Σύστημα ασφαλείας για κυψέλες μελιού

Security system for beehives

Πτυχιακή εργασία του

Βασιλειάδη Κωνσταντίνου

Επιβλέπων καθηγητής

Δρ. Νικολαΐδης Νικόλαος

ΑΛΕΞΑΝΔΡΕΙΟ ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΘΕΣΣΑΛΟΝΙΚΗΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΑΥΤΟΜΑΤΙΣΜΟΥ Τ.Ε.

Ευχαριστίες

Την συγκεκριμένη παράγραφο θα ήθελα να την αφιερώσω σε όλους όσους με βοήθησαν να ολοκληρωθεί επιτυχώς αυτή η εργασία. Θα ήθελα να ευχαριστήσω τον κ. Νικόλαο Νικολαΐδη, ο οποίος μου προσέφερε τη δυνατότητα να ενασχοληθώ με το συγκεκριμένο θέμα, καθώς και τα εφόδια, τόσο θεωρητικά όσο και πρακτικά, ώστε να το φέρω εις πέρας. Επιπλέον, θα ήθελα να ευχαριστήσω ιδιαίτερα τον πατέρα μου, Στέλιο Βασιλειάδη, και τη σύντροφό μου, Ηλεκτρολόγο Μηχανικό, Όλγα Τσιμαράκη, η βοήθεια των οποίων ήταν καθοριστική καθ' όλη τη διάρκεια εκπόνησης της εργασίας. Τέλος, θα ήθελα να ευχαριστήσω τους φίλους μου και την οικογένεια μου για την υποστήριξή τους και τη συμπαράστασή τους καθ' όλη τη διάρκεια των σπουδών μου.

Σύνοψη

Εδώ και πολλές χιλιετίες, ο άνθρωπος προσπαθεί να αναπτύξει προνομιούχους δεσμούς με τη μέλισσα αλλά ο κλάδος της μελισσοκομίας εξελισσόταν με αργούς ρυθμούς. Με το πέρας του Β' παγκοσμίου πολέμου όμως, ο κλάδος αυτός αναπτύχθηκε σημαντικά και σταδιακά καθιερώθηκε η σύγχρονη (νομαδική) μελισσοκομία.

Παρουσιάστηκαν έτσι, δύο σημαντικά προβλήματα στους μελισσοκόμους. Το πρώτο έγκειται στον αυξημένο κίνδυνο κλοπής των μελισσιών, λόγω της τοποθέτησής τους σε πολύ απομακρυσμένες περιοχές. Το δεύτερο πρόβλημα είναι η οικονομική επιβάρυνση που υφίσταται ο παραγωγός εξαιτίας των μετακινήσεών του από και προς τις κυψέλες για επιθεώρηση, τροφοδοσία, τρύγους ή άλλες επεμβάσεις.

Για να προσφέρουμε μια λύση στα παραπάνω, κατασκευάσαμε ένα απομακρυσμένο σύστημα ασφαλείας και παρακολούθησης μελισσιών, με την τοποθέτηση του οποίου σε κάθε κυψέλη, ο μελισσοκόμος θα ενημερώνεται σε τακτά χρονικά διαστήματα για την κατάσταση των μελισσιών του και θα ειδοποιείται άμεσα σε περίπτωση που υπάρξει απόπειρα κλοπής.

Το σύστημα κατασκευάστηκε γύρω από την πλακέτα Arduino UNO και οι λειτουργίες του προγραμματίστηκαν σε γλώσσα C++ με τη βοήθεια του λογισμικού Arduino IDE. Χρησιμοποιήθηκαν ακόμα τέσσερις δυναμοκυψέλες και ένας ενισχυτής σήματος, ο HX711 24bit ADC, ενώ για την επικοινωνία της κυψέλης με τον ιδιοκτήτη της χρησιμοποιήθηκε ένα GSM shield για τον Arduino Uno. Εκτός από το κύριο σύστημα, κατασκευάστηκε και μία κυψέλη σε κλίμακα ώστε να παρουσιάζεται εύκολα η ολοκληρωμένη λειτουργία του.

Αρχικά, περιγράφονται τα χαρακτηριστικά όλων των εξαρτημάτων που χρησιμοποιήθηκαν και αναλύεται ο ηλεκτρονικός και ο μηχανολογικός σχεδιασμός του συστήματος και η κατασκευή των αντίστοιχων τμημάτων. Στη συνέχεια, παρατίθεται και αναλύεται ο κώδικας με τον οποίο υλοποιήθηκαν οι διάφορες λειτουργίες του συστήματος και τέλος, δίνεται μια συνοπτική οικονομική ανάλυση και παρουσιάζονται οι μελλοντικές προοπτικές του.

Abstract

For many millennia, man has been trying to develop privileged ties with the bee, but the beekeeping industry has been slowly evolving. By the end of the Second World War, however, the industry grew significantly and the contemporary (nomadic) beekeeping was established.

With the establishment of Nomadic Beekeeping two major problems appeared for beekeepers. The first is the increased risk of theft of beehives due to their placement in very remote areas. The second problem is the financial burden for the producer due to his transportations to and from the hives for inspection, catering, harvesting or other interventions.

In order to provide a solution to the above, we have constructed a remote security and observation system for the beehives, which keeps the beekeeper being informed at regular intervals about the status of his beehives and notifies him immediately in an attempted theft.

The security system was built around the Arduino UNO board and its functions were programmed with the C ++ language and the help of the Arduino IDE software. Four load cells and a signal amplifier, HX711 24bit ADC, were used in the system, as well as a GSM shield for Arduino Uno which enables the communication of the beekeeper with the beehive. In addition to the main system, a scaled beehive was built, to easily present the system's complete operation procedure.

Initially, the features of all components used are described, and the electronic and mechanical design of the system and the construction of the corresponding parts are analyzed. Then, the code with which the various functions of the system were implemented is presented. Finally, a concise economic analysis and the system's future prospects are described.

Περιεχόμενα

Κατάλογος σχημάτων

Συντομογραφίες

PWM	**P**ulse **W**idth **M**odulation
USART	**U**iversal **S**ynchronous and **A**synchronous **R**eceiver-**T**ransmitter
GSM	**G**lobal **S**ystem for **M**obile communication
GPRS	**G**eneral **P**acket **R**adio **S**ervices
ADC	**A**nalog to **D**igital **C**onverter
WDT	**W**atch**D**og **T**imer
USB	**U**niversal **S**erial **B**us
ICSP	**I**n-**C**ircuit **S**erial **P**rogramming
GPS	**G**lobal **P**ositioning **S**ystem

1 Πρόλογος

1.1 Η μελισσοκομία

Μελισσοκομία ονομάζεται η τέχνη της εκτροφής των μελισσών. Οι μέλισσες έχουν την τάση να δημιουργούν φωλιές και να παραμένουν μέσα σε τρύπες, σε κουφάλες δέντρων κλπ. Αυτό οδήγησε τον άνθρωπο στη σκέψη ότι είναι δυνατό να τις συλλάβει και να τις βάλει να ζήσουν μέσα σε κάποιο κουτί, που να μοιάζει με κουφάλα δέντρου ή με τρύπα σε βράχο, προκειμένου να παράγουν μέλι γι' αυτόν. Έτσι, ο άνθρωπος άρχισε σταδιακά να ασχολείται με τη μελισσοκομία.

Σχήμα 1.1: Μελίσσι μέσα σε βράχο

Ιστορία και εξέλιξη της Μελισσοκομίας

Εδώ και πολλές χιλιετίες, όπως μαρτυρούν ορισμένες τοιχογραφίες καθώς και κείμενα και αρχαία κοσμήματα, ο άνθρωπος προσπαθεί να αναπτύξει προνομιούχους δεσμούς με τη μέλισσα.

Σχήμα 1.2: Μελισσοκομία στην αρχαιότητα

Για πάρα πολύ καιρό, η μελισσοκομική δραστηριότητα περιορίστηκε στη συλλογή. Μέχρι το μεσαίωνα δεν είχαν προστεθεί πολλά καινούρια πράγματα στη μελισσοκομία, εκτός ίσως από τον καπνό ώστε αυτές να μην αγριεύουν πολύ. Αργότερα η μελισσοκομία αποτέλεσε είδος ασχολίας των μοναχών στα μοναστήρια, όπου και αναπτύχθηκε σημαντικά. Με την εφεύρεση της κυψέλης με πλαίσια, τον 19ο αιώνα, γεννήθηκε η σύγχρονη μελισσοκομία για να γίνει σήμερα ένα καθ' ολοκληρία τμήμα της γεωργίας. Τον 20ο αιώνα, κατά τον οποίο γνώρισε και τη μεγαλύτερη ανάπτυξη, διεξήχθησαν σημαντικές έρευνες για να γνωρίσει κανείς καλύτερα την εξαιρετικά περίπλοκη βιολογία της μέλισσας. (1)

Σχήμα 1.3: Ένα τυπικό μελισσοκομικό καπνιστήρι

Σύγχρονη Μελισσοκομία στην Ελλάδα

Ειδικότερα, με το πέρας του Β' παγκοσμίου πολέμου, την ανάπτυξη του οδικού συστήματος της χώρας αλλά και την σταδιακή πρόσβαση και διάδοση της χρήσης των αυτοκινήτων, αναπτύχθηκε η νομαδική μελισσοκομία, δηλαδή η μετακίνηση των μελισσιών σε τοποθε-

σίες με την υψηλότερη ή την πιο επιθυμητή ποικιλία ανθοφορίας (πεύκα, θυμάρια, έλατα, κτλ.).

Σχήμα 1.4: Μελισσοκόμος επιθεωρεί μία κυψέλη

Σήμερα, θεωρείται ασύγκριτα η πιο αποτελεσματική και αποδοτική μέθοδος μελισσοκομίας.

1.2 Έμπνευση

Με την καθιέρωση της νομαδικής μελισσοκομίας παρουσιάστηκαν δύο πολύ σημαντικά προβλήματα στους μελισσοκόμους.

Δεδομένου ότι τα μελίσσια βρίσκονται σε μεγάλη απόσταση από την κατοικία του εκάστοτε παραγωγού, σε κάποια μη περιορισμένη και μη ελεγχόμενη περιοχή, ο κίνδυνος κλοπής τους είναι άμεσος.

Ένα ακόμα εξίσου σημαντικό πρόβλημα είναι η οικονομική επιβάρυνση που έχει ο παραγωγός εξαιτίας των μετακινήσεών του από και προς τις κυψέλες για επιθεώρηση, τροφοδοσία, τρύγους ή οποιαδήποτε άλλη επέμβαση.

1.3 Σκοπός

Λαμβάνοντας υπόψη τα παραπάνω, οδηγηθήκαμε στην απόφαση να κατασκευάσουμε ένα απομακρυσμένο σύστημα ασφαλείας και παρακολούθησης μελισσιών, με τον απλούστερο και οικονομικότερο τρόπο. Με την τοποθέτηση του συστήματος σε κάθε κυψέλη, ο μελισσοκόμος θα ενημερώνεται σε τακτά χρονικά διαστήματα για την κατάσταση των μελισσιών

του και θα ειδοποιείται άμεσα σε περίπτωση που υπάρξει απόπειρα κλοπής σε κάποια από αυτές. Με τον τρόπο αυτό, εξαλείφεται η ανάγκη για τακτικές επισκέψεις στα μελίσσια, μειώνοντας έτσι σε μεγάλο βαθμό τα συνολικά έξοδα του μελισσοκόμου.

Πιο συγκεκριμένα, το σύστημα πραγματοποιεί τις παρακάτω επιμέρους λειτουργίες:

- Μέτρηση του βάρους της κυψέλης κάθε πέντε λεπτά.
- Ενημέρωση του μελισσοκόμου κάθε 24 ώρες για την κατάσταση της κυψέλης μέσω GSM.
- Άμεση ενημέρωση του μελισσοκόμου σε περίπτωση κλοπής.

Τέλος, κάποια από τα ιδιαίτερα χαρακτηριστικά του πρωτότυπου συστήματος που κατασκευάστηκε είναι:

- Ενεργειακή αυτονομία τριών εβδομάδων.
- Ακρίβεια μέτρησης βάρους 100 γραμμαρίων.
- Ξύλινη κατασκευή βάσης ομοίων χαρακτηριστικών της κυψέλης, ώστε να είναι δυσδιάκριτη.
- Ρυθμιζόμενα πόδια και αλφάδια για τη σωστή τοποθέτηση της κυψέλης στο έδαφος ώστε να παρέχονται μετρήσεις χωρίς σφάλματα.

1.4 Ανταγωνισμός

Στην Ελληνική αγορά ο ανταγωνισμός δεν είναι μεγάλος. Η μελισσοκομία είναι ένας κλάδος της γεωργίας όπου η τεχνολογία δεν έχει εξελιχθεί ευρέως. Σε συνδυασμό με το κόστος που έχει ένα τεχνολογικό προϊόν, τα υπάρχοντα συστήματα ασφαλείας και παρακολούθησης είναι λίγα (λιγότερα από δέκα αυτή τη στιγμή) με το κόστος τους να ξεκινά από τα 200 ευρώ και να φτάνει μέχρι και τα 350.

Το σύστημα ασφαλείας που παρουσιάζεται παρακάτω, παρόλο που σε αυτό το στάδιο είναι απλά ένα πρωτότυπο, έχει μεγάλα περιθώρια αναβάθμισης με ποικίλους τρόπους. Σε επόμενο κεφάλαιο θα αναλυθούν οι μελλοντικές προοπτικές του συστήματος και θα παρουσιαστεί η δυνατότητα ποιοτικής υπεροχής συγκριτικά με τον ανταγωνισμό διατηρώντας παράλληλα το πολύ χαμηλό κόστος κατασκευής και λειτουργίας.

1.5 Διάρθρωση Πτυχιακής

Η παρούσα πτυχιακή εργασία αποτελείται από έξι κεφάλαια, τα οποία περιγράφονται συνοπτικά παρακάτω.

Στο Κεφάλαιο 1 γίνεται μια εισαγωγή στη μελισσοκομία και περιγράφονται τα δύο βασικότερα προβλήματα της Σύγχρονης (Νομαδικής) Μελισσοκομίας και ο ανταγωνισμός στην Ελλάδα. Παρουσιάζεται επίσης, ο σκοπός της παρούσας πτυχιακής και η δομή της.

Στο Κεφάλαιο 2 παρουσιάζεται ο ηλεκτρονικός σχεδιασμός, παρατίθενται τα υλικά που χρησιμοποιήθηκαν και αναλύεται η διαδικασία κατασκευής του συστήματος.

Στο Κεφάλαιο 3 αναλύεται, αντίστοιχα, ο μηχανολογικός σχεδιασμός και η κατασκευή, καθώς επίσης και τα υλικά που χρησιμοποιήθηκαν γι' αυτή.

Στο Κεφάλαιο 4 παρουσιάζεται ο τρόπος με τον οποίο έγινε η βαθμονόμηση των αισθητήρων, καθώς και η δομή του κώδικα.

Στο Κεφάλαιο 5 γίνεται μια συνοπτική οικονομική ανάλυση του συστήματος.

Στο Κεφάλαιο 6, το οποίο αποτελεί και τον επίλογο της εργασίας, παρουσιάζονται οι μελλοντικές προοπτικές του συστήματος έτσι ώστε, από πρωτότυπο, να περάσει στο επίπεδο ποιότητας του ανταγωνισμού αλλά και να το ξεπεράσει.

2 Ηλεκτρονικός Σχεδιασμός και Κατασκευή

Για να υλοποιηθεί το σύστημα ασφαλείας και παρακολούθησης χρησιμοποιήθηκαν τέσσερις δυναμοκυψέλες (load cells) και ένας ενισχυτής σήματος, ο HX711 24bit ADC, ενώ για την επικοινωνία της κυψέλης με τον ιδιοκτήτη της χρησιμοποιήθηκε ένα GSM shield για τον Arduino Uno. Όλα τα δεδομένα και αισθητήρια, καθώς και το GSM module είναι ελεγχόμενα από τον μικροελεγκτή της Atmel ATmega328p, ο οποίος είναι ενσωματωμένος στην πλακέτα ανάπτυξης Arduino Uno R3.

Παρακάτω θα αναλυθούν όλα τα εξαρτήματα καθώς και το ηλεκτρονικό κύκλωμα στο οποίο είναι συνδεμένα.

2.1 Arduino Uno R3

Το πιο βασικό ηλεκτρονικό εξάρτημα αυτής της πτυχιακής εργασίας είναι ο Arduino Uno, καθώς όλα είναι φτιαγμένα γύρω από αυτόν.

Σχήμα 2.1: Ο Arduino Uno R3

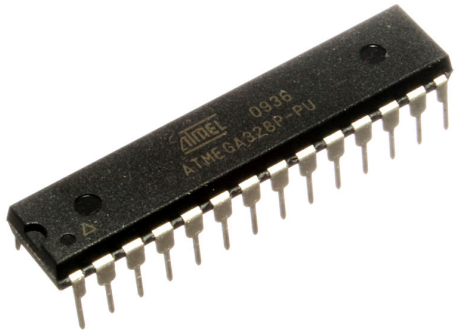
Ο Arduino Uno είναι μια πλακέτα ανάπτυξης βασισμένη στον μικροελεγκτή της ATMEL atmega328p. Διαθέτει 14 ψηφιακές εισόδους/εξόδους από τις οποίες οι 6 μπορούν να χρησιμοποιηθούν σαν PWM, 6 αναλογικές εισόδους, έναν κρύσταλλο 16MHz, υποδοχή USB (female) για σύνδεση με ηλεκτρονικό υπολογιστή, υποδοχή για εξωτερικό τροφοδοτικό, ένα ICSP header και ένα κουμπί «επαναφοράς» (reset button).(2)

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pins	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32KB (0.5KB used by bootloader)
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz

Ο προγραμματισμός του μικροελεγκτή γίνεται με το Arduino IDE, το λογισμικό ανοιχτού κώδικα της Arduino και η γλώσσα προγραμματισμού που χρησιμοποιήθηκε είναι η C++.

Περισσότερα για τον προγραμματισμό αναλύονται στο κεφάλαιο 4.

2.2 Ο μικροελεγκτής ATMEGA328P



Σχήμα 2.2: Ο μικροελεγκτής ATmega328p

Ο υψηλής απόδοσης και ταυτόχρονα χαμηλής κατανάλωσης μικροελεγκτής ATmega328p είναι ένας 8-bit, RISC-based ελεγκτής που, μεταξύ άλλων, συνδυάζει:(3)

- 32KB ISP flash μνήμη με δυνατότητες παράλληλης ανάγνωσης και εγγραφής
- 1024 bytes EEPROM και 2KB SRAM

- 23 εισόδους/εξόδους γενικού σκοπού
- 32 καταχωρητές γενικού σκοπού
- 3 χρονιστές (timers) με δυνατότητα σύγκρισης
- προγραμματιζόμενο USART
- προγραμματιζόμενο χρονιστή watchdog με εσωτερικό ταλαντωτή
- 5 κατηγορίες εξοικονόμησης ενέργειας επιλεγόμενες από το πρόγραμμα
- τάση λειτουργίας 1.8-5.5 Volts.

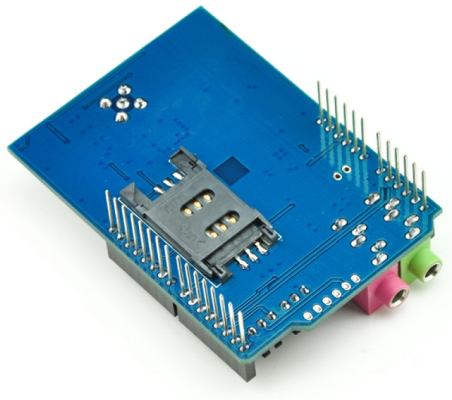
2.3 GSM/GPRS Shield (SIM900)

Το GSM Shield επιτρέπει την επικοινωνία μεταξύ του συστήματος και του μελισσοκόμου. Διαθέτει υποδοχή για κάρτα SIM τηλεφώνου και, με κατάλληλο προγραμματισμό, συνδέεται σε δίκτυο κινητής τηλεφωνίας. Με τον τρόπο αυτό, επιτυγχάνεται η επικοινωνία και η ενημέρωση του παραγωγού για την κατάσταση της κυψέλης του.

Συνδέεται απευθείας πάνω στο Arduino board, καθώς είναι φτιαγμένο στις ίδιες διαστάσεις και περιλαμβάνει τις ακόλουθες δυνατότητες:

- Σύνδεση στο δίκτυο κινητής τηλεφωνίας (GSM)
- Αποστολή και λήψη SMS
- Πραγματοποίηση κλήσεων (χρειάζεται σύνδεση μικροφώνου)
- Λήψη κλήσεων (χρειάζεται σύνδεση ηχείου ή κάποια άλλη έξοδο ήχου)
- Σύνδεση στο διαδίκτυο (GPRS)

(α′) Το GSM Shield SIM900



(β′) Η θέση υποδοχής για την κάρτα SIM βρίσκεται από κάτω

Σχήμα 2.3: Το GSM Shield όπως φαίνεται από την πάνω και κάτω πλευρά

Ο προγραμματισμός του GSM Shield αναλύεται στο Κεφάλαιο 4.

2.4 Αισθητήρες μέτρησης βάρους - Δυναμοκυψέλλες (Load Cells)

Η προστασία της κυψέλης καθώς και η παρακολούθηση της κατάστασής της, επιτυγχάνονται μέσω της μέτρησης βάρους. Ο μικροελεγκτής της πλακέτας Arduino λαμβάνει ανά τακτά χρονικά διαστήματα, μία ένδειξη βάρους από τους αισθητήρες και τη συγκρίνει με την προηγούμενη ένδειξη. Έτσι, σε περίπτωση που παρατηρηθεί μεγάλη διαφορά μεταξύ των δύο μετρήσεων, το σύστημα θεωρεί ότι η κυψέλη έχει κλαπεί και ενημερώνει άμεσα τον μελισσοπαραγωγό.

Ακόμα, σε προκαθορισμένα σημεία της ημέρας, ή και πιο αραιά (π.χ. μία φορά την εβδομάδα), το σύστημα ενημερώνει με μήνυμα SMS τον μελισσοκόμο για την κατάσταση της κυψέλης την δεδομένη στιγμή. Από τις μετρήσεις και τις εναλλαγές στο βάρος της κυψέλης ο μελισσοκόμος μπορεί να εξάγει πληροφορίες για την ποσότητα του μελιού, το ρυθμό αύξησής του κατά τους εαρινούς και καλοκαιρινούς μήνες και την κατάσταση της υγείας του πληθυσμού των μελισσών (ρυθμό μείωσης), κατά τους χειμερινούς μήνες.

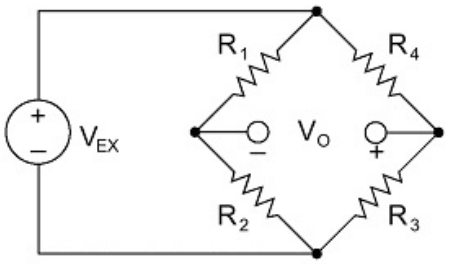
Δυναμοκυψέλη είναι ένα αισθητήριο βάρους. Η πίεση που ασκείται σε αυτήν από το βάρος μετατρέπεται σε ένα ηλεκτρικό, συνήθως, σήμα το οποίο είναι απευθείας ανάλογο με τη δύναμη που ασκείται. Το σήμα της δυναμοκυψέλης μεταφέρεται σε μια ηλεκτρονική διάταξη και αυτή προβάλλει την τιμή της δύναμης.

Μια δυναμοκυψέλη αποτελείται συνήθως από τέσσερις μετρητές τάσης σε διάταξη γέφυρας Wheatstone. Υπάρχουν επίσης, δυναμοκυψέλες ενός μετρητή τάσης (τεταρτημοριακή γέφυρα) ή δύο μετρητών (μισή γέφυρα). Το σήμα εξόδου μιας δυναμοκυψέλης είναι τυπικά της τάξεως των μερικών milliVolts, οπότε απαιτεί ενίσχυση προτού χρησιμοποιηθεί.(4)(5)

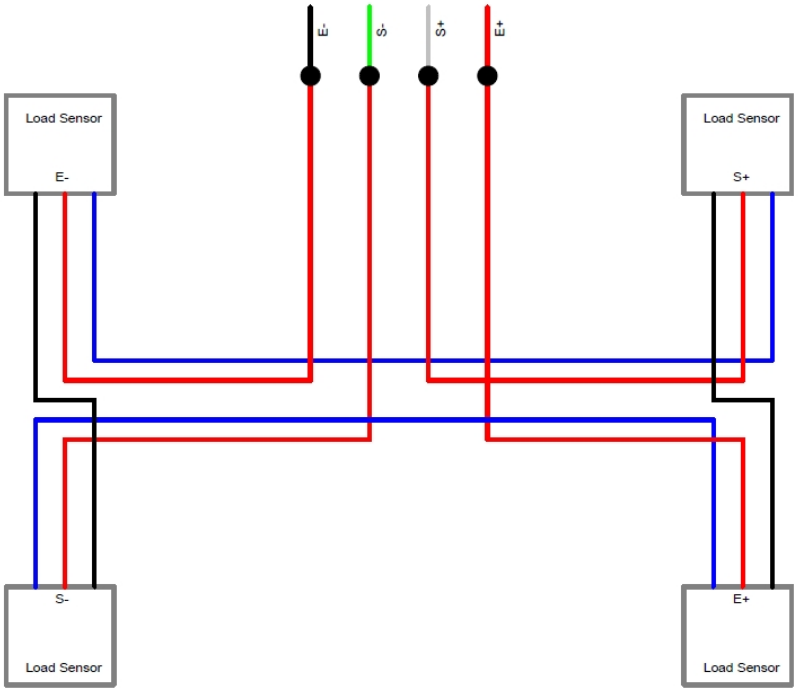
Σχήμα 2.4: Τέσσερις δυναμοκυψέλες

Για την υλοποίηση της εργασίας χρησιμοποιήθηκαν τέσσερις δυναμοκυψέλες (load cells) με δύο μετατροπείς τάσης (μισή γέφυρα Wheatstone) συνδεμένα μεταξύ τους με τέτοιο τρόπο ώστε να σχηματίζουν μια ολοκληρωμένη γέφυρα Wheatstone.

Το συνολικό διάστημα μετρήσεων είναι περίπου 0-150 κιλά.



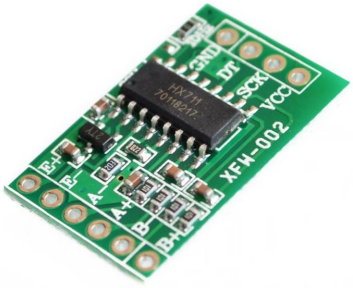
Σχήμα 2.5: Κύκλωμα γέφυρας Wheatstone



Σχήμα 2.6: Τέσσερις δυναμοκυψέλες συνδεμένες σε γέφυρα Wheatstone

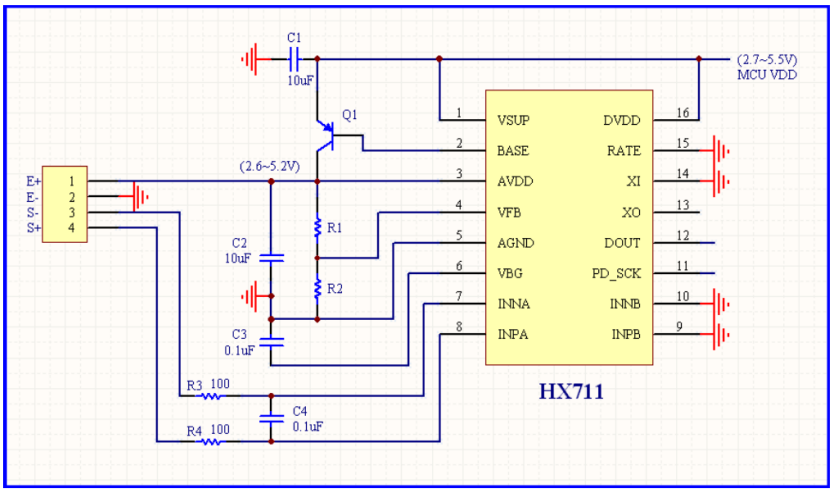
2.5 Ο ενισχυτής HX711 ADC

Όπως αναφέρθηκε προηγουμένως, οι δυναμοκυψέλες μετατρέπουν την δύναμη που ασκείται πάνω τους σε τάση. Επειδή όμως αυτή η τάση είναι της τάξης των millivolts, για να μπορέσει να αξιοποιηθεί θα πρέπει πρώτα να ενισχυθεί. Στην παρούσα εργασία, για να πραγματοποιηθεί αυτή η ενίσχυση χρησιμοποιήθηκε ο ενισχυτής σήματος HX711.



Σχήμα 2.7: Ο ενισχυτής σήματος HX711 ADC

Το ολοκληρωμένο HX711 είναι ένας υψηλής ακρίβειας μετατροπέας αναλογικού σήματος σε ψηφιακό, σχεδιασμένος ειδικά για ζυγαριές ακριβείας. Έχει δύο κανάλια εισόδου, επομένως μπορεί να δεχθεί σήματα από δύο δυναμοκυψέλες ή ζυγαριές ταυτόχρονα, και ενισχύει το σήμα που λαμβάνει 128 φορές.



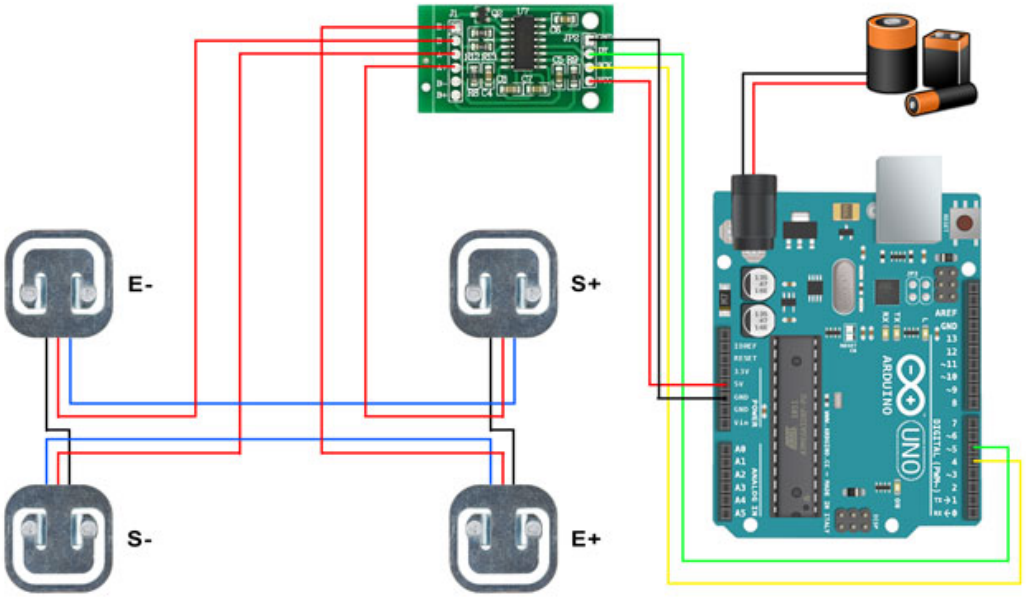
Σχήμα 2.8: Το ηλεκτρονικό κύκλωμα του ενισχυτή HX711(6)

Παρακάτω παρατίθενται τα σημαντικότερα χαρακτηριστικά του ολοκληρωμένου κυκλώματος HX711.(6)

- Διεπαφή Synchronous Serial Interface (απαιτείται 2 pin για λειτουργία)
- Πολυπλέκτης επιλογής καναλιού
- Προγραμματιζόμενος ενισχυτής κέρδους (PGA) 32, 64 και 128
- On-chip power-on-reset
- Εύρος τροφοδοσίας 2.6-5.5V
- 1.5mA μέγιστη κατανάλωση ρεύματος
- Θερμοκρασία λειτουργίας: -40 έως +85 °C

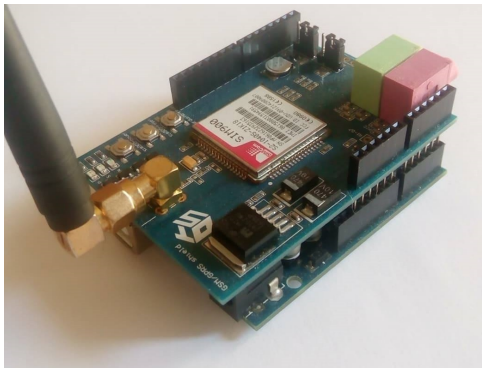
2.6 Ηλεκτρονικό Κύκλωμα

Όλα τα παραπάνω ηλεκτρονικά εξαρτήματα συνδέονται μεταξύ τους σε ένα πολύ απλό κύκλωμα, το οποίο φαίνεται στο Σχήμα 2.9.



Σχήμα 2.9: Το ηλεκτρονικό κύκλωμα του συστήματος ασφαλείας

Το GSM Shield δεν έχει σχεδιαστεί στο παραπάνω κύκλωμα καθώς συνδέεται κατευθείαν πάνω στην πλακέτα του Arduino, όπως φαίνεται στο Σχήμα 2.10.



Σχήμα 2.10: Ο Arduino Uno μαζί με GSM Shield

3 Μηχανολογικός Σχεδιασμός και Κατασκευή

Ένα σύστημα ασφαλείας για κυψέλες πρέπει να τηρεί κάποιες προϋποθέσεις για να μπορεί να θεωρηθεί αξιόπιστο

- απέναντι στις καιρικές συνθήκες στις οποίες θα εκτεθεί και
- στο γεωλογικό περιβάλλον, είτε αυτό είναι ένα επίπεδο χωμάτινο έδαφος όπως για παράδειγμα ένα χωράφι, είτε είναι ένα ανισόπεδο έδαφος γεμάτο πέτρες σε κάποια ορεινή περιοχή.

Επίσης, καθώς είναι αδύνατο για ένα μελισσοκόμο να εγκαταστήσει από ένα σύστημα ασφαλείας σε κάθε κυψέλη, εξαιτίας του μεγάλου κόστους, το εν λόγω σύστημα συνήθως εγκαθίσταται σε ορισμένες από αυτές, στρατηγικά τοποθετημένες, με την ελπίδα πως ένας επίδοξος κλέφτης θα αποπειραθεί να αποσπάσει μία από αυτές. Θα πρέπει λοιπόν το σύστημα να είναι όσο το δυνατόν καλύτερα κρυμμένο, να ταιριάζει με το περιβάλλον της κυψέλης και να μην ξεχωρίζει εύκολα από αυτήν. Για τους παραπάνω λόγους, το σύστημα κατασκευάστηκε ως βάση για την κυψέλη, στις ίδιες ακριβώς διαστάσεις και από το ίδιο υλικό.

Στο παρόν κεφάλαιο θα αναλυθεί η διαδικασία κατασκευής του μηχανολογικού μέρους του πρωτότυπου συστήματος, το οποίο κατασκευάστηκε σε κλίμακα $1:2$. Χάριν προσομοίωσης, κατασκευάστηκε και ένα μοντέλο κυψέλης στην ίδια κλίμακα.

Τα χαρακτηριστικά του πρωτότυπου συστήματος τα οποία και θα αναλυθούν παρακάτω είναι τα εξής:

- Κατασκευή βάσης σε κλίμακα $1:2$
- Τοποθέτηση αισθητήρων
- Άνω πλάκα και προστατευτικό κάλυμμα αυτής
- Τοποθέτηση ρυθμιζόμενων ποδιών βάσης και αλφαδιών για εύκολη ευθυγράμμιση της ζυγαριάς και αποφυγή λάθος ενδείξεων λόγω κεκλιμένου επιπέδου
- Κατασκευή μοντέλου κυψέλης σε κλίμακα $1:2$

3.1 Κατασκευή βάσης κυψέλης σε κλίμακα

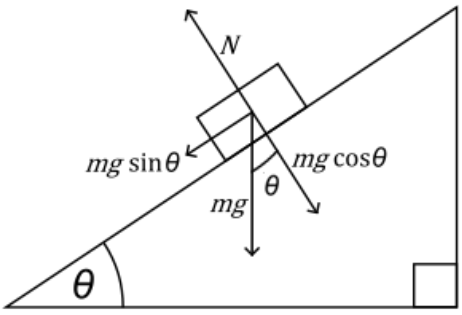
Για την κατασκευή χρησιμοποιήθηκαν ξύλα, βίδες, ξυλόκολλα και λαμαρίνα, σέγα χειρός για την κοπή των ξύλων, κατσαβίδια, δράπανο, τροχός κοπής μετάλλων, μέτρο και σκαρπέλα ξυλογλυπτικής (για την δημιουργία θέσεων για τις δυναμοκυψέλες).

Σε πρώτη φάση δημιουργήθηκε το βασικό κουτί, οι θέσεις για τις δυναμοκυψέλες και οι απαραίτητες τρύπες για την καλωδίωσή τους. Στη συνέχεια, κόπηκε το ξύλο που εξυπηρετεί ως άνω πλάκα. Τέλος, κόπηκε και διαμορφώθηκε ένα κομμάτι λαμαρίνας το οποίο τοποθετήθηκε πάνω από αυτή.

Η λαμαρίνα εξυπηρετεί δύο σημαντικούς σκοπούς. Ο πρώτος είναι η σταθερότητα. Με το σχήμα που της δόθηκε κρατά σταθερή την πλάκα, ώστε να μην υπάρχει η πιθανότητα να γλιστρήσει η κυψέλη μαζί με αυτή. Ο δεύτερος είναι η προστασία από τις καιρικές συνθήκες. Όπως φαίνεται στο Σχήμα 3.1, αγκαλιάζει την ξύλινη βάση μη επιτρέποντας έτσι στην υγρασία και την βροχή να παραμείνουν στην ξύλινη επιφάνεια και να τη διαβρώσουν.

Σχήμα 3.1: Η βάση στο πρώτο στάδιο κατασκευής

Σε δεύτερη φάση μας απασχόλησε το έδαφος στο οποίο θα τοποθετηθεί η κυψέλη και κατ' επέκταση, το σύστημα ασφαλείας. Μία υπό κλίση ζυγαριά θα μετρήσει μόνο την κάθετη συνιστώσα του πραγματικού βάρους ενός αντικειμένου, δηλαδή τη συνιστώσα που είναι κάθετη στο κεκλιμένο επίπεδο (Σχήμα 3.2).

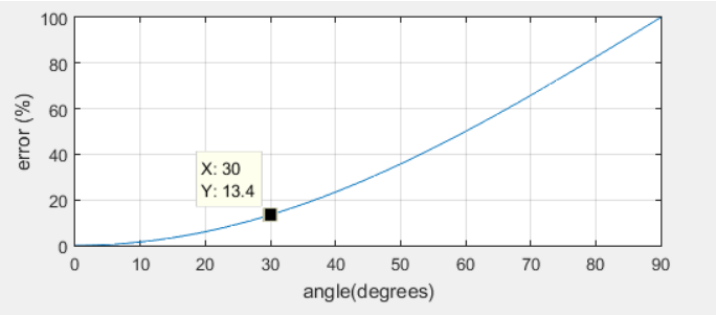


Σχήμα 3.2: Αντικείμενο υπό γωνία θ

Προκύπτει λοιπόν ένα σφάλμα μέτρησης, το οποίο υπολογίζεται από την παρακάτω σχέση:

$$error = \frac{Real\,Weight - Measured\,Weight}{Real\,Weight} = \frac{mg - mgcos\theta}{mg} = 1 - cos\theta$$

Από την σχέση αυτή προέκυψε το γράφημα του Σχήματος 3.3, στο οποίο φαίνεται πως, όσο μεγαλύτερη είναι η κλίση της κυψέλης, τόσο μεγαλύτερο θα είναι το σφάλμα μέτρησης του συστήματος.



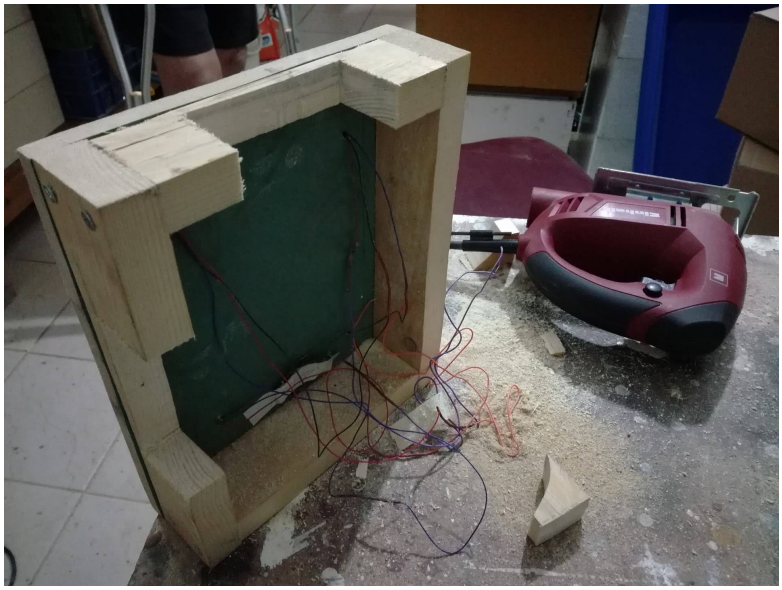
Σχήμα 3.3: Το σφάλμα μέτρησης συναρτήσει της κλίσης

Για το λόγο αυτό, προστέθηκαν στο σύστημα πόδια με ρυθμιζόμενο ύψος και τοποθετήθηκαν στα πλαϊνά τοιχώματα αυτής αλφάδια, έτσι ώστε να μπορεί ο παραγωγός εύκολα, την στιγμή που θα τοποθετήσει το σύστημα με την κυψέλη στο έδαφος, να ρυθμίσει την κλίση τους. Τα αλφάδια, δημιουργήθηκαν σε ξεχωριστά κομμάτια, και με τρόπο έτσι ώστε εύκολα να μπορεί να γίνει η ευθυγράμμισή τους με το υπόλοιπο σύστημα. Όπως φαίνεται στα Σχήματα 3.4 και 3.5, οι 2 βίδες στο κάθε ένα επιτρέπουν την ρύθμιση της κλίσης τους με μέγιστη ακρίβεια και αξιοπιστία.

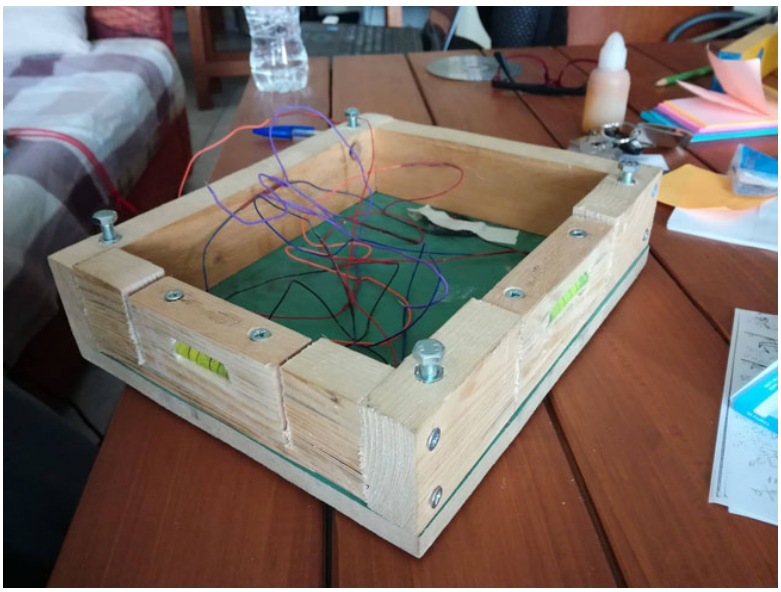
Σχήμα 3.4: Κατασκευή θέσης και τοποθέτηση αλφαδιών

Σχήμα 3.5: Τα αλφάδια ολοκληρωμένα, πριν ενσωματωθούν στη βάση

Σχήμα 3.6: Τροποποιημένη βάση για υποδοχή των αλφαδιών 1



Σχήμα 3.7: Η βάση τροποποιημένη για τα αλφάδια 2



Σχήμα 3.8: Η βάση τροποποιημένη με αλφάδια και ρυθμιζόμενα πόδια 1

Σχήμα 3.9: Η βάση τροποποιημένη με αλφάδια και ρυθμιζόμενα πόδια 2

3.2 Κατασκευή μοντέλου κυψέλης

Για το μοντέλο της κυψέλης χρησιμοποιήθηκε ξύλο, λαμαρίνα για το καπάκι της (όπως είναι και σε μια πραγματική) και κλείστρα.

Σχήμα 3.10: Το μοντέλο κυψέλης υπό κατασκευή και η βάση

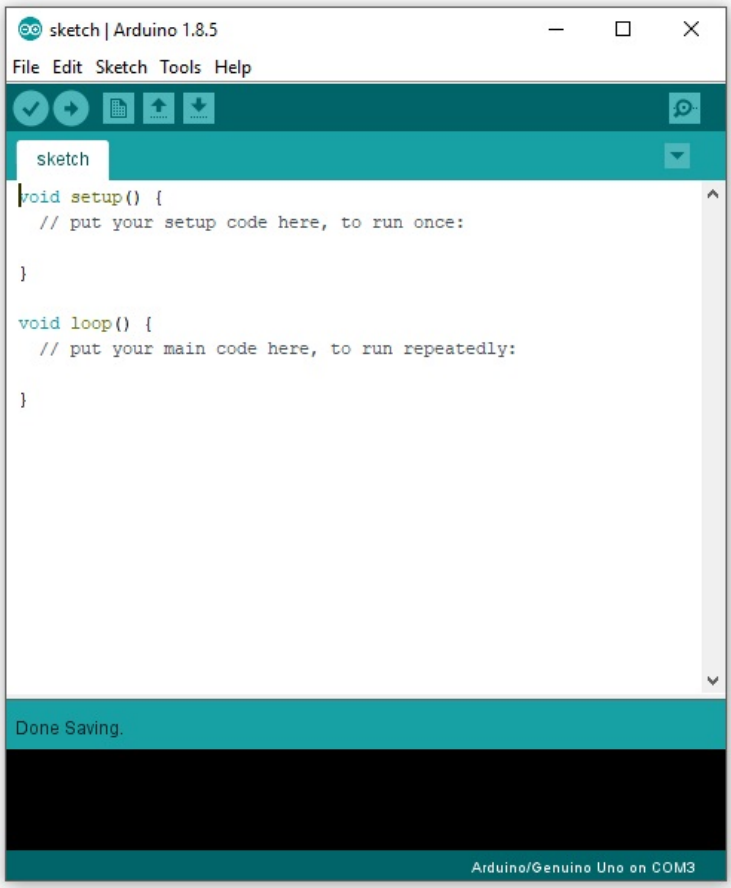
Σχήμα 3.11: Το μοντέλο κυψέλης υπό κατασκευή 2

Σχήμα 3.12: Το σύστημα ασφαλείας και η κυψέλη στην τελική τους μορφή

4 Προγραμματισμός, Βαθμονόμηση Αισθητήρων και Δοκιμές

Ο προγραμματισμός του μικροελεγκτή έγινε σε γλώσσα C++, στο Arduino IDE, το λογισμικό ανοιχτού κώδικα της Arduino. Για την υλοποίηση των επιθυμητών λειτουργιών του συστήματος χρησιμοποιήθηκαν οι κατάλληλες βιβλιοθήκες (7)(10)(11), οι οποίες συνοδεύουν τα εξαρτήματα GSM Shield και HX711.

Παρακάτω θα αναλυθεί τμηματικά ο κώδικας ανάλογα με τις λειτουργίες του συστήματος.



Σχήμα 4.1: Το περιβάλλον προγραμματισμού του μικροελεγκτή, Arduino IDE

4.1 Βαθμονόμηση Αισθητήρων

4.1.1 Βαθμονόμηση

Όπως αναφέρθηκε σε προηγούμενο κεφάλαιο, οι δυναμοκυψέλες μετατρέπουν τη δύναμη που ασκείται πάνω τους σε ηλεκτρικό σήμα της τάξης των millivolts. Για να μπορέσει να αξιοποιηθεί αυτό το σήμα πρέπει να γίνει βαθμονόμηση των αισθητήρων. Πρέπει, δηλαδή, να γίνει μετατροπή αυτών των σημάτων σε τιμές τις οποίες θα μπορεί να «διαβάσει» ο άνθρωπος. Αυτό γίνεται εύκολα με προσαρμογή του κώδικα και πολλαπλές δοκιμές με αντικείμενα γνωστού βάρους.

Συγκεκριμένα, η βαθμονόμηση γίνεται χρησιμοποιώντας την εξίσωση

$$\frac{Raw\,Data\,Reading\,(mV) - Scale\,Offset\,(mV)}{Calibration\,Factor} = Human\,Readable\,Data\,(kg)$$

όπου Raw Data Reading είναι η ένδειξη σε millivolts που δίνουν οι δυναμοκυψέλες για ένα αντικείμενο γνωστού βάρους, Scale Offset είναι το απόβαρο της ζυγαριάς και Calibration Factor είναι η σταθερά της εξίσωσης, η τιμή της οποίας καθορίζεται κάθε φορά από το διάστημα τιμών στο οποίο ανήκει η τρέχουσα μέτρηση των αισθητήρων.

Παρακάτω φαίνεται ο κώδικας που δημιουργήθηκε για την εύρεση της κατάλληλης τιμής της παραμέτρου βαθμονόμησης, calibration_factor.

Αφού αρχικοποιηθεί το calibration_factor σε μια προκαθορισμένη τιμή, ενεργοποιείται το σύστημα και δίνει στο χρήστη οδηγίες για τη σωστή χρήση του. Στη συνέχεια αφού υπολογιστεί το απόβαρο και μηδενιστεί η ένδειξη της μέτρησης, ξεκινά μία επανάληψη κατά την οποία επιστρέφονται συνεχώς η τιμή της μέτρησης (scale.get_units()) και η τιμή της παραμέτρου βαθμονόμησης (calibration_factor).(7)

```
#include "HX711.h"
#define DOUT  3
#define CLK  2
HX711 scale(DOUT, CLK);
float calibration_factor = 3655; //random starting value

void setup() {
  Serial.begin(9600);
  Serial.println("HX711 calibration sketch");
  Serial.println("Remove all weight from scale");
  Serial.println("After readings begin, place known weight on scale");
  Serial.println("Press + or a to increase calibration factor");
  Serial.println("Press - or z to decrease calibration factor");
  scale.set_scale();
  scale.tare(); //Reset the scale to 0
```

```
  long zero_factor = scale.read_average(); //Get a baseline reading
  Serial.print("Zero factor: ");
  Serial.println(zero_factor);
}

void loop() {
  scale.set_scale(calibration_factor); //Adjust to this calibration factor
  Serial.print("Reading: ");
  Serial.print(scale.get_units());
  Serial.print(" kg");
  Serial.print(" calibration_factor: ");
  Serial.print(calibration_factor);
  Serial.println();
  if(Serial.available())
  {
    char temp = Serial.read();
    if(temp == '+' || temp == 'a')
      calibration_factor += 10;
    else if(temp == '-' || temp == 'z')
      calibration_factor -= 10;
  }
}
```

Πραγματοποιώντας λοιπόν δοκιμές με γνωστά βάρη, το calibration_factor προσαρμόστηκε στην κατάλληλη τιμή ώστε η τιμή της μέτρησης του συστήματος να είναι η πραγματική. Τα αποτελέσματα φαίνονται στον παρακάτω πίνακα.

Γνωστά βάρη (kg)	0-1	1-4	4-5	5-6	6-7	7-8
calibration_factor	15685	15405	15105	15445	15215	14885

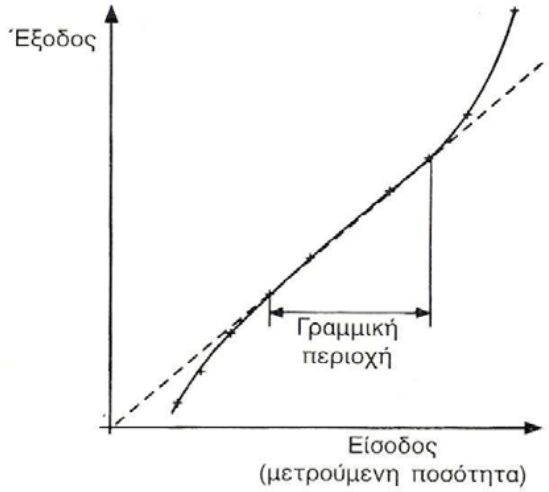
Γνωστά βάρη (kg)	8-9	9-10	10-11	11-12	12-13	13-14	>14
calibration_factor	15055	14745	14715	14635	14555	14835	14150

4.1.2 Γραμμικότητα και ακρίβεια αισθητήρων

Το σύστημα ασφαλείας είναι αξιόπιστο και έχει πολύ μεγάλη ακρίβεια μέτρησης. Το σφάλμα μιας μέτρησης εξαρτάται από διάφορους παράγοντες όπως, παραδείγματος χάριν, το πόσο κοντά βρίσκεται ένα μετρούμενο βάρος (π.χ. μια κυψέλη) στα άκρα του διαστήματος μέτρησης των αισθητήρων (στην συγκεκριμένη περίπτωση 0-150 κιλά), η θερμοκρασία αλλά και η ποιότητα των υλικών. Αυτό σημαίνει ότι οι μετρήσεις δεν είναι γραμμικές, με το φαινόμενο να ενισχύεται σημαντικά εάν συνδυαστούν οι παραπάνω παράγοντες.

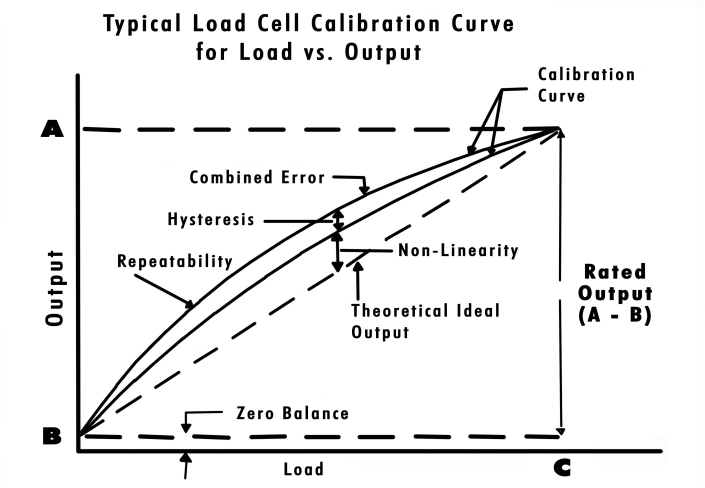
Ενδεικτικά στο γράφημα του Σχήματος 4.2 φαίνεται η μη γραμμικότητα της εξόδου μιας ιδανικής δυναμοκυψέλης, για σταθερό παράγοντα βαθμονόμησης, όπου είναι εύκολο να

παρατηρηθεί η ενίσχυση του σφάλματος στα άκρα του διαστήματος μέτρησης. Στην πραγματικότητα, το διάγραμμα εισόδου-εξόδου μιας δυναμοκυψέλης θα έχει πολύ περισσότερες διακυμάνσεις.



Σχήμα 4.2: Γραμμικότητα δυναμοκυψέλης(8)

Χωρίζοντας το εύρος μέτρησης του συστήματος σε μικρότερα τμήματα, καθένα από τα οποία έχουν το δικό τους παράγοντα βαθμονόμησης, η γραμμικοποίηση της εξόδου των επιμέρους διαστημάτων ήταν μια αρκετά εύκολη διαδικασία. Καθώς το φαινόμενο εμφανίζεται κυρίως στις χαμηλές περιοχές, το διάστημα 0-14 κιλά χρειάστηκε να χωριστεί σε πολλά μικρότερα τμήματα, ενώ για το υπόλοιπο διάστημα ένα και μόνο αρκεί.



Σχήμα 4.3: Τυπική καμπύλη βαθμονόμησης δυναμοκυψέλης(9)

Παρακάτω φαίνεται πως διαμορφώθηκε ο κώδικας ώστε να υλοποιηθεί η τμηματοποίηση του διαστήματος μέτρησης και η εναλλαγή του παράγοντα βαθμονόμησης.

Συγκεκριμένα, αφού αρχικοποιηθεί το calibration_factor σε μια προκαθορισμένη τιμή, ο μικροελεγκτής λαμβάνει μια μέτρηση από τις δυναμοκυψέλες και ανάλογα με το διάστημα στο οποίο αυτή ανήκει, αλλάζει εκ νέου το calibration_factor στην κατάλληλη τιμή (η οποία προέκυψε από τις αρχικές δοκιμές). Στη συνέχεια, χρησιμοποιώντας το νέο calibration_factor, το σύστημα επιστρέφει τη σωστή ένδειξη βάρους. Τέλος, για να επιτευχθεί η μέγιστη ακρίβεια, η διαδικασία επαναλαμβάνεται πέντε φορές. Έτσι, η πιθανότητα να ληφθεί λανθασμένη μέτρηση, πρακτικά μηδενίζεται και η ακρίβεια των μετρήσεων του συστήματος είναι εξαιρετική. Το σφάλμα παραμένει συνεχώς μικρότερο από 100 γραμμάρια, ενώ ακόμα και όταν η μέτρηση βρίσκεται στις «ευαίσθητες» περιοχές, κοντά στα άκρα του συνολικού διαστήματος μέτρησης, αυτό δεν ξεπερνάει τα 200 γραμμάρια.

```
//Because load cells are not linear, 5 measurements//
//are taken with the correct calibration factor//
for (i=0; i<5; i=i+1){
  weight = 0.453592 * scale.get_units();
  if ( weight <= 3) {
    calibration_factor = 3855;
  }
  else if (weight <= 4) {
    calibration_factor = 4255;
  }
  else if (weight <= 5) {
    calibration_factor= 5040;
  }
  else if (weight <= 6) {
    calibration_factor= 5283;
  }
  else if (weight <= 7) {
    calibration_factor= 5465;
  }
  else if (weight <= 8) {
    calibration_factor= 5500;
  }
  else if (weight <= 9) {
    calibration_factor= 5832;
  }
  else if (weight <= 10) {
    calibration_factor= 5955;
  }
  else {
    calibration_factor= 7505;
  }
  scale.set_scale(calibration_factor);
  weight =0.453592 * scale.get_units();
}
//------------------------------//
```

4.2 GSM και επικοινωνία με τον χρήστη

Η επικοινωνία με το χρήστη επιτυγχάνεται μέσω του GSM Shield. Το συγκεκριμένο εξάρτημα έχει τη δυνατότητα να συνδέεται στο δίκτυο κινητής τηλεφωνίας, ώστε να πραγματοποιεί και να δέχεται κλήσεις και SMS.

Το σύστημα ασφαλείας που κατασκευάστηκε, επικοινωνεί με τον εκάστοτε μελισσοκόμο σε δύο περιπτώσεις.

Η πρώτη είναι για μια απλή ενημέρωση, η οποία γίνεται μία φορά κάθε 24 ώρες. Δίνεται έτσι η δυνατότητα στον παραγωγό να ενημερώνεται καθημερινά, την ίδια ώρα της ημέρας, για το τρέχον βάρος της κυψέλης στην οποία έχει εγκατασταθεί το σύστημα, ώστε να μπορεί να συγκρίνει τις διάφορες μετρήσεις και να εξάγει τα επιθυμητά συμπεράσματα, με σκοπό την βελτίωση της εργασίας του.

Παρακάτω φαίνεται ο κώδικας που υλοποιεί τη λειτουργία ενημέρωσης του παραγωγού: (10)

```
 //float weight is converted to string and combined//
//with a text for the sms (The weight of the beehive is: xx)//
strweight = String(weight);
strsms = "The weight of the beehive is: " + strweight;
strsms.toCharArray(sms_text, 100);
//------------------------------//

//SMS is send with the current status of the beehive (every 24hs)//
if (t == 286) {
  delay(10000);
  gsm.begin(9600);
  delay(1000);
  sms.SendSMS("003069XXXXXXXX", sms_text);
  delay(1000);
  gsmpower();
  t = 0;
}
//---------------------------//
```

Όπως φαίνεται στο πρώτο κομμάτι του κώδικα, αρχικά μετατρέπεται η μεταβλητή *weight* στην οποία είναι αποθηκευμένη η μέτρηση, από *float*, σε *string* και έπειτα «προετοιμάζεται» το κείμενο που θα σταλθεί σε μήνυμα στο κινητό τηλέφωνο του μελισσοκόμου, *strsms*. Στο δεύτερο κομμάτι ενεργοποιείται το GSM Shield και στέλνει το μήνυμα στον αριθμό που είναι αποθηκευμένος στο πρόγραμμα.

Η δεύτερη περίπτωση στην οποία το σύστημα ασφαλείας χρειάζεται να επικοινωνήσει με τον μελισσοκόμο, είναι σε περίπτωση απόπειρας κλοπής.

Όπως φαίνεται στον παρακάτω κώδικα, κάθε φορά που λαμβάνεται μέτρηση από τις δυναμοκυψέλες, η τρέχουσα τιμή αποθηκεύεται ώστε να διατηρηθεί όταν ληφθεί επόμενη

μέτρηση. Με τον τρόπο αυτό, συγκρίνεται η τελευταία μέτρηση με την αμέσως προηγούμενη και, σε περίπτωση διαφοράς μεγαλύτερης των δύο κιλών, ενεργοποιείται η σημαία theft και το σύστημα εισέρχεται σε κατάσταση κλοπής. Στην κατάσταση αυτή, το σύστημα σταματά να παίρνει μετρήσεις και εισέρχεται σε έναν ατέρμονα βρόγχο και ειδοποιεί τον ιδιοκτήτη με SMS. Ο μοναδικός τρόπος για να λειτουργήσει και πάλι κανονικά το σύστημα είναι να κάνει ο μελισσοκόμος επανεκκίνηση του συστήματος. Απαιτείται δηλαδή, η φυσική του παρουσία.

```
if (prev_weight - weight > 2) {
   theft = true;
   while (true) {
     if (theft) {
       gsm.begin(9600);
         sms.SendSMS("003069XXXXXXXX", "ATTENTION! BEEHIVE STOLEN!");
         theft = false;
     }
   gsmpower();
   //Put code for security measures here//
   }
 }
 prev_weight = weight;
 //---------------------------//
```

Τέλος, αν χρειάζεται, ή θέλει, ο μελισσοκόμος μία «έκτακτη» ενημέρωση μπορεί να στείλει ένα μήνυμα στο σύστημα και αυτό θα του απαντήσει με μια αναφορά για την κατάσταση της κυψέλης του όπως κάνει κάθει 24 ώρες. Σε περίπτωση που ο αποστολέας του μηνύματος δεν είναι ο ιδιοκτήτης της κυψέλης, απλώς θα το αγνοήσει. Αμέσως μετά, διαγράφονται όλα τα εισερχόμενα μηνύματα για να διασφαλιστεί ότι δεν θα γεμίσει η μνήμη της κάρτας SIM.

```
//Check if there is incoming SMS and reply if it is from the owner//
if (t == y) {
  gsm.begin(9600);
  delay(10000);
  sms_position = sms.IsSMSPresent(SMS_UNREAD);
  delay(1000);
  if (sms_position)
  {
    sms.GetSMS(sms_position, phone_number, insms_text, 100);
    delay(500);
    //if SMS is from owner, reply.//
    if (strcmp( adminphone, phone_number ) == 0 ) {
      sms.SendSMS("003069XXXXXXXX", sms_text);
    }
    if (gsm.begin(9600)) {
     for (i = 1; i <= 20; i++) sms.DeleteSMS(i); //delete all messages//
    }
  }
```

```
    gsmpower();
    delay(500);
    y=y+12;
    if (y>=274) y=12;
}
//---------------------------//
```

4.3 Εξοικονόμηση ενέργειας

Ένα από τα πιο σημαντικά χαρακτηριστικά ενός συστήματος ασφαλείας για κυψέλες είναι η αυτονομία. Θα πρέπει να έχει όσο το δυνατόν μεγαλύτερη διάρκεια ζωής, με τη μικρότερη δυνατή κατανάλωση, χωρίς να χρειάζεται συχνή αλλαγή μπαταρίας, καθώς αυτό αυξάνει το κόστος του σε βάθος χρόνου. Γι' αυτό το λόγο, το σύστημα είναι κατάλληλα προγραμματισμένο έτσι ώστε να μην δουλεύει συνέχεια, αλλά να ενεργοποιείται μια λειτουργία εξοικονόμησης ενέργειας (*sleep/power down mode*).

Όπως φαίνεται και παρακάτω στον κώδικα, πρώτα πρέπει να τεθεί ο ενισχυτής HX711 σε λειτουργία εξοικονόμησης ενέργειας. Ύστερα, με τις κατάλληλες εντολές και για χρόνο που έχει δηλωθεί μέσα στο πρόγραμμα (στη συγκεκριμένη περίπτωση, για διαστήματα διάρκειας 5 λεπτών), μπαίνει σε λειτουργία εξοικονόμησης ενέργειας ο μικροελεγκτής. (11) Παράλληλα, το GSM Shield παραμένει απενεργοποιημένο και ενεργοποιείται μόνο για να εκτελέσει τις απαραίτητες λειτουργίες.

```
//Enter power saving mode//
scale.power_down(); //HX711 power down//
delay(1000);
sleep.pwrDownMode(); //Arduino power down//
sleep.sleepDelay(sleepTime);
delay(100);
scale.power_up(); //HX711 power up//
//---------------------------//
```

Όσο ο μικροελεγκτής βρίσκεται σε λειτουργία εξοικονόμησης ενέργειας χρησιμοποιεί μόνο τον χρονιστή watchdog, ενώ όλα οι υπόλοιποι χρονιστές και λειτουργίες του διακόπτονται. Με αυτό τον τρόπο, επιτυγχάνεται η μεγαλύτερη δυνατή εξοικονόμηση ενέργειας. Με τη χρήση του χρονιστή watchdog προέκυψε το πρόβλημα ότι ο μέγιστος χρόνος που μπορεί να μετρήσει είναι οχτώ δευτερόλεπτα, με το πέρας των οποίων κάνει επανεκκίνηση σε ολόκληρη την πλακέτα Arduino. Το συγκεκριμένο πρόβλημα λύθηκε εύκολα, με την επανεκκίνηση του WDT πριν περάσει το χρονικό όριο των οχτώ δευτερολέπτων και έναν μετρητή (counter), ο οποίος μετρά τον αριθμό των επανεκκινήσεων. Έτσι μπορεί εύκολα να καθοριστεί ο χρόνος κατά τον οποίο ολόκληρο το σύστημα θα βρίσκεται σε λειτουργία εξοικονόμησης ενέργειας.

Με την λειτουργία εξοικονόμησης ενέργειας η μέση κατανάλωση του συστήματος μειώθηκε στα 1,7mA από την αρχική τιμή των 50,67mA. Το γεγονός αυτό επέκτεινε την

αυτονομία του συστήματος από περίπου 5 μέρες (110.5 ώρες διάρκεια ζωής της μπαταρίας), σε 3 εβδομάδες (περισσότερες από 523 ώρες).

Παρόλο που υπάρχει εντυπωσιακή μείωση της κατανάλωσης ακόμα και τώρα, σε περίπτωση που το πρωτότυπο αυτό περάσει στην μαζική παραγωγή, μπορεί να βελτιωθεί ακόμα περισσότερο. Ένα ενδεικτικό παράδειγμα είναι ότι, εάν στη θέση της πλακέτας Arduino χρησιμοποιηθεί μόνος του ο μικροελεγκτής, χωρίς τα επιπλέον περιφερειακά (θύρα USB, LEDs, ICSP, κτλ.), η ηλεκτρική κατανάλωση του συστήματος μειώνεται στην τάξη των μA. Συνεπώς, η αυτονομία του συστήματος μπορεί να αυξηθεί στους έξι έως και δώδεκα μήνες!

4.4 Κώδικας

Σε αυτή την ενότητα παρουσιάζεται ο συνολικός κώδικας του συστήματος ασφαλείας, τα επιμέρους τμήματα του οποίου αναλύθηκαν στα προηγούμενα υποκεφάλαια.

```
#include <Sleep_n0m1.h>
#include "HX711.h"
#include "SIM900.h"
#include <SoftwareSerial.h>
#include "sms.h"

#define DOUT  5
#define CLK  4

HX711 scale(DOUT, CLK);
SMSGSM sms;

void gsmpower() {
  digitalWrite(GSM_ON, HIGH);
  delay(1200);
  digitalWrite(GSM_ON, LOW);
  delay(10000);
}

float calibration_factor = 15405;
float weight;
float prev_weight = 0;
String strweight;
String strsms;

int numdata;
boolean started = false;
boolean theft = false;
char smsbuffer[160];
char n[20];

char adminphone[] = "+306949419635";
```

```
char phone_number[] = "00000000000000";
char sms_position;
char sms_text[100];
char insms_text[100];
int i;
int y=12;
int t = 0;
Sleep sleep;
unsigned long sleepTime;

void setup()
{
  sleepTime = 300000;
  scale.set_scale();
  scale.tare();
  gsm.begin(9600);
  sms.SendSMS("00306949419635", "Scale Ready. Place the beehive");
  gsmpower();
  delay(30000);
  scale.set_scale(calibration_factor);

};

void loop()
{
  //Because load cells are not linear, 5 measurments//
  //are taken with the correct calibration factor//
  for ( i = 0; i < 5; i = i + 1) {
    weight = scale.get_units();
    if ( weight <= 1) {
      calibration_factor = 15685;
    }
    else if (weight <= 4) {
      calibration_factor = 15405;
    }
    else if (weight <= 5) {
      calibration_factor = 15105;
    }
    else if (weight <= 6) {
      calibration_factor = 15445;
    }
    else if (weight <= 7) {
      calibration_factor = 15215;
    }
    else if (weight <= 8) {
      calibration_factor = 14885;
    }
    else if (weight <= 9) {
      calibration_factor = 15055;
    }
    else if (weight <= 10) {
      calibration_factor = 14745;
```

```
    }
    else if (weight <= 11) {
      calibration_factor = 14715;
    }
    else if (weight <= 12) {
      calibration_factor = 14635;
    }
    else if (weight <= 13) {
      calibration_factor = 14555;
    }
    else if (weight <= 14) {
      calibration_factor = 14835;
    }
    else calibration_factor = 14150;
    scale.set_scale(calibration_factor);
    weight = scale.get_units();
  }
  //-------------------------------//

  //float weight is converted to string and combined//
  //with a text for the sms (The weight of the beehive is: xx)//
  strweight = String(weight);
  strsms = "The weight of the beehive is: " + strweight;
  strsms.toCharArray(sms_text, 100);
  //-----------------------------//

  //SMS is send with the current status of the beehive (every 24hs)//
  if (t == 286) {
    delay(10000);
    gsm.begin(9600);
    delay(1000);
    sms.SendSMS("003069XXXXXXXX", sms_text);
    delay(1000);
    gsmpower();
    t = 0;
  }
  //--------------------------//

  //Check if there is incoming SMS and reply if it is from the owner//
  if (t == y) {
    gsm.begin(9600);
    delay(10000);
    sms_position = sms.IsSMSPresent(SMS_UNREAD);
    delay(1000);
    if (sms_position)
    {
      sms.GetSMS(sms_position, phone_number, insms_text, 100);
      delay(500);

      //if SMS is from owner, reply.//
      if (strcmp( adminphone, phone_number ) == 0 ) {
        sms.SendSMS("003069XXXXXXXX", sms_text);
```

```
      }
      if (gsm.begin(9600)) {
       for (i = 1; i <= 20; i++) sms.DeleteSMS(i); //delete all messages//
      }
    }
    gsmpower();
    delay(500);
    y=y+12;
    if (y>=274) y=12;
  }
  //----------------------------//

  //Check if the beehive is stolen and alert the owner via SMS//
  if (prev_weight - weight > 2) {
    theft = true;
    while (true) {
      if (theft) {
        gsm.begin(9600);
        started = true;
        if (started = true) {
          sms.SendSMS("003069XXXXXXXX", "ATTENTION! BEEHIVE STOLEN!");
          theft = false;
        }
        gsmpower();
      }
    }
  }
  prev_weight = weight;
  //----------------------------//

  //Enter power saving mode//
  scale.power_down(); //HX711 power down//
  delay(1000);
  sleep.pwrDownMode(); //Arduino power down//
  sleep.sleepDelay(sleepTime);
  delay(100);
  scale.power_up(); //HX711 power up//
  //----------------------------//
  t = t + 1;
};
```

5 Οικονομική Ανάλυση

Ένα από τα σημαντικότερα χαρακτηριστικά του απομακρυσμένου συστήματος ασφαλείας και παρακολούθησης που κατασκευάστηκε, είναι το χαμηλό του κόστος. Τα ηλεκτρονικά εξαρτήματα που χρησιμοποιήθηκαν για την κατασκευή του πρωτοτύπου είναι σε μεγάλο βαθμό αυτά που θα χρησιμοποιηθούν και για την προς παραγωγή κατασκευή και, όπως φαίνεται και στον παρακάτω πίνακα, το κόστος για την κατασκευή του ήταν πολύ μικρό.

Ηλεκτρονικά		
Εξάρτημα	Πωλητής	Τιμή
Arduino Uno R3	Ebay	25 €
HX711	Ebay	2.5 €
Load Cells (Δυναμοκυψέλες) x4	Ebay	10 €
GSM Shield	Ebay	20 €
Καλώδια, Θερμοσυστελλόμενα, Καλάι, κτλ	Τοπικά καταστήματα ηλεκτρονικών	2 €
	Σύνολο	59.5 €

Το μηχανολογικό κόστος για την κατασκευή του συστήματος ήταν μηδενικό, καθώς χρησιμοποιήθηκαν ανακυκλωμένα ξύλα. Ενδεικτικά, παρουσιάζονται τιμές για το κόστος της μηχανολογικής κατασκευής.

Μηχανολογικά		
Εξάρτημα	Πωλητής	Τιμή
Ξυλεία	Τοπικό κατάστημα ξυλείας	8 €
Ξυλόκολλα, καρφιά, βίδες	Τοπικό κατάστημα υλικών/εξαρτημάτων	10 €
Λαμαρίνα	Τοπικό κατάστημα σιδηρικών	5-10 €
Αλφάδι	Τοπικό κατάστημα υλικών/εξαρτημάτων	3 €
	Σύνολο	31 €

Γενικό Σύνολο: 90.50 €

Οι παραπάνω είναι τιμές λιανικής. Εύκολα καταλαβαίνει κανείς πως το κόστος για μαζική παραγωγή του απομακρυσμένου συστήματος ασφαλείας και παρακολούθησης κυψελών θα είναι πολύ μικρότερο.

6 Μελλοντικές Προοπτικές

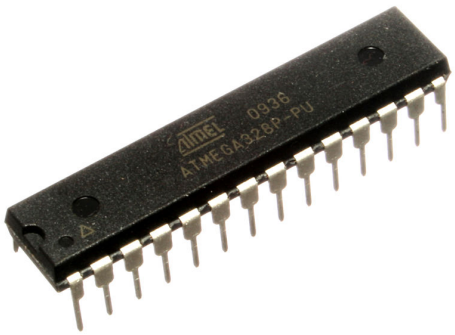
Η παρούσα πτυχιακή εργασία είναι ένα πρωτότυπο σύστημα ασφαλείας για κυψέλες μελιού.

Παρακάτω παρουσιάζονται διάφορες αναβαθμίσεις και προσθήκες στην λειτουργία του που θα μπορούσαν να εφαρμοστούν μελλοντικά. Αυτές είναι είτε βελτιώσεις στην απόδοση και τη λειτουργία των χαρακτηριστικών του, είτε προσθήκες νέων λειτουργιών.

6.1 Μεγαλύτερη εξοικονόμηση ενέργειας

Στην παρούσα πτυχιακή εργασία χρησιμοποιήθηκε ο Arduino board Uno R3 με τον μικροελεγκτή της ATMEL ATmega328p για εκπαιδευτικούς, καθώς και οικονομικούς λόγους. Ο Arduino Uno παρέχει έτοιμη την πλακέτα ανάπτυξης (USB connector, external power supply jack, pins έτοιμα προς χρήση, κ.ά.), κάτι το οποίο ήταν άκρως βοηθητικό κατά τη διαδικασία εκμάθησης.

Δυστυχώς όμως, όλο αυτό κοστίζει σε κατανάλωση ενέργειας στο σύστημα, καθώς ο Arduino έχει ελάχιστη ηλεκτρική κατανάλωση 50mA. Όταν το παρόν σύστημα τίθεται σε λειτουργία εξοικονόμησης ενέργειας, η μέση κατανάλωση του μειώνεται στο 1.7mA. Παρόλα αυτά, υπάρχει η δυνατότητα πολύ μεγαλύτερης βελτίωσης.



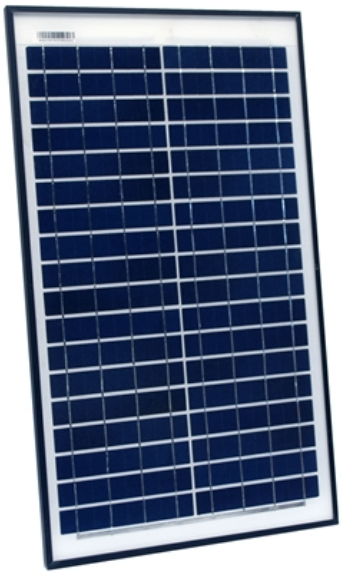
Σχήμα 6.1: Ο μικροελεγκτής ATmega328p

Κατασκευάζοντας και χρησιμοποιώντας το κατάλληλο κύκλωμα μόνο με τον μικροελεγκτή της ATMEL ATmega328p, χωρίς τα περιττά περιφερειακά του Arduino Uno, η μέση κατανάλωση του συστήματος ασφαλείας μπορεί να μειωθεί στην τάξη των 10μA. Με τη μείωση της ηλεκτρικής κατανάλωσης σε αυτές τις τιμές, η διάρκεια ζωής της μπαταρίας

μπορεί να παραταθεί, βελτιώνοντας έτσι την αυτονομία του συστήματος σε τουλάχιστον έξι μήνες έως και ένα χρόνο.

6.2 Αυτόνομο σύστημα τροφοδοσίας με ηλιακή ενέργεια

Παρόλο που το σύστημα έχει ήδη πολύ καλή αυτονομία και, όπως αναφέρθηκε και στο υποκεφάλαιο 6.1, μπορεί να αυξηθεί σημαντικά, το σύστημα εξακολουθεί να μην είναι πλήρως αυτόνομο. Αν, παραδείγματος χάριν, οι κυψέλες είναι τοποθετημένες σε κάποια ορεινή περιοχή με κακή κάλυψη δικτύου κινητής τηλεφωνίας, τότε το σύστημα θα «δυσκολεύεται» και θα καθυστερεί να συνδεθεί στο δίκτυο κάθε φορά που θα προσπαθεί να επικοινωνήσει με τον μελισσοκόμο. Αυτό θα έχει ως αποτέλεσμα η κατανάλωση να αυξάνεται δραματικά και η διάρκεια των μπαταριών να μειώνεται σημαντικά.



Σχήμα 6.2: Ηλιακό πάνελ

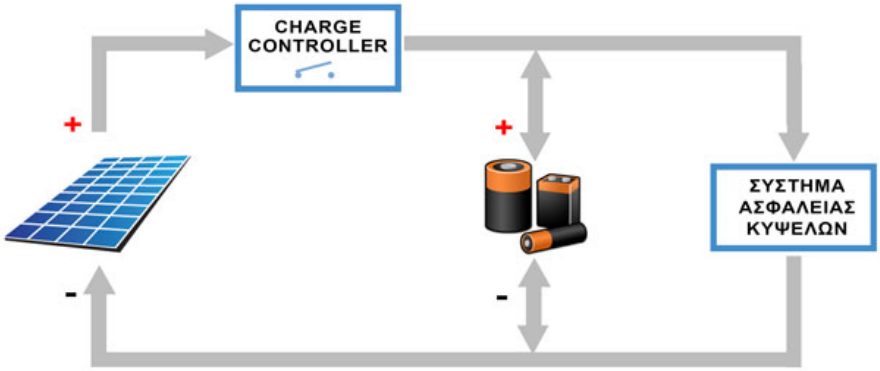
Το πρόβλημα αυτό μπορεί να λυθεί, χρησιμοποιώντας ηλιακή ενέργεια για την επαναφόρτιση των μπαταριών στην διάρκεια της ημέρας. Για να επιτευχθεί αυτό χρειάζονται τα εξής εξαρτήματα:

- Ηλιακό Πάνελ. Το ηλιακό πάνελ θα πρέπει να έχει τάση εξόδου 15-20% μεγαλύτερη από αυτή της μπαταρίας διότι αποδίδει πάντα λιγότερο από την αναγραφόμενη από τον κατασκευαστή τάση (αν, για παράδειγμα είναι εν μέρη υπό σκιά). Στην περίπτωσή μας, το σύστημά έχει τροφοδοσία 6V, οπότε το πάνελ θα πρέπει να αποδίδει

τουλάχιστον 7V στο κύκλωμα. Επίσης, για να γίνει με ασφάλεια η φόρτιση μιας μπαταρίας, πρέπει ο ρυθμός με τον οποίο φορτίζει να είναι στο ένα δέκατο της συνολικής χωρητικότητας της μπαταρίας, δηλαδή 800mA μέγιστο, καθώς το σύστημα χρησιμοποιεί μπαταρίες με χωρητικότητα 8000mA.

- Επαναφορτιζόμενες μπαταρίες. Όπως είναι φυσικό, θα πρέπει να αντικατασταθούν οι αλκαλικές μπαταρίες του συστήματος με επαναφορτιζόμενες.

- Ελεγκτή φόρτισης. Ο ελεγκτής φόρτισης, όταν επιτευχθεί η πλήρης φόρτιση των μπαταριών, διακόπτει το κύκλωμα μεταξύ του ηλιακού πάνελ και των μπαταριών αποτρέποντας έτσι την πιθανή καταστροφή αυτών.



Σχήμα 6.3: Κυκλωματική διάταξη συστήματος φόρτισης μπαταριών

Το παραπάνω αυτόνομο σύστημα τροφοδοσίας, ενώ μπορεί να προσφέρει πολύ μεγαλύτερη αυτονομία στο σύστημα ασφαλείας και παρακολούθησης, έχει και κάποια μειονεκτήματα.

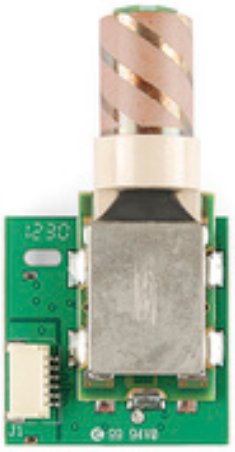
Το σημαντικότερο εξ αυτών είναι ότι αυξάνει πολύ σημαντικά το κόστος. Για να αντισταθμιστεί αυτή η αύξηση, το παραπάνω σύστημα θα ήταν προτιμότερο να χρησιμοποιηθεί ως σταθμός τροφοδοσίας, για περισσότερα από ένα συστήματα ασφαλείας ταυτόχρονα.

Το δεύτερο μειονέκτημα είναι η αδυναμία τοποθέτησης του συστήματος τροφοδοσίας, με τέτοιο τρόπο, ώστε να μην είναι εύκολα αντιληπτό από έναν επίδοξο κλέφτη. Το ηλιακό πάνελ πρέπει να είναι φανερό για να μπορεί να το χτυπάει ο ήλιος. Όπως όμως προαναφέρθηκε, ένας μελισσοκόμος, ο οποίος έχει στην κατοχή του πολλά μελίσσια (π.χ. πάνω από 150) δεν μπορεί να εγκαταστήσει συστήματα ασφαλείας σε όλα καθώς το κόστος θα είναι δυσβάσταχτο. Για το λόγο αυτό, συνήθως οι παραγωγοί εγκαθιστούν το σύστημα ασφαλείας σε λίγες κυψέλες, στρατηγικά τοποθετημένες ανάμεσα στις υπόλοιπες. Αυτό σημαίνει ότι δεν πρέπει να είναι εμφανές, ποιες έχουν σύστημα ασφαλείας και ποιες όχι,

για προφανείς λόγους. Ένα ηλιακό πάνελ δίπλα ή πάνω σε μία μόνο κυψέλη θα έκανε φανερό πως έχει σύστημα ασφαλείας εγκατεστημένο σε αυτή και συνεπώς, θα το καθιστούσε και άχρηστο.

Το τρίτο και τελευταίο πιθανό μειονέκτημα ενός αυτόνομου συστήματος τροφοδοσίας με ηλιακή ενέργεια είναι η διάρκεια ζωής της μπαταρίας. Όπως είναι γνωστό, η διάρκεια ζωής των επαναφορτιζόμενων μπαταριών μειώνεται όσο επαναφορτίζονται. Αυτό σημαίνει ότι η διάρκεια ζωής τους θα είναι πολύ μειωμένη, καθώς θα επαναφορτίζονται καθημερινά ανεξάρτητα από την κατανάλωση τους, κάνοντας το παραπάνω σύστημα ακόμα και ζημιογόνο σε κάποιες περιπτώσεις, αντί να το βελτιώσει (για παράδειγμα, αν χρειάζονται οι ακριβές επαναφορτιζόμενες μπαταρίες αντικατάσταση κάθε χρόνο, ενώ οι αλκαλικές μπαταρίες κάθε οχτώ μήνες με κόστος πέντε φορές μικρότερο).

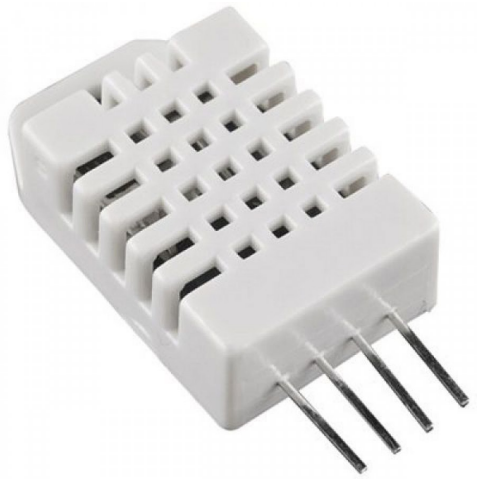
6.3 GPS



Σχήμα 6.4: Μία τυπική μονάδα GPS με κωνική κεραία

Η σημαντικότερη βελτίωση που μπορεί να γίνει στο υπάρχον πρωτότυπο σύστημα ασφαλείας είναι η προσθήκη GPS στην κυψέλη. Έτσι, σε περίπτωση που κλαπεί, θα ειδοποιηθεί ο μελισσοκόμος και θα ενεργοποιηθεί το GPS στην κυψέλη. Ακόμα και αν δεν προλάβει να αποτρέψει την κλοπή θα μπορεί εύκολα να βρει την κυψέλη του ακολουθώντας το στίγμα του GPS. Στο υπάρχον πρωτότυπο έχει ήδη γίνει πρόβλεψη γι' αυτή την προσθήκη στον κώδικα.

6.4 Αισθητήρας υγρασίας και θερμοκρασίας



Σχήμα 6.5: Ένας τυπικός αισθητήρας υγρασίας και θερμοκρασίας (*DHT22*)

Μια τελευταία βελτίωση που θα μπορούσε να γίνει στο σύστημα είναι η προσθήκη δύο ακόμη αισθητήρων, θερμοκρασίας και υγρασίας. Έτσι, ο μελισσοκόμος θα μπορεί να συλλέγει περισσότερα δεδομένα για την κατάσταση της κυψέλης του και να βγάλει πολύτιμα, αλλά και πιο ακριβή και σωστά συμπεράσματα για την υγεία της. Για παράδειγμα, αν η υγρασία είναι πολύ υψηλή μέσα στην κυψέλη πιθανώς να σημαίνει ότι έχει νερό μέσα ή αν η θερμοκρασία πέσει κάτω από τους 32 βαθμούς Κελσίου, τότε το μελίσσι είναι πολύ αδύναμο και χρειάζεται την άμεση επέμβαση του παραγωγού.

Αναφορές

[1] Henri Clement. *Σύγχρονη Μελισσοκομία.* Editions Rustica, Paris, 2002.

[2] Arduino. *Arduino Uno REV3.* `https://store.arduino.cc/usa/arduino-uno-rev3`

[3] ATMEL *ATmega328p datasheet.* San Jose CA, USA, 2015. www.atmel.com/ATmega328Pdatasheet

[4] Wikipedia. *Load Cell.* `https://en.wikipedia.org/wiki/Load_cell`

[5] GJCC. *Δυναμοκυψέλες: Τι είναι, κατηγορίες και το φαινόμενο ringing.* http://gjcc.gr/δυναμοκυψέλες-τι-είναι-κατηγορίες-κα/

[6] AVIA Semiconductor. *HX711 datasheet.* `https://www.mouser.com/ds/2/813/hx711_english-1022875.pdf`

[7] Bogde. *HX711 library.* Github. `https://github.com/bogde/HX711`

[8] John G. Webster. *The Measurement, Instrumentation and Sensors Handbook.* CRC Press LTD, Boca Raton FL, 1999.

[9] Hardy Process Solutions. *Load Cell Calibration Curve.* http://www.hardysolutions.com/scale-calibration-in-a-qms/

[10] Arduino. *GSM Library.* `https://www.arduino.cc/en/Reference/GSM`

[11] n0m1. *Sleep_n0m1. Arduino Library to place the arduino into sleep mode for a specific length of time, or a specific number of sleep cycles.* Github. `https://github.com/n0m1/Sleep_n0m1`