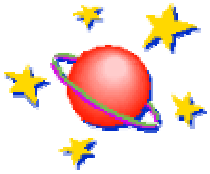
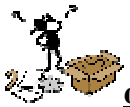


# Ενότητα 2η : *Λειτουργίες του Συνόλου Εντολών*

**Σκοπός** **Ο σκοπός της ενότητας αυτής είναι να παρουσιάσει τις εντολές που υποστηρίζονται από τους υπολογιστές CISC και RISC και τον τρόπο με τον οποίο αυτές εκτελούνται. Επίσης εξετάζονται οι διαφορές ανάμεσα σε αυτούς τους τύπους υπολογιστών και δίνονται παραδείγματα για την εκτέλεση των εντολών στους υπολογιστές VAX (CISC) και DLX (RISC).**

**Προσδοκώμενα Αποτελέσματα** **Όταν θα έχετε μελετήσει την ενότητα, θα είστε σε θέση να:**


**απαριθμείτε τις κατηγορίες των λειτουργιών που υποστηρίζουν οι υπολογιστές,**

**προσδιορίζετε το τρόπο με τον οποίο υποστηρίζουν τις λειτουργίες αυτές τα δύο είδη υπολογιστών RISC & CISC,**

**εξηγείτε τον τρόπο με τον οποίο οι εντολές μπορούν να προκαλέσουν αλλαγή στη ροή εκτέλεση του προγράμματος,**

**προσδιορίζετε τις διαφορές που υπάρχουν ανάμεσα στους υπολογιστές RISC & CISC όσον αφορά την εκτέλεση των εντολών που υποστηρίζουν,**

**εφαρμόζετε την εκτέλεση των εντολών του υπολογιστή DLX χρησιμοποιώντας κατάλληλο λογισμικό,**

**<u>εξηγείτε τον τρόπο με τον οποίο εκτελούνται οι εντολές των υπολογιστών DLX και VAX,</u>**

**PC σχετική διακλάδωση, ανεξαρτησία θέσης, PC - σχετική διευθυνσιοδότηση, καταχωρητής συνθήκης, κλήση διαδικασίας**

# Κατηγορίες εντολών

## ΔΡΑΣΤΗΡΙΟΤΗΤΑ 1

**Οι εντολές ενός υπολογιστή χρησιμοποιούν τελεστές και τελεστέους. Μπορείτε να εντοπίσετε τις διαφορές ανάμεσα στους δύο αυτούς όρους; Για περισσότερες λεπτομέρειες καλό θα ήταν να ανατρέξετε στην 1η ενότητα αυτού του κεφαλαίου.**

## ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 1

**Ο τελεστής αφορά στη λειτουργία που πρόκειται να εκτελέσει η εντολή και καθορίζεται από το πεδίο του κωδικού λειτουργίας της εντολής, ενώ ο τελεστέος αφορά στα δεδομένα που είναι απαραίτητα για την εκτέλεση της εντολής και μπορεί να είναι το περιεχόμενο ενός καταχωρητή ή μιας θέσης μνήμης, ή μία σταθερά.**

Οι τελεστές που υποστηρίζονται από τις περισσότερες αρχιτεκτονικές συνόλου εντολών είναι οι εξής:

| Τύπος εντολής | Επεξήγηση |
|---|---|
| **Αριθμητική και λογική** | Αριθμητικές και λογικές πράξεις μεταξύ ακεραίων όπως: πρόσθεση, αφαίρεση, λογικό ΚΑΙ, λογικό Ή |
| **Μεταφορά δεδομένων** | α)Μεταφορά από τον επεξεργαστή στην κύρια μνήμη ή αντιστρόφως<br>β)Μεταφορά από μία θέση κύριας μνήμης σε μία άλλη<br>γ)Μεταφορά δεδομένων στον επεξεργαστή |
| **Εντολή που αλλάζει τη ροή εκτέλεσης του προγράμματος** | Διακλάδωση με συνθήκη, μεταπήδηση, κλήση διαδικασίας και επιστροφή, παγίδες |
| **Συστήματος** | Κλήση λειτουργικού συστήματος, εντολές διαχείρισης εικονικής μνήμης |
| **Κινητής Υποδιαστολής** | Αριθμητικές πράξεις μεταξύ αριθμών κινητής υποδιαστολής όπως: πρόσθεση, πολ/σμός |
| **Δεκαδικός** | Αριθμητικές πράξεις μεταξύ δεκαδικών αριθμών: πρόσθεση, πολ/σμός, μετατροπές δεκαδικών σε χαρακτήρες |
| **Συμβολοσειρά** | Μετακίνηση / σύγκριση συμβολοσειρών, αναζήτηση σε συμβολοσειρές |

| | |
|---|---|
| **Γραφικά** | Πράξεις μεταξύ των pixels, πράξεις συμπίεσης/αποσυμπίεσης |

Όλοι οι υπολογιστές γενικά υποστηρίζουν ένα ολοκληρωμένο σύνολο εντολών για τις τρεις πρώτες κατηγορίες. Η υποστήριξη συναρτήσεων συστήματος ποικίλει από αρχιτεκτονική σε αρχιτεκτονική, αλλά όλοι οι υπολογιστές πρέπει να έχουν ορισμένες εντολές που να υποστηρίζουν βασικές συναρτήσεις συστήματος. Το πλήθος των εντολών που υποστηρίζουν τις τρεις τελευταίες κατηγορίες μπορεί να κυμαίνεται από μηδέν έως ένα εκτεταμένο σύνολο από ειδικές εντολές. Σε ορισμένους υπολογιστές ο μεταγλωττιστής συνθέτει δεκαδικές εντολές και εντολές για συμβολοσειρές χρησιμοποιώντας απλούστερες εντολές.

## Εντολές ελέγχου ροής εκτέλεσης του προγράμματος

Τέσσερις είναι οι διαφορετικοί τύποι εντολών αλλαγής ελέγχου ροής:

1. **Διακλάδωση με συνθήκη** - όταν η αλλαγή στη διεύθυνση της επόμενης εντολής γίνεται υπό κάποια συνθήκη.
2. **Μεταπήδηση** - όταν η αλλαγή στη διεύθυνση της επόμενης εντολής γίνεται χωρίς συνθήκη.
3. **Κλήση διαδικασίας**
4. **Επιστροφή από διαδικασία**

## ΔΡΑΣΤΗΡΙΟΤΗΤΑ 2

**Θυμάστε τον τρόπο με τον οποίο προσδιορίζεται η διεύθυνση της επόμενης εντολής που πρόκειται να εκτελεστεί; Να αναφέρετε ένα παράδειγμα αυτής της διαδικασίας, θεωρώντας πως κάθε θέση κύριας μνήμης καταλαμβάνει 4 bytes.**

## ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 2

**Η ΚΜΕ περιέχει έναν καταχωρητή ειδικής χρήσης που ονομάζεται απαριθμητής προγράμματος (program counter - PC), ο οποίος κρατάει τη διεύθυνση της επόμενης εντολής που πρόκειται να εκτελεστεί. Έστω ότι σε κάποια στιγμή το περιεχόμενο του PC είναι 250. Η εντολή που πρόκειται να εκτελεστεί είναι αυτή που βρίσκεται στη διεύθυνση μνήμης 250. Μετά την εκτέλεση της συγκεκριμένης εντολής το περιεχόμενο του PC αυξάνεται κατά 4. Επομένως η επόμενη εντολή που πρόκειται να εκτελεστεί είναι αυτή που βρίσκεται στη διεύθυνση 254.**

Οι εντολές ελέγχου προκαλούν αλλαγή στη ροή εκτέλεσης του προγράμματος. Σε αυτή την περίπτωση η διεύθυνση της επόμενης εντολής δεν προσδιορίζεται αυξάνοντας το περιεχόμενο του PC κατά 4 (στην περίπτωση που η εντολή καταλαμβάνει 4 bytes). Ωστόσο, η διεύθυνση στην οποία μετατίθεται η ροή εκτέλεσης του προγράμματος από μία εντολή ελέγχου, πρέπει πάντα να προσδιορίζεται (μοναδική εξαίρεση αποτελεί η επιστροφή από τη διαδικασία, αφού δεν είναι γνωστή κατά τη μεταγλώττιση). Σε μία εντολή ελέγχου, παρέχεται ένα πεδίο μετατόπισης (offset), το οποίο προστίθεται στον απαριθμητή προγράμματος και δίνει τη διεύθυνση της επόμενης εντολής. Επομένως η νέα τιμή του PC είναι PC+μετατόπιση. Τέτοιες εντολές ελέγχου ροής καλούνται **PC - σχετικές διακλαδώσεις**. Η χρήση της PC - σχετικής διευθυνσιοδότησης επιτρέπει στον κώδικα του προγράμματος να τρέχει, ανεξάρτητα από το που είναι φορτωμένος. Αυτή η ιδιότητα καλείται **ανεξαρτησία θέσης**.

Για να υλοποιήσουμε επιστροφές από διαδικασίες (return) και έμμεσες μεταπηδήσεις στις οποίες η διεύθυνση προορισμού δεν είναι γνωστή κατά τη διάρκεια της μεταγλώττισης, δεν μπορούμε να χρησιμοποιήσουμε την **PC - σχετική διευθυνσιοδότηση**. Θα πρέπει δηλαδή, να προσδιορίσουμε το στόχο δυναμικά, έτσι ώστε να μπορεί να αλλάξει κατά τη διάρκεια εκτέλεσης του προγράμματος. Σε αυτή την περίπτωση πρέπει να προσπελάσουμε έναν καταχωρητή, ο οποίος περιέχει τη διεύθυνση του στόχου.

Αυτές οι μεταπηδήσεις μέσω καταχωρητή είναι πολύ χρήσιμες στις εξής περιπτώσεις:

1. Στις εκφράσεις case ή switch που συναντώνται σε πολλές γλώσσες προγραμματισμού οι οποίες επιλέγουν μία από κάποιες συγκεκριμένες εναλλακτικές λύσεις που περιλαμβάνονται στην έκφραση.
2. Σε δυναμικά διαμοιραζόμενες βιβλιοθήκες (dynamically shared libraries), που επιτρέπουν σε μία βιβλιοθήκη να φορτωθεί μόνο όταν αυτή πραγματικά χρησιμοποιείται από ένα πρόγραμμα.
3. Σε εικονικές συναρτήσεις (virtual functions), σε αντικειμενοστραφείς γλώσσες όπως η C++, που επιτρέπουν σε διάφορες ρουτίνες να καλούνται με βάση τον τύπο των δεδομένων.

Οι πιο συχνά χρησιμοποιούμενες εντολές που χρησιμοποιούνται σε προγράμματα και προκαλούν αλλαγή στη ροή εκτέλεσης του προγράμματος είναι οι διακλαδώσεις με συνθήκη. Οι 2 κυριότερες τεχνικές που χρησιμοποιούνται για τον προσδιορισμό της συνθήκης διακλάδωσης είναι οι εξής:

| **Όνομα** | **Πώς γίνεται ο έλεγχος της συνθήκης** | **Πλεονεκτήματα** | **Μειονεκτήματα** |
|---|---|---|---|
| Καταχωρητής συνθήκης | Ελέγχεται το περιεχόμενο του καταχωρητή συνθήκης, στον οποίο έχει τοποθετηθεί το αποτέλεσμα της σύγκρισης. | Απλή | Χρησιμοποιεί κάθε φορά τον καταχωρητή |
| Σύγκριση και διακλάδωση | Η σύγκριση είναι μέρος της διακλάδωσης. Συχνά η σύγκριση περιορίζεται σε ένα υποσύνολο. | Μία εντολή για κάθε διακλάδωση, αντί για δύο. | Ίσως υπάρχει πάρα πολλή δουλειά για κάθε εντολή. |

**Καταχωρητής συνθήκης: Πολλοί υπολογιστές χρησιμοποιούν έναν καταχωρητή ειδικής χρήσης τα ψηφία του οποίου (flags) αλλάζουν κατά τη διάρκεια εκτέλεσης των εντολών ανάλογα με το είδος της εντολής και τα αποτελέσματά της. Για παράδειγμα το ψηφίο N (Negative) παίρνει την τιμή 1 αν το αποτέλεσμα μιας πράξης είναι αρνητικός αριθμός διαφορετικά, έχει την τιμή 0. Αντίστοιχα, το ψηφίο Z (Zero) παίρνει την τιμή 1 αν το αποτέλεσμα μιας πράξης είναι μηδέν. Με τον τρόπο αυτό, οι εντολές διακλάδωσης με συνθήκη έχουν τη μορφή BEQZ name (Branch if EQual) και ελέγχουν το αντίστοιχο ψηφίο του καταχωρητή συνθήκης για να αποφασίσουν αν θα πάνε στην επόμενη εντολή ή θα ακολουθήσουν τη διεύθυνση που ορίζει η διακλάδωση, δηλαδή το στόχο διακλάδωσης**.

# Λειτουργίες που υποστηρίζουν οι μηχανές RISC & CISC.

**CISC = Complex Instruction Set Computers**: Οι υπολογιστές αυτοί χαρακτηρίζονται από ένα σύνολο πολύπλοκων εντολών. Χαρακτηριστικό παράδειγμα τέτοιας μηχανής είναι ο VAX.
**RISC = Reduced Instruction Set Computers**: Οι υπολογιστές αυτοί χαρακτηρίζονται από ένα σύνολο εντολών με καθορισμένο μέγεθος. Χαρακτηριστικό παράδειγμα τέτοιας μηχανής είναι ο DLX. Όλοι οι υπολογιστές που δημιουργήθηκαν μετά το 1985, ανήκουν στην κατηγορία των RISC υπολογιστών.

**Στόχος των υπολογιστών CISC:** Οι ισχυροί τρόποι διευθυνσιοδότησης, οι ισχυρές εντολές και η αποδοτική κωδικοποίηση εντολών, οδήγησαν σε απλούς μεταγλωττιστές και πυκνότητα κώδικα.

**Στόχος των υπολογιστών RISC:** Οι απλές εντολές, οι απλοί τρόποι διευθυνσιοδότησης, το καθορισμένο μήκος εντολών και ο μεγάλος αριθμός καταχωρητών οδήγησαν σε υψηλή απόδοση χρησιμοποιώντας σωληνώσεις και σε απλή κατασκευή υλικού.

Οι λειτουργίες που υποστηρίζονται από τους περισσότερους υπολογιστές είναι:
1. Μεταφορά δεδομένων
2. Αριθμητικές και λογικές εντολές
3. Εντολές ελέγχου ροής εκτέλεσης του προγράμματος
4. Εντολές κινητής υποδιαστολής

## Μεταφορά δεδομένων

Οι εντολές μετακίνησης σε έναν υπολογιστή CISC μεταφέρουν δεδομένα μεταξύ δύο οποιονδήποτε διευθυνσιοδοτημένων περιοχών, ενώ σε έναν υπολογιστή RISC η μεταφορά δεδομένων γίνεται μεταξύ καταχωρητή και κύριας μνήμης.

### ΔΡΑΣΤΗΡΙΟΤΗΤΑ 3

**Μεταξύ ποιων περιοχών μεταφέρονται τα δεδομένα στις εντολές μετακίνησης σε έναν υπολογιστή CISC;Στην περίπτωση που δυσκολεύεστε να απαντήσετε, κρίνεται σκόπιμο να ανατρέξετε στην 1η υποενότητα «Κατηγορίες εντολών».**

## ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 3

**Οι εντολές μετακίνησης σε έναν υπολογιστή CISC μεταφέρουν δεδομένα μεταξύ:**
**α)του επεξεργαστή και της κύριας μνήμη,**
**β)θέσεων της κύριας μνήμης,**
**γ)περιοχών του επεξεργαστή.**

**Στους υπολογιστές CISC (VAX) μπορεί να γίνει μετακίνηση δεδομένων από μία διεύθυνση μνήμης σε μια άλλη χρησιμοποιώντας μόνο μία εντολή. Αντιθέτως, στους υπολογιστές RISC (DLX) χρειάζεται μια σειρά εντολών για τη μετακίνηση αυτή.**

**Εξαίρεση των υπολογιστών CISC όσον αφορά στη μεταφορά δεδομένων, αποτελεί μια ειδική εντολή μετακίνησης MOVA(Move Address) στον VAX, η οποία μετακινεί την 32-bit διεύθυνση του τελεστέου αντί των δεδομένων.**

Παρακάτω παρουσιάζουμε δύο πίνακες με εντολές μεταφοράς δεδομένων σε έναν υπολογιστή DLX και σε έναν υπολογιστή VAX.

Για τον **DLX**:

| **Τύπος εντολής/ κωδικός πράξης** | **Σημασία εντολής** |
|---|---|
| **Μεταφορά δεδομένων** | **Μετακίνηση δεδομένων μεταξύ καταχωρητή και μνήμης, ή μεταξύ του ακεραίων και FP ή ειδικών καταχωρητών.** |
| 1. LB, LW, LH | 1. Φόρτωση byte (Load Byte), φόρτωση λέξης (Load Word), φόρτωση μισής λέξης (Load Half word) |
| 2. LBU, LHU | 2. Φόρτωση byte χωρίς πρόσημο (Load Byte Unsigned), φόρτωση μισής λέξης χωρίς πρόσημο (Load Half word Unsigned) |
| 3. LF, LD | 3. Φόρτωση SP κινητής υποδιαστολής, φόρτωση DP κινητής υποδιαστολής |
| 4. SB, SW, SH | 4. Αποθήκευση byte (Store Byte), αποθήκευση λέξης (Store Word), αποθήκευση μισής λέξης (Store Half word) |
| 5. SF, SD | 5. Αποθήκευση SP κινητής υποδιαστολής, |

| | |
|---|---|
| 6. MOVI2S, MOVS2I<br>7. MOVF, MOVD<br>8. MOVFP2I, MOVI2FP | αποθήκευση DP κινητής υποδιαστολής<br>6. Μετακίνηση από /σε GPR* σε /από έναν καταχωρητή ειδικής χρήσης<br>7. Αντιγραφή ενός FP καταχωρητή ή ενός ζεύγους GP σε άλλον καταχωρητή ή ζεύγος<br>8. Μετακίνηση 32 bits από/σε FP καταχωρητές σε/από ακέραιους καταχωρητές |

## Παράδειγμα

Για την επεξήγηση της λειτουργίας που εκτελεί κάθε εντολή χρησιμοποιούνται συμβολισμοί. Στα παραδείγματα που ακολουθούν, οι συμβολισμοί ερμηνεύονται ως εξής:

- Το αριστερό βέλος με δείκτη n σημαίνει μεταφορά ποσότητας n-bit.
- Τα bit έχουν αριθμηθεί από το 0 έως το 31 με το 0 να έχει ανατεθεί στο πιο σημαντικό bit. Ο δείκτης 0 συμβολίζει το bit προσήμου, ενώ ο δείκτης 24..31 συμβολίζει τα 8 λιγότερο σημαντικά bit.
- Το σύμβολο Mem αντιπροσωπεύει την κύρια μνήμη, η θέση της οποίας καθορίζεται από ένα byte διευθύνσεως και μπορεί να μεταφέρει οποιονδήποτε αριθμό bytes.
- Ένα πεδίο παριστάνεται επίσης και με δύναμη. (π.χ το $0^{24}$ αναφέρεται σε ένα πεδίο από μηδενικά μήκους 24 bits).
- Το σύμβολο ## χρησιμοποιείται για τη σύνδεση δύο πεδίων και μπορεί να εμφανιστεί σε οποιαδήποτε πλευρά της πληροφορίας που μεταφέρεται.
- Το σύμβολο Regs αντιπροσωπεύει τον καταχωρητή.

**LB R1, 200(R2): Regs[R1] ← $_{32}$ (Mem[200+Regs[R2]]$_0$)$^{24}$ ## Mem[200+Regs[R2]]**

**LW R1, 100(R2): Regs[R1] ← $_{32}$ Mem[100+Regs[R2]]**

**LH R1, 50(R2): Regs[R1] ← $_{32}$ (Mem[50+Regs[R2]]$_0$)$^{24}$ ## Mem[50+Regs[R2]] ## Mem[51+Regs[R2]]**

**LBU R1, 20(R2): Regs[R1] ← $_{32}$ $0^{24}$ ## Mem[20+Regs[R2]]**

**LF F0, 150 (R3): Regs[F0] ← $_{32}$ Mem[150+Regs[R3]]**

**LD F0, 120 (R3): Regs[F0] ## Regs[R3] ← $_{64}$ Mem[120+Regs[R3]]**

**SB 400(R1), R2: Mem[400+Regs[R1]] ← $_{16}$ Regs[R2]$_{16...31}$**

* General Purpose Register: Καταχωρητής γενικής χρήσης.

**SW 180(R1), R2: Mem[180+Regs[R1]] ← $_{32}$ Regs[R2]**

**SH 12(R1), R2: Mem[12+Regs[R1]] ← $_{16}$ Regs[R2]$_{16...31}$**

**SF 24(R1), F0: Mem[24+Regs[R1]] ← $_{32}$ Regs[F0]**

**SD 40(R1), F0: Mem[40+Regs[R1]] ← $_{32}$ Regs[F0]**
**Mem[44+Regs[R1]] ← $_{16}$ Regs[F1]**

## ΔΡΑΣΤΗΡΙΟΤΗΤΑ 4

**Να εξηγήσετε αναλυτικά τον τρόπο με τον οποίο εκτελείται κάθε μία από τις εντολές του παραδείγματος. Προσπαθήστε να απαντήσετε στην ερώτηση χωρίς να κοιτάτε την επεξήγηση με τους συμβολισμούς.**

Για τον **VAX**:

| **Τύπος εντολής/ κωδικός πράξης** | **Σημασία εντολής** |
|---|---|
| **Μεταφορά δεδομένων** | **Μετακίνηση δεδομένων (* είναι ο τύπος των δεδομένων)** |
| 1. MOV* | 1. Μετακίνηση μεταξύ δύο τελεστέων (MOVE) |
| 2. MOVZB* | 2. Μετακίνηση ενός byte σε μισή λέξη ή λέξη, επεκτείνοντάς το με μηδενικά |
| 3. MOVA* | 3. Μετακίνηση της 32-bit διεύθυνσης ενός τελεστέου |
| 4. PUSH* | 4. Ώθηση τελεστέου στην στοίβα |

## ΔΡΑΣΤΗΡΙΟΤΗΤΑ 5

**Ποια είναι η λειτουργία της παρακάτω εντολής του VAX:**

**MOVB 150(R1), (R2)**

**αν [R1] = 50 και [R2]=300;**
**(Το B στην εντολή σημαίνει πως το δεδομένο που θα μετακινηθεί είναι μεγέθους 1 byte.)**

## ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 5

**Με την εντολή αυτή μετακινούνται τα δεδομένα που βρίσκονται στη θέση μνήμης 200 (η διεύθυνση υπολογίζεται προσθέτοντας το περιεχόμενο του καταχωρητή R1 που είναι το 50 με την τιμή 150), στη θέση μνήμης με διεύθυνση 300 (το περιεχόμενο του καταχωρητή R2).**

## Παράδειγμα

**Έστω ότι θέλουμε να μετακινήσουμε την τιμή 53 στον καταχωρητή R2.**
**α) Με ποιον τρόπο μπορεί να επιτευχθεί αυτό, χρησιμοποιώντας τον υπολογιστή VAX;**
**β) Με ποιον τρόπο μπορεί να επιτευχθεί αυτό, χρησιμοποιώντας τον υπολογιστή DLX;**

### Απάντηση:

α) Η εντολή που χρησιμοποιούμε στον VAX είναι:

**MOVB I↑ #53, R2**

Το σύμβολο I↑ χρησιμοποιείται για να δηλώσει το μέγεθος του απευθείας τελεστέου. Στη θέση του γράμματος I μπορεί να υπάρχει ένα από τα παρακάτω γράμματα:

B : τελεστέος μεγέθους 1 byte
W : τελεστέος μεγέθους 2 bytes
L : τελεστέος μεγέθους 4 bytes

β) Η εντολή που χρησιμοποιούμε στον DLX είναι:

**LB R2, #53**

Παρατηρούμε ότι στον υπολογιστή VAX ο τελεστέος προορισμού βρίσκεται στο τέλος, ενώ στον υπολογιστή DLX βρίσκεται αμέσως μετά το όνομα της εντολής.
Μετά την εκτέλεση της εντολής ο καταχωρητής R2 περιέχει την τιμή 53.

## ΔΡΑΣΤΗΡΙΟΤΗΤΑ 6

**Έστω η παρακάτω εντολή φόρτωσης του DLX: LW R1,50(R2).**

- **Αν το περιεχόμενο του καταχωρητή R2 είναι 100, ποιο θα είναι το αποτέλεσμα μετά την εκτέλεση της εντολής;**
- **Να εκτελέσετε την παραπάνω εντολή, χρησιμοποιώντας το λογισμικό που βρίσκεται στην ιστοσελίδα:**
  **http://hermes.di.uoa.gr/DLXSim/instruction/instr.html**
- **Να συγκρίνετε το αποτέλεσμα με την απάντηση που δώσατε.**

## ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 6

**Με την εντολή αυτή φορτώνεται στον καταχωρητή R1, το περιεχόμενο της διεύθυνσης μνήμης, η οποία υπολογίζεται προσθέτοντας την τιμή 50 στο περιεχόμενο του καταχωρητή R2. Επομένως γίνεται η πρόσθεση της τιμής 50 και του περιεχομένου του καταχωρητή R2 (100) και το αποτέλεσμα (150) φορτώνεται στον καταχωρητή R1.**

# Αριθμητικές και λογικές εντολές

Οι αριθμητικές και λογικές λειτουργίες των υπολογιστών CISC διαφέρουν σε δύο σημεία από αυτές των υπολογιστών RISC:

1. Ο τύπος δεδομένων στις πρώτες βρίσκεται αμέσως μετά το όνομα της εντολής, δηλαδή δε μεσολαβεί κενό ανάμεσά τους. Έτσι οι εντολές ADDB, ADDW και ADDL εκτελούνται πάνω σε 8-bit, 16-bit και 32-bit (αντίστοιχα) δεδομένα μνήμης ή καταχωρητών. Ωστόσο στους υπολογιστές RISC, μια εντολή ADD λειτουργεί σε ολόκληρο τον 32-bit καταχωρητή.
2. Με στόχο την μείωση του μεγέθους του κώδικα, στους υπολογιστές CISC η εντολή ADD προσδιορίζει τον αριθμό των τελεστέων, ενώ στους υπολογιστές RISC το πλήθος των τελεστέων που προσδιορίζονται είναι πάντα τρία.

## Παράδειγμα

**Σε έναν υπολογιστή MIPS (υπολογιστής RISC) η εντολή ADD $1,$1,$2 έχει μέγεθος 32 bit, ενώ η ίδια εντολή στον υπολογιστή VAX έχει την μορφή:**

**ADD L2 R1,R2 και έχει μέγεθος 24 bits: 8 bits για τον κωδικό λειτουργίας (opcode) και 8 bits για τον προσδιορισμό του κάθε τελεστέου.**

Παρακάτω παρουσιάζουμε δύο πίνακες με αριθμητικές και λογικές λειτουργίες ενός υπολογιστή DLX και ενός υπολογιστή VAX.

Για τον **DLX**:

| **Τύπος εντολής/ κωδικός πράξης** | **Σημασία εντολής** |
|---|---|
| **Αριθμητικές/λογικές** | **Πράξεις σε ακέραια ή λογικά δεδομένα σε GPRs.** |
| 1. ADD, ADDI, ADDU, ADDUI | 1. Πρόσθεση, απευθείας πρόσθεση (όλα τα απευθείας (Immediate-I) είναι 16 bits) προσημασμένη και μη. |
| 2. SUB, SUBI, SUBU, SUBUI | 2. Αφαίρεση, απευθείας αφαίρεση προσημασμένη και μη. |
| 3. MULT, MULTU, DIV, DIVU | 3. Πολλαπλασιασμός και διαίρεση, προσημασμένα και μη. Οι τελεστέοι πρέπει να είναι FP καταχωρητές. Όλες οι πράξεις παίρνουν και δίνουν τιμές 32-bit. |
| 4. AND, ANDI | 4. And, απευθείας and. |
| 5. OR, ORI, XOR, XORI | 5. Or, απευθείας or, αποκλειστικό or, απευθείας αποκλειστικό or. |
| 6. LHI | 6. Απευθείας φόρτωση του high-φορτώνει το πάνω μισό ενός καταχωρητή απευθείας. |
| 7. SLL, SRL, SRA, SLLI, SRLI, SRAI | 7. Ολισθήσεις: απευθείας (S_I) και μεταβλητή (S_) μορφή. Οι ολισθήσεις είναι αριστερές λογικές, δεξιές λογικές, δεξιές αριθμητικές. |
| 8. S_, S_I | 8. Συγκρίσεις: η συνθήκη «_» μπορεί να είναι LT (Less Than), GT (Greater Than), LE (Less or Equal), GE (Greater or Equal), EQ (EQual), NE (Not Equal). |

## Παράδειγμα

**ADD R1, R2, R3: Regs[R1] ← Regs[R2] + Regs[R3]**

**ADDI R1, R2, #20: Regs[R1] ← Regs[R2] + 20**

**LHI R1, #100: Regs[R1] ← Regs100 ## $0^{16}$**

**SLLI R1, R2, #44: Regs[R1] ← Regs[R2] << 44**

**SLT R1, R2, R3: if (Regs[R2] < Regs[R3]) Regs[R1] ← 1**
**else Regs[R1] ← 0**

## Παράδειγμα

**Η εντολή *ADDI R1,R2,#3* του DLX προσθέτει το περιεχόμενο του καταχωρητή R2 με την τιμή 3 και τοποθετεί το αποτέλεσμα στον καταχωρητή R1. Το γράμμα I δίπλα στο όνομα της εντολής δηλώνει ότι στην πράξη υπάρχει απευθείας τελεστέος(Immediate).**

## ΔΡΑΣΤΗΡΙΟΤΗΤΑ 7

- **Να περιγράψετε την εκτέλεση της εντολής του DLX: ADDI R1, R2, #28.**
- **Ποιο θα είναι το αποτέλεσμα αν το περιεχόμενο του καταχωρητή R2 είναι 122;**
- **Να εκτελέσετε την παραπάνω εντολή, χρησιμοποιώντας το λογισμικό που βρίσκεται στην ιστοσελίδα:**
  **http://hermes.di.uoa.gr/DLXSim/instruction/instr.html**
- **Να συγκρίνετε το αποτέλεσμα με την απάντηση που δώσατε.**

## ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 7

- **Η εντολή ADDI R1, R2, #28 προσθέτει το περιεχόμενο του καταχωρητή R2 με την τιμή 28 και τοποθετεί το αποτέλεσμα στον καταχωρητή R1.**
- **Το αποτέλεσμα θα είναι το 150 και θα εγγραφεί στον καταχωρητή R1.**

Για τον **VAX:**

| **Τύπος εντολής/ κωδικός πράξης** | **Σημασία εντολής** |
|---|---|
| **Αριθμητικές /λογικές** | **Πράξεις σε ακέραια ή λογικά δεδομένα (το * είναι ο τύπος δεδομένων)** |
| 1. ADD*<br>2. CMP* | 1. Πρόσθεση με 2 ή 3 τελεστέους<br>2. Σύγκριση και καθορισμός του κώδικα συνθήκης |

| | |
|---|---|
| 3. TST* | 3. Σύγκριση με το 0 και καθορισμός του κώδικα συνθήκης |
| 4. ASH* | 4. Αριθμητική ολίσθηση |
| 5. CLR* | 5. Καθαρισμός |
| 6. CVTB* | 6. Προέκταση προσήμου byte στο μέγεθος του τύπου δεδομένων |

**Παράδειγμα**

**Έστω η εντολή του υπολογιστή VAX:**

**ADDW3 LOC1,LOC2,R2**

**Η εντολή αυτή προσθέτει τα δεδομένα (2 bytes το καθένα, λόγω του γράμματος W στο όνομα της εντολής) που βρίσκονται στις θέσεις μνήμης LOC1 και LOC2 και τοποθετεί το αποτέλεσμα στον καταχωρητή R2. Παρατηρούμε ότι τόσο ο τύπος των τελεστέων (W – 16 bits) όσο και το πλήθος των τελεστέων (3) δηλώνονται δίπλα στο όνομα της εντολής.**

## ΔΡΑΣΤΗΡΙΟΤΗΤΑ 8

**Να εξηγήσετε τη λειτουργία της παρακάτω εντολής του υπολογιστή VAX;**

**ADDL3 W 200(R3), R2, R1**

**(Το γράμμα L που βρίσκεται στον κωδικό λειτουργίας δηλώνει ότι η πηγαίοι τελεστέοι στους οποίους θα εκτελεστεί η εντολή είναι μεγέθους 2 bytes.)**

## ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 8

**Η εντολή αυτή προσθέτει το περιεχόμενο του καταχωρητή R2 με έναν πηγαίο τελεστέο που βρίσκεται στη μνήμη. Η διεύθυνση του τελεστέου αυτού υπολογίζεται προσθέτοντας το περιεχόμενο του καταχωρητή R3 την τιμή 200. Το αποτέλεσμα της πρόσθεσης μεταξύ των δύο πηγαίων τελεστέων (περιεχόμενο R2 + (περιεχόμενο R3 + 200)), αποθηκεύεται στον καταχωρητή R1. Ο αριθμός 3 που βρίσκεται στον κωδικό εντολής, δηλώνει ότι οι τελεστέοι είναι τρεις.**

# Εντολές ελέγχου

Οι εντολές μεταπήδησης σε έναν υπολογιστή RISC έχουν δύο τρόπους να προσδιορίζουν την διεύθυνση στόχου διακλάδωσης:
**Α.** Η πρόσθεση μιας 26-bit προσημασμένης μετατόπισης (offset) στον PC, προσδιορίζει τη διεύθυνση στόχου.
**Β.** Ο προσδιορισμός ενός καταχωρητή που περιέχει την διεύθυνση στόχου.

Στους υπολογιστές RISC χρησιμοποιείται ο καταχωρητής R31 για την αποθήκευση της διεύθυνσης της εντολής που ακολουθεί την εντολή μεταπήδησης.
Όλες οι διακλαδώσεις είναι με συνθήκη και η διεύθυνση του στόχου διακλάδωσης υπολογίζεται προσθέτοντας μια 16-bit προσημασμένη μετατόπιση (offset) στον PC.
Σε αντίθεση με τους υπολογιστές RISC, στους υπολογιστές CISC υπάρχουν και διακλαδώσεις χωρίς συνθήκη, των οποίων το όνομα καθορίζεται από το μέγεθος της PC-σχετικής μετατόπισης (displacement). Για παράδειγμα η BRB (Brunch Byte) έχει μια 8-bit μετατόπιση.

Παρακάτω παρουσιάζουμε δύο πίνακες με εντολές ελέγχου σε έναν υπολογιστή DLX και σε έναν υπολογιστή VAX.

Για τον **DLX :**

| **Τύπος εντολής/ κωδικός πράξης** | **Σημασία εντολής** |
|---|---|
| **Έλεγχος** | **Διακλαδώσεις με συνθήκη και μεταπηδήσεις. PC-σχετικές ή μέσω καταχωρητών** |
| 1. BEQZ, BNEZ | 1. Διακλάδωση GRP ίσο (Branch if Equal to Zero) / άνισο (Branch if Not Equal to Zero) με το μηδέν. 16-bit offset από PC+4 |
| 2. BFPT, BFPF | 2. Έλεγχος σύγκρισης bit στον FP καταχωρητή κατάστασης και διακλάδωση. 16-bit offset από PC+4 |
| 3. J, JR | 3. Μεταπηδήσεις: 26-bit offset από PC+4 (Jump) ή στόχος σε καταχωρητή (Jump Register). |
| 4. J, JALR | 4. Μεταπήδηση και σύνδεση : σώσε PC+4 στον R31,στόχος είναι PC-σχετικός (Jump And Link) ή ένας καταχωρητής (Jump And Link Register). |
| 5. TRAP | 5. Μεταφορά στο λειτουργικό σύστημα σε ένα διάνυσμα διεύθυνσης |
| 6. RFE | 6. Επιστροφή στον χρησιμοποιούμενο κώδικα από μία εξαίρεση. |

## Παράδειγμα

**BEQZ R1, name: if (Regs[R1] == 0) PC ← name;**
**$((PC + 4) - 2^{15}) \leq name < ((PC + 4) + 2^{15})$**

**BNEZ R4, name: if (Regs[R1] != 0) PC ← name;**
**$((PC + 4) - 2^{15}) \leq name < ((PC + 4) + 2^{15})$**

**J name: PC ← name; $((PC + 4) - 2^{25}) \leq name < ((PC + 4) + 2^{25})$**

**JR R1: PC ← Regs[R1]**

## Παράδειγμα

**Να εξηγήσετε τον τρόπο με τον οποίο εκτελείται η παρακάτω εντολή του DLX:**

**JALR R2**

### Απάντηση:

Είναι μια εντολή μεταπήδησης και σύνδεσης. Με την εντολή αυτή σώζουμε την διεύθυνση της εντολής που ακολουθεί την εντολή μεταπήδησης στον καταχωρητή R31 και φορτώνουμε στον PC την τιμή του περιεχομένου του καταχωρητή R2.
Μπορούμε να εξηγήσουμε την εντολή αυτή χρησιμοποιώντας συμβολισμούς της C:

**Regs[R31] ← PC + 4 ;**
**PC ← Regs[R2]**

Έτσι η ροή του προγράμματος μετατίθεται στη διεύθυνση, η οποία βρίσκεται στον καταχωρητή R2.

## ΔΡΑΣΤΗΡΙΟΤΗΤΑ 9

**Να εξηγήσετε τη λειτουργία της παρακάτω εντολής του υπολογιστή DLX χρησιμοποιώντας συμβολισμούς της C:**

**BEQZ R4, name**

**Στην περίπτωση που δυσκολεύεστε, κρίνεται σκόπιμο να επαναλάβετε την ανάγνωση των εντολών ελέγχου του DLX και των παραδειγμάτων.**

## ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 9

**Είναι μια εντολή διακλάδωσης. Με την εντολή αυτή συγκρίνουμε το περιεχόμενο του καταχωρητή με την τιμή 0 και αν είναι ίσοι τότε ο PC φορτώνεται με την τιμή name. Μπορούμε να εξηγήσουμε την εντολή αυτή χρησιμοποιώντας συμβολισμούς της C:**

**If (Regs[R4] == 0)**
**PC ← name**

**Έτσι η ροή του προγράμματος μετατίθεται στη διεύθυνση name.**

Για τον **VAX:**

| **Τύπος εντολής/ κωδικός πράξης** | **Σημασία εντολής** |
|---|---|
| 1. BEQL, BNEQ | 1. Διακλάδωση αν είναι ίσο / άνισο |
| 2. BLEQ, BGEQ | 2. Διακλάδωση αν είναι μικρότερο από … Διακλάδωση αν είναι μεγαλύτερο από … |
| 3. BRB, BRW | 3. Διακλάδωση χωρίς συνθήκη με μια 8-bit ή 16-bit διεύθυνση |
| 4. JMP | 4. Μεταπήδηση χρησιμοποιώντας έναν οποιονδήποτε τρόπο διευθυνσιοδότησης για να καθοριστεί ο στόχος |
| 5. AOBLEQ | 5. Πρόσθεσε ένα στον τελεστέο: Διακλάδωση αν το αποτέλεσμα είναι μικρότερο ή ίσο του δεύτερου τελεστέου |
| 6. CASE_ | 6. Μεταπήδηση βασισμένη στον επιλογέα case. |

## Παράδειγμα

**Με ποιο τρόπο εκτελείται η παρακάτω σειρά εντολών του υπολογιστή VAX;**

**CMPL R6,(R7);**
**BGEQ exit1**

## Απάντηση:

Είναι μια εντολή σύγκρισης και διακλάδωσης. Η εντολή σύγκρισης θέτει τον κώδικα συνθήκης που θα χρησιμοποιηθεί από την εντολή διακλάδωσης που ακολουθεί. Έτσι με την εντολή αυτή συγκρίνονται το περιεχόμενο του καταχωρητή R6 με το περιεχόμενο της θέσης μνήμης που καθορίζεται από τον καταχωρητή R7. Αν το περιεχόμενο του καταχωρητή R6 είναι μεγαλύτερο ή ίσο του περιεχομένου της θέσης μνήμης τότε ο PC φορτώνεται με την τιμή exit1.
Μπορούμε να εξηγήσουμε την εντολή αυτή χρησιμοποιώντας συμβολισμούς της C:

**If (Regs[R6] >= Mem[Regs[R7]])**
**PC ← exit1**

### Κλήσεις διαδικασιών και επιστροφές

Για την κλήση διαδικασιών και για την επιστροφή από αυτές οι υπολογιστές CISC χρησιμοποιούν ένα σύνολο από δείκτες. Ο δείκτης στοίβας (SP) χρησιμοποιείται και στις μηχανές RISC και δείχνει στην αρχή της στοίβας. Ο δείκτης παραμέτρων (AP) δείχνει στη βάση της λίστας μεταβλητών ή παραμέτρων στην μνήμη, που χρησιμοποιούνται από την διαδικασία. Ο δείκτης πλαισίου (FP) δείχνει στη βάση της λίστας των τοπικών μεταβλητών της διαδικασίας, που βρίσκονται στη μνήμη.

## ΔΡΑΣΤΗΡΙΟΤΗΤΑ 10

**Θυμάστε τι είναι ο δείκτης στοίβας και με ποιον τρόπο αυξάνεται ή μειώνεται η τιμή του; Αν όχι, καλό θα ήταν να επαναλάβετε την ανάγνωση της υποενότητας «Αρχιτεκτονική στοίβας» της πρώτης ενότητας αυτού του κεφαλαίου.**

Στη συνέχεια παρουσιάζουμε έναν πίνακα με κλήσεις διαδικασιών και επιστροφές από αυτές, σε έναν υπολογιστή VAX. Οι αντίστοιχες εντολές για κλήσεις και επιστροφές διαδικασιών του DLX βρίσκονται στον πίνακα εντολών ελέγχου.

Για τον **VAX :**

| **Τύπος εντολής/ κωδικός πράξης** | **Σημασία εντολής** |
|---|---|
| 1. CALLS | 1. Κλήση διαδικασίας με παραμέτρους στην στοίβα |
| 2. CALLG | 2. Κλήση διαδικασιών με λίστα παραμέτρων τύπου FORTRAN |
| 3. JSB | 3. Μεταπήδηση σε υπορουτίνα, σώζοντας την διεύθυνση επιστροφής (στις μηχανές RISC η αντίστοιχη εντολή είναι JAL) |
| 4. RET | 4. Επιστροφή από κλήση διαδικασίας |

## Παράδειγμα

Η εντολή CALLS #n,PROC του υπολογιστή VAX περιέχει δύο τελεστέους. Ο πρώτος (n) καθορίζει το πλήθος των 32-bit παραμέτρων, που έχουν προηγουμένως τοποθετηθεί στην στοίβα. Ο δεύτερος (PROC) είναι η διεύθυνση της διαδικασίας που θα κληθεί.

# Κινητής υποδιαστολής

Και τα δύο είδη υπολογιστών (CISC & RISC) υποστηρίζουν πράξεις σε τελεστέους κινητής υποδιαστολής.

Στη συνέχεια παρουσιάζουμε δύο πίνακες με εντολές για δεδομένα κινητής υποδιαστολής σε μια μηχανή DLX και σε μια μηχανή VAX.

- Για τον **DLX:**

| **Τύπος εντολής/ κωδικός πράξης** | **Σημασία εντολής** |
|---|---|
| **Κινητής υποδιαστολής** | **FP πράξεις σε DP και SP τύπους** |
| 1. ADDD, ADDF | 1. Πρόσθεση DP, SP αριθμών |
| 2. SUBD, SUBF | 2. Αφαίρεση DP, SP αριθμών |
| 3. MULTD, MULTF | 3. Πολλαπλασιασμός DP, SP κινητής υποδιαστολής |
| 4. DIVD, DIVF | 4. Διαίρεση DP, SP κινητής υποδιαστολής |
| 5. CVTF2D, CVTF2I CVTD2F, CTD2I CVTI2F, CVTI2D | 5. Εντολές μετατροπής: $CVT_X2y$ μετατρέπει από τον τύπο x στον τύπο y, όπου x και y είναι I (ακέραιος), D (διπλής ακρίβειας), ή F (απλής ακρίβειας). Και οι δύο Τελεστέοι είναι FPRs. |
| 6. _D, _F | 6. DP και SP συγκρίσεις : «_» = LT, GT, LE, GE, EQ, NE. Θέτει bit στον FP καταχωρητή κατάστασης. |

(SP= απλή ακρίβεια, DP = διπλή ακρίβεια.)

Για τον **VAX:**

| **Τύπος εντολής/ κωδικός πράξης** | **Σημασία εντολής** |
|---|---|
| 1. ADDD_<br>2. SUBD_<br>3. MULTF_ | 1. Πρόσθεση 64-bit αριθμών κινητής υποδιαστολής (διπλής ακρίβειας)<br>2. Αφαίρεση 64-bit αριθμών κινητής υποδιαστολής (διπλής ακρίβειας)<br>3. Πολ/μος 32-bit αριθμών κινητής υποδιαστολής (απλής ακρίβειας) |

## Ανακεφαλαιώνοντας λοιπόν …

**Όλοι οι υπολογιστές υποστηρίζουν ένα σύνολο λειτουργιών και κάθε λειτουργία αποτελείται από ένα σύνολο εντολών.**

**Υπάρχει ένα σύνολο εντολών αλλαγής ροής ελέγχου προγράμματος, που μεταθέτουν τη ροή του προγράμματος, προσφέροντας ευελιξία στην εκτέλεση του προγράμματος.**

**Οι 2 κυριότερες τεχνικές που χρησιμοποιούνται για τον προσδιορισμό της συνθήκης διακλάδωσης είναι οι: καταχωρητής συνθήκης, σύγκριση και διακλάδωση.**

**Οι λειτουργίες που υποστηρίζονται από τους υπολογιστές RISC (DLX) και CISC (VAX) παρουσιάζουν αρκετές διαφορές.**