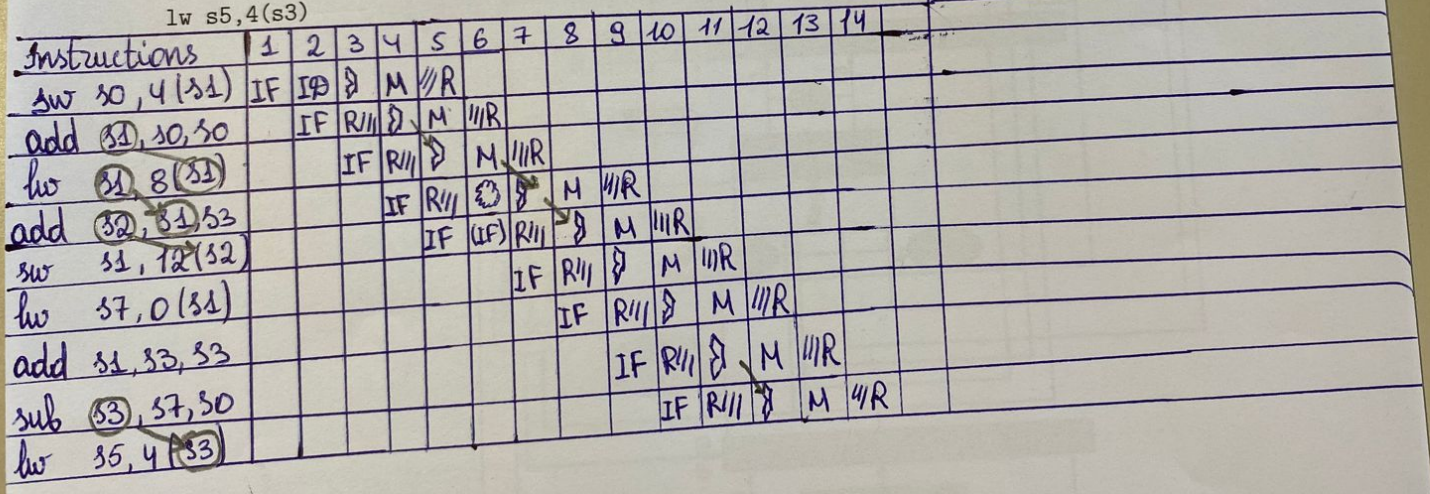
- Execution completes when the last instruction exits the pipeline.
- If you read and write the same register of the register file in the same clock cycle, the value that is read is the value that is being witten.

Show a figure of the pipeline with bubbles and forwardings.

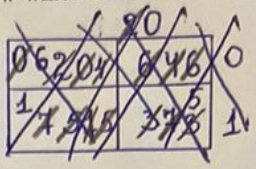
```
sw s0, 4(s1)
add s1, s0, s0
lw s1, 8(s1)
add s2, s1, s3
sw s1, 12(s2)
lw s7, 0(s1)
add s1, s3, s3
sub s3, s7, s0
lw s5,4(s3)
```

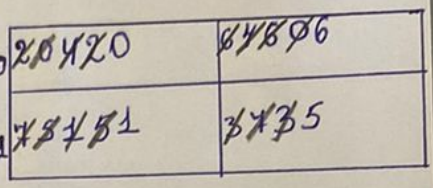
Clock cycles number: 14



Exercise 2. Consider a two-way associative cache with two sets and block of 4 words. The cache is initially empty and replacement of blocks is done by using LRU. Show which of the following memory accesses are hit and which are misses: block size: 16 bytes

	Hit/Miss	Block number	Set number
36,	M	2	0
32,	H	2	0
40,	H	2	0
124,	M	7	1
56,	M	3	1
104,	M	6	0
8,	M	0	0
88,	M	5	1
48,	H	3	1
76,	M	4	0
0,	H	0	0
100,	M	6	0
16,	M	1	1
64,	M	4	0
20,	H	1	1
52,	H	3	1
4,	M	0	0
80,	M	5	1
120,	M	7	1
44,	M	2	0
108,	M	6	0
92,	H	5	1
12,	M	0	0
60,	M	3	1
28,	M	1	1
84.	M	5	1





```
        lui s0, 0x10014
        lw s1, 0(s0)
        lw s2, 4(s0)
        lui s0, 0x10010
loop:   lw t0, 0(s0)
        addi t0, t0, s2
        sw t0, 0(s0)
        addi s0, s0, 4
        addi s1, s1, -1
        bne s1, zero, loop
        li a7, 10
        ecall
```

This program adds a constant to each entry of an array.

1. **Question A:** Assume the single clock architecture. What is the approximate total miss rate if you have a one-way associative cache with 8 blocks of 64 bytes? Do you get a lower miss rate if the cache is two-way associative with 4 blocks?
2. **Question B:** What is the speed-up (how faster is it) if we use the Risc-V multiple issue architecture with loop unrolling of 4 loops instead of the standard pipelined architecure? Show the code.

Question B: Number of clock cycles in standard pipelined architecure:

$6 \cdot 4096 = 24576$

Multiple issue architecture with loop unrolling:

$8 \cdot \frac{4096}{4} = 8192$

branch / ALU	lw / sw
addi t0, t0, s2	lw t0, 0(s0)
addi t1, t1, s2	lw t1, 4(s0)
addi t2, t2, s2	lw t2, 8(s0)
addi t3, t3, s2	lw t3, 12(s0)
addi s0, s0, 16	sw t0, 16(s0)
addi s1, s1, -4	sw t1, -12(s0)
bne s1, zero, loop	sw t3, -8(s0)
	sw t4, -4(s0)

$\text{speed-up} = \frac{6 \cdot 4096}{2 \cdot 4096} = 3\text{X}$

3 times faster.