

ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ
ΣΧΟΛΗ ΕΠΙΣΤΗΜΩΝ ΑΓΩΓΗΣ
ΠΑΙΔΑΓΩΓΙΚΟ ΤΜΗΜΑ ΔΗΜΟΤΙΚΗΣ
ΕΚΠΑΙΔΕΥΣΗΣ

Δημοσθένης Ε. Μπολανάκης

«Διερεύνηση και αξιολόγηση εκπαιδευτικών
πειραμάτων Φυσικής από απόσταση με τη
χρήση ασύρματων δικτύων»

Διδακτορική διατριβή

ΙΩΑΝΝΙΝΑ 2016
Δημοσθένης Ε. Μπολανάκης

«Διερεύνηση και αξιολόγηση εκπαιδευτικών πειραμάτων Φυσικής από απόσταση με τη χρήση ασύρματων δικτύων»

Διδακτορική διατριβή
που υποβλήθηκε στο Παιδαγωγικό Τμήμα Δημοτικής Εκπαίδευσης
της Σχολής Επιστημών Αγωγής
του Πανεπιστημίου Ιωαννίνων

Τριμελής Συμβουλευτική Επιτροπή

1. Κωνσταντίνος Θ. Κώτσης, Καθηγητής, Παιδαγωγικό Τμήμα Δημοτικής Εκπαίδευσης, Σχολή Επιστημών Αγωγής, Παν/μιο Ιωαννίνων, Επιβλέπων
2. Αναστάσιος Μικρόπουλος, Καθηγητής, Παιδαγωγικό Τμήμα Δημοτικής Εκπαίδευσης, Σχολή Επιστημών Αγωγής, Παν/μιο Ιωαννίνων
3. Θεόδωρος Λαόπουλος, Καθηγητής, Τμήμα Φυσικής, Σχολή Θετικών Επιστημών, Αριστοτέλειο Παν/μιο Θεσσαλονίκης

Επταμελής Εξεταστική Επιτροπή

1. Κωνσταντίνος Θ. Κώτσης, Καθηγητής, Παιδαγωγικό Τμήμα Δημοτικής Εκπαίδευσης, Σχολή Επιστημών Αγωγής, Παν/μιο Ιωαννίνων, Επιβλέπων
2. Αναστάσιος Μικρόπουλος, Καθηγητής, Παιδαγωγικό Τμήμα Δημοτικής Εκπαίδευσης, Σχολή Επιστημών Αγωγής, Παν/μιο Ιωαννίνων
3. Θεόδωρος Λαόπουλος, Καθηγητής, Τμήμα Φυσικής, Σχολή Θετικών Επιστημών, Αριστοτέλειο Παν/μιο Θεσσαλονίκης
4. Γεώργιος Α. Ευαγγελάκης, Καθηγητής, Τμήμα Φυσικής, Σχολή Θετικών Επιστημών, Παν/μιο Ιωαννίνων
5. Διονύσιος Βαβουγιός, Καθηγητής, Παιδαγωγικό Τμήμα Ειδικής Αγωγής, Σχολή Ανθρωπιστικών και Κοινωνικών Επιστημών, Παν/μιο Θεσσαλίας
6. Αικατερίνη Πλακίτση, Αναπληρώτρια Καθηγήτρια, Παιδαγωγικό Τμήμα Νηπιαγωγών, Σχολή Επιστημών Αγωγής, Παν/μιο Ιωαννίνων
7. Κώστας Γαβριλάκης, Επίκουρος Καθηγητής, Παιδαγωγικό Τμήμα Δημοτικής Εκπαίδευσης, Σχολή Επιστημών Αγωγής, Παν/μιο Ιωαννίνων

Η έγκριση της διδακτορικής διατριβής από το Παιδαγωγικό Τμήμα Δημοτικής Εκπαίδευσης της Σχολής Επιστημών Αγωγής του Πανεπιστημίου Ιωαννίνων δεν υποδηλώνει αποδοχή των απόψεων του συγγραφέα

(ν. 5343/32, άρθρο 202, §2)

*"One thing about trains:
It doesn't matter where they're going.
What matters is deciding to get on."*

*Train conductor,
The Polar Express*

Πρόλογος

Στη διεθνή βιβλιογραφία γίνεται συχνά λόγος για την ανεκτίμητη αξία του πειράματος στη διδασκαλία της Φυσικής, όσον αφορά την οικοδόμηση αντίληψης των διδασκομένων για τη βαθύτερη κατανόηση των φυσικών εννοιών [1-6]. Η ανάπτυξη εκπαιδευτικών πειραμάτων αποτελεί μία πολύπλοκη διεργασία, καθότι ο ερευνητής θα πρέπει γνωρίζει πολύ καλά τη διεθνή βιβλιογραφία ώστε να προβεί σε δόκιμη επιλογή του αντικειμένου διδασκαλίας που πρόκειται να εισάγει στο εργαστήριο, εφαρμόζοντας καινοτομίες τόσο σε εκπαιδευτικό επίπεδο, όσο και σε επίπεδο μηχανικής. Στη δεύτερη περίπτωση, η ανάπτυξη και υλοποίηση του εκπαιδευτικού εξοπλισμού θα πρέπει να διασφαλίζει την ορθή λειτουργία και ομαλή διεξαγωγή του πειράματος, καθώς και να προωθεί τη διαδικασία εκμάθησης του γνωστικού αντικειμένου στους διδασκόμενους.

Στη διεθνή βιβλιογραφία συναντώνται αρκετά παραδείγματα υλοποίησης εξοπλισμού απομακρυσμένης πρόσβασης σε εκπαιδευτικά πειράματα [7-13], τα οποία προορίζονται να συμπληρώσουν τις αδυναμίες του παραδοσιακού πειραματισμού. Τα πλεονεκτήματα του εξ αποστάσεως πειραματισμού δεν περιορίζονται μόνο στη δυνατότητα μελέτης από απόσταση, επεκτείνονται και στη δυνατότητα του εξ αποστάσεως ελέγχου πειραμάτων κατά τη διάρκεια παρουσίας των διδασκομένων στην τάξη [14], μέσω του δικτύου/διαδικτύου λόγω χάριν, μειώνοντας σημαντικά το κόστος εργαστηριακού εξοπλισμού. Στην παρούσα διδακτορική διατριβή πραγματοποιήθηκε ανάπτυξη, υλοποίηση και αξιολόγηση καινοτόμου εκπαιδευτικού εξοπλισμού που υποστηρίζει την εξ αποστάσεως πρόσβαση σε μετρήσεις Φυσικών μεγεθών και συγκεκριμένα ατμοσφαιρικής πίεσης, με τη χρήση του δικτύου/διαδικτύου. Η υλοποίηση του εξοπλισμού βασίστηκε στην ανάπτυξη ασύρματου δικτύου αισθητήρων τεχνολογίας MEMS και έχει εφαρμογή στη βαρομετρική υψομετρία.

Τα *Μικρο-Ηλεκτρομηχανικά Συστήματα* (*Micro-Electromechanical Systems – MEMS*) αποτελούν τεχνολογία αιχμής, η οποία φέρει τα χαρακτηριστικά μικρής συσκευασίας, χαμηλού κόστους και κατανάλωσης. Η τεχνολογία MEMS ενσωματώνει μία μηχανική λειτουργία, στην αλλαγή της οποίας παράγεται ηλεκτρικό σήμα. Λόγω της διεπιστημονικής φύσης των MEMS (επιστήμη των υλικών, συστήματα ελέγχου, αισθητήρες, σχεδιασμός ολοκληρωμένων συστημάτων, κ.ά.), η τεχνολογία αυτή έχει επιφέρει νέα αναπτυξιακή δυναμική στο βιομηχανικό αυτοματισμό [15] και έχει επηρεάσει σημαντικά και την εκπαίδευση, όπου πλέον συναντώνται αρκετά περιγράμματα σπουδών με αντικείμενο μελέτης τα MEMS [16-18].

Ακολουθώντας μια βιβλιογραφική ανασκόπηση στα πειράματα Φυσικής από απόσταση [19] και κατόπιν συνοψίζοντας τους επικρατέστερους τρόπους υλοποίησης εξοπλισμού για την εξ αποστάσεως συλλογή Φυσικών μεγεθών [20], στα πλαίσια της διδακτορικής διατριβής πραγματοποιήθηκε αρχικά αξιολόγηση των μετρήσεων και μεθοδολογιών που χρησιμοποιούνται στη βαρομετρική υψομετρία με τη χρήση του πρωτότυπου ερευνητικού εξοπλισμού [21,22]. Εν συνέχεια, πραγματοποιήθηκε αποτίμηση των προτεινόμενων εκπαιδευτικών πειραμάτων Φυσικής και του αντίκτυπου αυτών στην εκμάθηση των φοιτητών [23,24]. Επίσης προτάθηκε μία εναλλακτική μορφή υλοποίησης του παρόντος εξοπλισμού για την υποστήριξη της εργαστηριακής εκπαίδευσης ατόμων με κινητική αναπηρία [25]. Τέλος, αποτιμήθηκαν οι δυνατότητες που προσφέρουν οι σύγχρονες ψηφιακές τεχνολογίες στην πειραματική διδασκαλία της Φυσικής [26], καθώς και η διεπιστημονική έρευνα που απαιτείται για την υλοποίηση καινοτόμων πειραμάτων εργαστηριακής εκπαίδευσης σε ένα πεδίο που βρίσκεται ακόμη υπό διερεύνηση από την επιστημονική κοινότητα [27]. Η διατριβή εκπονήθηκε υπό την καθοδήγηση του Καθηγητή του Παιδαγωγικού Τμήματος Δημοτικής Εκπαίδευσης (ΠΤΔΕ) του Πανεπιστημίου Ιωαννίνων, κ. Κωνσταντίνο Θ. Κώση, και προορίζεται να καλύψει τις ανάγκες του Εργαστηρίου Εκπαίδευσης και Διδασκαλίας της Φυσικής.

Οργάνωση της διδακτορικής διατριβής

Το βιβλίο αυτό αποτελείται από 3 περιεκτικά κεφάλαια και 2 παραρτήματα, τα υποκεφάλαια και υποπαραρτήματα των οποίων δίδονται στον ακόλουθο πίνακα. Το Κεφάλαιο 1 διερευνά την αξία πειραματισμού στη διδασκαλία της Φυσικής, καθώς και την εφαρμογή των Τεχνολογιών Πληροφορίας και Επικοινωνίας (ΤΠΕ) στη διεξαγωγή πειραμάτων. Το Κεφάλαιο 2 αναλύει τη μεθοδολογία και τον πρωτότυπο εκπαιδευτικό εξοπλισμό που αναπτύχθηκε και εφαρμόστηκε στο Εργαστήριο Εκπαίδευσης και Διδασκαλίας της Φυσικής του ΠΤΔΕ του Πανεπιστημίου Ιωαννίνων, για τη διεξαγωγή πειραμάτων βαρομετρικής υψομετρίας απομακρυσμένης πρόσβασης με ασύρματο δίκτυο αισθητήρων τεχνολογίας MEMS (Micro-Electromechanical Systems). Το Κεφάλαιο 3 παρουσιάζει τα αποτελέσματα της έρευνας που διεξήχθη στα πλαίσια της διδακτορικής διατριβής. Τέλος, τα Παραρτήματα Α και Β παρέχουν έναν οδηγό εγκατάστασης και χρήσης του πρωτότυπου εκπαιδευτικού εξοπλισμού, καθώς και ανάλυση των μετρήσεων ατμοσφαιρικής πίεσης με τη χρήση των προγραμμάτων που αναπτύχθηκαν σε Matlab, ενώ το παράρτημα Γ παρουσιάζει το συνολικό κώδικα του συστήματος.

Κεφάλαιο/Παράρτημα	Υποκεφάλαιο/Υποπαραρτήμα
Κεφάλαιο 1: Διδασκαλία φυσικής και αξία πειραμάτων	- Εισαγωγή στη διδασκαλία της φυσικής και στην αξία του πειραματισμού - Εφαρμογές ΤΠΕ στη διεξαγωγή πειραμάτων - Αναφορές
Κεφάλαιο 2: Πειράματα βαρομετρικής υψομετρίας	- Εισαγωγή στους αισθητήρες MEMS - Μετρήσεις απόλυτου ύψους και θεωρία βαρομετρικής υψομετρίας - Πρωτότυπο σύστημα βαρομετρικής υψομετρίας - Αναφορές
Κεφάλαιο 3: Έρευνα και εκπαιδευτική αξιολόγηση πειραμάτων	- Αποτίμηση απλής και διαφορικής βαρομετρικής υψομετρίας - Αποτίμηση εκπαιδευτικών πειραμάτων - Αναφορές
Κεφάλαιο 4: Συμπεράσματα και μελ- λοντικές προοπτικές	- Επιστημονική έρευνα - Εκπαιδευτική έρευνα - Προοπτικές συνέχειας της έρευνας - Αναφορές
Παράρτημα Α: Οδηγός εγκατάστασης και χρήσης του συστήμα- τος	- Τρόποι λειτουργίας του συστήματος - Λειτουργία συλλογής μετρήσεων μέσω ηλεκτρονικού υπολογιστή - Λειτουργία συλλογής μετρήσεων μέσω δικτύου/διαδικτύου - Αναβάθμιση υλικολογισμικού στις συσκευές του συστήματος - Αναφορές

Κεφάλαιο/Παράρτημα	Υποκεφάλαιο/Υποπαράρτημα
Παράρτημα Β: Οδηγός ανάλυσης δεδο- μένων στο Matlab	• Μετρήσεις ατμοσφαιρικής πίεσης • Ανάλυση μετρήσεων διαφορικής βαρομετρικής υψομετρίας • Ανάλυση μετρήσεων απλής βαρομετρικής υψομετρίας • Μετρήσεις σε αεροστεγές δοχείο • Ανάλυση μετρήσεων σε αεροστεγές δοχείο • Αναφορές
Παράρτημα Γ: Κώδικας	• Υλικολογισμικό συστήματος • Λογισμικό συστήματος • Matlab κώδικας • Αναφορές

Ευχαριστίες

Στο σημείο αυτό θέλω να ευχαριστήσω τον επιβλέποντα της διατριβής μου, Καθηγητή κ. Κωνστανίνο Κώτση, για την πολύτιμη βοήθειά του και την εύστοχη καθοδήγησή στα κρίσιμα σημεία που αντιμετωπίστηκαν κατά την εκπόνηση της διατριβής, αλλά και την εμπιστοσύνη και ενθάρρυνσή του για αυτενέργεια και ανάληψη πρωτοβουλιών όπου κρίθηκε απαραίτητο. Ακόμη, ένα μεγάλο ευχαριστώ και για την ανοχή και στήριξή του, σε χρονικές περιόδους όπου έπρεπε να ανασταλεί η ενασχόληση με τη διδακτορική διατριβή για οικογενειακούς λόγους. Είμαι επίσης ιδιαίτερα ευγνώμων στα υπόλοιπα μέλη της τριμελούς επιτροπής, τους Καθηγητές κ. Αναστάσιο Μικρόπουλο και κ. Θεόδωρο Λαόπουλο για τη στήριξη και την εμπιστοσύνη που μου δείξαν, αλλά και τις πολύτιμες υποδείξεις τους. Ιδιαίτερα θα ήθελα να ευχαριστήσω τον συνεργάτη από τα παλιά, Καθηγητή κ. Θεόδωρο Λαόπουλο, για τις – ανεκτίμητης αξίας – συζητήσεις μας, καθώς και τις πολύτιμες ιδέες που μου έδωσε πάνω στο γνωστικό αντικείμενο των ηλεκτρονικών. Θερμές ευχαριστίες και στους εκπαιδευτικούς του Πειραματικού Σχολείου Πανεπιστημίου Θεσσαλονίκης Ελένη Πετρίδου και Σαπφώ Φωτιάδου, για την παροχή πολύτιμων δεδομένων αξιολόγησης από προσωπική τους εκπαιδευτική έρευνα. Τέλος, ένα μεγάλο ευχαριστώ στη γυναίκα μου Κατερίνα και στο μικρό μου γιο Μανώλη, για την ανεκτικότητα τους, και τις προσωπικές στιγμές που στερηθήκαμε προκειμένου να εκπληρωθεί η εκπόνηση της διατριβής μου.

Αναφορές

- [1] Γ. Στύλος, Κ.Θ. Κώτσης, και Α. Εμβαλωτής, “Πρακτικές εκπαιδευτικών Πρωτοβάθμιας Εκπαίδευσης στη διδασκαλία της Φυσικής (Α’ Μέρος)”, *Φυσικές Επιστήμες στην Εκπαίδευση*, τεύχος 5, σελ. 7-15, 2014.
- [2] Κ.Θ. Κώτσης, “Διδασκαλία της Φυσικής και Πείραμα”, *Πανεπιστήμιο Ιωαννίνων*, 2005, ISBN: 960-233-167-4.
- [3] I.T. Koponen and Terhi Mäntylä, “Generative role of experiments in physics and in teaching physics: a suggestion for epistemological reconstruction”, *Science & Education*, vol. 15, issue 1, pp. 31-54, 2006.
- [4] J. Lavonen, J. Jauhainen, I.T. Koponen, and K. Kurki-Suonio “Effect of a long-term in-service training program on teachers' beliefs about the role of experiments in physics education”, *International Journal of Science Education*, vol. 26, issue 3, pp. 309-329, 2004.
- [5] R. Trumper, “The physics laboratory – a historical overview and future perspectives”, *Science & Education*, vol. 12, issue 7, pp. 645-670, 2003.
- [6] E. Etkina, A. Heuvelen, D.T. Brookes, and D. Mills, “Role of experiments in physics instruction – a process approach”, *The Physics Teacher*, vol. 40, issue 6, pp. 351-355, 2002.
- [7] A. Saxena¹, M.M. Singh, and V. Singh³, “The state of art of MEMS in automation industries”, *Proc. of Innovative Trends in Applied Physical, Chemical, Mathematical Sciences and Emerging Energy Technology for Sustainable Development (APCMET '14)*, pp. 1-5, 2014.
- [8] Y. Lai and S.A. Bazaz, “Engineering curriculum development in microsystems”, *Proc. of the Canadian Engineering Education Association*, pp. 145-149, 2006.
- [9] L. Lin, “Curriculum development in microelectromechanical systems in mechanical engineering”, *IEEE Transactions on Learning Technologies*, vol. 44, no. 1, pp. 61-66, Feb 2001.
- [10] R.J. Pryputniewicz, “Current trends and future directions in MEMS”, *Experimental Mechanics*, vol. 52, no. 3, pp. 289-303, Mar 2012.
- [11] B. Fallery, S. Gerbaix, and R. Ologeanu, “Videotraining: a comparison between virtual class and remote class”, *International Symposium on Collaborative Technologies and Systems*, Irvine CA, 2008, pp. 440-445.
- [12] B. Hanson, P. Culmer, J. Gallagher, K. Page, E. Read, A. Weightman, and M. Levesley, “Remote laboratories in the curriculum”, *11th IASTED International on Conference Computers and Advanced Technology in Education*, Crete, 2008, pp. 160-165, 2008.
- [13] V. Koliass, I. Anagnostopoulos, and E. Kayafas, “Remote experiments in education: A survey over different platforms and application fields”, *11th International Conference on Optimization of Electrical and Electronic Equipment*, Brasov, 2008, pp. 181-188.
- [14] C. Gravier, J. Fayolle, B. Bayard, M. Ates, and J. Lardon, “Remote laboratories: proposed guidelines. 2nd International Conference on Digital Information Management”, Lyon France, 2007, pp. 786-792.
- [15] M.M.D. Amadou, M. Saad, J.P. Kenne, and V. Nerguizian, “Virtual and remote laboratories”, *1st IEEE International Conference on E-Learning in Industrial Electronics*, Hammamet, 2006, pp. 173-176.

- [16] A. Baccigalupi, C. De Capua, and A. Liccardo, "Overview on development of remote teaching laboratories: from labview to web services", *Instrumentation and Measurement Technology Conference, Sorrento*, 2006, pp. 992-997.
- [17] D.Z. Deniz, A. Bulancak, and G. Ozcan, "A novel approach to remote laboratories", *33rd ASEE/IEEE Frontiers in Education Annual Conference, Boulder CO*, 2003, pp. T3E/8-T3E/12.
- [18] M. Cooper, and J.M.M. Ferreira, "Remote laboratories extending access to science and engineering curricular", *IEEE Transactions on Learning Technologies*, 2(4), pp. 342-353, 2009.
- [19] Δ. Ε. Μπολανάκης και Κ. Κώτσης, "Χρήση Πειραμάτων Απομακρυσμένης Πρόσβασης με Ασύρματη Επικοινωνία στη Διδασκαλία Φυσικής: ανασκόπηση βιβλιογραφίας και προτάσεις", *6ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση*, Φλώρινα, 2009, σελ. 579-586.
- [20] Δ. Ε. Μπολανάκης και Κ. Κώτσης, "Απομακρυσμένος Έλεγχος Μετρήσεων Φυσικών Μεγεθών Πραγματικού Χρόνου: δυνατότητες και περιορισμοί", *9ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση*, Θεσσαλονίκη, 2015, σελ. 232-236.
- [21] D. E. Bolanakis, K. T. Kotsis, and T. Laopoulos, "A Prototype Wireless Sensor Network System for a Comparative Evaluation of Differential and Absolute Barometric Altimetry", *IEEE Aerospace and Electronic Systems Magazine*, 30(11), pp. 20-28, 2015.
- [22] D. E. Bolanakis, K. T. Kotsis, and T. Laopoulos, "Temperature Influence on Differential Barometric Altitude Measurements", *The 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS'2015)*, Warsaw, Poland, 2015, pp. 120-124.
- [23] D. E. Bolanakis, K. T. Kotsis, T. Laopoulos, "Work in Progress: Experiments on MEMS Barometric Pressure Sensors", *3ο Πανελλήνιο Συνέδριο Ηλεκτρονικής και Τηλεπικοινωνιών (PACET 2015)*, Ιωάννινα, 2015, σελ. 1-3.
- [24] D. E. Bolanakis, K. T. Kotsis, and T. Laopoulos, "Ethernet & PC-based Experiments on Barometric Altimetry using MEMS in a Wireless Sensor Network", *Computer Application in Engineering Education*, 24(3), pp. 428-442, 2016.
- [25] Δ. Ε. Μπολανάκης και Κ. Κώτσης, "Πείραμα για τη μέτρηση υψομετρικής διαφοράς: μία πρόταση για άτομα με κινητική αναπηρία", *Πανελλήνιο Συνέδριο στις Διδακτικές Προσεγγίσεις και Πειραματική Διδασκαλία στις Φυσικές Επιστήμες*, Θεσσαλονίκη, 2016, σελ. 205-213.
- [26] Κ. Κώτσης και Δ. Ε. Μπολανάκης, "Η συμβολή των σύγχρονων ψηφιακών τεχνολογιών σε πειράματα βαρομετρικής πίεσης: Μελέτη περίπτωσης", *10ο Πανελλήνιο και Διεθνές Συνέδριο «Οι ΤΠΕ στην Εκπαίδευση»*, Ιωάννινα, 2016, σελ. 1-8.
- [27] D. E. Bolanakis, K. T. Kotsis, and T. Laopoulos, "Interdisciplinary research towards the development of innovative experimentation on barometric altimetry", under review.

Περιεχόμενα

Πρόλογος	7
Οργάνωση της διδακτορικής διατριβής	9
Ευχαριστίες	11
Αναφορές	12
Διδασκαλία φυσικής και αξία πειραμάτων	21
Εισαγωγή στη διδασκαλία της φυσικής και στην αξία του πειραματισμού	22
Εφαρμογές ΤΠΕ στη διεξαγωγή πειραμάτων	23
Αναφορές	30
Πειράματα βαρομετρικής υψομετρίας	33
Εισαγωγή στους αισθητήρες MEMS	34
Μετρήσεις απόλυτου ύψους και θεωρία βαρομετρικής υψομετρίας	36
Πρωτότυπο σύστημα βαρομετρικής υψομετρίας	40
Αναφορές	55
Έρευνα και εκπαιδευτική αξιολόγηση πειραμάτων	57
Αποτίμηση απλής και διαφορικής βαρομετρικής υψομετρίας	59
Αποτίμηση εκπαιδευτικών πειραμάτων	74
Αναφορές	87
Συμπεράσματα και μελλοντικές προοπτικές	89
Επιστημονική έρευνα	91
Εκπαιδευτική έρευνα	93
Προοπτικές συνέχειας της έρευνας	94
Αναφορές	95
Οδηγός εγκατάστασης και χρήσης του συστήματος	97
Τρόποι λειτουργίας του συστήματος	98
Λειτουργία συλλογής μετρήσεων μέσω ηλεκτρονικού υπολογιστή	99
Λειτουργία συλλογής μετρήσεων μέσω δικτύου/διαδικτύου	106
Αναβάθμιση υλικολογισμικού στις συσκευές του συστήματος	107
Αναφορές	110
Οδηγός ανάλυσης δεδομένων στο Matlab	111
Μετρήσεις ατμοσφαιρικής πίεσης	112
Ανάλυση μετρήσεων διαφορικής βαρομετρικής υψομετρίας	115

Ανάλυση μετρήσεων απλής βαρομετρικής υψομετρίας	127
Μετρήσεις σε αεροστεγές δοχείο	134
Ανάλυση μετρήσεων σε αεροστεγές δοχείο	136
Αναφορές	140
Κώδικας	141
Υλικολογισμικό συστήματος	142
Λογισμικό συστήματος	216
Matlab κώδικας	221
Αναφορές	230

Πίνακες

Πίνακας 1—1	Δυνατότητες/περιορισμοί υποδομών μετάδοσης πληροφορίας	27
Πίνακας 2—1	Αισθητήρες βαρομετρικής πίεσης διαφόρων κατασκευαστών	35
Πίνακας 2—2	Κατανάλωση ρεύματος των καρτών της συσκευής μέτρησης	44
Πίνακας 3—1	Σφάλμα απόλυτου ύψους	66
Πίνακας 3—2	Απόκλιση στο σήμα εξόδου δύο αισθητήρων σε διάφορες πιέσεις	68
Πίνακας 3—3	Στατιστική ανάλυση μετρήσεων και σφάλμα απόλυτου ύψους	73
Πίνακας 3—4	Ερωτηματολόγιο αξιολόγησης	75
Πίνακας 3—5	Αποτίμηση κατανόησης των διδαχθέντων εννοιών	81
Πίνακας 3—6	Βαθμολογία φοιτητών στο τεστ αξιολόγησης	83
Πίνακας 4—1	Πειράματα βαρομετρικής υψομετρίας	93

Εικόνες

Εικόνα 1—1 Συλλογή μετρήσεων μέσω του δικτύου κινητής τηλεφωνίας	25
Εικόνα 1—2 Συλλογή μετρήσεων μέσω διαδικτύου	25
Εικόνα 1—3 Διάγραμμα βαθμίδων πρωτότυπης διάταξης	28
Εικόνα 2—1 Μέθοδος μέτρησης βαρομετρικού υψομέτρου	38
Εικόνα 2—2 Συσκευή του συντονιστή συστήματος	40
Εικόνα 2—3 Συσκευή μέτρησης της ατμοσφαιρικής πίεσης	41
Εικόνα 2—4 Συσκευή μέτρησης (διαδικασία φόρτισης)	42
Εικόνα 2—5 Διάγραμμα διασυνδέσεων της συσκευής συντονιστή	42
Εικόνα 2—6 Διάγραμμα διασυνδέσεων των συσκευών μέτρησης	43
Εικόνα 2—7 Πρωτόκολλο επικοινωνίας των συσκευών του συστήματος	44
Εικόνα 2—8 Διάγραμμα ροής υλικολογισμικού του συντονιστή	46
Εικόνα 2—9 Διάγραμμα ροής υλικολογισμικού των συσκευών μέτρησης	47
Εικόνα 2—10 Περιβάλλον λήψης μετρήσεων μέσω δικτύου/διαδικτύου	50
Εικόνα 2—11 Περιβάλλον λήψης μετρήσεων μέσω ηλεκτρονικού υπολογιστή	50
Εικόνα 2—12 Γραφικός προγραμματισμός στο Labview	52
Εικόνα 2—13 Λήψη μετρήσεων από τη γραμμή εντολών H/Y	53
Εικόνα 2—14 Αρχείο καταγραφής μετρήσεων στον H/Y	54
Εικόνα 2—15 Λήψη μετρήσεων μέσω δικτύου/διαδικτύου (4 συσκευές)	54
Εικόνα 3—1 Μετρήσεις εσωτερικού χώρου σε $Y=0\mu$	59
Εικόνα 3—2 Μετρήσεις εσωτερικού χώρου σε $Y=4,7\mu$	60
Εικόνα 3—3 Ανάλυση μετρήσεων εσωτερικού χώρου (τεχνική ΔΒΥ)	61
Εικόνα 3—4 Ανάλυση μετρήσεων εσωτερικού χώρου (τεχνική ΑΒΥ)	62
Εικόνα 3—5 Μετρήσεις εξωτερικού χώρου σε $Y=6,62\mu$	63
Εικόνα 3—6 Ανάλυση μετρήσεων εξωτερικού χώρου (τεχνική ΔΒΥ)	64
Εικόνα 3—7 Ανάλυση μετρήσεων εξωτερικού χώρου (τεχνική ΑΒΥ)	65
Εικόνα 3—8 Μετρήσεις σε αεροστεγές δοχείο	66
Εικόνα 3—9 Ανάλυση μετρήσεων σε αεροστεγές δοχείο	67
Εικόνα 3—10 Μετρήσεις σε ύψος $2,55\mu$ (50 δείγματα)	69
Εικόνα 3—11 Μετρήσεις σε ύψος $2,26\mu$ (200 δείγματα)	70
Εικόνα 3—12 Μετρήσεις σε ύψος $1,63\mu$ (8 δείγματα)	70
Εικόνα 3—13 Εργαστήριο Διδασκαλίας Φ.Ε. (ΠΤΔΕ) & Μετρήσεις Φοιτητών	78
Εικόνα 3—14 Αυτόνομο σύστημα υπολογισμού βαρομετρικού υψομέτρου	84
Εικόνα 3—15 Πειραματική διάταξη βαρομετρικής υψομετρίας για ΑΜΕΑ	85
Εικόνα 4—1 Αρχιτεκτονική πρωτότυπου συστήματος βαρομετρικής υψομετρίας	91
Εικόνα Α—1 Εγκατάσταση του MCHPFSUSB_Setup_v1.2.exe (Βήμα 1)	99
Εικόνα Α—2 Εγκατάσταση του MCHPFSUSB_Setup_v1.2.exe (Βήμα 2)	99
Εικόνα Α—3 Εγκατάσταση του MCHPFSUSB_Setup_v1.2.exe (Βήμα 3)	100
Εικόνα Α—4 Εγκατάσταση του MCHPFSUSB_Setup_v1.2.exe (Βήμα 4)	100
Εικόνα Α—5 Οδηγός επικοινωνίας του PIC18F26J50 με τη θύρα USB	100
Εικόνα Α—6 Σύνδεση συντονιστή με ηλεκτρονικό υπολογιστή	101
Εικόνα Α—7 Σύνδεση συντονιστή με H/Y (αναδυόμενο μήνυμα)	101
Εικόνα Α—8 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 1)	101
Εικόνα Α—9 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 2)	102
Εικόνα Α—10 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 3)	102

Εικόνα Α—11 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 4)	102
Εικόνα Α—12 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 5)	103
Εικόνα Α—13 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 6)	103
Εικόνα Α—14 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 7)	103
Εικόνα Α—15 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 8)	104
Εικόνα Α—16 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 9)	104
Εικόνα Α—17 Διαχείριση συσκευών των Windows: η συσκευή USB	104
Εικόνα Α—18 Το πρόγραμμα συλλογής μετρήσεων RemoteLab.exe	105
Εικόνα Α—19 Συλλογή μετρήσεων από το διαδίκτυο μέσω του Chrome	106
Εικόνα Α—20 Κάρτες της συσκευής του συντονιστή	107
Εικόνα Α—21 Εκκίνηση του αρχείου program-mcu.exe	108
Εικόνα Α—22 Κάρτες της συσκευής μέτρησης	109
Εικόνα Β—1 Μορφή καταγραφής δεδομένων σε αρχείο κειμένου	112
Εικόνα Β—2 Μετακίνηση της 1 ^{ης} συσκευής στα 1,1μ (μετρήσεις 101-200)	113
Εικόνα Β—3 Μετακίνηση της 2 ^{ης} συσκευής στα 1,1μ (μετρήσεις 101-200)	113
Εικόνα Β—4 Εισαγωγή των αρχείων μετρήσεων και εντολών στο Matlab	115
Εικόνα Β—5 Μετρήσεις ατμοσφαιρικής πίεσης σε ύψος $Y=0\mu$	116
Εικόνα Β—6 Μετρήσεις ατμοσφαιρικής πίεσης σε ύψος $Y=0\mu$ (διόρθωση P1)	117
Εικόνα Β—7 Μετρήσεις ατμοσφαιρικής πίεσης σε ύψος $Y=1,1\mu$	118
Εικόνα Β—8 Μετρήσεις ατμοσφαιρικής πίεσης σε ύψος $Y=1,1\mu$ (διόρθωση Pa)	120
Εικόνα Β—9 Γραφική ύψους (υπολογισμός με διεθνή βαρομετρική εξίσωση)	121
Εικόνα Β—10 Κατανομή δειγμάτων ύψους (διεθνής βαρομετρική εξίσωση)	122
Εικόνα Β—11 Γραφική ύψους (υπολογισμός με υψομετρική εξίσωση)	124
Εικόνα Β—12 Κατανομή δειγμάτων ύψους (υψομετρική εξίσωση)	125
Εικόνα Β—13 Μετρήσεις θερμοκρασίας σταθερής/κινητής συσκευής (T1/T2)	126
Εικόνα Β—14 Μετρήσεις ατμοσφαιρικής πίεσης κινητής συσκευής (P1/Pa)	127
Εικόνα Β—15 Γραφική ύψους (υπολογισμός με διεθνή βαρομετρική εξίσωση)	128
Εικόνα Β—16 Κατανομή δειγμάτων ύψους (διεθνής βαρομετρική εξίσωση)	129
Εικόνα Β—17 Γραφική ύψους (υπολογισμός με υψομετρική εξίσωση)	130
Εικόνα Β—18 Κατανομή δειγμάτων ύψους (υψομετρική εξίσωση)	131
Εικόνα Β—19 Μετρήσεις θερμοκρασίας κινητής συσκευής στα 0μ/1,1μ (T1/Ta)	132
Εικόνα Β—20 Μεταβολές ατμοσφαιρικής πίεσης	133
Εικόνα Β—21 Καταγραφή μετρήσεων σε αεροστεγές δοχείο	134
Εικόνα Β—22 Μορφή καταγραφής μετρήσεων σε αεροστεγές δοχείο	135
Εικόνα Β—23 Γραφική ενός αρχείου μετρήσεων (αεροστεγές δοχείο)	137
Εικόνα Β—24 Γραφική όλων των αρχείων μετρήσεων (αεροστεγές δοχείο)	138

Τύποι

Τύπος 2—1 Υψομετρική εξίσωση	36
Τύπος 2—2 Λύση υψομετρικής εξίσωσης ως προς z	37
Τύπος 2—3 Υπολογισμός απόλυτου ύψους με την υψομετρική εξίσωση	37
Τύπος 2—4 Διεθνής βαρομετρική εξίσωση	37
Τύπος 2—5 Υπολογισμός απόλυτου ύψους με τη διεθνή βαρομετρική εξίσωση	38
Τύπος 3—1 Υπολογισμός σφάλματος απόλυτου ύψους	66
Τύπος 3—2 Διεθνής βαρομετρική και υψομετρική εξίσωση	68
Τύπος 3—3 Υπολογισμός τυπικής απόκλισης s με τη συνάρτηση (Matlab) std	72
Τύπος 3—4 Διάστημα εμπιστοσύνης για τη μέση τιμή πληθυσμού μ	72
Τύπος 3—5 Υπολογισμός όγκου αέρα σε αεροστεγές δοχείο	83
Τύπος B—1 Διεθνής βαρομετρική εξίσωση	121
Τύπος B—2 Τυπική απόκλιση	123
Τύπος B—3 Υψομετρική εξίσωση	124

Διδασκαλία φυσικής και αξία πειραμάτων

Η Φυσική, η επιστήμη που διερευνά τη συμπεριφορά του σύμπαντος, χρησιμοποιεί το «ισχυρό» εργαλείο του πειράματος για την παρατήρηση και ανάλυση των φαινομένων της φύσης. Οι ραγδαίες τεχνολογικές εξελίξεις των τελευταίων δεκαετιών και η εξάπλωση των Τεχνολογιών Πληροφοριών και Επικοινωνίας (ΤΠΕ) στο χώρο της εκπαίδευσης, έχουν επιφέρει σημαντικές αλλαγές στη διεξαγωγή πειράματος ως μέσω διδασκαλίας της Φυσικής. Στο κεφάλαιο αυτό διερευνώνται οι νέες τάσεις υλοποίησης και διεξαγωγής πειραμάτων με τη χρήση των ΤΠΕ, δίνοντας έμφαση στην ανάπτυξη πειραμάτων απομακρυσμένης πρόσβασης που είναι και το αντικείμενο μελέτης της παρούσας διδακτορικής διατριβής.

1

ΣΤΟ ΚΕΦΑΛΑΙΟ ΑΥΤΟ

ΕΙΣΑΓΩΓΗ ΣΤΗ ΔΙΔΑΣΚΑΛΙΑ ΤΗΣ ΦΥΣΙΚΗΣ ΚΑΙ ΣΤΗΝ ΑΞΙΑ ΤΟΥ ΠΕΙΡΑΜΑΤΙΣΜΟΥ

Γίνεται μια εισαγωγή στις έννοιες Επιστήμης της Φυσικής, Διδακτική της Φυσικής, Πείραμα, και επισημαίνεται η αξία πειράματος στη διδασκαλία της Φυσικής μέσα από βιβλιογραφικές αναφορές, όπως κατατέθηκαν στο 6ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση, Φλώρινα, 2009.

ΕΦΑΡΜΟΓΕΣ ΤΠΕ ΣΤΗ ΔΙΕΞΑΓΩΓΗ ΠΕΙΡΑΜΑΤΩΝ ΦΥΣΙΚΗΣ

Διερευνώνται οι δυνατότητες και περιορισμοί πειραμάτων Φυσικής σε α) πραγματικό εργαστήριο, β) εικονικό εργαστήριο, και γ) πραγματικό εργαστήριο απομακρυσμένης πρόσβασης, με ιδιαίτερη έμφαση στην τελευταία κατηγορία, όπως κατατέθηκαν στο 9ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση, Θεσσαλονίκη, 2015.

Εισαγωγή στη διδασκαλία της φυσικής και στην αξία του πειραματισμού

Η Φυσική είναι η επιστήμη που συνδέεται με την παρατήρηση των φαινομένων της φύσης, δηλαδή με τη διερεύνηση τους και την ανάλυση των παραγόντων που επιδρούν και ασκούν επιρροή στα φαινόμενα, με απώτερο σκοπό την κατανόηση της συμπεριφοράς του σύμπαντος. Το «ισχυρό» εργαλείο του Φυσικού για την ανάλυση των φαινομένων της φύσης είναι το πείραμα. Σύμφωνα με τη Βικιπαίδεια, ως Πείραμα χαρακτηρίζεται η οποιαδήποτε έμπρακτη εφαρμογή της θεωρίας και γενικά ο κάθε έλεγχος της θεωρητικής γνώσης [1]. Η σύγχρονη επιστήμη της Φυσικής, στις μέρες μας, είναι συνυφασμένη με τη στενή συνεργασία θεωρίας και πειράματος.

Σύμφωνα με τα παραπάνω, ένα από τα ζητούμενα της σύγχρονης Διδακτικής της Φυσικής (δηλαδή του επιστημονικού κλάδου που διερευνά τη βελτίωση διδασκαλίας της τελευταίας) αποτελεί η διερεύνηση του ρόλου και αξίας του πειράματος ως μέρος της διδακτικής μεθοδολογίας και πρακτικής, με τη θέση του πειράματος να ορίζεται ιδιαίτερη σημαντική όσον αφορά την οικοδόμηση αντίληψης των διδασκομένων για τη βαθύτερη κατανόηση των φυσικών εννοιών [2-7]. Στις μέρες μας, η διδασκαλία της Φυσικής (αλλά και γενικότερα των Φυσικών Επιστημών) τείνει να προσανατολιστεί προς τη άμεση συσχέτιση διεξαγωγής πειραμάτων με τη χρήση Τεχνολογιών Πληροφοριών και Επικοινωνίας (ΤΠΕ) [8]. Η ανάπτυξη και υλοποίηση εργαστηρίων με τη χρήση των ΤΠΕ αποτελεί αντικείμενο μελέτης των σύγχρονων ερευνών [9,10].

Στη διδακτορική αυτή διατριβή αναπτύχθηκε και υλοποιήθηκε πρωτότυπη διάταξη πειραματισμού στη βαρομετρική υψομετρία με τη χρήση σύγχρονων εφαρμογών ΤΠΕ, όπως η υλοποίηση ασύρματου δικτύου αισθητήρων και διάταξης λήψης μετρήσεων μέσω δικτύου/διαδικτύου με τη χρήση ενσωματωμένου εξυπηρετητή (*embedded web server*). Αξίζει να σημειωθεί ότι, στη διεθνή βιβλιογραφία δεν εντοπίζεται σύστημα που να υλοποιεί πειραματική διάταξη βαρομετρικής υψομετρίας για εκπαιδευτικούς σκοπούς.

Εφαρμογές ΤΠΕ στη διεξαγωγή πειραμάτων

Οι ραγδαίες τεχνολογικές εξελίξεις των τελευταίων δεκαετιών σε συνδυασμό με τη ραγδαία εξάπλωση των ΤΠΕ στο χώρο της εκπαίδευσης, έχουν επιφέρει σημαντικές αλλαγές στη διεξαγωγή πειράματος στην διδασκαλία της Φυσικής. Στις μέρες μας εντοπίζονται δύο μεγάλες κατηγορίες εφαρμογής των ΤΠΕ στη διεξαγωγή πειραμάτων, που αποσκοπούν είτε στον εμπλουτισμό του παραδοσιακού δασκαλοκεντρικού συστήματος εργαστηριακής εκπαίδευσης, είτε ακόμη και στην αντικατάσταση του τελευταίου. Οι δύο αυτές κατηγορίες είναι οι ακόλουθες:

1. Η ανάπτυξη και υλοποίηση εικονικών πειραμάτων (προσομοιώσεις πειραματικής διαδικασίας).
2. Η ανάπτυξη και υλοποίηση εξοπλισμού απομακρυσμένης πρόσβασης σε πειράματα (εικονικά ή πραγματικά).

Τα πλεονεκτήματα υλοποίησης των δύο αυτών εφαρμογών ΤΠΕ στην εργαστηριακή εκπαίδευση της Φυσικής [11-18] (και όχι μόνο [19,20]) συνοψίζονται σε τρεις κυρίαρχες κατηγορίες:

- α) στη μείωση του κόστους του ακριβού, σε πολλές περιπτώσεις, εξοπλισμού που απαιτούνται στα πειράματα Φυσικής,
- β) στην παροχή ευελιξίας όσον αφορά το χρόνο οργάνωσης και εκτέλεσης πειραμάτων, που επιβαρύνεται συχνά με τον απαιτούμενο χρόνο εκμάθησης χειρισμού του εργαστηριακού εξοπλισμού,
- γ) στην αποφυγή έκθεσης των διδασκόμενων σε κινδύνους (όπως π.χ., στην έκθεση ακτινοβολίας [17]).

Στη διεθνή βιβλιογραφία υποστηρίζεται πολλές φορές η άποψη ότι η επαφή των διδασκόμενων με πραγματικό εργαστηριακό εξοπλισμό είναι η αποτελεσματικότερη μέθοδος συσχέτισης των θεωρητικών εννοιών με τις πραγματικές συνθήκες [21]. Δεν μπορούν όμως να αμφισβητηθούν οι περιπτώσεις βιβλιογραφικών ανασκοπήσεων που εντοπίζουν αρκετά παραδείγματα συγκριτικών μελετών όπου τα εικονικά πειράματα δεν υστερούν, ή και ακόμη υπερτερούν έναντι των πραγματικών [22].

Το βασικό πλεονέκτημα του πραγματικού εργαστηρίου έναντι του εικονικού, είναι ότι επιτρέπει στους διδασκόμενους να αλληλεπιδρούν πειραματικά με υλικά και μοντέλα [23,24] και να διερευνούν γεγονότα, εντός του εργαστηρίου ή εξ αποστάσεως, αλλάζοντας παραμέτρους, ενεργοποιώντας διαδικασίες και παρατηρώντας τα αποτελέσματα των ενεργειών τους [25]. Στις περιπτώσεις απομακρυσμένης πρόσβασης σε εργαστηριακό εξοπλισμό, είναι γεγονός ότι απουσιάζει αφενός η άμεση αλληλεπίδραση των εκπαιδευόμενων με τον διδάσκοντα και αφετέρου, περιορίζεται η οπτική επαφή με τον εργαστηριακό εξοπλισμό, η οποία στην αποδοτικότερη εφαρμογή της πραγματοποιείται μέσω κάμερας που αποδίδει συγκεκριμένη εικόνα του πειράματος [26].

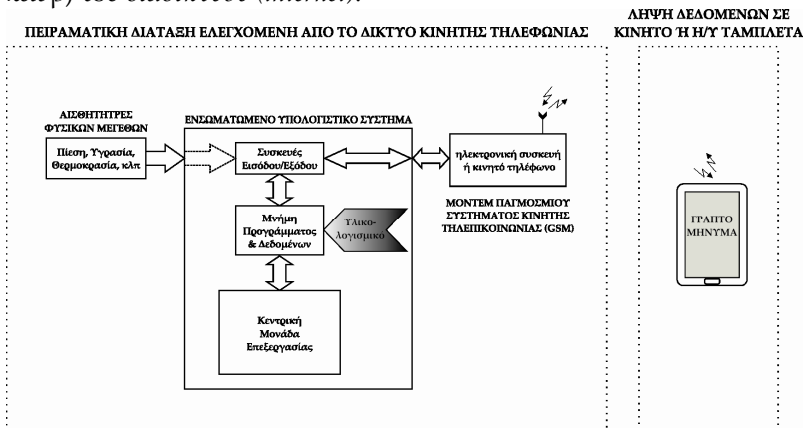
Τα πλεονεκτήματα του εξ αποστάσεως ελέγχου πειραμάτων δεν περιορίζονται μόνο στη δυνατότητα μελέτης από απόσταση, αλλά επεκτείνονται και στη δυνατότητα απομακρυσμένου ελέγχου πειραμάτων κατά τη διάρκεια παρουσίας των διδασκομένων στην τάξη [27]. Στο Εργαστήριο Εκπαίδευσης και Διδασκαλίας της Φυσικής του Παιδαγωγικού Τμήματος Δημοτικής Εκπαίδευσης (ΠΤΔΕ) του Πανεπιστημίου Ιωαννίνων αναπτύχθηκε και υλοποιήθηκε σύστημα απομακρυσμένης πρόσβασης πραγματικών πειραμάτων, το οποίο χρησιμοποιείται κατά την παρουσία των διδασκομένων στο εργαστήριο. Το σύστημα χρησιμοποιείται για την καταγραφή μετρήσεων ατμοσφαιρικής πίεσης και έχει εφαρμογή στη βαρομετρική υψομετρία [28,29], ενώ η εξ αποστάσεως συλλογή μετρήσεων πραγματοποιείται μέσω του δικτύου/διαδικτύου χρησιμοποιώντας ένα απλό πρόγραμμα περιήγησης στον Ιστό (π.χ. Internet Explorer, Chrome, κλπ.). Αξίζει να σημειωθεί ότι ο πρωτότυπος αυτός εξοπλισμός διαθέτει εκτός από τον τρόπο λειτουργίας συλλογής μετρήσεων μέσω δικτύου/διαδικτύου για την απομακρυσμένη πρόσβαση των πειραμάτων, και έναν δεύτερο τρόπο λειτουργίας για τη συλλογή μετρήσεων μέσω ηλεκτρονικού υπολογιστή. Η λειτουργία αυτή (σε συνδυασμό με πρότυπους κώδικες ανάλυσης μετρήσεων που αναπτύχθηκαν στο Matlab) υποστηρίζει την επίδειξη των πειραμάτων από το διδάσκοντα [30,31].

Ο λόγος υλοποίησης εργαστηριακού εξοπλισμού απομακρυσμένης πρόσβασης πραγματικών πειραμάτων που χρησιμοποιείται κατά την παρουσία των διδασκομένων στο εργαστήριο, έγκειται αφενός στη μείωση κόστους του εργαστηριακού εξοπλισμού (η συλλογή μετρήσεων από τους διδασκόμενους μέσω του δικτύου/διαδικτύου επιτρέπει την εγκατάσταση ενός μόνο συστήματος ανά εργαστήριο), και αφετέρου στην εστίαση της προσοχής των διδασκομένων στη ανάλυση των μετρήσεων και την επίλυση του προβλήματος (που είναι και το ζητούμενο της εκπαιδευτικής διαδικασίας). Στη διεθνή βιβλιογραφία αναφέρεται συχνά η ανάγκη αποφυγής αποπροσανατολισμού των διδασκομένων κατά τη διεξαγωγή πειραμάτων, λόγω των δυσκολιών εκμάθησης χειρισμού ενός πραγματικού εργαστηριακού εξοπλισμού [32,33].

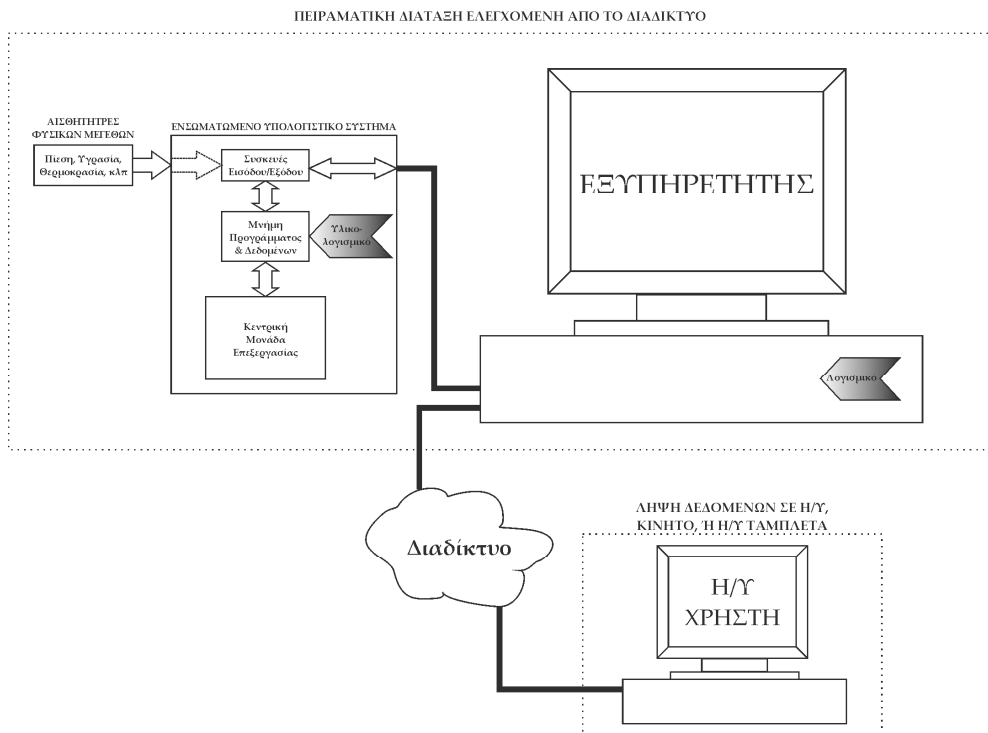
Στη συνέχεια του κεφαλαίου δίνονται οι γενικές αρχές σχεδίασης συστημάτων απομακρυσμένης πρόσβασης σε πραγματικά πειράματα, όπως παρουσιάστηκαν στο 9ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση [34]. Η απομακρυσμένη πρόσβαση στον εργαστηριακό εξοπλισμό πραγματοποιείται με την εκμετάλλευση των επικρατέστερων υποδομών μετάδοσης πληροφορίας, συνοψίζοντας τις δυνατότητες και περιορισμούς χρήσης σε κάθε περίπτωση. Οι επικρατέστερες υποδομές μετάδοσης πληροφορίας στις μέρες μας είναι δύο: α) το δίκτυο κινητής τηλεφωνίας και β) το διαδίκτυο.

Γενικές αρχές σχεδίασης εξοπλισμού απομακρυσμένης πρόσβασης σε πραγματικά πειράματα μέσω του δικτύου κινητής τηλεφωνίας και του διαδικτύου

Στις Εικόνες 1—1 και 1—2 δίνονται τα διαγράμματα βαθμίδων των συστημάτων απομακρυσμένης πρόσβασης σε εργαστηριακό εξοπλισμό μέσω α) της Υπηρεσίας Γραπτών Μηνυμάτων (ΥΓΜ) του Παγκόσμιου Συστήματος Κινητής Τηλεπικοινωνίας (ΠΣΚΤ) και β) του διαδικτύου (internet).



Εικόνα 1—1 Συλλογή μετρήσεων μέσω του δικτύου κινητής τηλεφωνίας



Εικόνα 1—2 Συλλογή μετρήσεων μέσω διαδικτύου

Στην 1^η περίπτωση, το σύστημα αποτελείται από τους αισθητήρες μετρήσεων φυσικών μεγεθών, τον ενσωματωμένο υπολογιστή και τη συσκευή μόντεμ. Το ενσωματωμένο υπολογιστικό σύστημα διαχειρίζεται τον έλεγχο του εξοπλισμού απομακρυσμένη πρόσβασης. Ο σχεδιαστής του συστήματος αναπτύσσει το υλικολογισμικό του ενσωματωμένου υπολογιστή που λαμβάνει την πληροφορία από τους αισθητήρες και αποστέλλει τις μετρήσεις μέσω του μόντεμ (με τη μορφή γραπτού μηνύματος). Το γραπτό μήνυμα λαμβάνεται από το δίκτυο κινητής τηλεφωνίας σε ηλεκτρονικό υπολογιστή (H/Y) ή κινητό τηλέφωνο. Το αίτημα λήψης μετρήσεων δίδεται και πάλι με τη μορφή γραπτού μηνύματος, το οποίο λαμβάνεται από το μόντεμ, αποκωδικοποιείται από τον ενσωματωμένο υπολογιστή και ενεργοποιείται η διαδικασία λήψης νέων μετρήσεων.

Στη 2^η περίπτωση, ο έλεγχος της πειραματικής διάταξης μοιράζεται μεταξύ του H/Y και του ενσωματωμένου υπολογιστή. Το αίτημα λήψης μετρήσεων δίδεται από τον H/Y χρήστη που συνδέεται στο δίκτυο/διαδίκτυο και επικοινωνεί με την πειραματική διάταξη μέσω της διεύθυνσης διαδικτύου (IP) της τελευταίας. Ο H/Y λαμβάνει την πληροφορία και επικοινωνεί με το ενσωματωμένο υπολογιστικό σύστημα μέσω κοινής συσκευής για την είσοδο/εξόδο δεδομένων (π.χ. USB). Ο ενσωματωμένος υπολογιστής επεξεργάζεται και αποστέλλει τα δεδομένα των αισθητήρων στον H/Y. Ο τελευταίος προωθεί με τη σειρά τις μετρήσεις στον H/Y χρήστη, δηλαδή είτε σε ένα H/Y (ή H/Y ταμπλέτα), είτε σε ένα κινητό με σύνδεση στο διαδίκτυο. Ο σχεδιαστής του συστήματος αναπτύσσει το υλικολογισμικό του ενσωματωμένου υπολογιστή καθώς και το λογισμικό που θέτει τον H/Y σε λειτουργία εξυπηρετητή και είναι υπεύθυνο για την επικοινωνία H/Y-ενσωματωμένου συστήματος. Μια παραλλαγή αυτής της διάταξης είναι δυνατή με τη χρήση ενσωματωμένου εξυπηρετητή, όπου ο έλεγχος της πειραματικής διάταξης γίνεται αποκλειστικά από τον ενσωματωμένο υπολογιστή.

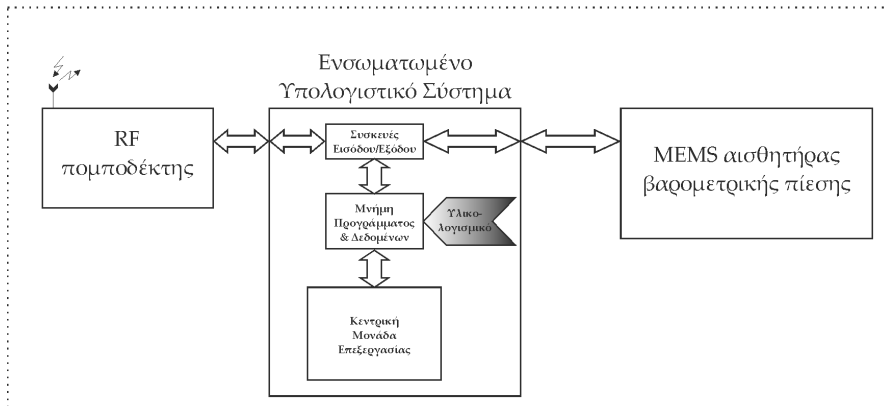
Στον **Πίνακα 1—1** παρουσιάζονται τα πλεονεκτήματα και μειονεκτήματα χρήσης των δύο διατάξεων. Η χρήση διαδικτύου χαρακτηρίζεται από περισσότερες δυνατότητες αλλά και περιορισμούς σε σύγκριση με τη χρήση του δικτύου κινητής τηλεφωνίας. Το μεγαλύτερο πλεονέκτημα του διαδικτύου είναι δυνατότητα αποστολής/λήψης περισσότερων δεδομένων. Ενδεικτικά, ο ηλεκτρονικός υπολογιστής (H/Y) χρήστη μπορεί να λάβει μετρήσεις με γραφική αναπαράσταση αντί απλών αριθμητικών δεδομένων. Από την άλλη, το δίκτυο κινητής τηλεφωνίας χαρακτηρίζεται από περισσότερες πιθανότητες λήψης/αποστολής δεδομένων. Στην περίπτωση αυτή όμως, η αποστολή και λήψη δεδομένων επιβαρύνεται με χρεώσεις των γραπτών μηνυμάτων. Τέλος, η υλοποίηση της πειραματικής διάταξης διαδικτύου με τη χρήση H/Y παρέχει μεγαλύτερη ευελιξία σε σύγκριση με την υλοποίηση ενσωματωμένου εξυπηρετητή, αλλά όπως είναι κατανοητό αυξάνεται σημαντικά το κόστος υλοποίησης.

Πίνακας 1—1 Δυνατότητες/περιορισμοί υποδομών μετάδοσης πληροφορίας

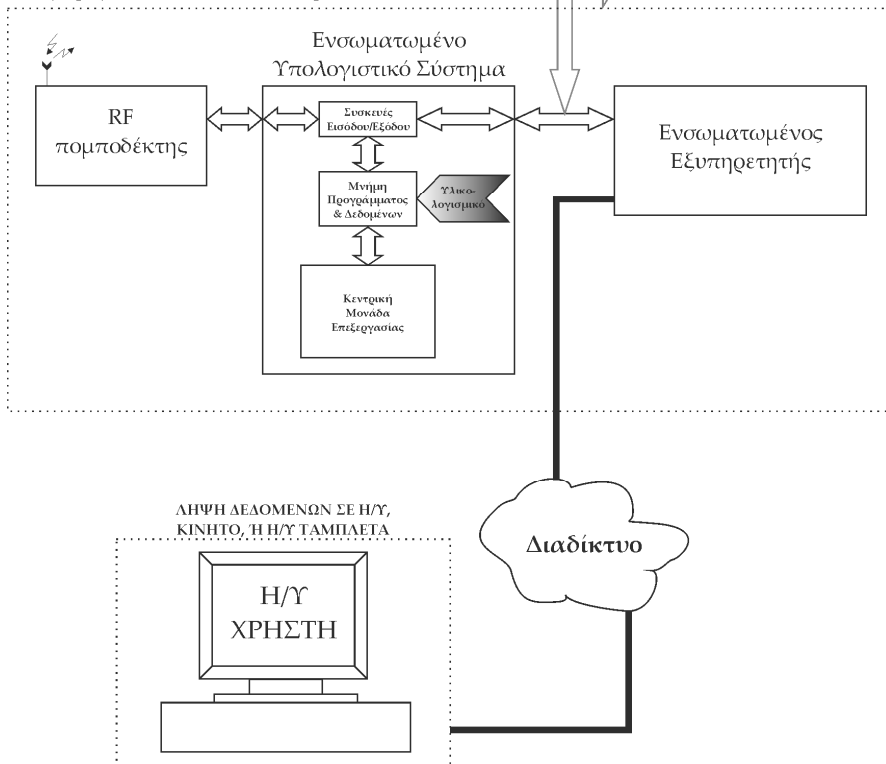
Υποδομή μετάδοσης πληροφορίας	Πλεονεκτήματα	Μειονεκτήματα
Κινητή τηλεφωνία	1. Περισσότερες πιθανότητες λήψης. 2. Χαμηλό κόστος.	1. Κόστος αποστολής sms. 2. Περιορισμένος αριθμός αποστολής/λήψης χαρακτήρων ανά sms.
Διαδίκτυο	1. Ευκολότερη διαχείριση πολλαπλών πελατών στην περίπτωση υλοποίησης της διάταξη με Η/Υ. 2. Δυνατότητα αποστολής πολλών πληροφοριών στους πελάτες και με γραφική αναπαράσταση των μετρήσεων (περιορισμένου, όμως, μεγέθους στην περίπτωση υλοποίησης της διάταξης με ενσωματωμένο εξυπηρετητή). 3. Χαμηλό κόστος υλοποίησης στην περίπτωση χρήσης ενσωματωμένου εξυπηρετητή.	1. Απαίτηση για στατική διεύθυνση διαδικτύου (IP). 2. Αδυναμία σύνδεσης στο διαδίκτυο με υπαιτιότητα του παρόχου της σύνδεσης. 3. Υψηλό κόστος διάταξης στην περίπτωση υλοποίησης με Η/Υ 4. Απαίτηση ανάπτυξης λογισμικού για τη διάταξη με Η/Υ. 5. Περιορισμένος αριθμός σύνδεσης πελατών στην περίπτωση υλοποίησης της διάταξη με ενσωματωμένο εξυπηρετητή.

Στην **Εικόνα 1—3** παρουσιάζεται το διάγραμμα βαθμίδων της πρωτότυπης διάταξης που αναπτύχθηκε για στα πλαίσια της διδακτορικής διατριβής. Η διάταξη έχει υλοποιηθεί με ενσωματωμένο εξυπηρετητή για τη λήψη των μετρήσεων μέσω δικτύου/διαδικτύου. Ο ενσωματωμένος εξυπηρετητής ελέγχεται από τη συσκευή του συστήματος που καλείται συντονιστής (*coordinator*), ενώ υπάρχει δυνατότητα σύνδεσης του τελευταίου με τη θύρα USB ενός ηλεκτρονικού υπολογιστή, για τη λήψη μετρήσεων μέσω προγράμματος Labview (το οποίο αναπτύχθηκε για την δυνατότητα επίδειξης των πειραμάτων από το διδάσκοντα). Η υλοποίηση της διάταξης έχει βασιστεί στην ανάπτυξη ενός ασύρματου δικτύου αισθητήρων, όπου ο συντονιστής του συστήματος λαμβάνει τις μετρήσεις από δύο ανεξάρτητες συσκευές μέτρησης μέσω του σύγχρονου πρωτοκόλλου ασύρματης ζεύξης IEEE 802.15.4. Η υλοποίηση δύο συσκευών μέτρησης πραγματοποιήθηκε για την δυνατότητα μέτρησης απόλυτου ύψους με τη λεγόμενη τεχνική διαφορικής βαρομετρικής υψομετρίας (*differential barometric altimetry*). Σύμφωνα με την τεχνική αυτή, δεδομένα ατμοσφαιρικής πίεσης λαμβάνονται ταυτόχρονα από δύο συσκευές μέτρησης που έχουν τοποθετηθεί στο χώρο σε διαφορά ύψους και κατόπιν, υπολογίζεται η τιμή απόλυτου ύψους μεταξύ των συσκευών.

ΣΥΣΚΕΥΗ ΜΕΤΡΗΣΗΣ



ΣΥΝΤΟΝΙΣΤΗΣ ΣΥΣΤΗΜΑΤΟΣ



Εικόνα 1—3 Διάγραμμα βαθμίδων πρωτότυπης διάταξης

Αξίζει να σημειωθεί ότι οι συσκευές μέτρησης χρησιμοποιούν αισθητήρες τεχνολογίας MEMS (Micro-Electromechanical Systems), οι οποίοι αποτελούν τεχνολογία αιχμής που χαρακτηρίζεται από μία εξαιρετικά μικρή συσκευασία, χαμηλό κόστος, αλλά και χαμηλή κατανάλωση ρεύματος. Η τεχνολογία αυτή έχει επιφέρει νέα αναπτυξιακή δυναμική στο βιομηχανικό αυτοματισμό [35] και τα δίκτυα ασύρματων αισθητήρων [36], έχει όμως επηρεάσει σημαντικά και το χώρο της εκπαίδευσης [37-39].

Αναφορές

- [1] Online: <https://en.wikipedia.org/wiki/Experiment/>.
- [2] Γ. Στύλος, Κ.Θ. Κώτσης, και Α. Εμβλωτής, “Πρακτικές εκπαιδευτικών Πρωτοβάθμιας Εκπαίδευσης στη διδασκαλία της Φυσικής (Α΄ Μέρος)”, *Φυσικές Επιστήμες στην Εκπαίδευση*, τεύχος 5, σελ. 7-15, 2014.
- [3] Κ.Θ. Κώτσης, “Διδασκαλία της Φυσικής και Πείραμα”, *Πανεπιστήμιο Ιωαννίνων*, 2005, ISBN: 960-233-167-4.
- [4] I.T. Koponen and Terhi Mäntylä, “Generative role of experiments in physics and in teaching physics: a suggestion for epistemological reconstruction”, *Science & Education*, vol. 15, issue 1, pp. 31-54, 2006.
- [5] J. Lavonen, J. Jauhainen, I.T. Koponen, and K. Kurki-Suonio “Effect of a long-term in-service training program on teachers' beliefs about the role of experiments in physics education”, *International Journal of Science Education*, vol. 26, issue 3, pp. 309-329, 2004.
- [6] R. Trumper, “The physics laboratory – a historical overview and future perspectives”, *Science & Education*, vol. 12, issue 7, pp. 645-670, 2003.
- [7] E. Etkina, A. Heuvelen, D.T. Brookes, and D. Mills, “Role of experiments in physics instruction – a process approach”, *The Physics Teacher*, vol. 40, issue 6, pp. 351-355, 2002.
- [8] Μ. Πατσαδάκης, “Πειραματική διδασκαλία-μάθηση των Φυσικών Επιστημών και Τεχνολογίες Πληροφορίας & Επικοινωνίας”, *Εκπαιδευτική Επικαιρότητα*, τόμος Α, τεύχος 6, σελ. 26-35, 2015.
- [9] Κ. Μέλη και Δ. Κολιόπουλος, “Κατασκευάζοντας μια εκπαιδευτική υπολογιστική προσομοίωση για τη διδασκαλία του πρώτου θερμοδυναμικού νόμου στις μεταβολές ιδανικών αερίων: επιστημολογικές και διδακτικές διαστάσεις”, *9ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση*, Θεσσαλονίκη, 2015, σελ. 196-200.
- [10] Σ. Αναγνωστάκης και Δ. Σταύρου, “Πειραματικές διατάξεις Φυσικών Επιστημών με εκπαιδευτική ρομποτική”, *9ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση*, Θεσσαλονίκη, 2015, σελ. 219-223.
- [11] Ε. Πετρίδου, Σ. Φωτιάδου, και Κ. Κουκουλάς, “Πραγματικό ή εικονικό εργαστήριο; έρευνα στη Β΄ Γυμνασίου”, *9ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση*, Θεσσαλονίκη, 2015, σελ. 242-246.
- [12] Κ.Θ. Κώτσης και Φ.Β. Ευαγγέλου, “Συγκριτική μελέτη της επίδρασης πραγματικών και εικονικών πειραμάτων στη μάθηση για το φαινόμενο του βρασμού του νερού σε μαθητές Ε΄ και ΣΤ΄ Δημοτικού Σχολείου”, *Θέματα Επιστημών και Τεχνολογίας στην Εκπαίδευση*, 7(1-2), σελ. 5-24, 2014.
- [13] Δ. Ψύλλος, “Μοντέλα και κόσμοι στους εικονικούς χώρους”, *5ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση*, Ιωάννινα, 2007, σελ. 30-41.
- [14] F. Schauer, M. Ozvoldova, and F. Lustig, (2008), “Real remote physics experiments across internet – inherent part of integrated e-learning”, *International Journal of Online Engineering*, vol. 4, no. 2, pp. 52-55, 2008.
- [15] C. Thomsen, H. Scheel, and S. Morgner, “Remote experiments in experimental physics”, *Proc. of ISPRS Tools and Techniques for E-Learning*, Potsdam, 2005, pp. 1-5.

- [16] S. Kocijancic, "Online experiments in physics and technology teaching", *IEEE Transactions on Education*, vol. 45, issue 1, pp. 26-32, 2002.
- [17] H. Lee, S-T Park, and K-C Yuk, "Remote control laboratory for physics experiments via the internet", *Journal of the Korean Physical Society*, vol. 41, no. 5, pp. 638-642, 2002.
- [18] U. Harms, "Virtual and remote labs in physics education", *Proc. of 2nd European Conference on Physics Teaching in Engineering Education, Budapest*, 2000, pp. 1-6.
- [19] C. Lazar and S. Carari, "A remote-control engineering laboratory", *IEEE Transactions on Industrial Electronics*, vol. 55, issue 6, pp. 2368-2375, 2008.
- [20] L. Leva and F. Donida, "Multifunctional remote laboratory for education in automatic control: the CrAutoLab experience", *IEEE Transactions on Industrial Electronics*, vol. 55, issue 6, pp. 2376-2385, 2008.
- [21] D.Z. Deniz, A. Bulancak, and G. Ozcan, "A novel approach to remote laboratories", *Proc. of 33rd ASEE/IEEE Frontiers in Education Annual Conference, Boulder CO*, 2003, pp. T3E/8-T3E/12.
- [22] Κ.Θ. Κώτσης και Φ.Β. Ευαγγέλου, "Γνωρίσματα ερευνών της Διεθνούς Βιβλιογραφίας σχετικά με τα μαθησιακά αποτελέσματα από τη σύγκριση εικονικών και πραγματικών πειραμάτων στη διδασκαλία και μάθηση της Φυσικής", *6ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση, Φλώρινα*, 2009, σελ. 336-343.
- [23] A. Hofstein and V.N. Lunetta, (2004) "The laboratory in science education: foundations for the twenty-first century", *Science Education*, vol. 88, issue 1, pp. 28-54, 2004.
- [24] M. Windschitl, "Using simulations in the middle school: does assertiveness of dyad partners influence conceptual change?" *International Journal of Science Education*, vol. 23, issue 1, pp. 17-32, 2001.
- [25] Δ. Ε. Μπολανάκης και Κ. Κώτσης, "Χρήση πειραμάτων απομακρυσμένης πρόσβασης με ασύρματη επικοινωνία στη διδασκαλία Φυσικής: ανασκόπηση βιβλιογραφίας και προτάσεις", *6ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση, Φλώρινα*, 2009, σελ. 579-586.
- [26] Γ. Γιαννέλος και Χ. Πολάτογλου, "Πραγματικά πειράματα Φυσικής από απόσταση: μια εφαρμογή σε LabVIEW για τη μελέτη του φαινομένου της διάθλασης", *9ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση, Θεσσαλονίκη*, 2015, σελ. 237-241.
- [27] M. Cooper and J.M.M. Ferreira, "Remote laboratories extending access to science and engineering curricular", *IEEE Transactions on Learning Technologies*, vol. 2, no. 4, pp. 342-353, 2009.
- [28] D. E. Bolanakis, K. T. Kotsis, and T. Laopoulos, "A Prototype Wireless Sensor Network System for a Comparative Evaluation of Differential and Absolute Barometric Altimetry", *IEEE Aerospace and Electronic Systems Magazine*, 30(11), pp. 20-28, 2015.
- [29] D. E. Bolanakis, K. T. Kotsis, and T. Laopoulos, "Temperature Influence on Differential Barometric Altitude Measurements", *The 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS'2015), Warsaw, Poland*, 2015, pp. 120-124.
- [30] D. E. Bolanakis, K. T. Kotsis, T. Laopoulos, "Work in Progress: Experiments on MEMS Barometric Pressure Sensors", *3ο Πανελλήνιο Συνέδριο Ηλεκτρονικής και Τηλεπικοινωνιών (PACET 2015), Ιωάννινα*, 2015, σελ. 1-3.

- [31] D. E. Bolanakis, K. T. Kotsis, and T. Laopoulos, "Ethernet & PC-based Experiments on Barometric Altimetry using MEMS in a Wireless Sensor Network", *Computer Application in Engineering Education*, 24(3), pp. 428-442, 2016.
- [32] R. Sallier, R. Missler, and A. Schutze, "VenDASys — A versatile experimentation platform for educational purposes", *2010 IEEE Education Engineering (EDUCON)*, pp. 205-210, 2010.
- [33] D.E. Bolanakis, E. Glavas, and G.A. Evangelakis, "An integrated microcontroller-based tutoring system for computer architecture laboratory course," *Int. J. Eng. Educ.*, vol. 26, no. 4, pp. 785-798, 2007.
- [34] Δ. Ε. Μπολανάκης και Κ. Κώτσης, "Απομακρυσμένος έλεγχος μετρήσεων Φυσικών μεγεθών πραγματικού χρόνου: δυνατότητες και περιορισμοί", *9ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση, Θεσσαλονίκη*, 2015, σελ. 232-236.
- [35] A. Saxena¹, M.M. Singh, and V. Singh³, "The state of art of MEMS in automation industries", *Proc. of Innovative Trends in Applied Physical, Chemical, Mathematical Sciences and Emerging Energy Technology for Sustainable Development (APCMET '14)*, pp. 1-5, 2014.
- [36] A.A. Ahmed, H. Shi, and Y. Shang, "A survey on network protocols for wireless sensor networks", *Proc. of International Conference on Information Technology: Research and Education (ITRE '03)*, NJ, USA, pp. 301-305, 2003.
- [37] Y. Lai and S.A. Bazaz, "Engineering curriculum development in microsystems", *Proc. of the Canadian Engineering Education Association*, pp. 145-149, 2006.
- [38] L. Lin, "Curriculum development in microelectromechanical systems in mechanical engineering", *IEEE Transactions on Learning Technologies*, vol. 44, no. 1, pp. 61-66, Feb 2001.
- [39] R.J. Pryputniewicz, "Current trends and future directions in MEMS", *Experimental Mechanics*, vol. 52, no. 3, pp. 289-303, Mar 2012.

Πειράματα βαρομετρικής υψομετρίας

Τα Μικρο-Ηλεκτρομηχανικά Συστήματα (Micro-Electromechanical Systems – MEMS) αποτελούν τεχνολογία εξαιρετικά μικρής συσκευασίας, χαμηλού κόστους και κατανάλωσης, η οποία ενσωματώνει μία μηχανική λειτουργία στην αλλαγή της οποίας παράγεται ηλεκτρικό σήμα. Η διεπισημονική αυτή φύση των MEMS (επιστήμη των υλικών, συστήματα ελέγχου, αισθητήρες, σχεδιασμός ολοκληρωμένων συστημάτων, κ.ά.) έχει επιφέρει νέα αναπτυξιακή δυναμική στο βιομηχανικό αυτοματισμό [1] και συνεπώς, έχει επηρεάσει σημαντικά και την εκπαίδευση όπου πλέον συναντώνται αρκετά περιγράμματα σπουδών με αντικείμενο μελέτης τα MEMS [2-4].

Στο κεφάλαιο αυτό γίνεται μία αναφορά στους αισθητήρες MEMS με έμφαση τους βαρομετρικούς αισθητήρες πίεσης, αναλύεται η θεωρία βαρομετρικής υψομετρίας, και κατόπιν παρουσιάζεται μία πρωτότυπη πειραματική διάταξη για τη συλλογή μετρήσεων ατμοσφαιρικής πίεσης με τη χρήση ασύρματων αισθητήρων τεχνολογίας MEMS.

2

ΣΤΟ ΚΕΦΑΛΑΙΟ ΑΥΤΟ

ΕΙΣΑΓΩΓΗ ΣΤΟΥΣ ΑΙΣΘΗΤΗΡΕΣ MEMS
Γίνεται μια εισαγωγή στους αισθητήρες MEMS με δίνεται έμφαση στις εφαρμογές βαρομετρικών αισθητήρων πίεσης.

ΜΕΤΡΗΣΕΙΣ ΑΠΟΛΥΤΟΥ ΥΨΟΥΣ ΚΑΙ ΘΕΩΡΙΑ ΒΑΡΟΜΕΤΡΙΚΗΣ ΥΨΟΜΕΤΡΙΑΣ
Γίνεται μια εισαγωγή στη δυνατότητα μέτρησης απόλυτου ύψους και κατόπιν αναλύεται η θεωρία υπολογισμού βαρομετρικού υψομέτρου, καθώς και οι τεχνικές απλής και διαφορικής βαρομετρικής υψομετρίας.

ΠΡΩΤΟΤΥΠΟ ΣΥΣΤΗΜΑ ΒΑΡΟΜΕΤΡΙΚΗΣ ΥΨΟΜΕΤΡΙΑΣ

Παρουσιάζεται ο πρωτότυπος εκπαιδευτικός και ερευνητικός εξοπλισμός που αναπτύχθηκε για την εξυπηρέτηση των εκπαιδευτικών αναγκών του Εργαστηρίου Εκπαίδευσης και Διδασκαλίας της Φυσικής του Παιδαγωγικού Τμήματος Δημοτικής Εκπαίδευσης (Π.Τ.Δ.Ε.) στο Πανεπιστήμιο Ιωαννίνων.

Εισαγωγή στους αισθητήρες MEMS

Οι αισθητήρες MEMS συναντώνται σε πληθώρα καθημερινών εφαρμογών και ηλεκτρονικών εφαρμογών αιχμής στη σημερινή εποχή, όπως π.χ. σε κινητά τηλέφωνα, αυτοκίνητα, ηλεκτρονικά συστήματα αεροσκαφών, κλπ. Οι πιο δημοφιλείς αισθητήρες MEMS είναι οι ακόλουθοι:

1. Τα επιταχυνσιόμετρα (*accelerometers*) τα οποία μετρούν γραμμική επιτάχυνση (στατική ή δυναμική).
2. Τα γυροσκόπια (*gyroscopes*) τα οποία μετρούν γωνιακή ταχύτητα.
3. Οι ψηφιακές πυξίδες (*digital compasses*) που δίνουν την πληροφορία προσανατολισμού του «σώματός» του σε σχέση με το βορρά.
4. Οι αισθητήρες βαρομετρικής πίεσης (*barometric pressure sensors*) που μετρούν το μοριακό βάρος του ατμοσφαιρικού αέρα.

Στα ηλεκτρονικά συστήματα αυτοκινήτων, επιταχυνσιόμετρα ανιχνεύουν τυχόν πρόσκρουση και ενεργοποιούν (όταν κρίνεται απαραίτητο) τους αερόσακους του αυτοκινήτου. Στα κινητά τηλέφωνα, γυροσκόπια (σε συνεργασία με επιταχυνσιόμετρα) ανιχνεύουν την κίνηση της συσκευής και περιστρέφουν αναλόγως την οθόνη του κινητού. Στα ηλεκτρονικά συστήματα αεροσκαφών, ψηφιακές πυξίδες ανιχνεύουν το μαγνητικό πεδίο και χρησιμοποιούνται (σε συνεργασία με τα επιταχυνσιόμετρα και τα γυροσκόπια) για την πλοήγηση του αεροσκάφους.

Οι βαρομετρικοί αισθητήρες πίεσης μετρούν την ατμοσφαιρική πίεση και χρησιμοποιούνται σε εφαρμογές πρόβλεψης καιρού, ενώ με την αναγωγή της ατμοσφαιρικής πίεσης σε βαρομετρικό υψόμετρο (*barometric altitude*) μπορούν να χρησιμοποιηθούν για τη βελτίωση της ακρίβειας στις συσκευές παγκόσμιου δορυφορικού συστήματος πλοήγησης (*Global Navigation Satellite System – GNSS¹*). Στη διεθνή βιβλιογραφία διατίθεται πληθώρα ερευνητικών προτάσεων που αφορούν τη χρήση βαρομετρικών αισθητήρων πίεσης για τον προσδιορισμό υψομέτρου σε συστήματα GNSS [5-7] και ηλεκτρονικά όργανα αεροσκαφών [8-10], αλλά ακόμη και για τον καθορισμό των γωνιών θέσης των τελευταίων [11-13].

Αισθητήρες βαρομετρικής πίεσης είναι διαθέσιμοι στο εμπόριο από διάφορους κατασκευαστές. Οι πιο εμπορικοί είναι οι αισθητήρες των εταιριών Bosch Sensortec, STMicroelectronics και Freescale Semiconductors. Παρόλα αυτά, διατίθενται στην αγορά αξιόλογοι αισθητήρες και από ετέρους κατασκευαστές, όπως η εταιρία Measurements Specialties και η Ελληνική εταιρία European Sensor Systems. Στον Πίνακα 2—1 παρουσιάζονται τα κυριότερα τεχνικά χαρακτηριστικά πέντε αισθητήρων των προαναφερθέντων κατασκευαστών [14-18] με τη σειρά που αναφέρθηκαν. Η επιλογή αισθητήρα γίνεται συνήθως με γνώμονα την ακρίβεια μέτρησης, το μέγεθος συσκευής, και την κατανάλωση ρεύματος.

¹ Ο δημοφιλέστερος, ίσως, όρος GPS (*Global Positioning System*) αναφέρεται ουσιαστικά στο GNSS σύστημα των ΗΠΑ.

Πίνακας 2—1 Αισθητήρες βαρομετρικής πίεσης διαφόρων κατασκευαστών

Αισθητήρας	Εύρος μέτρησης (range)	Ευκρίνεια μέτρησης (resolution)	Κατανάλωση ρεύματος (low-power)	Μέγεθος συσκευής (ΜxΠxΥ)	Πρωτόκολλο διασύνδεσης
BMP180	300-1100 mbar	0,02 mbar (RMS)	32 μ A	3,6 x 3,8 x 0,93 mm ³	I ² C
LPS331AP	260-1260 mbar	0,02 mbar (RMS)	30 μ A	3,0 x 3,0 x 1,0 mm ³	I ² C, SPI
MPL115A2	500-1150 mbar	1,5 mbar (RMS)	5 μ A	5,0 x 3,0 x 1,2 mm ³	I ² C
MS5607	10-1200 mbar	0,024 mbar (RMS)	12,5 μ A	5,0 x 3,0 x 1,0 mm ³	I ² C, SPI
ESCP2-M5B-001.2	650-1200 mbar	0,015 mbar (RMS)	Μη διαθέσιμη	7,4 x 5,1 x 1,95 mm ³	SPI, TWI

Αξίζει να σημειωθεί ότι, η αγορά αισθητήρων πίεσης αναμένεται να φθάσει τα 7,34 δισεκατομμύρια δολάρια το 2017 [19]. Λαμβάνοντας υπόψη τη πληθώρα εφαρμογών που βασίζονται σε βαρομετρικούς αισθητήρες πίεσης, στα πλαίσια της διδακτορικής διατριβής σχεδιάστηκε και υλοποιήθηκε πρωτότυπη πειραματική διάταξη για τη συλλογή μετρήσεων ατμοσφαιρικής πίεσης από δύο ασύρματες συσκευές μέτρησης, και τον μετέπειτα υπολογισμό του απόλυτου ύψους μεταξύ των δύο συσκευών. Στη συνέχεια παρουσιάζεται το θεωρητικό υπόβαθρο της βαρομετρικής υψομετρίας και του τρόπου υπολογισμού απόλυτου ύψους.

Μετρήσεις απόλυτου ύψους και θεωρία βαρομετρικής υψομετρίας

Μετρήσεις απόλυτου ύψους μεταξύ δύο σημείων στον άξονα z (z_A και z_B), ανεπηρεάστες από τις θέσεις στον άξονα τετμημένων (x_A , x_B) και τεταγμένων (y_A , y_B), μπορούν να πραγματοποιηθούν είτε με τη μέτρηση του γεωμετρικού υψομέτρου (*geometric altitude*) είτε με τη μέτρηση του βαρομετρικού υψομέτρου. Η μέτρηση γεωμετρικού υψομέτρου επιτυγχάνεται με συσκευές GNSS, οι οποίες λαμβάνουν συντεταγμένες από 4 τουλάχιστον δορυφόρους, αλλά μειονεκτούν λόγω της ακρίβειας της μέτρησης και της αδυναμίας εντοπισμού σήματος σε περιβάλλον εσωτερικού χώρου [20,21]. Η μέτρηση βαρομετρικού υψομέτρου επιτυγχάνεται με αισθητήρες μέτρησης της ατμοσφαιρικής πίεσης, οι οποίοι μετρούν την ασκούμενη πίεση στον αισθητήρα που προκαλείται από το βάρος των μορίων του αέρα και που μειώνεται με σταθερό ρυθμό όσο αυξάνει το υψόμετρο. Ωστόσο, ο υπολογισμός βαρομετρικού υψομέτρου είναι ιδιαίτερα ευπαθής στις καιρικές συνθήκες και μπορεί να αποδώσει σημαντικό σφάλμα μέτρησης, ιδιαίτερα σε περιβάλλον εξωτερικού χώρου [22]. Τα όργανα μέτρησης βαρομετρικού υψομέτρου καλούνται συνήθως *βαρόμετρα* (*barometers*).

$$P_{(z)} = P_0 e^{-(Z/H)}, \text{ (Υψομετρική εξίσωση)}$$

$$\text{όπου : } H = \frac{kT}{mg} \text{ (Ατμοσφαιρική κλίμακα ύψους)}$$

$$P_0 \text{ (Μέση τιμή ατμοσφαιρικής πίεσης στην επιφάνεια της θάλασσας σε Pa)} = 102325 \text{ (τυπική τιμή)}$$

$$k \text{ (Σταθερά Boltzmann's σε J / deg)} = 1.38 \cdot 10^{-23}$$

$$T \text{ (Μέση τιμή θερμοκρασίας του ατμοσφαιρικού στρώματος σε Kelvin)} = 273K (= 0^\circ C, \text{ τυπική τιμή})$$

$$m \text{ (Μέση τιμή μάζας των ατόμων του αέρα σε kilograms)} = 4.76 \cdot 10^{-26}$$

$$(22\% O_2 \text{ και } 78\% N_2 \Rightarrow 0.22 \cdot 2 \cdot 2.67 \cdot 10^{-26} \text{ Kg} + 0.78 \cdot 2 \cdot 2.3 \cdot 10^{-26} \text{ Kg})$$

$$g \text{ (Επιτάχυνση βαρύτητας σε meters / sec}^2\text{)} = 9.81$$

Τύπος 2—1 Υψομετρική εξίσωση

Ο υπολογισμός βαρομετρικού υψομέτρου βασίζεται στην *υψομετρική εξίσωση* (*hypsonetric equation*) που δίδεται στον **Τύπος 2—1**. Ως γνωστό η Γη περιβάλλεται από ένα στρώμα αερίων που ονομάζεται ατμόσφαιρα. Η σύνθεση της ατμόσφαιρας αποτελείται από 78% Άζωτο (N_2), 21% Οξυγόνο (O_2) και 1% άλλα αέρια. Η μάζα των ατόμων του αέρα έλκεται από τη Γη και συνεπώς, το βάρος του αέρα ασκεί δύναμη F σε κάθε επιφάνεια A που βρίσκεται σε επαφή με τον αέρα. Η δύναμη αυτή ονομάζεται ατμοσφαιρική (βαρομετρική) πίεση p (όπου $p=F/A$), και η τιμή της εξαρτάται από το υψόμετρο σε σχέση με την επιφάνεια της θάλασσας. Όσο αυξάνεται

νεται το υψόμετρο τόσο μειώνεται η ποσότητα του αέρα και συνεπώς, μειώνεται η μετρούμενη ατμοσφαιρική πίεση. Θεωρώντας αμελητέο το ποσοστό 1% των αερίων που περιβάλλουν την ατμόσφαιρα, τότε σε μια ισόθερμη ατμόσφαιρα η πίεση $P(z)$ ελαττώνεται εκθετικά με το ύψος z βάσει της υψομετρικής εξίσωσης. Η λύση της υψομετρικής εξίσωσης ως προς z δίνεται στον **Τύπο 2—2**, ενώ ο υπολογισμός απόλυτου ύψους μεταξύ δύο διαφορετικών βαρομετρικών υψομέτρων (όπως προκύπτει από την τελευταία σχέση), δίνεται στον **Τύπο 2—3**. Όπως παρατηρείται από την τελευταία σχέση, ο υπολογισμός απόλυτου ύψους με την υψομετρική εξίσωση απαιτεί (εκτός από τη μέτρηση ατμοσφαιρικής πίεσης) και τη μέτρηση της θερμοκρασίας, η οποία προκύπτει από τη μέση τιμή θερμοκρασίας των υψομέτρων A και B για τη θεώρηση ισόθερμης ατμόσφαιρας.

$$\ln P_{(z)} = \ln \left(P_0 e^{-(z/H)} \right) \Rightarrow \ln P_{(z)} = \ln P_0 + \ln e^{-(z/H)} \Rightarrow$$

$$\ln P_{(z)} - \ln P_0 = -\frac{z}{H} \Rightarrow z = H \cdot \ln P_0 - H \ln P_{(z)}$$

Τύπος 2—2 Λύση υψομετρικής εξίσωσης ως προς z

$$\begin{aligned} h &= P_{(zA)} - P_{(zB)} = \\ &= H \cdot \ln P_0 - H \ln P_{(zA)} - H \cdot \ln P_0 + H \ln P_{(zB)} = \\ &= H \left(\ln P_{(zB)} - \ln P_{(zA)} \right), \text{ όπου : } P_{(zA)} < P_{(zB)} \end{aligned}$$

Τύπος 2—3 Υπολογισμός απόλυτου ύψους με την υψομετρική εξίσωση

Μία πιο εμπειρική εκτίμηση του βαρομετρικού υψόμετρου, απαλλαγμένη από μετρήσεις θερμοκρασίας, επιτυγχάνεται με τη χρήση της διεθνούς βαρομετρική εξίσωση (*international barometric formula*), η οποία δίνεται στον **Τύπο 2—4**. Η εξίσωση αυτή είναι σε συμφωνία με το ιδεατό μοντέλο ατμόσφαιρας *ISA* (*International Standard Atmosphere*), σύμφωνα με το οποίο η ατμοσφαιρική πίεση και θερμοκρασία στο επίπεδο της θάλασσας ισούνται με 1023,24 Pa και 15 °C, αντιστοίχως. Ο υπολογισμός απόλυτου ύψους μεταξύ δύο διαφορετικών βαρομετρικών υψομέτρων (όπως προκύπτει από την τελευταία σχέση), δίνεται στον **Τύπο 2—5**.

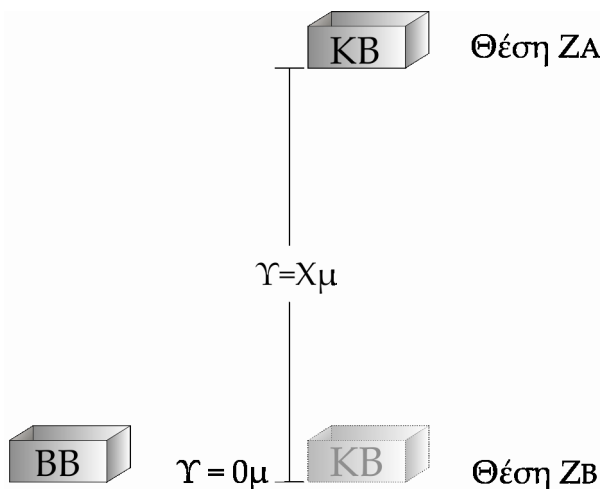
$$\begin{aligned} z &= 44330 \cdot \left(1 - \left(\frac{P_{(z)}}{P_0} \right)^{\frac{1}{5.255}} \right) = \\ &= 44330 - 44330 \left(\frac{1}{102325} \cdot P_{(z)} \right)^{0.1903} = \\ &= 44330 - 4935,125 \cdot P_{(z)}^{0.1903} \end{aligned}$$

Τύπος 2—4 Διεθνής βαρομετρική εξίσωση

$$\begin{aligned}
 h &= P_{(zA)} - P_{(zB)} = \\
 &= -4935,125 \cdot P_{(zA)}^{0.1903} + 4935,125 \cdot P_{(zB)}^{0.1903} = \\
 &= 4935,125 \cdot (P_{(zB)}^{0.1903} - P_{(zA)}^{0.1903}) \\
 &\text{όπου : } P_{(zA)} < P_{(zB)}
 \end{aligned}$$

Τύπος 2—5 Υπολογισμός απόλυτου ύψους με τη διεθνή βαρομετρική εξίσωση

Η μέτρηση βαρομετρικού υψομέτρου μπορεί να επιτευχθεί είτε με ένα βαρόμετρο που τοποθετείται σε δύο διαφορετικά σημεία (Α και Β) στον άξονα z, είτε με δύο ανεξάρτητα βαρόμετρα που μετρούν παράλληλα την τιμή υψομέτρου στο κάθε σημείο (z_A , z_B). Η πρώτη μέθοδος μέτρησης ονομάζεται *απλή βαρομετρική υψομετρία* (*single barometric altimetry*), ενώ η δεύτερη καλείται *διαφορική βαρομετρική υψομετρία* (*differential barometric altimetry*).



Εικόνα 2—1 Μέθοδος μέτρησης βαρομετρικού υψομέτρου

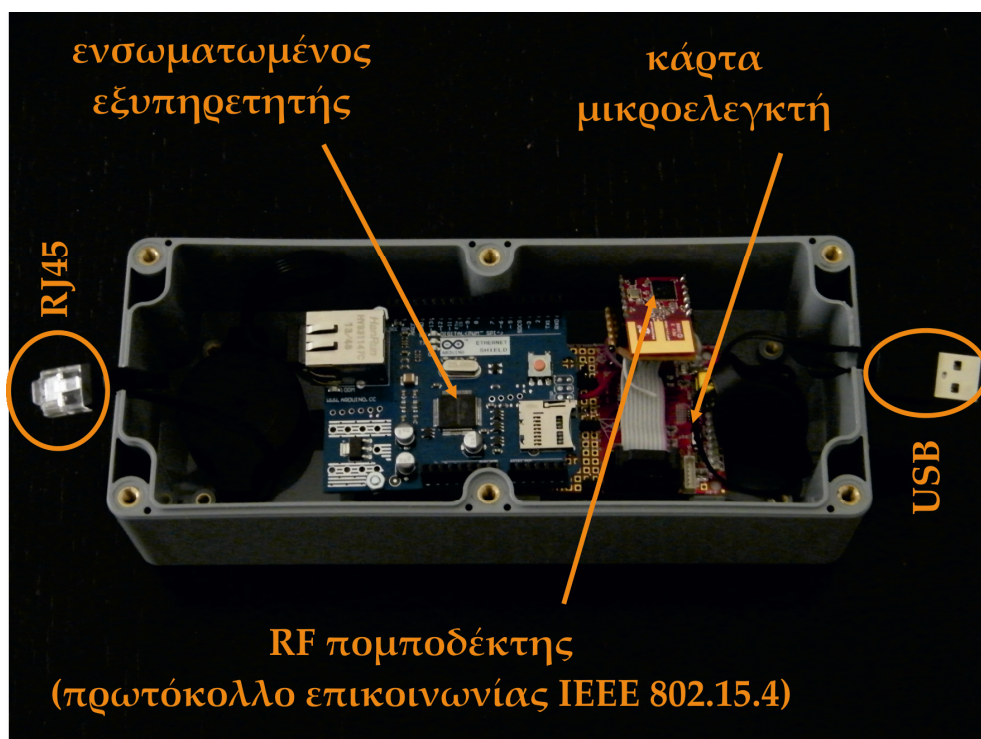
Στον **Εικόνα 2—1** παρουσιάζεται η διαδικασία μέτρησης βαρομετρικού υψομέτρου. Στην περίπτωση της διαφορικής βαρομετρικής υψομετρίας, τα δύο βαρόμετρα θα πρέπει να τοποθετηθούν αρχικά στο θέση Z_B για τη βαθμονόμηση των οργάνων. Η διαδικασία βαθμονόμησης απαιτείται διότι, παρατηρείται απόκλιση στο σήμα εξόδου δύο αισθητήρων που μετρούν την ίδια τιμή ατμοσφαιρικής πίεσης [23] και συνεπώς, η απόκλιση αυτή θα πρέπει να εκτιμηθεί και να απομακρυνθεί από τις τιμές ατμοσφαιρικής πίεσης για την ορθή εκτίμηση του απόλυτου ύψους. Κατόπιν, το ένα εκ των δύο βαρομέτρων τοποθετείται στη θέση Z_A και πραγματοποιείται η ταυτόχρονη μέτρηση των πιέσεων $P_{(zA)}$ και $P_{(zB)}$ από τις δύο συσκευές του συστήματος. Στη συνέχεια υπολογίζεται το απόλυτο ύψος ($Υ$) με τη χρήση είτε της υψομετρικής εξίσωσης, είτε της διεθνούς βαρομετρικής εξίσωσης. Επειδή η μία συσκευή μέτρησης παραμένει μόνιμα σταθερή στη θέση Z_B καλείται συνήθως *Βαρό-*

μετρο Βάσης (*base altimeter*), ενώ η ετέρα συσκευή μέτρησης αναφέρεται ως *Κινητό Βαρόμετρο* (*rover altimeter*).

Στην περίπτωση της απλής βαρομετρικής υψομετρίας, η μέτρηση των δύο πιέσεων $P_{(zA)}$ και $P_{(zB)}$ πραγματοποιείται αποκλειστικά από το κινητό βαρόμετρο (KB). Αξίζει να σημειωθεί ότι, η απλή βαρομετρική υψομετρία πλεονεκτεί έναντι της διαφορικής καθότι δεν απαιτείται βαθμονόμηση της συσκευής μέτρησης. Παρόλα αυτά, στη διεθνή βιβλιογραφία επιλέγεται συχνά η μέθοδος της διαφορικής βαρομετρικής υψομετρίας, λόγω της αυξημένης ακρίβειας μετρήσεων απόλυτου ύψους έναντι της απλής βαρομετρικής υψομετρίας [24-26].

Πρωτότυπο σύστημα βαρομετρικής υψομετρίας

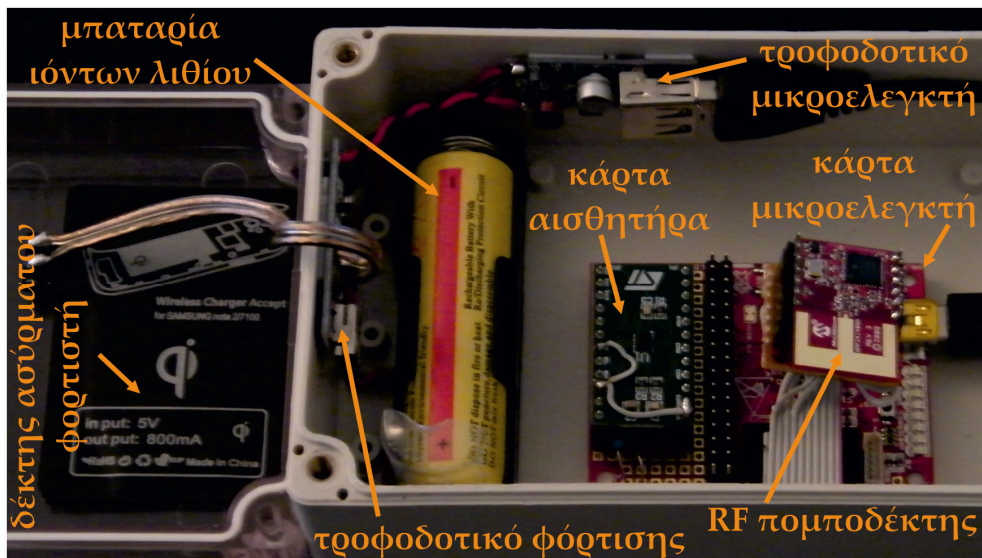
Στο Εργαστήριο Εκπαίδευσης και Διδασκαλίας της Φυσικής του Παιδαγωγικού Τμήματος Δημοτικής Εκπαίδευσης (του Πανεπιστημίου Ιωαννίνων) αναπτύχθηκε πρωτότυπος ερευνητικός εξοπλισμός για τη συλλογή μετρήσεων ατμοσφαιρικής πίεσης με εφαρμογή στη διαφορική βαρομετρική υψομετρία. Το σύστημα αποτελείται από δύο συσκευές μέτρησης που λαμβάνουν την ατμοσφαιρική πίεση από βαρομετρικούς αισθητήρες MEMS και αποστέλλουν τα δεδομένα ασύρματα σε μία τρίτη συσκευή, η οποία αποτελεί το συντονιστή του συστήματος. Ο τελευταίος επικοινωνεί με τον ηλεκτρονικό υπολογιστή του χρήστη είτε μέσω μιας θύρας USB, είτε μέσω του δικτύου/διαδικτύου. Στην πρώτη περίπτωση ο χρήστης ελέγχει τη διαδικασία λήψης μετρήσεων μέσω ενός προγράμματος που αναπτύχθηκε σε Labview, ενώ στη δεύτερη περίπτωση οι μετρήσεις λαμβάνονται μέσω ενός απλού προγράμματος περιήγησης στον ιστό (π.χ. Internet Explorer, Chrome, κλπ).



Εικόνα 2—2 Συσκευή του συντονιστή συστήματος

Στον Εικόνα 2—2 παρουσιάζεται η συσκευή του συντονιστή συστήματος. Η συσκευή αυτή αποτελείται από τρεις κάρτες: α) την κάρτα του μικροελεγκτή [27] που

αποτελεί το ενσωματωμένο σύστημα ελέγχου λειτουργίας του συντονιστή, β) την κάρτα ενσωματωμένου εξυπηρετητή [28] που είναι υπεύθυνη για την μετάδοση δεδομένων μέσω δικτύου/διαδικτύου, και γ) την κάρτα του RF πομποδέκτη [29] που χρησιμοποιείται για την ασύρματη επικοινωνία του συντονιστή με τις δύο συσκευές μέτρησης. Η σύνδεση της συσκευής στο δίκτυο/διαδίκτυο πραγματοποιείται μέσω του ακροδέκτη RJ45, ενώ η σύνδεση σε ηλεκτρονικό υπολογιστή επιτυγχάνεται μέσω του ακροδέκτη USB.



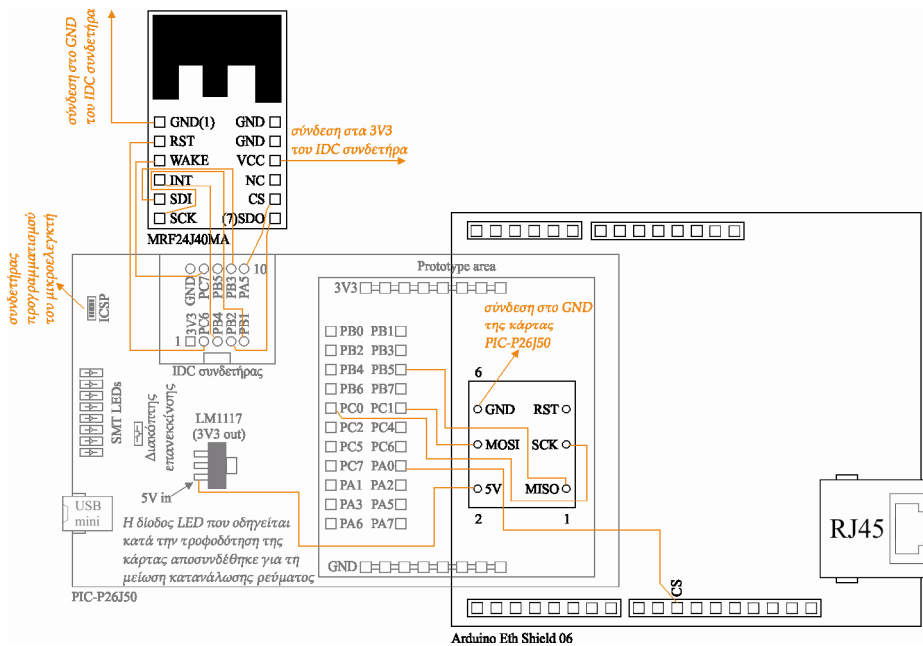
Εικόνα 2—3 Συσκευή μέτρησης της ατμοσφαιρικής πίεσης

Στον **Εικόνα 2—3** παρουσιάζεται η συσκευή μέτρησης ατμοσφαιρικής πίεσης. Η συσκευή αυτή αποτελείται από μία μπαταρία ιόντων λιθίου, έναν δέκτη ασύρματης φόρτισης και πέντε κάρτες: α) την κάρτα του μικροελεγκτή [27] που αποτελεί το ενσωματωμένο σύστημα ελέγχου λειτουργίας της συσκευής, β) την κάρτα του RF πομποδέκτη [29] που χρησιμοποιείται για την ασύρματη επικοινωνία με το συντονιστή του συστήματος, γ) την κάρτα αισθητήρα που μετράει την ατμοσφαιρική πίεση [30], δ) την κάρτα που λαμβάνει ισχύ από το δέκτη ασύρματης φόρτισης και φορτίζει την μπαταρία [31], και ε) μία κάρτα ανύψωσης τάσης από την εταιρία Shenzhen LC Technology Co, η οποία παρεμβάλλεται μεταξύ της μπαταρίας και της κάρτας του μικροελεγκτή. Στον **Εικόνα 2—4** παρουσιάζεται η συσκευή μέτρησης κατά τη διαδικασία ασύρματης φόρτισης της μπαταρίας².

² Για τη γρηγορότερη φόρτιση της μπαταρίας ενδείκνυται η αποσύνδεση της τροφοδοσίας από την κάρτα του μικροελεγκτή, όπως φαίνεται στην εικόνα. Κατά τη διαδικασία φόρτισης αλλάζει διαδοχικά η ένταση μια κόκκινης διόδου εκπομπής φωτός (LED) στην κάρτα φόρτισης, κατόπιν η διόδος LED παραμένει μόνιμα ανοικτή και μόλις ολοκληρωθεί η φόρτιση, σβήνει η κόκκινη διόδος LED και ανάβει μία πράσινη διόδος LED που βρίσκεται επίσης στην κάρτα φόρτισης.

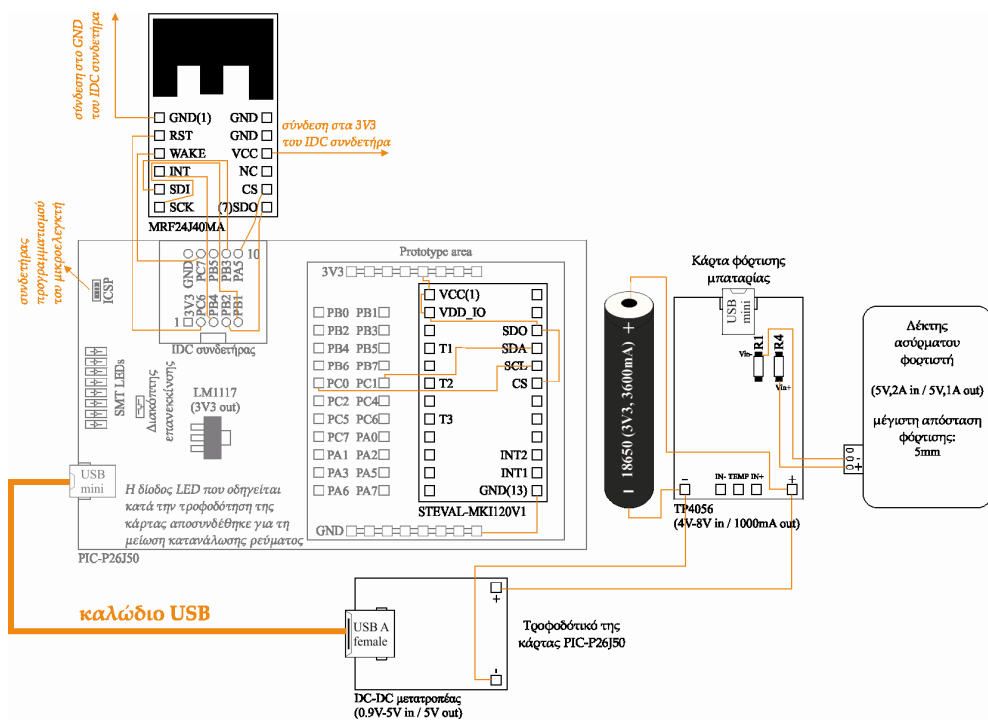


Εικόνα 2—4 Συσκευή μέτρησης (διαδικασία φόρτισης)



Εικόνα 2—5 Διάγραμμα διασυνδέσεων της συσκευής συντονιστή

Στην **Εικόνα 2—5** δίνονται τα διαγράμματα διασυνδέσεων των καρτών του συντονιστή συστήματος. Η επικοινωνία μικροελεγκτή με τον RF πομποδέκτη και με τον ενσωματωμένο εξυπηρετητή βασίζεται στο πρωτόκολλο SPI (Serial Peripheral Interface). Η τροφοδοσία του συντονιστή πραγματοποιείται μέσω του κυκλώματος διασύνδεσης USB mini, όπου μπορεί να χρησιμοποιηθεί ένας απλός φορτιστής κινητού που να φέρει το αντίστοιχο κύκλωμα διασύνδεσης. Ο πρωταρχικός προγραμματισμός του μικροελεγκτή πραγματοποιείται μέσω του κυκλώματος διασύνδεσης ICSP. Ένας πιθανός προγραμματιστής είναι ο PICkit2 [32]. Την πρώτη φορά φορτώνεται στο μικροελεγκτή ο κώδικας του bootloader³, ώστε η αναβάθμιση υλικολογισμικού (*firmware*) στο συντονιστή να μην απαιτεί τη χρήση του προγραμματιστή PICkit2. Μετά τον πρώτο προγραμματισμό του μικροελεγκτή, η εισαγωγή στον κώδικα του bootloader γίνεται μέσω του διακόπτη επανεκκίνησης που εντοπίζεται στην κάρτα του μικροελεγκτή.



Εικόνα 2—6 Διάγραμμα διασυνδέσεων των συσκευών μέτρησης

Στην **Εικόνα 2—6** δίνεται το διάγραμμα διασυνδέσεων της συσκευής μέτρησης

³ Ο κώδικας του bootloader εντοπίζεται στο φάκελο `FIRMWARE_bootloader26j50(bulkINTERFACE)`, ενώ ο κώδικας του λειτουργίας συντονιστή εντοπίζεται στον υποφάκελο `COORDINATOR` του φακέλου `FIRMWARE_RemoteLab(Ethernet-and-USB-interface)`. Για πληροφορίες αναβάθμισης υλικολογισμικού στο συντονιστή και στις συσκευές μέτρησης, βλ. Παράρτημα Α.

ατμοσφαιρικής πίεσης. Η επικοινωνία μικροελεγκτή με τον RF πομποδέκτη βασίζεται στο πρωτόκολλο SPI, ενώ επικοινωνία με τον αισθητήρα βαρομετρικής πίεσης βασίζεται στο πρωτόκολλο I²C (*Inter-Integrated Circuit*). Η τροφοδοσία της συσκευής μέτρησης γίνεται μέσω μπαταρίας ιόντων λιθίου χωρητικότητας 3600 mA, παρέχοντας αρκετές ώρες αυτονομίας στη συσκευή, ενώ η τροφοδότηση της κάρτας του μικροελεγκτή γίνεται μέσω του μετατροπέα DC-DC που ανυψώνει την τάση μπαταρίας από τα 3V3 στα 5V. Η κατανάλωση ρεύματος των καρτών της συσκευής μέτρησης δίνεται στον **Πίνακα 2—2**. Για τον προγραμματισμό του μικροελεγκτή των συσκευών μέτρησης⁴, ισχύει η ίδια διαδικασία που περιγράφηκε στην περίπτωση του συντονιστή.

Πίνακας 2—2 Κατανάλωση ρεύματος των καρτών της συσκευής μέτρησης

Κάρτα της συσκευής μέτρησης	Κατανάλωση ρεύματος (mA)
Κάρτα αισθητήρα βαρομετρικής πίεσης	2,3
Κάρτα μετατροπέα DC-DC (3V3 σε 5V)	0,1
Κάρτα μικροελεγκτή	8,8
Κάρτα πομποδέκτη RF (κατανάλωση κατά την αποστολή/λήψη δεδομένων)	49,8

Κάθε συσκευή του συστήματος χαρακτηρίζεται από μία ξεχωριστή διεύθυνση συσκευής για την επίτευξη της ασύρματης επικοινωνίας μεταξύ των συσκευών. Ο συντονιστής του συστήματος χρησιμοποιεί τη δεκαεξαδική διεύθυνση μηδέν (0x00), ενώ οι συσκευές μέτρησης χρησιμοποιούν τις δεκαεξαδικές διευθύνσεις 1 και 2 (0x01 και 0x02). Θα πρέπει να σημειωθεί ότι, η διεύθυνση 0x01 ταυτίζεται με την εντολή «καταγραφή της ατμοσφαιρικής πίεσης» ώστε οι συσκευές μέτρησης να λαμβάνουν ταυτόχρονη μέτρησης της βαρομετρικής πίεσης.

ΠΛΑΙΣΙΟ	1	2	3	4	5
ΣΥΝΤΟΝΙΣΤΗΣ	Αναμονή εντολής από Η/Υ ή δίκτυο	Αποστολή διεύθυνσης της ΣΜ1	Αναμονή απάντησης από τη ΣΜ1	Αναμονή απάντησης από τη ΣΜ1	Αναμονή απάντησης από τη ΣΜ1
ΣΥΣΚΕΥΗ ΜΕΤΡΗΣΗΣ (ΣΜ) 1	Αναμονή λήψης δεδομένων	Αναμονή λήψης δεδομένων	Λήψη διεύθυνσης από το συντονιστή	Μέτρηση της ατμοσφ. πίεσης	Αποστολή μέτρησης στο συντονιστή
ΣΥΣΚΕΥΗ ΜΕΤΡΗΣΗΣ (ΣΜ) 2	Αναμονή λήψης δεδομένων	Αναμονή λήψης δεδομένων	Λήψη διεύθυνσης από το συντονιστή	Μέτρηση της ατμοσφ. πίεσης	Αναμονή λήψης δεδομένων
ΠΛΑΙΣΙΟ	6	7	8	9	10
ΣΥΝΤΟΝΙΣΤΗΣ	Λήψη δεδομένων από τη ΣΜ1	Αποστολή διεύθυνσης της ΣΜ2	Αναμονή απάντησης από τη ΣΜ2	Αναμονή απάντησης από τη ΣΜ2	Λήψη δεδομένων από τη ΣΜ2
ΣΥΣΚΕΥΗ ΜΕΤΡΗΣΗΣ (ΣΜ) 1	Αναμονή λήψης δεδομένων	Αναμονή λήψης δεδομένων	Λήψη διεύθυνσης από το συντονιστή	Αναμονή λήψης δεδομένων	Αναμονή λήψης δεδομένων
ΣΥΣΚΕΥΗ ΜΕΤΡΗΣΗΣ (ΣΜ) 2	Αναμονή λήψης δεδομένων	Αναμονή λήψης δεδομένων	Λήψη διεύθυνσης από το συντονιστή	Αποστολή μέτρησης στο συντονιστή	Αναμονή λήψης δεδομένων

Εικόνα 2—7 Πρωτόκολλο επικοινωνίας των συσκευών του συστήματος

⁴ ο κώδικας λειτουργίας των συσκευών μέτρησης εντοπίζεται στους υποφακέλους END_DEVICE1 και END_DEVICE2 του φακέλου FIRMWARE_RemoteLab(Ethernet-and-USB-interface).

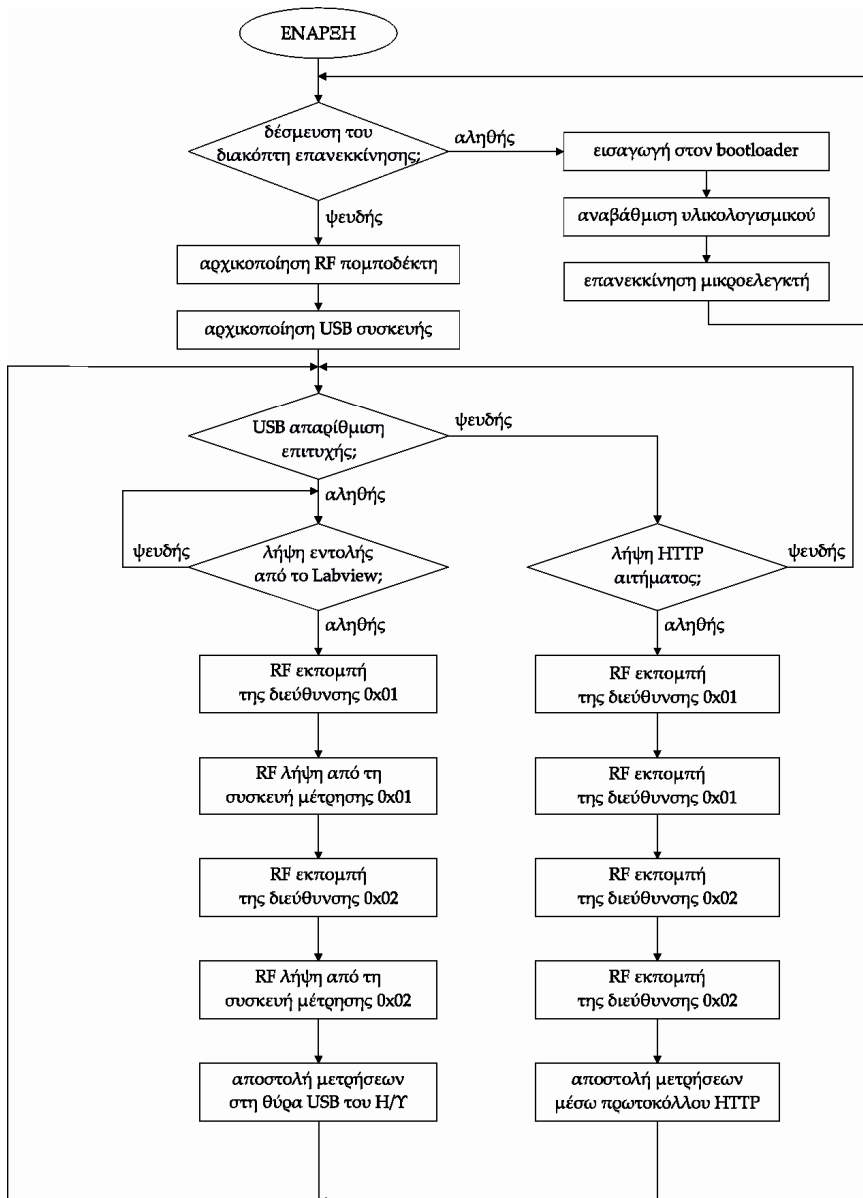
Στην **Εικόνα 2—7** δίνεται το πρωτόκολλο επικοινωνίας των συσκευών του συστήματος σε σχηματισμό πλαισίων. Στο πλαίσιο 1, ο συντονιστής αναμένει εντολή από είτε από το πρόγραμμα Labview που εκτελείται σε ηλεκτρονικό υπολογιστή, είτε από το πρόγραμμα περιήγησης στον ιστό (στην περίπτωση λειτουργίας του συστήματος μέσω δικτύου/διαδικτύου). Ταυτοχρόνως, οι συσκευές μέτρησης βρίσκονται σε λειτουργία ακρόασης και αναμονής εντολών από το συντονιστή. Στο πλαίσιο 2, ο συντονιστής αποστέλλει τη διεύθυνση της 1^{ης} συσκευής μέτρησης, η οποία ταυτίζεται με την εντολή «καταγραφή της ατμοσφαιρικής πίεσης» (ενώ οι συσκευές μέτρησης παραμένουν σε λειτουργία αναμονής δεδομένων). Στο πλαίσιο 3, ο συντονιστής εισάγεται σε λειτουργία αναμονής απάντησης από την 1^η συσκευή μέτρησης, ενώ την ίδια χρονική στιγμή η τελευταία λαμβάνει και αναγνωρίζει τη διεύθυνσή της. Ταυτοχρόνως, οι δύο συσκευές μέτρησης αναγνωρίζουν την εντολή «καταγραφή της ατμοσφαιρικής πίεσης» και συνεπώς, στο πλαίσιο 4 πραγματοποιείται μέτρηση της ατμοσφαιρικής πίεσης και από τις δύο συσκευές (ενώ ο συντονιστής παραμένει σε λειτουργία αναμονής δεδομένων από την 1^η συσκευή μέτρησης). Στο πλαίσιο 5, η 1^η συσκευή αποστέλλει τη μέτρηση ατμοσφαιρικής πίεσης στο συντονιστή, η οποία λαμβάνεται από τον τελευταίο στο πλαίσιο 6. Στο πλαίσιο 7, ο συντονιστής αποστέλλει τη διεύθυνση της 2^{ης} συσκευής μέτρησης. Η τελευταία, αναγνωρίζει τη διεύθυνσή της στο πλαίσιο 8 και αποστέλλει, στο πλαίσιο 9, τη μέτρηση ατμοσφαιρικής πίεσης (που καταγράφηκε προηγουμένως) πίσω στο συντονιστή. Ο συντονιστής λαμβάνει τη μέτρηση ατμοσφαιρικής πίεσης από τη 2^η συσκευή στο πλαίσιο 10, και η διαδικασία επαναλαμβάνεται από την αρχή.

Στην **Εικόνα 2—8** δίνεται το διάγραμμα ροής του υλικολογισμικού του συντονιστή συστήματος. Αρχικά, γίνεται έλεγχος δέσμευσης του διακόπτη επανεκκίνησης του μικροελεγκτή. Αν η συνθήκη είναι αληθής, γίνεται εισαγωγή του συντονιστή στον κώδικα bootloader για την αναβάθμιση του υλικολογισμικού συστήματος και στη συνέχεια, πραγματοποιείται επανεκκίνηση του μικροελεγκτή. Αν η συνθήκη είναι ψευδής, γίνεται αρχικοποίηση του RF πομποδέκτη και κατόπιν, αρχικοποίηση της συσκευής USB που διαθέτει ο μικροελεγκτής.

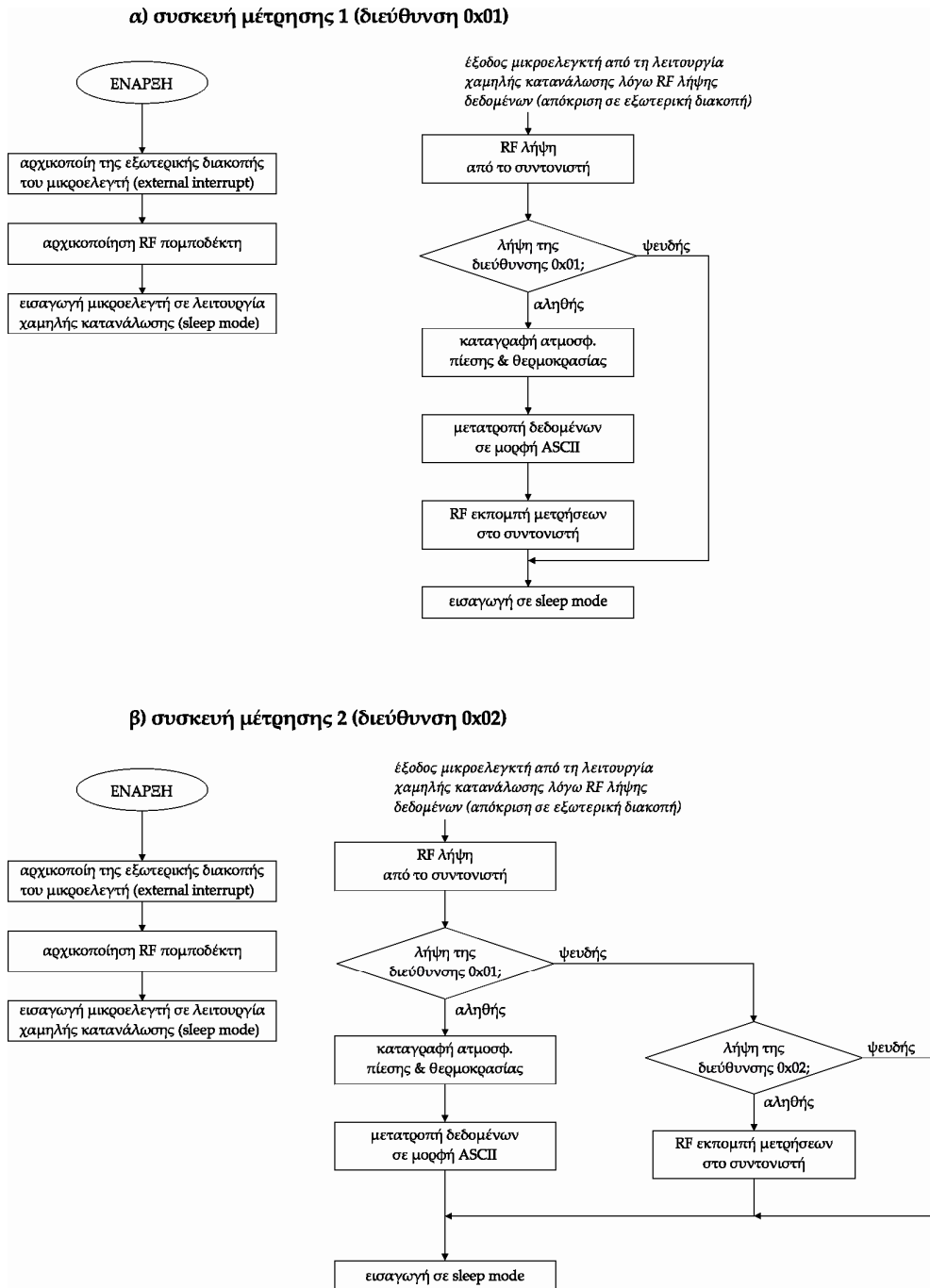
Στο σημείο αυτό ξεκινάει η εκτέλεση ενός βρόχου επαναλήψεων. Αρχικά ελέγχεται η επιτυχής απαρίθμηση της συσκευής USB. Αν η συνθήκη είναι αληθής, τότε το σύστημα εισάγεται σε λειτουργία λήψης μετρήσεων μέσω ηλεκτρονικού υπολογιστή. Συνεπώς, ο συντονιστής αναμένει τη λήψη εντολής από το εκτελέσιμο πρόγραμμα Labview. Μόλις ολοκληρωθεί η λήψη εντολής από τον ηλεκτρονικό υπολογιστή (Η/Υ), ο συντονιστής πραγματοποιεί εκπομπή της διεύθυνσης της 1^{ης} συσκευής μέτρησης. Ο συντονιστής λαμβάνει τις μετρήσεις από την 1^η συσκευή και κατόπιν, πραγματοποιεί εκπομπή της διεύθυνσης της 2^{ης} συσκευής μέτρησης. Μόλις ληφθούν τα δεδομένα και από τη 2^η συσκευή μέτρησης, ο συντονιστής στέλνει τις δύο μετρήσεις ατμοσφαιρικής πίεσης στη θύρα USB του ηλεκτρονικού υπολογιστή και επαναλαμβάνεται η εκτέλεση του βρόχου.

Αν, κατά την έναρξη του βρόχου επαναλήψεων, η απαρίθμηση USB επαληθευτεί

ψευδής, ο συντονιστής εισάγεται σε λειτουργία λήψης μετρήσεων μέσω δικτύου/διαδικτύου και αναμένει τη λήψη εντολής μέσω του πρωτοκόλλου μεταφοράς υπερκειμένου (*HyperText Transfer Protocol – HTTP*). Ακολουθείται η ίδια διαδικασία RF μετάδοσης δεδομένων για τη λήψη μετρήσεων και στη συνέχεια, ο συντονιστής αποστέλλει τις μετρήσεις σε μορφή HTTP και επαναλαμβάνεται η εκτέλεση του βρόχου.



Εικόνα 2—8 Διάγραμμα ροής υλικολογισμικού του συντονιστή



Εικόνα 2—9 Διάγραμμα ροής υλικολογισμικού των συσκευών μέτρησης

Στην **Εικόνα 2—9** δίνονται τα διάγραμμα ροής υλικολογισμικού των συσκευών μέτρησης. Αρχικά, οι μικροελεγκτές των συσκευών αρχικοποιούν τη λειτουργία εξωτερικής διακοπής, όπως παρουσιάζεται αριστερό τμήμα των δύο διαγραμμάτων ροής. Κατόπιν, γίνεται αρχικοποίηση του RF πομποδέκτη και στη συνέχεια οι μικροελεγκτές εισάγονται σε λειτουργία χαμηλής κατανάλωσης ρεύματος (sleep mode). Οι μικροελεγκτές των συσκευών μπορούν να «αφυπνιστούν» από τη λειτουργία χαμηλής κατανάλωσης (και να εισαχθούν σε κανονική λειτουργία εκτέλεσης κώδικα), με την εναλλαγή λογικού σήματος στον αντίστοιχο ακροδέκτη εξωτερικής διακοπής του μικροελεγκτή που ενεργοποιήθηκε πρωτίτερα. Ο ακροδέκτης εξωτερικής διακοπής οδηγείται από έναν ακροδέκτη του RF πομποδέκτη, ο οποίος αλλάζει την κατάσταση του σήματός του μόλις ολοκληρωθεί μία οποιαδήποτε λήψη δεδομένων. Συνεπώς, κατά την RF λήψη δεδομένων στις συσκευές μέτρησης, ο μικροελεγκτής των συσκευών εισάγεται στην κανονική λειτουργία εκτέλεσης κώδικα και εκτελεί τις εντολές που φαίνονται το δεξιό τμήμα των διαγραμμάτων ροής.

Κατά την έξοδο από τη λειτουργία χαμηλής κατανάλωσης, η 1^η συσκευή μέτρησης εξετάζει αν πραγματοποιήθηκε λήψη της διεύθυνσης 0x01. Αν ισχύει η συνθήκη, πραγματοποιείται καταγραφή της ατμοσφαιρικής πίεσης και θερμοκρασίας και κατόπιν, μετατρέπονται τα αριθμητικά δεδομένα μέτρησης σε χαρακτήρες του κώδικα ASCII (*American Standard Code for Information Interchange*). Η μετατροπή αυτή απαιτείται ώστε να μπορούν να τυπωθούν οι μετρήσεις στο πρόγραμμα περιήγησης στον ιστό (π.χ. Chrome) κατά τη λειτουργία του συστήματος μέσω δικτύου/διαδικτύου (χωρίς να απαιτείται περαιτέρω επεξεργασία των μετρήσεων από το συντονιστή του συστήματος). Κατά την ολοκλήρωση της διαδικασίας λήψης μετρήσεων, αλλά και στην περίπτωση όπου δεν η ισχύει η συνθήκη λήψης της διεύθυνσης 0x01, η συσκευή μέτρησης 1 εισάγεται και πάλι σε λειτουργία χαμηλής κατανάλωσης μέχρι την επόμενη RF λήψη δεδομένων.

Όσον αφορά τη 2^η συσκευή μέτρησης, κατά την έξοδο της τελευταίας από τη λειτουργία χαμηλής κατανάλωσης εξετάζεται αρχικά η συνθήκη λήψης της διεύθυνσης 0x01. Αν ισχύει η συνθήκη, πραγματοποιείται καταγραφή της ατμοσφαιρικής πίεσης και θερμοκρασίας και κατόπιν, μετατρέπονται τα αριθμητικά δεδομένα μέτρησης σε χαρακτήρες του κώδικα ASCII. Κατά την ολοκλήρωση της διαδικασίας λήψης μετρήσεων, αλλά και στην περίπτωση όπου δεν η ισχύει η συνθήκη λήψης της διεύθυνσης 0x01, η συσκευή μέτρησης 2 εισάγεται και πάλι σε λειτουργία χαμηλής κατανάλωσης μέχρι την επόμενη RF λήψη δεδομένων. Η 2^η συσκευή μέτρησης αποστέλλει τα δεδομένα στο συντονιστή συστήματος, όταν πραγματοποιηθεί λήψη της διεύθυνσης 0x02. Μετά την αποστολή των δεδομένων στο συντονιστή, η συσκευή αυτή εισάγεται και πάλι σε λειτουργία χαμηλής κατανάλωσης.

Οι κώδικες υλικολογισμικού του συντονιστή και των συσκευών μέτρησης του συστήματος έχουν αναπτυχθεί στη γλώσσα προγραμματισμού C, ενώ έχουν δημιουργηθεί βιβλιοθήκες οδήγησης των περιφερειακών συσκευών για την ευκολότερη ανάγνωση και αναβάθμιση του κώδικα. Η περιγραφή των επιμέρους βιβλιοθηκών

έχει ως εξής:

1. **W5100.h** περιλαμβάνει τις συναρτήσεις του ενσωματωμένου εξυπηρετητή (που εντοπίζεται στη συσκευή του συντονιστή συστήματος).
2. **LPS331AP.h** περιλαμβάνει τις συναρτήσεις του βαρομετρικού αισθητήρα MEMS (που εντοπίζεται στις συσκευές μέτρησης του συστήματος) για την καταγραφή της ατμοσφαιρικής λήψης και θερμοκρασίας, καθώς και τη μετατροπή των αριθμητικών δεδομένων σε ASCII χαρακτήρες.
3. **MRF24J40.h** περιλαμβάνει τις συναρτήσεις του RF πομποδέκτη (που εντοπίζεται σε όλες τις συσκευές του συστήματος) για την ασύρματη επικοινωνία μεταξύ των συσκευών του συστήματος.
4. **USB_bootloader.h** περιλαμβάνει τις οδηγίες του κώδικα bootloader (που εντοπίζεται σε όλες τις συσκευές του συστήματος) όπως π.χ., σε ποια περιοχή της μνήμης μικροελεγκτή είναι καταχωρημένο το υλικολογισμικό του bootloader, ποιες διόδους LED οδηγούνται στην κάρτα του μικροελεγκτή κατά την εισαγωγή του τελευταίου στον κώδικα bootloader, κ.ά.

Αξίζει να σημειωθεί ότι, είναι δυνατή η σύνδεση επιπρόσθετων συσκευών μέτρησης στο σύστημα. Στην περίπτωση εισαγωγής νέων συσκευών μέτρησης, απαιτούνται οι εξής αλλαγές υλικολογισμικού:

1. Στις νέες συσκευές μέτρησης φορτώνεται ο κώδικας υλικολογισμικού της συσκευής μέτρησης 2, πραγματοποιώντας αύξηση στον αριθμό διεύθυνσης της συσκευής. Η αλλαγή αυτή γίνεται στο αρχείο **EndDevice.h** (που βρίσκεται στο φάκελο END_DEVICE2) με αλλαγή της ακόλουθης σταθεράς (όπου X ο αύξων αριθμός διεύθυνσης).

```
#define EndDevice_Address_2 X
```

2. Στο συντονιστή συστήματος απαιτείται αλλαγή στο αρχείο **Coordinator.c** που βρίσκεται στο φάκελο COORDINATOR_revC(supports-up-to-4EPdevices), αλλάζουμε τον αριθμό συσκευών μέτρησης στην ακόλουθη σταθερά.

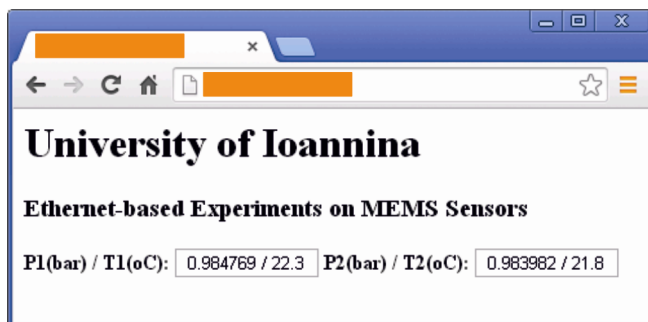
```
const int8 No_ofDevices = 2;
```

Η καταγραφή δεδομένων από τις νέες συσκευές μέτρησης μπορεί να πραγματοποιηθεί άμεσα από τη λειτουργία λήψης μετρήσεων μέσω δικτύου/διαδικτύου. Επιπλέον, έχει αναπτυχθεί κώδικας στη γλώσσα προγραμματισμού C που επιτρέπει τη λήψη μετρήσεων από ηλεκτρονικό υπολογιστή μέσω της γραμμής εντολών (*command prompt*). Στην περίπτωση χρήσης του κώδικα αυτού απαιτείται αλλαγή στο αρχείο **Pressure.c** που βρίσκεται στο φάκελο SOFTWARE_ansiC(for-COORDINATORrevC_supports-up-to-4EPdevices), αλλάζοντας τον αριθμό των χρησιμοποιούμενων συσκευών.

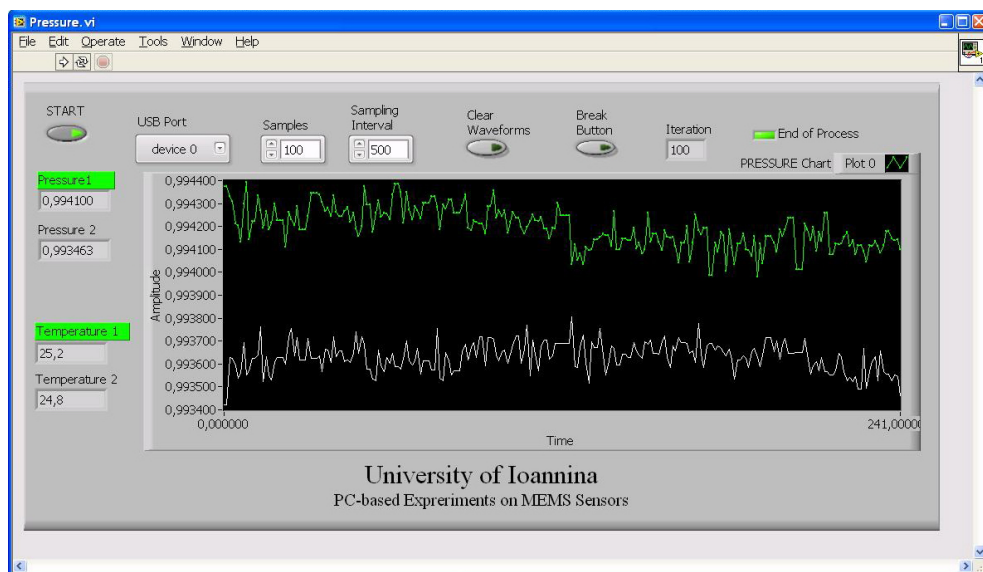
```
const int No_ofDevices = 2
```

Θα πρέπει να σημειωθεί ότι, για την κάθε συσκευή μέτρησης ο συντονιστής αποστέλλει 16 χαρακτήρες ASCII (16 Bytes) με τη μορφή "0.998415 25.4," ενώ

αποστέλλει όλες τις μετρήσεις συνολικά σε μία συναλλαγή (*transaction*)⁵. Λόγω περιορισμού του εύρους του καταχωρητή λήψης δεδομένων, που χρησιμοποιεί ο οδηγός (*driver*) επικοινωνίας του μικροελεγκτή με τη θύρα USB (που παρέχεται από την εταιρία Microchip [33]), στα 64 Bytes, στο σύστημα μπορούν να χρησιμοποιηθούν μέχρι και 4 συσκευές μέτρησης.



Εικόνα 2—10 Περιβάλλον λήψης μετρήσεων μέσω δικτύου/διαδικτύου



Εικόνα 2—11 Περιβάλλον λήψης μετρήσεων μέσω ηλεκτρονικού υπολογιστή

Στην Εικόνα 2—10 παρουσιάζεται το περιβάλλον συλλογής μετρήσεων μέσω δικτύου/διαδικτύου χρησιμοποιώντας το πρόγραμμα περιήγησης στον ιστό Chrome. Η ανανέωση των μετρήσεων πραγματοποιείται με την πληκτρολόγηση της διεύθυνσης δικτύου/διαδικτύου (*IP address*) με την οποία επικοινωνεί ο συντονισ-

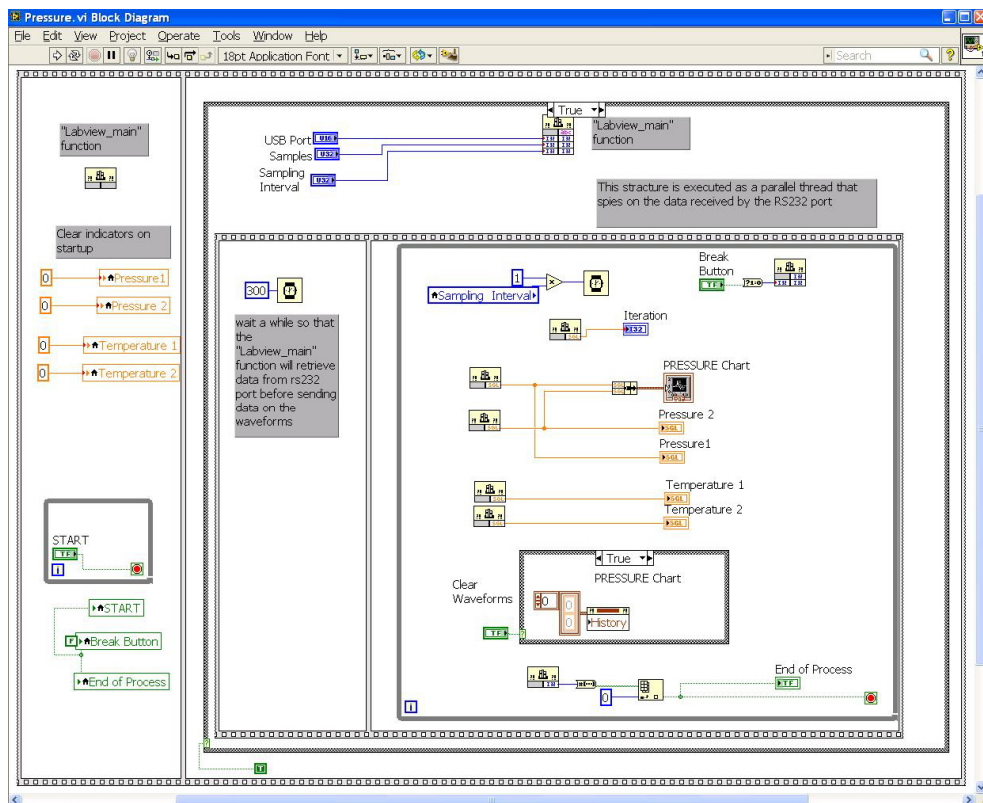
⁵ Στην τελευταία μέτρηση, ο χαρακτήρας του «κόμματος» (',') που διαχωρίζει τις μετρήσεις των επιμέρους συσκευών, αντικαθίστανται με τον χαρακτήρα «NULL» ('\0') σηματοδοτώντας το τέλος του πακέτου δεδομένων.

τής συστήματος.

Στην **Εικόνα 2—11** παρουσιάζεται το περιβάλλον συλλογής μετρήσεων μέσω ηλεκτρονικού υπολογιστή χρησιμοποιώντας το πρόγραμμα που αναπτύχθηκε στο Labview. Το τελευταίο πραγματοποιεί κλίσεις συναρτήσεων για την καταγραφή των μετρήσεων, οι οποίες αναπτύχθηκαν στην γλώσσα προγραμματισμού C για την ευκολότερη αναβάθμιση του προγράμματος Labview. Η περιγραφή των επιμέρους συναρτήσεων έχει ως εξής:

1. **Labview_init** περιλαμβάνεται στο αρχείο Pressure.c και δημιουργεί το αρχείο μετρήσεων Pressure.txt.
2. **Labview_main** περιλαμβάνεται στο αρχείο Pressure.c και αποτελείται την κύρια συνάρτηση του κώδικα για την επικοινωνία του συντονιστή με τη θύρα USB και την καταγραφή των μετρήσεων στο αρχείο Pressure.txt.
3. **return_Pressure1** περιλαμβάνεται στο αρχείο Pressure.h και χρησιμοποιείται για τη γραφική απεικόνιση της μετρούμενης ατμοσφαιρικής πίεσης που λαμβάνεται από την 1^η συσκευή μέτρησης, καθώς και την εκτύπωση της τρέχουσας τιμής πίεσης στο αντίστοιχο πλαίσιο αριθμητικής απεικόνισης (Pressure1) του Labview.
4. **return_Pressure2** περιλαμβάνεται στο αρχείο Pressure.h και χρησιμοποιείται για τη γραφική απεικόνιση της μετρούμενης ατμοσφαιρικής πίεσης που λαμβάνεται από τη 2^η συσκευή μέτρησης, καθώς και την εκτύπωση της τρέχουσας τιμής στο αντίστοιχο πλαίσιο αριθμητικής απεικόνισης (Pressure2) του Labview.
5. **return_Temperature1** περιλαμβάνεται στο αρχείο Pressure.h και χρησιμοποιείται για την εκτύπωση της τρέχουσας τιμής θερμοκρασίας, που λαμβάνεται από την 1^η συσκευή μέτρησης, στο αντίστοιχο πλαίσιο αριθμητικής απεικόνισης (Temperature1) του Labview.
6. **return_Temperature2** περιλαμβάνεται στο αρχείο Pressure.h και χρησιμοποιείται για την εκτύπωση της τρέχουσας τιμής θερμοκρασίας, που λαμβάνεται από τη 2^η συσκευή μέτρησης, στο αντίστοιχο πλαίσιο αριθμητικής απεικόνισης (Temperature2) του Labview.
7. **return_counter** περιλαμβάνεται στο αρχείο Pressure.h και χρησιμοποιείται για την εκτύπωση της τρέχουσας τιμής του μετρητή επαναλήψεων στο αντίστοιχο πλαίσιο αριθμητικής απεικόνισης (Iteration) του Labview, υποδεικνύοντας τον αριθμό μέτρησης πίεσης/θερμοκρασίας που πραγματοποιείται τη τρέχουσα χρονική στιγμή.
8. **break_function** περιλαμβάνεται στο αρχείο Pressure.h και λαμβάνει την τιμή του διακόπτη break_button του Labview, τερματίζοντας την εκτέλεση της συνάρτησης της κυρίας συναρτήσεως του κώδικα (Labview_main) στην περίπτωση δέσμευσης του διακόπτη αυτού.
9. **start_stop_function** περιλαμβάνεται στο αρχείο Pressure.h και επιστρέφει την τιμή μιας σημαίας, η οποία ανιχνεύει την πληροφορία την πληροφορία δέσμε-

υση του διακόπτη (εντός της κυρίας συνάρτησης Labview_main), και χρησιμοποιείται για τον τεματισμό του βρόχου καταγραφής των δεδομένων στις γραφικές του Labview κατά την ολοκλήρωση της τρέχουσας επανάληψης. Ο βρόχος αυτός εκτελείται παράλληλα με τη συνάρτηση Labview_main.



Εικόνα 2—12 Γραφικός προγραμματισμός στο Labview

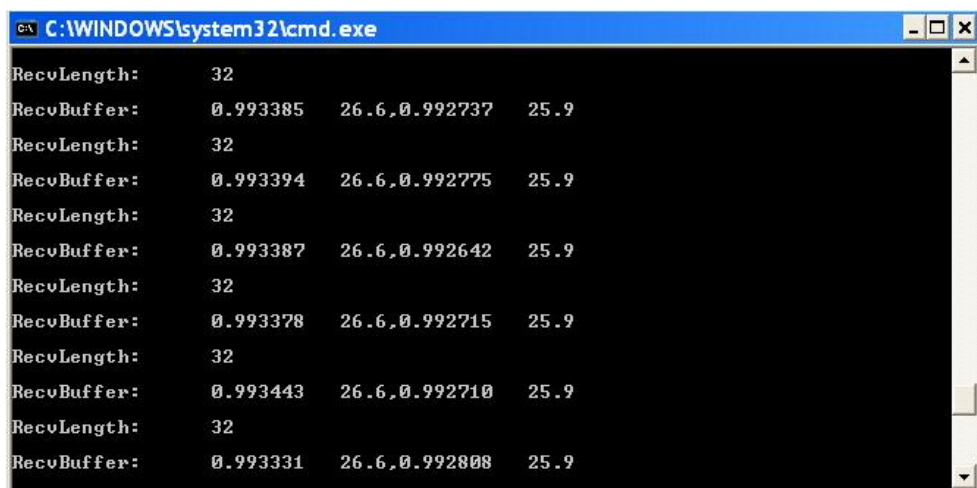
Ο κώδικας γραφικού προγραμματισμού Labview που καλεί τις παραπάνω συναρτήσεις, δίνεται στην Εικόνα 2—12. Όπως παρατηρείται, πρόκειται για έναν ιδιαίτερα μικρό κώδικα λόγω των χρησιμοποιούμενων συναρτήσεων C. Αναλυτικά, ο κώδικας περικλείεται σε μία δομή αλληλουχίας (*sequence structure*), όπου στο πρώτο «καρέ» της δομής πραγματοποιείται η εκτέλεση της συνάρτησης Labview_init η εκκαθάριση των πλήκτρων, γραφικών και πλαισίων αριθμητικής απεικόνισης του Labview. Η εκτέλεση του πρώτου «καρέ» της δομής επαναλαμβάνεται για όσο χρόνο παραμένει ανοικτός ο διακόπτης START (όπως ορίζεται από τη συνάρτηση While Loop του Labview). Με τη δέσμευση του διακόπτη START ξεκινάει η εκτέλεση του επόμενου «καρέ» της δομής. Στο «καρέ» αυτό εκτελούνται παράλληλα η συνάρτηση Labview_main και ο κώδικας εκτύπωσης των μετρήσεων στις γραφικές και πλαίσια αριθμητικής απεικόνισης του Labview. Ο κώδικας εκτύπωσης εντοπί-

ζεται σε βρόχο επαναλήψεων που περικλείεται στο 2^ο «καρέ» μιας ετέρας δομής δεδομένων (όπου στο 1^ο «καρέ» της δομής αυτής πραγματοποιείται απλώς μία καθυστέρηση χρόνου 300 msec). Ο βρόχος αυτός επαναλαμβάνεται ανά τακτά χρονικά διαστήματα, όπως ορίζεται από το πλαίσιο Sampling Interval του Labview (το οποίο καθορίζει ο χρήστης), ενώ ο τερματισμός του βρόχου πραγματοποιείται μέσω του διακόπτη Break Button του Labview. Σε μία δομή περίπτωσης (case structure) που εσωκλείεται εντός του βρόχου, είναι δυνατή η εκκαθάριση των γραφικών κατά τη δέσμευση του διακόπτη Clear Waveforms του Labview.

Στην **Εικόνα 2—13** παρουσιάζεται η λήψη δεδομένων χρησιμοποιώντας τη γραμμή εντολών ηλεκτρονικού υπολογιστή μέσω του εκτελέσιμου προγράμματος RemoteLab_cmd.exe. Για την εκτέλεση του αρχείου ανοίγουμε μία γραμμή εντολών και πληκτρολογούμε τις ακόλουθες εντολές.

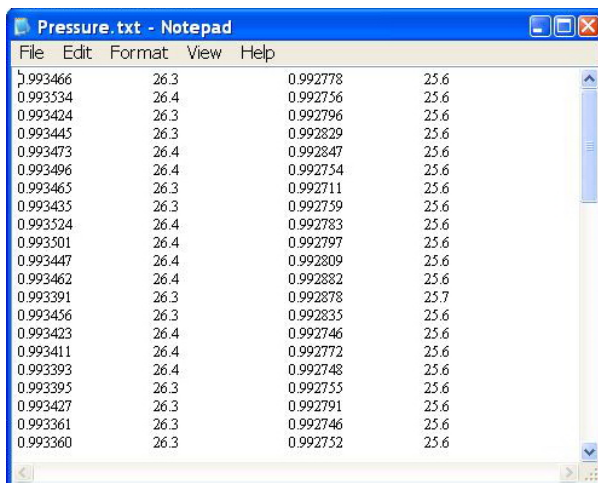
```
cd «διαδρομή δίσκου»  
RemoteLab_cmd
```

Στην πρώτη εντολή εισάγουμε τη διαδρομή δίσκου που βρίσκεται το αρχείο RemoteLab_cmd.exe, ενώ στη δεύτερη εντολή πληκτρολογούμε το όνομα του αρχείου για να αρχίσει η καταγραφή μετρήσεων. Οι ληφθέντες μετρήσεις ανά συσκευή διαχωρίζονται μεταξύ τους με κόμμα. Ως ένδειξη επιτυχούς λήψης μετρήσεων, το πρόγραμμα εκτυπώνει και τον αριθμό των ληφθέντων Bytes από τη θύρα USB του ηλεκτρονικού υπολογιστή (στην περίπτωση αυτή το σύστημα επικοινωνεί με δύο συσκευές μέτρησης και συνεπώς, λαμβάνονται 32 Bytes).



Εικόνα 2—13 Λήψη μετρήσεων από τη γραμμή εντολών Η/Υ

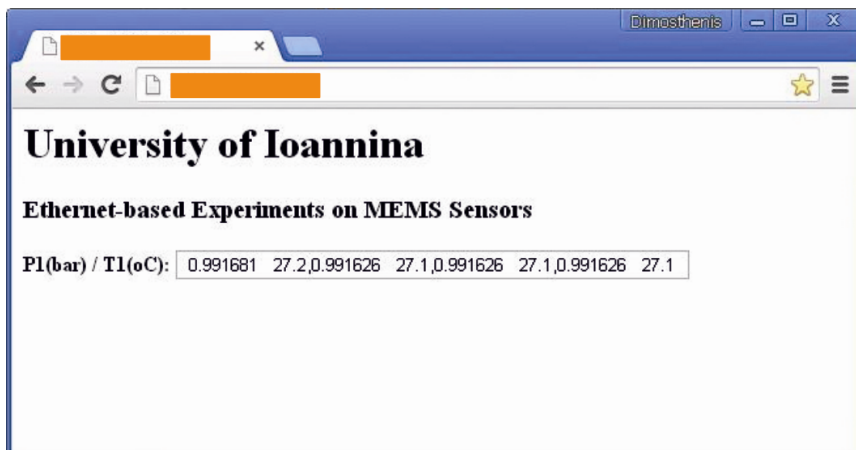
Το πρόγραμμα αυτό (όπως και το εκτελέσιμο πρόγραμμα Labview) καταχωρεί τις μετρήσεις στον τρέχοντα φάκελο του ηλεκτρονικού υπολογιστή, στο αρχείο που φέρει το όνομα Pressure.txt. Η μορφή των μετρήσεων παρουσιάζεται στην **Εικόνα 2—14**.



ID	P1 (bar)	T1 (°C)	P2 (bar)
0.993466	26.3	0.992778	25.6
0.993534	26.4	0.992756	25.6
0.993424	26.3	0.992796	25.6
0.993445	26.3	0.992829	25.6
0.993473	26.4	0.992847	25.6
0.993496	26.4	0.992754	25.6
0.993465	26.3	0.992711	25.6
0.993435	26.3	0.992759	25.6
0.993524	26.4	0.992783	25.6
0.993501	26.4	0.992797	25.6
0.993447	26.4	0.992809	25.6
0.993462	26.4	0.992882	25.6
0.993391	26.3	0.992878	25.7
0.993456	26.3	0.992835	25.6
0.993423	26.4	0.992746	25.6
0.993411	26.4	0.992772	25.6
0.993393	26.4	0.992748	25.6
0.993395	26.3	0.992755	25.6
0.993427	26.3	0.992791	25.6
0.993361	26.3	0.992746	25.6
0.993360	26.3	0.992752	25.6

Εικόνα 2—14 Αρχείο καταγραφής μετρήσεων στον Η/Υ

Στην Εικόνα 2—15 παρουσιάζεται η λήψη δεδομένων μέσω δικτύου/διαδικτύου από 4 συσκευές μέτρησης. Επειδή δε διατίθεται ο απαιτούμενος αριθμός συσκευών, το σύστημα θεωρεί τη μία εκ των δύο υπάρχοντων συσκευών μέτρησης ως την 3^η και 4^η συσκευή του συστήματος (και για το λόγω αυτό και οι μετρήσεις 2 έως 4 φέρουν την ίδια τιμή).



Εικόνα 2—15 Λήψη μετρήσεων μέσω δικτύου/διαδικτύου (4 συσκευές)

Αναφορές

- [1] A. Saxena¹, M.M. Singh, and V. Singh³, "The state of art of MEMS in automation industries", *Proc. of Innovative Trends in Applied Physical, Chemical, Mathematical Sciences and Emerging Energy Technology for Sustainable Development (APCMET '14)*, pp. 1-5, 2014.
- [2] Y. Lai and S.A. Bazaz, "Engineering curriculum development in microsystems", *Proc. of the Canadian Engineering Education Association*, pp. 145-149, 2006.
- [3] L. Lin, "Curriculum development in microelectromechanical systems in mechanical engineering", *IEEE Transactions on Learning Technologies*, vol. 44, no. 1, pp. 61-66, Feb 2001.
- [4] R.J. Pryputniewicz, "Current trends and future directions in MEMS", *Experimental Mechanics*, vol. 52, no. 3, pp. 289-303, Mar 2012.
- [5] J. Collin, O. Mezentsev, and G. Lachapelle, "Indoor positioning system using accelerometry and high accuracy heading sensors", *Proc. of GPS/GNSS 2003 Conference, Portland, OR, 2003*, pp. 1-7.
- [6] G. Retscher, "Augmentation of indoor positioning systems with a barometric pressure sensor for direct altitude determination in a multi-storey building", *CaGIS*, vol. 34, 2007, pp. 305-310.
- [7] A. Morrison, V. Renaudin, J.B. Bancroft, and G. Lachapelle "Design and testing of a multi-sensor pedestrian location and navigation platform", *Sensors*, vol. 12, 2012, pp. 3720-3738.
- [8] G. Retscher and M. Thienelt, "NAVIO – A Navigation and guidance service for pedestrians", *JGPS*, vol. 1, 2004, pp. 208-217.
- [9] S.J. Jung and D. Liccardo, "Automation of Small UAVs using a Low Cost Mems Sensor and Embedded Computing Platform", *Proc. of 25th Digital Avionics Systems Conference, Portland, OR, 2006*, pp. 1-9.
- [10] R.K. Arning and S. Sassen, "Flight control of micro aerial vehicles", *Proc. of AIAA Guidance, Navigation, and Control Conference and Exhibit, Providence, 2004*, pp. 1-12.
- [11] P. Paces and J. Popelka, "Performance evaluation of two altimeters intended for euler angles measurement", *Proc. of 2013 IEEE/AIAA 32nd Digital Avionics Systems Conference (DASC), East Syracuse, NY, 2013*, pp. 8C3-1 - 8C3-12.
- [12] J. Popelka and P. Paces, "Pressure based reference system for aircraft attitude measurement", *Proc. of 2013 IEEE/AIAA 32nd Digital Avionics Systems Conference (DASC), East Syracuse, NY, 2013*, pp. 4C4-1 - 4C4-14.
- [13] P. Paces, J. Popelka, and T. Levora, "Advanced display and position angles measurement systems, *Proc. of ICAS 2012 - 28th Congress of the International Council of the Aeronautical Sciences – Proceedings, Brisbane, Australia, 2012*, pp. P6.3.1-P6.3.14.
- [14] BMP180 Digital pressure sensor, Bosch Sensortec, 2013.
- [15] LPS331AP MEMS pressure sensor, STMicroelectronics, 2012.
- [16] MPL115A2 Miniature I2C Digital Barometer, Freescale Semiconductors, 2013.
- [17] MS5607 Miniature I2C Digital Barometer, Measurement Specialties, 2012.
- [18] ESCP2-M5B-001.2 MEMS Capacitive Pressure Sensor, European Sensor Systems, 2015.
- [19] Pressure Sensor Market (2012-2017) – Global Forecast, Trend & Analysis-Segmentation by technology [Piezoresistive, Capacitive, Electromagnetic (Reluctance), Resonant Solid State, Op-

- tical], *Applications [process and non process and Geography]*, available at <http://www.marketsandmarkets.com>, 2012.
- [20] Working paper presented by Hong Kong, China, "Use of barometric altitude and geometric altitude information in ADS-B message for ATC applications", *Proc. of Eighth Meeting of the South East Asia & Bay of Bengal Sub-regional ADS-B Implementation Working Group, Yangon, Myanmar*, 2012, pp. 1-4.
- [21] V. Zaliva and F. Franchetti, "Barometric and GPS altitude sensor fusion", *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing, Florence*, 2014, pp. 7575-7579.
- [22] M., Tanigawa, H. Luinge, L. Schipper, and P. Slycke, "Drift-free dynamic height sensor using MEMS IMU aided by MEMS pressure sensor", *Proc. of 5th Workshop on Positioning, Navigation and Communication, Hannover*, 2008. pp. 191-196.
- [23] W. Tang and Y-H Tsai, "Barometric altimeter short-term accuracy analysis", *IEEE Aerosp. Electron. Syst. Mag.*, vol. 20, 2005, pp. 24-26.
- [24] J. Parviainen, J. Kantola, and J. Collin, "Differential barometry in personal navigation", *Proc. of IEEE/ION Position, Location and Navigation Symposium, Monterey, CA*, 2008, pp. 148-152.
- [25] K. Liu, Y. Wang, and J. Wang, "Differential barometric altimetry assists floor identification in WLAN location fingerprinting study", *Principle and Application Progress in Location-Based Services, Springer International Publishing*, 2014, pp 21-29.
- [26] C. Bollmeyer, M. Pelka, H. Gehringy, and H. Hellbruck, "Evaluation of radio based, optical and barometric localization for indoor altitude estimation in medical applications", *Proc. of International Conference on Indoor Positioning and Indoor Navigation, Bexco, Busan, Korea*, 2014, pp. 1-10.
- [27] PIC-P26J50 Proto development board USER'S MANUAL, Olimex Ltd, 2012.
- [28] Arduino Ethernet Shield R3, Silver Telecom, 2009.
- [29] MRF24J40MA 2.4 GHz IEEE Std. 802.15.4™ RF Transceiver Module, Microchip, 2008.
- [30] STEVAL-MKI120V1 LPS331AP adapter board for a standard DIL24 socket, STMicroelectronics, 2013.
- [31] TP4056 1A Standalone Linear Li-Ion Battery Charger with Thermal Regulation in SOP-8, NanJing Top Power ASIC Corp.
- [32] PICkit2 Programmer/Debugger User's Guide, Microchip Technology Inc., 2008.
- [33] MCHPFSUSB Firmware User's Guide, Microchip Technology Inc., 2007.

Έρευνα και εκπαιδευτική αξιολόγηση πειραμάτων

Η ανάπτυξη εκπαιδευτικών πειραμάτων αποτελεί μία πολύπλοκη διεργασία, καθότι ο ερευνητής θα πρέπει γνωρίζει πολύ καλά τη διεθνή βιβλιογραφία ώστε να προβεί σε δόκιμη επιλογή του αντικειμένου διδασκαλίας που πρόκειται να εισάγει στο εργαστήριο, εφαρμόζοντας καινοτομίες τόσο σε εκπαιδευτικό επίπεδο, όσο και σε επίπεδο μηχανικής. Στη δεύτερη περίπτωση, η ανάπτυξη και υλοποίηση του εκπαιδευτικού εξοπλισμού θα πρέπει να διασφαλίζει την ορθή λειτουργία και ομαλή διεξαγωγή πειράματος, καθώς και να προωθεί τη διαδικασία εκμάθησης του γνωστικού αντικειμένου στους διδασκόμενους.

Στα πλαίσια της διατριβής αναπτύχθηκε εκπαιδευτικός εξοπλισμός που υποστηρίζει την εξ αποστάσεως πρόσβαση σε μετρήσεις φυσικών μεγεθών μέσω δικτύου/διαδικτύου, βασιζόμενος στην ανάπτυξη ασύρματου δικτύου αισθητήρων. Στη διεθνή βιβλιογραφία συναντώνται αρκετά παραδείγματα απομακρυσμένης πρόσβασης σε εκπαιδευτικά πειράματα [1-7], τα

3

ΣΤΟ ΚΕΦΑΛΑΙΟ ΑΥΤΟ

ΑΠΟΤΙΜΗΣΗ ΑΠΛΗΣ ΚΑΙ ΔΙΑΦΟΡΙΚΗΣ ΒΑΡΟΜΕΤΡΙΚΗΣ ΥΨΟΜΕΤΡΙΑΣ

Γίνεται αποτίμηση των πειραμάτων απλής και διαφορικής βαρομετρικής υψομετρίας με τη χρήση του πρωτότυπου ερευνητικού εξοπλισμού και παρουσιάζονται τα ερευνητικά αποτελέσματα που δημοσιεύτηκαν στο περιοδικό *IEEE Aerospace and Electronic Systems Magazine*, 2015 και στο συνέδριο *8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*, Warsaw, Poland, 2015.

ΑΠΟΤΙΜΗΣΗ ΕΚΠΑΙΔΕΥΤΙΚΩΝ ΠΕΙΡΑΜΑΤΩΝ

Γίνεται αποτίμηση των εκπαιδευτικών δυνατοτήτων πειραματισμού στη βαρομετρική υψομετρία με τη χρήση του πρωτότυπου ερευνητικού εξοπλισμού, όπως δημοσιεύθηκαν στο *3ο Πανελλήνιο Συνέδριο Ηλεκτρονικής και Τηλεπικοινωνιών (PACET 2015)*, Ιωάννινα, 2015 και στο περιοδικό *Wiley – Computer Applications in Engineering Education*, 2016. Ακόμη, δίνεται πρόταση αναβάθμισης του εξοπλισμού, για τον εμπλουτισμό πειραμάτων με MEMS αισθητήρες πίεσης, αλλά και την υλοποίηση αυτόνομου συστήματος για την εκπαίδευση ατόμων με κινητική αναπηρία. Η τελευταία πρόταση παρουσιάστηκε στο συνέδριο *Διδακτικές Προσεγγίσεις και Πειραματική Διδασκαλία στις Φυσικές Επιστήμες*, Θεσσαλονίκη, 2016.

πλεονεκτήματα των οποίων δεν περιορίζονται μόνο στη δυνατότητα μελέτης από απόσταση. Επεκτείνονται και στη δυνατότητα του εξ αποστάσεως ελέγχου πειραμάτων κατά τη διάρκεια παρουσίας των διδασκομένων στην τάξη [8] (μέσω του δικτύου/διαδικτύου παραδείγματος χάριν) μειώνοντας σημαντικά το κόστος εργαστηριακού εξοπλισμού.

Ακολουθώντας μια βιβλιογραφική ανασκόπηση στα πειράματα απομακρυσμένης πρόσβασης [9] και συνοψίζοντας τους επικρατέστερους τρόπους υλοποίησης εξοπλισμού συλλογής μετρήσεων από απόσταση [10], στα πλαίσια της διδακτορικής διατριβής πραγματοποιήθηκε αρχικά αποτίμηση των πειραμάτων απλής και διαφορικής βαρομετρικής υψομετρίας με τη χρήση του πρωτότυπου ερευνητικού εξοπλισμού [11,12] που αναπτύχθηκε για τις ανάγκες του Εργαστηρίου Εκπαίδευσης και Διδασκαλίας της Φυσικής του Παιδαγωγικού Τμήματος Δημοτικής Εκπαίδευσης του Πανεπιστημίου Ιωαννίνων). Εν συνέχεια πραγματοποιήθηκε αποτίμηση των προτεινόμενων εκπαιδευτικών πειραμάτων και του αντίκτυπου αυτών στην εκμάθηση των φοιτητών [13,14]. Τέλος προτάθηκε, η υλοποίηση εξοπλισμού για τη μέτρηση υψομετρικής διαφοράς, κατάλληλα σχεδιασμένου για την υποστήριξη της εργαστηριακής εκπαίδευσης ατόμων με κινητική αναπηρία [15].

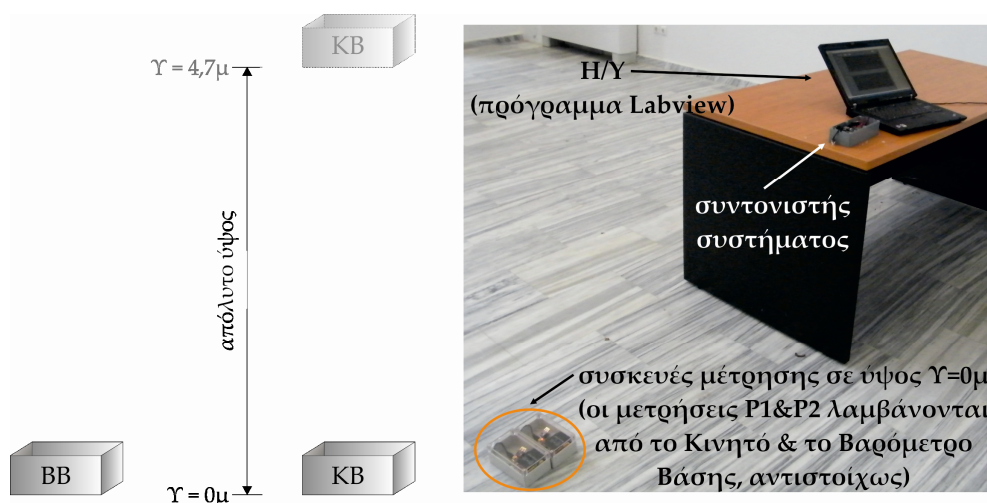
Αποτίμηση απλής και διαφορικής βαρομετρικής υψομετρίας

Για την αξιολόγηση της ορθής λειτουργίας και ομαλής διεξαγωγή πειραμάτων με τη χρήση του πρωτότυπου ερευνητικού εξοπλισμού, πραγματοποιήθηκε αποτίμησης της ακρίβειας υπολογισμού απόλυτου ύψους στην περίπτωση εφαρμογής της τεχνικής απλής και διαφορικής βαρομετρικής υψομετρίας. Μετρήσεις ελήφθησαν σε περιβάλλον τόσο εσωτερικού, όσο και εξωτερικού χώρου, και διερευνήθηκαν τα πλεονεκτήματα και μειονεκτήματα των δύο αυτών τεχνικών μέτρησης απόλυτου ύψους. Τα ερευνητικά αποτελέσματα της έρευνας αυτής παρουσιάζονται στο άρθρο “A Prototype Wireless Sensor Network System for a Comparative Evaluation of Differential and Absolute Barometric Altimetry” του περιοδικού *IEEE Aerospace and Electronic Systems Magazine*.

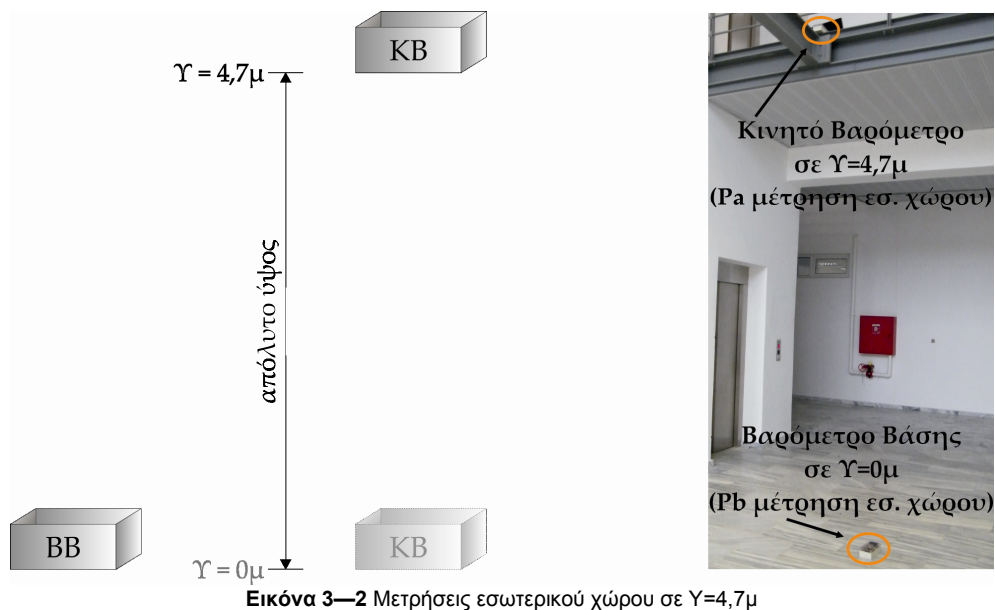
Εν συνεχεία πραγματοποιήθηκε αποτίμηση της ακρίβειας εκτίμησης απόλυτου ύψους στην περίπτωσης μέτρησης της ατμοσφαιρικής – μόνο – πίεσης και τη χρήση της διεθνούς βαρομετρικής εξίσωσης, και στην περίπτωση επιπρόσθετης μέτρησης της θερμοκρασίας και τη χρήσης της υψομετρικής εξίσωσης. Τα ερευνητικά αποτελέσματα της έρευνας αυτής παρουσιάζονται στο άρθρο “Temperature Influence on Differential Barometric Altitude Measurements”, του συνεδρίου *The 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS'2015), Warsaw, Poland, 2015*.

Μετρήσεις εσωτερικού χώρου

Στη συνέχεια παρουσιάζονται τα ερευνητικά αποτελέσματα αποτίμησης των τεχνικών απλής και διαφορικής βαρομετρικής υψομετρίας σε περιβάλλον εσωτερικού χώρου. Η ανάλυση των μετρήσεων πραγματοποιείται στο πρόγραμμα Matlab.



Εικόνα 3—1 Μετρήσεις εσωτερικού χώρου σε $Y=0\mu$

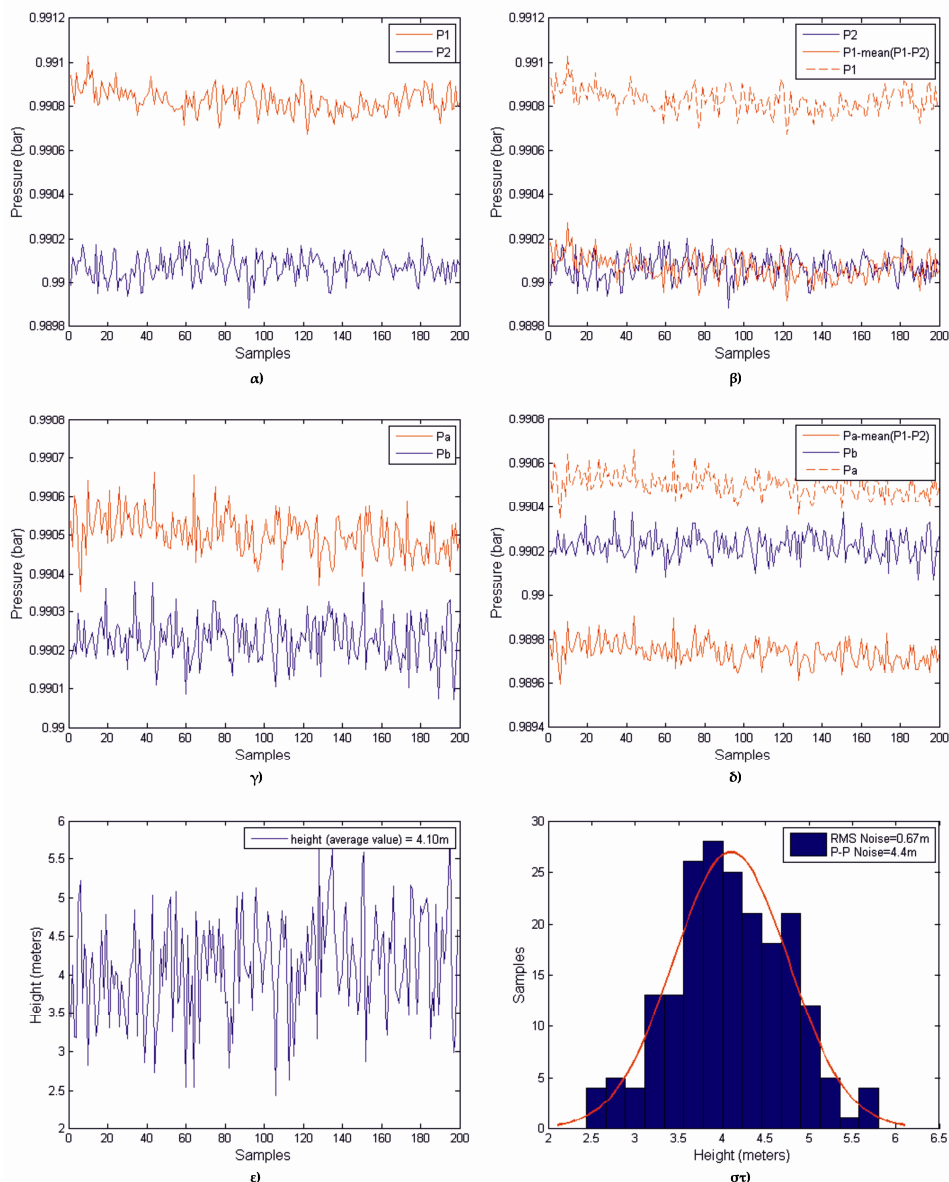


Εικόνα 3—2 Μετρήσεις εσωτερικού χώρου σε $\Upsilon=4,7\mu$

Στην **Εικόνα 3—1** δίνεται η τοποθέτηση των δύο συσκευών σε ύψος $\Upsilon=0\mu$ για τη βαθμονόμηση των αισθητήρων του βαρομέτρου βάσης (BB) και κινητού βαρομέτρου (KB). Όπως προαναφέρθηκε σε προηγούμενο κεφάλαιο, παρατηρείται απόκλιση στο σήμα εξόδου δύο αισθητήρων που μετρούν την ίδια τιμή ατμοσφαιρικής πίεσης [16]. Η απόκλιση αυτή πρέπει να εκτιμηθεί και να απομακρυνθεί από τις τιμές ατμοσφαιρικής πίεσης για την ορθή εκτίμηση του απόλυτου ύψους. Κατόπιν, το κινητό βαρόμετρο τοποθετείται σε διαφορά απόλυτου ύψους $\Upsilon=4,7\mu$ υψηλότερα από το βαρόμετρο βάσης (**Εικόνα 3—2**).

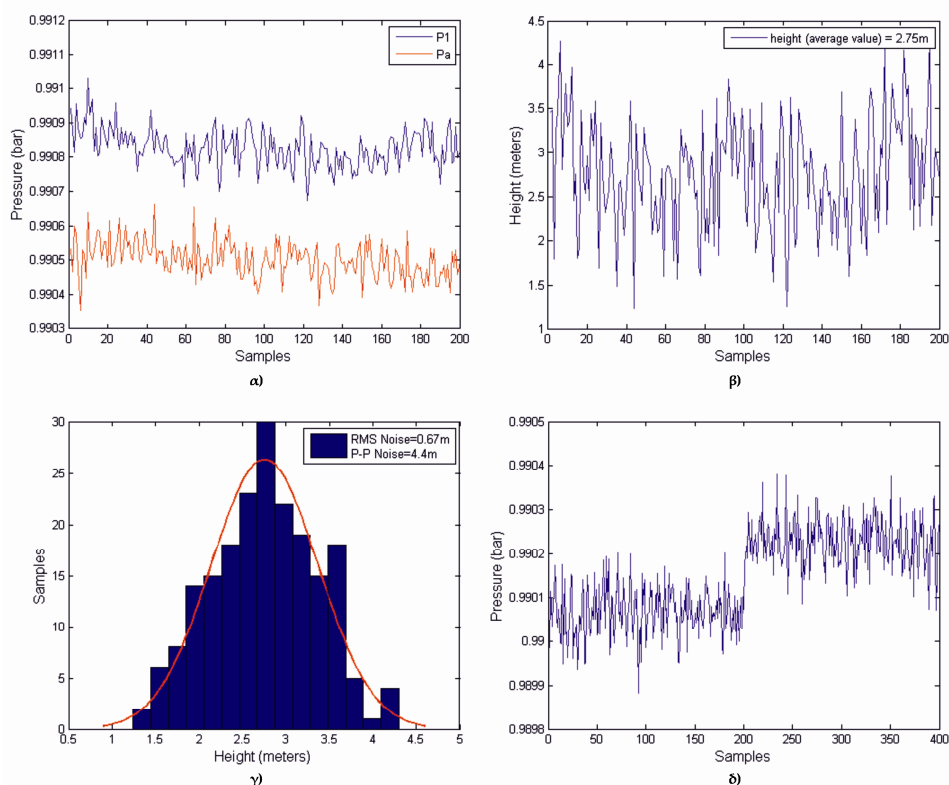
Στην **Εικόνα 3—3** πραγματοποιείται ανάλυση των μετρήσεων εσωτερικού χώρου με την τεχνική διαφορικής βαρομετρικής υψομετρίας (ΔΒΥ). Αναλυτικά, στην **Εικόνα 3—3α**) δίνονται οι μετρήσεις ατμοσφαιρικής πίεσης που λαμβάνονται από το κινητό βαρόμετρο (P1, κόκκινη γραφική) και το βαρόμετρο βάσης (P2, μπλε γραφική) σε ύψος $\Upsilon=0\mu$. Στην **Εικόνα 3—3β**) υπολογίζεται η μέση τιμή της διαφορά P1-P2, και η τιμή αυτή αφαιρείται από τα αρχικά δεδομένα του κινητού βαρομέτρου (P1, κόκκινη διακεκομμένη γραφική) για την απομάκρυνση της απόκλισης από το σήμα εξόδου του τελευταίου (P1, κόκκινη γραφική). Στην **Εικόνα 3—3γ**) δίνονται οι μετρήσεις ατμοσφαιρικής πίεσης που λαμβάνονται από το κινητό βαρόμετρο (P_a , κόκκινη γραφική) σε ύψος $\Upsilon=4,7\mu$ και το βαρόμετρο βάσης (P_b , μπλε γραφική) σε ύψος $\Upsilon=0\mu$. Επειδή σε αυτήν την περίπτωση το κινητό βαρόμετρο βρίσκεται σε υψηλότερο υψόμετρο από το βαρόμετρο βάσης, παρατηρείται μείωση στη μέτρηση ατμοσφαιρικής πίεσης του κινητού βαρομέτρου (P_a , κόκκινη γραφική), σε σχέση με την αντίστοιχη μέτρηση (P1, κόκκινη γραφική) της **Εικόνας 3—3α**). Συνεπώς, μειώνεται και η απόκλιση του σήματος εξόδου των αισθητήρων. Στην **Εικόνα 3—3δ**)

αφαιρείται από τη μέτρηση ατμοσφαιρικής πίεσης του κινητού βαρομέτρου σε $\gamma=4,7\mu$ (P1, κόκκινη διακεκομμένη γραφική), η μέση τιμή της διαφορά P1-P2 που υπολογίστηκε προηγουμένως. Όπως είναι αναμενόμενο, η διορθωμένη ατμοσφαιρική πίεση του κινητού βαρομέτρου που προκύπτει (P1, κόκκινη γραφική), είναι μικρότερη από την ατμοσφαιρική πίεση του βαρομέτρου βάσης που βρίσκεται σε χαμηλότερο υψόμετρο.



Εικόνα 3—3 Ανάλυση μετρήσεων εσωτερικού χώρου (τεχνική ΔY)

Στην **Εικόνα 3—3ε)** δίνεται η γραφική και υπολογίζεται μέση τιμή (4,1μ) του απόλυτου ύψους μεταξύ των δύο συσκευών μέτρησης, όπως προκύπτει από τις τιμές ατμοσφαιρικής πίεσης Pa (διακεκομμένη κόκκινη γραφική) και Pb (μπλε γραφική) της **Εικόνα 3—3δ)**. Η εκτίμηση του απόλυτου ύψους πραγματοποιείται με τη διεθνή βαρομετρική εξίσωση. Στην **Εικόνα 3—3στ)** υπολογίζεται η RMS και Peak-to-Peak (P-P) τιμή θορύβου του απόλυτου ύψους. Δεδομένου ότι ο θόρυβος στους αισθητήρες βαρομετρικής πίεσης είναι λευκός, και επειδή στις μετρήσεις εσωτερικού περιβάλλοντος μικρής χρονικής διάρκειας (1-2 λεπτά στη συγκεκριμένη περίπτωση) παρατηρούνται μικρές μεταβολές στην ατμοσφαιρική πίεση του χώρου, ο θόρυβος του απόλυτου ύψους ακολουθεί Γκαουσιανή κατανομή. Από την ανάλυση των μετρήσεων με την τεχνική διαφορικής βαρομετρικής υψομετρίας προκύπτει ότι η εκτίμηση απόλυτου ύψους (4,1μ) βρίσκεται πολύ κοντά στην θεωρητική τιμή (4,7μ).



Εικόνα 3—4 Ανάλυση μετρήσεων εσωτερικού χώρου (τεχνική ABY)

Στην **Εικόνα 3—4** πραγματοποιείται ανάλυση των ίδιων μετρήσεων με την τεχνική απλής βαρομετρικής υψομετρίας (ΔΒΥ), λαμβάνοντας δηλαδή υπόψη τις μετρήσεις της κινητής – μόνο – συσκευής του συστήματος. Αναλυτικά, στην **Εικόνα**

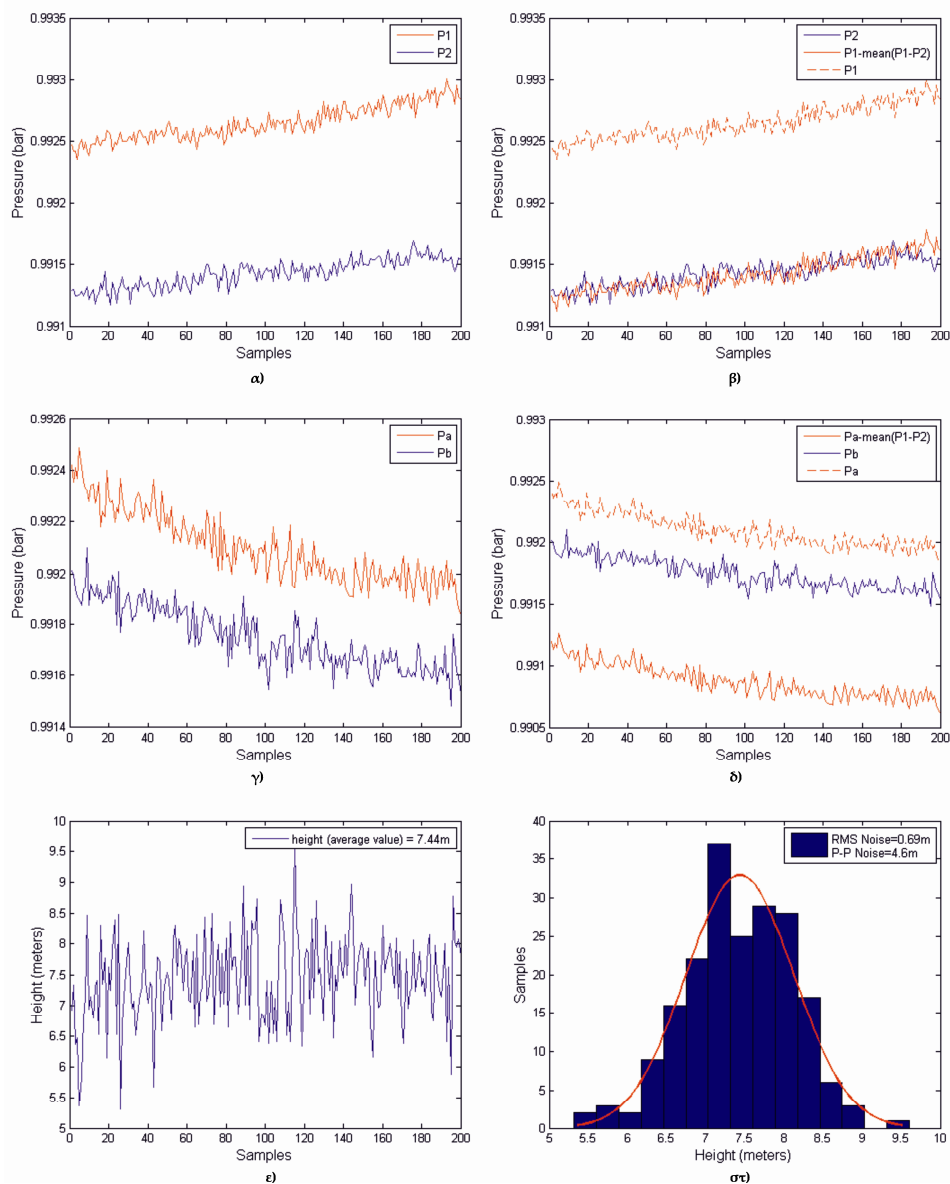
3—4α) δίνονται οι μετρήσεις ατμοσφαιρικής πίεσης που λαμβάνονται από το κινητό βαρόμετρο σε ύψος $Y=0\mu$ (P1, μπλε γραφική) και σε ύψος $Y=4,7\mu$ (Pa, κόκκινη γραφική). Στην **Εικόνα 3—4β)** δίνεται η γραφική και υπολογίζεται μέση τιμή (2,75μ) του απόλυτου ύψους, όπως προκύπτει από τις τιμές ατμοσφαιρικής πίεσης P1 και Pa. Στην **Εικόνα 3—4γ)** υπολογίζεται η RMS και Peak-to-Peak (P-P) τιμή θορύβου του απόλυτου ύψους και δίνεται η κατανομή θορύβου του τελευταίου. Από την ανάλυση των μετρήσεων με την τεχνική απλής βαρομετρικής υψομετρίας παρατηρείται μεγαλύτερο σφάλμα στην εκτίμηση απόλυτου ύψους (2,75μ) σε σύγκριση με την τεχνική διαφορικής βαρομετρικής υψομετρίας (4,1μ). Το σφάλμα αυτό προκύπτει από μεταβολές στην ατμοσφαιρική πίεση του χώρου, λόγω της χρονικής διάρκεια που απαιτείται για την τοποθέτηση του κινητού βαρομέτρου από τη θέση $Y=0\mu$ στη θέση $Y=4,7\mu$ και την επανάληψη καταγραφής μετρήσεων. Το μικρό αυτό χρονικό διάστημα (1-2 λεπτά) είναι αρκετό για να εισάγει σημαντικό σφάλμα στις μετρήσεις.

Στην **Εικόνα 3—4δ)** δίνεται η καταγραφή της ατμοσφαιρικής πίεσης από βαρόμετρο βάσης κατά τη διεξαγωγή του πειράματος απλής βαρομετρικής υψομετρίας. Τη χρονική στιγμή που τοποθετείται το κινητό βαρόμετρο στα 4,7μ και ξεκινάει η δειγματοληψία μετρήσεων, το βαρόμετρο βάσης καταγράφει μια μικρή μεταβολή στην ατμοσφαιρική πίεση του χώρου κατά 160μbar. Ανάγοντας την τιμή αυτή σε υψόμετρο με τη διεθνή βαρομετρική εξίσωση, προκύπτει σφάλμα ύψους της τάξης των 1,35μ. Μπορούμε λοιπόν να καταλήξουμε στο συμπέρασμα ότι στη διαφορική βαρομετρική υψομετρία, οι μεταβολές στην ατμοσφαιρική πίεση του χώρου γίνονται ταυτόχρονα αντιληπτές και από τις δύο συσκευές μέτρησης και συνεπώς, επηρεάζουν λιγότερο την εκτίμηση του απόλυτου ύψους. Το φαινόμενο αυτό είναι πιο ευδιάκριτο σε περιβάλλον εξωτερικού χώρου, όπου όπως θα δούμε στη συνέχεια, η τεχνική απλής βαρομετρικής υψομετρίας κρίνεται ακατάλληλη για τη μέτρηση απόλυτου ύψους σε εξωτερικούς χώρους λόγω των έντονων μεταβολών στην πίεση του ατμοσφαιρικού αέρα.

Μετρήσεις εξωτερικού χώρου



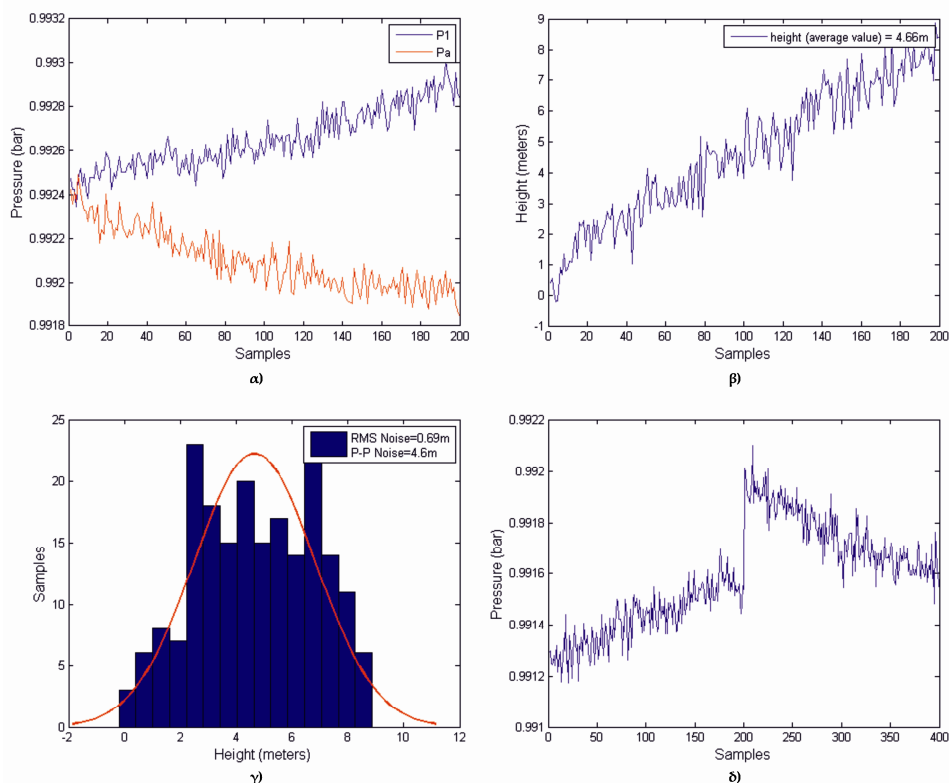
Στη συνέχεια παρουσιάζονται τα ερευνητικά αποτελέσματα αποτίμησης των τεχνικών απλής και διαφορικής βαρομετρικής υψομετρίας σε περιβάλλον εξωτερικού χώρου. Στην **Εικόνα 3—5** δίνεται η τοποθέτηση των δύο συσκευών σε ύψος $\Upsilon=6.62\mu$, αφού έχει προηγηθεί βαθμονόμηση των αισθητήρων με την τοποθέτηση των συσκευών μέτρησης στη θέση $\Upsilon=0\mu$.



Εικόνα 3—6 Ανάλυση μετρήσεων εξωτερικού χώρου (τεχνική ΔBY)

Στην **Εικόνα 3—6** πραγματοποιείται ανάλυση των μετρήσεων εξωτερικού χώρου με την τεχνική διαφορικής βαρομετρικής υψομετρίας ακολουθώντας τη διαδικασία που περιγράφηκε προηγουμένως. Η έντονη μεταβολή της ατμοσφαιρικής πίεσης λόγω των καιρικών φαινομένων είναι εμφανής στις γραφικές P1, P2 της **Εικόνας 3—6α)** και στις γραφικές Pa,Pb της **Εικόνας 3—6γ)**. Όμως επειδή η μεταβολή αυτή εφαρμόζεται ταυτόχρονα και στις δύο συσκευές μέτρησης, η εκτίμηση του απόλυτου ύψους (7,44μ) είναι αρκετά κοντά στην θεωρητική τιμή (6,62μ).

Στην **Εικόνα 3—7** πραγματοποιείται ανάλυση των μετρήσεων εξωτερικού χώρου με την τεχνική απλής βαρομετρικής υψομετρίας ακολουθώντας τη διαδικασία που περιγράφηκε προηγουμένως. Από τη γραφική του απόλυτου ύψους στην **Εικόνα 3—7β)**, αλλά και από την κατανομή των δειγμάτων αυτών (που δε μπορεί να θεωρηθεί πλέον Γκαουσιανής μορφής) στην **Εικόνα 3—7γ)**, γίνεται αντιληπτό ότι η τεχνική αυτή είναι ακατάλληλη για εξωτερικά περιβάλλοντα.



Εικόνα 3—7 Ανάλυση μετρήσεων εξωτερικού χώρου (τεχνική ABY)

Στον **Πίνακα 3—1** συνοψίζεται το % σφάλμα απόλυτου ύψους σε κάθε περίπτωση, χρησιμοποιώντας **Τύπο 3—1**.

Πίνακας 3—1 Σφάλμα απόλυτου ύψους

Περιβάλλον	Σφάλμα απόλυτου ύψους (ΔΒΥ)	Σφάλμα απόλυτου ύψους (ΑΒΥ)
Εσωτερικός χώρος	12,76%	41,49%
Εξωτερικός χώρος	12,38%	29,61%

$$\% \text{σφάλμα} = \left| \frac{\text{πειραματική τιμή} - \text{θεωρητική τιμή}}{\text{θεωρητική τιμή}} \right| \cdot 100$$

Τύπος 3—1 Υπολογισμός σφάλματος απόλυτου ύψους

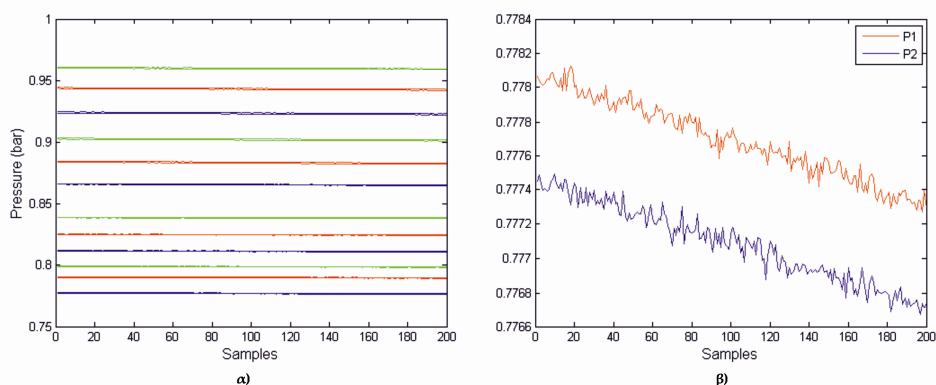
Μετρήσεις σε αεροστεγές δοχείο



Εικόνα 3—8 Μετρήσεις σε αεροστεγές δοχείο

Στα πλαίσια αποτίμησης της διαφορικής βαρομετρικής υψομετρίας διερευνήθηκε, με την αξιοποίηση της ασύρματης ζεύξης των συσκευών μέτρησης, η δυνατότητα μιας και μοναδικής βαθμονόμησης των αισθητήρων του συστήματος. Για την αποτίμηση αυτή χρησιμοποιήθηκε αεροστεγές δοχείο με αντλία αφαίρεσης του αέρα (**Εικόνα 3—8**).

Λήφθηκαν λοιπόν μετρήσεις σε διάφορα επίπεδα πίεσης (**Εικόνα 3—9**) και αξιολογήθηκε η απόκλιση στο σήμα εξόδου των αισθητήρων κάτω από τις ίδιες συνθήκες πίεσης, θερμοκρασίας, υγρασίας, κλπ. Η πτωτική κλίση των γραφικών πίεσης στην εικόνα **Εικόνα 3—9β**) οφείλεται στο γεγονός ότι η καταγραφή δεδομένων πραγματοποιήθηκε τοποθετώντας τη διάταξη μέτρησης σε ψυγείο. Ως γνωστό, η πίεση κλειστού αέρα μειώνεται κατά την πτώση της θερμοκρασίας. Αξίζει να σημειωθεί ότι η πειραματική αυτή διάταξη αποτελεί έναν εξαιρετικά χαμηλού κόστους εξοπλισμό μέτρησης, σε σύγκριση με τον εξαιρετικά ακριβό εξοπλισμό που χρησιμοποιείται για το χαρακτηρισμό αισθητήρων πίεσης (δηλ. ελεγκτή και σταθεροποιητή πίεσης, φιάλη αζώτου, κλπ).



Εικόνα 3—9 Ανάλυση μετρήσεων σε αεροστεγές δοχείο

Στον **Πίνακα 3—2** δίνεται η μέση απόκλιση του σήματος εξόδου των αισθητήρων, καθώς και η μέση απόκλιση απόλυτου ύψους (με την εφαρμογή της διεθνούς βαρομετρικής εξίσωσης), από 12 τυχαία επίπεδα πίεσης. Λόγω των τυχαίων αλλαγών της απόκλισης στο σήμα εξόδου των αισθητήρων, οδηγούμαστε στο συμπέρασμα ότι δεν είναι δυνατή η μία και μοναδική βαθμονόμηση των αισθητήρων συστήματος διαφορικής βαρομετρικής υψομετρίας. Η απόκλιση αυτή οφείλεται πιθανότατα στην καμπύλη απόκρισης των αισθητήρων, η οποία είναι διαφορετική για κάθε αισθητήρα λόγω ενδογενών κατασκευαστικών ανομοιοτήτων. Συνεπώς, για την εκάστοτε μέτρηση απόλυτου ύψους με σύστημα διαφορικής βαρομετρικής υψομετρίας θα πρέπει προπορεύεται η διαδικασία βαθμονόμησης των συσκευών μέτρησης που περιγράφηκε προηγουμένως.

Πίνακας 3—2 Απόκλιση στο σήμα εξόδου δύο αισθητήρων σε διάφορες πιέσεις

Ενδεικτικό επίπεδο πίεσης	Μέση απόκλιση πίεσης	Μέση απόκλιση απόλυτου ύψους
0,78 bar	606 μbar	6,25 μ
0,79 bar	625 μbar	6,36 μ
0,80 bar	507 μbar	5,12 μ
0,81 bar	584 μbar	5,82 μ
0,82 bar	498 μbar	4,90 μ
0,84 bar	470 μbar	4,56 μ
0,86 bar	514 μbar	4,86 μ
0,88 bar	908 μbar	8,43 μ
0,90 bar	909 μbar	8,30 μ
0,92 bar	863 μbar	7,74 μ
0,94 bar	850 μbar	7,49 μ
0,96 bar	773 μbar	6,71 μ

Ακρίβεια μέτρησης απόλυτου ύψους με την περαιτέρω μέτρηση θερμοκρασίας

Στα πλαίσια αποτίμησης της ακρίβειας εκτίμησης απόλυτου ύψους, διερευνήθηκε η ακρίβεια μετρήσεων χρησιμοποιώντας α) τη διεθνή βαρομετρική εξίσωση και β) την υψομετρική εξίσωση. Η πρώτη εξίσωση είναι σε συμφωνία με το *ιδεατό μοντέλο ατμόσφαιρας ISA (International Standard Atmosphere)*, σύμφωνα με το οποίο η ατμοσφαιρική πίεση και θερμοκρασία στο επίπεδο της θάλασσας ισούνται με 1023,24 Pa και 15 °C, αντιστοίχως, και συνεπώς δεν απαιτούνται μετρήσεις θερμοκρασίας για τον υπολογισμό του απόλυτου ύψους (**Τύπο 3—2α**). Η δεύτερη εξίσωση χρησιμοποιεί τη μέση τιμή θερμοκρασίας ανάμεσα στις δύο συσκευές μέτρησης για τον υπολογισμό της ατμοσφαιρικής κλίμακας ύψους *H* (**Τύπο 3—2β**).

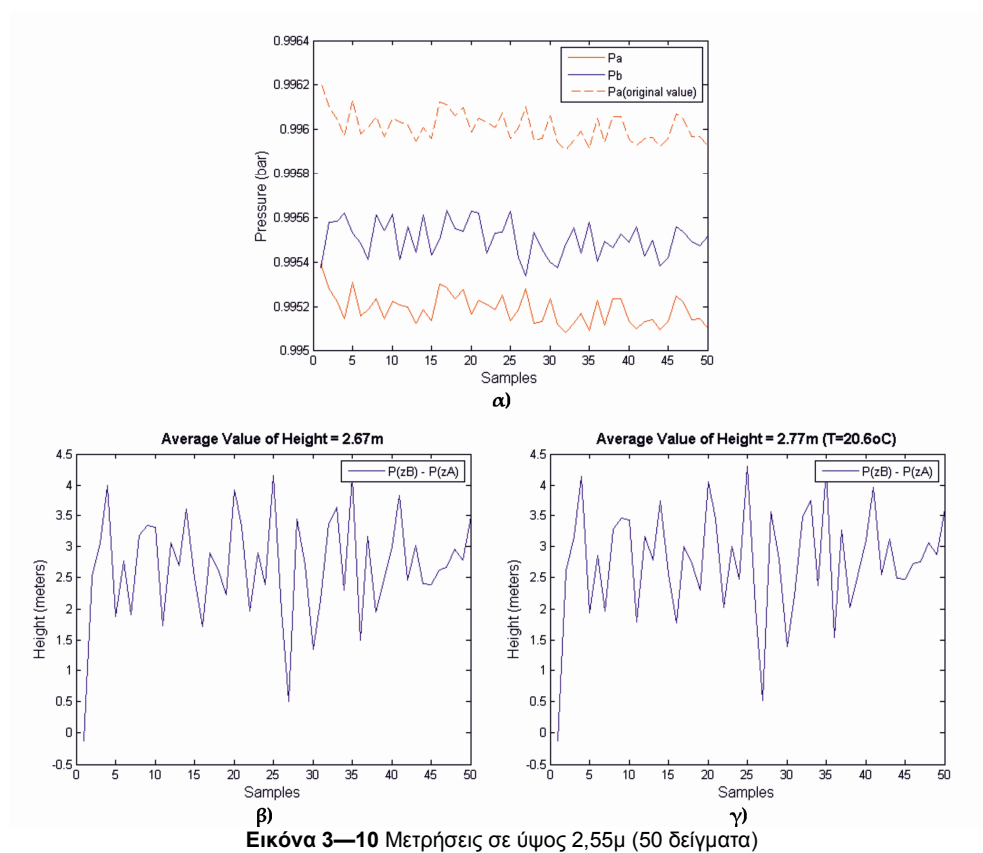
$$\begin{aligned}
 \alpha) h &= P_{(zA)} - P_{(zB)} = \\
 &= -4935,125 \cdot P_{(zA)}^{0.1903} + 4935,125 \cdot P_{(zB)}^{0.1903} = \\
 &= 4935,125 \cdot \left(P_{(zB)}^{0.1903} - P_{(zA)}^{0.1903} \right), \text{ όπου: } P_{(zA)} < P_{(zB)}
 \end{aligned}$$

$$\begin{aligned}
 \beta) h &= P_{(zA)} - P_{(zB)} = \\
 &= H \cdot \ln P_0 - H \ln P_{(zA)} - H \cdot \ln P_0 + H \ln P_{(zB)} = \\
 &= H \left(\ln P_{(zB)} - \ln P_{(zA)} \right), \text{ όπου: } P_{(zA)} < P_{(zB)} \\
 \text{και } H &= \frac{kT}{mg} \text{ (Ατμοσφαιρική κλίμακα ύψους)}
 \end{aligned}$$

Τύπος 3—2 Διεθνής βαρομετρική και υψομετρική εξίσωση

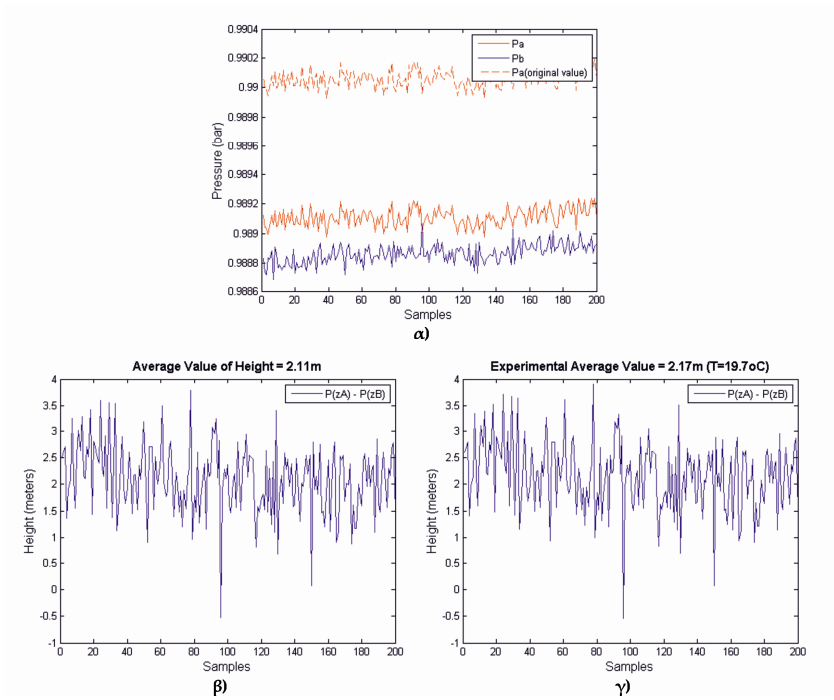
Για την εκτίμηση της ακρίβειας απόλυτου ύψους με τη χρήση των δύο εξισώσεων χρησιμοποιήθηκαν τρία σετ μετρήσεων με διαφορετικό αριθμό δειγμάτων. Τα δύο πρώτα πραγματοποιήθηκαν σε θερμοκρασία περιβάλλοντος και θεωρητικό ύψος

2,55μ (50 δείγματα μετρήσεων) και 2,26μ (200 δείγματα μετρήσεων), αντίστοιχως. Στο τελευταίο σετ επιχειρήθηκε μία πιο παράτολμη μέτρηση, τοποθετώντας τη μία εκ των δύο συσκευών μέτρησης στο ψυγείο για μερικά λεπτά ώστε να μειωθεί η θερμοκρασία της συσκευής και κατόπιν, πραγματοποιώντας μετρήσεις (εκτός ψυγείου) σε θεωρητικό ύψος 1,63μ (8 δείγματα μετρήσεων).

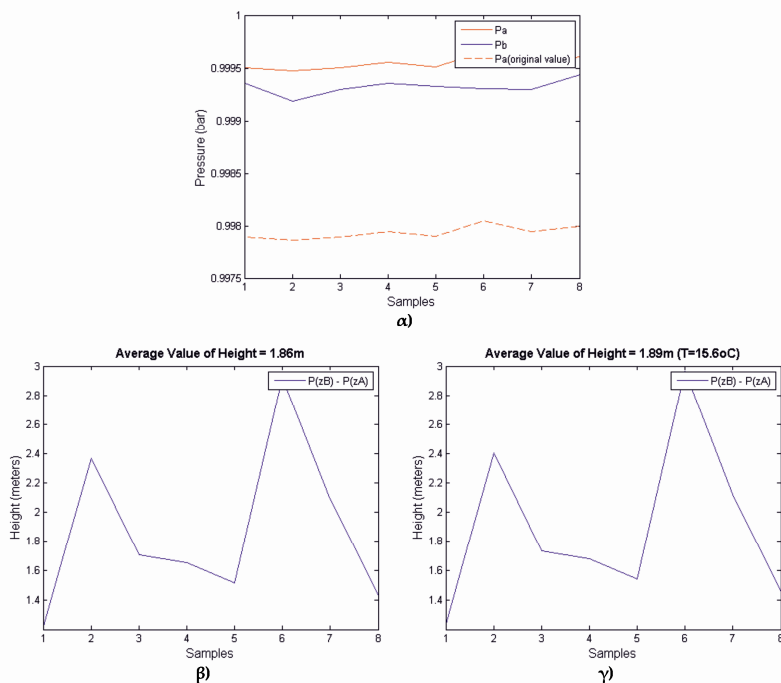


Εικόνα 3—10 Μετρήσεις σε ύψος 2,55μ (50 δείγματα)

Στην **Εικόνα 3—9** παρουσιάζεται το πρώτο σετ μετρήσεων. Αναλυτικά, στην **Εικόνα 3—9α)** δίνονται οι μετρήσεις ατμοσφαιρικής πίεσης P_a και P_b των δύο συσκευών με την απαραίτητη διόρθωση της απόκλισης στο σήμα εξόδου των αισθητήρων στην αρχική τιμή της γραφική P_a (κόκκινη διακεκομμένη γραμμή). Στις **Εικόνες 3—9β)** και **3—9γ)** δίνονται οι γραφικές και η μέση τιμή απόλυτου ύψους, όπως υπολογίστηκαν με τη χρήση της διεθνούς βαρομετρικής (2.67μ) και υψομετρικής εξίσωσης (2.77μ), αντίστοιχως. Στην τελευταία εικόνα δίνεται επίσης και η μέση τιμή θερμοκρασίας ανάμεσα στις συσκευές μέτρησης (20,6°C). Στις **Εικόνες 3—10** και **3—11** παρουσιάζονται οι αντίστοιχες γραφικές για τα δύο επόμενα σετ μετρήσεων.



Εικόνα 3—11 Μετρήσεις σε ύψος 2,26μ (200 δείγματα)



Εικόνα 3—12 Μετρήσεις σε ύψος 1,63μ (8 δείγματα)

Στη συνέχεια πραγματοποιείται στατιστική συσχέτιση (*statistical inference*) των τριών σετ μετρήσεων εφαρμόζοντας τη θεωρία εκτίμησης (*estimation theory*), ώστε να εξάγουμε συμπεράσματα όσον αφορά τα χαρακτηριστικά πληθυσμού των δειγμάτων. Γνωρίζοντας ότι ο αισθητήρας βαρομετρικής πίεσης παρεμβάλλεται από λευκό θόρυβο, υποθέτουμε ότι η γραφική του απόλυτου ύψους που προκύπτει από τις μετρήσεις ατμοσφαιρικής πίεσης θα έχει κανονική κατανομή, με γνωστή μέση τιμή πληθυσμού μ και άγνωστη τυπική απόκλιση πληθυσμού σ . Συνεπώς, χρησιμοποιείται η κατανομή του t κατά *Student* για τη στατιστική εκτίμηση των μετρήσεων. Η εκτίμηση αυτή είναι κατάλληλη και για το δείγμα των μετρήσεων με αριθμό δειγμάτων $n < 30$ (δηλαδή στο τελευταίο σετ μετρήσεων όπου $n=8$).

Έχουμε τρεις διαφορετικές μετρήσεις ατμοσφαιρικής πίεσης σε απόλυτο ύψος 2.55m, 2.26m, 1.63m με αριθμό δειγμάτων $n=50$, $n=200$, και $n=8$ αντιστοίχως, όπου η εκτίμηση του απόλυτου ύψους πραγματοποιείται με την εφαρμογή της α) διεθνούς βαρομετρικής εξίσωσης και β) υψομετρικής εξίσωσης. Τα δεδομένα των τριών μετρήσεων έχουν ως εξής:

A) Μετρήσεις σε απόλυτο ύψος 2.55 μέτρα ($Y=2.55m$)

1. αριθμός δειγμάτων (samples): $n = 50$
2. μέση τιμή πληθυσμού (population mean): $\mu = 2.55m$
3. τυπική απόκλιση πληθυσμού (population standard deviation): $\sigma = \text{άγνωστη}$
4. αριθμητικός μέσος δειγματοληψίας (sample mean: *the natural estimator of population mean* μ) :
 - α) $\bar{X} = 2.67m$ (εφαρμογή της διεθνούς βαρομετρικής εξίσωσης)
 - β) $\bar{X} = 2.77m$ (εφαρμογή της υψομετρικής εξίσωσης)
5. τυπική απόκλιση πραγματικής κατανομής δειγματοληψίας (sample standard deviation: *the natural estimator of population standard deviation* σ):
 - α) $s = 0.86m$ (εφαρμογή της διεθνούς βαρομετρικής εξίσωσης)
 - β) $s = 0.89m$ (εφαρμογή της υψομετρικής εξίσωσης)

B) Μετρήσεις σε απόλυτο ύψος 2.26 μέτρα ($Y=2.26m$)

1. αριθμός δειγμάτων (samples): $n = 200$
2. μέση τιμή πληθυσμού (population mean): $\mu = 2.26m$
3. τυπική απόκλιση πληθυσμού (population standard deviation): $\sigma = \text{άγνωστη}$
4. αριθμητικός μέσος δειγματοληψίας (sample mean: *the natural estimator of population mean* μ) :
 - α) $\bar{X} = 2.11m$ (εφαρμογή της διεθνούς βαρομετρικής εξίσωσης)
 - β) $\bar{X} = 2.18m$ (εφαρμογή της υψομετρικής εξίσωσης)
5. τυπική απόκλιση πραγματικής κατανομής δειγματοληψίας (sample standard deviation: *the natural estimator of population standard deviation* σ):
 - α) $s = 0.65m$ (εφαρμογή της διεθνούς βαρομετρικής εξίσωσης)
 - β) $s = 0.67m$ (εφαρμογή της υψομετρικής εξίσωσης)

Γ) Μετρήσεις σε απόλυτο ύψος 1.63 μέτρα ($\mu=1.63m$)

1. αριθμός δειγμάτων (samples): $n = 8$
2. μέση τιμή πληθυσμού (population mean): $\mu = 1.63m$
3. τυπική απόκλιση πληθυσμού (population standard deviation): $\sigma = \text{άγνωστη}$
4. αριθμητικός μέσος δειγματοληψίας (sample mean: *the natural estimator of population mean* μ) :
 - α) $\bar{x} = 1.86m$ (εφαρμογή της διεθνούς βαρομετρικής εξίσωσης)
 - β) $\bar{x} = 1.89m$ (εφαρμογή της υψομετρικής εξίσωσης)
5. τυπική απόκλιση πραγματικής κατανομής δειγματοληψίας (sample standard deviation: *the natural estimator of population standard deviation* σ):
 - α) $s = 0.56m$ (εφαρμογή της διεθνούς βαρομετρικής εξίσωσης)
 - β) $s = 0.57m$ (εφαρμογή της υψομετρικής εξίσωσης)

Επισημαίνεται ότι ο υπολογισμός $\bar{x}$ και s γίνεται μέσω του προγράμματος Matlab, όπου η χρήση της συνάρτησης `std` εφαρμόζει την παρακάτω εξίσωση που δίδεται στον **Τύπο 3—3**.

$$s = \sqrt{\sum_{i=1}^n (x_i - \bar{x})^2 / (n-1)}$$

Τύπος 3—3 Υπολογισμός τυπικής απόκλισης s με τη συνάρτηση (Matlab) `std`

$$\bar{x} \pm \left(t_{n-1, 1-\alpha/2} \right) \cdot sem, \text{ όπου } sem = \frac{s}{\sqrt{n}}$$

όπου $(t \cdot sem)$ το περιθώριο σφάλματος (MOE)

$$\text{οπότε, } LCL = \bar{x} - \left(t_{n-1, 1-\alpha/2} \right) \cdot \left(\frac{s}{\sqrt{n}} \right) \text{ και } UCL = \bar{x} + \left(t_{n-1, 1-\alpha/2} \right) \cdot \left(\frac{s}{\sqrt{n}} \right)$$

Τύπος 3—4 Διάστημα εμπιστοσύνης για τη μέση τιμή πληθυσμού μ

Στα τρία σύνολα μετρήσεων πραγματοποιούνται υπολογισμοί για διαστήματα εμπιστοσύνης (CIs – *confidence intervals*) 70%, 90%, 95%, 99.9% (ώστε να εντοπιστεί το διάστημα εμπιστοσύνης στο οποίο εμφανίζεται το μ), όπου η πιθανότητα της μεταβλητής t (*upper-tail probability*) ορίζεται αντιστοίχως 0.15, 0.05, 0.025, 0.0005. Οι τελευταίες τιμές αντιστοιχούν στις στήλες του t -πίνακα που χρησιμοποιείται για την στατιστική εκτίμηση των μετρήσεων και προκύπτουν ως εξής:

Π.χ. για την περίπτωση διαστήματος εμπιστοσύνης 99.9% $\rightarrow 100-99.9=0.1\% \rightarrow \alpha$ (*lack of confidence*) $=0.1/100=0.001$ (*two-tails probability*) $\rightarrow \alpha/2=0.001/2=0.0005$ (*one-tail probability*). Οι γραμμές του t -πίνακα (γνωστές και ως βαθμοί ελευθερίας) προκύπτουν από τον αριθμό των δειγμάτων μέτρησης ως $n-1$. Π.χ. για την περίπτωση των 50 δειγμάτων η επιλογή της γραμμής του t -πίνακα ορίζεται ως $n-1=49$. Συνεπώς ο βαθμός t (t score) προκύπτει από την προσδιορισμό στήλης-γραμμής του πίνακα. Το διάστημα εμπιστοσύνης για τη μέση τιμή πληθυσμού μ δίνεται στον **Τύπο 3—4**,

όπου *sem* (*standard error of the mean*) το εκτιμώμενο τυπικό λάθος του μέσου.

Στον **Πίνακα 3—3** δίνεται η στατιστική ανάλυση των τριών σετ μετρήσεων καθώς και το σφάλμα απόλυτου ύψους σε κάθε περίπτωση. Από τη στατιστική ανάλυση προκύπτει ότι στις μετρήσεις απόλυτου ύψους σε περιβάλλον εσωτερικού χώρου, η ακρίβεια μετρήσεων δεν επηρεάζεται σημαντικά από την επιπλέον καταγραφή της θερμοκρασίας. Συνεπώς, η μη-καταγραφή θερμοκρασίας απλουστεύει τη σχεδίαση του συστήματος και ελαττώνει το χρόνο δειγματοληψίας μεταξύ διαδοχικών μετρήσεων, ο οποίος κρίνεται πιο σημαντικός για την ακρίβεια μέτρησης λόγω των αλλαγών ατμοσφαιρικής πίεσης με την πάροδο του χρόνου.

Πίνακας 3—3 Στατιστική ανάλυση μετρήσεων και σφάλμα απόλυτου ύψους

Εξίσωση υπολογισμού απόλυτου ύψους	μ	$\bar{x}$	s	t_{score}	LCL	UCL	MOE	CI	Σφάλμα ύψους
Διεθνής βαρομετρική εξίσωση	2,55μ	2,67μ	0,86μ	1,0480	2,54μ	2,80μ	12,7%	70%	4,71%
	2,26μ	2,11μ	0,65μ	3,3401	1,96μ	2,26μ	15,4%	99,9%	6,64%
	1,63μ	1,86μ	0,56μ	1,8946	1,48μ	2,24μ	37,5%	90%	14,1%
Υψομετρική εξίσωση	2,55μ	2,77μ	0,89μ	2,0096	2,52μ	3,02μ	25,3%	95%	8,63%
	2,26μ	2,18μ	0,67μ	1,9720	2,09μ	2,27μ	9,3%	95%	3,96%
	1,63μ	1,89μ	0,57μ	1,8950	1,51μ	2,27μ	38,2%	90%	15,8%

μ : μέση τιμή πληθυσμού (*population mean*)

$\bar{x}$: αριθμητικός μέσος δειγματοληψίας (*sample mean*)

s: τυπική απόκλιση πραγματικής κατανομής δειγματοληψίας (*sample standard deviation*)

t_{score} : βαθμός t που προκύπτει από την προσδιορισμό στήλης-γραμμής του πίνακα t

LCL: κατώτερο επίπεδο εμπιστοσύνης (*lower confidence limit*)

UCL: ανώτερο επίπεδο εμπιστοσύνης (*upper confidence limit*)

MOE: περιθώριο σφάλματος (*margin of error*)

CI: διάστημα εμπιστοσύνης (*confidence interval*)

Αποτίμηση εκπαιδευτικών πειραμάτων

Ο πρωτότυπος εξοπλισμός σχεδιάστηκε ώστε να παρέχει στους διδασκόμενους ικανοποιητικό επίπεδο πειραματισμού στη βαρομετρική υψομετρία και στους MEMS αισθητήρες βαρομετρικής πίεσης. Η λειτουργία λήψης μετρήσεων μέσω του δικτύου/διαδικτύου επιτρέπει την εγκατάσταση ενός συστήματος ανά εργαστήριο, μειώνοντας σημαντικά το κόστος εργαστηριακού εξοπλισμού. Επιπλέον, με τη λειτουργία αυτή δεν απαιτείται η ανάπτυξη συγκεκριμένου λογισμικού προγράμματος για τον έλεγχο του συστήματος.

Σύμφωνα με τα παραπάνω, οι διδασκόμενοι δεν εμπλέκονται σε αρχικοποιήσεις του υλισμικού ή/και λογισμικού για τον έλεγχο του συστήματος, αλλά συλλέγουν τις μετρήσεις μέσω ενός απλού προγράμματος περιήγησης στον Ιστό (π.χ. Chrome). Συνεπώς, η προσοχή τους είναι εστιασμένη στην καταγραφή και ανάλυση των δεδομένων. Ο κίνδυνος απόσπασης της προσοχής των διδασκόμενων (κατά την διεξαγωγή του εργαστηρίου) από χρονοβόρες διενέργειες αρχικοποίησης του υλισμικού ή/και λογισμικού μέρους του εξοπλισμού, αποτελεί ένα ζήτημα που συχνά απασχολεί την ακαδημαϊκή κοινότητα κατά την ανάπτυξη εργαστηριακών εξοπλισμών [17,18].

Επιπροσθέτως, η λειτουργία λήψης μετρήσεων μέσω ηλεκτρονικού υπολογιστή, σε συνδυασμό με τους κώδικες Matlab που αναπτύχθηκαν για την αυτόματη ανάλυση των μετρήσεων, παρέχει στο διδάσκοντα τη δυνατότητα επίδειξης της διαδικασίας πειραματισμού στη βαρομετρική υψομετρία. Μετά από τη διαδικασία της επίδειξης οι διδασκόμενοι μπορούν να λάβουν, μέσω του προγράμματος περιήγησης που εκτελείται στον ηλεκτρονικό υπολογιστή τους (ή και ακόμη στο κινητό τους τηλέφωνο), δύο σετ μετρήσεων από 10 (παραδείγματος χάριν) δείγματα το καθένα. Το πρώτο σετ προκύπτει κατά την τοποθέτηση των συσκευών μέτρησης στο ίδιο ύψος $Y=0\text{m}$ για τη βαθμονόμηση των συσκευών, ενώ το δεύτερο προκύπτει κατά την τοποθέτηση των συσκευών σε διαφορά απόλυτου ύψους $Y=X\text{m}$ για τον υπολογισμό του τελευταίου. Αξίζει να σημειωθεί ότι, η καταγραφή μετρήσεων μέσω του δικτύου/διαδικτύου μπορεί να πραγματοποιηθεί ταυτόχρονα¹ από όλους του φοιτητές, ενώ οι μετρήσεις που καταγράφονται είναι διαφορετικές κάθε φορά λόγω του λευκού θορύβου που χαρακτηρίζει του αισθητήρες βαρομετρικής πίεσης. Οι φοιτητές μπορούν να προβούν στην ανάλυση των δύο σετ μετρήσεων που κατέγραψαν και να διερευνήσουν τα ακόλουθα:

¹ Ο ενσωματωμένος εξυπηρετητής που υπάρχει στο συντονιστή του συστήματος, δέχεται αιτήματα από έναν πελάτη κάθε φορά. Παρόλα αυτά, η διαδικασία λήψης μετρήσεων είναι αρκετά σύντομη (περίπου 1sec) και συνεπώς, οι φοιτητές μπορούν ταυτόχρονα να λαμβάνουν μετρήσεις. Σε περίπτωση αδυναμίας φόρτωσης της σελίδας στον Ιστό, απαιτείται απλώς μία ανανέωση στη σελίδα του προγράμματος περιήγησης για τη σύνδεση στον ενσωματωμένο εξυπηρετητή.

Πίνακας 3—4 Ερωτηματολόγιο αξιολόγησης

α/α	Ερώτηση	Πάρα πολύ (5)	Πολύ (4)	Αρκετά (3)	Λίγο (2)	Καθόλου (1)	Μέσος όρος
1	Σας βοήθησε η εκπαιδευτική προσέγγιση να κατανοήσετε την έννοια βαρομετρικού υψομέτρου και την πρακτική εφαρμογή υπολογισμού απόλυτου ύψους με τη μέτρηση ατμοσφαιρικής πίεσης;	1	3	7	2	0	3,23
2	Σας βοήθησε η εκπαιδευτική προσέγγιση να κατανοήσετε τα πλεονεκτήματα διαφορικής βαρομετρικής υψομετρίας σε σύγκριση με την απλή βαρομετρική υψομετρία, όσον αφορά την ακρίβεια των μετρήσεων απόλυτου ύψους;	1	6	6	0	0	3,62
3	Σας βοήθησε η εκπαιδευτική προσέγγιση να κατανοήσετε τις ιδιαιτερότητες διαφορικής βαρομετρικής υψομετρίας σε σύγκριση με την απλή βαρομετρική υψομετρία, όσον αφορά τη βαθμονόμηση των αισθητήρων ατμοσφαιρικής πίεσης;	0	7	5	1	0	3,46
4	Σας βοήθησε η εκπαιδευτική προσέγγιση να κατανοήσετε τις διαφορές των δύο εξισώσεων που χρησιμοποιούνται για τον υπολογισμό απόλυτου ύψους και τις επικείμενες επιπτώσεις στις προδιαγραφές του συστήματος: α) διεθνής βαρομετρική εξίσωση και β) υψομετρική εξίσωση.	0	2	6	5	0	2,77
5	Θεωρείται ικανοποιητική την επίδειξη του συστήματος βαρομετρικής υψομετρίας, μέσω της λειτουργίας λήψης μετρήσεων από ηλεκτρονικού υπολογιστή και με τη χρήση των προγραμμάτων αυτόματης ανάλυσης δεδομένων στο Matlab;	1	8	4	0	0	3,77
6	Θεωρείται ικανοποιητική λειτουργία λήψης μετρήσεων με το προτεινόμενο σύστημα βαρομετρικής υψομετρίας, μέσω δικτύου/διαδικτύου (χρησιμοποιώντας το πρόγραμμα περιήγησης στον	2	4	4	2	1	3,31

Για τη ανάλυση αυτή απαιτείται η λήψη μετρήσεων σε αεροστεγές δοχείο σε διάφορα επίπεδα πίεσης. Οι φοιτητές μπορούν να λάβουν για παράδειγμα, 3-4 σετ μετρήσεων από 10 δείγματα το καθένα, και να εκτιμήσουν αν υπάρχει γραμμικότητα της απόκλιση στο σήμα εξόδου των αισθητήρων.

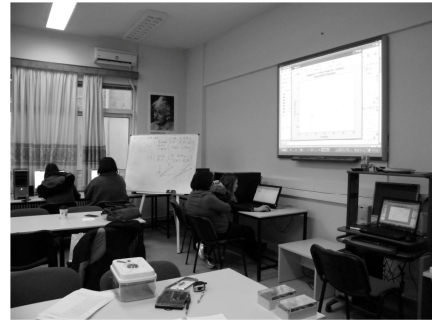
Από την τελευταία ανάλυση των μετρήσεων επιτυγχάνεται και ο έλεγχος των τεχνικών χαρακτηριστικών των MEMS αισθητήρων πίεσης, που κανονικά απαιτεί ιδιαίτερα ακριβό εξοπλισμό. Αξίζει να σημειωθεί ότι, τα προτεινόμενα πειράματα μπορούν να συνδυαστούν και με άλλα αντικείμενα διδασκαλίας, όπως π.χ., την ανάπτυξη κώδικα Matlab για την αυτόματη ανάλυση δεδομένων.

Δεκατρείς μεταπτυχιακοί φοιτητές κατά τη διάρκεια διεξαγωγής του εργαστηρίου Εκπαίδευσης και Διδασκαλίας της Φυσικής του ΠΤΔΕ Ιωαννίνων (**Εικόνες 3—13_{1,2}**), έλαβαν μέρος στη χρήση και αξιολόγηση του πρωτότυπου πειραματικού εξοπλισμού και των προτεινόμενων πειραμάτων βαρομετρικής υψομετρίας. Στις **Εικόνες 3—13₃₋₁₄** παρουσιάζονται 2 σετ μετρήσεων που έλαβαν οι φοιτητές κατά τη διάρκεια αξιολόγησης. Όσον αφορά το πρώτο σετ, η **Εικόνα 3—13₃** παρουσιάζει τις μετρήσεις που καταγράφηκαν από το βαρόμετρο βάσης (διακεκομμένη κόκκινη γραμμή) και το κινητό βαρόμετρο (μπλε γραμμή) στη θέση $Y=0\mu$, ενώ η κόκκινη γραμμή παρουσιάζει τις μετρήσεις του βαρόμετρου βάσης με την απαραίτητη διόρθωση στο σήμα εξόδου. Η **Εικόνα 3—13₄** παρουσιάζει τις μετρήσεις που καταγράφηκαν από το βαρόμετρο βάσης (διακεκομμένη κόκκινη γραμμή) στη θέση $Y=0\mu$ και το κινητό βαρόμετρο (μπλε γραμμή) στη θέση $Y=1,5\mu$, ενώ η κόκκινη γραμμή παρουσιάζει τις μετρήσεις του βαρόμετρου βάσης με την απαραίτητη διόρθωση στο σήμα εξόδου. Η **Εικόνα 3—13₅** παρουσιάζει τη γραφική απόλυτου ύψους εφαρμόζοντας την τεχνική διαφορικής βαρομετρικής υψομετρίας και χρησιμοποιώντας τη διεθνή βαρομετρική εξίσωση. Η **Εικόνα 3—13₆** παρουσιάζει τις μετρήσεις που καταγράφηκαν από το κινητό βαρόμετρο στη θέση $Y=0\mu$ (μπλε γραμμή) και στη θέση $Y=1,5\mu$ (κόκκινη γραμμή). Η **Εικόνα 3—13₇** παρουσιάζει τη γραφική απόλυτου ύψους εφαρμόζοντας την τεχνική απλής βαρομετρικής υψομετρίας και χρησιμοποιώντας τη διεθνή βαρομετρική εξίσωση. Η **Εικόνα 3—13₈** παρουσιάζει τις μεταβολές ατμοσφαιρικής πίεσης όπως καταγράφηκαν από το βαρόμετρο βάσης καθ' όλη τη διάρκεια λήψης μετρήσεων. Ακολουθώντας την ίδια ανάλυση δεδομένων, το δεύτερο σετ μετρήσεων παρουσιάζεται στις **Εικόνες 3—13₉₋₁₄**.

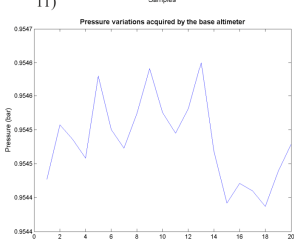
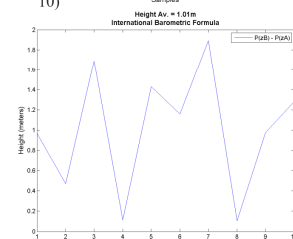
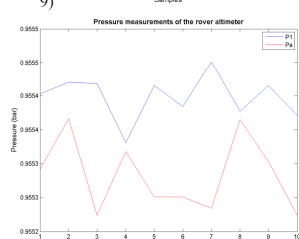
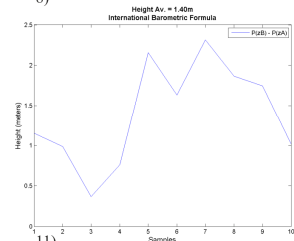
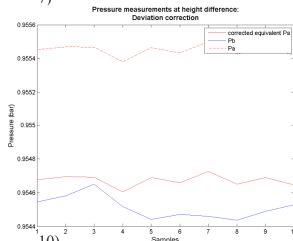
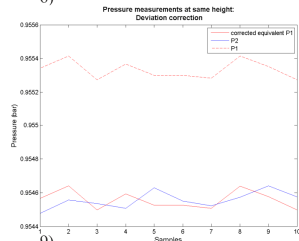
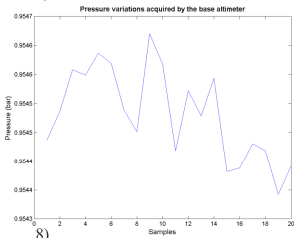
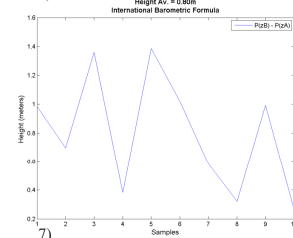
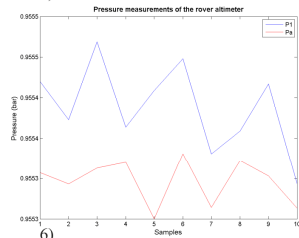
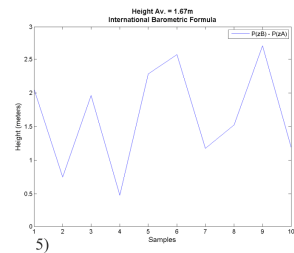
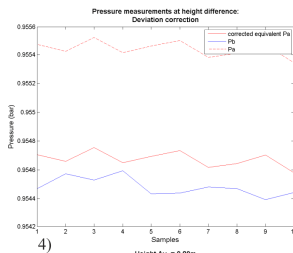
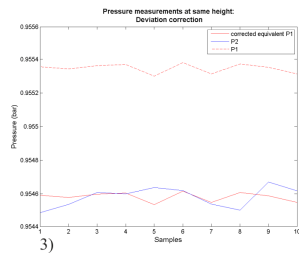
Στον **Πίνακα 3—4** δίνεται το ερωτηματολόγιο αξιολόγησης του πρωτότυπου συστήματος και της αξίας των προτεινόμενων πειραμάτων στη βαρομετρική υψομετρία που διανεμήθηκε στους φοιτητές του εργαστηρίου. Αξίζει να σημειωθεί ότι, 4 από τους φοιτητές διαθέτουν πτυχίο Φυσικής και παρακολουθούν το μεταπτυχιακό πρόγραμμα σπουδών (ΜΠΣ) «Ερευνα και Νέες Τεχνολογίες στη Διδασκαλία της Φυσικής», ενώ οι υπόλοιποι 9 είναι κάτοχοι διπλώματος του Παιδαγωγικού Τμήματος Δημοτικής Εκπαίδευσης και παρακολουθούν το μεταπτυχιακό πρόγραμμα σπουδών (ΜΠΣ) «Φυσικές Επιστήμες στην Εκπαίδευση».



1)



2)



Εικόνα 3—13 Εργαστήριο Διδασκαλίας Φ.Ε. (ΠΤΔΕ) & Μετρήσεις Φοιτητών

Οι πρώτες 4 ερωτήσεις του ερωτηματολογίου διερευνούν το ποσοστό στο οποίο

το προτεινόμενο εκπαιδευτικό πλαίσιο είχε αντίκτυπο στην κατανόηση των διδαχθέντων εννοιών που εξετάζονται από τα προτεινόμενα πειράματα Φυσικής. Αναλυτικά, η Ερ. 1 εξετάζει κατά πόσο η θεωρία και η πειραματική διαδικασία εκτίμησης απόλυτου ύψους έγινε κατανοητή από τους φοιτητές· η Ερ. 2 διερευνά κατά πόσο έγιναν κατανοητά τα πλεονεκτήματα της διαφορικής βαρομετρικής υψομετρίας όσον αφορά την ακρίβεια υπολογισμού του ύψους· η Ερ. 3 διερευνά κατά πόσο έγιναν κατανοητά τα μειονεκτήματα της διαφορικής βαρομετρικής υψομετρίας όσον αφορά τη βαθμονόμηση των συσκευών μέτρησης· η Ερ. 4 εξετάζει κατά πόσο κατανοήθηκαν οι επακόλουθες συνέπειες στο σχεδιασμό συστήματος κατά τη χρήση ενός εκ των δύο εξισώσεων που χρησιμοποιούνται για την αναγωγή των μετρήσεων ατμοσφαιρικής πίεσης σε υψόμετρο. Οι ερωτήσεις 5 και 6 εξετάζουν, αντιστοίχως, την αποτελεσματικότητα επίδειξης του συστήματος από το διδάσκοντα (μέσω του προγράμματος LabVIEW) και την αποτελεσματικότητας ελέγχου του συστήματος μέσω δικτύου/διαδικτύου. Οι ερωτήσεις 7 και 8 διερευνούν κατά πόσο η προτεινόμενη εκπαιδευτική διαδικασία τόνωσε το ενδιαφέρον τους για τα συστήματα βαρομετρικής υψομετρίας (Ερ. 7) και πόσο τους βοήθησε να κατανοήσουν τη θεωρία που συσχετίζεται με τη βαρομετρική υψομετρία (Ερ. 8), ενώ η ερώτηση 9 ζητά από τους φοιτητές να αξιολογήσουν τις ικανότητές τους σε τέτοιους είδους πειραματικές δραστηριότητες. Τέλος, οι 3 τελευταίες ερωτήσεις ζητούν από τους φοιτητές να εκφράσουν την άποψή τους αναφορικά με το εκπαιδευτικό επίπεδο του προτεινόμενου εκπαιδευτικού εξοπλισμού (Ερ. 10), των προτεινόμενων πειραμάτων (Ερ. 11), καθώς και της καινοτομίας αυτών σε σύγκριση με άλλα πειραματικής Φυσικής που γνωρίζουν ή έχουν διεξάγει στο παρελθόν (Ερ. 12).

Ο μέσος όρος βαθμολογίας των τριών πρώτων ερωτήσεων μπορεί να χαρακτηριστεί ικανοποιητικός, αν αναλογιστούμε ότι 9 από τους 13 φοιτητές προέρχονται από το χώρο παιδαγωγικών σπουδών και συνεπώς, δε διαθέτουν προηγούμενη εμπειρία σε παρόμοια περιβάλλοντα πειραματισμού. Ωστόσο η χαμηλή βαθμολογία στην ερώτηση 4 μπορεί να χαρακτηριστεί δίκαιη, καθώς η θεωρία των τύπων που χρησιμοποιούνται για την αναγωγή ατμοσφαιρικής πίεσης σε υψόμετρο αποδόθηκε αρκετά συνοπτικά, ώστε να αποφευχθεί η σύγχυση των φοιτητών από την πληθώρα των νέων πληροφοριών που συσχετίζονται με την προτεινόμενη εκπαιδευτική προσέγγιση. Στις ερωτήσεις 5 και 6 οι φοιτητές βρίσκουν πιο αποτελεσματική την επίδειξη του συστήματος από το διδάσκοντα σε σύγκριση με τη διαδικασία ελέγχου του συστήματος μέσω δικτύου/διαδικτύου. Αυτή είναι λογική κρίση καθώς ο πρώτος τρόπος λειτουργίας παρέχει στους διδασκόμενους γραφική αναπαράσταση των μετρήσεων σε πραγματικό χρόνο, και σε συνδυασμό με την γραφική ανάλυση δεδομένων στο Matlab, βοηθά σημαντικά την οικοδόμηση της σκέψης των διδασκόμενων. Στην ερώτηση 8 οι φοιτητές συμφωνούν ότι οι πειραματικές δραστηριότητες στη βαρομετρική υψομετρία παρέχουν καλύτερη κατανόηση των θεωρητικών εννοιών, από την άλλη όμως, στην ερώτηση 9 διαφωνούν με την άποψη ότι τους εξάπτουν το ενδιαφέρον στον τομέα αυτό. Αυτή η άποψη συσχετίζεται,

κατά κύριο λόγο, με το εκπαιδευτικό υπόβαθρο των περισσότερων φοιτητών (αυτών δηλαδή που προέρχονται από το χώρο των παιδαγωγικών σπουδών) και που κατ' επέκταση αντανακλάται στο χαμηλό βαθμό της ερώτησης 9. Τέλος, ιδιαίτερα ενθαρρυντική μπορεί να χαρακτηριστεί η βαθμολογία των τριών τελευταίων ερωτήσεων.

Στον **Πίνακα 3—5** δίνονται οι βαθμολογίες ερωτήσεων ενός τεστ, το οποίο δόθηκε στους φοιτητές με σκοπό να αξιολογήσει κατά πόσο ανταποκρίνεται στην πραγματικότητα, η αξιολόγηση που έγινε από τους φοιτητές στις 4 πρώτες ερωτήσεις του **Πίνακα 3—4** (και οι οποίες αφορούν τις σημαντικότερα σημεία διδασκίας της προτεινόμενης πειραματικής διαδικασίας). Αξίζει να σημειωθεί ότι, η μέθοδος αποτίμησης των διδασχθέντων εννοιών βασίστηκε στην έρευνα των Πετρίδου κ.ά. [19].

Οι 4 πρώτες ερωτήσεις του τεστ απαιτούν αιτιολόγηση, ενώ η τελευταία ερώτηση διερευνά την κριτική ικανότητα σκέψης των φοιτητών όσον αφορά τις μετρήσεις που έλαβαν στο εργαστήριο. Οι απαντήσεις των φοιτητών βαθμολογήθηκαν ως εξής:

Βαθμός 1: ελλιπής ή καθόλου απάντηση.

Βαθμός 2: μη ικανοποιητική απάντηση.

Βαθμός 3: μέση απόδοση (σωστή απάντηση αλλά ελλιπής ή καθόλου αιτιολόγηση).

Βαθμός 4: ικανοποιητική απάντηση.

Βαθμός 5: άριστη απάντηση.

Η βαθμολογία των δύο πρώτων ερωτήσεων (**Πίνακας 3—5**) είναι λογική αν ληφθεί υπόψη η βαθμολογία των δύο πρώτων ερωτήσεων του **Πίνακα 3—4**. Ωστόσο, ο χαμηλότερος βαθμός της ερώτησης 2 (**Πίνακας 3—5**) έγκειται στο γεγονός ότι κατά τη διάρκεια αξιολόγησης δεν πραγματοποιήθηκαν μετρήσεις απόλυτου ύψους σε εξωτερικό περιβάλλον. Η – κάτω από το μέσο όρο – βαθμολογία της ερώτησης 3 (**Πίνακας 3—5**) είναι σε συμφωνία με την απόκριση των φοιτητών στην ερώτηση 4 του **Πίνακα 3—4**. Η – κάτω από το (αν και πολύ κοντά στο) μέσο όρο – βαθμολογία της ερώτησης 4 (**Πίνακας 3—5**) είναι αποτέλεσμα της στάσης των φοιτητών κατά τη διάρκεια διεξαγωγής των πειραμάτων. Οι φοιτητές έδειξαν ιδιαίτερη δυσκολία στην κατανόηση της απόκλισης στο σήμα εξόδου δύο πανομοιότυπων αισθητήρων που μετρούν την ίδια τιμή ατμοσφαιρικής πίεσης. Τέλος, ικανοποιητική χαρακτηρίζεται η κριτική ικανότητα σκέψης των φοιτητών δεδομένης της βαθμολογίας της ερώτησης 5 (**Πίνακας 3—5**).

Πίνακας 3—5 Αποτίμηση κατανόησης των διδαχθέντων εννοιών

α/α	Ερώτηση	Βαθμός (5)	Βαθμός (4)	Βαθμός (3)	Βαθμός (2)	Βαθμός (1)	Μέσος όρος
1	Θέλετε να μετρήσετε το απόλυτο ύψος χρησιμοποιώντας σύστημα βαρομετρικής υψομετρίας, ενώ η ακρίβεια της μέτρησης κρίνεται εξαιρετικά σημαντική. Ποια τεχνική επιλέγετε, α) την τεχνική απλής βαρομετρικής υψομετρίας ή β) την τεχνική της διαφορικής βαρομετρικής υψομετρίας, και γιατί;	6	0	7	0	0	3,92
2	Ποια τεχνική επιλέγεται α) την τεχνική απλής βαρομετρικής υψομετρίας ή β) την τεχνική της διαφορικής βαρομετρικής υψομετρίας για τη μέτρηση απόλυτου ύψους σε εξωτερικό περιβάλλον, και γιατί;	0	2	11	0	0	3,15
3	Σε μία μέτρηση απόλυτου ύψους με σύστημα βαρομετρικής υψομετρίας ο χρόνος λήψης των μετρήσεων πρέπει να είναι όσο το δυνατόν μικρότερος, λόγω των αυξημένων μεταβολών ατμοσφαιρικής πίεσης που επηρεάζουν σημαντικά την ακρίβεια του απόλυτου ύψους. Ποια εκ των δύο σχέσεων προτείνεται για τη διεξαγωγή των μετρήσεων: α) την υψομετρική εξίσωση (απαιτεί μέτρηση θερμοκρασίας) ή β) τη διεθνή βαρομετρική εξίσωση (δεν απαιτεί μέτρηση θερμοκρασίας). Αιτιολογήστε την	3	1	1	0	8	2,31

α/α	Ερώτηση	Βαθμός (5)	Βαθμός (4)	Βαθμός (3)	Βαθμός (2)	Βαθμός (1)	Μέσος όρος
4	απάντησή σας. Θέλουμε να μετρήσουμε, διαδοχικά, 10 διαφορετικές τιμές απόλυτου ύψους με τη χρήση συστήματος διαφορικής βαρομετρικής υψομετρίας. Χρειάζεται βαθμονόμηση των αισθητήρων ατμοσφαιρικής πίεσης (σε ύψος $\gamma=0\mu$) μεταξύ των διαδοχικών μετρήσεων, ή μόνο μία φορά πριν την αρχική μέτρηση, και γιατί;	2	0	8	0	3	2,85
5	Που θεωρείτε ότι οφείλονται τυχόν διαφορές στην ακρίβεια απόλυτου ύψους ανάμεσα στις διαφορετικές μετρήσεις που καταγράφηκαν από τους διδασκόμενους; Κυκλώστε το σωστό. A) Σε λάθος χειρισμό του συστήματος από τον διδασκόμενο. B) Σε λάθος λειτουργία του συστήματος βαρομετρικής υψομετρίας. Γ) Σε μεταβολές της ατμοσφαιρικής πίεσης κατά τη διεξαγωγή του πειράματος. Δ) Στο μικρό αριθμό δειγμάτων που λήφθηκαν κατά τη διεξαγωγή του πειράματος. Ε) Σε μεταβολές της ατμοσφαιρικής πίεσης σε συνδυασμό με τον μικρό αριθμό που λήφθηκαν κατά τη διεξαγωγή του πειράματος.	3	4	3	1	2	3,38

Στον Πίνακα 3—6 δίνονται οι βαθμολογίες που έλαβαν οι φοιτητές στο τεστ αξιολόγησης. Τρεις από τους φοιτητές απέτυχαν στο τεστ, ενώ ο μέσος όρος γραπτών είναι 17,77/30.

Πίνακας 3—6 Βαθμολογία φοιτητών στο τεστ αξιολόγησης

Βαθμολογία φοιτητών (υψηλότερος βαθμός: 30)	10/30	14/30	17/30	20/30	21/30	22/30	28/30
Αριθμός φοιτητών που πήραν το βαθμό αυτό	2	1	2	1	4	1	1
Αριθμός φοιτητών που έλαβαν βαθμό κάτω από το μέσο όρο				3			
Αριθμός φοιτητών που έλαβαν βαθμό πάνω από το μέσο όρο				10			
Μέσος όρος γραπτών				17,77			

Προτάσεις αναβάθμισης: μέτρηση όγκου αέρα σε αεροστεγές δοχείο

$$\rho = \frac{1}{V} = \frac{\left(\frac{P}{Ra} \cdot T\right)(1+x)}{1+x \frac{Ra}{Rw}}$$

όπου : V (όγκος υγρού αέρα)

P (πίεση υγρού αέρα σε Pa)

T (θερμοκρασία σε Kelvin)

Ra (σταθερά αερίων για ξηρό αέρα σε $J \times Kg^{-1} \times grad^{-1}$) = 287

Rw (σταθερά αερίων για υδρατμούς σε $J \times Kg^{-1} \times grad^{-1}$) = 462

$x = \frac{0,62198 \cdot Pa}{Pa - Pw}$ (αναλογία υγρασίας : Kg / Kg)

Pa (ατμοσφαιρική πίεση υγρού αέρα σε Pa, psi)

Pw (μερική πίεση των υδρατμών σε υγρό αέρα σε Pa, psi)

και : $\varphi = Pw_{(mbar)} / Pws_{(mbar)} 100\%$ (σχετική υγρασία)

$Pws = e^{(77,3450 + 0,0057 T - 7235 / T)} / T^{8,2}$

(μερική πίεση κορεσμένου ατμού στην πραγματική θερμοκρασία ξηρού βόλβου σε mbar,

και T η θερμοκρασία ξηρού βόλβου του υγρού αέρα σε Kelvin)

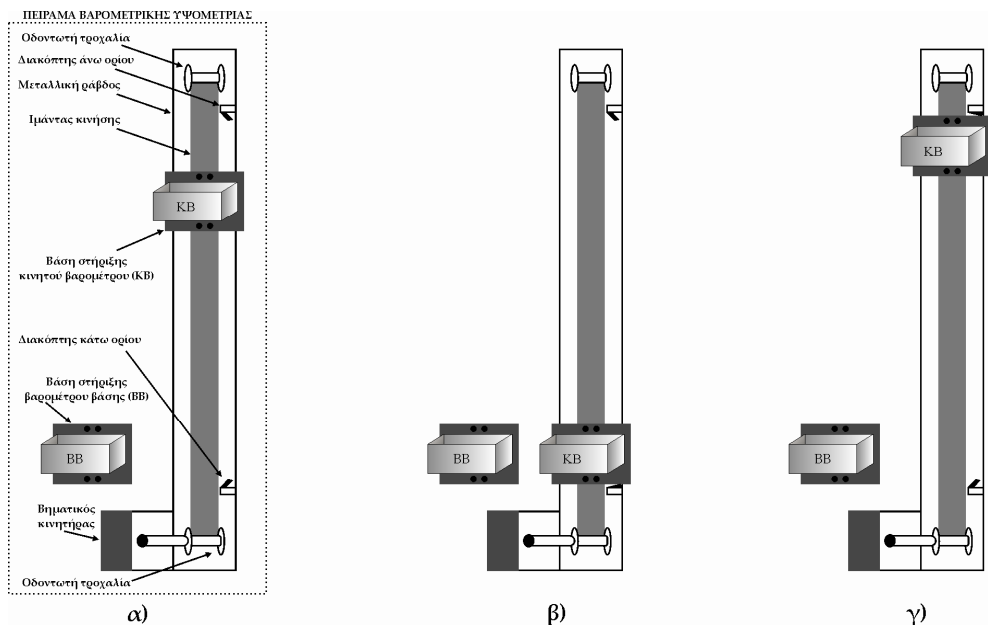
Τύπος 3—5 Υπολογισμός όγκου αέρα σε αεροστεγές δοχείο

Με μικρές αναβαθμίσεις υλισμικού (hardware) και υλικολογισμικού (firmware) στις συσκευές μέτρησης του συστήματος για την επιπλέον καταγραφή υγρασίας, είναι εφικτός ο υπολογισμός όγκου του αέρα σε αεροστεγές δοχείο. Η υλοποίηση αυτή καθιστά εφικτή τη διεξαγωγή ενός ακόμη πειράματος με MEMS αισθητήρες βαρομετρικής πίεσης, ενώ ο υπολογισμός του όγκου γίνεται μέσω του Τύπου 3—5.

Προτάσεις αναβάθμισης: πειραματισμός ατόμων με κινητική αναπηρία

Σύμφωνα με το Colorado State University², ο όρος *κινητική αναπηρία* αναφέρεται στην ανικανότητα ενός ατόμου να χρησιμοποιεί ένα ή περισσότερα από τα άκρα του/της, ή την έλλειψη ικανότητας να περπατήσει, να κατανοήσει, ή να ανυψώσει αντικείμενα. Είναι γενικά αποδεκτή η άποψη ότι, «η κοινωνική αντίληψη που τείνει να ταυτίσει την τυχόν αναπηρία ενός ατόμου με ολική ανικανότητα, στερεί στα άτομα με αναπηρία (AMEA) την ευκαιρία για σωστή εκπαίδευση και επιλογή καριέρας» [20]. Στις μέρες μας διατίθενται διάφορα λογισμικά όπως και διάφορες εφαρμογές λειτουργικών συστημάτων για τον έλεγχο του ηλεκτρονικού υπολογιστή (H/Y) μέσω φωνητικών εντολών που δίδονται από το χρήστη, ενώ με τη χρήση και ενσωματωμένων υπολογιστικών συστημάτων είναι δυνατός ο έλεγχος της αυτόνομης λειτουργίας και διεξαγωγής πειράματος.

Σύμφωνα με τα παραπάνω, στα πλαίσια της διδακτορικής διατριβής προσδιορίστηκαν οι προδιαγραφές αναβάθμισης του προτεινόμενου εξοπλισμού για την εργαστηριακή εκπαίδευση ατόμων με κινητική αναπηρία [21]. Στη συνέχεια δίνονται οι λεπτομέρειες σχεδίασης για την αυτόνομη λειτουργία του υπάρχοντος εξοπλισμού πειραματισμού στη βαρομετρική υψομετρία. Υλικά για την υλοποίηση του εξοπλισμού αυτού είναι διαθέσιμα στην ιστοσελίδα <http://openbuildspartstore.com/>, ενώ ιδέες υλοποίησης εντοπίζονται στην ιστοσελίδα <http://openbuilds.org/>.



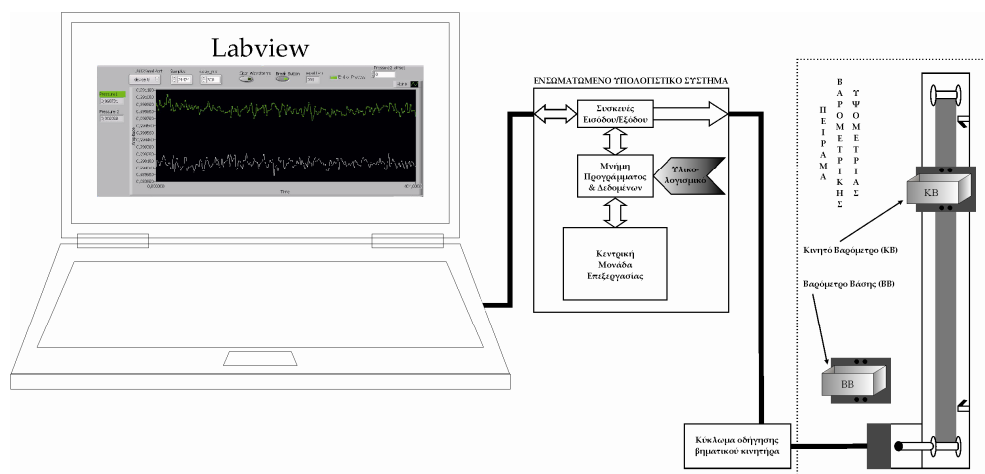
Εικόνα 3—14 Αυτόνομο σύστημα υπολογισμού βαρομετρικού υψομέτρου

² Module: Mobility Impairments, http://accessproject.colostate.edu/disability/modules/MI/tut_ML.php?display=pg_1

Στην **Εικόνα 3—14** παρουσιάζεται ο προτεινόμενος εξοπλισμός εγκατάστασης του πειράματος για την αυτόνομη λειτουργία του τελευταίου. Τα εξαρτήματα που απαιτούνται είναι τα εξής:

1. Μεταλλική ράβδος στη βάση και την κορυφή της οποίας τοποθετούνται δύο οδοντωτές τροχαλίες κινήσεως της συσκευής μέτρησης (κινητό βαρόμετρο).
2. Ιμάντας κινήσεως που τοποθετείται στις οδοντωτές τροχαλίες και στον οποίο εγκαθίσταται κατάλληλη βάση στήριξης.
3. Πρόσδεση του κινητού βαρομέτρου του συστήματος στη βάση στήριξης για τη μετακίνηση του τελευταίου κατά μήκος της μεταλλικής ράβδους.
4. Βηματικός κινητήρας για τον έλεγχο κίνησης του ιμάντα και διακοπή λειτουργίας του κινητήρα από δύο διακόπτες που τοποθετούνται στα άκρα της ράβδους (στο άνω και κάτω όριο).

Κατά την κίνηση το κινητού βαρομέτρου προς τα πάνω, η βάση στήριξης του KB δεσμεύει το διακόπτη στο άνω άκρο της και τερματίζεται η λειτουργία του βηματικού κινητήρα (**Εικόνα 3—14β**). Ο βηματικός κινητήρας παραμένει στο άνω όριο για όσο χρόνο απαιτείται για τη λήψη μετρήσεων από το συντονιστή του συστήματος και εν συνέχεια, ο κινητήρας λειτουργεί με αντίθετη φορά (μετακίνηση του KB προς το κάτω όριο). Το KB μετακινείται προς το κάτω όριο έως ότου δεσμευτεί ο αντίστοιχος διακόπτης της ράβδου (**Εικόνα 3—14γ**). Η δέσμευση του διακόπτη τερματίζει η λειτουργία του βηματικού κινητήρα στο κάτω όριο για όσο χρόνο απαιτείται για τη λήψη μετρήσεων από το συντονιστή του συστήματος (απαραίτητη διαδικασία για τη βαθμονόμηση των συσκευών μέτρησης). Το βαρόμετρο βάσης του συστήματος θα πρέπει να τοποθετηθεί σε κατάλληλο σημείο του χώρου, ώστε όταν το κινητό βαρόμετρο βρίσκεται στο κάτω όριο της ράβδου να ευθυγραμμίζεται οριζοντίως με το BB για την ορθή βαθμονόμηση των δυο αισθητήρων του συστήματος.



Εικόνα 3—15 Πειραματική διάταξη βαρομετρικής υψομετρίας για ΑΜΕΑ

Στην **Εικόνα 3–15** παρουσιάζεται ο προτεινόμενος εξοπλισμός για την εργαστηριακή εκπαίδευση ατόμων με κινητική αναπηρία. Απαιτείται, επιπλέον, η υλοποίηση ηλεκτρονικού κυκλώματος για τον έλεγχο του βηματικού κινητήρα. Το πείραμα μπορεί να ελέγχεται από κατάλληλα σχεδιασμένο λογισμικό (π.χ. Labview). Ο έλεγχος του ηλεκτρονικού υπολογιστή που θα εκκινεί το λογισμικό αυτό μπορεί να πραγματοποιηθεί μέσω φωνητικών εντολών από ήδη υπάρχοντα εμπορικά προγράμματα (ή και ελεύθερα λογισμικά). Η ανάλυση των μετρήσεων από τους διδασκόμενους πραγματοποιείται μέσω ειδικών προγραμμάτων που αναπτύχθηκαν σε Matlab, τα οποία πραγματοποιούν αυτόματη ανάλυση των μετρήσεων βαρομετρικής πίεσης.

Αναφορές

- [1] B. Fallery, S. Gerbaix, and R. Ologeanu, "Videotraining: a comparison between virtual class and remote class", *International Symposium on Collaborative Technologies and Systems*, Irvine CA, 2008, pp. 440-445.
- [2] B. Hanson, P. Culmer, J. Gallagher, K. Page, E. Read, A. Weightman, and M. Levesley, "Remote laboratories in the curriculum", *11th IASTED International on Conference Computers and Advanced Technology in Education*, Crete, 2008, pp. 160-165, 2008.
- [3] V. Kolias, I. Anagnostopoulos, and E. Kayafas, "Remote experiments in education: A survey over different platforms and application fields", *11th International Conference on Optimization of Electrical and Electronic Equipment*, Brasov, 2008, pp. 181-188.
- [4] C. Gravier, J. Fayolle, B. Bayard, M. Ates, and J. Lardon, "Remote laboratories: proposed guidelines. 2nd International Conference on Digital Information Management", Lyon France, 2007, pp. 786-792.
- [5] M.M.D. Amadou, M. Saad, J.P. Kenne, and V. Nerguizian, "Virtual and remote laboratories", *1st IEEE International Conference on E-Learning in Industrial Electronics*, Hammamet, 2006, pp. 173-176.
- [6] A. Baccigalupi, C. De Capua, and A. Liccardo, "Overview on development of remote teaching laboratories: from labview to web services", *Instrumentation and Measurement Technology Conference*, Sorrento, 2006, pp. 992-997.
- [7] D.Z. Deniz, A. Bulancak, and G. Ozcan, "A novel approach to remote laboratories", *33rd ASEE/IEEE Frontiers in Education Annual Conference*, Boulder CO, 2003, pp. T3E/8-T3E/12.
- [8] M. Cooper, and J.M.M. Ferreira, "Remote laboratories extending access to science and engineering curricular", *IEEE Transactions on Learning Technologies*, 2(4), pp. 342-353, 2009.
- [9] Δ. Ε. Μπολανάκης και Κ. Κώτσης, "Χρήση πειραμάτων απομακρυσμένης πρόσβασης με ασύρματη επικοινωνία στη διδασκαλία Φυσικής: ανασκόπηση βιβλιογραφίας και προτάσεις", *6ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση*, Φλώρινα, 2009, σελ. 579-586.
- [10] Δ. Ε. Μπολανάκης και Κ. Κώτσης, "Απομακρυσμένος έλεγχος μετρήσεων Φυσικών μεγεθών πραγματικού χρόνου: δυνατότητες και περιορισμοί", *9ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση*, Θεσσαλονίκη, 2015, σελ. 232-236.
- [11] D. E. Bolanakis, K. T. Kotsis, and T. Laopoulos, "A Prototype Wireless Sensor Network System for a Comparative Evaluation of Differential and Absolute Barometric Altimetry", *IEEE Aerospace and Electronic Systems Magazine*, 30(11), pp. 20-28, 2015.
- [12] D. E. Bolanakis, K. T. Kotsis, and T. Laopoulos, "Temperature Influence on Differential Barometric Altitude Measurements", *The 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS'2015)*, Warsaw, Poland, 2015, pp. 120-124.
- [13] D. E. Bolanakis, K. T. Kotsis, T. Laopoulos, "Work in Progress: Experiments on MEMS Barometric Pressure Sensors", *3ο Πανελλήνιο Συνέδριο Ηλεκτρονικής και Τηλεπικοινωνιών (PACET 2015)*, Ιωάννινα, 2015, σελ. 1-3.

- [14] D. E. Bolanakis, K. T. Kotsis, and T. Laopoulos, "Ethernet & PC-based Experiments on Barometric Altimetry using MEMS in a Wireless Sensor Network", *Computer Application in Engineering Education*, 24(3), pp. 428-442, 2016.
- [15] Δ. Ε. Μπολανάκης και Κ. Κώτσης, "Πείραμα για τη μέτρηση υψομετρικής διαφοράς: μία πρόταση για άτομα με κινητική αναπηρία", Πανελλήνιο Συνέδριο στις Διδακτικές προσεγγίσεις και πειραματική διδασκαλία στις Φυσικές Επιστήμες, Θεσσαλονίκη, 2016, σελ. 205-213.
- [16] W. Tang and Y-H Tsai, "Barometric altimeter short-term accuracy analysis", *IEEE Aerosp. Electron. Syst. Mag.*, vol. 20, 2005, pp. 24-26.
- [17] R. Sallier, R. Missler, and A. Schutze, "VenDASys — A versatile experimentation platform for educational purposes", *2010 IEEE Education Engineering (EDUCON)*, 2010, pp. 205-210.
- [18] D.E. Bolanakis, E. Glavas, and G.A. Evangelakis, "An integrated microcontroller-based tutoring system for computer architecture laboratory course," *Int. J. Eng. Educ.*, vol. 26, no. 4, pp. 785-798, 2007.
- [19] Ε. Πετρίδου, Σ. Φωτιάδου, και Κ. Κουκουλάς, "Πραγματικό ή Εικονικό Εργαστήριο; Έρευνα στη Β' Γυμνασίου", 9ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση, Θεσσαλονίκη, 2015, σελ. 242-246.
- [20] Σ. Κρητικού και Ι. Παλυβός, "Προσομοίωση πειραματικής διδασκαλίας για μαθητές με κινητικά προβλήματα", 2ο Πανελλήνιο Συνέδριο των Εκπαιδευτικών για τις ΤΠΕ, Σύρος, 2003, σελ. 46-57.
- [21] Δ. Ε. Μπολανάκης και Κ. Κώτσης, "Πείραμα για τη μέτρηση υψομετρικής διαφοράς: μία πρόταση για άτομα με κινητική αναπηρία", Πανελλήνιο Συνέδριο στις Διδακτικές προσεγγίσεις και πειραματική διδασκαλία στις Φυσικές Επιστήμες, Θεσσαλονίκη, 2016, σελ. 205-213.

Συμπεράσματα και μελλοντικές προοπτικές

Στα πλαίσια της διδακτορικής διατριβής επιχειρήθηκε η ανάπτυξη και υλοποίηση καινοτόμου εκπαιδευτικού εξοπλισμού, ο οποίος βρίσκει εφαρμογή σε ένα ερευνητικό αντικείμενο που είναι ακόμη υπό διερεύνηση από την επιστημονική ακαδημαϊκή κοινότητα. Το αντικείμενο αυτό αφορά τα πειράματα βαρομετρικής υψομετρίας με τη χρήση MEMS (Μικρο-Ηλεκτρομηχανικά Συστήματα) αισθητήρων.

Για την επίτευξη του σκοπού αυτού κρίθηκε απαραίτητη η διεξαγωγή επιστημονικής έρευνας, ώστε να απαντηθούν ζητήματα απαραίτητα για την εκπαίδευση των διδασκομένων, τα οποία όμως δεν έχουν διερευνηθεί πλήρως από την ακαδημαϊκή κοινότητα.

Κατόπιν, πραγματοποιήθηκε η απαιτούμενη εκπαιδευτική έρευνα για την επίτευξη της ολοκληρωμένη πειραματικής εκπαίδευσης των διδασκομένων. Στη συνέχεια του κεφαλαίου συνοψίζεται η επιστημονική και εκπαιδευτική έρευνα που διεξήχθη στα πλαίσια της διδακτορικής διατριβής και

4

ΣΤΟ ΚΕΦΑΛΑΙΟ ΑΥΤΟ

ΕΠΙΣΤΗΜΟΝΙΚΗ ΕΡΕΥΝΑ

Συνοψίζεται η επιστημονική έρευνα που διεξήχθη στα πλαίσια τη διδακτορικής διατριβής.

ΕΚΠΑΙΔΕΥΤΙΚΗ ΕΡΕΥΝΑ

Συνοψίζεται η εκπαιδευτική έρευνα που διεξήχθη στα πλαίσια τη διδακτορικής διατριβής.

ΠΡΟΟΠΤΙΚΕΣ ΣΥΝΕΧΕΙΑΣ ΤΗΣ ΕΡΕΥΝΑΣ

Παρουσιάζονται οι προοπτικές συνέχειας της ερευνητικής δραστηριότητας.

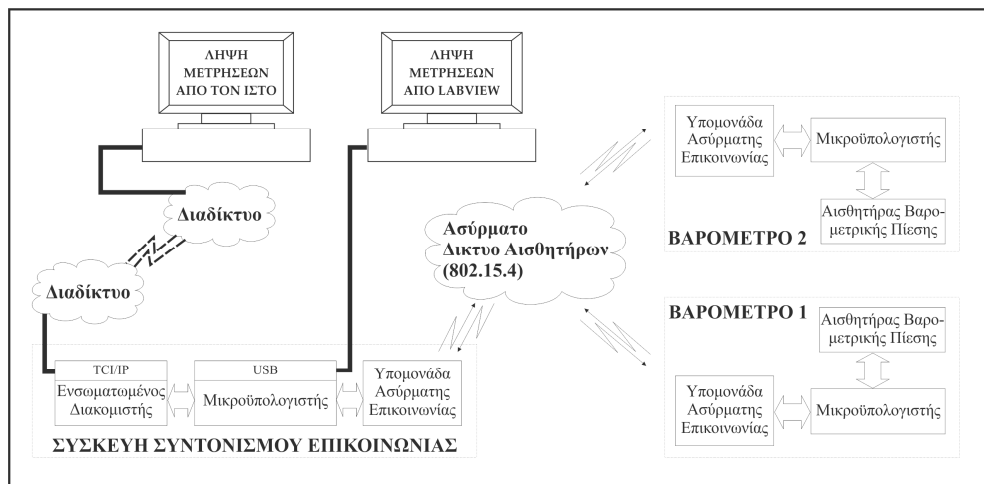
ακόμη, παρουσιάζονται οι προοπτικές συνέχειας της ερευνητικής αυτής δραστηριότητας.

Επιστημονική έρευνα

Η έρευνα στη βαρομετρική υψομετρία με τη χρήση MEMS αισθητήρων ατμοσφαιρικής πίεσης προσανατολίζεται σε συστήματα πλοήγησης εσωτερικών χώρων [1-2] και σε εφαρμογές αεροσκαφών [4,5]. Για παράδειγμα, στην πρώτη περίπτωση μπορεί να εντοπιστεί ο όροφος ενός μεγάλου κτηρίου (εμπορικού κέντρου, μουσείου, κλπ.), ενώ στη δεύτερη περίπτωση είναι δυνατή η εκτίμηση των γωνιών κλίσης αεροσκάφους (π.χ., των γωνιών κλίσης των πτερυγίων).

Και στις δύο περιπτώσεις υπολογίζεται το απόλυτο ύψος μεταξύ δύο σημείων στο χώρο με τη χρήση δύο διαφορετικών μεθόδων μέτρησης: α) απλή βαρομετρική υψομετρία (ΑΒΥ) και β) διαφορική βαρομετρική υψομετρίας (ΔΒΥ). Η ΑΒΥ μετράει το απόλυτο ύψος δύο σημείων με τη χρήση ενός, μόνο, αισθητήρα (άρα σε διαφορετική χρονική στιγμή), ενώ η ΔΒΥ μετράει ταυτόχρονα το απόλυτο ύψος δύο σημείων με δύο ανεξάρτητους αισθητήρες. Τα ερευνητικά αποτελέσματα που συναντώνται στην τρέχουσα διεθνή βιβλιογραφία [6-9], όσον αφορά τις δύο μεθόδους βαρομετρικής υψομετρίας, είναι τα ακόλουθα:

- Οι αισθητήρες βαρομετρικής πίεσης είναι ευαίσθητοι στις καιρικές αλλαγές και συνεπώς, παράγουν μεγάλο σφάλμα μέτρησης σε εξωτερικό περιβάλλον.
- Η ΔΒΥ παράγει μετρήσεις μεγαλύτερης ακρίβειας σε σύγκριση με την ΑΒΥ.
- Στην ΔΒΥ οι δύο αισθητήρες μέτρησης παρουσιάζουν απόκλιση στο σήμα εξόδου όταν μετρούν την ίδια τιμή πίεσης, και η απόκλιση αυτή οδηγεί σε μακροπρόθεσμη αστάθεια μέτρησης.



Εικόνα 4—1 Αρχιτεκτονική πρωτότυπου συστήματος βαρομετρικής υψομετρίας

Στα πλαίσια της διδακτορικής διατριβής αναπτύχθηκε και υλοποιήθηκε πρωτότυπος ερευνητικός και εκπαιδευτικός εξοπλισμός, οποίος λαμβάνει μετρήσεις από

δύο ανεξάρτητες συσκευές μέτρησης (**Εικόνα 4—1**) και συνεπώς, επιτρέπει την αποτίμηση των δύο μεθόδων βαρομετρικής υψομετρίας. Τα ερευνητικά αποτελέσματα που προέκυψαν από την επιστημονική αυτή έρευνα [10] συνοψίζονται στα ακόλουθα:

- Οι αλλαγές στην πίεση του ατμοσφαιρικού αέρα εφαρμόζονται ταυτόχρονα στους δύο αισθητήρες συστήματος ΔΒΥ και για το λόγο αυτό, η ΔΒΥ χαρακτηρίζεται από μετρήσεις μεγαλύτερης ακρίβειας συγκριτικά με την ΑΒΥ.
- Οι έντονες καιρικές αλλαγές σε εξωτερικό περιβάλλον καθιστούν την ΑΒΥ ακατάλληλη για μετρήσεις εξωτερικών χώρων. Ωστόσο, στη ΔΒΥ οι αλλαγές αυτές εφαρμόζονται ταυτόχρονα και στους δύο αισθητήρες και έτσι, η ΔΒΥ μπορεί να χρησιμοποιηθεί και σε εξωτερικό περιβάλλον (όπου, βέβαια, στην περίπτωση αυτή οι μετρήσεις είναι χαμηλότερης ακρίβειας).
- Η απόκλιση που παρουσιάζεται στο σήμα εξόδου των δύο αισθητήρων ΔΒΥ που μετρούν την ίδια τιμή πίεσης οφείλεται σε κατασκευαστικές ανομοιοότητες των αισθητήρων και συνεπώς, η καμπύλη απόκρισης των τελευταίων δεν μπορεί να ταυτίζεται. Για το λόγο αυτό, απαιτείται βαθμονόμηση των δύο αισθητήρων πριν από κάθε μέτρηση απόλυτης υψομετρικής διαφοράς, με την τοποθέτηση των δύο αισθητήρων στο ίδιο ύψος και την εκτίμηση της απόκλισης στην έξοδο τους τη δεδομένη χρονική στιγμή.

Το δεύτερο μέρος της επιστημονικής έρευνας εστιάστηκε στην αποτίμηση της ακρίβειας μέτρησης απόλυτους με τη χρήση δύο διαφορετικών εξισώσεων:

α) της υψομετρικής εξίσωσης (ΥΕ), η οποία απαιτεί την επιπρόσθετη καταγραφή θερμοκρασίας για την αναγωγή της ατμοσφαιρικής πίεσης σε υψόμετρο,

β) της διεθνούς βαρομετρικής εξίσωσης (ΔΒΕ), η οποία είναι σε συμφωνία με το μοντέλο ατμόσφαιρας ISA (International Standard Atmosphere το οποίο θεωρεί τη θερμοκρασία ατμόσφαιρας σταθερή στους 15°C. Συνεπώς, η αναγωγή της ατμοσφαιρικής πίεσης σε υψόμετρο με τη χρήση της ΔΒΕ είναι απαλλαγμένη από επιπρόσθετες μετρήσεις θερμοκρασίας. Τα ερευνητικά αποτελέσματα που προέκυψαν από την στατιστική ανάλυση μετρήσεων απόλυτου ύψους με τη χρήση των δύο προαναφερθέντων εξισώσεων [11] συνοψίζονται στα ακόλουθα:

- Οι δύο εξισώσεις αποδίδουν μετρήσεις ισοτιμής ακρίβειας και συνεπώς, είναι προτιμότερη η χρήση της ΔΒΕ για την αναγωγή ατμοσφαιρικής πίεσης σε απόλυτο ύψος καθότι δεν απαιτεί την επιπρόσθετη καταγραφή θερμοκρασίας. Αφενός, απλουστεύεται η σχεδίαση του συστήματος βαρομετρικής υψομετρίας και αφετέρου, μειώνεται το διάστημα μεταξύ διαδοχικών μετρήσεων (sampling interval), το οποίο κρίνεται ιδιαίτερα σημαντικό στην ακρίβεια μέτρησης (λόγω των μεταβολών πίεσης του ατμοσφαιρικού αέρα ακόμη και σε περιβάλλον εσωτερικού χώρου).

Εκπαιδευτική έρευνα

Στα πλαίσια τη διδακτορικής διατριβής πραγματοποιήθηκε αρχικά βιβλιογραφική έρευνα όσον αφορά την υλοποίηση πειραματικών εξοπλισμών απομακρυσμένης πρόσβασης με ασύρματη επικοινωνία [12], ώστε να γίνει δόκιμη επιλογή των ψηφιακών τεχνολογιών του πρωτότυπου εργαστηριακού εξοπλισμού και να προσδιοριστούν οι προδιαγραφές του συστήματος [13]. Αφού υλοποιήθηκε ο πειραματικός εξοπλισμός και απαντήθηκαν τα ερωτήματα που συσχετίζονται με τις μεθόδους βαρομετρικής υψομετρίας (ABY και ΔBY), καθώς και με τις δύο εξισώσεις αναγωγής ατμοσφαιρικής πίεσης σε απόλυτο υψόμετρο (ΥΕ και ΔΒΕ), προσδιορίστηκαν και εν συνεχεία αξιολογήθηκαν τα πειράματα εκπαίδευσης [14]. Τα προτεινόμενα πειράματα συνοψίζονται στον **Πίνακα 4—1**.

Πίνακας 4—1 Πειράματα βαρομετρικής υψομετρίας

α/α	Πείραμα
1	Υπολογισμός ακρίβειας απόλυτου ύψους με την τεχνική ΔBY και ΑΠΥ σε περιβάλλον εσωτερικού χώρου
2	Εκτίμηση αξιοπιστίας μετρήσεων απόλυτου ύψους με την τεχνική ΔBY και ΑΠΥ σε περιβάλλον εξωτερικού χώρου και υπολογισμός ακρίβειας σε περίπτωση αξιοπιστής μέτρησης
3	Εκτίμηση σφάλματος μέτρησης απόλυτου ύψους με τη χρήση της υψομετρικής εξίσωσης και της διεθνούς βαρομετρικής εξίσωσης
4	Διερεύνηση της δυνατότητας μιας και μοναδικής βαθμονόμησης των συσκευών μέτρησης στην περίπτωση της ΔBY
5	Αξιολόγηση ακρίβειας μετρήσεων που καταγράφηκαν α) μέσω Η/Υ και β) μέσω δικτύου/διαδικτύου

Η αξιολόγηση του συστήματος καθώς και των προτεινόμενων πειραμάτων πραγματοποιήθηκε από 13 μεταπτυχιακούς φοιτητές, όπου 4 από τους φοιτητές διέθεταν πτυχίο Φυσικής και παρακολουθούσαν το μεταπτυχιακό πρόγραμμα σπουδών (ΜΠΣ) «Έρευνα και Νέες Τεχνολογίες στη Διδασκαλία της Φυσικής», ενώ οι υπόλοιποι 9 ήταν κάτοχοι διπλώματος του Παιδαγωγικού Τμήματος Δημοτικής Εκπαίδευσης και παρακολουθούσαν το μεταπτυχιακό πρόγραμμα σπουδών (ΜΠΣ) «Φυσικές Επιστήμες στην Εκπαίδευση». Στους φοιτητές δόθηκε ερωτηματολόγιο αξιολόγησης του εξοπλισμού και των πειραμάτων και κατόπιν, δόθηκε τεστ αξιολόγησης των απαντήσεων των φοιτητών όπου προέκυψαν ικανοποιητικά αποτελέσματα [14]. Ακόμη, στα πλαίσια της διδακτορικής διατριβής προσδιορίστηκαν οι προδιαγραφές αναβάθμισης του προτεινόμενου εξοπλισμού για την εργαστηριακή εκπαίδευση ατόμων με κινητική αναπηρία [15].

Προοπτικές συνέχειας της έρευνας

Από τη συνολική έρευνα (επιστημονική και εκπαιδευτική) που διεξήχθη στα πλαίσια της διδακτορικής διατριβής επιβεβαιώθηκε ότι, είναι εφικτή η υλοποίηση εκπαιδευτικού εξοπλισμού σε ένα αντικείμενο μελέτης που δεν έχει διερευνηθεί πλήρως από την επιστημονική κοινότητα, με την προϋπόθεση ότι πραγματοποιείται διεπιστημονική συνεργασία και διεξάγεται παράλληλα τόσο επιστημονική όσο και εκπαιδευτική έρευνα. Η επιστημονική έρευνα καλείται να δώσει απαντήσεις σε ερωτήματα απαραίτητα για την εκπαίδευση των διδασκομένων, τα οποία δεν έχουν καλυφθεί πλήρως από τα τρέχοντα ερευνητικά αποτελέσματα. Η εκπαιδευτική έρευνα θέτει τις προδιαγραφές της ψηφιακής τεχνολογίας που θα χρησιμοποιηθεί στον εργαστηριακό εξοπλισμό, καθώς και των πειραμάτων που θα χρησιμοποιηθούν από τους φοιτητές και που θα εξασφαλίσουν την ομαλή και ολοκληρωμένη εργαστηριακή τους εκπαίδευση. Στη συνέχεια δίδονται προτάσεις μελλοντικής έρευνας τόσο σε εκπαιδευτικό όσο και σε επιστημονικό επίπεδο.

Με μικρές αναβαθμίσεις υλισμικού (hardware) και υλικολογισμικού (firmware) στις συσκευές μέτρησης του συστήματος για την επιπλέον καταγραφή υγρασίας, είναι εφικτός ο υπολογισμός όγκου του αέρα σε αεροστεγές δοχείο. Η υλοποίηση αυτή καθιστά εφικτή τη διεξαγωγή ενός ακόμη πειράματος με MEMS αισθητήρες βαρομετρικής πίεσης.

Η τρέχουσα υλοποίηση του εργαστηριακού εξοπλισμού επιτρέπει τη λήψη μετρήσεων από 4 διαφορετικές συσκευές μέτρησης. Συνεπώς, με την υλοποίηση δύο επιπλέον συσκευών μέτρησης, είναι δυνατή η αξιολόγηση ακρίβειας των αισθητήρων πίεσης στη ΔΒΥ, χρησιμοποιώντας δύο ζεύγη αισθητήρων από δύο διαφορετικούς κατασκευαστές.

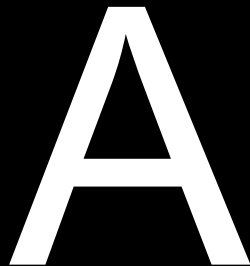
Τέλος, έχουν δοθεί προτάσεις αναβάθμισης του προτεινόμενου εξοπλισμού για την εργαστηριακή εκπαίδευση ατόμων με κινητική αναπηρία [15] και ένα ενδιαφέρον ζήτημα εκπαιδευτικής έρευνας θα ήταν η υλοποίηση αντίστοιχου εξοπλισμού και πειραμάτων.

Αναφορές

- [1] J. Collin, O. Mezentsev, and G. Lachapelle, "Indoor positioning system using accelerometry and high accuracy heading sensors," in *Proc. of GPS/GNSS 2003 Conference*, Portland, OR, 2003, pp. 1-7.
- [2] A. Morrison, V. Renaudin, J.B. Bancroft, and G. Lachapelle, "Design and testing of a multi-sensor pedestrian location and navigation platform," *Sensors*, vol. 12 (3), 2012, pp. 3720-3738.
- [3] G. Retscher, "Augmentation of indoor positioning systems with a barometric pressure sensor for direct altitude determination in a multi-storey building," *Cartography and Geographic Information Science*, vol. 34 (4), 2007, pp. 305-310.
- [4] R.K. Arning, and S. Sassen, "Flight control of micro aerial vehicles," in *Proc. of AIAA Guidance, Navigation, and Control Conference and Exhibit*, Providence, Rhode Island, 2004, pp. 1-12.
- [5] S.J. Jung, and D. Liccardo, "Automation of small UAVs using a low cost MEMS sensor and embedded computing platform," in *Proc. of 25th Digital Avionics Systems Conference*, Portland, OR, 2006, pp. 1-9.
- [6] L. Binghao, B. Harvey, and T. Gallagher, "Using barometers to determine the height for indoor positioning," in *Proc. of International Conference on Indoor Positioning and Indoor Navigation*, Montbeliard-Belfort, 2013, pp. 1-7.
- [7] C. Bollmeyer, M. Pelka, H. Gehringy, and H. Hellbruck, "Evaluation of radio based, optical and barometric localization for indoor altitude estimation in medical applications," in *Proc. of International Conference on Indoor Positioning and Indoor Navigation*, Bexco, Busan, Korea, 2014, pp. 1-10.
- [8] K. Liu, Y. Wang, and J. Wang, "Differential barometric altimetry assists floor identification in WLAN location fingerprinting study," in *Principle and Application Progress in Location-Based Services*, Springer International Publishing, 2014, pp. 21-29.
- [9] P. Paces, and J. Popelka, "Performance evaluation of two altimeters intended for euler angles measurement," in *Proc. of 2013 IEEE/AIAA 32nd Digital Avionics Systems Conference (DASC)*, East Syracuse, NY, 2013, pp. 8C3-1 - 8C3-12.
- [10] D. E. Bolanakis, K. T. Kotsis, and T. Laopoulos, "A Prototype Wireless Sensor Network System for a Comparative Evaluation of Differential and Absolute Barometric Altimetry", *IEEE Aerospace and Electronic Systems Magazine*, 30(11), pp. 20-28, 2015.
- [11] D. E. Bolanakis, K. T. Kotsis, and T. Laopoulos, "Temperature Influence on Differential Barometric Altitude Measurements", The 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS'2015), Warsaw, Poland, 2015, pp. 120-124.
- [12] Δ. Ε. Μπολανάκης και Κ. Κώτσης, "Χρήση Πειραμάτων Απομακρυσμένης Πρόσβασης με Ασύρματη Επικοινωνία στη Διδασκαλία Φυσικής: ανασκόπηση βιβλιογραφίας και προτάσεις", 6ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση, Φλώρινα, 2009, σελ. 579-586.
- [13] Δ. Ε. Μπολανάκης και Κ. Κώτσης, "Απομακρυσμένος Έλεγχος Μετρήσεων Φυσικών Μεγεθών Πραγματικού Χρόνου: δυνατότητες και περιορισμοί", 9ο Πανελλήνιο

- Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση, Θεσσαλονίκη, 2015, σελ. 232-236.
- [14] D. E. Bolanakis, K. T. Kotsis, and T. Laopoulos, "Ethernet & PC-based Experiments on Barometric Altimetry using MEMS in a Wireless Sensor Network", *Computer Application in Engineering Education*, 24(3), pp. 428-442, 2016.
- [15] Δ. Ε. Μπολανάκης και Κ. Κώτσης, "Πείραμα για τη μέτρηση υψομετρικής διαφοράς: μία πρόταση για άτομα με κινητική αναπηρία", Πανελλήνιο Συνέδριο στις Διδακτικές προσεγγίσεις και πειραματική διδασκαλία στις Φυσικές Επιστήμες, Θεσσαλονίκη, 2016, σελ. 205-213.

Οδηγός εγκα- τάστασης και χρήσης του συστήματος



ΣΤΟ ΠΑΡΑΡΤΗΜΑ ΑΥΤΟ

ΤΡΟΠΟΙ ΛΕΙΤΟΥΡΓΙΑΣ ΤΟΥ ΣΥΣΤΗ- ΜΑΤΟΣ

Γίνεται μία εισαγωγή στους δύο τρόπους λειτουργίας του συστήματος: α) λήψη μετρήσεων μέσω ηλεκτρονικού υπολογιστή και β) λήψη μετρήσεων μέσω δικτύου/διαδικτύου.

ΛΕΙΤΟΥΡΓΙΑ ΣΥΛΛΟΓΗΣ ΜΕΤΡΗΣΕΩΝ ΜΕΣΩ ΗΛΕΚΤΡΟΝΙΚΟΥ ΥΠΟΛΟΓΙΣΤΗ
Δίνονται οι οδηγίες εγκατάστασης και λειτουργίας του συστήματος μέσω ηλεκτρονικού υπολογιστή.

ΛΕΙΤΟΥΡΓΙΑ ΣΥΛΛΟΓΗΣ ΜΕΤΡΗΣΕΩΝ ΜΕΣΩ ΔΙΚΤΥΟΥ/ΔΙΑΔΙΚΤΥΟΥ
Δίνονται οι οδηγίες λειτουργίας του συστήματος μέσω δικτύου/διαδικτύου (δεν απαιτείται εγκατάσταση οδηγών στη συγκεκριμένη περίπτωση)

ΑΝΑΒΑΘΜΙΣΗ ΥΛΙΚΟΛΟΓΙΣΜΙΚΟΥ ΣΤΙΣ ΣΥΣΚΕΥΕΣ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ
Εξετάζεται ο τρόπος αναβάθμισης υλικολογισμικού των συσκευών του συστήματος.

Τρόποι λειτουργίας του συστήματος

Το σύστημα που αναπτύχθηκε στα πλαίσια της διδακτορικής διατριβής διαθέτει δύο τρόπους λειτουργίας. Ο πρώτος τρόπος λειτουργίας χρησιμοποιείται για τη λήψη μετρήσεων μέσω ηλεκτρονικού υπολογιστή (συγκεκριμένα μέσω της θύρας USB), ενώ ο δεύτερος χρησιμοποιείται για τη λήψη μετρήσεων μέσω δικτύου/διαδικτύου (χρησιμοποιώντας ένα απλό πρόγραμμα περιήγησης στον ιστό (*web*) όπως π.χ., Internet Explorer, Chrome, κλπ).

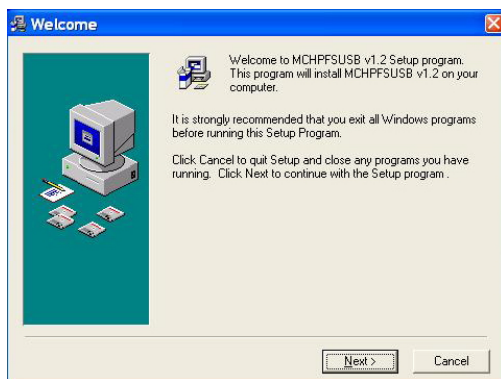
Στον πρώτο τρόπο λειτουργίας είναι απαραίτητη η εγκατάσταση του κατάλληλου οδηγού (*driver*) που καθιστά εφικτή την επικοινωνία του συντονιστή (*coordinator*) του συστήματος με τη θύρα USB του ηλεκτρονικού υπολογιστή, καθώς και του λογισμικού που επιτρέπει την έναρξη του εκτελέσιμου προγράμματος που αναπτύχθηκε σε Labview και που χρησιμοποιείται για τη συλλογή των μετρήσεων.

Στο δεύτερο τρόπο λειτουργίας δεν απαιτείται η εγκατάσταση κάποιου προγράμματος ή οδηγού. Κατά την εκκίνηση του συντονιστή, ο τελευταίος διερευνά εάν υπάρχει σύνδεση με τη θύρα USB του ηλεκτρονικού υπολογιστή και στην περίπτωση όπου εντοπίζεται σύνδεση, προχωρά στη λεγόμενη διαδικασία *απαρίθμησης (enumeration)* USB. Κατόπιν, ο συντονιστής λαμβάνει εντολές από το εκτελέσιμο πρόγραμμα Labview για τη συλλογή των μετρήσεων σε ηλεκτρονικό υπολογιστή. Σε περίπτωση που δεν εντοπίζεται σύνδεση με τη θύρα USB, ο συντονιστής εισάγεται σε λειτουργία λήψης μετρήσεων μέσω δικτύου/διαδικτύου.

Λειτουργία συλλογής μετρήσεων μέσω ηλεκτρονικού υπολογιστή

Την πρώτη φορά που συνδέεται ο συντονιστής του συστήματος στη θύρα USB ενός ηλεκτρονικού υπολογιστή απαιτείται η εκτέλεση συγκεκριμένων βημάτων για την αποκατάσταση της επικοινωνίας και τη λήψη μετρήσεων. Ακολουθως περιγράφεται η διαδικασία χρησιμοποιώντας λειτουργικό σύστημα Windows XP:

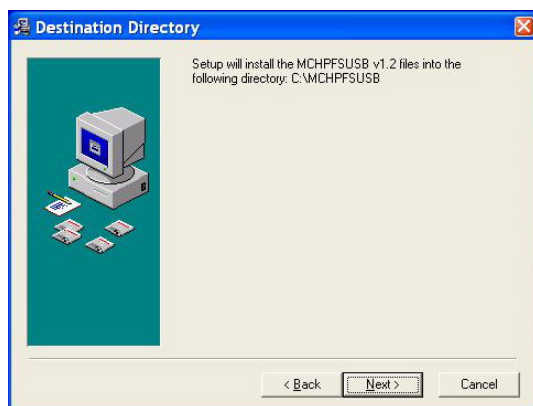
1. Εγκαθιστούμε το πρόγραμμα MCHPFSUSB_Setup_v1.2.exe που δίδεται από την εταιρία Microchip και περιλαμβάνει τον οδηγό επικοινωνίας του συντονιστή με τη θύρα USB ακολουθώντας τη διαδικασία των **Εικόνων Α—1** έως **Α—4**. Η εγκατάσταση αυτή δημιουργεί το φάκελο MCHPUSB και ο οδηγός επικοινωνίας του μικροελεγκτή PIC18F26J50 [1] (που περιλαμβάνεται στην συσκευή του συντονιστή) με τη θύρα USB εντοπίζεται στη διαδρομή του δίσκου C:\MCHPUSB\Pc\MCHPUSB Driver\Release (**Εικόνα Α—5**).



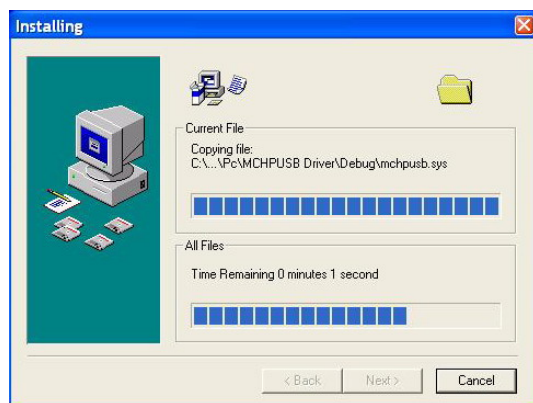
Εικόνα Α—1 Εγκατάσταση του MCHPFSUSB_Setup_v1.2.exe (Βήμα 1)



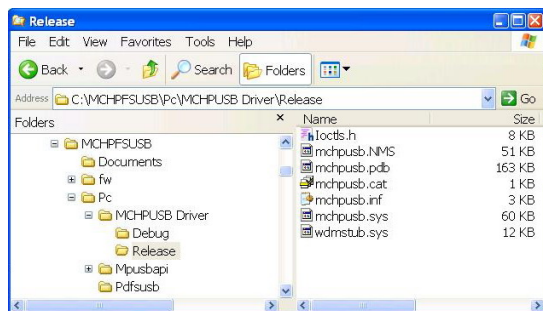
Εικόνα Α—2 Εγκατάσταση του MCHPFSUSB_Setup_v1.2.exe (Βήμα 2)



Εικόνα Α—3 Εγκατάσταση του MCHPFSUSB_Setup_v1.2.exe (Βήμα 3)

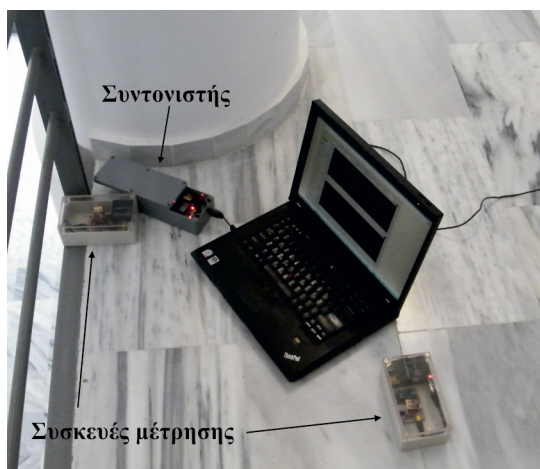


Εικόνα Α—4 Εγκατάσταση του MCHPFSUSB_Setup_v1.2.exe (Βήμα 4)



Εικόνα Α—5 Οδηγός επικοινωνίας του PIC18F26J50 με τη θύρα USB

2. Συνδέουμε τη συσκευή του συντονιστή σε μία θύρα USB του ηλεκτρονικού υπολογιστή (**Εικόνα Α—6**).



Εικόνα Α—6 Σύνδεση συντονιστή με ηλεκτρονικό υπολογιστή

3. Την πρώτη φορά που συνδέεται ο συντονιστής στη θύρα USB του ηλεκτρονικού υπολογιστή εμφανίζεται το μήνυμα της **Εικόνας Α—7**. Για την εγκατάσταση του οδηγού επικοινωνίας του μικροελεγκτή PIC18F26J50 με τη θύρα USB ακολουθούμε τα βήματα **Εικόνων Α—8** έως **Α—16**. Με το πέρας της εγκατάστασης θα πρέπει να εντοπίζεται στη διαχείριση συσκευών (*device manager*) των Windows η USB συσκευή που φαίνεται στην **Εικόνα Α—17**.



Εικόνα Α—7 Σύνδεση συντονιστή με Η/Υ (αναδυόμενο μήνυμα)



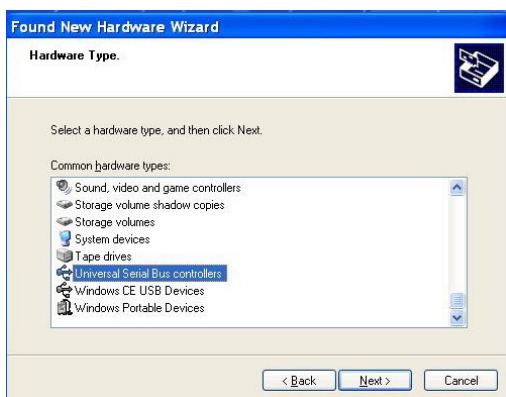
Εικόνα Α—8 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 1)



Εικόνα Α—9 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 2)



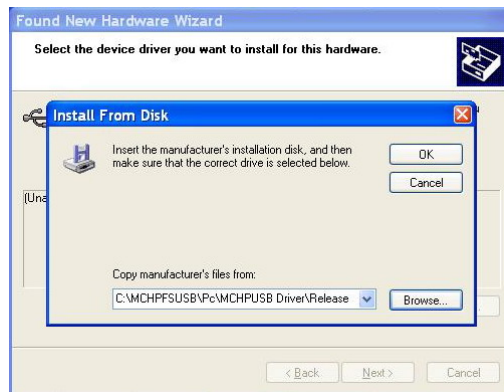
Εικόνα Α—10 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 3)



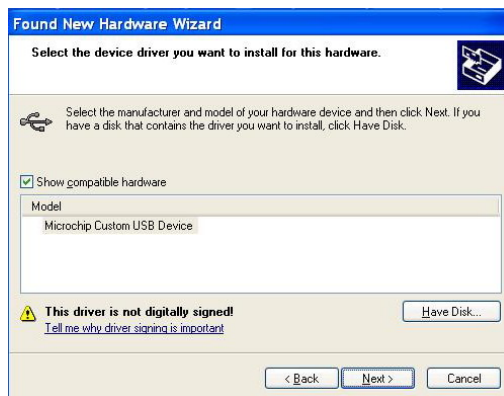
Εικόνα Α—11 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 4)



Εικόνα Α—12 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 5)



Εικόνα Α—13 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 6)



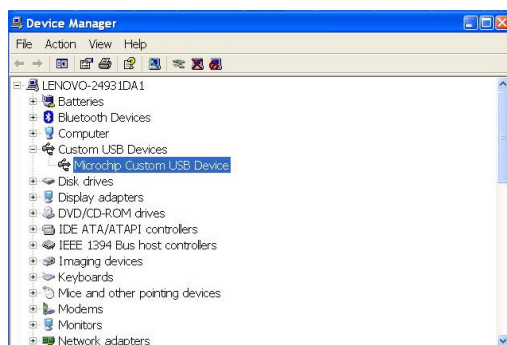
Εικόνα Α—14 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 7)



Εικόνα Α—15 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 8)

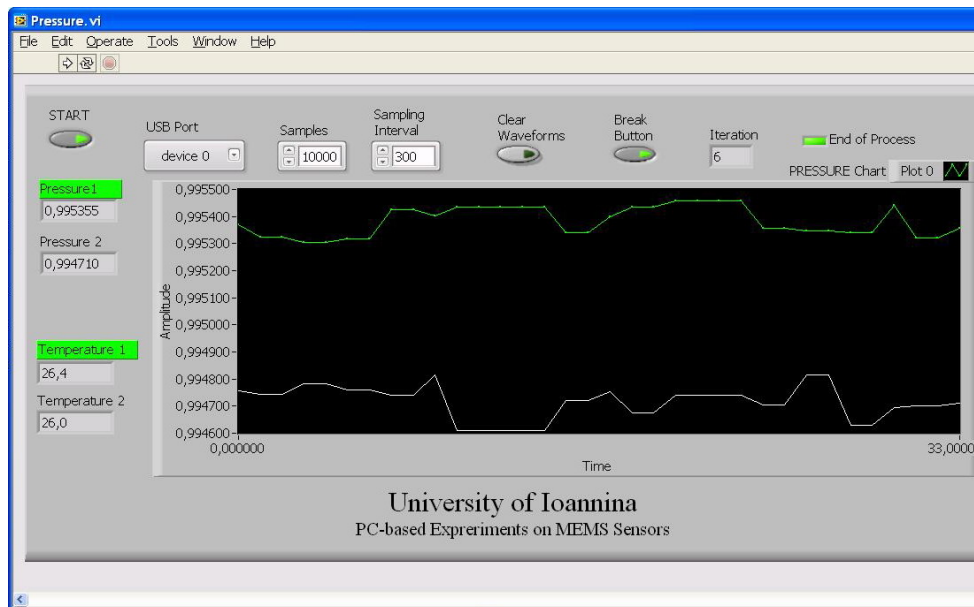


Εικόνα Α—16 Εγκατάσταση οδηγού επικοινωνίας του PIC18F26J50 (Βήμα 9)



Εικόνα Α—17 Διαχείριση συσκευών των Windows: η συσκευή USB

4. Στη συνέχεια θα πρέπει να γίνει η εγκατάσταση του προγράμματος LVRTE2011f3std.exe ώστε να είναι δυνατή η έναρξη λειτουργίας του εκτελέσιμου προγράμματος RemoteLab.exe. Το τελευταίο έχει αναπτυχθεί σε Labview και χρησιμοποιείται για τη λήψη και καταγραφή των μετρήσεων σε ηλεκτρονικό υπολογιστή.

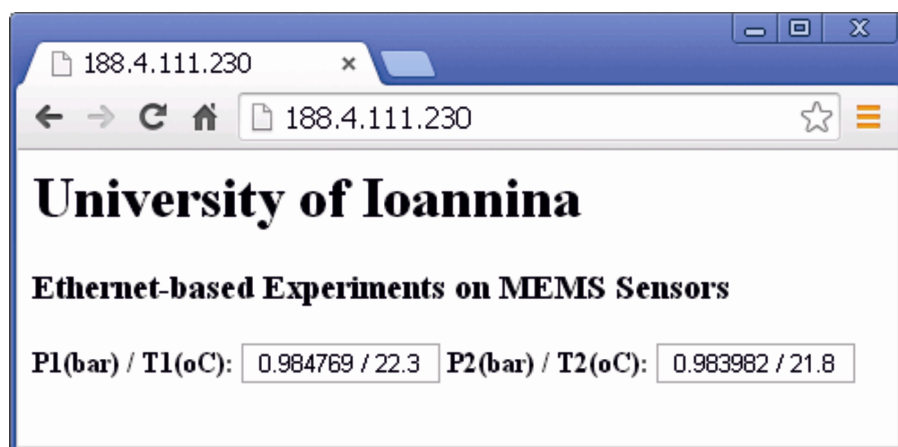


Εικόνα Α—18 Το πρόγραμμα συλλογής μετρήσεων RemoteLab.exe

5. Ακολουθώντας τα παραπάνω βήματα είναι πλέον εφικτή η λήψη μετρήσεων ατμοσφαιρικής πίεσης και θερμοκρασίας μέσω του προγράμματος RemoteLab.exe που εμφανίζεται στην **Εικόνα Α—18**. Ο χρήστης μπορεί να ορίσει τον επιθυμητό αριθμό δειγμάτων, το χρόνο καθυστέρησης μεταξύ διαδοχικών δειγμάτων σε msec (όπου η μικρότερη δυνατή τιμή είναι 400 msec), να πραγματοποιήσει εκκαθάριση των κυματομορφών από το πλήκτρο Clear Waveforms, αλλά και να διακόψει τη διαδικασία των μετρήσεων από το πλήκτρο Break Button. Η έναρξη λήψης μετρήσεων πραγματοποιείται με το πλήκτρο START, ενώ η καταγραφή των δειγμάτων γίνεται σε αρχείο κειμένου που φέρει το όνομα Pressure.txt, και το οποίο δημιουργείται αυτόματα στο φάκελο που περιλαμβάνεται το αρχείο RemoteLab.exe. Το αρχείο των μετρήσεων περιλαμβάνει 4 στήλες, όπου στις δύο πρώτες στήλες εντοπίζονται οι μετρήσεις ατμοσφαιρικής πίεσης και θερμοκρασίας, αντιστοίχως, της 1^{ης} συσκευής μέτρησης. Στις 2 επόμενες στήλες εντοπίζονται οι μετρήσεις ατμοσφαιρικής πίεσης και θερμοκρασίας, αντιστοίχως, της 2^{ης} συσκευής μέτρησης. Αξίζει να σημειωθεί ότι, κάθε φορά που γίνεται επανεκκίνηση λήψης μετρήσεων, οι προηγούμενες μετρήσεις παραμένουν στο αρχείο Pressure.txt.

Λειτουργία συλλογής μετρήσεων μέσω δικτύου/διαδικτύου

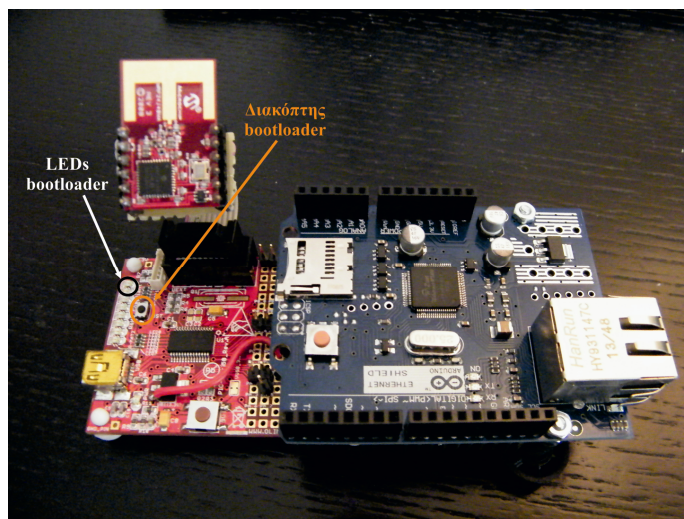
Όπως αναφέρθηκε προηγουμένως, η λειτουργία συλλογής μετρήσεων μέσω δικτύου/διαδικτύου δεν απαιτεί την εγκατάσταση κάποιου προγράμματος ή οδηγού. Η λήψη των δεδομένων πραγματοποιείται χρησιμοποιώντας ένα απλό πρόγραμμα περιήγησης δικτύου/διαδικτύου. Στην **Εικόνα Α—19** παρουσιάζεται η μορφή με την οποία εμφανίζονται οι μετρήσεις χρησιμοποιώντας το πρόγραμμα περιήγησης Chrome. Στη συγκεκριμένη περίπτωση η συλλογή μετρήσεων πραγματοποιείται μέσω του διαδικτύου (*internet*), ενώ τα δεδομένα ανανεώνονται κάθε φορά που πληκτρολογείται η διεύθυνση IP που έχει καταχωρηθεί στο υλικολογισμικό (*firmware*) του συντονιστή. Στο πρώτο πεδίο εντοπίζονται οι μετρήσεις ατμοσφαιρικής πίεσης και θερμοκρασίας (P1 και T1), αντιστοίχως, της 1^{ης} συσκευής μέτρησης. Στο δεύτερο πεδίο εντοπίζονται οι μετρήσεις ατμοσφαιρικής πίεσης και θερμοκρασίας (P2 και T2), αντιστοίχως, της 2^{ης} συσκευής μέτρησης. Στη λειτουργία αυτή ο χρήστης καταγράφει τις μετρήσεις είτε σε χαρτί είτε αρχείου κειμένου στον ηλεκτρονικό υπολογιστή με τη λειτουργία αντιγραφής-επικόλλησης. Αξίζει να σημειωθεί ότι, είναι δυνατή η αλλαγή της διεύθυνσης IP στο υλικολογισμικό του συντονιστή μέσω ειδικού εκτελέσιμου αρχείου που αναπτύχθηκε για το σκοπό αυτό, η χρήση του οποίου παρουσιάζεται στη συνέχεια.



Εικόνα Α—19 Συλλογή μετρήσεων από το διαδίκτυο μέσω του Chrome

Αναβάθμιση υλικολογισμικού στις συσκευές του συστήματος

Στη συσκευές του συστήματος έχει ενσωματωθεί ειδικός κώδικας στο υλικολογισμικό που καθιστά εφικτή την αναβάθμιση του τελευταίου. Ο ειδικός αυτός κώδικας καλείται *bootloader*. Η εισαγωγή των συσκευών του συστήματος σε λειτουργία *bootloader* πραγματοποιείται μέσω του διακόπτη επαναφοράς (*reset button*) που διατίθεται στις κάρτες του μικροελεγκτή PIC18F26J50 [2]. Στο σημείο αυτό θα πρέπει να σημειωθεί ότι οι κάρτες του μικροελεγκτή είναι κοινές και για τρεις συσκευές του συστήματος (συντονιστή και συσκευές μέτρησης).



Εικόνα Α—20 Κάρτες της συσκευής του συντονιστή

Στην Εικόνα Α—20 παρουσιάζονται οι κάρτες της συσκευής του συντονιστή. Η αναβάθμιση του υλικολογισμικού πραγματοποιείται μέσω της θύρας USB του ηλεκτρονικού υπολογιστή, χωρίς να απαιτείται η εγκατάσταση άλλου οδηγού καθώς ο κώδικας *bootloader* επικοινωνεί με τη θύρα USB μέσω του ίδιου οδηγού που χρησιμοποιείται για τη συλλογή των μετρήσεων (η εγκατάσταση του οποίου αναλύθηκε προηγουμένως).

Ακολουθώς περιγράφεται η διαδικασία αναβάθμισης τους υλικολογισμικού του συντονιστή:

1. Αρχικά συνδέουμε τη συσκευή του συντονιστή στην θύρα USB¹ του ηλεκτρονικού υπολογιστή.
2. Στη συνέχεια δεσμεύουμε το διακόπτη εκκίνησης (ή αλλιώς το διακόπτη *bootloader*). Στο σημείο αυτό ο κώδικας *bootloader* προχωρά στη διενέργεια

¹ Η σύνδεση του συντονιστή σε διαφορετική θύρα USB από αυτή που χρησιμοποιήθηκε προηγουμένως, απαιτεί ξανά την εγκατάσταση του οδηγού, όπως αυτή παρουσιάστηκε παραπάνω στο κείμενο.

απαρίθμησης USB και η επιτυχής διεξαγωγή της διαδικασίας αυτής, έχει ως αποτέλεσμα το άναμμα των δύο διόδων εκπομπής φωτός (LEDs) που εντοπίζονται στο πάνω αριστερό σημείο της κάρτας του μικροελεγκτή PIC18F26J50.

3. Η εκκίνηση αναβάθμισης του υλικολογισμικού πραγματοποιείται με την έναρξη του εκτελέσιμου προγράμματος program-mcu.exe (με διπλό αριστερό κλικ στο αρχείο αυτό), το οποίο αναπτύχθηκε στη γλώσσα προγραμματισμού C ειδικά για το σκοπό αυτό. Το αρχείο αυτό τοποθετείται στον ίδιο φάκελο με το δυαδικό αρχείο <όνομα-αρχείου>.hex που αποτελεί το υλικολογισμικό που θα φορτωθεί στο μικροελεγκτή του συντονιστή. Στον ίδιο φάκελο θα πρέπει επίσης να τοποθετηθεί και η βιβλιοθήκη mpusbapi.dll² που παρέχεται από την εταιρία Microchip [3]. Η εκκίνηση του εκτελέσιμου αρχείου program-mcu.exe φορτώνει το αρχείο <όνομα-αρχείου>.hex στη μνήμη του μικροελεγκτή και επαληθεύει στη ορθή μετάδοση των δεδομένων από τη θύρα USB. Η επιτυχής ολοκλήρωση της διαδικασίας αναδύει το μήνυμα Code successfully downloaded! στην κονσόλα εντολών (command prompt) και τερματίζει τη λειτουργία της τελευταίας (Εικόνα Α—21). Σε αντίθετη περίπτωση εμφανίζεται το μήνυμα Download error! Press any key to continue. Με την επιτυχή ολοκλήρωση του προγραμματισμού του μικροελεγκτή, ο συντονιστής εισάγεται αυτομάτως σε λειτουργία εκτέλεσης του – αναβαθμισμένου πλέον – υλικολογισμικού. Δεδομένου ότι οι συντονιστές είναι συνδεδεμένοι στη θύρα USB, ο κώδικας εισάγεται σε λειτουργία συλλογής δεδομένων μέσω ηλεκτρονικού υπολογιστή (πραγματοποιείται δηλαδή η διαδικασία απαρίθμησης USB).

```

C:\z\SAVE\PHDI\FIRMWARE_Eth-USB-based_RemoteLab_plus-temperature\COO...
:105010004B6F200E4C6F0D0E4D6F0A0E4E6F4F6B87
:105020000C6B3C0E8B6F020E8E6F4A0E8D6F0001E3
:105030000A0EC23F0020EEA6E4A0EE96EF2CF89F27E
:10504000F29E3F0E016EF76A000E54EC11F00A005A
:10505000090B0F5C0EEFF012EF0D70201090BF28ECA
:105060000C6E3C0E8B6F020E8E6F4A0E8D6F0001E3
:105070000A0EC23F002018C6B3C0E8B6F020E8E6F46
:10508000390E8D6F0001A0EC23F0220E02014A6F51
:105090003E0E4B6F200E4C6F0D0E4D6F0A0E4E6F75
:1050A0004F6B8C6B3C0E8B6F020E8E6F4A0E8D6FAA
:1050B000001A0EC23F002018A6B3C0E896F000115
:1050C00009EC23F001C089F23BC08AF202018C6B3
:1050D0003C0E8B6F8E6B01C08DF200010FE2C4F043
:1050E00002C08AF201C089F20201895305E18A5304
:1050F00003E1000114D0020101013BC099F200015B
:10510000F6EC23F007D00101060E006E002EFED74C
:105110000000000104D03BC08EF2F9EC21F0020146
:0A5120000001D0D4000E016E030060
:08FF8000CF412F7DF9FF172
:00000001FF

Code successfully downloaded!

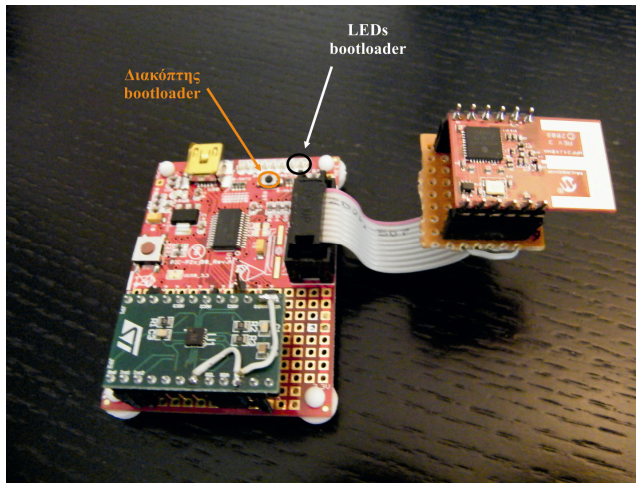
```

Εικόνα Α—21 Εκκίνηση του αρχείου program-mcu.exe

Στο σημείο αυτό θα πρέπει να σημειωθεί ότι ο κώδικας του συντονιστή που συνηθώς απαιτεί αναβάθμιση αφορά τις ρυθμίσεις διαδικτύου. Ένα ενδεικτικό τμήμα του κώδικα αυτού, το οποίο εντοπίζεται στο αρχείο Coordinator.h, έχει ως εξής:

² Το αρχείο αυτό εντοπίζεται στη διαδρομή C:\MCHPFSUSB\Pc\Mpusbapi\Dll\Borland_C\Source κατά την εγκατάσταση του MCHPFSUSB_Setup_v1.2.exe, όπως αυτή παρουσιάστηκε παραπάνω στο κείμενο.

```
// Global constants for W5100
unsigned char mac_addr[] = {0x00,0xE0,0xA0,0x00,0x9E,0x9F};
unsigned char ip_addr[]  = {192,168,1,177};
unsigned char sub_mask[] = {255,255,255,0};
unsigned char gtw_addr[] = {192,168,1,1};
```



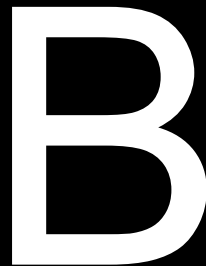
Εικόνα Α—22 Κάρτες της συσκευής μέτρησης

Αναβάθμιση υλικολογισμικού είναι δυνατή και στις συσκευές μέτρησης (Εικόνα Α—22) χρησιμοποιώντας παρόμοια διαδικασία προγραμματισμού. Στην περίπτωση αυτή θα πρέπει προηγηθεί η διαδικασία εγκατάστασης του οδηγού επικοινωνίας του μικροελεγκτή κατά την πρώτη σύνδεση των συσκευών μέτρησης στη θύρα USB του ηλεκτρονικού υπολογιστή.

Αναφορές

- [1] *PIC18F46J50 Data Sheet*, Microchip Technology Incorporated, 2012.
- [2] *PIC-P26J50 Proto development board USER'S MANUAL*, Olimex Ltd, 2012.
- [3] *MCHPFSUSB Firmware User's Guide*, Microchip Technology Incorporated, 2007.

Οδηγός ανάλυσης δεδομένων στο Matlab



ΣΤΟ ΠΑΡΑΡΤΗΜΑ ΑΥΤΟ

ΜΕΤΡΗΣΕΙΣ ΑΤΜΟΣΦΑΙΡΙΚΗΣ ΠΙΕΣΗΣ
Δίνονται οδηγίες καταγραφής μετρήσεων ατμοσφαιρικής πίεσης και εξετάζεται η μορφή και ονοματολογία των αρχείων μετρήσεων για την αυτόματη ανάλυση δεδομένων μέσω των διαθέσιμων προγραμμάτων Matlab.

ΑΝΑΛΥΣΗ ΜΕΤΡΗΣΕΩΝ ΔΙΑΦΟΡΙΚΗΣ ΒΑΡΟΜΕΤΡΙΚΗΣ ΥΨΟΜΕΤΡΙΑΣ
Δίνεται ένα παράδειγμα ανάλυσης μετρήσεων διαφορικής βαρομετρικής υψομετρίας μέσω του διαθέσιμου κώδικα Matlab: MatlabScript_DBA.m.

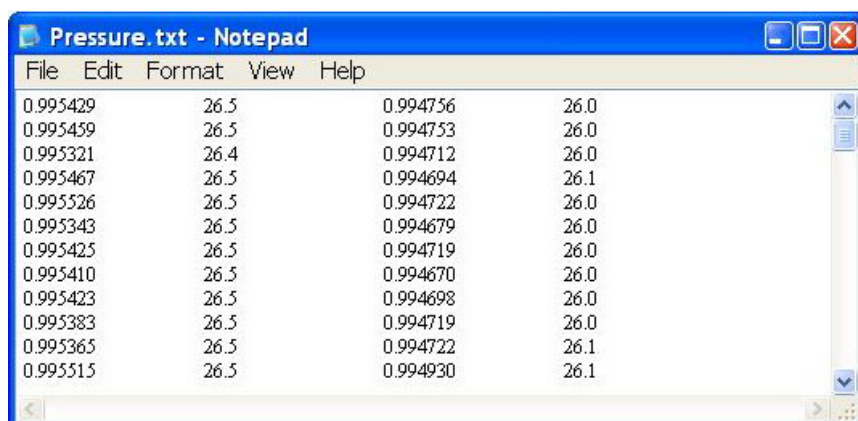
ΑΝΑΛΥΣΗ ΜΕΤΡΗΣΕΩΝ ΑΠΛΗΣ ΒΑΡΟΜΕΤΡΙΚΗΣ ΥΨΟΜΕΤΡΙΑΣ
Δίνεται ένα παράδειγμα ανάλυσης μετρήσεων απλής βαρομετρικής υψομετρίας μέσω του διαθέσιμου κώδικα Matlab: MatlabScript_SBA.m.

ΜΕΤΡΗΣΕΙΣ ΣΕ ΑΕΡΟΣΤΕΓΕΣ ΔΟΧΕΙΟ
Δίνονται οδηγίες καταγραφής μετρήσεων σε αεροστεγές δοχείο.

ΑΝΑΛΥΣΗ ΜΕΤΡΗΣΕΩΝ ΣΕ ΑΕΡΟΣΤΕΓΕΣ ΔΟΧΕΙΟ
Δίνεται ένα παράδειγμα ανάλυσης μετρήσεων σε αεροστεγές δοχείο μέσω του κώδικα MatlabScript_DEVIATION.m.

Μετρήσεις ατμοσφαιρικής πίεσης

Στα πλαίσια της διδακτορικής διατριβής αναπτύχθηκαν αρχεία εντολών (*script files*) αυτόματης ανάλυσης των μετρήσεων στο πρόγραμμα Matlab. Για την ανάλυση των δεδομένων ατμοσφαιρικής πίεσης και τον υπολογισμό απόλυτου ύψους ανάμεσα στις δύο συσκευές μέτρησης, κρίνεται η απαραίτητη η δημιουργία δύο αρχείων με το ακριβές όνομα PressureSame.txt και PressureDifferent.txt. Τα αρχεία αυτά περιλαμβάνουν τις μετρήσεις ατμοσφαιρικής πίεσης και θερμοκρασίας που προκύπτουν, αντιστοίχως, από την τοποθέτηση των συσκευών στο ίδιο ύψος¹ και κατόπιν στη διαφορά απόλυτου ύψους που πρόκειται να εκτιμηθεί.



Pressure.txt - Notepad				
File	Edit	Format	View	Help
0.995429	26.5	0.994756	26.0	
0.995459	26.5	0.994753	26.0	
0.995321	26.4	0.994712	26.0	
0.995467	26.5	0.994694	26.1	
0.995526	26.5	0.994722	26.0	
0.995343	26.5	0.994679	26.0	
0.995425	26.5	0.994719	26.0	
0.995410	26.5	0.994670	26.0	
0.995423	26.5	0.994698	26.0	
0.995383	26.5	0.994719	26.0	
0.995365	26.5	0.994722	26.1	
0.995515	26.5	0.994930	26.1	

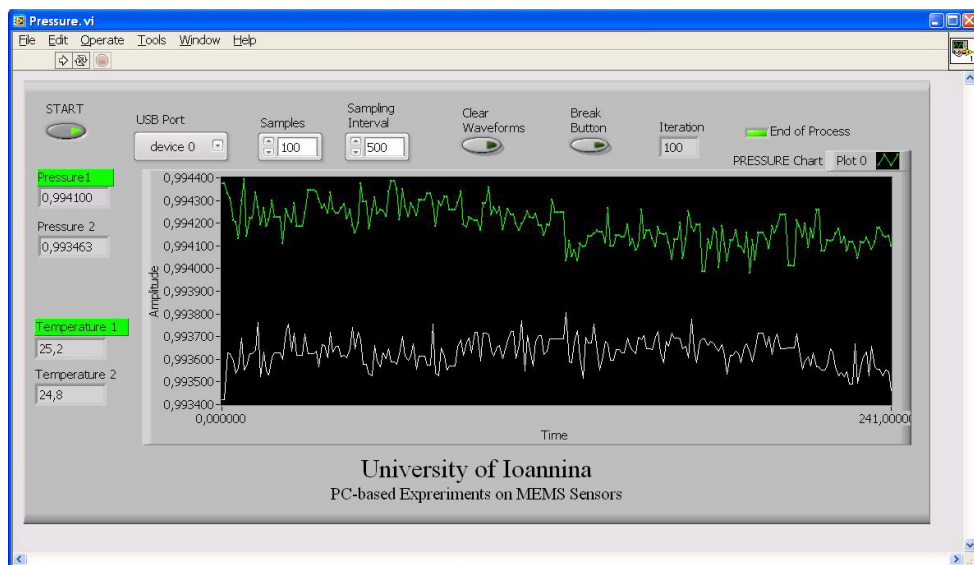
Εικόνα Β—1 Μορφή καταγραφής δεδομένων σε αρχείο κειμένου

Στην Εικόνα Β—1 δίδεται η μορφή η μορφή που θα πρέπει να έχουν τα δύο αρχεία. Στην περίπτωση αυτή απεικονίζεται το αρχείο Pressure.txt² που προκύπτει αυτόματα από το περιβάλλον συλλογής μετρήσεων που αναπτύχθηκε στο Labview. Στις δύο πρώτες στήλες εντοπίζονται οι μετρήσεις της 1^{ης} συσκευής μέτρησης, ενώ στις δύο τελευταίες στήλες εντοπίζονται οι μετρήσεις της 2^{ης} συσκευής μέτρησης. Όσον αφορά τον τρόπο λειτουργίας του συστήματος μέσω ηλεκτρονικού υπολογιστή, ο χρήστης χρειάζεται απλώς να καταγράψει τον επιθυμητό αριθμό μετρήσεων μέσω του εκτελέσιμου προγράμματος. Στη συνέχεια ο χρήστης διαχωρίζει τις μετρήσεις (ίδιου ύψους και διαφοράς ύψους) σε δύο αρχεία, δίνοντας σε κάθε αρχείο

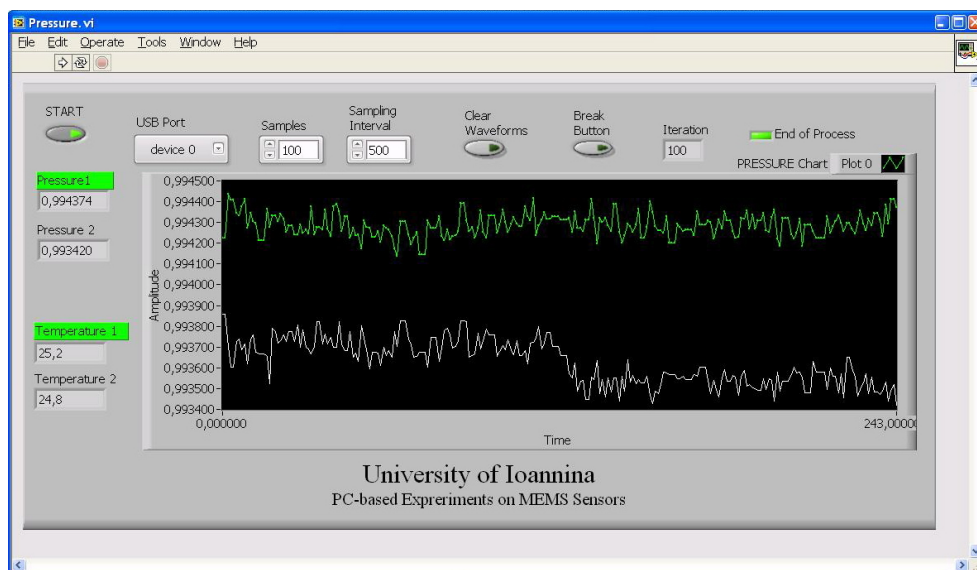
¹ Η αρχική αυτή μέτρηση προκύπτει από την αναγκαιότητα βαθμονόμησης των αισθητήρων καθότι, όπως έχει αναφερθεί σε προηγούμενο κεφάλαιο, οι αισθητήρες βαρομετρικής πίεσης παρουσιάζουν απόκλιση στο σήμα εξόδου κατά τη μέτρηση της ίδιας τιμής ατμοσφαιρικής πίεσης λόγω κατασκευαστικών ανομοιοτήτων.

² Αξίζει να σημειωθεί ότι έχει δημιουργηθεί το εκτελέσιμο αρχείο replace-decimal-point.exe (σε κώδικα Python3.4), το οποίο αντικαθιστά αυτόματα στο αρχείο Pressure.txt, όλους τους χαρακτήρες τελείας (.) σε κόμμα (,). Η μορφή αυτή είναι η αποδεκτή μορφή δεδομένων που δέχεται η Ελληνική έκδοση του Microsoft Excel (σε περίπτωση που κάποιος επιθυμεί να προβεί σε ανάλυση δεδομένων με το πρόγραμμα αυτό).

ίο το χαρακτηριστικό όνομα που αναφέρθηκε προηγουμένως. Στον τρόπο λήψης μετρήσεων μέσω δικτύου/διαδικτύου, ο χρήστης θα πρέπει να δημιουργήσει τα δύο αρχεία μετρήσεων και να μεταφέρει ο ίδιος τις μετρήσεις από τα αντίστοιχα πεδία του προγράμματος περιήγησης (π.χ. Chrome) στα αρχεία μετρήσεων, με τη λειτουργία της αντιγραφής-επικόλλησης (ή με την πληκτρολόγηση των τιμών).



Εικόνα Β—2 Μετακίνηση της 1^{ης} συσκευής στα 1,1μ (μετρήσεις 101-200)



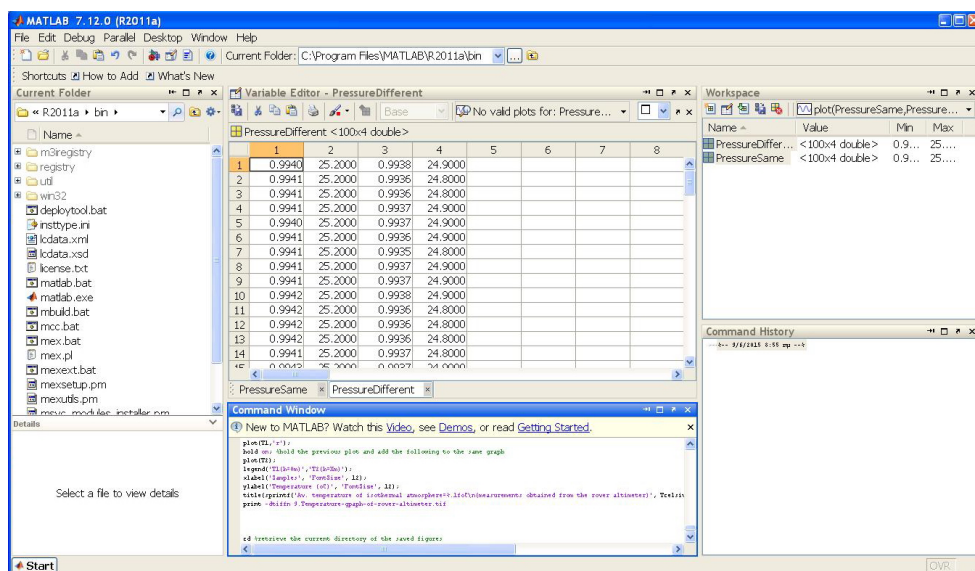
Εικόνα Β—3 Μετακίνηση της 2^{ης} συσκευής στα 1,1μ (μετρήσεις 101-200)

Στην **Εικόνα Β—2** δίνεται ένα παράδειγμα λήψης 200 μετρήσεων μέσω ηλεκτρονικού υπολογιστή, όπου στις 100 πρώτες μετρήσεις οι συσκευές έχουν τοποθετηθεί αρχικά στο ίδιο ύψος, ενώ στις 100 επόμενες μετρήσεις η πρώτη συσκευή (γραφική με πράσινο χρώμα) τοποθετήθηκε σε διαφορά απόλυτου ύψους 1,1μ υψηλότερα από τη δεύτερη συσκευή (γραφική με άσπρο χρώμα). Στην περίπτωση αυτή παρατηρείται μία σχετική μείωση στη μετρούμενη ατμοσφαιρική πίεση στη γραφική με το πράσινο χρώμα (μετρήσεις 101-200).

Στην **Εικόνα Β—3** δίνεται το αντίστοιχο παράδειγμα χρησιμοποιώντας ως κινητή συσκευή τη δεύτερη συσκευή του συστήματος και συνεπώς, η σχετική μείωση στη μετρούμενη ατμοσφαιρική πίεσης παρατηρείται στη γραφική με το άσπρο χρώμα. Αξίζει να σημειωθεί ότι, το αρχείο εντολών Matlab πραγματοποιεί ορθή ανάλυση δεδομένων ανεξάρτητα από το ποια συσκευή ορίζει ο χρήστης ως την κινητή συσκευή του συστήματος, αλλά και ανεξάρτητα αν το σήμα εξόδου της πρώτης συσκευής μέτρησης (Pressure 1) έχει μεγαλύτερη ή μικρότερη τιμή από το σήμα εξόδου της δεύτερης συσκευής (Pressure 2).

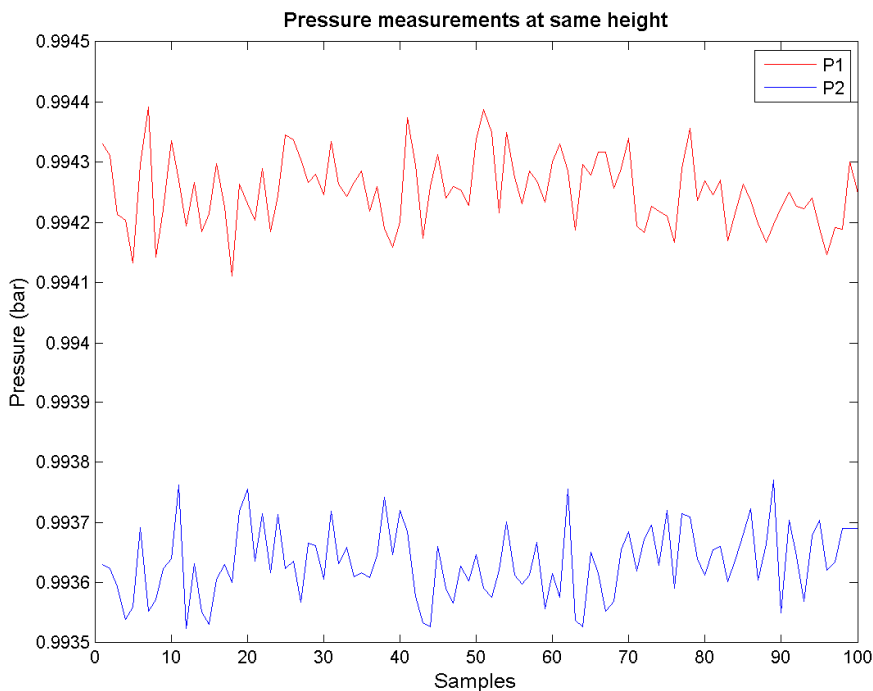
Ανάλυση μετρήσεων διαφορικής βαρομετρικής υψομετρίας

Στη συνέχεια αναλύεται το αρχείο εντολών Matlab που χρησιμοποιείται για τον υπολογισμό απόλυτου ύψους εφαρμόζοντας την τεχνική διαφορικής βαρομετρικής υψομετρίας (*differential barometric altimetry*) και φέρει το χαρακτηριστικό όνομα MatlabScript_DBA.m. Το παράδειγμα χρησιμοποιεί τα δεδομένα της **Εικόνας Β—2**.



Εικόνα Β—4 Εισαγωγή των αρχείων μετρήσεων και εντολών στο Matlab

Με την λειτουργία μεταφοράς-και-απόθεσης (*drag-and-drop*) του λειτουργικού συστήματος εισάγουμε τα δύο αρχεία μετρήσεων (PressureSame.txt και PressureDifferent.txt) στο παράθυρο Workspace του Matlab (**Εικόνα Β—4**). Κάνοντας διπλό αριστερό κλικ σε κάποιο από τα εισαγόμενα αρχεία, μπορούμε να δούμε (ή να επεξεργαστούμε) τις μετρήσεις στο παράθυρο Variable Editor. Κατόπιν αντιγράφουμε τον κώδικα του αρχείου MatlabScript_DBA.m στο παράθυρο Command Window του Matlab και πατάμε Enter. Με την τελευταία διενέργεια αναδύονται 9 παράθυρα γραφικής ανάλυσης των δεδομένων. Τα παράθυρα αυτά αποθηκεύονται αυτόματα με τη μορφή εικόνας .tif στη διαδρομή του δίσκου C:\Program Files\MATLAB\R2011a\bin. Στη συνέχεια εξετάζονται τα τμήματα του κώδικα για την κάθε γραφική ανεξάρτητα.



Εικόνα Β—5 Μετρήσεις ατμοσφαιρικής πίεσης σε ύψος $Y=0\mu$

Οι ακόλουθες εντολές αναπαριστούν γραφικά τις μετρήσεις του αρχείου PressureSame.txt. Αναλυτικά, στη γραμμή 1 φορτώνονται τα δεδομένα από το αρχείο PressureSame.txt στον πίνακα file. Στις γραμμές 2 και 3 φορτώνονται τα δεδομένα των στηλών 1 και 3 (του πίνακα file) στους μονοδιάστατους πίνακες P1 και P2, αντιστοίχως. Στις γραμμές 4 και 5 γίνεται εκτύπωση, σε εικόνα, των δεδομένων του μονοδιάστατου πίνακα P1 με κόκκινη γραφική (όπως ορίζεται από την οδηγία χρώματος 'r'). Στις γραμμές 6 και 7 γίνεται εκτύπωση, στην ίδια εικόνα (εντολή 6), των δεδομένων P2 με μπλε γραφική (η προκαθορισμένη επιλογή χρώματος όταν δε χρησιμοποιείται σχετική οδηγία). Στις γραμμές 7-11 εκτυπώνονται στην εικόνα, με τη σειρά, οι επεξηγήσεις γραφικών P1 και P2 (δηλαδή ποιο χρώμα αντιστοιχεί στη κάθε γραφική), η ετικέτες των αξόνων x και y και ο τίτλος της εικόνας. Τέλος, στη γραμμή 12 γίνεται η αποθήκευση της εικόνας σε μορφή .tif και με όνομα 1.Pressure-measurements-at-same-height. Το αποτέλεσμα εκτέλεσης των εντολών δίνεται στην **Εικόνα Β—5**.

```

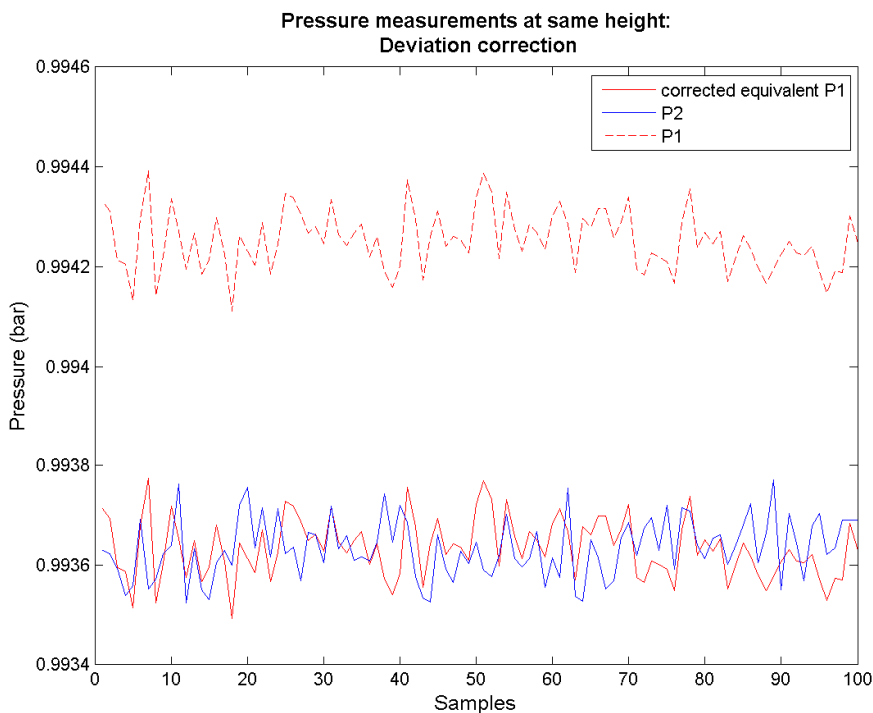
1   file = PressureSame;
2   P1=file(:,1);
3   P2=file(:,3);
4   fig = figure;
5   plot(P1,'r');
6   hold on;

```

```

7   plot(P2);
8   legend('P1','P2');
9   xlabel('Samples','FontSize',12);
10  ylabel('Pressure (bar)','FontSize',12);
11  title(sprintf('Pressure measurements at same height'),'FontSize',12,
    'FontWeight','bold');
12  print -dtiffn 1.Pressure-measurements-at-same-height.tif;

```



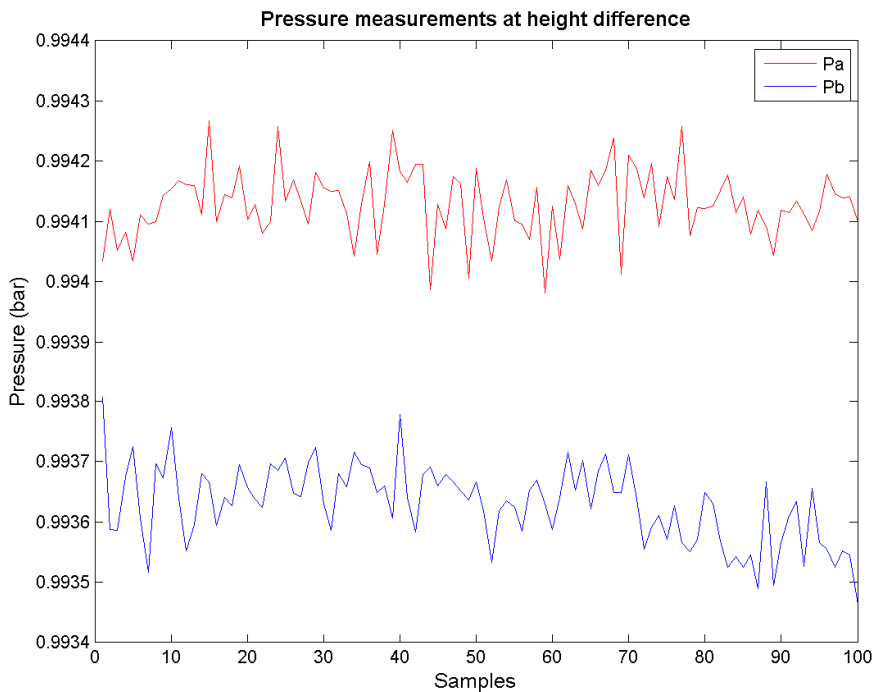
Εικόνα Β—6 Μετρήσεις ατμοσφαιρικής πίεσης σε ύψος $Y=0\text{m}$ (διόρθωση P1)

Οι ακόλουθες εντολές υπολογίζουν τη μέση τιμή της απόκλισης στο σήμα εξόδου των γραφικών P1 και P2 (γραμμές 13-21) και πραγματοποιούν διόρθωση στη γραφική P1 (γραμμή 16/γραμμή 20). Η συνθήκη if-else που χρησιμοποιείται στις γραμμές 13-21 διερευνά αν η γραφική P1 είναι μεγαλύτερη από την P2 και αν ισχύει η συνθήκη, αφαιρεί από την κάθε τιμή του πίνακα P1 τη μέση τιμή της απόκλισης PdifAv (γραμμή 16). Διαφορετικά, προσθέτει τη μέση τιμή της απόκλισης PdifAv (γραμμή 20). Στις γραμμές 22-31 εκτυπώνονται στην εικόνα οι γραφικές P1, P2 και P1b, όπου η τελευταία αποτελεί τη διορθωμένη ισοδύναμη της γραφική της P1. Η αρχική γραφική P1 εκτυπώνεται με διακεκομμένη κόκκινη γραμμή, όπως ορίζει η οδηγία `--r` στη γραμμή 27, ενώ η γραφική P2 με μπλε και η διορθωμένη γραφική P1b με κόκκινο χρώμα. Τέλος, στη γραμμή 32 γίνεται η αποθήκευση της εικόνας με όνομα 2.Pressure-measurements-at-same-height-Deviation-correction.tif. Το αποτέλεσμα εκτέλεσης των εντολών δίνεται στην **Εικόνα Β—6**.

```

13  if (P1>P2)
14  Pdif = P1-P2;
15  PdifAv=mean(Pdif);
16  P1b=P1-PdifAv;
17  else
18  Pdif = P2-P1;
19  PdifAv=mean(Pdif);
20  P1b=P1+PdifAv;
21  end
22  figure;
23  plot(P1b,'r');
24  hold on;
25  plot(P2);
26  hold on;
27  plot(P1, '--r');
28  legend('corrected equivalent P1', 'P2', 'P1', 'FontSize', 6);
29  xlabel('Samples', 'FontSize', 12);
30  ylabel('Pressure (bar)', 'FontSize', 12);
31  title(sprintf('Pressure measurements at same height:\n Deviation
correction'), 'FontSize', 12, 'FontWeight', 'bold');
32  print -dtiffn 2.Pressure-measurements-at-same-height-Deviation-
correction.tif;

```



Εικόνα Β—7 Μετρήσεις ατμοσφαιρικής πίεσης σε ύψος $Y=1,1\mu$

Οι ακόλουθες εντολές αναπαριστούν γραφικά τις μετρήσεις του αρχείου

PressureDifferent.txt. Αναλυτικά, στη γραμμή 33 φορτώνονται τα δεδομένα από το αρχείο PressureDifferent.txt στον πίνακα file. Στις γραμμές 34 και 35 φορτώνονται τα δεδομένα των στηλών 1 και 3 (του πίνακα file) στους μονοδιάστατους πίνακες Pa και Pb, αντιστοίχως. Στις γραμμές 36-43 γίνεται εκτύπωση των γραφικών Pa και Pb (με κόκκινο και μπλε χρώμα, αντιστοίχως) καθώς και των σχετικών επεξηγήσεων της εικόνας. Η γραφική Pa αντιστοιχεί στην κινητή συσκευή του συστήματος, η οποία βρίσκεται στη θέση 1,1μ υψηλότερα από τη σταθερή συσκευή (όπου η τελευταία παραμένει στην ίδια θέση όπως προηγουμένως). Τέλος, στη γραμμή 44 γίνεται η αποθήκευση της εικόνας με όνομα 3.Pressure-measurements-at-height-difference.tif. Το αποτέλεσμα εκτέλεσης των εντολών δίνεται στην **Εικόνα Β—7**.

```

33   file = PressureDifferent;
34   Pa=file(:,1);
35   Pb=file(:,3);
36   figure;
37   plot(Pa,'r');
38   hold on;
39   plot(Pb);
40   legend('Pa','Pb');
41   xlabel('Samples','FontSize',12);
42   ylabel('Pressure (bar)','FontSize',12);
43   title(sprintf('Pressure measurements at height difference'),
44         'FontSize',12,'FontWeight','bold');
44   print -dtiffn 3.Pressure-measurements-at-height-difference.tif;
```

Οι ακόλουθες εντολές πραγματοποιούν διόρθωση στη γραφική Pa, χρησιμοποιώντας τη μέση τιμή της απόκλισης PdifAv που υπολογίστηκε προηγουμένως (σε ύψος Υ=0μ). Η συνθήκη if-else που χρησιμοποιείται στις γραμμές 45-49 διερευνά αν η γραφική Pa είναι μεγαλύτερη από την Pb και αν ισχύει η συνθήκη, αφαιρεί από την κάθε τιμή του πίνακα Pa τη μέση τιμή της απόκλισης PdifAv (γραμμή 46). Διαφορετικά, προσθέτει τη μέση τιμή της απόκλισης PdifAv (γραμμή 48). Στις γραμμές 50-59 εκτυπώνονται στην εικόνα οι γραφικές Pa, Pb και Pa2, όπου η τελευταία αποτελεί τη διορθωμένη ισοδύναμη της γραφική της Pa. Η αρχική γραφική Pa εκτυπώνεται με διακεκομμένη κόκκινη γραμμή, ενώ η γραφική Pb με μπλε και η διορθωμένη γραφική Pa2 με κόκκινο χρώμα. Τέλος, στη γραμμή 60 γίνεται η αποθήκευση της εικόνας με όνομα 4.Pressure-measurements-at-height-difference-Deviation-correction.tif. Το αποτέλεσμα εκτέλεσης των εντολών δίνεται στην **Εικόνα Β—8**.

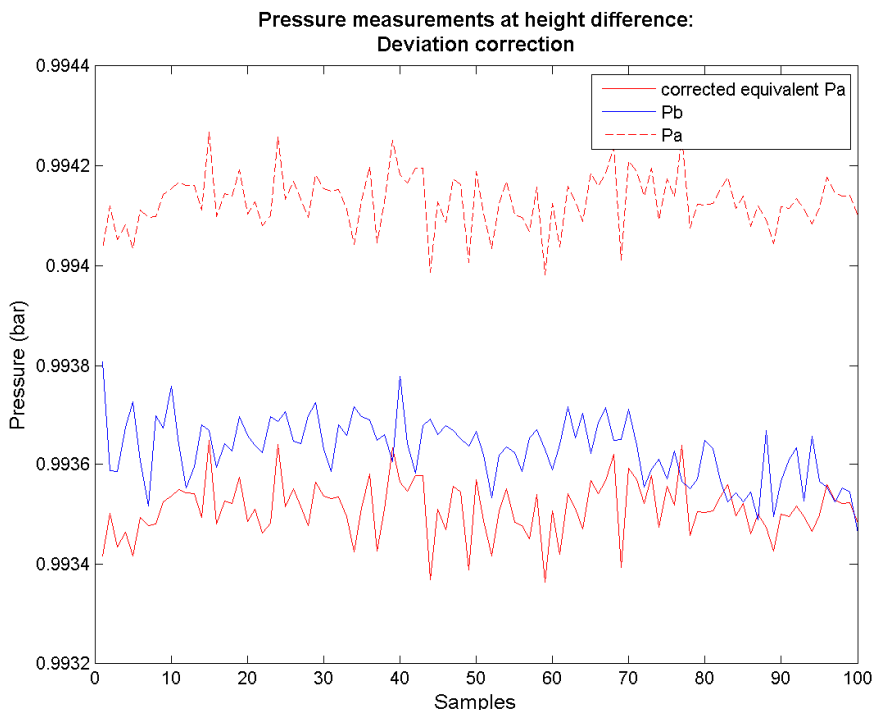
```

45   if (Pa>Pb)
46   Pa2=Pa-PdifAv;
47   else
48   Pa2=Pa+PdifAv;
49   end
50   figure;
51   plot(Pa2,'r');
52   hold on;
53   plot(Pb);
54   hold on;
55   plot(Pa,'--r');
```

```

56 legend('corrected equivalent Pa', 'Pb', 'Pa', 'FontSize', 6);
57 xlabel('Samples', 'FontSize', 12);
58 ylabel('Pressure (bar)', 'FontSize', 12);
59 title(sprintf('Pressure measurements at height difference:\nDeviation
correction'), 'FontSize', 12, 'FontWeight', 'bold');
60 print -dtiffn 4.Pressure-measurements-at-height-difference-Deviation-
correction.tif

```



Εικόνα Β—8 Μετρήσεις ατμοσφαιρικής πίεσης σε ύψος $Y=1,1\mu$ (διόρθωση Pa)

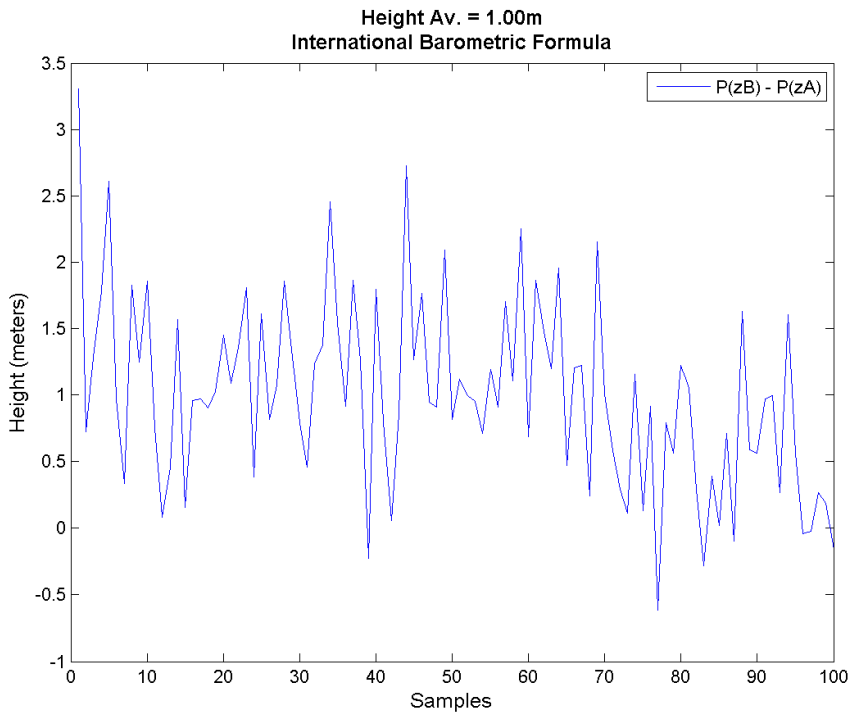
Οι ακόλουθες εντολές υπολογίζουν την απόλυτη διαφορά ύψους μεταξύ των δύο συσκευών μέτρησης με τη χρήση της διεθνούς βαρομετρικής εξίσωσης (*international barometric formula*) [1], όπως ορίζεται από τον **Τύπο Β—1**. Η συνθήκη if-else που χρησιμοποιείται στις γραμμές 61-65 διερευνά αν η διορθωμένη ισοδύναμη Pa_2 είναι μεγαλύτερη από την P_b και αν ισχύει η συνθήκη, τότε εκτελείται η γραμμή 62. Σε αντίθετη περίπτωση εκτελείται η γραμμή 64. Στο παράδειγμά μας ισχύει η αντιστοίχιση $P_{(ZA)}=Pa_2$ και $P_{(ZB)}=P_b$ και δεδομένου ότι $Pa_2 < P_b$, εκτελείται η γραμμή 64. Στη γραμμή 66 υπολογίζεται η μέση τιμή του απόλυτου ύψους και στις γραμμές 67-72 εκτυπώνεται η γραφική αναπαράσταση του τελευταίου. Στον τίτλο της εικόνας εκτυπώνεται επίσης και η μέση τιμή του μετρήσιμου ύψους με δύο δεκαδικά ψηφία, όπως ορίζεται από την οδηγία %.2f στη γραμμή 72. Στη συγκεκριμένη περίπτωση η μέση τιμή του μετρήσιμου ύψους είναι 1μ. Τέλος, στη γραμμή 73 γίνεται η αποθή-

κευση της εικόνας με όνομα 5.Height-Graph-International-Barometric-Formula.tif. Το αποτέλεσμα εκτέλεσης των εντολών δίνεται στην **Εικόνα Β—9**.

```

61 if (mean(Pa2) > mean(Pb))
62 height=4935.125 * ( power((Pa2*100000),0.1903) -
    power((Pb*100000),0.1903) );
63 else
64 height=4935.125 * ( power((Pb*100000),0.1903) -
    power((Pa2*100000),0.1903) );
65 end
66 average_height = mean(height);
67 figure
68 plot(height)
69 legend('P(zB) - P(zA)', 'FontSize', 6);
70 xlabel('Samples', 'FontSize', 12);
71 ylabel('Height (meters)', 'FontSize', 12);
72 title(sprintf('Height Av. = %.2fm\nInternational Barometric Formula',
    average_height), 'FontSize', 12, 'FontWeight', 'bold');
73 print -dtiffn 5.Height-Graph-International-Barometric-Formula.tif

```



Εικόνα Β—9 Γραφική ύψους (υπολογισμός με διεθνή βαρομετρική εξίσωση)

$$\begin{aligned}
 h &= P_{(zA)} - P_{(zB)} = -4935,125 \cdot P_{(zA)}^{0.1903} + 4935,125 \cdot P_{(zB)}^{0.1903} = \\
 &= 4935,125 \cdot (P_{(zB)}^{0.1903} - P_{(zA)}^{0.1903}), \quad \text{όπου: } P_{(zA)} < P_{(zB)}
 \end{aligned}$$

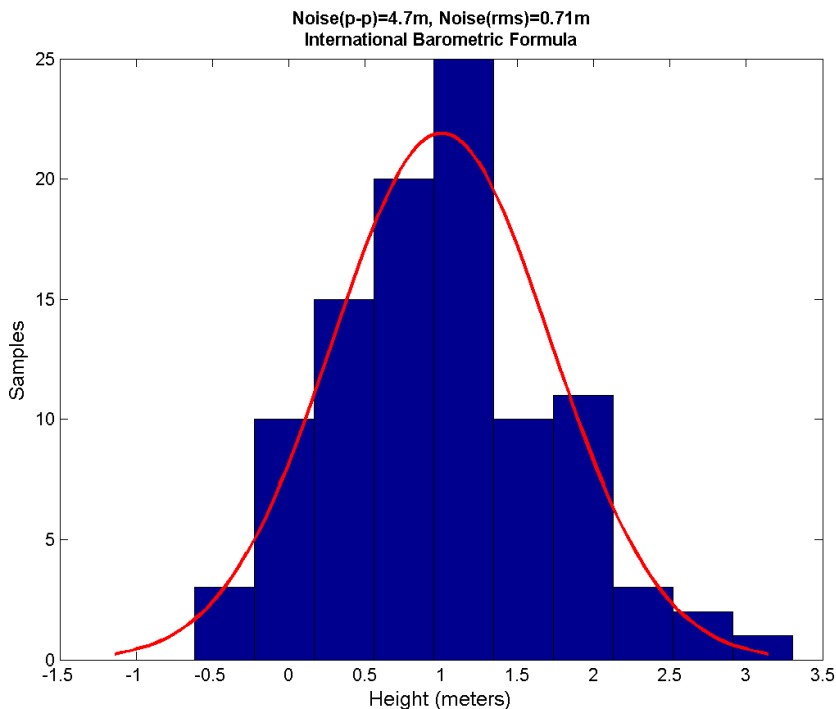
Τύπος Β—1 Διεθνής βαρομετρική εξίσωση

Οι ακόλουθες εντολές υπολογίζουν τον - γκαουσιανής κατανομής - RMS και Peak-to-Peak (P-P) θόρυβο του μετρίσιμου ύψους (γραμμές 78 και 79, αντιστοίχως) και αναπαριστούν γραφικά την κανονική κατανομή δειγμάτων του τελευταίου (γραμμές 74-77). Οι P-P και RMS και τιμές εκτυπώνονται στον τίτλο της εικόνας (γραμμή 80) και για τον υπολογισμό της P-P τιμής χρησιμοποιείται το 99,9% των δειγμάτων, όπως ορίζεται από τη σταθερά 6.6 (γραμμή 79). Η RMS τιμή του θορύβου υπολογίζεται μέσω της συνάρτησης std του Matlab (γραμμή 78), η οποία εφαρμόζει τον **Τύπο Β—2**. Τέλος, στη γραμμή 81 γίνεται η αποθήκευση της εικόνας με όνομα `.Noise-pp-and-rms-International-Barometric-Formula.tif`. Το αποτέλεσμα εκτέλεσης των εντολών δίνεται στην **Εικόνα Β—10**.

```

74 figure;
75 histfit(height);
76 xlabel('Height (meters)', 'FontSize', 12);
77 ylabel('Samples', 'FontSize', 12); %y-axis label
78 rmsHeight=std(height);
79 Height_pp=6.6*rmsHeight;
80 title(sprintf('Noise (p-p)=%.1fm,      Noise(rms)=%.2fm\nInternational
Barometric Formula', Height_pp, rmsHeight), 'FontSize', 11,
'FontWeight', 'bold');
81 print -dtiffn 6.Noise-pp-and-rms-International-Barometric-Formula.tif

```



Εικόνα Β—10 Κατανομή δειγμάτων ύψους (διεθνής βαρομετρική εξίσωση)

$$s = \sqrt{\sum_{i=1}^n (x_i - \bar{x})^2 / (n-1)}$$

Τύπος B—2 Τυπική απόκλιση

Η διαδικασία υπολογισμού του απόλυτου ύψους επαναλαμβάνεται με τη χρήση της υψομετρικής εξίσωσης (*hypsometric equation*), όπως ορίζεται από τον **Τύπο B—3**. Η εξίσωση αυτή χρησιμοποιεί και την πληροφορία της θερμοκρασίας, η οποία θεωρείται σταθερή ανάμεσα στις συσκευές μέτρησης (ισόθερμη ατμόσφαιρα). Συνεπώς, οι γραμμές 82-87 υπολογίζουν τη μέση τιμή της θερμοκρασίας σε βαθμούς Κελσίου, όπως αυτή προκύπτει από τις μετρήσεις κινητής συσκευής του συστήματος στη θέση Υ=1,1μ (γραμμή 82) και της σταθερής συσκευής στη θέση Υ=0μ (γραμμή 84). Ο υπολογισμός της ατμοσφαιρικής κλίμακας ύψους (*scale height of atmosphere*) πραγματοποιείται στις γραμμές 88-92. Στη συνέχεια ακολουθείται παρόμοια διαδικασία υπολογισμού και γραφικής αναπαράστασης του απόλυτου ύψους, όπως στο παράδειγμα υπολογισμού με τη χρήση της διεθνούς βαρομετρικής εξίσωσης, με τη διαφορά ότι στον τίτλο της εικόνας εκτυπώνεται και η πληροφορία της μέσης τιμής της θερμοκρασίας με ένα δεκαδικό ψηφίο (όπως ορίζει η οδηγία %1f στη γραμμή 105). Στη γραμμή 106 γίνεται η αποθήκευση της εικόνας με όνομα 7.Height-Graph-Hypsometric-Equation.tif. Το αποτέλεσμα εκτέλεσης των εντολών δίνεται στην **Εικόνα B—11**.

```

82 Ta2=mean(PressureDifferent(:,2))
83 Ta2=Ta2+273.15;
84 Tb=mean(PressureDifferent(:,4))
85 Tb=Tb+273.15;
86 T=(Ta2+Tb)/2
87 Tcelsius=T-273.15
88 k=1.38*power(10,-23); %Boltzmann's Constant (J/deg)
89 m=(0.22*(2*2.67*power(10,-26))) + (0.78*(2*2.3*power(10,-26))); %
Average atomic mass in kilograms
90 %The atmosphere consists of 22% of O2 (m= 2x2.67x10^-26 kg) and 78%
of N2 (m= 2x2.3x10^-26 kg)
91 g=9.81; %Acceleration of gravity (meters/sec^2)
92 H=(k*T)/(m*g);
93 if (mean(Pa2) > mean(Pb))
94 height2=H*(log(Pa2*100000)-log(Pb*100000));
95 else
96 height2=H*(log(Pb*100000)-log(Pa2*100000));
97 end
98 average_height2 = mean(height2);
99 figure
100 plot(height2)
101 legend('P(zB) - P(zA)', 'FontSize', 6);
102 xlabel('Samples', 'FontSize', 12);
103 ylabel('Height (meters)', 'FontSize', 12);
104 rmsHeight2=std(height2);
105 title(sprintf('Height Av. = %.2fm (Temp. Av. = %.1fC)\nHypsometric
Equation', average_height2, Tcelsius), 'FontSize', 12, 'FontWeight',
'bold');
106 print -dtiffn 7.Height-Graph-Hypsometric-Equation.tif

```

$$h = P_{(zA)} - P_{(zB)} =$$

$$= H \cdot \ln P_0 - H \ln P_{(zA)} - H \cdot \ln P_0 + H \ln P_{(zB)} =$$

$$= H (\ln P_{(zB)} - \ln P_{(zA)})$$

$$\text{όπου } P_{(zA)} < P_{(zB)} \text{ και } H = \frac{kT}{mg} \text{ (Ατμοσφαιρική κλίμακα ύψους)}$$

P_0 (Μέση τιμή ατμοσφαιρικής πίεσης στην επιφάνεια της θάλασσας σε Pa) = 102325 (τυπική τιμή)

k (Σταθερά Boltzmann's σε J / deg) = $1.38 \cdot 10^{-23}$

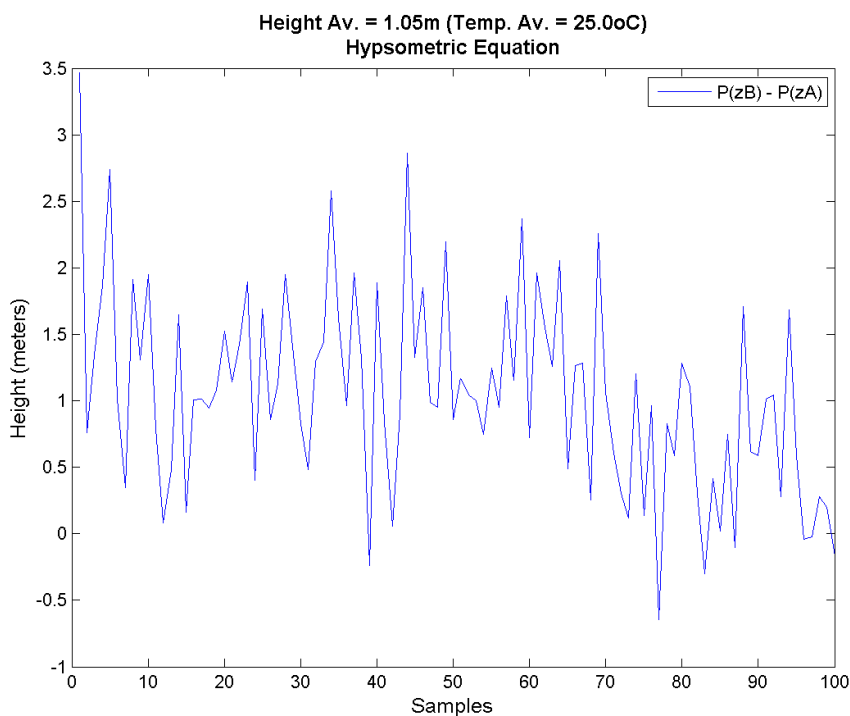
T (Μέση τιμή θερμοκρασίας του ατμοσφαιρικού στρώματος σε Kelvin) = 273K (= 0oC, τυπική τιμή)

m (Μέση τιμή μάζας των ατόμων του αέρα σε kilograms) = $4.76 \cdot 10^{-26}$

(22% O₂ και 78% N₂ $\Rightarrow 0.22 \cdot 2 \cdot 2.67 \cdot 10^{-26}$ Kg + $0.78 \cdot 2 \cdot 2.3 \cdot 10^{-26}$ Kg)

g (Επιτάχυνση βαρύτητας σε meters / sec²) = 9.81

Τύπος Β—3 Υψομετρική εξίσωση



Εικόνα Β—11 Γραφική ύψους (υπολογισμός με υψομετρική εξίσωση)

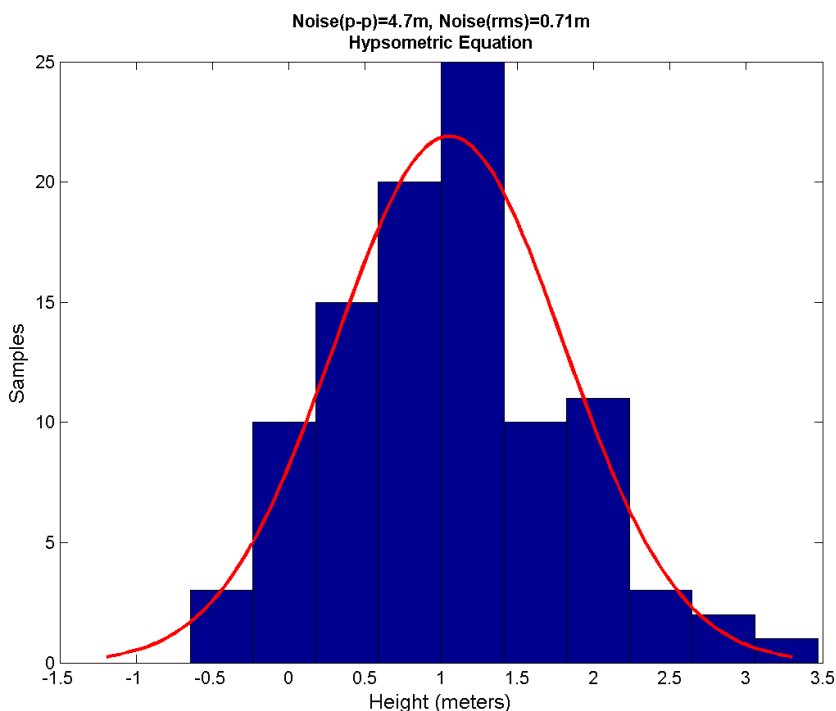
Ο υπολογισμός RMS και P-P θορύβου του μετρίσιμου ύψους, με τη χρήση της υψομετρικής εξίσωσης, επαναλαμβάνεται στις 107-112 (με τον ίδιο ακριβώς τρόπο

όπως στο παράδειγμα υπολογισμού με τη χρήση της διεθνούς βαρομετρικής εξίσωσης). Στη γραμμή 113 γίνεται η αποθήκευση της εικόνας με όνομα 8.Noise-pp-and-rms-Hypsometric-Equation.tif Το αποτέλεσμα εκτέλεσης των εντολών δίνεται στην **Εικόνα Β—12**.

```

107 figure;
108 histfit(height2);
109 xlabel('Height (meters)', 'FontSize', 12);
110 ylabel('Samples', 'FontSize', 12);
111 rmsHeight2=std(height2);
112 Height_pp2=6.6*rmsHeight2;
113 title(sprintf('Noise (p-p)=%.1fm,      Noise (rms)=%.2fm\nHypsometric
Equation', Height_pp, rmsHeight), 'FontSize', 11, 'FontWeight',
'bold');
114 print -dtiffn 8.Noise-pp-and-rms-Hypsometric-Equation.tif

```



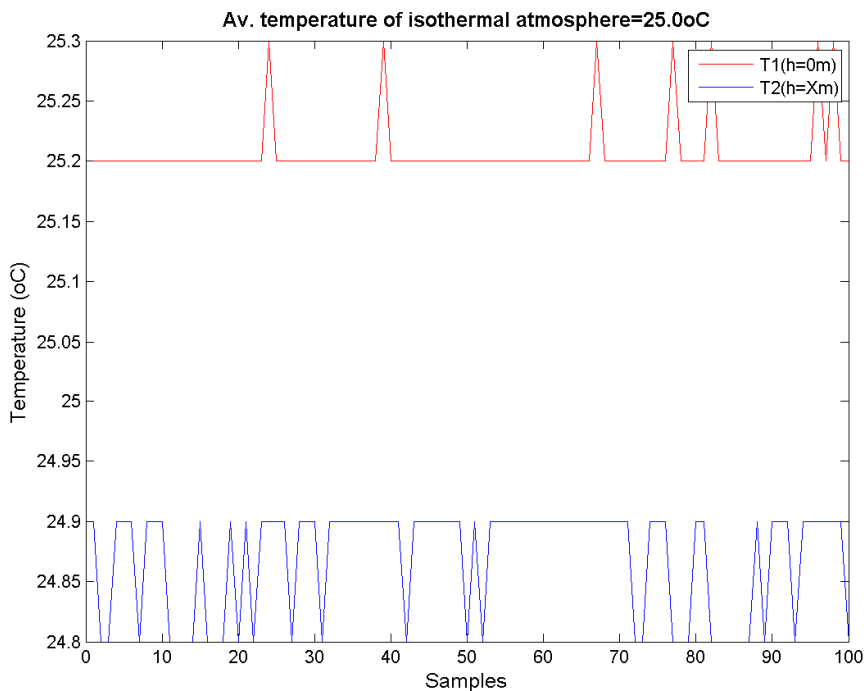
Εικόνα Β—12 Κατανομή δειγμάτων ύψους (υψομετρική εξίσωση)

Οι τελευταίες εντολές του κώδικα Matlab εκτυπώνουν τις γραφικές της μετρούμενης θερμοκρασίας, όπως λαμβάνεται από τη σταθερή συσκευή του συστήματος στη θέση $Y=0\text{m}$ (γραφική T1) και την κινητή συσκευή στη θέση $Y=1,1\text{m}$ (γραφική T2). Στη γραμμή 125 γίνεται η αποθήκευση της εικόνας με όνομα 9.Temperature-graph-of-altimeters.tif. Το αποτέλεσμα εκτέλεσης των εντολών δίνεται στην **Εικόνα Β—13**.

```

115 T1=PressureDifferent(:,2);
116 T2=PressureDifferent(:,4);
117 fig = figure;
118 plot(T1,'r');
119 hold on; %hold the previous plot and add the following to the same
graph
120 plot(T2);
121 legend('T1 (h=0m)', 'T2 (h=Xm) ');
122 xlabel('Samples', 'FontSize', 12);
123 ylabel('Temperature (oC)', 'FontSize', 12);
124 title(sprintf('Av. temperature of isothermal
atmosphere=%.1foC\n(measurements obtained from the rover altimeter)',
Tcelsius), 'FontSize', 12, 'FontWeight', 'bold');
125 print -dtiffn 9.Temperature-gpaph-of-altimeters.tif

```

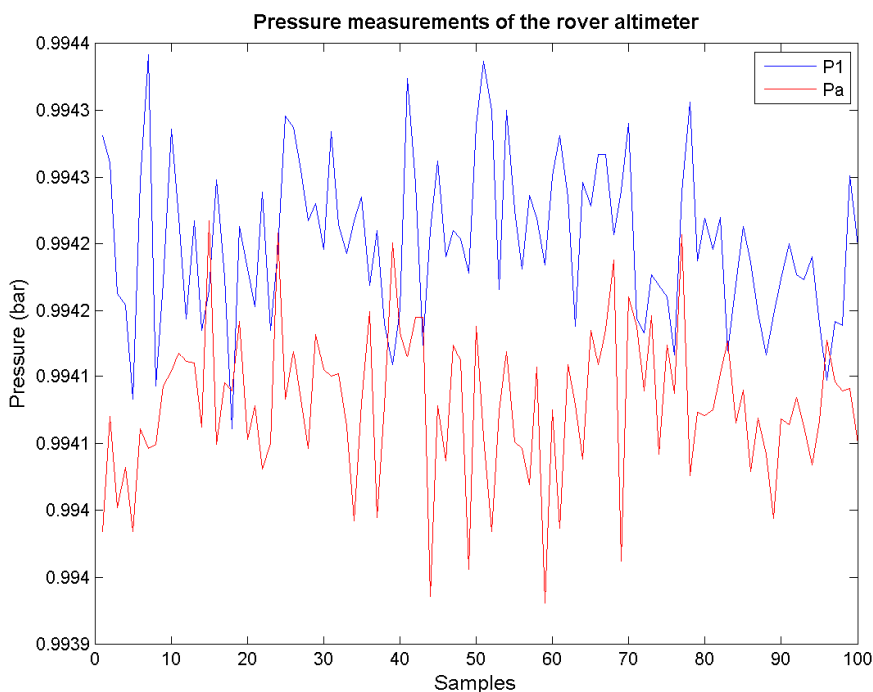


Εικόνα Β—13 Μετρήσεις θερμοκρασίας σταθερής/κινητής συσκευής (T1/T2)

Ανάλυση μετρήσεων απλής βαρομετρικής υψομετρίας

Στη συνέχεια αναλύεται το αρχείο εντολών Matlab που χρησιμοποιείται για τον υπολογισμό απόλυτου ύψους εφαρμόζοντας την τεχνική απλής βαρομετρικής υψομετρίας (*single barometric altimetry*) και φέρει το χαρακτηριστικό όνομα MatlabScript_SBA.m. Η τεχνική αυτή εφαρμόζεται στα δεδομένα που χρησιμοποιήθηκαν προωτύτερα, λαμβάνοντας υπόψη τις μετρήσεις της κινητής – αποκλειστικά – συσκευής του συστήματος. Στον κώδικα Matlab θα πρέπει να οριστεί αρχικά η σταθερά που αντιστοιχεί στην κινητή και σταθερή συσκευή του συστήματος. Η σταθερά αυτή προκύπτει από τον αριθμό στήλης των αρχείων με PressureSame.txt και PressureDifferent.txt, στην οποία εμπεριέχονται οι μετρήσεις της ατμοσφαιρικής πίεσης. Συνεπώς, οι πιθανές τιμές σταθεράς είναι 1 και 3, οι οποίες αντιστοιχούν στις συσκευές μέτρησης 1 και 2. Στις πρώτες δύο εντολές του παραδείγματος, ορίζεται ως κινητό υψόμετρο (*rover altimeter*) η συσκευή μέτρησης 1, ενώ ως σταθερό υψόμετρο (*base altimeter*) ορίζεται η συσκευή μέτρησης 2.

```
1   roverAltimeter = 1;  
2   baseAltimeter = 3;
```



Εικόνα Β—14 Μετρήσεις ατμοσφαιρικής πίεσης κινητής συσκευής (P1/Pa)

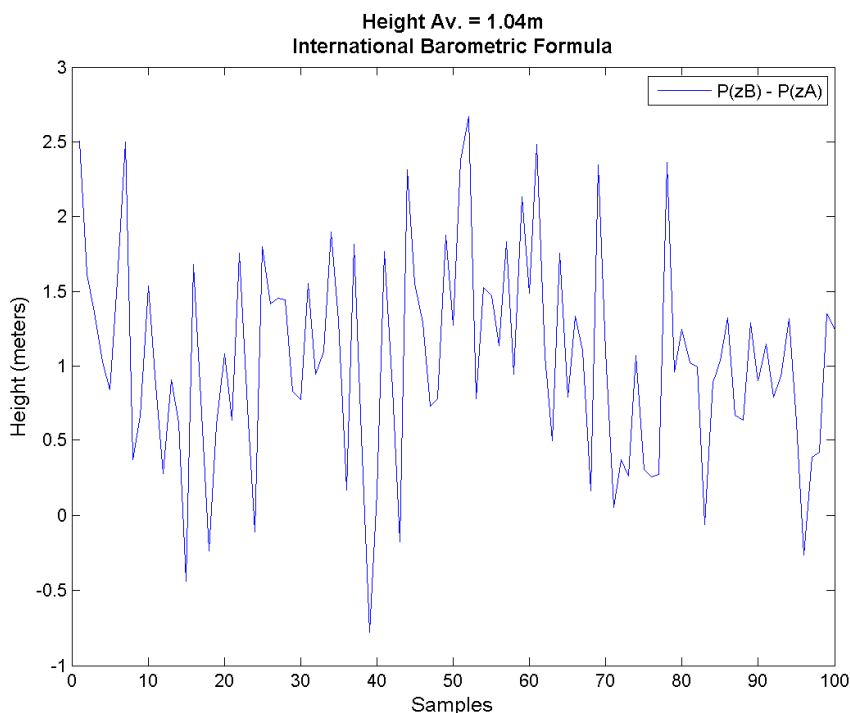
Οι ακόλουθες εντολές αναπαριστούν γραφικά τις μετρήσεις που κατέγραψε η

κινητή συσκευή του συστήματος στη θέση 0μ (αρχείο PressureSame.txt) και στη θέση 1,1μ (αρχείο PressureDifferent.txt). Η πρώτη γραφική ονομάζεται P1 και η δεύτερη Pa. Όπως είναι αναμενόμενο, η γραφική Pa είναι μικρότερη από την P1. Το αποτέλεσμα εκτέλεσης των εντολών δίνεται στην **Εικόνα Β—14**.

```

3   P1=PressureSame(:,roverAltimeter);
4   Pa=PressureDifferent(:,roverAltimeter);
5   plot(P1);
6   hold on;
7   plot(Pa,'r');
8   legend('P1', 'Pa', 'FontSize', 6);
9   xlabel('Samples', 'FontSize', 12);
10  ylabel('Pressure (bar)', 'FontSize', 12);
11  title(sprintf('Pressure measurements of the rover altimeter'),
12        'FontSize', 12, 'FontWeight', 'bold');

```



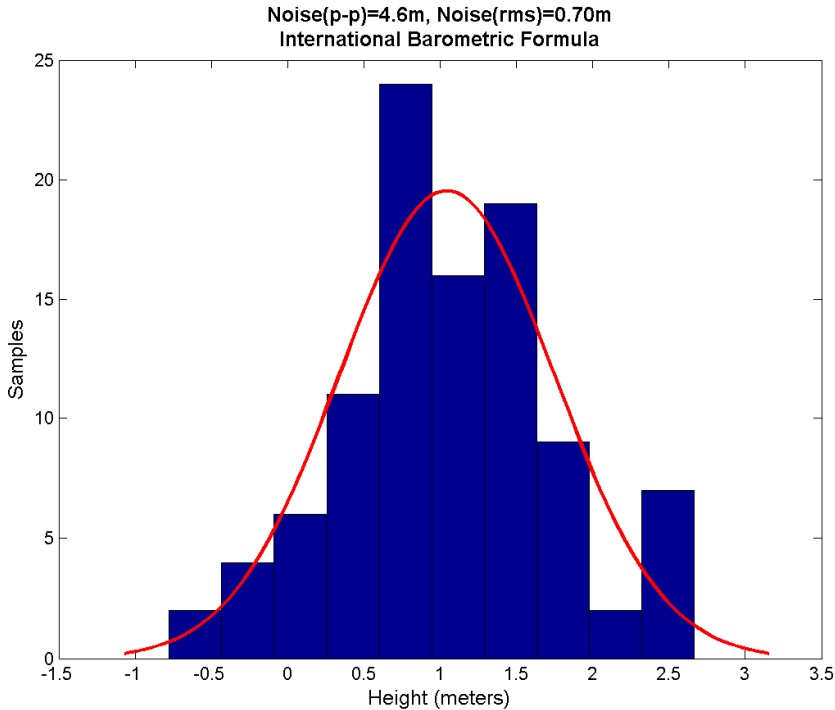
Εικόνα Β—15 Γραφική ύψους (υπολογισμός με διεθνή βαρομετρική εξίσωση)

Οι ακόλουθες εντολές υπολογίζουν την απόλυτη διαφορά ύψους μεταξύ των δύο σημείων που κατέγραψε η κινητή συσκευή του συστήματος, δηλαδή της θέσης 0μ και της θέσης 1,1μ, με τη χρήση της διεθνούς βαρομετρικής εξίσωσης. Το αποτέλεσμα εκτέλεσης των εντολών δίνεται στην **Εικόνα Β—15**.

```

13 height=4935.125 * ( power((P1*100000),0.1903) -
power((Pa*100000),0.1903) );
14 average_height = mean(height);
15 figure;
16 plot(height);
17 legend(sprintf('P(zB) - P(zA)'), 'FontSize', 6);
18 xlabel('Samples', 'FontSize', 12);
19 ylabel('Height (meters)', 'FontSize', 12);
20 title(sprintf('Height Av. = %.2fm\nInternational Barometric Formula',
average_height), 'FontSize', 12, 'FontWeight', 'bold');
21 print -dtiffn 2.Height-Graph-International-Barometric-Formula.tif

```



Εικόνα Β—16 Κατανομή δειγμάτων ύψους (διεθνής βαρομετρική εξίσωση)

Ο υπολογισμός RMS και P-P θορύβου του μετρίσιμου ύψους που εκτιμήθηκε με τη διεθνή βαρομετρική εξίσωση, καθώς και η γραφική αναπαράσταση της (γκαουσιανής) κατανομής δειγμάτων του ύψους, πραγματοποιείται στις ακόλουθες εντολές. Το αποτέλεσμα εκτέλεσης των εντολών δίνεται στην **Εικόνα Β—16**.

```

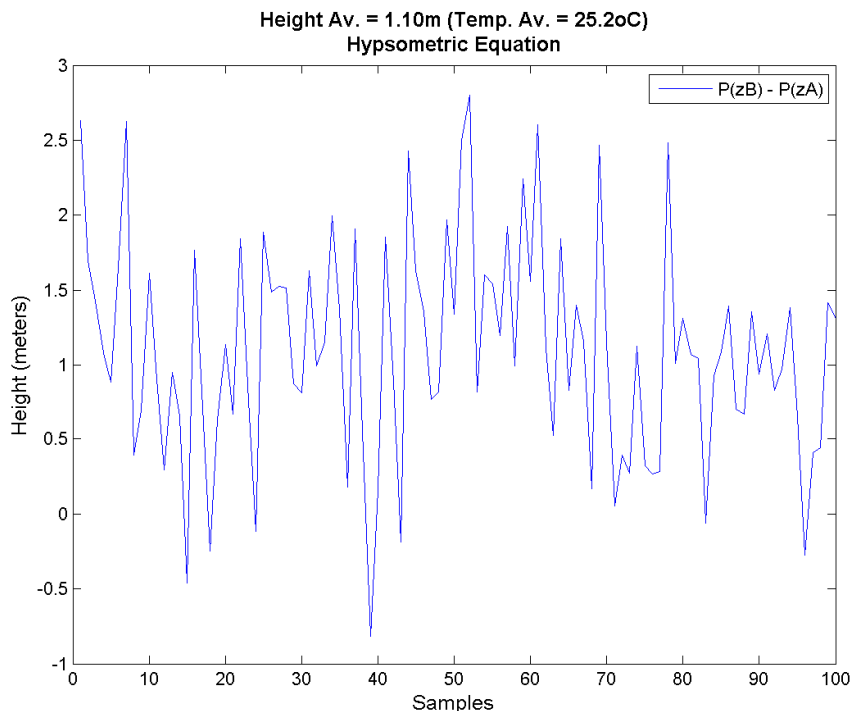
22 figure;
23 histfit(height);
24 rmsHeight=std(height);
25 Height_pp=6.6*rmsHeight;
26 %legend(sprintf('RMS      Noise=%.2fm\nP-P      Noise=%.1fm',      rmsHeight,
Height_pp));
27 xlabel('Height (meters)', 'FontSize', 12);
28 ylabel('Samples', 'FontSize', 12);

```

```

29 title(sprintf('Noise (p-p)=%.1fm,      Noise (rms)=%.2fm\nInternational
    Barometric Formula', Height_pp, rmsHeight), 'FontSize', 12,
    'FontWeight', 'bold');
30 print -dtiffn 3.Noise-pp-and-rms-International-Barometric-Formula.tif

```



Εικόνα Β—17 Γραφική ύψους (υπολογισμός με υψομετρική εξίσωση)

Οι ακόλουθες εντολές υπολογίζουν την απόλυτη διαφορά ύψους μεταξύ των δύο σημείων που κατέγραψε η κινητή συσκευή του συστήματος (0μ και 1,1μ), χρησιμοποιώντας την υψομετρική εξίσωση αυτή τη φορά. Θα πρέπει να σημειωθεί ότι, η μέση τιμή της θερμοκρασίας σε βαθμούς κελσίου προκύπτει από τις μετρήσεις της κινητής συσκευής στη θέση $Y=0\mu$ και $Y=1,1\mu$ (γραμμές 31-36). Οι τιμές θερμοκρασίας λαμβάνονται, αντιστοίχως, από τη δεύτερη στήλη του αρχείου PressureSame.txt (γραμμή 31) και τη δεύτερη στήλη του αρχείου PressureDifferent.txt (γραμμή 33). Το αποτέλεσμα εκτέλεσης των εντολών δίνεται στην **Εικόνα Β—17**.

```

31 T1=mean(PressureSame(:,roverAltimeter+1));
32 T1=T1+273.15;
33 Ta=mean(PressureDifferent(:,roverAltimeter+1));
34 Ta=Ta+273.15;
35 T=(T1+Ta)/2;
36 Tcelsius=T-273.15;
37 k=1.38*power(10,-23);
38 m=(0.22*(2*2.67*power(10,-26)))+(0.78*(2*2.3*power(10,-26)));
39 g=9.81;

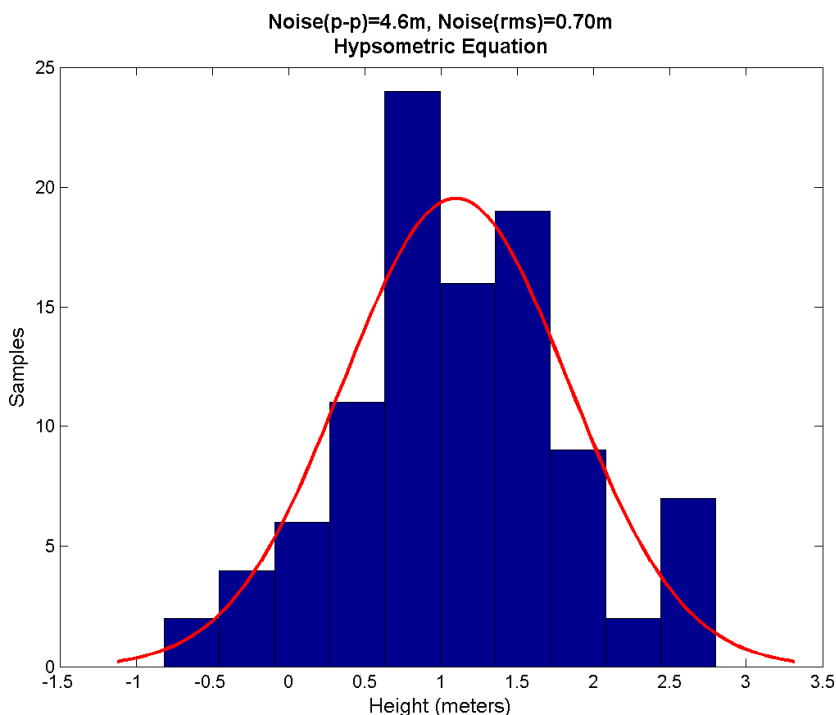
```



```

40 H=(k*T)/(m*g);
41 height2=H*(log(P1*100000)-log(Pa*100000));
42 average_height2 = mean(height2);
43 figure;
44 plot(height2);
45 legend('P(zB) - P(zA)', 'FontSize', 6);
46 xlabel('Samples', 'FontSize', 12);
47 ylabel('Height (meters)', 'FontSize', 12);
48 rmsHeight2=std(height2);
49 title(sprintf('Height Av. = %.2fm (Temp. Av. = %.1foC)\nHypsometric
Equation', average_height2, Tcelsius), 'FontSize', 12, 'FontWeight',
'bold');
50 print -dtiffn 4.Height-Graph-Hypsometric-Equation.tif

```



Εικόνα Β—18 Κατανομή δειγμάτων ύψους (υψομετρική εξίσωση)

Ο υπολογισμός RMS και P-P θορύβου του μετρίσιμου ύψους που εκτιμήθηκε με την υψομετρική εξίσωσης, καθώς και η γραφική αναπαράσταση της (γκαουσιανής) κατανομής δειγμάτων του ύψους, πραγματοποιείται στις ακόλουθες εντολές. Το αποτέλεσμα εκτέλεσης των εντολών δίνεται στην **Εικόνα Β—18**.

```

51 figure;
52 histfit(height2);
53 xlabel('Height (meters)', 'FontSize', 12);
54 ylabel('Samples', 'FontSize', 12);
55 rmsHeight2=std(height2);
56 Height_pp2=6.6*rmsHeight2;

```

```

57 title(sprintf('Noise (p-p)=%.1fm,          Noise(rms)=%.2fm\nHypsometric
Equation', Height_pp, rmsHeight), 'FontSize', 12, 'FontWeight',
'bold');
58 print -dtiffn 5.Noise-pp-and-rms-Hypsometric-Equation.tif

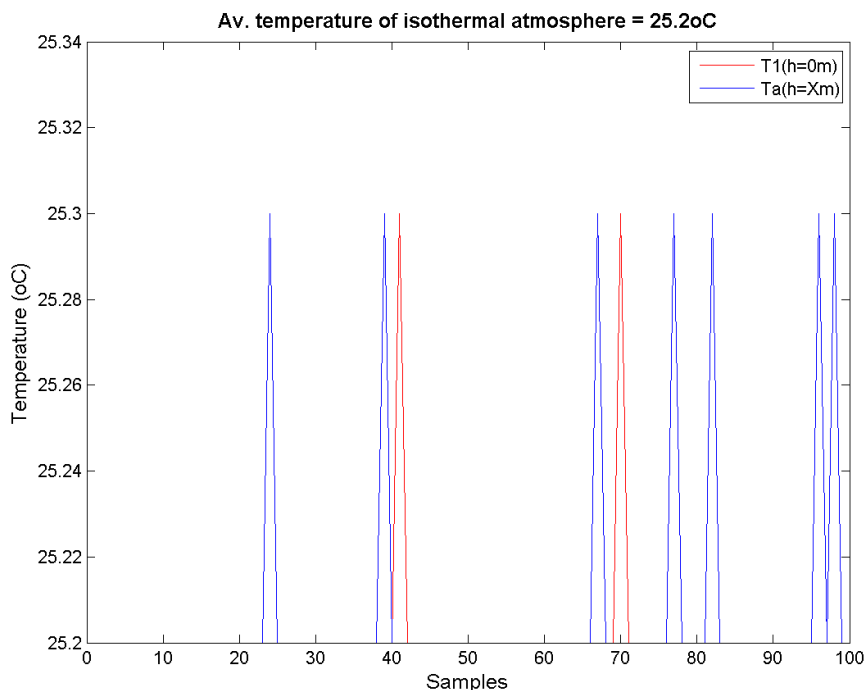
```

Οι ακόλουθες εντολές του κώδικα Matlab εκτυπώνουν τις γραφικές της μετρούμενης θερμοκρασίας, όπως λαμβάνεται από την κινητή συσκευή του συστήματος στη θέση $\Upsilon=0\mu$ (γραφική T1) και στη θέση $\Upsilon=1,1\mu$ (γραφική Ta). Το αποτέλεσμα εκτέλεσης των εντολών δίνεται στην **Εικόνα Β—19**.

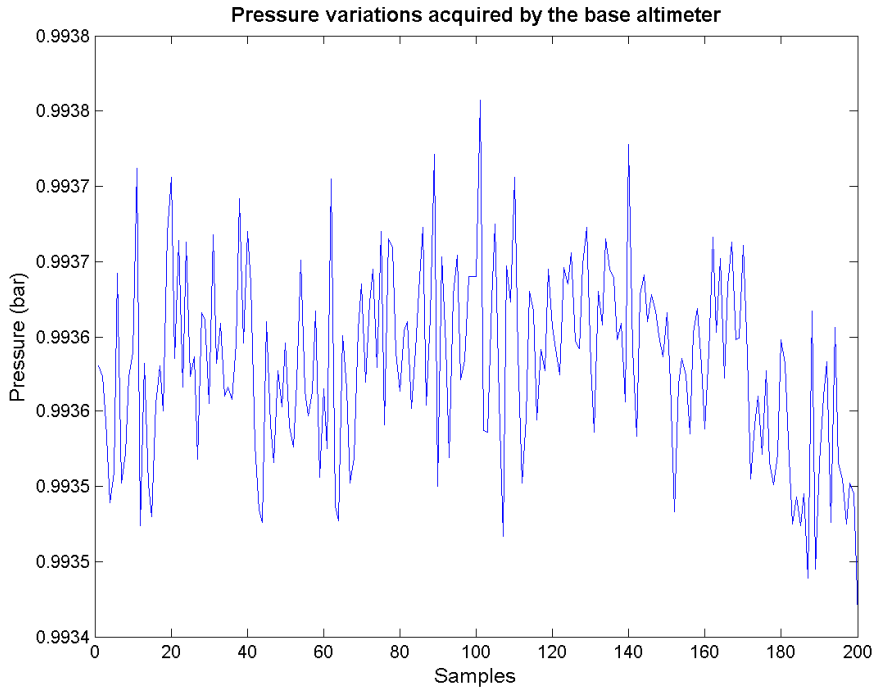
```

59 T1=PressureSame(:,roverAltimeter+1);
60 Ta=PressureDifferent(:,roverAltimeter+1);
61 fig = figure;
62 plot(T1,'r');
63 hold on; %hold the previous plot and add the following to the same
graph
64 plot(Ta);
65 legend('T1 (h=0m)', 'Ta (h=Xm) ');
66 xlabel('Samples', 'FontSize', 12);
67 ylabel('Temperature (oC)', 'FontSize', 12);
68 title(sprintf('Av. temperature of isothermal atmosphere = %.1foC',
Tcelsius), 'FontSize', 12, 'FontWeight', 'bold');
69 print -dtiffn 6.Temperature-gpaph-of-rover-altimeter.tif

```



Εικόνα Β—19 Μετρήσεις θερμοκρασίας κινητής συσκευής στα 0μ/1,1μ (T1/Ta)



Εικόνα Β—20 Μεταβολές ατμοσφαιρικής πίεσης

Οι τελευταίες εντολές του κώδικα Matlab αναδεικνύουν τις μεταβολές ατμοσφαιρικής πίεσης καθ' όλη τη διάρκεια διεξαγωγής του πειράματος. Η διενέργεια αυτή επιτυγχάνεται με τις μετρήσεις που κατέγραψε η σταθερή συσκευή του συστήματος. Στις γραμμές 71 και 72 λαμβάνονται οι μετρήσεις ατμοσφαιρικής πίεσης που κατέγραψε η σταθερής συσκευή (P2 και Pb), κατά τη διάρκεια που η κινητή συσκευή βρίσκεται στη θέση 0μ και στη θέση 1,1μ, αντιστοίχως. Στη γραμμή 73 γίνεται συγχώνευση των (μονοδιάστατων) πινάκων P2 και Pb στο μονοδιάστατο πίνακα P. Κατόπιν εκτυπώνεται η γραφική αναπαράσταση του πίνακα P (γραμμές 74-77). Το αποτέλεσμα εκτέλεσης των εντολών δίνεται στην **Εικόνα Β—20**.

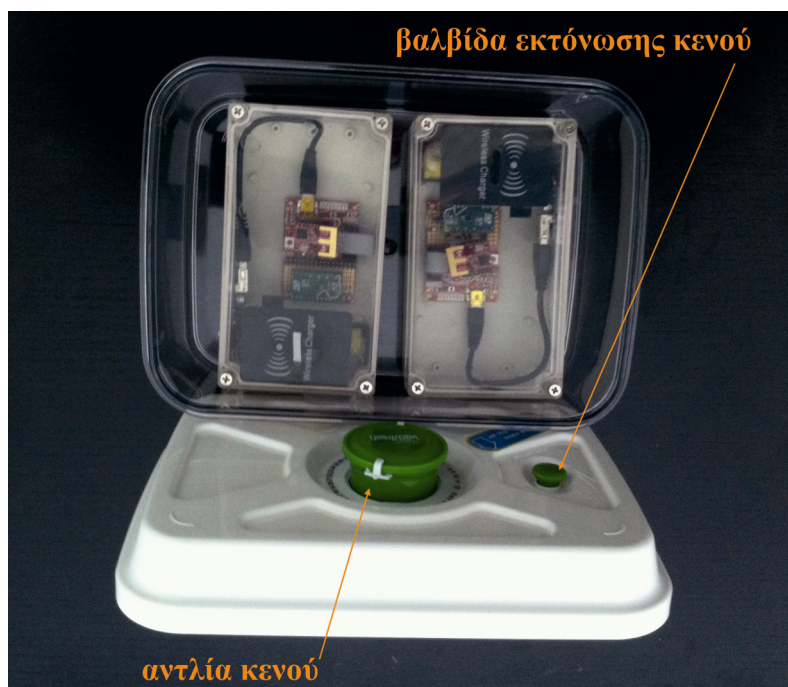
```

70 figure;
71 P2=PressureSame(:,baseAltimeter);
72 Pb=PressureDifferent(:,baseAltimeter);
73 P=cat(1,P2,Pb);
74 plot(P);
75 xlabel('Samples', 'FontSize', 12);
76 ylabel('Pressure (bar)', 'FontSize', 12);
77 title(sprintf('Pressure variations acquired by the base altimeter'),
       'FontSize', 12, 'FontWeight', 'bold');
78 print -dtiffn 7.Pressure-variations-determined-by-the-base-
    altimeter.tif

```

Μετρήσεις σε αεροστεγές δοχείο

Για την ανάλυση των δεδομένων που λαμβάνονται σε αεροστεγές δοχείο (Εικόνα Β—21) και τη μελέτη της απόκλισης στο σήμα εξόδου δύο αισθητήρων που μετρούν την ίδια τιμή πίεση κάτω από τις ίδιες συνθήκες (θερμοκρασία, υγρασία, κλπ.), ακολουθείται η εξής διαδικασία:

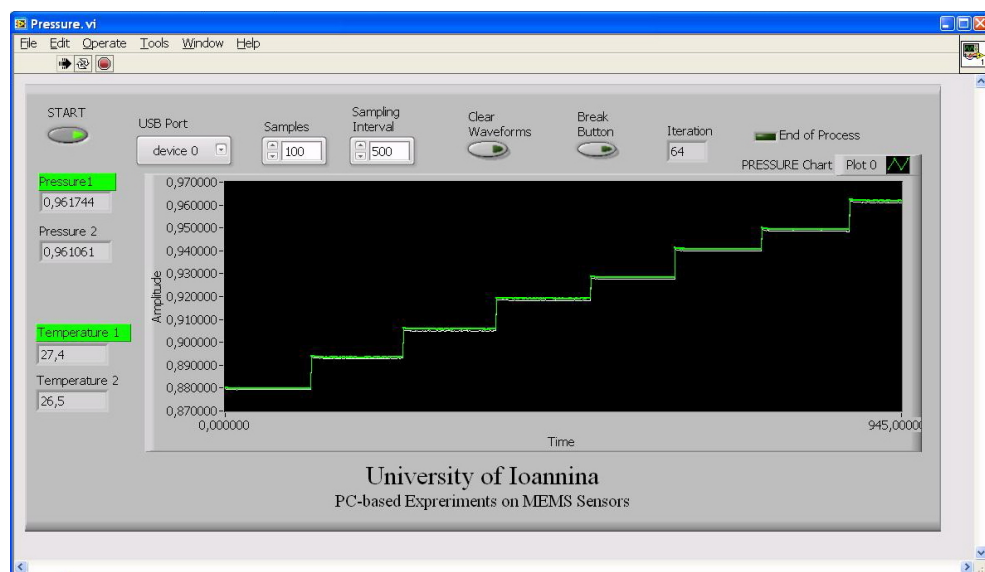


Εικόνα Β—21 Καταγραφή μετρήσεων σε αεροστεγές δοχείο

1. Τοποθετούνται οι δύο συσκευές μέτρησης στο αεροστεγές δοχείο και ενεργοποιείται η διαδικασία λήψης μετρήσεων μέσω του προγράμματος RemoteLab.exe.
2. Γίνεται αφαίρεση του αέρα από το δοχείο πιέζοντας επαναλαμβανόμενα την αντλία κενού (*vacuum pump*), έως ότου προσεγγίσουμε το κατώτατο επιθυμητό κατώφλι πίεσης. Σημειώνεται ότι, το όριο λειτουργίας του αισθητήρα των συσκευών είναι 260-1260 mbar [2].
3. Τερματίζεται η λειτουργία του προγράμματος RemoteLab.exe και διαγράφεται το αρχείο Pressure.txt από τον τρέχοντα φάκελο, ώστε να γίνει επανεκκίνηση της διαδικασίας λήψης μετρήσεων.
4. Πραγματοποιείται έναρξη καταγραφής μετρήσεων και λαμβάνεται το 1^ο αρχείο μετρήσεων, που φέρει όνομα Pressure.txt.

5. Πραγματοποιείται μετονομασία του αρχείου σε Pressure1.txt.
6. Αφαιρείται μικρή ποσότητα αέρα από το δοχείο μέσω της βαλβίδας εκτόνωσης κενού (*vacuum relief valve*) και επαναλαμβάνεται η διαδικασία καταγραφής μετρήσεων.
7. Πραγματοποιείται μετονομασία του νέου αρχείου, αυξάνοντας μοναδιαία τον αριθμό του.
8. Επαναλαμβάνονται τα βήματα 6 και 7 για όσα κατώφλια πίεσης επιθυμούμε. Στο τέλος τη διαδικασίας και μετά από N βήματα, προκύπτουν τα αρχεία Pressure1.txt, Pressure2.txt, ..., PressureN.txt. Τα αρχεία των μετρήσεων θα πρέπει να φέρουν, υποχρεωτικά, την ακριβή αυτή ονομασία.

Στην **Εικόνα Β—22** δίνεται ένα παράδειγμα καταγραφής μετρήσεων σε αεροστεγές δοχείο με το πρόγραμμα Labview.



Εικόνα Β—22 Μορφή καταγραφής μετρήσεων σε αεροστεγές δοχείο

Ανάλυση μετρήσεων σε αεροστεγές δοχείο

Στη συνέχεια αναλύεται το αρχείο εντολών Matlab που χρησιμοποιείται για τη μελέτη της απόκλισης στο σήμα εξόδου των αισθητήρων, το οποίο φέρει το χαρακτηριστικό όνομα MatlabScript_DEVIATION.m.

Στη γραμμή 1 ορίζεται η διαδρομή του δίσκου στην οποία καταχωρούνται τα αρχεία μετρήσεων. Στη γραμμή 2 καταχωρείται στη μεταβλητή BaseName το κοινό τμήμα του ονόματος των αρχείων, δηλαδή η λέξη Pressure χωρίς τον αριθμό που εντοπίζεται στο τέλος κάθε ονόματος. Στη γραμμή 3 ανιχνεύεται, και καταχωρείται στη μεταβλητή files, ο αριθμός των .txt αρχείων που εντοπίζονται στον τρέχοντα φάκελο. Για το λόγο αυτό, είναι σημαντικό να υπάρχουν άλλα .txt αρχεία στο συγκεκριμένο φάκελο. Στις γραμμές 4-7 υλοποιείται βρόχος επαναλήψεων που φορτώνει αυτόματα όλα τα .txt αρχεία στο παράθυρο Workspace του Matlab. Το όνομα κάθε αρχείου αλλάζει δυναμικά στη γραμμή 5, λαμβάνοντας τον αριθμό που υπάρχει στο τέλος του ονόματος μέσω της οδηγίας %d και της τιμής του μετρητή i του βρόχου.

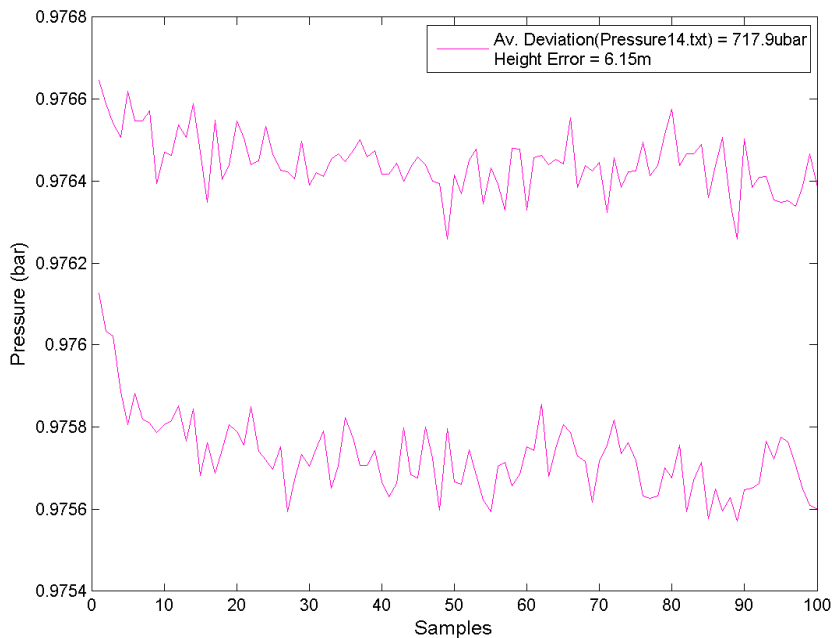
```
1 path='C:\Measurements';
2 BaseName='Pressure';
3 files = dir('*.txt');
4 for i=1:length(files)
5     currentFile = sprintf('%s%d.txt',BaseName,i);
6     load(currentFile);
7     end
```

Στις γραμμές 8-32 εκτυπώνονται σε διαφορετικές εικόνες τα ζεύγη μετρήσεων του κάθε ένα αρχείου. Αναλυτικά, στη γραμμή 8 δημιουργείται ένας χάρτης χρωμάτων (μέσω της συνάρτησης hsv) που αποτελείται από τόσα χρώματα, όσα και τα αρχεία μετρήσεων. Η εντολή αυτή χρησιμοποιείται ώστε κάθε ζεύγος γραφικών να αναπαρίσταται με διαφορετικό χρώμα. Μέσα σε ένα βρόχο επαναλήψεων (γραμμές 9-32), ανοίγει αρχικά ένα παράθυρο εικόνας (γραμμή 10). Στη συνέχεια φορτώνεται στη μεταβλητή FileName το όνομα του αρχείου, λαμβάνοντας το κοινό τμήμα ονόματος από τη μεταβλητή BaseName και τον αριθμό που περιγράφει το κάθε αρχείο από το μετρητή i του βρόχου (γραμμή 11). Τα δεδομένα του εκάστοτε αρχείου καταχωρούνται στον πίνακα P (γραμμή 12). Στις γραμμές 13-29 εκτυπώνεται στο τρέχον παράθυρο εικόνας το ζεύγος των μετρούμενων πιέσεων (P1, P2), ενώ ως λεζάντα εικόνας ορίζεται η μέση απόκλιση μεταξύ των πιέσεων των δύο αισθητήρων, καθώς και η εκτιμώμενη απόκλιση απόλυτου ύψους που οφείλεται στη διαφορά αυτή. Η απόκλιση απόλυτου ύψους εκτιμάται με τη διεθνή βαρομετρική εξίσωση. Στις γραμμές 30 και 31 αποθηκεύεται η εκάστοτε εικόνα στον τρέχοντα φάκελο, με όνομα ίδιο με αυτό του αρχείου δεδομένων και σε μορφή .tif (χωρίς συμπίεση όπως ορίζεται από την οδηγία tiffn). Στην **Εικόνα Β—23** δίνεται ένα παραδείγματα γραφικής που προκύπτει από την εκτέλεση του κώδικα αυτού.

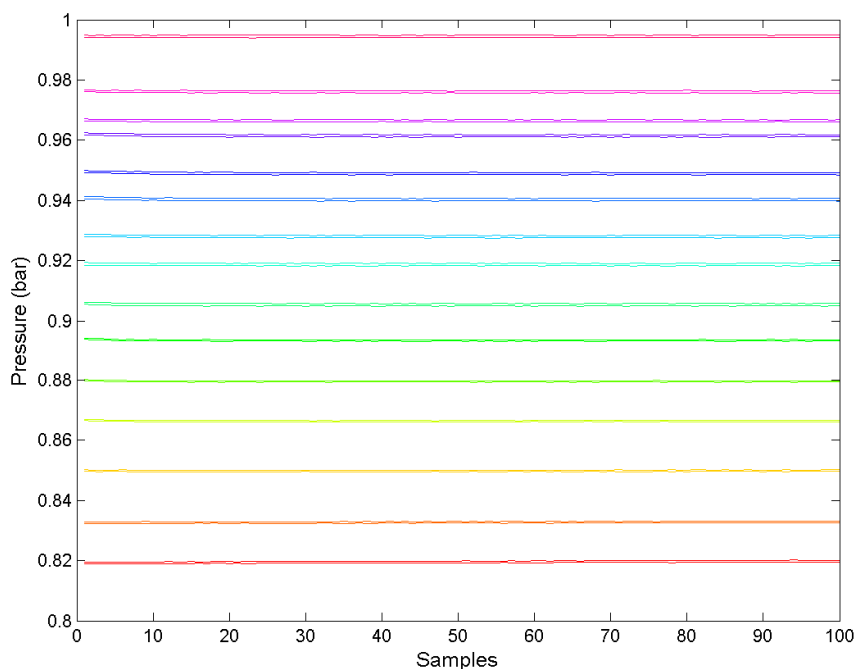
```

8     colormap=hsv(length(files));
9     for i = 1:length(files)
10      fig = figure;
11      FileName=[BaseName,num2str(i),'.txt']
12      P=dlmread(FileName);
13      P1=P(:,1);
14      P2=P(:,3);
15      Pdif = abs(P1-P2);
16      PdifAv=mean(Pdif);
17      if (mean(P1) > mean(P2))
18          height=4935.125 * ( power((P1*100000),0.1903) -
19                               power((P2*100000),0.1903) );
19      else
20          height=4935.125 * ( power((P2*100000),0.1903) -
21                               power((P1*100000),0.1903) );
22      end
23      average_height = mean(height);
24      plot(P1,'color',colormap(i,:));
25      hold on;
26      plot(P2,'color',colormap(i,:));
27      hold on;
28      legend(sprintf('Av. Deviation(%s) = %.1fubar\nHeight Error = %.2fm',
29                    FileName, PdifAv*1000000, average_height));
30      xlabel('Samples', 'FontSize', 12);
31      ylabel('Pressure (bar)', 'FontSize', 12);
32      currentFile = sprintf('%s%d.tif',BaseName,i);
33      print ('-dtiffn', currentFile);
34      end

```



Εικόνα Β—23 Γραφική ενός αρχείου μετρήσεων (αεροστεγές δοχείο)



Εικόνα Β—24 Γραφική όλων των αρχείων μετρήσεων (αεροστεγές δοχείο)

Ο ίδιος κώδικας επαναλαμβάνεται στις γραμμές 33-55, με τη διαφορά ότι το παράθυρο εικόνας δημιουργείται πριν το βρόχο (γραμμή 33) ώστε να γίνει εκτύπωση όλων των γραφικών στην ίδια εικόνα. Η αποθήκευση της εικόνας αυτής γίνεται πάλι εκτός βρόχου (γραμμές 54 και 55). Στην **Εικόνα Β—24** δίνεται ένα παραδείγματα εκτέλεσης του κώδικα αυτού, που προκύπτει από την ανάλυση 15 διαφορετικών αρχείων (Pressure1.txt, Pressure2.txt, ..., Pressure15.txt).

```

33  fig = figure;
34  for i = 1:length(files)
35      FileName=[BaseName,num2str(i),'.txt']
36      P=dlmread(FileName);
37      P1=P(:,1);
38      P2=P(:,3);
39      Pdif = abs(P1-P2);
40      PdifAv=mean(Pdif);
41      if (mean(P1) > mean(P2))
42          height=4935.125 * ( power((P1*100000),0.1903) -
43          power((P2*100000),0.1903) );
44      else
45          height=4935.125 * ( power((P2*100000),0.1903) -
46          power((P1*100000),0.1903) );
47      end
48      average_height = mean(height);
49      plot(P1,'color',colormap(i,:));

```



```
48     hold on;
49     plot(P2,'color',colormap(i,:));
50     hold on;
51     xlabel('Samples', 'FontSize', 12);
52     ylabel('Pressure (bar)', 'FontSize', 12);
53     end
54     currentFile = sprintf('%s-all.tif',BaseName);
55     print ('-dtiffn', currentFile);
```

Αναφορές

- [1] BMP180 Digital pressure sensor, Bosch Sensortec, 2013.
- [2] LPS331AP MEMS pressure sensor, STMicroelectronics, 2012.

Κώδικας



ΣΤΟ ΠΑΡΑΡΤΗΜΑ ΑΥΤΟ

ΥΛΙΚΟΛΟΓΙΣΜΙΚΟ ΣΥΣΤΗΜΑΤΟΣ

Δίδεται ο κώδικας του υλικολογισμικού μέρους του συστήματος.

ΛΟΓΙΣΜΙΚΟ ΣΥΣΤΗΜΑΤΟΣ

Δίδεται ο κώδικας του λογισμικού μέρους του συστήματος.

MATLAB ΚΩΔΙΚΑΣ

Δίδεται ο κώδικας ανάλυσης δεδομένων στο Matlab.

Υλικολογισμικό συστήματος

1. Συντονιστής συστήματος

Coordinator.c

```

1  #include <Coordinator.h>
2  #include <USB_bootloader.h>
3  #include "my_pic18_usb.h"           // Microchip PIC18Fxx5x hardware
   layer for usb.c
4  #include "my_usb_desc_bulk.h"       // USB Configuration and Device
   descriptors for this UBS device
5  #include "my_usb.c"                 // Handles usb setup tokens and get
   descriptor reports
6  #include <string.h>
7  #include <W5100.h>
8  #include <MRF24J40.h>
9
10 // Define number of End Point devices and length of Measurement data
11 const int8 No_ofDevices = 2;
12 const int8 singleDevice_length = 16;
13 // Preasure data variables
14 const int8 allDevices_length = No_ofDevices*singleDevice_length;
15 char singleDevice_data[singleDevice_length]; // Buffer for data
   storage of every single EP device measurements
16 char allDevices_data[allDevices_length]; // Buffer for data storage of
   every single EP device measurements
17 void read_measurements (int8 delay_for_each_device);
18
19 int main(void)
20 {
21
22
23     // W5100 variables
24     int8 sockstat;
25     int16 rsize;
26     int HTTP_request;
27     // MCU variables
28     RBPU = 0; // Enable port_b_pullups when defined inputs (PB4 is
   defined as EXT INT3)
29
30     // W5100 Ethernet IC Initialization
31     W5100_Init();
32
33     // MRF24J40 IEEE 802.15.4 radio Initialization
34     output_high(MRF24J40_CS);
35     output_high(MRF24J40_RST);
36     MRF24J40_init();
37
38     // USB Bulk Transfer initialization
39     char usb_buffer[7];
40     usb_init();
41
42
43     // Loop forever
44     for(;;)
45     {

```

```

46      //***** USB interface *****//
47      if(usb_enumerated())
48      {
49          LED_ON(LED2);
50          if (usb_kbhit(1))
51          {
52              usb_gets(1,usb_buffer,8,1); // Get PC command from USB
53              read_measurements (200); // Delay needed for the transaction
of each EP device
54              usb_puts(1, allDevices_data, allDevices_length, 0);
55          }
56      }
57      //***** Eth interface *****//
58      else
59      {
60          output_high(PIN_A3); // status led 1 (webserver receiving
request) deactivated
61          output_high(PIN_C2); // status led 2 (webserver receiving
request) deactivated
62          sockstat=W5100_spiRD(S0_SR);
63          switch(sockstat)
64          {
65              case SOCK_CLOSED:
66                  if (socket(sockreg,MR_TCP,TCP_PORT) > 0)
67                  {
68                      // Listen to Socket 0
69                      if (listen(sockreg) <= 0)
70                          delay_ms(1);
71                  }
72                  break;
73              case SOCK_ESTABLISHED:
74                  // Get the client request size
75                  rsize=recv_size();
76                  if (rsize > 0)
77                  {
78                      // Now read the client Request
79                      if (recv(sockreg,buf,rsize) <= 0) break;
80                      // Check the Request Header
81                      HTTP_request = strncmp (buf, "GET /", 5);
82                      if (HTTP_request == 0)
83                      {
84                          read_measurements (200);
85                          // Create the HTTP Response Header
86                          strcpy((char *)buf,"<h1>University of
Ioannina</h1>\r\n");
87                          strcat((char *)buf,"<h3>Ethernet-based Experiments on
MEMS Sensors</h3>\r\n");
88                          strcat((char *)buf,"<p><form method=\"POST\">\r\n");
89                          if (send(sockreg,buf,strlen((char *)buf)) <= 0) break;
90                          strcpy((char *)buf, "<strong>P(bar) / T(oC): <input
type=\"text\" size=60 value=\" \" ); //the last <SPACE> char is entered
on the left of the box
91                          //strcpy((char *)buf, "<strong>P(bar) / T(oC): <input
type=\"text\" style=\"height:100px\" value=\" \" );
92                          strcat((char *)buf,allDevices_data);
93                          strcat((char *)buf,"> \r\n");
94                          // Now Send the remaining HTTP Response
95                          if (send(sockreg,buf,strlen((char *)buf)) <= 0) break;

```

```

96         }
97         // Disconnect the socket
98         disconnect(sockreg);
99     }
100     else
101     {
102         delay_us(10);    // Wait for request
103     }
104     break;
105     case SOCK_FIN_WAIT:
106     case SOCK_CLOSING:
107     case SOCK_TIME_WAIT:
108     case SOCK_CLOSE_WAIT:
109     case SOCK_LAST_ACK:
110         // Force to close the socket
111         close(sockreg);
112         break;
113     }
114 }
115 }
116 return 0;
117 }
118
119 //***** Routine for reading data from all EP devices *****/
120 void read_measurements (int8 delay_for_each_device)
121 {
122     int8 i,j;
123     int16 timeout = 0;
124     for(j=0; j<No_ofDevices; ++j)
125     {
126         DATA_TX[0] = j+1; //Address 0x01 is also CONVERT command to
all End Devices
127         write_TX_normal_FIFO();
128         delay_ms(delay_for_each_device);
129         while (input(MRF24J40_INT))
130         {
131             delay_ms(1);
132             ++timeout;
133             if (timeout > 40)    //stop waiting if MRF24J40
not responding after 3sec
134             {
135                 break;
136             }
137         }
138         if (timeout > 40)
139         {
140             continue;
141         }
142         read_ZIGBEE_short(INTSTAT); // Read and flush register
INTSTAT
143         read_RX_FIFO();           // Read receive data
144         timeout = 0;
145         sprintf((char *)singleDevice_data,"%s",DATA_RX); //
e.g. 0+"0.998415      25.4"+'\0' (16 bytes), first byte is the
Coordinator address (DATA_RX[0])
146
147         for (i=0; i<singleDevice_length-1; ++i)
148         {

```

```
149     allDevices_data[i+(j*singleDevice_length)]=singleDevice_data[i];
150     }
151     if (j>0)
152     {
153         allDevices_data[(j*singleDevice_length)-1]=',';
154     }
155     }
156     allDevices_data[(No_ofDevices*singleDevice_length)-
157     1]='\0';
158 }
```

Coordinator.h

```

158 #include <18F26J50.h>
159 #device PASS_STRINGS = IN_RAM //allow pointers to constants; forces
    the compiler to copy the constant strings into an internal RAM buffer
160 #device ADC=16
161
162 #FUSES NOWDT //No Watch Dog Timer
163 #FUSES WDT128 //Watch Dog Timer uses 1:128 Postscale
164 #FUSES NOXINST //Extended set extension and Indexed
    Addressing mode disabled (Legacy mode)
165 #FUSES NOT1DIG //Secondary Oscillator Source may be
    select if T1CON.3 = 1
166 #FUSES NOFCMEN //Fail-safe clock monitor disabled
167 #FUSES NOIESO //Internal External Switch Over mode
    disabled
168 #FUSES RTCOSC_INT //RTCC uses Internal 31KHz Oscillator
    as reference source
169
170 #use delay(internal=8000000,USB_FULL)
171
172
173 // Pin & SPI clock definition for W5100 & MRF24J40
174 #define W5100_CS PIN_A0
175 #define W5100_MOSI PIN_C1
176 #define W5100_MISO PIN_B5
177 #define W5100_SCLK PIN_C0
178 #define MRF24J40_CS PIN_A5
179 #define MRF24J40_MOSI PIN_B3
180 #define MRF24J40_MISO PIN_B2
181 #define MRF24J40_SCLK PIN_B1
182 #define MRF24J40_WAKE PIN_C7 //RXD on OLIMEX board
183 #define MRF24J40_RST PIN_C6 //TXD on OLIMEX board
184 #define MRF24J40_INT PIN_B4 //SCL on OLIMEX board
185 #define W5100_HalfPeriod 1
186 #define MRF24J40_HalfPeriod 1
187
188 // Global variables/constants for W5100
189 unsigned char mac_addr[] = {0x00,0xE0,0xA0,0x00,0x9E,0x9F}; // MAC
    address from old PC
190 unsigned char ip_addr[] = {192,168,1,177}; //{178,128,27,56}; //{
    192,168,1,177}; //
191 unsigned char sub_mask[] = {255,255,255,0};
192 unsigned char gtw_addr[] = {192,168,1,1};
193
194 // Global variables/constants for W5100 & MRF24J40
195 // porb_b_pullup enable
196 #byte INTCON2 = 0xFF1
197 #bit RBPU = INTCON2.7
198
199 // usb bootloader definitions
200 #define LED2 PIN_A1
201 #define LED_ON(x) output_low(x)
202 #define LED_OFF(x) output_high(x)
203 #define USB_CABLE_IS_ATTACHED() input(PIN_A7)
204 #define USB_CONFIG_VID 0x04D8
205 #define USB_CONFIG_PID 0x000C
206 #define USB_CONFIG_BUS_POWER 500

```



```
207 #define USB_STRINGS_OVERWRITTEN
208
209 char USB_STRING_DESC_OFFSET[]={0,4,12};
210
211 char const USB_STRING_DESC[]={
212     //string 0 - language
213     4, //length of string index
214     0x03, //descriptor type (STRING)
215     0x09,0x04, //Microsoft Defined for US-English
216     //string 1 - manufacturer
217     8, //length of string index
218     0x03, //descriptor type (STRING)
219     'U',0,
220     'o',0,
221     'I',0,
222     //string 2 - product
223     22, //length of string index
224     0x03, //descriptor type (STRING)
225     'R',0,
226     'e',0,
227     'm',0,
228     'o',0,
229     't',0,
230     'e',0,
231     ' ',0,
232     'l',0,
233     'a',0,
234     'b',0
235 };
236
237 #define USB_EP1_TX_SIZE 63
238 #define USB_EP1_RX_SIZE 63
239 #include <my_pic18_usb.h>
240 #include <my_usb_desc_bulk.h>
241 #include <my_usb.c>
```

W5100.h

```

241 // Wiznet W5100 Register Addresses
242 #define MR          0x0000 // Mode Register
243 #define GAR          0x0001 // Gateway Address: 0x0001 to 0x0004
244 #define SUBR          0x0005 // Subnet mask Address: 0x0005 to
    0x0008
245 #define SAR          0x0009 // Source Hardware Address (MAC):
    0x0009 to 0x000E
246 #define SIPR          0x000F // Source IP Address: 0x000F to 0x0012
247 #define RMSR          0x001A // RX Memory Size Register
248 #define TMSR          0x001B // TX Memory Size Register
249 #define S0_MR          0x0400 // Socket 0: Mode Register Address
250 #define S0_CR          0x0401 // Socket 0: Command Register Address
251 #define S0_IR          0x0402 // Socket 0: Interrupt Register
    Address
252 #define S0_SR          0x0403 // Socket 0: Status Register Address
253 #define S0_PORT          0x0404 // Socket 0: Source Port: 0x0404 to
    0x0405
254 #define S0_TX_FSR          0x0420 // Socket 0: Tx Free Size Register:
    0x0420 to 0x0421
255 #define S0_TX_RD          0x0422 // Socket 0: Tx Read Pointer Register:
    0x0422 to 0x0423
256 #define S0_TX_WR          0x0424 // Socket 0: Tx Write Pointer
    Register: 0x0424 to 0x0425
257 #define S0_RX_RSR          0x0426 // Socket 0: Rx Received Size Pointer
    Register: 0x0425 to 0x0427
258 #define S0_RX_RD          0x0428 // Socket 0: Rx Read Pointer: 0x0428
    to 0x0429
259 #define TXBUFADDR          0x4000 // W5100 Send Buffer Base Address
260 #define RXBUFADDR          0x6000 // W5100 Read Buffer Base Address
261 // S0_MR values
262 #define MR_CLOSE          0x00 // Unused socket
263 #define MR_TCP            0x01 // TCP
264 #define MR_UDP            0x02 // UDP
265 #define MR_IPRAW          0x03 // IP LAYER RAW SOCK
266 #define MR_MACRAW          0x04 // MAC LAYER RAW SOCK
267 #define MR_PPPOE          0x05 // PPPoE
268 #define MR_ND              0x20 // No Delayed Ack(TCP) flag
269 #define MR_MULTI          0x80 // support multicasting
270 // S0_CR values
271 #define CR_OPEN            0x01 // Initialize or open socket
272 #define CR_LISTEN          0x02 // Wait connection request in tcp
    mode(Server mode)
273 #define CR_CONNECT          0x04 // Send connection request in tcp
    mode(Client mode)
274 #define CR_DISCON          0x08 // Send closing request in tcp mode
275 #define CR_CLOSE          0x10 // Close socket
276 #define CR_SEND            0x20 // Update Tx memory pointer and send
    data
277 #define CR_SEND_MAC          0x21 // Send data with MAC address, so
    without ARP process
278 #define CR_SEND_KEEP          0x22 // Send keep alive message
279 #define CR_RECV            0x40 // Update Rx memory buffer pointer
    and receive data
280 // S0_SR values
281 #define SOCK_CLOSED          0x00 // Closed
282 #define SOCK_INIT          0x13 // Init state

```

```

283 #define SOCK_LISTEN      0x14      // Listen state
284 #define SOCK_SYNSENT     0x15      // Connection state
285 #define SOCK_SYNRCV      0x16      // Connection state
286 #define SOCK_ESTABLISHED 0x17      // Success to connect
287 #define SOCK_FIN_WAIT    0x18      // Closing state
288 #define SOCK_CLOSING     0x1A      // Closing state
289 #define SOCK_TIME_WAIT   0x1B      // Closing state
290 #define SOCK_CLOSE_WAIT  0x1C      // Closing state
291 #define SOCK_LAST_ACK     0x1D      // Closing state
292 #define SOCK_UDP          0x22      // UDP socket
293 #define SOCK_IPRAW        0x32      // IP raw mode socket
294 #define SOCK_MACRAW       0x42      // MAC raw mode socket
295 #define SOCK_PPPOE        0x5F      // PPPOE socket
296 #define TX_BUF_MASK       0x07FF    // Tx 2K Buffer Mask:
297 #define RX_BUF_MASK       0x07FF    // Rx 2K Buffer Mask:
298 #define NET_MEMALLOC      0x05      // Use 2K of Tx/Rx Buffer
299 #define TCP_PORT          80         // TCP/IP Port
300 // W5100 Socket Register and Variables Used
301 int8 sockreg;
302 #define MAX_BUF 256
303 int8 buf[MAX_BUF];
304 // W5100 SPI opcodes
305 #define WIZNET_WRITE_OPCODE 0xF0
306 #define WIZNET_READ_OPCODE 0x0F
307
308 // W5100 routines definitions
309 int W5100_spiBYTE(int8 data);
310 void W5100_spiWR (int16 reg, int8 data);
311 int8 W5100_spiRD (int16 reg);
312 void W5100_Init(void);
313 void close(int8 sock);
314 void disconnect(int8 sock);
315 int8 socket(int8 sock,int8 eth_protocol,int16 tcpport);
316 int8 listen(int8 sock);
317 int16 send(int8 sock,int8 *buf,int16 buflen);
318 int16 recv(int8 sock,int8 *buf,int16 buflen);
319 int16 recv_size(void);
320 int strCOMPARE(char *s,char *t);
321
322 // W5100 SPI routines
323 int W5100_spiBYTE(int8 data)
324 {
325     int i=0;
326     int readdata = 0x00;
327     for (i=0;i<8;i++)
328     {
329         if (data & 0x80)
330         {
331             output_high(W5100_MOSI); // define W5100_MOSI
332         }
333         else
334         {
335             output_low(W5100_MOSI); // define W5100_MOSI
336         }
337         delay_us(W5100_HalfPeriod);
338         output_high(W5100_SCLK); //set clock
339         data = data << 1;
340         if(i>0)

```

```

341     {
342         readdata = readdata << 1;
343     }
344     if (input(W5100_MISO))
345     {
346         readdata = (readdata|0x01);
347     }
348     else
349     {
350         readdata = (readdata&0xFE);
351     }
352     delay_us(W5100_HalfPeriod);
353     output_low(W5100_SCLK); //reset clock
354 }
355 output_low(W5100_MOSI);
356 return (readdata);
357 }
358
359 void W5100_spiWR (int16 reg, int8 data)
360 {
361     output_low(W5100_CS);
362     W5100_spiBYTE(WIZNET_WRITE_OPCODE);
363     W5100_spiBYTE((reg & 0xFF00) >> 8);
364     W5100_spiBYTE(reg & 0x00FF);
365     W5100_spiBYTE(data);
366     output_high(W5100_CS);
367 }
368
369
370 int8 W5100_spiRD (int16 reg)
371 {
372     int8 readDATA = 0x00;
373     output_low(W5100_CS);
374     W5100_spiBYTE(WIZNET_READ_OPCODE);
375     W5100_spiBYTE((reg & 0xFF00) >> 8);
376     W5100_spiBYTE(reg & 0x00FF);
377     readDATA = W5100_spiBYTE(0x00); //// Send Dummy byte to read SPI
data
378     output_high(W5100_CS);
379     return (readDATA);
380 }
381
382 //W5100 CONTROL routines
383 void W5100_Init(void)
384 {
385     // W5100_CS pin is not active
386     output_high(W5100_CS);
387     // Setting the Wiznet W5100 Mode Register: 0x0000
388     W5100_spiWR(MR,0x80); // MR = 0b10000000;
389     // Setting the Wiznet W5100 Gateway Address (GAR): 0x0001 to 0x0004
390     W5100_spiWR(GAR + 0,gtw_addr[0]);
391     W5100_spiWR(GAR + 1,gtw_addr[1]);
392     W5100_spiWR(GAR + 2,gtw_addr[2]);
393     W5100_spiWR(GAR + 3,gtw_addr[3]);
394     // Setting the Wiznet W5100 Source Address Register (SAR): 0x0009 to
0x000E
395     W5100_spiWR(SAR + 0,mac_addr[0]);
396     W5100_spiWR(SAR + 1,mac_addr[1]);

```

```

397     W5100_spiWR(SAR + 2,mac_addr[2]);
398     W5100_spiWR(SAR + 3,mac_addr[3]);
399     W5100_spiWR(SAR + 4,mac_addr[4]);
400     W5100_spiWR(SAR + 5,mac_addr[5]);
401     // Setting the Wiznet W5100 Sub Mask Address (SUBR): 0x0005 to
402     0x0008
403     W5100_spiWR(SUBR + 0,sub_mask[0]);
404     W5100_spiWR(SUBR + 1,sub_mask[1]);
405     W5100_spiWR(SUBR + 2,sub_mask[2]);
406     W5100_spiWR(SUBR + 3,sub_mask[3]);
407     // Setting the Wiznet W5100 IP Address (SIPR): 0x000F to 0x0012
408     W5100_spiWR(SIPR + 0,ip_addr[0]);
409     W5100_spiWR(SIPR + 1,ip_addr[1]);
410     W5100_spiWR(SIPR + 2,ip_addr[2]);
411     W5100_spiWR(SIPR + 3,ip_addr[3]);
412     // Setting the Wiznet W5100 RX and TX Memory Size (2KB),
413     W5100_spiWR(RMSR,NET_MEMALLOC);
414     W5100_spiWR(TMSR,NET_MEMALLOC);
415     // Initial variable used
416     sockreg=0;
417 }
418
419 void close(int8 sock)
420 {
421     if (sock != 0) return;
422
423     // Send Close Command
424     W5100_spiWR(S0_CR,CR_CLOSE);
425     // Waiting until the S0_CR is clear
426     while(W5100_spiRD(S0_CR));
427 }
428
429 void disconnect(int8 sock)
430 {
431     if (sock != 0) return;
432     // Send Disconnect Command
433     W5100_spiWR(S0_CR,CR_DISCON);
434     // Wait for Disconnecting Process
435     while(W5100_spiRD(S0_CR));
436 }
437
438 int8 socket(int8 sock,int8 eth_protocol,int16 tcpport)
439 {
440     int8 retval=0;
441     if (sock != 0) return retval;
442
443     // Make sure we close the socket first
444     if (W5100_spiRD(S0_SR) == SOCK_CLOSED) {
445         close(sock);
446     }
447     // Assigned Socket 0 Mode Register
448     W5100_spiWR(S0_MR,eth_protocol);
449
450     // Now open the Socket 0
451     W5100_spiWR(S0_PORT,((tcp_port & 0xFF00) >> 8));
452     W5100_spiWR(S0_PORT + 1,(tcp_port & 0x00FF));
453     W5100_spiWR(S0_CR,CR_OPEN); // Open Socket

```

```

454     // Wait for Opening Process
455     while(W5100_spiRD(S0_CR));
456     // Check for Init Status
457     if (W5100_spiRD(S0_SR) == SOCK_INIT)
458         retval=1;
459     else
460         close(sock);
461
462     return retval;
463 }
464
465 int8 listen(int8 sock)
466 {
467     int8 retval = 0;
468     if (sock != 0) return retval;
469     if (W5100_spiRD(S0_SR) == SOCK_INIT) {
470         // Send the LISTEN Command
471         W5100_spiWR(S0_CR,CR_LISTEN);
472
473         // Wait for Listening Process
474         while(W5100_spiRD(S0_CR));
475         // Check for Listen Status
476         if (W5100_spiRD(S0_SR) == SOCK_LISTEN)
477             retval=1;
478         else
479             close(sock);
480     }
481     return retval;
482 }
483
484 int16 send(int8 sock,int8 *buf,int16 buflen)
485 {
486     int16 ptr,offaddr,realaddr,txsize,timeout;
487
488     if (buflen <= 0 || sock != 0) return 0;
489     // Make sure the TX Free Size Register is available
490     txsize=W5100_spiRD(SO_TX_FSR);
491     txsize=((txsize & 0x00FF) << 8 ) + W5100_spiRD(SO_TX_FSR + 1));
492     timeout=0;
493     while (txsize < buflen) {
494         delay_ms(1);
495         txsize=W5100_spiRD(SO_TX_FSR);
496         txsize=((txsize & 0x00FF) << 8 ) + W5100_spiRD(SO_TX_FSR + 1));
497         // Timeout for approx 1000 ms
498         if (timeout++ > 1000) {
499             // Disconnect the connection
500             disconnect(sock);
501             return 0;
502         }
503     }
504     // Read the Tx Write Pointer
505     ptr = W5100_spiRD(SO_TX_WR);
506     offaddr = ((ptr & 0x00FF) << 8 ) + W5100_spiRD(SO_TX_WR + 1));
507
508     while(buflen) {
509         buflen--;
510         // Calculate the real W5100 physical Tx Buffer Address
511         realaddr = TXBUFADDR + (offaddr & TX_BUF_MASK);

```

```

512         // Copy the application data to the W5100 Tx Buffer
513         W5100_spiWR(realaddr,*buf);
514         offaddr++;
515         buf++;
516     }
517     // Increase the S0_TX_WR value, so it point to the next transmit
518     W5100_spiWR(S0_TX_WR,(offaddr & 0xFF00) >> 8 );
519     W5100_spiWR(S0_TX_WR + 1,(offaddr & 0x00FF));
520
521     // Now Send the SEND command
522     W5100_spiWR(S0_CR,CR_SEND);
523
524     // Wait for Sending Process
525     while(W5100_spiRD(S0_CR));
526
527     return 1;
528 }
529
530 int16 recv(int8 sock,int8 *buf,int16 buflen)
531 {
532     int16 ptr,offaddr,realaddr;
533
534     if (buflen <= 0 || sock != 0) return 1;
535
536     // If the request size > MAX_BUF,just truncate it
537     if (buflen > MAX_BUF)
538         buflen=MAX_BUF - 2;
539     // Read the Rx Read Pointer
540     ptr = W5100_spiRD(S0_RX_RD);
541     offaddr = ((ptr & 0x00FF) << 8 ) + W5100_spiRD(S0_RX_RD + 1));
542
543     while(buflen) {
544         buflen--;
545         realaddr=RXBUFADDR + (offaddr & RX_BUF_MASK);
546         *buf = W5100_spiRD(realaddr);
547         offaddr++;
548         buf++;
549     }
550     *buf='\0';          // String terminated character
551
552     // Increase the S0_RX_RD value, so it point to the next receive
553     W5100_spiWR(S0_RX_RD,(offaddr & 0xFF00) >> 8 );
554     W5100_spiWR(S0_RX_RD + 1,(offaddr & 0x00FF));
555
556     // Now Send the RECV command
557     W5100_spiWR(S0_CR,CR_RECV);
558     delay_us(5);        // Wait for Receive Process
559
560     return 1;
561 }
562
563 int16 recv_size(void)
564 {
565     return ((W5100_spiRD(S0_RX_RSR) & 0x00FF) << 8 ) +
566     W5100_spiRD(S0_RX_RSR + 1);
567 }
568 // W5100 MISC routines

```

```
569 int strCOMPARE(char *s,char *t) //Returns the number-th character of
    text
570 {
571     int16 i,n;
572
573     n=strlen(t);
574     for(i=0;*(s+i); i++) {
575         if (strncmp(s+i,t,n) == 0)
576             return i;
577     }
578     return -1;
579 }
```


MRF24J40.h

```

580 // MRF24J40 Short Register Addresses
581 #define RXMCR      0x00
582 #define PANIDL     0x01
583 #define PANIDH     0x02
584 #define SADRL      0x03
585 #define SADRH      0x04
586 #define EADR0      0x05
587 #define EADR1      0x06
588 #define EADR2      0x07
589 #define EADR3      0x08
590 #define EADR4      0x09
591 #define EADR5      0x0A
592 #define EADR6      0x0B
593 #define EADR7      0x0C
594 #define RXFLUSH    0x0D
595 #define ORDER      0x10
596 #define TXMCR      0x11
597 #define ACKTMOUT    0x12
598 #define ESLOTG1     0x13
599 #define SYMTICKL    0x14
600 #define SYMTICKH    0x15
601 #define PACON0     0x16
602 #define PACON1     0x17
603 #define PACON2     0x18
604 #define TXBCON0    0x1A
605 #define TXNCON     0x1B
606 #define TXG1CON    0x1C
607 #define TXG2CON    0x1D
608 #define ESLOTG23    0x1E
609 #define ESLOTG45    0x1F
610 #define ESLOTG67    0x20
611 #define TXPEND     0x21
612 #define WAKECON     0x22
613 #define FRMOFFSET   0x23
614 #define TXSTAT     0x24
615 #define TXBCON1    0x25
616 #define GATECLK     0x26
617 #define TXTIME      0x27
618 #define HSYMTMRL    0x28
619 #define HSYMTMRH    0x29
620 #define SOFTRST     0x2A
621 #define SECCON0     0x2C
622 #define SECCON1     0x2D
623 #define TXSTBL      0x2E
624 #define RXSR        0x30
625 #define INTSTAT     0x31
626 #define INTCON_M    0x32
627 #define GPIO        0x33
628 #define TRISGPIO    0x34
629 #define SLPACK      0x35
630 #define RFCTL        0x36
631 #define SECCR2       0x37
632 #define BBREG0       0x38
633 #define BBREG1       0x39
634 #define BBREG2       0x3A
635 #define BBREG3       0x3B

```

```

636 #define BBREG4      0x3C
637 #define BBREG6      0x3E
638 #define CCAEDTH     0x3F
639 // MRF24J40 Long Register Addresses
640 #define RFCON0      0x200
641 #define RFCON1      0x201
642 #define RFCON2      0x202
643 #define RFCON3      0x203
644 #define RFCON5      0x205
645 #define RFCON6      0x206
646 #define RFCON7      0x207
647 #define RFCON8      0x208
648 #define SLPCAL0     0x209
649 #define SLPCAL1     0x20A
650 #define SLPCAL2     0x20B
651 #define RFSTATE     0x20F
652 #define RSSI        0x210
653 #define SLPCON0     0x211
654 #define SLPCON1     0x220
655 #define WAKETIME1   0x222
656 #define WAKETIMEH   0x223
657 #define REMCNTL     0x224
658 #define REMCNTH     0x225
659 #define MAINCNT0    0x226
660 #define MAINCNT1    0x227
661 #define MAINCNT2    0x228
662 #define MAINCNT3    0x229
663 #define ASSOEADR0   0x230
664 #define ASSOEADR1   0x231
665 #define ASSOEADR2   0x232
666 #define ASSOEADR3   0x233
667 #define ASSOEADR4   0x234
668 #define ASSOEADR5   0x235
669 #define ASSOEADR6   0x236
670 #define ASSOEADR7   0x237
671 #define ASSOSADR0   0x238
672 #define ASSOSADR1   0x239
673 #define UPNONCE0    0x240
674 #define UPNONCE1    0x241
675 #define UPNONCE2    0x242
676 #define UPNONCE3    0x243
677 #define UPNONCE4    0x244
678 #define UPNONCE5    0x245
679 #define UPNONCE6    0x246
680 #define UPNONCE7    0x247
681 #define UPNONCE8    0x248
682 #define UPNONCE9    0x249
683 #define UPNONCE10   0x24A
684 #define UPNONCE11   0x24B
685 #define UPNONCE12   0x24C
686
687 // Global variables/constants for MRF24J40
688 int8 LQI = 0, RSSI2 = 0, SEQ_NUMBER = 0x23;
689 int16 address_RX_FIFO = 0x300;
690 int16 address_TX_normal_FIFO = 0x000;
691 int8 ADDRESS_short_1[] = {1,1}; // Source address
692 int8 ADDRESS_short_2[] = {2,2}; // Destination address
693 int8 PAN_ID_1[] = {3,3}; // Source PAN ID

```

```

694  int8 PAN_ID_2[] = {3,3}; // Destination PAN ID
695  int8 ADDRESS_long_1[] = {1,1,1,1,1,1,1,1};
696  int8 ADDRESS_long_2[] = {2,2,2,2,2,2,2,2};
697  const unsigned int8 DATA_LENGTH = 32;
698  const unsigned int8 HEADER_LENGTH = 11;
699  int8 data_RX_FIFO[1 + HEADER_LENGTH + DATA_LENGTH + 2 + 1 + 1],
700      lost_data = 0;
701  int8 DATA_RX[DATA_LENGTH],
702      DATA_TX[DATA_LENGTH],
703      data_TX_normal_FIFO[DATA_LENGTH + HEADER_LENGTH + 2];
704
705  // MRF24J40 routines definitions
706  int8 MRF24J40_spiBYTE(int8 data);
707  void write_ZIGBEE_short(int8 address, int8 data_r);
708  int8 read_ZIGBEE_short(int8 address);
709  void write_ZIGBEE_long(int16 address, int8 data_r);
710  int8 read_ZIGBEE_long(int16 address);
711  void pin_reset(void);
712  void PWR_reset(void);
713  void BB_reset(void);
714  void MAC_reset(void);
715  void software_reset(void);
716  void RF_reset(void);
717  void enable_interrupt(void);
718  void set_channel(int8 channel_number);
719  void set_CCA_mode(int8 CCA_mode);
720  void set_RSSI_mode(int8 RSSI_mode);
721  void nonbeacon_PAN_coordinator_device(void);
722  void nonbeacon_coordinator_device(void);
723  void nonbeacon_device(void);
724  void set_ACK(void);
725  void set_not_ACK(void);
726  void set_encrypt(void);
727  void set_not_encrypt(void);
728  void start_transmit(void);
729  void set_IFS_recommended(void);
730  void set_IFS_default(void);
731  void set_reception_mode(int8 r_mode);
732  void set_frame_format_filter(int8 fff_mode);
733  void flush_RX_FIFO_pointer(void);
734  void read_RX_FIFO(void);
735  void write_TX_normal_FIFO(void);
736  void set_short_address(int8 * address);
737  void set_long_address(int8 * address);
738  void set_PAN_ID(int8 * address);
739  void set_WAKE_from_pin(void);
740  void pin_WAKE(void);
741  void enable_PLL(void);
742  void disable_PLL(void);
743  void set_TX_power(unsigned int8 power);
744  void init_ZIGBEE_basic(void);
745  void init_ZIGBEE_nonbeacon(void);
746  void MRF24J40_init(void);
747
748  // MRF24J40 SPI routine
749  int8 MRF24J40_spiBYTE(int8 data)
750  {
751      int8 i=0;

```

```

752  int8 readdata = 0x00;
753      for (i=0;i<8;i++)
754      {
755          if (data & 0x80)
756          {
757              output_high(MRF24J40_MOSI); // define MRF24J40_MOSI
758          }
759          else
760          {
761              output_low(MRF24J40_MOSI); // define MRF24J40_MOSI
762          }
763          delay_us(MRF24J40_HalfPeriod);
764          output_high(MRF24J40_SCLK); //set clock
765          data = data << 1;
766          if(i>0)
767          {
768              readdata = readdata << 1;
769          }
770          if (input(MRF24J40_MISO))
771          {
772
773              readdata = (readdata|0x01);
774          }
775          else
776          {
777              readdata = (readdata&0xFE);
778          }
779          delay_us(MRF24J40_HalfPeriod);
780          output_low(MRF24J40_SCLK); //reset clock
781      }
782      output_low(MRF24J40_MOSI);
783  return (readdata);
784  }
785
786  // W5100 CONTROL routines
787  void write_ZIGBEE_short(int8 address, int8 data_r)
788  {
789      output_low(MRF24J40_CS); //MRF24J40_CS = 0;
790      delay_us(MRF24J40_HalfPeriod);
791      address = ((address << 1) & 0b01111111) | 0x01; // calculating
792      addressing mode
793      MRF24J40_spiBYTE(address); // addressing register
794      MRF24J40_spiBYTE(data_r); // write data in register
795      delay_us(MRF24J40_HalfPeriod);
796      output_high(MRF24J40_CS); //;MRF24J40_CS = 1;
797  }
798
799  int8 read_ZIGBEE_short(int8 address)
800  {
801      int8 data_r = 0, dummy_data_r = 0;
802      output_low(MRF24J40_CS); //MRF24J40_CS = 0;
803      delay_us(MRF24J40_HalfPeriod);
804      address = (address << 1) & 0b01111110; // calculating
805      addressing mode
806      MRF24J40_spiBYTE(address); // addressing
807      register

```

```

806     data_r = MRF24J40_spiBYTE(dummy_data_r);           // read data
      from register
807     delay_us(MRF24J40_HalfPeriod);
808     output_high(MRF24J40_CS); // MRF24J40_CS = 1;
809     return data_r;
810 }
811
812 void write_ZIGBEE_long(int16 address, int8 data_r)
813 {
814     int8 address_high = 0, address_low = 0;
815     output_low(MRF24J40_CS); //MRF24J40_CS = 0;
816     delay_us(MRF24J40_HalfPeriod);
817     address_high = (((int8)(address >> 3)) & 0b01111111) | 0x80; //
calculating addressing mode
818     address_low = (((int8)(address << 5)) & 0b11100000) | 0x10; //
calculating addressing mode
819     MRF24J40_spiBYTE(address_high);           // addressing register
820     MRF24J40_spiBYTE(address_low);           // addressing register
821     MRF24J40_spiBYTE(data_r);                 // write data in
register
822     delay_us(MRF24J40_HalfPeriod);
823     output_high(MRF24J40_CS); //MRF24J40_CS = 1;
824 }
825
826 int8 read_ZIGBEE_long(int16 address)
827 {
828     int8 data_r = 0, dummy_data_r = 0;
829     int8 address_high = 0, address_low = 0;
830     output_low(MRF24J40_CS);
831     delay_us(MRF24J40_HalfPeriod);
832     address_high = (((int8)(address >> 3)) & 0b01111111) | 0x80;
//calculating addressing mode
833     address_low = (((int8)(address << 5)) & 0b11100000);
//calculating addressing mode
834     MRF24J40_spiBYTE(address_high);           // addressing register
835     MRF24J40_spiBYTE(address_low);           // addressing register
836     data_r = MRF24J40_spiBYTE(dummy_data_r); // read data from
register
837     delay_us(MRF24J40_HalfPeriod);
838     output_high(MRF24J40_CS);
839     return data_r;
840 }
841
842 void pin_reset(void)
843 {
844     output_low(MRF24J40_RST); // activate reset
845     delay_ms(5);
846     output_high(MRF24J40_RST); // deactivate reset
847     delay_ms(5);
848 }
849
850 void PWR_reset(void)
851 {
852     write_ZIGBEE_short(SOFTTRST, 0x04);           //0x04 mask for
MRF24J40_RSTPWR bit
853 }
854
855 void BB_reset(void)

```

```

856 {
857     write_ZIGBEE_short(SOFTTRST, 0x02);           //0x02 mask for
MRF24J40_RSTBB bit
858 }
859
860 void MAC_reset(void)
861 {
862     write_ZIGBEE_short(SOFTTRST, 0x01);           //0x01 mask for
MRF24J40_RSTMAC bit
863 }
864
865 void software_reset(void) // PWR_reset,BB_reset and MAC_reset at once
866 {
867     write_ZIGBEE_short(SOFTTRST, 0x07);
868 }
869
870 void RF_reset(void)
871 {
872     int8 temp = 0;
873     temp = read_ZIGBEE_short(RFCTL);
874     temp = temp | 0x04;                           //mask for RFMRF24J40_RST
bit
875     write_ZIGBEE_short(RFCTL, temp);
876     temp = temp & (!0x04);                         //mask for RFMRF24J40_RST
bit
877     write_ZIGBEE_short(RFCTL, temp);
878     delay_ms(1);
879 }
880
881 void enable_interrupt(void)
882 {
883     write_ZIGBEE_short(INTCON_M, 0x00);           //0x00 all interrupts are
enable
884 }
885
886 void set_channel(int8 channel_number) // 11-26 possible channels
887 {
888     if((channel_number > 26) || (channel_number < 11)) channel_number =
11;
889     switch(channel_number)
890     {
891         case 11:
892             write_ZIGBEE_long(RFCON0, 0x02); //0x02 for 11. channel
893             break;
894         case 12:
895             write_ZIGBEE_long(RFCON0, 0x12); //0x12 for 12. channel
896             break;
897         case 13:
898             write_ZIGBEE_long(RFCON0, 0x22); //0x22 for 13. channel
899             break;
900         case 14:
901             write_ZIGBEE_long(RFCON0, 0x32); //0x32 for 14. channel
902             break;
903         case 15:
904             write_ZIGBEE_long(RFCON0, 0x42); //0x42 for 15. channel
905             break;
906         case 16:
907             write_ZIGBEE_long(RFCON0, 0x52); //0x52 for 16. channel

```

```

908         break;
909     case 17:
910         write_ZIGBEE_long(RFCON0, 0x62); //0x62 for 17. channel
911         break;
912     case 18:
913         write_ZIGBEE_long(RFCON0, 0x72); //0x72 for 18. channel
914         break;
915     case 19:
916         write_ZIGBEE_long(RFCON0, 0x82); //0x82 for 19. channel
917         break;
918     case 20:
919         write_ZIGBEE_long(RFCON0, 0x92); //0x92 for 20. channel
920         break;
921     case 21:
922         write_ZIGBEE_long(RFCON0, 0xA2); //0xA2 for 21. channel
923         break;
924     case 22:
925         write_ZIGBEE_long(RFCON0, 0xB2); //0xB2 for 22. channel
926         break;
927     case 23:
928         write_ZIGBEE_long(RFCON0, 0xC2); //0xC2 for 23. channel
929         break;
930     case 24:
931         write_ZIGBEE_long(RFCON0, 0xD2); //0xD2 for 24. channel
932         break;
933     case 25:
934         write_ZIGBEE_long(RFCON0, 0xE2); //0xE2 for 25. channel
935         break;
936     case 26:
937         write_ZIGBEE_long(RFCON0, 0xF2); //0xF2 for 26. channel
938         break;
939     }
940     RF_reset();
941 }
942
943 void set_CCA_mode(int8 CCA_mode)
944 {
945     int8 temp = 0;
946     switch(CCA_mode)
947     {
948         case 1: //ENERGY ABOVE THRESHOLD
949             {
950                 temp = read_ZIGBEE_short(BBREG2);
951                 temp = temp | 0x80; //0x80 mask
952                 temp = temp & 0xDF; //0xDF mask
953                 write_ZIGBEE_short(BBREG2, temp);
954
955                 write_ZIGBEE_short(CCAEDTH, 0x60); //Set CCA ED
threshold to -69 dBm
956             }
957             break;
958         case 2: //CARRIER SENSE ONLY
959             {
960                 temp = read_ZIGBEE_short(BBREG2);
961                 temp = temp | 0x40; // 0x40 mask
962                 temp = temp & 0x7F; // 0x7F mask
963                 write_ZIGBEE_short(BBREG2, temp);
964             }

```

```

965         temp = read_ZIGBEE_short(BBREG2);    // carrier sense
threshold
966         temp = temp | 0x38;
967         temp = temp & 0xFB;
968         write_ZIGBEE_short(BBREG2, temp);
969     }
970     break;
971     case 3:                                     //CARRIER SENSE AND ENERGY ABOVE
THRESHOLD
972     {
973         temp = read_ZIGBEE_short(BBREG2);
974         temp = temp | 0xC0;                    //0xC0 mask
975         write_ZIGBEE_short(BBREG2, temp);
976
977         temp = read_ZIGBEE_short(BBREG2);    // carrier sense
threshold
978         temp = temp | 0x38;                    // 0x38 mask
979         temp = temp & 0xFB;                    // 0xFB mask
980         write_ZIGBEE_short(BBREG2, temp);
981
982         write_ZIGBEE_short(CCAEDTH, 0x60);    //Set CCA ED
threshold to -69 dBm
983     }
984     break;
985 }
986 }
987
988 void set_RSSI_mode(int8 RSSI_mode) // 1 for RSSI1, 2 for RSSI2 mode
989 {
990     int8 temp = 0;
991     switch(RSSI_mode)
992     {
993         case 1:
994         {
995             temp = read_ZIGBEE_short(BBREG6);
996             temp = temp | 0x80;                //0x80 mask for RSSI1 mode
997             write_ZIGBEE_short(BBREG6, temp);
998         }
999         break;
1000        case 2:
1001            write_ZIGBEE_short(BBREG6, 0x40);  //0x40 data for RSSI2 mode
1002            break;
1003        }
1004    }
1005
1006 void nonbeacon_PAN_coordinator_device(void)
1007 {
1008     int8 temp = 0;
1009     temp = read_ZIGBEE_short(RXMCR);
1010     temp = temp | 0x08;                        // 0x08 mask for PAN
coordinator
1011     write_ZIGBEE_short(RXMCR, temp);
1012     temp = read_ZIGBEE_short(TXMCR);
1013     temp = temp & 0xDF;                        // 0xDF mask for
MRF24J40_CSMA-CA mode
1014     write_ZIGBEE_short(TXMCR, temp);
1015     write_ZIGBEE_short(ORDER, 0xFF);          // BO, SO are 15
1016 }

```



```

1017
1018 void nonbeacon_coordinator_device(void)
1019 {
1020     int8 temp = 0;
1021     temp = read_ZIGBEE_short(RXMCR);
1022     temp = temp | 0x04; // 0x04 mask for coordinator
1023     write_ZIGBEE_short(RXMCR, temp);
1024     temp = read_ZIGBEE_short(TXMCR);
1025     temp = temp & 0xDF; // 0xDF mask for
MRF24J40_CSMA-CA mode
1026     write_ZIGBEE_short(TXMCR, temp);
1027     write_ZIGBEE_short(ORDER, 0xFF); // BO, SO are 15
1028 }
1029
1030 void nonbeacon_device(void)
1031 {
1032     int8 temp = 0;
1033     temp = read_ZIGBEE_short(RXMCR);
1034     temp = temp & 0xF3; // 0xF3 mask for PAN coordinator
and coordinator
1035     write_ZIGBEE_short(RXMCR, temp);
1036     temp = read_ZIGBEE_short(TXMCR);
1037     temp = temp & 0xDF; // 0xDF mask for MRF24J40_CSMA-
CA mode
1038     write_ZIGBEE_short(TXMCR, temp);
1039 }
1040
1041 void set_ACK(void)
1042 {
1043     int8 temp = 0;
1044     temp = read_ZIGBEE_short(TXNCON);
1045     temp = temp | 0x04; //0x04 mask for set ACK
1046     write_ZIGBEE_short(TXNCON, temp);
1047 }
1048
1049 void set_not_ACK(void)
1050 {
1051     int8 temp = 0;
1052     temp = read_ZIGBEE_short(TXNCON);
1053     temp = temp & (!0x04); //0x04 mask for set not
ACK
1054     write_ZIGBEE_short(TXNCON, temp);
1055 }
1056
1057 void set_encrypt(void)
1058 {
1059     int8 temp = 0;
1060     temp = read_ZIGBEE_short(TXNCON);
1061     temp = temp | 0x02; // mask for set encrypt
1062     write_ZIGBEE_short(TXNCON, temp);
1063 }
1064
1065 void set_not_encrypt(void)
1066 {
1067     int8 temp = 0;
1068     temp = read_ZIGBEE_short(TXNCON);
1069     temp = temp & (!0x02); // mask for set not
encrypt

```

```

1070     write_ZIGBEE_short(TXNCON, temp);
1071 }
1072
1073 void start_transmit(void)
1074 {
1075     int8 temp = 0;
1076     temp = read_ZIGBEE_short(TXNCON);
1077     temp = temp | 0x01; // mask for start transmit
1078     write_ZIGBEE_short(TXNCON, temp);
1079 }
1080
1081 // interframe spacing
1082 void set_IFS_recommended(void)
1083 {
1084     int8 temp = 0;
1085     write_ZIGBEE_short(RXMCR, 0x93); // Min SIFS Period
1086     temp = read_ZIGBEE_short(TXPEND);
1087     temp = temp | 0x7C; // MinLIFSPeriod
1088     write_ZIGBEE_short(TXPEND, temp);
1089     temp = read_ZIGBEE_short(TXSTBL);
1090     temp = temp | 0x90; // MinLIFSPeriod
1091     write_ZIGBEE_short(TXSTBL, temp);
1092     temp = read_ZIGBEE_short(TXTIME);
1093     temp = temp | 0x31; // TurnaroundTime
1094     write_ZIGBEE_short(TXTIME, temp);
1095 }
1096
1097 void set_IFS_default(void)
1098 {
1099     int8 temp = 0;
1100     write_ZIGBEE_short(RXMCR, 0x75); // Min SIFS Period
1101     temp = read_ZIGBEE_short(TXPEND);
1102     temp = temp | 0x84; // Min LIFS Period
1103     write_ZIGBEE_short(TXPEND, temp);
1104     temp = read_ZIGBEE_short(TXSTBL);
1105     temp = temp | 0x50; // Min LIFS Period
1106     write_ZIGBEE_short(TXSTBL, temp);
1107     temp = read_ZIGBEE_short(TXTIME);
1108     temp = temp | 0x41; // Turnaround Time
1109     write_ZIGBEE_short(TXTIME, temp);
1110 }
1111
1112 void set_reception_mode(int8 r_mode) // 1 normal, 2 error, 3
promiscuous mode
1113 {
1114     int8 temp = 0;
1115     switch(r_mode)
1116     {
1117     case 1:
1118     {
1119         temp = read_ZIGBEE_short(RXMCR); // normal mode
1120         temp = temp & (!0x03); // mask for normal mode
1121         write_ZIGBEE_short(RXMCR, temp);
1122     }
1123     break;
1124     case 2:
1125     {
1126         temp = read_ZIGBEE_short(RXMCR); // error mode

```

```

1127         temp = temp & (!0x01);           // mask for error mode
1128         temp = temp | 0x02;               // mask for error mode
1129         write_ZIGBEE_short(RXMCR, temp);
1130     }
1131     break;
1132     case 3:
1133     {
1134         temp = read_ZIGBEE_short(RXMCR);   // promiscuous mode
1135         temp = temp & (!0x02);             // mask for promiscuous
mode
1136         temp = temp | 0x01;                // mask for promiscuous
mode
1137         write_ZIGBEE_short(RXMCR, temp);
1138     }
1139     break;
1140 }
1141 }
1142
1143 void set_frame_format_filter(int8 fff_mode) // 1 all frames, 2
command only, 3 data only, 4 beacon only
1144 {
1145     int8 temp = 0;
1146     switch(fff_mode)
1147     {
1148     case 1:
1149     {
1150         temp = read_ZIGBEE_short(RXFLUSH); // all frames
1151         temp = temp & (!0x0E);             // mask for all frames
1152         write_ZIGBEE_short(RXFLUSH, temp);
1153     }
1154     break;
1155     case 2:
1156     {
1157         temp = read_ZIGBEE_short(RXFLUSH); // command only
1158         temp = temp & (!0x06);             // mask for command
only
1159         temp = temp | 0x08;                // mask for command
only
1160         write_ZIGBEE_short(RXFLUSH, temp);
1161     }
1162     break;
1163     case 3:
1164     {
1165         temp = read_ZIGBEE_short(RXFLUSH); // data only
1166         temp = temp & (!0x0A);             // mask for data only
1167         temp = temp | 0x04;                // mask for data only
1168         write_ZIGBEE_short(RXFLUSH, temp);
1169     }
1170     break;
1171     case 4:
1172     {
1173         temp = read_ZIGBEE_short(RXFLUSH); // beacon only
1174         temp = temp & (!0x0C);             // mask for beacon
only
1175         temp = temp | 0x02;                // mask for beacon
only
1176         write_ZIGBEE_short(RXFLUSH, temp);
1177     }

```

```

1178         break;
1179     }
1180 }
1181
1182 void flush_RX_FIFO_pointer(void)
1183 {
1184     int8 temp;
1185     temp = read_ZIGBEE_short(RXFLUSH);
1186     temp = temp | 0x01; // mask for flush RX FIFO
1187     write_ZIGBEE_short(RXFLUSH, temp);
1188 }
1189
1190 void read_RX_FIFO(void)
1191 {
1192     unsigned int8 temp = 0;
1193     int8 i = 0;
1194     temp = read_ZIGBEE_short(BBREG1); // disable receiving
1195     packets off air. // mask for disable
1196     temp = temp | 0x04; // mask for disable
1197     write_ZIGBEE_short(BBREG1, temp);
1198     for(i=0; i<128; i++)
1199     {
1200         if(i < (1 + DATA_LENGTH + HEADER_LENGTH + 2 + 1 + 1))
1201             data_RX_FIFO[i] = read_ZIGBEE_long(address_RX_FIFO + i);
1202         // reading valid data from RX FIFO
1203         if(i >= (1 + DATA_LENGTH + HEADER_LENGTH + 2 + 1 + 1))
1204             lost_data = read_ZIGBEE_long(address_RX_FIFO + i);
1205         // reading invalid data from RX FIFO
1206     }
1207     for(i=0; i<DATA_LENGTH; i++)
1208     {
1209         DATA_RX[i] = data_RX_FIFO[HEADER_LENGTH + 1 + i];
1210         // coping valid data
1211     }
1212     LQI = data_RX_FIFO[1 + HEADER_LENGTH + DATA_LENGTH + 2]; //
1213     coping valid data
1214     RSSI2 = data_RX_FIFO[1 + HEADER_LENGTH + DATA_LENGTH + 3]; //
1215     coping valid data
1216
1217     temp = read_ZIGBEE_short(BBREG1); // enable receiving
1218     packets off air. // mask for enable
1219     temp = temp & (!0x04); // mask for enable
1220     write_ZIGBEE_short(BBREG1, temp);
1221 }
1222
1223 void write_TX_normal_FIFO(void)
1224 {
1225     int8 i = 0;
1226     data_TX_normal_FIFO[0] = HEADER_LENGTH;
1227     data_TX_normal_FIFO[1] = HEADER_LENGTH + DATA_LENGTH;
1228     data_TX_normal_FIFO[2] = 0x01; //
1229     control frame
1230     data_TX_normal_FIFO[3] = 0x88;
1231     data_TX_normal_FIFO[4] = SEQ_NUMBER; //
1232     sequence number

```

```

1224     data_TX_normal_FIFO[5] = PAN_ID_2[1];           //
destinatoire pan
1225     data_TX_normal_FIFO[6] = PAN_ID_2[0];
1226     data_TX_normal_FIFO[7] = ADDRESS_short_2[0];     //
destination address
1227     data_TX_normal_FIFO[8] = ADDRESS_short_2[1];
1228     data_TX_normal_FIFO[9] = PAN_ID_1[0];           //
source pan
1229     data_TX_normal_FIFO[10] = PAN_ID_1[1];
1230     data_TX_normal_FIFO[11] = ADDRESS_short_1[0];    //
source address
1231     data_TX_normal_FIFO[12] = ADDRESS_short_1[1];
1232     data_TX_normal_FIFO[13] = DATA_TX[0];           // data
1233     //data_TX_normal_FIFO[14] = DATA_TX[1];
1234     //data_TX_normal_FIFO[15] = DATA_TX[2];
1235     for(i = 0; i < (HEADER_LENGTH + DATA_LENGTH + 2); i++)
1236     {
1237         write_ZIGBEE_long(address_TX_normal_FIFO + i,
data_TX_normal_FIFO[i]); // write frame into normal FIFO
1238     }
1239     set_not_ACK();
1240     set_not_encrypt();
1241     start_transmit();
1242 }
1243
1244 void set_short_address(int8 * address)
1245 {
1246     write_ZIGBEE_short(SADR_L, address[0]);
1247     write_ZIGBEE_short(SADR_H, address[1]);
1248 }
1249
1250 void set_long_address(int8 * address)
1251 {
1252     int8 i = 0;
1253     for(i = 0; i < 8; i++)
1254     {
1255         write_ZIGBEE_short(EADR0 + i, address[i]);    // 0x05 address of
EADR0
1256     }
1257 }
1258
1259 void set_PAN_ID(int8 * address)
1260 {
1261     write_ZIGBEE_short(PANIDL, address[0]);
1262     write_ZIGBEE_short(PANIDH, address[1]);
1263 }
1264
1265 void set_WAKE_from_pin(void)
1266 {
1267     int8 temp = 0;
1268     output_low(MRF24J40_WAKE);
1269     temp = read_ZIGBEE_short(RXFLUSH);
1270     temp = temp | 0x60;                               // mask
1271     write_ZIGBEE_short(RXFLUSH, temp);
1272     temp = read_ZIGBEE_short(WAKECON);
1273     temp = temp | 0x80;
1274     write_ZIGBEE_short(WAKECON, temp);
1275 }

```

```

1276
1277 void pin_WAKE(void)
1278 {
1279     output_high(MRF24J40_WAKE);
1280     Delay_ms(5);
1281 }
1282
1283 void enable_PLL(void)
1284 {
1285     write_ZIGBEE_long(RFCON2, 0x80);           // mask for PLL enable
1286 }
1287
1288 void disable_PLL(void)
1289 {
1290     write_ZIGBEE_long(RFCON2, 0x00);           // mask for PLL disable
1291 }
1292
1293 void set_TX_power(unsigned int8 power) // 0-31 possible variants
1294 {
1295     if((power < 0) || (power > 31)) power = 31;
1296     power = 31 - power;                        //0 max, 31
1297     min -> 31 max, 0 min
1298     power = ((power & 0b00011111) << 3) & 0b11111000; //
1299     calculating power
1300     write_ZIGBEE_long(RFCON3, power);
1301 }
1302
1303 void init_ZIGBEE_basic(void)
1304 {
1305     write_ZIGBEE_short(PACON2, 0x98);           // Initialize FIFOEN = 1 and
1306     TXONTS = 0x6
1307     write_ZIGBEE_short(TXSTBL, 0x95);           // Initialize RFSTBL = 0x9
1308     write_ZIGBEE_long(RFCON1, 0x01);           // Initialize VCOOPT = 0x01
1309     enable_PLL();                               // Enable PLL (PLLEN = 1)
1310     write_ZIGBEE_long(RFCON6, 0x90);           // Initialize TXFIL = 1 and
1311     20MRECV = 1
1312     write_ZIGBEE_long(RFCON7, 0x80);           // Initialize SLPCLKSEL = 0x2
1313     (100 kHz internal oscillator)
1314     write_ZIGBEE_long(RFCON8, 0x10);           // Initialize RFVCO = 1
1315     write_ZIGBEE_long(SLPCON1, 0x21);         // Initialize CLKOUTEN = 1 and
1316     SLPCLKDIV = 0x01
1317 }
1318
1319 void init_ZIGBEE_nonbeacon(void)
1320 {
1321     init_ZIGBEE_basic();
1322     set_CCA_mode(1); // Set CCA mode to ED and set threshold
1323     set_RSSI_mode(2); // RSSI2 mode
1324     enable_interrupt(); // Enables all interrupts
1325     set_channel(11); // Channel 11
1326     RF_reset();
1327 }
1328
1329 void MRF24J40_init(void)
1330 {
1331     pin_reset();
1332     RF_reset();
1333     set_WAKE_from_pin();

```

```
1328   set_long_address(ADDRESS_long_1);
1329   set_short_address(ADDRESS_short_1);
1330   set_PAN_ID(PAN_ID_1);
1331   init_ZIGBEE_nonbeacon();
1332   nonbeacon_PAN_coordinator_device();
1333   set_TX_power(31);
1334   set_frame_format_filter(1);           // 1 all frames, 3 data
                                         frame only
1335   set_reception_mode(1);               // 1 normal mode
1336   pin_WAKE();
1337 }
```

2. Συσκευή μέτρησης 01

EndDevice.c

```

1  #include "EndDevice.h"
2  #include "USB_bootloader.h"
3  #include "MRF24J40.h"
4  #include <string.h>
5  #include <math.h>
6  #include "LPS331AP.h"
7
8  #PIN_SELECT INT3=PIN_B4 // Peripheral Pin Select (EXT INT3)
9
10 #INT_EXT3
11 void ext_isr(void)
12 {
13     read_ZIGBEE_short(INTSTAT); // Read and flush register INTSTAT
14     read_RX_FIFO();             // Read 32 bytes and store them to
DATA_RX array
15     if (DATA_RX[0] == EndDevice_Address_1) // Evaluate Destination
address and reply if match
16     {
17         LPS331AP_convert();           // Read Pressure in
bar e.g. <0.993214 bar'\0'>;
18         DATA_TX[0] = Coordinator_Address;
19         strcat (DATA_TX, strPressure_Temperature); // Copy
Pressure value to trasmit array
20         read_ZIGBEE_short(INTSTAT);           // Read and flush
register INTSTAT; clear receive interrupt
21         write_TX_normal_FIFO();               // Transmit Pressure
data to coordinator
22         DATA_RX[0] = 0;                       // Reset End Point
transfer to ensure the subsequent reception
23     }
24     clear_interrupt(INT_EXT3);
25 }
26
27 void main()
28 {
29     RBPU = 0; //Enable port_b_pullups when defined inputs (PB4 is
defined as EXT INT3)
30
31     // EXT INT3 Initialization
32     ext_int_edge(3,H_TO_L);
33     clear_interrupt(INT_EXT3);
34     enable_interrupts(INT_EXT3);
35     enable_interrupts(GLOBAL);
36
37     // MRF24J40 IEEE 802.15.4 radio Initialization
38     output_high(MRF24J40_CS);
39     output_high(MRF24J40_RST);
40     MRF24J40_init();
41
42     DATA_TX[0] = 0;
43     output_high(PIN_A2); // Turn off bootloader's LED (Power consumption
consideration)
44
45     while (1)
46     {

```



```
47     sleep(); //Deep Sleep mode can wake-up from INT0 only and also
        processor will start execution from the
48         //Power-on Reset vector rather than branch to the
        interrupt vector.
49         //Therefore, the microcontroller is entered in regular
        Sleep mode
50     }
51 }
```

EndDevice.h

```

52  #include <18F26J50.h>
53  #device PASS_STRINGS = IN_RAM //allow pointers to constants; forces
    the compiler to copy the constant strings into an internal RAM buffer
54  #device ADC=16
55
56  #FUSES NOWDT //No Watch Dog Timer
57  #FUSES WDT128 //Watch Dog Timer uses 1:128 Postscale
58  #FUSES NOXINST //Extended set extension and Indexed
    Addressing mode disabled (Legacy mode)
59  #FUSES NOT1DIG //Secondary Oscillator Source may be
    select if T1CON.3 = 1
60  #FUSES NOFCMEN //Fail-safe clock monitor disabled
61  #FUSES NOIESO //Internal External Switch Over mode
    disabled
62  #FUSES RTCOSC_INT //RTCC uses Internal 31KHz Oscillator
    as reference source
63
64  #use delay(internal=8000000,USB_FULL)
65
66  // MRF24J40 address
67  #define Coordinator_Address 0
68  #define EndDevice_Address_1 1
69  #define EndDevice_Address_2 2
70
71  // Pin & SPI clock definition for MRF24J40
72  #define MRF24J40_HalfPeriod 1
73  #define MRF24J40_CS PIN_A5
74  #define MRF24J40_MOSI PIN_B3
75  #define MRF24J40_MISO PIN_B2
76  #define MRF24J40_SCLK PIN_B1
77  #define MRF24J40_WAKE PIN_C7 //RXD on OLIMEX board
78  #define MRF24J40_RST PIN_C6 //TXD on OLIMEX board
79  #define MRF24J40_INT PIN_B4
80
81  // I2C definition for LPS331AP
82  #define LSP331AP_SDA PIN_c1
83  #define LSP331AP_SCL PIN_c0
84
85  // porb_b_pullup enable registers
86  #byte INTCON2 = 0xFF1
87  #bit RBPU = INTCON2.7

```

LPS331AP.h

```

88 // LPS331AP variable/constant definitions
89 #define LSP331AP_I2CdelayUS 1
90 char strPressure_bar[9]; // Example of string pressure
    value: 0.998365'\0'
91 char strTemperature_celsius[5]; // Example of string temperature
    value: 25.4'\0'
92 char strPressure_Temperature[16]; // Transmit 23 bytes
93
94 // LPS331AP routine definitions
95 void LPS331AP_I2Cdelay(void);
96 void LSP331AP_I2Cstart(void);
97 void LSP331AP_I2Cstop(void);
98 int8 LSP331AP_I2Cwrite(int8 value);
99 int8 LSP331AP_I2Cread(int8 value);
100 int8 LPS331AP_convert(int8);
101
102 // LPS331AP I2C routine
103 void LPS331AP_I2Cdelay(void)
104 {
105     delay_us(LSP331AP_I2CdelayUS);
106 }
107
108 void LSP331AP_I2Cstart(void)
109 {
110     LPS331AP_I2Cdelay();
111     LPS331AP_I2Cdelay();
112     output_float(LSP331AP_SCL);
113     LPS331AP_I2Cdelay();
114     output_low(LSP331AP_SDA);
115     LPS331AP_I2Cdelay();
116     output_low(LSP331AP_SCL);
117 }
118
119 void LSP331AP_I2Cstop(void)
120 {
121     output_low(LSP331AP_SDA);
122     LPS331AP_I2Cdelay();
123     output_float(LSP331AP_SCL);
124     LPS331AP_I2Cdelay();
125     output_float(LSP331AP_SDA);
126 }
127
128 int8 LSP331AP_I2Cwrite(int8 value)
129 {
130     int8 ack=0, i=0;
131     for (i=0;i<9;i++)
132     {
133         if (i<8)
134         {
135             if(value & 0x80)
136             {
137                 output_float(LSP331AP_SDA);
138             }
139             else
140             {
141                 output_low(LSP331AP_SDA);

```

```

142     }
143     LPS331AP_I2Cdelay();
144     output_float(LSP331AP_SCL);
145     LPS331AP_I2Cdelay();
146     output_low(LSP331AP_SCL);
147     value = value <<1;
148     if (i==7)
149     {
150         output_float(LSP331AP_SDA);
151         LPS331AP_I2Cdelay();
152     }
153 }
154 else
155 {
156     output_float(LSP331AP_SCL);
157     ack = input(LSP331AP_SDA);
158     LPS331AP_I2Cdelay();
159     output_low(LSP331AP_SCL);
160     LPS331AP_I2Cdelay();
161 }
162 }
163 return ack;
164 }
165
166 int8 LSP331AP_I2Cread(int8 value)
167 {
168     int8 I2C_readData=0, i=0;
169     for (i=0;i<8;i++)
170     {
171         if (input(LSP331AP_SDA))
172         {
173             I2C_readData = (I2C_readData|0x01);
174         }
175         else
176         {
177             I2C_readData = (I2C_readData&0xFE);
178         }
179         if(i==7)
180         {
181             I2C_readData = I2C_readData;
182         }
183         else
184         {
185             I2C_readData = I2C_readData << 1;
186         }
187         output_float(LSP331AP_SCL);
188         LPS331AP_I2Cdelay();
189         output_low(LSP331AP_SCL);
190         LPS331AP_I2Cdelay();
191     }
192     if(value == 1)
193     {
194         output_low(LSP331AP_SDA);
195     }
196     else
197     {
198         output_float(LSP331AP_SDA);
199     }

```

```

200     LPS331AP_I2Cdelay();
201     output_float(LSP331AP_SCL);
202     LPS331AP_I2Cdelay();
203     output_low(LSP331AP_SCL);
204     output_float(LSP331AP_SDA);
205     LPS331AP_I2Cdelay();
206     return (I2C_readData);
207 }
208
209 // LPS331AP CONTROL routines
210 void LPS331AP_write(int8 reg, int8 data)
211 {
212     int8 ack = 0;
213     LSP331AP_I2Cstart(); //I2C start condition
214     ack = LSP331AP_I2Cwrite((0x5D<<1) & 0xFE); //send I2C device
address (LSB=0 to write)
215     if (ack==0)
216     {
217         reg = reg & 0x7F; //MSB=0 so that the address remains unchanged
in multiple writes
218         ack = LSP331AP_I2Cwrite(reg); //send I2C write register (if
reg. addr. MSB=0, it performs single write)
219         if(ack==0)
220         {
221             ack = LSP331AP_I2Cwrite(data); //send I2C write value
222         }
223     }
224     if(ack==0)
225     {
226         LSP331AP_I2Cstop();
227     }
228 }
229
230 int8 LPS331AP_read(int8 reg)
231 {
232     int8 RegisterValue = 0;
233     int8 ack = 0;
234     LSP331AP_I2Cstart(); //I2C start condition
235     ack = LSP331AP_I2Cwrite((0x5D<<1) & 0xFE); //send I2C device
address (LSB=0 to write)
236     if(ack==0)
237     {
238         reg = reg & 0x7F; //MSB=0 so that the address remains unchanged
in multiple reads
239         ack = LSP331AP_I2Cwrite(reg); //send register address
240         if(ack==0)
241         {
242             LSP331AP_I2Cstart(); //repeat start condition
243             ack = LSP331AP_I2Cwrite((0x5D<<1) | 0x01); //send I2C reg.
address (LSB=1 to read)
244             if(ack==0)
245             {
246                 RegisterValue = LSP331AP_I2Cread(0); //read byte and send
NACK from master to slave
247                 LSP331AP_I2Cstop(); //send stop
248             }
249         }
250     }

```

```

251     return RegisterValue;
252 }
253
254 void LPS331AP_convert(void)
255 {
256     int1 sign = 0;
257     int8 i;
258     int8 convert = 0;
259     unsigned int8 Pressure_hex[3]; // read 3 bytes for pressure and 5
bytes for temperature as well
260     unsigned int8 Temperature_hex[2]; // read 2 bytes for temperature
261     unsigned int16 Temperature;
262     unsigned int32 Pressure;
263     float32 Pressure_bar, Temperature_celsius;
264
265     LPS331AP_write(0x20,0x80); //CTRL_REG1: active device
266     LPS331AP_write(0x21,0x01); //CTRL_REG2: start a new conversion
267     do
268     {
269         convert = LPS331AP_read(0x21); //wait until CTRL_REG2[bit0] is
cleared (i.e., conversion is finished);
270     } while (convert != 0);
271
272     for(i = 0; i<3; i++)
273     {
274         Pressure_hex[i] = LPS331AP_read(0x28+i);
275     }
276     for(i = 0; i<2; i++)
277     {
278         Temperature_hex[i] = LPS331AP_read(0x2B+i);
279     }
280     // do the appropriate math conversions
281     Pressure = make32(Pressure_hex[2],Pressure_hex[1],Pressure_hex[0]);
282     Pressure_bar = (float)Pressure;
283     Pressure_bar = Pressure_bar / 4096 / 1000;
284     sprintf(strPressure_bar,"%1.6f",Pressure_bar);
285     Temperature = make16(Temperature_hex[1],Temperature_hex[0]);
286     if ( (Temperature & 0x8000) != 0 )
287     {
288         sign = 1;
289         Temperature = (~Temperature)+1; //two's complement notation 2^N-
Tadc (i.e., find the corresponding positive and set sign bit)
290     }
291     Temperature_celsius = (float) Temperature;
292     Temperature_celsius = (Temperature_celsius / 480);
293     if (sign)
294     {
295         Temperature_celsius = Temperature_celsius * (-1);
296         sign = 0;
297     }
298     Temperature_celsius = 42.5 + Temperature_celsius;
299     sprintf(strTemperature_celsius,"%2.1f",Temperature_celsius);
300     strPressure_Temperature[0] = strPressure_bar[0]; //
e.g., '0'
301     strPressure_Temperature[1] = strPressure_bar[1]; //
e.g., '.'
302     strPressure_Temperature[2] = strPressure_bar[2]; //
e.g., '9'

```

```
303     strPressure_Temperature[3] = strPressure_bar[3];           //
    e.g., '9'
304     strPressure_Temperature[4] = strPressure_bar[4];           //
    e.g., '8'
305     strPressure_Temperature[5] = strPressure_bar[5];           //
    e.g., '7'
306     strPressure_Temperature[6] = strPressure_bar[6];           //
    e.g., '6'
307     strPressure_Temperature[7] = strPressure_bar[7];           //
    e.g., '5'
308     strPressure_Temperature[8] = ' ';                           //
    e.g., ' '
309     strPressure_Temperature[9] = ' ';                           //
    e.g., ' '
310     strPressure_Temperature[10] = ' ';                          //
    e.g., ' '
311     strPressure_Temperature[11] = strTemperature_celsius[0];    //
    e.g., '2'
312     strPressure_Temperature[12] = strTemperature_celsius[1];    //
    e.g., '5'
313     strPressure_Temperature[13] = strTemperature_celsius[2];    //
    e.g., '.'
314     strPressure_Temperature[14] = strTemperature_celsius[3];    //
    e.g., '4'
315     strPressure_Temperature[15] = '\0';                          //
    e.g., '\0'
316     //usb_puts(1, strPressure_bar, 13, 0); //2*bytes, 0);
317     LPS331AP_write(0x20,0x00); //CTRL_REG1: deactivate device
318 }
```

MRF24J40.h

```

319 // MRF24J40 Short Register Addresses
320 #define RXMCR      0x00
321 #define PANIDL     0x01
322 #define PANIDH     0x02
323 #define SADRL      0x03
324 #define SADRH      0x04
325 #define EADR0      0x05
326 #define EADR1      0x06
327 #define EADR2      0x07
328 #define EADR3      0x08
329 #define EADR4      0x09
330 #define EADR5      0x0A
331 #define EADR6      0x0B
332 #define EADR7      0x0C
333 #define RXFLUSH    0x0D
334 #define ORDER      0x10
335 #define TXMCR      0x11
336 #define ACKTMOUT    0x12
337 #define ESLOTG1     0x13
338 #define SYMTICKL    0x14
339 #define SYMTICKH    0x15
340 #define PACON0      0x16
341 #define PACON1      0x17
342 #define PACON2      0x18
343 #define TXBCON0     0x1A
344 #define TXNCON      0x1B
345 #define TXG1CON     0x1C
346 #define TXG2CON     0x1D
347 #define ESLOTG23    0x1E
348 #define ESLOTG45    0x1F
349 #define ESLOTG67    0x20
350 #define TXPEND      0x21
351 #define WAKECON     0x22
352 #define FRMOFFSET   0x23
353 #define TXSTAT      0x24
354 #define TXBCON1     0x25
355 #define GATECLK     0x26
356 #define TXTIME      0x27
357 #define HSYMTMRL    0x28
358 #define HSYMTMRH    0x29
359 #define SOFTRST     0x2A
360 #define SECCON0     0x2C
361 #define SECCON1     0x2D
362 #define TXSTBL      0x2E
363 #define RXSR        0x30
364 #define INTSTAT     0x31
365 #define INTCON_M     0x32
366 #define GPIO        0x33
367 #define TRISGPIO    0x34
368 #define SLPACK      0x35
369 #define RFCTL       0x36
370 #define SECCR2      0x37
371 #define BBREG0      0x38
372 #define BBREG1      0x39
373 #define BBREG2      0x3A
374 #define BBREG3      0x3B

```



```

375 #define BBREG4      0x3C
376 #define BBREG6      0x3E
377 #define CCAEDTH      0x3F
378 // MRF24J40 Long Register Addresses
379 #define RFCON0       0x200
380 #define RFCON1       0x201
381 #define RFCON2       0x202
382 #define RFCON3       0x203
383 #define RFCON5       0x205
384 #define RFCON6       0x206
385 #define RFCON7       0x207
386 #define RFCON8       0x208
387 #define SLPCAL0      0x209
388 #define SLPCAL1      0x20A
389 #define SLPCAL2      0x20B
390 #define RFSTATE      0x20F
391 #define RSSI         0x210
392 #define SLPCON0      0x211
393 #define SLPCON1      0x220
394 #define WAKETIME1    0x222
395 #define WAKETIMEH    0x223
396 #define REMCNTL      0x224
397 #define REMCNTH      0x225
398 #define MAINCNT0     0x226
399 #define MAINCNT1     0x227
400 #define MAINCNT2     0x228
401 #define MAINCNT3     0x229
402 #define ASSOEADR0    0x230
403 #define ASSOEADR1    0x231
404 #define ASSOEADR2    0x232
405 #define ASSOEADR3    0x233
406 #define ASSOEADR4    0x234
407 #define ASSOEADR5    0x235
408 #define ASSOEADR6    0x236
409 #define ASSOEADR7    0x237
410 #define ASSOSADR0    0x238
411 #define ASSOSADR1    0x239
412 #define UPNONCE0     0x240
413 #define UPNONCE1     0x241
414 #define UPNONCE2     0x242
415 #define UPNONCE3     0x243
416 #define UPNONCE4     0x244
417 #define UPNONCE5     0x245
418 #define UPNONCE6     0x246
419 #define UPNONCE7     0x247
420 #define UPNONCE8     0x248
421 #define UPNONCE9     0x249
422 #define UPNONCE10    0x24A
423 #define UPNONCE11    0x24B
424 #define UPNONCE12    0x24C
425
426 // Global variables/constants for MRF24J40
427 int8 LQI = 0, RSSI2 = 0, SEQ_NUMBER = 0x23;
428 int16 address_RX_FIFO = 0x300;
429 int16 address_TX_normal_FIFO = 0x000;
430 int8 ADDRESS_short_1[] = {1,1}; // Source address
431 int8 ADDRESS_short_2[] = {2,2}; // Destination address
432 int8 PAN_ID_1[] = {3,3}; // Source PAN ID

```

```

433  int8 PAN_ID_2[] = {3,3}; // Destination PAN ID
434  int8 ADDRESS_long_1[] = {1,1,1,1,1,1,1,1};
435  int8 ADDRESS_long_2[] = {2,2,2,2,2,2,2,2};
436  const unsigned int8 DATA_LENGTH = 32; // ### define here the
      number of transmitted data ###
437  const unsigned int8 HEADER_LENGTH = 11;
438  int8 data_RX_FIFO[1 + HEADER_LENGTH + DATA_LENGTH + 2 + 1 + 1],
439      lost_data = 0;
440  int8 DATA_RX[DATA_LENGTH],
441      DATA_TX[DATA_LENGTH],
442      data_TX_normal_FIFO[DATA_LENGTH + HEADER_LENGTH + 2];
443
444  // MRF24J40 routines definitions
445  int8 MRF24J40_spiBYTE(int8 data);
446  void write_ZIGBEE_short(int8 address, int8 data_r);
447  int8 read_ZIGBEE_short(int8 address);
448  void write_ZIGBEE_long(int16 address, int8 data_r);
449  int8 read_ZIGBEE_long(int16 address);
450  void pin_reset(void);
451  void PWR_reset(void);
452  void BB_reset(void);
453  void MAC_reset(void);
454  void software_reset(void);
455  void RF_reset(void);
456  void enable_interrupt(void);
457  void set_channel(int8 channel_number);
458  void set_CCA_mode(int8 CCA_mode);
459  void set_RSSI_mode(int8 RSSI_mode);
460  void nonbeacon_PAN_coordinator_device(void);
461  void nonbeacon_coordinator_device(void);
462  void nonbeacon_device(void);
463  void set_ACK(void);
464  void set_not_ACK(void);
465  void set_encrypt(void);
466  void set_not_encrypt(void);
467  void start_transmit(void);
468  void set_IFS_recommended(void);
469  void set_IFS_default(void);
470  void set_reception_mode(int8 r_mode);
471  void set_frame_format_filter(int8 fff_mode);
472  void flush_RX_FIFO_pointer(void);
473  void read_RX_FIFO(void);
474  void write_TX_normal_FIFO(void);
475  void set_short_address(int8 * address);
476  void set_long_address(int8 * address);
477  void set_PAN_ID(int8 * address);
478  void set_WAKE_from_pin(void);
479  void pin_WAKE(void);
480  void enable_PLL(void);
481  void disable_PLL(void);
482  void set_TX_power(unsigned int8 power);
483  void init_ZIGBEE_basic(void);
484  void init_ZIGBEE_nonbeacon(void);
485  void MRF24J40_init(void);
486
487  // MRF24J40 SPI routine
488  int8 MRF24J40_spiBYTE(int8 data)
489  {

```

```

490     int8 i=0;
491     int8 readdata = 0x00;
492     for (i=0;i<8;i++)
493     {
494         if (data & 0x80)
495         {
496             output_high(MRF24J40_MOSI); // define MRF24J40_MOSI
497         }
498         else
499         {
500             output_low(MRF24J40_MOSI); // define MRF24J40_MOSI
501         }
502         delay_us(MRF24J40_HalfPeriod);
503         output_high(MRF24J40_SCLK); //set clock
504         data = data << 1;
505         if(i>0)
506         {
507             readdata = readdata << 1;
508         }
509         if (input(MRF24J40_MISO))
510         {
511
512             readdata = (readdata|0x01);
513
514         }
515         else
516         {
517             readdata = (readdata&0xFE);
518         }
519         delay_us(MRF24J40_HalfPeriod);
520         output_low(MRF24J40_SCLK); //reset clock
521     }
522     output_low(MRF24J40_MOSI);
523     return (readdata);
524 }
525
526 // W5100 CONTROL routines
527 void write_ZIGBEE_short(int8 address, int8 data_r)
528 {
529     output_low(MRF24J40_CS); //MRF24J40_CS = 0;
530     delay_us(MRF24J40_HalfPeriod);
531     address = ((address << 1) & 0b01111111) | 0x01; // calculating
addressing mode
532     MRF24J40_spiBYTE(address); // addressing register
533     MRF24J40_spiBYTE(data_r); // write data in register
534     delay_us(MRF24J40_HalfPeriod);
535     output_high(MRF24J40_CS); //;MRF24J40_CS = 1;
536 }
537
538 int8 read_ZIGBEE_short(int8 address)
539 {
540     int8 data_r = 0, dummy_data_r = 0;
541     output_low(MRF24J40_CS); //MRF24J40_CS = 0;
542     delay_us(MRF24J40_HalfPeriod);
543     address = (address << 1) & 0b01111110; // calculating
addressing mode
544     MRF24J40_spiBYTE(address); // addressing
register

```

```

545     data_r = MRF24J40_spiBYTE(dummy_data_r);           // read data
from register
546     delay_us(MRF24J40_HalfPeriod);
547     output_high(MRF24J40_CS); // MRF24J40_CS = 1;
548     return data_r;
549 }
550
551 void write_ZIGBEE_long(int16 address, int8 data_r)
552 {
553     int8 address_high = 0, address_low = 0;
554     output_low(MRF24J40_CS); //MRF24J40_CS = 0;
555     delay_us(MRF24J40_HalfPeriod);
556     address_high = (((int8)(address >> 3)) & 0b01111111) | 0x80; //
calculating addressing mode
557     address_low = (((int8)(address << 5)) & 0b11100000) | 0x10; //
calculating addressing mode
558     MRF24J40_spiBYTE(address_high);           // addressing register
559     MRF24J40_spiBYTE(address_low);           // addressing register
560     MRF24J40_spiBYTE(data_r);               // write data in
register
561     delay_us(MRF24J40_HalfPeriod);
562     output_high(MRF24J40_CS); //MRF24J40_CS = 1;
563 }
564
565 int8 read_ZIGBEE_long(int16 address)
566 {
567     int8 data_r = 0, dummy_data_r = 0;
568     int8 address_high = 0, address_low = 0;
569     output_low(MRF24J40_CS);
570     delay_us(MRF24J40_HalfPeriod);
571     address_high = (((int8)(address >> 3)) & 0b01111111) | 0x80;
//calculating addressing mode
572     address_low = (((int8)(address << 5)) & 0b11100000);
//calculating addressing mode
573     MRF24J40_spiBYTE(address_high);           // addressing register
574     MRF24J40_spiBYTE(address_low);           // addressing register
575     data_r = MRF24J40_spiBYTE(dummy_data_r); // read data from
register
576     delay_us(MRF24J40_HalfPeriod);
577     output_high(MRF24J40_CS);
578     return data_r;
579 }
580
581 void pin_reset(void)
582 {
583     output_low(MRF24J40_RST); // activate reset
584     delay_ms(5);
585     output_high(MRF24J40_RST); // deactivate reset
586     delay_ms(5);
587 }
588
589 void PWR_reset(void)
590 {
591     write_ZIGBEE_short(SOFTTRST, 0x04);           //0x04 mask for
MRF24J40_RSTPWR bit
592 }
593
594 void BB_reset(void)

```

```

595 {
596     write_ZIGBEE_short(SOFTTRST, 0x02);           //0x02 mask for
MRF24J40_RSTBB bit
597 }
598
599 void MAC_reset(void)
600 {
601     write_ZIGBEE_short(SOFTTRST, 0x01);           //0x01 mask for
MRF24J40_RSTMAC bit
602 }
603
604 void software_reset(void) // PWR_reset,BB_reset and MAC_reset at once
605 {
606     write_ZIGBEE_short(SOFTTRST, 0x07);
607 }
608
609 void RF_reset(void)
610 {
611     int8 temp = 0;
612     temp = read_ZIGBEE_short(RFCTL);
613     temp = temp | 0x04;                             //mask for RFMRF24J40_RST
bit
614     write_ZIGBEE_short(RFCTL, temp);
615     temp = temp & (!0x04);                         //mask for RFMRF24J40_RST
bit
616     write_ZIGBEE_short(RFCTL, temp);
617     delay_ms(1);
618 }
619
620 void enable_interrupt(void)
621 {
622     write_ZIGBEE_short(INTCON_M, 0x01);           //all interrupts are enable
except TX Normal FIFO transmission interrupt
623                                     //when TX Normal FIFO
transmission interrupt INT pin goes low for long time and transmission
is repeated
624 }
625
626 void set_channel(int8 channel_number) // 11-26 possible channels
627 {
628     if((channel_number > 26) || (channel_number < 11)) channel_number =
11;
629     switch(channel_number)
630     {
631         case 11:
632             write_ZIGBEE_long(RFCON0, 0x02); //0x02 for 11. channel
633             break;
634         case 12:
635             write_ZIGBEE_long(RFCON0, 0x12); //0x12 for 12. channel
636             break;
637         case 13:
638             write_ZIGBEE_long(RFCON0, 0x22); //0x22 for 13. channel
639             break;
640         case 14:
641             write_ZIGBEE_long(RFCON0, 0x32); //0x32 for 14. channel
642             break;
643         case 15:
644             write_ZIGBEE_long(RFCON0, 0x42); //0x42 for 15. channel

```

```

645         break;
646     case 16:
647         write_ZIGBEE_long(RFCON0, 0x52); //0x52 for 16. channel
648         break;
649     case 17:
650         write_ZIGBEE_long(RFCON0, 0x62); //0x62 for 17. channel
651         break;
652     case 18:
653         write_ZIGBEE_long(RFCON0, 0x72); //0x72 for 18. channel
654         break;
655     case 19:
656         write_ZIGBEE_long(RFCON0, 0x82); //0x82 for 19. channel
657         break;
658     case 20:
659         write_ZIGBEE_long(RFCON0, 0x92); //0x92 for 20. channel
660         break;
661     case 21:
662         write_ZIGBEE_long(RFCON0, 0xA2); //0xA2 for 21. channel
663         break;
664     case 22:
665         write_ZIGBEE_long(RFCON0, 0xB2); //0xB2 for 22. channel
666         break;
667     case 23:
668         write_ZIGBEE_long(RFCON0, 0xC2); //0xC2 for 23. channel
669         break;
670     case 24:
671         write_ZIGBEE_long(RFCON0, 0xD2); //0xD2 for 24. channel
672         break;
673     case 25:
674         write_ZIGBEE_long(RFCON0, 0xE2); //0xE2 for 25. channel
675         break;
676     case 26:
677         write_ZIGBEE_long(RFCON0, 0xF2); //0xF2 for 26. channel
678         break;
679     }
680     RF_reset();
681 }
682
683 void set_CCA_mode(int8 CCA_mode)
684 {
685     int8 temp = 0;
686     switch(CCA_mode)
687     {
688         case 1: //ENERGY ABOVE THRESHOLD
689             {
690                 temp = read_ZIGBEE_short(BBREG2);
691                 temp = temp | 0x80; //0x80 mask
692                 temp = temp & 0xDF; //0xDF mask
693                 write_ZIGBEE_short(BBREG2, temp);
694
695                 write_ZIGBEE_short(CCAEDTH, 0x60); //Set CCA ED
threshold to -69 dBm
696             }
697             break;
698         case 2: //CARRIER SENSE ONLY
699             {
700                 temp = read_ZIGBEE_short(BBREG2);
701                 temp = temp | 0x40; // 0x40 mask

```

```

702         temp = temp & 0x7F;           // 0x7F mask
703         write_ZIGBEE_short(BBREG2, temp);
704
705         temp = read_ZIGBEE_short(BBREG2);    // carrier sense
threshold
706         temp = temp | 0x38;
707         temp = temp & 0xFB;
708         write_ZIGBEE_short(BBREG2, temp);
709     }
710     break;
711     case 3:                               //CARRIER SENSE AND ENERGY ABOVE
THRESHOLD
712     {
713         temp = read_ZIGBEE_short(BBREG2);
714         temp = temp | 0xC0;               //0xC0 mask
715         write_ZIGBEE_short(BBREG2, temp);
716
717         temp = read_ZIGBEE_short(BBREG2);    // carrier sense
threshold
718         temp = temp | 0x38;               // 0x38 mask
719         temp = temp & 0xFB;               // 0xFB mask
720         write_ZIGBEE_short(BBREG2, temp);
721
722         write_ZIGBEE_short(CCAEDTH, 0x60);    //Set CCA ED
threshold to -69 dBm
723     }
724     break;
725 }
726 }
727
728 void set_RSSI_mode(int8 RSSI_mode) // 1 for RSSI1, 2 for RSSI2 mode
729 {
730     int8 temp = 0;
731     switch(RSSI_mode)
732     {
733         case 1:
734         {
735             temp = read_ZIGBEE_short(BBREG6);
736             temp = temp | 0x80;           //0x80 mask for RSSI1 mode
737             write_ZIGBEE_short(BBREG6, temp);
738         }
739         break;
740         case 2:
741             write_ZIGBEE_short(BBREG6, 0x40);    //0x40 data for RSSI2 mode
742             break;
743     }
744 }
745
746 void nonbeacon_PAN_coordinator_device(void)
747 {
748     int8 temp = 0;
749     temp = read_ZIGBEE_short(RXMCR);
750     temp = temp | 0x08;                   // 0x08 mask for PAN
coordinator
751     write_ZIGBEE_short(RXMCR, temp);
752     temp = read_ZIGBEE_short(TXMCR);
753     temp = temp & 0xDF;                   // 0xDF mask for
MRF24J40_CSMA-CA mode

```

```

754     write_ZIGBEE_short(TXMCR, temp);
755     write_ZIGBEE_short(ORDER, 0xFF); // BO, SO are 15
756 }
757
758 void nonbeacon_coordinator_device(void)
759 {
760     int8 temp = 0;
761     temp = read_ZIGBEE_short(RXMCR);
762     temp = temp | 0x04; // 0x04 mask for coordinator
763     write_ZIGBEE_short(RXMCR, temp);
764     temp = read_ZIGBEE_short(TXMCR);
765     temp = temp & 0xDF; // 0xDF mask for
MRF24J40_CSMA-CA mode
766     write_ZIGBEE_short(TXMCR, temp);
767     write_ZIGBEE_short(ORDER, 0xFF); // BO, SO are 15
768 }
769
770 void nonbeacon_device(void)
771 {
772     int8 temp = 0;
773     temp = read_ZIGBEE_short(RXMCR);
774     temp = temp & 0xF3; // 0xF3 mask for PAN coordinator
and coordinator
775     write_ZIGBEE_short(RXMCR, temp);
776     temp = read_ZIGBEE_short(TXMCR);
777     temp = temp & 0xDF; // 0xDF mask for MRF24J40_CSMA-
CA mode
778     write_ZIGBEE_short(TXMCR, temp);
779 }
780
781 void set_ACK(void)
782 {
783     int8 temp = 0;
784     temp = read_ZIGBEE_short(TXNCON);
785     temp = temp | 0x04; // 0x04 mask for set ACK
786     write_ZIGBEE_short(TXNCON, temp);
787 }
788
789 void set_not_ACK(void)
790 {
791     int8 temp = 0;
792     temp = read_ZIGBEE_short(TXNCON);
793     temp = temp & (!0x04); // 0x04 mask for set not
ACK
794     write_ZIGBEE_short(TXNCON, temp);
795 }
796
797 void set_encrypt(void)
798 {
799     int8 temp = 0;
800     temp = read_ZIGBEE_short(TXNCON);
801     temp = temp | 0x02; // mask for set encrypt
802     write_ZIGBEE_short(TXNCON, temp);
803 }
804
805 void set_not_encrypt(void)
806 {
807     int8 temp = 0;

```



```

808     temp = read_ZIGBEE_short(TXNCON);
809     temp = temp & (!0x02); // mask for set not
encrypt
810     write_ZIGBEE_short(TXNCON, temp);
811 }
812
813 void start_transmit(void)
814 {
815     int8 temp = 0;
816     temp = read_ZIGBEE_short(TXNCON);
817     temp = temp | 0x01; // mask for start transmit
818     write_ZIGBEE_short(TXNCON, temp);
819 }
820
821 // interframe spacing
822 void set_IFS_recomended(void)
823 {
824     int8 temp = 0;
825     write_ZIGBEE_short(RXMCR, 0x93); // Min SIFS Period
826     temp = read_ZIGBEE_short(TXPEND);
827     temp = temp | 0x7C; // MinLIFSPeriod
828     write_ZIGBEE_short(TXPEND, temp);
829     temp = read_ZIGBEE_short(TXSTBL);
830     temp = temp | 0x90; // MinLIFSPeriod
831     write_ZIGBEE_short(TXSTBL, temp);
832     temp = read_ZIGBEE_short(TXTIME);
833     temp = temp | 0x31; // TurnaroundTime
834     write_ZIGBEE_short(TXTIME, temp);
835 }
836
837 void set_IFS_default(void)
838 {
839     int8 temp = 0;
840     write_ZIGBEE_short(RXMCR, 0x75); // Min SIFS Period
841     temp = read_ZIGBEE_short(TXPEND);
842     temp = temp | 0x84; // Min LIFS Period
843     write_ZIGBEE_short(TXPEND, temp);
844     temp = read_ZIGBEE_short(TXSTBL);
845     temp = temp | 0x50; // Min LIFS Period
846     write_ZIGBEE_short(TXSTBL, temp);
847     temp = read_ZIGBEE_short(TXTIME);
848     temp = temp | 0x41; // Turnaround Time
849     write_ZIGBEE_short(TXTIME, temp);
850 }
851
852 void set_reception_mode(int8 r_mode) // 1 normal, 2 error, 3
promiscuous mode
853 {
854     int8 temp = 0;
855     switch(r_mode)
856     {
857     case 1:
858     {
859         temp = read_ZIGBEE_short(RXMCR); // normal mode
860         temp = temp & (!0x03); // mask for normal mode
861         write_ZIGBEE_short(RXMCR, temp);
862     }
863     break;

```

```

864         case 2:
865         {
866             temp = read_ZIGBEE_short(RXMCR);          // error mode
867             temp = temp & (!0x01);                    // mask for error mode
868             temp = temp | 0x02;                        // mask for error mode
869             write_ZIGBEE_short(RXMCR, temp);
870         }
871         break;
872         case 3:
873         {
874             temp = read_ZIGBEE_short(RXMCR);          // promiscuous mode
875             temp = temp & (!0x02);                    // mask for promiscuous
mode
876             temp = temp | 0x01;                        // mask for promiscuous
mode
877             write_ZIGBEE_short(RXMCR, temp);
878         }
879         break;
880     }
881 }
882
883 void set_frame_format_filter(int8 fff_mode)           // 1 all frames, 2
command only, 3 data only, 4 beacon only
884 {
885     int8 temp = 0;
886     switch(fff_mode)
887     {
888         case 1:
889         {
890             temp = read_ZIGBEE_short(RXFLUSH);        // all frames
891             temp = temp & (!0x0E);                    // mask for all frames
892             write_ZIGBEE_short(RXFLUSH, temp);
893         }
894         break;
895         case 2:
896         {
897             temp = read_ZIGBEE_short(RXFLUSH);        // command only
898             temp = temp & (!0x06);                    // mask for command
only
899             temp = temp | 0x08;                        // mask for command
only
900             write_ZIGBEE_short(RXFLUSH, temp);
901         }
902         break;
903         case 3:
904         {
905             temp = read_ZIGBEE_short(RXFLUSH);        // data only
906             temp = temp & (!0x0A);                    // mask for data only
907             temp = temp | 0x04;                        // mask for data only
908             write_ZIGBEE_short(RXFLUSH, temp);
909         }
910         break;
911         case 4:
912         {
913             temp = read_ZIGBEE_short(RXFLUSH);        // beacon only
914             temp = temp & (!0x0C);                    // mask for beacon
only

```

```

915         temp = temp | 0x02;                                // mask for beacon
          only
916         write_ZIGBEE_short(RXFLUSH, temp);
917     }
918     break;
919 }
920 }
921
922 void flush_RX_FIFO_pointer(void)
923 {
924     int8 temp;
925     temp = read_ZIGBEE_short(RXFLUSH);
926     temp = temp | 0x01;                                // mask for flush RX FIFO
927     write_ZIGBEE_short(RXFLUSH, temp);
928 }
929
930 void read_RX_FIFO(void)
931 {
932     unsigned int8 temp = 0;
933     int8 i = 0;
934     temp = read_ZIGBEE_short(BBREG1);                    // disable receiving
    packets off air.
935     temp = temp | 0x04;                                // mask for disable
    receiving packets
936     write_ZIGBEE_short(BBREG1, temp);
937     for(i=0; i<128; i++)
938     {
939         if(i < (1 + DATA LENGHT + HEADER LENGHT + 2 + 1 + 1))
940             data_RX_FIFO[i] = read_ZIGBEE_long(address_RX_FIFO + i);
    // reading valid data from RX FIFO
941         if(i >= (1 + DATA LENGHT + HEADER LENGHT + 2 + 1 + 1))
942             lost_data = read_ZIGBEE_long(address_RX_FIFO + i);
    // reading invalid data from RX FIFO
943     }
944     for(i=0; i</*DATA LENGHT*/1; i++)
945     {
946         DATA_RX[i] = data_RX_FIFO[HEADER LENGHT + 1 + i];
    // coping valid data (i.e., End Point address)
947     }
948     LQI = data_RX_FIFO[1 + HEADER LENGHT + DATA LENGHT + 2]; //
    coping valid data
949     RSSI2 = data_RX_FIFO[1 + HEADER LENGHT + DATA LENGHT + 3]; //
    coping valid data
950
951     temp = read_ZIGBEE_short(BBREG1);                    // enable receiving
    packets off air.
952     temp = temp & (!0x04);                                // mask for enable
    receiving
953     write_ZIGBEE_short(BBREG1, temp);
954 }
955
956 void write_TX_normal_FIFO(void)
957 {
958     int8 i = 0;
959     data_TX_normal_FIFO[0] = HEADER LENGHT;
960     data_TX_normal_FIFO[1] = HEADER LENGHT + DATA LENGHT;
961     data_TX_normal_FIFO[2] = 0x01;                                //
    control frame

```

```

962     data_TX_normal_FIFO[3] = 0x88;
963     data_TX_normal_FIFO[4] = SEQ_NUMBER; //
sequence number
964     data_TX_normal_FIFO[5] = PAN_ID_2[1]; //
destinatoin pan
965     data_TX_normal_FIFO[6] = PAN_ID_2[0];
966     data_TX_normal_FIFO[7] = ADDRESS_short_2[0]; //
destination address
967     data_TX_normal_FIFO[8] = ADDRESS_short_2[1];
968     data_TX_normal_FIFO[9] = PAN_ID_1[0]; //
source pan
969     data_TX_normal_FIFO[10] = PAN_ID_1[1];
970     data_TX_normal_FIFO[11] = ADDRESS_short_1[0]; //
source address
971     data_TX_normal_FIFO[12] = ADDRESS_short_1[1];
972     for(i=0; i<DATA LENGHT; i++)
973     {
974         data_TX_normal_FIFO[13 + i] = DATA_TX[i]; //
create data
975     }
976     for(i = 0; i < (HEADER LENGHT + DATA LENGHT + 2); i++)
977     {
978         write_ZIGBEE_long(address_TX_normal_FIFO + i,
data_TX_normal_FIFO[i]); // write frame into normal FIFO
979     }
980     set_not_ACK();
981     set_not_encrypt();
982     start_transmit();
983 }
984
985 void set_short_address(int8 * address)
986 {
987     write_ZIGBEE_short(SADRL, address[0]);
988     write_ZIGBEE_short(SADRH, address[1]);
989 }
990
991 void set_long_address(int8 * address)
992 {
993     int8 i = 0;
994     for(i = 0; i < 8; i++)
995     {
996         write_ZIGBEE_short(EADR0 + i, address[i]); // 0x05 address of
EADR0
997     }
998 }
999
1000 void set_PAN_ID(int8 * address)
1001 {
1002     write_ZIGBEE_short(PANIDL, address[0]);
1003     write_ZIGBEE_short(PANIDH, address[1]);
1004 }
1005
1006 void set_WAKE_from_pin(void)
1007 {
1008     int8 temp = 0;
1009     output_low(MRF24J40_WAKE);
1010     temp = read_ZIGBEE_short(RXFLUSH);
1011     temp = temp | 0x60; // mask

```

```

1012     write_ZIGBEE_short(RXFLUSH, temp);
1013     temp = read_ZIGBEE_short(WAKECON);
1014     temp = temp | 0x80;
1015     write_ZIGBEE_short(WAKECON, temp);
1016 }
1017
1018 void pin_WAKE(void)
1019 {
1020     output_high(MRF24J40_WAKE);
1021     Delay_ms(5);
1022 }
1023
1024 void enable_PLL(void)
1025 {
1026     write_ZIGBEE_long(RFCON2, 0x80);          // mask for PLL enable
1027 }
1028
1029 void disable_PLL(void)
1030 {
1031     write_ZIGBEE_long(RFCON2, 0x00);          // mask for PLL disable
1032 }
1033
1034 void set_TX_power(unsigned int8 power) // 0-31 possible variants
1035 {
1036     if((power < 0) || (power > 31)) power = 31;
1037     power = 31 - power;                      //0 max, 31
1038     min -> 31 max, 0 min
1039     power = ((power & 0b00011111) << 3) & 0b11111000; //
1040     calculating power
1041     write_ZIGBEE_long(RFCON3, power);
1042 }
1043
1044 void init_ZIGBEE_basic(void)
1045 {
1046     write_ZIGBEE_short(PACON2, 0x98);        // Initialize FIFOEN = 1 and
1047     TXONTS = 0x6
1048     write_ZIGBEE_short(TXSTBL, 0x95);        // Initialize RFSTBL = 0x9
1049     write_ZIGBEE_long(RFCON1, 0x01);        // Initialize VCOOPT = 0x01
1050     enable_PLL();                            // Enable PLL (PLLEN = 1)
1051     write_ZIGBEE_long(RFCON6, 0x90);        // Initialize TXFIL = 1 and
1052     20MRECVR = 1
1053     write_ZIGBEE_long(RFCON7, 0x80);        // Initialize SLPCLKSEL = 0x2
1054     (100 kHz internal oscillator)
1055     write_ZIGBEE_long(RFCON8, 0x10);        // Initialize RFVCO = 1
1056     write_ZIGBEE_long(SLPCON1, 0x21);      // Initialize CLKOUTEN = 1 and
1057     SLPCLKDIV = 0x01
1058 }
1059
1060 void init_ZIGBEE_nonbeacon(void)
1061 {
1062     init_ZIGBEE_basic();
1063     set_CCA_mode(1); // Set CCA mode to ED and set threshold
1064     set_RSSI_mode(2); // RSSI2 mode
1065     enable_interrupt(); // Enables all interrupts
1066     set_channel(11); // Channel 11
1067     RF_reset();
1068 }

```

```
1064 void MRF24J40_init(void)
1065 {
1066     pin_reset();
1067     RF_reset();
1068     set_WAKE_from_pin();
1069     set_long_address(ADDRESS_long_2);
1070     set_short_address(ADDRESS_short_2);
1071     set_PAN_ID(PAN_ID_2);
1072     init_ZIGBEE_nonbeacon();
1073     nonbeacon_PAN_coordinator_device();
1074     set_TX_power(31);
1075     set_frame_format_filter(1);           // 1 all frames, 3 data
                                           frame only
1076     set_reception_mode(1);               // 1 normal mode
1077     pin_WAKE();
1078
1079 }
```

3. Συσκευή μέτρησης 02

EndDevice.c

```

1  #include "EndDevice.h"
2  #include "USB_bootloader.h"
3  #include "MRF24J40.h"
4  #include <string.h>
5  #include <math.h>
6  #include "LPS331AP.h"
7
8  #PIN_SELECT INT3=PIN_B4 // Peripheral Pin Select (EXT INT3)
9
10 #INT_EXT3
11 void ext_isr(void)
12 {
13     read_ZIGBEE_short(INTSTAT); // Read and flush register INTSTAT
14     read_RX_FIFO();             // Read 32 bytes and store them to
DATA_RX array
15     if (DATA_RX[0] == EndDevice_Address_1) // CONVERT command for
all when Coordinator requests data from End Device 1
16     {
17         LPS331AP_convert();           // Read Pressure in
bar e.g. <0.993214 bar'\0'>;
18         DATA_RX[0] = 0;              // Reset End Point
transfer to ensure the subsequent reception
19     }
20     if (DATA_RX[0] == EndDevice_Address_2) //REPLY to
Coordinator if EndDevice_Address match
21     {
22         DATA_TX[0] = Coordinator_Address;
23         strcat (DATA_TX, strPressure_Temperature); // Copy
Pressure value to transmit array
24         read_ZIGBEE_short(INTSTAT); // Read and flush
register INTSTAT; clear receive interrupt
25         write_TX_normal_FIFO();      // Transmit Pressure
data to coordinator
26         DATA_RX[0] = 0;              // Reset End Point
transfer to ensure the subsequent reception
27     }
28     clear_interrupt(INT_EXT3);
29 }
30
31 void main()
32 {
33     RBPU = 0; //Enable port_b_pullups when defined inputs (PB4 is
defined as EXT INT3)
34
35     // EXT INT3 Initialization
36     ext_int_edge(3,H_TO_L);
37     clear_interrupt(INT_EXT3);
38     enable_interrupts(INT_EXT3);
39     enable_interrupts(GLOBAL);
40
41     // MRF24J40 IEEE 802.15.4 radio Initialization
42     output_high(MRF24J40_CS);
43     output_high(MRF24J40_RST);
44     MRF24J40_init();
45

```

```
46     DATA_TX[0] = 0;
47     output_high(PIN_A2); // Turn off bootloader's LED (Power consumption
consideration)
48
49     while (1)
50     {
51         sleep(); //Deep Sleep mode can wake-up from INT0 only and also
processor will start execution from the
52                 //Power-on Reset vector rather than branch to the
interrupt vector.
53                 //Therefore, the microcontroller is entered in regular
Sleep mode
54     }
55 }
```


EndDevice.h

```

56  #include <18F26J50.h>
57  #device PASS_STRINGS = IN_RAM //allow pointers to constants; forces
    the compiler to copy the constant strings into an internal RAM buffer
58  #device ADC=16
59
60  #FUSES NOWDT                      //No Watch Dog Timer
61  #FUSES WDT128                    //Watch Dog Timer uses 1:128 Postscale
62  #FUSES NOXINST                    //Extended set extension and Indexed
    Addressing mode disabled (Legacy mode)
63  #FUSES NOT1DIG                    //Secondary Oscillator Source may be
    select if T1CON.3 = 1
64  #FUSES NOFCMEN                    //Fail-safe clock monitor disabled
65  #FUSES NOIESO                     //Internal External Switch Over mode
    disabled
66  #FUSES RTCOSC_INT                 //RTCC uses Internal 31KHz Oscillator
    as reference source
67
68  #use delay(internal=8000000,USB_FULL)
69
70  // MRF24J40 address
71  #define Coordinator_Address 0
72  #define EndDevice_Address_1 1
73  #define EndDevice_Address_2 2
74
75  // Pin & SPI clock definition for MRF24J40
76  #define MRF24J40_HalfPeriod 1
77  #define MRF24J40_CS PIN_A5
78  #define MRF24J40_MOSI PIN_B3
79  #define MRF24J40_MISO PIN_B2
80  #define MRF24J40_SCLK PIN_B1
81  #define MRF24J40_WAKE PIN_C7 //RXD on OLIMEX board
82  #define MRF24J40_RST PIN_C6  //TXD on OLIMEX board
83  #define MRF24J40_INT PIN_B4
84
85  // I2C definition for LPS331AP
86  #define LSP331AP_SDA PIN_c1
87  #define LSP331AP_SCL PIN_c0
88
89  // porb_b_pullup enable registers
90  #byte INTCON2 = 0xFF1
91  #bit  RBPB   = INTCON2.7

```

LPS331AP.h

```

92 // LPS331AP variable/constant definitions
93 #define LSP331AP_I2CdelayUS 1
94 char strPressure_bar[9]; // Example of string pressure
value: 0.998365'\0'
95 char strTemperature_celsius[5]; // Example of string temperature
value: 25.4'\0'
96 char strPressure_Temperature[16]; // Transmit 23 bytes
97
98 // LPS331AP routine definitions
99 void LPS331AP_I2Cdelay(void);
100 void LSP331AP_I2Cstart(void);
101 void LSP331AP_I2Cstop(void);
102 int8 LSP331AP_I2Cwrite(int8 value);
103 int8 LSP331AP_I2Cread(int8 value);
104 int8 LPS331AP_convert(int8);
105
106 // LPS331AP I2C routine
107 void LPS331AP_I2Cdelay(void)
108 {
109     delay_us(LSP331AP_I2CdelayUS);
110 }
111
112 void LSP331AP_I2Cstart(void)
113 {
114     LPS331AP_I2Cdelay();
115     LPS331AP_I2Cdelay();
116     output_float(LSP331AP_SCL);
117     LPS331AP_I2Cdelay();
118     output_low(LSP331AP_SDA);
119     LPS331AP_I2Cdelay();
120     output_low(LSP331AP_SCL);
121 }
122
123 void LSP331AP_I2Cstop(void)
124 {
125     output_low(LSP331AP_SDA);
126     LPS331AP_I2Cdelay();
127     output_float(LSP331AP_SCL);
128     LPS331AP_I2Cdelay();
129     output_float(LSP331AP_SDA);
130 }
131
132 int8 LSP331AP_I2Cwrite(int8 value)
133 {
134     int8 ack=0, i=0;
135     for (i=0;i<9;i++)
136     {
137         if (i<8)
138         {
139             if(value & 0x80)
140             {
141                 output_float(LSP331AP_SDA);
142             }
143             else
144             {
145                 output_low(LSP331AP_SDA);

```

```
146     }
147     LPS331AP_I2Cdelay();
148     output_float(LSP331AP_SCL);
149     LPS331AP_I2Cdelay();
150     output_low(LSP331AP_SCL);
151     value = value <<1;
152     if (i==7)
153     {
154         output_float(LSP331AP_SDA);
155         LPS331AP_I2Cdelay();
156     }
157 }
158 else
159 {
160     output_float(LSP331AP_SCL);
161     ack = input(LSP331AP_SDA);
162     LPS331AP_I2Cdelay();
163     output_low(LSP331AP_SCL);
164     LPS331AP_I2Cdelay();
165 }
166 }
167 return ack;
168 }
169
170 int8 LSP331AP_I2Cread(int8 value)
171 {
172     int8 I2C_readData=0, i=0;
173     for (i=0;i<8;i++)
174     {
175         if (input(LSP331AP_SDA))
176         {
177             I2C_readData = (I2C_readData|0x01);
178         }
179         else
180         {
181             I2C_readData = (I2C_readData&0xFE);
182         }
183         if(i==7)
184         {
185             I2C_readData = I2C_readData;
186         }
187         else
188         {
189             I2C_readData = I2C_readData << 1;
190         }
191         output_float(LSP331AP_SCL);
192         LPS331AP_I2Cdelay();
193         output_low(LSP331AP_SCL);
194         LPS331AP_I2Cdelay();
195     }
196     if(value == 1)
197     {
198         output_low(LSP331AP_SDA);
199     }
200     else
201     {
202         output_float(LSP331AP_SDA);
203     }
```

```

204     LPS331AP_I2Cdelay();
205     output_float(LSP331AP_SCL);
206     LPS331AP_I2Cdelay();
207     output_low(LSP331AP_SCL);
208     output_float(LSP331AP_SDA);
209     LPS331AP_I2Cdelay();
210     return (I2C_readData);
211 }
212
213 // LPS331AP CONTROL routines
214 void LPS331AP_write(int8 reg, int8 data)
215 {
216     int8 ack = 0;
217     LSP331AP_I2Cstart(); //I2C start condition
218     ack = LSP331AP_I2Cwrite((0x5D<<1) & 0xFE); //send I2C device
address (LSB=0 to write)
219     if (ack==0)
220     {
221         reg = reg & 0x7F; //MSB=0 so that the address remains unchanged
in multiple writes
222         ack = LSP331AP_I2Cwrite(reg); //send I2C write register (if
reg. addr. MSB=0, it performs single write)
223         if(ack==0)
224         {
225             ack = LSP331AP_I2Cwrite(data); //send I2C write value
226         }
227     }
228     if(ack==0)
229     {
230         LSP331AP_I2Cstop();
231     }
232 }
233
234 int8 LPS331AP_read(int8 reg)
235 {
236     int8 RegisterValue = 0;
237     int8 ack = 0;
238     LSP331AP_I2Cstart(); //I2C start condition
239     ack = LSP331AP_I2Cwrite((0x5D<<1) & 0xFE); //send I2C device
address (LSB=0 to write)
240     if(ack==0)
241     {
242         reg = reg & 0x7F; //MSB=0 so that the address remains unchanged
in multiple reads
243         ack = LSP331AP_I2Cwrite(reg); //send register address
244         if(ack==0)
245         {
246             LSP331AP_I2Cstart(); //repeat start condition
247             ack = LSP331AP_I2Cwrite((0x5D<<1) | 0x01); //send I2C reg.
address (LSB=1 to read)
248             if(ack==0)
249             {
250                 RegisterValue = LSP331AP_I2Cread(0); //read byte and send
NACK from master to slave
251                 LSP331AP_I2Cstop(); //send stop
252             }
253         }
254     }

```

```

255     return RegisterValue;
256 }
257
258 void LPS331AP_convert(void)
259 {
260     int1 sign = 0;
261     int8 i;
262     int8 convert = 0;
263     unsigned int8 Pressure_hex[3]; // read 3 bytes for pressure and 5
bytes for temperature as well
264     unsigned int8 Temperature_hex[2]; // read 2 bytes for temperature
265     unsigned int16 Temperature;
266     unsigned int32 Pressure;
267     float32 Pressure_bar, Temperature_celsius;
268
269     LPS331AP_write(0x20,0x80); //CTRL_REG1: active device
270     LPS331AP_write(0x21,0x01); //CTRL_REG2: start a new conversion
271     do
272     {
273         convert = LPS331AP_read(0x21); //wait until CTRL_REG2[bit0] is
cleared (i.e., conversion is finished);
274     } while (convert != 0);
275
276     for(i = 0; i<3; i++)
277     {
278         Pressure_hex[i] = LPS331AP_read(0x28+i);
279     }
280     for(i = 0; i<2; i++)
281     {
282         Temperature_hex[i] = LPS331AP_read(0x2B+i);
283     }
284     // do the appropriate math conversions
285     Pressure = make32(Pressure_hex[2],Pressure_hex[1],Pressure_hex[0]);
286     Pressure_bar = (float)Pressure;
287     Pressure_bar = Pressure_bar / 4096 / 1000;
288     sprintf(strPressure_bar,"%1.6f",Pressure_bar);
289     Temperature = make16(Temperature_hex[1],Temperature_hex[0]);
290     if ( (Temperature & 0x8000) != 0 )
291     {
292         sign = 1;
293         Temperature = (~Temperature)+1; //two's complement notation 2^N-
Tadc (i.e., find the corresponding positive and set sign bit)
294     }
295     Temperature_celsius = (float) Temperature;
296     Temperature_celsius = (Temperature_celsius / 480);
297     if (sign)
298     {
299         Temperature_celsius = Temperature_celsius * (-1);
300         sign = 0;
301     }
302     Temperature_celsius = 42.5 + Temperature_celsius;
303     sprintf(strTemperature_celsius,"%2.1f",Temperature_celsius);
304     strPressure_Temperature[0] = strPressure_bar[0]; //
e.g., '0'
305     strPressure_Temperature[1] = strPressure_bar[1]; //
e.g., '.'
306     strPressure_Temperature[2] = strPressure_bar[2]; //
e.g., '9'

```

```

307     strPressure_Temperature[3] = strPressure_bar[3];           //
    e.g., '9'
308     strPressure_Temperature[4] = strPressure_bar[4];           //
    e.g., '8'
309     strPressure_Temperature[5] = strPressure_bar[5];           //
    e.g., '7'
310     strPressure_Temperature[6] = strPressure_bar[6];           //
    e.g., '6'
311     strPressure_Temperature[7] = strPressure_bar[7];           //
    e.g., '5'
312     strPressure_Temperature[8] = ' ';                           //
    e.g., ' '
313     strPressure_Temperature[9] = ' ';                           //
    e.g., ' '
314     strPressure_Temperature[10] = ' ';                          //
    e.g., ' '
315     strPressure_Temperature[11] = strTemperature_celsius[0];    //
    e.g., '2'
316     strPressure_Temperature[12] = strTemperature_celsius[1];    //
    e.g., '5'
317     strPressure_Temperature[13] = strTemperature_celsius[2];    //
    e.g., '.'
318     strPressure_Temperature[14] = strTemperature_celsius[3];    //
    e.g., '4'
319     strPressure_Temperature[15] = '\0';                          //
    e.g., '\0'
320     //usb_puts(1, strPressure_bar, 13, 0); //2*bytes, 0);
321     LPS331AP_write(0x20,0x00); //CTRL_REG1: deactivate device
322 }

```

MRF24J40.h

```

323 // MRF24J40 Short Register Addresses
324 #define RXMCR      0x00
325 #define PANIDL     0x01
326 #define PANIDH     0x02
327 #define SADRL      0x03
328 #define SADRH      0x04
329 #define EADR0      0x05
330 #define EADR1      0x06
331 #define EADR2      0x07
332 #define EADR3      0x08
333 #define EADR4      0x09
334 #define EADR5      0x0A
335 #define EADR6      0x0B
336 #define EADR7      0x0C
337 #define RXFLUSH    0x0D
338 #define ORDER      0x10
339 #define TXMCR      0x11
340 #define ACKTMOUT    0x12
341 #define ESLOTG1     0x13
342 #define SYMTICKL    0x14
343 #define SYMTICKH    0x15
344 #define PACON0     0x16
345 #define PACON1     0x17
346 #define PACON2     0x18
347 #define TXBCON0    0x1A
348 #define TXNCON     0x1B
349 #define TXG1CON    0x1C
350 #define TXG2CON    0x1D
351 #define ESLOTG23    0x1E
352 #define ESLOTG45    0x1F
353 #define ESLOTG67    0x20
354 #define TXPEND     0x21
355 #define WAKECON     0x22
356 #define FRMOFFSET   0x23
357 #define TXSTAT      0x24
358 #define TXBCON1    0x25
359 #define GATECLK     0x26
360 #define TXTIME      0x27
361 #define HSYMTMRL    0x28
362 #define HSYMTMRH    0x29
363 #define SOFTRST     0x2A
364 #define SECCON0     0x2C
365 #define SECCON1     0x2D
366 #define TXSTBL      0x2E
367 #define RXSR        0x30
368 #define INTSTAT     0x31
369 #define INTCON_M    0x32
370 #define GPIO        0x33
371 #define TRISGPIO    0x34
372 #define SLPACK      0x35
373 #define RFCTL       0x36
374 #define SECCR2      0x37
375 #define BBREG0      0x38
376 #define BBREG1      0x39
377 #define BBREG2      0x3A
378 #define BBREG3      0x3B

```

```

379 #define BBREG4      0x3C
380 #define BBREG6      0x3E
381 #define CCAEDTH      0x3F
382 // MRF24J40 Long Register Addresses
383 #define RFCON0       0x200
384 #define RFCON1       0x201
385 #define RFCON2       0x202
386 #define RFCON3       0x203
387 #define RFCON5       0x205
388 #define RFCON6       0x206
389 #define RFCON7       0x207
390 #define RFCON8       0x208
391 #define SLPCAL0      0x209
392 #define SLPCAL1      0x20A
393 #define SLPCAL2      0x20B
394 #define RFSTATE      0x20F
395 #define RSSI         0x210
396 #define SLPCON0      0x211
397 #define SLPCON1      0x220
398 #define WAKETIME1    0x222
399 #define WAKETIMEH    0x223
400 #define REMCNTL      0x224
401 #define REMCNTH      0x225
402 #define MAINCNT0     0x226
403 #define MAINCNT1     0x227
404 #define MAINCNT2     0x228
405 #define MAINCNT3     0x229
406 #define ASSOEADR0    0x230
407 #define ASSOEADR1    0x231
408 #define ASSOEADR2    0x232
409 #define ASSOEADR3    0x233
410 #define ASSOEADR4    0x234
411 #define ASSOEADR5    0x235
412 #define ASSOEADR6    0x236
413 #define ASSOEADR7    0x237
414 #define ASSOSADR0    0x238
415 #define ASSOSADR1    0x239
416 #define UPNONCE0     0x240
417 #define UPNONCE1     0x241
418 #define UPNONCE2     0x242
419 #define UPNONCE3     0x243
420 #define UPNONCE4     0x244
421 #define UPNONCE5     0x245
422 #define UPNONCE6     0x246
423 #define UPNONCE7     0x247
424 #define UPNONCE8     0x248
425 #define UPNONCE9     0x249
426 #define UPNONCE10    0x24A
427 #define UPNONCE11    0x24B
428 #define UPNONCE12    0x24C
429
430 // Global variables/constants for MRF24J40
431 int8 LQI = 0, RSSI2 = 0, SEQ_NUMBER = 0x23;
432 int16 address_RX_FIFO = 0x300;
433 int16 address_TX_normal_FIFO = 0x000;
434 int8 ADDRESS_short_1[] = {1,1}; // Source address
435 int8 ADDRESS_short_2[] = {2,2}; // Destination address
436 int8 PAN_ID_1[] = {3,3}; // Source PAN ID

```



```

437  int8 PAN_ID_2[] = {3,3}; // Destination PAN ID
438  int8 ADDRESS_long_1[] = {1,1,1,1,1,1,1,1};
439  int8 ADDRESS_long_2[] = {2,2,2,2,2,2,2,2};
440  const unsigned int8 DATA LENGHT = 32; // ### define here the
    number of transmitted data ###
441  const unsigned int8 HEADER LENGHT = 11;
442  int8 data_RX_FIFO[1 + HEADER LENGHT + DATA LENGHT + 2 + 1 + 1],
443      lost_data = 0;
444  int8 DATA_RX[DATA LENGHT],
445      DATA_TX[DATA LENGHT],
446      data_TX_normal_FIFO[DATA LENGHT + HEADER LENGHT + 2];
447
448  // MRF24J40 routines definitions
449  int8 MRF24J40_spiBYTE(int8 data);
450  void write_ZIGBEE_short(int8 address, int8 data_r);
451  int8 read_ZIGBEE_short(int8 address);
452  void write_ZIGBEE_long(int16 address, int8 data_r);
453  int8 read_ZIGBEE_long(int16 address);
454  void pin_reset(void);
455  void PWR_reset(void);
456  void BB_reset(void);
457  void MAC_reset(void);
458  void software_reset(void);
459  void RF_reset(void);
460  void enable_interrupt(void);
461  void set_channel(int8 channel_number);
462  void set_CCA_mode(int8 CCA_mode);
463  void set_RSSI_mode(int8 RSSI_mode);
464  void nonbeacon_PAN_coordinator_device(void);
465  void nonbeacon_coordinator_device(void);
466  void nonbeacon_device(void);
467  void set_ACK(void);
468  void set_not_ACK(void);
469  void set_encrypt(void);
470  void set_not_encrypt(void);
471  void start_transmit(void);
472  void set_IFS_recommended(void);
473  void set_IFS_default(void);
474  void set_reception_mode(int8 r_mode);
475  void set_frame_format_filter(int8 fff_mode);
476  void flush_RX_FIFO_pointer(void);
477  void read_RX_FIFO(void);
478  void write_TX_normal_FIFO(void);
479  void set_short_address(int8 * address);
480  void set_long_address(int8 * address);
481  void set_PAN_ID(int8 * address);
482  void set_WAKE_from_pin(void);
483  void pin_WAKE(void);
484  void enable_PLL(void);
485  void disable_PLL(void);
486  void set_TX_power(unsigned int8 power);
487  void init_ZIGBEE_basic(void);
488  void init_ZIGBEE_nonbeacon(void);
489  void MRF24J40_init(void);
490
491  // MRF24J40 SPI routine
492  int8 MRF24J40_spiBYTE(int8 data)
493  {

```

```

494  int8 i=0;
495  int8 readdata = 0x00;
496  for (i=0;i<8;i++)
497  {
498      if (data & 0x80)
499      {
500          output_high(MRF24J40_MOSI); // define MRF24J40_MOSI
501      }
502      else
503      {
504          output_low(MRF24J40_MOSI); // define MRF24J40_MOSI
505      }
506      delay_us(MRF24J40_HalfPeriod);
507      output_high(MRF24J40_SCLK); //set clock
508      data = data << 1;
509      if(i>0)
510      {
511          readdata = readdata << 1;
512      }
513      if (input(MRF24J40_MISO))
514      {
515
516          readdata = (readdata|0x01);
517      }
518      else
519      {
520          readdata = (readdata&0xFE);
521      }
522      delay_us(MRF24J40_HalfPeriod);
523      output_low(MRF24J40_SCLK); //reset clock
524  }
525  output_low(MRF24J40_MOSI);
526  return (readdata);
527 }
528
529 // W5100 CONTROL routines
530 void write_ZIGBEE_short(int8 address, int8 data_r)
531 {
532     output_low(MRF24J40_CS); //MRF24J40_CS = 0;
533     delay_us(MRF24J40_HalfPeriod);
534     address = ((address << 1) & 0b01111111) | 0x01; // calculating
535     addressing mode
536     MRF24J40_spiBYTE(address); // addressing register
537     MRF24J40_spiBYTE(data_r); // write data in register
538     delay_us(MRF24J40_HalfPeriod);
539     output_high(MRF24J40_CS); //;MRF24J40_CS = 1;
540 }
541
542 int8 read_ZIGBEE_short(int8 address)
543 {
544     int8 data_r = 0, dummy_data_r = 0;
545     output_low(MRF24J40_CS); //MRF24J40_CS = 0;
546     delay_us(MRF24J40_HalfPeriod);
547     address = (address << 1) & 0b01111110; // calculating
548     addressing mode
549     MRF24J40_spiBYTE(address); // addressing
550     register

```

```

549     data_r = MRF24J40_spiBYTE(dummy_data_r);           // read data
        from register
550     delay_us(MRF24J40_HalfPeriod);
551     output_high(MRF24J40_CS); // MRF24J40_CS = 1;
552     return data_r;
553 }
554
555 void write_ZIGBEE_long(int16 address, int8 data_r)
556 {
557     int8 address_high = 0, address_low = 0;
558     output_low(MRF24J40_CS); //MRF24J40_CS = 0;
559     delay_us(MRF24J40_HalfPeriod);
560     address_high = (((int8)(address >> 3)) & 0b01111111) | 0x80; //
calculating addressing mode
561     address_low = (((int8)(address << 5)) & 0b11100000) | 0x10; //
calculating addressing mode
562     MRF24J40_spiBYTE(address_high);           // addressing register
563     MRF24J40_spiBYTE(address_low);           // addressing register
564     MRF24J40_spiBYTE(data_r);                 // write data in
register
565     delay_us(MRF24J40_HalfPeriod);
566     output_high(MRF24J40_CS); //MRF24J40_CS = 1;
567 }
568
569 int8 read_ZIGBEE_long(int16 address)
570 {
571     int8 data_r = 0, dummy_data_r = 0;
572     int8 address_high = 0, address_low = 0;
573     output_low(MRF24J40_CS);
574     delay_us(MRF24J40_HalfPeriod);
575     address_high = (((int8)(address >> 3)) & 0b01111111) | 0x80;
//calculating addressing mode
576     address_low = (((int8)(address << 5)) & 0b11100000);
//calculating addressing mode
577     MRF24J40_spiBYTE(address_high);           // addressing register
578     MRF24J40_spiBYTE(address_low);           // addressing register
579     data_r = MRF24J40_spiBYTE(dummy_data_r); // read data from
register
580     delay_us(MRF24J40_HalfPeriod);
581     output_high(MRF24J40_CS);
582     return data_r;
583 }
584
585 void pin_reset(void)
586 {
587     output_low(MRF24J40_RST); // activate reset
588     delay_ms(5);
589     output_high(MRF24J40_RST); // deactivate reset
590     delay_ms(5);
591 }
592
593 void PWR_reset(void)
594 {
595     write_ZIGBEE_short(SOFTTRST, 0x04);           //0x04    mask    for
MRF24J40_RSTPWR bit
596 }
597
598 void BB_reset(void)

```

```

599 {
600     write_ZIGBEE_short(SOFTTRST, 0x02);           //0x02 mask for
MRF24J40_RSTBB bit
601 }
602
603 void MAC_reset(void)
604 {
605     write_ZIGBEE_short(SOFTTRST, 0x01);           //0x01 mask for
MRF24J40_RSTMAC bit
606 }
607
608 void software_reset(void) // PWR_reset,BB_reset and MAC_reset at once
609 {
610     write_ZIGBEE_short(SOFTTRST, 0x07);
611 }
612
613 void RF_reset(void)
614 {
615     int8 temp = 0;
616     temp = read_ZIGBEE_short(RFCTL);
617     temp = temp | 0x04;                             //mask for RFMRF24J40_RST
bit
618     write_ZIGBEE_short(RFCTL, temp);
619     temp = temp & (!0x04);                         //mask for RFMRF24J40_RST
bit
620     write_ZIGBEE_short(RFCTL, temp);
621     delay_ms(1);
622 }
623
624 void enable_interrupt(void)
625 {
626     write_ZIGBEE_short(INTCON_M, 0x01);           //all interrupts are enable
except TX Normal FIFO transmission interrupt
627                                     //when TX Normal FIFO
transmission interrupt INT pin goes low for long time and transmission
is repeated
628 }
629
630 void set_channel(int8 channel_number) // 11-26 possible channels
631 {
632     if((channel_number > 26) || (channel_number < 11)) channel_number =
11;
633     switch(channel_number)
634     {
635         case 11:
636             write_ZIGBEE_long(RFCON0, 0x02); //0x02 for 11. channel
637             break;
638         case 12:
639             write_ZIGBEE_long(RFCON0, 0x12); //0x12 for 12. channel
640             break;
641         case 13:
642             write_ZIGBEE_long(RFCON0, 0x22); //0x22 for 13. channel
643             break;
644         case 14:
645             write_ZIGBEE_long(RFCON0, 0x32); //0x32 for 14. channel
646             break;
647         case 15:
648             write_ZIGBEE_long(RFCON0, 0x42); //0x42 for 15. channel

```

```

649         break;
650     case 16:
651         write_ZIGBEE_long(RFCON0, 0x52); //0x52 for 16. channel
652         break;
653     case 17:
654         write_ZIGBEE_long(RFCON0, 0x62); //0x62 for 17. channel
655         break;
656     case 18:
657         write_ZIGBEE_long(RFCON0, 0x72); //0x72 for 18. channel
658         break;
659     case 19:
660         write_ZIGBEE_long(RFCON0, 0x82); //0x82 for 19. channel
661         break;
662     case 20:
663         write_ZIGBEE_long(RFCON0, 0x92); //0x92 for 20. channel
664         break;
665     case 21:
666         write_ZIGBEE_long(RFCON0, 0xA2); //0xA2 for 21. channel
667         break;
668     case 22:
669         write_ZIGBEE_long(RFCON0, 0xB2); //0xB2 for 22. channel
670         break;
671     case 23:
672         write_ZIGBEE_long(RFCON0, 0xC2); //0xC2 for 23. channel
673         break;
674     case 24:
675         write_ZIGBEE_long(RFCON0, 0xD2); //0xD2 for 24. channel
676         break;
677     case 25:
678         write_ZIGBEE_long(RFCON0, 0xE2); //0xE2 for 25. channel
679         break;
680     case 26:
681         write_ZIGBEE_long(RFCON0, 0xF2); //0xF2 for 26. channel
682         break;
683     }
684     RF_reset();
685 }
686
687 void set_CCA_mode(int8 CCA_mode)
688 {
689     int8 temp = 0;
690     switch(CCA_mode)
691     {
692         case 1: //ENERGY ABOVE THRESHOLD
693         {
694             temp = read_ZIGBEE_short(BBREG2);
695             temp = temp | 0x80; //0x80 mask
696             temp = temp & 0xDF; //0xDF mask
697             write_ZIGBEE_short(BBREG2, temp);
698
699             write_ZIGBEE_short(CCAEDTH, 0x60); //Set CCA ED
700             threshold to -69 dBm
701         }
702         break;
703         case 2: //CARRIER SENSE ONLY
704         {
705             temp = read_ZIGBEE_short(BBREG2);
706             temp = temp | 0x40; // 0x40 mask

```

```

706         temp = temp & 0x7F;           // 0x7F mask
707         write_ZIGBEE_short(BBREG2, temp);
708
709         temp = read_ZIGBEE_short(BBREG2);    // carrier sense
threshold
710         temp = temp | 0x38;
711         temp = temp & 0xFB;
712         write_ZIGBEE_short(BBREG2, temp);
713     }
714     break;
715     case 3:                               //CARRIER SENSE AND ENERGY ABOVE
THRESHOLD
716     {
717         temp = read_ZIGBEE_short(BBREG2);
718         temp = temp | 0xC0;               //0xC0 mask
719         write_ZIGBEE_short(BBREG2, temp);
720
721         temp = read_ZIGBEE_short(BBREG2);    // carrier sense
threshold
722         temp = temp | 0x38;               // 0x38 mask
723         temp = temp & 0xFB;               // 0xFB mask
724         write_ZIGBEE_short(BBREG2, temp);
725
726         write_ZIGBEE_short(CCAEDTH, 0x60);    //Set CCA ED
threshold to -69 dBm
727     }
728     break;
729 }
730 }
731
732 void set_RSSI_mode(int8 RSSI_mode) // 1 for RSSI1, 2 for RSSI2 mode
733 {
734     int8 temp = 0;
735     switch(RSSI_mode)
736     {
737         case 1:
738         {
739             temp = read_ZIGBEE_short(BBREG6);
740             temp = temp | 0x80;           //0x80 mask for RSSI1 mode
741             write_ZIGBEE_short(BBREG6, temp);
742         }
743         break;
744         case 2:
745             write_ZIGBEE_short(BBREG6, 0x40);    //0x40 data for RSSI2 mode
746             break;
747     }
748 }
749
750 void nonbeacon_PAN_coordinator_device(void)
751 {
752     int8 temp = 0;
753     temp = read_ZIGBEE_short(RXMCR);
754     temp = temp | 0x08;                 // 0x08 mask for PAN
coordinator
755     write_ZIGBEE_short(RXMCR, temp);
756     temp = read_ZIGBEE_short(TXMCR);
757     temp = temp & 0xDF;                 // 0xDF mask for
MRF24J40_CSMA-CA mode

```

```

758     write_ZIGBEE_short(TXMCR, temp);
759     write_ZIGBEE_short(ORDER, 0xFF);    // BO, SO are 15
760 }
761
762 void nonbeacon_coordinator_device(void)
763 {
764     int8 temp = 0;
765     temp = read_ZIGBEE_short(RXMCR);
766     temp = temp | 0x04;                // 0x04 mask for coordinator
767     write_ZIGBEE_short(RXMCR, temp);
768     temp = read_ZIGBEE_short(TXMCR);
769     temp = temp & 0xDF;                // 0xDF mask for
MRF24J40_CSMA-CA mode
770     write_ZIGBEE_short(TXMCR, temp);
771     write_ZIGBEE_short(ORDER, 0xFF);    // BO, SO are 15
772 }
773
774 void nonbeacon_device(void)
775 {
776     int8 temp = 0;
777     temp = read_ZIGBEE_short(RXMCR);
778     temp = temp & 0xF3;                // 0xF3 mask for PAN coordinator
and coordinator
779     write_ZIGBEE_short(RXMCR, temp);
780     temp = read_ZIGBEE_short(TXMCR);
781     temp = temp & 0xDF;                // 0xDF mask for MRF24J40_CSMA-
CA mode
782     write_ZIGBEE_short(TXMCR, temp);
783 }
784
785 void set_ACK(void)
786 {
787     int8 temp = 0;
788     temp = read_ZIGBEE_short(TXNCON);
789     temp = temp | 0x04;                // 0x04 mask for set ACK
790     write_ZIGBEE_short(TXNCON, temp);
791 }
792
793 void set_not_ACK(void)
794 {
795     int8 temp = 0;
796     temp = read_ZIGBEE_short(TXNCON);
797     temp = temp & (!0x04);            // 0x04 mask for set not
ACK
798     write_ZIGBEE_short(TXNCON, temp);
799 }
800
801 void set_encrypt(void)
802 {
803     int8 temp = 0;
804     temp = read_ZIGBEE_short(TXNCON);
805     temp = temp | 0x02;                // mask for set encrypt
806     write_ZIGBEE_short(TXNCON, temp);
807 }
808
809 void set_not_encrypt(void)
810 {
811     int8 temp = 0;

```

```

812     temp = read_ZIGBEE_short(TXNCON);
813     temp = temp & (!0x02); // mask for set not
encrypt
814     write_ZIGBEE_short(TXNCON, temp);
815 }
816
817 void start_transmit(void)
818 {
819     int8 temp = 0;
820     temp = read_ZIGBEE_short(TXNCON);
821     temp = temp | 0x01; // mask for start transmit
822     write_ZIGBEE_short(TXNCON, temp);
823 }
824
825 // interframe spacing
826 void set_IFS_recomended(void)
827 {
828     int8 temp = 0;
829     write_ZIGBEE_short(RXMCR, 0x93); // Min SIFS Period
830     temp = read_ZIGBEE_short(TXPEND);
831     temp = temp | 0x7C; // MinLIFSPeriod
832     write_ZIGBEE_short(TXPEND, temp);
833     temp = read_ZIGBEE_short(TXSTBL);
834     temp = temp | 0x90; // MinLIFSPeriod
835     write_ZIGBEE_short(TXSTBL, temp);
836     temp = read_ZIGBEE_short(TXTIME);
837     temp = temp | 0x31; // TurnaroundTime
838     write_ZIGBEE_short(TXTIME, temp);
839 }
840
841 void set_IFS_default(void)
842 {
843     int8 temp = 0;
844     write_ZIGBEE_short(RXMCR, 0x75); // Min SIFS Period
845     temp = read_ZIGBEE_short(TXPEND);
846     temp = temp | 0x84; // Min LIFS Period
847     write_ZIGBEE_short(TXPEND, temp);
848     temp = read_ZIGBEE_short(TXSTBL);
849     temp = temp | 0x50; // Min LIFS Period
850     write_ZIGBEE_short(TXSTBL, temp);
851     temp = read_ZIGBEE_short(TXTIME);
852     temp = temp | 0x41; // Turnaround Time
853     write_ZIGBEE_short(TXTIME, temp);
854 }
855
856 void set_reception_mode(int8 r_mode) // 1 normal, 2 error, 3
promiscuous mode
857 {
858     int8 temp = 0;
859     switch(r_mode)
860     {
861     case 1:
862     {
863         temp = read_ZIGBEE_short(RXMCR); // normal mode
864         temp = temp & (!0x03); // mask for normal mode
865         write_ZIGBEE_short(RXMCR, temp);
866     }
867     break;

```



```

868         case 2:
869         {
870             temp = read_ZIGBEE_short(RXMCR);           // error mode
871             temp = temp & (!0x01);                     // mask for error mode
872             temp = temp | 0x02;                         // mask for error mode
873             write_ZIGBEE_short(RXMCR, temp);
874         }
875         break;
876         case 3:
877         {
878             temp = read_ZIGBEE_short(RXMCR);           // promiscuous mode
879             temp = temp & (!0x02);                     // mask for promiscuous
mode
880             temp = temp | 0x01;                         // mask for promiscuous
mode
881             write_ZIGBEE_short(RXMCR, temp);
882         }
883         break;
884     }
885 }
886
887 void set_frame_format_filter(int8 fff_mode)           // 1 all frames, 2
command only, 3 data only, 4 beacon only
888 {
889     int8 temp = 0;
890     switch(fff_mode)
891     {
892         case 1:
893         {
894             temp = read_ZIGBEE_short(RXFLUSH);         // all frames
895             temp = temp & (!0x0E);                     // mask for all frames
896             write_ZIGBEE_short(RXFLUSH, temp);
897         }
898         break;
899         case 2:
900         {
901             temp = read_ZIGBEE_short(RXFLUSH);         // command only
902             temp = temp & (!0x06);                     // mask for command
only
903             temp = temp | 0x08;                         // mask for command
only
904             write_ZIGBEE_short(RXFLUSH, temp);
905         }
906         break;
907         case 3:
908         {
909             temp = read_ZIGBEE_short(RXFLUSH);         // data only
910             temp = temp & (!0x0A);                     // mask for data only
911             temp = temp | 0x04;                         // mask for data only
912             write_ZIGBEE_short(RXFLUSH, temp);
913         }
914         break;
915         case 4:
916         {
917             temp = read_ZIGBEE_short(RXFLUSH);         // beacon only
918             temp = temp & (!0x0C);                     // mask for beacon
only

```

```

919         temp = temp | 0x02;                                // mask for beacon
only
920         write_ZIGBEE_short(RXFLUSH, temp);
921     }
922     break;
923 }
924 }
925
926 void flush_RX_FIFO_pointer(void)
927 {
928     int8 temp;
929     temp = read_ZIGBEE_short(RXFLUSH);
930     temp = temp | 0x01;                                // mask for flush RX FIFO
931     write_ZIGBEE_short(RXFLUSH, temp);
932 }
933
934 void read_RX_FIFO(void)
935 {
936     unsigned int8 temp = 0;
937     int8 i = 0;
938     temp = read_ZIGBEE_short(BBREG1);                    // disable receiving
packets off air.
939     temp = temp | 0x04;                                // mask for disable
receiving packets
940     write_ZIGBEE_short(BBREG1, temp);
941     for(i=0; i<128; i++)
942     {
943         if(i < (1 + DATA LENGHT + HEADER LENGHT + 2 + 1 + 1))
944             data_RX_FIFO[i] = read_ZIGBEE_long(address_RX_FIFO + i);
// reading valid data from RX FIFO
945         if(i >= (1 + DATA LENGHT + HEADER LENGHT + 2 + 1 + 1))
946             lost_data = read_ZIGBEE_long(address_RX_FIFO + i);
// reading invalid data from RX FIFO
947     }
948     for(i=0; i</*DATA LENGHT*/1; i++)
949     {
950         DATA_RX[i] = data_RX_FIFO[HEADER LENGHT + 1 + i];
// coping valid data (i.e., End Point address)
951     }
952     LQI = data_RX_FIFO[1 + HEADER LENGHT + DATA LENGHT + 2]; //
coping valid data
953     RSSI2 = data_RX_FIFO[1 + HEADER LENGHT + DATA LENGHT + 3]; //
coping valid data
954
955     temp = read_ZIGBEE_short(BBREG1);                    // enable receiving
packets off air.
956     temp = temp & (!0x04);                                // mask for enable
receiving
957     write_ZIGBEE_short(BBREG1, temp);
958 }
959
960 void write_TX_normal_FIFO(void)
961 {
962     int8 i = 0;
963     data_TX_normal_FIFO[0] = HEADER LENGHT;
964     data_TX_normal_FIFO[1] = HEADER LENGHT + DATA LENGHT;
965     data_TX_normal_FIFO[2] = 0x01;                                //
control frame

```

```

966     data_TX_normal_FIFO[3] = 0x88;
967     data_TX_normal_FIFO[4] = SEQ_NUMBER; //
sequence number
968     data_TX_normal_FIFO[5] = PAN_ID_2[1]; //
destinatoin pan
969     data_TX_normal_FIFO[6] = PAN_ID_2[0];
970     data_TX_normal_FIFO[7] = ADDRESS_short_2[0]; //
destination address
971     data_TX_normal_FIFO[8] = ADDRESS_short_2[1];
972     data_TX_normal_FIFO[9] = PAN_ID_1[0]; //
source pan
973     data_TX_normal_FIFO[10] = PAN_ID_1[1];
974     data_TX_normal_FIFO[11] = ADDRESS_short_1[0]; //
source address
975     data_TX_normal_FIFO[12] = ADDRESS_short_1[1];
976     for(i=0; i<DATA LENGHT; i++)
977     {
978         data_TX_normal_FIFO[13 + i] = DATA_TX[i]; //
create data
979     }
980     for(i = 0; i < (HEADER LENGHT + DATA LENGHT + 2); i++)
981     {
982         write_ZIGBEE_long(address_TX_normal_FIFO + i,
data_TX_normal_FIFO[i]); // write frame into normal FIFO
983     }
984     set_not_ACK();
985     set_not_encrypt();
986     start_transmit();
987 }
988
989 void set_short_address(int8 * address)
990 {
991     write_ZIGBEE_short(SADRL, address[0]);
992     write_ZIGBEE_short(SADRH, address[1]);
993 }
994
995 void set_long_address(int8 * address)
996 {
997     int8 i = 0;
998     for(i = 0; i < 8; i++)
999     {
1000         write_ZIGBEE_short(EADR0 + i, address[i]); // 0x05 address of
EADR0
1001     }
1002 }
1003
1004 void set_PAN_ID(int8 * address)
1005 {
1006     write_ZIGBEE_short(PANIDL, address[0]);
1007     write_ZIGBEE_short(PANIDH, address[1]);
1008 }
1009
1010 void set_WAKE_from_pin(void)
1011 {
1012     int8 temp = 0;
1013     output_low(MRF24J40_WAKE);
1014     temp = read_ZIGBEE_short(RXFLUSH);
1015     temp = temp | 0x60; // mask

```

```

1016     write_ZIGBEE_short(RXFLUSH, temp);
1017     temp = read_ZIGBEE_short(WAKECON);
1018     temp = temp | 0x80;
1019     write_ZIGBEE_short(WAKECON, temp);
1020 }
1021
1022 void pin_WAKE(void)
1023 {
1024     output_high(MRF24J40_WAKE);
1025     Delay_ms(5);
1026 }
1027
1028 void enable_PLL(void)
1029 {
1030     write_ZIGBEE_long(RFCON2, 0x80);           // mask for PLL enable
1031 }
1032
1033 void disable_PLL(void)
1034 {
1035     write_ZIGBEE_long(RFCON2, 0x00);           // mask for PLL disable
1036 }
1037
1038 void set_TX_power(unsigned int8 power) // 0-31 possible variants
1039 {
1040     if((power < 0) || (power > 31)) power = 31;
1041     power = 31 - power;                        //0 max, 31
1042     min -> 31 max, 0 min
1043     power = ((power & 0b00011111) << 3) & 0b11111000; //
1044     calculating power
1045     write_ZIGBEE_long(RFCON3, power);
1046 }
1047
1048 void init_ZIGBEE_basic(void)
1049 {
1050     write_ZIGBEE_short(PACON2, 0x98); // Initialize FIFOEN = 1 and
1051     TXONTS = 0x6
1052     write_ZIGBEE_short(TXSTBL, 0x95); // Initialize RFSTBL = 0x9
1053     write_ZIGBEE_long(RFCON1, 0x01); // Initialize VCOOPT = 0x01
1054     enable_PLL(); // Enable PLL (PLLEN = 1)
1055     write_ZIGBEE_long(RFCON6, 0x90); // Initialize TXFIL = 1 and
1056     20MRECVR = 1
1057     write_ZIGBEE_long(RFCON7, 0x80); // Initialize SLPCLKSEL = 0x2
1058     (100 kHz internal oscillator)
1059     write_ZIGBEE_long(RFCON8, 0x10); // Initialize RFVCO = 1
1060     write_ZIGBEE_long(SLPCON1, 0x21); // Initialize CLKOUTEN = 1 and
1061     SLPCLKDIV = 0x01
1062 }
1063
1064 void init_ZIGBEE_nonbeacon(void)
1065 {
1066     init_ZIGBEE_basic();
1067     set_CCA_mode(1); // Set CCA mode to ED and set threshold
1068     set_RSSI_mode(2); // RSSI2 mode
1069     enable_interrupt(); // Enables all interrupts
1070     set_channel(11); // Channel 11
1071     RF_reset();
1072 }
1073

```

```
1068 void MRF24J40_init(void)
1069 {
1070     pin_reset();
1071     RF_reset();
1072     set_WAKE_from_pin();
1073     set_long_address(ADDRESS_long_2);
1074     set_short_address(ADDRESS_short_2);
1075     set_PAN_ID(PAN_ID_2);
1076     init_ZIGBEE_nonbeacon();
1077     nonbeacon_PAN_coordinator_device();
1078     set_TX_power(31);
1079     set_frame_format_filter(1);           // 1 all frames, 3 data
                                           frame only
1080     set_reception_mode(1);               // 1 normal mode
1081     pin_WAKE();
1082
1083 }
```



```

47     break_flag=0;
48     //configuration buffers for ESS112
49     unsigned char LPS331AP_pressure[]={"60\r"};
50     int LPS331AP_length=sizeof(LPS331AP_pressure)/sizeof(unsigned char);
51
52     //open usb port
53     DWORD selection;
54     selection=device; //USB device, e.g. device 0 , device 1, ...,
device 127
55     BYTE send_buf[64],receive_buf[64];
56     DWORD RecvLength; //Points to the number of bytes read.MPUSBRead
sets this value to zero before doing any work or error checking.
57     DWORD SentDataLength; //number of bytes written by this function
call
58
59     // Load DLL when it is necessary, i.e. on start-up!
60     LoadDLL();
61     // Always a good idea to initialize the handles
62     myOutPipe = myInPipe = INVALID_HANDLE_VALUE;
63
64     myOutPipe = MPUSBOpen(selection,vid_pid,out_pipe,MP_WRITE,0);
65     myInPipe = MPUSBOpen(selection,vid_pid,out_pipe,MP_READ,0);
66     if(myOutPipe == INVALID_HANDLE_VALUE || myInPipe ==
INVALID_HANDLE_VALUE)
67     {
68         printf("Failed to open data pipes.\r\n");
69     }
70
71
72     //read pressure and temperature data
73     labviewCnt=0;
74     while (labviewCnt<samples)
75     {
76         dwStart = GetTickCount(); //The resolution of the GetTickCount
function is limited to the resolution of the system timer (typically
10-16 msec).
77         //printf("%d\n", dwStart);
78         labviewCnt++;
79         for (i=0; i<64; ++i)
80         {
81             receive_buf[i]=0;
82         }
83
84         MPUSBWrite(myOutPipe,LPS331AP_pressure,LPS331AP_length-
1,&SentDataLength,40); //last argument is time-out interval in
milliseconds.
85         Sleep(delay_ms);
86
87         //READ Pressure & Temperature data
88
MPUSBRead(myInPipe, receive_buf, ExpectedBytes, &RecvLength, 40);
89
90         if (RecvLength==0) //if no data obtained skip this
iteration of loop while keeping number of samples unchanged
91         {
92             labviewCnt--;
93             continue;
94         }

```

```

95         printf("\nRecvLength:      %d\n", RecvLength);
96         printf("\nbuf_rx_LP331AP_01:    %s\n", receive_buf);
97
98         //Format Pressure 1 and Temperature 1
99         memcpy(Temperature, receive_buf+11, 4); //memcpy( target,
source, sizeof(source)); copy temperature from buffer "0.998415 /
25.4"+'\0' (21 bytes)
100        memcpy(Pressure, receive_buf, 8);
101        Pressure_bar1 = atof(receive_buf);
102        Temperature1 = atof(Temperature);
103
104        //Format Pressure 2 and Temperature 2
105        memcpy(Temperature, receive_buf+27, 4); //memcpy( target,
source, sizeof(source)); copy temperature from buffer "0.998415 /
25.4"+'\0' (21 bytes)
106        memcpy(Pressure, receive_buf+16, 8); //memcpy( target,
source, sizeof(source)); copy temperature from buffer "0.998415 /
25.4"+'\0' (21 bytes)
107        Pressure_bar2 = atof(Pressure);
108        Temperature2 = atof(Temperature);
109
110        //Write data to file
111        fprintf(fp, "%f\t\t", Pressure_bar1);
112        fprintf(fp, "%2.1f\t\t", Temperature1);
113        fprintf(fp, "%f\t\t", Pressure_bar2);
114        fprintf(fp, "%2.1f\n", Temperature2);
115
116        dwElapsed = dwElapsed + (GetTickCount() - dwStart);
117
118        if (break_flag)
119        {
120            break;
121        }
122    }
123
124    MPUSBClose(myOutPipe);
125    MPUSBClose(myInPipe);
126    fclose(fp);
127
128    start_stop_flag=1;
129    break_flag=0;
130    start_stop_function();
131    return "\n"; //return ( (char *)buf_ReadAllData );
132 }
133
134 //comment main() to create .dll library invoked by labview
135 //compile as follows:
136 //1) gcc -c filename.c
137 //2) gcc -shared -o filename.dll filename.o
138
139 /**
140 int main()
141 {
142
143     Labview_init();
144     char *str;
145     str = Labview_main (0, 10000, 400);
146     printf (".3s", str);

```



```
147  return(0) ;  
148  }  
149  /**/
```

Pressure.h

```

150 //set to 'static' otherwise the array will be released when the
    corresponding function returns
151 unsigned static char buf_rx_ESS112[24];
152 static int labviewCnt, start_stop_flag=0, break_flag=0;
153 static float Temperature1, Temperature2, Pressure_bar1, Pressure_bar2,
    relative_height;
154
155 extern int start_stop_function(void);
156 extern int break_function(int);
157 extern float return_CalibratedOutput(void);
158 extern float return_counter(void);
159
160 extern int start_stop_function(void)
161 {
162     return start_stop_flag;
163 }
164 extern int break_function(int break_button)
165 {
166     if (break_button)
167     {
168         break_flag=1;
169     }
170     else
171     {
172         break_flag=0;
173     }
174     return break_flag;
175 }
176
177 extern float return_counter(void)
178 {
179     return labviewCnt;
180 }
181 extern float return_Pressure1(void)
182 {
183     return Pressure_bar1;
184 }
185 extern float return_Pressure2(void)
186 {
187     return Pressure_bar2;
188 }
189 extern float return_height(void)
190 {
191     return relative_height;
192 }
193 extern float return_Temperature1(void)
194 {
195     return Temperature1;
196 }
197 extern float return_Temperature2(void)
198 {
199     return Temperature2;
200 }

```

Matlab κώδικας

1. Διαφορική βαρομετρική υψομετρία

MatlabScript_DBA.m

```

1  %-----
2  %-----DIFFERENTIAL BAROMETRIC ALTIMETRY-----
3  %-----
4  %--CALCULATIONS ADDRESSED BY THE INTERNATIONAL BAROMETRIC FORMULA--
5  %-----
6  %PRINT PRESSURE DATA AT SAME HEIGHT
7  file = PressureSame;
8  P1=file(:,1);
9  P2=file(:,3);
10 fig = figure;
11 plot(P1,'r');
12 hold on;
13 plot(P2);
14 legend('P1','P2');
15 xlabel('Samples', 'FontSize', 12);
16 ylabel('Pressure (bar)', 'FontSize', 12);
17 title(sprintf('Pressure measurements at same height'), 'FontSize', 12,
18 'FontWeight', 'bold');
19 %print -dtiffn 1.Pressure-measurements-at-same-height.tif;
20 %PRINT PRESSURE DATA AT SAME HEIGHT (DEVIATION CORRECTION)
21 if (P1>P2)
22 Pdif = P1-P2;
23 PdifAv=mean(Pdif);
24 P1b=P1-PdifAv;
25 else
26 Pdif = P2-P1;
27 PdifAv=mean(Pdif);
28 P1b=P1+PdifAv;
29 end
30 figure;
31 plot(P1b,'r');
32 hold on;
33 plot(P2);
34 hold on;
35 plot(P1, '--r');
36 legend('corrected equivalent P1', 'P2', 'P1', 'FontSize', 6);
37 xlabel('Samples', 'FontSize', 12);
38 ylabel('Pressure (bar)', 'FontSize', 12);
39 title(sprintf('Pressure measurements at same height:\n Deviation
40 correction'), 'FontSize', 12, 'FontWeight', 'bold');
41 %print -dtiffn 2.Pressure-measurements-at-same-height-Deviation-
42 correction.tif;
43 %PRINT PRESSURE DATA AT HEIGHT DIFFERENCE
44 file = PressureDifferent;
45 Pa=file(:,1);
46 Pb=file(:,3);
47 figure;
48 plot(Pa,'r');
49 hold on;

```

```

49  plot(Pb);
50  legend('Pa','Pb');
51  xlabel('Samples','FontSize',12);
52  ylabel('Pressure (bar)','FontSize',12);
53  title(sprintf('Pressure measurements at height difference'),
54  'FontSize',12,'FontWeight','bold');
55  %print -dtiffn 3.Pressure-measurements-at-height-difference.tif;
56  %PRINT PRESSURE DATA AT HEIGHT DIFFERENCE (DEVIATION CORRECTION)
57  if (Pa>Pb)
58  Pa2=Pa-PdifAv;
59  else
60  Pa2=Pa+PdifAv;
61  end
62  figure;
63  plot(Pa2,'r');
64  hold on;
65  plot(Pb);
66  hold on;
67  plot(Pa,'--r');
68  legend('corrected equivalent Pa','Pb','Pa','FontSize',6);
69  xlabel('Samples','FontSize',12);
70  ylabel('Pressure (bar)','FontSize',12);
71  title(sprintf('Pressure measurements at height difference:\nDeviation
72  correction'),'FontSize',12,'FontWeight','bold');
73  %print -dtiffn 4.Pressure-measurements-at-height-difference-Deviation-
74  correction.tif
75  %PLOT HEIGHT GRAPH DETERMINED BY THE INTERNATIONAL BAROMETRIC FORMULA
76  if (mean(Pa2) > mean(Pb))
77  height=4935.125 * ( power((Pa2*100000),0.1903) -
78  power((Pb*100000),0.1903) );
79  else
80  height=4935.125 * ( power((Pb*100000),0.1903) -
81  power((Pa2*100000),0.1903) );
82  end
83  average_height = mean(height);
84  figure
85  plot(height)
86  legend('P(zB) - P(zA)','FontSize',6);
87  xlabel('Samples','FontSize',12);
88  ylabel('Height (meters)','FontSize',12);
89  title(sprintf('Height Av. = %.2fm\nInternational Barometric Formula',
90  average_height),'FontSize',12,'FontWeight','bold');
91  %print -dtiffn 5.Height-Graph-International-Barometric-Formula.tif
92  %DETERMINE RMS & P-P NOISE USING THE INTERNATIONAL BAROMETRIC FORMULA
93  figure;
94  histfit(height);
95  xlabel('Height (meters)','FontSize',12);
96  ylabel('Samples','FontSize',12); %y-axis label
97  rmsHeight=std(height);
98  Height_pp=6.6*rmsHeight;
99  title(sprintf('Noise (p-p)=%.1fm, Noise(rms)=%.2fm\nInternational
100  Barometric Formula', Height_pp, rmsHeight), 'FontSize', 11,
101  'FontWeight','bold');
102  %print -dtiffn 6.Noise-pp-and-rms-International-Barometric-Formula.tif

```

```

99  %-----
100 %-CALCULATIONS ADDRESSED BY THE HYPSONOMETRIC EQUATION-
101 %-----
102
103 %PLOT HEIGHT GRAPH DETERMINED BY THE HYPSONOMETRIC EQUATION
104 Ta2=mean(PressureDifferent(:,2))
105 Ta2=Ta2+273.15;
106 Tb=mean(PressureDifferent(:,4))
107 Tb=Tb+273.15;
108 T=(Ta2+Tb)/2
109 Tcelsius=T-273.15
110 k=1.38*power(10,-23); %Boltzmann's Constant (J/deg)
111 m=(0.22*(2*2.67*power(10,-26))) + (0.78*(2*2.3*power(10,-26))); %
Average atomic mass in kilograms
112 %The atmosphere consists of 22% of O2 (m= 2x2.67x10^-26 kg) and 78%
of N2 (m= 2x2.3x10^-26 kg)
113 g=9.81; %Acceleration of gravity (meters/sec^2)
114 H=(k*T)/(m*g);
115 if (mean(Pa2) > mean(Pb))
116 height2=H*(log(Pa2*100000)-log(Pb*100000));
117 else
118 height2=H*(log(Pb*100000)-log(Pa2*100000));
119 end
120 average_height2 = mean(height2);
121 figure
122 plot(height2)
123 legend('P(zB) - P(zA)', 'FontSize', 6);
124 xlabel('Samples', 'FontSize', 12);
125 ylabel('Height (meters)', 'FontSize', 12);
126 rmsHeight2=std(height2);
127 title(sprintf('Height Av. = %.2fm (Temp. Av. = %.1fC)\nHypsometric
Equation', average_height2, Tcelsius), 'FontSize', 12, 'FontWeight',
'bold');
128 %print -dtiffn 7.Height-Graph-Hypsometric-Equation.tif
129
130 %DETERMINE RMS & P-P NOISE USING THE HYPSONOMETRIC EQUATION
131 figure;
132 histfit(height2);
133 xlabel('Height (meters)', 'FontSize', 12);
134 ylabel('Samples', 'FontSize', 12);
135 rmsHeight2=std(height2);
136 Height_pp2=6.6*rmsHeight2;
137 title(sprintf('Noise (p-p)=%.1fm, Noise(rms)=%.2fm\nHypsometric
Equation', Height_pp, rmsHeight), 'FontSize', 11, 'FontWeight',
'bold');
138 %print -dtiffn 8.Noise-pp-and-rms-Hypsometric-Equation.tif
139
140 %PLOT TEMPERATURE GRAPH AT HEIGHT POSITION 0 METERS AND X METERS USING
DATA OF THE ROVER ATIMETER
141 T1=PressureDifferent(:,2);
142 T2=PressureDifferent(:,4);
143 fig = figure;
144 plot(T1,'r');
145 hold on; %hold the previous plot and add the following to the same
graph
146 plot(T2);
147 legend('T1 (h=0m)', 'T2 (h=Xm)');
148 xlabel('Samples', 'FontSize', 12);

```

```
149 ylabel('Temperature (oC)', 'FontSize', 12);
150 title(sprintf('Av. temperature of isothermal atmosphere=%.1foC',
Tcelsius), 'FontSize', 12, 'FontWeight', 'bold');
151 %print -dtiffn 9.Temperature-gpaph-of-altimeters.tif
152
153
154
155 cd %retrieve the current directory of the saved figures
```

2. Απλή βαρομετρική υψομετρία

MatlabScript_SBA.m

```

1  %-----
2  %-----SINGLE BAROMETRIC ALTIMETRY-----
3  %-----
4  %-CALCULATIONS ADDRESSED BY THE INTERNATIONAL BAROMETRIC FORMULA-
5  %-----
6
7  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
8  %DEFINE BASE AND ROVER ALTIMETER; POSSIBLE VALUES: 1 (FOR DEVICE 1)
   AND 3 (FOR DEVICE 2)
9  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
10 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
11 roverAltimeter = 1;
12 baseAltimeter = 3;
13 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
14 %PRINT PRESSURE DATA OF ROVER ALTIMETER AT SAME HEIGHT AND HEIGHT
   DIFFERENCE
15 P1=PressureSame(:,roverAltimeter);
16 Pa=PressureDifferent(:,roverAltimeter);
17 plot(P1);
18 hold on;
19 plot(Pa,'r');
20 legend('P1', 'Pa', 'FontSize', 6);
21 xlabel('Samples', 'FontSize', 12);
22 ylabel('Pressure (bar)', 'FontSize', 12);
23 title(sprintf('Pressure measurements of the rover altimeter'),
   'FontSize', 12, 'FontWeight', 'bold');
24 %print -dtiffn 1.Pressure-data-at-same-height-and-height-
   difference.tif
25
26 %PLOT HEIGHT GRAPH DETERMINED BY THE INTERNATIONAL BAROMETRIC FORMULA
27 height=4935.125 * ( power((P1*100000),0.1903) -
   power((Pa*100000),0.1903) );
28 average_height = mean(height);
29 figure;
30 plot(height);
31 legend(sprintf('P(zB) - P(zA)'), 'FontSize', 6);
32 xlabel('Samples', 'FontSize', 12);
33 ylabel('Height (meters)', 'FontSize', 12);
34 title(sprintf('Height Av. = %.2fm\nInternational Barometric Formula',
   average_height), 'FontSize', 12, 'FontWeight', 'bold');
35 %print -dtiffn 2.Height-Graph-International-Barometric-Formula.tif
36
37 %DETERMINE RMS & P-P NOISE USING THE INTERNATIONAL BAROMETRIC FORMULA
38 figure;
39 histfit(height);
40 rmsHeight=std(height);
41 Height_pp=6.6*rmsHeight;
42 %legend(sprintf('RMS Noise=%.2fm\nP-P Noise=%.1fm', rmsHeight,
   Height_pp));
43 xlabel('Height (meters)', 'FontSize', 12);
44 ylabel('Samples', 'FontSize', 12);

```

```

45 title(sprintf('Noise (p-p)=%.1fm,          Noise (rms)=%.2fm\nInternational
Barometric Formula', Height_pp, rmsHeight), 'FontSize', 12,
'FontWeight', 'bold');
46 %print -dtiffn 3.Noise-pp-and-rms-International-Barometric-Formula.tif
47
48 %-----
49 %-CALCULATIONS ADDRESSED BY THE HYPSONETRIC EQUATION-
50 %-----
51
52 %PLOT HEIGHT GRAPH DETERMINED BY THE HYPSONETRIC EQUATION
53 T1=mean(PressureSame(:,roverAltimeter+1));
54 T1=T1+273.15;
55 Ta=mean(PressureDifferent(:,roverAltimeter+1));
56 Ta=Ta+273.15;
57 T=(T1+Ta)/2;
58 Tcelsius=T-273.15;
59 k=1.38*power(10,-23);
60 m=(0.22*(2*2.67*power(10,-26)))+(0.78*(2*2.3*power(10,-26)));
61 g=9.81;
62 H=(k*T)/(m*g);
63 height2=H*(log(P1*100000)-log(Pa*100000));
64 average_height2 = mean(height2);
65 figure;
66 plot(height2);
67 legend('P(zB) - P(zA)', 'FontSize', 6);
68 xlabel('Samples', 'FontSize', 12);
69 ylabel('Height (meters)', 'FontSize', 12);
70 rmsHeight2=std(height2);
71 title(sprintf('Height Av. = %.2fm (Temp. Av. = %.1foC)\nHypsometric
Equation', average_height2, Tcelsius), 'FontSize', 12, 'FontWeight',
'bold');
72 %print -dtiffn 4.Height-Graph-Hypsometric-Equation.tif
73
74 %DETERMINE RMS & P-P NOISE USING THE HYPSONETRIC EQUATION
75 figure;
76 histfit(height2);
77 xlabel('Height (meters)', 'FontSize', 12);
78 ylabel('Samples', 'FontSize', 12);
79 rmsHeight2=std(height2);
80 Height_pp2=6.6*rmsHeight2;
81 title(sprintf('Noise (p-p)=%.1fm,          Noise (rms)=%.2fm\nHypsometric
Equation', Height_pp, rmsHeight), 'FontSize', 12, 'FontWeight',
'bold');
82 %print -dtiffn 5.Noise-pp-and-rms-Hypsometric-Equation.tif
83
84 %PLOT TEMPERATURE GRAPH AT HEIGHT POSITION 0 METERS AND X METERS USING
DATA OF THE ROVER ATIMETER
85 T1=PressureSame(:,roverAltimeter+1);
86 Ta=PressureDifferent(:,roverAltimeter+1);
87 fig = figure;
88 plot(T1,'r');
89 hold on; %hold the previous plot and add the following to the same
graph
90 plot(Ta);
91 legend('T1 (h=0m)', 'Ta (h=Xm)');
92 xlabel('Samples', 'FontSize', 12);
93 ylabel('Temperature (oC)', 'FontSize', 12);

```



```
94 title(sprintf('Av. temperature of isothermal atmosphere = %.1foC',
Tcelsius), 'FontSize', 12, 'FontWeight', 'bold');
95 %print -dtiffn 6.Temperature-gpaph-of-rover-altimeter.tif
96
97 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
98 %PLOT PRESSURE VARIATIONS DETERMINED BY THE BASE ALTIMETER
99 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
100 figure;
101 P2=PressureSame(:,baseAltimeter);
102 Pb=PressureDifferent(:,baseAltimeter);
103 P=cat(1,P2,Pb);
104 plot(P);
105 xlabel('Samples', 'FontSize', 12);
106 ylabel('Pressure (bar)', 'FontSize', 12);
107 title(sprintf('Pressure variations acquired by the base altimeter'),
'FontSize', 12, 'FontWeight', 'bold');
108 %print -dtiffn 7.Pressure-variations-determined-by-the-base-
altimeter.tif
109
110
111
112 cd %retrieve the current directory of the saved figures
```

3. Μετρήσεις σε αεροστεγές δοχείο

MatlabScript_DEVIATION.m

```

1  %CLOSE PREVIOUS FIGURES AND CLEAR WORKSPACE/COMMAND WINDOW/VARIABLE
    EDITOR
2  close all;
3  clear all;
4
5  %DEFINE WORKING DIRECTORY
6  path='C:\zSAVES\#PhD\#PRESENTATION\Measurements-AirtightEnclosure';
7  cd(path);
8
9  %DEFINE THE BASE NAME OF THE PRESSURE FILES
10 BaseName='Pressure';
11
12 %LOAD ALL .TXT FILES OF THE CURRENT PATH AND SAVE THEM AS ARRAYS OF
    THE SAME FILE NAME
13 files = dir('*.txt');
14 for i=1:length(files)
15     %eval(['load ' files(i).name ' -ascii']);
16     currentFile = sprintf('%s%d.txt',BaseName,i);
17     load(currentFile);
18 end
19
20 %USE 'HSV' COLOR MAP TO CREATE DIFFERENT COLORS (AS MANY AS THE .TXT
    FILES) FOR EACH PAIR OF PRESSURE GRAPHS
21 colormap=hsv(length(files));
22 %LOOP1 (USE DIFFERENT FIGURE FOR EACH PAIR OF MEASUREMENTS)
23 for i = 1:length(files)
24     fig = figure;
25     FileName=[BaseName,num2str(i),'.txt']
26     P=dlmread(FileName);
27     P1=P(:,1);
28     P2=P(:,3);
29     Pdif = abs(P1-P2);
30     PdifAv=mean(Pdif);
31     if (mean(P1) > mean(P2))
32         height=4935.125 * ( power((P1*100000),0.1903) -
            power((P2*100000),0.1903) );
33     else
34         height=4935.125 * ( power((P2*100000),0.1903) -
            power((P1*100000),0.1903) );
35     end
36     average_height = mean(height);
37     plot(P1,'color',colormap(i,:));
38     hold on;
39     plot(P2,'color',colormap(i,:));
40     hold on;
41     legend(sprintf('Av. Deviation(%s) = %.1fubar\nHeight Error = %.2fm',
        FileName, PdifAv*100000, average_height));
42     xlabel('Samples', 'FontSize', 12);
43     ylabel('Pressure (bar)', 'FontSize', 12);
44     currentFile = sprintf('%s%d.tif',BaseName,i);
45     print ('-dtiffn', currentFile);
46 end
47
48 %LOOP2 (SHOW ALL GRAPHS IN A SIGNLE FIGURE)
49 fig = figure;

```

```
50     for i = 1:length(files)
51         FileName=[BaseName,num2str(i),'.txt']
52         P=dlmread(FileName);
53         P1=P(:,1);
54         P2=P(:,3);
55         Pdif = abs(P1-P2);
56         PdifAv=mean(Pdif);
57         if (mean(P1) > mean(P2))
58             height=4935.125 * ( power((P1*100000),0.1903) -
59             power((P2*100000),0.1903) );
60         else
61             height=4935.125 * ( power((P2*100000),0.1903) -
62             power((P1*100000),0.1903) );
63         end
64         average_height = mean(height);
65         plot(P1,'color',colormap(i,:));
66         hold on;
67         plot(P2,'color',colormap(i,:));
68         hold on;
69         xlabel('Samples', 'FontSize', 12);
70         ylabel('Pressure (bar)', 'FontSize', 12);
71         end
72         currentFile = sprintf('%s-all.tif',BaseName);
73         print ('-dtiffn', currentFile);
```

Αναφορές

- [1] D. E. Bolanakis, K. T. Kotsis, and T. Laopoulos, "Ethernet & PC-based Experiments on Barometric Altimetry using MEMS in a Wireless Sensor Network", *Computer Application in Engineering Education*, 24(3), pp. 428-442, 2016.
- [2] D. E. Bolanakis, K. T. Kotsis, and T. Laopoulos, "A Prototype Wireless Sensor Network System for a Comparative Evaluation of Differential and Absolute Barometric Altimetry", *IEEE Aerospace and Electronic Systems Magazine*, 30(11), pp. 20-28, 2015.
- [3] D. E. Bolanakis, K. T. Kotsis, and T. Laopoulos, "Temperature Influence on Differential Barometric Altitude Measurements", *The 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS'2015)*, Warsaw, Poland, 2015, pp. 120-124.
- [4] D. E. Bolanakis, K. T. Kotsis, T. Laopoulos, "Work in Progress: Experiments on MEMS Barometric Pressure Sensors", *3ο Πανελλήνιο Συνέδριο Ηλεκτρονικής και Τηλεπικοινωνιών (PACET 2015)*, Ιωάννινα, 2015, σελ. 1-3.
- [5] Δ. Ε. Μπολανάκης και Κ. Κώτσης, "Χρήση Πειραμάτων Απομακρυσμένης Πρόσβασης με Ασύρματη Επικοινωνία στη Διδασκαλία Φυσικής: ανασκόπηση βιβλιογραφίας και προτάσεις", *6ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση*, Φλώρινα, 2009, σελ. 579-586.
- [6] Δ. Ε. Μπολανάκης και Κ. Κώτσης, "Απομακρυσμένος Έλεγχος Μετρήσεων Φυσικών Μεγεθών Πραγματικού Χρόνου: δυνατότητες και περιορισμοί", *9ο Πανελλήνιο Συνέδριο Διδακτικής των Φυσικών Επιστημών και Νέων Τεχνολογιών στην Εκπαίδευση*, Θεσσαλονίκη, 2015, σελ. 232-236.
- [7] Δ. Ε. Μπολανάκης και Κ. Κώτσης, "Πείραμα για τη μέτρηση υψομετρικής διαφοράς: μία πρόταση για άτομα με κινητική αναπηρία", *Πανελλήνιο Συνέδριο στις Διδακτικές προσεγγίσεις και πειραματική διδασκαλία στις Φυσικές Επιστήμες*, Θεσσαλονίκη, 2016, σελ. 205-213.