Μικροεπεξεργαστές και περιφερειακά

Εαρινό Εξάμηνο 2022

1η Εργασία/Εργαστήριο

Λαδιάς Νικόλαος, Χρήστος Τριφηνόπουλος

A	10	J	2	S	26
B	42	K	36	T	54
C	12	L	3	U	75
D	21	M	19	V	15
E	7	N	1	W	6
F	5	O	14	X	59
G	67	P	51	Y	13
H	48	Q	71	Z	25
I	69	R	8		

Δεδομένο hash table

Η συνάρτηση "hash" σε ARM assembly

Η συνάρτηση που υλοποιήθηκε σε assembly δέχεται δύο ορίσματα: Την αλφαριθμητική ακολουθία την οποία θέλουμε να μετατρέψουμε και μια λίστα στην οποία έχουν αποθηκευτεί σε σειρά τα hashes των γραμμάτων, όπως αυτά δόθηκαν στην εκφώνηση. Επιστρέφει (μέσω του καταχωρητη r0) τον συνολικό hash της ακολουθίας. Η λογική που ακολουθήθηκε για να εξεταστούν οι τρεις περιπτώσεις (αριθμός/ κεφαλαίο γράμμα/ τίποτα από τα προηγούμενα) είναι η εξής:

a)Ελέγχεται αν το ASCII value του χαρακτήρα είναι στο range που αντιστοιχεί στο διάστημα (1,9).
b)Αν είναι εκτός του range προς τα κάτω γίνεται μεταπήδηση στο label "skip", δηλαδή ο χαρακτήρας δεν επηρεάζει κάπως το hash του αλφαριθμητικού.
c)Αν είναι εντός του range αφαιρείται από το αλφαριθμητικό ο χαρακτήρας ASCII που αντιστοιχεί στο μηδέν και το αποτέλεσμα αφαιρείται από το συνολικό hash. Στην συνέχεια εκτελείται το label "skip" για να μην εκτελεστεί και ο κώδικας του label "letter".
d)Αν βρίσκεται εκτός του range προς τα πάνω, γίνεται μεταπήδηση στο label "letter" όπου γίνεται έλεγχος για
ύπαρξη κεφαλαίου λατινικού γράμματος. Έπειτα εκτελείται έτσι κι αλλιώς ο κώδικας του label "skip" για να ελεγχθεί ο επόμενος χαρακτήρας.
e)Ελέγχεται αν ο ASCII του χαρακτήρα βρίσκεται στο διάστημα που αντιστοιχεί στα (A,Z) και αν δεν βρίσκεται γίνεται μεταπήδηση στο label "skip".

f)Αν δεν βρίσκεται εκτός των ορίων εκτελούνται κανονικά οι παρακάτω εντολές οι οποίες βρίσκουν το hash που αντιστοιχεί στο εκάστοτε κεφαλαίο γράμμα και το προσθέτουν στο συνολικό hash.

Για να υλοποιηθεί το βήμα **f** σωστά αφαιρείται ο χαρακτήρας ASCII που αντιστοιχεί στο A από το ASCII του αλφαριθμητικού, έτσι λαμβάνεται απόσταση του γράμματος από το A. Καθώς οι χαρακτήρες τύπου int στην C πιάνουν χώρο 4 byte στην μνήμη, το hash ενός γράμματος που απέχει **x** από το A θα βρίσκεται **4*x** θέσεις μετά την αρχή του hash table στην μνήμη. Η δείκτης στην αρχή του πίνακα αυτού υπάρχει αποθηκευμένος στον καταχωρητή **r5** οπότε εάν προστεθεί **4*x** σε αυτόν, δείχνει πλέον στο hash του αντίστοιχου γράμματος.

Debugging και testing

Κατά την προσπάθεια εκτέλεσης και debugging του προγράμματος, αντιμετωπίστηκαν διάφορα προβλήματα, η λεπτομερής ανάλυση της εκτέλεσης όμως με την βοήθεια της βηματικής σάρωσης του κώδικα καθώς και η εικόνα της μνήμης έκαναν την διαδικασία debugging πιο έυκολη. Με χρήση της δυνατότητας που παρέχει το Keil να εξετάζονται σε πραγματικό χρόνο οι τιμές σε κάθε καταχωρητή αλλά και κάθε θέση της μνήμης, πραγματοποιήθηκε αναλυτικό white-box testing της λειτουργίας του κώδικα, και εντοπίστηκαν σφάλματα άμεσα, στο χαμηλότερο επίπεδο.

Μια πρώτη διαπίστωση ήταν πως στην μνήμη οι τιμές των integers αποθηκεύονται ανά 4 θέσεις(όπως αναφέρθηκε παραπάνω), κάτι που εξαρχής δεν λήφθηκε υπόψη. Για να μπορέσει η ομάδα να το διαπιστώσει αυτό χρειάστηκε να εξεταστεί η μνήμη εσωτερικά , με την βοήθεια του memory κουμπιού του περιβάλλοντος του keil κατά την διάρκεια προφανώς ενός debug session. Ύστερα έγινε αντιληπτό πως προφανώς αυτό εξαρτάται και απο την δομή δεδομένων που επιλέγεται, όπως αναφέρθηκε σε int αυτή είναι 4 bytes, παρόλα αυτά πιθανή είναι και η υλοποίηση σε char type με αποτέλεσμα να πιάνει ένα byte στην μνήμη. Επιπλέον, στην προσπάθεια να παραμείνει όσο πιο απλός και σύντομος ο κώδικας, έγινε αρχικά μια προσπάθεια να γίνουν οι περισσότερες πράξεις με τους καταχωρητές r0 και r1 (πχ εύρεση index πίνακα) . Γρήγορα έγινε φανερό πως αυτή δεν ήταν καλή τακτική και χρησιμοποιήθηκαν επιπρόσθετοι καταχωρητές. Για παράδειγμα ο καταχωρητής r5 χρησιμοποιείται ως δείκτης στο hash table για να μην αλλάζει η τιμή του r1 ο οποίος έχει αποθηκευμένη την διεύθυνση του πρώτου του στοιχείου.

Ο καταχωρητής r2, στον κώδικα χρησιμοποιείται αρχικά για να διαβάσει τον τρέχων χαρακτήρα του string που δίνεται. Με τον οποίο γίνονται και οι αντίστοιχες συγκρίσεις όπως αναφέρθηκαν, ενώ έπειτα μετατρέπεται στον καταχωρητή στον οποίο σώζεται το index του πίνακα των αλφαριθμητικών του string. Για το σκοπό αυτό, αφαιρείται η τιμή ASCII του “A” για να δείχνει ο r2 στην πρώτη θέση του δεδομένου πίνακα hash της εκφώνησης. Όπως είναι γνωστό, οι πίνακες στην C στην πραγματικότητα είναι pointers στο πρώτο τους στοιχείο. Έτσι το hash table που αρχικοποιήθηκε σε κώδικα C εύκολα χρησιμοποιήθηκε από την assembly η οποία δεν ‘καταλαβαίνει’ δομές δεδομένων αλλά μόνο διευθύνσεις.

Για το testing, δοκιμάστηκαν διάφορα strings με διαφορετικούς χαρακτήρες καθώς και νούμερα αλλά και κενά. Το κενό αγνοείται ενώ το τελικό hash που παράγεται είναι σύμφωνο με τους κανόνες της εκφώνησης , δηλαδή προσθέτει hashes κεφαλαίων λατινικών και αφαιρεί τα νούμερα.