Leakage Energy Reduction in Banked Content Addressable Memories

Pedro Echeverría, José L. Ayala, Marisa López-Vallejo
Departamento de Ingeniería Electrónica
Universidad Politécnica de Madrid (Spain)
Email: {petxebe,jayala,marisa}@die.upm.es

Abstract— **The content-based access of CAMs makes them of great interest in look-up based operations. However, the large amounts of parallel comparisons required cause an expensive cost in power dissipation. In this work we present a novel banked pre-computation based architecture for low-power and storage-demanding applications where the reduction of both dynamic and leakage power consumption is addressed. Experimental results show that the proposed banked architecture reduces up to an 89% of dynamic power consumption during the search process while the leakage power consumption is also minimized by a 90%. The active area is decreased in a 10% while performance is also improved by a 70%.**

I. Introduction

A broad range of modern applications demands large storage devices with fast response. Content-Addressable Memories (CAMs) have emerged as one of the favorite devices for such applications [1]–[4]. In CAMs data are accessed based on their content, rather than their physical address. This functionality has shown to be specially efficient in lookup-based applications like TLBs [1], associative computing and data compression [3]. High-speed networks such as gigabit Ethernet and ATM switches also benefit from this particular structure [5].

However, CAMs pay a high hardware price for this content-based access because the memory cell must include comparison circuitry, negatively impacting the size/speed trade-off and complexity of the implementation. Usually, a 9-transistors cell is required instead of the 6-transistors cell used in SRAM. Moreover, the large amounts of parallel comparisons performed in conventional CAMs make the device to consume too much power, preventing the implementation of large scale CAMs in a single chip as the leading-edge applications demand.

In this paper we present a novel implementation of a CAM with low-power constraints. The proposed architecture is highly scalable and provides high-performance functioning at large sizes. Moreover, it achieves great savings for both dynamic and leakage power consumption. Therefore, this architecture overcomes previous limitations of the CAM implementation and makes it suitable to all the applications where a high-performance low-power data search functioning is needed.

Previous work on CAM design has focused only on either reducing the power consumption of the match line [6] or enhancing the search speed [7]. Nevertheless, both goals have not been addressed simultaneously in previous approaches.

Although many approaches addressing dynamic power dissipation have been reported [8]–[10], resulting circuit techniques have either substantial area overhead, deficiencies in noise immunity, or cannot be easily scaled without a negative impact on performance. Our work overcomes these limitations by a novel and effective design of the CAM architecture that also addresses leakage energy reduction in an efficient way.

The work presented here improves the energy savings obtained by Li et al. in [11], and recently extended by Noda [12] and Choi [10]. These recent works provide a low power implementation of the CAM based on the precomputation of an index parameter. Nevertheless, they are constrained to specific small sizes, lack of scalability and present an increased search delay. Moreover, the leakage energy consumption (which is one of the main issues regarding modern electronic design) has not been addressed by these approaches. Our work increases the dynamic power savings of these approaches and, also, reduces the leakage energy consumption of the memory to a minimum.

Our work is also based on a parameter precomputation-based architecture [11] (PB-CAM from now on); however, we are able to reduce the parameter word's size with respect to [11], decreasing in this way the logic complexity, area and power consumption related to this parameter. Moreover, the energy savings obtained with the proposed banked architecture (up to 88% of the dynamic power consumption and a 90% of the leakage power consumption) improve the previous implementations of similar technologies and also improve the scalability capabilities of architectures like [12] and [10].

Our research work on this field has already shown good results in terms of area and dynamic power consumption [13], [14]. This paper presents an improved architecture with a novel hardware mechanism to reduce the static power consumption and increase the dynamic energy savings.

The paper is composed as follows. Section II presents the first designed approach where the minimization of the dynamic power consumption is addressed, while the leakage power reduction technique is showed in section III. The experimental results are summarized in section IV. Finally, some conclusions are drawn.

1-4244-0395-2/06/$20.00 ©2006 IEEE.

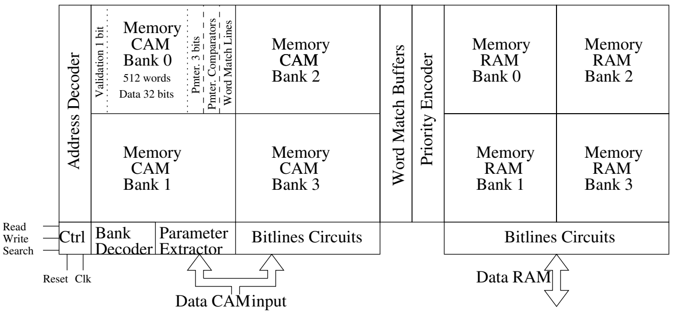


Fig. 1. Banked architecture (4 banks implementation).

II. Devised Architecture: Banked Approach

The first approach presented in this section describes a banked implementation of the mentioned PB-CAM [11] in order to reduce the dynamic power consumption during the search operation. The main idea of the PB-CAM is to use a parameter word (obtained by a formula) to perform the comparison process in a reduced number of memory positions, saving dynamic power consumption. However, the total power consumption of the logic can still be too high for low-power applications. Our architecture employs the pre-computed parameter to perform a power-aware ordering of the data. The parameter extracted allows us the classification of the memory contents based on the one's count. This classification can be used to store the memory contents in such an efficient way that the search operation is restricted to a smaller memory size.

The order of the memory data attending to the one's count parameter allows to split the memory architecture into independent banks. Every data in the bank has the same value for one subset of the parameter (ex. *N-least significant bits*). The logic needed for this ordering is very simple and does not present a serious overhead in terms of delay and energy consumption [14].

Due to the banked implementation of the memory, the operation of the architecture is restricted to just one bank every cycle (see Figure 1, where the RAM area corresponds to the memory where the output addresses associated to each tag will be found to complete a CAM-search engine). One of the advantages of this banked structure is the reduction of the dynamic power consumption due to the reduced charge in the bit lines (the driven line is simplified to the bit line of just one bank of the memory). This behavior is also shown by the parameter lines and also has a positive influence in the memory speed. The complexity of the logic shared for the banks (buffers, priority encoders and address decoders) is reduced when the bank approach is applied. This simplification saves area, power consumption and improves the delay of these devices.

III. Devised Architecture: Drowsy Approach

The devised architecture presented in section II has been improved to reduce the static power consumption of the architecture. This approach turns the unused memory banks into a low-power state and implements a pipeline which carefully manages the operations to avoid any performance penalty. This approach is based on the design of a low-power cell and a further pipelined sub-banked implementation.

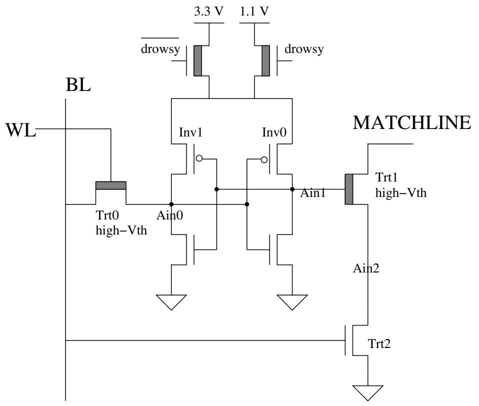


Fig. 2. 7-transistor CAM cell with drowsy support.

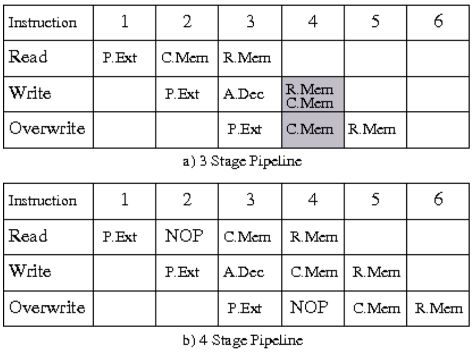
A. Low-Power Cell

As has been previously described, the memory area has been split into several banks, which can be independently accessed by the datapath. Then, a Dynamic Voltage Scaling (DVS) technique is applied to turn the unused banks into a low power state and thus save as much energy as possible in the system. However, when the objective is a memory device, the cost of recovering the lost information could hide any power saving or, at least, represent a very significant time penalty. Moreover, something has to be done in the powered-down banks to prevent the information from being lost.

An efficient approach to achieve the drowsy state is proposed by [15], where a DVS technique is exploited to reduce static power consumption. In fact, due to short-channel effects in deep-submicron processes, leakage current is significantly reduced with voltage scaling. Thus, the combined effect of reduced leakage current and voltage yields a drastic reduction in leakage power. This is the solution used in our approach to reduce energy consumption.

The method proposed by Flautner et al. utilizes DVS to reduce the leakage power of cache cells. By scaling the voltage of the cell to approximately 1.5 times V_{th}, the state of the memory cell can be maintained. Figure 2 shows the modified memory cell used to support the drowsy state. As can be observed, the dual power supply is switched to low V_{DD} when the cell is in drowsy state. It is necessary to use $high - V_{th}$ devices as pass transistors because the voltage on bit lines could destroy the cell contents. Before a memory position can be accessed, the power supply has to be switched to high V_{DD} to restore the contents and allow the access. The careful management of these operations along the pipeline presented in the next section takes care of this extra clock cycle to avoid the performance penalty.

Each bank of the CAM architecture counts with the additional logic required to implement the DVS state. Since the low power consumption state is selected for the whole bank instead of a specific memory position, the overhead of the control logic is greatly minimized.



Instruction	1	2	3	4	5	6
Read	P.Ext	C.Mem	R.Mem			
Write		P.Ext	A.Dec	R.Mem C.Mem		
Overwrite			P.Ext	C.Mem	R.Mem	

a) 3 Stage Pipeline

Instruction	1	2	3	4	5	6
Read	P.Ext	NOP	C.Mem	R.Mem		
Write		P.Ext	A.Dec	C.Mem	R.Mem	
Overwrite			P.Ext	NOP	C.Mem	R.Mem

b) 4 Stage Pipeline

Fig. 3. Structure of the proposed pipeline.

B. Pipeline

A clear way to improve the access time of a CAM is the use of a pipeline structure, which additionally promises greater scalability in the performance and density of the applications that make use of CAMs. The aforementioned DVS technique can take advantage of this pipeline to awake the drowsy cells one clock cycle before the access. Only one of the banks needs to be on while the rest can remain in the drowsy state saving leakage energy. In our approach the devised pipeline configuration includes the following operations:

- **READ operation:** EXT - SEARCH - READ_R
- **OVERWRITE operation:** EXT - SEARCH - WRITE_R
- **WRITE operation:** EXT - DEC - WRITE_CR

where READ is the read operation in the associated RAM memory after the tag is found in the CAM, OVERWRITE is the search and write operation in the RAM memory, and WRITE is the operation to write a tag and its data in both CAM and RAM memories. The pipeline stages defined within those operations are EXT (parameter extraction), SEARCH (data comparison in the CAM), DEC (decodification of internal address, common for both RAM and CAM), READ and WRITE.

However, this three stages pipeline shows a structural and data hazard, as depicted in Figure 3.a[1]. This hazard is produced in the CAM structure between the READ (or OVERWRITE) operation and the WRITE operation because the CAM area is simultaneously accessed by the second and third stages, respectively. This problem can be solved by including a fourth pipeline stage splitting the WRITE operation into WRITE_C and WRITE_R (see Figure 3.b). All the CAM accesses are in the third stage and the RAM accesses in the fourth.

- **READ:** EXT - NOP - SEARCH - READ_R
- **OVERWRITE:** EXT - NOP - SEARCH - WRITE_R
- **WRITE:** EXT - DEC - WRITE_C- WRITE_R

The second stage would mean a stall in the Read and Overwrite operations but now is used to wake up the memory cells from the drowsy state. Therefore, the pipeline is not stalled and the performance is not compromised.

C. Banks Subdivision

Once the leakage current control mechanism has been exposed, the natural goal consist on increasing the expected energy savings. The simplest idea is to divide the memory in as many banks as possible, using more parameter bits to decode the active bank. This technique presents the same advantages as those mentioned in the section II.

However, dividing the memory in so many banks has two very important drawbacks. Firstly, the unbalanced used of the banks. The 4-banks implementation in section II presents a homogeneous use of the banks (each bank with a 25% of the input tags) but if a third bit is introduced to split the CAM into 8 banks, the distribution of inputs varies from 14.48% to 10.52%[2] for the one's count parameter. And secondly, the complex layout that will require the memory, due to the common elements of the banks.

Therefore, another technique has been devised to preserve the homogeneous use of the banks and a realistic layout: the subdivision of each bank into a set of subbanks. The main idea is to combine the parameter decoding with a new ordering of the input tags, using in this case the value of some bits of the input tags. In this way, the tags found in the same bank are ordered in local subbanks attending to some bits (the tags belonging to the same subbank share the same value of some bits). This mechanism obtains a very homogeneous use of the subbanks without impacting the layout.

For example, in the previous 4-bank implementation, using any two bits of the tag to enable the bank subdivision, there will be 4 subbanks per bank (those banks correspond to the values 00, 01, 10 and 11 of any two tag bits). Unlike the 8-bank implementation, this 4-bank configuration with subbanking presents a very homogeneous use of the subbanks, with only 0.013% of maximal difference between subbanks. Moreover, the obtained layout remains without appreciable changes when the subbanking approach is applied.

One of the key advantages of this subbanking technique is that any memory operation will be done only in the proper subbank of the decoded bank, while the other subbanks of that bank as well as the other banks will remain at the drowsy mode. Moreover, there are also dynamic power and area saving advantages of this technique very similar to the ones presented for the bank implementation. For example, the tag bits used for the subbanking do not need to be stored. Also, the complexity of the common logic (address decoder and priority encoders) can be simplified by designing a single element for the subbank and sharing this design for every subbank in the architecture. And finally, the power consumption of the comparison operation is restricted to the working subbank, which increases the savings in this factor.

IV. Experimental Results

The experiments have been carried out with Spice simulations in the Cadence environment. The technology used is .35 μm from Austria MicroSystems.

[1] Resources (parameter extractor, address decoder, CAM and RAM memories) and data are shown in the plot.

[2] That is a difference of 27.4% between the most and the least used bank.

A. Banked Implementation

The banked architecture has been firstly evaluated in terms of the energy savings obtained. The simulated memory is the architecture described in Figure 1, implemented as a memory of 2048 positions and 32 bits per word, and split into 4 independent banks. Our approach decreases the energy consumption by a 76% (20.68 fJ/bit in the banked architecture with respect to 86 fJ/bit in the referenced [11]).

The area improvement achieved with the proposed architecture has also been evaluated. There is a reduced area improvement of the banked implementation with respect to the original PB-CAM, and almost constant when the number of bits per word is shifted. This difference is due to the savings in the parameter word length and comparison logic and reaches a 18% for an implementation with 128 bits per word. Also, the area savings obtained with the banked PB-CAM become more notorious for larger implementations (10% for an implementation with a 2048-bit parameter architecture) because the overhead incurred by the parameter word is lower than in the PB-CAM.

Finally, the performance of the design has been analyzed to assure the required fast response. The results show a 7.5 ns delay for the search operation, which also includes the data write into a RAM memory. The comparison of these performance results with those described by Lin in [11] shows how the delay occurred by the banked architecture is a 25% lower than the original PB-CAM (10 ns).

B. Drowsy Implementation

In terms of the energy savings obtained, this approach presents further improvements both in dynamic and leakage energy due to the subbanking and drowsy techniques. The simulated architecture is a CAM with 8192 positions[3] and 32 bits per word, implemented with 4 independent banks, 4 subbanks per bank, and the described 4-stage pipeline.

With this design, the dynamic consumption is reduced to 10.08 fJ/bit (a decrease of an 89% and 56% respect to the baseline architecture and our first approach, respectively). The leakage power consumption is reduced in a 90% because most of the memory remains in the drowsy state.

The throughput of the system is also improved (70% when compared with the baseline architecture, 60% when compared with the banked PB-CAM). The resulting area does not show significant differences with the banked approach since the area overhead of the $high - V_{th}$ transistors is minimal. A comparison of the two approaches presented in this paper, as well as the baseline architecture, can be found in table I.

TABLE I
COMPARISON OF THE THREE APPROACHES (SIMPLE, BANKED AND DROWSY PB-CAM).

	PB-CAM	B. PB-CAM	D. PB-CAM
Technology	$0.35\mu m$	$0.35\mu m$	$0.35\mu m$
CAM configuration	128 x 30	2048 x 32	8192 x 32
Power performance	86 fJ/bit	18.86 fJ/bit	9.06 fJ/bit
Operation delay	10 ns	7.5ns	12 ns
Throughput	100 MIPS	133 MIPS	333 MIPS

V. CONCLUSIONS

Nowadays, the limiting factor in applications where the CAMs play a critical role is the power consumption of these devices. The integration levels achieved by current technology processes have turned the area and performance factors into secondary actors. Search-based applications with high-performance constrains demand efficient implementations of content-addressable memories to cover the astringent constrains. The work presented in this paper has shown two efficient mechanisms to reduce the dynamic and static power consumption in the architecture by means of hardware modifications. These approaches do not compromise the performance and area improvements achieved with the architecture.

ACKNOWLEDGEMENTS

This work was supported by the Spanish Ministry of Science and Education under contract TIC2003-07036.

REFERENCES

[1] S. Swaminathan, S. B. Patel, J. Dieffenderfer, and J. Silberman, "Reducing Power Consumption during TLB Lookups in a PowerPC™Embedded Processor," in *International Symposium on Quality of Electronic Design*, 2005.

[2] S. Stas, "Associative Processing with CAMs," in *Northcon*, 1993.

[3] K. J. Lin and C. W. Wu, "A Low-Power CAM Design for LZ Data Compression," *IEEE Trans. on Computers*, vol. 49, no. 10, pp. 1139–1145, October 2000.

[4] Y. Tang *et al.*, "CAM-Based Label Search Engine for MPLS over ATM Networks," in *IEEE GLOBECOM*, 2001.

[5] H. Liu, "Reducing Routing Table Size Using Ternary-CAM," in *Symposium on High Performance Interconnects*, 2001.

[6] I. Arsovski and A. Sheikholeslami, "A current-saving match-line sensing scheme for content-addressable memories," in *IEEE Int. Solid-State Circuits Conf.*, 2003.

[7] F. Shafai *et al.*, "Fully parallel 30-MHz, 2.5-Mb CAM," *IEEE Journal of Solid-State Circuits*, vol. 33, no. 11, pp. 1690–1696, November 1998.

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

[9] K. Pagiamtzis and A. Sheikholeslami, "Pipelined match-lines and hierarchical search-lines for low-power content-addressable memories," in *IEEE CICC*, 2003.

[10] S. Choi, K. Sohn, and H.-J. Yoo, "A 0.7-fJ/Bit/Search 2.2-ns Search Time Hybrid-Type TCAM Architecture," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 1, pp. 254–260, January 2005.

[11] C.-S. Lin, J.-C. Chang, and B.-D. Liu, "A Low-Power Precomputation-Based Fully Paralel Content-Addressable Memory," *IEEE Journal of Solid-State Circuits*, vol. 38, no. 4, pp. 654–662, April 2003.

[12] H. Noda, K. Inoue, and M. Kuroiwa, "A Cost-Efficient High-Performance Dynamic TCAM With Pipelined Hierarchical Searching and Shift Redundancy Architecture," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 1, pp. 245–253, January 2005.

[13] P. Echeverría, J. L. Ayala, and M. López-Vallejo, "A Low-Power Pipelined CAM for High-Performance IP Routing," in *IEEE ICCDCS*, 2006.

[14] ——, "A Banked Precomputation-Based CAM Architecture for Low-Power Storage-Demanding Applications," in *IEEE MELECON*, 2006.

[15] K. Flautner, N. S. Kim, S. Martin, D. Blaauw, and T. N. Mudge, "Drowsy caches: Simple techniques for reducing leakage power," in *Proceedings of ISCA*, Alaska, USA, 2002, pp. 148–157.

[3] Notice the larger implementation with respect to the baseline architecture.