim
CS, Yonsei University
134, Shinchon-dong,
Seoul, 120-749, Korea
+82-2-2123-2718
sdkim@cs.yonsei.ac.kr

ABSTRACT
We present a selective filter-bank translation lookaside buffer (TLB) system with low power consumption for embedded processors. The proposed TLB is constructed as multiple banks with a small two-bank buffer, called as a filter-bank buffer, located above its associated bank. Either a filter-bank buffer or a main bank TLB can be selectively accessed based on two bits in the filter-bank buffer. Energy savings are achieved by reducing the number of entries accessed at a time, by using filtering and bank mechanism. The overhead of the proposed TLB turns out to be negligible compared with other hierarchical structures. Simulation results show that the Energy*Delay product can be reduced by about 88% compared with a fully associative TLB, 75% with respect to a filter-TLB, and 51% relative to a banked-filter TLB.

Categories and Subject Descriptors
B.3 [**Hardware**]: Memory Structures – *Design Styles*; *associative memories*, C.3 [**Computer Systems Organization**]: Special-Purpose and Application-Based Systems– *Real-time and embedded systems*, I.6 [**Computing Methodologies**]: Simulation and Modeling - *Simulation Output Analysis*

General Terms
Design, Experimentation, and Performance

Keywords
Translation lookaside buffer, low power consumption, filtering mechanism, and performance evaluation.

1. INTRODUCTION
Recent embedded processors support a virtual memory system through a hardware memory management unit (MMU) that translates virtual addresses to physical addresses. Moreover, those processors are widely used for multimedia and communication applications, requiring high-speed computing capability, high memory bandwidth, and effective memory hierarchy support. Also power consumption is a major factor in designing high performance embedded processors. In general, reducing power consumption architecturally in the memory system causes a more significant impact than many other techniques at the gate/circuit level.

The translation look-aside buffer (TLB) is an on-chip memory structure that caches only page table entries for recently used virtual to physical address translations [1]. Most present TLBs are typically implemented with static RAM cells for data, and content addressable memories (CAMs) for tags.

Conventional methods for reducing TLB power consumption are to make it hold fewer entries, apply a filtering or a block buffering mechanism, and utilize a bank structure [2,3,4]. When the number of TLB entries decreases, it brings about performance degradation. The filter-TLB mechanism, where a very small TLB is located above the conventional L1 TLB, causes a performance degradation because it increases the number of two-cycle accesses. Block buffering can be viewed as an approach similar to the filter mechanism. However, accessing the block buffer should be completed during one cycle for modern microprocessors with high clock frequency. The bank structure consumes less power than a fully associative TLB because only half of the CAM entries are looked up on each access to the TLB. But a drawback is performance degradation due to the tendency to encounter more capacity misses in a specific bank. But in case of a specific application, e.g., tomcatv, high performance may be achieved as particular access pattern.

Our TLB structure supports low power consumption for embedded processors. We use a selective filtering and bank mechanism that offers low power consumption via a simple hardware control mechanism. That is, either a filter-bank buffer or a main bank TLB can be selectively accessed using simple logic. The average accuracy of these operations is about 97% with respect to the total number of addresses generated by the CPU. Our scheme divides both the small filter structure into two banks and the main TLB structure into four banks to reduce power consumption. We show that energy is saved by accessing either the main bank or the filter-bank buffer selectively and by decreasing the number of fully associative TLB entries to be accessed at a time.

Simulation results show that the power consumption of the proposed TLB is about 90% less than the fully-associative TLB, 70% less than a filter-TLB, and 34% less than a banked-filter TLB. Also, the Energy*Delay metric is reduced by about 88%, 75%, and 51% compared with a fully associative TLB, a filter-TLB, and a banked-filter TLB respectively.

2. RELATED WORK

In both modern high performance microprocessors and embedded processors, the TLB within the processor chip is split into separate instruction and data TLBs (e.g., Strong-Arm, MIPS, Alpha, PowerPC, and Ultra-SPARC, etc.,) [5,6]. The miss ratio in TLBs tends to be very small because each entry refers to a page of memory. However, a TLB miss is accompanied by a long handling latency, i.e., on the order of tens to hundreds of cycles. Therefore, a fully associative TLB is typically used to obtain lower miss rates. But full associativity is very costly in terms of power consumption. To reduce power consumption, the total number of TLB entry tags accessed at the same time should be fewer than 64 or 128 [7]. However, higher performance can be achieved if more TLB entries are provided. One method for dealing with these conflicting goals is to divide the entire TLB space into several sub-TLBs so that the number of tags accessed together can be reduced to fewer than 32 or 64 [4]. This bank-TLB [4] consumes less power than a fully associative TLB because only portions of the CAM entries are activated on each access. The block buffering technique [3,11,12] is an approach to reducing power consumption without having any effect on performance. However, accessing the block buffer must be completed during one cycle for modern microprocessors with high clock frequency.

Another well-known TLB system, the filter (micro)-TLB is a hierarchical structure, where a very small TLB is located above the conventional L1 TLB [2]. In terms of power consumption, a filter-TLB turns out to be effective when combined with the instruction TLB due to its low miss ratio. But for the data TLB, the performance degradation of the filter-TLB, compared with a fully associative TLB, becomes significant.

Other TLB studies for low power consumption address memory cell redesign, such as modifying the CAM cell [7], using a low power RAM [8], and voltage reduction [9]. The work by Juan [7] proposes modifying the CAM cell by adding another transistor in the discharge path. The work by Itoh [8] proposes low-power circuit design techniques, such as pulsed word-line and sense circuitry. With these schemes, the access circuitry is enabled only long enough to ensure reliable reading and writing of memory cells. Work by Liu [9] proposes a method of supporting a lower supply voltage in designing memory systems. Supply voltage is one of the most important parameters controlling CMOS power consumption.

3. SELECTIVE FILTER-BANK TLB

Our goal is to design a new TLB system to support low power consumption for embedded processors. A mechanism based on a filtering and banked structure is presented, which achieves both fast access time and low power consumption.

The tag memory space in the fully associative TLB is implemented using a group of content addressable memories (CAMs), which have additional transistors to perform parallel comparisons for all the tag entries in memory. If the tag in any one entry is matched with the input tag placed on the bit lines, its corresponding match line remains high and all other match lines are pulled low, and the selected match line activates the associated word line of the SRAM. Thus its corresponding PPN (physical page number) information is read out from the data array. The structure of the fully associative TLB precludes the need for any external comparison logic or multiplexors, but its access time is longer than that of other organizations because the tag comparison cannot be simultaneously performed with reading the data from SRAM. In addition, for each access to the CAM, all match lines must be precharged high, and all match lines that do not produce a match signal must then be discharged. These precharge and discharge operations are responsible for a significant fraction of the TLB's energy dissipation.

With fully associative TLB structures, power consumption tends to increase abruptly as the number of TLB entries increases beyond 64 [7]. To keep power consumption low, the total number of TLB tags that are compared at once should thus be smaller than 64. However, higher performance can be achieved if more TLB entries are provided. We would like to allow as many entries as possible, while keeping the number of tags accessed together to fewer than 64. This is done by dividing the entire TLB space into separately accessed sub-TLBs. In our preliminary exploration of the design space, we simulated several configurations and determined that the most effective number of sub-TLBs for our benchmarks is four. Figure 1 illustrates the organization of the selective filter-bank TLB with its dynamic searching operation. As shown in Figure 1, the selective filter-bank TLB is constructed as four banks, and each bank consists of a main TLB and its associated two small filter-bank buffers. Two small banks are located above its associated main TLB in the hierarchy. To reduce power consumption, the two low-order address bits of the tag for any given VPN (virtual page number) are used to select a main bank.

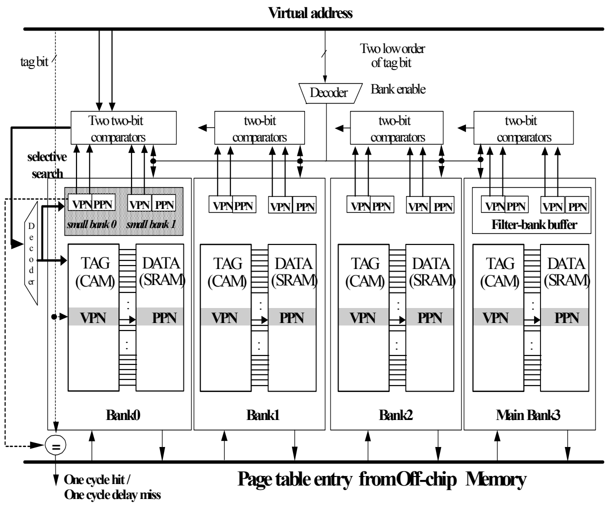

Figure 1. Selective filter-bank TLB

The filter-bank buffer associated with each bank stores a tag value and a data value for the most recently accessed VA (virtual address) belonging to its corresponding main bank module. Only one of two small banks is activated by using a third bit of the tag for any given VPN. A two-bit comparator compares two bits of the VPN tag in the filter-bank buffer with two bits of a newly generated VPN. This comparator consists of two XORs and one AND gate. In our simulations, the particular two bits that are used for the comparison are the fourth and fifth low-order bits of the

VPN. If more bits are used for the comparison, then higher accuracy can be achieved. But overhead, such as the comparison time and hardware cost, then increases. Conclusively, the detection of dynamic searching operation occurs from comparison of the number of five bits. Therefore, a newly generated address in 128-byte page boundary is detected correctly for accessing either a filter-bank buffer or a main bank selectively.

The selective filter-bank TLB is designed so that eight two-bit comparators can operate in parallel for fast access. The two-bit comparison time takes place during the bank selection period and thus can be almost completely hidden. When a two-bit comparator identifies a match, the filter-bank buffer in its corresponding bank module is enabled to check first. If the VPN in the corresponding filter-bank buffer and the newly generated VPN are the same, the PPN in the corresponding data buffer is fetched in one cycle. But if the VPN comparison is a mismatch, then a one cycle delay is incurred to search the main bank TLB. Of course, if the two-bit comparator identified a mismatch, the corresponding main bank TLB is just enabled directly and checked in one cycle, and the comparison of the new VPN with the filter-bank buffer is skipped. When a virtual address is generated, our scheme dynamically chooses whether to compare it with the VPN in the filter-bank buffer or its corresponding main bank TLB. In general, the LRU (least recently used) replacement policy produces the best miss rates since it minimizes conflicts. Unfortunately, the cost of implementing this policy in hardware is high, so we have chosen the FIFO (first in-first out) replacement policy for the proposed TLB.

1) Hit in the two-bit comparator for a chosen small bank module: When the CPU generates a virtual address, a subset of the address bits are used to select one of the four bank modules and one of the two small bank modules. If a hit occurs at the two-bit comparator in the enabled small bank module, then the tag part of the filter-bank buffer is enabled and compared for a match of the entire tag field. If the VPN in the filter-bank buffer and the newly generated VPN are identical, the PPN stored in the corresponding data part is sent to the cache and compared with the tag bits of the cache. But if the VPN in the filter-bank buffer differs from the generated VPN, the cache tag comparison is squashed at the filter-bank buffer and its corresponding main bank TLB is accessed for a match during the next cycle. Also during that cycle, the tag part in the filter-bank buffer is updated with the generated VPN in order to store the most recently referenced VPN. If a requested page is found in the main bank TLB, its action is the same as a conventional TLB hit. And also, the data part in the filter-bank buffer is updated at the same time. If the requested page misses in the main bank, the OS invokes its miss handling service.

2) Miss in the two-bit comparator for a chosen small bank module: If a miss occurs at the two-bit comparator, it means that VPN is definitely not in the tag part of corresponding small bank in filter-bank buffer. Thus, the filter-bank buffer comparison can be skipped. Instead, the corresponding main bank TLB is immediately searched in the first cycle, and the filter-bank buffer is simultaneously updated with the new VPN. The combination of a rapid, very-low-power test for the most recent VPN, with the ability to switch to main bank search without delay in most cases, results in significant power savings and minimal loss of performance. As we show in the next section, there are enough accesses to the most recent VPN to justify the use of the filter-bank buffer for power reduction, and the number of two-cycle accesses are sufficiently minimized by the two-bit comparison, where performance is only slightly reduced.

4. PERFORMANCE EVALUATION

Our simulation environment, performance metrics, and power consumption analysis are presented in this section. The benchmarks used in the trace-driven simulation are taken from SPEC95. Four performance metrics, i.e., miss ratio, average memory access time, power consumption, and Energy*Delay product are used to evaluate and compare the proposed TLB system with other approaches. Only data references are collected and used for the simulation. The DineroIV and CACTI simulators [11] were modified to simulate the proposed TLB system. The basic parameters for the simulation are presented in Table 1. These parameters are based on the values used for common 32-bit embedded processors (i.e., Hitachi SH4 or ARM920T).

Table 1. Simulation parameters

System parameters	Values
CPU clock	200 MHz
Memory clock	133 MHz
Memory bandwidth	1.6 Gbytes / sec
Memory latency	70*ns*

4.1 Accuracy and overhead of the selective searching operation

Many preliminary simulations were performed to explore the design space and establish the parameters of the design. For example, the proposed TLB uses two particular bits for initially checking whether the VPN in the filter-bank buffer and the generated VPN are potentially the same. Our simulations showed that when the low order fourth and fifth bits of any given VPN, are compared, it provides the most significant gain with the least overhead. Because of the number of variations explored, we do not present simulations of the different configurations, but instead focus on simulations that enable analysis of the performance and power saving that we achieve in comparison to prior research.

The one aspect of our design that has the potential to cause a loss of performance is that an incorrect prediction by the two-bit comparator can add an extra cycle to the TLB search. We refer to this as two-cycle search overhead, and it is shown in Figure 2. In this figure, two other TLB structures that are also subject to two-cycle overhead are compared with our design. The first one is a filter-TLB, constructed as a small TLB of 4 entries and an L1 TLB with 128 entries. The second one is a plain 4-way banked-TLB structure with four associated filter buffers. Figure 2 shows that the percentages of two-cycle accesses for the filter-TLB, the bank-filter TLB and our TLB turn out to be 22%, 15%, and 3% respectively. Thus, according to the simulation results, our scheme achieves the least overhead in comparison with the other hierarchical TLB structures.

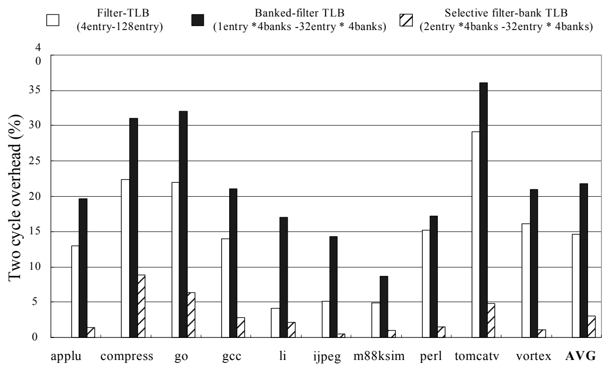

Figure 2. Two-cycle access overhead

Figure 3 shows the percentage of the each TLB hit that was found in the filter-bank buffers versus the main banks in our design. The filter-bank buffers account for over 80%~90% of the hits in most benchmarks.

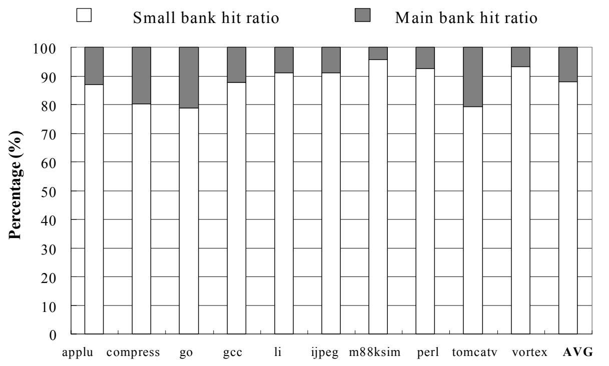

Figure 3. The percentage of the each hit portion for the tag buffers and the main banks

Clearly, significant amount of energy can be saved by avoiding accesses to the main bank TLB by 88% of the time, and instead the much lower power is used for filter-bank buffer logic. Because the two-cycle overhead of our design turns out to be negligible, compared with other hierarchical structures, we also avoid the pitfall of giving up performance in order to save power.

4.2 Miss ratio and average memory access time

In this section we compare three different TLB structures in terms of miss ratio and average memory access time. We do not consider page faults. We assume 15 cycles for miss handling, as in Table 1. CACTI circuit simulations [11] of the fully associative TLB, the small TLB of the filter-TLB, and the banks of a banked TLB show that accessing the fully associative TLB takes more than a single cycle, while the other structures can be accessed in one cycle. However, for our performance evaluations, we simply assume that all of these structures operate in one cycle.

Figure 4 and Figure 5 show the average miss ratio and the average memory access time, respectively for our design and the conventional TLB structures, i.e., a FA (fully-associative)-TLB with 128 entries and a filter-TLB constructed with a small TLB of 4 entries and an L1 TLB with 128 entries, and a four bank-TLB with an associated buffer entry above each bank of 32 entries. In Figure 4, most of the TLB structures can be seen to have similar average miss ratios. However, in terms of the average memory access time, the filter-TLB and the bank-filter TLB show greater performance degradation due to a large number of two-cycle accesses.

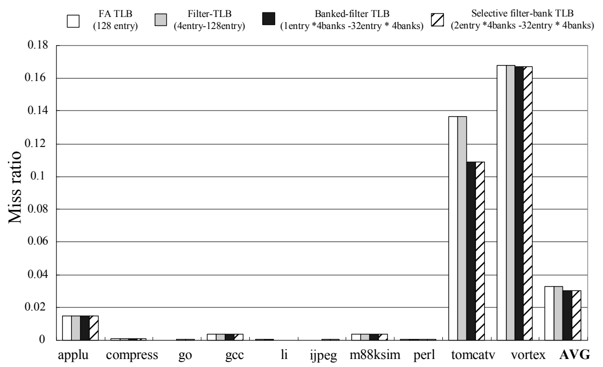

Figure 4. Miss ratios of the selective filter-bank TLB and other TLBs

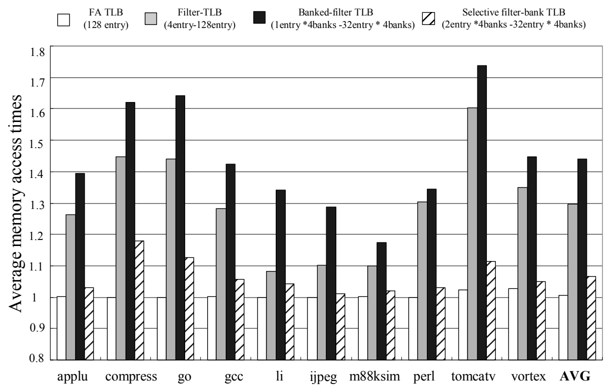

Figure 5. Average memory access times of the selective filter-bank TLB and other TLBs

4.3 Comparison of TLB power consumption

Because all of the entries are searched for every memory access in a fully-associative TLB, one might expect that it would be the worst structure in terms of power consumption. But this is true only when it has more than 64 or 128 entries. Figure 6 shows the average energy dissipation for a TLB access, for various TLB configurations. The fully associative TLB has less power consumption than a set associative TLB when the number of entries is small. This is because the set associative TLB is constructed with more sense amplifiers than the fully associative TLB and these have higher energy consumption. For a 128-entry fully-associative TLB, the energy dissipated at the match line and the bit lines in the CAM reaches the point that it consumes more power than any other TLB configuration.

The overall energy dissipation in the TLB can be divided into two parts, i.e., internal energy dissipation and external energy dissipation. The internal energy dissipation is the energy dissipation within the TLB system when the TLB is accessed. External energy dissipation includes driving the I/O pads for off-chip memory access and searching the data cache for the required page table entry. First, we evaluate power consumption for various TLB configurations using the CACTI simulator, which can calculate access times, cycle times, and power consumption for many types of hardware caches [11, 12].

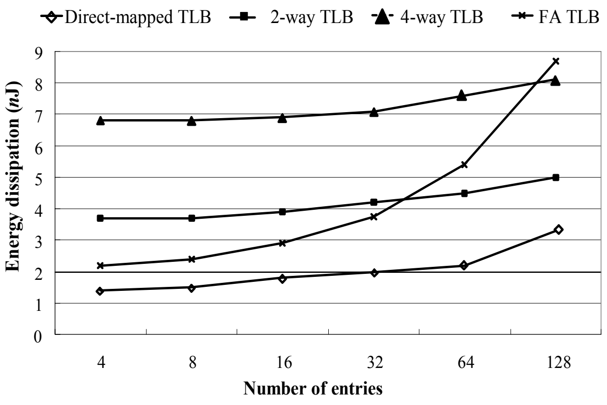

Figure 6. Energy dissipation for a variety of TLB

The CACTI simulator was modified for TLB simulation in several ways. First, the number of bits allocated to a TLB entry is not variable but fixed by the PTE (page table entry) size. Throughout this research, the PTE size was assumed to be 4 bytes. Second, in the cache, the length of the offset field within an address is determined by the size of a cache block, but in the TLB, a predefined page size determines the length of the page offset field for a given virtual address. In the simulation, it is assumed that the page size is 4 Kbytes but that the tag array has sufficient tag width to support a small page size of 1 Kbyte. Additionally there is one valid bit and an 8-bit extension address for each set in the tag array. Finally, CACTI could not originally simulate small caches with fewer than eight sets because its decoder architecture is based on a 3-to-8 decoder block. Thus, we modified the decoder architecture to simulate a 2-to-4 decoder block, which enables used of a 4-entry TLB. Our results are based on 0.8 μm technology with a 4.5 V supply voltage.

Table 2 shows the power consumption for each event corresponding to a TLB access. For a fully associative configuration, most of the power is consumed in the decode stage, where the tag comparison is performed. The significant difference in power consumption between a TLB with 128 entries and one with 64 entries comes from the growth in power consumed by the match line and bit lines in the CAM. Each entry of Table 2 shows the power dissipation for a TLB read hit, a TLB read miss, and a TLB write. As expected, the power consumption almost doubles when the number of entries increases from 32 to 64, and from 64 to 128 respectively.

Table 2. Power consumption per access for TLB sizes

# of entries in FA TLB	Read / Hit (*n*J)	Read / Miss (*n*J)	Write (*n*J)
4	2.2630	0.6154	0.2397
16	2.9209	1.1021	0.5526
32	3.7575	1.7511	0.7097
64	5.4200	3.0490	0.9908
128	8.7272	5.6447	1.8551

The average power consumption of the fully associative TLB is given by:

$$\boldsymbol{Avg.power = N_{hit} * P_{hit} + N_{miss} * P_{miss},} \qquad (1)$$

where N_{hit} and N_{miss} are the ratios of hits and misses in the TLB or small buffer . P_{hit} and P_{miss} are the power required to process a hit and a miss respectively. P_{miss} can be calculated as follows:

$$\boldsymbol{P_{miss} = P_{CAM} + P_{write} + P_{off},} \qquad (2)$$

where P_{CAM} is the power dissipated by all the entries when the tag part of the TLB is accessed, and P_{write} is the power dissipated by the data memory and tag memory in order to update an entry on a miss. P_{off} is the power dissipated by the cache and pads when a TLB miss occurs. Then P_{off} can be calculated as follows:

$$\boldsymbol{P_{off} = P_{cache\_acc} + M_{cache\_miss} * (P_{cache\_write} + P_{pad}),} \quad (3)$$

where P_{cache_acc} is the power used to access a cache block, M_{cache_miss} is the cache miss ratio, P_{cache_write} is the power for a cache write operation on a cache miss, and P_{pad} is the power dissipated at the on-chip pad slot. P_{pad} can be calculated as follows [11,12]. The capacitive load for off-chip destinations is assumed to be *20p*F. Also a 32KB 2-way set associative data cache with 32-byte block size is assumed, where the values of W_{data} and W_{addr} are also 32 bits. The basic parameters for the simulation are summarized in Table 3.

Table 3. Simulation parameters

$\boldsymbol{M_{cache\_miss}}$	0.05
$\boldsymbol{P_{cache\_acc}}$	21.291 *n*J
$\boldsymbol{P_{cache\_write}}$	10.145 *n*J
$\boldsymbol{P_{pad}}$	6.48 *n*J
$\boldsymbol{P_{off}}$	22.122 *n*J

Figure 7 presents the power consumption of the different TLB structures compared to our design for the same set of benchmarks. The power consumption data for the selective filter-bank TLB are obtained by considering all possible cases, such as the power consumed by the comparators, an additional multiplexor, and so on. These values are obtained from the CACTI model indirectly. The figure shows that a filter-TLB or a banked-filter TLB is a good structure in terms of power consumption because the hit ratio at the block buffer exceeds 80%. But their performance degradation tends to be significant because their small buffers are always compared before accessing the main TLB. In our design, the tag part of the filter-bank buffer is only compared when there is a hit in the two-bit comparators. As shown in this figure, power consumption in the proposed TLB can be reduced by about 90% compared with a fully-associative TLB, 70% with respect to a filter-TLB, and 34% compared with a banked-filter TLB.

Figure 8 shows the Energy*Delay product for the different TLB structures. This metric provides a basis to identify a specific TLB configuration that offers the best balance of both power and performance. Simulation results show that the Energy*Delay metric is reduced by about 88%, 75%, and 51% compared with a fully associative TLB, a filter-TLB, and a banked-filter TLB, respectively. Conclusively, the proposed selective filter-bank TLB offers the best result in terms of both performance and power consumption among all of the approaches.

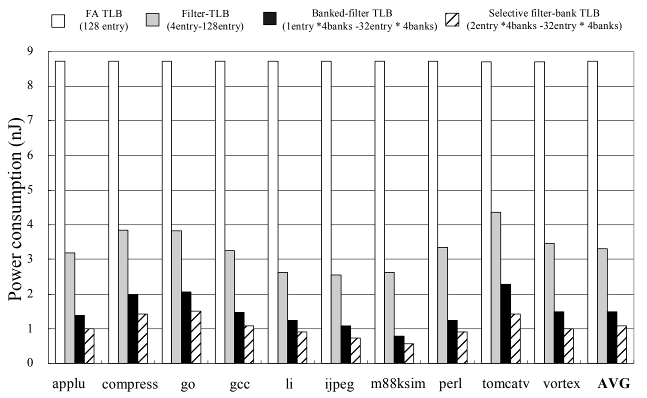

Figure 7. Power comparison of the selective filter-bank TLB and other TLBs

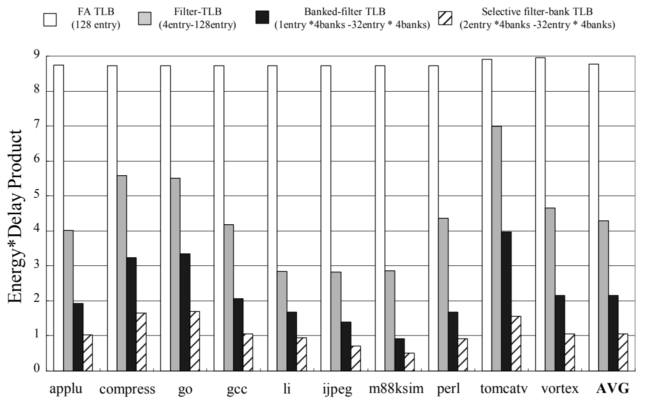

Figure 8. Energy*Delay product of the selective filter-bank TLB and other TLBs

5. CONCLUSION

In order to achieve high performance, recent TLB research for embedded processors tends to support many page entries via large TLB sizes. But in fully associative TLBs, all the entries are searched for every memory access. Because of this, they would be among the worst structures in terms of power consumption. When they have more than 64 or 128 entries, their power consumption is especially high. Therefore, in order to attain low power consumption, a banked-TLB was designed that divides one fully associative TLB space into four smaller fully associative TLBs. To further reduce power consumption, a selective searching mechanism is applied in the proposed TLB to compensate for the weaknesses in the filter-TLB. The amount of energy saved by the proposed TLB strongly depends on the filtering effect of the two-bit comparison that quickly selects between searching a main bank or a small bank in its buffer. This selection avoids the need to search the buffer on every access, thereby saving power. It also reduces the frequency of two-cycle accesses, which reduces the performance penalty incurred by previous low-power designs.

We showed that the average hit ratios of the filter-bank buffers and the main banks of the proposed TLB are 88% and 12% respectively. Simulation results show that the average memory access time of the proposed TLB is almost equal to that of a conventional fully-associative TLB. But the power consumption of the proposed TLB is about 90% less than the fully-associative TLB, 70% less than a filter-TLB, and 34% less than a banked-TLB with block buffering. Thus, the Energy*Delay metric is reduced by about 88%, 75%, and 51% compared with a fully associative TLB, a filter-TLB, and a banked-filter TLB, respectively.

6. ACKNOWLEDGMENTS

This work was supported by Samsung Co. “Design of Memory Management Unit for Embedded Processors” project.

7. REFERENCES

[1] T. M. Austin, et al., High-bandwidth address translation for multiple-issue processors in the Proceedings of the 23rd ACM Int’l Symp. on Computer Architecture, (May 1996) 158-167.

[2] J. Kin, M. Gupta, et al., The Filter Cache: An Energy Efficient Memory Structure in the Proceedings of Int’l Symp. Microarchitecture, (1997), 184-193.

[3] K. Ghose, et al., Reducing Power in Superscalar Processor Caches Using Subbanking, Multiple Line Buffers and Bit-Line Segmentation in the Proceedings of International Symposium on Low Power Electronics and Design, (August 1999), 70-75.

[4] S. Manne, et al., Low power TLB Design for High Performance Microprocessors. Univ. of Colorado Technical Report, 1997.

[5] B. L. Jacob, et al., Virtual Memory in Contemporary microprocessors. IEEE Micro, Vol. 18, No. 4, July/August 1998, pp. 60-75.

[6] B. L. Jacob, et al., Virtual Memory: Issues of Implementation, IEEE Computer, Vol. 31, No. 6, (June 1998), 33-43.

[7] T. Juan, et al., Reducing TLB Power Requirements in the Proceedings of ISLPED, 1997.

[8] K. Itoh, et al., Trends in Low-Power RAM Circuit Technologies in the Proceedings of the IEEE, vol. 83, no. 4, (April 1995), 524-543.

[9] D. Liu, et al., Trading Speed for Low Power by Choice of Supply and Threshold Voltages. IEEE Journal of Solid State Circuits, Vol. 28, No. 1, 1993.

[10] G. Reinman, et al., CACTI 3.0: An Integrated Cache Timing and Power, and Area Model. Compaq WRL Report, August 2001.

[11] M. B. Kamble, et al., Energy-Efficiency of VLSI Cache: A Comparative Study in the Proceedings of IEEE 10-th. Int’l. Conf. On VLSI Design, (Jan.1997), 261-267.

[12] M. B. Kamble, et al., Analytical Energy Dissipation Models for Low Power Caches in the Proceedings of ISLPED, Aug. 1997.