Implementation of FPGA-Based Ternary Content Addressable Memory Updating Mechanism Based on Reversible Logic

P. Pravallika, Dr. M. Sailaja
Department of ECE, University college of Engineering, Kakinada, JNTUK Kakinada, Andhra Pradesh, India

Abstract: **The primary aim of this conceptualization is to devise a highly effective updating mechanism for TCAM (Ternary Content-Addressable Memory). Within this project, we introduce two innovative mechanisms for enhancing FPGA-based TCAMs: the first is an expedited MUX-Update mechanism, while the second is a cost-efficient LUT-Update mechanism. We have developed a 64*36 Gate-Area Effective TCAM for experimental purposes. The MUX-Update mechanism delivers a remarkably swift update latency of W+1 clock cycles, all while utilizing a mere trio of Input/Output (I/O) pins. Meanwhile, the LUT-Update mechanism consistently maintains a low and predictable update latency of just 2 clock cycles, regardless of the TCAM's size, accomplished through the utilization of W I/O pins. Furthermore, to expand upon this concept, we have introduced a novel de-multiplexer design based on reversible gates, aimed at mitigating power constraints.**

Keywords: Look Up Table, Ternary Content-Addressable Memory, Field Programmable Gate Array, virtual memory management.

INTRODUCTION

Ternary content-addressable memory (TCAM) stands as a marvel in the realm of memory technologies, offering the ability to pinpoint the address of input data in a mere blink of a clock cycle [1], [2]. It proves its mettle in scenarios where the temporal gap between two critical events – the moment of input and the instant of the desired data (address) materializing at the output – carries paramount significance. Two main ways in which TCAM differs from regular RAM are its higher capacity and its lower power consumption. The X-bit—the don't care bit—returns an address rather than data; it stores a strange and versatile entity known as the 'X-bit'; the 'X-bit' is a storage area for a mysterious and flexible creature. Due to its ambiguity, this "I don't care" phrase may be matched with either "1" or "0."

The applications of Content-Addressable Memories (CAMs) traverse a vast spectrum, encompassing realms where the concurrent processing of data reigns supreme, rendering lightning-fast operations a necessity. Network routing, database administration, positioning systems, pattern recognition, processor-specific cache memory, and even the interesting subject of graph similarity searches have all benefited from the use of CAMs to speed up application development. Artificial intelligence, radar signal monitoring, bioinformatics, image processing, and many other fields have already adopted binary content-addressable memories (BCAMs) and transient content-addressable memories (TCAMs) as indispensable storage media [6].

Content-addressable memory (CAM) takes center stage by furnishing the sought-after search input's position in a solitary clock cycle [1]. CAM may be broken down into binary (BCAM) and ternary (TCAM) subtypes, depending on the number of bits it contains. While BCAM accommodates the binary duo of '1' and '0,' TCAM elevates the stakes, accommodating '1,' '0,' and the enigmatic 'X' (the don't-care) bit. Networking, signal processing, pattern recognition, access control lists, and microprocessor translation lookaside buffers (TLB) are just some of the many places you may see CAMs at work today.

Field-programmable gate arrays (FPGAs) have surged in popularity due to their profound hardware parallelism, malleable reconfigurability akin to software, and swiftness in prototyping. Block random-access memory (BRAM), multiplexers, and other specialised hardware components are all included in these FPGAs. The CAMs based on FPGAs use these capabilities to simulate a CM [2, 3]. They deliver search-latencies (lookup-latency) nearly equivalent to a single clock cycle, yet they stumble when it comes to updating, bogged down by a ponderous update-latency that disrupts the equilibrium necessary for synchronous update and search operations. To harness the full bandwidth of high-performance systems, memory updates must transpire at a brisk pace.

In conventional FPGA-based CAMs, the update-latency invariably scales with O(2N) [4], a rate deemed unacceptable for systems craving balance and efficiency. Innovations have sought to mitigate this latency, reducing it to O(N) [5] and further down to O(N/k) [6], contingent on the number of word groups stored. Despite these strides, the update-latency remains disproportionately high and unpredictable when juxtaposed with the consistently low (one clock cycle) and steadfast search-latency of FPGA-based CAMs. In areas, for example, organizing, where the CAM table goes through successive updates, the pace must match the search speed to attain the coveted high bandwidth.

The performance of FPGA-based CAMs deteriorates as the update-latency climbs in correlation with the quantity of put away words in existing models [7]. In the space of IP organizing, information bundles showing up at a hub necessitate buffering to compensate for the sluggish updates and forestall packet loss [8], [9]. Consequently, these protracted updates impose an additional overhead on the system, manifesting as an oversized buffer—a predicament that could be averted through the acceleration of the update process.

Within the scope of this letter, we provide two novel updating mechanisms: Alterations to the MUX and the LUT. These innovative mechanisms not only revamp TCAM in a manner that expedites updates but do so with a cost-effective approach, surpassing the conventional updating procedures of FPGA-based CAMs.

LITERATURE SURVEY

Within the realm of cutting-edge memory technologies, numerous pioneering techniques have emerged, each aiming to curtail power consumption while enhancing the efficiency of Content Addressable Memories (CAMs). In [9], a remarkable approach was introduced, christened the "selective precharge technique." This ingenious method entails dividing the match line into two distinct segments. Initially, the search operation embarks on its quest within the first segment, targeting the initial few bits of a word—essentially, a select subset of CAM cells. Only if a match is detected within this initial segment does the search extend its reach into the second segment.

In the quest for memory architectures endowed with qualities such as low-power consumption, cost-effectiveness, and unwavering reliability, [8] unveiled a formidable contender: the "fully parallel precomputation-based content addressable memory" (PBCAM). This design hinges on the prowess of precomputation, orchestrating an ingenious reduction in the number of comparisons during the second phase of the comparison process. To achieve this, a "one's count" approach takes center stage in precomputation. However, it's worth noting that the use of a chain of full adders for designing a "one's count" parameter extractor introduces a delay that escalates with the length of data bits.

Turning our attention to [7], we encounter yet another innovation in the pursuit of power-efficient CAMs—the "pipelining technique." Here, the search operation undergoes pipelining by segmenting the match lines into multiple distinct sections. Given that most stored words do not align in their initial segments, the search operation is strategically terminated for subsequent segments, thereby delivering power savings as a direct outcome. The crux of the matter lies in activating only a fraction of the matchline segments, contributing to a notable reduction in power consumption.

In a bid to enhance the efficacy of the low-power precomputation-based CAM (PBCAM) proposed in [8], [5] introduces a revolutionary "Block-XOR approach." This pioneering approach brings forth a Block-XOR parameter extractor, tailor-made for low-power PB-CAM. Notably, this paper furnishes both theoretical and empirical evidence to affirm the Block-XOR PB-CAM's proficiency in achieving substantial power reduction by curtailing the quantity of correlation tasks during the second period of the examination interaction. This underscores the approach's flexibility and adaptability in accommodating diverse design scenarios.

Beyond CAMs, [15] embarks on a fusion of Hamming and parity codes, aptly named "Matrix Code (MC)," as a safeguard for SRAM memory. Remarkably, MC outperforms Hamming, proficiently rectifying multiple errors per word with minimal decoding delay. However, it's essential to note that for Hamming code to correct errors, two vertical syndrome bits must be activated.

Moreover, [16] introduces an innovative amalgamation of Hamming code with a decimal algorithm. This ingenious blend facilitates the detection and correction of soft errors while imposing minimal delay overhead by incorporating integer values into the equation. Hamming code, renowned for generating parity codes [17], takes center stage in error correction.

Furthermore, the literature extends its focus to memory reliability enhancements, spanning Decimal Matrix Code (DMC) in comparison to single error remedy with twofold blunder identification in light of Hamming [3] and lattice code, as expounded in [20].

While typical check bit-based approaches exhibit the generator and equality really take a look at frameworks in methodical structure — straightforwardly planning message bits into codewords and affixing excess equality pieces to the message's end —these strategies tend to wane in efficacy when faced with multiple errors induced by hard faults. Potent schemes like Chip Kill, while robust, incur excessive overheads and are mismatched for implanted recollections.

In light of these challenges, this work pioneers a novel ECC (Error Correction Code) construction, marked by exceptionally low overheads, rendering it an ideal fit for low-cost IoT devices, which may sporadically encounter single-bit errors.

MUX-UPDATE MECHANISM

The forthcoming depiction unveils the ingeniously designed architecture of the MUX-Update mechanism, a technological marvel poised to transform TCAM updating. This mechanism bestows upon TCAM the remarkable capability to refresh an entry in a fixed span of merely two clock cycles. The initial clock cycle precisely coordinates the placement of the "storing bits" inside the G-AETCAM, making sure that they are always in even bit positions within each word location. As for the "masking bits," they are carefully placed in the odd bit places inside each word location during the second clock cycle, where they will remain intact.

The utilization of even and odd bit positions serves as a brilliant strategic maneuver, one that serves a dual purpose. Not only does it streamline the updating process, but it also optimizes the utilization of Input/Output (I/O) pins. This resourceful approach stems from the pragmatic recognition of the constraints imposed by the finite resources of FPGA, ensuring an efficient and effective updating process.

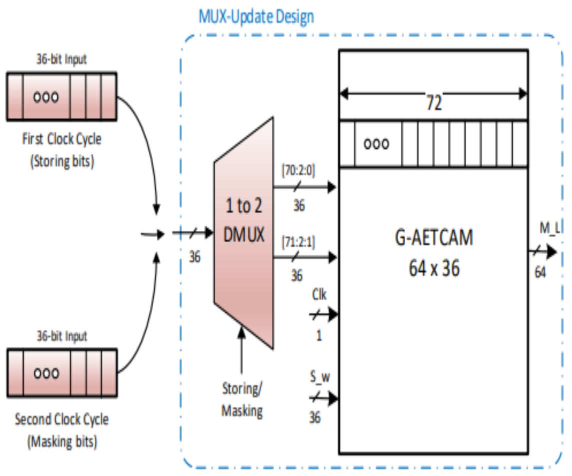


Fig 1. represents mux-update mechanism

The intriguing intricacies of this process are unveiled as we delve into the realm of digital manipulation. The select line for the 1-to-2 DMUX, responsible for handling the storing bits, stands resolutely at '0'. In a captivating contrast, the select line for managing the masking bits confidently asserts itself as '1', as beautifully depicted in Figure [1].

Algorithm 1 Updating 64×36 TCAM using MUX-Update Mechanism

```
Input: W TCAM bits (X, 1, 0), where W is 36.
Output: G-AETCAM bits (1, 0)
1:  G-AETCAM [0:71] = "000...00";
2:  if SM == '0' then
3:      for i ← 0 to i ← 35 do
4:          G-AETCAM [2*i] = TCAM [i];
5:      end for
6:  else
7:      for i ← 0 to i ← 35 do
8:          if TCAM [i] == '0' or TCAM [i] == '1' then
9:              G-AETCAM [(2*i)+1] = '0';
10:         else if (TCAM [i] == 'x') then
11:             G-AETCAM [(2*i)+1] = '1';
12:         end if
13:     end for
14: end if
    Note: SM: Storing/Masking bit.
```

This symphony of digital choreography forms the essence of the MUX Update mechanism, a technological virtuoso that orchestrates the updating of TCAM words with unparalleled precision. Its prowess is vividly portrayed through the rhythmic cadence of clock cycles, where Line #4 and Line #8 in Algorithm 1 stand as the heralds of this meticulously timed process. In a mere two clock cycles, the transformation of TCAM words is achieved, a testament to the elegance and efficiency of this mechanism.

LUT-UPDATE MECHANISM

Prepare to embark on a journey into the realm of cutting-edge technology, where the LUT-Update mechanism takes center stage. This architectural masterpiece promises nothing short of a revolution in TCAM updates. Picture a symphony conducted in W+1 clock cycles, where 'W' symbolizes the width of the TCAM, and each note plays a pivotal role in this digital masterpiece.

Our tale begins with the Bit-Select-Memory (BSM), a formidable ensemble comprising two 3-input LUTs. Their mission? To seamlessly deliver the precise values required to guide the MUX's actions when confronted with the intricate dance of 0s, 1s, and the enigmatic Xs that make up the TCAM. With a flourish of input pins - I0, I1, and Ix - these LUTs deftly decode the TCAM's enigmatic language.

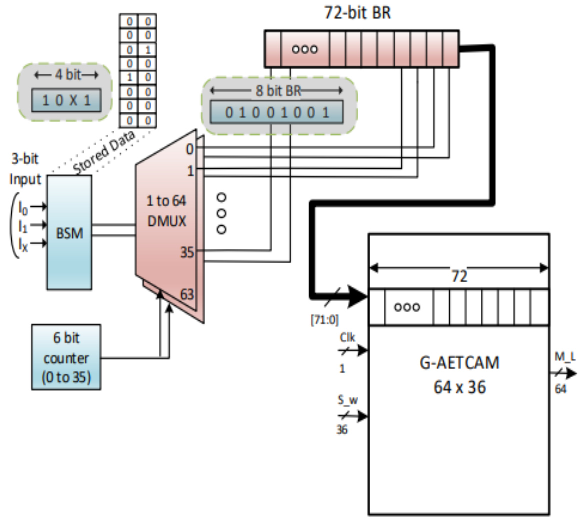


Fig. 2: lut-update proposal. (BSM: bit-select memory; BR: buffer register).

Consider, for instance, the TCAM entry "10X1," a mosaic of bits yearning for transformation, as vividly depicted in our illustrious figure. The BSM, akin to a virtuoso, stores a sequence of values - "01," "00," "10," and "01" - within its 8-bit buffer register (BR). And in the next heartbeat of the clock, this carefully orchestrated composition journeys from the BR to the corresponding nook within the TCAM's expansive memory.

Here's where the magic happens. In just five clock cycles, as 'W' proudly displays its value of 4, the TCAM metamorphosis unfolds. But remember, for a 36-piece word, this musical dance is rehashed a mesmerizing 36 times, each iteration filling the BR with the intricate tapestry of storing and masking bits. As the clock's pendulum swings once more, the contents of the BR gracefully waltz into their designated TCAM locations.

In the realm of algorithms, behold the elegant Switch statement in Algorithm 2, a digital maestro capable of interpreting the diverse melodies that the BSM presents. It elegantly selects from three possible cases, each harmoniously guiding the input values destined for the twin demultiplexers (DMUXes), each awaiting their moment in the spotlight.

Now, the number of times this grand performance repeats itself, within the for loop, is intricately tied to the TCAM's width, a testament to the dynamic nature of this captivating dance of technology. With each iteration, the TCAM memory is awakened and transformed, all orchestrated by the LUT-Update mechanism's artful design.

Algorithm 2 Updating TCAM using LUT-Update Mechanism

Input: TCAM bits (X, 1, 0)

Output: G-AETCAM bits (1, 0)

```
1: for $i \leftarrow 0$ to $i \leftarrow 35$ do
2:   Switch $I_x I_1 I_0$ do
3:     Case “001”:
         BR [2*$i$, (2*$i$)+1] = “00”;
4:     Case “010”:
         BR [2*$i$, (2*$i$)+1] = “01”;
5:     Case “100”:
         BR [2*$i$, (2*$i$)+1] = “10”;
6: end for
7: G-AETCAM [0:71] = BR [0:71];
```

Note: BR: Buffer Register; I_x, I_1 and I_0 are the three inputs of Lookup table shown in Fig.

In the symphony of TCAM updates, MUX-Update stands as a virtuoso, weaving a tapestry of efficiency that unfolds in a mere 2 clock cycles. As we peer into the heart of this orchestration, a striking revelation emerges, one illuminated by the graph in Figure 4.

The graph's narrative is crystal clear: as the TCAM's size expands, other available update designs [4], [5], [6] falter, weighed down by the encroaching specter of latency. It's a tale told through the steady ascent of update latency, a burden that increases in direct proportion to TCAM's growth.

Yet, amidst this sea of escalating latency, MUX-Update stands as an unswerving beacon of efficiency. Its capacity to update TCAM in a mere 2 clock cycles remains unrivaled, a testament to its clockwork precision. This revelation serves as a resounding testament to the MUX-Update's prowess, offering a glimpse into a world where speed and efficiency reign supreme.

PROPOSED METHOD DEMULTIPLEXER USING REVERSIBLE LOGIC GATES

REVERSIBLE LOGIC GATES:

In the enigmatic world of reversible logic gates, we encounter intricate devices that possess a unique property – a one-to-one mapping between their n inputs and n outputs. This distinctive characteristic allows us to decipher the outputs based on the inputs and, intriguingly, reverse-engineer the inputs from the outputs.

Yet, the synthesis of reversible circuits follows a peculiar set of rules. The concept of direct fan-out, where one input fans out to many outputs, is deemed non-reversible, and therefore, forbidden. To circumvent this constraint, reversible circuits employ additional gates, ingeniously orchestrating fan-out while maintaining reversibility.

In the realm of reversible circuit design, complexity and performance are underpinned by several critical parameters. First and foremost is the count of reversible gates employed in the circuit, aptly denoted as "N." Furthermore, we must consider the number of constant inputs, referred to as "CI," which are intentionally held at static values of 0 or 1 to synthesize the desired logical function.

Additionally, we encounter the concept of "garbage outputs" (GO), representing unused outputs within a reversible logic circuit. Surprisingly, these seemingly superfluous elements play a pivotal role in preserving reversibility, and their presence cannot be circumvented.

Lastly, we delve into the notion of "quantum cost" (QC), which encapsulates the circuit's cost in terms of primitive gate units. To ascertain this cost, we calculate the number of primitive reversible logic gates, often of dimensions 11 or 22, required to materialize the circuit.

Thus, within this cryptic domain, the intricacies of reversible logic gates come to light, marked by their one-to-one correspondence, nuanced synthesis rules, and a web of parameters defining their complexity and performance.

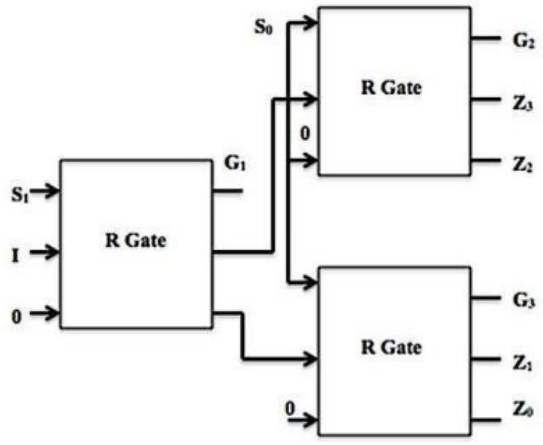


Fig 3.1x4 de-multiplexer using R-gate

In the intricate world of digital logic, the De-multiplexer (DEMUX) emerges as the counterpart to the Multiplexer (MUX). It serves as a remarkable device, tasked with the responsibility of taking a solitary input signal and meticulously selecting one among several output data lines, all connected to that very same input.

Visualize, if you will, a 1:4 De-multiplexer, a technological masterpiece hinging on the ingenuity of the R-gate. This wondrous creation boasts a single data input, ingeniously labeled as "I," complemented by two select lines, eloquently denoted as "S0" and "S1." These, in turn, orchestrate the symphony of four outputs – "Z0," "Z1," "Z2," and "Z3," each a distinctive note in the digital composition.

Now, let's delve into the world of reversibility, a captivating concept where the arrangement of information sources is meticulously mapped to an equivalent number of outputs, forging a seamless and harmonious correspondence. Reversibility finds its embodiment in the intricate gates that grace our designs, where power consumption is reduced to a remarkable minimum. A prime example is the illustrious R-gate, a 3x3 gate with three inputs – "A," "B," and "C" – and an ensemble of three outputs – "P," "Q," and "R."

The brilliance of the R-gate is unveiled as we examine its intricate logic. Its outputs are beautifully defined: "P" equals "A," "Q" equals "AB," and "R" equals "A'B+AC." This arrangement, while intricate, maintains a quantum cost of 4, a testament to its efficiency and elegance within the realm of digital logic.

In this realm, the DEMUX and its corresponding R-gate serve as beacons of ingenuity, simplifying and enhancing our understanding of the digital landscape.

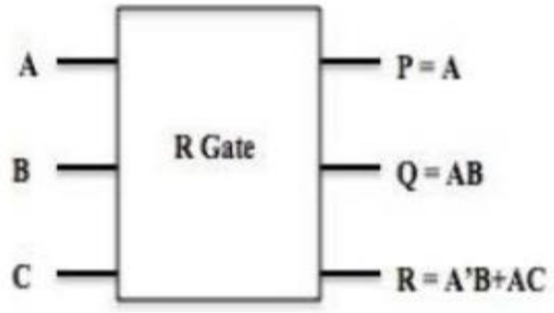


Fig 4. block diagram of R gate

In our journey through the labyrinthine world of reversible logic, we encounter the bedrock of reversibility in the form of truth tables. These tables, as exemplified in Table 1, adhere steadfastly to the principles of reversibility. They serve as the compass guiding our understanding of the digital realm.

Now, the advantages of the illustrious R-gate, when juxtaposed against its counterparts, unfurl as a tapestry of excellence. Its supremacy shines through in various facets: quantum cost, garbage output management, quantum depth, the handling of constant inputs, and the sheer number of quantum gates harnessed in its operation.

Within the realm of quantum circuits, a landscape of complexities and intricacies awaits. It is paramount to ascertain the integrity of our circuitry, and this task is entrusted to the art of fault testing. The systematic approach employed for this purpose is known as fault testing, a meticulous process that endeavors to uncover any blemishes within the circuitry.

In conventional reversible circuits, testing unfurls as a choreographed dance. Input states, carefully curated to be logically sound, are presented to the circuit. The resulting output states are meticulously measured and compared to the expected outputs of an ideal, faultless circuit. This collection of input vectors, akin to a set of musical notes, is what we refer to as the "test set."

However, should a fault emerge, the task at hand transcends mere detection; it transforms into a quest for precision. This is where fault confinement steps in, unveiling the location and nature of the fault within the circuit. We refer to the model that helps us understand these kinds of logical or functional breakdowns in the circuit as a "fault model."

Fault models have historically been focused on common types of faults, such as "stuck at faults," "bridge faults," and "delay faults," and have been carefully studied in the context of traditional irreversible circuits. There is, however, a growing need to enlarge our perspectives and the breadth of our study into test vectors in light of the recent development of reversible computing and quantum computation.

In recent times, perceptive minds have recognized this need and have embarked on a quest to craft fault models that transcend specific technologies. These contemporary fault models include:

1. Single Missing Gate Fault (SMF): The existence of a lone gate in a circuit is a rare occurrence.
2. Multiple Missing Gate Fault (MMF): Occurs when multiple gates are conspicuously absent within the circuit.
3. Repeated Gate Fault (RGF): Arises when the same gate is replicated successively, disrupting the circuit's harmony.
4. Partial Missing Gate Fault (PGF): Akin to an imperfect gate, where functionality falters.
5. Cross Point Fault: A situation in which certain gates' control points disappear while others get control points that were not intended.
6. Stuck-to Fault Model: Combining "single stuck-to fault (SSF)" for ones and "multiple stuck-to fault (MSF)" for zeros.

In this ever-evolving landscape, fault testing and the quest for fault models remain at the forefront, uncovering the mysteries and ensuring the integrity of our intricate circuits.

SIMULATION RESULTS

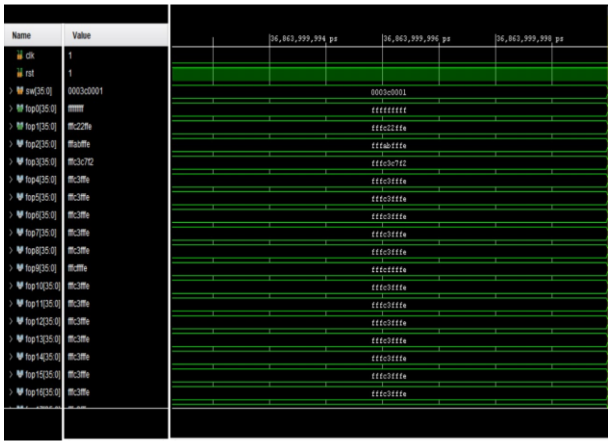

```
Name          Value        |36,863,999,994 ps  |36,863,999,996 ps  |36,863,999,998 ps
clk           1
rst           1
sw[35:0]      0003c0001    0003c0001
fop0[35:0]    fffffffff    fffffffff
fop1[35:0]    fffc22ffe    fffc22ffe
fop2[35:0]    fffabfffe    fffabfffe
fop3[35:0]    fffc3c7f2    fffc3c7f2
fop4[35:0]    fffc3fffe    fffc3fffe
fop5[35:0]    fffc3fffe    fffc3fffe
fop6[35:0]    fffc3fffe    fffc3fffe
fop7[35:0]    fffc3fffe    fffc3fffe
fop8[35:0]    fffc3fffe    fffc3fffe
fop9[35:0]    fffcffffe    fffcffffe
fop10[35:0]   fffc3fffe    fffc3fffe
fop11[35:0]   fffc3fffe    fffc3fffe
fop12[35:0]   fffc3fffe    fffc3fffe
fop13[35:0]   fffc3fffe    fffc3fffe
fop14[35:0]   fffc3fffe    fffc3fffe
fop15[35:0]   fffc3fffe    fffc3fffe
fop16[35:0]   fffc3fffe    fffc3fffe
```

Fig 5. represents simulation for mux-update

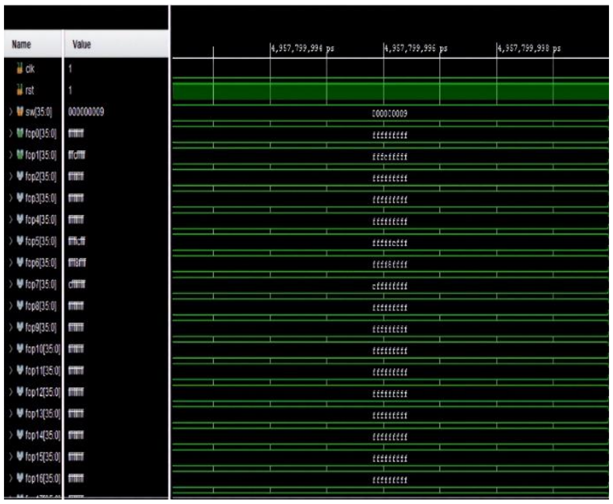
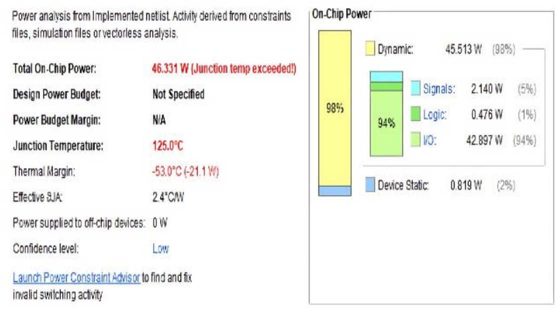


Fig 6. represents simulation for lut-update

POWER ANALYSIS OF PROPOSED MECHANISM



Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

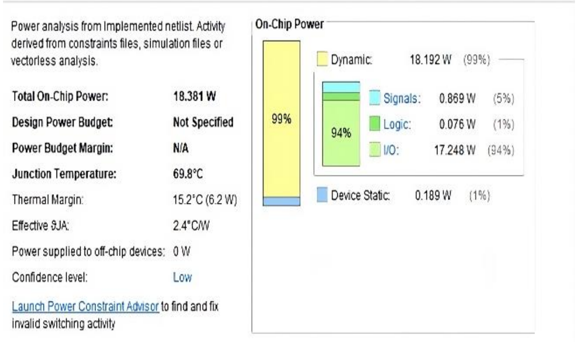
Total On-Chip Power:	46.331 W (Junction temp exceeded!)
Design Power Budget:	Not Specified
Power Budget Margin:	N/A
Junction Temperature:	125.0°C
Thermal Margin:	-53.0°C (-21.1 W)
Effective ϑJA:	2.4°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power

Dynamic:	45.513 W	(98%)
Signals:	2.140 W	(5%)
Logic:	0.476 W	(1%)
I/O:	42.897 W	(94%)
Device Static:	0.819 W	(2%)

Fig 7. power consumption of lut- update mechanism



Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	18.381 W
Design Power Budget:	Not Specified
Power Budget Margin:	N/A
Junction Temperature:	69.8°C
Thermal Margin:	15.2°C (6.2 W)
Effective ϑJA:	2.4°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power

Dynamic:	18.192 W	(99%)
Signals:	0.869 W	(5%)
Logic:	0.076 W	(1%)
I/O:	17.248 W	(94%)
Device Static:	0.189 W	(1%)

Fig 8. power consumption for proposed lut-update Mechanism

APPLICATION DEVELOPMENT

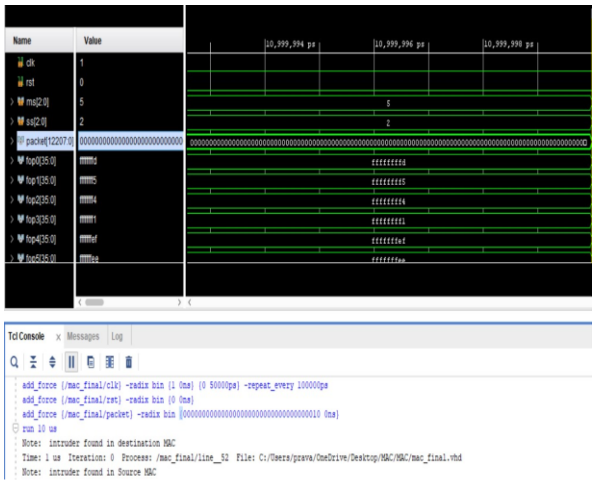
MAC

Switches on a LAN (Local Area Network) employ a MAC address table, also known as a Content Addressable Memory (CAM) table, to decide where data should be sent next. All communicating hardware devices have their own MAC address. When hardware device want to communicate with server; switch will check about Routing to server is authorisation. If source device MAC is already stored in switch MAC table, it is authorised. Ethernet protocol is used here for communication.

The address of a network interface card is given to a media access control address. the address of a media access control address is given to a media access control address is given to a network interface card. The Open Systems Interconnection network model makes use of these identifiers at the data link layer level. Routers and multilayer switches, which have several NICs, should each have a unique media access control (MAC) address. There are several applications for media access control (MAC) addresses:

- Static IP Assignment: Your IP address is a static IP address assigned to your computer.
- MAC Address Filtering: This feature is useful for networks since it restricts access to the system to devices having a predetermined MAC address.
- MAC Authentication: There are Internet service providers (ISPs) who may only let you connect devices with a certain MAC address to the web.
- Device Identification: A device's MAC address is used for identification purposes by many airport Wi-Fi networks and other public Wi-Fi networks.
- Device Tracking: Due of its unique characteristics, MAC may be used as a kind of surveillance.

RESULT



```
add_force {/mac_final/clk} -radix bin {1 0ns} {0 50000ps} -repeat_every 100000ps
add_force {/mac_final/rst} -radix bin {0 0ns}
add_force {/mac_final/packet} -radix bin {0000000000000000000000000000000000010 0ns}
run 10 us
Note:  intruder found in destination MAC
Time: 1 us  Iteration: 0  Process: /mac_final/line__52  File: C:/Users/prava/OneDrive/Desktop/MAC/MAC/mac_final.vhd
Note:  intruder found in Source MAC
```

Fig 9. represents for MAC intrusion

APPLICATIONS

In the ever-expanding realm of digital technology, a transformative tide is sweeping us into an era where the cloud reigns supreme, and data swells like an endless sea. This surge has ignited a burgeoning need for high-throughput search engines, and in this evolving landscape, Field-Programmable Gate Arrays (FPGAs) and their FPGA-based Content-Addressable Memories (CAMs) emerge as pivotal players.

Now, let's embark on a journey through the myriad applications where these remarkable CAMs leave their indelible mark. First and foremost, they lend their prowess to data compression, a crucial facet of modern computing that allows us to streamline and manage vast volumes of information with finesse.

In the microcosm of microprocessors, our trusty Translation Lookaside Buffers (TLBs) find solace in the embrace of CAMs. These buffers accelerate the retrieval of data by mapping virtual addresses to physical addresses, enhancing the overall efficiency of microprocessors.

But the influence of CAMs stretches far beyond the confines of silicon. In the enchanting realm of image recognition, they serve as the watchful eyes that decode and interpret the visual world around us, empowering machines to perceive and comprehend.

Venturing further, we find ourselves in the realm of Content-Centric Networks, where CAMs shine as beacons of efficiency. They facilitate content-based routing and retrieval, a paradigm shift in network design that places the content itself at the forefront of communication.

Yet, perhaps one of the most fascinating frontiers is Cognitive Information Processing, where CAMs contribute to the construction of intelligent systems that mimic human thought processes. These systems are poised to revolutionize decision-making, learning, and problem-solving on an unprecedented scale.

Finally, in the commercial domain, Traffic Classification Access Memory (TCAM) takes center stage within network routers. Its role is pivotal, as it diligently classifies incoming packets, swiftly and accurately determining their fate within the intricate web of network traffic.

CONCLUSION

In the grand finale of our exploration, we confront a critical concern—the prolonged update-latency of TCAM. This issue casts a shadow over the efficiency of search-based systems, a realm where Software-Defined Networks (SDNs) on FPGAs reign supreme.

However, our journey has not been in vain, for we have unveiled a beacon of hope. The techniques we have proposed breathe new life into the world of FPGA-based TCAM architecture. They hold the promise of swift TCAM entry updates, ushering in a new era of efficiency, all while judiciously managing the usage of precious Input/Output (I/O) pins.

In this crescendo of innovation, we have carved a path to a brighter future, one where TCAM architecture is endowed with an efficient update storage medium, surpassing the limitations of its predecessors. As the curtains draw to a close, we find ourselves standing at the threshold of a realm where power constraints recede, and the possibilities are boundless.

FUTURE SCOPE

In the vista of future possibilities, the world of FPGA-based CAMs unfolds as a canvas awaiting the strokes of innovation. What awaits us on this uncharted terrain is a vision of simplicity and efficiency, a harmonious marriage between Content-Addressable Memory (CAM) data and the versatile SRAM blocks.

Imagine a future where FPGA-based CAMs seamlessly translate into a simple one-to-one mapping, embracing the reconfigurability that promises to optimize hardware resources, reduce power consumption, and enhance throughput. This vision promises a future where FPGA vendors, cognizant of the growing relevance of CAMs in diverse applications, might well usher in a new era by embedding hard-core CAMs within their future devices. The aim? To supercharge the performance of searching-based applications and set new standards in the world of digital exploration.

As we stand on the precipice of the artificial intelligence revolution, the landscape of FPGA-based CAMs is poised for transformation. The boundaries of what is achievable continue to expand, unlocking new frontiers where the convergence of innovation and technology knows no limits.

REFERENCES

[1] AGRAWAL, B., AND SHERWOOD, T. Ternary CAM Power and Defer Model: Augmentations and

Utilizations. IEEE Trans. on VLSI Frameworks 16, 5 (May 2008).
[2] ARCANGELI, A., EIDUS, I., AND WRIGHT, C. Expanding memory thickness by utilizing ksm. OLS (2009).
[3] BANDI, N., SCHNEIDER, S., AGRAWAL, D., AND ABBADI, A. E. Equipment Speed increase of Data set Activities Utilizing Content Addressable Recollections. DaMoN (2005).
[4] BANDI, N., SCHNEIDER, S., AGRAWAL, D., AND ABBADI, A. E. Quick Information Stream Calculations utilizing Cooperative Recollections. SIGMOD (2007).
[5] BISWAS, S., FRANKLIN, D., SAVAGE, A., DIXON, R., SHERWOOD, T., AND CHONG, F. T. Multi-Execution: Multicore Reserving for Information Comparative Executions. ISCA (2010).
[6] GOEL, A., AND GUPTA, P. Little Subset Questions and Sprout Channels Utilizing Ternary Cooperative Recollections, with Applications. SIGMETRICS (2010).
[7] JACOB, B., NG, S. W., AND WANG, D. T. Memory frameworks: Store, measure, circle. Elsevier Inc. (2008).
[8] KOLLER, R., AND RANGASWAMI, R. I/o deduplication: Using content closeness to further develop I/o execution. Quick (2010).
[9] MEINERS, C. R., PATEL, J., NORIGE, E., TORNG, E., AND LIU, A. X. Quick Ordinary Articulation Matching involving Little TCAMs for Organization Interruption Identification and Anticipation Frameworks. USENIX ATC (2010).
[10] PAGIAMTZIS, K., AND SHEIKHOLESLAMI, A. Content Addressable Memory (CAM) Circuits and Models: An Instructional exercise and Review. IEEE Diary of Strong State Circuits 41, 3 (Blemish. 2006).
[11] SHINDE, R., GOEL, A., GUPTA, P., AND DUTTA, D. Closeness Search and Area Touchy Hashing utilizing Ternary Substance Addressable Recollections. SIGMOD (2010).
[12] TIWARI, M., AGRAWAL, B., MYSORE, S., VALAMEHR, J. K., AND SHERWOOD, T. A Little Reserve of Huge Reaches: Equipment Techniques for Effectively Looking, Putting away, and Refreshing Large Dataflow Labels. Miniature (2008).
[13] U. Sikder and T.- J. K. Liu, "Plan streamlining for NEM transfers carried out in BEOL layers," in 2017 IEEE SOI-3D-Subthreshold Microelectronics Innovation Bound together Gathering (S3S), 2017, pp. 1-3.
[14] H. Zhong et al., "A single Shot Revive: A Low-Power Low-Clog Approach for Dynamic Recollections," in IEEE Trans. Circuits Syst. II Express Briefs, vol. 67, no. 12, pp. 3402-3406, 2020.
[15] C. Chen et al., "Productive FPGAs utilizing nanoelectromechanical transfers," in Procedures of the eighteenth yearly ACM/SIGDA global conference on Field programmable entryway exhibits, 2010, pp. 273-282.
[16] S. Chong et al., "Nanoelectromechanical (NEM) transfers coordinated with CMOS SRAM for further developed security and low spillage," in 2009 Global Meeting on PC Helped Plan, 2009, pp. 478-484.
[17] X. Huang et al., "A nanoelectromechanical-switch-based warm administration for three dimensional incorporated many-center memory-processor framework," IEEE Trans. Nanotechnol., vol. 11, no. 3, pp. 588-600, 2012.
[18] H. Zhong, M. Gu, J. Wu, H. Yang, and X. Li, "Plan of almostnonvolatile implanted Measure utilizing nanoelectromechanical transfer gadgets," in 2020 Plan, Mechanization and Test in Europe Meeting and Presentation (DATE), 2020, pp. 1223-1228.
[19] R. Vattikonda, W. Wang, and Y. Cao, "Displaying and minimization of PMOS NBTI impact for powerful nanometer plan," in 2006 43rd ACM/IEEE Plan Robotization Gathering, 2006, pp. 1047-1052.
[20] M. A. Lastras-Montano et al, "Architecting energy productive crossbarbased memristive irregular access recollections," in Procedures of the 2015 IEEE/ACM NANOARCH' 15, 2015, pp. 1-6