International Journal of Engineering Studies.
ISSN 0975-6469 Volume 8, Number 2 (2016), pp. 167-185

http://www.ripublication.com

Low Power Correlative Register Sequencing for Content Addressable Memory

K.Suresh Kumar[1], Y.Rajasree Rao[2] and K.Manjunathachari[3]

[1]Asst.Prof., ECE Dept., SSJEC, Hyderabad, [2]Prof &Dean, St.Peter's Engg College, Hyderabad,.[3]Prof.&HOD, ECE Dept., GITAM University, Hyderabad, India.

Abstract

Low power has emerged as a principal theme in today's electronics industry. The need for low power has caused a major paradigm shift where power dissipation has become an important a consideration as performance and area. Memory applications are designed for high-speed operations with minimum area coverage for compact and reliable operation in current electronic industries. Though measures are taken in designing the Memory applications for low power consumptions there requires a enhancement for the optimization techniques in low power Memory applications. This paper focus on the development of low power Memory applications methodology on system level modeling and circuit level modeling for power optimization. This work develops a power optimization approach in bus transitions using Hamming coding scheme called 'correlative register sequencing' for transition power reduction in content addressable memory application.

Keywords: Content addressable memory, low power correlative register sequencing, and memory applications.

I. INTRODUCTION

Content-addressable memory (CAM) and associativememory (AM) are types of storage structures that allowsearching by content as opposed to searching by address. Suchmemory structures are used in diverse applications rangingfrom branch prediction in a processor to complex pattern recognition. In the past few years, many algorithms and hardware designs are proposed to accelerate pattern matching. The hardware approaches can be classified into two main categories, logic and memory architectures. The logic architectures mostly use on-chip logic resources of field-

programmable gate array (FPGA) to convert regular expression pattern into parallel state machines or combinatorial circuits because FPGA allows for updating new attack patterns. In [1] proposed algorithm to compile regular expression patterns into combinatorial circuits based on nondeterministic finite automaton (NFA). [2] Developed a module generator that shared common prefixes to reduce the circuit area on FPGA. [3] Presented a content-scanning module on FPGA for an internet firewall. [4] Improved area and throughput by adding pre-decoded wide parallel inputs to traditional NFA implementations. [5] Presented a pre-decoded multiple-pipeline shift-and-compare matcher which reduced routing complexity and comparator size by converting incoming characters into many bit lines. [6] proposed a sharing architecture which significantly reduces circuit areas by sharing common infix and suffix sub-patterns. From the perspectives of reconfigurability and scalability, memory architectures are attractive because memory is flexible and scalable. The Aho–Corasick (AC) algorithm [7] is the most popular algorithm which allows for matching multiple string patterns. [8] proposed a configurable string matching accelerator based on a memory implementation of the AC FSM. [9] Proposed the bit-split algorithm partitioning a large AC state machine into small state machines to significantly reduce the memory requirements. [10] Presented an FPGA implementation of the bit-split string matching architecture. [11] Proposed to reduce the memory size by relabeling states of AC state machine. Additionally, [12] proposed to use Label Transition Table and CAM-based Lookup Table to significantly reduce the memory size. [13], [14] proposed a hash-based pattern matching co-processor where memory is used to store the list of substrings and the state transitions. [15] Proposed a pattern matching algorithm which modifies the AC algorithm to consider multiple characters at a time. Furthermore, the content addressable memories (CAM) is also widely used for string matching because it can match the entire pattern at once when the pattern is shifted past the CAM. [16] Used CAM to perform parallel search at a high speed. [17] applied the pre-decoded technique for the CAM-based pattern matching to reduce the area. Additionally, [18] presented a ternary content addressable memory (TCAM)-based multiple-pattern matching which can handle complex patterns, correlated patterns, and patterns with negation. The hash-based approach was proposed to utilize Bloom filter for deep packet inspection. [19] proposed a hashing-table lookup mechanism utilizing parallel bloom filters to enable large number of fixed-length strings to be scanned in hardware. [20] proposed an intelligent gateway based on Bloom filter that provides Internet worm and virus protection in both local and wide area networks.

II. CONTENT ADDRESSABLE MEMORY

Previous generations of CPUs were implemented as discrete components and numerous small integrated circuits (ICs) on one or more circuit boards. Microprocessors, on the other hand, are CPUs manufactured on a very small number of ICs; usually just one. The overall smaller CPU size as a result of being implemented on a single die means faster switching time because of physical factors like decreased gate parasitic capacitance. This has allowed synchronous

microprocessors to have clock rates ranging from tens of megahertz to several giga hertz. Additionally, as the ability to construct exceedingly small transistors on an IC has increased, the complexity and number of transistors in a single CPU has increased dramatically. This widely observed trend is described by Moore's law, which has proven to be a fairly accurate predictor of the growth of CPU (and other IC) complexity up to date.While the complexity, size, construction, and general form of CPUs have changed drastically over the past sixty years, it is notable that the basic design and function has not changed much at all. Almost all common CPUs today can be very accurately described as Von Neumann stored-program machines.

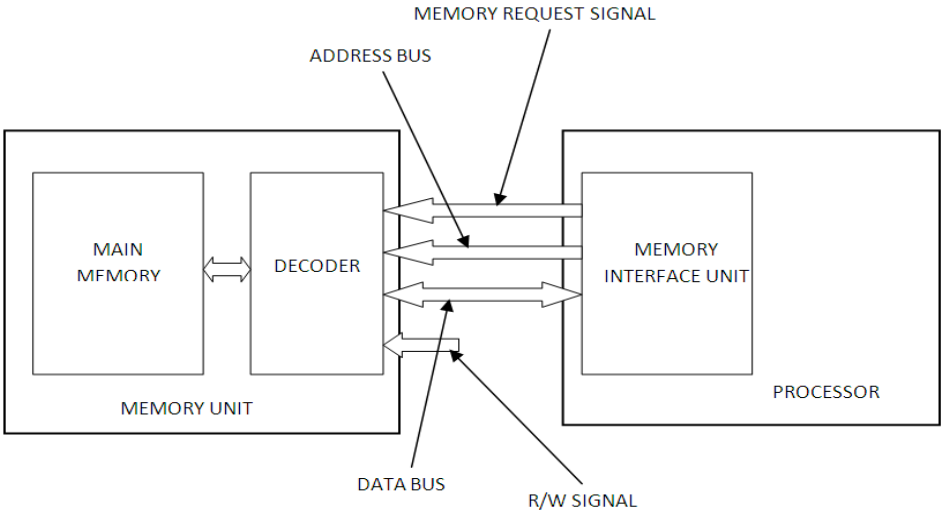


Figure 1. Example processor unit

Processor and memory have to funnel data on and off the bus that lies between them. The unit on the processor that sets up a memory address and issues memory requests has a fixed operating rate. Furthermore, the memory units and the bus have fixed operating rates. These rates are not reduced when more memory is added nor when more functional units are added to the processor. When attempting to extend a computer's architecture it is the processor-memory connection that forms the rate determining step which limits the computer's speed. This part of a von Neumann computer has come to be known as the 'von Neumann bottleneck'. Parallel computing is largely concerned with getting around this bottleneck by various ingenious means.

III. CORRELATIVE REGISTER LOGIC

Buses have been used as an efficient communication link among functional modules in very large scale integration (VLSI) systems. Whereas the size of functional modules decreases with the development of semiconductor technology, the size of VLSI chips increases, and so does the number of functional modules on a chip. Increasing communication requirements among the modules demand more complicated and more efficient buses. Currently, internal bus design plays important roles in the performance of a chip. Since too many modules rely on the buses for their

communication, the buses are usually heavily loaded so that they dissipate quite amount of power in operation. Activation of external buses consumes significant power as well, because many input–output (I/O) pins and large I/O drivers are attached to the buses. Typically, 50% of the total power is consumed at the I/Os for well-designed low-power chips by R.Wilson. Thus, reducing the power dissipated by buses becomes one of the most important concerns in low-power VLSI design. The dynamic power dissipated in a bus is expressed as the following N.Weste and K.Eshraghian:

$$P_{BUS} = \sum_{line} C_{load} V_{DD}^2 N_{trans} \quad (1)$$

Where C_{load} is the total load capacitance attached to a bus line, V_{DD} is the voltage swing at operation, and N_{trans} is the number of transitions per second.

There are two approaches to reduce the dynamic power of buses. One is to save the dynamic power per activation by reducing either C_{load} or V_{DD}. Reduced swing bus is an example of this approach Y.nakagone through H.zhang. The other is to reduce the number of bus activations by coding. Bus-invert (BI) coding by M.R.Stan and W.P.Burleson, Gray code by C.L.Su,C.Y.tsui and A.M.Despain, and the beach solution by L.Benini,g.Demicheli are some examples of this approach. Before the advent of internet and multimedia systems, most of I/O data used in VLSI chips are granulated data which are requested a periodically on demand and consist of few bytes of discrete information. The previous coding schemes were devised for the applications with this kind of I/O patterns. However, a new pattern of data transmission has become a great concern with the widespread use of Internet and multimedia systems. Data are transferred like a stream when used in applications such as MP3 players and video players. Once an operation is started, it requires transmitting large amounts of data from a few kilobytes to hundreds of megabytes. For example, web surfing or downloading files from the Internet involves transmission of a few kilobytes of data, while playing music or movies needs streaming data transmission up to a few gigabytes. As streaming becomes one of the major data transfer patterns, we have one more degree of freedom, i.e., the sequence of data that we can exploit to reduce the number of bus transitions during data transmission. A new coding scheme called sequence-switch coding (SSC) is proposed in this paper. It is different from previous transition-reduction coding schemes in that it is aimed at applications with the stream-type data transfer pattern. SSC reduces the number of bus transitions by rearranging the transmission sequence of data. An algorithm called sequencing algorithm is presented to show the feasibility of SSC. This algorithm reduces around 10% of bus transitions in transmission of the benchmark files. For the brevity of description, let us define some terms and notations first. A switched sequence is defined as the sequence of words transmitted according to an SSC algorithm. Let us express the hamming distance between a word, W and a bus, B, as H(W;B).Since I/O coding was proposed in to reduce transient noises, there have been many efforts to reduce the dynamic power of buses by coding which can minimize the number of bus transitions in transmission. BI coding is a general-purpose coding that is suitable for the transmission of uncorrelated data. Some

variations of BI such as partial BI (PBI) coding and weight-based BI coding have also been developed by exploiting some prior knowledge on data. For instruction address bus and data address bus, addresses are highly correlated, localized, and even sequential. Gray code, T0 code, inc-xor, and Working zone encoding can be more efficient than BI for such buses. The beach solution is a coding scheme designed for special-purpose applications.

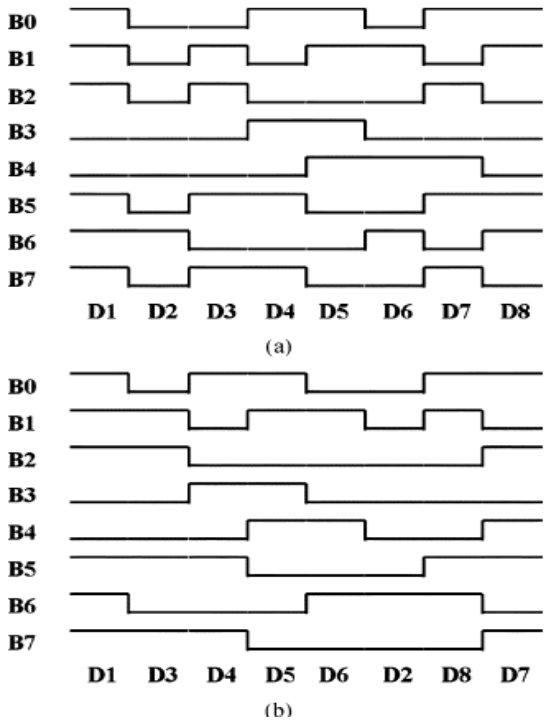


Figure 2. Illustration of the effect of transmission sequence on bus transitions. The waveform of eight-bit bus when the eight data are transmitted by the (a) original order and (b) a different order.

So far, most of the previous transition-reduction coding schemes have been developed for the granulated data, therefore, they have not considered the sequence of data as an important factor. SSC is the first general-purpose coding scheme that employs the sequence of data in reducing the number of bus transitions. SSC needs no prior information on data to be sent, and it can be applied to any application that transmits more than two data sequentially for most operations. Like multimedia systems, many applications require a .le or streaming data rather than a few bytes of granulated data as the input/output for their basic operations. The I/O data are transferred consecutively at a relatively constant rate. This kind of transfer pattern gives us a new opportunity to reduce the number of bus and I/O) transitions during data transmission. When a sequence of data moves through a bus, its transmission sequence can be chosen to minimize the number of bus transitions. For example, the effect of the sequence on the number of bus transitions is illustrated in Figure 1. Let us assume that there are eight data to be sent via a bus, and Figure.2 is the waveform of the bus when

these are transmitted without any modification. If we send the data with a different order, e.g., D1-D3-D4-D5-D6-D2-D8-D7, the waveform is changed. The number of transitions with the new order is 23 compared to 32 with the original order. About 28% of bus changing only the transmission order reduces transitions. SSC, which is based on this observation, is a method intended to reduce the power consumption of buses by changing only the data sequence. SSC does not try to find the optimum global sequence for the entire data for two reasons. First, it is an NP-complete problem: Finding

```
Unlimited_sequencing (stream in, bus B)
{
Register L;              // sequencing  register
Word N;                  // New incoming word
Bool S=0;                  // S-line
L=GetNewWord(in);    //the first word (sequencing )
Do {
N=GetNewWord(in)
If(N==Null) break;
If(H(L,B)+S)>H(N,B)+(1-S)   //compare hamming
distances
{ B=N; S=1;}     //switch sequence
else
{B=L; L=N; S=0;}   //send sequencing
} while(1)
B=L; L=N; S=0; //send the last word
}
```

Figure 3. Pseudo code for the encoder of unbounded Sequencing algorithm

The optimum sequence among N data is equivalent to the well-known N-city traveling-salesman problem (TSP) that is known as the NP-complete problem. Second, even if the optimum global sequence could be found easily, it might be impractical for real applications due to the decoding latency and implementation overheads. For example, the worst-case scenario occurs when the first data in the original sequence is scheduled to the last in the optimum one. In this case, the receiver cannot use any received data until all the data have arrived which causes unacceptable decoding delay, and the decoder should reserve a huge buffer to hold the data. For the above reasons, SSC must deal with sub optimum sequences that make much fewer transitions than the original sequence. SSC is a method, not a specific algorithm, and therefore, can be realized in many ways. If we use some heuristics, it is not difficult to

find a new algorithm. SSC with a more complex algorithm may achieve a better performance, but it is accompanied with more implementation overheads. Selection of a proper algorithm is a designer's tradeoff between performance and implementation complexity.

Unbounded Sequencing Algorithm

In this algorithm, only two most recent words are considered in switch operation. Fig. 3is the pseudo code for the encoder of the unbounded sequencing algorithm (UL). The encoder has a register called sequencing register to store a word temporarily. Any word that stayed in this register is tagged as a sequencing . At every cycle, two words, a sequencing and a new incoming word compete for transmission so that the winner is sent and the looser is held as the sequencing for next competition. The coding information is transmitted simultaneously with the word by an auxiliary line called S-line that is added in parallel to the bus. The S-line is set to 1 when a latter word is selected while it is reset to 0 whenever a sequencing is transmitted. If a switched sequence is compared with the original sequence, all words displaced from the original positions are sequencing s. Furthermore, the original sequence can be recovered from a switched sequence by relocating every sequencing to the position of its preceding sequencing ;

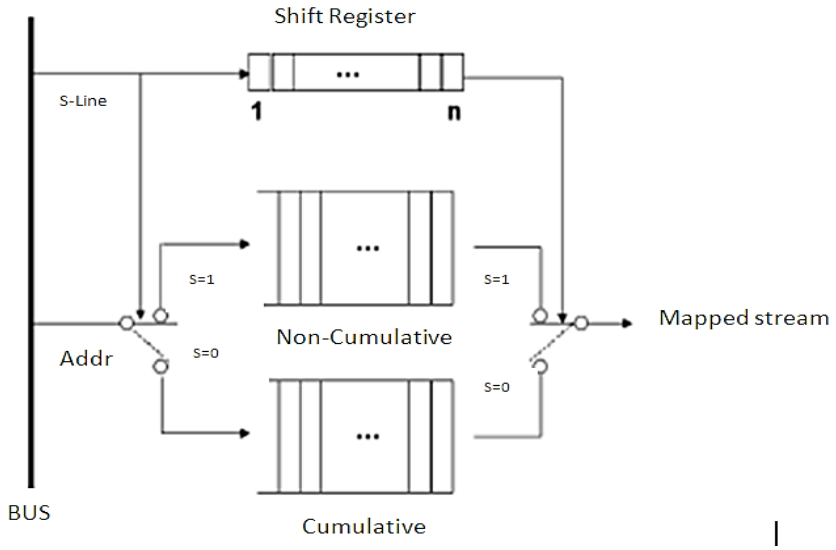


Figure 4. An Implementation of the decoder for Sequencing algorithm

the first sequencing is placed at the beginning of the sequence. Using this property, decoding can be achieved with two first in–first out (FIFO) buffers and a shift register (SR) as in Fig.4. The one buffer is reserved for sequencing s and the other is for non-sequencing s. The SR consists of n bits that are initially reset to 0. Assume that the decoder sends a decoded word at the first-half cycle while it receives a word from the bus at the second-half cycle. For every cycle, the decoder receives a word with an S-value from the bus, and stores the word ineither of the buffers according to the S-value. SR is shifted from SR1 to SRn, and the S-value goes to SR1. For the first n _ 1

cycle, i.e., until the first S-value reaches to SRn_1, the decoder only stores the received words. From nth cycle, the decoder starts to send a word to the next stage (client). According to the value of SRn, the decoder fetches a word from either of the buffers and sends it to the client. The latency of the decoder becomes n-1 cycles. Let us define lagging distance of a sequencing as the number of its losses in the competitions, and the maximum lagging distance (l_{max}) of a switched sequence as the biggest lagging distance of all sequencing s in the sequence. Then, n should be greater than or equal to l_{max} + 2, and the FIFOs should have at least l_{max} + 1 slots. The shortcoming of UL is that l_{max} is not predictable in advance. Because it can be a very large number, the decoder should prepare a huge number of slots.

As the gene patterns are represented in binary sequences, which are buffered as a large data set. These patterns are varied by few bit transitions, and the correlative nature is very high among each sequence. Due to the self-correlative nature among multiple sequences the probability of correlative error and miss-classification is high. The correlative property in binary pattern is observed by two well-known approaches, Hamming distance (HD) or Levenshtein Edit Distance (LED)[18]. Wherein the hamming distance is a measure of co-relative pattern match count defined by,

$$d_{HD}(s, s_1) = \sum_{s_{1i} \neq s_{2i}} 1, \quad i = 1, \ldots \ldots m \tag{2}$$

Where, d_{HD} measures the number of varying positions intwo strings s_1 and s_2 of equal length m.

for example,

s_1= ”ATCCATGC” and s_2 = ”ATGGATAC”

s_1 : ”ATCCATGC”

s_2 : ”ATGGATAC” =>d_{hm} = 2.

However such sequences measurement is carried out for equal length sequencing. In gene sequence however, the pattern insertion, deletion, or substitution is observed. In such a case Hamming distance computation is not effective. In such case Levenshtein Edit Distance (LED) is computed. The LED is the minimum number of edit operations to transforma string s_1 of length n into a string $s_2$of length m. For the processed signal with insertion, deletion or updation the d_{LED} is computed as,

$$w(a_i, b_j) = \begin{cases} 0, & if\ a_i = b_j \\ 1, & if\ a_i \neq b_j, substitution \end{cases} \tag{3}$$

$$d_{LED}(i,j) = min \begin{cases} d_{LED}(i-1, j-1) + w(s_{1i}, s_{2i}) \\ d_{LED}(i, j-1) + 1 \quad insertion \\ d_{LED}(i-1,) + 1 \quad deletion \end{cases} \tag{4}$$

For example, for two given pattern sequences, S_1 and S_2 ,

S_1 = ATCTGA

S_2 = ATTA

Then d_{LED} = 2 , as C in S_1 is inserted at 3 and G is deleted at 5.

In the process of pattern matching the hamming distance provides the similarity index and LED provide the cross correlative counts. However, applying these approaches for measuring distances, the bit pattern plays and important role. It is observed that more the pattern transition exist the probability of pattern matching error get higher.The pattern matching errors could be minimized via updated sequencing of binary patterns. To develop such approach in recent past [19] a 1-D Local binary pattern (LBP) representation of signal is presented. A LBP is locally transformed binary pattern used for the optimization of pseudo random nature of a binary pattern. They are more dominantly used as a filtration approach in image processing, signal processing, speech processing etc.[19].A LBP sequence is generated as uniform patterns if they have at mosttwo circularly bitwise transitions from 0 to 1 or vice versa, andnon-uniform patterns if otherwise. In uniform LBP mapping,one separate spectral analysis using histogram bins is used for each uniform patternand all non-uniform patterns are accumulated in a single bin. By grouping the non-uniformpatterns into one label, the noise in non-uniformpatterns is suppressed. The number of patterns is reducedsignificantly at the same time.
For example, “11110000”is a uniform pattern as $U = 2$, whereas “01010111” is a non-uniform pattern as $U = 6$. With the realignment of such pattern the error robustness is achieved, the LBP is hence robust to alignment error [20]. With this approach the LBP coding is applied over sequence of gene pattern, and the patterns are categorized into uniform and non-uniform patterns to reduce alignment and matching error. The spectral feature of the pattern distribution using Histogram bin is computed and the patterns are realigned to achieve lower transition matching error. For this local binary patterns the sequence are stored as a data set, and mining is then carried out over such binary pattern. However this patterns are very large data set, though the pattern are aligned to minimize the transition error, they need to processed with advanced technologies to mine the information faster.

IV. MODELING

For the optimization of power consumption during data transition a transition optimization for encoding and decoding architecture is developed. Figure 5.1 illustrates the operational block diagram for the bus transition minimization using unbounded Sequencing Algorithm.

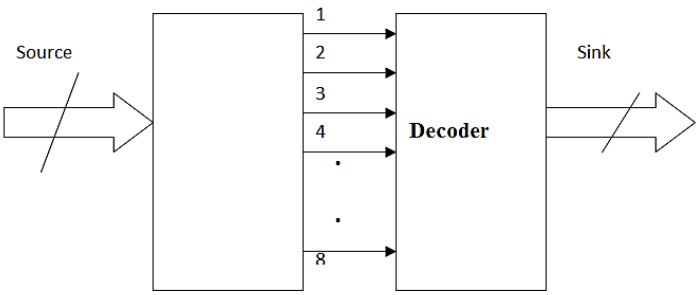


Figure 5. Functional Block Diagram for Transition optimization using Unbounded Sequencing Algorithm

The source data in binary format is passed to the encoding unit in parallel format and are encoded using Hamming code distance for transition minimizations. The transition minimization is carried with a parity code and is passed to the decoder for power consumption minimization. The internal functional units for the operation of the suggested architecture is as explained below.

The encoder unit consists of a comparator units and a counter unit for the generation of data stream during coding phase.

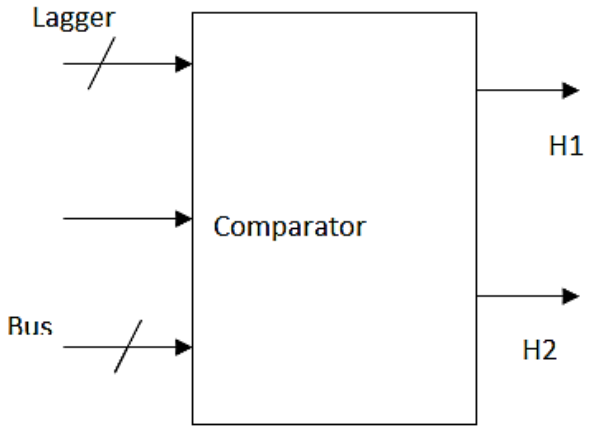


Figure 6. Comparator units for the operation of bus transition selection using hamming code

This unit takes the three input data Din, Bus, and Sequencing bus for the calculation of Hamming distance for Sequencing and input data with data Bus.

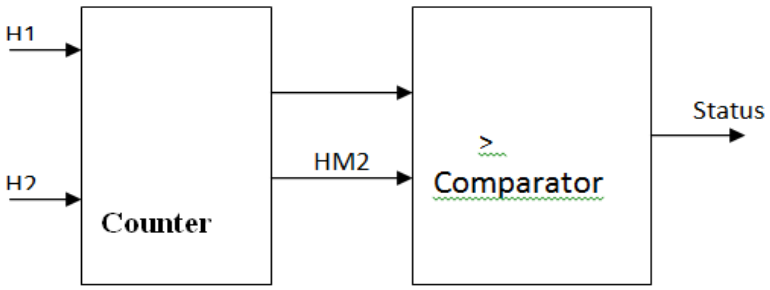


Figure 7. Hamming code comparator unit

This unit takes the hamming count and generates a count value based on the hamming distance passed the comparison is made and if the hamming distance1 is observed to

be greater than the second value then a status signal s= '0' is generated or else a '1' is generated.

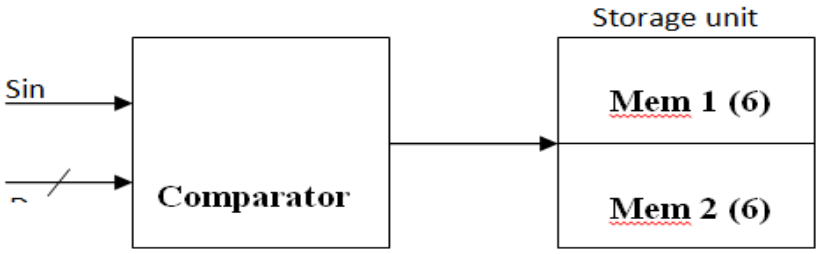


Figure 8. Decoder units with storage operation

Figure.8 shows the decoding operation of the designed system and the probable storage operation for the decoding operation storing the segregated information for s=1 in memory location (Mem1) and s=0 for memory location (mem2). Based on the memory location filled the data is segregated as block1 or block2 information.

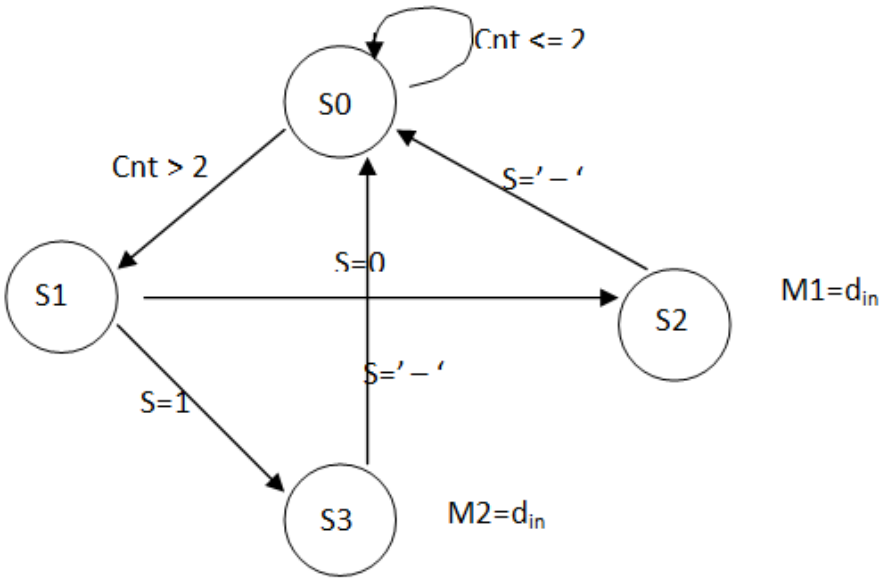


Figure 9. Decoding State Diagram for bit estimation

Figure.9 illustrates the state diagram for the developed decoder unit. The state operation used for the estimation of the bit stream is illustrated in figure 9 with 4 state transitions.

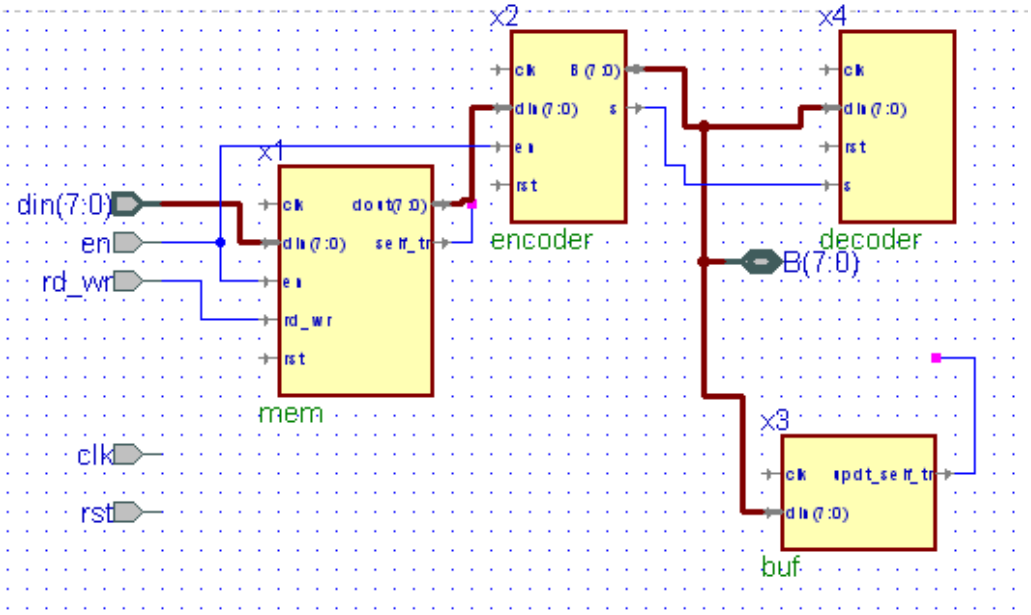


Figure 10: Implemented system diagram on Aldec's Block diagram

This figure illustrates the implemented design for the developed transition optimization algorithm for power reduction. The generated block diagram for the system is obtained from the Aldec simulator.

V. EXPERIMENTAL RESULTS

For the functional evaluation of the designed system the developed system is designed using VHDL definition and simulated on the Active-HDL simulator for it's operational verification. For the testing of the designed system a test bench file is generated where two frame data is passed. This frame data is read by the video_codec file and divide into 8 X 8 Block. The block data is processed for noise estimation, motion estimation, and compression. Under recovery the data is decoded back for it's regeneration. The obtained simulation results are as illustrated below. For the real time Realization and resource utilization the developed design is synthesized using Xilinx synthesizer. The obtained synthesis results and their logical, mechanical, and timing reports are also presented in the following section.

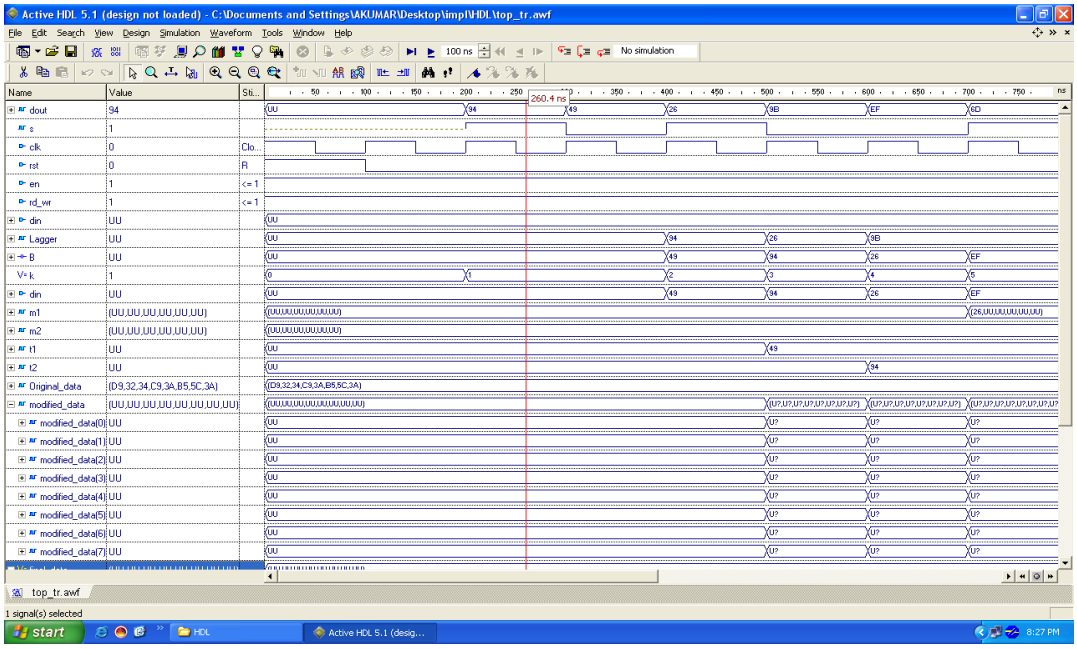


Figure 11. Simulation plot showing the observations made for the developed Sequencing algorithm

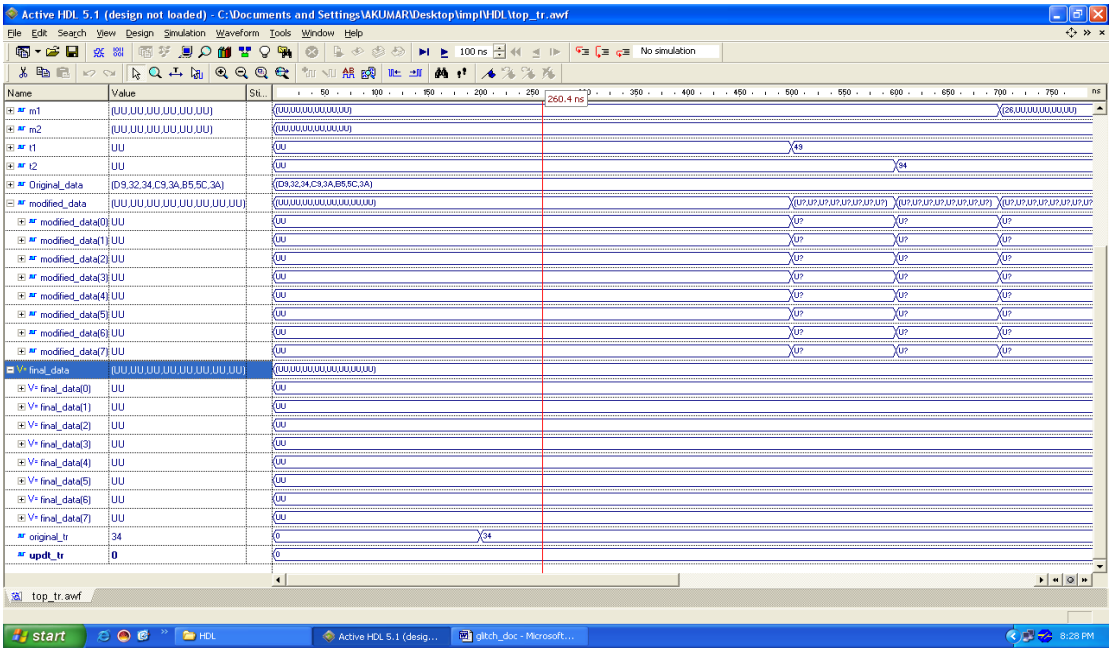


Figure 12. Figure illustrating the original data the modified data regenerated after coding

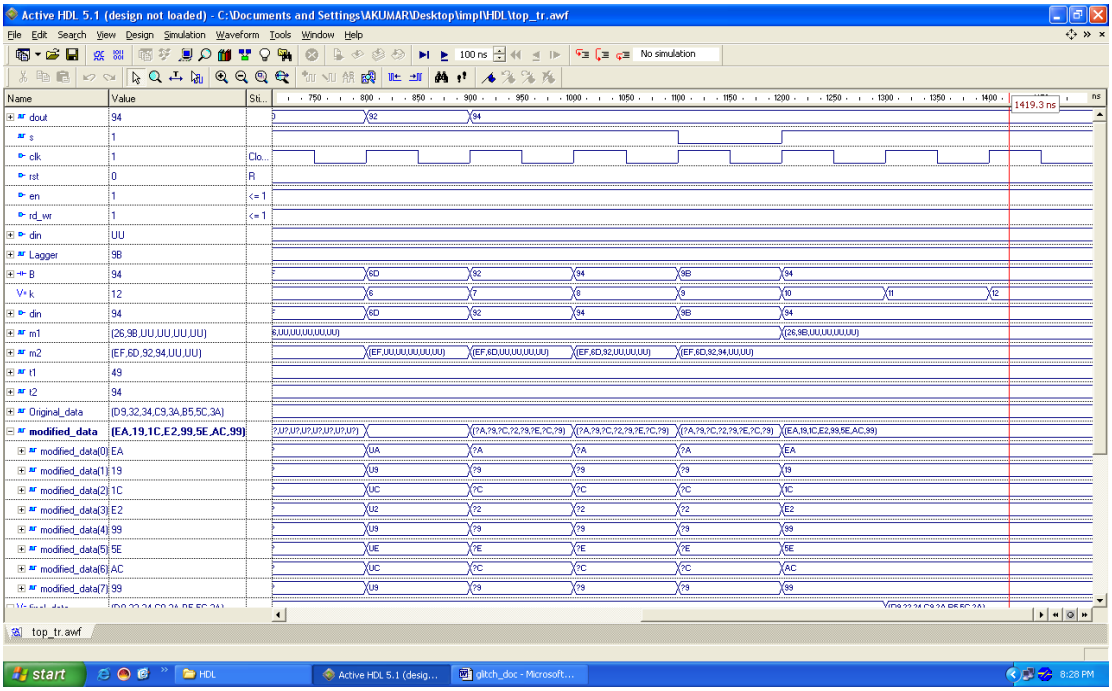


Figure 13. Figure illustrating the bus , Sequencing and input data line for the designed encoder and decoder unit

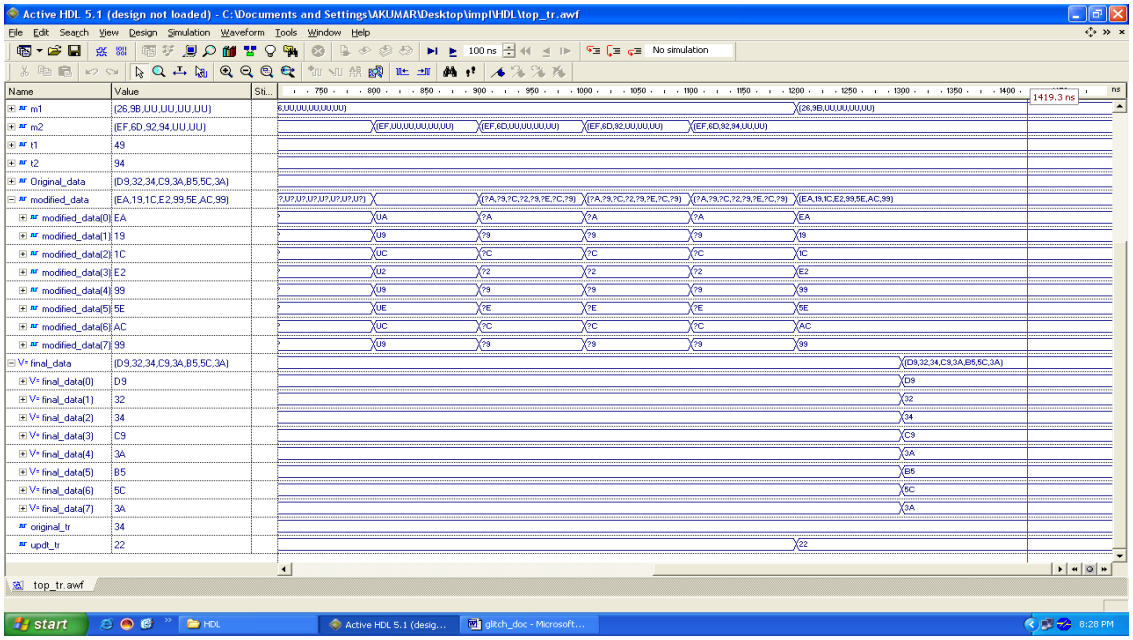
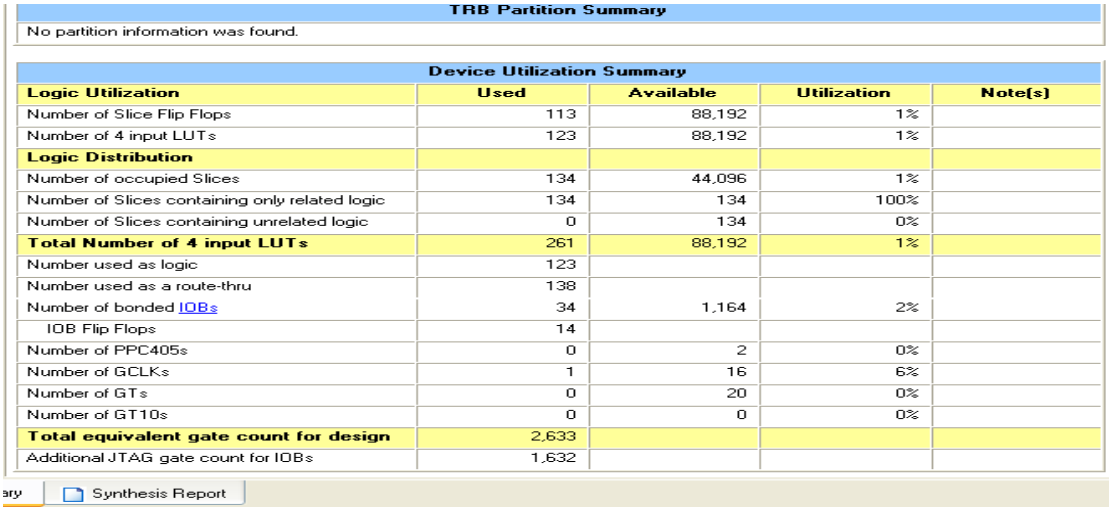


Figure 14. Figure illustrating the content of the memory buffered data for regeneration and the reconstructed data from it.

SYNTHESIS RESULTS

```
=========================================================
Final Results
Design Statistics
# IOs                    : 34
Cell Usage :
# BELS                   : 542
#      GND                : 1
#      INV                : 5
#      LUT1               : 124
#      LUT2               : 66
#      LUT3               : 15
#      LUT4               : 38
#      MUXCY              : 147
#      MUXF5              : 18
#      MUXF6              : 2
#      MUXF7              : 1
#      VCC                : 1
#      XORCY              : 124
# FlipFlops/Latches       : 127
#      FD                 : 15
#      FDE                : 80
#      FDR                : 32
# Clock Buffers           : 1
#      BUFGP              : 1
# IO Buffers              : 33
#      IBUF               : 17
#      OBUF               : 16
=========================================================
==TIMING REPORT
Clock Information:
Speed Grade: -6
   Minimum period: 7.693ns (Maximum Frequency: 129.984MHz)
   Minimum input arrival time before clock: 2.950ns
   Maximum output required time after clock: 3.615ns
   Maximum combinational path delay: No path found
```



TRB Partition Summary

No partition information was found.

Device Utilization Summary

Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops	113	88,192	1%	
Number of 4 input LUTs	123	88,192	1%	
Logic Distribution				
Number of occupied Slices	134	44,096	1%	
Number of Slices containing only related logic	134	134	100%	
Number of Slices containing unrelated logic	0	134	0%	
Total Number of 4 input LUTs	261	88,192	1%	
Number used as logic	123			
Number used as a route-thru	138			
Number of bonded IOBs	34	1,164	2%	
IOB Flip Flops	14			
Number of PPC405s	0	2	0%	
Number of GCLKs	1	16	6%	
Number of GTs	0	20	0%	
Number of GT10s	0	0	0%	
Total equivalent gate count for design	2,633			
Additional JTAG gate count for IOBs	1,632			

ary Synthesis Report

Figure 15. Summarized synthesis report for the developed estimation system

```
------------------------------------------------------------
Release 9.1i - XPower SoftwareVersion:J.30
Copyright (c) 1995-2007 Xilinx, Inc.  All rights reserved.
Design:      C:\Xilinx91i\BMA\video_codec.ncd
Preferences:  video_codec.pcf
Part:        2vpx70ff1704-6
Data version: PREVIEW,v1.0,05-28-03

Power summary:                  I(mA)    P(mW)
------------------------------------------------------------
Total estimated power consumption:            181
                        ---
Vccint 1.50V:       85      128
Vccaux 2.50V:       20       50
                Vcco25 2.50V:        2        4
                        ---
                   Clocks:        0        0
                   Inputs:        0        0
                    Logic:        0        0
                  Outputs:
                    Vcco25       0        0
                  Signals:        0        0
                        ---
          Quiescent Vccint  1.50V:       85      128
          Quiescent Vccaux  2.50V:       20       50
          Quiescent Vcco25  2.50V:        2        4
```

Thermal summary:

Estimated junction temperature: 25C
Ambient temp: 25C
Case temp: 25C
Theta J-A: 0C/W

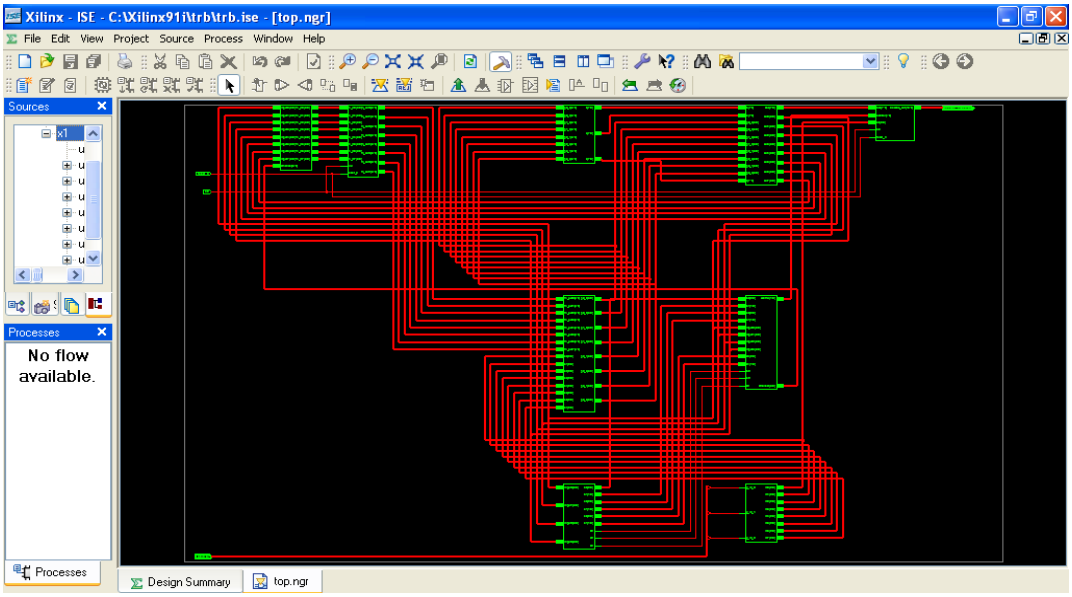


Figure 16. RTL view of the implemented system using Xilinx synthesizer.

VI. CONCLUSION

We have presented a memory-efficient pattern matching algorithm which can significantly reduce the number of states and transitions by merging pseudo-equivalent states while maintaining correctness of string matching. In addition, the new algorithm is complementary to other memory reduction approaches and provides further reductions in memory needs. The experiments demonstrate a significant reduction in memory footprint for data sets commonly used to evaluate IDS systems

REFERENCES

[1] D. Das, A. Roy, and H. Rahaman,‘‘Design of content addressable memoryarchitecture using carbon nanotubefield effect transistors,’’ in Progressin VLSI Design and Test, vol. 7373,

[2] H. Rahaman, S. Chattopadhyay, andS. Chattopadhyay, Eds. Berlin, Germany:Springer-Verlag, 2012, pp. 233–242,ser. Lecture Notes in Computer Science.[Online]. Available: http://dx.doi.org/10.1007/978-3-642-31494-0_27

[3] S. Kang et al., ‘‘A 0.1-_m 1.8-V 256-MBphase-change random access memory(PRAM) with 66-MHz synchronousburst-read operation,’’ IEEE J. Solid-StateCircuits, vol. 42, no. 1, pp. 210–218, Jan. 2007.

[4] K. Degawa et al., ‘‘A high-density ternarycontent-addressable memory usingsingle-electron transistors,’’ in Proc.IEEE 36th Int. Symp. Multiple-Valued Logic,2006, DOI: 10.1109/ISMVL.2006.6.

[5] M.-F. Chang et al., ‘‘17.5 A 3T1R nonvolatileTCAM using MLC ReRAM with sub-1nssearch time,’’ in Proc. IEEE Int. Solid-StateCircuits Conf., 2015, DOI: 10.1109/ISSCC.2015.7063054.

[6] K. Nii et al., ‘‘13.6 A 28 nm 400 MHz4-parallel 1.6 Gsearch/s 80MB ternaryCAM,’’ in IEEE Int. IEEE Solid-State CircuitsConf. Dig. Tech. Papers, 2014, pp. 240–241.

[7] S.Matsunaga et al., ‘‘A 3.14 um 2 4T-2MTJ-cellfully parallel TCAM based on nonvolatilelogic-in-memory architecture,’’ in Proc. IEEESymp. VLSI Circuits, 2012, pp. 44–45.

[8] J. Li, R. K. Montoye, M. Ishii, and L. Chang,‘‘1 MB 0.41_m22T-2R cell nonvolatileTCAM with two-bit encoding and clockedself-referenced sensing,’’ IEEE J. Solid-StateCircuits, vol. 49, no. 4, pp. 896–907, Apr. 2014.

[9] Y.-J. Hu, J.-F. Li, and Y.-J. Huang,‘‘3-D content addressable memoryarchitectures,’’ in Proc. IEEE Int.Workshop Memory Technol. DesignTesting, 2009, pp. 59–64.

[10] E. C. Oh and P. D. Franzon, ‘‘Designconsiderations and benefits ofthree-dimensional ternary contentaddressable memory,’’ in Proc. IEEECustom Integr. Circuits Conf., 2007,pp. 591–594.

[11] D. Bhattacharya, A. Bhoj, and N. Jha,‘‘Design of efficient content addressablememories in high-performance FinFETtechnology,’’ IEEE Trans. Very LargeScale Integr. (VLSI) Syst., vol. 23, no. 5,pp. 963–967, May 2015.

[12] A. McAuley and P. Francis, ‘‘Fast routingtable lookup using CAMs,’’ in Proc. IEEE12th Annu. Joint Conf. IEEE Comput.Commun. Soc., Netw.: Found. Future,1993, vol. 3, pp. 1382–1391.

[13] F. Yu, R. Katz, and T. Lakshman,‘‘Gigabit rate packet pattern-matchingusing TCAM,’’ in Proc. 12th IEEEInt. Conf. Netw. Protocols, Oct. 2004,pp. 174–183.

[14] K. Eshraghian, ‘‘Memristormos contentaddressable memory (MCAM): Hybridarchitecture for future high performancesearch engines,’’ IEEE Trans. Very LargeScale Integr. (VLSI) Syst., vol. 19, no. 8,pp. 1407–1417, Aug. 2011.

[15] Q. Guo, X. Guo, Y. Bai, and E. Ipek,‘‘A resistive TCAM accelerator fordata-intensive computing,’’ in Proc. 44thAnnu. IEEE/ACM Int. Symp. Microarchitect.,2011, pp. 339–350.

[16] W. Webber, A. Moffat, and J. Zobel,‘‘A similarity measure for indefiniterankings,’’ ACM Trans. Inf. Syst.,vol. 28, no. 4, p. 20, 2010.

[17] M. F. Porter, ‘‘An algorithm for suffixstripping,’’ Program, Electron. LibraryInf. Syst., vol. 14, no. 3, pp. 130–137, 1980.

[18] S. Narasimhan, M. Tabib-Azar, H. J. Chiel,and S. Bhunia, ‘‘Neural data compressionwith wavelet transform: A vocabularybased approach,’’ in Proc. 3rd Int.IEEE/EMBS Conf. Neural Eng., 2007,pp. 666–669.

[19] [82] S. Narasimhan, M. Cullins, H. J. Chiel, andS. Bhunia, ‘‘Wavelet-based neural patternanalyzer for behaviorally significant burstpattern recognition,’’ in Proc. IEEE 30thAnnu. Int. Conf. Eng. Med. Biol. Soc., 2008,pp. 38–41.

[20] [83] K. Pagiamtzis and A. Sheikholeslami,‘‘Content-addressable memory (CAM) circuitsand architectures: A tutorial and survey,’’IEEE J. Solid-State Circuits, vol. 41, no. 3,pp. 712–727, Mar. 2006.

[21] M. Herlihy and J. E. B. Moss,‘‘Transactional memory: Architecturalsupport for lock-free data structures,’’ACM SIGARCH Comput. Architect.News, vol. 21, no. 2, pp. 289–300,1993.

[22] Z. Yan, H. Jiang, D. Feng, L. Tian, andY. Tan, ‘‘SUV: A novel single-updateversion-management scheme forhardware transactional memorysystems,’’ in Proc. IEEE 26th Int.Parallel Distrib. Process. Symp., May 2012,pp. 131–143.