Neuromorphic Chiplet Architecture for Wide Area Motion Imagery Processing

Andreas G. Andreou*, Tomas Figliolia‡, Kayode Sanni‡, Thomas S. Murray‡, Gaspar Tognetti‡, Daniel R. Mendat*, Jamal L. Molin‡, Martin Villemur‡, Philippe O. Pouliquen*, Pedro Julian‡, Ralph Etienne-Cummings* and Isidoros Doxas†
*Department of Electrical and Computer Engineering, Johns Hopkins University, Baltimore, MD 21218
Email: andreou@jhu.edu
† work performed while at BAE Systems
‡ work performed while at Johns Hopkins University

***Abstract*—We present the system architecture for real-time processing of data that originates in large format tiled imaging arrays used in wide area motion imagery ubiquitous surveillance. High performance and high throughput is achieved through approximate computing and fixed point variable precision (6 bits to 18 bits) arithmetic. The architecture implements a variety of processing algorithms in what we consider today as Third Wave AI and Machine Intelligence ranging from convolutional networks (CNNs) to linear and non-linear morphological processing, probabilistic inference using exact and approximate Bayesian methods and Deep Neural Networks based classification. The processing pipeline is implemented entirely using event based neuromorphic and stochastic computational primitives. An emulation of the system architecture demonstrated processing in real-time 160 x 120 raw pixel data running on a reconfigurable computing platform (5 Xilinx Kintex-7 FPGAs). The reconfigurable computing implementation was developed to emulate the computational structures for a 2.5D System chiplet design, that was fabricated in the 55nm GF CMOS technology. To optimize for energy efficiency of a mixed level system, a general energy aware methodology is applied through the design process at all levels from algorithms and architecture all the way down to technology and devices, while at the same time keeping the operational requirements and specifications for the task at focus.**

***Index Terms*—Chiplets, 2.5D architecture, neuromorphic processing, mixed signal design, event based processing, wide area image processing**

I. Introduction

Technological advances in microelectronics fueled by the economic growth of the semiconductor industry i.e. Moore's law [1], have enabled the wide availability of CMOS image sensors with tens and even hundreds million pixels. Every mobile device today has at least two cameras and often three or four, and today (February 2024) there are more mobile communication devices in the (8.6 Billion) than there are people [2] (7.3 Billion)!

Data generated by the billions of CMOS cell-phone cameras create a challenge in the computing systems that need to process these data [3]. Hence a new engineering field, *Big Data* (BD) science, has emerged which is aimed at sophisticated statistical techniques, algorithms and computing systems necessary to deal with such massive amounts of data. Big data science aims at developing intelligent software and hardware that can process, analyze, and distill knowledge from the vast quantities of speech, sound, video and text, ultimately with as much nuance and depth of understanding as a human would. There are three key challenging attributes of processing real-time BD: *big volume, big velocity and big variety*. The field is extensive and multidisciplinary, spanning computer science, mathematics, signal processing, and statistics.

A. Wide Area Motion Imagery (WAMI) Data

Advances in optics [4], [5] and the proliferation of cheap CMOS image sensors have enabled the creation of commercially available larger tiled image arrays such as the Kestrel and Simera [6], CorvusEye 1500 [7] and Sentinel CA-247 [8], with billions of pixels based on essentially what is cell-phone camera technology. Wide area motion imagery (WAMI) [9] from giga-pixel sensor systems is a rapidly growing data resource for civilian and defense applications (see Figure 1). These air-borne systems, aboard a moving platform such as a small plane, a UAV or an aerostat, are capable of imaging objects with a resolution of 0.2 to 0.8 meters at a distance of a few kilometers with giga-pixel image sizes and temporal resolution of a few frames per second (3 to 15 fps) [10]. Advanced imaging technologies such as analog [11]–[13] or all digital [14], [15] event based cameras can circumvent the challenges of limited frame rates but the latter have not found their way yet into WAMI systems. Hence WAMI processing pipelines rely extensively on motion dynamic information.

Availability of full motion high resolution data over large, city-size, geographical areas, (100 square kilometers) offers unprecedented capabilities for situational awareness. The dynamic nature of the imagery offers insights about actions and patterns of activities that static images do not. Civilian applications of WAMI data allow for the monitoring and intelligent control of traffic across large geographical area and inference of a hierarchy of events and activities and ultimately to "life-patterns" [16]. Additional applications include the coordination of activities in disaster areas and the monitoring of wildlife. Algorithm development for WAMI tasks is facilitated through databases such as CLIF [17] and VIVID [18] and data management standards [19].

2024 Argentine Conference on Electronics (CAE) | 979-8-3503-0508-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/CAE59785.2024.10487119

Fig. 1: Examples of wide area imagery

B. Bio-inspired event based processing

In some ways, the human brain is the ultimate machine for real-time processing. We all have an uncanny ability to remember and recall facts (*big volume*), that are stored and recalled as a result of our daily multi-sensory experience with the world (*big variety*), and are derived from signals that inundate our senses every single moment (*big velocity*). Everything is done in real-time in a resilient and robust multiprocessor architecture that is so energy efficient that it barely produces enough heat to keep itself warm in the winter. Hence we believe that a brain-inspired approach that employs unconventional processing offers an alternative paradigm for such computing task.

The approach taken here is brain-inspired because at the **level of representation**, in neural computation, the temporal dynamics of spiking neurons encode information as UNARY representation. Spikes in biology or digital events in silicon systems can encode graded (analog) signals in time while at the same time employing the robustness of binary signaling. At this level we design abstract computational structures optimized for minimum energy that exploit UNARY based representations to compute likelihoods in graphical probabilistic models of inference and cognition. Such a probabilistic event based approach that was first proposed and used in engineered systems [20] provides a principled description of event generation rules that maximize the information transfer, while limiting the number of energy expensive events (spikes) that need to be communicated between successive layers in a neural architecture. In this work we incorporate Probabilistic Address Event representation (PrAER) [21] first introduced in [20] and subsequently used in other systems [22]. PrAER is a *unary* data representation that extends the original address event representation [23]–[26]. Hence, the design methodology adopted in this paper combines and merges key ideas from bio-inspired processing with basic concepts from the field of stochastic computation [27]–[31], yielding a heterogeneous approach where unary data is represented as both pulse density modulation (PDM) or random pulse density (RPDM). The system architecture comprises of multiple computing cores on a multicast MESH AER network on chip from [32]. The stochastic processing blocks are discussed with more detail in [33] and algorithmic details of approximate probabilistic inference through sampling can be found in these two publications [34], [35]. An overview and historical perspective of AI inference including neuromorphic systems can be found here [36].

In this paper we discuss a system architecture for a real-time *high velocity* BD processing that originates in large format tiled imaging arrays used in wide area motion imagery ubiquitous surveillance. High performance and high throughput is achieved through approximate computing and fixed point arithmetic in a variable precision (6 bits to 18 bits) architecture. The architecture implements a variety of processing algorithms classes ranging from convolutional networks (ConvNets) to linear and non-linear morphological processing, probabilistic inference using exact and approximate Bayesian methods and Deep Neural Network based classification. In this shared memory architecture, hardware development is done with *energy efficiency* as the prime engineering constraint, taking lead over other considerations and the overall approach is depicted in the overview diagram Figure 2.

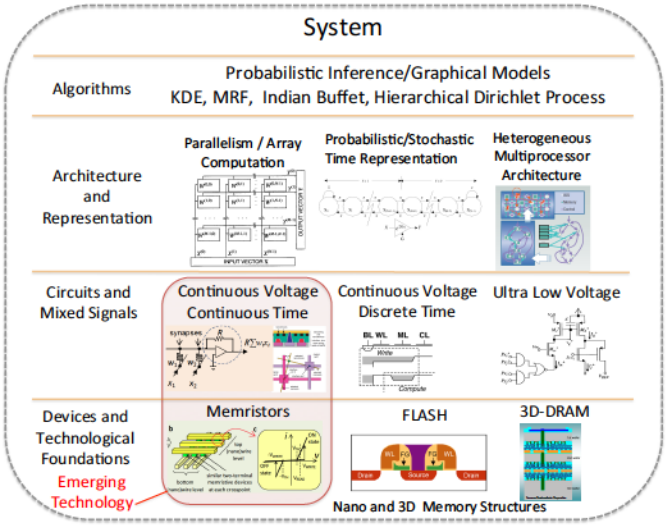

Fig. 2: The levels of abstraction in a computational system

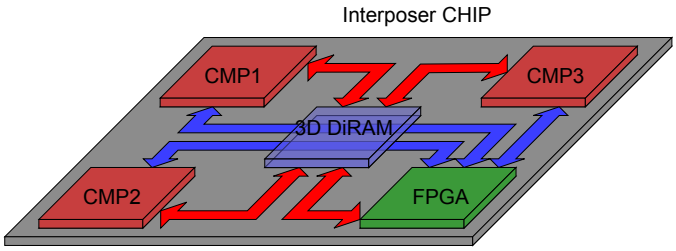

Fig. 3: **Overall Chiplet solution** with the interconnection of three of the four CMPs, an FPGA and a 3D-DiRAM stack. All of these chips will connect through an interposer fabricated in $1\mu m$ process with a size of $50mm$ by $64mm$.

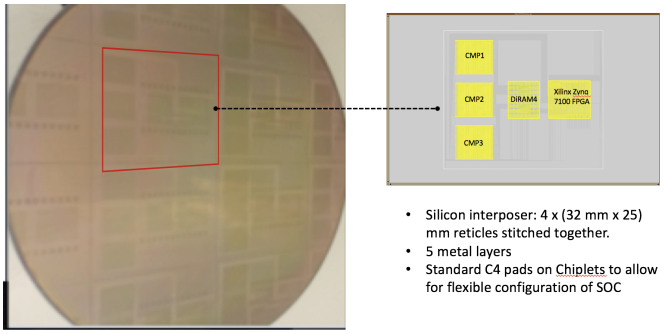

Fig. 4: **Interposer picture.** Picture of the fabricated interposer. Four $32mm$ by $25mm$ reticles stiched together, with 5 metal layers and standard C4 pads.

II. System Architecture

In this work, the design of four large chips is presented ($17.466mm$ by $14.133mm$) designed in $55nm$ GF process. Three of the four designed chip multiprocessors (CMPs) will be mounted on an interposer chip ($1\mu m$ process of size $50mm$ by $64mm$), along with a state-of-the-art die form FPGA (Xilinx Zynq 7100 FPGA) and a 3D-DiRAM memory (provided by *Tezzaron Semiconductor*). Several different options were considered for interconnecting all of the chips, but an interposer chip resulted to be the best one when power needs to be reduced as much as possible. The main advantage in building an interposer for connecting all of the units is that the capacitance of the lines in the interposer is much less than in a PCB, and additionally the overall design achieved is much more compact. This is depicted in Figure 3. Each of the four chips multiprocessor will perform processing on images of up to 400Mpixels through the usage of massively parallel processors. All of the CMPs will have access to an on-interposer 3D-DiRAM memory stack and additionally will communicate with an on-interposer FPGA. Each of these CMPs is composed of either 64 or 128 processing units (PUs). With the selection of different types of PUs on each chip, different image-processing flows can be achieved, and it is for this reason that the choice of these different PUs is desired to be something that can be easily changed without having to start the design of each chip all over again. The different types of PUs will be addressed in one of the lasts chapters, because there is no need in knowing what the PUs will do at this point. A picture of the interposer is shown in Figure 4.

In the designed chips modularity was exploited as much as possible, with the main objective of easing the task of assembling the final four chip designs. If no consideration is given to the content of each of the processing units on each chip, only on core chiplet design with 128 PUs. The chiplet with comprises of eight rows of 16 PUs each. The whole point in building modular chips using chiplets is that major changes can be applied to it without spending any additional time redesigning everything.

For the communication in between nodes a buffer-less mesh network was designed. This network will be called the *L2 network* (L2 stands for level two), and it has very particular and convenient characteristics that will be addressed in a later section. If a standard interface from the *L2 network* node to each processing that does not rely on any particular clock (asynchronous interface), then the top level design of the 128 PUs CMPs can be completely abstracted from the content of each of the PUs. Each of the PUs can have its own clock tree completely independent from any other clock in the system. This is the reason why a four phase handshaking interface was designed for the communication of each PU with the *L2 network*. This allows to place "dummy" PUs in the core chiplet, and replace them later with the final desired PUs. The *L2 network* can be seen in Figure 5 . The connection to the FPGA for the 128 PU chiplet is done through network nodes, $(1, 0)$. The communication between the FPGA and its network node uses the same protocol any of the PUs uses with its own node, with the difference that serializers and deserializers needed to be used for the FPGA due to the extremely wide network bus, which is over 300 bits. Additionally, so that throughput to and from the FPGA could be increased, bidirectional pads were used for the communication in between the *L2 network* and the FPGA.

Access to DDR memory is granted to each of the PU by incorporating an additional network. This network will be called *L1 network*, and it will facilitate communication for each of the PUs with the DDR memory through the *DDR DRAM PHY* block. This block translates read and write requests from the PUs to the 3D-DiRAM memory. This network is formed by independent token-ring networks on each of the rows in both designs. Each of these token-ring networks will have a dedicated *DDR DRAM PHY* port. A total of eight different token-ring networks will be present for this *L1 network*, and each PU will communicate with its *L1 network* again through a four phase handshake interface. The *L1 network* can be seen in Figure 6.

When communicating outside of the chips, it can either be done through the DDR memory or through the *L2 network* connecting to the on-interposer FPGA (see Figure 3). The *L2 network* will have an additional node, apart from the 128 previously mentioned nodes. This node has access only to the *L2 network*, and the processing unit assigned to this

979-8-3503-0508-1/24/$31.00 ©2024 IEEE

IEEE Catalog Number CFP24S41-ART

node is the external FPGA to which the *L2 network* connects through the left side pads of the chip. The FPGA is able to send and receive packets to and from the *L2 network* using the four phase handshaking protocol, and also has its own address, making the communication between FPGA and PUs completely transparent. The utilization of this asynchronous protocol in communicating the FPGA with the *L2 network* is very convenient as it does not require the equalization of any of the data bits lines with respect to a received clock.

Each of the CMP chiplets is $17466\mu m$ by $14133\mu m$ in size. Because of these large dimensions, it is impossible to expect the *Place & Route* tool to create clock trees with very low skew and slew. It is for this reason that a custom architecture was designed for the clock trees in the design. Long clock tree cells of size $\approx 1500\mu m$ by $\approx 50\mu m$ were designed. These cells take a clock input and generate several clock outputs along one or both long sides, with a skew of only 30ps, allowing clock speeds of up to 1.25GHz to be propagated through these cells. These cells allow clock trees to be built local to the outputs of these clock tree cells, making these clock trees much smaller and more reliable. In Figure 5 and 6 the different clock cells that allow both networks to be completely in sync can be seen. Similar cells were used for distributing asynchronous reset to the network.

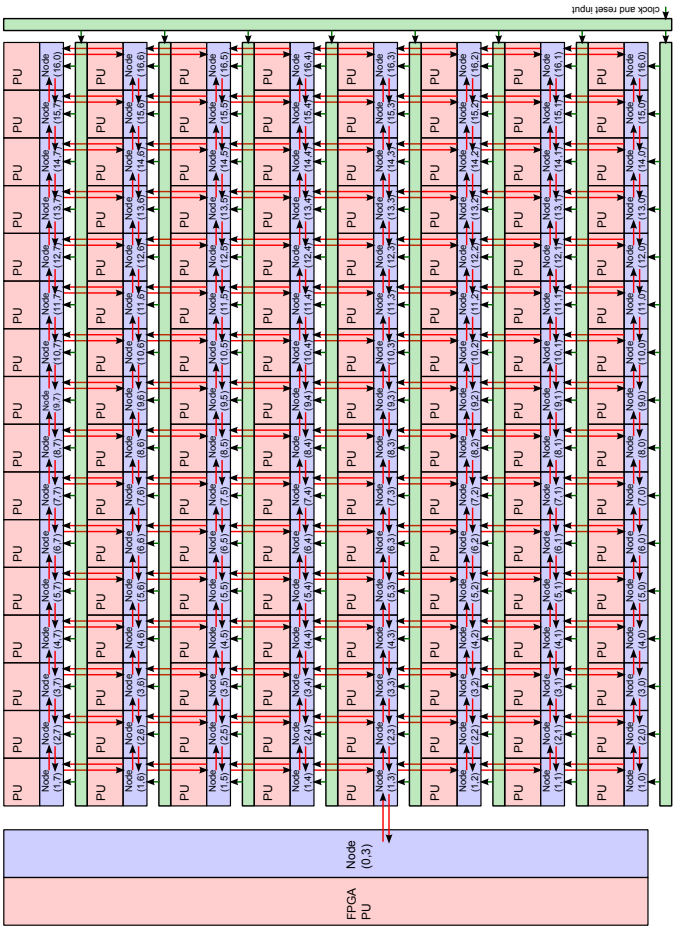

Fig. 5: ***L2 network* for the 128 PUs chiplet.** Communication to the FPGA is done through the $(1,3)$ node. The communication between node $(0,3)$ and the FPGA is done through bidirectional pads placed on the left of the chip. Each of the packets in the network contains 256 bits of data, making it really difficult to have that same number of pads in that communication. A serializer and deserializer are being used to send and receive between the *N2 network* and the FPGA. The green blocks distribute reset and clock signals.

III. Data Representation

One key innovation in the architecture described in this paper is the adoption of an internally heterogeneous representations for the mixed signal computational structures. These can be currents, voltages, charges or more formally the the signal representations shown in the panel Figure 7. The first two correspond to Continuous Value Continuous Time (CVCT) and Continuous Value Discrete Time (CVDT) and are often termed "analog". These include charge injection devices (CID), charge coupled devices (CCD) or switched capacitors (SC). The third is Discrete Value Discrete Time (DVDT) or what is known as digital and employed in Boolean computations. The last two panels depict the fourth less known signal representation for computational structures, the Discrete Value Continuous Time (DVCT) representation.

In the communication literature DVCT is known as pulse time modulation (PTM). PTM has become popular in the field optical communication because it is simple to implement, requires no digital codes, and the pulse format of the modulated carrier makes the scheme immune to channel nonlinearity. **We will rely on the temporal representation of information, such as PTM for implementing arithmetic functions and memory in the design of this system because this allows to attain accuracy and dynamic range in the sub-micron and deep-submicron CMOS technologies.** The switching speeds of the devices in the deep-submicron technologies are in the in the sub-20 ps range, hence investigating unconventional computational structures that exploit such temporal resolution is exciting. PTM techniques can be divided into two main classes. Isochronous PTM techniques, such as pulse width modulation (PWM), pulse position modulation (PPM) and pulse density or pulse frequency modulation (PDM or PFM), carry information in some characteristic of the pulse with respect to a predetermined time frame. In anisochronous PTM (APTM) techniques, there is no time-frame but pulses can occur in continuous time. Neural spikes correspond to anisochronous PFM. The Address Event Representation (AER) employed in this project is also an APTM technique, where each event encodes the time of the event and the address of the encoding processing node, so in this instance the event is not a single pulse (one bit) but rather a small numbers of bits encoding . Stochastic pulse time computation (SPTM) encodes the stochastic sampling of a probability density function of a signal or data at the originating node and is a key representation employed in the architecture. SPTM allows the processing of signals are low signal to noise

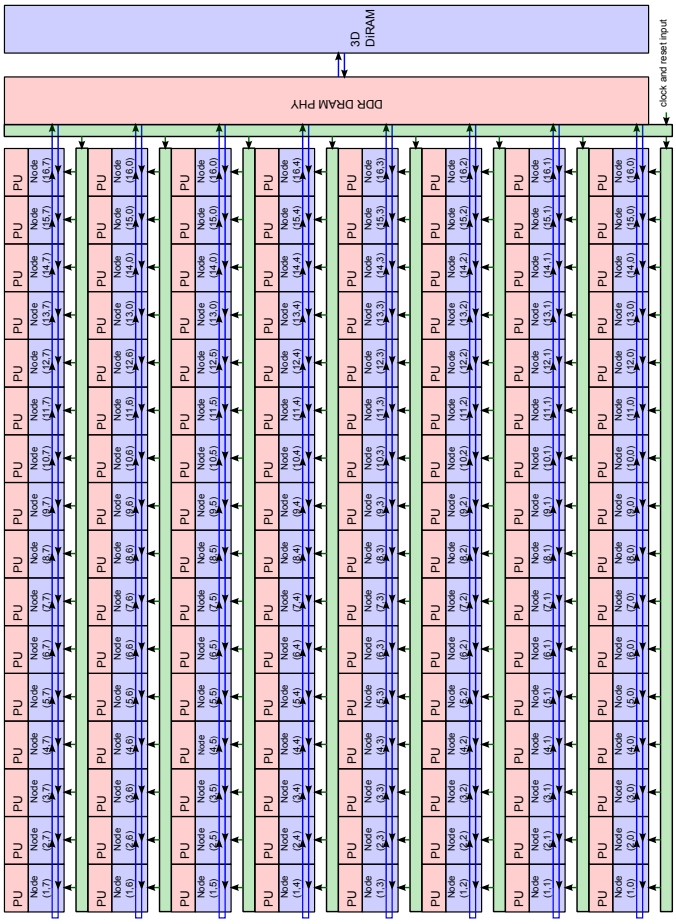

Fig. 6: ***L1 network* for the 128 PUs chiplet.** Eight different token-ring networks communicate with the *DDR DRAM PHY*. The communication between the *DDR DRAM PHY* and the DDR is done through two DDR buses, where each bus is composed of 66 signals, 64 data signals and two complementary clocks. The required pads communicating with the DDR memory are placed on the right side of the chips.

Time / Value	Discrete	Continuous
Continuous	**CVDT** CCD Switched cap PAM	**CVCT** Linear analog
Discrete	**DVDT** Binary digital Multivalue digital Isochronous PTM	**DVCT** Asynchronous digital Anisochronous PTM

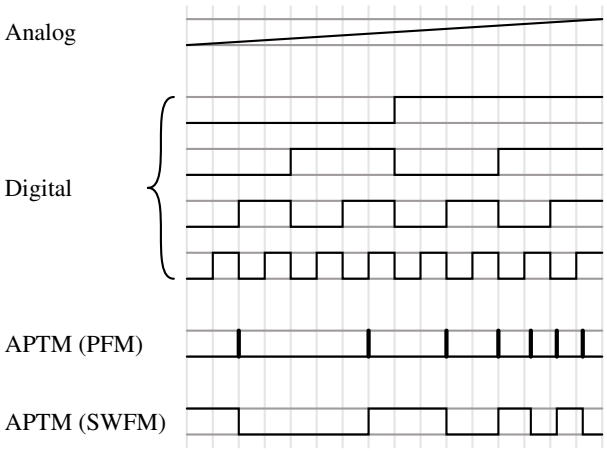

Fig. 7: (Left) Classification of circuits in terms of signal representations; (right) graphical description of the representation as a function of time.

ratio and hence with high energy efficiency. Finally, pulse code modulation (PCM) is the traditional encoding of digital information as a series of ordered pulses/events in the time series and since this represents the traditional approach of digital computing will not be further discussed here.

There four PTM data encodings that are employed. These encodings represent analog information as digital data on the time axis. Let's us define a frame T_F and the minimum attainable temporal resolution in a given technology T_R. Let us also define the pulse width T_W minimum pulse width T_{MIN}. In the isochronous or anisochronous and random pulse density modulation (PDM) , (APDM), (RPDM) if a wire must represent the probability of a variable A taking a value of one i.e. $P(A) = 1$ is encoded by N events/pulses such that $N = T_F/T_{MIN}$. Similarly the probability of a value equal to the least significant bit i.e. $P(A) = LSB$ is represented by just a single event/pulse within the frame T_F. In the pulse width modulation (PWM), the probability of $P(A) = 1$ corresponds to a pulse of width $T_W = T_F$ and $P(A) = LSB$ is represented by a pulse width $T_W = T_{MIN}$. In addition to regular numbers, PTM representations enable the representation of probabilities, the currency at the heart of the inference algorithms.

IV. Algorithms and Mixed-Signal Circuits Co-design

A reconfigurable computing based, processing pipeline architecture (Figure 8) was developed to emulate the computational structures for a 2.5D system that was fabricated in the 55nm CMOS technology. Mapping the algorithms on a reconfigurable computing platform has a dual goal: (i) algorithm exploration and (ii) architecture exploration. The processing flow begins with raw pixel values from a camera and implements de-Bayer interpolation, non-uniformity correction, camera motion compensation, background/foreground segmentation, object attributes extraction, object tracking and object classification. The processing pipeline is implemented entirely using event based neuromorphic and stochastic computational primitives. The system is capable of processing in real-time 160 x 120 raw pixel data running on a reconfigurable computing platform (5 Xilinx Kintex-7 FPGAs).

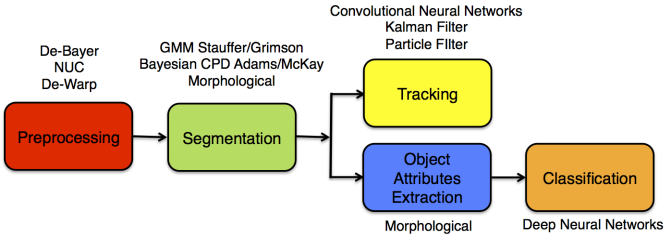

Fig. 8: Processing pipeline.

As a way of example to the structure of the computations, the following three sub-sections provide details for three specific components of the processing pipeline: the De-Bayer transformation, the change point detection and the morphological processing (median filtering/connected component analysis) that removes the noise after the change point detection step in the segmentation process. Unfortunately space limitations do not allow the full discussion of the algorithms and implementation; references in the bibliography and forthcoming publications will detail the mapping of the algorithm into the architecture. In WAMI data streams the spatial resolution of the imagery is relatively poor hence limiting the usefulness of appearance based object detectors. A typical object such as a car or a truck corresponds to only a few pixels (10-100). The motion of such relatively small size objects means that objects can move over large distances with current state of the art cameras that often have only 2-3 frames per second temporal resolution (see the large truck in the four frames sequence, bottom frame in Figure 1). The adopted motion imagery processing pipeline shown in Figure 8.

1) De-Bayer transformation (Pre-processing): The de-Bayer algorithm employed here is chosen for simplicity and to demonstrate the validity and performance of the adopted data representation (RPDM) and the computational structures (multiplexes and counters) that implement the multiplication and summing operations. The algorithm is a simple linear interpolation as discussed in Section 2.1 of the overview paper by Mashal et. al. [37]. Figure 9 (top) shows the 8 bit de-Bayered image processed using regular arithmetic. The image processed using the probabilistic event base computational structures that emulate the hardware are shown in Figure 9 (middle) for 256 events/time slots i.e. 8 bit and in Figure 9 (bottom) for 32 events/time slots i.e 5 bit. The computations for the de-Bayered processing were done with mixed signal vector-vector multiplier/processing blocks. Five generations of test chips where fabricated and tested with experimental results for mixed signal vector-vector multiplication that include power dissipation and energy efficiency can be found in Technical Reports #58 and #59.

2) On-line Change Point Detection (Segmentation): Change point analysis (CPA) also known as change point detection (CPD) is the identification of sudden and often small changes to the parameters or the output of a system that is in the form of sequential data. Often CPA is employed for the segmentation of a signal to facilitate the process of tracking,

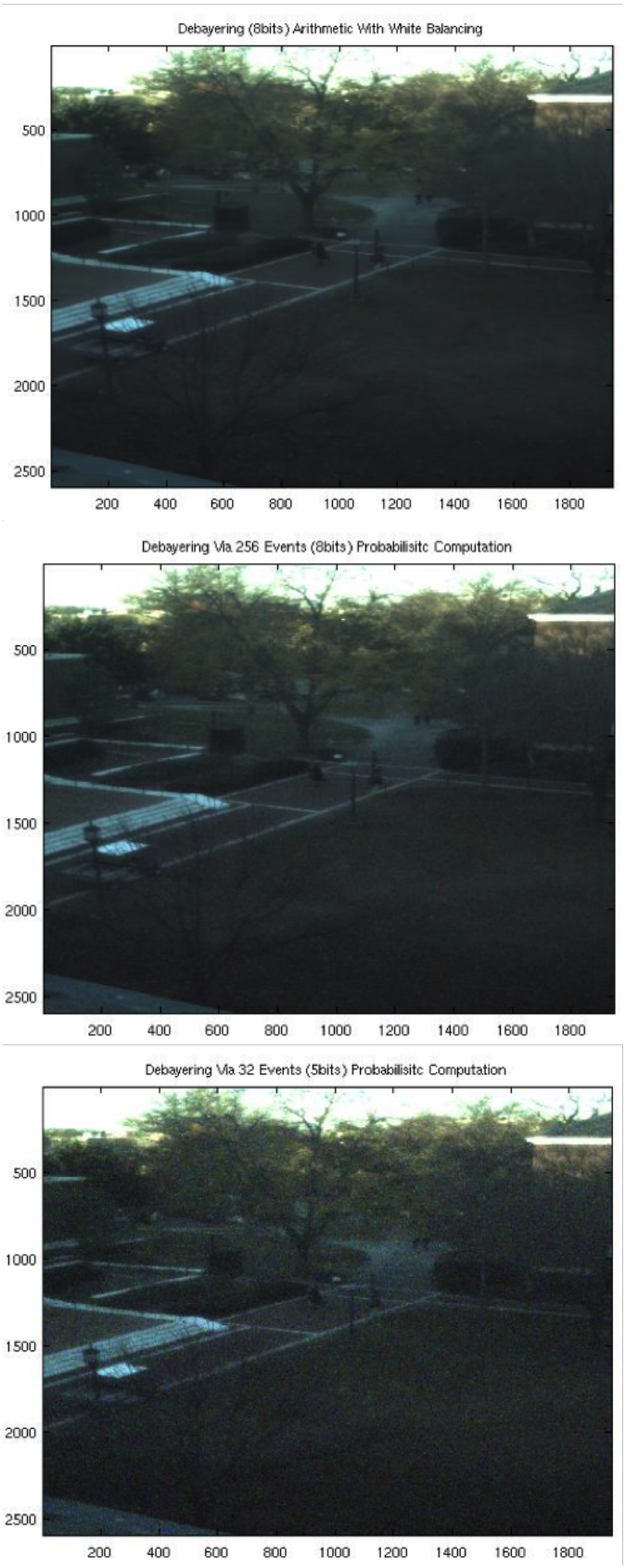

Fig. 9: De-Bayered color image: regular 8 bit arithmetic (top), 8 bit probabilistic (middle), 5 bit probabilistic (bottom)

TABLE I: Algorithms in the image processing pipeline.

Stage	Algorithm
Preprosessing	De-Bayer [37] (3x3 interpolation) Non-uniformity correction (8bit offset and gain correction) De-Warp [38] (bilinear transformation with 3x3 interpolation for address re-mapping
Segmentation	Stauffer / Grimson [39] (Gaussian mixture models) Change Point Detection [40] (exact Bayesian inference) Morphological processing [41] (connected component, median filter)
Tracking	Mean-shift [42] / Convolutional Network on binary blobs Kalman Filter [43]
Object Atributes	Morphological processing [41] (blob attributes, geometry)
Classification	Convolutional [44] / Deep Neural Networks

identification or recognition. Here we use the algorithm by Adams/McKay for online Change Point Detection [40].

The algorithm is fully implemented as VHDL code in the reconfigurable computer architecture. To demonstrate the validity of emulation in VHDL towards a full custom ASIC CMOS implementation we show here results from a test chip. The test chip architecture employs probabilistic event representation and computational structures that natively operate on probabilities. A fully programmable quad core CPD processor was synthesized from VHDL in 55nm CMOS Global Foundries technology and is tested as fully functional. All of the cores in the chip can be programmed, but they are all programmed in the same way, meaning that all of their parameters have to be the same for all of them. The random number generators used in the chip are programmable so that the LFSRs used can change their period depending on the computational time required, demonstrating the "precision on demand" capabilities in the architecture. The maximum number of bits in our LFSRs is 20 bits, and with this structure LFSRs of period from $2^{20}-1$ to $2^{3}-1$ can be accomplished. The CPD processor occupies an area of approximately 1mm x 1mm. The chips were mounted on a custom design board that communicates with an OpalKelly board featuring a Xilinx Spartan3 FPGA. The communication between the FPGA and the chip was done through two high-density connectors. All the signals driving the chip were provided by the FPGA, even the clock signal, giving the versatility of programming the chip clock frequency from the computer.

A Matlab program and VHDL code was written for testing the chip. This program, depending on the statistical parameters of the signals that need to be analyzed, calculates all the parameters for the CPD algorithm. Those parameters are translated into values that are sent to the chip to program the behavior of the processors. Figure 10 shows the results from the chips. All the four cores function properly, showing results comparable to the floating point implementation of the algorithm in Matlab.

3) Morphological Processing (Feature Extraction): The morphological processor is based on the simplicial [41], [45], [46] cellular neural network architecture and consists of three different modules: the cell array, the control module, and the function memory. The cell array contains a 64x64 dimensional array of units that need to access memory to perform a nonlinear programmable function. Each cell is interconnected

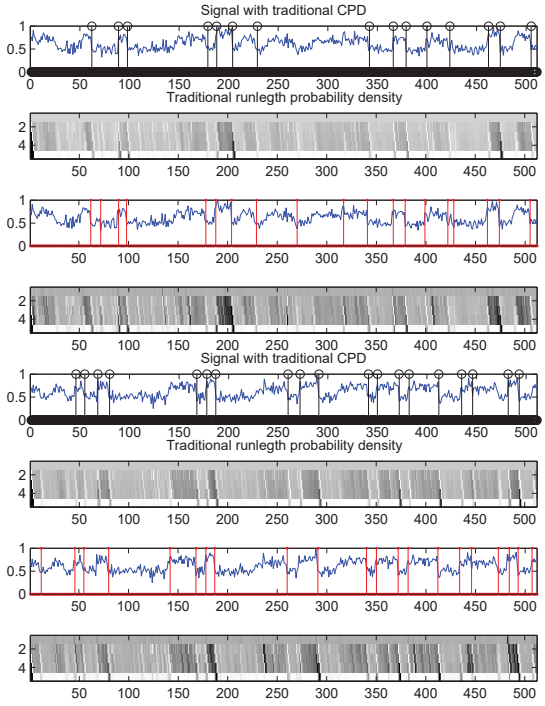

Fig. 10: Results from the CPD test chip compared with the implementation of the CPD algorithm in Matlab. All four cores on the chip are fully functional and operate at the simulated frequency (only data from two cores are shown here).

with five neighboring cells (that can be configured in an "x" or "+" configuration) and has two registers for storage namely, X and U, two registers for intermediate computation results, a 1-bit ALU (that can implement the following functions: and, or, nor, xor), two 5-bit equal comparator and several multiplexers. The Morphological Processor operation can be programmed to execute different programs depending on the memory content. In the application under study, the objective is to take an input frame produced by the Change Point Detection (CPD) block and apply image processing to determine whether there is an object in the scene. The output produced by the CPD is a frame where every pixel is represented by 1 bit. An example of a "program" with basic instructions employed to cleanup noisy binary data coming from the CPD processor is: *dilation, erosion, erosion, erosion, dilation, dilation, median*. Figure 11

shows one of the frames produced by the CPD and the output of the sequence of operations performed by the Morphological Processor.

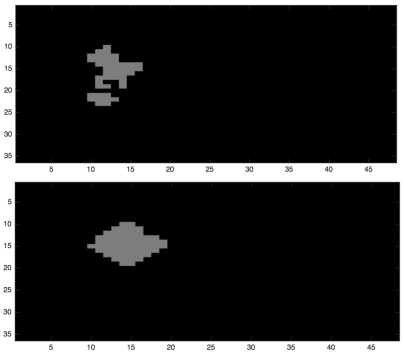

Fig. 11: (top) Frame produced by the CPD; (bottom) Output of the Morphological Processor.

A screen shot of results from the image processing pipeline up to the segmentation and feature extraction stage is shown in Figure 13. The system architecture outlined in this report has demonstrated real-time processing of 160 x 120 raw pixel data running on a reconfigurable computing platform (5 Xilinx Kintex-7 FPGAs). The image data is generated by standard cameras and hence the value of the pixels is encoded using pulse density modulation (PDM) so that it is compatible with the computational infrastructure.

V. Design for Extreme Energy Awareness

An energy aware and constraint design methodology must be applied at all levels of the system Figure 2, algorithmic, architectural, circuit and device. Techniques and improvements at the highest level of abstraction such as algorithmic and architectural will likely to produce energy efficient designs under the strict constraint budget while meeting the system level performance requirements. We begin with a simple calculation of how much is a pico-Joule. **What is a pico-Joule (pJ)?** A pico Joule is the energy to pass one pico ampere of current (10^{-12}A) in a circuit operated at 1Volt for a duration of 1 second. Alternatively, a pico Joule is the energy to charge a 1 pF (10^{-12}F) capacitor up and down an potential of 1Volt.

The energetic requirements for the computational structures has yielded important insights on the research and development for this project. At every level of abstraction in the Figure 2 there are methods to reduce power that have ramifications to the other levels. More important, any decisions and optimizations done at a given level must be consistent with the adjacent levels. The general design principles to yield an energy aware design are summarized here going from the lowest level of abstraction to the highest.

A. Technology/Device Level

1) The overall design methodology in the project is constraint by the technology, i.e. as technology scales there are severe limitations on the power supply voltage. Typical power supplies can not exceed 1.8 Volts. It is possible that a non state of the art process is employed that has high voltage devices that can operate at 2.5 Volts but this is not an option available to this project.
2) Modern fabrication processes, offer devices that have multiple threshold voltages (typically three) and multiple gate oxide thickness (typical two). These devices can be employed judiciously in the circuits to improve performance and meet energy efficiency requirements.
3) Noise (shot-noise) power and bandwidth (parasitic capacitances at nodes) impose fundamental constraints on the precision (in bits) available at each node in the circuits.

B. Circuits

1) Traditional wisdom suggests that the most effective way to operate with maximum efficiency is to reduce the power supply voltage and operate in sub threshold or near threshold, for both digital (DVDT) and non-digital circuits.
2) All digital (DVDT) computational blocks will be designed using an ultra low voltage design methodology with transistors operating in the sub threshold or near voltage threshold (NVT). A CMOS library was specifically designed to address the necessity for NVT operation.
3) All non-digital computational blocks will also be designed with devices operating in sub-threshold region and deep sub-threshold region with current levels in the pA range and 100fA range. This has ramifications on the bandwidth available which is going to be only a few 100Hz.
4) **The circuit design methodology that has been adopted in this project focuses on pulse time modulation (PTM) signal representations, with computations in what is traditionally known as "analog" (CVCT) being adjunct to the PTM based computations**. This is because with the power supply of 1.8 Volts available in the fabrication technology energy efficient circuits in the traditional view point (CVCT) are problematic.

C. Architecture and Representation

1) At the architecture and representation levels, we make the ultimate decisions as to which aspects of the processing will employ the particular representation i.e. CVCT (analog), DVDT (digital), as well as CVDT and DVCT (APTM).
2) The key architectural decision at the architecture level is the adoption of massive parallelism and pipelining in all signal representations. This allows for operating the circuits at lower speeds. Lower speed in the digital components of the system translates to lower voltages, that yield quadratic improvements in energy. Lower speed in the analog computational structures translates to reduced bandwidth and reduced integrated noise power.

979-8-3503-0508-1/24/$31.00 ©2024 IEEE

IEEE Catalog Number CFP24S41-ART

D. Algorithms

1) Any algorithm that will be developed needs to be compatible with the low precision hardware (4 to 8 bits) available in the non-digital (non-DVDT) circuits and something like 8-16 bit fixed point available in digital (DVDT) ultra-low voltage circuits. **It necessitates a complete re-thinking of the algorithm so that it will enable both parallelism and low precision i.e. approximate computation.**
2) At the algorithmic level we will also incorporate prior knowledge about the statistics of the input signal. This ultimately determines the ultimate encoding to minimize unnecessary switching activity for the digital part of the system or bias currents for the analog sub-systems.

E. System

1) A detail and careful analysis of the energetics in processing requirements of the project has revealed some rather surprising findings. **The actual energy cost of computation is relatively small as compared to the cost of the interfaces to the non-digital computational structures, and communications, i.e. moving data from one place to the other so that computations can be carried out**.
2) At the system level we have made the important decisions to incorporate the memory hierarchy as part of the integrated 2.5 interposer and 3D memory stack.
3) The clock distribution network and local generation of random numbers/variables is also part of the system level. Furthermore, the adoption of an APTM technique allows for the local generation of clocks without skew concerns that tend to add complexity to the circuit design of the clock distribution network in traditional digital computing systems, with associated energy costs.

VI. Physical Design Methodology for Mixed-Signal CMPs

When designing a system on chip, different approaches can be taken. One can perform flat *logical synthesis* and *Place & Route*, or a more modular bottom up approach can be taken. The term *logical synthesis* will be given to the translation of hardware description written in VHDL or Verilog, into a netlist of logical cells belonging to a particular standard cell library. On the other hand *Place & Route* tools will take that translation from the *logical synthesis*, and will perform the actual physical implementation of that netlist, laying down the actual layout for every single cell and performing the corresponding metal connections. For small areas, usually flat *Place & Route* results in a more efficient outcome regarding power, area and timing, mainly because the *Place & Route* tools used are provided with all of the degrees of freedom for the considered design. When bottom up approaches are used, at every level of hierarchy considered, the degrees of freedom are reduced, and then the result obtained might not be optimum. In the flat *Place & Route*, logic that might be repeated several times in different modules, could be just collapsed into a single unit, resulting in area reduction and less power dissipation. So, when would a bottom up approach be the answer? As technology advances, chip areas are increased, and feature sizes decreased, resulting in very large relative areas, for which the complexity and time used performing *Place & Route* increases exponentially. Sometimes the time span used in performing Place & Route very large designs could be weeks, and it is just with a simple modification to the architecture, that this process needs to be restarted. It is for this reason that modular designs are becoming more attractive, allowing to keep complexity limited at every level of hierarchy.

Bottom up approaches face challenges that flat designs do not. A wider understanding of the involved physical design is required. Power distribution and timing becomes more challenging as these aspects need to be analyzed individually for each block, and eventually their impact in an upper level of hierarchy. Modular designs additionally necessitate the specification of additional constraints, such as the timing requirements in the signals connecting each block to a top level. Dimensions need to be specified for each block. Shape and size for hierarchical blocks need to be carefully chosen when the broader picture is considered. The position of pins on each block becomes an important issue. A poor choice for their position might result in the passing of timing constraints locally to the block, but the violation of them when timing is analyzed in the upper level of hierarchy. Due to the complexity of the designs presented in this work, modular design is a must. Up to six levels of hierarchy were used in some of the blocks that compose the solution that will be proposed, making proper physical planning necessary. Fortunately, the choice of this physical planning path allowed to reduce the time performing *Place & Route* of the top level design for the different chips designed in this project to under a day.

Concepts such as tiling, block abutment and module repetition start showing up in large designs. In the face of tight timing constraints, abutment of blocks becomes a necessity, and with it, proper input/output delays need to be correctly equalized as well as the position of the pins connecting all of the abutting blocks. When having two blocks utilizing the same clock signal, one block might send data to the other block, and the later block might send data back as well. Let's consider the case of block A sending a bit of information to block B. The *input delay* constraint for block B is the time the bit at the output of block A takes to travel from the closest register in block A to the physical block pin. On the other hand the *output delay* for block A is the time that the before-mentioned bit travels from the input pin in block B to the closest register's input still in block B. These timings are shown in Figure 12. If the summation of the input and output delays in the abutting blocks is not less than the desired clock period, then no optimization in the upper level of hierarchy will ever allow that clock frequency to be achievable. It is for this reason that the choice of input and output delays in a block is of the highest importance. In the design proposed, the existence of several processing units (PUs) will be shown. These units will need to abut with each

other to create a compact overall design, where analysis of these timing constraints need to take place. When performing this abutment additional safe measures need to take place when designing these units, such as *Place & Route* blockages that will prevent any design rule violation, or cross-talk between internal block signals at the place where blocks abut.

An additional concept used in large designs is module repetition. Repetition of modules not only allow to have a more consistent timing and power distribution over the chip, but it also simplifies the synthesis of the whole chip. This is the case of the two networks on chip that have been implemented in the design of the chips presented in this work, where each node of these networks is a block that is repeated all over.

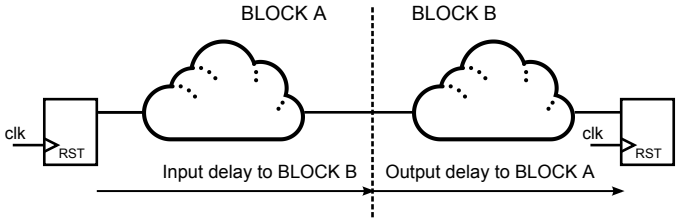

Fig. 12: **Input/Output Delays in a synthesized block** Two different delays need to be taken into account when connecting two blocks that are abutting is shown. The summation of both delays shown need to be less than the desired clock period.

When designing large chips, problems such as clock tree integrity, exponentially increasing time for synthesizing, placing and routing designs, and difficulty in performing minor changes to the design, become more problematic. With very long distances for a clock signal to travel, mismatch and variations along the die will make it very difficult for a clock tree to achieve the desired skew, slew and speed. Other options such as building H-trees with very strong drivers will work, but most likely a modular design will find this alternative very difficult to deal with, because of the specific places the clock drivers need to be placed. Tools such as *Cadence Innovus* will take an amount of time in performing *Place & Route* on a design that increases exponentially with the size of the design. Consequently if modularity is not exploited enough, one can be found in a situation where re-synthesizing projects could take days to *Place & Route*, probably because of just a minor change to the design. If on the other hand one can exploit as much as possible the bottom up approach synthesis, taking advantage of repetition in the design, the time spent could be reduced significantly. All of the CMPs were designed in this way, breaking the design into smaller and more approachable problems. The only disadvantage to this approach is that one has to have a very clear picture of the full chip layout, specially when talking about power planning.

A highly conductive power grid was designed on top of all of the chiplets. These grids hold the different power supplies for different voltage domains across the chip and additionally supply external and locally generated biases that are made available to all of the processing units. The locally generated biases are generated by a local band gap reference that will be placed as one of the PUs. 16 are the total number of external biases and 16 is also the total number of locally generated biases. These power grids were designed in metal 7 and 8, allowing the design of each processing unit to span from metal 1 to metal 6. If a power supply or bias is locally required in a processing unit, a simple connection to metal 7 can be generated. It is for this reason that it was decided that providing a template with the exact position of the power grid and standard pins connecting the PU to both networks was the way to go for everybody designing PUs. This template would ensure compatibility when placing or replacing PUs in the network. Additionally, the clock provided to the PU from the network node is a programmable one, so if more than one PU slot was needed for a particular design, as long as the different clocks provided to each of the PU slots are configured to have the same frequency, these clocks would actually match also in phase. This characteristic would allow local clock trees to have more than one root, making local trees have a reduction in their depth, allowing better reliability. This means that, the person designing that multi-slot PU can rely on several clock inputs that are in phase, reducing the complexity of the local clock trees.

VII. Conclusions

We have described a chiplet based system to do real-time *big velocity BD* processing that originates in large format tiled imaging arrays used in wide area motion imagery ubiquitous surveillance. High performance and high throughput is achieved through approximate computing and fixed point arithmetic in a variable precision (6 bits to 18 bits) architecture. The architecture implements a variety of processing algorithms in what we consider today as **Third Wave AI and Machine Intelligence** ranging from convolutional networks (ConvNets) to linear and non-linear morphological processing, probabilistic inference using exact and approximate Bayesian methods and Deep Neural Network based classification. The processing pipeline is implemented entirely using event based neuromorphic and stochastic computational primitives. An emulation of the system architecture demonstrated processing in real-time 160 x 120 raw pixel data running on a reconfigurable computing platform (5 Xilinx Kintex-7 FPGAs). The reconfigurable computing implementation was developed to emulate the computational structures for a 2.5D system the *nano-Abacus* with chiplets fabricated in the 55nm GF CMOS technology.

References

[1] G. E. Moore, “Cramming more components onto integrated circuits,” *Electronics*, vol. 38, no. 4, pp. 114–117, 1965.
[2] US-Census-Bureau. (2016, Feb.) Current Population. [Online]. Available: http://www.census.gov/popclock/world
[3] A. Szalay and J. Gray, “2020 computing: science in an exponential world,” *Nature*, vol. 440, no. 7083, pp. 413–414, Mar. 2006.
[4] D. J. Brady, M. E. Gehm, R. A. Stack, D. L. Marks, D. S. Kittle, D. R. Golish, E. M. Vera, and S. D. Feller, “Multiscale gigapixel photography,” *Nature*, vol. 486, no. 7403, pp. 386–389, June 2012.
[5] D. Brady. (2014, Feb.) AWARE2 Multiscale Gigapixel Camera. [Online]. Available: http://disp.duke.edu/projects/AWARE/

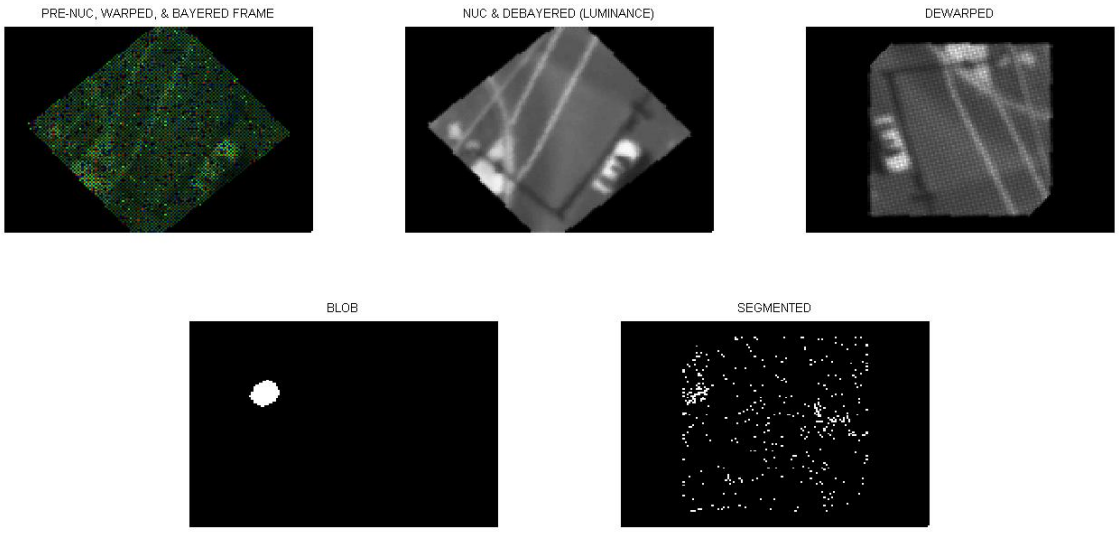

Fig. 13: Screen capture of the laptop displaying the results from image processing pipeline.

[6] Logos-Technologies, "Multi-Sensor, Wide-Area Persistent Surveillance," pp. 1–2, Sept. 2015.

[7] Harris-Corporation, "CorvusEye 1500," pp. 1–4, 2015.

[8] UTC-AerospaceSystems, "ISR Systems," pp. 1–7, Nov. 2015.

[9] R. Porter, A. Fraser, and D. Hush, "Wide-Area Motion Imagery," *IEEE Signal Processing Magazine*, vol. 27, no. 5, pp. 56–65, Sept. 2010.

[10] Logos-Technologies, "Simera: Lightweight, Wide-Area Persistent Surveillance Sensor for Aerostats," pp. 1–2, Sept. 2015.

[11] 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, Feb. 2003.

[12] E. Culurciello and A. G. Andreou, "CMOS image sensors for sensor networks," *Analog Integrated Circuits and Signal Processing*, vol. 49, no. 1, pp. 39–51, Oct. 2006.

[13] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128x128 120dB 15us Latency Asynchronous Temporal Contrast Vision Sensor," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, 2008.

[14] C. G. Rizk, P. O. Pouliquen, and A. G. Andreou, "Flexible Readout and Integration Sensor (FRIS): new class of imaging sensor arrays optimized for air and missile defense," *Johns Hopkins APL Technical Digest*, vol. 28, no. 3, pp. 252–253, Jan. 2010.

[15] J. H. Lin, P. O. Pouliquen, A. G. Andreou, A. C. Goldberg, and C. G. Rizk, "Flexible readout and integration sensors (FRIS): a bio-inspired, system-on-chip, event based readout architecture," in *Proceedings of SPIE: Infrared Technology and Applications XXXVIII Conference*, May 2012, pp. 8353–1N.

[16] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 747–757, 2000.

[17] SDMS. (2006) Columbus Large Image Format (CLIF-2006) Dataset.

[18] R. T. Collins, X. Zhou, and S. K. Teh, "An open source tracking testbed and evaluation web site," in *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS 2005)*, 2005.

[19] R. Thakkar, "A primer for dissemination services for Wide Aray Motion Imagery, Tech. Rep. OCG 12-077r1, Dec. 2012.

[20] D. H. Goldberg, G. Cauwenberghs, and A. G. Andreou, "Probabilistic synaptic weighting in a reconfigurable network of VLSI integrate-and-fire neurons," *Neural Networks*, vol. 14, pp. 781–793, 2001.

[21] A. S. Cassidy, J. Georgiou, and A. G. Andreou, "Design of silicon brains in the nano-CMOS era: spiking neurons, learning synapses and neural architecture optimization," *Neural Networks*, vol. 45, pp. 4–26, June 2013.

[22] B. V. Benjamin, P. Gao, E. McQuinn, S. Choudhary, A. R. Chandrasekaran, J. Bussat, R. Alvarez-Icaza, J. V. Arthur, P. A. Merolla, and K. Boahen, "Neurogrid: A Mixed-Analog-Digital Multichip System for Large-Scale Neural Simulations," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 699–716, 2014.

[23] M. Mahowald, "VLSI analogs of neuronal visual processing: a synthesis of form and function," Ph.D. dissertation, Ph.D Dissertation, California Institute of Technology, 1992.

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

[25] K. A. Boahen, "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, May 2000.

[26] T. Serrano-Gotarredona, A. G. Andreou, and B. Linares-Barranco, "AER image filtering architecture for vision-processing systems," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 46, no. 9, pp. 1064–1071, Sept. 1999.

[27] B. R. Gaines, "Stochastic Computing Systems," in *Advances in Information Systems Science*, J. F. Tou, Ed. New York: Plenum, 1969, pp. 38–172.

[28] B. D. Brown and H. C. Card, "Stochastic neural computation I: Computational elements," *IEEE Transactions on Computers*, vol. 50, no. 9, pp. 891–905, Sept. 2001.

[29] P. Li and D. Lilja, "Using stochastic computing to implement digital image processing algorithms," in *Proceedings of the 29th IEEE International Conference on Computer Design (ICCD)*, 2011, pp. 154–161.

[30] A. Alaghi and J. P. Hayes, "Survey of Stochastic Computing," *ACM Transactions on Embedded Computing Systems (TECS)*, pp. 1–25, 2012.

[31] P. Li, D. J. Lilja, W. Qian, M. D. Riedel, and K. Bazargan, "Logical Computation on Stochastic Bit Streams with Linear Finite State Machines," *IEEE Transactions on Computers*, pp. 1–1, 2014.

[32] C. Zamarreño-Ramos, A. Linares-Barranco, T. Serrano-Gotarredona, and B. Linares-Barranco, "Multicasting mesh AER: a scalable assembly approach for reconfigurable neuromorphic structured AER systems; application to ConvNets," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 7, no. 1, pp. 82–102, Feb. 2013.

[33] K. Sanni, G. Garreau, J. L. Molin, and A. G. Andreou, "FPGA Implementation of a Deep Belief Network Architecture for Character Recognition Using Stochastic Computation," in *Proceedings of the 49th Annual Conference on Information Sciences and Systems (CISS)*, Feb. 2015, pp. 1–5.

[34] D. R. Mendat, S. Chin, S. B. Furber, and A. G. Andreou, "Markov Chain Monte Carlo Inference on Graphical Models using Event-Based Processing on the SpiNNaker Neuromorphic Architecture," in *Proceed-*

ings of the 49th Annual Conference on Information Sciences and Systems (CISS), Feb. 2015, pp. 1–6.

[35] ——, "Neuromorphic Sampling on the SpiNNaker and Parallella Chip Multiprocessors," in *Proceedings of the 2016 IEEE 7th Latin American Symposium on Circuits and Systems (LASCAS)*, Mar. 2016, pp. 399–402.

[36] K. A. Sanni and A. G. Andreou, "A Historical Perspective on Hardware AI Inference, Charge-Based Computational Circuits and an 8bit Charge-Based Multiply-Add Core in {16nm FinFET CMOS}," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 3, pp. 532–543, Sept. 2019.

[37] R. A. Maschal, Jr, S. Young, J. Reynolds, K. Krapels, J. Fanning, and T. Corbin, "Review of Bayer Pattern Color Filter Array (CFA) Demosaicing with New Quality Assessment Algorithms, Tech. Rep. ARL-TR-5061, Jan. 2010.

[38] J. L. Molin, T. Figliolia, K. Sanni, I. Doxas, A. G. Andreou, and R. Etienne-Cummings, "FPGA emulation of a spike-based, stochastic system for real-time image dewarping," in *Proceedings of the 58th Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE, 2015, pp. 1–4.

[39] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proceedings of the 1999 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Oct. 1999, pp. 1–7.

[40] R. P. Adams and D. J. MacKay, "Bayesian Online Changepoint Detection," *arXiv.org*, Oct. 2007.

[41] P. S. Mandolesi, P. Julian, and A. G. Andreou, "A scalable and programmable simplicial CNN digital pixel processor architecture," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 51, no. 5, pp. 988–996, May 2004.

[42] R. T. Collins, "Mean-shift blob tracking through scale space," in *Proceedings of the 2003 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2003, pp. II–234–40.

[43] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME - Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[44] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech and time-series," in *The Handbook of Brain Theory and Neural Networks*, M. A. Arbib, Ed. MIT Press, 1995.

[45] R. Dogaru, P. Julian, L. O. Chua, and M. Glesner, "The simplicial neural cell and its mixed-signal circuit implementation: an efficient neural-network architecture for intelligent signal processing in portable multimedia applications," *IEEE Transactions on Neural Networks*, vol. 13, no. 4, pp. 995–1008, 2002.

[46] M. D. Federico, P. S. Mandolesi, P. Julian, and A. G. Andreou, "Experimental results of simplicial CNN digital pixel processor," *IET Electronics Letters*, vol. 44, no. 1, pp. 27–29, Jan. 2008.