3η Εργαστηριακή Ασκήσεων
Σχεδιασμός Ενσωματωμένων Συστημάτων

Λεωνίδας Αβδελάς | ΑΜ: 03113182

Ερώτημα 1ο:
Μετατροπή εισόδου από τερματικό

Για την άσκηση αυτή, χρησιμοποιήσαμε system calls, αφού ήταν αρκετά πιο εύκολα στην χρήση. Προσπαθήσουμε να σπάσουμε το πρόγραμμα σε διακριτά κομμάτια για να είναι πιο εύκολα στην διαχείρηση. Το πρόγραμμα ξεκινάει γράφοντας στο stdout και διαβάζοντας την είσοδο του χρήστη. Έπειτα κάνει τα παρακάτω:

- Ελέγχει αν το μήκος της εισόδου είναι 2 (ένας χαρακτήρας και το EOL). Αν είναι ελέγχει αν ο χαρακτήρας είναι q ή Q. Αν είναι καλεί το system call, exit για να τερματίσει.
- Αν δεν είναι περνάει στην συνάρτηση reformat string.

Η συνάρτηση reformat string δέχεται ως όρισμα στον **r5** το μήκος της συμβολοσειράς. Αυτό δεν είναι σύμφωνο με τα γνωστά conventions για συναρτήσεις σε ARM, αλλά μπορεί εύκολα να αλλάξει αν χρειαστεί.

Η συνάρτηση διατρέχει τους χαρακτήρες που υπάρχουν στην θέση μνήμης του **r1**, μέχρι να είναι το **r5** ίσο με 0. Όσο το κάνει αυτό, φορτώνει το στοιχείο που είναι στην θέση μνήμης **r1** στον **r0**. Έπειτα κάνει τα παρακάτω:

- Ελέγχει αν είναι μεγαλύτερο του ASCII χαρακτήρα 9. Αν είναι, τότε υποθέτει ότι είναι γράμμα. Αν δεν είναι ελέγχει αν είναι μεγαλύτερο από τον χαρακτήρα 0. Αν είναι τότε είναι αριθμός. Αν δεν είναι τότε, δεν τον αλλάζουμε αυτό τον χαρακτήρα και πηγαίνουμε στον επόμενο.
- Αν ο χαρακτήρας είναι αριθμός, ο νέος αριθμός που θέλουμε είναι ο αρχικός επαυξημένος κάτα 5 και mod 10. Αφού ξέρουμε ότι το ψηφίο δεν θα είναι ποτέ πάνω από 15, μπορούμε να βρούμε τον αρχικό αριθμό αφαιρώντας τον χαρακτήρα 0 σε ASCII και κάνοντας την πρόσθεση με 5. Αν ο αριθμός είναι μεγαλύτερος από 10, τότε αφαιρούμε το 10 για να φτάσουμε στο επιθυμητό αποτέλεσμα. Τέλος προσθέτουμε τον χαρακτήρα 0 ξανά για να επανακατασκευάσουμε τον χαρακτήρα μας.
- Αν ο χαρακτήρας είναι μεταξύ 65 και 90, τότε είναι κεφαλαίο γράμμα και προσθέτουμε 32 για να το κάνουμε μικρό.
- Αν ο χαρακτήρας είναι μεταξύ 97 και 122, τότε είναι μικρό γράμμα, οπότε αφαιρούμε 32 για να το κάνουμε κεφαλαίο.

Τέλος ελέγχουμε αν η συμβολοσειρά είναι ακριβώς 32 χαρακτήρες, τότε αν ο τελυεταίος χαρακτήρας δεν είναι EOL, προσθέτουμε ένα χαρακτήρα στο τέλος, ο οποίος είναι EOL.

Έπειτα αφαιρούμε όλους τους υπόλοιπους χαρακτήρες από το stdin. Αυτό το κάνουμε πάλι με read(). Αν στο read πάρουμε λιγότερους από 32 χαρακτήρες, τότε το stdin είναι καθαρό. Αλλιώς ελέγχουμε την οριακή περίπτωση που είναι ακριβώς 32 χαρακτήρες αυτοί που απομένουν. Σε αυτή την περίπτωση ο τελευταίος χαρακτήρας θα ήταν EOL, οπότε αν είναι επιστρέφουμε στην αρχή του προγράμματος, αλλιώς διαβάζουμε και άλλο.

Ερώτημα 2ο:
Επικοινωνία των guest και host μηχανημάτων μέσω σειριακής θύρας.

Το ερώτημα αυτό ήταν το πιο πολύπλοκο της εργασίας με διαφορά. Για την επικοινωνία μεταξύ των δύο μηχανημάτων και την ρύθμιση του διάυλου, χρησιμοποιήσαμε την βιβλιοθήκη *termios*. Στο guest μηχάνημα, αφού τρέχαμε σε assembly, καλέσαμε την εξωτερική συνάρτηση **tcsetattr**. Χρησιμοποιήσαμε canonical κατάσταση στον δίαυλο, αφού αυτό μας επέτρεπε με ευκολία να μεταφέρουμε την συμβολοσειρά μέχρι το EOL έυκολα. Μετά από διάφορες δοκιμές στις ρυθμίσεις, καταλήξαμε σε αυτές που έκαναν το σύστημα να δουλεύει αξιόπιστα.

Για να είναι βέβαιη η επικοινωνία, τρέχαμε πάντα τον guest πριν τον host.

Αρχικά ανοίγουμε την θύρα σε non-blocking mode, αλλά μετά την μετατρέπουμε σε blocking από την μερία του host. Για τον guest την ανοίγουμε κατευθείαν σε blocking. Αυτό μας βοήθησε να αποφύγουμε κάποια ανεξήγητα (για εμάς) σφάλματα.

Η μέτρηση χαρακτήρων γίνεται μέσω θέσεων μνήμης, κάθε μια από τις οποίες αντιστοιχεί σε έναν χαρακτήρα ASCII. Τις αρχικοποιούμε σε χαρακτήρα 0, γιατί αλλιώς είχαμε προβλήματα (τα οποία πάλι δεν μπορούσαμε να εξηγήσουμε, αλλά μάλλον σχετίζονται με το γεγονός ότι η θέσεις μνήμης ήταν κενές και ο χαρακτήρας \0 δεν αντιστοιχούσε στην μηδενική τιμή όπως περιμέναμε).

Ο guest διατρέχει την συμβολοσειρά που έστειλε ο host και αυξάνει σε ένα πίνακα τις τιμές εμφάνισης του χαρακτήρα που βρήκε. Μετά διατρέχει τον πίνακα και κρατάει το offset του πίνακα (το οποίο είναι το ASCII του χαρακτήρα) και την τιμή για αυτόν που έχει την μέγιστη εμφάνιση.

Τέλος στέλνει την τριπλέτα (χαρακτήρας, μήκος συμβολοσειράς, εμφανίσεις) για την συμβολοσειρά που στείλαμε.

Καθώς ένα byte μπορεί να έχει τιμή μέχρι 127 δεν ανησυχούμε μήπως κάποια από τις τιμές δεν χωράει σε αυτό, αφού το μήκος της συμβολοσειράς είναι 64 χαρακτήρες το πολύ και ακόμα και αν η μέτρηση ξεκινάει από τον χαρακτήρα 0, με κωδικό 48, πάλι η μέγιστη τιμή δεν φτάνεται.

Δυστυχώς το πρόγραμμα μας δεν παρέχει κάποια ασφάλεια για μεγαλύτερο μήκος συμβολοσειράς, αλλά ο χρήστης θα πρέπει να προσέχει, καθώς τα αποτελέσματα είναι undefined.

Ερώτημα 3ο: Σύνδεση κώδικα C με κώδικα assembly του επεξεργαστή ARM.

Εδώ κάθε συνάρτηση είναι αρκετά μικρή, οπότε είναι εύκολα κατανοητή από τον αναγνώστη, αν έχει υπόψιν ότι οι συμβολοσειρές τελειώνουν με χαρακτήρα \0 και

καταλαμβάνουν διαδοχικές τιμές στην μνήμη. Για όλες τις συναρτήσεις περιλαμβάνονται και κάποιοι έλεγχοι εγκυρότητας αποτελέσματος.