Αρχιτεκτονική Υπολογιστών

Ενότητα 11: Κρυφή Μνήμη

Δρ. Μηνάς Δασυγένης

mdasyg@ieee.org

Εργαστήριο Ψηφιακών Συστημάτων και Αρχιτεκτονικής Υπολογιστών

http://arch.ece.uowm.gr/mdasyg

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

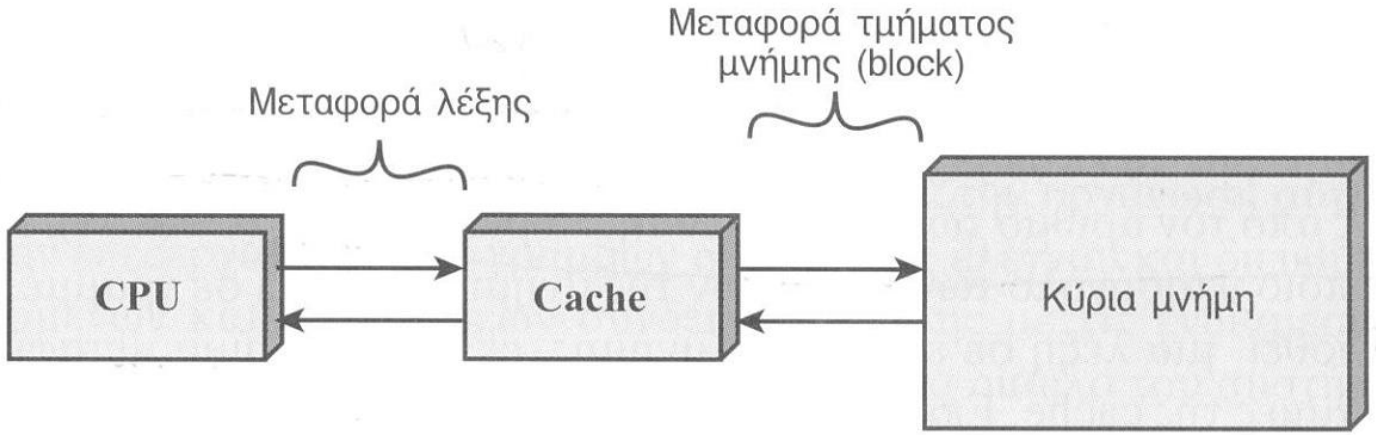
Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ψηφιακά Μαθήματα στο Πανεπιστήμιο Δυτικής Μακεδονίας**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Σκοπός ενότητας

- Η κατανόηση της σημαντικότητας της κρυφής μνήμης.
- Η κατανόηση της λειτουργίας της κρυφής μνήμης.

Η κρυφή μνήμη



Αναλογία της cache από την καθημερινότητα

Η cache παρομοιάζεται με ένα ψυγείο σε μια κουζίνα.

Ο μάγειρας αν δεν είχε ψυγείο θα έπρεπε όταν μαγείρευε κάτι να βγαίνει από το σπίτι του και να πηγαίνει στο μαγαζί να το προμηθευτεί.

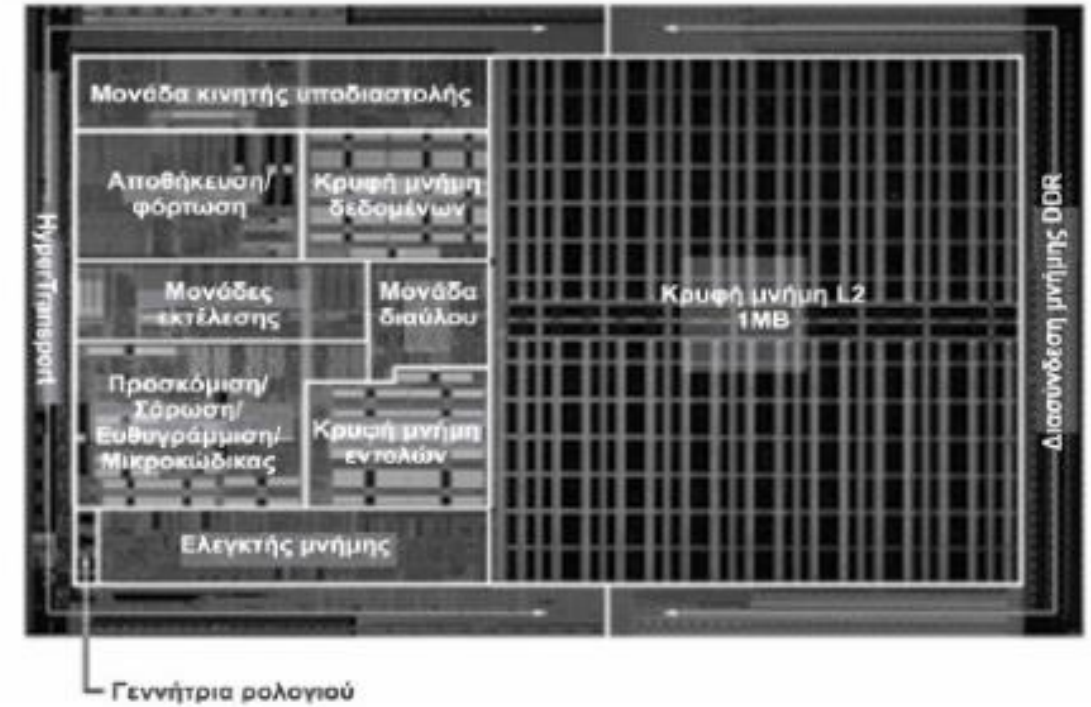
Χρησιμοποιώντας το ψυγείο μπορεί να προμηθευτεί μεγαλύτερες ποσότητες και να τις αποθηκεύσει προσωρινά αυξάνοντας την αποδοτικότητά του.

Τι είναι η κρυφή (ή σκιώδης) μνήμη;

- Η κρυφή μνήμη είναι μια πολύ γρήγορη και ακριβή μνήμη τεχνολογίας SRAM.
- Αν και μπορεί να βρίσκεται εκτός ολοκληρωμένου κυκλώματος, βρίσκεται εντός μαζί με τα επεξεργαστικά στοιχεία.
- Υπάρχει cache επιπέδου 1 (L1) πολύ μικρή και πολύ γρήγορη και χαμηλή κατανάλωση ενέργειας.
- Υπάρχει cache επιπέδου 2 (L2) μεγαλύτερη από την L1 αλλά λίγο πιο αργή και πιο ενεργοβόρα.
- Μπορεί να υπάρχει cache επιπέδου 3 (L3).

Κρυφή μνήμη εντός ολοκληρωμένου κυκλώματος

- Συνήθως κυριαρχεί σε μέγεθος στο chip του επεξεργαστή.
- Πολλαπλά επίπεδα κρυφής μνήμης.
  - Level. 1 (L1). Level 2 (L2), κλπ.

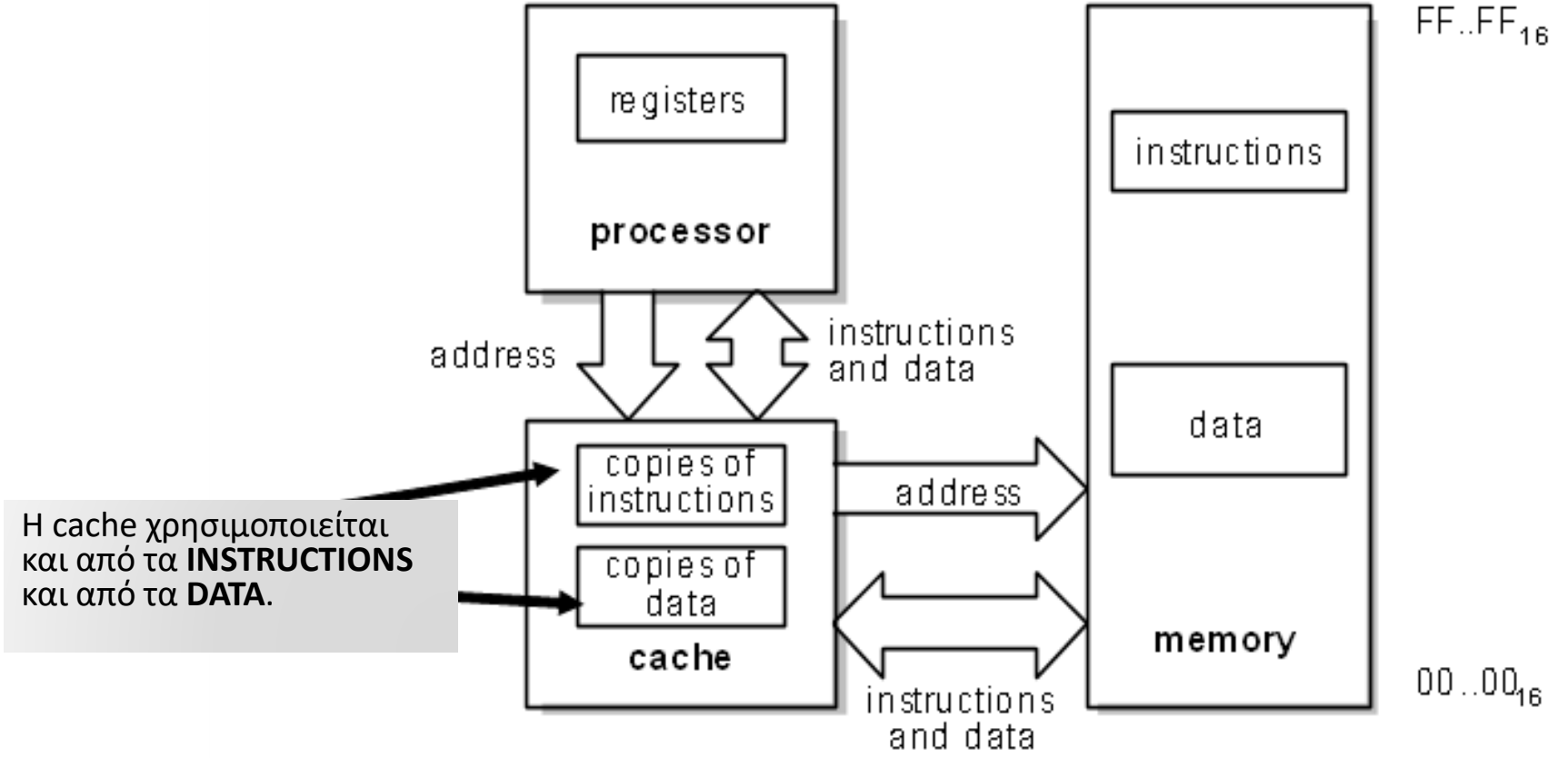


AMD Opteron

Πως συνδέεται η cache με την αρχιτεκτονική Von Neuman;

- Η αρχιτεκτονική **Von Neuman** έχει μια κοινή μνήμη και για δεδομένα και για εντολές.
- Μπορεί να αποτελείται από:
- Ενοποιημένη cache για instructions+data (unified cache).
- Ξεχωριστή cache (split cache) για instructions και ξεχωριστή cache για data (καλύτερη βελτιστοποίηση). Αυτός ο σχεδιασμός λέγεται σχεδιασμός κρυφής μνήμης Harvard.
- Κάποια επίπεδα να είναι ενοποιημένα (π.χ. Το L2,L3) και ένα ή παραπάνω να είναι ξεχωριστά (π.χ. L1).

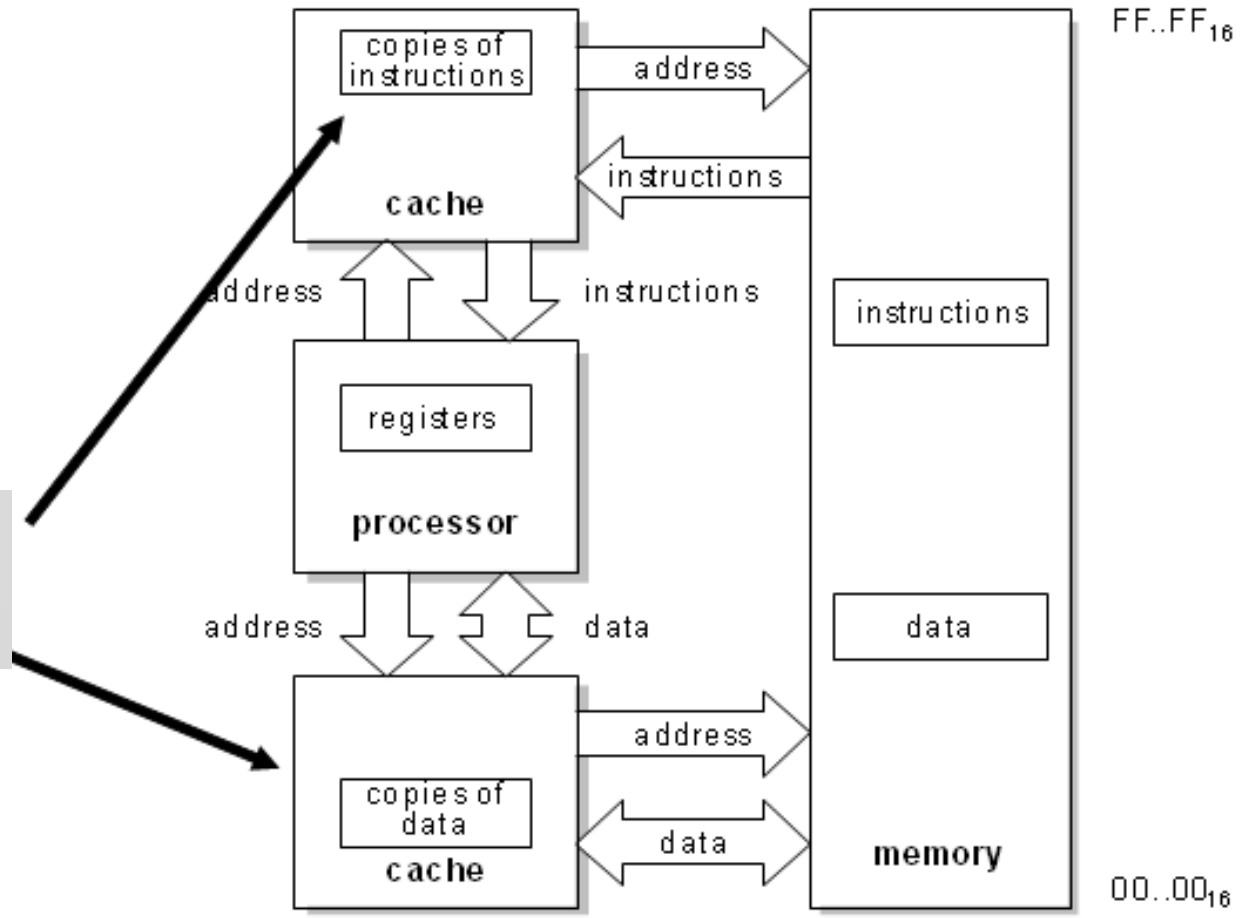
Ενοποιημένη (unified) cache για Instructions & Data



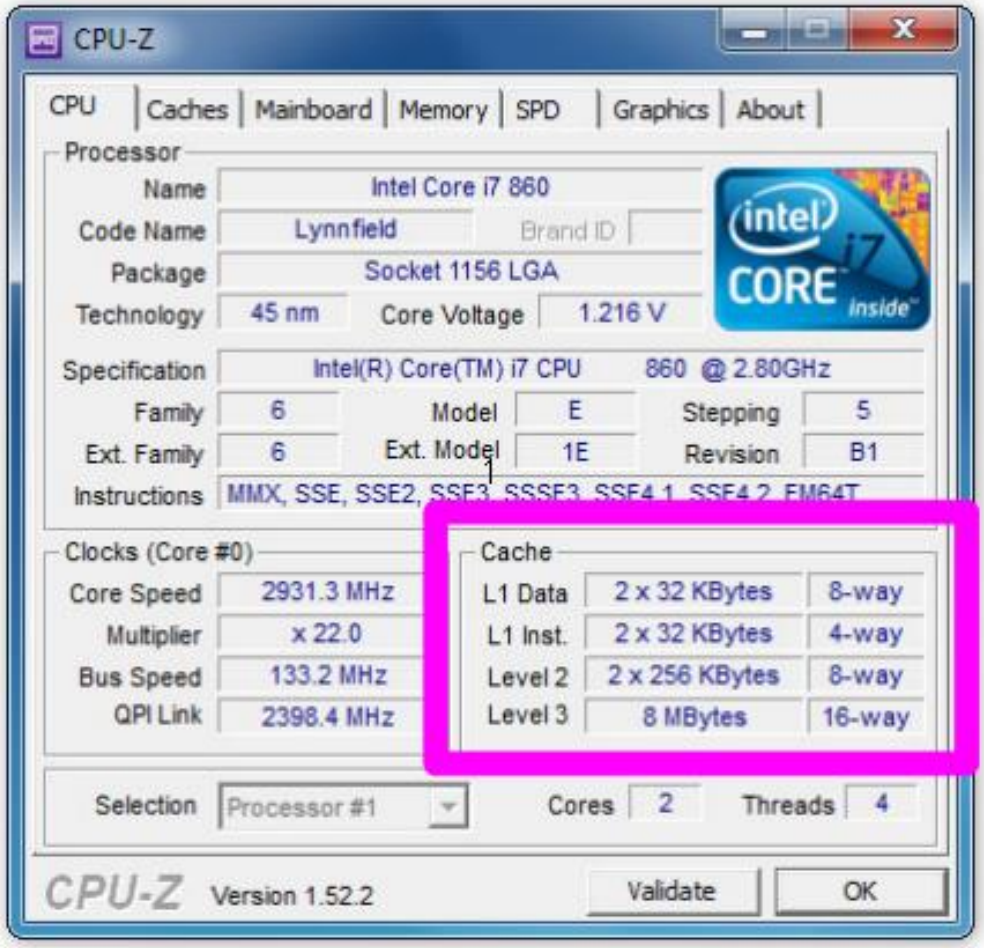
Η cache χρησιμοποιείται και από τα **INSTRUCTIONS** και από τα **DATA**.

Ξεχωριστή cache (split cache) για Instructions & Data

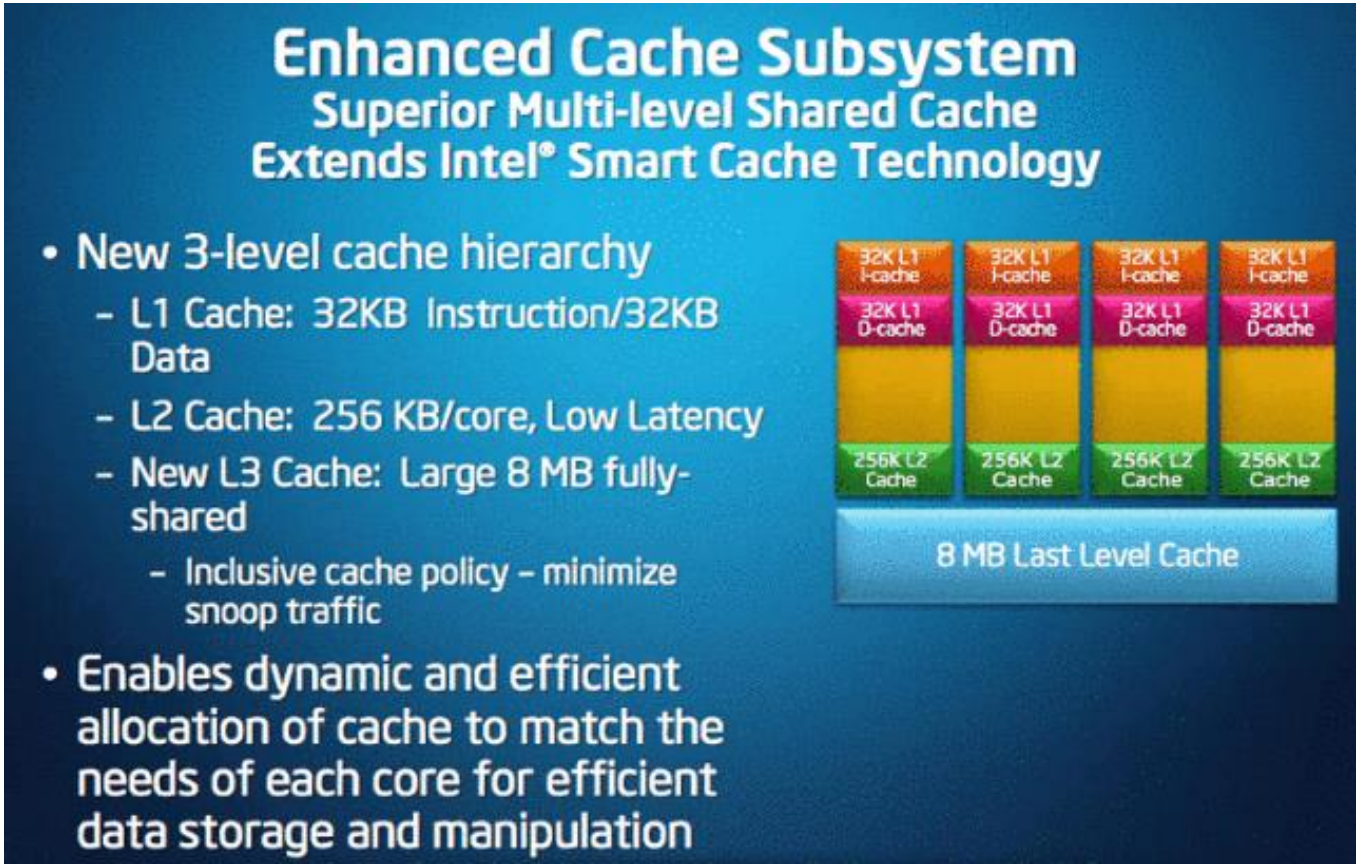
Ξεχωριστή cache για **INSTRUCTIONS** και **DATA**.



Οι κρυφές μνήμες σε ένα σύγχρονο επεξεργαστή I7 (1/2)



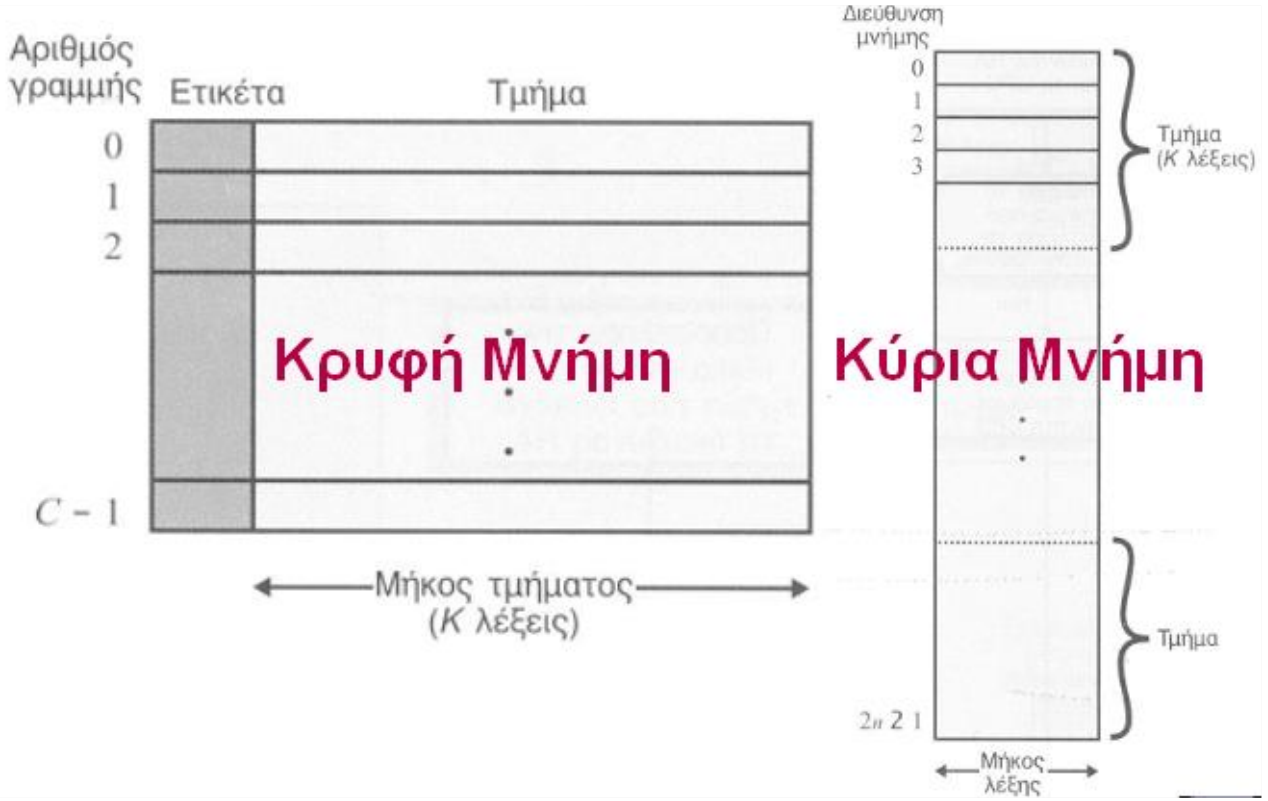
Οι κρυφές μνήμες σε ένα σύγχρονο επεξεργαστή I7 (2/2)



Οργάνωση της κρυφής μνήμης

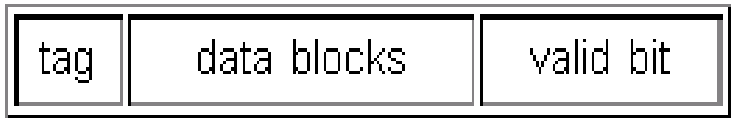
Ο αριθμός των γραμμών της cache είναι σημαντικά μικρότερος από τις γραμμές της κύριας μνήμης.

Η κρυφή μνήμη πάντα έχει υποσύνολο τμημάτων της κύριας μνήμης.



Λειτουργία της cache

- Η cache χωρίζεται σε γραμμές οι οποίες αποτελούνται από έναν αριθμό από Bytes.
- Η γραμμή της cache ονομάζεται block ή line.
- Κάθε γραμμή της cache αποτελείται από:

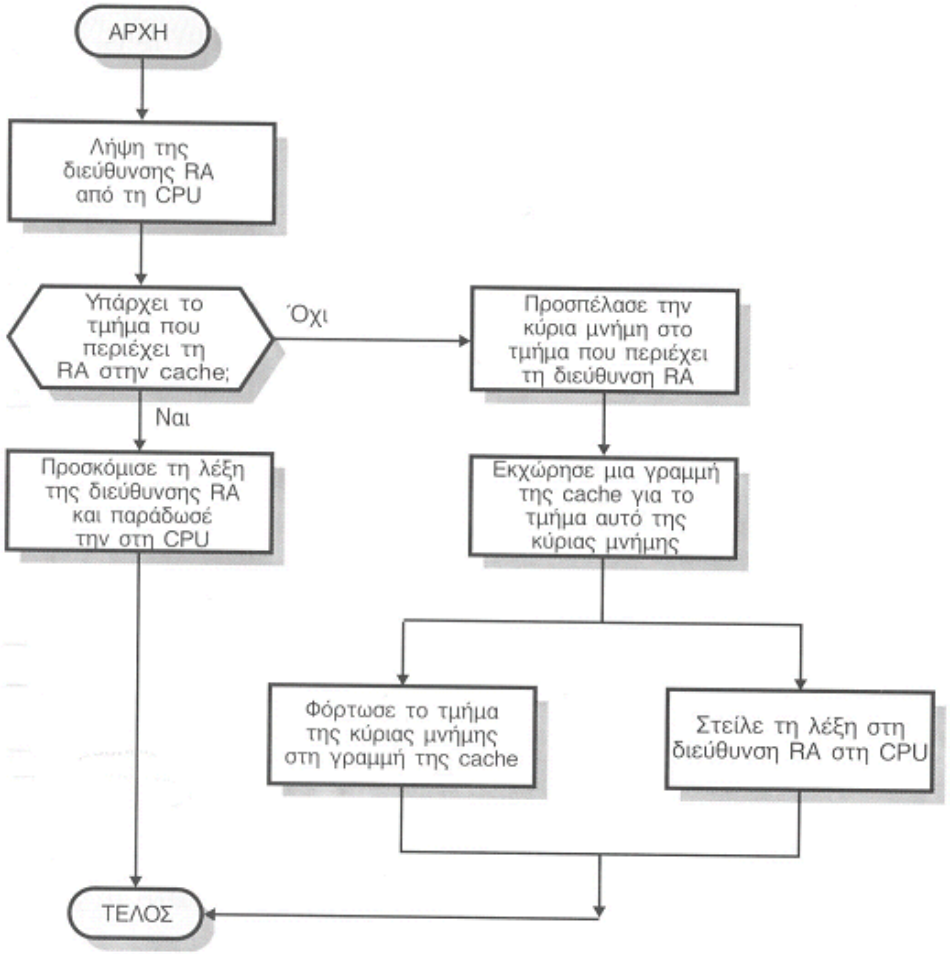


- Δεδομένα (data).
- Bit διεύθυνσης (tag).
- Επιπρόσθετα bit για κάποιες εξειδικευμένες αρχιτεκτονικές cache.
- Υπάρχουν διάφορες αρχιτεκτονικές cache μνήμης.

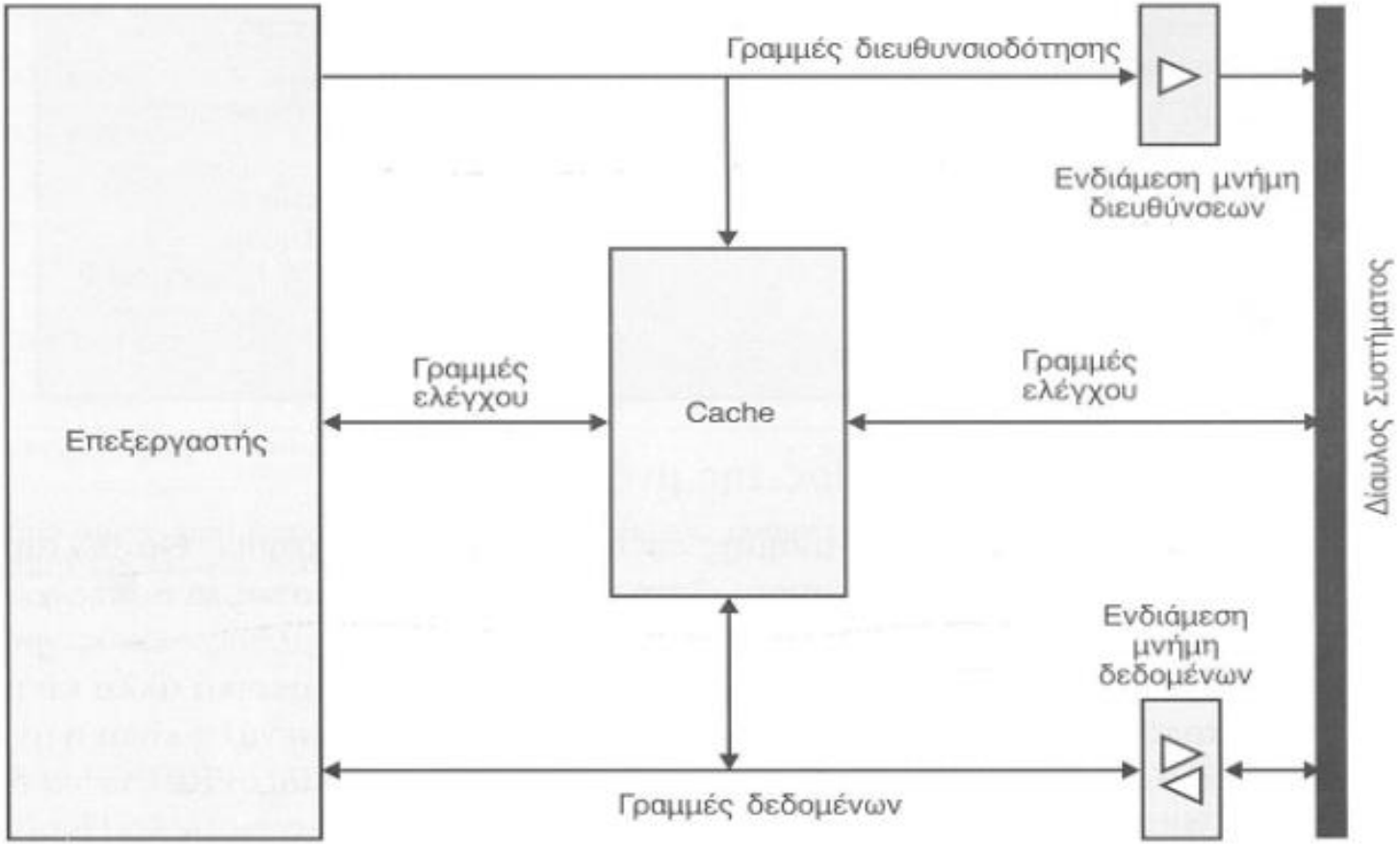
Εξηγείστε τη λειτουργία της ετικέτας tag

- Κάθε χρονική στιγμή κάποιο υποσύνολο των τμημάτων της κύριας μνήμης βρίσκεται στην κρυφή μνήμη.
- Η ετικέτα tag κάθε γραμμής μας δείχνει ποιο τμήμα της κύριας μνήμης βρίσκεται στη συγκεκριμένη γραμμή.
- Τα bit της ετικέτας tag είναι ένας αριθμός από τα πιο σημαντικά bit (MSB) της διεύθυνσης της κύριας μνήμης. Ο αριθμός των bit της ετικέτας καθορίζεται από τη δομή της cache, όπως θα φανεί σε επόμενες διαφάνειες.

Ενσωμάτωση της κρυφής μνήμης στην ιεραρχία μνήμης



Ενσωμάτωση της κρυφής μνήμης στην ιεραρχία μνήμης



Ποιο είναι το βέλτιστο μέγεθος της cache;

- Όσο πιο μεγάλη η cache, τόσο πιο αργή.
- Όσο πιο μεγάλη η cache, τόσο απαιτείται μεγαλύτερο εμβαδόν στην επιφάνεια (λόγω της μνήμης SRAM που απαιτεί αρκετά τρανζίστορ).
- Εξαρτάται από τη φύση της εργασίας.
- Δεν υπάρχει βέλτιστο μέγεθος.
- Ο Pentium 4 έχει 8KB L1, 256KB L2.

Εξηγήστε το V bit στις κρυφές μνήμες

- Όλες οι γραμμές (cache lines) στις κρυφές μνήμες εκτός από τα δεδομένα και το tag, έχουν και μια σειρά από επιπρόσθετα bit.
- Ένα από αυτά τα bit είναι το V bit (valid bit) ή bit εγκυρότητας.
- Το bit αυτό δείχνει κατά πόσο είναι έγκυρα τα στοιχεία που βρίσκονται εκεί.
- Η εγκυρότητα σημαίνει ότι τα δεδομένα που έχει η αντίστοιχη γραμμή συμφωνούν με τα δεδομένα που έχει η RAM για την αντίστοιχη διεύθυνση μνήμης.
- ...

Πότε η Cache δεν έχει έγκυρα δεδομένα;

- Υπάρχουν περιπτώσεις που η cache δεν έχει έγκυρα δεδομένα. Παραδείγματα:
- Αν δεν έχουν μεταφερθεί δεδομένα στην αντίστοιχη γραμμή cache (π.χ. Όταν ξεκινάει ο υπολογιστής).
- Αν δοθεί εντολή για άδειασμα της cache (cache flush).
- Αν έχουμε πολυ-πύρηνα συστήματα και ένας επεξεργαστής τροποποιήσει τα δεδομένα στην κοινή μνήμη του συστήματος, τότε στέλνεται στις ιδιωτικές κρυφές μνήμες των άλλων επεξεργαστών σήμα ακυρότητας της συγκεκριμένης διεύθυνσης και των δεδομένων που είχαν αντιγράψει.

Τι άλλα bit βρίσκονται συνήθως σε μια γραμμή cache;

- Εκτός από το bit εγκυρότητας μπορεί να υπάρχουν και τα εξής bit:
- Bit εγγραφής στη μνήμη (dirty bit). Αν είναι ενεργοποιημένο αυτό το bit τότε σημαίνει ότι έχει τροποποιηθεί κάποια γραμμή στη κρυφή μνήμη και αυτή η αλλαγή δεν έχει γραφεί στη μνήμη RAM (τεχνική write-back).
- Bit χρήσης. Τα bit αυτά προσδιορίζουν σε μια ομάδα από γραμμές της cache ποια θα απομακρυνθεί, προκειμένου να έρθει μια νέα γραμμή.
- Bit για πρωτόκολλα συνέπειας κρυφών μνημών σε πολυεπεξεργαστές, που κωδικοποιούν καταστάσεις όπως modified, exclusive, shared, invalid.

Θεμελιώδεις έννοιες σχετικά με τις caches

- **Τοποθέτηση Block:**

Σε ποια τοποθεσία στη cache θα τοποθετηθεί ένα καινούργιο block;

- **Αναγνώριση Block:**

Πως βρίσκεται ένα block στη cache;

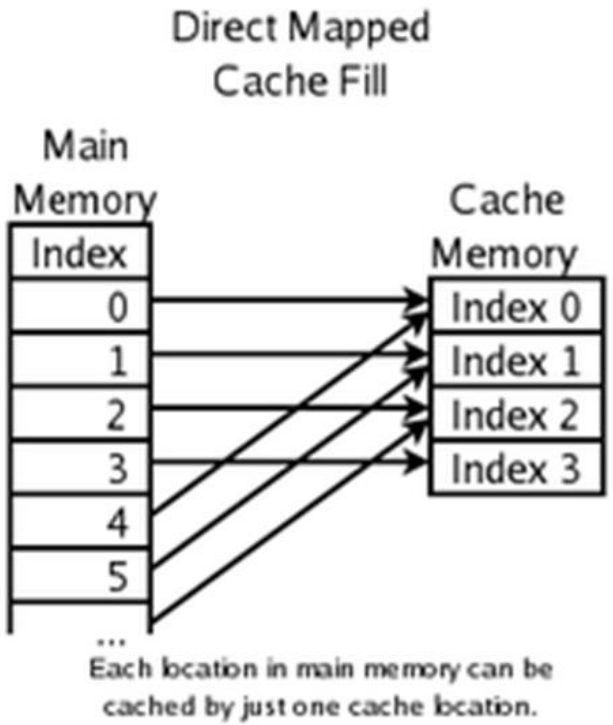
- **Αντικατάσταση Block:**

Ποιο block εξωθείται από τη cache;

- **Πολιτική Ενημέρωσης:**

Πως μεταφέρονται οι αλλαγές από τη cache στη μνήμη και αντίστροφα;

Τοποθέτηση: Σε πόσες θέσεις μπορεί να τοποθετηθεί ένα block;



- Η πιο απλή αρχιτεκτονική είναι η direct mapped (απευθείας απεικόνιση).
- Κάθε διεύθυνση της μνήμης μπορεί να τοποθετηθεί σε 1 μόνο γραμμή της cache.
- Ο υπολογισμός γίνεται από τη σχέση:
  - (Διεύθυνση μνήμης) MOD (αριθμών γραμμών cache).

Στο παράδειγμα η διεύθυνση 0 τοποθετείται στη γραμμή με index 0, όπως και η διεύθυνση 4, 8 κ.ο.κ.

Παράδειγμα Τοποθέτησης 1 με απευθείας αντιστοίχηση

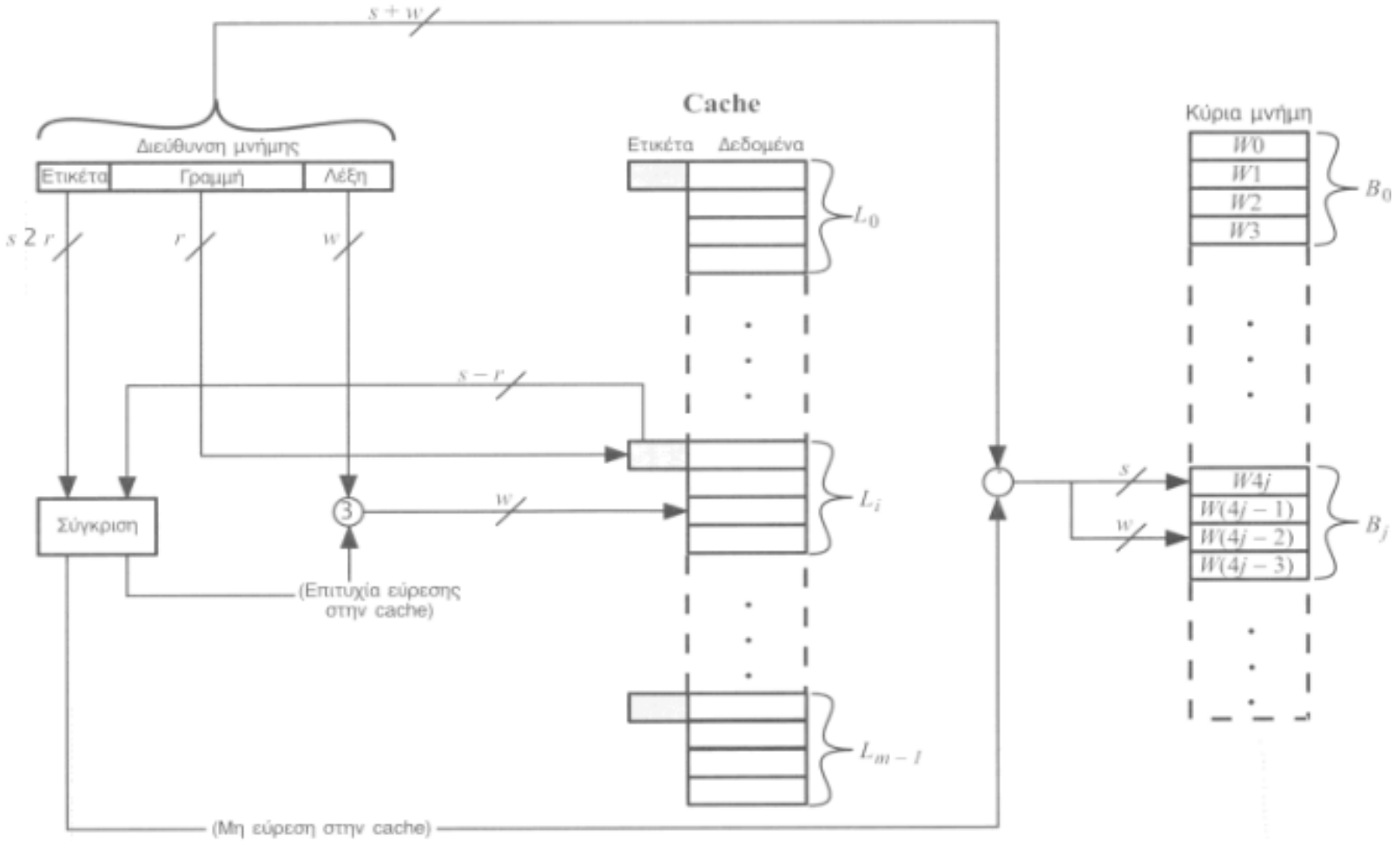
- Παράδειγμα1: Αν έχουμε cache 32KB με μέγεθος γραμμής 1Byte, τότε έχουμε 32768 γραμμές του 1Byte (δηλαδή 32768 indexes). Η διεύθυνση μνήμης 48.765.123 τότε θα τοποθετηθεί στη γραμμή της cache με index:

48.765.123 MOD 32768 = **6339**

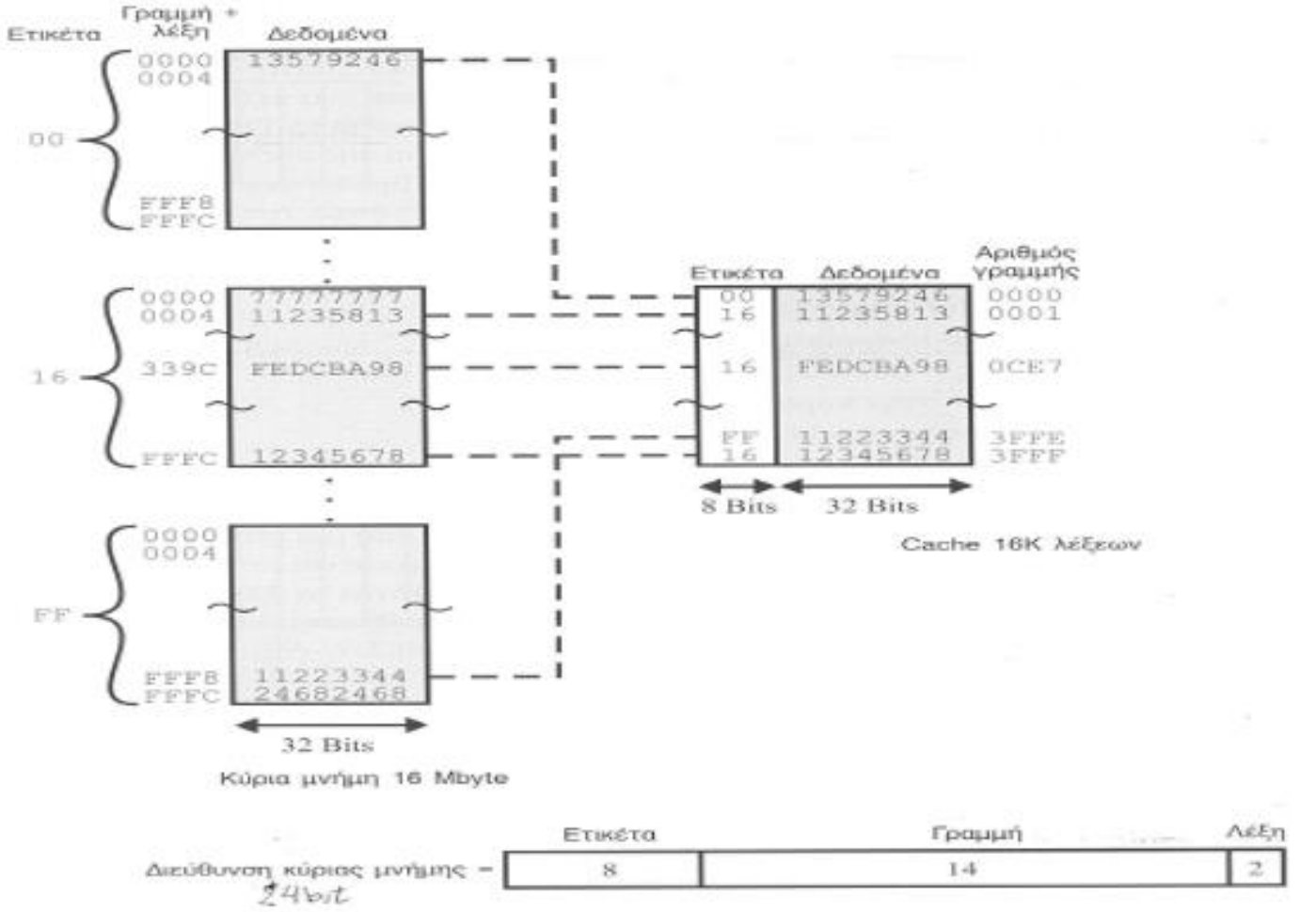
- Παρατηρήστε ότι επίσης η διεύθυνση της μνήμης 48.797.891 θα τοποθετηθεί επίσης στην ίδια γραμμή cache, αφού

48.797.891 MOD 32768 = **6339**

Η οργάνωση απευθείας χαρτογράφησης μνήμης



Παράδειγμα απευθείας χαρτογράφησης



Παράδειγμα Τοποθέτησης 2 (1/4)

- Παράδειγμα2: Αν έχουμε cache 32KB με μέγεθος γραμμής 32Byte, τότε έχουμε 1024 γραμμές κρυφής μνήμης με 32Byte η κάθε μια. Δηλαδή:
- τα Byte 0-31 αντιστοιχούν στην 1η γραμμή,
- τα Byte 32-63 αντιστοιχούν στη 2η γραμμή,
- ....,
- τα Byte 1024-1055 αντιστοιχούν στην 1η γραμμή.

(συνέχεια)

- ΠΡΟΣΟΧΗ: Αν το μέγεθος γραμμής είναι διαφορετικό από 1 Byte ο τρόπος υπολογισμού του index είναι διαφορετικός από μια πράξη mod (αναλύεται σε επόμενη διαφάνεια).

Παράδειγμα Τοποθέτησης 2 (2/4)

- Επειδή σε αυτήν την κρυφή μνήμη, μεταφέρονται κάθε φορά ομάδες (blocks) των 32Byte (32Byte απαιτούν 5 bit), αν θέλουμε να διαπιστώσουμε ποια είναι τα υπόλοιπα Bytes της γραμμής cache θα πρέπει τα 5 τελευταία bit να είναι ελεύθερα να έχουν οποιαδήποτε τιμή.
- Τα παρακάτω ανήκουν στην ίδια ομάδα: 110011001100?????

(104832) 110011001100**00000**

(104833) 110011001100**00001**

(104847) 110011001100**01111**

(104863) 110011001100**11111**

- Δηλαδή, αν απαιτηθεί και ένα μόνο Byte από το εύρος διευθύνσεων RAM 104832 – 104863, τότε θα μεταφερθούν και τα 32 Byte στην κρυφή μνήμη.

Παράδειγμα Τοποθέτησης 2 (3/4)

- Η πράξη λοιπόν που πρέπει να γίνει όταν έχουμε πολλαπλά Byte σε μια γραμμή cache είναι η πράξη τοποθέτησης 0 στα LSB για να βρούμε το χαμηλότερη διεύθυνση, και η πράξη τοποθέτησης 1 στα LSB για να βρούμε την υψηλότερη διεύθυνση. Θα τοποθετήσουμε τόσα 0 (έστω n), όσα απαιτούνται σύμφωνα με την εξίσωση 2^n = μέγεθος γραμμής cache.
- Στο παράδειγμά μας, $32=2^n$, δηλαδή n=5.
- Η τοποθέτηση 0 γίνεται με τη λογική πράξη AND.
- Αν έχουμε 20bit στο δίαυλο διευθύνσεων RAM, τότε η χαμηλότερη διεύθυνση είναι: (διεύθυνση μνήμης) **AND** 11111111111111100000.

Παράδειγμα Τοποθέτησης 2 (4/4)

- Αν έχουμε 20bit στο δίαυλο διευθύνσεων RAM,τότε η υψηλότερη διεύθυνση είναι:(διεύθυνση μνήμης) **AND** 11111111111111111111.

- Οι δύο αυτές διευθύνσεις αντιστοιχούν στη χαμηλότερη και υψηλότερη διεύθυνση στην ίδια γραμμή cache (με το ίδιο index).

Ποια είναι τα πλεονεκτήματα και τα μειονεκτήματα της απευθείας αντιστοίχησης;

- Απλή.
- Φθηνή.
- Χαμηλό ενεργειακό κόστος πρόσβασης.
- Υπάρχει μια μόνο αντιστοίχηση οπότε αν πολλαπλές διευθύνσεις RAM αντιστοιχούν στην ίδια γραμμή της cache και ένα πρόγραμμα ζητά επανειλημμένα από τα διαφορετικά τμήματα, τότε θα υπάρχουν πάρα πολλές αστοχίες.
- Αυτό το φαινόμενο ονομάζεται **παλινδρόμηση**.

Τοποθέτηση: Σε πόσες θέσεις μπορεί να τοποθετηθεί ένα block;

- Στην κρυφή μνήμη απευθείας απεικόνισης στην ίδια μοναδική γραμμή της κρυφής μνήμης αντιστοιχούν πολλαπλές διευθύνσεις μνήμης.
- Κάθε διεύθυνση μνήμης αντιστοιχεί σε μια μόνο γραμμή της κρυφής μνήμης (Αντιστοίχηση 1-προς-1).
- Μια τροποποίηση της κρυφής μνήμης είναι να υποστηρίζει την αντιστοίχηση μιας διεύθυνσης μνήμης σε πολλαπλές γραμμές cache. Αυτή η μνήμη ονομάζεται συσχετιστική, ή ***n***-δρόμων (n-way).

Λειτουργία της συσχετιστικής κρυφής μνήμης

- Σε μια κρυφή μνήμη ***n***-δρόμων μια διεύθυνση της μνήμης RAM, μπορεί να τοποθετηθεί σε οποιαδήποτε από τις n γραμμές του συνόλου που αντιστοιχεί.

- Επειδή μπορεί να τοποθετηθεί σε οποιαδήποτε από τις n γραμμές, υπάρχουν διάφορα θέματα που ανακύπτουν όπως, του που θα τοποθετηθεί ή ποια γραμμή θα αντικαταστήσει.

Παράδειγμα συσχετιστικής cache

- Έστω έχουμε μια cache 32KB, με μέγεθος γραμμής 32Byte και είναι 8 δρόμων (όπως του Intel I7).

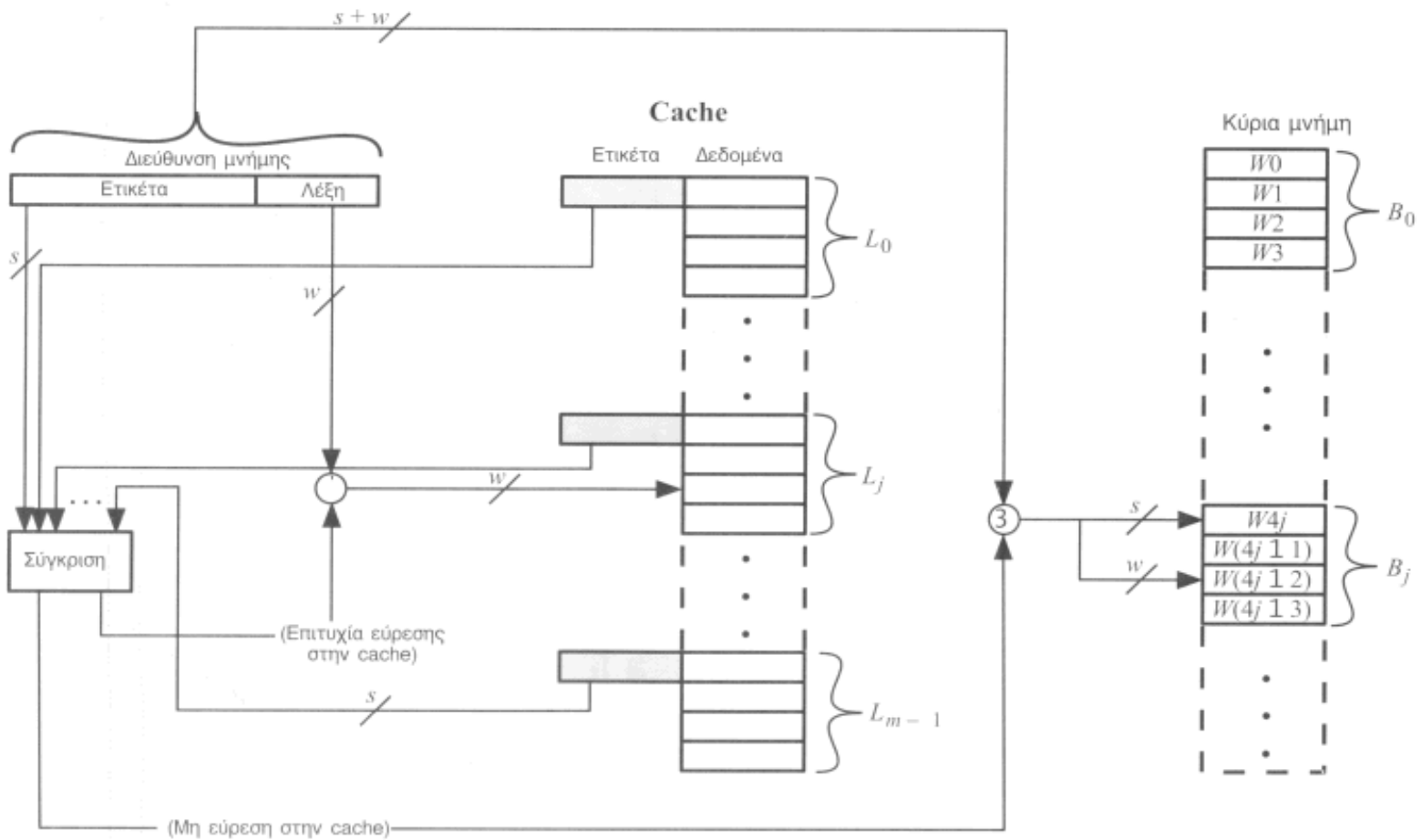
- 32KB με μέγεθος 32Byte σημαίνει ότι έχουμε 1024 γραμμές. Επειδή, όμως έχουμε κρυφή μνήμη 8 δρόμων, αυτό σημαίνει ότι οι γραμμές ομαδοποιούνται ανά 8, δηλαδή, έχουμε 128 ομάδες, με κάθε ομάδα να έχει 8 γραμμές. Οι 128 λοιπόν ομάδες αντιστοιχούν σε 128 index από 0 έως 127.

Βαθμός συσχέτισης κρυφής μνήμης

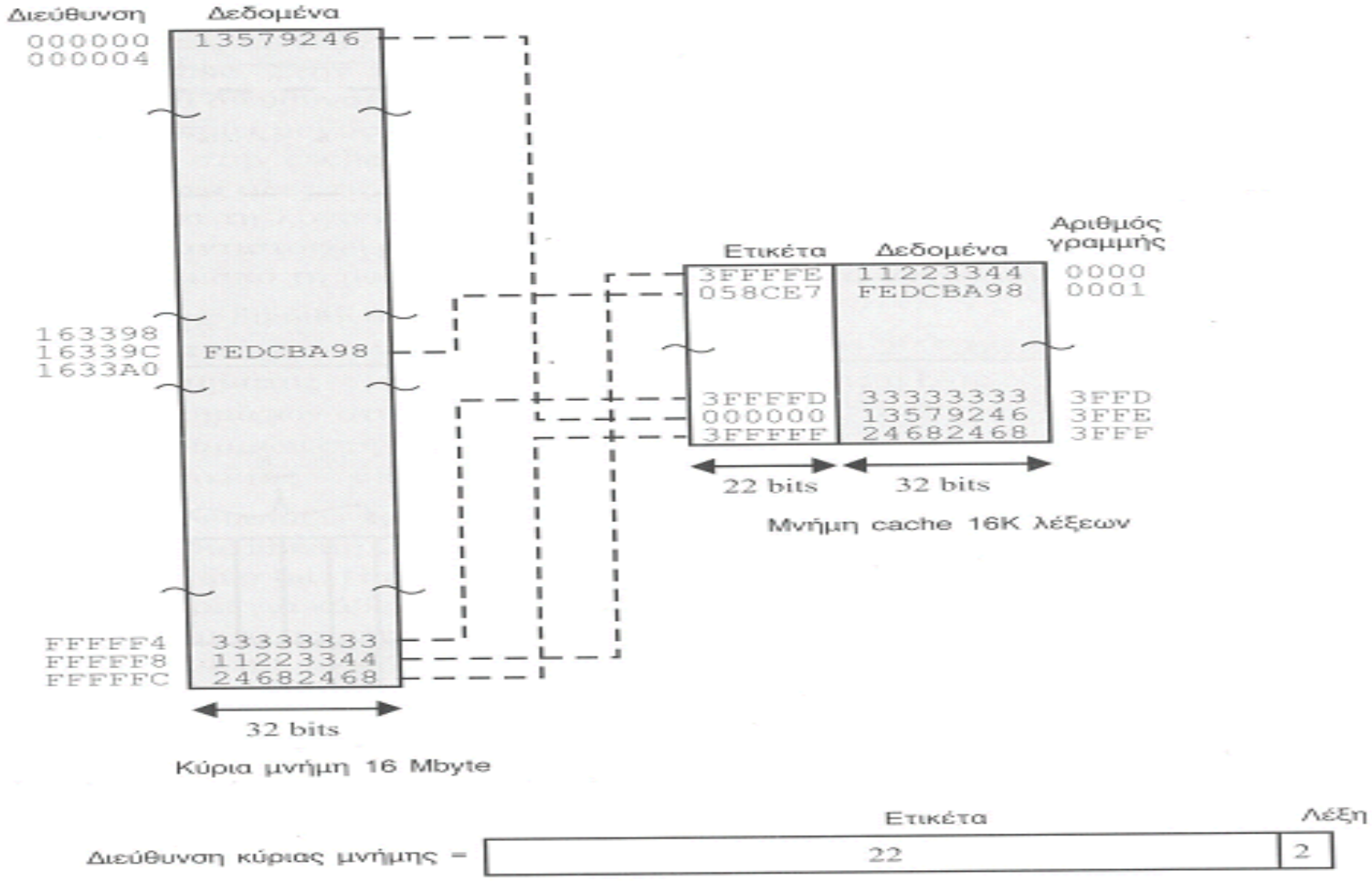
- Οι κρυφές μνήμες μπορεί να είναι 1 δρόμου (δηλαδή, απευθείας συσχέτισης), n δρόμων (δηλαδή, πολλαπλής συσχέτισης) και πλήρης συσχέτισης αν οποιαδήποτε διεύθυνση μπορεί να τοποθετηθεί οπουδήποτε.

- Όσο πιο μεγάλη συσχέτιση έχει τόσο μεγαλύτερη είναι η κατανάλωση ενέργειας.

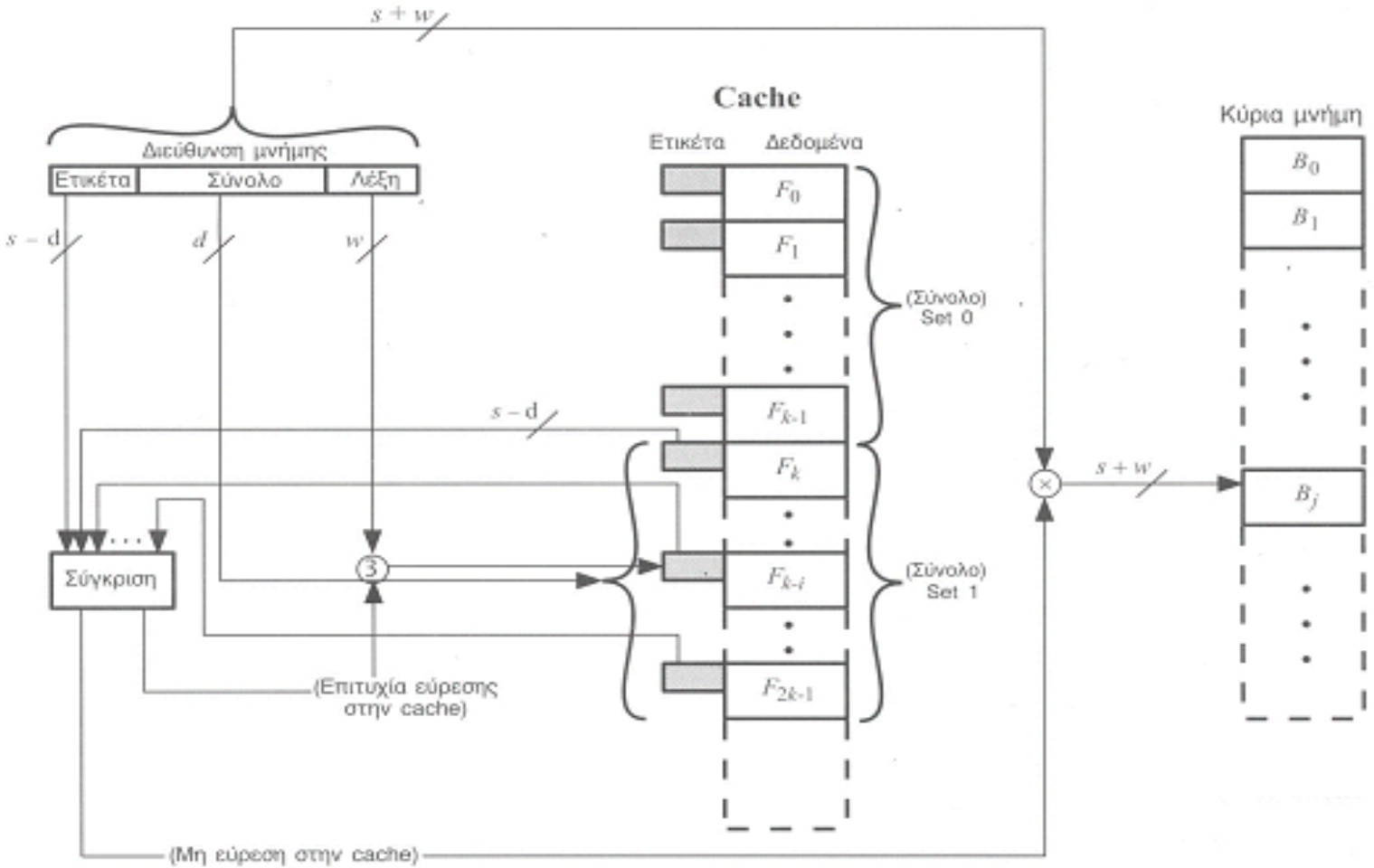
Η cache πλήρης συσχέτισης



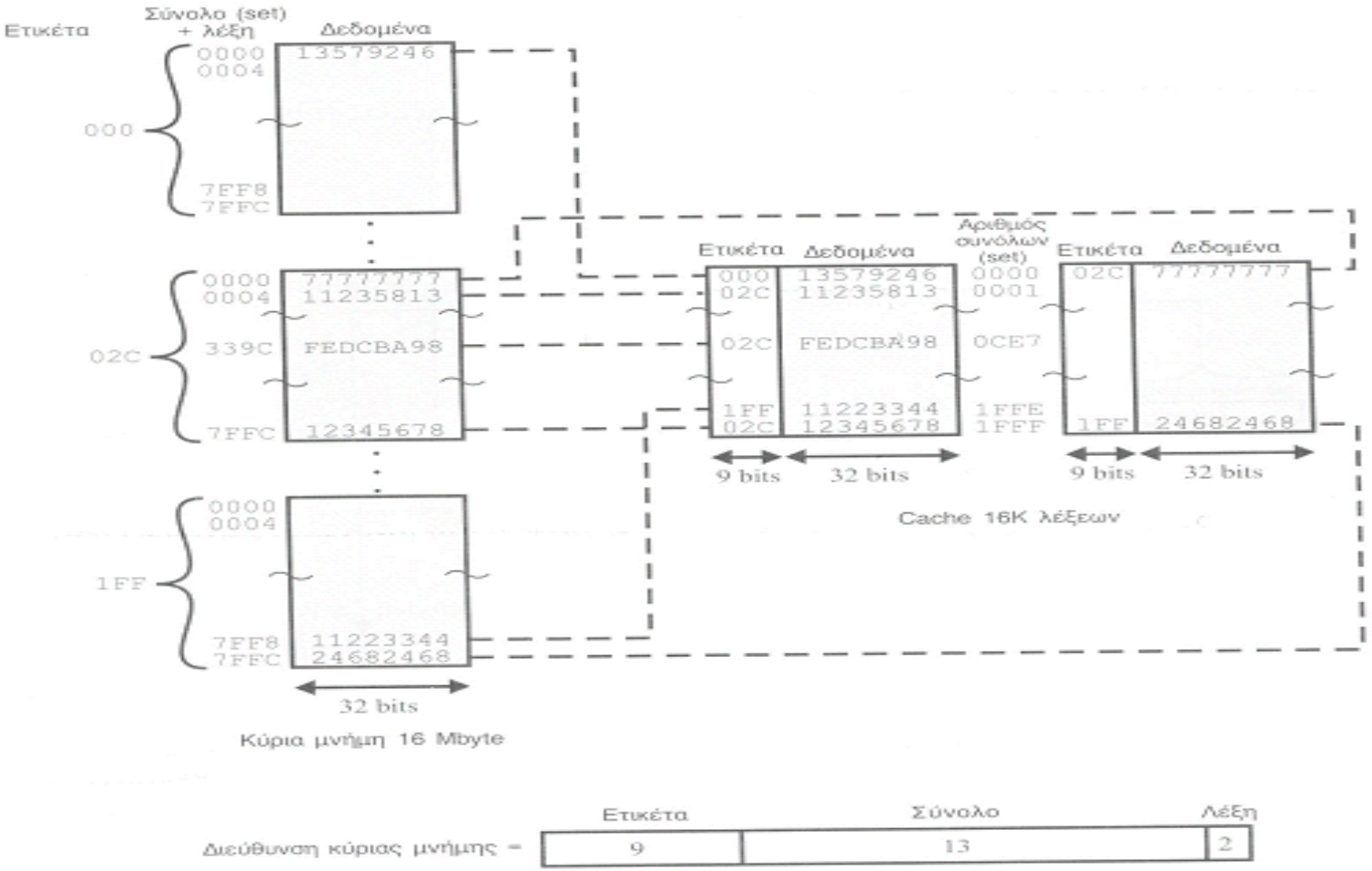
Παράδειγμα με cache πλήρης συσχέτισης



Οργάνωση cache με αντιστοίχηση n-δρόμων



Παράδειγμα κρυφής μνήμης με αντιστοίχηση 2 δρόμων



Πως συνδέεται η διεύθυνση μνήμης με το tag/index/displ; (1/5)

- Σε κάθε γραμμή cache υπάρχει ένα πεδίο από bits που ονομάζεται tag.
- Το tag είναι ένα υποσύνολο bit από τα συνολικά bit της διεύθυνσης μνήμης.
- Μια οποιαδήποτε διεύθυνση μνήμης RAM χωρίζεται σε 3 μέρη στην cache ως εξής:

- Δηλαδή, τα MSB της διεύθυνσης RAM αποτελούν το tag, στη συνέχεια κάποια bit δείχνουν το index (ή ομάδα), και τα LSB δείχνουν τη μετατόπιση.

Πως συνδέεται η διεύθυνση μνήμης με το tag/index/displ; (2/5)

- Αναλόγως την cache υπάρχουν ποικίλοι διαχωρισμοί της πραγματικής διεύθυνσης RAM στα 3 αυτά κομμάτια.
- Αν θεωρήσουμε ότι έχουμε συνολικά bit διεύθυνσης μνήμης RAM **p** bit, **k** bit για το index και **m** bit για το displacement, τότε τα tag bit είναι όσα περισσεύουν, δηλαδή p-k-m.
- Ισχύει πάντα:

αριθμός bit διεύθυνσης μνήμης =

tag bit nr. + index bit nr. + displ. bit nr.

Πως συνδέεται η διεύθυνση μνήμης με το tag/index/displ; (3/5)

- Το **k** προσδιορίζεται από τον αριθμό ομάδων της cache. Αν έχουμε π.χ. 128 (από 0 έως 127) ομάδες, αυτό σημαίνει ότι απαιτούνται 7 bit για να καλύψουν αυτό το εύρως.
- Γενικά, δεδομένου ότι γνωρίζουμε τον αριθμών των γραμμών της cache, το **k** υπολογίζεται από τη σχέση:

$$\lceil \log_2(cache\_rows) \rceil$$

Πως συνδέεται η διεύθυνση μνήμης με το tag/index/displ; (4/5)

- Σε κάθε γραμμή βρίσκεται ένας αριθμός από Bytes. Αυτό είναι το μέγεθος του cache block.
- Το cache block καθορίζει την ποσότητα των Bytes που θα πρέπει να μεταφερθούν κάθε φορά. Η κρυφή μνήμη δε μεταφέρει 1 Byte κάθε φορά, αλλά το προδιαγεγραμμένο μέγεθος από Bytes.
- Η κρυφή μνήμη μπορεί να μας δίνεται ότι έχει λέξεις (words) στο cache block. Θα πρέπει να μεταφράσουμε τις λέξεις σε Bytes. Για παράδειγμα αν έχουμε μια κρυφή μνήμη 4 λέξεων, με κάθε λέξη να είναι 4 Byte (32bit), τότε συνολικά έχουμε 16 Byte σε μια γραμμή της κρυφής μνήμης.

Πως συνδέεται η διεύθυνση μνήμης με το tag/index/displ; (5/5)

- Αν λοιπόν έχουμε 16 Byte σε μια γραμμή της cache, τότε το displacement (=η μετατόπιση πάνω στη γραμμή για να βρούμε το συγκεκριμένο Byte) απαιτεί 4 bit (για να καθοριστεί το Byte από 0 έως 15).
- Γενικά, αν γνωρίζουμε τον αριθμό data_blocks δηλαδή τον αριθμό από Bytes που έχουμε στη γραμμή cache τα bit του displacement καθορίζονται από τη σχέση:

$$\lceil \log_2(data\_blocks) \rceil$$

Πως καθορίζεται το tag; Παράδειγμα προσδιορισμού tag (1/3)

- Αν έχουμε λοιπόν μια RAM που χρησιμοποιεί διευθύνσεις ADDR των 32bit (δηλαδή μέγεθος από 0 έως 4GB), και μια cache 32KB, 8-way, 8 words των 32bit σε κάθε γραμμή cache, θα ισχύουν τα εξής:
- 8words * 32bit /8bit = 32 Byte block cache size.
  - Αριθμός bit για displacement 5 bit (0-31)
- 32KB / 32B = 1024 γραμμές.
- 1024 γραμμές / 8-way = 256 ομάδες.
  - Αριθμός bit για index 8 bit (0-255).
- 32bit διεύθυνση – 8bit index – 5bit displacement = 19bit tag.

=>Διακρίνουμε λοιπόν τα εξής πεδία:

- Displacement => ADDR[4:0] (δηλαδή από το 4 έως το 0)
- Index => ADDR[12:5]
- Tag =>ADDR[31:13]

Πως καθορίζεται το tag; Παράδειγμα προσδιορισμού tag (2/3)

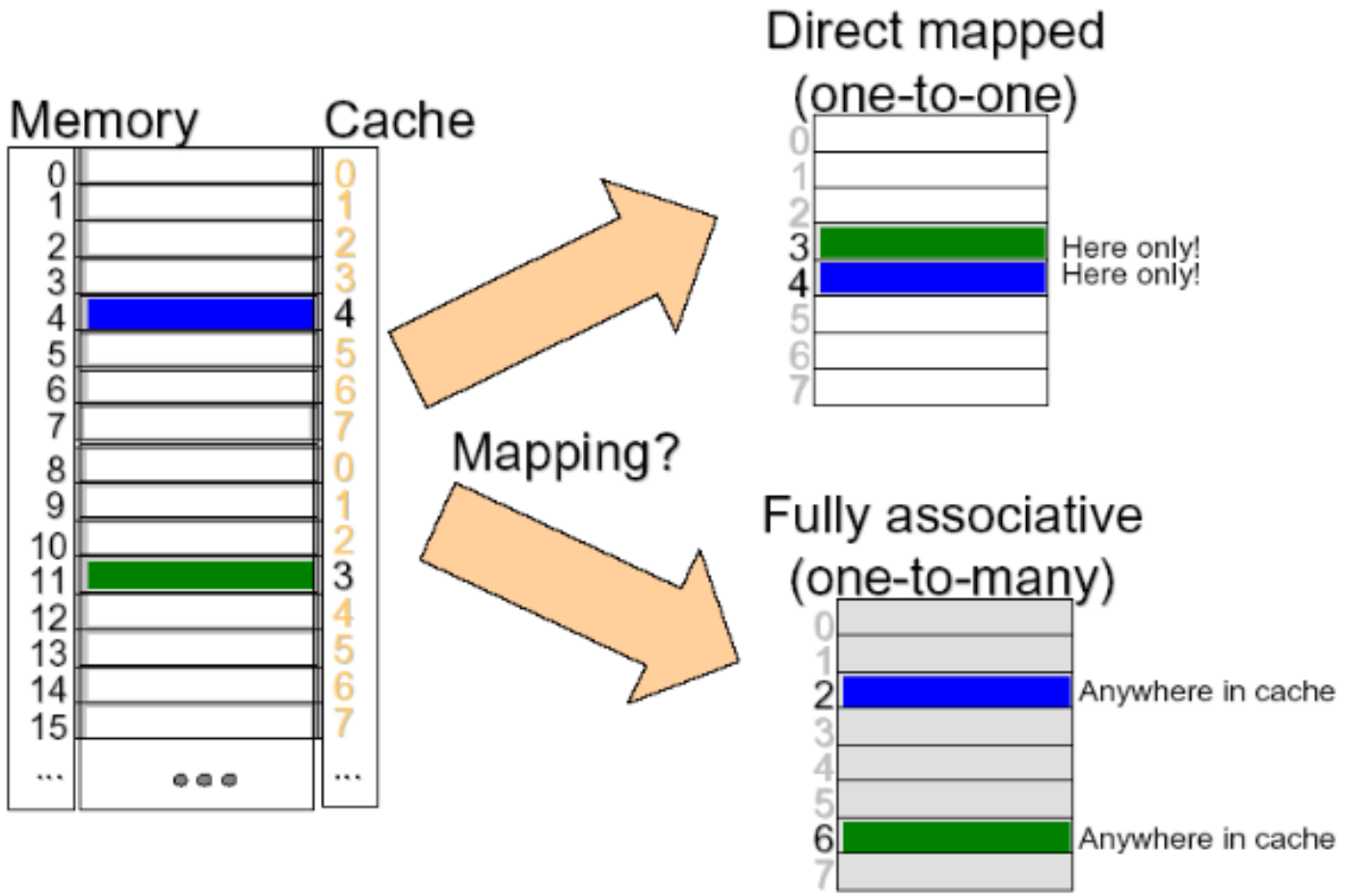
- Στο προηγούμενο παράδειγμα να καθοριστεί που θα τοποθετηθεί η διεύθυνση μνήμης 3.123.456.111 (ή 0xBA2C286F).
- Ο αριθμός αυτός έχει δυαδική αναπαράσταση:

  1011 1010 0010 1100 0010 1000 0110 1111
- Χωρίζεται η διεύθυνση στα εξής πεδία:
  - **Displacement** ADDR[4:0] 01111 = 15ο Byte
  - **Index** ADDR[12:5] 010000110 = 134η ομάδα
  - **Tag** ADDR[31:13] 1011 1010 0010 1100 001

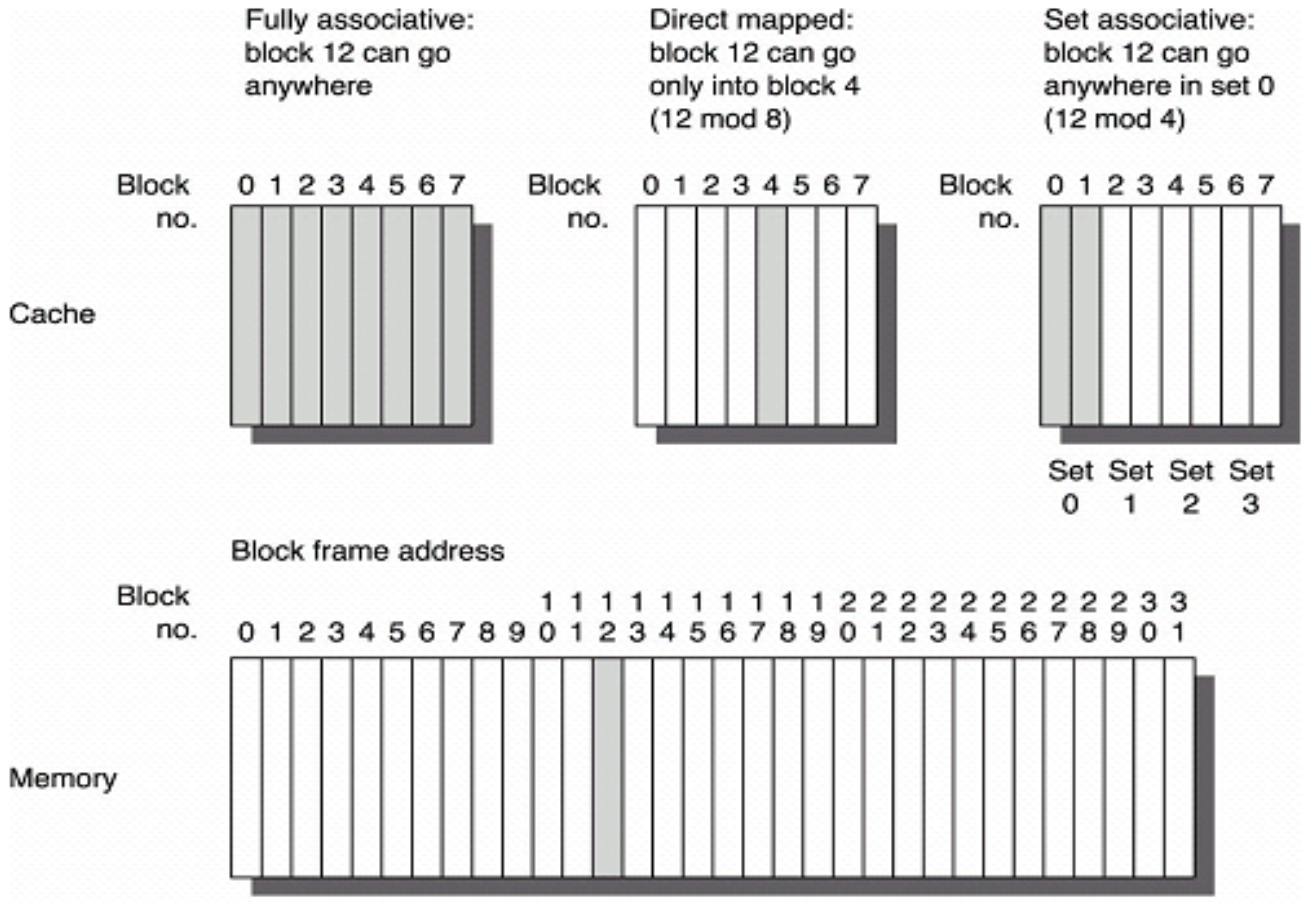
Πως καθορίζεται το tag; Παράδειγμα προσδιορισμού tag (3/3)

- Στο προηγούμενο παράδειγμα να καθοριστεί πόσα Byte θα μεταφερθούν μαζί με τη διεύθυνση αυτή και να καθοριστούν.
- 32 Byte είναι το block size, οπότε κάθε φορά θα μεταφέρονται 32 Byte.
- Θα μεταφερθούν τα Byte με το ίδιο tag και το ίδιο index (αφού ανήκουν στην ίδια γραμμή) ενώ το displacement θα έχει τιμές από 00000 έως 11111.
- Το πρώτο Byte1011 1010 0010 1100 0010 1000 0110 0000=3.123.456.096
- Το τελευταίο Byte1011 1010 0010 1100 0010 1000 0111 1111=3.123.456.127

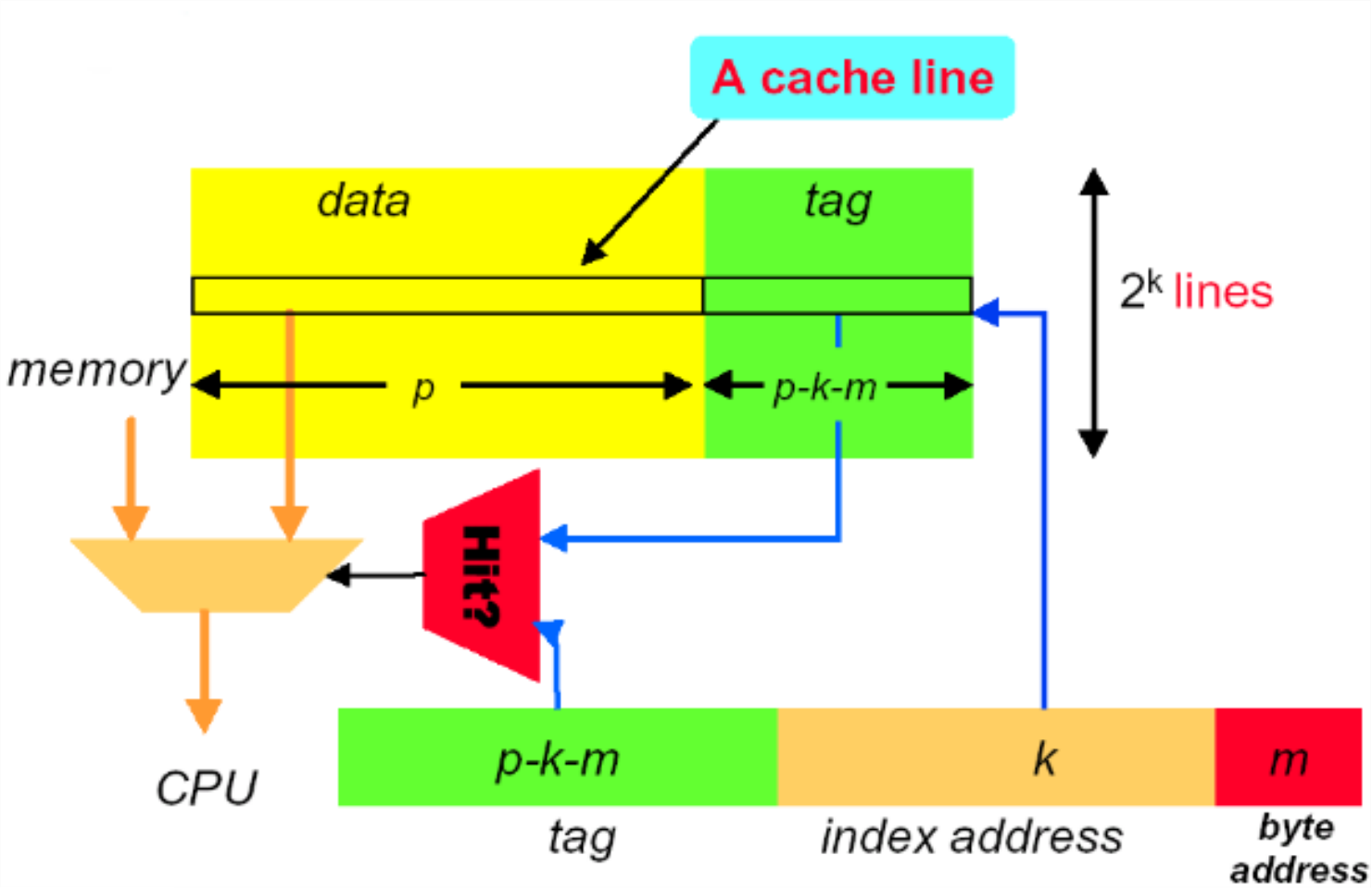
Τοποθέτηση: Σε πόσες θέσεις μπορεί να τοποθετηθεί ένα block; (1/2)



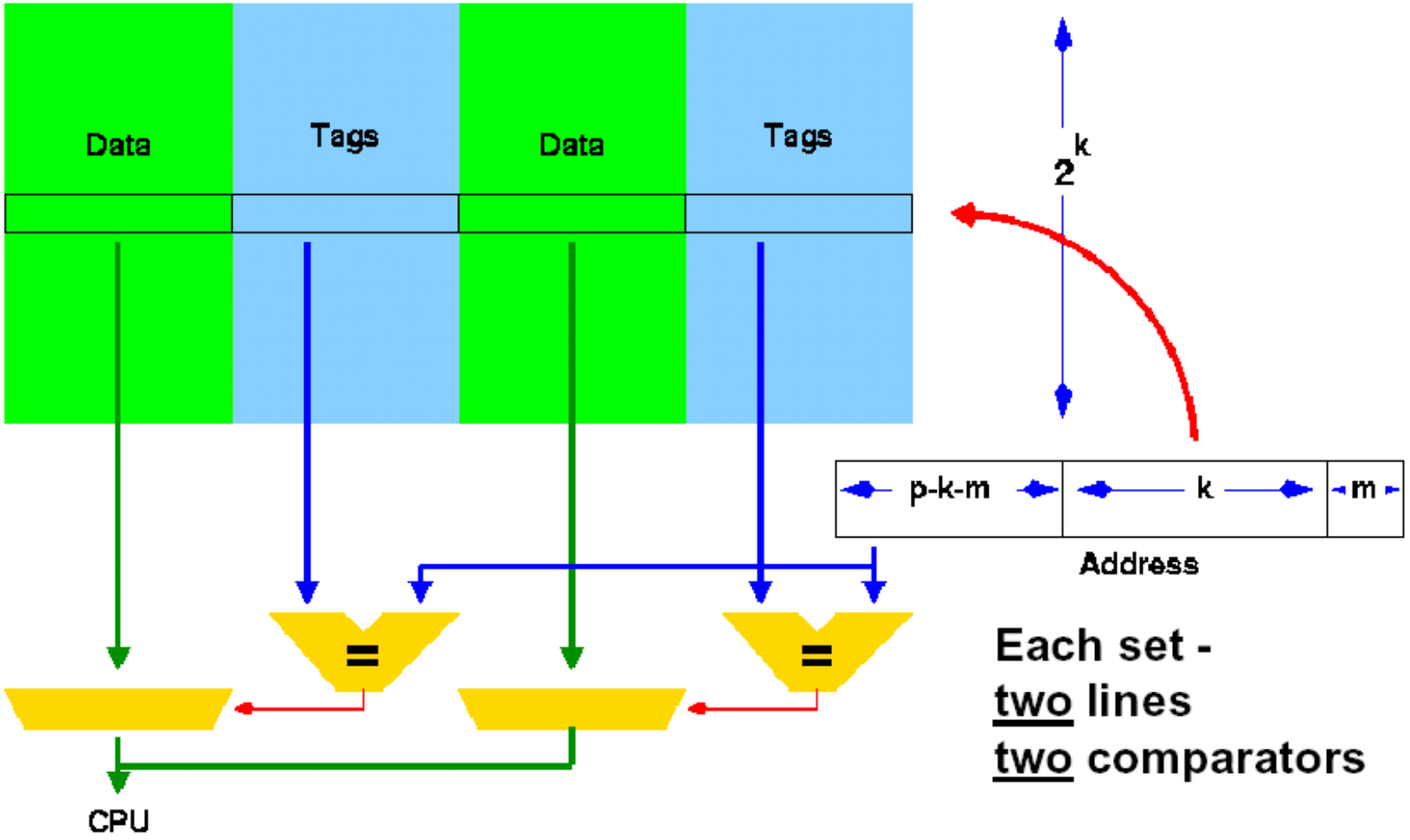
Τοποθέτηση: Σε πόσες θέσεις μπορεί να τοποθετηθεί ένα block; (2/2)



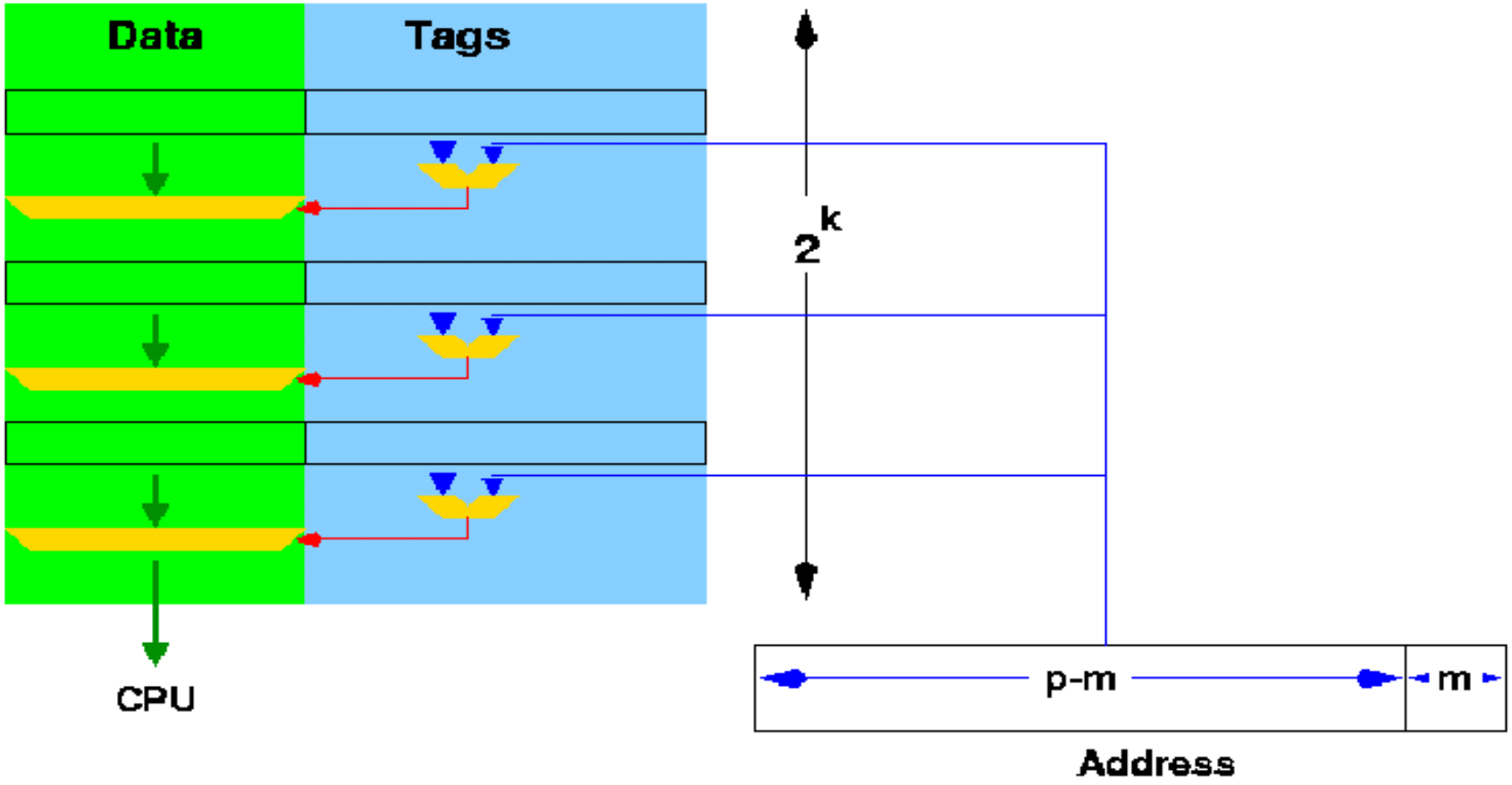
Cache απευθείας απεικόνισης



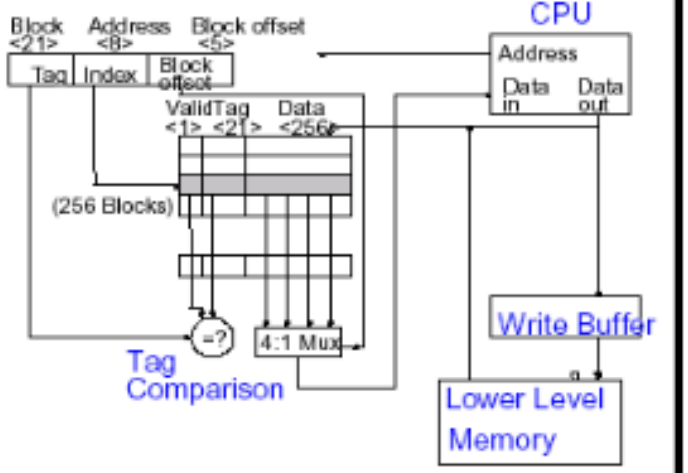
Συσχετιστική cache 2 γραμμών



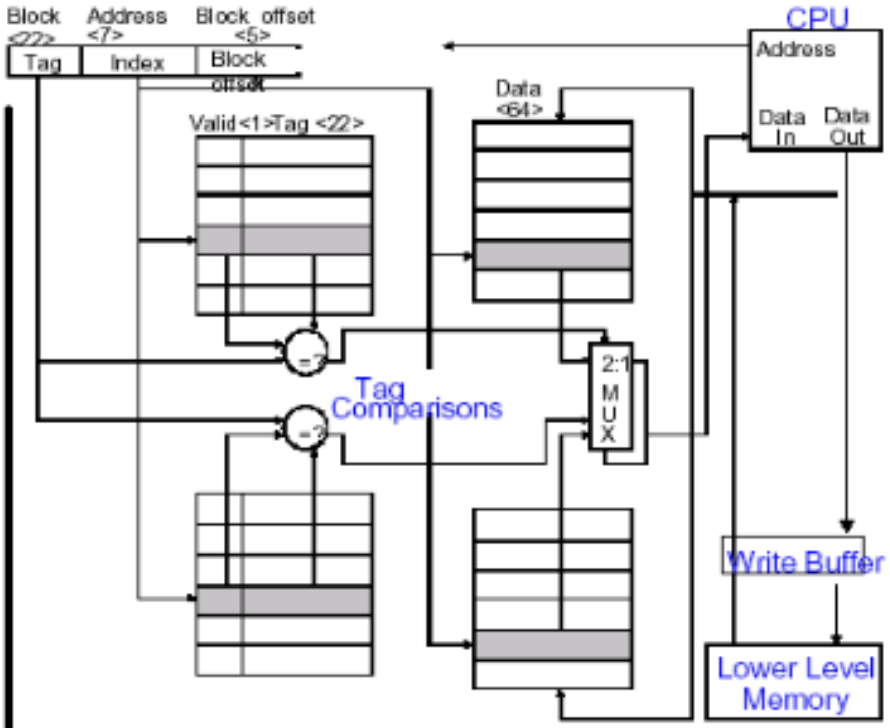
Πλήρως Συσχετιστική



Οι cache με συσχέτιση είναι ενεργοβόρες



- Direct mapped cache: (more) potential misses



- 2(4)-way associative: cost and energy inefficient

Τι ονομάζουμε μια αστοχία της cache;

- Όταν ο υπολογιστής απαιτήσει να διαβάσει μια διεύθυνση μνήμης, τότε ελέγχεται η κρυφή μνήμη για το αν υπάρχει ήδη. Συγκεκριμένα, βρίσκεται το index και ελέγχεται αν τα tag bit είναι τα ίδια με τη διεύθυνση.
- Αν τα tag bit είναι ίδια τότε μεταφέρονται στον υπολογιστή τα δεδομένα που απαιτεί, και έχουμε ευστοχία της κρυφής μνήμης (cache hit).
- Αν τα tag bit είναι διαφορετικά, τότε η κρυφή μνήμη επικοινωνεί με το επόμενο επίπεδο να φέρει τα δεδομένα και ο επεξεργαστής περιμένει. Μόλις έρθουν τα δεδομένα στην κρυφή μνήμη τοποθετούνται εκεί και στη συνέχεια προωθούνται στον επεξεργαστή.

Τι ονομάζουμε ποσοστό ευστοχίας, αστοχίας της cache;

- Το ποσοστό ευστοχίας (hit rate) είναι το κλάσμα των αιτήσεων μνήμης που εξυπηρετήθηκαν από τη cache χωρίς να γίνει πρόσβαση σε ανώτερο επίπεδο, προς τις συνολικές αιτήσεις μνήμης του επεξεργαστή.
- Παράδειγμα: Αν στις 1000 αιτήσεις μνήμης οι 950 αιτήσεις είναι εύστοχες στην κρυφή μνήμη, το ποσοστό ευστοχίας είναι 950/1000 = 95%.
- Το ποσοστό αστοχίας (miss rate) είναι ο αριθμός των αστοχιών (cache miss) προς τον αριθμό των συνολικών αιτήσεων μνήμης του επεξεργαστή.

Αστοχίες (misses) της Cache

- Υποχρεωτικές (compulsory):
  - 1η πρόσβαση σε ένα block.
  - Μπορεί να αντιμετωπιστεί με (προμετάκληση) prefetching.
- Χωρητικότητας (capacity):
  - Η Cache δε μπορεί να έχει όλα τα block της μνήμης.
  - Τα Blocks που δε χρησιμοποιούνται εξωθούνται.
  - Αποφεύγεται με την αύξηση του μεγέθους.
- Σύγκρουσης (conflict):
  - Πολλά blocks απεικονίζονται στο ίδιο set.
  - Αποφεύγεται με αυξημένη συσχετικότητα.

Παραδείγματα αστοχιών: Compulsory

```
for(i=0; i<10; i++)
  A[i] = f(B[i]);
```

Cache(@ i=2)

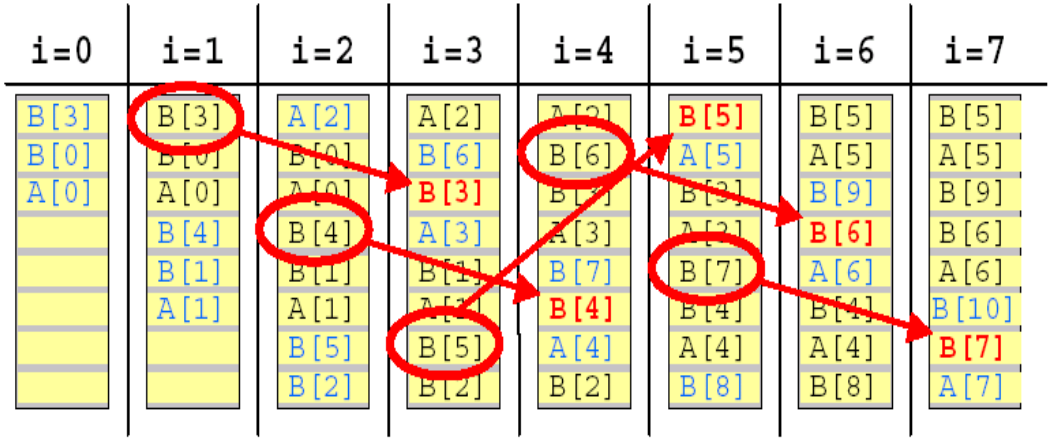
B[0]
A[0]
B[1]
A[1]
B[2]
A[2]

Cache(@ i=3)

- B[3], A[3] required
- B[3] never loaded before
  ⇒ loaded into cache
- A[3] never loaded before
  ⇒ allocates new line

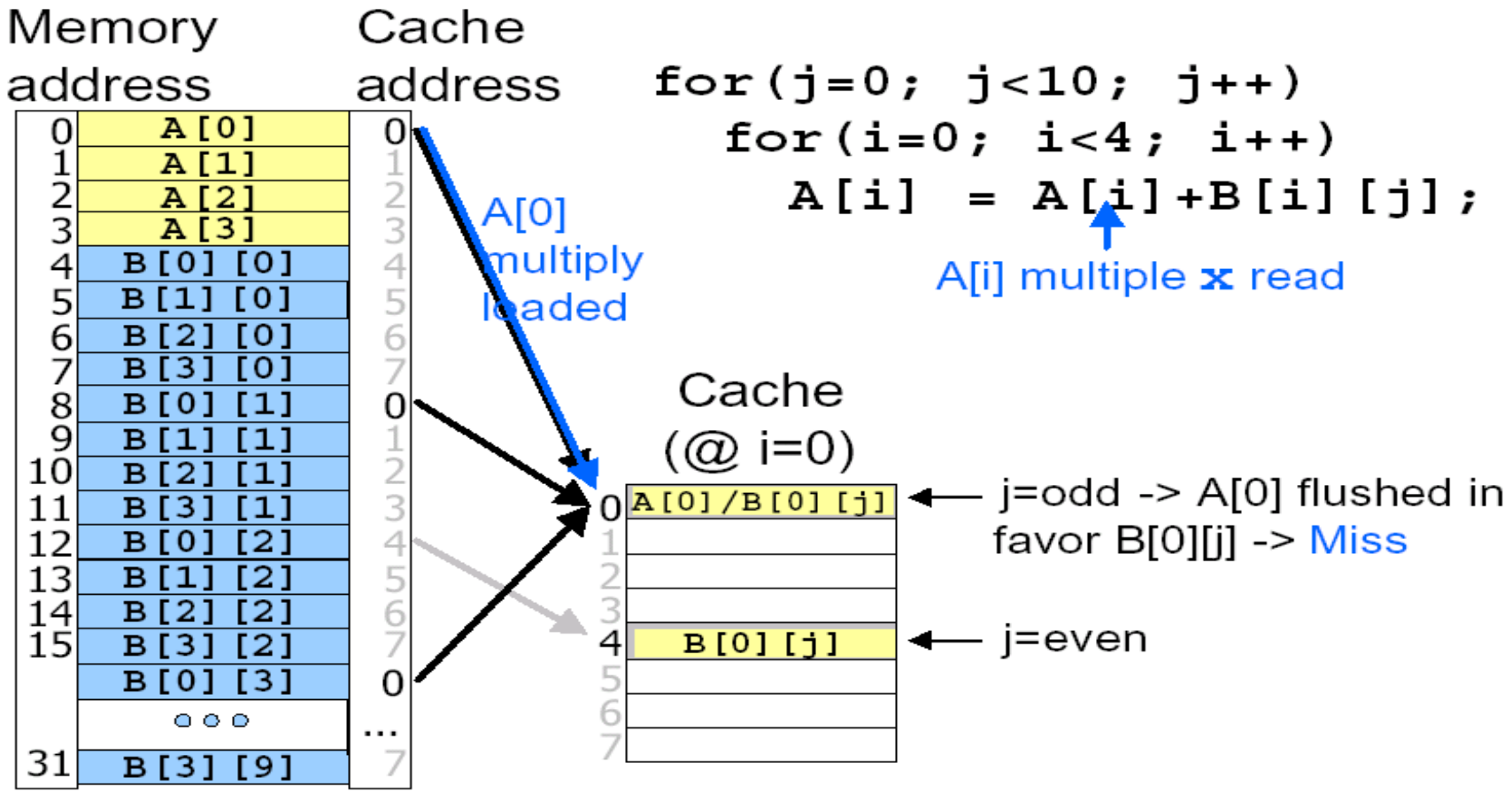
Παραδείγματα αστοχιών: Capacity

```
for(i=0; i<N; i++)
  A[i] = B[i+3]+B[i];
```

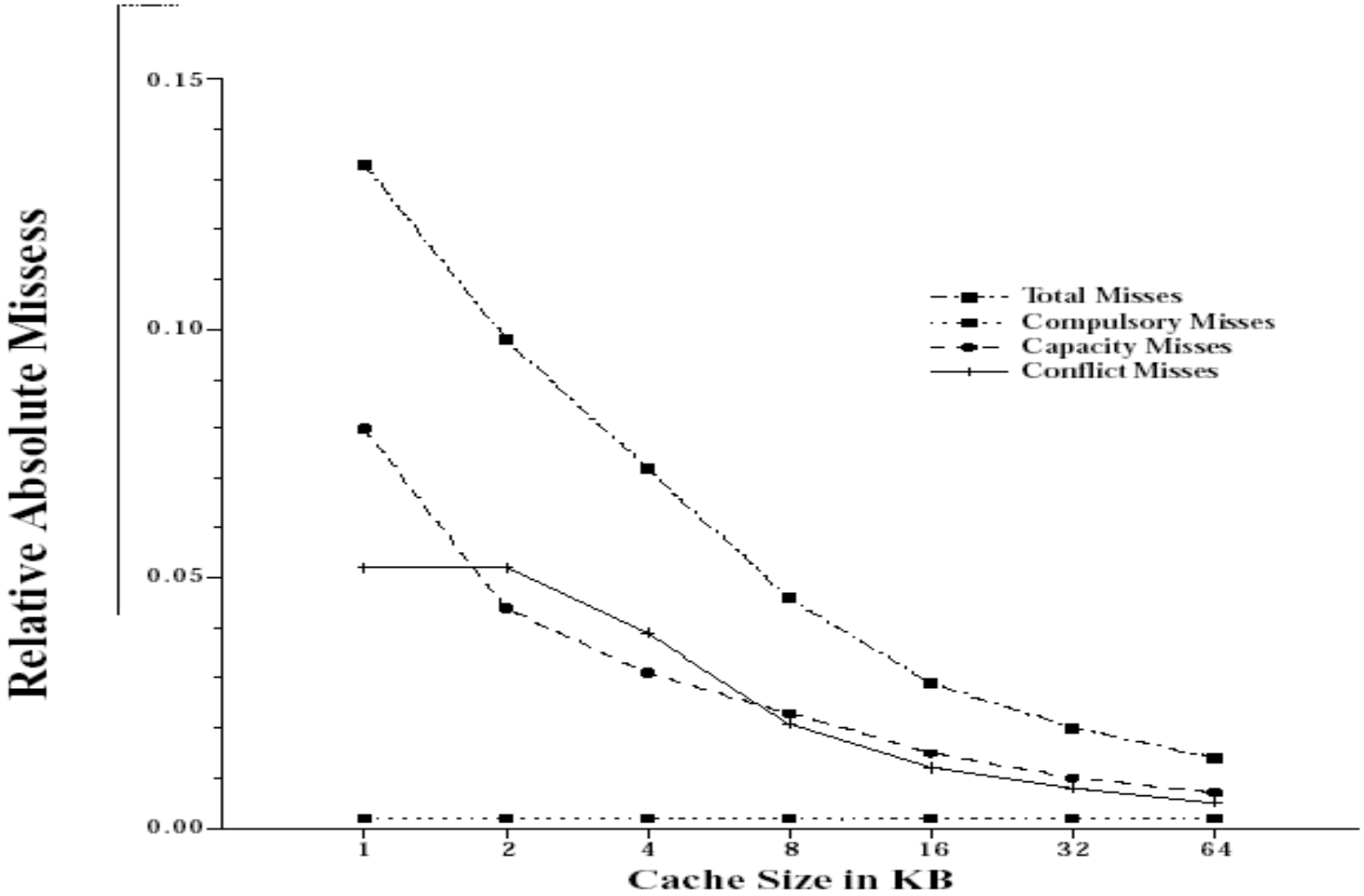


- Παράδειγμα με cache πλήρης συσχέτισης.
- Αν είχαμε μεγαλύτερη cache δε θα απομακρύνονταν το B[3], οπότε δε θα είχαμε αστοχία σε επόμενη αίτηση.

Παραδείγματα αστοχιών: Conflict

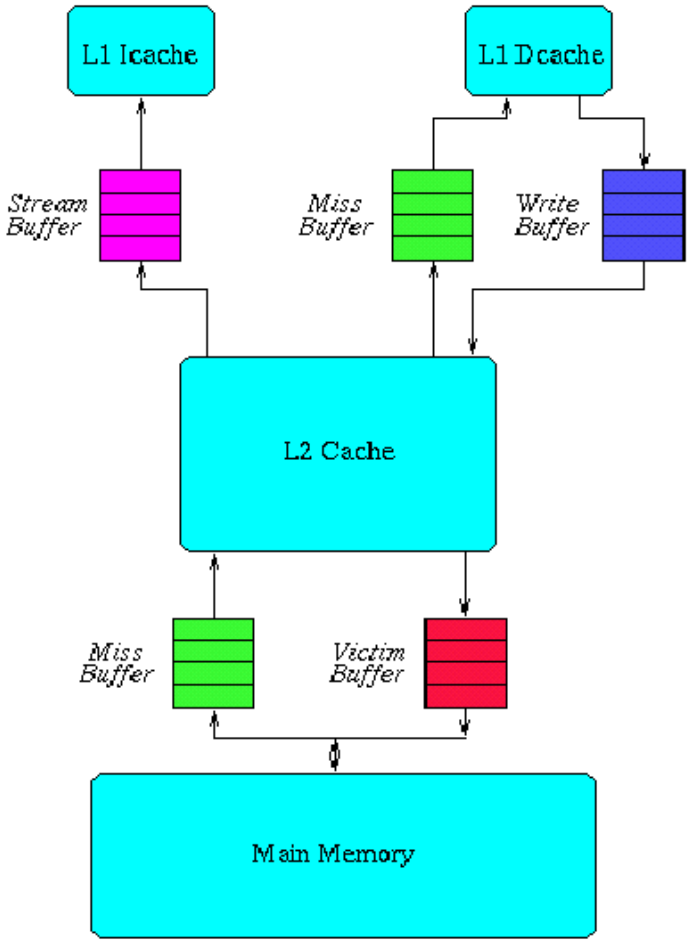


Οι 3 αστοχίες ως προς το μέγεθος της Cache



Βελτιστοποίηση της cache με επιπρόσθετα buffers

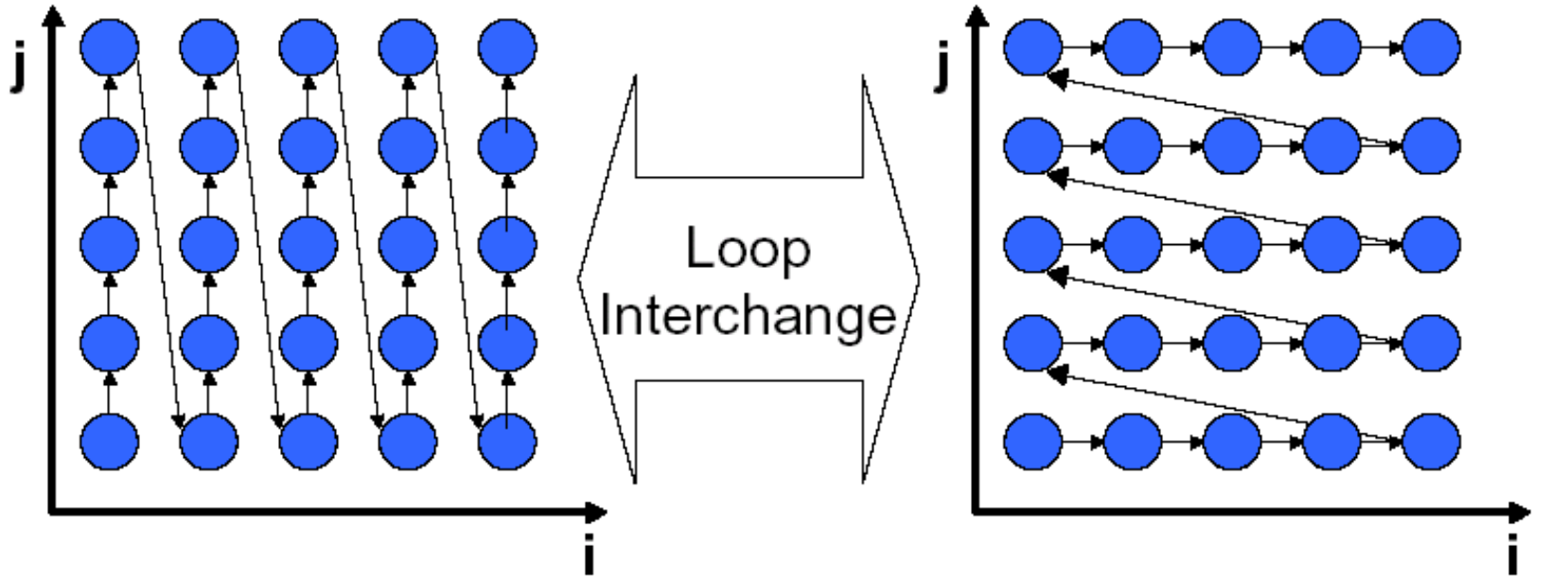
- Stream buffer: Προ ανάκληση των εντολών με παράλληλη εκτέλεση.
- Miss buffer: Κλειδωμένη-ελεύθερη λειτουργία.
- Write buffer: Απορροφά την ριπή και γράφει στον επεξεργαστή.
- Victim Buffer: Κρατάει τα τμήματα από την κρυφή μνήμη για να επιταχύνει το miss επεξεργαστή.



Βελτιστοποίηση της cache δεδομένων με αλγοριθμικούς μετασχηματισμούς

- Η cache μπορεί να βελτιστοποιηθεί χρησιμοποιώντας τους μετασχηματισμούς:
  - Loop Interchange.
  - Loop Fusion/Merge.
  - Loop Unrolling.
  - Loop Blocking/Tiling.
  - Software Prefetching.

Loop Interchange




```
for(i=0; i<W; i++)
  for(j=0; j<H; j++)
    A[i][j] = …;
```

```
for(j=0; j<H; j++)
  for(i=0; i<W; i++)
    A[i][j] = …;
```

Loop Fusion/Merge

for I_a = exp_1 to exp_2
  $A(I_a)$
for I_b = exp_1 to exp_2
  $B(I_b)$

for I = exp_1 to exp_2
  A(I)
  B(I)

Loop Unrolling

- for i=0 to 99 step 1
  b[i] = a[i] + a[i+1];

(a) original loop

- for i=0 to 49 step 2
  b[i+1] = a[i]+a[i+1];
  b[i+1] = a[i+1] + a[i+2];

(b) transformed loop

Loop blocking/tiling (1)

for I = 0 to exp_1 . exp_2

  A(I)

Tile size exp_1

Tile factor exp_2

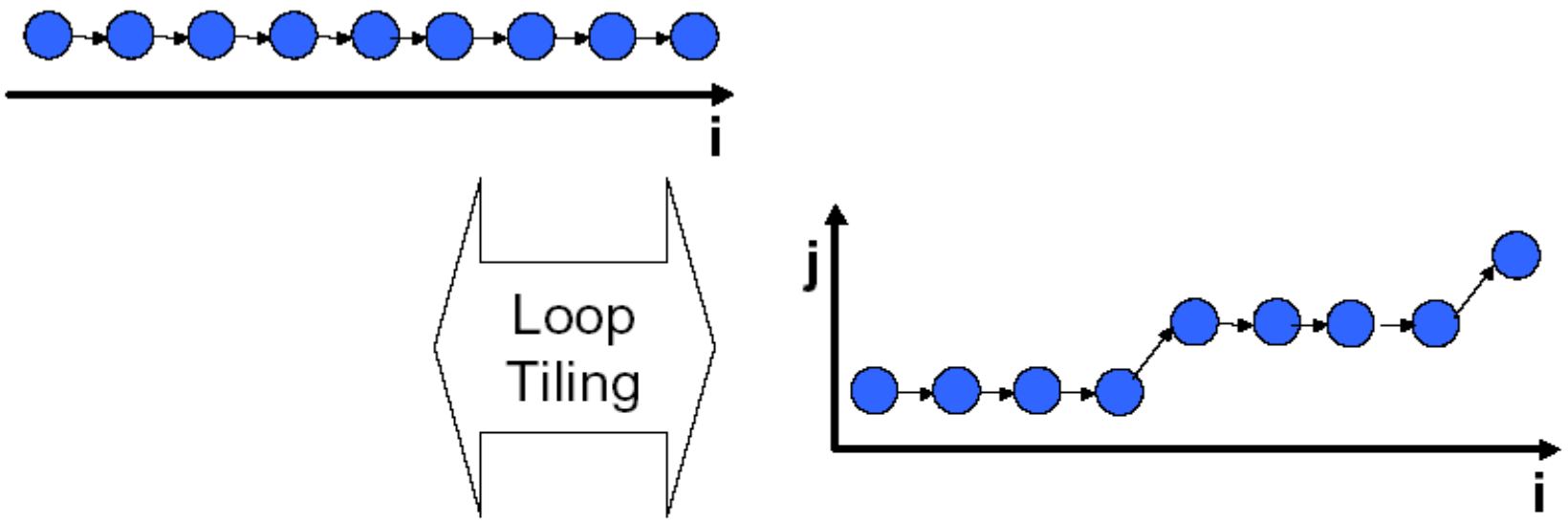
$\Longrightarrow$

for I_1 = 0 to exp_2

  for I_2 = exp_1.I_1 to (exp_1+1).I_1

    A(I_2)

Loop blocking/tiling (2)




```
for(i=0;i<9;i++)
  A[i] = ...;
```

```
for(j=0; j<3; j++)
  for(i=4*j; i<4*j+4;i++)
    if (i<9)
      A[i] = ...;
```

Ποιες είναι οι 2 πιο κοινές τακτικές ενημέρωσης της RAM

- **Εγγραφή προς τα πίσω** (write-back):

Οι πράξεις (τροποποίηση/εγγραφή) γίνονται μόνο στη μνήμη cache. Η κύρια μνήμη ενημερώνεται μόνο όταν η αντίστοιχη γραμμή της cache πρόκειται να εκδιωχθεί.

- **Εγγραφή από μέσα** (write-through):

Όλες οι πράξεις εγγραφής γίνονται στην κύρια μνήμη καθώς και στη μνήμη cache. Πάντα η κύρια μνήμη έχει έγκυρες τιμές. Μειονέκτημα: Παράγει σημαντική κυκλοφορία. Πλεονέκτημα: υπάρχει πάντα συνέπεια.

Πως αναγνωρίζεται ποιο block θα εκδιωχθεί;

- Όταν υπάρχει αστοχία της κρυφής μνήμης, τότε θα πρέπει να απομακρυνθεί μια cache line από την κρυφή μνήμη, προκειμένου να τοποθετηθούν τα νέα δεδομένα που θα έρθουν.
- Αν έχουμε cache απευθείας αντιστοίχησης τότε δεν υπάρχει δυνατότητα επιλογής.
- Αν έχουμε cache με συσχέτιση, τότε μπορεί να απομακρυνθεί μια οποιαδήποτε γραμμή από το σετ με το συγκεκριμένο index.
  - Για παράδειγμα αν έχουμε cache 2 δρόμων (2-way) τότε κάθε σετ αποτελείται από δύο γραμμές cache και μπορεί να απομακρυνθεί οποιοδήποτε από αυτές τις δύο γραμμές.

Με ποιον τρόπο αποφασίζεται ποια γραμμή θα εκδιωχθεί;

Χρησιμοποιούνται οι παρακάτω τεχνικές:

- Random - Τυχαία επιλογή.
- FIFO – η γραμμή που είχε γραφτεί πιο παλαιά.
- LRU – η γραμμή που έχει μείνει αχρησιμοποίητη για περισσότερη ώρα. Χρησιμοποιείται ένα bit (αν έχουμε 2 δρόμους) (bit χρήσης). Κάθε φορά που χρησιμοποιείται μια γραμμή από το σύνολο τότε γίνεται 1 και όλα τα use bit των άλλων γραμμών του συνόλου γίνονται 0. Για πολλαπλούς δρόμους χρησιμοποιούνται περισσότερα bit και μειώνονται κάθε φορά κατά 1. Δίνει τα καλύτερα αποτελέσματα.
- LFU – αντικαθιστά τη γραμμή που έχει χρησιμοποιηθεί τις λιγότερες φορές. Υλοποιείται με ένα απαριθμητή για κάθε γραμμή.
- Optimal – αντικαθιστά τη γραμμή που θα προκαλέσει τις λιγότερες αστοχίες στο μέλλον (δεν υλοποιείται).

Ποια τεχνική αντικατάστασης είναι καλύτερη;

	Associativity								
	Two-way			Four-way			Eight-way		
Size	LRU	Random	FIFO	LRU	Random	FIFO	LRU	Random	FIFO
16 KB	114.1	117.3	115.5	111.7	115.1	113.3	109.0	111.8	110.4
64 KB	103.4	104.3	103.9	102.4	102.3	103.1	99.7	100.5	100.3
256 KB	92.2	92.1	92.5	92.1	92.1	92.5	92.1	92.1	92.5

Figure 5.6 Data cache misses per 1000 instructions comparing least-recently used, random, and first in, first out replacement for several sizes and associativities. There is little difference between LRU and random for the largest-size cache, with LRU outperforming the others for smaller caches. FIFO generally outperforms random in the smaller cache sizes. These data were collected for a block size of 64 bytes for the Alpha architecture using 10 SPEC2000 benchmarks. Five are from SPECint2000 (gap, gcc, gzip, mcf, and perl) and five are from SPECfp2000 (applu, art, equake, lucas, and swim). We will use this computer and these benchmarks in most figures in this chapter.

Τι ονομάζεται συνοχή cache; Ποιες είναι οι δυνατές προσεγγίσεις;

- Εμφανίζεται όταν υπάρχουν ιδιωτικές cache και κοινή μνήμη σε πολυ-πύρηνα συστήματα.
- Μπορεί ένας επεξεργαστής να τροποποιήσει δεδομένα στην κύρια μνήμη, ενώ ένας άλλος επεξεργαστής να χρησιμοποιεί δεδομένα της ίδιας διεύθυνσης από τη cache, τα οποία θα είναι άκυρα.
- Αντιμετώπιση:
  - Παρακολούθηση του διαύλου μνήμης από όλες τις κρυφές μνήμες για εγγραφές.
  - Διαφάνεια, στις εγγραφές η μια cache ενημερώνει τις άλλες.
  - Κύρια μνήμη που δε μπορεί να μεταφερθεί σε cache.

Ποιο είναι το βέλτιστο μέγεθος της γραμμής cache;

- Αν γίνει πρόσβαση σε μια διεύθυνση η κρυφή μνήμη μεταφέρει και γειτονικά Byte.
- Με την αύξηση της γραμμής cache βελτιώνεται το hit rate.
- Από ένα σημείο και μετά, η περαιτέρω αύξηση μειώνει τις επιδόσεις.
- Όσο μεγαλύτερα τα τμήματα, τόσο μικρότερος ο αριθμός των γραμμών.
- Όσο μεγαλώνει το μέγεθος της γραμμής, τόσο μεγαλώνει η απόσταση των λέξεων, οπότε και η πιθανότητα να χρησιμοποιηθούν.

Τι ισχύει για τις ενοποιημένες και διαιρεμένες κρυφές μνήμες; (1/2)

- Διαιρεμένη κρυφή μνήμη:
  - Ξεχωριστή cache για τις εντολές και τα δεδομένα.
  - Επιτρέπει παράλληλη εκτέλεση εντολών.
  - Απαλείφει της διενέξεις για την cache μεταξύ της μονάδας προσκόμισης/αποκωδικοποίησης και της μονάδας εκτέλεσης.
  - Επιτρέπει καλύτερο pipeline.
  - Οι εντολές δεν απομακρύνουν δεδομένα και αντιστρόφως.
  - Η έναρξη πράξεων μνήμης γίνεται ανεξάρτητα σε κάθε μνήμη, κάτι που διπλασιάζει το εύρος ζώνης.

Τι ισχύει για τις ενοποιημένες και διαιρεμένες κρυφές μνήμες; (2/2)

- Ενοποιημένη κρυφή μνήμη
  - Απλή.
  - Εύκολα συνδέεται στο κύκλωμα.
  - Κοινή κρυφή μνήμη για εντολές και δεδομένα.
  - Δημιουργείται σημείο συμφόρησης.
  - Εξισορροπεί αυτόματα το φορτίο μεταξύ των προσκομίσεων εντολών και δεδομένων.
    - Αν υπάρχουν περισσότερα δεδομένα παρά εντολές, τότε η cache γεμίζει με δεδομένα και αντίστροφα.

Σύγκριση caches Intel Pentium 4, AMD Opteron

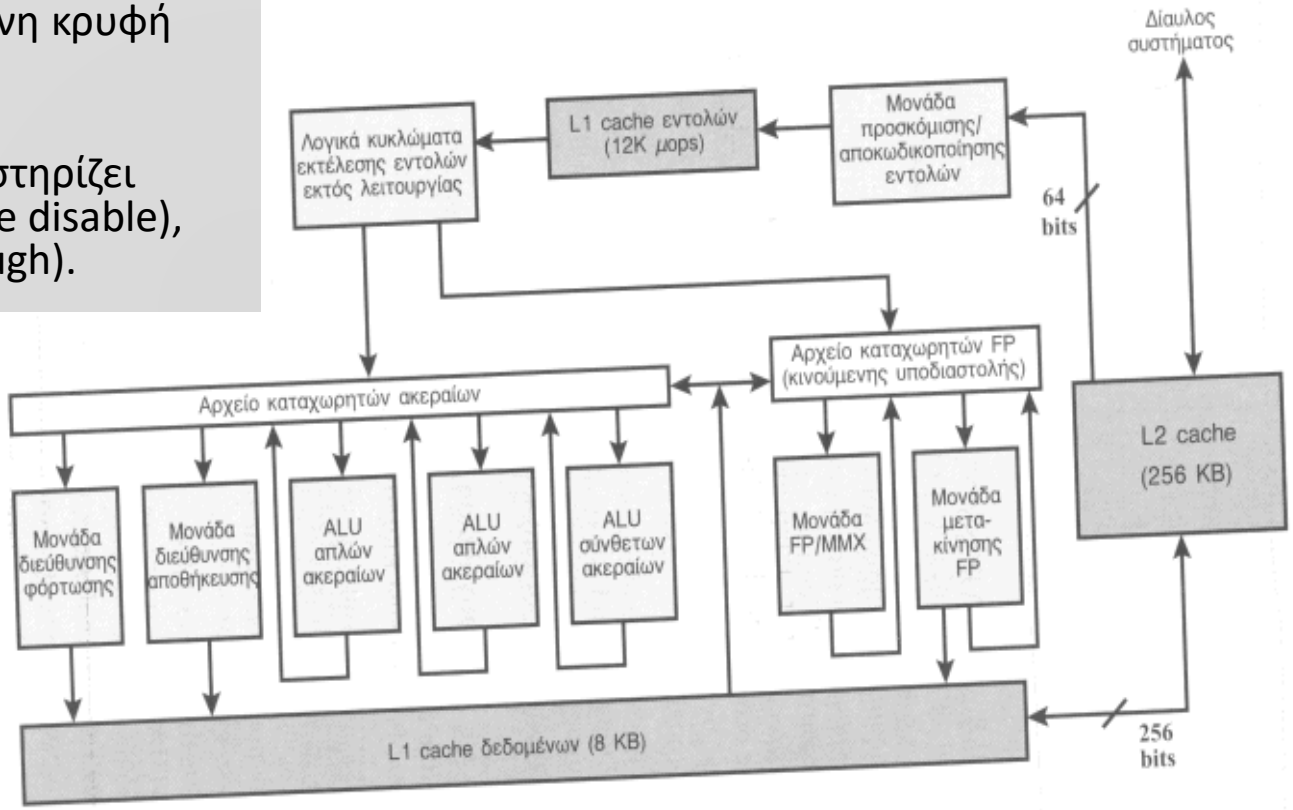
Characteristic	Intel Pentium P4	AMD Opteron
L1 cache organization	Split instruction and data caches	Split instruction and data caches
L1 cache size	8 KB for data, 96 KB trace cache for RISC instructions (12K RISC operations)	64 KB each for instructions/data
L1 cache associativity	4-way set associative	2-way set associative
L1 replacement	Approximated LRU replacement	LRU replacement
L1 block size	64 bytes	64 bytes
L1 write policy	Write-through	Write-back
L2 cache organization	Unified (instruction and data)	Unified (instruction and data)
L2 cache size	512 KB	1024 KB (1 MB)
L2 cache associativity	8-way set associative	16-way set associative
L2 replacement	Approximated LRU replacement	Approximated LRU replacement
L2 block size	128 bytes	64 bytes
L2 write policy	Write-back	Write-back

Intel Pentium IV (1/2)

- Intel Pentium4 (2003)
  - Application: desktop/server
  - Technology: 90nm (1/100x)
  - 55M transistors (20,000x)
  - 101 mm^2 (10x)
  - 3.4 GHz (10,000x)
  - 1.2 Volts (1/10x)
  - 32/64-bit data (16x)
  - 22-stage pipelined datapath
    - Later: 31-stage pipeline
  - 3 or 4 instructions per cycle (superscalar)
  - Two levels of on-chip cache
  - data-parallel (SIMD) instructions, hyperthreading

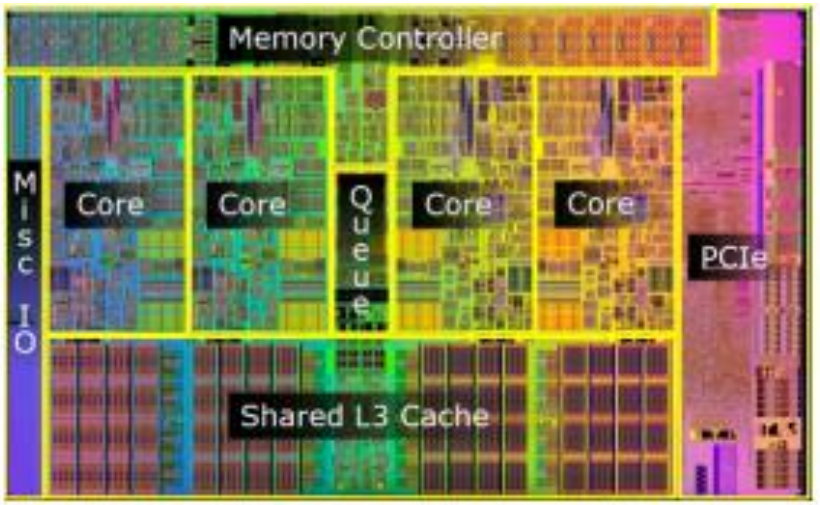
Intel Pentium IV (2/2)

- Οι Pentium ήταν οι πρώτοι x86 που είχαν διαιρεμένη κρυφή μνήμη.
- L1 data cache: Υποστηρίζει writeback, cd (cache disable), nw (not write-through).



Intel Core i7

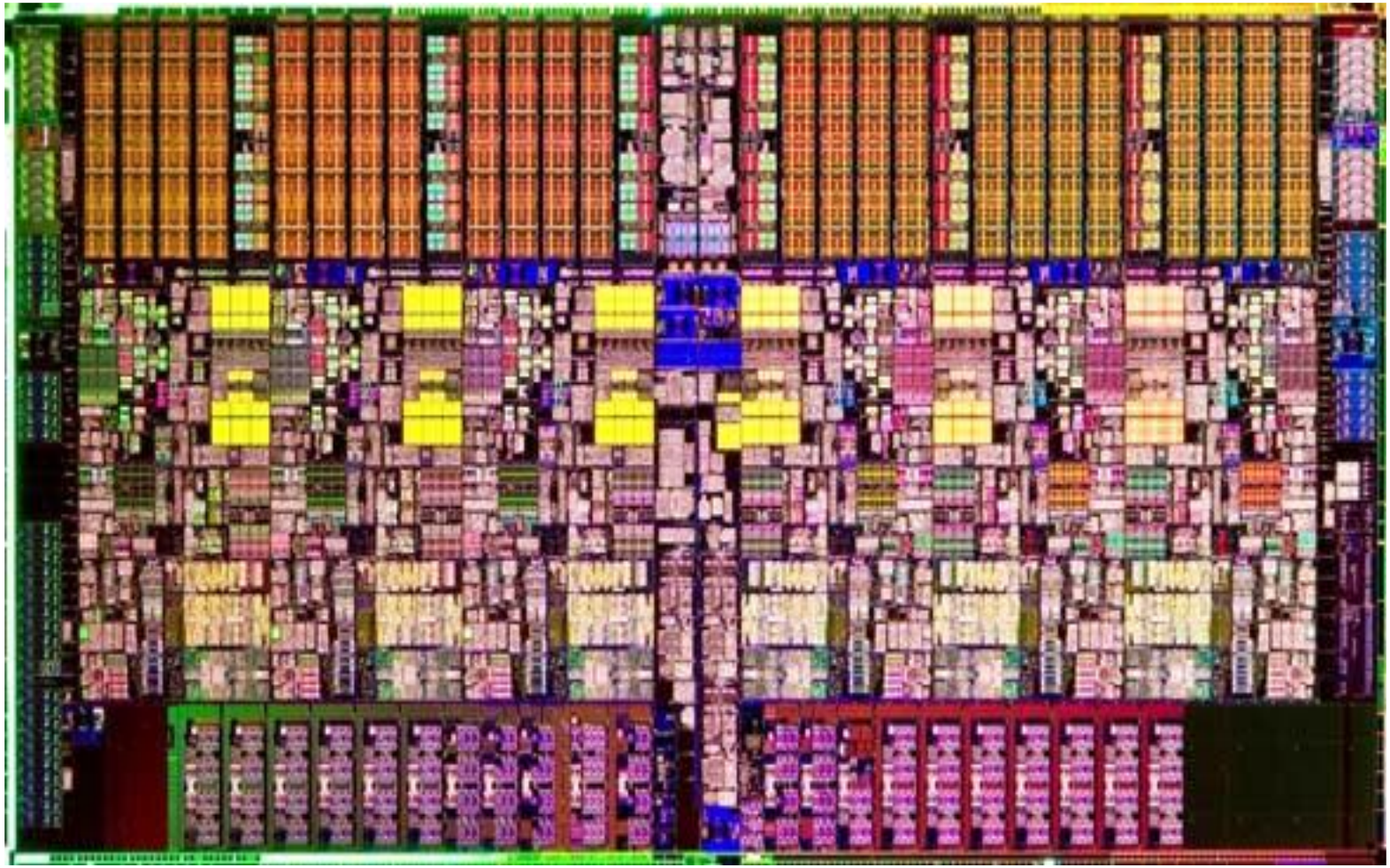
- Intel Core i7 (2009)
  - Application: desktop/server
  - Technology: 45nm (1/2x)



  - 774M transistors (12x)
  - 296 mm^2 (3x)
  - 3.2 GHz to 3.6 Ghz (~1x)
  - 0.7 to 1.4 Volts (~1x)

  - 128-bit data (2x)
  - 14-stage pipelined datapath (0.5x)
  - 4 instructions per cycle (~1x)
  - Three levels of on-chip cache
  - data-parallel (SIMD) instructions, hyperthreading
  - **Four-core multicore** (4x)

Floorplan I7



Άσκηση CACHE (1/3)

- Έστω έχουμε τις παρακάτω προσπελάσεις μνήμης με τη σειρά που δίνονται: 22,26, 22,26,16,3,16,18. Έχουμε 5 bit address bus. Μια κρυφή μνήμη L1 άμεσης αντιστοίχησης με 8 γραμμές του 1 Byte.
- Να συμπληρώσετε τον παρακάτω πίνακα:

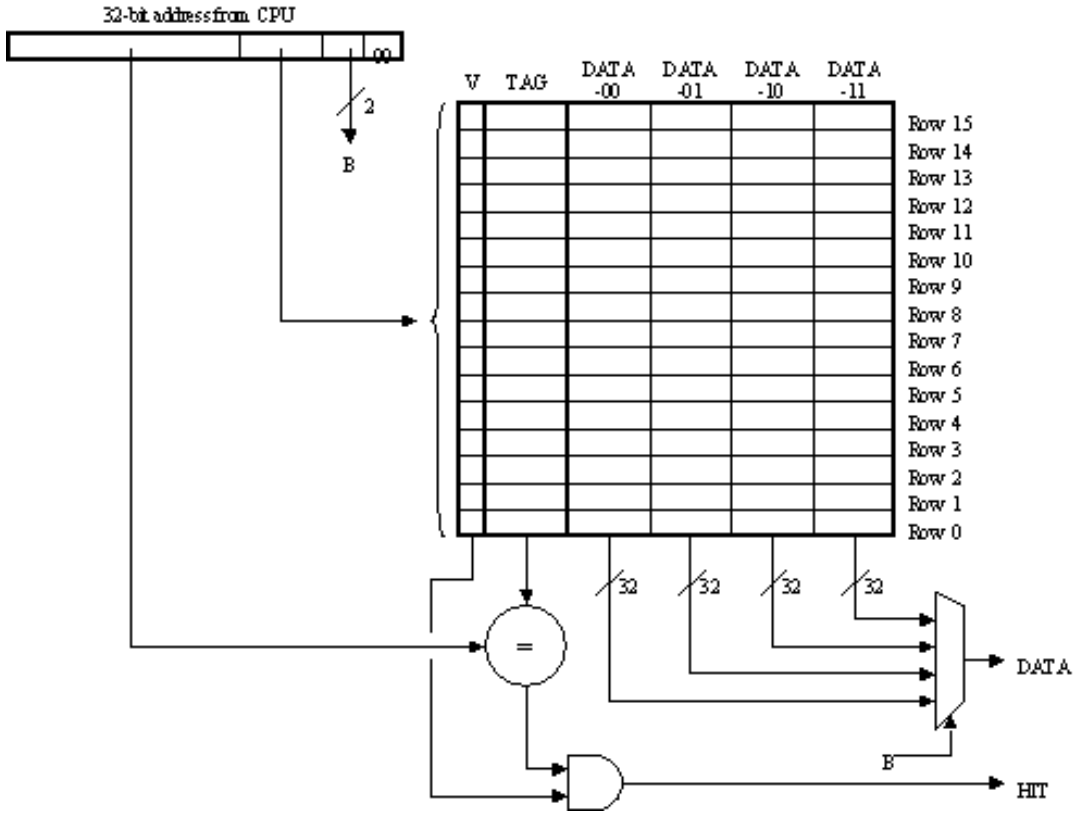
Πρόσβαση	Γραμμή Cache	Ευστοχία/Αστοχία	Tagcache

- Υπολογίστε το ποσοστό ευστοχίας.
- Δώστε την εικόνα των γραμμών της cache ύστερα από την 18η πρόσβαση.
- Πόσα bit κατ ελάχιστο απαιτεί μια cache line για δεδομένα 32bit.

Άσκηση CACHE (2/3)

- Θεωρήστε μια κρυφή μνήμη με 64 γραμμές, άμεσης απεικόνισης, με 16 Byte σε κάθε γραμμή.
- Που αντιστοιχίζεται η διεύθυνση 1200 (0x4B0);
- Ποιο είναι το tag/index/displacement για την παραπάνω διεύθυνση;
- Ποια άλλα Byte θα μεταφερθούν αν έχουμε αστοχία κρυφής μνήμης για τη συγκεκριμένη πρόσβαση;

Άσκηση CACHE (3/3)



- Μέγιστος αριθμός λέξεων
- Πόσα bit κάθε πεδίο.
- Που βρίσκεται η 0x2045C.
- Μπορούν οι διευθύνσεις 0x12368 και 0x322FF8 να είναι ταυτόχρονα στη cache;
- Πόσα Byte μεταφέρονται σε κάθε miss;

Η μνήμη TLB
(Translation Lookaside Buffer)

Τι είναι το TLB; (1/2)

- Είναι cache.
- Χρησιμοποιείται για την βελτίωση της ταχύτητας μετάφρασης διευθύνσεων μνήμης.
- Η μορφή της μνήμης είναι CAM (Content addressable memory).
- Η είσοδος είναι η virtual address και η έξοδος είναι η φυσική διεύθυνση της RAM.
- Είναι μια πολύ γρήγορη μνήμη.
- Αν γίνει miss τότε πρέπει να γίνει η πολύπλοκη διαδικασία υπολογισμού της διεύθυνσης.

Τι είναι το TLB; (2/2)

- Έχει συγκεκριμένη χωρητικότητα σε γραμμές.
- Κάνει την αντιστοίχηση διευθύνσεις μνήμης προγράμματος -> πραγματικές διευθύνσεις.
- Βρίσκεται ανάμεσα στο CPU και τη μνήμη.
- Χρησιμοποιείται όταν πρέπει να χρησιμοποιηθεί off-chip μνήμη. Αυτό θα συμβεί όταν συμβεί:
  - Instruction cache miss.
  - Data case miss.
  - TLB miss.

Τι είναι η Shadow Cache;

- Όταν απομακρύνεται μια γραμμή τότε αποθηκεύεται η διεύθυνση που αντιστοιχεί σε μια ειδική μνήμη που ονομάζεται shadow cache.
- Μερικές φορές απομακρύνονται γραμμές που χρησιμοποιούνται συνεχώς, από κάποιες άλλες μη συχνά χρησιμοποιούμενες προσβάσεις.
- Οι πληροφορίες που βρίσκονται σε αυτή τη μνήμη μπορεί να βοηθήσουν στην απόφαση ποια γραμμή θα απομακρύνουμε, όταν χρειαστεί.
- Οι διευθύνσεις που βρίσκονται σε αυτή τη μνήμη δεν απομακρύνονται αν υπάρχουν και στη cache.

Τι είναι το write buffer;

- Όταν υπάρχει ένα write miss τότε η cache καθυστερεί μέχρι να φέρει τη γραμμή και να τη γράψει.
- Μπορεί να αρθεί η καθυστέρηση αν χρησιμοποιηθεί ένα write buffer, μια ειδικού τύπου μνήμη, η οποία βρίσκεται ανάμεσα στο L1 και στο L2.
- Αποθηκεύει προσωρινά τα δεδομένα που θα γραφούν στη cache.
- Μάλιστα αν υπάρξει και ανάγνωση από την ίδια διεύθυνση, μπορεί να εξυπηρετηθεί από το write buffer.

Τι είναι το victim buffer;

- Κρατάει τις τελευταίες γραμμές που έχουν απομακρυνθεί.
- Αν συμβεί μια αστοχία ανάγνωσης και υπάρχουν τα δεδομένα στο victim buffer, τότε αυτά πολύ γρήγορα επαναφέρονται στη κρυφή μνήμη.
- Μπορεί να θεωρηθεί ότι αυξάνει και τη συσχετικότητα σε γραμμές που έχουν ιδιαίτερο φόρτο.
- Βοηθάει να αποφύγουμε την καθυστέρηση από την εκδίωξη συχνά χρησιμοποιούμενων γραμμών.

Write-allocate / write-no-allocate

- Όταν συμβεί μια αστοχία εγγραφής σε μια κρυφή μνήμη μπορεί να αντιμετωπιστεί με δυο τρόπους:
  - Write-allocate: Τα δεδομένα γράφονται στην κρυφή μνήμη κάνοντας έγκυρη τη συγκεκριμένη γραμμή (χρησιμοποιείται συνήθως στη write-back μνήμη).
  - Write-no-allocate: Τα δεδομένα γράφονται στο επόμενο επίπεδο, παρακάμπτοντας το συγκεκριμένο επίπεδο cache (π.χ. L1) και σημειώνοντας τη γραμμή μη έγκυρη (την L1), αφού δεν έχουν γραφεί σε αυτή τη cache (χρησιμοποιείται συνήθως στη write-through μνήμη).

Τέλος Ενότητας