LSU EE 7722 **Homework 3** **Due: 29 April 2024**

In class we spent much of the semester showing how recent generation GPUs can be used for matrix/matrix multiplication, though understanding that they were designed for a larger class of computations. In the next week we will look at some accelerators designed specifically for machine learning workloads that are dominated by matrix/matrix multiplication. These include Google's TPU chips, Cerebras wafer-scale systems, and Tesla's DOJO. Google's TPU chips have been developed over several generations so one can assume that they are a good approach for accelerating ML workloads. Cerebras is a newer product, so one must wait a few years to see if its approaches are truly useful.

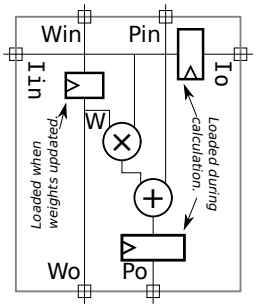
The NVIDIA GPUs, the Google TPUs, Cerebras' systems, and DOJO each take a different approach to accelerating these workloads.

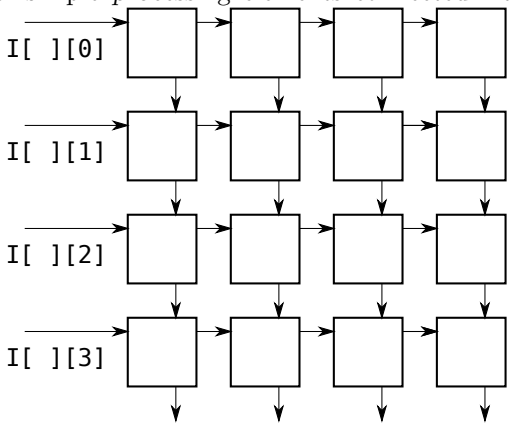
One way to look at them is by the path an operand takes to a functional unit. In the NVIDIA GPUs and operand's starting point can be DRAM, the L2 cache, the L1 cache (or shared memory), or a register. First consider a register. Each thread can read up to 255 registers. So the hardware must provide a path for each of those 255 registers to the operand ports of an FP32 unit (plus other functional units). The amount of energy it takes to move an operand from a register to the functional unit increases with the number of registers. So, if a thread could only access 16 registers the energy needed to move an operand from a register to a functional unit would be lower. Energy is often a limiter, for the obvious reason that electricity isn't free, but also because it determines an upper limit on clock frequency beyond which the chip can't kept from overheating.

As we've noted in class, with more registers one can hide more latency, which may be one reason NVIDIA GPUs have so many compared to CPU ISAs. (To make a fair comparison in a GPU one must consider the maximum number of registers accessible by all warps, since the register file must provide a register for all of them. For a CPU there are two other factors. First, Intel-64 implementations have two contexts. So, that doubles the number of registers. Another factor is that many Intel cores are dynamically scheduled, so when comparing the number of registers one must look at the size of the physical register file. To keep things simple we won't consider CPU ISAs here.)

The GPU L1 cache is larger (and varies by device). Energy per access is increased by the size itself and the fact that one must do a tag-store lookup to see if the data is cached. Even more energy is needed if the data is brought from the L2, and more from DRAM.

What sets the different accelerators apart then is the path taken by an operand. The simplest approach is a systolic array. A systolic array consists of of simple *processing elements* connected into a 2D mesh:





A systolic array can be used to compute many different things, the one illustrated above can be used for matrix/matrix multiplication. Notice that the value provided to each functional unit (multiplier or adder) is read from one register. That takes much less energy. Clearly one needs more than just a systolic array to make an accelerator. But the approach used by Google's TPUs is to have most of the computation done by big systolic arrays, realizing significant energy savings. Google's original TPU had one 256×256 systolic array capable of operating on 8b integer values, accumulating to 32-bit values [4]. Later designs had two or four 128×128 systolic arrays, augmented by 128-element vector units [3].

Intermediate between the conventional (relatively) memory hierarchy of the NVIDIA GPUs and the minimalist systolic array, is an old idea: a mesh (2D array) connection of simple processing elements, but not as simple as those in a systolic array.

Two examples are the Cerebras wafer-scale systems [5] and Tesla's DOJO [7]. These will be discussed in class next week (the week of 29 April 2024).

For this assignment, and to prepare for the final exam read the following papers. First, read the papers describing Google's TPU accelerators, all of which use large systolic arrays. The first paper, Jouppi 17 [4], describes Google's first TPU, now called TPUv1. Though TPUv1 is primitive compared to later designs, the paper does provide more detail, and so will be the basis of some questions in this assignment. That is, you need to understand that paper fairly well. The TPUv1 systolic array was limited to integer arithmetic, limiting its use to inference. Google's next GPUs TPUv2 and TPUv3 were designed to handle both inference and training. To support training these chips use systolic arrays that operate on 16-bit floating-point data, the BF16 (brain-float) format. These are described in several papers, see [3, 6], concentrate on [3]. Google has developed a TPUv4, those who are curious can read [2] and [1], but reading those papers is not required.

Several things set Cerebras systems apart from other accelerators. First, they span an entire wafer, not just a chip. (Normally chips are fabricated on wafers, which are then cut up into chips. In a wafer-scale system the wafer is not cut up, the entire wafer is used.) Start reading about Cerebras in [5] before Wednesday.

You should be able to get copies of all of these papers for free on campus. Off campus you might be asked to pay. Please E-mail me if you have any problems getting a free copy.

Problem 1: Answer the following questions about Google's TPUs as described by Norman Jouppi and his many collaborators. Several of the questions below are based on the TPUv1 paper [4].

In the matrix multiply code used in class (in `mm.cu`) we worked with two sizes.

(*a*) Describe the problem with multiplying matrix A, 96×32, by matrix B, 32×29700 on the TPUv1. Matrix elements are 8 bit integers, so data type is not the problem.

☐ Which matrix should be the weights? (Note that either A or B could be considered the weights.)

☐ How many cycles will it take to complete the multiplication?

☐ What fraction of the systolic array will be utilized?

(*b*) Describe the problem with multiplying matrix A, 1536×512, by matrix B, 512×2970 on the TPUv1. Matrix elements are 8 bit integers, so data type is not the problem.

☐ Which matrix should be the weights? (Note that either A or B could be considered the weights.)

☐ How many cycles will it take to complete the multiplication?

☐ What fraction of the systolic array will be utilized?

References:

[1] Jouppi, N., Kurian, G., Li, S., Ma, P., Nagarajan, R., Nai, L., Patil, N., Subramanian, S., Swing, A., Towles, B., Young, C., Zhou, X., Zhou, Z., and Patterson, D. A. TPU v4: An optically reconfigurable supercomputer for machine learning with hardware support for embeddings. In *Proceedings of the 50th Annual International Symposium on Computer Architecture* (New York, NY, USA, 2023), ISCA '23, Association for Computing Machinery. `https://doi.org/10.1145/3579371.3589350`.

[2] Jouppi, N. P., Hyun Yoon, D., Ashcraft, M., Gottscho, M., Jablin, T. B., Kurian, G., Laudon, J., Li, S., Ma, P., Ma, X., Norrie, T., Patil, N., Prasad, S., Young, C., Zhou, Z., and Patterson, D. Ten lessons from three generations shaped googles TPUv4i : Industrial product. In *2021 ACM/IEEE 48th Annual*

International Symposium on Computer Architecture (ISCA) (2021), pp. 1–14. http://dx.doi.org/10.1109/ISCA52012.2021.00010.

[3] Jouppi, N. P., Yoon, D. H., Kurian, G., Li, S., Patil, N., Laudon, J., Young, C., and Patterson, D. A domain-specific supercomputer for training deep neural networks. *Commun. ACM 63*, 7 (June 2020), 6778. https://doi.org/10.1145/3360307.

[4] Jouppi, N. P., Young, C., Patil, N., Patterson, D., Agrawal, G., Bajwa, R., Bates, S., Bhatia, S., Boden, N., Borchers, A., Boyle, R., Cantin, P.-l., Chao, C., Clark, C., Coriell, J., Daley, M., Dau, M., Dean, J., Gelb, B., Ghaemmaghami, T. V., Gottipati, R., Gulland, W., Hagmann, R., Ho, C. R., Hogberg, D., Hu, J., Hundt, R., Hurt, D., Ibarz, J., Jaffey, A., Jaworski, A., Kaplan, A., Khaitan, H., Killebrew, D., Koch, A., Kumar, N., Lacy, S., Laudon, J., Law, J., Le, D., Leary, C., Liu, Z., Lucke, K., Lundin, A., MacKean, G., Maggiore, A., Mahony, M., Miller, K., Nagarajan, R., Narayanaswami, R., Ni, R., Nix, K., Norrie, T., Omernick, M., Penukonda, N., Phelps, A., Ross, J., Ross, M., Salek, A., Samadiani, E., Severn, C., Sizikov, G., Snelham, M., Souter, J., Steinberg, D., Swing, A., Tan, M., Thorson, G., Tian, B., Toma, H., Tuttle, E., Vasudevan, V., Walter, R., Wang, W., Wilcox, E., and Yoon, D. H. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th Annual International Symposium on Computer Architecture* (New York, NY, USA, 2017), ISCA '17, ACM, pp. 1–12. http://doi.acm.org/10.1145/3079856.3080246.

[5] Lie, S. Cerebras architecture deep dive: First look inside the hardware/software co-design for deep learning. *IEEE Micro 43*, 3 (2023), 18–30. http://dx.doi.org/10.1109/MM.2023.3256384.

[6] Norrie, T., Patil, N., Yoon, D. H., Kurian, G., Li, S., Laudon, J., Young, C., Jouppi, N., and Patterson, D. The design process for Google's training chips: TPUv2 and TPUv3. *IEEE Micro 41*, 2 (2021), 56–63. http://dx.doi.org/10.1109/MM.2021.3058217.

[7] Talpes, E., Sarma, D. D., Williams, D., Arora, S., Kunjan, T., Floering, B., Jalote, A., Hsiong, C., Poorna, C., Samant, V., Sicilia, J., Nivarti, A. K., Ramachandran, R., Fischer, T., Herzberg, B., McGee, B., Venkataramanan, G., and Banon, P. The microarchitecture of DOJO, Tesla's exa-scale computer. *IEEE Micro 43*, 3 (2023), 31–39. http://dx.doi.org/10.1109/MM.2023.3258906.