

中国科学院大学计算机学院专业选修课

GPU架构与编程

第一课：课程介绍、GPU简介

赵地
中科院计算所
2025年秋季学期

讲授内容

➤课程介绍

➤GPU简介

- 定义、发展史、总体构架
- 功能（一）：图形图像处理
- 功能（二）：通用计算、编程语言
- 模拟器、生产商

教学目的

- ✓本课程是面向全校（本科生、研究生）开设的一门计算机系统结构方向的**专业选修课程**。
- ✓本课程旨在通过介绍GPU架构与编程，特别是GPGPU，运行机理、关键技术及科技发展史，帮助学生建立、贯通GPU架构与编程所需的知识体系，培养学生的GPU架构与编程的能力、系统思维能力和正确的历史唯物主义科技观。

适用对象（前置课程）

- ✓课程从基本概念出发，循序渐进、由浅入深讲授GPU架构与编程的相关专业知识，对学生专业知识基础要求不强，适用于选择攻读计算机系统结构方向的研究生。
- ✓建议前置课程：并行计算、C/C++编程；数字电路、计算机组成原理。

课程内容

- ✓周次2：GPU简介、课程简介、课程大作业
- ✓周次3：GPU的架构（GPU架构综述、SIMT核、片上网络NoC、L2、访存系统、等）
- ✓周次4、5、6：CUDA编程（英伟达原版教案）
- ✓周次7、8：PTX编程
- ✓周次9：AMD ROCm简介、OpenCL编程

课程内容

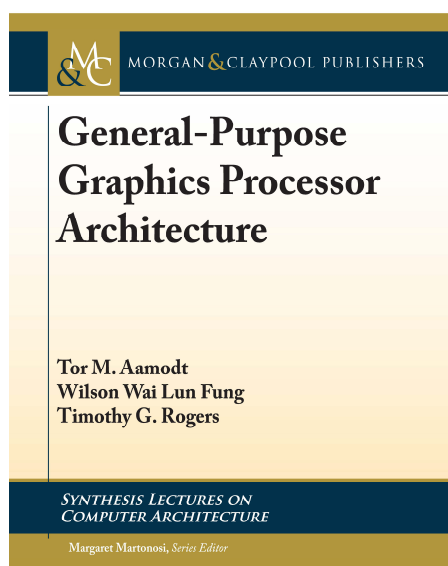
- ✓周次10：国产GPU编程、课程大作业（一）截止
- ✓周次11、12：OpenAI Triton编程
- ✓周次13、14：大模型（英伟达原版教案）
- ✓周次15：课程大作业（二）截止、课程大作业讲解、**颁奖典礼**、总复习
- ✓周次16：期末考试

上课时间

- ✓授课：2-16周；
- ✓每周三晚间（雁栖湖）：连续三节课（10-12 节），地点：教 1-002；
- ✓每周四晚间（玉泉路）：连续三节课（10-12节），地点：阶一5，同步直播 阶二5。

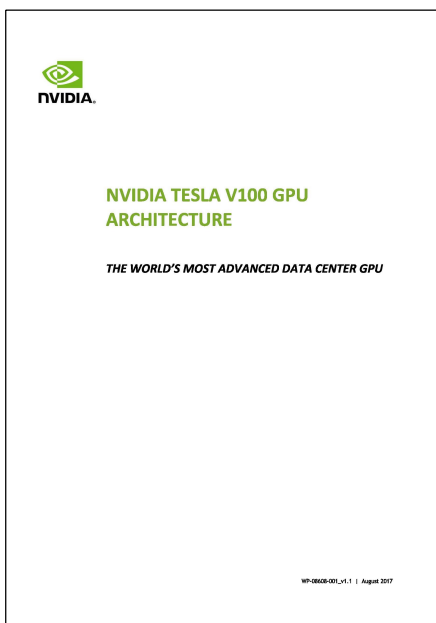
10	18:30 — 19:20
11	19:20 — 20:10
12	20:10 — 21:00

参考书：GPU的架构



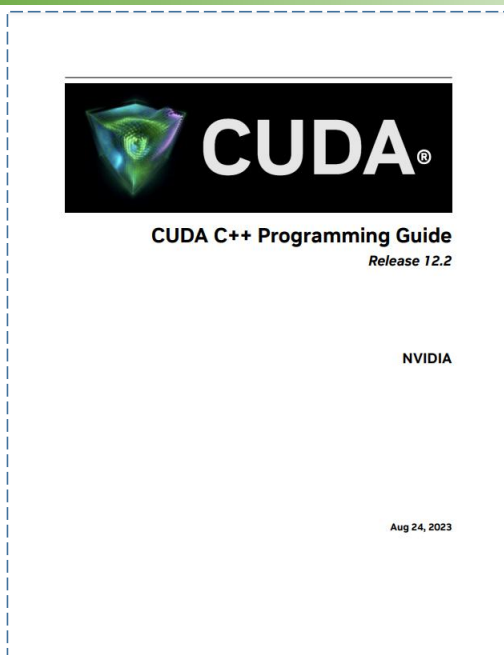
Tor M. Aamodt, Wilson Wai Lun Fung, Timothy G. Rogers, General-Purpose Graphics Processor Architectures, Springer Cham, 2018

参考书：GPU的架构



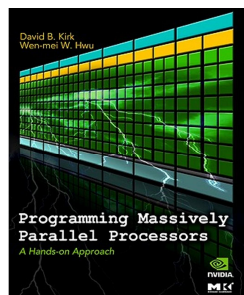
NVIDIA TESLA V100 GPU ARCHITECTURE, 2017

参考书：CUDA编程

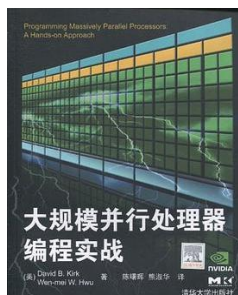


✓ https://docs.nvidia.com/cuda/pdf/CUDA_C_Programming_Guide.pdf

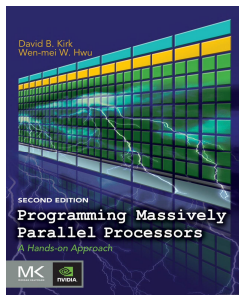
参考书：CUDA编程



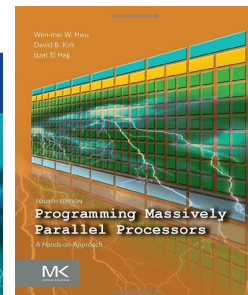
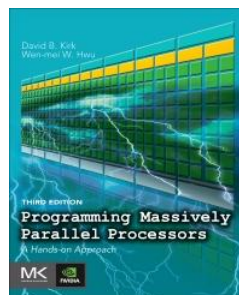
2010



2012



2017



2022

✓教材：Third Edition:

<https://www.sciencedirect.com/book/9780128119860/programming-massively-parallel-processors>

David B. Kirk, Wen-mei W. Hwu, Programming Massively Parallel Processors: A Hands-on Approach;

Contents

Preface.....	xv
Acknowledgements.....	xxi
CHAPTER 1 Introduction.....	1
1.1 Heterogeneous Parallel Computing.....	2
1.2 Architecture of a Modern GPU.....	6
1.3 Why More Speed or Parallelism?.....	8
1.4 Speeding Up Real Applications.....	10
1.5 Challenges in Parallel Programming.....	12
1.6 Parallel Programming Languages and Models.....	12
1.7 Overarching Goals.....	14
1.8 Organization of the Book.....	15
References.....	18
CHAPTER 2 Data Parallel Computing.....	19
2.1 Data Parallelism.....	20
2.2 CUDA C Program Structure.....	22
2.3 A Vector Addition Kernel.....	25
2.4 Device Global Memory and Data Transfer.....	27
2.5 Kernel Functions and Threading.....	32
2.6 Kernel Launch.....	37
2.7 Summary.....	38
Function Declarations.....	38
Kernel Launch.....	38
Built-in (Predefined) Variables.....	39
Run-time API.....	39
2.8 Exercises.....	39
References.....	41
CHAPTER 3 Scalable Parallel Execution.....	43
3.1 CUDA Thread Organization.....	43
3.2 Mapping Threads to Multidimensional Data.....	47
3.3 Image Blur: A More Complex Kernel.....	54
3.4 Synchronization and Transparent Scalability.....	58
3.5 Resource Assignment.....	60
3.6 Querying Device Properties.....	61
3.7 Thread Scheduling and Latency Tolerance.....	64
3.8 Summary.....	67
3.9 Exercises.....	67

vii

viii Contents

CHAPTER 4 Memory and Data Locality.....	71
4.1 Importance of Memory Access Efficiency.....	72
4.2 Matrix Multiplication.....	73
4.3 CUDA Memory Types.....	77
4.4 Tiling for Reduced Memory Traffic.....	84
4.5 A Tiled Matrix Multiplication Kernel.....	90
4.6 Boundary Checks.....	94
4.7 Memory as a Limiting Factor to Parallelism.....	97
4.8 Summary.....	99
4.9 Exercises.....	100
CHAPTER 5 Performance Considerations.....	103
5.1 Global Memory Bandwidth.....	104
5.2 More on Memory Parallelism.....	112
5.3 Warps and SIMD Hardware.....	117
5.4 Dynamic Partitioning of Resources.....	125
5.5 Thread Granularity.....	127
5.6 Summary.....	128
5.7 Exercises.....	128
References.....	130
CHAPTER 6 Numerical Considerations.....	131
6.1 Floating-Point Data Representation.....	132
Normalized Representation of M	132
Excess Encoding of E	133
6.2 Representable Numbers.....	134
6.3 Special Bit Patterns and Precision in IEEE Format.....	138
6.4 Arithmetic Accuracy and Rounding.....	139
6.5 Algorithm Considerations.....	140
6.6 Linear Solvers and Numerical Stability.....	142
6.7 Summary.....	146
6.8 Exercises.....	147
References.....	147
CHAPTER 7 Parallel Patterns: Convolution.....	149
7.1 Background.....	150
7.2 1D Parallel Convolution—A Basic Algorithm.....	153
7.3 Constant Memory and Caching.....	156
7.4 Tiled 1D Convolution with Halo Cells.....	160
7.5 A Simpler Tiled 1D Convolution—General Caching.....	165
7.6 Tiled 2D Convolution with Halo Cells.....	166

David B. Kirk, Wen-mei W. Hwu, Programming Massively Parallel Processors: A Hands-on Approach;

Contents ix

7.7 Summary	172
7.8 Exercises	173
CHAPTER 8 Parallel Patterns: Prefix Sum.....	175
8.1 Background	176
8.2 A Simple Parallel Scan	177
8.3 Speed and Work Efficiency	181
8.4 A More Work-Efficient Parallel Scan	183
8.5 An Even More Work-Efficient Parallel Scan	187
8.6 Hierarchical Parallel Scan for Arbitrary-Length Inputs	189
8.7 Single-Pass Scan for Memory Access Efficiency	192
8.8 Summary	195
8.9 Exercises	195
References	196
CHAPTER 9 Parallel Patterns—Parallel Histogram Computation ..	199
9.1 Background	200
9.2 Use of Atomic Operations	202
9.3 Block versus Interleaved Partitioning	206
9.4 Latency versus Throughput of Atomic Operations	207
9.5 Atomic Operation in Cache Memory	210
9.6 Privatization	210
9.7 Aggregation	211
9.8 Summary	213
9.9 Exercises	213
Reference	214
CHAPTER 10 Parallel Patterns: Sparse Matrix Computation	215
10.1 Background	216
10.2 Parallel SpMV Using CSR	219
10.3 Padding and Transposition	221
10.4 Using a Hybrid Approach to Regulate Padding	224
10.5 Sorting and Partitioning for Regularization	227
10.6 Summary	229
10.7 Exercises	229
References	230
CHAPTER 11 Parallel Patterns: Merge Sort	231
11.1 Background	231
11.2 A Sequential Merge Algorithm	233
11.3 A Parallelization Approach	234
11.4 Co-Rank Function Implementation	236

David B. Kirk, Wen-mei W. Hwu, Programming Massively Parallel Processors: A Hands-on Approach;

x Contents

11.5 A Basic Parallel Merge Kernel	241
11.6 A Tiled Merge Kernel	242
11.7 A Circular-Buffer Merge Kernel	249
11.8 Summary	256
11.9 Exercises	256
Reference	256
CHAPTER 12 Parallel Patterns: Graph Search.....	257
12.1 Background	258
12.2 Breadth-First Search	260
12.3 A Sequential BFS Function	262
12.4 A Parallel BFS Function	265
12.5 Optimizations	270
Memory Bandwidth	270
Hierarchical Queues	271
Kernel Launch Overhead	272
Load Balance	273
12.6 Summary	273
12.7 Exercises	273
References	274
CHAPTER 13 CUDA Dynamic Parallelism.....	275
13.1 Background	276
13.2 Dynamic Parallelism Overview	278
13.3 A Simple Example	279
13.4 Memory Data Visibility	281
Global Memory	281
Zero-Copy Memory	282
Constant Memory	282
Local Memory	282
Shared Memory	283
Texture Memory	283
13.5 Configurations and Memory Management	283
Launch Environment Configuration	283
Memory Allocation and Lifetime	283
Nesting Depth	284
Pending Launch Pool Configuration	284
Errors and Launch Failures	284
13.6 Synchronization, Streams, and Events	285
Synchronization	285
Synchronization Depth	285
Streams	286
Events	287

Contents xi

13.7 A More Complex Example	287
Linear Bezier Curves	288
Quadratic Bezier Curves	288
Bezier Curve Calculation (Without Dynamic Parallelism)	288
Bezier Curve Calculation (With Dynamic Parallelism)	290
Launch Pool Size	292
Streams	292
13.8 A Recursive Example	293
13.9 Summary	297
13.10 Exercises	299
References	301
A13.1 Code Appendix	301
CHAPTER 14 Application Case Study—non-Cartesian Magnetic Resonance Imaging.....	305
14.1 Background	306
14.2 Iterative Reconstruction	308
14.3 Computing \mathbf{f}^{HD}	310
Step 1: Determine the Kernel Parallelism Structure	312
Step 2: Getting Around the Memory Bandwidth Limitation	317
Step 3: Using Hardware Trigonometry Functions	323
Step 4: Experimental Performance Tuning	326
14.4 Final Evaluation	327
14.5 Exercises	328
References	329
CHAPTER 15 Application Case Study—Molecular Visualization and Analysis	331
15.1 Background	332
15.2 A Simple Kernel Implementation	333
15.3 Thread Granularity Adjustment	337
15.4 Memory Coalescing	338
15.5 Summary	342
15.6 Exercises	343
References	344
CHAPTER 16 Application Case Study—Machine Learning	345
16.1 Background	346
16.2 Convolutional Neural Networks	347
ConvNets: Basic Layers	348
ConvNets: Backpropagation	351
16.3 Convolutional Layer: A Basic CUDA Implementation of Forward Propagation	355

David B. Kirk, Wen-mei W. Hwu, Programming Massively Parallel Processors: A Hands-on Approach;

xii Contents

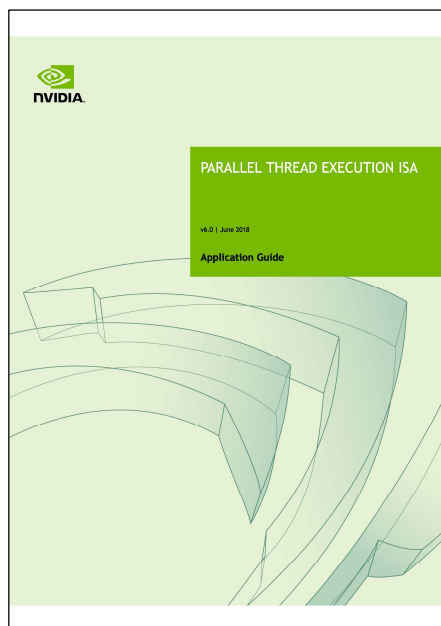
16.4 Reduction of Convolutional Layer to Matrix Multiplication	359
16.5 cuDNN Library	364
16.6 Exercises	366
References	367
CHAPTER 17 Parallel Programming and Computational Thinking	369
17.1 Goals of Parallel Computing	370
17.2 Problem Decomposition	371
17.3 Algorithm Selection	374
17.4 Computational Thinking	379
17.5 Single Program, Multiple Data, Shared Memory and Locality	380
17.6 Strategies for Computational Thinking	382
17.7 A Hypothetical Example: Sodium Map of the Brain	383
17.8 Summary	386
17.9 Exercises	386
References	386
CHAPTER 18 Programming a Heterogeneous Computing Cluster	387
18.1 Background	388
18.2 A Running Example	388
18.3 Message Passing Interface Basics	391
18.4 Message Passing Interface Point-to-Point Communication	393
18.5 Overlapping Computation and Communication	400
18.6 Message Passing Interface Collective Communication	408
18.7 CUDA-Aware Message Passing Interface	409
18.8 Summary	410
18.9 Exercises	410
Reference	411
CHAPTER 19 Parallel Programming with OpenACC	413
19.1 The OpenACC Execution Model	414
19.2 OpenACC Directive Format	416
19.3 OpenACC by Example	418
The OpenACC Kernels Directive	419
The OpenACC Parallel Directive	422
Comparison of Kernels and Parallel Directives	424
OpenACC Data Directives	425
OpenACC Loop Optimizations	430
OpenACC Routine Directive	432
Asynchronous Computation and Data	434

Contents xiii

19.4	Comparing OpenACC and CUDA.....	435
	Portability	435
	Performance.....	436
	Simplicity	436
19.5	Interoperability with CUDA and Libraries	437
	Calling CUDA or Libraries with OpenACC Arrays.....	437
	Using CUDA Pointers in OpenACC	438
	Calling CUDA Device Kernels from OpenACC	439
19.6	The Future of OpenACC.....	440
19.7	Exercises	441
CHAPTER 20 More on CUDA and Graphics Processing Unit Computing.....443		
20.1	Model of Host/Device Interaction.....	444
20.2	Kernel Execution Control	449
20.3	Memory Bandwidth and Compute Throughput.....	451
20.4	Programming Environment.....	453
20.5	Future Outlook	455
	References	456
CHAPTER 21 Conclusion and Outlook.....457		
21.1	Goals Revisited	457
21.2	Future Outlook	458
	Appendix A: An Introduction to OpenCL.....	461
	Appendix B: THRUST: a Productivity-oriented Library for CUDA.....	475
	Appendix C: CUDA Fortran.....	493
	Appendix D: An introduction to C++ AMP.....	515
	Index	535

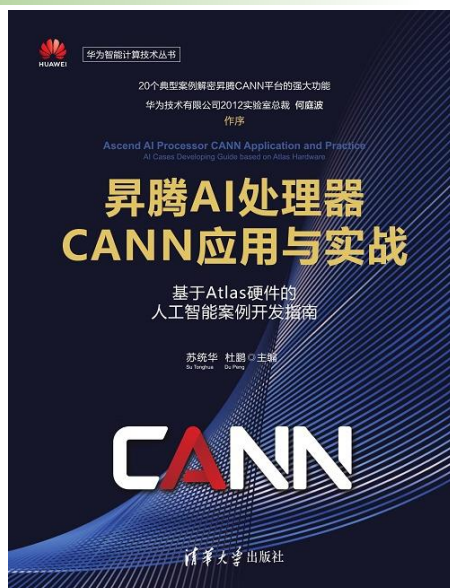
David B. Kirk, Wen-mei W. Hwu, Programming Massively Parallel Processors: A Hands-on Approach;

参考书：PTX编程



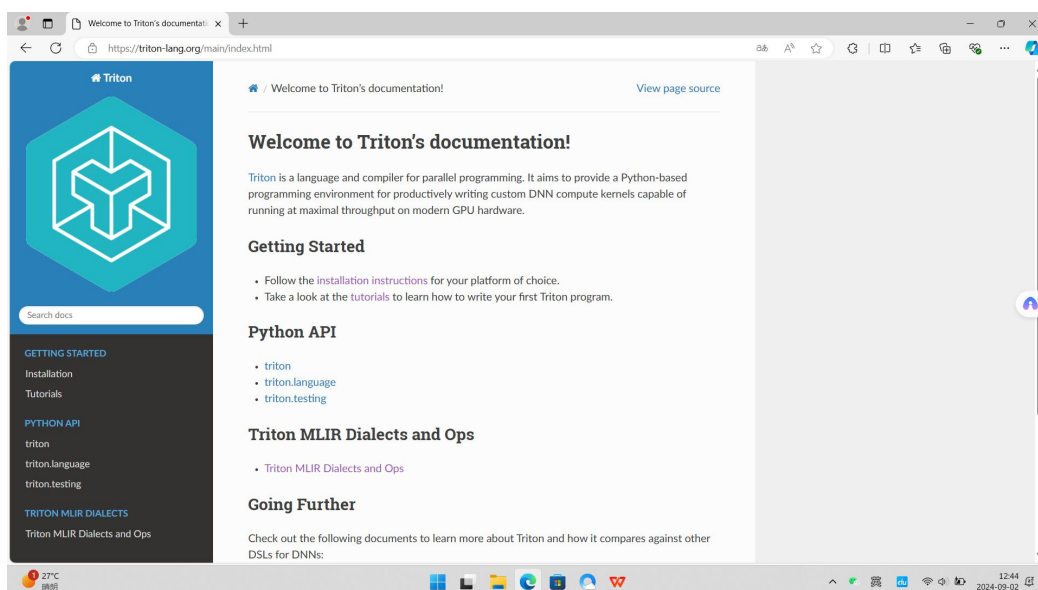
Parallel Thread Execution ISA Version 6.0, 2018

参考书：CANN编程



✓ <https://www.hiascend.com/zh/ascend-c>

参考书：Triton编程



✓ <https://triton-lang.org/main/index.html>

期末考试

- ✓时间：周次16；
- ✓考试内容：GPU架构与编程；
- ✓考试方式：线下、纸质试卷、开卷。

评分标准

- ✓课程大作业（一）：30%，后续发布加分题
- ✓课程大作业（二）：30%，后续发布加分题
- ✓期末考试（开卷）：30%
- ✓考勤：10%
- ✓满分：100%

组织形式

- ✓课程大作业（一）：统一评分
- ✓课程大作业（二）：**赛区、赛道**
- ✓期末考试（开卷）：统一评分
- ✓考勤：统一评分

组织形式

- ✓课程大作业（一）：国科大计算中心（八卡英伟达V100）
- ✓课程大作业（二）：各位同学选择一个赛道，每位同学只能选择一个赛道

	摩尔线程	沐曦
玉泉路校区	选择一个	
雁栖湖校区		

赛道一：摩尔线程

✓奖品池：

- ✓AIBOOK算力本，价值9999元，2台
- ✓摩尔线程MTT S80游戏显卡，价值1499元，4块
- ✓摩尔学院礼包（内含笔记本1本、摩尔学院徽章1枚、手机支架1个），6个
- ✓摩尔学院徽章，90个
- ✓每人价值100元的AutoDL算力券。



多元异构算力AI处理器
旗舰性能，顶级配置
原生AI系统，预装AI编程环境
本地大模型，多模态自然语言交互
千人千面AI智能体，多应用打通
轻薄简约，时尚高端



赛道二：沐曦

✓奖品池：

- ✓《沐曦异构并行计算软件栈——MXMACA C/C++程序设计入门教程》每人1册；
- ✓gitee算力券，每人50元；
- ✓“开源GPU创新生态赛”算力券，每人300元，地址：
<https://www.gitlink.org.cn/competitions/gitlinkGPU1>；
- ✓课程实验环境，免费。



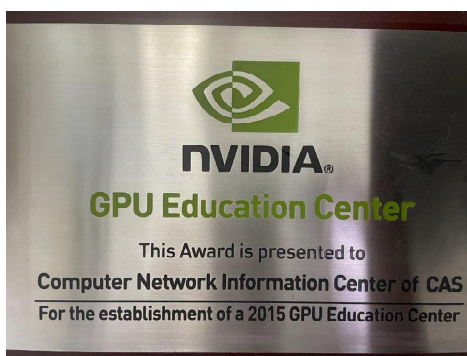
任课教师简介

- ✓赵地，中科院计算所副研究员；
- ✓2007年开始学习和研究GPU编程与架构；
- ✓在GPU领域发表论文多篇；



任课教师简介

- ✓2015 年度，赵地博士担任美国英伟达公司（NVIDIA）-中科院网络中心**GPU研究中心**（GPU Research Center）主任、美国英伟达公司（NVIDIA）-中科院网络中心**GPU教育中心**（GPU Education Center）主任。



任课教师简介

- ✓2016 年，中国科学院网络信息中心和美国英伟达公司（NVIDIA）联合主办2016大数据智能论坛（BDA2016），赵博士担任大会主席，并获得中国科学院国际合作局 2016年度国际会议资助；推动并成立了CNIC-英伟达“智慧医疗”联合实验室，担任实验室主任。



任课教师简介

- ✓2013 年、2014 年、2016 年、2017 年和 2019 年，前往美国硅谷参加英伟达公司举办的 GPU 技术大会（GPU Technology Conference），并与 NVIDIA 创始人兼首席执行官 Jensen Huang（黄仁勋）等顶尖 GPU 专家进行了深入的探讨；
- ✓2020 年、2021 年、2022 年、2023 年、2024 年，在线参加 GPU 技术大会（GPU Technology Conference）。

助教

✓线下

✓龚昊，计算机学院博士生

课程群

群聊: 2025秋《GPU架构
与编程》



该二维码7天内(9月30日前)有效，重新进入将更新



赵地
中国大陆

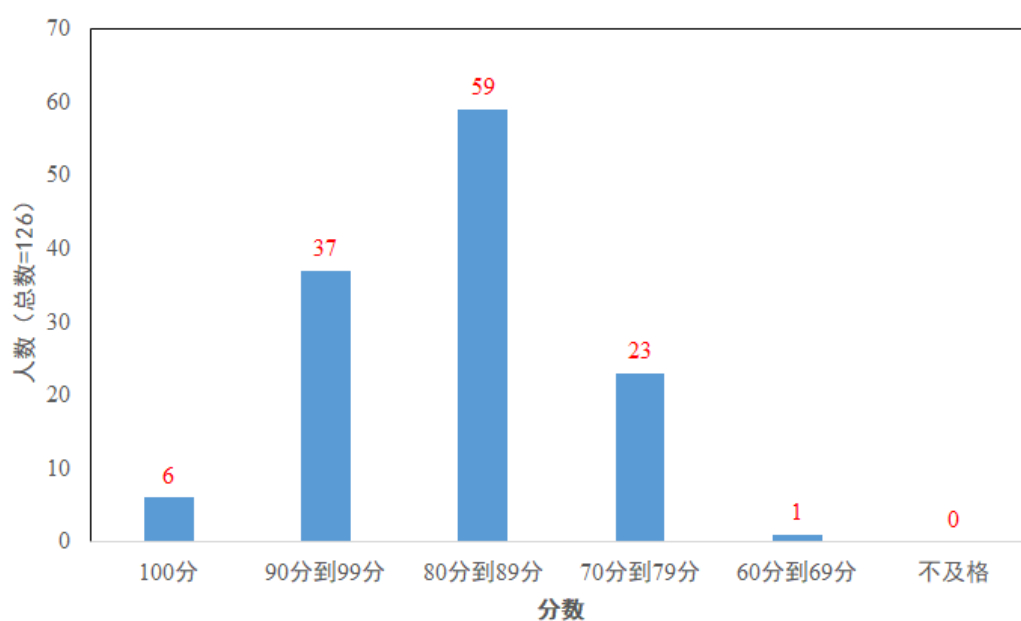


扫一扫上面的二维码图案，加我为朋友。

往届课程（23秋）：126人获得学分



往届课程（23秋）



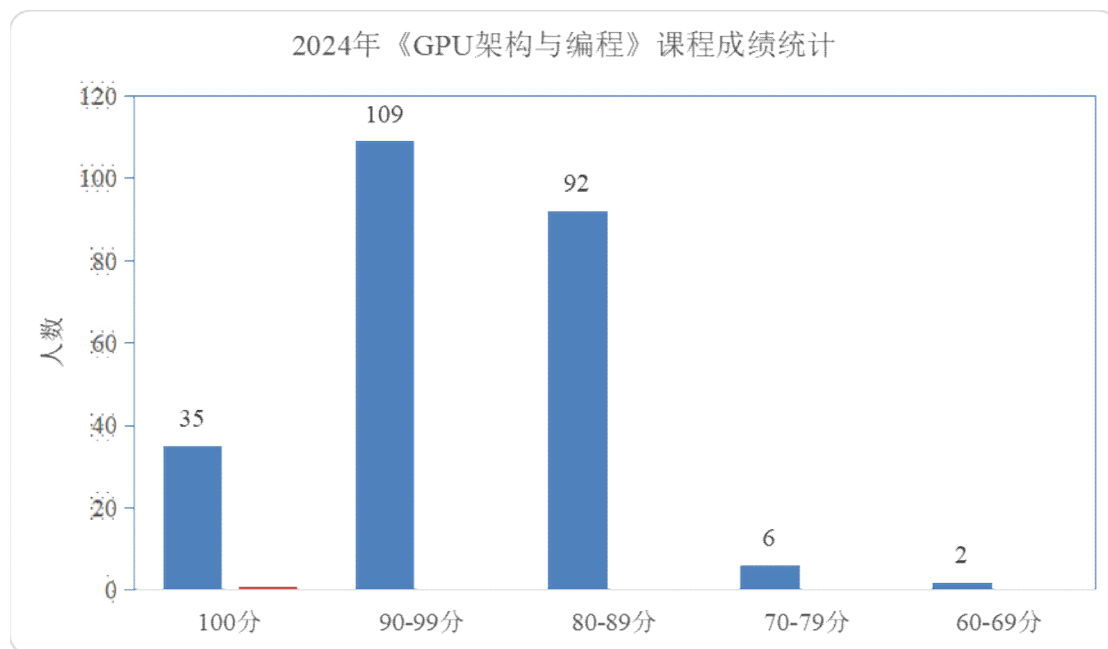
往届课程（23秋颁奖典礼）



往届课程（24秋）：244人获得学分



往届课程（24秋）



往届课程（24秋颁奖典礼）



摩尔线程董事长张建中先生颁奖



讲授内容

- 课程介绍
- GPU简介
 - 定义、发展史、总体构架
 - 功能（一）：图形图像处理
 - 功能（二）：通用计算、编程语言
 - 模拟器、生产商

GPU和GPGPU



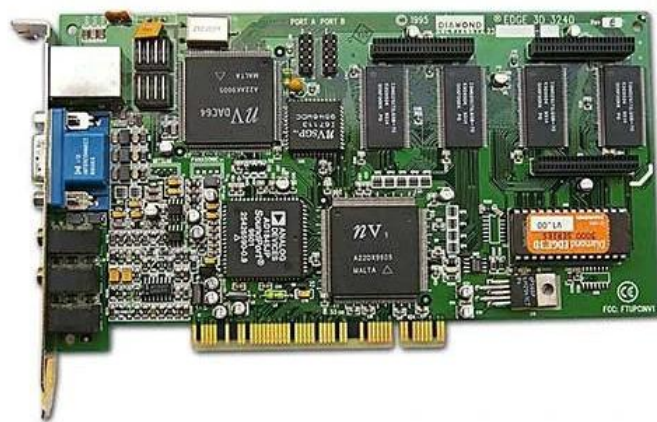
- ✓ GPU的全称是 Graphics Processing Unit，是一种用于图形图像处理的**处理器**；
- ✓ 因为图形和图像处理方面的特点，对于并行处理大块数据的算法，GPU的并行结构比通用CPU更高效。将GPU用于通用计算，形成了GPGPU。

GPU和GPGPU



AMD's 2016-2017 Datacenter Roadmap: x86, ARM, and GPGPU, <https://www.anandtech.com/show/9234/amds-20162017-datacenter-roadmap-x86-arm-and-gpgpu>

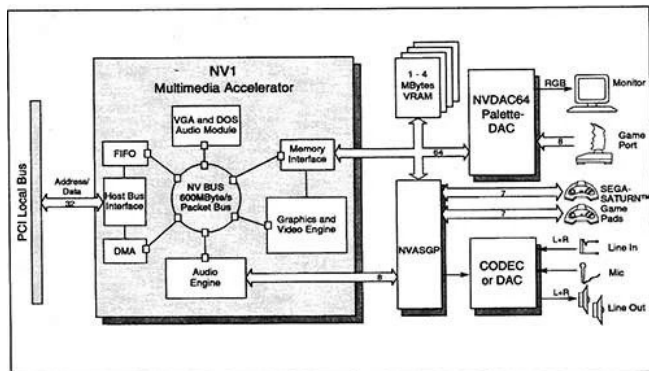
英伟达NV1



- ✓英伟达成立于1993年初，英伟达的第一代媒体加速器将使PC平台能够提供高性能的2D和3D图形、视频、数字操纵杆界面和高级音频功能，以加速快速增长的多媒体应用程序安装基础。
- ✓英伟达销售NV1 VRAM版本，而SGS提供DRAM版本STG2000。

Jon Peddie, Nvidia's Quadratic Processor, the NV1

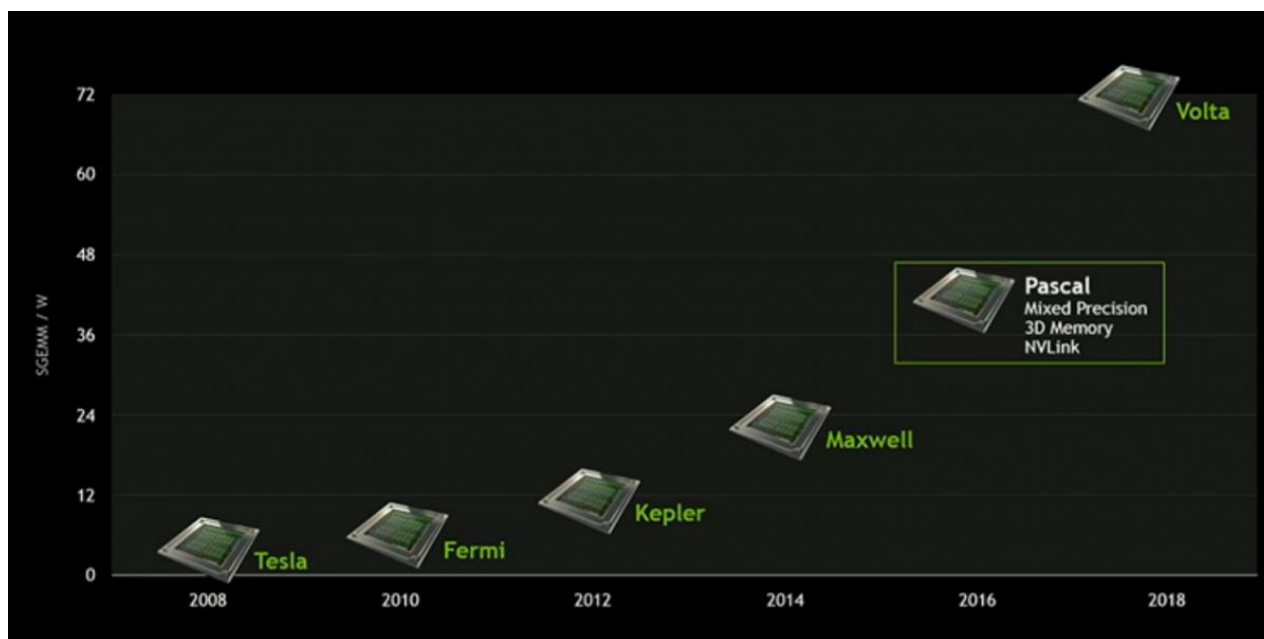
英伟达NV1



- ✓ 250万个晶体管，单芯片多媒体加速器；
- ✓ 支持行业标准：Windows 3.11 下所有 DirectX API 的加速、全动态视频加速（Indeo, MPEG, Cinepak）、等；
- ✓ 实时纹理映射 3D 图形；
- ✓ 加速所有 3D 标准：三角形、四边形和曲线。

Jon Peddie, Nvidia's Quadratic Processor, the NV1

GPU的发展史（英伟达GPU为例）



slides from Nvidia

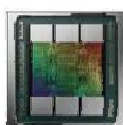
GPU的发展史（英伟达GPU为例）

Highest AI Compute in a Single GPU

- Build each GPU to reticle limit as intra-GPU communication provides:
 - Highest communication density
 - Lowest latency
 - Optimal energy efficiency



Volta
 >21 billion transistors
 815mm²
 TSMC 12nm FFN



Ampere
 >54 billion transistors
 826 mm²
 TSMC N7



Hopper
 >80 billion transistors
 814 mm²
 TSMC 4N



Blackwell
 >208 billion transistors
 >1600 mm²
 TSMC 4NP



slides from Nvidia

GPU的发展史：英伟达GB200

NVIDIA GB200 Grace Blackwell Superchip

- GB200 Grace Blackwell Superchip
 - 1 Grace CPU and 2 Blackwell GPUs
 - NVLINK-C2C interconnect
 - 40 PetaFLOPS FP4 / 20 PetaFLOPS FP8
- Grace Blackwell Compute Tray
 - 2 Grace CPUs and 4 Blackwell GPUs



High Bandwidth / Low Latency



Low-Power High Density



Key Value (KV) Caching



slides from Nvidia

GPU的发展史：英伟达GB200

5th Generation Tensor Cores — FP Formats Summary

- New FP4 and FP6 precision support
- New micro-scaled formats for FP4, FP6, and FP8
- 4x faster per-clock, per-SM FP4 vs. Hopper FP8
- Blackwell also increases operating frequency and SM count

Format	Hopper SM M100		Blackwell SM M100		Blackwell Speedup per clock per SM
	Dense	Sparsity	Dense	Sparsity	
FP16	2048	4096	4096	8192	2x
BF16	2048	4096	4096	8192	2x
FP8 (1/2-scale)	4096	8192	8192	16384	2x
FP6 (1/2-scale)	-	-	8192	16384	New! 2x of Hopper FP6
FP4 (1/2-scale)	-	-	16384	32768	New! 4x of Hopper FP6



slides from Nvidia

GPU的发展史：英伟达GB200

FP4 Inference Accuracy

- Data from Blackwell silicon with quantized FP4
- Excellent MMLU* scores for LLMs
- Same accuracy even for Nemotron-4 340B model
- Achieved with full-stack hardware, software and algorithm co-design

Blackwell FP4 and NVIDIA Quasar Quantization System Measured MMLU Scores

Model	BF16	Quantized FP4
Nemotron-4 15B	64.2	64.5
Nemotron-4 340B	81.1	81.1

Nemotron-4 340B Technical Report

<https://nvidia.github.io/nemotron-4-340B-BF16/>

*Massive Multitask Language Understanding



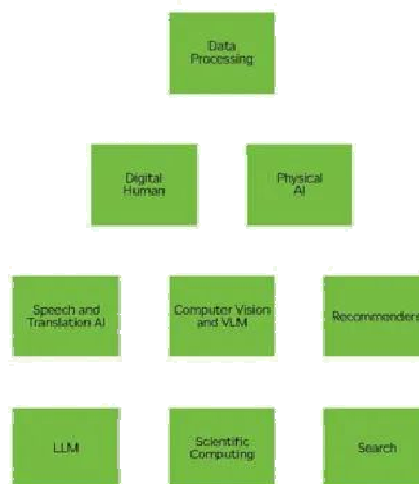
slides from Nvidia

GPU的发展史：英伟达GB200

Over 400 NVIDIA CUDA-X Libraries

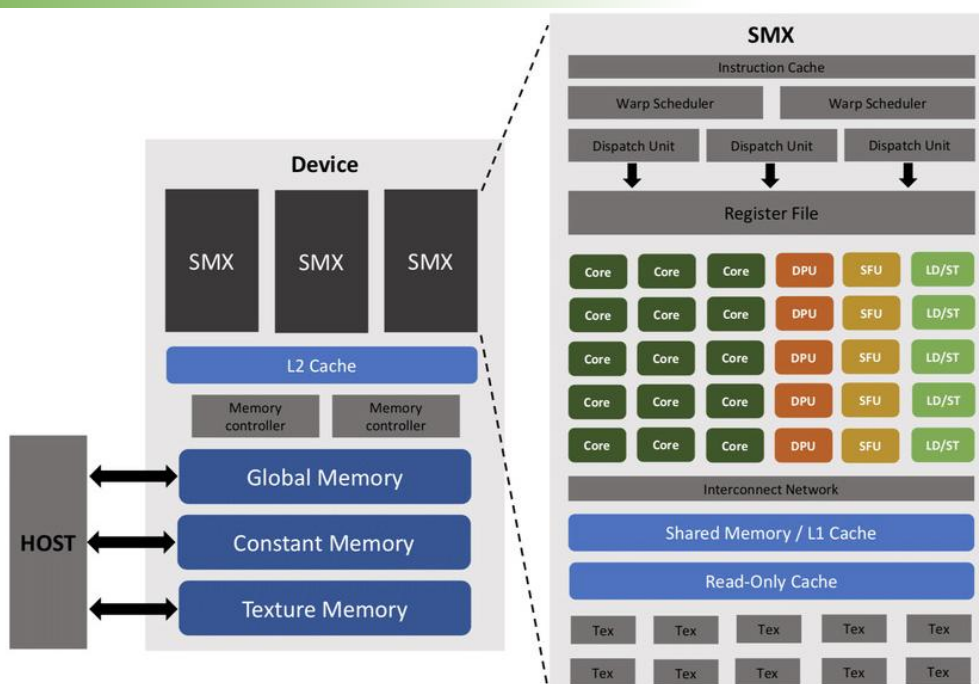
Blackwell optimized to deliver maximum performance

- Optimized libraries for each platform
- Targeting diverse application domains
- Built on our decades-long innovation
- Ever-expanding set of algorithms



slides from Nvidia

GPU架构（英伟达GPU为例）

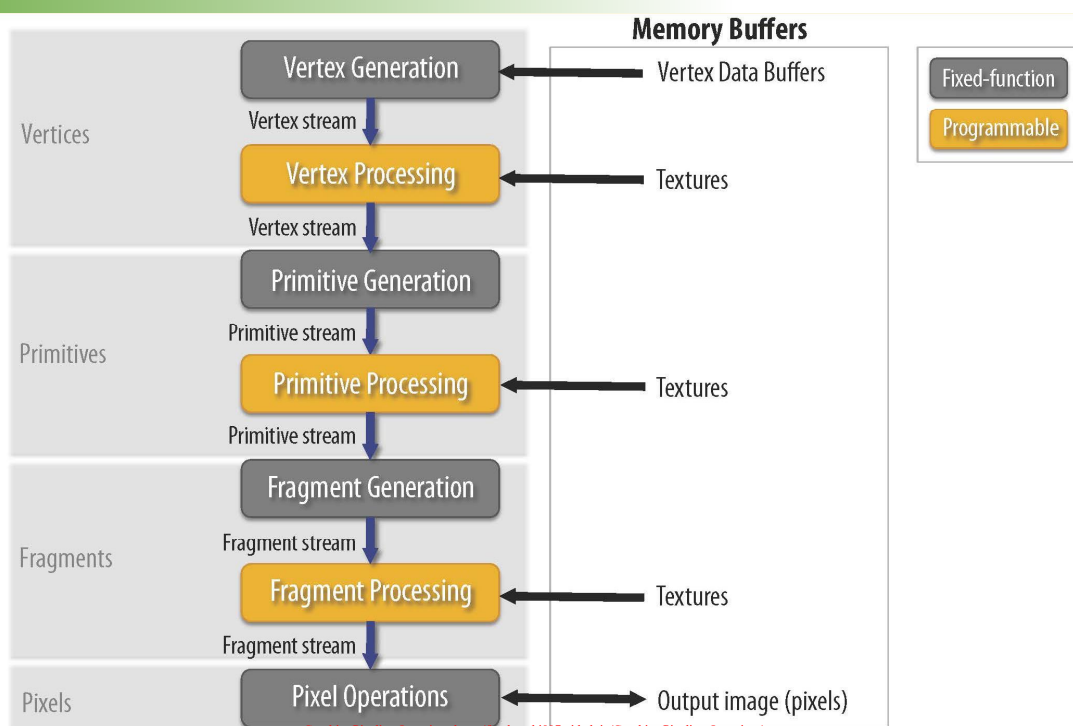


Yehia Arafat, Abdel-Hameed Badawy, Gopinath Chennupati, Nandakishore Santhi, Stephan Eidenbenz, Low Overhead Instruction Latency Characterization for NVIDIA GPGPUs, <https://arxiv.org/abs/1905.08778>

讲授内容

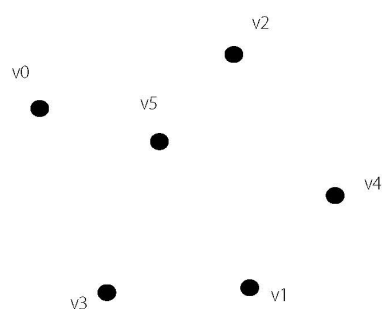
- 课程介绍
- GPU简介
 - 定义、发展史、总体构架
 - 功能（一）：图形图像处理
 - 功能（二）：通用计算、编程语言
 - 模拟器、生产商

GPU图形图像处理



顶点处理（Vertex Processing）

Vertices are transformed into “screen space”



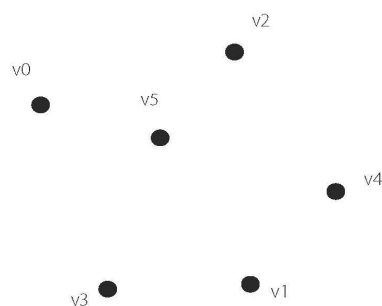
Vertices

✓每个顶点处理（Vertex Processing）都是独立的。

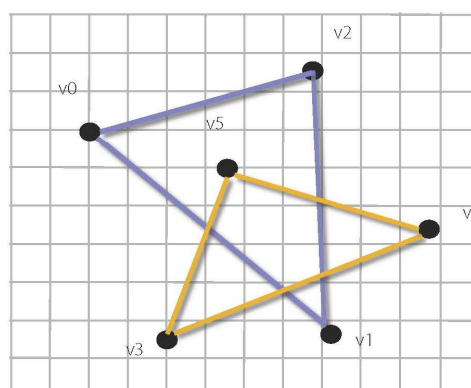
Graphics Pipeline Overview, <https://benkenobi007.github.io/Graphics-Pipeline-Overview/>

图元处理（Primitive Processing）

Then organized into primitives that are clipped and culled...



Vertices

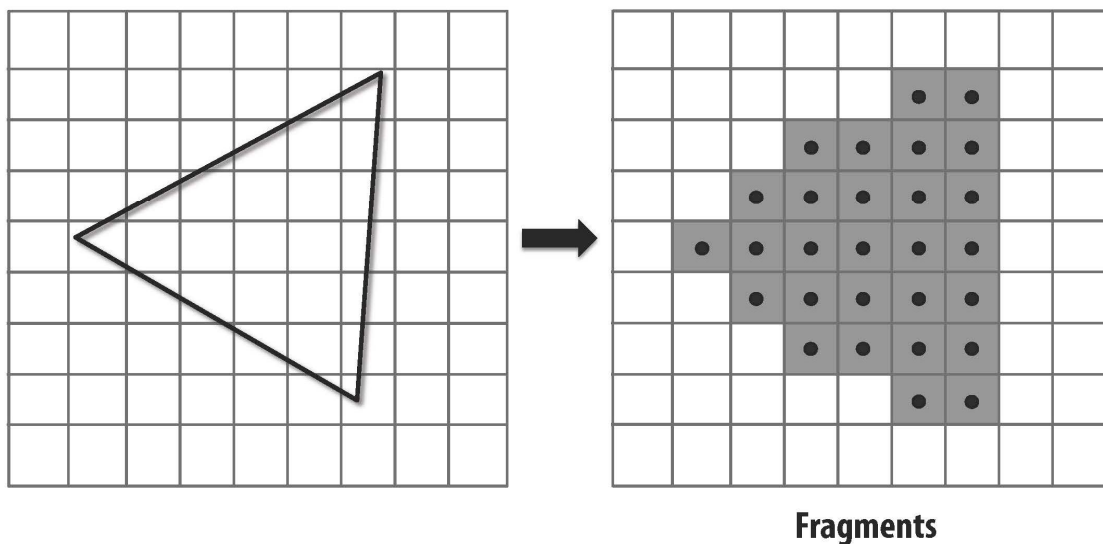


Primitives
(triangles)

Graphics Pipeline Overview, <https://benkenobi007.github.io/Graphics-Pipeline-Overview/>

光栅化（Rasterization）

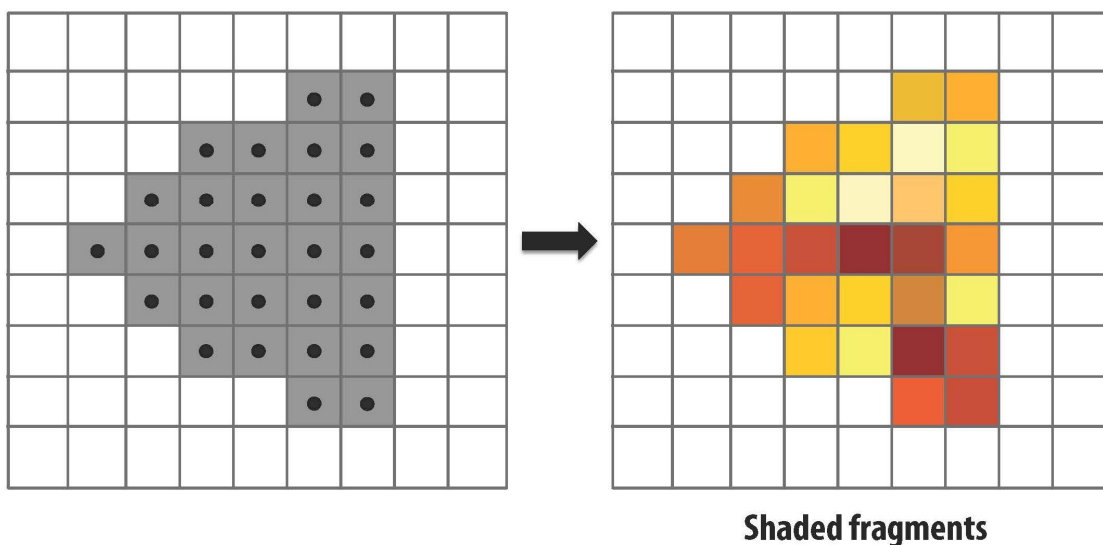
Primitives are rasterized into "pixel fragments"



Graphics Pipeline Overview, <https://benkenobi007.github.io/Graphics-Pipeline-Overview/>

光栅化（Rasterization）

Fragments are shaded to compute a color at each pixel

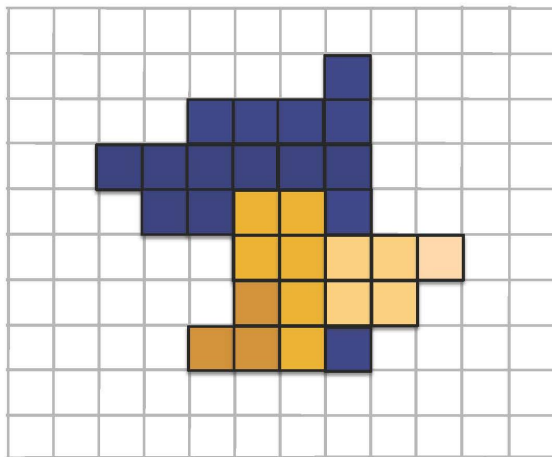


✓每个图元（Primitive）的光栅化（Rasterization）都是独立的。

Graphics Pipeline Overview, <https://benkenobi007.github.io/Graphics-Pipeline-Overview/>

像素操作 (Pixel Operation)

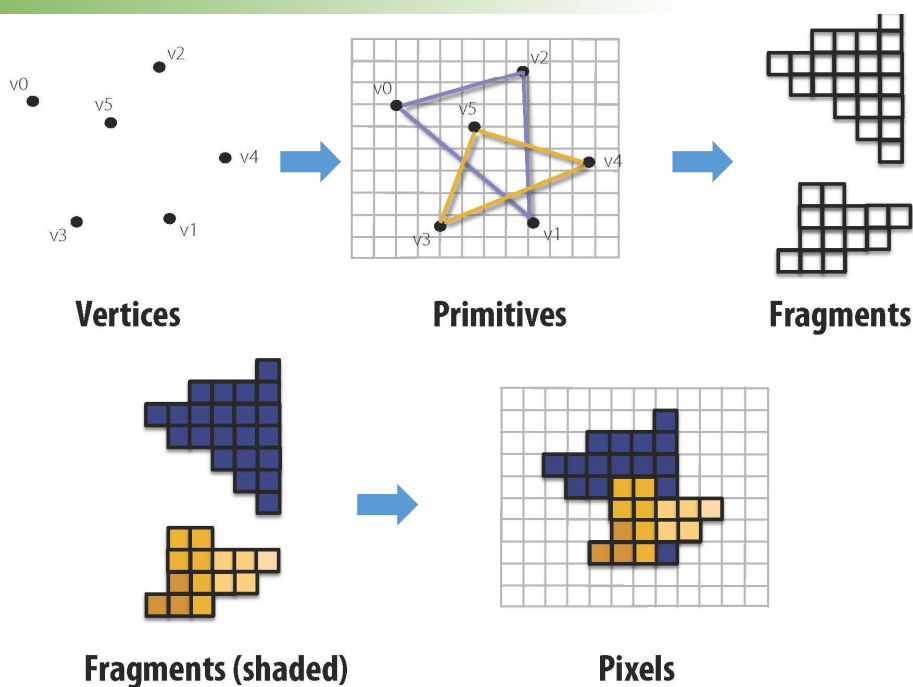
Fragments are blended into the frame buffer at their pixel locations (z-buffer determines visibility)



Pixels

Graphics Pipeline Overview, <https://benkenobi007.github.io/Graphics-Pipeline-Overview/>

流水线实体 (Pipeline Entities)

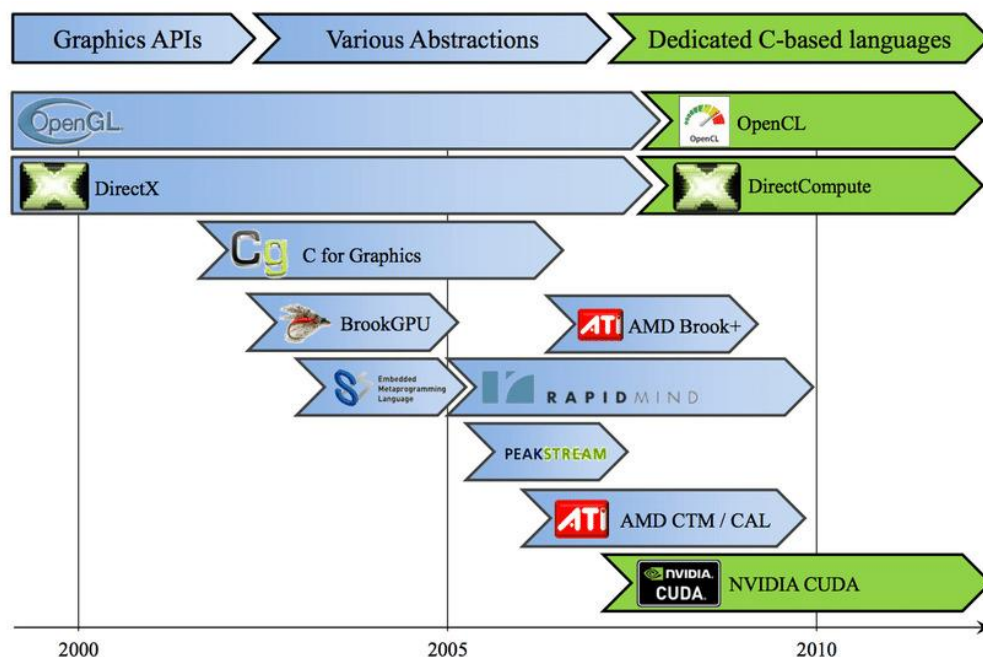


Graphics Pipeline Overview, <https://benkenobi007.github.io/Graphics-Pipeline-Overview/>

讲授内容

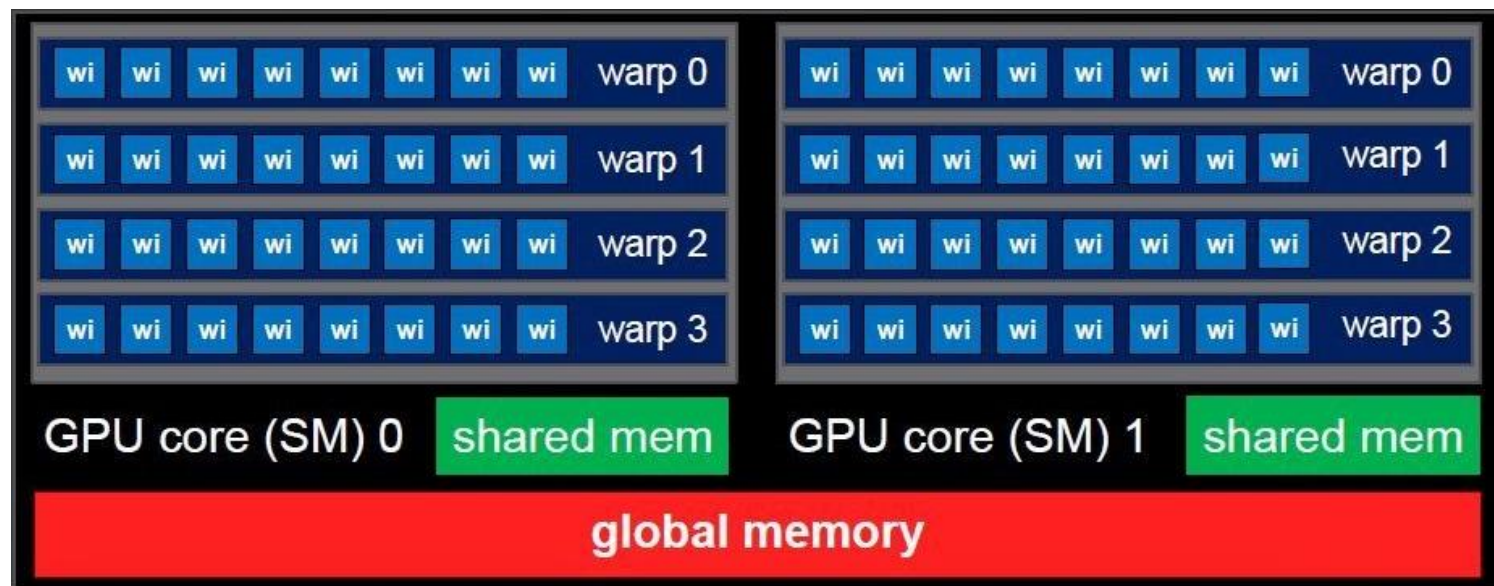
- 课程介绍
- GPU简介
 - 定义、发展史、总体构架
 - 功能（一）：图形图像处理
 - 功能（二）：通用计算、编程语言
 - 模拟器、生产商

GPU编程语言

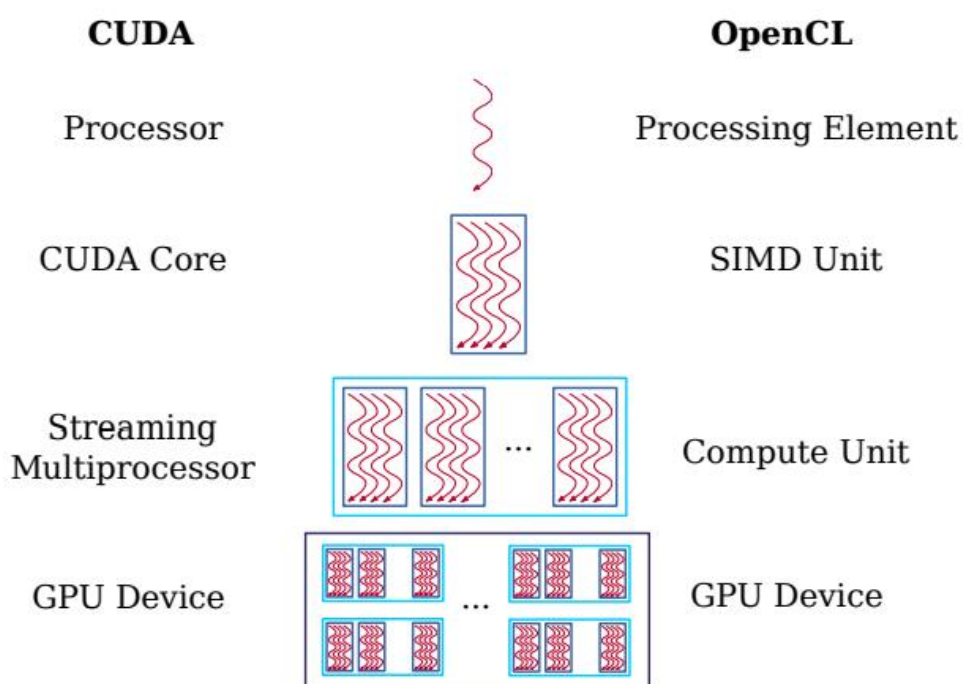


André R. Brodtkorb, Trond R. Hagen, Christian Schulz, Geir Hasle, GPU computing in discrete optimization. Part I: Introduction to the GPU, EURO Journal on Transportation and Logistics, 2013;

GPU架构：编程相关（英伟达GPU为例）



GPU编程语言：CUDA v.s. OpenCL



D. vom Bruch, Real-time data processing with GPUs in high energy physics, Journal of Instrumentation, 2020;

CUDA编程模型

Grid with Clusters

Thread Block Cluster

Thread Block



Thread Block



Thread Block



Thread Block



Thread Block Cluster

Thread Block



Thread Block



Thread Block

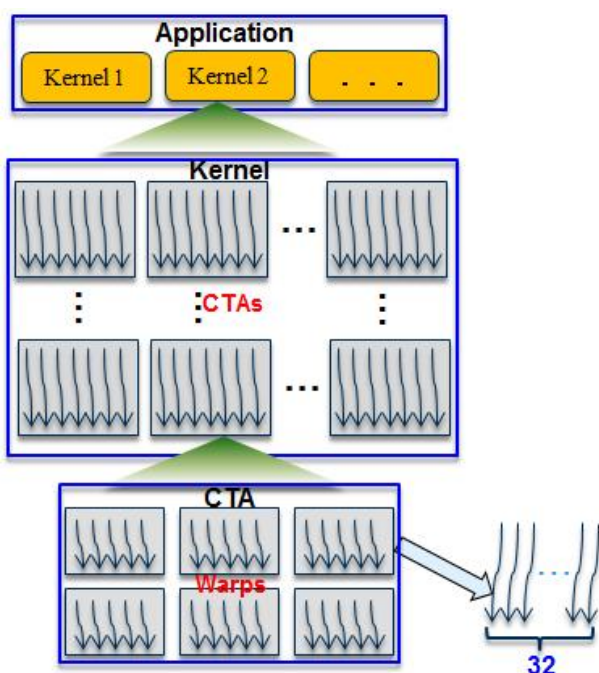


Thread Block



CUDA C++ Programming Guide, Nvidia;

CUDA编程模型



✓请描述thread是如何组织起来执行CUDA程序中kernel的?

Overview of GPGPU Architecture (NVIDIA Fermi Based), <https://cinwell.wordpress.com/2013/09/06/overview-of-gpu-architecture-fermi-based/>

讲授内容

- 课程介绍
- GPU简介
 - 定义、发展史、总体构架
 - 功能（一）：图形图像处理
 - 功能（二）：通用计算、编程语言
 - 模拟器、生产商

GPU模拟器



GPGPU-Sim



Accel-Sim



NVIDIA NVAS

GPU模拟器

GPU	指令集	模式	图形支持能力	软件栈支持
Ventus (承影)	RISC-V	SIMT	No	N/A
HWACHA	RISC-V	Vector	No	N/A
Simty	RISC-V	SIMT	No	N/A
MIAOW	AMD	SIMT	No	OpenCL
Scratch	AMD	SIMT	No	OpenCL
GPLGPU (kickstarter)	Custom		shaders 2D/3D Texture	
Theia	Custom	SIMT	Shaders Texture Units	OpenCL
FlexiGrip	Custom	SIMT	No	Custom
FGPU	Custom MIPS based ISA	SIMT	No	Custom 来源: 陈巍谈芯
NyuziRaster	Custom	SIMT	Fixed-Function Rasterizer	Custom
Vortex	RISC-V	SIMT	Shaders Texture Units	OpenCL/OpenGL

<https://zhuanlan.zhihu.com/p/494149559>

GPU模拟器

表 2 异构模拟器对比

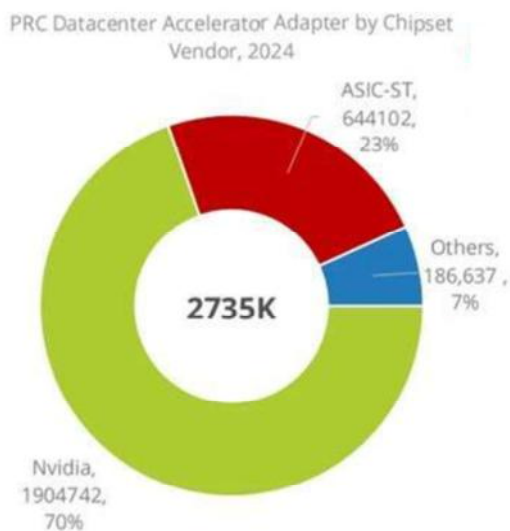
Simulator	Published Year	Developer and Link	Forward Mode	Citation Count	Language	Target Architecture	Operating System	Last Update Date
GPGPU-Sim ^[37]	2009	University of British Columbia https://github.com/gpgpu-sim/gpgpu-sim_simulations	Functional/ Cycle Accurate	1232	C++	GPGPU	×	2018-11
Qsilver ^[38]	2004	The University of Virginia Link not found	Cycle Accurate	96		GPU	×	
gem5-APU ^[39]	2015	AMD https://gem5.googlesource.com/amd/gem5	Cycle Accurate		C++ Python	APU	✓	2018-11
GPU Ocelot ^[40]	2009	Georgia Institute of Technology http://code.google.com/p/gpuocelot/	Functional	25	C++	GPU	×	2013-04
FusionSim ^[41]	2013	University of Toronto https://sites.google.com/site/fusionsimulator/	Cycle Accurate	16	C++	CPU-GPU	×	2012-06
MV5 ^[42]	2011	Argonne National Laboratory https://sites.google.com/site/mv5sim	Cycle Accurate	11	C++ Python	GPU-like SIMD/SIMT	×	2011-02
GpuTejas ^[43]	2014	Indian Institute of Technology http://www.cse.iitd.ac.in/tejas/gputejas/index.html	Cycle Accurate	5	Java	GPU	×	2019-05
gem5-gpu ^[44]	2014	University of Wisconsin-Madison https://gem5-gpu.cs.wisc.edu/wiki/	Functional/ Cycle Accurate	126	C++ Python	CPU-GPU	✓	2017-01
HSAemu ^[45]	2014	National Tsing Hua University https://github.com/SSLAB-HSA/HSAemu	Functional/ Cycle Accurate	10	C	Heterogeneous System	✓	2013-11
FATSEA ^[46]	2010	University of Murcia (Spain) Link not found	Cycle Accurate	1	C++	GPU	×	
Berra ^[47]	2009	Univ.de Perpignan https://forge.inria.fr/scm/?group_id=5827	Functional	100	C++	GPGPU	×	2015-08
ATTILA ^[48]	2006	Norwegian University of Science and Technology https://github.com/attila-gps/attila-sim	Cycle Accurate	112	C++	GPU	×	2015-04
Multi2Sim ^[49]	2012	Northeastern University https://github.com/Multi2Sim/multi2sim	Cycle Accurate	351	C++	CPU-GPU	×	2018-04
MacSim ^[50]	2012	Georgia Tech https://github.com/gthparch/macsim	Cycle Accurate	39	C++	CPU-GPU	×	2018-11
MGSim ^[51]	2013	Northeastern University https://github.com/svp-dev/mgsim	Cycle Accurate	19	C++ C	CPU-GPU	×	2017-04

Notes: ✓ means support operating system; × means not support operating system

<https://zhuanlan.zhihu.com/p/494149559>

知乎 @ 牛翠希尼莎聊

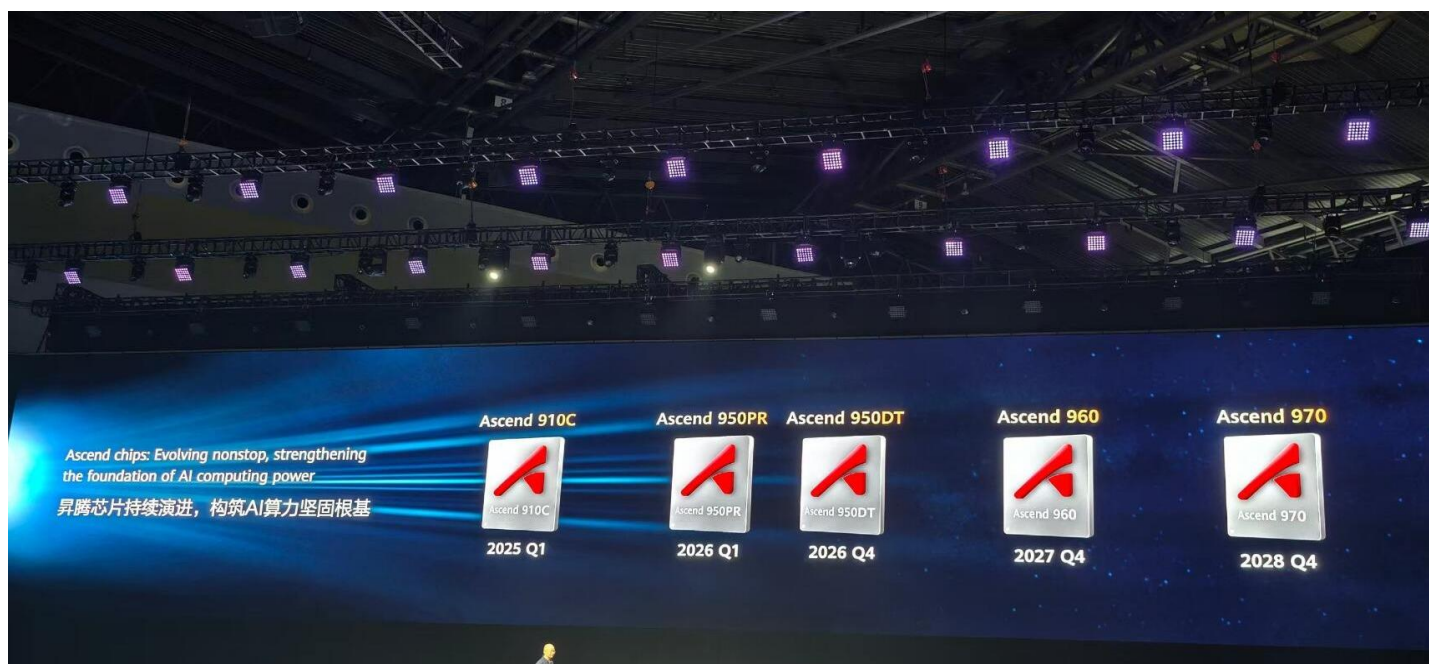
GPU的生产商：



2024年中国市场各品牌GPU市场份额

slides from IDG

国产GPU的生产商



华为昇腾AI芯片路线图公布 <https://wallstreetcn.com/articles/3755816?keyword=ai>

国产GPU的生产商

企业名称	成立时间	核心技术	主要产品	应用领域
景嘉微	2006 年 4 月	支持国产 CPU 和国产操作系统的自主知识产权 GPU	JM5400、JM7201、JM92 系列	军用工业、人工智能、云计算、金融、教育等领域
芯瞳半导体	2019 年 12 月	统一渲染 GPU 架构，具有高度可扩展的互联结构和计算阵列	GenBu01 GPU	信创、能源、交通、金融、人工智能等领域
璧仞科技	2019 年 9 月	自主原创的 GPU 芯片架构	BR100 GPU	人工智能、云计算、图形渲染、大数据等领域
摩尔线程	2020 年 10 月	3D 图形计算和高性能并行计算技术	MTT S60、MTT S2000、MTT S10	物理仿真、AI 计算、游戏娱乐、自动驾驶等领域
沐曦集成电路	2020 年 9 月	自主研发高性能 GPU 芯片架构及兼容国际主流生态的完整软件栈	MXN、MXC、MXG	深度学习、数据分析、物理仿真、云游戏和元宇宙等领域
瀚博半导体	2018 年 12 月	云端推理 DSA 架构、视频处理技术	SV100 系列云端推理芯片、VA1 通用 AI 推理加速卡	深度学习、AI 推理、云计算等领域
登临科技	2017 年 11 月	软件定义的异构人工智能计算平台	Goldwasser	自动驾驶、智慧城市、生物医疗、AI 计算等领域
天数智芯	2015 年 12 月	通用 GPU 云端芯片及超级算力系统提供商	7 纳米 GPGPU 高端自研云端训练芯片	自动驾驶、智慧医疗、智慧金融、智慧教育等领域
芯动科技	2007 年 10 月	一站式高速混合电路 IP 及芯片定制解决方案	风华 1 号	元宇宙、云游戏、云桌面、AI 计算等领域
龙芯中科	2010 年 4 月	自研的统一渲染架构	7A2000 桥片的 GPU 模块	金融、政务办公、网安、教育、通信、医疗等领域

现状与未来：国产GPU能否超越英伟达AMD？<https://www.163.com/dy/article/HGL8JE7K05312NX9.html>

THANKS