Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Εθνικό Μετσόβιο Πολυτεχνείο

Εργαστήριο Μικροϋπολογιστών

Τέταρτη εργαστηριακή άσκηση

Σπουδαστές
Παπαλεξανδράκης Εμμανουήλ (Α.Μ.: 03114203)
Παπασκαρλάτος Αλέξανδρος (Α.Μ.: 03111097)

Ημερομηνία Υποβολής Αναφοράς: 11 Νοεμβρίου 2018

Θα παρουσιάσουμε μια σύντομη και ουσιαστική ανάλυση των προγραμμάτων που κατασκευάσαμε.
Τα προγράμματα αυτά καθ' αυτά είναι σε ξεχωριστά αρχεία.
Κάποιες λεπτομέρειες θα παραλειφθούν, αλλά εάν ο αναγνώστης ενδιαφέρεται, μπορεί να ανατρέξει στον κώδικα και στα σχόλια αυτού, όπου έχουμε μια πιο αναλυτική παρουσίαση.

Άσκηση 1

Κατασκευάζουμε το πρόγραμμα που ζητείται.

Καταρχάς, έχουμε τις αναγκαίες αρχικοποιήσεις, για το δείκτη στοίβας, τον ορισμό των μεταβλητών-καταχωρητών, τον ορισμό εισόδων-εξόδων, την επίτρεψη των κατάλληλων διακοπών.

Έπειτα, το κύριο μέρος του προγράμματος είναι αρκετά απλό. Πρόκειται απλά για ένα αέναο loop κατά το οποίο, αυξάνουμε ένα μετρητή, προβάλλουμε το αποτέλεσμα στα led της θύρας A και περιμένουμε 200 msec (αξιοποιώντας τις γνωστές ρουτίνες χρονοκθυστέρησης).

Όταν θα προκύψει διακοπή, φτάνοντας στην αντίστοιχη ρουτίνα εξυπηρέτησης, πρώτα απ' όλα, πρέπει να λύσουμε το πρόβλημα του σπινθηρισμού. Λοιπόν, μηδενίζουμε το αντίστοιχο bit στον καταχωρητή GIFR (γράφωντας 1) και περιμένουμε ένα αμελητέο χρονικό διάστημα πριν ελέγξουμε την τιμή του. Εάν στον έλεγχο η τιμή του είναι 0, τότε συνεχίζουμε. Διαφορετικά, επαναλαμβάνουμε τη διαδικασία μέχρις ότου η τιμή του ισορροπήσει στο 0, και άρα δεν έχουμε επιπλέον αιτήματα διακοπής.

Στη συνέχεια, ελέγχουμε το bit PD0 που αντιστοιχεί στο διακόπτη/κουμπί 0 της θύρας D και ανν είναι στο ON, αυξάνουμε ένα μετρητή έγκυρων διακοπών και προβάλλουμε στα led της θύρας B. Έπειτα, σε κάθε περίπτωση, επιστρέφουμε στο κύριο πρόγραμμα.

Άσκηση 2

Το παρόν πρόγραμμα είναι ακριβώς όμοιο με την προηγούμενη άσκηση με εξαίρεση τη λειτουργία που επιτελείται στη διακοπή καθώς και το είδος της διακοπής που επιτρέπουμε. Πριν είχαμε INT0, ενώ τώρα έχουμε INT1.

Έχουμε αρχικοποιήσεις και κύριο πρόγραμμα μετρητή όπως πριν.

Στη ρουτίνα εξυπηρέτησης της διακοπής, καταρχάς ασχολούμαστε με το σπινθηρισμό ακριβώς όπως πριν.

Στη συνέχεια, λαμβάνουμε είσοδο από τη θύρα bit, και με 8 διαδοχικές περιστροφές και ελέγχους του bit 0, εξετάζουμε στην ουσία κάθε bit. Κάθε φορά που συναντάμε άσσο, αυξάνουμε ένα μετρητή διακοπτών κατά 1. Προβάλλουμε στα led της θύρας C και επιστρέφουμε στο κύριο πρόγραμμα.

Άσκηση 3

Κατασκευάζουμε το πρόγραμμα που ζητείται.
Αλγοριθμικά, ο κώδικας στην assembly είναι όμοιος με αυτόν στη C.

Σε αυτό το πρόγραμμα θα παίξουμε με το χρονιστή TCNT1.
Καταρχάς, κάνουμε κάποιους υπολογισμούς. Με συχνότητα ρολογιού CLK του AVR στα 8MHz και το prescaler να διαιρεί/επιβραδύνει τη συχνότητα κατά 1024 στο χρονιστή, έχουμε πως το 1 πραγματικό δευτερόλεπτο αντιστοιχεί σε $8\ 10^6 / 1024 =$ 7812,5 κύκλους του χρονιστή, αρα ο χρονιστής έχει συχνότητα 7812,5Hz.
Προκύπτει πως για 0.5 sec έχουμε 3905.25 κύκλους
2.5 sec έχουμε 19531.25 κύκλους
3 sec έχουμε 23437.5 κύκλους
Επειδή θα αξιοποιήσουμε τη διακοπή υπερχείλισης του χρονιστή, δηλαδή τη διακοπή που θα καλείται όταν ο χρονιστής φτάνει στο όριο του (65536 κύκλοι), θα βρούμε τις αντίστοιχες αρχικές τιμές του χρονιστή, ώστε αυτό να υπερχειλίσει μετά από το ζητούμενο διάστημα:
0.5 sec → αρχική τιμή: 65536 - 3906,25 = 61629,75 ~ 61629 κύκλοι
2.5 sec → αρχική τιμή: 65536 - 19531,25 = 46005,75 ~ 46005 κύκλοι
3 sec → αρχική τιμή: 65536 - 23437,5 = 42098,5 ~ 42098 κύκλοι

Και τώρα ξεκινάει ο κώδικας. Κάνουμε τις κατάλληλες αρχικοποιήσεις συμπεριλαμβανομένου την αρχικοποίηση του χρονιστή. Για την ώρα φροντίζουμε να έχει μηδενική τιμή και να μη μετράει (δηλαδή TCCR1B = 0x00).
Σημειώνουμε πως ορίζουμε και έναν καταχωρητή flag ως σημαία για το αν έχει πατηθεί ένα από τα δοσμένα κουμπιά τα τελευταία 3 δευτερόλεπτα.

Στο κύριο πρόγραμμα, ελέγχουμε συνεχώς το bit 0 της θύρας εισόδου B. Όσο αυτό είναι clear, περιμένουμε. Όταν γίνει set, επίσης περιμένουμε μέχρις ότου ξαναγίνει clear (επιτελούμε τη λειτουργία στο “άφημα” του κουμπιού).
Η κάτωθι λειτουργία θα εκτελεστεί όμοια όταν, αντ’ αυτού, ενεργοποιήσουμε τη διακοπή INT0.

Όταν θα γίνει το παραπάνω, ελέγχουμε το flag.
Εάν το flag είναι clear, αυτό σημαίνει πως πρόκειται για το πρώτο πάτημα (στα τελευταία 3 δευτερόλεπτα). Συνεπώς, θέτουμε το flag, ανάβουμε μόνο το MSB led, και αρχικοποιούμε το χρονιστή να υπερχειλίσει μετά από 3 sec. Ξεκινάμε το χρονιστή με ταχύτητα CLK/1024.
Εάν, από την άλλη, το flag είναι ήδη set, τότε πρέπει να γίνει ανανέωση. Συνεπώς, ανάβουμε όλα τα led και αρχικοποιούμε το χρονιστή να υπερχειλίσει μετά από 0.5 sec. Ξεκινάμε το χρονιστή με ταχύτητα CLK/1024.

Τα υπόλοιπα τα αναλαμβάνει η ρουτίνα εξυπηρέτησης διακοπής υπερχείλισης του χρονιστή.
Μπαίνοντας στη ρουτίνα αυτή, υπάρχουν δύο ενδεχόμενα. Είτε ο χρονιστής υπερχείλισε μετά από 3 sec οπότε και πρέπει απλά να σβήσουμε τα led ή ο χρονιστής υπερχείλισε μετά από τα πρώτα 0.5 sec της ανανέωσης.
Λοιπόν, καταρχάς ελέγχουμε εάν το LSB led είναι ON. Το LSB led είναι ενδεικτικό.

Εάν είναι ON, τότε όλα τα led είναι ON (δεν υπάρχει λειτουργία σε αυτήν την άσκηση που να ανάβει το LSB αλλά όχι όλα τα led). Οπότε, αφού όλα τα LSB έιναι ON, μόλις πέρασαν τα πρώτα 0.5 sec της ανανέωσης.
Συνεπώς, ανάβουμε μόνο το MSB led (σβήνοντας τα υπόλοιπα) και θέτουμε το χρονιστή να υπερχειλίσει μετά από 2.5 sec, καθώς έχουν περάσει ήδη 0.5 sec από τη στιγμή που πατήθηκε το κουμπί (και είναι 0.5 + 2.5 = 3 sec).

Διαφορετικά, εάν το LSB είναι OFF, αυτό σημαίνει πως μόνο το MSB είναι ON. Εφόσον μόλις υπερχείλισε ο χρονιστής, αυτό σημαίνει πως μόλις πέρασαν τα 3 sec (ή ισοδύναμα τα 0.5+2.5 sec) που το MSB πρέπει να μείνει αναμμένο.
Συνεπώς, σβήνουμε όλα τα led, κάνουμε clear το flag και σταματάμε το χρονιστή.

Προφανώς, και στις δύο περιπτώσεις, στο τέλος της ρουτίνας επιστρέφουμε στο κύριο πρόγραμμα.