5 Programação em *Assembly* do MIPS

Objetivos: São dois os objetivos deste laboratório: (i) aprender usar o simulador MARS; e (ii) escrever e testar um programa completo em *assembly*.

Preparação: Veja MARS_Tutorial.pdf e MARS_features.pdf em `http://www.inf.ufpr.br/roberto/ci210/assembly` .

5.1 Programa que Computa a Série de Fibonacci

Para executar o simulador MARS diga:

```
java -jar /home/soft/linux/mips/Mars.jar
```

Possivelmente, seria uma boa ideia acrescentar ao seu `~/.bashrc` a função que contenha esta linha de comando. Edite `~/.bashrc` e acrescente o seguinte, preferencialmente no final do arquivo:

```
function mars() { java -jar /home/soft/linux/mips/Mars.jar "$@" ; }
```

Isso feito, diga `source ~/.bashrc` e então invoque o simulador dizendo apenas `mars` .

Copie `http://www.inf.ufpr.br/roberto/ci210/assembly/fibonacci.s` para a sua área de trabalho e siga as instruções em MARS_Tutorial.pdf.

5.2 Programa que Computa o Fatorial

Copie `http://www.inf.ufpr.br/roberto/ci210/assembly/fatorial.s` para a sua área de trabalho e verifique se o programa produz resultados corretos. Se encontrar algum erro, corrija-o e verifique sua solução.

Programa 1: fatorial.s

```
# void main(void) { // fatorial iterativo

#     int i,j;
        .data                   # área de dados
vi:     .word 1                 # aloca espaço para vars globais
vj:     .word 1                 # apenas como exemplo...

resp:   .asciiz "fatorial de 5 = "      # constante tipo string

        .text                   # área de código
        .globl main             # define main como nome global
main:

#       j=1;
        addi $t1, $zero, 1

#       for(i=1; i <= n; i++)
        addi $t4, $zero, 1

for:    slti $t7,$t4,6          # n < 6     fat(5)
        beq $t7,$zero,fimfor

#       j = j*i;
        mult $t1,$t4
        mflo $t4

        addi $t4, $t4, 1

        j for

fimfor: addi $v0, $zero, 4      # imprime resposta
        la $a0, resp            # syscall(4) = imprime string
        syscall

        addi $v0, $zero, 1      # imprime inteiro
        addu $a0, $zero, $t1    # syscall(1) = imprime inteiro
        syscall

#     return(0);
        li $v0, 10              # termina programa
        syscall                 # syscall(10) = termina programa

# }
```

5.3 Programa para Copiar *Strings*

Traduza o Programa 2 para *assembly* do MIPS e verifique sua corretude com MARS. Use o código que imprime inteiros em `fibonacci.s` como modelo para o `printf`, alterando o tipo de saída de inteiro para *string*.

Os códigos das *syscalls* estão definidos em Help → MIPS → Syscalls.

A instrução **lb** rt, desl(rs) carrega o byte apontado por (extSinal(desl)+rs) no registrador rt.

A instrução **lbu** rt, desl(rs) é similar à **lb** mas não estende o sinal do byte carregado.

A instrução **sb** rt, desl(rs) armazena o byte menos significativo em rt no endereço apontado por (extSinal(desl)+rs).

Para alocar as *strings* em memória use a linha 8 do Programa 1 como exemplo. A diretiva `.asciiz` aloca uma *string* incluindo o `'\0'`, enquanto que `.ascii` aloca uma *string* sem o `'\0'`. Veja o tutorial ao Mars para a lista das diretivas que este provê.

As diretivas aceitas pelo montador do Mars não são as mesmas providas pelo mips-as.

Programa 2: strcpy.c

```
char fte[16]="abcd-efgh-ijkl-";
char dst[16]= { '\0' };                 // inicializa com '\0'

void main (void) {
   int i,f,n;

   i=1;                                 // inclui '\0' na contagem
   // copia e computa tamanho da cadeia, inclusive '\0'
   while( (dst[i] = fte[i]) != '\0' )  // atribui e então compara
      i++;
   dst[i] = '\0';

   // sua versão assembly de printf() deve ter um
   // número fixo de argumentos
   printf("fonte: %s \n", fte);
   printf("dest:  %s \n", dst);
   printf("tam: %d \n", i);

   return(0);
}
```

5.4 Mais do Fatorial

Este exercício é para aqueles que já utilizaram o Mars em outras disciplinas, e que chegaram a esta tarefa com tempo disponível. Para aqueles que utilizam Mars pela primeira vez, este exercício deve ser tentado fora do horário de aula.

Traduza o Programa 3 para *assembly* do MIPS e verifique sua corretude com MARS.

Programa 3: Duas versões do fatorial iterativo

```
void main (void) {
   int i,f,n;

   n=5;
   i=f=1;

   do {
      f = f * i;
      i = i + 1;
   } while (i <= n);

   printf("%d␣%d\n",n,f);

   f=1;
   i=n;
   while (i > 0) {
      f= f*i;
      i = i - 1;
   }
   printf("%d␣%d\n",n,f);

   return(0);
}
```

Referências

[RH14] *CI064 – Software Básico*, Roberto A Hexsel, 2014, http://www.inf.ufpr.br/roberto/ci064/swbas.pdf

EOF