

# X86-64

## Diretivas *assembly*

Programação em Sistemas Computacionais

João Pedro Patriarca ([jpatri@cc.isel.ipl.pt](mailto:jpatri@cc.isel.ipl.pt), [joao.patriarca@isel.pt](mailto:joao.patriarca@isel.pt)), Gabinete F.0.23 do edifício F

ISEL, ADEETC, LEIC

# Agenda

---

- Considerações sobre um ficheiro em *assembly*
- Diretivas que definem secções lógicas
- Diretivas que contribuem para a alocação de espaço
- Outras diretivas

# Considerações sobre um ficheiro em *assembly*

---

- A extensão de um ficheiro em *assembly* em ambiente Linux é `.s`
- O compilador de *assembly* da GNU é a ferramenta `as`. No entanto pode usar-se a ferramenta `gcc` desde que o ficheiro tenha extensão `.s`
- Comentários:
  - `# comentário de linha`
  - `/* comentário de bloco */`
- O conteúdo de um ficheiro em *assembly* deve ser organizado em três colunas

1ª coluna	2ª coluna	3ª coluna
	diretiva do compilador	operandos da diretiva
nome/símbolo definido pelo programador (ex: nome de funções, labels, ...) terminada pelo caracter ':'		
	mnemónica da instrução	operandos da instrução

# Secções lógicas

---

- As secções lógicas dividem o programa em áreas com diferentes objetivos
- O GCC usa o formato ELF (*Executable and Linkable Format*) para produzir o resultado da compilação
- Genericamente uma secção é definida com a diretiva  
`.section <section-name>`
- Exemplos de algumas secções com significados pré-estabelecidos

Secção	Descrição
<code>.text</code>	Código executável (contribui para a imagem do programa em memória)
<code>.data</code>	Variáveis globais iniciadas (contribui para a imagem do programa em memória)
<code>.bss</code>	Variáveis globais não iniciadas (contribui para a imagem do programa em memória e o espaço correspondente é iniciado com 0)
<code>.section .rodata</code>	Dados constantes e imutáveis (contribui para a imagem do programa em memória. Ex: <i>strings</i> , literais)
<code>.section .debug</code>	Informação simbólica para <i>debug</i> (quando se compila com a opção <code>-g</code> )

# Diretivas que alocam espaço

Diretiva		Secção	Descrição
<i>.align</i>	<i>exp1, exp2</i>	<i>.bss, .data, .rodata</i>	Alinha o próximo dado num endereço múltiplo de <i>exp1</i> e, se <i>exp2</i> presente, preenche com o byte <i>exp2</i> . Se <i>exp2</i> omissa, por norma, preenche com 0
<i>.ascii</i>	<i>"string"</i>	<i>.data, .rodata</i>	Zero ou mais <i>strings</i> separadas por ‘,’
<i>.asciz</i>	<i>"string"</i>	<i>.data, .rodata</i>	Zero ou mais <i>strings</i> terminadas por 0 separadas por ‘,’
<i>.byte</i>	<i>expressions</i>	<i>.data, .rodata</i>	Zero ou mais expressões codificadas em 8 bits
<i>.long/.int</i>	<i>expressions</i>	<i>.data, .rodata</i>	Zero ou mais expressões codificadas em 32 bits
<i>.quad</i>	<i>expressions</i>	<i>.data, .rodata</i>	Zero ou mais expressões codificadas em 64 bits
<i>.space</i>	<i>size, fill</i>	<i>.bss</i>	Aloca <i>size</i> bytes iniciados, opcionalmente, pelo valor <i>fill</i> . Se <i>fill</i> omissa, por norma, preenche com 0
<i>.word</i>	<i>expressions</i>	<i>.data, .rodata</i>	Zero ou mais expressões codificadas em 16 bits

# Outras diretivas

---

Diretiva		Descrição
<i>.equ/.set</i>	<i>symbol, exp</i>	Define símbolo <i>symbol</i> associando-lhe o valor <i>exp</i>
<i>.globl/.global</i>	<i>symbols</i>	Torna a sequência de símbolos separados por ‘,’ públicos (visível para o <i>linker</i> ). Por omissão, os símbolos são privados ao módulo
<i>.end</i>		Última diretiva de um módulo em <i>assembly</i>

# Exemplo de um ficheiro em *assembly*

```
.globl    f0, f1    # Os símbolos f0 e f1 ficam públicos e visíveis externamente
.text     # Secção com código
f0:       # Function f0 code
f1:       # Function f1 code

.data     # Secção com variáveis globais iniciadas
var1:     .byte     1, 2, 3    # Variável var1 com uma sequência de 3 bytes
        .align     4          # Alinha var2 num endereço múltiplo de 4 (introduz 1 byte)
var2:     .long     1, 2      # Variável var2 com uma sequência de 2 bytes
        .align     8          # Alinha var3 num endereço múltiplo de 8 (introduz 4 bytes)
var3:     .quad     0xABC     # Variável 3 com o valor 0xABC
        .bss        # Secção com variáveis globais não iniciadas
var4:     .space    16        # Variável var4 com uma sequência de 16 bytes
        .align     8          # Alinha var5 num endereço múltiplo de 8 (introduz 0 bytes)
var5:     .space    8         # Variável var5 com uma sequência de 8 bytes
        .end
```