Ξάνθη, Φεβρουάριος 2024

μΙΚΡΟΕΠΕΞΕΡΓΑΣΤΕΣ ΚΑΙ ΕΦΑΡΜΟΓΕΣ

ΕΞΑΜΗΝΙΑΙΑ ΕΡΓΑΣΙΑ 23  
Του Φοιτητή: Θοδωρή Σαπουνακη

(αεμ: 58406)

Θέμα

Υλοποίηση ενός συστήματος έκδοσης και επικύρωσης εισιτηρίων για Μέσα Μαζικής Μεταφοράς χρησιμοποιώντας ARM Assembly.

Περιεχόμενα

[2 ΕΙΣΑΓΩΓΗ 2](#_Toc159514475)

[3 ΕΚΦΩΝΗΣΗ ΚΑΙ ΑΠΑΙΤΗΣΕΙΣ 3](#_Toc159514476)

[4 ΥΛΟΠΟΙΗΣΗ 4](#_Toc159514477)

[4.1 ΠΛΗΚΤΡΟΛΟΓΙΑ 4Χ4 4](#_Toc159514478)

[4.1.1 ΚΩΔΙΚΑΣ ΠΛΗΚΤΡΟΛΟΓΙΩΝ 6](#_Toc159514479)

[4.2 ΣΥΝΑΡΤΗΣΗ MAIN 9](#_Toc159514480)

[4.3 ΣΥΝΑΡΤΗΣΗ ΕΚΔΟΣΗΣ 10](#_Toc159514481)

[4.4 ΣΥΝΑΡΤΗΣΗ ΕΠΙΚΥΡΩΣΗΣ 13](#_Toc159514482)

[5 ΕΠΙΛΟΓΟΣ 15](#_Toc159514483)

# ΕΙΣΑΓΩΓΗ

Η παρούσα εργασία πραγματοποιήθηκε στα πλαίσια του μαθήματος του 7ου εξαμήνου, Μικροεπεξεργαστές και Εφαρμογές του Διδάσκοντα Γ. Συρακούλη. Η υλοποίηση της έγινε στο περιβάλλον Keil uVision με γνώσεις που αποκτήσαμε στις διαλέξεις αλλά και στα εργαστήρια του μαθήματος. Η εργασία που μου ανατέθηκε είχε θέμα την δημιουργία ενός συστήματος έκδοσης και επικύρωσης εισιτηρίων για Μ.Μ.Μ.. Παρακάτω θα αναλύσω την λύση που κατέληξα και θα παραθέσω φωτογραφίες από τον κώδικα μου και τα σχόλια μου πάνω σε αυτόν.

# ΕΚΦΩΝΗΣΗ ΚΑΙ ΑΠΑΙΤΗΣΕΙΣ

Στην ενότητα αυτήν θα επικεντρωθώ στην εκφώνηση και θα επισημάνω τις απαιτήσεις του προβλήματος αλλά και τις θεωρήσεις που έκανα σε κάποια σημεία τα οποία δεν τα ορίζει η εκφώνηση.

Αρχικά η βασική απαίτηση είναι το σύστημα να κάνει και επικύρωση αλλά και έκδοση εισιτηρίων. Συγκεκριμένα στην έκδοση πρέπει να αποθηκεύεται ο κωδικός του εισιτηρίου, οι διαδρομές (οι οποίες μπορούν να είναι ή 1 ή 10), καθώς και ο χαρακτηρισμός του όσον αφορά τα μέσα συγκοινωνίας σε διαδοχικές θέσεις μνήμης. Στην επικύρωση θα ελέγχεται ο κωδικός του εισιτηρίου και εάν είναι έγκυρο θα αφαιρείται μια διαδρομή. Εάν οι διαδρομές γίνουν 0, τότε το εισιτήριο θα πρέπει να διαγράφεται από την μνήμη.

Η είσοδος των στοιχείων από τον χρήστη θα γίνεται μέσω 2 πληκτρολογίων 4x4 των οποίων τα πλήκτρα αντιστοιχούν σε αριθμού από το 0 έως το F. Τα στοιχεία εισόδου θα είναι το κουμπί που πατιέται σε κάθε πληκτρολόγιο, το οποίο θα μεταφράζεται σε χαρακτηρισμό εισιτηρίου και διαδρομές αντίστοιχα. Στην επόμενη ενότητα θα εξηγήσω αναλυτικά το πως σχεδίασα τα πληκτρολόγια.

Επίσης άλλοι είσοδοι που θα χρειαστώ εκτός των πληκτρολογίων είναι ο κωδικός εισιτηρίου για την επικύρωση και τα χρήματα, τα οποία πρέπει να είναι ≥ της τιμής του εισιτηρίου, για την έκδοση.

# ΥΛΟΠΟΙΗΣΗ

Έχοντας αναλύσει τις απαιτήσεις του προβλήματος μπορώ να αρχίσω να περιγράφω και την υλοποίηση την οποία έκανα.

Καταρχάς είναι φανερό ότι χρειάζομαι 2 συναρτήσεις, μια για έκδοση και μια για επικύρωση εισιτηρίου. Προτού αρχίσω να αναλύω την έκδοση πρέπει πρώτα να εστιάσω στο κομμάτι των πληκτρολογίων τα οποία θα χρησιμοποιηθούν από τον χρήστη για την έκδοση ενός εισιτηρίου.

## ΠΛΗΚΤΡΟΛΟΓΙΑ 4Χ4

Στην προηγούμενη ενότητα εξηγήθηκαν οι θεωρήσεις που έκανα για τα 2 πληκτρολόγια. Συνεπώς καταλαβαίνουμε ότι ανάλογα με το κουμπί που πατήσει ο χρήστης αλλάζουν και 2 μεταβλητές, μια για την στήλη και μια για την σειρά, οι οποίες ακολουθούν onehot αναπαράσταση. Δηλαδή όπως λέει και η εκφώνηση εάν πατηθεί το πλήκτρο 5, το οποίο βρίσκεται στην 2η στήλη και στην 2η σειρά, τότε οι μεταβλητές θα πάρουν τις τιμές 0010 και 0010. Αν για παράδειγμα πατηθεί το πλήκτρο Β(11) που βρίσκεται στην 3η σειρά και στην 4η στήλη τότε οι μεταβλητές θα πάρουν την τιμή 0100 και 1000 αντίστοιχα. ‘Αρα για κάθε πληκτρολόγιο θα χρειαστώ 2 εισόδους για να προσδιορίσω το πλήκτρο που πατήθηκε από τον χρήστη.

Στο πρώτο πληκτρολόγιο θα επιλέγει ο χρήστης το είδος εισιτηρίου που θέλει. Οι επιλογές που θα έχει θα είναι οι εξής:

* Λεωφορείο
* Τραμ
* Μετρό
* Τραίνο
* Συνδυασμοί 2 από τα παραπάνω
* Συνδυασμοί 3 από τα παραπάνω
* Όλα τα παραπάνω

Όσων αφορά το κόστος για λόγους ευκολίας όρισα τα εισιτήρια για Λεωφορείο, Μετρό, Τραμ, Τραίνο να κοστίζουν 1€, οι διπλοί συνδυασμοί 2€, οι τριπλοί 3€ και το εισιτήριο που τα περιέχει όλα 4€. Έτσι έχω καλύψει 15 από τα 16 πλήκτρα.

|  |  |  |  |
| --- | --- | --- | --- |
| **Πληκτρολόγιο 1** | | | |
| **Λεωφορείο(1€)** | **Τραίνο(1€)** | **Μετρό(1€)** | **Τραμ(1€)** |
| **Λεωφορείο και Τραίνο(2€)** | **Λεωφορείο και Μετρό(2€)** | **Λεωφορείο και Τραμ(2€)** | **Τραίνο και Μετρό(2€)** |
| **Τραίνο και Τραμ(2€)** | **Μετρό και Τραμ(2€)** | **Λεωφορείο Τραίνο και Μετρό(3€)** | **Τραίνο Μετρό και Τραμ(3€)** |
| **Μετρό Τραμ και Λεωφορείο(3€)** | **Τραμ Λεωφορείο και Τραίνο(3€)** | **Όλα τα ΜΜΜ(4€)** | **--** |

Στο δεύτερο πληκτρολόγιο θα επιλέγει ο χρήστης τις διαδρομές που θέλει. Στο συγκεκριμένο πρόβλημα η εκφώνηση ορίζει να υπάρχει η επιλογή ή 1 ή 10 διαδρομών με αποτέλεσμα να καλύπτονται δύο πλήκτρα. Σε ένα ρεαλιστικό σενάριο θα μπορούσε το δεύτερο πληκτρολόγιο να έχει παραπάνω επιλογές διαδρομών όπως 5, 15, 20 κοκ.

|  |  |  |  |
| --- | --- | --- | --- |
| **Πληκτρολόγιο 2** | | | |
| **1 Διαδρομή** | **10 Διαδρομές** | **--** | **--** |
| **--** | **--** | **--** | **--** |
| **--** | **--** | **--** | **--** |
| **--** | **--** | **--** | **--** |

Συνεπώς αφού το πληκτρολόγιο 2 έχει μία μόνο σειρά, θα χρειαστώ μια είσοδο αντί για δύο που έχει το πληκτρολόγιο 1.

### ΚΩΔΙΚΑΣ ΠΛΗΚΤΡΟΛΟΓΙΩΝ

Παρακάτω παραθέτω κάποια κομμάτια από τον κώδικα του πληκτρολογίου 1. Θα εξηγήσω στην επόμενη ενότητα την συνάρτηση main και τις εισόδους που όρισα. Προς το παρόν αυτό που έχει σημασία είναι ότι στον R0 περνάει η στήλη του πληκτρολογίου 1 και στον R1 η σειρά του, ενώ στον R2 η στήλη του πληκτρολογίου 2. Όλα αυτά σε onehot αναπαράσταση. Για παράδειγμα η 3ή στήλη αντιστοιχεί στον αριθμό 01002 (4)10.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Αρχικά κάνω compare τον R0 με το 1 ,2 ,4 ,8 για να κάνω branch στην στήλη που πρέπει. Αν ο αριθμός που έχω εισάγει στον R0 δεν είναι 1,2,4,8 τότε μετακινώ το 255 στον R6 που σημαίνει ότι έχω error. Αφού μετακινήθηκα στην στήλη που πρέπει, κάνω κάτι ανάλογο και με την σειρά. Αν πετύχω τον αριθμό της σειράς τότε θα βάλω στον R6 τον χαρακτηρισμό εισιτηρίου (0-F όπως ορίστηκε στον πίνακα πάνω) και στον R4 την τιμή του εισιτηρίου ανάλογα με το είδος του. Έπειτα το πρόγραμμα κάνει BX LR στην συνάρτηση της έκδοσης.

Ομοίως έχω προγραμματίσει και το πληκτρολόγιο 2.

A screen shot of a computer code

Description automatically generated

Το πληκτρολόγιο έχει μια σειρά οπότε το μόνο που έχω να κάνω είναι να διαχωρίσω την στήλη. Αυτό το πετυχαίνω με το CMP και έπειτα branch ανάλογα με το αποτέλεσμα. Αν ο αριθμός στον R2 είναι 1 τότε περνάει μια διαδρομή ενώ αν είναι 2 τότε περνάνε δέκα και έπειτα κάνω BX LR στην συνάρτηση της έκδοσης.

## ΣΥΝΑΡΤΗΣΗ MAIN

Προτού αναλύσω την συνάρτηση της έκδοσης πρέπει να εξηγήσω την συνάρτηση Main.

A screenshot of a computer program

Description automatically generated

Στην συνάρτηση main ορίζω τις 3 σταθερές για τα πληκτρολόγια, 1 για τα χρήματα και 1 για έναν πίνακα του οποίου χρειάζομαι την διεύθυνσή και τις θέσεις του για να αποθηκεύσω αργότερα αυτά που ζητούνται στην εκφώνηση. Επίσης ορίζω μια μεταβλητή kodikos για την συνάρτηση της επικύρωσης. Έπειτα ορίζω κάποιες τυχαίες τιμές στις εισόδους μου και στο συγκεκριμένο παράδειγμα, καλώ την συνάρτηση επικύρωσης μετά από τρεις κλήσεις της συνάρτησης έκδοσης. Για κάθε κλήση συνάρτησης αλλάζω και τις εισόδους. Το αποτέλεσμα που θέλω να έχω στην μνήμη, έπειτα από 3 εκδόσεις βάση αυτών των εισόδων είναι:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ΚΩΔ | ΔΙΑΔ | ΧΑΡΑΚ | ΚΩΔ | ΔΙΑΔ | ΧΑΡΑΚ | ΚΩΔ | ΔΙΑΔ | ΧΑΡΑΚ |
| 0 | A | 3 | 1 | 1 | 5 | 2 | A | 8 |

Αφού γίνουν οι εκδόσεις ακολουθεί μια επικύρωση του εισιτηρίου με κωδικό 2. Άρα οι τιμές στην μνήμη πρέπει να καταλήξουν κάπως έτσι:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ΚΩΔ | ΔΙΑΔ | ΧΑΡΑΚ | ΚΩΔ | ΔΙΑΔ | ΧΑΡΑΚ | ΚΩΔ | ΔΙΑΔ | ΧΑΡΑΚ |
| 0 | A | 3 | 1 | 1 | 5 | 2 | 9 | C |

## ΣΥΝΑΡΤΗΣΗ ΕΚΔΟΣΗΣ

Η συνάρτηση έκδοσης δέχεται ως εισόδους την stili\_pinaka1, seira\_pinaka1, stili\_pinaka2 στον R0,R1,R2 αντίστοιχα και επίσης τα xrimata στον R3 και την διεύθυνση του array στον R4.

A screenshot of a computer program

Description automatically generated

Ξεκινώντας μετακινώ την διεύθυνση του πίνακα η οποία βρίσκεται στον R4, στον R5 επειδή θα ξαναχρησιμοποιήσω τον R4 πιο κάτω. Κάνω adds 12 στην διεύθυνση αυτή για δύο λόγους. Ο πρώτος λόγος είναι ότι η διεύθυνση του πίνακα είναι ίδια με του sp, οπότε έτσι αφήνω 3 θέσεις κενές για το stack. Επίσης θα χρησιμέψει και όταν ξανακαλέσω την συνάρτηση έκδοσης ώστε τα καινούρια δεδομένα να αποθηκευτούν 3 θέσεις μετά την πρώτη αποθήκευση (αφού είπαμε πως θέλουμε για κάθε έκδοση να γίνονται 3 αποθηκεύσεις).

Κρατάω τον LR στο stack ώστε να μπορώ να κάνω exit από την συνάρτηση αργότερα. Καλώ την συνάρτηση του πίνακα 1, τσεκάρω για error και επαναλαμβάνω με αυτήν του πίνακα 2.

Μετά από αυτό θα έχω στον R5 την διεύθυνση που θα αποθηκεύσω, στον R3 τα χρήματα που έβαλε ο χρήστης, στον R4 την τιμή του εισιτηρίου και στον R2 τις διαδρομές που επιλέχθηκαν. Έπειτα θα πολλαπλασιάσω την τιμή του εισιτηρίου με τις διαδρομές που επιλέχθηκαν για να βρω την τελική τιμή. Η θεώρηση που έκανα εδώ για λόγους ευκολίας είναι ότι η μια διαδρομή θα κοστίζει μια φορά το εισιτήριο, ενώ οι δέκα διαδρομές θα κοστίζουν δέκα φορές το εισιτήριο. Εάν τα χρήματα που έβαλε ο χρήστης είναι λιγότερα του κόστους τότε έχω error και κάνω exit το πρόγραμμα.

Στην γραμμή 31 κάνω store τον R7 ο οποίος παρατηρώ από το keil ότι έχει την τιμή 0 οπότε τον θέτω ως τον πρώτο κωδικό εισιτηρίου. Στην επόμενη γραμμή κάνω store τις διαδρομές και στην παρεπόμενη τον χαρακτηρισμό. Έπειτα προσθέτω 1 στον κωδικό(R7) ώστε να τον ανανεώσω για την επόμενη έκδοση. Τέλος μετακινώ την διεύθυνση μνήμης στον R4 για να περάσει και αυτή στον καταχωρητή και να μείνει για την επόμενη έκδοση. Κάνω POP στο program counter τον Link Register που είχα αποθηκεύσει προηγουμένως και έτσι επιστρέφω πίσω στην main συνάρτηση.

A blue line with black text

Description automatically generated

Figure 1 - Kαταχωρητές στην αρχή της πρώτης έκδοσης



Figure 2 - Μνήμη στο τέλος της πρώτης έκδοσης

Έπειτα θα γίνει ξανά έκδοση με διαφορετικές εισόδους.

A screenshot of a computer

Description automatically generated

Figure 3 - Καταχωρητές στην αρχή της δεύτερης έκδοσης



Figure 4 - Μνήμη στο τέλος της δεύτερης έκδοσης

A white background with black text

Description automatically generated

Figure 5 - Kαταχωρητές στην αρχή της τρίτης έκδοσης



Figure 6 - Μνήμη στο τέλος της τρίτης έκδοσης

Όπως βλέπουμε τα αποτελέσματα είναι όπως τα περιμέναμε, άρα οι εκδόσεις εισιτηρίων έγιναν με επιτυχία.

## ΣΥΝΑΡΤΗΣΗ ΕΠΙΚΥΡΩΣΗΣ

Στον κώδικά μου η συνάρτηση επικύρωσης καλείται μετά από 3 εκδόσεις. Οι είσοδοί της είναι δυο, ο κωδικός εισιτηρίου και η διεύθυνση του πίνακα που χρησιμοποίησα και προηγουμένως για να αποθηκεύσω τα στοιχεία των εισιτηρίων. Ο τρόπος που σκέφτηκα να γίνεται η επικύρωση είναι να υπάρχει μια επαναληπτική διαδικασία μέσα στην οποία θα συγκρίνεται ο κωδικός που εισάγει ο χρήστης με κάθε 3η θέση μνήμης, αφού οι κωδικοί αποθηκεύονται κάθε τρείς θέσεις. Έπειτα αν η τιμή εισόδου είναι ίδια με την τιμή που βρίσκω στην μνήμη το εισιτήριο είναι valid και θα του αφαιρείται μια διαδρομή.

A screenshot of a computer

Description automatically generated

Αρχικά μηδενίζω το R2 για να το χρησιμοποιήσω αργότερα. Μετά προσθέτω στην διεύθυνση στην οποία θα ψάξω να βρω τον κωδικό το 12, δηλαδή 3 θέσεις μνήμης. Αυτό το είχα κάνει και στην έκδοση. Φορτώνω στον R3 την τιμή της θέσης μνήμης αυτής, η οποία θα είναι ο πρώτος κωδικός εισιτηρίου δηλαδή το 0. Συγκρίνω την τιμή αυτή με τον κωδικό στο R0 και αν δεν είναι ίδιες επαναλαμβάνω την διαδικασία 8 φορές ή μέχρι να βρεθεί ο κωδικός. Την επανάληψή αυτή την πετυχαίνω φορτώνοντας τις μέγιστες επαναλήψεις που θέλω στον R4 (στο κώδικά μου τις έβαλα 8) και έχοντας μια τιμή που θα αυξάνεται σε κάθε επανάληψή στον R2. Στο παράδειγμα αυτό η διαδικασία γίνεται τρεις φορές αφού τελικά στην τρίτη επανάληψη βρίσκεται ο κωδικός, συνεπώς ο PC μετακινείται στο valid.

Από εκεί και έπειτα βρίσκω τις διαδρομές του valid εισιτηρίου στην επόμενη θέση μνήμης και από αυτές αφαιρώ μια. Συγκρίνω την νέα τιμή με το 0 και αν είναι ίση τότε κάνω branch στην diagrafi δηλαδή στην σειρά 70. Τότε αποθηκεύω το 0 στην θέση μνήμης που βρισκόταν ο κωδικός εισιτηρίου και βγαίνω από την συνάρτηση. Αν τώρα δεν είναι 0 τότε αποθηκεύω την νέα τιμή στην θέση μνήμης που βρίσκονται οι διαδρομές. Τέλος κάνω BX LR για να επιστρέψω στην main.

Στο τέλος η μνήμη καταλήγει έτσι:



Figure 7 - Μνήμη στο τέλος της επικύρωσης

Τα πράσινα κελιά είναι αυτά στα οποία έχει γίνει αναζήτηση για τον κωδικό(πέρα από το προτελευταίο στο οποίο έγινε αποθήκευση νέα τιμή). Επίσης οι διαδρομές του εισιτηρίου με κωδικό 2 έχουν μειωθεί στις 9 άρα το πρόγραμμα δούλεψε όπως πρέπει αφού και έκανε αναζήτηση στα σωστά σημεία αλλά και μείωσε την σωστή τιμή και την αποθήκευσε στην σωστή διεύθυνση.

# ΕΠΙΛΟΓΟΣ

Στην εργασία αυτή χρησιμοποιώντας τις γνώσεις που έχω αποκομίσει από την θεωρία και τα εργαστήρια του μαθήματος, Mικροεπεξεργαστές και Εφαρμογές, υλοποίησα ένα σύνθετο πρόβλημα με ένα μικροεπεξεργαστή αρχιτεκτονικής ARM στο περιβάλλον keil uVision. Έπειτα από κάποιες θεωρήσεις μου σε κάποια σημεία, τα οποία δεν ήταν εμφανή, θεωρώ πως κάλυψα πλήρως το πρόβλημα που μου ζητήθηκε.

Από την δουλειά μου πάνω στο πρόβλημα έμαθα αρκετά για το πώς δουλεύει μια μνήμη σε έναν επεξεργαστή, πώς να χρησιμοποιώ σωστά το Stack και εντολές όπως PUSH και POP, πως δουλεύουν καταχωρητές όπως ο PC και ο LR αλλά και πολλά άλλα χρήσιμα πράγματα πάνω στην γλώσσα ARM assembly.