Ιόνιο Πανεπιστήμιο – Τμήμα Πληροφορικής
Παράλληλος Προγραμματισμός
2024-25

Ιεραρχίες μνήμης σε GPU

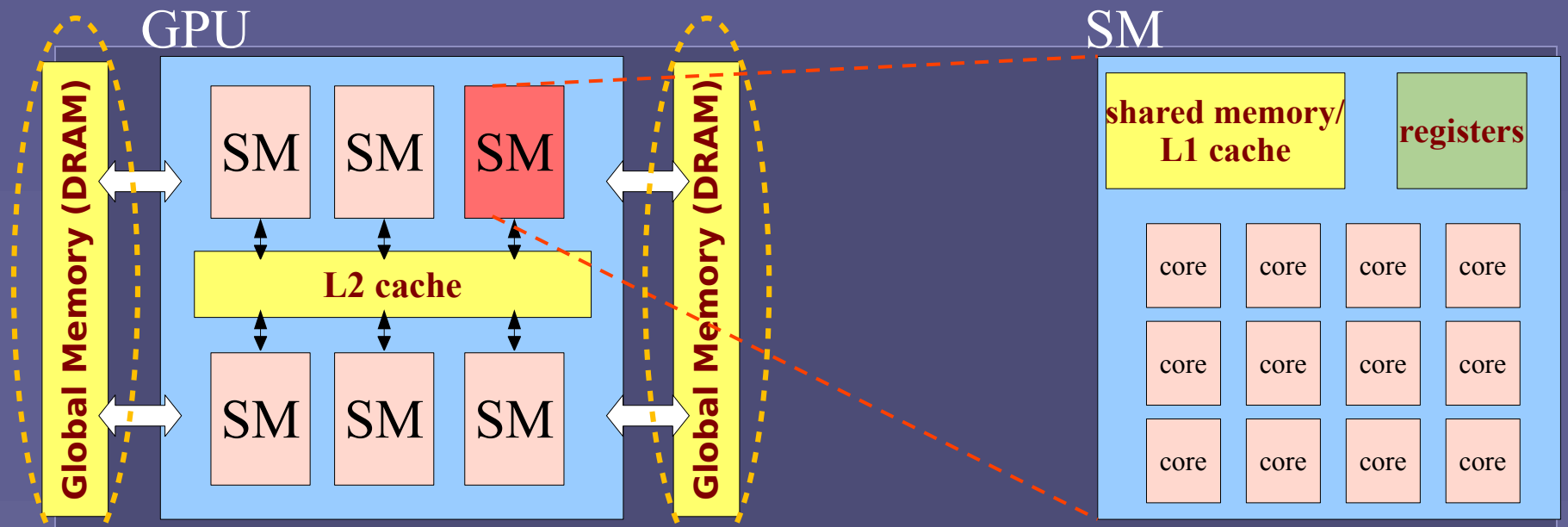
(Κύρια μνήμη, κρυφές μνήμες και shared memory)

https://mixstef.github.io/courses/parprog/

Το εργαστήριο του μαθήματος χρησιμοποιεί υπολογιστικούς πόρους AWS Cloud χρηματοδοτούμενους από το ΕΔΥΤΕ

Μ.Στεφανιδάκης

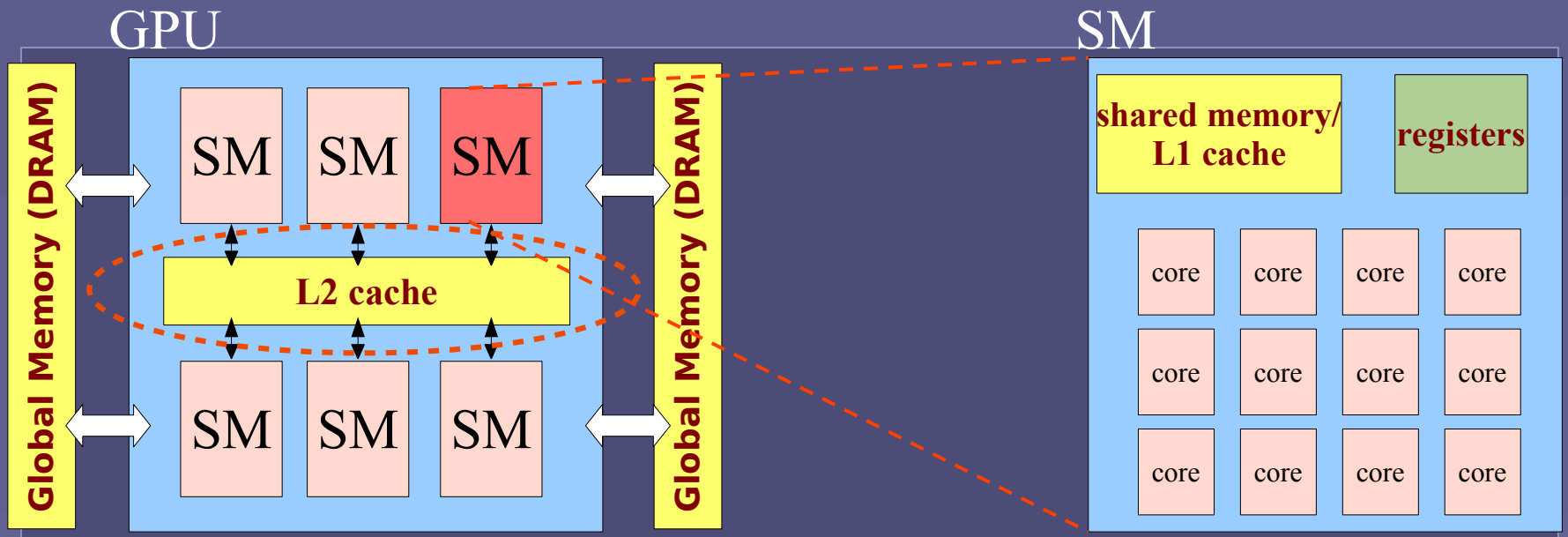
Τυπική ιεραρχία μνημών δεδομένων σε GPU



Global Memory (DRAM)

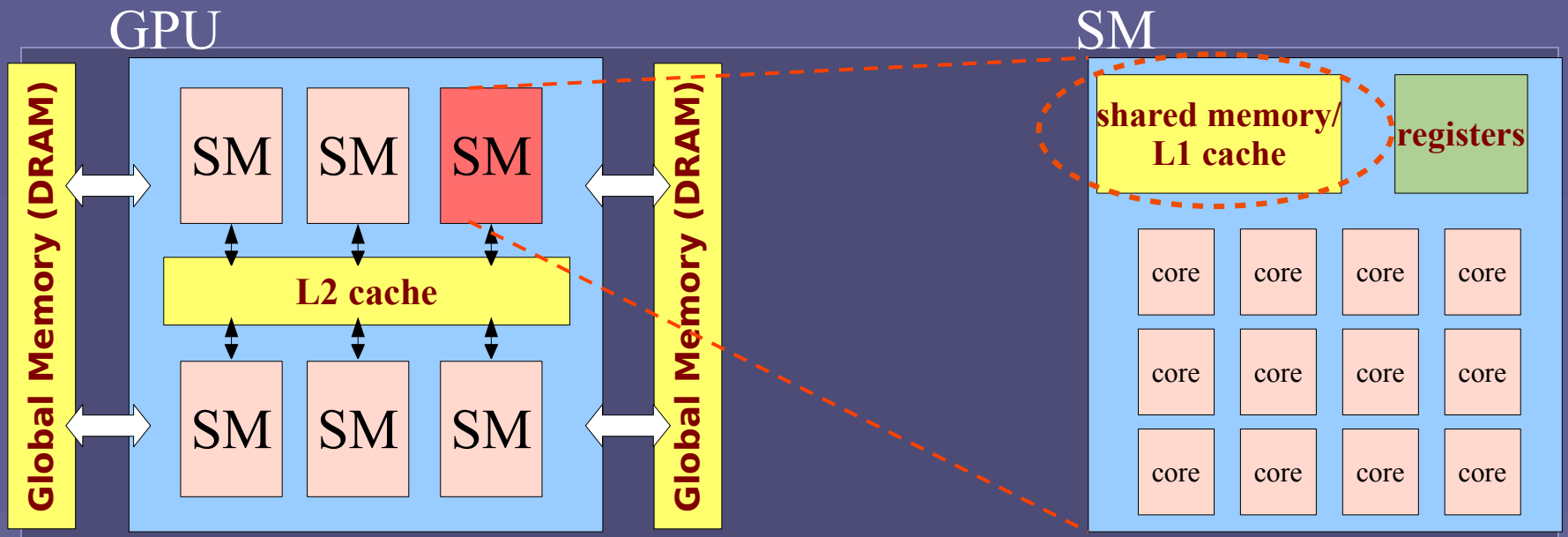
- Πρόσβαση: από κάθε thread και από host (CPU)
- Τυπική χωρητικότητα: σε GB, εκτός ή εντός chip GPU
- Χρόνος προσπέλασης: ~300 κύκλοι ρολογιού (GDDRx/HBM)
- Δυναμική δέσμευση χώρου από host (cudaMalloc, cudaFree)
  - Ή μεταβλητές με το keyword __device__
- Διάρκεια ζωής: CUDA Context (~διάρκεια εφαρμογής)

Τυπική ιεραρχία μνημών δεδομένων σε GPU



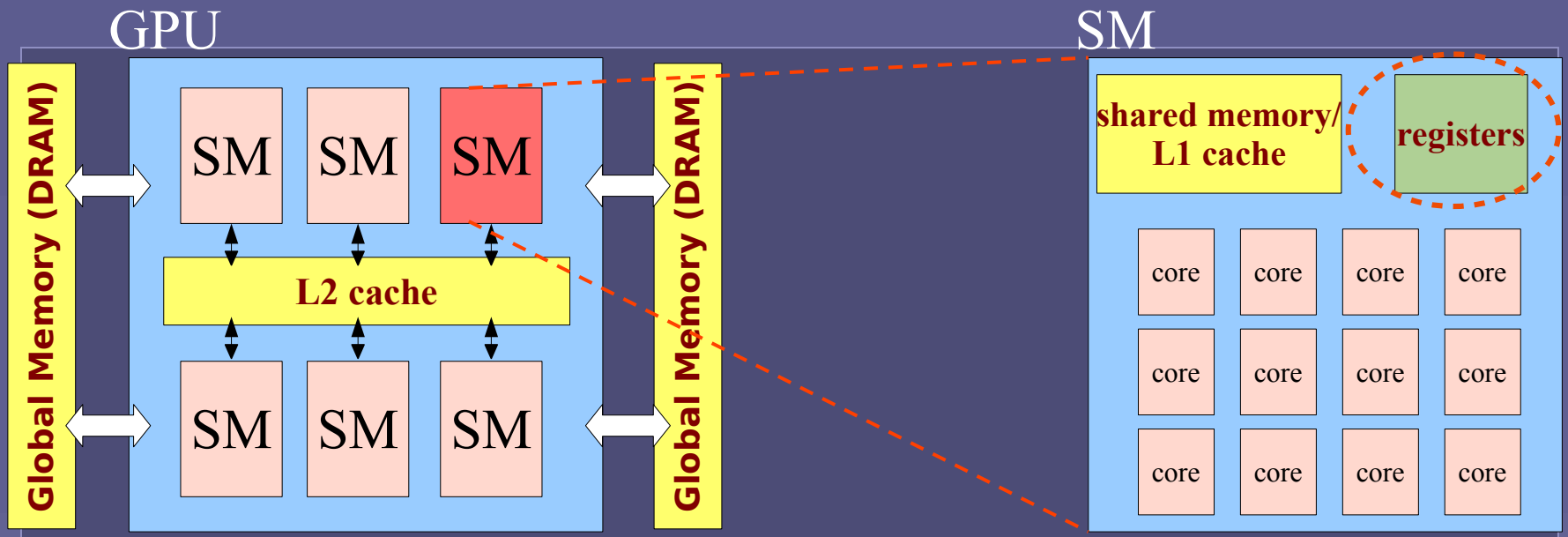
- L2 Cache
  - Πρόσβαση: από κάθε thread (διαφανώς, αυτόματη διαχείριση)
  - Ενιαία για δεδομένα/εντολές/σταθερές/υφές
  - Τυπική χωρητικότητα: μερικά MB, εντός chip GPU
  - Χρόνος προσπέλασης: ~200 κύκλοι ρολογιού

Τυπική ιεραρχία μνημών δεδομένων σε GPU



- L1 Data Cache
  - Πρόσβαση: από τα threads του ίδιου SM
  - Ο χώρος μοιράζεται μεταξύ L1, shared memory και, ανάλογα με την αρχιτεκτονική της GPU, μπορεί να φιλοξενεί και άλλους τύπους δεδομένων γραφικών (π.χ. υφές – textures)
  - Τυπική χωρητικότητα: 32 - 128 KB, μέσα σε κάθε SM
  - Χρόνος προσπέλασης: ~35 κύκλοι ρολογιού

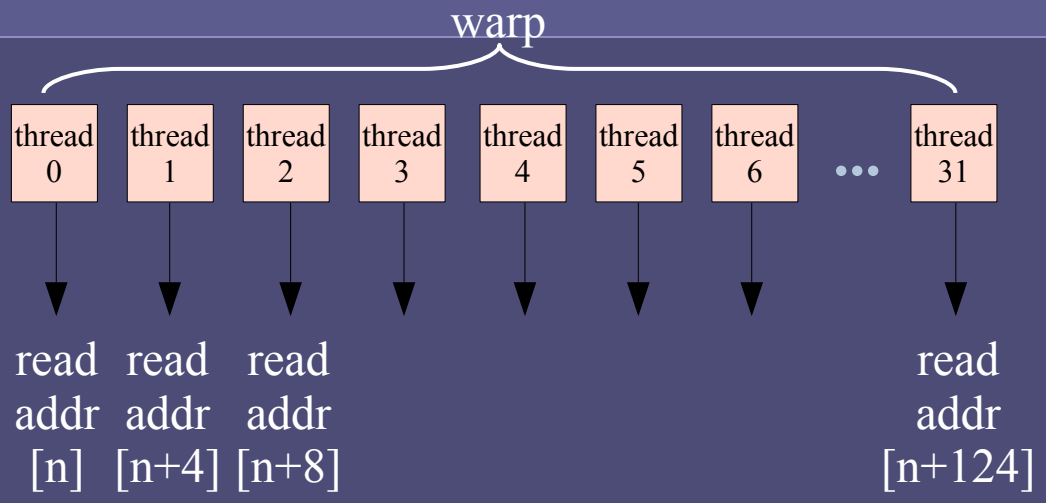
Τυπική ιεραρχία μνημών δεδομένων σε GPU



- Καταχωρητές (registers)
  - Πρόσβαση: κάθε thread έχει τους δικούς του καταχωρητές
  - Οι καταχωρητές ενός SM φιλοξενούν τους καταχωρητές κάθε thread για κάθε block που εκτελείται ταυτόχρονα στο SM
  - Τυπικό μέγεθος: 64K καταχωρητές των 32 bits
  - Οι καταχωρητές χρησιμοποιούνται όσο το δυνατόν περισσότερο από τον μεταγλωττιστή για τα δεδομένα
    - Ως ύστατη λύση (register spilling) → αποθήκευση στην μνήμη DRAM ("local memory" του κάθε thread)

Βέλτιστη προσπέλαση global memory

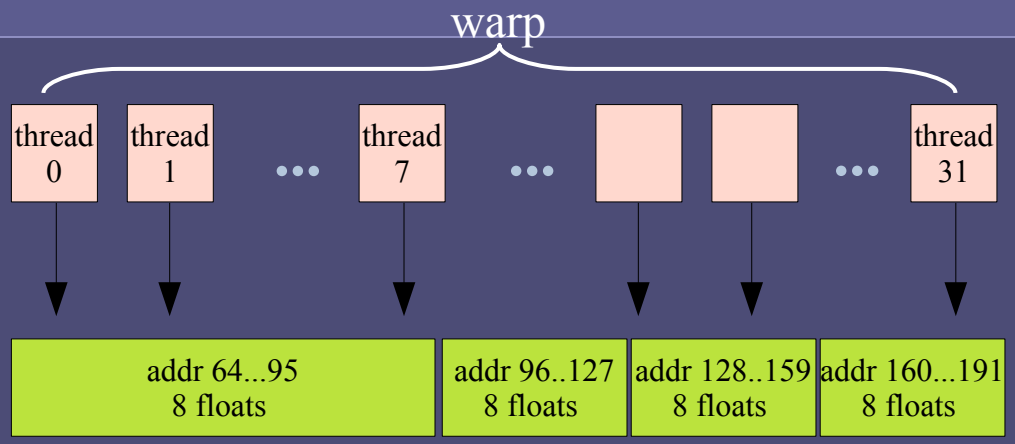
coalesced access



- Όταν τα threads ενός warp (ίδια εντολή) διαβάζουν ή γράφουν από/σε συνεχόμενες θέσεις μνήμης
  - Η GPU μπορεί να εκτελέσει την εντολή με τις ελάχιστες δυνατές προσπελάσεις στο global memory
  - Απαραίτητο για την αποδοτική λειτουργία σε παλαιότερες GPU όπου δεν υπήρχε κρυφή μνήμη
  - Οι μοντέρνες GPU έχουν κυφές μνήμες που αμβλύνουν σε μεγάλο βαθμό τις επιπτώσεις από την προσπέλαση μη συνεχόμενων διευθύνσεων

Ευθυγραμμισμένες προσπελάσεις

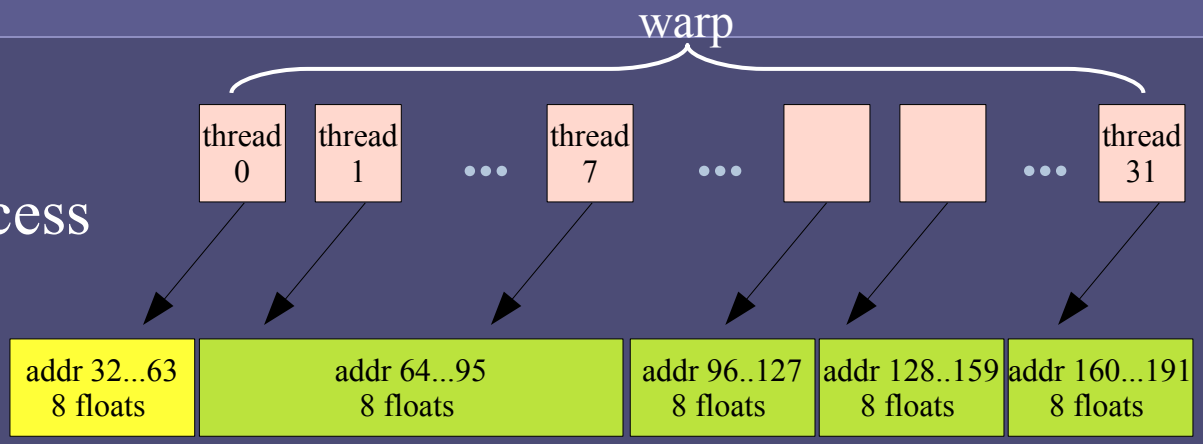
aligned access



- Οι μεταφορές γίνονται πάντοτε σε τμήματα των 32 bytes
  - Σε παλαιότερες GPU μπορεί να είναι διαφορετικά
  - Αν η προσπέλαση του πρώτου thread του warp γίνεται σε διεύθυνση πολλαπλάσιο του 32 τότε έχουμε ευθυγραμμισμένη προσπέλαση με τον ελάχιστο αριθμό μεταφερόμενων 32άδων
  - Αν π.χ. τα threads διαβάζουν ένα float (4 bytes) το καθένα, η μεταφορά για όλο το warp απαιτεί ανάγνωση 4*32 bytes /32 = 4 τμημάτων από τη μνήμη
  - Ομοίως αποδοτικό αν κάποια από τα threads δεν διαβάζουν
    - Ή αν διαβάζουν τις ίδιες διευθύνσεις με άλλη σειρά

Μη ευθυγραμμισμένες προσπελάσεις

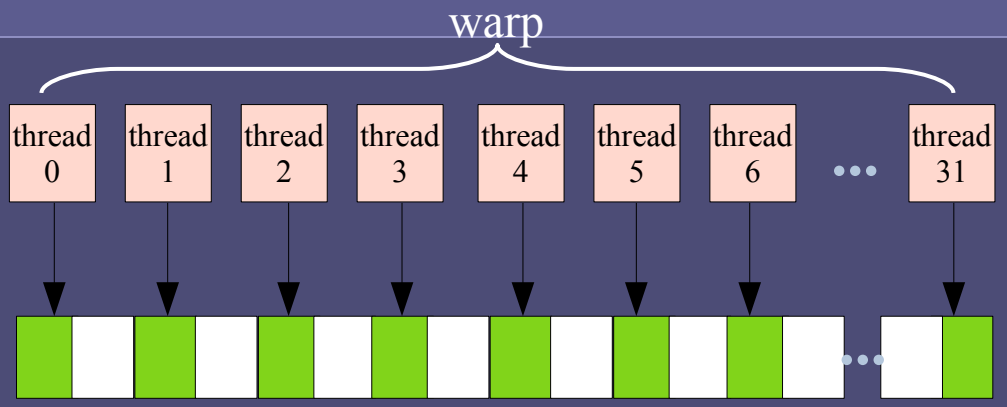
misaligned access



- Απαιτείται η μεταφορά ενός επιπλέον τμήματος 32 bytes
  - Το επιπλέον τμήμα δεν χρησιμοποιείται πλήρως από τα threads του warp
  - Αλλά θα βρίσκεται ήδη στην κρυφή μνήμη όταν ζητηθεί από «διπλανά» warps
  - Η ύπαρξη της κρυφή μνήμης στην περίπτωση αυτή ελαχιστοποιεί την επιβάρυνση της μεταφοράς του επιπλέον τμήματος

Επιζήμια προσπέλαση global memory

strided access



- Όταν δεν χρησιμοποιείται μέρος των μεταφερόμενων δεδομένων
  - Η κατάσταση χειροτερεύει όσο περισσότερο απέχουν μεταξύ τους οι προσπελάσεις γειτονικών threads
    - Όσο ο «διασκελισμός» (stride) μεγαλώνει
  - Το σχήμα αυτό προσπέλασης πρέπει να αποφεύγεται
    - Και αν απαιτείται από τον αλγόριθμο;

Shared Memory

- Μικρή γρήγορη μνήμη μέσα σε κάθε SM
  - Σε ιδανικές συνθήκες σχεδόν τόσο γρήγορη όσο οι καταχωρητές
  - Τυπικά μεγέθη 64KB έως 96KB
  - Μοιράζεται τον χώρο με L1 Cache/Texture memory
- Δέσμευση ατομικού χώρου ανά block
  - Για όλη τη διάρκεια εκτέλεσης του block
  - Διαθέσιμη σε όλα τα threads του block

Δήλωση χώρου από το shared memory

```
__global__ void sumReduction(float *a,float *psums) {
    __shared__ float buffer[THREADS];
```

- Στο παράδειγμα φαίνεται η δήλωση με στατικό μέγεθος, μέσα στον κώδικα του kernel
  - Υπάρχει και η δυνατότητα δήλωσης με δυναμικό μέγεθος (ως τρίτη παράμετρος στην εκκίνηση του kernel) αλλά δεν θα τη χρησιμοποιήσουμε εδώ

Άλλες χρήσεις Shared Memory

- Βελτίωση προσπέλασης κύριας μνήμης GPU
  - Το σύστημα μνήμης (global memory) της GPU επιτυγχάνει τη μέγιστη απόδοση όταν οι ζητούμενες λέξεις βρίσκονται σε γειτονικές (συνεχόμενες) διευθύνσεις (coalesced memory accesses)
    - Πάντως, οι μοντέρνες GPU έχουν ιεραρχίες μνήμης ικανές να αμβλύνουν σε μεγάλο βαθμό τις επιπτώσεις από την προσπέλαση μη συνεχόμενων διευθύνσεων
  - Εάν ένας αλγόριθμος απαιτεί την προσπέλαση μη συνεχόμενων διευθύνσεων (π.χ. striding σε προσπέλαση πινάκων) μπορεί να χρησιμοποιηθεί ο χώρος του shared memory για την πιο ευνοϊκή αναδιάταξη των δεδομένων
    - Το κόστος προσπέλασης του shared memory είναι πολύ μικρό, ακόμα και με τη χρήση μη συνεχόμενων διευθύνσεων

Συγκρούσεις προσπέλασης Shared Memory

- Η αρχιτεκτονική του shared memory επιτρέπει την ταυτόχρονη προσπέλαση από όλα τα threads ενός warp
  - 32 αναγνώσεις σε γειτονικές διευθύνσεις μπορούν να εκτελεστούν ταυτόχρονα
    - Το shared memory είναι χωρισμένο σε 32 μέρη (banks), τα οποία μπορούν να προσπελαστούν την ίδια στιγμή
  - Εάν οι ζητούμενες διευθύνσεις από τα threads πέφτουν στο ίδιο bank, τότε η προσπέλαση γίνεται σειριακά (bank conflict)
    - Αριθμός bank = διεύθυνση shared memory % 32
  - Για τη μέγιστη δυνατή απόδοση θα πρέπει να αποφεύγονται τα bank conflicts