Low-power Content Addressable Memory (CAM) Array for Mobile Devices

Khader Mohammad *, Aziz Qaroush *, Mahdi Washha *, Baker Mohammad†
* Department of Electrical and Computer Engineering, Birzeit University, Ramallah, Palestine
†Khalifa University, Abu Dhabi, United Arab Emirates

***Abstract*—Large-capacity content-addressable memory (CAM) is beneficial in a variety of applications that require high-speed lookup table. It is used extensively in low power CPU design, network routers, and cache controllers. Content addressable memory system includes CAM cells that contains a compare-circuit and a memory bit-cell that stores complementary bits. The compare circuit consists of complementary inputs to receive the complementary stored bits, and an input node to receive a single-ended reference bit. The main CAM design challenge is to reduce the power consumption associated with large amount of parallel switching circuitry, without sacrificing speed or density.**

This paper presents a novel CAM circuit level implementation aiming at reducing the comparator power and the crowbar current. Consequently, the average current consumption during CAM operation is reduced. In addition, the proposed circuit topology eliminates the need to route the complementary data which saves routing resources. Simulation results using 22nm process technology shows that the elimination of the crowbar current during writing operation saves 40% of power for single bit-cell CAM, while sharing the compare circuitry among 8 bit-cells CAM saves 14% of the power without any performance impact on area and delay.

***Keywords*—Low Power, CAM memory, caches, content addressable memory, memory architecture, SOC design, processors design, CAM cell, chip, design, XOR CAM.**

I. Introduction

In Random Access Memory (RAM) system, a memory address and a read control signal are send to the RAM module to retrieve or read the contents stored at the target memory address. Content Addressable Memories (CAMs) is defined from the functionality point of view as a memory with large amount of stored data that performs comparison between the input searched data and the stored data. CAM cell can perform all functions of the SRAM cell, including read and write operations given address and data. In addition, It is capable of performing the matching/comparing operation. As a kind of comparison between CAM and RAM, one can say that the CAM is an inverse of RAM in terms of functionality. At the reading operation, RAM retrieves the stored data for a given address while CAM returns the address corresponding to a given data word as shown in Fig. 1 [1]. In addition, the search is carried out serially when searching for a data in a RAM block. Consequently, many cycles are needed to find a particular data word. But in CAM-based memory, all addresses are searched in parallel fashion to retrieve the corresponding address that stores the desired word.

CAMs are gaining increased importance due to their parallel pattern matching property. This property makes them useful in variety of applications such as cache controllers where a quick search is needed. CAMs can be defined as a hardware search-based engines that are much faster than algorithmic approaches for search-intensive applications [2] like find matching contents in a database or table. For example, Fig. 2 [3] shows that the CAM system has been employed as a routing table where the match address output from the CAM is in fact a pointer used to retrieve associated data from the RAM. In this case the associated data is the router output port. Also, the CAM system may be viewed as a hardware embodiment of a software-based associative array. So, based on these domain of applications, the key/basic idea of the CAM is to compare the CAM data lines of the cell with contents of a certain bits, and then if the data is matched, the matched lines of the stored bits are raised and then the addresses at matched target data are returned. In addition to that, the CAM system may return the data word or other associated information [2].

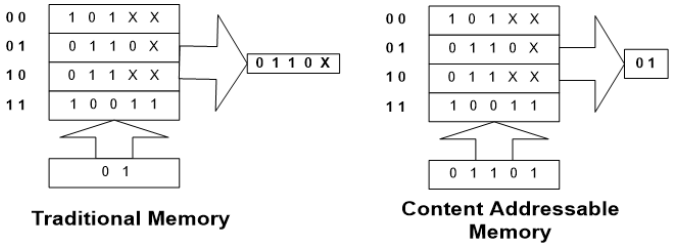


Fig. 1: RAM versus CAM in read operation [1]

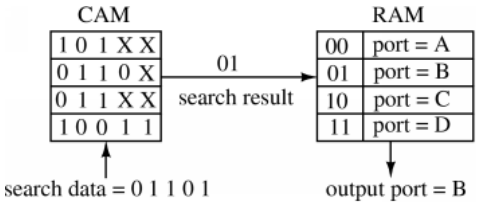


Fig. 2: Address lookup engine with CAM system [3]

The major use of CAM memory falls in the applications that require high search or lookup speed. However, the CAM search speediness comes at the cost of higher power consumption with large amount of parallel switching circuitry and larger silicon

area [4], which are the two main design parameters especially for Mobile devices [5]–[7]. Thus, the main goal of the recent dedicated research in CAM Memory area concentrates on reducing power consumption without slowing down the CAMs search speed. The accomplished work has been done at two levels: circuit level and architecture level. In circuit level, the CAM structure is implemented as NAND-gate or NOR-gate types while the architectural level uses simple SRAM cell as a bit storage and the comparison function is equivalent to XOR i.e. XNOR logic operation [4]. Indeed, using XOR as pass gate is widely used in CAM-based memory arrays which are susceptible to currents contention due to the variation on arrival time and delays in the input signals. The design of the elementary CAM chip cell can be abstracted as a cross product of SRAM and XNOR circuits as presented in [8].Nonetheless, there have been many attempts to reduce the transistor count and resulting area for the CAM XOR block in order to decrease the power consumption. A comprehensive review of different varieties of CAM cells, were presented in [9]. Other design approaches were presented in [4], [9]–[15].

Bit cells of a CAM system include compare-circuits to compare contents of the bit cells with reference bit value provided to the compare circuits. Conventional CAM compare-circuits are implemented with complementary or differential reference bit lines, which disadvantageously increase routing complexity and space requirements. Typically, the compare circuits include separate pass circuits. and thus the switching delays in the CAM cell can cause unwanted current contention between the separate pass circuits, which manifests itself as a crowbar current that wastes power and slows down CAM speed. The goal of this paper is to reduce both he active and the contention power in addition to reducing the required metal tracks to route signals. The proposed design is based on real product using CMOS technology. It reduces write energy compared to traditional real approaches. In addition, the proposed design saves on routing resources and interconnect power without any performance impact on chip area and delay.

The rest of the paper is organized as follows: Section II describes the functionality and the implementation of the CAM system. Section III illustrates the available techniques in reducing the power consumption for the CAM system. Section IV introduces our approach (for typical and custom CAM cell). In section V, the simulation results are studied and analyzed. Section VI concludes our proposed work.

II. CAM Functionality and Implementation

The overall CAM functionality is to take a search word and return the matching memory location, but one can think in different perspective of this operation as a fully programmable arbitrary mapping of a large space of the input search word to a smaller space of the output match location. As another perspective, the operation of a CAM can be seen as tag portion of a fully associative cache where the input of tag portion of a cache is compared to all addresses stored in the tag memory. Then, when matching is occurred, a single match line goes high, indicating the location of a match. Although many circuits are common to both CAMs and caches, in our work we concentrate on large capacity CAMs rather than on fully associative caches since they target smaller capacity and higher speed CAMs.

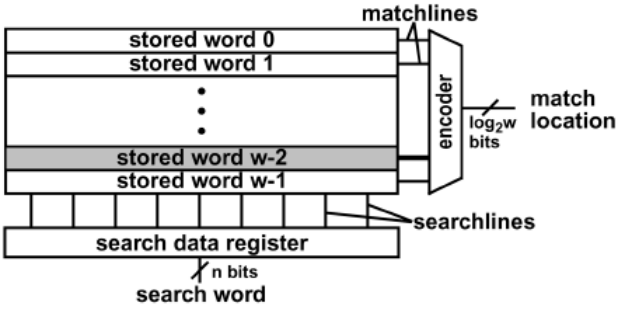


Fig. 3: Content-addressable memory (CAM) conceptual view searches over W words [4].

Content-addressable memory consists of W words as shown in Fig 3 [4]. The system input is a search word that will be broadcasted onto the search lines to be compared with a table of stored data. The CAM word size, in terms of bits, is often large, with an available and used implementations it ranging from 36 to 144 bits while the table size of the typical CAM is ranging between a few hundred to 32K entries, which correspond to an address space ranging from 7 bits to 15 bits. The match lines indicate whether a matching occurred between a stored word and the given search word. According to the match lines value (0 or 1), the location or address of word is determined by feeding the match lines to an encoder, assuming that the encoder is used in the systems where only a single match is expected. In addition to that there is often a hit signal, not shown in the Fig. 3, that is responsible to indicate the case when no matching location in the CAM is found.

The largest commercial implementations of single-chip CAMs are 18 Mbits size [16]; however, the largest CAMs stated in the literature are 9 Mbits size. Usually, as a rule of thumb, about half of the largest available SRAM chip size is the largest available CAM chip size where such rule of thumb refers to the fact states that the typical CAM cell contains two SRAM cells [17]. The capacity of CAM chips that have been published from 1985 to 2004 revealed an exponential growth rate of semiconductor memory circuits as reported in Fig. 4 [4].

From the CAM architecture side, an example of a small model is shown in Fig. 5 [4]. The CAM model in this example consists of 4 words where each word contains 3 bits arranged horizontally, which corresponding to 3 CAM cells. Also, for each word there is a match-line which is fed to match-line sense amplifiers (MLSAs), in addition to a differential search line-pair which corresponding to the search word bits. The CAM search process operates by loading first the search-data word into the search-data registers. Then, all match-lines are pre-charged, which put all of them in the match-state. Afterwards, the search-word is broadcasted onto the differential search lines using the search-line drivers, and then a

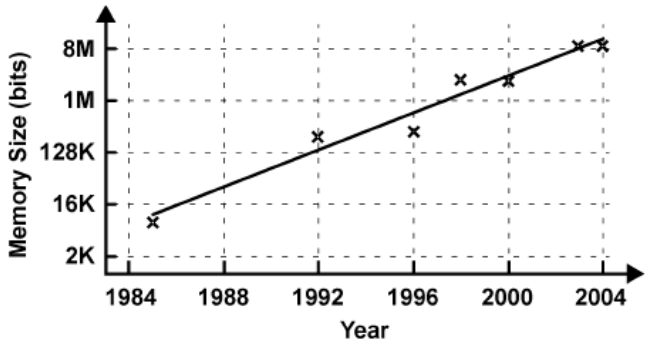


Fig. 4: CAM capacity (log scale) over year of publication [4].

comparison is performed between each stored bit of the CAM core cell and the corresponding search-lines bit. In the case of all bits are matched, the match-lines remain in the pre-charged high state and the match-lines that have miss bits are to ground. According to the match-lines state, the MLSA detects if the matching condition satisfied or not. At last, the matched location address is extracted by using the encoder.

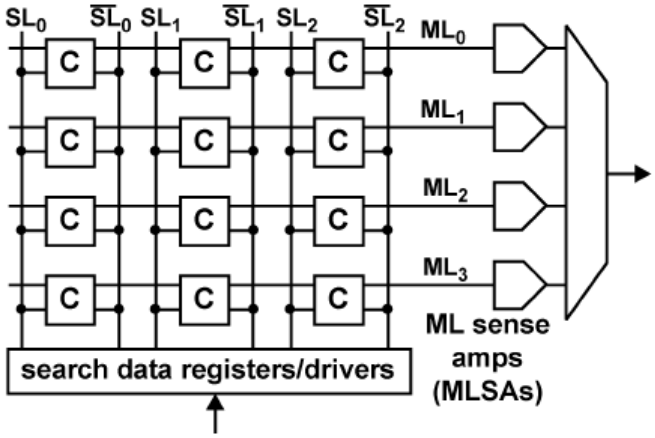


Fig. 5: Simple schematic CAM model for 4 words with 3 bits each, including individual core cells, differential search lines, and match line sense amplifiers (MLSAs) [4].

The bit comparison operation is logically equivalent to applying an XOR gate between the search bit and the stored bit; however, from the implementation-wise, the NOR and NAND cells are often used. When using the NOR cells, the comparison is done between the complement of the stored-bit D, and the complement of the search-data of the search-line SL, by using four transistors with minimum size as possible to keep high cell density. The four transistors aim at implementing a pulldown of XNOR gate with SL and D inputs where each pair of them creates a pulldown path from the match-line ML by connecting ML to ground. In the case of matching between SL and D, both pulldown paths are disabled, resulted in disconnecting ML from ground. For NAND cell implementation, it is different from the NOR implementation through using three comparison transistors rather than four when comparing between the stored-bit D, and corresponding search-data on the corresponding search-lines SL. This cell implementation becomes clear in particular when connecting multiple NAND cells in serial fashion. In such a case, joining the and nodes will form a word, taking into account that the pulldown path of a CMOS NAND gate is created by a serial nMOS chain of all transistors. However, satisfying the match condition for the entire word is dependent on being every cell in a word in a match state condition.

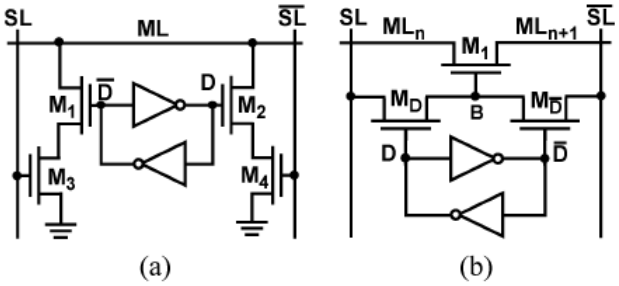


Fig. 6: CAM core cells for (a) 10-T NOR-type CAM and (b) 9-T NAND-type CAM [4].

The CAM core cells of the 10-T NOR-type and 9-T NAND type are shown in Fig. 6 [4] where SRAM is used as a data storage cell. For simplicity sake,the figure didn't show the SRAM access transistors and the associated bit lines, but it is clear that six of cell-transistors are accounted for both the SRAM storage and access transistors. The NOR cell has an important property concerning on providing a full rail voltage at all comparison transistors gates. On contrary, the NAND cell provides a reduced logic 1 voltage at node B.

Conventional CAM bit-cell CMOS implementation which is currently used by Intel is shown in Fig.7. The bit-cell (BIT) is written into by enabling the write word-line (WRWL) and driving the desired value through write bit-lines (WRBL/WRBLY). Then, the BIT value is compared against CAM-DATA and the MATCH is asserted when BIT and CAM-DATA values are same. However, this particular implementation has a crowbar current issue since there is one gate delay between BIT and BITX which opens one pass gate before closing the second as shown in Fig.7. Therefore, the new propose circuit will reduce this crowbar current. For more information about the Intel CAM design you can see the patents in [18], [19].

Fig.8 shows an example of CAM system in an exploded view using 8-connected conventional Intel CAM cells. Each CAM cell includes a memory/storage bit-cell (SRAM) and a compare circuit (CMPR). Bit-cell stores a bit (BIT) and a complementary bit (BITX) at corresponding nodes of the bit-cell. Bit cell may be static RAM (SRAM)or dynamic RAM (DRAM) [19], but the SRAM has been the preferred choice so far to build CAM-cells because of its robustness. Compare circuit CMPR includes complementary inputs (also referred to as differential inputs) to receive complementary stored bits

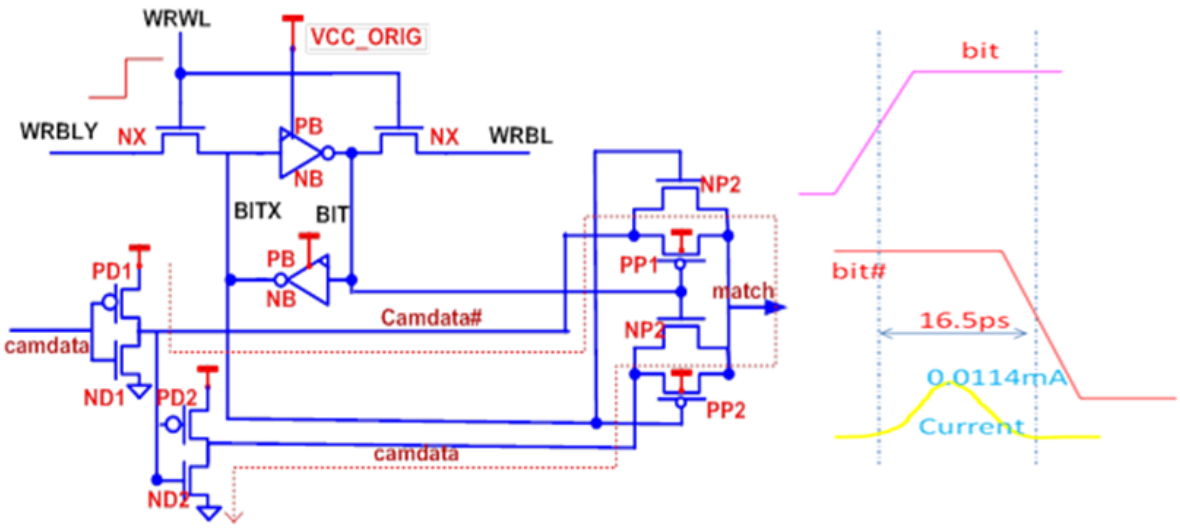


Fig. 7: Abstract view of the conventional Intel CAM bit-cell and the crowbar current during BIT write operation

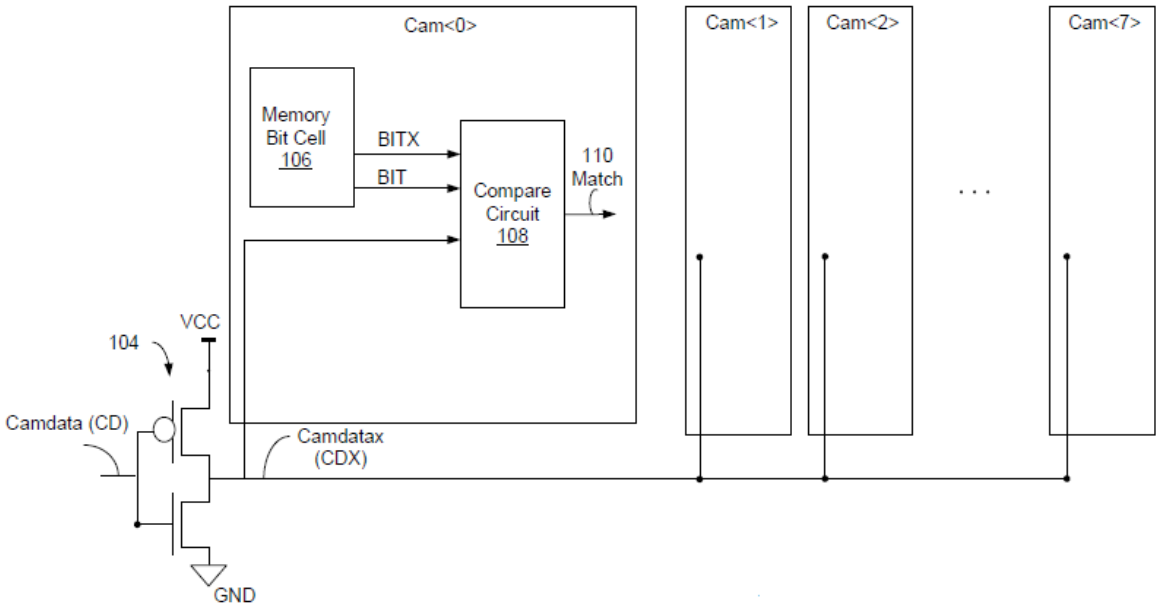


Fig. 8: Abstract view Intel CAM system using 8 connected cells

BIT and BITX, and a single-ended input to receive single ended inverted reference bit CDX. In the example of Fig.8, the reference bit inverter (INVX) receives a single-ended reference bit CAM Data (CD) and outputs a single-ended inverted reference bit CAM DataX (CDX), that will be provided to each CAM cells with talking into consideration that all of the circuits described here are connected to same supplies VCC (High) and VSS (Low) specifically the inverter INVX. As will be described later in detail, during the CAM search operation, the compare circuit in each CAM cell determines whether the single-ended inverted reference bit CDX matches data-bit stored locally in the CAM cells, and then asserts an output indicating the result of matching, i.e., whether the CAM cells stored data-bit matches the inverted reference-bit CDX. The compare circuit applies an XOR operation between the stored data-bit and the inverted reference-bit, which is equivalent to applies XNOR operation between the stored data-bit and the reference-bit.

Current research on CAM system proposed Memristor technology [20], [21] to be used for CAM-based memory. The main advantages of this technology are: (1) it is non-volatility (2) and can be scale better than CMOS technology. However, the Memristor technology is still not well developed and add a cost to CMOS since a new material (metal oxide) is needed. In addition, there are many challenges with memristor technology such as big variation, state drift, read disturb of the state during match.

III. CAM Power Reduction Techniques

Reducing the power dissipation by the CAM cells has been accomplished at two different levels where the first level is at the circuit design of the CAM cell, and the second level is at the architectural method used in connecting the CAM cells together. Most of the realized works have been focusing at the architecture level due to the difficulty of improving the design of the CAM cell at the circuit or transistor level. Although of that difficulty, the author of [8] describes a technique aims at reducing the power consumption and the area at the circuit level of the CAM. In addition , paper [22] discussed how to minimize power using serialization-widening with frequent value encoding for on chip signals like CAM signals. At the architecture level, most of the works have concentrated on pipelining (selective pre-charge) and pre-computation search approaches [4].

Pipelining or selective pre-charge, basically divide match-line into two segments and when performing a search on a data-bits, if the results of matching applied on the first segment fails there is no need to check the remaining bits. This will significantly reduce the power consumption. For example, when using uniform random data it only have to search (1/ 2) n of the rows. For n=3 this will save about 88 % of the match line power [23]. The approach can be generalized by dividing the match-line into many number of segments or stages and thus forming a pipeline [23]–[26]. If match in any stage is failed the next stages are shut-off resulting in power saving. Fig. 9 shows an example of pipelined match-line with four segments or stages compared to Non-pipelined match line [4]. In [24], the authors use a pipelined scheme by breaking the matching lines into several segments where the first segment consists of 8 bits and the rest segments of 34 bits each with 144 bits as a total word size. The work lies in evaluating the segments in the sequential fashion where the matching process in a segment begins based on the value of the matching line bit of the previous segment.

Another works [23], [27], [28] have exploited the pipeline scheme called pre-computation by storing additional information about the stored words likes storing the number of ones in the data word as shown in Fig. 10 [4] . Another work stores the parity bit as an additional information for each stored words by adding additional CAM cell to the original word CAMs cells [8]. For an input search data word, the computed parity bit is compared with the stored parity bit of each stored words. When a parity bit of the input word matches a parity bit of a stored word, the output match-line of the CAM cell is used as an enable bit to start the comparing process between the input word and the stored words. Also, in addition to parity bit pipelining, the matching operation between input word and stored words has been pipelined, but with segments of one bit only. Besides, In [23] a combination of the two power reduction techniques is used.

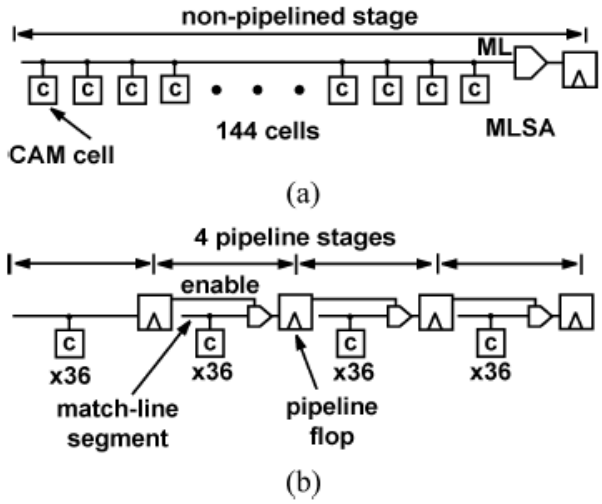


Fig. 9: Match-line schemes (a) Non-pipelined match-line (b) 4-Stage piplined match-line [4].

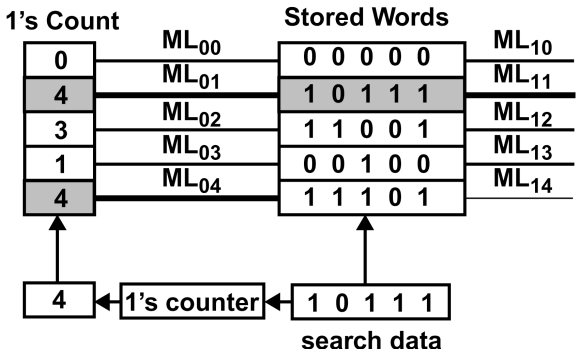


Fig. 10: Conceptual view of pre-computation-based CAM system [4].

It is evident that the proposed solutions at the architecture level improves the power consumption; however, they have several drawbacks which are summarized as following:

1) The latency is increased due to the subsequent matching and also in computing additional information as the case of parity bit.
2) Parity extractor circuit and its CAM cell may consume significant area.

With these drawbacks of the accomplished improvement at the architecture level, the circuit or transistor design still plays an important role since any improvement at this level directly will affect the architecture level. Thus, our work focuses on improving the power consumption of the CAM cell with minimum impact on latency and area.

IV. PROPOSED CAM-CELL CIRCUIT DESIGN

Conventional CAM bit-cell implementation shown in Fig.7 produces crowbar current during bit-cell write operation since there is one gate delay between BIT and BITX which opens one pass gate before closing the second. In this section, a novel CAM-cell implementation is presented to handle this problem and also to eliminates the need for routing CD and CDX to every CAM cell. Such elimination saves precious routing track and generates power saving in smaller node technology where wire cap is dominant.

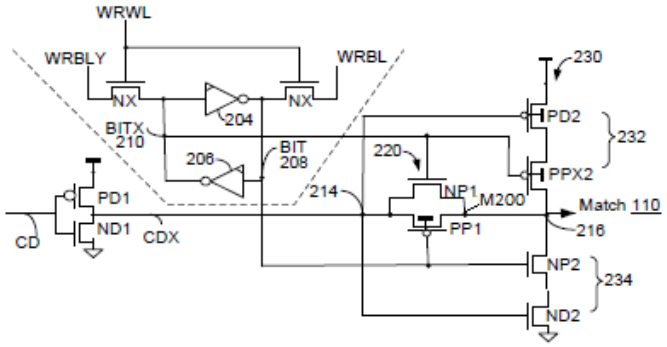


Fig. 11: Circuit diagram of proposed bit cell and compare circuit.

Fig.11 shows the circuit diagram of the proposed implementation of the CAM bit-cell along with its compare-circuit. The proposed compare-circuit includes circuitry which is controlled by the logic values of single-ended inverted reference bit CDX and the complementary stored bits BIT, BITX, which forms the inputs of the desired compare-circuit. The output of the compare-circuit is a match-bit which indicates whether the inverted reference bit CDX matches the stored bit BIT. The compare-circuit output a match-bit with logic value 1 when the inverted reference bit CDX differs from the stored bit BIT (matches the stored complementary bit BITX) and logic value 0 when the inverted reference bit CDX matches the stored bit BIT (differs from the stored complementary bit BITX). In CAM system, the use of single-ended reference data (i.e., the inverted reference bit CDX) instead of the complementary reference data has the following advantages: (1) reduces routing complexity and space, (2) significant reduction in the line driver power requirements, and (3) reduces complexity of the compare-circuit. In addition, the proposed bit-cell structure includes a cross-coupled inverters INVX1, INVX2 which are configured to store complementary bits BIT, BITX at the corresponding nodes BIT, BITX. In order to write the bits BIT, BITX to bit-cell, an external write circuits, not shown in Fig.11 , assert write bit-lines WRBL, WRBLX and write word-line WRWL in an appropriate way to energize write switches NX, which is commonly known method in the relevant state of arts.

The compare-circuit in the proposed implementation includes an input node CDX to receive single-ended inverted reference bit CDX, and an output node M200 to output match-bit. The compare-circuit also includes the following: a pass circuit N220 connected between input node N214 and output node M200, a pull-up circuit PUT connected between output node M200 and VCC, and a pull-down circuit PDT connected between the output node and VSS/GND. At any given time, only one of the pass circuit, pull-up circuit, and pull-down circuit drives output node M200 which is responsive to logic values of bits BIT, BITX, and CDX, i.e., the pass, pull-up, and pull-down circuits drive the output node in a mutually exclusive fashion.

The pass-circuit in the proposed implementation contains two complementary type transistors NP1, PP1 having their source-drain (S-D) current paths connected in parallel between input node N214 and output node M200, and their gates are driven or controlled by bits BITX, BIT, respectively. Pass circuit passes inverted reference bit CDX to the output node M200 only when BIT is logic 0 and BITX is logic 1, which turns ON (i.e., closes) the pass circuit.

Compare-circuit also includes an output switch stack 230, which including pull-up circuit 232 and pull-down circuit 234, coupled to output node 216. Output switch stack has a switch devices connected in series, that can be constructed as transistors PD2, PPX2, NP2, and ND2 having their S-D paths connected in series with each other to implement the switch stack. In addition, stack devices includes pull-up circuit PUT including ptype switch transistors PD2, PPX2 connected in series between output node M200 and VCC and having their gates driven by inverted reference bit CDX, and BITX, respectively. The pull-up circuit, pulls up output node M200 (and thus match bit) only when both CDX and BITX are 0. Stack PDT also includes pull-down circuit including ntype switch transistors ND2, NP2 connected in series between output node M200 and VSS/GND and having their gates driven by inverted reference bit CDX, and BIT, respectively. Pulldown circuit pulls down output 110 only when both CDX and BIT are 1. Compare-circuit produces match output 110 logic levels in response to the logic levels of bits BIT, BITX, and CDX in a manner summarized in Table I.

Based on Table I, it is evident that the compare-circuit functionally performs an XOR operation between bits CDX and BIT, which is, effectively equivalent to an XNOR operation between bits CD and BIT, under the control of differential bits BIT, BITX, and single-ended inverted reference bit CDX.

TABLE I: Match output that produced by compare-circuit.

BIT	BITX	CD	CDX	Output/Match	Pass Circuit	PD2	PPX2	ND2	NP2
1	0	1	0	1(=PUP)	OFF	ON	ON	OFF	ON
1	0	0	1	0(=PDN)	OFF	OFF	ON	ON	ON
0	1	1	0	0(=CDX)	ON	ON	OFF	OFF	OFF
0	1	0	1	1(=CDX)	ON	OFF	OFF	ON	OFF

During write operation to CAM cell, bits BIT, and BITX are written into the bit-cell. To perform a CAM compare operation, i.e., to determine whether reference bit CD matches the stored bit BIT and therefor assert output as a result of that compare, first the reference bit CD is asserted, and then the compare-circuit determines whether the stored bit BIT matches reference bit CD based on inverted reference bit CDX, stored BIT, and stored complementary bit BITX, which control pass circuit, pull-up circuit and pull-down circuit. According to

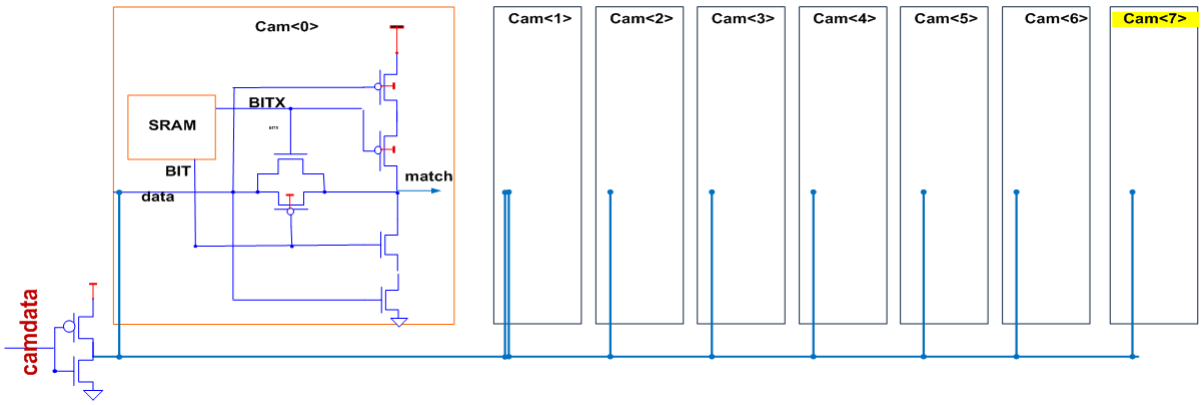
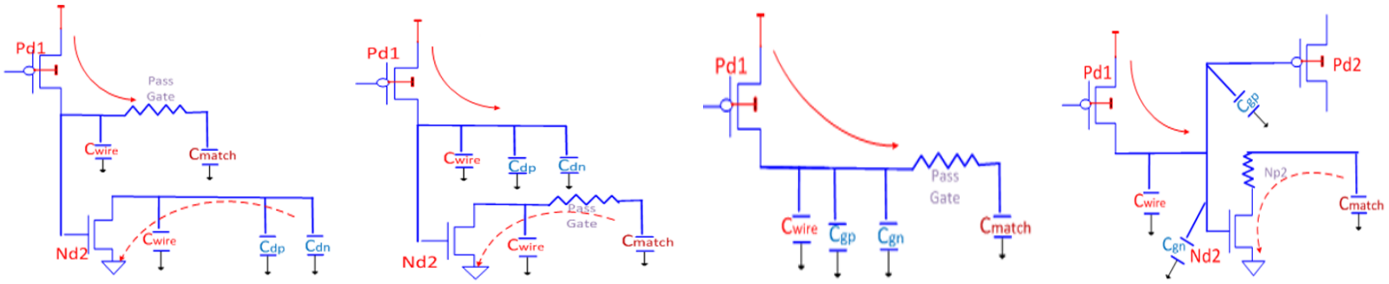


Fig. 12: Typical implementation of 8 bit-cell CAM system



(a) Original CAMdata=0→1, bit=0.

(b) Original CAMdata=0→1, bit=1.

(c) Proposed CAMdata=0→1, bit=0.

(d) Proposed CAMdata=0→1, bit=1.

Fig. 13: Charge and discharge path for original and proposed when CD=0 and BIT is 0 one time and 1 another time. Cgn/Cgp is PFET and NFET Gate device cap. Cdp/Cdn is PFET and NFET Diffusion cap. Cwire is Wire Cap. Cmatch is Cap on the match node.

table I, if the logic level of CD matches the logic level of BIT, then the compare-circuit produces match-bit at a first logic level (= 1). In contrary, if CD does not match BIT, then compare-circuit produces match bit at a second logic level (= 0). Fig. 13 shows the Charge and discharge path for the original and the proposed implementation when CD=0 and BIT is 0 one time and 1 another time.

Fig.12 shows an example of a block/circuit diagram regarding the proposed implementation of CAM system including multiple CAM cells that share compare circuitry. Each of the CAM cells in the diagram includes a portion of compare-circuit locally to one cell and repeated for each of the CAM cells. Also, it includes other portions that are connected to, and shared among, all other CAM cells , which thereby reducing power and area. Specifically, each compare cell includes its own local (or respective) pull-up transistor PPX2, and a second pull-up transistor PD2 connected in series with PPX2, and controlled by CDX, which in contrary to PPX2 connected to and shared among all of the other CAM cells compare-circuits. The source of one shared transistor PD2 is connected to VCC and its drain is connected to the sources of the local transistors PPX2 present in each of compare-circuits . In a similar manner, one pull-down transistor ND2 is shared among all of CAM cells compare-circuits. Fig.14 shows a block diagram of another example where CAM incorporating systems includes a 48 x 48 array of CAM cells. The 48 rows represent a 48 bit word width, and the 48 columns represent 48 words. The proposed design can be expanded to more entry and wider bits. This design example is to illustrate the circuit approach and multiple instances can be used to realize bigger CAM block.

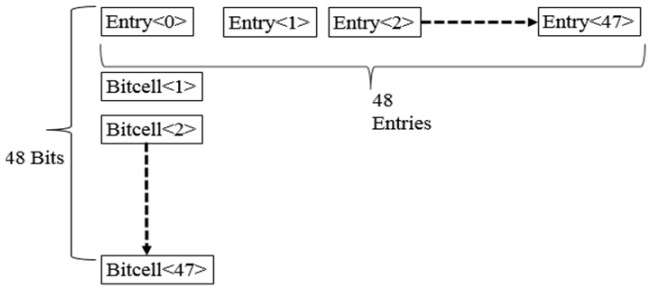


Fig. 14: Proposed block diagram for circuit using 48x48 CAM cells.

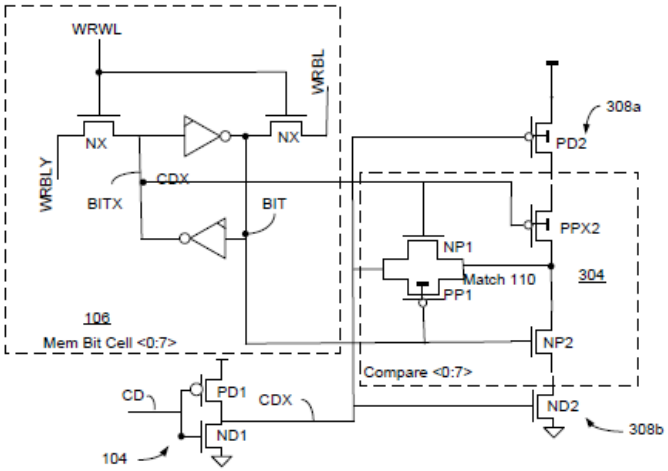


Fig. 15: Proposed circuit diagram of an example CAM system having multiple CAM cells.

Compare to the conventional Intel CAM cell implementation shown in Fig.7 which uses CD and its complement CDX to compare bit cell, the proposed CAM cell implementation which uses only the complement CDX as shown in Fig.15 have the following advantages:

- Eliminate the CD signal and thus save precious routing track and power from not having to switch it. Although CD driver is up sized to drive additional gate load in CAM bit cell, but the overall power savings are attained due to route (and its switching power) elimination.
- Reduce power consumption when the stored bit is 0 and thus the CAM operation is performed by effectively switching less cap. Fig.13 illustrates four cases of charging and discharging paths during CAM operation (CD switch from 0 1) when bit=0, and bit=1, respectively.
- Achieves ISO cell delay and output slope in configurations where CD inverter drives single or 8 bits cells. Cell delays and output slopes improve in case using of 16 bits cell configuration. The 8 bits-cell configuration is shown in Fig.12.

V. SIMULATION AND RESULTS

In this section, several conducted simulation experiments were made to compare the proposed implementation against the conventional Intel CAM cell implementation through different performance measures including current saving, cell area, delay and slop. The simulations performed using 22nm process technology and done under spice software. It is worthily mentioned that the proposed CAM implementation is based on real product design using CMOS technology. The memrstor technology is still in research and it is not well understood. In addition, the cost of adding memristor to typical CMOS technology is not small, hence it is not fair to compare the all CMOS solution with new technology that is not in use.

Fig.16 shows the simulation results waveform for both CAM cell implementations on single CAM bit-cell as described in Fig.7 and Fig.11. The 3rd chart in Fig.16 shows the current for both the proposed (Cyan) and original (yellow) designs. Table II shows the average current savings of 45% for Non Write CAM operation and 41% during Write operation for typical usage of this structure in Intel Atom cores front end cluster using one write followed by 20 CAM operations. Although the proposed implementation has a good current saving, it has no performance impact as shown in Table III. Also, addition of the two devices (ND2 and PD2) per CAM bit-cell increase its total device width by 6%, but it does not increase the net cell area. Even though there is a growth in the single CAM bit-cell area, the block area does not grow as column width since in this array implementations, it is set by metal routing and spacing for word lines above the cell.

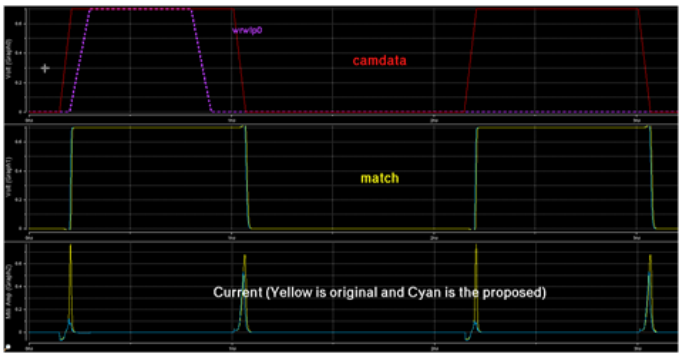


Fig. 16: Simulation waveform for both approaches on single CAM bit-cell.

TABLE II: Current saving for single CAM bit-cell for both implementation on Intel Atom cores front end cluster using one write followed by 20 CAM operations

Bit is = 1			
	Original	Proposed	Current Saving
Average Current No Write	11.10	6.64	40%
Average Current During Write	18.15	12.26	32%
Bit is = 0			
Average Current No Write	11.03	5.51	50%
Average Current During Write	14.4	7.32	49%
Average Current No Write			45%
Average Current During Write			41%

TABLE III: Delay comparison (ps) between original and proposed designs

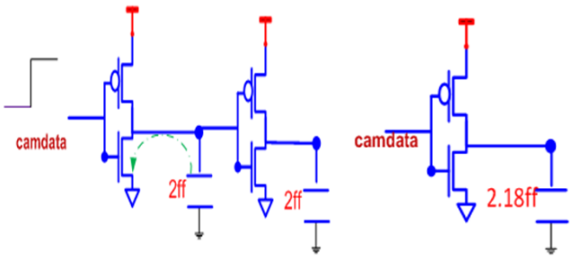
	Original	Proposed
Delay	40.9	39.6
Match_slope_fall	7.47	7.9
Match_slope_rise	3.67	3.94

Table IV shows the simulation results for both approaches on more realistic case in which 8 CAM bit-cells are connected as described in Fig.8 and Fig.12. The top level effective load models used are shown in figure Fig.17. The results in Table IV shows the average current savings of the proposed CAM cell

TABLE IV: Current saving for 8 cells with different cap and different stored bit value (1 write and 20 CAM operation)

Bit=1	**Cap (xff)**			**Cap (2xff)**		
	Original (2ff)(uA)	**Proposed (2.18ff)(uA)**	**Saving %**	**Original (2ff)(uA)**	**Proposed (2.18ff)(uA)**	**Saving %**
Average Current No Write (CAM Operation)	26.09	24.45	6%	26.92	24.77	8%
Average Current During Write	63.30	59.81	6%	64.38	60.23	6%
Bit=0	**Cap (xff)**			**Cap (2xff)**		
Average Current No Write (CAM Operation)	27.89	22.57	19%	30.70	23.92	22%
Average Current During Write	36.24	29.62	18%	36.24	29.62	18%
Average Current Saving No Write (CAM Operation)			13%			15%
Average Current Saving During Write			12%			12%

implementation for typical usage of this structure in processor cores front end cluster for one write followed by 20 CAM operations. The proposed implementation on average saves 13% of the current on No Write CAM operation for Bit=0 and Bit=1 on cap 2ff model and saves 15% of the current on No Write CAM operation for Bit=0 and Bit=1 on cap 2xff mode. Also, the proposed implementation on average saves 12% of the current during Write CAM operation for Bit=0 and Bit=1 on cap 2ff model and saves 12% of the current during Write CAM operation for Bit=0 and Bit=1 on cap 2xff mode. For mobile processor Front End Cluster (FEC) usage model (48 entries by 48 bits CAM cell structure) as shown in Fig.14, the proposed implementation has a total current saving of 985uA during CAM operation and 283uA during write operation as shown in Table V. So, as the number of bits increasing, current will always be less than the traditional approach since the selection logic circuit that select the number of bits, can be tuned to address any issue while current increase.



(a) **(b)**

Fig. 17: Model for 8 cells ((a)2ff is routing Cap and (b)2.18ff is gate and diffusion Cap).

In addittion, using the two devices ND2 and PD2 that can be shared across 8 cells as shown in Fig.12 reduces the total device width (Z) penalty per cell and thus it yields current savings of 14% as tabulated in Table VI with no performance degradation. The only disadvantage of this option is that the diffusion node is routed across the 8 cells leaving it susceptible to noise injection. The delay and slope(raise and fall) of the two approaches are similar with minor improvement in the proposed implementation as reported in Table VII.

TABLE V: Total Current saving example for 48 bit in 48 entries (1 write and 20 CAM operation)

		48 Entry	**48 Bit**
	Total Current Saving (mA)	**Total (uA)**	**Total (uA)**
CAM operation	0.0034202	0.985023	
Write Operation	0.0059166		0.283997

TABLE VI: Non-typical method current saving

Method 2 (8 cells share 2 added FET's)			
Bit=0			
	Original (uA)	**Proposed (uA)**	**Current Improvement**
Average Current No Write (CAM Operation)	27.89	20.99	25%
Average Current During Write	36.24	27.67	24%
Bit=1			
Average Current No Write (CAM Operation)	28.14	27.41	3%
Average Current During Write	65.97	63.30	4%
Average Current Saving No Write (CAM Operation)			14%
Average Current Saving During Write			14%

TABLE VII: Non-typical method Delay and slope

8 Cells with Shared FET					
Delay CAM data match orig (ps)	**Delay CAM data match prop(ps)**	**Slop fall orig(ps)**	**Slop rise orig(ps)**	**Slop rise prop(ps)**	**Slop fall prop(ps)**
40.42	38.18	10.28	5.99	5.47	9.48

So, based on the conducted experiment made and the results reported, the proposed CAM cell implementation have the following contributions:

- The proposed implementation saves routing tracks per cell. For example, it can save 4 tracks in a CAM system with 4-CAM cell.
- Power saving increases if the architecture requires more writes to the CAM cell.
- Reduce the amount of power:
  1) Saves on average 40% of the current per one cell.
  2) In more realistic case (8-cells CAM system), it saves on average 14% of the current on processor cores front end cluster for one write followed by 20 CAM operations.

3) Saves on average 0.009 mW power across all FEC CAM blocks (assuming AF=2%, 1 write 20 CAM).

- Single cell area has not changed and stayed as before.
- It has a minor improvement on delay and slope.
- It has to route the diffusion node across 8 cells.

VI. Conclusion

A Content Addressable Memory (CAM) is a memory unit that performs content matching instead of address decoding. CAM's are used in applications that required hiegh speed lookup operations like network routers and cache controllers. However, the CAM search speediness comes at the cost of higher power consumption and larger silicon area. CAM system includes CAM cells, each having a compare-circuit and a memory bit-cell that stores complementary bits. Conventional CAM bit-cell implementation produces crowbar current during bit-cell write operation since there is one gate delay between BIT and BITX which opens one pass gate before closing the second. In this paper, a novel CMOS CAM bit-cell implementation is presented to handle this problem and also to eliminates the need for routing CD and CDX to every CAM cell through using single-ended reference data (i.e., the inverted reference bit CDX) instead of the complementary reference data. In addittion, sharing resources across all cells reduces the total device width. The new implementation outperforms the conventional CAM cell implementation in terms of current saving without any performance impact in terms of area and delay.

References

[1] V. Tabatabaee, "Content addressable memories." http://www.isr.umd.edu/~vahidt/CAM.ppt.

[2] M. A. Scott Beamer, "Design of a low power content addressable memory (cam)," May 2009.

[3] K. Pagiamtzis, "Introduction to content-addressable memory (cam)." https://www.pagiamtzis.com/cam/camintro/.

[4] K. Pagiamtzis and A. Sheikholeslami, "Content-addressable memory (cam) circuits and architectures: a tutorial and survey," *Solid-State Circuits, IEEE Journal of*, vol. 41, pp. 712–727, March 2006.

[5] B. Mohammad, P. Bassett, J. Abraham, and A. Aziz, "Cache organization for embeded processors: cam-vs-sram," in *2006 IEEE International SOC Conference*, pp. 299–302, IEEE, 2006.

[6] B. Mohammad and J. Abraham, "A reduced voltage swing circuit using a single supply to enable lower voltage operation for sram-based memory," *Microelectronics Journal*, vol. 43, no. 2, pp. 110–118, 2012.

[7] B. Mohammad, M. T. Rab, K. Mohammad, and M. A. Suleman, "Dynamic cache resizing architecture for high yield soc," in *2009 IEEE International Conference on IC Design and Technology*, pp. 211–214, IEEE, 2009.

[8] D. GEORGIE, "Low power concept for content addressable memory (cam) chip design," *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, 2013.

[9] A. Patwary, H. Greub, Z. Wang, and B. Geuskens, "Bit-line organization in register files for low-power and high-performance applications," in *Electrical and Computer Engineering, 2006. ICECE '06. International Conference on*, pp. 505–508, Dec 2006.

[10] H. Miyatake, M. Tanaka, and Y. Mori, "A design for high-speed low-power cmos fully parallel content-addressable memory macros," *Solid-State Circuits, IEEE Journal of*, vol. 36, pp. 956–968, Jun 2001.

[11] K. J. Schultz, "Content-addressable memory core cells a survey," *Integration, the {VLSI} Journal*, vol. 23, no. 2, pp. 171 – 188, 1997.

[12] M. Elgebaly, *Energy Efficient Design for Deep Sub-micron CMOS VLSIs*. PhD thesis, University of Waterloo, Waterloo, Ontario, Canada, 2005.

[13] A. Alvandpour, R. Krishnamurthy, K. Soumyanath, and S. Borkar, "A low-leakage dynamic multi-ported register file in 0.13mm cmos," in *Proceedings of the 2001 International Symposium on Low Power Electronics and Design*, ISLPED '01, (New York, NY, USA), pp. 68–71, ACM, 2001.

[14] S. Thompson, I. Young, J. Greason, and M. Bohr, "Dual Threshold Voltages And Substrate Bias: Keys To High Performance, Low Power, 0.1 m Logic Designs," in *VLSI Technology, Symposium*, 1997.

[15] A. Patwary, B. Geuskens, and S. Lu, "Content addressable memory for low-power and high-performance applications," in *Computer Science and Information Engineering, 2009 WRI World Congress on*, vol. 3, pp. 423–427, March 2009.

[16] M. S. M. T. Sudalaimani1, "Design of high speed low power content addressable memory (cam) using parity bit and gated power matchline sensing," *International Journal for Research in Applied Science and Engineering Technology*, vol. 3, March 2015.

[17] S.-J. Ruan, C.-Y. Wu, and J.-Y. Hsieh, "Low power design of precomputation-based content-addressable memory," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 16, pp. 331–335, March 2008.

[18] I. Corporation, "Low capacitance content-addressable memory cell," 01 1996.

[19] I. Corporation, "Content addressable memory constructed from random access memory," 10 2005.

[20] O. K. S.-K. K. D. A. Kamran Eshraghian, Kyoung-Rok Cho and S.-M. S. Kang, "Memristor mos content addressable memory (mcam): Hybrid architecture for future high performance search engines," *IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS*, vol. 19, August 2011.

[21] R. S. C. M. O. Yuanfan Yang, Jimson Mathew and D. K. Pradhan, "Low cost memristor associative memory design for full and partial matching applications," *IEEE TRANSACTIONS ON NANOTECHNOLOGY*, vol. 15, May 2016.

[22] K. M. A. Kabeer and T. Taha, "On-chip power minimization using serialization-widening with frequent value encoding," *VLSI Design*, 2014.

[23] D. Georgiev, "Low power concept for content addressable memory (cam) chip design," *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, vol. 2, July 2013.

[24] K. Pagiamtzis and A. Sheikholeslami, "A low-power content-addressable memory (CAM) using pipelined hierarchical search scheme," *IEEE Journal of Solid-State Circuits*, vol. 39, pp. 1512–1519, September 2004.

[25] K. Pagiamtzis and A. Sheikholeslami, "Pipelined match-lines and hierarchical search-lines for low-power content-addressable memories," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC).*, p. 383386, IEEE, 2003.

[26] C.-S. L. . J.-C. C. . B.-D. Liu, "A low-power content-addressable memory (cam) using pipelined hierarchical search scheme," *IEEE J. Solid-State Circuits*, vol. 39, pp. 1512–1519, September 2004.

[27] J.-C. C. C.-S. Lin and B.-D. Liu, "Design for low-power, low-cost, and high-reliability precomputation-based content-addressable memory," in *Proc. Asia-Pacific Conf. Circuits Syst.*, p. 319324, Circuits and Systems, 2002.

[28] C.-S. L. . J.-C. C. . B.-D. Liu, "A low-power precomputation-based fully parallel content-addressable memory," *IEEE J. Solid-State Circuits*, vol. 38, pp. 654–662, April 2003.