**­­­­Computer Architecture Lab Report Week 10**

**Full name: Nguyễn Hồng Phúc**

**Student ID: 20225659**

Assignment 1

**Code:**

**Vẽ hình tam giác đều:**

.eqv HEADING 0xffff8010

.eqv MOVING 0xffff8050

.eqv LEAVETRACK 0xffff8020

.eqv WHEREX 0xffff8030

.eqv WHEREY 0xffff8040

.text

main:

addi $a0, $zero, 90

jal ROTATE

jal GO

addi $v0, $zero, 32 # Keep running by sleeping in 1000 ms

li $a0, 15000

syscall

addi $a0, $zero, 180

jal ROTATE

jal GO

addi $v0, $zero, 32 # Keep running by sleeping in 1000 ms

li $a0, 7000

syscall

# Mio sleep la 1 doan, ve hay khong tuy nguoi lap trinh

# sleep1:

addi $a0, $zero, 150

jal ROTATE

jal GO

jal UNTRACK # keep old track

jal TRACK # and draw new track line

addi $v0, $zero, 32 # Keep running by sleeping in 1000 ms

li $a0, 7000

syscall

# sleep2:

addi $a0, $zero, 270

jal ROTATE

jal GO

jal UNTRACK # keep old track

jal TRACK # and draw new track line

addi $v0, $zero, 32 # Keep running by sleeping in 1000 ms

li $a0, 7000

syscall

# sleep3:

addi $a0, $zero, 30

jal ROTATE

jal GO

jal UNTRACK # keep old track

jal TRACK # and draw new track line

addi $v0, $zero, 32 # Keep running by sleeping in 1000 ms

li $a0, 7000

syscall

end\_main:

jal UNTRACK # keep old track

addi $a0, $zero, 90

jal ROTATE

jal GO

addi $v0, $zero, 32 # Keep running by sleeping in 1000 ms

li $a0, 3000

syscall

jal STOP

li $v0, 10

syscall

GO:

li $t0, MOVING # change MOVING port

addi $t1, $zero, 1 # to logic 1,

sb $t1, 0($t0) # to start running

jr $ra

ROTATE:

li $t0, HEADING # change HEADING port

sw $a0, 0($t0) # to rotate robot

jr $ra

STOP:

li $t0, MOVING # change MOVING port to 0

addi $t1, $zero, 0 # to stop

sb $t1, 0($t0)

jr $ra

TRACK:

li $t0, LEAVETRACK # change LEAVETRACK port

addi $t1, $zero, 1 # to logic 1,

sb $t1, 0($t0) # to start tracking

jr $ra

UNTRACK:

li $t0, LEAVETRACK # change LEAVETRACK port to 0

addi $t1, $zero, 0 # to stop drawing tail

sb $t1, 0($t0)

jr $ra

**Kết quả:**

A screenshot of a computer

Description automatically generated

**Vẽ hình vuông Code:**

.eqv HEADING 0xffff8010

.eqv MOVING 0xffff8050

.eqv LEAVETRACK 0xffff8020

.eqv WHEREX 0xffff8030

.eqv WHEREY 0xffff8040

.text

main:

addi $a0, $zero, 90

jal ROTATE

jal GO

addi $v0,$zero,32 # Keep running by sleeping in 1000 ms

li $a0,15000

syscall

addi $a0, $zero, 180

jal ROTATE

jal GO

addi $v0,$zero,32 # Keep running by sleeping in 1000 ms

li $a0,7000

syscall

sleep1:

addi $a0, $zero, 90

jal ROTATE

jal GO

jal UNTRACK # keep old track

jal TRACK # and draw new track line

addi $v0,$zero,32 # Keep running by sleeping in 1000 ms

li $a0,7000

syscall

sleep2:

addi $a0, $zero, 180

jal ROTATE

jal GO

jal UNTRACK # keep old track

jal TRACK # and draw new track line

addi $v0,$zero,32 # Keep running by sleeping in 1000 ms

li $a0,7000

syscall

sleep3:

addi $a0, $zero, 270

jal ROTATE

jal GO

jal UNTRACK # keep old track

jal TRACK # and draw new track line

addi $v0,$zero,32 # Keep running by sleeping in 1000 ms

li $a0,7000

syscall

sleep4:

addi $a0, $zero, 0

jal ROTATE

jal GO

jal UNTRACK # keep old track

jal TRACK # and draw new track line

addi $v0,$zero,32 # Keep running by sleeping in 1000 ms

li $a0,7000

syscall

end\_main:

jal UNTRACK # keep old track

addi $a0, $zero, 90

jal ROTATE

jal GO

addi $v0,$zero,32 # Keep running by sleeping in 1000 ms

li $a0,3000

syscall

jal STOP

li $v0, 10

syscall

GO:

li $at, MOVING # change MOVING port

addi $k0, $zero,1 # to logic 1,

sb $k0, 0($at) # to start running

jr $ra

ROTATE:

li $at, HEADING # change HEADING port

sw $a0, 0($at) # to rotate robot

jr $ra

STOP:

li $at, MOVING # change MOVING port to 0

sb $zero, 0($at) # to stop

jr $ra

TRACK:

li $at, LEAVETRACK # change LEAVETRACK port

addi $k0, $zero,1 # to logic 1,

sb $k0, 0($at) # to start tracking

jr $ra

UNTRACK:

li $at, LEAVETRACK # change LEAVETRACK port to 0

sb $zero, 0($at) # to stop drawing tail

jr $ra

**Kết quả:**

A screenshot of a computer

Description automatically generated

**Vẽ hình ngôi sao Code:**

.eqv HEADING 0xffff8010

.eqv MOVING 0xffff8050

.eqv LEAVETRACK 0xffff8020

.eqv WHEREX 0xffff8030

.eqv WHEREY 0xffff8040

.text

main:

addi $a0, $zero, 90

jal ROTATE

jal GO

addi $v0, $zero, 32 # Keep running by sleeping for 1000 ms

li $a0, 15000

syscall

addi $a0, $zero, 180

jal ROTATE

jal GO

addi $v0, $zero, 32 # Keep running by sleeping for 1000 ms

li $a0, 7000

syscall

sleep1:

addi $a0, $zero, 162

jal ROTATE

jal GO

jal UNTRACK # Keep old track

jal TRACK # Draw new track line

addi $v0, $zero, 32 # Keep running by sleeping for 1000 ms

li $a0, 7000

syscall

sleep2:

addi $a0, $zero, 306

jal ROTATE

jal GO

jal UNTRACK # Keep old track

jal TRACK # Draw new track line

addi $v0, $zero, 32 # Keep running by sleeping for 1000 ms

li $a0, 7000

syscall

sleep3:

addi $a0, $zero, 90

jal ROTATE

jal GO

jal UNTRACK # Keep old track

jal TRACK # Draw new track line

addi $v0, $zero, 32 # Keep running by sleeping for 1000 ms

li $a0, 7000

syscall

sleep4:

addi $a0, $zero, 234

jal ROTATE

jal GO

jal UNTRACK # Keep old track

jal TRACK # Draw new track line

addi $v0, $zero, 32 # Keep running by sleeping for 1000 ms

li $a0, 7000

syscall

sleep5:

addi $a0, $zero, 18

jal ROTATE

jal GO

jal UNTRACK # Keep old track

jal TRACK # Draw new track line

addi $v0, $zero, 32 # Keep running by sleeping for 1000 ms

li $a0, 7000

syscall

end\_main:

jal UNTRACK # Keep old track

addi $a0, $zero, 90

jal ROTATE

jal GO

addi $v0, $zero, 32 # Keep running by sleeping for 1000 ms

li $a0, 3000

syscall

jal STOP

li $v0, 10

syscall

GO:

li $at, MOVING # Change MOVING port

addi $k0, $zero, 1 # to logic 1

sb $k0, 0($at) # to start running

jr $ra

ROTATE:

li $at, HEADING # Change HEADING port

sw $a0, 0($at) # to rotate the robot

jr $ra

STOP:

li $at, MOVING # Change MOVING port to 0

sb $zero, 0($at) # to stop

jr $ra

TRACK:

li $at, LEAVETRACK # Change LEAVETRACK port

addi $k0, $zero, 1 # to logic 1

sb $k0, 0($at) # to start tracking

jr $ra

UNTRACK:

li $at, LEAVETRACK # Change LEAVETRACK port to 0

sb $zero, 0($at) # to stop drawing tail

jr $ra

**Kết quả:**

A screenshot of a computer

Description automatically generated

Assignment 2

**Code:**

.eqv KEY\_CODE 0xFFFF0004 # ASCII code from keyboard, 1 byte

.eqv KEY\_READY 0xFFFF0000 # =1 if has a new keycode ?

# Auto clear after lw

.eqv DISPLAY\_CODE 0xFFFF000C # ASCII code to show, 1 byte

.eqv DISPLAY\_READY 0xFFFF0008 # =1 if the display has already to do

.text

li $k0, KEY\_CODE

li $k1, KEY\_READY

li $s0, DISPLAY\_CODE # chua ky tu can in ra man hinh

li $s1, DISPLAY\_READY

loop:

nop

WaitForKey:

lw $t1, 0($k1) # $t1 = [$k1] = KEY\_READY

beq $t1, $zero, WaitForKey # if $t1 == 0 then Polling

ReadKey:

lw $t0, 0($k0) # $t0 = [$k0] = KEY\_CODE

WaitForDis:

lw $t2, 0($s1) # $t2 = [$s1] = DISPLAY\_READY

beq $t2, $zero, WaitForDis # if $t2 == 0 then Polling

Kiemtra:

beq $t3, 1, KiemTraX

beq $t0, 101, Co

KiemTraX:

beq $t3, 2, KiemTraI

beq $t0, 120, Co

KiemTraI:

beq $t3, 3, KiemTraT

beq $t0, 105, Co

KiemTraT:

beq $t3, 4, Encrypt2

beq $t0, 116, Co

Encrypt:

addi $t3, $zero, 0

Encrypt2:

ChuHoa:

bgt $t0, 90, ChuThuong

blt $t0, 65, ChuThuong

addi $t0, $t0, 32

j ShowKey

ChuThuong:

bgt $t0, 122, ChuSo

blt $t0, 97, ChuSo

addi $t0, $t0, -32

j ShowKey

ChuSo:

bgt $t0, 57, Khac

blt $t0, 48, Khac

addi $t0, $t0, 0

j ShowKey

Khac:

addi $t0, $zero, 42

ShowKey:

sw $t0, 0($s0) # show key

nop

beq $t3, 4, Exit

j loop

Co:

addi $t3, $t3, 1

j Encrypt2

Exit:

li $v0, 10

syscall

**Kết quả:**

A screenshot of a computer

Description automatically generated

Assignment 3

**Code:**

.eqv HEADING 0xffff8010

.eqv MOVING 0xffff8050

.eqv LEAVETRACK 0xffff8020

.eqv WHEREX 0xffff8030

.eqv WHEREY 0xffff8040

.eqv KEY\_CODE 0xFFFF0004 # ASCII code from keyboard, 1 byte

.eqv KEY\_READY 0xFFFF0000 # =1 if has a new keycode ?

# Auto clear after lw

.eqv DISPLAY\_CODE 0xFFFF000C # ASCII code to show, 1 byte

.eqv DISPLAY\_READY 0xFFFF0008 # =1 if the display has already to do

.text

main:

li $t8, KEY\_CODE

li $t9, KEY\_READY

li $s0, DISPLAY\_CODE # chua ky tu can in ra man hinh

li $s1, DISPLAY\_READY

loop:

nop

WaitForKey:

lw $t1, 0($t9) # $t1 = [$k1] = KEY\_READY

beq $t1, $zero, WaitForKey # if $t1 == 0 then Polling

ReadKey:

lw $t0, 0($t8) # $t0 = [$k0] = KEY\_CODE

WaitForDis:

lw $t2, 0($s1) # $t2 = [$s1] = DISPLAY\_READY

beq $t2, $zero, WaitForDis # if $t2 == 0 then Polling

Kiemtra:

KiemTraE:

beq $t3, 1, KiemTraX

beq $t0, 101, Co

KiemTraX:

beq $t3, 2, KiemTraI

beq $t0, 120, Co

KiemTraI:

beq $t3, 3, KiemTraT

beq $t0, 105, Co

KiemTraT:

beq $t3, 4, Encrypt2

beq $t0, 116, Co

Encrypt:

addi $t3, $zero, 0

Encrypt2:

beq $t0, 65, sleepA

beq $t0, 97, sleepA

beq $t0, 87, sleepW

beq $t0, 119, sleepW

beq $t0, 68, sleepD

beq $t0, 100, sleepD

beq $t0, 83, sleepS

beq $t0, 115, sleepS

beq $t0, 32, Nghiem

beq $t0, 67, Ditiep

beq $t0, 99, Ditiep

ShowKey:

sw $t0, 0($s0) # show key

nop

j loop

Co:

addi $t3, $t3, 1

j Encrypt2

sleepW:

addi $a0, $zero, 0

jal ROTATE

jal GO

jal UNTRACK # keep old track

jal TRACK # and draw new track line

j ShowKey

sleepS:

addi $a0, $zero, 180

jal ROTATE

jal GO

jal UNTRACK # keep old track

jal TRACK # and draw new track line

j ShowKey

sleepD:

addi $a0, $zero, 90

jal ROTATE

jal GO

jal UNTRACK # keep old track

jal TRACK # and draw new track line

j ShowKey

sleepA:

addi $a0, $zero, 270

jal ROTATE

jal GO

jal UNTRACK # keep old track

jal TRACK # and draw new track line

j ShowKey

Nghiem:

jal STOP

j ShowKey

Ditiep:

jal GO

j ShowKey

end\_main:

GO:

li $at, MOVING # change MOVING port

addi $k0, $zero, 1 # to logic 1

sb $k0, 0($at) # to start running

jr $ra

ROTATE:

li $at, HEADING # change HEADING port

sw $a0, 0($at) # to rotate robot

jr $ra

STOP:

li $at, MOVING # change MOVING port to 0

sb $zero, 0($at) # to stop

jr $ra

TRACK:

li $at, LEAVETRACK # change LEAVETRACK port

addi $k0, $zero, 1 # to logic 1

sb $k0, 0($at) # to start tracking

jr $ra

UNTRACK:

li $at, LEAVETRACK # change LEAVETRACK port to 0

sb $zero, 0($at) # to stop drawing tail

jr $ra