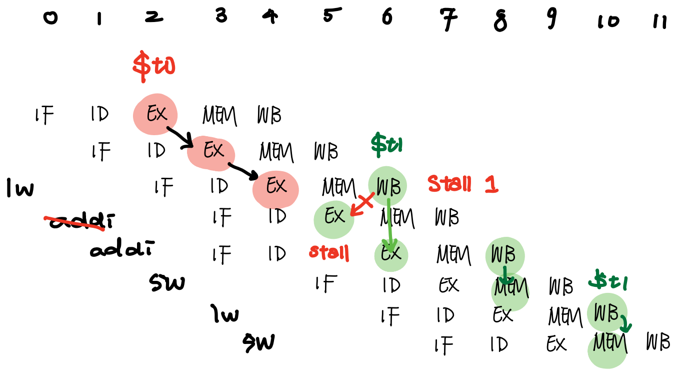
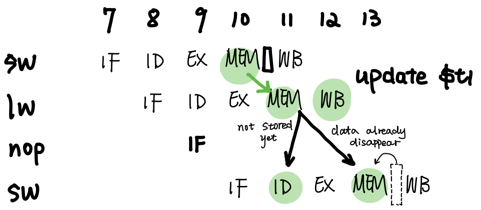
instruction fetch
instruction decode
Execute
Memory Access
Register Write Back

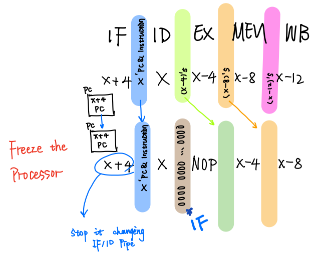
```
lui   $t0, 0xdead       # la    $t0, LABEL    (part 1)
ori   $t0, $t0, 0xbee4  # la    $t0, LABEL    (part 2)

lw    $t1, 0($t0)
addi  $t1, $t1,1
sw    $t1, 0($t0)

lw    $t1, 4($t0)
sw    $t1, 8($t0)

lw    $t1, 12($t0)
nop
sw    $t1, 16($t0)
```





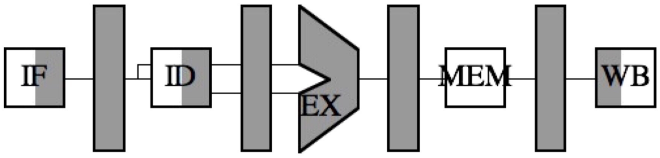
LOAD + ALU ≈ hazard

* loaded reg used as alu parameter

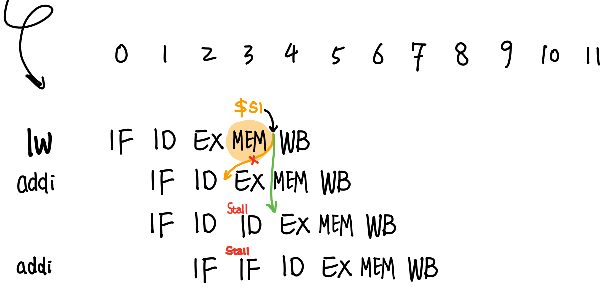


lw

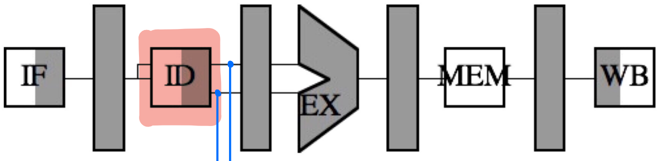
```
lw    $s0, 0($t0)
addi  $s1, $s0,1
addi  $s2, $s0,2
sw    $s0, 4($t0)

lw    $s3, 8($t0)
addi  $s4, $s3,100

lw    $s5, 12($t0)
nop
addi  $s5, 16($t0)
```

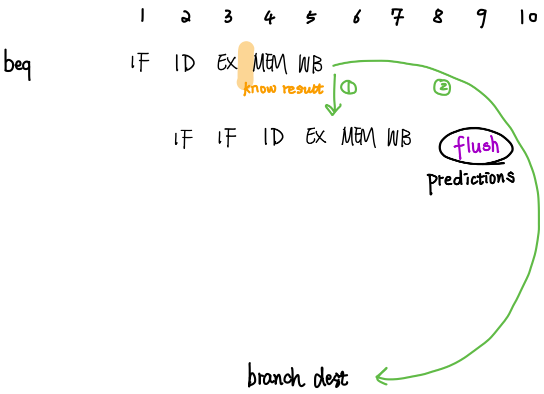


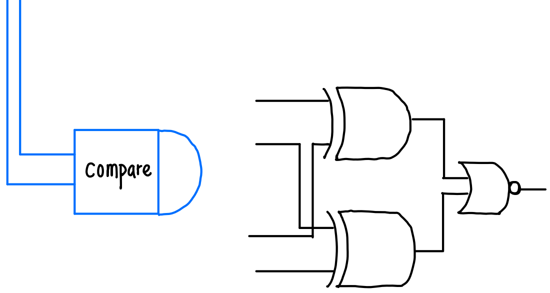
```
is_lw_stall_required():
    if (id_ex.memRead &&
        (id_ex.rt == if.rs || id_ex.rt == if.rt))
```



branch







Source	Machine	
add	beq	add (if needed)
beq	add	beq
	branch delay	nop