Ho Chi Minh City University of Technology
Faculty of Computer Science and Engineering

Computer Architecture

Practical session - Week 7 Autumn Semester 2018

Notes:

- The main purpose of this week is to continually practice procedure calls, especially non-leaf procedures.
- Although there are some questions that can be solved without recursive calling, students should use function/procedure callings in recursive to practice.
- Students are NOT requested to submit the MIPS programs.

Question 1. Given the following leaf procedure in ANSI C

```
void swap(int v[], int k)
{
 int temp;
 temp = v[k]
 v[k] = v[k+1];
 v[k+1] = temp;
}
```

Assume that the `$a0` register will store the base address of the `v` array while the `$a1` register keeps value `k`. The array `v` consists of 10 elements in integer and is pre-defined in the data section.

1. Write a main program the receive value `k` from user, check the value `k` and call the procedure swap if possible.
2. Watch the `$ra` register before and after the `jal` and `jr` instructions are executed.

Question 2. Given the following factorial MIPS program in a recursive form (as in the slide)

```
fact: addi $sp, $sp, -8      # adjust stack for 2 items
  sw   $ra, 4($sp)       # save return address
  sw   $a0, 0($sp)       # save argument
  slti $t0, $a0, 1       # test for n < 1
  beq  $t0, $zero, L1
  addi $v0, $zero, 1     # if so, result is 1
  addi $sp, $sp, 8       #   pop 2 items from stack
  jr   $ra               #   and return
L1:  addi $a0, $a0, -1      # else decrement n
  jal  fact              # recursive call
  lw   $a0, 0($sp)       # restore original n
  lw   $ra, 4($sp)       #   and return address
  addi $sp, $sp, 8       # pop 2 items from stack
  mul  $v0, $a0, $v0     # multiply to get result
  jr   $ra               # and return
```

1. Type the above procedure and write a main program that call the above procedure with different `n`, where `n` is in the `$a0` register. Watch the results
2. When `n` is 2, run the program step by step and watch the execution of instructions as well as the `$ra` register and values store/load to/from the stack.

——————the end——————