

## ΕΝΤΟΛΕΣ ASSEMBLY ΓΙΑ MIPS ΑΡΧΙΤΕΚΤΟΝΙΚΗ

- Αριθμητικές Εντολές

add ⇒ Πρόσθεση signed αριθμών

addi ⇒ Πρόσθεση signed αριθμού με σταθερά (immediate)

sub ⇒ Αφαίρεση signed αριθμών

mul ⇒ Πολλαπλασιάζει καταχωρητές 32-bit αλλά κόβει τα 32 LSB

mult ⇒ Πολλαπλασιάζει καταχωρητές 32-bit και χωρίζει σε HI και LO τα MSB και LSB αντίστοιχα. 32-bit το καθένα

div ⇒ signed διαίρεση η οποία χωρίζει σε HI και LO τα MSB και LSB αντίστοιχα. 32-bit το καθένα

divu ⇒ unsigned διαίρεση η οποία χωρίζει σε HI και LO τα MSB και LSB αντίστοιχα. 32-bit το καθένα

mflo ⇒ Αντιγράφει τα 32 LSB του LO σε έναν καταχωρητή για mult και div

mfhi ⇒ Αντιγράφει τα 32 MSB του LO σε έναν καταχωρητή για mult και div

- Εντολές Μεταφοράς Δεδομένων

lw ⇒ Φορτώνει μία λέξη (4 bytes) από τη μνήμη σε καταχωρητή

sw ⇒ Αποθηκεύει μία λέξη (4 bytes) από καταχωρητή στη μνήμη

lb ⇒ Φορτώνει ένα byte από τη μνήμη σε καταχωρητή (signed)

lbu ⇒ Φορτώνει ένα byte από τη μνήμη σε καταχωρητή (unsigned)

sb ⇒ Αποθηκεύει ένα byte από καταχωρητή στη μνήμη

lh ⇒ Φορτώνει μισή λέξη (2 bytes) από τη μνήμη (signed)

lhu ⇒ Φορτώνει μισή λέξη (2 bytes) από τη μνήμη (unsigned)

sh ⇒ Αποθηκεύει μισή λέξη (2 bytes) από καταχωρητή στη μνήμη

- Εντολές Διακλάδωσης

beq  $\Rightarrow$  Αλλαγή ροής αν δύο καταχωρητές είναι ίσοι

bne  $\Rightarrow$  Αλλαγή ροής αν δύο καταχωρητές είναι διάφοροι

- Λογικές Εντολές

sll  $\Rightarrow$  Λογική ολίσθηση προς τα αριστερά

srl  $\Rightarrow$  Λογική ολίσθηση προς τα δεξιά (unsigned)

slt  $\Rightarrow$  Θέτει 1 τον πρώτο καταχωρητή, αν ο δεύτερος είναι μικρότερος από τον τρίτο

slti  $\Rightarrow$  Όπως slt, αλλά συγκρίνει με σταθερά (immediate)

sltu  $\Rightarrow$  Όπως slt, αλλά χωρίς πρόσημο

sltui  $\Rightarrow$  Όπως slti, αλλά για unsigned αριθμούς

and  $\Rightarrow$  Λογικό AND μεταξύ δύο καταχωρητών

andi  $\Rightarrow$  Λογικό AND με σταθερά (immediate)

or  $\Rightarrow$  Λογικό OR μεταξύ δύο καταχωρητών

ori  $\Rightarrow$  Λογικό OR με σταθερά

nor  $\Rightarrow$  Λογικό NOR

- Εντολές "Άλματος"

jal  $\Rightarrow$  Μεταπήδηση σε υπορουτίνα (function call)

jr  $\Rightarrow$  Μεταπήδηση στη διεύθυνση που περιέχει καταχωρητής

- Λοιπές

lui  $\Rightarrow$  Φορτώνει την 16-bit σταθερά, στα αριστερά 16 bits (bits 31–16) ενός καταχωρητή, θέτοντας τα δεξιά 16 bits (bits 15–0) ίσα με μηδέν

ori  $\Rightarrow$  Φορτώνει τα δεξιά bits (0-15) με έναν άλλον αριθμό

## ΕΝΤΟΛΕΣ ΜΑΘΟΔΟΥ ΚΙΝΗΤΗΣ ΥΠΟΔΙΑΣΤΟΛΗΣ

- Κάθε εντολή πρέπει να τελειώνει με μια τελεία και ένα γράμμα που καθορίζει το μέγεθος των δεδομένων:

.s (single precision): Για float (32-bit)

.d (double precision): Για double (64-bit)

Χρησιμοποιούν ξεχωριστούς καταχωρητές (\$f0 - \$f31)

- Εντολές Μεταφοράς Δεδομένων

lwc1 ⇒ Φορτώνει έναν float από τη μνήμη

swc1 ⇒ Αποθηκεύει έναν float στην μνήμη

ldc1 ⇒ Φορτώνει δύο λέξεις ταυτόχρονα

- Αριθμητικές Εντολές

add.s

sub.s

mul.s

div.s

sqrt.s

abs.s

neg.s

## ΧΡΗΣΗ ΚΑΤΑΧΩΡΗΤΩΝ (32-Bit)

Αριθμός Καταχωρητή: 0  $\Rightarrow$  \$zero  $\Rightarrow$  constant value = 0

Αριθμός Καταχωρητή: 2-3  $\Rightarrow$  \$v0-\$v1  $\Rightarrow$  τιμές αποτελεσμάτων (από συναρτήσεις)

Αριθμός Καταχωρητή: 4-7  $\Rightarrow$  \$a0-\$a3  $\Rightarrow$  ορίσματα συναρτήσεων

Αριθμός Καταχωρητή: 8-15  $\Rightarrow$  \$t0-\$t7  $\Rightarrow$  προσωρινοί καταχωρητές

Αριθμός Καταχωρητή: 16-23  $\Rightarrow$  \$s0-\$s7  $\Rightarrow$  αποθηκευμένοι καταχωρητές

Αριθμός Καταχωρητή: 24-25  $\Rightarrow$  \$t8-\$t9  $\Rightarrow$  προσωρινοί καταχωρητές

Αριθμός Καταχωρητή: 28  $\Rightarrow$  \$gp  $\Rightarrow$  global pointer

Αριθμός Καταχωρητή: 29  $\Rightarrow$  \$sp  $\Rightarrow$  stack pointer

Αριθμός Καταχωρητή: 30  $\Rightarrow$  \$fp  $\Rightarrow$  frame pointer

Αριθμός Καταχωρητή: 32  $\Rightarrow$  \$ra  $\Rightarrow$  return pointer

## ΔΙΑΤΑΞΗ ΜΝΗΜΗΣ

Είναι ο τρόπος με τον οποίο είναι οργανωμένη η φυσική ή η εικονική μνήμη, κατά την εκτέλεση ενός κώδικα MIPS Assembly.

Περιγραφή τμημάτων:

- .text  $\Rightarrow$  κώδικας
- .data  $\Rightarrow$  αρχικοποιημένες σταθερές και μεταβλητές
- .bss  $\Rightarrow$  μη αρχικοποιημένες μεταβλητές
- Heap  $\Rightarrow$  δυναμική μνήμη
- Stack  $\Rightarrow$  τοπικές μεταβλητές και επιστροφές συναρτήσεων

## ΔΙΑΤΑΞΗ ΚΩΔΙΚΑ

Είναι ο τρόπος με τον οποίο δομείται και εν τέλη, γράφεται ο κώδικας MIPS για το πρόγραμμα. Η διάταξη του κώδικα, αντιστοιχεί στην διάταξη της μνήμης.

Περιγραφή τμημάτων:

- .data
- .text
- .globl main
- main
- ... κάθε άλλη υπορουτίνα/συνάρτηση του προγράμματος...

## ΜΟΡΦΗ ΕΝΤΟΛΩΝ R, I και J

- R - type

|        |    |    |    |       |       |
|--------|----|----|----|-------|-------|
| opcode | rs | rt | rd | shamt | funct |
|--------|----|----|----|-------|-------|

Οι R - τύπου εντολές, είναι αυτές που εκτελούνται μεταξύ καταχωρητών

opcode  $\Rightarrow$  κωδικός λειτουργίας  $\Rightarrow$  6 bits

rs  $\Rightarrow$  καταχωρητής πηγής 1  $\Rightarrow$  5 bits

rt  $\Rightarrow$  καταχωρητής πηγής 2  $\Rightarrow$  5 bits

rd  $\Rightarrow$  καταχωρητής προορισμού  $\Rightarrow$  5 bits

shamt  $\Rightarrow$  shift amount  $\Rightarrow$  5 bits

funct  $\Rightarrow$  όνομα πράξης  $\Rightarrow$  6 bits

- I - type

|        |    |    |     |
|--------|----|----|-----|
| opcode | rs | rt | imm |
|--------|----|----|-----|

Οι I - τύπου εντολές είναι αυτές εμπλέκουν και μία σταθερά (immediate) εκτός από καταχωρητές

opcode  $\Rightarrow$  κωδικός λειτουργίας  $\Rightarrow$  6 bits

rs  $\Rightarrow$  καταχωρητής πηγής 1  $\Rightarrow$  5 bits

rt  $\Rightarrow$  καταχωρητής πηγής 2  $\Rightarrow$  5 bits

imm  $\Rightarrow$  σταθερά ή offset  $\Rightarrow$  16 bits

- J - type

| opcode | addr |
|--------|------|
|--------|------|

Οι J - τύπου εντολές, είναι αυτές που εκτελούνται για μεταπήδηση σε μία συγκεκριμένη διεύθυνση στην οποία βρίσκεται μία άλλη εντολή του προγράμματος

opcode  $\Rightarrow$  κωδικός λειτουργίας  $\Rightarrow$  6 bits

address  $\Rightarrow$  διεύθυνση μεταπήδησης  $\Rightarrow$  26 bits

## MIPS MEMORY MAP

Στην αρχιτεκτονική MIPS, ο address space είναι χωρισμένος σε συγκεκριμένες περιοχές.

- Περιοχή 1: Reserved

**Εύρος:** 0x00000000 έως 0x003FFFFFFF

Αυτή η περιοχή δεν χρησιμοποιείται από τον προγραμματιστή, καθώς είναι δεσμευμένη για το σύστημα

- Περιοχή 2: Text Segment

**Έναρξη:** 0x00400000

Εδώ αποθηκεύονται οι εντολές του προγράμματός

Ο Program Counter ξεκινάει από εδώ

- Περιοχή 3: Static Data

**Έναρξη:** 0x10000000

Εδώ μπαίνουν οι καθολικές μεταβλητές και οι σταθερές που ξέρουμε το μέγεθός τους, πριν τρέξει το πρόγραμμα

- Περιοχή 4: Heap (Σωρός)

**Θέση:** Ξεκινάει αμέσως μετά τα Static Data και μεγαλώνει προς μεγαλύτερες διευθύνσεις

Χρησιμοποιείται για δυναμική δέσμευση μνήμης

- Περιοχή 5: Stack

**Έναρξη:** 0x7FFFFFFF

Εδώ αποθηκεύονται οι τοπικές μεταβλητές συναρτήσεων και οι διευθύνσεις επιστροφής

Η στοίβα μεγαλώνει προς μικρότερες διευθύνσεις

Ο Stack Pointer δείχνει πάντα στην κορυφή της στοίβας

- Περιοχή 6: Kernel (Πυρήνας) – 0x8000...

**Έναρξη:** 0x80000000

Αυτός είναι ο χώρος του Λειτουργικού Συστήματος

Ο χρήστης δεν έχει πρόσβαση εδώ. Οποιαδήποτε προσπάθεια ανάγνωσης από εδώ, οδηγεί σε σφάλμα (Exception)