Εργαστήριο 2

Αλέξανδρος Νικολάου (10367), Φίλιππος Ρωσσίδης (10379)

15 Μαΐου 2025

Εισαγωγή

Υλοποιούμε την παρακάτω λειτουργία στο board:

Μέσω UART ο χρήστης καλείται να δώσει έναν αριθμό. Οτιδήποτε δεν είναι αριθμός ή παύλα αγνοείται. Όταν πατηθεί το Enter ('\r') ο αριθμός αναλύεται: κάθε 0.5 δευτερόλεπτα διαβάζεται ένας χαρακτήρας όπου

- Αν είναι ζυγός: αναβοσβήνει το LED με περίοδο 400 ms (200 ms on, 200 ms off)
- Αν είναι περιττός: το LED αλλάζει κατάσταση
- Αν είναι παύλα ('-') ο αριθμός η διαδικασία επαναλαμβάνεται με τον αριθμό από την αρχή.

Αν πατηθεί το κουμπί κατά τη διάρκεια της ανάλυσης, το LED παγώνει στην τρέχουσα κατάσταση, η ανάλυση συνεχίζει κανονικά. Αν ξαναπατηθεί ξεπαγώνει. Το πλήθος των φορών που πατήθηκε μετριέται και τυπώνεται σε κάθε πάτημα. Αν πατηθεί κατά τη διάρκεια εισόδου αριθμού από τον χρήστη υιοθετούμε την προσέγγιση ότι δεν γίνεται τίποτα, εκτός από το να προστεθεί στο πλήθος πατημάτων. Εάν όλα τα ψηφεία του αριθμού περάσουν από την ανάλυση και ο αριθμός δεν τελειώνει σε παύλα, ή πατηθεί κάποιο πλήκτρο κατά την διάρκεια της ανάλυσης η διαδικασία εκτελείται από την αρχή.

Υπάρχει αναλυτικός σχολιασμός στον κώδικα ώστε να αποφευχθεί να μιλάμε για κάθε γραμμή στην αναφορά.

UART

Χρησιμοποιείται ο κώδικας που μας δόθηκε με τις απαραίτητες αλλαγές για την υλοποίηση των παραπάνω: Όταν δίνεται ένας χαρακτήρας από το πληκτρολόγιο, καλείται ο ISR (Interrupt Service Rutine) ο οποίος ελέγχει εάν ο χαρακτήρας που δόθηκε αντιστοιχεί σε τιμή ASCII και έπειτα τοποθετείται σε μία ουρά. Ορίζεται επίσης ένας πίνακας char με όνομα (buff) ο οποίος θα περιέχει την ολοκληρωμένη είσοδο, και μια θετική ακέραια τιμή buff_index η οποία κατά την είσοδο του αριθμού δείχνει την επόμενη θέση του buff όπου μπορεί να προστεθεί ψηφείο, και μετά την είσοδο δείχνει το τελευταίο στοιχείο. Είναι απαραίτητη διότι ο πίνακας buff έχει fixed size 128 chars.

Στην main υπάρχουν δύο σημεία στα οποία μπορεί να έρθει ο χαρακτήρας, εάν βρισκόμαστε σε κατάσταση όπου περιμένουμε είσοδο, ή κατά την ανάλυση των ψηφείων.

Στην πρώτη περίπρωση, βρισκόμαστε στο __WFI (Wait For Interrupt) στο σώμα κώδικα εντός μιας while όπου διαβάζονται οι χαρακτήρες. Μόλις έρθει το interrupt από τον ISR ελέγχεται ο χαρακτήρας, εάν είναι backspace τότε απλά τυπώνεται στην οθόνη ώστε να διαγραφτεί ένας χαρακτήρας και μειώνεται κατά ένα ο buff_index. Εάν δεν είναι backspace ελέγχεται αν είναι ανάμεσα στο 0,9 ή '-' ή '\r', τότε προστίθεται στον buff, στη θέση buff_index, η οποία έπειτα αυξάνεται κατά 1. Η while όπου διαβάζονται οι χαρακτήρες τερματίζει μόλις δει τον χαρακτήρα '\r', ή αν γεμήσει ο buff. Τέλος προστίθεται στην θέση buff_index ο χαρακτήρας '\0' για να δηλώνει τη λήξη του string και έτσι ο buff_index δείχνει τώρα στον τελευταίο χαρακτήρα.

Στην δεύτερη περίπτωση, μόλις έρθει ο interrupt εξαγόμαστε από την λούπα όπου διαβάζονται οι χαρακτήρες ώστε να αρχικοποιηθούν ξανά οι μεταβλητές και να ζητηθεί καινούργιος αριθμός.

—— Προτεινόμενη αλλαγή, να τρέξει στον ISR της UART αν $inputphase = 0$ τότε να τυπώσει $newinput$ και να θέσει $count = buffer intex - 1$ Η λουπα θα είναι απλά ωηιλε (ςντ != βυυφ ινδεξ) ΩΦΙ ——

Ο buff ορίζεται ως global μεταβλητή ώστε να μπορεί να διαβαστεί από τις ISR των interrupts.

GPIO Ακροδέκτες

Για την συγκεκριμένη εργασία, χρειάστηκε επιπλέον, να αξιοποιήσουμε και ορισμένους GPIO ακροδέκτες. Πιο συγκεκριμένα, εκείνους τους ακροδέκτες που αντιστοιχούν στο κουμπί του χρήστη πάνω στην πλακέτα και του πράσινου LED. Για τον ακροδέκτη του LED ανατρέξαμε στο αρχείο platform.h, ώστε να βρούμε σε ποια τιμή αντιστοιχεί, ώστε να την χρησιμοποιήσουμε στην αρχικοποίησή του (P_LED_R). Για αρχή καλούμε την

```
gpio_set_mode(P_LED_R, Output);
```

, η οποία ορίζει την λειτουργία του ακροδέκτη ως **έξοδος** και στέλνει ψηφιακά σήματα προς το εξωτερικό κύκλωμα. Με αυτόν τον τρόπο μπορούμε να ορίζουμε την τάση του ακροδέκτη ως HIGH - LED_ON (τιμή 1) ή LOW - LED_OFF (τιμή 0). Για τον σκοπό αυτό χρησιμοποιούμε την συνάρτηση

```
gpio_set(P_LED_R, LED_ON ή LED_OFF);
```

Όσον αφορά το κουμπί του χρήστη, τα βήματα για να το χρησιμοποιήσουμε είναι τα εξής:

- Ορίζουμε τον ακροδέκτη που του αντιστοιχεί (P_SW) σε λειτουργία PullUp (δηλαδή τον ορίζουμε ως είσοδο και λογικό υψηλό του (1) ορίζεται όταν ο διακόπτης δεν είναι πατημένος, ενώ το λογικό χαμηλό (0) όταν πατιέται).
- Έπειτα, ορίζουμε την συνάρτηση που θα καλείται κάθε φορά που θα πατιέται το κουμπί, ώστε να προκαλείται το interrupt. Χρησιμοποιούμε την

  ```
  gpio_set_callback(P_SW, freeze);
  ```

  η οποία με βάση το input που της δίνουμε καθορίζει τον handler που θα καλεί την συνάρτηση callback. Στην προκειμένη περίπτωση είναι ο

  ```
  EXTI15_10_IRQHandler(void);
  ```

- Τέλος, ορίζουμε ένα trigger, ώστε με το πάτημα του κουμπιού να εκτελείται απευθείας το interrupt. Συγκεκριμένα χρησιμοποιούμε την:

  ```
  gpio_set_trigger(P_SW, Falling);
  ```

  , οπού η επιλογή Falling χρησιμοποιείται, για να υποδείξει ότι θα εκτελείται όταν από λογικό υψηλό (1) καταλήξουμε σε λογικό χαμηλό (0). Αυτό είναι σωστό καθώς στην περίπτωσή μας, καθώς ο διακόπτης είναι PullUp και συνεπώς όταν πατιέται μεταφερόμαστε από λογικό υψηλό σε λογικό χαμηλό, δηλαδή ¨πέφτουμε¨.

Timer Interrupts

Χρησιμοποιούνται 2 timer interrupts για να υλοποιηθούν οι περιοδικές διαδικασίες:

1. Ο SysTick timer χρησιμοποιείται με περίοδο 0.5 δευτερόλεπτα για την ανάλυση κάθε χαρακτήρα. Οι συναρτήσεις χρήσης του ορίζονται στο αρχείο timer.h
2. Ο TIM2 χρησιμοποιείται για το blinking του LED. Με περίοδο 0.2 δευτερόλεπτα κάνει toggle το LED, εάν είναι ενεργοποιημένος. Οι συναρτήσεις ορίζονται στο αρχείο

Αποτελούν και οι δύο hardware timers που υπάρχουν στο board.
Στο σώμα του ISR του SysTick γίνεται η ανάλυση των ψηφείων.

Προτεραιότητες

Οι προτεραιότητες τίθονται έτσι ώστε button ¡ UART ¡ SysTick (character analysis timer) ¡ TIM2 (LED blinking timer) όπου με ˚ εννοούμε ότι έχει μικρότερη προτεραιότητα - γίνεται πρώτος σε περίπτωση σύγκρουσης. Η προτεραιότητες τίθνοντα με την NVIC, πχ η:

```
NVIC_SetPriority(TIM2_IRQn, 1);
```

θέτει τον TIM2 σε priority groop 1.

Η σειρά επιλέχθηκε ώστε αν πατηθεί το κουμπί, να μπορεί να παγώσει οποιαδήποτε διαδικασία πριν γίνουν αλλαγές, και αν δωθεί κάποιο γράμμα στην UART όσο γίνεται η ανάλυση, αυτή να σταματήσει απευθείας και να ξεκινήσει εκ νεόυ.

main

Στην main αρχικοποιούνται οι παραπάνω interrupts και έπειτα μπαίνουμε σε μια ατέρμονη λούπα όπου αν είμαστε σε στάδιο ανάγνωσης χαρακτήρων, λειτουργέι η uart όπως προαναφέρθηκε, αλλιώς περνάμε στο στάδιο της ανάλυσης όπου ενεργοποιούνται οι timers και λειτουργούν αναλύοντας έναν έναν τους χαρακτήρες εώς ότου διαβαστεί ολόκληρος ή έρθει από το πληκτρολόγιο καινούργιος χαρακτήρας, όποτε οι μεταβλητές που χρειάζεται επιστρέφουν στην αρχική τους τιμή και η διαδικασία εκτελείται από την αρχή.