# MC404

- **4 - 6 questões**

1. Manipulação de bits, overflow, número binário, número hexadecimal
2. Códigos escritos conforme ABI
3. Informações técnicas sobre interrupções
    4. Registradores calle-saved, caller-saved, outros

# Revisão - MC404

- IC: identifica quem interrompeu (interrupção externa)
    - Rever máscara
    - ABI: alinhamento de 16 bytes quando for pilha, se for word, 4 bytes, se for short, 2 bytes...
    - Preciso alinhar a seção .text (.align 4) para deixar as rotinas alinhadas
    - Preciso alinhar a seção .data respeitando as variáveis adicionadas
    - Sempre alocar sp depois de fp. Ou seja, sp = n×4 (sp) e fp = (n-1) (sp)

# Questões - MC404

**Questão 2.** Um programador começou a traduzir o código abaixo, escrito em C, para a linguagem de montagem do ARM. Você deve finalizar a tradução do código inserindo a instrução que falta no espaço em branco. O código deve seguir a ABI do ARM, utilizada durante o curso.

```
short misterious (short* v, int n) {
  return v[n];
}

misterious:
  ----------------------------------
  mov pc, lr
```

- **Resolução:** o endereço que gostaríamos de encontrar é o v[n] = v + n*2

```
misterios:

    slli a1, a1, 1 # Multiplicação por 2, sendo a1 = n
    add a1, a0, a1 # a0 = endereço de início do vetor, então se eu somo v[0] com n*2, tenho o endereço
    lh a0, 0(a1) # Carrego o valor short encontrado em a1 em a0, já que é o registrador de retorno
    ret # Encerra
```

**Questão 6.** Responda às perguntas abaixo:

a) Quais os registradores salvos automaticamente pelo *hardware* quando ocorre uma interrupção?

PC (guarda o endereço de memória da próxima instrução, MIE (habilita ou desabilita o tratamento de interrupções)

b) Suponha que a pilha do programa é descendente-cheia, mostre como empilhar o valor no registrador `R0` na pilha do programa com uma única instrução `STR`.

addi sp, sp, -4
sw a0, 0(sp)

c) Qual instrução do ARM gera uma *trap*?

ecall (interrupção do software)

d) Onde o endereço de PC é salvo no momento em que uma interrupção IRQ acontece?

MEPC: salva o valor de PC antes de setá-lo para o endereço de ISR

**Questão 7.** Traduza o program abaixo, em C, para a linguagem de montagem do ARM. Seu código deve seguir a ABI do ARM, vista durante o curso, e utilizar apenas registradores *caller-save*. Seu código deve conter comentários indicando quais instruções e diretivas são utilizadas para: a) alocação e inicialização de variáveis globais; b) leitura de variáveis globais; c) alocação, desalocação e inicialização de variáveis locais; e d) passagem de parâmetros.
Programa:

```
int y = 2653;
int foo()
{
    short x = 32;
    return bar(&x);
}
```

- **Resolução:**

.section .data # Nós vamos usar a seção .data, já que os dados já estão inicializados
.global y
.align 2 # Nesse caso, estou alinhando 2, já que será 2^2. Assim, o PC vai para o próximo múltiplo de 4
y: .word 2653 # Variável já estava definida
.text
.align 2 # Sempre alinho a rotina por 4. Apenas 1 desses no começo se tiver só instruções no arquivo
.global foo
foo:

li t0, 32 # Carrega o valor 32 na variável t0

addi sp, sp -16 # Preciso alocar espaço na pilha
sw ra, 12(sp) # Como o ra será o último acessado, ele precisa ser o primeiro alocado
sw fp, 8(sp)
sh t0, 4(sp) # Aloca o valor de x e, mesmo sendo short, eu aloco em 4

```
addi fp, sp, 16 # Atualiza o valor do framepointer

addi a0, sp, 6 # Carrega o endereço de x, que será sp + 6

jal bar # Chama a função bar

lw ra, 12(sp) # Carrega o endereço de retorno
lw fp, 8(sp) # Carrega o framepointer
addi sp, sp, 16 # Desempilha

ret
```