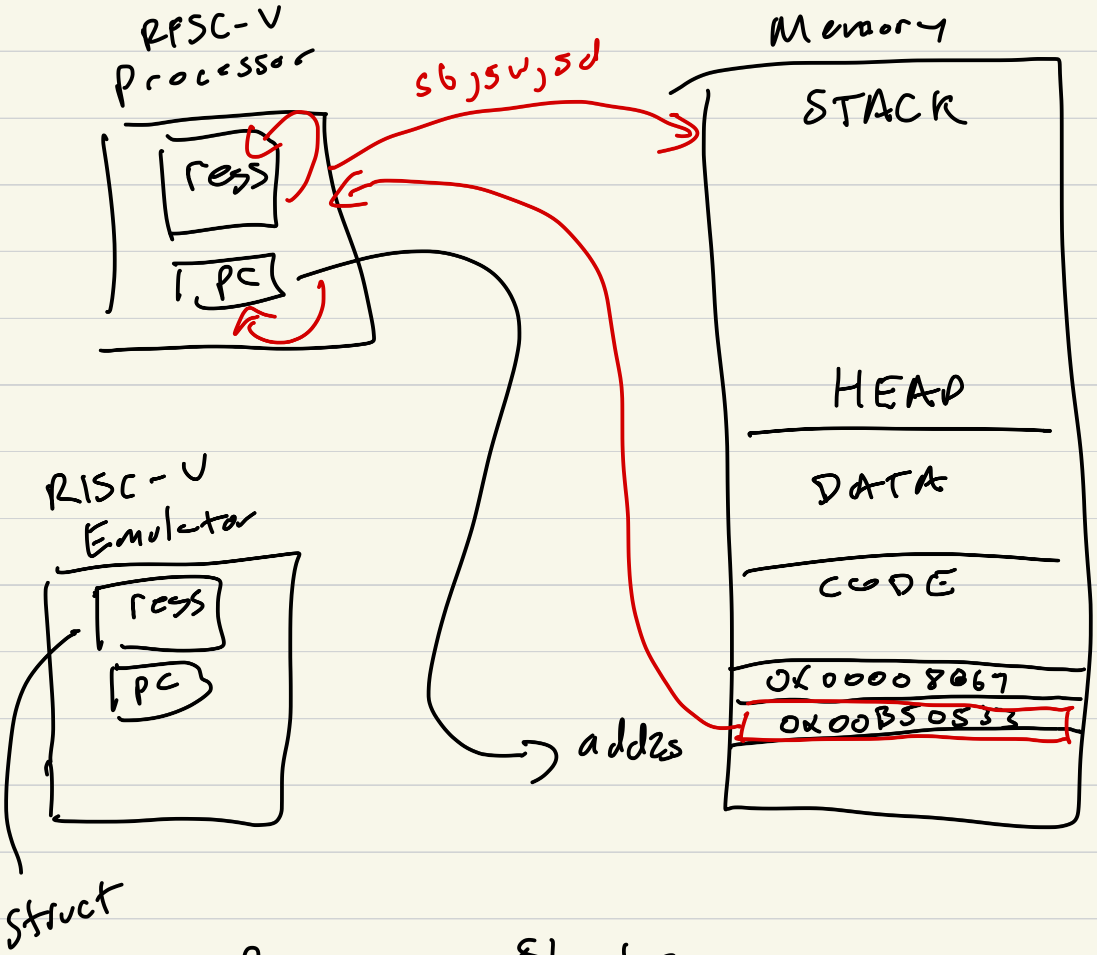
CS315-01 RISC-V Emulation



Processor State

Registers (32)
PC program counter
MEMORY — STACK

Implementation

Incremental approach

① Identify instruction ADDI

② Identify instruction format
i-type

③ Break down iw

0x0012 8293 addi $\underline{t0}$, $\underline{t0}$, $\underline{1}$

$$\underbrace{0000\ 0000\ 0001}_{\substack{\downarrow \\ 1}}\ \underbrace{0010\ 1}_{\substack{\text{rs1}\\ 5 \\ \text{reg} \\ \text{t0}}}\ \underbrace{000}_{\substack{\text{funct3}\\ 0}}\ \underbrace{0010\ 1}_{\substack{\text{rd}\\ 5 \\ \text{reg} \\ \text{t0}}}\ \underbrace{001\ 0011}_{\substack{\text{opcode}\\ \text{ADDI}}}$$

④ implement or add to
a type function.

5. get all fields for type

6. construct immediate value as needed.

7. update state

   a. update rd
   b. update pc

8. return

---

dealing with bits

```
uint32_t get_bits(uint64_t num,
                  uint32_t start,
                  uint3_t end)
```

rs1 = (iw >> $\underline{15}$) & 0b11111 ; ←

rs1 = get-bits(iw, $\underline{15}$, $\underline{5}$); ←

uint32_t get_bit (uint64_t num,
uint32_t which)

$\uparrow$ bit position

0b1

0b1 << 5

$\overbrace{1\ 00000}\ 0 \leftarrow$

-1

$\underbrace{0\ 11111}\ 1$

foo:

PC call boo $\longrightarrow$

PC+4 add

RA = PC+4

boo:

PC $\rightarrow$ add

save ra

call goo

restore ra

ret

---

immediates

addi t0, t1, $\underbrace{99}$

i-type immediate

[int64_t immU = get_bits(iw, 20, 12)

State update

$$\text{regs[rd]} = \underset{64}{\text{regs[rs1]}} + \underset{64}{\underline{\text{imm}}}$$

addi t0, t0, -3

0x FFD 2 8 2 9 3

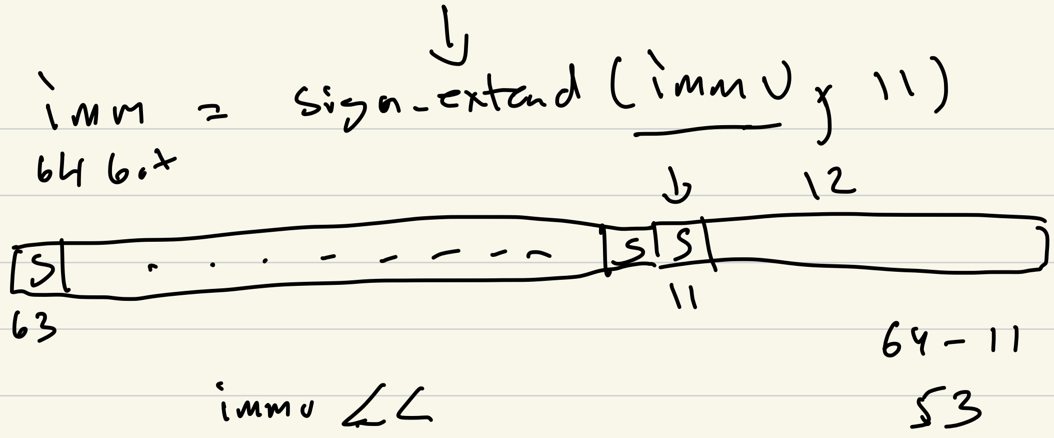
$\overbrace{\underbrace{1111\ 1111\ 1101}}^{11}$ 0010 1000 0010 $\underbrace{1001\ 0011}$

```
  0000 0000 0010
+              1
----------------
  0000 0000 0011  (3)
```

int64_t sign_extend(uint64 num,
                    uint32 start)

imm = sign_extend(immU, 11)

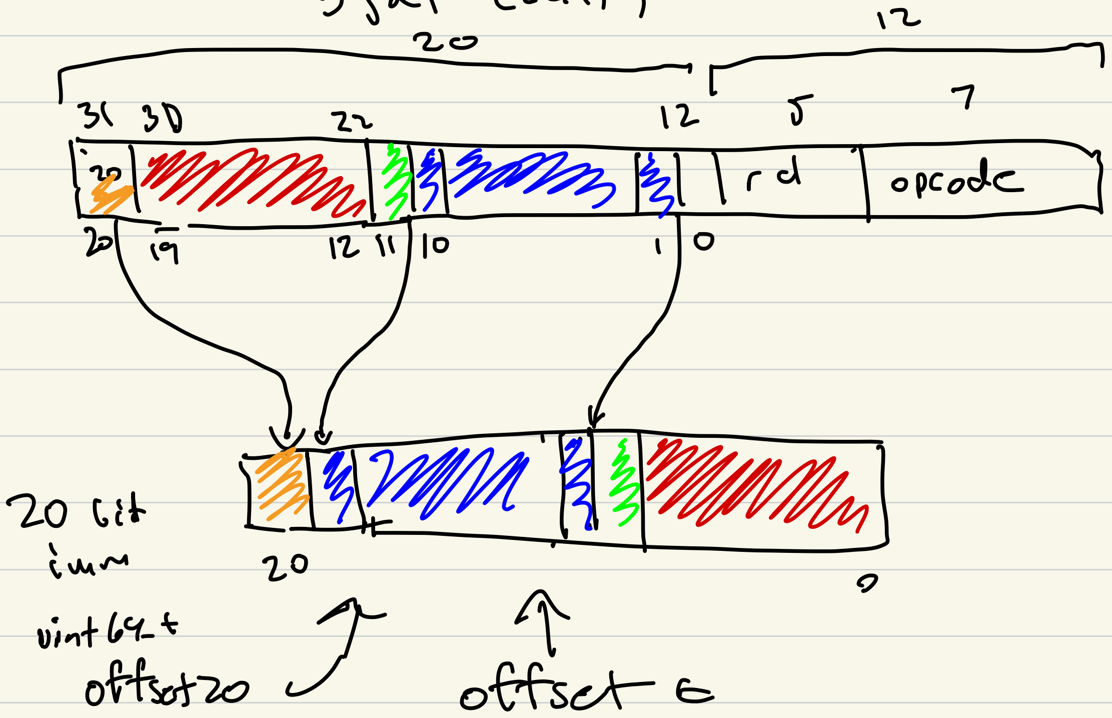
64 bit



immU <<

J-type immediates

JAL → jump (j)
→ jal (call)

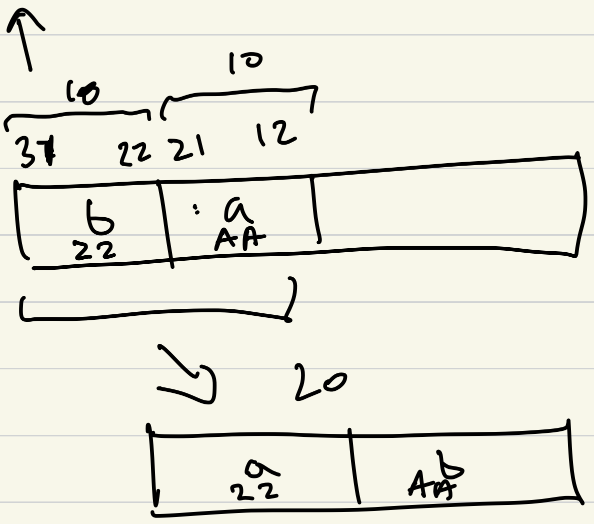


int64_t offset = signextend(offset20, 20)

offset = offset ~~*~~ 2;

pc = pc + offset

PC j loop



a = get_bits(iw, 12, 10)

b = get_bits(iw, 22, 10)

(a << 10) | b

```
   AA   00
|  AA   22
```