# Laboratory Exercise 10 - Part 2

*MarsBot, Keyboard and Display MMIO Simulator*

1. **Assignment 1:** *Điều khiển marsbot di chuyển theo hình tam giác đều, hình vuông, hình ngôi sao 5 cánh*
   - Mã nguồn:

```
#Dieu khien MARSBOT di chuyen theo hình tam giác deu, hình vuông, hình ngôi sao 5 cánh

.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:
goSKEWDOWN: addi $a0, $zero, 135 # Marsbot rotates 180*
 jal ROTATE
 jal GO
 sleep: addi $v0,$zero,32 # Keep running by sleeping in 1000 ms
 li $a0,5000
 syscall
 jal TRACK # draw track line
 addi $a0, $zero, 90 # Marsbot rotates 90* and start running
 jal ROTATE
 jal GO
sleep1: addi $v0,$zero,32 # Keep running by sleeping in 1000 ms
 li $a0,5000
 syscall

 jal UNTRACK # keep old track
 jal TRACK # and draw new track line
goDOWN: addi $a0, $zero, 180 # Marsbot rotates 180*
 jal ROTATE

sleep2: addi $v0,$zero,32 # Keep running by sleeping in 2000 ms
 li $a0,5000
 syscall
 jal UNTRACK # keep old track
 jal TRACK # and draw new track line
goLEFT: addi $a0, $zero, 270 # Marsbot rotates 270*
 jal ROTATE

sleep3: addi $v0,$zero,32 # Keep running by sleeping in 1000 ms
 li $a0,5000
 syscall
 jal UNTRACK # keep old track
 jal TRACK # and draw new track line

goASKEW:
 addi $a0, $zero, 0 # Marsbot rotates 120*
 jal ROTATE

sleep4:
 addi $v0,$zero,32 # Keep running by sleeping in 2000 ms
 li $a0,5000
 syscall

 jal UNTRACK # keep old track
 #jal TRACK # and draw new track line

 goRIGHT:
 addi $a0, $zero, 90
 jal ROTATE
```

```
sleep5: addi $v0,$zero,32 # Keep running by sleeping in 1000 ms
li $a0,10000
syscall
jal UNTRACK # keep old track
jal TRACK

goTRIANGLE1:
addi $a0, $zero, 150
jal ROTATE

sleep6: addi $v0,$zero,32 # Keep running by sleeping in 1000 ms
li $a0,6000
syscall
jal UNTRACK # keep old track
jal TRACK # and draw new track line

goTIRIANGLE2:
addi $a0, $zero, 270
jal ROTATE

sleep7:
addi $v0,$zero,32 # Keep running by sleeping in 1000 ms
li $a0,6000
syscall

jal UNTRACK # keep old track
jal TRACK # and draw new track line

goTIRIANGLE3:
addi $a0, $zero, 30
jal ROTATE
```

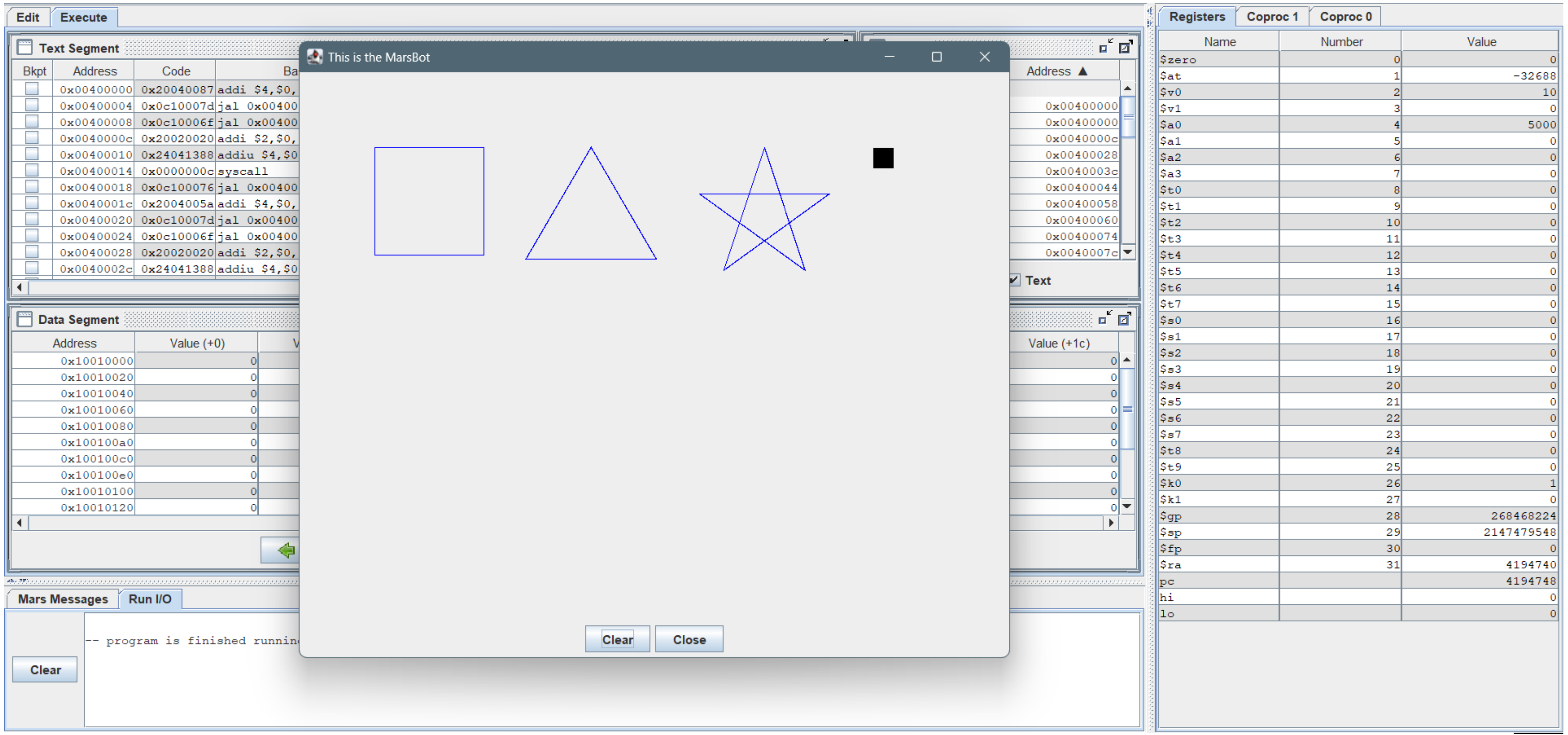
```
sleep8:
addi $v0,$zero,32 # Keep running by sleeping in 1000 ms
li $a0,6000
syscall
jal UNTRACK # keep old track
#jal TRACK # and draw new track line

goRIGHT2:
addi $a0, $zero, 90
jal ROTATE

sleep9:
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

goSTAR1:
addi $a0, $zero, 162
jal ROTATE

sleep10:
addi $v0,$zero,32 # Keep running by sleeping in 1000 ms
li $a0,6000
syscall
jal UNTRACK # keep old track
jal TRACK # and draw new track lin

goSTAR2:
addi $a0, $zero, 306
jal ROTATE
```

```
sleep11:
addi $v0,$zero,32 # Keep running by sleeping in 1000 ms
li $a0,6000
syscall
jal UNTRACK # keep old track
jal TRACK # and draw new track lin

goSTAR3:
addi $a0, $zero, 90
jal ROTATE

sleep12:
addi $v0,$zero,32 # Keep running by sleeping in 1000 ms
li $a0,6000
syscall
jal UNTRACK # keep old track
jal TRACK # and draw new track lin

goSTAR4:
addi $a0, $zero, 234
jal ROTATE

sleep13:
addi $v0,$zero,32 # Keep running by sleeping in 1000 ms
li $a0,6000
syscall
jal UNTRACK # keep old track
jal TRACK # and draw new track lin

goSTAR5:
addi $a0, $zero, 18
jal ROTATE
```

```
 sleep14:
 addi $v0,$zero,32 # Keep running by sleeping in 1000 ms
 li $a0,6000
 syscall


jal UNTRACK # keep old track
 #jal TRACK # and draw new track line

 goRIGHT3:
 addi $a0, $zero, 90
 jal ROTATE

 sleep15: addi $v0,$zero,32 # Keep running by sleeping in 1000 ms
 li $a0,5000
 syscall
 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:



| Bkpt | Address | Code | Ba |
|---|---|---|---|
| ☐ | 0x00400000 | 0x20040087 | addi $4,$0, |
| ☐ | 0x00400004 | 0x0c10007d | jal 0x00400 |
| ☐ | 0x00400008 | 0x0c10006f | jal 0x00400 |
| ☐ | 0x0040000c | 0x20020020 | addi $2,$0, |
| ☐ | 0x00400010 | 0x24041388 | addiu $4,$0 |
| ☐ | 0x00400014 | 0x0000000c | syscall |
| ☐ | 0x00400018 | 0x0c100076 | jal 0x00400 |
| ☐ | 0x0040001c | 0x2004005a | addi $4,$0, |
| ☐ | 0x00400020 | 0x0c10007d | jal 0x00400 |
| ☐ | 0x00400024 | 0x0c10006f | jal 0x00400 |
| ☐ | 0x00400028 | 0x20020020 | addi $2,$0, |
| ☐ | 0x0040002c | 0x24041388 | addiu $4,$0 |

| Address ▲ |
|---|
| 0x00400000 |
| 0x00400000 |
| 0x0040000c |
| 0x00400028 |
| 0x0040003c |
| 0x00400044 |
| 0x00400058 |
| 0x00400060 |
| 0x00400074 |
| 0x0040007c |

| Address | Value (+0) | V | Value (+1c) |
|---|---|---|---|
| 0x10010000 | 0 | | 0 |
| 0x10010020 | 0 | | 0 |
| 0x10010040 | 0 | | 0 |
| 0x10010060 | 0 | | 0 |
| 0x10010080 | 0 | | 0 |
| 0x100100a0 | 0 | | 0 |
| 0x100100c0 | 0 | | 0 |
| 0x100100e0 | 0 | | 0 |
| 0x10010100 | 0 | | 0 |
| 0x10010120 | 0 | | 0 |

```
-- program is finished runnin
```

| Name | Number | Value |
|---|---|---|
| $zero | 0 | 0 |
| $at | 1 | -32688 |
| $v0 | 2 | 10 |
| $v1 | 3 | 0 |
| $a0 | 4 | 5000 |
| $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 | 4194740 |
| pc | | 4194748 |
| hi | | 0 |
| lo | | 0 |

2. **Assignment 2:** *Nhập ký tự ở Keyboard và hiển thị ở Display: nhập ký tự thường => hiển thị ký tự hoa tương ứng, nhập ký tự hoa => hiển thị ký tự thường tương ứng, nhập ký tự số thì giữ nguyên, nhập ký tự khác => hiển thị ký tự *. Khi nhập chuỗi ký tự "exit" thì kết thúc chương trình.*
    - Mã nguồn:

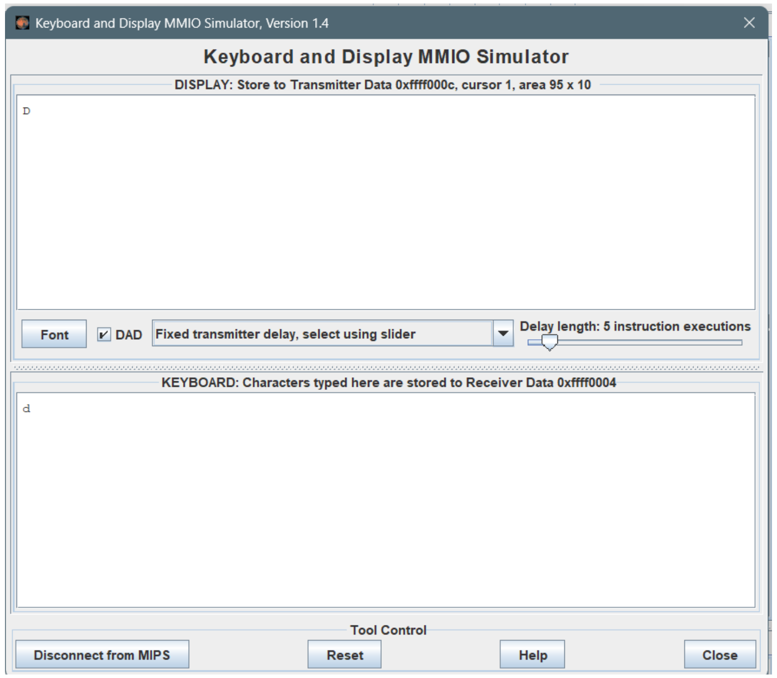
```
.eqv KEY_CODE 0xFFFF0004 # ASCII code from keyboard, 1 byte
.eqv KEY_READY 0xFFFF0000 # =1 if has a new keycode ?
# Auto clear after lw
.eqv DISPLAY_CODE 0xFFFF000C # ASCII code to show, 1 byte
.eqv DISPLAY_READY 0xFFFF0008 # =1 if the display has already to do
# Auto clear after sw
.text
```

```
li $k0, KEY_CODE
li $k1, KEY_READY
li $s0, DISPLAY_CODE
li $s1, DISPLAY_READY
li $s2, 0 # count the command characters entered correctly
loop: nop
beq $s2, 4, Exit
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


CheckCmd:
beq $s2, 0, FirstChar
beq $s2, 1, SecondChar
beq $s2, 2, ThirdChar
beq $s2, 3, FourthChar
FirstChar:
bne $t0, 'e', NotCmd
addi $s2, $s2, 1
j WaitForDis

SecondChar:
bne $t0, 'x', NotCmd
addi $s2, $s2, 1
j WaitForDis

ThirdChar:
bne $t0, 'i', NotCmd
addi $s2, $s2, 1
j WaitForDis

FourthChar:
```

```
bne $t0, 't', NotCmd
addi $s2, $s2, 1
j WaitForDis

NotCmd:
li $s2, 0

WaitForDis:
lw $t2, 0($s1) # $t2 = [$s1] = DISPLAY_READY
beq $t2, $zero, WaitForDis # if $t2 == 0 then Polling

ChangeChar:
ble $t0, '9', numberChar
ble $t0, 'Z', upperChar
ble $t0, 'z', lowerChar
j ChangeToStar

numberChar:
blt $t0, '0', ChangeToStar
j ShowKey

upperChar:
blt $t0, 'A', ChangeToStar
addi $t0, $t0, 32
j ShowKey

lowerChar:
blt $t0, 'a', ChangeToStar
addi $t0, $t0, -32
j ShowKey

ChangeToStar:
addi $t0, $zero, '*'
```

```
ShowKey: sw $t0, 0($s0) # show key
nop
j loop

Exit:
li $v0, 10
syscall
```

- Kết quả:



Keyboard and Display MMIO Simulator, Version 1.4

Keyboard and Display MMIO Simulator

DISPLAY: Store to Transmitter Data 0xffff000c, cursor 1, area 95 x 10

```
D
```

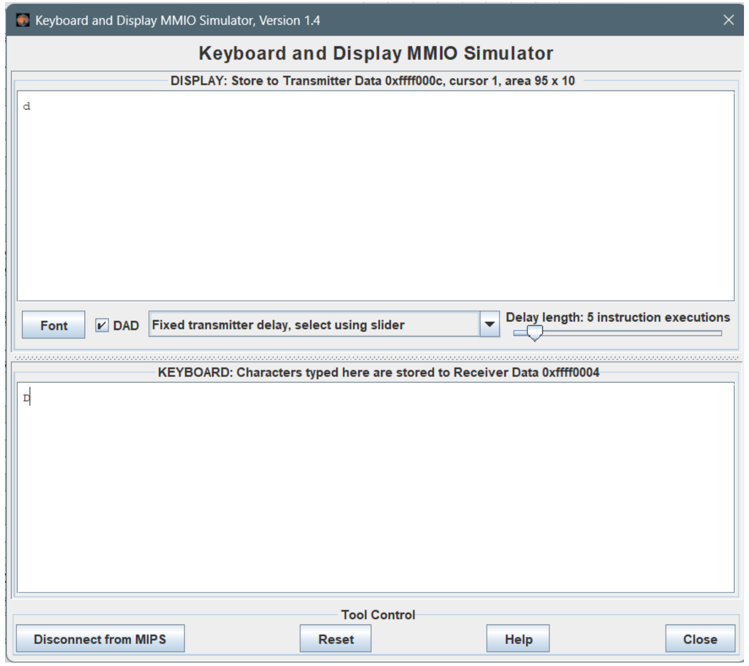
Fixed transmitter delay, select using slider

Delay length: 5 instruction executions

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

```
d
```

Tool Control



Keyboard and Display MMIO Simulator, Version 1.4

Keyboard and Display MMIO Simulator

DISPLAY: Store to Transmitter Data 0xffff000c, cursor 1, area 95 x 10

```
d
```

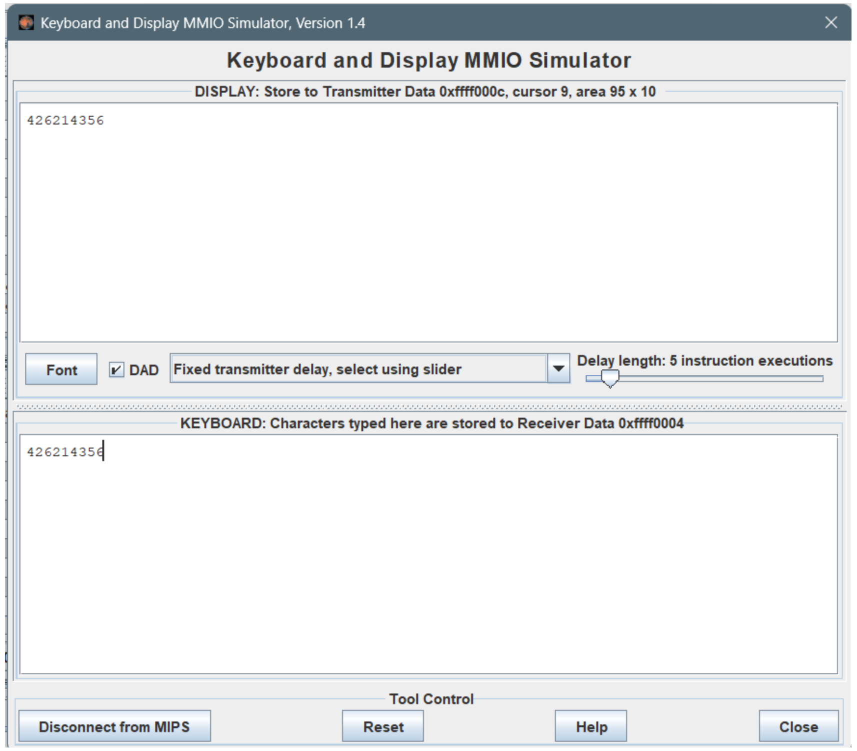
Fixed transmitter delay, select using slider

Delay length: 5 instruction executions

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

```
D
```

Tool Control


Keyboard and Display MMIO Simulator, Version 1.4

Keyboard and Display MMIO Simulator

DISPLAY: Store to Transmitter Data 0xffff000c, cursor 9, area 95 x 10

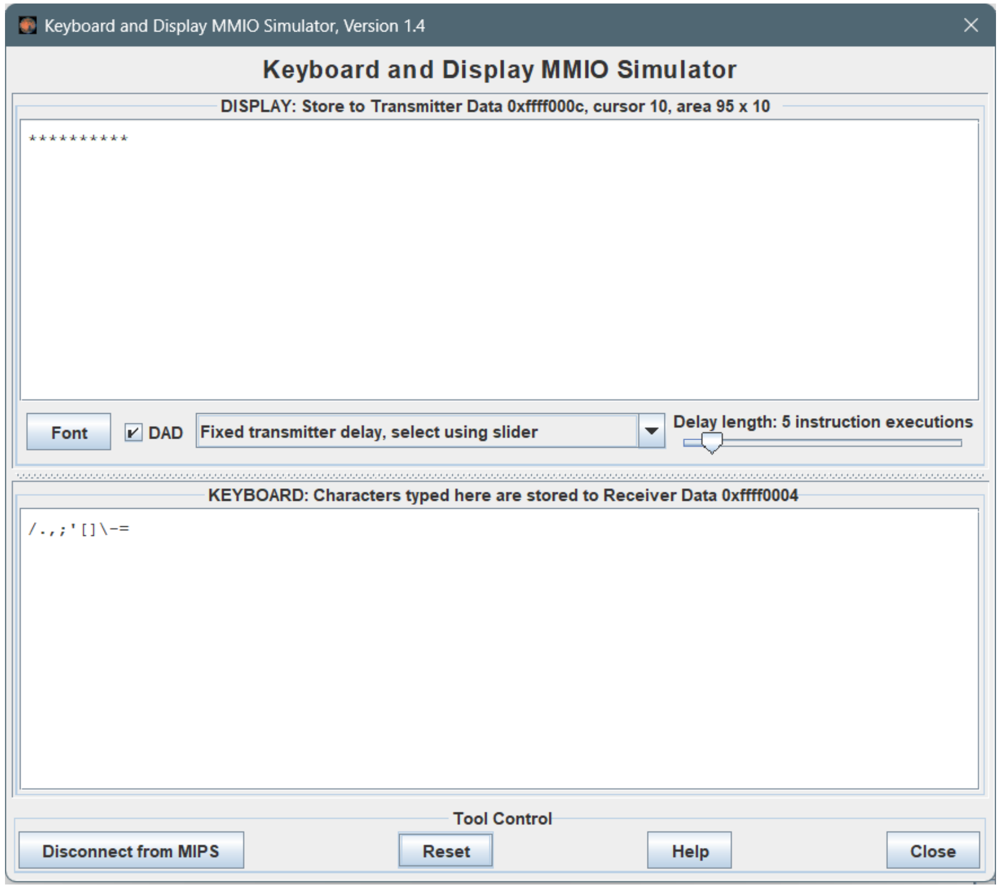
```
426214356
```

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

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

```
426214356
```

Tool Control


Keyboard and Display MMIO Simulator, Version 1.4

Keyboard and Display MMIO Simulator

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

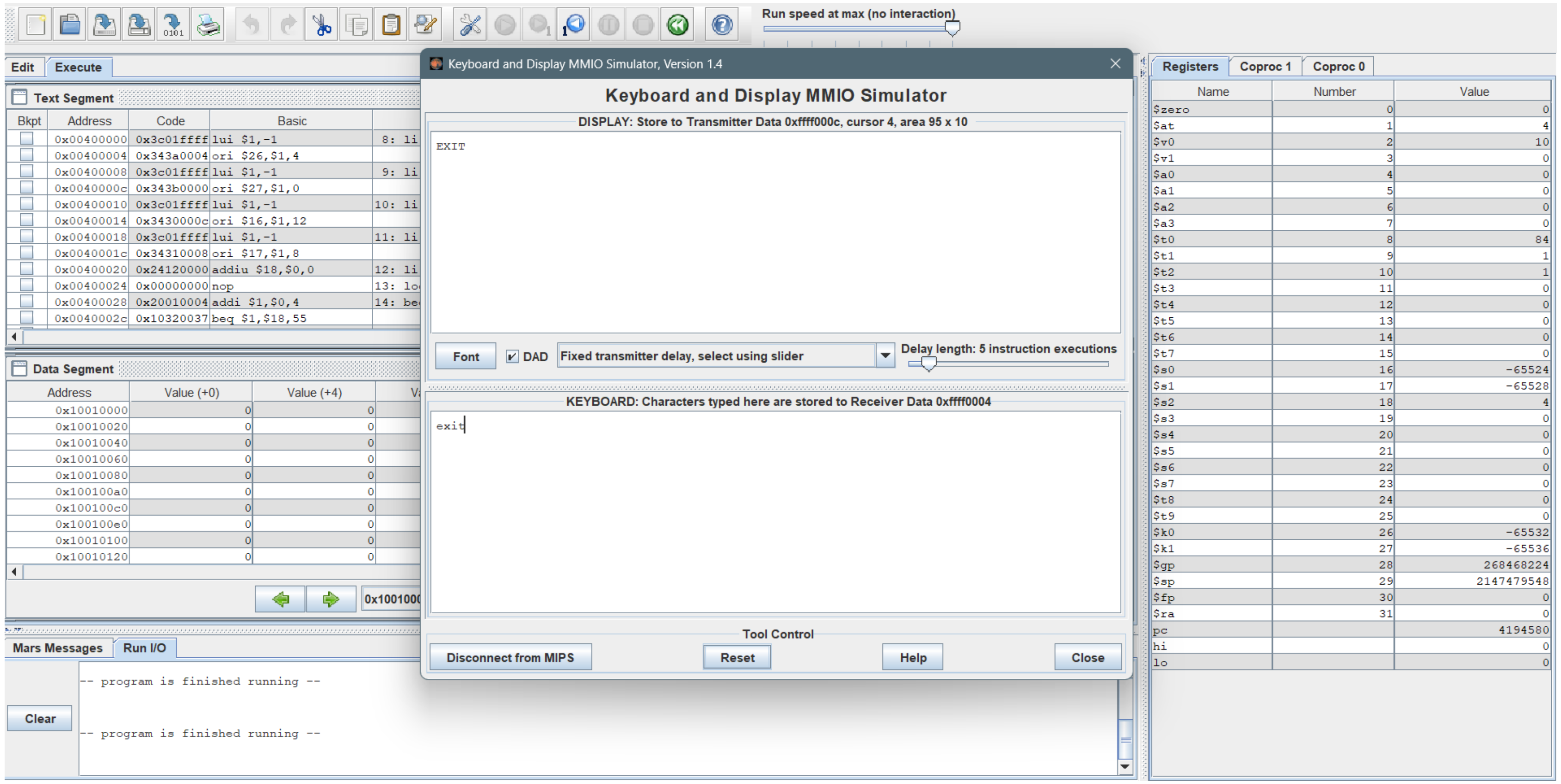
```
**********
```

[✓ DAD] Fixed transmitter delay, select using slider

Delay length: 5 instruction executions

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

```
/.,;'[]\-=
```



Text Segment

| Bkpt | Address | Code | Basic | |
|---|---|---|---|---|
| | 0x00400000 | 0x3c01ffff | lui $1,-1 | 8: li |
| | 0x00400004 | 0x343a0004 | ori $26,$1,4 | |
| | 0x00400008 | 0x3c01ffff | lui $1,-1 | 9: li |
| | 0x0040000c | 0x343b0000 | ori $27,$1,0 | |
| | 0x00400010 | 0x3c01ffff | lui $1,-1 | 10: li |
| | 0x00400014 | 0x3430000c | ori $16,$1,12 | |
| | 0x00400018 | 0x3c01ffff | lui $1,-1 | 11: li |
| | 0x0040001c | 0x34310008 | ori $17,$1,8 | |
| | 0x00400020 | 0x24120000 | addiu $18,$0,0 | 12: li |
| | 0x00400024 | 0x00000000 | nop | 13: lo |
| | 0x00400028 | 0x20010004 | addi $1,$0,4 | 14: be |
| | 0x0040002c | 0x10320037 | beq $1,$18,55 | |

Data Segment

| Address | Value (+0) | Value (+4) |
|---|---|---|
| 0x10010000 | 0 | 0 |
| 0x10010020 | 0 | 0 |
| 0x10010040 | 0 | 0 |
| 0x10010060 | 0 | 0 |
| 0x10010080 | 0 | 0 |
| 0x100100a0 | 0 | 0 |
| 0x100100c0 | 0 | 0 |
| 0x100100e0 | 0 | 0 |
| 0x10010100 | 0 | 0 |
| 0x10010120 | 0 | 0 |

0x100100

Keyboard and Display MMIO Simulator

DISPLAY: Store to Transmitter Data 0xffff000c, cursor 4, area 95 x 10

```
EXIT
```

[✓] DAD — Fixed transmitter delay, select using slider — Delay length: 5 instruction executions

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

```
exit
```

Tool Control

```
-- program is finished running --

-- program is finished running --
```

| Name | Number | Value |
|---|---|---|
| $zero | 0 | 0 |
| $at | 1 | 4 |
| $v0 | 2 | 10 |
| $v1 | 3 | 0 |
| $a0 | 4 | 0 |
| $a1 | 5 | 0 |
| $a2 | 6 | 0 |
| $a3 | 7 | 0 |
| $t0 | 8 | 84 |
| $t1 | 9 | 1 |
| $t2 | 10 | 1 |
| $t3 | 11 | 0 |
| $t4 | 12 | 0 |
| $t5 | 13 | 0 |
| $t6 | 14 | 0 |
| $t7 | 15 | 0 |
| $s0 | 16 | -65524 |
| $s1 | 17 | -65528 |
| $s2 | 18 | 4 |
| $s3 | 19 | 0 |
| $s4 | 20 | 0 |
| $s5 | 21 | 0 |
| $s6 | 22 | 0 |
| $s7 | 23 | 0 |
| $t8 | 24 | 0 |
| $t9 | 25 | 0 |
| $k0 | 26 | -65532 |
| $k1 | 27 | -65536 |
| $gp | 28 | 268468224 |
| $sp | 29 | 2147479548 |
| $fp | 30 | 0 |
| $ra | 31 | 0 |
| pc | | 4194580 |
| hi | | 0 |
| lo | | 0 |

3. **Assignment 3:** *Dùng keyboard điều khiển marsbot*
   *+ Space: bắt đầu / dừng di chuyển*
   *+ W: đi lên, S: đi xuống, A: sang trái, D: sang phải (viết hoa hoặc viết thường đều được)*
   - Mã nguồn:

```
#keyboard
.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
#mars bot
.eqv HEADING         0xffff8010 # Integer: An angle between 0 and 359
```

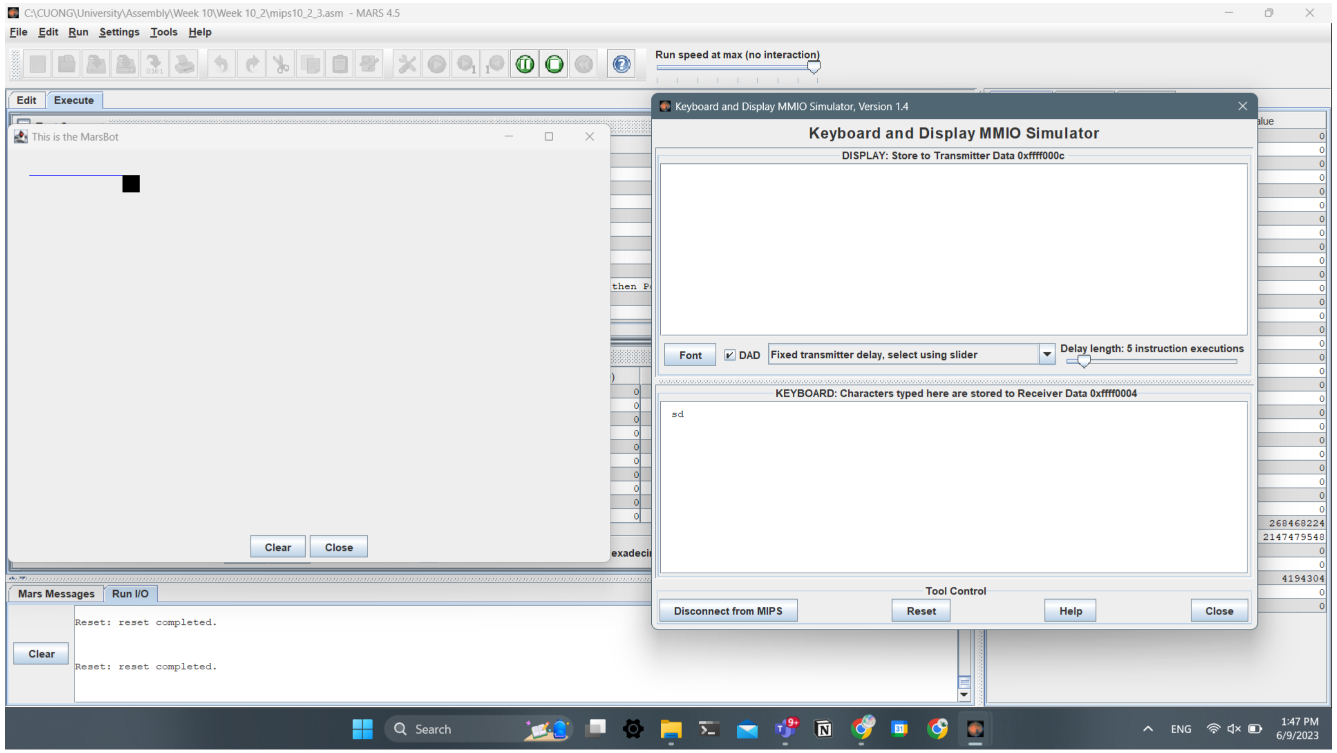
```
# 0 : North (up)
# 90: East (right)
# 180: South (down)
# 270: West (left)
.eqv MOVING             0xffff8050 # Boolean: whether or not to move
.eqv LEAVETRACK         0xffff8020 # Boolean (0 or non-0):
# whether or not to leave a track
.eqv WHEREX             0xffff8090 # Integer: Current x-location of MarsBot
.eqv WHEREY             0xffff8040 # Integer: Current y-location of MarsBot
.text
li $k0, KEY_CODE
li $k1, KEY_READY
li $s0, DISPLAY_CODE
li $s1, DISPLAY_READY

loop:

WaitForKey:
lw $t1, 0($k1) # $t1 = [$k1] = KEY_READY
beq $t1, $zero, WaitForKey # if $t1 == 0 then Polling
ReadKey:
lb $t0, 0($k0) # $t0 = [$k0] = KEY_CODE

CheckCmd:
beq $t0, ' ', ToggleMove
beq $t0, '\n', ToggleTrace
beq $t0, 'W', Up
beq $t0, 'w', Up
beq $t0, 'A', Left
beq $t0, 'a', Left
beq $t0, 'S', Down
beq $t0, 's', Down
beq $t0, 'D', Right
beq $t0, 'd', Right
j NextIteration


ToggleMove:
li $at, MOVING
lb $k0, 0($at)
beq $k0, 0, StopToGo
GoToStop:
jal STOP
li $k0, KEY_CODE
j NextIteration
StopToGo:
jal GO
li $k0, KEY_CODE
j NextIteration

ToggleTrace:
li $at, LEAVETRACK
lb $k0, 0($at)
beq $k0, 0, ToTrace
NotToTrace:
jal UNTRACK
li $k0, KEY_CODE
j NextIteration
ToTrace:
jal TRACK
li $k0, KEY_CODE
j NextIteration
Up:
add $a0, $zero, $zero
jal ROTATE
j NextIteration
```

```
Down:
addi $a0, $zero, 180
jal ROTATE
j NextIteration

Left:
addi $a0, $zero, 270
jal ROTATE
j NextIteration

Right:
addi $a0, $zero, 90
jal ROTATE
j NextIteration

NextIteration:
j loop
### for display
Exit:
li $v0, 10
syscall



#-----------------------------------------------------------
# GO procedure, to start running
# param[in] none
#-----------------------------------------------------------
GO:
li $at, MOVING # change MOVING port
addi $k0, $zero,1 # to logic 1,
sb $k0, 0($at) # to start running
```

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

```
#-----------------------------------------------------------
ROTATE:
li $at, HEADING # change HEADING port
sw $a0, 0($at) # to rotate robot
jr $ra
```

- Kết quả:


C:\CUONG\University\Assembly\Week 10\Week 10_2\mips10_2_3.asm - MARS 4.5
File Edit Run Settings Tools Help
Edit Execute
This is the MarsBot
Clear Close
Mars Messages Run I/O
Clear
Keyboard and Display MMIO Simulator, Version 1.4
Keyboard and Display MMIO Simulator
Font
Disconnect from MIPS Reset Help Close
Tool Control
Search
ENG
1:47 PM
6/9/2023

Run speed at max (no interaction)

DISPLAY: Store to Transmitter Data 0xffff000c

[✓] DAD Fixed transmitter delay, select using slider

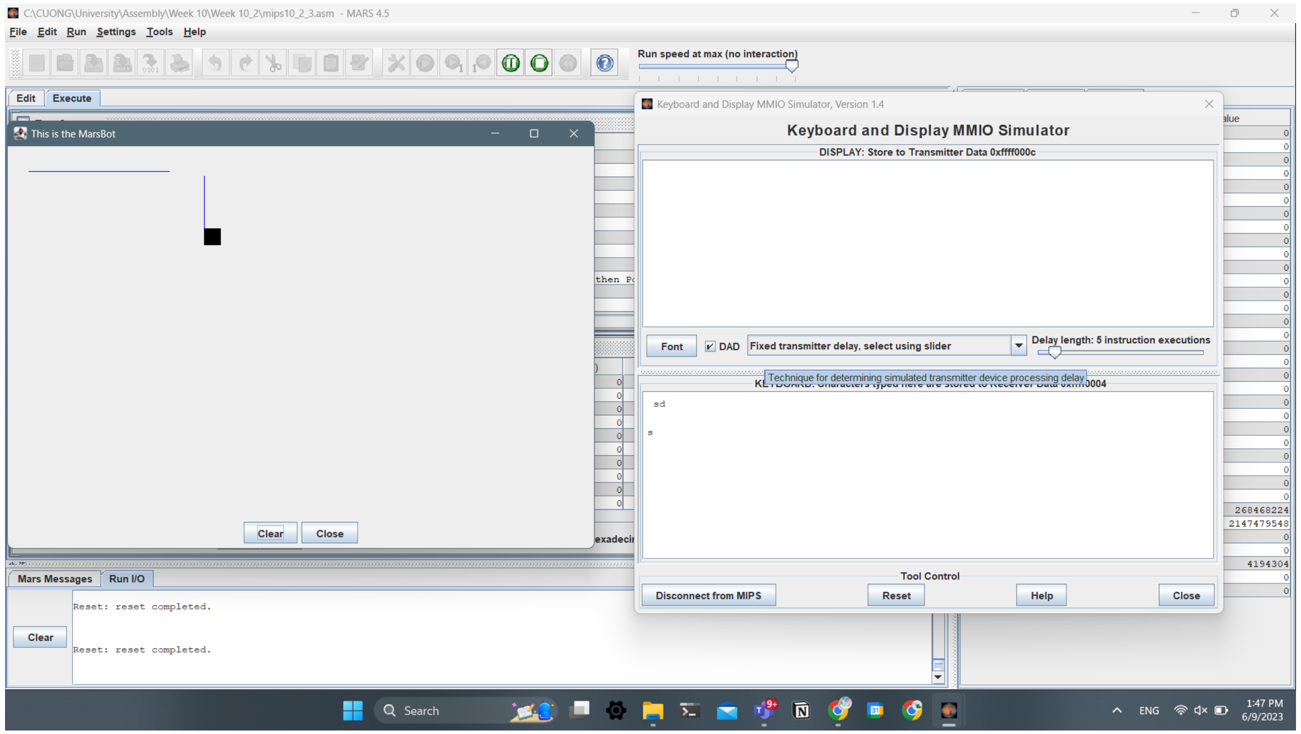
Delay length: 5 instruction executions

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

```
sd
```

```
Reset: reset completed.

Reset: reset completed.
```

```
268468224
2147479548
4194304
```


C:\CUONG\University\Assembly\Week 10\Week 10_2\mips10_2_3.asm - MARS 4.5
File Edit Run Settings Tools Help
Run speed at max (no interaction)
Edit Execute
This is the MarsBot
Clear Close
Mars Messages Run I/O
Clear
Keyboard and Display MMIO Simulator, Version 1.4
Font DAD
Delay length: 5 instruction executions
Tool Control
Disconnect from MIPS Reset Help Close
Search
ENG
1:47 PM
6/9/2023

Keyboard and Display MMIO Simulator

DISPLAY: Store to Transmitter Data 0xffff000c

Fixed transmitter delay, select using slider

Technique for determining simulated transmitter device processing delay

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

```
sd

s
```

```
Reset: reset completed.


Reset: reset completed.
```

```
268468224
2147479548
0
0
4194304
0
0
```