Arquitetura de Computadores

ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

– Exame da Época Normal –

14 de Janeiro de 2022

Duração: 90 minutos + 30 minutos de tolerância

Nome:_____ **Número:**_____

Notas Importantes:

Durante a prova pode consultar a bibliografia da disciplina (slides, livros, enunciados e material de apoio a trabalhos práticos). No entanto, não é permitido o uso de computadores ou qualquer outro dispositivo electrónico.

Este é um teste de escolha múltipla e deverá assinalar sem ambiguidades as respostas na tabela apresentada a baixo. Cada pergunta corretamente respondida vale cinco pontos; cada resposta errada desconta dois pontos; e cada pergunta não respondida vale zero pontos. Uma nota final abaixo de zero pontos, vale zero valores.

Respostas: (indicar de forma legível a resposta A, B, C ou D, debaixo do número da questão)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

1. Das instruções indicadas a seguir, indique qual a única que é uma instrução TAL:

a. `andi $4,$5,0x0000FF00`
b. `mul $a0,$t0,$t1`
c. `bge $t1,$t2,salto`
d. `ori $4,$5,0x300020`

2. Considere um computador equipado com memória cache em que o acesso à memória demora 70 ns, enquanto o acesso à cache demora 10 ns. Qual o *speedup* que se consegue obter no sistema se 70% dos acessos à memória resultarem numa *cache hit*.

a. 1.1 **b.** 5 **c.** 2.5 **d.** 11

3. Qual dos seguintes segmentos de código em C reproduz mais fielmente o ciclo em *assembly* indicado ao lado?

```
loop:
 slt        $t0,$s3,$s4
 bne        $t0,$0,out
 slt        $t1,$s5,$s4
 beq        $t1,$0,out
 addi       $s4,$s4,-1
 j loop
out:
```

a. `while(($s3>=$s4)||($s5<$s4)){$s4--;}`
b. `while(($s3<$s4)||($s5>=$s4)){$s4--;}`
c. `while(($s3<$s4)&&($s5<$s4)) {$s4--;}`
d. `while(($s3>=$s4)&&($s5<$s4)){$s4--;}`

4. Como se decompõe a instrução em Assembly do MIPS `addi $t1,$t0,0x10203040` em instruções TAL?

a)
```
lui  $at,0x1020
andi $at,$at,0x3040
add  $t1,$t0,$at
```

b)
```
lui $at,0x1020
ori $at,$at,0x3040
add $t1,$t0,$at
```

c)
```
lui  $at,0x3040
ori  $at,$at,0x1020
add  $t1,$t0,$at
```

d)
```
lui $at,0x3040
add $at,$at,0x1020
ori $t1,$t0,$at
```

5. Considere o excerto de código em *Assembly* do MIPS apresentado na caixa ao lado. Indique qual das opções representa o valor armazenado no registo $t3 após a execução deste excerto de código em *Assembly* do MIPS:

```
        .data
tab:    .word 5,15,10,20,25,30,35,40,45,50
        .text
        la      $t0,tab
        lw      $t1,4($t0)
        lw      $t2,12($t0)
        addi    $t3,$0,1
        slt     $t4,$t2,$t1
        bne     $t4,$0,cond2
        addi    $t3,$0,4
cond2:
        addi    $t3,$t3,1
fim_ciclo:
```

a. $t3=5 **c.** $t3=2
b. $t3=6 **d.** $t3=15

6. Considerando o programa em C ao lado, indique qual das seguintes opções é <u>VERDADEIRA</u>.

```
int main(){
  int a[] ={1,2,3,4,5,6,7};
  int *p1, *p2, **p3;
  p1 = a+5;
  p2 = p1-2;
  p3 = &p1;
  printf("%#X\n",*(*p3-3)+(*p2)-2);
  return 0;
}
```

a. O programa termina com um erro do tipo *Segmentation Fault*.
b. É apresentado no ecrã o valor `0x05`.
c. É apresentado no ecrã o valor `0x06`.
d. É apresentado no ecrã o valor `0x07`.

7. Tendo em conta o *datapath* de um processador MIPS e a normal execução de uma instrução, indique qual das seguintes afirmações é <u>VERDADEIRA</u>:

a. Na execução de uma instrução do tipo *branch*, o processador está ativo em todas as etapas do *datapath*.
b. Na execução de uma instrução do tipo *load*, o processador está inactivo na etapa 5
c. Na execução de uma instrução do tipo *jump*, o processador está *inactivo* nas etapas 4 e 5.
d. Na execução de uma instrução do tipo *store*, o processador está ativo na etapa 5 *datapath*.

8. Considere um sistema baseado num processador com um valor de CPI <u>REAL</u> igual a 5.0. O sistema possuí duas caches, uma para instruções e outra para dados. O CPI <u>IDEAL</u> deste sistema é igual a 2. A hit rate da cache de instruções é de 90%. A miss penalty para a memória de instruções é igual a 20 ciclos de relógio e a da memória de dados é igual a 25 ciclos de relógio. Sabendo que apenas 40% das instruções envolvem um acesso à memória de dados indique qual a *hit rate* da cache de dados?

a. 90% **b.** 80% **c.** 85% **d.** 10%

9. Considere uma hierarquia de memória (memória principal + memória cache do tipo *Direct Mapped*) tal que na estrutura de endereçamento, o campo *Index* ocupa 10 bits do endereço. Se, para o mesmo tamanho da memória principal, memória cache e tamanho de bloco da memória, a memória cache fosse do tipo *8-way set-associative*, quantos sets teria a memória *cache*?

a. 512 sets **b.** 1024 sets **c.** 128 sets **d.** 256 sets

10. Considere um certo programa que deve executar ordenadamente as operações OP1, OP2, OP3, OP4, OP5 sobre um conjunto de 3 dados distintos. Considere que a operação mais rápida demora 1ns a executar e a mais lenta demora 2ns a executar. Qual o tempo de execução global assumindo o modo de operação em pipeline (sem stalls de paragem) com pipeline vazia no início da execução do programa?

a. 6 ns **b.** 7 ns **c.** 30 ns **d.** 14 ns

11. Considere uma memória cache do tipo "4-way associative cache" com um tamanho total de 4 MB e blocos de 1024 bytes. Numa arquitetura de 32 bits, um endereço de memória de decompõe-se em:

a. "Tag" de 12 bits, "index" de 10 bits e "offset" de 10 bits.
b. "Tag" de 13 bits, "index" de 9 bits e "offset" de 10 bits.
c. "Tag" de 10 bits, "index" de 12 bits e "offset" de 10 bits.
d. "Tag" de 11 bits, "index" de 11 bits e "offset" de 10 bits.

12. Relativamente ao Assembly do MIPS, indique qual das seguintes afirmações é <u>VERDADEIRA</u>:

a. Nenhuma instrução do tipo I necessita de realocação na fase da *linkagem*.
b. As instruções do tipo R são sempre resolvidas na fase do "*assembling*".
c. A resolução de "*labels*" de instruções "*branches*" é feita pelo "*linker*".
d. As tabelas de símbolos são criadas na fase de pré-processamento e resolvidas na fase da *linkagem*.

13. Relativamente ao excerto de código ao lado indique qual dos seguintes códigos hexadecimais corresponde à codificação da instrução `beq $t0,$0,Loop`**.**

```
Loop: lw   $s0,0($a0)
      addi $s0,$s0,0x40000
      addi $a0,$a0,4
      slti $t0,$s0,100
      beq $t0,$0,Loop
```

a. `0x1100FFFB`
b. `0x1100FFF9`
c. `0x1100FFFA`
d. `0x11000005`

14. Considere o programa em C ao lado. Com base nos valores que as funções *printf* imprimem no ecrã, indique qual das seguintes opções é <u>VERDADEIRA</u>:

a. Os valores a imprimir são: 3, 2, 4, 5
b. Os valores a imprimir são: 3, 3, 4, 5
c. Os valores a imprimir são: 3, 3, 1, 4
d. Os valores a imprimir são: 3, 2, 4, 4

```
int main(){
    int X[]={3,2,1,0};
    int *p=X;

    *(p+1)=X[0];

    printf("%d, %d,",X[0],X[1]);

    X[3]=5;
    p=p+3;

    printf(" %d, %d",*(p-1),*p-1);
    return 0;
}
```

15. Considere a instrução em *Assembly* do MIPS dada pelo seguinte código hexadecimal `0x8C880100`**. Sabendo que os registos** `$t0` **e** `$a0` **são os registos** `#8` **e** `#4`**, respectivamente, indique qual das seguintes instruções representa a descodificação da instrução anterior:**

a. `lw $t0,128($a0)`
b. `lw $a0,256($t0)`
c. `sw $a0,256($t0)`
d. `lw $t0,256($a0)`

16. No trabalho prático 3 utilizaram-se dois displays de 7 segmentos do simulador MARS. Para programar o valor a mostrar no display da esquerda bastava escrever um byte no endereço 0xFFFF0011. Qual seria o resultado do trecho de programa em *assembly* apresentado ao lado?

```
...
addi $a0, $0, 0xFFFF0011
addi $t0, $0, 0x79
sb   $t0,0($a0)
...
```

a. Escreve um **E** no display.
b. Escreve um **6** no display.
c. Escreve um **5** no display.
d. Escreve um **F** no display.

17. Assumindo que a "label" `tabInt` **se refere a uma tabela de inteiros armazenada no endereço de memória** `0x10000448`**, e que as "label"** `funcA` **e** `funcB` **corresponde a uma referência externa ao ficheiro, indique quantas entradas na tabela de realocação gerará o seguinte código Assembly do MIPS?**

```
loop:
        lw      $t0,10($sp)
        lw      $t1,14($sp)
        addu    $a0,$t1,$t0
        jal     funca
        beq      $v0,$a0,end
        sw      $v0,18($sp)
        blt     $t0,50, loop
        la      $a0, tabInt
        sw      $t2,0($a0)
        jal     funcB
        move    $v0,$0
end:
        lw      $ra,8($sp)
        addiu $sp,$sp,24
                ...
```

a. 3 entradas na tabela de realocação.
b. 4 entradas na tabela de realocação.
c. 6 entradas na tabela de realocação.
d. 5 entradas na tabela de realocação.

18. Na compilação de um programa escrito em C e em *assembly*, qual dos seguintes comandos é válido para gerar um executável denominado «*exe*»:

a. gcc main.c func.s -O exe
b. gcc -c exe main.c func.s
c. gcc main.c func.s -o exe
d. gcc -c main.c func.s exe

19. Considerando o trecho de programa indicado ao lado, indique qual das afirmações é <u>FALSA</u>:

```
char str[] = "One STRING";

void main(){
 int i;
 char *temp;

 temp=(char *)malloc(strlen(str)+1);

 for(i=0;i<=strlen(str);i++)
    temp[i]=str[i];
}
```

a. A variável ***str*** vai ser armazenada na zona de dados estáticos.
b. As variáveis ***i*** e ***temp*** vão ser armazenadas na pilha.
c. A variável ***temp*** vai apontar para uma zona de memória no *heap*.
d. A variável ***str*** vai ser armazenada na zona de dados estáticos do programa, a variável ***i*** na pilha e a variável ***temp*** no *heap*.

20. Assumindo que o registo `$a0` **contém o valor** `0x11223344`**, indique qual é o valor armazenado no registo** `$t0` **após a execução do excerto de código *Assembly* do MIPS listado ao lado:**

```
        ori   $t0,$a0,0x0000FF00
        sll  $t0,$t0,8
        andi $t0,$t0,0xFFFF0000
        srl  $t0,$t0,8
        ori  $v0,$t0,0x00000055
```

a. 0x0022FF00
b. 0x22FF3300
c. 0x11FF2233
d. 0x22FF0044