ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

Τμημα ΗΜΜΥ. Τομεας Ηλεκτρονικης

Μικροεπεξεργαστες και Περιφερειακα

3η Εργασία

Συντάκτης :
Χρηστος Χουτουριδης
ΑΕΜ:8997
cchoutou@ece.auth.gr

Διδάσκον :
Παπαευσταθιου Ιωαννης
ygp@ece.auth.gr

28 Ιουνίου 2020

1. Εισαγωγη

Στην παρούσα εργασία το ζητούμενο ήταν η υλοποίηση ενός "έξυπνου θερμοστάτη" χρησιμοποιώντας αναπτυξιακό Nucleo-F401RE της εταιρίας ST Microelectronics. Η τελική εφαρμογή θα πρέπει να έχει οθόνη, leds αισθητήρα θερμοκρασίας και αισθητήρα εγγύτητας, υλικά που δεν είναι διαθέσιμα από το αναπτυξιακό. Για το σκοπό αυτό αποφασίσαμε να προχωρήσουμε την εργασία λιγάκι παραπάνω και να κατασκευάσουμε ένα shield για το Nucleo, το οποίο ενσωματώνει σε ένα τυπωμένο κύκλωμα όλα τα απαραίτητα στοιχεία. Τέλος, λόγο του ότι η διεπαφή χρήστη της εκφώνησης ήταν λιγάκι "φτωχή", πήραμε την πρωτοβουλία και υλοποιήσαμε ένα μικρό serial interface το οποίο μπορεί να χρησιμοποιηθεί και για να ρυθμίσει τον θερμοστάτη. Η ανάπτυξη έγινε σε γλώσσα C και χρησιμοποιήθηκε το εργαλείο Keil uVision. Ο κώδικας της εργασίας, της παρούσας αναφοράς, αλλά και τα σχέδια της κατασκευής υπάρχουν και στον προσωπικό server του συντάκτη, στο αποθετήριο για την εργασία.

2. Παραδοτεα

Στο παραδοτέο .zip αρχείο μπορείτε να βρείτε:

- Τον κατάλογο **/src** που περιέχει τον κώδικα της εφαρμογής.
- Τον κατάλογο **/Libraries** που περιέχει τον κώδικα των βιβλιοθηκών που χρησιμοποιήσαμε, όπως το CMSIS και το ST HAL, αλλά και τους drivers για τα περιφερειακά.
- Το αρχείο **report.pdf** που είναι η παρούσα αναφορά.
- Τον κατάλογο **/Keil** που περιέχει το project που χρησιμοποιήθηκε στο Keil.
  Στο project αυτό περιέχονται επιπλέων οι ρυθμίσεις καθώς και τα αρχεία από τις βιβλιοθήκες που χρησιμοποιήθηκαν.

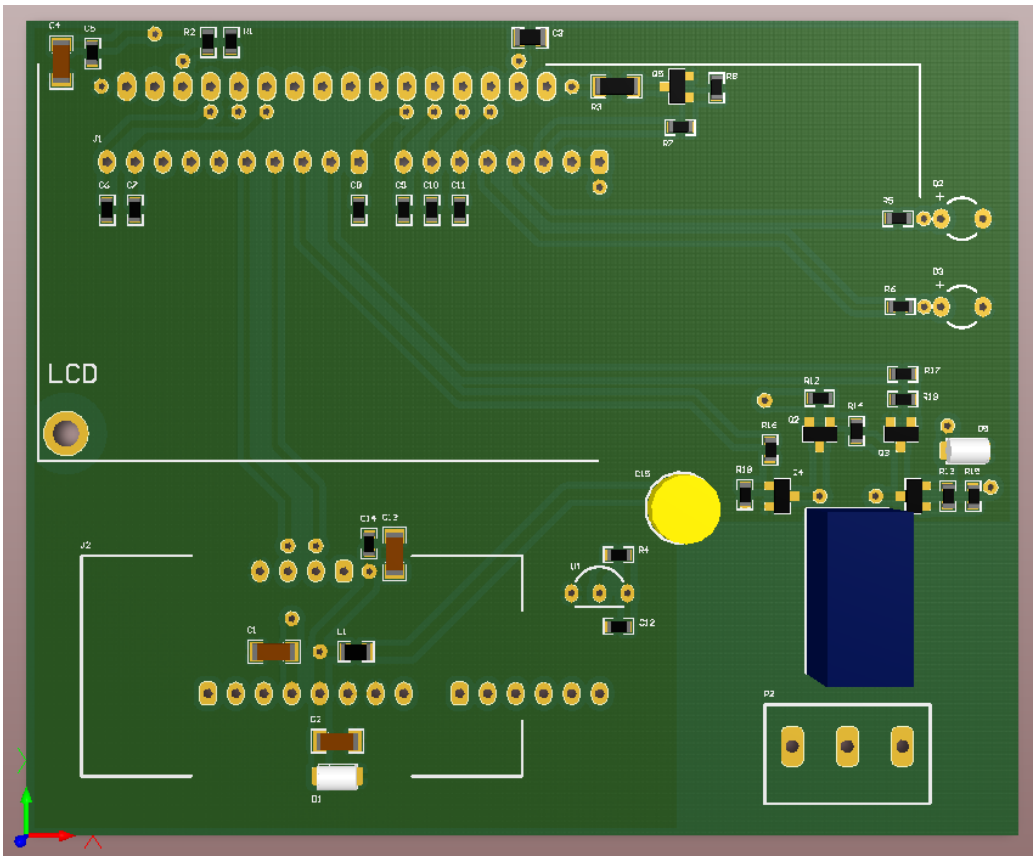
Για την παρούσα εργασία χρησιμοποιήσαμε το CMSIS και το STM32F4xx_HAL που παρέχει η εταιρία ST στο επίσημο αποθετήριό της. Η επιλογή αυτού έναντι των αρχείων από το STM32F4xx Std_Peripheral που περιεχόταν στο e-learning του μαθήματος έγινε για να συμβαδίσουμε με τις βιβλιοθήκες που προτείνει η κατασκευάστρια εταιρία. Επίσης η χρήση τους έγινε στο ίδιο χαμηλό επίπεδο της STM32F4xx_Std_Peripheral κάτι το οποίο προτείνεται και από τον διδάσκοντα. ***Εξάλλου μια υψηλότερου επιπέδου χρήση δεν θα είχε νόημα καθώς θα ξέφευγε από τον μαθησιακό χαρακτήρα της εργασίας***.

3. Υλοποιηση

3.1. Κατασκευή του shiled

Η υλοποίηση της εφαρμογής έγινε σε δύο φάσεις. Η πρώτη ήταν η ανάπτυξη και κατασκευή του shiled. Αυτό το κομμάτι δεν ήταν στα απαιτούμενα και γιαυτό δεν θα ασχοληθούμε πολύ παρουσιάζοντάς το. Θα πρέπει όμως να αναφέρουμε κάποια βασικά στοιχεία. Έτσι εν συντομία η κατασκευή απαρτίζεται από:

- Μια **οθόνη** υγρών κρυστάλλων 2 x 16, σε 4bit παράλληλη επικοινωνία με τον επεξεργαστή.
  Εκτός από την επικοινωνία ο επεξεργαστής ελέγχει και το back light μέσω ενός μικρού MOSFET, του MMBF170. Η οθόνη είναι σε μόνιμο write mode, μιας και ο ακροδέκτης RW είναι συνδεδεμένος μόνιμα στην γείωση.
- Ένα **relay** τύπου latching, το **G2SU-2** συνδεδεμένο μέσω H-γέφυρας.
  Η επιλογή latching relay έγινε με γνώμονα την πιθανή κατανάλωση της κατασκευής σε περίπτωση μελλοντικής χρήσης της. Σε μια τέτοια περίπτωση ενώ τα υπόλοιπα υλικά της κατασκευής λίγο πολύ θα μπορούσαν να περιορίσουν την κατανάλωση τους, το relay θα έπρεπε να καταναλώνει συνέχεια ρεύμα για να παραμένει οπλισμένο.
- Ένα αισθητήριο θερμοκρασίας τύπου **DS18B20**.
  Το αισθητήριο αυτό κάνει χρήση του πρωτοκόλλου επικοινωνίας 1-wire της Dallas semiconductors. Παρόλα αυτά στην κατασκευή είναι συνδεδεμένο στη σειριακή επικοινωνία του shield, επιτρέποντας έτσι τη χρήση του application note 214 της maxim, όπου περιγράφεται ένας τρόπος δημιουργίας του χρονισμού του 1-wire bus χρησιμοποιώντας τη σειριακή θύρα. Με τον τρόπο



Σχήμα 1: *3D απεικόνιση του PCB.*

αυτό ο επεξεργαστής αποδεσμεύεται από τον φόρτο να δημιουργεί ακριβείς χρονοκαθυστερήσεις, πιθανότατα απενεργοποιώντας τα interrupts και αφήνει ένα περιφερειακό του να εκτελέσει αυτή τη δουλειά.

- Ένα αισθητήριο εγγύτητας **HC-SR04** που λειτουργεί με υπερήχους.
- Διάφορα "**μικρούτσικα**" υλικά, όπως τρανζιστορ-άκια, αντιστάσεις, πυκνωτές, leds κ.α.

3.2. Οδηγός του Nucleo και του shiled

Η δεύτερη φάση της ανάπτυξης ήταν ο προγραμματισμός. Ο επεξεργαστής που φέρει το εν λόγο αναπτυξιακό είναι πολύ μεγαλύτερος από τις ανάγκες της παρούσας εργασίας. Έτσι τα υποσυστήματα που χρησιμοποιήσαμε ήταν ελάχιστα. Για την ακρίβεια έγινε χρήση:

- Του **SysTick timer** ώς βάση μέτρησης χρόνου.
  Η συχνότητα που επιλέξαμε είναι το 1Khz, το οποίο μας οδηγεί σε επαρκές βήμα 1msec .
- Του **GPIO** για την ανάγνωση του κουμπιού και την οδήγηση του LED της πλακέτας, αλλά και την οδήγηση όλων των ψηφιακών σημάτων του shield.
- Του **RCC** για τον έλεγχο και των σημάτων ρολογιού εσωτερικά του επεξεργαστή στα διάφορα υποσυστήματα.
- Του **Cycle count** μηχανισμού στον debugger.
  Ο εν λόγο μηχανισμός χρησιμοποιήθηκε για την μέτρηση του χρόνου απόκρισης του αισθητηρίου εγγύτητας.

Αξίζει ίσως σε αυτό το σημείο να αναφέρουμε πως για την μέτρηση χρόνου χρησιμοποιήσαμε μεταβλητές για απαρίθμηση των διακοπών του SysTick. Η εφαρμογή έτσι μπορούσε βλέποντας την τιμή τους να έχει εικόνα του χρόνου που έχει περάσει από το power up.

Για την οδήγηση των περιφερειακών του shield χρησιμοποιήσαμε επιπλέον:

- Τη σειριακή **USART2** για την επικοινωνία με το υπολογιστή.
  Τη θύρα αυτή τη χρησιμοποιήσαμε για την δημιουργία ενός command interface, μέσω του οποίου μπορούμε να δούμε την κατάσταση του θερμοστάτη, αλλά και να του αλλάξουμε τις ρυθμίσεις.
- Τη σειριακή **USART6** για το 1-wire.
  Η χρήση αυτής της θύρας έγινε όπως αναφέρθηκε και παραπάνω σύμφωνα με τις οδηγίες της maxim-ic (Dallas semiconductors).

Εδώ θα πρέπει να αναφέρουμε πως στο nucleo οι ακροδέκτες της σειριακής επικοινωνίας (D0-D1) είναι συνδεμένοι στην σειριακή του "USB to serial" του ST-LINK2. Για να συνδέαμε μέσω αυτών των ακροδεκτών τη σειριακή για το 1-wire του shield στον επεξεργαστή, θα έπρεπε να τοποθετούσαμε τα jumpers SB62, SB63. Έτσι η κατασκευή μας θα πληρούσε τις προδιαγραφές της εκφώνησης, αλλά δεν θα είχε την έξτρα δυνατότητα του serial console. Ανταυτού λοιπόν "αφήσαμε" το nucleo απείραχτο και συνδέσαμε τη σειριακή του shield στη θύρα USART6, μέσω των morpho headers. Αυτό μας αφήνει με το configuration που περιγράψαμε παραπάνω, ενώ η κατασκευή μας είναι η ίδια είτε υλοποιήσουμε μόνο τις προδιαγραφές της εκφώνησης είτε υλοποιήσουμε και το έξτρα command interface.

3.3. Οδηγός περιφερειακών

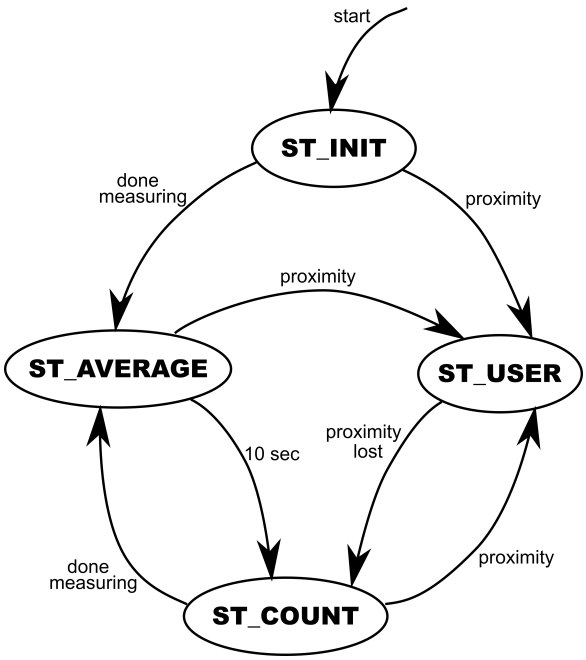
Στον κατάλογο Libraries/drivers/ εκτός από τους οδηγούς του nucleo και του shield, υπάρχουν οι οδηγοί των περιφερειακών πάνω στο shield, αλλά και οι επιπλέων λειτουργίες που χρειαστήκαμε για την ορθή λειτουργία τους. Οι οδηγοί είναι σε ζεύγη αρχείων κώδικα-κεφαλίδας(.c/.h) και ο καθένας αποτελεί ένα module. Η αναλυτική παρουσίαση του κώδικα εδώ θα πλάτιαζε χωρίς να προσφέρει τίποτε χρήσιμο. Θα αναφέρουμε όμως συνοπτικά το κάθε modul- άκι και τον τρόπο λειτουργίας του γενικά.

- **jiffies:**
  Στο module αυτό ρυθμίζουμε ένα timer του επεξεργαστή ώστε να μετράει αδιάκοπα μέχρι μία τιμή με auto-reload. Κάνοντας χρήση αυτού του timer μπορούμε στη συνέχεια να δημιουργήσουμε χρονοκαθυστερήσεις μικρότερες από το time base της εφαρμογής.
- **deque08:**
  Το module αυτό υλοποιεί μια double-ended queue. Την ουρά αυτή τη χρησιμοποιούμε στη σειριακή επικοινωνία με τον υπολογιστή για το serial console. Η υλοποίηση είναι "όσο object oriented γίνεται". Οι συναρτήσεις δηλαδή του module έχουν όλες ένα δείκτη σ΄ ένα "αντικείμενο" τύπου ουράς που παίζει το ρόλο του this pointer άλλων γλωσσών. Έτσι μπορούμε να χρησιμοποιήσουμε περισσότερες από μία ουρές στο ίδιο project με τον ίδιο κώδικα.
- **onewire_uart:**
  Το module αυτό υλοποιεί το πρωτόκολλο επικοινωνίας 1-wire κάνοντας χρήση της σειριακής. Ομοίως και εδώ η υλοποίηση είναι object oriented like. Εδώ όμως **κάνουμε ένα ακόμη κόλπο**. Αρχικά το module προϋποθέτει πως οι ακροδέκτες Tx-Rx της θύρας είναι βραχυκυκλωμένοι. Το βασικό "αντικείμενο" του module εσωτερικά έχει δύο δείκτες σε συναρτήσεις. Τη μία την καλεί για να γράψει στη σειριακή και ταυτόχρονα να διαβάσει το αποτέλεσμα της πραγματικής κατάστασης του bus κατά την προσπάθεια. Την άλλη τη χρησιμοποιεί για να αλλάξει το baudrate της θύρας. Σε όλο το σώμα του κώδικα του module γίνεται χρήση μόνο αυτών των δύο δεικτών και έτσι ***δεν υπάρχει καμία εξάρτηση από το hardware.*** Ο χρήστης του module μπορεί να το χρησιμοποιήσει σε οποιαδήποτε κατασκευή που πληροί τις προϋποθέσεις, να υλοποιήσει για το δικό του hardware τις δύο παραπάνω συναρτήσεις και να τις συνδέσει με το module. Κάτι τέτοιο κάναμε και εμείς εδώ, όπου υλοποιήσαμε τις συναρτήσεις *SHIELD_1W_RW()* και *SHIELD_1W_UART_BR()* και τις συνδέσαμε με το module.
- **alcd:**
  Το module αυτό υλοποιεί ένα οδηγό για την οθόνη. Ομοίως και εδώ η υλοποίηση είναι object oriented like, αλλά και απεξαρτημένη από το hardware μέσω δεικτών σε συναρτήσεις για τα pins DB[4..7], RS, EN, Back-light. Για τον χρονισμό των σημάτων εδώ κάνουμε χρήση των jiffies, καθώς οι χρόνοι που χρειαζόμαστε είναι πολύ μικρότεροι από το time base.
- Τέλος το **hal:**
  Το module αυτό είναι ο συνδετικός κρίκος της εφαρμογής με τους οδηγούς. Σε αυτό το αρχείο

επίσης είναι υλοποιημένες και οι λειτουργίες για την αναγνώριση της εγγύτητας και η ανάγνωση της θερμοκρασίας. *Αυτές* ***οι τελευταίες λειτουργίες είναι οι πιο "ριγμένες"*** *της εργασίας, καθώς ο συντάκτης από καθαρή τεμπελιά δεν αξιώθηκε να τις κάνει ξεχωριστά modules.* Εξάλλου φαίνεται και με μια ματιά πως οι λειτουργίες που είναι υλοποιημένες είναι οι άκρως απαραίτητες για την εφαρμογή μας.

3.4. Εφαρμογή

Έχοντας υλοποιήσει όλα τα παραπάνω η δουλειά μας για την εφαρμογή ήταν πολύ εύκολη. Έτσι στην βασική λειτουργία της εκφώνησης προσθέσαμε ορισμένα πράγματα. Για παράδειγμα η εφαρμογή έχει μια δομή **settings** στην οποία υπάρχουν οι ρυθμίσεις του θερμοστάτη. Πιο συγκεκριμένα αποθηκεύουμε τον τρόπο λειτουργίας, αν δηλαδή ο θερμοστάτης δουλεύει για **ψύξη ή θέρμανση**. Την θερμοκρασία λειτουργίας και μια υστέρηση για αυτήν. Την απόσταση κάτω από την οποία θεωρεί ο θερμοστάτης ότι έχει εγγύτητα και ομοίως μια απόσταση υστέρησης.



Σχήμα 2: *Μηχανή καταστάσεων της διεπαφής χρήστη.*

Ο κώδικας της εφαρμογής μας χωρίζεται σε τέσσερεις non-blocking συναρτήσεις που καλούνται σε βρόχο επανάληψης από την main.

1. **control():**
Η συνάρτηση αυτή αναλαμβάνει να διαβάσει τα αισθητήρια εγγύτητας και θερμοκρασίας και να υπολογίσει τη μέση τιμή τις θερμοκρασίας στο τέλος του κάθε παράθυρου. Η επικοινωνία με τις άλλες συναρτήσεις για την εγγύτητα και την ολοκλήρωση του κύκλου μετρήσεων γίνεται μέσω των σημαιών flag_proximity και signal_cycle αντίστοιχα. Ακόμα με βάση τις ρυθμίσεις και την τρέχουσα μέση θερμοκρασία, ορίζει τη σημαία της κατάστασης εξόδου flag_output.
2. **display():**
Η συνάρτηση αυτή υλοποιεί μια μηχανή καταστάσεων και αναλαμβάνει να εμφανίζει τα μηνύματα της οθόνης. Στο διάγραμμα 2 φαίνονται οι καταστάσεις και ο τρόπος με τον οποίο γίνεται η εναλλαγή. Σε κάθε κατάσταση η συνάρτηση διαμορφώνει ανάλογα το κείμενο στην οθόνη ώστε να πληρούνται οι προδιαγραφές της εργασίας.
3. **outputs():**
Η συνάρτηση αυτή ενεργοποιεί ή απενεργοποιεί τις εξόδους της κατασκευής μας, δηλαδή τα δύο led και το relay. Το πράσινο led ενεργοποιείται όταν ο θερμοστάτης είναι ρυθμισμένος για ψύξη και η μέση θερμοκρασία είναι πάνω από τη ρύθμιση. Το κόκκινο αντίστοιχα όταν ο θερμοστάτης είναι ρυθμισμένος για θέρμανση και η μέση θερμοκρασία είναι κάτω από τη ρύθμιση. Το relay ενεργοποιείται όταν θέλουμε ψύξη ή θέρμανση.
4. **console():**
Η συνάρτηση αυτή υλοποιεί ένα πολύ απλό command interface μέσω του οποίου μπορούμε από την σειριακή του υπολογιστή να δούμε όλες τις θερμοκρασίες και ρυθμίσεις της συσκευής. Ακόμα μπορούμε να αλλάξουμε όλες τις ρυθμίσεις.

Στο σημείο αυτό θα πρέπει να αναφέρουμε πως η λειτουργία των led είναι λιγάκι αλλαγμένη από την εκφώνηση. Ο λόγος είναι ότι ο εν λόγο θερμοστάτης πλέον έχει λειτουργία και ψύξης και θέρμανσης και θέλαμε να δώσουμε στο user interface ένα πιο "δεμένο" ύφος.

```
Welcome to minicom 2.7.1

OPTIONS: I18n
Compiled on May  3 2018, 15:20:11.
Port /dev/ttyACM0, 22:02:40

Press CTRL-A Z for help on special keys

help

Thermostat commands:
info              : Get information
set Tset  <val>   : Set Thermostat's temperature
set Thyst <val>   : Set Thermostat's hysteresis
set Pset  <val>   : Set Proximity
set Physt <val>   : Set Proximity's hysteresis
set mode heating  : Set Thermostat to heating mode
set mode cooling  : Set Thermostat to cooling mode

info

Thermostat data:
Tcur   = 25.8
Tav    = 25.9
Mode   = Heating
Output = Off
Tset   = 21.0, Hyst =  2.0
Pset   = 75.0, Hyst = 10.0

set mode cooling
OK
_
```

Σχήμα 3: *Στιγμιότυπο από την σειριακή επικοινωνία με τον θερμοστάτη.*

4. Συμπερασματα - Παρατηρησεις

Συνοψίζοντας την εμπειρία μας με την παρούσα εργασία δεν έχουμε να αναφέρουμε κάποιο ιδιαίτερο πρόβλημα ή δυσκολία. Η κατασκευή του τυπωμένου ευτυχώς δεν δημιούργησε εκπλήξεις με αποτέλεσμα η συγγραφή του κώδικα να γίνει απρόσκοπτα, αν και ομολογούμε ότι την κάναμε τελευταία στιγμή. Παρατηρώντας την υλοποίηση βέβαια δεν μπορούμε παρά να τονίσουμε και μια παράβλεψη. Ο θερμοστάτης δεν έχει κάποιο τρόπο να αποθηκεύει τις ρυθμίσεις στη flash. Αυτή η λειτουργία θα μπορούσε να είναι αιτία για να ξανασχοληθούμε στο μέλλον με την εργασία και ελπίζουμε αυτή μας η παράβλεψη να πέσει στην κατηγορία *"για το μάτι"*.