Aula prática 7

- Utilização da convenção do MIPS para passagem de parâmetros e uso dos registos.

Bernardo Cunha, José Luís Azevedo, Arnaldo Oliveira

MIPS – Convenção sobre a utilização de registos.

- Utilização convenção é um conjunto de regras acordadas entre todos os programadores de uma dada arquitetura por forma a garantir a consistência e o correto funcionamento dos software escrito para essa arquitetura.
- Por esse motivo, independente de quem escreve o código (*Assembly* ou máquina) e da forma como este é organizado, todos os programadores devem seguir de forma estrita essa convenção.

MIPS – Interface entre funções (sub-rotinas).

- Os primeiros 4 parâmetros a passar a uma sub-rotina (desde que caibam em 32 bits) devem utilizar os registos $a0 a $a4, sempre por esta ordem.

  "a" vem do inglês argument

- Nos casos em que a sub-rotina devolve um valor a um programa chamador, esse valor deve ser devolvido no registo $v0 ($v0 e $v1 quando se trata de um valor de 64 bits)

MIPS – Utilização de registos.

'a' vem de argument

- Os registos [`$t0` .. `$t9`], [`$v0` .. `$v1`], e [`$a0` .. `$a3`] podem ser livremente utilizados e alterados pelas sub-rotinas
- Os registos [`$s0` .. `$s7`] não podem, **na perspetiva do chamador**, ser alterados pelas sub-rotinas

's' vem de static

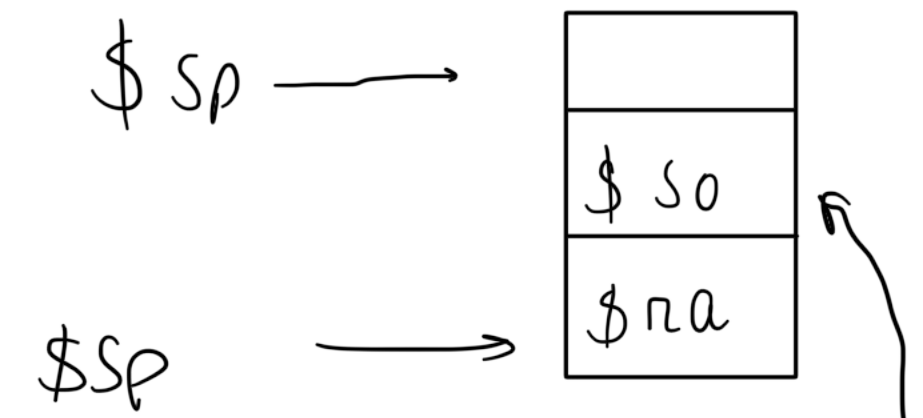
  - Se uma dada sub-rotina precisar de usar qualquer um dos registos `$s0` a `$s7` compete a essa sub-rotina **salvaguardar previamente o seu conteúdo**, repondo-o imediatamente antes de terminar
  - Ou seja, é seguro para o programa chamador usar um registo `$sn` para armazenar um valor que vai necessitar após a chamada à sub-rotina, uma vez que tem a garantia que esta não o modifica
- Uso de stacks: ver aula TP 9 e 10 a partir do slide 31

Considerações práticas sobre a utilização da convenção

- **sub-rotinas terminais** (sub-rotinas folha, i.e., que não chamam qualquer sub-rotina)
  - Só devem utilizar registos que não têm a responsabilidade de salvaguardar (`$t0..$t9`, `$v0..$v1` e `$a0..$a3`)

T significa temporary

- **sub-rotinas intermédias** (sub-rotinas que chamam outras sub-rotinas)
  - Devem utilizar os registos `$s0..$s7` para o armazenamento de valores que pretendam preservar durante a chamada à sub-rotina seguinte
    - A utilização de qualquer um dos registos `$s0` a `$s7` implica a sua prévia salvaguarda na memória externa logo no início da sub-rotina e a respetiva reposição no final
  - Devem utilizar os registos `$t0..$t9`, `$v0..$v1` e `$a0..$a3` para os restantes valores




```
sw   $ra , 0($sp)
sw   $s0 , 4($sp)
```

Regras para a implementação de sub-rotinas no MIPS

1. A sub-rotina chamadora, antes de chamar:
   - Passa os parâmetros; os 4 primeiros são passados nos registos $a0..$a3 e os restantes na *stack*.
   - Executa a instrução "jal".

2. A sub-rotina chamada, no início:
   - Salvaguarda na *stack* os registos $s0 a $s7 que pretende utilizar.
   - Salvaguarda o registo $ra no caso de a rotina também ser chamadora.

3. A sub-rotina chamada, no fim:
   - Coloca o valor de retorno em $v0 (exceto se for tipo *void*).
   - Restaura os registos $s0 a $s7 que salvaguardou no início.
   - Restaura o registo $ra (no caso de ter sido salvaguardado no início).
   - Retorna, executando a instrução "jr $ra".

4. A sub-rotina chamadora, após regresso:
   - Usa o valor de retorno que está em $v0.

5. A sub-rotina chamadora não pode assumir em caso algum que qualquer dos registos
   - $a0..$a3, $t0..$t9, $v0 e $v1 têm o conteúdo preservado pela rotina chamada.

6. A codificação da sub-rotina "main()" está sujeita às mesmas regras que se aplicam às restantes sub-rotinas.