Q1) B. ASSUME YOU HAVE \$t0=[0xCAFEBABE], HOW CAN YOU CHANGE IT TO

[0xBABACAF0] and [0x03CA1E00] Using only logic and shift operations.

Answer:

1-) For \$t0=[0xCAFEBABE] to [0xBABACAF0]: srl \$t1, \$t0, 16 # \$t1 = 0x0000CAFE sll \$t2, \$t0, 16 # \$t2 = 0xBABE0000 ori \$t1, \$t1, 0x000ACAF0 # \$t1 = 0x000ACAFE andi \$t2, \$t2, 0xBAB00000 # \$t2 = 0xBAB00000 andi \$t1, \$t1, 0x000ACAF0 # \$t1 = 0x000ACAF0 or \$t0, \$t1,\$t2 # \$t0 = 0xBABACAF0

2-) For \$t0=[0xCAFEBABE] to [0x03CA1E00]: srl \$t0, \$t0, 8 # \$t0 = 0x00CAFEBA ori \$t0, \$t0, 0x03000000 # \$t0 = 0x03CAFEBA andi \$t0, \$t0, 0x03CA1E00 # \$t0 = 0x03CA1E00

Q2) WHAT MIPS INSTRUCTION IS REPRESENTED BY THE FOLLOWING HEX 0x20090022

Answer:

We use I-format because of first 6 bits (addi is used with I-format) We know that for MIPS I-format Instruction:

op: 6 bits, rs: 5 bits, rt: 5 bits, constant: 16 bits

Firstly, we converted hexa to decimal and we found these values:

op: 001000 (addi operation)

rs: 0000 (\$zero)

rt: 01001 (\$t1)

constant: 0000 0000 0010 0010(34)

The result is:

Addi \$t1, \$zero, 34

Q3) Convert the following java code into MIPS. Write comment for every line of your MIPS instructions.

```
int i = 32;
int j = -3;
while (i>2)
i--;
j++;
```

Answer:

```
lw $t0, i
                                           # Load i (32) to register $t0
        lw $t1, j
                                           # Load j (-3) to register $t1
whileLoop:
                                          # start while loop
        addi $t2, $zero, 2
                                          # load immediate value 2 into $t2
                                          # set $t3 to 1 if i < 2, else 0
        slt $t3, $t0, $t2
        bne $t3, $zero, loopFinish
                                          # If $t3 is not equal to zero, jump to loopFinish (i <= 2)
        # Loop body
        addi $t0, $t0, -1
                                         #i = i - 1
        addi $t1, $t1, 1
                                         # j = j + 1
                                         # (j) Jump back to the start of the loop
        j whileLoop
                                         # loopFinish
loopFinish:
                                        # store updated "i" to memory
        sw $t0, i
        sw $t1, j
                                        # store updated "j" to memory
```

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