



Εθνικό Μετσόβιο Πολυτεχνείο  
Σχολή Ηλεκτρολόγων Μηχ. και Μηχανικών Υπολογιστών  
Εργαστήριο Υπολογιστικών Συστημάτων

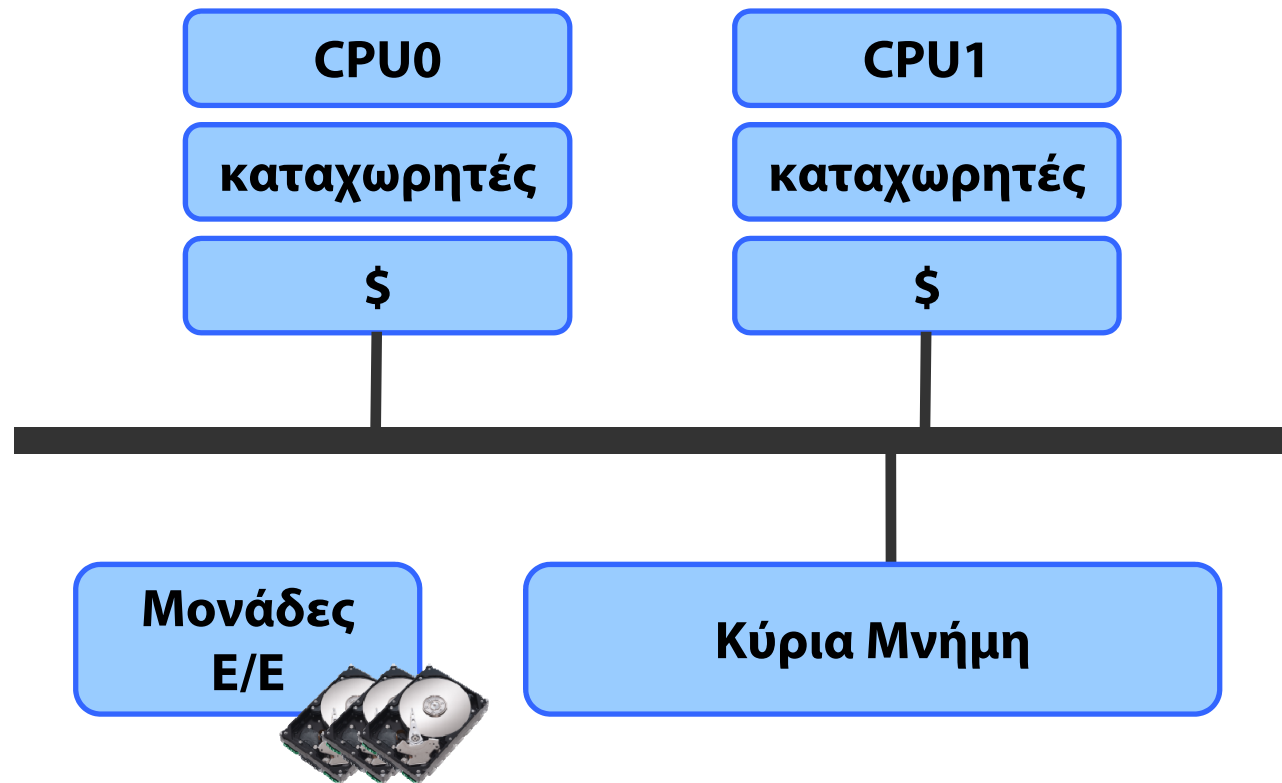
# Διαχείριση Κύριας Μνήμης

Λειτουργικά Συστήματα Υπολογιστών  
6ο Εξάμηνο, 2019-2020

# Διαχείριση Κύριας Μνήμης - Σύνοψη

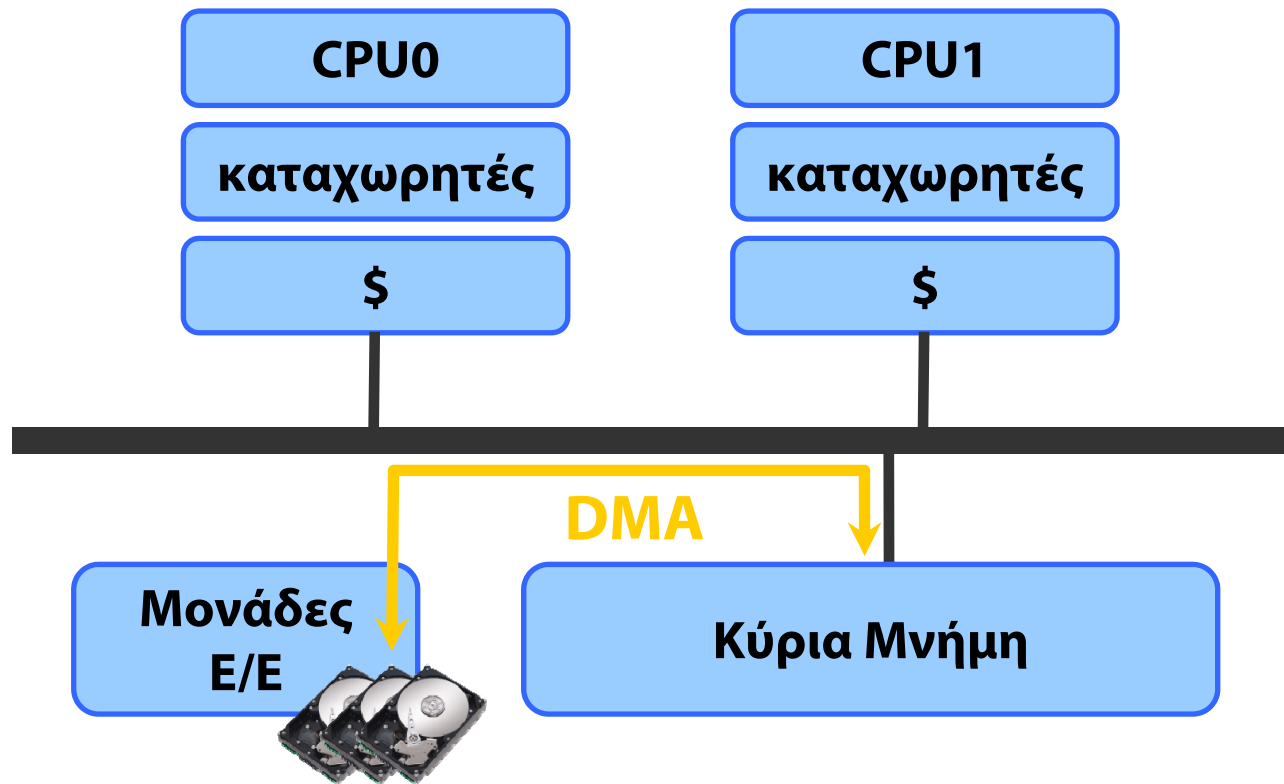
- ◆ Ιεραρχία μνήμης
- ◆ Μεταγλώττιση – φόρτωση – εκτέλεση κώδικα
- ◆ Καθορισμός διευθύνσεων
- ◆ Εναλλαγή διεργασιών
- ◆ Συνεχόμενη ανάθεση μνήμης
  - ➔ Στρατηγικές κατανομής, κατακερματισμός
- ◆ Σελιδοποίηση
  - ➔ Μετάφραση διευθύνσεων, πίνακες σελίδων, TLBs
  - ➔ Οργάνωση πινάκων σελίδων
- ◆ Κατάτμηση

# Κύρια Μνήμη (1)



- ◆ Κάθε CPU αναφέρεται απευθείας σε καταχωρητές και μνήμη
- ◆ Συσκευές E/E εκτελούν Απευθείας Πρόσβαση στη Μνήμη (Direct Memory Access - DMA)

# Κύρια Μνήμη (1)



- ◆ Κάθε CPU αναφέρεται απευθείας σε καταχωρητές και μνήμη
- ◆ Συσκευές Ε/Ε εκτελούν Απευθείας Πρόσβαση στη Μνήμη (Direct Memory Access - DMA)

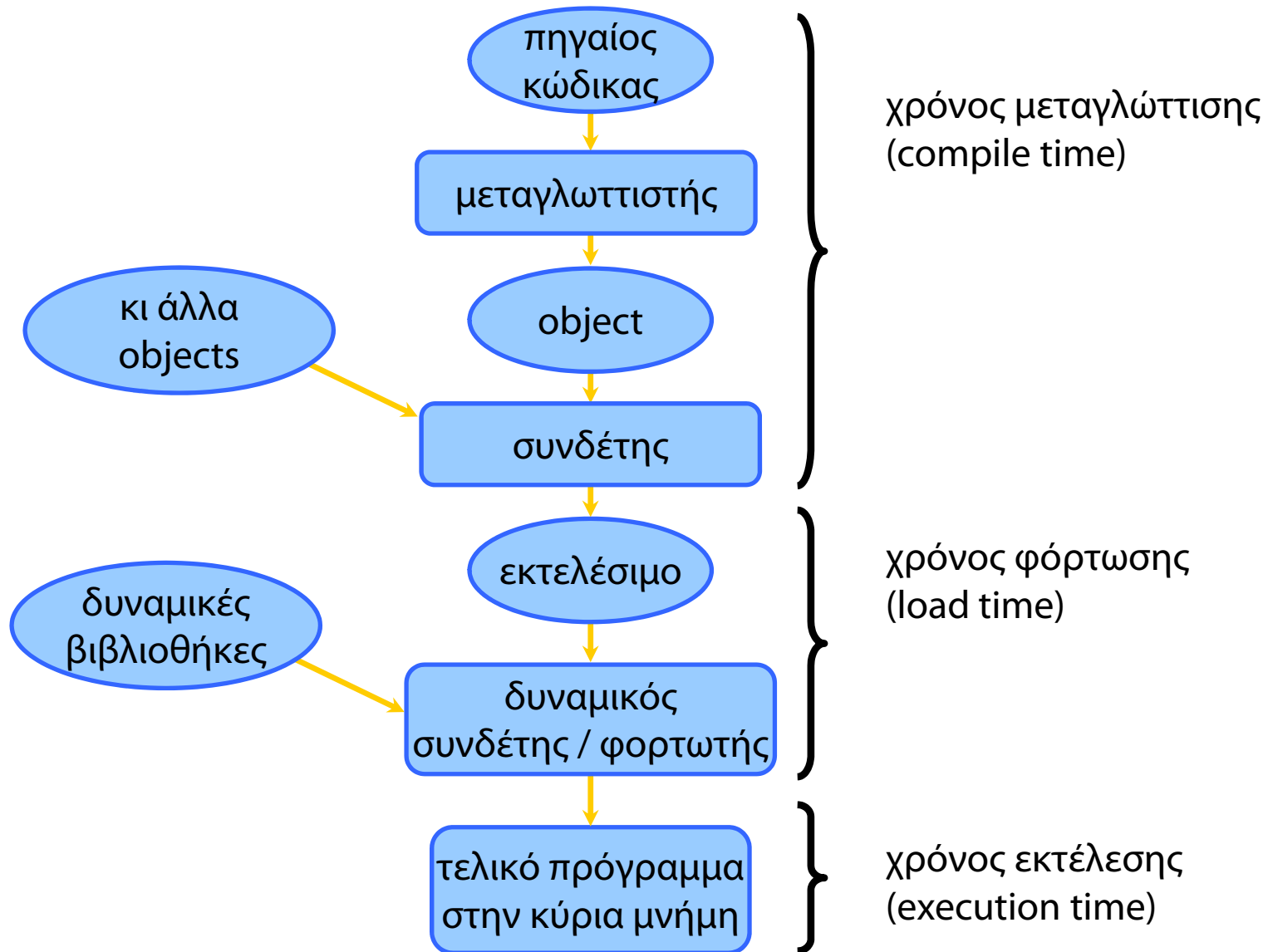
# Κύρια Μνήμη (2)

- ◆ Τα προγράμματα βρίσκονται στην Κύρια Μνήμη όταν εκτελούνται
  - ➔ Ένα εκτελέσιμο φορτώνεται από το δίσκο...
  - ➔ ... και εκτελείται μέσα σε μια διεργασία
- ◆ Η CPU αναφέρεται απευθείας μόνο σε καταχωρητές και διευθύνσεις μνήμης
  - ➔ Η κρυφή μνήμη μειώνει το κόστος πρόσβασης
    - Όταν υπάρχει χωρική ή χρονική τοπικότητα
- ◆ Πόσα προγράμματα είναι ταυτόχρονα φορτωμένα;
  - ➔ Πολυπρογραμματισμός
  - ➔ Ανάγκη για *προστασία μνήμης*

# Διαχείριση Κύριας Μνήμης - Σύνοψη

- ◆ Ιεραρχία μνήμης
- ◆ Μεταγλώττιση – φόρτωση – εκτέλεση κώδικα
- ◆ Καθορισμός διευθύνσεων
- ◆ Εναλλαγή διεργασιών
- ◆ Συνεχόμενη ανάθεση μνήμης
  - ➔ Στρατηγικές κατανομής, κατακερματισμός
- ◆ Σελιδοποίηση
  - ➔ Μετάφραση διευθύνσεων, πίνακες σελίδων, TLBs
  - ➔ Οργάνωση πινάκων σελίδων
- ◆ Κατάτμηση

# Μεταγλώττιση – Φόρτωση - Εκτέλεση



# Καθορισμός διευθύνσεων - Address Binding

- ◆ Address Binding: μεταβλητές και συναρτήσεις (σύμβολα) → διευθύνσεις μνήμης
- ◆ **Πότε;** Ξέρουμε πού θα βρίσκεται το πρόγραμμα στη μνήμη;
  - ➔ Στο χρόνο μεταγλώττισης:  
απόλυτος κώδικας
  - ➔ Στο χρόνο φόρτωσης:  
μετατοπίσιμος (relocatable) κώδικας
  - ➔ Στο χρόνο εκτέλεσης: με την υποστήριξη ειδικού υλικού για μετάφραση διευθύνσεων
    - λογικές / εικονικές διευθύνσεις → φυσικές διευθύνσεις



# Απόλυτος κώδικας

## Πηγαίος Κώδικας

```
int val;  
  
int func(void) {  
    val = 5;  
    func2();  
}  
  
int func2(void) {  
    ...  
}
```

Μεταγλωττιστής



## Τελικός Κώδικας

```
func ≡ 64    movl $0x5, 7010  
              70    call $5000  
              76    ret  
              ...  
func2 ≡ 5000    ... func2...  
  
val ≡ 7010    ... 4 bytes...
```

- ◆ Ακριβής, απόλυτος προσδιορισμός θέσεων μνήμης για τα σύμβολα (func, func2, val)
- ◆ Ο παραγόμενος κώδικας ξέρει πού βρίσκεται και πού είναι τα δεδομένα του

# Μετατοπίσιμος κώδικας (1)

## Πηγαίος Κώδικας

```
int val;  
  
int func(void) {  
    val = 5;  
    func2();  
}  
  
int func2(void) {  
    ...  
}
```

Μεταγλωττιστής



- ◆ Ο παραγόμενος κώδικας δεν ξέρει πού βρίσκεται
- ◆ πίνακας μετατόπισης (relocation table)

# Μετατοπίσιμος κώδικας (1)

## Πηγαίος Κώδικας

```
int val;  
  
int func(void) {  
    val = 5;  
    func2();  
}  
  
int func2(void) {  
    ...  
}
```

Μεταγλωττιστής



## Μετατοπίσιμος κώδικας (πχ. object)

```
func  movl $0x5, val  
      call func2  
      ret  
func2 ... func2...  
val   ... 4 bytes...
```

- ◆ Ο παραγόμενος κώδικας δεν ξέρει πού βρίσκεται
- ◆ πίνακας μετατόπισης (relocation table)

# Μετατοπίσιμος κώδικας (1)

## Πηγαίος Κώδικας

```
int val;  
  
int func(void) {  
    val = 5;  
    func2();  
}  
  
int func2(void) {  
    ...  
}
```

Μεταγλωττιστής



## Μετατοπίσιμος κώδικας (πχ. object)

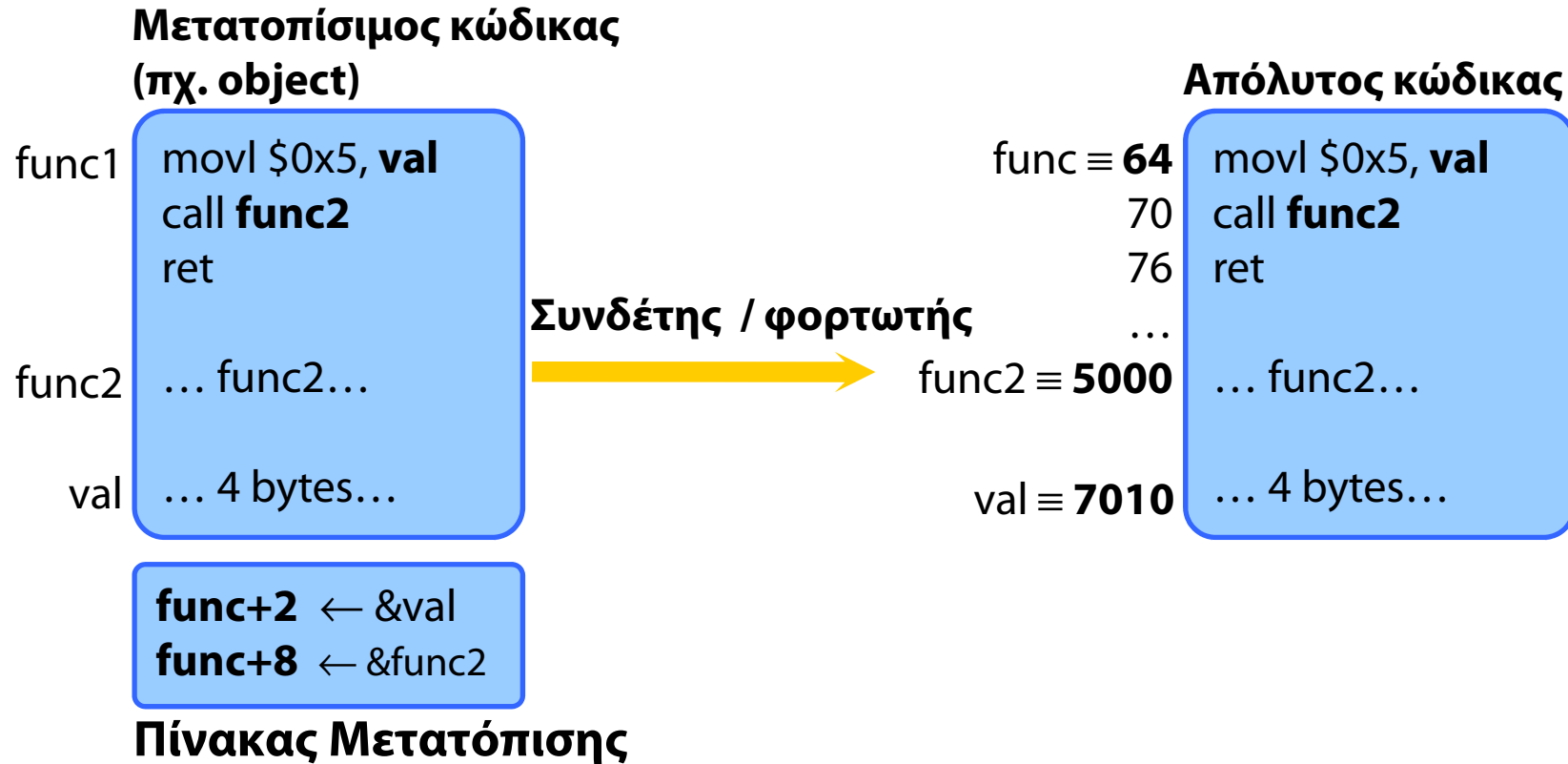
```
func  movl $0x5, val  
      call func2  
      ret  
func2  ... func2...  
val   ... 4 bytes...
```

```
func+2 ← &val  
func+8 ← &func2
```

## Πίνακας Μετατόπισης

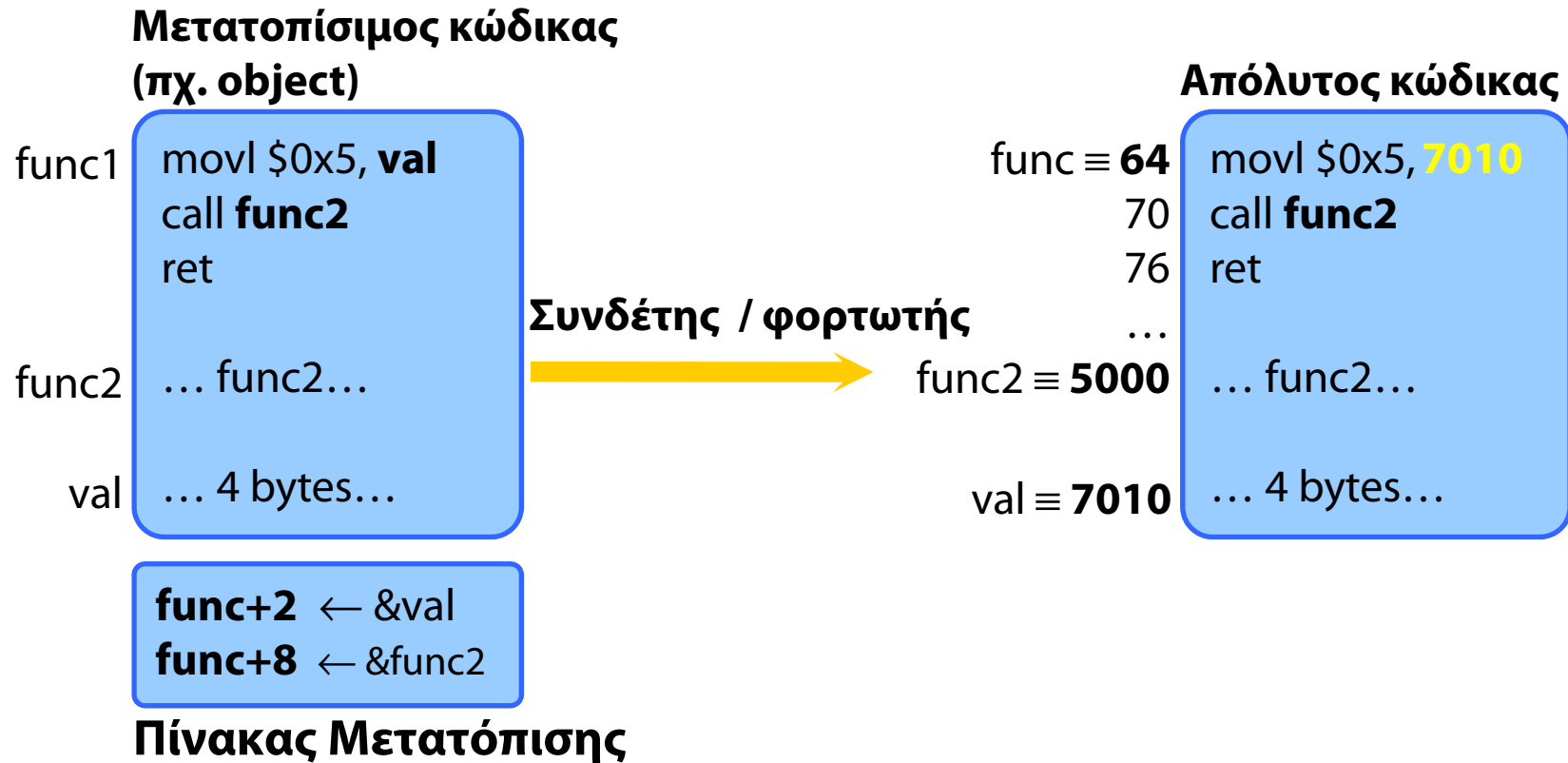
- ◆ Ο παραγόμενος κώδικας δεν ξέρει πού βρίσκεται
- ◆ πίνακας μετατόπισης (relocation table)

# Μετατοπίσιμος κώδικας (2)



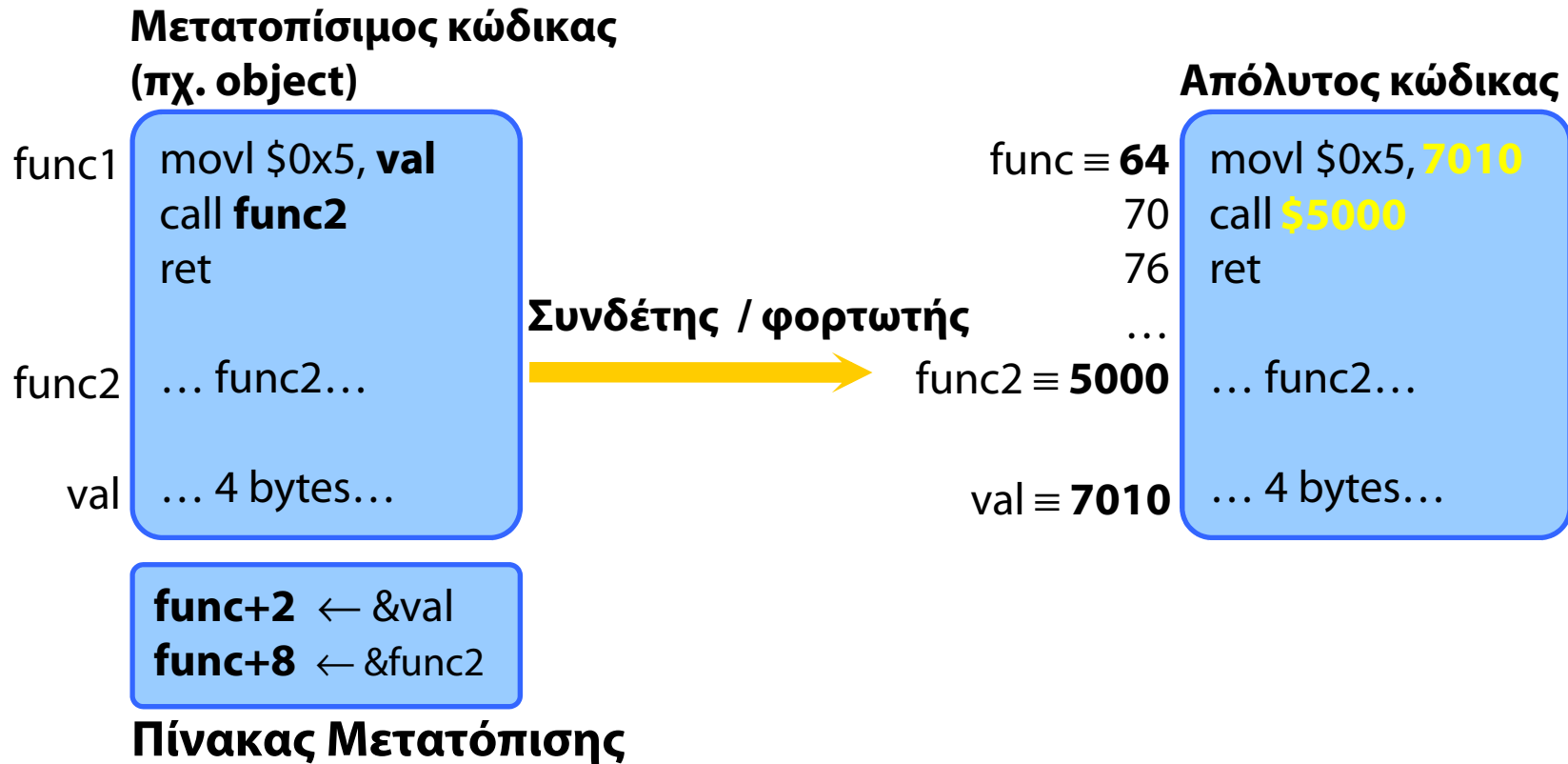
- ◆ Ο κώδικας διορθώνεται με βάση τον πίνακα μετατόπισης
- ◆ Στο χρόνο μεταγλώττισης (συνδέτης) ή εκτέλεσης (φορτωτής)

# Μετατοπίσιμος κώδικας (2)



- ◆ Ο κώδικας διορθώνεται με βάση τον πίνακα μετατόπισης
- ◆ Στο χρόνο μεταγλώττισης (συνδέτης) ή εκτέλεσης (φορτωτής)

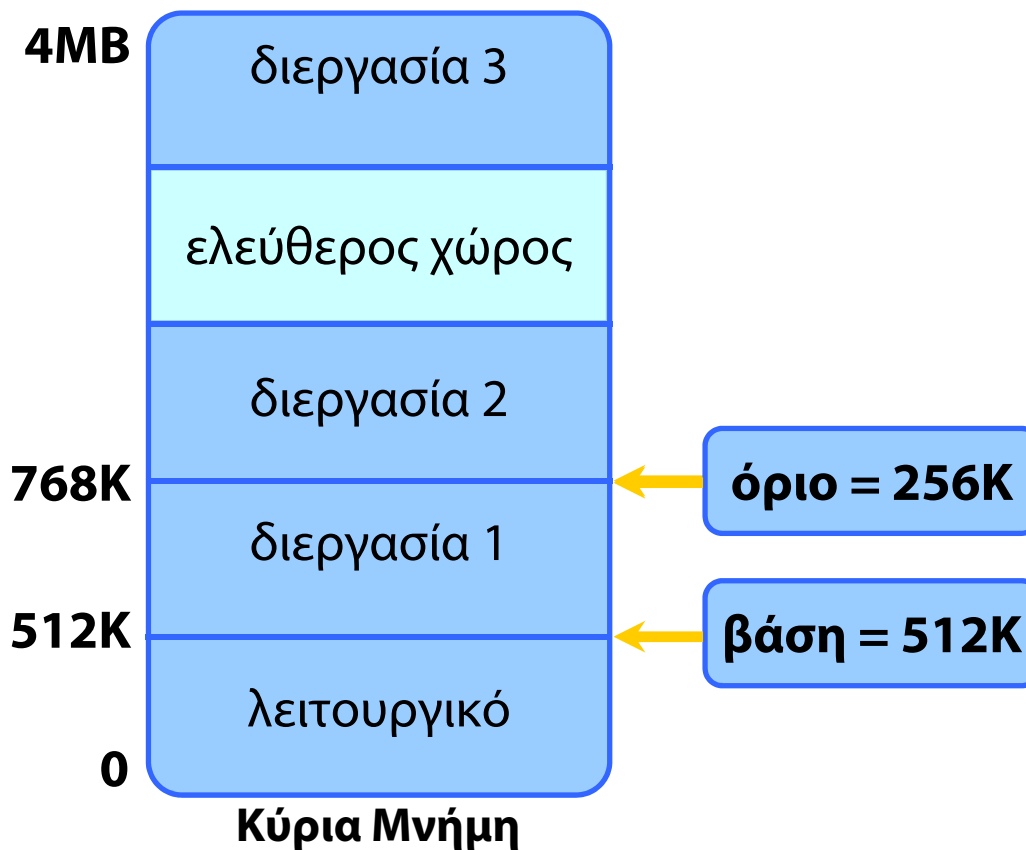
# Μετατοπίσιμος κώδικας (2)



- ◆ Ο κώδικας διορθώνεται με βάση τον πίνακα μετατόπισης
- ◆ Στο χρόνο μεταγλώττισης (συνδέτης) ή εκτέλεσης (φορτωτής)

# Μετάφραση στο χρόνο εκτέλεσης (1)

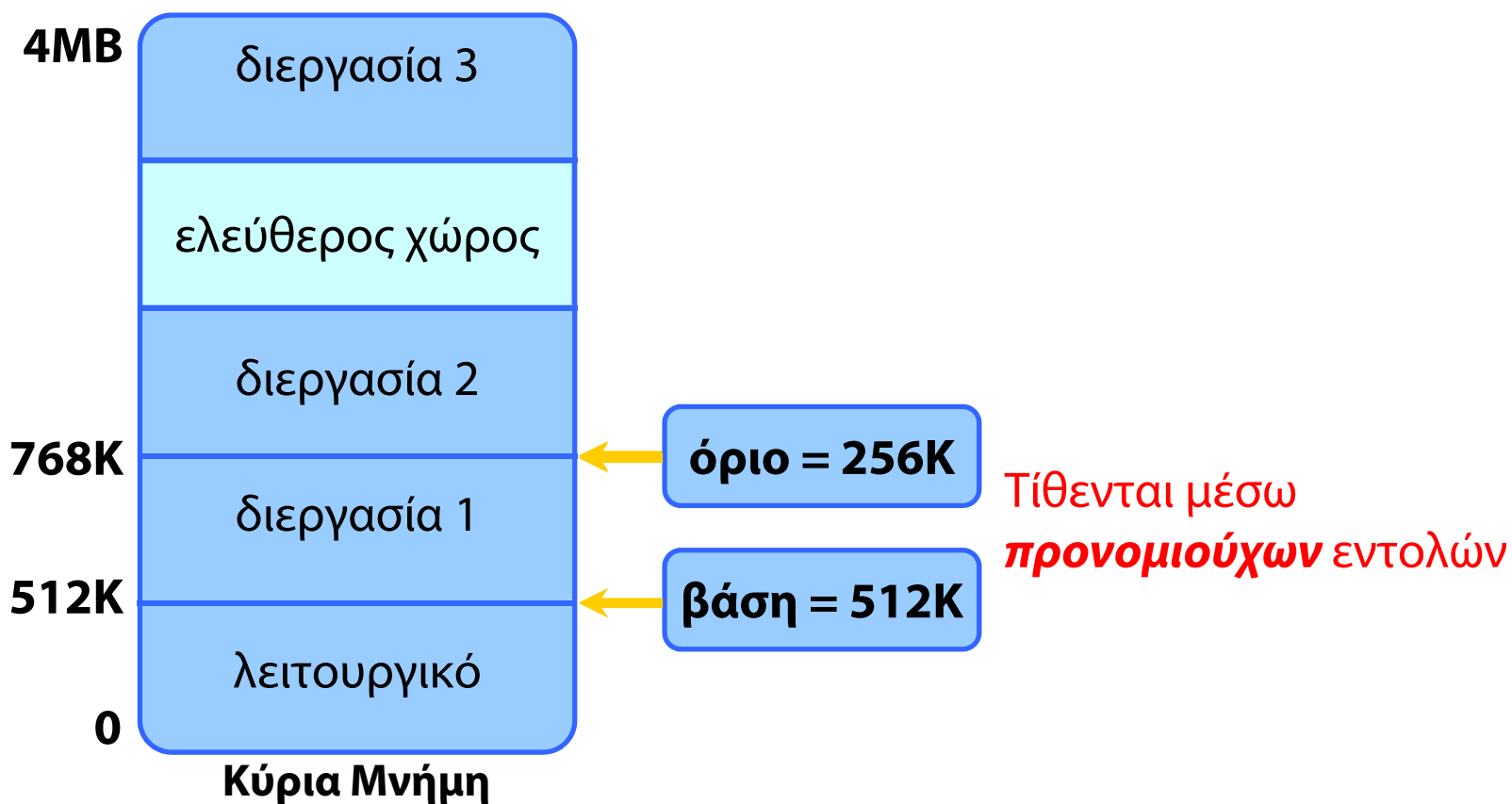
- ◆ Χωριστές περιοχές μνήμης ανά διεργασία
  - ➔ Καταχωρητής βάσης, καταχωρητής ορίου
  - ➔ Δυνατότητα πολυπρογραμματισμού με προστασία μνήμης





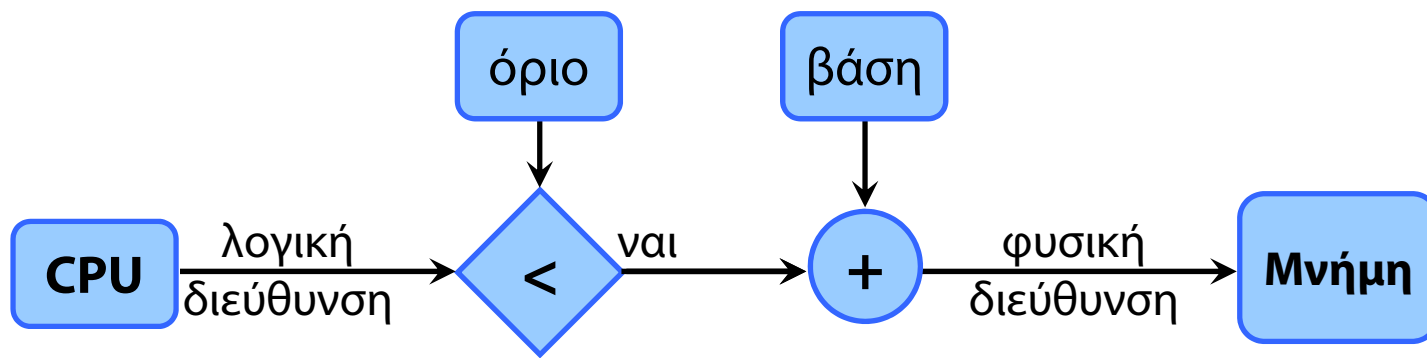
# Μετάφραση στο χρόνο εκτέλεσης (1)

- ◆ Χωριστές περιοχές μνήμης ανά διεργασία
  - ➔ Καταχωρητής βάσης, καταχωρητής ορίου
  - ➔ Δυνατότητα πολυπρογραμματισμού με προστασία μνήμης



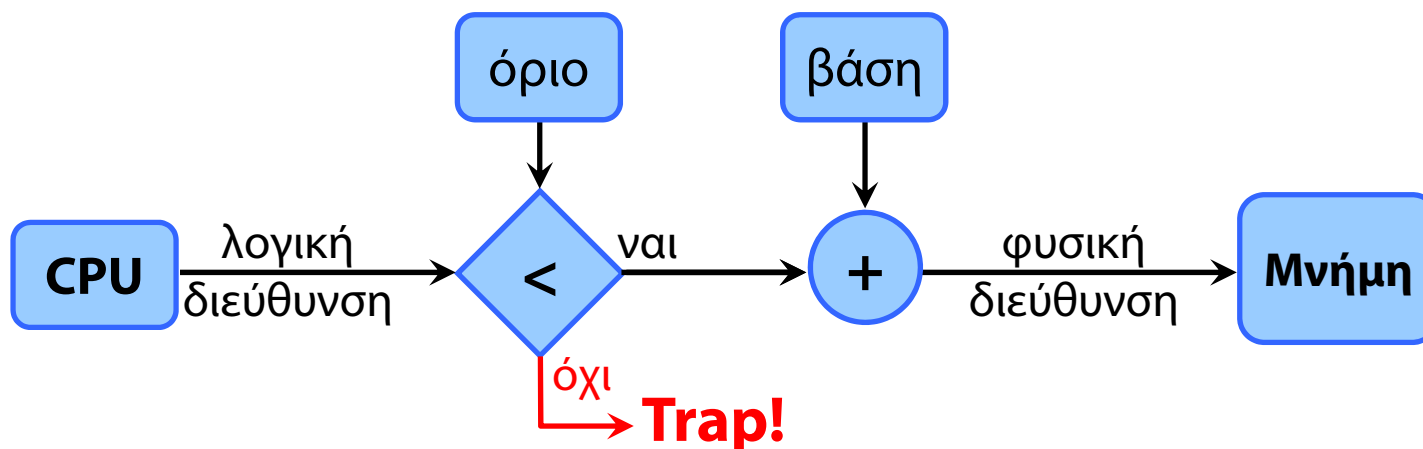
## Μετάφραση στο χρόνο εκτέλεσης (2)

- ◆ *Λογικές / εικονικές διευθύνσεις*: διευθύνσεις που βλέπει ο κώδικας που εκτελείται
- ◆ *Φυσικές διευθύνσεις*: διευθύνσεις που βλέπει η κύρια μνήμη πάνω στο διάδρομο
- ◆ Ειδικό υλικό για τη μετάφραση
  - ➔ Μονάδα διαχείρισης μνήμης – MMU
- ◆ Για την απλή περίπτωση «βάση – όριο»

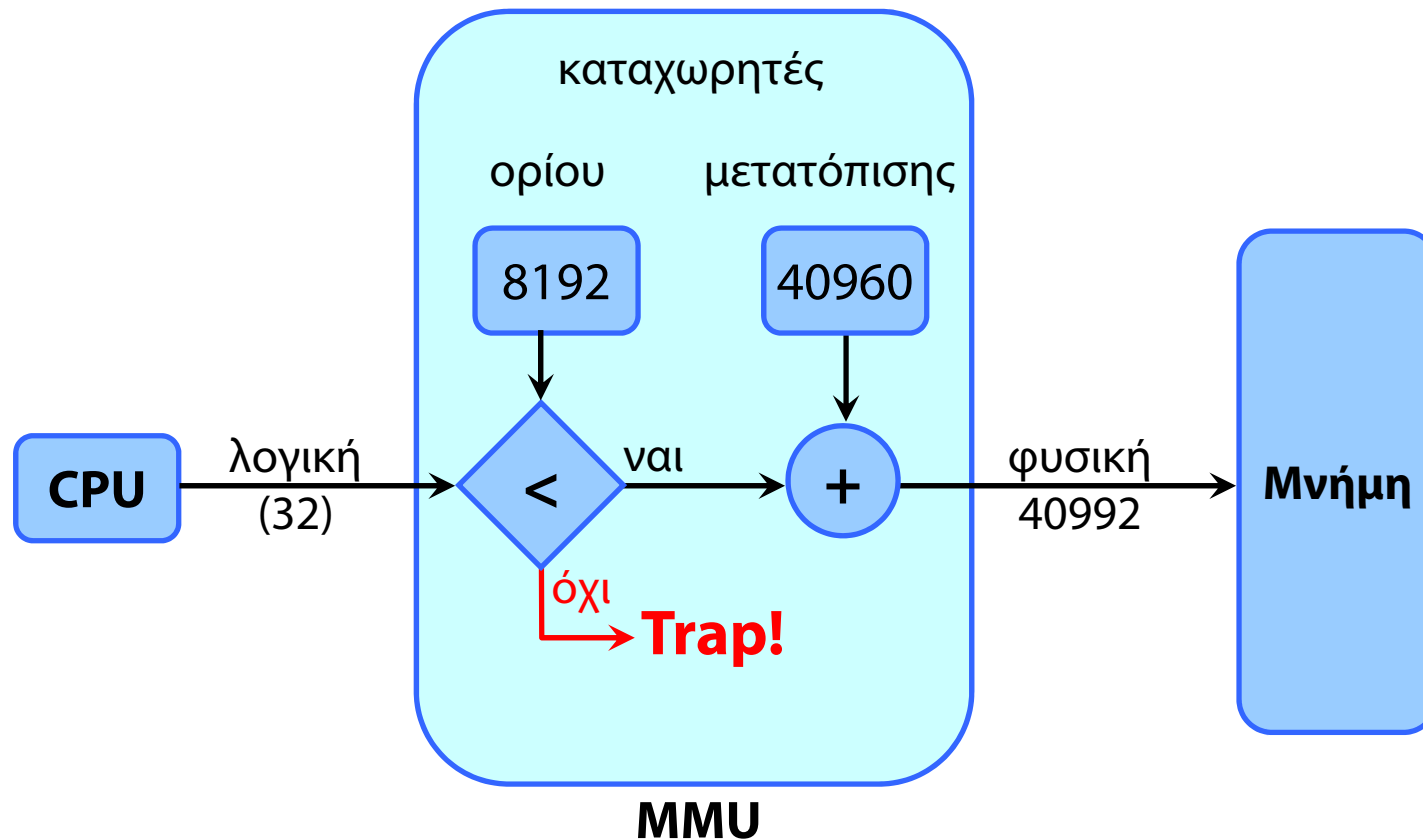


## Μετάφραση στο χρόνο εκτέλεσης (2)

- ◆ *Λογικές / εικονικές διευθύνσεις*: διευθύνσεις που βλέπει ο κώδικας που εκτελείται
- ◆ *Φυσικές διευθύνσεις*: διευθύνσεις που βλέπει η κύρια μνήμη πάνω στο διάδρομο
- ◆ Ειδικό υλικό για τη μετάφραση
  - ➔ Μονάδα διαχείρισης μνήμης – MMU
- ◆ Για την απλή περίπτωση «βάση – όριο»



# Μετάφραση στο χρόνο εκτέλεσης (3)



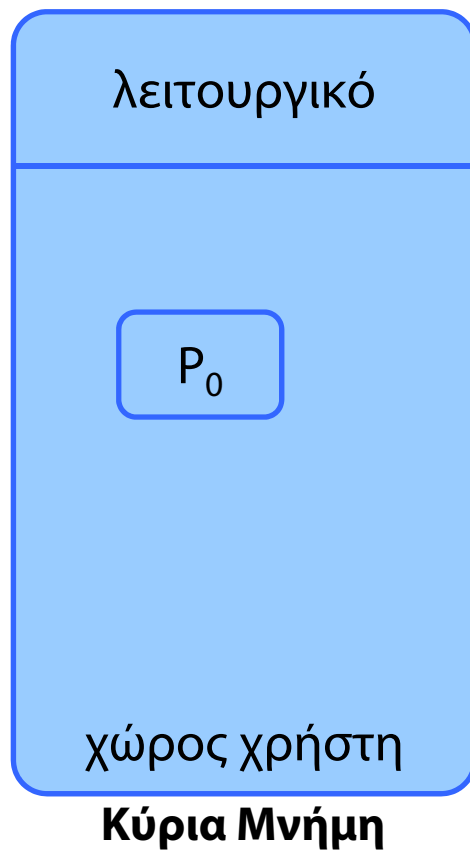
# Διαχείριση Κύριας Μνήμης - Σύνοψη

- ◆ Ιεραρχία μνήμης
- ◆ Μεταγλώττιση – φόρτωση – εκτέλεση κώδικα
- ◆ Καθορισμός διευθύνσεων
- ◆ **Εναλλαγή διεργασιών**
- ◆ Συνεχόμενη ανάθεση μνήμης
  - ➔ Στρατηγικές κατανομής, κατακερματισμός
- ◆ Σελιδοποίηση
  - ➔ Μετάφραση διευθύνσεων, πίνακες σελίδων, TLBs
  - ➔ Οργάνωση πινάκων σελίδων
- ◆ Κατάτμηση

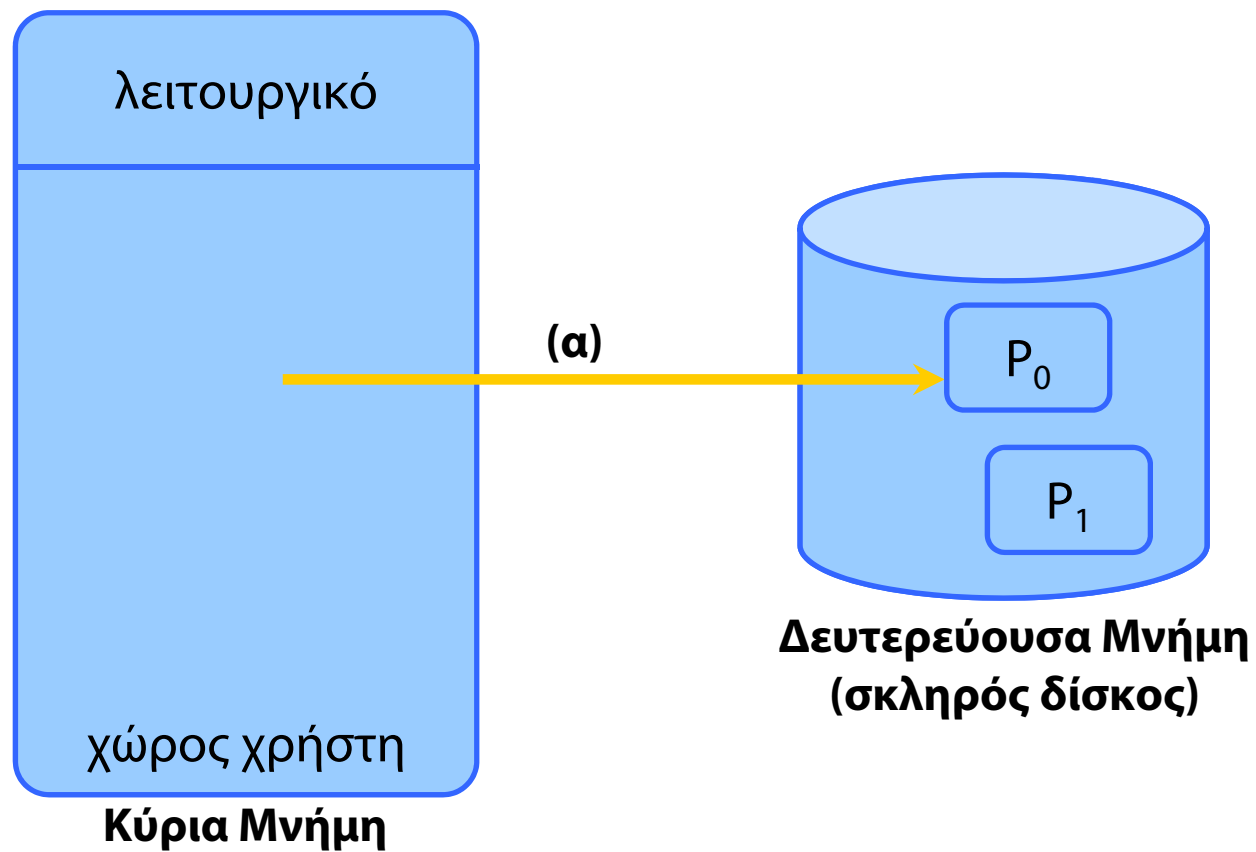
# Εναλλαγή (swapping) (1)

- ◆ Οι συνολικές απαιτήσεις μνήμης των διεργασιών, μπορεί να ξεπερνούν το μέγεθος της κύριας μνήμης
- ◆ *Εναλλαγή*: διεργασίες φεύγουν από τη μνήμη και πάνε στο δίσκο
- ◆ Το ΛΣ τις επαναφέρει όταν μπορούν να συνεχίσουν
- ◆ *Roll out, roll in*: αν έρθει διεργασία με υψηλότερη προτεραιότητα, στείλε μία χαμηλότερης προτεραιότητας στο δίσκο
- ◆ Ο δίσκος είναι πολύ πιο αργός από την ΚΜ
  - ➔ Κόστος ανάλογο του μεγέθους της μνήμης της διεργασίας
- ◆ Όταν μια διεργασία επανέλθει, πού πηγαίνει;
  - ➔ Δέσμευση διευθύνσεων στο χρόνο εκτέλεσης

# Εναλλαγή (swapping) (2)

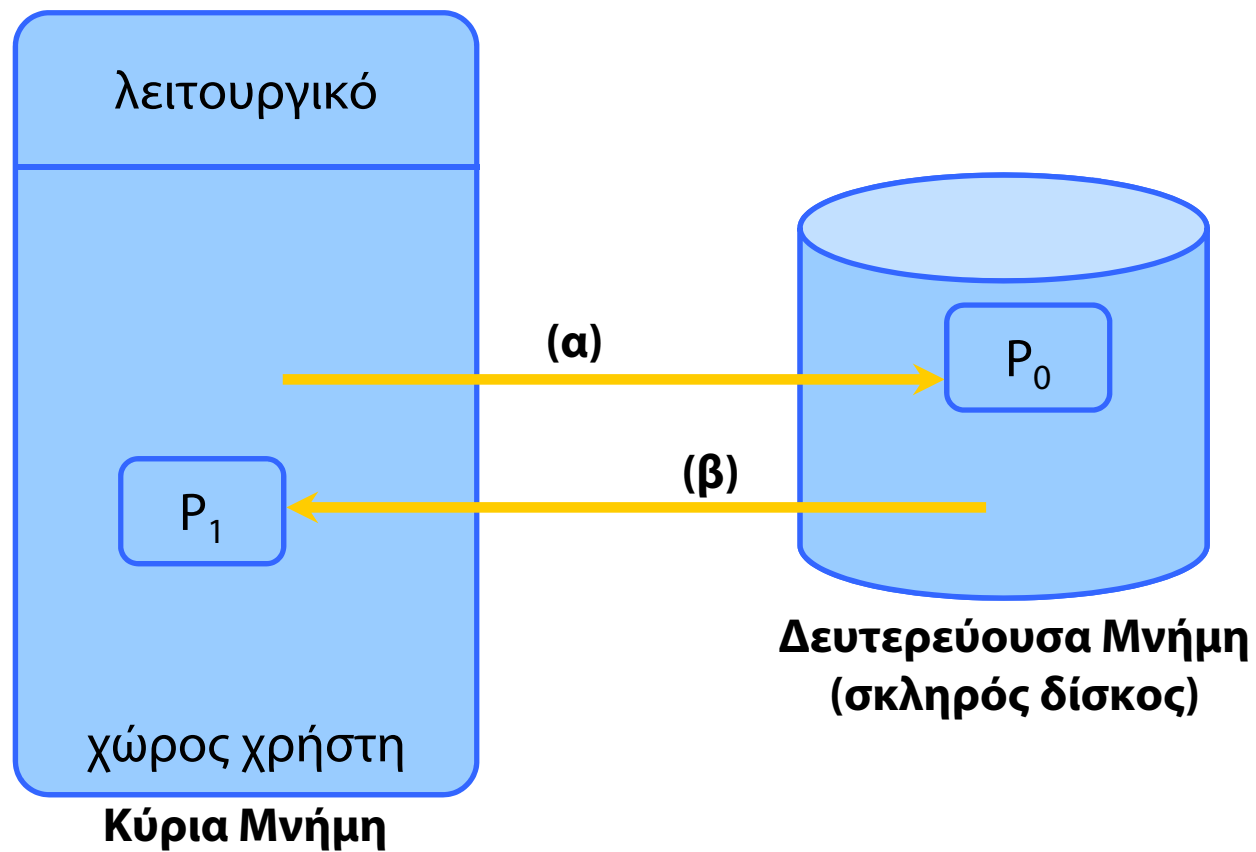


# Εναλλαγή (swapping) (2)





# Εναλλαγή (swapping) (2)



# Διαχείριση Κύριας Μνήμης - Σύνοψη

- ◆ Ιεραρχία μνήμης
- ◆ Μεταγλώττιση – φόρτωση – εκτέλεση κώδικα
- ◆ Καθορισμός διευθύνσεων
- ◆ Εναλλαγή διεργασιών
- ◆ **Συνεχόμενη ανάθεση μνήμης**
  - ➔ **Στρατηγικές κατανομής, κατακερματισμός**
- ◆ Σελιδοποίηση
  - ➔ Μετάφραση διευθύνσεων, πίνακες σελίδων, TLBs
  - ➔ Οργάνωση πινάκων σελίδων
- ◆ Κατάτμηση

# Συνεχόμενη ανάθεση μνήμης (1)

- ◆ Υποθέσεις
  - ➔ Το ΛΣ παραμένει μονίμως φορτωμένο σε ένα εύρος διευθύνσεων (συνήθως χαμηλές)
  - ➔ Κάθε νέα διεργασία έχει συγκεκριμένες απαιτήσεις σε μνήμη
- ◆ Το ΛΣ φορτώνει τη διεργασία σε συνεχές (contiguous) τμήμα μνήμης για να εκτελεστεί
- ◆ Απλή μέθοδος
  - ➔ προस्ताσία μνήμης με βάση + όριο
- ◆ Διαμερίσεις για κάθε διεργασία
  - ➔ Σταθερού ή μεταβλητού μεγέθους;

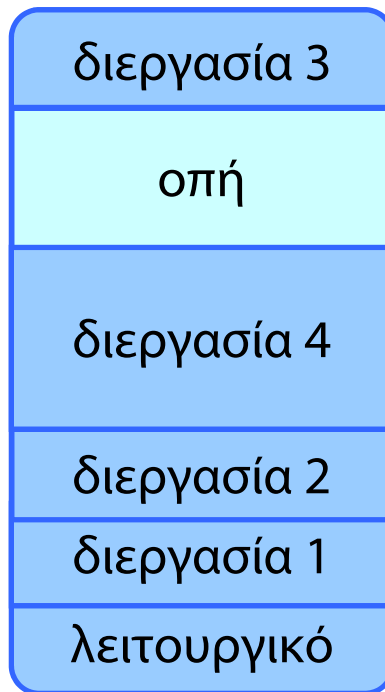
## Συνεχόμενη ανάθεση μνήμης (2)

- ◆ Στατικές διαμερίσεις
  - ➔ διαμέριση 1: αρχή = 256KB, μέγεθος = 256KB
  - ➔ διαμέριση 2: αρχή = 512KB, μέγεθος = 1024KB
  - ➔ διαμέριση 3: αρχή = 1536KB, μέγεθος = 2048KB
- ◆ Οι διεργασίες περιμένουν μέχρι κατάλληλη διαμέριση να γίνει διαθέσιμη

## Συνεχόμενη ανάθεση μνήμης (2)

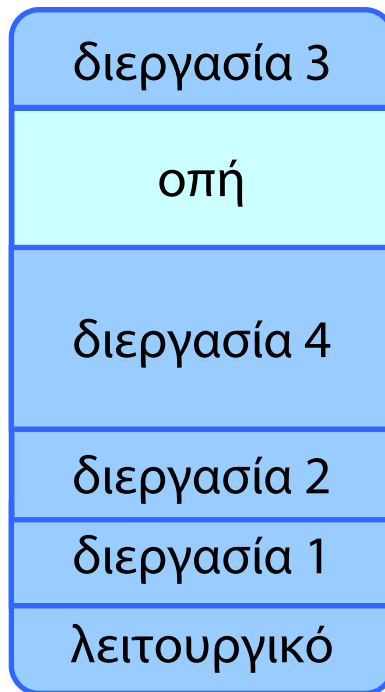
- ◆ Στατικές διαμερίσεις
  - ➔ διαμέριση 1: αρχή = 256KB, μέγεθος = 256KB
  - ➔ διαμέριση 2: αρχή = 512KB, μέγεθος = 1024KB
  - ➔ διαμέριση 3: αρχή = 1536KB, μέγεθος = 2048KB
- ◆ Οι διεργασίες περιμένουν μέχρι κατάλληλη διαμέριση να γίνει διαθέσιμη
- ◆ Άκαμπτο σχήμα ανάθεσης μνήμης
  - ➔ Πλέον δεν χρησιμοποιείται σχεδόν ποτέ

## Συνεχόμενη ανάθεση μνήμης (3)



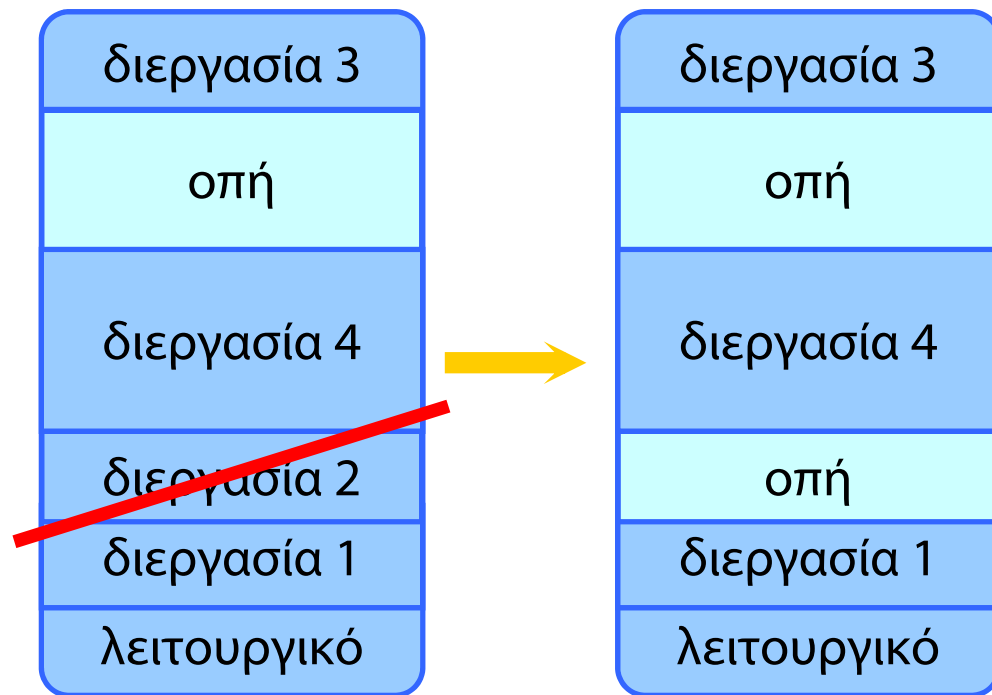
- ◆ Δυναμική ανάθεση μνήμης
  - ➔ διαμερίσεις μεταβλητού μεγέθους
  - ➔ το ΛΣ τηρεί στοιχεία για τις δεσμευμένες και ελεύθερες περιοχές της μνήμης
  - ➔ δεσμευμένες περιοχές και οπές ανάμεσά τους

## Συνεχόμενη ανάθεση μνήμης (3)



- ◆ Δυναμική ανάθεση μνήμης
  - ➔ διαμερίσεις μεταβλητού μεγέθους
  - ➔ το ΛΣ τηρεί στοιχεία για τις δεσμευμένες και ελεύθερες περιοχές της μνήμης
  - ➔ δεσμευμένες περιοχές και οπές ανάμεσά τους

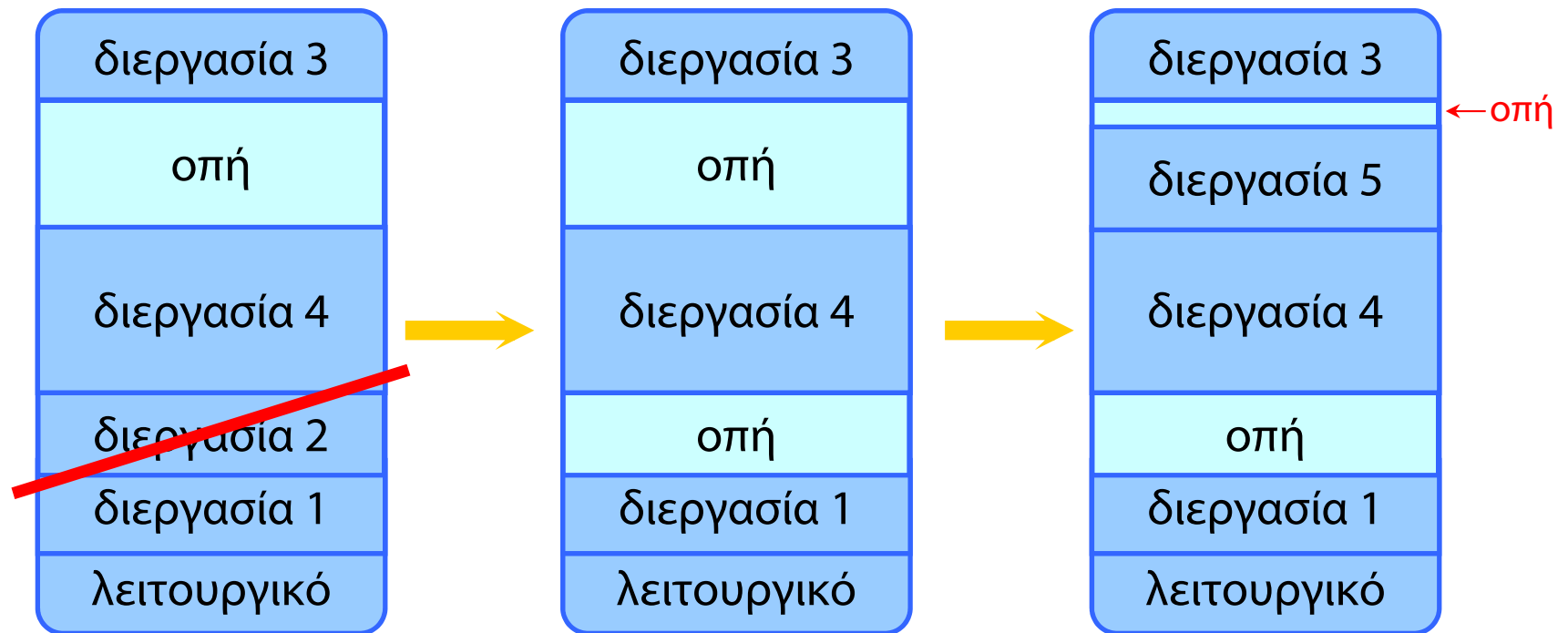
## Συνεχόμενη ανάθεση μνήμης (3)



- ◆ Δυναμική ανάθεση μνήμης
  - ➔ διαμερίσεις μεταβλητού μεγέθους
  - ➔ το ΛΣ τηρεί στοιχεία για τις δεσμευμένες και ελεύθερες περιοχές της μνήμης
  - ➔ δεσμευμένες περιοχές και οπές ανάμεσά τους

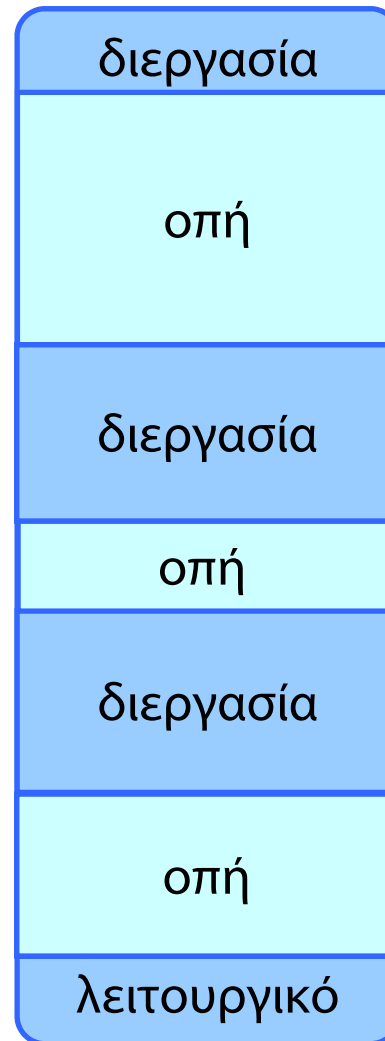


## Συνεχόμενη ανάθεση μνήμης (3)



- ◆ Δυναμική ανάθεση μνήμης
  - ➔ διαμερίσεις μεταβλητού μεγέθους
  - ➔ το ΛΣ τηρεί στοιχεία για τις δεσμευμένες και ελεύθερες περιοχές της μνήμης
  - ➔ δεσμευμένες περιοχές και οπές ανάμεσά τους

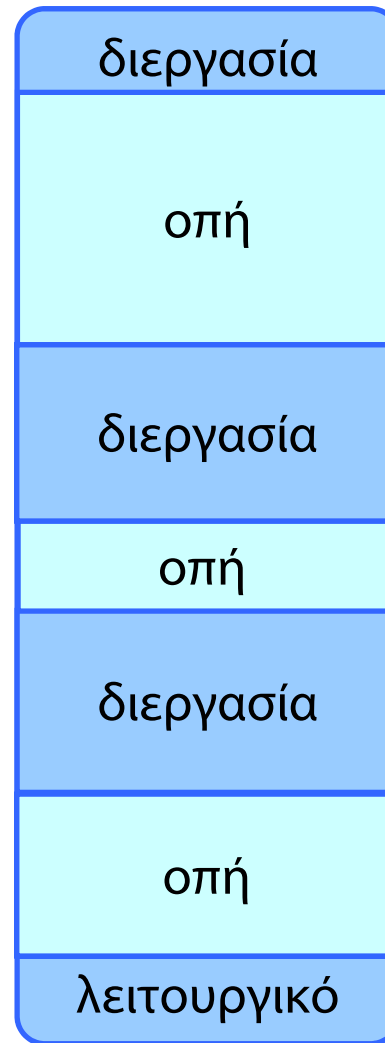
# Πρόβλημα δυναμικής εκχώρησης (1)



**Κύρια Μνήμη**

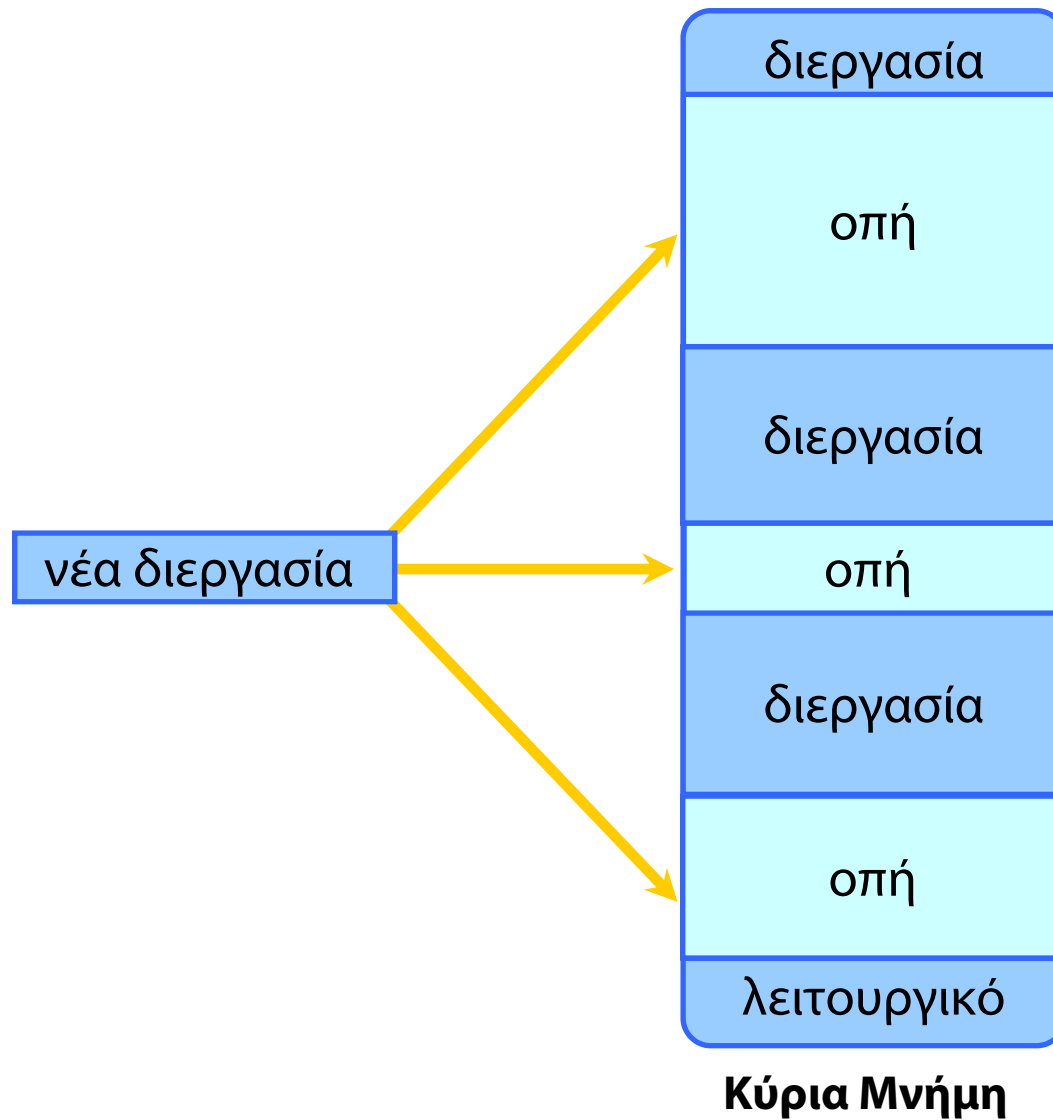
# Πρόβλημα δυναμικής εκχώρησης (1)

νέα διεργασία

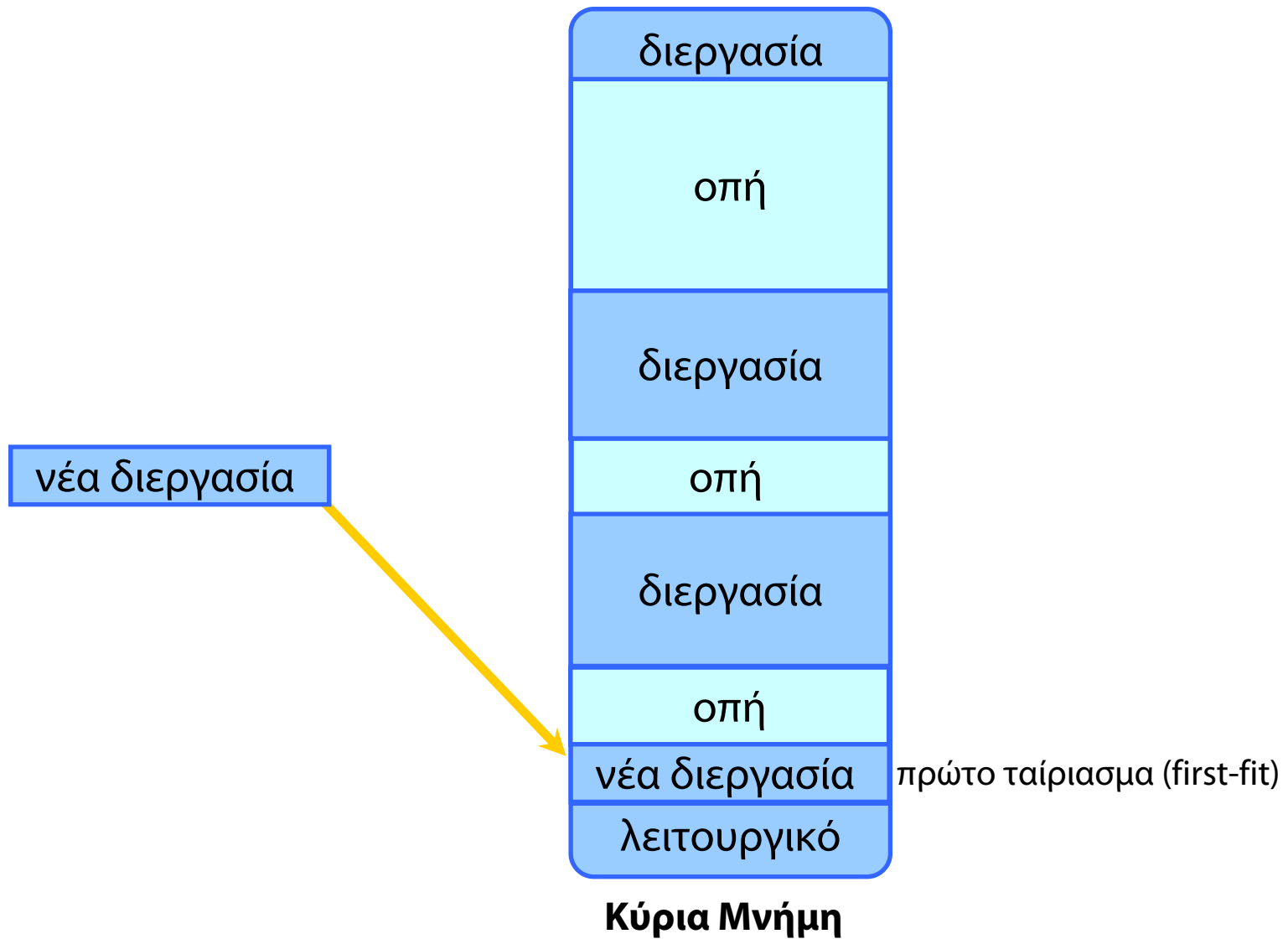


**Κύρια Μνήμη**

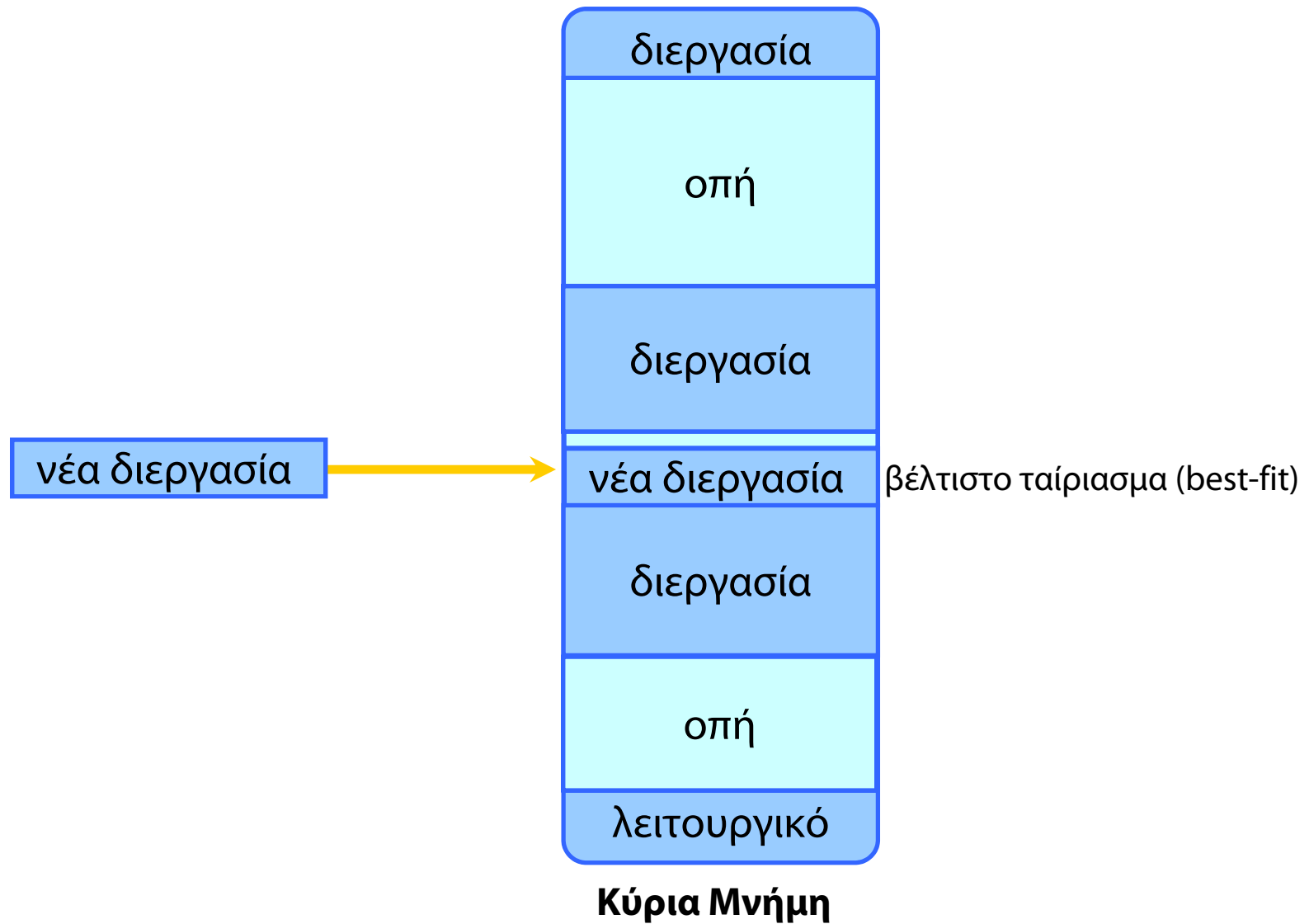
# Πρόβλημα δυναμικής εκχώρησης (1)



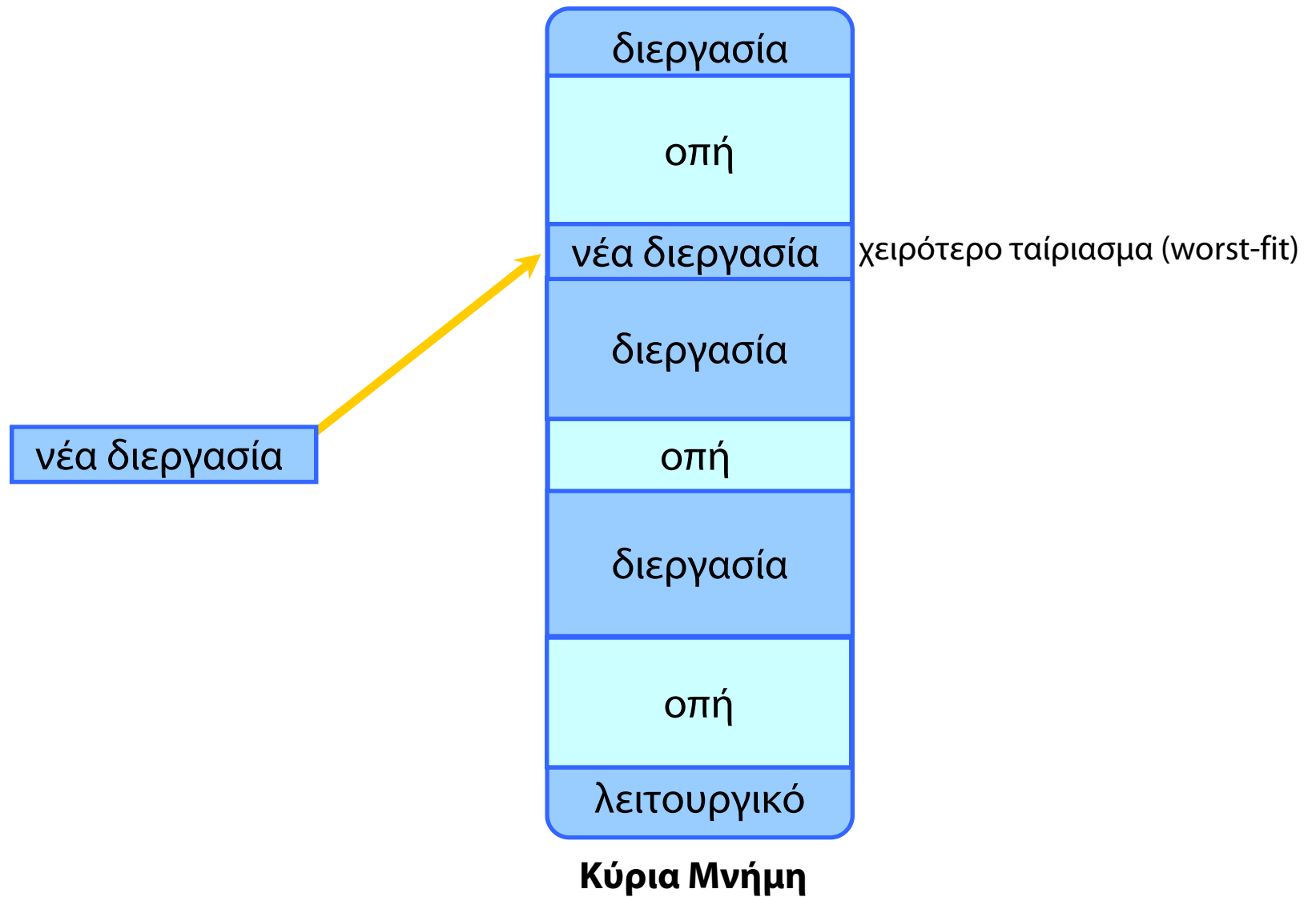
# Πρόβλημα δυναμικής εκχώρησης (2)



# Πρόβλημα δυναμικής εκχώρησης (1)



# Πρόβλημα δυναμικής εκχώρησης (1)



# Ποια στρατηγική είναι η καλύτερη;

## ◆ Πρώτο ταίριασμα

→ στο πρώτο που θα βρει, είτε από την αρχή είτε από εκεί που είχε σταματήσει

→ γρήγορο

## ◆ Βέλτιστο ταίριασμα

→ δεσμεύει τη μικρότερη οπή που είναι αρκετά μεγάλη

→ τις ελέγχει όλες, εκτός αν η λίστα είναι ταξινομημένη

## ◆ Χειρότερη τοποθέτηση

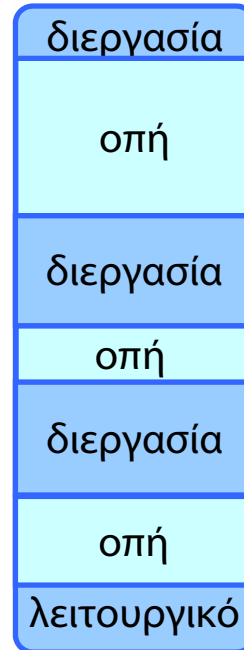
→ απομένουν οι μεγαλύτερες δυνατές οπές

## ◆ Οι προσομοιώσεις ευνοούν first-fit, best-fit



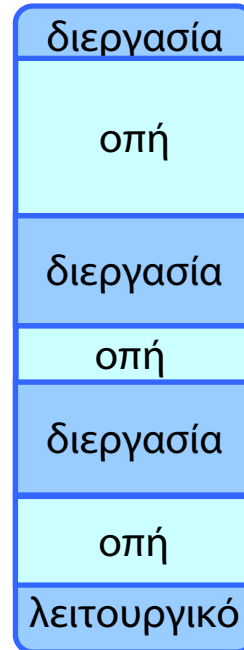
# Κατακερματισμός - fragmentation

νέα  
διεργασία



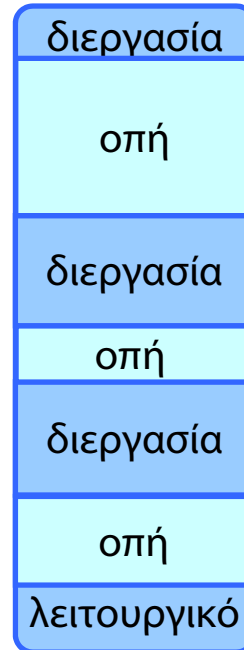
# Κατακερματισμός - fragmentation

νέα  
διεργασία



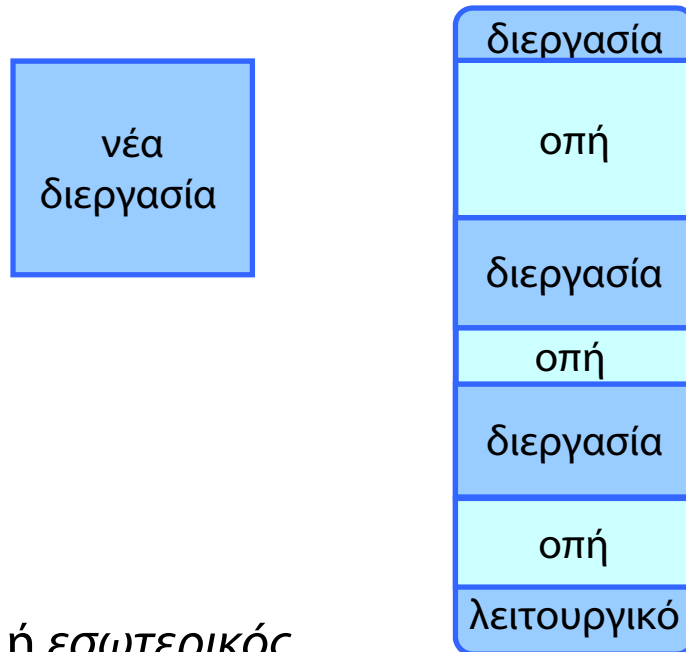
# Κατακερματισμός - fragmentation

νέα  
διεργασία



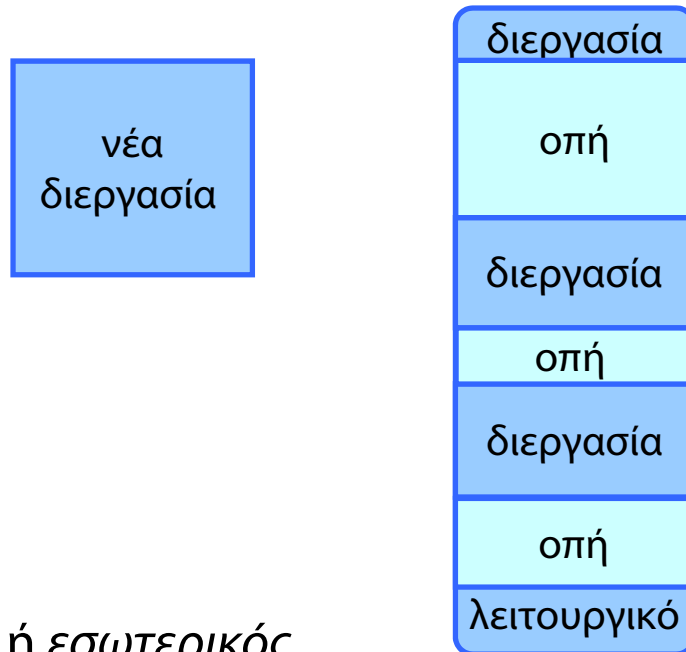
- ◆ *Εξωτερικός ή εσωτερικός*

# Κατακερματισμός - fragmentation



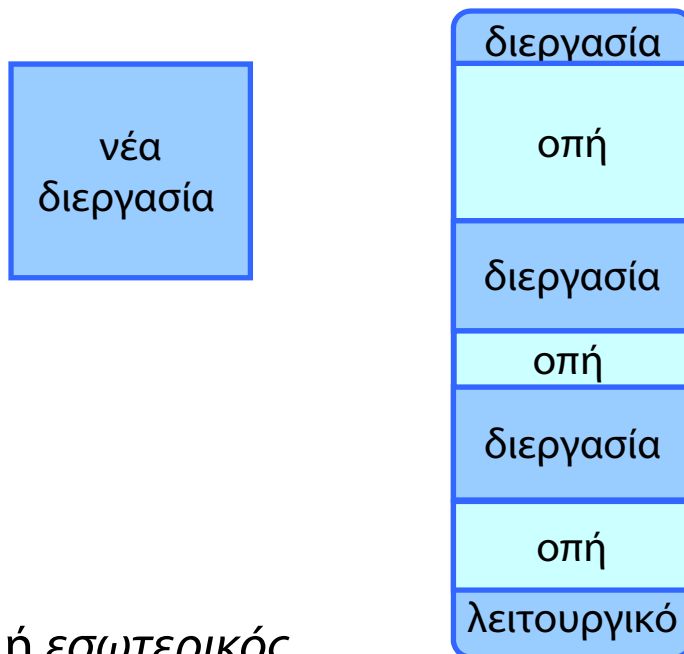
- ◆ *Εξωτερικός ή εσωτερικός*
- ◆ **Εξωτερικός:** ο συνολικός χώρος υπάρχει, αλλά η διεργασία δεν χωράει
  - ➔ πολλές μικρές οπές
  - ➔ κανόνας του 50%: με first-fit, για  $N$  τμήματα,  $0.5N$  χαμένη μνήμη

# Κατακερματισμός - fragmentation



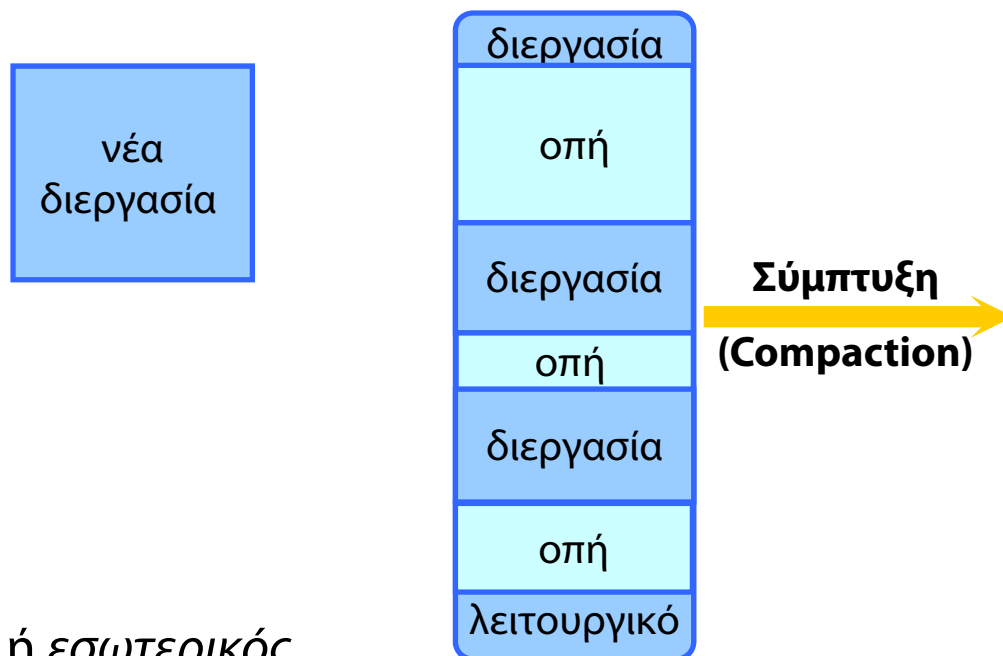
- ◆ *Εξωτερικός ή εσωτερικός*
- ◆ **Εξωτερικός:** ο συνολικός χώρος υπάρχει, αλλά η διεργασία δεν χωράει
  - ➔ πολλές μικρές οπές
  - ➔ κανόνας του 50%: με first-fit, για  $N$  τμήματα,  $0.5N$  χαμένη μνήμη

# Κατακερματισμός - fragmentation



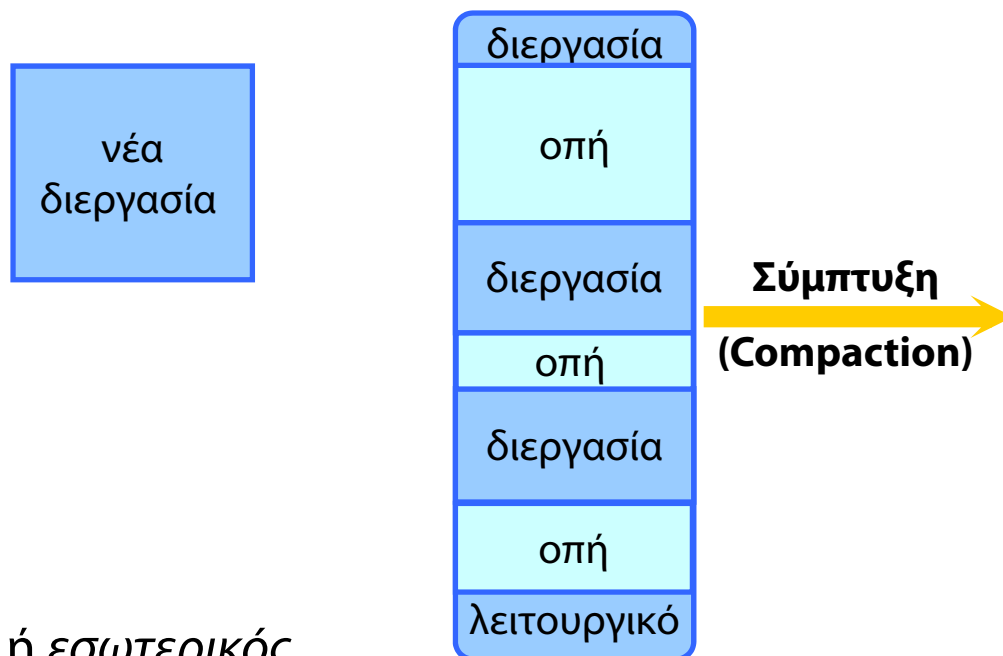
- ◆ *Εξωτερικός ή εσωτερικός*
- ◆ **Εξωτερικός:** ο συνολικός χώρος υπάρχει, αλλά η διεργασία δεν χωράει
  - ➔ πολλές μικρές οπές
  - ➔ κανόνας του 50%: με first-fit, για  $N$  τμήματα,  $0.5N$  χαμένη μνήμη
- ◆ **Εσωτερικός:** η διεργασία δεσμεύει περισσότερα απ' όσα χρειάζεται
  - ➔ η ανάθεση γίνεται σε τμήματα σταθερού μεγέθους
  - ➔ αχρησιμοποίητη μνήμη μέσα σε διαμερίσεις

# Κατακερματισμός - fragmentation



- ◆ *Εξωτερικός ή εσωτερικός*
- ◆ **Εξωτερικός:** ο συνολικός χώρος υπάρχει, αλλά η διεργασία δεν χωράει
  - ➔ πολλές μικρές οπές
  - ➔ κανόνας του 50%: με first-fit, για  $N$  τμήματα,  $0.5N$  χαμένη μνήμη
- ◆ **Εσωτερικός:** η διεργασία δεσμεύει περισσότερα απ' όσα χρειάζεται
  - ➔ η ανάθεση γίνεται σε τμήματα σταθερού μεγέθους
  - ➔ αχρησιμοποίητη μνήμη μέσα σε διαμερίσεις

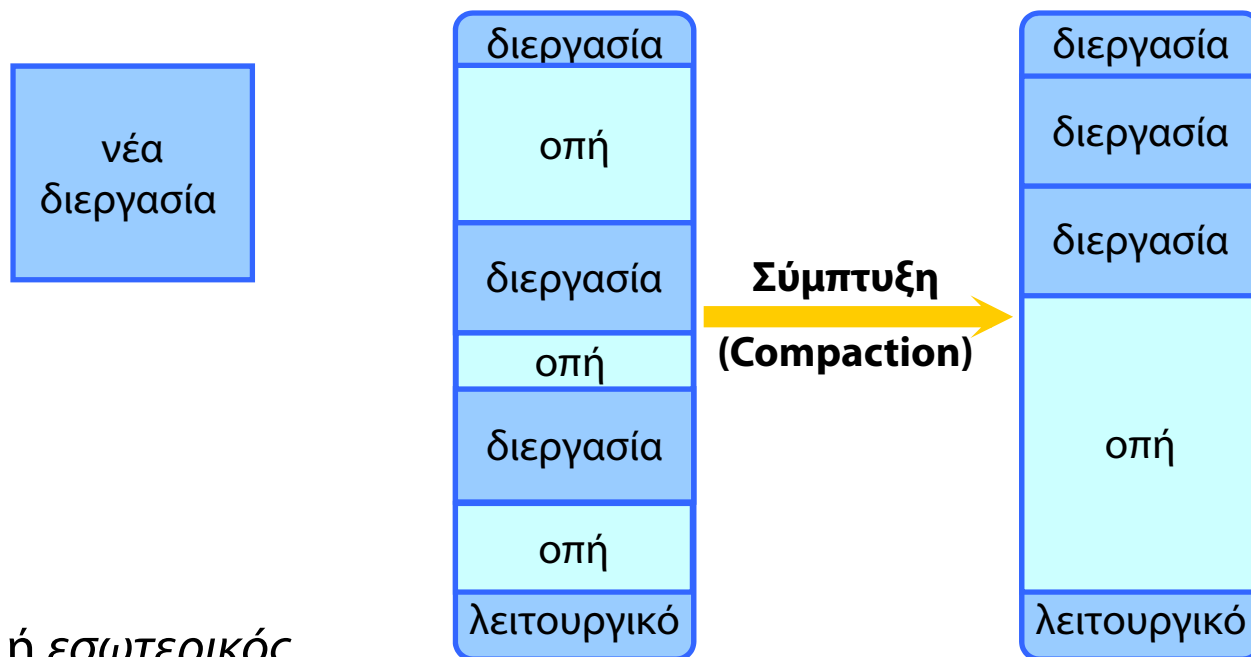
# Κατακερματισμός - fragmentation



- ◆ *Εξωτερικός ή εσωτερικός*
- ◆ **Εξωτερικός:** ο συνολικός χώρος υπάρχει, αλλά η διεργασία δεν χωράει
  - ➔ πολλές μικρές οπές
  - ➔ κανόνας του 50%: με first-fit, για  $N$  τμήματα,  $0.5N$  χαμένη μνήμη
- ◆ **Εσωτερικός:** η διεργασία δεσμεύει περισσότερα απ' όσα χρειάζεται
  - ➔ η ανάθεση γίνεται σε τμήματα σταθερού μεγέθους
  - ➔ αχρησιμοποίητη μνήμη μέσα σε διαμερίσεις

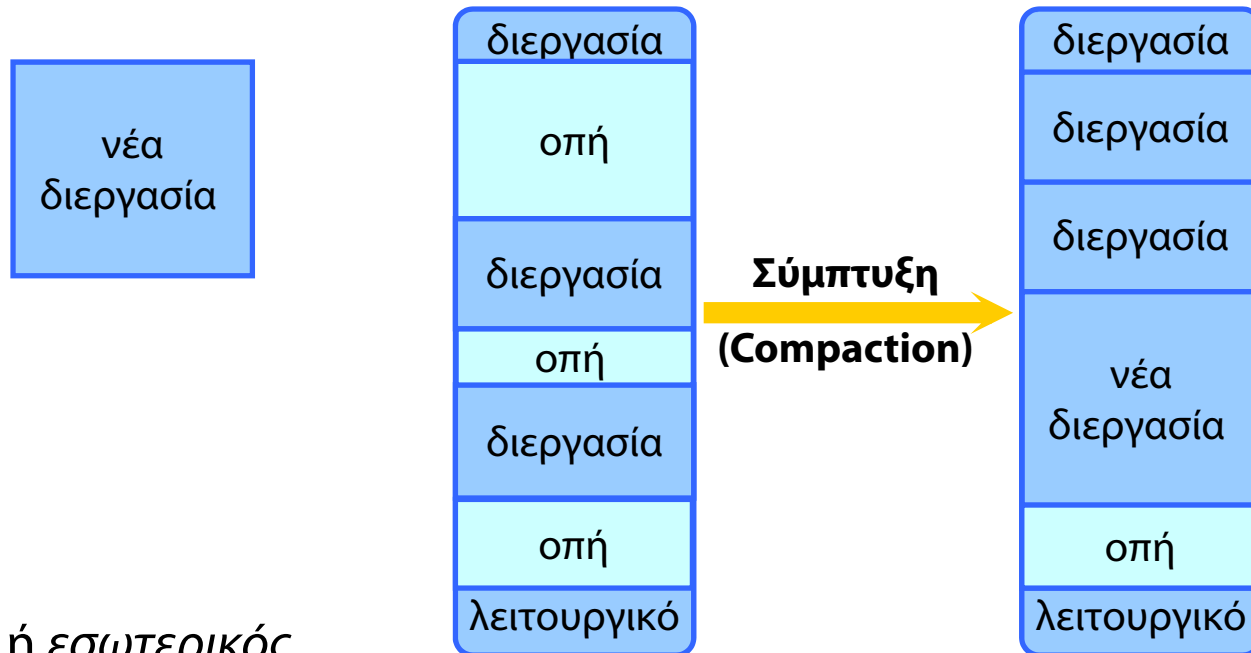


# Κατακερματισμός - fragmentation



- ◆ *Εξωτερικός ή εσωτερικός*
- ◆ **Εξωτερικός:** ο συνολικός χώρος υπάρχει, αλλά η διεργασία δεν χωράει
  - ➔ πολλές μικρές οπές
  - ➔ κανόνας του 50%: με first-fit, για  $N$  τμήματα,  $0.5N$  χαμένη μνήμη
- ◆ **Εσωτερικός:** η διεργασία δεσμεύει περισσότερα απ' όσα χρειάζεται
  - ➔ η ανάθεση γίνεται σε τμήματα σταθερού μεγέθους
  - ➔ αχρησιμοποίητη μνήμη μέσα σε διαμερίσεις

# Κατακερματισμός - fragmentation



- ◆ *Εξωτερικός ή εσωτερικός*
- ◆ **Εξωτερικός:** ο συνολικός χώρος υπάρχει, αλλά η διεργασία δεν χωράει
  - ➔ πολλές μικρές οπές
  - ➔ κανόνας του 50%: με first-fit, για  $N$  τμήματα,  $0.5N$  χαμένη μνήμη
- ◆ **Εσωτερικός:** η διεργασία δεσμεύει περισσότερα απ' όσα χρειάζεται
  - ➔ η ανάθεση γίνεται σε τμήματα σταθερού μεγέθους
  - ➔ αχρησιμοποίητη μνήμη μέσα σε διαμερίσεις

# Σύμπτυξη



- ◆ Μετακίνηση διεργασιών στη μνήμη
  - ➔ δημιουργούνται λιγότερες, μεγαλύτερες οπές
- ◆ Είναι πάντα εφικτή;

# Σύμπτυξη

- ◆ Μετακίνηση διεργασιών στη μνήμη
  - ➔ δημιουργούνται λιγότερες, μεγαλύτερες οπές
- ◆ Είναι πάντα εφικτή;
  - ➔ με καθορισμό διευθύνσεων κατά τη φόρτωση, όχι
- ◆ Απαιτείται υποστήριξη από το υλικό
  - ➔ καθορισμός διευθύνσεων στο χρόνο εκτέλεσης
  - ➔ βάση + όριο
- ◆ Κόστος;

# Σύμπτυξη

- ◆ Μετακίνηση διεργασιών στη μνήμη
  - ➔ δημιουργούνται λιγότερες, μεγαλύτερες οπές
- ◆ Είναι πάντα εφικτή;
  - ➔ με καθορισμό διευθύνσεων κατά τη φόρτωση, όχι
- ◆ Απαιτείται υποστήριξη από το υλικό
  - ➔ καθορισμός διευθύνσεων στο χρόνο εκτέλεσης
  - ➔ βάση + όριο
- ◆ Κόστος;
  - ➔ αντιγραφή περιοχών μνήμης
  - ➔ νέες τιμές σε καταχωρητές βάσης - ορίου

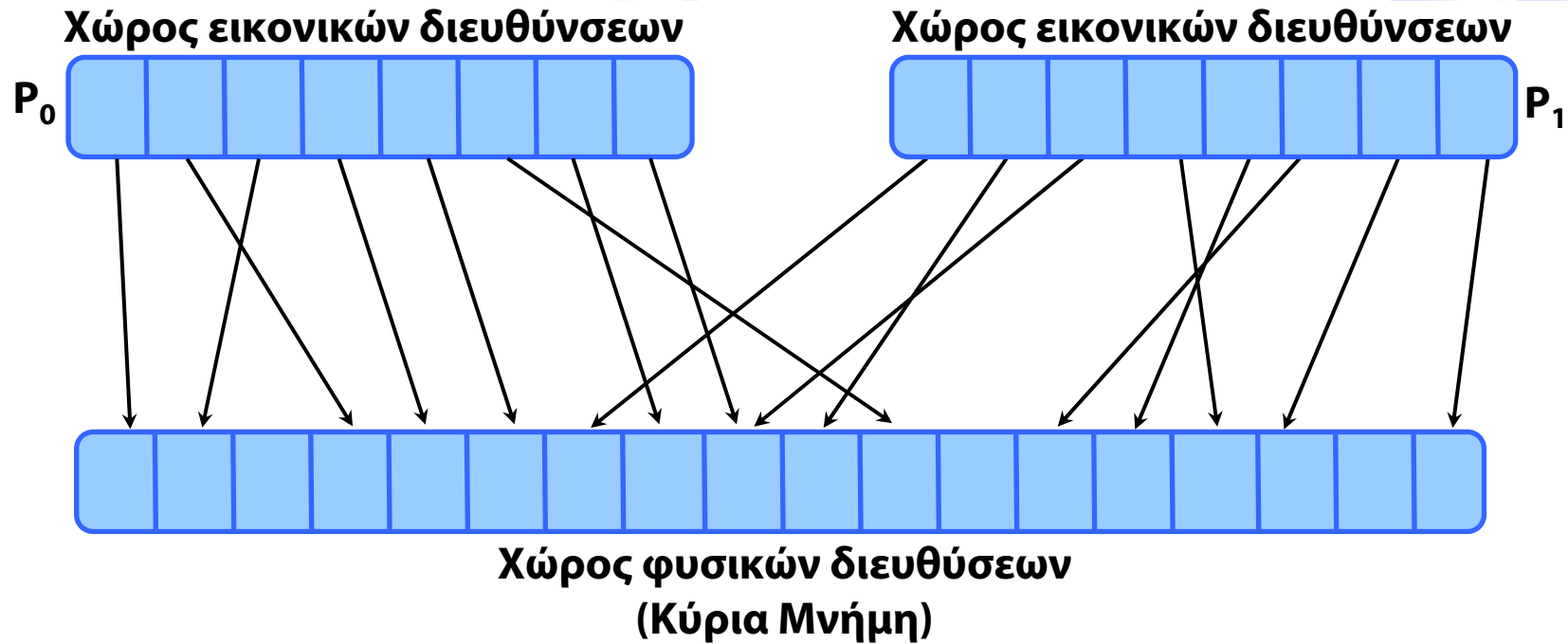
# Διαχείριση Κύριας Μνήμης - Σύνοψη

- ◆ Ιεραρχία μνήμης
- ◆ Μεταγλώττιση – φόρτωση – εκτέλεση κώδικα
- ◆ Καθορισμός διευθύνσεων
- ◆ Εναλλαγή διεργασιών
- ◆ Συνεχόμενη ανάθεση μνήμης
  - ➔ Στρατηγικές κατανομής, κατακερματισμός
- ◆ Σελιδοποίηση
  - ➔ Μετάφραση διευθύνσεων, πίνακες σελίδων, TLBs
  - ➔ Οργάνωση πινάκων σελίδων
- ◆ Κατάτμηση

# Σελιδοποίηση (1)

- ◆ Ο πιο συχνός τρόπος μετάφρασης διευθύνσεων
- ◆ Ο χώρος φυσικών διευθύνσεων χωρίζεται σε *πλαίσια* σταθερού μεγέθους (π.χ., 4096 bytes)
- ◆ Ο χώρος λογικών / εικονικών διευθύνσεων χωρίζεται σε *σελίδες*, ίδιου μεγέθους με τα πλαίσια
- ◆ Κάθε σελίδα αντιστοιχίζεται σε *οποιοδήποτε* πλαίσιο
  - ➔ χωρίς περιορισμό συνεχόμενης αποθήκευσης
  - ➔ η διεργασία ζει σε διάσπαρτα φυσικά τμήματα
  - ➔ αλλά σε γραμμικό χώρο εικονικών διευθύνσεων
- ◆ Το *Υλικό (MMU)* αναλαμβάνει τη μετάφραση

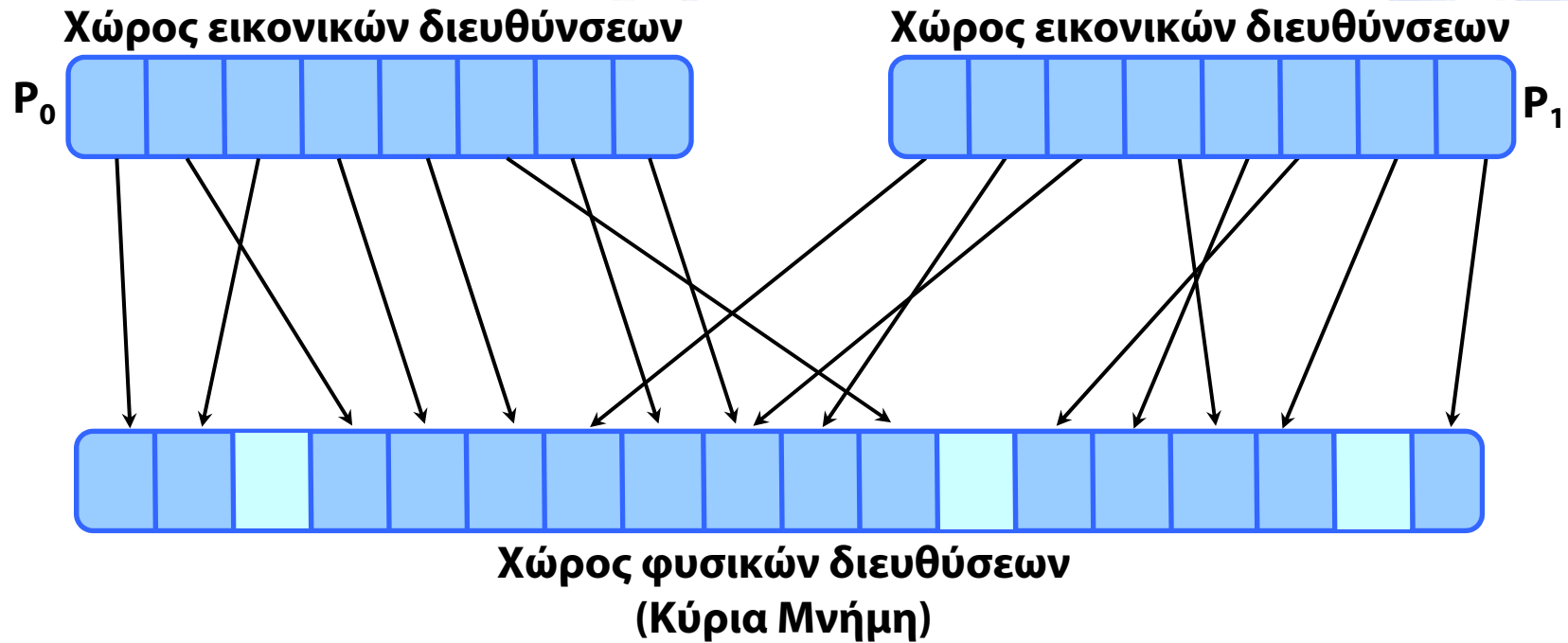
## Σελιδοποίηση (2)



- ◆ Διακριτοί, γραμμικοί χώροι εικονικών διευθύνσεων
- ◆ Προστασία μνήμης
  - ➔ Μια διεργασία δεν μπορεί καν να αναφερθεί σε ξένες διευθύνσεις

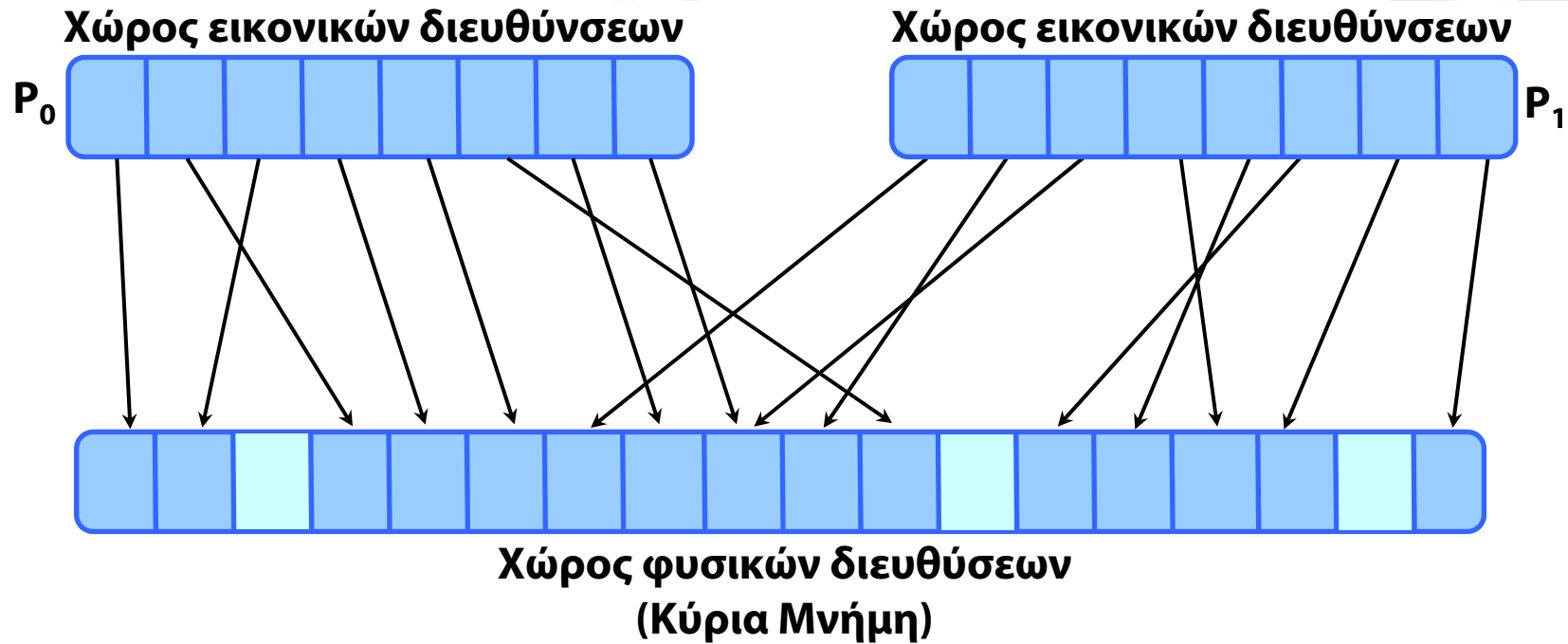


# Σελιδοποίηση (2)



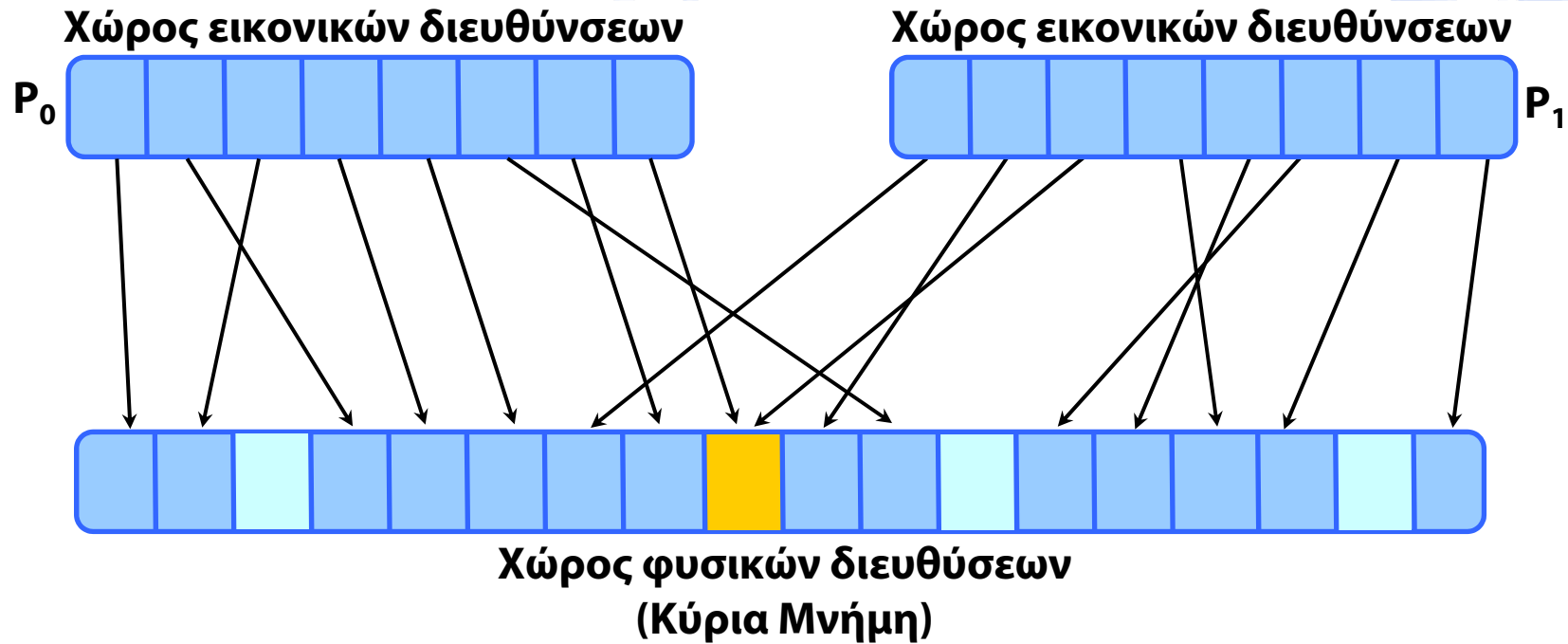
- ◆ Διακριτοί, γραμμικοί χώροι εικονικών διευθύνσεων
- ◆ Προστασία μνήμης
  - ➔ Μια διεργασία δεν μπορεί καν να αναφερθεί σε ξένες διευθύνσεις

## Σελιδοποίηση (2)



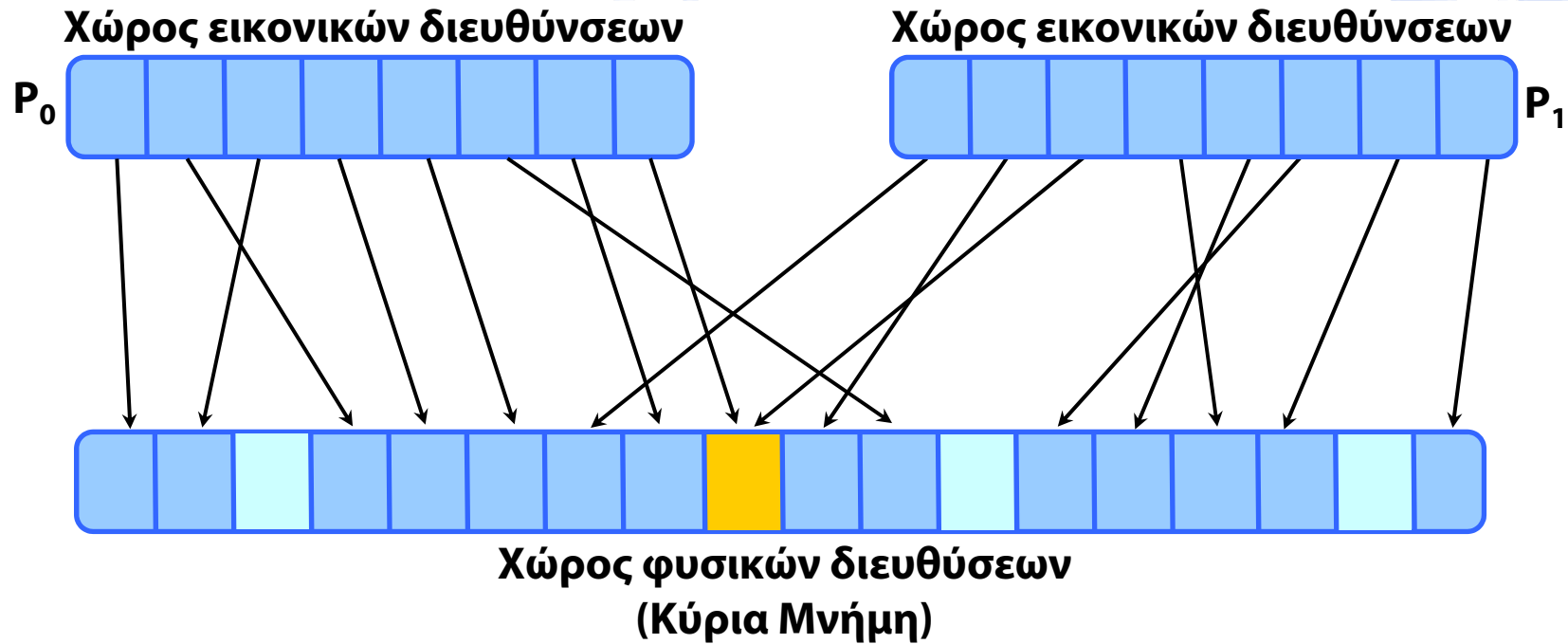
- ◆ Διακριτοί, γραμμικοί χώροι εικονικών διευθύνσεων
- ◆ Προστασία μνήμης
  - ➔ Μια διεργασία δεν μπορεί καν να αναφερθεί σε ξένες διευθύνσεις

## Σελιδοποίηση (2)



- ◆ Διακριτοί, γραμμικοί χώροι εικονικών διευθύνσεων
- ◆ Προστασία μνήμης
  - ➔ Μια διεργασία δεν μπορεί καν να αναφερθεί σε ξένες διευθύνσεις
- ◆ Μοιραζόμενη μνήμη, με αντιστοίχιση στο ίδιο πλαίσιο

## Σελιδοποίηση (2)



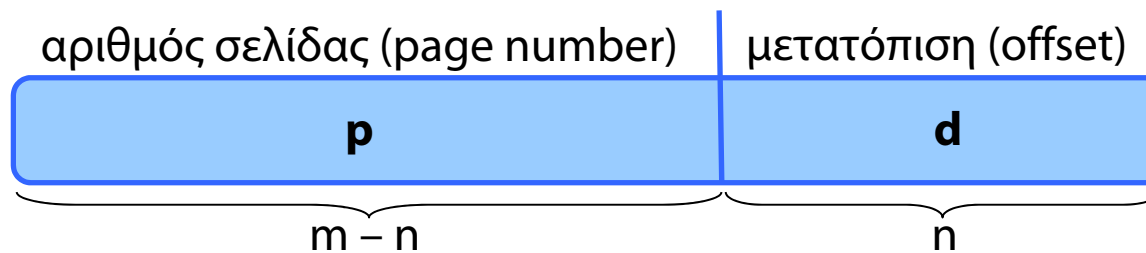
- ◆ Διακριτοί, γραμμικοί χώροι εικονικών διευθύνσεων
- ◆ Προστασία μνήμης
  - ➔ Μια διεργασία δεν μπορεί καν να αναφερθεί σε ξένες διευθύνσεις
- ◆ Μοιραζόμενη μνήμη, με αντιστοίχιση στο ίδιο πλαίσιο
- ◆ Δεν έχει εξωτερικό κατακερματισμό

## Σελιδοποίηση (3)

- ◆ Μετάφραση διευθύνσεων με *πίνακες σελίδων*
  - ➔ Τηρούνται από το ΛΣ, τους συμβουλεύεται το υλικό
- ◆ Μέγεθος σελίδας; 4 KB – 16MB
  - ➔ Μεγάλο μέγεθος → μικρό κόστος διαχείρισης
    - μικρότεροι πίνακες σελίδων
    - εντονότερος εσωτερικός κατακερματισμός
  - ➔ Μικρό μέγεθος
    - αποφεύγεται ο εσωτερικός κατακερματισμός
    - μεγαλύτερο κόστος διαχείρισης

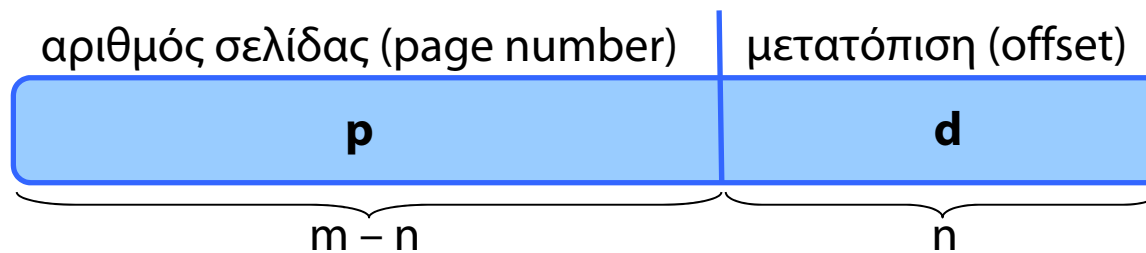
# Σελιδοποίηση - Μετάφραση διευθύνσεων

- ◆ Μέγεθος σελίδας  $2^n$ . **Γιατί** πρέπει να είναι δύναμη του 2;
- ◆ Διεύθυνση των  $m$  bits, χώρος εικονικών διευθύνσεων  $2^m$
- ◆ *Εικονική διεύθυνση*: αριθμός σελίδας + μετατόπιση



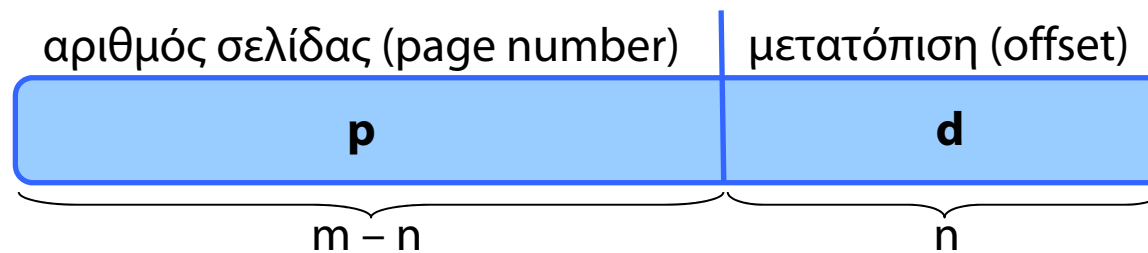
# Σελιδοποίηση - Μετάφραση διευθύνσεων

- ◆ Μέγεθος σελίδας  $2^n$ . **Γιατί** πρέπει να είναι δύναμη του 2;
- ◆ Διεύθυνση των  $m$  bits, χώρος εικονικών διευθύνσεων  $2^m$
- ◆ *Εικονική διεύθυνση*: αριθμός σελίδας + μετατόπιση



# Σελιδοποίηση - Μετάφραση διευθύνσεων

- ◆ Μέγεθος σελίδας  $2^n$ . **Γιατί** πρέπει να είναι δύναμη του 2;
- ◆ Διεύθυνση των  $m$  bits, χώρος εικονικών διευθύνσεων  $2^m$
- ◆ *Εικονική διεύθυνση*: αριθμός σελίδας + μετατόπιση

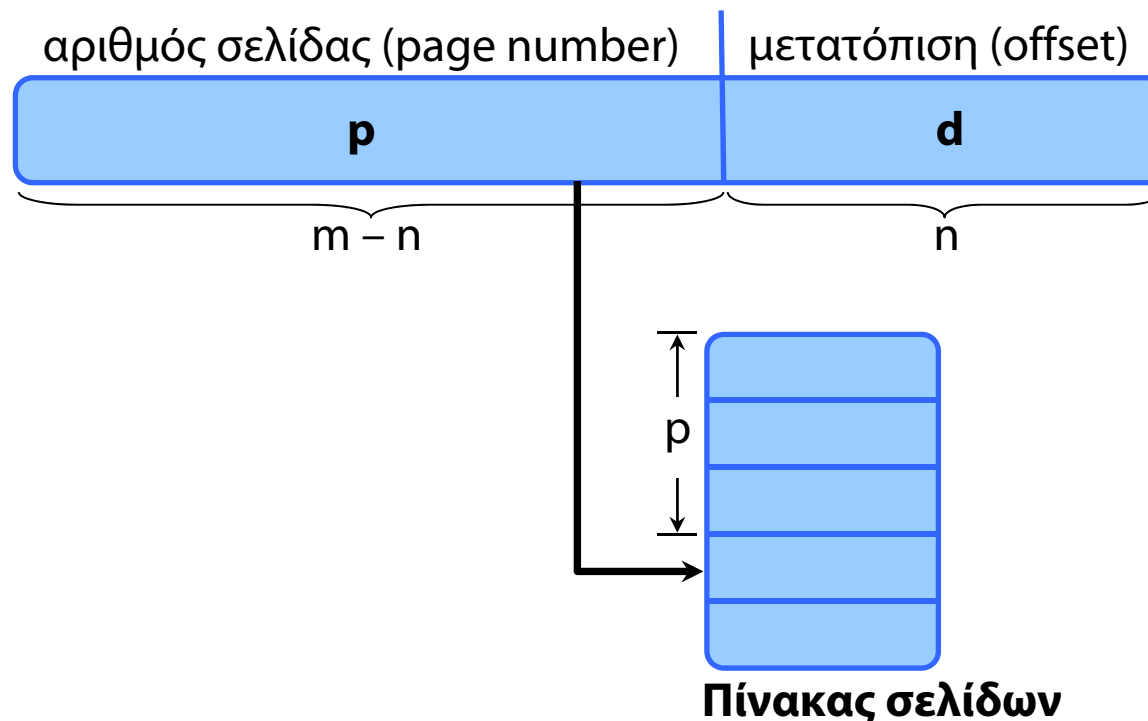


**Πίνακας σελίδων**



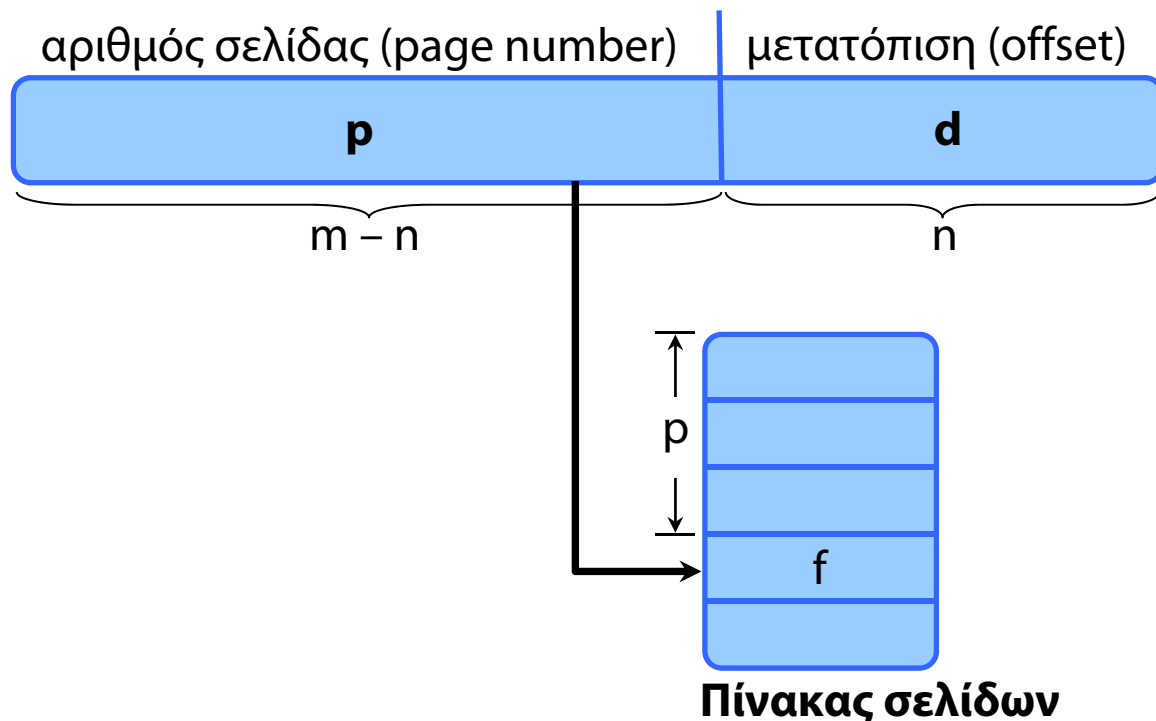
# Σελιδοποίηση - Μετάφραση διευθύνσεων

- ◆ Μέγεθος σελίδας  $2^n$ . **Γιατί** πρέπει να είναι δύναμη του 2;
- ◆ Διεύθυνση των  $m$  bits, χώρος εικονικών διευθύνσεων  $2^m$
- ◆ *Εικονική διεύθυνση*: αριθμός σελίδας + μετατόπιση

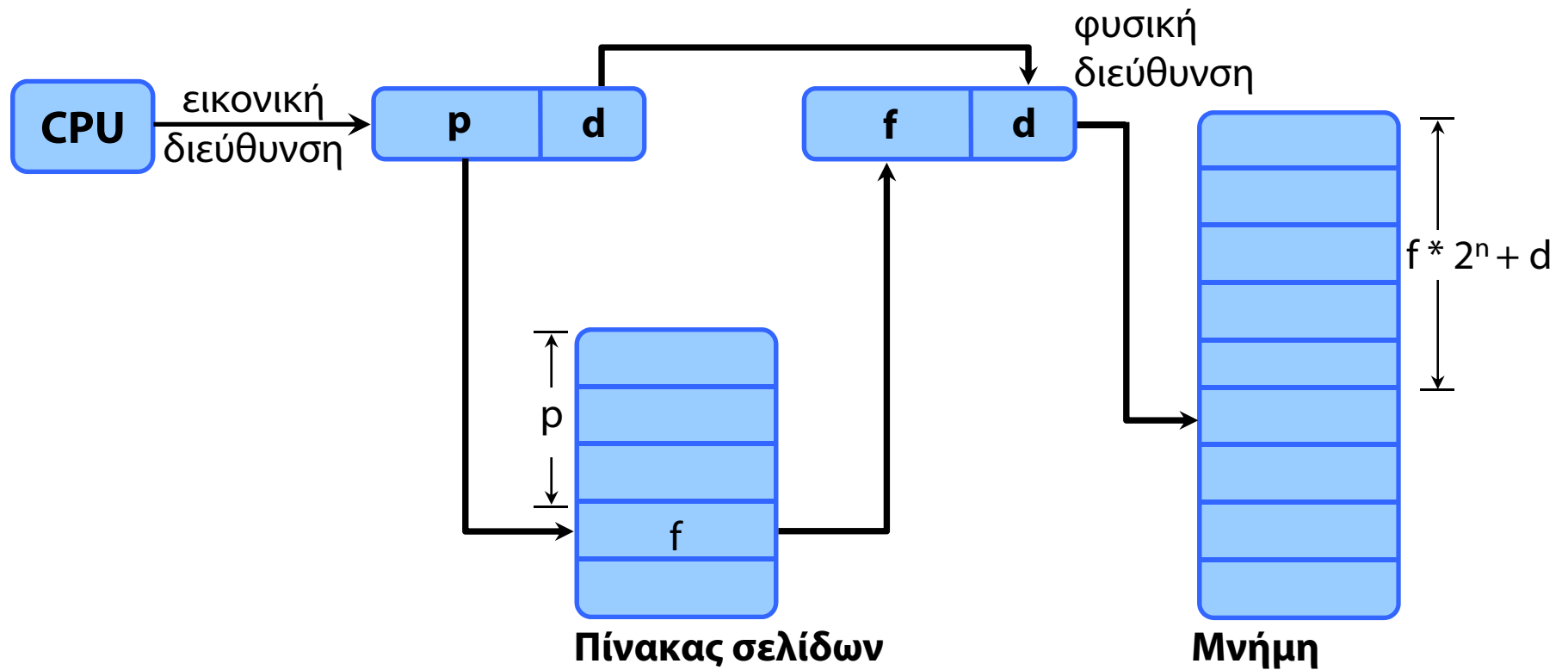


# Σελιδοποίηση - Μετάφραση διευθύνσεων

- ◆ Μέγεθος σελίδας  $2^n$ . **Γιατί** πρέπει να είναι δύναμη του 2;
- ◆ Διεύθυνση των  $m$  bits, χώρος εικονικών διευθύνσεων  $2^m$
- ◆ *Εικονική διεύθυνση*: αριθμός σελίδας + μετατόπιση



# Υποστήριξη υλικού για σελιδοποίηση



# Μοντέλο σελιδοποίησης

|          |
|----------|
| σελίδα 0 |
| σελίδα 1 |
| σελίδα 2 |
| σελίδα 3 |
| σελίδα 4 |

**Εικονική  
Μνήμη**

|   |
|---|
| 2 |
| 5 |
| 1 |
| 7 |
| 3 |

**Πίνακας  
σελίδων**

|   |  |
|---|--|
| 0 |  |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |

**Φυσική  
Μνήμη**

# Μοντέλο σελιδοποίησης

|          |
|----------|
| σελίδα 0 |
| σελίδα 1 |
| σελίδα 2 |
| σελίδα 3 |
| σελίδα 4 |

**Εικονική  
Μνήμη**

|   |
|---|
| 2 |
| 5 |
| 1 |
| 7 |
| 3 |

**Πίνακας  
σελίδων**

|   |  |
|---|--|
| 0 |  |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |

**Φυσική  
Μνήμη**

# Μοντέλο σελιδοποίησης

|          |
|----------|
| σελίδα 0 |
| σελίδα 1 |
| σελίδα 2 |
| σελίδα 3 |
| σελίδα 4 |

**Εικονική  
Μνήμη**

|   |
|---|
| 2 |
| 5 |
| 1 |
| 7 |
| 3 |

**Πίνακας  
σελίδων**

|   |          |
|---|----------|
| 0 |          |
| 1 |          |
| 2 | σελίδα 0 |
| 3 |          |
| 4 |          |
| 5 |          |
| 6 |          |
| 7 |          |
| 8 |          |

**Φυσική  
Μνήμη**

# Μοντέλο σελιδοποίησης

|          |
|----------|
| σελίδα 0 |
| σελίδα 1 |
| σελίδα 2 |
| σελίδα 3 |
| σελίδα 4 |

**Εικονική  
Μνήμη**

|   |
|---|
| 2 |
| 5 |
| 1 |
| 7 |
| 3 |

**Πίνακας  
σελίδων**

|   |          |
|---|----------|
| 0 |          |
| 1 |          |
| 2 | σελίδα 0 |
| 3 |          |
| 4 |          |
| 5 | σελίδα 1 |
| 6 |          |
| 7 |          |
| 8 |          |

**Φυσική  
Μνήμη**

# Μοντέλο σελιδοποίησης

|          |
|----------|
| σελίδα 0 |
| σελίδα 1 |
| σελίδα 2 |
| σελίδα 3 |
| σελίδα 4 |

**Εικονική  
Μνήμη**

|   |
|---|
| 2 |
| 5 |
| 1 |
| 7 |
| 3 |

**Πίνακας  
σελίδων**

|   |          |
|---|----------|
| 0 |          |
| 1 | σελίδα 2 |
| 2 | σελίδα 0 |
| 3 |          |
| 4 |          |
| 5 | σελίδα 1 |
| 6 |          |
| 7 |          |
| 8 |          |

**Φυσική  
Μνήμη**



# Μοντέλο σελιδοποίησης

|          |
|----------|
| σελίδα 0 |
| σελίδα 1 |
| σελίδα 2 |
| σελίδα 3 |
| σελίδα 4 |

**Εικονική  
Μνήμη**

|   |
|---|
| 2 |
| 5 |
| 1 |
| 7 |
| 3 |

**Πίνακας  
σελίδων**

|   |          |
|---|----------|
| 0 |          |
| 1 | σελίδα 2 |
| 2 | σελίδα 0 |
| 3 |          |
| 4 |          |
| 5 | σελίδα 1 |
| 6 |          |
| 7 | σελίδα 3 |
| 8 |          |

**Φυσική  
Μνήμη**

# Μοντέλο σελιδοποίησης

|          |
|----------|
| σελίδα 0 |
| σελίδα 1 |
| σελίδα 2 |
| σελίδα 3 |
| σελίδα 4 |

**Εικονική  
Μνήμη**

|   |
|---|
| 2 |
| 5 |
| 1 |
| 7 |
| 3 |

**Πίνακας  
σελίδων**

|   |          |
|---|----------|
| 0 |          |
| 1 | σελίδα 2 |
| 2 | σελίδα 0 |
| 3 | σελίδα 4 |
| 4 |          |
| 5 | σελίδα 1 |
| 6 |          |
| 7 | σελίδα 3 |
| 8 |          |

**Φυσική  
Μνήμη**

# Μοντέλο σελιδοποίησης

|          |
|----------|
| σελίδα 0 |
| σελίδα 1 |
| σελίδα 2 |
| σελίδα 3 |
| σελίδα 4 |

**Εικονική  
Μνήμη**

|   |
|---|
| 2 |
| 5 |
| 1 |
| 7 |
| 3 |

**Πίνακας  
σελίδων**

|   |          |
|---|----------|
| 0 |          |
| 1 | σελίδα 2 |
| 2 | σελίδα 0 |
| 3 | σελίδα 4 |
| 4 |          |
| 5 | σελίδα 1 |
| 6 |          |
| 7 | σελίδα 3 |
| 8 |          |

**Φυσική  
Μνήμη**

- ◆ Ποιος δίνει μνήμη στις διεργασίες;

# Μοντέλο σελιδοποίησης

|          |
|----------|
| σελίδα 0 |
| σελίδα 1 |
| σελίδα 2 |
| σελίδα 3 |
| σελίδα 4 |

**Εικονική  
Μνήμη**

|   |
|---|
| 2 |
| 5 |
| 1 |
| 7 |
| 3 |

**Πίνακας  
σελίδων**

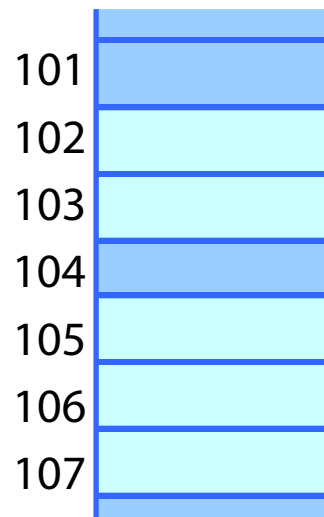
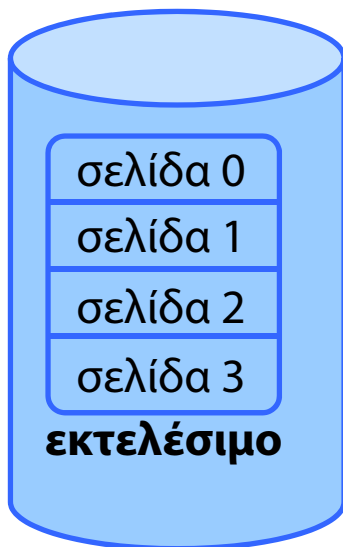
|   |          |
|---|----------|
| 0 |          |
| 1 | σελίδα 2 |
| 2 | σελίδα 0 |
| 3 | σελίδα 4 |
| 4 |          |
| 5 | σελίδα 1 |
| 6 |          |
| 7 | σελίδα 3 |
| 8 |          |

**Φυσική  
Μνήμη**

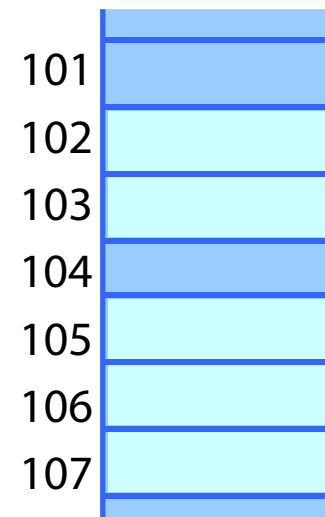
- ◆ Ποιος δίνει μνήμη στις διεργασίες;
  - ➔ Το ΛΣ βρίσκει διαθέσιμα πλαίσια και τα απεικονίζει
- ◆ Το ΛΣ γνωρίζει (π.χ. λίστα) ποια πλαίσια είναι ελεύθερα (πίνακας πλαισίων)

# Ανάθεση πλαισίων σε νέα διεργασία

Λίστα ελεύθερων πλαισίων:  
102, 105, 107, 103, 106



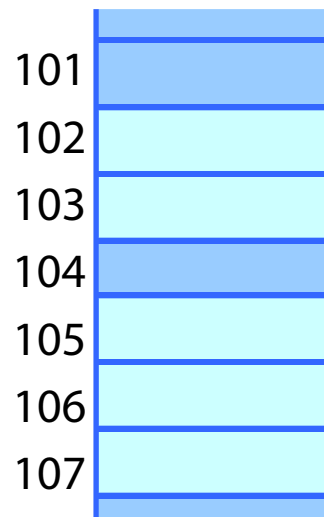
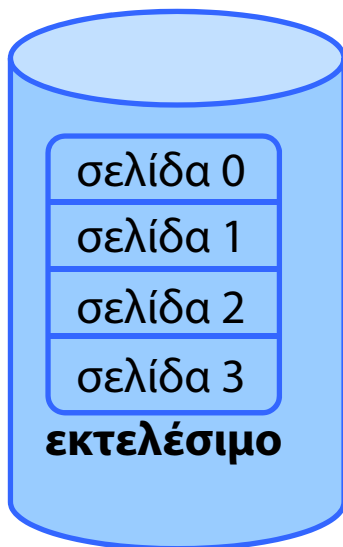
**Φυσική Μνήμη  
(πριν)**



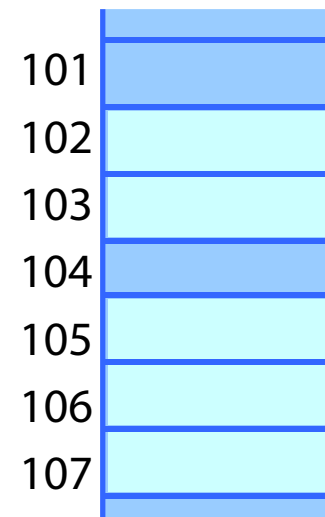
**Φυσική Μνήμη  
(μετά)**

# Ανάθεση πλαισίων σε νέα διεργασία

Λίστα ελεύθερων πλαισίων:  
102, 105, 107, 103, 106



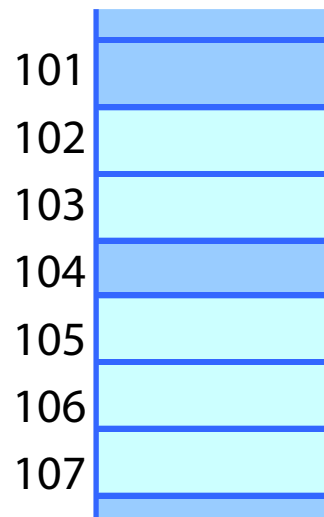
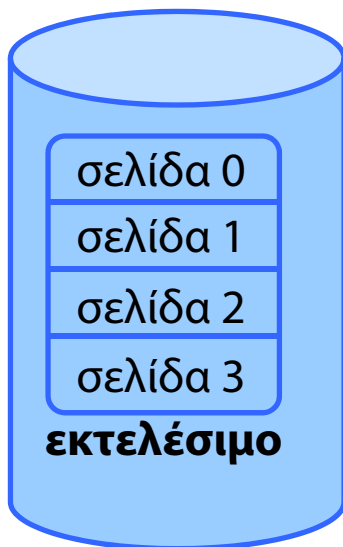
**Φυσική Μνήμη  
(πριν)**



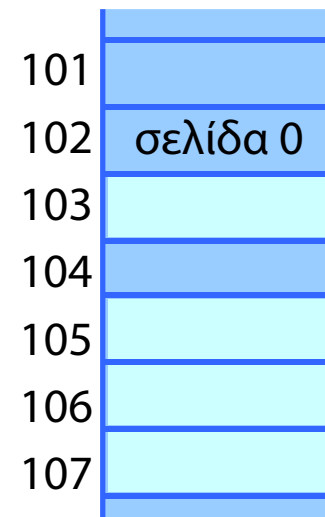
**Φυσική Μνήμη  
(μετά)**

# Ανάθεση πλαισίων σε νέα διεργασία

Λίστα ελεύθερων πλαισίων:  
102, 105, 107, 103, 106



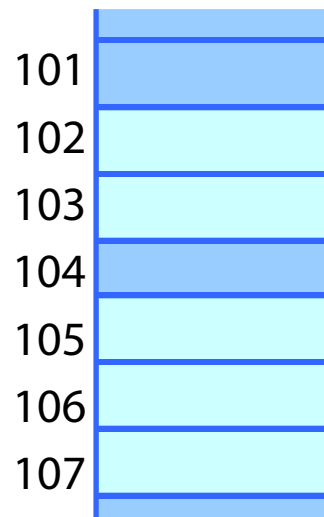
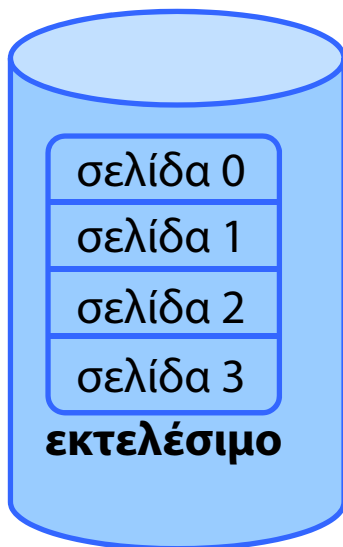
**Φυσική Μνήμη  
(πριν)**



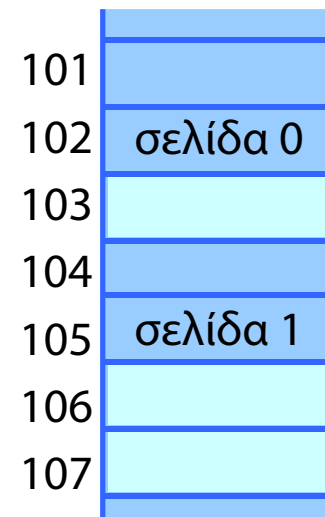
**Φυσική Μνήμη  
(μετά)**

# Ανάθεση πλαισίων σε νέα διεργασία

Λίστα ελεύθερων πλαισίων:  
102, 105, 107, 103, 106



**Φυσική Μνήμη  
(πριν)**

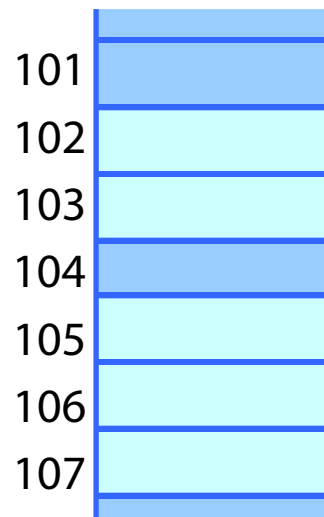
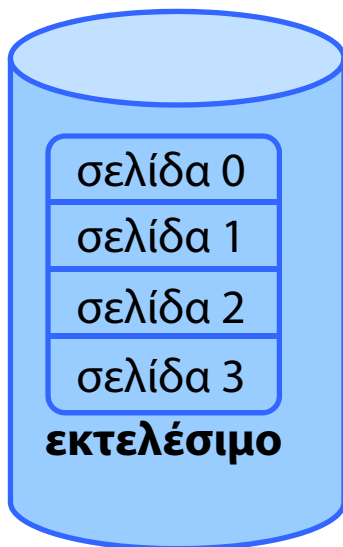


**Φυσική Μνήμη  
(μετά)**

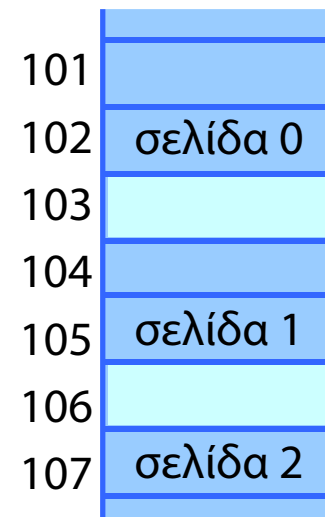


# Ανάθεση πλαισίων σε νέα διεργασία

Λίστα ελεύθερων πλαισίων:  
102, 105, 107, 103, 106



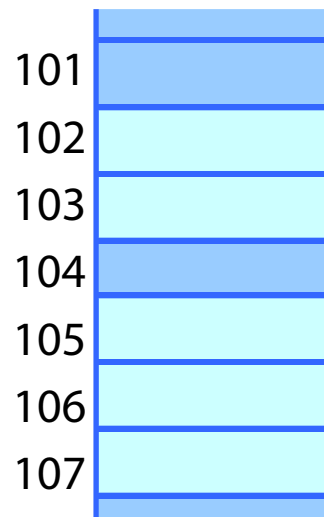
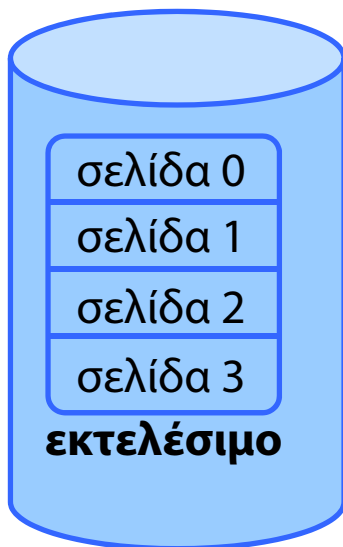
**Φυσική Μνήμη  
(πριν)**



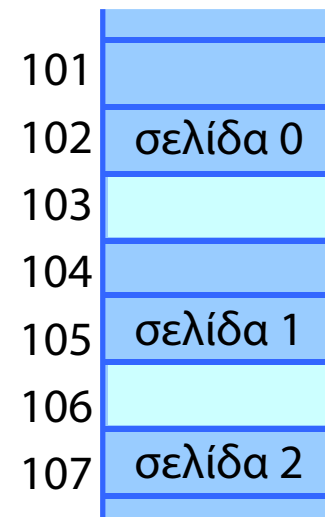
**Φυσική Μνήμη  
(μετά)**

# Ανάθεση πλαισίων σε νέα διεργασία

Λίστα ελεύθερων πλαισίων:  
102, 105, 107, 103, 106



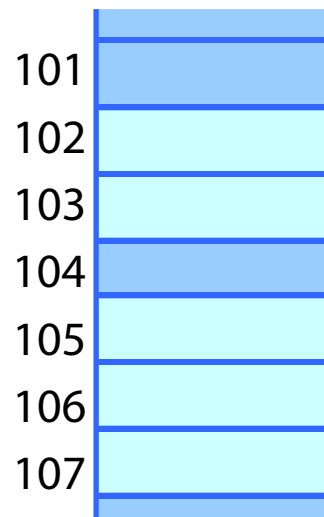
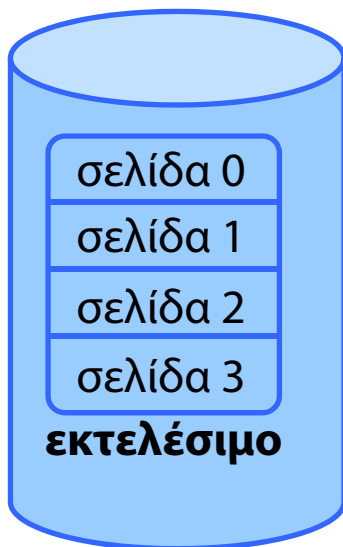
**Φυσική Μνήμη  
(πριν)**



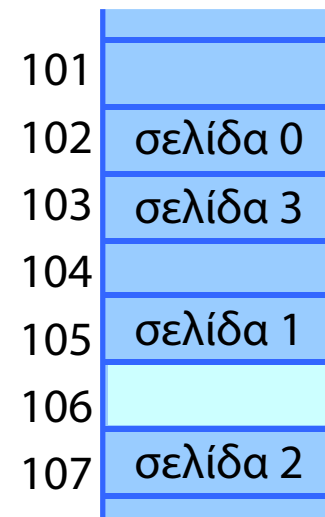
**Φυσική Μνήμη  
(μετά)**

# Ανάθεση πλαισίων σε νέα διεργασία

Λίστα ελεύθερων πλαισίων:  
102, 105, 107, 103, 106



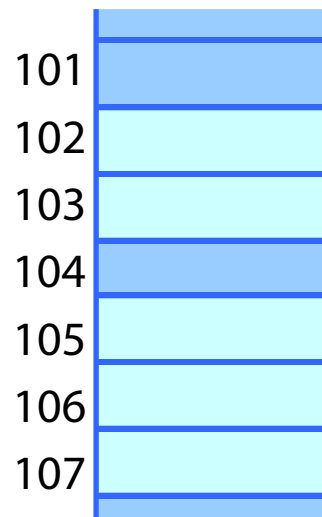
**Φυσική Μνήμη  
(πριν)**



**Φυσική Μνήμη  
(μετά)**

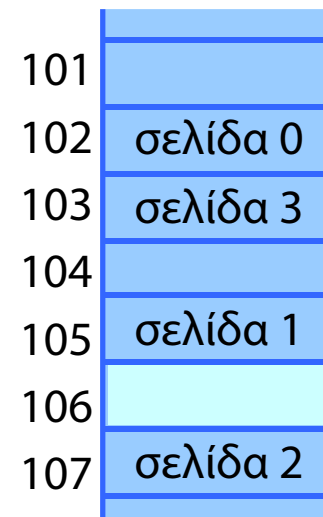
# Ανάθεση πλαισίων σε νέα διεργασία

Λίστα ελεύθερων πλαισίων:  
102, 105, 107, 103, 106

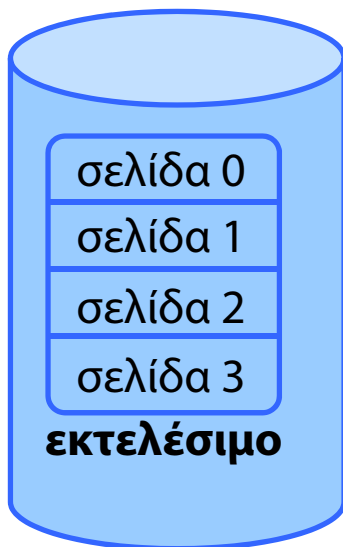


**Φυσική Μνήμη  
(πριν)**

Λίστα ελεύθερων πλαισίων:  
106



**Φυσική Μνήμη  
(μετά)**



# Υποστήριξη υλικού για σελιδοποίηση

## ◆ Πώς υλοποιείται ο πίνακας σελίδων;

➔ Ειδικοί, αφιερωμένοι καταχωρητές (DEC PDP-11)

- ναι, αλλά είχε 16-bit διευθύνσεις και μέγεθος σελίδας 8KB

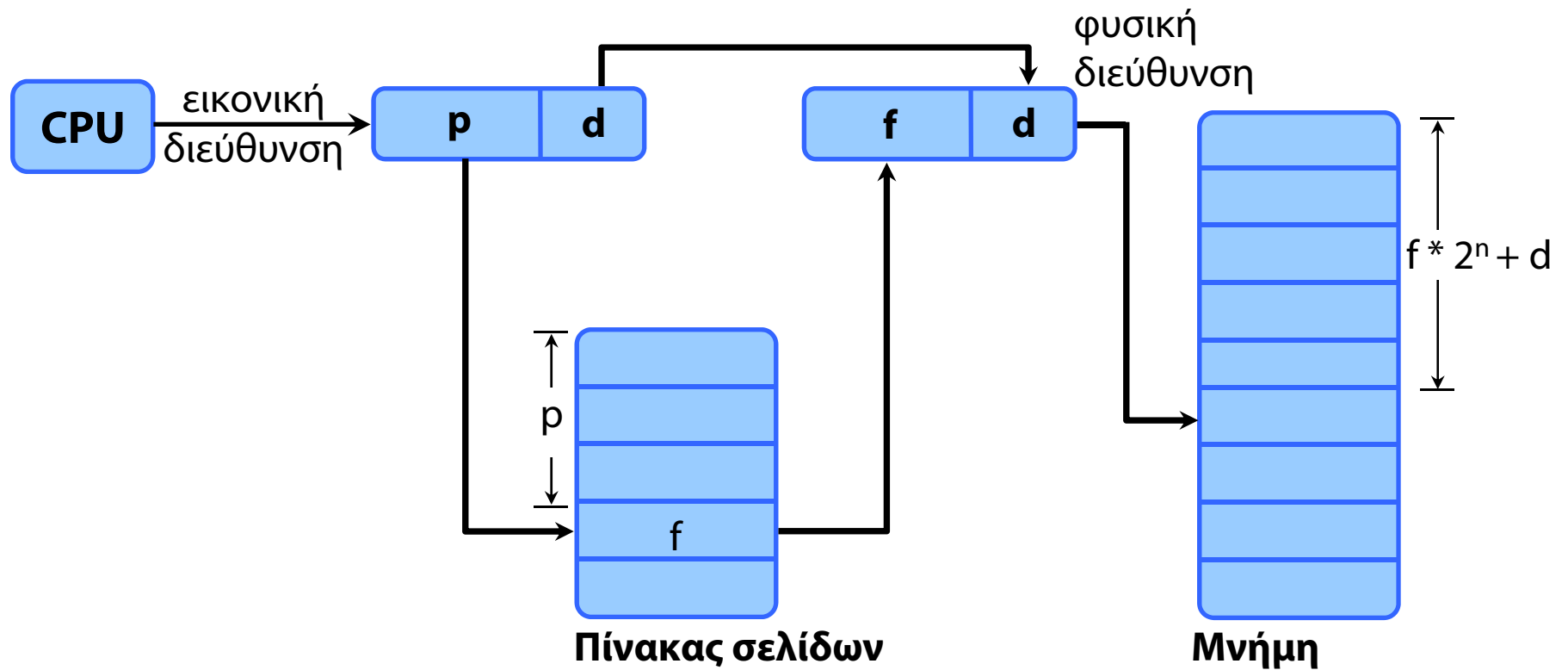
➔ Πίνακας σελίδων στη μνήμη

- Καταχωρητής βάσης πίνακα σελίδων – PTBR  
(πχ. %cr3 στον x86)
- Πόσες προσβάσεις χρειάζονται στη μνήμη για *κάθε* πρόσβαση από τη CPU;

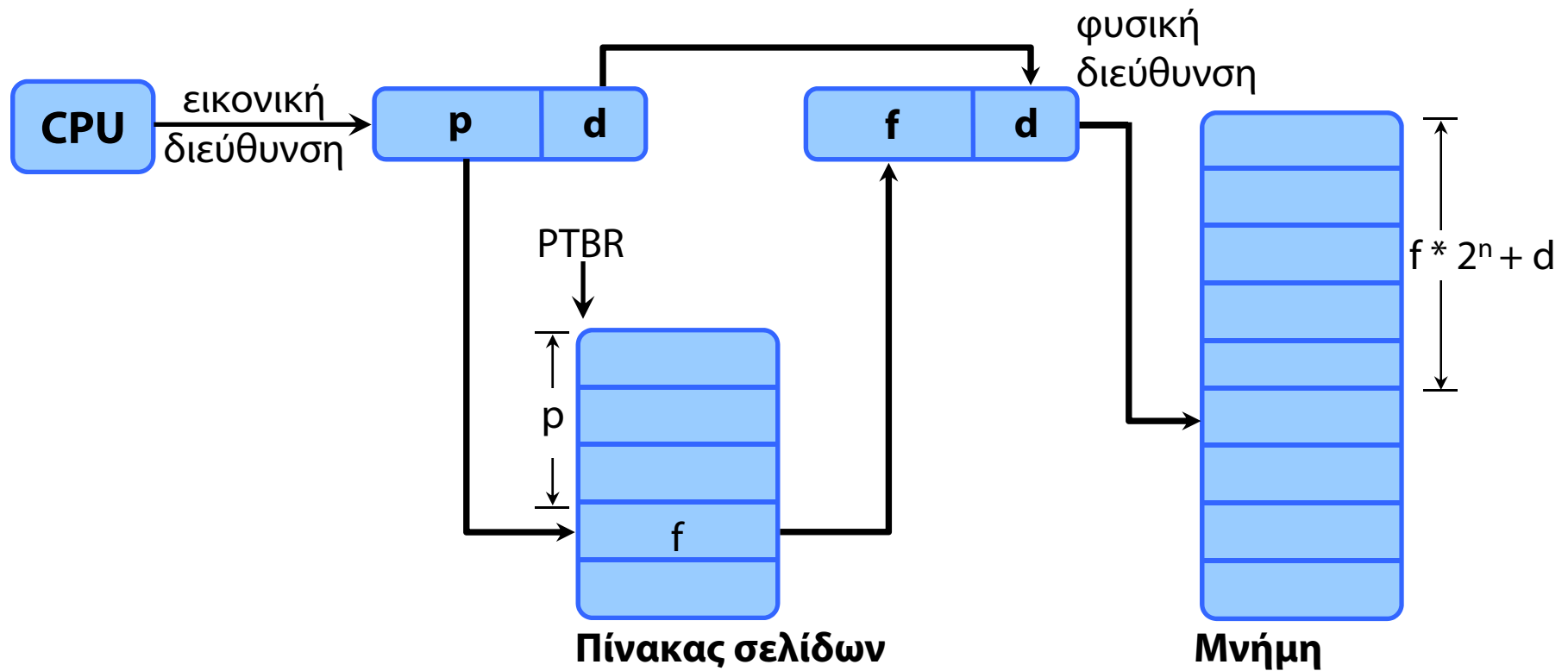
# Υποστήριξη υλικού για σελιδοποίηση

- ◆ Πώς υλοποιείται ο πίνακας σελίδων;
  - ➔ Ειδικοί, αφιερωμένοι καταχωρητές (DEC PDP-11)
    - ναι, αλλά είχε 16-bit διευθύνσεις και μέγεθος σελίδας 8KB
  - ➔ Πίνακας σελίδων στη μνήμη
    - Καταχωρητής βάσης πίνακα σελίδων – PTBR (πχ. %cr3 στον x86)
    - Πόσες προσβάσεις χρειάζονται στη μνήμη για *κάθε* πρόσβαση από τη CPU;
- ◆ Κρυφή μνήμη για τις μεταφράσεις:  
Translation Look-aside Buffer (TLB)
  - ➔ Συσχετιστική, πολύ γρήγορη μνήμη

# Υποστήριξη υλικού για σελιδοποίηση

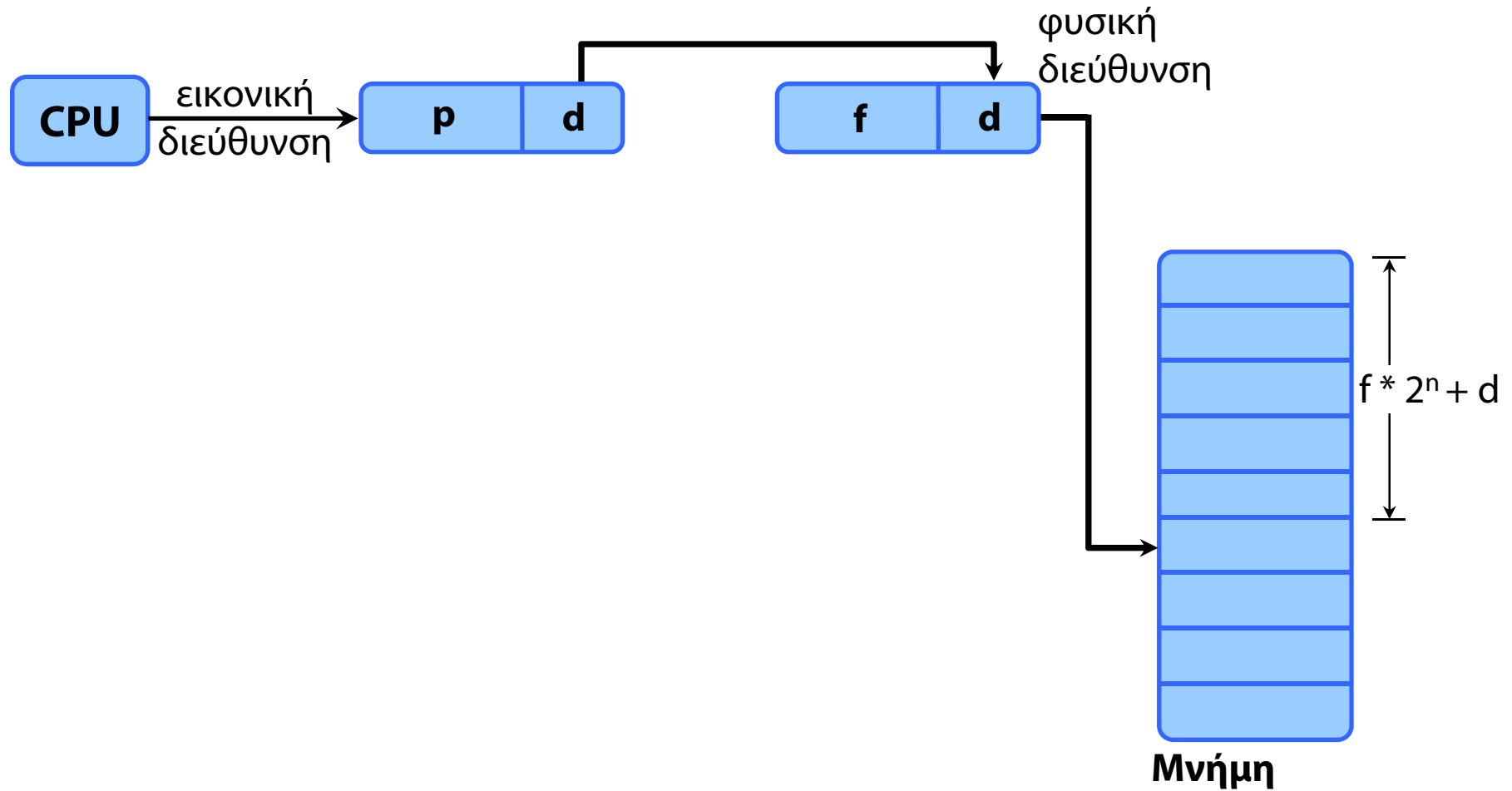


# Υποστήριξη υλικού για σελιδοποίηση

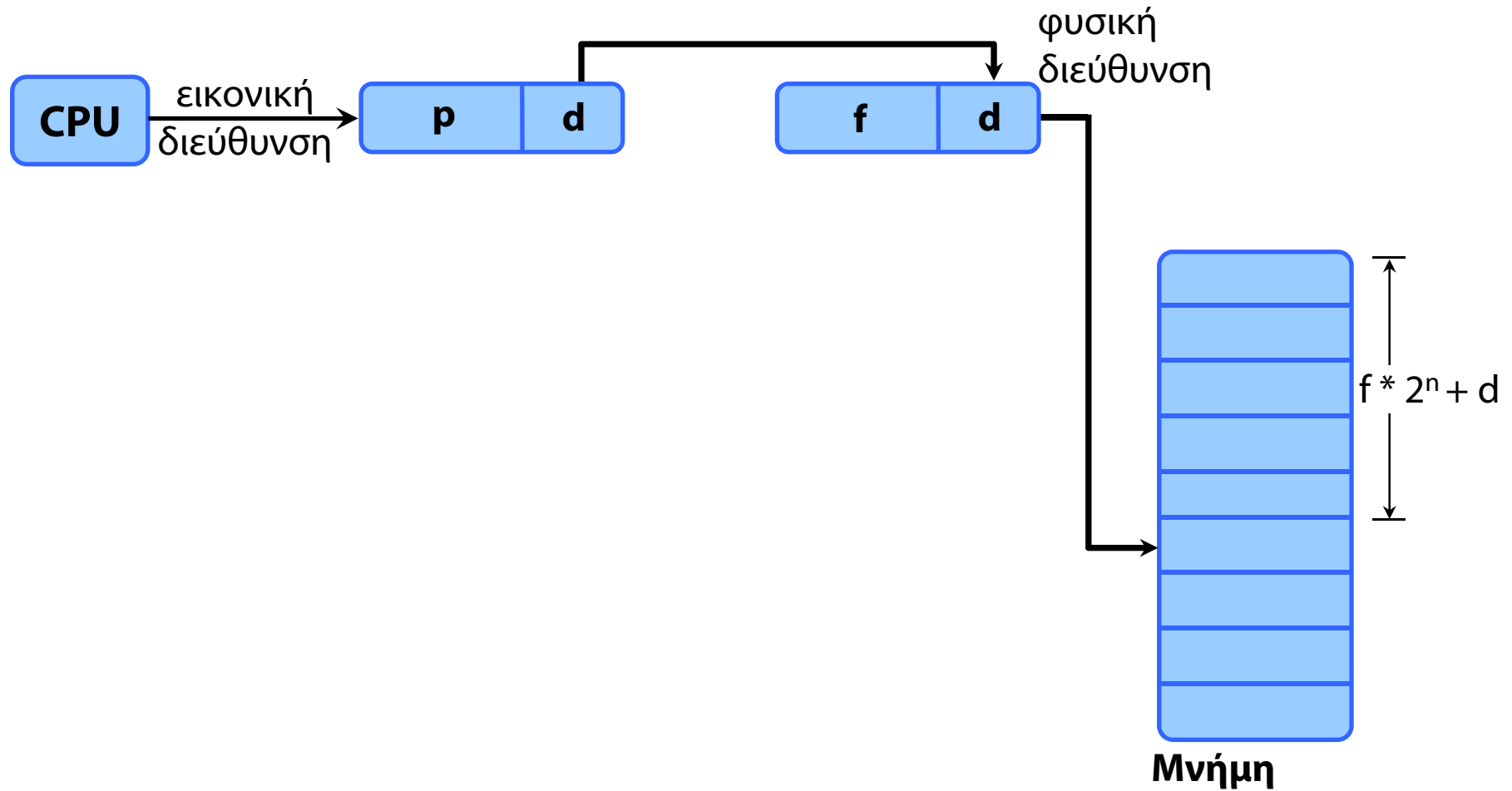




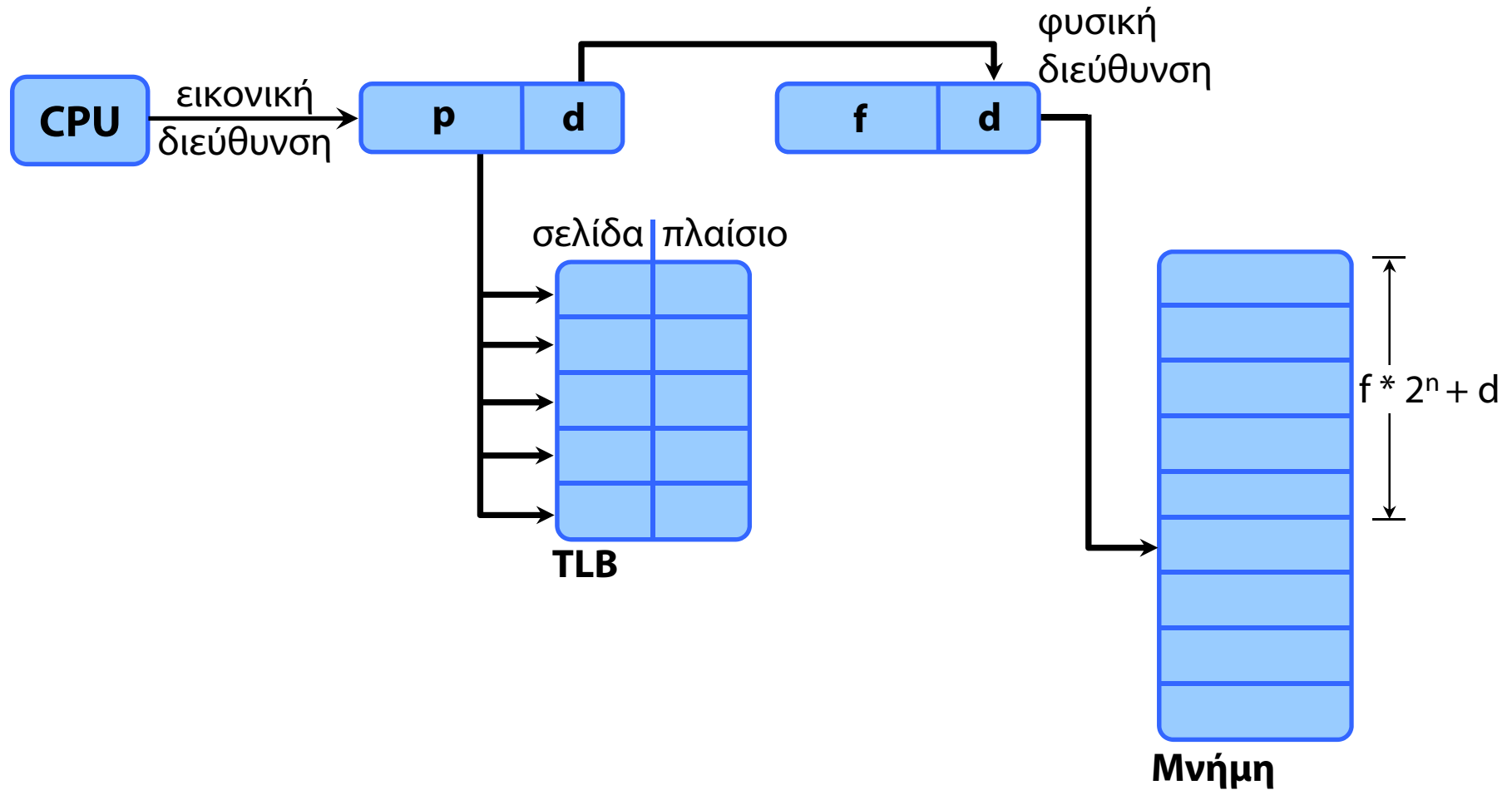
# Υποστήριξη υλικού για σελιδοποίηση – TLB (1)



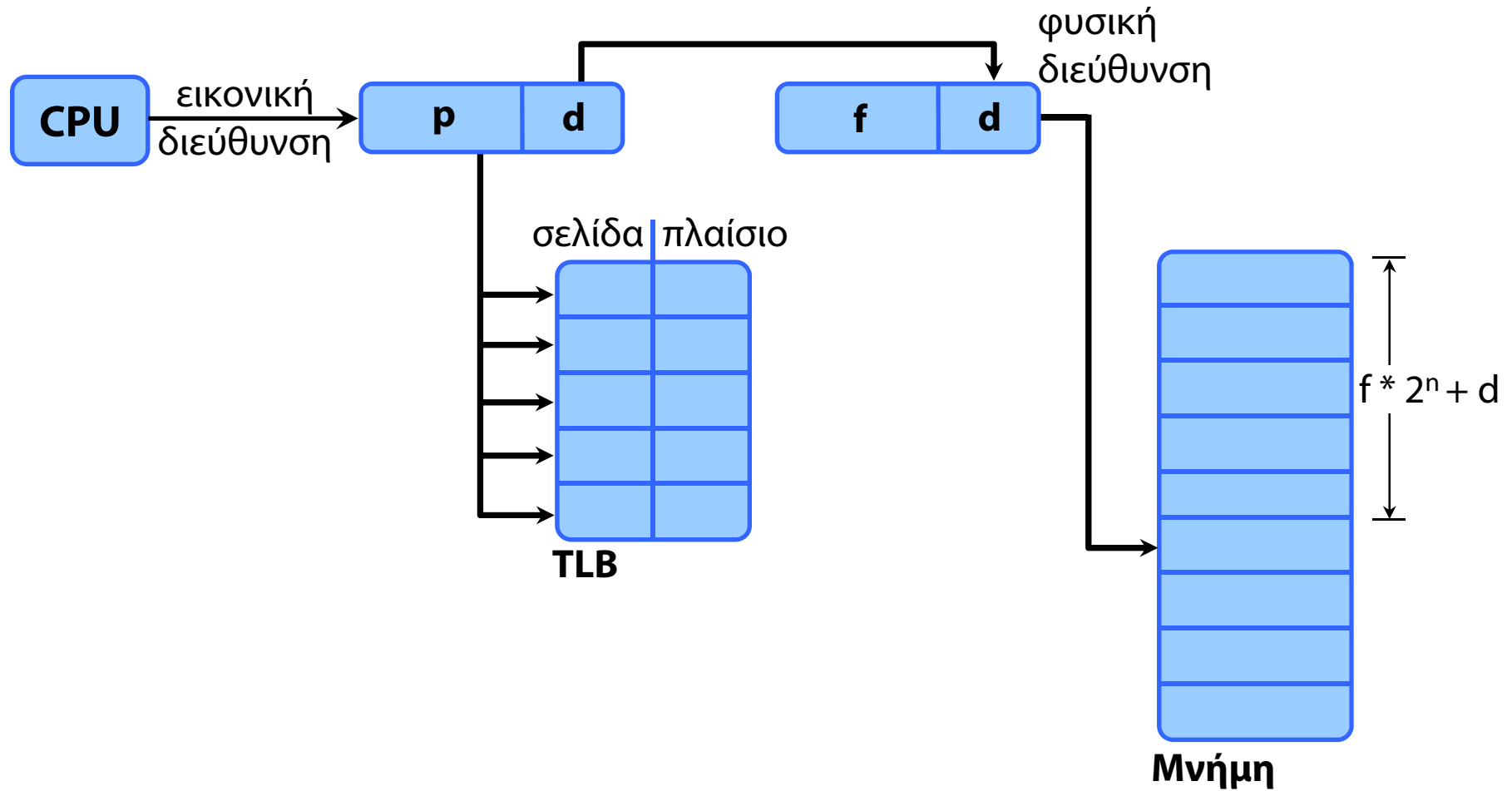
# Υποστήριξη υλικού για σελιδοποίηση – TLB (1)



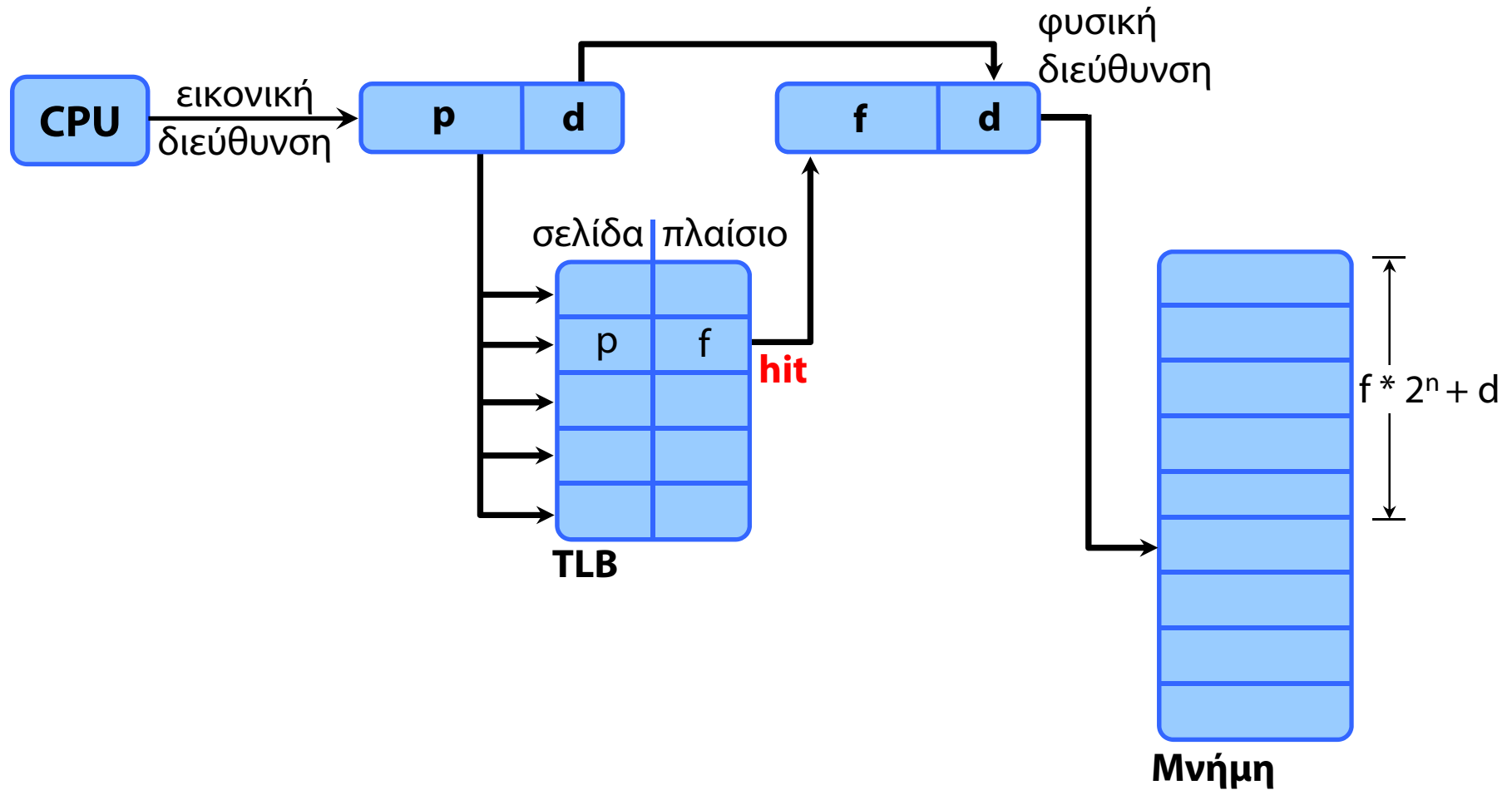
# Υποστήριξη υλικού για σελιδοποίηση – TLB (1)



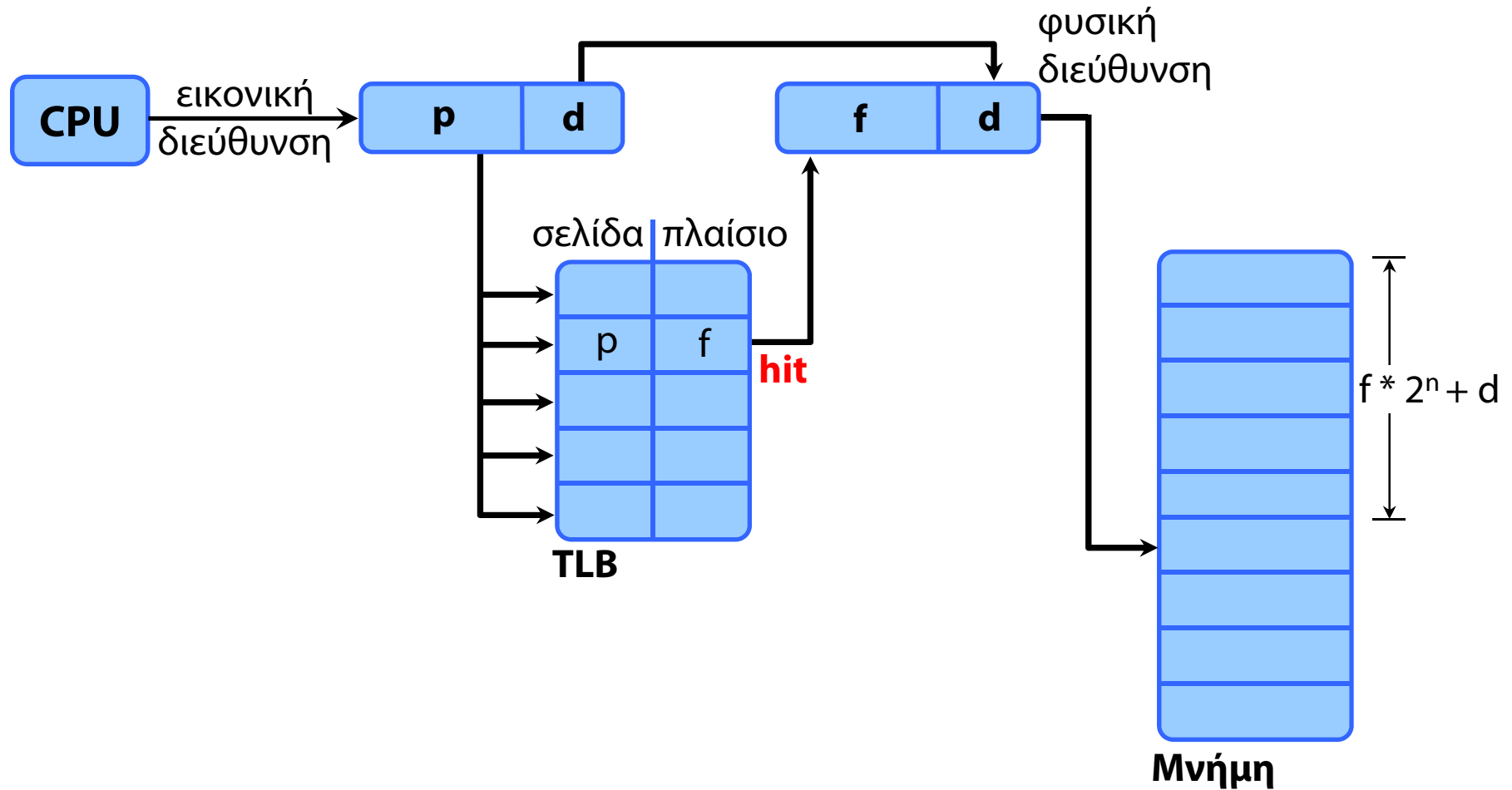
# Υποστήριξη υλικού για σελιδοποίηση – TLB (1)



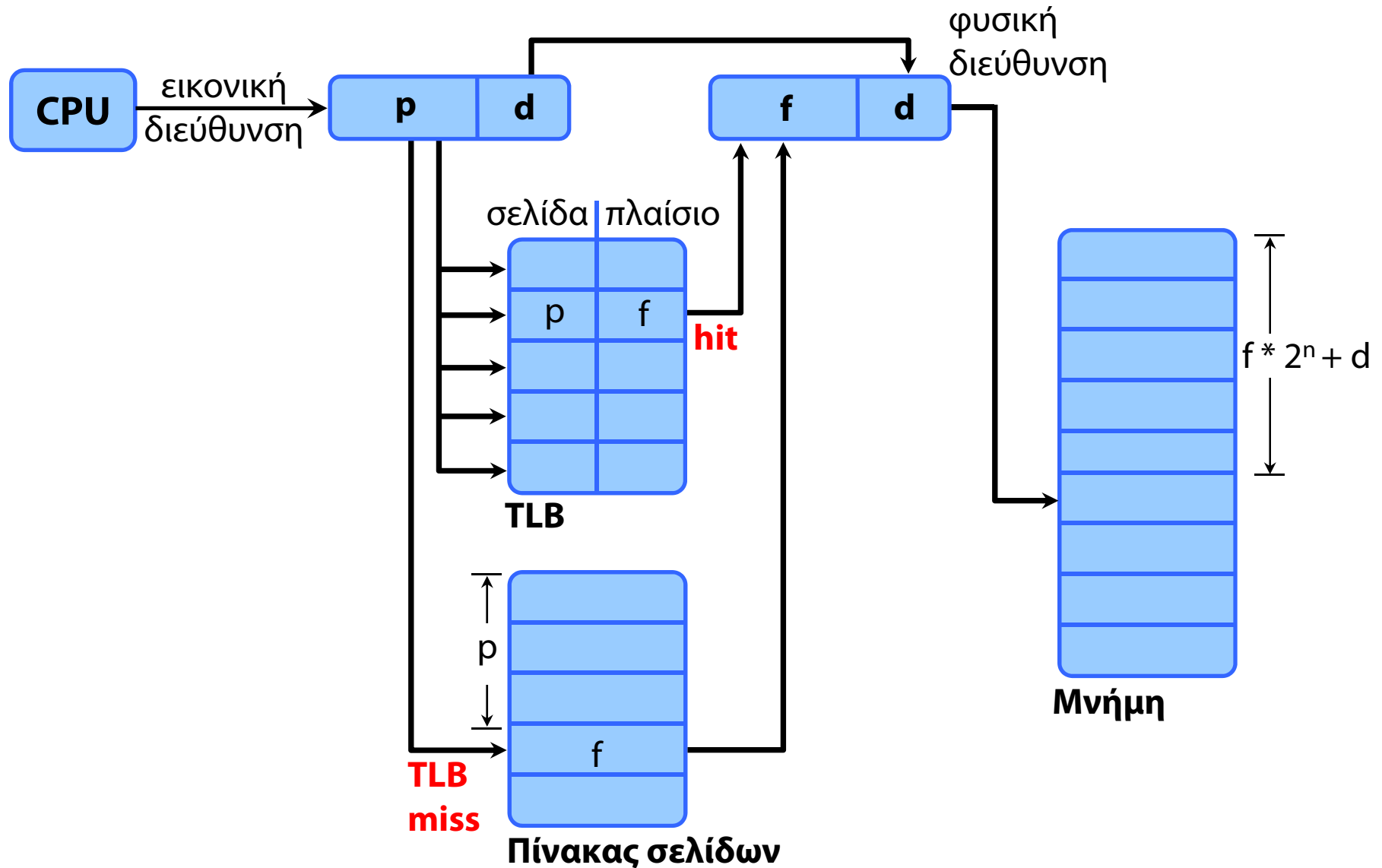
# Υποστήριξη υλικού για σελιδοποίηση – TLB (1)



# Υποστήριξη υλικού για σελιδοποίηση – TLB (1)



# Υποστήριξη υλικού για σελιδοποίηση – TLB (1)



## Υποστήριξη υλικού για σελιδοποίηση – TLB (2)

- ◆ Ο TLB έχει μικρό αριθμό εγγραφών (64-1024)
- ◆ Τι συμβαίνει αν υπάρξει TLB miss;
  - ➔ διάσχιση του πίνακα σελίδων στη μνήμη (page table walk) – κοστίζει πολύ
- ◆ Ποιος κάνει τη διάσχιση;



## Υποστήριξη υλικού για σελιδοποίηση – TLB (2)

- ◆ Ο TLB έχει μικρό αριθμό εγγραφών (64-1024)
- ◆ Τι συμβαίνει αν υπάρξει TLB miss;
  - ➔ διάσχιση του πίνακα σελίδων στη μνήμη (page table walk) – κοστίζει πολύ
- ◆ Ποιος κάνει τη διάσχιση;
  - ➔ *H CPU*: Hardware-managed TLBs (Intel x86)
  - ➔ *To ΛΣ*: Software-managed TLBs (MIPS, SPARC)
- ◆ Με software-managed TLB, το ΛΣ αποφασίζει ελεύθερα για την οργάνωση του πίνακα σελίδων

## Υποστήριξη υλικού για σελιδοποίηση – TLB (3)



- ◆ Τι συμβαίνει σε context switch;

## Υποστήριξη υλικού για σελιδοποίηση – TLB (3)

- ◆ Τι συμβαίνει σε context switch;
  - ➔ Οι εγγραφές του TLB είναι άκυρες
    - Καθαρισμός TLB – TLB flush
  - ➔ Καλύτερο: Αναγνωριστικά χώρου διευθύνσεων
    - Address-Space ID (ASID) για κάθε εγγραφή του TLB
    - Νήματα της ίδιας διεργασίας → ίδιο αναγνωριστικό

## Υποστήριξη υλικού για σελιδοποίηση – TLB (3)

- ◆ Τι συμβαίνει σε context switch;
  - ➔ Οι εγγραφές του TLB είναι άκυρες
    - Καθαρισμός TLB – TLB flush
  - ➔ Καλύτερο: Αναγνωριστικά χώρου διευθύνσεων
    - Address-Space ID (ASID) για κάθε εγγραφή του TLB
    - Νήματα της ίδιας διεργασίας → ίδιο αναγνωριστικό

## Υποστήριξη υλικού για σελιδοποίηση – TLB (3)

- ◆ Τι συμβαίνει σε context switch;
  - ➔ Οι εγγραφές του TLB είναι άκυρες
    - Καθαρισμός TLB – TLB flush
  - ➔ Καλύτερο: Αναγνωριστικά χώρου διευθύνσεων
    - Address-Space ID (ASID) για κάθε εγγραφή του TLB
    - Νήματα της ίδιας διεργασίας → ίδιο αναγνωριστικό
- ◆ Πραγματικός χρόνος πρόσβασης
  - ➔ Αν  $h$  = TLB hit ratio,  $\epsilon$  = κόστος αναζήτησης στο TLB,  $\tau$  = χρόνος πρόσβασης στη μνήμη
  - ➔ Effective Access Time =  $(\epsilon + \tau) h + (\epsilon + 2\tau) (1 - h)$

# Σελιδοποίηση – Προστασία (1)

- ◆ Επιβολή δικαιωμάτων πρόσβασης
  - ➔ με bits προστασίας ανά σελίδα εικονικής μνήμης
- ◆ Π.χ., για μοιραζόμενη μνήμη
  - ➔ Μία διεργασία μπορεί να γράψει, οι άλλες μόνο διαβάζουν
- ◆ Bits πρόσβασης
  - ➔ **R**ead, **W**rite, **eX**ecute
- ◆ Bit εγκυρότητας
  - ➔ **V**alid, **I**nvalid
- ◆ Σε περίπτωση μη επιτρεπόμενης πρόσβασης;

# Σελιδοποίηση – Προστασία (1)

- ◆ Επιβολή δικαιωμάτων πρόσβασης
  - ➔ με bits προστασίας ανά σελίδα εικονικής μνήμης
- ◆ Π.χ., για μοιραζόμενη μνήμη
  - ➔ Μία διεργασία μπορεί να γράψει, οι άλλες μόνο διαβάζουν
- ◆ Bits πρόσβασης
  - ➔ **R**ead, **W**rite, **eX**ecute
- ◆ Bit εγκυρότητας
  - ➔ **V**alid, **I**nvalid
- ◆ Σε περίπτωση μη επιτρεπόμενης πρόσβασης;
  - ➔ Trap! Εξαίρεση σελίδας – **Page fault**

# Σελιδοποίηση – Προστασία (2)

|          |
|----------|
| σελίδα 0 |
| σελίδα 1 |
| σελίδα 2 |
| σελίδα 3 |
| σελίδα 4 |

**Εικονική  
Μνήμη**

|   |
|---|
| 2 |
| 5 |
| 1 |
| 7 |
| 0 |

**Πίνακας  
σελίδων**

|   |          |
|---|----------|
| 0 |          |
| 1 | σελίδα 2 |
| 2 | σελίδα 0 |
| 3 |          |
| 4 |          |
| 5 | σελίδα 1 |
| 6 |          |
| 7 | σελίδα 3 |
| 8 |          |

**Φυσική  
Μνήμη**



# Σελιδοποίηση – Προστασία (2)

|          |
|----------|
| σελίδα 0 |
| σελίδα 1 |
| σελίδα 2 |
| σελίδα 3 |
| σελίδα 4 |

**Εικονική  
Μνήμη**

|   |      |   |
|---|------|---|
| 2 | r, x | v |
| 5 | r, x | v |
| 1 | r    | v |
| 7 | r, w | v |
| 0 |      | i |

**Πίνακας  
σελίδων**

|   |          |
|---|----------|
| 0 |          |
| 1 | σελίδα 2 |
| 2 | σελίδα 0 |
| 3 |          |
| 4 |          |
| 5 | σελίδα 1 |
| 6 |          |
| 7 | σελίδα 3 |
| 8 |          |

**Φυσική  
Μνήμη**

## Σελιδοποίηση – Προστασία (2)

|          |
|----------|
| σελίδα 0 |
| σελίδα 1 |
| σελίδα 2 |
| σελίδα 3 |
| σελίδα 4 |

**Εικονική  
Μνήμη**

|   |      |   |
|---|------|---|
| 2 | r, x | v |
| 5 | r, x | v |
| 1 | r    | v |
| 7 | r, w | v |
| 0 |      | i |

**Πίνακας  
σελίδων**

|   |          |
|---|----------|
| 0 |          |
| 1 | σελίδα 2 |
| 2 | σελίδα 0 |
| 3 |          |
| 4 |          |
| 5 | σελίδα 1 |
| 6 |          |
| 7 | σελίδα 3 |
| 8 |          |


**Φυσική  
Μνήμη**

- ◆ Ποιες σελίδες είναι κειμένου-κώδικα, ποιες δεδομένων;

# Στον πραγματικό κόσμο

- ◆ Πόσο μεγάλος είναι ο πίνακας σελίδων;
- ◆ 32-bit εικονική διεύθυνση, μέγεθος σελίδας 4KB, 32-bit φυσική διεύθυνση:
  - ➔  $2^{32} / 4096 = 1048576$  εγγραφές, 4MB ανά διεργασία...
- ◆ Και σε 64-bit μηχανήματα;
- ◆ Οι διεργασίες δεν χρησιμοποιούν το σύνολο του χώρου εικονικών διευθύνσεων
- ◆ Αποδοτικότερη οργάνωση πίνακα σελίδων
  - ➔ Ιεραρχική σελιδοποίηση
  - ➔ Κατακερματισμένοι πίνακες σελίδων
  - ➔ Ανεστραμμένοι πίνακες σελίδων

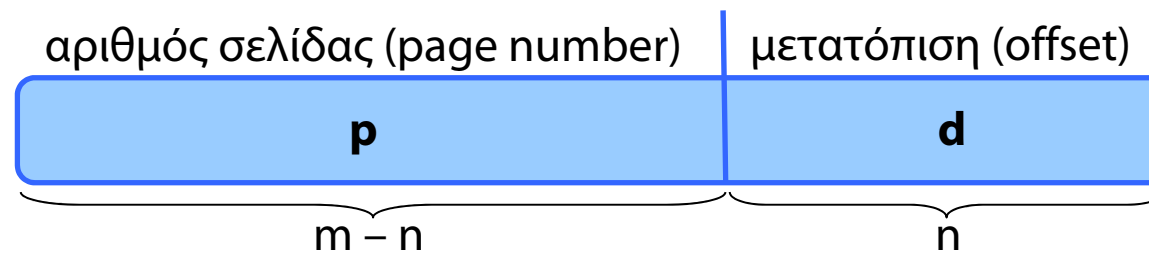
# Ιεραρχική σελιδοποίηση



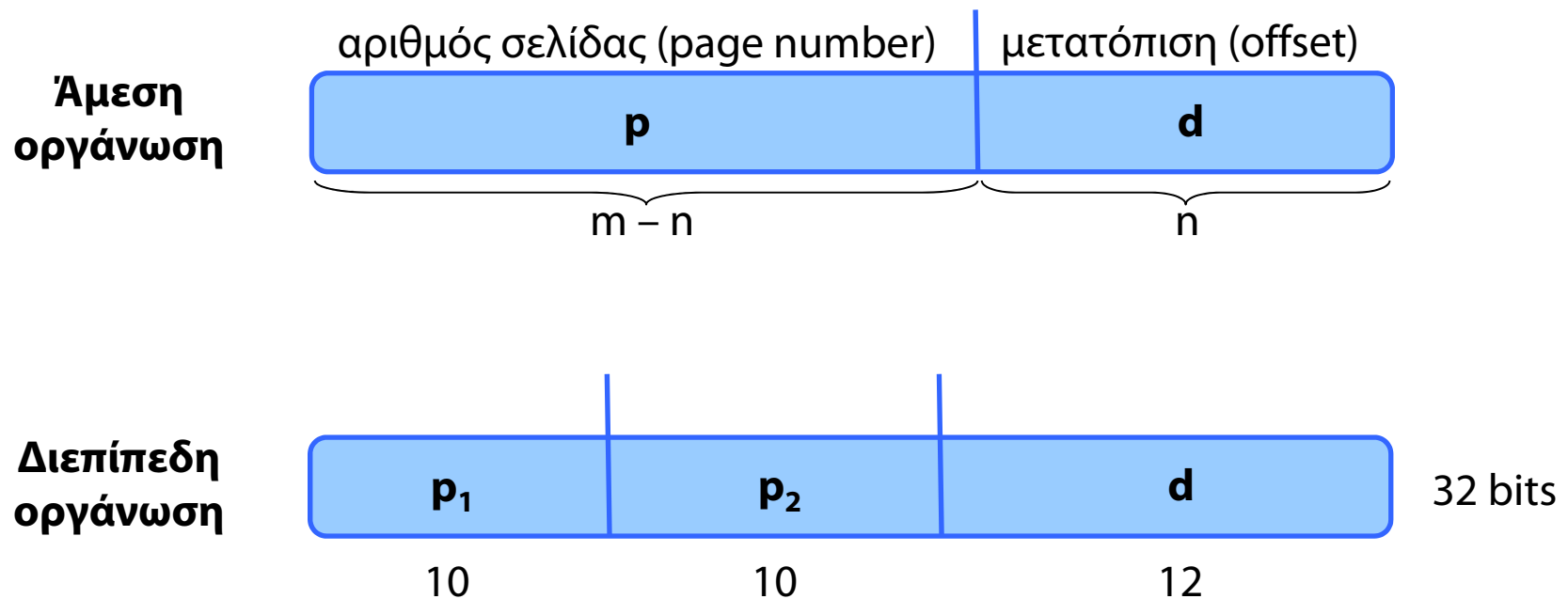
- ◆ Διαίρεση του χώρου λογικών διευθύνσεων σε περισσότερους πίνακες σελίδων
- ◆ Ιεραρχικά, π.χ. διεπίπεδη οργάνωση
  - ➔ Ο ίδιος ο πίνακας σελίδων σελιδοποιείται

# Διεπίπεδη Σελιδοποίηση (1)

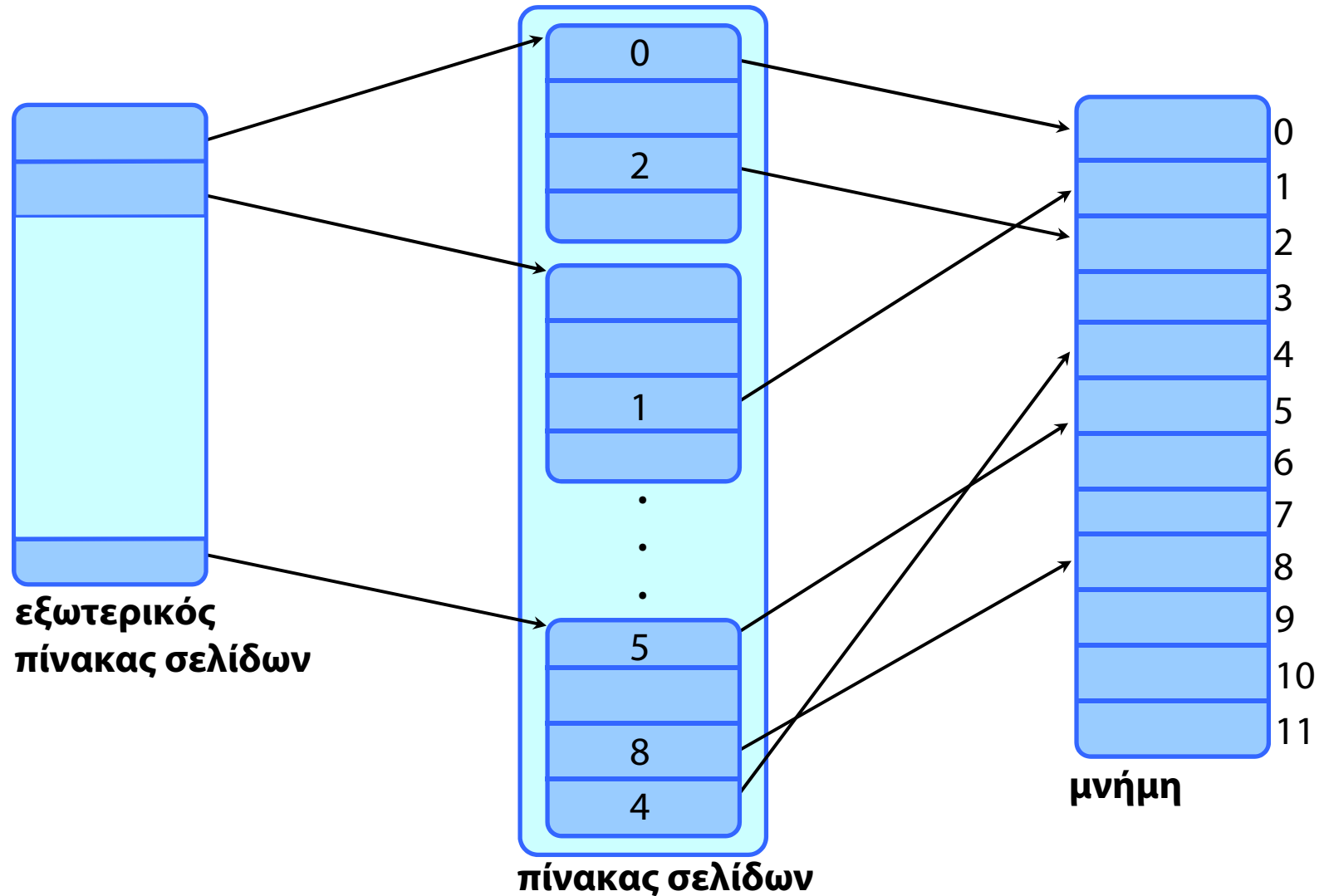
Άμεση  
οργάνωση



# Διεπίπεδη Σελιδοποίηση (1)



## Διεπίπεδη Σελιδοποίηση (2)



# Διάσχιση σε διεπίπεδη οργάνωση

$p_1$

$p_2$

$d$



# Διάσχιση σε διεπίπεδη οργάνωση

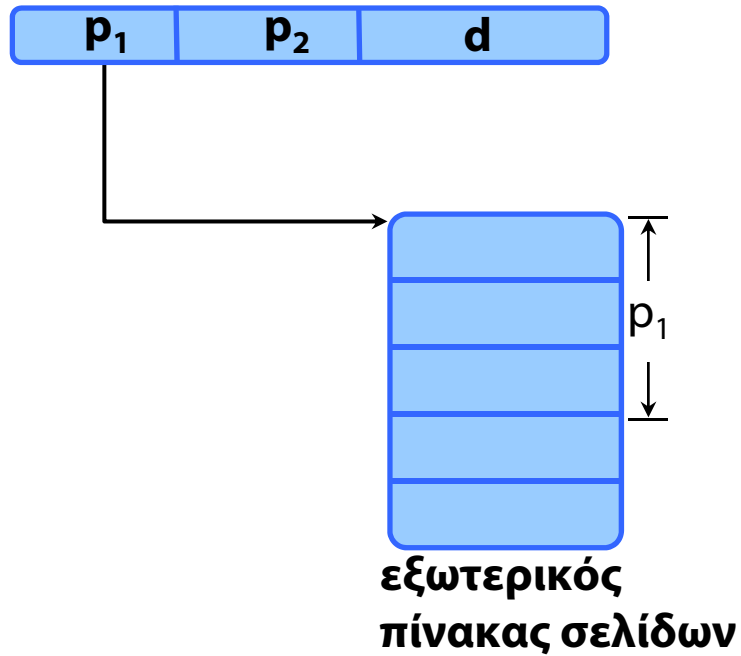
|       |       |     |
|-------|-------|-----|
| $p_1$ | $p_2$ | $d$ |
|-------|-------|-----|

**εξωτερικός  
πίνακας σελίδων**

**σελίδα  
πίνακα σελίδων**

**πλαίσιο μνήμης**

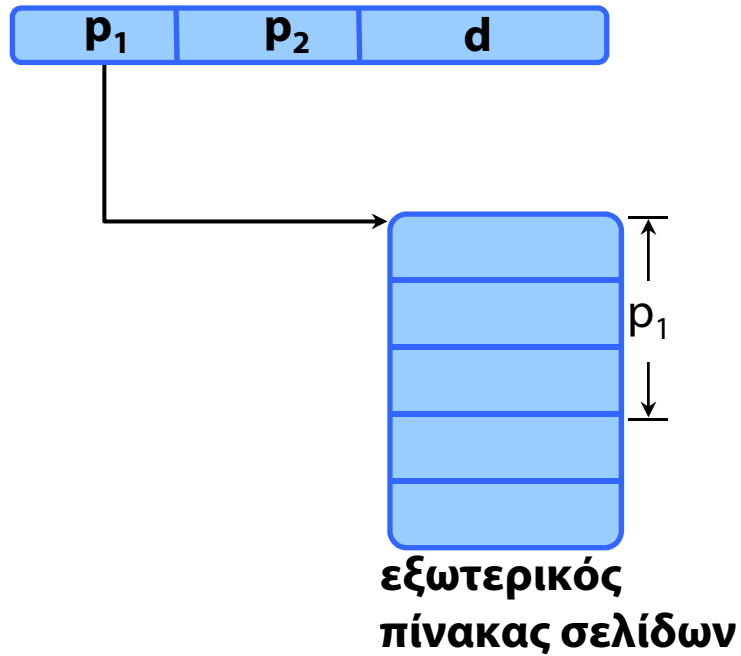
# Διάσχιση σε διεπίπεδη οργάνωση



σελίδα  
πίνακα σελίδων

πλαίσιο μνήμης

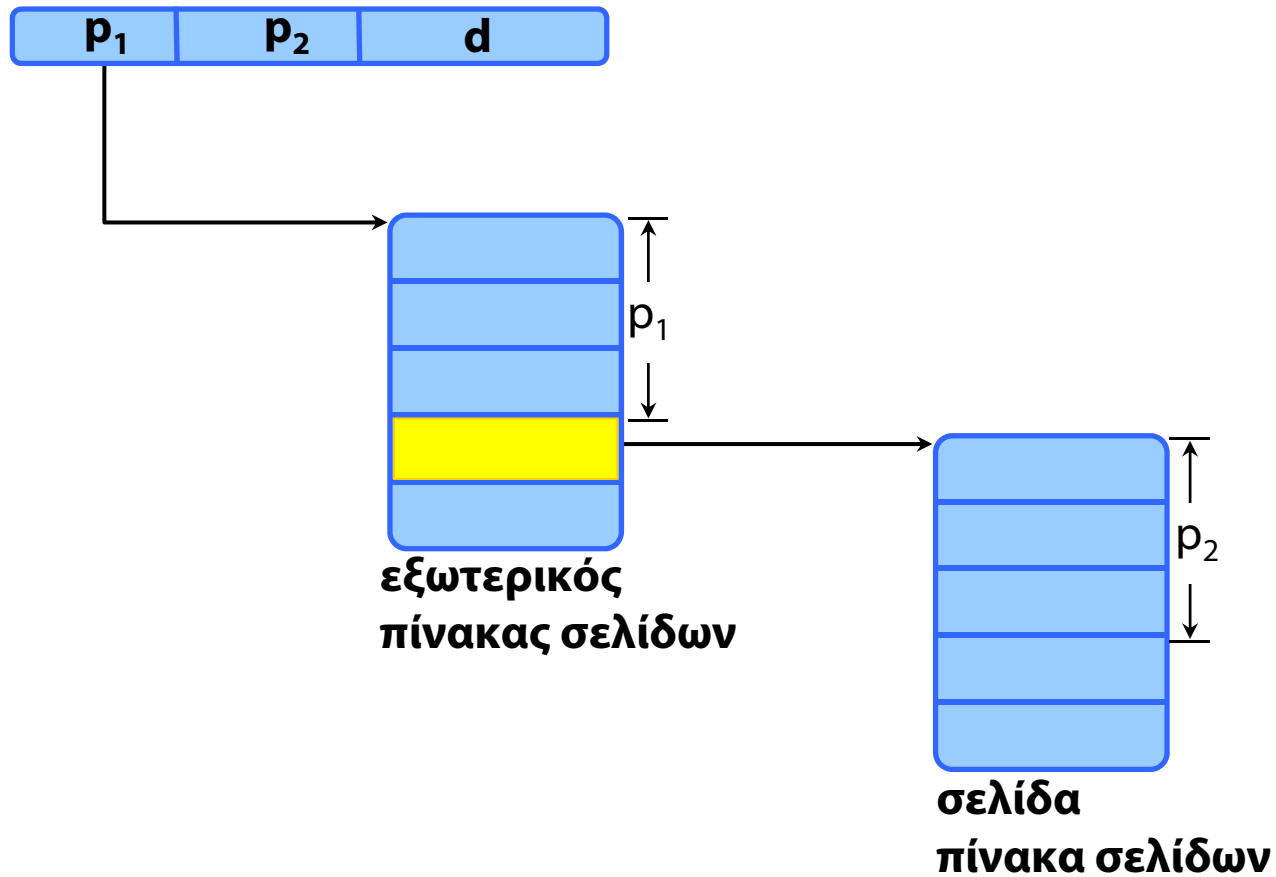
# Διάσχιση σε διεπίπεδη οργάνωση



σελίδα  
πίνακα σελίδων

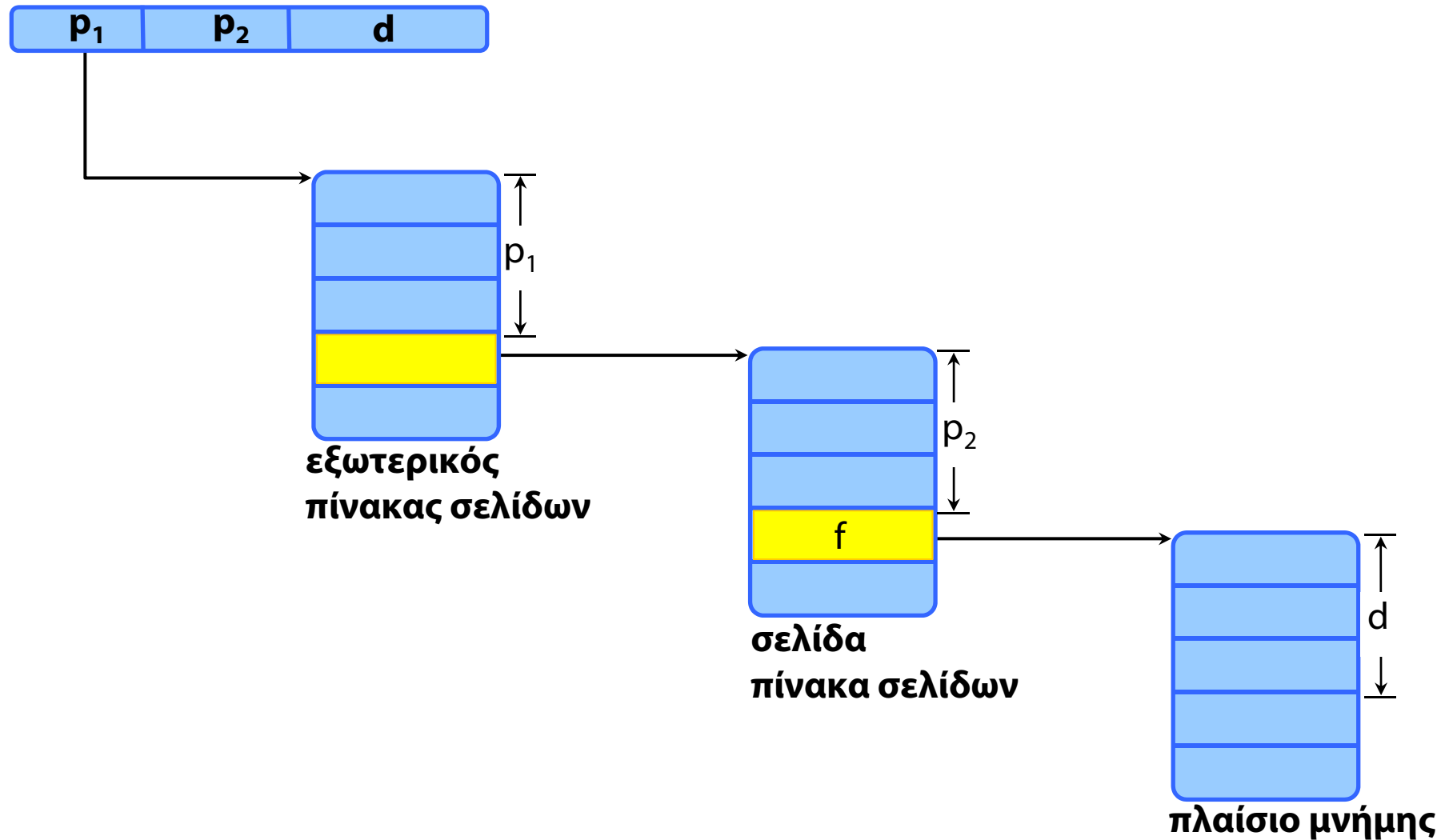
πλαίσιο μνήμης

# Διάσχιση σε διεπίπεδη οργάνωση

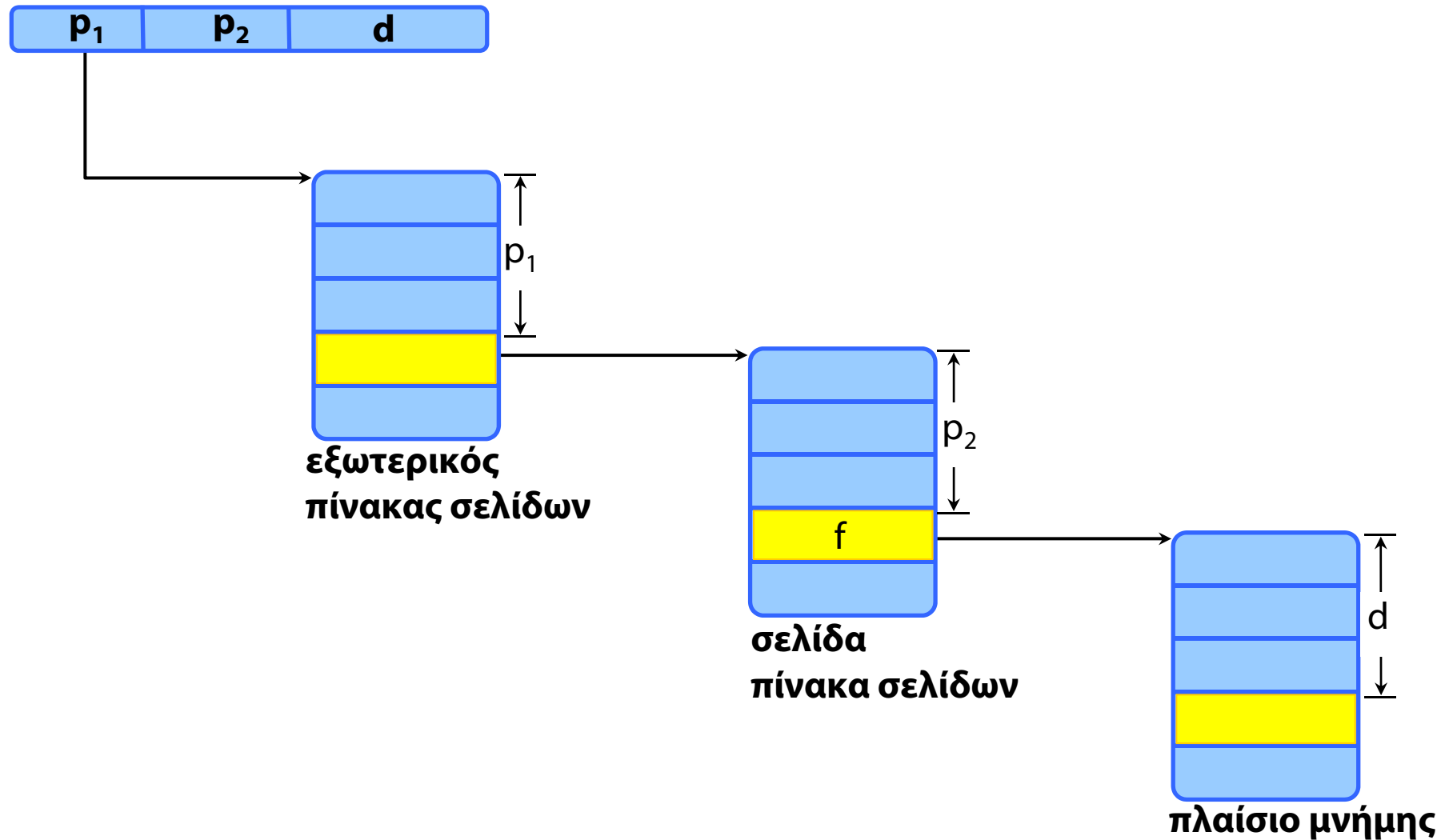


πλαίσιο μνήμης

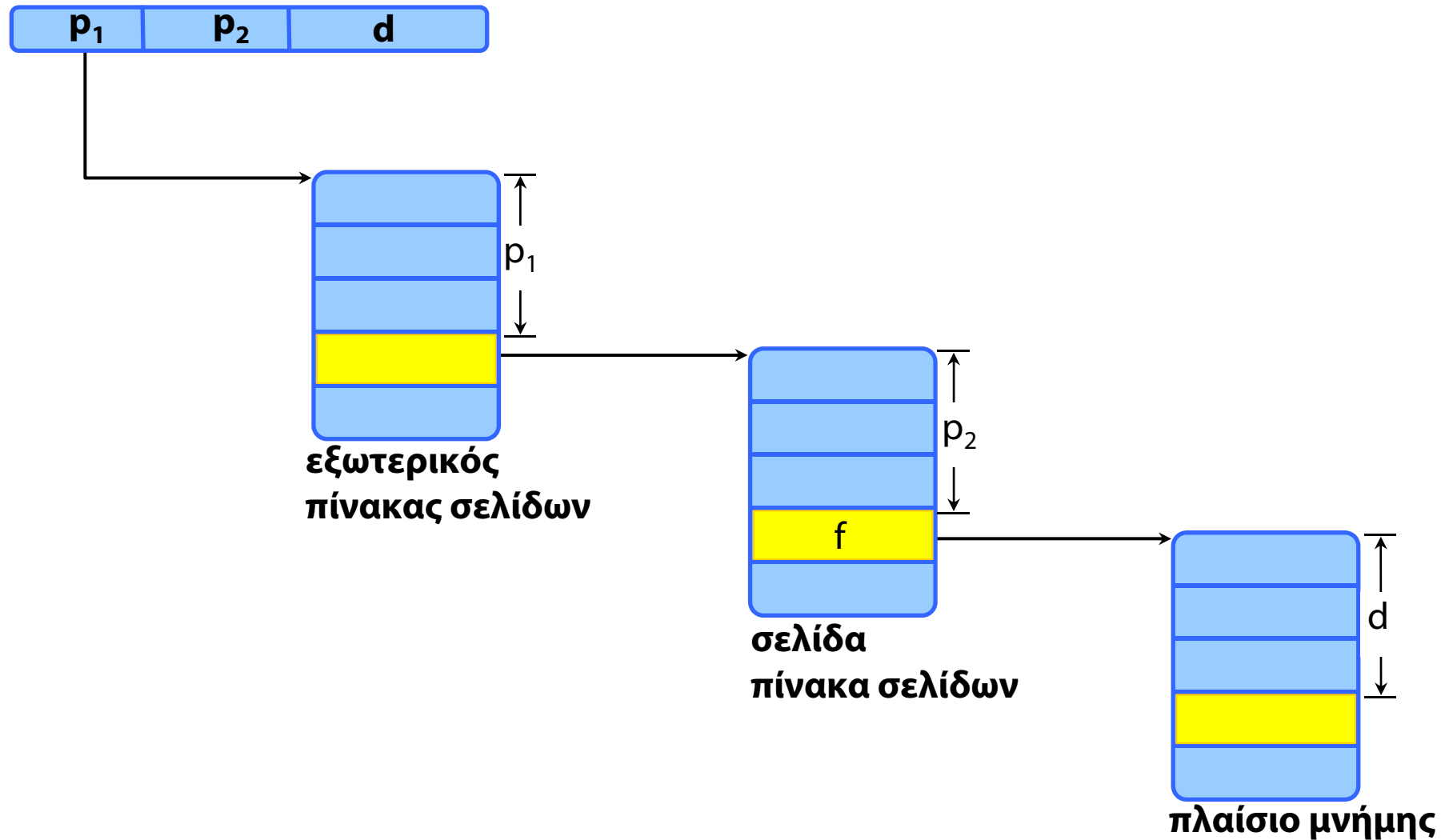
# Διάσχιση σε διεπίπεδη οργάνωση



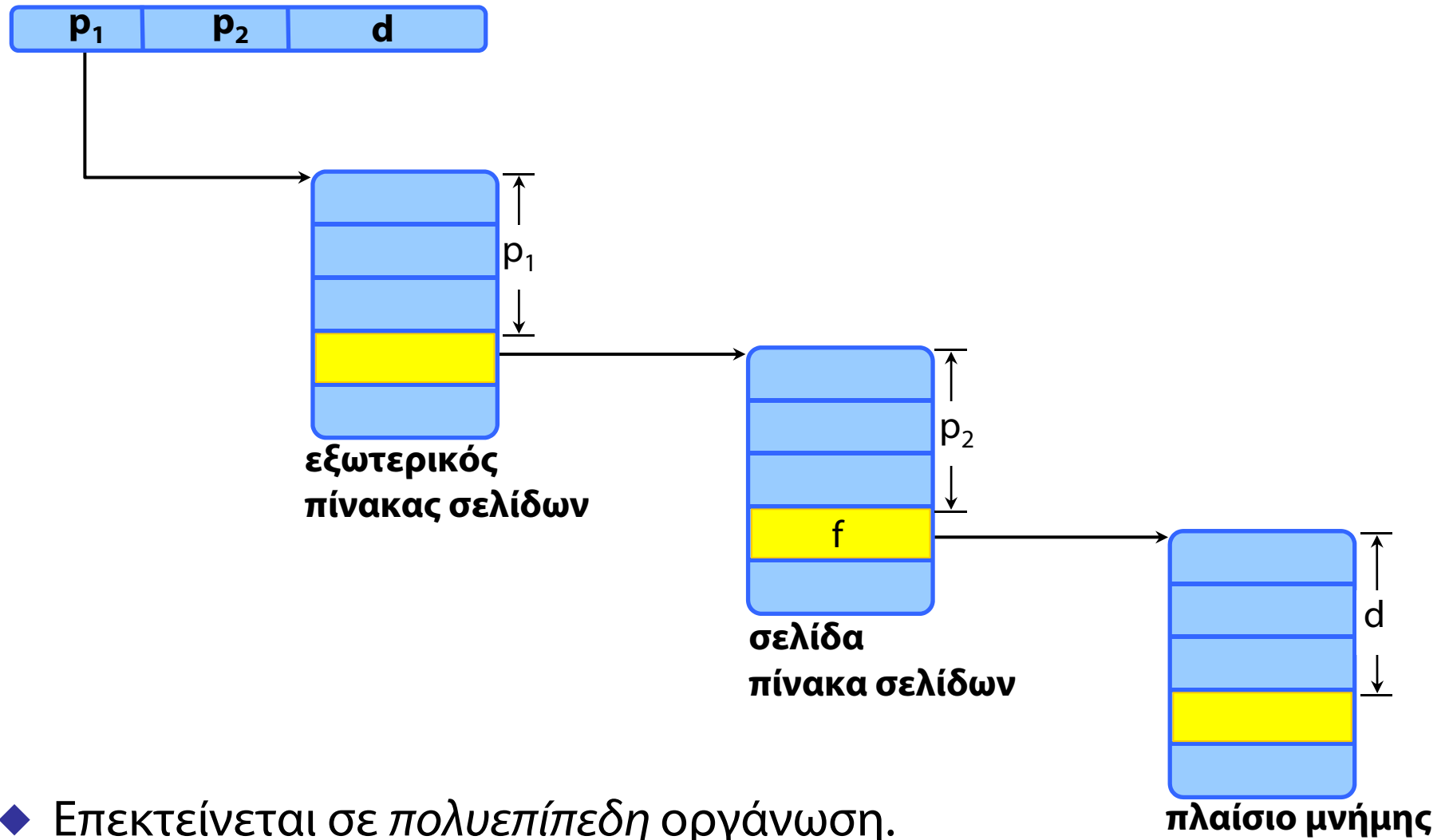
# Διάσχιση σε διεπίπεδη οργάνωση



# Διάσχιση σε διεπίπεδη οργάνωση

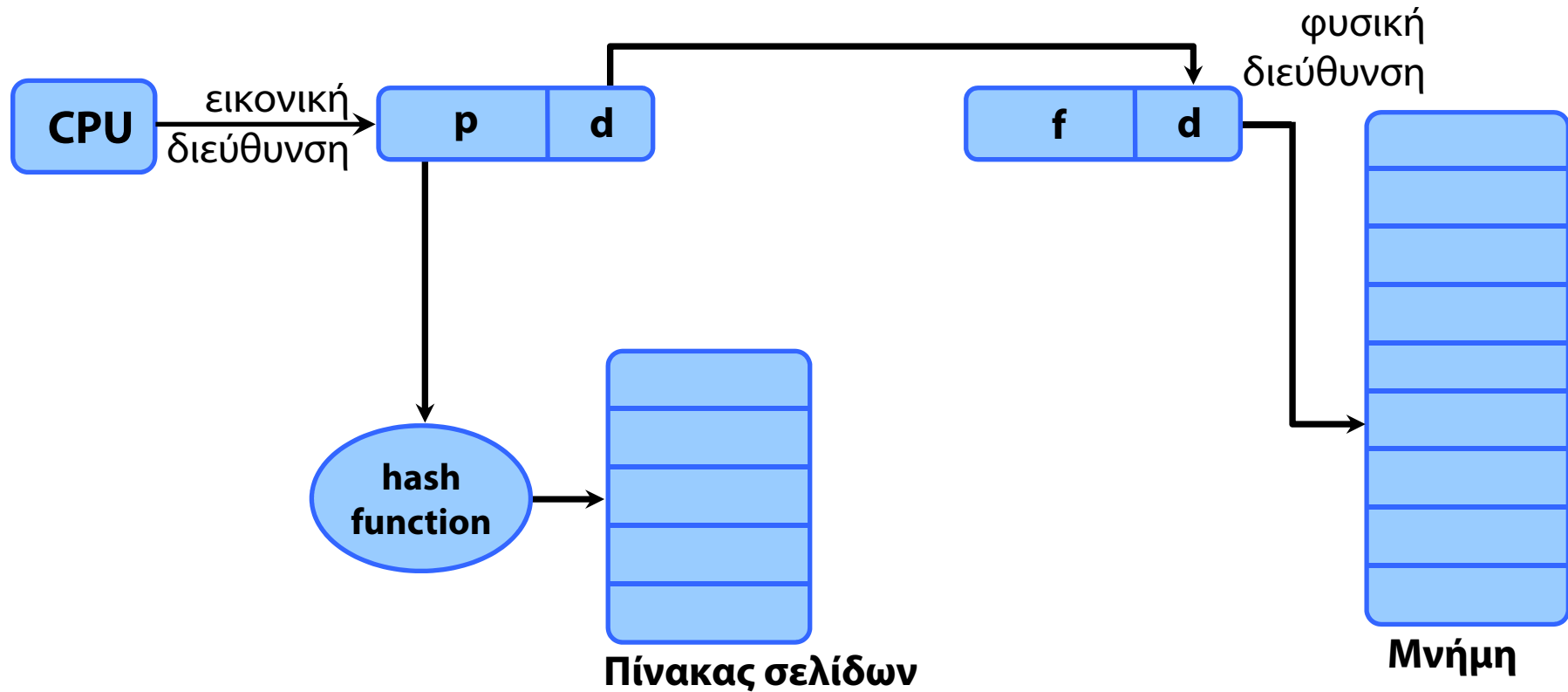


# Διάσχιση σε διεπίπεδη οργάνωση



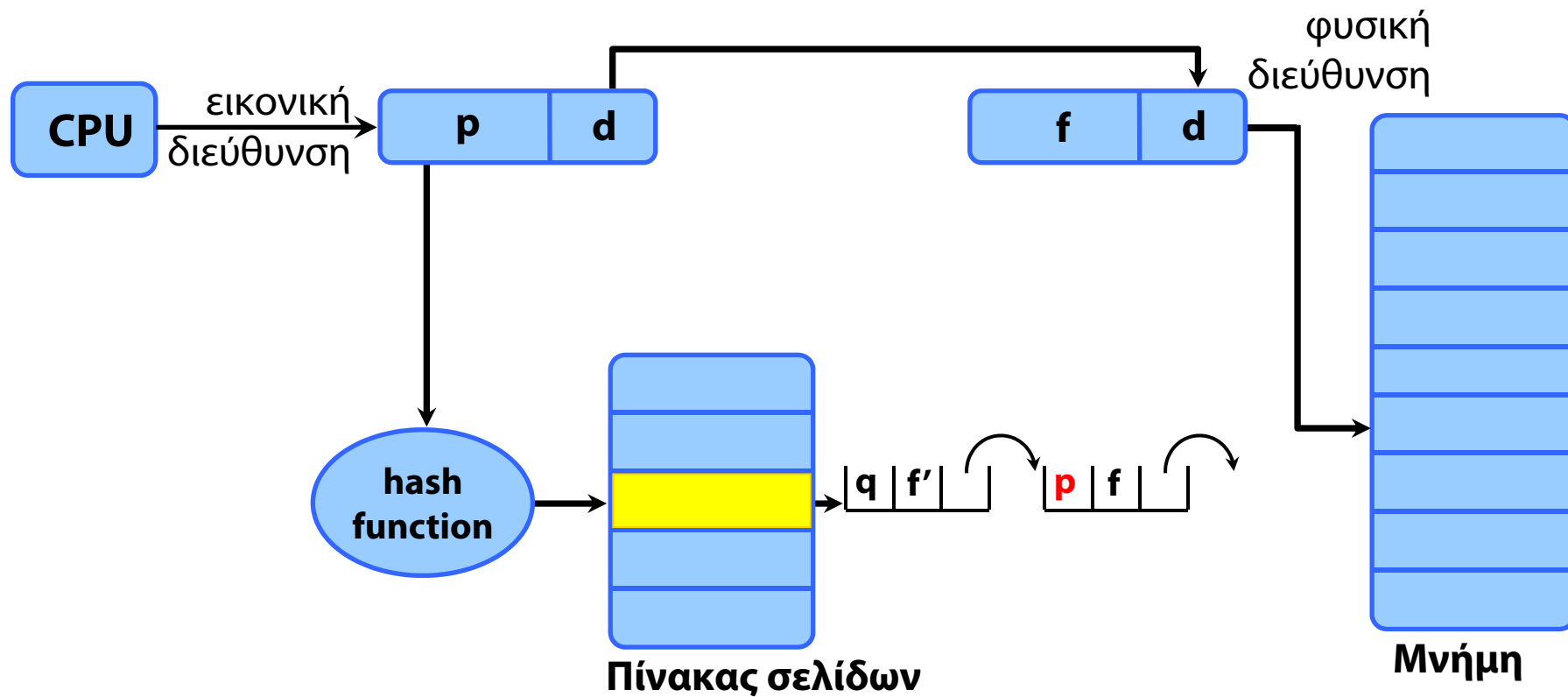


# Κατακερματισμένοι πίνακες σελίδων



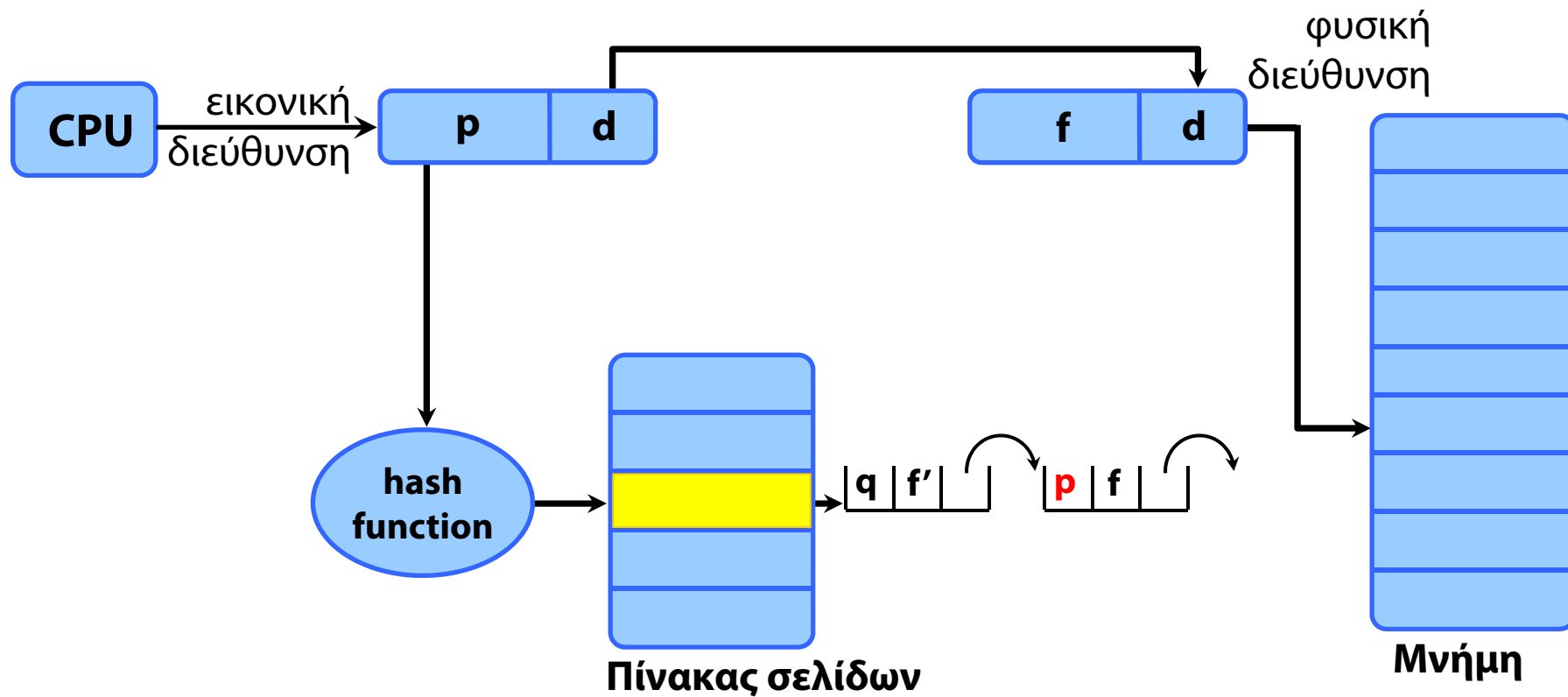
- ◆ Βολικοί για πολύ μεγάλους χώρους διευθύνσεων, π.χ. 64-bit
  - ➔ Solaris σε 64-bit UltraSPARC

# Κατακερματισμένοι πίνακες σελίδων



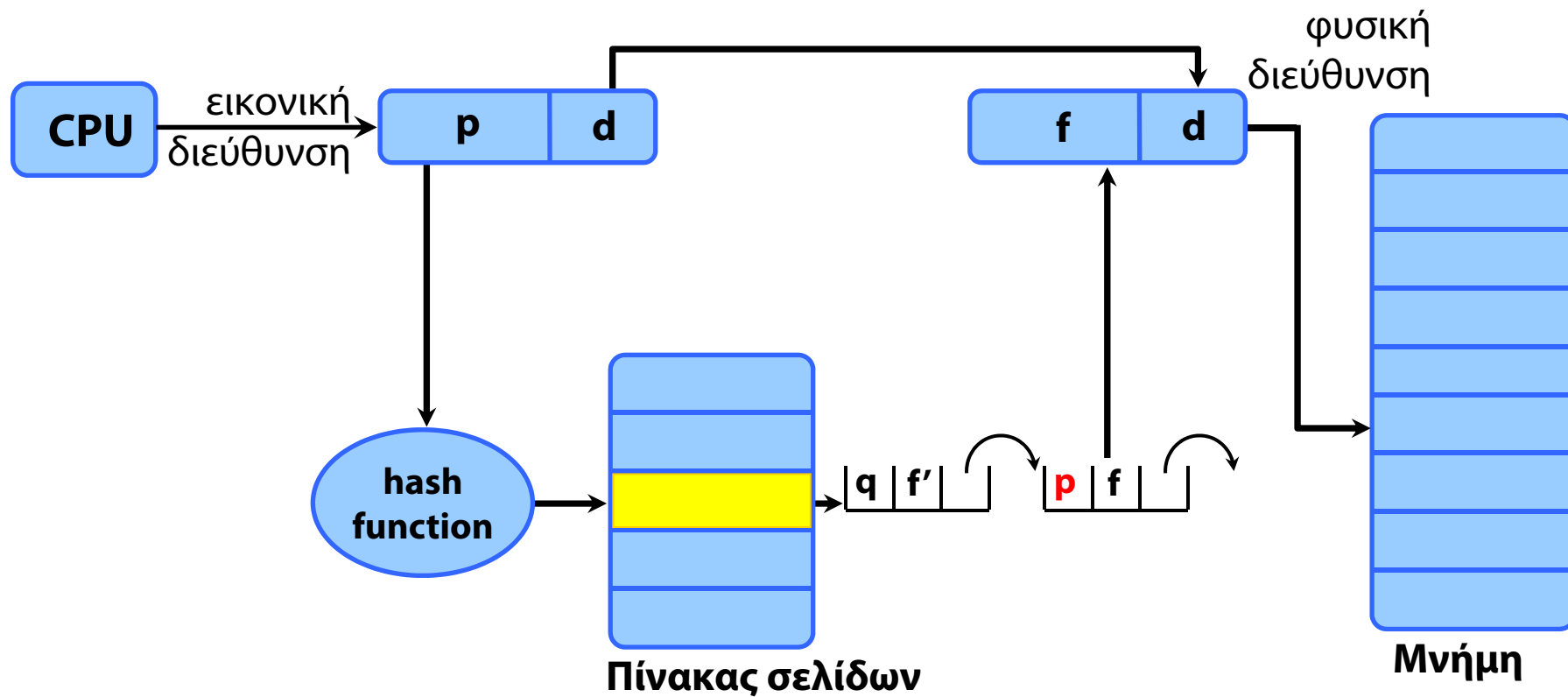
- ◆ Βολικοί για πολύ μεγάλους χώρους διευθύνσεων, π.χ. 64-bit
  - ➔ Solaris σε 64-bit UltraSPARC

# Κατακερματισμένοι πίνακες σελίδων



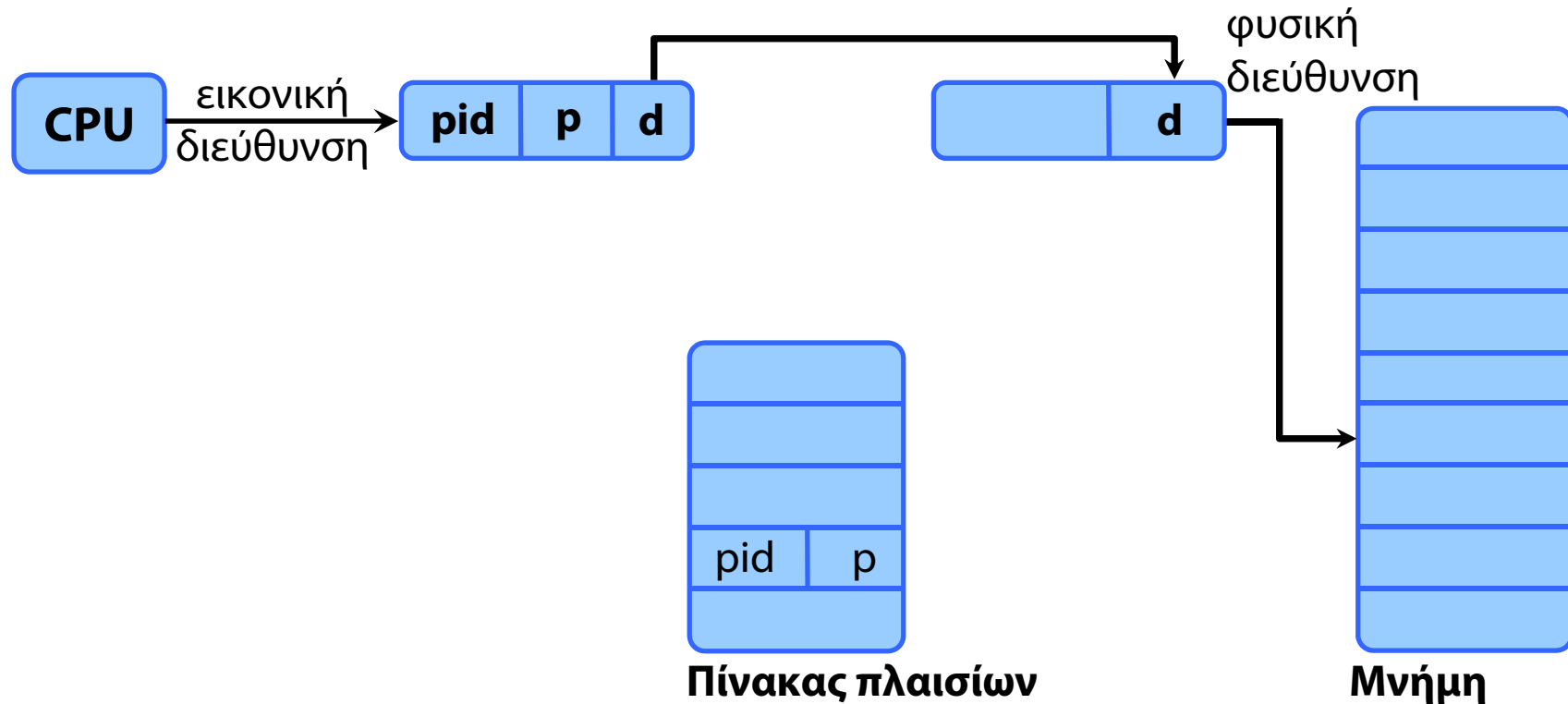
- ◆ Βολικοί για πολύ μεγάλους χώρους διευθύνσεων, π.χ. 64-bit
  - ➔ Solaris σε 64-bit UltraSPARC

# Κατακερματισμένοι πίνακες σελίδων



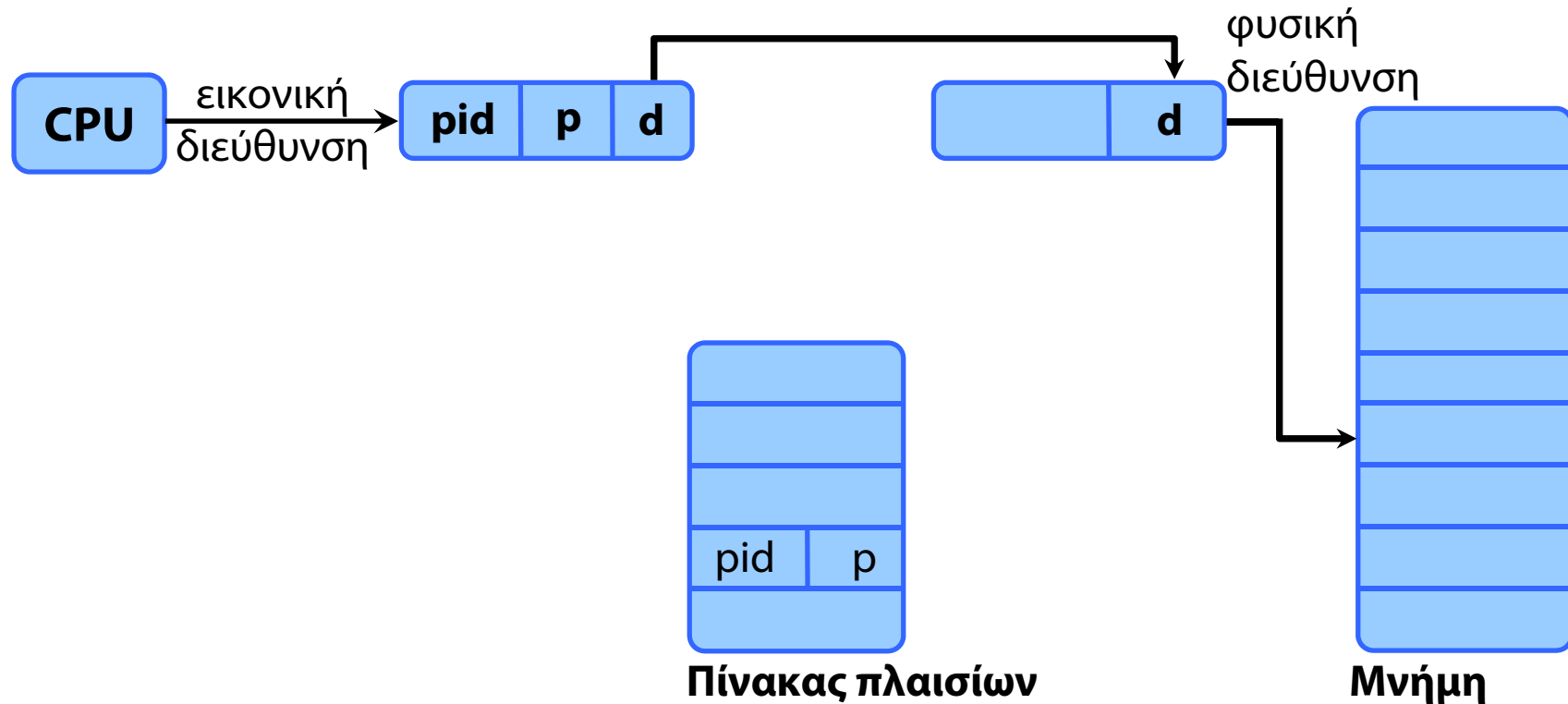
- ◆ Βολικοί για πολύ μεγάλους χώρους διευθύνσεων, π.χ. 64-bit
  - ➔ Solaris σε 64-bit UltraSPARC

# Ανεστραμμένοι πίνακες σελίδων



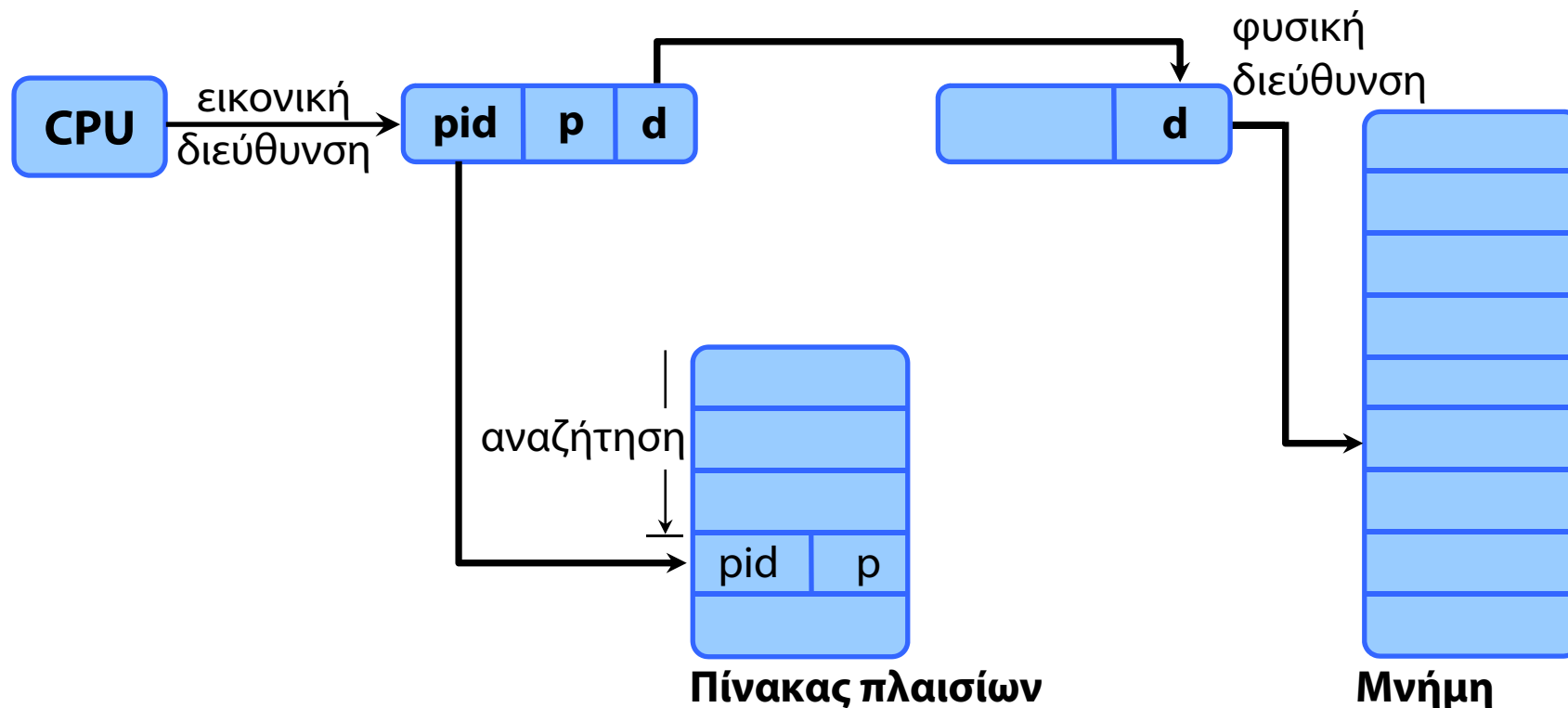
- ◆ Πίνακας πλαισίων αντί για πίνακας σελίδων
- ◆ Κρατάει την αντίστροφη πληροφορία, για κάθε πλαίσιο φυσικής μνήμης
  - ➔ αριθμός πλαισίου → { διεργασία, σελίδα }
- ◆ Ακριβή αναζήτηση στον πίνακα, μετριάζεται με πίνακες κατακερματισμού
  - ➔ TLB → hash table → πίνακας πλαισίων
- ◆ Μοιραζόμενη μνήμη;

# Ανεστραμμένοι πίνακες σελίδων



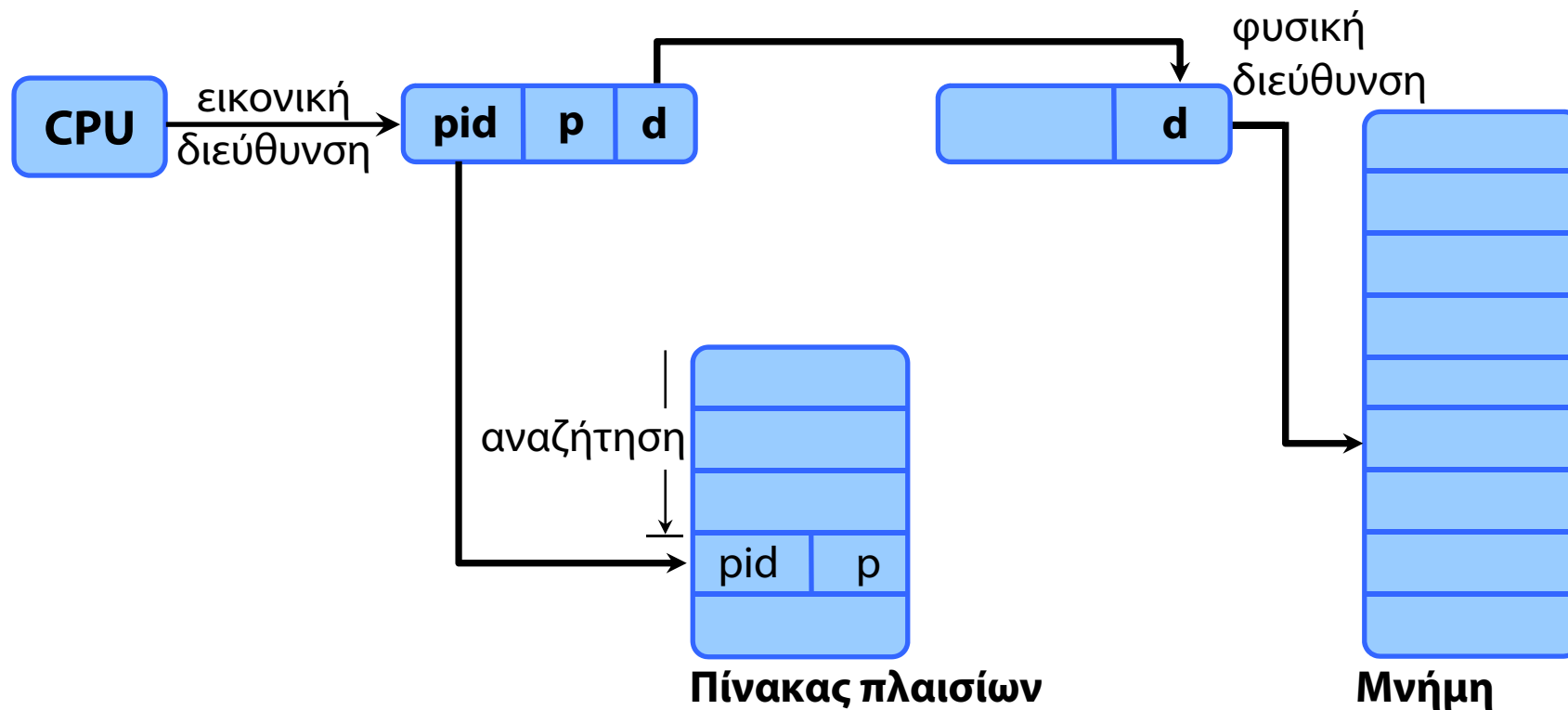
- ◆ Πίνακας πλαισίων αντί για πίνακας σελίδων
- ◆ Κρατάει την αντίστροφη πληροφορία, για κάθε πλαίσιο φυσικής μνήμης
  - ➔ αριθμός πλαισίου → { διεργασία, σελίδα }
- ◆ Ακριβή αναζήτηση στον πίνακα, μετριάζεται με πίνακες κατακερματισμού
  - ➔ TLB → hash table → πίνακας πλαισίων
- ◆ Μοιραζόμενη μνήμη;

# Ανεστραμμένοι πίνακες σελίδων



- ◆ Πίνακας πλαισίων αντί για πίνακας σελίδων
- ◆ Κρατάει την αντίστροφη πληροφορία, για κάθε πλαίσιο φυσικής μνήμης
  - ➔ αριθμός πλαισίου → { διεργασία, σελίδα }
- ◆ Ακριβή αναζήτηση στον πίνακα, μετριάζεται με πίνακες κατακερματισμού
  - ➔ TLB → hash table → πίνακας πλαισίων
- ◆ Μοιραζόμενη μνήμη;

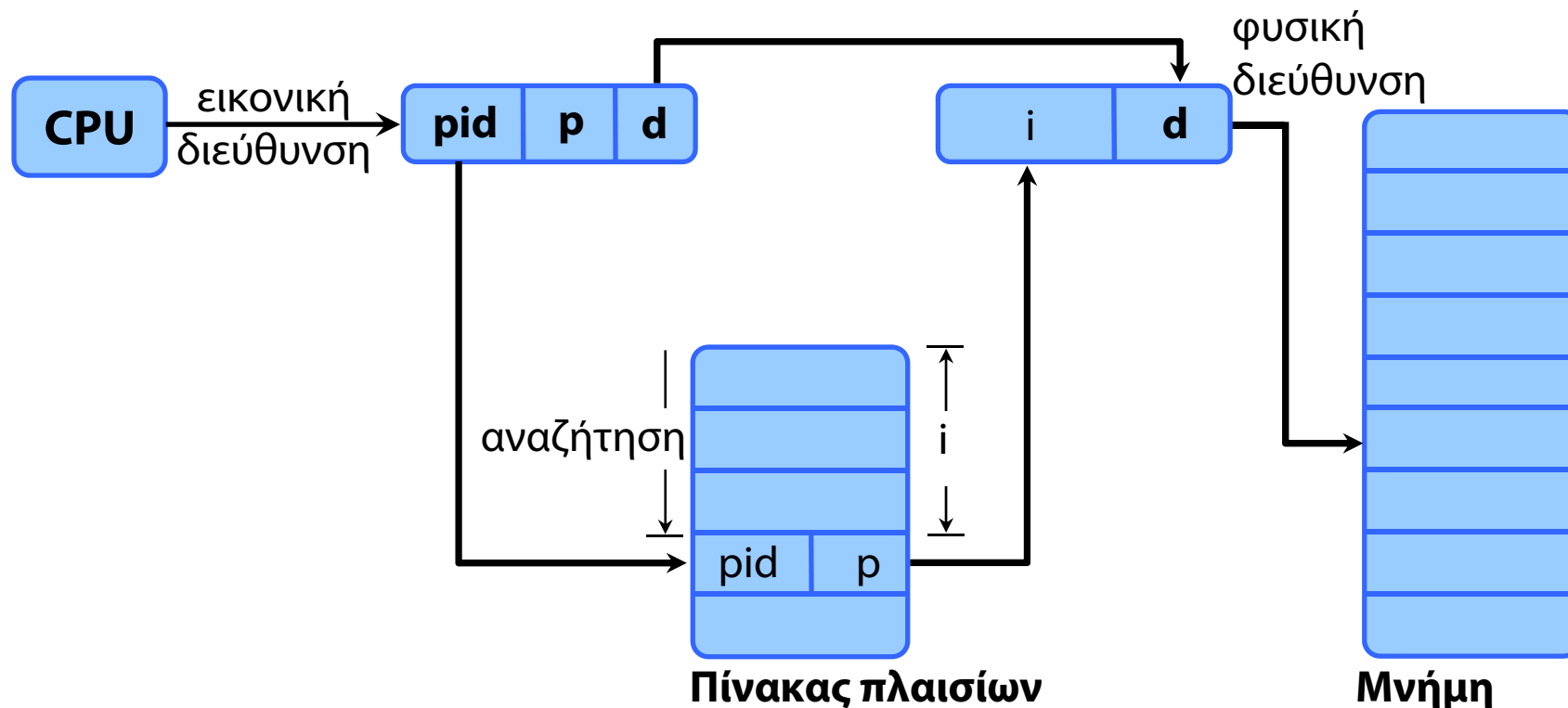
# Ανεστραμμένοι πίνακες σελίδων



- ◆ Πίνακας πλαισίων αντί για πίνακας σελίδων
- ◆ Κρατάει την αντίστροφη πληροφορία, για κάθε πλαίσιο φυσικής μνήμης
  - ➔ αριθμός πλαισίου → { διεργασία, σελίδα }
- ◆ Ακριβή αναζήτηση στον πίνακα, μετριάζεται με πίνακες κατακερματισμού
  - ➔ TLB → hash table → πίνακας πλαισίων
- ◆ Μοιραζόμενη μνήμη;



# Ανεστραμμένοι πίνακες σελίδων

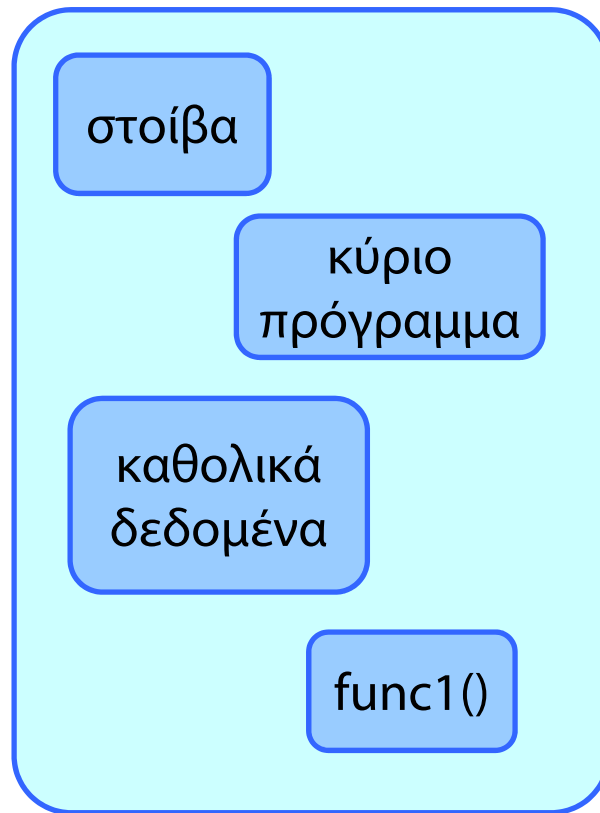


- ◆ Πίνακας πλαισίων αντί για πίνακας σελίδων
- ◆ Κρατάει την αντίστροφη πληροφορία, για κάθε πλαίσιο φυσικής μνήμης
  - αριθμός πλαισίου → { διεργασία, σελίδα }
- ◆ Ακριβή αναζήτηση στον πίνακα, μετριάζεται με πίνακες κατακερματισμού
  - TLB → hash table → πίνακας πλαισίων
- ◆ Μοιραζόμενη μνήμη;

# Διαχείριση Κύριας Μνήμης - Σύνοψη

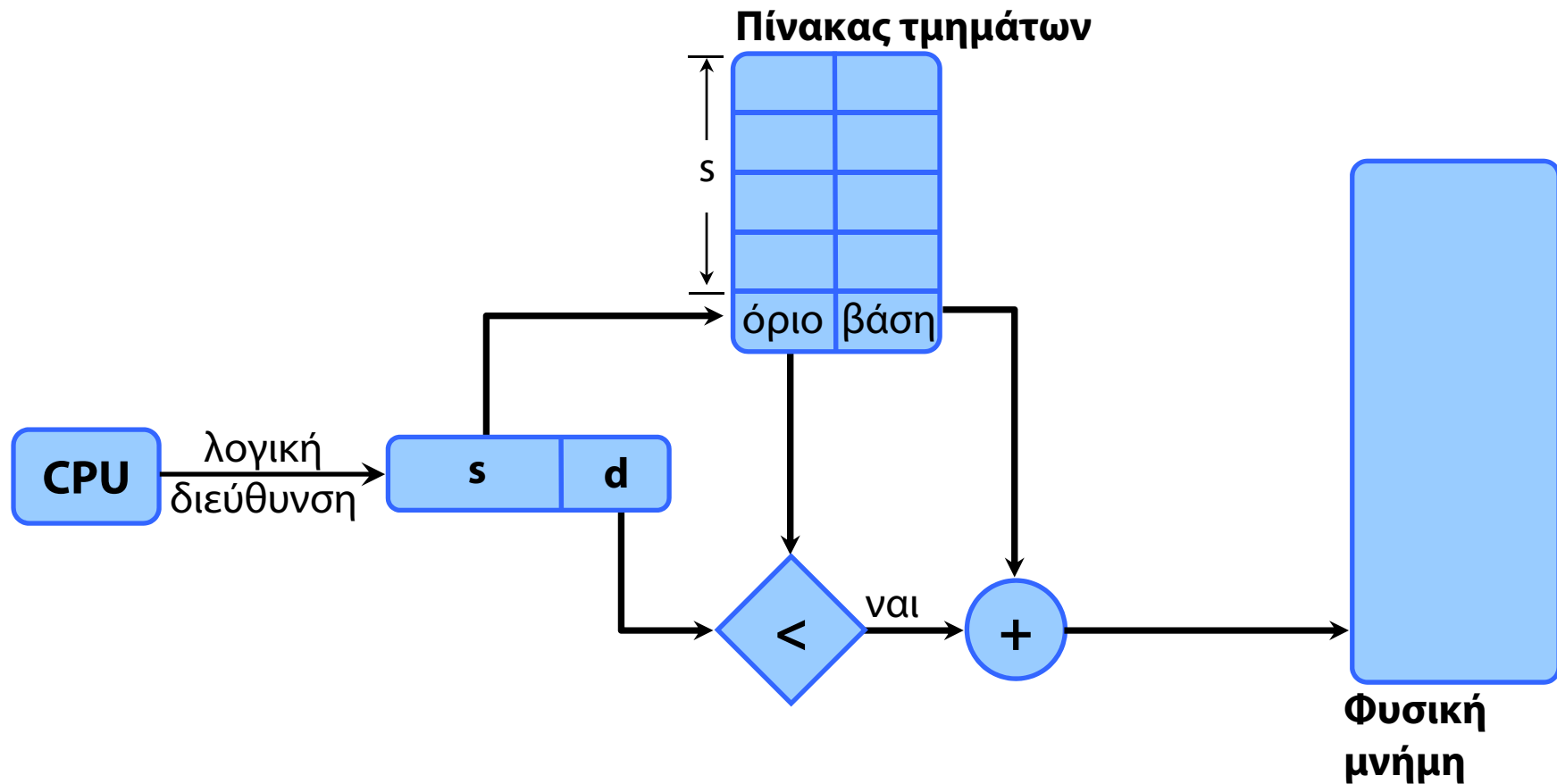
- ◆ Ιεραρχία μνήμης
- ◆ Μεταγλώττιση – φόρτωση – εκτέλεση κώδικα
- ◆ Καθορισμός διευθύνσεων
- ◆ Εναλλαγή διεργασιών
- ◆ Συνεχόμενη ανάθεση μνήμης
  - ➔ Στρατηγικές κατανομής, κατακερματισμός
- ◆ Σελιδοποίηση
  - ➔ Μετάφραση διευθύνσεων, πίνακες σελίδων, TLBs
  - ➔ Οργάνωση πινάκων σελίδων
- ◆ Κατάτμηση

# Κατάτμηση – Segmentation (1)



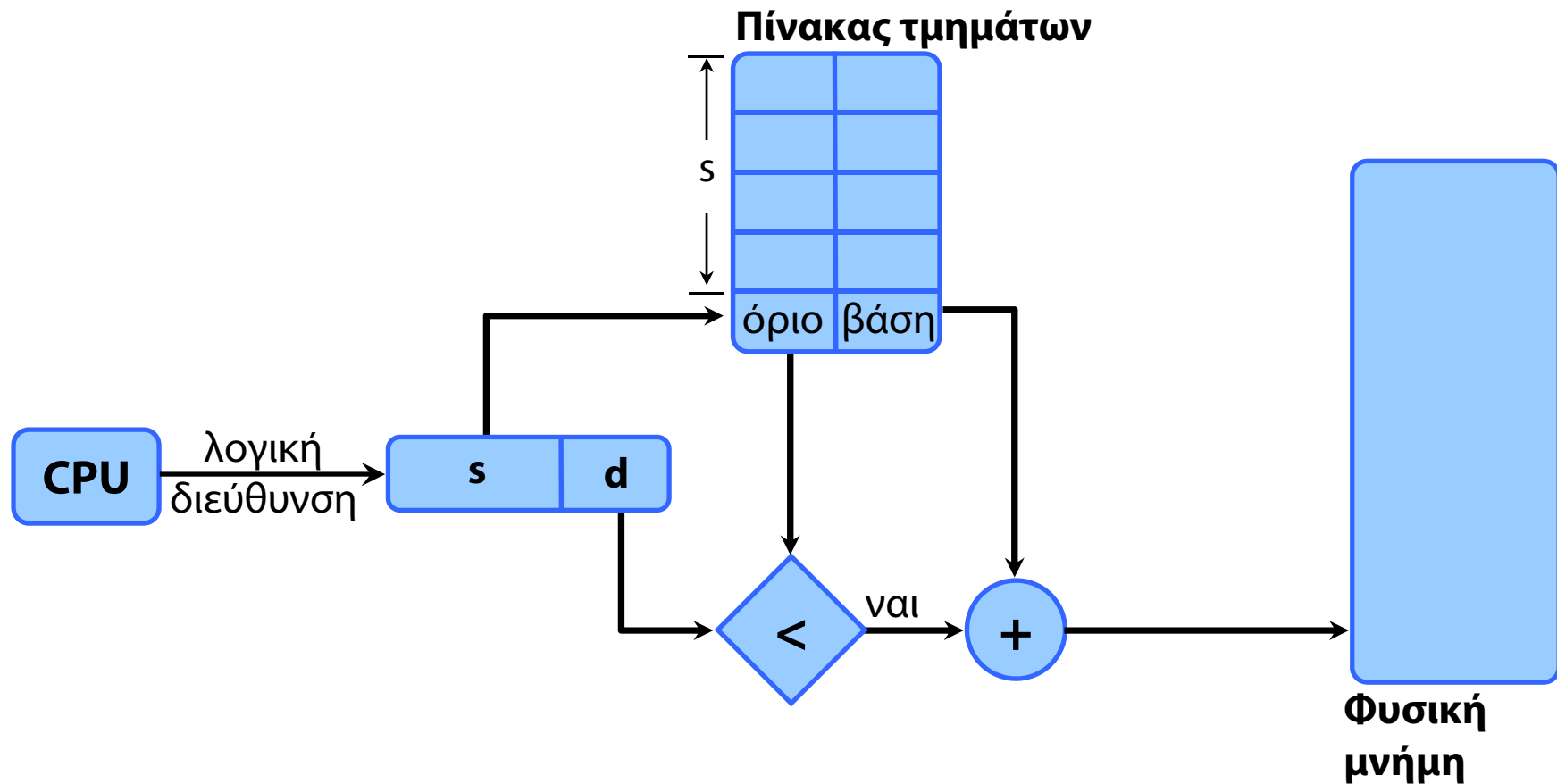
- ◆ Χωριστά, αριθμημένα, τμήματα μνήμης
- ◆ Κάθε λογική διεύθυνση είναι ένα ζεύγος { **s**, **d** }
- ◆ Το ΛΣ διαχειρίζεται τμήματα: { βάση τμήματος, όριο τμήματος }

## Κατάτμηση – Segmentation (2)



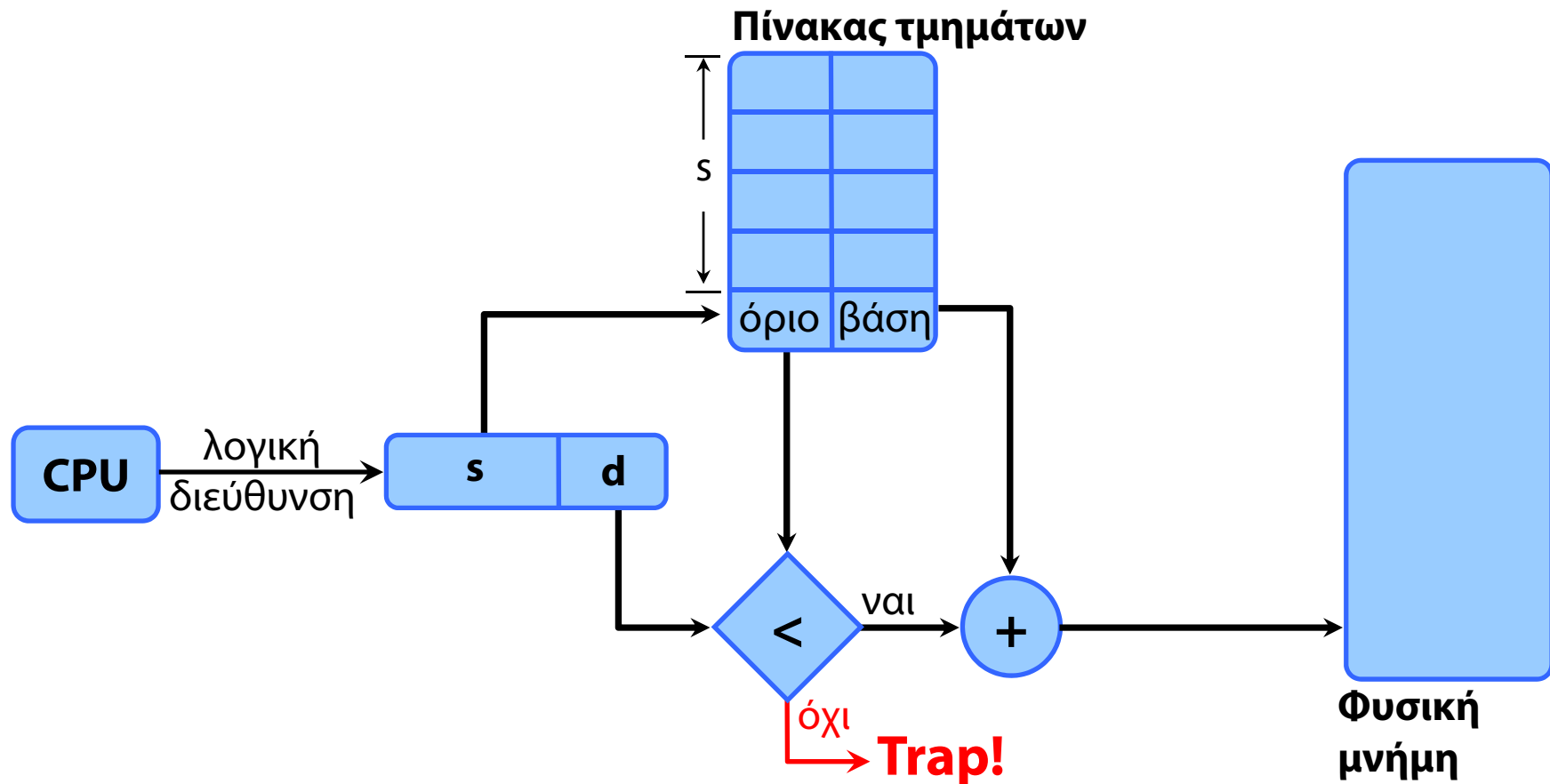
- ◆ Μπορεί να συνδυαστεί και με σελιδοποίηση (Intel x86)
  - ➔ στην πράξη (Linux σε i386) προτιμάται ένας ενιαίος, γραμμικός χώρος

## Κατάτμηση – Segmentation (2)



- ◆ Μπορεί να συνδυαστεί και με σελιδοποίηση (Intel x86)
  - ➔ στην πράξη (Linux σε i386) προτιμάται ένας ενιαίος, γραμμικός χώρος

## Κατάτμηση – Segmentation (2)



- ◆ Μπορεί να συνδυαστεί και με σελιδοποίηση (Intel x86)
  - ➔ στην πράξη (Linux σε i386) προτιμάται ένας ενιαίος, γραμμικός χώρος

Ερωτήσεις;



Ερωτήσεις;





Ερωτήσεις;



και στη λίστα:

**[OS@lists.cslab.ece.ntua.gr](mailto:OS@lists.cslab.ece.ntua.gr)**