

ARBITRATED AER IMAGE CODING SCHEMES

¹Mohamad Susli, ¹Farid Boussaid, ²Chen Shoushun, and ²Amine Bermak

¹School of Electrical, Electronic and Computer Engineering
The University of Western Australia, Perth Australia

²Department of Electrical and Electronic Engineering, Hong Kong University of Science and
Technology
Clear Water Bay, Kowloon, Hong Kong, SAR.

Abstract. In this paper, we present image coding schemes based on Address-Event Representation (AER) for image capture and transmission. Potential arbitration schemes are explored and their performance is evaluated in terms of event generator element, latency and output bit stream compression using an in-house AER simulator.

Key words: Address-Event Generation, arbitration, pulse coding, pixel, latency, compression.

1. Introduction

In digital cameras, images are acquired by reading out sequentially a photosensitive pixel array [1]. The brightness of each pixel will depend on the amount of light that falls on the photosensitive cell as well as on the duration of the integration period. In the case of high resolution imaging systems, conventional read out results in significantly reduced frame rate and leads to high power consumption since row and column pixel selection circuitry will need to be active for a longer period of time. In this paper, we explore the potential benefits of an imaging system based on Address-Event-Representation (AER). This biologically inspired data representation is modeled after the transmission of neural information in biological systems [2][3]. AER can be used in an imaging system to mediate access to the digital output bus, which constitutes the single communication channel. Each time an event occurs (for instance when a predefined voltage is reached), a spike is generated by a pixel and a request for bus access is made to a peripheral arbiter. The latter takes the pixel address and places it on the bus. As a result, the asynchronous bus will carry a flow of pixel addresses. At the receiver end of the bus, address and time information are combined to retrieve the original data (e.g. pixel brightness value). In an AER-based imaging system, pixel read-out is initiated by the pixel itself. As a result, bus access is granted more frequently to active pixels (i.e., pixels that have generated events) than less active pixels, which will in turn consume much less communication bandwidth. The AER communication protocol makes efficient use of the available output bandwidth since read out can be achieved at any time upon request. In terms of power consumption, AER is also more efficient than the conventional fixed time-slot (synchronous) allocation of resources; this because not all pixels are likely to require

computation/communication resources at the same time, hence there is no waste of resources. In the next section, we describe the basic building blocks of an AER based-imaging system. Section 3 presents the AER simulator developed to investigate potential AER imaging schemes. Performance results are discussed in Section 4. Finally, concluding remarks and perspectives are given in Section 5.

2. AER-Based Imaging

2.1 Event Generator

In an AER-based imaging system, each pixel comprises an event generator, which is used to request access to the output bus, each time a pixel has reached a predefined threshold voltage. The output of the event-generator pixel can be either a single pulse or a sequence of pulses [4]. In the latter case, the event-generator is referred to as Pulse Frequency Modulated (PFM) with the inter-spike interval a linear function of the pixel brightness value. In the case of a single output pulse, the event-generator is referred to as Pulse Width Modulated (PWM) because the duration of the pulse width is inversely proportional to the pixel brightness value. The PWM event-generator offers lower power consumption (a single transition) at the cost of a non-linear response [5]. Both PWM and PFM schemes encode illumination information in the time domain, providing noise immunity by quantization and redundancy. In addition, representing intensity in the temporal domain, allows each pixel to have a large dynamic range (up to 200dB by modulating the reset voltage), since the integration time is not dictated by a global scanning clock. Moreover, time encoding ensures a relative insensitivity to the ongoing aggressive reduction in supply voltage that is expected to continue for the next generation of deep submicron silicon processes.

2.2 Arbitration

The situation when two pixels send their values simultaneously on a serial bus can be resolved using arbitration. The basic idea is to setup a queue where each pixel independently announces that it is ready to send its address (when the threshold is reached) and then awaits an acknowledgement from a control unit. When the acknowledgement is received, the pixel removes its request and resets itself in order to be ready for the next frame. The simplified model of the signaling is shown below in Fig. 1.

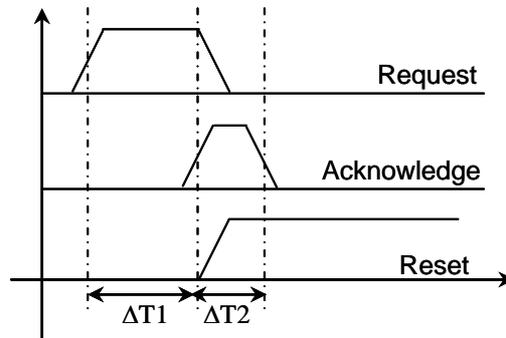


Fig. 1. Pixel signaling

Note that there are two types of timing errors. There is a signaling delay (handshaking error) labeled ΔT_2 and a waiting error ΔT_1 called the arbitration delay. Now we turn to the question of the acknowledgement itself. When faced with multiple requests how would one choose which ones to acknowledge? The solution is to use a circuit element known as an arbiter. An arbiter's function is to decide which request came first and then acknowledge it. In the situation where two requests come at the same time, only one of them will be chosen and acknowledged.

From the basic two-request line input arbiter (Fig. 2), we can build up an arbiter for a bigger system by connecting the acknowledgement lines into another arbiter's request line. This way the outputs of two arbiters become the input of one arbiter, forming a four-request line arbitration block. Since each arbitration block will have its own delay time, it is trivial to show that this delay will be logarithmic with respect to the number of request lines. (This is of course if we assume all delays are equal, normally two temporally close requests will generate a larger delay) This delay is the arbitration delay ΔT_1 mentioned previously and it is critical that it should be minimized in order to increase the image quality in the camera.

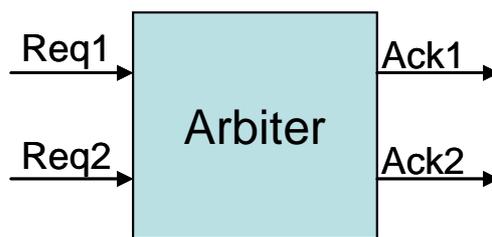


Fig. 2. Arbiter signals

3. AER-Based Imaging

An AER simulator was developed to evaluate the performance of possible AER image coding implementation schemes. The simulator was developed in C++. It is

extremely fast, requiring less than a second to simulate the behavior of a 128x128 pixel array. The block diagram of the simulator is shown in Fig. 3.

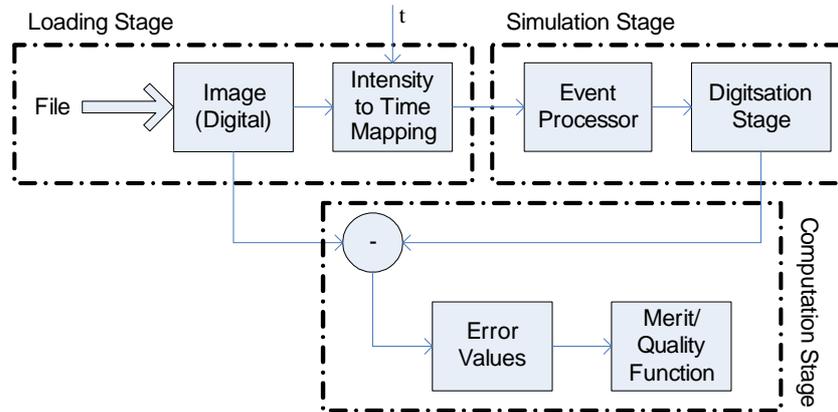


Fig. 3. AER Simulator block diagram

The AER simulator system uses an event-wheel based on the heap data structure. This event-wheel stores the upcoming events such as a pin going high, an arbiter receiving an edge or a photodiode discharging down to its threshold voltage. The heap data structure insures that the events are processed sequentially in time. However, the event-wheel has additional functionalities which processes some events with higher priority. Most processed events will typically generate events in the future. These are pushed on to the event-wheel, which will continue to process events until it is empty, in which case the simulation ends. The initial state of the camera is assumed to be a reset state, where no pixels are discharging and all buffers are cleared. An image is then loaded, whose digital values are translated into discharge times (time until first spike values). A global integration signal is then simulated, where all pixels create an event in the future at the time when they are expected to reach their threshold voltage. The state of the pixels are thus in the “integrating state” and will eventually be processed one by one into their next state. Since the simulator is event-based, even though this is simulated sequentially, it is done so con-currently in time. The current time is never incremented manually; instead it is refreshed with the latest event’s value. By operating in this fashion, delays will propagate themselves and thus simplify the operation of the simulator greatly. The buffers hold the locations of the pixels which have fired and are awaiting arbitration. This is to simulate the memory realized by the physical buffers. This buffer structure is routed into the neighboring arbiters using a look up table, based on row or column location. This speeds up simulation immensely, however it needs to be initialized once before simulation can commence. The arbiters accurately simulate the logical behavior of the real cell. There are two events which effect row and column arbiters, “arb_edge” and “arb_update”. The former is an event which is called whenever an arbiter’s neighboring cell changes its output. Since it takes some time window in order to

decide on the arbitrated signal (if any), some propagation time is taken before the new even “arb_update” is generated. This will then also have a propagation delay of its own, depending on whether the priority or non-priority line has been selected. This models the real life behavior, since the pull strength of the non-priority line is weaker, it will take longer to create a transition. Finally, arbiters toggle priority by simply swapping a logic bit upon selection of a line. When the simulation ends, each pixel will store a value which contains the total time until it receives a column acknowledgement. This is taken to be equivalent to the time which it would be placed on the AER stream. Thus by creating a new heap and pushing on these locations and times, an AER stream is formed by this simulator. In order to re-digitize the image, a simulation illustrated by Figure 3 is leveraged. Thus a comparison can be made between the original image, and the errors generated by arbitration and AER.

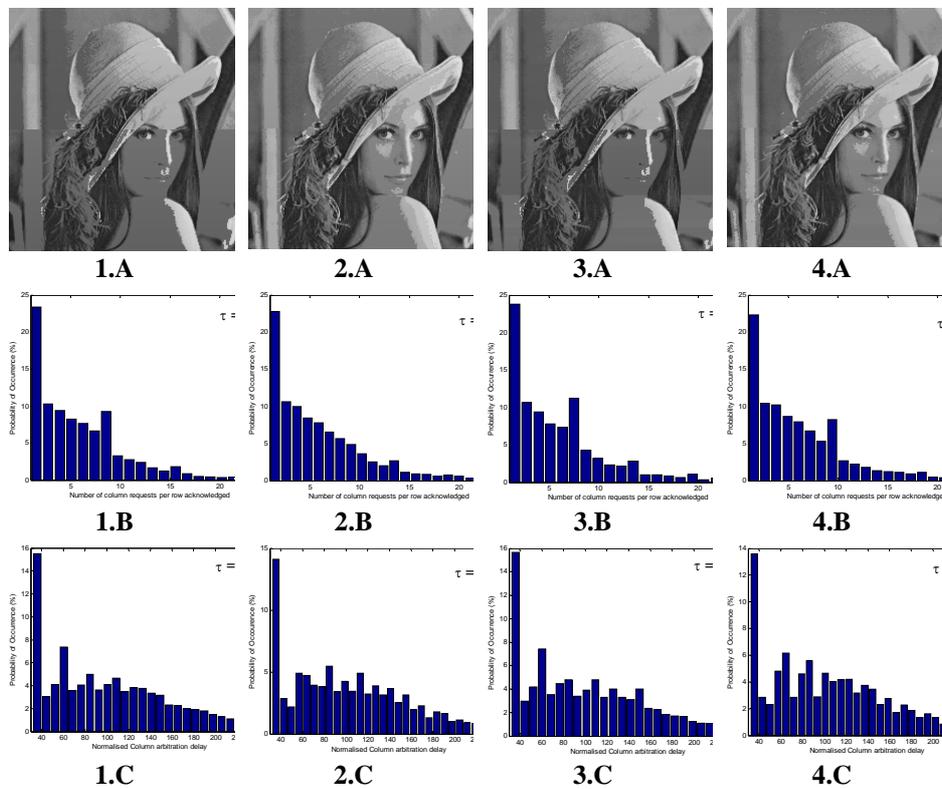


Fig. 4. Fixed arbitration (1.A-C and 2.A-C) and Fair arbitration (3.A-C and 4.A-C): Delay and number of requests as a function of the intensity-to-time conversion factor τ . The original image Lena is 256×256.

4. Results

4.1 Latency

The AER simulator emulates the finite time required for arbitration of an image. It thus introduces additional timing errors as illustrated in Figure 4. In AER, brighter pixels are favored because their integration threshold is reached faster than darker pixels. As a result, brighter pixels will request the output bus more often than darker ones. This results in an unfair allocation of the bandwidth. Two different arbitration schemes were examined, with images 1 and 2 utilizing a fixed arbiter, and images 3 and 4 utilizing a fair arbiter. In the fixed arbiter, priority is given always to the same input lines. In contrast, a fair arbiter toggles the priority between input lines. It should be noted that fair arbitration gives a subtle enhancement of the output image. In addition to different arbitration schemes, different values of τ were used in order to evaluate the degradation of image quality, τ representing the conversion factor used by the AER simulator to map intensity into the time domain (Fig. 3). This effect is quite pronounced, with the upper half of the image gaining priority over the lower half. An AER imager typically processes its columns after a row has been chosen to have a greater priority over the others which are waiting. Thus the time it takes to process any row is compounded over all waiting rows. In order to evaluate this timing delay, a histogram (images 1-4B) was constructed of the number of pixels per row which make a request once their row has been acknowledged. The results (which were cropped to eliminate negligible bins) show that fair arbitration tends to reduce the amount of requests on average, but only a small amount for this image. Finally, the delay of each row was recorded, and plotted in a histogram representation, for each respective image. As intuitively expected, there is a direct correlation between the number of pixel requests and the delay, on average.

4.2 Event Generator

In Figure 5, we see the difference between PWM and PFM encoding, with different values of τ . Both of these simulations are run using fair arbitration. Figures 1-3A show the PWM encoding scheme at values of the intensity-to-time factor τ of 1.1, 0.7 and 0.3 respectively. We see that for a value of 1.1, arbitration causes only a few minor losses in contrast. Lowering τ to 0.7, the error begins to dominate in one half of the image and at 0.3 the loss in contrast is apparent everywhere in the image. We can see the error due to PFM encoding in figures 1-3B, for the same values as before, however with a capture time of 0.1. This value represents the amount of time



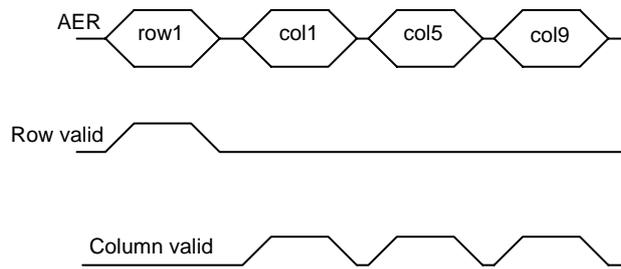
Fig. 5. PFM and PWM encoding as a function of the intensity-to-time conversion factor τ .

the simulation is run upon which all events are blocked and the occurrences are counted immediately. For a value of τ of 1.1, the image quality is greater than that of the PWM encoding scheme when judged by a human observer. However, the image has lost a great deal of contrast. At a value of 0.7, the image loses more contrast, however the quality can be judged to remain high than that of the TFS image. Finally at a value of 0.3, the imager begins to suffer from arbiter starvation, where a single row arbiter is receiving requests faster than it is dropping them, leaving one half of the image without acknowledgement.

4.3 Output bit stream compression

A possible encoding scheme, which eliminates arbiters for the column requests, is that of a sequential column scan (Fig. 6). A sequential scan reads column requests as packets, sequentially, and encodes them in a binary signal. For example, the stream will contain 0 for a pixel which has not yet generated an event and it will contain a 1 when it has. This has the advantage that the column delay is constant, and this even for the worst case for which all pixels of a row will fire simultaneously. For larger pixel arrays, this advantage is quickly lost as it is unlikely that many pixels of the same row would fire simultaneously (Fig. 4).

AER with Column arbitration:



AER with Sequential Column Scan:

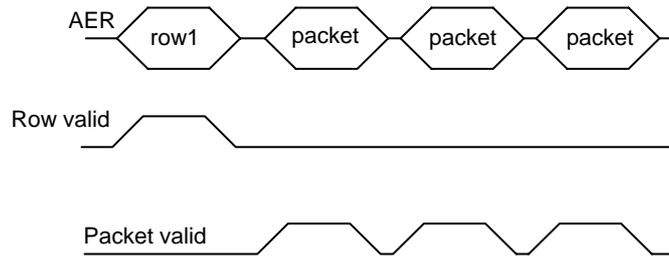


Fig. 6. Output bit stream timing for an acknowledged row.

5. Conclusion

In this paper, we have explored the potential benefits of AER image coding schemes for high speed low power image capture and transmission. An in-house AER simulator was developed to examine potential arbitration schemes such as fixed and fair arbitration protocols and evaluate their performance evaluated in terms of event generator element, latency and output bit stream compression.

Acknowledgements.

The work described in this paper was supported in part by a grant from the Australian Research Council.

References

1. E. Fossum, "CMOS image sensors: Electronic camera-on-a-chip," *IEEE Trans. Electron Devices*, vol. 44, no. 10, pp. 1689-1698, Oct. 1997.
2. M. Sivilotti, "Wiring considerations in analog VLSI systems with applications to field programmable networks," Ph.D. dissertation, California Institute of Technology, Pasadena, 1991.
3. M. A. Mahowald, "VLSI analogs of neuronal visual processing: a synthesis of form and function," Ph.D. dissertation, California Institute of Technology, Pasadena, 1992.
4. E. Culurciello, R. Etienne-Cummings, and K. A. Boahen, "A biomorphic digital image sensor," *IEEE Journal of Solid-State Circuits*, vol. 38, pp. 281-294, 2003.
5. A. Kitchen, A. Bermak, and A. Bouzerdoum, "A Digital Pixel Sensor Array With Programmable Dynamic Range," *IEEE Transactions on Electron Devices*, Vol. 52, Issue 12, pp.2591-2601, Dec. 2005.
6. K. A. Boahen, "Point-to-point connectivity between neuromorphic chips using address events," *IEEE Trans. on Circuits and Systems II*, ol. 47, Issue 5, pp. 416 - 434, May 2000.
7. E. Culurciello, R. Etienne-Cummings, and K. Boahen, "Arbitrated address-event representation digital image sensor," *Electronics Letters*, vol.37, Issue 24, pp. 1443-1445, 2001.
8. Van Rullen, and S. J. Thorpe, "Rate Coding Versus Temporal Order Coding: What the Retinal Ganglion Cells Tell the Visual Cortex," *Neural Computation*, vol. 13, pp. 1255-1283, 2001.