Multi-ANN embedded system based on a custom 3D-DRAM

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

***Abstract*—Machine Learning in the form of Artificial Neural Networks (ANNs) has gained traction over the last few years especially in applications such as image recognition and speech recognition. These particular applications typically employ a subset of ANNs known as Convolutional Neural Networks (CNNs) which re-use parameters and thus reduce main memory bandwidth. However, there are other types of ANN that do not provide reuse opportunities such as autoencodersand Long Short-term memory (LSTM). It is generally accepted that dynamic random-access memory (DRAM) is required to store the ANN parameters of useful sized ANNs. To achieve a given performance, CNN-specific implementations utilize cache-like structures using static random-access memory (SRAM) which mimimizes accesses to the slower DRAM. Most research has focused on implementing CNNs but because of their extensive use of SRAM have both ANN size restrictions and performance degradation when used in applications that utilize other types of ANN. This work considers embedded applications employing multiple disparate generic ANNs which, assuming there are limited reuse opportunities in the form of re-use or batch processing, will require usable memory bandwidth on the order of tens of** Tbit/s**. By demonstrating effective use of the very wide bus of a customized 3D-DRAM, this work demonstrates a conservative 3X power improvement and 6X area improvement over similar ANN systems along with the ability to keep pace with the DRAM roadmap over the foreseeable future.**

***Index Terms*—machine learning, embedded system, Deep Neural Networks (DNNs), CNN, neural network**

I. Introduction

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.

USEFUL DNNs often require hundreds of thousands of Artificial Neurons (ANes). Within the network, each ANe can have hundreds of feeder (pre-synaptic) ANes. 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 ANe state and most significantly the pre-synapic ANe connection weights for each of the ANes. 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 (GPUs) 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 ANN processing and are restricted by available SRAM and they are power hungry. These limitations limit the effectiveness of GPUs.

Much of the ANN Application-Specific Integrated Circuit (ASIC) and Application-Specific Instruction-set Processor (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 when there are reuse opportiunities such as with CNNs or applications that have batch processing opportunities, such as cloud applications. But given an embedded system requiring multiple disparate ANNs, where reuse opportunities do not exist, these implementations do not provide the required performance. Another technology that has been considered over the last decade is Three-Dimensional Integrated Circuit (3DIC). Implementations such as Neurostream [1], NeuroCube[4] and Tetris [5] employ Hybrid Memory Cube (HMC)

3D-DRAM in a 3DIC system, and all have demonstrated improvements over non-3DIC systems and GPUs. However, for the most part these systems employ the 3D-DRAM as a disjoint memory with the ANN processing elements being an appendage to the DRAM. In particular, some 3DIC systems that employ 3D-DRAM achieve the required performance by optimistically targeting a 15nm process where this work achieves the required performance by tightly integrating the ANN processing with the memory controller while targeting a less optimistic 28nm process.

This work considers future embedded systems where multiple ANNs are employed [6] and that each of the ANNs will be of similar size to useful-sized ANNs [7]. In addition, as ANNs fulfill their potential and users start to employ them in various applications, there is no reason to believe these applications will exhibit the same stationary characteristcs that make CNNs effective in image processing. Therefore, this work provides support to DNNs that do not provide ANN parameter reuse [8] and suggests that these types of applications will require that all ANN parameters in main memory be accessed in real-time. This work coins the phrase "goldilocks bandwidth" when applied to ANN systems where the system provides the bandwidth required to read all ANN parameters in a real-time sampling rate of 16 ms. Anything exceeding the required bandwidth is considered "too hot" and therefore a waste and anything less than the required bandwidth is "too cold" and therefore inadequate. This work provides support for all types of DNNs including LSTM, autoencoders and restricted bolzmann machines (RBMs) and does not focus on particular ANNs such as CNNs.

Although there is a lot of debate regarding number formats for ANNs, this work also assumes single-precision floating point. This work assumes any useful ANN will be of a similar size to [7], this work employs a baseline ANN similar to that shown in Table I, with 772 thousand ANes and an average fanin to each ANe of 1650. However, the assumption is the convolutional layers employ kernels with non-shared weights. A system employing 10 ANNs for various disparate functions and an average processing time of 16 ms will require an average or "goldilocks" bandwidth of 26 Tbit/s (1) and a capacity of 8.0 GB (2).

Maximum bandwidth

$$\begin{aligned}
&= \sum_{\mathbf{n}=0}^{\mathbf{N_n}} \Big(\frac{\overline{\mathbf{N}}_\mathbf{a} \cdot \overline{\mathbf{C}}_\mathbf{p} \cdot \mathbf{b_w}}{\overline{\mathbf{T}}_\mathbf{p}} \Big) \text{bit/s} \\
&= \sum_{\mathbf{n}=0}^{9} \Big(\frac{772 \times 10^3 \cdot 1.65 \times 10^3 \cdot (32+1)}{16 \times 10^{-3}} \Big) \\
&= \sum_{\mathbf{n}=0}^{9} 2.63 \,\text{Tbit/s} \\
&\approx 26 \,\text{Tbit/s} \qquad (1)
\end{aligned}$$

where $\mathbf{N}_n$ is the number of ANNs
$\mathbf{N}_a$ is the average number of ANes
$\mathbf{C_p}$ is the average number of connections
$\mathbf{b_w}$ is the number of bits per parameter
and $\mathbf{T_p}$ is the processing time

$$\begin{aligned}
\text{ANN Memory} &= \sum_{\mathbf{n}=0}^{\mathbf{N_n}} \Big(\big(\overline{\mathbf{N}}_\mathbf{p} + \overline{\mathbf{N}}_\mathbf{a} \big) \cdot \mathbf{b_w} \Big) \text{Gbit} \\
&= \sum_{\mathbf{n}=0}^{9} \Big(\big(202 \times 10^6 + 772 \times 10^3 \big) \cdot 32 \\
&= \sum_{\mathbf{n}=0}^{9} 6.49 \,\text{Gbit} \\
&= 64.9 \,\text{Gbit} \equiv 8.1 \,\text{GB} \qquad (2)
\end{aligned}$$

where $\mathbf{N}_n$ is the number of ANNs
$\mathbf{N}_p$ is the number of parameters per ANN
$\mathbf{N}_a$ is the number of ANes per ANN
and $\mathbf{b_w}$ is the number of bits per parameter

By employing a 3DIC system along with customizations to a standard 3D-DRAM, this work demonstrates a system that is able implement multiple useful sized DNNs whilst maintaining the required average ("goldilocks") memory bandwidth

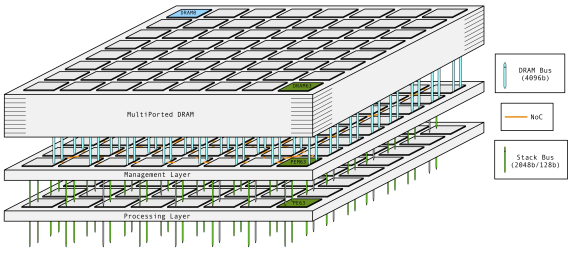
		Layers											
		1	2	3	4	5	6	7	8	9	10	11	
Type		Input	Locally	Pooling	Locally	Pooling	Locally	Locally	Locally	Fully	Fully	Fully	
Dimensions	X	256	55	27	27	13	13	13	13	4096	4096	1024	
	Y	256	55	27	27	13	13	13	13	1	1	1	
	Z	3	96	96	256	256	384	384	256	1	1	1	
Filter Dimensions	X	na	11	2	5	2	3	3	3	13	4096	4096	
	Y	na	11	2	5	2	3	3	3	13	1	1	
	Z	na	3	1	96	1	256	384	384	256	1	1	
Stride		na	4	2	2	2	1	1	1	na	na	na	Aggregate
Pre-synaptic Fanin		na	363	4	2400	4	2304	3456	3456	43 264	4096	4096	1650
Number of ANe		196 608	290 400	69 984	186 624	43 264	64 896	64 896	43 264	4096	4096	1024	772 544
Number of Weights		na	34 848	na	614 400	na	884 736	1 327 104	884 736	177 209 344	16 777 216	4 194 304	2.02×10^8

TABLE I: Baseline ANN layer configuration [7]

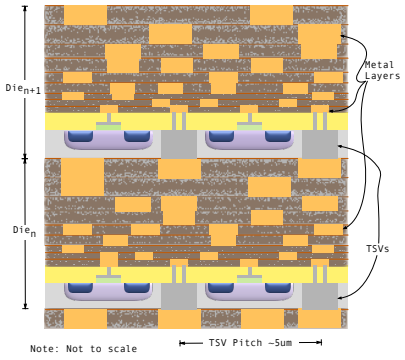
of 26 Tbit/s. This work provides at least a 6X power improvement and a 3X area improvement over similar state of the art systems [1][4][5][9] when those systems are scaled to meet the requirements of these embedded applications. This work removes a reliance on SRAM to achieve high performance thus allowing the proposed design to be utilized in embedded 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 when implementing multiple fully connected DNNs or other similar ANNs such as LSTM.

II. System Description

The primary objectives of this work was to consider applications that are unable to take advantage of memory reuse opportunities and therefore not able to achieve high performance using local SRAM, consider embedded devices that will apply many disparate ANNs to perform various system functions, and assume that embedded applications have space and power limitations. This work employs 3DIC technology along with a proposed custom very wide databus 3D-DRAM. The objective was to demonstrate that this very wide databus 3D-DRAM can be used effectively and demonstrate that such a system can stay within the 3DIC footprint of the DRAM. 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.



(a) DRAM, Manager and PE Layers



(b) Example Stack profile with metal layers and TSVs [10]

Fig. 1: 3DIC System Stack

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.

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 baseline system with:

- single precision floating point for computations
- the Tezzaron DiRAM4 DRAM [11]

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.

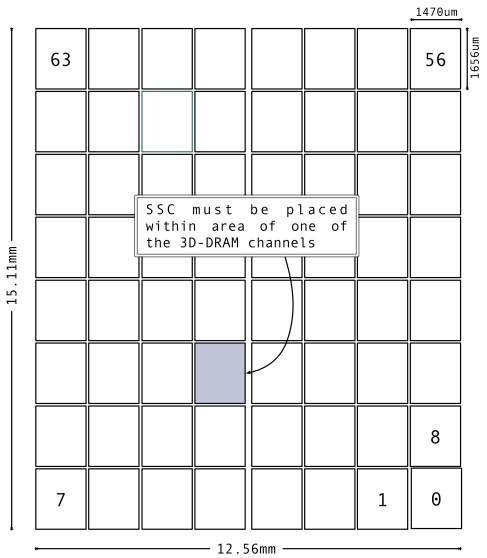


Fig. 2: DiRAM4 DRAM Physical Interface Layout[11][12] showing area for SSC

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

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 ANe 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 ANes

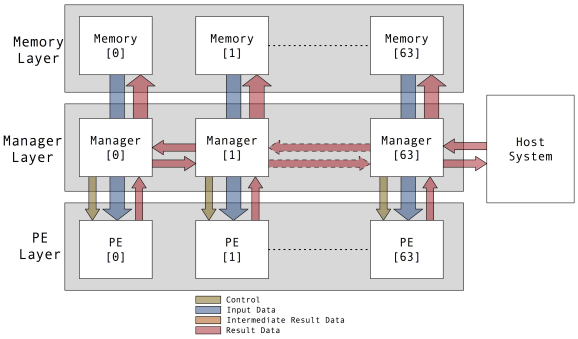


Fig. 3: System Diagram

- Provide the pre-synaptic neuron weights and states to the processing engine execution lanes
- Instruct the manager where to store the resulting ANe 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 block (StOp)
  - Processes data from the manager on-the-fly without storing in local SRAM
- Single-Instruction Multiple-Data (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
  - region-of-interest (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

Instructions contain sub-instructions called descriptors. These descriptors contain the information to control the various operations associated with the processing of a group of ANes. 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.

Fig. 4: Instruction 4-tuple

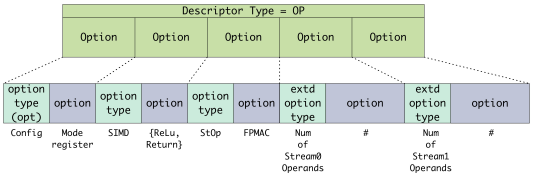


Fig. 5: Descriptor 5-tuple

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

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

In many cases the ANe 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 Operation block (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 ANe 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

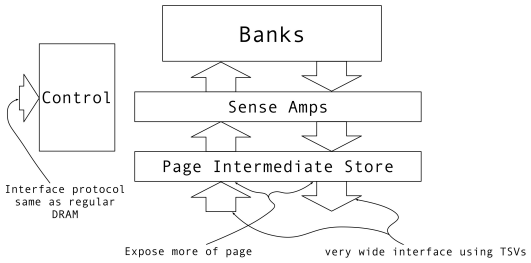


Fig. 6: Exposing more of the DRAM page

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.

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).

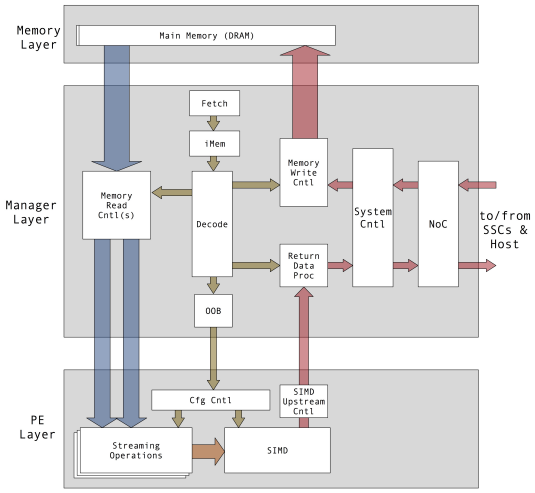


Fig. 7: Sub-System Column (SSC) Block Diagram

B. DRAM Write Mask

For every group of ANes processed, the state of the group of ANes is written back to memory. Typically this would require a read/modify/write of a DRAM cacheline. In the case writing back 32 ANe 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 objective of this work was to design a system able to accelerate multiple disparate ANNs in embedded systems. Given that these systems cannot effectively utilize SRAM, the main objective was to demonstrate a system that can operate efficiently using a customized 3D-DRAM with a very wide data-bus. 3DIC technology was exploited, including 3D-DRAM, because a major theme of this work is that 3DIC provides many benefits, including reduced energy use, lower area requirements, and high bandwidth. It was necessary to show that the proposed system can maintain the required data bandwidth while staying within the physical footprint of the 3D-DRAM.

Because its the technology node employed for some recent GPUs and other ASICs such as [13], the target technology chosen was 28nm. The 28nm standard cell library available to this team did not have register files, therefore the design was synthesized using an available 65nm technology node and then scaled to 28nm. Some representative portions of the design were synthesized using the 28 nm libraries to obtain scaling numbers. The final scaling numbers are shown in Table II.

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 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 III.

TABLE III: 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
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 %
Local Memory + Control[a]	1	17.7 %
Misc	1	4.6 %

(b) PE

[a] A small amount of scratchpad memory was provided between stOps and SIMD but in practice could be much smaller. It is not used in any of the fanin tests.

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 [7] 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 V.

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 [11]. The power dissipated in the TSVs were estimated from [14]. 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 IV.

TABLE IV: Power Estimates

Technology Node	Clock Frequency	Total Expected Power	Testcase
28nm	500 MHz	64W	CONV-294
28nm	700 MHz	88W	CONV-294

(a) Power Dissipation

Block Name	Percentage Contribution
Manager	66.6 %
PE	28.0 %
DRAM	2.5 %
DRAM TSVs	1.8 %
Stack Bus TSVs	1.2 %

(b) Power Contribution

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

VI. Conclusions

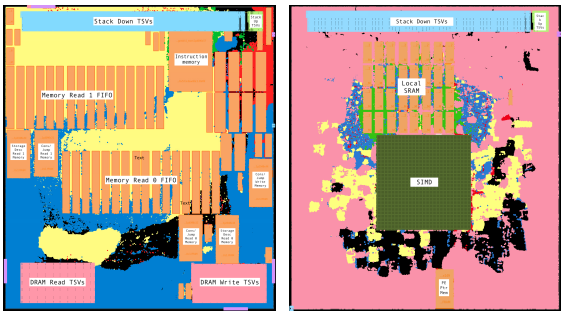
There have been many attempts to accelerate ANNs. Many have shown excellent performance

		Logic power			Memory power		
Logic Area	Memory Area	Internal	Net switching	Leakage	Internal	Net switching	Leakage
2.68	2.78	5.07	1.21	4.12×10^{-4}	5.07	NA	NA

TABLE II: 68 nm to 28 nm scaling numbers

TABLE V: Fanin Bandwidth Tests

Test	Average Bandwidth At Frequency 500 MHz	700 MHz
CONV2 [7]	∼ 25 Tbit/s	∼ 35 Tbit/s
CONV-294	∼ 26 Tbit/s	∼ 37 Tbit/s
CONV-300	∼ 27 Tbit/s	∼ 37 Tbit/s
CONV-500	∼ 29 Tbit/s	∼ 41 Tbit/s
CONV-1000	∼ 31 Tbit/s	∼ 43 Tbit/s
CONV-2500	∼ 32 Tbit/s	∼ 45 Tbit/s
FC-350	∼ 28 Tbit/s	∼ 39 Tbit/s
FC-500	∼ 29 Tbit/s	∼ 41 Tbit/s
FC-1000	∼ 31 Tbit/s	∼ 43 Tbit/s
FC-7 [7]	∼ 32 Tbit/s	∼ 45 Tbit/s



(a) Manager (b) PE

Fig. 8: Manager and PE Die layouts

mainly when implementing CNNs. The improvement mostly comes from the ability to hold kernel weights and/or ANe activations in local SRAM. 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 embedded 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 3DIC system that at the surface provides relatively low floating point operations per second (FLOPS), but considering the target application is memory bound, this work demonstrates a 3X power improvement and 6X area improvement over similar ANN systems [9], [1], [4], [5].

VII. Acknowlembeddedments

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] E. Azarkhish, D. Rossi, I. Loi, and L. Benini, "Neurostream: Scalable and energy efficient deep learning with smart memory cubes," *arXiv preprint arXiv:1701.06420*, 2017.

[2] Y. Chen, T. Luo, S. Liu, S. Zhang, L. He, J. Wang, L. Li, T. Chen, Z. Xu, N. Sun, and O. Temam, "Dadiannao: A machine-learning supercomputer," in *47th Annual IEEE/ACM International Symposium on Microarchitecture*, Dec 2014, pp. 609–622.

[3] T. Luo, S. Liu, L. Li, Y. Wang, S. Zhang, T. Chen, Z. Xu, O. Temam, and Y. Chen, "Dadiannao: A neural network supercomputer," *IEEE Transactions on Computers*, vol. 66, no. 1, pp. 73–88, Jan 2017.

[4] D. Kim, J. Kung, S. Chai, S. Yalamanchili, and S. Mukhopadhyay, "Neurocube: A programmable digital neuromorphic architecture with high-density 3d memory," in *Computer Architecture (ISCA), 2016 ACM/IEEE 43rd Annual International Symposium on*. IEEE, 2016, pp. 380–392.

[5] M. Gao, J. Pu, X. Yang, M. Horowitz, and C. Kozyrakis, "Tetris: Scalable and efficient neural network acceleration with 3d memory," in *Proceedings of the Twenty-Second International Conference on Architectural Support for Programming Languages and Operating Systems*. ACM, 2017, pp. 751–764.

[6] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[7] 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.

[8] A. Coates, B. Huval, T. Wang, D. Wu, B. Catanzaro, and N. Andrew, "Deep learning with cots hpc systems," in *International Conference on Machine Learning*, 2013, pp. 1337–1345.

[9] Y. Chen, T. Chen, Z. Xu, N. Sun, and O. Temam, "Diannao family: Energy-efficient hardware accelerators for machine learning," *Commun. ACM*, vol. 59, no. 11, pp. 105–112, Oct. 2016. [Online]. Available: http://doi.acm.org/10.1145/2996864

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

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

[12] R. Patti, "2.5d and 3d integration technology update," *Additional Conferences (Device Packaging, HiTEC, HiTEN, & CICMT)*, vol. 2014, no. DPC, pp. 1–35, 2014. [Online]. Available: https://doi.org/10.4071/2014DPC-ta12

[13] 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," in *Proceedings of the 44th Annual International Symposium on Computer Architecture*. ACM, 2017, pp. 1–12.

[14] 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 earned his Ph.D. in Electrical and Computer Engineering from North Carolina State University in 2018. 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.