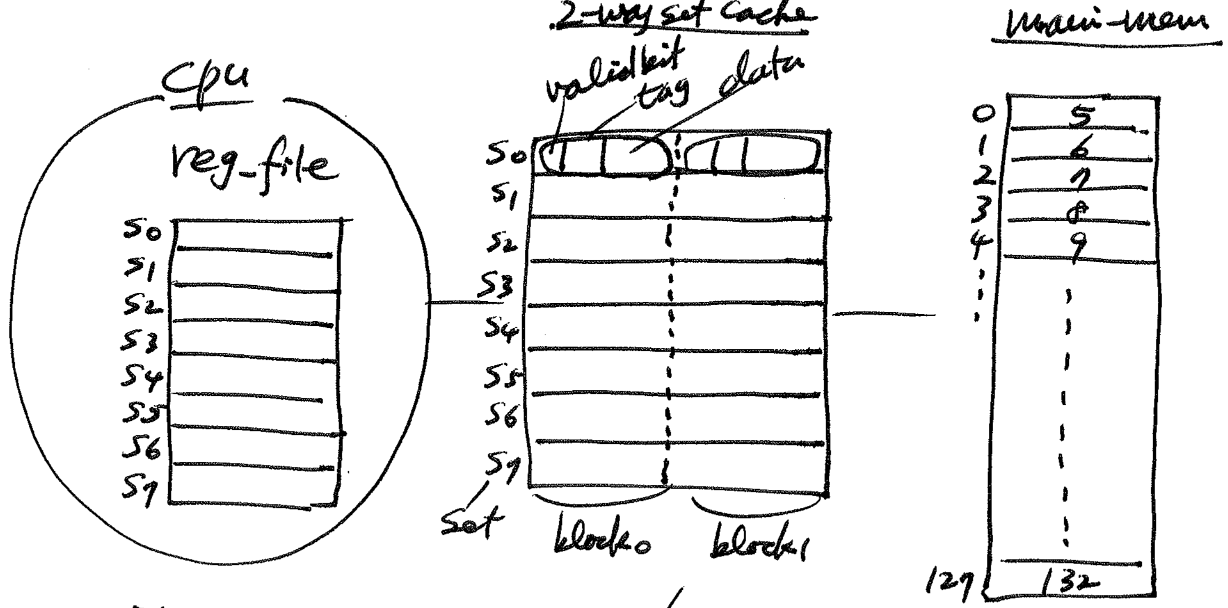
— mem. sim. program

data structures needed



int reg_file[8];

int mem[128];

```
struct block
{ int valid;
  int tag;
  int data;
}
struct block[8][2];
```

— during simulation, use int types and values; when displaying the contents, use bit_print().

ex) int x=7
    cout X; → bit_print(x);

32

0000 -- ···· ·· 00111

CS113 — Sim. prog.

6	5	5	16
op	rs	rt	offset

logic:

1. get an instr. and decode — opcode — 35 (load), 43 (store)
   - base reg. — $zero, always ∅.
   - offset value
   - rt — target of load, source of store
     - can be $s0 ~ s7
     - 16 -- 23
2. Compute effective address
   - rs + offset ⇒ byte address
   - then /4 ⇒ word address.
3. determine cache set# and Tag;
   - set#: wordaddr MOD (%) 8
   - Tag: word-addr DIV (/) 8
4. Cache hit/mis operation;

repeat

once having index (set#) and Tag from word-addr,

Search cache:

using index, find set;

then check valid bit == 1, if so,
check Tag field == Tag from word-addr.

if match → hit
else → mis

(*) → 4 cases:

load [read hit ① — cache data → reg.file[rt];
load [read mis ② → next page

store [write hit ③ → reg.file[rt] → cache data; (update cache only) WB cache
store [write mis ④ → reg.file[rt] → memory[word-addr.]; (update mem. only)

↓

↓

— Case ② (read miss)

(1) select victim block from 2 blocks;

2. if victim's valid bit == 1, update memory; → How to know mem. addr? (A) ↓ see below

(actually, "dirty" bit)

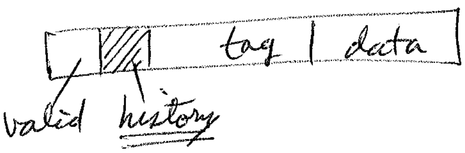
3. bring (copy) memory block to victim block;
 data data

4. Set valid bit=1 and Tag field;

5. reg.file[rt] ← cache block data;

→ How to select victim block? — LRU

⇒ use history bit (1/0)



— whenever a block is accessed (read hit, write hit, read miss),

(history ← 1
 and the other block's history ← φ.

(A) — How to determine mem. address for victim block? (when valid bit == 1)

— we know cache set# (index),
 and the victim block has Tag field;

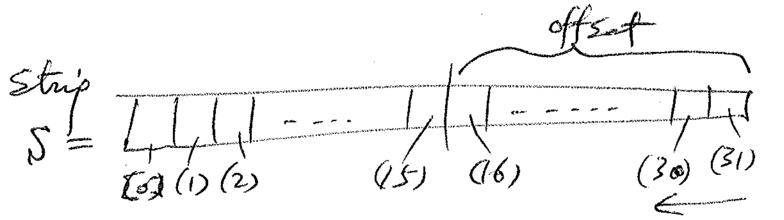
⇒ $\boxed{\text{Tag} * \underbrace{8}_{\text{tot\# of sets}} + \underbrace{\text{index}}_{\text{from addr. computation}}}$

ex) $127/8 = \underset{Q}{15}, \underset{R}{7}$

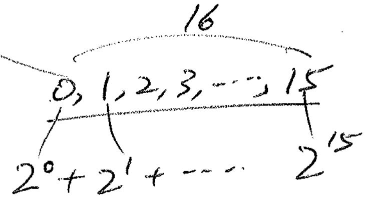
⇒ $\underset{\text{tag}}{Q} * 8 + \underset{\text{set\# (index)}}{R} = \underset{\text{mem. addr.}}{\underline{127}}$

— from instr (I-type),

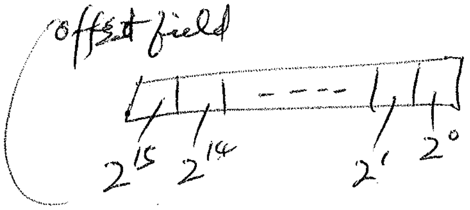
How to extract offset value (int)?



```
int offset=ϕ;
for(int i=31; i>15; i--)
  if (s.at(i) == '1')
    offset += power(2, 31-i)
```



Then offset/4 ⇒ word address //.


offset field
2^15 2^14
2^1 2^0