ANALOG VLSI SPIKING NEURAL NETWORK WITH ADDRESS DOMAIN PROBABILISTIC SYNAPSES

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

Department of Electrical and Computer Engineering
Johns Hopkins University
Baltimore, MD 21218, USA

ABSTRACT

We present an analog VLSI address-event transceiver containing an array of integrate-and-fire neurons and a scheme for implementing a reconfigurable neural network with probabilistic synapses. Neural "spikes" are transmitted through address-event representation—the address of the sending neuron is communicated through an asynchronous request and acknowledgment cycle. Continuous-valued synaptic weights are implemented by probabilistically routing address events. Results from a prototype system with 1,024 analog VLSI integrate-and-fire neurons, each with up to 128 probabilistic synapses, demonstrate these concepts in an image processing application.

1. INTRODUCTION

The brain's remarkable ability to process information in a parallel and distributed manner is enabled by its massively connected architecture. Unfortunately, the extensive connectivity of the brain is impossible to directly implement in VLSI due to the limitations of connectivity within and between microchips. There is another characteristic of neural systems, however, that enables us to overcome this problem. Neurons represent their activity as action potentials or "spikes"—continuous-time, discrete-value signals. We can take advantage of the temporally sparse nature of spike coding and the high bandwidth of VLSI systems to overcome the connectivity problem by time-multiplexing signals from many connections on the same data bus.

Address-event representation (AER) is a communication protocol that uses time-multiplexing to emulate extensive connectivity [1] (Fig. 1). We have an array of cells that encode their activity in the form of spikes (the sender) and we want to transmit these activities to another array of cells (the receiver). The "brute force" approach would be to use one wire for each pair of cells, requiring N wires for 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 a shared 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 required number of wires from N to $\sim \log_2 N$. 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).

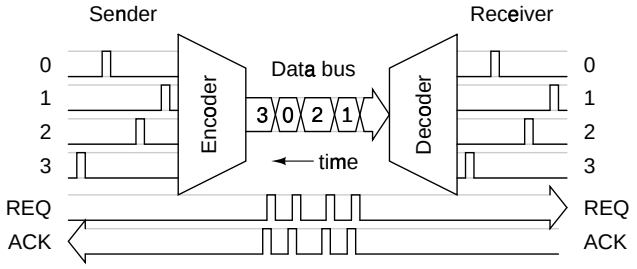


Figure 1: Address-event representation. Sender events are encoded into an address, sent over the bus, and decoded. Handshaking signals REQ and ACK are required to ensure that only one cell pair is communicating at a time. Note that the time axis goes from right to left.

AER implements a one-to-one connection topology, which is appropriate for emulating the optic and auditory nerves [1, 2]. To implement more complex neural circuits, convergent and divergent connections are required. Several authors have discussed and implemented methods of enhancing the connectivity of AER systems to this end [3, 4, 5, 6]. 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. After introducing the scheme, we explore some theoretical issues that arise in integrate-and-fire neural networks with probabilistic synapses. Finally, we describe a hardware realization of the scheme and report results from the prototype system for an image processing task.

2. ADDRESS DOMAIN COMPUTATION

We augment the traditional AER system to create a scalable, reconfigurable architecture that is sufficient for implementing a wide range of network topologies. We map a two-layer neural network to the AER framework by means of a look-up table circuit (Fig. 2). Each row in the table corresponds to a single synaptic

This work was supported by DARPA/ONR MURI N00014-95-1-409 and a DARPA/ONR contract (acoustic MEMS). The authors thank Pamela Abshire, Marc H. Cohen, and the attendees of the 2000 Telluride Workshop on Neuromorphic Engineering for helpful discussions. Microchip fabrication was provided by MOSIS.

connection—it contains information about the sender location, the receiver location, and the weight of the connection (polarity and magnitude). A sender cell can transmit to multiple receiver cells, enabling convergent and divergent connections. A circuit interprets the weight of a connection as the *probability* that the AE will be transmitted from the sender cell to the receiver cell.

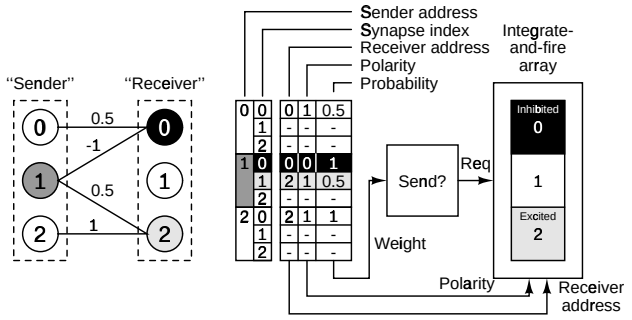


Figure 2: Mapping of a two-layer network into an AE look-up table with transmission probabilities. In this example, sender 1 sends an AE. An inhibitory AE is transmitted to receiver 0 with 100% probability, and then an excitatory AE is transmitted to receiver 2 with 50% probability. In the implementation, 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.

The receiver consists of cells that integrate AEs from multiple locations. Each cell has a potential that changes with activity. An excitatory (inhibitory) AE causes the potential to be incremented (decremented). The potential is initialized to zero and cannot go below zero; when the potential exceeds a threshold, the cell sends an AE as output and the potential is reset to zero. For this reason, the cells are called *integrate-and-fire* (IF) cells. Because the IF array both *trans*mits and re*ceives* AEs, we call the array an IF transceiver.

The combination of the look-up table circuit and the IF transceiver comprises a module that can be connected both in series and in parallel to create large-scale neural systems. The connectivity of the modules can be reconfigured by altering the contents of the look-up tables. Synaptic plasticity can be implemented on the fly by altering not only the transmission probabilities, but also the connection topology.

3. THEORETICAL ISSUES

Integrate-and-fire neural networks with probabilistic synapses exhibit drastically different dynamical properties than neural networks that encode their synaptic weights and activities with continuous variables (e.g. a McCulloch-Pitts neural network or other mean-rate abstraction). Before describing our hardware implementation of an integrate-and-fire neural network, we must address these issues.

3.1. Statistics of probabilistic transmission

Probabilistic synapses have interesting statistical properties. When the presynaptic spike events are Poisson distributed, the probabilistic transmission does not alter the statistics of the activity. More formally, if the presynaptic activity is Poisson distributed and encoded by event rate λ, and the probabilistic transmission is modeled as a Bernoulli process with parameter w, the postsynaptic events will be Poisson with rate $w\lambda$ [7, pp. 47-8]. In other words, the rate will be reduced, but the regularity will be unchanged. If the presynaptic events have a regular interval, however, the probabilistic nature of the synapses will make the activity less regular, adding stochasticity to the system.

3.2. Rectification dynamics in integrate-and-fire cells

In a McCulloch-Pitts (MP) neural network, a rectifying activation function can model the response of the IF cell to the integrated synaptic contributions. In this simplifying model, the output spike count, K_{out}, is approximately proportional to the rectified difference between the excitatory spike count K_{E}, and the inhibitory spike count K_{I}, as given by

$$K_{\text{out}} = \begin{cases} \left\lfloor \frac{s(K_{\text{E}} - K_{\text{I}})}{\theta} \right\rfloor & \text{if } K_{\text{E}} > K_{\text{I}} \\ 0 & \text{if } K_{\text{E}} < K_{\text{I}} \end{cases} \quad (1)$$

where s is the potential step size, θ is the threshold, and $\lfloor \cdot \rfloor$ represents the flooring operation. Eq. 1 tells us that the threshold crossing will occur at the $\theta/(2p-1)$ event, where p is the probability of an excitatory event, and $1-p$ is the probability of an inhibitory event.

In reality, the effect of the IF dynamics on the rectifying response is not as simple as Eq. 1 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 MP model[1]. Fig. 3(b) shows a plot of the difference between the threshold crossing times in the MP model and the Markov model, normalized to the MP 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.

4. IMPLEMENTATION

To demonstrate these ideas, we implemented and tested a prototype system. The system consists of a 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.

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

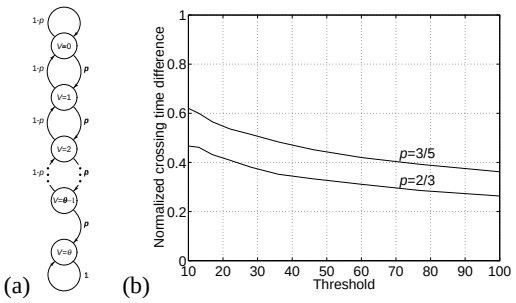


Figure 3: (a) State-transition diagram for a Markov chain model of the potential of an IF cell. This model was used to determine the time of threshold crossing. p is the probability that an event is excitatory. (b) Comparison of McCulloch Pitts (MP) and Markov models for an IF cell for two values of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

4.1. Address-event integrate-and-fire transceiver

The address-event IF transceiver was designed on a 1.5×1.5 mm^2 die in a 0.5 μm, 5 Volt process ($\lambda = 0.3$ μm). The IF transceiver receives AEs as input, integrates them, and transmits AEs as output. Incoming AEs are decoded and directed to one of the 1,024 randomly accessible cells. An output address encoding system independently services spiking event requests in the array by scanning rows and then columns for events, and then sending outgoing AEs.

A schematic of the VLSI IF cell is shown in Figure 4(a). It contains 14 transistors and takes up an area of $68 \times 68\ \lambda^2$. The cell has an $\sim$88 fF storage capacitor which holds the potential V_{STORE}. V_{STORE} is initialized to V_{DD}, which corresponds to zero stored voltage. Excitatory (inhibitory) events bring V_{STORE} towards GND (V_{DD}).

Transistors M1-M4 serve to select the cell and increment or decrement the potential accordingly (Fig. 4(b)). 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 $\overline{\mathrm{CPOL}} = \mathrm{GND}$ ($\overline{\mathrm{CPOL}} = V_{\mathrm{DD}}$), M4 (M3) is on and current flows from (to) the capacitor, incrementing (decrementing) the potential. If $\overline{\mathrm{CPOL}} = V_{\mathrm{DD}}/2$, the potential is unchanged. The switch injection-free operation of the charge pump allows increments and decrements as small as 50 μV [8].

M5, M6, and M7 (the drain of which is normally connected to V_{STORE}) comprise a latching comparator. When V_{STORE} approaches the threshold set by V_{THRESH} and V_{BIAS}, V_{COMP} is pulled high, turning on M7, which pulls down V_{STORE}. This positive feedback forces V_{STORE} to GND.

V_{COMP} drives the gates of M12 and M13, which form the pull-down of wired-NOR gates for column and row, respectively. When V_{COMP} goes high, $\overline{\mathrm{RREQ}}$ is activated. When the row scanner finds the active $\overline{\mathrm{RREQ}}$ signal, it activates RSCAN, which in turn activates $\overline{\mathrm{CREQ}}$. When the column scanner finds the active $\overline{\mathrm{CREQ}}$, an AE is sent off the chip. The resetting of V_{STORE} is controlled by M8-M11, which comprise a CMOS NOR gate where GND is gated by M7. When the outgoing AE is acknowledged, the activation of $\overline{\mathrm{RACK}}$ and $\overline{\mathrm{CACK}}$ resets V_{STORE} to zero potential, at voltage V_{DD}.

4.2. Address-event routing

The IF transceiver operates in conjunction with a RAM, which stores the look up table, and a microcontroller, which probabilistically gates the transmission of AEs. In our experimental setup, all of the elements were placed on a printed circuit board and interfaced with a PC.

The board is capable of operating in several modes. In programming mode, the contents of the look-up table are loaded into the RAM and the network topology is configured or reconfigured. The sender address, receiver addresses, synaptic weights and polarities are supplied by the PC while the microcontroller scrolls through the synapse indices (refer to Fig. 2).

In feedforward mode, incoming AEs are sent from the PC to the RAM, and the microcontroller scrolls through all of the synapses projecting from the sender 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.

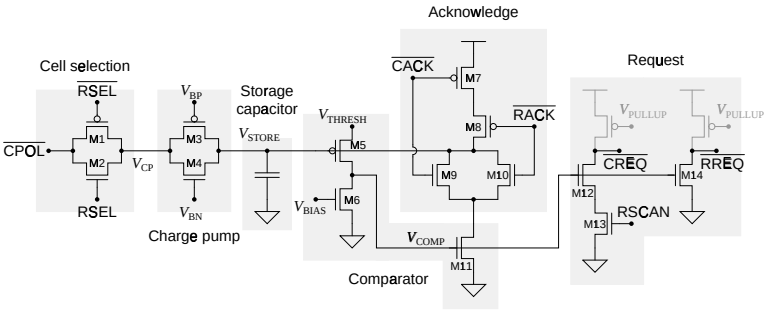
The system can also operate in a recurrent mode, where output AEs are routed from the transceiver to the RAM. The RAM then projects events back to the transceiver, as before. The feedforward and recurrent modes can be combined to create networks that have both hidden units and output units.

5. RESULTS AND DISCUSSION

As a proof of concept, we examined an image filtering problem (Fig. 5). We used an image from a Matlab(R) demo as our test image (Fig. 5(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 using Matlab (Fig. 5(b)), implementing the model described in Eq. 1. Then, we performed the filtering in the address-domain with our VLSI system (Fig. 5(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 $\sim 1,160,000$ events were sent, corresponding to $2,550$ for the brightest input pixel. V_{THRESH} and V_{BIAS} were set to 2.5 V and 0.8 V, giving a firing threshold of 1.24 V. The excitatory bias (V_{BN}) was such that 40 spikes were required to reach threshold. The inhibitory bias (V_{BP}) was tuned until the experimental results matched that those of the rectified convolution. At that point, inhibitory events were 7 times as strong as excitatory events. We ran a detailed Matlab simulation of the system that incorporated the probabilistic transmission of events and the rectifying properties of the IF cells (Fig. 5(d)). The threshold and excitation/inhibition ratio were set to match the experimental system.

If we consider the concepts presented in Sec. 3.2, 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. 5(c)) and the simulation results (Fig. 5(d)) display some noise as compared to the rectified convolution (Fig. 5(b)). This is primarily due to the quantization of the output intensity to $\sim$20 levels. There is some minor additional



(a)

(b)

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

Figure 4: VLSI integrate-and-fire cell. (a) Circuit schematic. See text for details. (b) Truth table for the cell selection circuit.

noise in the experimental results mainly due to transistor mismatch in the charge pump cells.

The experiment ran in less than 5 minutes, while the simulation ran for more than 2 hours. 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 retinas, silicon cochleas, or other transceivers, the slow PC can be circumvented. The microcontroller can be replaced by either an FPGA or integrated into the transceiver for further gains in operating speed.

6. CONCLUSIONS

We have demonstrated that AER can facilitate *computation* in addition to communication. 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 reconfigurable 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-module system, investigation of the potential for plasticity and learning in the address domain, and the utilization of coding schemes that rely on spike timing.

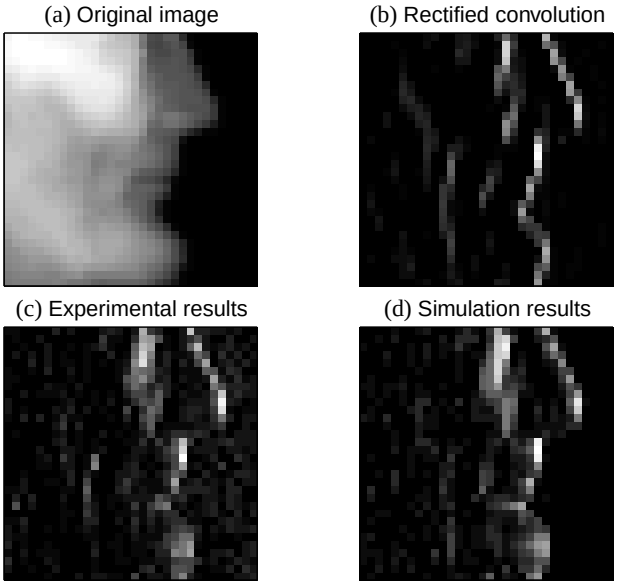


Figure 5: Filtering in the address domain to enhances vertical edges. (a) Input image, scale $= [0, 2550]$. (b) Rectified convolution, arbitrary scale. (c) Experimental results from the VLSI system, scale $= [0, 22]$. (d) Simulation results, scale $= [0, 18]$.

7. REFERENCES

[1] M. Mahowald, *An analog VLSI system for stereoscopic vision*. Boston: Kluwer Academic Publishers, 1994.

[2] J. Lazzaro, J. Wawrzynek, M. Mahowald, M. Sivilotti, and D. Gillespie, "Silicon auditory processors as computer peripherals," *IEEE Trans. Neural Networks*, vol. 4, no. 3, pp. 523–528, 1993.

[3] K. A. Boahen, "Point-to-point connectivity between neuromorphic chips using address events," *IEEE Trans. Circuits and Systems—II: Analog and Digital Signal Processing*, vol. 47, no. 5, pp. 416–434, 2000.

[4] S. R. Deiss, R. J. Douglas, and A. M. Whatley, "A pulse-coded communications infrastructure for neuromorphic systems," in *Pulsed Neural Networks* (W. Maas and C. M. Bishop, eds.), pp. 157–178, Cambridge, MA: MIT Press, 1999.

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

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

[7] E. Parzen, *Stochastic Processes*. San Francisco: Holden-Day, 1962.

[8] G. Cauwenberghs, "Analog VLSI stochastic perturbative learning architectures," *Analog Integrated Circuits and Signal Processing*, vol. 13, pp. 195–209, 1997.