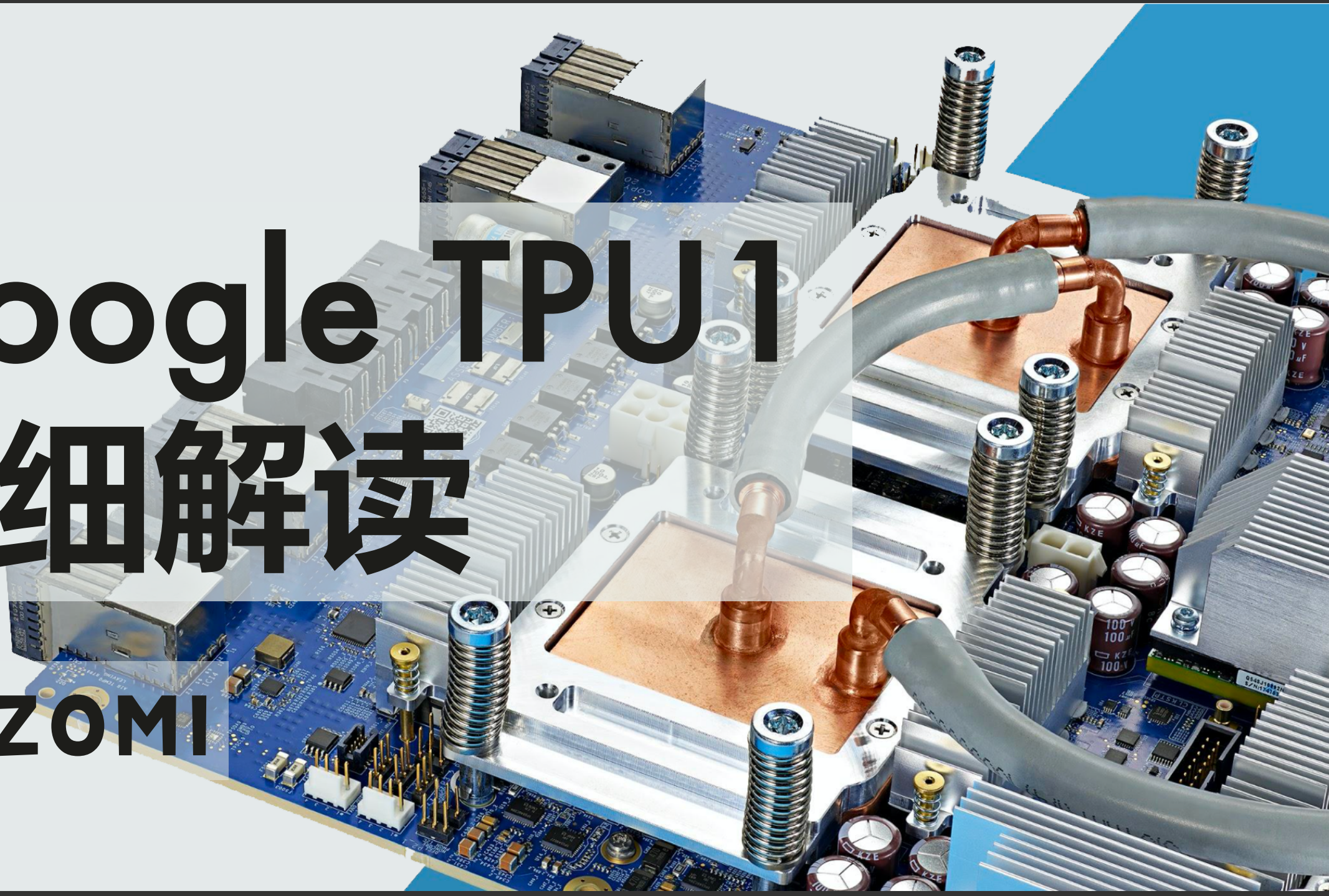
Google TPU1 详细解读



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
   - 通用图形处理器 GPU
   - AI专用处理器 NPU/TPU
3. **GPU详解**
   - 英伟达GPU架构发展
   - Tensor Core和NVLink
4. **国外 AI 芯片**
   - 特斯拉 DOJO 系列
   - 谷歌 TPU 系列
5. **国内 AI 芯片**
   - 壁仞科技芯片架构
   - 寒武纪科技芯片架构
6. **AI芯片的思考**
   - SIMD&SIMT与编程体系
   - AI芯片的架构思路与思考

Talk Overview

I. 国外 AI 芯片

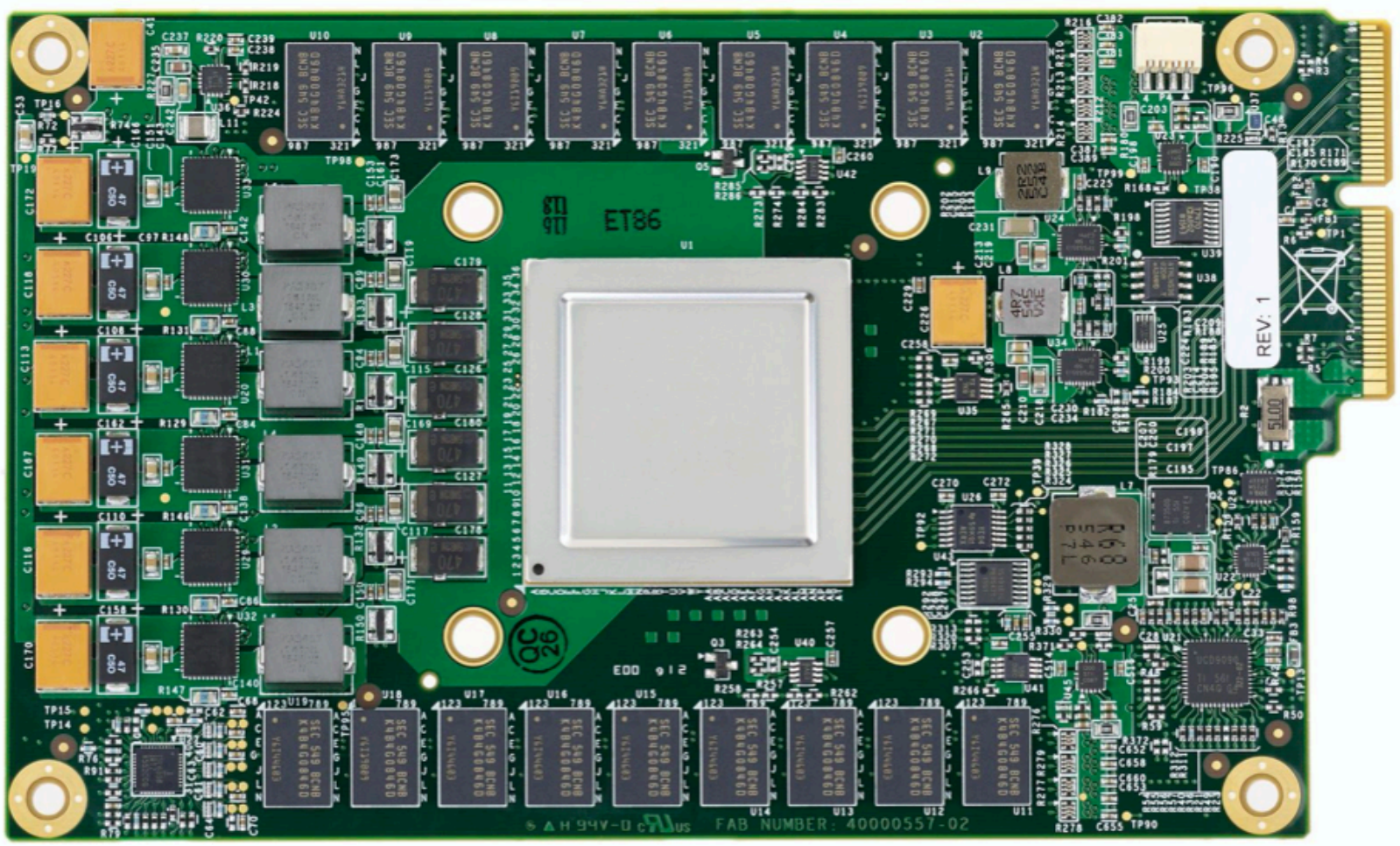
- 英伟达 GPU 芯片架构剖析
- 特斯拉 DOJO 芯片架构剖析
- 谷歌 TPU 芯片架构剖析

Talk Overview

I. 国外 AI 芯片

- 英伟达 GPU 芯片架构剖析
- 特斯拉 DOJO 芯片架构剖析
- 谷歌 TPU 芯片架构剖析
  - TPU 历史发展
  - TPU1 脉动阵列细节
  - TPU2 第一款训练卡
  - TPU3 性能 POD 超算
  - TPU4 超级互联

TPU Printed Circuit Board

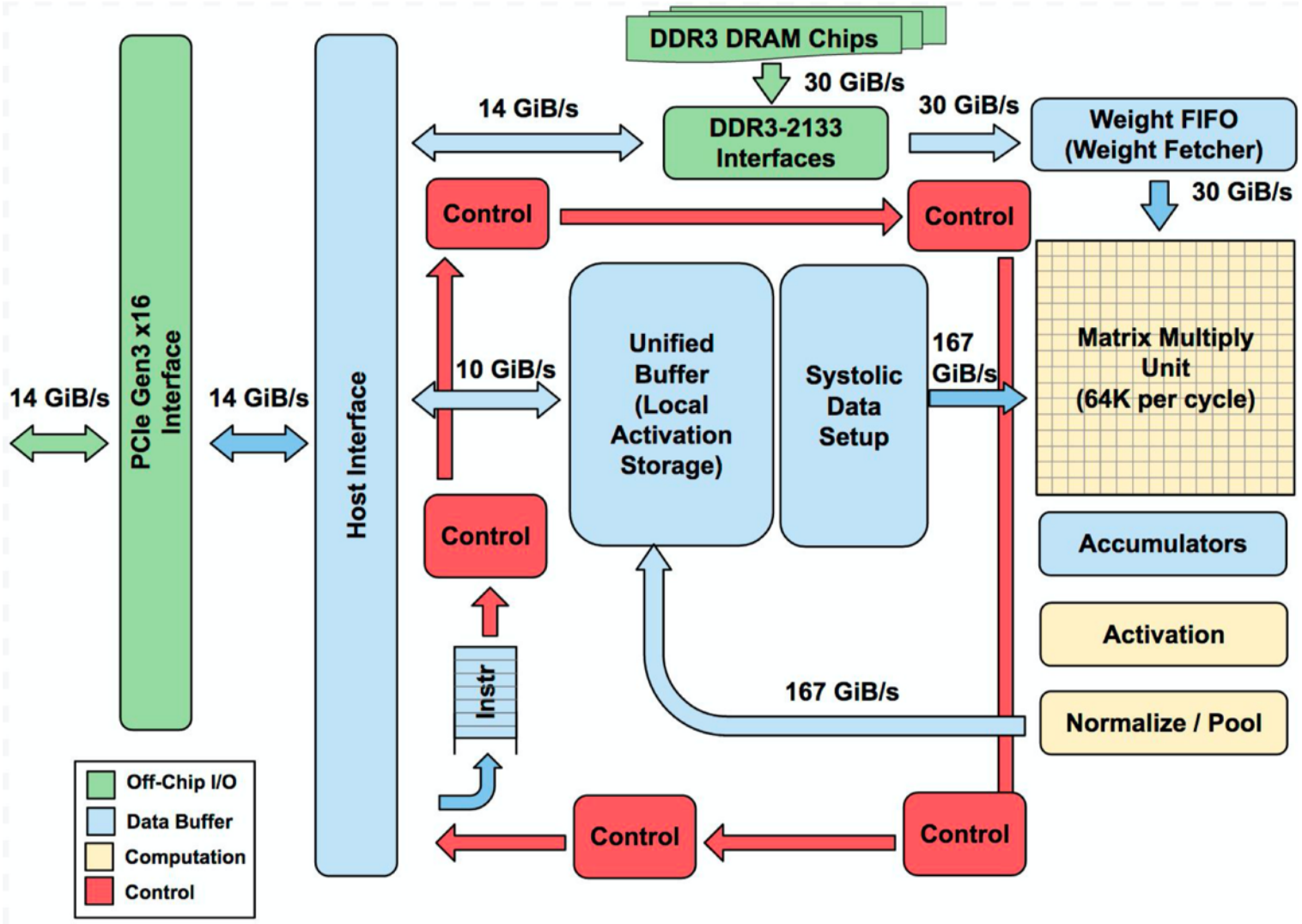


TPU Application

TPU Application

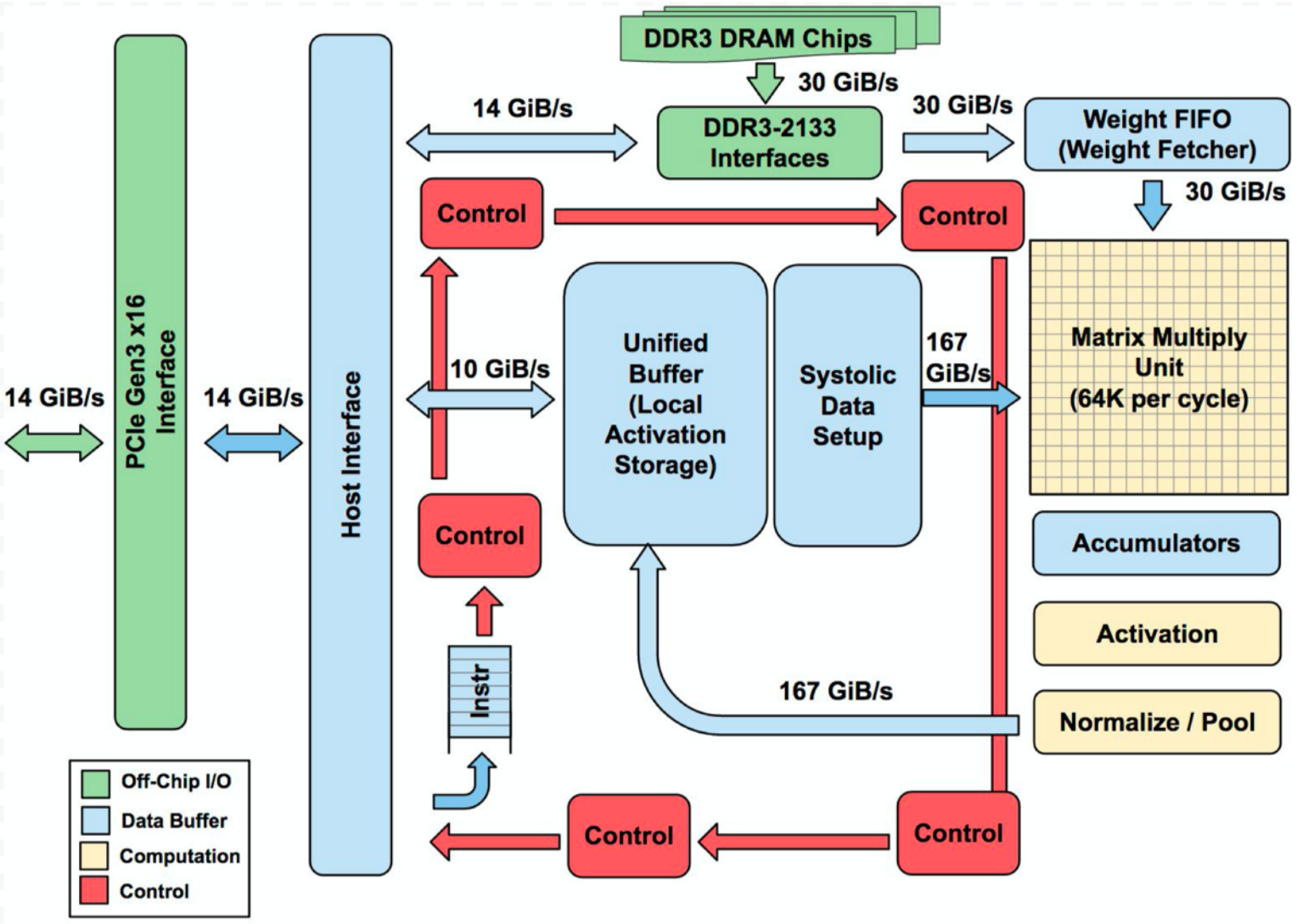
1. TPU1芯片架构

TPU1 芯片架构


DDR3 DRAM Chips
30 GiB/s
14 GiB/s
DDR3-2133 Interfaces
30 GiB/s
Weight FIFO (Weight Fetcher)
30 GiB/s
Control
Control
14 GiB/s
PCIe Gen3 x16 Interface
14 GiB/s
Host Interface
10 GiB/s
Unified Buffer (Local Activation Storage)
Systolic Data Setup
167 GiB/s
Matrix Multiply Unit (64K per cycle)
Control
Accumulators
Activation
Instr
167 GiB/s
Normalize / Pool
Control
Control
Off-Chip I/O
Data Buffer
Computation
Control

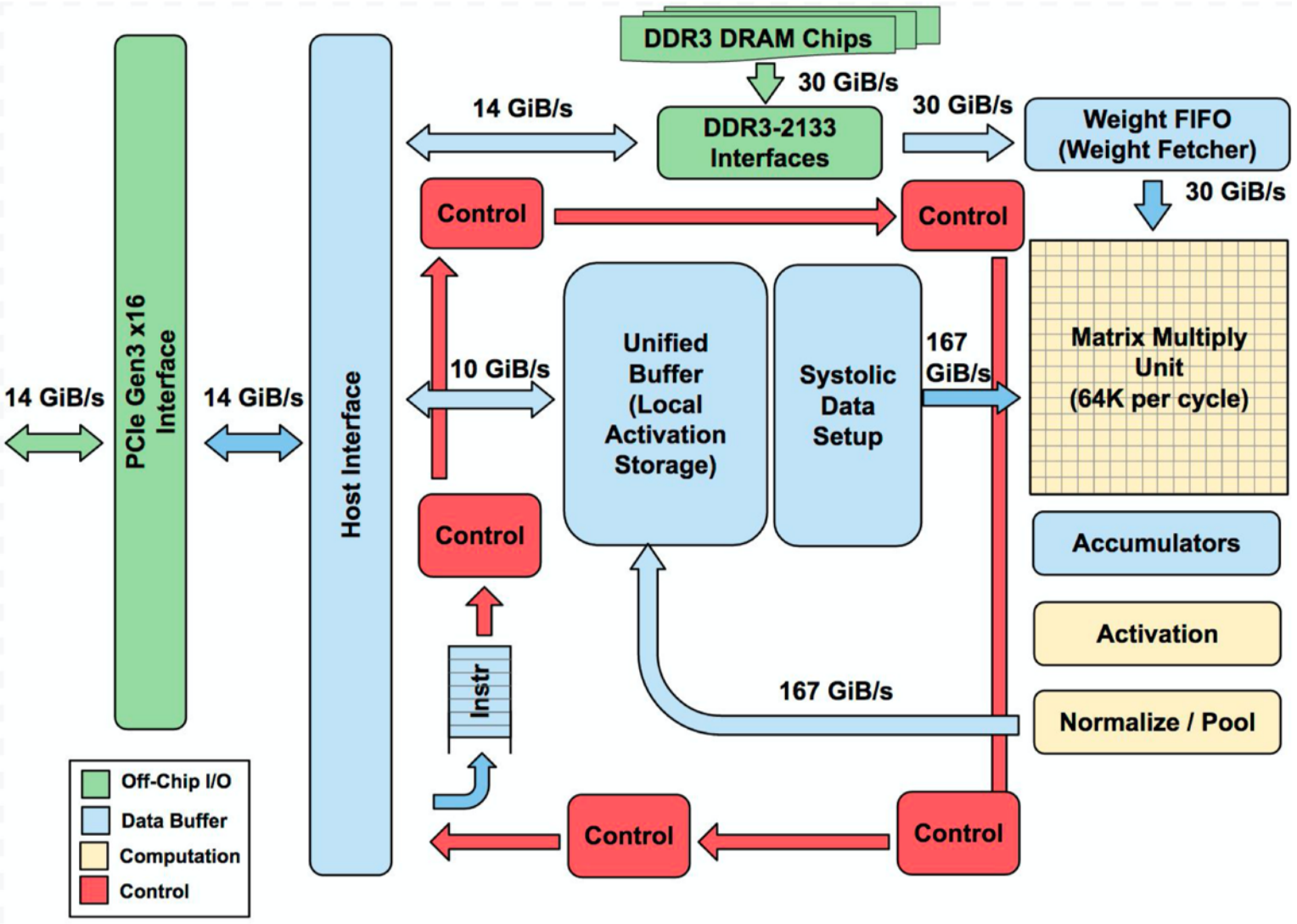
TPU1 芯片架构

- Weight FIFO：主要负责从 8GB off-chip DDR3 DRAM 上读取 Weight 权重参数进入计算单元 MM Unit；



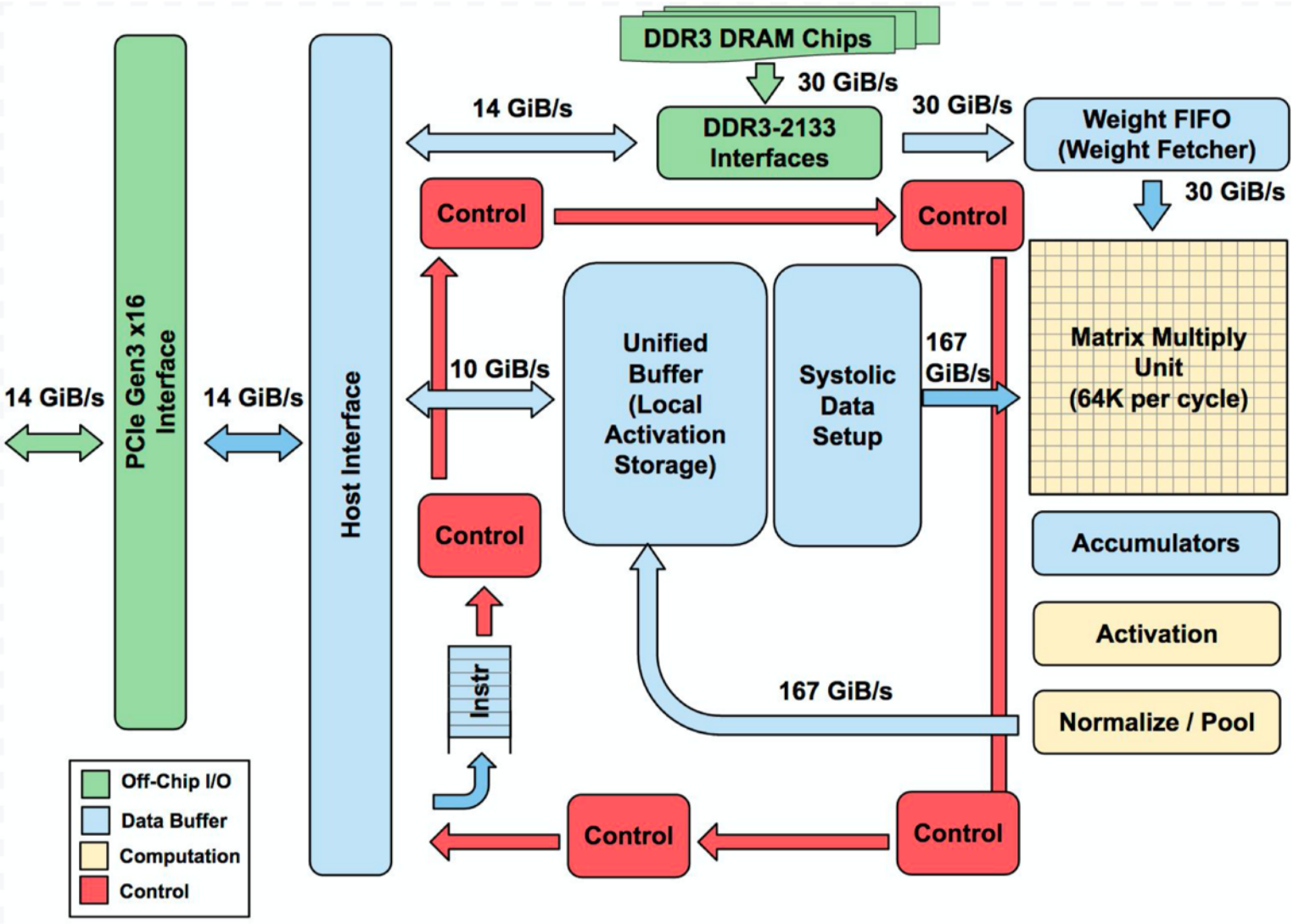
TPU1 芯片架构

- MMU：提供 256 x 256 x 8bit 的乘加计算，以脉动阵列形式；
- 每周期可以计算 256 个乘法结果，计算出结果为 16bit；
- 矩阵单元中包含一个 64KB Weight Tile，以及一个双缓存单元；



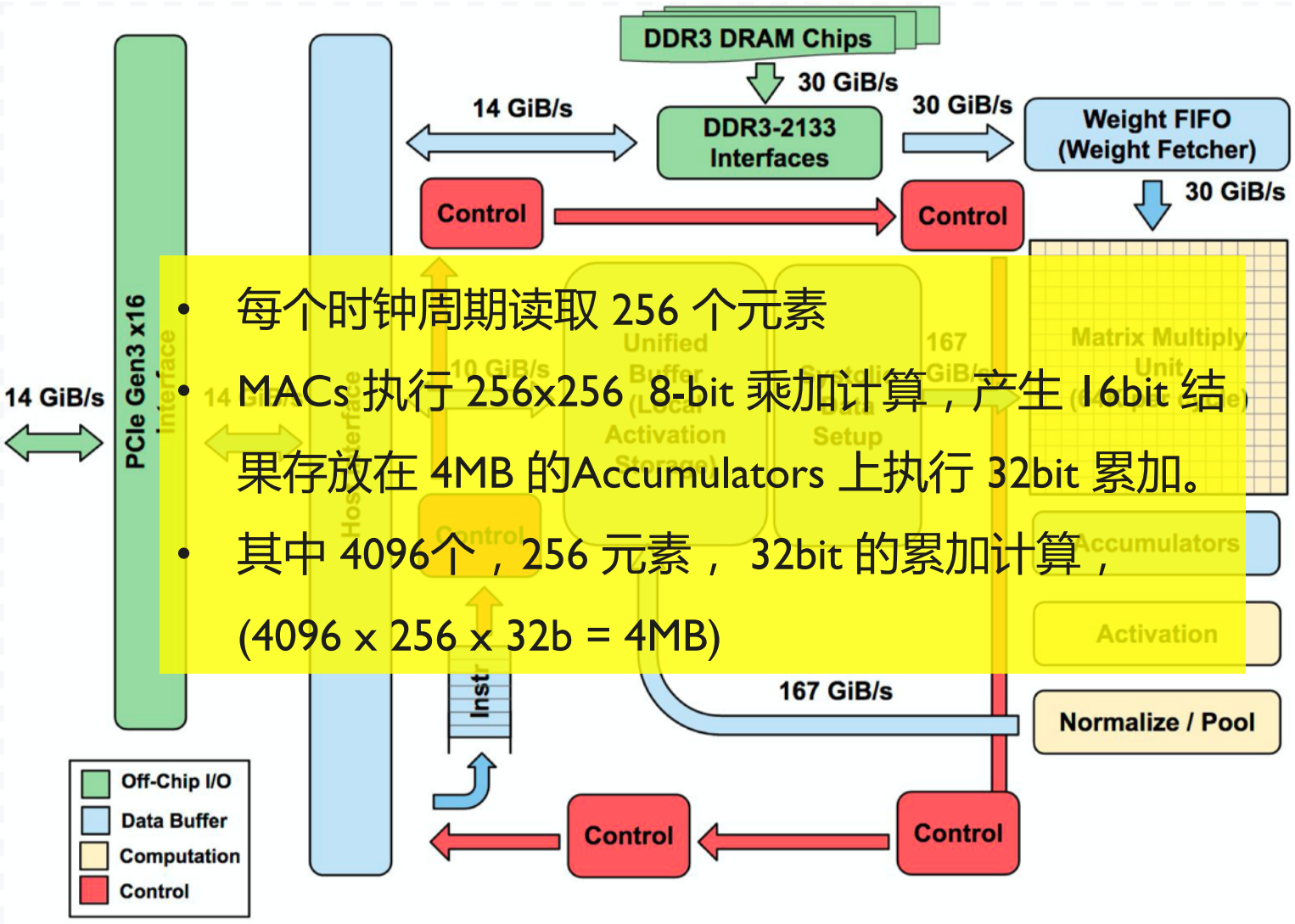
TPU1 芯片架构

- Accumulators：提供 4 MiB，4098 x 256 x 32 bit 大小的累加单元，用来收集乘法产生 16 bit 结果；
- 4096：单次 MM 操作需要2048个单元，双缓存则需要 4096 个单元才能满足计算需求；



TPU1 芯片架构

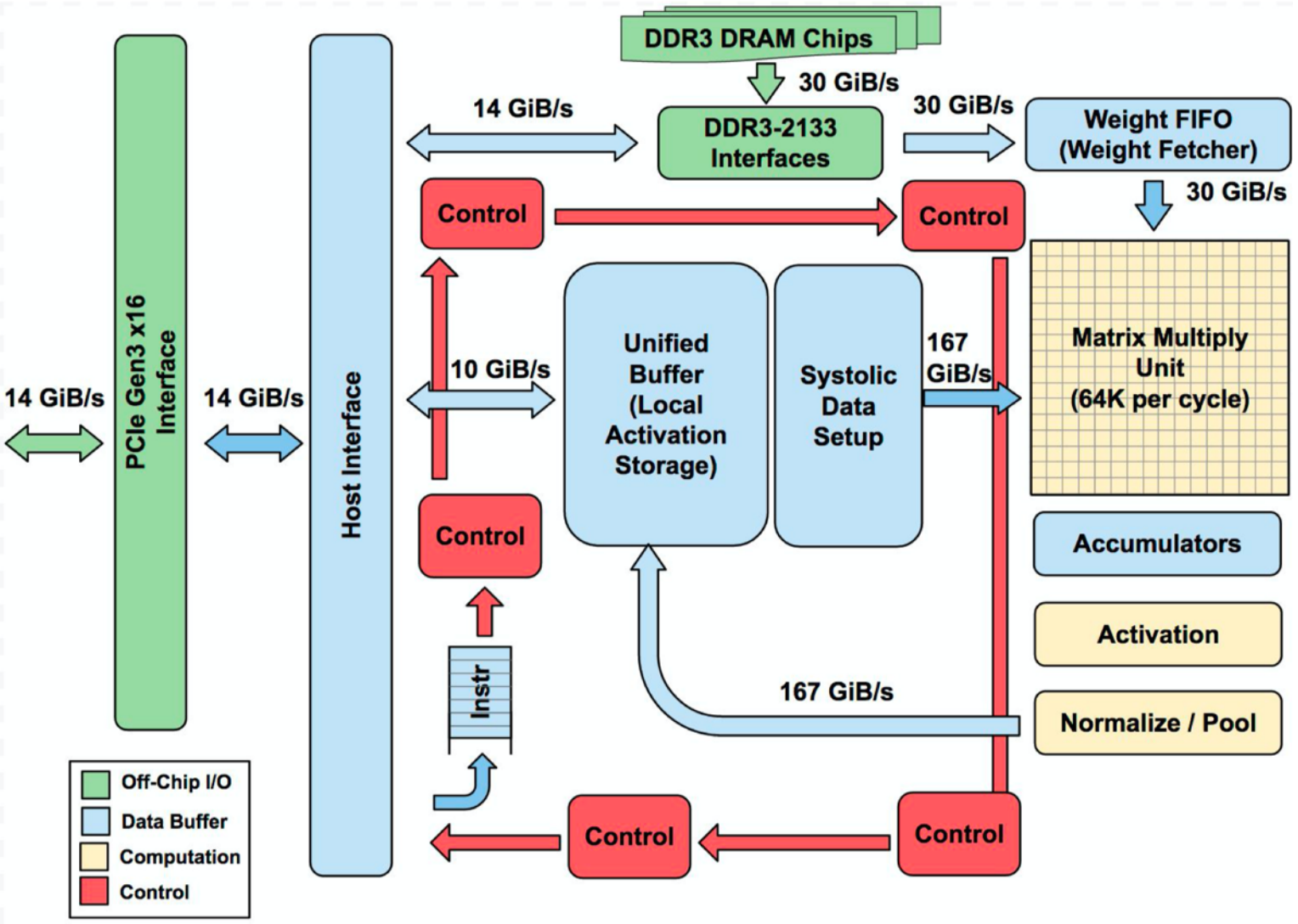
- Accumulators：提供 4 MiB，4098 x 256 x 32 bit 大小的累加单元，用来收集乘法产生 16 bit 结果；
- 4096：单次 MM 操作需要2048个单元，双缓存则需要 4096 个单元才能满足计算需求；



- 每个时钟周期读取 256 个元素
- MACs 执行 256x256 8-bit 乘加计算，产生 16bit 结果存放在 4MB 的Accumulators 上执行 32bit 累加。
- 其中 4096个，256 元素， 32bit 的累加计算，(4096 x 256 x 32b = 4MB)

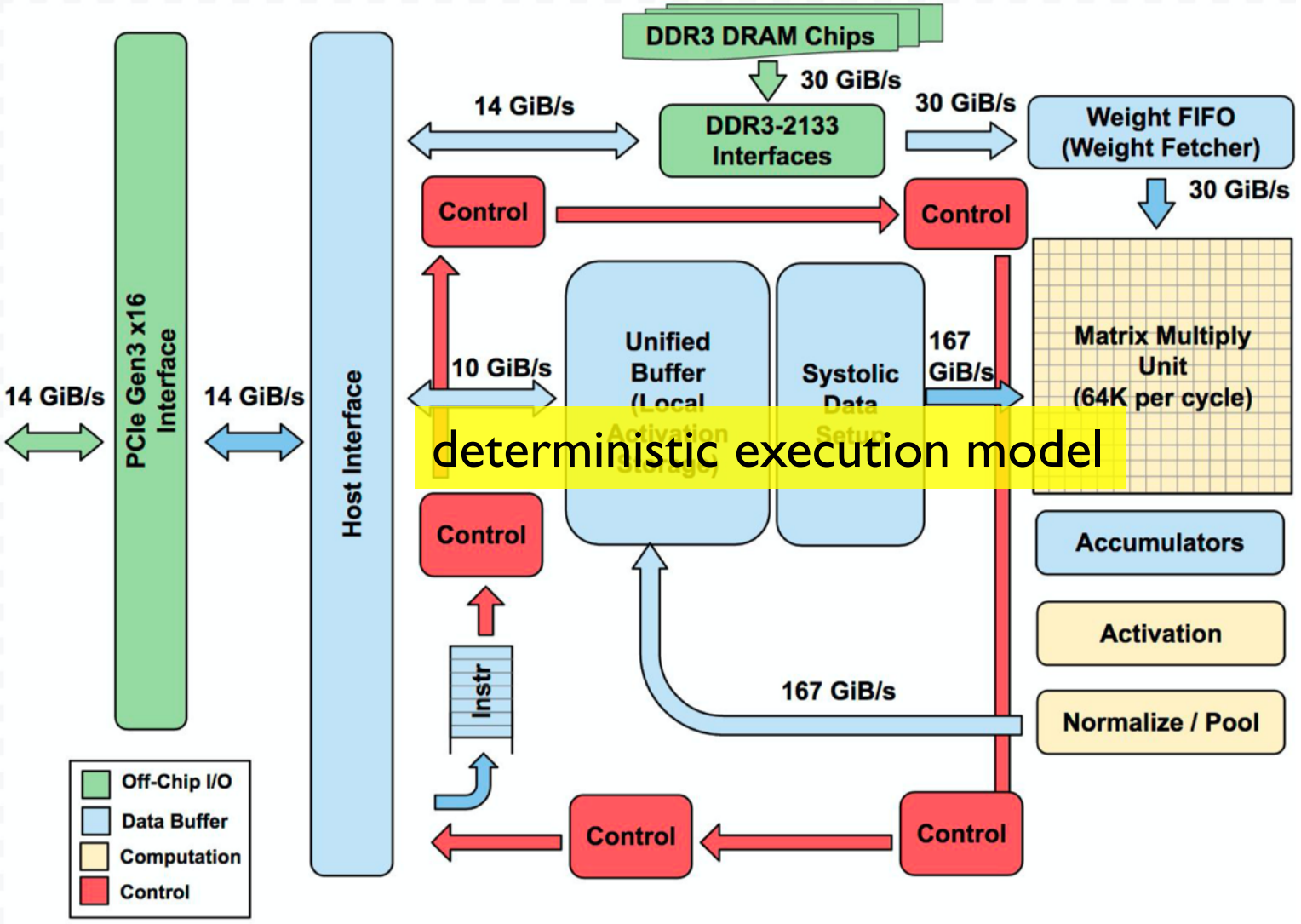
TPU1 芯片架构

- **控制指令 Control**：指令通过 PCIe 总线从 host 主机传到 TPU Core 中，以4级流水线执行。
- **指令来源**：CPU 向 TPU 发送指令让它执行
- **CSIC 指令**：指令采用CSIC 指令集类型，总共有12条指令



TPU1 芯片架构

- **控制指令 Control**：指令通过 PCIe 总线从 host 主机传到 TPU Core 中，以4级流水线执行。
- **指令来源**：CPU 向 TPU 发送指令让它执行
- **CSIC 指令**：指令采用CSIC 指令集类型，总共有12条指令



TPU CISIC 关键指令

CISC 每条指令平均时钟周期（ CPI ，clock cycles per instruction ）为10~20，为了控制 MUX、UB 和 AU 等模块进行计算，Google 定义了十几个专门为神经网络推理而设计的高级指令：

1. Read_Host_Memory：从 CPU host 读取数据到 Unified Buffer
2. Read_Weights：从 Weight DRAM 读取数据到 Weight FIFO
3. MatrixMultiply/Convolve：执行乘法或卷积计算
4. Activate：执行ReLU，Sigmoid等激活计算
5. Write_Host_Memory：把计算结果数据从 Unified Buffer 输出到 CPU host

MM计算：矩阵操作中，大小为 B x 256 矩阵乘以 256*256 权重矩阵，得到 B x 256 输出，共需要 B 个流水线时钟周期

TPU Instruction Set Architecture v1.5

TPU Instruction Set Architecture v1.5

https://cloud.google.com/blog/products/ai-machine-learning/an-in-depth-look-at-googles-first-tensor-processing-unit-tpu

Revision 6

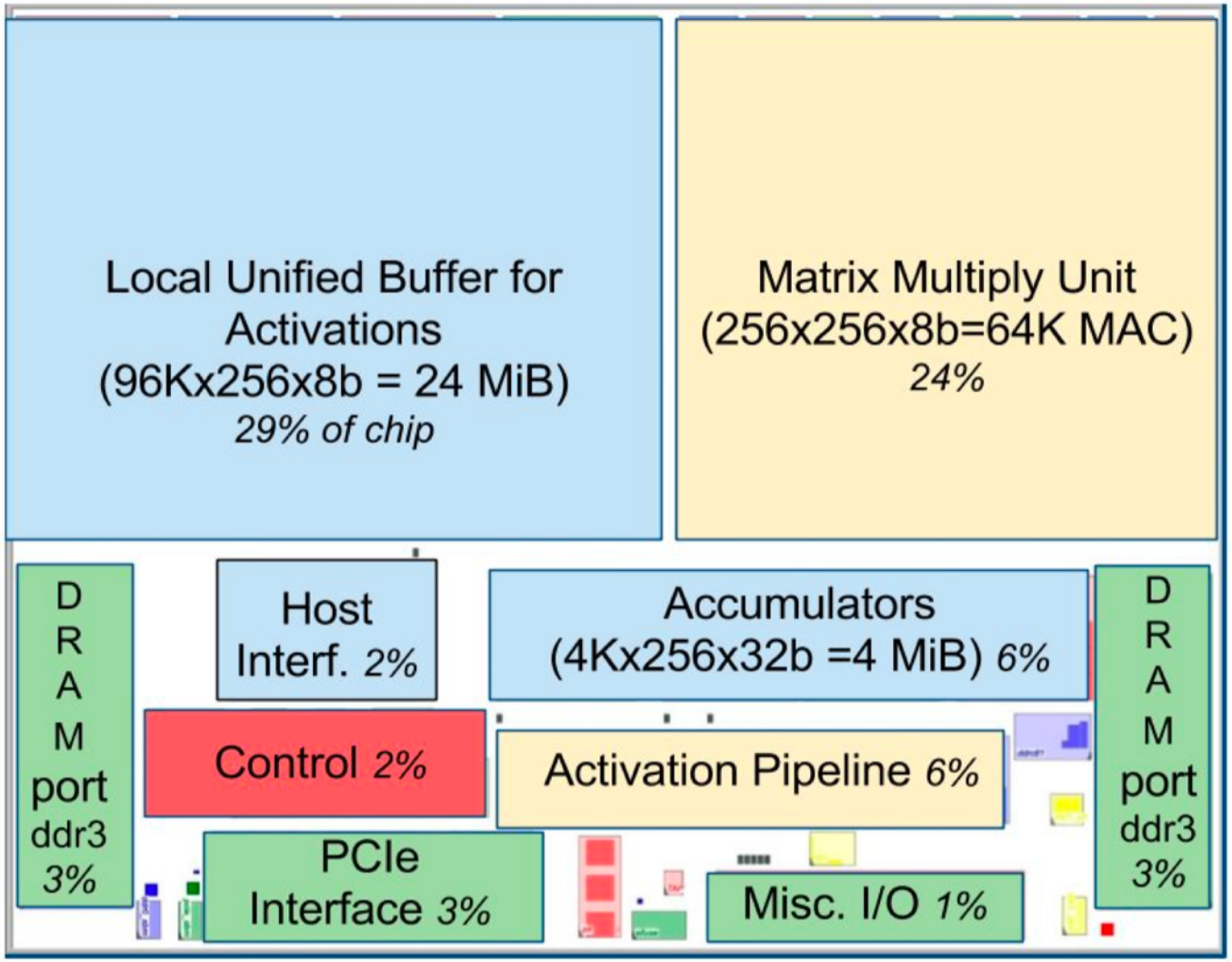
May 23rd 2016

Domipheus Labs

http://labs.domipheus.com

TPU1 芯片布局图：专用电路和大量缓存，适合推理工作流程

- 由于读取大的 SRAM 比矩阵计算，需要消耗更多的能耗，矩阵单元使用收缩执行，通过减少统一缓冲区的读写来节省功耗；
- TPU的控制单元更小，只占了整个 Floor Plan 2%，给片上存储和计算单元留下更多的空间；

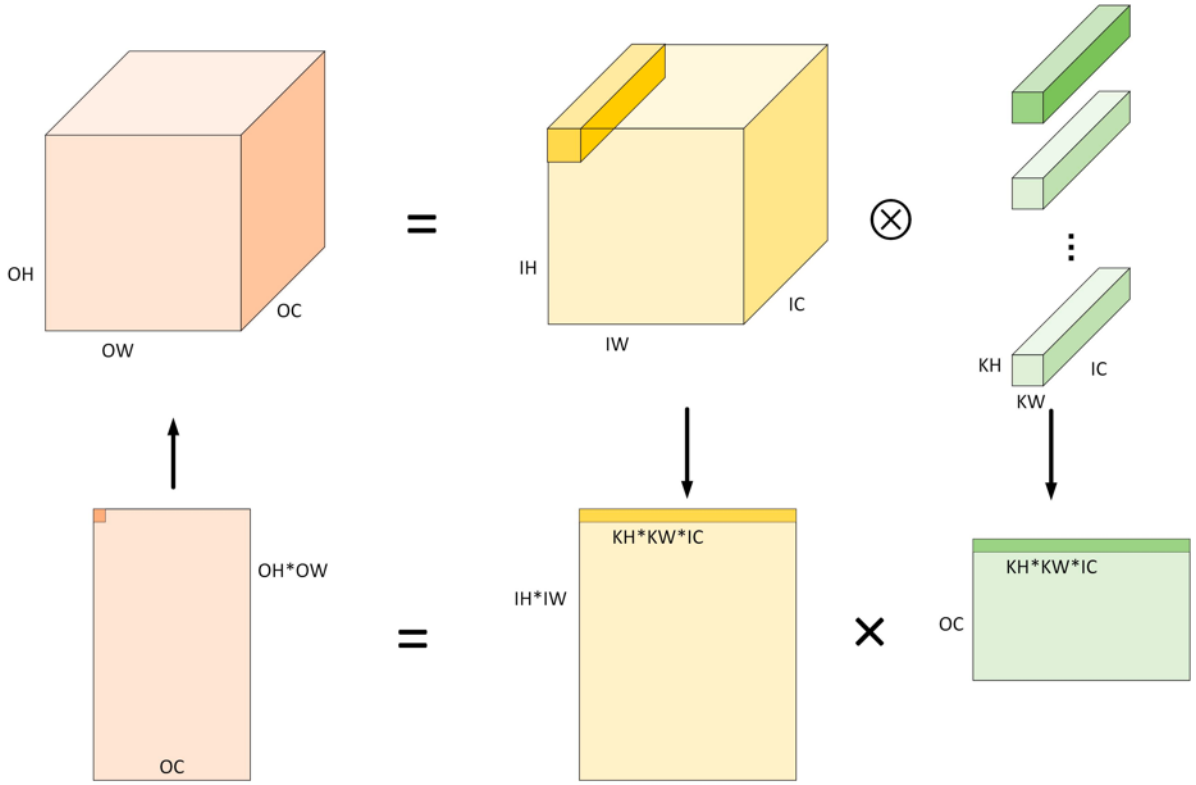


2. 脉动阵列

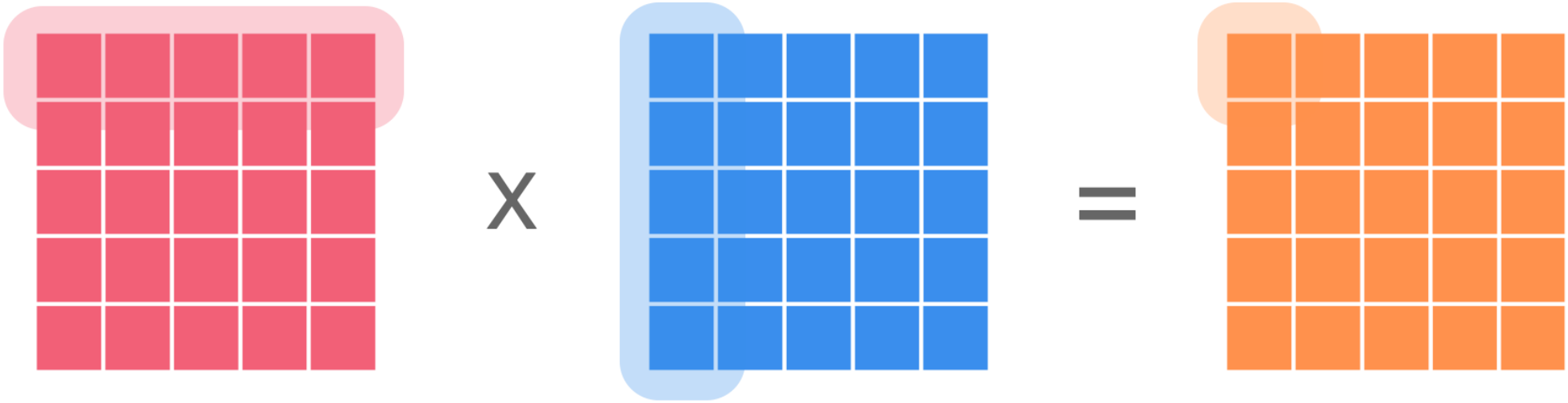
Systolic array

Img2col 算法过程

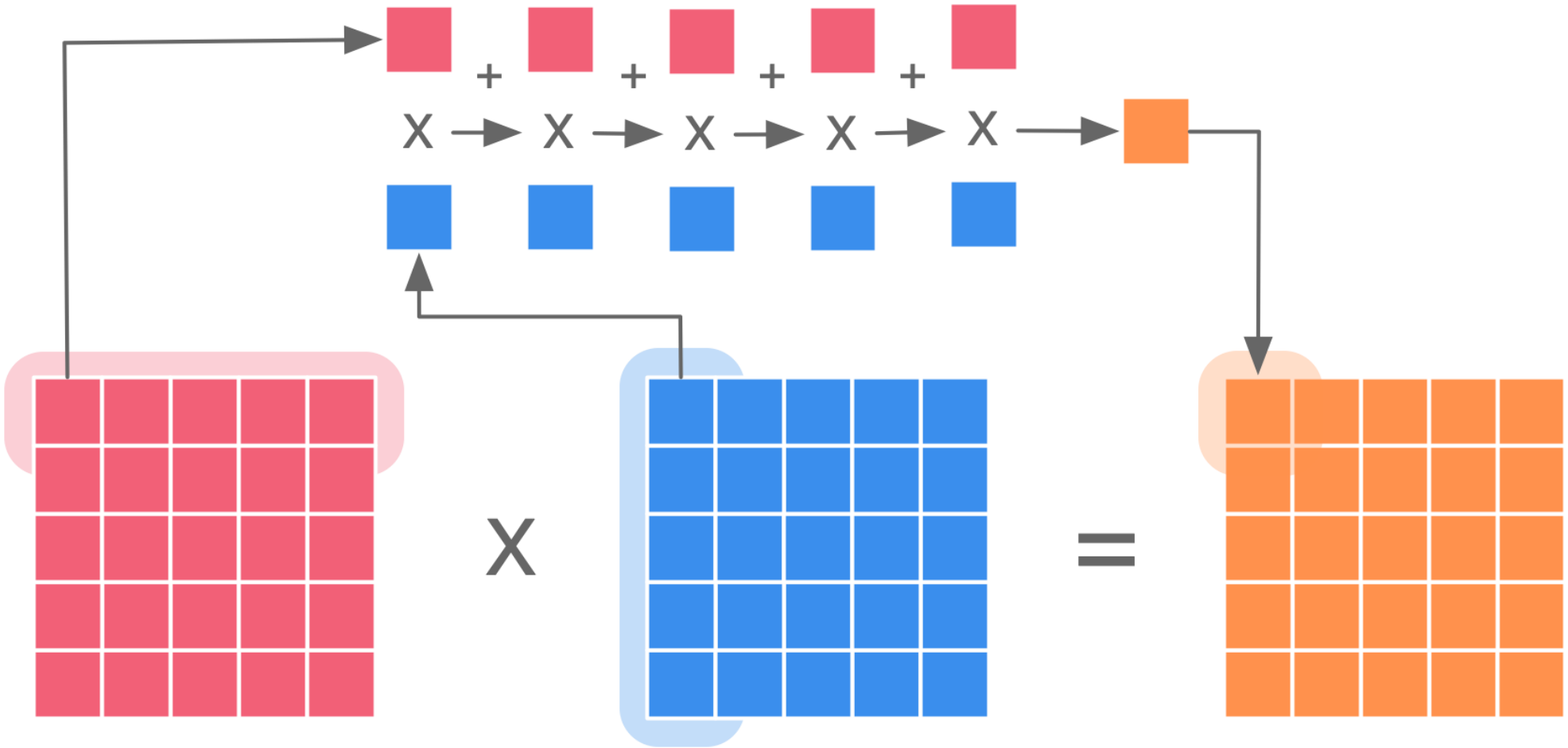
- 通过数据重排，完成 Im2col 的操作之后会得到一个输入矩阵，卷积的 Weights 也可以转换为一个矩阵，卷积的计算就可以转换为两个矩阵相乘的求解，得到最终的卷积计算结果。



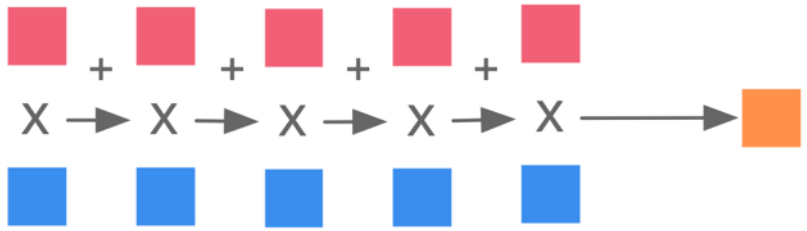
矩阵乘 MM 计算



矩阵乘 MM 计算



计算强度与矩阵乘的关系



For an NxN matrix:

- N row elements multiply with N column elements
- N additions create the final result
- This is done N^2 times, once for each result element

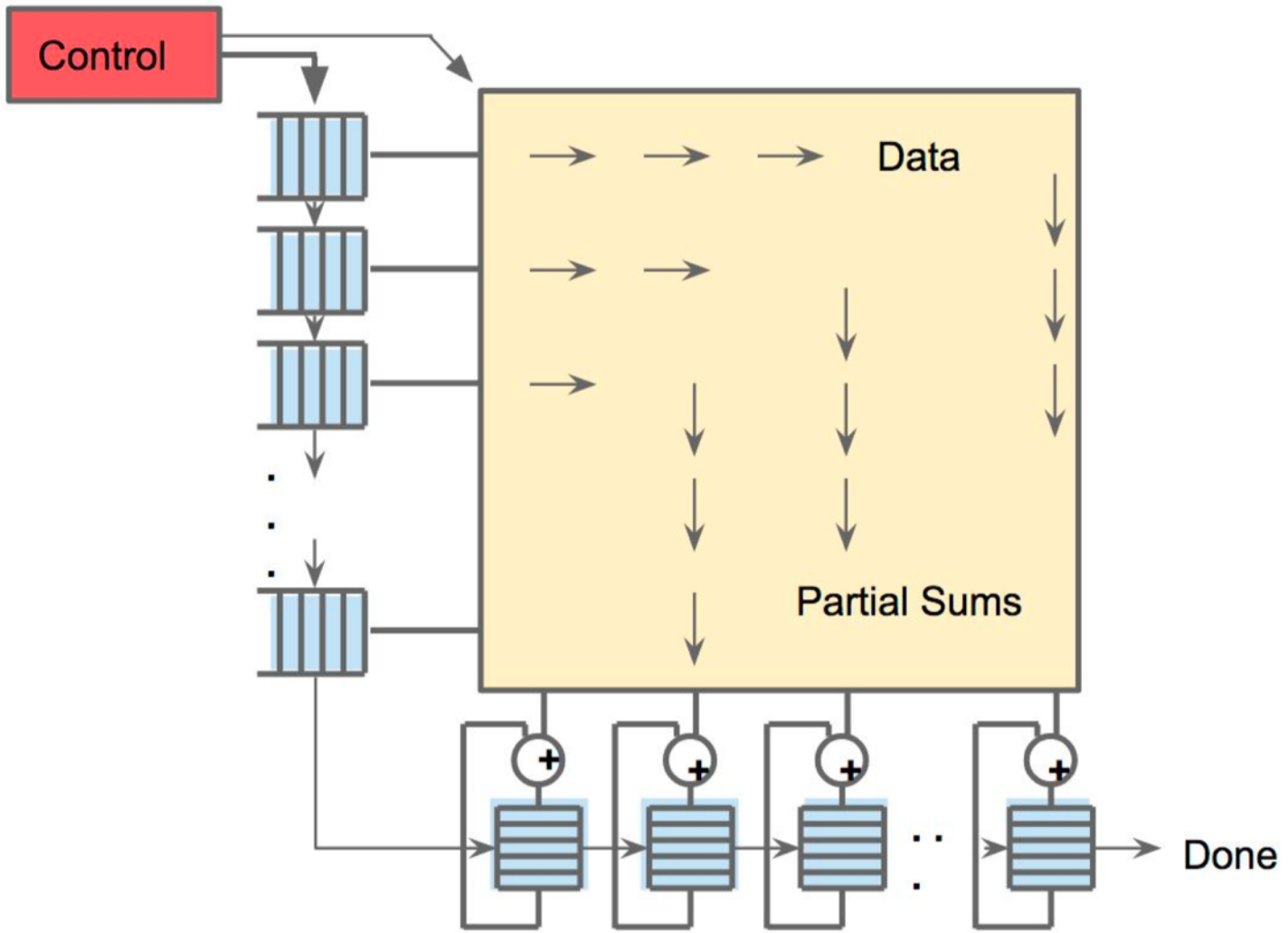
Arithmetic complexity is therefore: $N^2 \times (2N) = 2N^3 = O(N^3)$

Number of data loads: $O(N)$

Arithmetic intensity scales as: $N^3 / N^2 = O(N)$

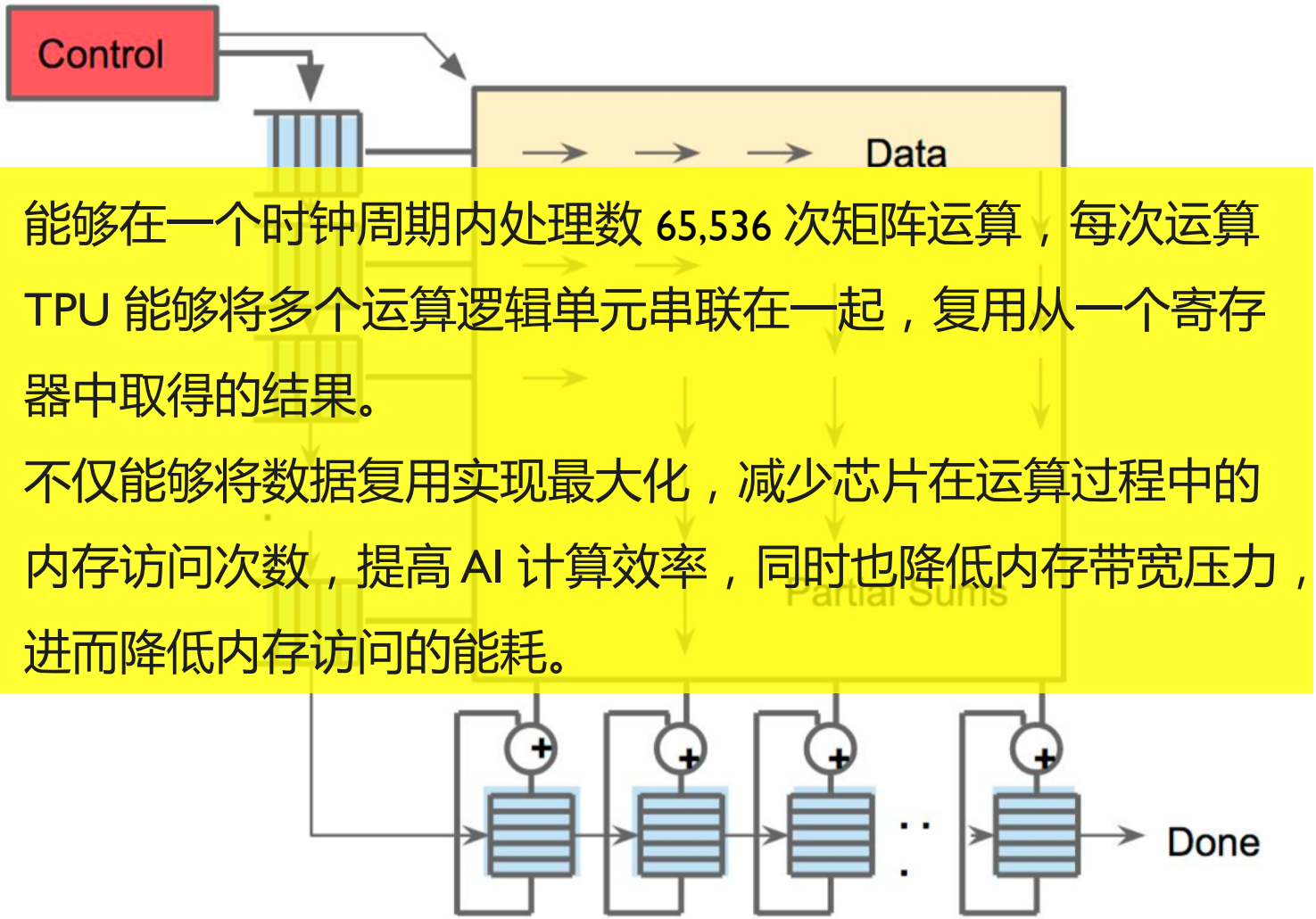
脉动阵列

- 数据一波一波根据 FIFO 队列从左 流入 MM Unite 计算，流出到下面的寄存器中存储，看起来像心脏脉动血流一样。
- TPU 中计算阵列是个二维的并行流水线乘法器，数据一波波脉动流走。



TPU 脉动阵列

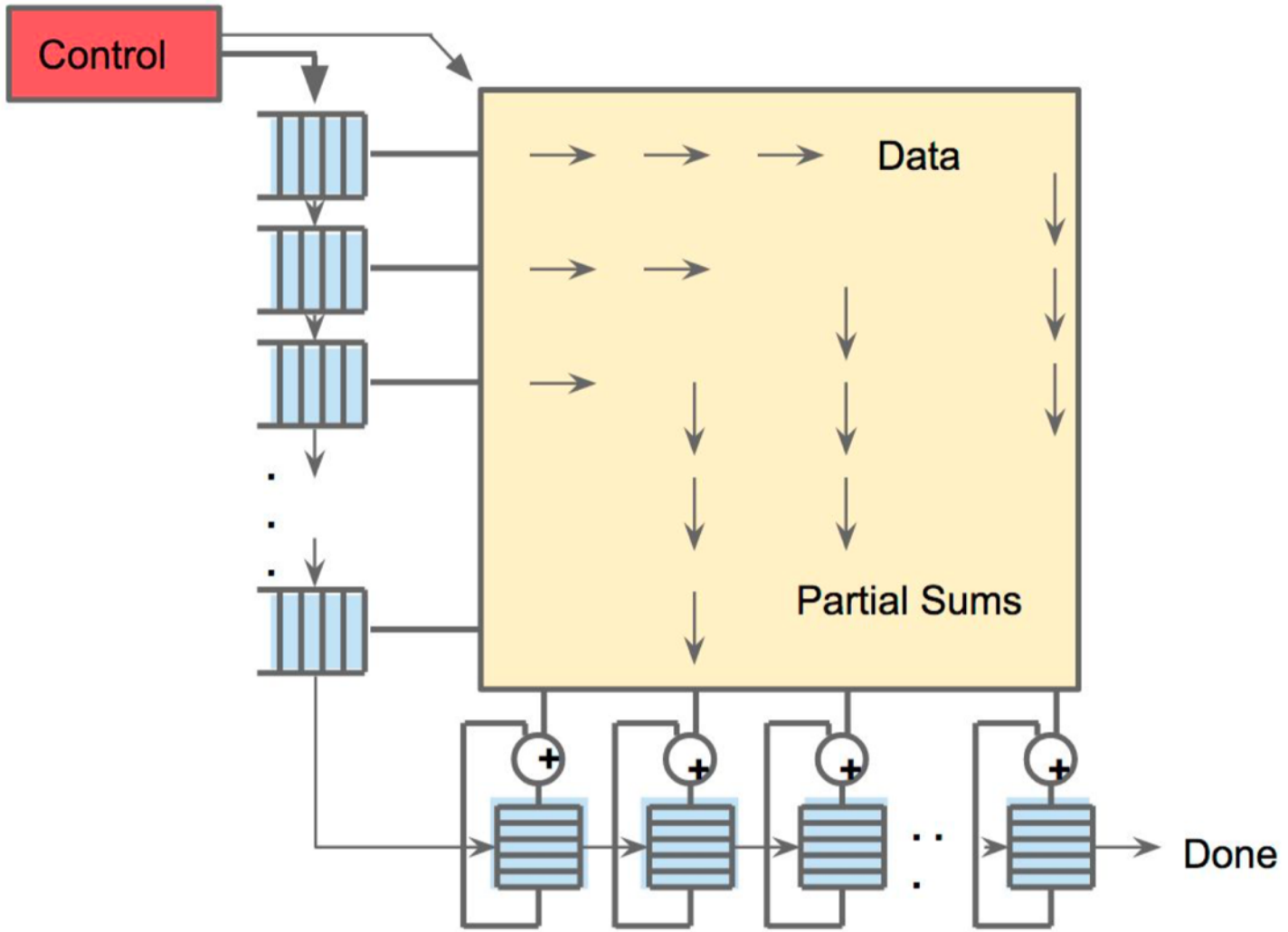
- 数据一波一波根据 FIFO 队列从左 流入 MM Unite 计算，流出到下面的寄存器中存储，看起来像心脏脉动血流一样。
- TPU 中计算阵列是个二维的并行流水线乘法器，数据一波波脉动流走。



- 能够在一个时钟周期内处理数 65,536 次矩阵运算，每次运算 TPU 能够将多个运算逻辑单元串联在一起，复用从一个寄存器中取得的结果。
- 不仅能够将数据复用实现最大化，减少芯片在运算过程中的内存访问次数，提高 AI 计算效率，同时也降低内存带宽压力，进而降低内存访问的能耗。

TPU 脉动阵列

- 脉动阵列：权重数据从上侧 weight 预先加载，输入数据 input 从左侧进入，输出数据从下侧输出；
- 给定的256-element乘累加运算通过矩阵作为对角波前（diagonal wavefront）移动；

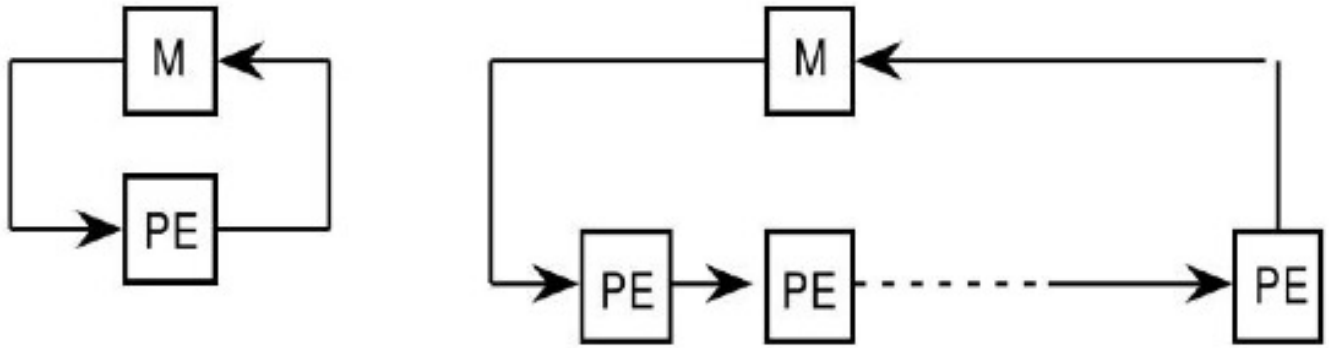


脉动阵列原理

Slides from Shaaban

Systolic Architectures

- Replace single processor with an array of regular processing elements
- Orchestrate data flow for high throughput with less memory access

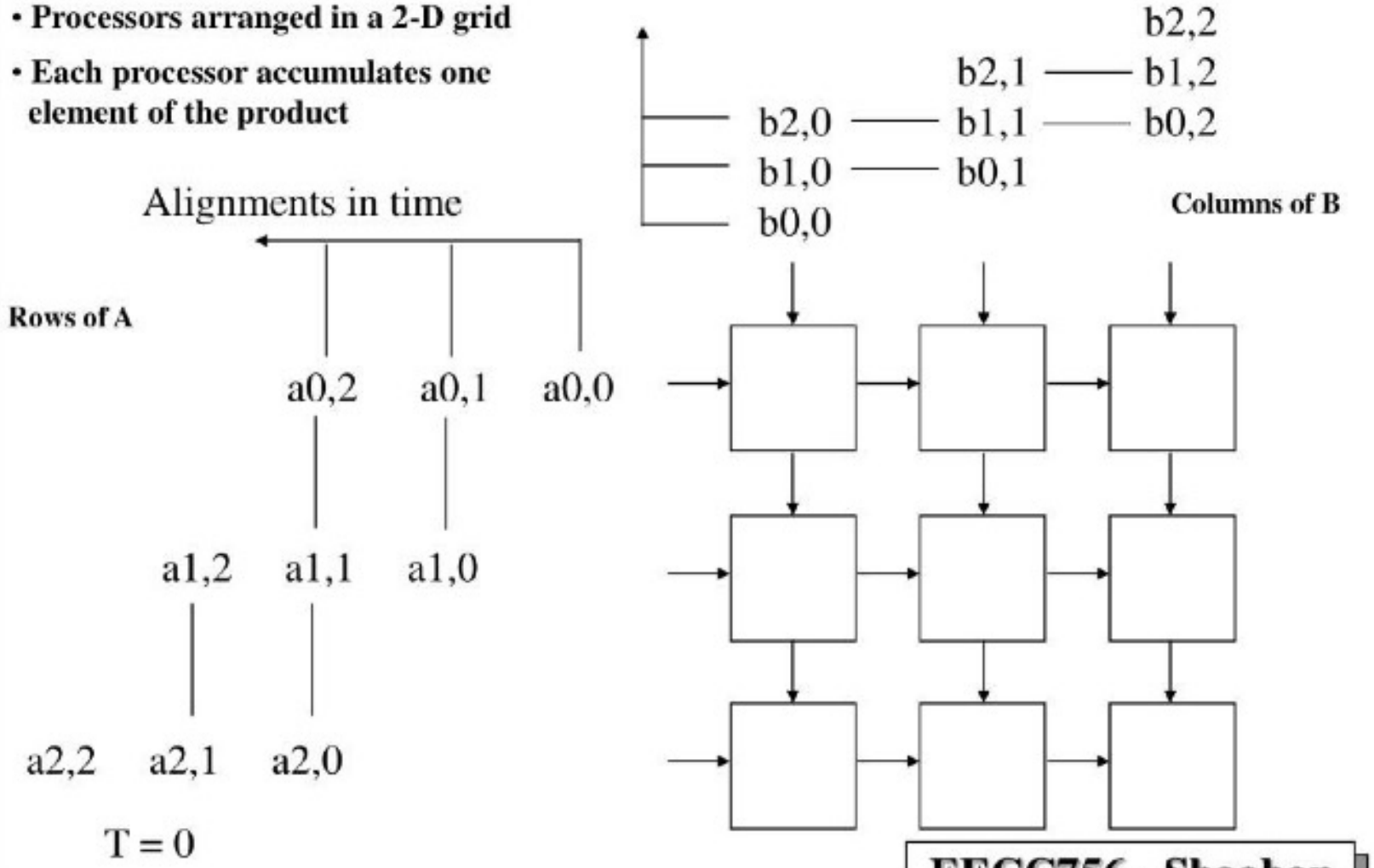


- Different from pipelining
  - Nonlinear array structure, multidirection data flow, each PE may have (small) local instruction and data memory
- Different from SIMD: each PE may do something different
- Initial motivation: VLSI enables inexpensive special-purpose chips
- Represent algorithms directly by chips connected in regular pattern

脉动阵列原理

Systolic Array Example:
3x3 Systolic Array Matrix Multiplication

- Processors arranged in a 2-D grid
- Each processor accumulates one element of the product

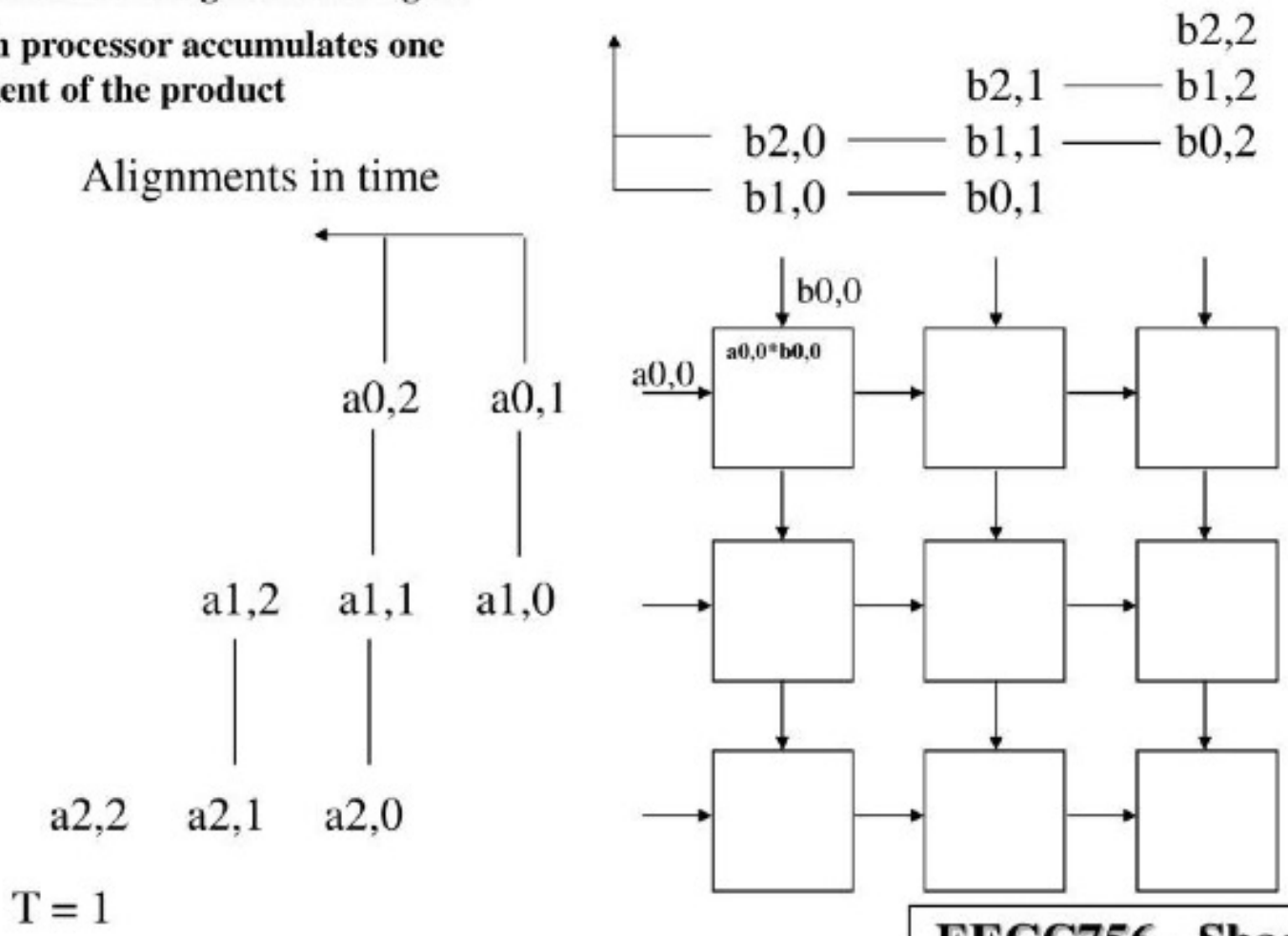


Example source: http://www.cs.hmc.edu/courses/2001/spring/cs156/

脉动阵列原理

Systolic Array Example:
3x3 Systolic Array Matrix Multiplication

- Processors arranged in a 2-D grid
- Each processor accumulates one element of the product

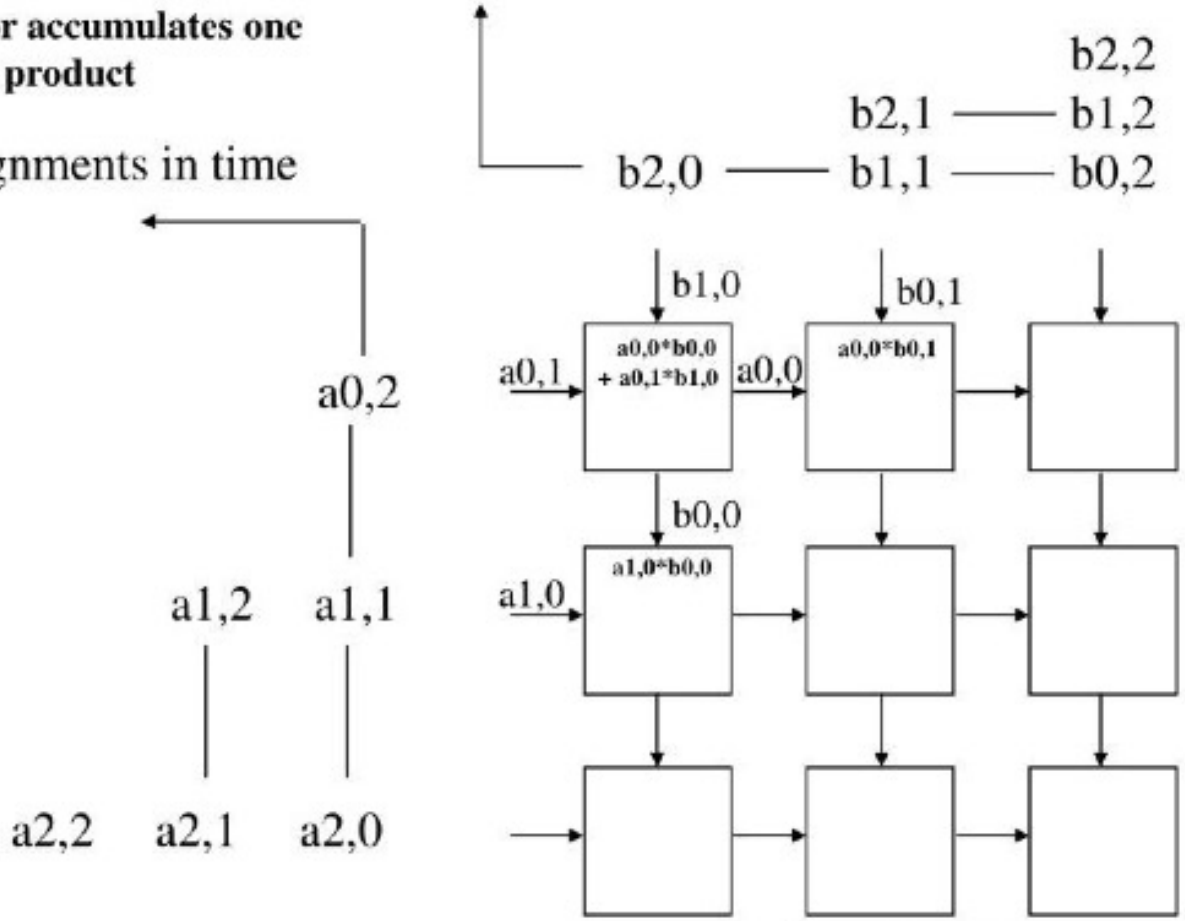


Example source: http://www.cs.hmc.edu/courses/2001/spring/cs156/

脉动阵列原理

Systolic Array Example:
3x3 Systolic Array Matrix Multiplication

- Processors arranged in a 2-D grid
- Each processor accumulates one element of the product

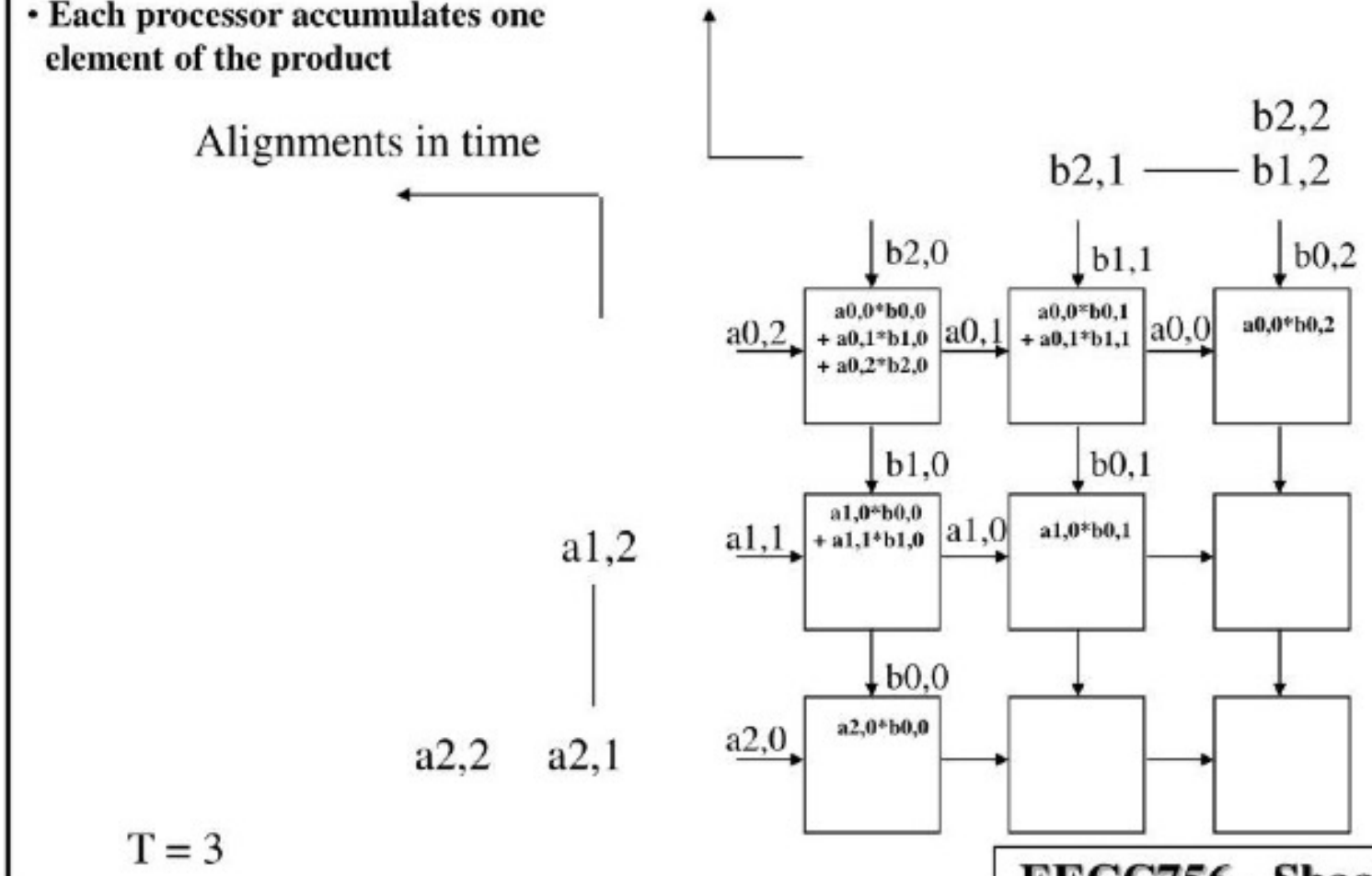


Example source: http://www.cs.hmc.edu/courses/2001/spring/cs156/

脉动阵列原理

Systolic Array Example:
3x3 Systolic Array Matrix Multiplication

- Processors arranged in a 2-D grid
- Each processor accumulates one element of the product

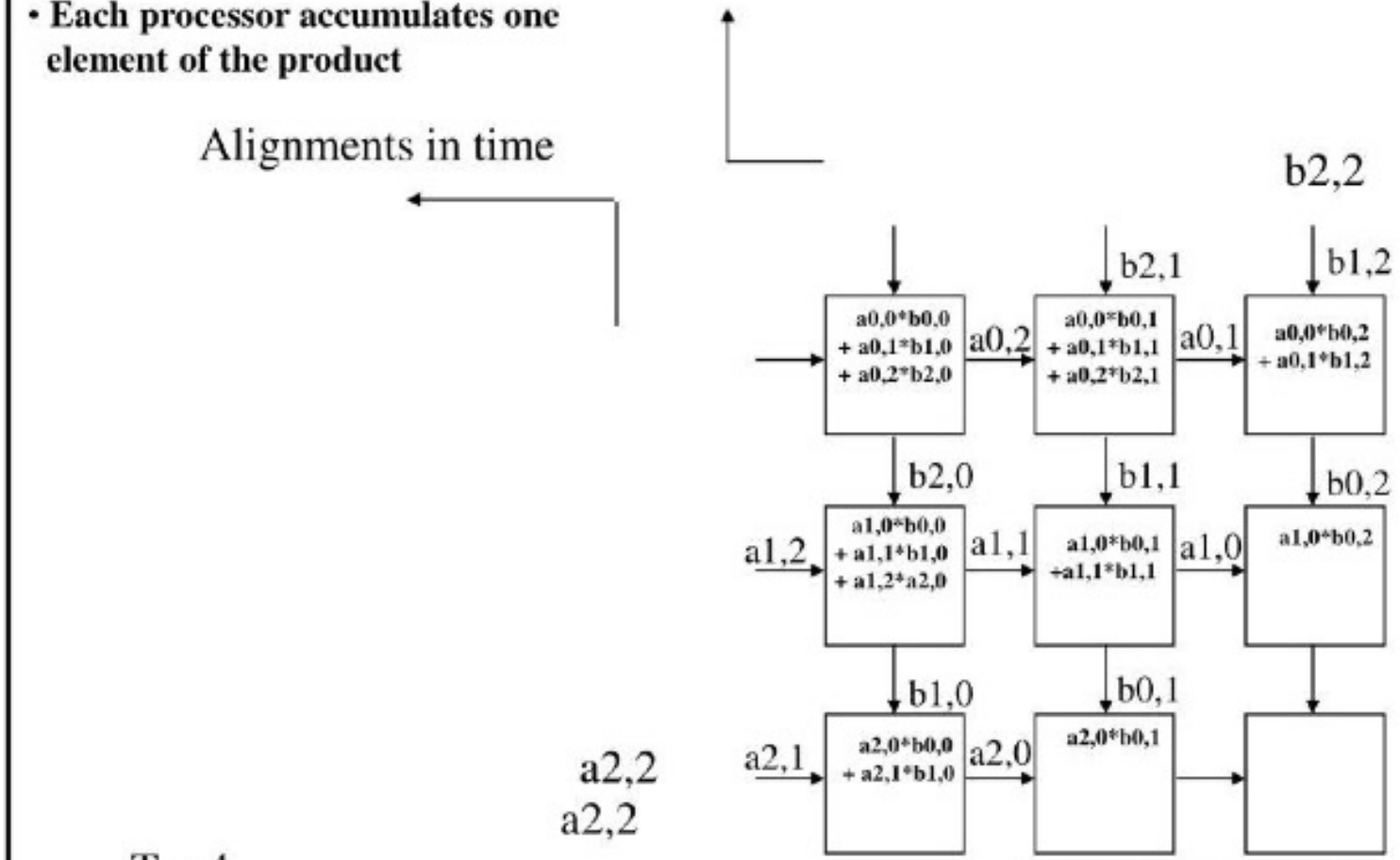


Example source: http://www.cs.hmc.edu/courses/2001/spring/cs156/

脉动阵列原理

Systolic Array Example:
3x3 Systolic Array Matrix Multiplication

- Processors arranged in a 2-D grid
- Each processor accumulates one element of the product



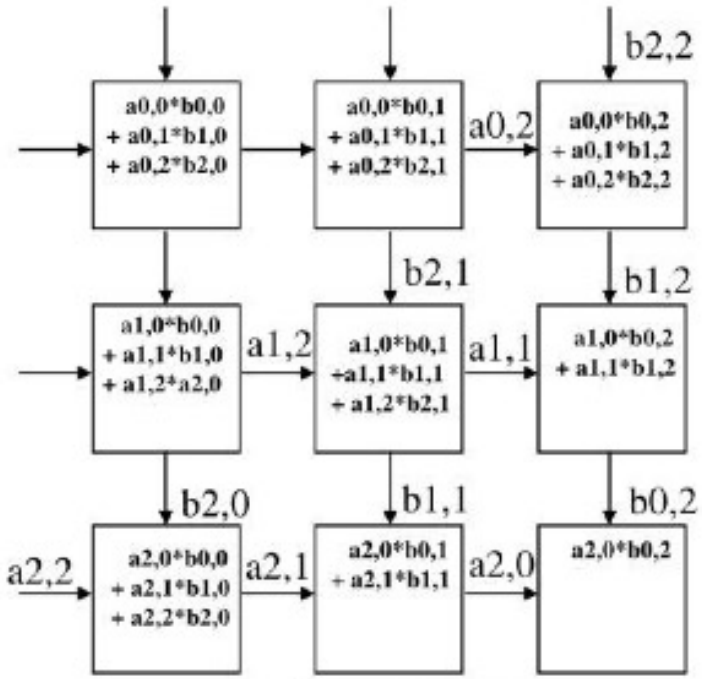
Example source: http://www.cs.hmc.edu/courses/2001/spring/cs156/

脉动阵列原理

Systolic Array Example:
3x3 Systolic Array Matrix Multiplication

- Processors arranged in a 2-D grid
- Each processor accumulates one element of the product

Alignments in time



T = 5

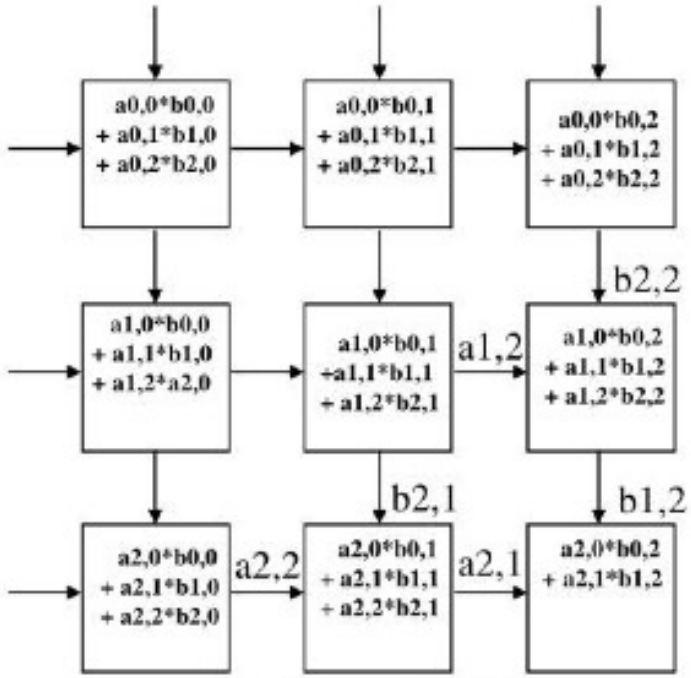
Example source: http://www.cs.hmc.edu/courses/2001/spring/cs156/

脉动阵列原理

Systolic Array Example:
3x3 Systolic Array Matrix Multiplication

- Processors arranged in a 2-D grid
- Each processor accumulates one element of the product

Alignments in time



T = 6

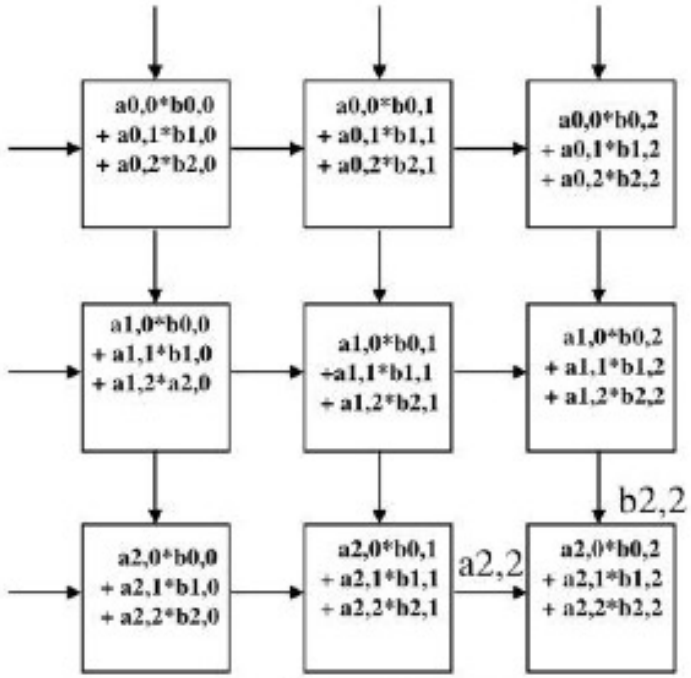
Example source: http://www.cs.hmc.edu/courses/2001/spring/cs156/

脉动阵列原理

Systolic Array Example:
3x3 Systolic Array Matrix Multiplication

- Processors arranged in a 2-D grid
- Each processor accumulates one element of the product

Alignments in time



Done

T = 7

Example source: http://www.cs.hmc.edu/courses/2001/spring/cs156/

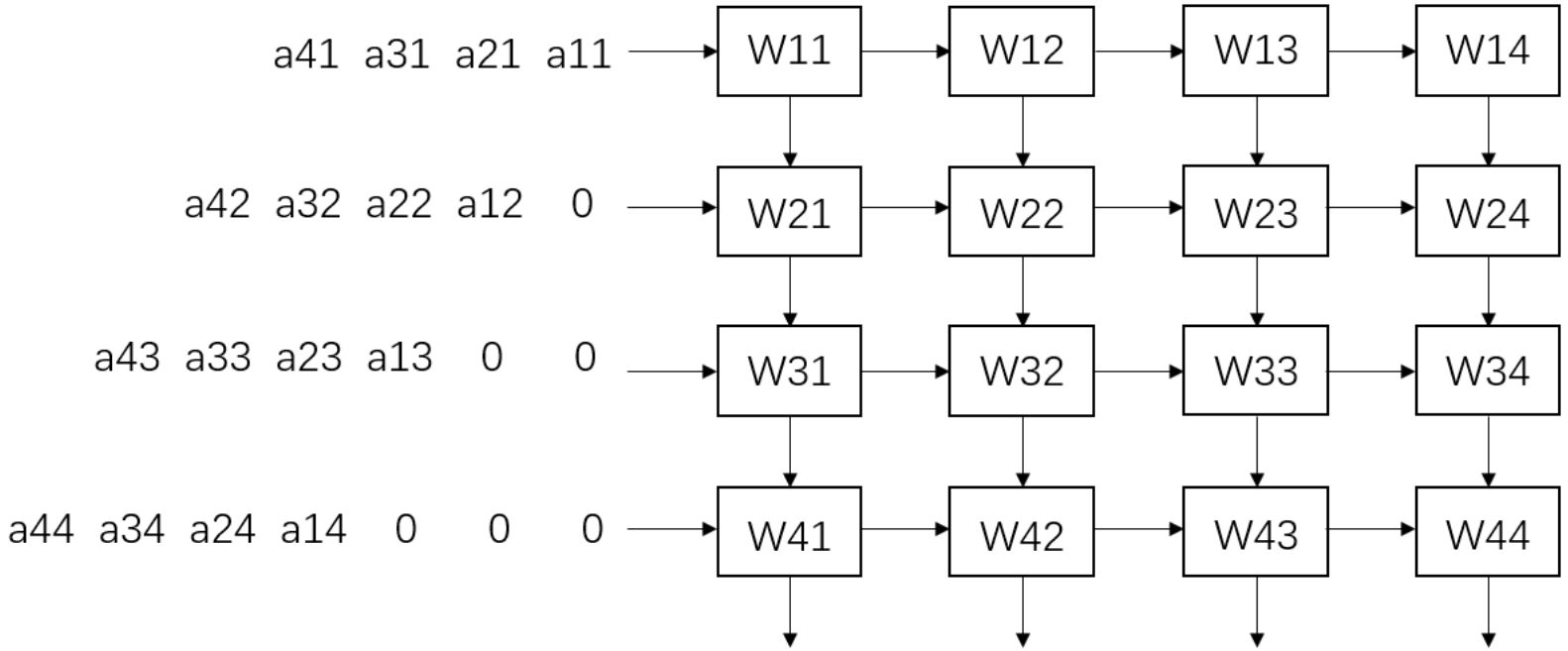
TPU 脉动阵列计算延迟

- 程序不感知 MMU 的脉动实现，从性能角度需要考虑其中的延迟，因为数据一级级的传导，意味着延迟的步步累加。

- CISC 指令使用四级流水线，用其他指令的执行与 MatrixMultiply 指令重叠，进而隐藏延迟。

TPU 脉动阵列：给定256乘积运算操作，以对角波的形式通过阵列

- 脉动阵列：权重数据从上侧 weight 预先加载，输入数据 input 从左侧进入，输出数据从下侧输出；
- 给定的256-element乘累加运算通过矩阵作为对角波前（diagonal wavefront）移动；



3. 竞品对比

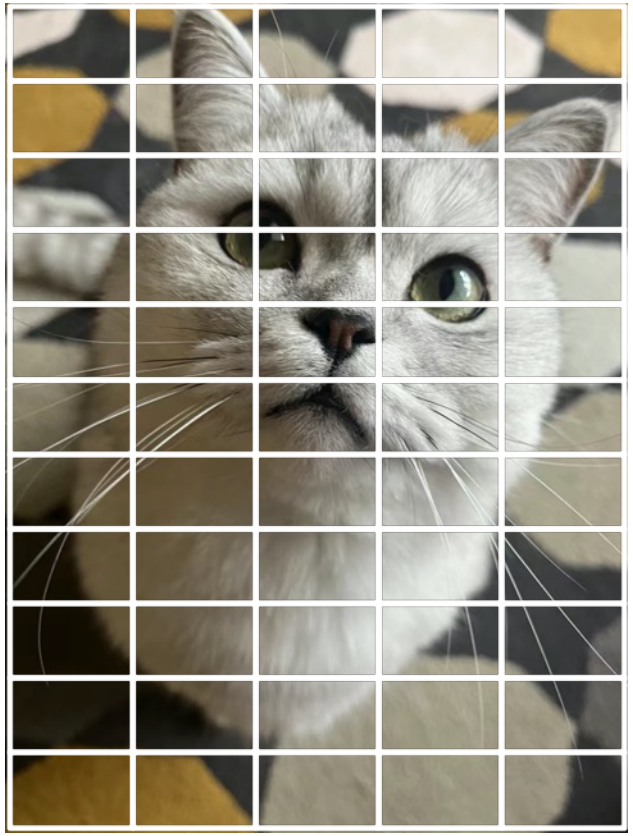
CPU、GPU、TPU1 服务器对比

- TPU1 采用的是 SIMD 模式，确定性执行的方式比 CPU 和 GPU 时变优化更符合神经网络的执行要求，减少缓存、乱序执行、多线程、多处理、预取等功能都有助于提高 TPU 计算吞吐，而不是降低延迟

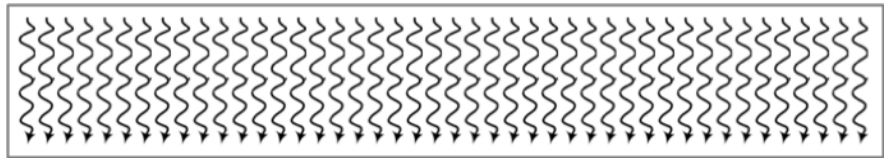
Model	*Die*										*Benchmarked Servers*				
	mm²	*nm*	*MHz*	*TDP*	*Measured*		*TOPS/s*		*GB/s*	*On-Chip Memory*	*Dies*	*DRAM Size*	*TDP*	*Measured*	
					Idle	*Busy*	8b	FP						*Idle*	*Busy*
Haswell E5-2699 v3	662	22	2300	145W	41W	145W	2.6	1.3	51	51 MiB	2	256 GiB	504W	159W	455W
NVIDIA K80 (2 dies/card)	561	28	560	150W	25W	98W	--	2.8	160	8 MiB	8	256 GiB (host) + 12 GiB x 8	1838W	357W	991W
TPU	NA*	28	700	75W	28W	40W	92	--	34	28 MiB	4	256 GiB (host) + 8 GiB x 4	861W	290W	384W

Table 2. Benchmarked servers use Haswell CPUs, K80 GPUs, and TPUs. Haswell has 18 cores, and the K80 has 13 SMX processors. Figure 10 has measured power. The low-power TPU allows for better rack-level density than the high-power GPU. The 8 GiB DRAM per TPU is Weight Memory. GPU Boost mode is not used (Sec. 8). SECDEC and no Boost mode reduce K80 bandwidth from 240 to 160. No Boost mode and single die vs. dual die performance reduces K80 peak TOPS from 8.7 to 2.8. (*The TPU die is ≤ half the Haswell die size.)

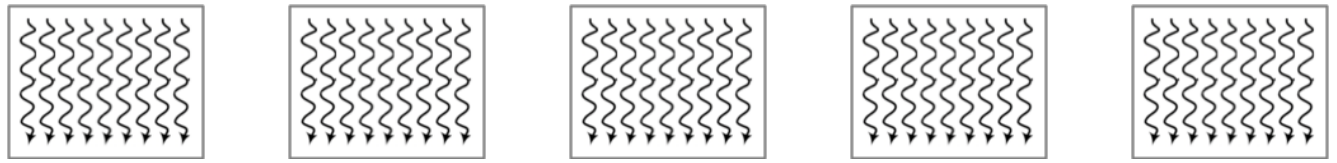
线程分层执行



网格 Grid 表示所有要执行的任务



网格 Grid 中包含了很多相同线程 Threads 数量的块 Blocks



块 Block 中的线程数独立执行

可以通过本地数据共享

同步交换数据

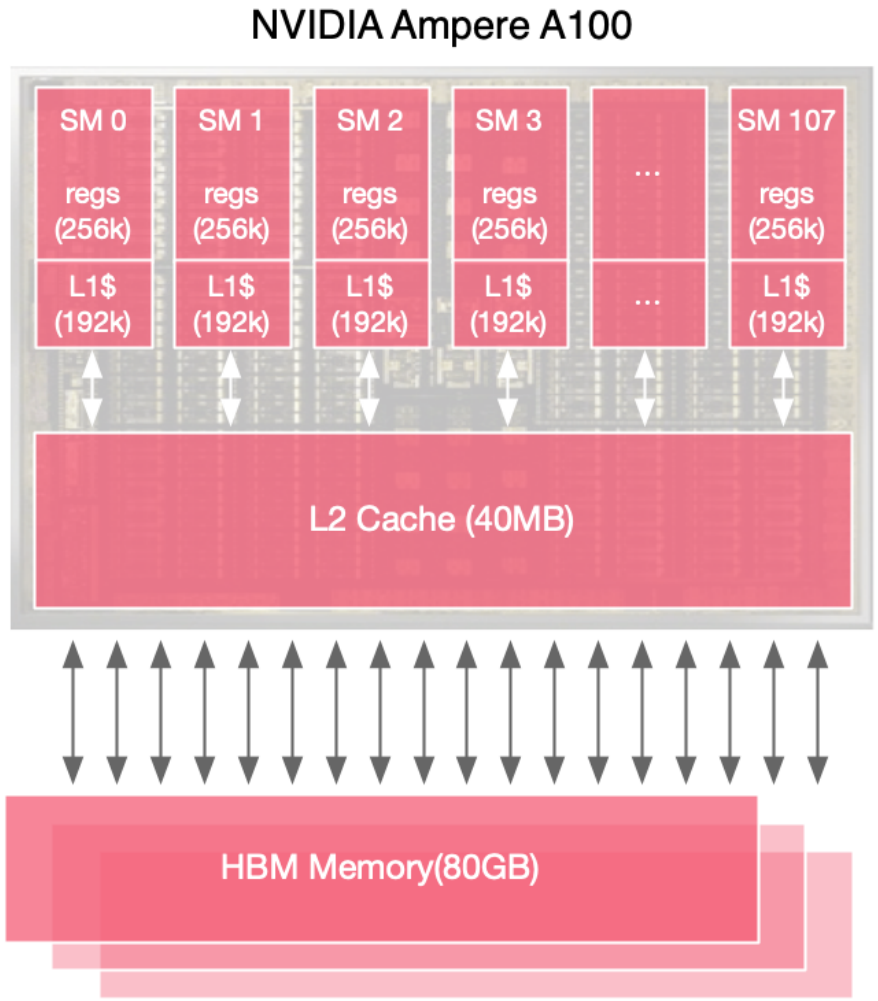
GPU 缓存机制

B/W Latency

13x 1x

3x 5x

1x 15x



Data Location	Latency (ns)	Threads Required
L1 Cache	27	32,738
L2 Cache	150	37,500
HBM	404	39,264
NVLink	700	13,125
PCIe	1470	2297

CPU、GPU、TPU1 服务器对比

- **计算性能**：每秒92万亿次计算，92Top/s=700MHz * 65535 *2，其中 65536 为每周期执行的乘法操作数，X2 为在 MMU 中同时还会有 65536 次加法。脉动阵列平稳运行时，每周期可以计算 256 个乘法结果；

Model	*Die*										*Benchmarked Servers*				
	mm²	*nm*	*MHz*	*TDP*	*Measured*		*TOPS/s*		*GB/s*	*On-Chip Memory*	*Dies*	*DRAM Size*	*TDP*	*Measured*	
					Idle	*Busy*	8b	FP						*Idle*	*Busy*
Haswell E5-2699 v3	662	22	2300	145W	41W	145W	2.6	1.3	51	51 MiB	2	256 GiB	504W	159W	455W
NVIDIA K80 (2 dies/card)	561	28	560	150W	25W	98W	--	2.8	160	8 MiB	8	256 GiB (host) + 12 GiB x 8	1838W	357W	991W
TPU	NA*	28	700	75W	28W	40W	92	--	34	28 MiB	4	256 GiB (host) + 8 GiB x 4	861W	290W	384W

Table 2. Benchmarked servers use Haswell CPUs, K80 GPUs, and TPUs. Haswell has 18 cores, and the K80 has 13 SMX processors. Figure 10 has measured power. The low-power TPU allows for better rack-level density than the high-power GPU. The 8 GiB DRAM per TPU is Weight Memory. GPU Boost mode is not used (Sec. 8). SECDEC and no Boost mode reduce K80 bandwidth from 240 to 160. No Boost mode and single die vs. dual die performance reduces K80 peak TOPS from 8.7 to 2.8. (*The TPU die is ≤ half the Haswell die size.)

CPU、GPU、TPU1 服务器对比

- **片上缓存**：TPU1 选择将片上缓存做的较大，从而节省片外访存消耗。

Model	Die										Benchmarked Servers				
	mm^2	nm	MHz	TDP	Measured		TOPS/s		GB/s	On-Chip Memory	Dies	DRAM Size	TDP	Measured	
					Idle	Busy	8b	FP						Idle	Busy
Haswell E5-2699 v3	662	22	2300	145W	41W	145W	2.6	1.3	51	51 MiB	2	256 GiB	504W	159W	455W
NVIDIA K80 (2 dies/card)	561	28	560	150W	25W	98W	--	2.8	160	8 MiB	8	256 GiB (host) + 12 GiB x 8	1838W	357W	991W
TPU	NA*	28	700	75W	28W	40W	92	--	34	28 MiB	4	256 GiB (host) + 8 GiB x 4	861W	290W	384W

Table 2. Benchmarked servers use Haswell CPUs, K80 GPUs, and TPUs. Haswell has 18 cores, and the K80 has 13 SMX processors. Figure 10 has measured power. The low-power TPU allows for better rack-level density than the high-power GPU. The 8 GiB DRAM per TPU is Weight Memory. GPU Boost mode is not used (Sec. 8). SECDEC and no Boost mode reduce K80 bandwidth from 240 to 160. No Boost mode and single die vs. dual die performance reduces K80 peak TOPS from 8.7 to 2.8. (*The TPU die is ≤ half the Haswell die size.)

CPU、GPU、TPU1 服务器对比

- **量化**：训练阶段使用是 FP32 的精度，而 TPU1 首推推理阶段使用 8bit integer。这是 TPU 比较具有前瞻性意义的里程碑技术点。

Model	*Die*										*Benchmarked Servers*				
	mm²	*nm*	*MHz*	*TDP*	*Measured*		*TOPS/s*		*GB/s*	*On-Chip Memory*	*Dies*	*DRAM Size*	*TDP*	*Measured*	
					Idle	*Busy*	8b	FP						*Idle*	*Busy*
Haswell E5-2699 v3	662	22	2300	145W	41W	145W	2.6	1.3	51	51 MiB	2	256 GiB	504W	159W	455W
NVIDIA K80 (2 dies/card)	561	28	560	150W	25W	98W	--	2.8	160	8 MiB	8	256 GiB (host) + 12 GiB x 8	1838W	357W	991W
TPU	NA*	28	700	75W	28W	40W	92	--	34	28 MiB	4	256 GiB (host) + 8 GiB x 4	861W	290W	384W

Table 2. Benchmarked servers use Haswell CPUs, K80 GPUs, and TPUs. Haswell has 18 cores, and the K80 has 13 SMX processors. Figure 10 has measured power. The low-power TPU allows for better rack-level density than the high-power GPU. The 8 GiB DRAM per TPU is Weight Memory. GPU Boost mode is not used (Sec. 8). SECDEC and no Boost mode reduce K80 bandwidth from 240 to 160. No Boost mode and single die vs. dual die performance reduces K80 peak TOPS from 8.7 to 2.8. (*The TPU die is ≤ half the Haswell die size.)

CPU、GPU、TPU1 服务器对比

- 深度学习推理计算性能：Batch Size 越大，单次迭代的计算量越大；TPU 因为大的缓存可以存放更多的数据 Batch Size，IPS 最高，是 GPU 的近 10 倍。

Type	*Batch*	*99th% Response*	*Inf/s (IPS)*	*% Max IPS*
CPU	16	7.2 ms	5,482	42%
CPU	64	21.3 ms	13,194	100%
GPU	16	6.7 ms	13,461	37%
GPU	64	8.3 ms	36,465	100%
TPU	200	7.0 ms	225,000	80%
TPU	250	10.0 ms	280,000	100%

思考

- Google 在 2015 年就能部署TPU1 ASIC 张量处理器，这意味着从芯片立项开始应该往前推 2 年，AI 系统的思想理念非常超前，连英伟达当时候还没有出现 Tensor Core。
- 下一轮矩阵计算必须等到上一轮的激活函数计算完成后才能开始，因此会出现一个很明显的 delay slot，就是因为结果在被 UB 读取前会等待一个显式同步。

思考

- TPU1 做对了什么？有哪些超越时代的设计？
- TPU1 针对目前AI的发展，没有做什么？

Thank you

把AI系统带入每个开发者、每个家庭、每个组织，构建万物互联的智能世界

Bring AI System to every person, home and organization for a fully connected, intelligent world.

ZOMI

Course chenzomi12.github.io

GitHub github.com/chenzomi12/DeepLearningSystem