Universidade de Aveiro, DETI

Arquitetura de Computadores I, Teste Prático 1 – 10/11/2018

Ano Letivo 2018/19 - 1º Semestre

Nº Mec.: _____ **Nome:** _____ **MIEET / MIECT**

NOTE BEM: Leia atentamente todas as questões, comente o código usando a linguagem C e respeite a convenção de passagem de parâmetros e salvaguarda de registos que estudou. Na tradução para o *Assembly* do MIPS respeite rigorosamente os aspetos estruturais e a sequência de instruções indicadas no código original fornecido.
O código em C apresentado pode não estar funcionalmente correcto, pelo que **não deve ser interpretado**.
Este teste é constituído por 4 folhas. **Preencha o cabeçalho em todas elas**.

1) Codifique em *Assembly* do MIPS a seguinte função **muc()**:

```
char to_up(char);
int muc(char *s)
{
  int i,count = 0;
  for (i=0; s[i] != '\0'; i++)
  {
    if ((s[i] >= 'a') && (s[i] <= 'z'))
    {
      s[i] = to_up(s[i]);
      count++;
    }
  }
  return count;
}
```

Variável	Registo(s)
s	$a0->$s0
i	$s1
count	$s2
&s[i]	$s3

Função Intermédia

2) Codifique em *Assembly* do MIPS a seguinte função **main()**:

```
#define N    10
#define LEN  20
int fun(char *);
int max(int *, int);
int main(void)
{
  static char str[LEN];
  static int cnt[N];
  int i=0;
  do
  {
    read_string(str, LEN);
    cnt[i] = fun(str);
    print_string(str);
  } while (++i < N);
  print_int10(max(cnt, N));
  return 0;
}
```

Variável	Registo(s)
i	$s0
&cnt[i]	$t0

Função Intermédia

3) Codifique em *Assembly* do MIPS a seguinte função **crc()**:

```
unsigned int crc(unsigned int *pack, int count)
{
  int su16, sum=0;
  while( count > 0 )
  {
    sum += *pack++;
    count--;
  }
  su16 = sum >> 16;
  if( su16 > 0)
  {
    sum = su16 + (sum & 0xffff) ;
  }
  return (~sum) & 0xffff ;
}
```

Variável	Registo(s)
pack	$a0
count	$a1
su16	$a2
sum	$v0

Função Terminal

4) Analise o programa *Assembly* seguinte e responda às questões:

```
        .data                   # 0x10010000
D1:     .byte   1,2             #
S1:     .asciiz "AC1-2018"      # Códigos ASCII-> '0':0x30, 'A':0x41, '-':0x2D
        .align  2               #
A1:     .space  48              #
D2:                             #
        .text                   # 0x00400000
        .globl  main            #
main:   la      $t0,S1          #
        la      $t1,A1          #
        ori     $t2,$0,8        #
        addi    $t2,$t2,-1      #
        sll     $t2,$t2,2       #
        addu    $t2,$t2,$t1     #
C1:     lb      $t3,0($t0)      #
        sw      $t3,0($t2)      #
        add     $t0,$t0,1       #
C2:     addi    $t2,$t2,-4      #
        bge     $t2,$t1,C1      #
        jr      $ra             #
```

a) O espaço total de memória ocupado pela *string* referenciada pelo *label* "**S1**" é. _____

b) O conteúdo de cada uma das posições de memória ocupadas pela *string* (do endereço mais baixo para o mais alto) é:_____

c) Considerando que o segmento de dados começa no endereço **0x10010000**, o valor dos registos **$t0** e **$t1** após a execução das duas primeiras instruções do trecho de código é:

$t0:_____ **$t1**:_____

d) Se "**A1**" for o endereço inicial de um *array* de inteiros, a dimensão máxima desse *array* é: _____, calculada como:_____ e o endereço de memória do elemento **A1[2]** é:

_____

e) O número total de bytes de memória usado pelo segmento de dados (**D2-D1**) é: _____

_____

f) Considerando que a primeira instrução do trecho de código fornecido está armazenada a partir do endereço **0x00400000**, os endereços a que correspondem os *labels* "**C1**" e "**C2**" são (tenha em atenção as instruções virtuais do código e a respetiva decomposição em instruções nativas):

C1:_____ **C2**:_____

g) O valor do registo **$t2** calculado na instrução "**addu**" é: _____

_____

_____

h) Na execução do programa, o número de vezes que o ciclo é realizado é: _____, controlado pela evolução do conteúdo do registo: _____

i) O valor das *words* de 32 bits armazenadas pelo programa nas posições **A1[2]** e **A1[6]** do *array* é:

A1[2]:_____ **A1[6]**:_____

j) O valor dos registos **$t0** e **$t2** no fim do programa é, **$t0**:_____, **$t2**:_____