RÉPUBLIQUE
FRANÇAISE
Liberté
Égalité
Fraternité

ENAC

Programmation CUDA

S. Puechmorel

2023

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

ENAC

Plan

L'évolution des cartes graphiques

Années 1980 : contrôleurs video

Ces circuits permettaient d'afficher sur un tube cathodique des informations stockées en mémoire. Ils fournissaient des fonctionnalités de base, essentiellement orientées autour de la gestion de la mémoire vidéo et de la génération des signaux de synchronisation.

Figure: Le contrôleur vidéo MC6845. [1]

[1] https://commons.wikimedia.org/w/index.php?curid=976920

L'évolution des cartes graphiques

Contrôleurs graphiques

Apparus vers la fin des années 1980, ils apportent des fonctionnalités graphiques, telles le tracé de segment, et gèrent une mémoire distincte de celle de l'unité centrale.

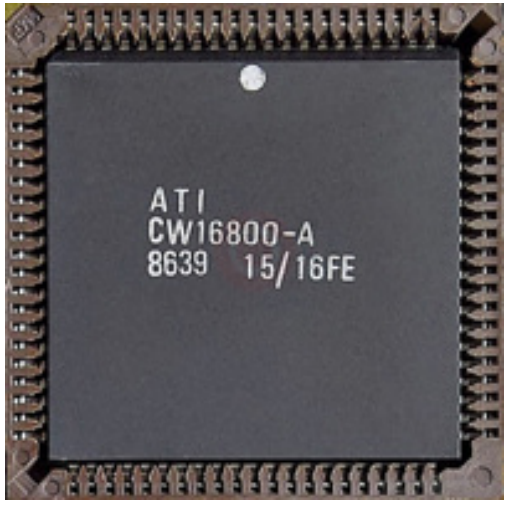

Figure: Contrôleurs graphiques.

RÉPUBLIQUE FRANÇAISE
Liberté Égalité Fraternité

ENAC

L'évolution des cartes graphiques

Les processeurs graphiques 3D.

Disponibles pour le grand public depuis le milieu des années 1990, ils incluent des fonctionnalités d'affichage en trois dimensions. Parallèlement, des bibliothèques logicielles font leur apparition (OpenGL, Direct3D). L'affichage d'une scène est réalisé à travers un pipeline graphique : transformation de sommets, projection, traçage.

Figure: Contrôleur 3D ATI Rage.

L'évolution des cartes graphiques

Les processeurs progammables

En 2001, la société NVIDIA introduit sur le marché la gamme de processeurs graphiques GeForce 3 qui permettent de programmer les étapes du pipeline graphique.

Figure: Contrôleur 3D programmable.

L'évolution des cartes graphiques

Le calcul

Fin 2006, NVIDIA lance la gamme GeForce 8 et l'environnement de développement CUDA qui permet d'exploiter la puissance de calcul des cartes graphiques pour des applications générales. Les performances théoriques sont impressionnantes : de l'ordre de celles obtenues avec un superordinateur, mais pour une enveloppe énergétique bien inférieure.

Figure: Carte graphique CUDA.

Synthèse de l'évolution des cartes NVIDIA

Année	Carte	Architecture	Cœurs NVIDIA CUDA	RAM	Puissance
1995	NV1	dizaines de micromètres (µm)	?	2-4 Mo	2 Watts
...	...	...	...	...	...
2017	GTX 1080 Ti	Volta 16 nm	3584	11 Go	257 watts
2019	GTX 2080 Ti	Turing 12 nm	4352	11 Go	290 Watts
2020	RTX 3090	Ampere 8 nm	10 496	24 Go	350 Watts
2022	RTX 4090	Ada Lovelace 4 nm	18 000	24 Go	450-600 Watts

Les multiprocesseurs de flux (SM)

- Héritier du pipeline graphique, le multiprocesseur de flux ("Streaming multiprocessor", SM) est une entité de traitement comportant des séquenceurs, plusieurs unités de traitement numérique et une mémoire locale.
- Un processeur graphique (GPU) regroupe plusieurs multiprocesseurs.
- Les multiprocesseurs exécutent des blocs de processus de façon **indépendante** et peuvent accéder à une mémoire partagée.
- Pour un développeur sur une architecture conventionnelle, un multiprocesseur s'apparente à un cœur de calcul vectoriel.

Architecture matérielle

Les multiprocesseurs de flux (SM)

- À l'intérieur d'un multiprocesseur, les processus s'exécutent de façon concurrente, mais peuvent communiquer via la mémoire locale ou être synchronisés.
- Les processus sont regroupés par blocs, appelés "warps" (chaînes), qui se voient affecter le même séquenceur d'instructions.
- Le modèle associé est dit "SIMT" pour "Single Instruction Multiple Thread".

RÉPUBLIQUE
FRANÇAISE
Liberté
Égalité
Fraternité

ENAC

Les unités de calcul

- Le nombre d'unités de calcul par multiprocesseur dépend des générations de cartes. Pour l'architecture Ampère, on trouve 64 ou 128 unités flottantes 32bits, 32 ou 2 unités flottantes 64bits, 64 unités de calcul entier sur 32 bits, 16 unités spéciales (fonctions transcendantes), 4 cœurs de calcul tensoriel et 4 ordonnanceurs.
- Un ordonnanceur est affecté à une chaîne. Tous les processus de la chaîne exécutent la même Instruction au même moment.
- En cas d'instruction conditionnelle, il peut y avoir divergence de code à l'intérieur d'une chaîne, ce qui se traduit par la mise en attente d'un ou plusieurs processus dont l'exécution se poursuivra après celle de la branche principale.

Architecture matérielle

L'ordonnancement des processus

- Le code à exécuter sur le GPU est appelé noyau ("kernel".)
- Le programmeur décide du nombre de processus affectés à un même noyau et les répartit en blocs.
- Un bloc sera pris en charge par un multiprocesseur libre.
- Dans un bloc, des chaînes de 32 processus identifiés par des entiers consécutifs sont constituées.
- Depuis l'architecture Volta, chaque processus possède ses propres compteur de programme et pile d'appel, ce qui permet un contrôle plus fin, en particulier en cas de divergence.

Architecture matérielle

La mémoire

- Chaque multiprocesseur possède une mémoire locale très rapide, pouvant être partagée entre les processus d'un même bloc. Elle est organisée en 32 banques pouvant être utilisées simultanément. Idéalement, chaque processus d'une chaîne accède à sa propre banque. La capacité de cette mémoire varie entre 64kB et 228kB selon les générations.
- Les multiprocesseurs partagent une mémoire globale, plus lente, mais en mesure de stocker beaucoup plus de données. Les cartes de dernière génération, comme la RX4090, embarquent 24Gb de RAM.
- L'architecture 9.0 introduit la notion de cluster de blocs et de mémoire partagée distribuée.

Les mémoires spécialisées

La mémoire de textures

- Les GPUs étant initialement conçus pour des applications de rendu graphique 3D, certaines mémoires dédiées sont présentes.
- La mémoire de textures est particulièrement intéressante lorsque l'on cherche à stocker des données bidimensionnelles que l'on souhaite ensuite interpoler.
- Cette mémoire, chargée par le CPU, ne peut être modifiée par un programme CUDA.

À retenir

Le GPU

- Un bloc est affecté à un multiprocesseur, les processus d'une même chaîne exécutent la même Instruction en parallèle.
- Il faut donc penser avant tout en termes de chaînes (32 processus).
- Un branchement dans un même chaîne entraîne l'inactivation temporaire de processus.

La mémoire

- Privilégier l'utilisation de la mémoire partagée, bien plus rapide que la mémoire globale.
- S'efforcer d'avoir des accès contigus pour les processus d'une même chaîne.
- Penser à utiliser la mémoire des textures si nécessaire.