Stack

interrupts/exceptions 6.7.2, 7.7

Stack grows down from Ox7fff
points to Ox00010000

Section 6.4.6

precondition

do not call

no args, no return value

args, return value

local & global variables

Memory Allocation (fig 6.3), "Green Card"

sp ← stack

sp ← kernel

sp ← static

sp ← text

Fig 6.23 (modified)

main() {
    0x00 main: jal simple
    nop
}

void simple() {
    0x1000 simple: jr $ra
    nop
}

What goes into $ra in jal?

What is the line after at simple+4 is jr $0?

000000000000010000

Op
simple args, return value
int y;
0x1000 addi $a0, $0, 3
1004 addi $a1, $0, 8
y = plus (3, 8);
1008 jal plus
1010 add $s0, $v0, $0

int plus (int a, b) { 2008 plus:
add $v0, $a1, $a1
jr $ra
hop

return a+b;

more than 4 args:
1st 4 go in $a0-3
rest go on stack.

what if I need $v0?
use it. Caller responsibility
what if I need $s0?
$ra?
$a0?
#v0?
expected.
what if I need extra space?
local vars on stack.

could we get rid of the nop after jal?
no
after jr?
no
what is $ra in $ra at 0x200c 1010
could we swap 1004 & 1008?
yes
could we swap 100c & 1010?
no

when would I need to save $ra?
if procedure calls another.

text function() {
int a, b; c[10];
int $v0, $sp, $s1
addi $sp, $sp, -10
sw $s0, 0($sp)
sw $s1, 36($sp)
c[0] = a;
c[9] = b;
}
Stack frame: fig 6.27

Before & after

Main

a() { }

b() { }

c() { }