Εργαστήριο Προηγμένα Μικροϋπολογιστικά Συστήματα

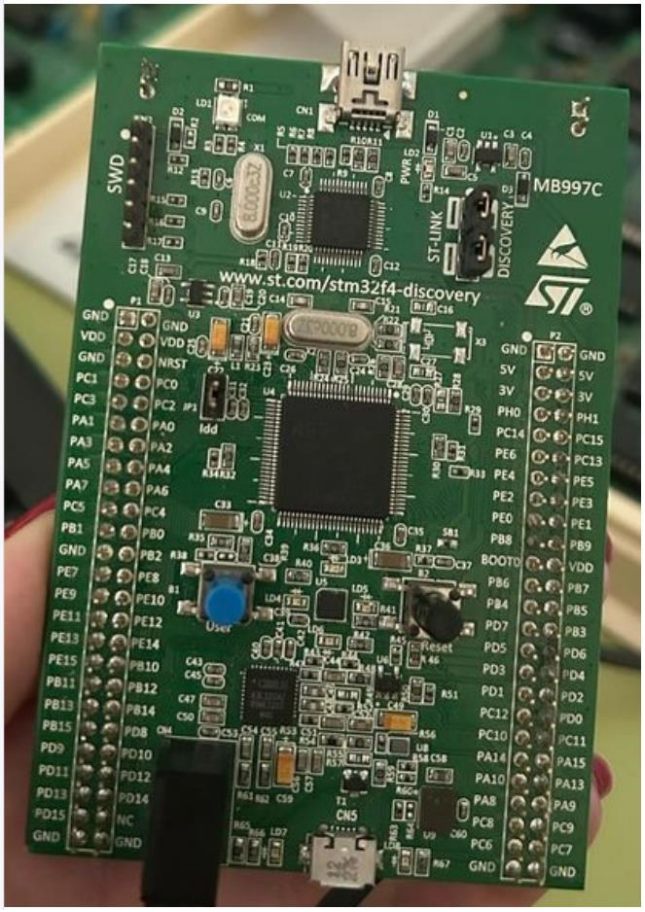
3η Εργαστηριακή Άσκηση

Ομάδα Β3

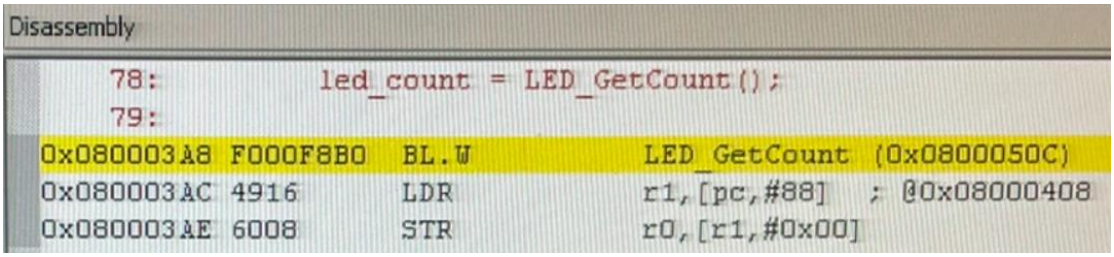
Κουκιάσας Δημήτριος 1072650

Μπασαγιάννη Γεωργία 1084016

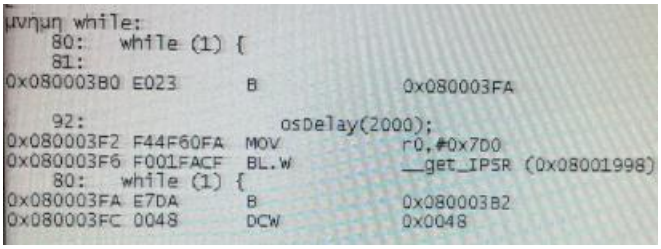
Η συγκεκριμένη εργαστηριακή άσκηση πραγματοποιήθηκε με τη χρήση ενός μικροϋπολογιστικού συστήματος (kit) το ARM. Συνδέσαμε το ARM με τον υπολογιστή και ανοίξαμε το Blinky_Vanila από το eclass, τρέχοντας τον κώδικα με το αντίστοιχο όνομα σε περιβάλλον Keil uVision5.



- Για να δούμε σε ποιες εντολές Assembly αναλύεται η εντολή **led_count = LED_GetCount();** ανοίγουμε το παράθυρο disassembly στο περιβάλλον που βρισκόμαστε και οι εντολές που ψάχνουμε είναι:



- Από το παράθυρο disassembly και πάλι βλέπουμε ότι η συνάρτηση **while(1)** αναλύεται στις εντολές που φαίνονται στο παρακάτω στιγμιότυπο:



Για να βρούμε πόση μνήμη καταλαμβάνει η συνάρτηση θα αφαιρέσουμε από τη διεύθυνση της τελευταίας εντολής assembly τη διεύθυνση της πρώτης, δηλαδή 0x80003FC – 0x80003B0. Άρα η διεύθυνση που καταλαμβάνει η συνάρτηση είναι 0x000004C.

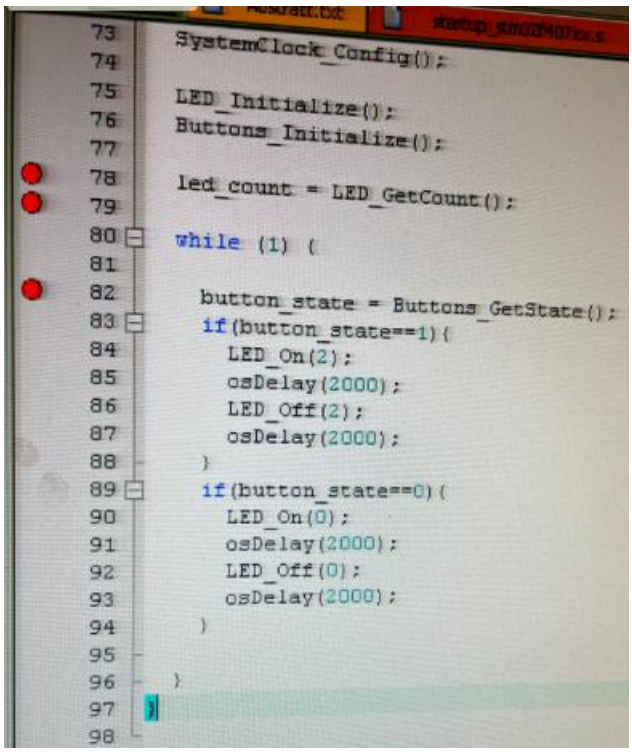
Συνεχίζουμε εισάγοντας τις μεταβλητές **led_count** και **button_state** στο παράθυρο watch και παρακολουθούμε την εξέλιξη των περιεχομένων της κάθε μεταβλητής.

- Για τη μεταβλητή **led_count** βλέπουμε το παράθυρο registers και συγκεκριμένα τον R1 καθώς εκεί αποθηκεύεται η μεταβλητή σε επίπεδο assembly. Έτσι, παρατηρούμε ότι η συγκεκριμένη μεταβλητή αποθηκεύεται στη διεύθυνση 0x00000010.
- Αντίστοιχα, για τη μεταβλητή **button_state** ακολουθούμε την ίδια διαδικασία και παρατηρούμε ότι αποθηκεύεται στη διεύθυνση 0x20000000.
- Τρόπος λειτουργίας επεξεργαστή: thread mode
- Επίπεδο προτεραιότητας κώδικα: privileged

Άσκηση 1

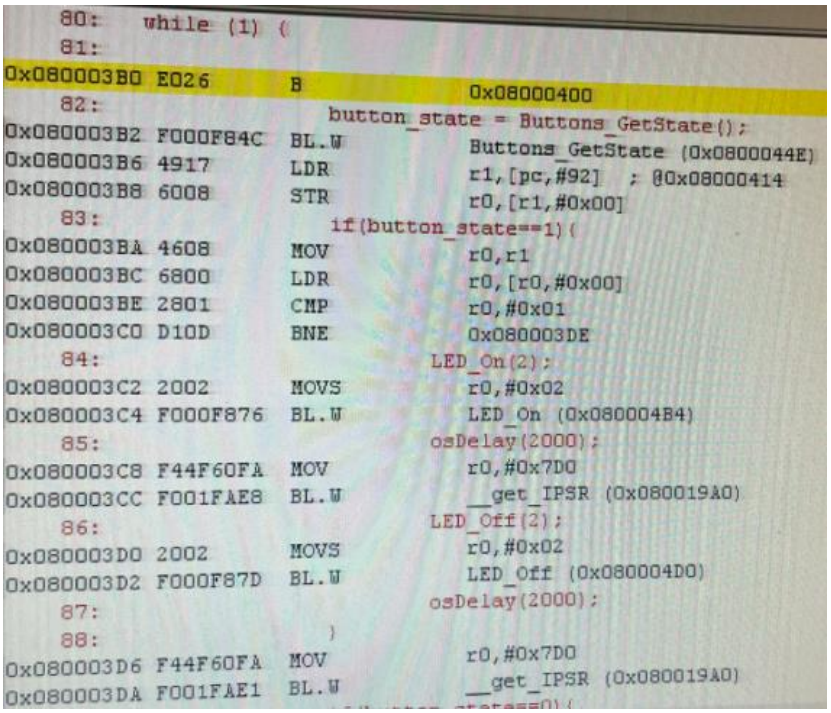
Στη συγκεκριμένη άσκηση τροποποιήσαμε τον κώδικα που μας δόθηκε με σκοπό να αναβοσβήνει το κόκκινο led αν το user button είναι πατημένο και να αναβοσβήνει το πράσινο led αν το user button δεν είναι πατημένο Ο κώδικας

που προέκυψε μετά τις τροποποιήσεις φαίνεται στην παρακάτω εικόνα και με την εκτέλεση του λάβαμε τα επιθυμητά αποτελέσματα.

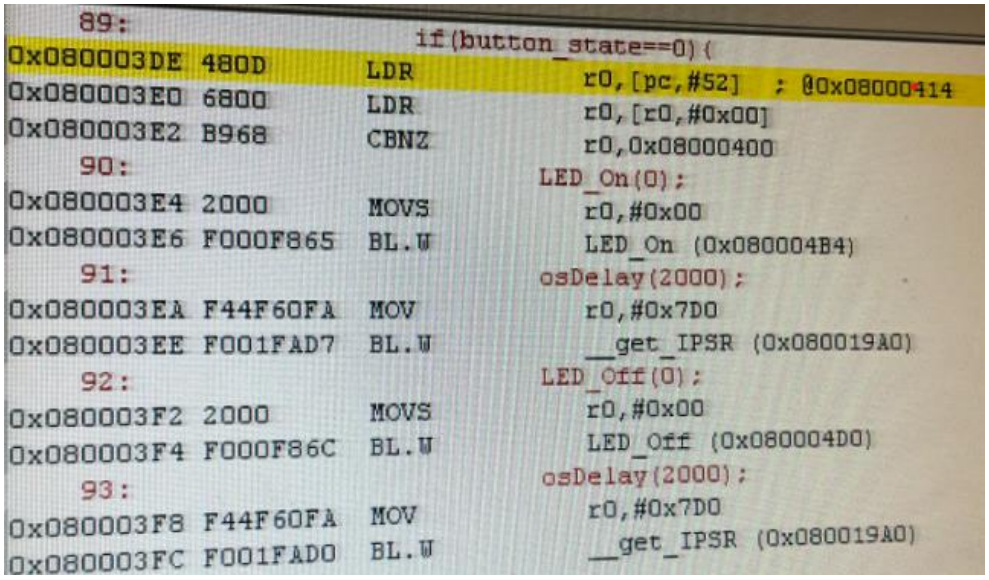


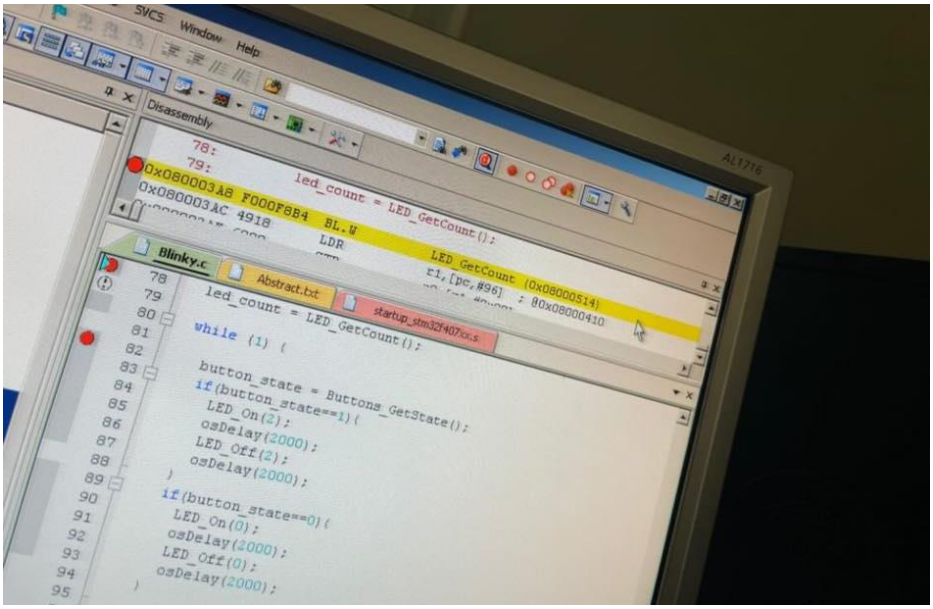
```
73  SystemClock_Config();
74
75  LED_Initialize();
76  Buttons_Initialize();
77
78  led_count = LED_GetCount();
79
80  while (1) {
81
82    button_state = Buttons_GetState();
83    if(button_state==1){
84      LED_On(2);
85      osDelay(2000);
86      LED_Off(2);
87      osDelay(2000);
88    }
89    if(button_state==0){
90      LED_On(0);
91      osDelay(2000);
92      LED_Off(0);
93      osDelay(2000);
94    }
95
96  }
97 }
98
```

- Για να βρούμε τις εντολές assembly της συνάρτηση while(1) του κώδικα μας, ανοίγουμε το παράθυρο disassembly και οι ζητούμενες εντολές φαίνονται στις δύο φωτογραφίες που ακολουθούν.



```
    80:   while (1) {
    81:
0x080003B0 E026      B          0x08000400
    82:     button_state = Buttons_GetState();
0x080003B2 F000F84C  BL.W       Buttons_GetState (0x0800044E)
0x080003B6 4917      LDR        r1,[pc,#92]  ; @0x08000414
0x080003B8 6008      STR        r0,[r1,#0x00]
    83:     if(button_state==1){
0x080003BA 4608      MOV        r0,r1
0x080003BC 6800      LDR        r0,[r0,#0x00]
0x080003BE 2801      CMP        r0,#0x01
0x080003C0 D10D      BNE        0x080003DE
    84:       LED_On(2);
0x080003C2 2002      MOVS       r0,#0x02
0x080003C4 F000F876  BL.W       LED_On (0x080004B4)
    85:       osDelay(2000);
0x080003C8 F44F60FA  MOV        r0,#0x7D0
0x080003CC F001FAE8  BL.W       __get_IPSR (0x080019A0)
    86:       LED_Off(2);
0x080003D0 2002      MOVS       r0,#0x02
0x080003D2 F000F87D  BL.W       LED_Off (0x080004D0)
    87:       osDelay(2000);
    88:     }
0x080003D6 F44F60FA  MOV        r0,#0x7D0
0x080003DA F001FAE1  BL.W       __get_IPSR (0x080019A0)
```



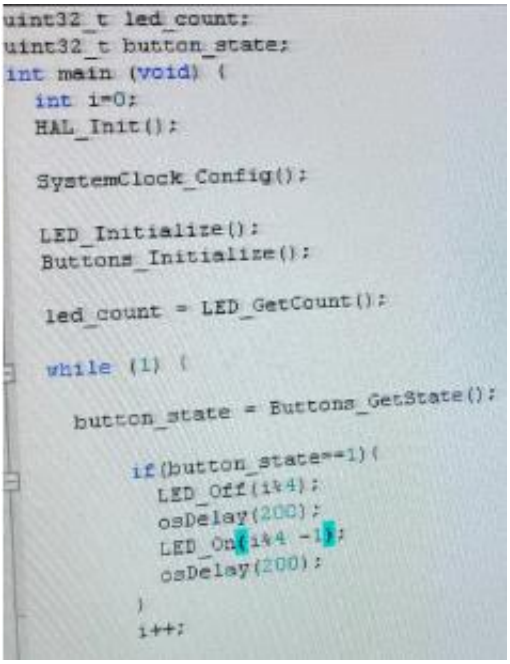


- Για τον εντοπισμό των απολύτων διευθύνσεων από τις οποίες ξεκινούν οι LED συναρτήσεις στηριζόμαστε ξανά στο παράθυρο disassembly.

Συνάρτηση	**Αρχική Διεύθυνση**
LED_GetCount	0x08000514
LED_On	0x080004B4
LED_Off	0x080004D0

Άσκηση 2

Για τη συγκεκριμένη άσκηση απαιτούταν η τροποποίηση του κώδικα έτσι ώστε κατά την εκτέλεση του σε κάθε πάτημα να γίνεται αλλαγή χρώματος. Ο κώδικας που προέκυψε μετά τις τροποποιήσεις φαίνεται στην παρακάτω εικόνα.



```
uint32_t led_count;
uint32_t button_state;
int main (void) {
  int i=0;
  HAL_Init();

  SystemClock_Config();

  LED_Initialize();
  Buttons_Initialize();

  led_count = LED_GetCount();

  while (1) {

    button_state = Buttons_GetState();

        if(button_state==1){
          LED_Off(i%4);
          osDelay(200);
          LED_On(i%4 -1);
          osDelay(200);

        }
        i++;
```

Ωστόσο κατά την εκτέλεση του κώδικα δεν άναβε ποτέ το LED 0 δηλαδή το πράσινο. Αυτό θα επιτευχθεί με την τροποποίηση της μεταβλητής i σε (i+1)%4 το οποίο και δεν προλάβαμε να δοκιμάσουμε.

Η while(1) θα έδειχνε στο παράθυρο disassembly με όμοιο τρόπο όπως και στην άσκηση 1 τις εντολές assembly από τις οποίες αποτελείται καθώς επίσης και οι απόλυτες διευθύνσεις για τις συναρτήσεις LED_GetCount, LED_On, LED_Off αλλάζουν αφού άλλαξε κι ο κώδικας αλλά δεν προλάβαμε να βγάλουμε τις απαραίτητες φωτογραφίες.