Ιόνιο Πανεπιστήμιο – Τμήμα Πληροφορικής

Αρχιτεκτονική Υπολογιστών

2024-25

Εικονική Μνήμη

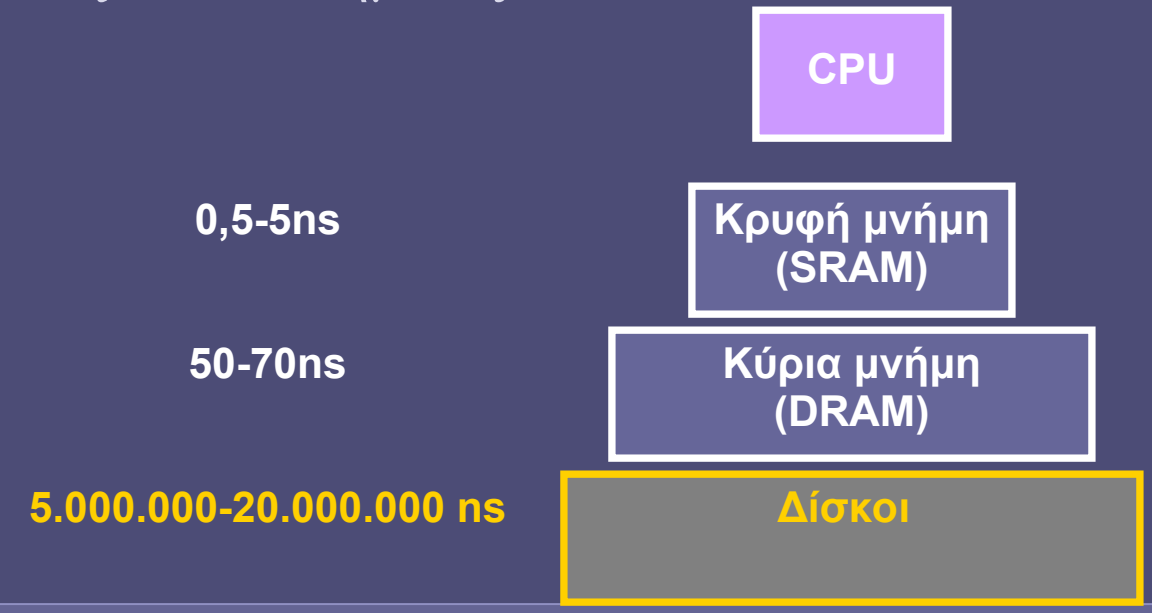
(και ο ρόλος της στην ιεραρχία μνήμης)

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

Μ.Στεφανιδάκης

Επεκτείνοντας την Ιεραρχία Μνήμης

- Σε μια ιεραρχία μνήμης κάθε υψηλότερο επίπεδο δρα ως «κρυφή μνήμη» για το αμέσως χαμηλότερο
  - Μπορούμε να θεωρήσουμε ότι η κύρια μνήμη λειτουργεί ως «κρυφή μνήμη» των δίσκων
    - Τα περιεχόμενα στην **κύρια μνήμη** είναι υποσύνολο εκείνων στους **δίσκους** του συστήματος



Εικονική μνήμη (virtual memory)

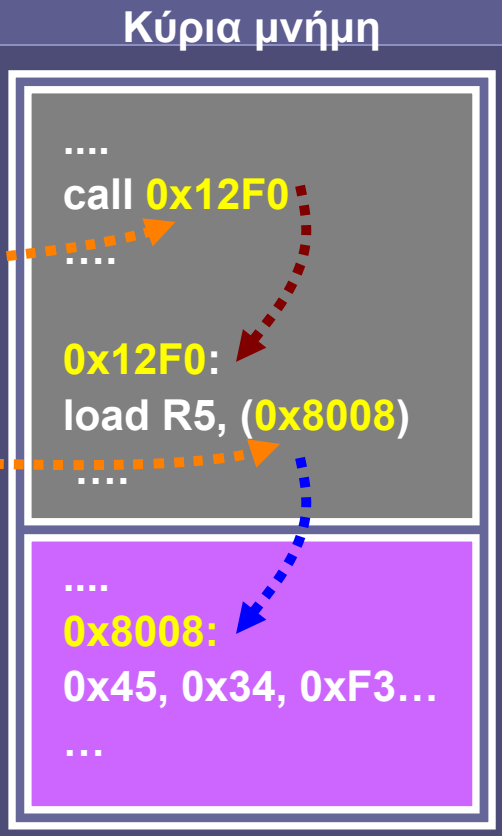
- Για ποιον λόγο εμφανίστηκε;
  - Στους πρώτους υπολογιστές το μέγεθος της κύριας μνήμης ήταν περιορισμένο
  - Ακόμα και στην περίπτωση του μονοπρογραμματισμού η κύρια μνήμη ήταν ανεπαρκής
  - Η ανάγκη για μεγαλύτερο μέγεθος κύριας μνήμης εντάθηκε με την εμφάνιση ΛΣ με υποστήριξη πολυπρογραμματισμού
    - Κάθε πρόγραμμα που εκτελείται πρέπει να έχει τουλάχιστον ένα μέρος του κώδικα και των δεδομένων του στην κύρια μνήμη
    - Έλλειψη μνήμης: αδυναμία ταυτόχρονης διατήρησης πολλών προγραμμάτων στην κύρια μνήμη

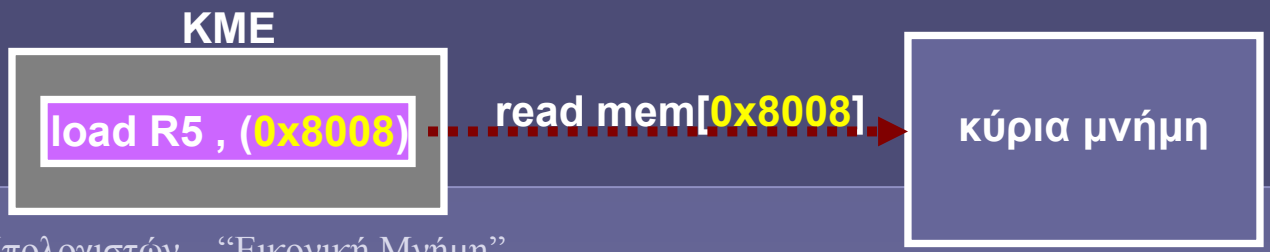
Εικονική μνήμη (virtual memory)

- Η λύση που δόθηκε: εικονική μνήμη
  - Μέρος των δεδομένων μπορεί να βρίσκεται στους δίσκους του συστήματος
  - Μεταφορά στην κύρια μνήμη όταν χρειαστεί
  - Πιθανότατα αντικαθιστώντας άλλα τμήματα δεδομένων
    - Τα τελευταία μεταφέρονται πίσω στους δίσκους
  - Μετάφραση εικονικών διευθύνσεων προγραμμάτων σε φυσικές διευθύνσεις
  - Το πρώτο σύστημα εικονικής μνήμης παρουσιάστηκε το 1962 (Atlas computer)
    - Core memory 16K λέξεων (εύρος λέξης = 48 bits)
    - Magnetic drums 96K λέξεων
    - Ενοποιημένη προσπέλαση στα δύο είδη μνήμης με τη βοήθεια του υλικού και του λειτουργικού συστήματος

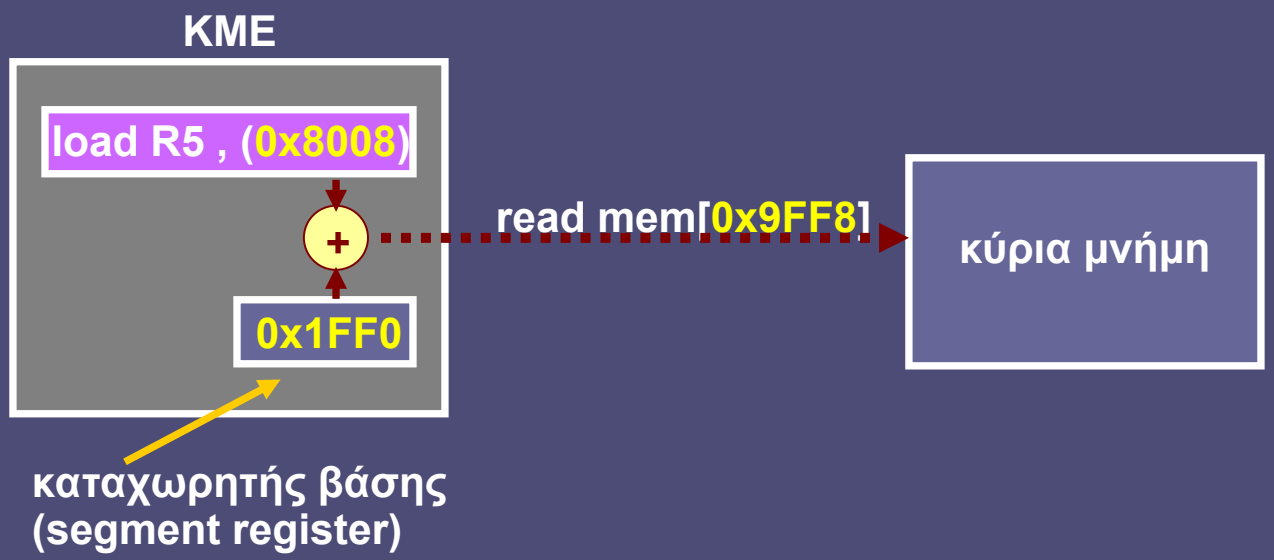
Χώρος διευθύνσεων προγράμματος

- Address Space
  - Εκτελούμενο πρόγραμμα στη μνήμη:
  - Διευθύνσεις κώδικα
    - Εντολές διακλάδωσης
  - Διευθύνσεις δεδομένων
    - Εντολές load-store
- Πριν την εικονική μνήμη:
  - Κάθε πρόγραμμα χρησιμοποιούσε φυσικές διευθύνσεις της κύριας μνήμης
  - Τι συνέβαινε αν το πρόγραμμα δεν φορτωνόταν πάντοτε στον ίδιο χώρο μνήμης;





Πολυπρογραμματισμός πριν την εικονική μνήμη



- Ειδικοί καταχωρητές βάσης για κώδικα και δεδομένα
  - Διευκόλυνση τοποθέτησης προγραμμάτων οπουδήποτε στη μνήμη
  - Αλλαγή τιμής καταχωρητών βάσης ανά πρόγραμμα
  - Δεν λύνεται το θέμα της ανεπαρκούς μνήμης
    - Αλλά είναι ένα πρώτο βήμα προς τις εικονικές διευθύνσεις

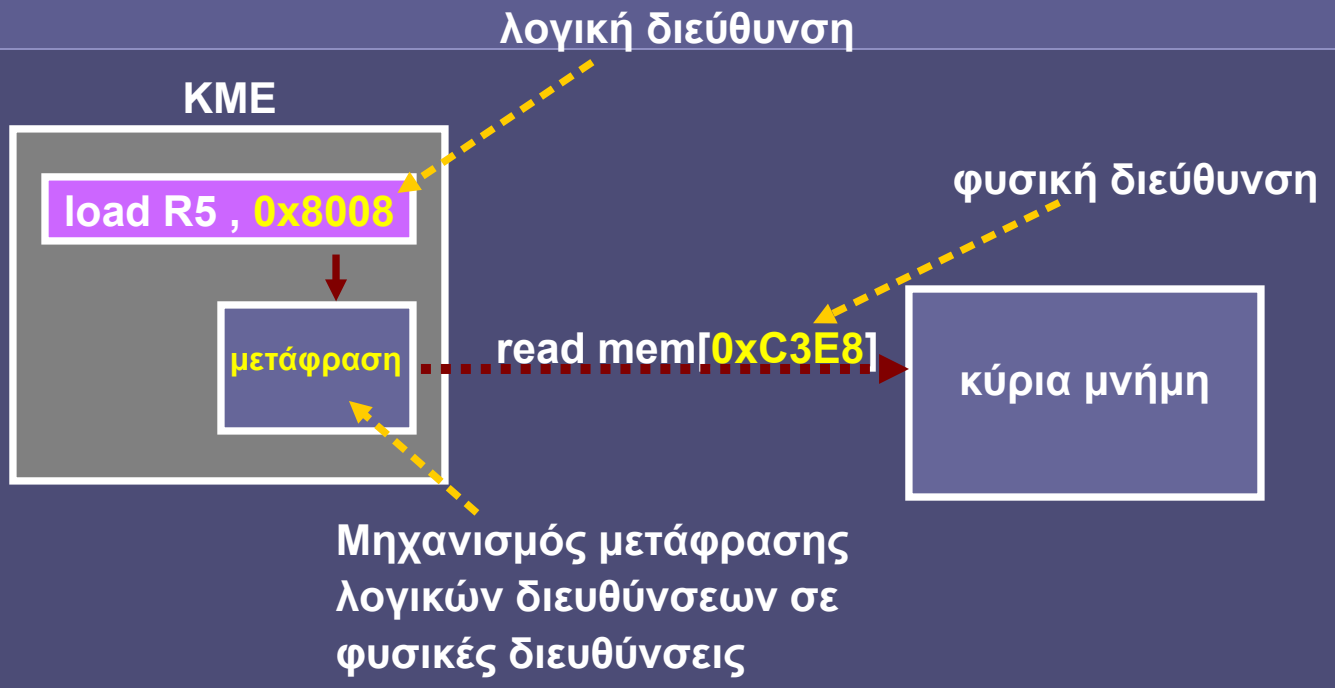
Μεταβαίνοντας σε εικονικές διευθύνσεις

- Το προηγούμενο σχήμα
  - Εισήγαγε την αποσύνδεση των λογικών (εικονικών) διευθύνσεων των προγραμμάτων από τις φυσικές διευθύνσεις κύριας μνήμης
  - Με μια απλή αντιστοιχία:

φυσική διεύθυνση = λογική διεύθυνση + καταχωρητής βάσης

  - Απαιτείται υποστήριξη από το υλικό (ΚΜΕ)
  - Το πρόγραμμα μπορεί να φορτωθεί σε οποιαδήποτε θέση μνήμης (relocation)
    - Δεν περιέχει αναφορές σε φυσικές διευθύνσεις
  - Εισάγεται η έννοια των ξεχωριστών χώρων διευθύνσεων (κώδικα, δεδομένων...) ανά πρόγραμμα

Λογικές (εικονικές) διευθύνσεις: η γενική ιδέα



- Για τη μετάφραση των λογικών διευθύνσεων σε φυσικές απαιτείται συνδυασμένη υποστήριξη από ΚΜΕ, κρυφή μνήμη και λειτουργικό σύστημα
- Η εικονική μνήμη δεν είναι «υποχρεωτική»: υπάρχουν «μικροί» επεξεργαστές (μικροελεγκτές – microcontrollers, MCUs) που δεν χρειάζεται να παρέχουν μηχανισμό εικονικής μνήμης

Σύστημα εικονικής μνήμης

- Ποιος ο ρόλος ενός συστήματος εικονικής μνήμης;
  - Ο αρχικός λόγος για τη χρήση της εικονικής μνήμης
    - Η δυνατότητα χρήσης εικονικής μνήμης πέρα από όση είναι πραγματικά διαθέσιμη και η διευκόλυνση του πολυπρογραμματισμού
  - Σημαντικότερο σήμερα, ιδίως στο cloud computing!
    - Η υλοποίηση μηχανισμών προνομίων προσπέλασης μνήμης από κάθε διεργασία και η προστασία των δεδομένων μεταξύ διαφορετικών διεργασιών
- Ποιος διαχειρίζεται την εικονική μνήμη;
  - Διαχείριση από το λειτουργικό σύστημα
  - Υποστήριξη από το υλικό (ΚΜΕ/κρυφή μνήμη)

Υλοποίηση εικονικής μνήμης

- Εκμετάλλευση της αρχής της τοπικότητας
  - Μερικά μέρη μόνο των προγραμμάτων είναι «ενεργά» κάθε στιγμή
    - Χρησιμοποιώντας (τη στιγμή αυτή) μικρό μέρος από τον συνολικό χώρο διευθύνσεών τους (κώδικα και δεδομένα)
  - Το λειτουργικό σύστημα αφαιρεί από την κύρια μνήμη και τοποθετεί στους δίσκους τα τμήματα μνήμης που δεν έχουν χρησιμοποιηθεί στο πρόσφατο παρελθόν
    - Συνεπώς θα χρειαστούν στο άμεσο μέλλον με μικρή πιθανότητα

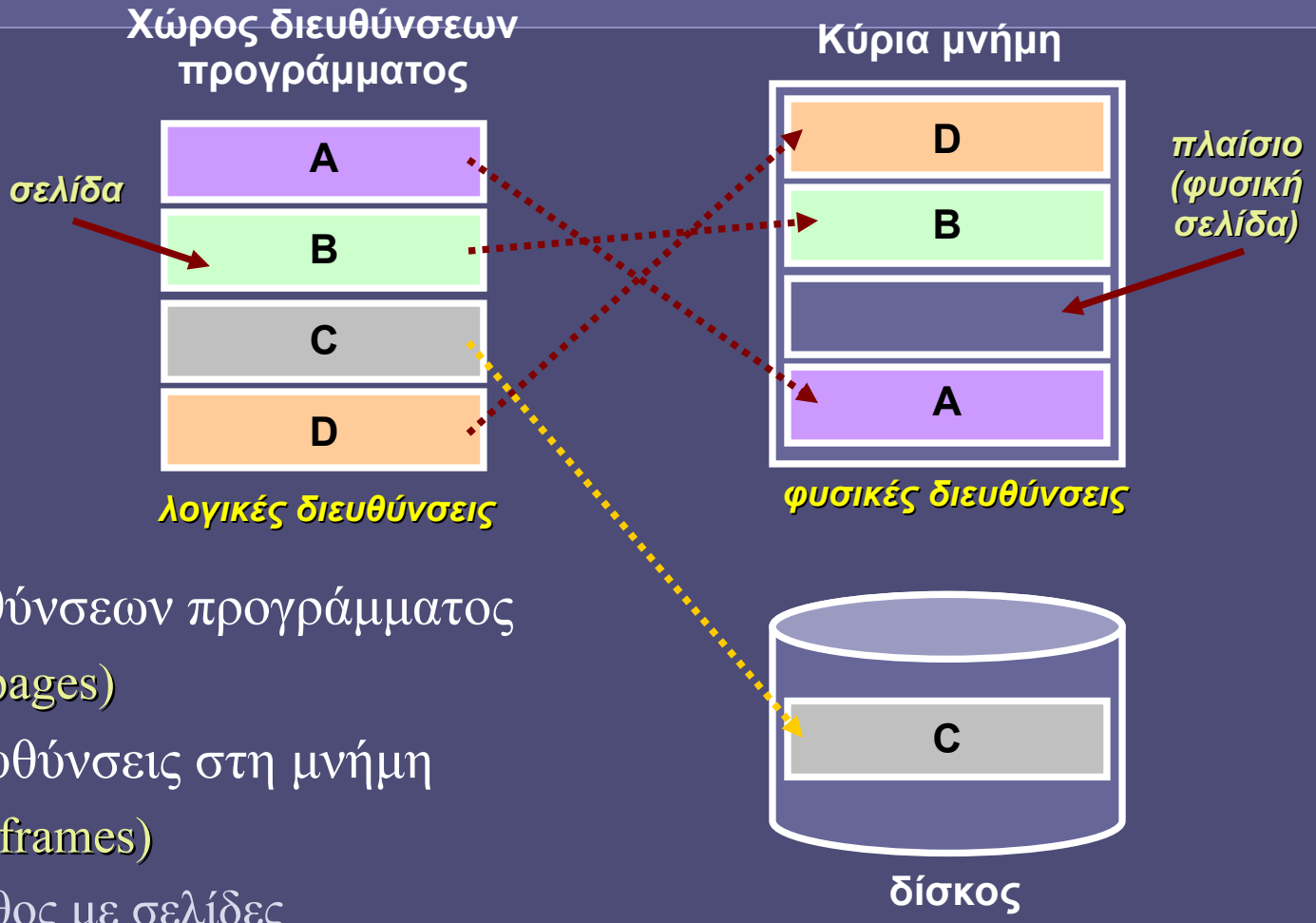
Υλοποίηση εικονικής μνήμης

- Η διαχείριση της εικονικής μνήμης έχει ομοιότητες με τη διαχείριση κρυφής-κύριας μνήμης
  - Χωρίς βέβαια τη μετάφραση από λογικές σε φυσικές διευθύνσεις
- Η διαφορά όμως στο κόστος προσπέλασης μεταξύ κύριας μνήμης και δίσκων είναι πολύ μεγαλύτερη (100.000 φορές και πλέον) από το κόστος προσπέλασης μεταξύ κρυφής και κύριας μνήμης
  - Συνεπώς η διαχείριση των μεταφορών δεδομένων από και προς τους δίσκους και η τοποθέτηση – αντικατάσταση στην κύρια μνήμη πρέπει να γίνεται με διαφορετικό τρόπο

Σελιδοποίηση (paging)

- Ο χώρος λογικών διευθύνσεων κάθε προγράμματος χωρίζεται σε σελίδες (pages)
  - Συγκεκριμένα μεγέθη σελίδας (4KB, 2MB, 1GB)
    - 4KB είναι το «κλασσικό μέγεθος», ταιριάζει με το μπλοκ μεταφοράς δεδομένων από/προς δίσκους
      - Μεγαλύτερο μέγεθος (=λιγότερες σελίδες ανά πρόγραμμα) → λιγότεροι πόροι για διαχείριση
      - Αλλά και αύξηση του αχρησιμοποίητου χώρου
- Η κύρια μνήμη (φυσικές διευθύνσεις) χωρίζεται σε φυσικές σελίδες (ή πλαίσια, frames) με μέγεθος ίσο με της σελίδας
  - Κάθε σελίδα τοποθετείται σε ένα ελεύθερο πλαίσιο
  - Ευκολία τοποθέτησης και αντικατάστασης σελίδων στην κύρια μνήμη

Βασικό σχήμα σελιδοποίησης



- Χώρος διευθύνσεων προγράμματος
  - Σελίδες (pages)
- Φυσικές διευθύνσεις στη μνήμη
  - Πλαίσια (frames)
  - Ίδιο μέγεθος με σελίδες
  - Κάποιες σελίδες μπορεί να βρίσκονται στον δίσκο

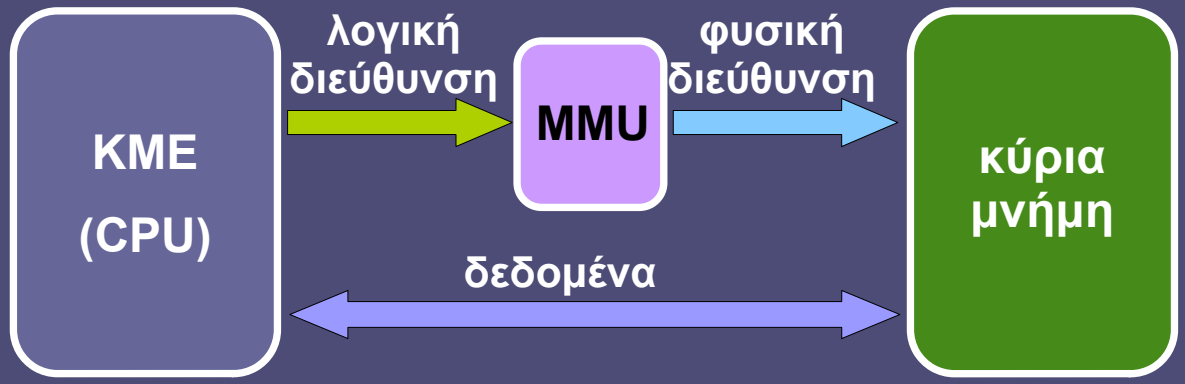
Σελιδοποίηση κατ'απαίτηση (demand paging)

- Οι σελίδες των προγραμμάτων (κώδικας-δεδομένα) βρίσκονται αρχικά μόνο στον δίσκο
  - Το ΛΣ τις σημειώνει ως "απούσες" από τη μνήμη
- Όταν η εκτελούμενη εντολή προσπελάσει δεδομένα από μια "απούσα" σελίδα, δημιουργείται ένα σφάλμα σελιδοποίησης (page fault) και το ΛΣ τη φορτώνει σε ένα πλαίσιο στη μνήμη
  - Ενδεχομένως εκτοπίζοντας πίσω στον δίσκο μια άλλη σελίδα από τη μνήμη
- Στη συνέχεια το λειτουργικό επιστρέφει τον έλεγχο στο πρόγραμμα που προκάλεσε το σφάλμα σελιδοποίησης
  - Η εντολή που διακόπηκε εκτελείται ξανά

Κρίσιμα σημεία στη σχεδίαση εικονικής μνήμης

- Η μείωση των page faults είναι επιβεβλημένη
  - Page faults: μεγάλο κόστος σε κύκλους αναμονής (1-10Μκύκλοι)
  - Οι σελίδες τοποθετούνται οπουδήποτε μέσα στη μνήμη
    - Σχήμα ανάλογο των fully-associative κρυφών μνημών
- Οι σελίδες πρέπει να έχουν ικανό μέγεθος για εξισορρόπηση του κόστους προσπέλασης του δίσκου
- Η διαχείριση της εικονικής μνήμης γίνεται από το λειτουργικό σύστημα
  - Δυνατότητα χρήσης πολυπλοκότερων αλγορίθμων για τοποθέτηση-αντικατάσταση σελίδων στη μνήμη

Εικονική μνήμη: το υλικό

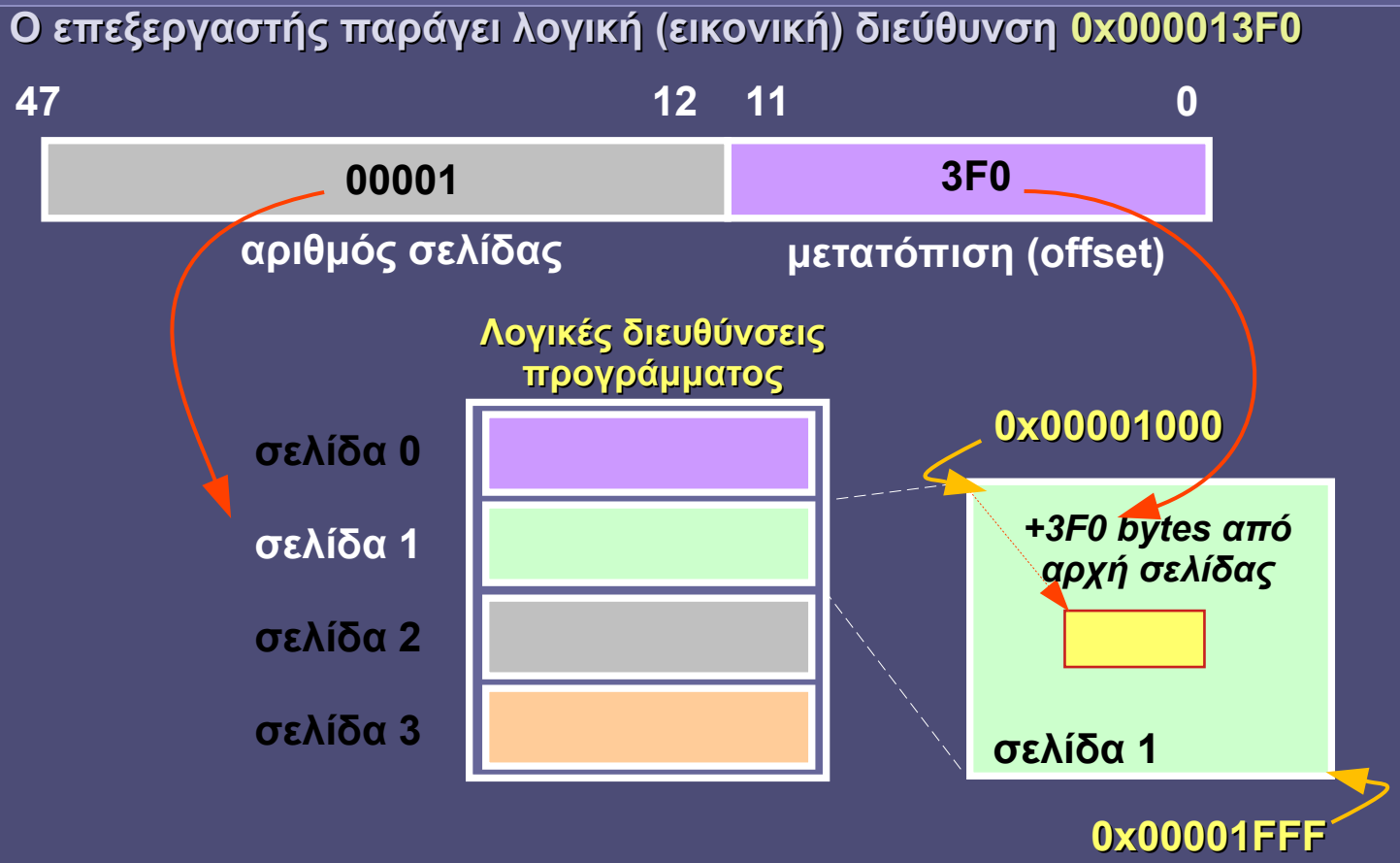


- Σύστημα διαχείρισης μνήμης
  - Memory management unit – MMU
  - Βρίσκεται μέσα στον επεξεργαστή
  - Μεταφράζει τις λογικές (εικονικές) διευθύνσεις κάθε προγράμματος σε φυσικές διευθύνσεις μνήμης
  - Από το διπλανό σχήμα απουσιάζει η κρυφή μνήμη, η οποία συνδέεται στενά με τον μηχανισμό εικονικής μνήμης

Σελίδες και λογικές (εικονικές) διευθύνσεις

- Ο επεξεργαστής μπορεί να παράγει έως και 2^N λογικές διευθύνσεις
- Η φυσική μνήμη μπορεί να έχει έως 2^M φυσικές διευθύνσεις
- Το μέγεθος σελίδας είναι 2^K bytes
- Τα N και M μπορούν να έχουν οποιαδήποτε σχέση
- Η μετάφραση γίνεται στον επεξεργαστή (MMU)

Παράδειγμα με μέγεθος σελίδας 4KB

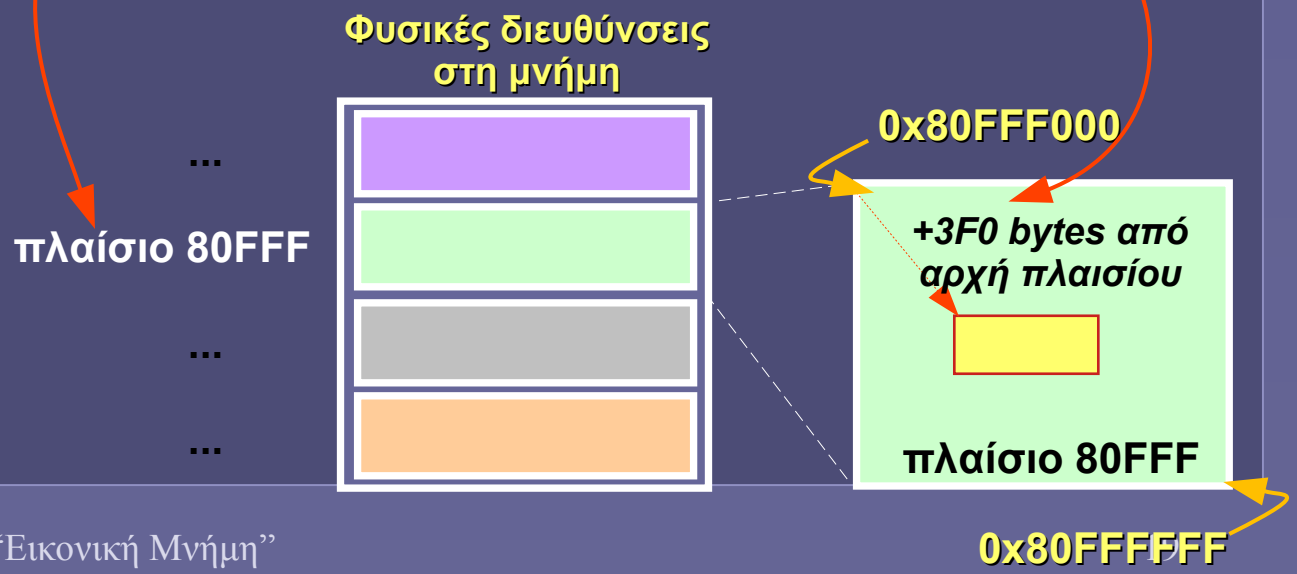


- Μέγεθος σελίδας 4KB (2^{12}) → K = 12
- Έστω ότι λογικές και φυσικές διευθύνσεις έχουν εύρος 48 bits → N = M = 48

Παράδειγμα (συνέχεια)

Ο επεξεργαστής παράγει λογική (εικονική) διεύθυνση: 0x000013F0

Στη μνήμη στέλνεται φυσική διεύθυνση: 0x80FFF3F0

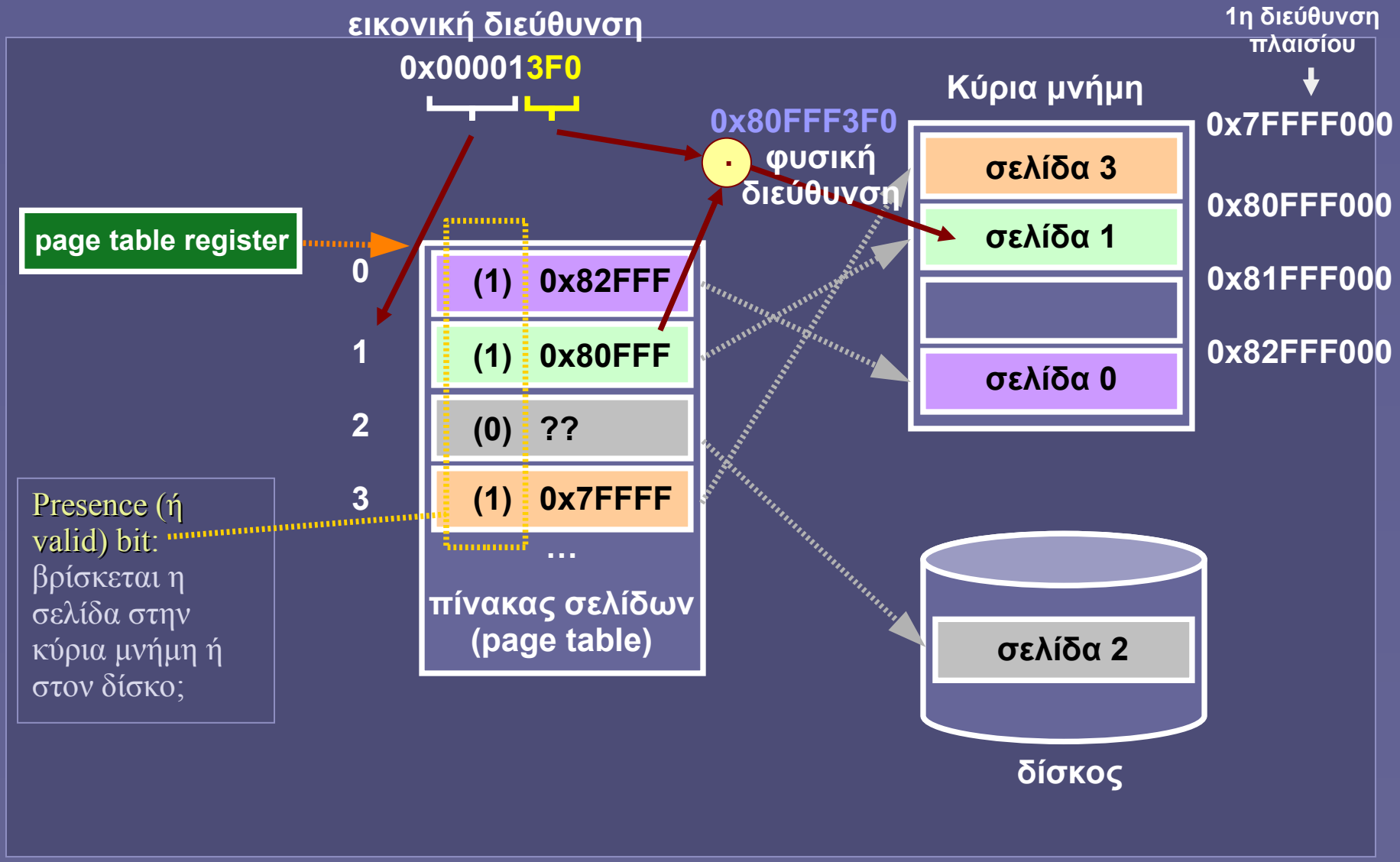


Έστω ότι η σελίδα 0x00001 βρίσκεται στη φυσική σελίδα (πλαίσιο) 0x80FFF

Πίνακας σελίδων

- Η μονάδα διαχείρισης μνήμης (MMU) χρειάζεται τις αντιστοιχίες σελίδων και πλαισίων για τη μετάφραση από λογικές σε φυσικές διευθύνσεις
  - Στο προηγούμενο παράδειγμα: πώς γνωρίζει η μονάδα MMU ότι η σελίδα 0x00001 βρίσκεται στη φυσική σελίδα (πλαίσιο) 0x80FFF;
- Οι αντιστοιχίες αυτές αποθηκεύονται σε έναν πίνακα σελίδων (page table)
  - Βρίσκεται στην κύρια μνήμη
    - Page table register: διεύθυνση αρχής του πίνακα
  - Έχει μέγεθος (θεωρητικά) ίσο με τον μέγιστο αριθμό σελίδων ενός προγράμματος – μια γραμμή πίνακα ανά σελίδα του προγράμματος
  - Διαχείριση από το λειτουργικό σύστημα

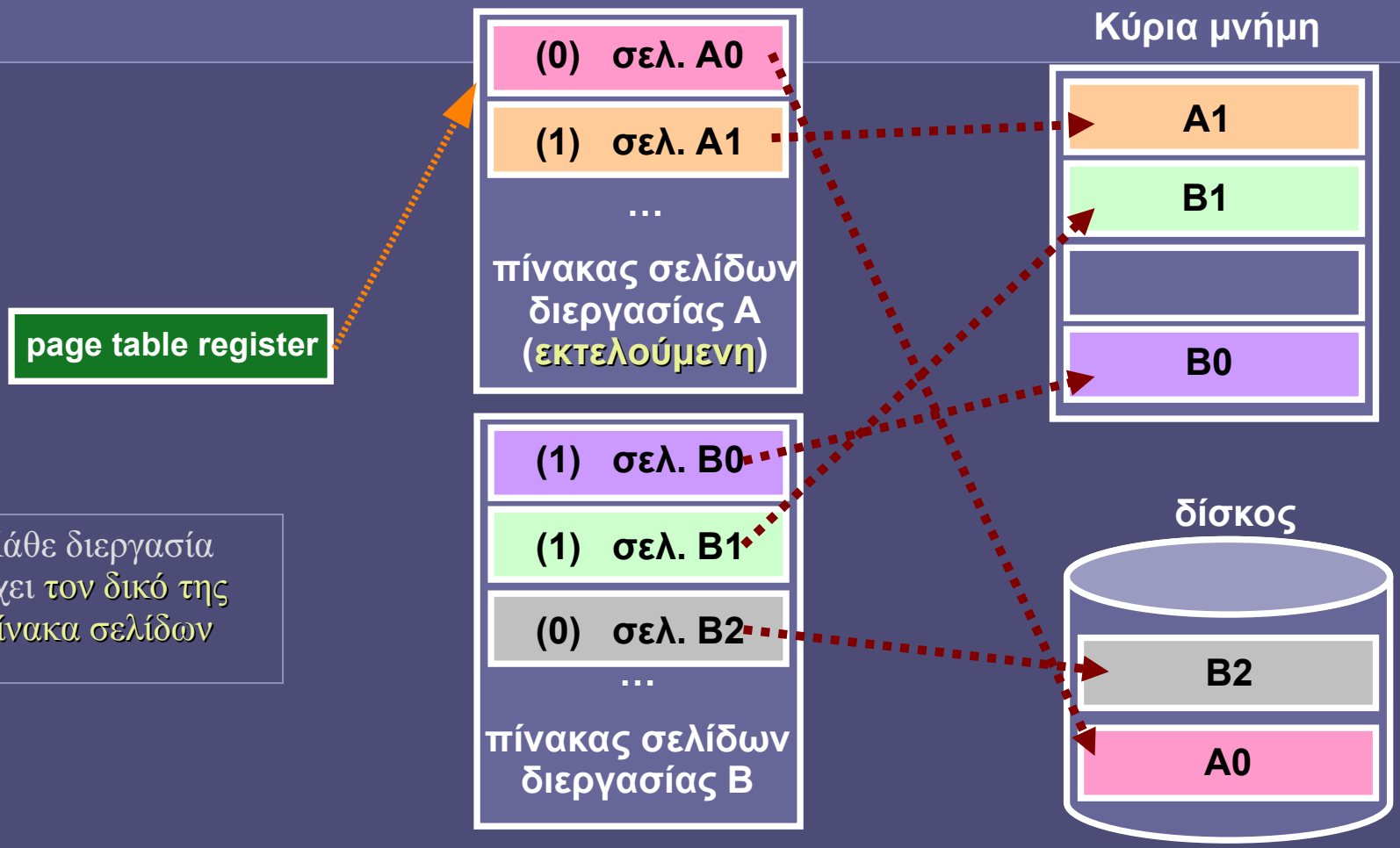
Μετάφραση λογικών διευθύνσεων



Πίνακας σελίδων (συνέχεια)

- Πρόσθετη πληροφορία σε κάθε γραμμή του πίνακα σελίδων
  - Presence (ή valid) bit
    - Βρίσκεται η σελίδα στη μνήμη ή στον δίσκο;
  - Dirty bit
    - Έχει γίνει εγγραφή δεδομένων στη σελίδα;
  - Accessed (ή reference ή use) bit
    - Ενημερώνεται από MMU σε κάθε προσπέλαση
    - Καταγράφεται σε τακτά διαστήματα από το ΛΣ για την υλοποίηση των αλγορίθμων LRU κατά την αντικατάσταση σελίδων
  - Μέγεθος σελίδας
    - Αν υποστηρίζονται εναλλακτικά μεγέθη
  - Δικαιώματα χρήσης σελίδας
    - read only/read write, user/supervisor …
    - Ανάλογα με αρχιτεκτονική επεξεργαστή

Πίνακες σελίδων και πολλαπλά προγράμματα



Κάθε διεργασία έχει τον δικό της πίνακα σελίδων

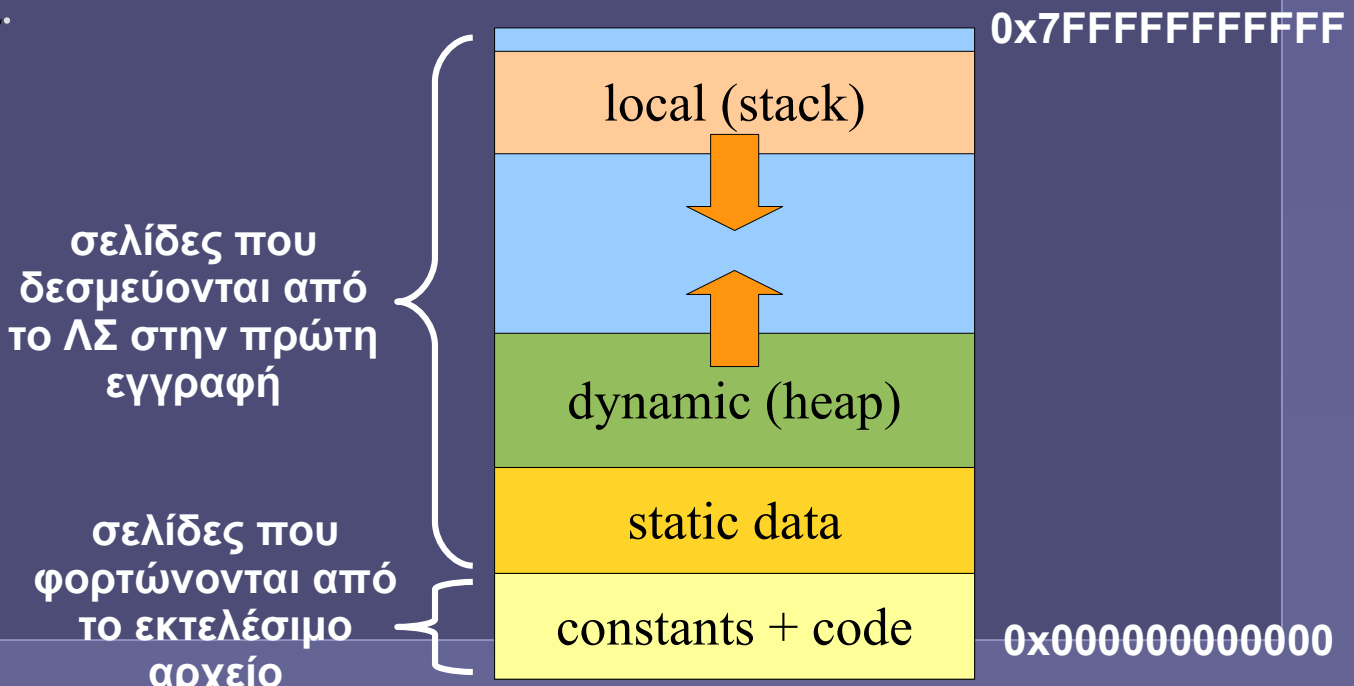
Εάν το λειτουργικό σύστημα φροντίσει να αντιστοιχίσει σωστά τις σελίδες κάθε προγράμματος σε διαφορετικές διευθύνσεις στην κύρια μνήμη, τότε κανένα πρόγραμμα δεν μπορεί να προσπελάσει δεδομένα άλλου προγράμματος

Εικονική Μνήμη και Προστασία Προσπέλασης

- Προστασία προσπέλασης σελίδων
  - Με διαφορετικούς πίνακες σελίδων ανά πρόγραμμα (διεργασία) είναι αδύνατη η προσπέλαση «ξένων» σελίδων
- User mode και Supervisor Mode (επίπεδα προνομίων)
  - Σε user mode δεν είναι δυνατή η προσπέλαση του πίνακα σελίδων και των αντίστοιχων καταχωρητών
    - Υπάρχουν αρχιτεκτονικές με περισσότερα από 2 επίπεδα προνομίων
- Ελεγχόμενη προσπέλαση κώδικα ΛΣ
  - System call exceptions
  - Μετάβαση σε κώδικα του λειτουργικού συστήματος με «ανύψωση» των προνομίων

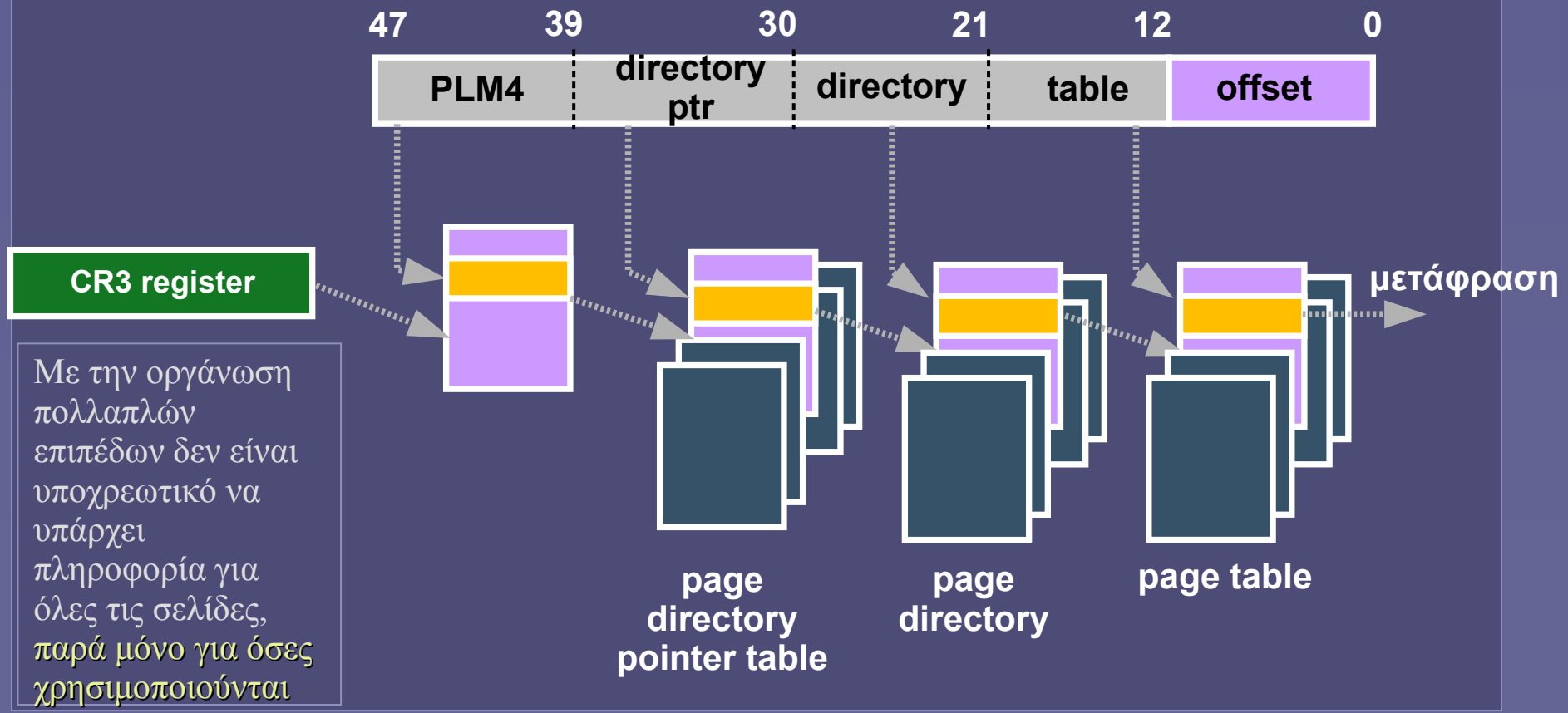
Μέγεθος πίνακα σελίδων

- Τα 64-bit προγράμματα μπορούν να «δουν» έναν πολύ μεγάλο χώρο λογικών διευθύνσεων
  - Π.χ. με 48 bits διεύθυνσης, 0...0x7FFFFFFFFFFF = 128TB
    - Δεν χρησιμοποιείται όλος ο χώρος διευθύνσεων
  - Δεν είναι δυνατό να έχουμε πίνακα σελίδων με γραμμές για κάθε πιθανή σελίδα.



Πίνακας σελίδων πολλαπλών επιπέδων

- Παράδειγμα: αρχιτεκτονική x86-64, σελίδα 4KB
  - 4 (ή και 5) επίπεδα πινάκων για 48-bit (ή 57-bit) διευθύνσεις



Με την οργάνωση πολλαπλών επιπέδων δεν είναι υποχρεωτικό να υπάρχει πληροφορία για όλες τις σελίδες, παρά μόνο για όσες χρησιμοποιούνται

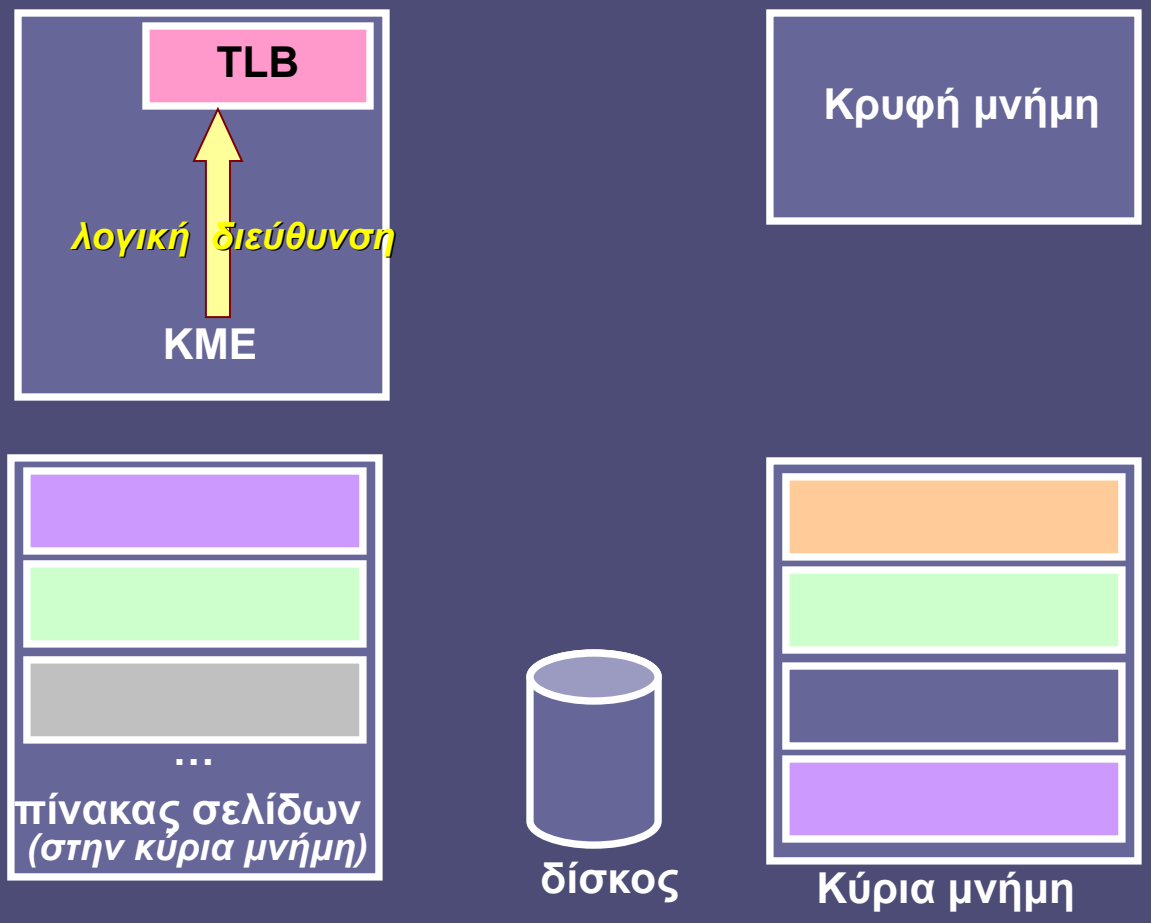
Translation-Lookaside Buffer

- Το πρόβλημα με τους πίνακες σελίδων
  - Βρίσκονται στην κύρια μνήμη
  - Θεωρητικά, για κάθε προσπέλαση μνήμης απαιτείται μια δεύτερη (τουλάχιστον) προσπέλαση στον πίνακα σελίδων, επίσης στη μνήμη
    - Μη αποδεκτή χρονική επιβάρυνση
- Translation-Lookaside Buffer (TLB)
  - Μικρή «κρυφή μνήμη» για πρόσφατες μεταφράσεις λογικών διευθύνσεων (στη μονάδα MMU)
    - Ο αριθμός σελίδας χρησιμοποιείται ως ετικέτα (tag)
    - Περιέχει μόνο έγκυρες σελίδες (όχι «απούσες»)
    - Διατηρείται συγχρονισμένη με πίνακες σελίδων από το ΛΣ
    - Κατά την εναλλαγή διεργασιών στην ΚΜΕ πρέπει να «αδειάζει»
      - Εναλλακτικά, χρήση ID διεργασίας μαζί με ετικέτα

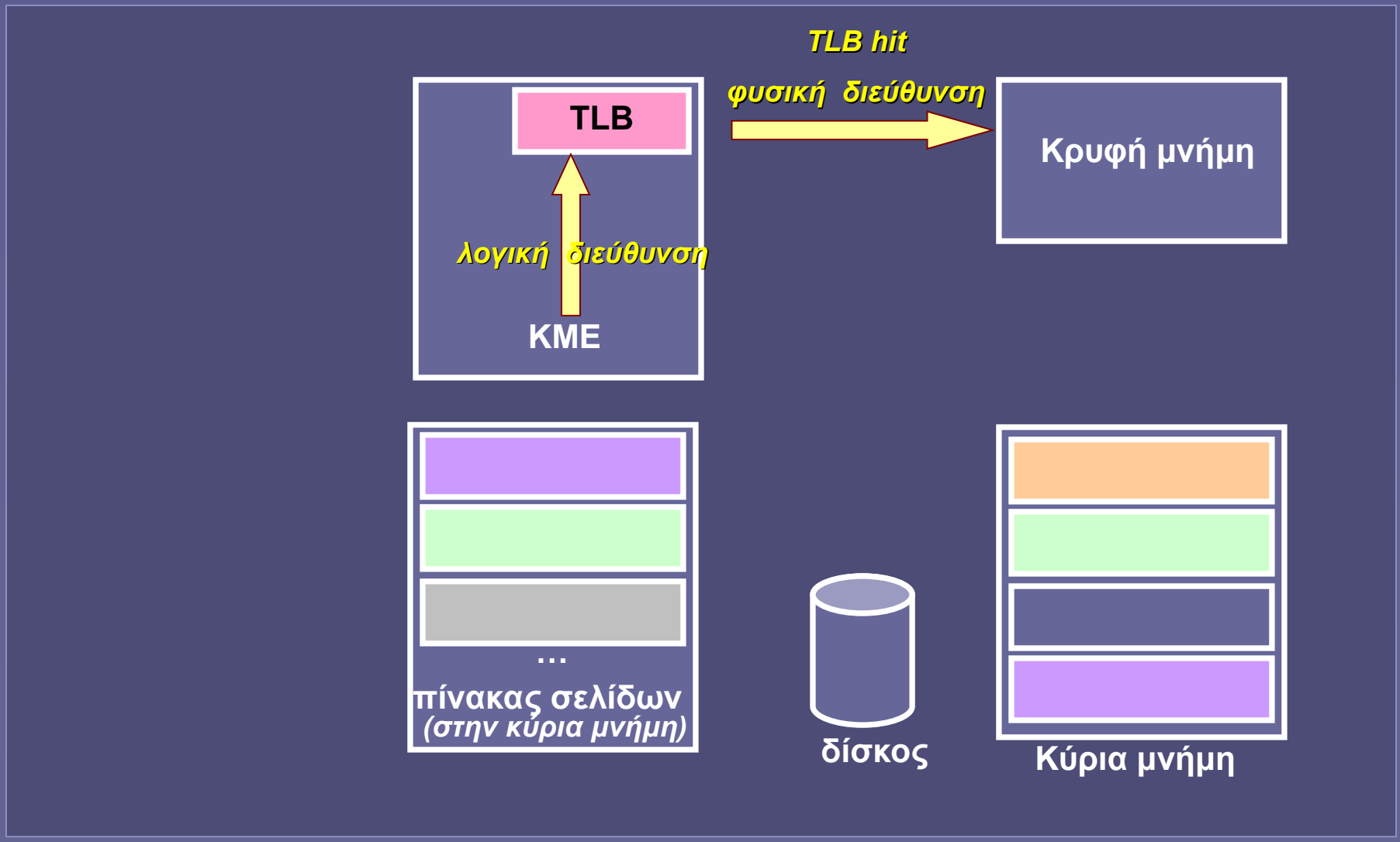
Translation-Lookaside Buffer (2)

- TLB miss
  - Προσπέλαση πίνακα σελίδων στη μνήμη και απόκτηση μετάφρασης που λείπει
    - Ενημέρωση από ΛΣ ή από την ίδια τη μονάδα MMU (“page table walking”)
- Χαρακτηριστικά TLB
  - Ξεχωριστά TLB για εντολές και δεδομένα
  - 16-512 θέσεις, 1-2 γραμμές του πίνακα σελίδων ανά θέση
    - Μικρό μέγεθος, τοπικότητα σελίδων πολύ μεγάλη
    - Συχνά πλήρως προσεταιριστικό
  - Προσπέλαση < 1 κύκλο ρολογιού
  - Παρατηρούμενο Miss rate: 0.01% - 1%
  - Όπως και στις πραγματικές κρυφές μνήμες, μπορεί να υπάρχει L1 και L2 TLB

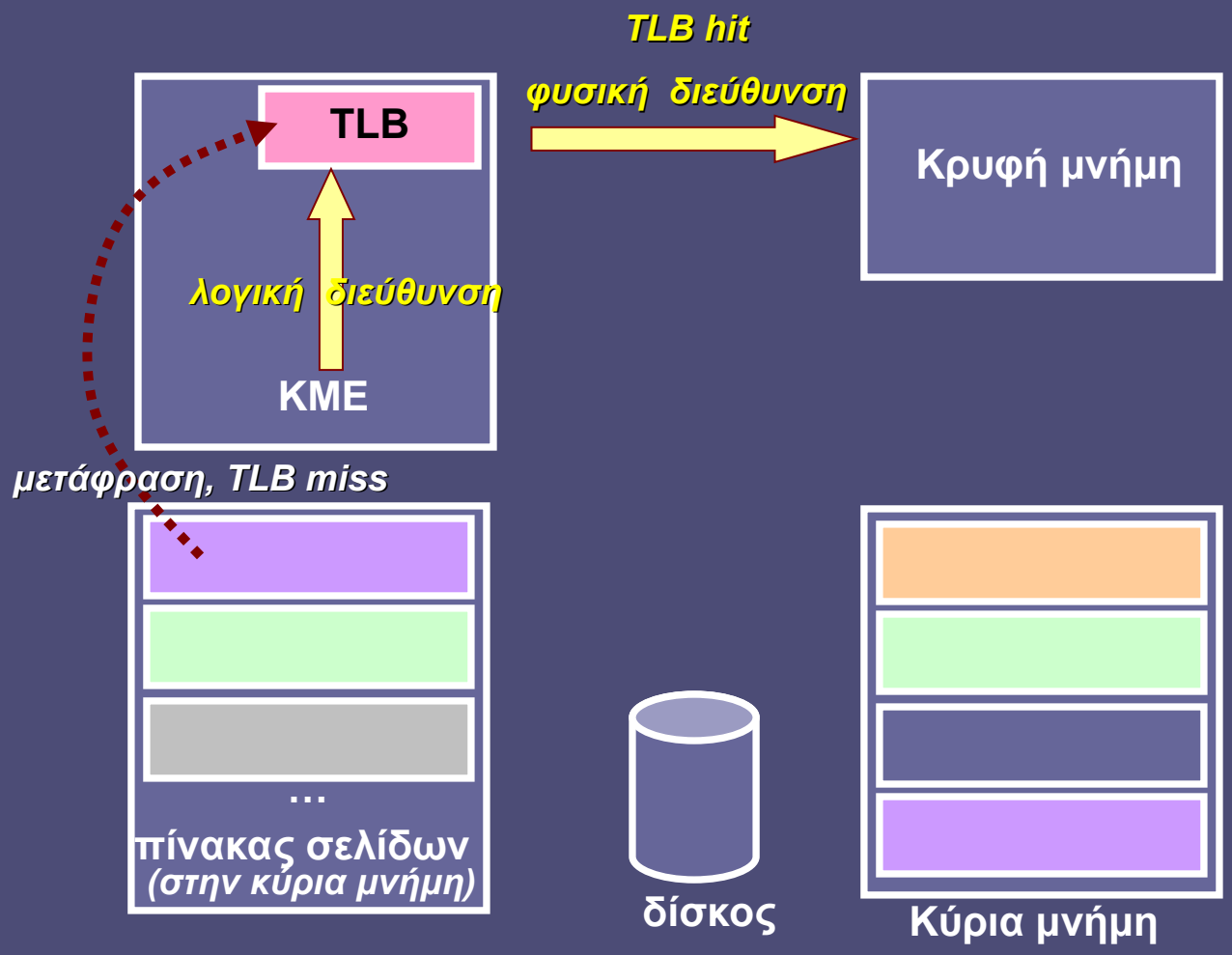
Προσπέλαση μνήμης: η συνολική εικόνα



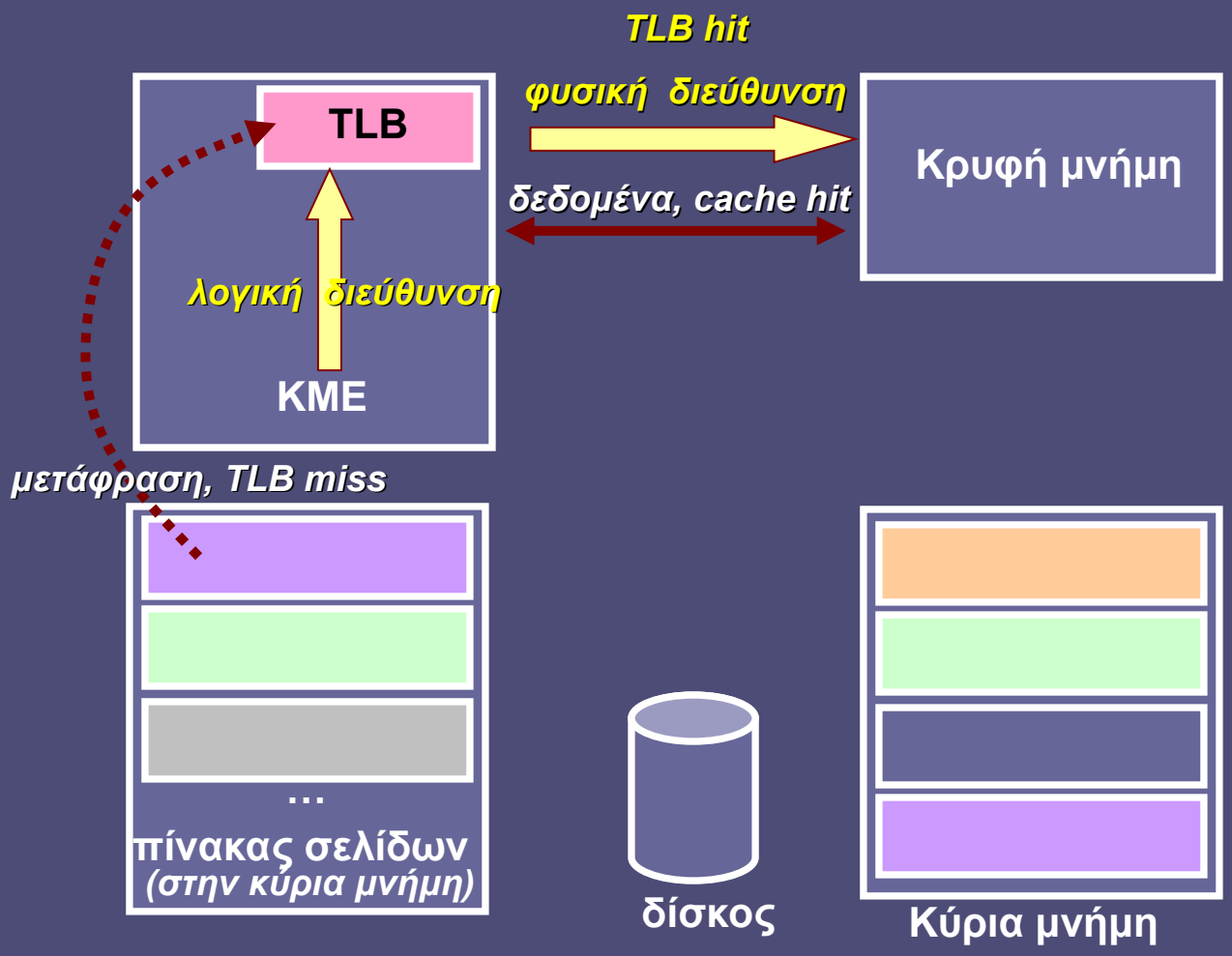
Προσπέλαση μνήμης: η συνολική εικόνα



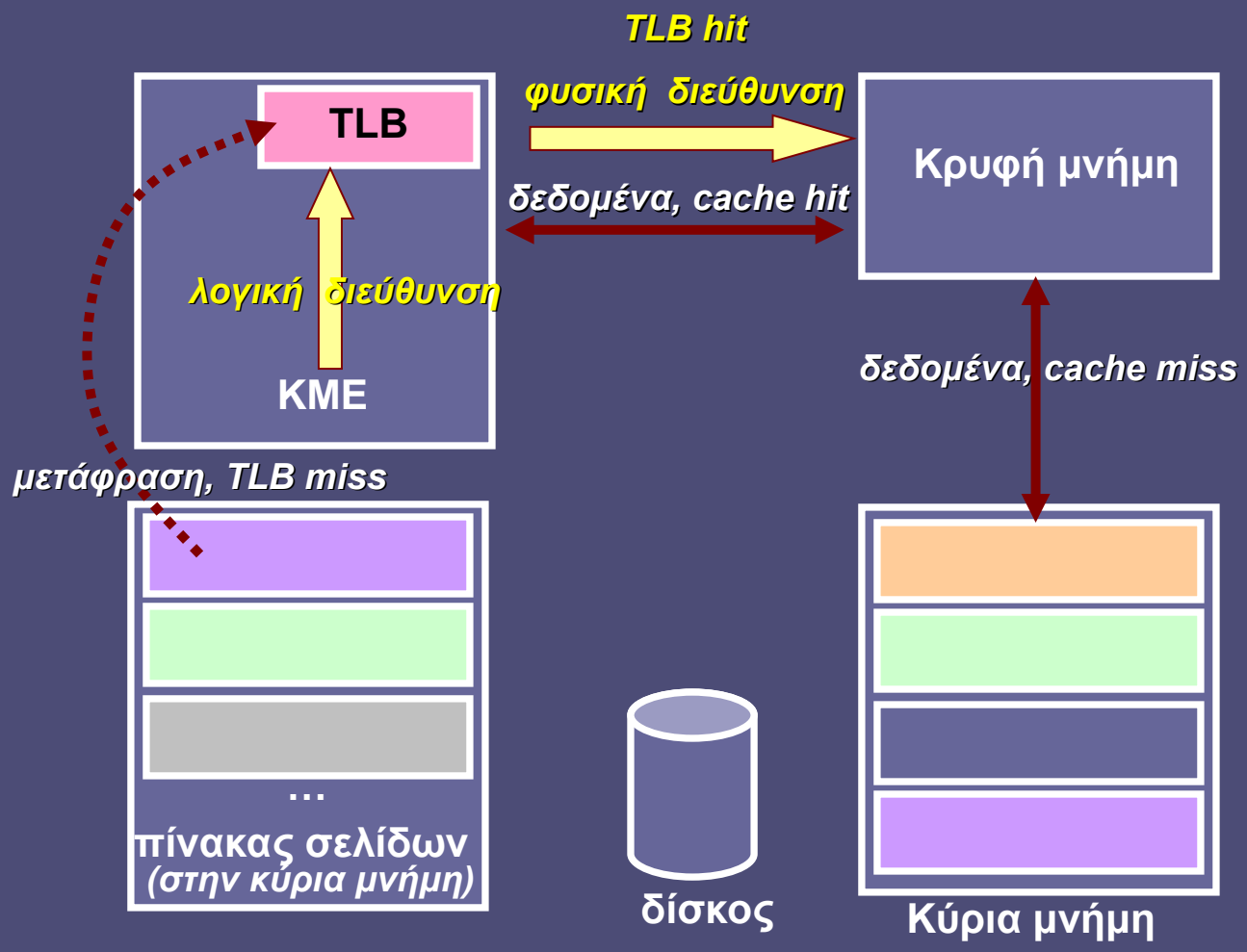
Προσπέλαση μνήμης: η συνολική εικόνα



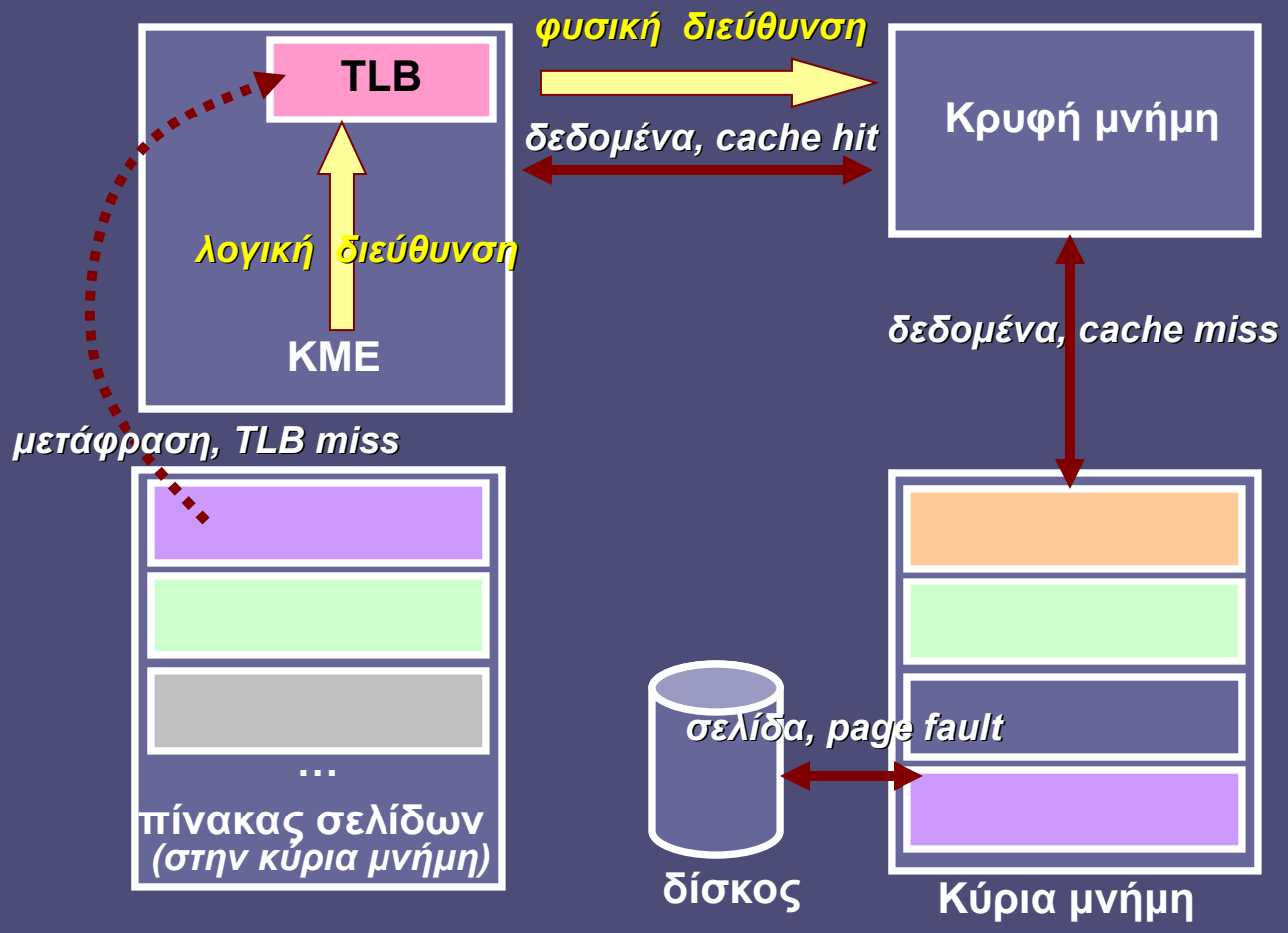
Προσπέλαση μνήμης: η συνολική εικόνα



Προσπέλαση μνήμης: η συνολική εικόνα

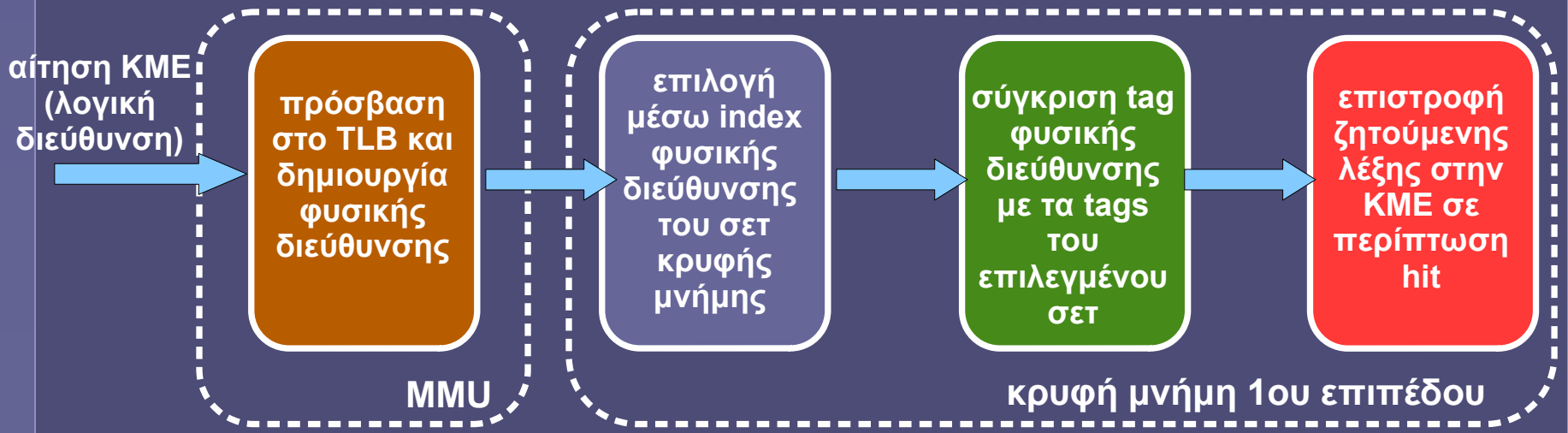


Προσπέλαση μνήμης: η συνολική εικόνα



TLB και κρυφή μνήμη 1ου επιπέδου

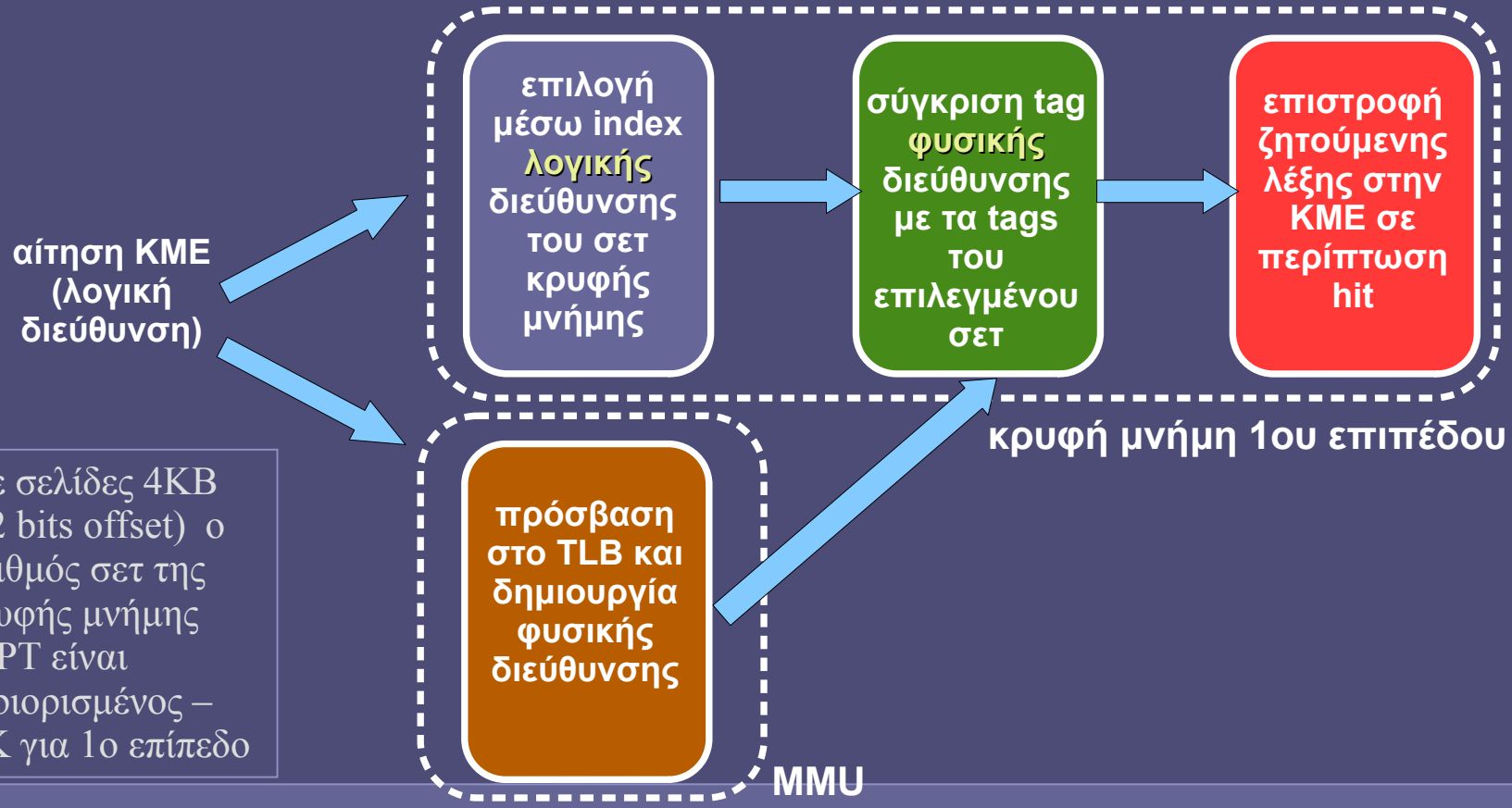
- Η κρυφή μνήμη 1ου επιπέδου (κοντά στην ΚΜΕ) πρέπει να είναι πολύ γρήγορη σε περίπτωση hit
  - Με την εισαγωγή της εικονικής μνήμης παρεμβάλλεται και το στάδιο της μετάφρασης από λογική σε φυσική διεύθυνση
  - Πώς μπορούμε να επιταχύνουμε τη διαδικασία;



Virtually Indexed Physically Tagged (VIPT) caches

Αν εξασφαλίσουμε ότι το index είναι μέσα στο offset της λογικής διεύθυνσης, δεν χρειάζεται μετάφραση για να το χρησιμοποιήσουμε

- Το offset παραμένει το ίδιο στη φυσική διεύθυνση



Με σελίδες 4KB (12 bits offset) ο αριθμός σετ της κρυφής μνήμης VIPT είναι περιορισμένος – OK για 1ο επίπεδο