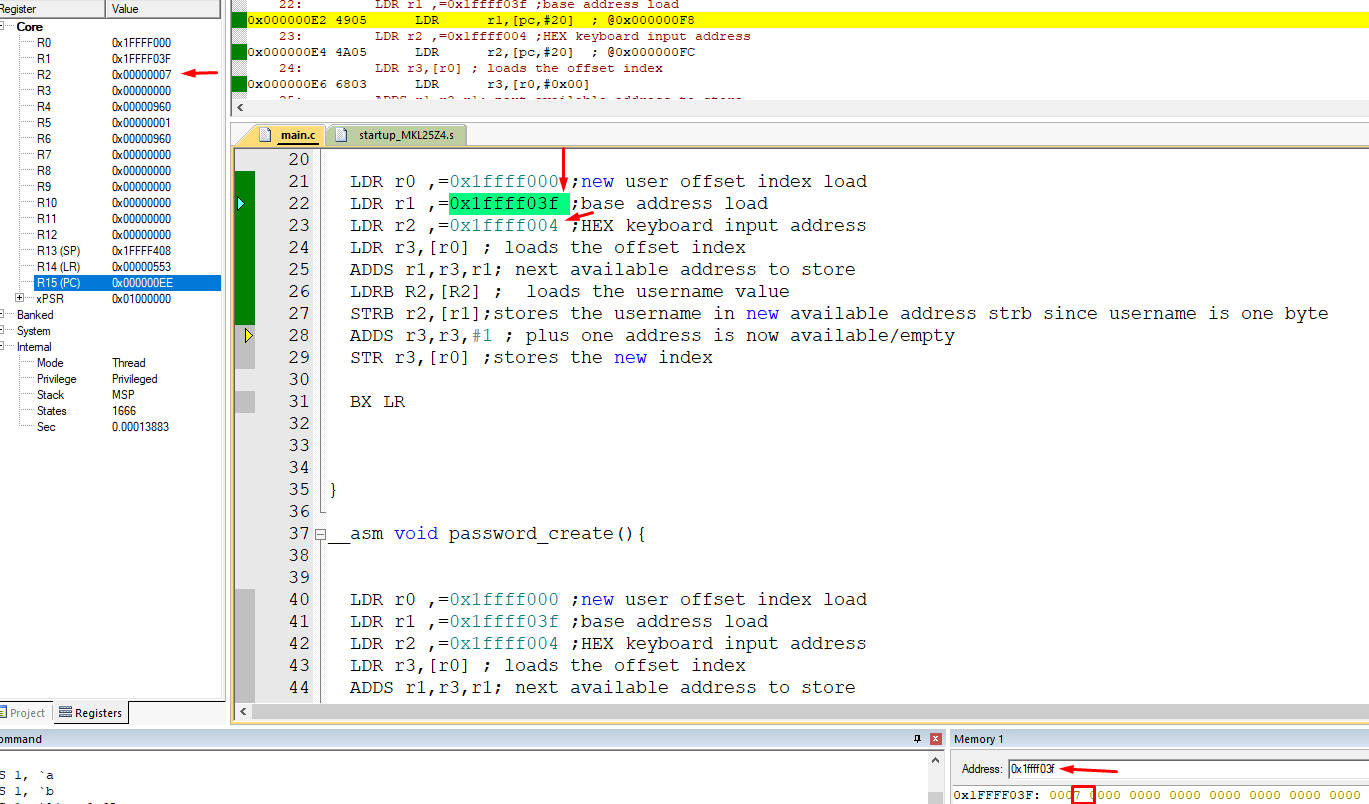
**ΕΡΓΑΣΙΑ ΣΤΟΥΣ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΕΣ ΚΑΙ ΕΦΑΡΜΟΓΕΣ**

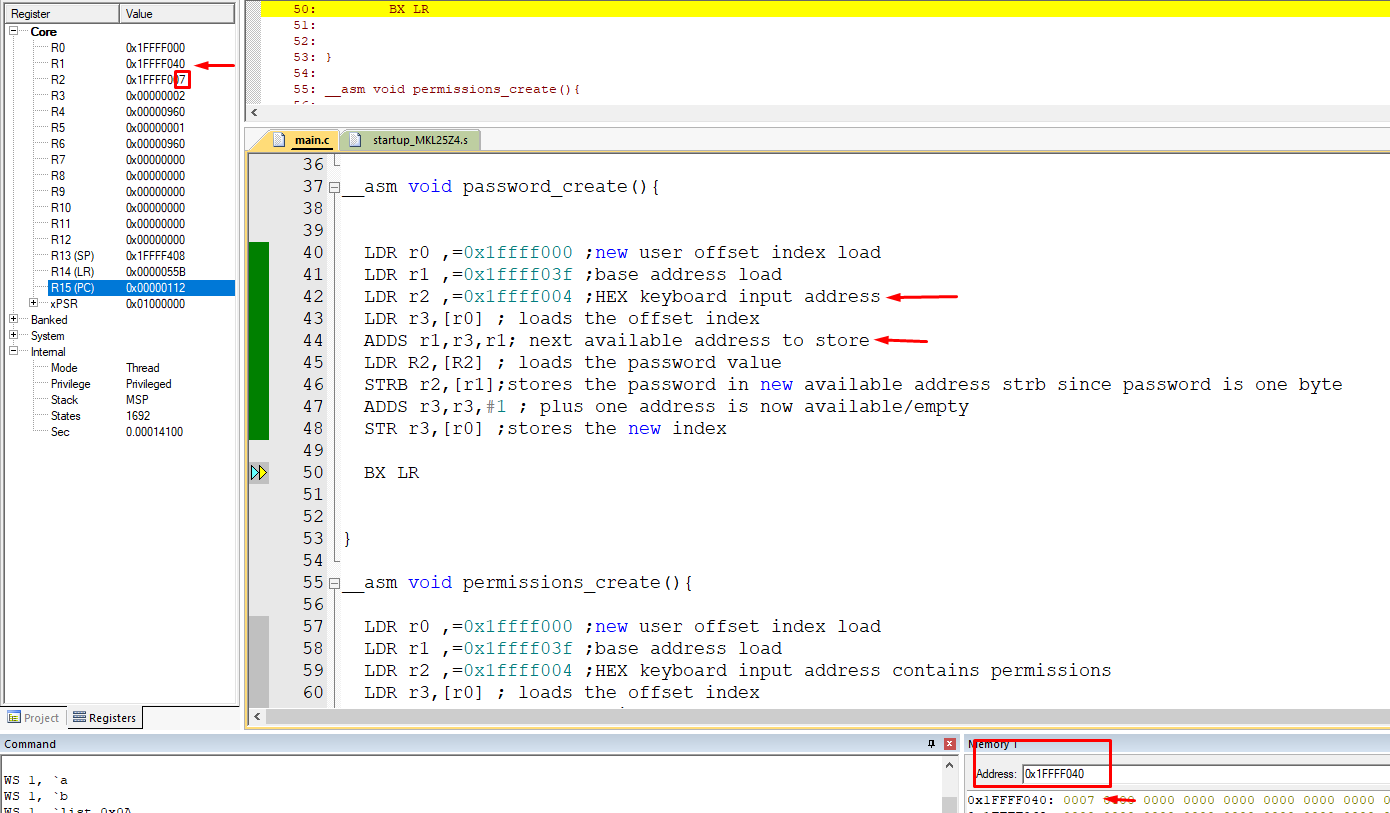
Η εργασία αφορά την υλοποίηση ενός ελεγκτή ο οποίος δέχεται εισόδους από δυο πληκτρολόγια των οποίων οι τιμές αποθηκεύονται σε θέσεις μνήμης της RAM .Αρχικά οι τιμές του πρώτου δεκαεξαδικού πληκτρολογίου αποθηκεύονται στην θέση μνήμης 0x1ffff004 (hex) ενώ οι τιμές του δευτέρου πληκτρολογίου αποθηκεύονται στην θέση μνήμης 0x1ffff008(hex).Οι τιμές των πληκτρολογίων προέρχονται από αισθητήρες και πριν αποθηκευτούν αντιστοιχίζονται σε δυαδική κωδικοποίηση .

Αρχικά στην main καλώ την συνάρτηση **initialize()** με την οποία δημιουργώ τον πρώτο χρήστη ο οποίος πάντα έχει δικαιώματα admin. Ποιο συγκεκριμένα στην συνάρτηση **initialize()** αρχικοποιώ τον **offset index** ο οποίος πάντα θα μου δείχνει στην πρώτη ελεύθερη θέση (την μετατόπιση από την αρχική διεύθυνση 0x1ffff03f ) στην οποία μπορώ να αποθηκεύσω τα στοιχεία του νέου χρήστη . Η αρχικοποίηση γίνεται μέσω της συνάρτησης **index\_init()** . Έπειτα καλώ την συνάρτηση **username\_create()** στην οποία φορτώνω από την διεύθυνση μνήμης 0x1ffff004 δηλαδή από το πρώτο πληκτρολογίου το username . Αυτήν την τιμή την αποθηκεύω στο άθροισμα της διεύθυνσης που μου δείχνει το base address (0x1ffff03f) και ο **offset index.** Μετά καλώ την συνάρτηση **password\_create()** η οποία έχει ακριβώς την ίδια λογική με την **username\_create** αλλά εδώ αποθηκεύω το κωδικό . Και στις δυο περιπτώσεις μετά το τέλος τον διεργασιών αυξάνω τον **offset index** κατά ένα ώστε να δείχνει στην αμέσως επόμενη ελεύθερη θέση μνήμης . Τελικώς ακολουθεί η συνάρτηση **admin\_status\_enable()** η οποία δίνει στο χρήστη admin status και αυξάνει κατά ένα τον δείκτη offset.

Μπορώ να δώσω κάποιες αρχικές τιμές στον admin μέσω της βοηθητικής συνάρτησης **testvalue2()** η οποία προσομοιάζει την είσοδο του πληκτρολογίου . Ουσιαστικά αντικαθιστά στην διεύθυνση που πηγαίνουν οι είσοδοι του πληκτρολογίου μια τιμή που θα επιλέξουμε εμείς χειροκίνητα .

Έστω username και password 0007 C:\Users\Infinity\Desktop\kKGIxPp9QL6t3v-E7W0gGQ.png





Η πρώτη φωτογραφεία αντιστοιχεί στο username ενώ η δεύτερη στο password.

Μετά το καθορισμό των παραπάνω στοιχείων by default ο χρήστης γίνεται admin μέσω της συνάρτησης **admin\_status\_enable().**

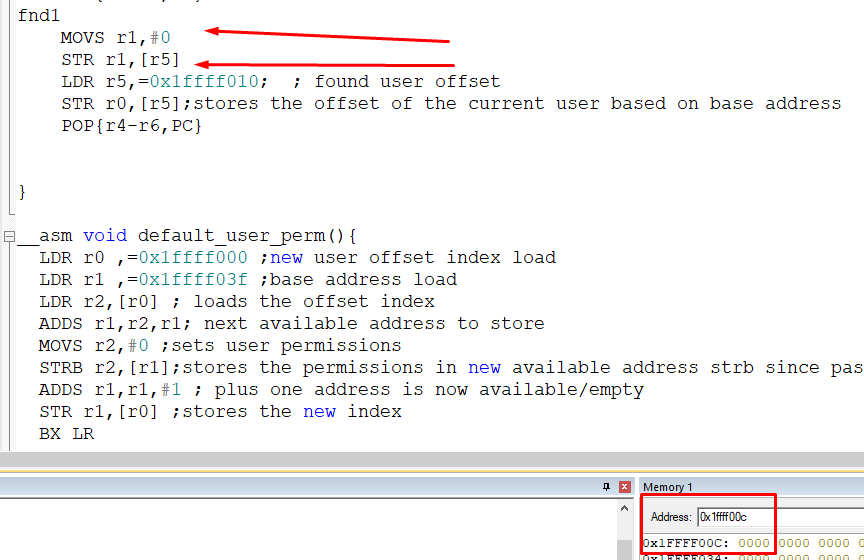
Έπειτα το initialization ο ελεγκτής επιστρέφει σε idle κατάσταση. Για να βγει από εκεί ο χρήστης πρέπει να εισάγει από το δεύτερο πληκτρολόγιο την τιμή 0. Χρησιμοποιώ την συνάρτηση **testvalue1()** ώστε να δοκιμάσω τιμές οι οποίες αντιστοιχούν στο εκάστοτε input του χρήστη . Αυτές οι τιμές εισάγονται στο χώρο διευθύνσεων του δευτέρου πληκτρολογίου. Ο έλεγχος για το αν θα βγει από την κατάσταση idle γίνεται από την συνάρτηση **idle\_state\_exit()** .Αν η τιμή δεν είναι 0 επιστρέφει σε κατάσταση idle μέχρι να του έρθει η κατάλληλη είσοδος (0).

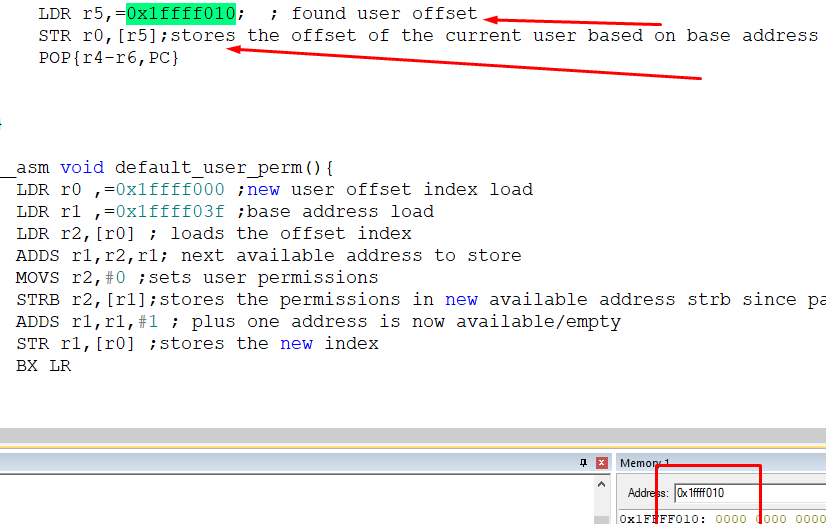
Μετά αφού περάσει την κατάσταση idle θα του ζητηθούν να εισάγει τα στοιχεία του. Πάλι με την **testvalue2()** θα προσομοιώσω την είσοδο του keyboard. Σε περίπτωση που είναι λάθος ξαναγυρνάει σε κατάσταση αναμονής . Έστω πάλι οι είσοδοι 0007.

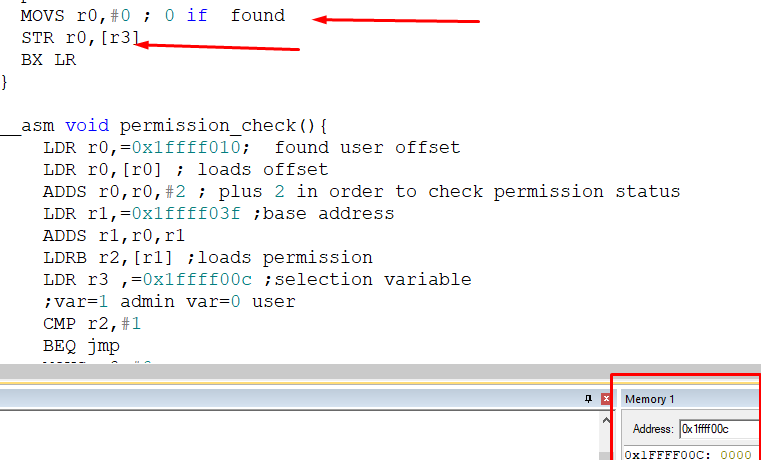
Ανάλογα με το αν είναι σωστή η απάντηση ο δείκτης **selection variable**

Θα πάρει μια ανάλογη true ή false τιμή και θα χρησιμοποιηθεί στην main για το αν θα γυρίσει σε idle ή θα συνεχίσει τις διεργασίες . H διεύθυνση του παραπάνω δείκτη είναι η **0x1ffff00c.**

Στο παρακάτω παράδειγμα του username η τιμή του **selection variable** είναι 0 και συμβολίζει ότι βρέθηκε .Επίσης ο δείκτης **found user offset** μας δείχνει την απόσταση του χρήστη από την base address (το username του πιο συγκεκριμένα , εδώ είναι 0 το οoffset αφού είναι ο πρώτος user(admin).

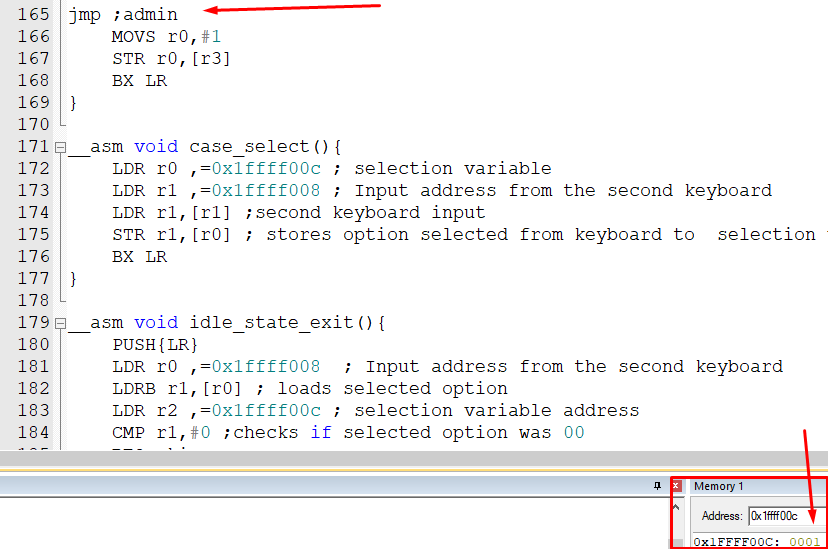






Όταν ελέγχω για το κωδικό αφού γνωρίζω το offset του χρήστη προσθέτω +1 και ελέγχω το κωδικό που έβαλα και το κωδικό που έχω στην μνήμη. Κάνω +1 καθώς ο δείκτης founder user offset μου δείχνει την θέση του ονόματος , ο κωδικός είναι μια θέση(byte) μακριά . Στην περίπτωση που είναι σωστό ο δείκτης selection variable γίνεται 0 όπως και φαίνεται. Τα παραπάνω υλοποιούνται με τις συναρτήσεις **username\_check()** και **password\_check().**

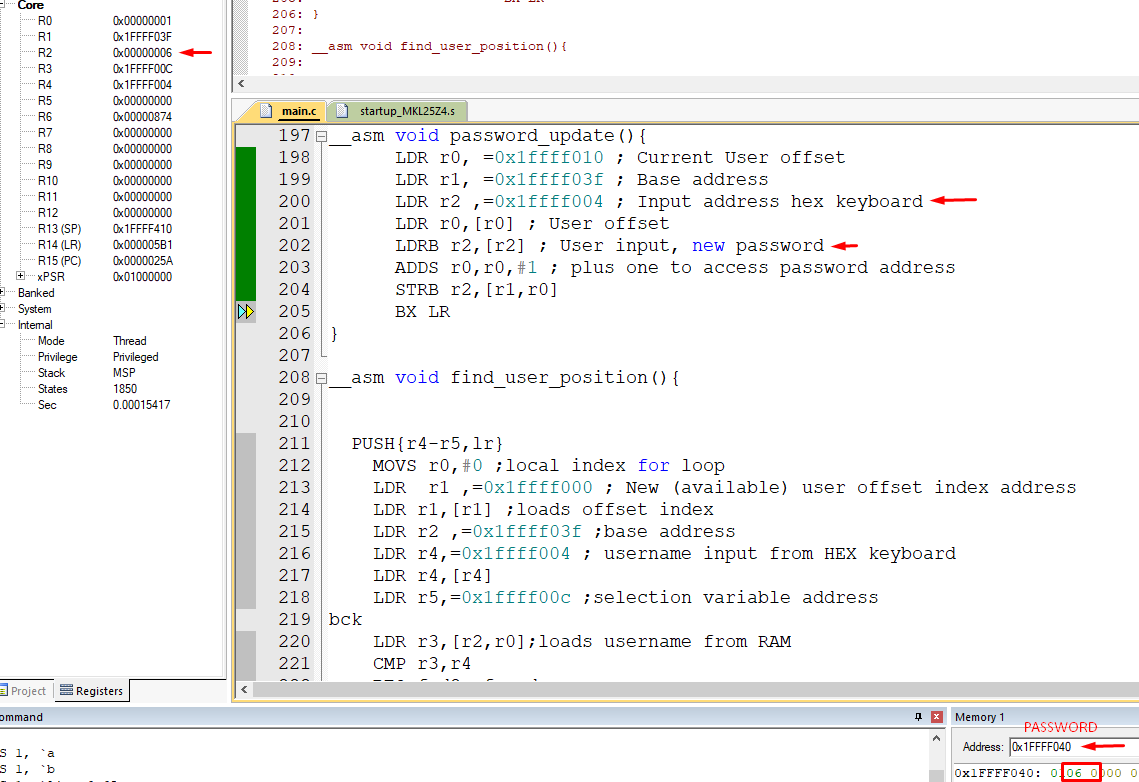
Έπειτα ακολουθεί ο καθορισμός των δικαιωμάτων. Εδώ για να δούμε τα δικαιώματα προσθέτω +2 στο δείκτη **found user offset** . Ανάλογα με τα δικαιώματα ο δείκτης παίρνει τιμές 1 για admin και 0 για user . Τα παραπάνω υλοποιούνται με την συνάρτηση **permission\_check()** .



Μιας και ελέγχουμε το πρώτο χρήστη που είναι by default admin το αποτέλεσμα βγαίνει 1 και αποθηκεύτε στο δείκτη **selection variable**.Ανάλογα με τα δικαιώματα προκύπτουν και οι ανάλογες επιλογές.

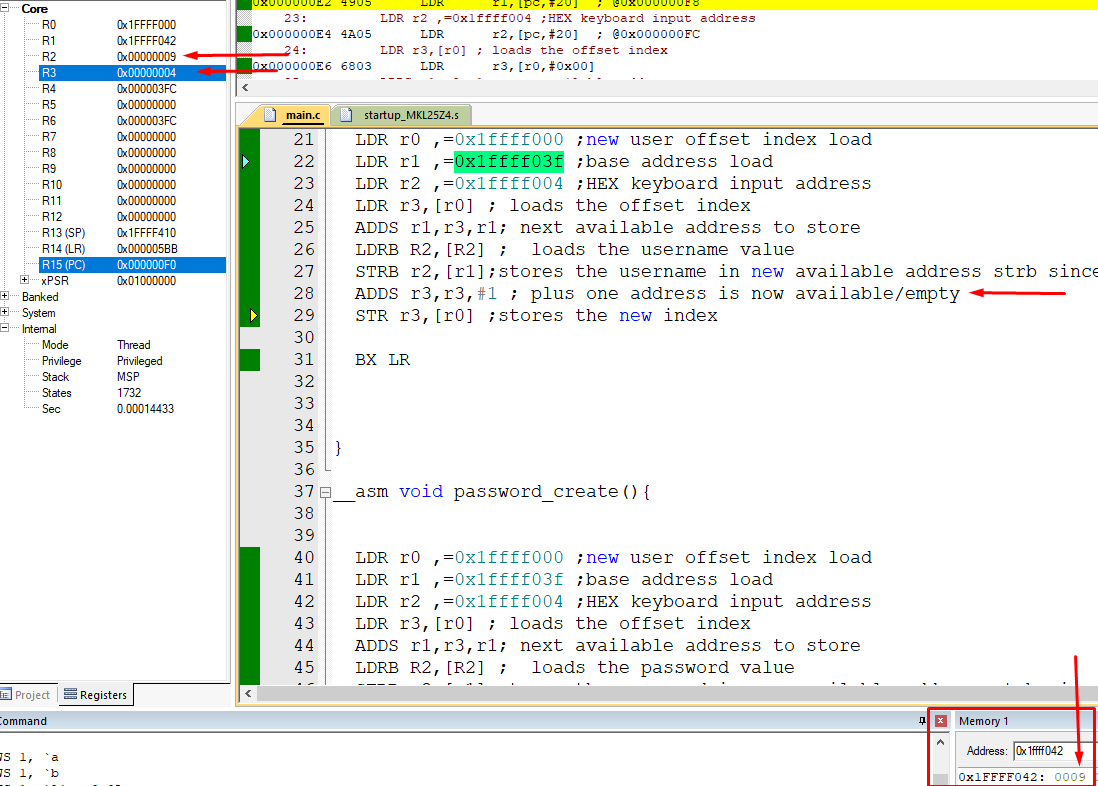
Δοκιμάζουμε την περίπτωση του admin ο οποίος έχει και τις περισσότερες επιλογές . Οι διάφορες δοκιμές θα γίνουν μέσω της **testvalue3()** η οποία προσομοιάζει την είσοδο του δευτέρου πληκτρολογίου όπως και η 1 .

Στην περίπτωση που η είσοδος είναι 1 τότε ο ελεγκτής γυρνάει σε idle. Στην περίπτωση 2 αλλάζω το κωδικό του χρήστη εδώ με την **testvalue4()** προσομοιάζω το input του πρώτου πληκτρολογίου και θέτω νέο password 6



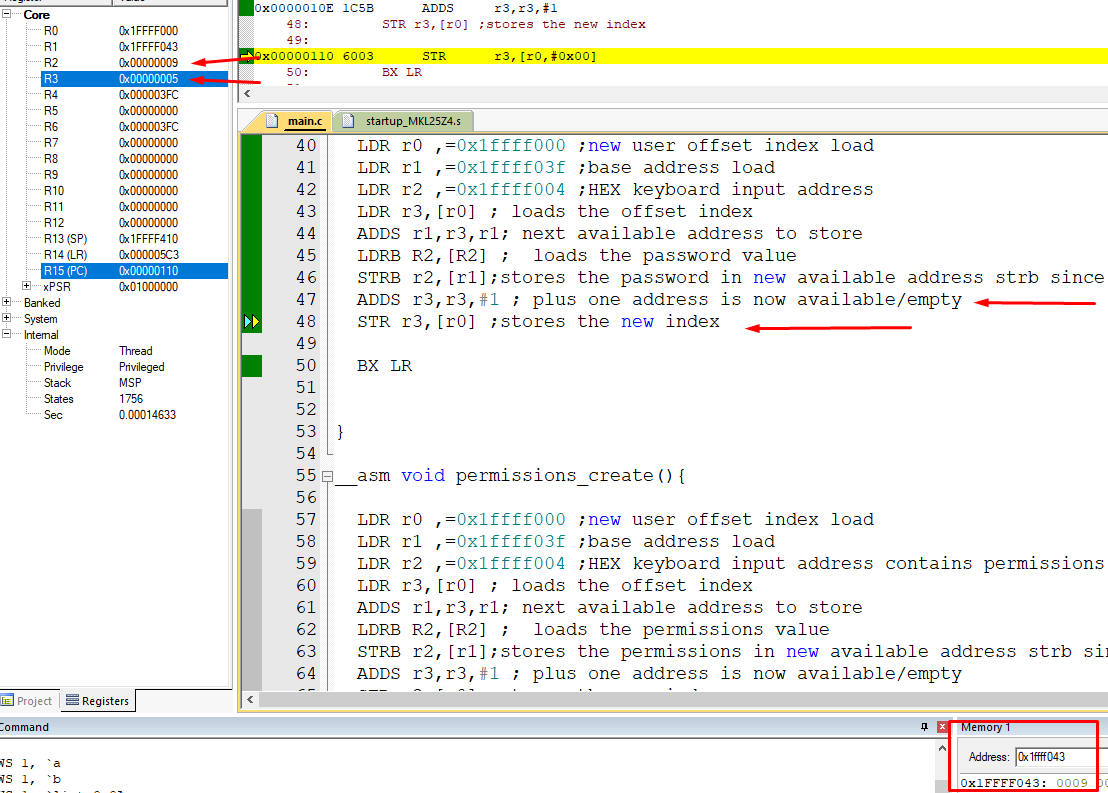
Το παλιό password ήταν 7 και μέσω της συνάρτησης έγινε 6 .Tα παραπάνω έγιναν με την βοήθεια της συνάρτησης **password\_update().**

Έπειτα έχουμε την δημιουργία ενός νέου χρήστη δηλαδή την περίπτωση 3 . Τόσο admin όσο και user μπορούν να δημιουργήσουν νέους χρήστες αλλά ένας χρήστης μπορεί να δημιουργήσει απαραίτητα έναν νέο χρήστη όχι admin. Ο νέος χρήστης έχει username και password 09.

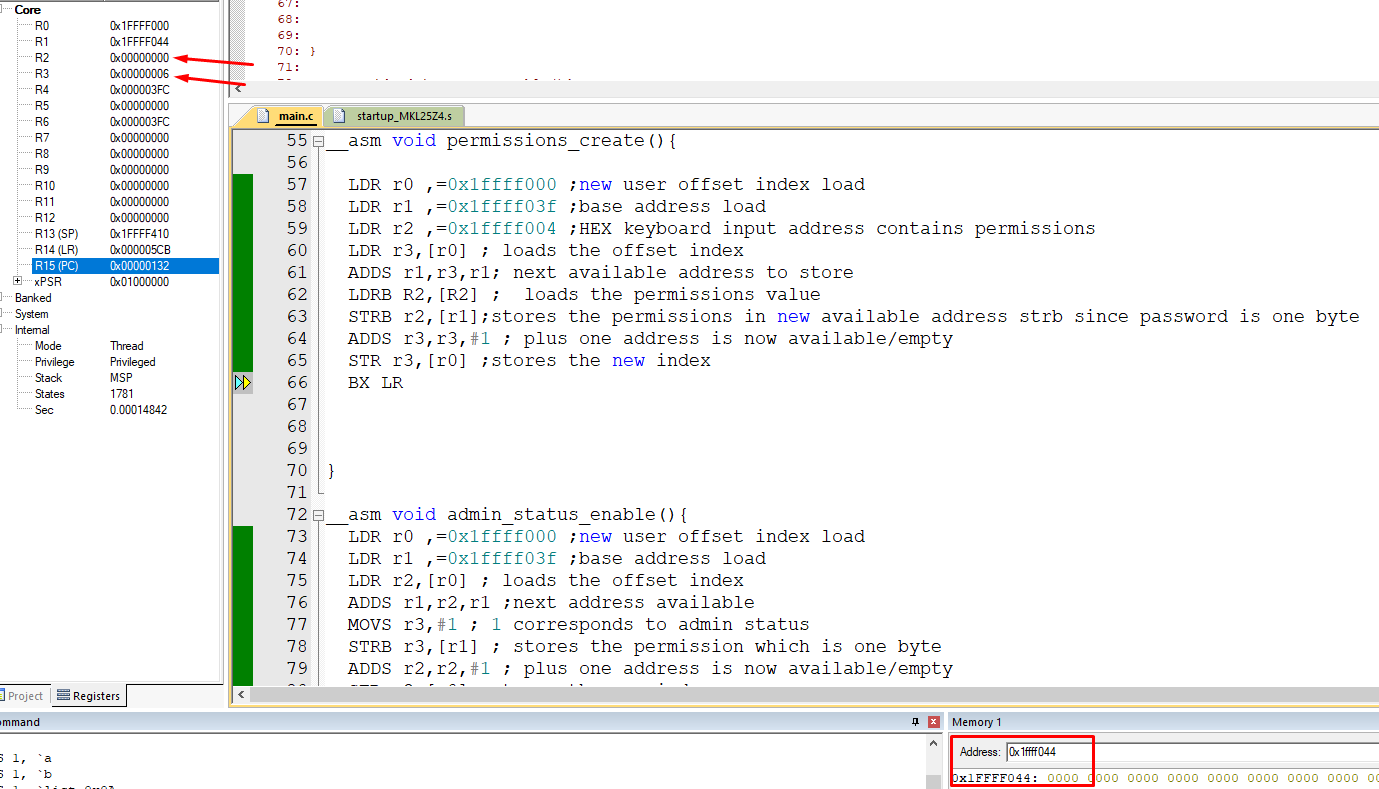


Εδώ βλέπουμε τον καταχωρητή r2 ο οποίος έχει την είσοδο που πήραμε από το πληκτρολόγιο και τον καταχωρητή r3 ο οποίος δείχνει στην νεα διαθέσιμη θέση όπου και θα μπει ο κωδικός. Στην αρχή της συνάρτησης ήταν μια θέση μικρότερη άρα στην 3 . Επίσης βλέπω στην μνήμη ότι αποθηκεύτηκε το username .To παραπάνω έγινε με την συνάρτηση

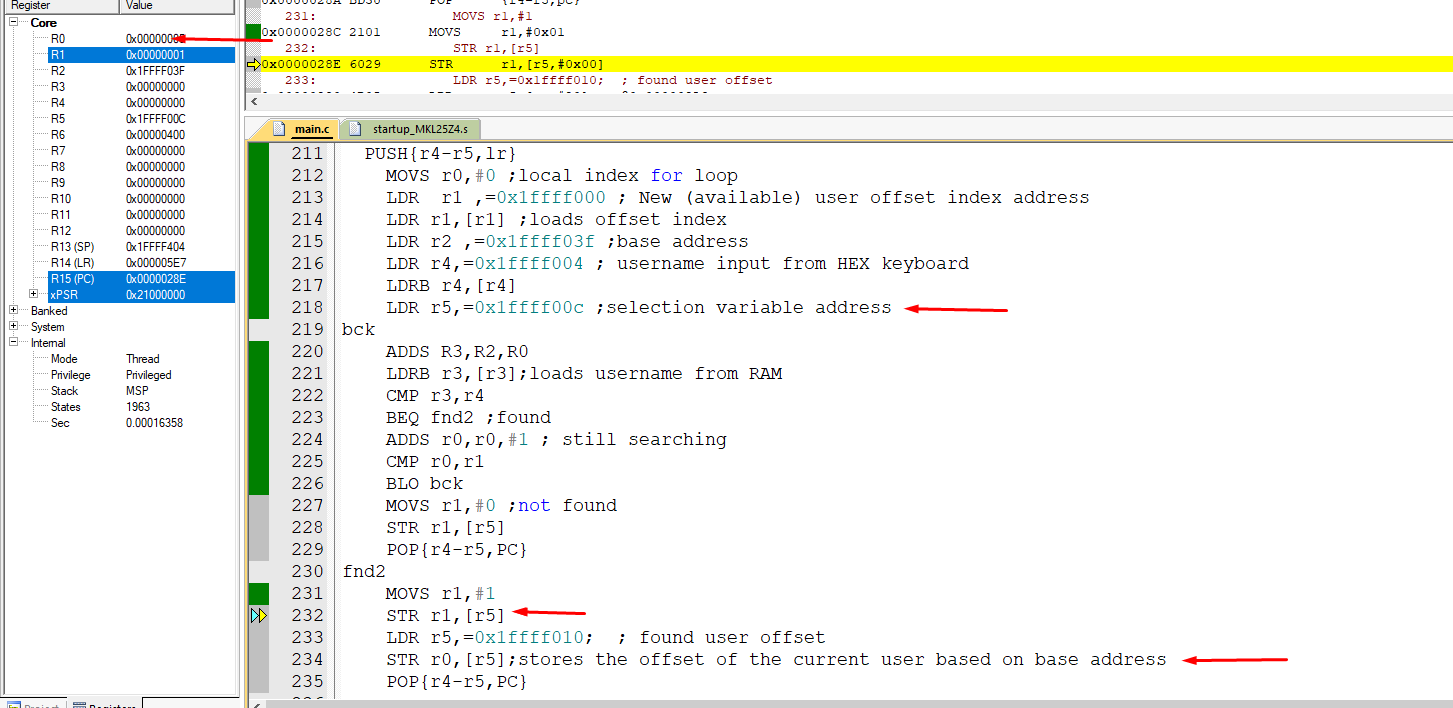
**username\_create() .**



Εδώ βλέπουμε τον καταχωρητή r2 ο οποίος έχει την είσοδο που πήραμε από το πληκτρολόγιο και τον καταχωρητή r3 ο οποίος δείχνει στην νέα διαθέσιμη θέση όπου και θα μπει το permission . Στην αρχή της συνάρτησης ήταν μια θέση μικρότερη άρα στην 4 . Επίσης βλέπω στην μνήμη ότι αποθηκεύτηκε το password . To παραπάνω έγινε με την συνάρτηση **password\_create() .**

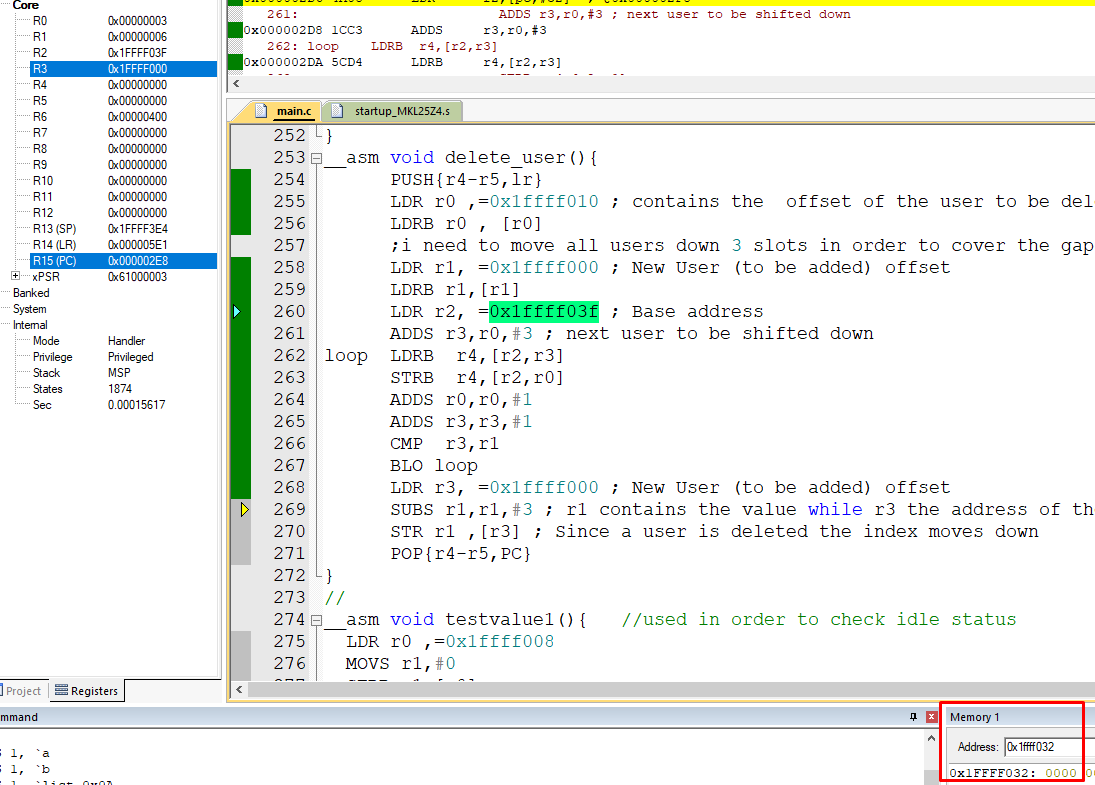


Εδώ βλέπουμε τον καταχωρητή r2 ο οποίος έχει την είσοδο που πήραμε από το πληκτρολόγιο και τον καταχωρητή r3 ο οποίος δείχνει στην νέα διαθέσιμη θέση όπου θα μπει ο νέος χρήστης . Εδώ ορίσαμε τα permissions να είναι 0 άρα θα είναι χρήστης . Αν είχαμε 1 τότε θα ήταν admin. Το παραπάνω έγινε με την συνάρτηση **permissions\_create()** . Στην περίπτωση που αντί για admin είχα user τότε θα είχαμε την συνάρτηση **default\_user\_perm()** όπου by default δίνει την τιμή 0 στον νέο χρήστη .

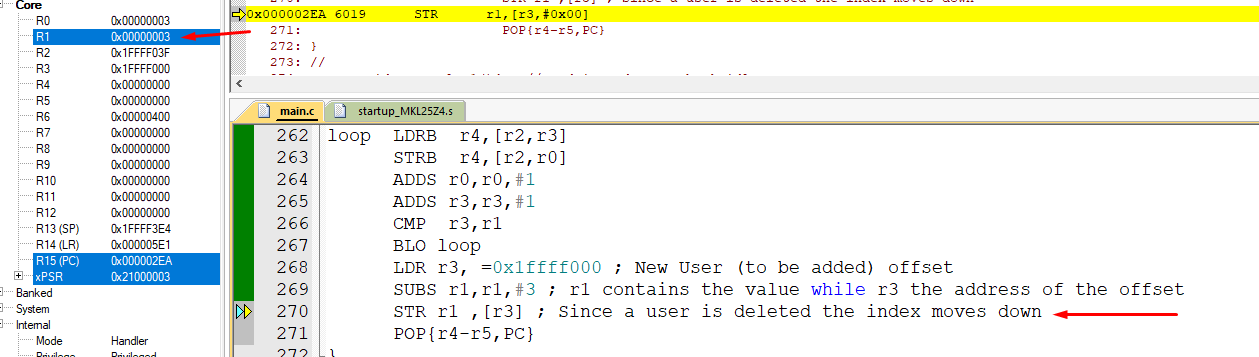
Μετά έχουμε την περίπτωση 5 η οποία είναι μόνο στην περίπτωση που έχουμε δικαιώματα admin και με αυτή μπορούμε να αλλάξουμε τα δικαιώματα των υπολοίπων καταχωρήσεων . Πριν γίνει αυτό καλούμε την συνάρτηση **find\_user\_position()** η οποία ψάχνει σειριακά ανά 3 θέσεις για να βρει το username και επιστρέφει την θέση του μέσω του δείκτη found user offset. Έστω ότι αλλάζω τα δικαιώματα του χρήστη που έβαλα πριν .

Εδώ όπως φαίνεται η συνάρτηση αρχικά στον δείκτη **selection variable** αποθηκεύει την τιμή του r1 ανάλογα με το αν βρέθηκε το username και έπειτα αποθηκεύει την τιμή του r0 που δείχνει το offset του χρήστη που βρέθηκε βάση του **base address** και το αποθηκεύει στο δείκτη **current user offset .**

Έπειτα με την συνάρτηση **change\_permissions()** ανάλογα με το offset που βρήκε για τον τωρινό χρήστη και προσθέτοντας +2 για να πάει στην περιοχή των permissions αλλάζει την τιμή της καταχώρησης βάση της τιμής που πήρε από το πληκτρολόγιο.

Τελικώς έχω την συνάρτηση **delete\_user()** για την περίπτωση 04 όπου διαγραφώ τον χρήστη ο οποίος πληκτρολογήθηκε . Εδώ πάλι θα χρησιμοποιήσω την συνάρτηση **find\_user\_position()** για να βρω τη θέση του χρήστη που θέλω να αφαιρέσω . Στην συγκεκριμένη μέθοδο μετά την αφαίρεση του ζητουμένου χρήστη κατεβάζω όλες τις καταχωρήσεις κατά 3 θέσεις ώστε να καλύψουν το κενό που δημιουργήθηκε και να μην υπάρχουν κενά στην μνήμη. Αρά είναι δυναμική διαγραφή. Επίσης αλλάζω και τον δείκτη **new user offset** τον οποίω μειώνω κατά 3 θέσεις .

Εδώ φαίνεται ότι οι 3 θέσεις που αφορούσαν το νέο χρήστη πλέον δεν υπάρχουν .Για περισσότερες εισαγωγές εκεί θα βρισκόταν ο επόμενος χρήστης .



Επίσης ο δείκτης που μας δείχνει την επόμενη διαθέσιμη θέση βρίσκεται στον καταχωρητή r1 και πλέον δείχνει 3 . Αυτό σημαίνει ότι με 3 μετατοπίσεις από την base address θα μπορούμε να τοποθετήσουμε το νέο χρήστη .Τα παραπάνω έγιναν με τη συνάρτηση **delete\_user() .**

ΣΥΝΑΡΤΗΣΕΙΣ ΚΑΙ ΔΕΙΚΤΕΣ

O δείκτης var(=var2) χρησιμοποιείται ως **selection variable** με τον οποίο μπορώ να ελέγχω τα διάφορα if στην main . O δείκτης αυτός βρίσκεται στην διεύθυνση 0x1ffff00c.

Ο δείκτης **offset** μας δείχνει πόσο απέχει ο χρήστης που ψάχνουμε /επιλέξαμε από την base address . O δείκτης αυτός βρίσκεται στην διεύθυνση 0x1ffff010.

Η είσοδος του πρώτου πληκτρολογίου βρίσκεται στην διεύθυνση

0x1ffff004.

Η είσοδος του δευτέρου πληκτρολογίου βρίσκεται στην διεύθυνση

0x1ffff008.

Η **base address** πάνω στην οποία ξεκινάνε και αποθηκεύονται τα στοιχεία των χρηστών είναι 0x1ffff03f.

O **new user offset** μας δείχνει την απόσταση από την base address στην οποία μπορώ να βάλω τον νέο χρήστη . Ο δεικτης αυτό βρισκεται στην διεύθυνση 0x1ffff000.

Με τις συναρτήσεις **testvalue** μπορώ να προσομοιάζω τις εισόδους των πληκτρολογίων .

Με την συνάρτηση **initialize()** ο χρήστης καλείται να εισάγει τα στοιχεία του πρώτου default admin. Έπειτα γίνονται κάποιες ακόμα αρχικοποιήσεις.

Με την συνάρτηση **idle\_state\_exit()**  ελέγχω το ποτέ ο εκλεκτής θα βγει από την idle κατάσταση το οποίο γίνεται με την είσοδο της τιμής 0.

Με την συνάρτηση **username\_create()**  ο χρήστης καλείται να εισάγει το username του χρήστη που δημιουργείται.

Με την συνάρτηση **password\_create()**  ο χρήστης καλείται να εισάγει το password του χρήστη που δημιουργείται.

Mε την συνάρτηση **admin\_status\_enable()** δίνεται στο νέο χρήστη by default admin privileges.

Με την συνάρτηση **username\_check()**  ελέγχεται αν υπάρχει το όνομα του χρήστη που πληκτρολογήθηκε .

Με την συνάρτηση **password\_check()**  ελέγχεται αν ο κωδικός που πληκτρολογήθηκε αντιστοιχεί στο σωστό κωδικό του ονόματος του χρήστη που πληκτρολογήθηκε πιο πριν .

Με την συνάρτηση **permission\_check() β**ρίσκουμε τι προνομία έχει ο χρήστης που πληκτρολογήθηκε .

Με την συνάρτηση **case\_select()** ανάλογα με την είσοδο επιλέγεται η κατάλληλη λειτουργεία του ελεγκτή .

Με την συνάρτηση **find\_user\_position()** βρισκω το offset του χρήστη που πληκτρολογήθηκε βάση το base address.

Με την συνάρτηση delete\_user() διαγράφω τον χρήστη που επιλέχθηκε .

Ταυτόχρονα με αυτό μετατοπίζω όλους τους χρήστες προς τα κάτω ώστε να καλυφθεί το κενό. Μειώνω ακόμη τον δείκτη **new user offset** κατά 3 .

ΚΩΔΙΚΑΣ

/\*----------------------------------------------------------------------------

\*----------------------------------------------------------------------------\*/

#include <MKL25Z4.H>

#include <stdio.h>

int \*var = (int\*)0x1ffff00c ; // selection variable

int \*offset = (int\*)0x1ffff010; // found user offset

int control=1;

\_\_asm void index\_init(){

LDR R0,=0x1ffff000 ;loads offset index address

MOVS R1,#0

STR R1,[R0]

BX LR

}

\_\_asm void username\_create(){

LDR r0 ,=0x1ffff000 ;new user offset index load

LDR r1 ,=0x1ffff03f ;base address load

LDR r2 ,=0x1ffff004 ;HEX keyboard input address

LDR r3,[r0] ; loads the offset index

ADDS r1,r3,r1; next available address to store

LDRB R2,[R2] ; loads the username value

STRB r2,[r1];stores the username in new available address strb since username is one byte

ADDS r3,r3,#1 ; plus one address is now available/empty

STR r3,[r0] ;stores the new index

BX LR

}

\_\_asm void password\_create(){

LDR r0 ,=0x1ffff000 ;new user offset index load

LDR r1 ,=0x1ffff03f ;base address load

LDR r2 ,=0x1ffff004 ;HEX keyboard input address

LDR r3,[r0] ; loads the offset index

ADDS r1,r3,r1; next available address to store

LDRB R2,[R2] ; loads the password value

STRB r2,[r1];stores the password in new available address strb since password is one byte

ADDS r3,r3,#1 ; plus one address is now available/empty

STR r3,[r0] ;stores the new index

BX LR

}

\_\_asm void permissions\_create(){

LDR r0 ,=0x1ffff000 ;new user offset index load

LDR r1 ,=0x1ffff03f ;base address load

LDR r2 ,=0x1ffff004 ;HEX keyboard input address contains permissions

LDR r3,[r0] ; loads the offset index

ADDS r1,r3,r1; next available address to store

LDRB R2,[R2] ; loads the permissions value

STRB r2,[r1];stores the permissions in new available address strb since password is one byte

ADDS r3,r3,#1 ; plus one address is now available/empty

STR r3,[r0] ;stores the new index

BX LR

}

\_\_asm void admin\_status\_enable(){

LDR r0 ,=0x1ffff000 ;new user offset index load

LDR r1 ,=0x1ffff03f ;base address load

LDR r2,[r0] ; loads the offset index

ADDS r1,r2,r1 ;next address available

MOVS r3,#1 ; 1 corresponds to admin status

STRB r3,[r1] ; stores the permission which is one byte

ADDS r2,r2,#1 ; plus one address is now available/empty

STR r2,[r0] ;stores the new index

BX LR

}

\_\_asm void username\_check(){

PUSH{r4-r6,LR}

MOVS r0,#0 ;local index for loop

LDR r1 ,=0x1ffff000 ; New (available) user offset index address

LDR r1,[r1] ;loads offset index

LDR r2 ,=0x1ffff03f ;base address

LDR r4,=0x1ffff004 ; username input from HEX keyboard

LDRB r4,[r4]

LDR r5,=0x1ffff00c ;selection variable address

back

ADDS r6,r2,r0 ;username address

LDRB r3,[r6];loads username from RAM

CMP r3,r4

BEQ fnd1 ; user was found

ADDS r0,r0,#1 ; still searching

CMP r0,r1

BLO back

MOVS r1,#1 ;not found

STR r1,[r5]

POP{r4-r6,PC}

fnd1

MOVS r1,#0

STR r1,[r5]

LDR r5,=0x1ffff010; ; found user offset

STR r0,[r5];stores the offset of the current user based on base address

POP{r4-r6,PC}

}

\_\_asm void default\_user\_perm(){

LDR r0 ,=0x1ffff000 ;new user offset index load

LDR r1 ,=0x1ffff03f ;base address load

LDR r2,[r0] ; loads the offset index

ADDS r1,r2,r1; next available address to store

MOVS r2,#0 ;sets user permissions

STRB r2,[r1];stores the permissions in new available address strb since password is one byte

ADDS r1,r1,#1 ; plus one address is now available/empty

STR r1,[r0] ;stores the new index

BX LR

}

\_\_asm void password\_check(){

LDR r0,=0x1ffff010; ; found user offset

LDR r0,[r0]

ADDS r0,r0,#1 ; plus in order to go to password address

LDR r1,=0x1ffff03f ;base address

LDRB r1,[r1,r0] ; password stored in RAM

LDR r2,=0x1ffff004 ; password input addr from HEX keyboard

LDRB r2,[r2] ;hex password

LDR r3 ,=0x1ffff00c; selection variable

CMP r1,r2

BEQ skp

MOVS r0,#1 ; 1 if not found

STR r0,[r3]

BX LR

skp

MOVS r0,#0 ; 0 if found

STR r0,[r3]

BX LR

}

\_\_asm void permission\_check(){

LDR r0,=0x1ffff010; found user offset

LDR r0,[r0] ; loads offset

ADDS r0,r0,#2 ; plus 2 in order to check permission status

LDR r1,=0x1ffff03f ;base address

ADDS r1,r0,r1

LDRB r2,[r1] ;loads permission

LDR r3 ,=0x1ffff00c ;selection variable

;var=1 admin var=0 user

CMP r2,#1

BEQ jmp

MOVS r0,#0

STR r0,[r3]

BX LR

jmp ;admin

MOVS r0,#1

STR r0,[r3]

BX LR

}

\_\_asm void case\_select(){

LDR r0 ,=0x1ffff00c ; selection variable

LDR r1 ,=0x1ffff008 ; Input address from the second keyboard

LDRB r1,[r1] ;second keyboard input

STRB r1,[r0] ; stores option selected from keyboard to selection variable

BX LR

}

\_\_asm void idle\_state\_exit(){

PUSH{LR}

LDR r0 ,=0x1ffff008 ; Input address from the second keyboard

LDRB r1,[r0] ; loads selected option

LDR r2 ,=0x1ffff00c ; selection variable address

CMP r1,#0 ;checks if selected option was 00

BEQ skip

MOVS r3,#0 ;0 if its not found

STR r3,[r2]

BX LR

//POP{PC}

skip

MOVS r3,#1 ;1 if its found

STR r3,[r2]

POP{PC}

//BX LR

}

\_\_asm void password\_update(){

LDR r0, =0x1ffff010 ; Current User offset

LDR r1, =0x1ffff03f ; Base address

LDR r2 ,=0x1ffff004 ; Input address hex keyboard

LDR r0,[r0] ; User offset

LDRB r2,[r2] ; User input, new password

ADDS r0,r0,#1 ; plus one to access password address

STRB r2,[r1,r0]

BX LR

}

\_\_asm void find\_user\_position(){

PUSH{r4-r5,lr}

MOVS r0,#0 ;local index for loop

LDR r1 ,=0x1ffff000 ; New (available) user offset index address

LDR r1,[r1] ;loads offset index

LDR r2 ,=0x1ffff03f ;base address

LDR r4,=0x1ffff004 ; username input from HEX keyboard

LDRB r4,[r4]

LDR r5,=0x1ffff00c ;selection variable address

bck

ADDS R3,R2,R0

LDRB r3,[r3];loads username from RAM

CMP r3,r4

BEQ fnd2 ;found

ADDS r0,r0,#3 ; still searching

CMP r0,r1

BLO bck

MOVS r1,#0 ;not found

STR r1,[r5]

POP{r4-r5,PC}

fnd2

MOVS r1,#1

STR r1,[r5]

LDR r5,=0x1ffff010; ; found user offset

STR r0,[r5];stores the offset of the current user based on base address

POP{r4-r5,PC}

}

\_\_asm void change\_permissions(){

LDR r0 ,=0x1ffff010 ; contains the offset

LDR r0 , [r0] ;

ADDS r0,r0,#2 ;plus two in order to go to the password position

LDR r1, =0x1ffff03f ; Base address

ADDS r1,r0,r1 ;address of the permissions

LDR r2 ,=0x1ffff004 ;permissions chosen by the user

LDRB r2,[r2]

STRB r2,[r1];stores the new permissions

BX LR

}

\_\_asm void delete\_user(){

PUSH{r4-r5,lr}

LDR r0 ,=0x1ffff010 ; contains the offset of the user to be deleted

LDRB r0 , [r0]

;i need to move all users down 3 slots in order to cover the gap

LDR r1, =0x1ffff000 ; New User (to be added) offset

LDRB r1,[r1]

LDR r2, =0x1ffff03f ; Base address

ADDS r3,r0,#3 ; next user to be shifted down

loop LDRB r4,[r2,r3]

STRB r4,[r2,r0]

ADDS r0,r0,#1

ADDS r3,r3,#1

CMP r3,r1

BLO loop

LDR r3, =0x1ffff000 ; New User (to be added) offset

SUBS r1,r1,#3 ; r1 contains the value while r3 the address of the offset

STR r1 ,[r3] ; Since a user is deleted the index moves down

POP{r4-r5,PC}

}

//

\_\_asm void testvalue1(){ //used in order to check idle status

LDR r0 ,=0x1ffff008

MOVS r1,#0

STRB r1,[r0]

bx lr

}

\_\_asm void testvalue2(){ //used in order to check username and password

LDR r0 ,=0x1ffff004

MOVS r1,#7

STRB r1,[r0]

bx lr

}

\_\_asm void testvalue3(){ //used in order to check the available options

LDR r0 ,=0x1ffff008

MOVS r1,#0

STRB r1,[r0]

bx lr

}

\_\_asm void testvalue4(){ //used in order to input new password

LDR r0 ,=0x1ffff004

MOVS r1,#6

STRB r1,[r0]

bx lr

}

\_\_asm void testvalue5(){ //used in order to create the new user

LDR r0 ,=0x1ffff004

MOVS r1,#9

STRB r1,[r0]

bx lr

}

\_\_asm void testvalue6(){ //used in order to create the permissions of the new user (admin case)

LDR r0 ,=0x1ffff004

MOVS r1,#0

STRB r1,[r0]

bx lr

}

//

void initialize(){

//index initialization

index\_init();

// printf("Welcome to admin initialization \n");

// printf("Please type your username \n");

testvalue2();

username\_create();

//printf("Please type your passworld \n");

testvalue2();

password\_create();

//admin status enable

admin\_status\_enable();

//printf("Exiting admin initialization \n");

return;

}

int main(void)

{

int \*var2=var;

initialize();

while(1){

//printf("IDLE STATE \n");

//printf("Press 00 to leave idle state\n");

testvalue1(); //second keyboard input

idle\_state\_exit();

if((\*var2)==0){ // if its 0 it returns to idle state due to wrong(not valid) input

//printf("Wrong input....returns to idle \n");

continue;}

//printf("Please type your username \n");

testvalue2(); //username input

username\_check();

if(\*var2){ // if its 1 it returns to idle state due to wrong(not valid) input

//printf("Wrong username....returns to idle \n");

continue;

}

//printf("Please type your password \n");

testvalue2();

password\_check();

if(\*var2){ // if its 1 it returns to idle state due to wrong(not valid) input

// printf("Wrong password....returns to idle \n");

continue;

}

//\*\*\*\*\*\*\*\*\*\*\* admin and user options\*\*\*\*\*\*\*\*\*\*\*\*//

permission\_check(); //decides if its user or admin

//var=1 admin var=0 user

if(\*var2){ //admin

while(control){

// printf("Available Options:\n 01 User Exit \n 02 Password Change \n 03 New User Creation \n 04 Delete User \n 05 Admin mode Enable/Disable \n");

// printf("Type your input \n");

testvalue3();

case\_select(); //input selection

switch(\*var2){

case 1:

control=0; // 01 exits while and then returns to idle since the loop is over

break;

case 2:

// printf("Type your new password \n");

testvalue4();

password\_update();

break;

case 3:

// printf("Type the username of the new user \n");

testvalue5();

username\_create();

// printf("Type the password of the new user \n");

testvalue5();

password\_create();

// printf("Type the permissions of the new user 1 for admin 0 for user \n");

testvalue6();

permissions\_create();

break;

case 4:

// printf("Please type the user to be deleted \n"); //his name

testvalue5();

find\_user\_position();

if(\*var2==0){

// printf("User not found returning to options \n");

break;}

delete\_user();

break;

case 5:

// printf("Please type user to have his permissions changed \n"); //his name

testvalue5();

find\_user\_position();

if(\*var2==0){

// printf("User not found returning to options \n");

break;}

// printf("Please type the permissions you want to change 1 for admin 0 for user \n");

testvalue6();

change\_permissions();

break;

default :

// printf("Invalid input \n" );

break;

}

}

}

else{ //user

while(control){

// printf("Available Options:\n 01 User Exit \n 02 Password Change \n 03 New User Creation \n ");

// printf("Type your input \n");

testvalue3();

case\_select();//input selection

switch(\*var2){

case 1:

control=0; // 01 exits while and then returns to idle since the loop is over

break;

case 2:

//printf("Type your new password \n");

testvalue4();

password\_update();

break;

case 3:

// printf("Type the username of the new user \n");

testvalue5();

username\_create();

// printf("Type the password of the new user \n");

testvalue5();

password\_create();

default\_user\_perm(); // a user can only create a user

break;

// normal user doesnt have the option to change permissions or delete users

default :

// printf("Invalid input \n" );

break;

}

}

}

control=1; // resets control status in order to be used for the next loop

}

while (1) ;

}

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ARM University Program Copyright © ARM Ltd 2013\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*