2ο Εργαστήριο Αρχιτεκτονικής Η/Υ: MIPS assembly: Αλφαριθμητικά (strings) και επαναλήψεις

Α. Ευθυμίου

Παραδοτέο: Τετάρτη 8 Μάρτη 2017, 23:00

Το αντικείμενο αυτής της άσκησης είναι ένα πρόγραμμα που διατρέχει τα στοιχεία ενός αλφαριθμητικού (string) και ψάχνει τη μεγαλύτερη λέξη μέσα σε αυτό.

Θα πρέπει να έχετε μελετήσει τα μαθήματα για τη γλώσσα assembly του MIPS που αντιστοιχούν μέχρι και την 4η διάλεξη.

Σε αρχικό μάθημα εξηγήθηκε η σημασία της ακαδημαϊκής δεοντολογίας. Ο κώδικας που θα παραδώσετε θα πρέπει να είναι αποτέλεσμα ατομικής εργασίας. Τα προγράμματα που θα παραδοθούν θα ελεγχθούν για ομοιώτητα με ειδικό λογισμικό. Επιτρέπεται, και είνα θεμιτή, η συνεργασία και ανταλλαγή πληροφοριών σε πρακτικά θέματα, για παράδειγμα σύνταξη εντολών assembly, λύση προβλημάτων σχετικών με το git, αλλά όχι η συνεργασία στη συγγραφή κώδικα.

Μή ξεχάσετε να επιστρέψετε τα παραδοτέα που αναφέρονται στο τέλος του κειμένου για να πάρετε βαθμό γι'αυτή την εργαστηριακή άσκηση!

1 Αλφαριθμητικά, strings σε assembly

Ένα αλφαριθμητικό είναι ένας πίνακας χαρακτήρων αποθηκευμένος στη μνήμη. Οι χαρακτήρες είναι bytes και χρησιμοποιούν την κωδικοποίηση ASCII. Το τέλος του αλφαριθμητικού παριστάνεται με ένα byte που έχει την τιμή 0.

Για την άσκηση θεωρούμε ότι μια λέξη στο αλφαριθμητικό είναι μια συνεχόμενη ακολουθία από χαρακτήρες, πρίν από τους οποίους βρίσκεται ο κενός χαρακτήρας (κωδικός 32_{ten}) ή η αρχή του αλφαριθμητικού και μετά από τους χαρακτήρες υπάρχει το κενό ή τελειώνει το αλφαριθμητικό. Οι λέξεις δεν μπορούν να έχουν κενά μέσα τους. Δεν θα χωρίζουν οι λέξεις με σημεία στίξης ή άλλους ειδικούς χαρακτήρες. Θα μπορούν όμως να υπάρχουν συνεχόμενα κενά, κενά στην αρχή ή το τέλος του αλφαριθμητικού και μπορεί ένα αλφαριθμητικό να είναι άδειο, δηλαδή ο πρώτος (και τελευταίος) χαρακτήρας του να έχει την τιμή 0.

Προσοχή μην μπερδεύτε την έννοια της λέξης, ως μιας σειράς χαρακτήρων/γραμμάτων, με αυτή της λέξης ενός υπολογιστή (32 bit στο μάθημα)!

2 Περιγραφή του αλγόριθμου

Δίνεται η διεύθυνση μνήμης του αλφαριθμητικού στον καταχωρητή `$a0`. Ο πρώτος χαρακτήρας θα βρίσκεται σε αυτή την διεύθυνση, αλλά μπορεί να έχει τιμή 0, που σημαίνει ότι το αλφαριθμητικό είναι άδειο. Οι επόμενοι χαρακτήρες θα βρίσκονται σε διευθύνσεις με αύξουσα τιμή.

Όταν βρεθεί η μεγαλύτερη λέξη, θα πρέπει να επιστραφεί η διεύθυνση μνήμης από όπου ξεκινάει στον καταχωρητή `$v1`. Αν υπάρχουν πολλές λέξεις με το μέγιστο μέγεθος, θα πρέπει να επιστραφεί η τελευταία από αυτές. Αν το αλφαριθμητικό είναι άδειο, ως διεύθυνση της μεγαλύτερης λέξης θα επιστρέφεται η διεύθυνση του αλφαριθμητικού.

Σε γενικές γραμμές ο αλγόριθμος διατρέχει το αλφαριθμητικό από την αρχή προς το τέλος και «θυμάται» την αρχή τής μέχρι τώρα μεγαλύτερης λέξης και την αρχή της τρέχουσας λέξης. Όταν ολοκληρωθεί η τρέχουσα λέξη γίνεται έλεγχος αν είναι μεγαλύτερη από αυτή που ήδη γνωρίζουμε. Όταν βρεθεί ο ειδικός χαρακτήρας τέλους (τιμή 0), τελειώνει και η τελευταία λέξη και μπορούμε να τερματίσουμε τον αλγόριθμο.

Επειδή είναι δύσκολο να γράψει κανείς προγράμματα όπου δεν μπορεί να καθορίσει τα ονόματα των μεταβλητών, μπορείτε, αρχικά, να γράψετε ένα είδος ψευτοκώδικα assembly με κανονικές εντολές MIPS αλλά με μεταβλητές, με κατάλληλα ονόματα, αντί για καταχωρητές. Μετά αντιστοιχίστε τις μεταβλητές

σε καταχωρητές, κρατώντας την πληροφορία σε σχόλια μέσα στον κώδικα, και κάντε την αντικατάσταση.

Αφού καταλήξετε σε μια υλοποίηση που δίνει σωστά αποτελέσματα, προσπαθείστε να ελαχιστοποιήσετε τον αριθμό εντολών που εκτελούνται.

3 Σκελετός προγράμματος

Για να πάρετε τα αρχεία της εργαστηριακής άσκησης, μεταβείτε στον κατάλογο εργασίας που είχατε κλωνοποιήσει από το αποθετήριό σας στο GitHub. (cd <όνομα χρήστη GitHub>-labs). Μετά, κάνετε τα παρακάτω βήματα:

```
git remote add lab02_starter https://github.com/UoI-CSE-MYY402/lab02_starter.git
git fetch lab02_starter
git merge lab02_starter/master -m "Fetched lab02 starter files"
```

Θα δείτε ότι θα εμφανιστεί ένας κατάλογος lab02. Μεταβείτε σε αυτόν: `cd lab02`. Εκεί θα βρείτε το αρχείο `lab02.asm`, που περιέχει το σκελετό του προγράμματος.

Θα δείτε ότι περιέχει λίγες γραμμές που:

- διαβάζουν την διεύθυνση του αλφαριθμητικού στον καταχωρητή `$a0` (γραμμή 12)
- αρχικοποιούν τη διεύθυνση της μεγαλύτερης λέξης στον καταχωρητή `$v1` (γραμμή 13)
- τελειώνουν την εκτέλεση με ένα syscall (γραμμές 22-23)

Τα σχόλια δείχνουν σε ποιό σημείο θα γράψετε τον δικό σας κώδικα.

4 MARS Debugging

Τώρα που τα προγράμματά σας γίνονται μεγαλύτερα και πιο πολύπλοκα, ο απλοϊκός τρόπος αποσφαλμάτωσης με εκτέλεση εντολή προς εντολή, δεν είναι πλέον αποδοτικός. Αν αντιμετωπίζετε ένα πρόβλημα που εμφανίζεται σχετικά αργά στην εκτέλεση του προγράμματος θα πρέπει να εκτελέσετε εντολή προς εντολή για πολλή ώρα μέχρι να το βρείτε.

Για να αποφύγετε αυτή τη ταλαιπωρία, μπορείτε να χρησιμοποιήσετε breakpoints - ειδικά σημάδια του προσομοιωτή που σταματούν την εκτέλεση όταν έρθει η σειρά της εντολής όπου βάλατε το breakpoint. Πριν προσθέσετε ένα breakpoint, πρέπει να κάνετε assemble τον κώδικά σας. Αν δεν υπάρχουν λάθη, στο execute tab, παράθυρο Text Segment, που περιέχει το πρόγραμμά σας, θα δείτε αριστερά από κάθε εντολή ένα check-box. Σε όποια εντολή θέλετε να βάλετε breakpoint κάντε κλικ στο check-box ώστε να είναι σημαδεμένο. Από εδώ και πέρα λίγο πρίν εκτελεστεί αυτή η εντολή, ο MARS θα σταματάει και θα μπορείτε να δείτε τις τιμές καταχωρητών και μνήμης. Επίσης θα μπορείτε να συνεχίσετε την εκτέλεση, είτε εντολή προς εντολή ώστε να βρείτε το λάθος, είτε μέχρι το επόμενο breakpoint.

5 Παραδοτέο

Το μόνο παραδοτέο είναι το αρχείο lab02.asm. Στον κατάλογο test θα βρείτε το `lab02_tester.py` που εξετάζει 5 περιπτώσεις που το πρόγραμμά σας θα πρέπει να αντιμετωπίζει με επιτυχία. Δεν χρειάζεται να αλλάξετε κάποια αναμενόμενη τιμή ή κάτι άλλο σε αυτό το αρχείο για την άσκηση.

Πρέπει να κάνετε commit τις αλλαγές σας και να τις στείλετε (push) στο αποθετήριό σας στο GitHub για να βαθμολογηθούν πριν από την καταληκτική ημερομηνία!

Τα προγράμματά σας θα βαθμολογηθούν για την ορθότητά τους, την ποιότητα σχολίων και τη ταχύτητα εκτέλεσής τους. Το τελευταίο σημαίνει ότι πρέπει να είναι σύντομα και ο αριθμός εντολών, ειδικά μέσα σε βρόγχο, να είναι όσο γίνεται μικρότερος.