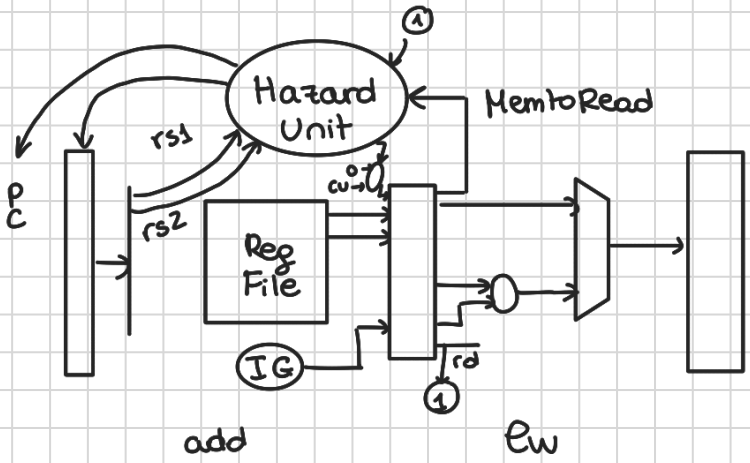
CONTROL HAZARDS

```
lw   x6, 0(x7)
add  x8, x8, x6
```



Hazard Unit → to detect the hazard

So, for this case, we need to know:

- rs1
  rs2
  MemtoRead
  rd from ew stage

LOGIC BEHIND

if ID/EX. MemRead and (ID/EX.rd = IF/ID.rs1 or ID/Ex.rd = IF/ID rs2)

stall the pipeline.

A false positive will introduce a bubble even if its not needed

- if x6 = 0
- this hazard unit can detect if an instruction is a lw but it won't understand what's the next instruction which may cause it to introduce unnecessary bubble.

  ex. lw x6, 0(x7)
  add; x8, x9, imm

BRANCH PREDICTION

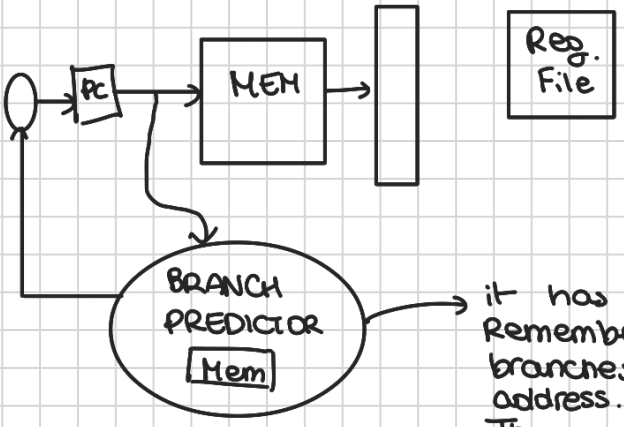
- this is done by the hardware

↓

for i = 1 to 100000

ciclo:

Usually you get the first and last prediction wrong.

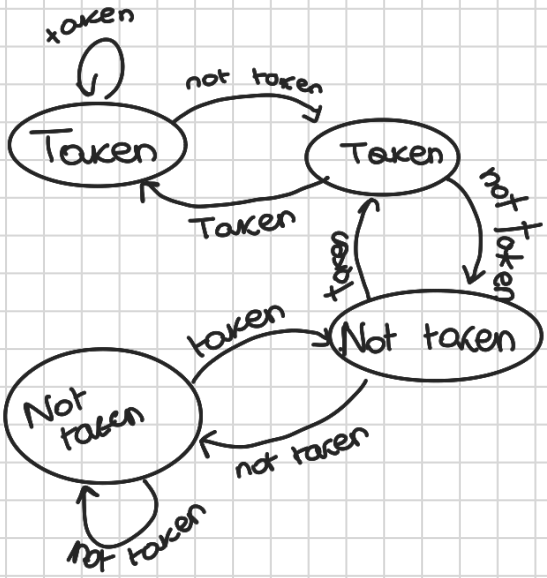


it has a small memory.
Remembers the previous branches by storing their address.
The result is also stored.

We lose one clock cycle by doing a wrong prediction

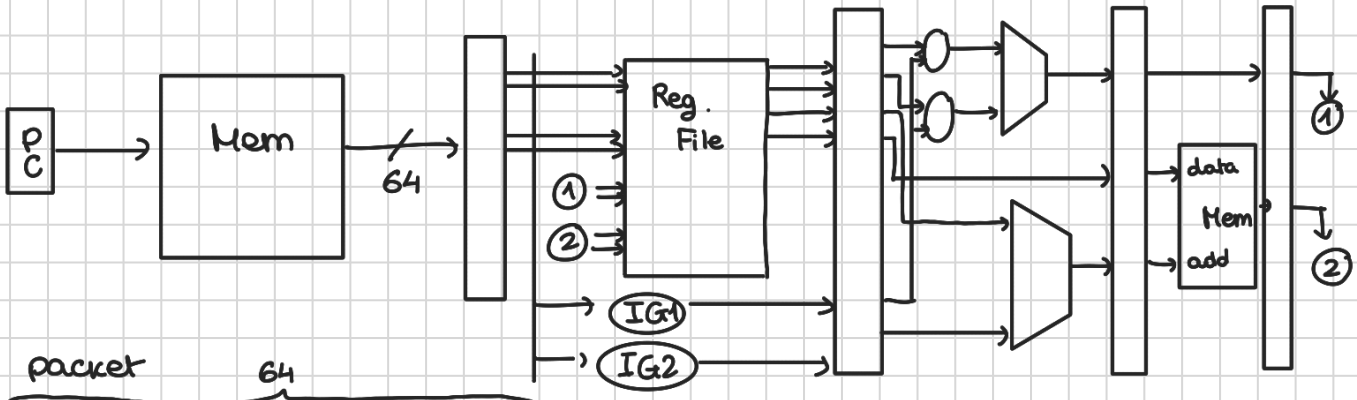
for i = 1 to 100000
  for j = 1 to 4

ciclo1
ciclo2



MULTIPLE ISSUE

- static multiple issue
  ↳ you can't put any pair of instr.



packet 64

branch/ALU	lw/sw

→ it's divided like this because the architecture will be easier

load two instr. instead of one (they are connected, packets)

- data

```
x: .word
y: .word
z: .word
n: .word 100000
```

- text

```
    lw  t0, n
    la  t1, x
    la  t2, y
    la  t3, z
ciclo: 1 lw  t4, 0(t1)
       2 lw  t5, 0(t2)
       3 →
       4 add t6, t5, t4
       5 sw  t6, 0(t3)
       6 addi t1, t1, 4
       7 addi t2, t2, 4
       8 addi t3, t3, 4
       9 addi t0, t0, -1
      10 bne t0, zero, ciclo
      11 ∅
```

RIORDINARE ③

(we dont have hazard unit ↓)

ALU/BRANCH	LW/SW
ciclo: addi t1, t1, 4	lw t4, 0(t1)
addi t2, t2, 4	lw t5, 4(t2)
addi t0, t0, -1	
add t6, t5, t4	
addi t3, t3, 4	sw t6, 0(t3)
bne t0, zero, ciclo	

PERFORMANCE

① **Single Clock Cycle** 200MHz

9 clock cycles · 10^6 = cycles

$\frac{9 \cdot 10^6}{200 \cdot 10^6} = 0{,}045\ s$ → 200 MHz

$1\ MHz = 1\ Hz \cdot 10^6$

$1\ GHz = 1\ Hz \cdot 10^9$

$0{,}045\ s \cdot 10^3 = 45\ ms$ *

$1\ ms = 10^6\ ns \rightarrow 45 \cdot 10^6\ ns$

② **Pipeline arch.** 1 GHz (no branch pred.)

∅ ↑ since there's no branch prediction it will load the next instruction

$11 \cdot 10^6\ ns = 11\ ms$

③ **Pipeline arch** 1GHz (branch predic.) (code re-ordering)

no bubble

↳ $9 \cdot 10^6\ ns = 9\ ms$

④ **Multiple issue arch** 1GHz (branch prediction)

6 loops

↓

$6 \cdot 10^6\ ns = 6\ ms$

```
add, t0, t0, -2      lw t4, 0(t1)
add, t3, t3, 8       lw t5, 0(t2)
add, t1, t1, 8       lw t8, 4(t1)
add t6, t4, t5       lw t9, 4(t2)
add, t2, t2, 8       sw t6, -8(t3)
add t10, t8, t9
bne t0, zero, ciclo  sw t10, -4(t3)
```

→ ⑤ MULTIPLE ISSUE ARCH 1 GHz
(Loop unrolling + branch prediction)

reduces penalty branches

7ns · $500 \cdot 10^3$ = 3,5 ms

* Let's say it was 4 GHz and not 200 MHz.

1 GHz = 1Hz · 10^9 so

$$\frac{9\times10^6}{4\times10^9} = 0{,}002'25 \leftarrow s$$

$$\text{Ans}\times10^3 = 2{,}25 \leftarrow ms$$