Assembler Directives in Mips

Directives (Οδηγίες) Assembler στη MIPS είναι οδηγίες που βοηθούν τον assembler να οργανώσει τα δεδομένα, να ορίσει θέσεις μνήμης και να ελέγξει τη διαδικασία συναρμολόγησης (the assembly process).
Χρησιμοποιούνται για τη ρύθμιση του προγράμματος, καθοδηγούν τον assembler αλλά δεν παράγουν πραγματικές οδηγίες CPU*.
Χρησιμοποιούνται για:

1. Καθορισμό δεδομένων- Defining data (π.χ. μεταβλητές, συμβολοσειρές, πίνακες),
2. οργάνωση ενοτήτων - Organizing sections (π.χ. κώδικας, δεδομένα, στοίβα),
3. ρύθμιση διευθύνσεων μνήμης - Setting memory addresses και
4. μακροορισμοί- Macro definitions

*Στην αρχιτεκτονική MIPS, η CPU (Central Processing Unit) είναι υπεύθυνη για την εκτέλεση εντολών, τη διαχείριση καταχωρητών, το χειρισμό της πρόσβασης στη μνήμη (μόνο οι “εντολές” φόρτωσης/αποθήκευσης έχουν πρόσβαση στη μνήμη) και την εκτέλεση αριθμητικών και λογικών πράξεων.

Directive	Purpose	Example
`.text`	Marks the start of the code section	`.text`
`.data`	Marks the start of the data section	`.data`
`.globl`	Declares a global label (entry point)	`.globl main`
`.word`	Reserves memory for an integer (32-bit)	`var: .word 5`
`.byte`	Reserves memory for a byte (8-bit)	`flag: .byte 1`
`.half`	Reserves memory for a half-word (16-bit)	`num: .half 12`
`.asciiz`	Stores a **null-terminated** string	`str: .asciiz "Hello"`
`.ascii`	Stores a **string without null termination**	`text: .ascii "Hello"`
`.space`	Allocates uninitialized space	`buffer: .space 20`
`.align`	Aligns data in memory (power of 2)	`.align 2` (aligns to 4 bytes)

Παράδειγμα Φορτώνει και εκτυπώνει μια συμβολοσειρά και έναν ακέραιο.

```
.data
message: .asciiz "Hello\n"   # Define a string
value:   .word 42                    # Define an integer

.text
.globl main                          # Define main as a global label

main:
        la $a0, message              # Load address of message
        li $v0, 4                    # System call: Print string
        syscall                      # Print the message

        lw $t0, value                # Load value from memory
        li $v0, 1                    # System call: Print integer
        move $a0, $t0                # Move value to print argument
        syscall                      # Print the integer

        li $v0, 10                   # Exit system call
        syscall
```

Βιβλιογραφία: MIPS Assembly Language using QtSpim Ed Jorgensen Version 1.0 January 2013