INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Licenciatura em Engenharia de Eletrónica e Telecomunicações e de Computadores
e
Licenciatura em Engenharia Informática e de Computadores

2.º Trabalho Prático

de

Arquitetura de Computadores

Introdução à Programação em *Assembly*

21 de outubro de 2019

1. Objetivos

Este trabalho tem como principais objetivos o exercício da programação em *assembly* do P16, incluindo a organização dos programas em rotinas e a introdução a um ambiente de programação nesta linguagem.

2. Descrição do Trabalho

O trabalho consiste no desenvolvimento e teste de programas para *i*) operação de números inteiros, *ii*) manipulação de *arrays* em memória e *iii*) invocação de funções.

Os programas devem ser escritos em linguagem *assembly* do P16, podendo o seu teste ser realizado recorrendo ao simulador do P16 ou sobre o sistema SDP16.

Para cada um dos exercícios propostos, deve ser escrito um programa de teste que permita verificar e demonstrar o comportamento da função realizada em diversos cenários de utilização.

3. Tipos

Na especificação dos exercícios, os tipos numéricos definidos têm os seguintes significados:

`int8_t` - inteiro com sinal a 8 bits	`uint8_t` - inteiro sem sinal a 8 bits
`int16_t` - inteiro com sinal a 16 bits	`uint16_t` - inteiro sem sinal a 16 bits

4. Especificação dos Exercícios

4.1. Implemente, em *assembly* do P16, a função `multiply` que realiza a multiplicação, com sinal, de `M` por `m`.

```
#define WIDTH 8

int16_t  multiply ( int8_t M, int8_t m ) {
   int16_t a = M << WIDTH;
   int16_t s = (-M) << WIDTH;
   int16_t p = m & 0xFF;
   int8_t  p_1 = 0;

   for ( uint8_t i = 0; i < WIDTH; i++) {
      if ( (p & 0x1) == 0 && p_1 == 1 )
         p += a;
      else if ( (p & 0x1) == 1 && p_1 == 0 )
         p += s;
      p_1 = p & 0x1;
      p >>= 1;
   }
   return p;
}
```

4.2. Considere a função `partial_area`, que calcula a área de um quadrilátero cujos vértices são P1, P2 e as respetivas projeções verticais no eixo dos x. O *array* `vertices` contém as coordenadas em x dos vértices P1 e P2 nos índices `idx1` e `idx2`, respetivamente, ocupando as coordenadas em y de cada vértice as posições seguintes no *array*.

```
int16_t partial_area ( int8_t vertices[], uint16_t idx1, uint16_t idx2 ){
   int8_t width = vertices[ idx2 ] - vertices[ idx1 ];
   int8_t height = ( vertices[ idx1+1 ] + vertices[ idx2+1 ] ) / 2;
   return multiply( width, height );
}
```

a) Implemente, em *assembly* do P16, a função `partial_area`.

b) Determine, em número de *bytes*, a quantidade de memória de código ocupada pela função `partial_area`.

c) Supondo que a função `partial_area` recebe `{ 10, 10, 20, 20 }` como argumento de `vertices` e 0 e 2 como argumentos de `idx1` e `idx2`, respetivamente, indique o número de ciclos de relógio gastos na execução da função desenvolvida (excluindo a função `multiply`).

4.3. Implemente, em *assembly* do P16, a função `area`, que calcula a área de um polígono com `num_vert` vértices. As coordenadas dos vértices são definidas seguindo o sentido horário em posições consecutivas do *array* `vertices`, em que as coordenadas em x e y de cada vértice ocupam posições adjacentes, por esta ordem.

```
int16_t area( int8_t vertices[], uint8_t num_vert ){
   if( num_vert < 3 )
      return -1;
   int16_t res = 0;
   uint16_t count;
   for( count = 0; count < num_vert - 1; ++count ){
      res += partial_area( vertices, count * 2 , ( count + 1 ) * 2 );
   }
   res += partial_area( vertices, (num_vert – 1) * 2 , 0 );
   return res;
}
```

4.4. Considere as definições seguintes e a função `main`.

```
#define MAX_RESULTS 3
int8_t pol1[] = { 10, 10, 20, 20 };
int8_t pol2[] = {  0,  5, 10, -5,  0, -15, -10,  -5 };
int8_t pol3[] = { 10,  0, 20, 10, 30,   0,  20, -10 };
int16_t areas[MAX_RESULTS];

int main( void ){
   areas[0] = area( pol1, 2 );
   areas[1] = area( pol2, 4 );
   areas[2] = area( pol3, 4 );
   return 0;
}
```

a) Implemente, em *assembly* do P16, as definições referidas e a função `main`.

b) Verifique e comente os resultados da execução produzidos na variável `areas`.

5. Avaliação

O trabalho deve ser realizado em grupo, conta para o processo de avaliação da unidade curricular, estando sujeito a discussão final, e tem a duração de 2 semanas.

A apresentação da solução proposta por cada grupo decorre em sessão de laboratório, em data a combinar com o docente responsável pela lecionação das aulas da respetiva turma.

Após essa apresentação, cada grupo deverá entregar ao docente o trabalho desenvolvido, na forma de listagens dos programas realizados, devidamente indentadas e sucintamente comentadas.

As respostas às alíneas b) e c) do exercício 4.2 e da alínea b) do exercício 4.4 devem ser incluídas na própria listagem, sob a forma de comentários.