

# Assembleur Groupe G (minias)

Pierre-Alexandre Bazin, Mathis Bouverot & Arthur Rousseau

## 1 Utilisation

Pour compiler minias : `make`.

Pour assembler un fichier `file.ms` (sortie dans `output.txt`): `./minias file.ms -write-rom output.txt`.

## 2 Syntaxe

Les commentaires commencent par `;` et s'étendent jusqu'à la fin de la ligne.

Un programme est une suite de sections.

Une section commence par `.section name` où `name` est soit `text` soit `data`, et est une suite d'instructions et de labels.

Exemple de programme :

```
.section data
x: .space 2
.section text
add %rbx %rbx 42
mov y %rbx
.section data
y:
```

## 3 Registres

Il y a 7 registres minias nommés `rax`, `rbx`, `rcx`, `rdx`, `rex`, `rfx`, `rgx`.

Ces registres sont sur 16 bits.

Le registre ISA `rz` n'est pas accessible au programmeur.

Pour accéder à l'adresse  $x$  de la ram, en supposant  $x$  dans un registre, il faut écrire le nom du registre entre parenthèses.

Exemple : `mov (%rax) %rbx`

## 4 Instructions

Les instructions ne sont pas exactement celles de l'ISA.

Les instructions ISA `addi`, `subi`, etc. n'existent pas dans minias (utiliser `add`, `sub`, etc. à la place).

Les instructions ISA `lw`, `sw` n'existent pas dans minias (utiliser `mov` à la place).

L'instruction minias `mov` est traduite en différentes instructions ISA (`lw`, `sw`, `addi`, etc.) en fonction des arguments.

Syntaxe des instructions :

- add, sub, or, nand, xor, nxor, and, nor, sll, srl, sra, seq, slt :

`instr dest source1 source2`

- `instr` : nom de l'instruction
- `dest` : registre
- `source1` : registre
- `source2` : registre ou immediate

Exemple : `add %rax %rbx 42` (fait `rax ← rbx + 42`).

- beq, bne, ble, blt, bge, bgt :

`instr arg1 arg2 label`

- `instr` : nom de l'instruction
- `arg1` : registre
- `arg2` : registre
- `label` : chaîne de caractères

Exemple : `beq %rax %rbx .L0`

- mov :

`mov dest source`

- `dest` : registre ou mémoire
- `source` : registre, mémoire ou immediate

Si `dest` ou `source` est un emplacement mémoire, le deuxième doit être un registre.

- jmp :

`jmp addr`

- `addr` : registre ou label

- jal :

`jal rsave addr`

- `addr` : registre ou label
- `rsave` : registre

Sauvegarde PC+4 dans `rsave`, puis jump vers `addr`.

- .space :

`.space n`

- `n` : entier positif

Peut se trouver uniquement dans la section data. Permet de 'sauter' des adresses. Exemple :

```
.section data
.space 2
x:
```

Ici `x` est à l'adresse 2 dans la RAM.

## 5 Labels

Un label est une chaîne de caractères comprenant des chiffres, des underscores, des points et au moins une lettre (minuscule ou majuscule).

Un label peut apparaître avant une instruction dans la section text, ou n'importe où dans la section data, et est suivi de deux points.