DESIGN OF PARAMETER EXTRACTOR IN LOW POWER PRECOMPUTATION BASED CONTENT ADDRESSABLE MEMORY

Saroja pasumarti,
Asst.professor,
Department Of Electronics and Communication Engineering,
Chaitanya Engineering College,
Andhra Pradesh, India.
Saroja.pasumarti@gmail.com.

Lakshmi Kadali,
Asst.Professor,
Department Of Electronics and Communication Engineering,
Chaitanya Engineering College,
Andhra Pradesh, India.
kadalilakshmi@gmail.com.

Swathi.Bankapalli
Asst.Professor,
Department Of Electronics and Communication Engineering,
Chaitanya Engineering College,
Andhra Pradesh, India.
Swathi.bankapalli@gmail.com

Abstract-- Content-addressable memory (CAM) is frequently used in applications, such as lookup tables, databases, associative computing, and networking, that require high-speed searches due to its ability to improve application performance by using parallel comparison to reduce search time. Although the use of parallel comparison results in reduced search time, it also significantly increases power consumption. In this paper, we propose a Block-XOR approach to improve the efficiency of low power pre computation- based CAM (PB-CAM). Through mathematical analysis, we found that our approach can effectively reduce the number of comparison operations by 50% on average as compared with the ones-count approach for 15-bit-long inputs. In our experiment, we used Synopsys Nanosim to estimate the power consumption in TSMC 0.35- m CMOS technology. Compared with the ones-count PB-CAM system, the experimental results show that our proposed approach can achieve on average 30% in power reduction and 32% in power performance reduction. The major contribution of this paper is that it presents theoretical and practical proofs to verify that our proposed Block-XOR PB-CAM system can achieve greater power reduction without the need for a special CAM cell design. This implies that our approach is more flexible and adaptive for general designs.

Key Words—Content-addressable memory (CAM), low-power, pre computation.

I. INTRODUCTION

Most memory devices store and retrieve data by addressing specific memory locations. As a result, this path often becomes the limiting factor for systems that rely on fast memory accesses. The time required to find an item stored in memory can be reduced considerably if the item can be identified for access by its content rather than by its address. A memory that is accessed in this way is called content-addressable memory or CAM.

It provides a performance advantage over other memory search algorithms, such as binary or tree-based searches or look-aside tag buffers, by comparing the desired information against the entire list of pre-stored entries simultaneously, often resulting in an order-of-magnitude reduction in the search time.

Classically a CAM is defined as a functional memory with a large amount of stored data that simultaneously compares the input search data with the stored data. Once matching data are found, their addresses are returned as output as shown in figure 1.1.

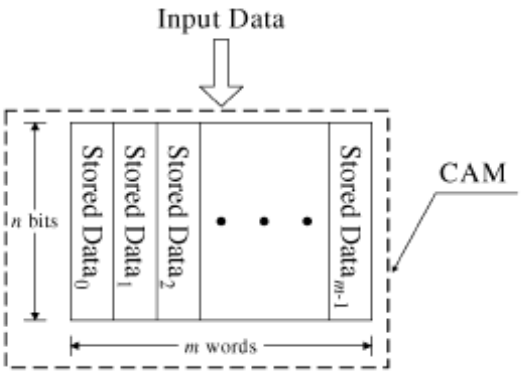

Figure1.1 .conventional CAM architecture

Unlike standard computer memory (random access memory or RAM) in which the user supplies a memory address and the RAM returns the data word stored at that address, a CAM is designed such that the user supplies a data word and the CAM searches its entire memory to see if that data word is stored anywhere in it. If the data word is found, the CAM returns a list of one or more storage addresses where the word was found (and in some architecture, it also returns the data word, or other associated pieces of data). Thus, a CAM is the hardware embodiment of what in software terms would be called an associative array.

Content-addressable memories (CAMs) are hardware search engines that are much faster than algorithmic approaches for search-intensive applications. CAMs are composed of conventional semiconductor memory (usually SRAM) with added comparison circuitry that enables a search operation to complete in a single clock cycle. CAM can be used as looks up table that suite for AES applications.

A content-addressable memory (CAM) is a critical device for applications involving asynchronous transfer mode (ATM), communication networks, databases, lookup tables, tag directories, data compression, pattern-recognition and security or encryption information on a packet-by-packet basis for high-performance data switches, firewalls, bridges and routers due to its high-speed data search capability. The vast number of comparison operations required by CAMs consumes a large amount of power. So pre computation techniques have evolved.

II BACKGROUND

Since content addressable memory (CAM) is frequently used in applications, that require high-speed searches, and because of its ability to improve application performance by using parallel comparison, it results in reduced search time. But it also significantly increases power consumption. So the main CAM-design challenge is to reduce power consumption associated with the large amount of parallel active circuitry, without sacrificing speed or memory density.

2.1 Power saving CAM architecture:

Architectural technique for saving power, which applies to binary CAM, is pre-computation. Pre-computation stores some extra information along with each word that is used in the search operation to save power. These extra bits are derived from the stored word, and used in an initial search before searching the main word. If this initial search fails, then the CAM aborts the subsequent search, thus saving power.

2.2 PB-CAM Architecture

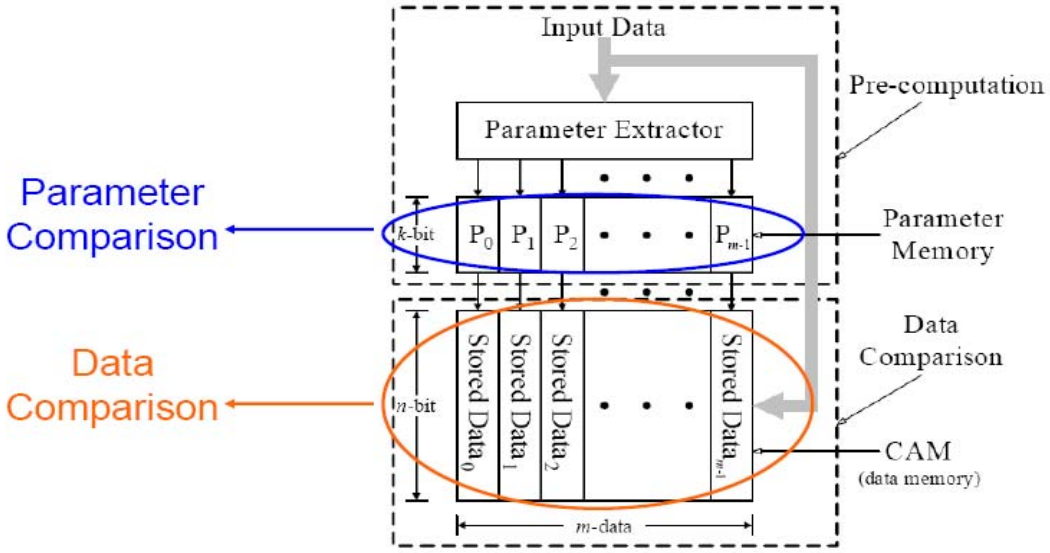

Figure.2.1.memory organization of PB-CAM architecture

Figure 2.1 shows the memory organization of the PB-CAM architecture which consists of data memory, parameter memory, and parameter extractor, where k << n. To reduce massive comparison operations for data searches, the operation is divided into two parts. In the first part, the parameter extractor extracts a parameter from the input data, which is then compared to parameters stored in parallel in the parameter memory. If no match is returned in the first part, it means that the input data mismatch the data related to the stored parameter. Otherwise, the data related to those stored parameters have to be compared in the second part. It should be noted that although the first part must access the entire parameter memory, the parameter memory is far smaller than that of the CAM (data memory). Moreover, since comparisons made in the first part have already filtered out the unmatched data, the second part only needs to compare the data that match from the first part.

The PB-CAM exploits this characteristic to reduce the comparison operations, thereby saving power. Therefore, the parameter extractor of the PB-CAM is critical, because it determines the number of comparison operations in the second part. So, the parameter extractor plays a significant role since this circuit determines the number of comparison operations required in the second part. Therefore, the design goal of the parameter extractor is to filter out as many unmatched data as possible to minimize the required number of comparison operations in the second part. Two parameter extractors are discussed, namely One's count parameter extractor and Block-XOR parameter extractor.

2.2.1 Design concept of the PB-CAM:

A general CAM architecture usually consists of data memory with valid bit field, address decoder, bit line pre charger, word match circuit, and address priority encoder, as shown. The memory organization of the traditional CAM consists of the data memory and the valid bit field, where the valid bit field indicates the availability of stored data. In the data searching operation, the input data is sent into CAM to compare with all valid data stored in CAM simultaneously, and an address from among those matches of comparison is sent to the output. In this architecture, the CAM circuit performs large amount of comparison operations to identify all valid data stored in CAM during each data searching operation. This comparison consumes most of the total CAM power.

To minimize power consumed during the comparison, one of the best approaches is to reduce most of the comparison operations. Based on this idea, a novel architecture is developed for low-power CAM circuit design called PB-CAM. To address the proposed low-power PB-CAM architecture, the design concept for this architecture is introduced. The memory organization of the proposed PB-CAM architecture, as shown in Fig. 2, is composed of the data memory, the parameter memory, and the parameter extractor. In the data writing

operation, the parameter extractor extracts the parameter of the input data, and then stores the input data and its parameter into the data memory and the parameter memory, respectively.

In the data searching operation, in order to reduce the large amount of comparison operations, the operation is separated into two comparison processes. In the first comparison process, the parameter extractor extracts the parameter of the input data, and the parameter comparison circuits then compare the parameter of the input data with all parameters stored in the parameter memory in parallel. The data related to this stored parameter concurrently mismatches the input data, if the stored parameter mismatches the parameter of the input data. Otherwise, the data related to this stored parameter has yet to be identified. Using the first comparison process results, the input data is only compared with those unidentified data to identify any match in the second comparison process.

Based on the two comparison processes, if a majority part of the stored parameter mismatch the parameter of the input data, then the number of comparisons in the second comparison process is largely reduced. The function of this parameter comparison process data in the first comparison process and then reduces most of the comparisons in second comparison process.

In the proposed PB-CAM architecture, the parameter extractor dominates the majority parts of comparison power, since this circuit decides the number of unidentified data remaining after the parameter comparison process. In addition, both parameter memory and parameter comparison circuit of the PB-CAM architecture need extra hardware cost and power dissipation compared with traditional CAM architecture. Therefore, the design idea of the parameter extractor is to filter as many unmatched data in the parameter comparison process with the probable shortest bit length of the parameter. Some functions are used to realize the parameter extraction in the proposed CAM architecture, such as ones count function, parity function, and remainder function. In this paper, the PB-CAM architecture adopts ones count function to perform the parameter extraction, because the ones count function not only filters a large amount of unmatched data with a small bit length, but also reduces the transistor count of the proposed PB-CAM cell to seven transistors

2.2 One's count approach:

For ones count approach, with an n-bit data length, there are n+1 types of ones count (from 0 ones to n ones count). Further, it is necessary to add an extra type of ones count to indicate the availability of stored data. Therefore, the minimal bit length of the parameter is equal to log (n+ 2). The below fig 5 shows the conceptual view of one's count approach. The extra information holds the number of ones in the stored word. For example, in fig.10, when searching for the data word, 01001101, the pre-computation circuit the number of ones (which is four in this case). The number four is compared on the left-hand side to the stored one's count. Only match lines PML_5 and PML7 match, since only they have a one's count of four.

In the data-memory stage in figure3.2, only two comparisons actively consume power and only match line PML5 results in a match. The 14-bit ones-count parameter extractor is implemented with full adders as shown in Fig. 3.3.

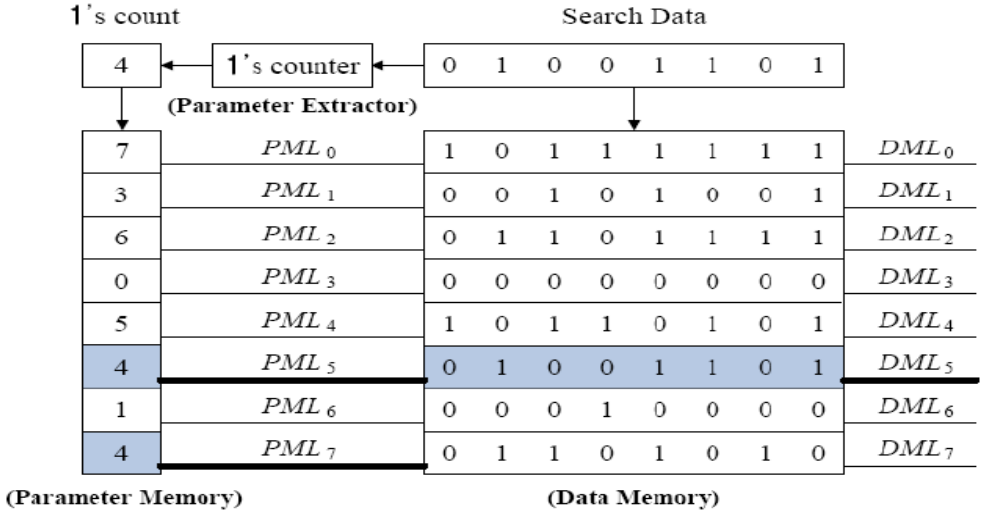

Figure 2.2. conceptual view of one's count approach

2.2.1 Mathematical Analysis:

For a 14-bit length input data, all the input data contain 2^{14} numbers, and the number of input data related to the same parameter for ones count approach is $\binom{14}{n}$, where n is a type of ones-count (from 0 to 14 ones-counts). Then we can compute the average probability that the parameter occurs. The average probability can be determined by

$$\text{Average probability} = \frac{\binom{14}{n}}{2^{14}}. \quad (1)$$

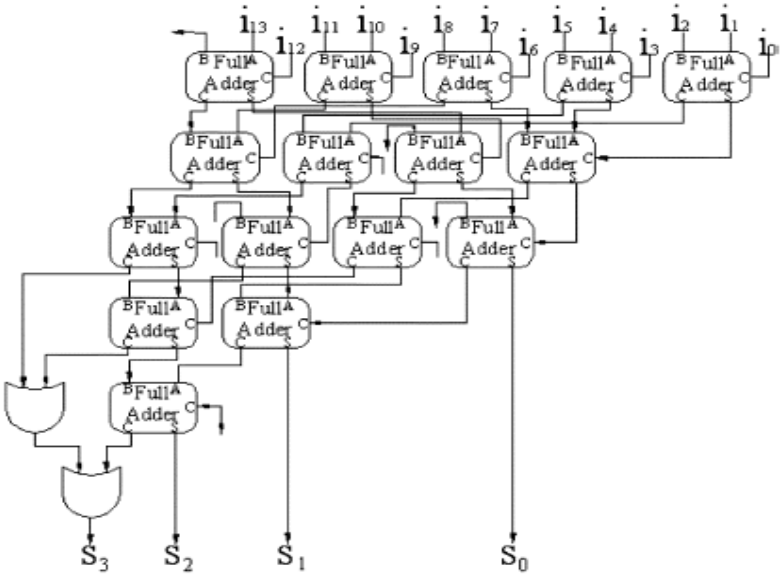

Figure 2.3 :14-bit ones-count parameter extractor

Table I
Number Of Data Related To The Same Parameters And Average Probabilities For The Ones Count Approach

Parameter		Number of data related to the same parameter	Average probability
0000	0	1	0.01%
0001	1	14	0.09%
0010	2	91	0.56%
0011	3	364	2.22%
0100	4	1001	6.11%
0101	5	2002	12.22%
0110	6	3003	18.33%
0111	7	3432	20.95%
1000	8	3003	18.33%
1001	9	2002	12.22%
1010	10	1001	6.11%
1011	11	364	2.22%
1100	12	91	0.56%
1101	13	14	0.09%
1110	14	1	0.01%
1111	15	valid bit	

Table I lists the number of data related to the same parameter and their average probabilities for the input data that is 14-bit in length. For example, if a match occurs in the first part of the comparison with the parameter 2, the maximum number of required comparison operations for the second part is $\binom{14}{2} = 91$.With conventional CAMs, the comparison circuit must compare all stored data, whereas with the ones-count PB-CAMs, a large amount of unmatched data can be initially filtered out, reducing comparison operations for minimum power consumption in some cases. However, the average probabilities of some parameters, such as 0, 1, 2, 12, 13, and 14 are less than 1%.

In Table I, parameters with over 2000 comparison operations range between 5 and 9. However, the summation of the average probabilities for these parameters is close to 82%. Although the number of

comparison operations required for ones-count PB-CAMs is fewer than that of conventional CAMs, ones-count PB-CAMs fail to reduce the number of comparison operations in the second part when the parameter value is between 5 and 9, thereby consuming a large amount of power. From the Table I we can see that random input patterns for the ones-count approach demonstrate the Gaussian distribution characteristic. The Gaussian distribution will limit any further reduction of the comparison operations in PB-CAMs.

2.3 Block –XOR approach:

The key idea behind this method is to reduce the number of comparison operations by eliminating the Gaussian distribution. For a 14-bit input data, if we can distribute the input data uniformly over the parameters, then the number of input data related to each parameter would be $2^{14}/15 = 1093$, and the maximum number of required comparison operations would be $2^{14}/15 = 1093$ for each case in the second part of the comparison process. Compared with the ones-count approach, this approach can reduce comparison operations by a minimum of 909 and a maximum of 2339 (i.e., for parameter value is from 5 to 9) for 82% of the cases. Based on these observations, a new parameter extractor called Block-XOR, which is shown in Figure 2.4, is used to achieve the previous requirement.

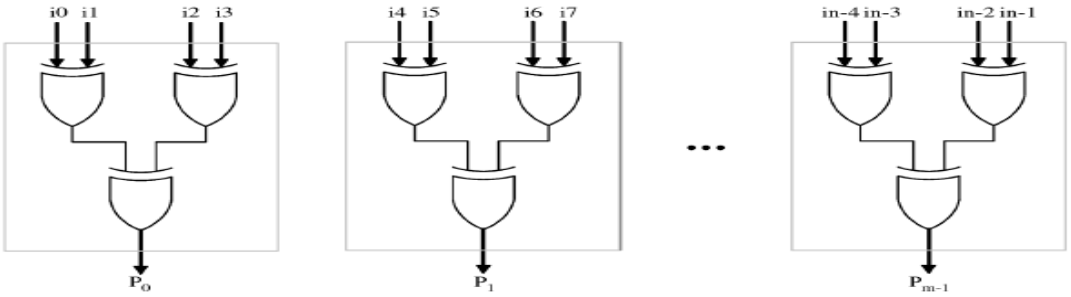

Figure 2.4.concept of n-bit Block-XOR block diagram.

In this approach, we first partition the input data bit into several blocks, from which an output bit is computed using XOR logic operation for each of these blocks. The output bits are then combined to become the input parameter for the second part of the comparison process. To compare with the ones-count approach, we set the bit length of the parameter to [log (n+ 2)]. Where n is the bit length of the input data. Therefore, the number of blocks is [n/ log (n+2)] in this approach. Taking the 14-bit input length as an example, the bit length of the parameter is log (14+2) = 4-bit, and the number of blocks is [14/ log(14+2)] = 4 . Accordingly, all the blocks contain 4 bits except the last one, which contains the remainder 2 bits as shown in the upper part of Figure 2.5.

However, the concept of Block-XOR approach does not provide a, valid bit for checking whether the data is valid; hence it cannot be applied to the PB-CAM directly. For this reason, modified architecture is used as shown in the lower part of Figure 2.5 to provide a valid bit and to guarantee the uniform distribution property of the Block-XOR approach. We added a multiplexer to select the correct parameter.

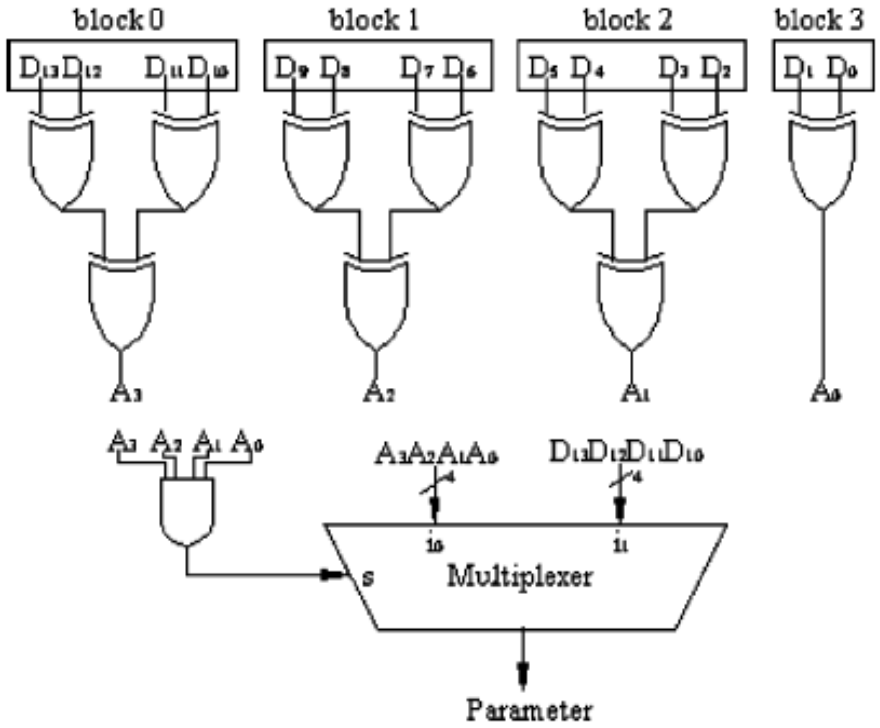

Figure 2.5: Structure of Block-XOR approach with valid bit.

The selected signal is defined as

S = A3A2A1A0 . …… (2)

According to (2), if the parameter is "0000 to 1110" (S = "0"), the multiplexer will transmit the i0 data as the output. In other words, the parameter does not change. Otherwise, (A3A2A1A0 =''1111", S ='‘1"), the first block of the input data becomes the new parameter, and "1111" can then be used as the valid bit. The case where the first block is "1111" was not considered, because the "1111" block bits will result in "0" for one of the four parameter bits.

2.3.1 Mathematical Analysis:

The concept of Block-XOR approach is to uniformly distribute the parameter over the input data. By the rule of product, the number of input data that results in the same parameter (without valid bit) is 8 *8*8*2 = 1024. Consequently, the average probability can be determined as 1024/(1024*16)*100% = 6.25%. Accordingly, the maximum number of comparison operations is 1024 for each parameter in the second part. Obviously, the concept of Block-XOR approach can reduce the comparison operations, hence minimize power consumption.

Table II
Number Of Data Related To The Same Parameters And Average Probabilities For The Block -XOR Approach

Parameter		Number of data related to the same parameter	Average probability
0000	0	1024	6.25%
0001	1	1152	7.03%
0010	2	1152	7.03%
0011	3	1024	6.25%
0100	4	1152	7.03%
0101	5	1024	6.25%
0110	6	1024	6.25%
0111	7	1152	7.03%
1000	8	1152	7.03%
1001	9	1024	6.25%
1010	10	1024	6.25%
1011	11	1152	7.03%
1100	12	1024	6.25%
1101	13	1152	7.03%
1110	14	1152	7.03%
1111	15	valid bit	

Table II lists the number of input data that result in the same parameter for the proposed Block-XOR PB-CAM (i.e., with valid bit). When the parameter is "1111", the new parameter is provided by the first block with an output bit of "1" so that the number of input data for those parameters is 1024 + (1024/8) = 1152, and the average probability is (1152/ (1024 * 7 + 1152 * 8)) * 100% = 7.03%. As can be seen from Tables I and II, the Block-XOR PB-CAM results in at least 850 fewer comparison operations in 82% of the cases. In other words, in most cases, the Block-XOR PB-CAM required far fewer comparison operations than the ones-count approach for parameter values between 5 and 9. For example, when the parameter is 7, the proposed Block-XOR PB-CAM requires 2284 fewer comparison operations than the ones-count approach.

III MAIN THRUST OF THE PAPER

To make the parameter extractor of the block-xor PB-BAM more useful for specific data types, we take into account the different characteristic of logic gates to synthesize the parameter extractors for different data types. As can be seen in Table I ,Table II. if the input bits of each partition block is set into *l*, the bit length of the parameter (i.e. the number of blocks) will be $\lceil n/l \rceil$, where *n* is the bit length of the input data, and then the levels in each partition block equal $\lceil log_2 l$ We observe that when the input bits of each partition block decreases, the mismatch rate and the number of comparison operations in each data comparison process will decrease (this is because that the Fig. 3.1. *n*-bit block diagram of the proposed parameter extractor architecture. Combinations of the parameter increase).

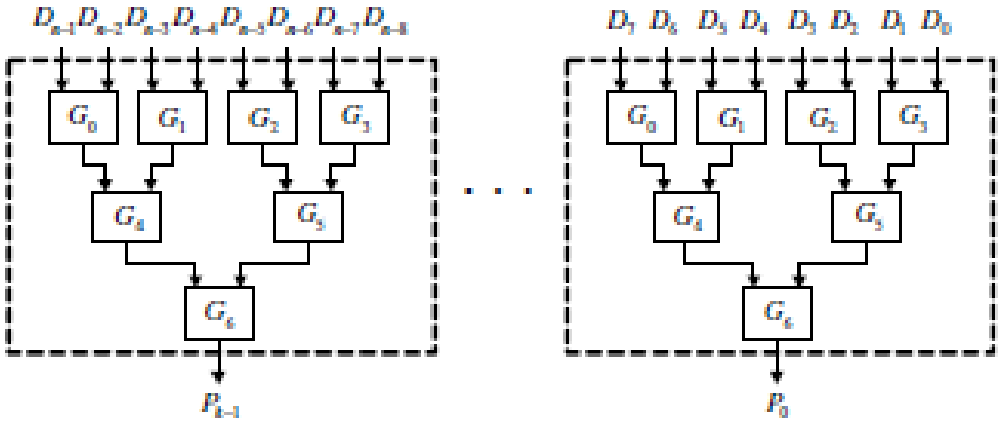

Figure 3.1 n-bit block diagram of the proposed parameter extractor architecture.

Although the increasing parameter bit length can decrease the mismatch rate and the number of comparison operations in each data comparison process, the parameter memory size must be increased. In other words, it increases the power consumption of the parameter memory as well. As we stated in Section II-A, when the PBCAM performs data searching operation, it must compare the entire parameter memory. To avoid wasting the large mount of power in the parameter memory, we set the input of each partition block to 8 bits. Fig. 3.1 shows the proposed parameter extractor architecture.

We first partition the input data bit into several blocks, G0~G6 in each block stand for different logic gates, from which an output bit is computed using synthesized logic operation for each of these blocks. The output bits are then combined to become the parameter for data comparison process. The objective of our work is to select the proper logic gates in Fig. 3.1 so that the parameter ($Pk-1$, $Pk-2$, $\cdots$, $P0$) can reduce the number of data comparison operations as many as possible. In our proposed parameter extractor, the bit length of the parameter is set into $\lceil n/8 \rceil$, and then the levels in each partition block equal $\lceil log_2 8 \rceil$ (which is 3). Suppose that we use basic logic gates (AND, OR, XOR, NAND, NOR, and NXOR) to synthesize a parameter extractor for a specific data type, which has $(6^7)^{\lceil n/8 \rceil}$ different logic combinations based on the proposed parameter extractor. Obviously, the optimal combination of the parameter extractor can not be found in polynomial time.

IV CONCLUSION

In this paper, a gate-block selection algorithm was proposed. The proposed algorithm can synthesize a proper parameter extractor of the PB-CAM for a specific data type. The experimental results confirmed that the proposed PB-CAM effectively save power by reducing the number of comparison operations in the data comparison process. In addition, the proposed parameter extractor can compute parameter bits in parallel with only three logic gate delays for any input bit length (i.e. constant delay of search operation). As shown in our experimental results, our proposed PB-CAM is very suitable for specific applications such as embedded systems.

V REFERENCES

[1] K. Pagiamtzis and A. Sheikholeslami, "Content-addressable memory (CAM) circuits and architectures:A tutorial and survey," *IEEE J. Solid- State Circuits*, vol. 41, no. 3, pp. 712–727, Mar. 2006.

[2] H. Miyatake, M. Tanaka, and Y. Mori, "A design for high-speed-low power CMOS fully parallel content-addressable memory macros," *IEEE J. Solid-State Circuits*, vol. 36, no. 6, pp. 956–968, Jun. 2001.

[3] I. Arsovski, T. Chandler, and A. Sheikholeslami, "A ternary content addressable memory (TCAM) based on 4T static storage and including a current-race sensing scheme," *IEEE J. Solid-State Circuits*, vol. 38,no. 1, pp. 155–158, Jan. 2003.

[4] I. Arsovski and A. Sheikholeslami, "A mismatch-dependent power allocation technique for match-line sensing in content-addressable memories," *IEEE J. Solid-State Circuits*, vol. 38, no. 11, pp. 1958–1966, Nov. 2003.

[5] Y. J. Chang, S. J. Ruan, and F. Lai, "Design and analysis of low power cache using two-level filter scheme," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 11, no. 4, pp. 568–580, Aug. 2003.

Saroja pasumarti born on 28th may 1986, she received B.Tech degree in Electronics and communication Engineering from Thandra Paparaya institute of science and technology in 2007 and M.Tech from Andhra university in 2011. She is currently working as Asst.Professor of ECE Dept in Chaitanya Engineering College.

Lakshmi kadali born on 29th feb 1984, she received B.Tech degree in Electronics and communication Engineering from PRAGATI Engineering college in 2006 and M.Tech from Chaitanya Engineering College. She is currently working as Asst.Professor Of ECE Dept in Chaitanya Engineering College.