PROGRAMOWANIE WEKTOROWE I RÓWNOLEGŁE

Wykład dla kierunku: Matematyka stosowana i technologie informatyczne

ARCHITEKTURA PROCESORÓW GRAFICZNYCH GPU

HIERARCHIA PAMIĘCI CUDA

Alokacja pamięci posiada określoną hierarchię. Kompilator CUDA C automatycznie obsługuje alokację pamięci, programiści CUDA mają możliwość optymalizacji wykorzystania pamięci. Na kolejnych slajdach zostaną przedstawione kluczowe pojęcia dotyczące hierarchii pamięci CUDA.

REJESTRY

Rejestry:

to pamięć przydzielana poszczególnym wątkom (rdzenie CUDA). Ponieważ rejestry istnieją w pamięci GPGPU i są dedykowane poszczególnym wątkom, dane przechowywane w nich mogą być przetwarzane szybciej niż jakiekolwiek inne dane. Alokacja pamięci w rejestrach jest obsługiwana przez kompilatory, a nie kontrolowana przez programistów CUDA.

PAMIĘĆ TYLKO DO ODCZYTU

Pamięć tylko do odczytu (Read Only Memory) to pamięć wbudowana w multiprocesory strumieniowe (SM) GPU. Jest używana do określonych zadań, takich jak pamięć tekstur, do której można uzyskać dostęp za pomocą funkcji tekstur CUDA. W wielu przypadkach pobieranie danych z pamięci tylko do odczytu może być szybsze i wydajniejsze niż korzystanie z pamięci globalnej.

PAMIĘĆ PODRĘCZNA L1 / PAMIĘĆ WSPÓŁDZIELONA

Pamięć podręczna poziomu 1 (cache L1) i pamięć współdzielona (shared) to pamięć wbudowana w układ GPGPU, która jest współdzielona w ramach bloków wątków (bloki CUDA) - SM. Pamięć podręczna L1 i pamięć współdzielona są wbudowane w układ i są szybsze niż pamięć podręczna L2 i pamięć globalna. Podstawowa różnica między pamięcią podręczną L1, a pamięcią współdzieloną polega na tym, że użycie pamięci współdzielonej jest kontrolowane przez kod programu, podczas gdy pamięć podręczna L1 jest kontrolowana sprzętowo.

PAMIĘĆ PODRĘCZNA L2

Dostęp do pamięci podręcznej poziomu 2 (cache L2) mają wszystkie wątki we wszystkich blokach CUDA.

Pamięć podręczna L2 przechowuje fragmenty pamięci globalnej. Pobieranie danych z pamięci podręcznej L2 jest szybsze niż pobieranie danych z pamięci globalnej.

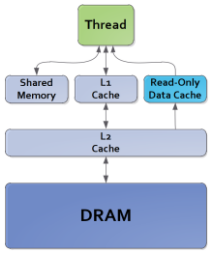
PAMIĘĆ GLOBALNA

Pamięć globalna to pamięć znajdująca się w pamięci DRAM urządzenia poza układem GPGPU. Pamięć globalna jest pamięcią RAM zamontowaną na karcie graficznej. Pobieranie danych z pamięci globalnej jest znacznie wolniejsze niż pobieranie ich z pamięci podręcznej L2.

PAMIĘĆ GLOBALNA

Pamięć globalna jest podłączona do układu GPGPU poprzez magistrale adresową, danych i sterującą. Rozmiar magistrali adresowej zależy od rozmiaru pamięci, natomiast szerokość magistrali danych zależy od układu GPGPU i może wynosić: 128, 192, 256, 320 lub 384 bity. Szersza szyna danych pozwala na większy transfer danych pomiędzy pamięcią globalną, a układem GPGPU.

HIERARCHIA PAMIĘCI CUDA



TAKSONOMIA FLYNNA

Flynn wyróżnił cztery klasy:

SISD (ang. single-instruction-single-data) równoważną przetwarzaniu całkowicie sekwencyjnemu;

SIMD (ang. single-instruction-multiple-data), gdzie wykonuje się te same operacje na różnych zbiorach danych;

MISD (ang. multiple-instruction-single-data) wykonuje się różne operacje na tym samym zbiorze danych (systolic arrays);

MIMD (ang. Multiple-instruction-multiple-data), gdzie różne operacje wykonywane są na różnych zbiorach danych.

UZUPEŁNIENIE TAKSONOMII FLYNNA

SIMT (ang. single-instruction-multiple-thread), gdzie wykonuje się te same operacje na różnych zbiorach danych przez różne wątki.

HIERARCHIA OBLICZENIOWA CUDA

Zasoby przetwarzania w CUDA zostały zaprojektowane, aby pomóc zoptymalizować wydajność w przypadku użycia GPU. Trzy podstawowe elementy hierarchii to wątki, bloki wątków i siatki bloków.

WĄTKI

Wątek to dynamiczny ciąg działań wykonywanych na podstawie programu przez pojedynczy rdzeń GPGPU.

Rdzeń CUDA to równoległy procesor, który wykonuje obliczenia matematyczne zmiennoprzecinkowe w GPGPU Nvidii. Wszystkie dane przetwarzane przez GPU są przetwarzane przez rdzenie CUDA.

Każdy rdzeń CUDA ma swoją własną pamięć rejestrów, które nie są dostępne dla innych wątków/rdzeni.

Nowoczesne procesory graficzne mają setki, a nawet tysiące rdzeni CUDA.

WĄTKI

Związek między mocą obliczeniową, a rdzeniami CUDA nie jest idealnie liniowy, ogólnie rzecz biorąc – i zakładając, że wszystko inne jest równe – im więcej rdzeni CUDA ma GPGPU, tym większą ma moc obliczeniową.

Istnieje jednak wiele wyjątków od tej ogólnej reguły. Na przykład różne mikroarchitektury GPU mogą wpływać na wydajność i sprawić, że GPU z mniejszą liczbą rdzeni CUDA będzie wydajniejsze.

BLOKI WĄTKÓW

Blok wątków — lub blok CUDA — to grupa wątków (rdzeni) CUDA, które mogą być wykonywane razem szeregowo lub równolegle.

Logiczne pogrupowanie rdzeni umożliwia wydajniejsze mapowanie danych.

Bloki wątków współdzielą pamięć. Obecna architektura CUDA ogranicza liczbę wątków na blok do 1024. Każdy wątek w danym bloku CUDA może uzyskać dostęp do tej samej pamięci współdzielonej.

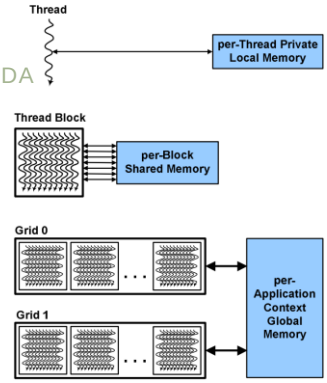
SIATKI BLOKÓW

Siatki bloków (Block grids) stanowią następną warstwę abstrakcji ponad blokami wątków. Siatki bloków to grupy bloków wątków o tym samym jądrze (kernel).

Siatki mogą być używane do równoległego wykonywania większych obliczeń (np. tych, które wymagają więcej niż 1024 wątków).

UWAGA: ponieważ różne bloki wątków nie mogą korzystać z tej samej pamięci współdzielonej, to muszą się synchronizować inaczej niż wątki w bloku.

HIERARCHIA CUDA



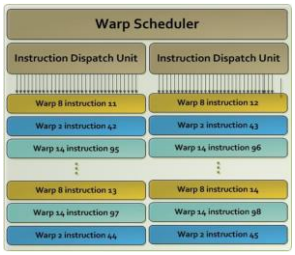
PLANISTA OSNOWY

Planista osnowy (Warp Scheduler) w SM planuje wątki w grupach zwanych osnowami (warp) po 32 równoległe wątki.

Każdy Warp Scheduler zawiera dwie jednostki wysyłania instrukcji.

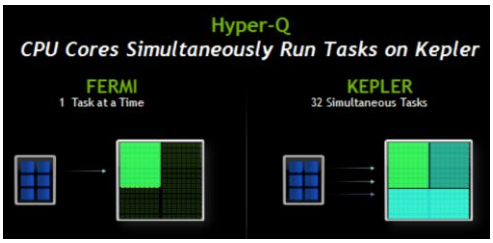
W każdym cyklu można wysłać dwie niezależne instrukcje na warp.

PLANISTA OSNOWY



HYPER-Q

Możliwość uruchamiania wielu zadań jednocześnie.

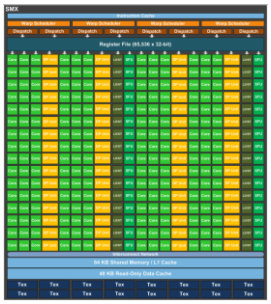


WYBRANE ARCHITEKTURY GPGPU - NVIDIA

- Kelvin
- Rankine
- Curie
- Tesla
- Fermi
- Kepler
- Maxwell
- Pascal
- Turing
- Ampere
- Ada Lovelace

Nazwy pochodzą od nazwisk znanych ludzi

ARCHITEKTURA KEPLER – GTX6X0



Moduł SMX zawiera:
- 192 rdzenie pojedynczej precyzji
- 64 rdzenie podwójnej precyzji
- 32 jednostki specjalne,
- 32 jednostki odczytująco-zapisujące.

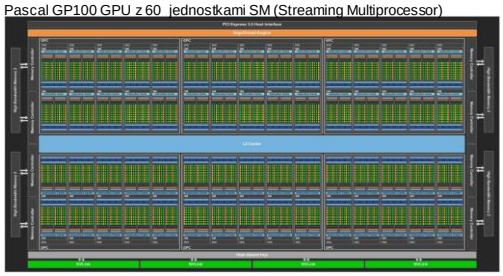
ARCHITEKTURA KEPLER – GTX6X0

Porównanie Fermi i Kepler GPU

	FERMI GF100	FERMI GF104	KEPLER GK104	KEPLER GK110	KEPLER GK210
Compute Capability	2.0	2.1	3.0	3.5	3.7
Threads / Warp	32				
Max Threads / Thread Block	1024				
Max Warps / Multiprocessor	48		64		
Max Threads / Multiprocessor	1536		2048		
Max Thread Blocks / Multiprocessor	8		16		
32-bit Registers / Multiprocessor	32768		65536		131072
Max Registers / Thread Block	32768		65536		65536
Max Registers / Thread	63			255	
Max Shared Memory / Multiprocessor	48K				112K
Max Shared Memory / Thread Block	48K				
Max X Grid Dimension	2^16-1		2^32-1		
Hyper-Q	No			Yes	
Dynamic Parallelism	No			Yes	

ARCHITEKTURA PASCAL – GTX10X0

Pascal GP100 GPU z 60 jednostkami SM (Streaming Multiprocessor)



ARCHITEKTURA PASCAL – GTX10X0

Układ zawiera:
- 6 jednostek Graphics Processing Cluster (GPC) po 10 SM
- 4 MB cache L2

Moduł SM zawiera:
- 64 rdzenie pojedynczej precyzji
- 32 rdzenie podwójnej precyzji
- 16 jednostek specjalne,
- 16 jednostek odczytująco-zapisujące
- 64 KB pamięci współdzielonej.

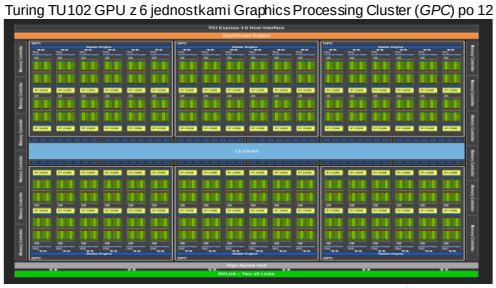
ARCHITEKTURA PASCAL – GTX10X0

Porównanie Kepler, Maxwell i Pascal GPU

GPU	Kepler GK110	Maxwell GM200	Pascal GP100
Compute Capability	3.5	5.2	6.0
Threads / Warp	32	32	32
Max Warps / Multiprocessor	64	64	64
Max Threads / Multiprocessor	2048	2048	2048
Max Thread Blocks / Multiprocessor	16	32	32
Max 32-bit Registers / SM	65536	65536	65536
Max Registers / Block	65536	32768	65536
Max Registers / Thread	255	255	255
Max Thread Block Size	1024	1024	1024
Shared Memory Size / SM	16 KB/32 KB/48 KB	96 KB	64 KB

ARCHITEKTURA TURING – RTX20X0

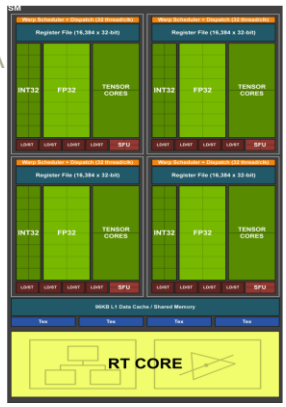
Turing TU102 GPU z 6 jednostkami Graphics Processing Cluster (*GPC*) po 12 SM



ARCHITEKTURA TURING – RTX20X0

Układ zawiera:
- 6 jednostek Graphics Processing Cluster (GPC) po 12 SM
- 6 MB cache L2

Moduł SM zawiera:
- 2 rdzenie podwójnej precyzji
- 64 rdzenie pojedynczej precyzji
- 64 rdzenie całkowite
- 4 jednostki specjalne,
- 16 jednostek odczytująco-zapisujących
- 8 rdzeni Tensor
- 1 jednostkę śledzenia promieni
- 96 KB pamięci współdzielonej/L1.



ARCHITEKTURA TURING – RTX20X0

Porównanie Pascal i Turing GPU

GPU Features	GTX 1080Ti	RTX 2080 Ti
Architecture	Pascal	Turing
GPCs	6	6
TPCs	28	34
SMs	28	68
CUDA Cores / SM	128	64
CUDA Cores / GPU	3584	4352
Tensor Cores / SM	NA	8
Tensor Cores / GPU	NA	544
RT Cores	NA	68
GPU Base Clock MHz (Reference / Founders Edition)	1480 / 1480	1350 / 1350

ARCHITEKTURA AMPERE – RTX30X0

Ampere GA102 GPU z 7 jednostkami Graphics Processing Cluster (*GPC*) po 12 SM

ARCHITEKTURA AMPERE – RTX30X0

Układ zawiera:
7 jednostek Graphics Processing Cluster (GPC) po 12 SM
6 MB cache L2
Moduł SM zawiera:
- 2 rdzenie podwójnej precyzji
- 128 rdzeni pojedynczej precyzji
- w tym 64 rdzenie całkowite
- 4 jednostki specjalne,
- 16 jednostek odczytująco-zapisujących
- 4 rdzeni Tensor
- 1 jednostkę śledzenia promieni
- 128 KB pamięci współdzielonej/L1.

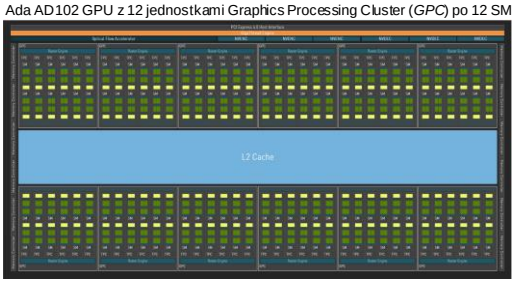
ARCHITEKTURA AMPERE – RTX30X0

Porównanie Turing i Ampere GPU

Graphics Card	GeForce RTX 2080 Founders Edition	GeForce RTX 2080 Super Founders Edition	GeForce RTX 3080 10 GB Founders Edition
GPU Codename	TU104	TU104	GA102
GPU Architecture	NVIDIA Turing	NVIDIA Turing	NVIDIA Ampere
GPCs	6	6	6
TPCs	23	24	34
SMs	46	48	68
CUDA Cores / SM	64	64	128
CUDA Cores / GPU	2944	3072	8704
Tensor Cores / SM	8 (2nd Gen)	8 (2nd Gen)	4 (3rd Gen)
Tensor Cores / GPU	368	384 (2nd Gen)	272 (3rd Gen)
RT Cores	46 (1st Gen)	48 (1st Gen)	68 (2nd Gen)

ARCHITEKTURA ADA– RTX40X0

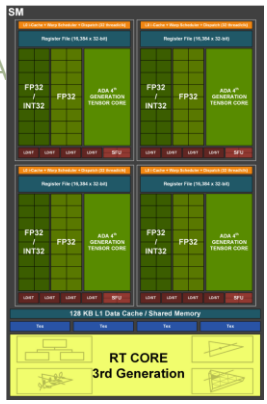
Ada AD102 GPU z 12 jednostkami Graphics Processing Cluster (*GPC*) po 12 SM



ARCHITEKTURA ADA – RTX40X0

Układ zawiera:
- 12 jednostek Graphics Processing Cluster (GPC) po 12 SM
- 96 MB cache L2

Moduł SM zawiera:
- 2 rdzenie podwójnej precyzji
- 128 rdzeni pojedynczej precyzji
- w tym 64 rdzenie całkowite
- 4 jednostki specjalne,
- 16 jednostek odczytująco-zapisujących
- 4 rdzeni Tensor
- 1 jednostkę śledzenia promieni
- 128 KB pamięci współdzielonej/L1.



ARCHITEKTURA ADA– RTX40X0

Porównanie Turing, Ampere i Ada GPU

Graphics Card	GeForce RTX 2080 Ti	GeForce RTX 3090 Ti	GeForce RTX 4090
CUDA Cores	4352	10752	16384
GPCs	6	7	11
TPCs	34	42	64
SMs	68	84	128
Tensor Cores	544 (2nd Gen)	336 (3rd Gen)	512 (4th Gen)
RT Cores	68 (1st Gen)	84 (2nd Gen)	128 (3rd Gen)

ARCHITEKTURA HOPPER

Hopper H100 GPU z 8 jednostkami Graphics Processing Cluster (*GPC*) po 18 SM

ARCHITEKTURA HOPPER

Układ zawiera:
- 8 jednostek Graphics Processing Cluster (GPC) po 18 SM
- 60 MB cache L2

Moduł SM zawiera:
- 64 rdzenie podwójnej precyzji
- 128 rdzeni pojedynczej precyzji
- 64 rdzenie całkowite
- 4 jednostki specjalne,
- 16 jednostek odczytująco-zapisujących
- 4 rdzenie Tensor gen. 4
- 256 KB pamięci współdzielonej/L1.

ARCHITEKTURA HOPPER

Porównanie architektur Ampere i Hopper dla centrów danych

GPU Features	NVIDIA A100	NVIDIA H100 SXM5	NVIDIA H100 PCIe
GPU Architecture	NVIDIA Ampere	NVIDIA Hopper	NVIDIA Hopper
GPU Board Form Factor	SXM4	SXM5	PCIe Gen 5
SMs	108	132	114
TPCs	54	66	57
FP32 Cores / SM	64	128	128
FP32 Cores / GPU	6912	16896	14592
FP64 Cores / SM (excl. Tensor)	32	64	64
FP64 Cores / GPU (excl. Tensor)	3456	8448	7296
INT32 Cores / SM	64	64	64
INT32 Cores / GPU	6912	8448	7296
Tensor Cores / SM	4	4	4
Tensor Cores / GPU	432	528	456