Multi-ANN Edge System based on a Custom 3DIC DRAM

Lee B. Baker, Paul Franzon *Fellow, IEEE*

***Abstract*—Machine Learning in the form of Deep Neural Networks (DNN) have gained traction over the last few years. They get good press in applications such as image recognition and speech recognition. There have been implementations that use different number formats from double precision floating point to eight bit integers, but in all cases these useful ANNs have significant memory requirements to store the connection weights (parameters) therefore requiring Dynamic Random Access memory (DRAM) to store the AN parameters.**

There have been many successful attempts to accelerate ANNs, but in most cases the focus is on a subset of the DNN known as the Convolutional Neural network (CNN). CNNs assume a significant amount of reuse of the weights connecting ANs and thus they can take advantage of local memory (SRAM).

Much of the ANN application specific (ASIC/ASIP) research has focused on taking advantage of the performance and ease of use of Static Random Access Memory (SRAM). These implementations can be shown to be effective with specific ANN architectures, such as CNNs where the ANN parameters can be stored in SRAM in a cache-like architecture avoiding constant accessing of the "slower" DRAM. In addition, to achieve a high performance, these rely on processing a batch of inputs, such as processing a batch of images or voice recordings using the same ANN.

The work in this paper considers "edge" applications which perform multiple complex functions, such as navigation, engine monitoring etc. for systems employed "in the field". The target applcation assumes that a) there is no access to cloud servers, b) the system processes a disparate set of useful sized ANNs and c) there are not opportuities to store and reuse portions of the ANN in SRAM. Therefore this work makes no assumptions regarding opportuities to store and reuse portions of the ANN in local SRAM, such as with CNNs or batch processing. These requirements dictate the need for DRAM for ANN storage and a sustained average bandwidth of 10's of $\mathrm{Tbit/s}$ when accessing that DRAM.

This work employs 3D integrated circuit technology and a proposed custom 3D-DRAM. By employing 3DIC technology, this work takes advantage of the reduced energy and area and increased connectivity and bandwidth to allow the DRAM to be employed efficiently. This work demonstrates the required average bandwidth exceeding 10's of $\mathrm{Tbit/s}$ when run against various fully and locally connected ANN layers providing a potentially 10-50X performance improvement over existing ASIC/ASIPs or GPUs.

***Index Terms*—machine learning, edge system, DNN, CNN, neural network**

I. Introduction

USEFUL DNNs often require hundreds of thousands of ANs. Within the network, each AN can have hundreds of feeder (pre-synaptic) ANs. With popular DNNs, there are often tens of layers. So in these ANNs, the memory requirements are significant. The storage is required for the input, the AN state and most significantly the weights for each of the ANs. This storage requirement often results in gigabytes of memory.

When these ANNs are required to be solved in fractions of a second, the processing and memory bandwidth becomes prohibitive.

In most cases, Graphics processing Units (GPU) are used to implement large ANNs. In many ANN architectures, such as Convolutional ANNs (CNN), they are quite effective. However, we should not forget they are not optimized purely for NN processing and are restricted by available SRAM and they are power hungry. These limitations limit the effectiveness of GPUs.

L. B. Baker, and P. Franzon are with the Department of Electrical and Computer Engineering, North Carolina State University, 2410 Campus Shore Dr., Raleigh NC 27606 Tel/Fax: 919-515-5460/5523 Email: lbbaker@ncsu.edu, paulf@ncsu.edu

Manuscript received Month Day, 2016; revised Month Day, 2016.

Much of the ANN application specific (ASIC/ASIP) research has focused on taking advantage of the performance and ease of use of Static Random Access Memory or SRAM. These implementations can be shown to be effective with specific ANN architectures (CNN), server applications or the "toy examples" but when a system requires multiple disparate ANNs in an edge application, these implementations do not provide the required flexibility, storage capacity and deterministic performance.

Another technology that has been considered over the last decade is 3D integrated circuit technology (3DIC). This 3DIC technology stacks multiple die together to form a system-on-chip with potentially disparate technology for each die in the stack. By staying within the die footprint, 3DIC technology promises high connectivity and consequently high bandwidth along with lower power all within a smaller footprint.

As a metric, this work asumes that any useful DNN will employ 100's of thousands of ANs. Although there is a lot of debate regarding number formats for ANNs, this work also assumes single-precision floating point. Assuming an ANN with 250K neurons and an average fanin to each AN of 2000, a system employing 10 ANNs for various disparate functions and an average processing time of $10\,\mathrm{ms}$ suggests a average bandwidth of $16\,\mathrm{Tbit/s}$ (1).

$$\begin{aligned}
&\text{Average Bandwidth}\\
&=\sum_{\mathbf{n}=0}^{\bar{\mathbf{N_n}}}\left(\frac{\bar{\mathbf{N_a}}\cdot\bar{\mathbf{C_p}}\cdot\bar{\mathbf{b_w}}}{\bar{\mathbf{T_p}}}\right)\\
&=\sum_{\mathbf{n}=0}^{9}\left(\frac{250\times10^{4}\cdot2\times10^{3}\cdot32}{10\times10^{-3}}\right)\\
&=16\,\mathrm{Tbit/s}
\end{aligned}\tag{1}$$

where $\mathbf{N}_n$ is the number of ANNs

$\mathbf{N}_a$ is the average number of ANs

$\mathbf{C_p}$ is the average number of connections

and $\mathbf{T_p}$ is the processing time

Regardless of the combination of ANNs, this work suggests that these edge systems will require memory bandwidth of the order of 10's of $\mathrm{Tbit/s}$.

This work demonstrates a system that is able implement multiple useful sized DNNs whilst maintaining an average memory bandwidth of $>30\,\mathrm{Tbit/s}$. This is made possible by ensuring the system stays within the die stack footprint of a typical 3DIC DRAM . This work removes a reliance on SRAM to achieve high performance thus allowing the proposed design to be utilized in edge applcations when processing multiple disparate ANNs at or near real-time. Although not optimized for specific ANNs, such as CNNs, this work demonstrates the potential for real-time performance at the edge when implementing fully connected DNNs or other similar ANNs such as LSTM.

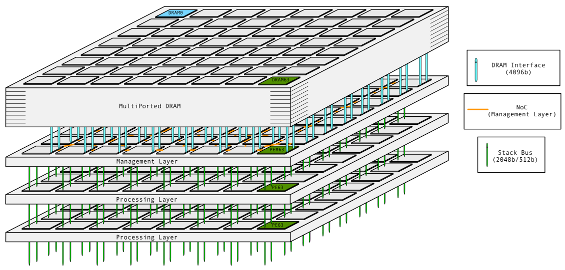
II. System Description

The primary objectives of this work was to a) consider systems that are unable to take advantage of memory reuse opportunities and therefore not able to achieve high performance using local SRAM to store ANN parameters or the ANN input, b) acknowledge that DRAM is required for storage of ANN parameters, c) that many edge devices will likely apply many disparate ANNs to perform various system functions, and d) it is assumed that many edge applications will have space and power limitations.

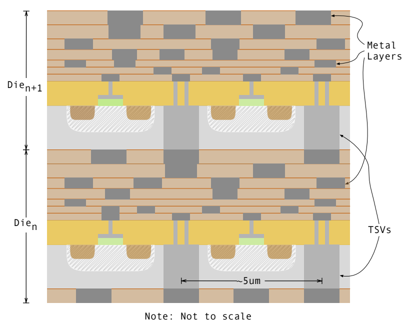
This work employs 3DIC technology along with a custom 3D-DRAM. The objective was to demonstrate that a pure 3DIC system can implement multiple disparate ANNs. By staying within the 3DIC footprint and taking advantage of high density through-silicon-vias (TSV) this work is able to maintain a significantly higher bandwidth over 2D or 2.5D ASIC/ASIP solutions.

The 3DIC system die stack (figure 1) includes the 3D-DRAM with a system manager below and one or more processing layers below the manager.

3D-DRAM has recently become available in standards such as High Bandwidth Memory (HBM) and Hybrid Memory Cube (HMC) and proprietary devices such as the DiRAM4 available from Tezzaron. These technologies provide high capacity within a small footprint.



(a) DRAM, Manager and PE Layers



(b) Example Stack profile with metal layers and TSVs [1]

Fig. 1: 3DIC System Stack

In the case of HBM and DiRAM4, the technology can be combined with additional custom layers to provide a system solution.

The question becomes, can a useful system co-exist within the same 3D footprint?

This work targeted a baselne system with:

- single precision floating point for computations
- the Tezzaron DiRAM4 DRAM [2]

The work includes customizing the interface to a 3D-DRAM, researching data structures to describe storage of ANN parameters, designing a memory manager with micro-coded instructions and a processing engine (PE) layer. The targeted 3D-DRAM, the Tezzaron DiRAM4 is a 3D-DRAM employs multiple memory array layers in conjunction with a control and IO layer. The memory is formed from 64 disjoint sub-memories each providing upwards of 1Gigabit with a total capacity of at least 64 gigabit. The system is designed such that a sub-system, known as a sub-system column (SSC) operates on one of the 64 disjoint memories within the 3D-DRAM (see figure 2).

When the sub-system columns need to share data or neuron activations, the data is passed between SSCs using a mesh connected network-on-chip (NoC).

A control and data block diagram of the 3DIC stack showing the 64 sub-system columns can be seen in figure 3. A block diagram of the sub-system column can be seen in figure 7.

An overview of the various blocks and interconnects are given below:

A. 3D-DRAM

The targeted 3D-DRAM, the Tezzaron DiRAM4 is customized to provide a 2048-bit wide bus. A read to the memory using a burst of two cycles provides access to an entire page within a bank. These customizations to support this very wide bus are discussed in IV. The wide bus is connected to the manager using TSVs and the manager directs portions of the wide bus to each lane to the PE.

B. Manager Layer

The Manager block is the main controller in the system. The operations required to process an ANN are formed from individual instructions which are decoded by the Manager. These instructions include descriptors to describe memory read operations, processing engine operations and memory write operations. The manager reads these system instructions from an instruction memory, decodes the instruction and configures the various blocks in the system. The configuration includes:

- initiate operand reads from DRAM
- prepare the processing engine (PE) to operate on the operands
- prepare the result processing engine to take the resulting neuron activations from the PE and write those results back to the DRAM
- replicate the resulting neuron activation's to neighbor managers for processing of other ANN layers

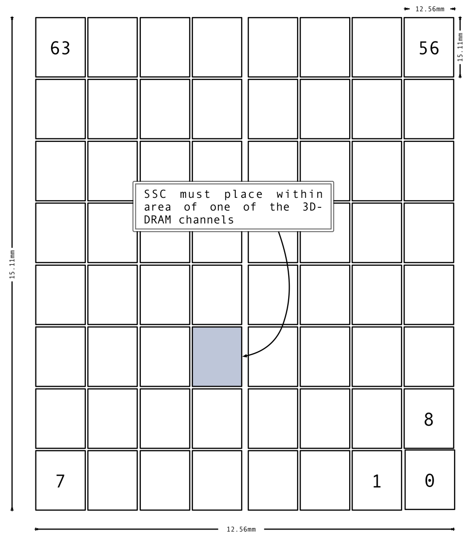


Fig. 2: DiRAM4 DRAM Physical Interface Layout[2][3] showing area for SSC

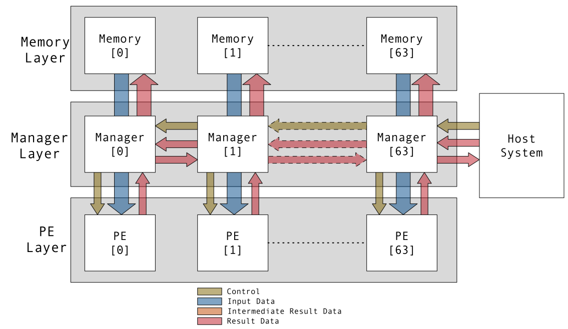


Fig. 3: System Diagram

C. Processing Layer

The PE is able to operate on data streamed directly from the DRAM via the Manager layer. The PE is configured by the manager to perform operations on the operand data streamed from the manager. In the baseline system, the main operation is to perform multiply-accumulates on 32 execution lanes of two operands. These operands typically are the pre-synaptic neuron activation's and the connection weights. The PE also performs the activation function on the result of the MAC to generate the neuron activation value. These 32 activation values are sent back to the Manager layer.

D. Layer Interconnect

The layers are connected using through-silicon-vias (TSVs) which provide high connection density, high bandwidth and low energy. Figure 1b shows an example of two die connected using TSVs.

E. Inter-Manager Communication

A Network-on-Chip (NoC) allows each management block to communicate with other managers. During configuration and/or computations, data must be transported between managers. This inter-manager communication is provided by an NoC. When computing an ANN across multiple processors, often neuron activation data must be shared. Each manager contains a NoC module and all the managers are connected in a mesh network. The NoC bandwidth was chosen to ensure the results can be multicast to any destination manager without an adverse affect on the pipelined instructions. The NoC bus width is 64 data bits plus control signals running at the system clock rate. Future testing may require additional NoC bandwidth.

III. System Operations

In the context of this system and AN state calculation, the basic operations to determine the state of a neuron is to:

- Inform the Manager and PE which operations are to be performed
- Instruct the manager to access the states of the pre-synaptic neurons
- Instruct the manager to access the weights of the connections from the pre-synaptic ANs
- Provide the pre-synaptic neuron weights and states to the processing engine execution lanes
- Instruct the manager where to store the resulting AN state back to memory

This work has researched an instruction architecture to describe the above operations. These instructions are decoded by the manager.

In the baseline system, the manager is not responsible for performing specific algorithm operations but is responsible for coordinating the various data flows and configuration of the modules that make up the system.

The managers primary responsibility is:

- Instruction decode
- Internal Configuration messages
- Operand read
- Result write

In the baseline system, the PE is responsible for the main algorithm operations.

The PE has three major blocks:

- Streaming operation function (stOp)
  - processes data from the manager on-the-fly without storing in local SRAM
- SIMD
  - processes the data from the stOp function
    - neuron activation function such as ReLu
    - perform non vector operations such as softmax conversion using local SIMD functions, such as e^x and divide
  - sends the result back to the manager
- DMA/local memory controller
  - transfer configuration data to PE conroller or to store stOp results to a small local SRAM which can be used for access by SIMD or by the stOp function

A. Manager Operations

1) Instructions: The instructions include information to control the following operations.

- To the Manager
  - ROI Storage descriptor
  - Parameter/Weight Storage Descriptor
    - Broadcast or Vectored
  - Result write storage descriptor
    - include descriptors for all destination managers
- To the PE
  - stOp operation
  - SIMD operation
  - Number of active lanes
  - Operand Vector length

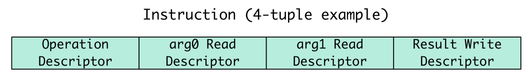


Fig. 4: Instruction 4-tuple

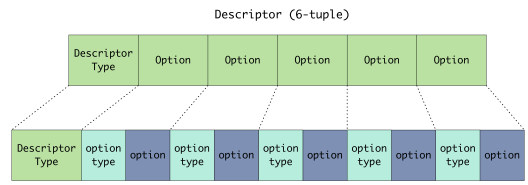


Fig. 5: Descriptor 6-tuple

Instructions contain sub-instructions called descriptors. These descriptors contain the information to control the various operations associated with the processing of a group of ANs. The group size is related to the number of execution lanes which for the baseline system is 32. A group can be anywhere from 1-32. It should be said that unless the group size consistently approaches 32 the system performance will be poor.

An instruction will typically have four descriptors:

1) Operation
2) Memory read for operand stream 0
3) Memory read for operand stream 1
4) Result Write

The instruction is an n-tuple where the tuple elements are descriptors and the number of elements can vary based on the operation being performed. In figure 4 is shown the format of a 4-tuple instruction which is used to perform an activation calculation for a group of neurons.

The fields within the descriptor are n-tuples where the first tuple element describes the descriptors operation followed by an m-tuple whose elements contain the options required for the operation.

These option elements are a two-tuple with option and associated value. The format of a 6-tuple descriptor can be seen in figure 5 .

2) Write Back to Memory: When the PE has processed the group of ANs, the new AN states are sent back to the manager for storing in the DRAM.

In many cases the AN activations from a particular PE have to be replicated not only to the local manager but also to other managers. This is handled with the network-on-chip (NoC). When the result has to be replicated to other managers, the data is sent, along with storage information over the NoC to all destination managers.

B. PE Operations

1) Streaming Operations (stOp): The operations performed by the stOp are primarily multiple-accumulate with a transfer to the SIMD or to local memory.

In most cases, the stOp module will operate on the AN state and weights provided by the manager and provide the result to the SIMD.

2) SIMD: The SIMD is a 32-lane processor with some builtin special functions including e^x and divide to allow on-the-fly operations.

The SIMD will take the result provided by the stOp and perform additonal operations such as neuron activation, pooling or softMax. The result will then be transmitted back to the manager.

3) Configuration: To configure the PE operations, the manager extracts two pointers from the instruction and sends them in a configuration packet to the PE. These pointers index into a small local memory which provides a program counter (PC) to the function to be performed by the SIMD and a configuration entry for the operation to be performed by the stOp.

A detailed block diagram of the sub-system column (SSC) can be seen in figure 7.

IV. Suggested DRAM Customizations

Accessing a "typical" DRAM involves opening a page in a bank, reading or writing a portion of the contents of the page then closing the page.

Typically a bank may contain of the order of a few thousand pages and a page may contain of the order of a few thousand bits.

Once the page is open, the user accesses a portion of the requested page over a bus. With PCB based DRAMs the bus might vary from four to 16 bits wide, but with 3D DRAMs, such as HBM the bus might be up to 128 bits wide.

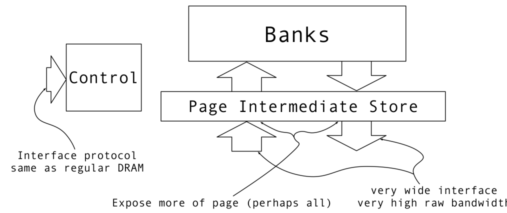


Fig. 6: Exposing more of the DRAM page

A. Expose more of the Page

This work achieves the increase in bandwidth by proposing that the DRAM expose more of its currently open page.

Without the limitations of having to transfer data beyond the chip stack, this work suggests exposing a larger portion of the page over a 2048-bit wide bus. By staying within the 3D footprint, this bus can be implemented using fine pitch TSVs. (see figure 6).

B. DRAM Write Mask

For every group of ANs processed, the state of the group of ANs is written back to memory.

Typically this would require a read/modify/write of a DRAM cacheline. In the case writing back 32 AN states into a 4096 bit cacheline means the read/modify/write is inefficient. To minimize the inefficiency, a customization to the DRAM is the addition of a write data mask to the DRAM write path eliminating the additional read.

V. Results

The objectives of this work was to design a system able to accelerate ANNs in customer facing systems implemented at the edge. Given that these systems cannot effectively utilize SRAM, the main objective was to demonstrate a system that can operate efficiently using 3D-DRAM.

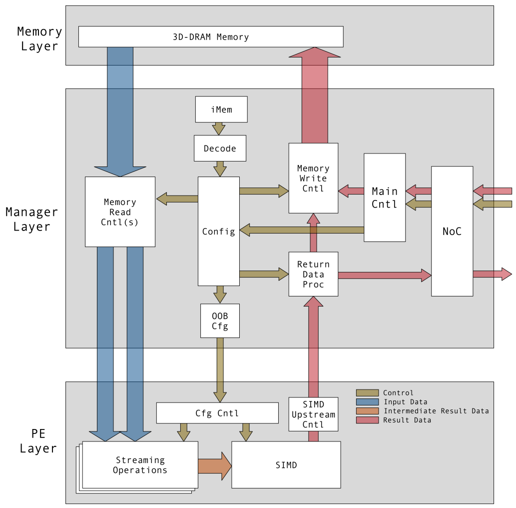


Fig. 7: Sub-System Column (SSC) Block Diagram

The system decodes instructions, sends configuration to various functions, pre-fetchs and pipelines data. This parallelism allows the system to constantly stream data whilst results from previous operations are being operated on.

To demonstrate such a system, this work targeted 3DIC technology including 3D-DRAM. This work proposes that if a system can be purely in 3DIC, the system can take advantage of the benefits of 3DIC which includes reduced energy, area and high bandwidth. In addition, this work proposes that given the system is 3DIC, then a customized DRAM would provide a significant bandwidth boost over typical implementations using standard DRAM.

The target technology node was 28nm because its the technology node employed for some recent GPUs and other ASICs such as [4]. As a 28nm was not available to this team, the design was synthesized using an available 65nm technology node and then scaled to 28nm.

The primary control and datapaths of the system have been simulated in a system verilog environment. Initial synthesis timing closure at a frequency of $500\,\mathrm{MHz}$ is complete.

Initial place and route for the Manager and PE are shown in figure 8. The area contribution of each block within the Manager and PE can be seen in table I.

TABLE I: Area Contribution

Block Name	Instances	Percentage Contribution
Memory Controller	1	15.0 %
NoC	1	7.1 %
Read Control	2	53.1 %
Write Control	1	7.4 %
Instruction Proc	1	1.6 %
Return Data Proc	1	1.6 %
Misc	1	14.2 %

(a) Manager

Block Name	Instances	Percentage Contribution
Local Memory + Control	1	17.7 %
Operation Decode	1	3.4 %
Return Data Control	1	1.5 %
SIMD Control	1	8.1 %
SIMD	1	19.3 %
Streaming Operations	32	43.3 %
Streaming Op Control	1	2.1 %
Misc	1	4.6 %

(b) PE

The parasitics were extracted from these layouts and simulated against a group of operations. The operations simulated were based on the expected lower and upper limits of pre-synaptic fanin. These testcases were based on layers similar to CONV2 and FC-7 from [5] and represent a pre-synaptic fanin of 225 and 4000 respectively. Additional testcases were employed representing pre-synaptic fanins of 294, 300, 500 and 1000. Both locally connected (CONV) and fully connected (FC) type fanins were tested. The results showing sustained average bandwidth can be seen in table III.

The simulation generated an activity file which was then used by the Synopsys® Primetime-PX™ power analysis tool to obtain power and bandwidth estimates. The DRAM accesses were captured and DRAM energy dissipation calculated from [2]. The power dissipated in the TSVs were estimated from [6]. These estimates were used to estimate power dissipation for operating frequencies of 500 MHz and 700 MHz. The estimated overall power along with per block contribution are shown in table II.

TABLE II: Power Estimates

Technology Node	Clock Frequency	Total Expected Power	Testcase
28nm	500 MHz	48W	CONV-294
28nm	700 MHz	65W	CONV-294

(a) Power Dissipation

Block Name	Percentage Contribution
Manager	61.1 %
PE	31.6 %
DRAM	3.4 %
DRAM TSVs	2.4 %
Stack Bus TSVs	1.6 %

(b) Power Contribution

As bus efficiency is the main metric, table III shows sustained average bandwidth over the fanin testcases.

TABLE III: Fanin Bandwidth Tests

Test	Average Bandwidth At Frequency 500 MHz	700 MHz
CONV2 [5]	∼22 Tbit/s	∼30 Tbit/s
CONV-294	∼23 Tbit/s	∼31 Tbit/s
CONV-300	∼25 Tbit/s	∼34 Tbit/s
CONV-500	∼26 Tbit/s	∼38 Tbit/s
CONV-1000	∼30 Tbit/s	∼41 Tbit/s
FC-350	∼26 Tbit/s	∼36 Tbit/s
FC-500	∼27 Tbit/s	∼38 Tbit/s
FC-1000	∼30 Tbit/s	∼42 Tbit/s
FC-7 [5]	∼31 Tbit/s	∼43 Tbit/s

VI. Conclusions

There have been many attempts to accelerate ANNs. Many have shown excellent performance mainly when implementing CNNs. The improvement mostly comes from the ability to hold kernel weights and/or AN activations in local SRAM.

(a) Manager (b) PE

Fig. 8: Manager and PE Die layouts

Another method of employing local memory is often due to pooling of batch requests, especially in server applications. This local storage allows the system to take advantage of the low latency and random access benefits of SRAM whilst performing multiple operations on that data. When considering applications where this local storage cannot be used effectively, all these implementations suffer a large degradation in performance.

This work considers edge applications where a system is processing requests with a disparate set of ANNs. The assumption is that local SRAM is no longer effective and performance is based on DRAM bandwidth. This work considers 3DIC technology and a customized 3D-DRAM is proposed.

The customized 3D-DRAM combined with a design based on custom instructions and operation descriptors allows the system to achieve high levels of memory bandwidth efficiency.

There is no doubt existing CNN accelerators that take advantage of batch processing achieve a performance that is difficult to better, but applying these systems to this works target application exposes those systems DRAM bandwidth limitations.

This work demonstrates a 3D-DRAM system that given the target application, provides a potentially 10-50X performance improvement over existing ASIC/ASIPs or GPUs.

VII. Acknowledhements

This work was funded in part by DARPA and AFRL under FA8650-15-1-7518 and DARPA and

ONR under N00014-17-1-3013, as part of the CHIPS program.

REFERENCES

[1] ITRS, "International technology roadmap for semiconductors 2.0," 2015. [Online]. Available: http://www.itrs2.net/itrs-reports.html

[2] *DiRAM4-64Cxx Cached Memory Subsystem*, Tezzaron Semiconductor, July 2015, rev. 0.04.

[3] R. Patti, "2.5 d and 3d integration technology update," *Additional Papers and Presentations*, vol. 2014, no. DPC, pp. 1–35, 2014.

[4] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers *et al.*, "In-datacenter performance analysis of a tensor processing unit," *arXiv preprint arXiv:1704.04760*, 2017.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[6] Y. Liu, W. Luk, and D. Friedman, "A compact low-power 3d i/o in 45nm cmos," in *2012 IEEE International Solid-State Circuits Conference*. IEEE, 2012, pp. 142–144.

Lee B. Baker received a B.S. degree in Electrical Engineering from Brighton Polytechnic, UK, an M.S. degree in Electrical Engineering from Villanova University, USA and an M.B.A. degree from North Carolina State University, USA in 1983, 1994 and 2009 respectively. He is currently pursuing the Ph.D. degree with the Electrical and Computer Engineering Department at North Carolina State University. His current research interests include acceleration of artificial neural networks.

Paul Franzon is currently the Cirrus Logic Distinguished Professor of Electrical and Computer Engineering at North Carolina State University. He earned his Ph.D. from the University of Adelaide, Adelaide, Australia in 1988. He has also worked at AT&T Bell Laboratories, DSTO Australia, Australia Telecom and three companies he cofounded, Communica, LightSpin Technologies and Polymer Braille Inc. His current interests center on the technology and design of complex microsystems incorporating VLSI, MEMS, advanced packaging and nano-electronics. He has lead several major efforts and published over 300 papers in these areas. In 1993 he received an NSF Young Investigators Award, in 2001 was selected to join the NCSU Academy of Outstanding Teachers, in 2003, selected as a Distinguished Alumni Professor, and received the Alcoa Research Award in 2005. He served with the Australian Army Reserve for 13 years as an Infantry Solider and Officer. He is a Fellow of the IEEE.