G-AETCAM
Gate-Based Area-Efficient Ternary Content-Addressable Memory on FPGA
IRFAN, Muhammad; ULLAH, Zahid

Published in:
IEEE Access

Published: 01/01/2017

Document Version:
Final Published version, also known as Publisher's PDF, Publisher's Final version or Version of Record

Publication record in CityU Scholars:
Go to record

Published version (DOI):
10.1109/ACCESS.2017.2756702

Publication details:

Citing this paper
Please note that where the full-text provided on CityU Scholars is the Post-print version (also known as Accepted Author Manuscript, Peer-reviewed or Author Final version), it may differ from the Final Published version. When citing, ensure that you check and use the publisher's definitive version for pagination and other details.

General rights
Copyright for the publications made accessible via the CityU Scholars portal is retained by the author(s) and/or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights. Users may not further distribute the material or use it for any profit-making activity or commercial gain.

Publisher permission
Permission for previously published items are in accordance with publisher's copyright policies sourced from the SHERPA RoMEO database. Links to full text versions (either Published or Post-print) are only available if corresponding publishers allow open access.

Take down policy
Contact lbscholars@cityu.edu.hk if you believe that this document breaches copyright and provide us with details. We will remove access to the work immediately and investigate your claim.

Download date: 18/12/2022
G-AETCAM: Gate-Based Area-Efficient Ternary Content-Addressable Memory on FPGA

MUHAMMAD IRFAN AND ZAHID ULLAH, (Member, IEEE)
Department of Electrical Engineering, CECOS University of IT and Emerging Sciences, Peshawar 25100, Pakistan

Corresponding author: Muhammad Irfan (irfan2inform@gmail.com)

ABSTRACT
This paper presents a novel architecture for ternary content-addressable memory (TCAM), using G-AETCAM cells, which outputs the address of the provided input data. The proposed architecture is a matrix of G-AETCAM cells arranged in the form of rows and columns using flip-flop as a memory element and a control logic circuitry consisting of logic gates. One G-AETCAM cell encodes the input and stored bit into one encoded bit which results in a match-line after passing from the AND-gate-array. Many architectures configure random-access memory (RAM) available on FPGAs as content-addressable storage architecture but are efficient for specific size and deal with data only in ascending or descending order while the proposed architecture provides freedom in size with no chance of making a single memory cell as useless and can store ternary data in any order. RAM-based TCAMs require pre-processing for the storage of TCAM words while the proposed design does not involve any pre-processing. The proposed architecture reduces the transistor count by a factor of 25.55, as compared with the RAM-based TCAM, which ultimately reduces area and increases the speed of operations. The proposed architecture is successfully implemented on Xilinx Virtex-6 FPGA for the size $64 \times 36$ and achieves a speed of 358 MHz.

INDEX TERMS Content-addressable storage, field-programmable gate arrays, flip-flops, logic gates, memory, RAM-based TCAM.

I. INTRODUCTION
Ternary content-addressable memory (TCAM) provides the address for the input data in one and only one clock cycle [1], [2]. It is efficient for high speed lookup applications where the delay between two events, the time at which the input is given and the time at which the desired data (address) appears at the output, matters. It is different from the typical random-access memory (RAM) in two ways. One is that it returns address rather than data and second is that additional to storing of ‘1’ (logic one) and ‘0’ (logic zero) it can store X-bit which is known as don’t-care bit. This don’t-care bit results the comparison with both ‘1’ as well as ‘0’ as a match.

CAMs have a wide range of applications which involve concurrent processing of data to attain high speed of operations. It can be used to accelerate any application ranging from network routing, database management, positioning system [3], pattern recognition to processor specific cache memories [4] and even in graph similarity search [5]. Binary content-addressable memories (BiCAMs) as well as TCAMs are already applied as storing media in the field of artificial intelligence, radar signals tracking, bioinformatics, image processing and many others [6].

The rest of the paper is organized as follows: Section II explains the related previous work. Section III provides the motivation and key contributions of the proposed architecture. Section IV explains the G-AETCAM architecture, its generic form and algorithms for storage as well as search operation. Section V describes the area consumption (transistor count) of the proposed architecture with reference to other architectures. Section VI is about FPGA implementation and results of G-AETCAM architecture. Section VII has conclusions and directions about the future work.
The architecture presented in patent [8] also requires pre-processing of the CAM data which is time consuming and the optimal partitioning scheme is also not available. CAM based on hashing technique [9] has the drawback of bucket overflow, collision as well as the re-hashing process is time consuming.

CAMs proposed in [10]–[13] use the memory blocks that are available on field-programmable gate array (FPGA) with some logic blocks but all of these require pre-processing of the CAM data to be stored in the RAM blocks. These Block RAM (BRAM) based CAMs are good for specific size configuration of the CAM and TCAM memory but in other configurations, it leaves some of the static random-access memory (SRAM) cells as useless. This is just because the BRAM of the FPGA is fixed in size (18K and 36K) [14]. For instance, if we need 82K SRAM memory, two blocks of 36K BRAM will be declared while for further 10K, one block of 18K BRAM will be declared. In this case 8K SRAM cells are wasted by keeping the fact that for large memory distributed RAM is not efficient.

To summarize, the proposed architecture is beneficial in comparison with the BRAM based implementation as it does not leave any cell as useless, is beneficial in comparison with the distributed RAM based implementation as it does not require any sort of pre-processing or partitioning scheme and is beneficial in comparison with both RAM based implementation as it deals with the CAM data in any (ascending or descending) order. TCAM presented in [15] also require architectural elements for configuring of bit position table, validation memory and address position address generator while the proposed architecture is free from this sort of pre-processing and overhead.

A similar design to our proposed architecture is presented in [16] about BiCAM. It does not deal with the third storage case which is an X-bit (don’t-care). It is also area-efficient with reference to the RAM-based architectures but compared to our proposed architecture, it is a BiCAM rather than a TCAM.

III. MOTIVATION AND CONTRIBUTIONS

A. MOTIVATION

Conventional TCAM architectures are implemented as a standalone application-specific integrated circuit (ASIC) which are expensive and not scalable as compared to the RAM-based TCAM architectures [10]. Thus there is a need to introduce an architecture that could be used on FPGAs which has higher configurability, parallel computation [17], low to market time, near-ASIC performance and high clock speeds. Researchers proposed several RAM-based architectures for TCAM as well as BiCAM but almost all of it require pre-processing for storing of the TCAM bits which consumes time as well as hardware resources.

In our paper we have focussed on saving chip area which ultimately saves the power, cost and increases speed. We propose a novel architecture, with transistor count less than the previous architectures, for TCAM which uses different media for storage of the memory bit and gate logic for comparison of the stored bit and input bit. Our propose design is technically similar to the typical NOR CAM [1] at Gate level and does not require any pre-processing as well as can deal with the data in any (ascending/descending) order.

B. KEY CONTRIBUTIONS

Key contributions of the proposed architecture compared to the already available and discussed designs are:

- Our proposed TCAM architecture can handle the stored data in any ORDER. Unlike the RAM-based TCAMs, no pre-processing is needed for its correct functionality.
- Unlike the previous work, the proposed TCAM does not involve any sort of partitioning which is an overhead on the whole system. The proposed architecture saves time, hardware resources as well as area by skipping the partitioning scheme that is a part of the previous architectures.
- The number of transistors compared to the RAM-based TCAM is reduced by a factor of 25.55 which saves area in terms of hardware resources and that is why we call our architecture as Area-Efficient.
- The proposed architecture provides freedom in configuring the size (Depth, m × Width, n) of TCAM memory, unlike the RAM based CAMs which leaves some of the RAM cells, if configured using BRAMs (not distributed RAM, which is inefficient for large sizes), as useless as previously discussed in section II.

IV. PROPOSED ARCHITECTURE

A. TERMINOLOGY

Some terminologies that are used in explaining the proposed architecture are given in Table 1.

TABLE 1. Different terminologies (notations).

<table>
<thead>
<tr>
<th>Notation</th>
<th>Terminology</th>
</tr>
</thead>
<tbody>
<tr>
<td>m</td>
<td>Depth of the G-AETCAM memory</td>
</tr>
<tr>
<td>n</td>
<td>Width of the G-AETCAM memory</td>
</tr>
<tr>
<td>M</td>
<td>Position in a row. It varies from 1 to m</td>
</tr>
<tr>
<td>N</td>
<td>Position in a Column. It varies from 1 to n</td>
</tr>
<tr>
<td>C_gate</td>
<td>Comparison Gate</td>
</tr>
<tr>
<td>M_gate</td>
<td>Masking Gate</td>
</tr>
<tr>
<td>St_w[M][N]</td>
<td>One bit of the stored word in row M and column N</td>
</tr>
<tr>
<td>S_w[N]</td>
<td>One bit of Search word at location N</td>
</tr>
<tr>
<td>E_w[M][N]</td>
<td>One bit of the Encoded word in row M and column N</td>
</tr>
<tr>
<td>G-AETCAM_cell[M][N]</td>
<td>G-AETCAM cell in row M and column N</td>
</tr>
</tbody>
</table>

B. G-AETCAM MATRIX

Gate-based area-efficient TCAM (G-AETCAM) architecture is a matrix of cells known as G-AETCAM cells arranged in the form of rows and columns. Fig. 1 shows a G-AETCAM cell consisting of a Masking Element (M_el), a Storage Element (St_el), a Comparison gate (C_gate) and a Masking
One G-AETCAM cell. (En_w: Encoded word, S_w: Search word).

A 4x4 G-AETCAM architecture.

A 4x4 G-AETCAM architecture with 16 G-AETCAM cells having four rows and four columns is shown in Fig. 2.

AND-gate-array consists of four 4-input AND gates. The output from the four G-AETCAM cells in a single row is fed to the corresponding AND gate of the AND-gate-array which gives a match-line (M_L) as output. When a 4-bit S_w is given at the input, first bit (S_w[1]) will be given as input to all of the G-AETCAM cells in first column, second (S_w[2]) in second, third (S_w[3]) in third, and fourth bit (S_w[4]) will be given as input to all of the G-AETCAM cells in fourth column of the G-AETCAM matrix. A match-line results from the bit-wise ANDing of all four bits (En_w[M][1], En_w[M][2], En_w[M][3] and En_w[M][4]) of the M^{th} Encoded word (En_w[M]).

One match-line (M_L[M]) represents a MATCH or MISMATCH of the Search word with the corresponding row or ultimately with the corresponding stored word. M varies from 1 to 4 in this case. M_L[1] represents match-line for the first row, M_L[2] represents match-line for the second row, M_L[3] represents match-line for the third row and M_L[4] represents match-line for the fourth row of the G-AETCAM memory. These four match-lines serve as INPUT to the priority encoder.

C. GENERALIZATION OF THE PROPOSED ARCHITECTURE

An m\times n G-AETCAM architecture is proposed in which m represents the number of stored words in the TCAM memory while n represents the number of bits in each word as shown in Fig. 3. In other words, m shows the depth while n shows the width of the TCAM memory. The number of G-AETCAM cells will be the product of m and n. The number of AND gates in the AND-gate-array will be m with n inputs. These n inputs are the n bits of each En_w. The Encoded words are also m in number. The output of each AND gate is a match-line which are m in number. The m match-lines will be provided to a priority encoder, which results the address of the input Search word in the proposed memory architecture.

D. POPULATING G-AETCAM

TCAM_bit[i] represents the virtual bit of the conventional TCAM memory. TCAM_bit[i] could be ‘1’, ‘0’ or ‘X’.
Algorithm 1 Populating G-AETCAM
Input: TCAM_bit (One bit from conventional TCAM) 
Output: St_el_bit (One bit value that needs to be stored in Storage Element) 
        M_el_bit (One bit value that needs to be stored in Masking Element) 
1: for i ← 1 to i ← N_t do 
2:     if TCAM_bit[i] = ‘0’ then 
3:         St_el_bit[i] = ‘0’ 
4:         M_el_bit[i] = ‘0’ 
5:     else if TCAM_bit[i] = ‘1’ then 
6:         St_el_bit[i] = ‘1’ 
7:         M_el_bit[i] = ‘0’ 
8:     else if TCAM_bit[i] = ‘X’ then 
9:         St_el_bit[i] = ‘1’ 
10:        M_el_bit[i] = ‘1’ // St_el_bit[i] = ‘0’ 
11: end if 
12: end for 

N_t = m × n = Total number of bits in TCAM. 
m: Number of TCAM words. 
n: Number of bits per TCAM word.

St_el_bit[i] and M_el_bit[i] are the corresponding bit value (‘1’ or ‘0’) of St_el and M_el that needs to be stored physically in the G-AETCAM memory. Algorithm 1 explains the populating of G-AETCAM cells with ‘1’, ‘0’ and ‘X’ bits. If the TCAM_bit is ‘1’, place ‘1’ in St_el and ‘0’ in M_el. If the TCAM_bit is ‘0’, place ‘0’ in St_el and ‘0’ in M_el. What if the TCAM_bit is ‘X’ (a don’t-care bit)?

To store a don’t-care bit in the G-AETCAM cell, place ‘1’ or ‘0’ in the corresponding St_el flip-flop and ‘1’ in the corresponding M_el flip-flop. This configuration will make ‘1’ as well as ‘0’ as MATCH to the corresponding bit of the input Search word.

Algorithm 2 Search Operation in G-AETCAM
Input: S_w[N] (Search word; N starts from 1 to n) 
Output: E_w[M][N] (Encoded word; M starts from 1 to m while N starts from 1 to n) 
1: for M ← 1 to M ← m do 
2:     for N ← 1 to N ← n do 
3:         E_w[M][N] = M_el[M][N] or ( St_el[M][N] xnor S_w[N] ) 
4:     end for 
5: end for 

E. SEARCHING G-AETCAM
The S_w is applied to each row of the G-AETCAM matrix simultaneously. In each row, the Nth bit of the Search word (S_w[N]) is provided to the Nth G-AETCAM cell as input. Algorithm 2 explains the search operation of the G-AETCAM memory in which line number 3 and 4 is one instruction representing the control logic of each G-AETCAM cell.

F. AN EXAMPLE
Fig. 4 shows a G-AETCAM architecture having conventional TCAM data given in Table 2 with their corresponding addresses. Each G-AETCAM cell has two one-bit values. One (on the left) shows the bit value stored in M_el and other (on the right) shows the bit value stored in St_el. M_el of the G-AETCAM cell in row number 2 and column number 2 (G-AETCAM_cell[2][2]) stores ‘1’, which makes the bit value at that location as don’t-care bit. Now for input S_w equal to “0 0 1 0” or “0 1 1 0”, M_L[2] will be high. Therefore both input data values will be found at location “0 1” by the said G-AETCAM architecture. Here one can see the ORDER of the stored data in G-AETCAM memory, which does not matter for the correct functionality of the proposed architecture.

V. TRANSISTOR COUNT
Let’s compare the 64×36 configuration of the proposed TCAM architecture with the latest TCAM architecture (UE-TCAM) in terms of transistor count. Transistor count n_T is the number of transistors used by the whole architecture. UE-TCAM proposed in [13], if configured as 64×36 TCAM, use 16 BRAMs of size 18K using Virtex-6 FPGA [16]. The total number of transistors used by the UE-TCAM architecture will be:

\[
n_{TUE} = 16 \times 18K \text{ SRAM cells} 
\]

\[
= 294912 \text{ SRAM cells} 
\]

\[
n_{TUE} = 1769472 \text{ transistors} 
\]
The number of transistors per SRAM cell are 6 [18]. The number of transistors used by UE-TCAM architecture is even more, as we are neglecting the logic circuitry in transistor count and taking only BRAMs into account.

The proposed architecture (G-AETCAM) of the same size consists of 4608 flip-flops, 2304 2-input XNOR gates, 2304 2-input OR gates and 64 36-input AND gates. The number of transistors in one flip-flop is 8, in one 2-input XNOR gate is 6, in one 2-input OR gate is 6 and in one 36-input AND gate is 74 [19]. The total number of transistors for the proposed architecture will be:

$$n_{T_{G-AE}} = (4608 \times 8) + 2 \times (2304 \times 6) + (64 \times 74) \text{ transistors}$$

$$n_{T_{G-AE}} = 69248 \text{ transistors}$$

To summarize, the latest TCAM architecture of size 64×36 use 1769472 transistors while the proposed architecture (G-AETCAM) use 69248 transistors. The decrease factor (DF) is the ratio of the number of transistors in previous architecture and the number of transistors in the proposed architecture which is 25.55 in case of UE-TCAM.

$$DF = n_{T_{UE}} / n_{T_{G-AE}}$$

$$DF = 25.55$$

Fig. 5 shows a graph of transistor count vs the previous TCAM architectures as well as the proposed G-AETCAM architecture. It clearly shows the decrease in number of transistors by the proposed architecture in comparison with the previous architectures. Table 3 shows the DF of the proposed architecture in area (transistor count) in comparison with other architectures.

### VI. FPGA IMPLEMENTATION AND RESULTS

The proposed architecture has been implemented successfully on Xilinx FPGA using VHDL (VHSIC Hardware Description Language) for the size configuration of 63×36, 64×36, 63×35 and 64×35. The one bit difference in depth as well as width is kept to support our claim that no memory cell will be left as useless in any configuration which is not true if the same configurations are implemented using BRAMs available on FPGA.

Some BiCAM as well as TCAM architectures presented previously like [7] could not be implemented on FPGA for the sample size of 64×36. We have done the implementation on FPGA to support our claim that our architecture can be successfully implemented on FPGA. Comparison in terms of hardware resources on FPGA would not be fair as we use different memory element while comparison in terms of transistor count results in considerable decrease factor which is already discussed in section V.

The FPGA implementation results, for the size 64×36 on Xilinx Virtex-6 FPGA device XC6VLX760 with −2 speed grade, are summarized in Table 4. Performance was evaluated on the basis of Place & Route results generated from the Xilinx ISE Design tool 14.5. Xilinx Power Analyzer tool is used to find the dynamic power consumption at 100 MHz with 1.0 V core voltage. The proposed architecture itself does not include a priority encoder. G-AETCAM architecture in 64×36 configuration utilizes only 1% of the hardware resources (SRs and LUTs) on the said FPGA device which is the beauty of our work.

### VII. CONCLUSION AND FUTURE WORK

We have opened a new area in the field of TCAM by proposing an architecture which uses a different storage media than the existing TCAMs and has a decrease factor of 25.55 in terms of transistor count with reference to the existing efficient TCAM (Ultra-efficient TCAM). Future work includes exploring scalability of G-AETCAM architecture and further reduction of the hardware resource usage on FPGA. Implementing clock gating or partitioning of the G-AETCAM architecture to decrease power consumption would be another direction of research in this field.
REFERENCES


MUHAMMAD IRFAN received the B.Sc. degree in electrical engineering from the University of Engineering and Technology, Peshawar, Pakistan, in 2013, and the M.S. degree in electrical engineering from the Lahore University of Management Sciences, Lahore, Pakistan, in 2016, respectively. During his M.S. studies, he was a Teaching Assistant for two consecutive semesters at the Lahore University of Management Sciences. He is currently with the Department of Electrical Engineering, CECOS University of IT and Emerging Sciences, Peshawar. His research interest includes the FPGA-based designs such as BiCAM and TCAM architectures.

ZAHID ULLAH (M’16) received the B.Sc. degree (Hons.) in computer system engineering from the University of Engineering and Technology, Peshawar, Pakistan, in 2006, the M.S. degree in electronic, electrical, control, and instrumentation engineering from Hanyang University, South Korea, in 2010, and the Ph.D. degree in electronic engineering from the City University of Hong Kong, Hong Kong, in 2014. He is currently an Assistant Professor with the Department of Electrical Engineering, CECOS University of IT and Emerging Sciences, Peshawar. He has over ten years of teaching and research experience. He has authored prestigious journal and conference papers, and patents in his name in SRAM-based TCAM. His research includes low-power/high-speed CAM design on FPGA, low-power/high-speed VLSI design, and embedded systems.

***