PERGAMON

Neural Networks 14 (2001) 781–793

Neural Networks

www.elsevier.com/locate/neunet

2001 Special issue

Probabilistic synaptic weighting in a reconfigurable network of VLSI integrate-and-fire neurons

David H. Goldberg*, Gert Cauwenberghs, Andreas G. Andreou

Department of Electrical and Computer Engineering, Johns Hopkins University, 3400 N. Charles St., 105 Barton Hall, Baltimore, MD 21218, USA

Received 3 January 2001; revised 21 March 2001; accepted 21 March 2001

Abstract

We present a scheme for implementing highly-connected, reconfigurable networks of integrate-and-fire neurons in VLSI. Neural activity is encoded by spikes, where the address of an active neuron is communicated through an asynchronous request and acknowledgement cycle. We employ probabilistic transmission of spikes to implement continuous-valued synaptic weights, and memory-based look-up tables to implement arbitrary interconnection topologies. The scheme is modular and scalable, and lends itself to the implementation of multi-chip network architectures. Results from a prototype system with 1024 analog VLSI integrate-and-fire neurons, each with up to 128 probabilistic synapses, demonstrate these concepts in an image processing task. © 2001 Elsevier Science Ltd. All rights reserved.

Keywords: Integrate-and-fire neuron; Spiking neuron; Probabilistic synapse; Address-event representation; Analog VLSI

1. Introduction

The human brain's impressive computational abilities are, to a large extent, attributable to its ability to process information in a parallel and distributed manner. This parallel, distributed architecture is enabled by the massive connectivity of the brain's neurons. Such an architecture gives rise to a system whose computational capabilities are much greater than the sum of its constituent parts.

The brain uses action potentials or 'spikes' to transmit information both internally and externally. Spike communication facilitates robust long-distance communication by means of self-restoring, all-or-none signals. The brain has developed elaborate information coding schemes based on its massively connected architecture and spike communication. Only recently have researchers begun to unravel the mystery of how neural systems encode sensory information with spikes (Rieke, Warland, de Ruyter van Steveninck & Bialek, 1997).

The brain has on the order of 10^{14} synaptic connections (Koch, 1999, p.87) and a power budget on the order of 15 W (Aiello & Wheeler, 1995). In a structure with so many connections, it is crucial that communication between neurons be energy efficient. It is possible that neural systems employ spike communication not just because it is robust to noise, but energy efficient as well. It has been suggested that neural coding schemes are in some sense optimized with respect to the tradeoff between information capacity and power consumption (Abshire & Andreou, 2000; Levy & Baxter, 1995).

The brain's performance in image processing and pattern recognition tasks far exceeds that of today's state-of-the-art artificial systems. Massively connected architectures and spike coding realize in the brain a computer that is very different from conventional serial digital computers. We believe that in order to close the performance gap between artificial and natural systems, we must design computers that draw inspiration from the brain's function and structure. Modern complementary metal-oxide-semiconductor (CMOS) very large-scale integration (VLSI) technology provides us with a medium that is well-suited to the implementation of neural systems. MOS devices can operate in regimes where their physics are identical to that of key neural structures, and VLSI technology enables extremely dense integration of these devices (Mead, 1990). We believe that the coupling of this technology with neurally inspired architectures will allow us to exploit the potential of spike coding schemes, and ultimately enable us to close the performance gap.

The massive connectivity of the brain is impossible to directly implement in VLSI due to the limitations on connectivity within and between microchips. However, we can take advantage of the temporally sparse nature of spike codes and the high bandwidth of VLSI systems in order to

* Corresponding author. Tel.: +1-410-516-8361; fax: +1-410-516-8313.
E-mail address: goldberg@jhu.edu (D.H. Goldberg).

PII: S0893-6080(01)00057-0

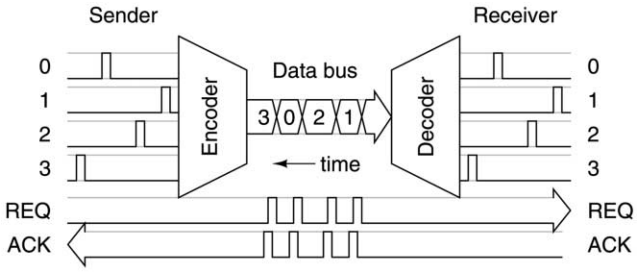


Fig. 1. Address-event representation. Sender events are encoded as addresses, sent over the bus, and decoded at the receiver. Handshaking signals REQ and ACK are required to ensure that only one cell is communicating at a time. Note that the time axis goes from right to left in this figure.

overcome this connectivity problem by time-multiplexing signals from many connections on the same data bus. Address-event representation (AER) is a communication protocol that was developed for this purpose (Mahowald, 1994) (Fig. 1). Suppose we have an array of cells that encode their activity in the form of spikes, and we want to transmit these activities to another array of cells. The 'brute force' approach to transmitting the activities would be to use one wire for each pair of cells, requiring N wires for each of N cell pairs. In an AER system, however, the location of a spike on the sender is encoded as an address, which is sent across the data bus. The receiver decodes the address and reconstructs the sender's activity. Handshaking signals REQ and ACK are required to ensure that only one cell pair is using the data bus at a time. This scheme reduces the number of wires required from N to $\log_2 N$. Two pieces of information uniquely identify a spike: its location, which is explicitly encoded as an address, and the time that it occurs, which need not be explicitly encoded because time represents itself. The encoded spike is called an *address-event* (AE).

Because AER was originally formulated to emulate the optic nerve (Mahowald, 1994) and the auditory nerve (Lazzaro, Wawrzynek, Mahowald, Sivilotti & Gillespie, 1993), it implements a one-to-one connection topology. To implement more complex neural circuits, convergent and divergent connections are required. Several authors have discussed and implemented methods of extending the connectivity of AER systems to this end (Boahen, 2000; Deiss, Douglas & Whatley, 1999; Grossberg, Carpenter, Schwartz, Mingolla, Bullock, Gaudiano et al., 1997; Higgins & Koch, 1999). These methods call for a memory-based projective field mapping which enables the projection of an address-event to multiple receiver locations.

In this paper, we propose a scheme that employs probabilistic synaptic weighting in conjunction with AER and an integrate-and-fire transceiver to implement reconfigurable neural architectures in VLSI. We demonstrate that AER can facilitate *computation* in addition to communication. In Section 2, we describe this scheme. In Section 3, we discuss some theoretical issues that arise in networks of integrate-and-fire neurons with probabilistic synapses. In Section 4, we describe how our framework can be applied to implement the early stages of the boundary contour system (Grossberg & Mingolla, 1985), a biological theory of vision processing. In Section 5, we describe a hardware prototype system that demonstrates these ideas. In Section 6, we report experimental results from the prototype system in an image processing task. Finally, in Section 7, we conclude the paper and discuss some future goals.

2. Address domain computation

We augment the traditional AER system to create a scalable, reconfigurable architecture that is capable of implementing a wide range of network topologies. A routing circuit between the sender and receiver probabilistically routes AEs to multiple destinations in a receiving array of integrate-and-fire (IF) cells. The IF cells enable the temporal and spatial integration of excitatory and inhibitory AEs, and send AEs as output (Fig. 2).

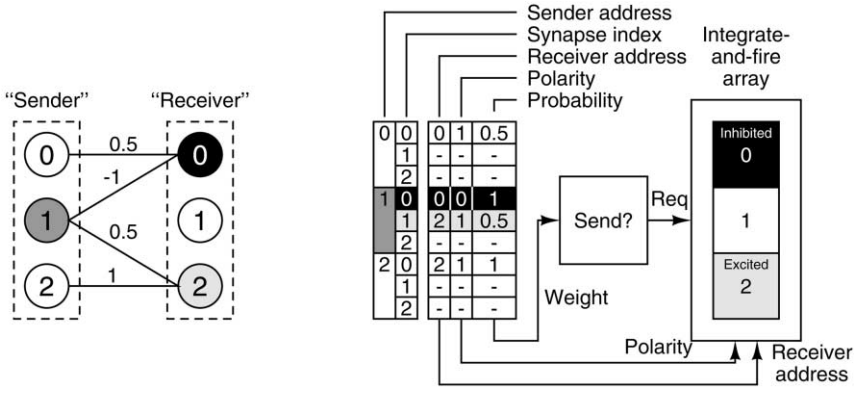


Fig. 2. Mapping of a two-layer network into an AE look-up table with transmission probabilities. In this example, sender 1 sends an AE. An inhibitory AE is transmitted to receiver 0 with 100% probability, and then an excitatory AE is transmitted to receiver 2 with 50% probability. In the implementation, the synaptic connection table is stored in a random-access memory (RAM). The first two columns comprise the memory address, and the remaining columns comprise the memory data.

A two-layer network of arbitrary interconnection topology can be mapped to the AER framework by means of a memory-based look-up table. The first layer of the network can be thought of as the 'sender' and the second layer can be thought of as the 'receiver'. AEs from the sender go to the look-up table, which contains information about the destinations of the AE, the polarity of the received AE, and the weight of the connection. The possibility of multiple AE destinations enables convergent and divergent connections.

The weight of a connection can be interpreted as the probability of transmitting an AE from a sender location to a receiver location. The system probabilistically gates the transmission of AEs to implement synaptic weighting. This scheme can only implement synaptic weights between -1 and $+1$. To obtain weights with strength greater than unity, we can simply iterate the process M times, effectively scaling the weight by a factor M.

Each IF cell in the receiver integrates excitatory and inhibitory AEs from several locations and sends AEs as output. Because the IF array both transmits and receives AEs, we call it a transceiver. Each cell has an associated potential which is incremented (decremented) by excitatory (inhibitory) AEs. The potential is initialized to zero and cannot go below zero; when the potential exceeds a threshold, the cell sends an output AE and the potential is reset to zero.

The integrate-and-fire model is analogous to cells in the nervous system that receive EPSPs and IPSPs at the dendrite and sum them to give a membrane potential. When the membrane potential exceeds a threshold, a spike is initiated and propagates down the axon and the membrane potential is reset. An important characteristic of the IF cell is that it is a rectifying element—when its activity is encoded as a rate, it is not capable of expressing negative quantities.

The combination of the routing circuit and the IF transceiver comprises a module that can be connected both in parallel and in series to create large-scale, multi-layer neural systems. The connectivity of the modules can easily be reconfigured by altering the contents of the look-up table. Synaptic plasticity can be implemented on the fly by altering not only the transmission probabilities, but also the connection topology. This provides a reconfigurable extension to on-chip learning (Cauwenberghs & Bayoumi, 1999).

3. Modeling issues

Networks of integrate-and-fire neurons exhibit drastically different dynamical properties than networks of McCulloch–Pitts neurons or other mean-rate abstractions of information processing in neural systems (Maass & Bishop, 1999). In addition, the probabilistic nature of the synaptic weighting is a source of stochasticity in neural systems, which are known to be highly stochastic (Koch, 1999). Before describing our hardware implementation of networks of integrate-and-fire neurons with probabilistic synapses, we address issues that arise in their modeling and affect the computation that they perform.

3.1. Probabilistic synaptic weighting

The stochastic nature of the synapse is seldom used in models of neural systems. One reason for introducing probabilistic synaptic weighting in our address-event based system is one of convenience of implementation, since transmitted events carry no weight by themselves and sending multiple events to implement synaptic weighting is inefficient. We justify this choice by considering that the stochastic encoding has little impact on the dynamics of spiking neurons. Furthermore, we argue that a probabilistic, fixed-amplitude model for the synapse is no less physiologically justified than a deterministic model with variable amplitude.

3.1.1. Neurological basis

Physiological studies show that a synaptic efficacy w can be expressed as the combined effect of three physical mechanisms:

$$w = abc \tag{1}$$

where a represents the number of quantal neurotransmitter release sites, b is the probability of synaptic release per site, and c is a measure of postsynaptic effect of the synapse (Koch, 1999). Many neural network models neglect the quantal nature of synaptic transmission ($a =$ constant) and assume a presynaptic event is always transmitted ($b = 1$), thus absorbing all of the weight variations in the c variable. Our implementation instead holds c constant ($c = 1$) and varies b (the transmission probability) and possibly a (the number of synaptic iterations, M in Section 2). While the two approaches differ, it is clear that the resulting synaptic weighting quantitatively gives the same results. The equivalence is analyzed in further detail in Section 3.1.3.

3.1.2. Poisson statistics

An important consideration of the probabilistic synapse weighting is the effect is has on the statistics of the postsynaptic potential, thereby affecting the dynamics of the integrate-and-fire neuron to which it connects. We will investigate the limiting case where presynaptic spiking activity of a given mean rate is either Poisson distributed or deterministic with regular inter-spike interval (ISI).

When the input activity is encoded by spike rate and the intervals of the events are approximately Poisson distributed, probabilistic transmission does not significantly alter the statistics of the input. This can be demonstrated if we model the probabilistic transmission as a Bernoulli process, as modulating a Poisson process with rate λ by a Bernoulli process with parameter p gives a Poisson process with rate $p\lambda$ (Parzen, 1962).

If the events have a regular interval, however, the probabilistic nature of the synapses will add stochasticity to the system. To quantify this, we must first introduce a measure of a spike train's regularity. The coefficient of variation of the ISI distribution is a useful measure of the variability of a spike train (Koch, 1999). If T is a random variable that represents the ISI, the coefficient of variation is given by

$$C_V = \frac{\sqrt{\mathrm{Var}[T]}}{E[T]}. \tag{2}$$

For a periodic spike train, $C_V = 0$, and for a Poisson spike train, $C_V = 1$. If we have a periodic spike train with an ISI of τ and we use a Bernoulli process with parameter p to gate the spikes, the ISI of the resulting process is given by the geometric distribution

$$P_T(t = k\tau) = q^{k-1}p, \qquad k \geq 1, \tag{3}$$

where $q = 1 - p$ and k is an integer. This distribution has a mean of τ/p and a variance of $(\tau/p)^2 q$. For this ISI distribution,

$$C_V = \frac{\sqrt{(\tau/p)^2 q}}{\tau/p} = \sqrt{q}. \tag{4}$$

As q approaches unity, we obtain a Poisson distribution for the transmitted spike train. Notice however, that events only occur at integer multiples of the ISI τ, and so the distribution of the Bernoulli modulated spike train is never truly Poisson.

Of course there is more to a spike train than the firing rate and variance, and we must also consider the effect that the probabilistic synaptic weighting has on time-domain correlation between pairs of spike trains. For instance, phase information (Lyon & Shamma, 1996), coincidence detection (Simmons, Saillant, Ferragamo, Haresign, Dear, Fritz et al., 1996) and inter-spike interval statistics (Ghitza, 1986) are central to models of the auditory system. More generally, temporal correlations also govern mechanisms of learning and memory (Gerstner, Kempter, van Hemmen & Wagner, 1996).

3.1.3. Correlations and synaptic plasticity

Models of synaptic plasticity frequently involve, in one fashion or another, a correlation between presynaptic and postsynaptic activity. To validate the functional equivalence between probabilistic and deterministic synaptic weighting, we briefly consider the effect of probabilistic weighting on the correlation between events. The precise effect on neural dynamics is intractable because of the nonlinearity of the integrate-and-fire neural activity, but one can make simple observations by considering correlations between postsynaptic events *preceding* the neural transfer function.

Probabilistic synaptic weighting can be represented as a transmission probability on a graph. Let $\Pr(x)$ be the probability that a presynaptic spiking event ($x = 1$) occurs in a given time window, and $\Pr(y)$ be the corresponding probability of a postsynaptic event ($y = 1$) in the same window. Then

$$\Pr(y) = \Pr(\mathrm{y}|\mathrm{x})\Pr(x) = w\Pr(x) \tag{5}$$

where w represents the conditional probability of the synapse transmitting an event received. Clearly, the mean rate (or expected value) of y equals that of x scaled by the synaptic probability w. An identical mean rate would have been obtained for a deterministic synapse with amplitude weighting of same strength, $y = wx$. This equivalence is consistent with the product form of the terms b and c in the expression of w in Eq. (1).

The equivalence extends directly to correlations between events. Let $\Pr(x_1, x_2)$ be the joint probability that presynaptic events at synapses 1 and 2 occur within a given time window,[1] and $\Pr(y_1, y_2)$ the probability of postsynaptic events at synapses 1 and 2 in the same window. Then

$$\Pr(y_1, y_2) = \Pr(y_1, y_2|x_1, x_2)\Pr(x_1, x_2). \tag{6}$$

Since each synapse can be modeled as an independent stochastic process,

$$\Pr(y_1, y_2|x_1, x_2) = \Pr(y_1|x_1)\Pr(y_2|x_2) = w_1 w_2 \tag{7}$$

where $w_i = \Pr(y_i|x_i)$ represents the transmission probability of the synapse from x_i to y_i. Substituting Eq. (7) into Eq. (6) gives

$$\Pr(y_1, y_2) = w_1 w_2 \Pr(x_i, x_2). \tag{8}$$

As a consequence, the cross-correlation (or mean-rate coincident activity) of coincident postsynaptic events is that of the presynaptic events scaled by the product of the synaptic weights:

$$E(y_1 \cdot y_2) = w_1 w_2 E(x_1 \cdot x_2). \tag{9}$$

An identical expression for the cross-correlation would have been obtained using deterministic synaptic weighting with amplitudes w_1 and w_2, i.e. $y_1 = w_1 x_1$ and $y_2 = w_2 x_2$. This simple and intuitive result extends directly to correlations of higher order between multiple postsynaptic events. It does not, however, account for the effect of dynamics and saturation in the IF neuron.

3.2. Integrate-and-fire neuronal transfer function

In the ubiquitous McCulloch–Pitts model of neural computation, the neuron activation function implements rectification and saturation of an otherwise linear response to synaptic units. The rectifying and saturating dynamics of IF neurons performs a similar nonlinear activation function, but only to a first-order approximation. This is an important consideration in interpreting the results we obtain from the prototype system, as will be evident in Section 6.

[1] Identical arguments apply to the case where event 1 occurs in a given window *preceding* event 2.

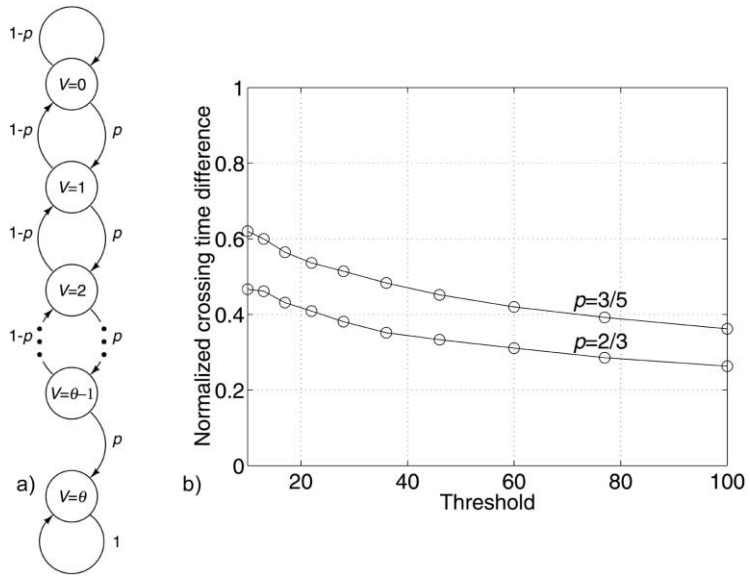


Fig. 3. (a) State-transition diagram for a Markov chain model of the potential of an IF cell. This model was used to determine the time of threshold crossing. p is the probability that an event is excitatory. (b) Comparison of the McCulloch–Pitts and Markov models for an IF cell for two values of p. The y-axis shows the difference between threshold crossing times in the McCulloch–Pitts model and the Markov model, normalized to the McCulloch–Pitts model.

3.2.1. Rectification dynamics

In a McCulloch–Pitts abstraction, a rectifying activation function could model the response of the IF cell to the integrated synaptic contributions. In this simplifying model, the output spike count is approximately proportional to the rectified difference between the excitatory spike count, K_E, and the inhibitory spike count, K_I, as given by

$$K_{out} = \begin{cases} \left\lfloor \dfrac{s(K_E - K_I)}{\theta} \right\rfloor & \text{if } K_E > K_I; \\ 0 & \text{if } K_E < K_I, \end{cases} \tag{10}$$

where s is the potential step size, θ is the threshold, and $\lfloor\cdot\rfloor$ represents the flooring operation. On average, the threshold crossing occurs after $\theta/(2p-1)$ events, where p is the probability that an incoming event is excitatory, and $1-p$ is the probability that the event is inhibitory.

In reality, the effect of the IF dynamics on the rectifying response is not as simple as Eq. (10) suggests. Because the potential is clamped to zero whenever the net input is negative, the order in which inhibitory and excitatory events arrive matters. By constructing a probabilistic model of the IF cell, we can estimate the effect of dynamics in the rectification to first order. The state of the potential of the IF cell can be modeled as a Markov chain, as depicted in Fig. 3(a). By iterating the state-transition matrix of the Markov model, we can empirically determine the probability distribution of the potential state. In the Markov model, the positive bias induced by the rectification will cause the threshold crossing on average to occur earlier than in the McCulloch–Pitts model.[2] Fig. 3(b) shows a plot of the difference between the threshold crossing times in the McCulloch–Pitts model and the Markov model, normalized to the McCulloch–Pitts model threshold crossing time. The difference between the two decreases as the threshold increases, as the most likely state of the probability distribution has more time to move away from the zero state ($V=0$) where inhibitory spikes can be lost. The difference is less pronounced when the ratio of excitatory to inhibitory events increases, as this too shifts the probability mass away from the zero state.

3.2.2. Saturation dynamics

Real neurons have a maximum firing frequency which is set by their refractory period. The refractory period is due to the slow hyperpolarization of the membrane following an action potential. Saturation is also inherent in the hardware implementation discussed in Section 5, but mainly with a different physical origin. Although there is a refractory period associated with resetting the potential after a spike, the maximum firing frequency is determined for the most post by the global activity of the IF transceiver. It is characteristic of AER systems that the bandwidth of the channel is dynamically allocated to the active cells, and the speed of operation is a function of the number of cells that are simultaneously active (Apsel & Andreou, 2001).

4. Application: boundary contour system

We can use the address domain computation scheme to implement large multi-chip VLSI neural networks. One example of a network architecture that lends itself to

[2] We take the threshold crossing time as the earliest time in which the threshold state ($V=\theta$) is the most likely state.

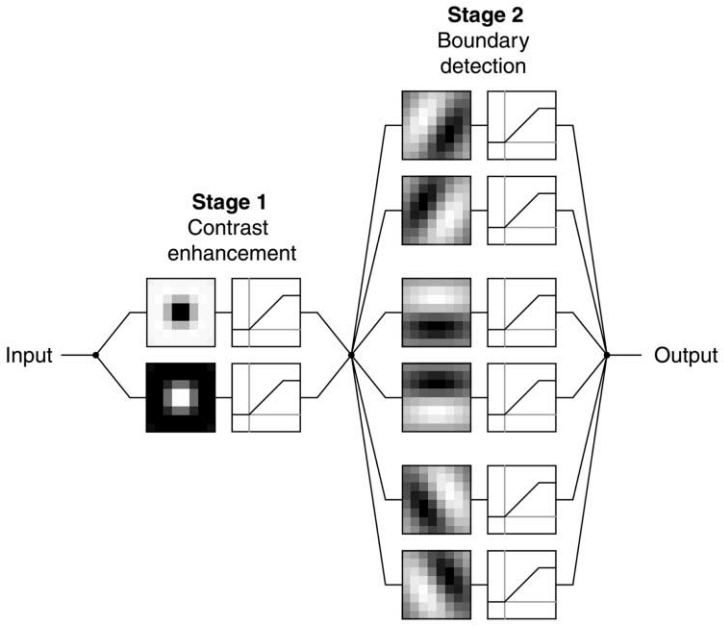


Fig. 4. Example mapping of early stages in the BCS architecture onto a table-based transceiver architecture. The patterns represent 8×8 look-up table filter kernels and the transfer function symbol represents an integrate-and-fire transceiver. Stage 1 performs normalization and contrast enhancement. Stage 2 extracts boundaries of different orientations in parallel, and then combines them.

efficient implementation by our scheme is the boundary contour system (BCS) for image segmentation and boundary completion in the presence of clutter and occlusion (Grossberg & Mingolla, 1985; Grossberg, Mingolla & Williamson, 1995; Mingolla, Ross & Grossberg, 1999).

There are several different approaches to implementing this system in VLSI. Our approach is one in which the connectivity is fully programmable, and long-range connections can be implemented as efficiently as short-range connections. This has certain advantages over a scheme in which all connectivity is hardwired, which provides higher computational bandwidth and focal-plane operation but at the expense of significant overhead in wiring resources implementing nearest-neighbor and diffusive spatial kernels (Cauwenberghs & Waskiewicz, 1999). It is also possible to combine hardwired short-range cellular interconnects on-chip with reconfigurable long-range interconnects using AER transduction (Serrano-Gotarredona, Andreou & Linares-Barranco, 1999).

Fig. 4 illustrates one possible scheme of mapping the early stages of the BCS onto a system of look-up tables and IF transceivers. The patterns are 8×8 look-up table filter kernels, which are followed by a symbol that represents the transfer function of transceiver array. The symbol is intended to evoke the rectifying and saturating characteristics of the IF cell. In the first stage, the image is normalized and contrast is enhanced by the summing of the output of ON-center OFF-surround cells and OFF-center ON-surround cells. In the second stage, multiple orientations are extracted in parallel and these parallel outputs are combined to extract the boundaries of the input image.

To illustrate how the BCS architecture can be realized with neural filters, Fig. 5 shows the simulated output of the system in Fig. 4, using rectified convolutions to model the look-up tables and transceivers.

Next, we describe the system we developed in VLSI hardware to validate the general architecture and experimental results from a fabricated prototype.

5. Implementation

To demonstrate these ideas, we implemented and tested a prototype system. The system consists of a printed circuit board with a full custom integrated circuit 32×32-cell address-event integrate-and-fire transceiver, a $128\text{ k} \times 16$ RAM for storage of the routing table and synaptic weights, and a microcontroller which probabilistically gates the transmission of AEs and handles the handshaking between the transceiver and the outside world.

5.1. Address-event integrate-and-fire transceiver

The address-event IF transceiver was designed on a $1.5 \times 1.5\ \text{mm}^2$ die in a $0.5\ \mu\text{m}$ process ($\lambda = 0.3\ \mu\text{m}$). A photograph of the chip die is shown in Fig. 6. The IF transceiver is so named because it receives AEs as input, integrates them, and transmits AEs as output. Incoming AEs are decoded and directed to one of the 1024 randomly accessible cells. An output address encoding system independently services spiking event requests in the array and sends outgoing AEs.

Each transceiver cell is a VLSI IF neuron with all of the

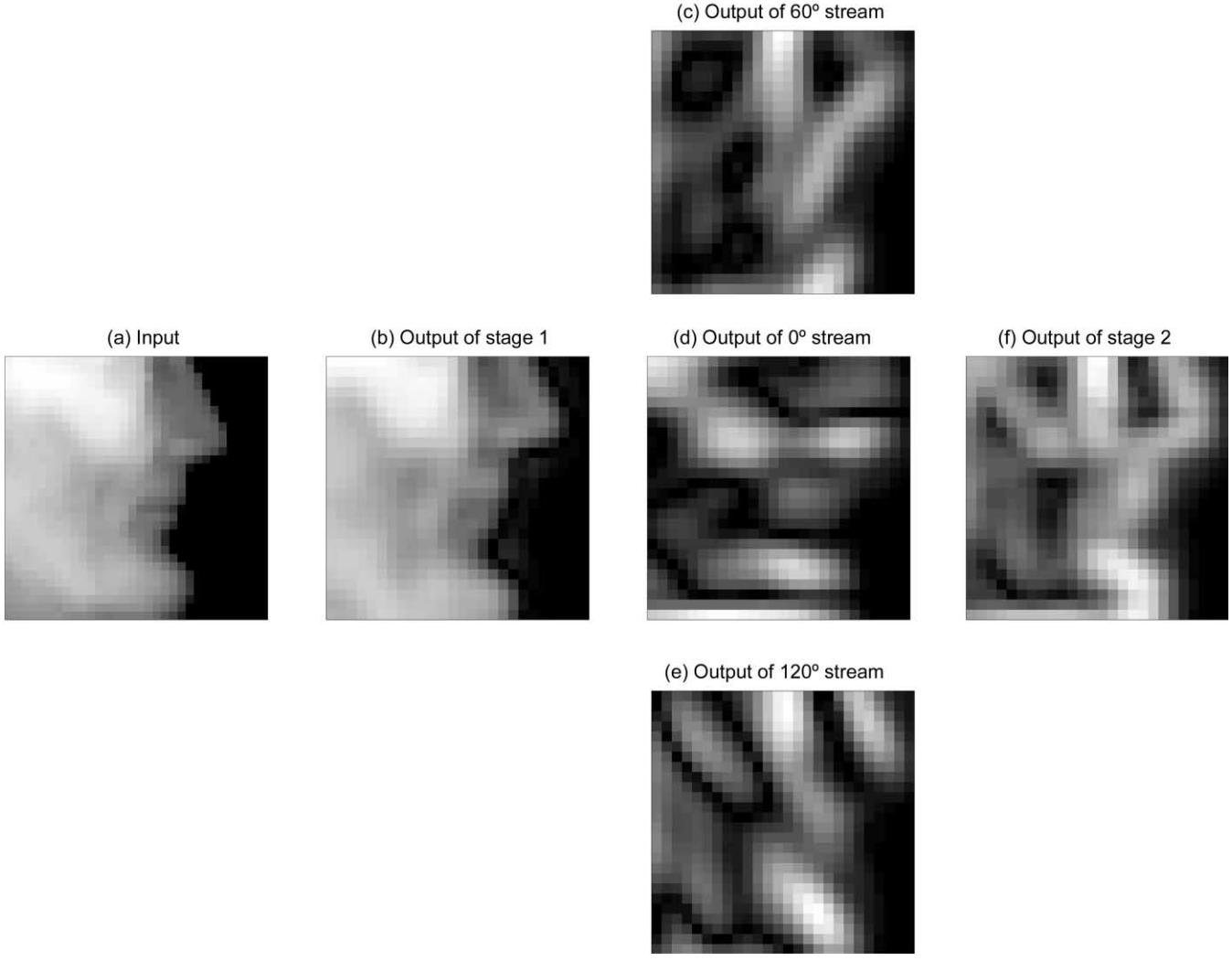


Fig. 5. Results of a simulation where the early stages of the BCS are mapped to neural filter kernels, implemented as convolutions followed by rectifications. (a) Input. (b) Output of Stage 1. (c)–(e) Output of parallel orientation streams. (f) Output of Stage 2.

properties of IF neurons that we have addressed in Section 3. Fig. 7 shows a circuit simulation of an IF cell that receives a repeating sequence of each three excitatory AEs followed by one inhibitory AE. A charge pump circuit (Cauwenberghs & Yariv, 1994) removes charge from or deposits charge on a capacitor which stores the potential, as shown in the $-I_{\mathrm{STORE}}$ trace of Fig. 7. The potential is represented as the active low voltage V_{STORE} on the storage capacitor: zero potential is $V_{\mathrm{DD}} = 5$ V; excitatory events bring V_{STORE} towards GND (zero) and inhibitory events bring V_{STORE} towards V_{DD}. The polarity of the V_{STORE} trace in Fig. 7 is inverted so as to evoke a more conventional snapshot of the potential in an IF neuron. V_{STORE} is gradually 'incremented' towards the threshold, which is 1.24 V. When V_{STORE} reaches the threshold, positive feedback drives it to GND, which activates a request REQ. The output encoding system on the periphery of the IF array transmits the event off the chip and activates the acknowledgement ACK, which resets V_{STORE} to V_{DD} and REQ to GND.

5.1.1. Transceiver operation

A schematic of a 4×4 version of the transceiver is shown in Fig. 8. Each IF cell is randomly accessible. A monostable circuit controls the timing of the input request/acknowledge cycle (INACK/INREQ). The lower half of the incoming address is decoded into column selection information, which is combined with $\overline{\mathrm{POL}}$ and the monostable pulse to generate the $\overline{\mathrm{CPOL}}$ signals which go to all of the columns in the array. The upper half of the address is decoded into row selection information to generate the RSEL signals. For the encoding of the output, a ring of flip-flops scans the row for a request ($\overline{\mathrm{RREQ}}$). When a row request is found, a similar ring scans the columns for a column request ($\overline{\mathrm{CREQ}}$). The halted row scanning pulse (RSCAN) is fed back into the array so that the column scanner only scans cells in the active row. When both row and column requests are found, the transceiver sends an output request.

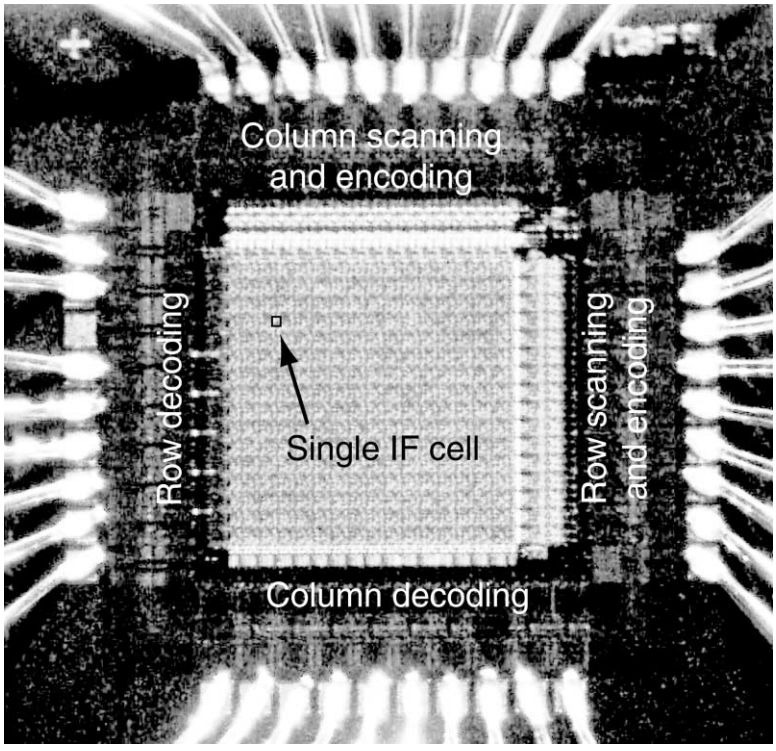


Fig. 6. Photograph of the address-event integrate-and-fire transceiver chip die.

5.1.2. Integrate-and-fire circuit

A schematic of the VLSI implementation of the IF cell is shown in Fig. 9. It contains 14 transistors and takes up an area of $68 \times 68\ \lambda^2$. The cell has an ~88 fF storage capacitor which holds the potential V_{STORE}.

Transistors M1–M4 serve to select the cell and increment or decrement the potential accordingly. When RSEL and $\overline{\mathrm{RSEL}}$ are activated (row selection), the value of $\overline{\mathrm{CPOL}}$ (column selection) is passed to V_{CP}. M3 and M4 comprise a charge pump that injects charge on or removes charge from V_{STORE}. V_{BP} and V_{BN} bias M3 and M4 in the subthreshold regime. If $\overline{\mathrm{CPOL}} = \mathrm{GND}$, M4 is on, charge is removed from the capacitor, incrementing the potential. If $\overline{\mathrm{CPOL}} = V_{\mathrm{DD}}$, M3 is on, charge is injected on the capacitor, decrementing the potential. If $\overline{\mathrm{CPOL}} = V_{\mathrm{DD}}/2$ (the column is not selected), then the potential is unchanged. The switch injection-free operation of the charge pump allows increments and decrements as small as 50 μV (Cauwenberghs, 1997). The table in Fig. 9 summarizes the control of the cell selection and charge pump.

M5, M6 and M11 (the drain of which is normally connected to V_{STORE}) comprise a simple latching comparator. When V_{STORE} approaches the threshold set by V_{THRESH} and V_{BIAS}, V_{COMP} is pulled high, turning on M11, which pulls down V_{STORE}. This positive feedback forces V_{STORE} to GND. V_{COMP} drives the gates of M12 and M14, which form the pulldown of wired-NOR gates for column and row, respectively. The output of the wired-NORs, $\overline{\mathrm{CREQ}}$ and $\overline{\mathrm{RREQ}}$, represent the column and row output requests of the IF cell. $\overline{\mathrm{CREQ}}$ can only become active when the row scanner selects the row (RSCAN is activated). M7–M10 can be thought of as a CMOS NOR where GND is gated by M11. The activation of $\overline{\mathrm{RACK}}$ and $\overline{\mathrm{CACK}}$ resets V_{STORE} to zero potential, at voltage V_{DD}.

5.2. Address-event routing

The IF transceiver operates in conjunction with a RAM and a microcontroller, which implement the network topology and the probabilistic synaptic computation. In our experimental setup, all of the elements were placed on a printed circuit board and interfaced with a PC. Fig. 10 shows the board in three different configurations. Fig. 10(a) shows the programming mode, where the network is configured or reconfigured. RAM address and data are supplied by the PC while the microcontroller scrolls through the synapse indices.

In feedforward mode (Fig. 10(b)), incoming AEs are sent from the PC to the RAM, and the microcontroller scrolls through all of the synapses projecting from the input address. For each incoming AE, the microcontroller generates a random number which is compared to the synaptic weight magnitudes. If a weight magnitude is larger than the random number, the event is projected to the transceiver address that corresponds to the synapse. Output AEs are sent from the IF transceiver to the PC where they are recorded.

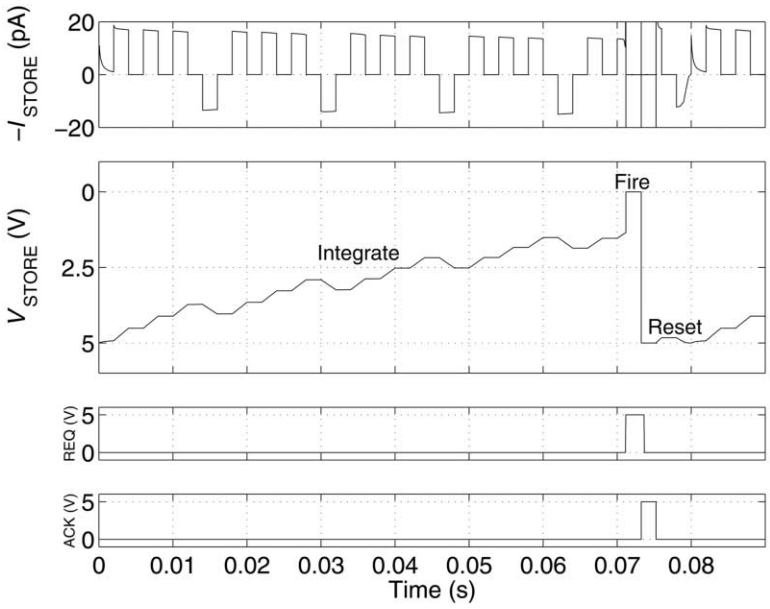


Fig. 7. Example waveforms in the VLSI integrate-and-fire cell illustrating an integrate-and-fire cycle. The waveforms were obtained from a Spectre® simulation of the VLSI integrate-and-fire cell. The input to the cell was a repeating sequence of three excitatory AEs followed by an inhibitory AE, shown in terms of the current I_{STORE} flowing onto the storage capacitor. See text for further details.

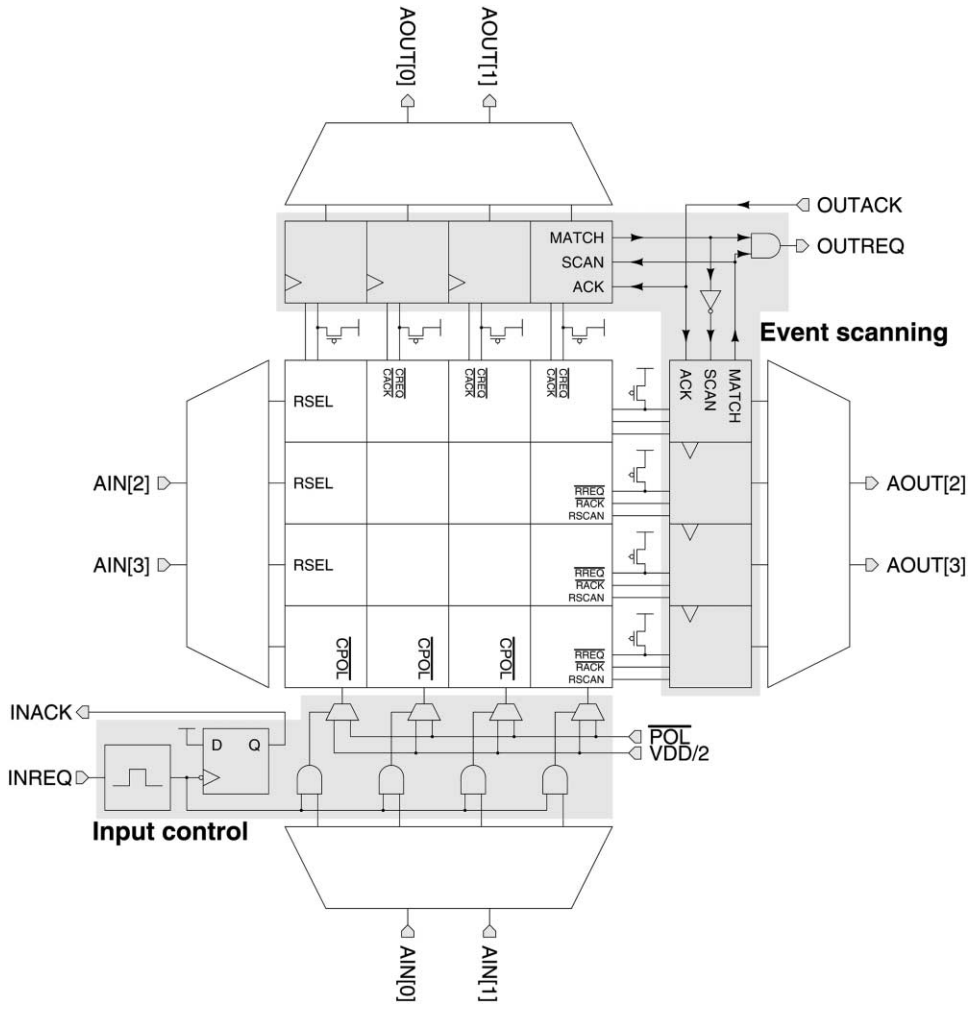
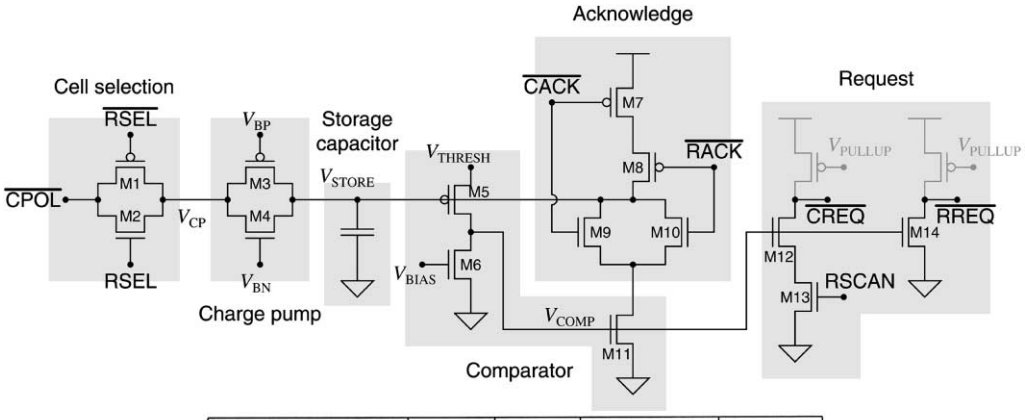


Fig. 8. Schematic diagram of a 4 × 4 cell version of the address-event integrate-and-fire transceiver. See text for details.



Operation	RSEL	$\overline{\text{RSEL}}$	$\overline{\text{CPOL}}$	V_{CP}
Row inactive	0	1	Don't care	High-Z
Column inactive	1	0	$V_{DD}/2$	$V_{DD}/2$
Increment	1	0	0	0
Decrement	1	0	1	1

Fig. 9. VLSI integrate-and-fire cell. The cell selection block takes input from the periphery and converts it to input for the charge pump (see table in inset). The charge pump removes charge from or deposits charge on the storage capacitor. The comparator detects whether the potential exceeds the threshold. Once it does, the comparator latches and the column and row request signals are activated. After the event is communicated off the chip, the column and row acknowledge signals reset the potential on the storage capacitor.

The system can also operate in a recurrent mode (Fig. 10(c)), where output AEs are routed from the transceiver to the RAM. The RAM then projects events back to the transceiver, as before. The feedforward and recurrent modes can be combined to create networks that have both hidden units and output units.

6. Experimental results

As a proof of concept, we examined an image filtering problem (Fig. 11). We used an image from a Matlab® demo as our test image (Fig. 11(a)), and a one-dimensional Laplacian that enhances vertical edges as our filter ([1 −2 1]). First, we performed simply a convolution followed by a rectification (Fig. 11(b)). Then, we performed the filtering in the address-domain with our VLSI system (Fig. 11(c)). The number of times an event was sent from a pixel in the input image was proportional to the pixel's intensity. A total of ~1,160,000 events were sent, corresponding to 2550 for the brightest input pixel. V_{THRESH} and V_{BIAS} were set to 2.5 and 0.8 V, giving a firing threshold of 1.24 V (zero potential at 5 V). The excitatory bias (V_{BN}) was set such that 40 spikes were required to reach threshold. The inhibitory bias (V_{BP}) was tuned until the experimental results matched those of the rectified convolution. At that point, inhibitory events were seven times as strong as excitatory events. We also ran a detailed Matlab simulation of the system that incorporated the probabilistic transmission of events and the rectifying properties of the IF cells (Fig. 11(d)). The threshold and excitation/inhibition ratio were set to match the experimental system.

If we consider the concepts presented in Section 3.2.1, we can see why such strong inhibition was required to match the rectified convolution results. At a threshold level of 40 events, the positive bias in the response due to rectification is significant. Therefore, we adjusted the inhibitory strength to counteract the positive bias. As shown in Fig. 3, increasing the threshold also mitigates this effect, but this requires more time in order to get a satisfactory number of spikes.

Both the experimental results (Fig. 11(c)) and the simulation results (Fig. 11(d)) display some noise as compared with the rectified convolution (Fig. 11(b)). This is primarily due to the quantization of the output intensity to ~20 levels. There is some minor additional noise in the experimental results mainly due to transistor mismatch in the charge pump cells.

The experiment ran in less than 5 min, while the simulation ran for more than 2 h. The speed of the experimental system was limited by the response of the I/O card in the PC and the 5 MHz clock speed of the microcontroller. The I/O interface is mainly for purposes of characterization and acquisition; in actual applications interfacing with silicon

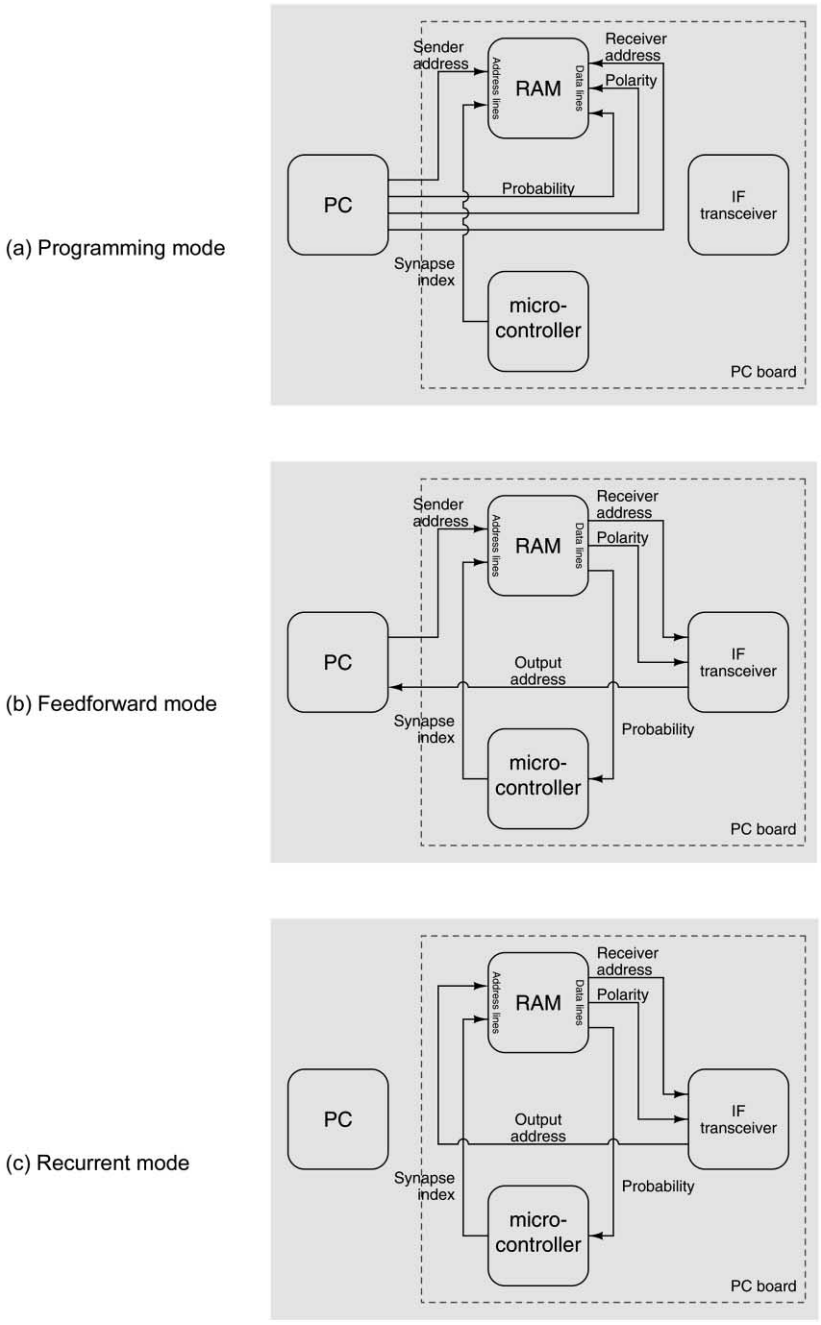


Fig. 10. Block diagrams of the prototype system. Handshaking signals between the PC and the microcontroller and the IF transceiver are omitted for clarity. (a) In programming mode, the system is configured or reconfigured by writing the contents of the look-up table from the PC to the RAM. The microcontroller scrolls through the synapse indices. (b) In feedforward mode, the PC sends and receives events. (c) In recurrent mode, the output of the IF transceiver is connected to the RAM and the PC is removed from the loop.

retinas, silicon cochleas, or other transceivers, the slow PC could be circumvented. The microcontroller could be replaced by either a field-programmable gate array (FPGA) or integrated into the transceiver for further gains in operating speed. In such a system, the transceiver chip limits the maximum available speed to about 10^8 events/s, and the system would be capable of running the same experiment in less than 0.1 s.

7. Conclusions

We have presented an architecture for performing computations in the address domain. This approach enables the implementation of massively connected networks of integrate-and-fire neurons in VLSI. We have employed probabilistic synaptic weighting and memory-based look-up tables to implement reconfigur-

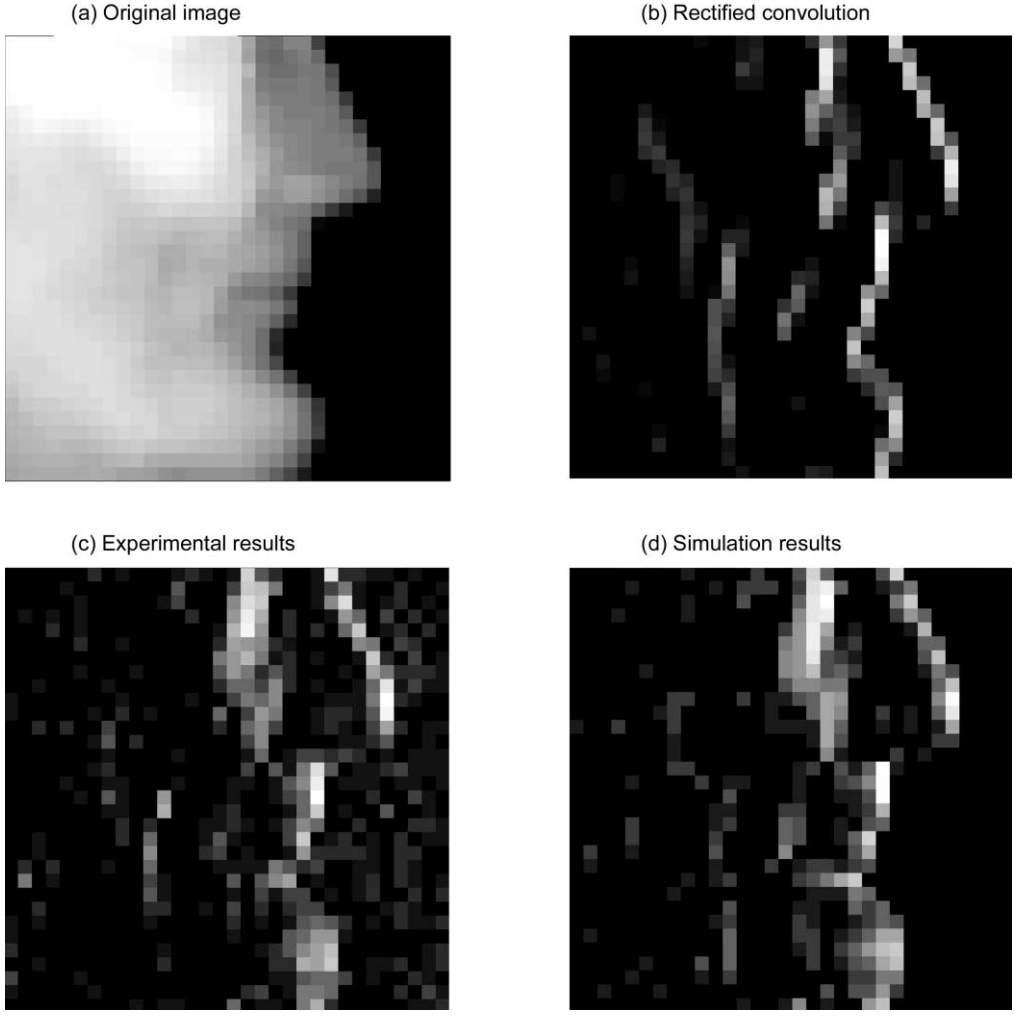


Fig. 11. Filtering in the address domain with a rectified Laplacian that enhances vertical edges. (a) Input image, scale = [0, 2550]. (b) Rectified convolution, arbitrary scale. (c) Experimental results from the VLSI system, scale = [0, 22]. (d) Simulation results, scale = [0, 18].

able connectivity. While the results here used one look-up table and transceiver, the architecture is scalable and is well suited to multi-chip systems. Many modules can potentially be connected in series and in parallel to implement large-scale, multi-layered neural processing systems.

Current efforts are focused on completely integrating the system, the assembly of a multi-chip BCS system, investigation of the potential for plasticity and learning in the address domain, and the utilization of coding schemes that rely on spike timing. We also hope to explore the extension of the concepts presented here to the optoelectronic domain (Apsel, Kalayjain, Andreou, Simonis, Chang, Datta et al., 2000).

Acknowledgements

This work was supported by DARPA/ONR MURI N00014-95-1-409 and a DARPA/ONR contract N00014-00-C-0315. The authors thank Pamela Abshire, Marc H. Cohen, and the attendees of the 2000 Telluride Workshop on Neuromorphic Engineering for helpful discussions, and George Coles for assistance with the photomicrograph. Integrated circuit fabrication was provided by MOSIS.

References

Abshire, P., & Andreou, A. G. (2000). Relating information capacity to a biophysical model for blowfly photoreceptors. *Neurocomputing*, *32–33*, 9–16.

Aiello, L. G., & Wheeler, P. (1995). The expensive-tissue hypothesis: the brain and the digestive system in human and primate evolution. *Current Anthropology*, *36* (2), 199–221.

Apsel, A., & Andreou, A. (2001). An analysis of data reconstruction efficiency using stochastic encoding and an integrating receiver. *IEEE Trans. Circuits and Systems II—Analog and Digital Signal Processing* (submitted).

Apsel, A., Kalayjian, Z., Andreou, A., Simonis, G., Chang, W., Datta, M., & Koley, B. (2000). Edge orientation enhancement using optoelectronic VLSI and asynchronous pulse coding. In *2000 IEEE International*

Symposium on Circuits and Systems: Emerging Technologies for the 21st Century (vol. 2, pp. 297–300). Lausanne, Switzerland: Presses Polytech. Univ. Romandes.

Boahen, K. (2000). Point-to-point connectivity between neuromorphic chips using address events. *IEEE Trans. Circuits and Systems—II: Analog and Digital Signal Processing*, *47* (5), 416–434.

Cauwenberghs, G. (1997). Analog VLSI stochastic perturbative learning architectures. *Analog Integrated Circuits and Signal Processing*, *13*, 195–209.

Cauwenberghs, G., & Bayoumi, M. (1999). *Learning on silicon*, Norwell, MA: Kluwer Academic.

Cauwenberghs, G., & Waskiewicz, J. (1999). Focal-plane analog VLSI cellular implementation of the boundary contour system. *IEEE Trans. Circuits and Systems—I: Fundamental Theory and Applications*, *46* (2), 327–334.

Cauwenberghs, G., & Yariv, A. (1994). Fault-tolerant dynamic multilevel storage in analog VLSI. *IEEE Trans. Circuits and Systems II—Analog and Digital Signal Processing*, *41*, 827–829.

Deiss, S. R., Douglas, R. J., & Whatley, A. M. (1999). A pulse-coded communications infrastructure for neuromorphic systems. In W. Maass & C. M. Bishop, *Pulsed neural networks* (pp. 157–178). Cambridge, MA: MIT Press.

Gerstner, W., Kempter, R., van Hemmen, J. L., & Wagner, H. (1996). A neuronal learning rule for sub-millisecond temporal coding. *Nature*, *383*, 76–78.

Ghitza, O. (1986). Auditory nerve representation as a front-end for speech recognition in a noisy environment. *Computer Speech and Language*, *1* (2), 109–130.

Grossberg, S., & Mingolla, E. (1985). Neural dynamics of perceptual grouping: textures, boundaries, and emergent segmentation. *Perception & Psychophysics*, *38* (2), 141–171.

Grossberg, S., Carpenter, G., Schwartz, E., Mingolla, E., Bullock, D., Gaudiano, P., Andreou, A., Cauwenberghs, G., & Hubbard, A. (1997). Automated vision and sensing systems at Boston University. In T. M. Strat, *Proceedings 1997 Image Understanding Workshop* (pp. 1491–1531), vol. 2. San Francisco: Morgan Kaufmann.

Grossberg, S., Mingolla, E., & Williamson, J. (1995). Synthetic aperture radar processing by a multiple scale neural system for boundary and surface representation. *Neural Networks*, *8*, 1005–1028.

Higgins, C. M., & Koch, C. (1999). Multi-chip neuromorphic motion processing. In D. Wills & S. DeWeerth, *Proceedings 20th Anniversary Conference on Advanced Research in VLSI* (pp. 309–323). Los Alamitos, CA: IEEE Computer Society.

Koch, C. (1999). *Biophysics of computation: information processing in single neurons*, New York: Oxford University Press.

Lazzaro, J., Wawrzynek, J., Mahowald, M., Sivilotti, M., & Gillespie, D. (1993). Silicon auditory processors as computer peripherals. *IEEE Trans. Neural Networks*, *4* (3), 523–528.

Levy, W. B., & Baxter, R. A. (1995). Energy efficient neural codes. *Neural Computation*, *8*, 531–543.

Lyon, R., & Shamma, S. (1996). Auditory representations of timbre and pitch. In H. L. Hawkins, T. A. McMullen, A. N. Popper & R. R. Fay, *Auditory computation* (pp. 221–270). New York: Springer.

Maass, W. & Bishop, C. M. (1999). Pulsed neural networks Cambridge, MA: MIT Press.

Mahowald, M. (1994). *An analog VLSI system for stereoscopic vision*, Boston: Kluwer Academic.

Mead, C. (1990). Neuromorphic electronic systems. *Proc. IEEE*, *78*, 1629–1636.

Mingolla, E., Ross, W., & Grossberg, S. (1999). A neural network for enhancing boundaries and surfaces in synthetic aperture radar images. *Neural Networks*, *12*, 499–511.

Parzen, E. (1962). *Stochastic processes*, San Francisco: Holden-Day.

Rieke, F., Warland, D., de Ruyter van Steveninck, R., & Bialek, W. (1997). *Spikes: exploring the neural code*, Cambridge, MA: MIT Press.

Serrano-Gotarredona, T., Andreou, A., & Linares-Barranco, R. (1999). AER image filtering architecture for vision-processing systems. *IEEE Trans. Circuits and Systems—I: Fundamental Theory and Applications*, *46* (9), 1064–1071.

Simmons, J. A., Saillant, P. A., Ferragamo, M. J., Haresign, T., Dear, S. P., Fritz, J., & McMullen, T. A. (1996). Auditory computations for biosonar target imaging in bats. In H. L. Hawkins, T. A. McMullen, A. N. Popper & R. R. Fay, *Auditory computation* (pp. 401–468). New York: Springer.