Hardware-Enabled Artificial Intelligence

William J. Dally[1], C. Thomas Gray[2], John Poulton[2], Brucek Khailany[3], John Wilson[2], and Larry Dennison[4]
NVIDIA, [1]Santa Clara, CA, [2]Durham, NC, [3]Austin, TX, [4]Westford, MA
bdally@nvidia.com

Abstract— **The current resurgence of artificial intelligence is due to advances in deep learning. Systems based on deep learning now exceed human capability in speech recognition [1], object classification [4], and playing games like Go [9]. Deep learning is enabled by powerful, efficient computing hardware [5]. The algorithms used have been around since the 1980s [7], but it has only been in the last few years - when powerful GPUs became available to train networks - that the technology has become practical. This paper discusses the circuit challenges in building deep-learning hardware both for inference and for training.**

INTRODUCTION

This paper considers the design of systems for deep neural networks – both training and inference – and suggests opportunities where circuit design can have a positive impact on these systems. Two example systems are shown in Figure 1. To train large networks in a reasonable period of time, we consider a system built from 128 GPUs with high-bandwidth interconnection. For inference, we consider an SoC that processes video streams from an HD camera.

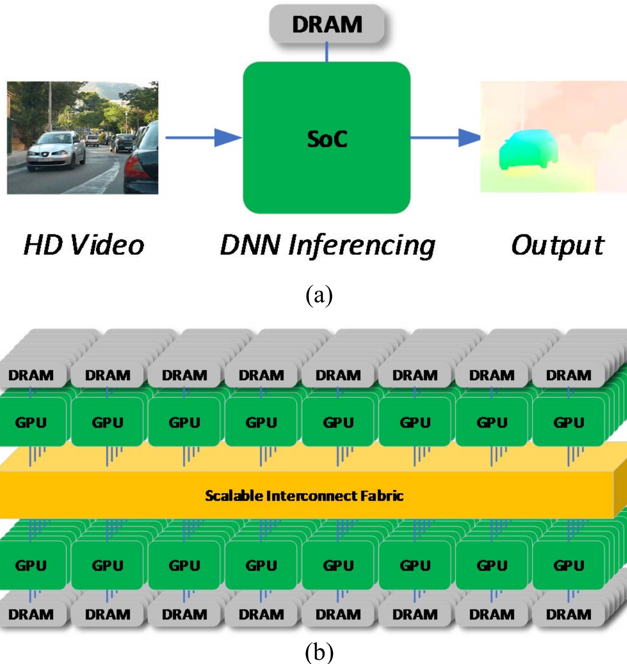

Figure 1: Example Systems
(a) Sketch of system for real-time inference on HD video
(b) Sketch of 128-GPU system for deep NN training

We start by examining the core multiply and add operations needed to implement a convolutional neural network. Training and inference have very different arithmetic needs. We go on to look at the communication needed by the training system at three levels: on-chip, on-module, and between modules.

CORE OPERATIONS

Deep neural network computations, as shown in Figure 2, are dominated by convolutions and matrix multiplies. Training requires high dynamic range (at least 200dB) and hence floating-point arithmetic. Multiplies require at least 16-bit floating-point and the results are summed using 32-bit floating point. The high dynamic range makes these operations unsuitable for analog computation, so the circuit focus for training is on efficient implementation of floating-point units.

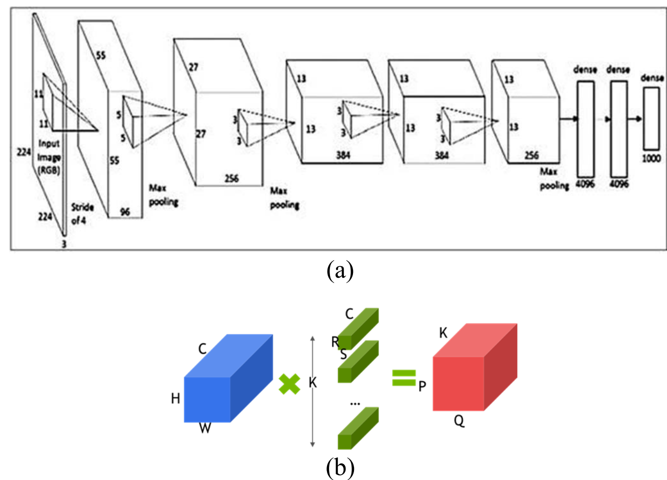

Figure 2: Deep Neural Network Computation
(a) Example: Neural network computation in AlexNet [6]
(b) Each convolutional layer processes *c* channels of *h*×*w* images with *r*×*s* filters to produce *k* channels of *p*×*q* output images

In contrast, inference can be performed at very low precision using 8-bit (or less) integer multiplies and summing with enough precision to avoid overflow. Such computations could potentially be performed using analog circuits. However, most proposed analog approaches are not competitive with state-of-the-art CMOS digital implementations.

Consider a deep-learning inference task using a convolutional neural network, similar to the network shown in Figure 2, but with 20 layers. A typical layer has c = 128 input channels, k = 128 output channels, a feature map size of $h \times w$ = 240 × 135, and a kernel size of $r \times s$ = 3 × 3. This reflects a typical layer processing an HD image after 3 stages of 2:1 pooling. We consider weights represented by b = 3 bits – exponentially coded to represent the values {-1, -1/2, -1/4, 0, 1/8, 1/4, 1/2, 1} – and activations represented by a = 4 bits encoded as 2's complement signed integers. With a batch size of N=1, the layer requires $brsck$ = 440k bits of weights and $ahw(c+k)$ = 33M bits of activations.

The core operation is a shift-and-add. The 4-bit activation is shifted by 0-3 bits, selectively complemented or zeroed, and added to a 16-bit running sum. In 16nm logic, operating at 0.6V, this operation requires about 10fJ. Each weight is reused hw = 32k times and each activation is reused rsk = 1.2k times so memory access energy can be amortized by tiling. A total of $hwckrs$ = 4.8G shift and add operations are performed per layer consuming 48µJ of energy. For a 20-layer network, a total of 96G shift and adds are performed consuming about 1mJ of energy. At 30 frames per second, running this single network takes 30mW.

The key circuit challenge for inference is to perform a multiply-and-add operation using the minimum amount of energy. This is best achieved by co-designing the data representation, the circuit, and the model to achieve the lowest energy without loss of accuracy. The "exponent-only" data representation described above, is an example of an energy-efficient data representation.

On-Chip Communication

Training deep neural networks is communication intensive. Consider the example GPU system shown in Figure 3, with 80 streaming multiprocessors (SMs) that each perform 1.5TFLOPs of FP16 in the form of 64 × 64 matrix multiply-add operations. Each matrix multiply-add performs 512KFLOPs and requires 16K bytes of input – 32 FLOPS/byte. In a typical 16 nm technology, the core operation – performing a 16-bit floating-point multiply-add as part of each dot product computation in the matrix multiply-add – consumes 0.6 pJ. Across 80 SMs, this requires 36W. Typically, input and output data must be staged in a dense SRAM. A naïve implementation would fetch 3 input operands and store a result at the cost of around 1.4 pJ, more than the computation itself. A typical deep neural network training architecture can exploit the ample data reuse in a matrix multiply-add operation to minimize the required SRAM bandwidth and reduce the SRAM power dissipated in the communication system.

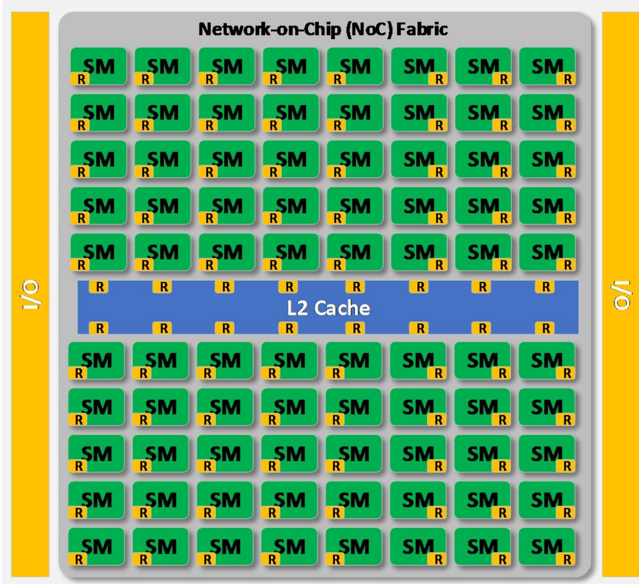

Figure 3: Sketch of an example GPU with SMs connected via router interfaces to a NoC fabric

Ultimately, sufficient bandwidth to feed the SMs must be provided via on-chip communication systems such as a Network-on-Chip (NoC). On-chip communication systems are measured by their energy efficiency (pJ/bit-mm) and density (bits/s-mm). Conventional signaling over on-chip wires is dense but energy inefficient. Energy consumption in a CMOS-driven wire is ~ CV^2, where C is the capacitance per unit length of a wire (about 200fF/mm and independent of scaling) and V is the supply voltage. In present day chips, energy efficiency for on-chip wires is between 20-40 fJ/bit-mm, and it will not improve, since supply voltages are scaling very slowly. Consider again the example GPU system shown in Figure 3, feeding the 64×64 matrix multiplies in each SM requires about 50GB/s of bandwidth, a total of 4TB/s of bandwidth for the 80SMs. For a large chip, the average Manhattan wiring distance is 20mm so the net communication requirement is 80TB-mm/s. At 40fJ/bit-mm, this requires 26W. Since the V^2 term in the CMOS-driven wire energy consumption is $V_{supply} \times V_{signal}$, we can get some benefit from low-swing signaling. In practice this may reduce energy/bit by ½, at the expense of circuit complexity and is likely to be practical only for long-range on-chip communications.

Charge recycling signaling (CRS)[12] 'stacks' CMOS wire repeaters (Figure 4) so that the supply current is used to supply 2 (or more) communications paths in series. Since both supply and signal voltages are divided by 2, we can achieve a 4× improvement in energy efficiency. To make this work, the current consumption in the two halves of the stack must be exactly equal; we demonstrated [12] the idea of swapping traffic between the two layers to achieve this balance with very simple CMOS inverter circuitry and no external control or supply circuitry.

When data must be moved long distances on-chip, special semi-custom communications paths ('fabrics') are generally used to help conserve power and area. With many wires running in parallel in the same direction, cross-talk between neighboring wires accumulates along the fabric and quickly degrades timing margins, because the interacting signals are correlated in time and space. This problem is typically handled by inserting re-clocking stages that 'reset' timing between clock and data periodically along the fabric. Fabrics often have symmetric traffic flowing in both directions; if neighboring wires are carrying contra-flowing traffic (Figure 5), cross-talk does not cause accumulation of timing error, since the interactions between wires are not correlated. This contra-flow idea [12] allows very long fabrics to operate without the need for periodic re-timing, saving the considerable power of the re-clocking stages. At present, the barrier to implementing these energy-saving methods is lack of support for special signaling methods in timing verification tools, a solvable problem.

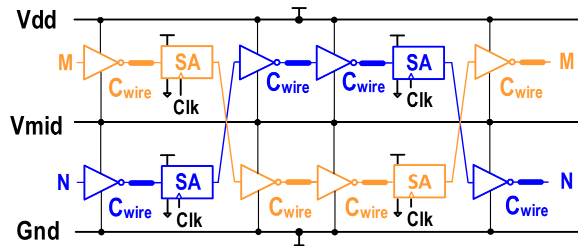

Figure 4: Balanced charge-recycling signaling

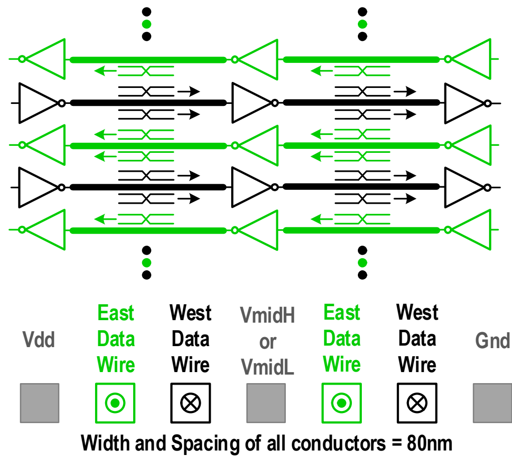

Figure 5: Balanced charge-recycling bus with contraflow

These 'fabrics' satisfy on-chip bandwidth requirements by using four routing layers to provide bandwidth density greater than 4GB/s/µm, thereby using only 1mm of die width/height to achieve 4TB/s of bandwidth. By dedicating four routing layers for both vertical and horizontal routing, we provide porosity which allows these 'fabrics' to cross each other.

On-Module Communication

At the next level of our system, we envision multiple GPU chips connected to each other, and to DRAMs on a multi-chip module [2] as shown in the Figure 6 example. Each GPU requires 1TB/s of DRAM bandwidth, and this number increases with GPU performance. The inter-GPU links require a comparable amount of bandwidth to provide a relatively flat memory system.

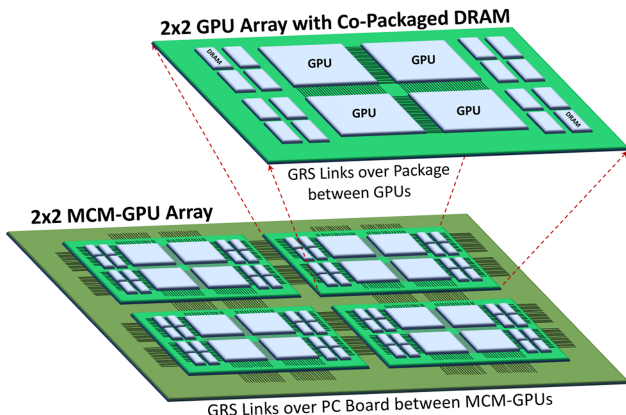

Figure 6: On-package and Short-reach off-package communication links

The key metrics of a short-reach signaling system are bits/s-mm of chip periphery and energy/bit. Multi-chip GPU modules require short reach signaling that is dense (1Tb/s-mm) and energy efficient (1pJ/b). Total power is limited by packaging constraints on power networks and by heat removal. Packaging technology also confines signal pins to the periphery of a chip; in practice there are only a few hundred usable signal pins on each of the four edges of a large GPU die mounted on an organic package or a thousand pins per side on a silicon interposer. Since signal pins are severely limited, signaling speed (bits/s) must be maximized. To support 1TB/s over a few hundred pins requires signaling rates of 20Gb/s or more. To keep signaling as a low percentage of chip power, total signaling power can be no higher than a few 10s of watts, so energy efficiency of 1pJ/bit or better is needed. Conventional short-distance, high-speed serial links are typically 4-6pJ/bit.

Single-ended (SE) signaling is an attractive option for short-reach links, requiring only 1 pin/signal, but power-efficient SE links are difficult to build. Ground-referenced signaling (GRS) [8] [10] [11] avoids many of the usual problems of SE signaling. The ground network, typically the lowest impedance network in a system, is used as the signaling *voltage reference*, eliminating the need for a precise matched reference at the receiver. Return currents flow only on ground, providing a *clean line termination*. Signals must be driven symmetrically positive and negative with respect to ground; GRS uses an output-multiplexed pair of charge pumps, shown in Figure 7, to produce these voltage levels. Each pump alternately charges a capacitor to the supply voltage, then dumps the charge into the line with polarity determined by its data bit. Since the charge-pump transmitter draws the same charging current from the supply regardless of data polarity, *simultaneous switching noise* is largely eliminated. GRS links operate robustly at speeds exceeding 25Gb/s/pin dissipating about 1pJ/bit and can be used to connect chips on an MCM or closely spaced packaged chips over a PC board.

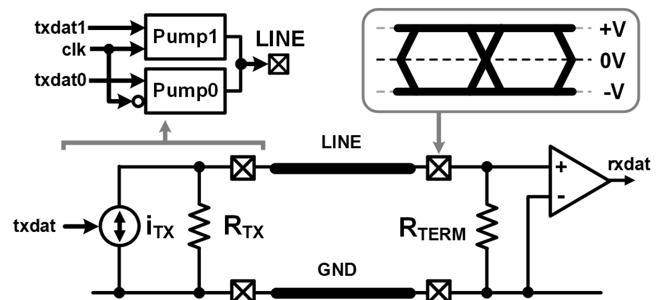

Figure 7: Ground Referenced Signaling

Another strategy to improve pin efficiency is sending multiple bits of data on each transmitted symbol. PAM4, for example, encodes 2 data bits on 4 voltage levels (Figure 8), thus doubling bits/s per pin.

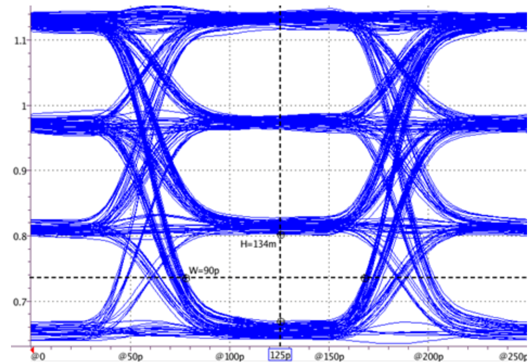

Figure 8: PAM4 data transmission.

PAM4 requires 3 references that must be maintained in the face of system variations. The horizontal eye opening in PAM4 signals is also reduced, thanks to multiple overlapping signal trajectories, so timing precision may need to be nearly as high as for PAM2 running at 2× the symbol rate. PAM4 is energetically unfavorable; the available signal swing is divided into 3 parts, effectively reducing the swing per bit by a factor of 1/3, or -10dB. PAM4 was first used only in cases where channel attenuation between ½ Nyquist and Nyquist fell by more than -10dB. Today it is being deployed in advanced links, such as those for electro-optical interfaces, because transistor switching speed is no longer increasing with scaling, and higher modulation factor is the only

way to continue to increase bandwidth per pin. Based on recent published experimental links, PAM4 costs about 4× energy per bit to achieve a 2× increase in bits/s/pin.

INTER-MODULE COMMUNICATION

At the top level of our system, multiple GPU modules are connected. If *data parallelism* is exploited, gradients for every weight in the network must be exchanged for each training batch. Consider training ResNet 50 with a batch size of 1024 on 128 nodes (a sub-batch size of 8 per node). Each node must exchange 25M gradients after processing 8 inputs – about 64GOPS. With 120TOPS per GPU and 4 communications per GPU per exchange, each GPU requires 400GB/s of I/O bandwidth for parameter exchange.

As at the lower levels of the system, this communication must be dense and energy efficient. The key to achieving energy efficiency at this level of the system is co-design of the channel and the signaling system. A conventional channel involving package balls, printed circuit board striplines with several through-hole vias, and connectors has numerous impedance discontinuities that require heroic equalization – and commensurate power levels of 10-15pJ/bit.

A cleaner channel can be realized by connecting modules directly to one another using flexible printed-circuit board links [3]. Connecting from the top-side of one module to the top-side of a second module eliminates the through-module vias, the package balls, the printed circuit board vias, and the connectors. The resulting channel is very clean – with just frequency dependent attenuation. A variation of GRS can deal with up to 15dB of this attenuation, and repeaters can be inserted in longer flex links to give greater reach. Using dense packaging with liquid cooling reduces the required reach for each link.

Figure 9 shows two PCBs with GPUs connected by flex circuit links from top side of module to top side of module. Flex circuits can be constructed to provide 3-to-4 signaling lanes per mm. When signaling at 25Gbps, 40mm of package edge can support 400-500GB/s of I/O bandwidth per GPU. The relatively smooth frequency characteristics of the flex interconnect and smooth transitions from PCB to flex enable the use of simple equalization techniques and energy efficient links less than 2pJ/b.

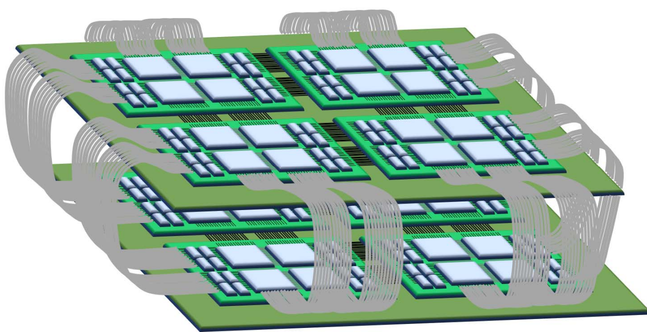

Figure 9: Flex printed circuit board links

CONCLUSION

The current revolution in deep learning has been fueled by the availability of hardware fast enough to train deep networks in a reasonable amount of time. For inference, the core operations are low precision multiply-adds and can be realized with energy less than 10fJ per "connection". A typical vision network can be realized with power dissipation of 30mW. The main circuit challenge is to co-design the data representation and the circuit to minimize the energy of the core operation.

Training modern networks in a reasonable amount of time requires clusters of many GPUs. The core operations are 16-bit floating point multiply-adds – with 32-bit floating point accumulation. Training is communication intensive placing a premium on dense, efficient communication circuits at the chip, module, and system level. Optimized signaling can reduce the cost of on-chip signaling by 4× and off-chip signaling by 5-8× compared to conventional approaches.

ACKNOWLEDGMENTS

The research described here benefitted from the work of many of our teammates at NVIDIA. This research was developed, in part, with funding from the Defense Advanced Research Projects Agency (DARPA). The views, opinions, and/or findings contained in this article/presentation are those of the author(s)/presenter(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.
.

REFERENCES

[1] Amodei, D., et al. "Deep speech 2: End-to-end speech recognition in English and Mandarin." *International Conference on Machine Learning*. 2016.

[2] Arunkumar, A., et al., "MCM-GPU: Multi-Chip-Module GPUs for Continued Performance Scalability," *ISCA* 2017.

[3] Braunisch, H., et al., "High-Speed Flex-Circuit Chip-to-Chip Interconnects," *Trans. Advanced Packaging*, Feb. 2008.

[4] He, K., et al. "Deep residual learning for image recognition," *CVPR*, 2016.

[5] Keckler, S., et al. "GPUs and the future of parallel computing." *IEEE Micro* 31.5 (2011): 7-17.

[6] Krizhevsky A., et al., "ImageNet Classification with Deep Convolutional Neural Networks," *NIPS,* 2012.

[7] LeCun, Y., et al. "Backpropagation applied to handwritten zip code recognition." *Neural computation* 1.4 (1989): 541-551.

[8] Poulton, J., et al., "A 0.54pJ/bit 20Gbit/sec Ground-Referenced Single-Ended Short Haul Serial Link in 28nm CMOS for Advanced Packaging Applications," *JSSC*, Dec. 2013.

[9] Silver, D., et al., "Mastering the game of Go with deep neural networks and tree search." *Nature*, *529*(7587), pp.484-489, 2016.

[10] Turner, W., et al., "Ground-Reference Signaling for Intra-Chip and Short-Reach Chip-to-Chip Interconnects," *CICC*, 2018.

[11] Wilson, J., et al., "A 1.17pJ/b 25Gb/s Ground-Referenced Single-Ended Serial Link for Off- and On-Package Communication in 16nm CMOS Using a Process- and Temperature-Adaptive Voltage Regulator," *ISSCC*, 2018.

[12] Wilson, J., et al., "A 6.5-to-23.3fJ/b/mm Balanced Charge-Recycling Bus in 16nm FinFET CMOS at 1.7-to-2.6Gb/s/wire with Clock Forwarding and Low-Crosstalk Contraflow Wiring," *ISSCC*, 2016.