Διαχείριση Μνήμης

Τρίτη 21-Μαρτίου-2023

1

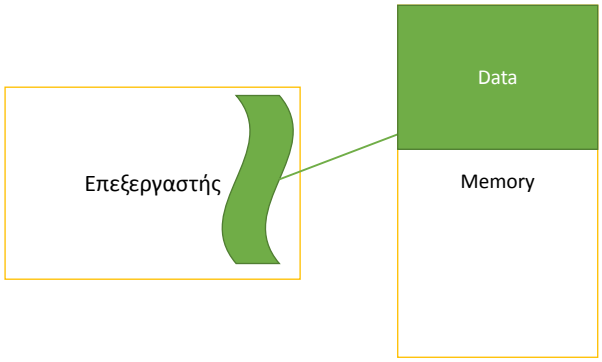
- Τι είναι η εικονική μνήμη ?

- Γιατί είναι σημαντική ?

2

Πρώτα συστήματα…

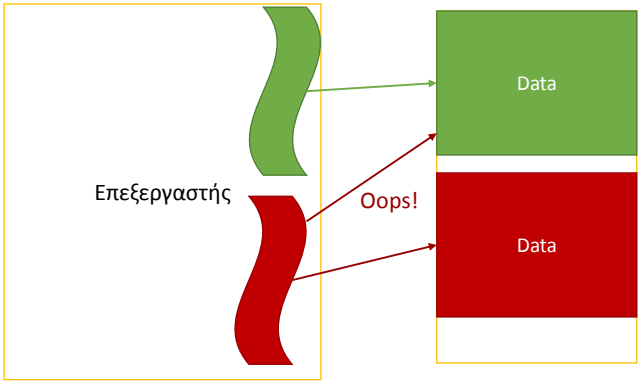
- Τα προγράμματα προσπελαύνουν απευθείας την φυσική μνήμη



3

Πρώιμα Συστήματα…

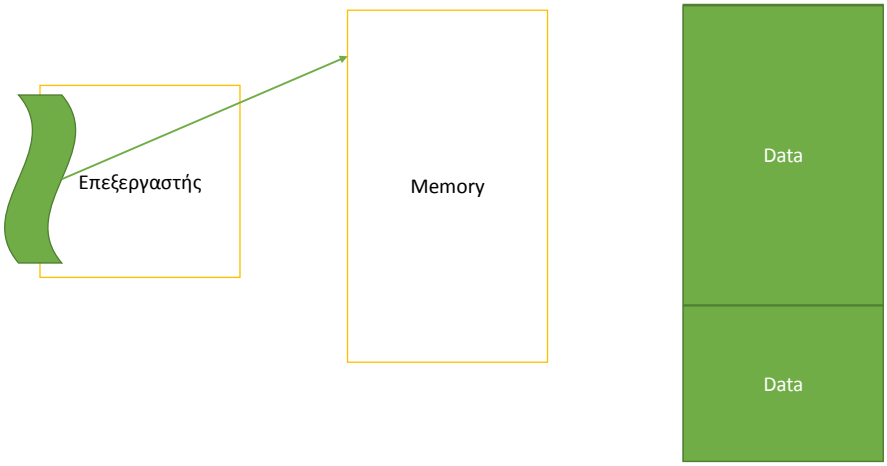
- Τα προγράμματα προσπελαύνουν απευθείας την φυσική μνήμη
- Δεν υπάρχει προστασία μεταξύ των προγραμμάτων
- Complex loading



4

Πρώιμα Συστήματα……

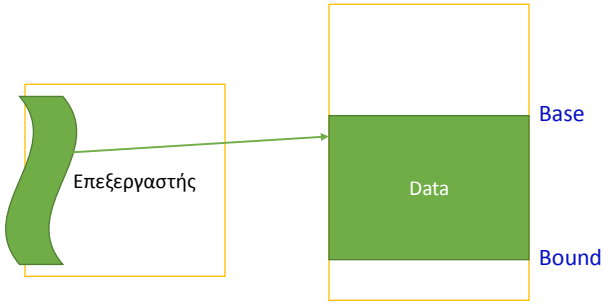
- Αν τα δεδομένα ήταν μεγαλύτερα από την μνήμη, τότε οι εφαρμογές έπρεπε να το αντιμετωπίσουν μόνες τους (π.χ., overlays)



5

Πρώιμα Συστήματα …

- Εικονική μνήμη: δίνει σε κάθε process την ψευδαίσθηση της δικής του μνήμης



- Πρώτες υλοποιήσεις: segments (base + bound)
  - Η μνήμη δεσμεύεται σε συνεχόμενα κομμάτια
  - Η μετάφραση γίνεται από το υλικό
  - Μη αποδοτική διαχείριση του χώρου!

6

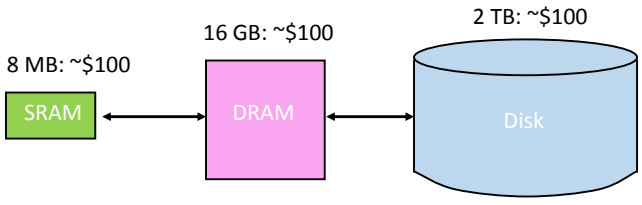
Γιατί Εικονική Μνήμη?

- Υπάρχουν 3 βασικοί λόγοι για την χρήση της Virtual Memory (VM):
  1. Επιτρέπει στην κύρια μνήμη (DRAM) να ενεργεί ως “cache” για τον δίσκο
  2. Απλοποιεί την διαχείριση μνήμης
  3. Προστασία των διαστημάτων διευθύνσεων
- Αλλά η VM λειτουργεί πολύ διαφορετικά από τις SRAM caches. Γιατί?
  - Ας αρχίσουμε με την διερεύνηση του πρώτου λόγου ...

7

Η DRAM ως Cache του Δίσκου

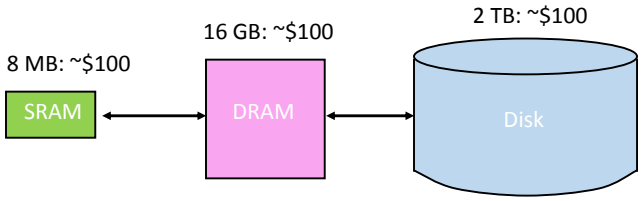
- Ο χώρος διευθύνσεων που ορίζεται από όλα bit διεύθυνσης είναι τεράστιος:
  - 32-bit addresses: ~4,000,000,000 (4 billion) bytes
  - 64-bit addresses: ~16,000,000,000,000,000,000 (16 quintillion) bytes
- Ο χώρος αποθήκευσης στο δίσκο είναι περίπου ~100X φθηνότερος σε σχέση με το DRAM storage
  - 2 TB of DRAM: ~ $10,000
  - 2 TB of disk: ~ $100
- Για να προσπελαστούν πολύ μεγάλες ποσότητες δεδομένων με έναν αποδοτικό/φθηνό τρόπο , τότε πρέπει ο όγκος δεδομένωνν να βρίσκεται στον δίσκο



8

Η SRAM ως Cache της DRAM

- DRAM vs. disk σε σχέση με SRAM vs. DRAM
  - Καθυστέρηση πρόσβασης :
    - Η DRAM είναι ~100X πιο αργή από την SRAM
    - Ο δίσκος είναι ~100,000X πιο αργός από την DRAM
    - Ο SSD ??
  - Ποια είναι η σημασία της τοπικότητας.... (**spatial locality**):
    - Για το πρώτο Byte απαιτείται ~100,000X πιο αργή πρόσβαση σε σχέση με τα επόμενα (σε συνεχόμενα bytes από τον disk)
    - Όμως υπάρχει ~4X βελτίωση όταν έχουμε page-mode σε σχέση με τις κανονικές προσβάσεις στην DRAM
  - Μέγεθος "Cache" :
    - Η κύρια μνήμη είναι ~1000X μεγαλύτερη από το μέγεθος της SRAM cache
  - Διαφορετικός τρόπος διευθυνσιοδότησης (η μνήμη έχει διευθύνσεις ενώ ο δίσκος έχει sector address)



9

Συνέπεια των Χαρακτηριστικών στη Σχεδίαση

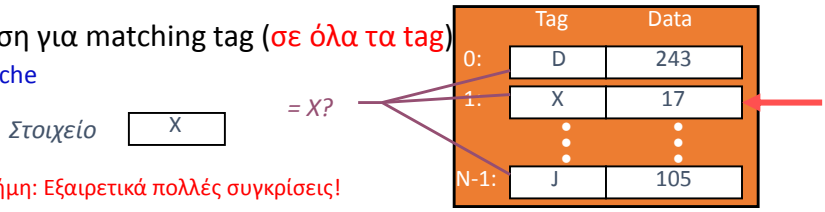
- Αν η DRAM έπρεπε να σχεδιαστεί ως μία SRAM cache, πως θα έπρεπε να ρυθμίσουμε τις ακόλουθες παραμέτρους?
  - Μέγεθος γραμμής?
  - Βαθμό προσεταιριστικότητας?
  - Πολιτική αντικατάστασης (για την περίπτωση της προσεταιριστικής)?
  - Write through ή write back?
- (Ποιο θα ήταν το αποτέλεσμα αυτών των επιλογών σε: miss rate, hit time, miss penalty, tag overhead, …)
- Πως υλοποιείται μια κρυφή μνήμη multi-GB, πλήρως προσεταιριστική?

10

Αναζήτηση ενός Στοιχείου

1. Αναζήτηση για matching tag (σε όλα τα tag)

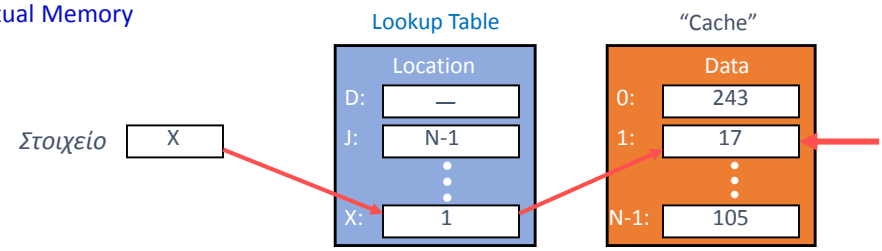
SRAM cache



Εικονική Μνήμη: Εξαιρετικά πολλές συγκρίσεις!

2. Χρήση έμμεσης αναζήτησης για να εύρεση της πραγματικής θέσης του στοιχείου

Virtual Memory



11

Ποιο το Κόστος των Tag?

- Πόσα tags? Πόσο μεγάλα είναι?
- Συμβατική SRAM cache
  - Υπάρχουν Tags για κάθε στοιχείο που γίνεται cached !
  - Το Tag αποθηκεύει την διεύθυνση του στοιχείου (ουσιαστικά τα υπόλοιπα bits της διεύθυνσης)
- Εικονική μνήμη ως έμμεση cache
  - Υπάρχουν Tags για κάθε στοιχείο (είτε είναι cached είτε όχι)
  - Το Tag αποθηκεύει την διεύθυνση του στοιχείου (τα υπόλοιπα bits της διεύθυνσης)
- Η κύρια διαφορά είναι στον αριθμό των tags
  - Πως μπορούν να διαχειριστούν τα tags της εικονικής μνήμης?
  - Στρατηγική: αποθήκευση των tags στην κύρια memory και βάλε μια cache να κρατάς τα πιο πρόσφατα tags

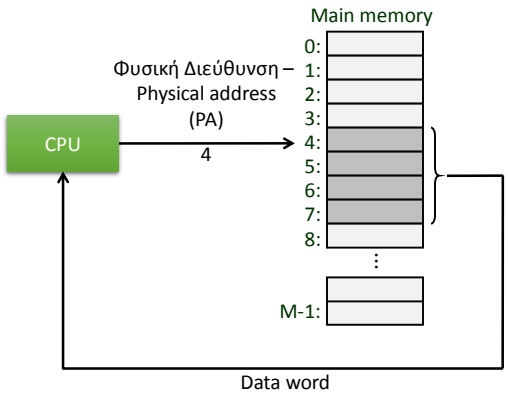
12

Χώρος Διευθύνσεων

- Χώρος Εικονικών Διευθύνσεων: Έστω ότι υποστηρίζονται N = 2^n εικονικές διευθύνσεις
  {0, 1, 2, 3, …, N-1}
- Χώρος Φυσικών Διευθύνσεων: Έστω ότι έχουμε M = 2^m φυσικές διευθύνσεις
  {0, 1, 2, 3, …, M-1}
- Απαιτείται να υπάρχει διάκριση μεταξύ data (bytes) και των διευθύνσεων (addresses)
- Κάθε δεδομένο μπορεί να έχει πολλές διευθύνσεις
- Κάθε byte στην κύρια μνήμη έχει:
  μία φυσική διεύθυνση, μια (ή περισσότερες) εικονικές διευθύνσεις

13

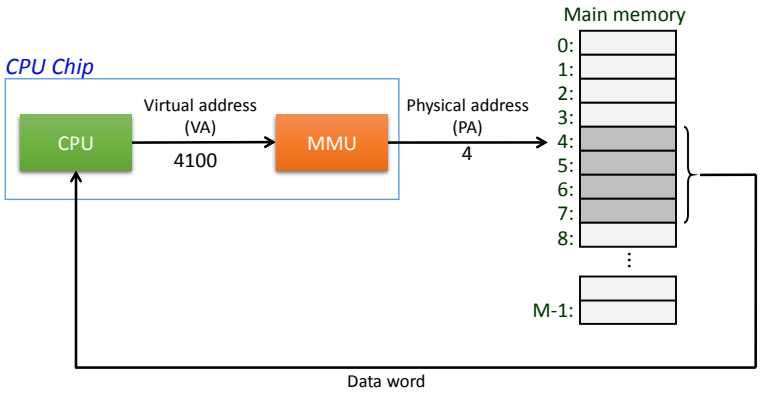
Σύστημα με Φυσικές Διευθύνσεις



- Χρησιμοποιείται σε αρκετά “απλά” συστήματα, όπως ενσωματωμένα συστήματα (microcontrollers) σε αυτοκίνητα, ασανσέρ, και ψηφιακά μικροσυστήματα

14

Σύστημα με ΕικονικήΔιευθυνσιοδότηση



- Χρησιμποιείται σε όλους τους servers, desktops, and laptops
- Μία από τις μεγαλύτερες ιδέες στην computer science

15

Why Virtual Memory? (Further Details)

(1) Η Εικονική μνήμη επιτρέπει αποδοτική χρήση της περιορισμένης κύριας μνήμης (RAM)
- Χρήση της RAM ως cache για τμήματα του χώρου εικονικών διευθύνσεων
- Διατήρηση μόνο των ενεργών περιοχών του χώρου εικονικών διευθύνσεων στη μνήμη
  - Μεταφορά δεδομένων πίσω και εμπρός κατά απαίτηση

(2) Η Εικονική μνήμη απλοποιεί την διαχείριση μνήμης για τους προγραμματιστές
- Κάθε process απολαμβάνει έναν πλήρη, ιδιωτικό και γραμμικό χώρο διευθύνσεων

(3) Η Εικονική μνήμη εξασφαλίζει απομόνωση των χώρων διευθύνσεων
- Μία διεργασία δεν μπορεί να παρέμβει σε άλλης την περιοχή μνήμης
  - Διότι λειτουργούν σε διαφορετικούς χώρους διευθύνσεων
- Διεργασίες του χρήστη δεν μπορούν να έχουν πρόσβαση σε privileged information
  - Διαφορετικές περιοχές των χώρων διευθύνσεων έχουν different permissions

16

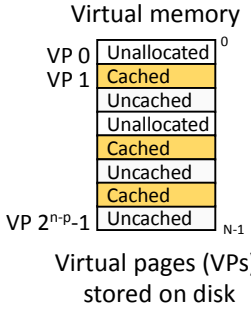
Λόγοι Δημιουργίας VM

- 3 λόγοι για την υλοποίηση Virtual Memory (VM):
  1. Επιτρέπει στην κύρια μνήμη (DRAM) να ενεργεί ως “cache” για τον δίσκο
  2. Απλοποιεί την διαχείριση μνήμης
  3. Προστασία των διαστημάτων διευθύνσεων
- Για να λυθεί το #1 αποδοτικά, εισαγάγαμε έναν τρόπο *indirection*
- Με αυτή τη λύση είναι επίσης εύκολο να λυθεί #2 και το #3:
  - Απλοποίηση της διαχείρισης μνήμης:
    - Ευέλικτη απεικόνιση-αντστοίχιση των εικονικών σε φυσικές διευθύνσεις
  - Προστασία των χώρων διευθύνσεων:
    - Πληροφορία για προστασία μπορεί να αποθηκευτεί στον πίνακα αντιστοίχισης (lookup table)
    - Και να ελέγχει κάθε αίτημα για πρόσβαση στην φυσική μνήμη

17

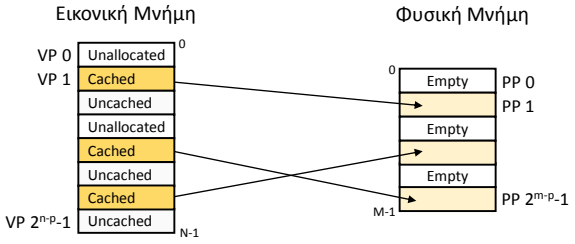
Η Εικονική Μνήμη: Λειτουργία ως Cache

- *Η Εικονική Μνήμη* είναι ένας πίνακας από N συνεχόμενα bytes
  - Μπορεί να θεωρηθεί ότι η VM υποστηρίζεται από τον χώρο στον δίσκο
- Τα περιεχόμενα του πίνακα στον disk γίνονται cached στην *φυσική μνήμη* (στην *DRAM ως cache*)
  - Αυτά τα μπλοκ της cache ονομάζονται σελίδες (*pages*) (size is $P = 2^p$ bytes)



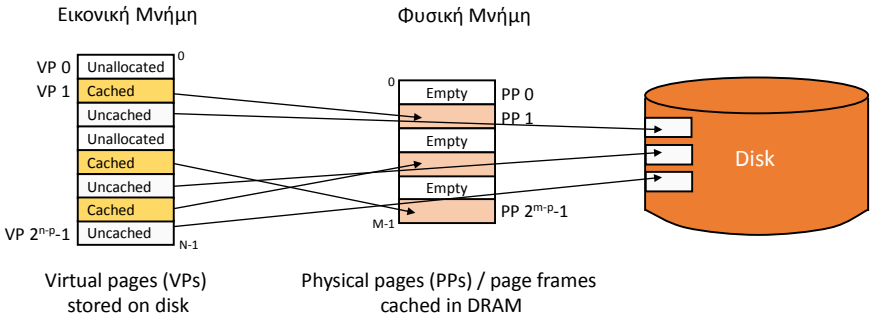
18

Η Εικονική Μνήμη: Λειτουργία ως Cache (2)



19

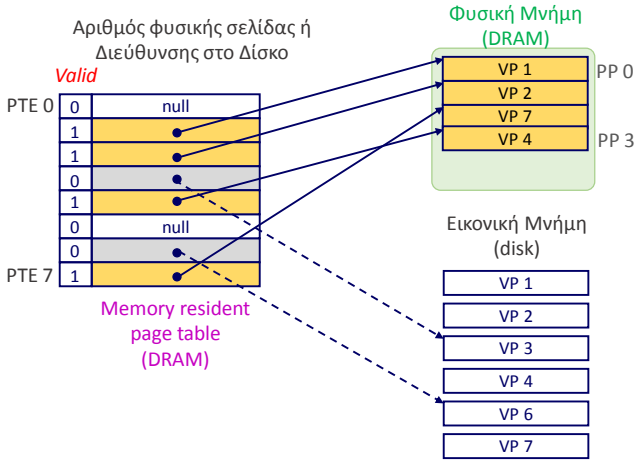
Η Εικονική Μνήμη: Λειτουργία ως Cache (3)



20

Πίνακας Σελίδων Εικονικής Μνήμης

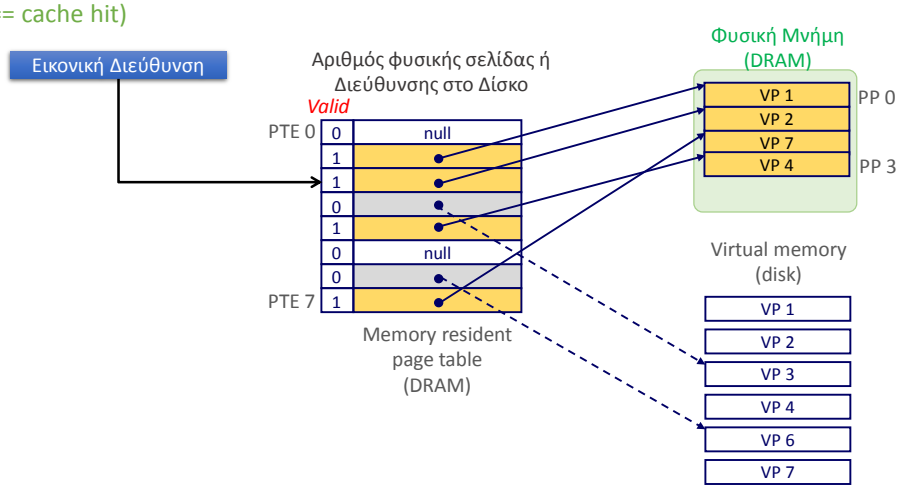
- *Πίνακας Σελίδων (Page table):* είναι ένας πίνακας από τους αριθμούς των σελίδων - page table entries (PTEs) – που κρατά τις αντιστοιχίσεις των εικονικών στις φυσικές σελίδες (== tags)
  - Είναι μια δομή δεδομένων που διαχειρίζεται ο πυρήνας για κάθε διεργασία in DRAM



21

Επιτυχής Πρόσβαση σε Σελίδα (Page Hit)

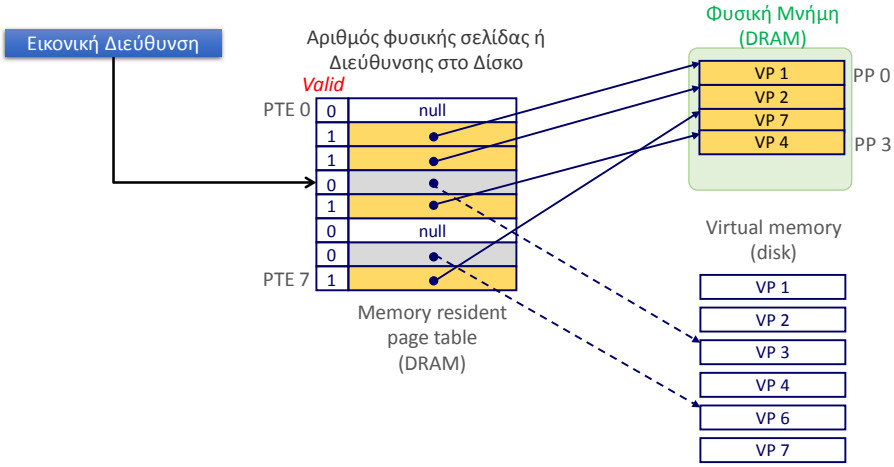
- *Page hit:* αναφορά σε λέξη της Εικονικής μνήμης (VM word) που βρίσκεται στην φυσική μνήμη (== cache hit)



22

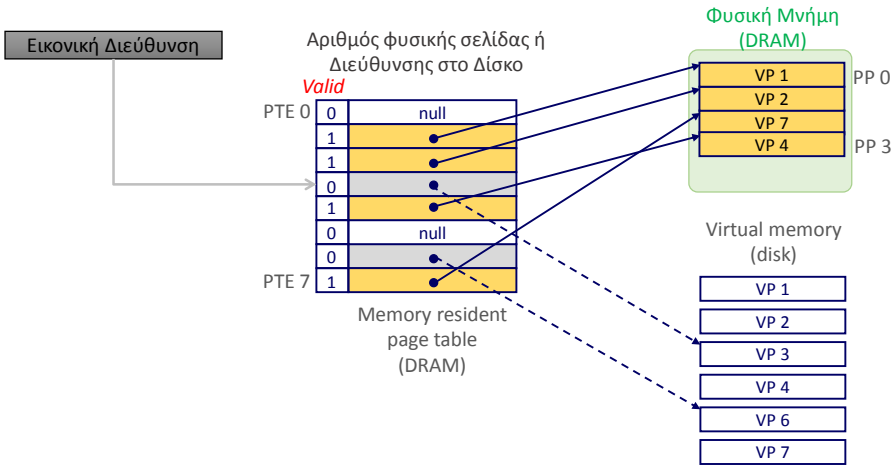
Αποτυχημένη Πρόσβαση σε Σελίδα (Page Hit)

- *Page fault:* αναφορά σε λέξη της Εικονικής μνήμης (VM word) που δε βρίσκεται στην φυσική μνήμη (== cache miss)



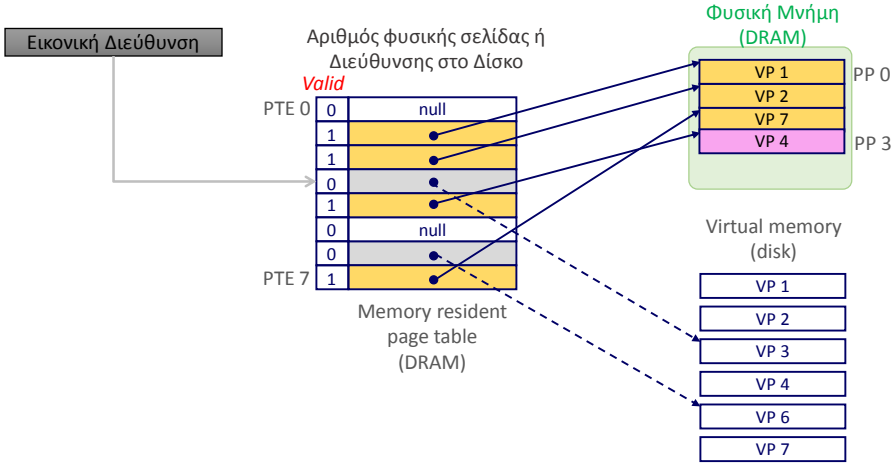
Αντιμετώπιση ενός Page Fault

- Μία αποτυχημένη πρόσβαση σε σελίδα (Page miss) προκαλεί ένα Λάθος Σελίδας (page fault) (an exception – με αποτέλεσμα την αντιμετώπισή του από το software!)



Αντιμετώπιση ενός Page Fault (2)

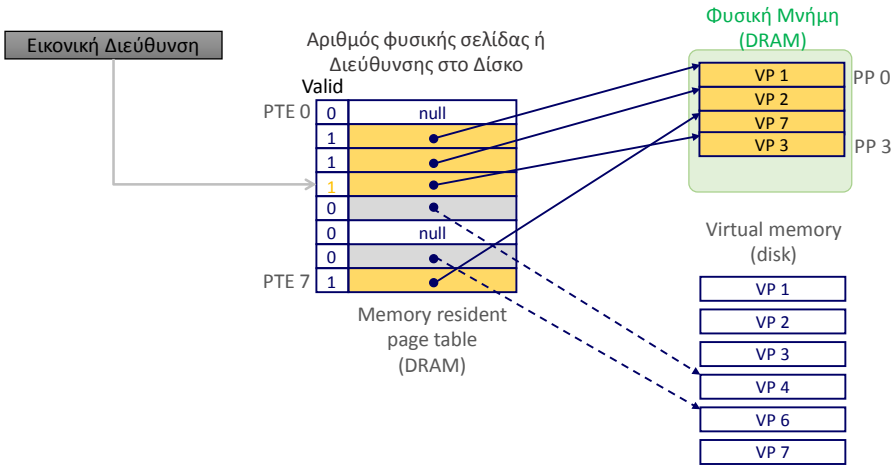
- Μία αποτυχημένη πρόσβαση σε σελίδα (Page miss) προκαλεί ένα Λάθος Σελίδας (page fault)
- Ο διαχειριστής λαθών σελίδας (Page fault handler) επιλέγει τη σελίδα θύμα (victim) (εδώ η VP 4)



25

Αντιμετώπιση ενός Page Fault (3)

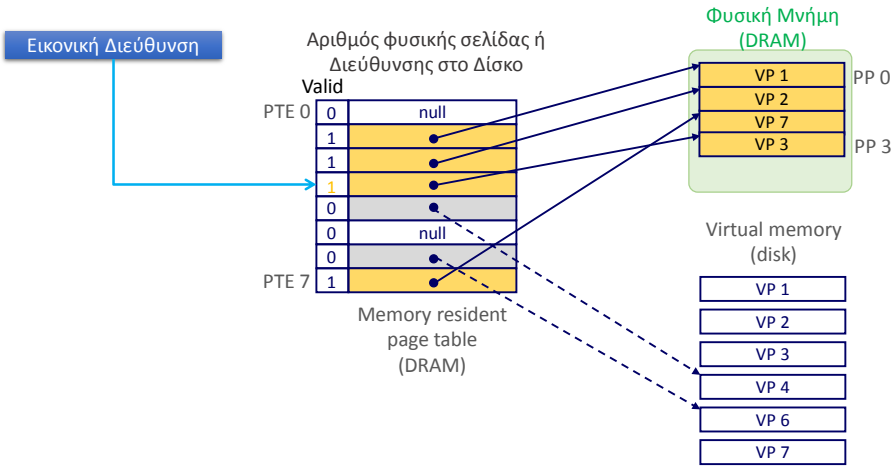
- Ο διαχειριστής λαθών σελίδας (Page fault handler) επιλέγει τη σελίδα θύμα (victim) (εδώ η VP 4)



26

Αντιμετώπιση ενός Page Fault (4)

- Η εντολή που προκάλεσε το λάθος σελίδας (page fault) είναι έτοιμη να ξανα-αρχίσει: page hit!



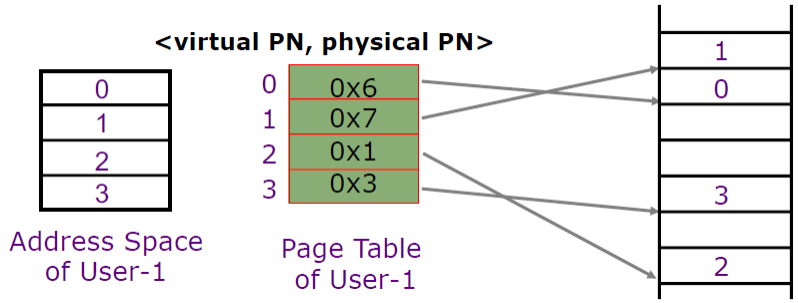
Η Τοπικότητα Προσφέρει τη Λύση ξανά!

- Η Εικονική μνήμη είναι αποτελεσματική εξ'αιτίας της τοπικότητας
- Σε οποιαδήποτε χρονική στιγμή, τα προγράμματα έχουν την τάση να κάνουν πρόσβαση σε ένα σετ ενεργές σελίδες που ονομάζεται *working set*
  - Προγράμματα με καλύτερη τοπικότητα στον χρόνο συνήθως έχουν μικρότερα working sets
- If (working set size < main memory size)
  - Τότε έχουμε καλή απόδοση για μία διεργασία μετά το τέλος των αναγκαστικών αποτυχιών (compulsory misses)
- If (SUM(working set sizes) > main memory size)
  - *Thrashing:* καταρράκωση της απόδοσης καθώς σελίδες μετακινούνται (copied) συνεχώς μέσα-έξω από την μνήμη

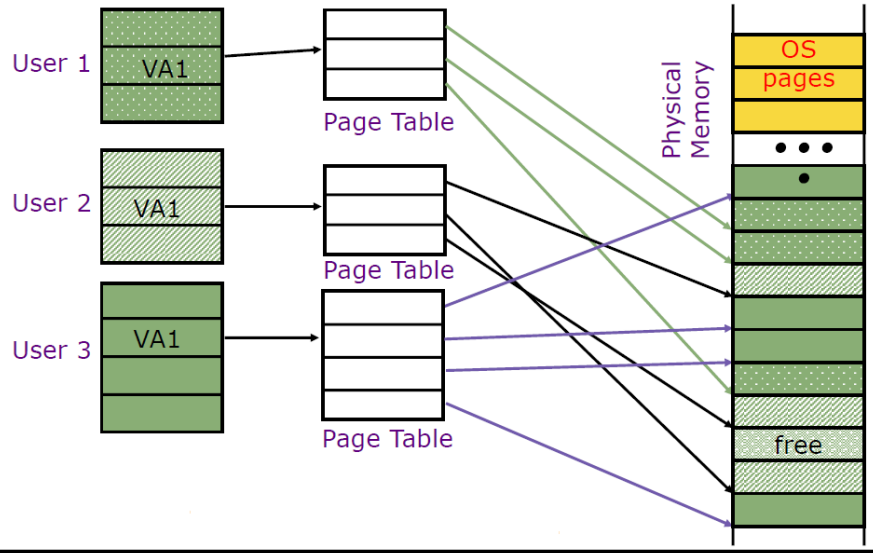
Συνέχεια...

Συστήματα με Εικονική Μνήμη

- Η διεύθυνση που στέλνει ο επεξεργαστής μπορεί να την δει κανείς ως <αριθμό σελίδας, offset>
- Ένας πίνακας σελίδων περιέχει την διεύθυνση αρχής της φυσικής σελίδας

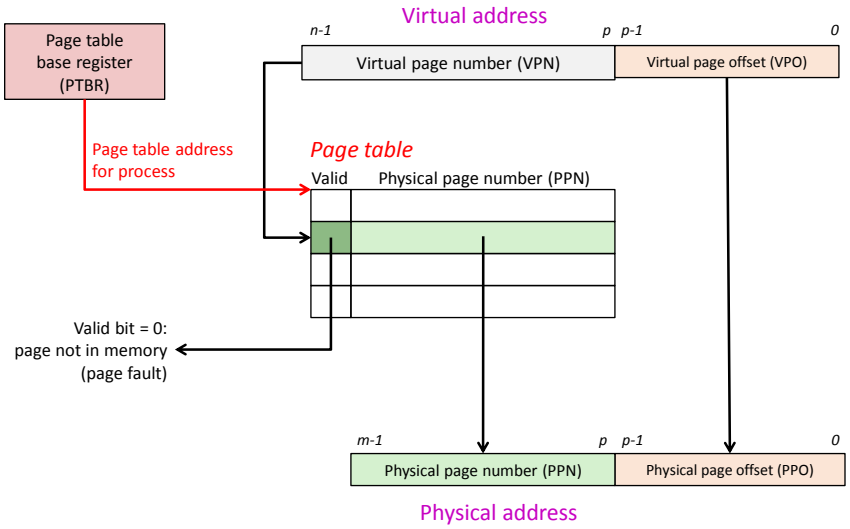


Χώρος Εφαρμογών του Χρήστη



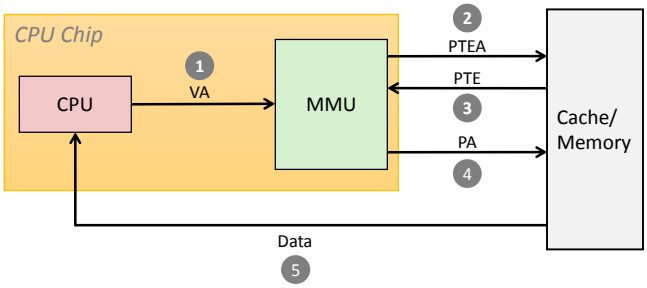
31

Μετάφραση Διεύθυνσης με έναν Πίνακα Σελίδων



39

Μετάφραση Διεύθυνσης: Page Hit



1) Ο επεξεργαστής στέλνει την εικονική διεύθυνση στην MMU

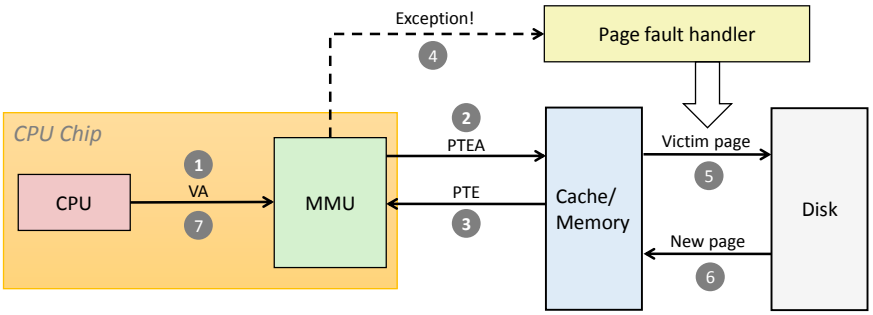
2-3) Η MMU φέρνει τα PTE από τον πίνακα σελίδων στην μνήμη

4) Η MMU στέλνει την φυσική διεύθυνση στην cache/memory

5) Η Cache/memory στέλνει τα δεδομένα στον επεξεργαστή

40

Μετάφραση Διεύθυνσης: Page Fault



1 Ο επεξεργαστής στέλνει την εικονική διεύθυνση στην MMU

2-3) Η MMU φέρνει τα PTE από τον πίνακα σελίδων στην μνήμη

4) Το Valid bit είναι 0, οπότε η MMU ενεργοποιεί την εξαίρεση για σφάλμα σελίδας

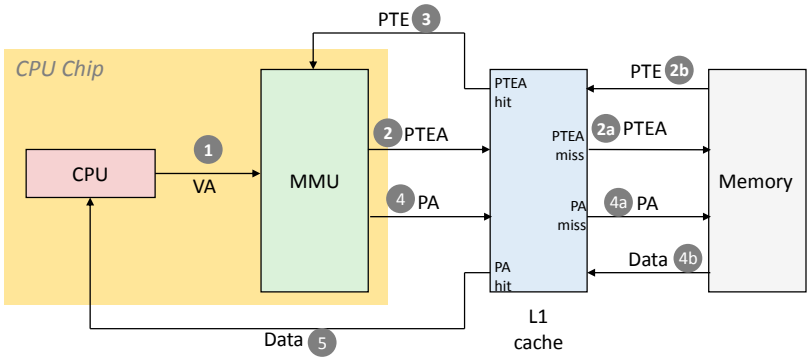
5) Ο χειριστής σφάλματος βρίσκει το θύμα (και, αν είναι dirty, στέλνει τη στέλνει τη σελίδα στο δίσκο)

6) Ο χειριστής σφάλματος φέρνει την νέα σελίδα και ενημερώνει το PTE στην μνήμη

7) Ο χειριστής σφάλματος επιστρέφει στην αρχική διαδικασία, ξεκινώντας ξανά την αναφορά στην μνήμη

41

Σχεδίαση VM και Cache: σε ποιο chip?



VA: virtual address, PA: physical address,
PTE: page table entry, PTEA = PTE address

42

Μειονέκτημα:

- Αναφορά στη μνήμη 2 φορές για κάθε προσπέλαση? ...καθυστέρηση...

43

Επιτάχυνση της Μετάφρασης με TLB

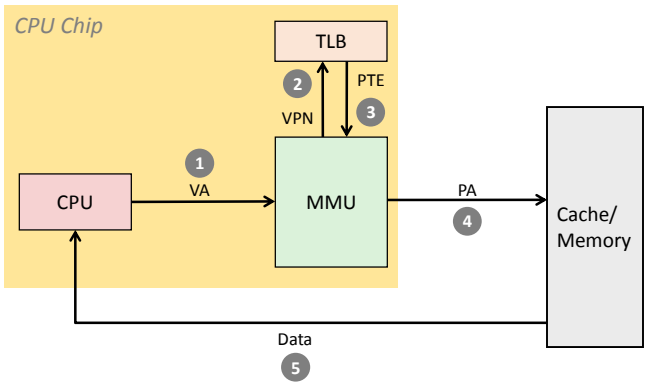
- Page table entries (PTEs) αποθηκεύονται στην L1 κρυφή μνήμη σαν οποιαδήποτε άλλη λέξη
  - Τα PTEs μπορεί να αντικατασταθούν από άλλες αναφορές στη μνήμη
  - ένα PTE hit απαιτεί μία μικρή καθυστέρηση (L1 delay)

- Λύση: *Translation Lookaside Buffer* (TLB)
  - Είναι μια μικρή cache μέσα στο MMU
  - Κρατάει τις απεικονίσεις των αριθμών της εικονικής σελίδας στην φυσική σελίδα

- Τα TLBs έχουν ψηλά hit rates με λίγα στοιχεία. γιατί?
  - 512 entries ➔ αναφορές που εκτείνονται σε 512 * 4KB = 2MB

44

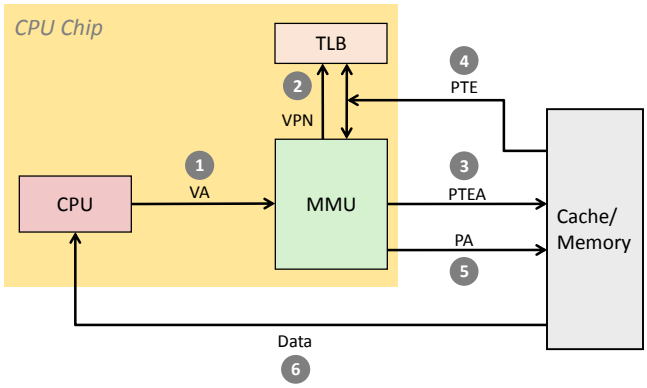
TLB Hit



Ένα TLB hit γλυτώνει μία πρόσβαση στη μνήμη !

45

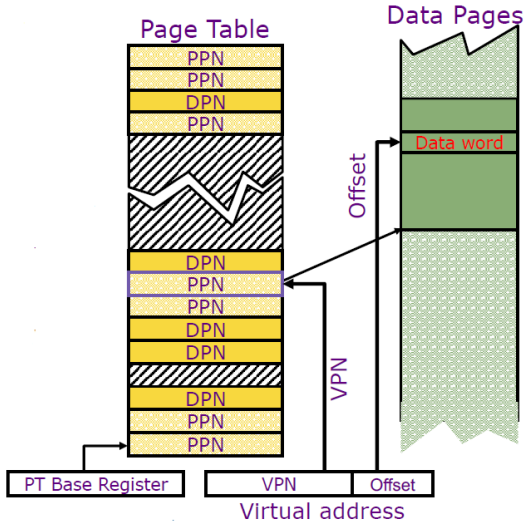
TLB Miss



Ένα TLB miss συνεπάγεται μία επιπλέον πρόσβαση στη μνήμη (PTE)
ευτυχώς τα TLB misses είναι σπάνια...

Μοντέρνα Συστήματα με Εικονική Μνήμη

- Γραμμικός Πίνακας Σελίδων
- Μία γραμμή του Πίνακα Σελίδων (Page Table Entry) περιέχει:
  - Ένα bit αν η σελίδα υπάρχει
  - Τον αριθμό φυσικής σελίδας (PPN) για μια σελίδα που βρίσκεται στην μνήμη
  - Τον αριθμό σελίδας στο δίσκο (DPN) για μια σελίδα που βρίσκεται στον δίσκο
- Το λειτουργικό σύστημα αρχικοποιεί τον καταχωρητή βάσης (PT Base Register) όποτε γίνεται ενεργή μια νέα διεργασία του χρήστη

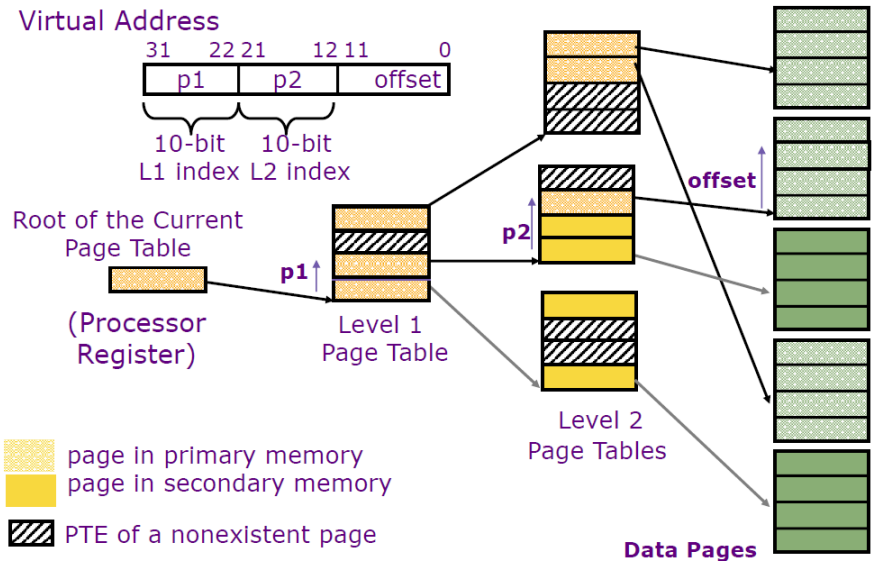


Μέγεθος ενός Πίνακα Σελίδων (Γραμμικός)

- Έστω διευθύνσεις 32-bit, σελίδες μεγέθους 4KB, PTE μεγέθος 4 Byte
  - 2^{20} PTEs, συνεπώς απαιτείται πίνακας σελίδων 4 MB ανά χρήστη
  - 4 GB swap space απαιτείται για την αποθήκευση όλου του χώρου εικονικών σελίδων
- Λύση: σελίδες μεγαλύτερου μεγέθους ?
  - Εσωτερικός κατακερματισμός (υπο-χρησιμοποίηση της μνήμης σε μία σελίδα)
  - Μεγαλύτερο κόστος σε περίπτωση page fault (χρόνος μεταφοράς από δίσκο)
- Τι γίνεται αν έχουμε διευθύνσεις 64-bit ?
  - Με σελίδες 1MB απαιτείται 2^{44} 8-byte PTEs (και συνεπώς 35 TB!)

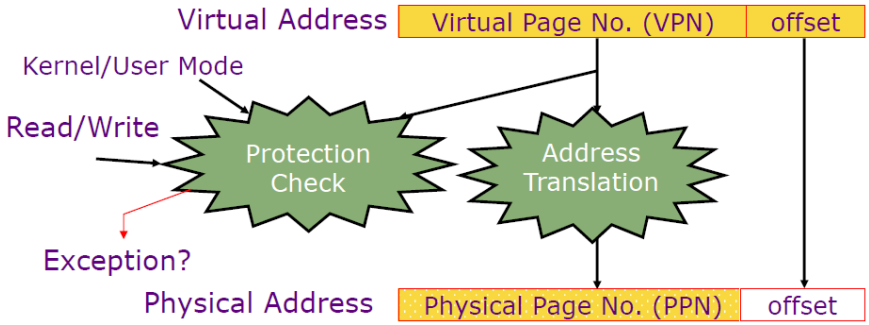
48

Πίνακας Σελίδων με Ιεραρχία



49

Μετάφραση Διευθύνσεων και Προστασία

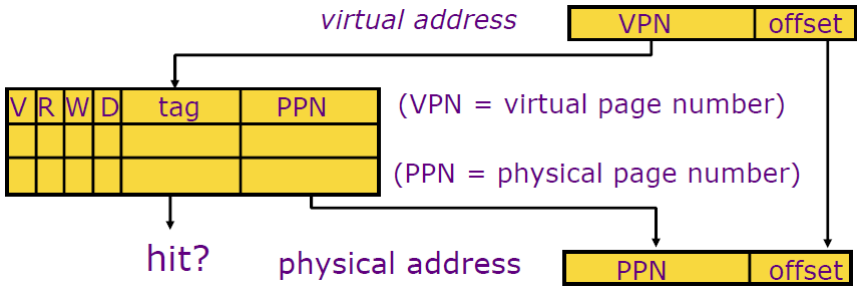


- Κάθε πρόσβαση σε εντολές και δεδομένα απαιτεί μετάφραση διεύθυνσης και έλεγχο για προστασία της πρόσβασης
- Μία καλή σχεδίαση συστήματος με Εικονική Μνήμη πρέπει να είναι γρήγορη (με κόστος ~ 1 κύκλο ρολογιού) και να μην απαιτεί πολύ χώρο

50

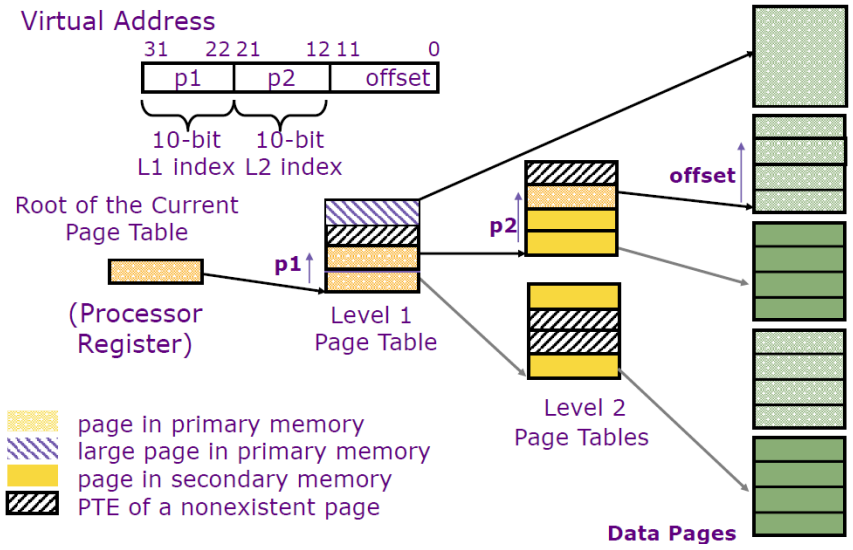
Επιτάχυνση Μετάφρασης: TLB

- Translation Lookaside Buffer: είναι μια cache για την μετάφραση διευθύνσεων
  - TLB hit → Μετάφραση σε 1 κύκλο
  - TLB miss → *Page Table Walk για φόρτωση της cache*



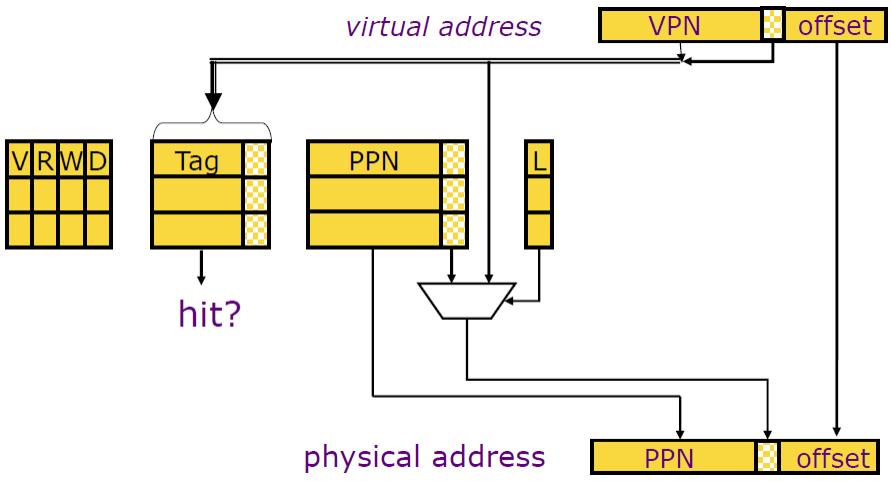
51

Υποστήριξη για Σελίδες Μεταβλητού Μεγέθους



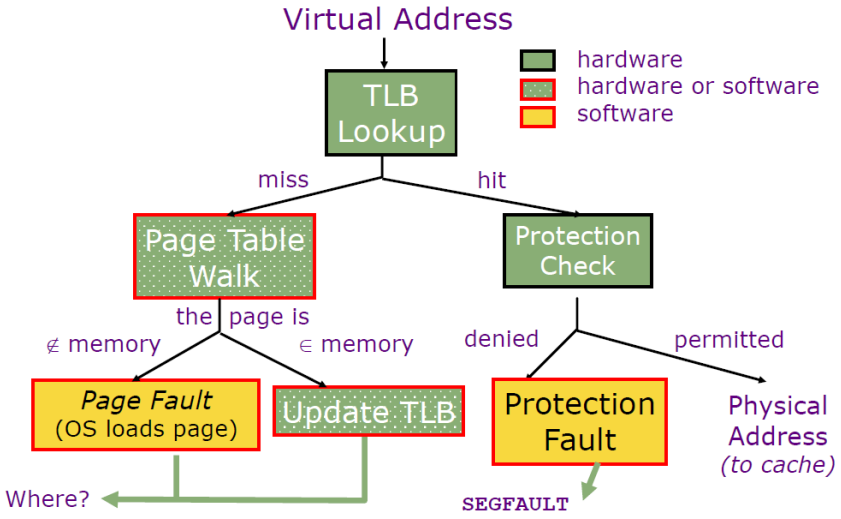
52

TLB για Σελίδες Μεταβλητού Μεγέθους



53

Εικονική Διεύθυνση: Συνολική Διαδικασία

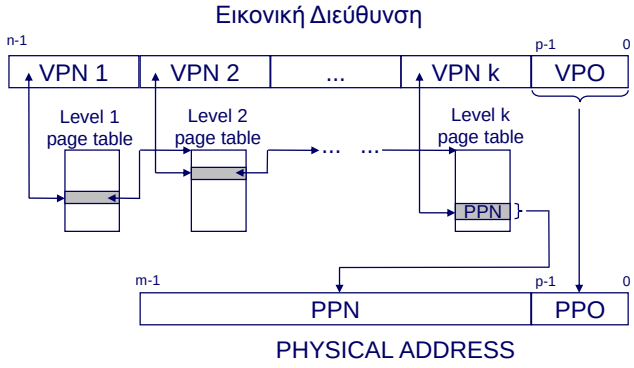


54

55

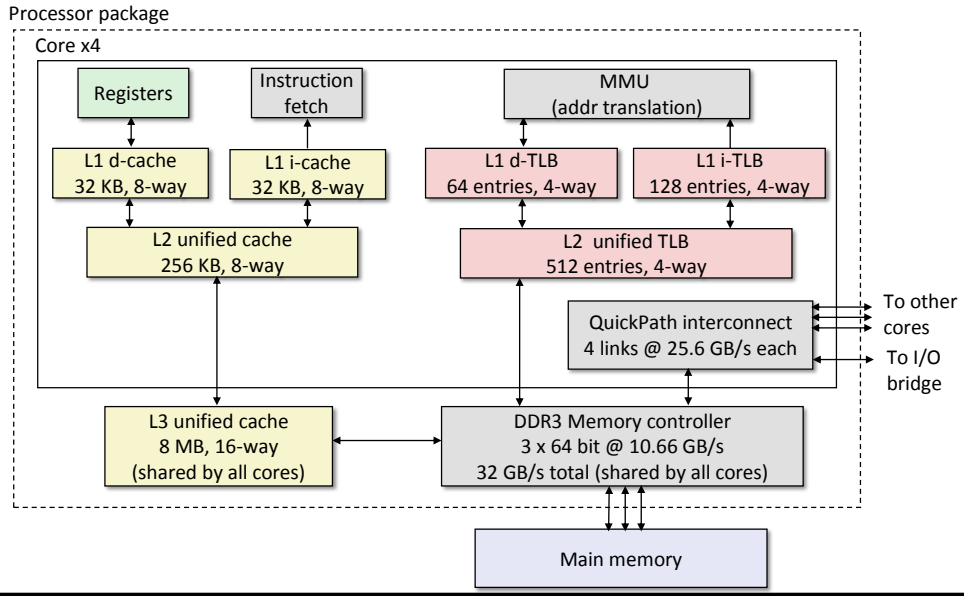
69

Μετάφραση με χρήση Πίνακα k-Επιπέδων



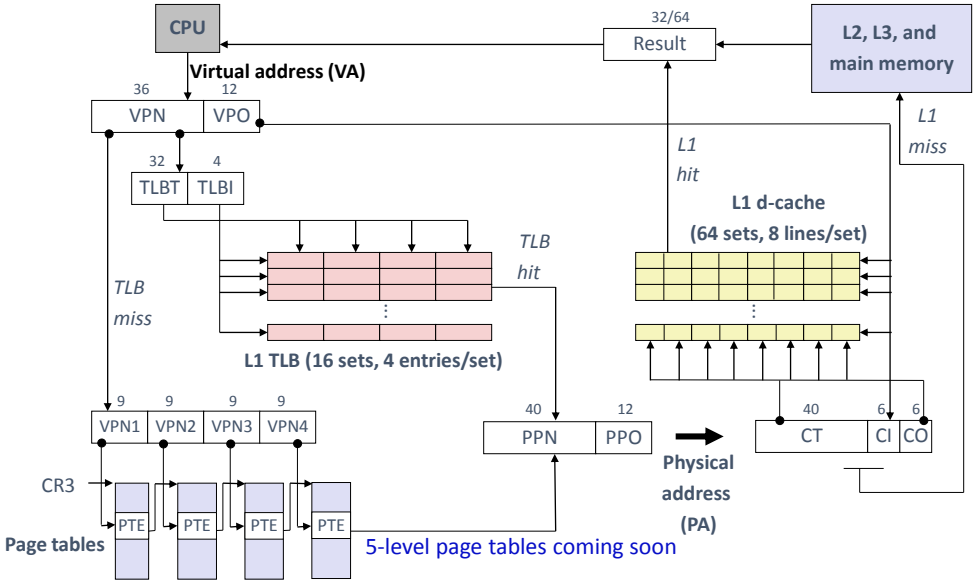
70

Το Σύστημα Μνήμης του Intel Core i7



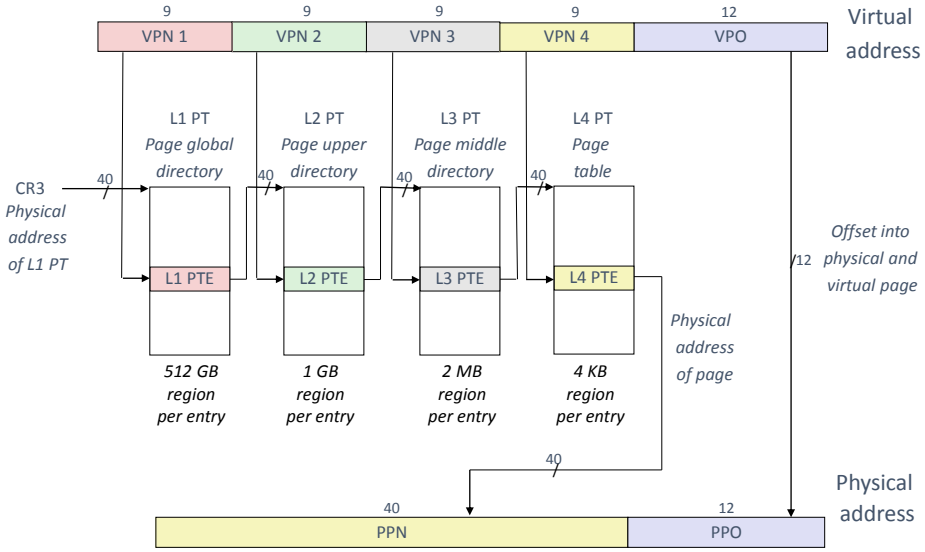
71

Μετάφραση Διεύθυνσης στον Core i7



73

Μετάφραση Διεύθυνσης με τον Πίνακα Σελίδων στον Core i7



Core i7 Level 1-3 Page Table Entries

63	62–52	51–12	11–9	8	7	6	5	4	3	2	1	0
XD	Unused	Page table physical base address	Unused	G	PS		A	CD	WT	U/S	R/W	P=1
Available for OS (page table location on disk)												P=0

Each entry references a 4K child page table

P: Child page table present in physical memory (1) or not (0).

R/W: Read-only or read-write access access permission for all reachable pages.

U/S: user or supervisor (kernel) mode access permission for all reachable pages.

WT: Write-through or write-back cache policy for the child page table.

CD: Caching disabled or enabled for the child page table.

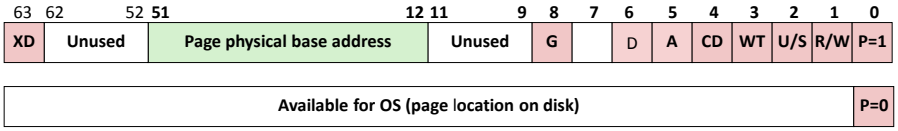
A: Reference bit (set by MMU on reads and writes, cleared by software).

PS: Page size either 4 KB or 4 MB (defined for Level 1 PTEs only).

G: Global page (don't evict from TLB on task switch)

Page table physical base address: 40 most significant bits of physical page table address (forces page tables to be 4KB aligned)

Core i7 Level 4 Page Table Entries



Each entry references a 4K child page

P: Child page is present in memory (1) or not (0)

R/W: Read-only or read-write access permission for child page

U/S: User or supervisor mode access

WT: Write-through or write-back cache policy for this page

CD: Cache disabled (1) or enabled (0)

A: Reference bit (set by MMU on reads and writes, cleared by software)

D: Dirty bit (set by MMU on writes, cleared by software)

G: Global page (don't evict from TLB on task switch)

Page physical base address: 40 most significant bits of physical page address (forces pages to be 4KB aligned)

76

Πρόσφατη Έρευνα στην Εικονική Μνήμη

- Προβλήματα:
  - Τα πολλά επίπεδα ανακατεύθυνσης/έμμεσης διευθυνσιοδότησης είναι αργά
  - Είναι δύσκολο να έχεις αποτελεσματική απεικόνιση μεγάλων (~ GBs) working sets όταν υπάρχουν μικρά TLB

- Ποιοί ήταν οι λόγοι για την σχεδίαση Εικονικής μνήμης με σελίδες ?
  - Να γίνει πιο αποτελεσματική η χρήση της περιορισμένης DRAM ?
  - Να απλοποιηθεί η διαχείριση μνήμης για τους προγραμματιστές ✓
  - Να υπάρχει προστασία στα προγράμματα ✓

- Μηχανισμοί για υποστήριξη μεγάλων περιοχών μνήμης
  - Γιγάντιες σελίδες (2MB, 1GB), supported in current hardware & Linux
  - Segments *[Karakostas et al, ISCA'15]*
  - Συνένωση μικρών (Fine-grain) σελίδων σε συνεχόμενο χώρο *[Park et al, ISCA'17]*

- Αυτές οι ιδέες ρισκάρουν την αποδοτική χρήση της DRAM – απαιτείται εκχώρηση συνεχόμενου χώρου στην φυσική μνήμη

77

Συμπεράσματα

- Σημαντικά προβλήματα στην σχεδίαση Εικονικής Μνήμης με στόχο
  - Την αποδοτική χρήση της φυσικής μνήμης
  - Την απλοποίηση της διαχείρισης μνήμης
  - Την προστασία όταν υπάρχει διαμοιρασμός
- Υλοποίηση με χρήση DRAM ως cache
  - Η σχεδίαση είναι διαφορετική από τις caches του επεξεργαστή
- Η επέκταση αυτής της σχεδίασης ενέχει πολλές παραμέτρους στο συνολικό επεξεργαστικό σύστημα
  - Η βελτίωση χρήσης Εικονικής Μνήμης αποτελεί μια ενεργή ερευνητική περιοχή

78