An Asynchronous Spike Event Coding Scheme for Programmable Analog Arrays

Luiz Carlos Gouveia, Thomas Jacob Koickal and Alister Hamilton
School of Engineering and Electronics
University of Edinburgh, EH9 3JL, U.K.
Email: L.Gouveia@ed.ac.uk, {Thomas.Koickal, Alister.Hamilton}@ee.ed.ac.uk

***Abstract*— This paper presents a spike event coding scheme for the communication of analog signals in programmable analog arrays. In the scheme presented here no events are transmitted when the signals are constant leading to low power dissipation and traffic reduction in analog arrays. The design process and the implementation of the scheme in a programmable array context are explained. The validation of the presented scheme is performed using a speech signal. Finally, we demonstrate how the event coded scheme can perform summation of analog signals without additional hardware.**

I. Introduction

A field programmable analog array (FPAA) consists of several configurable analog blocks (CABs) that are programmed to perform specific signal processing functions. An important design problem in a FPAA is the communication of analog signals between CABs. Previous implementations of FPAAs used crossbars or matrix switches [1] [2]. These approaches suffer from signal distortion due to voltage drops, parasitic capacitances along the wires and switches and through signal interference. In an alternative approach [3], pulse width modulated signals were used for transmitting analog information in the array. However, this approach requires a global clock signal to synchronize the transmission.

Recently, asynchronous signal processing based on signal dependent sampling strategies has received increasing interest [4]. A motivation for this type of study is drawn from biology where the brain processes signals in analog domain and transmits them as time events [5]. In event based sampling schemes, the intersampling intervals are quantized instead of the signal amplitude and an event is triggered whenever the input signal crosses prespecified levels along the amplitude domain [6]. In [7], signal amplitude information is coded into the timing sequence and this scheme has been extended to recover information from spiking neurons [8].

In this paper we present a spike event coding scheme for transmitting analog signals between CABs. There are some benefits in using an event based coding scheme in a programmable analog array. First, an event coding transmits signals based on demand. This leads to a better utilization of resources due to traffic reduction. Second, in contrast to synchronous signal processing, event based processing benefit from low energy dissipation, freedom from clock skew, immunity to metastable behavior and low crosstalk. Third, event based processing transmits signals as digital spikes and hence it is more suited to communication between distant CABs, even in different ICs, than analog signals [1] [2], allowing greater scalability.

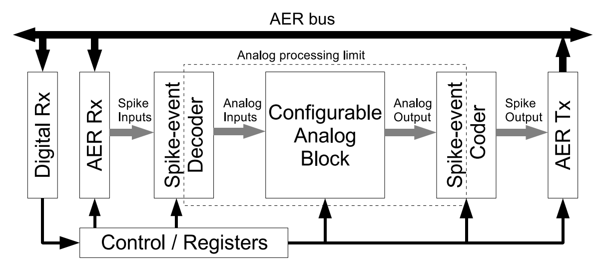


Fig. 1. Block diagram of the spike event communication interface to a configurable analog block (CAB). Spike events are transmitted across the array using AER protocol.

II. Spike Event Coding Scheme

A. Description

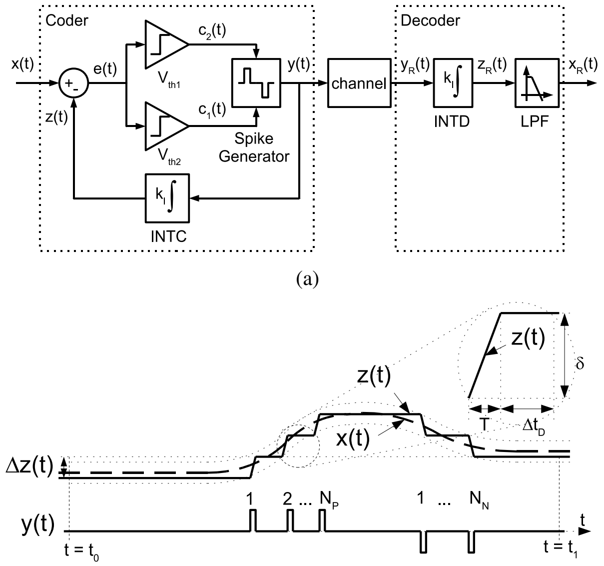
The block diagram representation of a CAB with the spike event communication interface is shown in Fig. 1. Analog signals from the CAB form the input to the spike event coder. In a programmable array context, these spikes can be transmitted between CABs using an AER protocol [9]. The destination CAB reads spikes through an AER receiver and these spike inputs are converted to analog signals at the spike event decoder. The control registers are used for configuration and control of the circuit block.

The spike event coding scheme is shown in Fig. 2(a). This scheme is based on the principle of irregular sampling schemes described in [4] and [6], where it was used to implement asynchronous A/D converters. The event coder tracks the input signal by bounding an error signal given by:

$$e(t) = x(t) - z(t) = x(t) - \int y(t)dt \tag{1}$$

where $e(t)$ is the error between the analog input signal $x(t)$ and coder feedback integrator (INTC) output $z(t)$. The coder output spikes $y(t)$ are sent to the communication channel and to the input of the feedback integrator. The decoder output $x_R(t)$ is an analog signal given by:

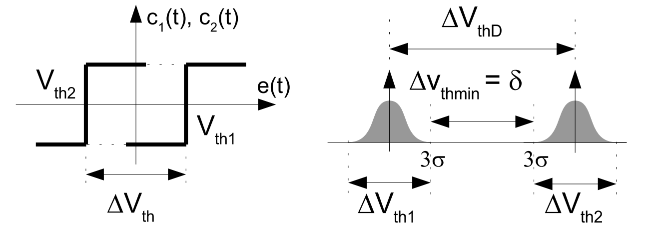
$$x_R(t) = LPF(z_R(t)) \approx \overline{z_R(t)} = \overline{\int y_R(t)dt} \tag{2}$$



(a)

(b)

Fig. 2. (a) Event coding block diagram and (b) waveforms example. An example of predict behavior of some signals are shown in b), where δ is the tracking step, T is the spike width and Δt_D is the time interval between successive spikes.



(a) (b)

Fig. 3. Comparators threshold design. a) Comparators transfer function and b) Comparator offsets ΔV_{th1} and ΔV_{th1} are used to design the threshold difference ΔV_{thD}.

If the channel is ideal, $y_R(t) = y(t)$ and

$$x_R(t) \approx \overline{\int y(t)dt} = \overline{z(t)} = \overline{x(t) - e(t)} \tag{3}$$

The error between the decoder output $x_R(t)$ and coder input $x(t)$ is bounded, $|e|_{max} \leq \delta$, where δ, the tracking step or quantization error, is a system parameter.

The outputs of the event coder are represented by positive and negative fixed short duration pulses (spikes). These spikes are generated by the spike generator when the comparators change their states. Each positive or negative spike generates an incremental or decremental change (δ) at the output of the both integrators (INTC and INTD). Although δ can be varied based on the characteristics of the input signal, in this paper we consider the case for fixed δ only.

The change in the output of both integrators is given by:

$$\Delta z(t) = \delta \left(Np - Nn\right) \tag{4}$$

where $Np(Nn)$ is the number of previous positive (negative) spikes since t_0 as shown in Fig. 2(b).

B. Design

In this section we discuss the design process of the spike event coder and decoder. The first step in the design of the event coder is to determine the tracking step δ:

$$\delta = \frac{\Delta x(t)_{max}}{2^{N_B}} \tag{5}$$

where $\Delta x(t)_{max}$ is the input dynamic range and N_B is the desired resolution in bits.

The tracking step is used to design the comparators thresholds difference $\Delta V_{th} = V_{th1} - V_{th2}$, as shown in Fig. 3(a). Ideally, this difference is equal to the tracking step δ. However, due to the comparators offset the actual ΔV_{th} is bounded ($\Delta V_{thD} + 6\sigma \geq \Delta V_{th} \geq \Delta V_{thD} - 6\sigma$) by a function of the comparator offset standard deviation σ and the designed thresholds difference ΔV_{thD} (Fig. 3(b)). Therefore, the comparators thresholds difference is designed to meet the specification:

$$\Delta V_{thD} \geq \delta + 6\sigma \tag{6}$$

Another design parameter is the spike width T and it is designed according to the input signal and the AER system characteristics. In order to reduce the overload of the communication channel, the spike generator sets a minimum period for the interval between two successive output spikes. This "refractory period" is given by $\Delta t_{Dmin} = kT$. Using Δt_{Dmin} and the specification of the maximum derivative of the input $\left|\frac{dx(t)}{dt}\right|_{max}$, the spike width is determined:

$$T = \frac{\delta}{(k+1)\left|\frac{dx(t)}{dt}\right|_{max}} \tag{7}$$

In an AER system, one of the most important specification is the output spike frequency. The coder output spike frequency is a function of the magnitude of the input derivative:

$$f = \frac{1}{T + \Delta t_D} = \frac{\left|\frac{dx(t)}{dt}\right|}{\delta} \tag{8}$$

From (8), we see that this event coding scheme presents a null output activity when the input signal is constant. This characteristic is beneficial in a programmable analog framework where significant number of bias signals are present.

From (7) and (8), the maximum output frequency is:

$$f_{max} = \frac{1}{(k+1)T} \tag{9}$$

The spike width T and the tracking step δ are used to design the coder and decoder integrator gains given by $K_I = \frac{\delta}{T}$.

Finally, the pole of the decoder low pass filter (LPF) is a key design parameter as it improves the resolution by attenuating the undesirable out-band high frequency harmonics generated during the decoding process. Ideally, the filter should provide total rejection of out-band harmonics with zero in-band attenuation. However, practical implementation of this characteristic being unrealizable, the dominant pole of the filter is designed to be near the cutoff frequency ω, for an input signal with bandwidth of ω.

III. Simulation Results

The event coding scheme was simulated using a speech signal and a pure tone as the coder input. In order to depict the coder functionality clearly, the coder was implemented to provide resolution of 4 bits.

Response to a Speech Signal: The response of the spike event coding to the speech signal is shown in Fig. 4. The speech signal is shown in Fig. 4(a). The decoded signal $x_R(t)$ at the output of a first-order LPF shows a close match with the input speech signal $x(t)$ (see expanded plot Fig. 4(b)). The pole of the LPF was designed to be at 4 kHz (allocated voice bandwidth). Fig. 4(c) and 4(d) demonstrate two important characteristics. First, Fig. 4(c) shows the error $e(t)$ is bounded by ΔV_{th}. Second, Figs. 4(b) and 4(d) show that no spikes are transmitted when the input signal is relatively constant thereby reducing the communication traffic and leading to a better utilization of the resources.

THD measurement: The THD of the system was measured using a 4 kHz sine wave as the input signal. Two LPFs were designed to demonstrate the influence of the pole design: one with the pole at 4 kHz (LPF1) and the second at 40 kHz (LPF2). The coder input $x(t)$, the decoder integrator output $z_R(t)$ and the LPF1 and LPF2 outputs $x_{R1}(t)$ and $x_{R2}(t)$, respectively, are shown in Fig. 5(a). Fig. 5(b) shows the frequency spectrum of the output signals.

The specified resolution of spike event coder is obtained at the decoder integrator output (4.0 bits). The resolution increases to 5.3 bits and 6.7 bits using the filters LPF2 and LPF1, respectively. As stated in Section II, the improvement in resolution in LPF1 is attributed to the larger attenuation of the harmonics because the pole is designed at a lower frequency.

The influence of the refractory period Δt_{Dmin} on the coder performance is shown in Fig. 5(a). Because the initial state of the coder integrator was set to zero and $x(t_0) = 1$, the error signal $e(t)$ is initially greater than δ. The error decreases for each successive output spike occurrence. By choosing $\Delta t_{Dmin} \approx 4.6\mu s$ and $T \approx 100ns$ and according to (9), the maximum output spike frequency is 213 kHz. Therefore, the refractory period Δt_{Dmin} determines the initial tracking speed of the coder.

IV. Computational Application

In a programmable analog array, an important requirement is the capability of adding analog signals; the summation of the outputs of hundreds or thousand synapses in a neuromorphic system is an example. Due to the large number of operators, it is desirable that this operation can be performed without additional hardware like the summation of currents in analog

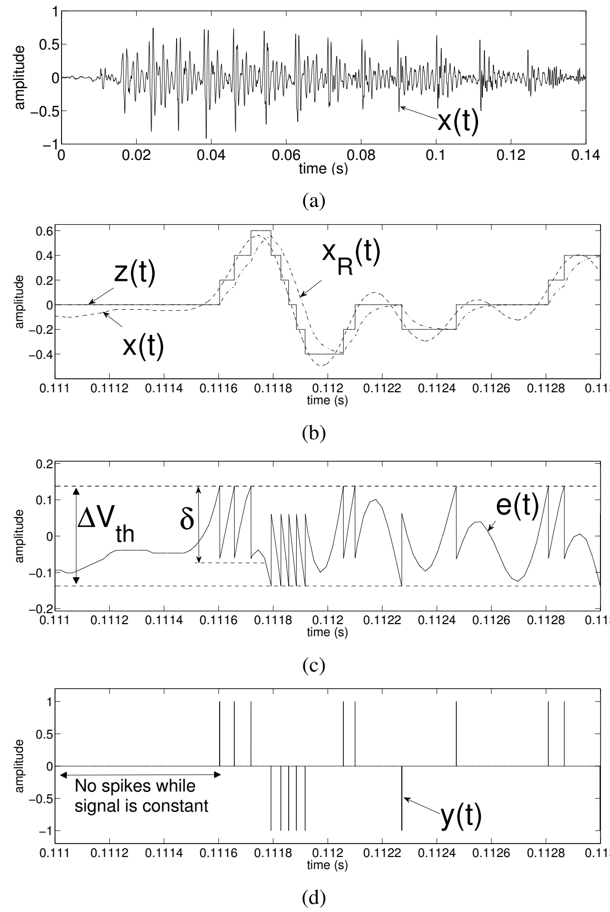


Fig. 4. Example of coding and decoding of a speech signal. (a) The complete speech signal $x(t)$. (b) The expanded plot showing the decoding $x_R(t)$ of the speech signal and the integrator output $z(t)$. (c) The error signal $e(t)$ bounded by the difference of comparators thresholds ΔV_{th}. (d) The spike event coder output $y(t)$.

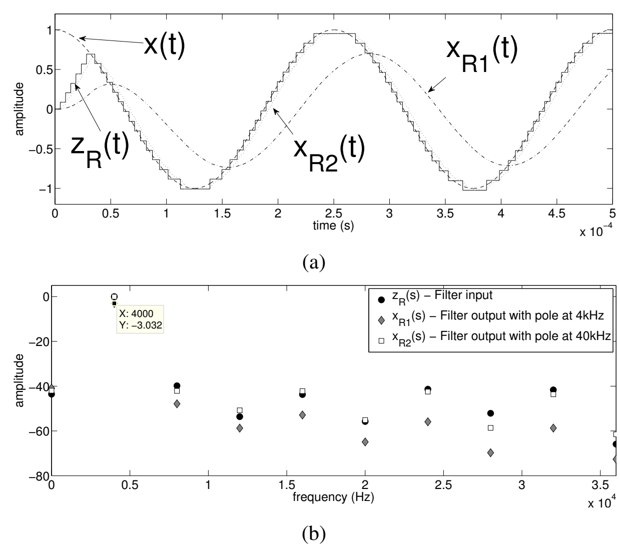


Fig. 5. THD simulation. (a) Decoder output for a 4 kHz sine wave. The integrator output $z(t)$ tracks the input $x(t)$. The signals $x_{R1}(t)$ and $x_{R2}(t)$ are the outputs of filters: the first with the pole at 4kHz and the second at 40kHz. The initial state of both integrators are set to zero. (b) The frequency spectrum of LPF input, $Z_R(s)$ and outputs $X_{R1}(s)$ and $X_{R2}(s)$ were computed for the last period (250 μs < t < 500 μs).

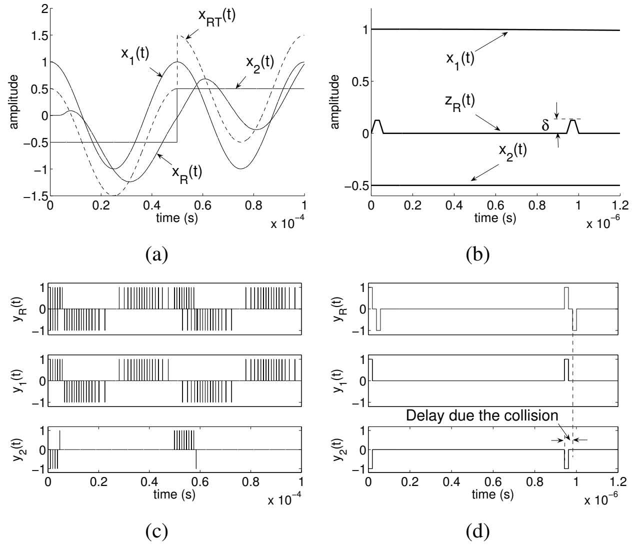


Fig. 6. Example of summing operation. The inputs $x_1(t)$ and $x_2(t)$, the decoder output $x_R(t)$ and a predicted result $x_{RT}(t)$ are shown in graphs a) and b). The spike outputs, $y_1(t)$ and $y_2(t)$, and an AER arbiter output $y_R(t)$ are shown in c) and d). The graphs b) and d) show of the effect of spike collisions: an error with amplitude δ appears on the output $x_R(t)$ between the transmissions of $y_1(t)$ and $y_2(t)$, which were involved in a collision.

domain [10]. Here we show how summation is performed with event coding without any extra hardware.

Using (4), the summation signal $s(t)$ with j operators is:

$$s(t) = \delta \left(\sum_{i=1}^{j} N_{pi} - \sum_{i=1}^{j} N_{ni} \right) \tag{10}$$

where N_{pi} and N_{ni} are the number of positive and negative spikes, respectively, received from the ith operator.

Since AER protocol is used to transmit spike events, collisions during the access to the channel are possible and an arbiter is used to resolve them. Collisions lead to an error in the summation process. This error is given by $\epsilon = \delta\ (N_{pc} - N_{nc})$, where $N_{pc}(N_{nc})$ is the number of positive (negative) spikes in the collision. One possible resolution of the conflicts is performed by queuing and transmitting successively all events involved in the collision (1-persistent). This method was used for the simulation.

Simulation: The results showing the summation of a sine signal $x_1(t)$ and a step signal $x_2(t)$ are shown in Fig. 6, together with the decoder output $x_R(t)$ and the predicted result $x_{RT}(t)$. The coders outputs $y_1(t)$ and $y_2(t)$ and the decoder input $y_R(t)$ are shown in Fig. 6(c).

The expanded results in Fig. 6(b) show the effect of spike collision (Fig. 6(d)) in the summation result using a 1-persistent arbiter: an error (with amplitude δ in this case) between the transmission of the spikes $y_1(t)$ and $y_2(t)$.

The decoder output $x_R(t)$ follows the predicted result $x_{RT}(t)$, except for the limited tracking time in the step signal coding and for the spike collisions in the AER bus.

V. Conclusions

In this paper we presented a spike event coding scheme for the communication of analog signals in a programmable array. The scheme transmits spike events based on input signal activity thereby providing efficient utilization of resources and lower power consumption. Further the events are transmitted digitally providing improved scalability in building large programmable arrays. The methodology of the scheme and the parameters design process were presented. The functionality of the event coded scheme was validated through simulations. We demonstrated how event coding can be used to add analog signals without extra hardware; an important feature in programmable analog systems. Currently the circuits of the spike event communication interface are being implemented on a chip to interface CABs in a programmable array developed by the authors [11].

Acknowledgment

The authors would like to thank Engineering and Physical Sciences Research Council (EPSRC), for supporting this work under the grant to University of Edinburgh (EP/C015789/1)

References

[1] E. K. Lee and P. G. Gulak, *A CMOS field-programmable analog array*. In IEEE Journal of Solid-State Circuits, vol. 26, no. 12, pp. 1860-1867, December 1991.

[2] D. R. D'Mello, and P. G. Gulak, *Design approaches to field-programmable analog integrated circuits*. In Special Issue on Programmable Analog Systems, Analog Integrated Circuits and Signal Processing, Hingham: Kluwer Academic Publishers, vol. 17, no. 1-2, pp. 7-34, September 1998.

[3] K. Papathanasiou, T. Brandtner and A. Hamilton, *Palmo: pulse-based signal processing for programmable analog VLSI*. In IEEE Transactions on Circuits and Systems-II, Analog and Digital Signal Processing, vol. 49, no. 6, pp. 379-389, June 2002.

[4] Y. W. Li, K. L. Shepard and Y. P. Tsividis, *Continuous-time digital signal processors*, In Proceedings of eleventh IEEE International Symposium on Asynchronous Circuits and Systems, pp. 138-143, March 2005.

[5] W. Gerstner, *Spiking neurons*. In Pulsed neural networks. W. Maass and C. Bishop eds., Cambridge: MIT Press, pp. 4-53, 1998.

[6] E. Allier, G. Sicard, L. Fesquet and M. Renaudin, *A new class of asynchronous A/D converters based on time quantization*, In Proceedings of Ninth International Symposium on Asynchronous Circuits and Systems, pp. 196-205, May 2003.

[7] A. A. Lazar, L. T. Toth, *Perfect recovery and sensitivity analysis of time encoded bandlimited signals*, IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, vol.51, no.10, pp. 2060-2073, October 2004.

[8] D. Wei and J. G. Harris, *Signal reconstruction from spiking neuron models*. In Proceedings of the International Symposium on Circuits and Systems, vol. 5, pp. 353-356, May 2004.

[9] K. Boahen, *Point-to-point connectivity between neuromorphic chips using address events*. In IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, vol. 47, no. 5, pp. 416-434, May 2000.

[10] G. Indiveri, E. Chicca, R. Douglas, *A VLSI array of low-power spiking neurons and bistable synapses with spike-timing dependent plasticity*, IEEE Transactions on Neural Networks, vol. 17, no. 1, pp. 211-221, Jan. 2006.

[11] T. Koickal, A. Hamilton and L. Gouveia, *Programmable analog VLSI architecture based upon event coding*, Second NASA/ESA Conference on Adaptive Hardware and Systems, pp. 554-562, August 2007.