

Sistemas de Computação

Modelo do 1º Teste

- O Centro de Informática de uma escola dispõe de 113_4 (base 4) computadores portáteis e 74_8 (octal) *desktops*. **Calcule** o total de computadores disponíveis, em **binário** (com 8 bits) e em **hexadecimal**. Apresente todos os cálculos que efetuar.
- Considere as seguintes sequências de **6 bits**, cujos valores em octal são 76_8 e 24_8 . Cada um desses valores representa um inteiro em complemento para 2. **Adicione** os dois valores e apresente o resultado em **octal**, mostrando os cálculos que fizer.
- Considere um formato de representação de números em vírgula flutuante, baseado na recomendação IEEE 754, com 12 bits: 1 bit para sinal, 5 bits para o expoente e 6 bits para a parte fracionária.
 - Calcule** o valor na base 10 que corresponde ao valor $0x49a$ escrito naquele formato, sabendo que o valor de um número normalizado neste formato vem dado por $V_n = (-1)^S * 1.F_2 * 2^{E-\text{excesso}}$ (Nota: o valor do excesso é um nº ímpar). Apresente todos os cálculos que efetuar.
 - Represente neste formato o valor $-17.625 * 512$, apresentando todos os cálculos que fizer.
- Um sistema de computação com um processador IA-16 (*little endian*) acaba de fazer o *fetch* e a decodificação da instrução `addw 6(%bp), %ax` e vai iniciar a sua execução. O seu estado (parcial) é descrito por:
 - conteúdo de alguns registos:


```
%ip = 0x300e
%ax = 0x0010
%sp = 0x0100
%bp = 0xa8a6
```
 - conteúdo de algumas células de memória, em hexadecimal, a partir de cada um dos endereços:


```
0x3008: 05 00 ef ff 01 1a 24 1f
0x3010: 1a 02 5f 17 ab 23 7f 2b
...
0xa8a8: 43 f8 ff 01 43 2b 45 23
0xa8b0: ef ff 01 98 23 e5 c3 23
```

Mostre, em hexadecimal, o bloco de informação que circula nos barramentos de endereços e de dados.
- Considere o estado parcial de um PC com um processador IA-16 (*little endian*), ilustrado em baixo, com um excerto de código duma função em execução e uma visão da memória.

Registos	Memória (código)	Memória (dados)
%ip = 0x4050	...	0x6ffc: 0x50 0x00
%bp = 0x7000	0x404f inc %cx	0x6ffe: 0x10 0x70
%ax = 0x2311	0x4050 pushw %ax	0x7000: 0xb5 0x40
%cx = 0x0011	0x4051 addw %ax, 4(%bp)	0x7002: 0x20 0x00
%sp = 0x6ffe	0x4054 popw %ax	0x7004: 0x11 0x00
	0x4055 ...	0x7006: 0x01 0x00
		0x7008: 0xff 0x00

Após a execução das próximas 2 instruções **indique** o conteúdo dos seguintes locais no PC:

- Nos registos `%ip` e `%sp`
- Nas células de memória nos seguintes endereços: `0x7003` e `0x7004`

6. Considere o seguinte excerto de código *assembly* do IA-32:

```
leal 4(%eax), %ebx
addl (%eax), %ebx
```

Comente, justificadamente, a veracidade das seguintes afirmações:

- durante a execução de ambas instruções ocorrem 2 acessos à memória (sem contabilizar o *fetch* das instruções em memória),
- no contexto deste excerto de código *assembly*, a instrução `leal` pode ser substituída por um `movl` sem modificar o valor final do registo `%ebx`.

7. Considere o seguinte código C e parte do *assembly* correspondente para IA-32:

Função C	Assembly
<pre>int soma_e_mult (int a, int b) { return (a + b) * b; }</pre>	<pre>... add %edx,%eax imul %edx,%eax ...</pre>

Comente, justificadamente, a veracidade das seguintes afirmações:

- antes da instrução `add`, o registo `%eax` contém o operando `b`
- são feitas 2 escritas no registo `%eax`.

8. Considere o seguinte excerto de código que resultou da compilação para IA-32 de um programa imperativo escrito na linguagem C:

Assembly			
8048331:	b8 00 00 00 00	mov	\$0x0,%eax
8048336:	ba 00 00 00 00	mov	\$0x0,%edx
804833b:	83 c0 05	add	\$0x5,%eax
804833e:	42	inc	%edx
804833f:	83 fa 03	cmp	\$0x3,%edx
8048342:	7e f7	jle	804833b

Identifique uma possível estrutura de controlo C que tenha originado este código.