Using Cache to Reduce Power in Content-Addressable Memories (CAMs)

Kostas Pagiamtzis and Ali Sheikholeslami
Department of Electrical and Computer Engineering
University of Toronto, Canada
{pagiamt,ali}@eecg.toronto.edu

***Abstract*— We propose using caching to save power in content-addressable memories (CAMs). By using a small cache along with the CAM, we avoid accessing the larger and higher power CAM. For a cache hit rate of 90%, the cache-CAM (C-CAM) saves 80% power over a conventional CAM, for a cost of 15% increase in silicon area. Even at a low hit rate of 50%, a power savings of 40% is achieved. The proposed C-CAM is employed in the design of a testchip demonstrating a 2.6 fJ/bit/search in a 0.18 µm CMOS process.**

I. Introduction

Content addressable memories simultaneously compare an input word to all the contents of memory and return the address of matching locations. CAMs with large capacities speed up the operation of search-intensive tasks such as packet forwarding and classification in routers, database lookups, and compression [1], [2]. The main challenge in CAM design is to reduce power while maintaining speed and low area.

Fig. 1 depicts the basic CAM circuit structure of a CAM word which is made of CAM cells. A CAM cell compares its stored bit against its corresponding search bit provided on the search-line (SL). The combined search result for the entire word is generated on the match-line (ML). The match-line sense amplifier (MLSA) senses the state of the ML and outputs a full logic swing signal. At the circuit level, research has focussed on saving power by reducing the ML and SL signal swing [3]–[5] or using low-power current-based approaches [6], [7]. Even with these circuit innovations, the simple CAM circuit structure of Fig. 1 has a very high power consumption if it is used directly to construct a large capacity CAM. Fig. 2 shows two previously proposed CAM architectures to reduce power consumption: the pipeline architecture [4] and the block-select architecture [5], [8]. The pipeline architecture saves power by splitting the MLs into several stages, each stage being activated only on the condition that all its previous stages have matched the search data. Since most words miss in early stages this scheme saves power. The block-select architecture divides the CAM into several blocks based on extra data bits (e.g., 2 data bits to select one of 4 blocks). Ideally, only a single block is active thus saving power. Despite the progress of these architecture, the power dissipation of large-capacity CAM is still too high.

This paper proposes a cache-CAM (C-CAM) that can potentially reduce power consumption between 40-80% depending on the cache hit rate. Fig. 3 represents a simplified view of the proposed C-CAM, where the cache holds frequently accessed CAM data. This caching concept is similar to that of caching in processor systems. Unlike processor cache systems where the main purpose of the cache is to increase performance, the purpose of caching CAM is to reduce power consumption. In C-CAM, every search operation that results in a match from the smaller cache saves power by avoiding a search in the larger and higher-power CAM. In the remainder of this paper, we demonstrate how power is saved using our testchip, and discuss the trade-offs that affect the cache hit rate and the overall power dissipation.

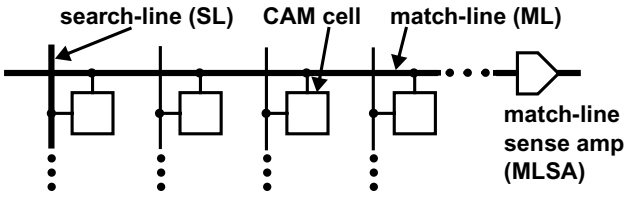


Fig. 1. CAM circuit structure.

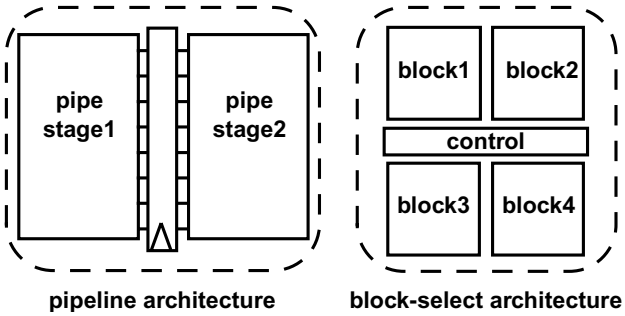


Fig. 2. Block diagrams of pipeline architecture and block-select architecture.

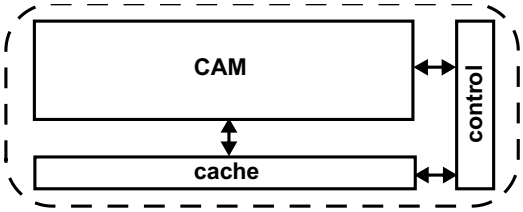


Fig. 3. Simplified block diagrams of our proposed C-CAM.

II. Cache-CAM (C-CAM)

The power savings in a C-CAM implementation depends heavily on the trade-off between the cache size and the cache hit rate. To illustrate this trade-off, Fig. 4 plots the power consumption of a 32k C-CAM system versus the cache size. We plot the portion of power consumed by the cache, the portion consumed by the CAM, and the total power. The

Authorized licensed use limited to: The University of Toronto. Downloaded on January 2, 2010 at 12:24 from IEEE Xplore. Restrictions apply.

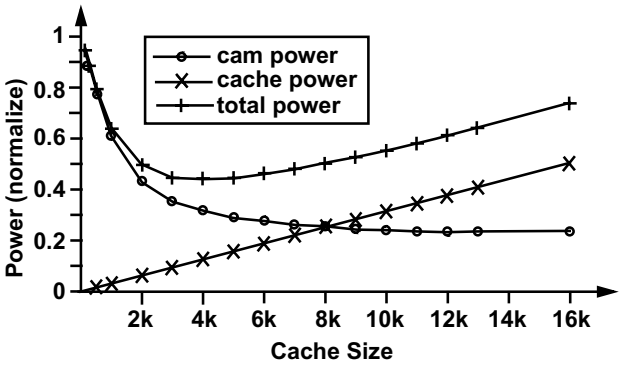


Fig. 4. Power versus cache size for a C-CAM with 32k entries. The power consumption is normalized to that of a conventional CAM with 32k entries.

reported power is normalized to the power dissipation of a conventional 32k CAM. As expected, the cache power consumption increases linearly with the cache size. The CAM power, on the other hand, decreases with the cache size, due to the increased cache hit rate and thus less frequent access to the CAM. The total C-CAM power is at a minimum at about the 4k cache size, and increases for larger cache sizes as the cache hit rate levels off.

The plot of Fig. 4 does not include the overhead power due to the cache replacement policy. The cache replacement policy is one of the factors that determines the cache hit rate. Thus a high-performance policy would reduce the CAM power in Fig. 4 by avoiding misses that search the full CAM. The cost of this reduction in CAM power is the power consumption of the circuitry implementing the cache replacement policy. Therefore we desire a cache-replacement that generates a high cache hit rate while consuming little power. We have incorporated a range of three cache replacement policies in our C-CAM testchip to experimentally determine the trade-offs (see Section IV).

III. Testchip Architecture

For the purpose of testing, we have combined the cache and CAM onto a single die and fabricated a testchip in 0.18 µm CMOS process that uses a 1.8 V nominal supply voltage. Fig. 5 is a block diagram of the testchip architecture. The two main blocks are the CAM, which has 160 entries of 144 bits, and the cache, which has 16 entries of 144 bits. To minimize the number of input pins on the testchip, we use a serial interface to load the CAM data into the CAM BL shift registers at the top of the diagram. We choose the appropriate wordline by serially loading a logic '1' into the WL shift registers. The WL shift registers along with the CAM BL shift registers control the data written into the CAM. We also use a serial interface to load the search data into the cache SL shift registers shown at the bottom of the diagram. The CAM SL registers and cache BL registers are responsible for exchanging data between the CAM and the cache. The operation of these registers will become apparent below in the description of the three-stage pipeline. Just below the cache, in

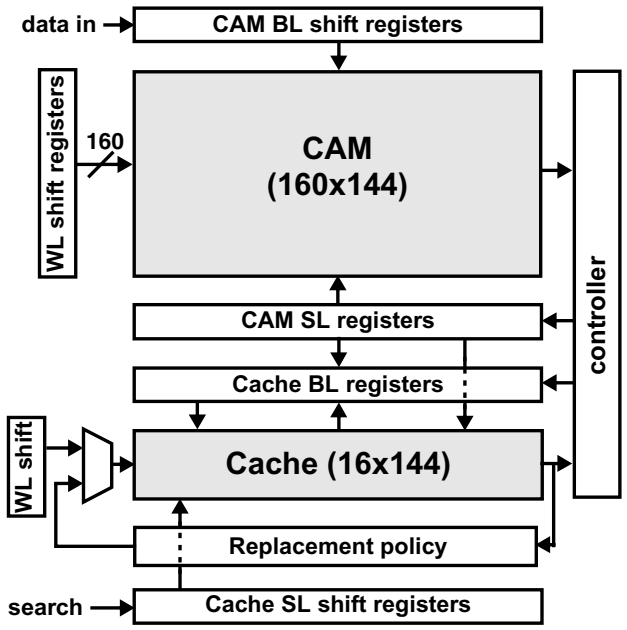


Fig. 5. The testchip incorporates a CAM with 160 entries and a cache with 16 entries, both with 144-bit words. The CAM SL and cache BL registers hold data moving between the CAM and the cache.

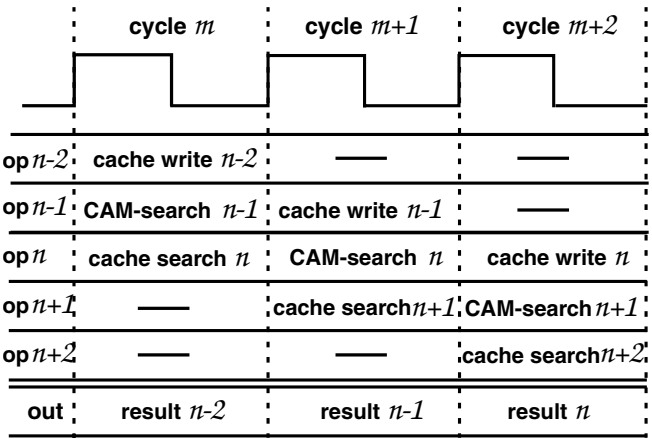


Fig. 6. Timing diagram illustrating the three-stage pipeline operation of the testchip.

the diagram, is the replacement policy block that implements three different replacement policies. The replacement policy block (described in Section IV) feeds cache wordlines, which can be bypassed and controlled by the cache WL shift registers if necessary. The controller on the right of the diagram governs the operation of the overall system by enabling a CAM search operation in the event of a miss in the cache and enabling a CAM write in the case of a successful CAM search.

The testchip operates with a three-stage pipeline as shown in the timing diagram of Fig. 6. Each system search operation takes a horizontal slot (labeled op $n-2$, op $n-1$, etc.), with the bottom slot reserved for indicating when the output is available. From the external point of view, we refer to a

Authorized licensed use limited to: The University of Toronto. Downloaded on January 2, 2010 at 12:24 from IEEE Xplore. Restrictions apply.

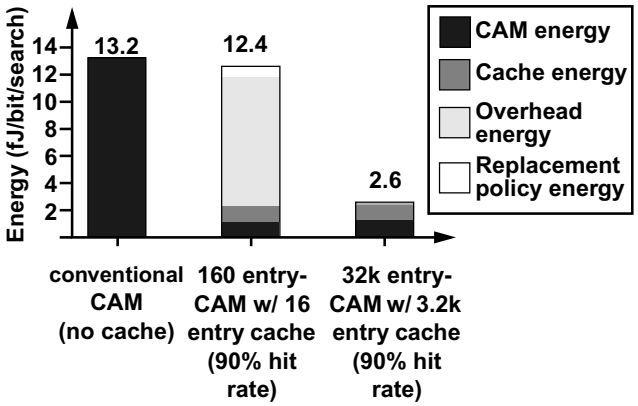


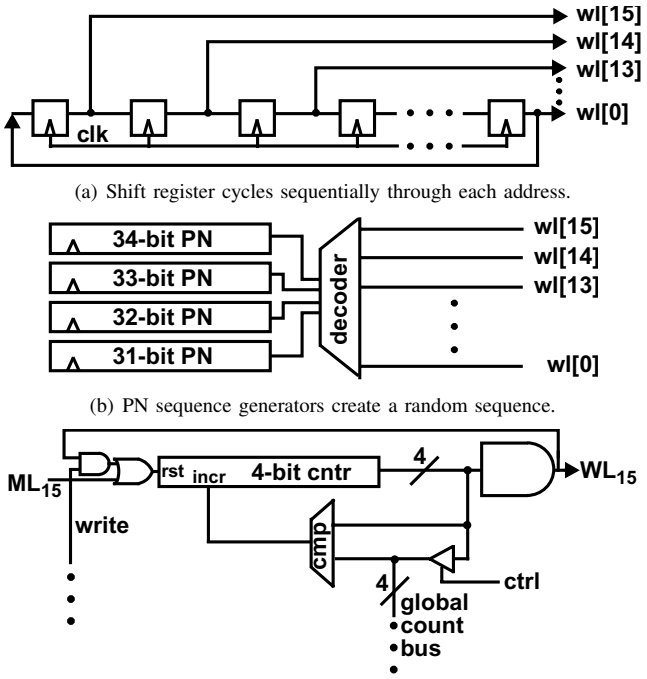
Fig. 7. Graph illustrating the energy savings using the caching technique. Energy is plotted for a conventional CAM, the testchip configuration of a 160 entry CAM with 16 entry cache, and a larger 32k CAM with 3.2k entry cache. Energy is shown for the case of a 90% hit rate.

search as a system search. A system search may be internally composed of a cache search resulting in a match or a cache search resulting in a miss followed by a CAM search resulting in a match. In the timing diagram, we assume all system searches cause a cache miss followed by match in the CAM. At op n, for example, a system search takes three cycles to complete (cycles n, $n+1$, $n+2$). In the first cycle (cycle m), the search word is applied to the cache and is labeled cache search n in the diagram. Since we assume a cache miss in cycle m, in the next cycle the search word is loaded into the CAM SL registers (of Fig. 5) and applied to the CAM. This operation is labeled CAM-search n in the timing diagram. Since the CAM search results in a match, in cycle $m+2$ the search result is both available externally and written back into the cache. We see that in cycle $m+2$, the cache is accessed for both a write by op n and a search by op $n+2$. By supporting a cache search and write in a single cycle, we keep the system search latency to only three cycles.

Fig. 7 compares the simulated energy of a C-CAM against that of a conventional CAM. We plot the energy in fJ/bit/search for a conventional CAM, the implemented testchip, and for a larger CAM with 32k entries of 144 bits using a cache with 3.2k entries. Both cached systems assume a hit rate of 90%. The energy savings for the testchip configuration with the 160 entry CAM is 6%. The savings is small because of the relatively large overhead due to the CAM SL registers and cache BL registers and writes into the cache (called overhead energy in the graph) and the overhead due to the replacement policy. For the case of the larger 32k system, the overhead energy and replacement policy energy do not increase significantly with the size of the system, so that they are amortized over a larger number of bits. Thus the savings in power is 80% over the conventional CAM.

IV. Cache Replacement Implementations

The cache replacement policy decides which cache entry should be removed when a new entry is inserted into a full



(a) Shift register cycles sequentially through each address.

(b) PN sequence generators create a random sequence.

(c) A count at each cache location determines the least recently used location.

Fig. 8. Cache replacement policy circuits for (a) sequential replacement, (b) random replacement, and (c) LRU replacement.

cache. The replacement policy and characteristics of the input data stream together determine the resulting cache hit rate for a given cache size. A low-complexity replacement policy may consume little power; however, it results in a low cache hit rate leading to high overall system power dissipation. On the other hand, a high-complexity replacement policy may consume so much power that it overwhelms the savings resulting from the high cache hit rate.

To explore this spectrum of replacement policies, we include three cache replacement policies on the testchip: sequential replacement, random replacement, and least-recently used (LRU) replacement. The sequential replacement policy simply increments an address counter sequentially to point to the location of the new entry. We have implemented the sequential replacement policy with a shift register, as shown in Fig. 8(a). This implementation consumes little power but results in a low hit rate. The random replacement policy chooses a random cache location for any new cache entry. The implementation of the random replacement policy consists of four pseudo-random number (PN) sequence generators and a decoder, as shown in Fig. 8(b). This implementation consumes slightly more power than the sequential policy but generally results in an increased hit rate. Finally, the LRU policy of Fig. 8(c) tracks the relative time when each cache location receives a hit and replaces the location that is least recently used. This policy is implemented in the elementary form described in [9] and has the highest complexity and power consumption compared to the other two replacement policies, but offers a moderately higher cache hit

Authorized licensed use limited to: The University of Toronto. Downloaded on January 2, 2010 at 12:24 from IEEE Xplore. Restrictions apply.

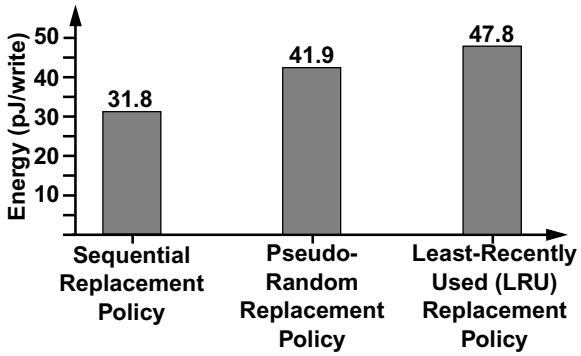


Fig. 9. The energy per cache write used by the three cache replacement policies implemented on the testchip.

rate than the random policy.

Fig. 9 plots the simulated energy consumption for the three cache replacement policies implemented on the testchip. The power is reported in units of pJ/write. After the cache is initially filled, the cache replacement policies are active only when there is a write to the cache. Since a cache replacement occurs on every cache miss, the portion of cycles where a cache write occurs is equal to the cache miss rate. The LRU policy actually consumes power even during cache search cycles (i.e. including cycles that do not have a cache write) since LRU counters are updated even on cache searches. Thus, for the LRU policy, there is an extra power consumption that occurs on every search in addition to the power consumption per write. However, this power consumption is small compared to the other sources of overhead that are present on every cycle (due to the CAM SL and cache BL registers).

V. Testchip Measurement

Fig. 10 is an annotated photomicrograph of the fabricated testchip. To save silicon area, the testchip includes only 160 entries for CAM and 16 entries for cache. To measure power contributions of each block separately, the testchip employs a total of five separate power supplies (one for each of CAM, cache, overhead, replacement policy, and test circuitry). The replacement policy block contains three different cache replacement policies each of which can be independently enabled to measure their power consumption. Our early measurement results indicate that the CAM portion of the power is 10 times the cache portion, as expected. Also, the power consumption of the LRU replacement policy is the highest among the three policies implemented on the testchip. Further power measurements are required to obtain the power proportions of other components in this particular design.

VI. Conclusion

This paper proposes C-CAM which uses caching to save power in CAM. Through simulation and early testchip measurements, we show that a cache added to a 32k CAM can save as much as 80% of power consumption for hit rates of 90%. Even for hit rates as low as 50% the power savings are still 40%.

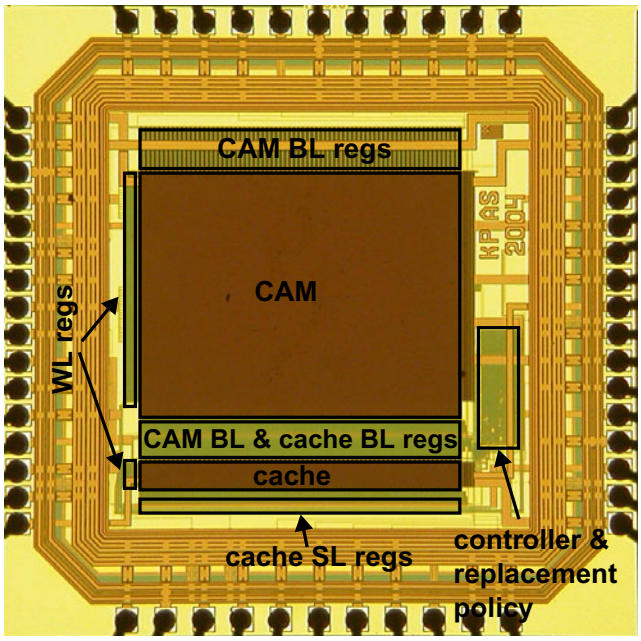


Fig. 10. Photomicrograph of testchip fabricated in 0.18 µm CMOS process.

Acknowledgment

We thank Oleksiy Tyshchenko for aiding in chip verification. Also, we gratefully acknowledge financial support from the Natural Sciences and Engineering Research Council of Canada (NSERC) and an Ontario Graduate Scholarship in Science and Technology (OGSST) and resource for testchip design, fabrication, and testing from the Canadian Microelectronics Corporation (CMC).

References

[1] T.-B. Pei and C. Zukowski, "Putting routing tables in silicon," *IEEE Network Magazine*, vol. 6, no. 1, pp. 42–50, January 1992.

[2] L. Chisvin and R. J. Duckworth, "Content-addressable and associative memory: Alternatives to the ubiquitous RAM," *IEEE Computer*, vol. 22, no. 7, pp. 51–64, July 1989.

[3] H. Miyatake, M. Tanaka, and Y. Mori, "A design for high-speed low-power CMOS fully parallel content-addressable memory macros," *IEEE J. Solid-State Circuits*, vol. 36, no. 6, pp. 956–968, June 2001.

[4] K. Pagiamtzis and A. Sheikholeslami, "A low-power content-addressable memory (CAM) using pipelined hierarchical search scheme," *IEEE J. Solid-State Circuits*, vol. 39, no. 9, pp. 1512–1519, September 2004.

[5] G. Kasai, Y. Takarabe, K. Furumi, and M. Yoneda, "200MHz/200MSPS 3.2W at 1.5V Vdd, 9.4Mbits ternary CAM with new charge injection match detect circuits and bank selection scheme," in *Proceedings of the IEEE Custom Integrated Circuits Conference*, 2003, pp. 387–390.

[6] I. Arsovski, T. Chandler, and A. Sheikholeslami, "A ternary content-addressable memory (TCAM) based on 4T static storage and including a current-race sensing scheme," *IEEE J. Solid-State Circuits*, vol. 38, no. 1, pp. 155–158, January 2003.

[7] I. Arsovski and A. Sheikholeslami, "A mismatch-dependent power allocation technique for match-line sensing in content-addressable memories," *IEEE J. Solid-State Circuits*, vol. 38, no. 11, pp. 1958–1966, November 2003.

[8] F. Zane, G. Narlikar, and A. Basu, "CoolCAMs: Power-efficient TCAMs for forwarding engines," in *IEEE INFOCOM*, vol. 1, 2003, pp. 42–52.

[9] V. C. Hamacher, Z. G. Vranesic, and S. G. Zaky, *Computer Organization*, 4th ed. Toronto: McGraw-Hill, 1996.

Authorized licensed use limited to: The University of Toronto. Downloaded on January 2, 2010 at 12:24 from IEEE Xplore. Restrictions apply.