# Q1)

a.) Write the MIPS code for both of the main and func procedures below. If you use any saved registers save them to stack.

ANSWER:

```
main:

        addi $a1, $zero, 1          # a1 = 1
        addi $a2, $zero, 3          # a2 = 3 (pow = 3)
        addi $a3, $zero, 2          # a3 = 2
        jal func                    # brach to func and saved $ra
        add $t0, $zero, $v0         # saved return value of func to b
        j exit                      # exit

func:
        addi $sp, $sp, -16          # adjust stack for 2 items
        sw $ra, 12($sp)             # saved return address
        sw $a1, 8($sp)              # saved a1
        sw $a2, 4($sp)              # saved a2
        sw $a3, 0($sp)              # saved a3

        slti $t0, $a2, 2            # if a2 < 2 , t0 = 1
        beq $t0, $zero, recurse     # if t0 = 0 , go recurse

        addi $a3, $a1, 1            # b = a + 1
        add $v0, $a3, $zero         # v0 = b

        lw $ra, 12($sp)             # loud original ra
        addi $sp, $sp, 16           # restore stack
        jr $ra                      # back

recurse:
        addi $a2, $a2, -1           # pow = pow - 1
        jal func                    # return func
        lw $a3, 0($sp)              # load original a3
        lw $a2, 4($sp)              # load original a2
        lw $a1, 8($sp)              # load original a1
        lw $ra, 12($sp)             # load original ra

        add $t1, $a1, $a3           # make (a+b) and saved on t1
        mul $v0, $v0, $t1           # (a + b) * func(a, pow - 1, b)

        addi $sp, $sp, 16           # restore stack
        jr $ra                      # back
```

b.) Write down the values of stack that are changed during this call (memory below 0x1030 5010)?

ANSWER:

when first calling, stack is :

0x10304994 - a3

0x10304998 - a2

0x10305002 - a1

0x10305006 - ra

when second calling, stack is :

0x10304994 - a3

0x10304998 - a2

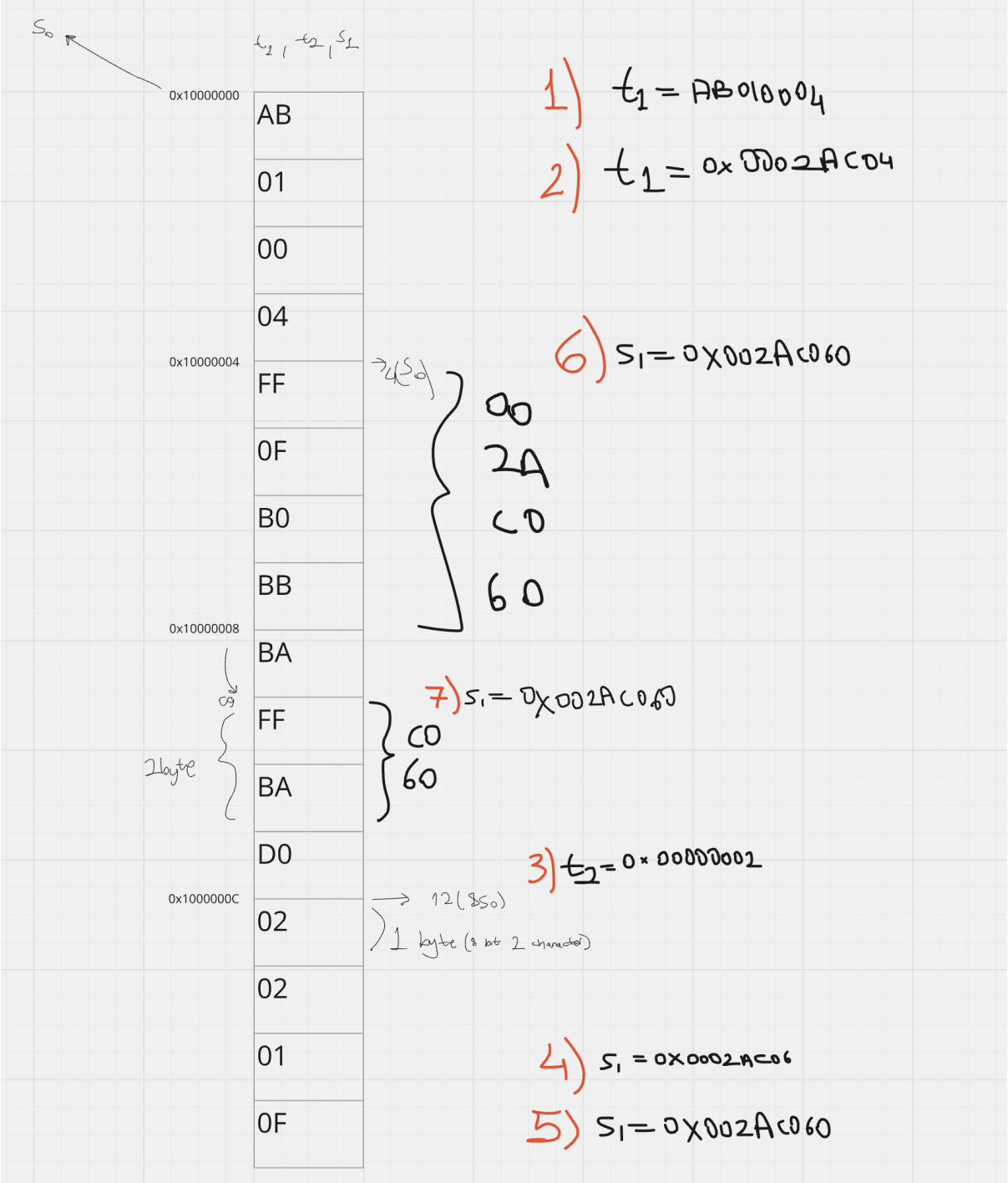
0x10305002 - a1

0x10305006 - ra

function is finished. Stack is restored ( 0x1030 5010 )

# Q2)

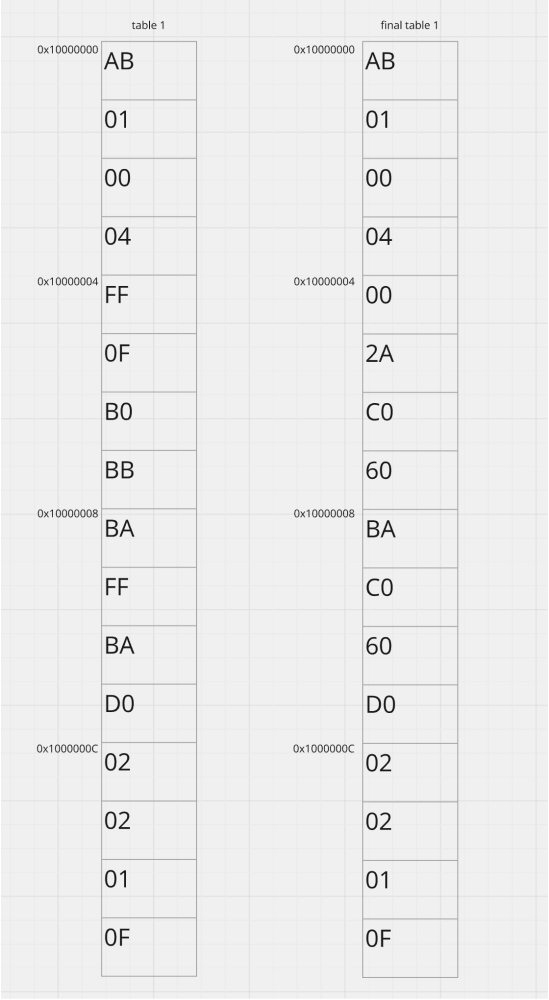
>> Firstly we want to explain how we create structure. We use mira for drawing memory and process.



a) Comment each line of this code to explain what it performs?

1. lw $t1, 0($s0) # loud word(4byte) from s0 to t1 [ t1 = 0xAB010004 ]
2. srl $t1, $t1, 14 # shift 14 bits t1 right location [ t1 = 0x0002AC04 ]
3. lbu $t2, 12($s0) # loud byte(8bits) from 12(s0) [ t2 = 0x00000002 ]
4. addu $s1, $t1, $t2 # s1 = t1 + t2 [ s1 = 0x0002AC06 ]
5. sll $s1, $s1, 4 # shift 4 bits s1 left location [ s1 = 0x002AC060 ]
6. sw $s1, 4($s0) # store s1(word) to 4(s0) [ we showed details in the picture ]
7. sh $s1, 9($s0) # store s1(halfword) to 9(s0) [ we showed details in the picture ]

b) Write the final state of the changed cells in Table 1 after above code is run.




c) Write the final values registers $s1, $ t1, $t2

**$s1**: 0x002AC060
**$t1**: 0x0002AC04
**$t2**: 0x00000002

Selim Ayaydın 21COMP1034 - - - - - - - İhsan Eren Erben 22SOFT1055