TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

----- ☙ 🕮 ❧ -----

ĐỒ ÁN MÔN HỌC

Đề tài: **Project cuối kì**

Học phần: **Thực hành Kiến trúc máy tính**

Mã lớp : ***131001***

Curiosity Marsbot

Vẽ hình trên màn hình Bitmap

Giảng viên: **Ths. Nguyễn Văn Hiên**

Nhóm sinh viên thực hiện:

STT	Họ và tên	MSSV
1	Hoàng Minh Ngọc	20200440

Hà Nội, năm 2022

Bài 1: Curiosity Marsbot

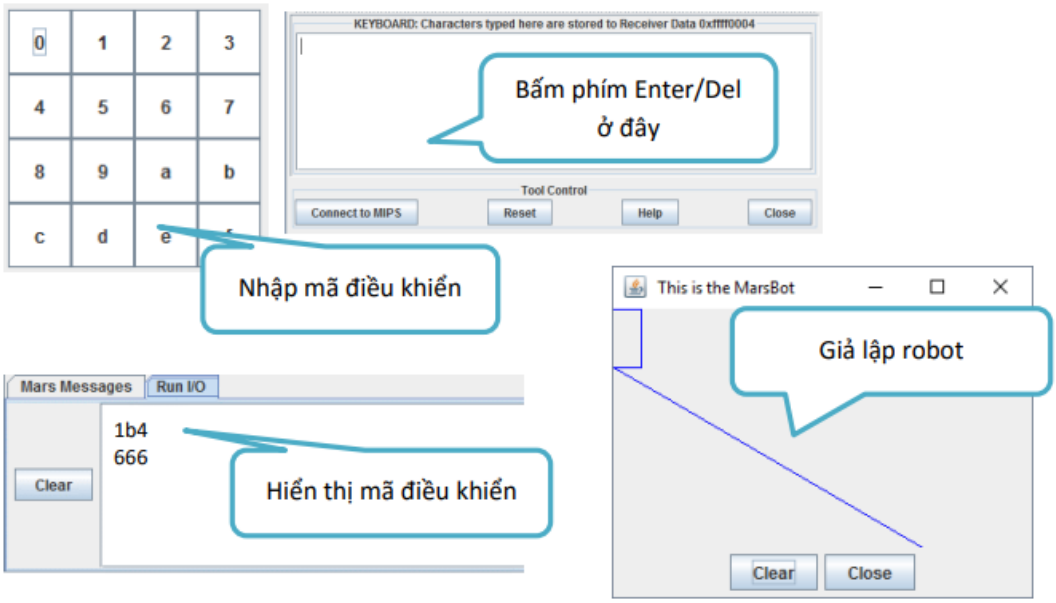
Xe tự hành Curioisity Marsbot chạy trên sao Hỏa, được vận hành từ xa bởi các lập trình viên trên Trái Đất. Bằng cách gửi đi các mã điều khiển từ một bàn phím ma trận, lập trình viên điều khiển quá trình di chuyển của Marbot như sau:

Mã điều khiển	Ý nghĩa
1b4	Marbot bắt đầu chuyển động
c68	Marbot đứng im
444	Rẽ trái 90* so vơi phương chuyển động gần đây và giữ hướng mới
666	Rẽ phải 90* so vơi phương chuyển động gần đây và giữ hướng mới
dad	Bắt đầu để lại vết trên đường
cbc	Chấm dứt để lại vết trên đường
999	Tự động quay trở lại theo lộ trình ngược lại. Không vẽ vết, không nhận mã khác cho tới khi kết thúc lộ trình ngược. Mô tả: Marsbot được lập trình để nhớ lại toàn bộ lịch sử các mã điều khiển và khoảng thời gian giữa các lần đổi mã. Vì vậy, nó có thể đảo ngược lại lộ trình để quay về điểm xuất phát (dù có thể lệch một chút do hàm syscall sleep không thực sự thời gian thực)

Sau khi nhận mã điều khiển, Curioisity Marsbot sẽ không xử lý ngay, mà phải đợi lệnh kích hoạt mã từ bàn phím Keyboard & Display MMIO Simulator. Có 2 lệnh như vậy:

Kích hoạt mã	Ý nghĩa
Phím Enter	Kết thúc nhập mã và yêu cầu Marbot thực thi
Phím Del	Xóa toàn bộ mã điều khiển đang nhập dở dang.

Hãy lập trình để Marsbot có thể hoạt động như đã mô tả.
Đồng thời bổ sung thêm tính năng: mỗi khi gửi một mã điều khiển cho Marsbot, hiển thị mã đó lên màn hình console để người xem có thể giám sát lộ trình của xe.



I) Ý tưởng thực hiện

- Sử dụng Digital Lab Sim, Keyboard and Display và MarsBot

- Lấy tín hiệu	- Digital Lab Sim	+ Nạp mã điều khiển
	- MarsBot	+ Màn hình Robot di chuyển
	- Keyboard and Display MMIO Simulator	+ Xác nhận lệnh thực thi
	- Run I/O	+ Xuất ra màn hình bằng syscall

Ta sử dụng cấu trúc dữ liệu Stack để phục vụ cho lệnh 999 quay trở lại vị trí ban đầu, mọi lệnh khác sẽ là lưu vào Stack và lệnh 999 là đưa ra khỏi Stack

Đầu tiên ta cần thiết lập Digital Lab Sim nhập mã điều khiển

Khởi tạo các giá trị ban đầu

```
.eqv IN_ADRESS_HEXA_KEYBOARD 0xFFFF0012
.eqv OUT_ADRESS_HEXA_KEYBOARD 0xFFFF0014
.eqv KEY_CODE 0xFFFF0004          # ma ASCII tu ban phim, 1 byte
.eqv KEY_READY 0xFFFF0000         # =1 neu co ma khoa moi
                                  # tu dong xoa sau lw
#------------------------------------------------------------------
```

Khởi tạo, nhận các giá trị trên toàn bộ các hàng, cột của Digital Lab Sim

```
.data
# gia tri nhan vao
#0-3
        .eqv KEY_0 0x11
        .eqv KEY_1 0x21
        .eqv KEY_2 0x41
        .eqv KEY_3 0x81
#4-7
        .eqv KEY_4 0x12
        .eqv KEY_5 0x22
        .eqv KEY_6 0x42
        .eqv KEY_7 0x82
#8-b
        .eqv KEY_8 0x14
        .eqv KEY_9 0x24
        .eqv KEY_a 0x44
        .eqv KEY_b 0x84
#c-f
        .eqv KEY_c 0x18
        .eqv KEY_d 0x28
        .eqv KEY_e 0x48
        .eqv KEY_f 0x88
```

Tạo ra các biến chứa chuỗi kí tự để so sánh

```
        .eqv KEY_f 0x88
#------------------------------------------------------------------
#tao cac lenh chuc nang
        MOVE_CODE: .asciiz "1b4"        # di chuyen
        STOP_CODE: .asciiz "c68"        # dung im
        GO_LEFT_CODE: .asciiz "444"     # re trai 90 do
        GO_RIGHT_CODE: .asciiz "666"    # re phai 90 do
        TRACK_CODE: .asciiz "dad"       # ve
        UNTRACK_CODE: .asciiz "cbc"     # ko ve
        GO_BACK_CODE: .asciiz "999"     # quay tro lai, ko ve. ko nhan ma toi khi het
        WRONG_CODE: .asciiz "Da nhap ma dieu khien sai !"
#------------------------------------------------------------------
        inputMaDieuKhien: .space 50 # dau vao ma dieu khien
        lengthMaDieuKhien: .word 0  # ma khiem soat do dai
        nowHeading: .word 0
#-----------------------------------------------------
# duong di cua masbot duoc luu tru vao mang lichsu
```

Ta cần lưu lịch sử di chuyển của MarsBot vào Stack, ta cần lưu vào 1 mảng tên là lichsu để định nghĩa dưới dạng một cấu trúc

```
#-------------------------------------------------------------
# duong di cua masbot duoc luu tru vao mang lichsu
# moi 1 canh duoc luu tru duoi dang 1 cau truc
# 1 cau truc co dang (x, y, z)
# trong do:      x, y la toa do diem dau tien cua canh
#                z la huong cua canh do
# mac dinh:      cau truc dau tien se la (0,0,0)
# do dai duong di ngay khi bat dau la 12 bytes (3x 4byte)
#-------------------------------------------------------------
        lichsu: .space 600
        lengthlichsu: .word 12          #bytes

"
```

Tạo ra các câu lệnh để quản trị hệ thống

```
#================================================================================
.text
main:
        li $k0, KEY_CODE        # ma khoa
        li $k1, KEY_READY       # khoa san sang
#---------------------------------------------------------
# ngat ma tran ban phim 4x4 trong Digital Lab Sim
#---------------------------------------------------------
        li $t1, IN_ADRESS_HEXA_KEYBOARD
        li $t3, 0x80 # bit 7 = 1 de kich hoat
        sb $t3, 0($t1)
#---------------------------------------------------------
loop:           nop
WaitForKey:
        lw $t5, 0($k1)                  #$t5 = [$k1] = khoa san sang
        beq $t5, $zero, WaitForKey      #neu $t5 == 0 quay lai WaitForKey cho tin hieu
        nop
        beq $t5, $zero, WaitForKey      #neu $t5 == 0 quay lai WaitForKey cho tin hieu
ReadKey:
        lw $t6, 0($k0)                  #$t6 = [$k0] = KEY_CODE
        beq $t6, 127 , continue         #if $t6 == Del thi xoa toan bo dau vao
                                        #127 la ma Delete xong ASCII
        bne $t6, '\n' , loop            #neu t6 = \n thi quay lai loop de lay code dieu khien
        nop
        bne $t6, '\n' , loop
```

Wait For Key chờ lấy khóa để chuẩn bị thực thi

Read Key chờ tín hiệu thực thi từ Keyboard and Display MMIO Simulator

MỤC LỤC TRA CỨU CÁC THỦ TỤC

Check mã điều khiển, phân luồng các chức năng

```
CheckMaDieuKhien:       #kiem tra ma
        la $s2, lengthMaDieuKhien       #ma kiem soat do dai
        lw $s2, 0($s2)
        #---------------
        bne $s2, 3, pushErrorMess       #thong bao ma dieu khien dua vao la sai

        la $s3, MOVE_CODE               #lenh di chuyen
        jal isEqualString               #kiem tra chuoi
        beq $t0, 1, go

        la $s3, STOP_CODE               #lenh dung
        jal isEqualString               #kiem tra chuoi
        beq $t0, 1, stop

        la $s3, GO_LEFT_CODE            #lenh re trai
        jal isEqualString               #kiem tra chuoi
        beq $t0, 1, goLeft

        la $s3, GO_RIGHT_CODE           #lenh re phai
        jal isEqualString               #kiem tra chuoi
        beq $t0, 1, goRight

        la $s3, TRACK_CODE              #lenh ve
        jal isEqualString               #kiem tra chuoi
        beq $t0, 1, track

        la $s3, UNTRACK_CODE            #lenh khong ve
        jal isEqualString               #kiem tra chuoi

    la $s3, GO_BACK_CODE                #lenh quay lai
    jal isEqualString                   #kiem tra chuoi
    beq $t0, 1, goBack

    beq $t0, 0, pushErrorMess           #ma dieu khien dua vao la sai
```

In ra các mã điều khiển trên Mars Messages
Bằng lời gọi hệ thống syscall
Continue lặp lại các bước trên ở dòng 88
Để thực hiện các lệnh tiếp theo

```
printMaDieuKhien:       # in ra ma dieu khien tren Mars Messages
        li $v0, 4
        la $a0, inputMaDieuKhien        #dau vao cua code
        syscall
        nop

continue:                               #tiep theo
        jal removeMaDieuKhien           #Loai bo chuoi InputMaDieuKhien
        nop
        j loop                          #quay lai nhan key moi
        nop
        j loop
```

```
#-------------------------------------------------------------
# thu tuc goBack, dieu khien MarsBot quay lai
# param[in]      lichsu array, lengthlichsu array
#-------------------------------------------------------------
```

```
goBack: la $s7, lichsu          # ma tran
        la $s5, lengthlichsu    # byte toi da
        lw $s5, 0($s5)
        add $s7, $s7, $s5
begin:  addi $s5, $s5, -12      #lui lai 1 cau truc
        addi $s7, $s7, -12      #vi tri cua thong tin ve canh cuoi cung
        lw $s6, 8($s7)          #huong cua canh cuoi cung
        addi $s6, $s6, 180      #nguoc lai huong cua canh cuoi cung

        la $t8, nowHeading      #marsbot quay nguoc lai
        sw $s6, 0($t8)
        jal ROTATE
```

1

```
go_to_first_point_of_edge:      #toi diem dau cua canh
        lw $t9, 0($s7)          #toa do x cua diem dau tien cua canh
        li $t8, WHEREX          #toa do x hien tai
        lw $t8, 0($t8)

        bne $t8, $t9, go_to_first_point_of_edge

        lw $t9, 4($s7)          #toa do y cua diem dau tien cua canh
        li $t8, WHEREY          #toa do y hien tai
        lw $t8, 0($t8)

        bne $t8, $t9, go_to_first_point_of_edge

        beq $s5, 0, finish

        j begin
```

2

```
finish: jal STOP
        la $t8, nowHeading
        add $s6, $zero, $zero
        sw $s6, 0($t8)
        la $t8, lengthlichsu
        sw $s5, 0($t8)
        jal ROTATE
        j printMaDieuKhien

#cap nhat heading

#cap nhat lengthlichsu = 0
```

3

Thủ tục storelichsu lưu vào Stack

Lưu vào Stack

```
#luu lai tung ki tu vao cac bien
#sp tro toi dinh cua stack
addi $sp,$sp,4
sw $t1, 0($sp)
addi $sp,$sp,4
sw $t2, 0($sp)
addi $sp,$sp,4
sw $t3, 0($sp)
addi $sp,$sp,4
sw $t4, 0($sp)

addi $sp,$sp,4
sw $s1, 0($sp)
addi $sp,$sp,4
sw $s2, 0($sp)
addi $sp,$sp,4
sw $s3, 0($sp)
addi $sp,$sp,4
sw $s4, 0($sp)
```

Các thủ tục

```
#thu tuc
li $t1, WHEREX
lw $s1, 0($t1)          #s1 = x
li $t2, WHEREY
lw $s2, 0($t2)          #s2 = y

la $s4, nowHeading
lw $s4, 0($s4)          #s4 = now heading

la $t3, lengthlichsu
lw $s3, 0($t3)          #$s3 = lengthlichsu (dv: byte)

la $t4, lichsu
add $t4, $t4, $s3       #vi tri de luu tru vao mang lichsu
```

Chuyển dữ liệu từ thanh ghi ra t4

```
sw $s1, 0($t4)
sw $s2, 4($t4)
sw $s4, 8($t4)

addi $s3, $s3, 12

sw $s3, 0($t3)
#store x
#store y
#store heading

#cap nhat lengthlichsu
#12 = 3 (word) x 4 (bytes)
```

Khôi phục lại các giá trị trong Stack (lấy giá trị ra khỏi Stack)

```
#khoi phuc lai gia tri trong stack
lw $s4, 0($sp)
addi $sp,$sp,-4
lw $s3, 0($sp)
addi $sp,$sp,-4
lw $s2, 0($sp)
addi $sp,$sp,-4
lw $s1, 0($sp)
addi $sp,$sp,-4
lw $t4, 0($sp)
addi $sp,$sp,-4
lw $t3, 0($sp)
addi $sp,$sp,-4
lw $t2, 0($sp)
addi $sp,$sp,-4
lw $t1, 0($sp)
addi $sp,$sp,-4

jr $ra
nop
jr $ra
```

Các thủ tục khác

```
#-----------------------------------------------------------
# thu tuc track ,dieu khien MarsBot de theo doi va in ma dieu khien
# param[in] none
#-----------------------------------------------------------
track:  jal TRACK
        j printMaDieuKhien
#-----------------------------------------------------------
```

```
#-----------------------------------------------------------
# quy trinh untrack, bo theo doi,
#dieu khien MarsBot de bo theo doi va in ma dieu khien
#-----------------------------------------------------------
untrack: jal UNTRACK
         j printMaDieuKhien
#-----------------------------------------------------------
```

```
#-----------------------------------------------------------
# thu tuc go, dieu khien MarsBot de di va in ra ma dieu khien
# param[in] none
#-----------------------------------------------------------
go:     jal GO
        j printMaDieuKhien
#-----------------------------------------------------------
#-----------------------------------------------------------
# thu tuc stop, dieu khien MarsBot dung va in ra ma dieu khien
# param[in] none
#-----------------------------------------------------------
stop:   jal STOP
        j printMaDieuKhien
#-----------------------------------------------------------
#-----------------------------------------------------------
# thu tuc goRight , dieu khien MarsBot sang trai va in ma dieu khien
# param[in] none
#-----------------------------------------------------------
goRight:la $s5, nowHeading
        lw $s6, 0($s5)          # $s6 = $s5 = n?Heading
        addi $s6, $s6, 90       # ( tang len 90 )
        sw $s6, 0($s5)          # cap nhat nowHeading
        jal storelichsu         # luu tru duong dan cua MarsBot vao stack
        jal ROTATE              # dieu khien robot xoay
        j printMaDieuKhien      # in ra ma dieu khien
#-----------------------------------------------------------
#-----------------------------------------------------------
# thu tuc goLeft,dieu khien MarsBot sang phai va in ma dieu khien
# param[in] none
#-----------------------------------------------------------
goLeft: la $s5, nowHeading
        lw $s6, 0($s5)          # $6 = nowHeading
        addi $s6, $s6, -90      # cong s6 them -90, xoay phai 90 do
        sw $s6, 0($s5)          # cap nhat nowHeading
        jal storelichsu         # luu tru duong dan cua MarsBot vao stack
        jal ROTATE              # dieu khien robot xoay
        j printMaDieuKhien      # in ra ma dieu khien
#-----------------------------------------------------------
```

```
#-----------------------------------------------------------
# thu tuc pushErrorMess , thong bao ma bi sai
# param[in] none
#-----------------------------------------------------------
pushErrorMess: li $v0, 4
        la $a0, inputMaDieuKhien
        syscall
        nop

        li $v0, 55
        la $a0, WRONG_CODE
        syscall
        nop
        nop
        j continue
        nop
        j continue
#-----------------------------------------------------------
```

```
#-----------------------------------------------------------
# thu tuc removeMaDieuKhien , loai bo chuoi ma dieu kien dau vao
#                              inputMaDieuKhien = ""
# param[in] none
#-----------------------------------------------------------
removeMaDieuKhien:
        #sao luu vao stack
        addi $sp,$sp,4
        sw $t1, 0($sp)
        addi $sp,$sp,4
        sw $t2, 0($sp)
        addi $sp,$sp,4
        sw $s1, 0($sp)
        addi $sp,$sp,4
        sw $t3, 0($sp)
        addi $sp,$sp,4
        sw $s2, 0($sp)

        #thu tuc
        la $s2, lengthMaDieuKhien
        lw $t3, 0($s2)                          #$t3 = lengthMaDieuKhien
        addi $t1, $zero, -1                     #$t1 = -1 = i
        addi $t2, $zero, 0                      #$t2 = '\0'
        la $s1, inputMaDieuKhien
        addi $s1, $s1, -1
for_loop_to_remove: addi $t1, $t1, 1            #i++
        add $s1, $s1, 1                         #$s1 = inputMaDieuKhien + i
        sb $t2, 0($s1)                          #inputMaDieuKhien[i] = '\0'
```

```
        bne $t1, $t3, for_loop_to_remove          #if $t1 <=3 continue loop
        nop
        bne $t1, $t3, for_loop_to_remove

        add $t3, $zero, $zero
        sw $t3, 0($s2)                            #lengthMaDieuKhien = 0

        #khoi phuc
        lw $s2, 0($sp)
        addi $sp,$sp,-4
        lw $t3, 0($sp)
        addi $sp,$sp,-4
        lw $s1, 0($sp)
        addi $sp,$sp,-4
        lw $t2, 0($sp)
        addi $sp,$sp,-4
        lw $t1, 0($sp)
        addi $sp,$sp,-4

        jr $ra
        nop
        jr $ra
#-------------------------------------------------------------
```

```
#-------------------------------------------------------------
# thu tuc isEqualString , kiem tra inputMaDieuKhien
#                                  kiem tra co bang voi chuoi s ( luu trong $s3)
#                                  Do dai 2 chuoi la nhu nhau
# param[in] $s3, di chi la 1 chuoi
# param[out] $t0,1 neu bang nhau, 0 neu ko bang nhau
#-------------------------------------------------------------
```

```
#-------------------------------------------------------
isEqualString:
        #sao luu vao stack
        addi $sp,$sp,4
        sw $t1, 0($sp)
        addi $sp,$sp,4
        sw $s1, 0($sp)
        addi $sp,$sp,4
        sw $t2, 0($sp)
        addi $sp,$sp,4
        sw $t3, 0($sp)

        #thu tuc
        addi $t1, $zero, -1                  #$t1 = -1 = i
        add $t0, $zero, $zero
        la $s1, inputMaDieuKhien             #$s1 = inputMaDieuKhien
for_loop_to_check_equal: addi $t1, $t1, 1                    #i++

        add $t2, $s1, $t1                    #$t2 = inputMaDieuKhien + i
        lb $t2, 0($t2)                       #$t2 = inputMaDieuKhien[i]

        add $t3, $s3, $t1                    #$t3 = s + i
        lb $t3, 0($t3)                       #$t3 = s[i]
```

1

```
isEqual:
        #sao luu vao stack
        lw $t3, 0($sp)
        addi $sp,$sp,-4
        lw $t2, 0($sp)
        addi $sp,$sp,-4
        lw $s1, 0($sp)
        addi $sp,$sp,-4
        lw $t1, 0($sp)
        addi $sp,$sp,-4

        add $t0, $zero, 1        #cap nhat $t0
        jr $ra
        nop
        jr $ra
```

3

2

```
bne $t2, $t3, isNotEqual              #if $t2 != $t3 -> isNotequal

bne $t1, 2, for_loop_to_check_equal   #if $t1 <=2 tiep tuc toi loop
nop
bne $t1, 2, for_loop_to_check_equal
```

```
isNotEqual:
        #restore
        lw $t3, 0($sp)
        addi $sp,$sp,-4
        lw $t2, 0($sp)
        addi $sp,$sp,-4
        lw $s1, 0($sp)
        addi $sp,$sp,-4
        lw $t1, 0($sp)
        addi $sp,$sp,-4

        add $t0, $zero, $zero    #cap nhat $t0
        jr $ra
        nop
        jr $ra
```

4

```
#-----------------------------------------------------------
# thu tuc pushErrorMess , thong bao ma bi sai
# param[in] none
#-----------------------------------------------------------
"
pushErrorMess: li $v0, 4
        la $a0, inputMaDieuKhien
        syscall
        nop

        li $v0, 55
        la $a0, WRONG_CODE
        syscall
        nop
        nop
        j continue
        nop
        j continue
```

```
#-------------------------------------------------------------
# thu tuc GO , bat dau chay
# param[in] none
#-------------------------------------------------------------
GO:     #sao luu vao stack
        addi $sp,$sp,4
        sw $at,0($sp)
        addi $sp,$sp,4
        sw $k0,0($sp)
        #thu tuc
        li $at, MOVING          # lenh kich hoat duoc thuc thi
        addi $k0, $zero,1       # to logic 1,
        sb $k0, 0($at)          # bat dau chay
        #khoi phuc
        lw $k0, 0($sp)
        addi $sp,$sp,-4
        lw $at, 0($sp)
        addi $sp,$sp,-4

        jr $ra
        nop
        jr $ra
```

```
#-------------------------------------------------------------
# thu tuc STOP , dung chay
# param[in] none
#-------------------------------------------------------------
STOP:   #sao luu
        addi $sp,$sp,4
        sw $at,0($sp)
        #thu tuc
        li $at, MOVING          # thay doi sang MOVING
        sb $zero, 0($at)        # dung lai
        #khoi phuc
        lw $at, 0($sp)
        addi $sp,$sp,-4

        jr $ra
        nop
        jr $ra
#-------------------------------------------------------------
```

```
#-----------------------------------------------------------
# thu tuc TRACK , ve duong
# param[in] none
#-----------------------------------------------------------
TRACK:   #sao luu
         addi $sp,$sp,4
         sw $at,0($sp)
         addi $sp,$sp,4
         sw $k0,0($sp)
         #thu tuc
         li $at, LEAVETRACK          # thay doi sang LEAVETRACK
         addi $k0, $zero,1           # to logic 1,
         sb $k0, 0($at)              # to start tracking
         #khoi phuc
         lw $k0, 0($sp)
         addi $sp,$sp,-4
         lw $at, 0($sp)
         addi $sp,$sp,-4

         jr $ra
         nop
         jr $ra
#-----------------------------------------------------------
```

```
#-------------------------------------------------------------
# thu tuc UNTRACK , dung ve duong
# param[in] none
#-------------------------------------------------------------
UNTRACK:
        #sao luu
        addi $sp,$sp,4
        sw $at,0($sp)
        #thu tuc
        li $at, LEAVETRACK      # thay doi sang cong LEAVETRACK va cho = 0
        sb $zero, 0($at)        # dung ve
        #khoi phuc
        lw $at, 0($sp)
        addi $sp,$sp,-4

        jr $ra
        nop
        jr $ra
#-------------------------------------------------------------
#-------------------------------------------------------------------
# thu tuc ROTATE_RIGHT , dieu khien robot xoay
# param[in] nowHeading variable, store heading at present
#-------------------------------------------------------------------
ROTATE:
         #sao luu
         addi $sp,$sp,4
         sw $t1,0($sp)
         addi $sp,$sp,4
         sw $t2,0($sp)
         addi $sp,$sp,4
         sw $t3,0($sp)
         #thu tuc
         li $t1, HEADING            # thay doi xong sang HEADING
         la $t2, nowHeading
         lw $t3, 0($t2)             # t3 = nowHeading
         sw $t3, 0($t1)             # xoay robot
         #khoi phuc
         lw $t3, 0($sp)
         addi $sp,$sp,-4
         lw $t2, 0($sp)
         addi $sp,$sp,-4
         lw $t1, 0($sp)
         addi $sp,$sp,-4

         jr $ra
         nop
         jr $ra
#-------------------------------------------------------------------
```

```
#-------------------------------------------------------------
# thu tuc ROTATE_LEFT , xoay robot sang ben phai
# param[in] nowHeading variable, store heading at present
#-------------------------------------------------------------
ROTATE_LEFT:
        #sao luu
        addi $sp,$sp,4
        sw $t1,0($sp)
        addi $sp,$sp,4
        sw $t2,0($sp)
        addi $sp,$sp,4
        sw $t3,0($sp)
        #thu tuc
        li $t1, HEADING          # thay doi cong HEADING
        la $t2, nowHeading
        lw $t3, 0($t1)           # $t2 = HEADING
        addi $t3, $t3, -90       # xoay sang ben phai nen -90
        sw $t3, 0($t2)           # cap nhat nowHeading
        sw $t3, 0($t1)           # xoay robot
        #khoi phuc
        lw $t3, 0($sp)
        addi $sp,$sp,-4
        lw $t2, 0($sp)
        addi $sp,$sp,-4
        lw $t1, 0($sp)
        addi $sp,$sp,-4

        jr $ra
        nop
        jr $ra
#=============================================================
```

```
#============================================================================
# GENERAL INTERRUPT SERVED ROUTINE cho tat ca interrupts
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

```
.ktext 0x80000180
#-----------------------------------------
# luu cac tap tin vao stack
#-----------------------------------------
backup: #sao luu tu thanh ghi vao stack
        addi $sp,$sp,4
        sw $ra,0($sp)
        addi $sp,$sp,4
        sw $t1,0($sp)
        addi $sp,$sp,4
        sw $t2,0($sp)
        addi $sp,$sp,4
        sw $t3,0($sp)
        addi $sp,$sp,4
        sw $a0,0($sp)
        addi $sp,$sp,4
        sw $at,0($sp)
        addi $sp,$sp,4
        sw $s0,0($sp)
        addi $sp,$sp,4
        sw $s1,0($sp)
        addi $sp,$sp,4
        sw $s2,0($sp)
        addi $sp,$sp,4
        sw $t4,0($sp)
        addi $sp,$sp,4
        sw $s3,0($sp)
#-----------------------------------------
```

```
#-----------------------------------------------------
# thu tuc
#-----------------------------------------------------
get_cod:        #chuyen dia chi
        li $t1, IN_ADRESS_HEXA_KEYBOARD         #dia chi vao
        li $t2, OUT_ADRESS_HEXA_KEYBOARD        #dia chi ra
scan_row1:      #nhan tin hieu dong 1
        li $t3, 0x81
        sb $t3, 0($t1)
        lbu $a0, 0($t2)         #dia chi ra
        bnez $a0, get_code_in_char
scan_row2:      #nhan tin hieu dong 2
        li $t3, 0x82
        sb $t3, 0($t1)
        lbu $a0, 0($t2)         #dia chi ra
        bnez $a0, get_code_in_char
scan_row3:      #nhan tin hieu dong 3
        li $t3, 0x84
        sb $t3, 0($t1)
        lbu $a0, 0($t2)         #dia chi ra
        bnez $a0, get_code_in_char
scan_row4:      #nhan tin hieu dong 4
        li $t3, 0x88
        sb $t3, 0($t1)
        lbu $a0, 0($t2)         #dia chi ra
        bnez $a0, get_code_in_char
```

```
get_code_in_char:
        beq $a0, KEY_0, case_0
        beq $a0, KEY_1, case_1
        beq $a0, KEY_2, case_2
        beq $a0, KEY_3, case_3
        beq $a0, KEY_4, case_4
        beq $a0, KEY_5, case_5
        beq $a0, KEY_6, case_6
        beq $a0, KEY_7, case_7
        beq $a0, KEY_8, case_8
        beq $a0, KEY_9, case_9
        beq $a0, KEY_a, case_a
        beq $a0, KEY_b, case_b
        beq $a0, KEY_c, case_c
        beq $a0, KEY_d, case_d
        beq $a0, KEY_e, case_e
        beq $a0, KEY_f, case_f
```

```
        #luu vao s0 nhung kieu cha
case_0: li $s0, '0'
        j store_code
case_1: li $s0, '1'
        j store_code
case_2: li $s0, '2'
        j store_code
case_3: li $s0, '3'
        j store_code
case_4: li $s0, '4'
        j store_code
case_5: li $s0, '5'
        j store_code
case_6: li $s0, '6'
        j store_code
case_7: li $s0, '7'
        j store_code
case_8: li $s0, '8'
        j store_code
case_9: li $s0, '9'
        j store_code
case_a: li $s0, 'a'
        j store_code
case_b: li $s0, 'b'
        j store_code
case_c: li $s0, 'c'
        j store_code
case_d: li $s0, 'd'
        j store_code
case_e: li $s0, 'e'
        j store_code
case_f: li $s0, 'f'
        j store_code
```

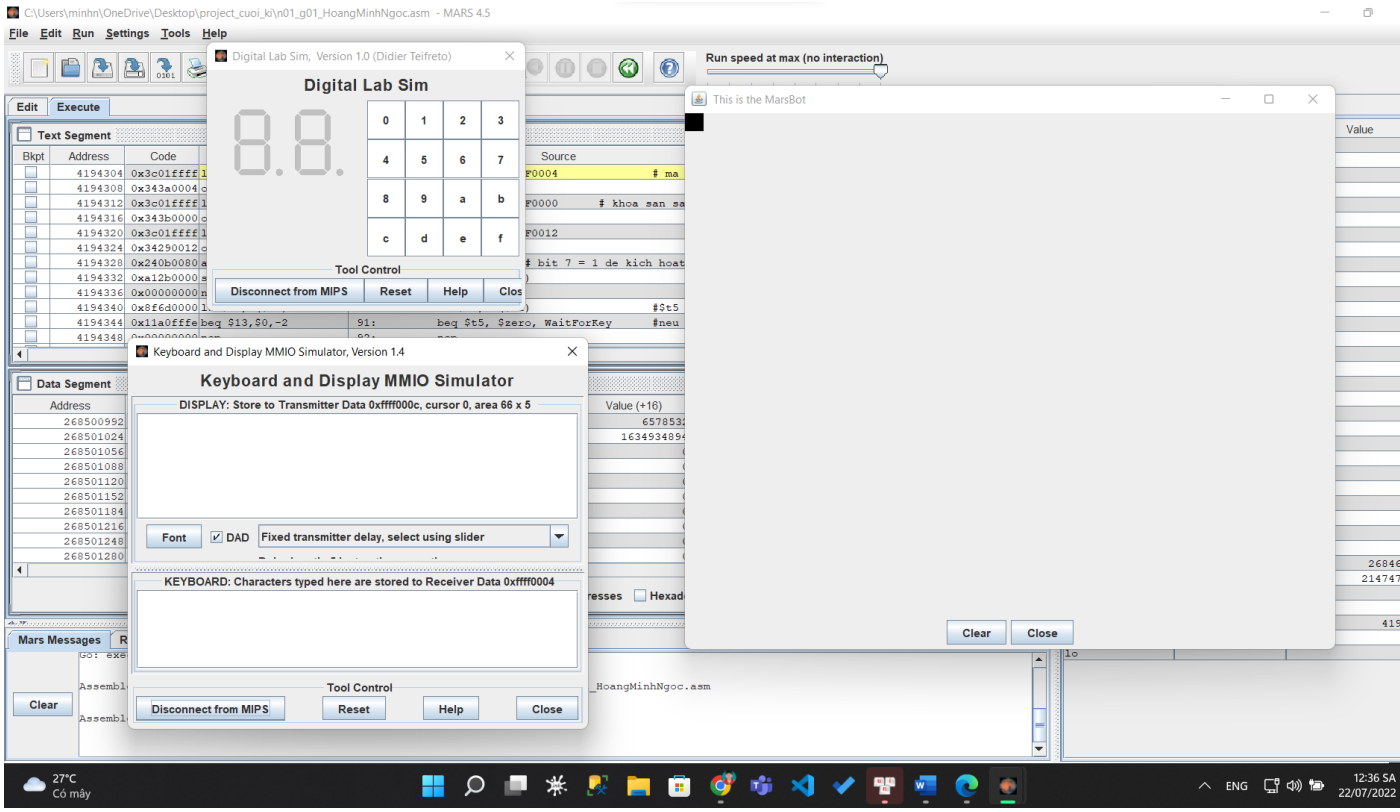
```
store_code:
        la $s1, inputMaDieuKhien
        la $s2, lengthMaDieuKhien
        lw $s3, 0($s2)                  #$s3 = so ki tu trong mang inputMaDieuKhien
        addi $t4, $t4, -1               #$t4 = i
        for_loop_to_store_code:
                addi $t4, $t4, 1
                bne $t4, $s3, for_loop_to_store_code
                add $s1, $s1, $t4       #$s1 = inputMaDieuKhien + i
                sb  $s0, 0($s1)         #inputMaDieuKhien[i] = $s0

                addi $s0, $zero, '\n'   #them ky tu '\n' vao cuoi chuoi
                addi $s1, $s1, 1        #them ky tu '\n' vao cuoi chuoi
                sb  $s0, 0($s1)         #them ky tu '\n' vao cuoi chuoi


                addi $s3, $s3, 1
                sw $s3, 0($s2)          #cap nhat do dai cua InputMaDieuKhien
```

```
#-------------------------------------------------------------
# Dang gia tra ve dia chi cua main routine
# epc <= epc + 4
#-------------------------------------------------------------
next_pc:
        mfc0 $at, $14               # $at <= Coproc0.$14 = Coproc0.epc
        addi $at, $at, 4            # $at = $at + 4 (next instruction)
        mtc0 $at, $14               # Coproc0.$14 = Coproc0.epc <= $at
#-------------------------------------------------------------
```

```
#-----------------------------------------------
# khoi phuc tap tin trong stack
#-----------------------------------------------
restore:
        lw $s3, 0($sp)
        addi $sp,$sp,-4
        lw $t4, 0($sp)
        addi $sp,$sp,-4
        lw $s2, 0($sp)
        addi $sp,$sp,-4
        lw $s1, 0($sp)
        addi $sp,$sp,-4
        lw $s0, 0($sp)
        addi $sp,$sp,-4
        lw $at, 0($sp)
        addi $sp,$sp,-4
        lw $a0, 0($sp)
        addi $sp,$sp,-4
        lw $t3, 0($sp)
        addi $sp,$sp,-4
        lw $t2, 0($sp)
        addi $sp,$sp,-4
        lw $t1, 0($sp)
        addi $sp,$sp,-4
        lw $ra, 0($sp)
        addi $sp,$sp,-4
return: eret #tra lai ngoai le return Exception
```

DEMO

Khởi động



Video demo link : https://youtu.be/8lyCwBKIq6Q

Do mấy cái lỗi định dạng nên em xin phép được up qua link sau ạ

Mã QR để xem video demo

SCAN ME

Code trong MIPS (MARS 4.5)

```
#Truong Cong Nghe Thong Tin & Truyen Thong - Dai Hoc Bach Khoa Ha Noi
#Bo mon Ky thuat may tinh
#IT3280 - Thuc hanh kien truc may tinh
#Hoang Minh Ngoc 20200440 - Ma lop 131001
#Curiosity Marsbot

.eqv IN_ADRESS_HEXA_KEYBOARD 0xFFFF0012
.eqv OUT_ADRESS_HEXA_KEYBOARD 0xFFFF0014
.eqv KEY_CODE 0xFFFF0004    # ma ASCII tu ban phim, 1 byte
.eqv KEY_READY 0xFFFF0000   # =1 neu co ma khoa moi
                # tu dong xoa sau lw
#-------------------------------------------------------------------------
--
# Marsbot
.eqv HEADING 0xffff8010 # so nguyen: tu 0 den 359 do
            # 0 : Di len ( Bac )
            # 90: Sang phai ( Dong )
            # 180: Di xuong ( Nam )
            # 270: Sang Trai ( Tay )
.eqv MOVING 0xffff8050 # Boolean cho lenh kich hoat thuc thi
.eqv LEAVETRACK 0xffff8020 # Boolean (0 or non-0):
            # whether or not to leave a track
.eqv WHEREX 0xffff8030 # Integer: vi tri x cua MarsBot
.eqv WHEREY 0xffff8040 # Integer: vi tri y cua MarsBot

#=========================================================================
==
#=========================================================================
==
.data
# gia tri nhan vao
#0-3
    .eqv KEY_0 0x11
    .eqv KEY_1 0x21
    .eqv KEY_2 0x41
    .eqv KEY_3 0x81
#4-7
    .eqv KEY_4 0x12
    .eqv KEY_5 0x22
    .eqv KEY_6 0x42
    .eqv KEY_7 0x82
#8-b
    .eqv KEY_8 0x14
    .eqv KEY_9 0x24
    .eqv KEY_a 0x44
    .eqv KEY_b 0x84
```

```
#c-f
    .eqv KEY_c 0x18
    .eqv KEY_d 0x28
    .eqv KEY_e 0x48
    .eqv KEY_f 0x88
#-------------------------------------------------------------------------
--
#tao cac lenh chuc nang
    MOVE_CODE: .asciiz "1b4"        # di chuyen
    STOP_CODE: .asciiz "c68"        # dung im
    GO_LEFT_CODE: .asciiz "444" # re trai 90 do
    GO_RIGHT_CODE: .asciiz "666"    # re phai 90 do
    TRACK_CODE: .asciiz "dad"   # ve
    UNTRACK_CODE: .asciiz "cbc" # ko ve
    GO_BACK_CODE: .asciiz "999" # quay tro lai, ko ve. ko nhan ma toi khi het
    WRONG_CODE: .asciiz "Da nhap ma dieu khien sai !"
#-------------------------------------------------------------------------
--
    inputMaDieuKhien: .space 50 # dau vao ma dieu khien
    lengthMaDieuKhien: .word 0  # ma khiem soat do dai
    nowHeading: .word 0
#-----------------------------------------------------------
# duong di cua masbot duoc luu tru vao mang lichsu
# moi 1 canh duoc luu tru duoi dang 1 cau truc
# 1 cau truc co dang {x, y, z}
# trong do:     x, y la toa do diem dau tien cua canh
#       z la huong cua canh do
# mac dinh: cau truc dau tien se la {0,0,0}
# do dai duong di ngay khi bat dau la 12 bytes (3x 4byte)
#-----------------------------------------------------------
    lichsu: .space 600
    lengthlichsu: .word 12      #bytes

#=========================================================================
==
#=========================================================================
==
.text
main:
    li $k0, KEY_CODE        # ma khoa
    li $k1, KEY_READY   # khoa san sang
#-----------------------------------------------------------
# ngat ma tran ban phim 4x4 trong Digital Lab Sim
#-----------------------------------------------------------
    li $t1, IN_ADRESS_HEXA_KEYBOARD
    li $t3, 0x80 # bit 7 = 1 de kich hoat
    sb $t3, 0($t1)
#-----------------------------------------------------------
```

```
loop:       nop
WaitForKey:
    lw $t5, 0($k1)          #$t5 = [$k1] = khoa san sang
    beq $t5, $zero, WaitForKey  #neu $t5 == 0 quay lai WaitForKey cho tin hieu
    nop
    beq $t5, $zero, WaitForKey  #neu $t5 == 0 quay lai WaitForKey cho tin hieu
ReadKey:
    lw $t6, 0($k0)          #$t6 = [$k0] = KEY_CODE
    beq $t6, 127 , continue     #if $t6 == Del thi xoa toan bo dau vao
                    #127 la ma Delete xong ASCII
    bne $t6, '\n' , loop        #neu t6 = \n thi quay lai loop de lay code
dieu khien
    nop
    bne $t6, '\n' , loop
CheckMaDieuKhien:   #kiem tra ma
    la $s2, lengthMaDieuKhien   #ma kiem soat do dai
    lw $s2, 0($s2)
    #----------------
    bne $s2, 3, pushErrorMess   #thong bao ma dieu khien dua vao la sai

    la $s3, MOVE_CODE       #lenh di chuyen
    jal isEqualString       #kiem tra chuoi
    beq $t0, 1, go

    la $s3, STOP_CODE       #lenh dung
    jal isEqualString       #kiem tra chuoi
    beq $t0, 1, stop

    la $s3, GO_LEFT_CODE        #lenh re trai
    jal isEqualString       #kiem tra chuoi
    beq $t0, 1, goLeft

    la $s3, GO_RIGHT_CODE       #lenh re phai
    jal isEqualString       #kiem tra chuoi
    beq $t0, 1, goRight

    la $s3, TRACK_CODE      #lenh ve
    jal isEqualString       #kiem tra chuoi
    beq $t0, 1, track

    la $s3, UNTRACK_CODE        #lenh khong ve
    jal isEqualString       #kiem tra chuoi
    beq $t0, 1, untrack

    la $s3, GO_BACK_CODE        #lenh quay lai
    jal isEqualString       #kiem tra chuoi
    beq $t0, 1, goBack
```

```
    beq $t0, 0, pushErrorMess   #ma dieu khien dua vao la sai

printMaDieuKhien:   # in ra ma dieu khien tren Mars Messages
    li $v0, 4
    la $a0, inputMaDieuKhien        #dau vao cua code
    syscall
    nop

continue:           #tiep theo
    jal removeMaDieuKhien   #Loai bo chuoi InputMaDieuKhien
    nop
    j loop          #quay lai nhan key moi
    nop
    j loop
#-------------------------------------------------------------
# thu tuc storelichsu, luu tru duong dan cua MarsBot den bien duong dan
# param[in]     nowHeading variable
#       lengthlichsu variable
#-------------------------------------------------------------
storelichsu:
    #luu lai tung ki tu vao cac bien
    #sp tro toi dinh cua stack
    addi $sp,$sp,4
    sw $t1, 0($sp)
    addi $sp,$sp,4
    sw $t2, 0($sp)
    addi $sp,$sp,4
    sw $t3, 0($sp)
    addi $sp,$sp,4
    sw $t4, 0($sp)

    addi $sp,$sp,4
    sw $s1, 0($sp)
    addi $sp,$sp,4
    sw $s2, 0($sp)
    addi $sp,$sp,4
    sw $s3, 0($sp)
    addi $sp,$sp,4
    sw $s4, 0($sp)

    #thu tuc
    li $t1, WHEREX
    lw $s1, 0($t1)      #s1 = x
    li $t2, WHEREY
    lw $s2, 0($t2)      #s2 = y

    la $s4, nowHeading
    lw $s4, 0($s4)      #s4 = now heading
```

```
    la $t3, lengthlichsu
    lw $s3, 0($t3)      #$s3 = lengthlichsu (dv: byte)

    la $t4, lichsu
    add $t4, $t4, $s3   #vi tri de luu tru vao mang lichsu
    #chuyen de lieu tu thanh ghi ra t4 ( 3 ki tu dieu khien )
    sw $s1, 0($t4)      #store x
    sw $s2, 4($t4)      #store y
    sw $s4, 8($t4)      #store heading

    addi $s3, $s3, 12   #cap nhat lengthlichsu
                #12 = 3 (word) x 4 (bytes)
    sw $s3, 0($t3)

    #khoi phuc lai gia tri trong stack
    lw $s4, 0($sp)
    addi $sp,$sp,-4
    lw $s3, 0($sp)
    addi $sp,$sp,-4
    lw $s2, 0($sp)
    addi $sp,$sp,-4
    lw $s1, 0($sp)
    addi $sp,$sp,-4
    lw $t4, 0($sp)
    addi $sp,$sp,-4
    lw $t3, 0($sp)
    addi $sp,$sp,-4
    lw $t2, 0($sp)
    addi $sp,$sp,-4
    lw $t1, 0($sp)
    addi $sp,$sp,-4

    jr $ra
    nop
    jr $ra
#-------------------------------------------------------------
# thu tuc goBack, dieu khien MarsBot quay lai
# param[in]     lichsu array, lengthlichsu array
#-------------------------------------------------------------
goBack: la $s7, lichsu      # ma tran
    la $s5, lengthlichsu    # byte toi da
    lw $s5, 0($s5)
    add $s7, $s7, $s5
begin:  addi $s5, $s5, -12  #lui lai 1 cau truc
    addi $s7, $s7, -12  #vi tri cua thong tin ve canh cuoi cung
    lw $s6, 8($s7)      #huong cua canh cuoi cung
    addi $s6, $s6, 180  #nguoc lai huong cua canh cuoi cung
```

```
    la $t8, nowHeading  #marsbot quay nguoc lai
    sw $s6, 0($t8)
    jal ROTATE

go_to_first_point_of_edge:  #toi diem dau cua canh
    lw $t9, 0($s7)      #toa do x cua diem dau tien cua canh
    li $t8, WHEREX      #toa do x hien tai
    lw $t8, 0($t8)

    bne $t8, $t9, go_to_first_point_of_edge

    lw $t9, 4($s7)      #toa do y cua diem dau tien cua canh
    li $t8, WHEREY      #toa do y hien tai
    lw $t8, 0($t8)

    bne $t8, $t9, go_to_first_point_of_edge

    beq $s5, 0, finish

    j begin

finish: jal STOP
    la $t8, nowHeading
    add $s6, $zero, $zero
    sw $s6, 0($t8)      #cap nhat heading
    la $t8, lengthlichsu
    sw $s5, 0($t8)      #cap nhat lengthlichsu = 0
    jal ROTATE
    j printMaDieuKhien
#-------------------------------------------------------------
# thu tuc track ,dieu khien MarsBot de theo doi va in ma dieu khien
# param[in] none
#-------------------------------------------------------------
track:  jal TRACK
    j printMaDieuKhien
#-------------------------------------------------------------
# quy trinh untrack, bo theo doi,
#dieu khien MarsBot de bo theo doi va in ma dieu khien
#-------------------------------------------------------------
untrack: jal UNTRACK
    j printMaDieuKhien
#-------------------------------------------------------------
# thu tuc go, dieu khien MarsBot de di va in ra ma dieu khien
# param[in] none
#-------------------------------------------------------------
go:     jal GO
    j printMaDieuKhien
```

```
#-------------------------------------------------------------
# thu tuc stop, dieu khien MarsBot dung va in ra ma dieu khien
# param[in] none
#-------------------------------------------------------------
stop:   jal STOP
    j printMaDieuKhien
#-------------------------------------------------------------
# thu tuc goRight , dieu khien MarsBot sang trai va in ma dieu khien
# param[in] none
#-------------------------------------------------------------
goRight:la $s5, nowHeading
    lw $s6, 0($s5)      # $s6 = $s5 = n?Heading
    addi $s6, $s6, 90   # ( tang len 90 )
    sw $s6, 0($s5)      # cap nhat nowHeading
    jal storelichsu     # luu tru duong dan cua MarsBot vao stack
    jal ROTATE      # dieu khien robot xoay
    j printMaDieuKhien  # in ra ma dieu khien
#-------------------------------------------------------------
# thu tuc goLeft,dieu khien MarsBot sang phai va in ma dieu khien
# param[in] none
#-------------------------------------------------------------
goLeft: la $s5, nowHeading
    lw $s6, 0($s5)      # $6 = nowHeading
    addi $s6, $s6, -90  # cong s6 them -90, xoay phai 90 do
    sw $s6, 0($s5)      # cap nhat nowHeading
    jal storelichsu     # luu tru duong dan cua MarsBot vao stack
    jal ROTATE      # dieu khien robot xoay
    j printMaDieuKhien  # in ra ma dieu khien
#-------------------------------------------------------------
# thu tuc removeMaDieuKhien , loai bo chuoi ma dieu kien dau vao
#               inputMaDieuKhien = ""
# param[in] none
#-------------------------------------------------------------
removeMaDieuKhien:
    #sao luu vao stack
    addi $sp,$sp,4
    sw $t1, 0($sp)
    addi $sp,$sp,4
    sw $t2, 0($sp)
    addi $sp,$sp,4
    sw $s1, 0($sp)
    addi $sp,$sp,4
    sw $t3, 0($sp)
    addi $sp,$sp,4
    sw $s2, 0($sp)

    #thu tuc
    la $s2, lengthMaDieuKhien
```

```
    lw $t3, 0($s2)              #$t3 = lengthMaDieuKhien
    addi $t1, $zero, -1         #$t1 = -1 = i
    addi $t2, $zero, 0          #$t2 = '\0'
    la $s1, inputMaDieuKhien
    addi $s1, $s1, -1
for_loop_to_remove: addi $t1, $t1, 1        #i++
    add $s1, $s1, 1             #$s1 = inputMaDieuKhien + i
    sb $t2, 0($s1)              #inputMaDieuKhien[i] = '\0'

    bne $t1, $t3, for_loop_to_remove        #if $t1 <=3 continue loop
    nop
    bne $t1, $t3, for_loop_to_remove

    add $t3, $zero, $zero
    sw $t3, 0($s2)              #lengthMaDieuKhien = 0

    #khoi phuc
    lw $s2, 0($sp)
    addi $sp,$sp,-4
    lw $t3, 0($sp)
    addi $sp,$sp,-4
    lw $s1, 0($sp)
    addi $sp,$sp,-4
    lw $t2, 0($sp)
    addi $sp,$sp,-4
    lw $t1, 0($sp)
    addi $sp,$sp,-4

    jr $ra
    nop
    jr $ra
#-------------------------------------------------------------
# thu tuc isEqualString , kiem tra inputMaDieuKhien
#               kiem tra co bang voi chuoi s ( luu trong $s3)
#               Do dai 2 chuoi la nhu nhau
# param[in] $s3, di chi la 1 chuoi
# param[out] $t0,1 neu bang nhau, 0 neu ko bang nhau
#-------------------------------------------------------------
-
isEqualString:
    #sao luu vao stack
    addi $sp,$sp,4
    sw $t1, 0($sp)
    addi $sp,$sp,4
    sw $s1, 0($sp)
    addi $sp,$sp,4
    sw $t2, 0($sp)
    addi $sp,$sp,4
```

```
    sw $t3, 0($sp)

    #thu tuc
    addi $t1, $zero, -1         #$t1 = -1 = i
    add $t0, $zero, $zero
    la $s1, inputMaDieuKhien            #$s1 = inputMaDieuKhien
for_loop_to_check_equal: addi $t1, $t1, 1           #i++

    add $t2, $s1, $t1           #$t2 = inputMaDieuKhien + i
    lb $t2, 0($t2)              #$t2 = inputMaDieuKhien[i]

    add $t3, $s3, $t1           #$t3 = s + i
    lb $t3, 0($t3)              #$t3 = s[i]

    bne $t2, $t3, isNotEqual            #if $t2 != $t3 -> isNotequal

    bne $t1, 2, for_loop_to_check_equal #if $t1 <=2 tiep tuc toi loop
    nop
    bne $t1, 2, for_loop_to_check_equal
isEqual:
    #sao luu vao stack
    lw $t3, 0($sp)
    addi $sp,$sp,-4
    lw $t2, 0($sp)
    addi $sp,$sp,-4
    lw $s1, 0($sp)
    addi $sp,$sp,-4
    lw $t1, 0($sp)
    addi $sp,$sp,-4

    add $t0, $zero, 1   #cap nhat $t0
    jr $ra
    nop
    jr $ra
isNotEqual:
    #restore
    lw $t3, 0($sp)
    addi $sp,$sp,-4
    lw $t2, 0($sp)
    addi $sp,$sp,-4
    lw $s1, 0($sp)
    addi $sp,$sp,-4
    lw $t1, 0($sp)
    addi $sp,$sp,-4

    add $t0, $zero, $zero   #cap nhat $t0
    jr $ra
    nop
```

```
    jr $ra
#-------------------------------------------------------------
# thu tuc pushErrorMess , thong bao ma bi sai
# param[in] none
#-------------------------------------------------------------
-
pushErrorMess: li $v0, 4
    la $a0, inputMaDieuKhien
    syscall
    nop

    li $v0, 55
    la $a0, WRONG_CODE
    syscall
    nop
    nop
    j continue
    nop
    j continue
#-------------------------------------------------------------
# thu tuc GO , bat dau chay
# param[in] none
#-------------------------------------------------------------
GO:     #sao luu vao stack
    addi $sp,$sp,4
    sw $at,0($sp)
    addi $sp,$sp,4
    sw $k0,0($sp)
    #thu tuc
    li $at, MOVING      # lenh kich hoat duoc thuc thi
    addi $k0, $zero,1   # to logic 1,
    sb $k0, 0($at)      # bat dau chay
    #khoi phuc
    lw $k0, 0($sp)
    addi $sp,$sp,-4
    lw $at, 0($sp)
    addi $sp,$sp,-4

    jr $ra
    nop
    jr $ra
#-------------------------------------------------------------
# thu tuc STOP , dung chay
# param[in] none
#-------------------------------------------------------------
STOP:   #sao luu
    addi $sp,$sp,4
    sw $at,0($sp)
```

```
    #thu tuc
    li $at, MOVING      # thay doi sang MOVING
    sb $zero, 0($at)        # dung lai
    #khoi phuc
    lw $at, 0($sp)
    addi $sp,$sp,-4

    jr $ra
    nop
    jr $ra
#------------------------------------------------------------
# thu tuc TRACK , ve duong
# param[in] none
#------------------------------------------------------------
TRACK:  #sao luu
    addi $sp,$sp,4
    sw $at,0($sp)
    addi $sp,$sp,4
    sw $k0,0($sp)
    #thu tuc
    li $at, LEAVETRACK   # thay doi sang LEAVETRACK
    addi $k0, $zero,1    # to logic 1,
    sb $k0, 0($at)       # to start tracking
    #khoi phuc
    lw $k0, 0($sp)
    addi $sp,$sp,-4
    lw $at, 0($sp)
    addi $sp,$sp,-4

    jr $ra
    nop
    jr $ra
#------------------------------------------------------------
# thu tuc UNTRACK , dung ve duong
# param[in] none
#------------------------------------------------------------
UNTRACK:
    #sao luu
    addi $sp,$sp,4
    sw $at,0($sp)
    #thu tuc
    li $at, LEAVETRACK  # thay doi sang cong LEAVETRACK va cho = 0
    sb $zero, 0($at)    # dung ve
    #khoi phuc
    lw $at, 0($sp)
    addi $sp,$sp,-4

    jr $ra
```

```
    nop
    jr $ra
#-------------------------------------------------------------
# thu tuc ROTATE_RIGHT , dieu khien robot xoay
# param[in] nowHeading variable, store heading at present
#-------------------------------------------------------------
ROTATE:
    #sao luu
    addi $sp,$sp,4
    sw $t1,0($sp)
    addi $sp,$sp,4
    sw $t2,0($sp)
    addi $sp,$sp,4
    sw $t3,0($sp)
    #thu tuc
    li $t1, HEADING     # thay doi xong sang HEADING
    la $t2, nowHeading
    lw $t3, 0($t2)      # t3 = nowHeading
    sw $t3, 0($t1)      # xoay robot
    #khoi phuc
    lw $t3, 0($sp)
    addi $sp,$sp,-4
    lw $t2, 0($sp)
    addi $sp,$sp,-4
    lw $t1, 0($sp)
    addi $sp,$sp,-4

    jr $ra
    nop
    jr $ra
#-------------------------------------------------------------
# thu tuc ROTATE_LEFT , xoay robot sang ben phai
# param[in] nowHeading variable, store heading at present
#-------------------------------------------------------------
ROTATE_LEFT:
    #sao luu
    addi $sp,$sp,4
    sw $t1,0($sp)
    addi $sp,$sp,4
    sw $t2,0($sp)
    addi $sp,$sp,4
    sw $t3,0($sp)
    #thu tuc
    li $t1, HEADING         # thay doi cong HEADING
    la $t2, nowHeading
    lw $t3, 0($t1)      # $t2 = HEADING
    addi $t3, $t3, -90  # xoay sang ben phai nen -90
    sw $t3, 0($t2)      # cap nhat nowHeading
```

```
    sw $t3, 0($t1)      # xoay robot
    #khoi phuc
    lw $t3, 0($sp)
    addi $sp,$sp,-4
    lw $t2, 0($sp)
    addi $sp,$sp,-4
    lw $t1, 0($sp)
    addi $sp,$sp,-4

    jr $ra
    nop
    jr $ra
#================================================================================
==
# GENERAL INTERRUPT SERVED ROUTINE cho tat ca interrupts
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~
.ktext 0x80000180
#---------------------------------------------------------
# luu cac tap tin vao stack
#---------------------------------------------------------
backup:     #sao luu tu thanh ghi vao stack
    addi $sp,$sp,4
    sw $ra,0($sp)
    addi $sp,$sp,4
    sw $t1,0($sp)
    addi $sp,$sp,4
    sw $t2,0($sp)
    addi $sp,$sp,4
    sw $t3,0($sp)
    addi $sp,$sp,4
    sw $a0,0($sp)
    addi $sp,$sp,4
    sw $at,0($sp)
    addi $sp,$sp,4
    sw $s0,0($sp)
    addi $sp,$sp,4
    sw $s1,0($sp)
    addi $sp,$sp,4
    sw $s2,0($sp)
    addi $sp,$sp,4
    sw $t4,0($sp)
    addi $sp,$sp,4
    sw $s3,0($sp)
#---------------------------------------------------------
# thu tuc
#---------------------------------------------------------
get_cod:        #chuyen dia chi
```

```
    li $t1, IN_ADRESS_HEXA_KEYBOARD     #dia chi vao
    li $t2, OUT_ADRESS_HEXA_KEYBOARD        #dia chi ra
scan_row1:  #nhan tin hieu dong 1
    li $t3, 0x81
    sb $t3, 0($t1)
    lbu $a0, 0($t2)     #dia chi ra
    bnez $a0, get_code_in_char
scan_row2:  #nhan tin hieu dong 2
    li $t3, 0x82
    sb $t3, 0($t1)
    lbu $a0, 0($t2)     #dia chi ra
    bnez $a0, get_code_in_char
scan_row3:  #nhan tin hieu dong 3
    li $t3, 0x84
    sb $t3, 0($t1)
    lbu $a0, 0($t2)     #dia chi ra
    bnez $a0, get_code_in_char
scan_row4:  #nhan tin hieu dong 4
    li $t3, 0x88
    sb $t3, 0($t1)
    lbu $a0, 0($t2)         #dia chi ra
    bnez $a0, get_code_in_char
get_code_in_char:
    beq $a0, KEY_0, case_0
    beq $a0, KEY_1, case_1
    beq $a0, KEY_2, case_2
    beq $a0, KEY_3, case_3
    beq $a0, KEY_4, case_4
    beq $a0, KEY_5, case_5
    beq $a0, KEY_6, case_6
    beq $a0, KEY_7, case_7
    beq $a0, KEY_8, case_8
    beq $a0, KEY_9, case_9
    beq $a0, KEY_a, case_a
    beq $a0, KEY_b, case_b
    beq $a0, KEY_c, case_c
    beq $a0, KEY_d, case_d
    beq $a0, KEY_e, case_e
    beq $a0, KEY_f, case_f

    #luu vao s0 nhung kieu char
case_0: li $s0, '0'
    j store_code
case_1: li $s0, '1'
    j store_code
case_2: li $s0, '2'
    j store_code
case_3: li $s0, '3'
```

```
    j store_code
case_4: li $s0, '4'
    j store_code
case_5: li $s0, '5'
    j store_code
case_6: li $s0, '6'
    j store_code
case_7: li $s0, '7'
    j store_code
case_8: li $s0, '8'
    j store_code
case_9: li $s0, '9'
    j store_code
case_a: li $s0, 'a'
    j store_code
case_b: li $s0, 'b'
    j store_code
case_c: li $s0, 'c'
    j store_code
case_d: li $s0, 'd'
    j store_code
case_e: li $s0, 'e'
    j store_code
case_f: li $s0, 'f'
    j store_code
store_code:
    la $s1, inputMaDieuKhien
    la $s2, lengthMaDieuKhien
    lw $s3, 0($s2)          #$s3 = so ki tu trong mang inputMaDieuKhien
    addi $t4, $t4, -1       #$t4 = i
    for_loop_to_store_code:
        addi $t4, $t4, 1
        bne $t4, $s3, for_loop_to_store_code
        add $s1, $s1, $t4   #$s1 = inputMaDieuKhien + i
        sb  $s0, 0($s1)     #inputMaDieuKhien[i] = $s0

        addi $s0, $zero, '\n'   #them ky tu '\n' vao cuoi chuoi
        addi $s1, $s1, 1        #them ky tu '\n' vao cuoi chuoi
        sb  $s0, 0($s1)     #them ky tu '\n' vao cuoi chuoi


        addi $s3, $s3, 1
        sw $s3, 0($s2)      #cap nhat do dai cua InputMaDieuKhien

#-------------------------------------------------------
# Dang gia tra ve dia chi cua main routine
# epc <= epc + 4
#-------------------------------------------------------
```

```
next_pc:
    mfc0 $at, $14       # $at <= Coproc0.$14 = Coproc0.epc
    addi $at, $at, 4        # $at = $at + 4 (next instruction)
    mtc0 $at, $14       # Coproc0.$14 = Coproc0.epc <= $at
#-----------------------------------------------------------
# khoi phuc tap tin trong stack
#-----------------------------------------------------------
restore:
    lw $s3, 0($sp)
    addi $sp,$sp,-4
    lw $t4, 0($sp)
    addi $sp,$sp,-4
    lw $s2, 0($sp)
    addi $sp,$sp,-4
    lw $s1, 0($sp)
    addi $sp,$sp,-4
    lw $s0, 0($sp)
    addi $sp,$sp,-4
    lw $at, 0($sp)
    addi $sp,$sp,-4
    lw $a0, 0($sp)
    addi $sp,$sp,-4
    lw $t3, 0($sp)
    addi $sp,$sp,-4
    lw $t2, 0($sp)
    addi $sp,$sp,-4
    lw $t1, 0($sp)
    addi $sp,$sp,-4
    lw $ra, 0($sp)
    addi $sp,$sp,-4
return: eret #tra lai ngoai le return Exception
```

Bài 2:

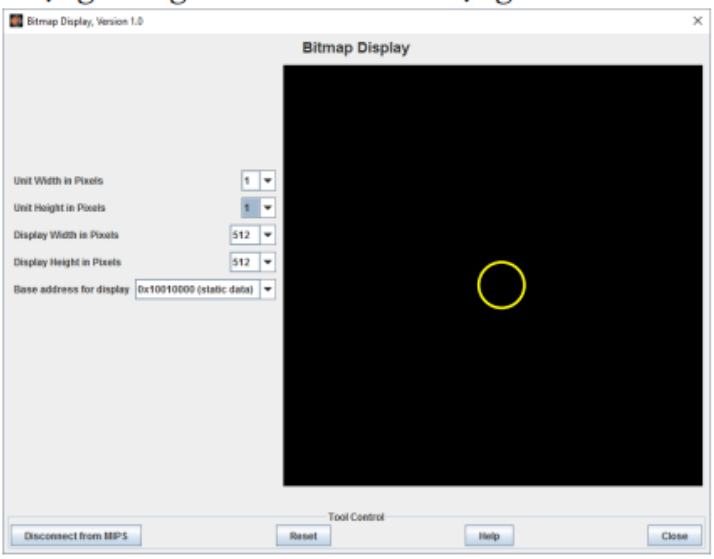
Viết một chương trình sử dụng MIPS để vẽ một quả bóng di chuyển trên màn hình mô phỏng Bitmap của Mars). Nếu đối tượng đập vào cạnh của màn hình thì sẽ di chuyển theo chiều ngược lại.

Yêu cầu

- Thiết lập màn hình ở kích thước 512x512. Kích thước 1 pixel 1x1.
- Quả bóng là một đường tròn

Chiều di chuyển phụ thuộc vào phím người dùng bấm, gồm có (di chuyển lên (W), di chuyển xuống (S), Sang trái (A), Sang phải (D) trong bộ giả lập Keyboard and Display MMIO Simulator). Tốc độ bóng di chuyển là không đổi. Vị trí bóng ban đầu ở giữa màn hình.

Gợi ý: Để làm một đối tượng di chuyển thì chúng ta sẽ xóa đối tượng ở vị trí cũ và vẽ đối tượng ở vị trí mới. Để xóa đối tượng chúng ta chỉ cần vẽ đối tượng đó với màu là màu nền



1) Ý tưởng thực hiện

– sử dụng Keyboard and Display và Bitmap

Keyboard and Display	Nhận các nút điều hướng
Bitmap	Vẽ hình quả bóng di chuyển

Cài đặt các thông số mặc định

```
# Truong Cong Nghe Thong Tin & Truyen Thong - Dai Hoc Bach Khoa Ha Noi
# Bo mon Ky thuat may tinh
# IT3280 - Thuc hanh kien truc may tinh
# Hoang Minh Ngoc 20200440 - Ma lop 131001
# Ve hinh tren Bitmap
# Di chuyen qua bong tren man hinh Bitmap cua Mars
# Su dung cac phim a, s, d, w de di chuyen
# Kich thuoc man hinh bitmap 512 * 512
# Kich thuoc o vuong don vi 1 * 1
# Base_Address 0x10010000

# Bo khoi dong mac dinh
.eqv KEY_CODE 0xFFFF0004      # ASCII code to show, 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        # chua ki tu nhap vao
        li $k1, KEY_READY       # kiem tra da nhap phim nao chua
        li $s2, DISPLAY_CODE    # hien thi ky tu
        li $s1, DISPLAY_READY   # kiem tra xem man hinh da san sang hien thi chua
```

Vị trí của chấm đầu tiên, tọa độ vẽ hình

```

        addi    $s7, $0, 512                    #luu do rong man hinh vao s7
        #circle:
        addi    $a0, $0, 256            #x = 256
        addi    $a1, $0, 256            #y = 256
        addi    $a2, $0, 20             #r = 20
        addi    $s0, $0, 0x00FFFF00     # mau vang
        jal     DrawCircle
        nop
```

Bắt đầu chương trình chính

```
34. moving:                 # nhan ki tu tu ban phim va di chuyen
35.     # so sanh trong ma ASCII
36.     beq $t0,97,left         # 97 = 'a'
37.     beq $t0,100,right       # 100 = 'd'
38.     beq $t0,115,down        # 115 = 's'
39.     beq $t0,119,up          # 119 = 'w'
40.     j Input
41.
```

Xác định xem hướng di chuyển mong muốn của người đang điều khiển bằng cách so sánh kí tự ASCII

Lệnh sang trái

```
42.     left:               # di chuyen sang trai
43.         addi $s0,$0,0x00000000  # gia tri mau bang 0, mau den, xoa diem
44.         jal DrawCircle      # xoa hinh tron vi tri cu, bang cach tô đen
45.         addi $a0,$a0,-1     # hoanh do tam hinh tron moi giam di 1
46.         add $a1,$a1, $0     # tung do tam giu nguyen
47.         addi $s0,$0,0x00FFFF00  # mau vang
48.         jal DrawCircle      # ve hinh tron vi tri moi
49.         jal Pause
50.         bltu $a0,20,reboundRight # nảy sang bên phải
51.         j Input
```

Lệnh di chuyển sang trái, khi vẽ quả bóng màu vàng thì lệnh dịch trái này sẽ vẽ lại quả bóng bằng màu đen, sau đó cập nhật các tọa độ của hình tròn, sau đó lại vẽ hình tròn mới bằng màu vàng dòng 47, 48, lệnh Pause để cho quả bóng chạy đỡ bị quá nhanh, lệnh dòng 50 để xác định xem đã tới cạnh hay chưa, bằng cách nhỏ hơn bán kính của nó thì sẽ bị đảo lại

```
86.     reboundRight:
87.         li $t3 100
88.         sw $t3,0($k0)   # thay đổi giá trị ở địa chỉ KEY_CODE để nhảy đến right
89.         j Input
```

Tương tự như vậy ta có bên phải

Lệnh sang phải

```
52.     right:              # di chuyển sang phải
53.         addi $s0,$0,0x00000000  #
54.         jal DrawCircle      # xoá hình tròn cũ
55.         addi $a0,$a0,1      # hoành độ tăng 1
56.         add $a1,$a1, $0     # tung độ giữ nguyên
57.         addi $s0,$0,0x00FFFF00
58.         jal DrawCircle      # vẽ hình tròn mới
59.         jal Pause
60.         bgtu $a0,492,reboundLeft # đập thành thì nảy sang trái
61.         j Input
```

Lệnh di chuyển sang phải, khi vẽ quả bóng màu vàng thì lệnh dịch phải này sẽ vẽ lại quả bóng bằng màu đen, sau đó cập nhật các tọa độ của hình tròn, sau đó lại vẽ hình tròn mới bằng màu vàng dòng 57, 58, lệnh Pause để cho quả bóng chạy đỡ bị quá nhanh, lệnh dòng 60 để xác định xem đã tới cạnh hay chưa, bằng cách nhỏ hơn bán kính của nó thì sẽ bị đảo lại

```
86.     reboundRight:
87.         li $t3 100
88.         sw $t3,0($k0)   # thay đổi giá trị ở địa chỉ KEY_CODE để nhảy đến right
89.         j Input
```

Vậy thì cũng tương tự như phải – trái thì ta cũng có lên xuống

```
62.     up:                 # di chuyển lên trên
63.         addi $s0,$0,0x00000000
64.         jal DrawCircle      # xoá hình tròn cũ
65.         addi $a1,$a1,-1     # tung độ giảm 1
66.         add $a0,$a0,$0      # hoành độ giữ nguyên
67.         addi $s0,$0,0x00FFFF00
68.         jal DrawCircle      # vẽ hình tròn mới
69.         jal Pause
70.         bltu $a1,20,reboundDown # đập thành thì nảy xuống dưới
71.         j Input
72.     down:
73.         addi $s0,$0,0x00000000
74.         jal DrawCircle  # xoá hình tròn cũ
75.         addi $a1,$a1,1  # tung độ tăng 1
76.         add $a0,$a0,$0  # hoành độ giữ nguyên
77.         addi $s0,$0,0x00FFFF00
78.         jal DrawCircle  # ve hình tròn mới
79.         jal Pause
80.         bgtu $a1,492,reboundUp  #đập thành thì nảy lên trên
81.         j Input
```

```
90.     reboundDown:
91.         li $t3 115
92.         sw $t3,0($k0)    # thay đổi giá trị ở địa chỉ KEY_CODE để nhảy đến down
93.         j Input
94.     reboundUp:
95.         li $t3 119
96.         sw $t3,0($k0)   # thay đổi giá trị ở địa chỉ KEY_CODE để nhảy đến up
97.         j Input
```

Hàm sẵn sàng để nhận lệnh từ Keyboard and Display

```
98.  Input:
99.      ReadKey: lw $t0, 0($k0) # $t0 = [$k0] = KEY_CODE
100.     j moving
101.
```

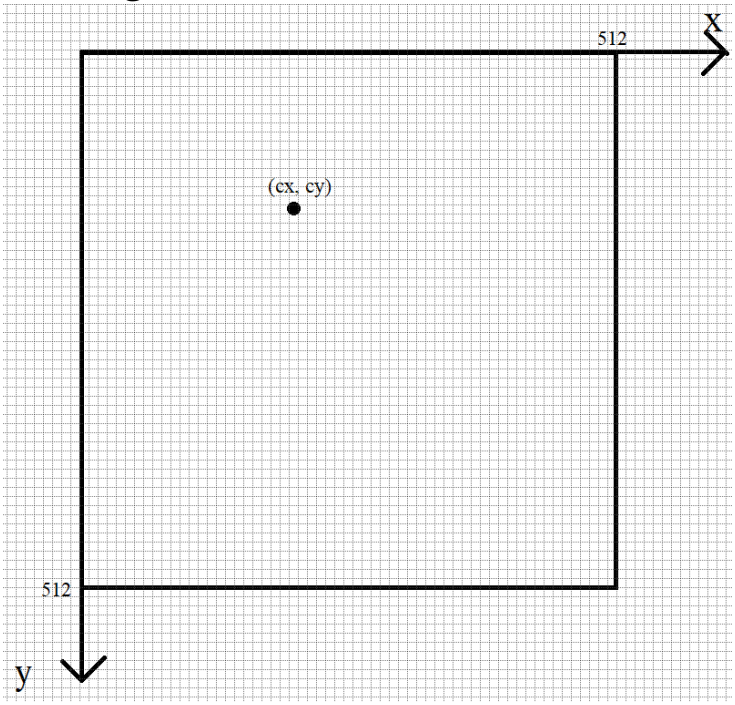
Hàm delay làm chậm lại, tốc đọ 0,05 s, nhưng em thấy hơi chậm nên là khóa nó lại

```
102. Pause:
103.     # delay ve
104.     addiu $sp,$sp,-4
105.     sw $a0, ($sp)
106.     #la $a0, 5   #system_sleep
107.     #li $v0, 32  #syscall value for sleep
108.     #syscall
109.
110.     lw $a0,($sp)
111.     addiu $sp,$sp,4
112.     jr $ra
```

1. Vẽ hình tròn bán kính trên màn hình bitmap.

Ta vẽ hình tròn bán kính 20.

Ta dùng thuật toán Bresenham để vẽ hình tròn (dòng 107-285):

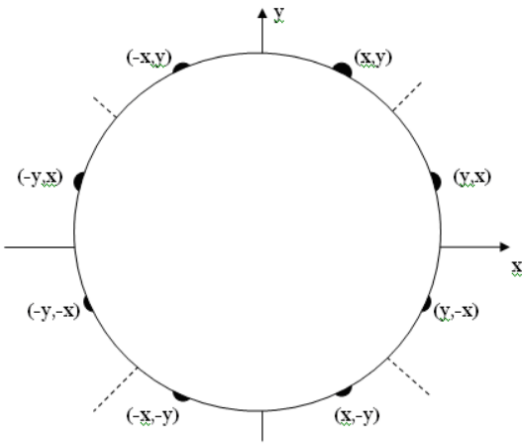


Toạ độ hoá màn hình bitmap

Ta đưa hệ toạ độ Oxy vào màn hình bitmap, gọi toạ độ tâm mỗi hình tròn cần vẽ là (cx, cy).

Toạ độ của các điểm cần được tô màu là (cx + x, cy +y).

Do tính đối xứng của đường tròn nên ta chỉ cần vẽ 1/8 đường tròn, sau đó lấy đối xứng qua 2 trục tọa độ và 2 đường phân giác thì ta vẽ được cả đường tròn.



Với mỗi giá trị x, y không âm nhận được thoả mãn, ta sẽ vẽ được 8 điểm:
(cx + x, cy + y),
(cx - x, cy + y),
(cx - x, cy - y),
(cx + x, cy - y),
(cx + y, cy + x),
(cx - y, cy + x),
(cx - y, cy - x),
(cx + y, cy - x).

Để có được các cặp giá trị (x, y), ta sử dụng thuật toán Bresenham:

- Bước 1: khởi tạo $(x_1, y_1) = (20, 0)$; $p_1 = 3 - 2R = -37$.
- Bước i+1: nếu $p_i < 0$ thì $(x_{i+1}, y_{i+1}) = (x_i, y_i + 1)$;
  $p_{i+1} = p_i + 4y_i + 6$.
  nếu $p_i < 0$ thì $(x_{i+1}, y_{i+1}) = (x_i - 1, y_i + 1)$; $p_{i+1} = p_i + 4(y_i - x_i) + 10$.

2. Giải thích và chạy code:

Đầu tiên, ta khai báo, đặt tên địa chỉ các ô nhớ (dòng 8 – dòng 10).

Đưa địa chỉ của các ô nhớ KEY_CODE, KEY_READY, DISPLAY_CODE, DISPLAY_READY vào các thanh ghi (dòng 16 – dòng 20).

Độ rộng màn hình bitmap vào thanh ghi *$s7* (dòng 21).

Sau đó ta lưu toạ độ tâm và bán kính hình tròn vào các thanh ghi để ta vẽ hình tròn đầu tiên ở tâm màn hình (dòng 23 – dòng 28).

Từ dòng 29 đến dòng 96 ta sẽ đọc ký tự từ Keyboard and Display MMIO Simulator và di chuyển hình tròn. Ban đầu chưa có kí tự nhập vào nên dòng 31, 32, 33, 34 không nhảy, dòng 35 nhảy đến *Input*, vùng lệnh *Input* ta lưu ký tự nhập vào (được lưu ở địa chỉ KEY_CODE, nếu có nhập) vào *$t0* sau đó ta quay lại *moving*. Ở vùng lệnh *moving* ta so sánh giá trị ký tự ở *$t0* với ‘a’, ‘s’, ‘d’, ‘w’. Giả sử *$t0* = ‘a’ (các giá trị ‘d’, ‘s’, ‘w’ tương tự) thì nhảy đến vùng lệnh *left*. Ở vùng lệnh *left* ta di chuyển hình tròn sang trái xoá hình tròn đã có (tô màu đen các pixel trên đường tròn, dòng 38, 39), dịch chuyển tâm của hình tròn sang trái (giảm hoành độ đi 1, giữ nguyên tung độ, dòng 40, 41) rồi vẽ hình tròn với toạ độ tâm như vậy. Nếu hình tròn đập vào thành (hoành độ = 20) thì nhảy đến vùng lên *reboundRight,* ở đây ô nhớ KEY_CODE được thay đổi giá trị là ‘d’ (dòng 83) rồi nhảy xuống *Input* , ở đây *$t0* = ‘d’ rồi nhảy về *moving* và di chuyển hình tròn sang phải (nảy sang phải). Trong một khoảng thời gian không có phím mới được

nhập, giá trị *$t0* sẽ không đổi và hình tròn sẽ di chuyển liên tục về 1 hướng và đập ngược trở lại nếu va vào thành. Nếu phím được nhập khác với 'a', 'd', 'w', 's' thì vòng lặp của khối *Input* thêm với các lệnh so sánh ở *moving* sẽ lặp liên tục cho đến khi đọc được phím chính xác, trong lúc này quả bóng sẽ đứng yên.

Vùng lệnh *DrawCircle* để vẽ hình tròn, vùng lệnh này được viết dựa trên thuật toán từ dòng 127 đến dòng 135 để đưa giá trị các thanh ghi vào trong stack để không mất giá trị các thanh ghi khi quay lại vùng lệnh trước. Dòng 137 đến 142, khởi tạo giá trị p_1= 3 – 2r, x_1 = r =20, y_1=0. Tiếp theo ta bước vào vòng lặp *DrawCircleLoop* với điều kiện lặp là x >= y (dòng 139). Dòng 149, nhảy đến vùng lệnh *plot8points* để vẽ 8 điểm từ hai số (x,y). Ở vùng lệnh *plot8points* ta nhảy đến vùng lệnh *plot4points* để vẽ 4 điểm. Dòng 221, 222, toạ độ tâm hình tròn được lưu vào *$t0*, *$t1*. Dòng 224, 225, toạ độ điểm cần tô màu được lưu vào *$a0*, *$a2*, Dòng 221 nhảy đến vùng lệnh *SetPixel* để tô màu điểm đó. Ở vùng lệnh *SetPixel*, dòng 273 lưu địa chỉ bắt đầu bộ nhớ màn hình vào *$s1*, những dòng tiếp theo tính toán địa chỉ bộ nhớ của điểm cần tô màu (toạ độ (a,b)): *$s1* = *$s1* + (512*b+a)*4. Dòng 280 lưu giá trị màu vào địa chỉ của điểm cần tô màu. Quay lại tiếp tục vùng lệnh *plot4points*, ta tô màu 3 điểm (cx - x, cy + y), (cx - x, cy - y), (cx + x, cy - y), rồi quay về tiếp tục vùng lệnh *plot8points*. Ở đây, ta đảo giá trị của x và y rồi nhảy đến *plot4point* để vẽ nốt 4 điểm còn lại. Thoát ra khỏi vùng lệnh *plot8point*, ở vùng lệnh *DrawCircleLoop* ta tính toán các điểm (x, y) tiếp theo để vẽ.

```
DrawCircle:
    # Dung thuat toan Bresenham de ve duong tron
    # Toa do tam duong tron can ve la (cx, cy)
    # Toa do diem can to mau la (cx +- x, cy +- y)
    # $a0 = cx va hoanh do pixel can to mau
    # $a2 = tung do pixel can to mau
    # s0 = colour mau vang
    # Khoi tao x = r = 20, y = 0
    # Cach tinh cac cap (x, y) tiep theo:
    # p = 3 - 2*r
    # neu p < 0 thi  p = p + 4y + 6 va (x, y) = (x, y+1)
    # neu p > 0 thi p = p + 4(y-x) +10 va (x, y) = (x-1, y+1)
    # neu y > x thi ta ve xong hinh tron, thoat khoi khối lệnh
```

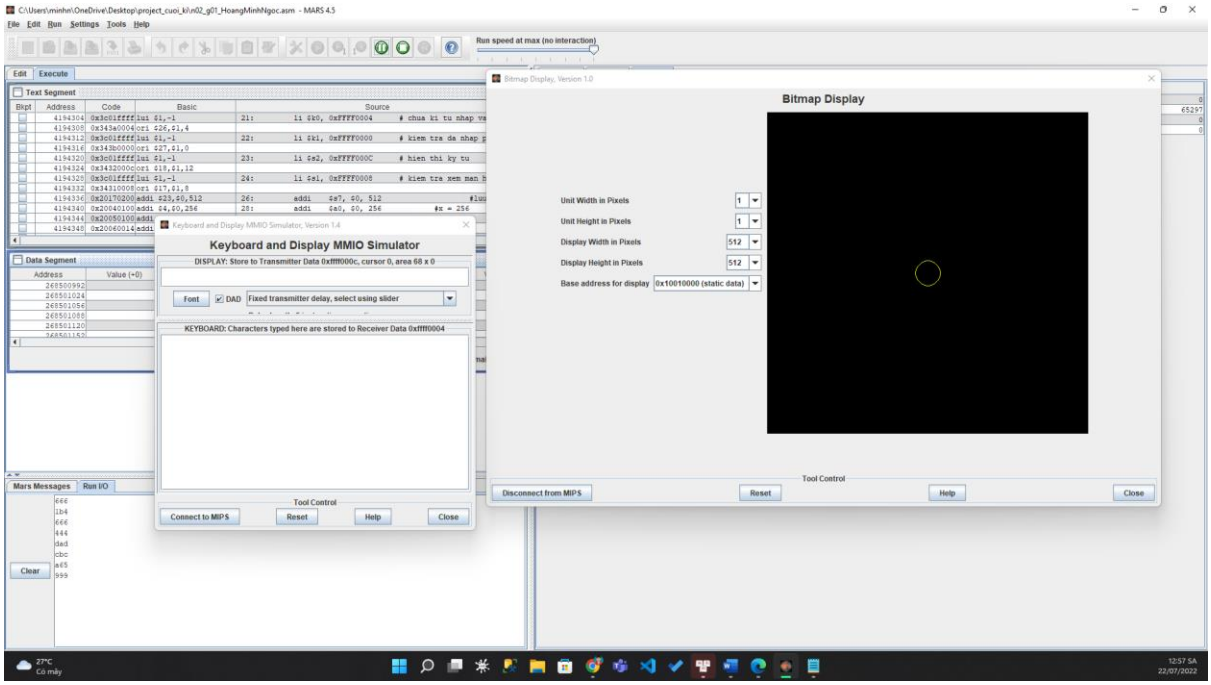
```
    # sao lưu giá trị vào Stack để lát trả lại giá trị cũ
    addiu   $sp, $sp, -32
    sw  $ra, 28($sp)
    sw  $a0, 24($sp)
    sw  $a1, 20($sp)
    sw  $a2, 16($sp)
    sw  $s4, 12($sp)
    sw  $s3, 8($sp)
    sw  $s2, 4($sp)
    sw  $s0, ($sp)

    #code goes here
    sll     $t4, $a2, 1
    li  $t3, 3
    sub $s2, $t3, $t4           # p = 3 - 2r
    add $s3, $0, $a2            #x = r = 20
    add $s4, $0, $0         #y = 0

    DrawCircleLoop:
    bgt     $s4, $s3, exitDrawCircle #neu y > x thi ta ve xong hinh tron, thoat khoi vong lap
    nop

    #ve 4 diem doi xung nhau qua truc hoanh va truc tung, roi hoan doi x, y cho nhau de ve tiep 4 diem doi xung qua phan giac
    jal plot8points
    nop
    # Tinh toan (x, y) tiep theo, roi ve cac diem moi
    sll $t3, $s4, 2             # $t3 = 4y
    add     $s2, $s2, $t3
    addi    $s2, $s2, 6         # p = p + 4y +6
    addi    $s4, $s4, 1         # y = y + 1

    blt $s2, 0, DrawCircleLoop      #if error >= 0, start loop again
    nop

    sll $t4, $s3, 2
    sub $s2, $s2, $t4
    addi    $s2, $s2, 4
    addi    $s3, $s3, -1

    j   DrawCircleLoop
    nop

    exitDrawCircle:

    lw  $s0, ($sp)
    lw  $s2, 4($sp)
    lw  $s3, 8($sp)
    lw  $s4, 12($sp)
    lw  $a2, 16($sp)
    lw  $a1, 20($sp)
    lw  $a0, 24($sp)
    lw  $ra, 28($sp)

    addiu   $sp, $sp, 32

    jr  $ra
    nop
```

```
plot8points:
    addiu   $sp, $sp, -4
    sw  $ra, 0($sp)

    jal plot4points
    nop

    beq     $s4, $s3, skipSecondplot
    nop

    # doi gia tri x va y cho nhau de ve 4 diem doi xung qua duong phan giac
    add $t2, $0, $s4            # t = y
    add $s4, $0, $s3            # y = x
    add $s3, $0, $t2            # x = t

    jal plot4points
    nop

    # doi gia tri x va y về như cũ
    add $t2, $0, $s4            # t = y
    add $s4, $0, $s3            # y = x
    add $s3, $0, $t2            # x = t

    skipSecondplot:

    lw  $ra, ($sp)
    addiu   $sp, $sp, 4

    jr  $ra
    nop
```

```
plot4points:

    addiu   $sp, $sp -4
    sw  $ra, ($sp)

    #$a0 = a0 + s3, $a2 = a1 + s4
    add $t0, $0, $a0            # $t0 luu cx
    add $t1, $0, $a1            # $t1 luu cy

    add $a0, $t0, $s3           # hoanh do diem can to mau: cx + x
    add $a2, $t1, $s4           # tung do diem can to mau: cy + y

    jal SetPixel                    # (cx + x, cy + y)
    nop

    sub $a0, $t0, $s3           #cx - x
    #add    $a2, $t1, $s4           #cy + y

    beq $s3, $0, skipXnotequal0     #if s3 (x) equals 0, skip
    nop

    jal     SetPixel                # (cx - x, cy +y)
    nop

    skipXnotequal0:
    sub $a2, $t1, $s4           #cy - y (a0 already equals cx - x)
    jal     SetPixel                    # (cx - x, cy - y)
    nop

    add $a0, $t0, $s3

    beq $s4, $0, skipYnotequal0
    nop

    jal SetPixel                #(cx + x, cy - y)
    nop

    skipYnotequal0:

    add $a0, $0, $t0
    add $a2, $0, $t1

    lw  $ra, ($sp)
    addiu   $sp, $sp, 4

    jr  $ra
    nop
```

```
    nop
SetPixel:
    #a0 x
    #a1 y
    #s0 colour
    addiu   $sp, $sp, -20           # Save return address on stack
    sw  $ra, 16($sp)
    sw  $s1, 12($sp)
    sw  $s0, 8($sp)         # Save original values of a0, s0, a2
    sw  $a0, 4($sp)
    sw  $a2, ($sp)

    lui $s1, 0x1001         # Dia chi bat dau bo nho man hinh
    sll $a0, $a0, 2
    add $s1, $s1, $a0
    mul     $a2, $a2, $s7
    mul $a2, $a2, 4
    add $s1, $s1, $a2           # Dia chi pixel can to mau

    sw  $s0, ($s1)          # Luu gia tri màu vao dia chi pixel

    lw  $a2, ($sp)          #retrieve original values and return address
    lw  $a0, 4($sp)
    lw  $s0, 8($sp)
    lw  $s1, 12($sp)
    lw  $ra, 16($sp)
    addiu   $sp, $sp, 20

    jr  $ra
    nop
```

Demo



khởi động

Video demo link: https://youtu.be/7drbVEr0TDw

QR video demo bài 2

Code trong MIPS (MARS 4.5)

```
# Truong Cong Nghe Thong Tin & Truyen Thong - Dai Hoc Bach Khoa Ha Noi
# Bo mon Ky thuat may tinh
# IT3280 - Thuc hanh kien truc may tinh
# Hoang Minh Ngoc 20200440 - Ma lop 131001
# vẽ hình trên Bitmap
# Di chuyen qua bong tren man hinh Bitmap cua Mars
# Su dung cac phim a, s, d, w de di chuyen
# Kich thuoc man hinh bitmap 512 * 512
# Kich thuoc o vuong don vi 1 * 1
# Base_Address 0x10010000

# Bo khoi dong mac dinh
.eqv KEY_CODE 0xFFFF0004        # ASCII code to show, 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    # chua ki tu nhap vao
    li $k1, KEY_READY   # kiem tra da nhap phim nao chua
    li $s2, DISPLAY_CODE    # hien thi ky tu
    li $s1, DISPLAY_READY   # kiem tra xem man hinh da san sang hien thi chua

    addi    $s7, $0, 512            #luu do rong man hinh vao s7
    #circle:
    addi    $a0, $0, 256        #x = 256
    addi    $a1, $0, 256        #y = 256
    addi    $a2, $0, 20     #r = 20
    addi    $s0, $0, 0x00FFFF00     # mau vang
    jal     DrawCircle
    nop
moving:                 # nhan ki tu tu ban phim va di chuyen
    # so sanh trong ma ASCII
    beq $t0,97,left         # 97 = 'a'
    beq $t0,100,right       # 100 = 'd'
    beq $t0,115,down        # 115 = 's'
    beq $t0,119,up          # 119 = 'w'
    j Input

    left:               # di chuyen sang trai
        addi $s0,$0,0x00000000  # gia tri mau bang 0, mau den, xoa diem
        jal DrawCircle      # xoa hinh tron vi tri cu, bang cach tô đen
```

```
        addi $a0,$a0,-1     # hoanh do tam hinh tron moi giam di 1
        add $a1,$a1, $0     # tung do tam giu nguyen
        addi $s0,$0,0x00FFFF00  # mau vang
        jal DrawCircle      # ve hinh tron vi tri moi
        jal Pause
        bltu $a0,20,reboundRight # nảy sang bên phải
        j Input
    right:              # di chuyển sang phải
        addi $s0,$0,0x00000000  #
        jal DrawCircle      # xoá hình tròn cũ
        addi $a0,$a0,1      # hoành độ tăng 1
        add $a1,$a1, $0     # tung độ giữ nguyên
        addi $s0,$0,0x00FFFF00
        jal DrawCircle      # vẽ hình tròn mới
        jal Pause
        bgtu $a0,492,reboundLeft # đập thành thì nảy sang trái
        j Input
    up:                 # di chuyển lên trên
        addi $s0,$0,0x00000000
        jal DrawCircle      # xoá hình tròn cũ
        addi $a1,$a1,-1     # tung độ giảm 1
        add $a0,$a0,$0      # hoành độ giữ nguyên
        addi $s0,$0,0x00FFFF00
        jal DrawCircle      # vẽ hình tròn mới
        jal Pause
        bltu $a1,20,reboundDown # đập thành thì nảy xuống dưới
        j Input
    down:
        addi $s0,$0,0x00000000
        jal DrawCircle  # xoá hình tròn cũ
        addi $a1,$a1,1  # tung độ tăng 1
        add $a0,$a0,$0  # hoành độ giữ nguyên
        addi $s0,$0,0x00FFFF00
        jal DrawCircle  # ve hình tròn mới
        jal Pause
        bgtu $a1,492,reboundUp  #đập thành thì nảy lên trên
        j Input
    reboundLeft:
        li $t3 97
        sw $t3,0($k0)    # thay đổi giá trị ở địa chỉ KEY_CODE để nhảy đến
left
        j Input
    reboundRight:
        li $t3 100
        sw $t3,0($k0)   # thay đổi giá trị ở địa chỉ KEY_CODE để nhảy đến
right
        j Input
    reboundDown:
```

```
        li $t3 115
        sw $t3,0($k0)    # thay đổi giá trị ở địa chỉ KEY_CODE để nhảy đến
down
        j Input
    reboundUp:
        li $t3 119
        sw $t3,0($k0)   # thay đổi giá trị ở địa chỉ KEY_CODE để nhảy đến up
        j Input
Input:
    ReadKey: lw $t0, 0($k0) # $t0 = [$k0] = KEY_CODE
    j moving

Pause:
    # delay ve
    addiu $sp,$sp,-4
    sw $a0, ($sp)
    #la $a0, 5   #system_sleep
    #li $v0, 32  #syscall value for sleep
    #syscall

    lw $a0,($sp)
    addiu $sp,$sp,4
    jr $ra
DrawCircle:
    # Dung thuat toan Bresenham de ve duong tron
    # Toa do tam duong tron can ve la (cx, cy)
    # Toa do diem can to mau la (cx +- x, cy +- y)
    # $a0 = cx va hoanh do pixel can to mau
    # $a2 = tung do pixel can to mau
    # s0 = colour mau vang
    # Khoi tao x = r = 20, y = 0
    # Cach tinh cac cap (x, y) tiep theo:
    # p = 3 - 2*r
    # neu p < 0 thi  p = p + 4y + 6 va (x, y) = (x, y+1)
    # neu p > 0 thi p = p + 4(y-x) +10 va (x, y) = (x-1, y+1)
    # neu y > x thi ta ve xong hinh tron, thoat khoi khối lệnh

    addiu   $sp, $sp, -32
    sw  $ra, 28($sp)
    sw  $a0, 24($sp)
    sw  $a1, 20($sp)
    sw  $a2, 16($sp)
    sw  $s4, 12($sp)
    sw  $s3, 8($sp)
    sw  $s2, 4($sp)
    sw  $s0, ($sp)

    #code goes here
```

```
    sll     $t4, $a2, 1
    li  $t3, 3
    sub $s2, $t3, $t4           # p = 3 - 2r
    add $s3, $0, $a2            #x = r = 20
    add $s4, $0, $0         #y = 0

    DrawCircleLoop:
    bgt     $s4, $s3, exitDrawCircle #neu y > x thi ta ve xong hinh tron,
thoat khoi vong lap
    nop

    #ve 4 diem doi xung nhau qua truc hoanh va truc tung, roi hoan doi x, y
cho nhau de ve tiep 4 diem doi xung qua phan giac
    jal plot8points
    nop
    # Tinh toan (x, y) tiep theo, roi ve cac diem moi
    sll $t3, $s4, 2             # $t3 = 4y
    add     $s2, $s2, $t3
    addi    $s2, $s2, 6         # p = p + 4y +6
    addi    $s4, $s4, 1         # y = y + 1

    blt $s2, 0, DrawCircleLoop      #if error >= 0, start loop again
    nop

    sll $t4, $s3, 2
    sub $s2, $s2, $t4
    addi    $s2, $s2, 4
    addi    $s3, $s3, -1

    j   DrawCircleLoop
    nop

    exitDrawCircle:

    lw  $s0, ($sp)
    lw  $s2, 4($sp)
    lw  $s3, 8($sp)
    lw  $s4, 12($sp)
    lw  $a2, 16($sp)
    lw  $a1, 20($sp)
    lw  $a0, 24($sp)
    lw  $ra, 28($sp)

    addiu   $sp, $sp, 32

    jr  $ra
    nop
```

```
plot8points:
    addiu   $sp, $sp, -4
    sw  $ra, 0($sp)

    jal plot4points
    nop

    beq     $s4, $s3, skipSecondplot
    nop

    # doi gia tri x va y cho nhau de ve 4 diem doi xung qua duong phan giac
    add $t2, $0, $s4            # t = y
    add $s4, $0, $s3            # y = x
    add $s3, $0, $t2            # x = t

    jal plot4points
    nop

    # doi gia tri x va y về như cũ
    add $t2, $0, $s4            # t = y
    add $s4, $0, $s3            # y = x
    add $s3, $0, $t2            # x = t

    skipSecondplot:

    lw  $ra, ($sp)
    addiu   $sp, $sp, 4

    jr  $ra
    nop

plot4points:

    addiu   $sp, $sp -4
    sw  $ra, ($sp)

    #$a0 = a0 + s3, $a2 = a1 + s4
    add $t0, $0, $a0            # $t0 luu cx
    add $t1, $0, $a1            # $t1 luu cy

    add $a0, $t0, $s3           # hoanh do diem can to mau: cx + x
    add $a2, $t1, $s4           # tung do diem can to mau: cy + y

    jal SetPixel                    # (cx + x, cy + y)
    nop

    sub $a0, $t0, $s3           #cx - x
    #add    $a2, $t1, $s4           #cy + y
```

```
    beq $s3, $0, skipXnotequal0     #if s3 (x) equals 0, skip
    nop

    jal     SetPixel                # (cx - x, cy +y)
    nop

    skipXnotequal0:
    sub $a2, $t1, $s4           #cy - y (a0 already equals cx - x)
    jal     SetPixel                    # (cx - x, cy - y)
    nop

    add $a0, $t0, $s3

    beq $s4, $0, skipYnotequal0
    nop

    jal SetPixel                #(cx + x, cy - y)
    nop

    skipYnotequal0:

    add $a0, $0, $t0
    add $a2, $0, $t1

    lw  $ra, ($sp)
    addiu   $sp, $sp, 4

    jr  $ra
    nop
SetPixel:
    #a0 x
    #a1 y
    #s0 colour
    addiu   $sp, $sp, -20           # Save return address on stack
    sw  $ra, 16($sp)
    sw  $s1, 12($sp)
    sw  $s0, 8($sp)         # Save original values of a0, s0, a2
    sw  $a0, 4($sp)
    sw  $a2, ($sp)

    lui $s1, 0x1001         # Dia chi bat dau bo nho man hinh
    sll $a0, $a0, 2
    add $s1, $s1, $a0
    mul     $a2, $a2, $s7
    mul $a2, $a2, 4
    add $s1, $s1, $a2           # Dia chi pixel can to mau
```

```
sw  $s0, ($s1)          # Luu gia tri màu vao dia chi pixel

lw  $a2, ($sp)          #retrieve original values and return address
lw  $a0, 4($sp)
lw  $s0, 8($sp)
lw  $s1, 12($sp)
lw  $ra, 16($sp)
addiu   $sp, $sp, 20

jr  $ra
nop
```