ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ Η/Υ
ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΩΝ & ΥΛΙΚΟΥ
ΕΡΓΑΣΤΗΡΙΑΚΕΣ ΑΣΚΗΣΕΙΣ ΓΙΑ ΤΟ ΜΑΘΗΜΑ:
ΗΡΥ 201 ΨΗΦΙΑΚΟΙ ΥΠΟΛΟΓΙΣΤΕΣ

ΧΕΙΜΕΡΙΝΟ ΕΞΑΜΗΝΟ 2023-2024

Εργαστήριο 4:

Υπορουτίνες, Πέρασμα Ορισμάτων και Αποτελεσμάτων, Επεξεργασία σε Επίπεδο Λέξης, Δομημένος Κώδικας

ΕΚΠΟΝΗΣΗ: Δρ. Ε.Σωτηριάδης, Ι.Πολογιώργη, Αν. Καθ. Σ.Ιωαννίδης

ΕΚΔΟΣΗ : 2.0
Χανιά

Α. Σκοπός του εργαστηρίου

Σκοπός του εργαστηρίου είναι η δημιουργία απλού, δομημένου κώδικα με main() που καλεί υπορουτίνες. Πιο συγκεκριμένα η δημιουργία κώδικα για υπορουτίνες, το πέρασμα ορισμάτων προς και αποτελεσμάτων από υπορουτίνες μέσω των καταχωρητών $a0-$a3 και $v0-$v1 αντίστοιχα, καθώς και απλή επεξεργασία δεδομένων.

Σαν κώδικας, μεγάλο μέρος του παρόντος εργαστηρίου είναι από κώδικα που ήδη έχετε, και επομένως το καινούργιο μέρος σε μεγάλο μέρος του εργαστηρίου αυτού είναι να αλλάξετε την δομή του (π.χ. υπορουτίνες) αλλά και το περιεχόμενό του μόνο στην αντικατάσταση των εντολών LB/SB με εντολές LW/SW.

Για αυτό το εργαστήριο δεν χρειάζεται η χρήση στοίβας για το $ra και τα ορίσματα.

Β. Ζητούμενα

Λειτουργικά, το εργαστήριο αυτό είναι απολύτως πανομοιότυπο με το προηγούμενο εργαστήριο. Δηλαδή, "……*θα ζητάμε από τον χρήστη να εισαγάγει ένα ένα γράμματα του λατινικού αλφαβήτου τα οποία θα τα αποθηκεύουμε στη μνήμη μέχρι ο χρήστης να εισάγει το σύμβολο @. Στην συνέχεια θα αντιγράφεται η συμβολοσειρά επεξεργασμένη στη μνήμη όπου θα αφαιρούνται τα ειδικά σύμβολα και θα μένουν μόνο οι αλφαριθμητικοί χαρακτήρες και τέλος θα τυπώνεται στην κονσόλα το περιεχόμενο της μνήμης με την επεξεργασμένη συμβολοσειρά.……"*. Όλο το παραπάνω εδάφιο είναι από το Εργαστήριο 3. Τώρα όμως θα γίνεται διαφορετικά η επεξεργασία. Για να μην επαναλαμβάνουμε την εκφώνηση του Εργαστηρίου 3, παρέχουμε μόνο συνοπτικά στοιχεία – τα υπόλοιπα δείτε τα σε εκείνη την εκφώνηση.

Γ. Περιγραφή της λειτουργίας του κώδικα

Όπως και στο Εργαστήριο 3, δεσμεύουμε χώρο 100 Bytes για καθένα από τα δύο strings, κλπ. Τώρα όμως η δομή του κώδικά μας ακολουθεί μία δομή που θα μπορούσε να περιγραφεί ως εξής:

```
main() {
init()
...
get_string(int string1, int num_characters) ;
...
process_string (int string1, int string2, int num_characters);
...
output_string(int string2);
exit();
}
```

Οι τελίτσες είναι ο απολύτως ελάχιστος κώδικας, όπως π.χ. το να βάλουμε στο $a0-$a3 ορίσματα για υπορουτίνες. Οι υπορουτίνες έχουν τα εξής ορίσματα και λειτουργικότητα:

init()

Αρχικοποίηση: εδώ (αντίθετα με τις άλλες υπορουτίνες) δεν περνάμε ορίσματα και δεν περιμένουμε απαντήσεις γιατί θεωρούμε ότι αφορούν global μεταβλητές που θα διαβάζουν οι υπορουτίνες (προσοχή: πρέπει να μπουν σε καταχωρητές τύπου $s και να μην πανωγράφονται από υπορουτίνες). Εδώ είναι οι διευθύνσεις για τις συμβολοσειρές “Please Enter Your Character:” και “The String Is:”, καθώς και ότι άλλο χρειάζεστε για να λειτουργήσει το πρόγραμμα. Γενικά είναι πολύ καλή η δομή σε κώδικα (ακόμη και υψηλού επιπέδου) όπου το main έχει init/process/output και είναι πολύ μικρό. Η init επίσης φροντίζει κάποιοι καταχωρητές να έχουν π.χ. διευθύνσεις για strings, κλπ.

get_string(int string1, int num_characters)

Στην υπορουτίνα αυτή περνάμε σαν όρισμα μέσω του $a0 την βασική διεύθυνση της μνήμης (string1) όπου θα αποθηκεύσουμε την συμβολοσειρά εισόδου. Την είσοδο την διαβάζουμε byte προς byte με το κατάλληλο syscall, όπως στο Εργαστήριο 3, αλλά δεν την αποθηκεύουμε byte προς byte όπως στο Εργαστήριο 3, αλλά με SW. Αυτό σημαίνει ότι πρέπει να «πακετάρουμε» με κώδικα τα bytes σε λέξεις, τις οποίες αποθηκεύουμε, και ταυτόχρονα να ελέγχουμε κατά πόσον ήρθε ο τερματικός χαρακτήρας @. Με δεδομένο ότι αργότερα θα εκτυπώσουμε την συμβολοσειρά, είναι αποδεκτό όταν έρθει ο χαρακτήρας @ να γράψουμε αντ’ αυτού στην μνήμη τον χαρακτήρα newline(‘\n’) ή ότι χρησιμοποιήσατε στο Εργαστήριο 3). Η υπορουτίνα αυτή επιστρέφει μέσω του $v0 τον αριθμό των χαρακτήρων της συμβολοσειράς num_characters (ο αριθμός αυτός μπορεί να περιλαμβάνει και τον τερματικό χαρακτήρα ή να μην τον περιλαμβάνει, με την κατανόηση ότι θα πρέπει να χρησιμοποιηθεί κατάλληλα και με συνέπεια στην επόμενη υπορουτίνα). Προσοχή: ο αριθμός των sw είναι η οροφή του ενός τετάρτου του αριθμού των χαρακτήρων num_characters.

process_string (int string1, int string2, int num_characters);

Στην υπορουτίνα αυτή περνάμε μέσω των $a0-$a2 την βασική διεύθυνση μνήμης για την αρχική συμβολοσειρά και για τηνσυμβολοσειρά που θα προκύψει μετά από την δική μας επεξεργασία, καθώς και τον αριθμό των χαρακτήρων της συμβολοσειράς (αλλά όχι του αριθμού των προσπελάσεων στην μνήμη, βλ. σχόλιο παραπάνω), που θα χρησιμοποιήσουμε για δημιουργία κατάλληλου βρόχου. Οι χαρακτήρες του string1 διαβάζονται λέξη-λέξη με το LW από την μνήμη, μεσολαβεί λειτουργικά η ίδια επεξεργασία όπως στο Εργαστήριο 3 *σε καταχωρητές και σε επίπεδο λέξης* για το τι κρατάμε και τι πετάμε, και γράφονται λέξη-λέξη με το SW στο string2, μαζί με τον χαρακτήρα newline στο τέλος ολόκληρης της συμβολοσειράς. Η υπορουτίνα αυτή επιστρέφει στους καταχωρητές $v0, $v1 τον νέο αριθμό χαρακτήρων που προέκυψε από την επεξεργασία, καθώς και τον ακέραιο αριθμό του κατά πόσους χαρακτήρες μειώθηκε η συμβολοσειρά εισόδου.

output_string(int string2);

Είναι η έξοδος της επεξεργασμένης συμβολοσειράς, στην οποία περνάμε μόνο στο $a0 το string2, και με το κατάλληλο syscall, όπως στο Εργαστήριο 3, δείχνουμε την έξοδο στην κονσόλα.

BONUS 15% Δημιουργία φωλιασμένης υπορουτίνας και χρήση στοίβας.

Για περαιτέρω εξάσκηση πέρα από το σκοπό της άσκησης σας ζητάμε να καλείτε την υπορουτίνα output_string από την process_string και όχι από το main(). Αυτό απαιτεί τη χρήση της στοίβας για την αποθήκευση τουλάχιστον του $ra.

Σκεφτείτε πως θα υλοποιήσετε τον κώδικα πριν το κάνετε. Ενδεικτικά, είναι λιγότερη δουλειά, πιο γρήγορος και πιο μικρός ο κώδικας αν αρχικοποιείτε κάθε καταχωρητή που θα γραφτεί στο string2 σε κάποια τιμή (ενδεικτικά μπορεί να είναι τέσσερα newline), πανωγράφετε ότι χρειάζεται, και μόλις γραφτούν τέσσερις χαρακτήρες τον αντιγράφετε στην μνήμη (και μην ξεχάσετε ότι οι δείκτες είναι διευθύνσεις λέξης και όχι byte, δηλαδή αυξάνουν κατά 4). Η παραπάνω πρόταση για κωδικοποίηση αντικαθιστά κώδικα με βρόχο που «γεμίζει» την τελευταία λέξη byte προς byte.

Το bonus ισχύει μόνο για κώδικες που δουλεύουν σωστά και είναι τεκμηριωμένοι.

Οι φωλιασμένες συναρτήσεις και η χρήση στοίβας έχουν μπει ως bonus καθώς είναι θέμα της επόμενης άσκησης. Για την άσκηση αυτή πρέπει να επικεντρωθείτε στη σωστή κλήση συναρτήσεων με την διαχείριση του $ra όπου χρειάζεται, το πέρασμα των ορισμάτων και την τήρηση των συμβάσεων.

«Λεπτομέρειες»: Εκτός από τις υπορουτίνες έχουμε και τις συμβολοσειρές με την διεπαφή με τον χρήστη.

Δ. Λειτουργικότητα

Ολόιδια με του Εργαστηρίου 3.Ο χρήστης δεν μπορεί να ξέρει την διαφορά.

Ε. Βήματα Υλοποίησης του Εργαστηρίου

Για να υλοποιήσετε το πρόγραμμα είναι καλύτερα να το χωρίσετε σε διακριτά τμήματατα οποία να υλοποιηθούν στα εξής βήματα.

1^{ο} Βήμα – Δημιουργία «Σκελετού» του Νέου Προγράμματος, Χωρίς Κώδικα Υπορουτινών ακόμη

Δημιουργήσετε το main και τις τέσσερις υπορουτίνες, όπου το main έχει τέσσερα διαδοχικά jal και κατόπιν κάνει exit, οι δε υπορουτίνες έχουν μόνο jr $ra. Αυτό θα σας επιτρέπει να δουλεύετε και να αποσφαλματώνετε ένα μόνο μέρος του κώδικα. Ακόμη δεν έχουμε καθόλου κώδικα για είσοδο, έξοδο ή επεξεργασία.

2^{ο} Βήμα – Copy/Paste από το Εργαστήριο 3 των Έτοιμων Κωδίκων, μία-μία τις Υπορουτίνες και Αρχικά με ίδιο τρόπο όπως στο Εργαστήριο 3

Εδώ επαναλαμβάνουμε τον κώδικα όπως τον είχαμε πριν, αλλά μέσα από τις καινούργιες υπορουτίνες. Συνεχίζουμε με τα LB/SB και δεν κάνουμε καμμία απολύτως αλλαγή. Επιβεβαιώνουμε ότι έχουμε την ίδια λειτουργικότητα με το Εργαστήριο 3, όμως μέσα από υπορουτίνες αυτή τη φορά. Κρατάμε αντίγραφο του κώδικα για να μπορούμε να επιστρέψουμε σε αυτό αν στραβώσουν τα πράγματα (γενικά αυτό είναι καλή πρακτική, οπότε δεν θα το αναφέρουμε συνέχεια). Ακόμη δεν περνάμε

ορίσματα/αποτελέσματα μέσω των καταχωρητών $a0-$a3, $v0-$v1.

3ο Βήμα – Αλλαγή Χρήσης Καταχωρητών

Γράψετε κατ' αρχήν στο χαρτί ποια είναι η δική σας χρήση καταχωρητών στο πρόγραμμα, και βάλετέ το και στα σχόλια του προγράμματος (στα Αγγλικά – όχι Ελληνικά, όχι Greeklish). Κατόπιν κάνετε βήμα-βήμα τις αλλαγές στον κώδικα ώστε οι υπορουτίνες να καλούνται με τα σωστά $a0-$a3, $v0-$v1 για ορίσματα. Ακόμη δεν έχουμε βγάλει τα LB/SB, αλλά πλέον έχουμε λειτουργικό κώδικα όπου τα επόμενα βήματα είναι να αλλάξουμε τις υπορουτίνες, μία-μία τη φορά.

4ο Βήμα – Επεξεργασία

Εδώ πάμε σε μία-μία τις υπορουτίνες για να κάνουμε επεξεργασία σε λέξεις αντί για bytes. Είναι σημαντικό να δουλεύουμε μόνο με μία υπορουτίνα την φορά και να την αποσφαλματώνουμε πριν πάμε στην επόμενη. Κατά κύριο λόγο η δουλειά γίνεται στηνget_stringκαι στην process_string. Η λειτουργικότητα πρέπει να διατηρείται.

Ανάλογα με τα κέφια μπορείτε να κάνετε και το BONUSαφού ολοκληρώσετε όλα τα άλλα βήματα.

Σημείωση: ο κώδικας που θα παραδοθεί θα είναι ένας και όχι διαφορετικές εκδόσεις για κάθε βήμα. Αν κάποια ομάδα δεν καταφέρει να τελειώσει τον κώδικα θα πρέπει να αναφέρει μέχρι πιο βήμα υλοποίησης έχει φτάσει.

ΣΤ. Παραδοτέα – Βαθμολογία

Ένα .zip αρχείο που περιλαμβάνει:

Α) ένα .pdf αρχείο με screenshots αποτελέσματος των δυνατών περιπτώσεων που δείχνει την καλή λειτουργία του συστήματος
Β) το αρχείο .asm με τον πηγαίο κώδικα της κανονικής άσκησης, με πάρα πολύ καλά σχόλια (και block comments και line comments).
Γ)το αρχείο .asm με τον πηγαίο κώδικα της bonus άσκησης, με πάρα πολύ καλά σχόλια (και block comments και line comments)όπου στον τίτλο του αρχείου θα έχει προστεθεί και η λέξη bonus, για όσους επιλέξουν να παραδώσουν το bonus.

Σημειώσεις :

1. Πρέπει να έχετε εξαιρετικά καλά σχόλια στον κώδικα στα Αγγλικά (όχι Ελληνικά, όχι Greeklish), σε κάθε μέρος του προγράμματος να έχετε σχόλια στην αρχή για το ποιους καταχωρητές χρησιμοποιείτε και πως, να κάνετε σωστή χρήση καταχωρητών, κλπ. (20% του βαθμού του εργαστηρίου).
2. Η τήρηση των συμβάσεων είναι αναγκαία συνθήκη για ένα σωστό πρόγραμμα

assembly. Συνεπώς η μη τήρηση των συμβάσεων έχει βαθμολογικό αντίκτυποέως 30% του βαθμού

3. Η διαπίστωση αντιγραφής σε οποιοδήποτε σκέλος της άσκησης οδηγεί στην απόρριψη από το σύνολο των εργαστηριακών ασκήσεων. Αυτό γίνεται οποιαδήποτε στιγμή στη διάρκεια του εξαμήνου.

ΚΑΛΗ ΕΠΙΤΥΧΙΑ