# CS 315-01 RISC-V Assembly 8

Structs and linked lists

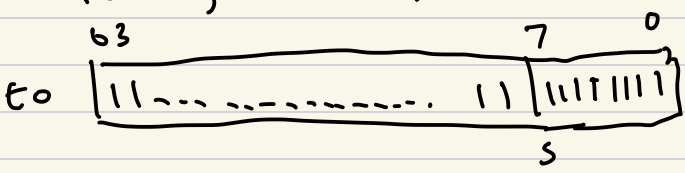
## RISC-V Memory Instructions

Loads / Stores

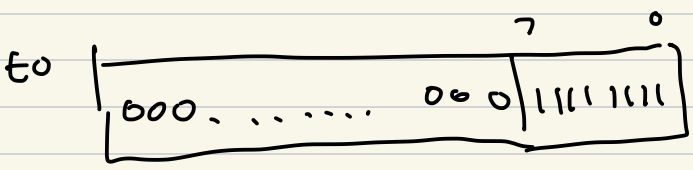
| ld / sd | lw / sw | lb / sb | lh / sh |
|---|---|---|---|
| 64 bits | 32 bits | 8 bits | 16 bits |

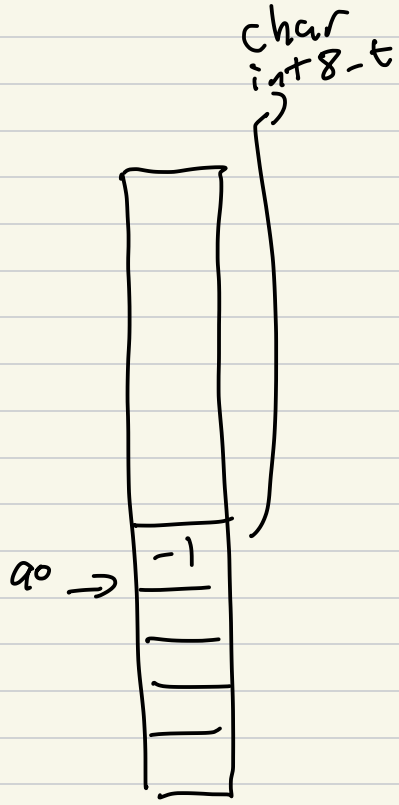
lb load byte (signed)
lbu load byte unsigned

lb t0, (a0) sign extension




lbu t0, (a0)



lw
lwu

lh
lhu

ld
~~ldu~~

all work the same

No "u" versions of stores

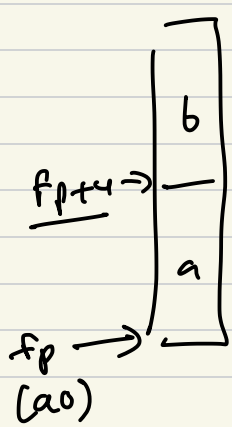
---

Structs

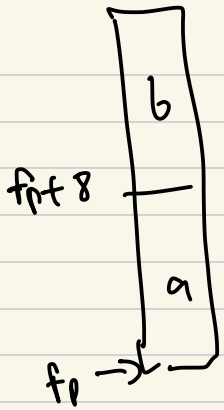
C

```
struct foo_st {
    int a;
    int b;
};

struct foo_st foo;

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

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

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

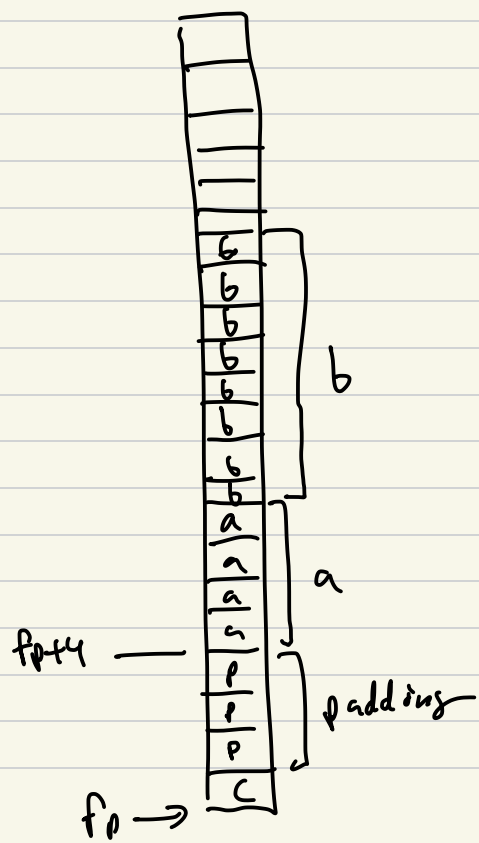


```
struct foo_st foo {
    int64_t a;
    int64_t b;
}

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



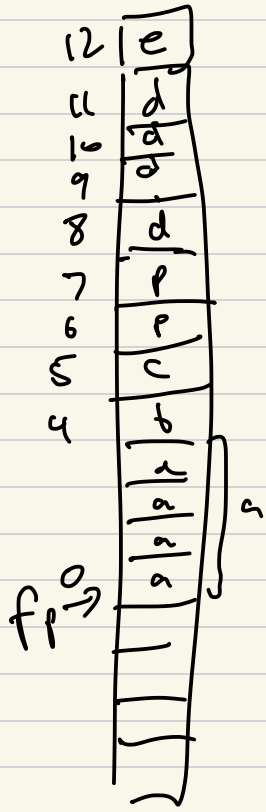

---

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

# struct foo_st
0   char c
4   int a
8   int64_t b
```

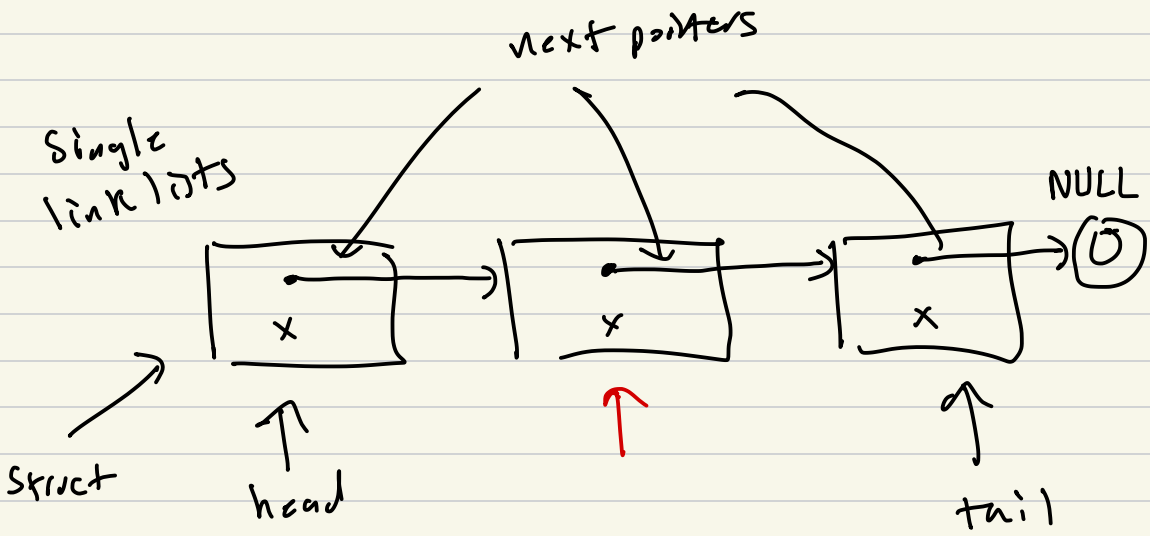


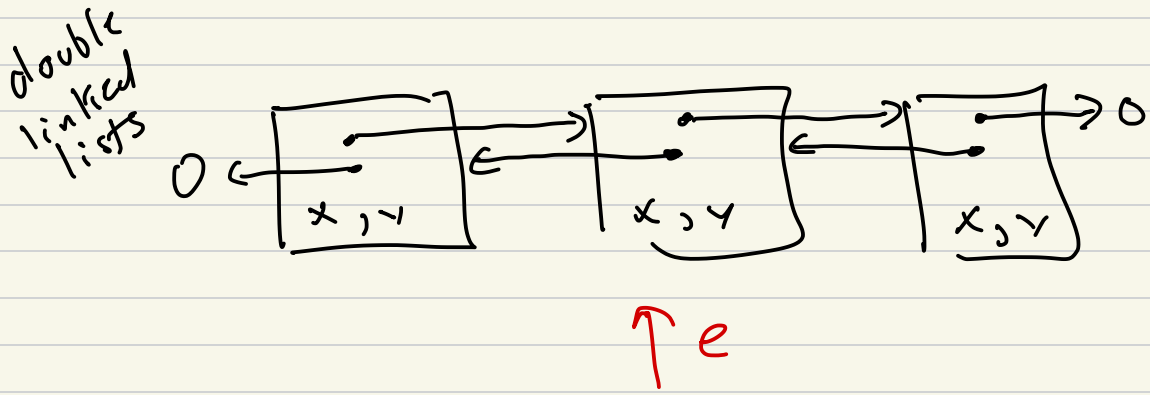
```
struct foo_st {
    int a;
    char b;
    char c;
    int d;
    char e;
}
```

```
# struct foo_st
# 0
# 4
# 5
# 8
# 12
```



# Linked Lists



```
struct node_st {
    struct node_st *nextp;      # offset: 0
    int value;                  # offset: 8
};
```

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

all 8 bytes (64 bit)

---

main()

```
n0.value = 11
n0.next_p = &n1

n1.value = 22
n1.next_p = &n2
```