Μικροεπεξεργαστές και Περιφερειακά

Εργαστήριο 1

Αλέξιος-Γεώργιος Πάκας

Γεώργιος Ρίζος

**Main:**

Καλούμε με extern τις συναρτήσεις μορφής ARM Assembly, στην αρχή του κώδικα.

Ύστερα δημιουργούμε μια συνάρτηση void που θα διαβάζει εναν προς εναν τον χαρακτήρα που θα στέλνεται στις άλλες συναρτήσεις.

Ακολουθεί η main όπου εκτελει τον βασικό κώδικα σε uart.

Πρόβλημα:

Χρησιμοιούμε sprintf διότι η uart δεν παίρνει int ορισματα και παίρνει char, οπότε αποθηκεύουμε σε ενα char πίνακα το αποτέλεσμα int.

**Hash:**

Ο κώδικας για την hash γράφτηκε με την ακόλουθη λογική:

1. Η ετικέτα .global hash\_calculate δηλώνει την έναρξη της συνάρτησης hash\_calculate.
2. Ακολουθεί η έναρξη της συνάρτησης hash\_calculate.
3. Οι τρεις αρχικές εντολές MOV r3, #0, MOV r4, #0, και MOV r5, =char\_values, MOV r1, r0 , MOV r6, #0 αρχικοποιούν τους καταχωρητές r3, r4 και r6 στην τιμή 0, το r1 στο r0 και r5 στο πινακα τιμών char\_values .
4. Ακολουθεί μια ετικέτα με το όνομα loop, που σηματοδοτεί την έναρξη του βρόχου επεξεργασίας των χαρακτήρων.
5. Μέσα στο βρόχο, ο κώδικας διαβάζει έναν χαρακτηρα από τον πίνακα που περνάει ως όρισμα και ελέγχει τον τύπο του.
6. Αν ο χαρακτήρας είναι γράμμα, αναλογα με το αν είναι κεφαλαίο ή πεζό, πραγματοποιείται μια αντίστοιχη λειτουργία υπολογισμού του hash, ειτε εκτελείτε το lower\_case ειτε συνεχίζει στην ιδια λούπα για το κεφαλαίο.
7. Αν ο χαρακτήρας είναι αριθμός, πραγματοποιείται μια διαφορετική λειτουργία υπολογισμού του hash η numeric.
8. Ο βρόχος συνεχιζεται μέχρι να εξεταστει κάθε χαρακτήρας.
9. Όταν ο χαρακτήρας είναι τελικά μηδέν (CMP r0, #0, BEQ end\_loop), ο βρόχος τερματίζεται.
10. Η τιμή του hash αποθηκεύεται στον καταχωριτή r3 (STR r3, [r2]) και η συνάρτηση επιστρέφει την τιμή αυτή (MOV r0, r3, BX lr).

Και οι δυο λειτουργίες ειτε lower\_case είτε η αρχική λειτουργία/λουπα για τα κεφαλαία, εχουν την ιδια λογική. Αφαιρώ απο το γραμμα που πήρα, απο την συμβολοσείρα, ειτε το a μικρο είτε Α κεφαλαιο αντίστοιχα, και ετσι χρησιμοποιώντας και τον πινακα ascii εχω την "αποσταση" απο το αλφα(μικρο και κεφαλαιο αντιστοιχα). Σύμφωνα με αυτην την αποσταση πηγαίνω στον πινακα char\_values που ειναι αποθηκευμένος στο r5, και αφαιρώ/προσθέτω αντιστοιχα για μικρο και κεφαλαίο γράμμα την τιμή που μου εδωσε ο πινακας.

Πήγαμε στην αρχή να κάνουμε τον κώδικα σύμφωνα με τον πίνακα ASCII , αλλά γρήγορα καταλάβαμε οτι δεν συνάδει με την λογικη του προβλήματος οπότε φτιάξαμε ενα δικό μας πίνακα data, με τις τιμές που δώθηκαν

**Sum:**

Ο κώδικας για την sum γράφτηκε με την ακόλουθη λογική:

1. Έλεγχος εάν ο δοθείς αριθμός είναι μηδέν ή αρνητικός.
2. Αν ο αριθμός είναι μη θετικός, επιστρέφεται μηδέν.
3. Αλλιώς, αρχικοποιείται μια μεταβλητή αποτελέσματος σε μηδέν και μειώνεται ο δοθείς αριθμός κατά ένα.
4. Καλείται αναδρομικά η ίδια συνάρτηση με τον αριθμό που έχει μειωθεί κατά ένα.
5. Το αποτέλεσμα της αναδρομικής κλήσης προστίθεται στον αρχικό αριθμό.
6. Η συνάρτηση επιστρέφει το τελικό αποτέλεσμα.

**Mono\_digit:**

Η συνάρτηση mono\_digit\_calculator λαμβάνει έναν ακέραιο αριθμό ως είσοδο και υπολογίζει το άθροισμα των ψηφίων του αριθμού αυτού. Αρχικά, η συνάρτηση αποθηκεύει τη διεύθυνση επιστροφής του προγράμματος στη στοίβα (PUSH {LR}) για να μπορεί να επιστρέψει στο σημείο κλήσης μετά την ολοκλήρωση. Στη συνέχεια, αντιγράφει την τιμή του hash από τον καταχωρητή R0 στον R1 για να εργαστεί με αυτήν.

Η συνάρτηση χρησιμοποιεί τον καταχωρητή R0 για να αποθηκεύσει το τελικό αποτέλεσμα, δηλαδή το άθροισμα των ψηφίων του αριθμού. Στη συνέχεια, θέτει την τιμή 10 στον καταχωρητή R5, καθώς αυτή η τιμή θα χρησιμοποιηθεί για τις πράξεις διαίρεσης και πολλαπλασιασμού.

Κατά τη διάρκεια κάθε επανάληψης, η συνάρτηση διαιρεί την τρέχουσα τιμή του hash με το 10 (αποθηκευμένο στον R5) και αποθηκεύει το αποτέλεσμα της διαίρεσης στον R3. Στη συνέχεια, πολλαπλασιάζει τον πηλίκο με το 10 και αφαιρεί το αποτέλεσμα από την αρχική τιμή του hash για να πάρει το υπόλοιπο. Το υπόλοιπο προστίθεται στον καταχωρητή R2, ο οποίος αποθηκεύει το συνολικό άθροισμα των ψηφίων. Η διαδικασία αυτή επαναλαμβάνεται μέχρι να γίνει μηδέν ο καταχωρητής R3 και έτσι καταλαβαίνουμε ότι πλέον ο αριθμός μας είναι μονοψήφιος. Όταν αυτό συμβεί, ο καταχωρητής R1 ενημερώνεται με τη νέα τιμή του καταχωρητή R3 και ελέγχεται η τιμή του καταχωρητή R3. Εάν η νέα τιμή του R1 είναι μεγαλύτερη από μηδέν, η συνάρτηση συνεχίζει τη διαδικασία μείωσης και αθροίσματος Επίσης, ελέγχουμε εαν το R2 (που το μεταφέρω στο R1 ) είναι μεγαλύτερο του 9 (δηλαδή διψήφιος αριθμός) που είναι η πρόσθεση των μονοψηφίων, έτσι ωστενα ξανακάνει τη διαδικασία μέχρι να γίνει μονοψήφιος. Αλλιώς, το αποτέλεσμα έχει βρεθεί και επιστρέφεται στον καλούντα.

Πρόβλημα:

Εάν ο αριθμός της hash είναι αρνητικός δυστυχώς η συνάρτηση δεν λειτουργεί.