**REPORT WEEK 10\_2**

Họ và tên: Nguyễn Văn Hưng

MSSV: 20225634

Lớp: Thực hành kiến trúc máy tính – Mã lớp: 147799

1. **Assignment 1:**
2. Đề bài: Điều khiển marsbot di chuyển theo hình tam giác đều, hình vuông, hình ngôi sao 5 cánh
3. Code:

.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: jal UNTRACK # draw track line

addi $a0, $zero, 135 # Marsbot rotates 90\* and start running

jal ROTATE

jal GO

sleep1: addi $v0,$zero,32 # Keep running by sleeping in 1000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

tamgiac1: addi $a0, $zero, 150 # Marsbot rotates 180\*

jal ROTATE

sleep2: addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

tamgiac2: addi $a0, $zero, 270 # Marsbot rotates 270\*

jal ROTATE

sleep3: addi $v0,$zero,32 # Keep running by sleeping in 1000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

tamgiac3:addi $a0, $zero, 30 # Marsbot rotates 120\*

jal ROTATE

sleep4: addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

resert1:

jal UNTRACK # keep old track

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

jal ROTATE

jal GO

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

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

vuong1: addi $a0, $zero, 90 # Marsbot rotates 180\*

jal ROTATE

addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

vuong2: addi $a0, $zero, 180 # Marsbot rotates 270\*

jal ROTATE

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

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

vuong3:addi $a0, $zero, 270 # Marsbot rotates 120\*

jal ROTATE

addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

vuong4:addi $a0, $zero, 0 # Marsbot rotates 120\*

jal ROTATE

addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

resert2:

jal UNTRACK # keep old track

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

jal ROTATE

jal GO

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

li $a0,10000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

sao1: addi $a0, $zero, 162 # Marsbot rotates 180\*

jal ROTATE

addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

sao2: addi $a0, $zero, 306 # Marsbot rotates 270\*

jal ROTATE

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

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

sao3:addi $a0, $zero, 90 # Marsbot rotates 120\*

jal ROTATE

addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

sao4:addi $a0, $zero,234 # Marsbot rotates 120\*

jal ROTATE

addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

sao5:addi $a0, $zero,18 # Marsbot rotates 120\*

jal ROTATE

addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

end\_main:

jal STOP

li $v0, 10

syscall

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

# 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

jr $ra

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

# STOP procedure, to stop running

# param[in] none

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

STOP: li $at, MOVING # change MOVING port to 0

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

jr $ra

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

# 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

jr $ra

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

# 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

jr $ra

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

# 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

jr $ra

1. Giải thích

**Các thủ tục**

* **GO**: Bắt đầu di chuyển robot bằng cách đặt MOVING thành 1.
* **STOP**: Dừng robot bằng cách đặt MOVING thành 0.
* **TRACK**: Bắt đầu vẽ dấu bằng cách đặt LEAVETRACK thành 1.
* **UNTRACK**: Dừng vẽ dấu bằng cách đặt LEAVETRACK thành 0.
* **ROTATE**: Thay đổi góc di chuyển của robot bằng cách đặt giá trị vào HEADING.

**Chương trình chính**

Chương trình chính điều khiển MarsBot để thực hiện các di chuyển lần lượt cho từng cạnh và vẽ dấu. Mỗi khi kết thúc 1 hình thì vẽ 1 đường ngang ko vết để sang hình mới.

1. Kết quả

A blue star and a black rectangle

Description automatically generated

1. **Assignment 2:**
2. Đề bài: Nhập ký tự ở Keyboard và hiển thị ở Display: nhập ký tự thường => hiển thị ký tự hoa tương ứng, nhập ký tự hoa => hiển thị ký tự thường tương ứng, nhập ký tự số thì giữ nguyên, nhập ký tự khác => hiển thị ký tự \*. Khi nhập chuỗi ký tự "exit" thì kết thúc chương trình.
3. Code:

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

# Auto clear after sw

.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:

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:

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

1. Giải thích

Chương trình MIPS này thực hiện các chức năng sau:

1. **Chờ phím nhấn từ bàn phím**: Chương trình liên tục kiểm tra để xem có phím mới được nhấn hay không. Nó sử dụng địa chỉ KEY\_READY để kiểm tra trạng thái của bàn phím.
2. **Đọc mã ASCII từ bàn phím**: Khi có phím mới được nhấn, chương trình đọc mã ASCII của phím đó từ địa chỉ KEY\_CODE.
3. **Chờ màn hình sẵn sàng để hiển thị ký tự**: Trước khi hiển thị ký tự, chương trình kiểm tra xem màn hình có sẵn sàng không thông qua địa chỉ DISPLAY\_READY.
4. **Kiểm tra và xử lý các ký tự đặc biệt**: Chương trình có các bước để kiểm tra xem ký tự nhập vào có phải là một trong các ký tự của chuỗi "EXIT" không. Nếu nhập lần lượt các ký tự "E", "X", "I", "T", chương trình sẽ thoát.
5. **Xử lý ký tự hoa/thường và các ký tự khác**:
   * Nếu ký tự là chữ hoa (A-Z), nó sẽ được chuyển thành chữ thường tương ứng (a-z).
   * Nếu ký tự là chữ thường (a-z), nó sẽ được chuyển thành chữ hoa tương ứng (A-Z).
   * Nếu ký tự là chữ số (0-9), nó sẽ được giữ nguyên.
   * Các ký tự khác sẽ được chuyển thành dấu sao (\*).
6. **Hiển thị ký tự lên màn hình**: Sau khi xử lý, ký tự được gửi đến màn hình thông qua địa chỉ DISPLAY\_CODE.
7. **Lặp lại**: Chương trình tiếp tục quay lại bước đầu để chờ phím nhấn mới và lặp lại quy trình cho đến khi nhận được chuỗi "EXIT", khi đó chương trình sẽ kết thúc.
8. Kết quả

A screenshot of a computer

Description automatically generated

1. **Assignment 3:**
2. Đề bài: Dùng keyboard điều khiển marsbot

+ Space: bắt đầu / dừng di chuyển

+ W: đi lên, S: đi xuống, A: sang trái, D: sang phải (viết hoa hoặc viết thường đều được)

Cần vẽ đường di chuyển của marsbot.

1. 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

# Auto clear after sw

.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 por

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

1. Giải thích

Chương trình có các chức năng chính như sau:

1. **Chờ và Đọc Phím Nhấn**: Chương trình liên tục kiểm tra xem có phím mới được nhấn hay không và đọc mã ASCII của phím đó.
2. **Kiểm Tra Chuỗi "EXIT"**: Chương trình kiểm tra xem các ký tự nhập vào có tạo thành chuỗi "EXIT" không để thoát chương trình.
3. **Điều Khiển Robot**: Dựa trên các phím nhấn (W, A, S, D, Space, C), chương trình thực hiện các hành động tương ứng như di chuyển robot, dừng lại, hoặc tiếp tục di chuyển.

**Chi Tiết Hoạt Động**

1. **Chờ và Đọc Phím Nhấn**:
   * Sử dụng **KEY\_READY** để kiểm tra nếu có phím mới được nhấn.
   * Nếu có phím mới, đọc mã ASCII từ **KEY\_CODE**.
2. **Kiểm Tra Chuỗi "EXIT"**:
   * Các bước kiểm tra từng ký tự để xem chuỗi "EXIT" có được nhập vào hay không.
   * Nếu đúng chuỗi, thoát khỏi chương trình.
3. **Xử Lý Các Ký Tự Đặc Biệt**:
   * Kiểm tra và thực hiện các hành động dựa trên các ký tự nhập vào:
     + **W** hoặc **w**: Di chuyển robot về phía trước.
     + **A** hoặc **a**: Xoay robot về bên trái và di chuyển.
     + **S** hoặc **s**: Xoay robot về phía sau và di chuyển.
     + **D** hoặc **d**: Xoay robot về bên phải và di chuyển.
     + **Space** (32): Dừng robot.
     + **C** hoặc **c**: Tiếp tục di chuyển robot
4. Kết quả

A screenshot of a computer

Description automatically generated