Single Cycle Access Cache for the Misaligned Data and Instruction Prefetch

Joon-Seo Yim, Hee-Choul Lee, Tae-Hoon Kim, Bong-Il Park,
Chang-Jae Park, In-Cheol Park, and Chong-Min Kyung
Department of Electrical Engineering
KAIST
Taejon, 305-701, Korea
Tel: +82-42-866-0700
Fax: +82-42-866-0702
e-mail: kyung@dalnara.kaist.ac.kr

Abstract— In microprocessors, reducing the cache access time and the pipeline stall is critical to improve the system performance. To overcome the pipeline stall caused by the misaligned multi-words data or multi cycle accesses of prefetch codes which are placed over two cache lines, we proposed the *Separated Word-line Decoding (SEWD)* cache. SEWD cache makes it possible to access misaligned multiple words as well as aligned words in one clock cycle. This feature is invaluable in most microprocessors because the branch target address is usually misaligned, and many of data accesses are misaligned. 8K-byte SEWD cache chip consists of 489,000 transistors on a die size of 0.853 $\times$ 0.827 cm^2 and is implemented in 0.8 μm DLM CMOS process operating at 60 MHz.

I. Introduction

The advancement of VLSI implementation technology leads to the high-performance microprocessors that operate at more than 500 MHz and integrates three million transistors on a single chip. The performance of cache, which occupies large silicon area as much as 40%, has become the bottleneck of microprocessor. Hence, many researches are now focusing on the performance enhancement of cache in microprocessors[1].

In the state-of-the-art microprocessors, the data bus size is as much as 64 bits. If the memory operands are placed over a cache line boundary, the pipeline should stall until the required full word data are available. For the example shown in Fig.1, the misaligned data access needs two extra cycle penalty. At the first cycle, lower line($line[i]$) is read out from cache and saved in a read buffer temporally, and higher line($line[i+1]$) is read out at the second cycle. The lower word of read buffer is aligned with higher word using a shift matrix. During the two extra cycles, the pipeline is stalled, which is called here *cache line boundary problem* due to the address misalignment.

In the superscalar microprocessor, the bandwidth of the instruction prefetch should be large in order that there is no shortage of instructions to execute in the multiple pipes. Generally the amount of code prefetch is the same

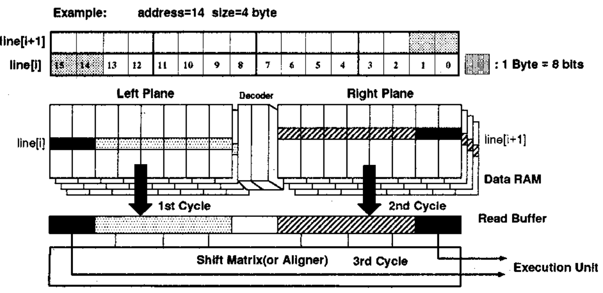

Fig. 1. Cache Line Boundary Problem Handling

as the cache line size. In CISC microprocessors, especially in VAX or X86, the instruction length varies from 1 byte to 15 bytes[2]. Therefore, the branch target positions are random in the cache line. Moreover, the frequency of branch instruction is so high, one per four or five instructions[2], that the branch target address is usually misaligned. To minimize the address misalignment penalty, the high level compilers often fill the NOP(No Operation) codes into an original program to align the target branch to the cache line boundary. According to the benchmark reports[3], the aligned code improves the performance by 30% over the misaligned code.

We propose a "*Separated Word-line Decoding(SEWD)*" architecture which can access the misaligned branch target code or the misaligned data in one cycle as well as aligned code/data.

II. SEWD Architecture

To handle the misalignment in a single cycle, two adjacent cache line should be accessed simultaneously. In a traditional architecture, the decoding patterns are same for the left and right plane in Fig.1. Therefore, two adjacent word-line can not be driven simultaneously.

We modified the decoder to access the adjacent cache line concurrently. For a misaligned address, $line[i]$ at the left plane and $line[i+1]$ at the right plane should be driven. That is, at the left plane, $line[i]$ is selected regardless of misalignment or not, but at the right plane, $line[i]$

is selected for the aligned case and $line[i+1]$ for the misaligned case. To make this mechanism possible, the left plane uses a normal decoder in Fig.2(a), while the right plane uses a SEWD decoder in Fig.2(b). If an address is not aligned, `side` signal is activated to drive $line[i+1]$, otherwise `self` signal is activated to drive $line[i]$ in a SEWD decoder.

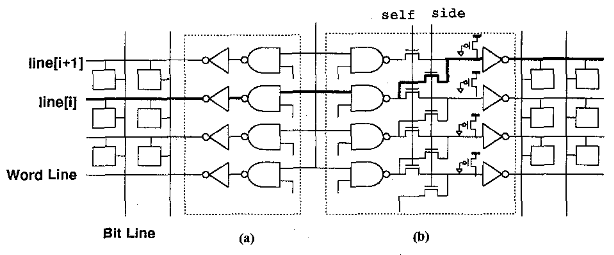

Fig. 2. (a) Normal decoder at left plane (b) SEWD decoder at right plane

Using a SEWD decoder, two adjacent lines in data RAM can be accessed. But the adjacent lines in cache are not guaranteed to be successive at main memory. To know whether the two adjacent cache lines have a same tag address, the adjacent tag RAM lines should also be accessed simultaneously. We should read out $tag[i]$ and $tag[i+1]$ and compare them with physical address simultaneously to give $hit[i]$ and $hit[i+1]$. This is impossible in traditional tag RAM architecture as shown in Fig. 3(a).

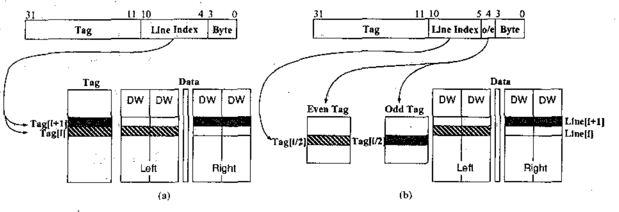

Fig. 3. (a) Traditional Architecture, and (b) SEWD Architecture

To handle this problem, we also divide tag RAM into even and odd parts called as "*Even/Odd Alternating Tag Array*" as shown in Fig.3(b). The LSB of line index in address determines the even/odd array in SEWD architecture as shown in Fig.3(b). The result of simultaneous read out can be one of three cases : complete hit(both $line[i]$ and $line[i+1]$ is `hit`), partial hit(one of them is `hit`), and complete miss(both are `miss`). Generally, cache hit ratio is over 98%[2], therefore, most of the misaligned data/code can be accessed in one cycle.

III. Chip Implementation

To implement the SEWD architecture, the tag array is broken into two parts, it needs extra read/write circuits and comparators. The SEWD tag area increased by 15.8%, but total cache area overhead is just 2.35% in our 0.8μm CMOS process. The SEWD reduced the bitline length to 1/2, thus improved the hit detection timing from 7.18ns to 5.59ns. This is another important benefit of SEWD architecture.

To perform the complicated micro-level operations within one clock cycle, and to minimize the power consumption, the self-timed circuit technique is utilized. All the internal timing signals are generated with timing modeler. We uses a pulsed predecoder and a strobed sense amplifier during the minimum required time to minimize the dynamic current.

For our design of 4-way set associative 8K-byte on-chip cache. Data RAM consists of 128 lines, with each line consisting of 16 bytes, where each RAM cell is based on 11.2μm $*$ 18.8μm 6 transistor SRAM cell. Fig. 4 shows the photograph of test chip, This test chip is fabricated in 0.8 μm DLM CMOS process. A total number of transistors is 489,000 and the chip area is 0.853 $\times$ 0.827 cm^2. The chip was proven to work correctly up to 60MHz.

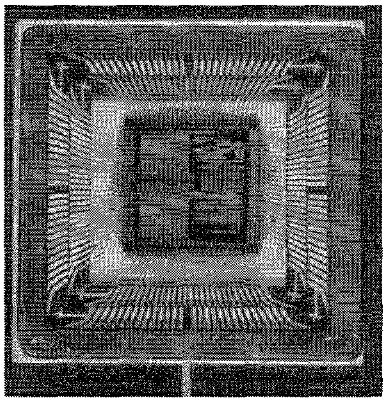

Fig. 4. Photograph of Test Chip

IV. Conclusion

We designed on-chip cache with a proposed separated word-line decoding architecture to minimize the misalignment penalty for the multiple words data and misaligned branch target prefetch. In the chip implementation in 0.8 μm DLM CMOS technology, the critical path timing was reduced from 7.18ns to 5.59ns, at the area increment of only 2.3%. This is a significant benefit of SEWD architecture since the cache is normally belonging to one of critical paths in high-performance processors.

References

[1] M.Motomura, et.al.,"Cache-Processor Coupling: A Fast and Wide On-Chip Data Cache Design", *IEEE JSSC*, vol.30(4), pp.375-382, Apr.,1995

[2] J.L. Hennessy and D.A. Patterson, "Computer Architecture A Quantitative Approach", Morgan Kaufmann Publishers,1990

[3] Tom R. Halfhill, "Intel Launches Rocket in a socket", Byte, pp.92-108, May, 1993