**Aluno:** Mateus Alves da Rocha **Matrícula:** 11/0132661

Data: 31/03/2017

Para cada questão, escreva funções em C e/ou sub-rotinas na linguagem Assembly do MSP430. Reaproveite funções e sub-rotinas de uma questão em outra, se assim desejar. Leve em consideração que as sub-rotinas são utilizadas em um código maior, portanto utilize adequadamente os registradores R4 a R11. As instruções da linguagem Assembly do MSP430 se encontram ao final deste texto.

1. (a) Escreva uma função em C que calcule a raiz quadrada 'x' de uma variável 'S' do tipo float, utilizando o seguinte algoritmo: após 'n+1' iterações, a raiz quadrada de 'S' é dada por

x(n+1) = (x(n) + S/x(n))/2

O protótipo da função é:

unsigned int Raiz\_Quadrada(unsigned int S);

unsigned int Raiz\_Quadrada(unsigned int S){

int t=1;

float x = S/2;

while (t!=0)

{

float x\_anterior = x;

x= (x\_anterior + S/x\_anterior)/2;

float aux = x-x\_anterior;

if (aux >= 0.0)

if (aux < 0.001)

t=0;

else

if (-aux < 0.001)

t=0;

}

return x;

}

(b) Escreva a sub-rotina equivalente na linguagem Assembly do MSP430. A variável 'S' é fornecida pelo registrador R15, e a raiz quadrada de 'S' (ou seja, a variável 'x') é fornecida pelo registrador R15 também.

Raiz\_Quadrada:

Mov.w R15, R13 ; R13 = R15 = S

RRA.w R13 ; R13 = S/2;

While\_loop:

Mov.w R13, R12 ; R12= x\_anterior = S/2;

PUSH R12 ; Guarda R12 na pilha

Call #Div\_unsigned ; chama a função para calcular S/x\_anterior

Add.w R12, R15; ; soma o resultado da divisão com R12

RRA.W R15; ; resultado da soma dividido por 2

POP R12 ; recupera R12 da pilha

Mov.w R15, R14; ; R14 recebe o resultado

Sub.w R12, R14 ; R14 = X – X\_anterior

TST R14 ; R14 = 0?

Jeq Exit\_loop

Jmp while\_loop

Exit\_loop:

Div\_unsigned:

Clr.w R11 ; i=0

For\_div:

cmp R15, R12 ; compara R15 = S com R12 = x\_anterior

jl End\_for\_div ; sai do for

sub R12, R15 ; R15 = R15 – R12

inc R13 ; Incrementa R13

Jmp For\_div ; repete o loop

End\_for\_div

Mov R13, R15 ; retorna o valor final da divisao para R15

ret

2. (a) Escreva uma função em C que calcule 'x' elevado à 'N'-ésima potência, seguindo o seguinte protótipo:

#include <stdio.h>

#include <stdlib.h>

unsigned int mult\_unsigned (unsigned int a, unsigned int b){

unsigned int c = 0;

int i = 0;

for (i=1; i<=b; i++)

{

c += a;

}

return c;

}

unsigned int div\_signed (unsigned int a, unsigned int b){

unsigned int i;

for (i =0; a>=b; i++)

{

a = a-b;

}

return i;

}

int potencia(int x, int N){

int sign=0;

int p=1;

if(N < 0)

{

N = -N;

sign = 1;

}

while ( N>0)

{

p = mult\_unsigned (x,p);

N--;

}

if(sign ==1)

return div\_signed(1, p);

else

return p;

}

int main ()

{

int S = 2;

int a = 3;

float x ;

x = potencia (2,3);

printf("%.2f\n", x);

return(0);

}

(b) Escreva a sub-rotina equivalente na linguagem Assembly do MSP430. 'x' e 'n' são fornecidos através dos registradores R15 e R14, respectivamente, e a saída deverá ser fornecida no registrador R15.

mult\_unsigned: clr R13 ; c=0

mov #1, R12 ; i=1

for\_mult: cmp R12, R14

jl end\_for\_mult

add R15, R13

inc R12

jmp for\_mult

end\_for\_mult: mov R13, R15

ret

div\_unsigned: clr R13 ; i=0

for\_div: cmp R14, R15

jl end\_for\_div

sub R14, R15

inc R13

jmp for\_div

end\_for\_div: mov R13, R15

ret

potencia: push R4

push R5

clr R4 ; sign=0

mov #1, R5 ; p=1

cmp #1, R14

jl while\_pot

inv R14

inc R14

mov #1, R4

while\_pot: tst R14

jz while\_pot\_end

push R15

push R14

mov R5, R14

call mult\_unsigned

mov R15, R5

pop R14

pop R15

dec R14

jmp while\_pot

while\_pot\_end: cmp #1, R4

jne pot\_end

mov #1, R15

mov R5, R14

call div\_unsigned

pop R5

pop R4

ret

pot\_end: mov R5, R15

pop R5

pop R4

ret

3. Escreva uma sub-rotina na linguagem Assembly do MSP430 que calcula a divisão de 'a' por 'b', onde 'a', 'b' e o valor de saída são inteiros de 16 bits. 'a' e 'b' são fornecidos através dos registradores R15 e R14, respectivamente, e a saída deverá ser fornecida através do registrador R15.

4. Escreva uma sub-rotina na linguagem Assembly do MSP430 que calcula o resto da divisão de 'a' por 'b', onde 'a', 'b' e o valor de saída são inteiros de 16 bits. 'a' e 'b' são fornecidos através dos registradores R15 e R14, respectivamente, e a saída deverá ser fornecida através do registrador R15.

rest\_of\_division: cmp R14, R15

jl rest\_end ; R15 < R14 termina o laço

sub R14, R15 ; R15 = R15 - R14

jmp rest\_of\_division ; repete o laço

rest\_end: ret

5. (a) Escreva uma função em C que indica a primalidade de uma variável inteira sem sinal, retornando o valor 1 se o número for primo, e 0, caso contrário. Siga o seguinte protótipo:

int Primalidade(unsigned int x);

(b) Escreva a sub-rotina equivalente na linguagem Assembly do MSP430. A variável de entrada é fornecida pelo registrador R15, e o valor de saída também.

6. Escreva uma função em C que calcula o duplo fatorial de n, representado por n!!. Se n for ímpar, n!! = 1\*3\*5\*...\*n, e se n for par, n!! = 2\*4\*6\*...\*n. Por exemplo, 9!! = 1\*3\*5\*7\*9 = 945 e 10!! = 2\*4\*6\*8\*10 = 3840. Além disso, 0!! = 1!! = 1.

O protótipo da função é:

unsigned long long DuploFatorial(unsigned long long n);

7. (a) Escreva uma função em C que calcula a função exponencial da seguinte forma:

Considere o cálculo até o termo n = 20. O protótipo da função é double ExpTaylor(double x);

(b) Escreva a sub-rotina equivalente na linguagem Assembly do MSP430, mas considere que os valores de entrada e de saída são inteiros de 16 bits. A variável de entrada é fornecida pelo registrador R15, e o valor de saída também.

8. Escreva uma sub-rotina na linguagem Assembly do MSP430 que indica se um vetor esta ordenado de forma decrescente. Por exemplo:

[5 4 3 2 1] e [90 23 20 10] estão ordenados de forma decrescente.

[1 2 3 4 5] e [1 2 3 2] não estão.

O primeiro endereço do vetor é fornecido pelo registrador R15, e o tamanho do vetor é fornecido pelo registrador R14. A saída deverá ser fornecida no registrador R15, valendo 1 quando o vetor estiver ordenado de forma decrescente, e valendo 0 em caso contrário.

9. Escreva uma sub-rotina na linguagem Assembly do MSP430 que calcula o produto escalar de dois vetores, 'a' e 'b':

O primeiro endereço do vetor 'a' deverá ser passado através do registrador R15, o primeiro endereço do vetor 'b' deverá ser passado através do registrador R14, e o tamanho do vetor deverá ser passado pelo registrador R13. A saída deverá ser fornecida no registrador R15.

10. (a) Escreva uma função em C que indica se um vetor é palíndromo. Por exemplo:

[1 2 3 2 1] e [0 10 20 20 10 0] são palíndromos.

[5 4 3 2 1] e [1 2 3 2] não são.

Se o vetor for palíndromo, retorne o valor 1. Caso contrário, retorne o valor 0. O protótipo da função é:

int Palindromo(int vetor[ ], int tamanho);

(b) Escreva a sub-rotina equivalente na linguagem Assembly do MSP430. O endereço do vetor de entrada é dado pelo registrador R15, o tamanho do vetor é dado pelo registrador R14, e o resultado é dado pelo registrador R15.