ΑΝΑΦΟΡΑ ΕΡΓΑΣΤΗΡΙΟΥ 6

Ομάδα LAB20130164

ΚΑΡΑΜΠΑΣΟΓΛΟΥ ΔΗΜΗΤΡΙΟΣ
ΠΕΤΡΑΚΟΣ ΜΑΝΩΛΗΣ

ΣΚΟΠΟΣ ΤΟΥ ΕΡΓΑΣΤΗΡΙΟΥ

Ο σκοπός του εργαστηρίου είναι να κατανοήσουμε τον τρόπο με τον όποιο γράφονται και διαβάζονται χαρακτήρες από το πληκτρολόγιο προς την κονσόλα χωρίς syscall, καθώς και να εμβαθύνουμε περαιτερω στη γλώσσα ASSEMBLY. Τέλος, καταλάβαμε ότι η τεχνική polling είναι μια σειρά διαρκών ενεργειών για τον έλεγχο των συσκευών ,μέχρι να καταλάβουμε ότι αυτές οι συσκευές είναι έτοιμες για να δεχθούν ή να πάρουν δεδομένα.

ΠΕΡΙΓΡΑΦΗ ΖΗΤΟΥΜΕΝΩΝ

Μας ζητήθηκε να υλοποιήσουμε τις συναρτήσεις Write_ch και Read_ch με βάση την τεχνική polling, καθώς και τη συνάρτηση main που τις καλεί, με σκοπό να ελέγξουμε την ορθή λειτουργία τους. Στην συνέχεια, χρησιμοποιώντας αυτές τις συναρτήσεις και όχι syscall έπρεπε να γραψουμε ένα πλήρες πρόγραμμα που διαβάζει μια συμβολοσειρά από το πληκτρολόγιο, την μετατρέπει σε κεφαλαία και εκτυπώνει τη διαμορφωμένη συμβολοσειρά στην κονσόλα.

ΠΕΡΙΓΡΑΦΗ ΕΚΤΕΛΕΣΗΣ

Στην συνάρτηση main καλείται η συνάρτηση read_string η οποία καλεί κατ' επανάληψη την συνάρτηση read_ch μέχρι να δοθεί “enter” από τον χρήστη ή να φτάσει τους 10 χαρακτήρες και αποθηκεύει σε ένα string τους χαρακτήρες που δέχεται από αυτήν. Η read_ch διαβάζει έναν χαρακτήρα τον οποίον πρώτα στέλνει σαν όρισμα στην write_ch και μετά επιστρέφει τον ίδιο χαρακτήρα στην συνάρτηση απ’ όπου καλέστηκε και η write_ch εκτυπώνει τον χαρακτήρα που δέχεται ως όρισμα. Αφού αποθηκευτεί όλη η συμβολοσειρά που έχει πληκτρολογήσει ο χρήστης, στην συνέχεια γίνεται η μετατροπή όλων των χαρακτήρων σε κεφαλαία και αποστέλλεται στην print_string.
Η print_string στέλνει ως ορίσματα έναν-έναν τους χαρακτήρες της συμβολοσειράς στην συνάρτηση write_ch η οποία και τους εκτυπώνει.

ΣΥΜΠΕΡΑΣΜΑ

Σε αυτό το εργαστηρίο καταλάβαμε πως επιτυγχάνεται η επικοινωνία του επεξεργαστή με δυο περιφεριακές συσκευές(το πληκτρολόγιο σαν είσοδο και την οθόνη σαν έξοδο), ενω δουλεύοντας την τεχνική polling εξοικειωθήκαμε με το πώς δουλεύει η μνήμη και οι καταχωρήτες στην Assembly του MIPS.

ΚΩΔΙΚΑΣ

```
.data
.globl main
.globl inputString
.globl outputString
.globl Print_string
.globl Read_string
.globl Write_ch
.globl Read_ch
.globl kefalaia

inputString: .asciiz "\ndwse String: "
outputString: .asciiz "\napotelesma: "
string:
.align 1
.space 100

.text
main:
la $a0, inputString
jal Print_string

jal Read_string

la $a0, outputString
jal Print_string

jal kefalaia

la $a0, string
jal Print_string

li $v0, 10
syscall

#------------------------------------------------------------
Print_string:
addiu $sp, $sp, -8
sw $ra, 0($sp)
sw $s0, 4($sp)

move $s0, $a0                          #string gia ektipwsh
printLoop:
lb $a0, ($s0)                          #pernei ena char
beq $a0, 0, exitPrint                 #if char = \0 stop print

jal Write_ch

addi $s0, $s0, 1                       #paei sto epomeno char
j printLoop
exitPrint:
```

```
lw $s0, -4($sp)
lw $ra, 0($sp)
addiu  $sp, $sp, 8
jr $ra

#-------------------------------------------------------------------------
-------------------------------------------------------------------------
-----------------
Write_ch:
checkWrite:
lw $t0, 0xffff0008($zero)
and $t0, $t0, 0x00000001                    #apomonosh ready bit

beq $t0, 0x00000001, write                  #if ready bit = 1 write char
j checkWrite

write:
sb $a0, 0xffff000c($zero)                   #fortoma tou transmiter data

jr $ra

#-------------------------------------------------------------------------
-------------------------------------------------------------------------
-----------------
Read_string:
addiu $sp, $sp, -8
sw $ra, 0($sp)
sw $s0, 4($sp)

la $s0, string                              #dieu8unsh string

writeLoop:
jal Read_ch
sb $v0, 0($s0)                              #apo8ikeush char

move $a0, $v0                               #ektipwsh char
jal Write_ch

lb $t0, 0($s0)
beq $t0, 10, exitRead                        #if char = \n stop write

addi $s0, $s0, 1
j writeLoop

exitRead:
li $t0, 0
sb $t0, 0($s0)                             #an path8ei to enter bazw to \0
sth 8esh tou gia na douleuei swsta

lw $s0, -4($sp)
lw $ra, 0($sp)
addiu  $sp, $sp, 8
```

```
jr $ra

#-----------------------------------------------------------------------
-----------------------------------------------------------------------
------------------
Read_ch:
checkRead:
lw $t0, 0xffff0000($zero)
and $t0, $t0, 0x00000001                    #apomonosh ready bit

beq $t0, 0x00000001, read                   #if ready bit = 1 read char
j checkRead

read:
lb $v0, 0xffff0004($zero)                   #diavasma tou transmiter data

jr $ra

#-----------------------------------------------------------------------
-----------------------------------------------------------------------
------------------
kefalaia:

la $t0, string

arxh:
lb $t1, 0($t0)

bgt $t1, 122, check                         #an o char den einai pezo gramma
na paei sto epomeno
blt $t1, 97, check

addi $t1, $t1, -32                         #me -32 ta peza ginontai kefalaia
sb $t1, 0($t0)
check:

beq $t1, 0, telos                          #an o char einai /0 teleiwnei
addi $t0, $t0, 1
j arxh

telos:
jr $ra
```