# Hierarchical Token Rings for Address-Event Encoding

Prafull Purohit
Yale University
prafull.purohit@yale.edu

Rajit Manohar
Yale University
rajit.manohar@yale.edu

***Abstract*—Address-event representation (AER) is an event-driven, neuromorphic inter-chip encoding and communication protocol originally proposed to communicate location and timing information of sparse neural events between neuromorphic chips. The protocol is widely used in bio-inspired, event-based vision sensors to communicate visual events. The same approach has been explored for scientific imaging applications, but the performance of existing encoding schemes degrades in the presence of large sensor arrays and a wide range of event rates. In this paper, we introduce a new AER encoding scheme based on hierarchical token-rings (HTR), which blends event-based and scanning approaches. We show that HTR offers significant improvement in latency, throughput, and power compared to existing tree-based approaches in the context of scientific imaging applications.**

## I. Introduction

Silicon neuromorphic systems have a large distributed array of computation units ("neurons") that operate at relatively slow Biological timescales. Each computation unit has extremely high output fan-out, which makes direct point-to-point wiring intractable when designing large-scale systems with millions or billions of neurons. The standard approach to this problem is to leverage the speed of modern CMOS and use time-multiplexed wires [20].

As shown in Fig. 1, a simple implementation of such a system can be designed by taking a large number of relatively slow inputs, representing information to be transferred, and time-multiplexing them on a high-speed communication channel to send data across the interface. Depending on the application domain, for example neural processing unit, vision system or imaging system, the computational unit are neurons or pixels and the communicated information consists of a spike, event, or pixel data. Address-Event communication, sparse readout, asynchronous readout, and event-driven readout are some of the names used to describe the mechanism of communicating sparse data between populations of such computational units.

Address-Event-Representation (AER) is an example of a communication protocol to encode neural events and time-multiplex that information onto a shared output channel. Early work by Sivilotti [2] and Mahowald [3] used AER to communicate between neuromorphic chips using spikes and later work demonstrated its use in several neuromorphic chips (e.g. [4], [6], [16]). AER has been used in

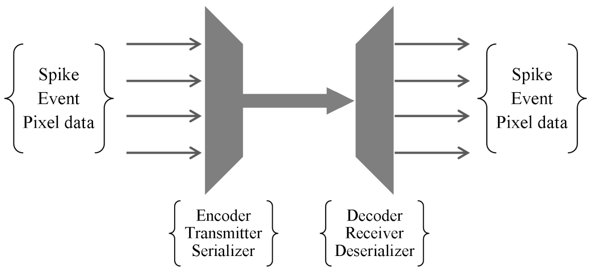


Fig. 1: Address Event Encoding

many applications including imagers based on the silicon retina [8]–[10], [13], [15], [19] and many other bio-inspired systems. A similar event-driven approach was developed for imaging devices such as pixel array detectors (PADs) and monolithic active pixel sensors (MAPS) for scientific applications [21], [22]. Some of these access topologies are examined and compared in [8] for throughput, latency, and power consumption. Even though AER-based communication has been demonstrated in various neuromorphic and imaging system, different approaches have benefits and drawbacks. Encoding based on a binary tree topology [7] is suitable for applications with low event-rates and small array size; however, latency and throughput degrades when the array size increases or when bursts of events arrive in a short time frame. Another approach is to use a ring-based topology [16]. This approach can quickly service a burst of localized events but suffers when sparse events are far apart in space.

In this paper, we present a new parameterized address event encoding scheme based on hierarchical token rings that is tailored to the requirements of scientific imaging applications. The proposed design is able to scan through burst of spatially correlated events and reduces token travel distance for sparse events. As shown in Fig. 3(a), an arbitration mechanism is implemented using hierarchical token rings. Leaf servers (gray boxes) manage communication requests from a pixel array. A single token travels through hierarchical rings to provide multiplexed access to the shared output bus in a fast and timely manner. By changing the size of the rings and the number of levels of hierarchy, this approach can be tailored to the needs of different applications.

We discuss some imaging applications from parts of the electromagnetic spectrum outside visible light and challenges they pose for AER-based readout (Section II). In Section III we discuss previously designed encoding schemes for AER and our motivation for designing a new encoding scheme. Details of the proposed encoding scheme and circuit implementation are provided in Section IV. Section V presents results and comparisons with previous schemes and Section VI concludes this paper.

## II. Imaging applications

In this section we look into few techniques in electron microscopy and synchrotron radiation imaging. We also discuss some of the imaging modes and capabilities which are essential for a good imaging device.

### *A. Electron Microscopy*

An electron microscope is a complex scientific instrument which uses beam of accelerated electrons to capture high resolution images of biological and non-biological images. Over the years, it has been used to study materials in a wide range of research areas such as life sciences, material science, pharmaceuticals, semiconductors, and electronics manufacturing. Scanning transmission electron microscopy (STEM) is a technique used in electron microscopy to study the structure of materials at the nano-scale. A highly focused electron beam raster scans a thin specimen and the diffraction pattern resulting from their interaction is recorded with the help of an imaging device. This recorded data contains quantitative information about the sample and helps in extracting properties such as thickness, strain & tilt, polarity, and electric & magnetic fields with atomic resolution [23]. A full diffraction image at every scan position is often recorded for post-processing.

Several imaging modes exist for extracting specific information about the specimen by observing well established regions of interest in images. Some of the common modes are Bright Field (BF), Annular Bright Field (ABF), and Annular Dark Field (ADF). A full diffraction image is acquired and masking techniques are used during post processing to extract information. A few segmented detectors have been designed specifically for such modes by hardwiring pixel arrangements to match a particular mask but this approach makes the detector unsuitable for general imaging applications.

Applications such as single particle experiments and cryo-EM focus on reducing dose to minimize adverse effects in organic specimens [17]. A low dose means less signal reaches the detector and therefore fewer events are generated. Fig. 3(a) in [18] shows the image of a specimen collected under a low dose condition. Due to very low flux, sparse electron hits are clearly visible in the image. An imaging device should be able to identify individual electron hits and communicate this information.

### *B. Synchrotron radiation imaging*

A synchrotron light source is a large machine that generates bright x-ray beams to study static and quasi-static properties of materials and molecules. For example, synchrotron radiation imaging techniques are used for structural analysis of crystalline and amorphous material. In such techniques, an x-ray beam transmits through the sample and the projection image is recorded on the detector. The intensity distribution in the captured image is then used to analyze the structural properties of a material. Computed tomography (CT) is another technique which provides 3D structural properties of the specimen. In this technique, multiple images are taken at different angles of rotation to reconstruct a 3D representation of the specimen. Data collection time can be improved in such experiments by using a fast framing detector. X-ray diffraction imaging is another technique for characterizing materials where an x-ray beam passes through the sample and the diffraction pattern resulting from their interaction is recorded. Applications such as scanning transmission x-ray microscopy (STXM) and coherent X-ray diffraction imaging (CXDI) often require a large number of diffraction images. A mechanism for reading out selected regions of diffraction can help in reducing data storage requirements and increase effective frame rate.

X-ray photon correlation spectroscopy (XPCS) is a technique to study the dynamics of a material by analyzing temporal correlations among photons scattered by the material [5]. Often these temporal correlations are computed after acquiring images using a conventional imaging device. [14] reported that only 5% of pixels contained useful data while compressing images. In order to detect intensity changes at timescales of the dynamic process, a detector must have good spatial and temporal resolution. A large pixel array improves spatial resolution but limits temporal resolution due to limited frame rate of the detector. A detector capable of reading out sparse speckle data can significantly improve readout speed, allowing study of fluctuations at a faster timescale.

In the last few years, both fields have reported many technological advancements and new imaging methods with development of pixel array detector (PAD) and monolithic active pixel sensor (MAPS) devices. These devices read out data from all pixels sequentially and offer very limited flexibility in reading out data from selected pixels. After considering all of the imaging applications presented above, it is clear that a detector used for electron microscopy and x-ray imaging should be able to efficiently read images in one of the three modes: sparse readout (mode-S); cluster/region-of-interest readout (mode-C); and full-frame readout (mode-F).

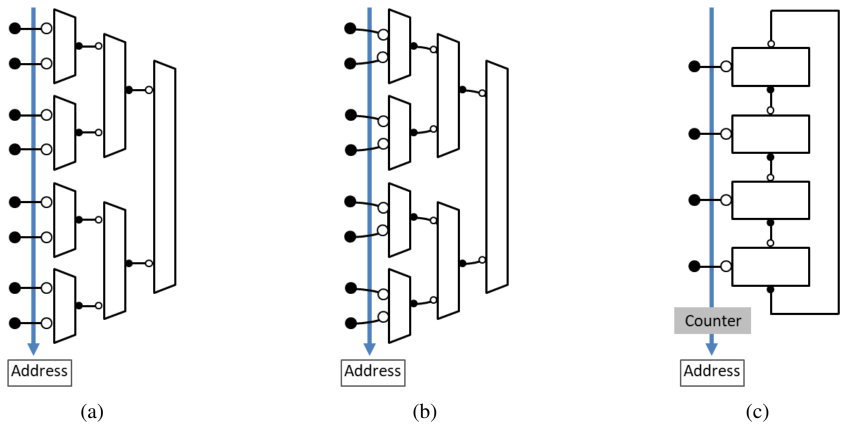


Fig. 2: Different address-event encoding schemes. (a) Binary tree. (b) Greedy tree. (c) Token-ring. ⊸ denotes the passive end of the channel and ⟜• denotes the active end of the channel.

## III. AER Topologies

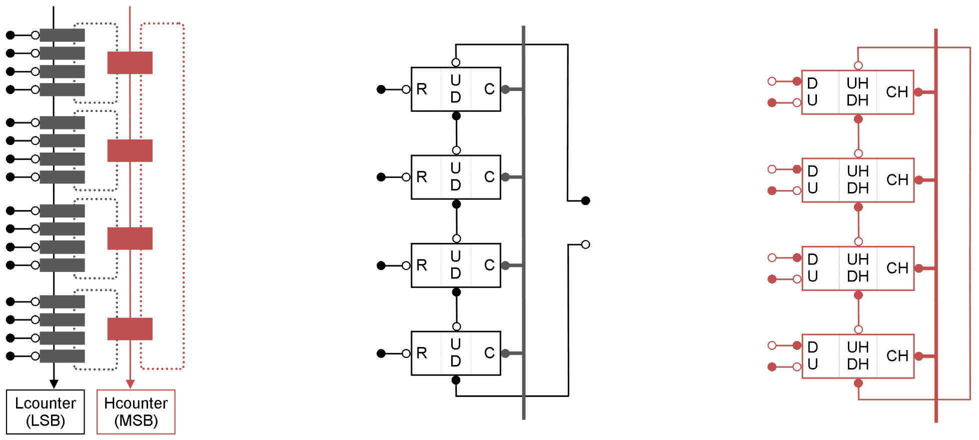
Address-event encoding is an adaptation of time-division multiplexing for communicating spike information between neuromorphic chips [2]. In a typical AER design (Fig. 1) an encoder waits for an incoming event and multiplexes that information as an encoded address on the output bus. The addresses are encoded based on the location of incoming event and sent across the output bus as they occur. A major benefit of this encoding scheme is that timing information is represented by the relative position of event address in the output stream if events are spaced apart in time. When multiple events occur, an arbitration mechanism picks one of them and sends its address on the output bus. This arbitration process adds extra delay and results in loss of timing precision.

Mahowald proposed the first VLSI implementation of a silicon retina that mimics the behavior of the human retina [3]. Since then various AER-based silicon retinas have been designed. Rather than sampling pixel values, AER retina pixels asynchronously output address events when they detect a significant signal [20]. In recent years, use of address-event encoding has attracted attention from various research groups working on vision sensors [10], [13], [15], [19]. Based on the arbitration mechanism used when multiple events arrive and how the address of the event location is encoded, a few address-event encoding schemes have emerged for use in such devices.

### A. Existing AER approaches

*a) Tree-based arbitration:* Proposed by Lazzaro et al. [4] and Mahowald [3], the binary tree approach is one of the earliest and most commonly used encoding scheme based on arbitration. Fig. 2(a) shows the structure of an 8-input binary tree encoder. In this scheme, incoming event requests travel through a hierarchical structure of 2-input arbiters. The arbiter at the root level picks the winning subtree, and this signal propagates back down the tree. This process continues until the input gets an acknowledgment to access the shared output bus. When multiple requests arrive in a short time window, one of the requests wins the arbitration while all other requests are queued until the winning input releases the output bus. At that point, the encoder selects another input request and grants access to the shared output bus. Since requests are queued, pending events are preserved at the expense of timing information of each event. One major drawback of this scheme is a reduction in throughput with increase in number of inputs because every request has to propagate through $\log_2(N)$ stages. This increase in delay can be significant for image sensors for two reasons. First, for applications where multiple events occur in close spatial locality i.e. when neighboring inputs request access to the output bus, the scheme performs unnecessary arbitration at each stage of the binary tree. This delay gets worse with the number of events. The second issue is that as the number of events increases, a typical AER-based readout becomes less power efficient and readout latency increases [25]. Hence, this approach is best suited for mode-S, but not mode-C or mode-F.

*b) Greedy trees:* Boahen [7] proposed the greedy tree as an improvement to the original binary tree in situations where multiple input requests arrive within a very short time period. In such cases, when one input returns access to the shared bus, access is granted to the neighboring inputs without returning it to the root. This is illustrated by curved arrows in Fig. 2(b). In the case of two simultaneous events, the token has to travel a distance that depends on the locations of the two events. In particular, the token has to travel to at least the common ancestor of the two events. If event requests are sparse, a greedy tree performs very much like the original binary tree. In order to gain benefits by sharing token with sister requests, both requests should arrive before a request for token is acknowledged by arbiter



(a) (b) (c)

Fig. 3: (a) Hierarchical token-ring encoding showing two levels of rings; (b) Ring of leaf servers (Lserver); (c) Ring of higher servers (Hserver). R, U, D, and C are used as abbreviations for Request, Up, Down, and Counter channels. ⊸ denotes the passive end of the channel and −● denotes the active end of the channel.

in higher stage. This timing requirement restricts use of greedy tree for general applications. Hence, this approach is best suited for mode-S, and for limited mode-C scenarios.

*c) Address encoding schemes:* Once an event is granted access to the shared output bus through arbitration (either by binary or greedy tree), its location is encoded in the form of an address. The address encoding in early work [3], [4] used a logarithmic encoder where each acknowledge output drives $\log(N)$ address lines. As the number of input events increases, it results in large load on the acknowledge outputs which affects overall speed and power. Georgiou [11] proposed a distributed encoding arrangement where address bits are encoded in every stage as the token moves through arbiter tree. This scheme reduced the number of transistors on each acknowledge output and improved delay on the address output.

*d) Token rings and counter-based encoding:* An address-event encoding using token-ring was presented by Imam [16] to improve performance when events arrive in bursts. As shown in Fig. 2(c), the arbitration is implemented using a token ring of mutual exclusion elements and a shared counter keeps track of the token. When a request arrives, a token circulates in the ring until it reaches the request input, and grants access to the shared output channel. The counter updates address as token moves in the ring. As a result, this design doesn't require a logarithmic encoder block for address encoding. One important benefit of this approach is that it can quickly scan through a section of the ring when a cluster of request arrives and arbitrate between inputs when requests are sparse. A major drawback of this encoding scheme is increased delay when the token has to travel long distance for each input request. Hence, this approach is well-suited for mode-F and certain mode-C scenarios when the cluster size is large.

### B. Proposed hierarchical token ring

We present a new encoding scheme based on hierarchical token-rings (HTR) which can service sparse events like a binary tree and quickly scan through a section of the array like a linear token ring. It supports imaging modes with event activities ranging from few events (mode-S) to a cluster (mode-C). The full-frame mode (mode-F) can be viewed as a really large cluster. As shown in Fig. 3(a), the design consists of multiple leaf token rings which are connected to each other through a higher level token ring. The higher level ring allows a token to quickly travel from one leaf ring to another. This can be repeated in a hierarchical fashion. Fig. 3(b) illustrates the design of a leaf ring. It consists of processes called *leaf servers* (Lserver) which receives a request from the pixel array. When a pixel wants to communicate an event, it requests access to the shared output bus. The Lserver grants access if it has the token; otherwise it requests the token from the neighboring Lserver. Similarly, the neighbor provides the token if it holds the token; otherwise it requests one from its neighbor and passes it to the Lserver requesting token. A token moves from one Lserver to another through this process, eventually leading to access to the shared output bus.

Similarly, a ring of *higher server* (Hserver) connects multiple leaf rings as illustrated in Fig. 3(c). A state variable in each Hserver indicates one of the three possible locations for the token: (i) Hserver has the token; (ii) leaf ring has the token; or (iii) Hserver and its leaf ring does not have the token. In case (iii), the token is located in one of the other Hservers or its leaf ring. When a Lserver receives request for the token and it doesn't hold the token, it sends a request to the neighboring Lserver. The request is sent to the Hserver

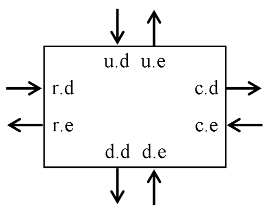


Fig. 4: Leaf server (Lserver)

if none of the Lservers in that ring have the token. Next, the Hserver provides the token if it holds it; otherwise it propagates the request to its neighboring Hserver.

Two separate counters, Lcounter and Hcounter, track the location of the token as it moves in the hierarchical rings. Hcounter tracks the location of the token in the higher ring and provides the most significant bits of the address output. Lcounter, on the other hand, tracks the location of the token within a leaf ring and provides the least significant bits of the address output. When a Hserver passes token to another Hserver, it increments the shared Hcounter to indicate which Hserver or it leaf ring has the token. Similarly, when the token passes from one Lserver to another in a leaf ring, the shared Lcounter is incremented. Since the token can be present in only one leaf ring at a time, one Lcounter can be shared by all leaf rings.

The scheme can be extended to multiple levels of hierarchy (using Hserver processes). By changing the number of processes in the rings and the number of levels of hierarchy, the design can be tailored to different application scenarios.

## IV. Circuit implementation

In this section, we present implementation details of our circuit. We start by describing the circuit functionality using Communicating Hardware Processes (CHP) and use Martin's synthesis method [1] to translate them into production rules for CMOS implementation. The non-deterministic selection are handled using a standard CMOS arbiter consisting of a latch and metastability filter [12]. For a communication channel, request signals are named with a ".d" suffix, and enable (inverted sense of the acknowledge) signals end with ".e."

### A. *Leaf server*

Leaf servers (Lserver), as shown in Fig. 3(a) & (b), are the processes at the lowest level of token-rings and serve input requests. Each Lserver communicates with the input request and neighboring Lserver above through passive communication channel $R$ (Request) and $U$ (Up) respectively. Active communication channels $D$ (Down) and $C$ (Counter) are used to communicate with Lserver below and the address counter. Wires on each communication channel are shown in Fig. 4 and its CHP description is given by:

```
$*[[\overline{R} \longrightarrow [b \longrightarrow skip [] \neg b \longrightarrow D!]; b\uparrow; R?$
 $| \overline{U} \longrightarrow [b \longrightarrow skip [] \neg b \longrightarrow D!]; C!; b\downarrow; U?$
 $]]$
```

When the server receives a communication request on channel $R$, it performs a four-phase handshake if it has the token. If it doesn't have the token, the server requests a token from the server below using a four-phase handshake on channel $D$ and updates the token variable. Similarly, when the process receives a request on channel $U$ from server above, it sends the token, updates the local token variable, and completes the four-phase handshake on channel $U$. If the process receives a request on both channels, an arbiter selects one of them. The shared address counter which keeps track of the token location is also updated before finishing the handshake on channel $U$.

The handshaking expansion used is given below:

```
$*[[r.d \longrightarrow$
     $[w \longrightarrow skip$
     $[] v \longrightarrow d.d\uparrow; [\neg d.e]; w\uparrow; d.d\downarrow; [d.e]$
     $]; r.e\downarrow; [\neg r.d]; r.e\uparrow$
 $| u.d \longrightarrow$
     $[w \longrightarrow skip$
     $[] v \longrightarrow d.d\uparrow; [\neg d.e]; w\uparrow; d.d\downarrow; [d.e]$
     $]; u.e\downarrow; c.d\uparrow; [\neg c.e]; v\uparrow; c.d\downarrow; [c.e]; [\neg u.d]; u.e\uparrow$
 $]]$
```

The dual rail variable $(w, v)$ is used to encode the $b$ variable from the CHP. This handshaking expansion is directly translated into CMOS-implementable production rules.

### B. *Higher server*

Higher server (Hserver), shown in Fig. 3 (a) & (c), are the processes in higher token rings that connect to each other as well as the lower level rings. Fig. 5 shows wires on each communication channel and its CHP is given below:

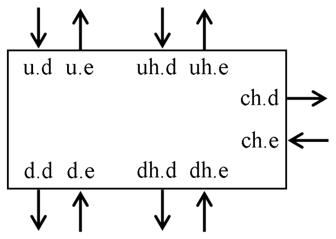


Fig. 5: Higher level server (Hserver).

```
$*[[\overline{U} \longrightarrow [b = 0 \longrightarrow skip$
              $[] b = 1 \longrightarrow D!$
              $[] b = 2 \longrightarrow DH!$
              $]; b := 1; U?$
 $| \overline{UH} \longrightarrow [b = 0 \longrightarrow skip$
              $[] b = 1 \longrightarrow D!$
              $[] b = 2 \longrightarrow DH!$
              $]; CH!; b := 2; UH?$
 $]]$
```

Similar to Lserver, a local variable ($b$) is used in each Hserver to represent token position. The variable is implemented using a one-of-three encoding with three wires, and indicates the token position as:

- b=0: Hserver has the token
- b=1: Leaf ring has the token
- b=2: Hserver and it's leaf ring doesn't have the token

A token moves in the leaf ring through channels $U$ (Up) and $D$ (Down). Channels $UH$ (Up-Hierarchical) and $DH$ (Down-Hierarchical) are used for token movement in the higher ring. Hserver tracks the location of the token. If a token request is received from either the lower or higher ring, it is either handled locally (b=0), propagated to the lower ring (b=1), or to the higher ring (b=2). The local state is updated to reflect the new token location as necessary.

A shared channel $CH$ (Counter-Hierarchical) is used for communicating with a counter representing the high order bits of the token address. Performing a full four-phase handshake on $CH$ before completing communication on $UH$ ensures that token address is updated correctly.

### C. Counter

A ripple counter is used to track the token movement and provide its address in the usual way. We design the counters using the optimized QDI circuit templates with internal state developed by Bingham [24].

## V. Results and discussion

The hierarchical token-ring encoding discussed above has been designed in a 65nm bulk CMOS process and simulated to verify correct operation. To gain better understanding, we designed existing encoding schemes and compared their performance against our design through pre-layout SPICE simulation with `Xyce`, a high-quality open-source SPICE simulator [26]. A small capacitance was added on the output of every gate to account for parasitic loads. We used the same methodology for simulating our proposed circuits as well as previous AER designs.

We considered three readout arrangements (sparse event, cluster, full frame) and evaluate performance of different address-event encoding schemes. In a sparse mode, a random request was selected from the $K$ inputs and delay from `req` to `ack` was measured for the input request. An average latency for $K$ such measurements, one from each input, was calculated. For cluster and full frame readout, we extracted different delays from SPICE simulation and computed the overall delay to keep simulation time tractable. For HTR design, we used a two-level tree and selected $\sqrt{K}$ processes in each ring for a symmetric design; this can be modified as required by the application. The delay for each encoding scheme is summarized in Table I in terms of hop-count for token movement for handling individual events. The delay numbers for each mode are estimated based on the expression for 16, 64, and 256 inputs; for each case, the measured delay from SPICE simulation presented in parenthesis which takes into account the differences in circuit complexity.

The shared counter scheme with a shared channel to increment the counter imposes overhead on token movement. This is because each process has to complete its full handshake on the counter increment channel before it can relinquish the token; otherwise, mutual exclusion on access to the increment channel would be violated.[1]

It is clear that HTR performs best when all three modes must be supported. It offers significant improvements in delay compared to the linear token ring for sparse mode but the real benefit comes with increased event activity. For a full frame readout, HTR design reduces token movement to $K + 2H$ compared to $2K * (\log_2 K - 1)$ for a binary tree. SPICE measurement for cluster and full frame mode in greedy tree were not done because the delay in such cases is highly dependent on the timing of event arrival and accurately measuring such delay depends on the response time of the pixel and its noise characteristics [25], which is beyond the scope of this paper.

Note that while the HTR scheme is slower than the tree-based approach in the case of sparse events, the performance in this case is not as critical compared to the full-frame scenario for scientific imaging applications. Most of the applications in scientific imaging are still based on capturing regions of 2D images and few applications even require hundreds to thousands of images. On the other hand, event-based applications which truly benefit from a sparse readout imaging device are very limited so far. This is one of the reason why most of the efforts are focused on improving readout speed of the frame based imaging devices. A readout scheme that offers improvements in case of the sparse and cluster readout without sacrificing performance in full frame mode would allow the scientific community to explore event-based approaches without a dedicated event-based imaging device.

In summary, the encoding scheme based on hierarchical token rings provides performance that blends the benefits of existing encoding schemes across a range of activity patterns. It handles sparse events without traveling long distances and scans neighboring events quickly in case of an event cluster.

An interesting property of hierarchical rings worth exploring in future work is that the number of cells in the leaf ring and higher rings can be changed depending on the application requirements. Further extensions can be made by increasing the levels in the hierarchy. For example, Fig. 6 illustrates how hierarchical rings can be turned into a topology that is closer to a binary tree by using only two cells in every ring. Each ring in this case functions as *lazy*

[1]We remark that a direct encoder can also be used with the token-ring scheme, but without the optimization from Georgiou [11].

TABLE I: Delay estimate for different address-event encoding. $K$ = total number of inputs, $H$ & $L$ are number of inputs in leaf & higher rings, and cluster size $M = K/2$

*Assuming the delay in moving a token to the cluster is small compared to the delay of handling events
+Delay is measured with assumption that token always starts with initial location (at Reset) and moves to one of the Lserver

| | Delay | K=16 | K=64 | K=256 |
|---|---|---|---|---|
| Sparse event+ | | | | |
| Binary tree | $2 * (\log_2 K - 1)$ | 6 (1.4 ns) | 10 (2.1 ns) | 14 (2.8 ns) |
| Greedy tree | $2 * (\log_2 K - 1)$ | 6 (1.5 ns) | 10 (2.2 ns) | 14 (3.0 ns) |
| Token-ring | $(K + 1)/2$ | 8.5 (8.2 ns) | 32.5 (34.6 ns) | 128.5 (140.2 ns) |
| Hier-ring | $(H + L)/2$ where $K = H * L$ | 4 (4.2 ns) | 8 (8.4 ns) | 16 (16.8 ns) |
| Event cluster ($K/2$ events in the middle) | | | | |
| Binary tree | $2M * (\log_2 K - 1)$ | 48 (11.2 ns) | 320 (67.2 ns) | 1,792 (358.4 ns) |
| Greedy tree | $3M - 6(M/K)$ | 21 | 93 | 381 |
| Token-ring* | $M$ | 8 (7.7 ns) | 32 (34.1 ns) | 128 (139.7 ns) |
| Hier-ring* | $M + 2(H/2)$ | 12 (12 ns) | 40 (42.6 ns) | 144 (149.6 ns) |
| Full frame | | | | |
| Binary tree | $2K * (\log_2 K - 1)$ | 96 (22.4) | 640 (134.4 ns) | 3,584 (716.8 ns) |
| Greedy tree | $3K - 6$ | 42 | 186 | 762 |
| Token-ring | $K$ | 16 (17.6 ns) | 64 (70.4 ns) | 256 (281.6 ns) |
| Hier-ring | $K + 2H$ | 24 (25.6 ns) | 80 (86.4 ns) | 288 (313.6 ns) |

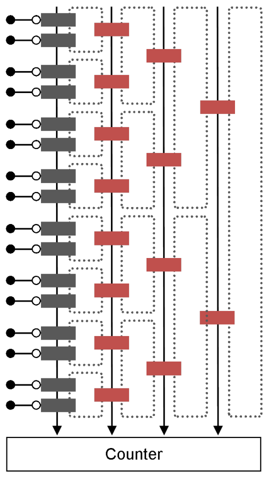


Fig. 6: Binary tree using hierarchical token-rings. $\multimap$ denotes the passive end of the channel and $-\!\bullet$ denotes the active end of the channel.

arbiter, which would hold the token unless requested by another process in the system. An additional benefit of such design is the relaxed timing requirement for the sister input compared to a greedy arbiter.

In the case of two cells per ring, the topology can be further optimized to a *lazy tree encoding*. In such encoding scheme, the nearest-neighbor ring connection is eliminated, and a state variable in each cell indicates one of the three possible location for the token: (i) left input has the token; (ii) right input has the token; or (iii) some higher level cell has the token. Input requests are forwarded to the appropriate location. A major benefit of this scheme is that token does not return to the root after handling input requests and any new local request can be serviced quickly. The distributed address encoding can still be used in this case.

## VI. Conclusion

In this paper, we discussed few scientific imaging applications where the imaging device often works under activity patterns ranging from few events per frame to reading out a full frame. Existing encoding schemes based on arbitration tree work well for sparse events but suffer from increased latency as the event rate increases. On the other hand, a scanning approach results in increased power consumption from unnecessary readout and affects temporal resolution of sparse events. Designing a custom image sensor for each type of activity rates is not sustainable due to cost and time required in design, fabrication and testing. We presented a new AER encoding scheme based on hierarchical token rings which blends the benefits of both event-based and scanning approaches. A circulating token is used to provide mutually exclusive access to the shared output bus for readout. In case of sparse events, the hierarchical structure of the ring allowed the token to bypass groups of inputs and move quickly to the destination. By using a counter to track the token location, our design naturally scales to large arrays due to reduced load from address encoding logic. Finally, a qualitative comparison for different encoding schemes shows that hierarchical token rings outperform trees in scanning mode, and simultaneously outperform token rings for sparse events.

## Appendix

The circuit functionality is described using Communicating Hardware Processes (CHP) language and key notation of the CHP syntax are summarized below:

- Skip: No operation
- Send: $X!v$ means send the value of $v$ over channel $X$.
- Receive: $X?v$ means receive a value on channel $X$ and store it in variable $v$.
- Probe: $\overline{X}$ determines if there is a pending communication on a channel $X$
- Assignment: $a := b$ means assign the value of $b$ to $a$.
- Sequential Composition: $S1; S2$ means execute statements $S1$ and $S2$ sequentially
- Parallel Composition: $S1, S2$ means execute statements $S1$ and $S2$ in parallel
- Deterministic Selection: $[G1 \rightarrow S1 \ [] \ ... \ [] \ Gn \rightarrow Sn]$ waits until one of the guards $(G1, G2...Gn)$ is *true* and then execute corresponding statement. Requires that the guards must be mutually exclusive
- Non-Deterministic Selection: $[G1 \rightarrow S1 \mid ... \mid Gn \rightarrow Sn]$ is same as the Deterministic Selection except guards don't have to be mutually exclusive
- Repetition: $*[S]$ infinitely repeats statement $S$

## References

[1] A. J. Martin, "Compiling communicating processes into delay-insensitive VLSI circuits," 1986.

[2] M. A. Sivilotti, "Wiring considerations in analog VLSI systems, with application to field-programmable networks," Ph.D. dissertation, California Institute of Technology, 1991.

[3] M. Mahowald, "VLSI analogs of neuronal visual processing: A synthesis of form and function," 1992.

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

[5] S. Dierker, R. Pindak, R. Fleming, I. Robinson, and L. Berman, "X-ray photon correlation spectroscopy study of brownian motion of gold colloids in glycerol," *Physical Review Letters*, vol. 75, no. 3, p. 449, 1995.

[6] K. A. Boahen, "Communicating neuronal ensembles between neuromorphic chips," in *Neuromorphic systems engineering*, Springer, 1998, pp. 229–259.

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

[8] E. Culurciello, R. Etienne-Cummings, and K. Boahen, "High dynamic range, arbitrated address event representation digital imager," in *ISCAS 2001. The 2001 IEEE International Symposium on Circuits and Systems (Cat. No. 01CH37196)*, IEEE, vol. 3, 2001, pp. 505–508.

[9] E. Culurciello, R. Etienne-Cummings, and K. A. Boahen, "A biomorphic digital image sensor," *IEEE Journal of Solid-State Circuits*, vol. 38, no. 2, pp. 281–294, 2003.

[10] K. A. Boahen, "A burst-mode word-serial address-event link-I: Transmitter design," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 51, no. 7, pp. 1269–1280, 2004.

[11] J. Georgiou and A. Andreou, "High-speed, address-encoding arbiter architecture," *Electronics Letters*, vol. 42, no. 3, pp. 170–171, 2006.

[12] D. J. Kinniment, *Synchronization and arbitration in digital systems*. John Wiley & Sons, 2008.

[13] P. Lichtsteiner, C. Posch, and T. Delbruck, "A $128 \times 128$ 120dB $15\mu s$ latency asynchronous temporal contrast vision sensor," *IEEE journal of solid-state circuits*, vol. 43, no. 2, pp. 566–576, 2008.

[14] T. Madden, P. Jemian, S. Narayanan, A. Sandy, M. Sikorski, M. Sprung, and J. Weizeorick, "Fpga-based compression of streaming x-ray photon correlation spectroscopy data," in *IEEE Nuclear Science Symposuim & Medical Imaging Conference*, IEEE, 2010, pp. 730–733.

[15] C. Posch, D. Matolin, and R. Wohlgenannt, "A QVGA 143 dB dynamic range frame-free PWM image sensor with lossless pixel-level video compression and time-domain CDS," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 1, pp. 259–275, 2010.

[16] N. Imam and R. Manohar, "Address-event communication using token-ring mutual exclusion," in *2011 17th IEEE International Symposium on Asynchronous Circuits and Systems*, IEEE, 2011, pp. 99–108.

[17] D. B. Carlson, J. E. Evans, and K. Maaz, "Low-dose imaging techniques for transmission electron microscopy," *The transmission electron microscope*, vol. 95, no. 3, pp. 85–98, 2012.

[18] D. Contarato, P. Denes, D. Doering, J. Joseph, and B. Krieger, "High speed, radiation hard cmos pixel sensors for transmission electron microscopy," *Physics Procedia*, vol. 37, pp. 1504–1510, 2012.

[19] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, "A $240 \times 180$ 130dB $3\mu s$ latency global shutter spatiotemporal vision sensor," *IEEE Journal of Solid-State Circuits*, vol. 49, no. 10, pp. 2333–2341, 2014.

[20] S.-C. Liu, T. Delbruck, G. Indiveri, A. Whatley, and R. Douglas, *Event-based neuromorphic systems*. John Wiley & Sons, 2014.

[21] G. M. Williams, J. Rhee, A. Lee, and S. D. Kevan, "Pixelated detector with photon address event driven time stamping and correlation," *IEEE Transactions on Nuclear Science*, vol. 61, no. 4, pp. 2323–2332, 2014.

[22] J. M. Margarit, *Low-Power CMOS Digital Pixel Imagers for High-Speed Uncooled PbSe IR Applications*. Springer, 2016.

[23] D. A. Muller, K. X. Nguyen, M. W. Tate, P. Purohit, C. Chang, M. Cao, and S. M. Gruner, "An electron microscope pixel array detector as a universal stem detector," *Microscopy and Microanalysis*, vol. 22, no. S3, pp. 478–479, 2016.

[24] N. Bingham and R. Manohar, "Qdi constant-time counters," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 1, pp. 83–91, 2018.

[25] iniVation AG, "Understanding the performance of neuromorphic event-based vision sensors," Tech. Rep., 2020. [Online]. Available: https://inivation.com/wp-content/uploads/2020/05/White-Paper-May-2020.pdf.

[26] Sandia National Laboratories, *Xyce parallel electronic simulator*, version 7.2, Nov. 2, 2020. [Online]. Available: https://xyce.sandia.gov.