

**ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ**

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΕΡΓΑΣΤΗΡΙΟ ΗΛΕΚΤΡΟΝΙΚΗΣ

**Σχεδιασμός Ενσωματωμένων Συστημάτων**

9ο εξάμηνο

4η Εργαστηριακή Άσκηση:

**Εργασία σε assembly του επεξεργαστή ARM**

03114681: Maliganis Nikolaos

# Μετατροπή εισόδου από τερματικό

Σε αυτό το μέρος της άσκησης θα αναπτυχθεί προγράμματος σε assembly του επεξεργαστή ARM, το οποίο λαμβάνει ως είσοδο συμβολοσειρά 32 χαρακτήρων και θα τη μετασχηματίζει σύμφωνα με τα παρακάτω:

Στην περίπτωση κατά την οποία η είσοδος είναι γράμμα του λατινικού αλφαβήτου, θα μετασχηματίζεται σε:

* Θα μετατρέπει τα πεζά γράμματα σε κεφαλαία.
* Θα μετατρέπει τα κεφαλαία σε πεζά.
* Σε περίπτωση όπου ο χαρακτήρας βρίσκεται στο εύρος [‘0’, ‘9’]:
  + Έλεγχος εάν η είσοδος είναι μικρότερη του 5, θα προσθέτει 5.
  + Έλεγχος εάν η είσοδος είναι μεγαλύτερη ή ίση του 5, θα αφαιρεί 5.
* Θα παραμένουν οι χαρακτήρες αμετάβλητοι σε κάθε άλλη περίπτωση.
* Θα τερματίζει την λειτουργία σε περίπτωση εισόδου συμβολοσειράς μήκους μονάδας και χαρακτήρα ‘Q’ ή ‘q’.
* Σε όλες τις άλλες περιπτώσεις θα συνεχίζει η λειτουργία ατέρμον.

Για την υλοποίηση του συγκεκριμένου προγράμματος έχουμε τις εξής εκτελέσεις:

* Λήψη εισόδου από τον χρήστη. Αυτό υλοποιείται με την χρήση της write system call (r7=4).
* Ανάγνωση με τη χρήση της read system call (r7=3), διαβάζοντας ως είσοδο συμβολοσειρά μήκους 33 χαρακτήρων (33-Bytes) και με τη χρήση της stdin, 33+EOL
* Ανάγνωση και έλεγχος του μήκους σε περίπτωση συμβολοσειράς μήκους 2 χαρακτήρων, για τις περιπτώσεις Q+1 ή q+1, όπου 1 EOL.
* Έλεγχος πάντα του χαρακτήρα στην πρώτη θέση και σύγκριση ελέγχου για περιπτώσεις τερματισμού, ‘Q’ ή ‘q’.
* Έλεγχος της αποθηκευμένης συμβολοσειράς και του αποθηκευμένου χαρακτήρα για τον απαραίτητο μετασχηματισμό. Η θέση αυτή είναι η r1
* Αναγνώριση χαρακτήρα και έλεγχος διαστήματος μεταξύ [’a’, ’z’], όπου εκτελείται η πράξη της αφαίρεσης σε ASCII Code του κωδικού 32, ώστε να υλοποιηθεί ο μετασχηματισμός του Upper-Case, και αντικατάστασης στην τρέχουσα θέση μνήμης.
* Αναγνώριση χαρακτήρα και έλεγχος διαστήματος μεταξύ [’Α’, ’Ζ’], όπου εκτελείται η πράξη της πρόσθεσης σε ASCII Code του κωδικού 32, ώστε να υλοποιηθεί ο μετασχηματισμός του Lower-Case, και αντικατάστασης του αποτελέσματος στην τρέχουσα θέση μνήμης.
* Αναγνώριση χαρακτήρα και έλεγχος διαστήματος μεταξύ [’Α’, ’Ζ’], όπου εκτελείται η πράξη της πρόσθεσης σε ASCII Code του κωδικού 32, ώστε να υλοποιηθεί ο μετασχηματισμός του Lower-Case, και αντικατάστασης του αποτελέσματος στην τρέχουσα θέση μνήμης.
* Αναγνώριση χαρακτήρα και έλεγχος διαστήματος μεταξύ [’0’, ’9’], πραγματοποιείται η πράξη της σύγκρισης με τον αριθμό 5 και σε περίπτωση μικρότερου θα υλοποιηθεί η πράξη της πρόσθεσης σε ASCII Code του αριθμού 5 ενώ σε διαφορετική περίπτωση θα υλοποιηθεί η πράξη της αφαίρεσης σε ASCII Code του αριθμού 5 και αντικατάστασης του αποτελέσματος στην τρέχουσα θέση μνήμης.
* Εκτέλεση επανάληψης των τεσσάρων παραπάνω αναγνωρίσεων έως ότου γίνει εντοπισμός του χαρακτήρα ASCII κωδικού 10 που συμβολίζει χαρακτήρα αλλαγή γραμμής σε περιπτώσεις μέτρησης λιγότερων των 32 χαρακτήρων, καθώς επίσης και στην περίπτωση μέτρησης 33 χαρακτήρων.
* Εκτύπωση της μετασχηματισμένης συμβολοσειράς
* Επιπλέον, σε περιπτώσεις που η συμβολοσειρά εισόδου περιέχει λιγότερους από 32 χαρακτήρες μπορεί με κλήση της write system call να τυπωθεί στο έξοδο και στο stdout το σύνολο των χαρακτήρων που είναι αποθηκευμένο στο r4.

Υπάρχουν συνοδευτικοί κώδικες.

# Μετατροπή εισόδου από τερματικό

**2.1 Εισαγωγή**

Στο δεύτερου μέρους της άσκησης θα υλοποιηθούν να δημιουργηθούν 2 προγράμματα, ένα σε C στο host μηχάνημα και ένα σε assembly του ARM στο guest μηχάνημα τα οποία θα επικοινωνούν μέσω εικονικής σειριακής θύρας.

Για την υλοποίηση των συγκεκριμένων προγραμμάτων, έχουμε τις εξής εκτελέσεις:

* Λήψη εισόδου string μεγέθους έως 64 χαρακτήρων.
* Το string αυτό αποστέλλεται μέσω σειριακής θύρας στο guest μηχάνημα
* Γίνεται υπολογισμός και απάντησης ποιος από τους εισερχομένους χαρακτήρες του string εμφανίζεται με την μεγαλύτερη συχνότητα εμφάνισης και το πλήθος που εμφανίστηκε.
* Στην περίπτωση που δύο ή παραπάνω χαρακτήρες έχουν μέγιστη συχνότητα εμφάνισης, το πρόγραμμα επιστρέφει στον host, τον χαρακτήρα με τον μικρότερο ascii κωδικό
* Στην παραπάνω περίπτωση να εξαιρείται ο κενός χαρακτήρας.
* Εκτέλεση qemu και ενεργοποίηση της εικονικής σειριακής /dev/pts/x.
* Ενεργοποίηση του tty και χρήση της βιβλιοθήκης termios.
* Άνοιγμα της σειριακής θήρας και από τον host και τον guest και κατάλληλη ρύθμιση μέσα από το configuration (baud Rate e.t.c.)
* Κοινές ρυθμίσεις και για τα 2 παραπάνω Components

**2.2.1 Host**

1. Αρχικά,ελέγχουμεότιγίνεταισωστήεκτέλεσητουαρχείουκαιδιαβάζουμεαπότοstdinτησυμβολοσειρά που θα στείλουμε στον guest. Πρέπει να έχει μήκος μέχρι 64 bytes (65 bytes αν λάβουμε υπόψη μας και τον ειδικό χαρακτήρα FL).

Υπάρχουν συνοδευτικοί κώδικες.

# Σύνδεση κώδικα C με κώδικα assembly του επεξεργαστή ARM

Στο 3ο μέρος της άσκησης θα γίνει συνδυασμός συναρτήσεων και κώδικας στη γλώσσα C με κλήσεις μέσα από assembly του ARM. Θα υλοποιηθούν σε ARM assembly 4 συναρτήσεις C, οι οποίες είναι: strlen, strcpy, strcmp και strcat στο header file string.h.

Όλες οι παραπάνω συναρτήσεις στην Assembly θα χρησιμοποιηθούν στην *string\_manipulation.c*.

Πιο συγκεκριμένα για την υλοποίηση θα αναπτυχθεί ένα αρχείο για κάθε περίπτωση:

* Συνάρτηση **size\_t strlen(const char \*s);**  η οποία θα επιστρέφει τον αριθμό των char – bytes που περιέχονται στην συμβολοσειρά. Στην **strlen.s** που υλοποιήθηκε δέχεται ως όρισμα εισόδου τον καταχωρητή r0 με τη διεύθυνση μιας εισερχομένης συμβολοσειράς και επιστρέφει στον ίδιο καταχωρητή τον αριθμό char του sizeof της συμβολοσειράς.
* Συνάρτηση **char \*strcpy(char \*s1, const char \*s2);** η οποία αντιγράφει τους χαρακτήρες chars της συμβολοσειράς στην οποία δείχνει ο pointer s2 στην συμβολοσειρά που δείχνει ο pointer s1. Σε αυτή την υλοποίηση τοποθετούμε στον καταχωρητή r0 τον pointer της s1 και στον καταχωρητή r1 τον pointer της s2 συμβολοσειράς. Η υλοποίηση πραγματοποιείται με βρόχο όπου πριν από την κάθε επανάληψη πραγματοποιείται αύξηση της διεύθυνσης προορισμού και στην συνέχεια εκτελείται η πράξη της αντιγραφής από την r1 στην r0. Τέλος αφού ολοκληρωθεί η αντιγραφή, τοποθετείται στον r0 η αρχική διεύθυνση
* Συνάρτηση **int strcmp(const char \*s1, const char \*s2);** η οποία συγκρίνει τα char της συμβολοσειράς στην οποία δείχνει ο pointer s1 με την συμβολοσειρά που δείχνει ο pointer s2. Οι επιστροφές είναι 1, 0, -1 εάν είναι μεγαλύτερη, ίση και μικρότερη αντίστοιχα. Σε αυτή την υλοποίηση τοποθετούμε στον καταχωρητή r0 τον pointer της s1 και στον καταχωρητή r1 τον pointer της s2 συμβολοσειράς. Η υλοποίηση πραγματοποιείται με βρόχο. Σε κάθε επανάληψη γίνεται έλεγχος του χαρακτήρα στη συγκεκριμένη θέση για τον καταχωρητή r0 με τον καταχωρητή r1 αυξάνοντας τον index για τον επόμενο έλεγχο.
* Συνάρτηση **char \*strcat(char \*s1, const char \*s2);** η οποία αντιγράφει τα char της συμβολοσειράς στην οποία δείχνει ο pointer s2 στην συμβολοσειρά που δείχνει ο pointer s, προσθέτοντας τους χαρακτήρες στο τέλος του

• **strcat.s**Η τελευταία συνάρτηση δέχεται ως είσοδο στον καταχωρητή r0 τη διεύθυνση μιας συμβολοσειράς s1 και στον r1 τη διεύθυνση μιας συμβολοσειράς s2 και αντιγράφει τη δεύτερη στο τέλος της πρώτης. Όπως και για την strcpy, αποθηκεύουμε την αρχική διεύθυνση της s1, στον καταχωρητή r4.  
Η υλοποίηση της συνάρτησης χωρίζεται σε δύο μέρη. Αρχικά, διατρέχουμε τη συμβολοσειρά s1 μέχρι να φτάσουμε στον τερματικό χαρακτήρα. Ειδικότερα, όπως κάναμε και για τις υπόλοιπες συναρτήσεις, διαβάζουμε έναν χαρακτήρα τη φορά και προχωράμε στην επόμενη θέση μνήμης, μέχρι να εντοπίσουμε το τέλος της συμβολοσειράς. Όταν συμβεί αυτό, ελαττώνουμε τη διεύθυνση κατά 1, επειδή επιθυμούμε να αντικαταστήσουμε το τερματικό σύμβολο με τον πρώτο χαρακτήρα της s2. Ύστερα, ακολουθούμε διαδικασία όμοια με αυτή της strcpy, διαβάζοντας έναν έναν τους χαρακτήρες της s2 και αποθηκεύοντας τους στη διεύθυνση που δείχνει ο r0, η οποία αυξάνεται κατά 1 σε κάθε επανάληψη.  
Μόλις αντιληφθούμε ότι έχει αντιγραφεί και ο τερματικός χαρακτήρας της δεύτερης συμβολοσειράς βγαίνουμε από το βρόχο. Επαναφέρουμε, τέλος, τη διεύθυνση της συμβολοσειράς που κατασκευάσαμε, η οποία ταυτίζεται με τη διεύθυνση που αποθηκεύσαμε στην αρχή στον r4 και η συνάρτηση μπορεί πλέον να επιστρέψει.