Exemplo 2 - Trabalhando com dados em memória

Detalhe importante: O acesso à memória do SPIM em nossos programas deve ser feito com valores acima da posicao inicial do global pointer (10008000h). Esta é a parte da memória do SPIM que iremos utilizar para armazenar nossos dados. Mais a frente veremos também como armazenar dados "constantes" de uma forma mais prática, sem a necessidade de utilizar as instruções de store.

Vamos fazer um exercício simples de acesso a memória:

Tendo-se um array de 100 elementos (words) que inicia no endereço de memória 5000 (em direção aos endereços crescentes) transfira este array para o endereço 6000.

Primeiro precisamos carregar na memória este array de elementos que foi considerado no enunciado. Como as posições de memória 5000 e 6000 estão fora da área que temos acesso na memória, utilizaremos o $gp (global pointer) + 5000 como endereço inicial do array fonte. Já para o array destino, utilizaremos o valor de $gp + 6000.

```
.text
.globl main
main:
move $s0,$gp
addi $s0,$s0,5000          # Ponteiro para os dados do array fonte ($gp) + 5000
move $s2,$s0
addi $s2,$s2,400           # Marcador para indicar o final do array $gp + (100posições de 4bytes)
```

Desta forma temos os ponteiros necessários para trabalhar com o primeiro array. O $s0 será o ponteiro e será incrementado sempre em 4 posições para apontar para a próxima word (próximo elemento). Vamos armazenar nele um dado qualquer, como por exemplo um valor incrementado sempre em 9 (9, 18, 27, 36...)

Detalhe importante: Para marcar pontos importantes no programa, utilizamos "labels" (rótulos). É através deles que executamos funções como jump e branch. Para definir um label, coloque um identificador seguido do sinal de dois pontos, ex: "repetir:" e escreva o código. Neste exercício faremos um loop para preencher os 100 elementos do array, por isso vamos precisar de um label para chamar a cada iteração. Quando o label é chamado (através de um bne por exemplo) a próxima instrução a ser executada é a da linha seguinte ao label.

```
li $t0,9                   # Carrega no reg. temporário $t0 um valor para ser armazenado no array
dados:
sw $t0,0($s0)              # Armazena o valor na posição do array apontada por $s0
addi $s0,$s0,4             # Aponta para a próxima posição no array (incrementa em 4 o ponteiro)
addi $t0,$t0,9             # Altera o valor a ser armazenado no array (incrementa em 9)
bne $s0,$s2,dados          # Enquanto não chegar ao fim do array, repete o laço
```

Vamos executar esta primeira parte do programa para testarmos o armazenamento dos valores do array. Salve o arquivo com o nome de "*exercicio2a.s*" e abra-o no SPIM. Execute o código (F5 e depois OK). Vamos verificar se ocorreu tudo bem. Na janela dos registradores o $s0 deverá estar em

10009518h que é a marca do final do array (o array inicia em 10009388h e o último elemento está em 10009514Ch). O registrador $t0 contém o valor 38Dh (909 em decimal), ou seja, o valor que seria armazenado na posição seguinte a última. Até aqui tudo Ok.

Agora verificaremos os valores em memória. Abra a janela Data Segment (Window, Data Segment). Ela deverá estar assim:

```
      DATA
[0x10000000]...[0x10009384]   0x00000000
[0x10009384]                    0x00000000  0x00000009  0x00000012
[0x10009390]                    0x0000001b  0x00000024  0x0000002d  0x00000036
[0x100093a0]                    0x0000003f  0x00000048  0x00000051  0x0000005a
[0x100093b0]                    0x00000063  0x0000006c  0x00000075  0x0000007e
[0x100093c0]                    0x00000087  0x00000090  0x00000099  0x000000a2
[0x100093d0]                    0x000000ab  0x000000b4  0x000000bd  0x000000c6
[0x100093e0]                    0x000000cf  0x000000d8  0x000000e1  0x000000ea
[0x100093f0]                    0x000000f3  0x000000fc  0x00000105  0x0000010e
[0x10009400]                    0x00000117  0x00000120  0x00000129  0x00000132
[0x10009410]                    0x0000013b  0x00000144  0x0000014d  0x00000156
[0x10009420]                    0x0000015f  0x00000168  0x00000171  0x0000017a
[0x10009430]                    0x00000183  0x0000018c  0x00000195  0x0000019e
[0x10009440]                    0x000001a7  0x000001b0  0x000001b9  0x000001c2
[0x10009450]                    0x000001cb  0x000001d4  0x000001dd  0x000001e6
[0x10009460]                    0x000001ef  0x000001f8  0x00000201  0x0000020a
[0x10009470]                    0x00000213  0x0000021c  0x00000225  0x0000022e
[0x10009480]                    0x00000237  0x00000240  0x00000249  0x00000252
[0x10009490]                    0x0000025b  0x00000264  0x0000026d  0x00000276
[0x100094a0]                    0x0000027f  0x00000288  0x00000291  0x0000029a
[0x100094b0]                    0x000002a3  0x000002ac  0x000002b5  0x000002be
[0x100094c0]                    0x000002c7  0x000002d0  0x000002d9  0x000002e2
[0x100094d0]                    0x000002eb  0x000002f4  0x000002fd  0x00000306
[0x100094e0]                    0x0000030f  0x00000318  0x00000321  0x0000032a
[0x100094f0]                    0x00000333  0x0000033c  0x00000345  0x0000034e
[0x10009500]                    0x00000357  0x00000360  0x00000369  0x00000372
[0x10009510]                    0x0000037b  0x00000384  0x00000000  0x00000000
[0x10009520]...[0x10040000]   0x00000000
```

Aqui o SPIM apresenta os dados carregados em memória. Perceba que ele mostra somente as posições ocupadas. Os dados no array devem ser os seguintes: o valor 9 na primeira posição, o valor 18 na segunda e assim sucessivamente até o valor 900 na última posição. Podemos conferir os valores armazenados no nosso array: 9h, 12h, 1Bh... = 9, 18, 27... O último valor é 384h = 900. Tudo Ok até aqui!

Agora podemos continuar com o exercício. Vamos fazer a cópia dos dados para o array destino.

```
move $s0,$gp

addi $s0,$s0,5000          # Definimos novamente o ponteiro para os dados do
                           array fonte ($gp + 5000)
move $s1,$gp

addi $s1,$s1,6000          # Ponteiro para os dados do array destino ($gp +
                           6000)
transfere:

lw $t0,0($s0)              # Armazena em t0 o conteúdo da posição apontada por
                           $s0 (array fonte)
sw $t0,0($s1)              # Armazena no array destino (apontado por $s1) o
                           valor carregado
addi $s0,$s0,4             # Incrementa s0 em 4 (para chegar-se ao próximo
                           elemento no array fonte)
addi $s1,$s1,4             # Incrementa s1 em 4 (para chegar-se ao próximo
                           elemento no array destino)
bne $s0,$s2,transfere      # Enquanto s0 não chegar em 400 (100 elementos),
                           repete o laço
```

Salve novamente o arquivo ("*exercicio2b.s*") e execute-o.

Agora a janela de dados vai apresentar os dois arrays, sendo que o segundo foi armazenado da posição 10009770h para cima. Observe que esta posição é o $gp (10008000h) + 6000.

[0x10009520]...[0x1000976c]	0x00000000			
[0x1000976c]	0x00000000			
[0x10009770]	0x00000009	0x00000012	0x0000001b	0x00000024
[0x10009780]	0x0000002d	0x00000036	0x0000003f	0x00000048
[0x10009790]	0x00000051	0x0000005a	0x00000063	0x0000006c
[0x100097a0]	0x00000075	0x0000007e	0x00000087	0x00000090
[0x100097b0]	0x00000099	0x000000a2	0x000000ab	0x000000b4
[0x100097c0]	0x000000bd	0x000000c6	0x000000cf	0x000000d8
[0x100097d0]	0x000000e1	0x000000ea	0x000000f3	0x000000fc
[0x100097e0]	0x00000105	0x0000010e	0x00000117	0x00000120
[0x100097f0]	0x00000129	0x00000132	0x0000013b	0x00000144
[0x10009800]	0x0000014d	0x00000156	0x0000015f	0x00000168
[0x10009810]	0x00000171	0x0000017a	0x00000183	0x0000018c
[0x10009820]	0x00000195	0x0000019e	0x000001a7	0x000001b0
[0x10009830]	0x000001b9	0x000001c2	0x000001cb	0x000001d4
[0x10009840]	0x000001dd	0x000001e6	0x000001ef	0x000001f8
[0x10009850]	0x00000201	0x0000020a	0x00000213	0x0000021c
[0x10009860]	0x00000225	0x0000022e	0x00000237	0x00000240
[0x10009870]	0x00000249	0x00000252	0x0000025b	0x00000264
[0x10009880]	0x0000026d	0x00000276	0x0000027f	0x00000288
[0x10009890]	0x00000291	0x0000029a	0x000002a3	0x000002ac
[0x100098a0]	0x000002b5	0x000002be	0x000002c7	0x000002d0
[0x100098b0]	0x000002d9	0x000002e2	0x000002eb	0x000002f4
[0x100098c0]	0x000002fd	0x00000306	0x0000030f	0x00000318
[0x100098d0]	0x00000321	0x0000032a	0x00000333	0x0000033c
[0x100098e0]	0x00000345	0x0000034e	0x00000357	0x00000360
[0x100098f0]	0x00000369	0x00000372	0x0000037b	0x00000384
[0x10009900]...[0x10040000]	0x00000000			

Pronto! Os dados foram transferidos para o array destino.