BÁO CÁO TUẦN 10(2)

Bài 1:

a, Tam giác đều:

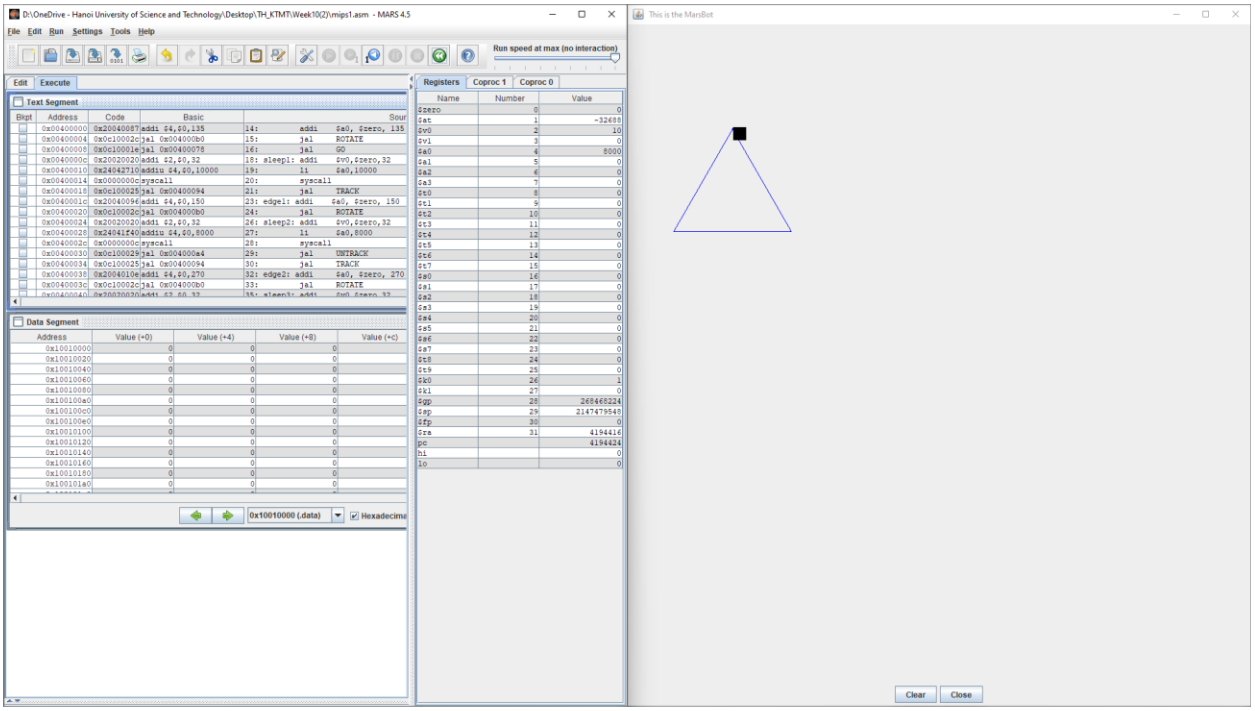
- *Chương trình:*

```
.eqv  HEADING    0xffff8010       # Integer: An angle between 0 and 359
                                  # 0 : North (up)
                                  # 90: East (right)
                                  # 180: South (down)
                                  # 270: West  (left)
.eqv  MOVING     0xffff8050       # Boolean: whether or not to move
.eqv  LEAVETRACK 0xffff8020       # Boolean (0 or non-0):
                                  # whether or not to leave a track
.eqv  WHEREX     0xffff8030       # Integer: Current x-location of MarsBot
.eqv  WHEREY     0xffff8040       # Integer: Current y-location of MarsBot
.text
main:
            addi      $a0, $zero, 135 # Marsbot rotates 135* and start running
            jal       ROTATE
            jal       GO
sleep1: addi   $v0,$zero,32    # Keep running by sleeping in1000 ms
            li        $a0,10000
            syscall
            jal    TRACK              # And draw new track line
edge1: addi   $a0, $zero, 150    # Marsbot rotates 150*
            jal    ROTATE

sleep2: addi      $v0,$zero,32    # Keep running by sleeping in 2000 ms
            li    $a0,8000
            syscall
```

```
        jal    UNTRACK       # Keep old track

        jal    TRACK         # And draw new track line

edge2: addi      $a0, $zero, 270 # Marsbotrotates 270*

        jal       ROTATE

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

        li     $a0,8000

        syscall

        jal    UNTRACK       # Keep old track

        jal    TRACK         # And draw new track line

edge3:  addi      $a0, $zero, 30   # Marsbot rotates 30*

        jal       ROTATE

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

        li     $a0,8000

        syscall

        jal    UNTRACK       # Keep old track

        jal       STOP

        li        $v0, 10

        syscall

end_main:

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

# GO procedure, to start running

# param[in]    none

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

GO:    li         $at, MOVING    # change MOVING port

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

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

        jr        $ra

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

# STOP procedure, to stop running
# param[in]    none
#---------------------------------------------------------
STOP:   li          $at, MOVING     # change MOVING port to 0
        sb          $zero, 0($at)   # to stop
        jr          $ra
#---------------------------------------------------------
# TRACK procedure, to start drawing line
# param[in]    none
#---------------------------------------------------------
TRACK:  li          $at, LEAVETRACK # change LEAVETRACK port
        addi        $k0, $zero,1    # to  logic 1,
        sb          $k0, 0($at)     # to start tracking
        jr          $ra
#---------------------------------------------------------
# UNTRACK procedure, to stop drawing line\
# param[in]    none
#---------------------------------------------------------
UNTRACK:li          $at, LEAVETRACK # change LEAVETRACK port to 0
        sb          $zero, 0($at)   # to stop drawing tail
        jr          $ra
#---------------------------------------------------------
# ROTATE procedure, to rotate the robot
# param[in]    $a0, An angle between 0 and 359
#                 0 : North (up)
#                 90: East  (right)
#                180: South (down)
#                270: West  (left)
#---------------------------------------------------------
ROTATE: li          $at, HEADING    # change HEADING port
```

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

- *Kết quả:*



D:\OneDrive - Hanoi University of Science and Technology\Desktop\TH_KTMT\Week10(2)\mips1.asm - MARS 4.5

Text Segment

Bkpt	Address	Code	Basic	Sour
□	0x00400000	0x20040087	addi $4,$0,135	14: addi $a0, $zero, 135
□	0x00400004	0x0c10002c	jal 0x004000b0	15: jal ROTATE
□	0x00400008	0x0c10001e	jal 0x00400078	16: jal GO
□	0x0040000c	0x20020020	addi $2,$0,32	18: sleep1: addi $v0,$zero,32
□	0x00400010	0x24042710	addiu $4,$0,10000	19: li $a0,10000
□	0x00400014	0x0000000c	syscall	20: syscall
□	0x00400018	0x0c100025	jal 0x00400094	21: jal TRACK
□	0x0040001c	0x20040096	addi $4,$0,150	23: edge1: addi $a0, $zero, 150
□	0x00400020	0x0c10002c	jal 0x004000b0	24: jal ROTATE
□	0x00400024	0x20020020	addi $2,$0,32	26: sleep2: addi $v0,$zero,32
□	0x00400028	0x24041f40	addiu $4,$0,8000	27: li $a0,8000
□	0x0040002c	0x0000000c	syscall	28: syscall
□	0x00400030	0x0c100029	jal 0x004000a4	29: jal UNTRACK
□	0x00400034	0x0c100025	jal 0x00400094	30: jal TRACK
□	0x00400038	0x2004010e	addi $4,$0,270	32: edge2: addi $a0, $zero, 270
□	0x0040003c	0x0c10002c	jal 0x004000b0	33: jal ROTATE
□	0x00400040	0x20020020	addi $2,$0,32	35: sleep3: addi $v0,$zero,32

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)
0x10010000	0	0	0	
0x10010020	0	0	0	
0x10010040	0	0	0	
0x10010060	0	0	0	
0x10010080	0	0	0	
0x100100a0	0	0	0	
0x100100c0	0	0	0	
0x100100e0	0	0	0	
0x10010100	0	0	0	
0x10010120	0	0	0	
0x10010140	0	0	0	
0x10010160	0	0	0	
0x10010180	0	0	0	
0x100101a0	0	0	0	

0x10010000 (.data) ☑ Hexadecima

Name	Number	Value
$zero	0	0
$at	1	-32688
$v0	2	10
$v1	3	0
$a0	4	8000
$a1	5	0
$a2	6	0
$a3	7	0
$t0	8	0
$t1	9	0
$t2	10	0
$t3	11	0
$t4	12	0
$t5	13	0
$t6	14	0
$t7	15	0
$s0	16	0
$s1	17	0
$s2	18	0
$s3	19	0
$s4	20	0
$s5	21	0
$s6	22	0
$s7	23	0
$t8	24	0
$t9	25	0
$k0	26	1
$k1	27	0
$gp	28	268468224
$sp	29	2147479548
$fp	30	0
$ra	31	4194416
pc		4194424
hi		0
lo		0

b, Hình vuông:

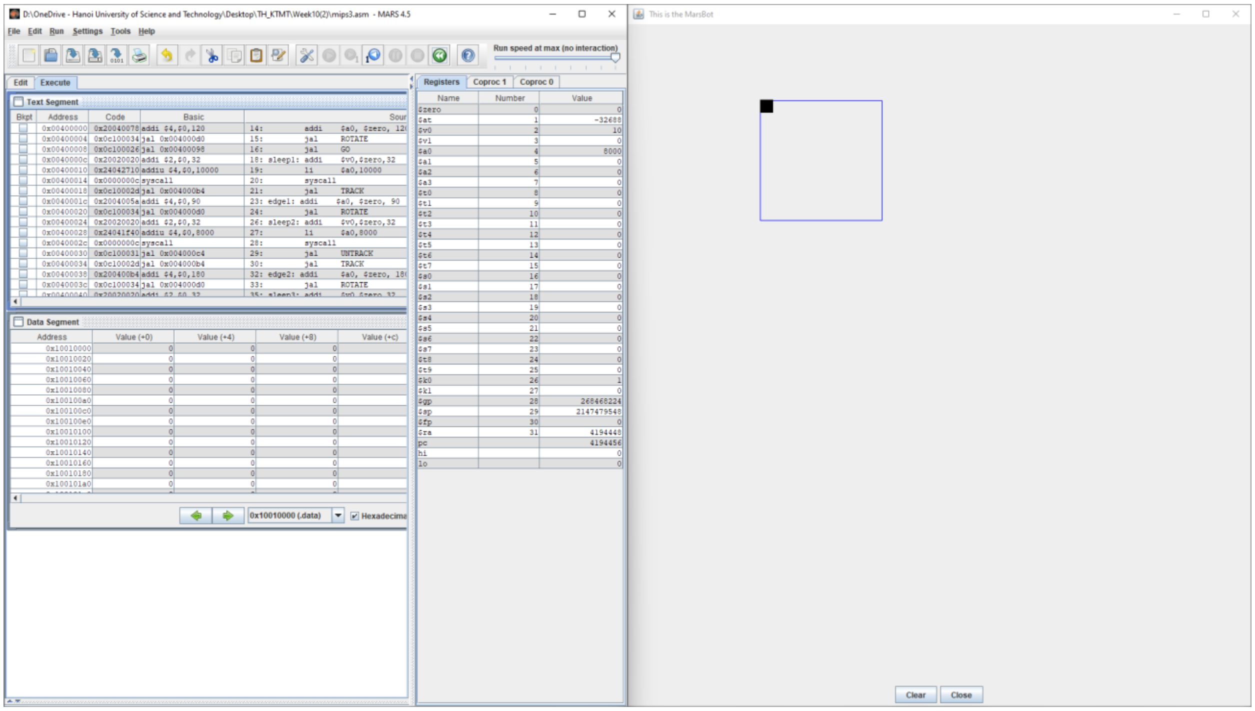
- *Chương trình:*

```
.eqv  HEADING    0xffff8010        # Integer: An angle between 0 and 359
                                   # 0 : North (up)
                                   # 90: East (right)
                                   # 180: South (down)
                                   # 270: West  (left)
.eqv  MOVING     0xffff8050        # Boolean: whether or not to move
.eqv  LEAVETRACK 0xffff8020        # Boolean (0 or non-0):
                                   # whether or not to leave a track
.eqv  WHEREX     0xffff8030        # Integer: Current x-location of MarsBot
.eqv  WHEREY     0xffff8040        # Integer: Current y-location of MarsBot
.text
main:
        addi    $a0, $zero, 120 # Marsbot rotates 135* and start running
        jal     ROTATE
        jal     GO
sleep1: addi    $v0,$zero,32    # Keep running by sleeping in1000 ms
        li      $a0,10000
        syscall
        jal   TRACK                # And draw new track line
edge1: addi   $a0, $zero, 90       # Marsbot rotates 90*
        jal   ROTATE
sleep2: addi     $v0,$zero,32      # Keep running by sleeping in 2000 ms
        li    $a0,8000
        syscall
        jal   UNTRACK        # Keep old track
        jal   TRACK        # And draw new track line
```

```
edge2: addi      $a0, $zero, 180 # Marsbotrotates 180*
            jal        ROTATE
sleep3: addi   $v0,$zero,32    # Keep running by sleeping in 1000 ms
            li     $a0,8000
            syscall
            jal    UNTRACK        # Keep old track
            jal    TRACK          # And draw new track line
edge3:  addi      $a0, $zero, 270 # Marsbot rotates 270*
            jal        ROTATE
sleep4: addi   $v0,$zero,32    # Keep running by sleeping in 2000 ms
            li     $a0,8000
            syscall
            jal    UNTRACK        # Keep old track
            jal        TRACK              # And draw new track line
 edge4:  addi     $a0, $zero, 0     # Marsbot rotates 0*
            jal        ROTATE
sleep5: addi   $v0,$zero,32    # Keep running by sleeping in 2000 ms
            li     $a0,8000
            syscall
            jal    UNTRACK        # Keep old track
            jal        TRACK              # And draw new track line
            jal        STOP
            li          $v0, 10
            syscall
end_main:
#-----------------------------------------------------------
# GO procedure, to start running
# param[in]    none
#-----------------------------------------------------------
```

```
GO:     li          $at, MOVING     # change MOVING port
        addi        $k0, $zero,1    # to  logic 1,
        sb          $k0, 0($at)     # to start running
        jr          $ra
#-----------------------------------------------------------
# STOP procedure, to stop running
# param[in]    none
#-----------------------------------------------------------
STOP:   li          $at, MOVING     # change MOVING port to 0
        sb          $zero, 0($at)   # to stop
        jr          $ra
#-----------------------------------------------------------
# TRACK procedure, to start drawing line
# param[in]    none
#-----------------------------------------------------------
TRACK:  li          $at, LEAVETRACK # change LEAVETRACK port
        addi        $k0, $zero,1    # to  logic 1,
        sb          $k0, 0($at)     # to start tracking
        jr          $ra
#-----------------------------------------------------------
# UNTRACK procedure, to stop drawing line\
# param[in]    none
#-----------------------------------------------------------
UNTRACK:li          $at, LEAVETRACK # change LEAVETRACK port to 0
        sb          $zero, 0($at)   # to stop drawing tail
        jr          $ra
#-----------------------------------------------------------
# ROTATE procedure, to rotate the robot
# param[in]    $a0, An angle between 0 and 359
#               0 : North (up)

#               90: East  (right)

#              180: South (down)

#              270: West  (left)

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

ROTATE: li         $at, HEADING    # change HEADING port

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

            jr         $ra
```

- *Kết quả:*


D:\OneDrive - Hanoi University of Science and Technology\Desktop\TH_KTMT\Week10(2)\mips3.asm - MARS 4.5
File Edit Run Settings Tools Help
Run speed at max (no interaction)
Edit Execute
Registers Coproc 1 Coproc 0
This is the MarsBot
Clear
Close

Text Segment

Bkpt	Address	Code	Basic	Sour
□	0x00400000	0x20040078	addi $4,$0,120	14: addi $a0, $zero, 12
□	0x00400004	0x0c100034	jal 0x004000d0	15: jal ROTATE
□	0x00400008	0x0c100026	jal 0x00400098	16: jal GO
□	0x0040000c	0x20020020	addi $2,$0,32	18: sleep1: addi $v0,$zero,32
□	0x00400010	0x24042710	addiu $4,$0,10000	19: li $a0,10000
□	0x00400014	0x0000000c	syscall	20: syscall
□	0x00400018	0x0c10002d	jal 0x004000b4	21: jal TRACK
□	0x0040001c	0x2004005a	addi $4,$0,90	23: edge1: addi $a0, $zero, 90
□	0x00400020	0x0c100034	jal 0x004000d0	24: jal ROTATE
□	0x00400024	0x20020020	addi $2,$0,32	26: sleep2: addi $v0,$zero,32
□	0x00400028	0x24041f40	addiu $4,$0,8000	27: li $a0,8000
□	0x0040002c	0x0000000c	syscall	28: syscall
□	0x00400030	0x0c100031	jal 0x004000c4	29: jal UNTRACK
□	0x00400034	0x0c10002d	jal 0x004000b4	30: jal TRACK
□	0x00400038	0x200400b4	addi $4,$0,180	32: edge2: addi $a0, $zero, 18
□	0x0040003c	0x0c100034	jal 0x004000d0	33: jal ROTATE
□	0x00400040	0x20020020	addi $2,$0,32	35: sleep3: addi $v0,$zero,32

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)
0x10010000	0	0	0	
0x10010020	0	0	0	
0x10010040	0	0	0	
0x10010060	0	0	0	
0x10010080	0	0	0	
0x100100a0	0	0	0	
0x100100c0	0	0	0	
0x100100e0	0	0	0	
0x10010100	0	0	0	
0x10010120	0	0	0	
0x10010140	0	0	0	
0x10010160	0	0	0	
0x10010180	0	0	0	
0x100101a0	0	0	0	

0x10010000 (.data) ☑ Hexadecima

Name	Number	Value
$zero	0	0
$at	1	-32688
$v0	2	10
$v1	3	0
$a0	4	8000
$a1	5	0
$a2	6	0
$a3	7	0
$t0	8	0
$t1	9	0
$t2	10	0
$t3	11	0
$t4	12	0
$t5	13	0
$t6	14	0
$t7	15	0
$s0	16	0
$s1	17	0
$s2	18	0
$s3	19	0
$s4	20	0
$s5	21	0
$s6	22	0
$s7	23	0
$t8	24	0
$t9	25	0
$k0	26	1
$k1	27	0
$gp	28	268468224
$sp	29	2147479548
$fp	30	0
$ra	31	4194448
pc		4194456
hi		0
lo		0

C, Hình sao

- *Chương trình:*

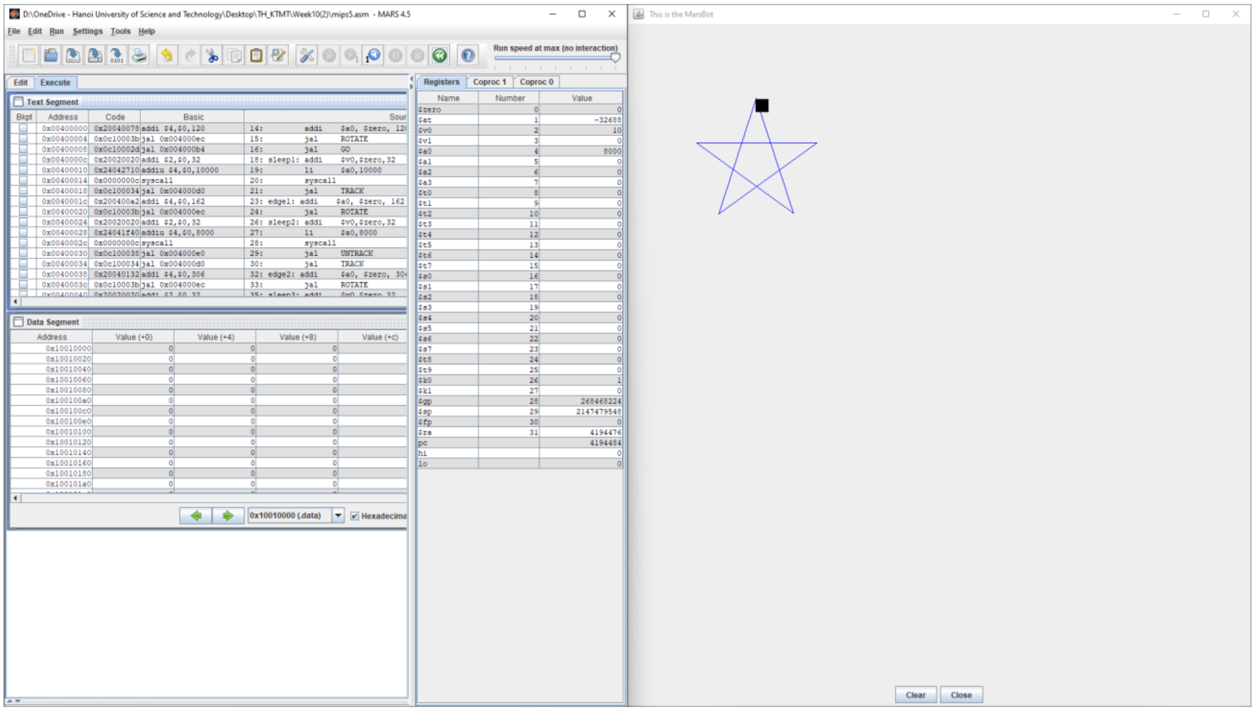
```
.eqv  HEADING    0xffff8010        # Integer: An angle between 0 and 359
                                   # 0 : North (up)
                                   # 90: East (right)
                                   # 180: South (down)
                                   # 270: West  (left)
.eqv  MOVING     0xffff8050        # Boolean: whether or not to move
.eqv  LEAVETRACK 0xffff8020        # Boolean (0 or non-0):
                                   # whether or not to leave a track
.eqv  WHEREX     0xffff8030        # Integer: Current x-location of MarsBot
.eqv  WHEREY     0xffff8040        # Integer: Current y-location of MarsBot
.text
main:
        addi     $a0, $zero, 120 # Marsbot rotates 135* and start running
        jal      ROTATE
        jal      GO
sleep1: addi   $v0,$zero,32    # Keep running by sleeping in1000 ms
        li       $a0,10000
        syscall
        jal    TRACK               # And draw new track line
edge1: addi    $a0, $zero, 162     # Marsbot rotates 162*
        jal    ROTATE
sleep2: addi     $v0,$zero,32      # Keep running by sleeping in 2000 ms
        li     $a0,8000
        syscall
        jal    UNTRACK       # Keep old track
        jal    TRACK       # And draw new track line
edge2: addi      $a0, $zero, 306 # Marsbotrotates 306*
```

```
        jal     ROTATE
sleep3: addi   $v0,$zero,32   # Keep running by sleeping in 1000 ms
        li    $a0,8000
        syscall
        jal   UNTRACK      # Keep old track
        jal   TRACK      # And draw new track line
edge3:  addi    $a0, $zero, 90 # Marsbot rotates 90*
        jal     ROTATE
sleep4: addi   $v0,$zero,32   # Keep running by sleeping in 2000 ms
        li    $a0,8000
        syscall
        jal   UNTRACK      # Keep old track
        jal     TRACK           # And draw new track line
edge4:  addi    $a0, $zero, 234 # Marsbot rotates 234*
        jal     ROTATE
sleep5: addi   $v0,$zero,32   # Keep running by sleeping in 2000 ms
        li    $a0,8000
        syscall
        jal   UNTRACK      # Keep old track
        jal     TRACK           # And draw new track line
edge5:  addi    $a0, $zero, 18   # Marsbot rotates 18*
        jal     ROTATE
sleep6: addi   $v0,$zero,32   # Keep running by sleeping in 2000 ms
        li    $a0,8000
        syscall
        jal   UNTRACK      # Keep old track
        jal     TRACK           # And draw new track line
        jal     STOP
        li      $v0, 10
```

```
          syscall
end_main:
#-----------------------------------------------------------
# GO procedure, to start running
# param[in]    none
#-----------------------------------------------------------
GO:     li          $at, MOVING     # change MOVING port
          addi      $k0, $zero,1    # to  logic 1,
          sb        $k0, 0($at)     # to start running
          jr        $ra
#-----------------------------------------------------------
# STOP procedure, to stop running
# param[in]    none
#-----------------------------------------------------------
STOP:   li          $at, MOVING     # change MOVING port to 0
          sb        $zero, 0($at)   # to stop
          jr        $ra
#-----------------------------------------------------------
# TRACK procedure, to start drawing line
# param[in]    none
#-----------------------------------------------------------
TRACK:  li          $at, LEAVETRACK # change LEAVETRACK port
          addi      $k0, $zero,1    # to  logic 1,
          sb        $k0, 0($at)     # to start tracking
          jr        $ra
#-----------------------------------------------------------
# UNTRACK procedure, to stop drawing line\
# param[in]    none
#-----------------------------------------------------------
```

```
UNTRACK:li      $at, LEAVETRACK # change LEAVETRACK port to 0
        sb      $zero, 0($at)   # to stop drawing tail
        jr      $ra
#-----------------------------------------------------------
# ROTATE procedure, to rotate the robot
# param[in]    $a0, An angle between 0 and 359
#                0 : North (up)
#                90: East  (right)
#               180: South (down)
#               270: West  (left)
#-----------------------------------------------------------
ROTATE: li      $at, HEADING    # change HEADING port
        sw      $a0, 0($at)     # to rotate robot
        jr      $ra
```

- Kết quả:



D:\OneDrive - Hanoi University of Science and Technology\Desktop\TH_KTMT\Week10(2)\mips5.asm - MARS 4.5

Text Segment

Bkpt	Address	Code	Basic	Sour
□	0x00400000	0x20040078	addi $4,$0,120	14: addi $a0, $zero, 12
□	0x00400004	0x0c10003b	jal 0x004000ec	15: jal ROTATE
□	0x00400008	0x0c10002d	jal 0x004000b4	16: jal GO
□	0x0040000c	0x20020020	addi $2,$0,32	18: sleep1: addi $v0,$zero,32
□	0x00400010	0x24042710	addiu $4,$0,10000	19: li $a0,10000
□	0x00400014	0x0000000c	syscall	20: syscall
□	0x00400018	0x0c100034	jal 0x004000d0	21: jal TRACK
□	0x0040001c	0x200400a2	addi $4,$0,162	23: edge1: addi $a0, $zero, 162
□	0x00400020	0x0c10003b	jal 0x004000ec	24: jal ROTATE
□	0x00400024	0x20020020	addi $2,$0,32	26: sleep2: addi $v0,$zero,32
□	0x00400028	0x24041f40	addiu $4,$0,8000	27: li $a0,8000
□	0x0040002c	0x0000000c	syscall	28: syscall
□	0x00400030	0x0c100038	jal 0x004000e0	29: jal UNTRACK
□	0x00400034	0x0c100034	jal 0x004000d0	30: jal TRACK
□	0x00400038	0x20040132	addi $4,$0,306	32: edge2: addi $a0, $zero, 30
□	0x0040003c	0x0c10003b	jal 0x004000ec	33: jal ROTATE
□	0x00400040	0x20020020	addi $2,$0,32	35: sleep3: addi $v0,$zero,32

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)
0x10010000	0	0	0	
0x10010020	0	0	0	
0x10010040	0	0	0	
0x10010060	0	0	0	
0x10010080	0	0	0	
0x100100a0	0	0	0	
0x100100c0	0	0	0	
0x100100e0	0	0	0	
0x10010100	0	0	0	
0x10010120	0	0	0	
0x10010140	0	0	0	
0x10010160	0	0	0	
0x10010180	0	0	0	
0x100101a0	0	0	0	

0x10010000 (.data) ☑ Hexadecima

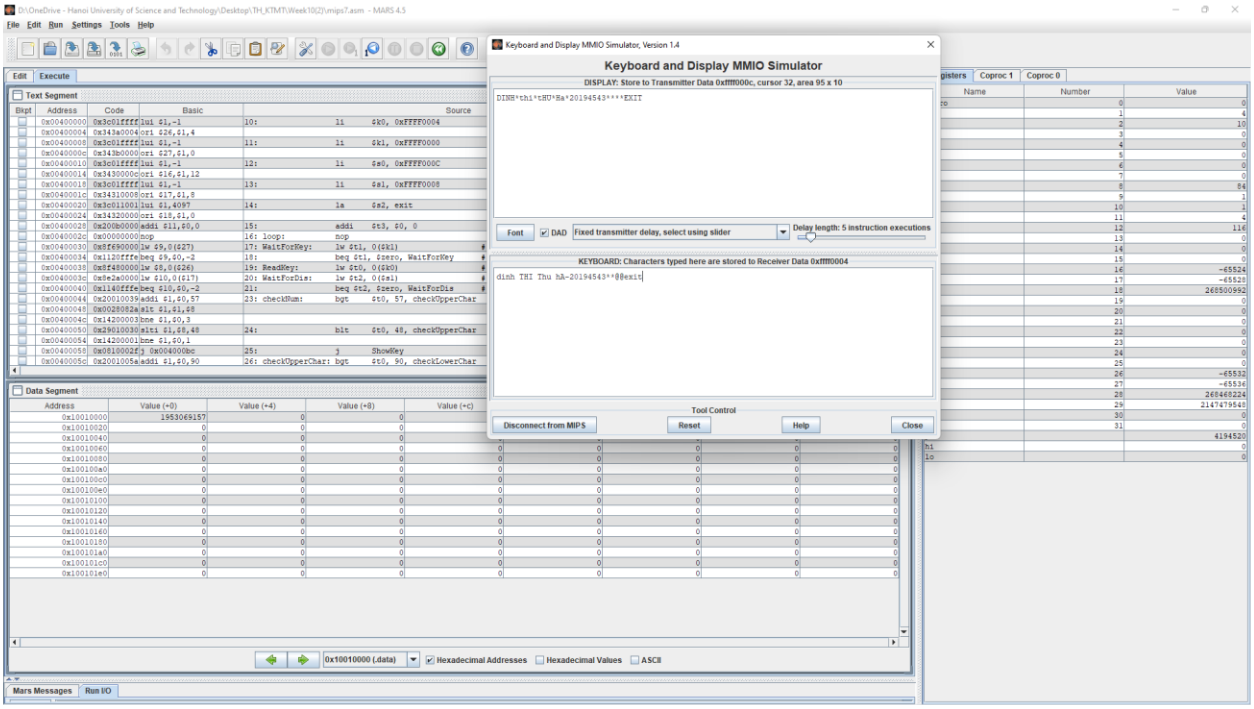
Name	Number	Value
$zero	0	0
$at	1	-32688
$v0	2	10
$v1	3	0
$a0	4	8000
$a1	5	0
$a2	6	0
$a3	7	0
$t0	8	0
$t1	9	0
$t2	10	0
$t3	11	0
$t4	12	0
$t5	13	0
$t6	14	0
$t7	15	0
$s0	16	0
$s1	17	0
$s2	18	0
$s3	19	0
$s4	20	0
$s5	21	0
$s6	22	0
$s7	23	0
$t8	24	0
$t9	25	0
$k0	26	1
$k1	27	0
$gp	28	268468224
$sp	29	2147479548
$fp	30	0
$ra	31	4194476
pc		4194484
hi		0
lo		0

Bài 2:

- Chương trình:

```
.eqv KEY_CODE   0xFFFF0004                  # ASCII code from keyboard, 1 byte
.eqv KEY_READY 0xFFFF0000                   # =1 if has a new keycode ?
# Auto clear after lw
.eqv DISPLAY_CODE 0xFFFF000C                # ASCII code to show, 1 byte
.eqv DISPLAY_READY 0xFFFF0008               # =1 if the display has already to do
# Auto clear after sw
.data
exit:           .asciiz "exit"
.text
                li      $k0, KEY_CODE
                li      $k1, KEY_READY
                li      $s0, DISPLAY_CODE
                li      $s1, DISPLAY_READY
                la      $s2, exit
                addi    $t3, $0, 0
loop:           nop
WaitForKey:     lw $t1, 0($k1)              # $t1 = [$k1] = KEY_READY
                beq $t1, $zero, WaitForKey  # if $t1 == 0 then Polling
ReadKey:        lw $t0, 0($k0)              # $t0 = [$k0] = KEY_CODE
WaitForDis:     lw $t2, 0($s1)              # $t2 = [$s1] = DISPLAY_READY
                beq $t2, $zero, WaitForDis  # if $t2 == 0 then Polling
Encrypt:        # change input key
checkNum:       bgt     $t0, 57, checkUpperChar
                blt     $t0, 48, checkUpperChar
                j       ShowKey
checkUpperChar: bgt     $t0, 90, checkLowerChar
                blt     $t0, 65, checkLowerChar
                addi    $t0, $t0, 32
                j       ShowKey
checkLowerChar: bgt     $t0, 122, Simbol
                blt     $t0, 97, Simbol
checkExit:      add     $t4, $t3, $s2
                lb      $t4, 0($t4)
                bne     $t4, $t0, reset
                addi    $t3, $t3, 1
                addi    $t0, $t0, -32
                j       ShowKey
reset:          addi    $t3, $0, 0
                beq     $t0, 'e', checkExit
                addi    $t0, $t0, -32
                j       ShowKey
Simbol:         addi    $t0, $0, 42
ShowKey:        sw      $t0, 0($s0)         # show key
                nop
                beq     $t3, 4, end
continue:       j       loop
end:
                addi    $v0, $0, 10
                syscall
```

- Kết quả:



D:\OneDrive - Hanoi University of Science and Technology\Desktop\TH_KTMT\Week10(2)\mips7.asm - MARS 4.5

Text Segment

Bkpt	Address	Code	Basic	Source
□	0x00400000	0x3c01ffff	lui $1,-1	10: li $k0, 0xFFFF0004
□	0x00400004	0x343a0004	ori $26,$1,4	
□	0x00400008	0x3c01ffff	lui $1,-1	11: li $k1, 0xFFFF0000
□	0x0040000c	0x343b0000	ori $27,$1,0	
□	0x00400010	0x3c01ffff	lui $1,-1	12: li $s0, 0xFFFF000C
□	0x00400014	0x3430000c	ori $16,$1,12	
□	0x00400018	0x3c01ffff	lui $1,-1	13: li $s1, 0xFFFF0008
□	0x0040001c	0x34310008	ori $17,$1,8	
□	0x00400020	0x3c011001	lui $1,4097	14: la $s2, exit
□	0x00400024	0x34320000	ori $18,$1,0	
□	0x00400028	0x200b0000	addi $11,$0,0	15: addi $t3, $0, 0
□	0x0040002c	0x00000000	nop	16: loop: nop
□	0x00400030	0x8f690000	lw $9,0($27)	17: WaitForKey: lw $t1, 0($k1) #
□	0x00400034	0x1120fffe	beq $9,$0,-2	18: beq $t1, $zero, WaitForKey #
□	0x00400038	0x8f480000	lw $8,0($26)	19: ReadKey: lw $t0, 0($k0) #
□	0x0040003c	0x8e2a0000	lw $10,0($17)	20: WaitForDis: lw $t2, 0($s1) #
□	0x00400040	0x1140fffe	beq $10,$0,-2	21: beq $t2, $zero, WaitForDis #
□	0x00400044	0x20010039	addi $1,$0,57	23: checkNum: bgt $t0, 57, checkUpperChar
□	0x00400048	0x0028082a	slt $1,$1,$8	
□	0x0040004c	0x14200003	bne $1,$0,3	
□	0x00400050	0x29010030	slti $1,$8,48	24: blt $t0, 48, checkUpperChar
□	0x00400054	0x14200001	bne $1,$0,1	
□	0x00400058	0x0810002f	j 0x004000bc	25: j ShowKey
□	0x0040005c	0x2001005a	addi $1,$0,90	26: checkUpperChar: bgt $t0, 90, checkLowerChar

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)				
0x10010000	1953069157	0	0					
0x10010020	0	0	0					
0x10010040	0	0	0					
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0
0x100100e0	0	0	0	0	0	0	0	0
0x10010100	0	0	0	0	0	0	0	0
0x10010120	0	0	0	0	0	0	0	0
0x10010140	0	0	0	0	0	0	0	0
0x10010160	0	0	0	0	0	0	0	0
0x10010180	0	0	0	0	0	0	0	0
0x100101a0	0	0	0	0	0	0	0	0
0x100101c0	0	0	0	0	0	0	0	0
0x100101e0	0	0	0	0	0	0	0	0

0x10010000 (.data) ☑ Hexadecimal Addresses □ Hexadecimal Values □ ASCII

Keyboard and Display MMIO Simulator, Version 1.4

Keyboard and Display MMIO Simulator

DISPLAY: Store to Transmitter Data 0xffff000c, cursor 32, area 95 x 10

```
DINH*thi*tHU*Ha*20194543****EXIT
```

Font ☑ DAD Fixed transmitter delay, select using slider — Delay length: 5 instruction executions

KEYBOARD: Characters typed here are stored to Receiver Data 0xffff0004

```
dinh THI Thu hA-20194543**@@exit
```

Tool Control

Name	Number	Value
ro	0	0
	1	4
	2	10
	3	0
	4	0
	5	0
	6	0
	7	0
	8	84
	9	1
	10	1
	11	4
	12	116
	13	0
	14	0
	15	0
	16	-65524
	17	-65528
	18	268500992
	19	0
	20	0
	21	0
	22	0
	23	0
	24	0
	25	0
	26	-65532
	27	-65536
	28	268468224
	29	2147479548
	30	0
	31	0
		4194520
hi		0
lo		0

Bài 3:

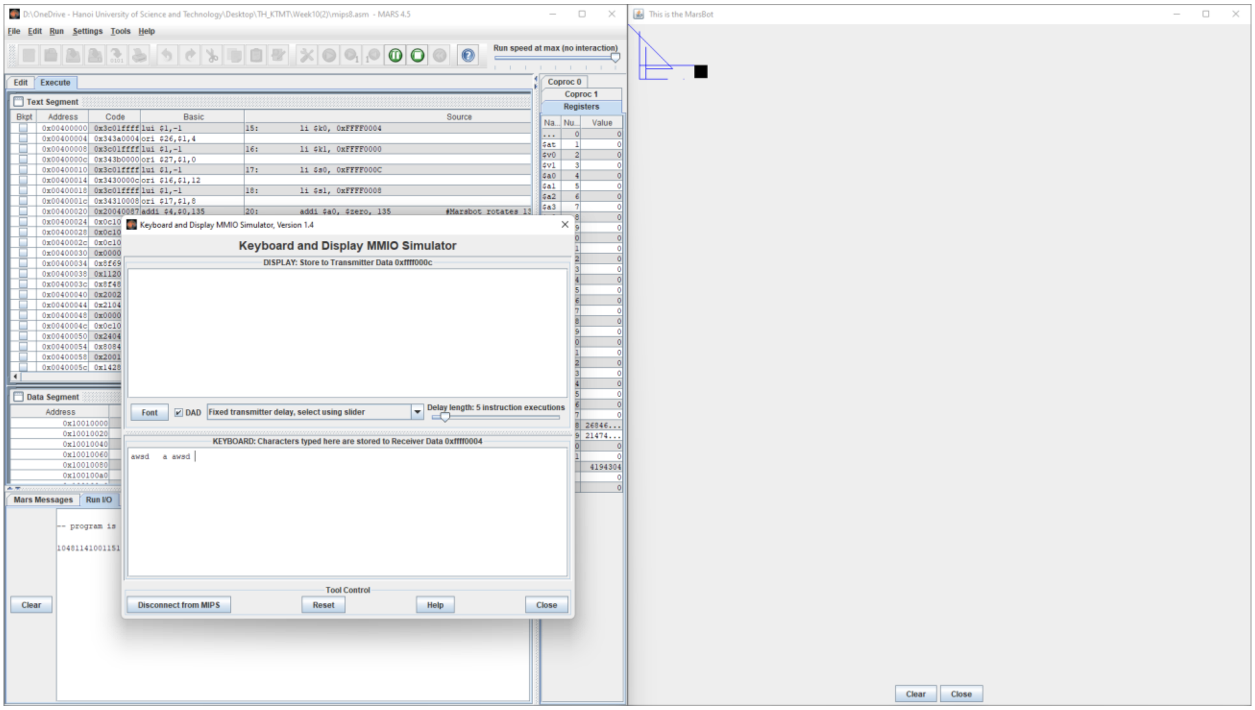
- Chương trình:

```
.eqv KEY_CODE 0xFFFF0004 # ASCII code from keyboard, 1 byte

.eqv KEY_READY 0xFFFF0000 # =1 if has a new keycode ?

# Auto clear after lw

.eqv DISPLAY_CODE 0xFFFF000C # ASCII code to show, 1 byte

.eqv DISPLAY_READY 0xFFFF0008 # =1 if the display has already to do

# Auto clear after sw

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

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

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

# whether or not to leave a track

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

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

.data

.text

        li $k0, KEY_CODE

        li $k1, KEY_READY

        li $s0, DISPLAY_CODE

        li $s1, DISPLAY_READY


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

        jal ROTATE

        jal TRACK                       #Start draw

        jal GO

loop: nop

WaitForKey: lw $t1, 0($k1) # $t1 = [$k1] = KEY_READY

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

ReadKey: lw $t0, 0($k0) # $t0 = [$k0] = KEY_CODE
```

```
                addi    $v0, $0, 1
                addi    $a0, $t0, 0
                syscall
Control:
                jal UNTRACK             # keep old track
                li      $a0, MOVING
                lb      $a0, 0($a0)
SPACE:  bne     $t0, ' ', UP
                bne     $a0, $0, stop
go:             jal     GO
                j       continue
stop:   jal     STOP
                j       continue
UP:             bne     $t0, 'w', DOWN
                addi $a0, $zero, 0
                j       rotate
DOWN:   bne     $t0, 's', LEFT
                addi $a0, $zero, 180
                j       rotate
LEFT:   bne     $t0, 'a', RIGHT
                addi $a0, $zero, 270
                j       rotate
RIGHT:  bne     $t0, 'd', continue
                addi $a0, $zero, 90
                j       rotate
rotate:
                jal ROTATE
                jal TRACK                       # and draw new track line
                j       continue
```

```
continue:       j loop
end_main:
        addi    $v0, $0, 10
        syscall

GO:     li $at, MOVING # change MOVING port
                addi $a0, $zero,1 # to logic 1,
                sb $a0, 0($at) # to start running
                jr $ra
STOP:   li $at, MOVING # change MOVING port to 0
                sb $zero, 0($at) # to stop
                jr $ra
TRACK:  li $at, LEAVETRACK # change LEAVETRACK port
                addi $a0, $zero,1 # to logic 1,
                sb $a0, 0($at) # to start tracking
                jr $ra
UNTRACK:li $at, LEAVETRACK # change LEAVETRACK port to 0
                sb $zero, 0($at) # to stop drawing tail
                jr $ra
ROTATE: li $at, HEADING # change HEADING port
                sw $a0, 0($at) # to rotate robot
                jr $ra
```

- Kết quả:



D:\OneDrive - Hanoi University of Science and Technology\Desktop\TH_KTMT\Week10(2)\mips8.asm - MARS 4.5

Bkpt	Address	Code	Basic	Source
□	0x00400000	0x3c01ffff	lui $1,-1	15: li $k0, 0xFFFF0004
□	0x00400004	0x343a0004	ori $26,$1,4	
□	0x00400008	0x3c01ffff	lui $1,-1	16: li $k1, 0xFFFF0000
□	0x0040000c	0x343b0000	ori $27,$1,0	
□	0x00400010	0x3c01ffff	lui $1,-1	17: li $s0, 0xFFFF000C
□	0x00400014	0x3430000c	ori $16,$1,12	
□	0x00400018	0x3c01ffff	lui $1,-1	18: li $s1, 0xFFFF0008
□	0x0040001c	0x34310008	ori $17,$1,8	
□	0x00400020	0x20040087	addi $4,$0,135	20: addi $a0, $zero, 135 #Marsbot rotates 13
□	0x00400024	0x0c10		
□	0x00400028	0x0c10		
□	0x0040002c	0x0c10		
□	0x00400030	0x0000		
□	0x00400034	0x8f69		
□	0x00400038	0x1120		
□	0x0040003c	0x8f48		
□	0x00400040	0x2002		
□	0x00400044	0x2104		
□	0x00400048	0x0000		
□	0x0040004c	0x0c10		
□	0x00400050	0x2404		
□	0x00400054	0x8084		
□	0x00400058	0x2001		
□	0x0040005c	0x1428		

Na...	Nu...	Value
...	0	0
$at	1	0
$v0	2	0
$v1	3	0
$a0	4	0
$a1	5	0
$a2	6	0
$a3	7	0
	8	0
	9	0
	0	0
	1	0
	2	0
	3	0
	4	0
	5	0
	6	0
	7	0
	8	0
	9	0
	0	0
	1	0
	2	0
	3	0
	4	0
	5	0
	6	0
	7	0
	8	26846...
	9	21474...
	0	0
	1	0
		4194304
		0
		0

Address
0x10010000
0x10010020
0x10010040
0x10010060
0x10010080
0x100100a0

```
-- program is

10481141001151
```

Keyboard and Display MMIO Simulator, Version 1.4

Keyboard and Display MMIO Simulator

DISPLAY: Store to Transmitter Data 0xffff000c

☑ DAD

Fixed transmitter delay, select using slider

Delay length: 5 instruction executions

KEYBOARD: Characters typed here are stored to Receiver Data 0xffff0004

```
awsd   a awsd
```

This is the MarsBot