

Αρχιτεκτονική Υπολογιστών

Κεφάλαιο #5 (α)

Ιεραρχίες Μνήμης

Διονύσης Πνευματικάτος

pnevmati@cslab.ece.ntua.gr

5ο εξάμηνο ΣΗΜΜΥ – Ακαδημαϊκό Έτος: 2019-20

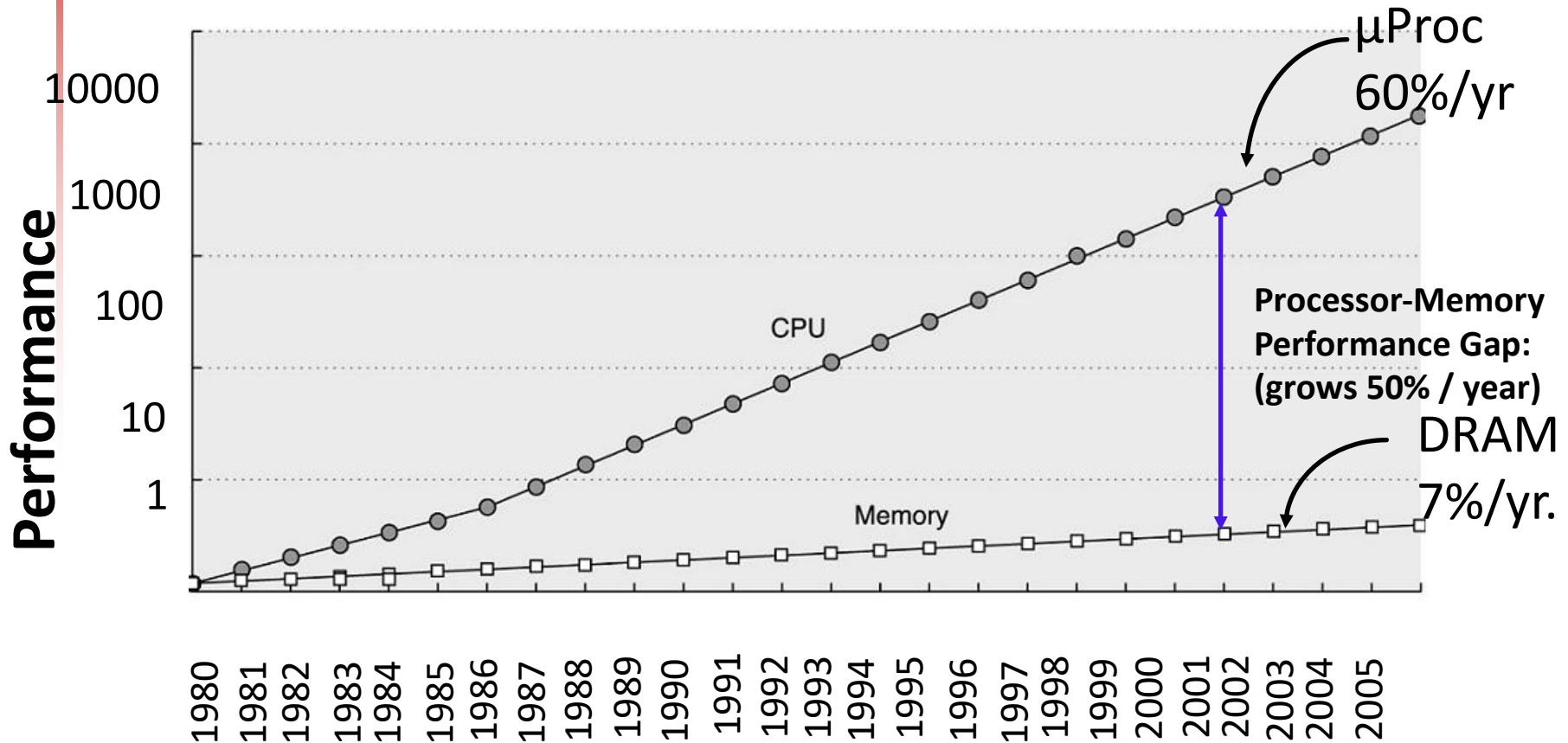
Τμήμα 3 (ΠΑΠΑΔ-Ω)

<http://www.cslab.ece.ntua.gr/courses/comparch/>

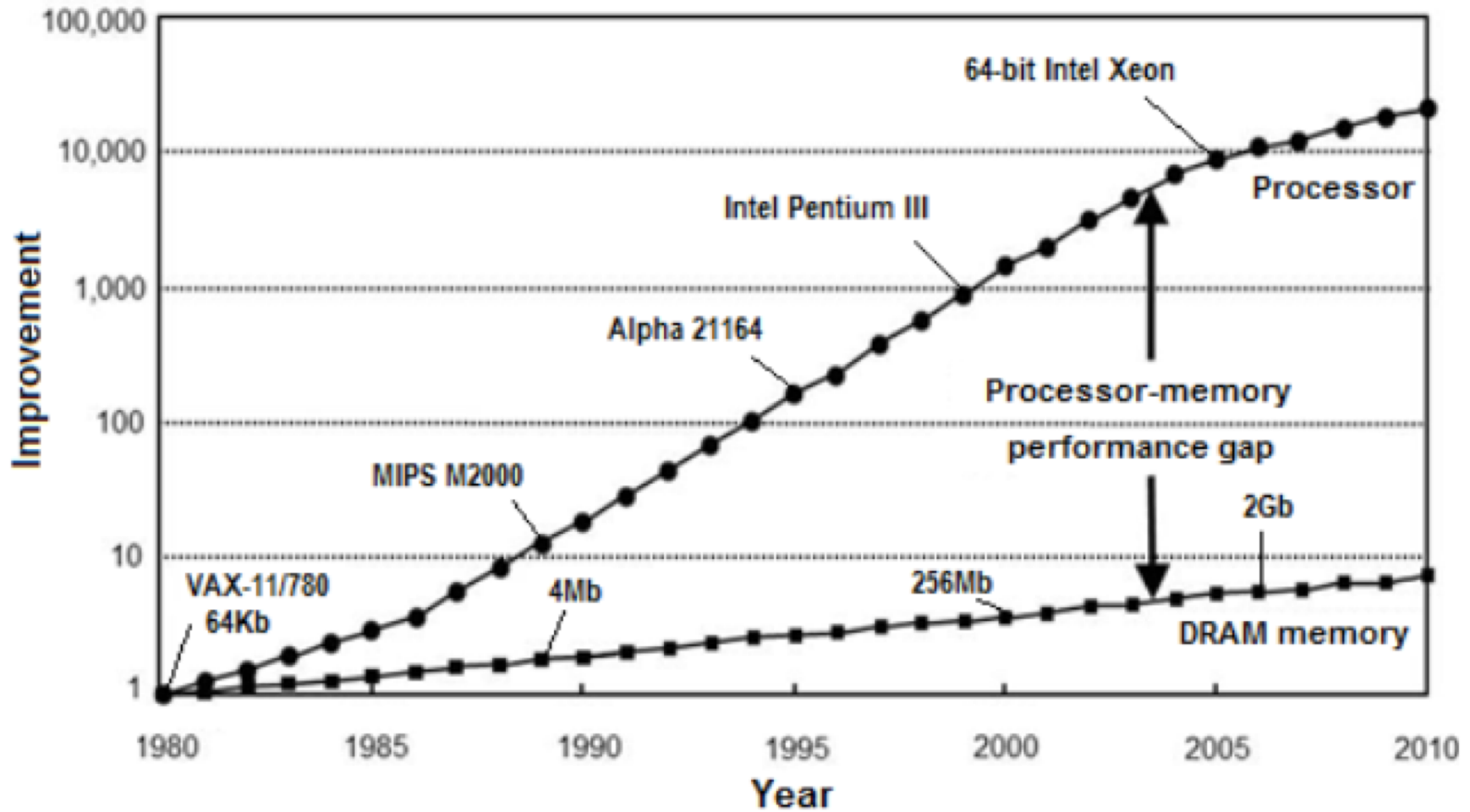
Τεχνολογία μνήμης

- Στατική RAM (Static RAM – SRAM)
 - 0.5ns – 2.5ns, \$2000 – \$5000 ανά GB
- Δυναμική RAM (Dynamic RAM – DRAM)
 - 50ns – 70ns, \$20 – \$75 ανά GB
- Μαγνητικός δίσκος
 - 5ms – 20ms, \$0.20 – \$2 ανά GB
- Ιδανική μνήμη
 - Χρόνος προσπέλασης της SRAM
 - Χωρητικότητα και κόστος/GB του δίσκου

Διαφορά Επίδοσης Επεξεργαστή/μνήμης (DRAM)



Διαφορά Επίδοσης Επεξεργαστή/μνήμης (DRAM)



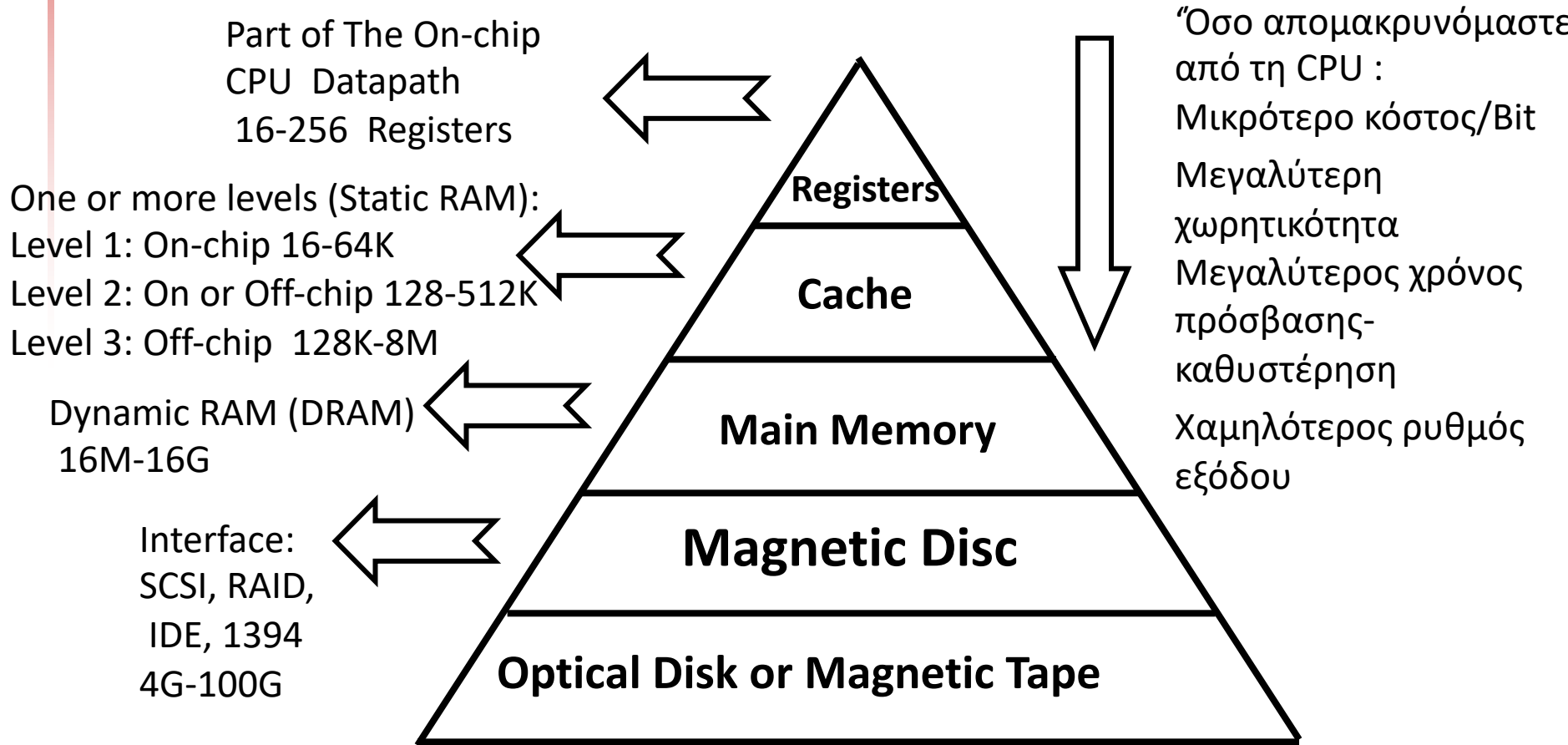
Αρχή της τοπικότητας (locality)

- Τα προγράμματα προσπελάζουν ένα μικρό μέρος του χώρου δ/νσεών τους κάθε στιγμή
- Χρονική τοπικότητα (temporal locality)
 - Αντικείμενα που προσπελάστηκαν πρόσφατα είναι πιθανό να προσπελαστούν πάλι σύντομα
 - π.χ., εντολές σε ένα βρόχο, μεταβλητές επαγωγής (induction variables)
- Χωρική τοπικότητα (spatial locality)
 - Αντικείμενα κοντά σε αυτά που προσπελάστηκαν πρόσφατα είναι πιθανό να προσπελαστούν σύντομα
 - π.χ., προσπέλαση εντολών στη σειρά, δεδομένα πινάκων

Εκμετάλλευση της τοπικότητας

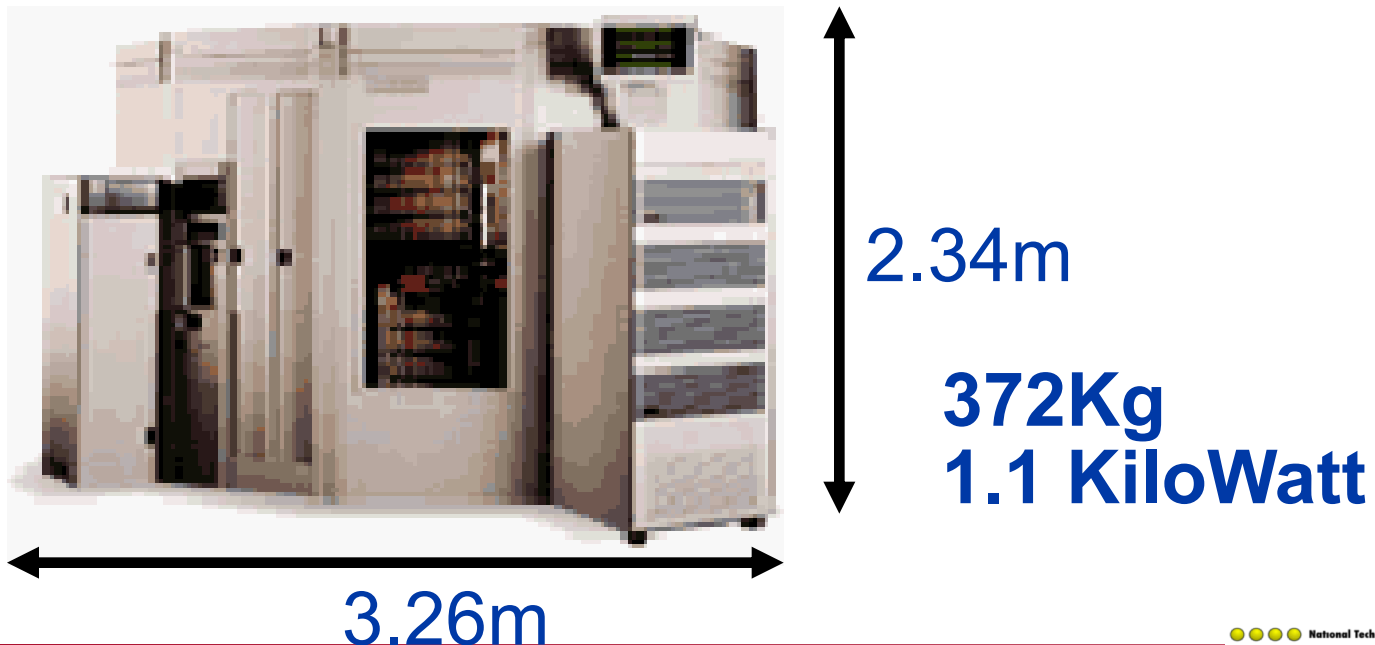
- Ιεραρχία μνήμης
- Αποθήκευσε τα πάντα στο δίσκο
- Αντίγραψε τα πρόσφατα προσπελασθέντα (και τα κοντινά τους) αντικείμενα από το δίσκο σε μια μικρότερη μνήμη DRAM
 - Κύρια μνήμη
- Αντέγραψε τα πιο πρόσφατα προσπελασθέντα (και τα κοντινά τους) αντικείμενα από τη DRAM σε μια μικρότερη μνήμη SRAM
 - Κρυφή μνήμη (cache) προσαρτημένη στη CPU

Ιεραρχία μνήμης



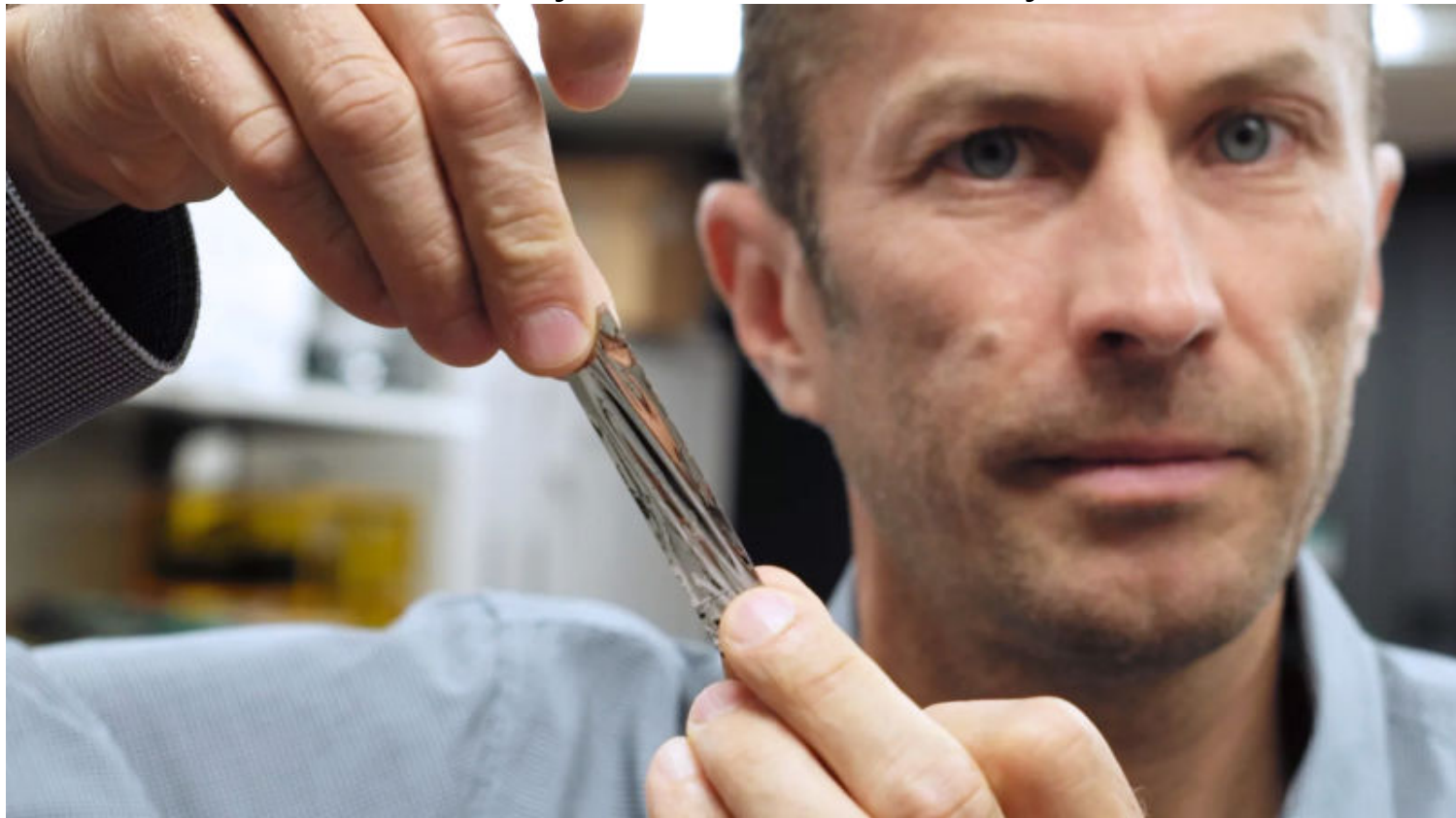
Automated Cartridge System: StorageTek Powderhorn 9310 (~2000)

- In 2000: 6000 x 50 GB 9830 tapes = 300 TBytes (uncompressed) compared to 1 hard disk ~100Gbytes
 - Library of Congress: all information in the world; in 1992, ASCII of all books = 30 TB
 - Exchange up to 450 tapes per hour (8 secs/tape)
- 1.7 to 7.7 Mbyte/sec per reader, up to 10 readers



Tapes again and again!

- Sony: 25GB/in² on July 2017
- “Why the Future of Data Storage is (Still) Magnetic Tape”, IEEE Spectrum, Aug 28, 2018 <https://tinyurl.com/ycc7drk4>
- Prediction: 91 Gb/in² by 2025, 200 Gb/in² by 2028



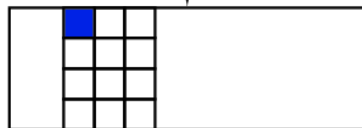
Τυπικές Αρχιτεκτονικές και caches

- **IBM Power 3 (1998):**
 - L1 = 64 KB, 128-way set associative
 - L2 = 4 MB, direct mapped, line size = 128, write back
- **Compaq EV6 (Alpha 21264):**
 - L1 = 64 KB, 2-way assoc., line size= 32
 - L2 = 4 MB (or >), direct mapped, line size = 64
- **HP PA:** no L2
 - PA8500, PA8600: L1 = 1.5 MB
 - PA8700: L1 = 2.25 MB
- **AMD Athlon:** L1 = 64 KB, L2 = 256 KB
- **Intel Pentium 4:** L1 = 8 KB, L2 = 256 KB
- **Intel Itanium:**
 - L1 = 16 KB, 4-way associative
 - L2 = 96 KB, 6-way associative
 - L3 = off chip, size varies
- **IBM Power 8 (2014):**
 - 4, 6, 8, 10 or 12 chiplets (core+L1+L2)
 - L1 = 32KB + 64 KB
 - L2 = 512KB, direct mapped, line size = 128, write back
 - L3 = #chiplets x 8MB
 - up to 1 TB of memory per socket
- **AMD Jaguar (2013):**
 - L1 = 32KB + 32KB per core
 - L2 = 1-2MB, shared by 2-4 cores
- **Intel Haswell (2013):**
 - 2-4, 6-8, 8+ cores
 - L1 = 32KB + 32KB per core
 - L2 = 256KB per core
 - L3 = 4 – 20MB

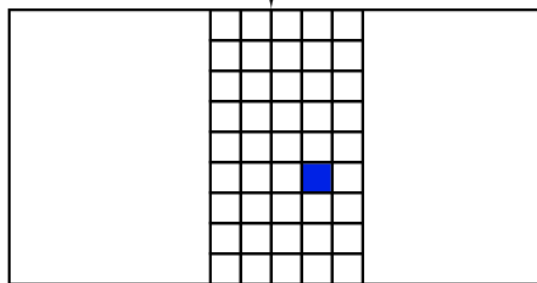
Επίπεδα ιεραρχίας μνήμης

- **Μπλοκ – block** (επίσης λέγεται γραμμή – line): μονάδα αντιγραφής
 - Μπορεί να περιέχει πολλές λέξεις
- Αν τα δεδομένα που προσπελάζονται βρίσκονται στο ανώτερο επίπεδο
 - **Ευστοχία (hit)**: προσπέλαση ικανοποιείται από το ανώτερο επίπεδο
 - Λόγος ευστοχίας (hit ratio): ευστοχίες/προσπελάσεις
- Αν τα δεδομένα που προσπελάζονται απουσιάζουν
 - **Αστοχία (miss)**: το μπλοκ αντιγράφεται από το χαμηλότερο επίπεδο
 - Απαιτούμενος χρόνος: ποινή αστοχίας (miss penalty)
 - Λόγος αστοχίας (miss ratio): αστοχίες/προσπελάσεις = $1 - \text{λόγος ευστοχίας}$
 - Στη συνέχεια τα δεδομένα που προσπελάζονται παρέχονται από το ανώτερο επίπεδο

Επεξεργαστής



Μεταφορά δεδομένων



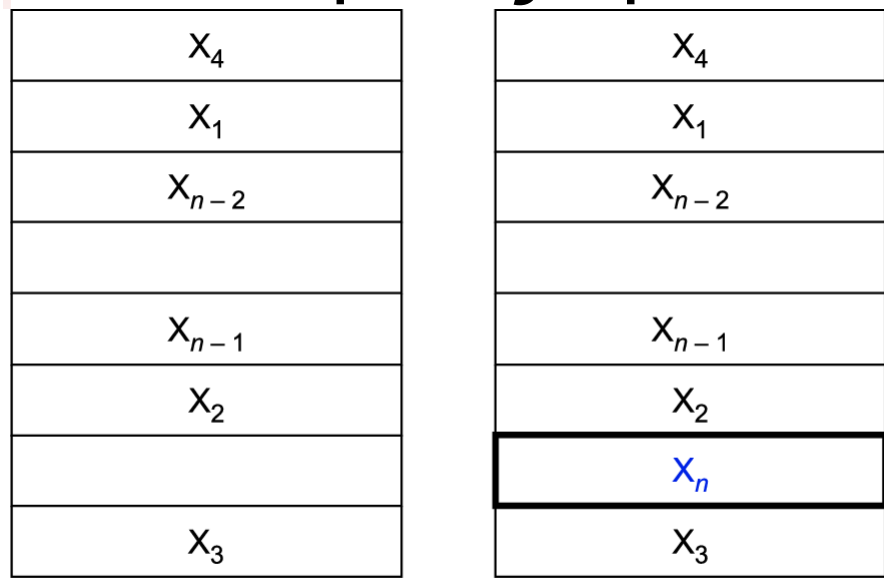
Ορολογία

- **hit** : το block βρίσκεται σε κάποια θέση του εξεταζόμενου επιπέδου μνήμης
 - **hit rate** : hits/συνολικές προσπελάσεις μνήμης
 - **hit time** : χρόνος προσπέλασης των δεδομένων
- **miss** : το block δεν υπάρχει στο εξεταζόμενο επίπεδο μνήμης
 - **miss rate** : $1 - (\text{hit rate})$
 - **miss penalty** : (χρόνος μεταφοράς των δεδομένων ενός block στο συγκεκριμένο επίπεδο μνήμης) + (χρόνος απόκτησης των δεδομένων από την CPU)
 - **access time** : χρόνος απόκτησης της 1ης λέξης
 - **transfer time** : χρόνος απόκτησης των υπόλοιπων λέξεων

Κρυφή μνήμη (cache memory)

- Κρυφή μνήμη (cache memory)
 - Το επίπεδο της ιεραρχίας μνήμης που είναι πλησιέστερα στη CPU

- Δεδομένες προσπελάσεις X_1, \dots, X_{n-1}, X_n

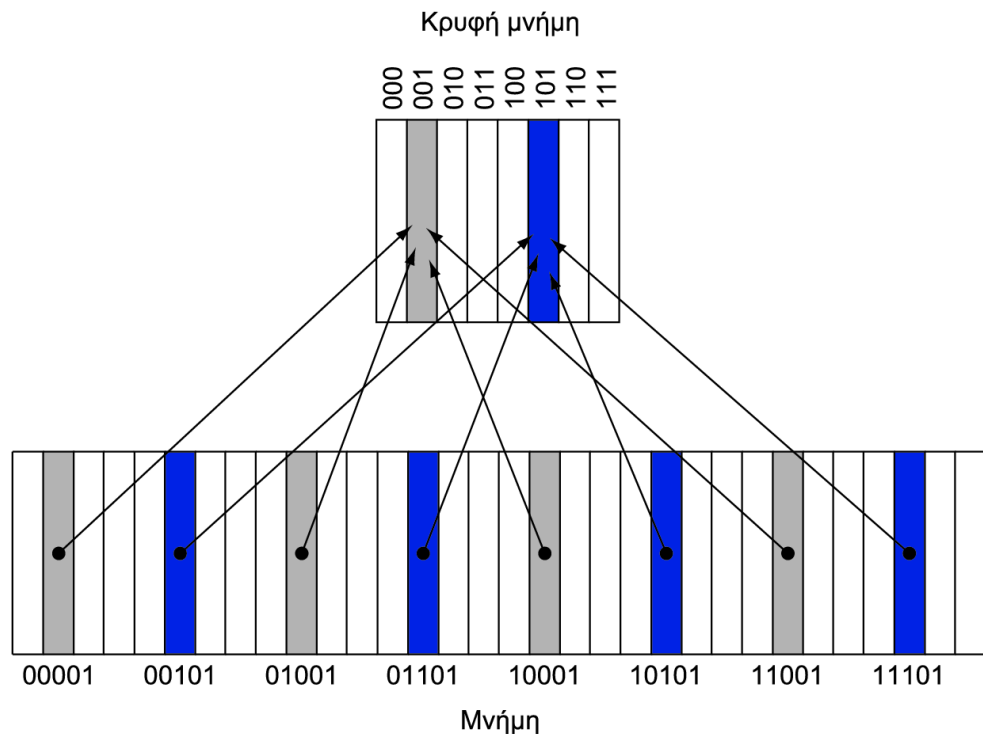


- Πώς γνωρίζουμε αν τα δεδομένα είναι παρόντα;
- Πού κοιτάζουμε;

α. Πριν από την αναφορά στο X_n β. Μετά από την αναφορά στο X_n

Κρυφή μνήμη άμεσης απεικόνισης

- Η θέση καθορίζεται από τη διεύθυνση
- Άμεση απεικόνιση (direct mapping): μόνο μία επιλογή
 - (Διεύθυνση μπλοκ) modulo (#Μπλοκ κρυφής μνήμης)



- Πλήθος μπλοκ είναι δύναμη του 2
- Χρήση των χαμηλής ταξής bit της διεύθυνσης

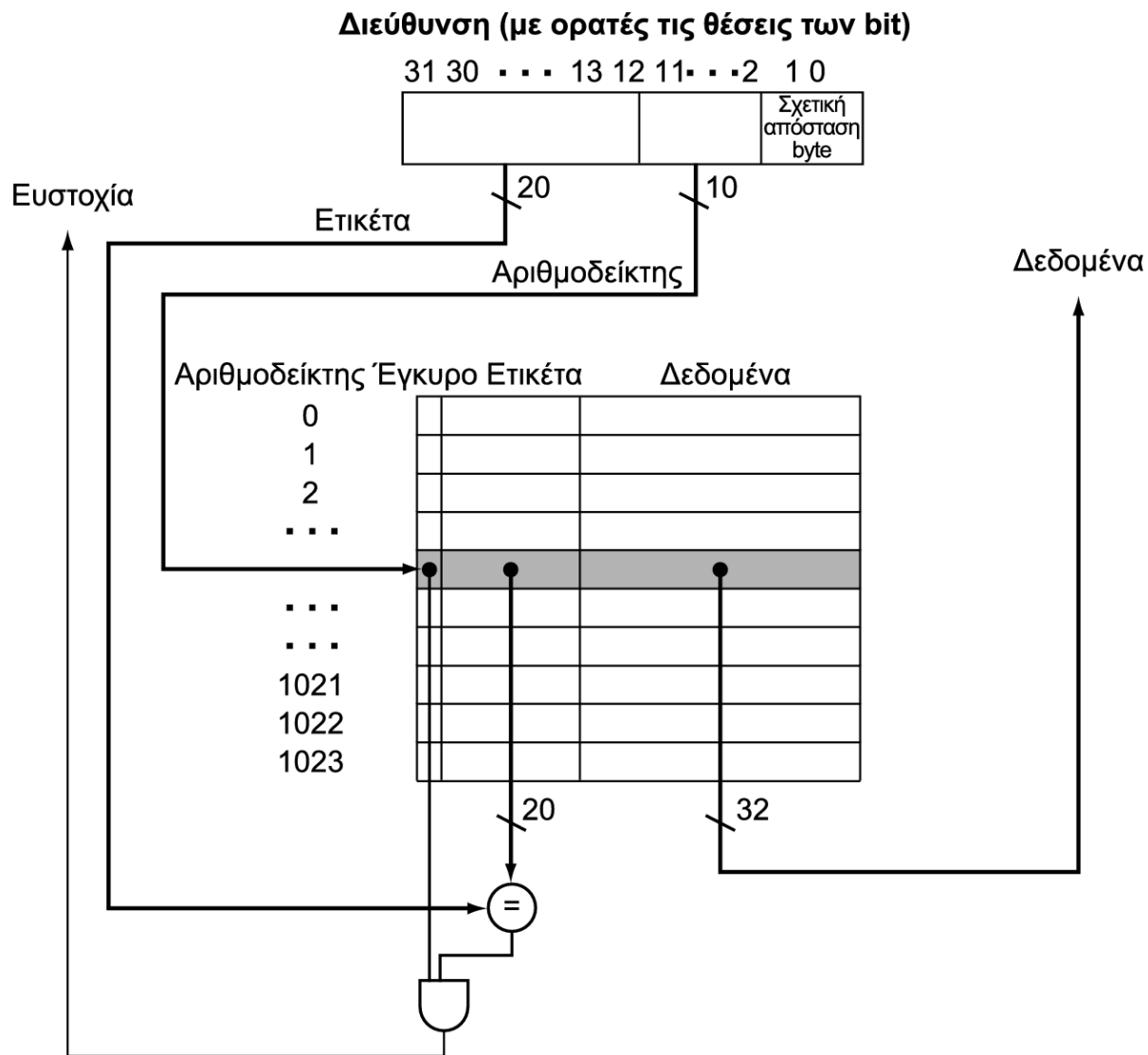
4 Ερωτήσεις για τις caches

- Πού μπορεί να τοποθετηθεί ένα block σε ένα ψηλότερο επίπεδο στην ιεραρχία μνήμης;
 - **Τοποθέτηση block :**
 - direct-mapped, fully associative, set-associative
- Πώς βρίσκουμε ένα block στα διάφορα επίπεδα μνήμης;
 - **Αναγνώριση ενός block :**
 - Tag / Block
- Ποιο από τα ήδη υπάρχοντα block της cache πρέπει να αντικατασταθεί σε περίπτωση ενός miss;
 - **Μηχανισμός αντικατάστασης block :**
 - Random, Least Recently Used (LRU), FIFO
- Τι συμβαίνει όταν μεταβάλλουμε το περιεχόμενο ενός block;
 - **μηχανισμοί εγγραφής :**
 - write-through ή write-back
 - write-allocate ή no-write-allocate

Ετικέτες και έγκυρα bit

- Πώς γνωρίζουμε ποιο συγκεκριμένο μπλοκ αποθηκεύεται σε μια θέση της κρυφής μνήμης;
 - Αποθήκευση της δ/νσης του μπλοκ μαζί με τα δεδομένα
 - Στη πραγματικότητα, χρειάζονται μόνο τα bit υψηλής τάξης
 - Ονομάζονται ετικέτα (tag)
- Και αν δεν υπάρχουν δεδομένα σε μια θέση;
 - Έγκυρο (valid) bit: 1 = παρόντα, 0 = όχι παρόντα
 - Αρχικά 0

Υποδιαίρεση της διεύθυνσης



Παράδειγμα κρυφής μνήμης

- 8 μπλοκ, 1 λέξη/μπλοκ, άμεσης απεικόνισης
- Αρχική κατάσταση

Αριθμοδείκτης	V	Ετικέτα	Δεδομένα
000	N		
001	N		
010	N		
011	N		
100	N		
101	N		
110	N		
111	N		

Παράδειγμα κρυφής μνήμης

Δ/νση λέξης	Δυαδική δ/νση	Ευστοχία/αστοχία	Μπλοκ κρυφής μνήμης
22	10 110	Αστοχία	110

Αριθμοδείκτης	V	Ετικέτα	Δεδομένα
000	N		
001	N		
010	N		
011	N		
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Παράδειγμα κρυφής μνήμης

Δ/νση λέξης	Δυαδική δ/νση	Ευστοχία/αστοχία	Μπλοκ κρυφής μνήμης
26	11 010	Αστοχία	010

Αριθμοδείκτης	V	Ετικέτα	Δεδομένα
000	N		
001	N		
010	Y	11	Mem[11010]
011	N		
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Παράδειγμα κρυφής μνήμης

Δ/νση λέξης	Δυαδική δ/νση	Ευστοχία/αστοχία	Μπλοκ κρυφής μνήμης
22	10 110	Ευστοχία	110
26	11 010	Ευστοχία	010

Αριθμοδείκτης	V	Ετικέτα	Δεδομένα
000	N		
001	N		
010	Y	11	Mem[11010]
011	N		
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Παράδειγμα κρυφής μνήμης

Δ/νση λέξης	Δυαδική δ/νση	Ευστοχία/αστοχία	Μπλοκ κρυφής μνήμης
16	10 000	Αστοχία	000
3	00 011	Αστοχία	011
16	10 000	Ευστοχία	000

Αριθμοδείκτης	V	Ετικέτα	Δεδομένα
000	Y	10	Mem[10000]
001	N		
010	Y	11	Mem[11010]
011	Y	00	Mem[00011]
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

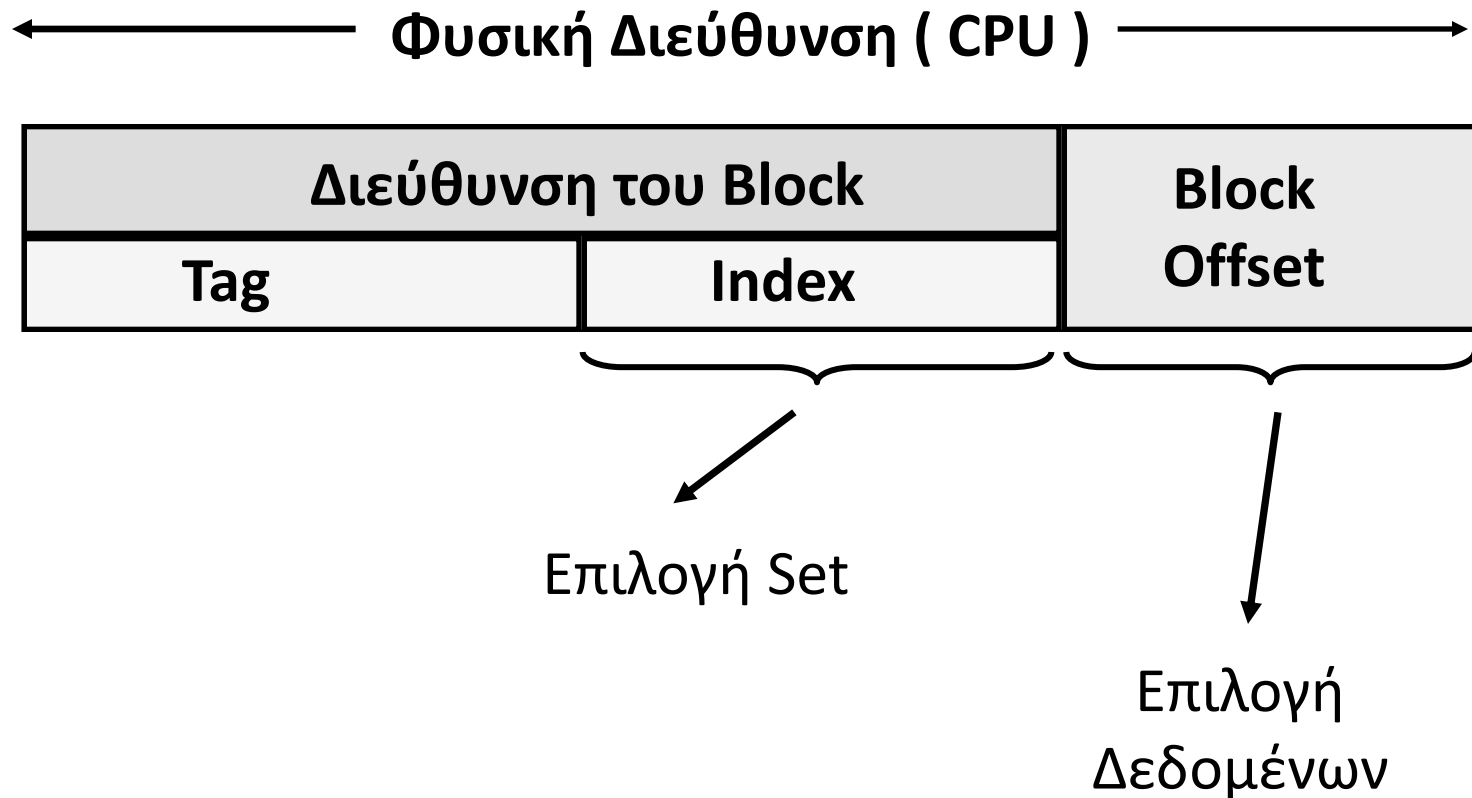
Παράδειγμα κρυφής μνήμης

Δ/νση λέξης	Δυαδική δ/νση	Ευστοχία/αστοχία	Μπλοκ κρυφής μνήμης
18	10 010	Αστοχία	010

Αντικατάσταση του παλαιού block 010 (26_{10}) με το νέο 18_{10}

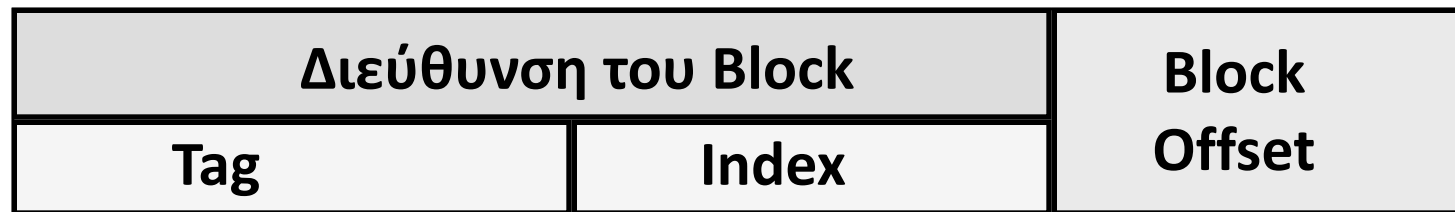
Αριθμοδείκτης	V	Ετικέτα	Δεδομένα
000	Y	10	Mem[10000]
001	N		
010	Y	10	Mem[10010]
011	Y	00	Mem[00011]
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Τα πεδία διεύθυνσης



Τα πεδία διεύθυνσης

← Φυσική Διεύθυνση (CPU) →



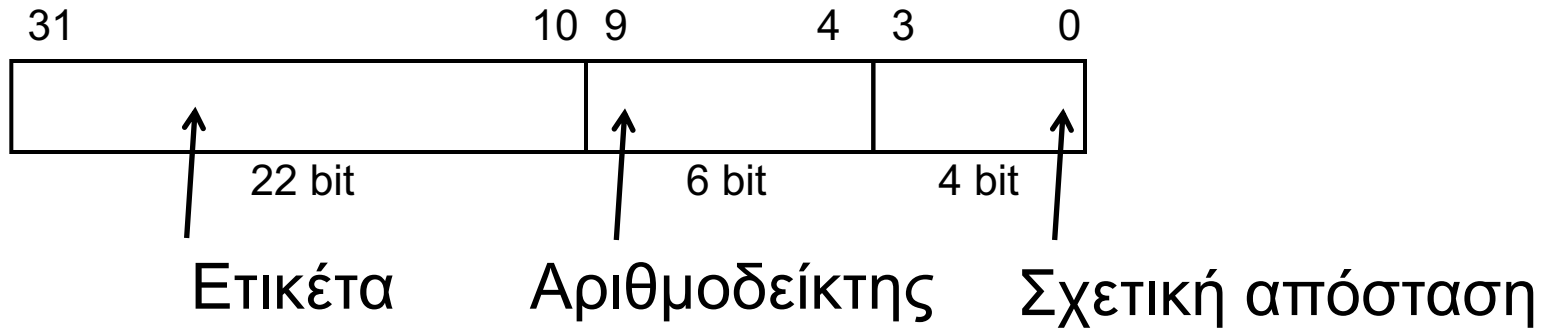
Μέγεθος block offset = $\log_2(\text{μέγεθος block})$

Μέγεθος Index = $\log_2(\text{Συνολικός αριθμός blocks/associativity})$

Μέγεθος tag = μέγεθος address - μέγεθος index - μέγεθος offset

Παράδειγμα: μεγαλύτερο μέγεθος μπλοκ

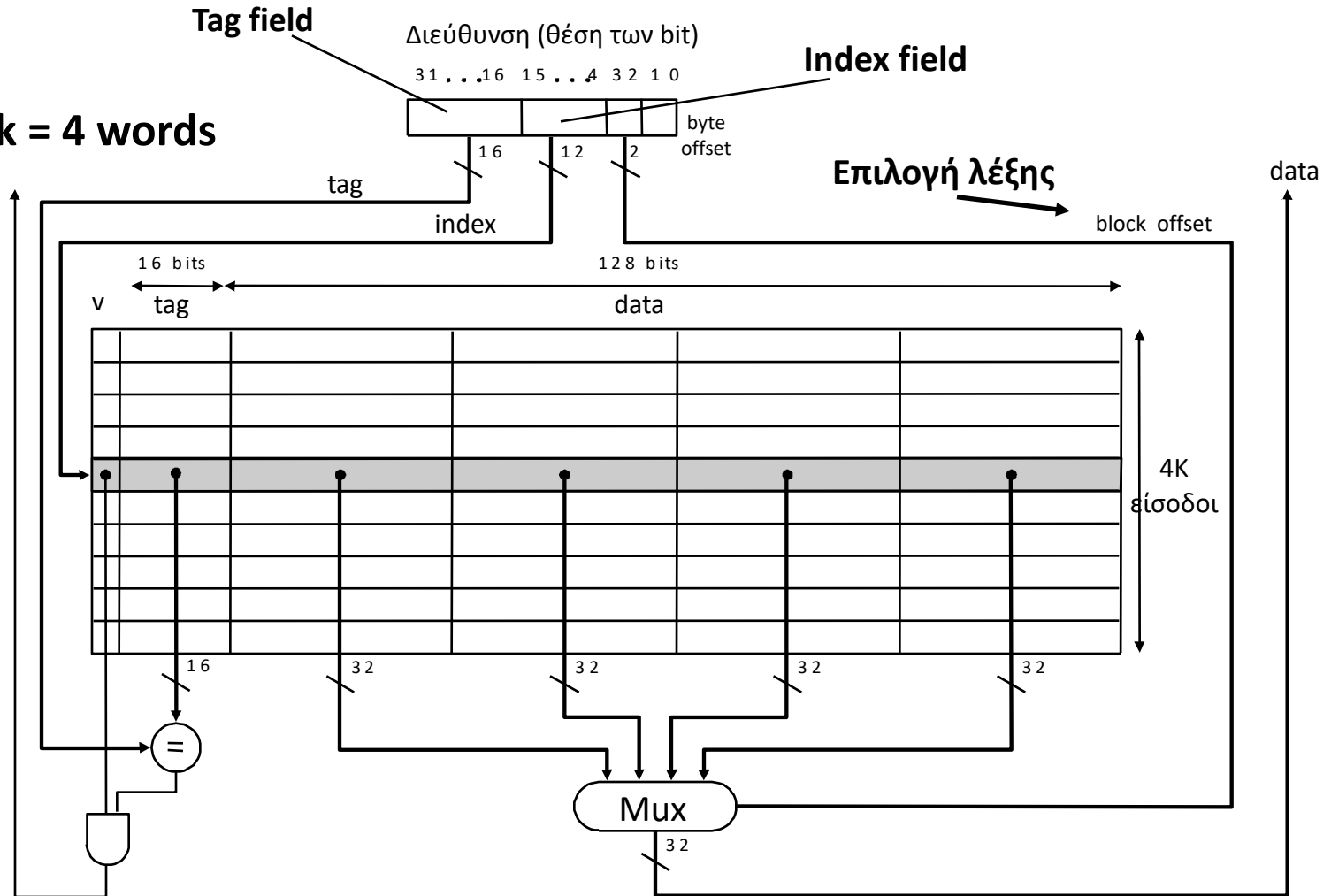
- 64 μπλοκ, 16 byte/μπλοκ
 - Σε ποιο αριθμό μπλοκ απεικονίζεται η διεύθυνση 1200;
- Διεύθυνση μπλοκ = $\lfloor 1200/16 \rfloor = 75$
- Αριθμός μπλοκ = $75 \text{ modulo } 64 = 11$



Παράδειγμα: Direct Mapped, 16B block

4K blocks

Κάθε block = 4 words



Καλύτερη αξιοποίηση της spatial locality

Ζητήματα μεγέθους μπλοκ

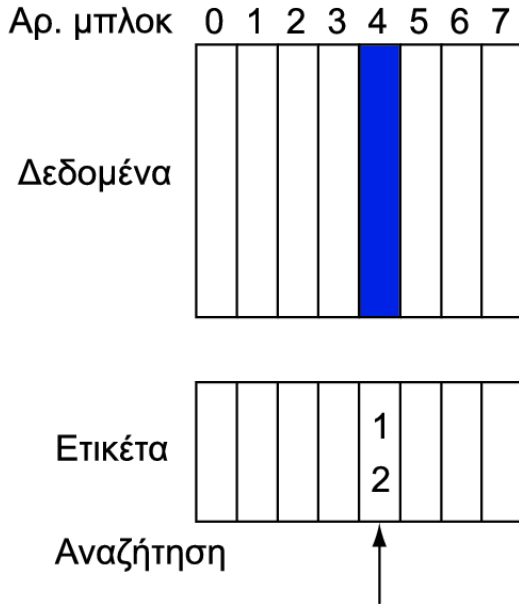
- Μεγαλύτερα μπλοκ θα μειώσουν το ρυθμό αστοχίας
 - Εκμετάλλευση χωρικής τοπικότητας
- Αλλά σε κρυφή μνήμη σταθερού μεγέθους
 - Μεγαλύτερα μπλοκ \Rightarrow λιγότερα μπλοκ
 - Περισσότερος ανταγωνισμός \Rightarrow αυξημένος ρυθμός αστοχίας
 - Μεγαλύτερα μπλοκ \Rightarrow «μόλυνση» (pollution)
- Μεγαλύτερη ποινή αστοχίας
 - Μπορεί να ξεπεράσει το όφελος του μειωμένου ρυθμού αστοχίας
 - Η πρόωρη επανεκκίνηση (early restart) και η πολιτική «κρίσιμη λέξη πρώτα» (critical-word-first) βοηθούν

Συσχετιστικές κρυφές μνήμες

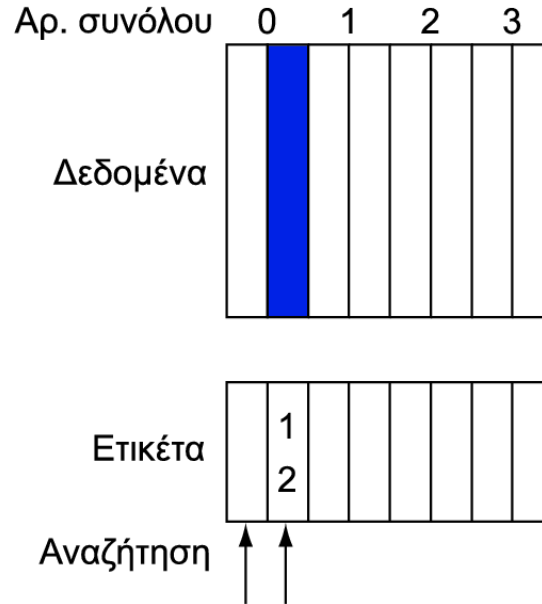
- Πλήρως συσχετιστική (fully associative)
 - Κάθε μπλοκ μπορεί να πάει σε οποιαδήποτε καταχώριση της κρυφής μνήμης
 - Απαιτεί ταυτόχρονη αναζήτηση όλων των καταχωρίσεων
 - Συγκριτής σε κάθε καταχώριση (ακριβό)
- Συσχετιστική συνόλου n δρόμων (n -way set associative)
 - Κάθε σύνολο περιέχει n καταχωρίσεις
 - Ο αριθμός μπλοκ καθορίζει το σύνολο
 - (Αριθμός μπλοκ) modulo (#Συνόλων στη κρυφή μνήμη)
 - Ταυτόχρονη αναζήτηση όλων των καταχωρίσεων ενός δεδομένου συνόλου
 - n συγκριτές (λιγότερο ακριβό)

Παράδειγμα συσχετιστικής κρυφής μνήμης

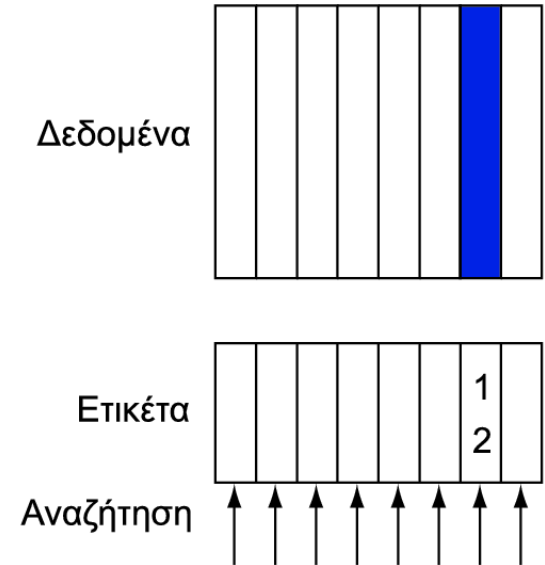
Άμεσης απεικόνισης



Συσχετιστική συνόλου



Πλήρως συσχετιστική



Οργάνωση Set Associative Cache

1-way set associative (direct mapped)

block	Tag	Data
0		
1		
2		
3		
4		
5		
6		
7		

2-way set associative

Set	Tag	Data	Tag	Data
0				
1				
2				
3				

4-way set associative

Set	Tag	Data	Tag	Data	Tag	Data	Tag	Data
0								
1								

8-way set associative (fully associative)

Tag	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data

Χωρητικότητα
cache : 8 words

Παράδειγμα συσχετιστικότητας

- Σύγκριση κρυφών μνημών με 4 μπλοκ
 - Άμεσης απεικόνισης, συσχετιστική συνόλου 2 δρόμων, πλήρως συσχετιστική
 - Ακολουθία προσπελάσεων μπλοκ: 0, 8, 0, 6, 8
- Άμεσης απεικόνισης

Δ/νση μπλοκ	Αριθμοδείκτης κρυφής μνήμης	Ευστοχία /αστοχία	Περιεχόμενα κρυφής μνήμης μετά την προσπέλαση			
			0	1	2	3
0	0	miss	Mem[0]			
8	0	miss	Mem[8]			
0	0	miss	Mem[0]			
6	2	miss	Mem[0]		Mem[6]	
8	0	miss	Mem[8]		Mem[6]	

Παράδειγμα συσχετιστικότητας

- Συσχετιστική συνόλου 2 δρόμων

Δ/νση μπλοκ	Αριθμο-δείκτης κρυφής μνήμης	Ευστοχία/αστοχία	Περιεχόμενα κρυφής μνήμης μετά την προσπέλαση			
			Set 0		Set 1	
0	0	miss	Mem[0]			
8	0	miss	Mem[0]	Mem[8]		
0	0	hit	Mem[0]	Mem[8]		
6	0	miss	Mem[0]	Mem[6]		
8	0	miss	Mem[8]	Mem[6]		

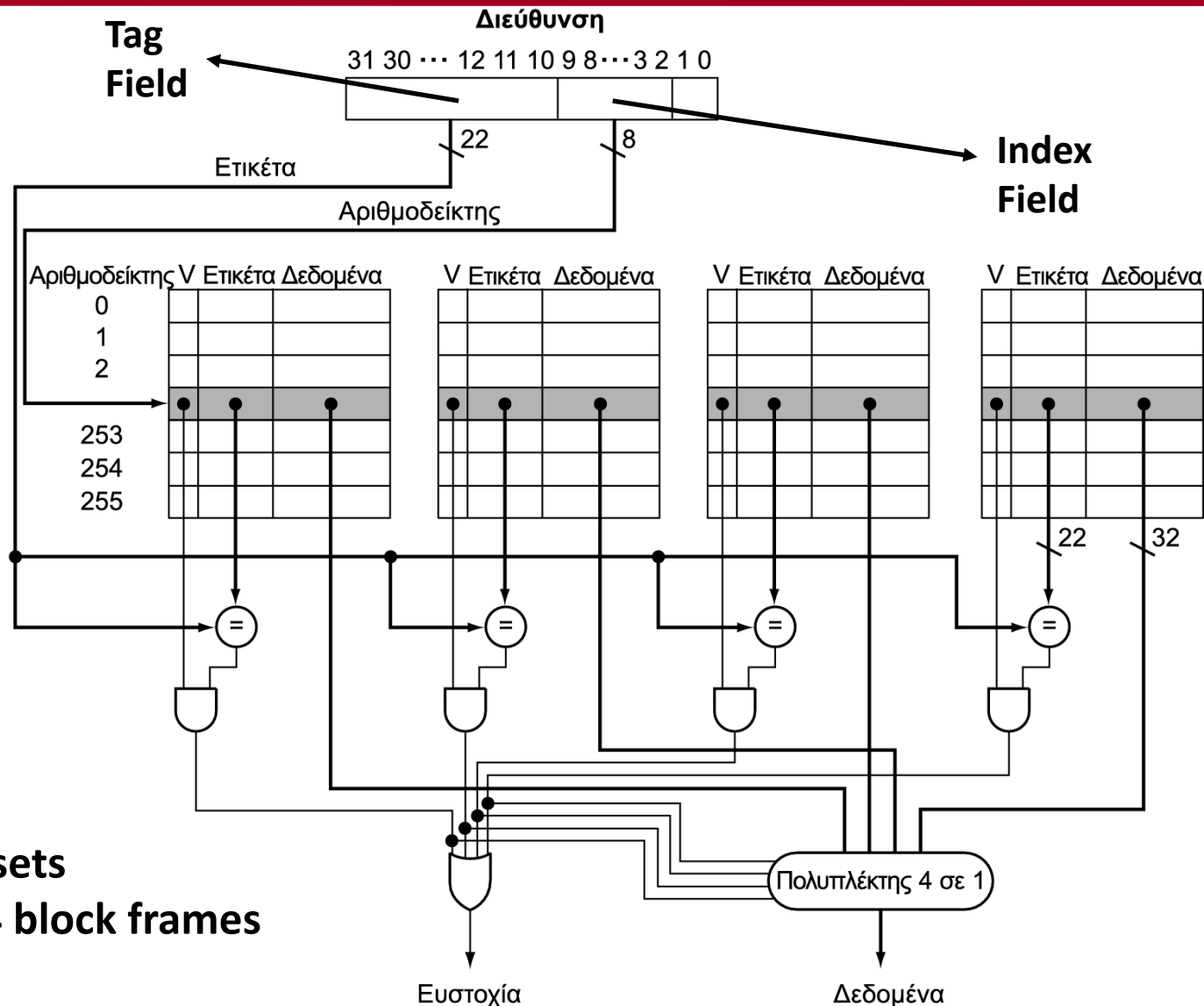
- Πλήρως συσχετιστική

Δ/νση μπλοκ		Ευστοχία/αστοχία	Περιεχόμενα κρυφής μνήμης μετά την προσπέλαση			
0		miss	Mem[0]			
8		miss	Mem[0]	Mem[8]		
0		hit	Mem[0]	Mem[8]		
6		miss	Mem[0]	Mem[8]	Mem[6]	
8		hit	Mem[0]	Mem[8]	Mem[6]	

Πόση συσχετιστικότητα;

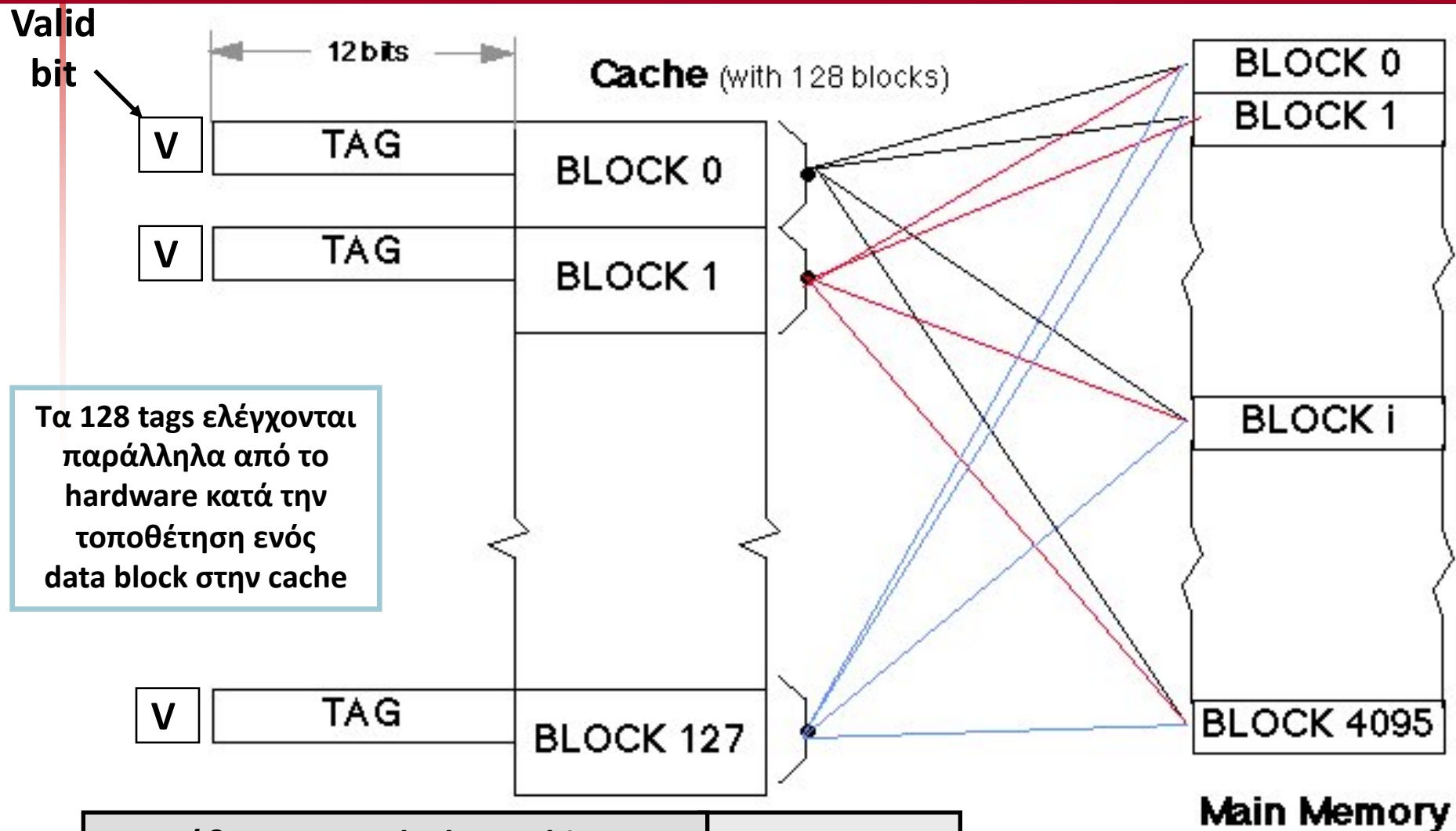
- Αυξημένη συσχετιστικότητα μειώνει το ρυθμό αστοχίας
 - Αλλά με μειούμενα οφέλη όσο αυξάνεται
- Προσομοίωση συστήματος με κρυφή μνήμη δεδομένων (D-cache) 64KB, μπλοκ των 16 λέξεων, μετροπρ/τα SPEC2000
 - 1 δρόμου: 10.3%
 - 2 δρόμων: 8.6%
 - 4 δρόμων: 8.3%
 - 8 δρόμων: 8.1%

Κρυφή μνήμη συσχειστικής συνόλου (4-Way Set Associative Cache)



256 sets
1024 block frames

Fully Associative Case



Διεύθυνση του Block = 12bits	Block Offset = 4bits
Tag = 12bits	

Αστοχίες κρυφής μνήμης

- Σε περίπτωση ευστοχίας, η CPU συνεχίζει κανονικά
- Σε περίπτωση αστοχίας
 - Καθυστερεί η διοχέτευση της CPU
 - Προσκομίζει το μπλοκ από το επόμενο επίπεδο της ιεραρχίας
 - Αστοχία κρυφής μνήμης εντολών
 - Επανεκκίνηση προσκόμισης εντολής
 - Αστοχία κρυφής μνήμης δεδομένων
 - Ολοκλήρωση προσπέλασης δεδομένων

Πολιτική αντικατάστασης

- Άμεσης απεικόνισης: καμία επιλογή
- Συσχετιστική συνόλου
 - Προτίμησε τη μη έγκυρη καταχώριση, αν υπάρχει μία
 - Αλλιώς, διάλεξε ανάμεσα στις καταχωρίσεις του συνόλου
- Λιγότερο πρόσφατα χρησιμοποιημένα (Least-recently used – LRU)
 - Διάλεξε αυτή που δε χρησιμοποιήθηκε για το μεγαλύτερο διάστημα
 - Απλή για 2δρόμων, διαχειρίσιμη για 4δρόμων, υπερβολικά δύσκολη από εκεί και πέρα
- Τυχαία
 - Δίνει περίπου την ίδια απόδοση με την LRU για μεγάλη συσχετιστικότητα

Μηχανισμοί αντικατάστασης block

- Random (τυχαία) – επιλογή ενός τυχαίου block με βάση κάποια ψευδοτυχαία ακολουθία
 - απλή υλοποίηση στο hardware
- LRU (least recently used) – αντικαθίσταται το block που δεν έχει χρησιμοποιηθεί για περισσότερη ώρα
 - Πολύ καλές επιδόσεις (μικρό miss rate)
 - Ακριβή υλοποίηση στο hardware
 - Χρησιμοποιείται συνήθως (πλήρης ή κατά προσέγγιση – Pseudo-LRU)
- FIFO (first in - first out) - αντικαθίσταται το block που έχει εισαχθεί πρώτο στην cache (εύκολο στην υλοποίηση)

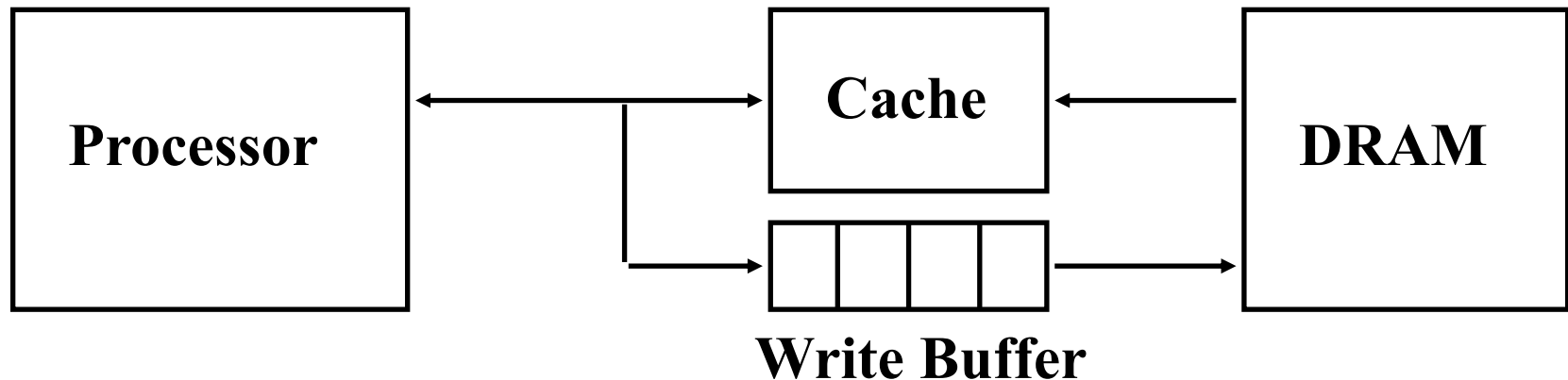
Ταυτόχρονη εγγραφή (write-through)

- Σε ευστοχία εγγραφής δεδομένων, θα μπορούσε να γίνει μόνο ενημέρωση του μπλοκ στην κρυφή μνήμη
 - Αλλά τότε η κρυφή μνήμη και η μνήμη θα είναι ασυνεπείς
- Ταυτόχρονη εγγραφή (write through): ενημέρωσε και τη μνήμη
- Αποτέλεσμα: οι εγγραφές να διαρκούν περισσότερο
 - π.χ., αν το βασικό CPI είναι 1, το 10% των εντολών είναι αποθηκεύσεις, και η εγγραφή στη μνήμη διαρκεί 100 κύκλους
 - Πραγματικό CPI = $1 + 0.1 \times 100 = 11$
- Λύση: προσωρινή μνήμη εγγραφής (*write buffer*)
 - Κρατά δεδομένα που περιμένουν να γραφούν στη μνήμη
 - Η CPU συνεχίζει αμέσως
 - Καθυστερεί στην εγγραφή μόνο αν η προσωρινή μνήμη εγγραφής είναι ήδη γεμάτη

Ετερόχρονη εγγραφή (write-back)

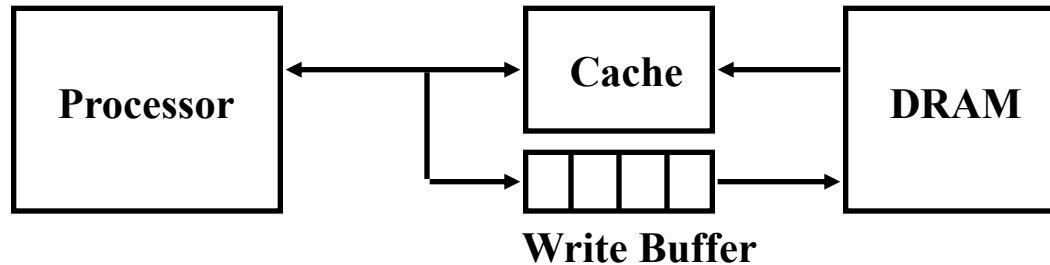
- Write back
- Εναλλακτική: σε ευστοχία εγγραφής δεδομένων, ενημέρωσε μόνο το μπλοκ στην κρυφή μνήμη
 - Διάκριση «καθαρών» (clean – δεν έχουν γραφεί) και «ακάθαρτων» (dirty – έχουν γραφεί) μπλοκ
- Όταν ένα ακάθαρμο μπλοκ αντικαθίσταται
 - Γράψε το πίσω στη μνήμη
 - Μπορεί να χρησιμοποιήσει μια προσωρινή μνήμη εγγραφής ώστε να αντικατασταθεί το μπλοκ που θα διαβαστεί πρώτο

Write Buffer for Write Through

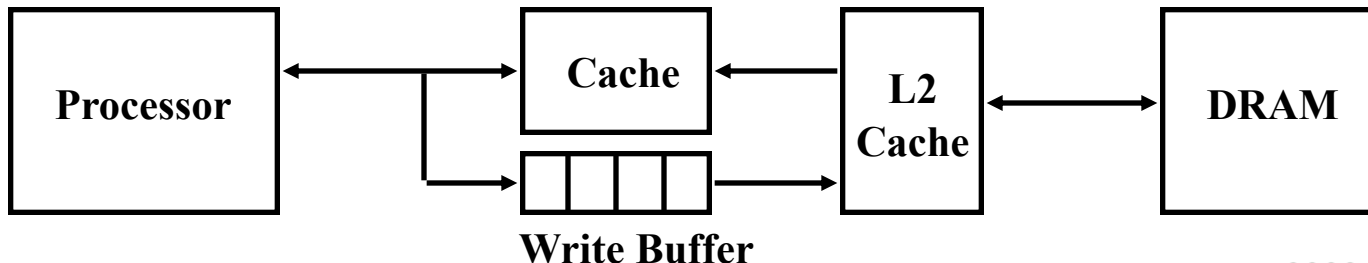


- A Write Buffer is needed between the Cache and Memory
 - Processor: writes data into the cache and the write buffer
 - Memory controller: write contents of the buffer to memory
- Write buffer is just a FIFO:
 - Typical number of entries: 4
 - Must handle bursts of writes
 - Works fine if: Store frequency (w.r.t. time) $\ll 1 / \text{DRAM write cycle}$

Write Buffer Saturation



- Store frequency (w.r.t. time) $> 1 / \text{DRAM write cycle}$
 - If this condition exist for a long period of time (CPU cycle time too quick and/or too many store instructions in a row):
 - Store buffer will overflow no matter how big you make it
 - The CPU Cycle Time \leq DRAM Write Cycle Time
- Solution for write buffer saturation:
 - Use a write back cache
 - Install a second level (L2) cache: (does this always work?)



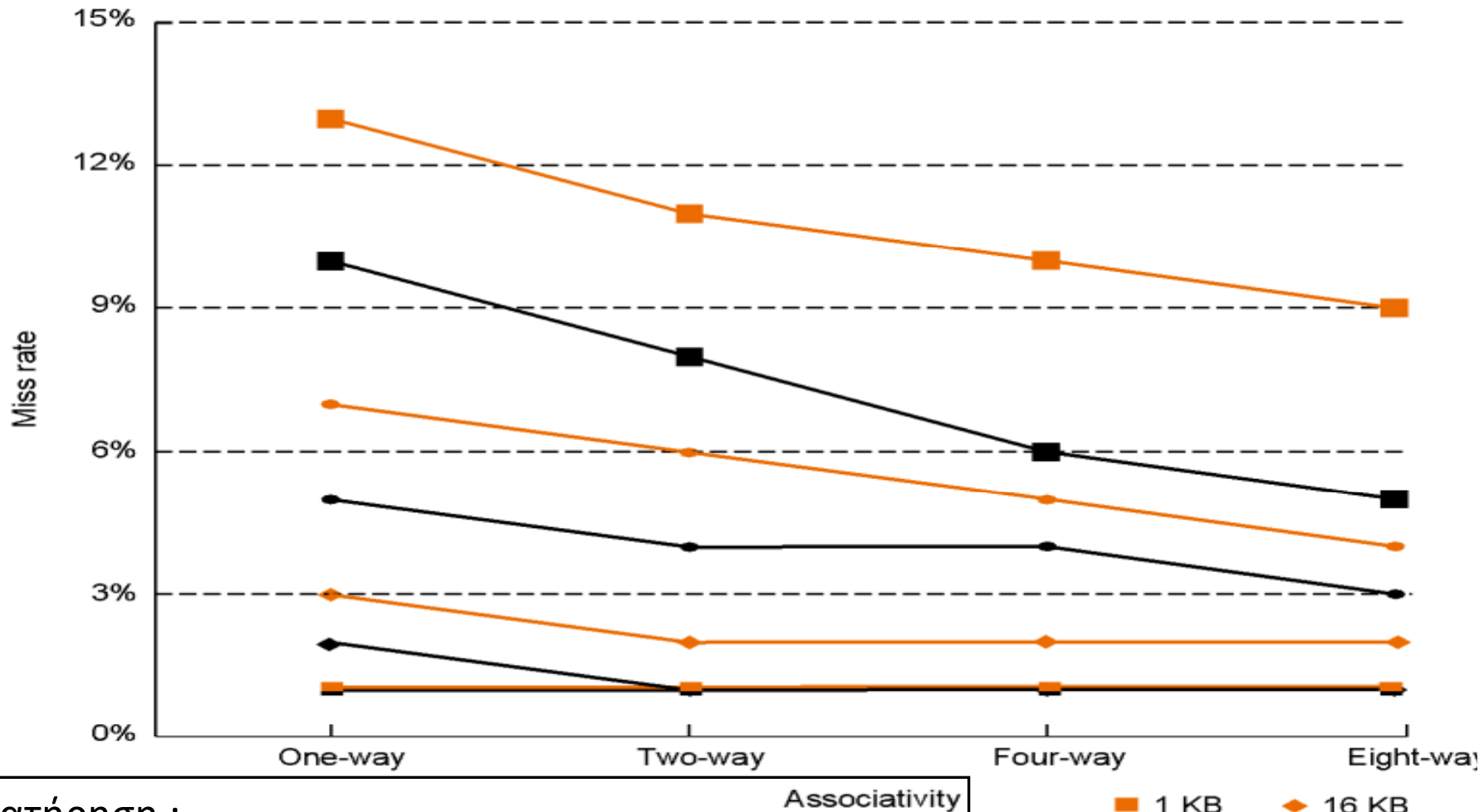
RAW Hazards from Write Buffer!

- Write-Buffer Issues: Could introduce RAW Hazard with memory!
 - Write buffer may contain only copy of valid data \Rightarrow Reads to memory may get wrong result if we ignore write buffer
- Solutions:
 - Simply wait for write buffer to empty before servicing reads:
 - Might increase read miss penalty (old MIPS 1000 by 50%)
 - Check write buffer contents before read (“fully associative”);
 - If no conflicts, let the memory access continue
 - Else grab data from buffer
- Can Write Buffer help with Write Back?
 - Read miss replacing dirty block
 - Copy dirty block to write buffer while starting read to memory
 - CPU stall less since restarts as soon as do read

Κατανομή εγγραφών (Write allocation)

- Τι πρέπει να γίνει σε αστοχία εγγραφής;
- Εναλλακτικές για ταυτόχρονη εγγραφή
 - Κατανομή σε αστοχία (allocate on miss): προσκόμιση του μπλοκ
 - Εγγραφή από γύρω (write around): όχι προσκόμιση του μπλοκ
 - Αφού τα προγράμματα συχνά γράφουν ένα ολόκληρο μπλοκ πριν το διαβάσουν (π.χ., απόδοση αρχικών τιμών)
- Για την ετερόχρονη εγγραφή
 - Συνήθως προσκομίζεται το μπλοκ

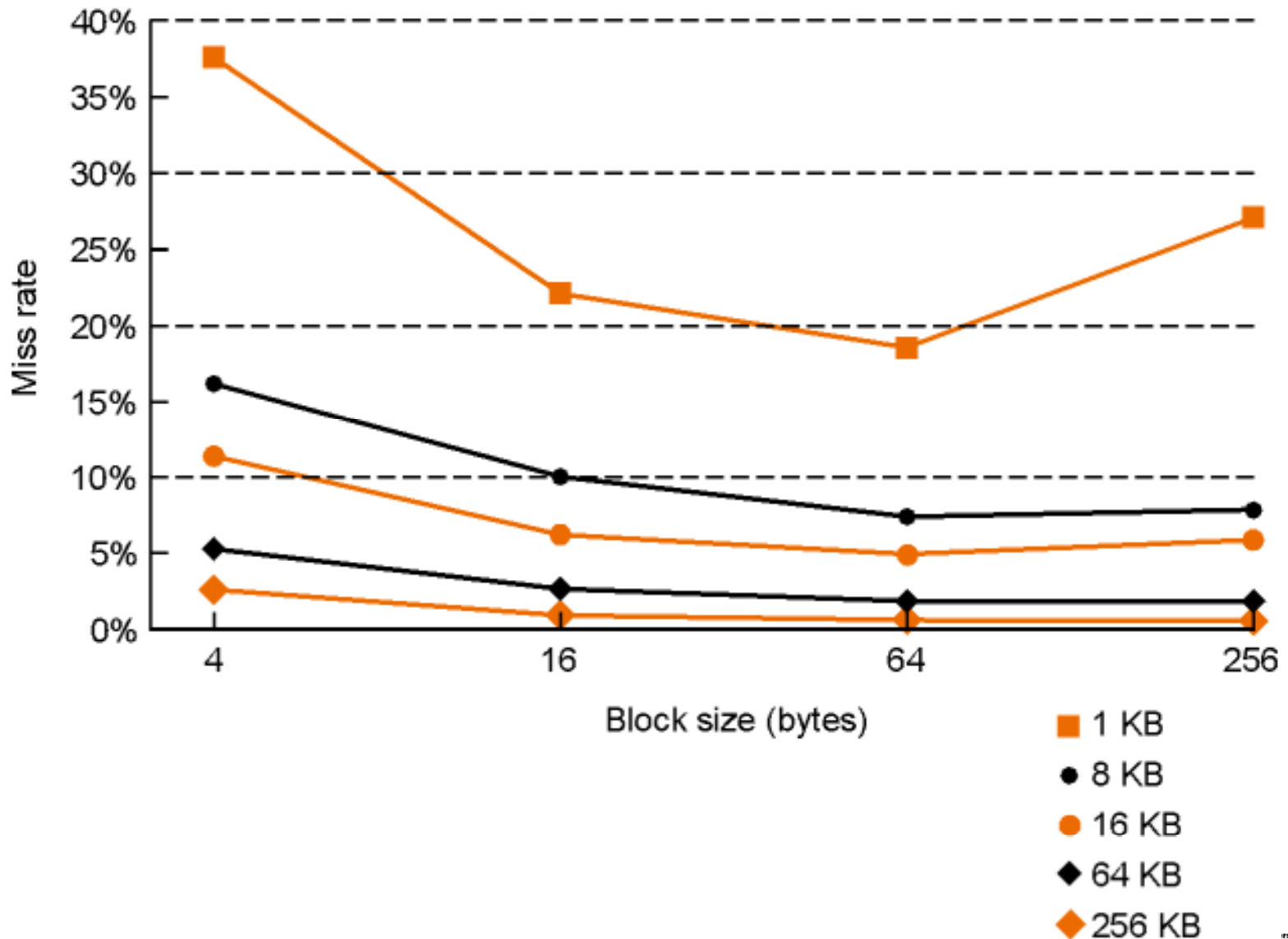
Cache Associativity



Παρατήρηση :

Μια 4-way cache έχει σχεδόν το ίδιο hit rate με μια direct-mapped cache διπλάσιου μεγέθους

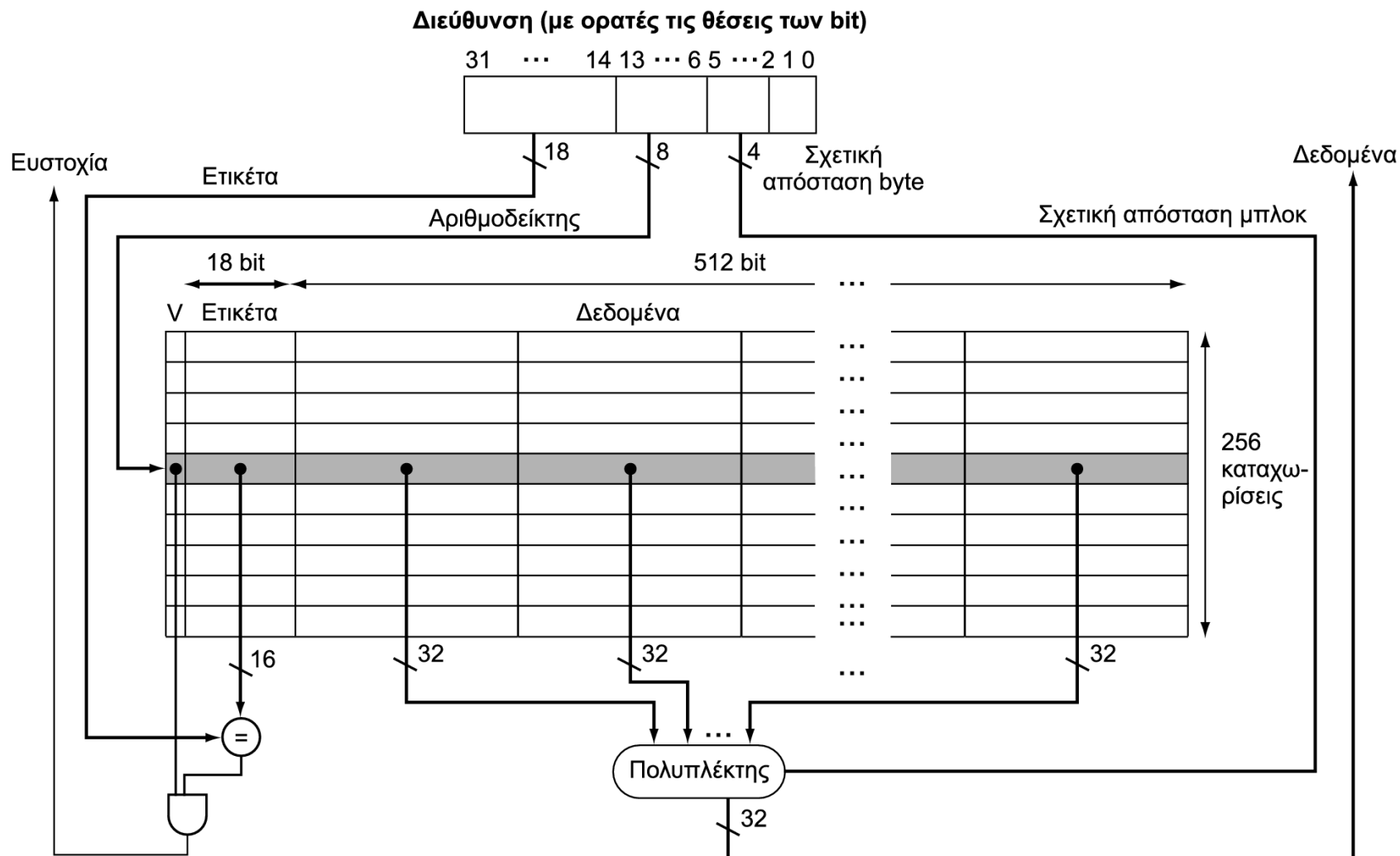
Μέγεθος Block και Miss Rate



Παράδειγμα: Intrinsicity FastMATH

- Ενσωματωμένος επεξεργαστής MIPS
 - Διοχέτευση 12 σταδίων
 - Προσπέλαση εντολής και δεδομένου σε κάθε κύκλο
- Διαιρεμένη (split) κρυφή μνήμη: ξεχωριστή I-cache και D-cache
 - Η κάθε μία των 16KB: 256 μπλοκ × 16 λέξεις ανά μπλοκ
 - D-cache: ταυτόχρονη ή ετερόχρονη εγγραφή
- Ρυθμοί αστοχίας SPEC2000
 - I-cache: 0.4%
 - D-cache: 11.4%
 - Σταθμισμένος μέσος όρος: 3.2%

Παράδειγμα: Intrinsic FastMATH



Μέτρηση απόδοσης κρυφής μνήμης

- Συστατικά του χρόνου CPU
 - Κύκλοι εκτέλεσης προγράμματος
 - Περιλαμβάνει το χρόνο ευστοχίας κρυφής μνήμης
 - Κύκλοι καθυστέρησης (stall) μνήμης
 - Κυρίως από αστοχίες κρυφής μνήμης
- Με απλουστευτικές παραδοχές:

Memory stall cycles

$$= \frac{\text{Memory accesses}}{\text{Program}} \times \text{Miss rate} \times \text{Miss penalty}$$

$$= \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Misses}}{\text{Instruction}} \times \text{Miss penalty}$$

Παράδειγμα απόδοσης κρυφής μνήμης

- Δίνονται
 - Ρυθμός αστοχίας κρυφής μνήμης εντολών (I-cache) = 2%
 - Ρυθμός αστοχίας κρυφής μνήμης δεδομένων (D-cache) = 4%
 - Ποινή αστοχίας = 100 κύκλοι
 - Βασικό CPI (ιδανική κρυφή μνήμη) = 2
 - Οι εντολές load & store είναι το 36% των εντολών
- Κύκλοι αστοχίας ανά εντολή
 - I-cache: $0.02 \times 100 = 2$
 - D-cache: $0.36 \times 0.04 \times 100 = 1.44$
- Πραγματικό CPI = $2 + 2 + 1.44 = 5.44$
 - Η ιδανική CPU είναι $5.44/2 = 2.72$ φορές ταχύτερη

Μέσος χρόνος προσπέλασης

- Ο χρόνος ευστοχίας είναι επίσης σημαντικός για την απόδοση
- Μέσος χρόνος προσπέλασης μνήμης (Average memory access time – AMAT)
 - $AMAT = \text{Χρόνος ευστοχίας} + \text{Ρυθμός αστοχίας} \times \text{Ποινή αστοχίας}$
- Παράδειγμα
 - CPU με ρολόι του 1 ns, χρόνο ευστοχίας = 1 κύκλος, ποινή αστοχίας = 20 κύκλοι, ρυθμός αστοχίας I-cache = 5%
 - $AMAT = 1 + 0.05 \times 20 = 2ns$
 - 2 κύκλοι ανά εντολή

Περίληψη της απόδοσης

- Όταν αυξάνει η απόδοση της CPU
 - Η ποινή αστοχίας γίνεται πιο σημαντική
- Μείωση του βασικού CPI
 - Μεγαλύτερο ποσοστό του χρόνου δαπανάται σε καθυστερήσεις μνήμης
- Αύξηση του ρυθμού ρολογιού
 - Οι καθυστερήσεις μνήμης (σταθερού χρόνου) αποτελούν περισσότερους κύκλους CPU
- Δεν μπορούμε να αγνοήσουμε τη συμπεριφορά της κρυφής μνήμης όταν αξιολογούμε την απόδοση του συστήματος

Πολυεπίπεδες κρυφές μνήμες

- Κύρια κρυφή μνήμη (L-1) συνδέεται με τη CPU
 - Μικρή, αλλά γρήγορη
- Η κρυφή μνήμη δευτέρου επιπέδου (level-2 cache) εξυπηρετεί αστοχίες της κύριας κρυφής μνήμης
 - Μεγαλύτερη, πιο αργή, αλλά και πάλι ταχύτερη από τη κύρια μνήμη
- Η κύρια μνήμη εξυπηρετεί αστοχίες της κρυφής μνήμης L-2
- Μερικά/πολλά συστήματα υψηλών επιδόσεων περιλαμβάνουν και κρυφή μνήμη L-3

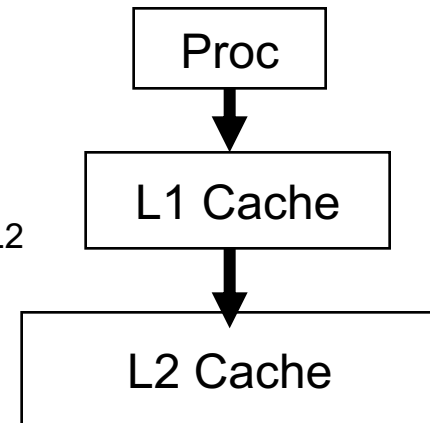
Multi-Level Caches

■ L2 Equations

$$\text{AMAT} = \text{Hit Time}_{L1} + \text{Miss Rate}_{L1} \times \text{Miss Penalty}_{L1}$$

$$\text{Miss Penalty}_{L1} = \text{Hit Time}_{L2} + \text{Miss Rate}_{L2} \times \text{Miss Penalty}_{L2}$$

$$\text{AMAT} = \text{Hit Time}_{L1} + \text{Miss Rate}_{L1} \times (\text{Hit Time}_{L2} + \text{Miss Rate}_{L2} \times \text{Miss Penalty}_{L2})$$



■ Definitions:

- **Local miss rate** — misses in this cache divided by the total number of memory accesses to this cache (Miss rate_{L2})
- **Global miss rate** — misses in this cache divided by the total number of memory accesses generated by the CPU (Miss Rate_{L1} x Miss Rate_{L2})
- Global Miss Rate is what matters

Παράδειγμα πολυεπίπεδης κρυφής μνήμης

- Δίνονται
 - Βασικό CPU CPI = 1, ρυθμός ρολογιού = 4GHz
 - Ρυθμός ρολογιού ανά εντολή = 2%
 - Χρόνος προσπέλασης κύριας μνήμης = 100ns
- Μόνο με μία κύρια κρυφή μνήμη (L-1)
 - Ποινή αστοχίας = $100\text{ns}/0.25\text{ns} = 400$ κύκλοι
 - Πραγματικό CPI = $1 + 0.02 \times 400 = 9$

Παράδειγμα (συνεχ.)

- Τώρα προσθέτουμε και κρυφή μνήμη L-2
 - Χρόνος προσπέλασης = 5ns
 - Καθολικός ρυθμός αστοχίας προς την κύρια μνήμη = 0.5%
- Αστοχία στην L-1 και ευστοχία στην L-2
 - Ποινή = $5\text{ns}/0.25\text{ns} = 20$ κύκλοι
- Αστοχία και στην L-1 και στην L-2
 - Επιπλέον ποινή = 500 κύκλοι
- $\text{CPI} = 1 + 0.02 \times 20 + 0.005 \times 400 = 3.4$
- Λόγος απόδοσης = $9/3.4 = 2.6$

Ζητήματα πολυεπίπεδων κρυφών μηνμών

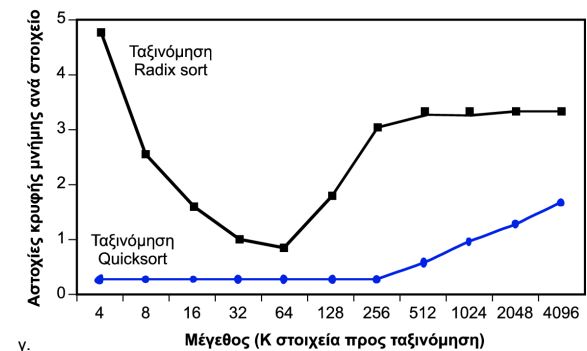
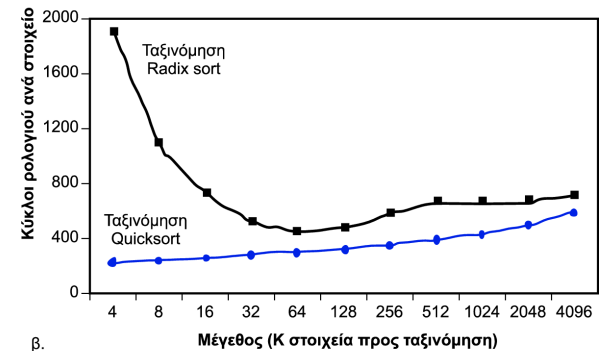
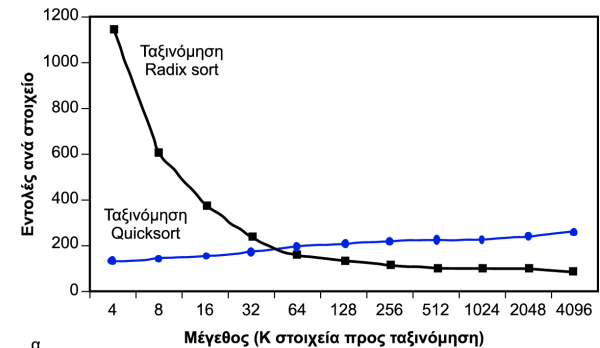
- Κύρια κρυφή μνήμη L-1
 - Εστιάζει στον ελάχιστο χρόνο ευστοχίας
- Κρυφή μνήμη L-2
 - Εστιάζει στο χαμηλό ρυθμό αστοχίας για να αποφύγει τις προσπελάσεις της κύριας μνήμης
 - Ο χρόνος ευστοχίας έχει μικρότερη συνολική επίδραση
- Αποτελέσματα
 - Η κρυφή μνήμη L-1 είναι συνήθως μικρότερη από την περίπτωση μίας μοναδικής κρυφής μνήμης
 - Το Associativity της L-2 είναι μεγαλύτερο!
 - Το μέγεθος μπλοκ της L-1 είναι μικρότερο από το μέγεθος μπλοκ της L-2

Αλληλεπιδράσεις με προηγμένες CPU

- Οι εκτός σειράς (out-of-order) CPU μπορούν να εκτελούν εντολές κατά τη διάρκεια αστοχίας κρυφής μνήμης
 - Η εκκρεμής εντολή store παραμένει στη μονάδα φόρτωσης/αποθήκευσης (load/store unit)
 - Οι αλληλεξαρτώμενες εντολές περιμένουν στους σταθμούς κράτησης (reservation stations)
 - Οι ανεξάρτητες εντολές συνεχίζουν
- Η επίδραση της αστοχίας εξαρτάται από τη ροή δεδομένων του προγράμματος (data flow)
 - Πολύ δυσκολότερη η ανάλυση
 - Χρήση προσομοίωσης συστήματος

Αλληλεπιδράσεις με το λογισμικό

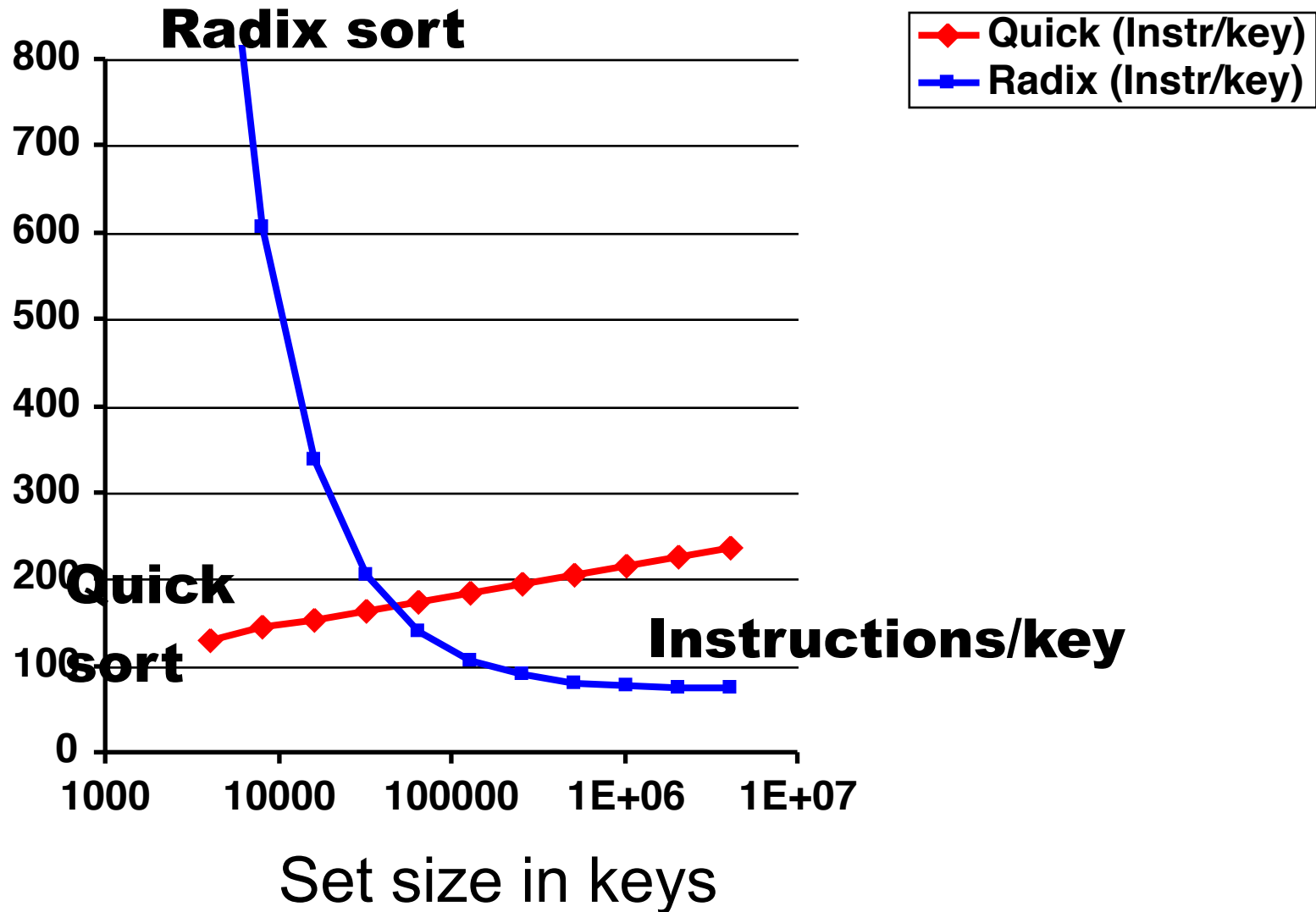
- Οι αστοχίες εξαρτώνται από τα μοτίβα προσπέλασης μνήμης
 - Συμπεριφορά του αλγορίθμου
 - Βελτιστοποίηση του μεταγλωττιστή για προσπελάσεις μνήμης



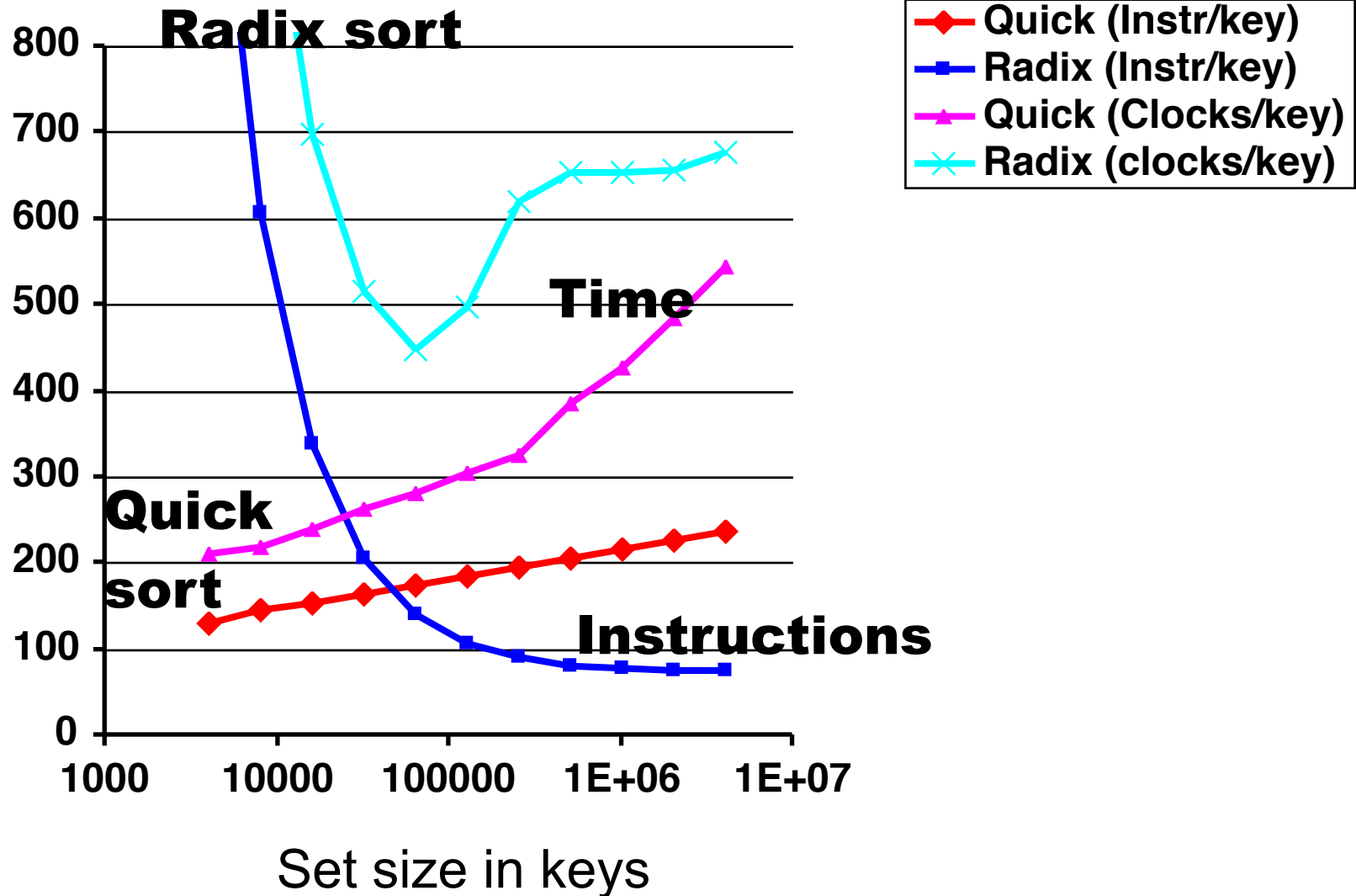
Impact of Memory Hierarchy on Algorithms

- CPU time is a function of (ops, cache misses) vs. just $f(\text{ops})$. What does this mean to Compilers, Data structures, Algorithms?
- “The Influence of Caches on the Performance of Sorting” by A. LaMarca and R.E. Ladner. Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, January, 1997, 370-379.
 - Quicksort: fastest comparison based sorting algorithm when all keys fit in memory
 - Radix sort: also called “linear time” sort because for keys of fixed length and fixed radix a constant number of passes over the data is sufficient independent of the number of keys
- DEC Alphastation 250, 32 byte blocks, direct mapped L2 2MB cache, 8 byte keys, from 4000 to 4000000

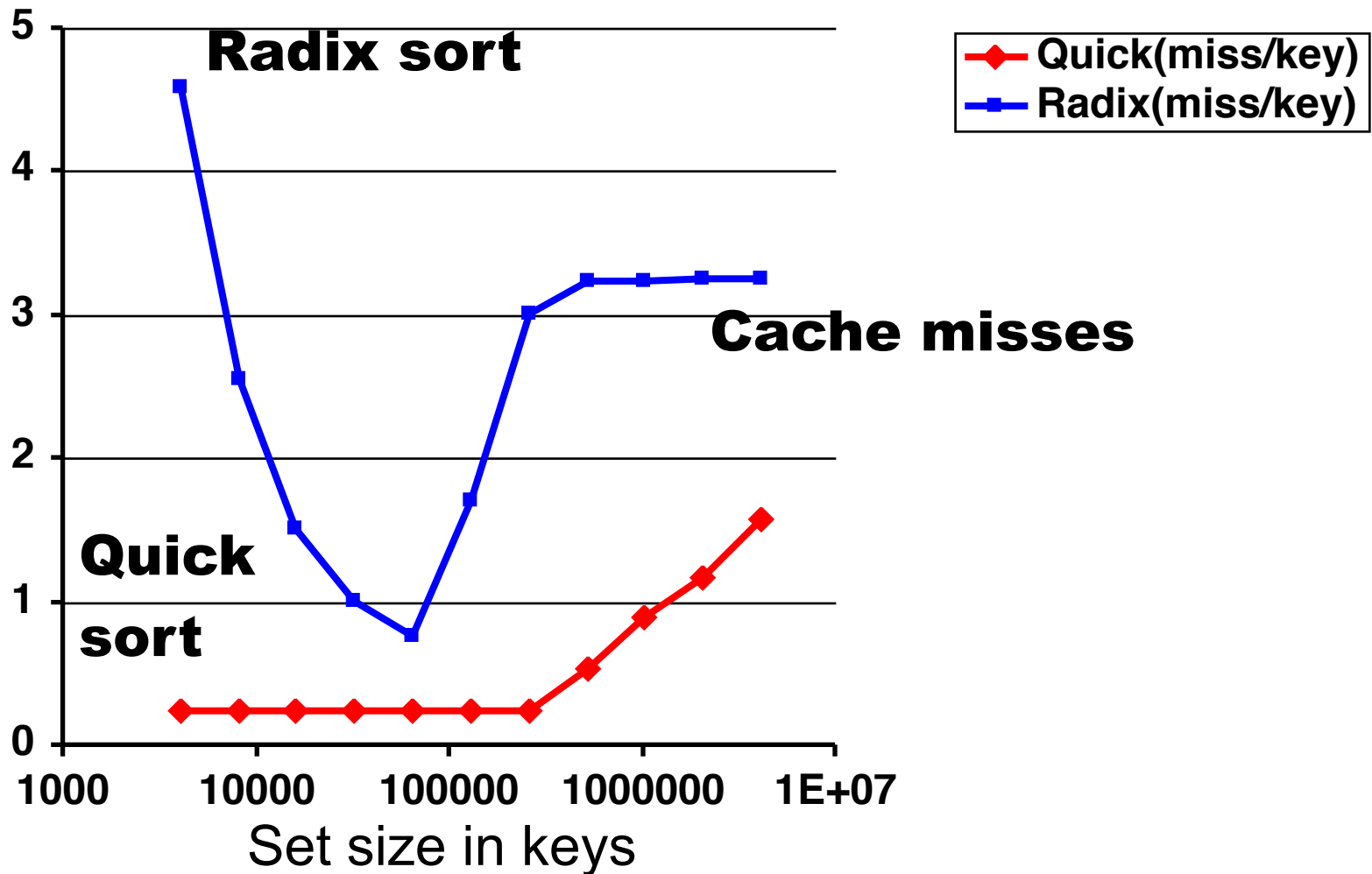
Quicksort vs. Radix Instructions varying #keys



Quicksort vs. Radix Instructions & Time varying #keys



Quicksort vs. Radix Cache misses varying #keys:



What is proper approach to fast algorithms?

FSM ελεγκτή κρυφής μνήμης

