Εργασία 2

Βογιατζής Χαρίσιος
AEM:9192

May 15, 2025

Github Source Code (Lab 2)

1 Εισαγωγή

Ο στόχος της δεύτερης εργαστηριακής άσκησης είναι η ανάπτυξη ενός συστήματος ανάλυσης ψηφίων για μικρο-ελεγκτές ARM. Το σύστημα λαμβάνει έναν αριθμό μέσω σειριακής επικοινωνίας (UART), τον αναλύει ψηφίο προς ψηφίο, και παρέχει οπτική ανάδραση μέσω ενός LED και UART logging. Η λειτουργικότητα επεκτείνεται με την ανίχνευση συνεχούς λειτουργίας και τη δυνατότητα παγώματος/επαναφοράς της κατάστασης του LED μέσω ενός κουμπιού, χωρίς να διακόπτεται η κύρια λογική ανάλυσης ή η σειριακή επικοινωνία.

Οι κύριες λειτουργίες περιλαμβάνουν:

- Λήψη αριθμητικής εισόδου μέσω UART.
- Διαδοχική ανάλυση κάθε ψηφίου του αριθμού με χρονικό διάστημα 0.5 δευτερολέπτων ανάμεσα στα ψηφία.
- Διαφορετική οπτική ένδειξη LED για ζυγά και μονά ψηφία:
  - Μονά ψηφία: Εναλλαγή της κατάστασης του LED (ON/OFF).
  - Ζυγά ψηφία: Αναβόσβημα του LED (200ms ON, 200ms OFF) για όσο διαρκεί η ανάλυση του ψηφίου.
- Υποστήριξη "συνεχούς λειτουργίας" (continuous mode) εάν η είσοδος τελειώνει με τον χαρακτήρα '-'.
- Διαχείριση πατήματος κουμπιού μέσω εξωτερικής διακοπής για το πάγωμα (lock) και την επαναφορά (unlock) της φυσικής κατάστασης του LED, ενώ η λογική κατάσταση και η σειριακή καταγραφή συνεχίζονται κανονικά.
- Συνεχής καταγραφή (logging) των ενεργειών και των καταστάσεων του συστήματος μέσω UART.

2 Υλοποίηση

2.1 Διακοπές (Interrupts)

Ο χρονοδιακόπτης SysTick χρησιμοποιείται για τη διαχείριση του χρονισμού των βασικών λειτουργιών. Συγκεκριμένα, ρυθμίζεται να παράγει διακοπές κάθε 1ms. Αυτές οι διακοπές αξιοποιούνται για την υλοποίηση δύο χρονιστών λογισμικού: ενός για την ανάλυση κάθε ψηφίου της ακολουθίας εισόδου (με περίοδο 500ms) και ενός για το αναβόσβημα του LED όταν ανιχνεύεται ζυγό ψηφίο (με κύκλο 200ms ON / 200ms OFF).

Η διακοπή λήψης UART (UART RX) ενεργοποιείται κάθε φορά που λαμβάνεται ένας νέος χαρακτήρας μέσω της σειριακής θύρας. Οι λαμβανόμενοι χαρακτήρες αποθηκεύονται σε μια προσωρινή ουρά (buffer) για επεξεργασία από την κύρια λογική του προγράμματος. Αυτός ο μηχανισμός επιτρέπει την ασύγχρονη λήψη της ακολουθίας ψηφίων από τον χρήστη.

Η εξωτερική διακοπή (EXTI) που συνδέεται με το κουμπί χρήστη χρησιμοποιείται για την εναλλαγή της κατάστασης παγώματος (freeze/unfreeze) του LED. Όταν το κουμπί πατηθεί, η αντίστοιχη ISR (Interrupt Service Routine) αλλάζει την κατάσταση μιας σημαίας (flag) που υποδεικνύει αν το LED πρέπει να παραμένει στη φυσική του κατάσταση ή να ακολουθεί τις εντολές της λογικής ανάλυσης ψηφίων. Η διακοπή αυτή έχει σχεδιαστεί ώστε να μην επηρεάζει την τρέχουσα ανάλυση ψηφίων, επιτρέποντας μόνο την οπτική παύση της ανάδρασης του LED.

2.2 Βασικές Δομές Δεδομένων

Για την οργάνωση της λογικής του συστήματος, χρησιμοποιούνται ορισμένες βασικές δομές δεδομένων και μεταβλητές κατάστασης:

- **AppState**: Μια απαρίθμηση (enum) που ορίζει τις κύριες καταστάσεις λειτουργίας του συστήματος. Οι καταστάσεις αυτές περιλαμβάνουν τυπικά:
  - `APP_STATE_INIT`: Αρχική κατάσταση κατά την εκκίνηση του συστήματος, όπου πραγματοποιούνται οι απαραίτητες αρχικοποιήσεις περιφερειακών και μεταβλητών.
  - `APP_STATE_IDLE`: Το σύστημα βρίσκεται σε αδράνεια, αναμένοντας νέα είσοδο (αριθμό) από τον χρήστη μέσω UART.
  - `APP_STATE_RECEIVING_INPUT`: Το σύστημα βρίσκεται σε διαδικασία λήψης χαρακτήρων από τη σειριακή θύρα UART για τη σύνθεση του αριθμού προς ανάλυση.
  - `APP_STATE_START_ANALYSIS`: Μετά την επιτυχή λήψη ενός αριθμού, το σύστημα εισέρχεται σε αυτή την κατάσταση για να προετοιμαστεί για την έναρξη της ανάλυσης των ψηφίων του.
  - `APP_STATE_ANALYZING_DIGIT`: Το σύστημα αναλύει ένα-ένα τα ψηφία του ληφθέντος αριθμού, εκτελώντας την αντίστοιχη λογική για το LED (αναβόσβημα για ζυγά, εναλλαγή για μονά).
  - `APP_STATE_CONTINUOUS_BLINK`: Ειδική κατάσταση που χρησιμοποιείται κατά το αναβόσβημα του LED για τα ζυγά ψηφία, διαχειριζόμενη τους κύκλους ON/OFF εντός του διαστήματος ανάλυσης του ψηφίου.

  Η μετάβαση μεταξύ των καταστάσεων καθοδηγείται από τις εισόδους του χρήστη (UART, κουμπί) και τους χρονιστές.
- **Ουρά Εισόδου UART (UART Input Queue)**: Μια κυκλική ουρά (circular buffer) χρησιμοποιείται για την προσωρινή αποθήκευση των χαρακτήρων που λαμβάνονται από τη σειριακή θύρα UART. Η ISR λήψης UART προσθέτει τους εισερχόμενους χαρακτήρες στην ουρά, ενώ η κύρια λογική του προγράμματος τους αφαιρεί και τους επεξεργάζεται. Αυτό αποσυνδέει την ασύγχρονη λήψη δεδομένων από την επεξεργασία τους.
- **Global Flags**:
  - `continuous_mode_active`: Υποδεικνύει εάν το σύστημα βρίσκεται σε συνεχή λειτουργία, επαναλαμβάνοντας την ανάλυση του τελευταίου αριθμού.
  - `led_current_state_on`: Αποθηκεύει την τρέχουσα λογική κατάσταση του LED (ON/OFF), ανεξάρτητα από το αν είναι παγωμένο.
  - `led_should_blink`: Σημαία που υποδεικνύει ότι το LED πρέπει να βρίσκεται σε κατάσταση αναβοσβήματος (για ζυγά ψηφία).
  - `led_frozen`: Υποδεικνύει εάν η φυσική κατάσταση του LED είναι "παγωμένη" από το πάτημα του κουμπιού. Όταν είναι `true`, το LED δεν αλλάζει την εμφάνισή του.
  - `uart_char_received_flag`: (volatile boolean) Τίθεται από την UART ISR όταν ένας νέος χαρακτήρας τοποθετείται στην ουρά λήψης, σηματοδοτώντας στον κύριο βρόχο ότι υπάρχουν δεδομένα για επεξεργασία.
  - `button_pressed_flag`: Τίθεται από την EXTI ISR όταν ανιχνευτεί πάτημα του κουμπιού, σηματοδοτώντας στην κύρια λογική να επεξεργαστεί την ενέργεια.
  - `new_input_interrupt_flag`: Τίθεται εάν ληφθεί νέα είσοδος UART ενώ βρίσκεται σε εξέλιξη η ανάλυση ενός αριθμού, επιτρέποντας τη διακοπή της τρέχουσας ανάλυσης.

2.3 Ροή Προγράμματος (Program Flow)

Η ροή του προγράμματος είναι βασισμένη σε έναν ατέρμονα βρόχο (main loop) που συντονίζει τις λειτουργίες του συστήματος, αντιδρώντας σε γεγονότα που σηματοδοτούνται από τις ρουτίνες εξυπηρέτησης διακοπών (ISRs) μέσω καθολικών σημαιών.

1. **Αρχικοποίηση**: Κατά την εκκίνηση, το σύστημα αρχικοποιεί τα περιφερειακά (GPIOs για LED και κουμπί, UART, SysTick timer, EXTI για το κουμπί) και τις καθολικές μεταβλητές, συμπεριλαμβανομένης της ουράς UART και των σημαιών. Η αρχική κατάσταση του συστήματος ορίζεται σε `APP_STATE_IDLE`.

2. **Κύριος Βρόχος (Main Loop)**: Ο κύριος βρόχος εκτελείται συνεχώς και είναι υπεύθυνος για:

   - **Έλεγχος Σημαιών Διακοπών**: Περιοδικά ελέγχει τις σημαίες που τίθενται από τις ISRs.
   - **Επεξεργασία Εισόδου UART**: Όταν η σημαία `uart_char_received_flag` είναι ενεργή:
     (a) Η ακολουθία χαρακτήρων διαβάζεται από την ουρά UART.
     (b) Γίνεται έλεγχος για τον ειδικό χαρακτήρα '-' στο τέλος της ακολουθίας για την ενεργοποίηση της συνεχούς λειτουργίας (`continuous_mode_active`).
     (c) Τα ψηφία εξάγονται και αποθηκεύονται για ανάλυση.
     (d) Το σύστημα μεταβαίνει στην κατάσταση `APP_STATE_ANALYZING_DIGIT` για να ξεκινήσει την επεξεργασία του πρώτου ψηφίου.
     (e) Η σημαία `uart_char_received_flag` καθαρίζεται.
   - **Διαχείριση Πατήματος Κουμπιού**: Όταν η σημαία `button_pressed_flag` είναι ενεργή:
     (a) Η κατάσταση της σημαίας `led_frozen` αντιστρέφεται.
     (b) Καταγράφεται μέσω UART το πάτημα του κουμπιού και η νέα κατάσταση παγώματος του LED.
     (c) Η φυσική κατάσταση του LED ενημερώνεται άμεσα: αν το `led_frozen` είναι `true`, το LED παγώνει στην τρέχουσα οπτική του κατάσταση. Αν είναι `false`, το LED παίρνει την τιμή της `led_current_state_on`.
     (d) Η σημαία `button_pressed_flag` καθαρίζεται.
   - **State Machine**: Η συμπεριφορά του συστήματος εξαρτάται από την τρέχουσα `AppState`:
     - `APP_STATE_IDLE`: Το σύστημα αναμένει νέα είσοδο.
     - `APP_STATE_ANALYZING_DIGIT`:
       * Ένα ψηφίο κάθε φορά αναλύεται. Ο χρονιστής SysTick (μέσω λογισμικού χρονιστή) καθορίζει το διάστημα των 500ms για την ανάλυση κάθε ψηφίου.
       * Αν το ψηφίο είναι ζυγό: το LED τίθεται σε κατάσταση αναβοσβήματος (`APP_STATE_LED_BLINKING`) για 200ms ON / 200ms OFF. Η λογική κατάσταση `led_current_state_on` παραμένει αμετάβλητη.
       * Αν το ψηφίο είναι μονό: η λογική κατάσταση του LED (`led_current_state_on`) αντιστρέφεται.
       * Η φυσική κατάσταση του LED ενημερώνεται (`leds_set()`) μόνο αν `led_frozen` είναι `false`.
       * Το αποτέλεσμα της ανάλυσης καταγράφεται μέσω UART.
       * Μετά την πάροδο των 500ms, το σύστημα προχωρά στο επόμενο ψηφίο ή, αν δεν υπάρχουν άλλα, μεταβαίνει σε `APP_STATE_IDLE` (ή `APP_STATE_CONTINUOUS_WAIT` αν είναι ενεργοποιημένη η συνεχής λειτουργία).
     - `APP_STATE_LED_BLINKING`: Ο λογισμικός χρονιστής που οδηγείται από το SysTick διαχειρίζεται το αναβόσβημα του LED. Μετά την ολοκλήρωση της περιόδου αναβοσβήματος για το τρέχον ζυγό ψηφίο (που συμπίπτει με τα 500ms της ανάλυσης του ψηφίου), το σύστημα επιστρέφει για να επεξεργαστεί το επόμενο ψηφίο ή να ολοκληρώσει την ανάλυση.
     - `APP_STATE_CONTINUOUS_WAIT`: Αν η συνεχής λειτουργία είναι ενεργή, μετά την ανάλυση όλων των ψηφίων, το σύστημα εισέρχεται σε μια σύντομη αναμονή (π.χ. 500ms) πριν ξεκινήσει εκ νέου την ανάλυση της ίδιας ακολουθίας ψηφίων.

3. **Αλληλεπίδραση ISR - Main Loop**:

   - Η **UART RX ISR** είναι υπεύθυνη για την ασύγχρονη λήψη χαρακτήρων και την τοποθέτησή τους στην ουρά εισόδου. Όταν ανιχνεύσει το τέλος μιας εισόδου (π.χ. χαρακτήρας newline), θέτει τη σημαία `uart_char_received_flag`.
   - Η **Button EXTI ISR** ανιχνεύει το πάτημα του κουμπιού, αυξάνει έναν μετρητή πατημάτων (προαιρετικά, για logging) και θέτει τη σημαία `button_pressed_flag`. Η ISR είναι σχεδιασμένη να είναι σύντομη για να μην καθυστερεί άλλες διακοπές.
   - Η **SysTick ISR** εκτελείται περιοδικά (π.χ. κάθε 1ms) και ενημερώνει τους μετρητές των λογισμικών χρονιστών που χρησιμοποιούνται από την κύρια λογική για τον χρονισμό της ανάλυσης ψηφίων και του αναβοσβήματος του LED. Δεν θέτει άμεσα σημαίες για τον κύριο βρόχο, αλλά ο κύριος βρόχος ελέγχει τις τιμές αυτών των χρονιστών.

3 Προκλήσεις Λύσεις

Κατά την ανάπτυξη του συστήματος αντιμετωπίστηκαν ορισμένες προκλήσεις, οι οποίες επιλύθηκαν με συγκεκριμένες τεχνικές και προσαρμογές στον κώδικα:

- **Timer Initialization Crash**:
  - **Πρόκληση**: Το πρόγραμμα παρουσίαζε crash κατά την αρχικοποίηση του SysTick.
  - **Λύση**: Η ανάλυση έδειξε ότι υπήρχε ένα race condition. Ο timer ξεκινούσε πριν η callback function οριστεί πλήρως, οδηγώντας σε NULL pointer dereference κατά το πρώτο interrupt του timer, παρ'όλο που δεν είχε γίνει enabled.
- **Button Interrupts**:
  - **Πρόκληση**: Η ανίχνευση του πατήματος του κουμπιού, που ήταν αρχικά ρυθμισμένη για πτώση ακμής (falling edge), ήταν αναξιόπιστη. Η ISR του EXTI περιέχει έναν έλεγχο της τρέχουσας κατάστασης του pin (`if(p->IDR (1<<IRQ_pin_index))`), ο οποίος αποτύγχανε επειδή τη στιγμή του ελέγχου (μετά την πτώση της ακμής), το pin ήταν ήδη σε κατάσταση LOW.
  - **Λύση**:
    1. **Άμεση Αντιμετώπιση**: Η διακοπή ρυθμίστηκε να ενεργοποιείται με ανύψωση ακμής (rising edge), δεδομένου ότι το κουμπί συνδέεται με pull-up αντίσταση, οπότε το πάτημα δημιουργεί πτώση ακμής και το άφημα ανύψωση. Έτσι, η ενέργεια (πάγωμα/ξεπάγωμα LED) εκτελείται κατά το άφημα του κουμπιού. Αυτό παρέκαμψε το πρόβλημα του ελέγχου κατάστασης εντός της ISR.

4 Testing

Για δοκιμή της ορθής λειτουργίας δοκιμάστηκαν διάφορες ακολουθίες εισόδου αριθμών, συμπεριλαμβανομένων κενών εισόδων, αριθμών με ζυγά και μονά ψηφία, την ενεργοποίηση και απενεργοποίηση της συνεχούς λειτουργίας, καθώς και την αλληλεπίδραση με το πάγωμα του LED μέσω του κουμπιού κατά τη διάρκεια της ανάλυσης.

5 Source Code

Github: https://github.com/charisvt/micro-lab2