ARCHITETTURA DEI CALCOLATORI

[Esempi in Verilog]

A CURA DI ALESSANDRO PAGHI

PROFESSORE: Roberto Giorgi (http://www3.diism.unisi.it/people/person.php?id=73&aa=2015)

LINK AL CORSO ANNO 2015/2016:
http://www3.diism.unisi.it/FAC/index.php?bodyinc=didattica/inc.insegnamento.php&id=55204&aa=2015

FREQUENTAZIONE: Sconsigliata.

```
#Ciclo For

.data

    str_stop:   .asciiz "Stop For\n"
    str_cont:   .asciiz "Continue For\n"

.text

main:

    #STAT1
    add     $t0, $zero, $zero

    for:

    #STAT2
    slt     $t1, $t0, 10

    #CONTROLLO TRUE
    beq     $t1, 0, fine

    #STAT4
    la      $a0, str_cont
    addi    $v0, $zero, 4
    syscall

    #STAT3
    addi    $t0, $t0, 1
    j for

    fine:
    la      $a0, str_stop
    addi    $v0, $zero, 4
    syscall

    addi    $v0, $zero, 10
    syscall
```

```
#Ciclo While

.data
    str_min:    .asciiz "Minore di 10\n"
    str_mag:    .asciiz "Maggiore di 10\n"
.text

main:

    inizio:

    #STAT1
    slt     $t1, $t0, 10

    #CONTROLLO TRUE
    beq     $t1, 0, fine

    #STAT2
    addi    $t0, $t0, 1
    j       inizio

    fine:
    la      $a0, str_mag
    add     $v0, $zero, 4
    syscall

    add     $v0, $zero, 10
    syscall
```

```
#Ciclo Do While

.data

    str_cont:   .asciiz "Cont Do\n"
    str_fine:   .asciiz "Fine Do\n"

.text
    main:

    do:

    #STAT1
    addi    $t0, $t0, 1

    la $a0, str_cont
    addi $v0, $zero, 4
    syscall

    #STAT2
    slt     $t1, $t0, 10

    #CONTROLLO TRUE
    bne     $t1, $zero, do

    la $a0, str_fine
    addi $v0, $zero, 4
    syscall

    addi    $v0, $zero, 10
    syscall
```

```
#Case Switch

.data

    str_print:  .asciiz "Inserisci Valore: "
    str_caso0:  .asciiz "Valore 0\n"
    str_caso1:  .asciiz "Valore 1\n"
    str_caso2:  .asciiz "Valore 2\n"
    str_caso3:  .asciiz "Valore 3\n"
    str_def:    .asciiz "Valore Def\n"

.text

main:

    la $a0, str_print
    addi $v0, $zero, 4
    syscall

    addi $v0, $zero, 5
    syscall

    add $s0, $zero, $v0

    #SWITCH

    bne $s0, 0, case1
    la $a0, str_caso0
    addi $v0, $zero, 4
    syscall
    j fine

    case1:
    bne $s0, 1, case2
    la $a0, str_caso1
    addi $v0, $zero, 4
    syscall
    j fine

    case2:
    bne $s0, 2, case3
    la $a0, str_caso2
    addi $v0, $zero, 4
    syscall
    j fine

    case3:
    bne $s0, 3, default
    la $a0, str_caso3
    addi $v0, $zero, 4
    syscall
    j fine

    default:
    la $a0, str_def
    addi $v0, $zero, 4
    syscall
```

```
fine:
addi $v0, $zero, 10
syscall
```

```
.data
    str_ins:        .asciiz "Fattoriale di = "
    str_ris:        .asciiz "Risultato = "

.text
.globl main

fact:

    addi $sp, $sp, -8
    sw $a0, 4($sp)          #alloco $a0 con offset di 4
    sw $ra, 0($sp)          #alloco $ra con offset di 0

    beq $a0, $zero, fine

    addi $a0, $a0, -1
    jal fact

fine:

    lw $t0, 0($sp)          #dealloco $ra
    lw $t1, 4($sp)          #dealloco $a0

    bne $t1, $zero, second
    addi $v1, $zero, 1
    j first

    second:
    mult $v1, $t1
    mflo $v1

    first:

    addi $sp, $sp, 8

    jr $t0

main:

    la $a0, str_ins
    addi $v0,$zero,4
    syscall

    addi $v0, $zero, 5
    syscall

    add $a0, $zero, $v0

    jal fact

    la $a0, str_ris
    addi $v0,$zero,4
    syscall

    add $a0, $zero, $v1
    addi $v0,$zero,1
    syscall
```

```
    addi $v0, $zero, 10
    syscall
```

```
.data

    datai:      .word 1
    dataf:      .float 2.3
    datad:      .double 4.5

    veti_3:     .word 0, 0, 0
    vetf_3:     .float 0.0, 0.0, 0.0
    vetd_3:     .double 0.0, 0.0, 0.0

    mati_3x3:   .word 0, 0, 0, 0, 0, 0, 0, 0, 0
    matf_3x3:   .float 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
    matd_3x3:   .double 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0

    str_prov:   .asciiz "Prova\n"

.text

main:

    #Copia di una mat-vettore di float in memoria
    ##############################################
    la      $t0, mat_f3x3           #indirizzo di matf_3x3
    addi    $sp, $sp, -36           #spazio per matf_3x3

    lwc1    $f0, 0($t0)             #1° valore di matf_3x3
    swc1    $f0, 0($sp)             #alloco il valore
    lwc1    $f0, 4($t0)             #2° valore di matf_3x3
    swc1    $f0, 4($sp)             #alloco il valore
    lwc1    $f0, 8($t0)             #3° valore di matf_3x3
    swc1    $f0, 8($sp)             #alloco il valore
    lwc1    $f0, 12($t0)            #4° valore di matf_3x3
    swc1    $f0, 12($sp)            #alloco il valore
    lwc1    $f0, 16($t0)            #5° valore di matf_3x3
    swc1    $f0, 16($sp)            #alloco il valore
    lwc1    $f0, 20($t0)            #6° valore di matf_3x3
    swc1    $f0, 20($sp)            #alloco il valore
    lwc1    $f0, 24($t0)            #7° valore di matf_3x3
    swc1    $f0, 24($sp)            #alloco il valore
    lwc1    $f0, 28($t0)            #8° valore di matf_3x3
    swc1    $f0, 28($sp)            #alloco il valore
    lwc1    $f0, 32($t0)            #9° valore di matf_3x3
    swc1    $f0, 32($sp)            #alloco il valore

    add     $a0, $zero, $sp         #$a0 = ind. copia di matf_3x3

    #float mat[i][j] 3x3
    #$s0 = j - $s1 = i - $a0 = ind.mat
    ##############################################
    sll     $t0, $s0, 2             #$t0 = j*4
    addi    $t1, $zero, 12          #$t1 = 3*4 = 12
    mult    $s1, $t1                #i*3*4
    mflo    $t2                     #$t2 = i*3*4
    add     $t2, $t2, $t0           #$t2 = j*4 + i*3*4
    add     $t2, $t2, $a0           #aggiungo indirizzo iniziale di mat

    lwc1    $f6, 0($t2)             #load float mat[k][j]
```

```
#double mat[i][j] 3x3
#$s0 = j - $s1 = i - $a0 = ind.mat
###############################################
sll     $t0, $s0, 3             #$t0 = j*8
addi    $t1, $zero, 24          #$t1 = 3*8 = 24
mult    $s1, $t1                #i*3*8
mflo    $t2                     #$t2 = i*3*8
add     $t2, $t2, $t0           #$t2 = j*8 + i*3*8
add     $t2, $t2, $a0           #aggiungo indirizzo iniziale di mat

ldc1    $f6, 0($t2)             #load double mat[k][j]

#SYSCALL1#############################
#Print Integer - $a0=value############
la      $s0, datai
lw      $a0, 0($s0)
addi    $v0, $zero, 1
syscall

#SYSCALL2#############################
#Print Float - $f12=value#############
la      $s0, dataf
lwc1    $f12, 0($s0)
addi    $v0, $zero, 2
syscall

#SYSCALL3#############################
#Print Double - $f12=value############
la      $s0, datad
ldc1    $f12, 0($s0)
addi    $v0, $zero, 3
syscall

#SYSCALL4#############################
#Print String - $a0=ind.stringa######
la      $a0, str_prov
addi    $v0, $zero, 4
syscall

#SYSCALL5#############################
#Read Integer - $v0=value#############
addi    $v0, $zero, 5
syscall

#SYSCALL6#############################
#Read Float - $f0=value###############
addi    $v0, $zero, 6
syscall

#SYSCALL7#############################
#Read Double - $f0=value##############
addi    $v0, $zero, 7
syscall

#SYSCALL8#############################
#Read String##########################
```

```
    #$a0=indirizzo di allocamento########
    #$a1=num. caratteri da leggere +1####

    #SYSCALL9############################
    #Memory Allocation####################
    #$a0=num. byte da allocare############
    #$v0=indirizzo del blocco#############

    #SYSCALL10###########################
    #Uscita dal programma#################
    addi $v0, $zero, 10
    syscall

    #SYSCALL11###########################
    #Print Char - $a0=integer#############

    #SYSCALL12###########################
    #Read Char - $v0=result###############
```