CS 315-01 Project 04 Midterm

Midterm on Thursday
You can bring one page of notes (Front and back)

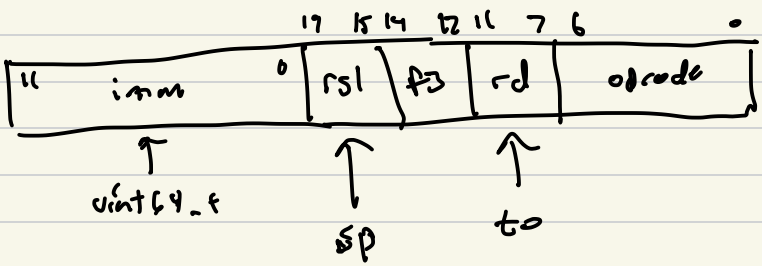
loads and stores

lw load word

lw t0, 8(sp)

offset (8) — base (sp)

target_address = base + offset
= base + imm



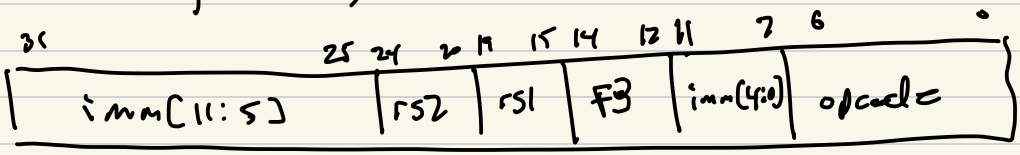
int64_t imm = sign_extend(vimm, 11)

uint64_t target_address;

target_address = rsp->regs[rs1] + imm;

rsp->regs[rd] = *((uint32_t *) target_address);
word

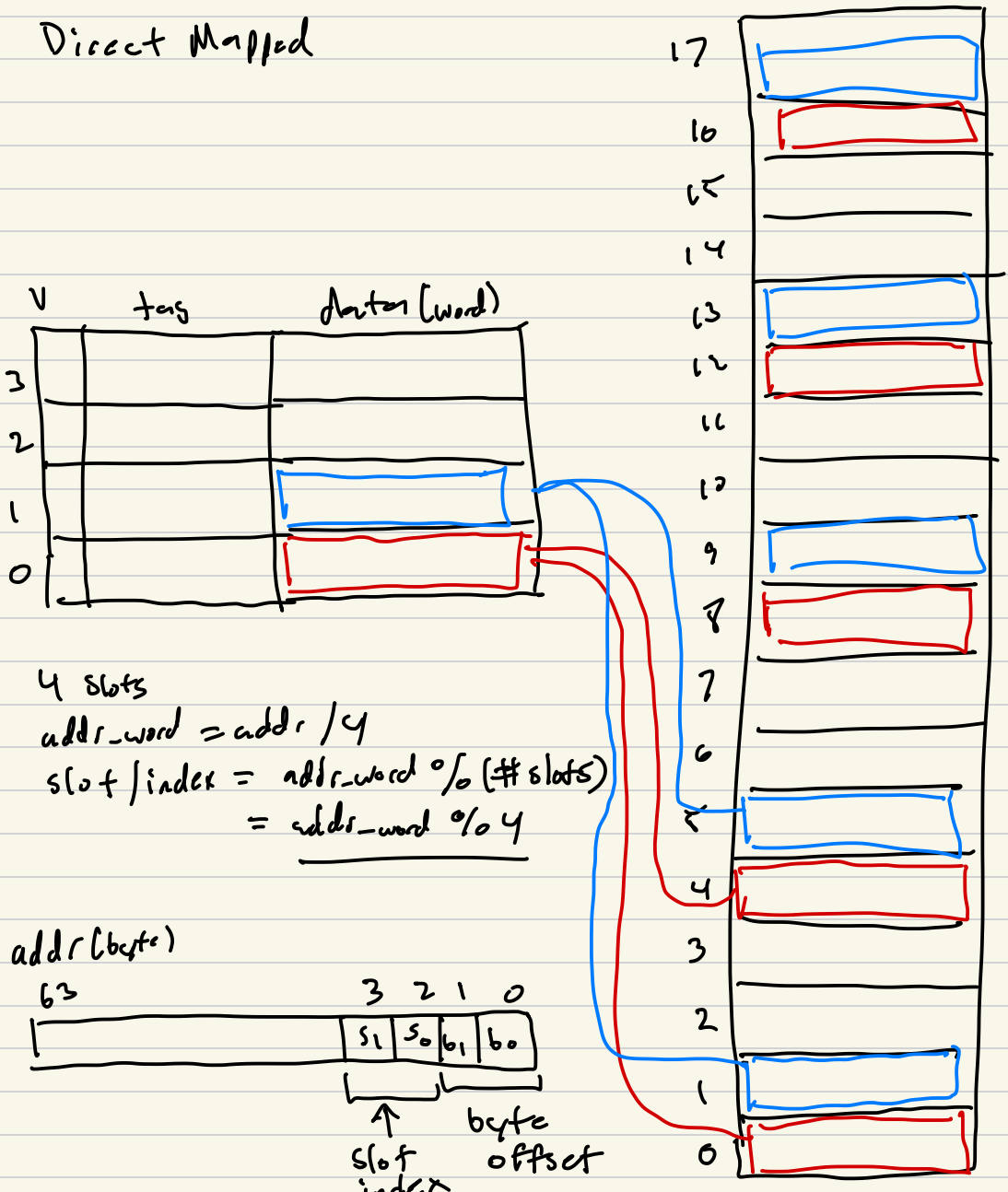
sw t0, 8(sp)




```
uint64_t imm11_5;
uint64_t imm4_0;

uint64_t uimm;

int64_t imm = sign_extend(uimm, 11)

uint64_t target_address;

target_address = rsp->regs[rs1] + imm;

*((uint32_t*) target_address = (uint32_t) rsp->regs[rs2];
```

Cache

Direct Mapped



4 slots

addr_word = addr / 4

slot index = addr_word % (# slots)

= addr_word % 4

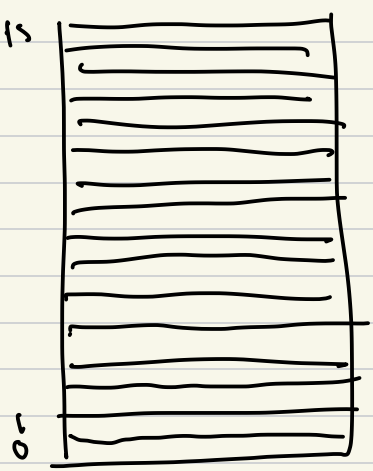
addr (byte)

slot_index = (addr >> 2) & 0b11;

Word addr	byte addr	bits (tags)
0	0	0000 0000 0000 0000
1	4	0000 0000 0000 0100
4	16	0000 0000 0001 0000
5	20	0000 0000 0001 0100
8	32	0000 0000 0010 0000
9	36	0000 0000 0010 0100

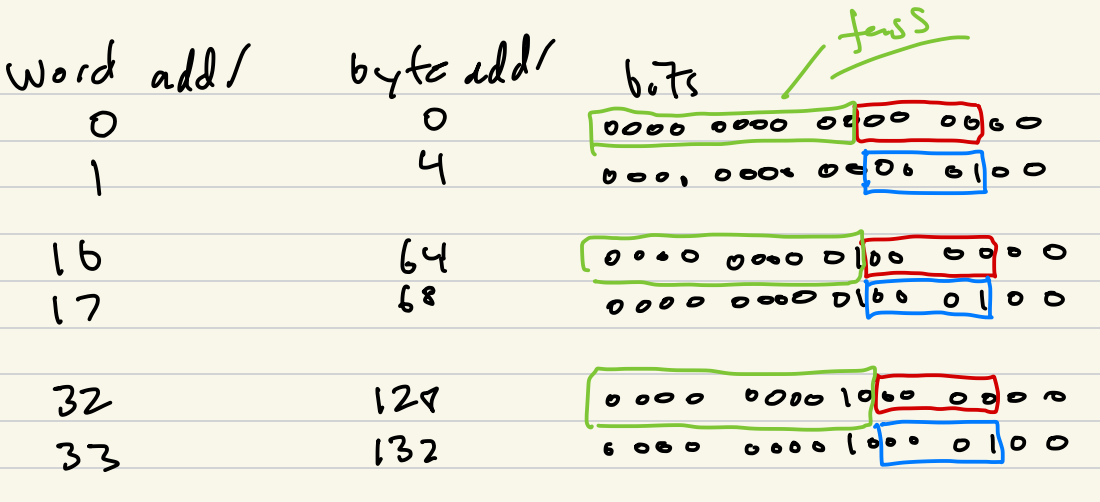
Cache Direct Mapped

#slots = 16

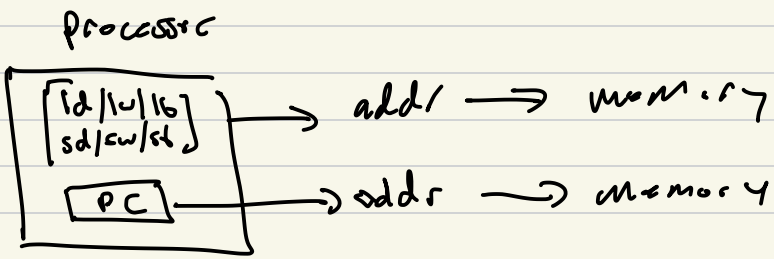


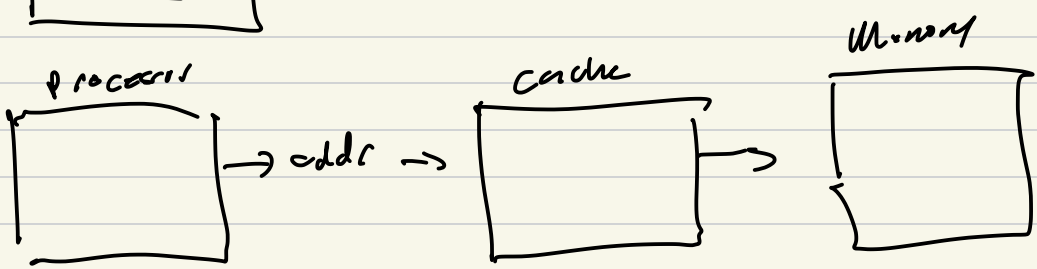
addr_word = addr / 4;

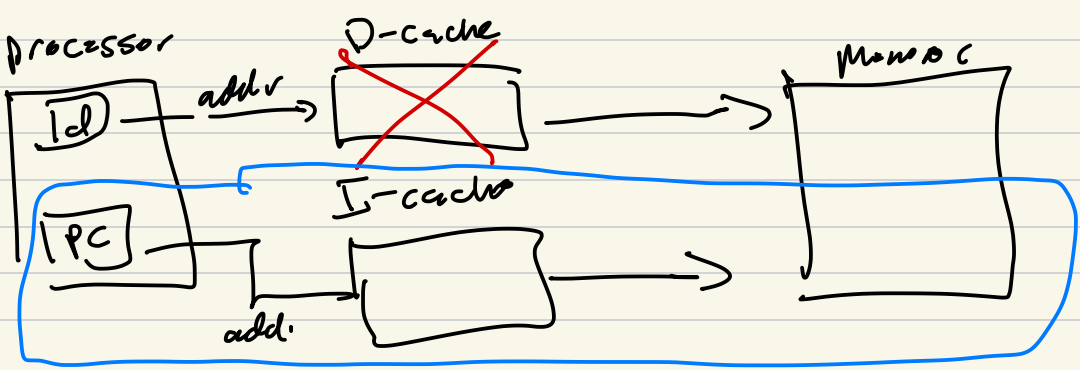
slot index = addr_word % 16



Instruction Cache vs Data Cache







Cache Block Size

Why?

Spatial

addr_word = addr / 4

addr_block = addr_word / 4

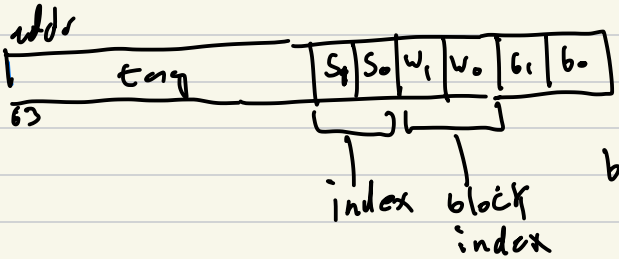
addr_block = addr / 16

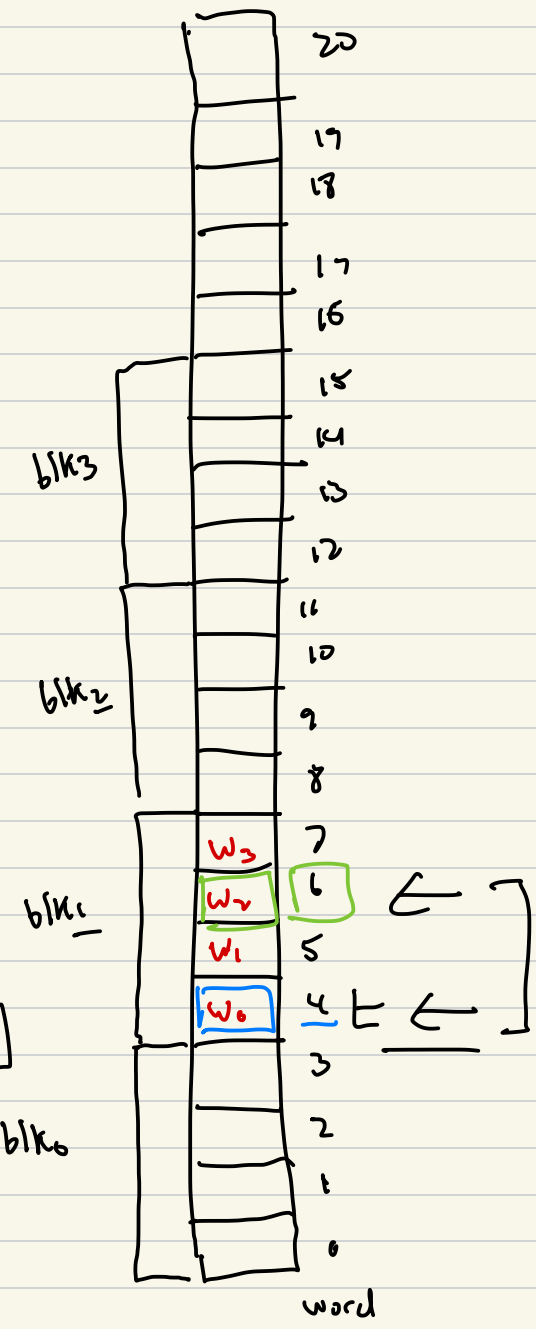
slotindex = addr_block % 4

addr = 16

addr_word = 4

addr_block = 1





slot_index = (addr >> 4) & 0b11;

block_index = (addr >> 2) & 0b11;

Miss

addr_word = 6
addr = 24

rem = addr_word % 4
= 6 % 4 = 2

addr_word_block_start = addr_word - rem;

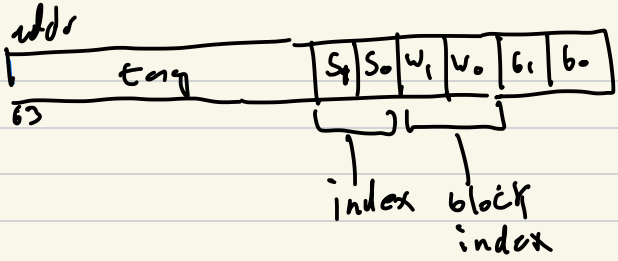
= 4

addr = addr_word * 4;

addr = 16

uint32_t block[4];

block[0] = *((uint32_t *) addr);
block[1] = *((uint32_t *) addr+4);



word addr	byte addr	bits
6	24	0000 0000 0001 1000
4	16	0000 0000 0001 0000

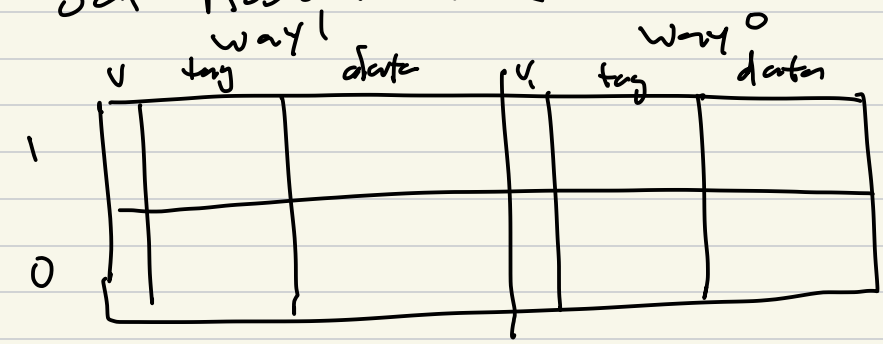
addr_block_start = (addr >> 4) << 4

addr_block_mask = 0xFFFF FFFF FFFF 0000

addr_block_start = addr & addr_block_mask

Cache

Set Associative



Set index