Name: Nguyễn Thanh Long

ID: ITITRG18012

**Lab 02**

**2.1)**



**2.2)**



The immediate value (-1) is shifted left 16 bits and stored in the register $1. The lower 16 bits are zeroes then bitwise “or” a register and an immediate value (-5) and stores the result in a register

**2.3)**



The immediate value (-21829) is shifted left 16 bits and stored in the register $1. The lower 16 bits are zeroes then bitwise “or” a register and an immediate value (-13091) and stores the result in a register

**3.1)**

****

**3.2)**

****

.data # the data segment

prompt: .asciiz "Guess a number (1 - 1000): "

win: .asciiz "You win!!\n"

lose: .asciiz "You lose!!\n"

newline: .asciiz "\n"

.text # the code segment

.globl main

main:

li $t0, 0x1fa

# print out the prompt

la $a0, prompt

li $v0, 4

syscall

# read in an integer

li $v0, 5

syscall

move $t1, $v0

bne $t0, $t1, LOSE

beq $t0, $t1, WIN

WIN:

# print out "win"

la $a0, win

li $v0, 4

syscall

jr $ra # return to caller (\_\_start)

LOSE:

# print out "lose"

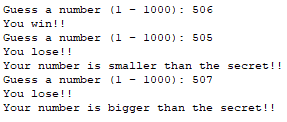
la $a0, lose

li $v0, 4

syscall

jr $ra # return to caller (\_\_start)

**3.3)**

****

.data # the data segment

prompt: .asciiz "Guess a number (1 - 1000): "

win: .asciiz "You win!!\n"

lose: .asciiz "You lose!!\n"

msgbigger: .asciiz "Your number is bigger than the secret!!\n"

msgsmaller: .asciiz "Your number is smaller than the secret!!\n"

newline: .asciiz "\n"

.text # the code segment

.globl main

main:

li $t0, 0x1fa

# print out the prompt

la $a0, prompt

li $v0, 4

syscall

# read in an integer

li $v0, 5

syscall

move $t1, $v0

bne $t0, $t1, LOSE

beq $t0, $t1, WIN

WIN:

# print out "win"

la $a0, win

li $v0, 4

syscall

jr $ra # return to caller (\_\_start)

LOSE:

# print out "lose"

la $a0, lose

li $v0, 4

syscall

bgt $t1, $t0, BIGGER # if $t1 > $t0 then target

blt $t1, $t0, SMALLER # if $t1 < $t01 then target

jr $ra # return to caller (\_\_start)

BIGGER:

# print out "BIGGER"

la $a0, msgbigger

li $v0, 4

syscall

jr $ra

SMALLER:

# print out "SMALLER"

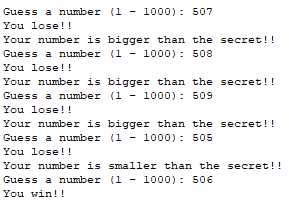
la $a0, msgsmaller

li $v0, 4

syscall

jr $ra

**3.4)**

****

.data # the data segment

prompt: .asciiz "Guess a number (1 - 1000): "

win: .asciiz "You win!!\n"

lose: .asciiz "You lose!!\n"

msgbigger: .asciiz "Your number is bigger than the secret!!\n"

msgsmaller: .asciiz "Your number is smaller than the secret!!\n"

newline: .asciiz "\n"

.text # the code segment

.globl main

main:

li $t0, 0x1fa

j REQ\_INPUT

REQ\_INPUT:

# print out the prompt

la $a0, prompt

li $v0, 4

syscall

# read in an integer

li $v0, 5

syscall

move $t1, $v0

bne $t0, $t1, LOSE

beq $t0, $t1, WIN

WIN:

# print out "win"

la $a0, win

li $v0, 4

syscall

jr $ra # return to caller (\_\_start)

LOSE:

# print out "lose"

la $a0, lose

li $v0, 4

syscall

bgt $t1, $t0, BIGGER # if $t1 > $t0 then target

blt $t1, $t0, SMALLER # if $t1 < $t0 then target

jr $ra # return to caller (\_\_start)

BIGGER:

# print out "BIGGER"

la $a0, msgbigger

li $v0, 4

syscall

j REQ\_INPUT

SMALLER:

# print out "SMALLER"

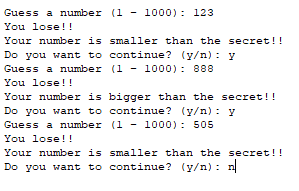
la $a0, msgsmaller

li $v0, 4

syscall

j REQ\_INPUT

**3.5)**

****

.data # the data segment

prompt: .asciiz "\nGuess a number (1 - 1000): "

win: .asciiz "You win!!\n"

lose: .asciiz "You lose!!\n"

msgbigger: .asciiz "Your number is bigger than the secret!!\n"

msgsmaller: .asciiz "Your number is smaller than the secret!!\n"

msgCont: .asciiz "Do you want to continue? (y/n): "

newline: .asciiz "\n"

.text # the code segment

.globl main

main:

li $t0, 0x1fa

j REQ\_INPUT

REQ\_INPUT:

# print out the prompt

la $a0, prompt

li $v0, 4

syscall

# read in an integer

li $v0, 5

syscall

move $t1, $v0

# compare 2 register and branch

bne $t0, $t1, LOSE

beq $t0, $t1, WIN

WIN:

# print out "win"

la $a0, win

li $v0, 4

syscall

jr $ra # return to caller (\_\_start)

LOSE:

# print out "lose"

la $a0, lose

li $v0, 4

syscall

bgt $t1, $t0, BIGGER # if $t1 > $t0 then target

blt $t1, $t0, SMALLER # if $t1 < $t0 then target

jr $ra # return to caller (\_\_start)

BIGGER:

# print out "BIGGER"

la $a0, msgbigger

li $v0, 4

syscall

j FLAG

SMALLER:

# print out "SMALLER"

la $a0, msgsmaller

li $v0, 4

syscall

j FLAG

FLAG:

# print out "msgFlag"

la $a0, msgCont

li $v0, 4

syscall

# read in an character

li $v0, 12

syscall

move $t1, $v0

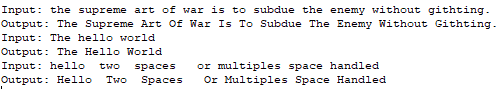
bne $t1, 'y', ENDGAME

j REQ\_INPUT

ENDGAME:

jr $ra

**4)**

****

.data # the data segment

prompt: .asciiz "Input: "

output: .asciiz "Output: "

buffer: .space 100

.text # the code segment

.globl main

main:

# print out the prompt

la $a0, prompt

li $v0, 4

syscall

# read a string

li $v0,8

la $a0, buffer

li $a1, 100

syscall

# &t0 = buffer

la $t0, buffer

# &t1 = 0

li $t1, 0

lb $t1, 0($t0)

# check if t0[0] is lower case

ble $t1, 97, loop

# t1 = t1 - 32 --> upper the character

addi $t1, $t1, -32

# t0[0] = t1

sb $t1, 0($t0)

# &t0 = t0 + 1

addi $t0, $t0, 1

loop:

# t1 = t0[0]

lb $t1, 0($t0)

# if (t1 == 0) done

beqz $t1, done

# if (t1 == ' ') toUpper

beq $t1, ' ', toUpper

# &t0 = t0 + 1

addi $t0, $t0, 1

j loop

toUpper:

# &t0 = t0 + 1

addi $t0, $t0, 1

# \*t1 = &t0[0]

lb $t1, 0($t0)

# if (t1 == ' ') toUpper

beq $t1, ' ', toUpper

# t1 = t1 - 32 --> upper the character

addi $t1, $t1, -32

# t0[0] = t1

sb $t1, 0($t0)

# &t0 = t0 + 1

addi $t0, $t0, 1

j loop

done:

# print out the result

la $a0, output

li $v0, 4

syscall

la $a0, buffer

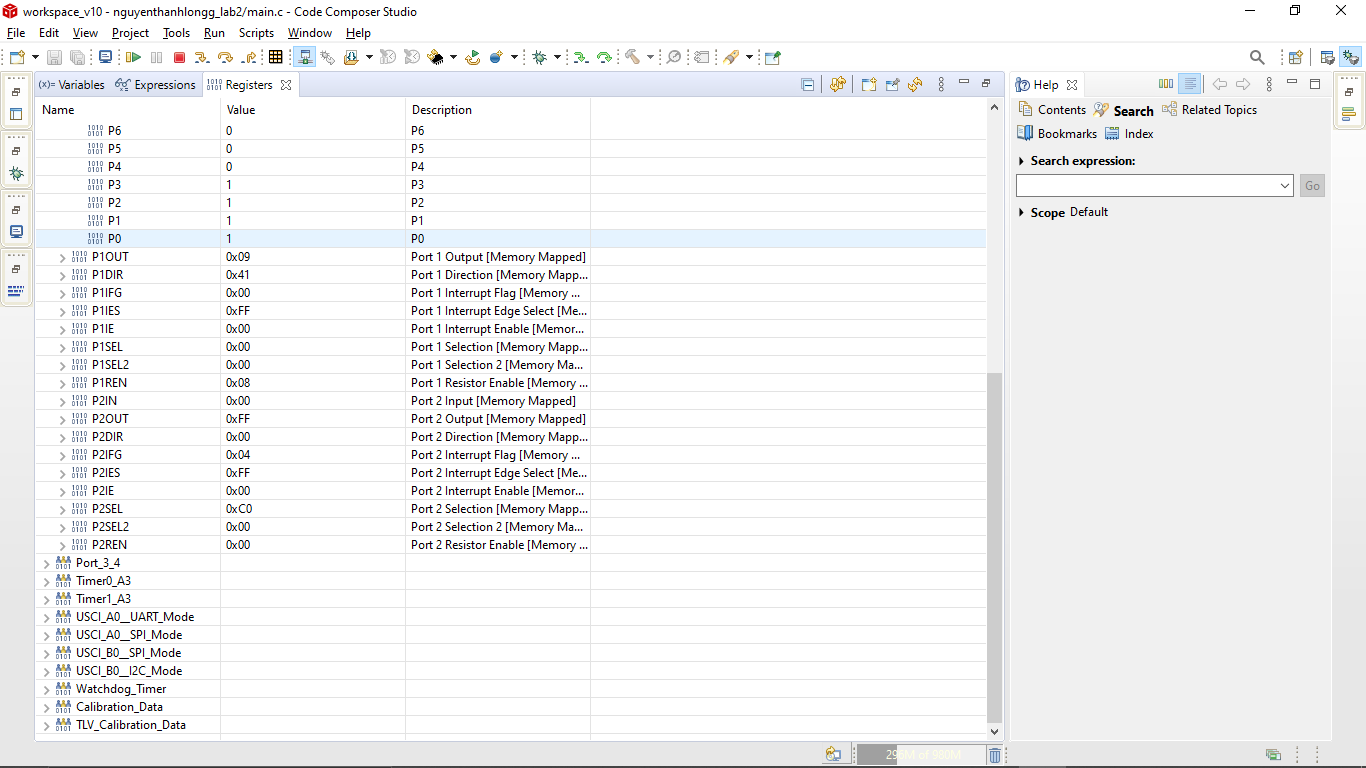
li $v0, 4

syscall

jr $ra # return to caller (\_\_start)

II/

Step2:



P1OUT: 0x09

P1IN: 0x06

P1DIR:0x00

P1REN:0x00

P1IFG:0x30

Step 3 :

