Fully Digital AER Convolution Chip for Vision Processing

Luis Camuñas-Mesa, Antonio Acosta-Jiménez, Teresa Serrano-Gotarredona, and Bernabé Linares-Barranco

Instituto de Microelectrónica de Sevilla, IMSE-CNM, CSIC and Universidad de Sevilla. E-mail: luiscamu@imse.cnm.es

Abstract

We present a neuromorphic fully digital convolution microchip for Address Event Representation (AER) spike-based processing systems. This microchip computes 2-D convolutions with a programmable kernel in real time. It operates on a pixel array of size 32 x 32, and the kernel is programmable and can be of arbitrary shape and size up to 32 x 32 pixels. The chip receives and generates data in AER format, which is asynchronous and digital. The paper describes the architecture of the chip, the test setup, and experimental results obtained from a fabricated prototype.

1. Introduction

Conventional image processing systems operate on sequences of frames. An image is captured each frame period (which is 25-30 ms for commercial video cameras) and the processing algorithm is applied for each acquired frame. But biological brains do not work this way. In a biological vision system the information is acquired and processed continuously in time. Biological retinae send the image information to the visual cortex coded as spikes (also called events). When the activity level of a certain pixel reaches a threshold the pixel sends a spike. Very active pixels send more spikes, and spikes propagate through the processing chain as soon as they are generated without waiting for the whole image to be processed.

Consider, for example, the vision processing system shown in Fig. 1 where the image from the retina is processed by two layers of convolutions. If the system in Fig. 1 is frame-based, each layer of convolutions will have to wait until the previous layer finishes processing the whole image to start its operation. However, if information between chips is sent as asynchronous spikes, spikes are communicated and processed by the next layer as soon as they are generated.

In fact, the system in Fig. 1 resembles the architecture of biological vision systems where the image from the retina is sent to the visual cortex. The visual cortex is organized as layers of neurons. Neurons from each layer project their output to a population of neurons of the following layer, thus performing a projection field or, equivalently, a convolution operation [1].

The Address-Event-Representation (AER) protocol allows asynchronous massive interconnection of neuromorphic processing chips. The AER protocol was first proposed in 1991 in Caltech [2] to solve the problem of massive interconnectivity between populations of neurons located in different chips. Fig. 2 illustrates the interconnection of two chips through a point-to-point AER link. Assume that chip1 in Fig. 2 contains a matrix of M x N pixels that have to be interconnected with the pixels of chip2, which are also arranged as a M x N matrix of pixels. The AER protocol multiplexes all these connections through a high-speed digital bus. Each pixel in chip 1 (or sender chip) converts an input signal into a train of pulses whose frequency is proportional to its signal level. These pulses are arbitrated and the address of the sending pixel is coded on the fast digital bus asynchronously with handshaking. Chip 2 (or the receiver chip) decodes the arriving address and sends a spike to the corresponding neuron so that the original signal can be reconstructed.

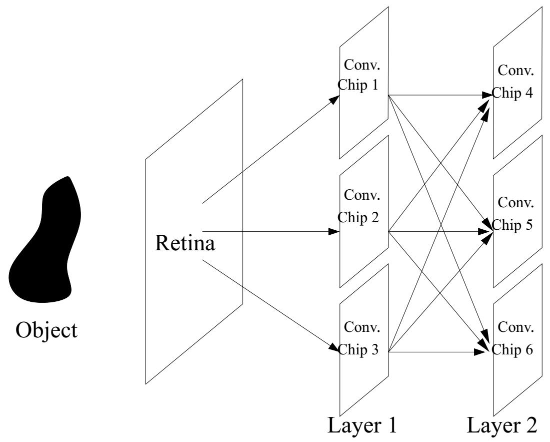

Fig. 1. Multilayer vision processing system

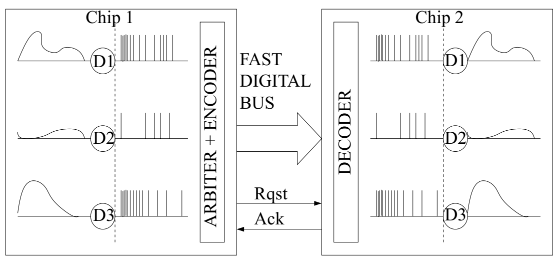

Fig. 2. Address Event Representation (AER) protocol

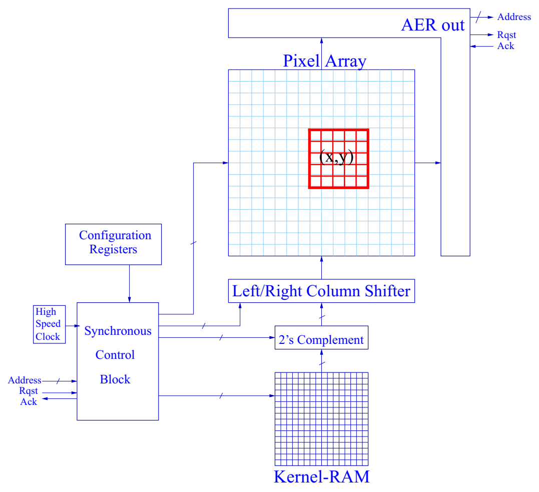

Fig. 3. Architecture of the Convolution Chip

As an additional advantage, the AER protocol allows to perform operations in the address space as addresses travel from chip to chip. For instance, by inserting a look-up table transforming the addresses in the digital bus, an image rotation or any other image transformation can be easily performed on the fly. Furthermore, if for each event sent to the receiver chip, an event is sent not only to its corresponding pixel but to all the pixels in its neighborhood, a projection field can be implemented. We additionally weight the events sent to the neighbor pixels with a stored kernel, such that the resulting image is the convolution of the input image with the stored kernel [3].

An AER convolution chip has already been published [4]. The chip described in [4] is a mixed-signal design. The peripheral circuitry for codification and decodification of the pulses is digital, as well as the programming/storing of the kernel and the control of the convolution operation. However, the internal operation of the pixel is done using low power current mode analog circuits. In this paper we describe the architecture of a fully digital convolution chip. The operation is performed with higher precision while the pixel area is only 20% bigger than the old analog pixel.

2. Architecture of the Convolution Chip

The chip reported in this paper implements 2-D convolutions with programmable and arbitrary-sized kernels. A kernel of size up to 32 x 32 is programmed and stored in a static RAM. Each time an input event is received, the kernel is added to a neighborhood of pixels centered around the coordinate of the incoming event. The system-level architecture of the chip is shown in Fig. 3.

The chip contains the following blocks:

- Static RAM that holds the stored kernel. Each kernel weight is stored with 5 bits plus a sign bit in 2's complement representation.

- Array of 32 x 32 digital pixels. The architecture of the pixel is shown in Fig. 4. Each pixel is basically an adder that accumulates the value of the incoming kernel weight until the accumulator reaches a threshold. When the threshold is reached, an output spike is generated to start communication with the periphery and the accumulator is reset. When the periphery answers, the pixel activates another signal indicating if the limit reached was the positive or the negative one. The accumulator has 17 bits plus a sign bit.

- Synchronous controller. This block performs the sequencing of all operations for each input event, which basically consists of adding row by row the kernel onto the array of pixels.

- High-speed clock, used by the synchronous controller. It is possible to use either this internal clock or an external one.

- Configuration registers, that store several parameters loaded at startup through a serial port.

- 2's Complement, a block that computes the 2's complement of the RAM kernel data before sending it to the pixels. This block is necessary because the input events entering the chip are signed. If the incoming event has positive sign the weights are left unchanged. However, if the incoming event is negative the weights are inverted by the 2's complement block. This way, the sign multiplication is done in the periphery instead of complicating the pixel.

- Left/Right Column Shifter, which is necessary to center the kernel around the input pixel.

- AER-out, asynchronous circuitry for arbitrating and sending out the output address events generated by the array of pixels.

3. Experimental results

A fully digital convolution chip has been designed and fabricated in the AMS 0.35 µm CMOS process. The chip contains 32 x 32 pixels, and each pixel has an area

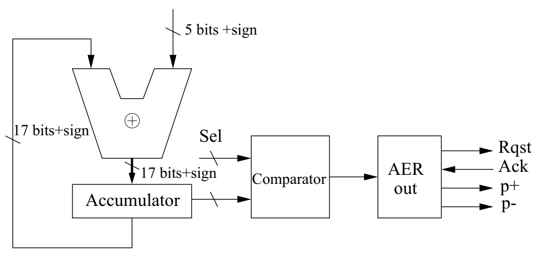

Fig. 4. Architecture of the Convolution Pixel

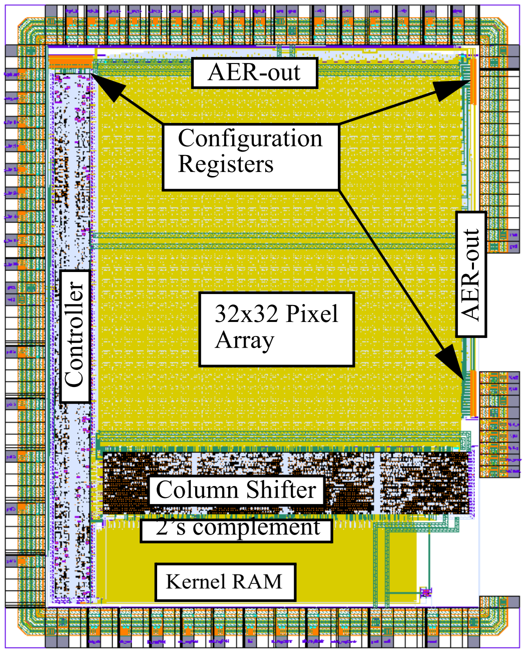

Fig. 5. Layout of the Convolution Chip

of 95 x 100 μm^2. The total chip area is 5.4 x 4.3 mm^2. The layout of the whole chip can be seen in Fig. 5.

A. Setup configuration

We have used a set of PCBs and software to test the basic operation of the chip: configure the chip, send input images and collect output data. The basic setup configuration is shown in Fig. 6.

There are 3 different boards:

- The convolution chip board includes the digital convolution chip with all the analog bias voltages. It has one configuration input connector and two address event connectors (one for input and one for output).

- The configuration board receives the values of the kernel through the serial port and writes it properly into the chip's static RAM. It also receives the configuration data and writes it onto the chip's configuration registers.

- The USB-AER board can be used with two different configurations: as input or output. When working as input, the computer sends a list of AER events through the USB port, and these events are loaded in this board. Then, when activated, the board sends the events to the convolution chip. When this board is configured as output, it records the AER events generated by the convolution chip and sends them to the computer through the USB port [5]. Detailed descriptions of these AER boards can be found in [6].

B. Measured results

In order to check the basic behavior of the convolution chip, we have performed some simple experiments.

In these experiments, we have selected a small part of a real image to be processed with a specific kernel. Fig. 7 shows the input image, of size 32 x 32. The bar on the right indicates the frequencies in Hz relative to each level of grey. Each pixel of the image will be translated into a train of AER spikes such that their frequency is proportional to the pixel intensity.

For this test, we have decided to program the gabor filter kernel for vertical edge extraction shown in Fig. 8. The ideal convolution of the input image with the chosen kernel has been calculated and can be seen in Fig. 9. Then, implementing the same operation with the convolution chip produces the image shown in Fig. 10. As we can see in the pictures, both outputs have very similar spiking frequencies, so the basic behavior of the convolution chip is correct.

4. Conclusions

A fully digital convolution chip has been designed and fabricated. This chip has 32 x 32 pixels and a programmable and arbitrary-sized kernel. It does not perform frame-based image processing, but event-based. It receives input spikes from a previous stage (which can be a retina, another convolution chip or any other AER based processing module) and applies the programmed

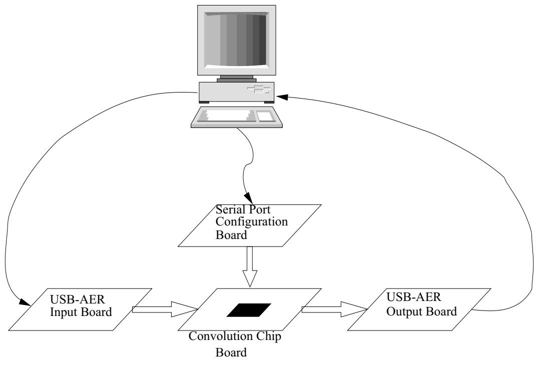

Fig. 6. Test setup configuration

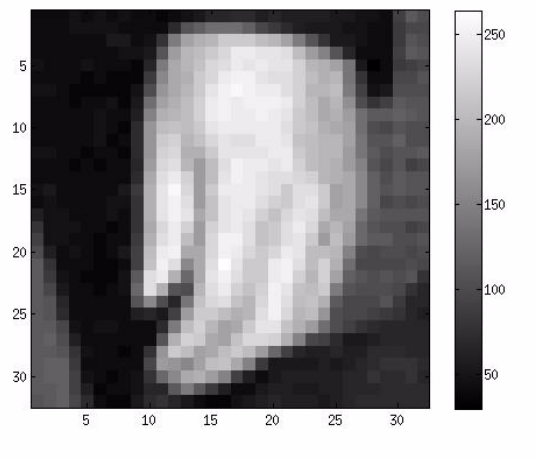

Fig. 7. Input image (spike frequency in Hz)

kernel to generate output spikes that may be processed by the next layer of the chain.

Although the size of the chip is only 32 x 32, larger arrays can be built by tiling M x N chips, so that larger resolution images can be processed.

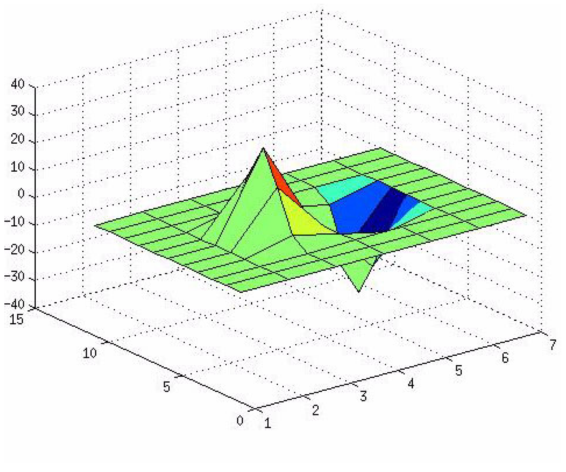

Fig. 8. Vertical edge-extraction kernel

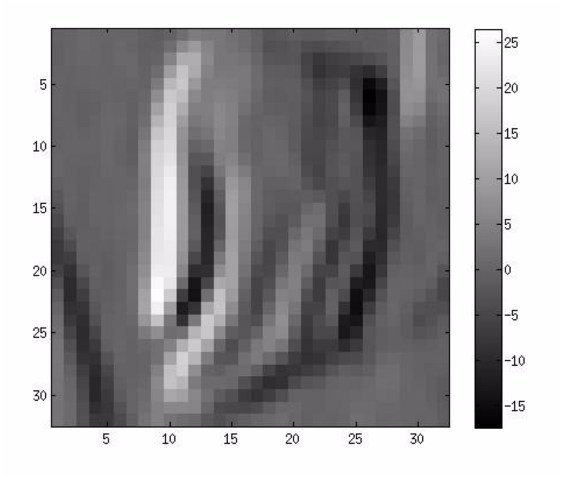

Fig. 9. Ideal output image (spike frequency in Hz)

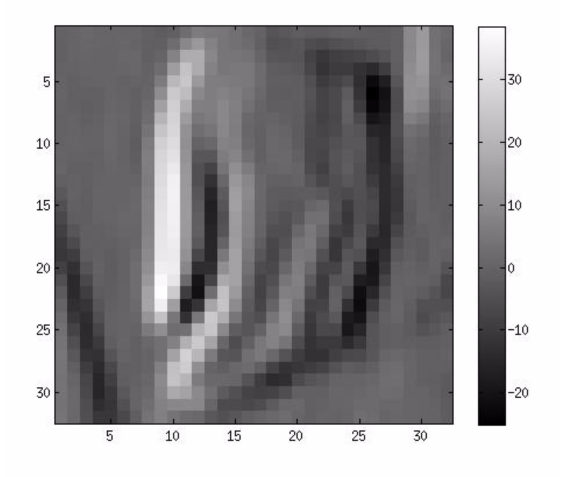

Fig. 10. Experimental output image (spike frequency in Hz)

A new version of the chip with larger resolution is being designed with the help of the experimental results obtained from this prototype.

5. Acknowledgements

The work in this manuscript was supported by EU grant IST-2001-34124 (CAVIAR), Spanish grants TIC-2003-08164-C03-01 (SAMANTA) and TEC2006-11730-C03-01 (SAMANTA II), and local administration from Andalucía grant P06-TIC-01417 (Brain System). LCM was supported by a grant of the Spanish Ministry of Education and Science Research Fellowship Program (FPI), which is co-funded by the European Social Fund.

6. References

[1] G. M. Shepherd, *The synaptic organization of the brain*, Oxford University Press, 3rd Edition, 1990

[2] M. Sivilotti, “Wiring considerations in analog VLSI systems with application to field-programmable networks”, Ph.D. dissertation, Comp. Sci. Div., California Inst. Technol., Pasadena, CA, 1991

[3] T. Serrano-Gotarredona, A. G. Andreou and B. Linares-Barranco, “AER image filtering architecture for vision processing systems”, *IEEE Trans. Circuits Syst. II*, Analog Digit. Signal Process., vol. 46, no. 9, pp. 1064-1071, Sep. 1999

[4] R. Serrano-Gotarredona, T. Serrano-Gotarredona, A. Acosta-Jiménez and B. Linares-Barranco, “A Neuromorphic Cortical-Layer Microchip for Spike-Based Processing Vision Systems”, in *IEEE Transactions on Circuits and Systems, Part I*, vol. 53, No. 12, pp. 2548-2564, December 2006

[5] A. Linares-Barranco, G. Jimenez-Moreno, B. Linares-Barranco and A. Civit-Ballcels, “On algorithmic rate-coded AER generation”, *IEEE Trans. Neural Netw.*, vol.17, no. 3, pp. 771-788, May 2006

[6] F. Gomez-Rodriguez and R. Paz-Vicente et al., “AER tools for communications and debugging”, in *Proc. IEEE Int. Symp. Circuits Syst., (ISCAS’06)*, May 2006, pp. 3253-3256