Εργαστήριο 3 - Ομάδα 3

Γαλανομάτης Σωκράτης - Τσακιρίδου Δήμητρα Μαρία

- Υλοποίηση κώδικα:

Κώδικας για τον αισθητήρα θερμοκρασίας/υγρασίας:

Μετά την παροχή ρεύματος, δεν πρέπει να σταλούν εντολές στον αισθητήρα για τουλάχιστον 1 δευτερόλεπτο για να σταθεροποιηθεί, οπότε η DHT11_Init θέτει το pin σε κατάσταση εισόδου με pull-up resistor και κάνει delay 1 δευτερόλεπτο πριν ξεκινήσει οποιαδήποτε επικοινωνία με τον αισθητήρα.

Με βάση το datasheet (MCU Sends out Start Signal), δηλαδή ο μικροελεγκτής πρέπει να στείλει σήμα εκκίνησης στον αισθητήρα, το οποίο περιλαμβάνει να θέσει το pin σε χαμηλή κατάσταση για τουλάχιστον 18 ms. Μετά από αυτό, (DHT Responses to MCU) πρέπει να θέσει το pin σε υψηλή κατάσταση και να περιμένει για 20-40 μs για να λάβει την απάντηση του αισθητήρα. Επομένως, η DHT11_Start θέτει το pin σε έξοδο και ακολουθεί αυτά τα βήματα και τέλος θέτει το pin σε κατάσταση εισόδου για να μπορεί να λάβει την απάντηση από τον αισθητήρα.

Κάθε bit ξεκινάει με ένα χαμηλό παλμό διάρκειας 50 μs. Αν ο επόμενος υψηλός παλμός διαρκεί περίπου 26-28 μs, το bit είναι '0', ενώ αν διαρκεί περίπου 70 μs, το bit είναι '1'. Έτσι, η DHT11_ReadByte περιμένει μέχρι να ανιχνευθεί το τέλος αυτού του χαμηλού παλμού, δηλαδή μέχρι το σήμα να γίνει υψηλό, μετά την καθυστέρηση των 40μs ο υψηλός παλμός θα είναι ακόμα σε εξέλιξη αν το bit είναι '1' (διάρκεια περίπου 70 μs), ενώ αν είναι '0', θα έχει ήδη τελειώσει (διάρκεια περίπου 26-28 μs). Τέλος, αν είναι 1 διαβάζει ένα bit από τον αισθητήρα, το αποθηκεύει στη σωστή θέση στο byte και περιμένει μέχρι το τέλος του υψηλού παλμού.
*(1 << (7 - i)): δημιουργεί ένα bit στη θέση (7 - i), και το bitwise OR |= το προσθέτει στο byte.

Η συνολική μετάδοση δεδομένων είναι 40 bit (5 byte), τα οποία λαμβάνονται με αυτήν τη σειρά: 8 bit ακέραια τιμή υγρασίας, 8 bit δεκαδική τιμή υγρασίας, 8 bit ακέραια τιμή θερμοκρασίας, 8 bit δεκαδική τιμή θερμοκρασίας, 8 bit checksum (πρέπει να είναι ίσο με το άθροισμα των πρώτων τεσσάρων byte). Η συνάρτηση DHT11_ReadTemperature διαβάζει τα 5 byte, ελέγχει αν το checksum είναι σωστό και αν ναι, αποθηκεύει τις τιμές θερμοκρασίας και υγρασίας στις κατάλληλες θέσεις και επιστρέφει 1.

Στον κώδικα της main():
queue_init(&rx_queue, 128): Αρχικοποιεί την ουρά για τη λήψη δεδομένων μέσω UART.
uart_init(115200): Αρχικοποιεί την επικοινωνία UART με ταχύτητα 115200 baud.
uart_set_rx_callback(uart_rx_callback): Καθορίζει τη συνάρτηση επιστροφής κλήσης για τη λήψη δεδομένων μέσω UART.
uart_enable(): Ενεργοποιεί την επικοινωνία UART.
gpio_set_mode(ext_led_pin, Output): Θέτει τον εξωτερικό LED σε κατάσταση εξόδου.
Ο ΑΕΜ κωδικός ελέγχεται και αποθηκεύεται στην μεταβλητή user_id, με τρόπο παρόμοιο με του project_2

Αρχικοποίηση των περιφερειακών:
DHT11_Init(): Αρχικοποιεί τον αισθητήρα θερμοκρασίας και υγρασίας DHT11.
TouchSensor_Init(): Αρχικοποιεί τον αισθητήρα αφής.
gpio_set_callback(PA_1, touch_callback): Θέτει τη συνάρτηση επιστροφής κλήσης για τον αισθητήρα αφής.
gpio_set(ext_led_pin, LED_OFF): Αρχικοποιεί τον εξωτερικό LED σε κατάσταση OFF.

Αρχικοποίηση του timer:
timer_init(1000000): Αρχικοποιεί τον χρονοδιακόπτη με περίοδο 1 δευτερολέπτου.
timer_set_callback(timer_callback): Θέτει τη συνάρτηση επιστροφής κλήσης για τον χρονοδιακόπτη.
timer_enable(): Ενεργοποιεί τον χρονοδιακόπτη.

Ενεργοποίηση interrupts:
__enable_irq(): Ενεργοποιεί τα interrupts

Κυρίως λούπα:
Αν η σημαία read_sensor_flag είναι ενεργοποιημένη, διαβάζονται οι τιμές θερμοκρασίας και υγρασίας από τον αισθητήρα.
Αν η ανάγνωση είναι επιτυχής, οι τιμές εκτυπώνονται μέσω UART.
Αν το touch_press_count είναι άρτιος αριθμός, εκτυπώνεται και η υγρασία.

Αν η σημαία touch_event_flag είναι ενεργοποιημένη, εκτυπώνεται ένα μήνυμα που δείχνει τον αριθμό των πιέσεων του αισθητήρα αφής.
Αν είναι η πρώτη πίεση, αλλάζει η περίοδος εκτύπωσης θερμοκρασίας σύμφωνα με τα τελευταία δύο ψηφία του user_id, αλλιώς αν η πίεση είναι περιττός αριθμός, η περίοδος εκτύπωσης ρυθμίζεται σε 3 δευτερόλεπτα.

Η void uart_rx_callback(uint8_t rx), όπως και στο project_2, αποθηκεύει τους ληφθέντες χαρακτήρες στην ουρά rx_queue.

Η void timer_callback(void) αυξάνει τον μετρητή timer_counter και ελέγχει την κατάσταση του LED.
Ο timer έχει περίοδο 1sec οπότε κάθε τόσο καλεί τη συνάρτηση επιστροφής timer_callback, η οποία αυξάνει το timer_counter κατά ένα. Όταν το timer_counter φτάσει την τιμή της temp_print_period, η σημαία **read_sensor_flag** ενεργοποιείται για ανάγνωση του αισθητήρα. Μετά την εκτέλεση των αντίστοιχων λειτουργιών, το timer_counter επαναφέρεται σε μηδέν, ξεκινώντας ξανά τον μετρητή για την επόμενη περίοδο.

Η void TouchSensor_Init(void) θέτει το pin του αισθητήρα αφής σε κατάσταση εισόδου με pull-down και ορίζεται το trigger στο rising edge. Δηλαδή, θα είναι σε λογικό 0 όταν δεν υπάρχει σήμα, έτσι ώστε να ανιχνεύεται η αλλαγή από λογικό 0 σε λογικό 1.

Η void touch_callback(void) αυξάνει τον μετρητή πιέσεων touch_press_count και ενεργοποιεί τη σημαία **touch_event_flag** για όταν πατηθεί ο διακόπτης.

Η get_last_two_digits_sum(int id) υπολογίζει και επιστρέφει το άθροισμα των δύο τελευταίων ψηφίων του ΑΕΜ.

Στη void handle_led_logic(void) αν η θερμοκρασία υπερβεί το TEMP_THRESHOLD_HIGH, ανάβει το LED, αν η θερμοκρασία είναι κάτω από το TEMP_THRESHOLD_LOW, σβήνει το LED και εάν η θερμοκρασία είναι μεταξύ των δύο τιμών, αναβοσβήνει το LED. Το LED το χειριζόμαστε με την timer_callback(void), καθώς έχει περίοδο 1s και μας βολεύει.

- Προβλήματα που αντιμετωπίσαμε:

Στο συγκεκριμένο πρότζεκτ κυρίως μας απασχόλησε περισσότερο το να κατανοήσουμε πώς από τα datasheets των αισθητήρων θα γράψουμε δικούς μας drivers συνδιάζοντας και εκείνους που μας δίνονται έτοιμοι και να επιλέξουμε ποιες συναρτήσεις θα αξιοποιήσουμε και από ποιους ("timer.h", "gpio.h", "delay.h", "queue.h"). Πρακτικά προβλήματα που αντιμετωπίσαμε είναι ότι καμιά φορά μετά το debug, κολλούσε όταν πήγαινε να διαβάσει από τον αισθητήρα dht11, καθώς είχαμε αρχικά ξεχωριστές συναρτήσεις για θερμοκρασία και υγρασία, αλλά λύθηκε όταν κάναμε μία που αποθηκεύει και τα δύο bytes. Ένα ακόμη πρόβλημα το οποίο λύθηκε όμως πολύ γρήγορα, αφορά την εμφάνιση των εκτυπώσεων μέσω UART, καθώς στην αρχή δεν είχαμε συμπεριλάβει καλά τα /r και /n μία μόνο γραμμή. Επίσης, για να χειριστούμε την κατάσταση του LED, στην αρχή δοκιμάσαμε κάποια delay, αλλά με testing στο εργαστήριο καταλήξαμε πως το να την βάλουμε μέσα στην callback του timer είναι η καλύτερη επιλογή. Τέλος, αρκετό χρόνο μας πήρε να μεταφέρουμε τις περισσότερες ενέργειες που θα κάναμε στις ISR στον κώδικα της main() για να μην κολλάει ο timer.

- Testing:

Το testing, δεν μπορούσε να γίνει με την χρήση του simulator, παρά μόνο όταν ήμασταν στο εργαστήριο έχοντας την πλακέτα και τους κατάλληλους αισθητήρες. Έτσι, αφού κάναμε build τον κώδικα, ξεκινούσαμε το debug session όπου για να ελέγξουμε αν η υλοποίηση μας είναι σωστή, βλέπαμε πως μεταβάλλονταν οι τιμές των registers στον πίνακα αριστερά, ποιες τιμές αποθηκεύονταν στις μεταβλητές μας και πότε εκτυπώνονται, στην αρχή με printf και έπειτα με UART. Για να ελέγξουμε όσες περιπτώσεις μπορούμε, στο debugging τρέχαμε τον κώδικα με διάφορα breakpoints, ώστε να δούμε σε ποια σημεία κολλάει και σε ποια δεν μπαίνει καθόλου.