Architectures for Associative Memories Using Current-Mode Analog MOS Circuits

Kwabena A. Boahen   Philippe O. Pouliquen
Andreas G. Andreou   Aleksandra Pavasovic

Electrical and Computer Engineering
The Johns Hopkins University
Baltimore, Maryland 21218

Scalable architectures for the implementation of associative memories that produce regular and dense designs are presented. A combination of low power consumption and good performance is achieved by using Current-Mode MOS circuits operating in subthreshold conduction. The design methodology and a 48-neuron memory chip are described.

1 Introduction

Biological information processing systems outperform modern digital machines in problems that require processing large amounts of fuzzy, noisy, real-world data, such as pattern recognition and classification. The shortcomings of conventional approaches have forced computer scientists and engineers to borrow paradigms from biology and cognitive science to solve problems in sensory perception and machine intelligence. One such paradigm is the *associative memory* that handles noisy and even novel inputs, has fault-tolerant properties, and the potential of a massively parallel implementation. Various forms of this model have been investigated by Kohonen [1,2] and Hopfield [3].

The *smart memories* project, using an elegant five transistor memory cell design [4], and work by Jones et al [5] emphasized digital VLSI content addressable memories for specialized computing engines. Furthermore, new parallel-programming languages, such as Linda [6], use *associative lookup* to create and coordinate processes. Digital implementations of more powerful associative memory models which search for *the best* match instead of an exact match have been pursued [7]. However, digital electronic implementation cannot match the massive parallelism and truly concurrent processing possible with analog circuitry, nor can they handle degraded signals [8].

In this paper we present two Analog VLSI architectures for associative memories that use current signals and employ native device physics to implement area-efficient computational primitives. Subthreshold Current-Mode circuits are used to achieve high functionality, density and performance. The design methodology is applicable to analog neural systems [9] as well as digital systems [10] and mixed analog/digital systems [8]. This approach offers a viable alternative to bipolar designs when low voltage operation and low power consumption are of prime concern.

The paper closes with a description of the chip that we fabricated using the first architecture, its performance, and the new architecture that evolved from that experience.

2 Associative Memory Models

Let $X = (x_1, x_2, \ldots, x_n)^T$ and $Y = (y_1, y_2, \ldots, y_m)^T$ represent the states of two neuron layers, of size n and m respectively [1]. A generic associative memory operates as follows:

In the *store* mode, the current state of each layer is stored, forming the association (A, B).

In the *recall* mode, the network converges to the stored state (A, B) nearest to its initial state (X, Y).

If the patterns in each stored pair (A, B) are identical, the memory is said to be *autoassociative,* otherwise it is *heteroassociative.*

Neurons receive inputs from neurons in another layer through synapses. A neuron's *activation* is the linear sum of these inputs weighted by the synaptic efficacies. We make the following distinctions:

- A *thresholding* neuron has two discrete states, $x = \pm 1$, determined by the sign of its activation, v. Thus $x = \mathrm{sgn}(v)$.
- A *non-thresholding* neuron's output equals its activation.

The network we introduce is both bidirectional and symmetric:

- In a *bidirectional* network, neurons in the A layer determine the states of those in the B layer, and vice versa.

[1]Column vectors will be denoted by capital letters and their components by small letters with appropriate subscripts.

- In a *symmetric* network, if the input to neuron i from neuron j is weighted by w_{ij}, then $w_{ji} = w_{ij}$.

2.1 Three-Layer Model

This model has two input/output (I/O) layers, F_A and F_B, with n and m thresholding neurons, respectively, and a hidden layer, F_U, with s non-thresholding neurons (Figure 1). Our network stores up

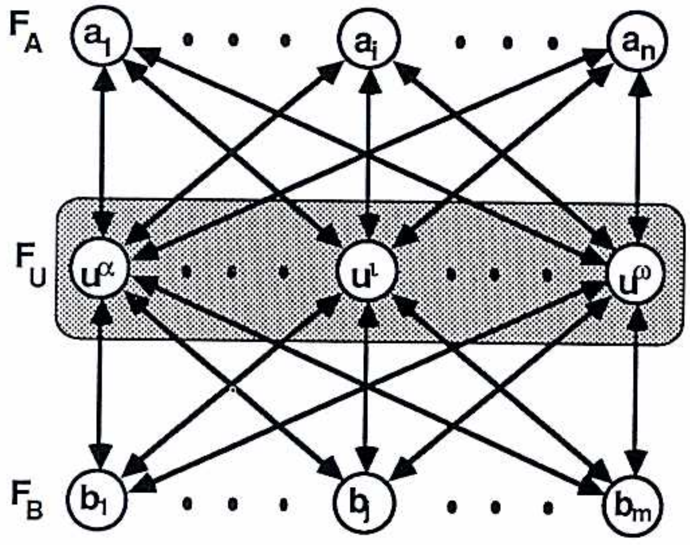

Figure 1: Three-Layer Bidirectional Associative Memory model. A middle-layer neuron (hidden unit) is assigned to each association stored.

to s associations, labeled by the index set Ω, which are programmed as follows:

> A hidden unit is assigned to each association. For association (A^ρ, B^ρ) the weights between the chosen hidden unit (also labeled with the superscript ρ) and neurons in F_A and F_B are simply set to the corresponding component of A^ρ or B^ρ. Thus, for F_A, $w_{\rho i} = w_{i\rho} = a_i^\rho$, and for F_B, $w_{\rho j} = w_{j\rho} = b_j^\rho$.

A similar, non-bidirectional network, was studied by Baum et. al. [11]. During recall, the states of neurons in either F_A or F_B are initialized. All the neurons are then allowed to update their states by thresholding their activation. We shall show that if the stored vectors satisfy certain conditions the vector pair closest to the initial state is recalled.

Formally, if the F_A neurons are initialized to A, then the recalled association (A^σ, B^σ) has the property

$$A^T A^\sigma = \max_{\rho \in \Omega} A^T A^\rho$$

Since the activation, v_j, of the jth neuron in F_B is

$$v_j = \sum_{\rho \in \Omega} w_{j\rho} u^\rho = \sum_{\rho \in \Omega} w_{j\rho} \sum_{i=1}^{n} w_{\rho i} a_i$$

and $w_{j\rho} = b_j^\rho$, $w_{\rho i} = a_i^\rho$, we have

$$v_j = \sum_{\rho \in \Omega} b_j^\rho \sum_{i=1}^{n} a_i^\rho a_i = \sum_{\rho \in \Omega} b_j^\rho A^T A^\rho \tag{1}$$

Rewriting this equation as

$$v_j = b_j^\sigma A^T A^\sigma + \sum_{\rho \in \Omega, \rho \neq \sigma} b_j^\rho A^T A^\rho$$

we find that $y_j = \mathrm{sgn}(v_j) = b_j^\sigma$ if

1. The inner product $A^T A^\sigma$ between the input vector A and the target vector A^σ is positive, and

2. The sum of the inner products between the input vector and the other stored vectors is less than $A^T A^\sigma$.

Under these conditions the jth neuron's state becomes b_j^σ and the closest vector B^σ is recalled. Feeding B^σ back through the network yields A^σ if the above conditions hold for the B vectors as well. The condition $A^T A^\sigma > 0$ guarantees that the complement of the target vector is not recalled. If these conditions fail, the recalled vector will be a combination of the stored vectors.

2.2 Equivalent Networks

This three layer network is equivalent to Kosko's two layer Bidirectional Associative Memory (BAM) [12].

Indeed, a two-layer BAM with n neurons in F_A and m neurons in F_B has an $n \times m$ connection matrix $M(= [m_{ij}])$ which is the sum of outer products AB^T :

$$m_{ij} = \sum_{\rho \in \Omega} a_i^\rho b_j^\rho \tag{2}$$

During recall, the activation of the jth neuron in F_B is

$$v_j = \sum_{i=1}^{n} m_{ij} a_i \tag{3}$$

Using Equation 2 and reversing the order of summation, we find

$$v_j = \sum_{i=1}^{n} a_i \sum_{\rho \in \Omega} a_i^\rho b_j^\rho = \sum_{\rho \in \Omega} b_j^\rho \sum_{i=1}^{n} a_i a_i^\rho$$

which is Equation 1.

Kosko proved that every real matrix M is a bidirectionally stable associative memory [12]. Therefore, the three layer BAM also has convergent trajectories for any set of stored vectors.

From Equation 2, it should be obvious that if $A^\rho = B^\rho$ for all ρ, the connection matrix is symmetric as in a Hopfield net [3] with n neurons.

2.3 Hardware Requirements

Of the two BAM models, the three layer one has the highest storage and computational efficiency, making it the best candidate for VLSI.

An $n \times n$ two layer BAM has n^2 weights which take on *integer* values, $|m_{ij}| \leq s$ (Equation 2), where s is the number of associations stored. These weights require $\log_2 s$ bits and a sign bit. Hence, a total of $n^2(\log_2 s + 1)$ bits are required to store s associations. Dividing the hardware bits required by the number of information bits stored, we obtain the storage *inefficiency*. In this case it is

$$b_{2L} = n(log_2 s + 1)/2s$$

With $s = n = 32$, for example, $b_{2L} = 3$.

A three-layer BAM of the same size has $2n + s$ neurons and $2ns$ *bipolar* weights, each represented by a single bit. Thus, the s vector pairs ($2n$ bits each) are stored using the optimal number of bits, that is, $b_{3L} = 1$.

The three-layer BAM is essentially a pair of two-layer BAMs. Therefore, the three-layer model requires twice as much hardware for the recall operation. However, in the two-layer model, stored binary representations for the weights must be adjusted by

$$m'_{ij} = m_{ij} + a_i b_j$$

(See Equation 2) for each new association (A, B), whereas synapses in the three-layer BAM are programmed by storing the vectors A and B directly.

It should be pointed out that the analysis would be different if the weights could be stored *and* manipulated in *analog* form. Clearly, our choice to implement the three-layer model was influenced by the lack of a compact nonvolatile analog memory element in VLSI technology.

2.4 Architecture

Our architecture is based on a regular array of BAM cells. Each BAM cell consists of two synapses and a one-bit memory. This pair of synapses provides two-way communication (bidirectionality) between neurons in the I/O layers and the hidden layer. The memory bit determines the state of both synapses (symmetry). Figure 2 shows a 3×3 BAM that stores up to four associations (one vector pair per row). This figure illustrates how neurons in the three layers

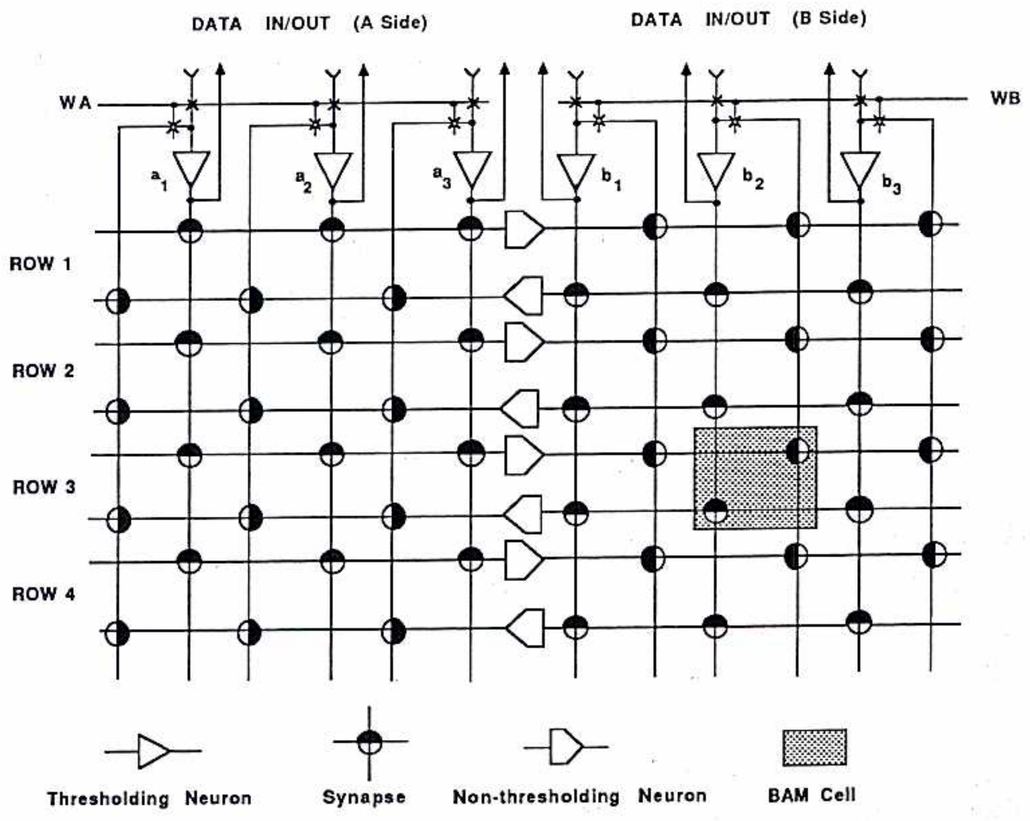

Figure 2: Three-Layer BAM Chip Architecture. The BAM cell is replicated to produce networks of any size.

communicate through the BAM cells. The input and output lines of the thresholding neurons at the top run vertically, while those of the nonthresholding neurons in the middle run horizontally. In general, communication in a BAM with n neurons in each I/O layer and $2s$ hidden-layer neurons is supported by two $n \times s$ BAM cell arrays. However, the number of neurons in the I/O layers need not be the same.

Each association (A^ρ, B^ρ) is stored in the BAM arrays, such that bit a_i^ρ (or b_j^ρ) of vector A^ρ (B^ρ) is stored in the BAM cell at row ρ and column a_i (b_j).

In the recall mode, the input vector is presented to one side, for example the A side, and WA is asserted. This initializes the state of the A neurons (refer to Figure 2). At the same time, the feedback is decoupled, allowing the A neurons to launch the network toward the desired stable state. After WA is deasserted, the network relaxes, converging towards the recalled association.

3 Technology

Analog electronic implementations of associative memory models require high degrees of connectivity, that is, large fan-in and fan-out. Our implementation uses transconductances as coupling elements to achieve large fan-out. These transconductances are simply MOS transistors operating in subthreshold conduction. Small voltage signals are applied to the isolated gate of the transistor to obtain low conductance current outputs at the drain. The fan-in problem is solved by using neurons with current inputs and obtaining the sum of all these currents on a single input line.

Although our circuits operate with very small subthreshold currents, reasonable speeds are achieved by keeping voltage swings small to reduce slewing time. This is possible because of the logarithmic dependence of the gate voltage on the channel current in the subthreshold region [9]. On the other hand, when delivering a current signal to a circuit, both voltage swings and propagation delays are inversely proportional to the input conductance. Thus, by taking advantage of the high transconductance of MOSFET's in subthreshold conduction [13], we obtain a good power/speed trade-off.

Dynamic power dissipation and supply noise are reduced as a result of the smaller capacitive charging/discharging currents produced by the voltage swings. Concurrently, quiescent dissipation is low because small current signals are used. This technology yields relatively fast analog circuits with power dissipation levels compatible with wafer-scale integration.

3.1 Subthreshold MOSFET Operation

We operate the MOS transistor in the "off" region, characterized by $V_{gs} < V_{th}$. This is referred to as the *weak-inversion* or *subthreshold conduction* region. In this region MOS devices are barrier controlled with transconductance similar to that of BJTs [13].

The transfer characteristics are shown in Figure 3. Notice that the channel current I_{ds} is exponentially dependent on the gate voltage V_{gs}

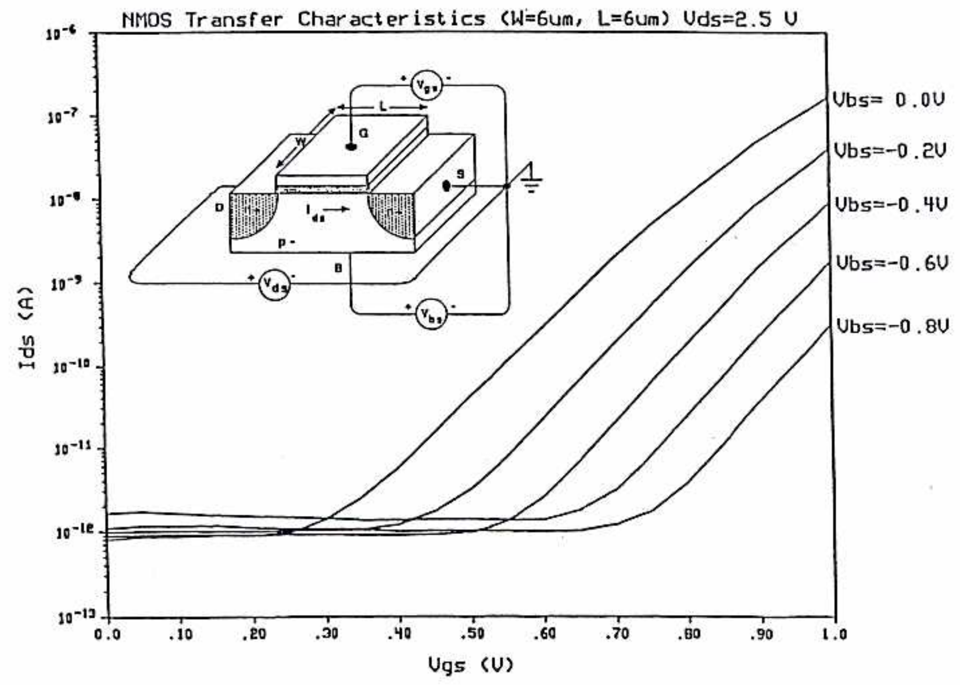

Figure 3: Subthreshold Characteristics for an N-type MOS transistor. The variation of the channel current with the substrate voltage is included to point out that the MOS transistor is a four terminal device.

and bulk (local substrate) voltage V_{bs} over nearly *six* decades. In the saturation region, the drain current in an n-type transistor is given by

$$I_{dsat} = \left(\frac{W}{L}\right) I_0 e^{(\frac{1}{\gamma}V_{gs}+\frac{1}{\eta}V_{bs})/U_T} \qquad V_{ds} > V_{dsat} \simeq 4U_T \qquad (4)$$

W, L are the *effective* channel width and length, respectively.

I_0 is a process-dependent parameter.

γ, η measure the ineffectiveness of the gate and substrate potentials in reducing the barrier. The values $\gamma = 1.9$ and $\eta = 3.4$ for the characteristics shown are typical for digital-oriented CMOS.

U_T $(= \frac{kT}{q})$ is the thermal voltage. It is $26mV$ at room temperature.

Typically, I_{ds} changes by a decade for a $120mV$ change in V_{gs} or a $280mV$ change in V_{bs}. From Equation 4, the transconductance is

$$g_m = \frac{\partial I_{dsat}}{\partial V_{gs}} = \frac{I_{dsat}}{\gamma U_T} \tag{5}$$

An empirical relationship for the drain conductance is

$$g_{dsat} = \frac{I_{dsat}}{V_0 + V_{ds}} \tag{6}$$

where V_0 is the Early voltage (typically about $55V$). This relation captures the slope in the output characteristic caused by the dependence of L on V_{ds} [14]. These equations sacrifice accuracy for simplicity; they are only meant for rough design calculations.

The fact that $g_m \gg g_{dsat}$ makes the MOS transistor a versatile circuit element:

- In the common-source configuration, it is a *transconductance amplifier*, with a low-conductance current output at the drain terminal.
- In the common-drain configuration, it is a *voltage follower,* with a low-impedance voltage output at the source terminal.
- In the common-gate configuration, it is a fast *current buffer,* with a low-conductance current output at the drain.

Using voltage signals as well as current signals makes it possible to employ MOS transistors in all three configurations.

Current mirrors are the primary computational element used in current-mode circuits. In addition to replicating currents, the mirroring operation is used to invert and scale currents. Thus, the current mirror is a trivial example of a translinear circuit, from which more general algebraic (non-linear) functions can be designed using the *Translinear Principle* [15]. However, variations in substrate voltage, geometry, or doping can produce variations in the output current [16].

4 Circuits

Using the technology previously described, we designed the circuits required to implement associative memories.

4.1 Neurons

Thresholding neurons are simple MOS inverters which receive bipolar current inputs from the synapses. These currents are integrated over time by the interconnect capacitance, so that the input voltage represents activation. Neurons switch to the +1 state (or the −1 state) when this voltage exceeds (falls below) the inverter's threshold ($V_{inv} \approx V_{dd}/2$), and remain in the same state when the net input current is zero.

Thresholding neurons drive the synapses through the bias circuit, which performs two key functions: It allows the synaptic current levels to be externally programmed, and provides an interface between the thresholding neuron and the synapses. Given the state a of the thresholding neuron and a programmed current level, the bias circuit generates a voltage V_a (Figure 4a). An identical circuit generates $V_{\overline{a}}$ using $\overline{a}$. These voltages produce copies of the programmed current in

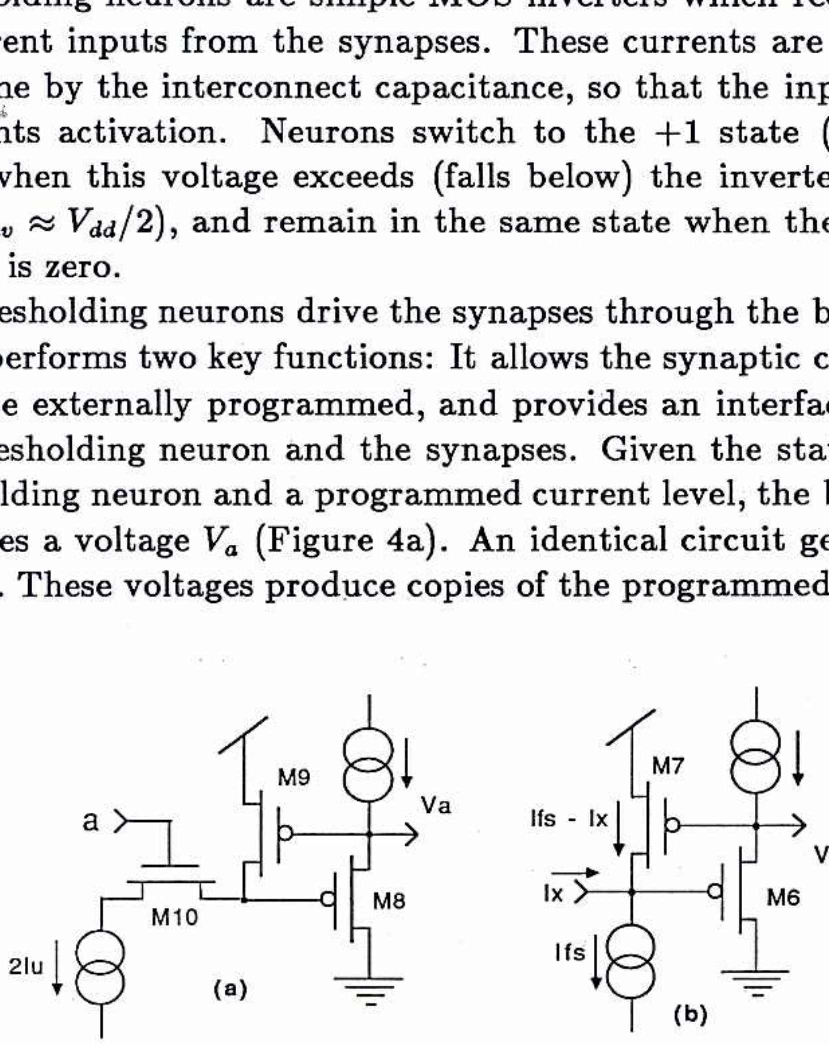

Figure 4: Circuit diagrams for (a) the bias circuit and (b) a non-thresholding neuron. All transistors are minimum-size ($3\mu m \times 6\mu m$).

the synapses. The circuit operates as follows: When a is high, M_{10} is switched on and M_9 sources the programmed current, or $2I_u$, where I_u is the desired unit synaptic current. The gate voltage of M_9 is set appropriately by feedback through M_8 which senses and corrects any current imbalance. This produces a low-impedance voltage output at V_a.

Nonthresholding neurons accept a bipolar input current I_x and generate an output voltage V_x which drives the synapses directly. The circuit used is shown in Figure 4b. An identical circuit is fed $-I_x$ to produce $V_{\overline{x}}$. These circuits operate like the bias circuit. I_{fs} is simply a DC shift introduced to guarantee that there is always current in M_7; it is set to the full–scale current input. The output voltages are used to produce copies of the current in M_7 $(I_{fs} \pm I_x)$ elsewhere.

4.2 Synapses

The output current of a synapse is given by

$$I_{out} = cI_{in} \tag{7}$$

where $c = \pm 1$ is the state of the synapse. The input current I_{in} may have either sign. Thus the synapse performs a four-quadrant multiplication by a one-bit weight. The circuit used is shown in Figure 5. Instead of supplying the input current I_{in} directly to the synapses it is

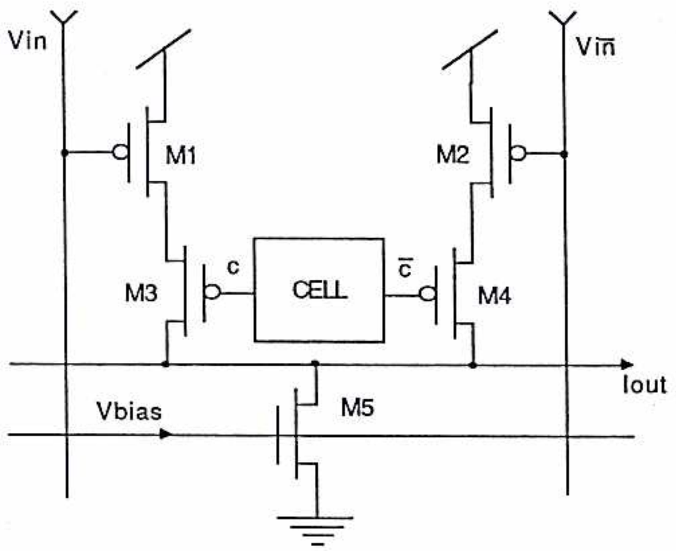

Figure 5: Circuit diagram for a synapse. All transistors are minimum size.

encoded as a pair of voltages V_{in} and $V_{\overline{in}}$ that are applied to the gates of transistors M_1 and M_2. They are set to obtain a drain currents of $I_{fs} - I_{in}$ and $I_{fs} + I_{in}$, respectively. I_{fs} is removed at the output by M_5 which is biased with V_{bias}. This scheme allows I_{in} to be replicated in several synapses using the same lines.

The state c of the synapse is represented by a voltage at $GND(-1)$ or at V_{dd} $(+1)$ in the memory cell. In the former case, M_3 is on and M_4 is off so that M_1 and M_3 together supply $I_{fs} - I_{in}$ to the output node. In the latter case, the reverse is true, hence M_2 and M_4 supply $I_{fs} + I_{in}$. Clearly, if M_5 subtracts I_{fs} the desired operation is obtained (Equation 7).

When the thresholding neurons drive the synapses we have $I_{in} = aI_u$ and $I_{fs} = I_u$ so that

$$I_{out} = caI_u \tag{8}$$

where a and c are the states of the thresholding neuron and the synapse, respectively. On the other hand, when the synapses are driven by the nonthresholding neuron we find $I_{in} = I_x$ and

$$I_{out} = cI_x \tag{9}$$

where I_x is the input current to the nonthresholding neuron. In the former case the synapses compute the scalar products which give the activation of the nonthresholding neurons while in the latter they compute weighted sums to determine the activation of the thresholding neurons. Refer to Equation 1.

We have implemented these functions using simple circuit configurations and minimum-size transistors. Consequently, their accuracy is highly dependent on the fabrication process, i.e. variations of g_m, g_{dsat}, and I_0. The rationale behind this approach is that by studying the short-comings of these simple circuits we can justify any additional complexity and thereby develop an efficient design methodology.

5 Chip

5.1 Design

An experimental chip was designed and fabricated through MOSIS (production run M8AV-VERDEAN $3\mu m$ p-well CMOS) using the neural circuits and the architecture presented in the previous sections. A microphotograph of the die is shown in Figure 6.

There are two 8×16 BAM cell arrays, 16 thresholding neurons on either side, and 16 nonthresholding neurons in the middle. The BAM cell size was limited by the large number of global interconnects running between the synapses and neurons. There are six vertical lines carrying $V_a, V_{\overline{a}}, V_{b1}, I_{out}, V_{dd}$, and GND, and four horizontal lines carrying $V_x, V_{\overline{x}}, I_{out}$ and V_{b2}. The former were run in metal-2 while the latter were run in metal-1.

We used shift registers to store the vectors, transferring data with polysilicon lines between registers while making it available to the

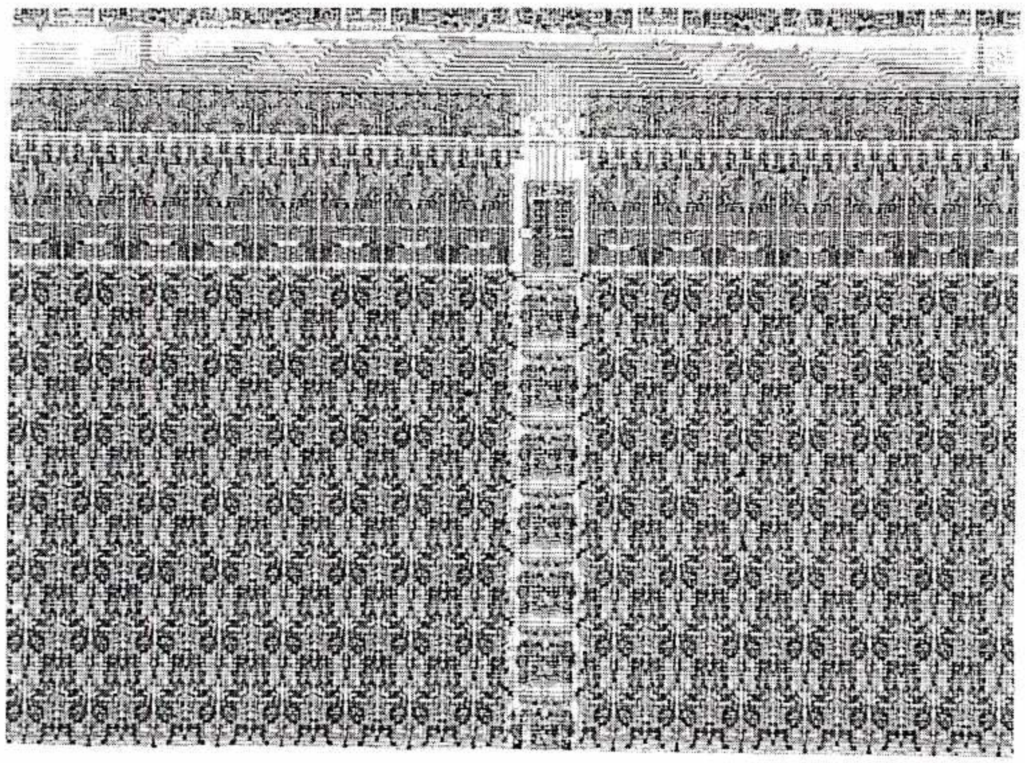

Figure 6: Die Microphotograph. The die size is $2.3mm \times 3.4mm$ with $4.8mm^2$ of active area and 7900 transistors.

synapses. A thirteen-transistor complementary static-dynamic shift register cell design was used. Data shifts downwards, the top shift register gets data from the bus and the data in the bottom register is lost. The final BAM cell had twenty-three transistors (five in each synapse) and a layout of size $104\lambda \times 48\lambda$.

The current design incorporates some important improvements over the original version. The first chip had sized-up output devices in the bias circuits and nonthresholding neurons to provide the necessary fan-out capability. However, testing showed that recall rates were limited by the slewing time of the synapses which had to drive inputs of the thresholding neurons from the rails to V_{inv}. In the new designs, minimum-size devices are used throughout. This reduced the power consumption and allowed an eighth row of BAM cells to be added and the bias circuits to be redesigned to reduce the voltage swings at the output.

5.2 Performance

We evaluated recall performance not only for eight different vector pairs, but two and four as well, simply by storing multiple copies of each vector. As we increased the number of (different) stored vectors, recall degraded, i.e. when presented with a vector that is actually stored, the BAM would recall a different vector pair (not necessarily one that was stored).

When we stored two vectors that differed in 12 or 14 bits, presenting either vector resulted in the *complement of the other vector* being recalled. However, the complements of these vectors were recalled correctly. This problem did not occur with vectors that differed in 8 or 10 bits: The actual vectors *and* their complements were always exactly recalled. This supported our suspicion that in the synapses which computed scalar products, the magnitude of the output current was larger for a mismatch than for a match. Our results indicate that a mismatch is 1.7 to 2.2 times stronger than a match. Similarly, when we presented a vector that was exactly midway between the stored vectors, recalls were biased towards 1's. It is possible that the extra mirroring operations required to drive M_1 and M_2 (see Figure 5) introduce these errors. This can be avoided by using a design that is symmetric in this respect.

We also observed that when both stored vectors had the same number of 1's, and the presented vector was midway between the two, the chip recalled the vector that was closest to the bottom of the BAM arrays (ie in the eighth storage location). This behavior may be due to a gradation in the transistor parameters.

Recall performance was good for two stored vectors when they differed by at least four bits, but was significantly better when they differed by at least six bits. When more than two vectors were stored, the chip performed well only if the vectors were orthogonal. Storing eight vectors reliably required choosing them such that they all had the same number of ones. Under these circumstances, the chip had sixteen stable states, all of which were within a Hamming distance of four from the actual stored vectors.

However, no matter what was stored in the chip, or what was presented for recall, the chip always relaxed rapidly. A test structure was used to measure the speed of the circuits: Two neurons were connected through two synapses with different states. This asymmetrical connection made the circuit oscillate at approximately 1 MHz with a bias current of $0.5\mu A$. Therefore a generalized BAM structure should relax in about $2\mu S$.

6 Improvements

To reliably recall large numbers of patterns the nonthresholding neurons must perform a *nonlinear expansion.* That is, small inputs must be attenuated while large inputs are amplified. This reduces the interference between the stored patterns. We investigated exponential functions for this purpose. Although results from simulations were encouraging, we found that simple current-domain circuits were not suitable for their implementation. A more natural solution is a "winner-take-all" network that picks the hidden unit with the largest input and shuts off all others. The nice feature of this network is that the maximum output can be normalized whereas an exponential is not well-behaved for large inputs. This network may be implemented with a current steering circuit using a single global line [17]. In this section we introduce the circuits required to implement this improved architecture.

To reach levels of density comparable to CAMs and SRAMs we need to reduce the number of interconnects required drastically. This is possible using the two-transistor synapse shown in Figure 7. The

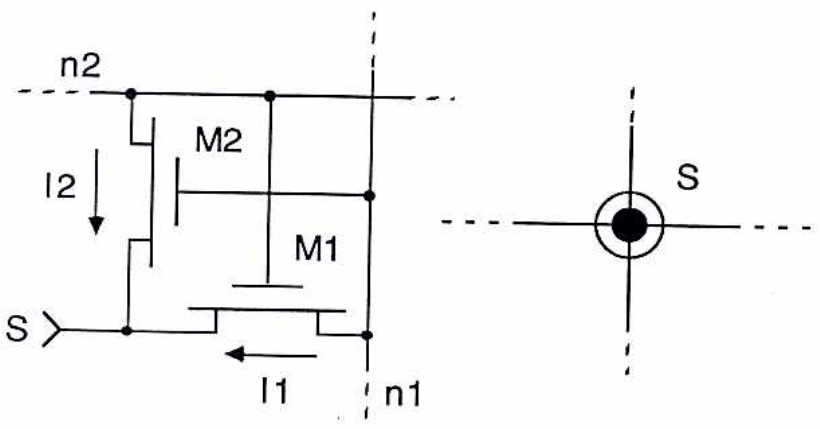

Figure 7: Two-transistor synapse circuit and its symbol. This circuit provides two-way communication between neurons connected to nodes n_1 and n_1. The connection is turned off by setting s high.

idea is to use a *single* line to send information to and from the synapses. Voltage signals are sent down the line while current signals are fed back. When s is low the voltages are transformed into currents at the synapse to provide two-way communication. If the voltages applied exceed $V_{dsat} \approx 100mV$, the transistors are in saturation so that small voltage swings do not change the current. When

s is high, $|V_{gs}|$ is the difference between the voltages at nodes n_1 and n_2 for both transistors. Both transistors turn off if this difference is small. The symbol for the synapse, which is basically a bidirectional transconductance, is also shown in Figure 7.

The neurons at n_1 and n_2 communicating through the synapse apply voltages to the gates of M_2 and M_1 while sensing the currents I_1 and I_2, respectively. A neuron may communicate through several synapses. Currents from these synapses simply sum onto the neuron's I/O line, while the voltage on this line is transformed to a current at each synapse. This neuron is realized by the four-transistor circuit shown in Figure 8. It consists of a current buffer (M_4), indicated

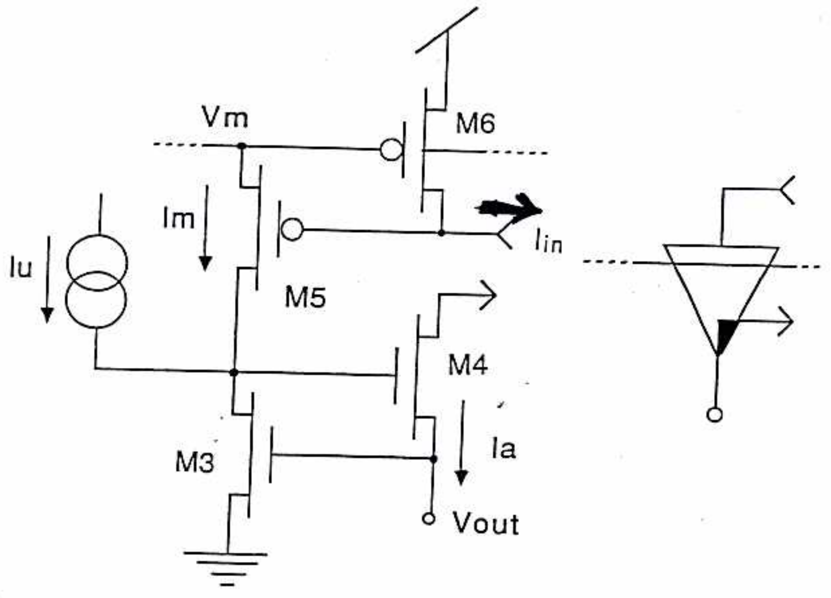

Figure 8: Four-transistor neuron circuit and its symbol. This circuit drives the synapses while buffering current signals feeding into the neuron. The neuron has a spontaneous level of activity determined by I_u and is inhibited by lowering V_m.

by the black triangle in the symbol, and a thresholding input circuit (M_5, M_6). The current in M_3 determines the output voltage, feedback through M_4 keeps the output impedance low. M_6 shuts off M_5 if I_{in} does not exceed the saturation current in M_6. Thus, V_m sets the threshold at which the current I_m from the global line is dumped into M_3. The neuron has a spontaneous level of activity I_u.

The new architecture is shown in Figure 9. The BAM cell has two complementary synapses—one is off while the other is on. The I/O unit has two neurons that compete to recall a zero or a one. The winning neuron sinks $I_m = I_u$. This current is mirrored in the

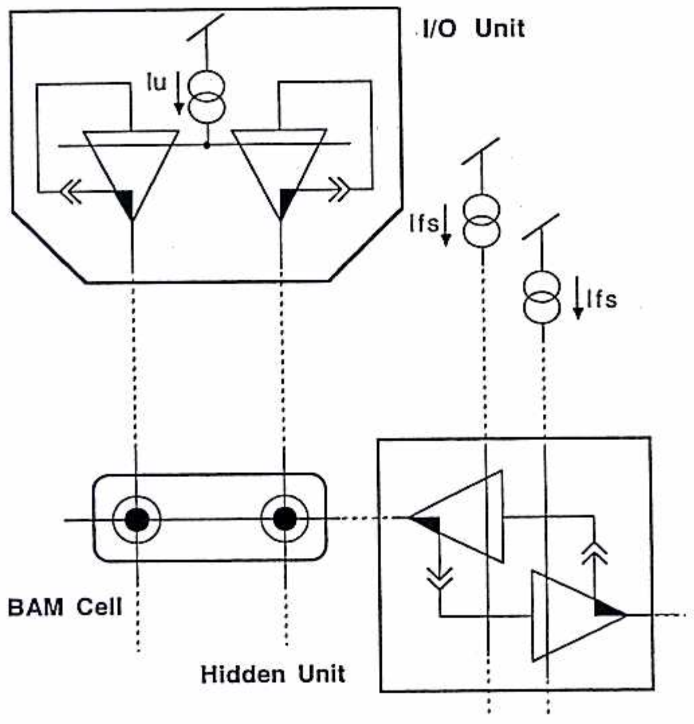

Figure 9: Improved architecture showing organization of I/O units, hidden units and the BAM cells. The two neurons in the I/O units compete to recall a one or a zero. The hidden units are organized as a "winner-take-all" network.

BAM cell if its state matches the state of the I/O unit. Currents from the BAM cells sum into neurons in the hidden units. These neurons compete to sink $I_m = I_{fs}$. The neuron in the row with the most matches wins and mirrors I_{fs} in the BAM cells on the other side. If I_{fs} is sufficiently large the I/O unit is forced to match the state of the BAM cell in the winning row.

7 Conclusion

We have designed and fabricated a dense, repeatedly programmable, neural model for a heteroassociative memory. We obtain high density by using local storage at the expense of fault tolerance. However, we can store two or three copies of each vector and still use less digital memory than a distributed matrix scheme.

Current-mode circuits operating in the subthreshold region are used to achieve large fan-in and fan-out and low power dissipation. A scalable architecture results from employing coupling elements in a highly regular structure and avoiding the use of resistors. The speed of our network is limited by the ability of each synapse to charge/discharge its local parasitic load and not by the size of the network. By keeping voltage swings small we obtain fast operation. Using current inputs allows the interconnects to perform useful computation, and thus permits more efficient use of the silicon.

The system described in this paper has evolved around a simple principle: *"Communication is Computation."* Perhaps that is how biological information processing systems circumvent the bottlenecks of traditional computing.

Acknowledgments

We would like to thank Prof. C. R. Westgate for his support, Kim Strohbehn for helping out with the CAD tools, and Fernando Pineda for critically reviewing the theory.

Special thanks is due Prof. Carver Mead for making available a preprint of [9], used as the text for the Analog VLSI class at Johns Hopkins, which inspired many of our ideas. This chip was a class project whose fabrication was funded by NSF.

References

[1] T. Kohonen, *Associative Memory: A System Theoretic Approach,* Springer Verlag, Berlin, Heidelberg, New York 1977.

[2] T. Kohonen, *Self-Organization and Associative Memory,* Springer Verlag, (2nd edition), Berlin, Heidelberg New York, 1988.

[3] J.J. Hopfield, "Neural networks and physical systems with emergent computational abilities," *Proc. Nat. Acad. Sci. USA,* vol. 79, pp. 2554-2558, 1982.

[4] J.P. Wade and C.G. Sodini, "A Dynamic Cross-Coupled Bit-Line Content Addressable Memory Cell for High-Density Arrays," *IEEE J. of Solid-State Circuits,* vol. 22, pp. 119-121, 1987.

[5] S.R. Jones, I.P. Jalowiecki, S.J. Hedge and R.M. Lea, "A 9-kbit Associative Memory for High-Speed Parallel Processing Applications," *IEEE J. of Solid-State Circuits,* vol. 23, pp. 543-548, 1988.

[6] N. Carriero and D. Gelernter, "Applications Experience with Linda," *Proceedings ACM/SIGPLAN Symposium on Parallel Programming,* July 1988, pp. 173–187.

[7] T. Kohonen, *Content–Addressable Memories,* Springer Verlag, Berlin, Heidelberg, New York 1980.

[8] J. C. Platt and J. J. Hopfield, "Analog Decoding Using Neural Networks," in *AIP Conference Proceedings 151: Neural Networks for Computing,* Snowbird, UT. Editor: J. S. Denker. AIP 1986.

[9] C.A. Mead, *Analog VLSI and Neural Systems,* Addison-Wesley, Reading, MA (in press).

[10] M. Kameyama, S. Kawahito and T. Higuchi, "A Multiplier Chip with Multiple–Valued Bidirectional Current–Mode Logic Circuits," *Computer,* pp 43–56, April 1988.

[11] E. B. Baum, J. Moody, and F. Wilczek, "Internal Representation for Associative Memory," *Biol. Cybern.,* vol. 59, pp. 217-228, 1988.

[12] B. Kosko, "Bidirectional Associative Memories," *IEEE Trans. Syst. Man. Cybern.,* vol. SMC-18, pp. 49-60, Jan./Feb. 1988.

[13] E. A. Vittoz and J. Fellrath, "CMOS Analog Integrated Circuits Based on Weak Inversion Operation," *IEEE J. Solid–State Circuits,* vol. SC–12, pp. 224-231, June 1977.

[14] Mary Ann C. Maher and C.A. Mead, "A Physical Charge-Control Model for MOS Transistors," in *Advanced research in VLSI: Proceedings of the 1987 Stanford Conference,* Paul Losleben, Ed., The MIT press, pp. 211-229, 1987.

[15] B. Gilbert, "A Monolithic 16–Channel Analog Array Normalizer," in *IEEE Journal of Solid State Circuits,* Vol. SC-19, No. 6, Dec. 1984.

[16] A. Pavasovich, A.G. Andreou and C.R. Westgate, "An Investigation of Minimum–Size, Nano–Power, MOS Current Mirrors for Analog VLSI systems," JHU Electrical and Computer Engineering Technical Report, JHU/ECE 88-10.

[17] J. Lazzaro, C. A. Mead, "A Physical Realization of the Winner–Take–All Function," *NIPS '88*, Denver CO, Nov. 1988.