Αναφορά 3ης Άσκησης

Ονοματεπώνυμο: Ιωάννης Λαμπρινίδης

Στόχος της τρίτης εργαστηριακής άσκησης ήταν η εξοικείωση με τη σειριακή θύρα RS-232. Σκοπός της άσκησης ήταν η ενσωμάτωση της λειτουργικότητας της θύρας στη 7-segment LED οθόνη που είχε υλοποιήσει για το 2ο εργαστήριο.

Αρχικά, έγιναν οι απαραίτητες αλλαγές στον κώδικα του δεύτερου εργαστηρίου για να ικανοποιούνται τα προαπαιτούμενα:

- Αλλάξαμε την θέση αποθήκευσης δεδομένων από τη flash στην RAM και η αποκωδικοποίηση παρέμεινε στη flash.
- Προστέθηκε ο χαρακτήρας 0x0A για την αρχικοποίηση της μνήμης.
- Οι αρχικοποιήσεις τοποθετήθηκαν σε ξεχωριστές ρουτίνες για την καλύτερη ανάγνωση και οργάνωση του προγράμματος. Στο interrupt του timer δεν έγινε κάποια αλλαγή.

Στη συνέχεια βρέθηκε η τιμή του καταχωρητή ubrr έτσι ώστε να έχουμε ταχύτητα 9600 baud με ρολόι 10MHz. $UBRR = \frac{Fosc}{16 * BAUD} - 1$.(Χρησιμοποιήθηκε ο πρώτος τύπος από τον πίνακα 60 της σελίδας 147 του manual για ασύγχρονη λειτουργία.)

Το simulator εμφανίζει προβλήματα σχετικά με τη θύρα usart και τα stimuli file για αυτό το λόγω για την ανάγνωση δεδομένων χρησιμοποιήθηκε ο καταχωρητής r15 και για τη διάδοση δεδομένων ο καταχωρητής του timer2. Ο κώδικας που θα έπρεπε να υπάρχει για την ορθή χρήση της θύρας βρίσκεται σε σχόλια.

Ρουτίνες που δημιουργήθηκαν:

- Initialise _ : Οι ρουτίνες με αυτό το αρχικό είναι οι αρχικοποιήσεις της συγκεκριμένης λειτουργίας και δεν έχουν ιδιαίτερη σημασία εκτός από τη ρουτίνα InitialiseRT στην οποία δεν ενεργοποιείται το interrupt enable USART_UDRE διότι όπως αναφέρθηκε λόγω δυσλειτουργίας του simulator ο καταχωρητής δεν μπορεί να γραφτεί άρα το interrupt ενεργοποιείται συνέχεια.

- ClearScreen: Η ρουτίνα αρχικοποιεί τις θέσης της μνήμης που αποθηκεύεται η BCD απεικόνιση με τον χαρακτήρα 0x0A ο οποίος μεταφράζεται στον 0xFF ο οποίος αρχικοποιεί τα 7-segment LED.

- MoveMemory: Η ρουτίνα μεταφέρει κάθε αποθηκευμένο byte στην επόμενη θέση μνήμης που σχετίζεται με τον αριθμό προς εμφάνιση. Η υλοποίηση της ρουτίνας εμφανίζει το εξής θέμα ότι για n bytes χρειάζομαι n + 1 θέσεις μνήμης γιατί η μικρότερη θέση μνήμης γίνεται duplicate. Δεν έγινε κάποια προσπάθεια διόρθωσης του κώδικα γιατί η συγκεκριμένη υλοποίηση έχει το πλεονέκτημα ότι δεν θέλει πολλούς κύκλους για κάθε θέση μνήμης αλλά ούτε και πολλούς ελέγχους.

- Usart_UDRE: Η ρουτίνα αυτή ενεργοποιείται από το αντίστοιχο interrupt για να στείλει το μήνυμα ok. Το polling για το πότε θα μεταδώσει byte η ρουτίνα γίνεται στον ατέρμονο βρόγχο της main(Επειδή το ok στέλνεται σε κάθε περίπτωση και θεωρούμε πως δεν υπάρχει σφάλμα κατά τη μετάδοση των δεδομένων δεν υπάρχει κάποιος έλεγχος για το σήμα).

- USART_RXC: Η συγκεκριμένη ρουτίνα διαβάζει ένα byte από τη θύρα και στη συνέχεια παραχωρεί τον έλεγχο στη ρουτίνα Control για την επεξεργασία των δεδομένων και μετά ενεργοποιεί το interrupt UDRE για να σταλθεί το ok signal.

- Control: Η ρουτίνα δέχεται το byte που διαβάστηκε στον καταχωρητή r18 και στη συνέχεια ανάλογα τον ascii χαρακτήρα έχουμε τις εξής λειτουργικότητες:
  1)Άμα έρθει το N ή το C θα καλέσουμε τη ρουτίνα ClearScreen.
  2)Άμα έρθει ο χαρακτήρας 0x0a τότε θα ενεργοποιηθεί το interrupt ρουτίνας Usart_UDRE.
  3)Άμα έρθει ascii νούμερο τότε θα καλεστεί η συνάρτηση SaveToMemory.

- SaveToMemory: Η ρουτίνα δέχεται ως όρισμα ένα νούμερο σε κωδικοποίηση ascii, βρίσκει πιο νούμερο είναι και στη συνέχεια το αποθηκεύει στη μνήμη.

- EnableTrasmitCall: Η ρουτίνα αρχικοποιεί τον register X στη θέση του πρώτου αριθμού προς εμφάνιση και καλεί τη ρουτίνα EnableTransmit.

- EnableTransmit: Ενεργοποιεί το interrupt enable USART_UDRE.

- DisableTransmit: Απενεργοποιεί το interrupt enable USART_UDRE.

Στο παρακάτω διάγραμμα βλέπουμε τη ροή λειτουργίας του προγράμματος.