CS315-02 RISC-V Assembly 8

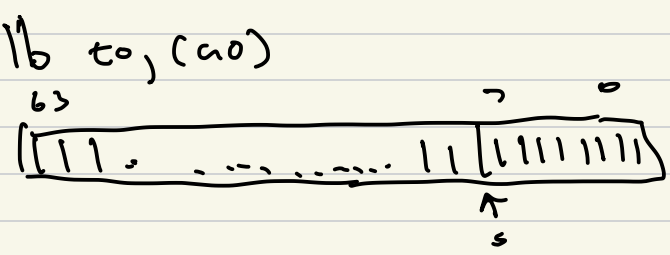
structs and linked lists

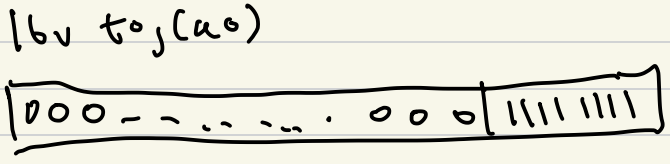
RISC-V Memory Instructions
loads / stores

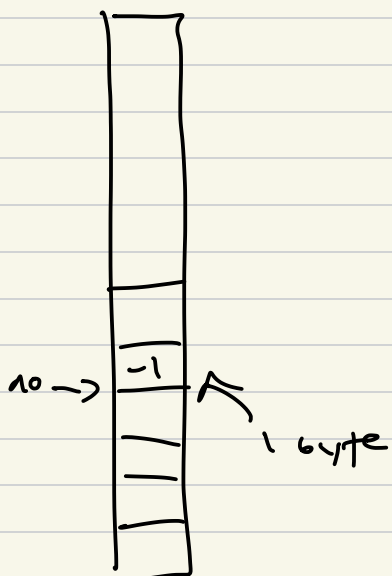
ld / sd	lw / sw	lb / sb	lh / sh
64 bit	32 bit	8 bit	16 bit
			halfword

lb load byte (sign extension)

lbu load byte unsigned







lw
lwu

lh
lhu

ld
~~tdu~~

all work the same

---

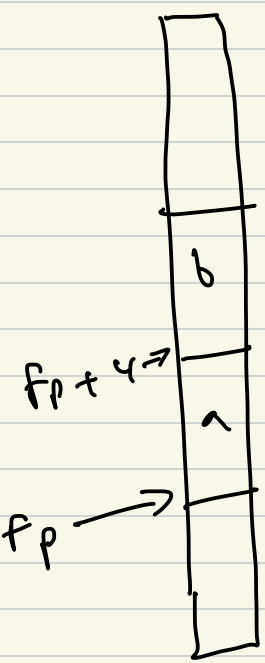
structs

```
Struct foo_st {
    int a;    // 0
    int b;    // 4
};

struct foo_st foo;

struct foo_st *fp;
fp = &foo;
```

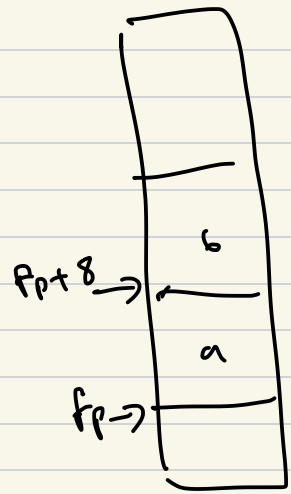
```
# a0 - *fp
get_a_s:
        lw a0, 0(a0)
        ret
```

```
get_b_s:
        lw a0, 4(a0)
        ret
```

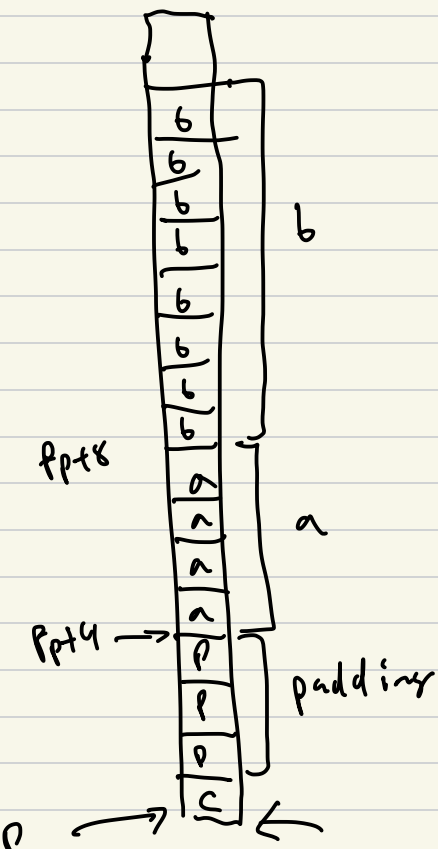
get_a_c (struct foo_st *fp)
get_b_c (struct foo_st *fp)



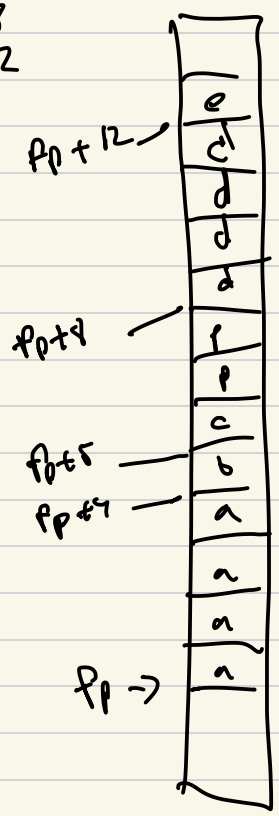
```
struct foo_st foo {
    int64_t a;
    int64_t b;
};

# struct foo_st
# a : offset 0
# b : offset 8
```



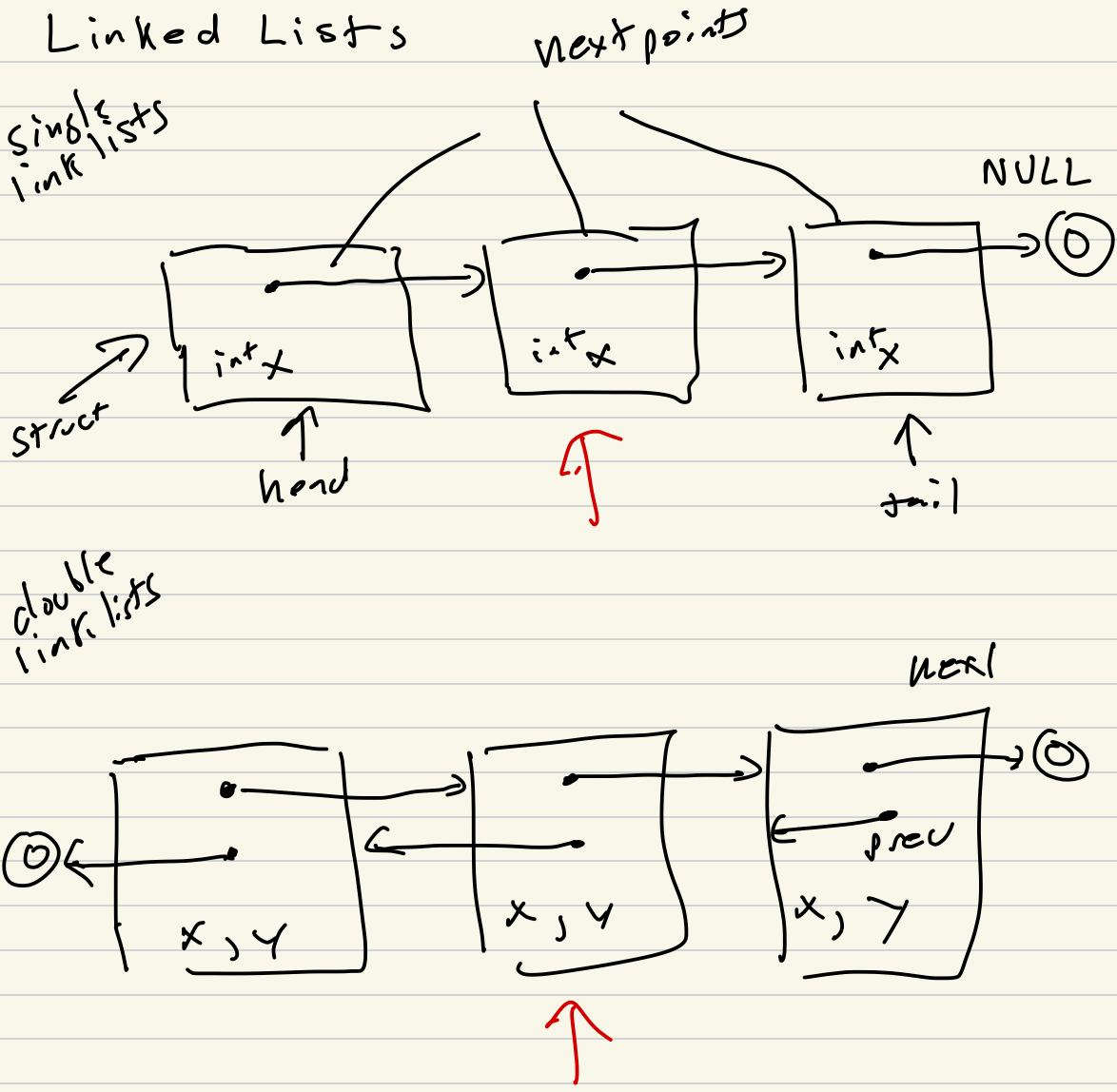
```
struct foo_st {
    char c;
    int a;
    int64_t b;
};

# struct foo_st
# c - offset 0
# a - offset 4
# b - offset 8
```



```
struct foo_st {
    int a;      # 0
    char b;     # 4
    char c;     # 5
    int d;      # 8
    char e;     # 12
};
```



Linked Lists



```
struct node_st {
    struct node_st *next_p;   # 0
    int value;                # 8
}

int *ip;
char *cp;
struct foo_st *fp;
```

all 8 bytes (64 bits)

```
main() {
    struct node_st n0, n1, n2, n3

    n0.value = 11
    n0.next_p = &n1

    n1.value = 22
    n1.next_p = &n0
```