1η Άσκηση Στην Αρχιτεκτονική Υπολογιστών

Ακ. Έτος 2013 – 2014, 5ο Εξάμηνο, Σχολή ΗΜ&ΜΥ

Τμήμα Λ - Ω

Ημερομηνία Παράδοσης: 20/12/2013

Απορίες στο: ca2013-2014t2@cslab.ece.ntua.gr

Μέρος Α

Δίνονται τα παρακάτω δύο προγράμματα γραμμένα σε C, καθώς και η αντίστοιχη μετάφρασή τους σε assembly MIPS. Ωστόσο υπάρχουν **ορισμένα κενά τα οποία χρίζουν συμπλήρωσης** προκειμένου ο κώδικας να τρέχει πλέον στη μηχανή MIPS.

Υπενθυμίζουμε ότι ο **$0** ή **$zero** περιέχει ως τιμή πάντα το 0 και ότι οι εντολές **sll** και **srl** πραγματοποιούν shift αριστερά ή δεξιά αντίστοιχα, τόσες θέσεις όσες ορίζει ο τελευταίος τελεστέος της εντολής.

```
int i=0;
int result = -1;

while ( i<10 ){
    if ( result < arr[i] ){
      result=arr[i];
      result/=2;
      arr[i] -=result;
    }
    I++;
}
```

```
        add   $3     $0          $zero
        addi  $7     $zero       -1
        addi  $8     $zero       40
loop:   bge   $3     $8          done
        lw    $9     400($3)
        bge   $7     $9          go_on
        addi  $7     $9          $0
        srl   $7     $7          1
        sub   $9     $9          $7
        sw    $9     400($3)
go_on:  addi  $3     $3          4
        j     loop
done:   halt
```

```
while (((x>>2) & 0x01)!=0)
{
    x = x>>1;
    count *=2;
    if (count<10)
       break;
}
/**
*  Δίνεται ότι η μεταβλητή
*  x
*  είναι στον $2 και η count
*  στον $1
 **/
```

```
        addi    $3      $0      1
        addi    $7      $0      10
while:  srl     $4      $2      2
        and     $5      $4      $3
        beq     $5      $0      finish
        srl     $2      $2      1
        sll     $1      $1      1
        blt     $1      $7      finish
        j       while
finish: halt
```

Μέρος Β

Έχετε διεξάγει ένα πείραμα λαμβάνοντας 15 μετρήσεις, διαφορετικού είδους και καταγράφοντας και τα αντίστοιχα σφάλματα ως αριθμούς κινητής υποδιαστολής. Σφάλματα και μετρήσεις έχουν αποθηκευτεί σε 2 μονοδιάστατους πίνακες στη μνήμη, σε πλήρη αντιστοιχία. Αυτό σημαίνει ότι έστω M ο πίνακας μετρήσεων και S ο πίνακας σφαλμάτων, τότε το σφάλμα της μέτρησης M[k} θα είναι το S[k]. **Οι δύο πίνακες βρίσκονται διαδοχικά και συνεχόμενα στη μνήμη.**

Κατασκευάστε πρόγραμμα σε assembly MIPS το οποίο:

1. Σε περίπτωση που η μέτρηση είναι μικρότερη του αντίστοιχου σφάλματος, η μέτρηση θα αντικαθίσταται από το μηδέν
2. Εντοπίζει ποια μέτρηση (k) είναι εκείνη στην οποία εντοπίζεται το μέγιστο σφάλμα $(\{k: S[k] > S[i] \; \forall \; i \in [0,15), \; i \neq k \})$
3. Παράγει δύο **συνεχόμενους** πίνακες:
   a. Ο πρώτος (A) θα περιέχει τη μέγιστη δυνατή τιμή κάθε μέτρησης η οποία προφανώς ορίζεται ως A[i] = M[i]+S[i]
   b. Ο δεύτερος (B) την ελάχιστη τιμή που ορίζεται ως B[i] = M[i]-S[i]

Σημειώνεται ότι ο πίνακας που κρατά τις μετρήσεις ξεκινάει από τη θέση **400** ενώ ο πίνακας A ξεκινά από τη θέση μνήμης **800**.

Να παραδοθεί ο κώδικας των τριών μερών.

Λύση:

```
        addi $t1, $zero, 14
        mtc1 $zero, $f2          # μηδενίζουμε τον καταχωρητή
                                 #μέγιστου σφάλματος
        add $t3, $zero, $zero    # μηδενίζουμε τον καταχωρητή
                                 #που κρατάει την τιμή του μέγιστου
                                 #σφάλματος
zero_check:
        add $t2 , $t1, $zero
        sll $t2, $t2, 2
        lwc1 $f0, 400($t2)
        lwc1 $f1, 460($t2)       #15*4 = 60 byte παρακάτω είναι
                                 # το πρώτο σφάλμα
        c.lt.s $f1, $f0
        bc1t sfalma_mikrotero    # σφάλμα < μέτρσηση ?
        mtc1 $zero, $f0
        swc1 $f0, 400($t2)
sfalma_mikrotero:
        c.lt.s $f1, $f2          # σφάλμα < current_max_σφάλμα ?
        bc1t check_next
        mov.s $f2, $f1
        add $t3, $t1, $zero      #επιλέγουμε τον $t3 ως τον δείκτη στη
                                 #μέτρηση με το μεγαλύτερο σφάλμα
check_next:
        add.s $f4, $f0, $f1
        swc1 $f4, 800($t2)
        sub.s $f4, $f0, $f1
        swc1 $f4, 860($t2)
        addi $t1, $t1, -1
        bge $t1, $zero, zero_check
        halt
```

Μέρος Γ

Να γραφεί ρουτίνα σε MIPS η οποία δέχεται ως παραμέτρους ένα χαρακτήρα (στον καταχωρητή $a0) και την αρχική διεύθυνση ενός πίνακα S (στον καταχωρητή $a1). Ο πίνακας περιέχει 100 χαρακτήρες. Η ρουτίνα διατρέχει τον πίνακα S και επιστρέφει στον καταχωρητή $v0 το πλήθος των εμφανίσεων του χαρακτήρα, που λήφθηκε ως όρισμα μέσω του $a0.

Λύση:

```
proc:   addi $sp, $sp, -4          # μεγαλώνουμε τη στοίβα
        sw $ra, 0($sp)             # αποθηκεύουμε τη διεύθυνση επιστροφής
        addi $t1, $zero, 99        #μετράμε από 0 έως 99 → δίνει 100
        add $s0, $zero, $zero      #μετρητής πλήθους εμφανίσεων χαρακτήρα
loop:
        lb $s1, 0($a1)
        bne $s1, $a0 go_on
        addi $s0, $s0, 1
go_on:
        addi $t1, $t1, -1
        addi $a1, $a1, 1
        bge $t1, $zero, loop
        nop
exit:
        add $v0, $s0, $zero        #επιστροφή αποτελέσματος στο $v0
        lw $ra, 0($sp)             #ανάκτηση της διεύθυνσης επιστροφής
        addi $sp, $sp, 4           #μείωση στοίβας
        jr $ra                     #επιστροφή στο σημείο κλήσης
        nop
```

Σημείωση: Για τη διευκόλυνσή σας, προτρέπουμε να χρησιμοποιήσετε τον προσομοιωτή Qtspim (http://spimsimulator.sourceforge.net/)

Παραδοτέο της άσκησης θα είναι ηλεκτρονικό κείμενο (pdf, docs ή odt) που θα περιέχει τις απαντήσεις των τριών μερών**. Στο ηλεκτρονικό κείμενο να αναφέρεται στην αρχή τα στοιχεία σας (όνομα, επώνυμο, ΑΜ).** Το όνομα του παραδοτέου αρχείου θα πρέπει να είναι στη μορφή **επίθετοXXXXXXXX** όπου Χ τα ψηφία του αριθμού μητρώου σας.

Οι κώδικες που θα παραδοθούν ως απαντήσεις στο Β και Γ μέρος, οφείλουν να φέρουν αναλυτικά σχόλια για την κατανόηση της άσκησης από τους διδάσκοντες.

Παράδοση στο: http://www.cslab.ece.ntua.gr/courses/comparch/submit