Εργαστήριο 2 - Αναφορά

Λαμπρινός Χατζηιωάννου, Γιώργος Σελιβάνωφ

May 27, 2024

Contents

Για την υλοποίηση της 2ης εργασίας, αποφασίστηκε να μην χρησιμοποιηθούν `uart_prints` μέσα στα interrupts (ή στα callbacks αυτών), ώστε να διατηρηθεί το execution time τους στο ελάχιστο. Αντ'αυτού όλες οι ενέργειες που απαιτούν `uart_print` εκτελούνται στην `main` εκμεταλλευόμενοι `queues`.

1 AEM μέσω UART

Για την ανάγνωση του AEM μέσω UART χρησιμοποιήθηκαν οι συναρτήσεις που δόθηκαν στα πλαίσια του εργαστηρίου (`uart.c / uart.h`). Αντί η `main` να αναμένει ολόκληρο το input του χρήστη, πλέον χειρίζεται τον κάθε χαρακτήρα ξεχωριστά, ώστε να μπορεί να εκτελέσει και άλλες ενέργειες σε κάθε επανάληψη του loop (π.χ. Εκτύπωση της αλλαγής της κατάστασης του LED). Ως εκ τούτου, το enter αναγνωρίζεται ομοίως με το backspace, εκτελώντας μια ξεχωριστή συνάρτηση κατά τον εντοπισμό του (`checkAEM()`).

```
while(1)
    if (!( checkUart(&buff_index,buff) || checkMessg() ))
      __WFI();
```

2 Συμπεριφορά LED ανάλογα με το AEM

Για το blink του LED ορίστηκε ένας timer με την χρήση των `timer.c` και `timer.h` που δίνονται στα πλαίσια του εργαστηρίου, ο οποίος απλώς κάνει toggle το `GPIO PA_5` (User LED GPIO

των SMT32 board). Κατά τον έλεγχο του AEM με την `checkAEM()`, ο συγκεκριμένος timer ενεργοποιείται ή απενεργοποιείται αναλόγως με τον εάν το AEM είναι μονός ή ζυγός αριθμός αντίστοιχα.

Να σημειωθεί ότι παρότι στην εκφώνηση αναφέρεται πως τα παραπάνω πρέπει να γίνουν με την χρήση `ISR`, δεν βρήκαμε πως η χρήση ISR θα ήταν εφικτή/ενδεδειγμένη στο συγκεκριμένο στάδιο. Το input από τον χρήστη γίνεται στο στάδιο της UART, στο οποίο υπάρχει ήδη `ISR` μέσα στην οποία δεν θα ήταν σκόπιμο να ενσωματωθεί όλος αυτός ο έλεγχος.

3 Πάτημα του διακόπτη

Τα ζητούμενα e) και f) ουσιαστικά ζητάνε το πάτημα του διακόπτη να κάνει toggle το LED και να εκτυπώνει τον coutner των πατημάτων. Χρησιμοποιώντας της συναρτήσεις που δίνονται στα `gpio.c` και `gpio.h`, ορίσαμε μια **interrupt callback** η οποία κάνει toggle το LED και προσθέτει το κατάλληλο μήνυμα στη message queue, ώστε να εκτυπωθεί ο counter από την main.

4 Message Queue

Για την εκτύπωση μηνυμάτων μέσω $\text{uart}_{\text{print}}$ από τις διάφορες ISR, έγινε χρήση ενός message queue. Οι `ISR` προσθέτουν integers στο queue, οι οποίοι αντιστοιχίζονται στις αντίστοιχες ενέργειες:

1. Εκτύπωση μηνύματος για απενεργοποίηση του LED
2. Εκτύπωση μηνύματος για ενεργοποίηση του LED
3. Αύξηση κατά 1 και εκτύπωση του counter

Με αυτόν τον τρόπο, δεν επιβαρύνονται τα interrupts από την $\text{uart}_{\text{print}}$, διατηρώντας τον χρόνο εκτέλεσής τους χαμηλό.

```
if (queue_dequeue(&msg_queue, &lastState)) {
  switch (lastState) {
  case 0:
    sprintf(buffer, "Led is off.\r\n"); break;
  case 1:
    sprintf(buffer, "Led is on.\r\n"); break;
  case 2:
    sprintf(buffer, "Keypresses pressed %d .\r\n", ++switchPresses); break;
  }
```