

Ιόνιο Πανεπιστήμιο – Τμήμα Πληροφορικής
Αρχιτεκτονική Υπολογιστών
2015-16

Αρχιτεκτονικές Συνόλου Εντολών (II)

(Δομή Εντολών και Παραδείγματα)

<http://di.ionio.gr/~mistral/tp/comparch/>

Μ.Στεφανιδάκης



Αρχιτεκτονική Συνόλου Εντολών

- ISA

- Τι περιγράφει;
 - Διαθέσιμες πράξεις/λειτουργίες
 - Κωδικοποίηση λειτουργιών
 - Μορφή των δεδομένων εισόδου-εξόδου
 - Operands
 - Μέθοδοι προσπέλασης μνήμης
 - Προέλευση των δεδομένων
 - Χώροι προσωρινής αποθήκευσης
 - Καταχωρητές
 - Διακοπές και καταστάσεις σφάλματος
 - Ποια η “αντίδραση” του επεξεργαστή

Σχεδιασμός Συνόλου Εντολών

- ISA

- Συμβιβασμός μεταξύ:
 - Κόστους και απόδοσης υλικού
 - Βαθμού υποστήριξης λογισμικού
 - Άλλων παραγόντων όπως η κατανάλωση ενέργειας
 - Το υλικό καταναλώνει ενέργεια, υπό τον έλεγχο του λογισμικού όμως!
 - Μη επικάλυψης λειτουργιών εντολών (orthogonality)

Κωδικοποίηση Εντολών Μηχανής

- ISA



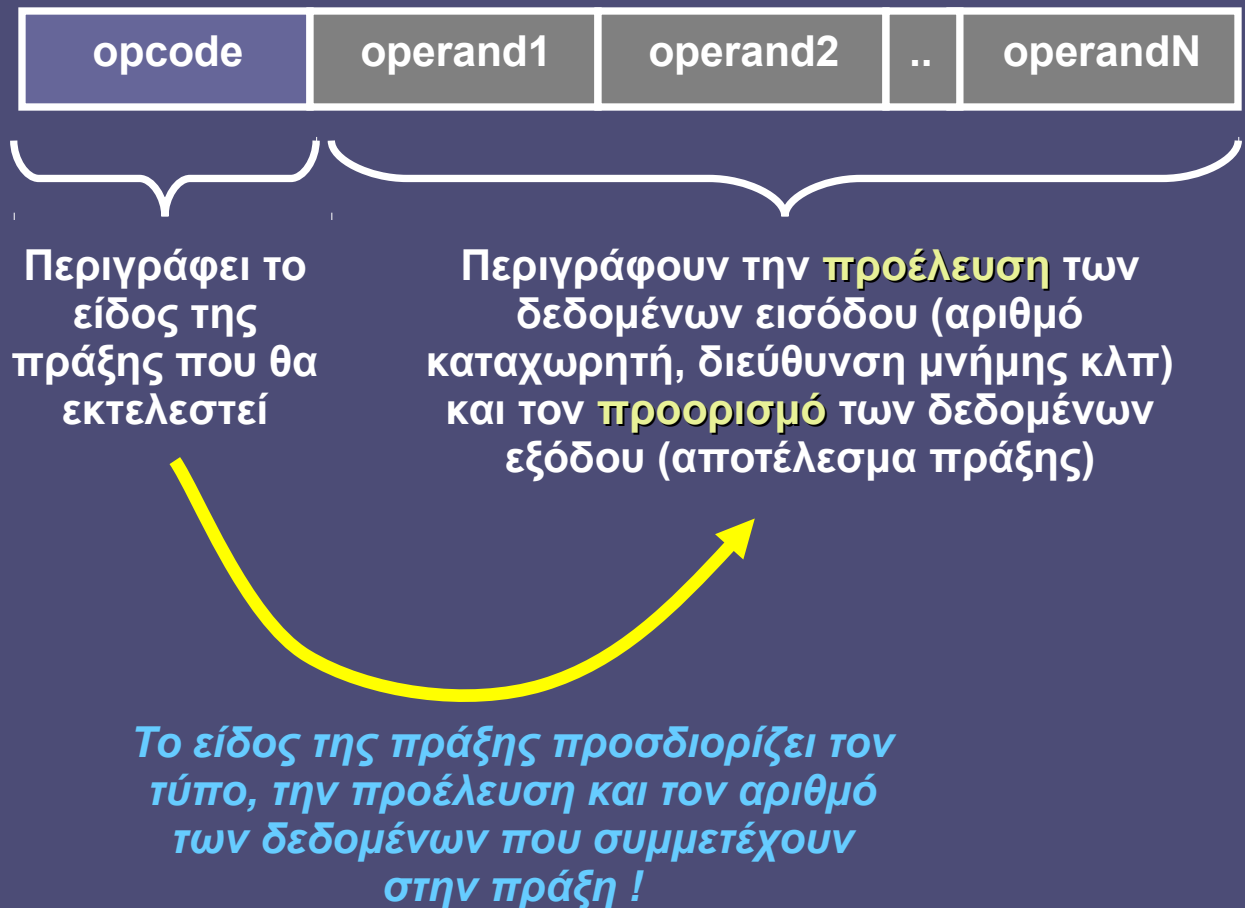
- Σειρά δυαδικών ψηφίων
 - Μεταβλητού μήκους
 - Περισσότερο συμπαγή προγράμματα
 - Πολυπλοκότερο υλικό!
 - Σταθερού μήκους
 - Απλούστερη και ταχύτερη λήψη-αποκωδικοποίηση
 - Μεγαλύτερα προγράμματα
 - Μέθοδοι συμπίεσης

;

Γιατί είναι
ταχύτερη η λήψη
και
αποκωδικοποίηση
των εντολών
σταθερού μήκους;

Κωδικοποίηση Εντολών Μηχανής

- ISA



Εντολές Μηχανής και Assembly

- ISA

- Εντολές μηχανής
 - Σειρές από bits (ομάδες bytes ανά εντολή)
 - Τα bits αυτά κωδικοποιούν **όλα τα χαρακτηριστικά** κάθε εντολής
- Assembly
 - **Μνημονικός** τρόπος αναπαράστασης εντολών
 - Ευκολότερη κατανόηση από τον άνθρωπο
 - Προσοχή:
 - MOV R1,R2 // π.χ. R1 => R2
 - MOV R1,#100 // π.χ. R1 => mem[100]
 - παρόλο που το μνημονικό MOV είναι ίδιο, δεν πρόκειται για την ίδια εντολή!



Κάθε διαφορετική
αρχιτεκτονική
επεξεργαστών έχει
διαφορετική
γλώσσα assembly!

Τα παραδείγματά μας: Intel IA-32 (64)

- ISA



Και η
ανταγωνίστρια
εταιρία AMD
παράγει
επεξεργαστές
συμβατούς με την
αρχιτεκτονική
αυτή

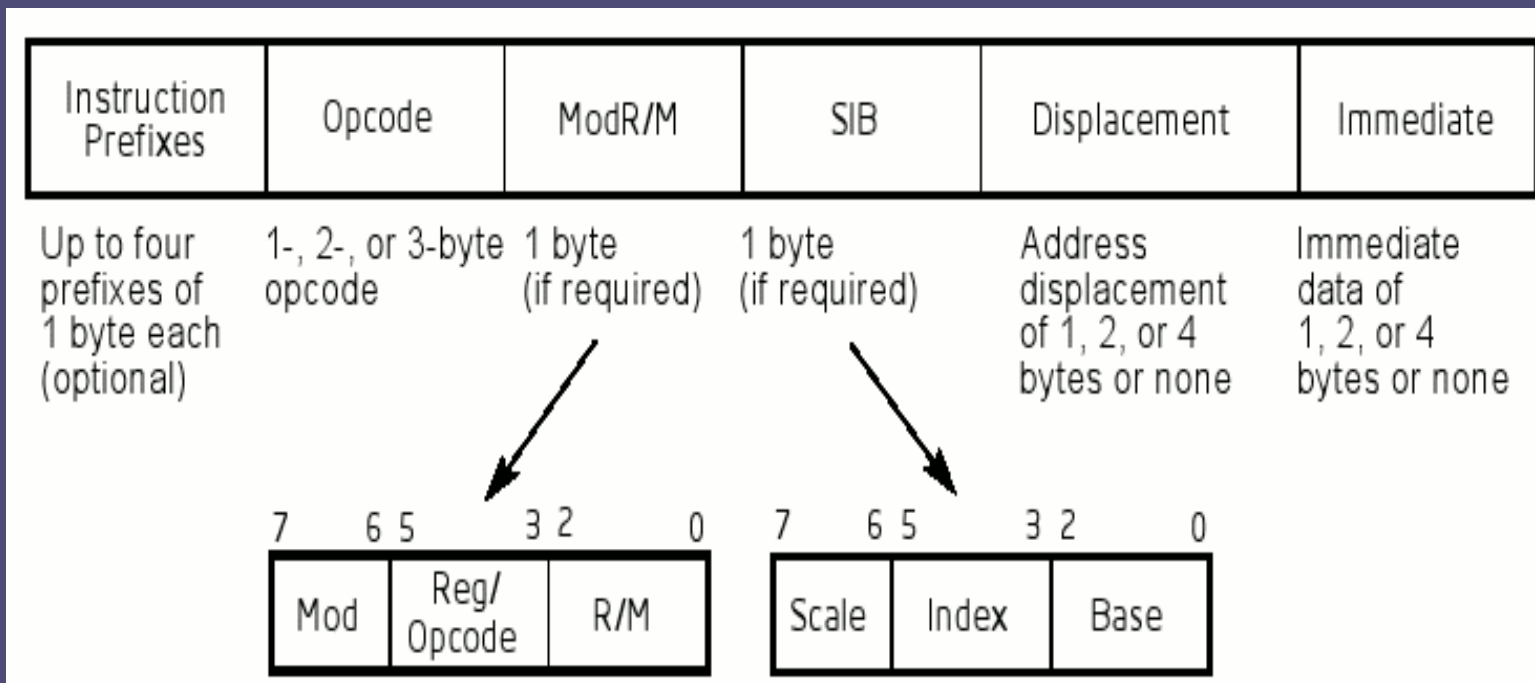
- Αρχιτεκτονική επεξεργαστών Intel IA-32
 - Μία πολύπλοκη αρχιτεκτονική επεξεργαστών
 - με σύνθετο σετ εντολών
 - διατήρηση συμβατότητας με παλαιότερους επεξεργαστές της Intel (legacy compatibility)
 - Με πολύ μεγάλη εγκατεστημένη βάση
 - Συστήματα PCs
 - Όχι η βέλτιστη αρχιτεκτονική αλλά:
 - Το μεγάλο μερίδιο που κατέχει στην αγορά επιτρέπει στην Intel έρευνα και ανάπτυξη για διαρκείς βελτιώσεις απόδοσης
 - “Τελικά η αρχιτεκτονική αυτή είναι βέλτιστη μέσα στην τεράστια πολυπλοκότητά της”

Τα παραδείγματά μας: Intel IA-32 (64)

- ISA
 - Οικογένειες επεξεργαστών
 - Pentium
 - Pentium Pro, II, III
 - P6
 - Pentium 4, D (IA-32 – σε νεώτερες εκδόσεις και 64)
 - Netburst
 - Pentium M
 - Intel Core, Core Duo
 - Intel Core (IA-32 -64)
 - Intel Core 2 Duo
 - ...νεώτερες αρχιτεκτονικές
- Intel® 64 and IA-32 Architectures Software Developer's Manuals
<http://www.intel.com/products/processor/manuals/index.htm>

Intel IA-32: η γενική μορφή των εντολών

- Μεταβλητός αριθμός bytes ανά εντολή (1-17)!



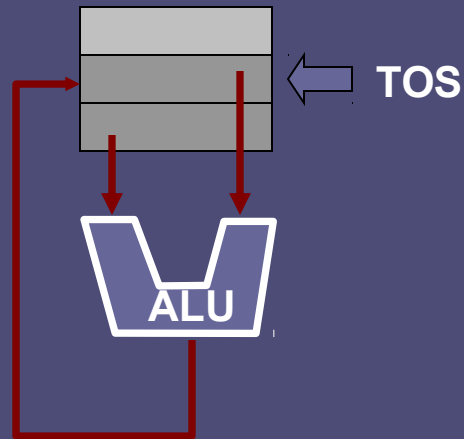
Προέλευση και αποθήκευση δεδομένων

- ISA
- Προέλευση δεδομένων

- Προέλευση δεδομένων – αποθήκευση αποτελεσμάτων μιας πράξης
 - **Operand addressing**
 - Εξαρτάται από την αρχιτεκτονική του επεξεργαστή
 - Στους πρώτους επεξεργαστές
 - Stack (σωρός-στοίβα)
 - Accumulator (συσσωρευτής)
 - Μεταγενέστεροι υπολογιστές
 - Καταχωρητές – Μνήμη (register-memory)
 - Κυρίως καταχωρητές (register-register ή load-store)
 - μόνο **ανάγνωση** και **εγγραφή** μνήμης

Αρχιτεκτονική σωρού (stack)

- ISA
- Προέλευση δεδομένων

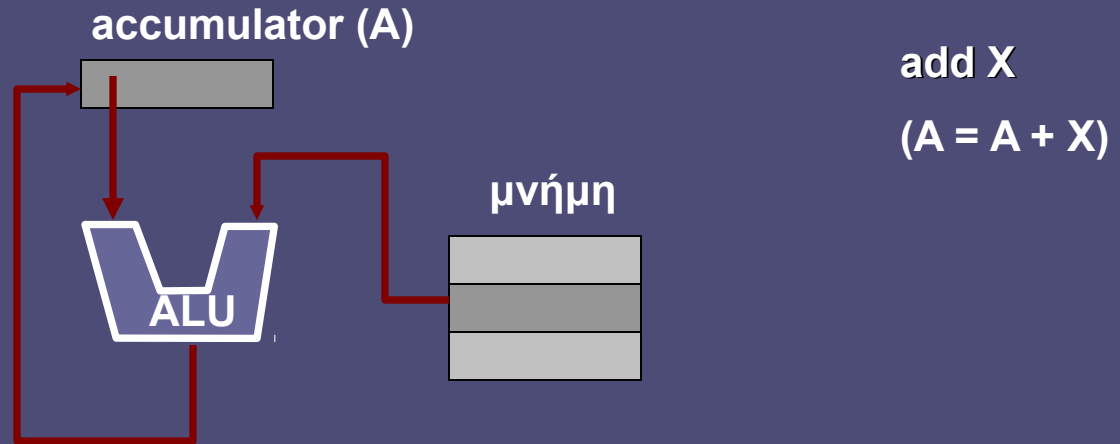


push A
push B
add
pop A

- Πηγή-προορισμός προσδιορίζονται έμμεσα
 - Δεν περιγράφονται στην εντολή!
 - 0-address architecture
 - Δημοφιλές σχήμα κατά τη δεκαετία του 60
 - Δύσκολη προσπέλαση σωρού, απαιτούνται πολλαπλές αντιμεταθέσεις και αντιγραφές
 - Το σχήμα επιζεί στην αρχιτεκτονική της Java VM

Αρχιτεκτονική συσσωρευτή (accumulator)

- ISA
- Προέλευση δεδομένων



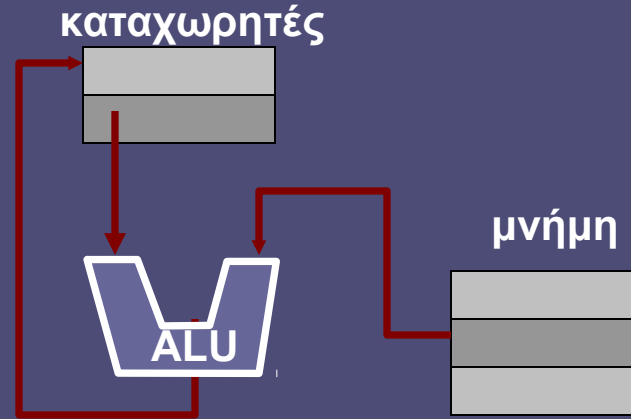
- Μία πηγή - θέση αποθήκευσης του αποτελέσματος είναι πάντα ο συσσωρευτής
 - 1-address architecture
 - Αρχιτεκτονική των πρώτων υπολογιστών!

Αρχιτεκτονικές με καταχωρητές (1)

- ISA
- Προέλευση δεδομένων



Καταχωρητές: προσωρινές θέσεις αποθήκευσης αποτελεσμάτων, η γενίκευση της ιδέας του συσσωρευτή.



add R2, R1, mem(100)
(R2 <= R1 + mem[100])

▪ Memory-register

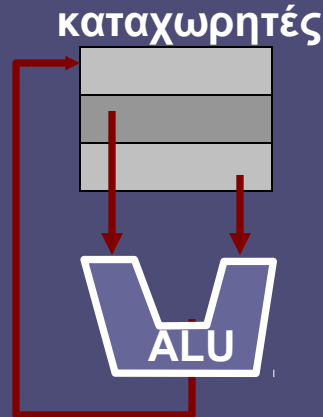
- Οποιαδήποτε εντολή μπορεί να προσπελάσει τη μνήμη

▪ Όμως:

- Πολλαπλές προσπελάσεις μνήμης
- Λήψη εντολής – Λήψη δεδομένων εντολής
- Πολύπλοκη εκτέλεση εντολής σε στάδια
- Συνωστισμός στον δίαυλο επικοινωνίας με μνήμη

Αρχιτεκτονικές με καταχωρητές (2)

- ISA
- Προέλευση δεδομένων



add R1, R2,R3
(R1 <= R2 + R3)

- **Register-register (load-store)**
 - Μόνο εντολές **load-store** μπορούν να προσπελάσουν τη μνήμη
- **Η αρχιτεκτονική των σύγχρονων επεξεργαστών**
 - Οι καταχωρητές προσπελαύνονται πολύ γρήγορα
 - Χρειάζονται λιγότερα bits για να επιλεγούν
 - Οι μεταγλωττιστές αναθέτουν μεταβλητές σε καταχωρητές

;

Είναι επιθυμητός
ένα πολύ μεγάλος
αριθμός
καταχωρητών;

Η εξέλιξη της αρχιτεκτονικής εντολών

- ISA
- Προέλευση δεδομένων

- Οι πρώτοι υπολογιστές (.. - '60)
 - Αρχιτεκτονική συσσωρευτή και αργότερα σωρού
 - Ικανοποιητική λύση λόγω της απλής τεχνολογίας των μεταγλωττιστών
- Πολύπλοκες αρχιτεκτονικές ('70 - ..)
 - Ενσωμάτωση σύνθετων μορφών εντολών και μεθόδων προσπέλασης μνήμης
 - Προσπάθεια υποστήριξης υψηλών γλωσσών προγραμματισμού – μείωσης κόστους λογισμικού
 - Πολλά χαρακτηριστικά μένουν αχρησιμοποίητα!
 - **Complex Instruction Set Computers (CISC)**

Η εξέλιξη της αρχιτεκτονικής εντολών

- ISA
- Προέλευση δεδομένων

- **Reduced Instruction Set Computers (RISC)**
(’80 - ...)
 - Απλούστερες και φθηνότερες load-store αρχιτεκτονικές με σταθερό μήκος εντολών
 - Μεγαλύτερη απόδοση – ταχύτερη εκτέλεση εντολών
 - Ευνοείται από την αφθονία υλικού χαμηλού κόστους και την προηγμένη τεχνολογία των μεταγλωττιστών
 - Ακόμα και ο μοναδικός επιζών επεξεργαστής με εντολές CISC (αρχιτεκτονική x86), μεταφράζει εσωτερικά σε εντολές RISC...

IA-32 (64): memory-register

- ISA
- Προέλευση δεδομένων

- Πολλές εντολές γενικού σκοπού
 - πέρα από τις load και store
 - έχουν ως πηγή ή προορισμό τη μνήμη

AND—Logical AND

Opcode	Instruction
...	
81 /4 iw	AND r/m16, imm16
81 /4 id	AND r/m32, imm32
...	
21 /r	AND r/m16, r16
21 /r	AND r/m32, r32
REX.W + 21 /r	AND r/m64, r64
22 /r	AND r8, r/m8
...	
23 /r	AND r32, r/m32
REX.W + 23 /r	AND r64, r/m64
...κλπ..	

IA-32: καταχωρητές

- ISA
- Προέλευση δεδομένων

i

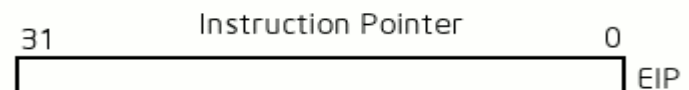
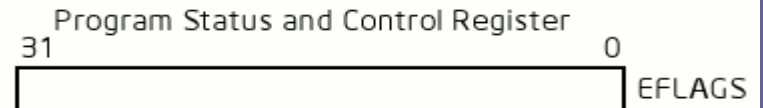
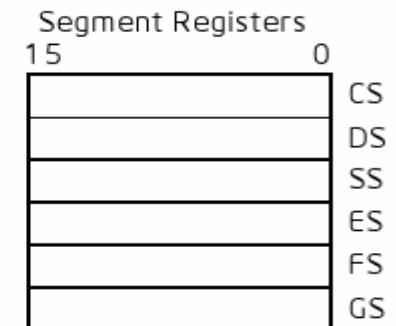
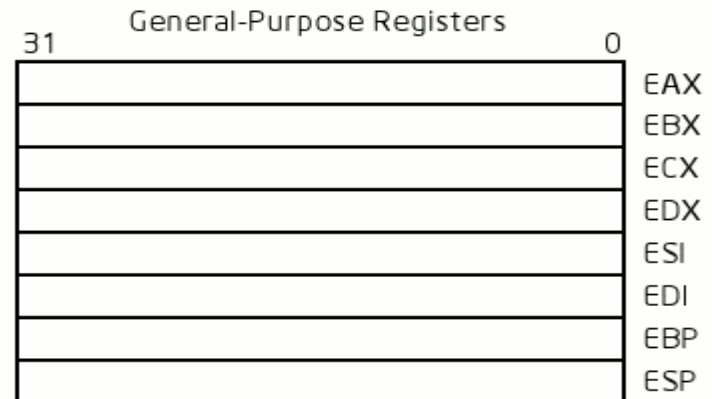
Εκτός από τους καταχωρητές γενικού σκοπού, οι επεξεργαστές IA-32 έχουν πρόσθετους καταχωρητές για πράξεις κινητής υποδιαστολής και πράξεις streaming και multimedia.

**γενικού σκοπού και
σχηματισμός
διευθύνσεων**

**τμηματοποίηση
(σε επόμενα
μαθηματα)**

**καταχωρητής
κατάστασης**

“program counter”



Μέθοδοι προσπέλασης μνήμης

- ISA
- Προέλευση δεδομένων
- Προσπέλαση μνήμης

- Τουλάχιστον κάποιες εντολές προσπελαίνουν τη μνήμη
 - για ανάγνωση ή εγγραφή δεδομένων
 - Πώς σχηματίζεται η διεύθυνση προσπέλασης;
 - Η γενική ιδέα: υποβοήθηση του λογισμικού
 - Διαφορετικός σχηματισμός διεύθυνσης για
 - Τοπικές μεταβλητές
 - Δείκτες (έμμεση προσπέλαση)
 - Στατικά δεδομένα
 - Διάσχιση πινάκων
 - (Σταθερές τιμές)
- Υποστήριξη ανάλογα με αρχιτεκτονική

Μέθοδοι προσπέλασης μνήμης

- ISA
- Προέλευση δεδομένων
- Προσπέλαση μνήμης

- Στο σχηματισμό της διεύθυνσης μνήμης μπορούν να συμμετέχουν:
 - Απόλυτες τιμές διεύθυνσης
 - Καταχωρητές
 - Σταθερές τιμές μετατόπισης (offsets)

πιθανή
χρήση



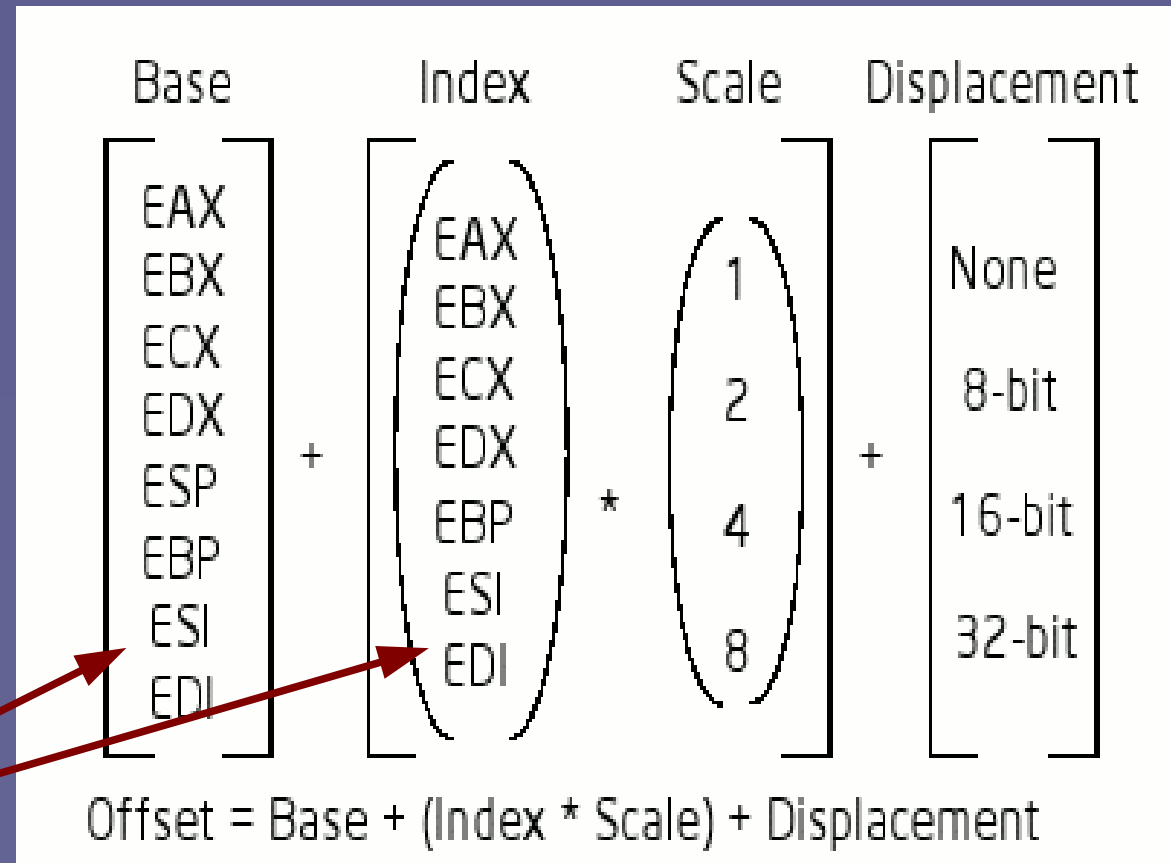
<i>displacement</i>	mem[offs+reg]	τοπικές
<i>register indirect</i>	mem[reg]	δείκτες
<i>indexed</i>	mem[reg1+reg2]	πίνακες
<i>direct</i>	mem[addr]	στατικές
<i>memory indirect</i>	mem[mem[reg]]	*δείκτες
<i>auto-increment</i>	mem[reg++]	πίνακες
<i>scaled</i>	mem[offs+reg1+reg2*d]	πίνακες

;

Πώς
κωδικοποιούνται
οι μέθοδοι
προσπέλασης
μνήμης μέσα στην
εντολή;

IA-32 (64): Σχηματισμός διευθύνσεων

- ISA
- Προέλευση δεδομένων
- Προσπέλαση μνήμης



καταχωρητές

Εντολές: κατηγορίες λειτουργιών

- ISA
- Προέλευση δεδομένων
- Προσπέλαση μνήμης
- Κατηγορίες εντολών

- Βασικές κατηγορίες
 - Αριθμητικές και λογικές πράξεις
 - Μεταφορά δεδομένων
 - Από-πρός Καταχωρητές και Μνήμη
 - Έλεγχος ροής εκτέλεσης
 - Διακλαδώσεις και κλήσεις ρουτινών
- Άλλες κατηγορίες
 - Ειδικές εντολές συστήματος
 - ΛΣ, ιδεατή μνήμη
 - Επεξεργασία πολλαπλών δεδομένων
 - Χρήσιμο για γραφικά, σειρές χαρακτήρων, multimedia

IA-32: εντολές μεταφοράς δεδομένων

Table 7-1. Move Instruction Operations

Type of Data Movement	Source →Destination
From memory to a register	Memory location →General-purpose register Memory location →Segment register
From a register to memory	General-purpose register →Memory location Segment register →Memory location
Between registers	General-purpose register →General-purpose register General-purpose register →Segment register Segment register →General-purpose register General-purpose register →Control register Control register →General-purpose register General-purpose register →Debug register Debug register →General-purpose register
Immediate data to a register	Immediate →General-purpose register
Immediate data to memory	Immediate →Memory location

ΙΑ-32: Εντολές διακλάδωσης

- ISA
- Προέλευση δεδομένων
- Προσπέλαση μνήμης
- Κατηγορίες εντολών

- Με ή χωρίς συνθήκη
 - Σε απόλυτη ή σχετική διεύθυνση
 - 16 ή 32 bits απόλυτη διεύθυνση
 - πιθανώς σε μνήμη ή καταχωρητή
 - 8, 16 ή 32 bits σχετική διεύθυνση
 - Συνθήκη: καταχωρητής κατάστασης EFLAGS
 - Μεταξύ τμημάτων κώδικα
 - της ίδιας ή διαφορετικής διεργασίας
- Κλήση συναρτήσεων
 - Όπως η διακλάδωση με αποθήκευση πρόσθετης πληροφορίας (program stack)
 - για επιστροφή από συνάρτηση στο σημείο μετά την κλήση

Εντολές διακλάδωσης

- ISA
- Προέλευση δεδομένων
- Προσπέλαση μνήμης
- Κατηγορίες εντολών

;

Πού θα μπορούσε να αποθηκευτεί η διεύθυνση επιστροφής μετά από κλήση συνάρτησης;

- Κλήση συναρτήσεων και επιστροφή
 - Αποθήκευση της τρέχουσας διεύθυνσης εκτέλεσης (καταχωρητή PC)
 - Πριν τη μετάβαση στη συνάρτηση
 - Χρήση αποθηκευμένης τιμής
 - Κατά την επιστροφή (return)

frame
pointer

