

OSI Handout Week02.a

1. Background: RISC-V assembly I

a) registers
[see "RISC-V registers" in reference page]

b) **addi** rd, rs1, immediate

rd = rs1 + immediate

c) **sw** rs2, offset(rs1)

Memory[rs1 + offset] = rs2

d) **mv** rd, rs1

rd = rs1

e) **call** rd, symbol

rd = pc+4
pc = &symbol

If rd is omitted, ra is implied.

f) **ret**

pc = ra

2. Context switch in user-space:

a) void ctx_start(void** old_sp, void* new_sp);

This will be used when starting a new thread.
It will save registers on the old stack,
store current stack pointer to "old_sp",
switch stack to the "new_sp",
and
finally call ctx_entry().

b) void ctx_switch(void** old_sp, void* new_sp);

This will be used for context switch.
It will save registers on the old stack,
store current stack pointer to "old_sp",
switch stack to the "new_sp",
and
restore registers from the new stack,
finally return (to ra).

3. grass/context.S

```

1  ctx_start:
2      addi sp,sp,-64
3      sw s0,4(sp)      /* Save callee-saved registers */
4      sw s1,8(sp)
5      sw s2,12(sp)
6      sw s3,16(sp)
7      sw s4,20(sp)
8      sw s5,24(sp)
9      sw s6,28(sp)
10     sw s7,32(sp)
11     sw s8,36(sp)
12     sw s9,40(sp)
13     sw s10,44(sp)
14     sw s11,48(sp)
15     sw ra,52(sp)     /* Save return address */
16     sw sp,0(a0)     /* Save the current stack pointer */
17     mv sp,a1        /* Switch the stack */
18     call ctx_entry /* Call ctx_entry() */
19
20  ctx_switch:
21     addi sp,sp,-64
22     sw s0,4(sp)     /* Save callee-saved registers */
23     sw s1,8(sp)
24     sw s2,12(sp)
25     sw s3,16(sp)
26     sw s4,20(sp)
27     sw s5,24(sp)
28     sw s6,28(sp)
29     sw s7,32(sp)
30     sw s8,36(sp)
31     sw s9,40(sp)
32     sw s10,44(sp)
33     sw s11,48(sp)
34     sw ra,52(sp)   /* Save return address */
35     sw sp,0(a0)   /* Save the current stack pointer */
36     mv sp,a1     /* Switch the stack */
37     lw s0,4(sp)  /* Restore callee-saved registers */
38     lw s1,8(sp)
39     lw s2,12(sp)
40     lw s3,16(sp)
41     lw s4,20(sp)
42     lw s5,24(sp)
43     lw s6,28(sp)
44     lw s7,32(sp)
45     lw s8,36(sp)
46     lw s9,40(sp)
47     lw s10,44(sp)
48     lw s11,48(sp)
49     lw ra,52(sp) /* Restore return address */
50     addi sp,sp,64
51     ret

```

4. An example use of `ctx_start+ctx_entry`

```
void thread_create(void (*f)(void *), void *arg, unsigned int stack_size) {  
    tcb = create_thread_control_block();  
    old_tcb = current_running_thread_control_block();  
    ... // do something necessary  
  
    void **old_sp = ... // old stack pointer's address in old_tcb  
    void *new_sp = ... // new stack pointer in tcb  
  
    ctx_start(old_sp, new_sp);  
}  
  
void ctx_entry(void){  
    ... // do something useful  
    (*f)(arg); // run function "f" received by "thread_create"  
    ... // wrap up  
}
```