Journal of **N**onlinear **A**nalysis and **O**ptimization
Vol. 14, Issue. 1 : 2023
ISSN : **1906-9685**

TERNARY-BASED SOFT ERROR-RESISTANT SRAM MEMORY: ADDRESSING BY CONTENT AND ENHANCED WITH CHECKSUM METHOD

[#1]KOKKULA SHAILAJA, ***Assistant. Professor,***
[#2]MAVURAPU SWAPNA, ***Assistant. Professor,***
Department of Electronics Communication Engineering,
SREE CHAITANYA INSTITUTE OF TECHNOLOGICAL SCIENCES, KARIMNAGAR, TS.

Abstract: Ternary Content Addressable Memory (TCAM), which is based on Static Random Access Memory (SRAM), is used in Software Defined Networking (FPGAs) packet classification and open flow applications. As a result, SRAM-based TCAMs will be required in all FPGA-based systems to reduce the risk of soft mistakes, boost search speed, and decrease critical path time. SRAM-based TCAM can identify single-bit parity faults with reduced critical route overhead, saving money and reducing response times. Using an SRAM-based, binary-encoded TCAM table, this technology enables quick error correction. The goal of this study is to create a 1024x40 SRAM-based TCAM with improved error detection and correction capabilities as well as increased security via the Checksum technique. The project's success in terms of area, latency, and power consumption was demonstrated by building and synthesizing it in Verilog HDL on a Xilinx Vertex 5 FPGA.

***Keywords*:** SRAM (static random-access memory), TCAM (ternary content addressable memory), FPGAs (field programmable gate arrays), Soft errors, and Addressable Memory are all examples of addressable memory.

1. INTRODUCTION

Charged particles can disrupt information stored in memory cells, resulting in silent errors. Silent errors are becoming more common in computer systems that must be exceedingly reliable. There are signs that the in-memory technology is failing. Charged particles are emitted when semiconductors are subjected to radiation. A single weak connection might cause the entire system to fail. Because TCAM is used to generate SDNs, it is crucial in routers and other high-availability networking systems. The TCAM is an important component of networking hardware that is necessary for tasks such as packet routing, packet forwarding, and packet categorization. Because routers and other network components must be built with particular care to handle soft errors, soft error avoidance is a difficult TCAM problem to solve. Field programmable gate arrays (FPGAs) are programmable integrated circuits that can be updated after they have been manufactured by the end user or designer. A language known as HDL is used to define the FPGA settings. The language used to describe ASICs (purpose-built circuits) is similar. FPGAs nowadays have so many logic circuits and RAM blocks that they can do exceedingly complex digital calculations with ease. TCAM functionality is often illustrated using data stored in the on-board memory of SRAM-based FPGA systems. An inaccurate match address could be caused by temporary difficulties, block RAM (BRAM), or dispersed RAM. This could lead to an incorrect identification or mismatch. When a soft mistake occurs, the affected SRAM word must be changed so that lookups deliver proper matching data. However, keeping up with SRAM-based TCAM systems without significantly reducing search or critical path delay is difficult. This brief study describes a simple, low-cost, and uncomplicated method of protecting SRAM-enabled TCAMs while slowing down search. Regression testing is straightforward, requires little preparation or planning, and relies on extensive testing. To compensate for minor

inaccuracies, the suggested technique makes use of a redundant duplicate of the TCAM table stored in binary code in SRAM-based TCAM devices. Even when functioning in the background, the proposed failure system retains a high searching speed, allowing for concurrent searches. Because of its rapid reaction time for the majority of the transaction lifetime, the suggested error-correction approach produces an optimal TCAM configuration for lookups.

2. TCAM ON FPGAS BASED ON SRAM

To generate TCAM, new FPGAs make use of on-chip SRAM. A "0" TCAM authoritarian leadership is preserved by storing a "1" in RAM, a "x" state is preserved by storing a "1" in both SRAM and RAM, and a "1" quantity is preserved by storing a "1" in SRAM with this type of 2X1 RAM. As a result, a 1X1 TCAM is achievable. A C-bit TCAM layout can be created utilizing a 1-bit SRAM with 2C locations. TCAM associates each word with a list of all possible C-bit patterns that it matches or does not match. The address words of an SRAM represent the C-bit TCAM architecture. A C-bit wide SRAM with 2C locations can be used to generate a B-word wide TCAM table. SRAM-based TCAMs were partitioned because they couldn't handle the huge TCAM bit patterns required by the researchers. TCAM partitions D bit patterns with a width of W bits into C bit chunks and executes them with a 2C D size Order. A simple SRAM-based TCAM configuration. The 4-bit patterns of a 4-word deep TCAM are divided into two 4-by-2 halves. These are activated by using two 4-by-4 SRAM. Examine the different indexed digits (1001). While reading from the first SRAM, bits 0 and 1 get the third word (1100), while bits 0 and 1 retrieve the second phrase (1001). ANDing the collected SRAM bits yields a final match result of 1000, indicating a match according to rule R0.

TCAM can be implemented in a variety of ways on FPGAs. Small TCAMs are built from flip flops (FFs) present in FPGA logic blocks. Since we've already addressed this, let's just state that most modern FPGA-based TCAMs use BRAM, also known as distribution RAM. TCAMs are constructed using shallow SRAMs, which are more energy-efficient memory chips. Because a Xilinx BRAM design can have a maximum depth of 512 bits, only 29 BRAM bits are required to handle nine TCAM bits. The ideal setup sizes for 18kb and 36kb BRAMs, respectively, are 51236 and 51272, assuming a piece width of 9 bits in the TCAM. These TCAMs use distributed RAM and have 30 TCAM bits per SLICEM. Each of the four 64-bit LUTRAMs is turned into a 32-bit 6-port dual-channel RAM with the fundamental essentials. This means that one bit of TCAM takes 853 bits of distribution RAM, whereas 30 bits of TCAM require 256 bits of distribution RAM. SRAMs based on distributed RAM should have a depth of 32 bits and a TCAM chunk width of 5 bits. TCAM requires BRAMs to be created, whereas LUTRAMs are used for specialized system components.

Soft errors are easier to detect in TCAM setups with higher resource requirements. Slice files, LUTRAMs, and BRAMs take up 34%, 5%, and 1% of the total area of a 28-nanometer field-programmable gate array (FPGA). As a result, the occurrence of SEUs is significantly reduced. Because of this, BRAM-based TCAM systems are less dependable and more prone to minor errors than their distributed RAM and SR equivalents. The FPGA device's BRAM failure rates were employed in this scenario to highlight the project's real-time soft error rate for each event. Vertex-5 has a failure rate of 27%. When the ranges of two words overlap, TCAM words have the wildcard state "x," which allows a 1-bit word to match numerous phrases. In a multiple-word match, just the lowest-order word is communicated. To order TCAM words, priorities must be employed. Lower memory addresses receive priority words. If a TCAM word is changed, the terms in the TCAM table may be updated, and the sequence of the words may change. When a precise match between incoming terms is found, the lookup process is ended by update steps. Because of the prominence of software-defined networking and open flow, network switches will need to be changed more frequently in the future. SRAM-based TCAM solutions on FPGAs alter a TCAM item in the TCAM data table to simulate the TCAM function's behavior. This must be done several times in order for the order constraint to hold. By recycling data from previously stored binary-encoded TCAM material, the proposed ER-TCAM corrects minor faults in SRAMs that imitate TCAMs.

3. PROPOSED METHODOLOGY

SRAM is used to classify traffic and implement open flow applications in a (SDN) (FPGAs). Field engineering arrays include ternary programmable memory (TCAM) as part of their architecture. To reduce the risk of soft errors, boost search performance, and decrease critical path time, all FPGA-based devices will require SRAM-based TCAMs. Finding single-bit parity faults with the cheapest essential path reduces the cost and response time of SRAM-based TCAM, and a TCAM bar counter SRAM based on binary coded TCAM is used to simplify error repair at slow reaction times. The goal of this study is to show that an error-correcting 1024x40 SRAM-based TCAM, created with a Xilinx Vertex 5 FPGA and Verilog HDL code, is feasible in terms of size, latency, and power consumption. In SRAM-based TCAMs, error correction necessitates the storage of duplicate data. As a TCAM cell can be formed of three SRAM bits, data saved in SRAMs capable of storing TCAM is at least 2D C bits in size ("0""00," "1""01," and "x""10"). As previously stated, the original TCAM's contents are also saved on the chip for use with subsequent SRAM-based TCAMs. This renders the data contained in the design-configured SRAMs (2C x D bits) entirely worthless. To compensate for intermittent SRAM failures, ER-TCAM employs data redundancy at the system level. This key principle underpins ER-TCAM. The TCAM table divisions require the two SRAMs shown in the image to work.

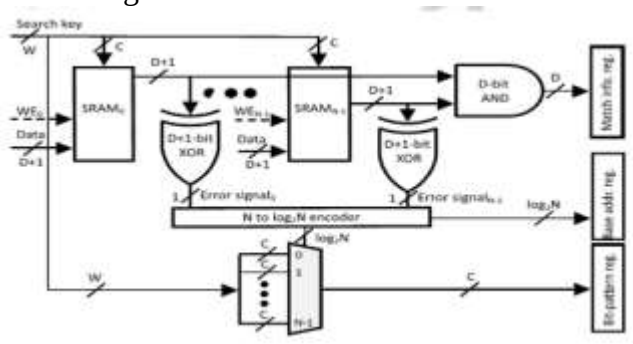

Figure 1: ER-TCAM suggested architecture error detection

In order to locate an SBU, the ER-TCAM, as indicated in the picture, augments each SRAM word by one bit. First, every piece of SRAM was verified for defects during lookups. The ER-TCAM corrects misspelled words by referring to the TCAM database for extra information.

The proposed layout for the ER-TCAM to identify errors is depicted in Figure 1. When a lookup with a search key is performed, the SRAM reading bit is EX-ORed to generate an error signal. The TCAM system encodes the N SRAM error signals into a log2N-bit error message to determine which SRAM is faulty. The error code and any relevant search-key bit patterns are sent to the error-correction module. The system then outputs an error message of length log2N bits, highlighting each faulty SRAM. It sends the error code and any applicable search-key bit patterns to the error-fixing program.

4. ARCHITECTURE OF THE PROPOSED ER-TCAM

The entire configuration of the ER-TCAM is depicted in Figure 2. A gaffe read/write controller is included, as well as an ECV compute unit, an address generation unit (AGU), and an SRAM for storing TCAM database tables. With each repetition, the MOD-D counter generates a new set of log2D bits. To choose the SRAM word within the sub-block, the log2D bits at the complement's bottom are read. The most critical bits of the SRAM ID indicate the start of the SRAM sub-block in question. AGU can now see all of the binary-coded integers in the appropriate section of the TCAM database. A match bit is created every cycle by comparing the read TCAM words to the C-bit pattern. The ECV can be determined after collecting this bit for D clock cycles. Once the read/write controller sends an enable signal for student success, the ECV can be written over the damaged SRAM word.

The base SRAMs of the TCAM can be accessed for retrieval operations, allowing the ER-TCAM to ease search activities throughout the error-correction process. The ER-TCAM turns these SRAMs into dual-port RAM by reading and writing in a certain order within the same clock cycle. Once the ECV has been calculated and written to the write port of SRAM, the error correcting device inhibits the ER-TCAM's search and rescue operations. The error rate of an SRAM that stores a binary-encoded TCAM table is significantly lower as compared to SRAMs that renew TCAM tables due to the amount of the soft error. By storing the binary-encoded TCAM table in ECC, the

ER-TCAM may be able to continue using SRAM with minimum memory and effort spent correcting errors.

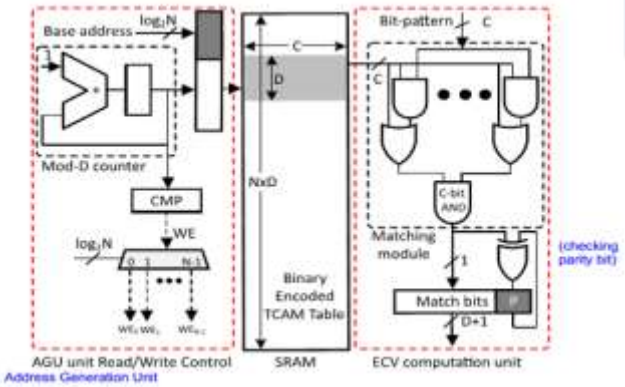

Figure 2: tool proposed for ER-TCAM flaw correction

5. METHODN FOR CALCULATING THECHECKSUM

Many individuals feel that checksums outperform LRC, VRC, and CRC when it comes to identifying errors in higher layer protocols. It aids top-layer protocols in detecting problems. On the receiving end of a communication, a checksum generator is used. At the receiving end, a checksum verification is used. The Sender's hash generator divides the data into identical n-bit chunks. This object is approximately 16 inches long. To put these distinct components together into a useful whole, the one's complement technique is used. The final step is to add to the newly created bit. Before transmitting the preceding data unit to the receiving machine, the checksum is computed and used.

The collected data and checksum are subsequently sent to the checksum verifier by the user. If the data unit is divided into many pieces, the checksum analyzer adds up all the pieces of the same length. Because checksum is a component, it must be one of these sections. When a job is completed, it is praised for its quality. The data is correct if the adjusted result is zero. If the receiver rejects the input due to a dataset error, a non-zero result is obtained.

6. EXPERIMENTAL RESULTS AND ANALYSIS

Table: ER-TCAM based on a Ternary SRAM Cell and Capable of Withstanding Soft Errors: Xilinx Vertex-5 FPGA-based Content-Addressable Memory

Table: 1. a comparison of existing techniques is provided.

TCAM Size	1024x40	
	Parity	Checksum
Slice Register	39	39
LUT	96	91
Occupied Slices	51	36
Number of IOBs	71	71
Delay (ns)	2.402	2.292
Power (W)	3.314	3.315

Fig:3. A Comparison of Bit Parity and Checksum Characteristics

Single-bit parity testing makes it easy to locate mistakes in the Error Correction Module with minimal processing lag and effort. In the process of mistake repair, binary decoding is used repeatedly. TCAM approaches use SRAM to keep a TCAM table that can be changed to adjust for minor faults. The suggested error-correction technique is so efficient that it can handle numerous searches at the same time, even when running in the background. The outcomes of a simulation with a single error fixed. Verilog code is used to test the design's functionality. To validate the new architecture, we used Modalism for Verilog code simulation and Xilinx ISE for design implementation. The BRAM architecture has a storage capacity of 1024 by 40 bytes for simulation data. This allows us to compare the efficacy of the Enhanced plan to the status quo.

7. CONCLUSION

This study recommends updating memory-based and model TCAMs with on-chip copies of the original TCAM data in order to discover and repair flaws. This not only saves space, but also reduces the latency in critical links and

speeds up reactions. With minimal inference and optimal travel time, the suggested technique discovers single-bit parity concerns. To overcome faults in SRAM-based TCAMs, the suggested error resilience technique, ER-TCAM, leverages the binary-encoded TCAM table. SRAMs with TCAM functionality can be used for search operations during the background error-correction process. The suggested error-correction approach has no effect on data channel processing. With an EDD of 8 nanoseconds and a known error-correction time of 260 nanoseconds, the ER-TCAM on the Vertex-5 FPGA device could do up to 2 billion searches per second. Some error-correction algorithms, on the other hand, have extremely long error-correction times and perform badly. The EDD of the ER-TCAM is at least twice that of previous systems of its sort. The ER-TCAM could be used in systems where SRAM-based TCAMs are used to speed up specific activities, such as data networks.

REFERENCES

1. P. He, W. Zhang, H. Guan, K. Salaminian, and G. Xie, "Partial order theory for rapid TCAM updates," IEEE/ACM Trans. Newt., vol. 26, no. 1, pp. 217–230, Feb. 2018. [2] P. He, W. Zhang,
2. H. Guan, K. Salaminian, and G. Xie, "Partial order theory for fast TCAM updates," IEEE/ACM
3. "FAS: Using FPGA to accelerate and secure SDN software switches," Secure. Common. Newt., vol. 2018, Jan. 2018, Art. no. 5650205, W. Fu, T. Li, and Z. Sun.
4. T. Li, H. Liu, and H. Yang, "Design and characterization of SEU hardened circuits for SRAM-based FPGA," IEEE Trans. Very Large-Scale Integral (VLSI) Syst., vol. 27, no. 6, Feb. 2019, pp. 1276–1283.
5. T. Li, H. Yang, H. Zhao, N. Wang, Y. Wei, and Y. Jia, "Investigation into SEU effects and hardening techniques in SRAM based FPGA," in Proc. 17th Eur. Conf. Radiant. Effects Common. Syst. (RADECS), pp. 1–5.
6. A. Ramos, R. G. Toral, P. Reviriego, and J. A. Maestro, "An ALU protection methodology for soft processors on SRAM-based FPGAs," IEEE Trans. Computer., vol. 68, no. 9, March 2019, pp. 1404–1410.
7. S. Pontarelli, P. Reviriego, and J. A. Maestro, "Parallel D-pipeline: A cuckoo hashing implementation for enhanced throughput," IEEE Trans. Computer, vol. 65, no. 1, March 2015, pp. 326–331.
8. P. Reviriego, A. Ullah, and S. Pontarelli, "PR-TCAM: Efficient TCAM emulation on Xilinx FPGAs utilising partial reconfiguration," IEEE Trans. Large-Scale Integral (VLSI) Syst., vol. 27, no. 8, pp. 1952–1956, Mar. 2019.
9. Xilinx Product Guide PG190, Ternary Content Addressable Memory (TCAM) Search IP for SD Net V1.0; Xilinx, San Jose, CA, USA, Nov. 2017.
10. W. Jiang, "Scalable Ternary content addressable memory implementation utilising FPGAs," in Proceedings of the 9th ACM/IEEE Symp. Archit. Newt. Common. Syst., pp. 71–82, 2013.
11. G-AETCAM: Gate-based area-efficient Ternary content-addressable memory on FPGA," IEEE Access, vol. 5, pp. 20785–20790, 2017. M. Irfan and Z. Ullah, "G-AETCAM: Gate-based area-efficient Ternary content-addressable memory on FPGA," IEEE Access, vol. 5, pp. 20785–20790, 2017.