Báo cáo bài tập lớn

Học phần:Thực hành kiến trúc máy tính

Nhóm 12:

Mã lớp học: 147789

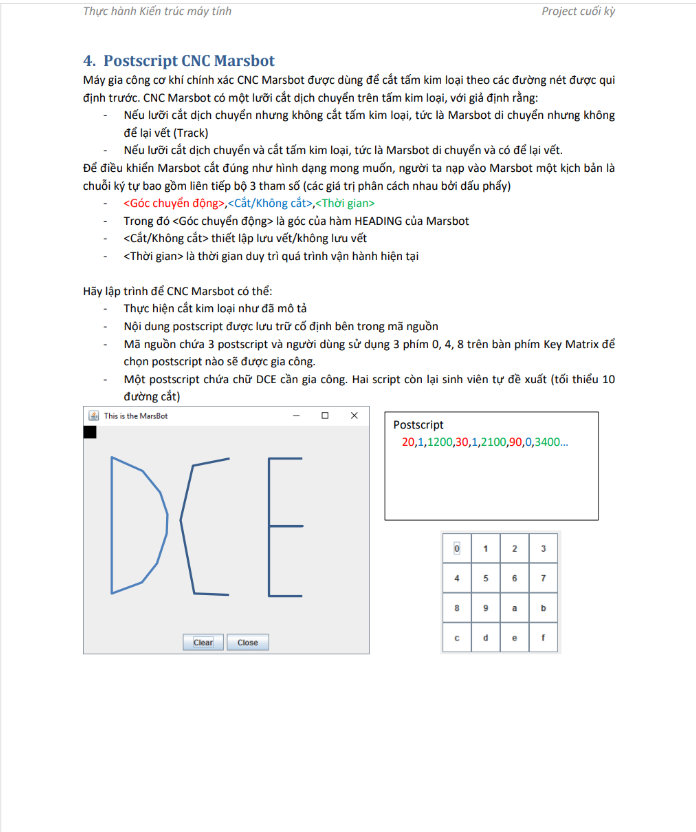
Giảng viên: ThS. Lê Bá Vui

Sinh viên thực hiện

Họ và tên: Nguyễn Đức Bình

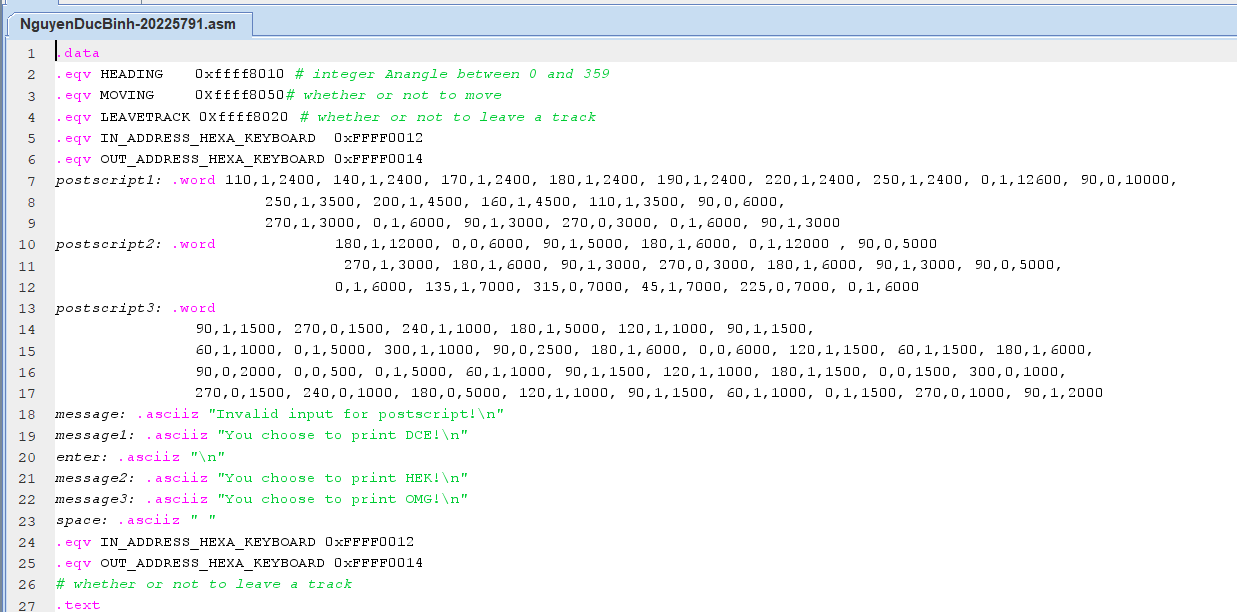
Mssv: 20225791

Đề tài: 4



1: Phân tích cách làm và thuật toán

B1: Khai báo địa chỉ quan trọng để truy cập vào marsbot và keyboard matrix và một số biến string để in ra màn hình



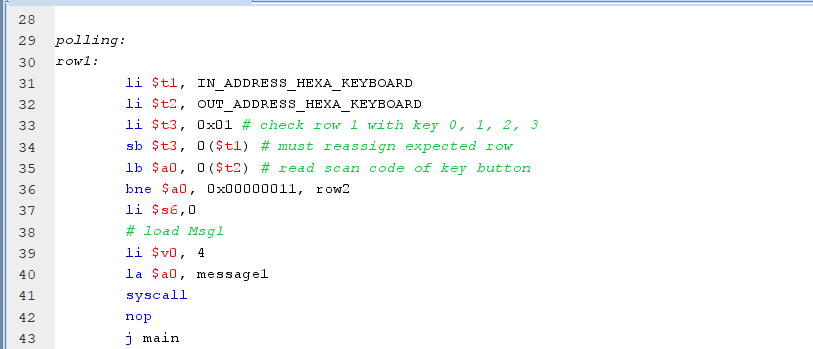
B2:

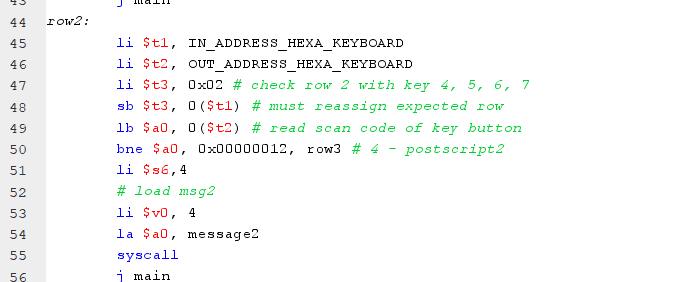
Kiểm tra từng cột từ 1 đến 3 xem có xuất hiện số 0 hoặc 4 hoặc 8

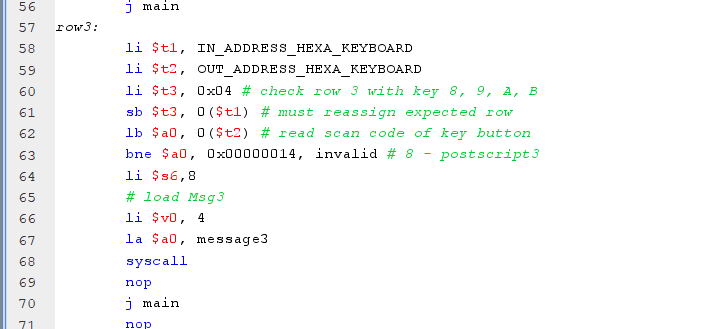
Nếu có thì sẽ nhảy đến j main

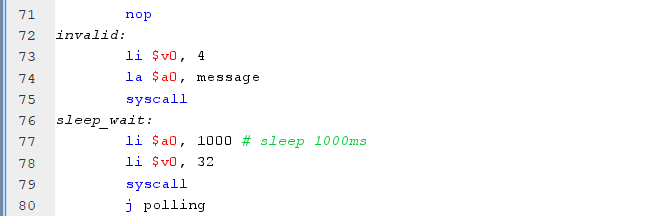
Nếu không sẽ tiếp tục in ra dòng chữ đầu vào không hợp lệ, và sẽ nhập lại cho đến khi nào nhập 1 trong 3 số kia thì thôi và ở đây khi đầu vào không hợp lệ sẽ in ra các dòng chữ cách nhau 1000ms

Và lưu trữ số được bấm trong $s6



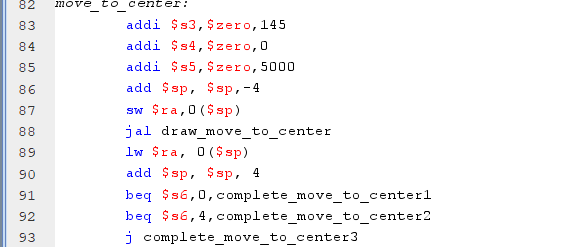


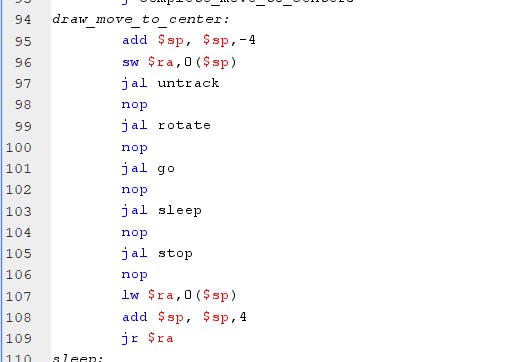




B3:

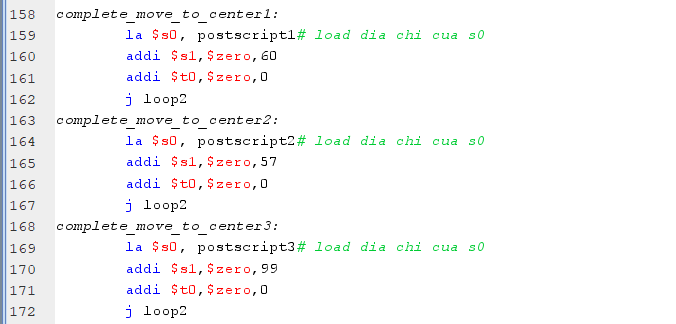
Sau khi chọn xong đến bước di chuyển ra giữa với các tham số s3 lưu góc dịch chuyển,s4 là 0 tương đương với không để lại dấu vết, s5 là di chuyển với thời gian 5000 giây. Để thực hiện gọi các hàm untrack không để lại dấu vết,rotate dịch chuyển với góc bao nhiêu, hàm go cho nó di chuyển , hàm sleep cho di chuyển 5000ms, và sau đó dừng lại với lệnh stop. Và với tham số $s6 được lưu ở trên cho biết poistrait nào được chọn sau khi vẽ xong sẽ đến việc thực hiện hình đó





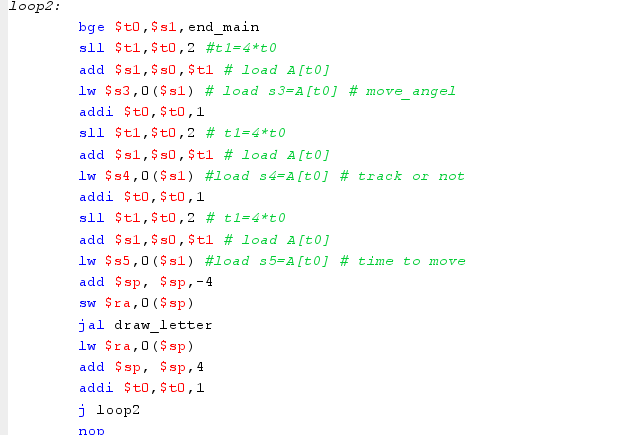
B4:

Khai báo để trước khi chuẩn bị vẽ hình



B5:

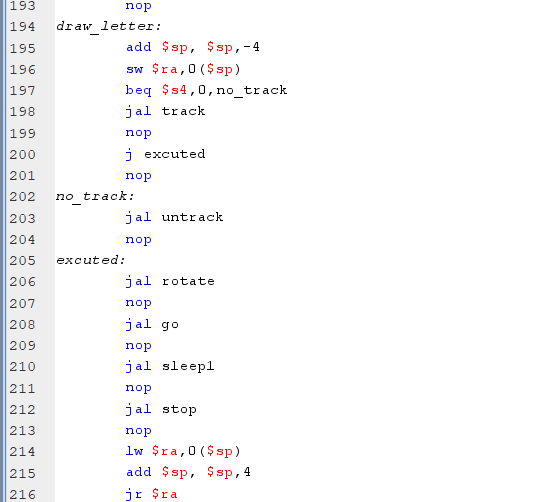
Vòng lặp để thực hiện vẽ hình , và trong mỗi vòng lặp sẽ lưu 3 tham số góc chuyển động, để lại dấu vết hay không, và di chuyển với thời gian bao nhiêu



B6:

Vẽ chữ

Với việc lưu trữ s4 là giá trị góc ,s5 là để lại dấu vết hay không , s6 để là thời gian di chuyển thì gọi theo thứ tự này t sẽ in được các chữ theo thứ tự đã lưu trong poistrait. Và để nó giữ lại các vết cũ hàm sleep1 gọi lại lần lượt hai hàm untrack và track để có thể giữ lại nét cũ



Mã nguồn:

.data

.eqv HEADING 0xffff8010 # integer Anangle between 0 and 359

.eqv MOVING 0Xffff8050# whether or not to move

.eqv LEAVETRACK 0Xffff8020 # whether or not to leave a track

.eqv IN\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0012

.eqv OUT\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0014

postscript1: .word 110,1,2400, 140,1,2400, 170,1,2400, 180,1,2400, 190,1,2400, 220,1,2400, 250,1,2400, 0,1,12600, 90,0,10000,

250,1,3500, 200,1,4500, 160,1,4500, 110,1,3500, 90,0,6000,

270,1,3000, 0,1,6000, 90,1,3000, 270,0,3000, 0,1,6000, 90,1,3000

postscript2: .word 180,1,12000, 0,0,6000, 90,1,5000, 180,1,6000, 0,1,12000 , 90,0,5000

270,1,3000, 180,1,6000, 90,1,3000, 270,0,3000, 180,1,6000, 90,1,3000, 90,0,5000,

0,1,6000, 135,1,7000, 315,0,7000, 45,1,7000, 225,0,7000, 0,1,6000

postscript3: .word

90,1,1500, 270,0,1500, 240,1,1000, 180,1,5000, 120,1,1000, 90,1,1500,

60,1,1000, 0,1,5000, 300,1,1000, 90,0,2500, 180,1,6000, 0,0,6000, 120,1,1500, 60,1,1500, 180,1,6000,

90,0,2000, 0,0,500, 0,1,5000, 60,1,1000, 90,1,1500, 120,1,1000, 180,1,1500, 0,0,1500, 300,0,1000,

270,0,1500, 240,0,1000, 180,0,5000, 120,1,1000, 90,1,1500, 60,1,1000, 0,1,1500, 270,0,1000, 90,1,2000

message: .asciiz "Invalid input for postscript!\n"

message1: .asciiz "You choose to print DCE!\n"

enter: .asciiz "\n"

message2: .asciiz "You choose to print HEK!\n"

message3: .asciiz "You choose to print OMG!\n"

space: .asciiz " "

.eqv IN\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0012

.eqv OUT\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0014

# whether or not to leave a track

.text

polling:

row1:

li $t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li $t2, OUT\_ADDRESS\_HEXA\_KEYBOARD

li $t3, 0x01 # check row 1 with key 0, 1, 2, 3

sb $t3, 0($t1) # must reassign expected row

lb $a0, 0($t2) # read scan code of key button

bne $a0, 0x00000011, row2

li $s6,0

# load Msg1

li $v0, 4

la $a0, message1

syscall

nop

j main

row2:

li $t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li $t2, OUT\_ADDRESS\_HEXA\_KEYBOARD

li $t3, 0x02 # check row 2 with key 4, 5, 6, 7

sb $t3, 0($t1) # must reassign expected row

lb $a0, 0($t2) # read scan code of key button

bne $a0, 0x00000012, row3 # 4 - postscript2

li $s6,4

# load msg2

li $v0, 4

la $a0, message2

syscall

j main

row3:

li $t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li $t2, OUT\_ADDRESS\_HEXA\_KEYBOARD

li $t3, 0x04 # check row 3 with key 8, 9, A, B

sb $t3, 0($t1) # must reassign expected row

lb $a0, 0($t2) # read scan code of key button

bne $a0, 0x00000014, invalid # 8 - postscript3

li $s6,8

# load Msg3

li $v0, 4

la $a0, message3

syscall

nop

j main

nop

invalid:

li $v0, 4

la $a0, message

syscall

sleep\_wait:

li $a0, 1000 # sleep 1000ms

li $v0, 32

syscall

j polling

main:

move\_to\_center:

addi $s3,$zero,145

addi $s4,$zero,0

addi $s5,$zero,5000

add $sp, $sp,-4

sw $ra,0($sp)

jal draw\_move\_to\_center

lw $ra, 0($sp)

add $sp, $sp, 4

beq $s6,0,complete\_move\_to\_center1

beq $s6,4,complete\_move\_to\_center2

j complete\_move\_to\_center3

draw\_move\_to\_center:

add $sp, $sp,-4

sw $ra,0($sp)

jal untrack

nop

jal rotate

nop

jal go

nop

jal sleep

nop

jal stop

nop

lw $ra,0($sp)

add $sp, $sp,4

jr $ra

sleep:

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

addi $a0,$s5,0

syscall

nop

jr $ra

nop

sleep1:

add $sp, $sp,-4

sw $ra,0($sp)

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

addi $a0,$s5,0

syscall

nop

jal untrack # keep old track

nop

jal track # and draw new track line

nop

lw $ra,0($sp)

add $sp, $sp,4

jr $ra

go: li $at, MOVING # change MOVING port

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

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

nop

jr $ra

nop

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

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

nop

jr $ra

nop

track: li $at, LEAVETRACK # change LEAVETRACK port

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

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

nop

jr $ra

nop

untrack:li $at, LEAVETRACK # change LEAVETRACK port to 0

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

nop

jr $ra

nop

rotate: li $at, HEADING # change HEADING port

sw $s3, 0($at) # to rotate robot

nop

jr $ra

nop

complete\_move\_to\_center1:

la $s0, postscript1# load dia chi cua s0

addi $s1,$zero,60

addi $t0,$zero,0

j loop2

complete\_move\_to\_center2:

la $s0, postscript2# load dia chi cua s0

addi $s1,$zero,57

addi $t0,$zero,0

j loop2

complete\_move\_to\_center3:

la $s0, postscript3# load dia chi cua s0

addi $s1,$zero,99

addi $t0,$zero,0

j loop2

loop2:

bge $t0,$s1,end\_main

sll $t1,$t0,2 #t1=4\*t0

add $s1,$s0,$t1 # load A[t0]

lw $s3,0($s1) # load s3=A[t0] # move\_angel

addi $t0,$t0,1

sll $t1,$t0,2 # t1=4\*t0

add $s1,$s0,$t1 # load A[t0]

lw $s4,0($s1) #load s4=A[t0] # track or not

addi $t0,$t0,1

sll $t1,$t0,2 # t1=4\*t0

add $s1,$s0,$t1 # load A[t0]

lw $s5,0($s1) #load s5=A[t0] # time to move

add $sp, $sp,-4

sw $ra,0($sp)

jal draw\_letter

lw $ra,0($sp)

add $sp, $sp,4

addi $t0,$t0,1

j loop2

nop

draw\_letter:

add $sp, $sp,-4

sw $ra,0($sp)

beq $s4,0,no\_track

jal track

nop

j excuted

nop

no\_track:

jal untrack

nop

excuted:

jal rotate

nop

jal go

nop

jal sleep1

nop

jal stop

nop

lw $ra,0($sp)

add $sp, $sp,4

jr $ra

end\_main:

li $v0,10

syscall

Kết quả:

