DHL-cache: dynamic per history length adjustment for low-power L2 cache

H.W. Joo and E.Y. Chung✉

The significant power dissipation of the on-chip L2 cache is a major concern in modern microprocessors. This Letter proposes a unique low-power technique for a high-associative large L2 cache that has fragmented locality by L1 cache. The dynamic per history length adjustment cache (DHL-cache) dynamically selects the qualified way candidates to be accessed and controls the way-prediction window size based on the access history pattern. With a high degree of the way-prediction accuracy, the DHL-cache shows a 55.3% energy-delay product improvement over the location-cache with minimum hardware support. Therefore, the DHL-cache alleviates the power limit issue for L2 cache, even with high associativity and fragmented locality.

Introduction: The continuous development of the silicon process has delivered dramatic performance improvements in microprocessors. Moreover, the size of on-chip L2 cache has recently become larger with higher associativity in order to meet the increased high performance requirements. However, high-associativity cache structures aggravate the power limit issue for the modern microprocessors. To alleviate power limit issues, extensive studies have been performed. The most effective low dynamic power techniques are the way-predicting (WP) technique [1] and the way-determination (WD) technique, such as the location-cache [2]. Even though the WD scheme shows a certain degree of way prediction accuracy, it requires not only large hardware resources but also an additional delay to trace where the appropriate data is located in L2 cache. The WP scheme requires less hardware resources than that of WD; however, it is only compatible for the strong locality property as well as low-associativity. Especially the L2 cache locality is easily fragmented by L1 cache replacement, the WP is not suitable for L2 cache. Thus, unlike existing WP techniques, this Letter devises a dynamic per history length adjustment policy (DHL-cache) to achieve a high way-prediction rate even for a high-associative large L2 cache with fragmented locality based on way-usage pattern history.

Motivation: As there are many mixed-data usage patterns with fragmented locality that depend on the L1 cache replacement conditions [3], the WP scheme shows low performance at the L2 cache. Even if the working set is larger than the available L2 cache size, frequent conflict misses more severely erode the locality. As identifying the proper access pattern is a key factor for improving way-prediction accuracy, we adopt a history-based algorithm that has benefit of correctly classifying each access pattern. However, the access patterns are mixed in L2 cache, the current access pattern should be sorted with valid history information only. Because the access history consists of different patterns, sampling of the valid history information is a precondition for properly determining the pattern. Fig. 1 illustrates an example of the mixed-data usage histories assuming the three different patterns exist in the L2 cache with multiple way usage. As no correlation exists among the data indicated with dotted lines, it could be considered noise for determining the proper locality for each pattern. Only the data indicated with solid lines have valid correlations among the data stream. Consequently, the sorted valid history information helps to find where an appropriate way candidate is located. Thus, the DHL-cache utilises the dynamic per history length adjustment to improve the way-prediction accuracy.

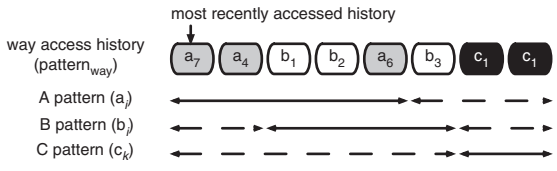

Fig. 1 *Example of mixed patterns in data flow*

Proposed model: As illustrated in Fig. 2, the DHL-cache consists of the way-prediction buffer (WPB) with a weighted-least recently used replacement policy for the zero-reuse line [4], auxiliary pattern history table (aPHT), history mask bit, and control logic. The accessed way-number information should be stored in chronological order in the WPB for each set. Further, when a prediction or cache miss occurs, the proposed cache operates like conventional cache, and the correct way-number is updated to the most recently used (MRU) position of the WPB entry. To obtain valid history information, the DHL-cache stores history indicator bits for each set in the aPHT and adopts a selective history length adjustment to ensure reliability using the history mask bit. The aPHT, which is a type of shift register, stores 16 history indicator bits for every access per each set. The history indicator bits are generated by the control logic as shown in Table 1. Further, the history mask bit determines the valid history length for properly classifying the access pattern. The history mask bit sets the range from the MSB of the history indicator bits to the number of past history utilisations. For example, in the Fig. 2, the three history mask bits imply that the DHL-cache uses only three recently accessed histories to determine current pattern. Using the 'And' operation between the 16 history indicator bits from the aPHT and the history mask bit, the modified history pattern is generated. Control logic determines current access pattern with the modified history pattern. With such an algorithm, the DHL-cache could utilise the dynamic per history length adjustment for every set to identify accurate current access pattern history with minimum additional logic requirement.

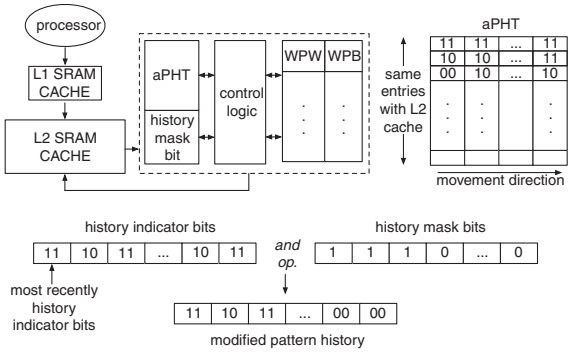

Fig. 2 *DHL-cache structure*

Table 1: History indicator bits

	Prediction hit	Prediction miss
Cache hit	11	10
Cache miss	01	Not-happened

Architectural policy: While the existing WP scheme uses only the last cache history information alone to predict the next cache way candidate, the DHL-cache selects several qualified way candidates within meaningful past cache history information to precisely predict the current cache access pattern. The DHL-cache dynamically controls the way-prediction window (WPW) size and the proper history length adjustment to discriminate the current data pattern. If the WPW size is n for a certain cache set, the DHL-cache draws n way candidates from the MRU position of the WPB. As the WPB stores the accessed way number in chronological order, the DHL-cache does not have to store additional tag information for way-prediction.

Fig. 3 depicts the finite state machine diagram of the DHL-cache. If the DHL-cache encounters a cache-miss (01) in the modified pattern history, the DHL-cache enters state *S1*; it flushes history information stored in the aPHT. Thus, the DHL-cache sets the WPW size and the history mask bit to the maximum value for learning new pattern. If the cache has m-associativity, the maximum value of the WPW size is m. If the DHL-cache encounters a cache-hit (1x), the DHL-cache enters state *S2*; it indicates that the access history pattern does not exist or the access history pattern has been changed. However, for both cases, the existing access pattern information is no longer valid. Thus, it decreases the history mask bit by 1 to remove noise history pattern information to grasp proper access pattern. Further, if previous state is *S1*, it increases the WPW size by 1 to search more appropriate way candidate. If the DHL-cache detects a subsequent prediction hit pattern on cache hit (1111) in the modified pattern history streams, it indicates that the DHL-cache grasps proper access pattern. Thus, the DHL-cache enters state *S3*; it decreases the WPW size by 1 to save power dissipation and fixes history mask bit. Due to the proposed policy, the DHL-cache achieves way-prediction accuracy improvement

through dynamic history length adjustment with minimum hardware support.

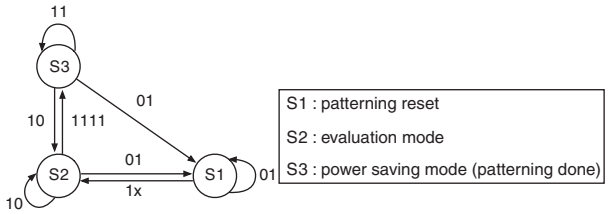

Fig. 3 *Finite state machine diagram of DHL-cache*

Methodologies: Dynamic power-dissipation parameters were measured using the CACTI v6.5 with a 32 nm process. The way-prediction rate of the DHL-cache was measured using a modified Simplescalar v3.0.e. Table 2 provides the cache parameter configurations, which are similar to those used in Intel's Sandy Bridge processor [5].

Table 2: Parameter configurations

Memory latency and power configuration	
Latency of cache hit and prediction hit	12 cycles
Latency of cache hit and prediction miss	25 cycles
Latency of cache miss	43 cycles
L2 cache access energy consumption	0.053 nJ
L2 cache miss penalty energy consumption	0.526 nJ

Simulation result: Fig. 4 shows the number of employed ways and the improvements in the energy-delay product (EDP) of the DHL-cache normalised to the location-cache. The results show that the DHL-cache achieves a 55.3% EDP improvement for an 8-way associative 256 KB L2 cache with a 64B block size. Moreover, the DHL-cache employs 3.5 ways on average. Especially for integer benchmarks, the DHL-cache shows a 69% EDP improvement when 3.86 ways are employed. Because the DHL-cache significantly improves way-prediction accuracy, it could avoid unnecessary prediction miss penalties. Further, as all operations run in the background, no additional delay is required. Therefore, the DHL-cache achieves the high way-prediction rate necessary to maximise power efficiency by selecting several qualified way candidates based on the adaptive history length adjustment algorithm for large L2 cache.

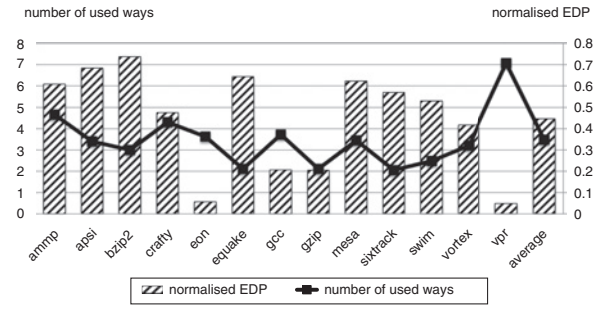

Fig. 4 *EDP improvement and number of used ways compared with the location-cache*

Conclusion: As the market requires high-performance microprocessors even for hand-held devices, the size of L2 cache and the associativity has been increased to meet the demands. Due to high associativity and fragmented locality of L1 cache, way prediction for L2 cache is too difficult to achieve sufficient way-prediction accuracy. The DHL-cache alleviates the power limit issue with the use of the dynamic per history length adjustment algorithm, even though L2 cache has adverse conditions for the way prediction. Furthermore, as the DHL-cache operates without tag information, it requires less than 1% additional area compared with that of a conventional cache memory cell. Therefore, the DHL-cache resolves the power limit issue with minimal performance loss and area overhead.

Acknowledgment: This work was supported by the ICT R&D program of MSIP/IITP [2016 (R7177-16-0233), Development of Application Program Optimization Tools for High Performance Computing Systems].

Submitted: *26 April 2016* E-first: *29 June 2016*
doi: 10.1049/el.2016.1465

H.W. Joo and E.Y. Chung (*School of Electrical and Electronic Engineering, Yonsei University, Seoul, Korea*)

✉ E-mail: eychung@yonsei.ac.kr

References

1 Inoue, K., Ishihara, T., and Murakami, K.: 'Way-predicting set-associative cache for high performance and low energy consumption'. Proc. of ISLPED, San Diego, CA, USA, August 1999, pp. 273–275
2 Min, R., Wen-Ben, J., and Hu, Y.: 'Location cache: A low-power L2 cache system'. Proc. of ISLPED, Newport Beach, CA, USA, August 2004, pp. 120–125
3 Jaleel, A., Theobald, K.B., and Emer, J.: 'High performance cache replacement using re-reference interval prediction (RRIP)'. Proc. 34th Int. Symp. on Computer Architecture (ISCA-37), 2010, San Diego, CA, USA, pp. 95–102
4 Moinuddin, K.: 'Adaptive insertion policies for high performance caching'. Proc. 34th Int. Symp. on Computer Architecture (ISCA-34), 2007, San Diego, CA, USA, pp. 381–391
5 Linley, G.: 'Sandy Bridge spans generations'. Microprocessor Report, Mountain View, CA, USA, 2010