IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received February 22, 2018, accepted March 28, 2018, date of publication April 3, 2018, date of current version April 25, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2822311

Efficient TCAM Design Based on Multipumping-Enabled Multiported SRAM on FPGA

INAYAT ULLAH[1], ZAHID ULLAH[2], (Member, IEEE), AND JEONG-A LEE[1], (Senior Member, IEEE)
[1]Department of Computer Engineering, Chosun University, Gwangju 61452, South Korea
[2]Department of Electrical Engineering, CECOS university of IT & Emerging Sciences, Peshawar 25000, Pakistan

Corresponding author: Jeong-A Lee (jalee@chosun.ac.kr)

This work was supported in part by the National Research Foundation of Korea through the Ministry of Science and ICT under Grant NRF-2016R1A2B4010382 and in part by the Korea Institute of Energy Technology Evaluation and Planning and Ministry of Trade, Industry and Energy of the Republic of Korea under Grant 20164010201020.

ABSTRACT Ternary content-addressable memory (TCAM)-based search engines play an important role in networking routers. The search space demands of TCAM applications are constantly rising. However, existing realizations of TCAM on field-programmable gate arrays (FPGAs) suffer from storage inefficiency. This paper presents a multipumping-enabled multiported SRAM-based TCAM design on FPGA, to achieve an efficient utilization of SRAM memory. Existing SRAM-based solutions for TCAM reduce the impact of the increase in the traditional TCAM pattern width from an exponential growth in memory usage to a linear one using cascaded block RAMs (BRAMs) on FPGA. However, BRAMs on state-of-the-art FPGAs have a minimum depth limitation, which limits the storage efficiency for TCAM bits. Our proposed solution avoids this limitation by mapping the traditional TCAM table divisions to shallow sub-blocks of the configured BRAMs, thus achieving a memory-efficient TCAM memory design. The proposed solution operates the configured simple dual-port BRAMs of the design as multiported SRAM using the multipumping technique, by clocking them with a higher internal clock frequency to access the sub-blocks of the BRAM in one system cycle. We implemented our proposed design on a Virtex-6 xc6vlx760 FPGA device. Compared with existing FPGA-based TCAM designs, our proposed method achieves up to 2.85 times better performance per memory.

INDEX TERMS Block RAM (BRAM), field-programmable gate array (FPGA), memory architecture, multiported memory, multipumping, SRAM-based TCAM.

I. INTRODUCTION

Ternary content-addressable memory (TCAM) compares an input word with its entire stored data in parallel, and outputs the matched word's address. TCAM stores data in three states: 0, 1, and X (don't care). Traditional TCAMs are built in application-specific integrated circuit (ASIC), and offer high-speed search operations in a deterministic time.

TCAM is widely employed to design high-speed search engines and has applications in networking, artificial-intelligence, data compression, radar signal tracking, pattern matching in virus-detection, gene pattern searching in bioinformatics, image processing, and to accelerate various database search primitives [1]–[3]. The Internet-of-things and big-data processing devices employ TCAM as a filter when storing signature patterns, and achieve a substantial reduction in energy consumption by reducing wireless data transmissions of invalid data to cloud servers [4], [5].

Field-programmable gate arrays (FPGAs) emulate TCAM using static random-access memory (SRAM), by addressing SRAM with TCAM contents. Each SRAM word corresponds to a specific TCAM pattern, and stores information on its existence for all possible data of the TCAM table. The increase in the number of TCAM pattern bits results in an exponential growth in memory usage. This exponential growth in memory usage has been reduced to linear growth by cascading multiple SRAM blocks in the design of TCAM on FPGA in previous work [6], [7].

Contemporary FPGAs implement block-RAM (BRAM) in the silicon substrate, and offer a high speed. For example, Xilinx Virtex-6 xc6vlx760 FPGA contains 720 BRAMs of size 36 Kb [8], and provide operating frequencies of greater than 500 MHz [9]. Designers utilize these high-speed SRAM blocks to design SRAM-based TCAMs on FPGA.

In existing SRAM-based solutions, the storage capacity of a BRAM for TCAM bits is limited by its higher SRAM/TCAM ratio $\frac{2^9}{9}$, because of its minimum depth limitation of 512 × 72 when configured in simple dual-port mode on FPGA [8]. For example, the design methodologies proposed in [10], [11], and [12], require a total of 56, 40, and 40 BRAMs of size 36 Kb, respectively, to implement an 18 Kb TCAM.

Excessive usage of BRAMs in the design of TCAM can result in a lack of BRAMs for other parts of the system on FPGA. Furthermore, the limited amount of BRAM resources on FPGA can compel designers to implement TCAMs in distributed RAM using SLICEM, resulting in the consumption of many slices, and a limitation on the maximum clock frequency of the design. This problem becomes more severe for the design of large storage capacity TCAMs. The efficient utilization of SRAM memory is imperative for the design of TCAMs on FPGAs.

The design of memory-efficient TCAMs requires shallow SRAM blocks on FPGAs. Multipumping-based multiported SRAM emulates the sub-blocks of a dual port SRAM block as multiple shallow SRAM blocks, by operating SRAM with a higher frequency clock, allowing access to its sub-blocks in one system cycle. Researchers have designed efficient multiported memories using BRAMs on FPGA [13]–[16].

Existing FPGA-based TCAM design methodologies offer lower operational frequencies. This is mainly because of the complex wide signals routing between BRAMs and logic resulting from excessive usage of BRAMs and complex priority encoding units synthesized in logic slices for deeper traditional TCAMs. For example, the FPGA realizations of TCAM using BRAMs in [7] and [17] achieve operational frequencies of 139 MHz and 133 MHz to emulate 150 Kb and 89 Kb TCAMs, respectively. The highest operational frequency achieved in the previous studies [6], [10]–[12] is 202 MHz for the implementation of an 18 Kb TCAM on FPGA.

The demand for efficient utilization of SRAM memory in the design of TCAM and the speed provided by existing FPGA-based TCAM solutions make the use of multipumping based multiported SRAM more practical for designing TCAM memory on FPGA. Our proposed TCAM design aims to achieve efficient memory utilization with a high throughput.

The contributions of this work are as follows:

- A novel multipumping-enabled multiported SRAM-based TCAM architecture, which achieves efficient memory utilization, is proposed.
- Our proposed approach presents a scalable and modular TCAM design on FPGA.

TABLE 1. List of basic notations used.

Notations	Description
D	Depth of traditional TCAM
W	Width of traditional TCAM
R_D	Depth of the configured SRAM blocks
R_W	Width of the configured SRAM blocks
$log_2(R_D)$	Address bits of the SRAM block
P	Number of sub-blocks in an SRAM block / Multipumping factor
R_D/P	Depth of the sub-blocks in an SRAM block
M	Rows of the TCAM divisions / Rows of the TCAM memory units
N	Columns of the TCAM divisions / Columns of the TCAM memory units

- The proposed design is more practical for large storage capacities, owing to the reduced routing complexity achieved by the use of fewer BRAMs and the reduced *AND* operation complexity. The novel optimization technique of *AND*-accumulating SRAM words in the proposed TCAM memory units divides the overall *AND* operation complexity of the design.
- The proposed design is implemented on a state-of-the-art FPGA. A detailed comparison of our proposed design with existing methods is performed with respect to the performance per memory. Our proposed design achieves a performance that is up to 2.85× higher per memory.

The remainder of this paper is organized as follows. Section II surveys related work. The proposed design is described in Section III. Section IV details the implementation setup and results of this work. The performance evaluation of the proposed design is detailed in Section V. Section VI concludes this work. Table 1 describes the basic notations used in paper.

II. RELATED WORK

The CAM design methodologies presented in [18] and [19] are based on the hashing technique, which has the inherent drawback of bucket overflow. Moreover, when implemented in hardware this has an expensive overhead from re-hashing.

The CAM designs presented in [20] and [21] suffer from inefficient memory usage. The increase in pattern width results in an exponential growth in memory usage, thus making them infeasible for implementation in hardware. Our proposed solution reduces this growth to linear, as the wide pattern TCAMs are implemented by cascading BRAMs on FPGA.

The SRAM-based TCAMs presented in [10]–[12] store the TCAM presence and address information in separate BRAMs on FPGAs, resulting in an excessive usage of BRAMs. Our proposed design stores the TCAM presence and address information in the same BRAM, thus efficient memory utilization.

Xilinx presented two types of FPGA applications in [22]: a CAM design using BRAM resources and a TCAM design using the shift register (SRLE16). The first application emulates CAM rather than TCAM, and suffers from higher

SRAM memory usage. The second application consumes one 16-bit shift register look-up table (SRL16E) of SLICEM resources on FPGA to emulate every two bits of a TCAM table. Its implementation for large storage capacity designs suffers from routing and timing problems. Our proposed design has a reduced routing complexity for TCAM designs with large storage capacities, because of its lower usage of BRAMs and reduced *AND* operation complexity.

Recently binary CAM and TCAM designs built using logic resources (SLICEL) on FPGA are presented in [23] and [24] respectively. Practically the TCAM implemented using logic resources on FPGA would be of limited storage capacity, owing to the routing congestion and timing challenges. Moreover, the update of data in a TCAM design built using look-up tables (LUTs) is slow compared with SRAM-based TCAMs and requires hardware overhead of dynamic partial reconfiguration controller [25].

A hierarchical search scheme on FPGA is presented for SRAM-based CAM in [17], which reduces its average power consumption by stopping subsequent search operations if a match is found in the previous SRAM block. However, in the worst-case scenario all SRAM blocks are searched. Thus, the worst-case power consumption remains high. The FPGA realization of TCAM presented in [26] stores the TCAM word presence and address information separately in Xilinx distributed RAM and BRAM, respectively. This reduces the average power consumption of the design, as the look-up in BRAMs is avoided if a match is not found in the distributed RAM. However, the worst-case power consumption remains high, with a lower overall system throughput.

The FPGA realizations of TCAM presented in [6], [7], and [17] store the presence and address information of TCAM words in the same SRAM block. However, this approach suffers from higher SRAM memory utilization due to the limited TCAM bits storage capacity of BRAMs resulting from the minimum depth limitation on its configuration in FPGAs.

Our proposed TCAM design exploits the efficient utilization of SRAM memory by mapping TCAM divisions to shallow sub-blocks of BRAMs on FPGA. Furthermore, it operates high-speed BRAMs in the design as multipumping-enabled multiported SRAM, maintaining a high system throughput.

III. PROPOSED DESIGN

A. MULTIPUMPING-ENABLED MULTIPORTED SRAM

The multipumping technique multiplies the ports of a dual ported SRAM block by internally clocking it at an integral multiple of the external system clock [13], [15], [16], [27]. The addresses and data are registered and provided access to the SRAM block in a circular order by using mod P counter bits as shown in Figure 1. Several designs utilize multipumping for the implementation of efficient multiported memory [28]–[30].

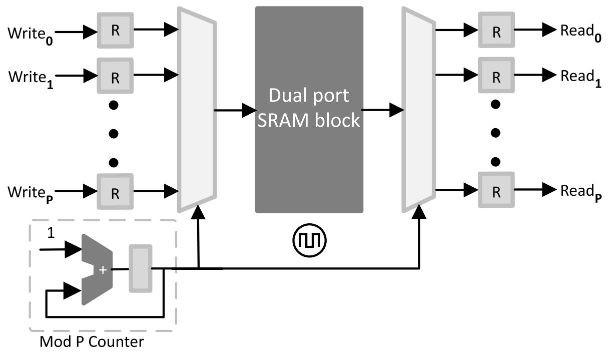

FIGURE 1. Multipumping-based multiported memory: the SRAM block is clocked at an integral multiple of *P*, allowing *P* access during one external clock cycle.

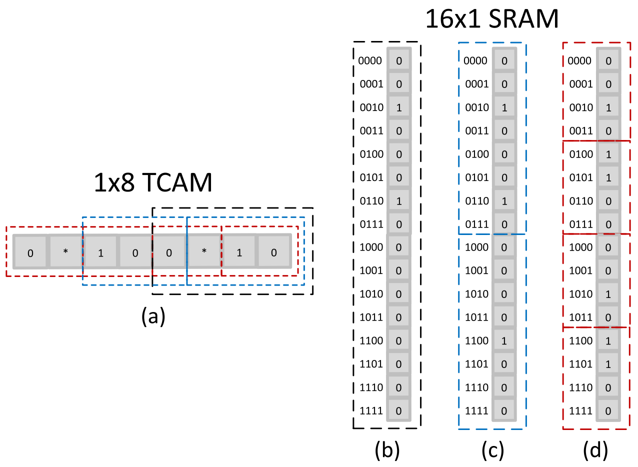

FIGURE 2. (a) A conventional TCAM of 1 × 8; (b) An 16 × 1 SRAM without multipumping emulating 1 × 4 TCAM; (c) An 16 × 1 SRAM with a multipumping factor of *P* = 2 emulating 1 × 6 TCAM; (d) An 16 × 1 SRAM with a multipumping factor of *P* = 4 emulating 1 × 8 TCAM.

B. BASIC IDEA

In the SRAM-based implementation of TCAM, the depth of the traditional TCAM determines the width of SRAM memory, and the width of the traditional TCAM is encoded as the address of the SRAM memory. The basic concept of the proposed multipumped SRAM-based TCAM implementation achieving increased memory efficiency is shown in Figure 2. Figure 2(a) shows a 1 × 8 traditional TCAM table, and Figure 2(b) shows the implementation of the four TCAM bits (0*10) by using a 16×1 SRAM block. Figure 2(c) shows the implementation of six TCAM bits (100*10) by using 16 × 1 SRAM block, which has been multipumped two times, each SRAM sub-block of size 8 × 1 emulating three TCAM bits. Figure 2(d) shows the implementation of eight TCAM bits (0*1000*10) by using 16 × 1 SRAM block, which has been multipumped four times, each SRAM sub-block of size 4 × 1 emulating two TCAM bits. Thus, designing TCAM using multipumping-enabled multiported SRAM in Figure 2(c) and (d) achieved a higher SRAM memory efficiency (i.e. fewer SRAM bits are utilized per TCAM

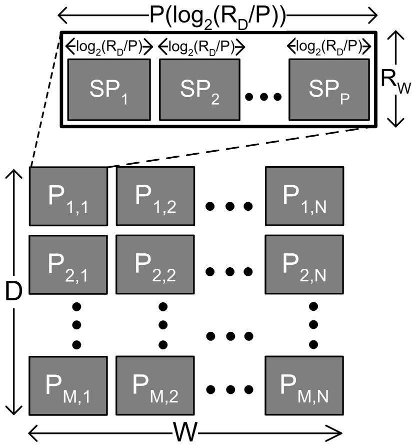

FIGURE 3. Proposed partitioning of the traditional TCAM table.

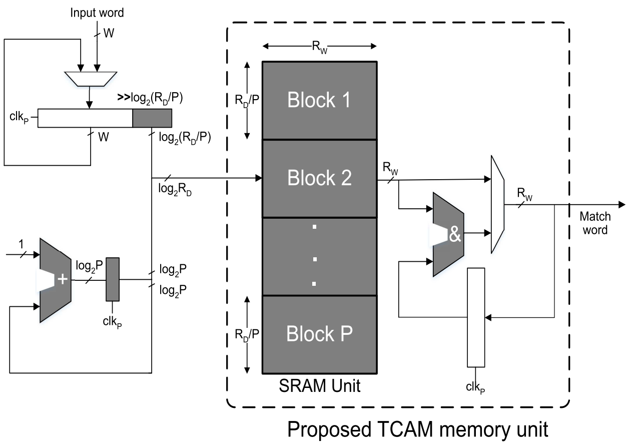

FIGURE 4. Basic architecture of the proposed TCAM memory.

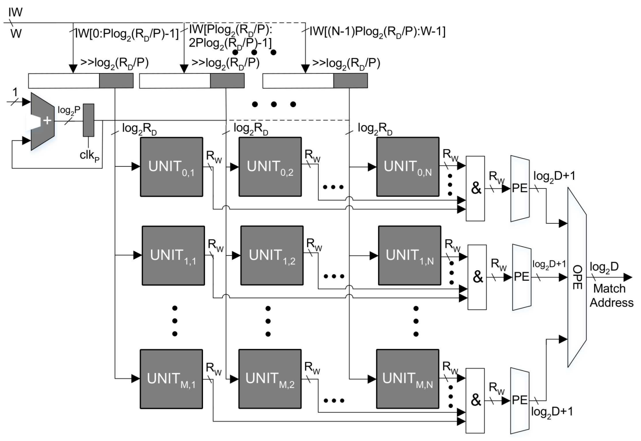

FIGURE 5. Organization of the proposed TCAM memory units for a large storage capacity: (*IW*: input word, *PE*: priority encoder, *OPE*: overall priority encoder).

bit) when compared with that of multipumping-less SRAM-based TCAM design in Figure 2(b). The TCAM bits storage capacity of the SRAM block increases with multipumping.

A multiported SRAM block of size $R_D \times R_W$ with a multipumping factor of P implements a traditional TCAM table of size $Plog_2(R_D/P) \times R_W$, each SRAM sub-block of size $(R_D/P) \times R_W$ emulating $log_2(R_D/P) \times R_W$ TCAM data, as shown in Figure 3 and 4. Our proposed design achieves increased TCAM bits storage capacity with an increase in multipumping factor P.

C. PROPOSED PARTITIONING OF TRADITIONAL TCAM TABLE

We partition the traditional TCAM table of size $D \times W$ into $M \times N$ partitions such that each partition consists of P parts of $log_2(R_D/P) \times R_W$ size as shown in Figure 3. Our proposed TCAM design uses its configured SRAM blocks of $R_D \times R_W$ size as multiported SRAM, constituting P sub-blocks of size $(R_D/P) \times R_W$ as shown in Figure 4.

Each sub-block of the SRAM stores $log_2(R_D/P) \times R_W$ size divisions of the traditional TCAM. Consequently the P sub-blocks of the multiported SRAM memory in our proposed design stores a traditional TCAM division of size $Plog_2(R_D/P) \times R_W$ as shown in the Figures 3 and 4. Similarly, the $M \times N$ TCAM divisions of size $Plog_2(R_D/P) \times R_W$ are mapped to the SRAM blocks of the $M \times N$ TCAM memory units in the proposed design, as shown in Figures 3 and 5.

D. BASIC ARCHITECTURE OF THE PROPOSED TCAM MEMORY

The basic architecture of our proposed TCAM memory design is shown in Figure 4. It is operated by two fully synchronized clocks, a system clock clk_S and internal clock clk_P, such that clk_P is P times faster than clk_S. An incoming TCAM word is registered in a W-bit shift register using the system clock clk_S. The log_2P-bit counter generates a sequence of log_2P-bit numbers in P internal clock cycles. It is initialized to zero upon reset and it rolls over after every P internal clock cycles. The log_2P-bits from the counter are concatenated with the $log_2(R_D/P)$ bits from the shift register to make the log_2R_D-bit address space of the SRAM. At the positive edge of the internal clock clk_P, the SRAM address is executed such that log_2P-bits from the counter constitute its most significant bits, and points to the start of the corresponding sub-block in SRAM and the lower $log_2(R_D/P)$ bits from the shift register selects an SRAM word in the sub-block.

The read SRAM words are *AND*-accumulated for each cycle in an R_W-bit register using clk_P. Similarly, the look-up is completed for a W-bit input word by reading and *AND*-accumulating SRAM words from each sub-block of the SRAM in P internal clock cycles or one system cycle. Consequently, the P *AND*-accumulated SRAM words are produced as match word using clk_S. The timing diagram in Figure 6 elaborates the search operation of the proposed TCAM memory architecture shown in Figure 4 with a multi-pumping factor of $P = 2$.

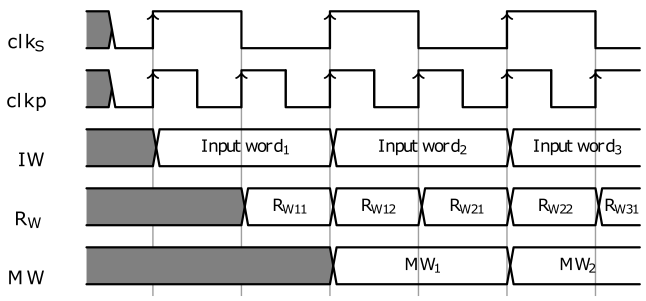

FIGURE 6. Timing diagram for the search operation in our proposed TCAM with a multipumping factor $P = 2$: (IW: input word, R_W: SRAM word read, MW: match word).

E. MODULAR ARCHITECTURE

TCAM design of large storage capacity is implemented as a cascade of $M \times N$ proposed design TCAM memory units as shown in Figure 5. An incoming W-bit TCAM word is divided into N sub-words of $Plog_2(R_D/P)$-bits with the bit ranges shown in Figure 5. The resultant sub-words are stored in N shift registers of size $Plog_2(R_D/P)$-bits on clk_S. The log_2R_D-bit indexes from the N shift registers are provided to the corresponding M TCAM memory units of the N columns of the proposed design in parallel using clk_P, as shown in Figure 5. All TCAM memory units of the design operate in parallel using clk_P. The R_W-bit match words from each row of the TCAM memory units are bit-wise *ANDed* on clk_S, and the results are provided to the associated priority encoder (PE) units. The log_2D-bit match address and the match information from each PE unit are provided to the overall priority encoder unit, which eventually forwards a match address based on the priority. The proposed TCAM design registers an input word and produces a match word as output on clk_S.

The update of a TCAM word is performed in each TCAM memory unit of the design in parallel. The worst-case update latency of the proposed design comprises R_D/P system cycles.

F. EFFECT OF MULTIPUMPING SRAM ON THE MEMORY USAGE AND THROUGHPUT

Multipumping results in a useful reduction in SRAM memory usage for the design of TCAM on FPGA. The configured SRAM memory blocks in our proposed design with the multipumping factor of P implements traditional TCAM divisions of size $Plog_2(R_D/P) \times R_W$ as shown in Figure 4. The TCAM bits storage capacity of SRAM blocks in the proposed design increases with an increase in P. The upper bound on the multipumping factor P is $R_D/2$, i.e. $R_D/2$ sub-blocks in the SRAM and each sub-block consists of two SRAM words.

Multipumping divides the achievable internal clock frequency of the design by the multipumping factor, to obtain the operating frequency of the overall system [13]–[16]. Although an increase in the multipumping factor P results in a higher memory efficiency for the design of TCAM, only the use of small multipumping factors is practical in order to avoid a significant drop in the operating frequency of the overall system. Overall multipumping factor P controls a tradeoff between the SRAM memory efficiency and speed of the proposed design.

TABLE 2. FPGA resource utilization of the proposed design.

Proposed design	TCAM size ($D \times W$)	Slice registers	LUTs	BRAMs (36 Kb)
CASE-I ($P = 4$)	512×28	536	968	8
CASE-II ($P = 2$)	512×32	1593	1515	16
CASE-III ($P = 4$)	1024×140	6287	7516	80

IV. IMPLEMENTATION SETUP AND RESULTS

To verify our proposed design we implemented it on a Xilinx Virtex-6 FPGA device (xc6vlx760). The proposed design was implemented using the Xilinx ISE 14.7 design tool, and verified through behavioral and post-route simulations using an ISim simulator.

We implemented our proposed design cases I and II on the Xilinx Virtex-6 FPGA device for 512×28 (14 Kb) and 512×32 (16 Kb) TCAM tables, with multipumping factors of $P = 4$ and $P = 2$, respectively. Our proposed design CASE-III implements a large TCAM table of size 1024×140 (140 Kb), with a multipumping factor of $P = 4$. We have selected small multipumping factors of $P = 4, 2$, and 4, in our proposed design cases I, II, and III, to avoid lower operating frequencies of the overall system.

Table 2 lists the FPGA resource utilization slice registers (SRs), look-up tables, and BRAMs for the implementation of our proposed design cases I, II, and III. The post place & route results show that the proposed design cases I, II, and III could achieve internal clock frequencies of 475 MHz, 475 MHz, and 349 MHz and multipumping factors of $P = 4$, 2, and 4, giving the system clock frequencies of 119 MHz, 237 MHz, and 87 MHz, respectively.

V. PERFORMANCE EVALUATION

The performance of our proposed design is evaluated based on its comparison with the existing SRAM-based TCAM solutions on FPGAs.

A. SRAM MEMORY UTILIZATION

SRAM-based TCAM solutions implement a traditional TCAM of depth D and width W by cascading SRAM blocks of size $R_D \times R_W$ on FPGAs. The minimum overall SRAM memory requirement of the existing SRAM-based TCAM solutions on FPGAs can be formulated as (1) shown below:

$$\sum_{M=1}^{\frac{D}{R_W}} \sum_{N=1}^{\frac{W}{log_2R_D}} (R_D \times R_W) = \left(\frac{D}{R_W}\right)\left(\frac{W}{log_2R_D}\right)(R_D \times R_W) = DW\left(\frac{R_D}{log_2R_D}\right) \quad (1)$$

The overall memory requirement of the proposed design for the implementation of a $D \times W$ size traditional TCAM using

TABLE 3. Performance per memory comparison of the proposed TCAM with previous approaches.

Architecture	FPGA	TCAM size ($D \times W$)	Speed (MHz)	BRAMs (36 Kb)	Memory usage (Kb)	Throughput (Gb/s)	Performance per memory ($(Gb/s \times TCAM\ Depth)/Kb$)
Locke- [22]	Virtex-6	512×36	166	64	2304	5.84	1.3
Jiang- [7]	Virtex-7	1024×150	97 (139)	272	9792	14.26	1.49
Qian- [17]	Virtex-6	504×180	133	140	5040	23.38	2.34
REST- [26]	Kintex-7	72×28	35 (50)	1	36	0.96	1.92
HP-TCAM- [10]	Virtex-6	512×36	118	56	2016	4.15	1.05
Z-TCAM- [11]	Virtex-6	512×36	159	40	1440	5.59	1.99
E-TCAM- [12]	Virtex-6	512×36	164	40	1440	5.77	2.05
UE-TCAM- [6]	Virtex-6	512×36	202	32	1152	7.1	3.16
Proposed CASE-I	Virtex-6	512×28	119	8	288	3.25	5.78
Proposed CASE-II	Virtex-6	512×32	237	16	576	7.41	6.59
Proposed CASE-III	Virtex-6	1024×140	87	80	2880	11.90	4.25

$R_D \times R_W$ size SRAM blocks is devised as (2) shown below:

$$\sum_{M=1}^{\frac{D}{R_W}} \sum_{N=1}^{\frac{W}{Plog_2(R_D/P)}} (R_D \times R_W) = \left(\frac{D}{R_W}\right)\left(\frac{W}{Plog_2(R_D/P)}\right)(R_D \times R_W) = DW\left(\frac{R_D}{Plog_2(R_D/P)}\right) \quad (2)$$

Equation (2) describes that the SRAM memory usage of our proposed design is $\frac{R_D}{Plog_2(R_D/P)}$ times that of the corresponding traditional TCAM table of size $D \times W$.

Our proposed design achieves a considerable reduction in the SRAM memory usage by a factor of $\frac{1}{P[1-log_2P/log_2R_D]}$, when compared with that of the existing approaches as described using (3) as follows:

$$\frac{DW\left(\frac{R_D}{Plog_2(R_D/P)}\right)}{DW\left(\frac{R_D}{log_2R_D}\right)} = \frac{log_2R_D}{Plog_2(R_D/P)} = \frac{log_2R_D}{P[log_2R_D - log_2P]} = \frac{1}{P[1 - log_2P/log_2R_D]} \quad (3)$$

The usage of BRAMs in our proposed design is compared with those of previous approaches in Column 5 of Table 3. Our proposed TCAM design CASE-I emulates a 14 Kb traditional TCAM, achieving a lower BRAMs utilization of 8 BRAMs compared with the usage of 56, 40, 40, 32, and 64 BRAMs for previous approaches in [10]–[12], [22], and [26], respectively for an 18 Kb traditional TCAM emulation. The proposed design CASE-III emulates a large TCAM of size 1024×140 using 80 BRAMs. It achieves a lower BRAMs utilization compared with the large TCAM implementations of size 1024×150 and 504×180 in the previous approaches [7] and [17], using 272 and 140 BRAMs, respectively.

B. THROUGHPUT

The operational speed of our proposed design is compared with those of previous approaches in column 4 of Table 3. Our proposed design cases I and II emulates traditional TCAM of size 14 Kb and 16 Kb achieving operating frequencies of 119 MHz and 237 MHz with multipumping factors of $P = 4$ and 2 respectively. The operating frequency of our proposed design CASE-II is higher than previous works [10]–[12], [22], and [26] for an 18 Kb traditional TCAM emulation.

Our proposed design methodology is more useful for the design of large storage capacity TCAMs. The TCAM memory units of our proposed design *AND*-accumulate SRAM words from the sub-blocks of the SRAM blocks in each system cycle, reducing the complexity of the *AND* operation units of the overall architecture, as shown in Figures 4 and 5. This further prevents the *AND* operation units from limiting the operating frequency of wide pattern TCAMs designs on FPGA. Our proposed design uses fewer BRAMs, thus alleviating the overall routing complexity of the design on FPGA. The divided *AND* operation complexity and reduced routing complexity makes our proposed design more practical for large storage capacity TCAMs.

The system frequency of our proposed design CASE-III emulating a large capacity TCAM of 140 Kb is 87 MHz, which is comparable with the maximum achievable frequency 97 MHz in previous work [7] implementing a large size TCAM of 150 Kb. While the SRAM memory usage of our proposed design CASE-III is 70% lower than that of [7].

Our proposed design provides increased design flexibility in terms of the speed vs memory usage tradeoff. The designer must consider the important design factors such as the required storage capacity, relative availability of BRAMs on the target FPGA, and required throughput for the selection of the multipumping factor in our proposed design.

C. PERFORMANCE PER MEMORY

Considering the time-space tradeoff, we used the performance evaluation metric *performance per memory* from [31], given by (4).

$$\frac{Throughput(Gb/s)}{Normalized\ Memory\ [Memory(Kb)/TCAM\ Depth]} \quad (4)$$

Table 3 compares the performance per memory of our design with previous FPGA-based TCAMs. The depth and pattern width of traditional TCAMs implemented in previous studies are listed in the third column. For a fair comparison, the speed results of the compared works with technology differences are normalized to 40 nm, using (5) from [32]. The speed results in parenthesis represent the original data reported in the respective papers.

$$T^* = T \times \left[\frac{40(nm)}{Technology(nm)}\right] \times \left[\frac{VDD}{1.0}\right] \quad (5)$$

where T represents the original delay time, and T^* denotes the normalized delay time for 40 nm CMOS technology with a supply voltage of 1.0 V. The proposed design cases I and II implemented 14 Kb and 16 Kb traditional TCAMs using 288 Kb and 576 Kb SRAM memory with operating frequencies of 119 MHz and 237 MHz, respectively. The proposed design cases I and II achieved a performance per memory of 5.78 $((Gb/s \times TCAMDepth)/Kb)$ and 6.58 $((Gb/s \times TCAMDepth)/Kb)$, respectively.

Table 3 shows that the performance per memory of the proposed design cases I and II are 1.83 times higher than that of UE-TCAM [6], which was the highest among the existing methods. Our proposed design CASE-III emulates a large TCAM of size 1024 × 140, achieving the performance per memory of 4.25 $((Gb/s \times TCAMDepth)/Kb)$, which is 2.85 times higher than for large TCAM of size 1024 × 150 in the existing study [7].

Our proposed design scales well in terms of the performance when evaluated for the design of a large storage capacity. Table 3 shows that the performance per memory of our proposed design CASE-III is slightly lower than the proposed design CASE-I (with the same multipumping factor of $P = 4$) while the implemented TCAM size of CASE-III is ten times greater than that of CASE-I.

VI. CONCLUSIONS AND FUTURE WORK

Re-configurable hardware FPGAs emulate TCAM functionality using SRAM memory. Existing SRAM-based solutions of TCAM on FPGAs achieve inefficient memory usage and offer lower operational frequencies. We have presented a memory-efficient design of TCAM, based on multipumping-enabled multiported SRAM, by operating the SRAM blocks in the design at a frequency that is multiple times higher than that of the overall system. This allows reading from its sub-blocks to take place within one system cycle. The FPGA implementation results show that the performance per memory of our proposed design is up to 2.85 times higher than for existing SRAM-based TCAM solutions on FPGA.

Our proposed solution is general, and can be applied to many applications. Our future work will include the application of the proposed design to various applications.

REFERENCES

[1] B. Agrawal and T. Sherwood, "Ternary CAM power and delay model: Extensions and uses," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 16, no. 5, pp. 554–564, May 2008.

[2] M. Imani, A. Rahimi, and T. S. Rosing, "Resistive configurable associative memory for approximate computing," in *Proc. IEEE Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2016, pp. 1327–1332.

[3] N. Mohan, W. Fung, D. Wright, and M. Sachdev, "Design techniques and test methodology for low-power TCAMs," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 6, pp. 573–586, Jun. 2006.

[4] L.-Y. Huang *et al.*, "ReRAM-based 4T2R nonvolatile TCAM with 7x NVM-stress reduction, and 4x improvement in speed-wordlength-capacity for normally-off instant-on filter-based search engines used in big-data processing," in *Symp. VLSI Circuits Dig. Tech. Papers*, Jun. 2014, pp. 1–2.

[5] M.-F. Chang *et al.*, "A 3T1R nonvolatile TCAM using MLC ReRAM for frequent-off instant-on filters in IoT and big-data processing," *IEEE J. Solid-State Circuits*, vol. 52, no. 6, pp. 1664–1679, Jun. 2017.

[6] Z. Ullah, M. K. Jaiswal, R. C. C. Cheung, and H. K. H. So, "UE-TCAM: An ultra efficient SRAM-based TCAM," in *Proc. IEEE Region 10 Conf. (TENCON)*, Nov. 2015, pp. 1–6.

[7] W. Jiang, "Scalable ternary content addressable memory implementation using FPGAs," in *Proc. 9th ACM/IEEE Symp. Archit. Netw. Commun. Syst.*, 2013, pp. 71–82.

[8] *Virtex-6 FPGA Memory Resources User Guide*, Xilinx, San Jose, CA, USA, 2014. [Online]. Available: http://www.xilinx.com

[9] P. Alfke, "Creative uses of block RAM," Xilinx, San Jose, CA, USA, White Paper WP335, 2008.

[10] Z. Ullah, K. Ilgon, and S. Baeg, "Hybrid partitioned SRAM-based ternary content addressable memory," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 59, no. 12, pp. 2969–2979, Dec. 2012.

[11] Z. Ullah, M. K. Jaiswal, and R. C. C. Cheung, "Z-TCAM: An SRAM-based architecture for TCAM," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 23, no. 2, pp. 402–406, Feb. 2015.

[12] Z. Ullah, M. K. Jaiswal, and R. C. C. Cheung, "E-TCAM: An efficient SRAM-based architecture for TCAM," *Circuits, Syst. Signal Process.*, vol. 33, no. 10, pp. 3123–3144, Oct. 2014.

[13] C. E. LaForest and J. G. Steffan, "Efficient multi-ported memories for FPGAs," in *Proc. 18th Annu. ACM/SIGDA Int. Symp. Field Program. Gate Arrays*, 2010, pp. 41–50.

[14] A. Abdelhadi and G. G. F. Lemieux, "Modular switched multiported SRAM-based memories," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 9, no. 3, p. 22, 2016.

[15] H. E. Yantir, S. Bayar, and A. Yurdakul, "Efficient implementations of multi-pumped multi-port register files in FPGAs," in *Proc. Euromicro Conf. Digit. Syst. Design (DSD)*, Sep. 2013, pp. 185–192.

[16] C. E. LaForest. *Multi-Ported Memories for FPGAs*. Accessed: Nov. 10, 2017. [Online]. Available: http://fpgacpu.ca/multiport/index.html

[17] Z. Qian and M. Margala, "Low power RAM-based hierarchical CAM on FPGA," in *Proc. Int. Conf. ReConFigurable Comput. FPGAs (ReConFig)*, Dec. 2014, pp. 1–4.

[18] P. Mahoney, Y. Savaria, G. Bois, and P. Plante, "Parallel hashing memories: An alternative to content addressable memories," in *Proc. 3rd Int. IEEE-NEWCAS Conf.*, Jun. 2005, pp. 223–226.

[19] S. Cho, J. R. Martin, R. Xu, M. H. Hammoud, and R. Melhem, "CA-RAM: A high-performance memory substrate for search-intensive applications," in *Proc. IEEE Int. Symp. Perform. Anal. Syst. Softw. (ISPASS)*, Apr. 2007, pp. 230–241.

[20] S. V. Kartalopoulos, "RAM-based associative content-addressable memory device, method of operation thereof and ATM communication switching system employing the same," U.S. Patent 6 097 724, Aug. 1, 2000.

[21] M. Somasundaram, "Circuits to generate a sequential index for an input number in a pre-defined list of numbers," U.S. Patent 7 155 563, Dec. 26, 2006.

[22] K. Locke. (2011). *Xilinx Application Note: XAPP1151—Parameterizable Content-Addressable Memory*. [Online]. Available: http://www.xilinx.com

[23] Z. Ullah, "LH-CAM: Logic-based higher performance binary CAM architecture on FPGA," *IEEE Embedded Syst. Lett.*, vol. 9, no. 2, pp. 29–32, Jun. 2017.

[24] 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.

[25] A. Kulkarni and D. Stroobandt, "MiCAP-Pro: A high speed custom reconfiguration controller for dynamic circuit specialization," *Des. Autom. Embedded Syst.*, vol. 20, no. 4, pp. 341–359, 2016.

[26] A. Ahmed, K. Park, and S. Baeg, "Resource-efficient SRAM-based ternary content addressable memory," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 4, pp. 1583–1587, Apr. 2017.

[27] N. Manjikian, "Design issues for prototype implementation of a pipelined superscalar processor in programmable logic," in *Proc. IEEE Pacific Rim Conf. Commun., Comput. Signal Process. (PACRIM)*, vol. 1. Aug. 2003, pp. 155–158.

[28] H. Yokota, "Multiport memory system," U.S. Patent 4 930 066, May 29, 1990.

[29] B. A. Chappell, T. I. Chappell, M. K. Ebcioglu, and S. E. Schuster, "Virtual multi-port RAM employing multiple accesses during single machine cycle," U.S. Patent 5 542 067, Jul. 30, 1996.

[30] G. S. Ditlow *et al.*, "A 4R2W register file for a 2.3 GHz wire-speed POWER processor with double-pumped write operation," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2011, pp. 256–258.

[31] H. Nakahara, T. Sasao, H. Iwamoto, and M. Matsuura, "LUT cascades based on edge-valued multi-valued decision diagrams: Application to packet classification," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 6, no. 1, pp. 73–86, Mar. 2016.

[32] P.-T. Huang and W. Hwang, "A 65 nm 0.165 fJ/Bit/search 256×144 TCAM macro design for IPv6 lookup tables," *IEEE J. Solid-State Circuits*, vol. 46, no. 2, pp. 507–519, Feb. 2011.

INAYAT ULLAH received the bachelor's degree in computer system engineering from the University of Engineering and Technology, Peshawar, Pakistan, in 2007. He is currently pursuing the Ph.D. degree with the College of Electronics and Information Engineering, Chosun University, South Korea. Since 2008 he has been a Faculty Member with the Department of Electrical Engineering, Federal Urdu University of Arts, Science & Technology, Pakistan. His areas of interest are digital design, computer architecture, parallel processing, memory architecture, and re-configurable 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 serving as an Associate Professor with the Department of Electrical Engineering, CECOS University of IT & Emerging Sciences, Peshawar, Pakistan. He has authored prestigious journal and conference papers and holds patents in his name in the field of FPGA-based TCAM. His research interests include low power/high speed CAM design on FPGA, low power/high speed VLSI design, and embedded systems.

JEONG-A LEE (M'84–SM'01) received the B.S. degree (Hons.) in computer engineering from Seoul National University in 1982, the M.S. degree in computer science from Indiana University Bloomington, in 1985, and the Ph.D. degree in computer science from the University of California, Los Angeles in 1990. From 1990 to 1995, she was an Assistant Professor with the Department of Electrical and Computer Engineering, University of Houston. Since 1995 she has been affiliated with Chosun University, South Korea. From 2008 to 2009, she served as a Program Director of ECE division, National Research Foundation of Korea. She has authored and co-authored over 100 reviewed journal and conference papers. Her research activities cover high performance computer architectures, memory architecture, approximate computing, self-aware computing, and reliable computing. She is a member of the National Academy of Engineering in South Korea.