TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG

**********📖**********

BÁO CÁO BÀI TẬP LỚN

Bộ môn: Thực Hành Kiến Trúc Máy Tính

Giảng viên: Nguyễn Đức Tiến

Nhóm thực hiện: 14

Lớp: Việt Nhật B –K58

Sinh viên: *Phạm Đình Chiến – MSSV: 20130401 - Đề 06*

Nguyễn Duy Long – MSSV: 20132373 - Đề 04

I. Đề 04:

1. Phân Tích

Máy gia công cơ khí chính xác CNC Marsbot được dùng để cắt tấm kim loại theo các đường nét được qui định trước. CNC Marsbot có một lưỡi cắt dịch chuyển trên tấm kim loại, với giả định rằng:

- Nếu lưỡi cắt dịch chuyển nhưng không cắt tấm kim loại, tức là Marsbot di chuyển nhưng không để lại vết (Track)

- Nếu lưỡi cắt dịch chuyển và cắt tấm kim loại, tức là Marsbot di chuyển và có để lại vết. Để điều khiển Marsbot cắt đúng như hình dạng mong muốn, người ta nạp vào Marsbot một mảng cấu trúc gồm 3 phần tử:

- <Góc chuyển động>, <Thời gian>, <Cắt/Không cắt>

- Trong đó <Góc chuyển động> là góc của hàm HEADING của Marsbot

- <Thời gian> là thời gian duy trì quá trình vận hành hiện tại

- <Cắt/Không cắt> thiết lập lưu vết/không lưu vết

Hãy lập trình để CNC Marsbot có thể:

- Thực hiện cắt kim loại như đã mô tả

- Nội dung postscript được lưu trữ cố định bên trong mã nguồn

- Mã nguồn chứa 3 postscript và người dùng sử dụng 3 phím 0, 4, 8 trên bàn phím Key Matrix để chọn postscript nào sẽ được gia công.

- Một postscript chứa chữ DCE cần gia công. Hai script còn lại sinh viên tự đề xuất (tối thiểu 10 đường cắt)

2. Thuật Toán

Khởi tạo 1 chuỗi các số (mảng số) các số int miêu tả trạng thái chuyển động của CNC MArbot. Mỗi trạng thái là bộ 3 số liên tiếp nhau, Các số được phân cách với nhau bằng dấu ",".

Sau đó, load địa chỉ của phần tử đầu của script(a1) và phần tử cuối script(a2) , thực hiện vòng lặp kiểm tra a1<= a2 thì đọc bộ 3 số . Ban đầu, đọc dữ liệu word từ bộ nhớ đưa vào thanh ghi, lấy giá trị <Cắt/Không Cắt > bằng : lw a0 , 8(a1) #a0 = script[a1 +2] -> . Sau đó lấy giá trị của <Góc chuyển động> bằng: lw a0, 0(a1) #a0 = script[a1] . Sau đó lấy giá trị <Thời gian > bằng lw a0, 4(a1) #a0 =script[a1+1] . Sauk hi lấy được các

thông số chuyển động của 1 vết cắt, ta sẽ gọi các hàm thực hiện và tăng con trỏ ở a1 lên 12 để dich tiếp đến thông số của vết cắt tiếp theo. Tiếp tục cho đến khi gặp con trỏ của phần tử cuối cùng thì kết thúc script.

Do chưa được tối ưu, nên mỗi lần vẽ hình khác phải reset Digital Lab Sim, Marsbot và chạy lại chương trình

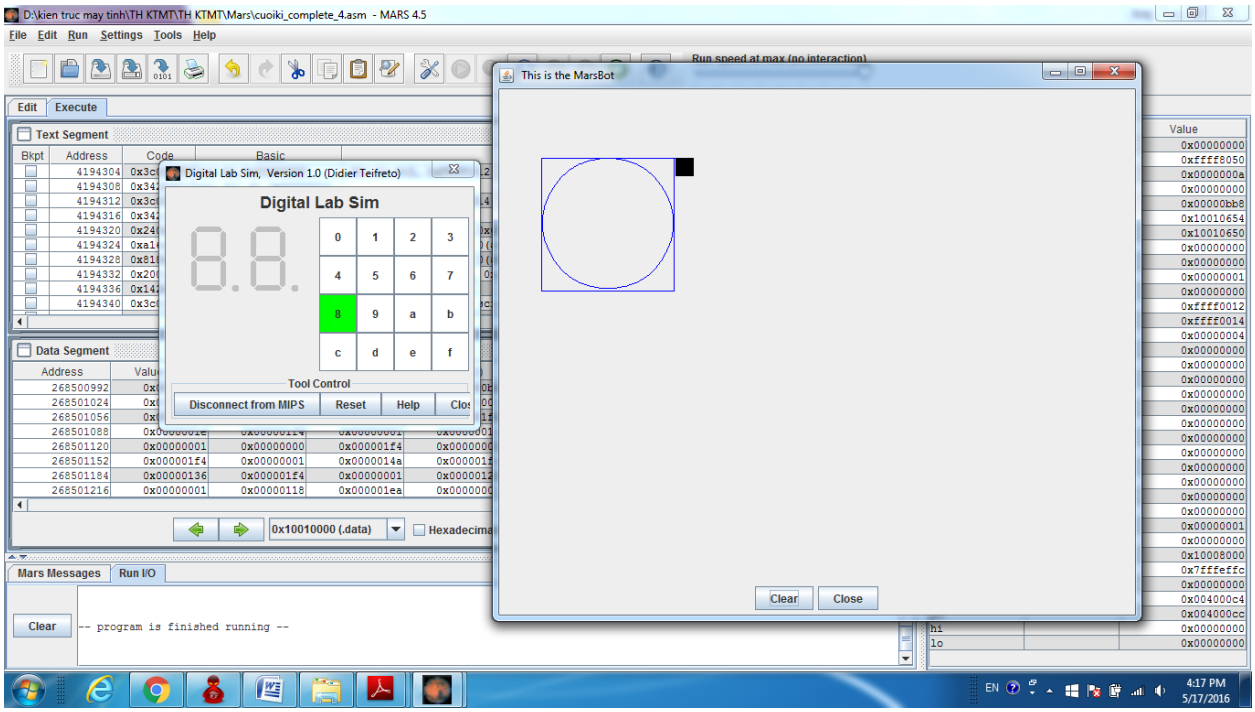
3. Mã Nguồn

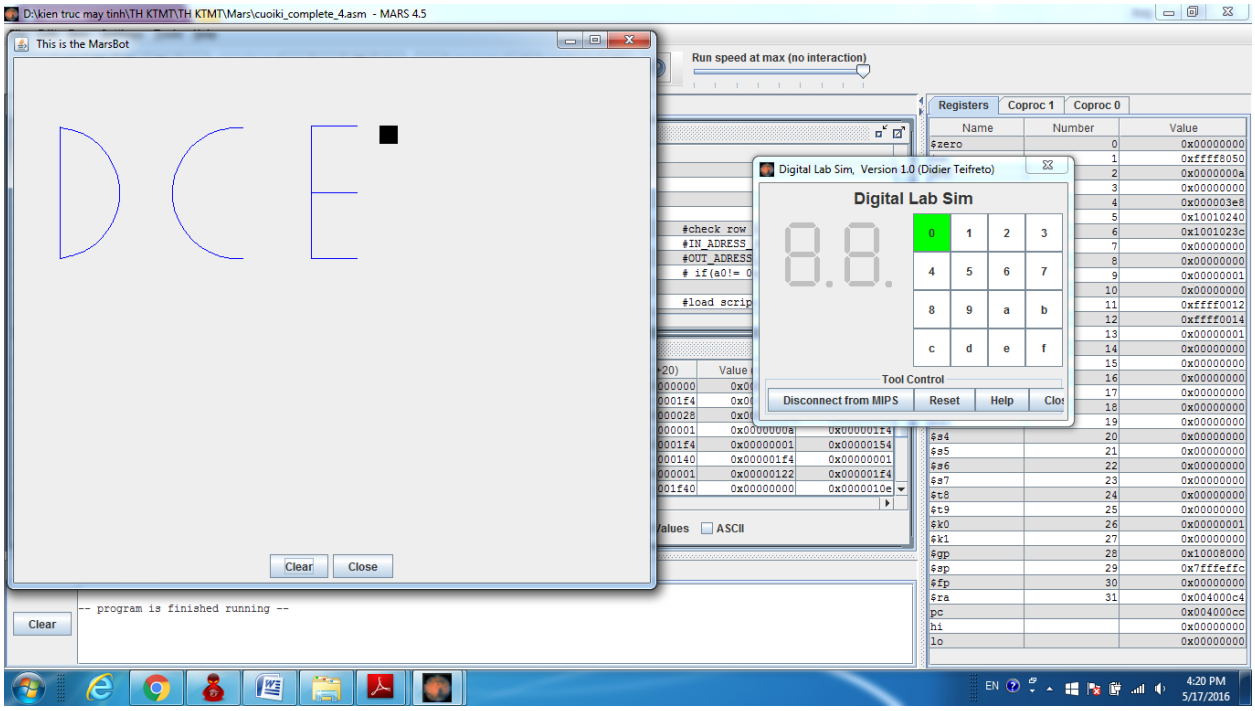
```
.eqv HEADING 0xffff8010    # Integer: An angle between 0 and 359
                       # 0 : North (up)
                       # 90: East (right)
                       # 180: South (down)
                       # 270: West (left)
.eqv MOVING 0xffff8050          # Boolean: whether or not to move
.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
.eqv OUT_ADRESS_HEXA_KEYBOARD 0xFFFF0014
.eqv IN_ADRESS_HEXA_KEYBOARD 0xFFFF0012

.data
script1: .word
90,2000,0,180,3000,0,180,5790,1,80,500,1,70,500,1,60,500,1,50,500,1,40,500,1
,30,500,1,20,500,1,10,500,1,0,500,1,350,500,1,340,500,1,330,500,1,320,500,1,
310,500,1,300,500,1,290,500,1,280,490,1,90,8000,0,270,500,1,260,500,1,250,50
0,1,240,500,1,230,500,1,220,500,1,210,500,1,200,500,1,190,500,1,180,500,1,17
0,500,1,160,500,1,150,500,1,140,500,1,130,500,1,120,500,1,110,500,1,100,500,
1,90,500,1,90,5000,0,270,2000,1,0,2900,1,90,2000,1,270,2000,1,0,2900,1,90,20
00,1,90,1000,0
script2: .word
90,5000,0,180,3000,0,270,500,1,260,500,1,250,500,1,240,500,1,230,500,1,220,5
00,1,210,500,1,200,500,1,190,500,1,180,500,1,170,500,1,160,500,1,150,500,1,1
40,500,1,130,500,1,120,500,1,110,500,1,100,500,1,90,500,1,90,2000,0,0,5800,1
,180,2900,1,90,2000,1,0,2900,1,180,5800,1,90,2000,0,0,5800,1,90,2000,0,90,20
00,1,270,2000,1,180,2900,1,90,2000,1,270,2000,1,180,2900,1,90,2000,1,90,2000
,0,0,5800,1,150,6697,1,0,5800,1,90,2000,0
script3: .word
90,5000,0,180,3000,0,270,500,1,260,500,1,250,500,1,240,500,1,230,500,1,220,5
00,1,210,500,1,200,500,1,190,500,1,180,500,1,170,500,1,160,500,1,150,500,1,1
40,500,1,130,500,1,120,500,1,110,500,1,100,500,1,90,500,1,80,500,1,70,500,1,
60,500,1,50,500,1,40,500,1,30,500,1,20,500,1,10,500,1,0,500,1,350,500,1,340,
500,1,330,500,1,320,500,1,310,500,1,300,500,1,290,500,1,280,500,1,90,1000,0
end_script : .word
.text
li $t3, IN_ADRESS_HEXA_KEYBOARD
li $t4, OUT_ADRESS_HEXA_KEYBOARD

#-----------------------------------------------------------
     #quet gia tri nhan dc tu ban phim
polling:
     li $t5, 0x01                    #check row 0,1,2,3 ->  check a0==0?
     sb $t5, 0($t3)                  #IN_ADRESS_HEXA_KEYBOARD
     lb $a0, 0($t4)                  #OUT_ADRESS_HEXA_KEYBOARD
```

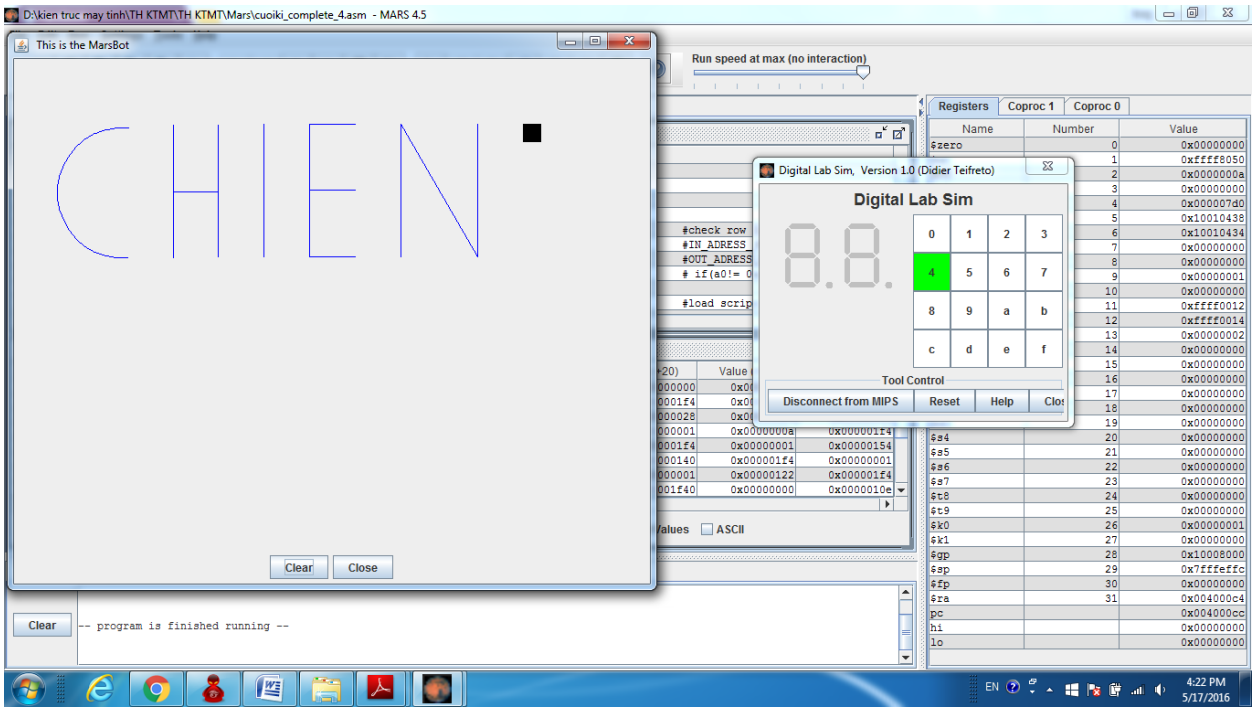
```
      bne $a0, 0x11, SCAN_4                 # if(a0!= 0x11) -> SCAN_4
      #else (a0 == 0x11) load addres script1 ,script2
      la $a1, script1                   #load script1 a1  = adress(script1[0])
      la $a2 ,script2                   #load script2 a2  = adress(script2[0])
      j START                           #jump START
      SCAN_4:
      li $t5, 0x02                      #check row 4,5,6,7 -> check ==4?
      sb $t5, 0($t3)                    #in_adress_hexa_keyboard
      lb $a0, 0($t4)                    #out_adress_hexa_keyboard
      bne $a0, 0x12, SCAN_8             #if(a0!=4)  SCAN_8
      #else (a0==4) load adress script2, script3
      la $a1, script2                   #load script2 a1  = adress(script2[0])
      la $a2, script3                   #load script3 a2  = adress(script3[0])
      j START                           #jump START
      SCAN_8:
      li $t5, 0X04                      #check row 8,9,a,b ->check a==8 ?
      sb $t5, 0($t3)                    #in_adress_hexa_keyboard
      lb $a0, 0($t4)                    #out_adress_hexa_keyboard
      bne $a0, 0x14, back_polling       #if(a0!=8) -> back_to_polling
      #else (a0==8) load address script3 , end_script
      la $a1, script3                   #load script3 a1 = adress(script3[0])
      la $a2, end_script                #load end_script a2 =
adress(end_script)
      j START                           #jump START
back_polling: j polling                 #jump polling

#-------------------------------------------------------------
   START:
      #a2 = a2-4 -> a2 = adress(cript[last]) -> VD : a1 =
address(script[0]) , a2 = adress(script[n-1])
      addi $a2,$a2,-4
      jal GO                            #moving bot
#-------------------------------------------------------------
      #doc script cho marbot
   READ_SCRIPT:
      slt $t1 , $a2 , $a1               #check a2 <a1 ?
      bne $t1, $zero , END              # if (a2<a1) ket thuc END
      lw $a0 ,  8($a1)                  #else a0 = 8($a1) -> a0 = gia tri
track cua bo script ->set marbot track/untrack
      jal TRACK
      lw $a0, 0($a1)                    #a0 = 0($a1) -> a0 = gia tri goc cua
bo script -> set marbot rotate
      jal ROTATE
      lw $a0,4($a1)                     #a0 = 4($a1) -> a0 = gia tri thoi gian
cua bo scrpit ->set sleep time
      jal SLEEP
      jal UNTRACK                       # untrack postscript truoc
      addi $a1,$a1,12                   #dich thanh ghi a1 qua bo 3 so da doc
      j READ_SCRIPT                     # quay lai vong lap
#-------------------------------------------------------------
   END:
      jal STOP
      li $v0, 10
      syscall
#-------------------------------------------------------------
   GO:
      li $at, MOVING                    # change MOVING port
```

```
      addi $k0, $zero,1                 #to logic 1,
      sb $k0, 0($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
      sw $a0, 0($at)                    # to start tracking/untracking
      jr $ra
#-----------------------------------------------------------
  ROTATE:
      li $at, HEADING                   # change HEADING port
      sw $a0, 0($at)                    # to rotate robot
      jr $ra
#-----------------------------------------------------------
  SLEEP:
      addi $v0,$zero,32                 #sleep time
      syscall
      jr $ra
#-----------------------------------------------------------
  UNTRACK:
      li $at, LEAVETRACK                # change LEAVETRACK port to 0
      sb $zero, 0($at)                  # to stop drawing tail
      jr $ra
```

4. Hình Ảnh Kết Quả Mô Phỏng




D:\kien truc may tinh\TH KTMT\TH KTMT\Mars\cuoiki_complete_4.asm - MARS 4.5
This is the MarsBot
D C E
Clear
Close
Run speed at max (no interaction)
Digital Lab Sim, Version 1.0 (Didier Teifreto)
Digital Lab Sim
Tool Control
Disconnect from MIPS
Reset
Help
Registers
Coproc 1
Coproc 0
Name
Number
Value
-- program is finished running --
4:20 PM
5/17/2016


D:\kien truc may tinh\TH KTMT\TH KTMT\Mars\cuoiki_complete_4.asm - MARS 4.5
This is the MarsBot
C H I E N
Clear
Close
Run speed at max (no interaction)
Digital Lab Sim, Version 1.0 (Didier Teifreto)
Digital Lab Sim
Tool Control
Disconnect from MIPS
Reset
Help
Registers
Coproc 1
Coproc 0
Name
Number
Value
-- program is finished running --
4:22 PM
5/17/2016

II. Đề 06:

1. Phân Tích

Hàm FindSubStr()

Viết chương trình sử dụng MIPS để tìm các vị trí xuất hiện của xâu kí tự con trong xâu kí tự lớn. Chương trình sẽ yêu cầu người dùng nhập vào một xâu và một xâu cần tìm kiếm (cả 2 xâu đều là xâu ASCII). Đầu ra của chương trình sẽ là chỉ số nơi mà xâu cần tìm kiếm xuất hiện trong xâu gốc. Hàm tìm kiếm cho phép tìm theo các cách

1. Không phân biệt chữ hoa, chứ thường

2. Phân biệt chữ hoa, thường

Ví dụ:

Input:

string1: DAI HOC BACH KHOA LA dai hoc da nganh

string2: HOC

không phân biệt hoa thường

Output:

4 25

2. Thuật Toán (Code C)

```
#include<stdio.h>
#include<String.h>
void main(){
   char String[200], subString[20];
   int i=0,j,k=0 ;
   printf("Input:\n");
   do{
      printf("\tstring1: ");
      gets(String);
   }while(String[0] == NULL);
   do{
```

```
        printf("\tstring2: ");
        gets(subString);
    }while(subString[0] == NULL);
    printf("Co phan biet chu hoa - thuong hay khong ?(y/n)");
    scanf("%d",&j);
    while(getchar()!= '\n');
    if(j == 'n' || j == 'N')
        upCase(&String, &subString);
    j = 0;
    printf("Output:\n\t");
    while(String[i]!='\0') {     // trong khi chua duyet het chuoi 1
        if(String[i]==subString[0]) {    // neu ky tu tai vi tri i cua 1 = ky tu dau tien cua 2
            j=1;                    // j = 0 -> bien dem ky tu cua 2
        while(subString[j]!='\0' && subString[j]== String[j+i])    // trong khi chua duyet het 2 va cac ky tu trung
nhau
                j++;                // chuyen sang ky tu tiep theo cua 2
        if(subString[j]=='\0'){       // neu duyet het chuoi 2 -> bao vi tri ton tai trong 1
                printf("%d ", i);
                k = 1;
            }
        }
        i++;
    }
    if(k==0)
        printf("Not found ");
}
void upCase(char *String, char *subString){
    int i = 0, j = 0;
    while(String[i] != '\0'){
```

```
    if(String[i]>=97 && String[i] <= 122)
        String[i] = String[i] - 32;
    i++;
  }
  while(subString[j] != '\0'){
    if(subString[j]>=97 && subString[j] <= 122)
        subString[j] = subString[j] - 32;
    j++;
  }
}
```

3. Mã Nguồn

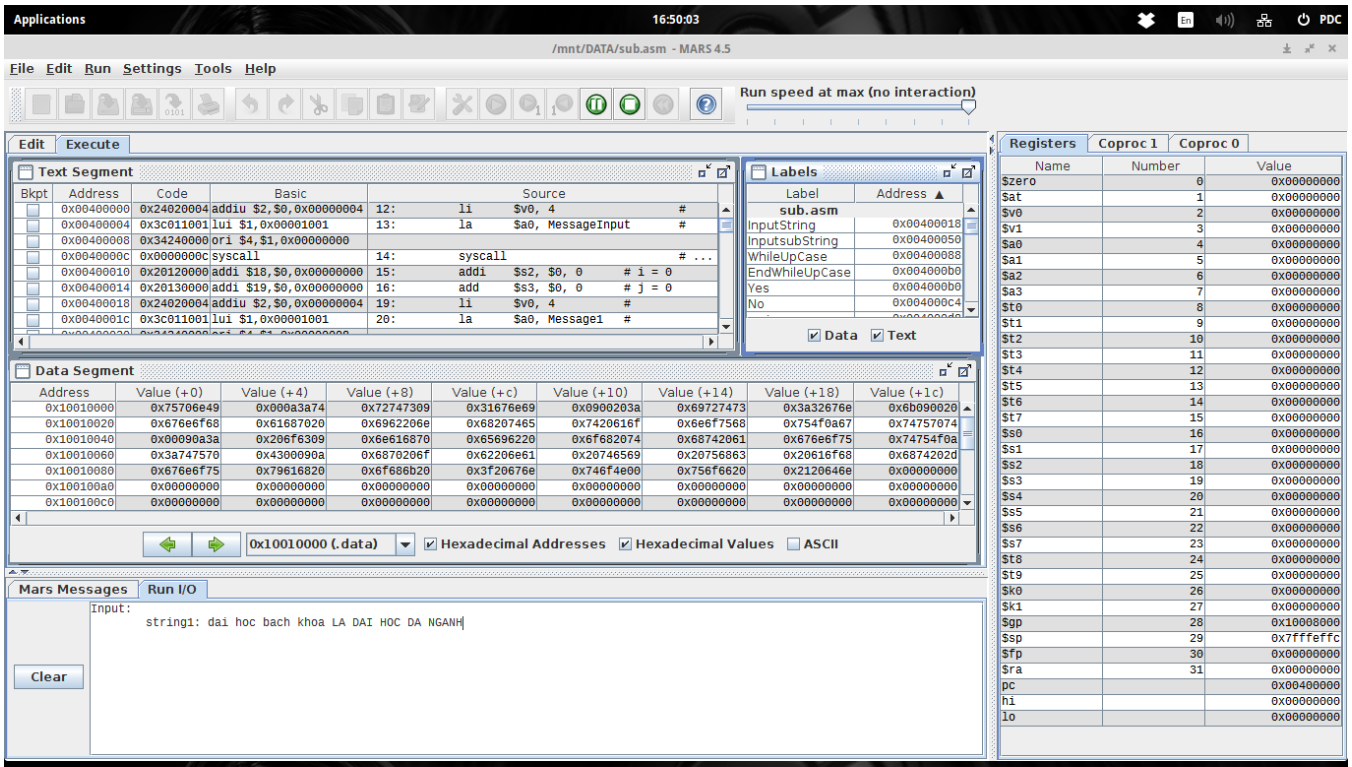
```
.data
MessageInput:    .asciiz    "Input:\n"
Message1:  .asciiz    "\tstring1: "
Message2:  .asciiz    "\tstring2: "
Message3:  .asciiz    "\tkhong phan biet hoa thuong\nOutput:\n    "
Message4:  .asciiz    "\tco phan biet hoa thuong\nOutput:\n "
Message5:  .asciiz    "Co phan biet chu hoa - thuong hay khong ?"
Message6:  .asciiz    "Not found !"
String:          .space     100
subString: .space     100
.text
      li    $v0, 4                #
      la    $a0, MessageInput     #
      syscall                     # print Input:
#---------------------------------------------------------------------
InputString:
      li    $v0, 4          #
      la    $a0, Message1   #
      syscall               # print String1:

      li    $v0, 8          #
      la    $a0, String     #
      li    $a1, 10         #
      syscall               # input to String:


      lb    $t3, 0($a0)           # t3 = String[0]
      addi  $at, $0, 10
      beq   $t3, $at, InputString # if String[0] = \n -> Loop
      nop


      add   $s0, $a0, $0          # String = s0
#---------------------------------------------------------------------
InputsubString:
```

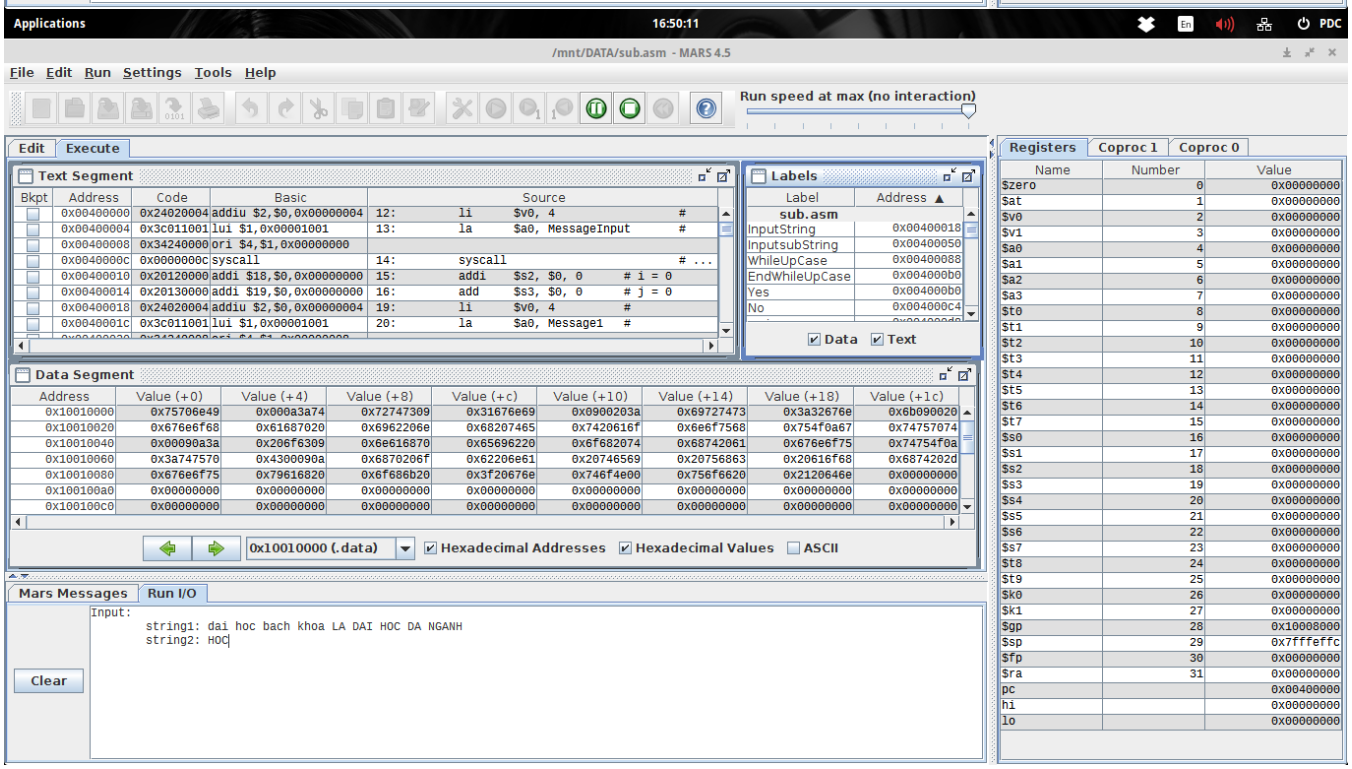
```
      li    $v0, 4          #
      la    $a0, Message2   #
      syscall               # print String2:

      li    $v0, 8          #
      la    $a0, subString  #
      li    $a1, 10    #
      syscall               # input to subString

      lb    $t4, 0($a0)           # t4 = subString[0]
      addi  $at, $0, 10
      beq   $t4, $at, InputsubString   # if subString[0] = \n -> Loop
      nop
      add   $s1, $a0, $0          # subString = s1
#------------------------------------------------------------------------
WhileUpCase:
      li    $v0, 50
      la    $a0, Message5
      syscall
      addi  $at, $0, 0
      beq   $a0, $at, Yes         # co phan biet hoa - thuong
      nop
      addi  $at, $0, 1
      beq   $a0, $at, No          # khong phan biet hoa thuong
      nop
      addi  $at, $0, 2
      beq   $a0, $at, WhileUpCase # chon lai
      nop
EndWhileUpCase:
#------------------------------------------------------------------------
Yes:
      li    $v0, 4                #
      la    $a0, Message4         #
      syscall                     # print co phan biet hoa thuong
      j     main
#------------------------------------------------------------------------
No:
      li    $v0, 4                #
      la    $a0, Message3         #
      syscall                     # print khong phan biet hoa thuong
      addi  $t9, $a0, 1           # flag upCase ?
#------------------------------------------------------------------------
main:
      addi  $s2, $0, 0 # i = 0
      add   $s3, $0, 0 # j = 0
While:
      add   $t1, $s2, $s0         # addr of String[i] in t1
      lb    $t3, 0($t1)           # t3 = String[i]
      lb    $t4, 0($s1)           # t4 = subString[0]
      beq   $t3, $0, EndWhile     # if String[i] = \0 -> exit while
      nop
      addi  $at, $0, 10
      beq   $t3, $at, EndWhile    # if String[i] = \n -> exit While
      nop
      If:
            beq   $t9, $0, notUpCase1
            nop
```

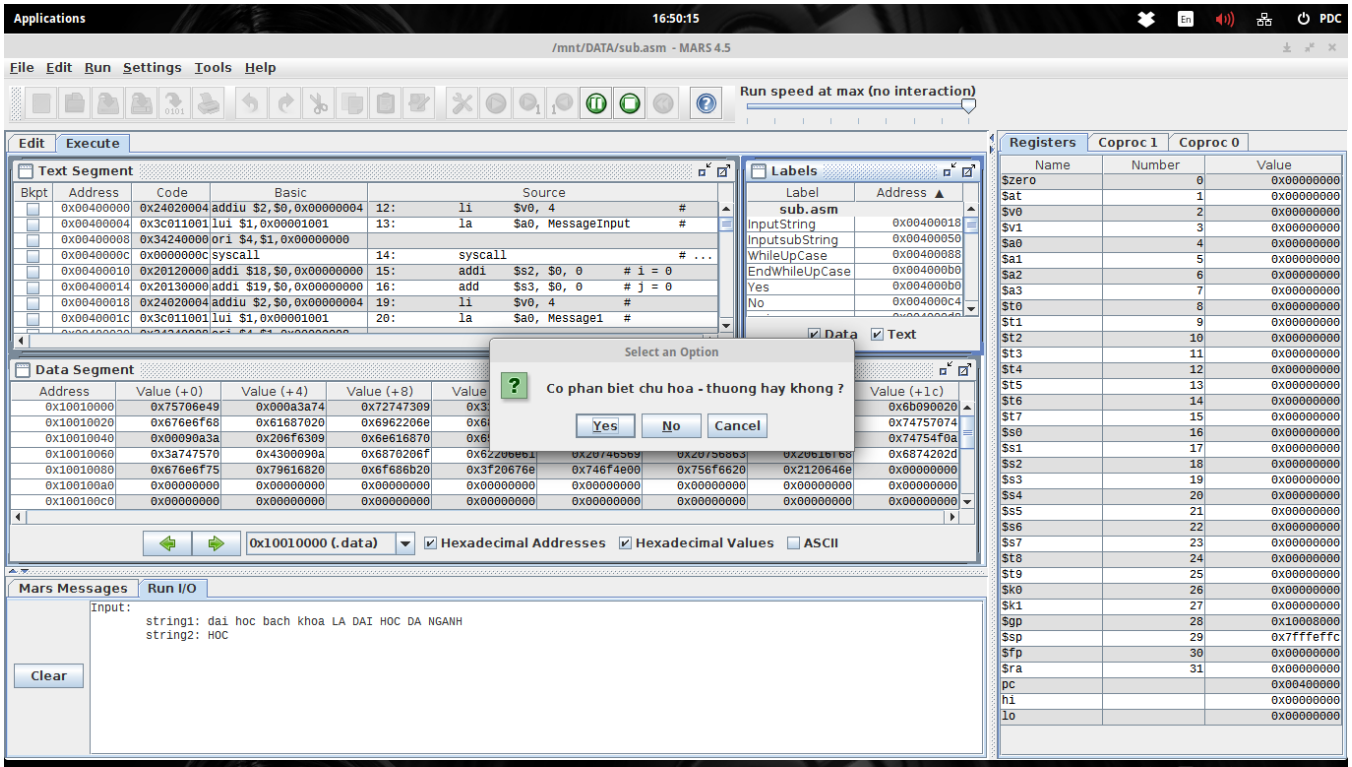
```
                jal   UpCase
                nop
            notUpCase1:
            bne   $t3, $t4, Add    # if String[i] != subString[0] -> exit
If
            nop
            add   $s3, $0, 1 # j = 1
            While2:
                add   $t5, $s3, $t1    # t5 = addr[i + j]
                lb    $t3, 0($t5)      # t6 = String[i+j}
                add   $t7, $s3, $s1    # t7 = addr of subString[j]
                lb    $t4, 0($t7)      # t4 = suString[j]
                beq   $t9, $0, notUpCase2
                nop
                    jal   UpCase
                    nop
                notUpCase2:
                beq   $t4, $0, EndWhile2    # if subString[j] == \0 ->
exit while2
                nop
                addi  $at, $0, 10
                beq   $t4, $at, EndWhile2   # if subString[j] == \n ->
exit while2
                nop
                bne   $t4, $t3, EndWhile2   # if subString[j] !=
String[i+j] -> exit while2
                nop
                addi  $s3, $s3, 1           # j = j + 1
                j     While2                # Loop2
            EndWhile2:
            PrintPos:
                If2:
                    addi  $at, $0, 10
                    bne   $t4, $at, EndIf2 # if subString[j] != \n ->
exit If2
                    nop
                    add   $a0, $s2, $0     # a0 = s2 = i
                    li    $v0, 1           #
                    syscall                # print pos

                    addi  $a0, $0, 32      # print space
                    li    $v0, 11          #
                    syscall                #

                    addi  $t0, $0, 1 # k = 1    -> co subString hay
k ?
                EndIf2:
                    bne   $t4, $0, EndPrintPos  # if subString[j] !=
\0 -> exit PrintPos
                    nop
                    add   $a0, $s2, $0     # a0 = s2 = i
                    li    $v0, 1           #
                    syscall                # print pos

                    addi  $a0, $0, 32      # print space
                    li    $v0, 11          #
                    syscall                #
```

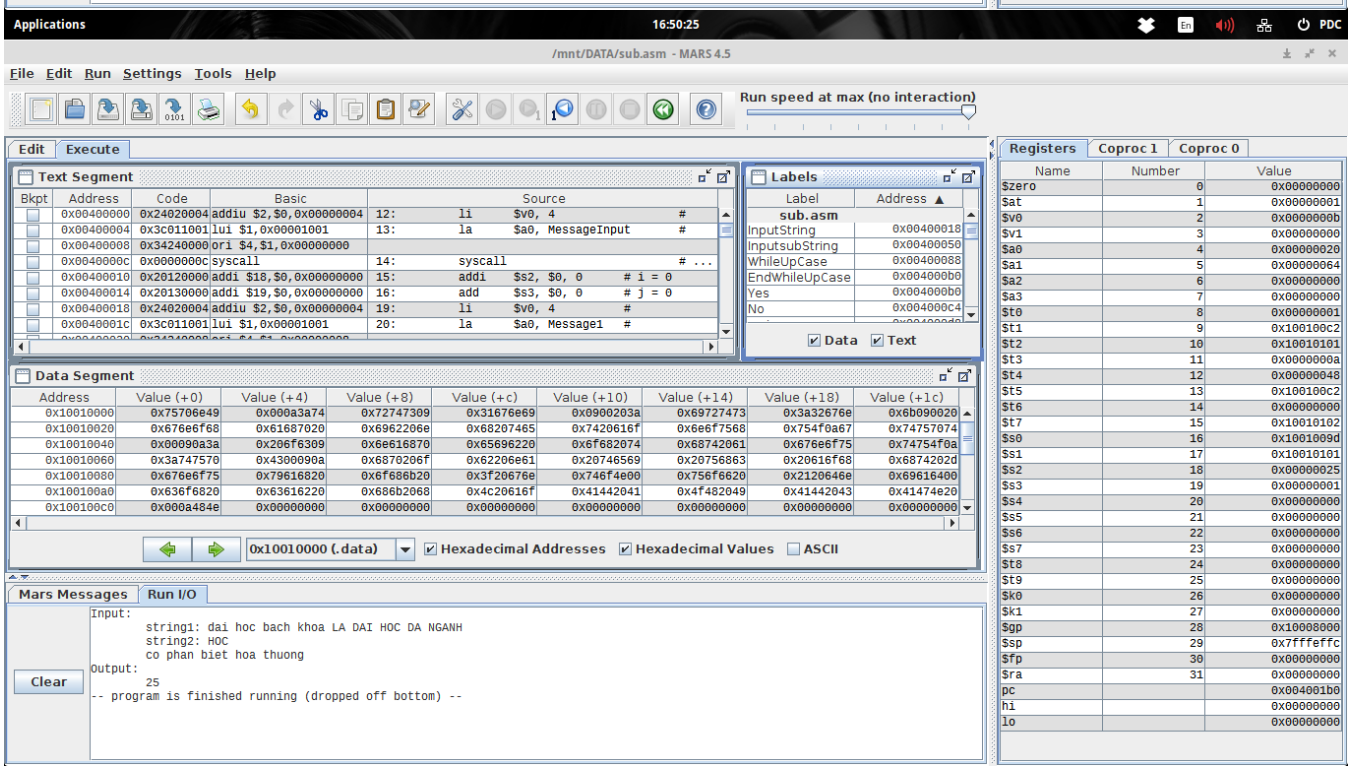
```
                    addi  $t0, $0, 1 # k = 1    -> co subString hay
k ?
          EndPrintPos:
     EndIf:
     Add: addi  $s2, $s2, 1                # i = i + 1
          j     While                 # Loop
#-------------------------------------------------------------------
UpCase:
     IFUpCase1:
          sub   $t8, $t3, 97               # if String[i] >= 97 &&
String[i] <= 122
          bltz  $t8, EndIfUpCase1
          nop         #
          sub   $t8, $t3, 122              #
          bgtz  $t8, EndIfUpCase1
          nop         #
          sub   $t3, $t3, 32               # String[i] = String[i] -
32
     EndIfUpCase1:

     IfUpCase2:
          sub   $t8, $t4, 97               # if subString[j] >= 97 &&
String[j] <= 122
          bltz  $t8, EndIfUpCase2
          nop         #
          sub   $t8, $t4, 122              #
          bgtz  $t8, EndIfUpCase2
          nop         #
          sub   $t4, $t4, 32               # subString[j] = String[j]
- 32
     EndIfUpCase2:
     jr    $ra
#-------------------------------------------------------------------
EndWhile:
     addi  $at, $0, 1
     beq   $t0, $at, Exit        # if t7 == 1 -> Exit
     nop
     li    $v0, 4                # else print Not found !
     la    $a0, Message6         #
     syscall              #
Exit:
```

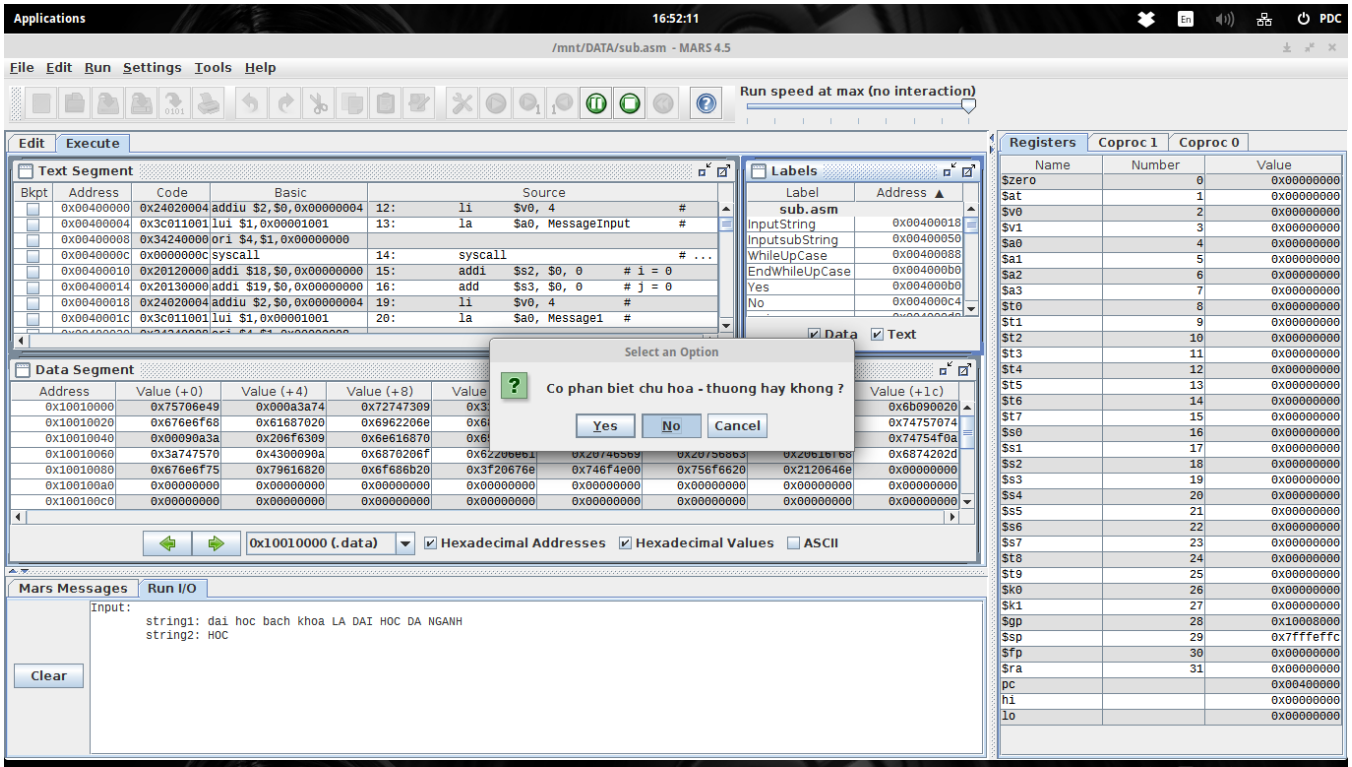
4. Hình Ảnh Kết Quả Mô Phỏng

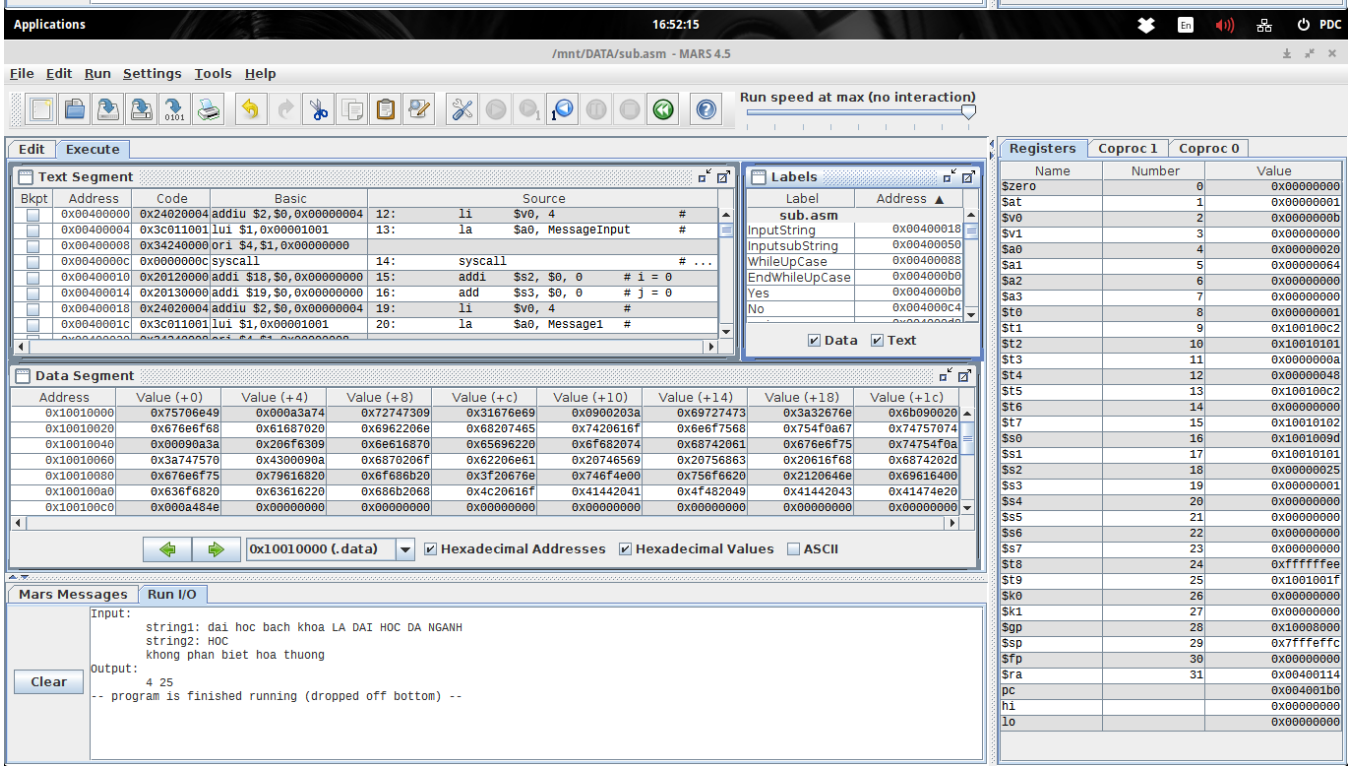





Applications
16:50:15
/mnt/DATA/sub.asm - MARS 4.5
File Edit Run Settings Tools Help
Run speed at max (no interaction)
Edit Execute
Text Segment
Labels
Data Segment
Select an Option
Co phan biet chu hoa - thuong hay khong ?
Yes
No
Cancel
Registers
Coproc 1
Coproc 0
Mars Messages
Run I/O
Input:
string1: dai hoc bach khoa LA DAI HOC DA NGANH
string2: HOC
Clear


Applications
16:50:25
/mnt/DATA/sub.asm - MARS 4.5
File Edit Run Settings Tools Help
Run speed at max (no interaction)
Edit Execute
Text Segment
Labels
Data Segment
Registers
Coproc 1
Coproc 0
Mars Messages
Run I/O
Input:
string1: dai hoc bach khoa LA DAI HOC DA NGANH
string2: HOC
co phan biet hoa thuong
Output:
25
-- program is finished running (dropped off bottom) --
Clear


Applications
16:52:11
/mnt/DATA/sub.asm - MARS 4.5
File Edit Run Settings Tools Help
Run speed at max (no interaction)
Edit Execute
Text Segment
Labels
Data Segment
Select an Option
Co phan biet chu hoa - thuong hay khong ?
Yes
No
Cancel
Registers
Coproc 1
Coproc 0
Mars Messages
Run I/O
Input:
string1: dai hoc bach khoa LA DAI HOC DA NGANH
string2: HOC
Clear


Applications
16:52:15
/mnt/DATA/sub.asm - MARS 4.5
File Edit Run Settings Tools Help
Run speed at max (no interaction)
Edit Execute
Text Segment
Labels
Data Segment
Registers
Coproc 1
Coproc 0
Mars Messages
Run I/O
Input:
string1: dai hoc bach khoa LA DAI HOC DA NGANH
string2: HOC
khong phan biet hoa thuong
Output:
4 25
-- program is finished running (dropped off bottom) --
Clear