Digital Design of Discrete Exponential Bidirectional Associative Memory *

Chua-Chin Wang & Chih-Lwan Fan
Department of Electrical Engineering
National Sun Yat-Sen University
Kaohsiung, Taiwan 80424

Abstract

The exponential bidirectional associative memory (eBAM) was proved to be a systematically stable high-capacity memory. Considering the implementation of such an eBAM, we adopt the digital logic methodology basing on several factors, such as scalability and speed. In order to realize the eBAM by digital circuitry only, some special design is required such that the exponential function can be implemented without the loss of operating speed. For example, the exponent 8-to-9 high- speed value generator. Besides, the traditional add/sub accumulator costs too much area when the dimension of patterns is large. A pipelined increment/decrement accumulator is proposed in the design, which can also speed up the addition or subtraction besides the saving of chip area.

1 Introduction

After Kosko [3] proposed the *bidirectional associative memory* (BAM), many researchers threw efforts on improving its intrinsic poor capacity and implementing the BAM with hardware circuits. Among those researchers, Wang *et al.* [6] proposed two alternatives, *multiple training* and *dummy augmentation*, to enhance BAM's ability to find the global minimum; Simpson proposed an intraconnected BAM and a high-order autocorrelator [4]; and Tai *et al.* [5] proposed a high-order BAM; Wang *et al.* [8] developed a *weighted learning* algorithm for BAM. However, we have pointed out that all of these improvements pay a high price of increasing the complexity of the network but only get little enhancement of the capacity [7]. Chiueh and Goodman [1] proposed an exponential Hopfield associative memory motivated by the MOS transistor's exponential drain current dependence on the gate voltage in the subthreshold region such that the VLSI implementation of an exponential function is feasible.

After we estimated the impressive capacity of an eBAM [7], it becomes very interesting to explore the possibility of the hardware realization of such a neural network. Even though the neural networks implemented with MOS operating in the subthreshold region have the advantages of low power and compatibility with VLSI circuits, [2], employing only digital

*This research was partially supported by National Science Council under grant NSC 84-0408-E-110-003.

logic to realize such a network still remains a possibility. It is well known that digital logic possesses advantages of noise margin, design scalability, and less complexity. The only bottle neck is the exponential function used in such a network. In this paper, we first show the architecture and design methodology of the eBAM is presented, including the realization of exponential function needed in such a network, fast increment/decrement accumulator, and so on. Then, thorough simulation of the logic design of the eBAM is shown.

2 Digital Design of Exponential BAM

Before proceeding the hardware implementation of the eBAM, we have to restate the framework of the eBAM neural network as a background knowledge [7].

2.1 Theory of eBAM

Suppose we are given M bipolar pattern pairs, which are $\{(X_1, Y_1), (X_2, Y_2), ..., (X_M, Y_M)\}$, where $X_i = (x_{i1}, x_{i2}, ..., x_{in})$, $Y_i = (y_{i1}, y_{i2}, ..., y_{ip})$, $X_i \neq X_j$, $i \neq j$, and $Y_i \neq Y_j$, $i \neq j$. We use the following evolution equations in the recall process of the eBAM :

$$\begin{aligned} y_k &= \begin{cases} 1, & \text{if } \sum_{i=1}^{M} y_{ik} b^{X_i \cdot X} \geq 0 \\ -1, & \text{if } \sum_{i=1}^{M} y_{ik} b^{X_i \cdot X} < 0 \end{cases} \\ x_k &= \begin{cases} 1, & \text{if } \sum_{i=1}^{M} x_{ik} b^{Y_i \cdot Y} \geq 0 \\ -1, & \text{if } \sum_{i=1}^{M} x_{ik} b^{Y_i \cdot Y} < 0 \end{cases} \end{aligned} \tag{1}$$

where b is a positive number, $b > 1$, "$\cdot$" represents the inner product operator, x_k and x_{ik} are the kth bits of X and the X_i, respectively, and y_k and y_{ik} are for Y and the Y_i, respectively. The reasons for using an exponential scheme are to enlarge the attraction radius of every stored pattern pair and to augment the desired pattern in the recall reverberation process. We adopt the SNR approach to compute the capacity of the exponential BAM [7], $SNR_{eBAM} = \frac{2^{n-1} b^4}{2(M-1)(1+b^{-4})^{n-1}}$ where n is assumed to the $\min(n, p)$ without any loss of generality.

2.2 Architecture of Digital eBAM

Referring to Fig. 1, which shows the entire architecture of the digital eBAM, we divide the design into 4 major parts, X & Y RAM planes with shift controllers, the 8-to-9 exponent value generator, and increment/decrement accumulators (IDA).

2.2.1 RAM planes with shift controllers

The RAM planes for the digital eBAM are slightly different from the traditional DRAM or SRAM. There is no address decoder in our design. In contrast, we use a series of shifters to label which RAM module will be written with data next. The block diagram of X-Y RAM plane is shown in Fig. 2. Note that in the initialization of the chip, all of the shifters' contents will be set to "0", except the first 8×1 RAM module. Every single 8×1 RAM module is used to store 1 8-bit X or Y vector. Next data is only allowed to be loaded into the module of which the shifter is containing "1".

The detailed schematic diagram of the RAM cell and the shift controller is shown in Fig. 3. If the controller contains "0", then N6, N11, N12 and N10 are OFF. The RAM cell is isolated from the bus. When the controller is "1" and $\overline{\textit{Data Reload}}/\textit{Recall}$ is low, it is a write operation. N11, N12, P1 are ON, while N6 is OFF and then N10 is OFF. The data on Write Bus will be latched into RAM cell through N7 and N8 by $\overline{\textit{Enable}}$. When the controller is "1" and $\overline{\textit{Data Reload}}/\textit{Recall}$ is high, it is a read operation. N11, N12, P1 are OFF, while N6 is ON. The gate of N10 is connected to the inverting port of RAM cell. Data Read Bus is precharged when $\overline{\textit{Enable}}$ is high. The data will be presented on the bus when $\overline{\textit{Enable}}$ is low.

2.2.2 Exponent value generator

Our design is dedicated for 8-bit data vectors, though the same methodology can be applied to any dimension. Hence, the result of the inner product of the exponent, e.g., $X \cdot X_i$, can only be one of the following integers, $-8, -6, -4, \ldots, 0, \ldots, 4, 6, 8$. That is, there are 9 possibilities for the exponent. This indicates a 8-to-9 exponent decoder is necessary, because the inner product of X and X_i is one 8-bit vector, while the result is one of the 9 possibility. The result of such a value generator is only one "H" (high or "1") and the rest 8 outputs are all "L". The algorithm is shown in Fig. 4. The architecture of the decoder is half of a 8×8 box. If the bit of the XOR of X and X_i is 0, then move horizontally one grid to the right; if the bit is 1, then move upwardly one grid. The procedure starts form MSB to LSB of the inner product. An example is also shown in Fig. 4. For instance, if the result of the inner product is (1 0 0 0 0 1 0 1), then the output for -2 should be high, and the rest are all 0. The circuit of such a generator is shown in Fig. 5.

2.2.3 Increment/Decrement accumulator

Since the eBAM uses a summation, Eqn.(1), adders or accumulators are needed. The coefficients of the exponents in Eqn.(1) are either 0 or 1 in the binary system, or -1 or +1 in the bipolar system. This suggests accumulators are more suitable because the adders will consume too much chip area when the number of stored patterns increases. Moreover, we have to generate the exponent function by using the 8-to-9 value generator as described in the previous section. The result of the generator is taken as the input of the accumulator. The reason why the accumulators are used is all of the stored patterns are sequentially XORed. Hence, the result of the summation of those XORed stored patterns has to be kept to be used for the next stored pattern.

Because the design is dedicated for 8-bit vectors, we then need 8 complete IDA units and 12 regenerated IDA units to prevent overflow for storing less than 8 pattern pairs. The difference between a complete IDA unit and a regenerated IDA unit is that the complete IDA units have additional "Bit" and "I/D" signals coming from the exponent value generator, but the regenerated IDA units don't. Besides, the regenerated IDA units are smaller in terms of the area. The block diagram of cascaded IDA is shown in Fig. 6. Referring to Fig. 6, "I/D" denotes the y_{ik} if X is the retrieval pattern and vice versa; "2^i" denotes the result from the 8-to-9 exponent value generator. If I/D bit is 1, it means increment; if 0, it means decrement. In Table 1, "B" is the borrow bit, "C" the carry bit, and D the data bit currently in that accumulator unit. The next problem is how to design such units employed in the IDA. The truth tables, ground paths and circuit diagrams of the complete IDA unit and the regenerated IDA unit are respectively shown in Fig. 7, 8, 9, which detailedly tabulates the Boolean functions of $B, C, D, Bit, I/D$ and $D_{next}, C_{next}, B_{next}$. D denotes the current data bit in the unit, while Bit is the result from the exponent value generator. However, it is very inefficient to simplify such a huge adder by either K-map or other simplification methods. Hence, we decompose the adder into three parts in order to reduce the design complexity. That is, the $D_{next}, C_{next}, B_{next}$ are respectively designed. Fig. 10 shows the circuit for a single bit IDA unit. This unit will be reset when "initial" is low. In the end of the operation, only the data of the sign bit will be output according to Eqn.(1). The $\overline{Clock}$ controls the 2-phase operation of the adder; in phase 1, the logic operation is completed; and then in phase 2, the current data is updated with new data.

3 Simulation Analysis

The IRSIM of MAGIC is used to verify the performance of the chip. Fig. 11 is a timing diagram of recalling a stored pattern pair when the given retrieval pattern has one bit error. The time to recall the correct pattern is 80 ns. In short, we conclude the approximate time to recall a pattern pair in this chip is about (5 ns)$\times$(no. of pattern pairs)+(5 ns) $\times$(20 stages of pipeline transfer). The chip layout is shown in Fig. 12. The area of the chip is 1400×1400 μm^2.

4 Conclusion

The eBAM was still proved to be a high capacity associative memory which is worthy of hardware implementation. Our work has proven the pure digital realization is a feasible possibility. The exponent value generator and the pipeline IDA units are the key factors to make the digital design come true. Though the chip area is not economical, it is still in a reasonable range. In contrast, the speed of the chip is pretty fast.

References

[1] T. D. Chiueh, and R. M. Goodman, "High-capacity exponential associative memory," *Proc. IJCNN*, vol. I, pp. 153-160, 1988.

[2] T. D. Chiueh, and R. M. Goodman, "Recurrent correlation associative memories," *IEEE Trans. on Neural Networks*, vol. 2, no. 2, pp. 275-284, 1991.

[3] B. Kosko, "Bidirectional associative memory," *IEEE Trans. Systems Man Cybernet*, vol. 18, no. 1, pp. 49-60, Jan./Feb. 1988.

[4] P. K. Simpson, "Higher-ordered and intraconnected bidirectional associative memory," *IEEE Trans. Systems Man Cybernetics*, vol. 20, no. 3, May/June 1990.

[5] H. M. Tai, C. H. Wu, and T. L. Jong, "High-order bidirectional associative memory," *Electron. Lett.*, 25, pp. 1424-1425, 1989.

[6] Y.-F. Wang, J. B. Cruz, Jr., and J. H. Mulligan, Jr., "Two coding strategies for bidirectional associative memory," *IEEE Trans. Neural Network*, vol. 1, no. 1, Mar. 1990.

[7] C.-C. Wang, and H.-S. Don, "An analysis of high-capacity discrete exponential BAM," *IEEE Trans.. on Neural Networks*, vol. 6, no. 2, pp. 492-496, March 1995.

[8] T. Wang, X. Zhuang, and X. Xing, "Weighted learning of bidirectional associative memories by global minimization," *IEEE Trans. on Neural Networks*, vol. 3, no. 6, pp. 1010-1018, Nov. 1992.

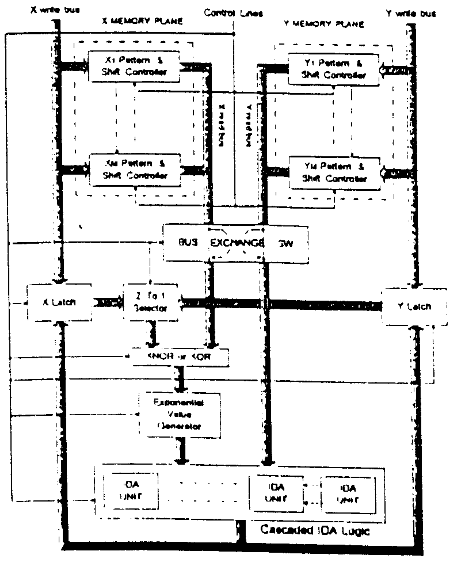

FIG.1 System Block Diagram

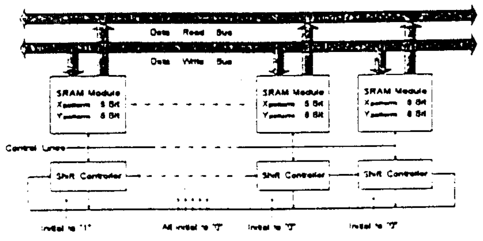

FIG.2 X-Y Memory Plane With Shift Controllers

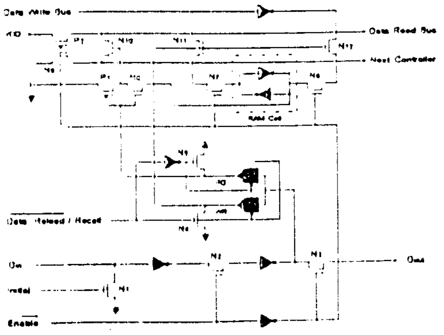

FIG.3 Shift Controller with RAM Cell

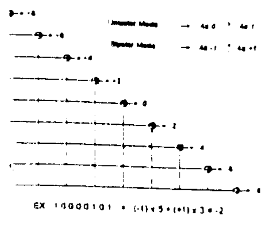
FIG.4 2's Exponential Value Generation Algorithm

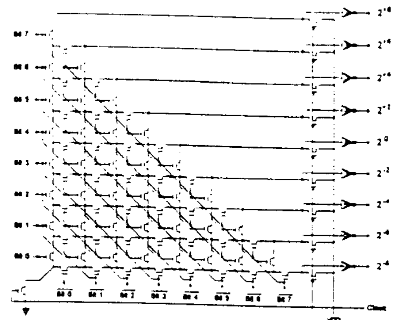
FIG.5 2's Exponential Value Generation Circuit

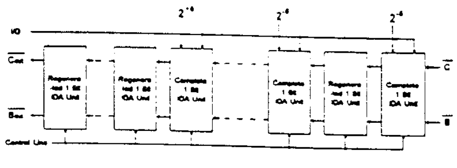
FIG.6 Cascaded INC/DEC Accumulator

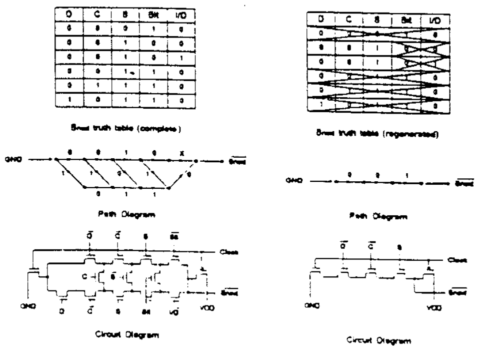
FIG.7 Bnext Generator

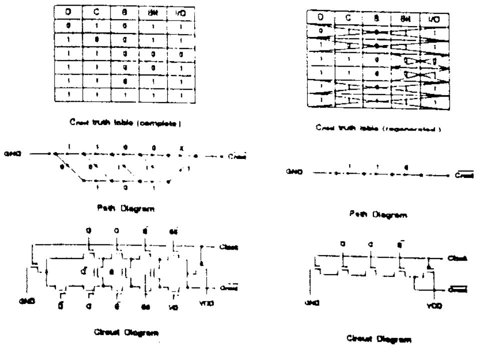
FIG.8 Cnext Generator

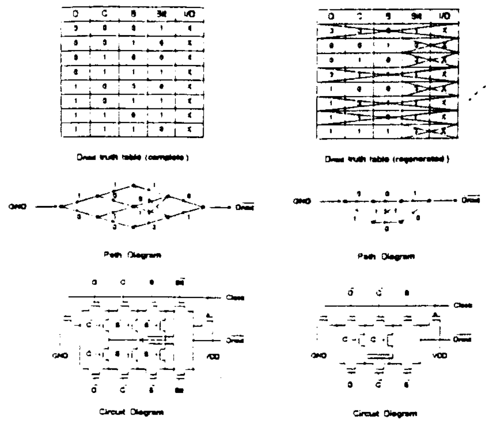
FIG.9 Dnext Generator

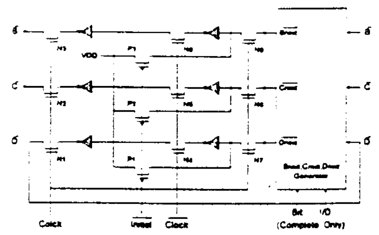
FIG.10 1 Bit INC/DEC Accumulator Unit Circuit Diagram

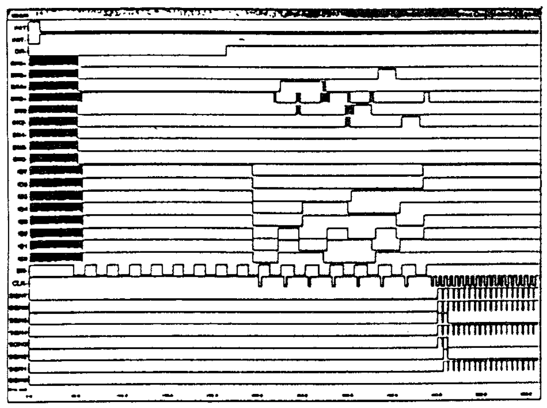
FIG.11 Recall Time Diagram

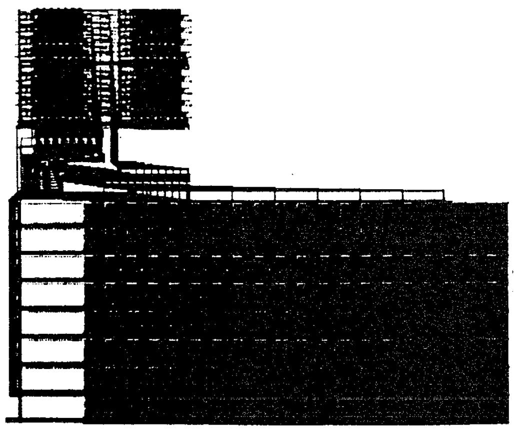
FIG.12 Chip Layout