DESIGN OF LOW POWER PRE-COMPUTATION BASED CAM USING XOR AND GATE BLOCK SELECTION SCHEME

Pallavi Shivatare[1], V.G.Raut[2]

[1] *PG Student, Department of E&TC, SCOE, Maharashtra, (India)*

[2]*Assistant Professor, Department of E&TC, SCOE, Maharashtra, (India)*

ABSTRACT

Content-addressable memory (CAM) is a special type of computer memory which provides a fast data search operation in single clock cycle. Content-addressable memory (CAM) is frequently used in applications such as lookup tables, databases, associative computing, and networking. However the high speed of CAM increases the power consumption. This paper presents low power techniques i.e. One's count, XOR approach and Gate block selection to improve power efficiency of pre-computation based CAM. In this experiment VHDL modeling is used and implemented using Xilinx to estimate the power consumption. Compared with the ones count PB-CAM system the experimental result shows that proposed approach can achieve 30 % power reduction.

Keywords: Content Addressable Memory, Gate Block Selection, One's Count, Pre-Computation, XOR

I INTRODUCTION

Content addressable memory (CAM) is the special type of memory in which data can be identified by its content instead of its address. CAM compares input search data with stored data, and returns the address of the matching data. CAM has fast search capability. CAMs can be used in a wide variety of applications such as asynchronous transfer mode (ATM), communication networks, databases, lookup tables, data compression. Due to its vast number of comparison operation CAM consumes large amount of power.

There are many methods for reducing CAM power consumption which are focused on reducing match-line power [3], [4], [12] or by dividing the match line [5], [6] ,[13],performance degradation, pre-computation techniques [1], [8], low power cache [9]-[11], search architecture [14], [15]. One's count pre-computation-based CAM [8] (PB-CAM) that achieves low-power. Furthermore to improve the efficiency of PB-CAMs, the parameter extractor architecture is modified.

In this work, two efficient techniques for reducing power for the PB-CAM.are Compared with the one's count technique. Proposed techniques achieve power reduction.

II CAM ARCHITECTURE

2.1 CAM overview

Content addressable memory (CAM) provides a fast data search function by comparing the search data with all of the stored data in a single clock cycle. CAMs can be used in a wide variety of applications requiring high search speeds. However due to high speed of a CAM, silicon area and power consumption increases. For lager CAM the power problem further increases. To reduce the power consumption the CAM can be modified at architecture level.

We can compare CAM to RAM. RAM produces the data for a given address. But in case of CAM, it produces an address for a given data word. For RAM, data search is done serially. Thus, finding a particular data word can take many cycles. In case of CAM searches all addresses in parallel and produces the address storing a particular word. CAM supports writing "don't care" bits into words of the memory. The don't care bit can be used as a mask for CAM comparisons. The output of the CAM can be encoded. It ensures duplicate data is not written into the CAM.

2.2 Operation of CAM

Fig.1 shows general CAM architecture. In this block, data is stored already in the CAM memory. The input to the system is search word. The input search data is broadcasted to searchlines and then it is compared with the stored data.

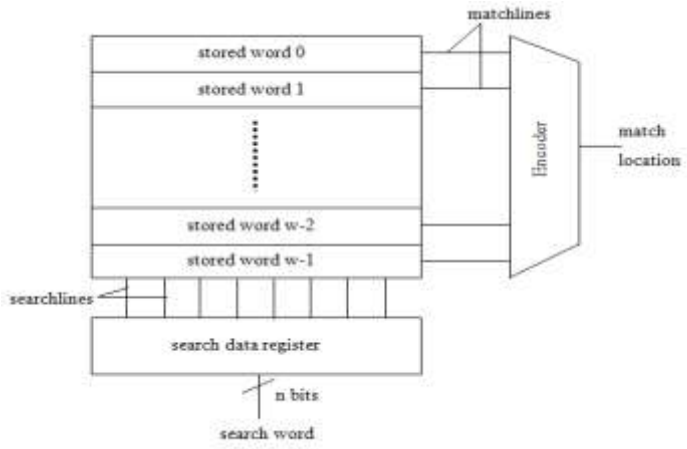

Fig.1 CAM Architecture [2]

Each stored word has matchline which indicates that whether the search data and stored data are same or different. Then matchlines are connected to encoder. The encoder gives the match location corresponding to match line which in match state. CAM does this searching operation at very high speed. So it requires large amount power. In order to reduce the power consumption we have introduced PB-CAM (pre computation-Based CAM) architecture.

III PRE-COMPUTATION BASED CAM ARCHITECTURE

Pre-computation is the technique applied to the CAM at architectural level. This technique stores some extra information along with each data. These extra information (extra bits) are generated from stored data and it is

used for initial search operation. Initially if this searching operation fails then it will stop further search and thus saves power.

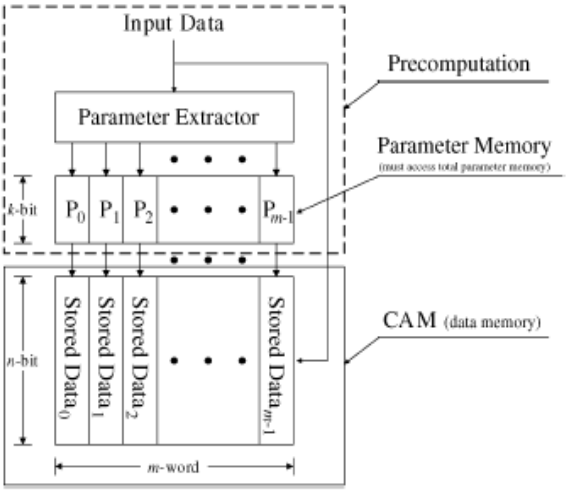

Fig.2 Pre-computation based CAM Architecture.

Fig. 2 shows the architecture of the PB-CAM which consists of data memory, parameter memory, and parameter extractor, where k < n. To reduce number of comparison for search operation, the operation is divided into two parts. The first consist of input data, parameter extractor and parameter memory, and the second part contains CAM memory (data memory). To reduce power consumption this pre computation-based content addressable memory is introduced as in the first part. For further pre-computation techniques here only the parameter extractor block is changed. 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 an important role.

3.1 One's count parameter extractor

Fig.3 shows the 14-bit ones-count parameter extractor. In this method input is 14 bit and it will produce 4-bit output (S_0, S_1, S_2, and S_3). The ones-count parameter extractor is implemented with full address. As name suggested it will count number of ones. If the output is 0101 means it will select parameter P_5 in parameter memory. If there is no match in the first part, it means that the input search data mismatches with thedata related to the stored parameter. Otherwise, the data related to those stored parameters have to be compared in the second part. Output of parameter extractor has been is given in Table I. The number of data for each parameter and its average probability is tabulated in Table 1.

The average probability can be determined by,

$$\text{Average probability} = \frac{\text{the number of input data related to the same parameter}}{\text{Total number of input data}} \qquad (1)$$

$$\text{Average propability} = \frac{\binom{14}{n}}{2^{14}}$$

Here n is a type of one's count (from 0 to 14 ones-counts).

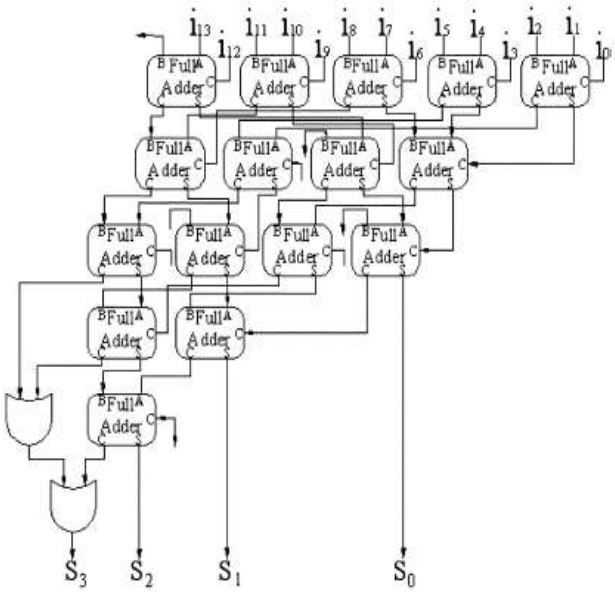

Fig.3 14-bit one's count parameter extractor

Table 1: Output of One's count parameter extractor

1 Parameter		No. 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	

For example average probability for parameter P_5 can be calculated as follows.

Number of data related to P_5 is

$\binom{14}{5}$ = 14*13*12*11*10/5*4*3*2*1 = 2002

The average probability = 2002 / 2^14

= 2002/1638 = 0.12219

Probability (%) = 12.22%

The disadvantage of one's-count technique is that from parameter 5 to 9 the number of data related to the same parameter is around 2000-3000. Hence comparison operations also increased simultaneously which will also increase the power consumption. Ones-count PB-CAMs fail to reduce the number of comparison operations in the second part. As it can be seen in Table I, random input patterns for the ones-count approach demonstrate the Gaussian distribution characteristic. Here the Gaussian distribution will limit any further reduction of the comparison operations in PB-CAMs. So, we replaced Block-XOR technique for reducing the amount power consumption.

IV PROPOSED PARAMETER EXTRACTOR ARCHITECTURE FOR PB-CAM

4.1 XOR Gate Parameter Extractor

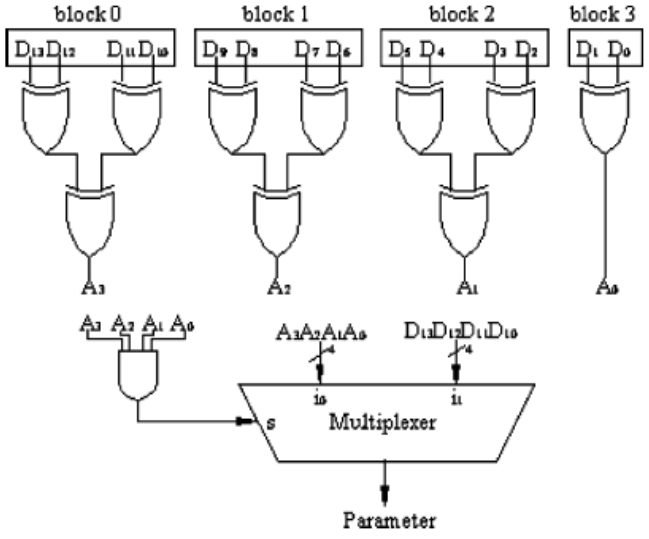

Fig.4 14-bit XOR parameter extractor

In this technique 14-bit input data is distributed uniformly over the parameters, then the number of input data related to each parameter would be $2^{14}/15$ =1093. So the maximum number of required comparison operations would be same i.e. 1093 for each case in the second part of the comparison process. This approach can reduce comparison operations by a several number of times as compared to one's count.

Fig.4 shows the block XOR parameter extractor. In this block XOR concept the input data is first is applied to several blocks. These blocks are several XOR logic combinations from which an output bit is computed. The output bits are then combined to become the input parameter for the second part of the comparison Process. However, the concept of Block-XOR approach does not provide a valid bit for checking whether the data is valid. For this reason the multiplexer is used. It will select the correct parameter.

The selected signal is defined as,

$$s=A_3.A_2.A_1.A_0 \qquad (2)$$

If the parameter is "0000 - 1110" (s = "0"), the multiplexer will transmit the i0 data as the output. In other words, the parameter will not change. Otherwise, ($A_3A_2A_1A_0$ =''1111", s ="'1"), the first block of the input data will become the new parameter, and "1111" can then be used as the valid bit. Note that the case where "1111" was not considered, because for case "1111", s='1' then I will select "D_{13} D_{12} D_{11} D_{10}". So then finally we have avoided the "1111". Table 2 shows the output of blocked XOR parameter extractor.

Table 2: Output of XOR extractor parameter

1 Parameter		No. 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	

4.2 Gate Block Selection Technique

In XOR gate PB-CAM parameter extractor we consider only XOR logic gate. So to make the parameter extractor more useful for specific data types, we consider different characteristic of logic gates. Fig.5 shows the proposed parameter extractor architecture. There are several partition block and each block contains several sub-blocks G0-G6. Each sub-block stands for different logic gates. Output bit is computed using synthesized logic operation for each of these sub-blocks. Then the output bits will become the parameter for data comparison process. The main objective is to select the proper logic gates in Fig.5 so that the parameter can reduce the number of data comparison operations.

For this parameter extractor, the bit length of the parameter is set into $[n/8]$, and then the levels in each partition block equal $[log_2 8]$(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. This parameter extractor will have different logical combinations. So to select appropriate optimal combination gate block algorithm is proposed. The mathematical analysis is given as below.

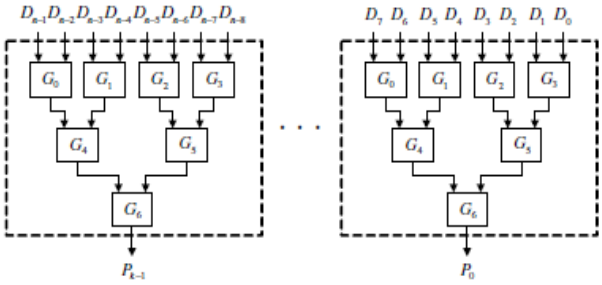

Fig.5 *n*-bit block diagram of the proposed parameter extractor architecture.

The probability function P of the output signal *Y* is given by,

$$P_Y(y) = \begin{cases} 1-p & y=0, \\ p & y=1, \\ 0 & otherwise \end{cases} \qquad (3)$$

Here 2-input logic gate is considered and p is the probability of output signal that is in one state. The average number of comparison operations in each data search operation can be calculated as,

$$C_{avg} = N_0(1-p) + N_1 .p$$

$$= N_0\left(\frac{N_0}{N_0+N_1}\right) + N_1\left(\frac{N_1}{N_0+N_1}\right)$$

$$= \frac{{N_0}^2 + {N_1}^2}{N_0+N_1} \qquad (4)$$

Where *N*0 is the number of zero*s*, and *N*1 is the number of ones.

Table 3 shows an example. Here 2 input logic gates are considered for parameter extractor and C_{avg} is calculated for each logic gate. The best selection for this case are OR and NOR gates because they require the least average number of comparison operations (which is 3). When we use the inverse relation of logic gates (NAND, NOR, XOR) to generate the parameter, the average number of comparison operations for each data search operation required in the PB-CAM will be the same. To reduce the complexity of proposed algorithm we select only NAND, NOR, and XOR gates to synthesize the parameter extractor for our implementation. Fig.6 shows our proposed gate block selection algorithm.

Table 3: Truth table and average no. of comparison operation of basic logic gates

A	*B*	*AND*	*OR*	*XOR*	*NAND*	*NOR*
0	0	0	0	0	1	1
1	0	0	1	1	1	0
1	0	0	1	1	1	0
0	0	0	0	0	1	1
1	1	1	1	0	0	0
0	0	0	0	0	1	1
C_{avg}		4.33	3	3.33	4.33	3

Algorithm to select proper logic gates for specific data :

Input data= $(D_0, D_1, \cdots, D_{n-1})$

n: bit length of the input data.
l: number of input bits for each partition block.

Step 1 : *Record*
$NAND\_parameter(k)=\overline{D_{2i} \cdot D_{2i+1}}$
$NOR\_parameter(k)=\overline{D_{2i} + D_{2i+1}}$
$XOR\_parameter(k)=D_{2i} \oplus D_{2i+1}$
for i, k=0, 1, · · ·, (n/2)-1, ∀ input patterns

Step 2 : *Compute*
$NAND\_C_{avg}(k)$
$NOR\_C_{avg}(k)$
$XOR\_C_{avg}(k)$
using Equ. 4, ∀ k

Step 3 : *Select a logic gate with the minimal* $C_{avg}(k)$, $\forall k$

Step 4 : *If generated parameter bits* $> \lceil n/l \rceil$,
repeat Step 1 to Step 3,
and use previous generated parameter as input data.
else
finish.

Fig.6 Gate-Block Selection Algorithm

V SIMULATION RESULTS

In this work the PB-CAM is simulated using VHDL. The waveforms for XOR and gate block selection techniques are as shown in Figures (7, 8, 9 and 10).

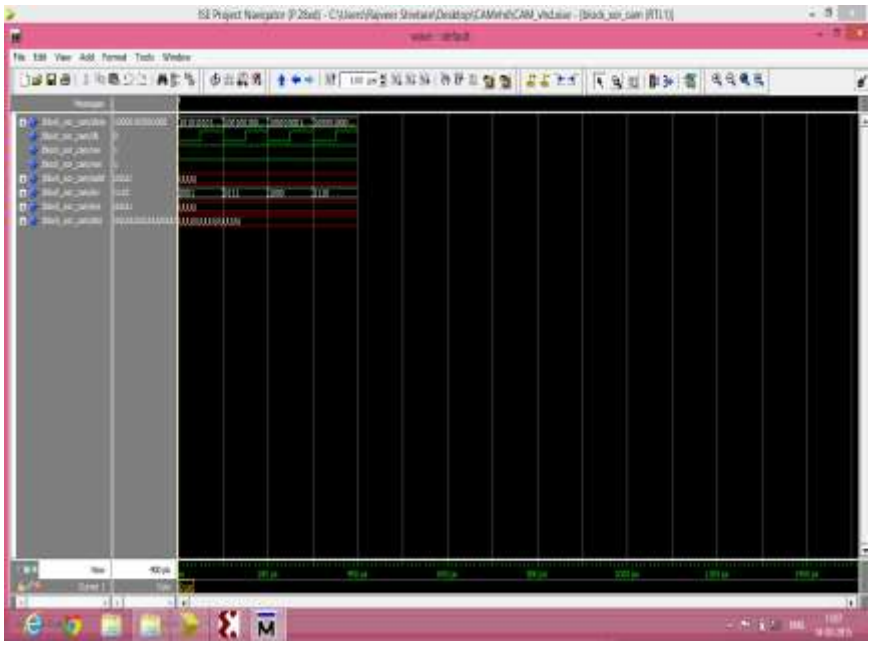

Fig.7 Data write operation for CAM using XOR technique

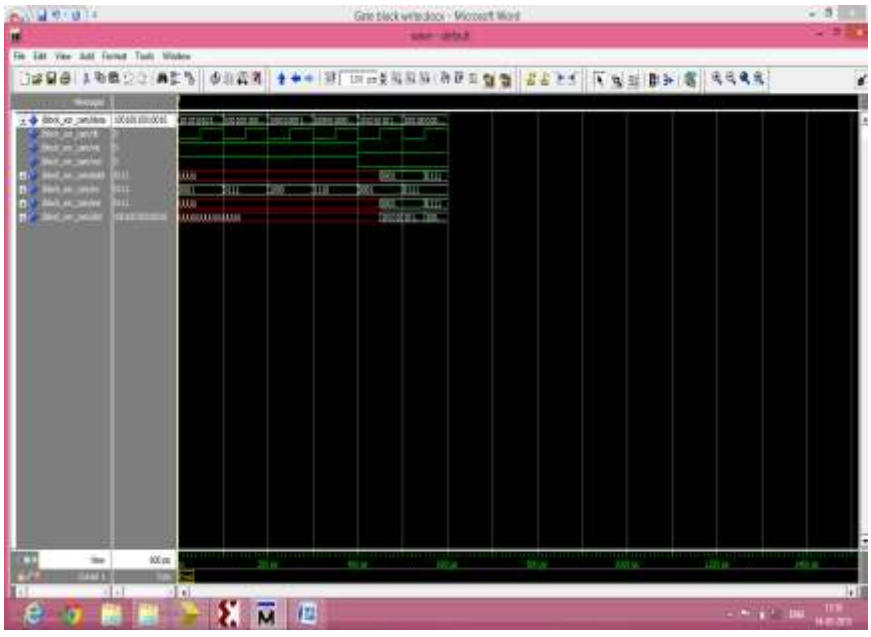

Fig.8 Data read operation for CAM using XOR technique

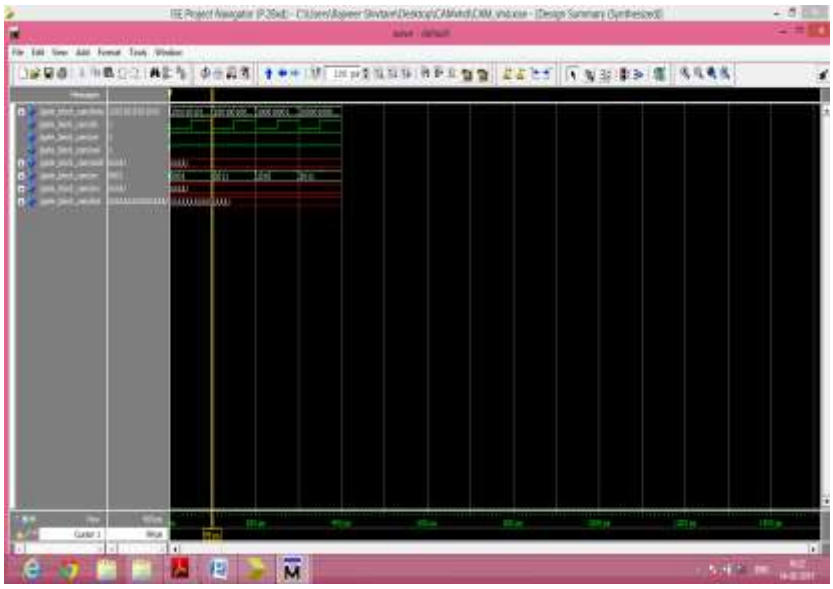

Fig.9 Data write operation for CAM using Gate block selection technique

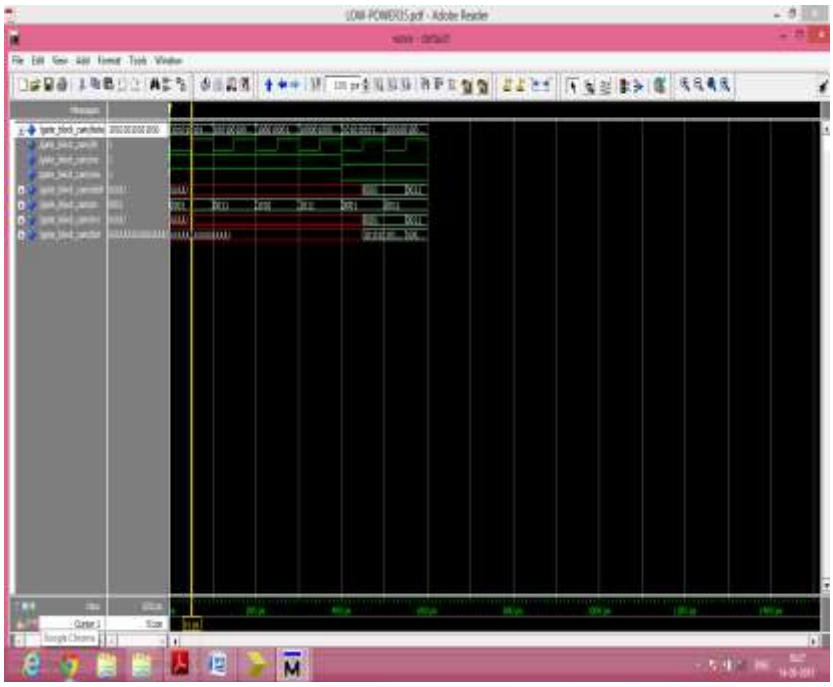

Fig.10 Data read operation for CAM using Gate block selection technique

VI CONCLUSION

In this paper low power pre-computation based content addressable memory (PB-CAM) is simulated in VHDL. Simulation results and mathematical analysis shows that the XOR and gate-block selection techniques can effectively save power as compared to one's count. It also reduces the number of comparison operations. From power report it shows that power consumption is also reduced. This PB-CAM performs data searching operation in exactly one clock cycle. So it is flexible and adaptive for the low power and high speed search applications

such as asynchronous transfer mode (ATM), communication networks, databases, lookup tables, data compression.

The power can be further reduced by using XNOR approach instead of XOR approach for parameter extractor design of CAM.

VII ACKNOWLEDGEMENT

I would like to thank Mrs. V.G. Raut, Associate professor, SCOE, Pune University for her support and valuable guidelines to prepare this paper. I also thank Mr. M.B. Mali, HOD, SCOE, Pune University for his help and support to complete this work.

REFERANCES

[1] S. Jeeva, S. Bharati, *Power Efficient Architecture of Pre-computation Based Content Addressable Memory,* ICCCI , Jan. 2012.

[2] 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, Mar. 2006, pp. 712–727.

[3] 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, Jun. 2001, pp. 956–968.

[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,Nov. 2003, pp. 1958–1966.

[5] K. Pagiamtzis and A. Sheikholeslami, *A low-power content-addressable Memory (CAM) using pipelined hierarchical search scheme* , IEEE J. Solid-State Circuits, vol. 39, no. 9 Sep. 2004, pp. 1512–1519.

[6] 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, , Jan. 2003, pp. 155– 158.

[7] K. Pagiamtzis and A. Sheikholeslami, Using *cache to reduce power in Content-addressable memories (CAMs),* in Proc. IEEE Custom integer. Circuits Conf., Sep. 2005, pp. 369–372.

[8] C. S. Lin, J. C. Chang, and B. D. Liu, *A low-power pre computation based Fully parallel content-addressable memory,* IEEE J. Solid-State Circuits, vol. 38, no. 4, Apr. 2003, pp. 622–654.

[9] 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.

[10] K. Vivekanandarajah, T. Srikanthan, and S. Bhattacharyya, *Dynamic filter cache for low power instruction memory hierarchy,* in Proc. EuromicroSymp. Digit. Syst. Des., Sep. 2004, pp. 607–610.

[11] R. Min, W. B. Jone, and Y. Hu, *Location cache: A low-power L2cache system,* in Proc. Int. Symp. Low Power Electron.Des., Apr.2004, pp. 120–125.

[12] K. H. Cheng, C. H.Wei, and S. Y. Jiang, *Static divided word matching line for low-power content addressable memory design,* in Proc. IEEEInt. Symp. Circuits Syst., May 2004, vol. 2, pp. 23–26.

[13] S. Hanzawa, T. Sakata, K. Kajigaya, R. Takemura, and T. Kawahara, *A large-scale and low-power CAM architecture featuring a one-hotspot block code for IP-address lookup in a network router*, IEEE J. Solid-State Circuits, vol. 40, no. 4, pp. 853–861, Apr. 2005.

[14] Y. Oike, M. Ikeda, and K. Asada, *A high-speed and low-voltage associative co-processor with exact Hamming/Manhattan- distance estimation using word-parallel and hierarchical search architecture*, IEEE J. Solid-State Circuits, vol. 39, no. 8, pp. 1383– 1387, Aug. 2004.

[15] K. Pagiamtzis and A. Sheikholeslami, *A low-power content-addressable memory (CAM) using pipelined hierarchical search scheme*, IEEE J. Solid-State Circuits, vol. 39, no. 9, pp. 1512–1519, Sep. 2004.