

# A Linguagem de Máquina: Operações Básicas

## Arquitetura de Computadores

Charles Tim Batista Garrocho

Instituto Federal do Paraná – IFPR  
Campus Goioerê

`charles.garrocho.com/AC2016`

`charles.garrocho@ifpr.edu.br`

Técnico em Informática



INSTITUTO FEDERAL

# Linguagem de Montagem Assembly

- Todas as instruções aritméticas e lógicas com três operandos.
- A ordem dos operandos é fixa (destino primeiro).

```
[label:] [operando], [operando], [operando] [# comentario]
```

Sintaxe de instruções assembly:

- 1 Label: opcional, identifica bloco do programa.
- 2 Operandos: Registradores ou memória.
- 3 Comentários: opcional, tudo que vem depois do #.



INSTITUTO FEDERAL

# Linguagem de Montagem Assembly do MIPS

## Exemplo Simples de Compilação.

Some **B** com **C** e coloque o resultado em **A**.

Instrução de Linguagem de Montagem:

```
add $s0, $s1, $s2
```

Equivalente ao comando em Linguagem Python:

```
a = b + c
```



INSTITUTO FEDERAL

# Linguagem de Montagem Assembly do MIPS

## Exemplo Complexo de Compilação.

Uma instrução um tanto complexa contém as cinco variáveis **f**, **g**, **h**, **i** e **j**:

$$f = (g + h) - i$$

A primeira instrução MIPS calcula a soma de **g** e **h**. Temos de colocar o resultado em algum lugar, de modo que o compilador crie uma variável temporária, chamada **t0**:

```
add $t0, $s0, $s1 # Variável temporária t0 contém g + h
```

Finalmente, a instrução de subtração subtrai **i** da primeira soma e coloca a diferença na variável **s3**, completando o código compilado:

```
sub $s3, $s2, $t0 # s3 recebe t0 - s2, que é (g + h) - i
```



INSTITUTO FEDERAL

Em **Assembly**, estamos manipulando **registradores** do MIPS.

Em **código Python** (sem compilação), estamos manipulando **posições da memória**.

A associação entre posições da memória e registradores é realizada pelo **compilador** Python.



# 1º Princípio de Projeto MIPS

## Simplicidade favorece regularidade.

Mais que três operandos por instrução exigiria um projeto de hardware mais complicado.

**Código 1 Python:** `A = B + C + D`

**Código 1 MIPS:** `add $t0, $s1, $s2 #t0 = s1 + s2`

*Continuação 1:* `add $t0, $t0, $s3 #t0 = t0 + s3`

**Código 2 Python:** `E = F - A`

**Código 2 MIPS:** `sub $s0, $s4, $t0 #s0 = s4 - t0`



INSTITUTO FEDERAL

## **Menor Significa mais rápido.**

Uma quantidade maior que 32 registradores exigiria:

- Um ciclo de clock maior.
- Formato de instruções maior, para comportar mais bits de endereçamento.



Traduza para o assembly do MIPS os códigos Python a seguir e calcule a quantidade de variáveis e quantidade de instruções necessárias em MIPS:

- $a = (c + (g - h)) + ((k + l) - t)$
- $p = h - b$   
 $k = (a + (h - (p - g))) - ((k + l) - t)$
- $g = (a + b) - d$   
 $d = (g - h)$   
 $e = (g - d) - (f + w) - b$

