

ΑΝΑΦΟΡΑ ΕΡΓΑΣΙΑΣ 2- Μικροεπεξεργαστές και Περιφερειακά

Στη δεύτερη εργασία κληθήκαμε να δημιουργήσουμε μια ενσωματωμένη συσκευή που θα μετρά πόσο γρήγορα μπορεί ένας άνθρωπος να πατήσει ένα διακόπτη ως απάντηση σε ένα *LED* που είτε ανάβει είτε σβήνει. Έτσι υλοποιήθηκε σε γλώσσα C, κώδικας που μετρά το χρόνο από τη στιγμή που το *LED* αναψε/εσβησε –ανάλογα ποιά εκδοχή του πειράματος εκτελείται- για 5 πειράματα, υπολογίζει το μέσο όρο και τον αποθηκεύει σε μία θέση μνήμης.

Ανάλυση Κώδικα

Στο αρχείο *main.c*, αφού εισάγουμε τις απαραίτητες βιβλιοθήκες, μέσα στην *main* συνάρτηση αρχικοποιούμε τα *LEDs* με κλήσης της *leds_init()* και της *leds_set(0,0,0)*, σετάρουμε τα *debug* σήματα και το διακόπτη της πλακέτας ώστε να λειτουργεί με *rising edge*, να είναι αρχικά συνδεδεμένος με την *pull up* αντίσταση της πλακέτας και να καλεί τον *interrupt handler button_press_isr* (του οποίου η λειτουργία θα εξηγηθεί παρακάτω) όταν πατηθεί. Έπειτα, ενεργοποιούμε τα *Interrupts* και θέτουμε κάποιες μεταβλητές που θα μας βοηθήσουν να υλοποιήσουμε την επιθυμητή λειτουργικότητα, όπως δύο *counters cycles* και *countExps* που θα μετρούν ο ένας τον αριθμό των κυκλών που πέρασαν έως ότου ο χρήστης πατήσει το διακόπτη από τη στιγμή που άναψε/έσβησε το *LED* και ο άλλος τον αριθμό των πειραμάτων που έχουν εκτελεστεί, αντίστοιχα. Επίσης, δηλώνουμε ένα πίνακα *arrayOfResponse* πέντε θέσεων όπου θα αποθηκεύονται οι χρόνοι από τα πέντε πειράματα που εκτελούνται, μια μεταβλητή *average* που θα κρατά τον μέσο όρο από αυτά τα πέντε πειράματα και θα τον αποθηκεύει τελικά σε μία θέση μνήμης και μια ακόμα μεταβλητή *random_delay* που θα αποθηκεύει τον –τυχαίο– χρόνο που θα μεσολαβεί μεταξύ των εκτελέσεων των πειραμάτων, κάνοντας την προσέγγιση πιο ρεαλιστική.

Μέσα σε μία ατέρμονη *while* υπάρχει το κυριο μέρος του κώδικα, ώστε αυτός να εκτελείται ασταμάτητα. Καθώς θέλουμε να εκτελούνται πεντάδες πειραμάτων, αμέσως μετά υπάρχει μια ακόμα *while* που θα τρέχει έως ότου η μεταβλητή *countExps* πάρει την τιμή 5, όταν δηλαδή θα έχουν εκτελεστεί 5 πειράματα. Μέσα στη δεύτερη *while*, αρχικά γίνεται *toggle* το *debug* σήμα της *main* και στη *random_delay* αποθηκεύεται η καθυστέρηση που έχει μια τυχαία τιμή μεταξύ 2 και 7 δευτερολέπτων. Στη συνέχεια, θα πρέπει να ελεγχθεί ποιος από τους δύο τύπους πειραμάτων πρέπει να εκτελεστεί, κάτι που γίνεται με τις εντολές *#ifdef, #ifndef* που ελέγχουν αν έχει γίνει ή όχι *define* του **TYPE_OF_EXPERIMENT**. Εάν το **TYPE_OF_EXPERIMENT** έχει οριστεί ("on") τότε θα εκτελεστεί το πρώτο πείραμα όπου ο χρήστης πρέπει να αντιδράσει όταν το *LED* ανάψει, αλλιώς το δεύτερο. Με τη χρήση των εντολών *#ifdef, #ifndef* επιτυγχάνουμε ο *compiler* να παρακάμψει το αντίστοιχο block εντολών που βρίσκεται μεταξύ των *#ifdef - #endif* και *#ifndef - #endif* εάν η αντίστοιχη συνθήκη (ο ορισμός ή ο μη ορισμός του **TYPE_OF_EXPERIMENT**) δεν ικανοποιείται. Εάν το **TYPE_OF_EXPERIMENT** έχει οριστεί (*#ifdef - #endif*) τότε καλείται η *delay_ms()* με όρισμα τη *random_delay* που μπλοκάρει την εκτέλεση του κώδικα για *random_delay ms*, για να υπάρχει μια τυχαιότητα όπως αναφέρθηκε και αμέσως μετά καλείται η *leds_set(1,1,1)* που ανάβει το *LED*. Εάν όμως το **TYPE_OF_EXPERIMENT** δεν έχει οριστεί, τότε μετά καλείται η *leds_set(1,1,1)* που ανάβει το *LED*, έπειτα η *delay_ms()* με όρισμα τη *random_delay* και μετά η *leds_set(0,0,0)* ώστε να σβήσει το *LED*. Από τη στιγμή που ανάψει/σβήσει το *LED* μέχρις ότου ο χρήστης πατήσει τον διακόπτη που θα σημάνει το σβήσιμο/άναμμα του *LED* μεσολαβεί κάποιος χρόνος ή μέτρηση του οποίου γίνεται με χρήση μιας *while* που ελέγχει μια μεταβλητή που ονομάζεται *stop* και αλλάζει τιμή μόνο μέσω του *interrupt handler* που καλείται με το πάτημα του κουμπιού. Η μεταβλητή *stop* έχει οριστεί ως *volatile* έξω από τη *main*, ώστε να αποφευχθούν οι βελτιστοποιήσεις του *compiler* και να αναγκαστεί να διαβάζει κάθε φορά τη τιμή της μεταβλητής από τη μνήμη. Μεσα στη *while* αυξάνουμε τη μεταβλητή *cycles* σε κάθε επανάληψη ενώ καλούμε και την *delay_cycles(1500)* που μπλοκάρει την εκτέλεση του προγράμματος για 1500 κυκλούς σε κάθε επανάληψη, κάνοντας έτσι την *cycles* να μετράει ποσες φορές πέρασαν 1500 κύκλοι. Με αυτόν τον τρόπο μετράμε τους κύκλους με περισσότερη ακρίβεια, ενώ ακόμα και στην περίπτωση που το κουμπί πατηθεί αμέσως πριν την κλήση της *delay*, η απόκλιση σε χρόνο θα είναι μικρή δεδομένου ότι το ρολόι του επεξεργαστή είναι στα 16MHz (μεταβλητή *SystemCoreClock*). Έτσι, όταν το κουμπί πατηθεί καλείται ο *interrupt handler*, μεσα στον οποίο πρώτα τίθεται 1 το *debug* σήμα του *isr*, ελέγχεται αν το σωστό *pin* του *source* είναι ενεργοποιημένο, που σημαίνει ότι όντως πατήθηκε το κουμπί, αλλάζει την τιμή της *stop* από 0 σε 1 και θέτει το *debug* σήμα στο 0. Τότε η *while* σταματάει και αμέσως μετά καλείται είτε η *leds_set(0,0,0)* –το *led* σβήνει– είτε η *leds_set(1,1,1)* –το *led* ανάβει–, ανάλογα τον τύπο του πειράματος (που ελέγχεται ξανά με *#ifdef, #ifndef*). Η μεταβλητή *cycles* αποθηκεύεται στον πίνακα *arrayOfResponse* και έπειτα μηδενίζεται για να μετρήσει τους κύκλους του επόμενου πειράματος, η μεταβλητή *stop* τίθεται ξανά 0 και η *countExps* αυξάνεται κατά ένα.

Μετά την εκτέλεση των πέντε πειραμάτων, ελέγχουμε ξανά αν έχει οριστεί **TYPE_OF_EXPERIMENT** ώστε σε περίπτωση που εκτελούνταν το δεύτερο πείραμα, το *led* να σβήσει. Παρακάτω με μια *while* υπολογίζουμε τον μέσο χρόνο που μας

ενδιαφέρει, πολλαπλασιάζοντας κάθε τιμή του πίνακα με το 1500 (για να βρούμε τον πραγματικό αριθμό των κύκλων) και με την περίοδο του επεξεργαστή (0.0000625-για να βρούμε τον πραγματικό χρόνο σε ms) ,αθροίζοντας όλα τα στοιχεία και διαιρώντας με το πέντε. Ο μέσος όρος των χρόνων αντίδρασης(σε ms)που προκύπτει αποθηκεύεται σε μία θέση μνήμης που επιλέχθηκε ,με τη βοήθεια μιας συνάρτησης σε *assembly* που παίρνει σαν όρισμα την τιμή του μέσου όρου και την κάνει *store* στην επιλεγμένη διεύθυνση. Τέλος, η *countExps* τίθεται στο 0(ώστε να είναι έτοιμη να μετρήσει την επόμενη πεντάδα πειραμάτων) ,οπως και ο μετρητής *i* και η μεταβλητή *average* ,ενώ για να είναι αντιληπτό ,οτι η πεντάδα πειραμάτων έχει τελειώσει, καλούμε την ***delay_ms***(15000) που θα κρατήσει το *led* σβηστό προτού ξεκινήσει η επόμενη.

Testing

Για να διαπιστωθεί η σωστή λειτουργία της εφαρμογής χρησιμοποιήθηκε το ***watch window*** του *keil* σε συνδιασμό με κάποια ***break points*** σε διάφορα σημεία του κώδικα για να ελεγχθεί οτι κάποιες καίριες μεταβλητές παίρνουν σωστές τιμές . Κάποιες απο αυτές που ελέχθηκαν ήταν : η *random_delay* ωστε να έχουμε διαφορετική καθυστέρηση μεταξύ δύο πειραμάτων ,ο πίνακας *arrayOfResponse* με τους χρόνους αντίδρασης ώστε να σιγουρευτώ οτι οι χρόνοι ανταποκρίνονται στην πραγματικότητα καθώς επίσης και η διεύθυνση μνήμης για να διαπιστωθεί ότι γίνεται σωστά το *store* σε αυτή, σε κάθε πεντάδα πειραμάτων.

Μετρήσεις

Στον παρακάτω πίνακας φαίνονται οι τιμές των χρόνων αντίδρασης του χρήστη για 4 διαφορετικές πεντάδες πειραμάτων πρώτου τύπου(όπου ο χρήστης καλείται να πατήσει το διακόπτη όταν το LED ανάψει).

	1 ^η αντίδραση	2 ^η αντίδραση	3 ^η αντίδραση	4 ^η αντίδραση	5 ^η αντίδραση	Μέση τιμή
1 ^{ος} Γύρος	370ms	301ms	270ms	280ms	276ms	299ms
2 ^{ος} Γύρος	329ms	490ms	313ms	312ms	296ms	348ms
3 ^{ος} Γύρος	364ms	306ms	292ms	245ms	300ms	301ms
4 ^{ος} Γύρος	316ms	327ms	286ms	270ms	288ms	297ms

Οι τιμές αυτές ελήφθησαν με χρήση του *watch window* του *keil* και ενός *break point* αμέσως μετά τον υπολογισμό και της μέσης τιμής. Στην εικόνα 1 φαίνεται, ο πίνακας *arrayOfResponse* στον οποίο τοποθετήθηκαν οι χρόνοι(σε ms) των αντιδράσεων του πρώτου γύρου πειραμάτων καθώς και η τιμή του *average* που προκύπτει ενώ στην εικόνα 2 βλέπουμε την τιμή του μέσου όρου να έχει αποθηκευτεί στην διεύθυνση 0x20000450.

Name	Value	Type
arrayOfResponse	0x20000740	_int64[5]
[0]	370	_int64
[1]	301	_int64
[2]	270	_int64
[3]	280	_int64
[4]	276	_int64
average	299	_int64

Εικόνα 1: Watch Window του Keil

Address	Value
0x20000450	0000000299

Εικόνα2: Memory Window