Open access • Proceedings Article • DOI:10.1109/CICC.2006.320887

A Soft-Error Tolerant Content-Addressable Memory (CAM) Using An Error-Correcting-Match Scheme — Source link

K. Pagiamtzis, N. Azizi, Farid N. Najm

Institutions: University of Toronto

Published on: 01 Sep 2006 - Custom Integrated Circuits Conference

Topics: Soft error, Hamming distance and Content-addressable memory

A Soft-Error Tolerant Content-Addressable Memory (CAM) Using An Error-Correcting-Match Scheme

Kostas Pagiamtzis, Navid Azizi, and Farid N. Najm
Department of Electrical and Computer Engineering
University of Toronto, Toronto, Ontario, Canada
{pagiamt, nazizi, najm}@eecg.toronto.edu

Abstract— **Modern integrated circuits require careful attention to the soft-error rate (SER) resulting from bit upsets, which are normally caused by alpha particle or neutron hits. These events, also referred to as single-event upsets (SEUs), will become more problematic in future technologies. This paper presents a binary content-addressable memory (CAM) design with high immunity to SEUs. Conventionally, error-correcting codes (ECC) have been used in SRAMs to address this issue, but these techniques are not immediately applicable to CAMs because they depend on processing the full contents of the memory word outside the array, which is not possible in a normal CAM access. The proposed design consists of a new matching technique that uses coding to increase the Hamming distance between words, in conjunction with a modified matchline sensing scheme. The result is a CAM design that reduces the SER with no increase in delay or power dissipation, and with only a 12% increase in area.**

I. Introduction

Content-addressable memories (CAMs) are SRAM memories enhanced with comparison transistors that enable searching a word across all memory contents in a single clock cycle [1]. A CAM returns the location of the input word, effectively performing a table lookup operation. The table lookup operation speeds up a variety of lookup-intensive applications, but the most pervasive use of CAM today is in routers for the purposes of packet forwarding and classification [2].

The bit storage in CAMs uses SRAM cells, which are susceptible to soft errors caused primarily by alpha-particle and neutron radiation [3]. In an SRAM memory, the soft-error rate (SER) is reduced to an acceptable level by using error-control coding (ECC). Redundant bits are added to each memory word that are used by error-correction circuitry to correct any bit errors during the read operation. ECC techniques are not immediately applicable to CAMs because they typically depend on processing the full contents of the memory word outside the array. This is not possible in a normal CAM access as all memory words are searched simultaneously.

One method for avoiding soft errors in CAMs is to use DRAM cells, which have high soft-error immunity, as the storage instead of SRAM cells [4]. Using DRAM cells, however, results in increased design complexity and fabrication costs. Another method for reducing the SER in CAM implements an embedded DRAM (eDRAM) block alongside an SRAM-based CAM [5]; the eDRAM block, which includes ECC circuitry, is used to continuously write correct data into the CAM. Thus, in the worst case, any soft error in the CAM is overwritten in the amount of time it takes to refresh all entries in the CAM. One problem with this approach is that single-event upsets (SEUs) that happen before a word is overwritten lead to incorrect operation in the CAM. Furthermore the additional eDRAM block has a high area overhead.

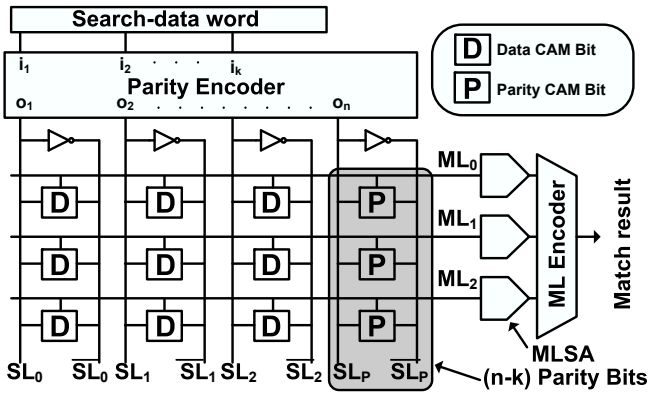

Fig. 1. Simplified block diagram of the proposed CAM using error-correcting match with three words. Each word has three data bits and one parity bit.

Fig. 1 shows a simplified block diagram of our proposed error-correcting-match scheme that tolerates soft errors in CAMs. The input search-data word is fed into a parity encoder that outputs the search data along with generated parity bits as the search codeword into the CAM. To tolerate single-bit errors, we modify the matchline sense amplifier (MLSA) so that words that either match exactly or have a single-bit miss constitute a successful search, while all other cases constitute an unsuccessful search. The new scheme reduces the SER with no increase in delay or power; the main cost of our scheme is due to the increased silicon area for the parity bits.

The remainder of this paper proceeds as follows. Section II reviews CAM basics and describes the conventional current-race matchline (ML) scheme. Section III describes the error-correcting code used and how we maintain correct operation in a CAM in the presence of SEUs using an error-correcting ML scheme. Section IV presents simulation results in a 0.18 µm CMOS process that verify the error-correcting match scheme operates correctly. Section V discusses the extension of this scheme to allow for correct operation under multiple bit upsets and Section VI concludes the paper.

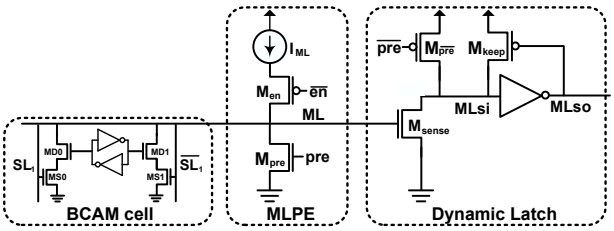

Fig. 2. Schematic of current-race matchline sensing scheme [6].

II. Background

We describe the operation of a CAM using Fig. 1. A CAM search operation consists of the three phases of data broadcast, word comparison, and ML encoding. The data broadcast phase consists of driving the input search-data word onto the complementary searchlines (SLs) labeled SL_i, and $\overline{\mathrm{SL}}_i$. In the word comparison phase, the broadcast search-data word is compared to each stored word in parallel and the results appear on the matchlines (MLs) labeled ML_i. A binary CAM (BCAM) cell, which uses a single SRAM cell for storage, can hold either a logic 0 or 1. During the comparison operation, each CAM cell compares its SL bit to the stored bit. If at least one cell in a word has a mismatch (or miss) between its SL bit and its stored bit, there will be a path from the ML to ground (CAM cell details shown in Fig. 2). On the other hand, if all cells in a word match, there is no path from the ML to ground. The MLSA detects the state of the matchline (match or miss) and outputs a logic high for a match and a logic low for a miss. Finally, the ML encoder maps the MLSA outputs to a binary-encoded match result. Overall, a CAM implements the function of a table-lookup in a single clock cycle operation.

Fig. 2 shows a typical MLSA using the current-race scheme [6]. The MLSA consists of two circuits: the ML precharge and evaluate (MLPE) circuit and a dynamic latch. The ML is the input to the circuit, and and the output is the ML sense output (MLso). The scheme operates by first asserting the pre signal to pre-discharge the ML to ground and to precharge the ML sense input, MLsi, to the supply voltage. Once the precharge is complete, the $\overline{\mathsf{en}}$ signal is asserted, connecting the current source, I_{ML}, to the ML. In the case of a match, there is no path from ML to ground, so the ML voltage increases linearly until the current source is shut off. In the case of a miss, there is at least one CAM cell path from the ML to ground, and therefore the matchline charges to a final voltage of $I_{ML}R$, where R is the resistance of the pulldown path (R varies with the number of bits that miss). The MLSA detects the difference between the match case and the miss cases with threshold, V_{tn}, the threshold voltage of of the NMOS transistor $\mathsf{M}_{\mathsf{sense}}$. If the ML traverses past the V_{tn} trip point, then MLsi is pulled down and the output of the half-latch flips to a logic high, indicating a match. The miss cases do not cross the V_{tn} threshold and thus leave the half-latch in the precharge state of a miss.

III. Error-Correcting-Match Scheme

Our proposed error-correcting-match scheme makes two main modifications to the CAM search. First, we add parity bits to each CAM word, as shown in Fig. 1, in order to increase the minimum Hamming distance between words. Hamming distance is defined as the number of bit locations that differ between two words and the minimum Hamming distance of a code determines how many errors the code is able to correct. Second, we modify the current-race matchline sensing scheme so that both a match and a one-bit miss constitute a successful search and all other cases constitute an unsuccessful search. Since we redefine the nature of a match for our proposed scheme, we use the term *successful search* for the remainder of the paper to define the cases where the input word matches the stored word, and the term *unsuccessful search* to define the cases where the input word mismatches the stored word.

A. Coding Scheme

To increase the Hamming distance between two words, each stored word, which is composed of 72 CAM cells is augmented with nine extra CAM cells to store the parity of a $(81, 72, 4)$ code. The notation (n, k, d) defines n as the total number of bits, k as the number of information bits (and thus $n - k$ is the number of parity bits), and d is the minimum distance.. By obtaining a code with a distance of four, words that differ by one bit will have stored codewords that differ by four bits. This difference allows matches and misses to be distinguished even in the presence of soft errors because a matching word with a 1-bit upset will become a 1-bit miss in the worst case; a 1-bit miss is a successful search in our scheme. Similarly, a mismatched word with a 1-bit upset will result in a 3-bit miss in the worst case. A 3-bit miss is a unsuccessful search in our scheme. Table I shows the results of an SEU in the bits of the CAM word. The table shows that even in the presence of a single-bit error, the resulting ML state still appropriately corresponds to a successful search (match or 1-bit miss) or an unsuccessful search (3-bit miss or 4-bit miss).

B. Encoder Design

The $(81, 72, 4)$ code is obtained by shortening the $(256, 247, 4)$ extended Hamming code [7]. Since the complexity of a parallel implementation of an encoder is determined by number of 1's in its generator matrix [8], we selectively shorten the extended Hamming code by removing the rows with the largest number of 1's (largest weight). Then to reduce the worst case delay of the encoder, row operations on the new generator matrix reduce the maximum column weight in the parity portion of the generator matrix. These optimizations

CAM Content	SRAM Bit Before SEU	SRAM Bits After SEU	Search for '0' ML State	Search for '1 ML State
0	0	1	1-bit Miss	3-bit Miss
1	1	0	3-bit Miss	1-bit Miss

Unsuccessful Successful

TABLE I
SEU possibilities in CAM bits

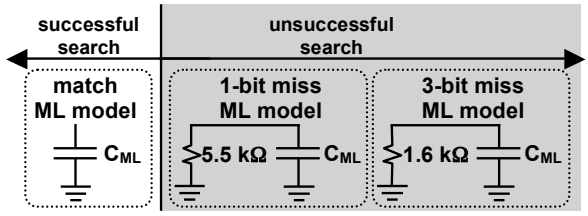

Fig. 3. ML models for the conventional current-race scheme.

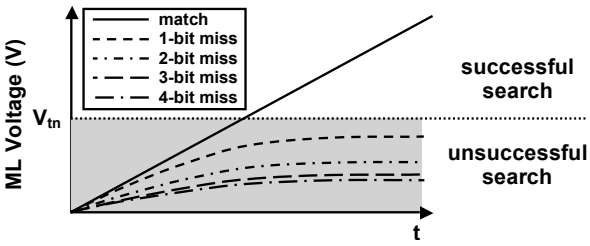

Fig. 4. Waveform of five matchline cases versus time in the conventional scheme.

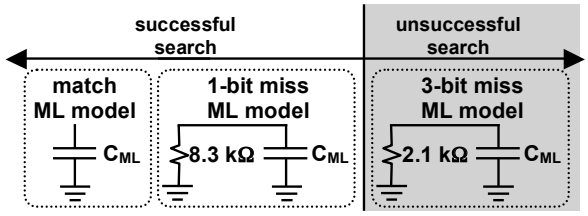

Fig. 5. ML models for the error-correcting-match scheme.

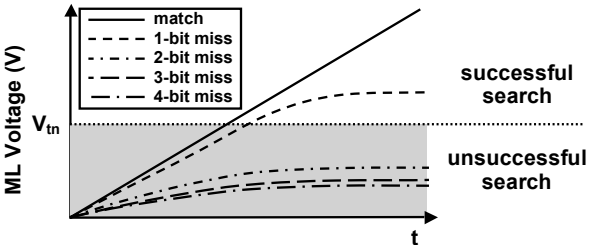

Fig. 6. Waveform of five matchline cases versus time in the error-correcting match scheme.

result in a code that requires, at worst, a logical XOR of 25 bits to generate each of the 9 parity bits. The 25-input XOR gate was implemented by cascading five levels of two-input XOR gates.

C. Matchline Sense Amplifier

Figs. 3 and 4 summarize the operation of the conventional current-race scheme using two figures. Fig. 3 depicts simplified circuit models for the ML for three cases: a match, a 1-bit miss, and a 3-bit miss. Fig. 3 shows that for the match case the matchline is modeled as a capacitor, C_{ML}, composed of the ML parasitic wire capacitance and the drain capacitance of the CAM cell transistors. For the 1-bit miss case, the ML is modeled as the capacitor, C_{ML}, in parallel with the CAM cell pulldown resistance of 5.5 kΩ. The 3-bit miss case is the same as the 1-bit miss, but the pulldown resistance is decreased to 1.6 kΩ due to the extra CAM cell paths. Fig. 4 plots how the ML charges over time for five different cases: match case, 1-bit miss, 2-bit miss, 3-bit miss, and 4-bit miss. We see that only the match case crosses the MLSA threshold and thus it is the only case that results in a successful match.

Figs. 5 and 6 summarize the operation of our proposed error-correcting-match scheme that modifies the current-race method. Fig. 5 shows that in our scheme both the match case and the 1-bit miss case result in a successful search. A 3-bit miss or higher is considered an unsuccessful search. For the purpose of this paper, we correct at most a single-bit upset, so 2-bit misses are not possible since we use a code with a minimum Hamming distance of four. Section V examines how to extend this scheme to account for multiple bit errors.

Fig. 6 plots how the ML charges over time for five different cases: match case, 1-bit miss, 2-bit miss, 3-bit miss, 4-bit miss, 5-bit miss. Both the match case and the 1-bit miss cross the V_{tn} threshold of the MLSA and therefore result in a successful search. To force the 1-bit miss to cross the threshold and to increase the margin between an 1-bit miss and a 3-bit miss, we make two modifications to the current-race scheme. First, the magnitude of the current source I_{ML} is increased from about 30 µA in the conventional scheme to about 55 µA in our error-correcting-match scheme. Second, we lower the voltage-swing of the SLs so that the effective pulldown resistance of the CAM cell increases. (due to the lower gate voltage on transistor MLS*i* in Fig. 2).

Fig. 5 shows that the difference in resistance between a 1-bit miss and a 3-bit miss grows from 3.9 kΩ (5.5 kΩ − 1.6 kΩ) in the conventional scheme to 6.2 kΩ (8.3 kΩ − 2.1 kΩ) in our error-correcting-match scheme. This increased difference in resistance between a 1-bit miss and a 3-bit miss, by almost 60%, increases the available sense margin. The decrease in the SL voltage swing has the added benefit of reducing SL power consumption. Since we increase the power consumption of the MLs by increasing the I_{ML} current, we use the decrease in SL power to maintain the same overall power consumption as a conventional CAM.

IV. Simulation Results

For both the conventional CAM and the error-correcting-match CAM, we simulate a 512×72 CAM block in a 0.18 µm CMOS process using a 1.8 V supply voltage. The schematic was annotated with extracted parasitics using a typical CAM cell area [9]. We designed the error-correcting match CAM to have the same power consumption and speed of operation as the conventional CAM using the current-race sensing scheme. Since our modified MLSA consumes more power than the current-race scheme, we save power by reducing the voltage-swing of the SLs from 1.8 V to 1.0 V.

Fig. 7 is a bar graph that plots the energy for both the conventional current-race CAM and our error-correcting CAM. The energy is reported in the units of fJ/bit/search and is divided into the ML and SL energy components. The error-correcting-match scheme increases the ML energy from 3.80 fJ/bit/search to 6.09 fJ/bit/search, but this energy cost is recovered by reducing the SL swing and saving 2.30 fJ/bit/search SL energy. We assume that the lower SL voltage is generated from the 1.8 V supply and thus the energy savings is directly proportional to the reduction in swing. If the reduced SL voltage is available

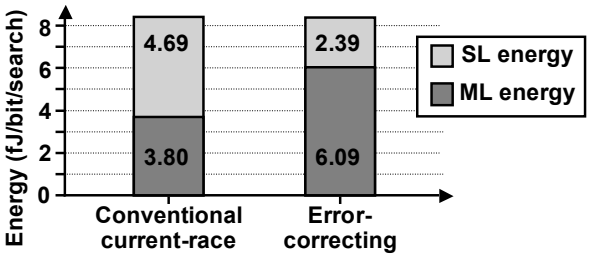

Fig. 7. The energy of the conventional current-race scheme and the error-correcting-match scheme divided into the ML energy component and the SL energy component.

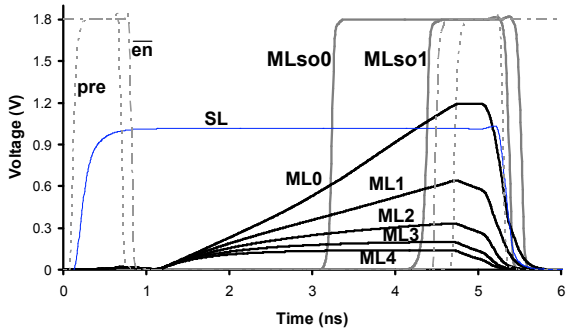

Fig. 8. Waveform plot of a single-cycle search operation in the error-correcting-match scheme.

from a switching power supply, the SL energy savings would increase to 3.47 fJ/bit/search.

Fig. 8 shows the simulated timing waveforms for a single-cycle search operation in the error-correcting-match scheme. The cycle begins by pulsing the pre signal and driving the SL signal to the CAM cell. Once precharge is complete, the $\overline{\text{en}}$ signal initiates the ML sensing by connecting the I_{ML} current source to the ML. The diagram shows the five different ML cases, labeled as MLn, where n represents the number of bits that miss in the word. A replica ML that is programmed to have a 1-bit miss controls the shut-off of the $\overline{\text{en}}$ signal. The storage bits in the replica ML are immune to soft errors because they are forced to the appropriate values by connecting their storage nodes to the power supply or to ground. The encoder delay of 1.0 ns is less than the CAM minimum clock cycle time and thus encoding can be performed in a pipeline stage before the search. The encoder consumes 112 fJ/search, which is equivalent to the power consumption of 120 bits in the CAM array; thus the encoder adds only 0.3% energy overhead to our relatively small size CAM.

V. Discussion

Throughout this paper, we have assumed that our system need only produce correct matches for a single-bit error per word. Thus we have used an extended Hamming code that has a minimum Hamming distance of four. While we could have used a code with a Hamming distance of three, which can correct a single-bit error, the extra bit of Hamming distance helps increase the sense margin of the MLSA.

In general, we can increase the number of bit-errors per word that are search-corrected by increasing the minimum Hamming distance of the code. For example, a distance six code would produce correct results for up to two bit-errors per word. The increased immunity obtained by using such a code results in additional area overhead and ML parasitic capacitance for each additional parity bit.

While this scheme produces correct matches during the search operation, it does not correct upset bits back to their original state. To correct the errors, each word can be serially read, corrected, and written back into the CAM in the background, or the method of [5] can be used. Furthermore this scheme does not depend on any special processing steps and is compatible with SER-reducing layout and process methods.

VI. Conclusion

This paper presented an error-correcting-match scheme for CAMs that is tolerant to bit errors in the stored contents. The scheme adds parity bits to each word and modifies the MLSA so that matches and 1-bit misses constitute a successful search, and all other cases constitute an unsuccessful search. We modify the current-race ML scheme to create our error-correcting ML scheme. Our design also reduces the SL voltage swing to increase the difference in ML pulldown resistance between a 1-bit miss and a 3-bit miss. Reducing the SL swing additionally reduces the power consumed by the SLs, compensating for the increased power consumption due to the error-correcting MLSA. The error-correcting-match scheme adds an area overhead of only 12% to correct 1-bit SEUs in CAMs.

Acknowledgment

We thank Prof. Frank R. Kschischang, of the University of Toronto for valuable discussions on the coding theory portion of this work.

References

[1] K. Pagiamtzis and A. Sheikholeslami, "Content-addressable memory (CAM) circuits and architectures: A tutorial and survey," *IEEE J. Solid-State Circuits*, vol. 41, no. 3, pp. 712–727, March 2006.

[2] H. J. Chao, "Next generation routers," *Proc. IEEE*, vol. 90, no. 9, pp. 1518–1558, September 2002.

[3] R. C. Baumann, "Soft errors in advanced semiconductor devices—Part I: Three radiation sources," *IEEE Trans. Device Mat. Rel.*, vol. 1, no. 1, pp. 17–22, March 2001.

[4] H. Noda, K. Inoue, M. Kuroiwa, F. Igaue, *et al.*, "A cost-efficient high-performance dynamic TCAM with pipelined hierarchical search and shift redundancy architecture," *IEEE J. Solid-State Circuits*, vol. 40, no. 1, pp. 245–253, January 2005.

[5] H. Noda, K. Dosaka, F. Morishita, K. Arimoto, *et al.*, "A soft-error immune maintenance-free TCAM architecture with associated embedded DRAM," in *IEEE Custom Integrated Circuits Conf.*, September 2005, pp. 451–454.

[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] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*. New Jersey: Prentice Hall, 1995.

[8] S. Gregori, A. Cabrini, O. Khouri, and G. Torelli, "On-chip error correcting techniques for new-generation flash memories," *Proc. IEEE*, vol. 91, no. 4, pp. 602–616, April 2003.

[9] P.-F. Lin and J. B. Kuo, "A 0.8-V 128-kb four-way set-associative two-level CMOS cache memory using two-stage wordline/bitline-oriented tag-compare (WLOTC/BLOTC) scheme," *IEEE J. Solid-State Circuits*, vol. 37, no. 10, pp. 1307–1317, October 2002.