Trabalho de Asc I

Inês Maria Veríssimo 40102
Ricardo Mochila 37762

27 de Maio de 2018

1 Introdução

Através do uso da linguagem assembly em Mips foi desenvolvida uma calculadora que permite ao utilizador escrever todos os algarismos que pretende operacionar, bem como os respetivos operadores. A calculadora em RPN - Notação polaca inversa, tem como objetivo reconhecer e resolver o input e devolver um valor.

2 Desenvolvimento

Primeiramente foi necessário alocar espaço na pilha e identificar o código ASCII de todos os elementos a utilizar.

Em seguida, percorrendo o input, entre os "espaços"faz-se o reconhecimento de cada caracter para determinar se este será alocado na pilha ou lido como um operador, ou string. Este processo repete-se até ao fim do input (caracter nulo).

Após a identificação dos operadores ou strings, o programa executa a função correspondente. Foram implementadas as seguintes funções operacionais: soma, subtração, multiplicação, divisão, neg, dup, swap e sqrt, clear e off.

Soma Soma primeiros algarismos da pilha. Mais especificamente, lê os dois primeiros bits (os do topo da pilha) e faz a soma dos mesmos.

```
soma:
lb $t0, -4($t7)
lb $t1, -3($t7)
add $t0, $t0, $t1
sb $zero, -3($t7)
sb $zero, -4($t7)
sb $t0, -4($t7)
addi $s0, $s0, 1
addi $t7, $t7, -1
```

A função volta para a função "ler_ input"

Subtração Subtrai os primeiros algarismos da pilha.

```
subtracao:
lb $t0, -4($t7)
lb $t1, -3($t7)
sub $t0, $t1, $t0
sb $zero, -3($t7)
sb $zero, -4($t7)
sb $t0, -4($t7)
addi $s0, $s0, 1
addi $t7, $t7, -1
```

A função volta para a função "ler_ input"

Multiplicação multiplica os primeiros elementos da pilha e devolve o seu resultado, substituindo-o na pilha.

```
multiplica:
lb $t0, -4($t7)
lb $t1, -3($t7)
mult $t0, $t1
mflo $v0
sb $zero, -3($t7)
sb $zero, -4($t7)
sb $v0, -4($t7)
addi $s0, $s0, 1
addi $t7, $t7, -1
```

A função volta para a função "ler_ input"

Divisão Implementa o quociente entre os primeiros elementos da pilha e devolve o seu resultado, substituindo-o na pilha.

```
divisao:
lb $t0, -4($t7)
lb $t1, -3($t7)
div $t1, $t0
mflo $v0
sb $zero, -3($t7)
sb $zero, -4($t7)
sb $v0, -4($t7)
addi $s0, $s0, 1
addi $t7, $t7, -1
```

A função volta para a função "ler_ input"

Neg É adicionado entre os algarismos numéricos, no input, e calcula o simétrico do que estiver anterior à string. Após a negação, guarda o inteiro no array "pilhaNumeros".

```
negar:
sub $t6, $zero, $t6
sb $t6, -2($t7)
addi $t7, $t7, 1
addi $s0, $s0, 1
```

A função volta para a função "ler_ input"

Dup Duplica o caracter que se encontrar no topo da pilha

```
dup:
lb $t6, -3($t7)
sb $t6, -4($t7)
addi $s0, $s0, 1
```

A função volta para a função "ler_ input"

Swap Troca posição dos dois operandos do topo da pilha.

```
swap:
lb $t5, -4($t7)
lb $t6, -3($t7)
sb $t6, -4($t7)
sb $t5, -3($t7)
addi $s0, $s0, 1
```

A função volta para a função "ler_ input"

Sqrt Implementa a raiz quadrada do elemento numérico que se anterior à string.

```
sqrt:
lb $t6, -3($t7)
add $t5, $t6, $zero
add $t1, $zero, $zero
add $t2, $zero, $zero
addi $t2, $t2, 2
div $t8, $t6, $t2
mflo $t8
   loop:
   div $t5, $t6, $t5
   mflo $t5
   add $t6, $t6, $t5
   div $t6, $t6, $t2
   mflo $t6
   addi $t1, $t1, 1
   blt $t1, $t8, loop
   nop
sb $t6, -3($t7)
addi $s0, $s0, 1
```

Esta função necessita de um loop para realizar as operações necessária ao desenvolvimento da raiz quadrada do número pretendido.
volta para a função "ler_ input"

Clear Repões todos os registos a zero e regressa à função pedir_ input, para se iniciarem novamente cálculos.

```
clear:
add $t0, $zero, $zero
add $t1, $zero, $zero
add $t2, $zero, $zero
add $t3, $zero, $zero
add $t4, $zero, $zero
add $t5, $zero, $zero
add $t6, $zero, $zero
add $t7, $zero, $zero
add $t8, $zero, $zero
```

Off Termina o programa (desliga a calculadora)

Nota Todas as operações acima descritas realizam-se independentemente de já se ter ou não operacionado algum tipo de operação anteriormente (pois a pilha é constantemente modificada e o input percorrido).

ler_input Esta função lê cada caracter à vez, grava-os no array da pilha1. Verifica também se o array "pilha"já terminou e passa ao seguinte caracter. Reconhece se o caracter é um número, uma string ou um operador, através do código ASCII. Cada caracter éreconhecido após ser encontrado um "espaço"no input.Seguidamente, para se prosseguirem as operações, passa-se ao espaço seguinte da "pilha1"

converte Esta função tem como objetivo carregar na posição 0 "zero"o número que se encontrar no registo $t0. Este número é convertido de ASCII para inteiro. guardado no array "pilhaNumeros". Passa-se ao seguinte elemento da "pilhaNumeros"e repete-se o ciclo.

ler_string É a função capaz de determinar, pelo código ASCII, a operação pretendida pelo utilizador. Após isso, regressa à função "ler_input".

Foram adicionadas mensagens, nomeadamente de identificação dos alunos que desenvolveram o trabalho, do pedido de input e do respetivo resultado,

3 Registos Utilizados

Registos	Descrição
$t0	Contém o input do utilizador.
$t1	É normalmente utilizado para guardar o segundo caracter a ser operado.
$t2	Contém a soma em código ASCII dos caracteres da string (exemplo swap=0x01bb).
$t6	Contém o resultado da função "converte".
$t7	É utilizado para devolver o valor da operação.
$s0	Em algumas funções, como a soma, faz com que se passe ao elemento seguinte do array.

4 Codigos ASCII utilizados

Simbolo	Codigo Hexadecimal
+	0x002b
-	0x002d
/	0x002f
*	0x002a
Swap	0x01bb
Neg	0x013a
Dup	0x0149
Sqrt	0x01ca
Clear	0x0207
Off	0x013b

5 Por melhorar

Foram encotradas dificuldades no retorno da pilha, para mera informação do utilizador, e, também, na utlização de números não compreendidos entre 0-9. Dificuldades estas que tentarão ser resolvidas posteriormente à entrega do trabalho.

6 Conclusão

Foi desenvolvida uma calculadora em RPN - Notação polaca inversa, que possibilita o input de todos os algarismos e de todos os operadores apenas numa linha e que devolve um resultado. Este trabalho foi importante na medida em que permitiu o melhor entendimento da linguagem assembly e o modo como se podem desenvolver trabalhos na cadeira de Arquitetura de Sistemas e Computadores de forma eficiente e rápida.