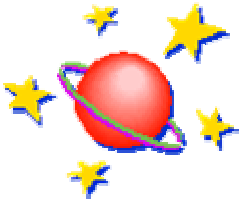


Ενότητα 3η : *Σχεδίαση συνόλου εντολών*

Σκοπός **Ο σκοπός της ενότητας αυτής είναι να παρουσιάσει τους τρόπους κωδικοποίησης και διευθυνσιοδότησης του συνόλου εντολών.**

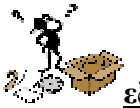
Προσδοκώμενα Αποτελέσματα **Όταν θα έχετε μελετήσει την ενότητα, θα είστε σε θέση να:**

απαριθμείτε τους κυριότερους τύπους τελεστέων,

εξηγείτε τους λόγους για τους οποίους τίθενται περιορισμοί ευθυγράμμισης στα δεδομένα,

αναγνωρίζετε τους τρόπους διευθυνσιοδότησης,

αναφέρετε τους τρόπους κωδικοποίησης του συνόλου εντολών,



εξηγείτε τον τρόπο με τον οποίο κωδικοποιούνται οι εντολές των υπολογιστών VAX και DLX.

ευθυγράμμιση, ενεργή διεύθυνση, προσδιοριστής διεύθυνσης

Τύποι και μεγέθη τελεστέων

Ο βασικός τρόπος με τον οποίο ορίζεται ο τύπος ενός τελεστέου είναι η κωδικοποίησή του στον κωδικό λειτουργίας (opcode) της εντολής.

Οι συνήθεις τύποι τελεστέων είναι οι εξής:

Τύπος τελεστέου	Μέγεθος
Χαρακτήρας με ASCII κωδικοποίηση	1 byte
Μισή λέξη (short integer)	2 byte
Λέξη (integer)	4 byte
Κινητής υποδιαστολής απλής ακρίβειας	4 byte
Κινητής υποδιαστολής διπλής ακρίβειας	8 byte

Σχήμα 2.3.1 - Οι τύποι των τελεστέων και το μέγεθός τους σε bytes. Το μέγεθος ενός τελεστέου εξαρτάται μόνο από τον τύπο του.

Εκτός από τους παραπάνω τύπους τελεστέων υπάρχουν και άλλοι οι οποίοι χρησιμοποιούνται λιγότερο στην αρχιτεκτονική των υπολογιστών. Τέτοιοι τύποι τελεστέων είναι:

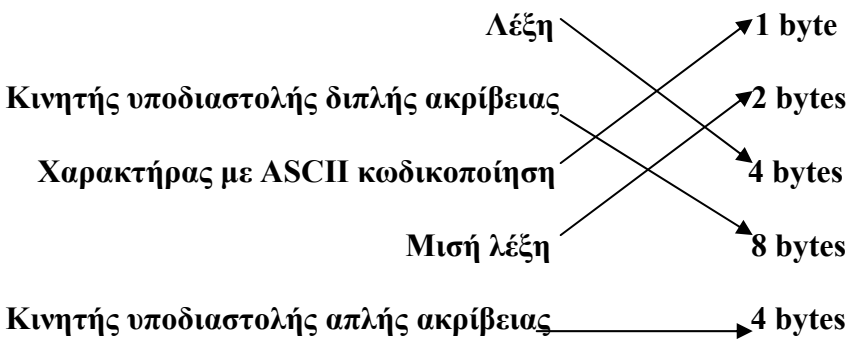
- Συμβολοσειρές χαρακτήρων: ορισμένες αρχιτεκτονικές υπολογιστών υποστηρίζουν εντολές που εκτελούν πράξεις σε συμβολοσειρές χαρακτήρων. Οι πράξεις αυτές είναι συνήθως αρκετά περιοριστικές και αντιμετωπίζουν το κάθε byte της ακολουθίας σαν ένα μονό χαρακτήρα. Τυπικές πράξεις που υποστηρίζονται για ακολουθίες χαρακτήρων είναι οι συγκρίσεις και οι μετακινήσεις.
- Συμπιεσμένοι δεκαδικοί: ορισμένες αρχιτεκτονικές (για εμπορικούς καθαρά λόγους), υποστηρίζουν έναν δεκαδικό τύπο που ονομάζεται συμπιεσμένος δεκαδικός ή δυαδικά κωδικοποιημένος δεκαδικός και χρησιμοποιούνται 4 ψηφία για την κωδικοποίηση των τιμών 0-9. Άρα, δύο δεκαδικά ψηφία συμπιέζονται σε κάθε byte.
- Ασυμπίεστοι δεκαδικοί: υπάρχουν αριθμητικές ακολουθίες χαρακτήρων που ονομάζονται ασυμπίεστοι δεκαδικοί (unpacked decimal) και παρέχονται από τις εντολές: συμπίεση και αποσυμπίεση, για την μετατροπή από και προς αυτούς.

ΔΡΑΣΤΗΡΙΟΤΗΤΑ 1

Να αντιστοιχίσετε τους παρακάτω τύπους τελεστέων με το αντίστοιχο μέγεθός τους σε bytes.

Λέξη	**1 byte**
Κινητής υποδιαστολής διπλής ακρίβειας	**2 byte**
Χαρακτήρας με ASCII κωδικοποίηση	**4 byte**
Μισή λέξη	**8 byte**
Κινητής υποδιαστολής απλής ακρίβειας	**4 byte**

ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 1



ΔΡΑΣΤΗΡΙΟΤΗΤΑ 2

Ας υποθέσουμε ότι έχουμε έναν τελεστέο - λέξη, και ότι η μνήμη είναι χωρισμένη σε θέσεις του ενός byte. Ξεκινώντας από τη θέση μνήμης 1000, γίνεται εγγραφή του τελεστέου αυτού. Ποια θα είναι η επόμενη θέση μνήμης που μπορεί να εγγραφεί ένα νέο δεδομένο; Να επιλέξετε μία από τις παρακάτω απαντήσεις και να αιτιολογήσετε την επιλογή σας.

Α.1001
Β.1002
Γ.1004

ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 2

Η σωστή απάντηση είναι η Γ. 1004.
Κάθε θέση μνήμης καταλαμβάνει 1 byte και η λέξη καταλαμβάνει 4 bytes. Επομένως με την εγγραφή της λέξης στη μνήμη καταλαμβάνονται οι θέσει από 1000 έως και 1003. Άρα το επόμενο δεδομένο θα εγγραφεί στη θέση μνήμης 1004.

Πως ερμηνεύονται οι διευθύνσεις μνήμης

Οι διευθύνσεις ερμηνεύονται ανάλογα με τη διεύθυνση και το μήκος του «αντικειμένου» στο οποίο γίνεται προσπέλαση.
Υπάρχουν δύο διαφορετικοί τρόποι σύμφωνα με τους οποίους μπορούν να οργανωθούν τα bytes σε μία λέξη:

1. **Οργάνωση byte Little Endian:** Το λιγότερο σημαντικό byte της λέξης (little end) τοποθετείται στη μικρότερη διεύθυνση και το περισσότερο σημαντικό byte (big end) τοποθετείται στη μεγαλύτερη διεύθυνση.
2. **Οργάνωση Byte Big Endian:** Το λιγότερο σημαντικό byte της λέξης (little end) τοποθετείται στη μεγαλύτερη διεύθυνση και το περισσότερο σημαντικό byte (big end) τοποθετείται στη μικρότερη διεύθυνση.

ΔΡΑΣΤΗΡΙΟΤΗΤΑ 3

Να απαριθμήσετε τους τρόπους σύμφωνα με τους οποίους μπορούν να οργανωθούν τα bytes σε μία λέξη. Ποια η διαφορά ανάμεσά τους. Στην περίπτωση που δεν μπορείτε να απαντήσετε στην ερώτηση αυτή, κρίνεται σκόπιμο να επαναλάβετε την ανάγνωση της προηγούμενης παραγράφου.

Η διεύθυνση ενός δεδομένου εξαρτάται από την οργάνωση της μνήμης.

Σε πολλούς υπολογιστές, η πρόσβαση σε «αντικείμενα» μεγαλύτερα από 1 byte δεν είναι εφικτή, με αποτέλεσμα τα «αντικείμενα» να χρειάζονται **ευθυγράμμιση**. Η πρόσβαση σε ένα «αντικείμενο» μεγέθους s bytes το οποίο βρίσκεται σε διεύθυνση byte A, ευθυγραμμίζεται αν:

A mod s = 0

Διευθυνσιοδοτούμενο «αντικείμενο»	Ευθυγραμμίζεται στα offsets του byte	Δεν ευθυγραμμίζεται στα offsets του byte
Byte	0,1,2,3,4,5,6,7	Ποτέ
Μισή λέξη	0,2,4,6	1,3,5,7
Λέξη	0,4	1,2,3,5,6,7
Διπλή λέξη	0	1,2,3,4,5,6,7

Ποια είναι η σημασία στο να θέσουμε περιορισμούς ευθυγράμμισης;
Η μη ευθυγραμμισμένη πρόσβαση στη μνήμη απαιτεί πολλαπλές ευθυγραμμισμένες αναφορές, αφού η μη ευθυγραμμισμένη αναφορά προκαλεί επιπλοκές στο υλικό του υπολογιστή (η μνήμη είναι τυπικά ευθυγραμμισμένη στα όρια μίας λέξης). Άρα, ακόμα και στους υπολογιστές που επιτρέπουν μη ευθυγράμμιση, τα προγράμματα με ευθυγραμμισμένες προσβάσεις τρέχουν γρηγορότερα.

Ακόμα και αν τα δεδομένα είναι ευθυγραμμισμένα, τα δεδομένα των καταχωρητών πρέπει να είναι επίσης ευθυγραμμισμένα (τα bytes και οι μισές λέξεις). Ανάλογα με την εντολή που πρέπει να εκτελεστεί, ο υπολογιστής μπορεί να χρειαστεί να κάνει επέκταση προσήμου στην ποσότητα προκειμένου να είναι ευθυγραμμισμένα τα δεδομένα.

ΔΡΑΣΤΗΡΙΟΤΗΤΑ 4

Τι είναι η ευθυγράμμιση των δεδομένων και για ποιο λόγο χρησιμοποιείται από τους υπολογιστές; Στην περίπτωση που δεν μπορείτε να απαντήσετε στην ερώτηση αυτή, κρίνεται σκόπιμο να επαναλάβετε την ανάγνωση της προηγούμενης παραγράφου.

Τρόποι Διευθυνσιοδότησης

Οι αρχιτεκτονικές χρησιμοποιούν διάφορους τρόπους διευθυνσιοδότησης, προκειμένου να προσδιορίσουν τη διεύθυνση του «αντικειμένου» στο οποίο θα γίνει προσπέλαση. Για παράδειγμα, ένας τρόπος διευθυνσιοδότησης μπορεί να προσδιορίσει μία σταθερά, έναν καταχωρητή ή μία περιοχή στη μνήμη. Όταν μία περιοχή στη μνήμη χρησιμοποιείται, η πραγματική διεύθυνση μνήμης, καθορισμένη από τον τρόπο διευθυνσιοδότησης, καλείται **ενεργή διεύθυνση.**

Απευθείας ή **κυριολεκτικοί** τρόποι διευθυνσιοδότησης συνήθως θεωρούνται οι τρόποι διευθυνσιοδότησης της μνήμης (παρόλο που η τιμή στην οποία γίνεται προσπέλαση βρίσκεται στον κώδικα των εντολών).
Ο τρόπος διευθυνσιοδότησης που εξαρτάται από τον απαριθμητή προγράμματος (PC) ονομάζεται **PC – σχετική διευθυνσιοδότηση**. Ο τρόπος αυτός, χρησιμοποιείται κυρίως στον προσδιορισμό διευθύνσεων κώδικα, στις εντολές που ελέγχουν τη ροή εκτέλεσης του προγράμματος.
Στον πίνακα που ακολουθεί (σχήμα 2.3.2), απεικονίζονται τρόποι διευθυνσιοδότησης, η σημασία τους, καθώς και πότε αυτοί χρησιμοποιούνται.

Τρόπος Διευθυνσιοδότησης	**Παράδειγμα εντολής**	**Σημασία**	**Πότε χρησιμοποιείται**
Καταχωρητή	Add R4, R3	Regs[R4]← Regs[R4]+ Regs[R3]	Όταν μια τιμή φυλάσσεται σε καταχωρητή
Απευθείας	Add R4, # 3	Regs[R4]← Regs[R4] + 3	Για σταθερές
Με μετατόπιση	Add R4, 100(R1)	Regs[R4]← Regs[R4]+ Mem[100+Regs[R1]]	Πρόσβαση τοπικών μεταβλητών
Έμμεσος καταχωρητή	Add R4, (R1)	Regs[R4]← Regs[R4]+ Mem[Regs[R1]]	Πρόσβαση με χρήση δείκτη ή υπολογισμένη διεύθυνση
Με δείκτη	Add R3, (R1+ R2)	Regs[R3]← Regs[R3]+ Mem[Regs[R1]+ Regs[R2]]	Συχνά χρήσιμος σε διευθυνσιοδότηση πινάκων : R1= βάση πίνακα, R2=ποσότητα δείκτη
Άμεσος ή Απόλυτος	Add R1, (1001)	Regs[R1]← Regs[R1]+ Mem[1001]	Συχνά χρήσιμος για πρόσβαση στατικών δεδομένων. Η σταθερά διεύθυνσης ίσως χρειαστεί να γίνει μεγάλη

Έμμεσος μνήμης	Add R1, @(R3)	Regs[R1]← Regs[R1]+ Mem[Mem[Regs[R3]]]	Αν R3 είναι διεύθυνση του δείκτη p, παίρνουμε ουσιαστικά το *p
Αυτοαύξησης	Add R1, (R2)+	Regs[R1]← Regs[R1]+ Mem[Regs[R2]] Regs[R2] ← Regs[R2]+d	Χρήσιμος για κίνηση μέσα σε πίνακες που βρίσκονται μέσα σε βρόγχο. Ο R2 δείχνει την αρχή του πίνακα. Κάθε αναφορά αυξάνει τον R2 κατά ποσότητα d.
Αυτοελάττωσης	Add R1, -(R2)	Regs[R2] ← Regs[R2]-d Regs[R1]← Regs[R1]+ Mem[Regs[R2]]	Το ίδιο με την αυτοαύξηση. Αυτοαύξηση/ αυτοελάττωση μπορούν να χρησιμοποιηθούν σαν push και pop για υλοποίηση στοίβας.
Κλίμακας	Add R1,100(R2)[R3]	Regs[R1] ← Regs[R1]+ Mem[100+Regs[R2]+ Regs[R3] *d]	Χρησιμοποιείται σε πίνακες με δείκτες. Μπορεί να χρησιμοποιηθεί σαν τρόπος διευθυνσιοδότησης με δείκτη σε μερικούς υπολογιστές.

Σχήμα 2.3.2 – Τρόποι διευθυνσιοδότησης, η σημασία τους και που αυτοί χρησιμοποιούνται. Το σύμβολο Mem αναφέρεται στη μνήμη και το Regs στους καταχωρητές. Σε τρόπους διευθυνσιοδότησης αυτοαύξησης, αυτοελάττωσης και κλίμακας, η μεταβλητή d καθορίζει το μέγεθος του τμήματος δεδομένων στο οποίο γίνεται πρόσβαση (αν η εντολή έχει πρόσβαση σε 1,2,4 ή 8 bytes). Αυτό σημαίνει ότι αυτοί οι τρόποι διευθυνσιοδότησης είναι χρήσιμοι μόνο όταν τα στοιχεία στα οποία γίνεται πρόσβαση είναι γειτονικά στη μνήμη.

ΔΡΑΣΤΗΡΙΟΤΗΤΑ 5

Ποιος τρόπος διευθυνσιοδότησης χρησιμοποιείται στην παρακάτω εντολή και πόσες προσβάσεις στη μνήμη απαιτούνται; Να αιτιολογήσετε την απάντησή σας.

ADD R1, @ (R2)

ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 5

Ο τρόπος διευθυνσιοδότησης που χρησιμοποιείται είναι ο έμμεσος μνήμης. Μπορούμε να απεικονίσουμε συμβολικά την εκτέλεση της εντολής ως εξής:

Regs[R1] ← Regs[R1] + Mem[Mem[Regs[R2]]]

Όπως μπορούμε να δούμε, χρειάζονται δύο προσβάσεις στη μνήμη. Στην πρώτη πρόσβαση, ανακαλείται η θέση στη μνήμη που βρίσκεται στη διεύθυνση η οποία προσδιορίζεται από το περιεχόμενο του καταχωρητή R2. Στη συνέχεια, το περιεχόμενο της θέσης μνήμης αυτής δείχνει τη νέα διεύθυνση της οποίας το περιεχόμενο θα προστεθεί με το περιεχόμενο του καταχωρητή R1.

Οι τρόποι διευθυνσιοδότησης έχουν και κάποιες άλλες επιδράσεις:

- Μειώνουν σημαντικά τον αριθμό των εντολών.
- Συνεισφέρουν στο να γίνει πιο πολύπλοκη η δημιουργία μιας μηχανής
- Συχνά αυξάνουν το μέσο CPI μηχανών που χρησιμοποιούν αυτούς τους τρόπους.

Οι τρόποι διευθυνσιοδότησης Απευθείας και με Μετατόπιση κυριαρχούν μέσα στο σύνολο των εναλλακτικών λύσεων. Ορισμένες ιδιότητες αυτών των ευρέως χρησιμοποιούμενων τρόπων είναι οι εξής:

❖ Τρόπος διευθυνσιοδότησης με Μετατόπιση

Η επιλογή του μεγέθους των πεδίων μετατόπισης είναι σημαντική επειδή επηρεάζει άμεσα το μήκος εντολής.

❖ Απευθείας ή Κυριολεκτικός τρόπος διευθυνσιοδότησης

Η απευθείας πρόσβαση μπορεί να χρησιμοποιηθεί στις αριθμητικές ή λογικές εντολές, σε συγκρίσεις (κυρίως για τις εντολές διακλάδωσης) και σε μετακινήσεις στις οποίες μία σταθερά αναζητείται σ' έναν καταχωρητή.

Όπως οι τιμές μετατόπισης, έτσι και το μέγεθος των απευθείας τιμών των διευθύνσεων επηρεάζει το μήκος των εντολών. Κατά κύριο λόγο, χρησιμοποιούνται οι άμεσες τιμές που είναι μικρές,. Ωστόσο, οι μεγάλες απευθείας τιμές χρησιμοποιούνται κυρίως για υπολογισμούς διευθυνσιοδότησης.

Παράδειγμα

Ποια εντολή θα χρησιμοποιήσουμε για να προσθέσουμε την τιμή 53 στον καταχωρητή R1 και ποιος ο τρόπος διευθυνσιοδότησης, για την κάθε μια από τις παρακάτω περιπτώσεις :

Α. Η τιμή 53 δίνεται απευθείας
Β. Η τιμή 53 βρίσκεται τοποθετημένη στον καταχωρητή R2
Γ. Η τιμή 53 βρίσκεται στη θέση μνήμης 1001
Δ. Η τιμή 53 βρίσκεται στη μνήμη. Η διεύθυνση είναι καταχωρημένη στον καταχωρητή R2

Χρησιμοποιήστε συμβολισμούς για να περιγράψετε τη λειτουργία της εντολής.

Απάντηση:

Α. Η εντολή για την 1η περίπτωση είναι:

ADD R1,#53

Στην εντολή αυτή χρησιμοποιείται ο απευθείας τρόπος διευθυνσιοδότησης.
Περιγραφή εντολής:

Regs[R1] ← Regs[R1] + 53

Β. Η εντολή για την 2η περίπτωση είναι:

ADD R1,R2

Στην εντολή αυτή χρησιμοποιείται ο τρόπος διευθυνσιοδότησης με καταχωρητή. Ο καταχωρητής R2 περιέχει την τιμή 53.
Περιγραφή εντολής :

Regs[R1] ← Regs[R1] + Regs[R2]

Γ. Η εντολή για την 3η περίπτωση είναι:

ADD R1,(1001)

Στην εντολή αυτή χρησιμοποιείται ο άμεσος τρόπος διευθυνσιοδότησης. Στη διεύθυνση 1001 βρίσκεται η τιμή 53.
Περιγραφή εντολής :

Regs[R1] ← Regs[R1] + Mem[1001]

Δ. Η εντολή για την 4η περίπτωση είναι:

ADD R1,(R2)

Στην εντολή αυτή χρησιμοποιείται ο έμμεσος τρόπος διευθυνσιοδότησης με καταχωρητή. Η τιμή 53 βρίσκεται στη θέση μνήμης, η διεύθυνση της οποίας βρίσκεται καταχωρημένη στον καταχωρητή R2.
Περιγραφή εντολής :

Regs[R1] ← Regs[R1] + Mem[Regs[R2]]

Παράδειγμα

Έστω ότι έχουμε έναν πίνακα με στοιχεία τύπου λέξεων 2 bytes, τα οποία είναι τοποθετημένα σε συνεχόμενες θέσεις στη μνήμη, η οποία είναι διευθυνσιοδοτημένη κατά byte. Ποια σειρά εντολών πρέπει να χρησιμοποιηθεί για να προσθέσουμε τα δύο πρώτα στοιχεία του πίνακα, όταν γνωρίζουμε ότι το πρώτο στοιχείο βρίσκεται στη θέση μνήμης 100;
Το αποτέλεσμα της πρόσθεσης να τοποθετηθεί στον καταχωρητή R4,ο οποίος αρχικά έχει την τιμή 0. Χρησιμοποιήστε μόνο εντολές πρόσθεσης.

A. Χρησιμοποιώντας μόνο τον απόλυτο τρόπο διευθυνσιοδότησης
B. Χρησιμοποιώντας τον τρόπο διευθυνσιοδότησης με αυτοαύξηση και καταχωρητή. Χρησιμοποιήστε τον καταχωρητή R1 που έχει αρχικά την τιμή 0.

Η εντολή ADD δέχεται μόνο δύο τελεστέους.

Απάντηση:

Α. Η σειρά εντολών για την 1η περίπτωση είναι:

ADD R4,(100);
ADD R4,(102)

Περιγραφή εντολής :

Regs[R4] ← Regs[R4] + Mem[100];
Regs[R4] ← Regs[R4] + Mem[102]

Αρχικά προσθέτουμε το περιεχόμενο του καταχωρητή R4, το οποίο είναι 0, με το 1ο στοιχείο του πίνακα που βρίσκεται στη θέση μνήμης 100. Τώρα ο καταχωρητής R4 περιέχει το 1ο στοιχείο του πίνακα. Στη συνέχεια προσθέτουμε στο περιεχόμενο του καταχωρητή R4(1ο στοιχείο του πίνακα) το περιεχόμενο της θέσης μνήμης 102, στην οποία βρίσκεται το 2ο στοιχείο του πίνακα. Έτσι έχουμε προσθέσει τα δύο πρώτα στοιχεία του πίνακα και το αποτέλεσμα το έχουμε τοποθετήσει στον καταχωρητή R4.

Β. Η σειρά εντολών για τη 2η περίπτωση είναι:

ADD R1,#100 ;
ADD R4,(R1)+ ;
ADD R4,(R1)

Περιγραφή εντολής :

Regs[R1] ← Regs[R1] + 100 ;
Regs[R4] ← Regs[R4] + Mem[Regs[R1]] ;
Regs[R1] ← Regs[R1] + 2 ;
Regs[R4] ← Regs[R4] + Mem[Regs[R1]]

Αρχικά προσθέτουμε την τιμή 100 στον καταχωρητή R1. Τώρα ο καταχωρητής R1 περιέχει την τιμή 100 και η θέση μνήμης που καθορίζεται από τον καταχωρητή R1 (Mem[Regs[R1]]) περιέχει το 1ο στοιχείο του πίνακα. Χρησιμοποιώντας στην πρόσθεση τον τρόπο διευθυνσιοδότησης με αυτοαύξηση, προσθέτουμε το 1ο στοιχείο

του πίνακα με το περιεχόμενο του καταχωρητή R4 (δηλαδή με το 0) και αυξάνουμε το περιεχόμενο του R1 κατά 2, ώστε να δείχνει στην επόμενη λέξη (δηλαδή στο 2° στοιχείο του πίνακα). Έτσι στην επόμενη πρόσθεση προσθέτουμε το 2° στοιχείο του πίνακα με το περιεχόμενο του καταχωρητή R4 (1° στοιχείο του πίνακα) και το αποτέλεσμα το τοποθετούμε στον καταχωρητή R4.

ΔΡΑΣΤΗΡΙΟΤΗΤΑ 6

Να συμπληρώσετε την παρακάτω πρόταση χρησιμοποιώντας μία ή περισσότερες από τις απαντήσεις που ακολουθούν.
Ο τρόπος διευθυνσιοδότησης @-Rn είναι:
Α. Έμμεσος μνήμης
Β. Καταχωρητή
Γ. Δείκτη
Δ. Αυτοαύξησης
Ε. Αυτοελάττωσης

ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 6

Ο τρόπος διευθυνσιοδότησης @-Rn είναι:
Α. Έμμεσος μνήμης
Ε. Αυτοελάττωσης

ΔΡΑΣΤΗΡΙΟΤΗΤΑ 7

- **Ποιος τρόπος διευθυνσιοδότησης χρησιμοποιείται στην παρακάτω εντολή;**

ADD R1, @(R2)+

- **Να εξηγήσετε τον τρόπο με τον οποίο εκτελείται η εντολή αυτή, χρησιμοποιώντας συμβολισμούς.**

ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 7

- **Ο τρόπος διευθυνσιοδότησης που χρησιμοποιείται είναι ο έμμεσος μνήμης με αυτοαύξηση.**
- **Regs[R1] ← Regs[R1] + Mem[Mem[Regs[R2]]] ;**
  Regs[R2] ← Regs[R2] + d
  Το d καθορίζει το μέγεθος του τμήματος δεδομένων στο οποίο γίνεται η πρόσβαση.

ΔΡΑΣΤΗΡΙΟΤΗΤΑ 8

Να απαριθμήσετε τους βασικούς τρόπους διευθυνσιοδότησης και να δώσετε ένα παράδειγμα για κάθε έναν. Να συγκρίνετε την απάντησή σας με τον πίνακα του σχήματος 2.3.2.

ΔΡΑΣΤΗΡΙΟΤΗΤΑ 9

Ποια εντολή θα χρησιμοποιήσουμε για να εκτελεστούν οι παρακάτω λειτουργίες και ποιος είναι ο τρόπος διευθυνσιοδότησης σε κάθε περίπτωση; Να εξηγήσετε την εκτέλεση κάθε εντολής χρησιμοποιώντας συμβολισμούς.

α) Πρόσθεση του περιεχομένου του καταχωρητή R1 και του καταχωρητή R2 και τοποθέτηση του αποτελέσματος στον καταχωρητή R2.

β) Πρόσθεση της τιμής 24 στον καταχωρητή R3.

γ) Πρόσθεση του περιεχομένου της θέσης μνήμης με διεύθυνση 251 στον καταχωρητή R4.

ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 9

α) ADD R2, R1

Ο τρόπος διευθυνσιοδότησης είναι με καταχωρητή.

Regs[R2] ← Regs[R2] + Regs[R1]

β) ADD R3, #24

Ο τρόπος διευθυνσιοδότησης είναι απ' ευθείας.

Regs[R3] ← Regs[R3] + 24

γ) ADD R4, (251)

Ο τρόπος διευθυνσιοδότησης είναι άμεσος.

Regs[R4] ← Regs[R4] + Mem[251]

ΔΡΑΣΤΗΡΙΟΤΗΤΑ 10

Έστω ότι ένας πίνακας περιέχει στοιχεία τύπου λέξης με μέγεθος 4 bytes. Τα στοιχεία του πίνακα είναι τοποθετημένα σε συνεχόμενες θέσεις στη μνήμη, η οποία είναι διευθυνσιοδοτημένη κατά byte.

- **Να περιγράψετε τη σειρά εντολών που πρέπει να χρησιμοποιηθεί για να γίνει πρόσθεση των δύο πρώτων στοιχείων του πίνακα και το αποτέλεσμα να τοποθετηθεί στον καταχωρητή R1. (Χρησιμοποιήστε μόνο εντολές πρόσθεσης.)**
- **Να εξηγήσετε την εκτέλεση κάθε εντολής χρησιμοποιώντας συμβολισμούς.**
- **Ποιο θα είναι το αποτέλεσμα αν το πρώτο στοιχείο του πίνακα βρίσκεται στη θέση μνήμης 44 και έχει τιμή 20, το δεύτερο στοιχείο έχει τιμή 150 και το περιεχόμενο του καταχωρητή R1 έχει τιμή 50;**
- **Να απαντήσετε στα παραπάνω ερωτήματα χρησιμοποιώντας τον:**
  α) απόλυτο τρόπο διευθυνσιοδότησης,
  β) τρόπο διευθυνσιοδότησης με αυτοαύξηση και καταχωρητή.
  (Να χρησιμοποιήσετε τον καταχωρητή R2 που η αρχική του τιμή είναι 100.)

ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 10

α) Αρχικά προστίθεται το 1^{ο} στοιχείο του πίνακα, δηλαδή το περιεχόμενο της θέσης μνήμης 44 (20) και το περιεχόμενο του καταχωρητή R1 (50). Το αποτέλεσμα (70) τοποθετείται στον καταχωρητή R1. Στη συνέχεια προστίθεται το 2^{ο} στοιχείο του πίνακα που είναι περιεχόμενο της θέσης μνήμης 48 (150) και το περιεχόμενο του καταχωρητή R (70). Το αποτέλεσμα (220) τοποθετείται στον καταχωρητή R1.

Η σειρά εντολών είναι: **ADD R1,(44);**
ADD R1,(48)

Περιγραφή εντολής : **Regs[R1] ← Regs[R1] + Mem[44];**
Regs[R1] ← Regs[R1] + Mem[48]

β) Αρχικά προστίθεται η τιμή 44 με το περιεχόμενου του καταχωρητή R2 (100). Το αποτέλεσμα (144) τοποθετείται στον καταχωρητή R2. Το νέο περιεχόμενο του καταχωρητή R2 είναι η θέση μνήμης που υπάρχει το 1^{ο} στοιχείο του πίνακα. Χρησιμοποιώντας στην πρόσθεση τον τρόπο διευθυνσιοδότησης με αυτοαύξηση, προστίθεται το 1^{ο} στοιχείο του πίνακα (20) με το περιεχόμενο του καταχωρητή R1 (50) και αυξάνουμε το περιεχόμενο του R1 κατά 4, ώστε να δείχνει στην επόμενη λέξη (δηλαδή στο 2^{ο} στοιχείο του πίνακα, που βρίσκεται στη θέση 148). Στη συνέχεια προστίθεται το 2^{ο} στοιχείο του πίνακα (150) με το περιεχόμενο του καταχωρητή R1 (70) και το αποτέλεσμα (220) το τοποθετούμε στον καταχωρητή R1.

Η σειρά εντολών είναι: **ADD R2,#44 ;**
ADD R1,(R2)+ ;
ADD R1,(R2)

Περιγραφή εντολής : **Regs[R2] ← Regs[R2] + 44 ;**
Regs[R1] ← Regs[R1] + Mem[Regs[R2]] ;
Regs[R2] ← Regs[R2] + 4 ;
Regs[R1] ← Regs[R1] + Mem[Regs[R2]]

Κωδικοποίηση συνόλου εντολών

Ο τύπος και το μέγεθος των τελεστέων επηρεάζουν το μέγεθος του μεταγλωττισμένου προγράμματος, την κωδικοποίησή των εντολών (σε δυαδική μορφή), την αποκωδικοποίησή τους από την ΚΜΕ αλλά και την εκτέλεση της εντολής από αυτήν.

ΔΡΑΣΤΗΡΙΟΤΗΤΑ 11

Θυμάστε με ποιον τρόπο καθορίζεται η λειτουργία που πρόκειται να εκτελεστεί από μία εντολή; Για περισσότερες λεπτομέρειες καλό θα ήταν να ανατρέξετε στην 1η ενότητα αυτού του κεφαλαίου και συγκεκριμένα στην υποενότητα «Αρχιτεκτονικές συνόλου εντολών».

ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 11

Κάθε κωδικοποιημένη εντολή αποτελείται από πεδία. Ένα από τα πεδία αυτά καθορίζει τη λειτουργία που πρόκειται να εκτελεστεί, δηλαδή τον τελεστή και ονομάζεται πεδίο κωδικού λειτουργίας (opcode field).

Για την κωδικοποίηση των εντολών, έχει μεγάλη σημασία ο τρόπος που θα κωδικοποιήσουμε τους τρόπους διευθυνσιοδότησης στις εντολές και εξαρτάται από τους τρόπους διευθυνσιοδότησης και το βαθμό ανεξαρτησίας μεταξύ των κωδικών λειτουργίας και των τρόπων αυτών. Για παράδειγμα, ορισμένοι υπολογιστές διαθέτουν έναν έως πέντε τελεστέους με δέκα τρόπους διευθυνσιοδότησης για κάθε τελεστέο. Αυτό έχει ως αποτέλεσμα να προκύπτει ένας μεγάλος αριθμός συνδυασμών, έτσι ώστε να χρειάζεται ένας χωριστός **προσδιοριστής διεύθυνσης** για κάθε τελεστέο: Ο προσδιοριστής διεύθυνσης καθορίζει τον τρόπο διευθυνσιοδότησης που χρησιμοποιείται για την προσπέλαση του τελεστέου. Στο άλλο άκρο είναι οι υπολογιστές φόρτωσης-αποθήκευσης (ή καταχωρητή-καταχωρητή) με έναν μόνο τελεστέο μνήμης και έναν ή δύο τρόπους διευθυνσιοδότησης. Είναι φανερό, σ' αυτή την περίπτωση, ότι ο τρόπος διευθυνσιοδότησης μπορεί να κωδικοποιηθεί σαν τμήμα του κωδικού λειτουργίας.

Κατά την κωδικοποίηση των εντολών, τόσο ο αριθμός των καταχωρητών όσο και ο αριθμός των τρόπων διευθυνσιοδότησης έχουν σημαντική επίδραση στο μέγεθος των εντολών, αφού το πεδίο του τρόπου διευθυνσιοδότησης και το πεδίο του καταχωρητή μπορούν να εμφανιστούν πολλές φορές σε μια εντολή. Πράγματι, στις περισσότερες εντολές καταναλώνονται πολλά περισσότερα bits στην κωδικοποίηση τρόπων διευθυνσιοδότησης και πεδίων καταχωρητών απ' ό,τι στον καθορισμό του κωδικού λειτουργίας

Επειδή οι τρόποι διευθυνσιοδότησης και τα πεδία καταχωρητών καλύπτουν ένα μεγάλο ποσοστό από τα bits της εντολής, η κωδικοποίησή τους θα επηρεάσει σημαντικά την αποκωδικοποίηση των εντολών.

Υπάρχουν 3 τρόποι για την κωδικοποίηση του συνόλου εντολών:

- **Μεταβλητή κωδικοποίηση**
- **Σταθερή κωδικοποίηση**
- **Υβριδική κωδικοποίηση**

Μεταβλητή κωδικοποίηση

Λειτουργία και αριθμός τελεστέων	Προσδιορι-στής διεύθυνσης #1	Πεδίο διεύθυνσης #1	…	Προσδιορι-στής διεύθυνσης #n	Πεδίο διεύθυνσης #n

Η μεταβλητή κωδικοποίηση επιτρέπει εικονικά σε όλους τους τρόπους διευθυνσιοδότησης να χρησιμοποιούνται από όλες τις εντολές. Συνήθως χρησιμοποιείται όταν υπάρχουν πολλοί τρόποι διευθυνσιοδότησης και πράξεις.

Η μεταβλητή κωδικοποίηση χρησιμοποιείται από τους υπολογιστές CISC (π.χ. VAX).

Παρακάτω παρουσιάζεται μια σειρά από παραδείγματα κωδικοποίησης εντολών του VAX.

Παράδειγμα

Πώς κωδικοποιούνται οι παρακάτω εντολές του VAX και ποια τα μεγέθη τους:

Α. CLRL (R4), αν ο έμμεσος τρόπος διεθυνσιοδότησης έχει κωδικό 6.

Β. CLRW B↑7(R2), αν ο τρόπος διευθυνσιοδότησης με μετατόπιση (B↑) έχει κωδικό 10.

Γ. CLRW W↑ 100(R2), αν ο τρόπος διευθυνσιοδότησης με μετατόπιση (W↑) έχει κωδικό 12.

Δ. CLRW L↑ 1000(R2), αν ο τρόπος διευθυνσιοδότησης με μετατόπιση (L↑) έχει κωδικό 14.

Ε. ADDW3 (R2), 50(R3), R1, αν ο τρόπος διευθυνσιοδότησης άμεσος με καταχωρητή έχει κωδικό 5, ο έμμεσος με καταχωρητή έχει κωδικό 6 και ο τρόπος με μετατόπιση έχει κωδικό 12.

Απάντηση:

Α. CLRL (R4): Με την εντολή αυτή μηδενίζεται το περιεχόμενο της θέσης μνήμης, η διεύθυνση της οποίας βρίσκεται στον καταχωρητή R4.

1 byte

CLRL		Opcode
6	R4	Operand Specifier
4 bits	**4 bits**	

Όπως παρατηρούμε το μέγεθος της εντολής είναι 2 bytes. Στο πρώτο πεδίο βρίσκεται κωδικοποιημένη η πράξη καθώς και το πλήθος των τελεστέων. Στο δεύτερο πεδίο βρίσκεται ο προσδιοριστής τελεστέου (operand specifier): στα 4 πρώτα bits βρίσκεται κωδικοποιημένος ο τρόπος διευθυνσιοδότησης του τελεστέου (στο συγκεκριμένο παράδειγμα είναι το 6) και στα επόμενα 4 καθορίζεται ο καταχωρητής που χρησιμοποιείται.

Β. CLRW B↑7(R2): Με την εντολή αυτή μηδενίζεται το περιεχόμενο μεγέθους 16 – bits (λόγω του γράμματος W στην εντολή CLR_) που βρίσκεται στη μνήμη. Η διεύθυνση της μνήμης υπολογίζεται προσθέτοντας την απευθείας τιμή 7 (μετατόπιση) στο περιεχόμενο του καταχωρητή R2. Το σύμβολο B↑ δηλώνει ότι η τιμή 7(μετατόπιση) είναι μεγέθους 1 bytes.

1 byte

CLRW		Opcode
10	R2	Operand Specifier
7		

Όπως παρατηρούμε το μέγεθος της εντολής είναι 3 bytes. Στο πρώτο πεδίο βρίσκεται κωδικοποιημένη η πράξη καθώς και το πλήθος των τελεστέων. Στο δεύτερο πεδίο, μεγέθους 2 bytes, βρίσκεται ο προσδιοριστής τελεστέου: στα 4 πρώτα bits βρίσκεται κωδικοποιημένος ο τρόπος διευθυνσιοδότησης του τελεστέου (στο παράδειγμά μας είναι το 10), στα επόμενα 4 βρίσκεται ο καταχωρητής R2 και το τελευταίο byte περιέχει την τιμή μετατόπισης.

Γ. CLRW W↑ 100(R2): Με την εντολή αυτή μηδενίζεται το περιεχόμενο μεγέθους 16 – bits (λόγω του γράμματος W στην εντολή CLR_) που βρίσκεται στη μνήμη. Η διεύθυνση της μνήμης υπολογίζεται προσθέτοντας την απευθείας τιμή 100(μετατόπιση) στο περιεχόμενο του καταχωρητή R2. Το σύμβολο W↑ δηλώνει ότι η τιμή 100(μετατόπιση) είναι μεγέθους 2 bytes.

1 byte

CLRW		Opcode
12	R2	Operand Specifier
100		

Όπως παρατηρούμε το μέγεθος της εντολής είναι 4 bytes. Στο πρώτο πεδίο βρίσκεται κωδικοποιημένη η πράξη καθώς και το πλήθος των τελεστέων. Στο δεύτερο πεδίο, μεγέθους 3 bytes, βρίσκεται ο προσδιοριστής τελεστέου: στα 4 πρώτα bits βρίσκεται κωδικοποιημένος ο τρόπος διευθυνσιοδότησης του τελεστέου (στο παράδειγμά μας είναι το 12), στα επόμενα 4 βρίσκεται ο καταχωρητής R2 και στα δύο τελευταία bytes περιέχεται η τιμή μετατόπισης που είναι 100.

Δ. CLRW L↑ 1000(R2): Με την εντολή αυτή μηδενίζεται το περιεχόμενο μεγέθους 16 – bits (λόγω του γράμματος W στην εντολή CLR_) που βρίσκεται στη μνήμη. Η διεύθυνση της μνήμης υπολογίζεται προσθέτοντας την απευθείας τιμή 1000(μετατόπιση) στο περιεχόμενο του καταχωρητή R2. Το σύμβολο L↑ δηλώνει ότι η τιμή 100(μετατόπιση) είναι μεγέθους 4 bytes.

1 byte

CLRW		Opcode
14	R2	Operand Specifier
1000		

Όπως παρατηρούμε το μέγεθος της εντολής είναι 6 bytes. Στο πρώτο πεδίο βρίσκεται κωδικοποιημένη η πράξη καθώς και το πλήθος των τελεστέων. Στο δεύτερο πεδίο, μεγέθους 5 bytes, βρίσκεται ο προσδιοριστής τελεστέου: στα 4 πρώτα bits βρίσκεται κωδικοποιημένος ο τρόπος διευθυνσιοδότησης του τελεστέου (στο παράδειγμά μας είναι το 14), στα επόμενα 4 βρίσκεται ο καταχωρητής R2 και στα τέσσερα τελευταία bytes περιέχεται η τιμή μετατόπισης που είναι 1000.

Ε. ADDW3 (R2), 50(R3), R1: Με την εντολή αυτή προσθέτουμε δύο πηγαίους τελεστέους οι οποίοι βρίσκονται στη μνήμη και το αποτέλεσμα το περνάμε στον καταχωρητή R1. Ο πρώτος τελεστέος βρίσκεται στη διεύθυνση μνήμης η οποία είναι καταχωρημένη στον καταχωρητή R2. Ο δεύτερος βρίσκεται στη διεύθυνση μνήμης η οποία υπολογίζεται προσθέτοντας την τιμή 50(μετατόπιση) στο περιεχόμενο του καταχωρητή R3.

1 byte

ADDW3		Opcode
6	R2	Operand Specifier1
12	R3	Operand Specifier2
50		
5	R1	Operand Specifier3

Όπως παρατηρούμε το μέγεθος της εντολής είναι 5 bytes. Στο πρώτο πεδίο βρίσκεται κωδικοποιημένη η πράξη καθώς και το πλήθος των τελεστέων (που στο παράδειγμά μας είναι τρεις). Στο δεύτερο πεδίο μεγέθους 1byte βρίσκεται ο προσδιοριστής τελεστέου του 1^{ου} τελεστέου: στα 4 πρώτα bits βρίσκεται κωδικοποιημένος ο τρόπος διευθυνσιοδότησης του 1^{ου} τελεστέου (στο παράδειγμά μας είναι το 6), στα επόμενα 4 βρίσκεται ο καταχωρητής R1. Στο τρίτο πεδίο μεγέθους 2 bytes βρίσκεται ο προσδιοριστής τελεστέου του 2^{ου} τελεστέου και στο τελευταίο πεδίο μεγέθους 1 byte βρίσκεται ο προσδιοριστής τελεστέου του 3^{ου} τελεστέου (που είναι και ο τελεστέος προορισμού).

ΔΡΑΣΤΗΡΙΟΤΗΤΑ 12

- **Να εξηγήσετε με ποιον τρόπο έχει κωδικοποιηθεί η παρακάτω εντολή του VAX.**

ADDI3 R1, 737(R2), (R3)

- **Ποιο είναι το συνολικό μήκος της εντολής σε bytes;**

ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 12

Η κωδικοποίηση της εντολής είναι μεταβλητή. Το όνομα ADDI3 εκφράζει μία 32-bit εντολή ακεραίων με τρεις τελεστέους, και αυτός ο κωδικός πράξης καταλαμβάνει 1 byte. Ένας προσδιοριστής τελεστέου του VAX είναι 1 byte, με τα 4 πρώτα bits να καθορίζουν τον τρόπο διευθυνσιοδότησης και τα επόμενα 4 bits τον καταχωρητή που χρησιμοποιείται σ' αυτόν τον τρόπο διευθυνσιοδότησης. Ο πρώτος προσδιοριστής τελεστέου (R1) προσδιορίζει διευθυνσιοδότηση καταχωρητή με χρήση του καταχωρητή R1, και αυτός ο προσδιοριστής έχει μήκος 1 byte. Ο δεύτερος προσδιοριστής τελεστέου (737(R2)) προσδιορίζει διευθυνσιοδότηση με δείκτη. Έχει δύο τμήματα : Το πρώτο είναι 1 byte που καθορίζει τον 16-bit τρόπο διευθυνσιοδότησης με δείκτη και τον καταχωρητή βάσης (R2). Το δεύτερο τμήμα είναι η μετατόπιση μήκους 2byte (737). Ο τρίτος προσδιοριστής τελεστέου (R3) καθορίζει τον τρόπο διευθυνσιοδότησης εμμέσου καταχωρητή, χρησιμοποιώντας τον καταχωρητή R3. Επομένως, αυτή η εντολή έχει δύο προσπελάσεις στα δεδομένα της μνήμης, και το συνολικό μήκος εντολής είναι: 1 + (1) + (1+2) + (1) = 6 bytes.

Σταθερή κωδικοποίηση

Λειτουργία	Πεδίο διεύθυνσης #1	Πεδίο διεύθυνσης #2	Πεδίο διεύθυνσης #3

Στη σταθερή κωδικοποίηση, η πράξη και ο τρόπος διευθυνσιοδότησης περιέχονται στον κωδικό λειτουργίας. Η κωδικοποίηση αυτή είναι αποδοτικότερη όταν υπάρχουν λίγοι τρόποι διευθυνσιοδότησης και λειτουργίες.

Η αντιπαράθεση ανάμεσα στη μεταβλητή κωδικοποίηση και την σταθερή κωδικοποίηση είναι το μέγεθος των προγραμμάτων απέναντι στην εύκολη αποκωδικοποίηση από την κεντρική μονάδα επεξεργασίας. Η σταθερή κωδικοποίηση χρησιμοποιείται από τις μηχανές RISC (π.χ. DLX, MIPS).

Παρακάτω παρουσιάζεται μια σειρά από παραδείγματα κωδικοποίησης εντολών του DLX.

Παράδειγμα

Πώς κωδικοποιούνται οι παρακάτω εντολές του DLX:

Α. LB R1, 50(R2)
Β. ADD R1, R2, R3
Γ. J name

Απάντηση:

Α. LB R1,50(R2)
Με την εντολή αυτή φορτώνουμε στον καταχωρητή R1, το περιεχόμενο της διεύθυνσης μνήμης, η οποία υπολογίζεται προσθέτοντας την τιμή 50 στο περιεχόμενο του καταχωρητή R2.

6	5	5	16 bits
LB	R2	R1	50

Στο πρώτο πεδίο (0..5 bits) βρίσκεται ο κωδικός της εντολής (εντολή φόρτωσης). Στα επόμενα 5 bits(6..10 bits) βρίσκεται ο καταχωρητής R2 (πηγαίος τελεστέος). Στα bits 11..15 βρίσκεται ο καταχωρητής προορισμού R1. Τέλος στα 16 τελευταία bits βρίσκεται η τιμή της μετατόπισης.

Β. ADD R1, R2, R3
Με την εντολή αυτή προσθέτουμε τα περιεχόμενα των καταχωρητών R2 και R3 και το αποτέλεσμα το τοποθετούμε στον καταχωρητή R1.

6	5	5	5	11 bits
opcode	R2	R3	R1	**ADD**

Στο πρώτο πεδίο (0..5 bits) βρίσκεται κωδικοποιημένη η πληροφορία ότι η εντολή είναι μία αριθμητική/λογική εντολή που χρησιμοποιεί μόνο καταχωρητές. Στο 2^{ο} και 3^{ο} πεδίο βρίσκονται οι πηγαίοι τελεστέοι που είναι καταχωρημένοι στους καταχωρητές R2 και R3. Στο 4^{ο} πεδίο (16..20 bits) βρίσκεται ο καταχωρητής προορισμού R1. Στο τελευταίο πεδίο (21..31 bits) βρίσκεται κωδικοποιημένη η πράξη

που στο παράδειγμά μας είναι η πρόσθεση (ADD). Παρατηρούμε ότι ο τρόπος διευθυνσιοδότησης των τελεστέων είναι άμεσος με καταχωρητή.

Γ. J name
Με την εντολή αυτή η ροή του προγράμματος μετατίθεται στη διεύθυνση name.

6	**26** **bits**
J	offset

Στο πεδίο των 26 bits βρίσκεται η τιμή (offset) που θα προστεθεί στον PC ώστε να δώσει το στόχο της μεταπήδησης (name).

Υβριδική κωδικοποίηση

Λειτουργία και αριθμός τελεστέων	Προσδιορι-στής διεύθυνσης #1	Πεδίο διεύθυνσης #1	...	Προσδιορι-στής διεύθυνσης #n	Πεδίο διεύθυνσης #n

Λειτουργία	Προσδιορι-στής διεύθυνσης #1	Προσδιορι-στής διεύθυνσης #2	Πεδίο διεύθυνσης

Λειτουργία	Προσδιορι-στής διεύθυνσης	Πεδίο διεύθυνσης #1	Πεδίο διεύθυνσης #2

Η υβριδική κωδικοποίηση είναι ουσιαστικά ο συνδυασμός των δύο παραπάνω μεθόδων. Μειώνει τη διαφοροποίηση στο μέγεθος και τη δουλειά της μεταβλητής αρχιτεκτονικής, αλλά παρέχει πολλαπλά μήκη εντολών με αποτέλεσμα τη μείωση του μεγέθους του κώδικα.

Η υβριδική κωδικοποίηση χρησιμοποιείται στους υπολογιστές IBM 360/370 και Intel 80x86.

ΔΡΑΣΤΗΡΙΟΤΗΤΑ 13

Να απαριθμήσετε τις μεθόδους κωδικοποίησης των εντολών και να εξηγήσετε τις διαφορές τους. στην περίπτωση που δυσκολεύεστε κρίνεται σκόπιμο να επαναλάβετε την ανάγνωση της προηγούμενης παραγράφου «Κωδικοποίηση συνόλου εντολών».

ΔΡΑΣΤΗΡΙΟΤΗΤΑ 14

- **Με ποιον τρόπο κωδικοποιείται η παρακάτω εντολή του υπολογιστή VAX και ποιο είναι το μέγεθός της σε bytes;**

MOVB B↑ 50(R1),(R2)

Να θεωρήσετε πως ο έμμεσος καταχωρητή τρόπος διευθυνσιοδότησης έχει κωδικό 6 και ο τρόπος διευθυνσιοδότησης με μετατόπιση (B↑) έχει κωδικό 10.

- **Να εξηγήστε την εντολή χρησιμοποιώντας συμβολισμούς.**

ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 14

Με την εντολή αυτή μετακινείται το περιεχόμενο της μνήμης, μεγέθους 1 byte (λόγω του γράμματος B που βρίσκεται κολλημένο στην εντολή MOV_), με διεύθυνση που υπολογίζεται χρησιμοποιώντας τον τρόπο διευθυνσιοδότησης με μετατόπιση (50(R1)) στη διεύθυνση μνήμης που υπολογίζεται χρησιμοποιώντας τον τρόπο διευθ/σης έμμεσο με καταχωρητή. Έτσι, η τιμή που θα μετακινηθεί βρίσκεται στη διεύθυνση, η οποία υπολογίζεται προσθέτοντας την τιμή 50 (μετατόπιση) στο περιεχόμενο του καταχωρητή R1. Το πρόθεμα B↑ σημαίνει ότι η τιμή της μετατόπισης είναι μεγέθους 1 byte. Η διεύθυνση στην οποία θα μετακινηθεί η τιμή αυτή είναι καταχωρημένη στον καταχωρητή R2.

1 byte

MOVB		Opcode
6	R1	Operand Specifier1
50		
10	R2	Operand Specifier2

Όπως παρατηρούμε το μέγεθος της εντολής είναι 4 bytes. Στο 1^{ο} byte βρίσκεται κωδικοποιημένη η πράξη. Στα δύο επόμενα bytes βρίσκεται ο προσδιοριστής τελεστέου του 1^{ου} τελεστέου. Στα 4 πρώτα bits του προσδιοριστή διεύθυνσης βρίσκεται κωδικοποιημένος ο τρόπος διευθυνσιοδότησης του τελεστέου (που στην άσκησή μας είναι 6) και στα επόμενα 4 bits βρίσκεται ο καταχωρητής R1. Στο 2^{ο} byte του προσδιοριστή διεύθυνσης βρίσκεται η μετατόπιση 50. Στο τελευταίο byte της εντολής βρίσκεται ο προσδιοριστής τελεστέου του 2^{ου} τελεστέου.

ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 14 (συνέχεια)

Χρησιμοποιώντας συμβολισμούς, η εντολή εκτελείται ως εξής:

Mem[Regs[R2]] ← Mem[50 + Regs[R1]]

ΔΡΑΣΤΗΡΙΟΤΗΤΑ 15

- **Με ποιον τρόπο κωδικοποιείται η παρακάτω εντολή του VAX και ποιο είναι το μέγεθός της σε bytes;**

ADDW3 (R1), W↑ 400(R2), @ B↑ 200(R3)

Να θεωρήσετε πως ο έμμεσος με καταχωρητή τρόπος διευθυνσιοδότησης έχει κωδικό 6, ο τρόπος διευθυνσιοδότησης με μετατόπιση (W↑) έχει κωδικό 12 και ο έμμεσος με μετατόπιση (B↑) έχει κωδικό 11.

- **Να εξηγήσετε την εντολή χρησιμοποιώντας συμβολισμούς.**

ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 15

Με την εντολή αυτή προστίθενται 2 τελεστέοι μεγέθους 16 bits (λόγω του γράμματος W που βρίσκεται κολλημένο στην εντολή ADD_). Για τον προσδιορισμό του 1ου τελεστέου χρησιμοποιείται ο έμμεσος καταχωρητή τρόπος διευθυνσιοδότησης. Έτσι ο 1ος τελεστέος βρίσκεται στη διεύθυνση μνήμης, η οποία είναι καταχωρημένη στον καταχωρητή R1.

Για τον προσδιορισμό του 2ου τελεστέου χρησιμοποιείται ο τρόπος διευθυνσιοδότησης με μετατόπιση. Έτσι ο 2ος τελεστέος βρίσκεται στη διεύθυνση μνήμης, η οποία υπολογίζεται προσθέτοντας την τιμή 400(μετατόπιση) στο περιεχόμενο του καταχωρητή R2. Το πρόθεμα W↑ σημαίνει ότι η τιμή της μετατόπισης είναι μεγέθους 16 bits (2 bytes).

Για τον προσδιορισμό του τελεστέου προορισμού χρησιμοποιείται ο έμμεσος τρόπος διευθυνσιοδότησης με μετατόπιση. Έτσι ο τελεστέος προορισμού βρίσκεται στη διεύθυνση μνήμης, η οποία υπολογίζεται προσθέτοντας την τιμή 200 (μετατόπιση) στο περιεχόμενο του καταχωρητή R3.

1 byte

ADDW3		Opcode
6	R1	Operand Specifier1
12	R2	Operand Specifier2
400		
11	R3	Operand Specifier3
200		

ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 15 (συνέχεια)

Όπως παρατηρούμε, το μέγεθος της εντολής είναι 7 bytes. Στο 1^{ο} byte βρίσκεται κωδικοποιημένη η πράξη, καθώς και το πλήθος των τελεστέων (3). Στο 2^{ο} byte βρίσκεται ο προσδιοριστής τελεστέου του 1^{ου} τελεστέου. Στα τρία επόμενα bytes βρίσκεται ο προσδιοριστής τελεστέου του 2^{ου} τελεστέου. Επειδή η τιμή της μετατόπισης (400) είναι μεγέθους 16 bits, καταλαμβάνει 2 bytes. Τέλος, στα δύο τελευταία bytes βρίσκεται ο προσδιοριστής τελεστέου του 3^{ου} τελεστέου. Η τιμή της μετατόπισης 200 είναι μεγέθους 1 byte.

Χρησιμοποιώντας συμβολισμούς, η εντολή εκτελείται ως εξής:

Mem[Mem[200 + Regs[R3]]]←Mem[Regs[R1]] + Mem[400+Regs[R2]]

Ανακεφαλαιώνοντας λοιπόν …

Ο βασικός τρόπος ορισμού των τύπων τελεστέων γίνεται με την κωδικοποίηση στον κωδικό λειτουργίας της εντολής, ενώ υπάρχουν 5 διαφορετικοί τύποι τελεστέων που χρησιμοποιούνται στη πλειοψηφία των σημερινών αρχιτεκτονικών.

Το μήκος ενός δεδομένου και η διεύθυνσή του καθορίζουν τον τρόπο που θα προσπελάσουμε το επιθυμητό αντικείμενο στη μνήμη. Υπάρχουν διαφορετικές οργανώσεις των byte μιας λέξης οι οποίες χρησιμοποιούνται για τις αναγκαίες ευθυγραμμίσεις της μνήμης.

Οι αρχιτεκτονικές μπορεί να χρησιμοποιήσουν 10 διαφορετικούς τρόπους διευθυνσιοδότησης, οι οποίοι επηρεάζουν την όλη λειτουργία του υπολογιστή.

Η κωδικοποίηση του συνόλου εντολών του υπολογιστή, καθορίζει το μέγεθος του προγράμματος, την κωδικοποίησή τους σε δυαδική μορφή και την εκτέλεση αυτού από την ΚΜΕ.

Οι βασικοί τρόποι κωδικοποίησης του συνόλου εντολών είναι: σταθερή, μεταβλητή και υβριδική κωδικοποίηση.