Poisson AER generator: Inter-Spike-Intervals Analysis

A. Linares-Barranco, D. Cascado, G. Jiménez, A. Civit.
Arquitectura y Tecnología de Computadores.
Universidad de Sevilla.
Av. Reina Mercedes s/n, 41012-Sevilla, SPAIN
alinares@atc.us.es

M. Oster[1], B. Linares-Barranco[2].
[1]Institute of Neuroinformatics. UNI - ETH Zürich.
Winterthurerstrasse 190, CH-8057 Zurich, Switzerland
[2]Instituto de Microelectrónica de Sevilla. CSIC.
Av. Reina Mercedes s/n. Edificio CICA. 41012-Sevilla. SPAIN

***Abstract*— Address-Event-Representation (AER) is a communication protocol for transferring asynchronous events between VLSI chips, originally developed for bio-inspired processing systems (for example, image processing). Such systems may consist of a complicated hierarchical structure with many chips that transmit data among them in real time, while performing some processing (for example, convolutions). To develop AER based systems for image processing it is very convenient to have available some kind of tool for generating AER streams from on-computer stored images. In this paper we present a hardware method for generating AER streams with Poisson statistics in real time from a sequence of images stored in a computer's memory. We quantify that the events generated follow a Poisson distribution using the Kolmogorov-Smirnov test. We have developed a USB-AER board, based on the Xilinx Spartan II FPGA and the Cygnal 8051 microcontroller, developed by our RTCAR group have been used for the analysis.**

I. INTRODUCTION

Address-Event-Representation (AER) was proposed in 1991 by Sivilotti [1] for transferring the state of an array of analog time dependent values from one chip to another. It uses mixed analog and digital principles and exploits spikes for coding information. Figure 1 explains the principle behind the AER basics. The emitter chip contains an array of cells (like, for example, the pixels of a camera or an artificial retina chip) where each cell implements a continuously varying time dependent state that change with a slow time constant (in the order of *ms*). Each cell or pixel includes a local oscillator (VCO) that generates digital pulses of minimum width (a few nano-seconds). The rate of pulses is proportional to the state of the cell (or pixel intensity for a retina) assuming spike rate coding is used. Each time a pixel generates a pulse (which is called "event"), it communicates with the array periphery and a digital word representing a code or address for that pixel is placed on the external inter-chip digital bus (the AER bus). Additional handshaking lines (Acknowledge and Request) are used for completing the asynchronous communication. The inter-chip AER bus operates at the maximum possible speed. In the receiver chip the pulses are directed to the pixels whose code or address was on the bus. In this way, cells with the same address in the emitter and receiver chips are virtually connected with a stream of pulses. The receiver cell integrates the pulses and reconstructs the original low frequency continuous-time waveform. Cells that are more active access the bus more frequently than those less active.

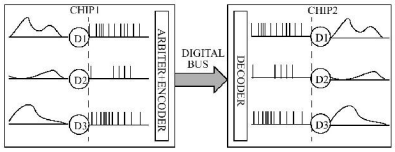

Figure 1: Rate-Coded AER inter-chip communication scheme.

Transmitting the cell addresses allows performing extra operations on the events while they travel from one chip to another. For example in a retina, the activity of the pixels in the array represents the input image. By translating the address of the events during transmission, the image can be shifted or rotated. This translation of the address can be achieved by inserting properly coded EEPROMs. Furthermore, the image transmitted by one chip can be received by many receiver chips in parallel, by properly handling the asynchronous communication protocol. The event-based nature of the AER protocol also allows for very efficient convolution operations within a receiver chip [2].

There is a growing community of AER protocol users for bio-inspired applications in vision and audition systems, as demonstrated by the success in the last years of the AER group at the Neuromorphic Engineering Workshop series [3]. The goal of this community is to build large multi-chip and multi-layer hierarchically structured systems capable of performing complex massively-parallel processing in real time. The success of such systems will strongly depend on the availability of robust and efficient development and debugging AER-tools. One such tool is a computer interface that allows not only reading an AER stream into a computer and displaying it in real-time, but also the opposite: from images available in the computer's memory, a synthetic

AER stream is generated to emulate a dedicated VLSI AER emitter chip [4][5].

In the following sections we present and compare three hardware implementations of one of the already existing methods for synthetic AER generation (the Random method) [8], by evaluating the nature of the distribution of the events respect to the inter spike intervals (ISIs). For this analysis we have used a hardware USB-AER interface.

II. SYNTHETIC AER GENERATION BY HARDWARE

There are many algorithms to map an analog value into an AER stream of addresses [8]. In a rate coding scheme, the frequency of appearance of the address of a given cell must be proportional to the value of the cell. The precise timing of the address pulses is not critical. The pulses can be slightly shifted from their nominal positions because the AER receivers will integrate them to recover the original pixel waveform. Pulse trains recorded from cortical neurons are often modelled by Poisson statistics. In this distribution, the time of each pulse is independent of the time of its predecessor, which allows modelling the statistics of unknown input distributions. Many bio-inspired systems therefore choose Poisson distribution for their spike trains [9][10].

From the software methods proposed in [8] that generate AER streams with Poisson statistics, we present a hardware implementation of the 'Random' method. To reduce the resources needed by the hardware, some modifications have been made. In the following paragraphs we explain the method and the hardware implementation.

The 'Random' method generates an AER stream without any intermediate buffering of events, as was done by the software solution [8]. The analog values of the pixels are digitalized and stored in a matrix of NxN entries with N a power of 2. Each entry can have up to k grey levels, with k=255 in this implementation. From this matrix, the algorithm generates the events dynamically in an iterative way. The algorithm qualifies which address of the AER stream is generated at which time. At each iteration, or 'time slot', an event is generated, or not. This method uses a Linear Feedback Shift Register (LFSR) [6][7] for selecting the pixel of the image in charge of sending an event, and also to decide if the event is going to be sent or not. The LFSR has a resolution of *log(NxNxk)*, and the random number obtained for each time slot is divided into:

1. An index for selecting a pixel of the image and
2. A probability which is used to decide if an event is generated or not, dependent on the gray level of the pixel.

Using the LSFR as a random number generator to select the pixel and to decide if an event is sent or not results exactly in a Poisson distribution of the times of the events. We chose this 'Random' method, since it can be implemented efficiently in hardware. The next section explains in more details the implementation issues.

III. RANDOM METHOD

This method is an implementation of Linear Feedback Shift Register (LFSR) based random number generators. LFSR random number generators are based on a linear recurrence of the form:

$$x_n = (a_1 x_{n-1} + \quad + a_p x_{n-p}) \bmod 2 \tag{1}$$

where p>1 is the order of the recurrence, a_p=1, and $a_j \in \{0,1\}$ for each j. This recurrence is always purely periodic and the period length of its longest cycle is 2^p-1 if and only if its characteristic polynomial

$$P(z) = -\sum_{i=0}^{p} a_i z^{p-i} \tag{2}$$

is a primitive polynomial over the Galois field with p elements [11].

With these premises and limiting the maximum number of address events necessary to transmit an image, we know the number of bits needed for the LFSR and the primitive polynomial. For this implementation, the limit corresponds to a 64x64 image of 256 gray levels, which implies a 20-bit LFSR.

The **P(z)** used for 20 bits (p=20) is:

$$P(z) = z^{20} + z^{17} + 1 \tag{3}$$

which corresponds to the LFSR of Figure 2, where bit 0 is the z^{20} element.

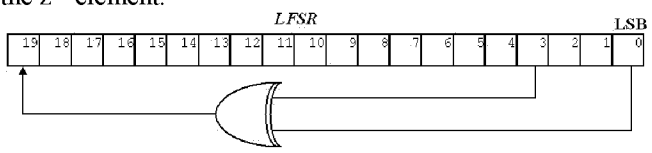

Figure 2: Linear Feedback Shift Register for random synthetic AER generation.

After a reset, all bits are '1', which is the seed of the random number generator. The generated 20-bit numbers are divided in two parts: the 8 more significant bits are the probability and the index of the pixel in the image are the other 12 bits. The method works as follows:

- For each time slot, the LFSR generates a random 20-bit number.
- The 12 less significant bits are used to address a pixel and reads its gray level of the image stored in memory.
- Once addressed that pixel, its gray level is compared with the 8 more significant bits of the LFSR.
- If the gray level of the pixel is greater or equal to the 8 MSB of the LFSR, an event is transmitted with the 12 LSB of the LFSR as the address.
- In the other case, no event is produced for this time slot.

The LFSR ensures that each possible event (gray level) of each pixel is obtained only once within $(2^{20}-1)$ total events.

To improve the Poisson distribution we have considered three variations on the LFSR. The comparisons are presented in the next section. The 20-bit LFSR will generate 2^{20}-1 numbers in a random sequence, but once the LFSR is initialized, the sequence is deterministic. After 2^{20}-1, the

sequence will repeat itself. For low gray levels, for example a gray level of 1, the time between consecutive events will be always the same. This results in a uniform distribution, not a Poisson distribution. So the method using the LFSR as presented above results in events generated with more uniform distributions for low gray levels and more Poisson distributions for high gray levels. To improve this distribution of events we now discuss three modifications.

There are two parameters to the LFSR: the seed and the length. In the next paragraphs we present two variations modifying the seed, and one modifying the length:

A. LFSR with an incremental seed. In this case, the seed of the LFSR is changed for every 220-1 numbers generated, and the seed is changed using a counter.

B. LFSR with a bitwise seed. Now the seed is changed attending to a bitwised decremental counter.

C. 28-bit LFSR. The 8 additional bits are neither used for the pixel selection, nor for the gray level comparison. The 8-MSb are used for the gray level comparator, and the 12-LSb are used for the pixel selection. This increase the Random periodic sequence.

IV. Inter-Spike-Intervals Distribution Analysis

In this section we compare the Inter-Spike-Intervals (ISIs) of this hardware synthetic AER generation method with the normalized distribution that it should have, using the Kolmogorov-Smirnov statistical test.

In neuro-inspired systems, signals can often be modelled by a Poisson distribution [9][10]. The Poisson distribution is being described by the following formula [11]:

$$P_n(T) = \frac{(\lambda T)^n}{n!} e^{-\lambda T} \tag{4}$$

where P is the probability of having n events in time interval T. The distribution of ISIs is the probability that no third event occurs in the interval between two events. This is the exponential distribution:

$$P_0(T) = e^{-\lambda T} \tag{5}$$

A USB-AER board with a Spartan II 200 FPGA [13] has been used to implement the Random method. A 64x64 RAM is written through USB, then an LFSR is used to index the memory and to decide if an event has to be sent. The selected events are queued into a FIFO and a Finite State Machine ensures the asynchronous protocol of the output AER bus. The time between consecutive events depends on the delay of the receiver. This hardware introduces small time (hundreds of ns) differences between events. A 64x64 black image with different gray values for only one pixel is used for the analysis. Another USB-AER board, configured as a datalogger, was used for capturing events and their timestamp, controlled through MATLAB. Figure 3 shows the cumulative probability distribution of ISIs: the expected exponential distribution (continuous curve) versus the measured distribution (stair curve) generated by the Random method with 20-bit LFSR without changing the seed. The gray levels were varied from 50 to 250, in steps of 10, and the value 255. For each curve 32K events were recorded. For high gray levels, the distributions are close to each other, which implies that the statistics of the spike train generated with the hardware Random method follows closely the Poisson distribution.

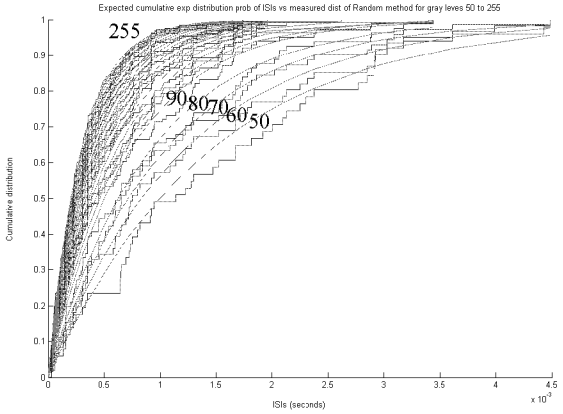

Figure 3: Expected cumulative Exponential ISIs distribution versus measured ISIs distribution generated by the Random method for gray levels.

We used the Kolmogorov-Smirnov (KS) test to quantify how good the observed distributions of ISIs follow the theoretical exponential distribution. Figure 4 shows the result of applying the KS test to ISI distribution obtained with different gray levels, for 64 pixels (the diagonal), 128Kevents, and for the three implementations A, B and C presented. The test is passed if the result is below 5%.

The 64 pixels do not show the same behaviour. This is due to the LFSR. For each pixel the sequence of numbers that the LFSR generates is different. For this reason, we also calculated the average across all pixels. The KS test is only passed by the C implementation for gray levels larger than 90. In all other cases the KS test criterion is not reached.

For small gray levels for implementations A and B, the generated events do not follow a Poisson distribution. The KS-test is never passed for the average of all pixels, only for some pixels with a gray level of more than 90. This is due to the LFSR because all the possible numbers obtained from the LFSR are used to produce a sequence of events, before the sequence repeats itself. For a low intensity, this results in only a few different ISIs. There are only small improvements in the versions of the algorithm that change the seed. The seed will change the start point of the 2^{20}-1 sequence of random numbers, but not the sequence. We address this problem by increasing the number of bits in the LFSR (C), so the shift register has a longer period than a frame. The KS-test has the best result for the C implementation, where the average of all pixels passes the test for gray levels above 90, and there are pixels that pass the test with a gray level above 70.

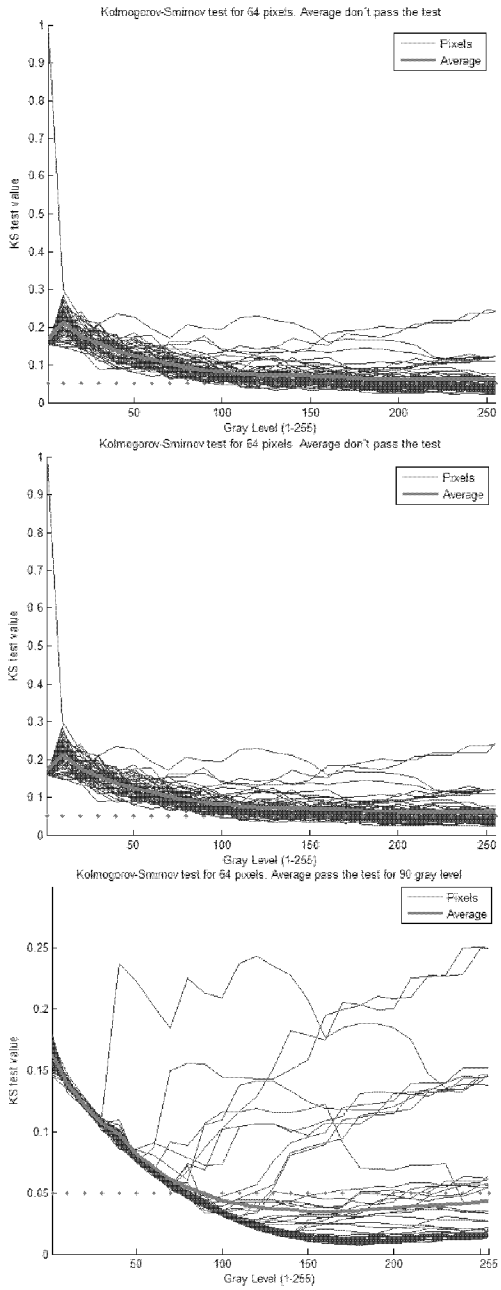

Figure 4: Kolmogorov-Smirnov test of the Random method distribution A(first), B(middle) and C(last). Doted-line: KS-test frontier. Bold-line: average for the 64 pixels KS-statistic.

V. CONCLUSIONS

We presented three hardware implementation of the Random method for synthetic AER generation. We demonstrated that the statistics of the events follow a Poison distribution quite well. Therefore, it could be a very realistic method to be used for neuro-inspired systems, as AER systems. For low frequency of events, the accuracy is increased by introducing more bits in the LFSR implementation. In contrast to others existing AER generators based on the transmission of raw events, like [12], this USB-AER board directly generates Poisson spike trains instead of sequencing raw spike trains. These results have been obtained by using the USB-AER board [13] from MATLABworking under two functionalities: a syntethic AER generator, and a datalogger for capturing the events with their timestamps. These tools are very useful for testing, debugging and interfacing AER system [14].

ACKNOWLEDGMENTS

This work was in part supported by EU grant IST-2001-34124 (CAVIAR), and spanish grant TIC-2003-08164-C03-02 (SAMANTA).

REFERENCES

[1] M. Sivilotti, Wiring Considerations in analog VLSI Systems with Application to Field-Programmable Networks, Ph.D. Thesis, California Institute of Technology, Pasadena CA, 1991.

[2] Teresa Serrano-Gotarredona, Andreas G. Andreou, Bernabé Linares-Barranco. "AER Image Filtering Architecture for Vision-Processing Systems". IEEE Transactions on Circuits and Systems. Fundamental Theory and Applications, Vol. 46, N0. 9, September 1999.

[3] A. Cohen, R. Douglas, C. Koch, T. Sejnowski, S. Shamma, T. Horiuchi, and G. Indiveri, *Report to the National Science Foundation: Workshop on Neuromorphic Engineering*, Telluride, Colorado, USA, June-July 2004. [www.ini.unizh.ch/telluride]

[4] Kwabena A. Boahen. "Communicating Neuronal Ensembles between Neuromorphic Chips". Neuromorphic Systems. Kluwer Academic Publishers, Boston 1998.

[5] Misha Mahowald. VLSI Analogs of Neuronal Visual Processing: A Synthesis of Form and Function. PhD. Thesis, California Institute of Technology Pasadena, California, 1992.

[6] Pierre L'Ecuyer, François Panneton. "A New Class of Linear Feedback Shift Register Generators". Proceedings of the 2000 Winter Simulation Conference.

[7] Linear Feedback Shift Register V2.0. Xilinx Inc. October 4, 2001. http://www.xilinx.com/ipcenter.

[8] A. Linares-Barranco, G. Jimenez-Moreno, A. Civit-Ballcels, and B. Linares-Barranco. "On Algorithmic Rate-Coded AER Generation". Accepted for publication on IEEE Transaction on Neural Networks.

[9] P. Dayan and L. Abbot, Theoretical Neuroscience (MIT Press, Cambridge, MA, 2001)

[10] F. Rieke, D. Worland, R. de Ruyter van Steveninck, W. Bialek. "Spikes: Exploring the Neural Code". The MIT Press, 1999.

[11] J.R. Cogdell. "Modeling Random Systems". Pearson PH, 2004.

[12] Dante, V. and Del Giudice, P. and Whatley, A. M. "PCI-AER Hardware and Software for Interfacing to Address-Event Based Neuromorphic Systems". The Neuromorphic Engineer, n2, 5-6, 2005.

[13] R. Paz, F. Gomez-Rodriguez, M. A. Rodriguez, A. Linares-Barranco, G. Jimenez, A. Civil. "Test Infrastructure for Address-Event-Representation Communications". IWANN 2005. LNCS 3512. pp 518-526. Springer Verlag.

[14] R. Serrano-Gotarredona, M. Oster, P.. Lichtsteiner, A. Linares-Barranco, R. Paz, F. Gomez-Rodriguez, H. Kolle Riis, T. Delbrück, S.C. Liu, S. Zahnd, A.M. Whatley, R. Douglas, P. Häfliger, G. Jimenez, A. Civit, T. Serrano-Gotarredona, A. Acosta, B. Linares-Barranco. AER Building Blocks for Multi-Layer Multi-Chip Neuromorphic Vision Systems. Accepted at NIPS 2005. Vancouver.