ĐẠI HỌC BÁCH KHOA HÀ NỘI
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

BÀI TẬP LỚN

THỰC HÀNH KIẾN TRÚC MÁY TÍNH

Giảng viên hướng dẫn: ThS. **Lê Bá Vui**

Lớp: 130939 – Kỳ: 20212

Nhóm sinh viên thực hiện:

1. Ngô Trung Hiếu - 20194560 - Đề tài 2
2. Đỗ Vũ Dũng - 20194520 - Đề tài 9

Hà Nội, tháng 7 năm 2022

Mục lục

Lời mở đầu

Nhóm 1 gồm 2 thành viên: Ngô Trung Hiếu và Đỗ Vũ Dũng.
Bản báo cáo khái quát quá trình thực hiện 2 bài tập lớn là bài 2 và bài 9.
Gồm các nội dung chính sau:

- Đề bài
- Phân tích đề bài
- Cách làm
- Mã nguồn
- Kết quả chạy mô phỏng

Bản báo cáo sẽ không tránh khỏi những sai sót. Nhóm rất mong nhận được ý kiến góp ý của thầy giáo và các bạn.

Chúng em chân thành cảm ơn.

Bài 2: Vẽ hình trên màn hình Bitmaps

Đề bài:

Viết chương trình vẽ một quả bóng hình tròn 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 pixel 1x1.

- 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), tăng tốc độ (Z), giảm tốc độ (X) trong bộ giả lập Keyboard and Display MMIO Simulator).

- Vị trí bóng ban đầu ở giữa màn hình

1. Phân tích yêu cầu:

- Vẽ một hình tròn ở giữa màn hình có tâm O(256, 256), bán kính R.
- Sử dụng các nút bấm di chuyển lên (W), di chuyển xuống (S), sang trái (A), sang phải (D) để đổi hướng di chuyển của quả bóng.
- Khi quả bóng va vào thành thì bật ngược lại.
- Sử dụng các nút bấm thay đổi tốc độ: tăng tốc độ (Z), giảm tốc độ (X).

2. Cách làm:

- **Vẽ đường tròn**

Ta có phương trình đường tròn ban đầu là

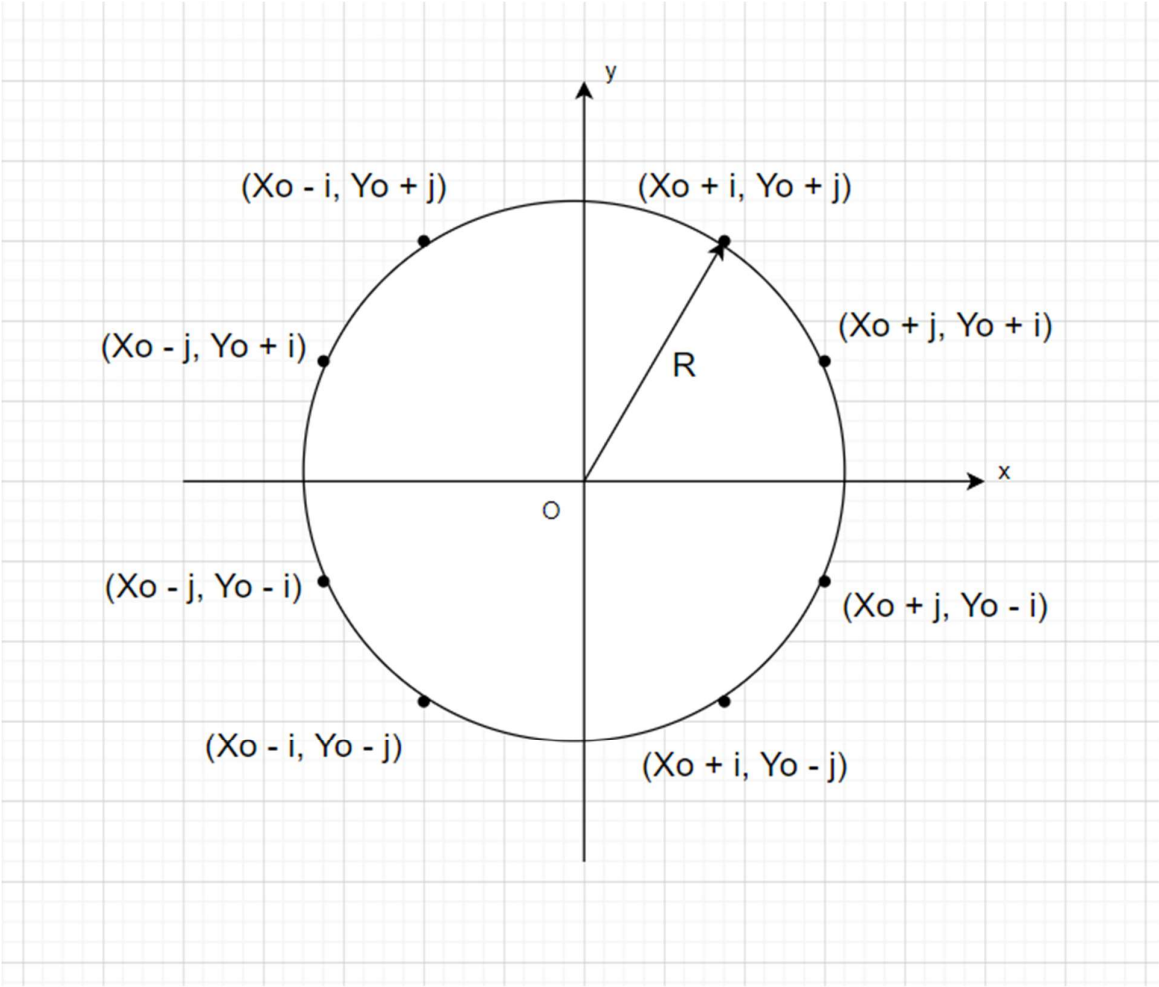
$(x - a)^2 + (y - b)^2 = r^2$ với vị trí tâm gốc là (0, 0) => a = b = 0

Do đó phương trình đường tròn có dạng

$x^2 + y^2 = r^2$

=> $y = \sqrt{(r^2 - x^2)}$

$\Leftrightarrow j = \sqrt{(r^2 - i^2)}$



- Duyệt i chạy từ i -> R lần lượt ta suy ra được $j = \sqrt{(r^2 - i^2)}$
- Lưu j vào 1 mảng
- Sử dụng vòng lặp để vẽ đường tròn. Tạo 8 điểm trên đường tròn: (Xo + i, Yo + j), (Xo + j, Yo + i), (Xo + i, Yo - j), (Xo + j, Yo - i), (Xo - i, Yo - j), (Xo - j, Yo - i), (Xo - i, Yo + j), (Xo - j, Yo + i)
- Cho i chạy từ i -> R tương ứng lần lượt với từng phần tử trong mảng lưu phần tử j. Để tạo từng điểm ảnh. Vòng lặp chạy sẽ tạo đường tròn.

- **Di chuyển**

Lưu các nút bằng các kí tự trong bảng mã ASCII vào một thanh ghi. Sau khi ấn phím lên, xuống, trái, phải, sẽ kiểm tra từng trường hợp

một. Ấn phím Enter đường tròn sẽ chuyển về màu đen trùng với màu màn hình. Chương trình kết thúc.

- **Thay đổi tốc độ**

Sử dụng service sleep với code in $v0 để thay đổi tốc độ. Cho 5 mốc tốc độ là VERYSLOW, SLOW, NORMAL, FAST, VERYFAST. Gắn tốc độ ban đầu là NORMAL khi thay đổi tốc độ thì thực hiện so sánh xem thuộc tốc độ nào tiếp tục chạy chương trình. Nếu tốc độ hiện tại ở VERYSLOW thì ấn X tốc độ vẫn chỉ ở VERYSLOW, ngược lại khi tốc độ hiện tại ở VERYFAST thì ấn Z tốc độ vẫn chỉ ở VERYFAST.

3. Mã nguồn:

```
.eqv SCREEN     0x10010000  #Man hinh bitmap
.eqv GREEN  0x0000FF00
.eqv BACKGROUND     0x00000000
.eqv KEY_A  97
.eqv KEY_S  115
.eqv KEY_D  100
.eqv KEY_W  119
.eqv KEY_Z  122
.eqv KEY_X  120
.eqv KEY_ENTER  0x0000000a
.eqv DELTA_X    10
.eqv DELTA_Y    10
.eqv VERYFAST   10  # 10ms
.eqv FAST   70  # 70ms
.eqv NORMAL 130 # 130ms
.eqv SLOW   190 # 190ms
.eqv VERYSLOW   250 # 250ms
.eqv KEY_CODE   0xFFFF0004  # Ki tu go vao
.eqv KEY_READY  0xFFFF0000  # Kiem ki tu da san sang de doc chua

.macro delay_time(%time)    # delay 1 khoang thoi gian tinh bang mili giay
    li $a0, %time
    li $v0, 32
    syscall
.end_macro

.macro set_color_and_draw_circle(%color)
    li $s5, %color  # Dat mau
    jal draw_circle # de xoa duong tron cu.
.end_macro

.macro add_point(%r1, %r2, %r3)
    add $sp, $sp, -12
    sw %r1, 0($sp)
    sw %r2, 4($sp)
```

```
    sw %r3, 8($sp)
.end_macro

.macro add_position(%r1, %r2, %r3, %r4)
    add $sp, $sp, -16
    sw %r1, 0($sp)
    sw %r2, 4($sp)
    sw %r3, 8($sp)
    sw %r4, 12($sp)
.end_macro

.macro get_point(%r1, %r2, %r3)
    lw %r1, 0($sp)
    lw %r2, 4($sp)
    lw %r3, 8($sp)
    add $sp, $sp, 12
.end_macro

.macro get_position(%r1, %r2, %r3, %r4)
    lw %r1, 0($sp)
    lw %r2, 4($sp)
    lw %r3, 8($sp)
    lw %r4, 12($sp)
    add $sp, $sp, 16
.end_macro

.kdata
    CIRCLE_ARRAY:   .space 512
.text
li $s0, 256 # Xo = 256      Toa do X cua tam duong tron
li $s1, 256 # Yo = 256      Toa do Y cua tam duong tron
li $s2, 24  # R = 24        Ban kinh cua tam duong tron
li $s3, 512 # SCREEN_WIDTH = 512    Be ngang man hinh
li $s4, 512 # SCREEN_HEIGHT = 512   Chieu cao man hinh
li $s5, GREEN   # Dat mau hinh tron
li $t6, NORMAL  # Dat toc do ban dau la NORMAL
mul $s6, $s2, $s2   # R^2
li $s7, 0       # dx = 0
li $t8, DELTA_Y     # dy = 10
li $t7, 0   # kiem tra de khoi tao hinh tron ban dau tai giua man hinh

jal start_draw_circle
nop

programLoop:
read_from_keyboard:
    lw $k1, KEY_READY   # Vong lap cho ban phim san sang
    beqz $k1, check_position
    nop
    lw $k0, KEY_CODE        # Doc ky tu ban phim
    beq $k0, KEY_A, case_a  # Kiem tra nut A
    beq $k0, KEY_S, case_s  # Kiem tra nut S
    beq $k0, KEY_D, case_d  # Kiem tra nut D
    beq $k0, KEY_W, case_w  # Kiem tra nut W
    beq $k0, KEY_Z, case_z  # Kiem tra nut Z
```

```
    beq $k0, KEY_X, case_x  # Kiem tra nut X
    beq $k0, KEY_ENTER, case_enter  # Kiem tra nut ENTER
    j check_position
case_a:
    addi $t7, $zero, 1
    jal move_to_left
    j check_position
case_s:
    addi $t7, $zero, 1
    jal move_to_down
    j check_position
case_d:
    addi $t7, $zero, 1
    jal move_to_right
    j check_position
case_w:
    addi $t7, $zero, 1
    jal move_to_up
    j check_position
case_z:
    addi $t7, $zero, 1
    beq $t6, VERYFAST, notMinus # kiem tra neu dat toc do VERYFAST thi se kho
ng tang toc hon duoc nua
    subi $t6, $t6, 60
    notMinus:
    set_color_and_draw_circle(BACKGROUND)
    j draw
case_x:
    addi $t7, $zero, 1
    beq $t6, VERYSLOW, notPlus  # kiem tra neu dat toc do VERYSLOW thi se kho
ng giam toc hon duoc nua
    addi $t6, $t6, 60
    notPlus:
    set_color_and_draw_circle(BACKGROUND)
    j draw
case_enter:
    j endProgram

check_position:
checkRightExtreme:
    add $v0, $s0, $s2
    add $v0, $v0,$s7
    ble $v0, $s3, checkLeftExtreme  # Neu Xo + R + DELTA_X > SCREEN_WIDTH thi
 move_to_left
    jal move_to_left
    nop
checkLeftExtreme:
    sub $v0, $s0, $s2
    add $v0, $v0, $s7
    ble $zero, $v0, checkTopExtreme # Neu Xo - R + DELTA_X < 0 thi move_to_ri
ght
    jal move_to_right
    nop
checkTopExtreme:
    sub $v0, $s1, $s2
```

```
    add $v0, $v0, $t8
    ble $zero, $v0, checkBottomExtreme  # Neu Yo - R + DELTA_Y < 0 thi move_t
o_down
    jal move_to_down
    nop
checkBottomExtreme:
    add $v0, $s1, $s2
    add $v0, $v0, $t8
    beq $t7, 1, activeCircle
    ble $v0, $s4, drawInit # ve duong tron ban dau tai giua man hinh
    activeCircle:
    ble $v0, $s4, draw  # neu Yo + R + DELTA_Y > SCREEN_HEIGHT thi move_to_up
    jal move_to_up
    nop

#---------------------------------------------------------------------------
----------------------------------------
#   Xoa duong tron cu va ve duong tron moi
#---------------------------------------------------------------------------
----------------------------------------

drawInit:
    set_color_and_draw_circle(BACKGROUND) # Ve duong tron trung mau nen
    addi $s0, $s0, 0            # vi tri ban dau x = 256
    addi $s1, $s1, 0            # vi tri ban dau y = 256
    set_color_and_draw_circle(GREEN)    # Ve duong tron moi
    beq $t6, VERYSLOW, veryslow
    beq $t6, SLOW, slow
    beq $t6, NORMAL, normal
    beq $t6, FAST, fast
    beq $t6, VERYFAST, veryfast

draw:
    set_color_and_draw_circle(BACKGROUND) # Ve duong tron trung mau nen
    add $s0, $s0, $s7       # Cap nhat toa do moi
    add $s1, $s1, $t8       # cua duong tron
    set_color_and_draw_circle(GREEN)    # Ve duong tron moi
    beq $t6, VERYSLOW, veryslow
    beq $t6, SLOW, slow
    beq $t6, NORMAL, normal
    beq $t6, FAST, fast
    beq $t6, VERYFAST, veryfast

endProgram:             # Ket thuc chuong trinh
    set_color_and_draw_circle(BACKGROUND)
    li $v0, 10
    syscall

veryslow:
    delay_time(VERYSLOW)        # Dat toc do VERYSLOW
    j programLoop
slow:
    delay_time(SLOW)        # Dat toc do SLOW
    j programLoop
normal:
```

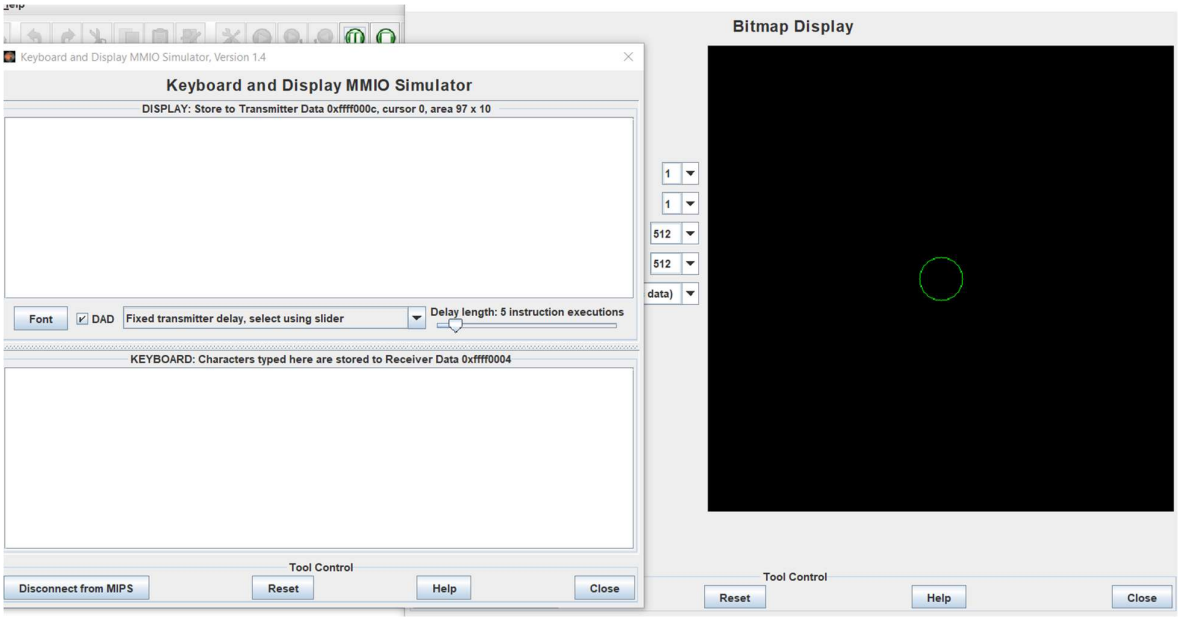
```
    delay_time(NORMAL)      # Dat toc do NORMAL
    j programLoop
fast:
    delay_time(FAST)        # Dat toc do FAST
    j programLoop
veryfast:
    delay_time(VERYFAST)        # Dat toc do VERYFAST
    j programLoop

#------------------------------------------------------------------------------
--------------------------------------
#   Stack Position luu lai vi tri( toa do) cac diem cua duong tron
#   Tao mang du lieu luu toa do cac diem cua duong tron
#   Luu lai cac gia tri tuong ung cua j khi i chay tu 0 -> R
#------------------------------------------------------------------------------
--------------------------------------

start_draw_circle:
    add_position($ra, $t0, $t3, $t5)
    li $t0, 0       # i = 0
    la $t5, CIRCLE_ARRAY
loop:
    slt $v0, $t0, $s2   # Neu i > R -> end_start_draw_circle
    beqz $v0, end_draw_circle
    mul $t3, $t0, $t0   # i^2
    sub $t3, $s6, $t3   # $t3 = R^2 - i^2
    move $v0, $t3
    jal sqrt
    sw $a0, 0($t5)      # Luu j = sqrt(R^2 - i^2) vao mang du lieu
    addi $t0, $t0, 1        # i = i + 1
    add $t5, $t5, 4
    j loop
end_draw_circle:
    get_position($ra, $t0, $t3, $t5)
    jr $ra
    nop

#------------------------------------------------------------------------------
--------------------------------------
#   Ve diem tren duong tron
#   Ve dong thoi 2 diem (X0 + i, Y0 +j ) va (Xo + j, Xo + i)
#   Tham so $a0 = i ; $a1 = j
#------------------------------------------------------------------------------
--------------------------------------

draw_circle_point:
    add_point($t1, $t2, $t4)

    add $t1, $s0, $a0   # Xi = Xo + i
    add $t4, $s1, $a1   # Yi = Yo + j
    mul $t2, $t4, $s3   # Yi * SCREEN_WIDTH
    add $t1, $t1, $t2   # Yi * SCREEN_WIDTH + Xi (Toa do 1 chieu cua diem an
h)
    sll $t1, $t1, 2     # Dia chi tuong doi cua diem anh
    sw $s5, SCREEN($t1) # Ve diem anh
```

```
    add $t1, $s0, $a1   # Xi = Xo + j
    add $t4, $s1, $a0   # Yi = Yo + i
    mul $t2, $t4, $s3   # Yi * SCREEN_WIDTH
    add $t1, $t1, $t2   # Yi * SCREEN_WIDTH + Xi (Toa do 1 chieu cua diem an
h)
    sll $t1, $t1, 2     # Dia chi tuong doi cua diem anh
    sw $s5, SCREEN($t1) # Ve diem anh
    get_point($t1, $t2, $t4)
    jr $ra

#-------------------------------------------------------------------------------
---------------------------------------
#   Ve duong tron
#-------------------------------------------------------------------------------
---------------------------------------

draw_circle:
    add_position($ra, $t0, $t1, $t3)
    li $t0, 0   # init i = 0
loop_drawCircle:
    slt $v0, $t0, $s2
    beqz $v0,  end_drawCircle

    sll $t1, $t0, 2
    lw $t3, CIRCLE_ARRAY($t1) # Load j to $t3

    move $a0, $t0   # $a0 = i
    move $a1, $t3   # $a1 = j
    jal draw_circle_point   # ve 2 diem (Xo + i, Yo + j), (Xo + j, Yo + i)
    sub $a1, $zero, $t3
    jal draw_circle_point   # ve 2 diem (Xo + i, Yo - j), (Xo + j, Yo - i)
    sub $a0, $zero, $t0
    jal draw_circle_point   # ve 2 diem (Xo - i, Yo - j), (Xo - j, Yo - i)
    add $a1, $zero, $t3
    jal draw_circle_point   # ve 2 diem (Xo - i, Yo + j), (Xo - j, Yo + i)

    addi $t0, $t0, 1 # i = i + 1
    j loop_drawCircle
end_drawCircle:
    get_position($ra, $t0, $t1, $t3)
    jr $ra

#   Ham tinh can bac hai

#   $v0 = S, $a0 = sqrt(S)
sqrt:
    mtc1 $v0, $f0
    cvt.s.w $f0, $f0
    sqrt.s $f0, $f0
    cvt.w.s $f0, $f0
    mfc1 $a0, $f0
    jr $ra

#-------------------------------------------------------------------------------
---------------------------------------
```

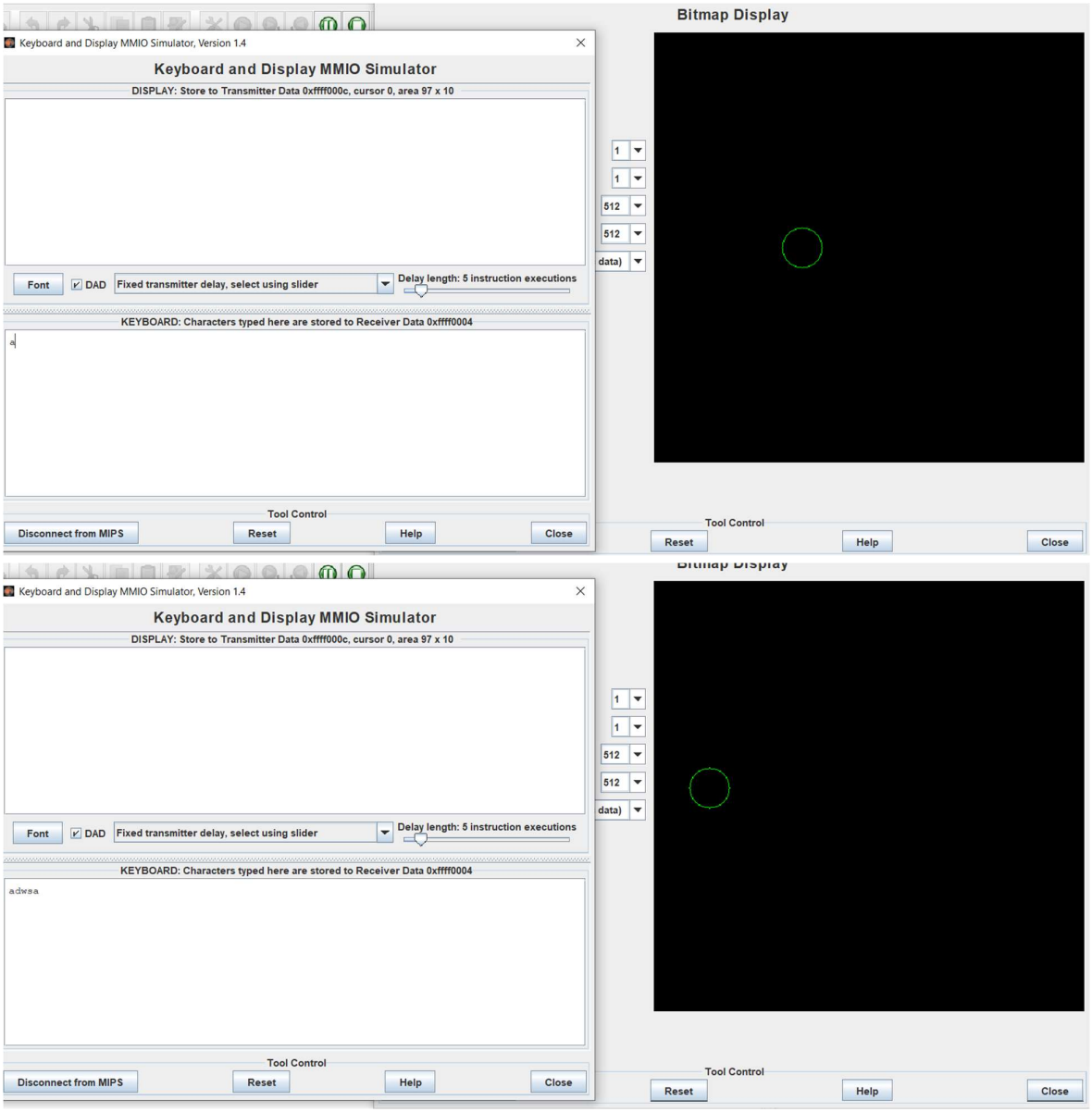
```
#   Di chuyen
#-------------------------------------------------------------------------------
--------------------------------------

move_to_left:
    li $s7, -DELTA_X
    li $t8, 0
    jr $ra
move_to_right:
    li $s7, DELTA_X
    li $t8, 0
    jr $ra
move_to_up:
    li $s7, 0
    li $t8, -DELTA_Y
    jr $ra
move_to_down:
    li $s7, 0
    li $t8, DELTA_Y
    jr $ra
```

4. Kết quả chạy mô phỏng:

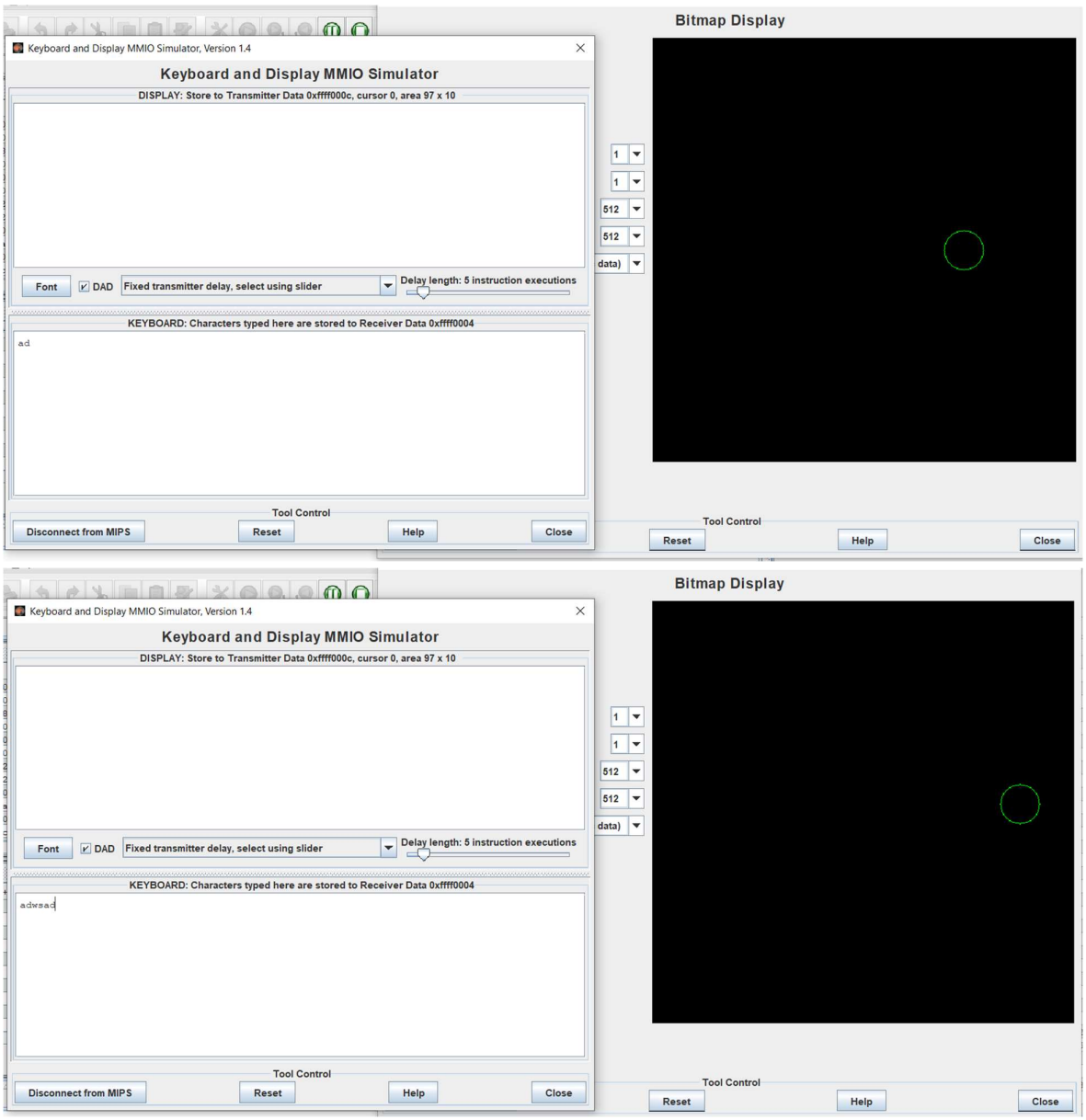
Chạy chương trình: vòng tròn nằm giữa màn hình



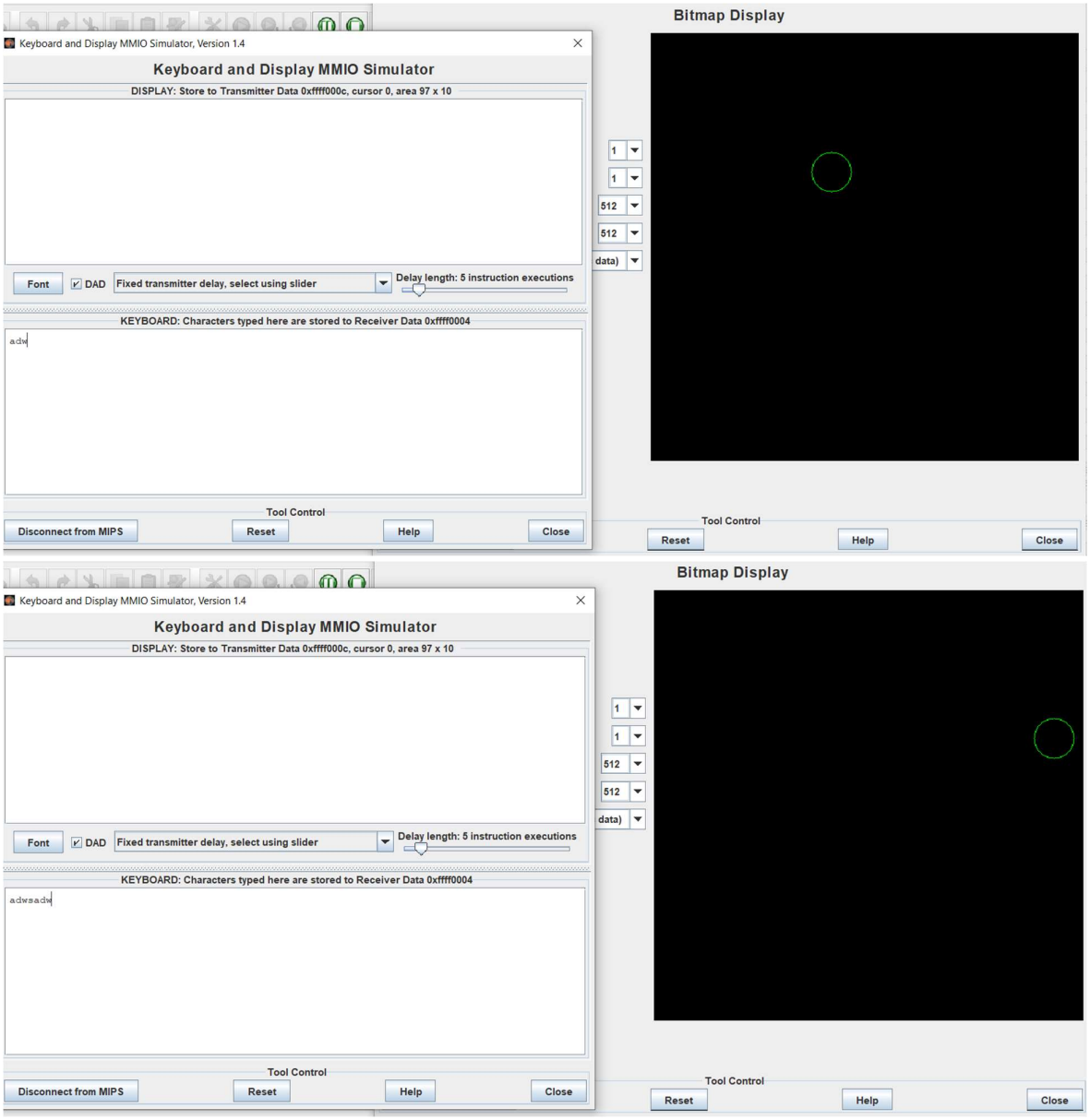
- Nhập a: sang trái nếu vượt quá SCREEN_WIDTH sẽ nảy lại



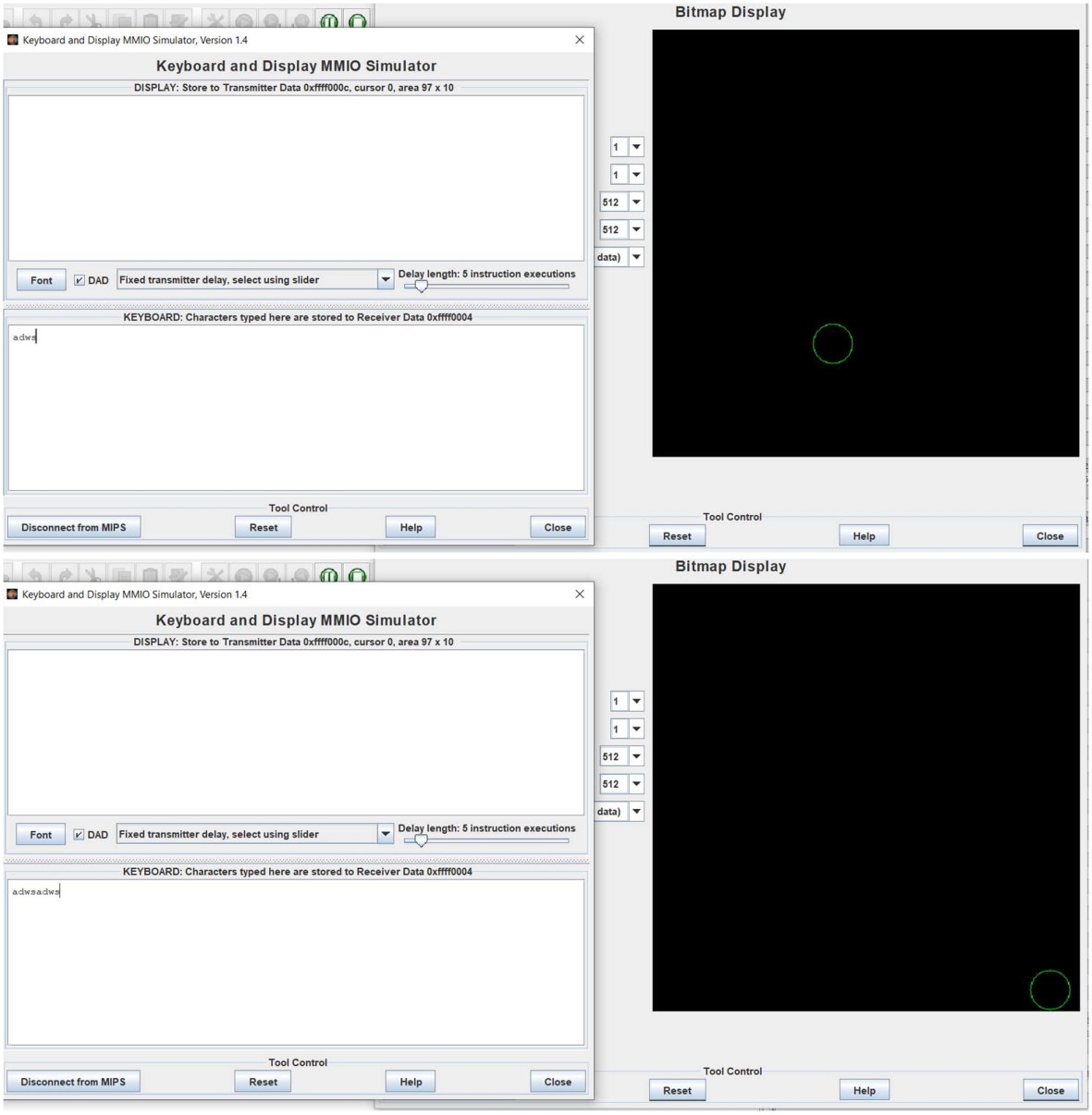
- Nhập b: sang phải nếu vượt quá SCREEN_WIDTH sẽ nảy lại



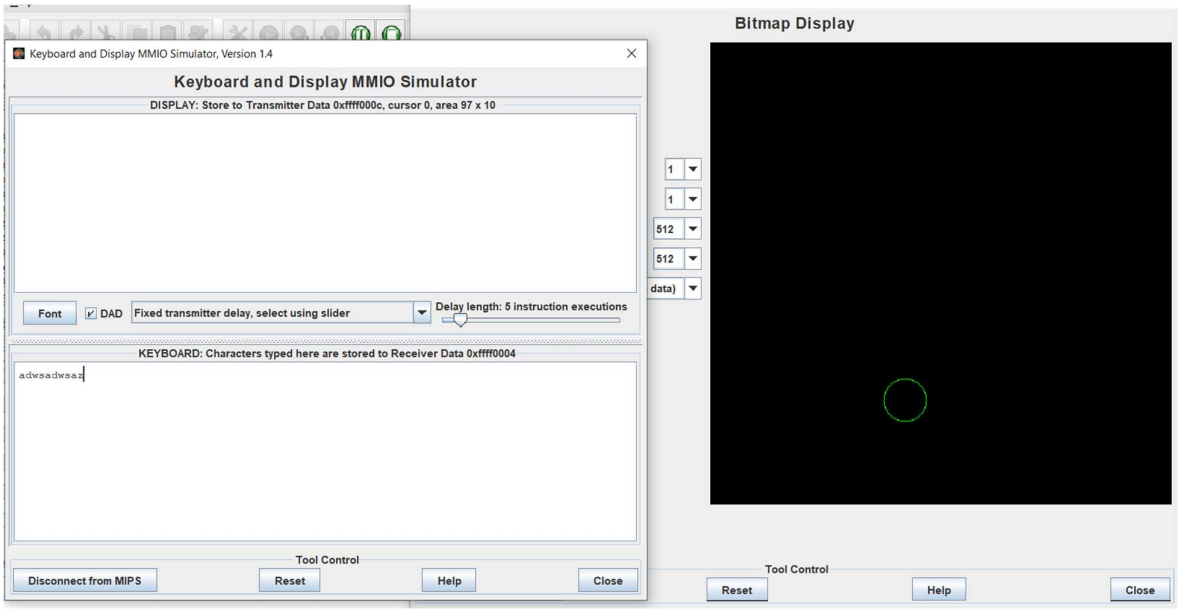
- Nhập w: lên trên nếu vượt quá SCREEN_WIDTH sẽ nảy lại



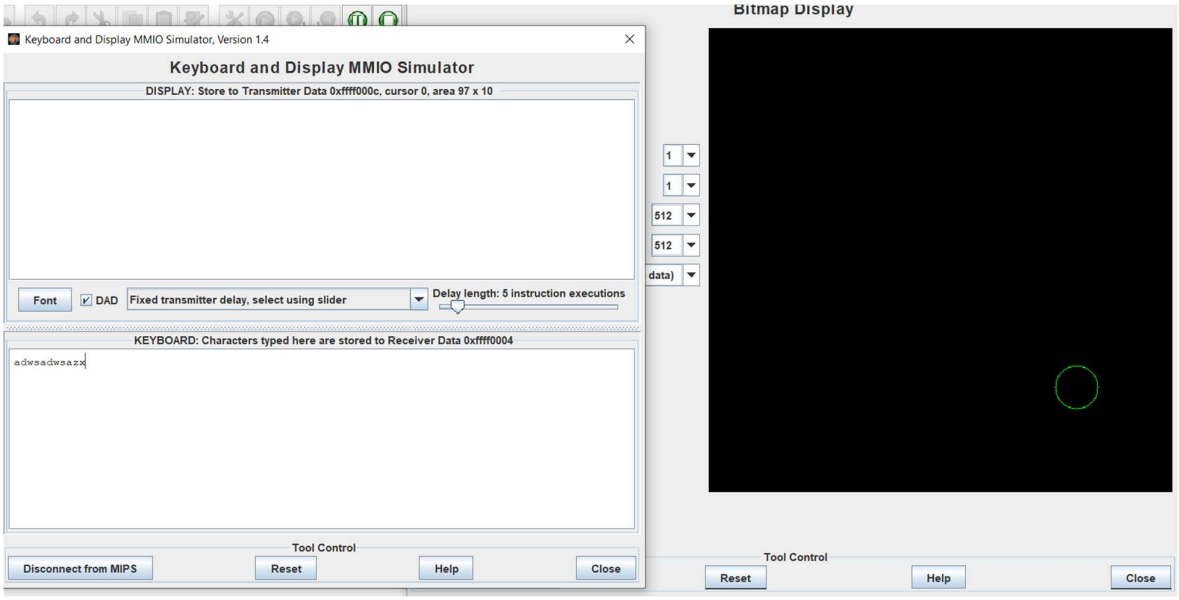
- Nhập s: xuống dưới nếu vượt quá SCREEN_WIDTH sẽ nảy lại



- Nhấn z: tăng tốc



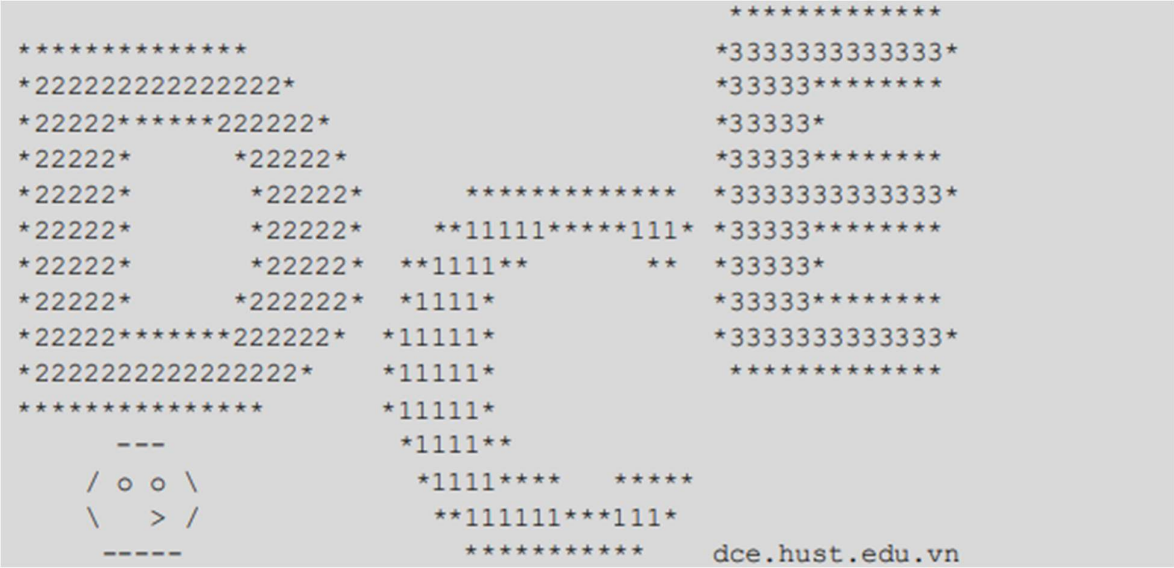
- Nhấn x: giảm tốc



Bài 9: Vẽ hình bằng kí tự ASCII

Đề bài:

*Cho hình ảnh đã được chuyển thành các kí tựASCII như hình vẽ. Đây là hình của chữ DCE có viền * và màu là các con số.*



- *Hãy hiển thị hình ảnh trên lên giao diện console(hoặc giao diện Display trong công cụ giả lập Keyboard and Display MMIO Simulator).*
- *Hãy sửa ảnh để các chữ cái DCE chỉ còn lại viền, không còn màu số ở giữa, và hiển thị.*
- *Hãy sửa ảnh để hoán đổi vị trí của các chữ, thành ECD, và hiển thị. Để đơn giản, các hoạ tiết đính kèm cũng được phép di chuyển theo*
- *Hãy nhập từ bàn phím kí tự màu cho chữ D, C, E, rồi hiển thị ảnh trên với màu mới.*

Chú ý: ngoài vùng nhớ lớn chứa ảnh được chứa sẵn trong code, không được tạo thêm vùng nhớ mới để chứa ảnh hiệu chỉnh.

1. Phân tích yêu cầu:

- Hiển thị hình ảnh trên lên giao diện console (hoặc giao diện Display trong công cụ giả lập Keyboard and Display MMIO Simulator).

- Sửa ảnh để các chữ cái DCE chỉ còn lại viền, không còn màu số ở giữa, và hiển thị.
- Sửa ảnh để hoán đổi vị trí của các chữ, thành ECD, và hiển thị. Để đơn giản, các hoạ tiết đính kèm cũng được phép di chuyển theo
- Nhập từ bàn phím kí tự màu cho chữ D, C, E, rồi hiển thị ảnh trên với màu mới.

2. Cách làm:

Tách hình ảnh thành từng string theo từng dòng. (16 dòng => 16 string)

- Hiển thị hình ảnh

- Sử dụng vòng lặp để in ra từng string tương ứng từng dòng

- Hiển thị hình ảnh chỉ có viền, không có màu

- Duyệt từng kí tự theo từng dòng

- Nếu gặp chữ số (≥ 0 & ≤ 9) thì thay kí tự đó bằng space để xóa màu. Nếu gặp kí tự khác chữ số thì in ra như bình thường

- Hiển thị ECD

- Chia hình ảnh thành 4 phần:
    - Chữ D: Từ cột 0 đến cột 22
    - Chữ C: Từ cột 23 đến cột 42
    - Chữ E: Từ cột 43 đến cột 58
    - Còn lại: Từ cột 59 đến 61

- In từng kí tự theo từng dòng lần lượt các kí tự từ vị trí 43 => 58, sau đó từ cột 23 => 42, sau đó từ cột 0 => 22 và từ cột 59 => 61

- Đổi màu

- Lưu các màu hiện tại của D, C, E lần lượt vào các thanh ghi t1, t3, t5

- Nhập màu muốn thay đổi lần lượt cho D, C, E và lưu vào các thanh ghi t2, t4, t6. Nếu số nhập không phải màu từ 0 => 9 yêu cầu nhập lại

- Duyệt từng kí tự theo từng dòng lần lượt theo từng chữ cái
    - Chữ D (0->23): so sánh kí tự hiện tại với thanh ghi t1 lưu màu lúc đầu của D , nếu bằng nhau thì lưu giá trị màu mới (t2) vào kí tự hiện tại, nếu không thì duyệt kí tự tiếp theo. Nếu kí tự hiện tại ở vị trí 24 thì duyệt sang chữ C
    - Tương tự với chữ C (24-> 43) và E(44->61)

- Sau khi duyệt hết 1 dòng thì in ra và duyệt dòng tiếp theo cho đến khi hết 16 dòng

3. Mã nguồn:

```
.data
String0: .space 5000
String1:  .asciiz "                                          ************
  \n"
String2:  .asciiz " ****************                       *3333333333333
* \n"
String3:  .asciiz " *2222222222222222*                      *33333********
  \n"
String4:  .asciiz " *22222*******222222*                    *33333*
  \n"
String5:  .asciiz " *22222*       *22222*                   *33333********
  \n"
String6:  .asciiz " *22222*        *22222*      *************  *3333333333333
* \n"
String7:  .asciiz " *22222*        *22222*    **11111*****111* *33333********
  \n"
String8:  .asciiz " *22222*        *22222*  **1111**       **  *33333*
  \n"
String9:  .asciiz " *22222*       *22222*   *1111*            *33333********
  \n"
String10: .asciiz " *22222*******222222*   *11111*            *3333333333333
* \n"
String11: .asciiz " *2222222222222222*     *11111*             *************
  \n"
String12: .asciiz " ****************       *11111*
  \n"
String13: .asciiz "       ---               *1111**
  \n"
String14: .asciiz "     / o o \\              *1111****   *****
   \n"
String15: .asciiz "     \\   > /               **111111***111*
   \n"
String16: .asciiz "      -----                  ***********    dce.hust.edu.v
n \n"

Message1: .asciiz"\n\n---------------------------------------- Print DCE ----
----------------------------------------\n\n"
Message2: .asciiz"\n\n---------------------------------- Print without color
 ----------------------------------------\n\n"
Message3: .asciiz"\n\n------------------------------- Change position of D &
 E ----------------------------------\n\n"
Message4: .asciiz"\n\n------------------------------------ Change color ----
----------------------------------------\n\n"

StringD: .asciiz"New color of D (0->9): "
StringC: .asciiz"New color of C (0->9): "
```

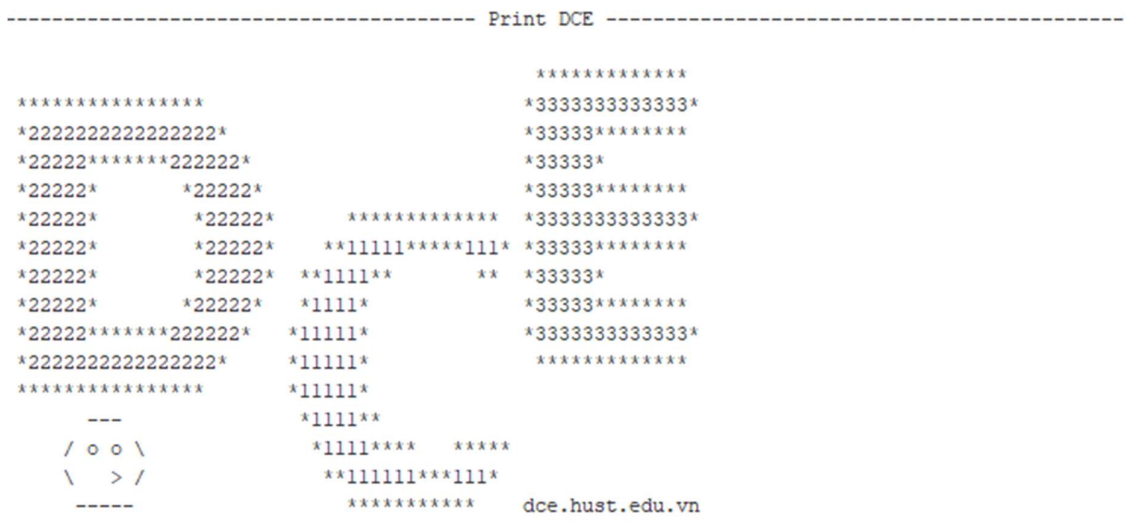
```
StringE: .asciiz"New color of E (0->9): "
.text

############## print ###############
title1: la $a0, Message1
    li $v0,4
    syscall
main1:  li $s0,0                # i=0
    la $s2, String1
Loop_print: addi $a0,$s2,0
        li $v0,4
        syscall
        addi $s2, $s2, 62       # next string(line)
        addi $s0, $s0,1         # i++
        beq $s0, 16, end_print      # if i=16 => end print (line =16)
        j Loop_print
end_print:

############ IN KHONG MAU ############
title2: la $a0, Message2
    li $v0,4
    syscall
main2:  li $s0,0                #i=0
    la $s2, String1
Loop:   li $s1,0                #j=0
print_line: add $t1, $s2, $s1       # t1 = address of stringX[j]
        lb $t2, 0($t1)          # t2 = stringX[j]
        blt $t2, 48, printc         # t2 < 48('0') jump printc
        bgt $t2, 57, printc         # t2 > 57('9') jump printc
        addi $t2, $zero, 32
printc: addi $a0, $t2, 0    # print character
    li $v0,11
    syscall
    addi $s1,$s1,1      # j= j+1
    beq $s1,62, next_line
    j print_line
next_line:  addi $s0,$s0,1  #i= i+1
    addi $s2,$s2,62     # next string
    beq $s0,16, end_loop
    j Loop
end_loop:

############ change D & E #############
title3: la $a0, Message3
    li $v0,4
    syscall
# D: 0-> 22
# C: 23 -> 42
# E: 43 -> 58
main3:  li $s0,0        #i=0
    la $s2, String1
Loop1:  li $s1,43       # j=43
Print_E:
    add $t0, $s2, $s1   # t0 = address of stringX[j]
    lb $t2, 0($t0)      # t2 = stringX[j]
```

```
    addi $a0, $t2, 0
    li $v0, 11      # print character
    syscall
    addi $s1,$s1,1      # j++
    beq $s1,59,Loop2    # if j = 59 jump loop2
    j Print_E
Loop2:  li $s1,23       # j=23
Print_C:
    add $t0, $s2, $s1   # t0 = address of stringX[j]
    lb $t2, 0($t0)      # t2 = stringX[j]
    addi $a0, $t2, 0
    li $v0, 11      # print character
    syscall
    addi $s1,$s1,1
    beq $s1,43,Loop3    # if i =43 jump loop3
    j Print_C
Loop3:  li $s1,0        # j=0
Print_D: add $t0, $s2, $s1  # t0 = address of stringX[j]
    lb $t2, 0($t0)      # t2 = stringX[j]
    addi $a0, $t2, 0
    li $v0, 11      # print character
    syscall
    addi $s1,$s1,1
    beq $s1,23,Loop4    # if i=23 jump loop4
    j Print_D
Loop4: li $s1,60        # j=59
Print: add $t0, $s2, $s1    # t0 = address of stringX[j]
    lb $t2, 0($t0)      # t2 = stringX[j]
    addi $a0, $t2, 0
    li $v0, 11      # print character
    syscall
    addi $s1,$s1,1
    beq $s1,62,Line     # if j = 62 => next line
    j Print
Line:   addi $s0,$s0,1
    beq $s0,16,end      # if i = 16 end
    addi $s2,$s2,62
    j Loop1
end:

############## Change color ##########
title4: la $a0, Message4
    li $v0,4
    syscall
####
    li $t1, 50 # color base of D
    li $t3, 49 # color base of C
    li $t5, 51 # color base of E

#### input color
Input_D: li $v0,4
    la $a0, StringD
    syscall
    li $v0,5
    syscall
```

```
    blt $v0,0, Input_D  #if v0<0 input again
    bgt $v0,9, Input_D  #if v0>9 input again
    addi $t2,$v0,48     # '0' ascii: 48 , t2: store new color of D
Input_C: li $v0,4
    la $a0, StringC
    syscall
    li $v0,5
    syscall
    blt $v0,0, Input_C  #if v0<0 input again
    bgt $v0,9, Input_C  #if v0>9 input again
    addi $t4,$v0,48     # t4: store new color of C
Input_E: li $v0,4
    la $a0, StringE
    syscall
    li $v0,5
    syscall
    blt $v0,0, Input_E  #if v0<0 input again
    bgt $v0,9, Input_E  #if v0>9 input again
    addi $t6,$v0,48     #t6: store new color of E

####
main4:  li $s0, 0       # i=0
    la $s2,String1

row:    li $s1,0        # j=0
check:  add $t0,$s2, $s1    # t0 = address of StringX[j]
    lb $a0,($t0)        # a0 = stringX[j]
checkD: bgt $s1, 23, checkC     # if j>23 check C
    beq $a0, $t1, fixD  # if stringX[j] = color base => fix
    j next_char
fixD:   sb $t2, 0($t0)      # stote new color into stringX[j]
    j next_char
checkC: bgt $s1, 43, checkE     # if j>43 check E
    beq $a0, $t3, fixC  # if stringX[j] = color base => fix
    j next_char
fixC:   sb $t4, 0($t0)      # stote new color into stringX[j]
    j next_char
checkE:             #bgt $s1, 23, checkC # if j>23 check C
    beq $a0, $t5, fixE  # if stringX[j] = color base => fix
    j next_char
fixE:   sb $t6, 0($t0)      # stote new color into stringX[j]
    j next_char
next_char:  addi $s1,$s1, 1 # j++
        beq $s1,62,end_row
        j check
end_row:    addi $a0, $s2,0 # print line
    li $v0,4
    syscall
    addi $s0,$s0,1
    beq $s0,16, end_change
    addi $s2,$s2,62
    j row
end_change:
```

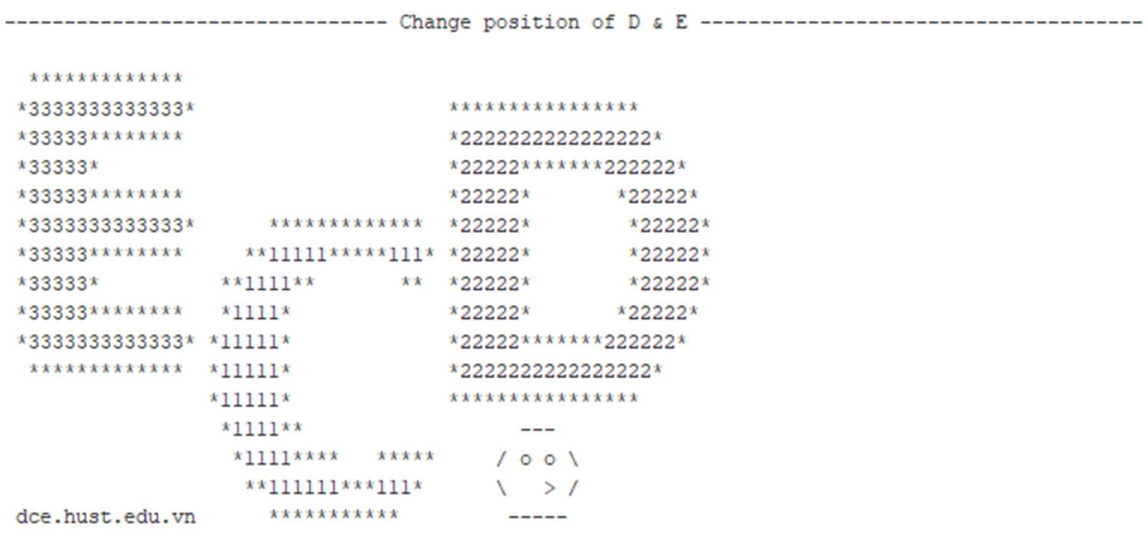
4. Kết quả chạy mô phỏng:

- In ra màn hình DCE



- In ra màn hình DCE không màu

- Đổi vị trí DCE thành ECD



- Đổi màu cho DCE