AI 芯片 – AI 计算体系

比特位

ZOMI

Talk Overview

1. AI 计算体系

- 深度学习计算模式
- 计算体系与矩阵运算

2. AI 芯片基础

- 通用处理器 CPU
- 从数据看 CPU 计算
- 通用图形处理器 GPU
- AI专用处理器 NPU/TPU
- 计算体系架构的黄金10年

Talk Overview

I. AI 计算体系与矩阵运算

- Key Metrics – AI芯片关键指标
- Matrix Multiplication – 矩阵运算
- Bit Width – 比特位数
- Specialized Hardware – 专用硬件

Question?

四年前的我

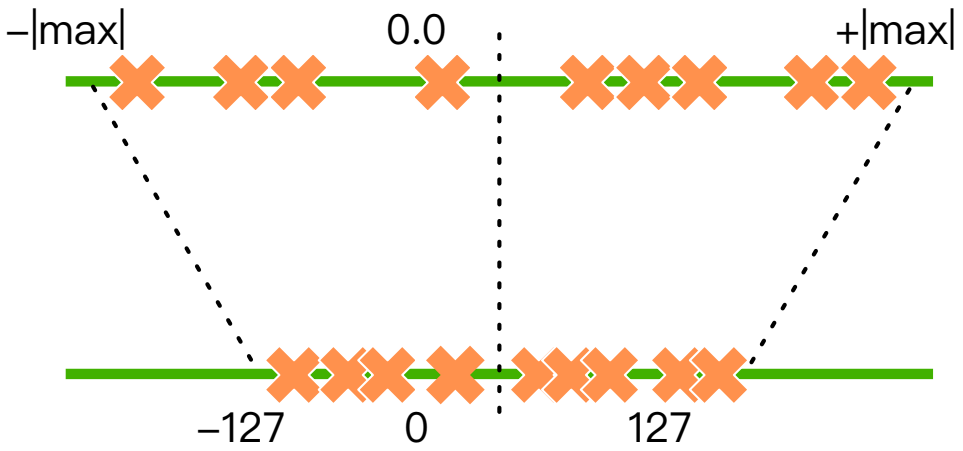
- 为什么硬件不提供 Int8 的指令？！
- 为什么硬件不支持 Int8 的比特为数？！
- 那我搞得量化算法怎么落地？怎么加速？
- 硬件同事赶紧搞搞起来！Int4 最好也支持起来！我和混合 bit(int4/int8) 加速！

比特位数
Bits Width

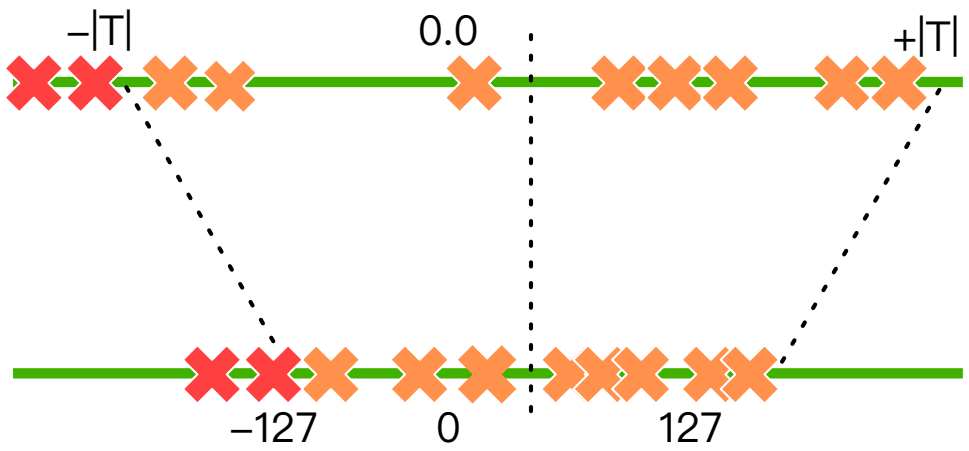
量化原理

- 模型量化桥接了定点与浮点，建立了一种有效的数据映射关系，使得以较小的精度损失代价获得了较好的收益

No Saturation: map |max| to 127



Saturation above | threshold| to 127



量化原理

量化方法比较

量化方法	功能	经典适用场景	使用条件	易用性	精度损失	预期收益
量化训练 (QAT)	通过 Finetune 训练将模型量化误差降到最小	对量化敏感的场景、模型，例如目标检测、分割、OCR 等	有大量带标签数据	好	极小	减少存续空间4X，降低计算内存
静态离线量化 (PTQ Static)	通过少量校准数据得到量化模型	对量化不敏感的场景，例如图像分类任务	有少量无标签数据	较好	较少	减少存续空间4X，降低计算内存
动态离线量化 (PTQ Dynamic)	仅量化模型的可学习权重	模型体积大、访存开销大的模型，例如 BERT 模型	无	一般	一般	减少存续空间2/4X，降低计算内存

什么决定比特位宽 Bit Width ?

- 在AI流程里，不同数据会使用不同类型格式，才能保证精度达标。

- **训练 Training**: weights, activation, partial sums, gradients and weight update.
- **推理 Inference**: weights, activations, partial sums

- （所需的精度可能因数据类型而异 ）

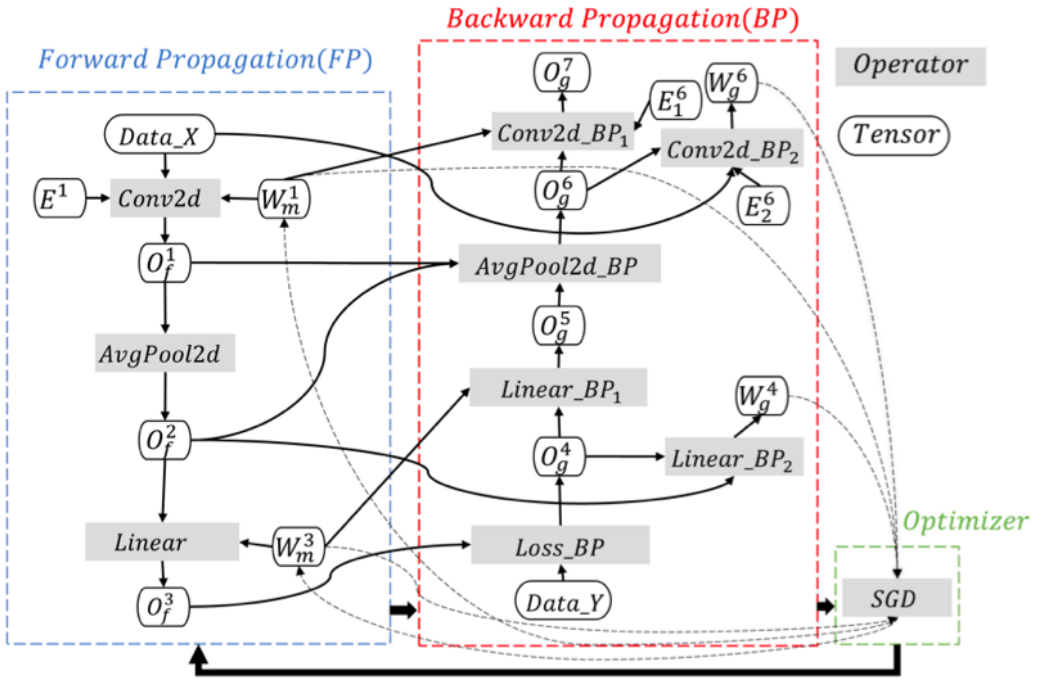
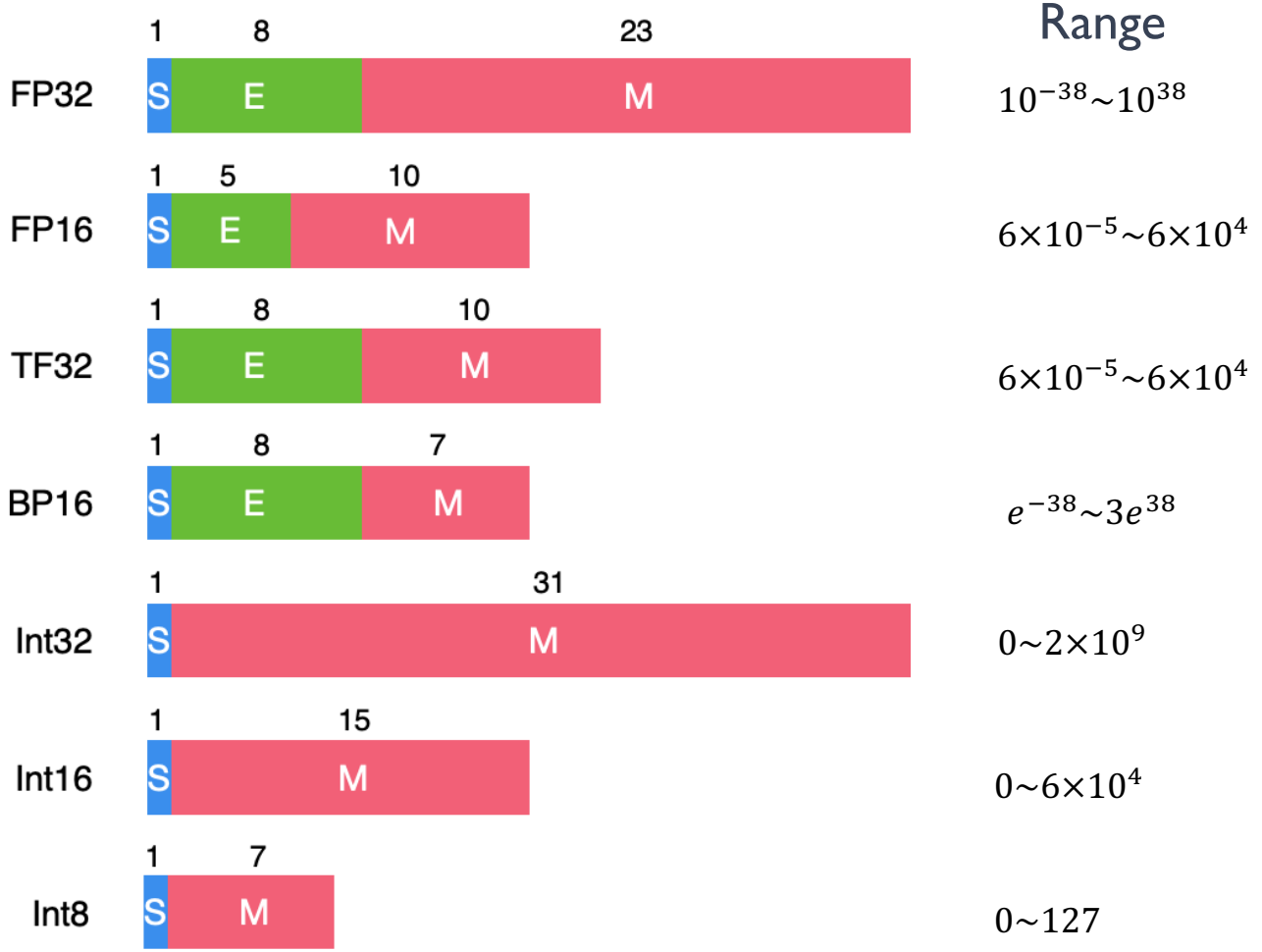


Figure 3: Computation graph for training the DL model in Figure 2. Ovals represent tensors in which W stands for weight tensor, O for In/Out tensor, and E for ephemeral tensor. Rectangles are operators.[2] Dash lines denote weight updates by SGD.

什么决定比特位宽 Bit Width ?

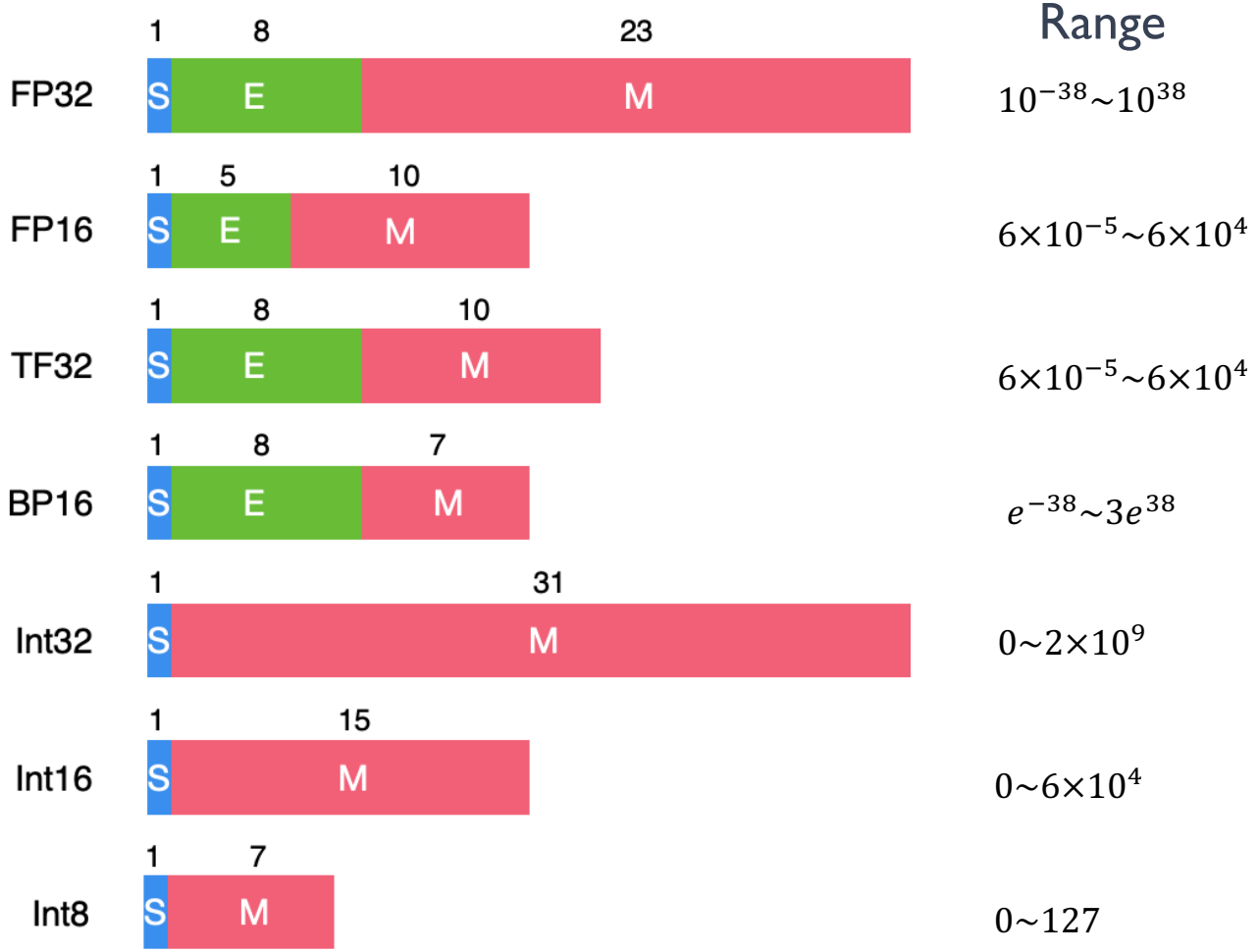
- Number of unique values Precision
  - e.g., M-bits to represent 2^M values
- Dynamic range of values
  - e.g., E-bits to scale value by $2^{(E-127)}$
- Signed or unsigned values
  - e.g., signed requires one extra bit(S)
- **总比特数： S+E+M**



什么决定比特位宽 Bit Width ?

- **Floating Point(FP)** allows range to change for each value(E-bits)
- **Fixed Point(Int)** has fixed range

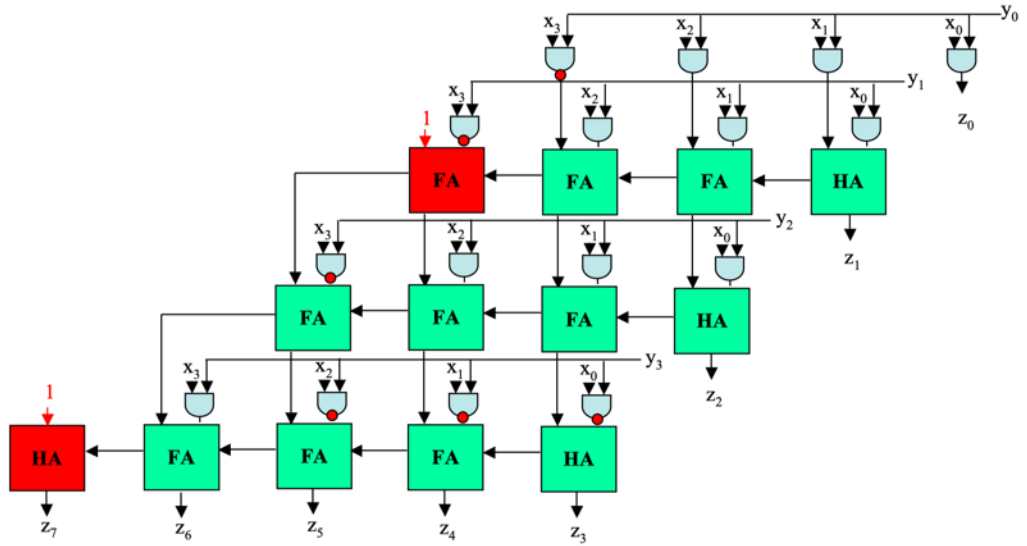
- CPU 或者 GPU 默认采用 FP32



降低精度（降低 bit width）

- 对于 MAC 的输入和输出，能够有效减少数据的搬运和存储开销
  - Smaller Memory -> Lower energy
- 减少 MAC 计算的开销和代价
  - e.g., int8 x int8 -> int16；fp16 x fp16 -> fp32

$$
\begin{array}{rcccccccl}
 & & & & x_3 & x_2 & x_1 & x_0 & \text{Multiplicand} \\
 & & & \times & y_3 & y_2 & y_1 & y_0 & \text{Multiplier} \\
\hline
 & & & x_3y_0 & x_2y_0 & x_1y_0 & x_0y_0 & & \\
 & & x_3y_1 & x_2y_1 & x_1y_1 & x_0y_1 & & & \text{Partial Product} \\
 & x_3y_2 & x_2y_2 & x_1y_2 & x_0y_2 & & & & \\
+ & x_3y_3 & x_2y_3 & x_1y_3 & x_0y_3 & & & & \\
\hline
z_7 & z_6 & z_5 & z_4 & z_3 & z_2 & z_1 & z_0 & \text{Result}
\end{array}
$$



降低位宽对功耗和芯片面积的影响

Relative Energy Cost

Operation	Energy(pJ)
8b Add	0.03
16b Add	0.05
32b Add	0.1
16b FP Add	0.4
32b FP Add	0.9
8b Multiply	0.2
32b Multiply	3.1
16b FP Multiply	1.1
32b FP Multiply	3.7
32b SRAM Read	5
32b DRAM Read	640

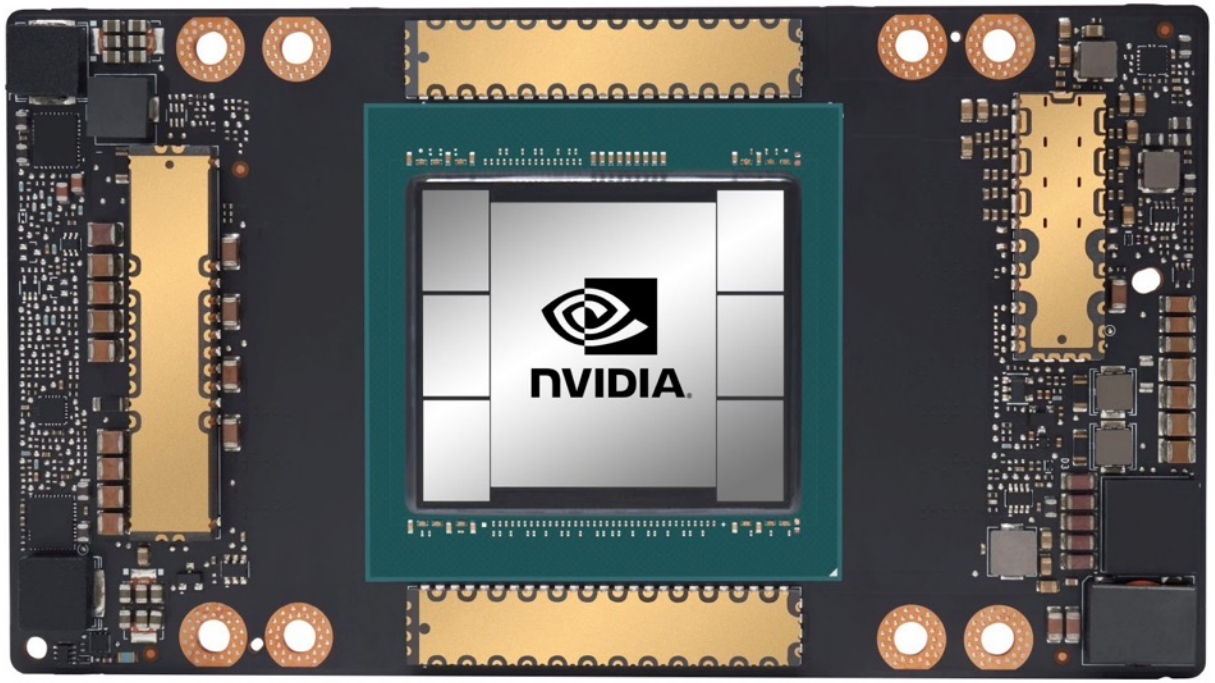
Relative Area Cost

Operation	Area(um^2)
8b Add	36
16b Add	67
32b Add	127
16b FP Add	1360
32b FP Add	4184
8b Multiply	282
32b Multiply	3495
16b FP Multiply	1640
32b FP Multiply	7700
32b SRAM Read	N/A
32b DRAM Read	N/A

市面产品

- 市面上已经推出 8-bit 推理 & 16-bit float 训练的产品

华为昇腾 910



NVIDIA A100

AI 芯片设计的思考

- **Reduce number of unique values – Precision(M-bits)**
  - Default: Uniform quantization (values are equally spaced out)
  - None-uniform quantization (spacing can be computed)
  - Fewer unique values can make transforms and compression more effective
- **Reduce dynamic range(E-bits)**
  - If Possible, fix range (used fixed point)
  - Share range across group of values (weights for a layer or channel)
- Tradeoff between number of bits allocated to **M-bits and E-bits**

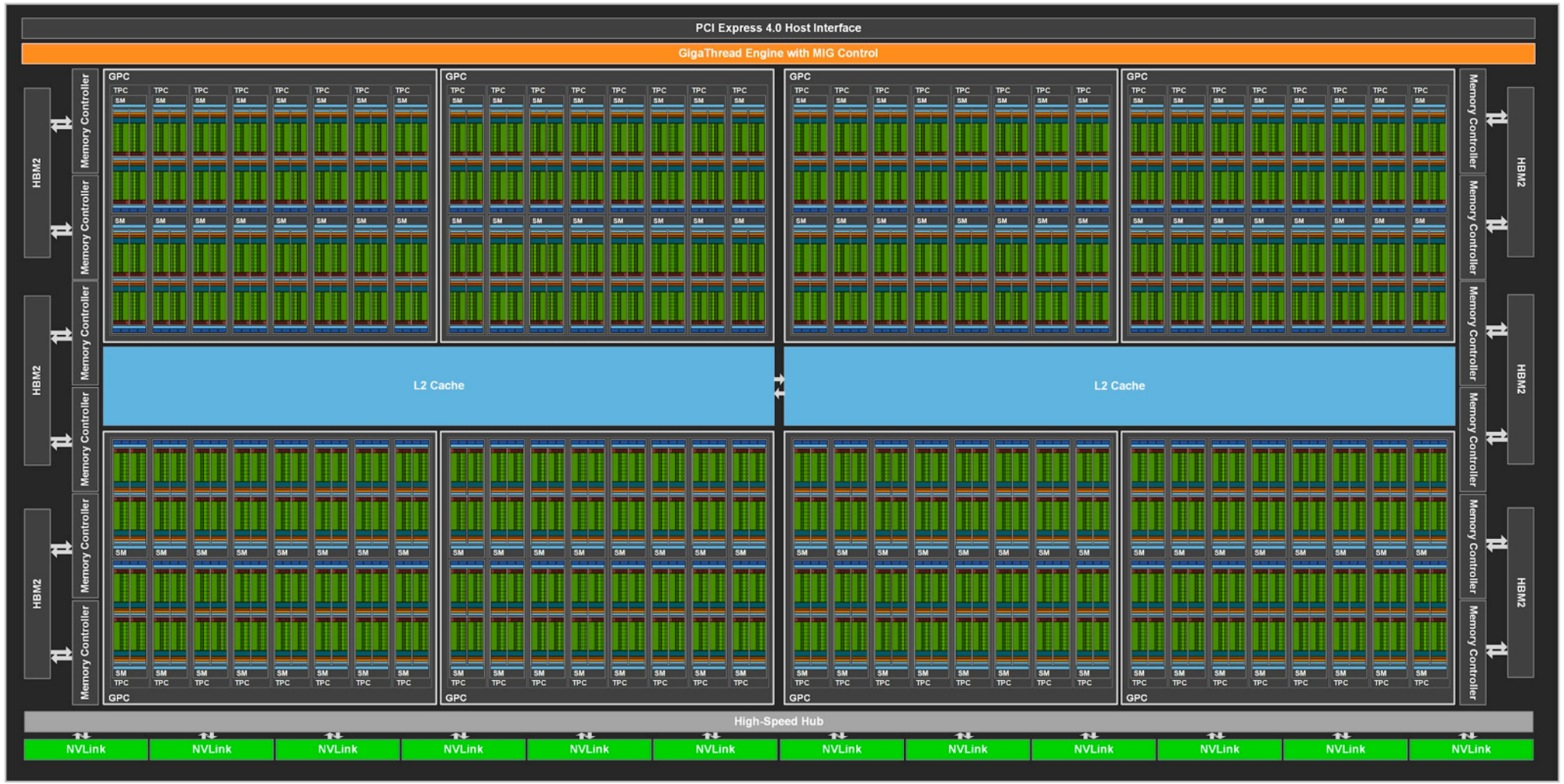
S=1, E=5, M=10, Total bits=32

S=1, E=8, M=7, Total bits=19

AI 芯片设计的思考

- **对精度的影响 Impact on Accuracy**
  - 需要考虑不同数据集（NLP/CV）、不同任务
  - 不同网络模型之间的差异进行测评（e.g., classification > detection）
- **训练和推理的数据位宽**
  - 32bit float 可以作为弱基线；
  - 对于训练使用 FP16、BF16、TF32；
  - 推理 CV 任务以 int8 为主，NLP 以 FP16为主，大模型 int8/FP16 混合；
- **权衡硬件的成本开销**
  - 支持额外的数据位宽需要引入更多的电路
  - 新增多少额外的数据位宽合适？

AI 芯片设计的思考



AI 芯片设计的思考

Question?

四年前的我

- 为什么硬件不提供 Int8 的指令？！
- 为什么硬件不支持 Int8 的比特为数？！
- 那我搞得量化算法怎么落地？怎么加速？
- 硬件同事赶紧搞搞起来！Int4 最好也支持起来！我和混合 bit(int4/int8) 加速！

现在得我

- everything is not that simple ~
- 这是个系统工程，需要考虑AI计算体系、硬件架构、系统成本等XXX

Data Sparsity

数据稀疏

数据稀疏性

- **减少 MACs 计算**
  - 0×0 都为零，此类计算可以减少 MACs
  - 减少不必要计算，从而降低功耗

- **减少数据搬运**
  - 如果发现一个数据为 0，可以避免对另外数据的搬运
  - 只传输/搬运非 0 数据

- CPU/GPU/NPU 对随机稀疏计算并不友好，因此对稀疏矩阵计算需要专用硬件

引用