A High Performance Sequential MRU Cache Using Valid-Bit Pre-Decision Search Algorithm

Hsin-Chuan Chen
Department of Electrical Engineering, St. John's University
robin@mail.sju.edu.tw

ABSTRACT

MRU (most recently used) cache is one of the set-associative caches that emphasize implementation of associativity higher than two. However, the access time is increased because the MRU information must be fetched before accessing the sequential MRU cache. In this paper, focusing on the sequential MRU cache with sub-block placement, we propose an MRU cache scheme that separates the valid bits from data memory and uses these valid bits to decide to reduce the unnecessary access number of memory banks. By this approach, the probability of the front hits is thus increased, and it significantly helps in improving the average access time of the sequential MRU cache without valid-bit pre-decision search especially for large associativity and small sub-block size.

1: INTRODUCTIONS

The cache memory has played an important role to reduce the speed gap between processor and main memory. In several cache organizations, a direct-mapped cache has fast hit access time because the CPU can directly read data from the data bank without waiting for the tag checking. However, the direct-mapped cache has a higher miss rate. On the other hand, a set-associative cache has a lower miss rate because blocks from the main memory can map into any cache block of one fixed set in cache, but it needs higher hardware complexity and suffers from a longer hit access time [4]. Therefore, how to maintain a low overall average access time is an important issue in the design of a set-associative cache.

In the past, researches related to the set-associative cache, such as hash-rehash cache (HR-cache) proposed by Agarwal et al. [8] and column-associative cache (CA-cache) proposed by Agarwal and Pudar [1] are based on the organization of the direct-mapped cache, and they use hashing functions to improve the miss rate. The difference between these two approaches is that the CA-cache uses a rehash bit associated with each cache block to indicate that the data stored in the cache block will be found using the second hash function. Another set-associative cache scheme, predictive sequential associative cache (PSA-cache) [2], uses a steering bit table (SBT) to dynamically determine which block is probed first when the cache is accessed. By providing prediction information, the miss rate of PSA-cache can be as low as that of 2-way set-associative caches, and the cycle time is similar to that of the direct-mapped caches. However, those cache schemes described above are not suitable for the implementation of set-associative caches with higher associativity. Therefore, the MRU (Most Recently Used) cache [6][7] that is similar to PSA-cache but uses an MRU table to determine the first probed block location in a set of the cache, while a set is referred to, the probability to find the correct block location in this set at the first time is very high [10]. Therefore, the MRU cache can be considered to develop set-associative caches with higher associativity.

In this paper, a new sequential MRU cache scheme with sub-block placement is proposed; however, those corresponding valid bits of sub-blocks can be used to decide which sub-blocks need to be probed in advance. The proposed sequential MRU cache without much hardware cost thus can effectively improve the average access time due to reducing many unnecessary probes of the tag and data memories during the search period.

2: SEQUENTIAL MRU CACHE

Kessler's scheme [5] uses a sequential search to find the desired block in a set according to the content of the MRU table. In this sequential cache (SMRU cache), both the tag memory and the data memory are single bank, and only one comparator is required. The MRU table stores the block bits that represent the most recently used block number for each set and determine the search order which is from most-recently-used (MRU) to least-recently-used (LRU). For example: in a 4-way set-associative MRU cache, if the MRU block list for one set is "01001110", that means the search order of the locations is 1, 0, 3 and 2. The block bits indicating the present desired block location are used to associate the set bits of main memory address to form an effective address as accessing the tag bank and data bank. Due to using a true LRU replacement policy, the MRU block list for each set can be maintained by the cache system. In this paper, we focus on the sequential MRU cache with sub-block placement, and the following sub-sections will discuss the operations of the sequential MRU cache and the sub-block placement.

2.1: OPERATIONS

The operations of the sequential MRU cache are described as follows [11]:

(1) While a set of the cache is referred to, the cache system fetches the MRU table to obtain the MRU block list, and then these block bits of the MRU block list are used to form the address of the tag bank and data bank. This operation should be prior to accessing the tag bank and data bank.
(2) According to the first block bits taken from the MRU block list, the first MRU block location is probed.
(3) The cache system checks the tag of the selected block location. If the first hit occurs, the block data are read out from the data bank like the direct-mapped cache; however, two access cycles are required for the first probe.
(4) If no first hit occurs, the cache system continuously checks the rest blocks in this set and selects the next probed block from the MRU block list until all tags of this set are examined.
(5) When a miss occurs, the cache system will take more cycles to refill a new block from the lower-level memory to perform the replacement operation.

Because of sequential search, the sequential MRU cache has a longer average access time than that of other cache schemes [3]. However, the sequential MRU cache with high associativity can be used as a low cost level 2 cache in a two-level multiprocessor cache architectures to reduce memory interconnection traffic [5].

2.2: SUB-BLOCK PLACEMENT

Increasing block size will reduce the tag memory size for an on-chip cache design [12]; however, the large miss penalty is incurred due to large block size. Usually, the sub-block placement [13], which only refills a part of the entire block into the cache when the miss occurs, is an appropriate approach to reduce the miss penalty. In this cache scheme, each data block is divided into several sub-blocks, and each sub-block has a corresponding valid bit to indicate if this sub-block exists in the cache. Therefore, for a set-associative cache with sub-block placement, when the cache is accessed, in addition to tag checking of all ways, the corresponding valid bits of all ways must be checked together.

3: PROPOSED MRU CACHE

For design of the sequential caches, a large number of front hits help in achieving a low average access time. When a sequential MRU cache employs the sub-block placement to reduce its miss penalty, fortunately, the valid bits can be used to pre-eliminate the unnecessary search times for each cache access, such that it can make the original rear hits become more front hits. Based on this idea, a new sequential MRU cache with valid-bit pre-decision search (called SMRU-V cache) is proposed to reduce the average access time.

3.1: VALID-BIT PRE-DECISION SEARCH

In the conventional sequential MRU cache, the search order always starts from the MRU block to the LRU block one by one. Even though the present probed block does not exist (i.e. the valid bit = "0"), it still must complete checking the present block before probing the next block, which means this search is redundant. In the proposed SMRU-V cache, the search order is the same as that of the SMRU cache. However, the valid bits of the sub-blocks for different ways in one set can be used to decide which sub-blocks need to be examined during the search process. For an *n*-way SMRU-V cache, the valid-bit pre-decision search algorithm is shown in Fig. 1, and Fig. 2 illustrates two search approaches at 4-way, respectively. According to the valid-bit pre-decision algorithm, the proposed SMRU-V cache can make block 2 become the 2nd hit from the original 4th hit for the SMRU cache, and thus it only requires two search times. Consequently, for a cache with small sub-block size, such a search algorithm can achieve more front hits and reduce many unnecessary search times with the valid bits being "0" on a cache hit, and it also can help in reducing the miss search times even when a cache miss occurs.

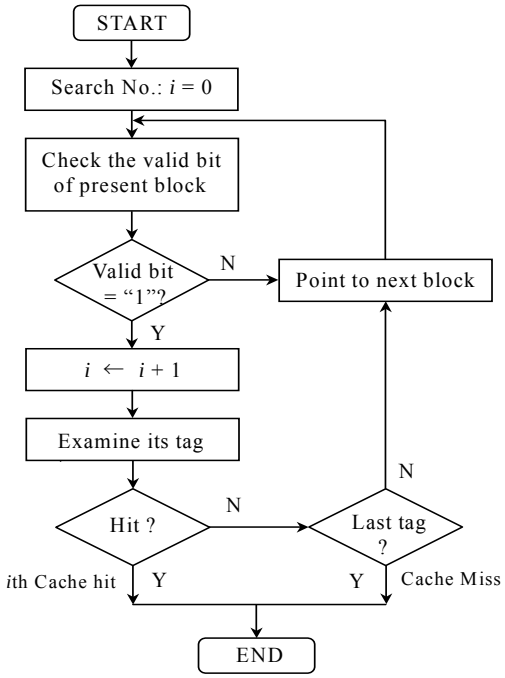

Fig. 1 Valid-bit pre-decision search algorithm

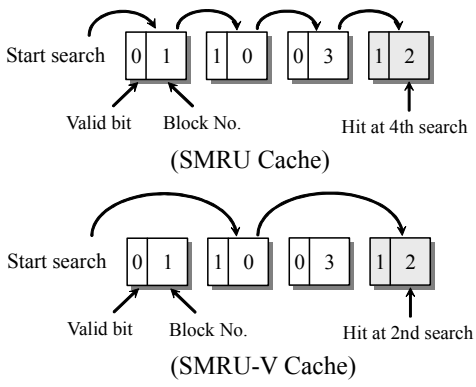

Fig. 2 Search approaches for SMRU cache and SMRU-V cache

3.2: ARCHITECTURE

The architecture of the SMRU-V cache shown in Fig. 3 only modifies the data memory organization of the original sequential MRU cache, which the valid bits of all sub-blocks are also separated from the data memory bank, and they are organized as a single *n*-bit valid-bit bank and each bit represents one valid bit of the accessed sub-block for each way. The bit order is from the MRU way (MSB bit) to the LRU way (LSB bit) for each set. When the cache is referred to, all memory banks including the MRU table and valid-bit bank are accessed concurrently. The content of the MRU table is the same as that of the SMRU cache, and the search order is from the MRU block location to the LRU block location. The valid bits stored in the valid-bit bank can be read to decide which block locations are needed to be probed when the MRU block bits are taken from the MRU table. Note that the LRU replacement circuit of the cache system can maintain the content of the MRU table and valid-bit bank after each cache access.

3.3: OPERATIONS

The main difference between the SMRU cache and SMRU-V cache is their search process, and the operations of the SMRU-V cache are showed as follows:

(1) While a set of the cache is referred to, the cache system fetches the MRU table and the valid-bit bank, and the control circuit takes the first MRU block bits that its corresponding valid bit is "1" from the MRU block list to form the address of the tag bank and data bank for the first MRU block.

(2) The cache system checks the tag of the first MRU block location selected by the first MRU block bits. Simultaneously, these bits are also used to speculatively select the data of the first MRU block location.

(3) If the first hit occurs, similar to the direct-mapped cache, the desired block data are directly read out from one of the *n* data banks; however, two access cycles are required for the first probe.

(4) If the first hit does not occur, according to the valid bits with "1", the control circuit selects the next MRU block from the MRU block list in order, and checks the rest blocks in this set until all tags of this set are examined. If any hit is found again, the last selected MRU block bits are used to select the desired block data.

(5) When a miss occurs, the cache system will take more cycles to refill a new block from the lower-level memory to perform the replacement operation. Simultaneously, the status of the valid-bit bank will be maintained.

In our proposed cache architecture, due to the valid-bit pre-decision search and the support of LRU replacement algorithm, most of the increased front hits are first hits. Therefore, the first hit rate will be higher than that of the sequential MRU cache without valid-bit pre-decision search.

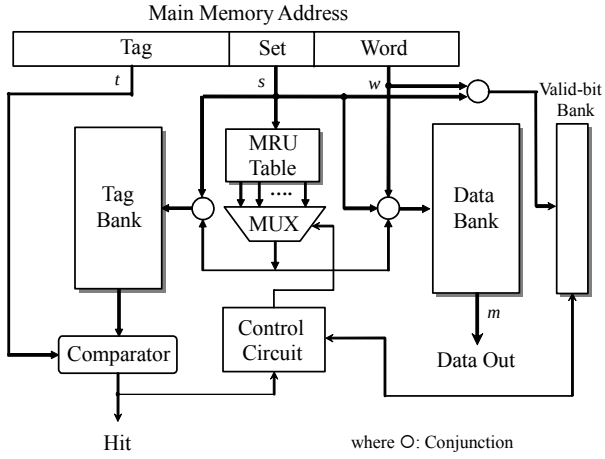

Fig. 3 Architecture of SMRU-V cache

3.4: OVERHEADS

In this proposed SMRU-V cache, due to concurrent accesses of the MRU table and valid-bit bank at the first cycle, almost no extra access time is needed. To implement the valid-bit pre-decision search, the control logic indeed requires the extra hardware components instead of the binary counter within the control logic of the conventional SMRU cache. In this search decision circuit [15], the valid bits are latched by D-type flip-flops, and then sent to the priority encoder that decides to select the desired MRU block bits according to the corresponding valid bit with "1" at each search clock from MRU block to LRU block. Simultaneously, the output of the priority encoder also feedbacks to the decoder to clear the previous searched priority input after each clock. Table 1 shows the required hardware components of the control logic for two sequential caches at 4-way. The incurred delay time compared with the SMRU cache is caused by this search decision circuit, and it can be neglected due to only two-level logical gate propagation (pass through priority encoder) compared with the access time of memory banks. Therefore, without much delay time, the proposed SMRU-V cache still maintains the low cost implementation as that of the conventional sequential MRU cache.

Table 1 Hardware Comparison at 4-way

Cache Types / Required Hardware	SMRU	SMRU-V
D flip-flops	2	4
Logic Gates	7	13

4: PERFORMANCE METRICS

In the sequential MRU caches, the average access time mainly depends on the first hit rate, the number

distribution of hits in different times and hardware complexity. If most of the cache accesses are first hit and few search times are required for each cache access, the total average access time is low even a sequential search technique is used. Based on the operations of the SMRU cache in Section 2, the average access time (T_{SMRU}) for an associativity of n can be given by:

$$T_{SMRU} = H_1 \times 2 + \left[\sum_{i=2}^{n} H_i \times (i+1)\right] + (n+1+P) \times M \qquad (1)$$

where H_i is the ith hit rate of the cache; M is the miss rate of the cache, and P is the miss penalty that depends on the sub-block size. For an n-way SMRU-V cache, the equation of its average access time ($T_{SMRU\text{-}V}$) similar to that of the SMRU cache can be expressed by:

$$T_{SMRU\text{-}V} = H_1' \times 2 + \left[\sum_{i=2}^{n} H_i' \times (i+1)\right] + (m+1+P) \times M \qquad (2)$$

However, the ith hit rate of the cache (H_i') and miss search times ($m \leqq n$) differ from the SMRU cache due to the valid-bit pre-decision search. Because the valid bits of all MRU blocks at the first hit are always equal to "1", and thus the first hits of the SMRU-V cache contains the original first hits and some new increased first hits that come from the original rear hits. Here, H_1' and H_i' are respectively given by:

$$H_1' = H_1 + \sum_{k=2}^{n} H_k \times V_{k1} \qquad (3)$$

$$H_i' = H_i \times \prod_{j=1}^{i-1} (1 - V_{ij}) + \sum_{k=i+1}^{n} H_k \times V_{ki} \qquad (4)$$

where V_{ij} (V_{ki}) denotes the probability to become the jth (ith) hit from the original ith (kth) hit. Therefore, when the associativity is high (i.e., n is large) and the sub-block size is small, the improvement in average access time of the SMRU-V cache is more significant than that of the SMRU cache.

5: SIMULATION RESULTS

To evaluate and analyze the performance of the proposed cache architectures, we use a trace-driven cache simulator (Dinero) [14] to simulate the access behaviors of all sequential MRU caches. In our simulation, all cache architectures have the same cache size (= 32 KB), block size (= 32 Bytes), and replacement policy (LRU). According to the operations of the different cache schemes, the average access time of the proposed cache can be evaluated by re-modeling Dinero to trace various trace programs [14] which some are belonged to SPEC benchmark suite such as SPICE, GCC, and XLISP.

For the MRU caches using a sequential search, the number of hits in different times is also an important factor to influence the average access time. Because the SMRU-V cache using a valid-bit pre-decision search algorithm can reduce unnecessary search times, many original rear hits become the front hits, and we find that the first hits increase 5% on average. Consequently, its front hits are more than other sequential MRU caches. Basically, when the associativity increases or the sub-block decreases, the first hit rate of the SMRU cache will decrease [9]. However, the proposed SMRU-V cache can significantly improve the first hit rate of the SMRU caches especially for the cache with a large associativity and a small sub-block size (shown in Fig. 4).

In our simulation, we change the range of the sub-block size to observe the average access time of two sequential MRU caches shown in Fig. 5. When the sub-block size decreases at the fixed associativity = 32, the average access time of the SMRU cache decrease at first due to the reduction of miss penalty. However, the miss rate increases and the first hit rate decreases as the sub-block size decreases, and thus the average access time increases again until the sub-block size = 4 bytes. When the sub-block size is less than 4 bytes, the average access time starts decreasing because the reduction of miss penalty becomes more obvious. Contrary to the variety trend of the SMRU cache, the average access time of the proposed SMRU-V is always less than that of the SMRU cache as the sub-block size decreases.

Fig. 6 indicates the improvement of the SMRU-V cache over the SMRU cache in average access time. Due to that the front hits of the SMTU-V cache increase as the sub-block size decreases, consequently, except for the sub-block size = 32 bytes, the improved rate IMR_{TAS} of the SMRU-V cache will increase as the associativity increases at the fixed sub-block size, or as the sub-block size decreases at the fixed associativity.

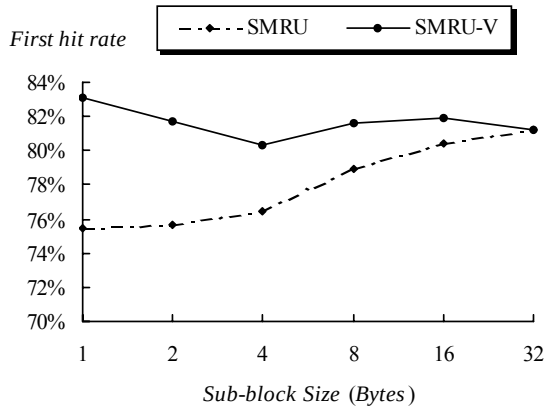

Fig. 4 First hit rate vs. sub-block size at 32-way

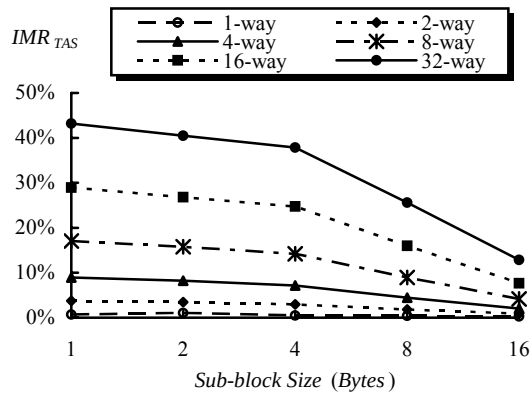

Fig. 5 Improved rate in access time

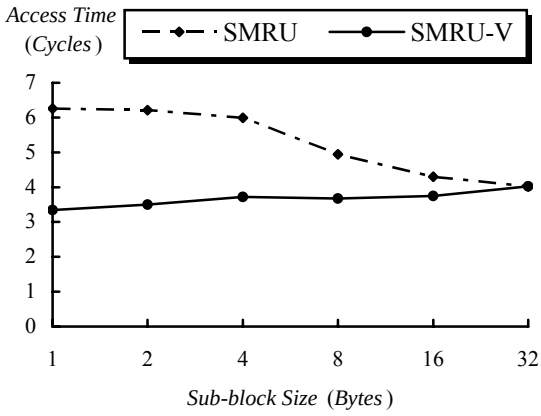

Fig. 6 Average access time vs. sub-block size at 32-way

6: CONCLUSIONS

In this paper, a valid-bit pre-decision search algorithm applied to the sequential MRU cache is proposed to improve the average access time of the conventional sequential MRU cache with sub-block placement. Without adding much hardware in our proposed MRU cache scheme, many unnecessary search times are eliminated and more first hits are obtained by the valid-bit pre-decision. From simulation results, the improved rate in average access time can achieve about 40% on average at 32-way and the sub-block size = 4 bytes, even at the associativity = 4, the proposed SMRU-V still has about 7% improvement. Therefore, for achieving more significant improvement in the average access time, the proposed SMRU-V cache is suitable for large associativity and small sub-block size. Moreover, being a level two cache, the proposed SMRU-V cache still maintains the benefit of low cost implementation as that of the conventional sequential MRU cache.

7: REFERENCES

[1] Agarwal and S. D. Pudar, "Column-associative caches: A technique for reducing the miss rate of direct-Mapped caches", *Proc. 20th Annual International Symposium on Computer Architecture*, pp. 179-190, 1993.

[2] B. Calder, D. Grunwald, and J. Emer, "Predictive sequential associative cache", *Proc. 2nd International Symposium on High Performance Computer Architecture*, pp. 244-253, Feb. 1996,.

[3] C. Zhang, X. Zhang, and Y. Yan, "Two fast and high-Associative cache schemes", *IEEE Micro*., vol. 17, pp. 40-49, 1997.

[4] M. D. Hill, "A case for direct-Mapped caches", *IEEE Computer*, vol. 21, pp. 25-40, 1988.

[5] R. Kessler, R. Jose, A. Lebeck and M. Hill, "Inexpensive implementations of set-associativity", *Proc. 16th Annual International Symposium on Computer Architecture*, pp. 131-139, May 1989.

[6] K. So and R. Rechtschaffen, "Cache operations by MRU change", *IEEE Transactions on Computers*, vol. 37, pp. 700-709, 1988.

[7] C. Wu, Y. Hsu, and Y. Liu, "A quantitative evaluation of cache types", *Proc. 26th Hawaii International Conference on System Sciences*, vol. 1, pp. 476-485, 1993.

[8] A. Agarwal, J. Hennessy, and M. Horowitz, "Cache performance of operating systems and multiprogramming", ACM Transactions on Computer Systems, vol. 6, pp. 393-431, 1988.

[9] A. Seznec, "DASC cache", *Proc. 1st IEEE Symposium on High-Performance Computer Architecture*, pp. 134-143, 1995.

[10] K. Inoue, T. Ishihara, and K. Murakami, "A high-performance and low-power cache architecture with speculative way-selection", *IEICE Transactions on Electron*, vol. E83-C, pp. 186-193, 2000.

[11] H. C. Chen, J. S. Chiang and Y. S. Lin, "A fast sequential MRU cache with competitive hardware cost", *2001 The 2nd International Conference on Parallel and Distributed Computing, Application and Technologies*, pp. 220-227, Jul. 2001.

[12] M. D. Hill and A. J. Smith, "Experimental evaluation of on-chip microprocessor cache memories", *Proc. 11th Annual International Symposium on Computer Architecture*, pp. 158-166, 1984.

[13] J. L. Hennessy and D. A. Patterson, "Computer architecture a quantitative approach", *Morgan Kaufman Publishers, Inc*., 2nd Edition, pp. 412-413, 1997.

[14] M. Hill, DINERO III Cache Simulator: Code and Documentation, *University of Wisconsin at Madison*, 1998.

[15] J. P. Hayes, "Computer Architecture and Organization", *The McGraw-Hill Companies, Inc.*, 2nd Ed., pp. 449-458, 1988.