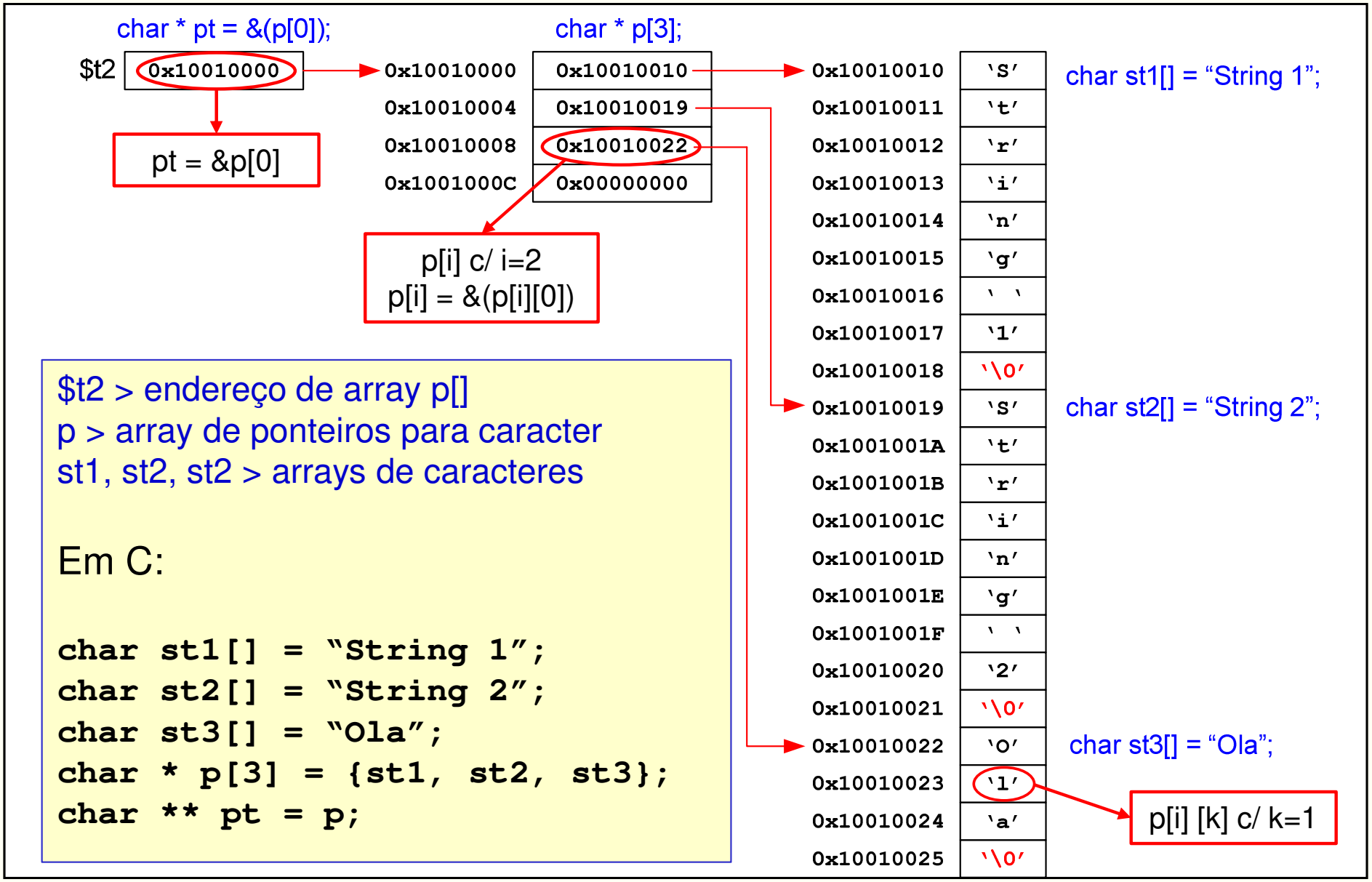
Aula prática 6

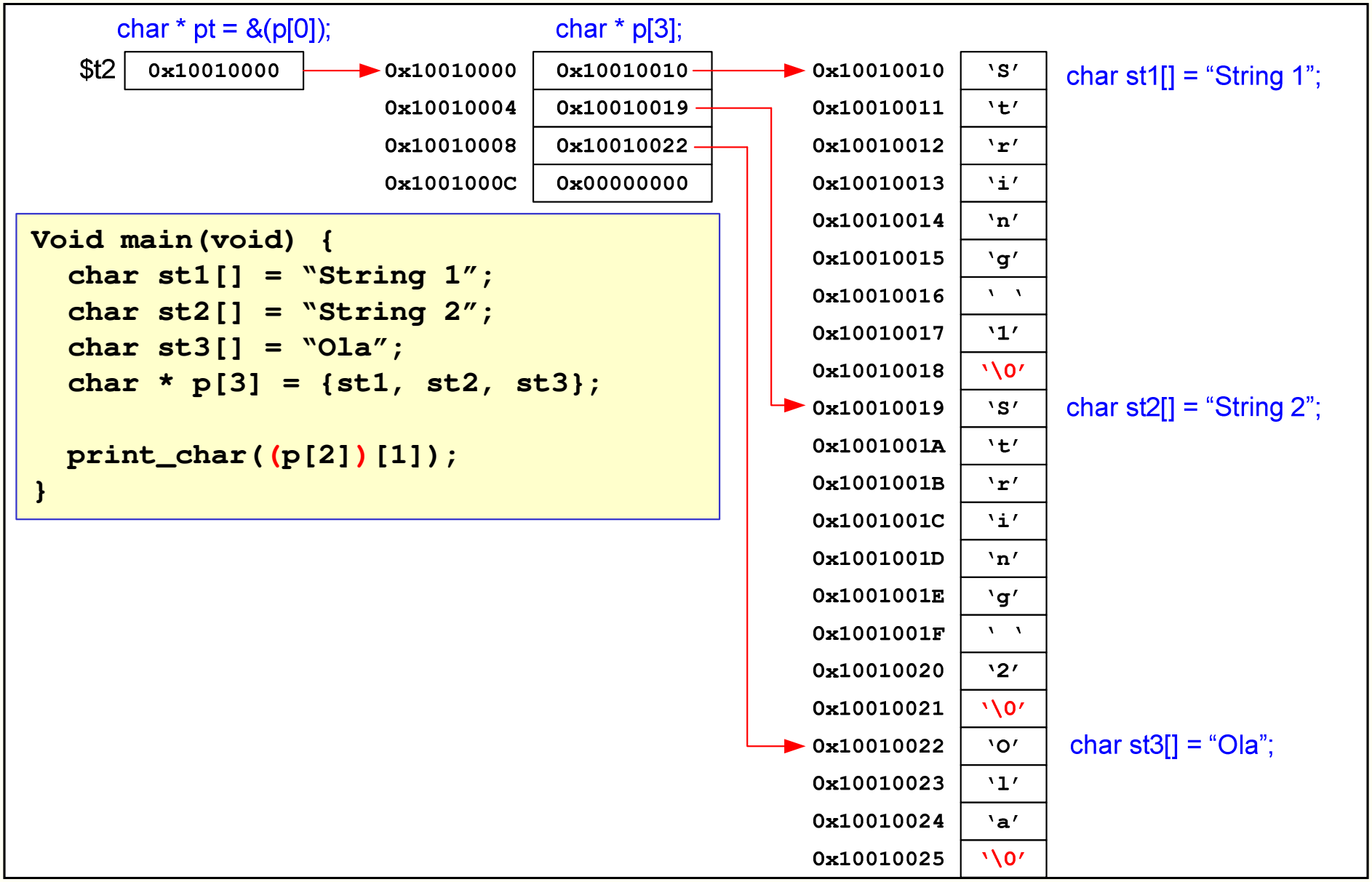
- Utilização de ponteiros em linguagem C – arrays de ponteiros para caracter.
- Tradução para assembly.

Bernardo Cunha, José Luís Azevedo, Arnaldo Oliveira

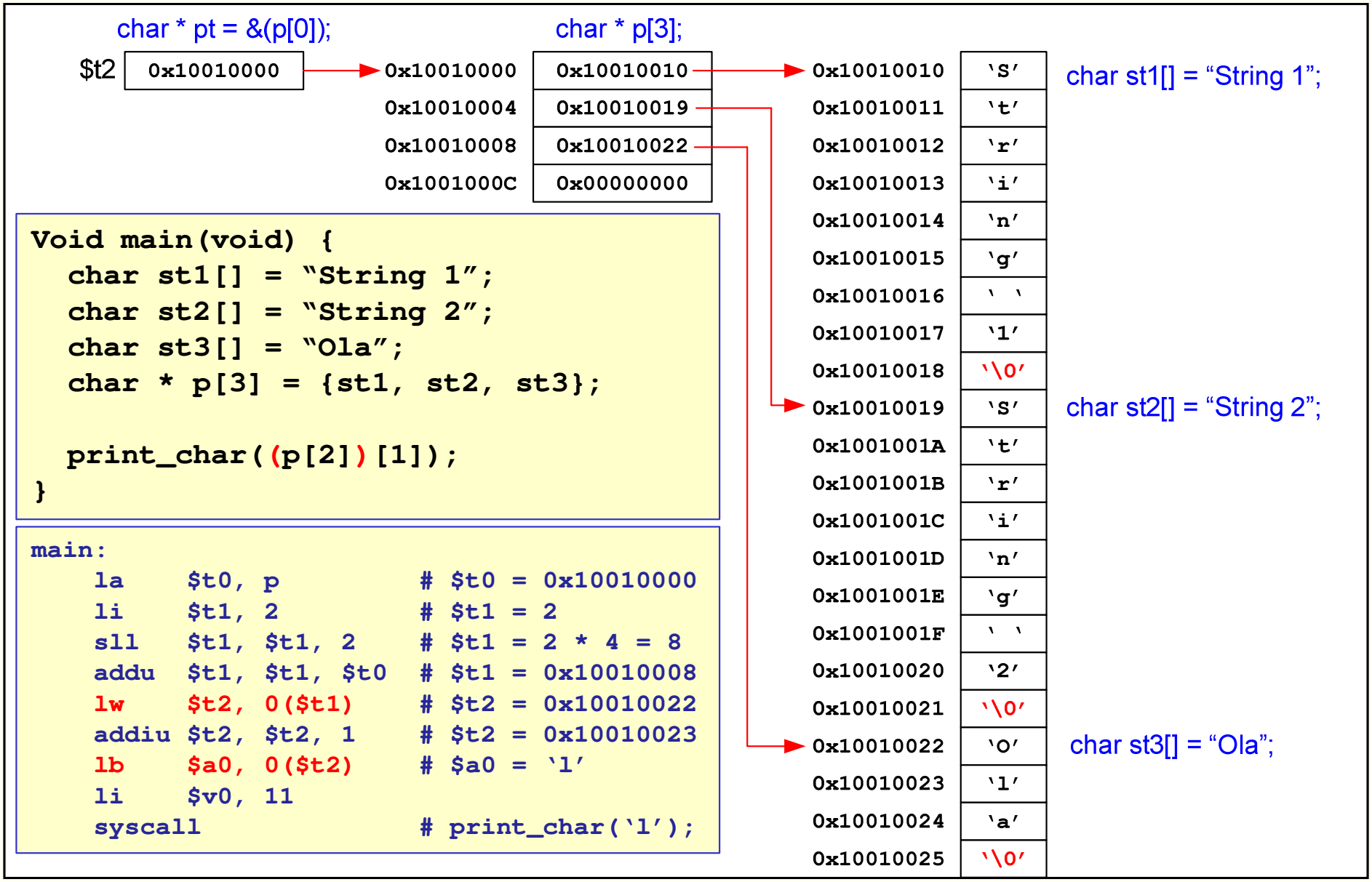
Arrays de ponteiros para caracter.



Arrays de ponteiros para caracter.



Arrays de ponteiros para caracter.




```
Void main(void) {
  char st1[] = "String 1";
  char st2[] = "String 2";
  char st3[] = "Ola";
  char * p[3] = {st1, st2, st3};

  print_char((p[2])[1]);
}
```

```
main:
    la    $t0, p          # $t0 = 0x10010000
    li    $t1, 2          # $t1 = 2
    sll   $t1, $t1, 2     # $t1 = 2 * 4 = 8
    addu  $t1, $t1, $t0   # $t1 = 0x10010008
    lw    $t2, 0($t1)     # $t2 = 0x10010022
    addiu $t2, $t2, 1     # $t2 = 0x10010023
    lb    $a0, 0($t2)     # $a0 = 'l'
    li    $v0, 11
    syscall               # print_char('l');
```

O array 'p' não contém Strings, mas sim ponteiros! cada ponteiro ocupa 4 bytes (dado que o MIPS é uma Arquitetura de 32 bits (4 bytes)). Assim

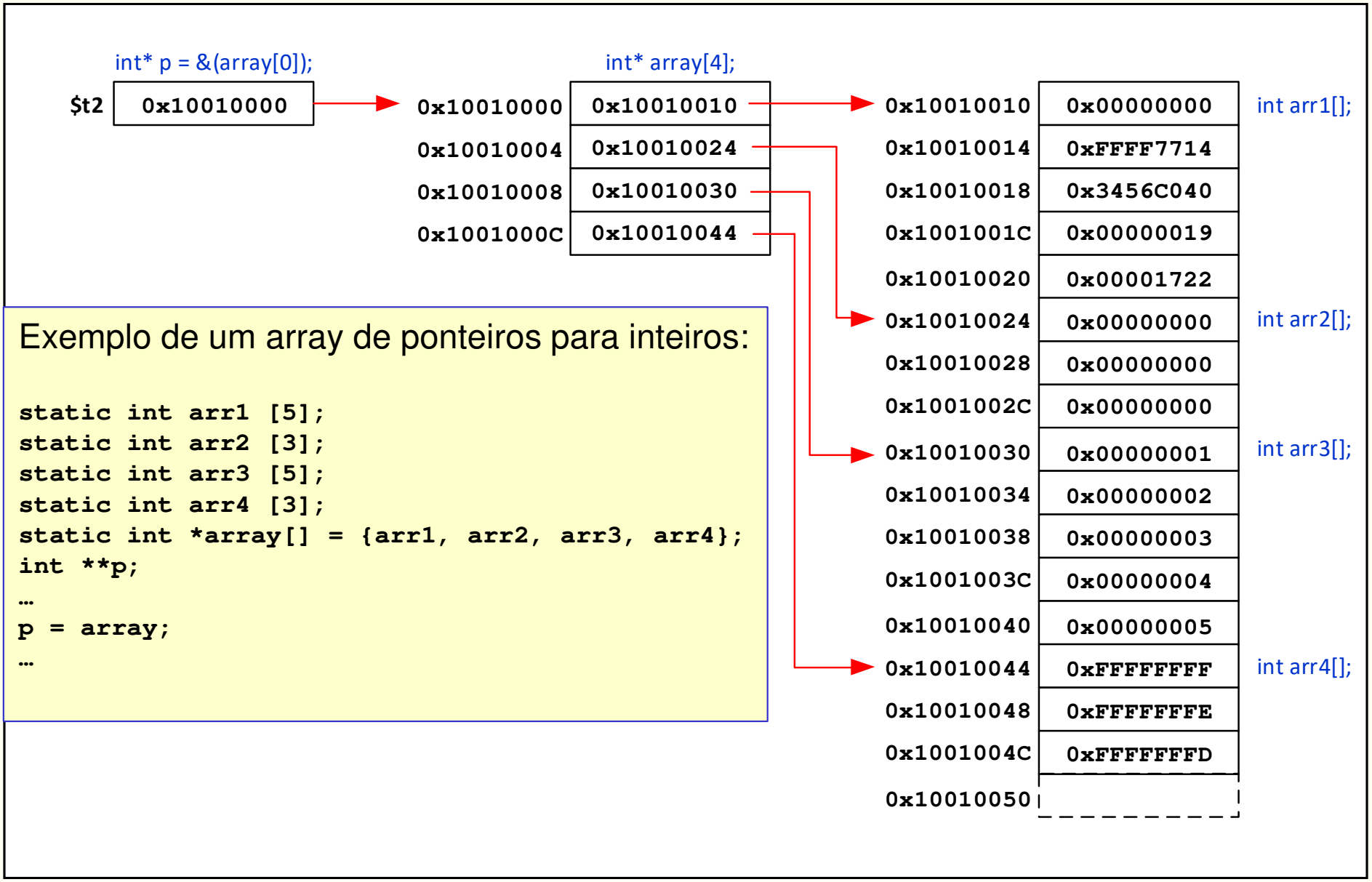
```
        .eqv    SIZE, 3
        .data
s0:     .asciiz "Array"
s1:     .asciiz "de"
s2:     .asciiz "ponteiros"
array:  .word   s0, s1, s2

        .text
        .globl main

# MAPA DE REGISTOS
# i:            $t0
# &(array[0]):  $t1
# &(array[i]):  $t2
main:                           # void main(void){
        li      $t0, 0          #       int i = 0;
        la      $t1, array      #       &(array[0]);
while:                          #       while(){
        bge     $t0, SIZE, endw #               if( i >= SIZE ) break;
                                #  Ao fazer i * 4 estamos a avançar de ponteiro em ponteiro
                                #  e não dentro das strings. Cada String "array", "de", "ponteiros"
                                #  está armazenada num endereço separado, e é o ponteiro armazenado no
                                #  'array' que aponta para o inicio de cada string.cada
        sll     $t4, $t0, 2     #               i *= 4;
        addu    $t2, $t1, $t4   #               &(array[i]);
        li      $v0, 4          #
        lw      $a0, 0($t2)     #
        syscall                 #               print_string(array[i]);
        li      $v0, 11         #
        li      $a0, '\n'       #
        syscall                 #               print_char('\n')
        addi    $t0, $t0, 1     #               i++;
        j       while           #
endw:                           #       }
        jr      $ra             #}
```

A iteração i*4 permite-nos mover de ponteiro em ponteiro no array, e não dentro das strings.

Arrays de ponteiros para inteiros.



Exemplo de um array de ponteiros para inteiros:

```
static int arr1 [5];
static int arr2 [3];
static int arr3 [5];
static int arr4 [3];
static int *array[] = {arr1, arr2, arr3, arr4};
int **p;
…
p = array;
…
```