

Arquitetura de Computadores 2017-18

Ficha 4

Tópicos: *Introdução ao assembly.*

1. Código hello.s

Considere o seguinte exemplo de programa em *assembly* Intel/Linux semelhante ao visto na aula teórica 8 (e ficheiro no CLIP). Relembre o que este faz.

```
EXIT = 1          # usando simbolos para constantes
WRITE = 4
LINUX_SYSCALL = 0x80

.data             # secção de dados (variaveis)
msg: .ascii      "Hello, world!\n" # um vetor de caracteres
msglen = . - msg # msglen representa o tamanho do vetor

.text            # secção de código
.global _start  # exportar o simbolo _start (inicio do programa)
_start: movl    $msglen,%edx
        movl    $msg,%ecx
        movl    $1,%ebx
        movl    $WRITE,%eax # pedir write ao sistema
        int     $LINUX_SYSCALL # chama o sistema

        movl    $0,%ebx
        movl    $EXIT,%eax # pedir o exit ao sistema
        int     $LINUX_SYSCALL # chama o sistema
```

a) Obtenha o executável respetivo e teste a sua execução. Para tal *assemble* e ligue para obter o executável usando a sequencia de comandos:

```
as -o hello.o hello.s
ld -o hello hello.o
```

Execute depois o programa `hello`.

b) Execute em seguida sob o controlo do *debugger*, executando passo a passo. Lembre-se que para o *debugger* mostrar o seu código fonte, tal como o escreveu, deve usar a opção `-g` no `as`. Exemplo:

```
as -g -o hello.o hello.s
ld -o hello hello.o
```

Coloque um *breakpoint* (por exemplo logo em `_start`) e use o comando *“layout regs”* para que seja mostrado o estado dos registos do CPU enquanto executa o programa passo a passo. Confirme o efeito de cada `movl` nos registos do CPU. Pode imprimir o valor das variáveis com o comando *print*, como antes, mas podemos ter de forçar a interpretação correta. Exemplo: `print (char[14])msg .`

c) Observe agora o código presente no executável usando o comando `objdump`. Se o seu programa se chamar *“hello”* use o seguinte comando:

```
objdump -d hello
```

Procure na listagem produzida o código a seguir ao símbolo `_start`. Deve verificar que aparece a representação hexadecimal do código máquina e o seu significado em *assembly*, que deve ser idêntico ao código fonte.

d) Repita a experiência de *disassembly* da alínea anterior, mas com um outro executável qualquer, por exemplo um programa seu de uma aula passada (desenvolvido na linguagem C). Procure a função `main` e verifique o respetivo código produzido pelo compilador. Note que existe mais código, colocado pelo compilador e *linker*, necessário para o seu programa iniciar a execução pela função `main` com os respetivos argumentos, assim como para a chamada das funções da biblioteca do C.