CSSE 232
Computer Architecture I

Procedures II

Class Status

Reading for today

- 2.13

Outline

- Nested procedures
- Recursive procedures

Nested Procedures

- Procedures that call other procedures
- For nested call, caller needs to save on the stack:
  - Its return address
  - Any arguments and temporaries needed after the call
- Restore from the stack after the call

Nested Procedure Call

```
int add(int g, int h){
  int f;
  int i;
  i = func(5);
  f = g + h + i;
  return f;
}
```

Arguments g and h in $a0, $a1. Variable f must be stored in $s0. Result goes in $v0

Nested Procedure Call

```
int add(int g, int h){
  int f;
  int i;
  i = func(5);
  f = g + h + i;
  return f;
}
```

Arguments g and h in $a0, $a1. Variable f must be stored in $s0. Result goes in $v0

```
 Add:
 addi $sp, $sp, -16 #allocate storage for 4 items on stack
 sw   $ra, 0($sp)
 sw   $a0, 4($sp)
 sw   $a1, 8($sp)
 sw   $s0, 12($sp)
 addi $a0, $0, 5    #prepare argument
 jal  func          #make procedure call
 lw   $ra, 0($sp)   #restore some stack data
 lw   $a0, 4($sp)
 lw   $a1, 8($sp)
 add  $s0, $a0, $a1
 add  $s0, $s0, $v0
 add  $v0, $s0, $0  #prepare return value
 lw   $s0, 12($sp)  #finally restore s0
 addi $sp, $sp, 16  #restore stack pointer
 jr   $ra           #return to caller
```

Recursive Example

```
int fact (int n)
{
  if (n < 1)
    return 1;
  else
    return n * fact(n - 1);
}
```

Argument n in $a0, result in $v0

Recursive Example

```
int fact (int n)
{
  if (n < 1)
    return 1;
  else
    return n * fact(n - 1);
}
```

Argument n in $a0, result in $v0

```
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
```

Questions?

- Nested procedures
- Recursive procedures