**BÀI THỰC HÀNH TUẦN 10**

**KIẾN TRÚC MÁY TÍNH (Phần 2)**

Họ và tên: Đinh Huy Dương

MSSV: 20215020

**Bài 1:**

Hình tam giác:

.eqv HEADING 0xffff8010 # Integer: An angle between 0 and 359

# 0 : North (up)

# 90: East (right)

# 180: South (down)

# 270: West (left)

.eqv MOVING 0xffff8050 # Boolean: whether or not to move

.eqv LEAVETRACK 0xffff8020 # Boolean (0 or non-0):

# whether or not to leave a track

.eqv WHEREX 0xffff8030 # Integer: Current x-location of MarsBot

.eqv WHEREY 0xffff8040 # Integer: Current y-location of MarsBot

.text

main:

addi $a0, $zero, 120 # Marsbot rotates 120\* and start running

jal ROTATE

nop

jal GO

nop

addi $v0,$zero,32 # Keep running by sleeping in 5000 ms

li $a0,5000

syscall

first\_side:

jal TRACK # draw track line

nop

addi $a0, $zero, 150 # Marsbot rotates 150\* and start running

jal ROTATE

nop

addi $v0,$zero,32 # Keep running by sleeping in 6000 ms

li $a0,6000

syscall

nop

jal UNTRACK

nop

second\_side:

jal TRACK

nop

addi $a0, $zero, 270 # Marsbot rotates 270\* and start running

jal ROTATE

nop

addi $v0,$zero,32 # Keep running by sleeping in 6000 ms

li $a0,6000

syscall

jal UNTRACK

nop

third\_side:

jal TRACK

nop

addi $a0, $zero, 30 # Marsbot rotates 30\* and start running

jal ROTATE

nop

addi $v0,$zero,32 # Keep running by sleeping in 6000 ms

li $a0,6000

syscall

jal UNTRACK

nop

done:

jal STOP

nop

li $v0,10

syscall

end\_main:

#-----------------------------------------------------------

# GO procedure, to start running

# param[in] none

#-----------------------------------------------------------

GO:

li $at, MOVING # change MOVING port

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

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

nop

jr $ra

nop

#-----------------------------------------------------------

# STOP procedure, to stop running

# param[in] none

#-----------------------------------------------------------

STOP:

li $at, MOVING # change MOVING port to 0

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

nop

jr $ra

nop

#-----------------------------------------------------------

# TRACK procedure, to start drawing line

# param[in] none

#-----------------------------------------------------------

TRACK:

li $at, LEAVETRACK # change LEAVETRACK port

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

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

nop

jr $ra

nop

#-----------------------------------------------------------

# UNTRACK procedure, to stop drawing line

# param[in] none

#-----------------------------------------------------------

UNTRACK:

li $at, LEAVETRACK # change LEAVETRACK port to 0

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

nop

jr $ra

nop

#-----------------------------------------------------------

# ROTATE procedure, to rotate the robot

# param[in] $a0, An angle between 0 and 359

# 0 : North (up)

# 90: East (right)

# 180: South (down)

# 270: West (left)

#-----------------------------------------------------------

ROTATE:

li $at, HEADING # change HEADING port

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

nop

jr $ra

nop

A screenshot of a computer

Description automatically generated with medium confidence

Hình vuông:

.eqv HEADING 0xffff8010 # Integer: An angle between 0 and 359

# 0 : North (up)

# 90: East (right)

# 180: South (down)

# 270: West (left)

.eqv MOVING 0xffff8050 # Boolean: whether or not to move

.eqv LEAVETRACK 0xffff8020 # Boolean (0 or non-0):

# whether or not to leave a track

.eqv WHEREX 0xffff8030 # Integer: Current x-location of MarsBot

.eqv WHEREY 0xffff8040 # Integer: Current y-location of MarsBot

.text

main:

addi $a0, $zero, 120 # Marsbot rotates 120\* and start running

jal ROTATE

nop

jal GO

nop

addi $v0,$zero,32 # Keep running by sleeping in 5000 ms

li $a0,5000

syscall

first\_side:

jal TRACK # draw track line

nop

addi $a0, $zero, 90 # Marsbot rotates 150\* and start running

jal ROTATE

nop

addi $v0,$zero,32 # Keep running by sleeping in 6000 ms

li $a0,6000

syscall

nop

jal UNTRACK

nop

second\_side:

jal TRACK

nop

addi $a0, $zero, 180 # Marsbot rotates 270\* and start running

jal ROTATE

nop

addi $v0,$zero,32 # Keep running by sleeping in 6000 ms

li $a0,6000

syscall

jal UNTRACK

nop

third\_side:

jal TRACK

nop

addi $a0, $zero, 270 # Marsbot rotates 30\* and start running

jal ROTATE

nop

addi $v0,$zero,32 # Keep running by sleeping in 6000 ms

li $a0,6000

syscall

jal UNTRACK

nop

fourth\_side:

jal TRACK

nop

addi $a0, $zero, 0 # Marsbot rotates 30\* and start running

jal ROTATE

nop

addi $v0,$zero,32 # Keep running by sleeping in 6000 ms

li $a0,6000

syscall

jal UNTRACK

nop

done:

jal STOP

nop

li $v0,10

syscall

end\_main:

#-----------------------------------------------------------

# GO procedure, to start running

# param[in] none

#-----------------------------------------------------------

GO:

li $at, MOVING # change MOVING port

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

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

nop

jr $ra

nop

#-----------------------------------------------------------

# STOP procedure, to stop running

# param[in] none

#-----------------------------------------------------------

STOP:

li $at, MOVING # change MOVING port to 0

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

nop

jr $ra

nop

#-----------------------------------------------------------

# TRACK procedure, to start drawing line

# param[in] none

#-----------------------------------------------------------

TRACK:

li $at, LEAVETRACK # change LEAVETRACK port

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

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

nop

jr $ra

nop

#-----------------------------------------------------------

# UNTRACK procedure, to stop drawing line

# param[in] none

#-----------------------------------------------------------

UNTRACK:

li $at, LEAVETRACK # change LEAVETRACK port to 0

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

nop

jr $ra

nop

#-----------------------------------------------------------

# ROTATE procedure, to rotate the robot

# param[in] $a0, An angle between 0 and 359

# 0 : North (up)

# 90: East (right)

# 180: South (down)

# 270: West (left)

#-----------------------------------------------------------

ROTATE:

li $at, HEADING # change HEADING port

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

nop

jr $ra

nop

A screenshot of a computer

Description automatically generated with medium confidence

Hình sao:

.eqv HEADING 0xffff8010 # Integer: An angle between 0 and 359

# 0 : North (up)

# 90: East (right)

# 180: South (down)

# 270: West (left)

.eqv MOVING 0xffff8050 # Boolean: whether or not to move

.eqv LEAVETRACK 0xffff8020 # Boolean (0 or non-0):

# whether or not to leave a track

.eqv WHEREX 0xffff8030 # Integer: Current x-location of MarsBot

.eqv WHEREY 0xffff8040 # Integer: Current y-location of MarsBot

.text

main:

addi $a0, $zero, 120 # Marsbot rotates 120\* and start running

jal ROTATE

nop

jal GO

nop

addi $v0,$zero,32 # Keep running by sleeping in 5000 ms

li $a0,5000

syscall

first\_side:

jal TRACK # draw track line

nop

addi $a0, $zero, 162 # Marsbot rotates 150\* and start running

jal ROTATE

nop

addi $v0,$zero,32 # Keep running by sleeping in 6000 ms

li $a0,6000

syscall

nop

jal UNTRACK

nop

second\_side:

jal TRACK

nop

addi $a0, $zero, 306 # Marsbot rotates 270\* and start running

jal ROTATE

nop

addi $v0,$zero,32 # Keep running by sleeping in 6000 ms

li $a0,6000

syscall

jal UNTRACK

nop

third\_side:

jal TRACK

nop

addi $a0, $zero, 90 # Marsbot rotates 30\* and start running

jal ROTATE

nop

addi $v0,$zero,32 # Keep running by sleeping in 6000 ms

li $a0,6000

syscall

jal UNTRACK

nop

fourth\_side:

jal TRACK

nop

addi $a0, $zero, 234 # Marsbot rotates 30\* and start running

jal ROTATE

nop

addi $v0,$zero,32 # Keep running by sleeping in 6000 ms

li $a0,6000

syscall

jal UNTRACK

nop

fifth\_side:

jal TRACK

nop

addi $a0, $zero, 18 # Marsbot rotates 30\* and start running

jal ROTATE

nop

addi $v0,$zero,32 # Keep running by sleeping in 6000 ms

li $a0,6000

syscall

jal UNTRACK

nop

done:

jal STOP

nop

li $v0,10

syscall

end\_main:

#-----------------------------------------------------------

# GO procedure, to start running

# param[in] none

#-----------------------------------------------------------

GO:

li $at, MOVING # change MOVING port

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

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

nop

jr $ra

nop

#-----------------------------------------------------------

# STOP procedure, to stop running

# param[in] none

#-----------------------------------------------------------

STOP:

li $at, MOVING # change MOVING port to 0

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

nop

jr $ra

nop

#-----------------------------------------------------------

# TRACK procedure, to start drawing line

# param[in] none

#-----------------------------------------------------------

TRACK:

li $at, LEAVETRACK # change LEAVETRACK port

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

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

nop

jr $ra

nop

#-----------------------------------------------------------

# UNTRACK procedure, to stop drawing line

# param[in] none

#-----------------------------------------------------------

UNTRACK:

li $at, LEAVETRACK # change LEAVETRACK port to 0

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

nop

jr $ra

nop

#-----------------------------------------------------------

# ROTATE procedure, to rotate the robot

# param[in] $a0, An angle between 0 and 359

# 0 : North (up)

# 90: East (right)

# 180: South (down)

# 270: West (left)

#-----------------------------------------------------------

ROTATE:

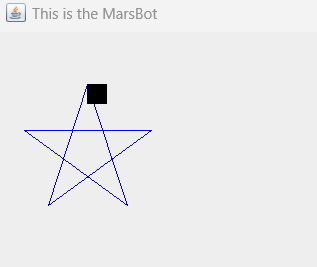
li $at, HEADING # change HEADING port

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

nop

jr $ra

nop



**Bài 2:**

.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

# Auto clear after sw

.text

li $k0, KEY\_CODE

li $k1, KEY\_READY

li $s0, DISPLAY\_CODE

li $s1, DISPLAY\_READY

li $s5,1 # mask so it can only be 0/1

li $t8,122 # $t8 = z

li $t9,90 # $t9 = Z

addi $s2,$0,0x30

addi $s3,$0,0x39

loop:

nop

WaitForKey:

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

nop

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

nop

#-----------------------------------------------------

ReadKey:

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

nop

#-----------------------------------------------------

WaitForDis:

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

nop

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

nop

#-----------------------------------------------------

Encrypt:

slti $t1,$t0,97 # check str[i] < a

sgt $t2,$t0,$t8 # check str[i] > z

slti $t3,$t0,65 # check str[i] < A

sgt $t4,$t0,$t9 # check str[i] > Z

nor $t5,$t1,$t2 # (str[i] < a) NOR (str[i] > z) = a <= str[i] <= z

and $t5,$s5,$t5

nor $t6,$t3,$t4 # (str[i] < A) NOR (str[i] > Z) = A <= str[i] <= Z

and $t6,$s5,$t6

or $t7,$t5,$t6

beq $t7,$0,skip2

beq $t5,$s5,upper # if a< str[i] <z, go to upper

beq $t6,$s5,lower # if A< str[i] <Z, go to lower

skip:

j ShowKey

#------------------------------------------

# Procedure: Upper

upper:

addi $t0,$t0,-32 # Upper(str[i])

j skip

#------------------------------------------

# Procedure: Lower

lower:

addi $t0,$t0,32 # Lower(str[i])

j skip

#------------------------------------------

skip2:

slt $t1,$t0,$s2 # check str[i] < 0

sgt $t2,$t0,$s3 # check str[i] > 9

nor $t3,$t1,$t2

and $t3,$s5,$t3

beq $t3,$s5,skip

addi $t0,$0,0x2a

j skip

#-----------------------------------------------------

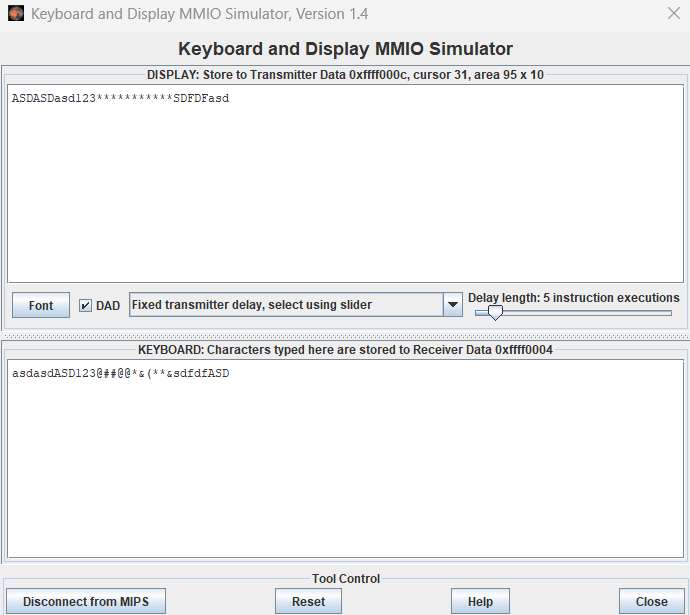
ShowKey:

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

nop

j loop

nop

****