ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

ΤΜΗΜΑ ΗΜΜΥ. ΤΟΜΕΑΣ ΗΛΕΚΤΡΟΝΙΚΗΣ

ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΕΣ ΚΑΙ ΠΕΡΙΦΕΡΕΙΑΚΑ

2η Εργασία

24 Μαΐου 2020

Συντάκτης :
ΧΡΗΣΤΟΣ ΧΟΥΤΟΥΡΙΔΗΣ ΑΕΜ:8997
cchoutou@ece.auth.gr

Διδάσκον :
ΠΑΠΑΕΥΣΤΑΘΙΟΥ ΙΩΑΝΝΗΣ
ygp@ece.auth.gr

1. Εισαγωγη

Στην παρούσα εργασία το ζητούμενο ήταν η υλοποίηση μιας εφαρμογής για το αναπτυξιακό Nucleo-F401RE της εταιρίας ST Microelectronics, η οποία θα μετρά το χρόνο απόκρισης ενός ατόμου σε κάποιο οπτικό ερέθισμα. Η ανάπτυξη έγινε σε γλώσσα C και χρησιμοποιήθηκε το εργαλείο Keil uVision. Ομοίως και σε αυτή την εργασία, κατά την ανάπτυξη δεν προκύψαν ιδιαίτερα προβλήματα, επομένως η αναφορά θα αφιερωθεί κυρίως στην υλοποίηση και τον έλεγχο του προγράμματος. Ο κώδικας της εργασίας αλλά και της παρούσας αναφοράς υπάρχει και στο προσωπικό αποθετήριο για την εργασίας, του συντάκτη.

2. Παραδοτεα

Στο παραδοτέο .zip αρχείο μπορείτε να βρείτε:

- Τον κατάλογο **/src** που περιέχει τον κώδικα της εφαρμογής.
- Τον κατάλογο **/Libraries** που περιέχει τον κώδικα των βιβλιοθηκών που χρησιμοποιήσαμε, όπως το CMSIS και το ST HAL.
- Το αρχείο **report.pdf** που είναι η παρούσα αναφορά.
- Τον κατάλογο **/Keil** που περιέχει το project που χρησιμοποιήθηκε στο Keil.
  Στο project αυτό περιέχονται επιπλέων οι ρυθμίσεις καθώς και τα αρχεία από τις βιβλιοθήκες που χρησιμοποιήθηκαν.

Για την παρούσα εργασία χρησιμοποιήσαμε το CMSIS και το STM32F4xx_HAL που παρέχει η εταιρία ST στο επίσημο αποθετήριό της. Η επιλογή αυτού έναντι των αρχείων από το STM32F4xx Std_Peripheral που περιεχόταν στο e-learning του μαθήματος έγινε για να συμβαδίσουμε με τις βιβλιοθήκες που προτείνει ή κατασκευάστρια εταιρία. Επίσης η χρήση τους έγινε στο ίδιο χαμηλό επίπεδο της STM32F4xx_Std_Peripheral κάτι το οποίο προτείνεται και από τον διδάσκοντα. ***Εξάλλου μια υψηλότερου επιπέδου χρίση δεν θα είχε νόημα καθώς θα ξέφευγε από τον μαθησιακό χαρακτήρα της εργασίας***.

3. Υλοποιηση

Η υλοποίηση της εφαρμογής έγινε σε δύο μέρη. Στο πρώτο υλοποιήσαμε ένα driver interface wrapper πάνω από το Nucleo board. Σε αυτό υπάρχουν βασικές συναρτήσεις για την αρχικοποίηση και λειτουργία του board που αφορούν όμως μόνο τις λειτουργίες που χρειαζόμαστε για την εφαρμογή. Στο δεύτερο υλοποιήσαμε την "λογική" της εφαρμογής η οποία είναι ανεξάρτητη από το υλικό στο οποίο τρέχει. Με τον τρόπο αυτό μπορούμε να μεταφέρουμε την εφαρμογή σε διαφορετικό hardware υλοποιώντας μόνο τον driver και φροντίζοντας αυτός να παρέχει το ίδιο API.

3.1. Οδηγός του Nucleo

Ο επεξεργαστής που φέρει το εν λόγο αναπτυξιακό είναι πολύ μεγαλύτερος από τις ανάγκες της παρούσας εργασίας. Έτσι τα υποσυστήματα που χρησιμοποιήσαμε ήταν ελάχιστα. Για την ακρίβεια έγινε χρήση:

- Του **SysTick timer** ώς βάση μέτρησης χρόνου.
  Η συχνότητα που επιλέξαμε είναι το 1Khz, με αποτέλεσμα η ακρίβειά μας να περιορίζεται στο 1msec . Αυτό βέβαια δεν αποτελεί πρόβλημα καθώς ο μέσος χρόνος απόκρισης ενός ανθρώπου σε κάποιο οπτικό ερέθισμα είναι περίπου 200 msec, κάτι που θέτει το σφάλμα μας γύρω στο 0.5%.
- Του **GPIO** για την ανάγνωση του κουμπιού και την οδήγηση του LED της πλακέτας.
- Του **RCC** για τον έλεγχο και των σημάτων ρολογιού εσωτερικά του επεξεργαστή στα διάφορα υποσυστήματα.
- Του **Cycle count** μηχανισμού στον debugger.
  Ο εν λόγο μηχανισμός υλοποιήθηκε στο αρχείο του οδηγού αλλά **δεν χρησιμοποιήθηκε** στην εφαρμογή καθώς η ακρίβειά του είναι πολλές τάξεις μεγαλύτερη από την απαιτούμενη.

Αξίζει ίσως σε αυτό το σημείο να αναφέρουμε πως για την μέτρηση χρόνου χρησιμοποιήσαμε μεταβλητές για απαρίθμηση των διακοπών του SysTick. Η εφαρμογή έτσι μπορούσε βλέποντας την τιμή τους να έχει εικόνα του χρόνου που έχει περάσει από το power up.

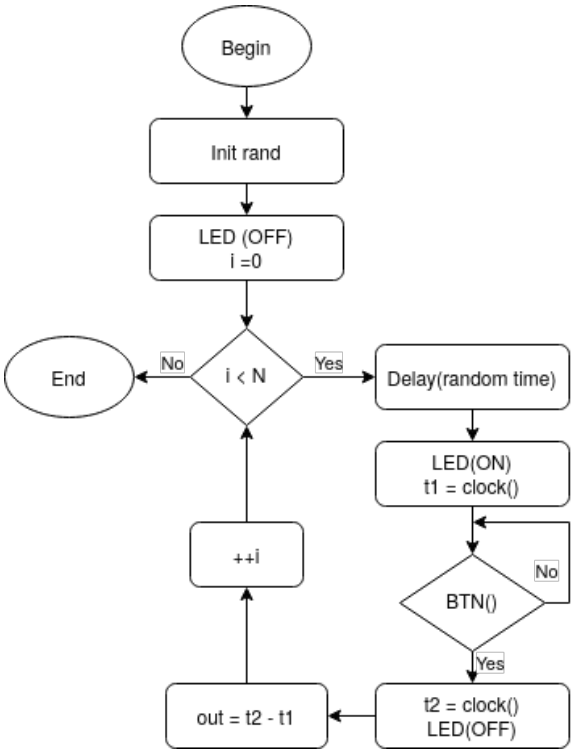
3.2. Εφαρμογή

Η εφαρμογή που αναλαμβάνει τη λογική του πειράματος αποτελείται από δύο τμήματα. Το τμήμα που αναλαμβάνει να πραγματοποιήσει τις μετρήσεις και το τμήμα που υπολογίζει τα στατιστικά. Στην εν λόγο εργασία μας ζητήθηκε η εμφάνιση των αποτελεσμάτων μέσω του debugger, έτσι δεν υπάρχει κάποιο τμήμα που να εμφανίζει δεδομένα στο χρήση.

Ποιο συγκεκριμένα για τις μετρήσεις έχουμε υλοποιήσει δύο συναρτήσεις.

- Την ***leading()*** η οποία εκτελεί το πείραμα του πρώτου τύπου.
- Την ***trailing()*** η οποία εκτελεί το πείραμα του δεύτερου τύπου.

Στο σχήμα 1 φαίνεται το διάγραμμα ροής της *leading()* . Πρόκειται για μια απλή blocking mode συνάρτηση η οποία σε βρόχο περιμένει ένα τυχαίο χρονικό διάστημα, έπειτα ανάβει το LED και ξανα-περιμένει μέχρι ο χρήστης να πατήσει το πλήκτρο. Στο βρόχο μετά το άναμμα του LED και μετά το πάτημα του πλήκτρου γίνεται ανάγνωση της CPU time (που αντιστοιχεί σε msec.) Έτσι ο συνολικός χρόνος απόκρισης του χρήστη είναι η διαφορά τους. Την διαφορά τους την αποθηκεύουμε σε ένα πίνακα στη μνήμη (όρισμα out) ώστε μετέπειτα να επεξεργαστούμε και να υπολογίσουμε τα στατιστικά.

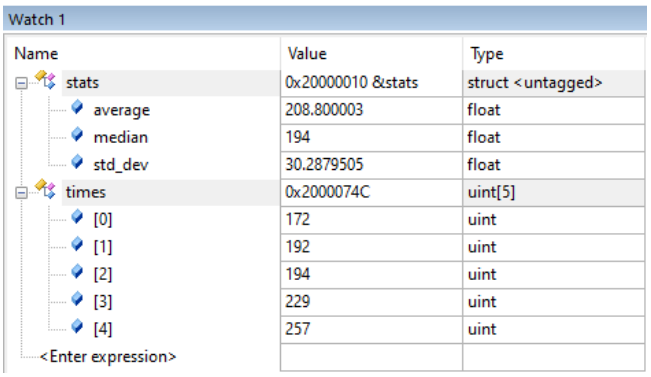


Σχήμα 1: *Διάγραμμα ροής του πρώτου mode λειτουργίας.*

Η συνάρτηση *trailing()* είναι όμοια με μόνη διαφορά τη συμπεριφορά του LED. Στην περίπτωση της *leading()* το ανάβουμε και μετράμε, ενώ στην περίπτωση της *trailing()* το σβήνουμε.

Για στατιστική ανάλυση των μετρήσεων υλοποιήσαμε 3 συναρτήσεις.

- Την ***average()*** η οποία υπολογίζει το μέσο όρο των μετρήσεων και επιστρέφει το αποτέλεσμα σε έναν αριθμό κινητής υποδιαστολής.



Σχήμα 2: *Στιγμιότυπο αποτελεσμάτων των συναρτήσεων για τα στατιστικά μέσω του debugger. Στη δομή stat κρατάμε τα στατιστικά στοιχεία και στον πίνακα times τις μετρήσεις από την τρέχουσα συνεδρία.*

- Την ***median()*** η οποία υπολογίζει τον διάμεσο των μετρήσεων και επιστρέφει το αποτέλεσμα σε έναν αριθμό κινητής υποδιαστολής.
- Την ***std_deviation()*** η οποία υπολογίζει την απόκλιση των μετρήσεων και επιστρέφει το αποτέλεσμα σε έναν αριθμό κινητής υποδιαστολής.

Ένα παράδειγμα αποτελεσμάτων από δεδομένα που τροφοδοτούνται σε αυτές τις συναρτήσεις φαίνεται και στο σχήμα 2.

4. Μετρησεις

Παρακάτω παραθέτουμε ένα πίνακα με μετρήσεις που κάναμε με το Nucleo. Εκτελέσαμε 5 μετρήσεις με το πρώτο mode λειτουργίας και με 5 με το δεύτερο.
Οι μετρήσεις από το πρώτο είναι:

A/A	$Average[msec]$	$Median[msec]$	$std.Deviation[msec]$
1	191.8	205	22.2566833
2	235.4	215	38.2130852
3	200.4	201	14.122323
4	201.4	183	23.3974361
5	207.0	185	36.4197769

Οι μετρήσεις από το δεύτερο είναι:

A/A	$Average[msec]$	$Median[msec]$	$std.Deviation[msec]$
1	226.2	208	34.1256523
2	224.8	230	18.0930939
3	208.8	197	34.3359871
4	217.0	215	22.7068272
5	227.4	215	29.8167744

5. Συμπερασματα - Παρατηρησεις

Συνοψίζοντας την εμπειρία μας με την παρούσα εργασία δεν έχουμε να αναφέρουμε κάποιο ιδιαίτερο πρόβλημα ή δυσκολία. Το μόνο μεμπτό σημείο, με το οποίο βέβαια δεν καθυστερήσαμε ιδιαίτερα ήταν η φτωχή υποστήριξη του Keil για την qsort. Από την εμπειρία μας με τον compiler gcc, ποτέ δεν είχαμε αντιμετωπίσει παρόμοια προβλήματα και η άρνηση του Keil ήταν κάτι που μας άφησε αρνητικές εντυπώσεις. Παρόλα αυτά δεν αναλωθήκαμε να ιχνηλατήσουμε το πρόβλημα, αλλά παρουσιάσαμε μια ταπεινή υλοποίηση της bobble sort, που για το μέγεθος των μετρήσεών μας είναι ιδανική.