Σελίδα τίτλου

ΠΑΝΕΠΙΣΤΗΜΙΟ
ΙΩΑΝΝΙΝΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΟΔΗΓΟΣ ΑΝΑΠΤΥΞΗΣ ΕΝΟΣ ΑΣΥΡΜΑΤΟΥ ΔΙΚΤΥΟΥ ΑΙΣΘΗΤΗΡΩΝ ΚΑΙ ΤΗΣ ΕΦΑΡΜΟΓΗΣ ΣΥΛΛΟΓΗΣ ΤΩΝ ΔΕΔΟΜΕΝΩΝ ΤΟΥ

Σέργιος Βαλλίδης

Επιβλέπων: Γεώργιος Τσουμάνης

Άρτα, Σεπτέμβριος, 2021

A GUIDE FOR DEVELOPING A WIRELESS SENSOR NETWORK AND ITS DATA COLLECTION APPLICATION

Εγκρίθηκε από τριμελή εξεταστική επιτροπή

Αρτα, 23/9/2021

ΕΠΙΤΡΟΠΗ ΑΞΙΟΛΟΓΗΣΗΣ

1. Επιβλέπων καθηγητής

   Γεώργιος Τσουμάνης,

2. Μέλος επιτροπής

   Κωσταντίνος Αγγέλης,

3. Μέλος επιτροπής

   Νικόλαος Γιαννακέας

Δήλωση μη λογοκλοπής

Δηλώνω υπεύθυνα και γνωρίζοντας τις κυρώσεις του Ν. 2121/1993 περί Πνευματικής Ιδιοκτησίας, ότι η παρούσα πτυχιακή εργασία είναι εξ ολοκλήρου αποτέλεσμα δικής μου ερευνητικής εργασίας, δεν αποτελεί προϊόν αντιγραφής ούτε προέρχεται από ανάθεση σε τρίτους. Όλες οι πηγές που χρησιμοποιήθηκαν (κάθε είδους, μορφής και προέλευσης) για τη συγγραφή της περιλαμβάνονται στη βιβλιογραφία.

Βαλλίδης, Σέργιος

Υπογραφή

ΕΥΧΑΡΙΣΤΙΕΣ

Ευχαριστώ θερμά τον επιβλέποντα καθηγητή μου, κ. Τσουμάνη Γεώργιο για την εποικοδομητική συνεργασία και καθοδήγηση κατά τη διάρκεια εκπόνησης της εργασίας.

ΠΕΡΙΛΗΨΗ

Αντικείμενο της παρούσας πτυχιακής εργασίας είναι η γνωριμία μας με τα ασύρματα δίκτυα αισθητήρων Wireless Sensors Networks (WSNs) και κυρίως εκείνα που κύρια λειτουργία τους είναι η μέτρηση απόστασης και κλίσης, σε πραγματικό χρόνο. Κύριος στόχος είναι η κατάρτιση ενός οδηγού που θα μας επιτρέψει να δημιουργήσουμε ένα ασύρματο δίκτυο αισθητήρων μαζί με την εφαρμογή συλλογής των δεδομένων του.

Πιο συγκεκριμένα, στο πρώτο κεφάλαιο ασχολούμαστε με την αρχιτεκτονική των ασύρματων δικτύων αισθητήρων. Αναλύουμε το επίπεδο εφαρμογής, το επίπεδο μεταφοράς, το επίπεδο δικτύου , το επίπεδο ζεύξης δεδομένων, το φυσικό επίπεδο, καθώς και τα διάφορα πρότυπα και πρωτόκολλα επικοινωνίας και οι τοπολογίες που εφαρμόζονται.

Στο δεύτερο κεφάλαιο αναλύουμε τα εξαρτήματα και τους αισθητήρες που χρησιμοποιήσαμε για την δημιουργία ενός ασύρματου δικτύου αισθητήρων το οποίο αποτελείται από δυο κόμβους.

Στο τρίτο κεφάλαιο , εστιάζουμε στην εφαρμογή που χρησιμοποιήσαμε και αναλύουμε τα αναγκαία βήματα προκειμένου να δημιουργήσουμε την δικιά μας εφαρμογή συλλογής δεδομένων.

Στο τελευταίο κεφάλαιο γίνεται μια σύντομη περιγραφή της συνδεσμολογίας των κόμβων με τους αισθητήρες και αναλύουμε τον κώδικα προκειμένου να λειτουργήσει το δίκτυο αισθητήρων μας.

Τέλος, παρουσιάζουμε την εφαρμογή μας , ώστε να κατανοήσουμε την χρησιμότητα των ασύρματων αισθητήρων.

Λέξεις-κλειδιά: Ασύρματα δίκτυα αισθητήρων, Ασύρματη επικοινωνία, αισθητήρες, τοποθέτηση κόμβων

ABSTRACT

The purpose of writing this thesis is our acquaintance with Wireless Sensor Networks (WSNs) and especially those whose main function is the measurement of distance and inclination, in real time.

The main goal is the development of a guide that will allow us to create a wireless network of sensors and its data collection application.

In detail, in the first chapter our attention is focused on the architecture of wireless sensor networks. We analyze the application level, the transfer level, the network level, the data link level, the physical level, as well as the various communication standards and protocols and the topologies that are applied.

In the second chapter we analyze the components and sensors that we used to create a wireless network of sensors which consists of two nodes.

In the third chapter, we focus on the application we used and analyze the necessary steps to create our own data collection application.

In the last chapter there is a brief description of the connections of the nodes with the sensors and we analyze the code for our sensor network to work.

Finally, we present our application, to understand the usefulness of wireless sensors.

Keywords: Wireless sensor networks, Wireless communication, sensors, node placement

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

1 ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝΠίνακας 1 Χαρακτηριστικά Arduino UNO

ΚΑΤΑΛΟΓΟΣ ΔΙΑΓΡΑΜΜΑΤΩΝ/ΕΙΚΟΝΩΝ

ΠΙΝΑΚΑΣ ΣΥΝΤΟΜΟΓΡΑΦΙΩΝ

TEI-H..…….……..Τεχνολογικό Εκπαιδευτικό Ίδρυμα Ηπείρου

ΑΔΑ..……..…..Ασύρματο Δίκτυο Αισθητήρων

WSN .……...Wireless Sensor Network

CBM …….……...Condition-based maintenance

TCP….……...……. Transmission Control Protocol

UDP….……. User Datagram Protocol

SPI……..……Serial Peripheral Interface

CE……...……..Chip Enable

CSN……Chip Select Not

MEMES…….…….Micro Electro Mechanical System

VCC ……....……Voltage Common Collector

GND……...……. Ground

RST……. Reset

I/O…….….. Input/Output

GPIO ……..……..General-Purpose Input/Output

CH_PD…….…….… Chip Power-Down

TX ……...…….…… Transmitting pin

RX …….……Receiving pin

ΕΙΣΑΓΩΓΗ

Οι τεχνολογικές εξελίξεις στην επιστήμη των υπολογιστών και των τηλεπικοινωνιών, έχουν οδηγήσει στην εμφάνιση κατανεμημένων Ασύρματων Δικτύων Αισθητήρων (ΑΔΑ / Wireless Sensor Network - WSN) που είναι σε θέση να παρατηρούν τον φυσικό κόσμο, να επεξεργάζονται τα δεδομένα, να λαμβάνουν αποφάσεις με βάση τις παρατηρήσεις και να διεξάγουν κατάλληλες ενέργειες. Ένα ΑΔΑ αποτελείται συνήθως από αυτονόμους διασκορπισμένους αισθητήρες που χρησιμοποιούνται για τον εντοπισμό και την καταγραφή φυσικών ή περιβαλλοντικών δεδομένων, όπως η θερμοκρασία, ο ήχος, η πίεση, τα οποία (δεδομένα) συλλέγονται, μέσω του δικτύου, σε μια συγκεκριμένη τοποθεσία. Τα πιο μοντέρνα συστήματα αισθητήρων είναι αμφίδρομα, κάτι που τους επιτρέπει τόσο να δέχονται πληροφορίες όσο και να επιτρέπουν τον έλεγχο του κάθε αισθητήρα.

Ένα ΑΔΑ αποτελείται από κόμβους, οι οποίοι δύναται να είναι από ένας έως και χιλιάδες, ενώ κάθε κόμβος φέρει έναν ή και πολλούς αισθητήρες οι οποίοι συσσωρεύουν δεδομένα. Κάθε κόμβος του δικτύου αισθητήρων αποτελείται από μερικά χαρακτηριστικά κομμάτια: (1) ένα ραδιοπομποδέκτη με μια εσωτερική κεραία ή μια σύνδεση σε μια εξωτερική κεραία, (2) ένα μικροελεγκτή, (3) ένα ηλεκτρονικό κύκλωμα για τη διασύνδεση με τους αισθητήρες, και (4) μια πηγή ενέργειας, συνήθως μια μπαταρία ή μια ενσωματωμένη μορφή συλλογής ενέργειας. Το κόστος των αισθητήριων κόμβων ποικίλει αναλόγως με την πολυπλοκότητα και το μέγεθος. Οι περιορισμοί σε μέγεθος και κόστος έχουν ως αποτέλεσμα αντίστοιχους περιορισμούς σε πόρους όπως ενέργεια, μνήμη, υπολογιστική ταχύτητα και στο εύρος ζώνης των επικοινωνιών. (Feng Xia, 2007)

Το κίνητρο για την ανάπτυξη των ΑΔΑ ήταν οι στρατιωτικές εφαρμογές, όπως η παρακολούθηση των πεδίων μάχης (Tatiana Bokareva, 2006). Ωστόσο, στη σύγχρονη εποχή τέτοια δίκτυα βρίσκονται σε εφαρμογή σε διάφορες περιβαλλοντικές, καταναλωτικές και βιομηχανικές κατασκευές. Τέτοιες εφαρμογές είναι, για παράδειγμα, η παρακολούθηση της ποιότητας και ρύπανσης του αέρα, η ανίχνευση δασικών πυρκαγιών και κατολισθήσεων, ο έλεγχος ποιότητας υδάτων και η πρόληψη φυσικών καταστροφών.

Όσον αφορά στη βιομηχανική παρακολούθηση, ΑΔΑ έχουν αναπτυχθεί για τη βασική συντήρηση των μηχανημάτων (Condition-based maintenance - CBM), δεδομένου ότι προσφέρουν σημαντική εξοικονόμηση κόστους και επιτρέπουν νέες λειτουργίες. Μια άλλη εφαρμογή των ασυρμάτων δικτυών δεδομένων εξυπηρετεί τον παθητικό εντοπισμό και την παρακολούθηση, η οποία προτείνεται λόγω του χαμηλού κόστους της εν λόγω τεχνολογίας,

αξιοποιώντας τις ιδιότητες των ασύρματων ζεύξεων που είναι εγκατεστημένα σε ένα δίκτυο υποδομής ΑΔΑ (F. Viani, 2010).

Τέλος, ΑΔΑ εφαρμόζονται ολοένα και περισσότερο στο «έξυπνο» σπίτι, τα οποία μέσω της επιλεκτικής διαχείρισης αντικειμένων καθημερινής χρήσης τα εξελίσσουν σε «έξυπνες συσκευές», επιτυγχάνοντας σημαντική βελτίωση στην ποιότητα των υπηρεσιών που παρέχονται στους χρήστες τους (Debnath, et al., 2012), (Surie, 2008).

1. Το μοντέλο του συστήματος

Τα δίκτυα αισθητήρων ασχολούνται με το χώρο και το χρόνο, δηλαδή με την τοποθεσία, την κάλυψη και το συγχρονισμό των δεδομένων. Τα εισερχόμενα δεδομένα είναι ο κοινός παρονομαστής ενός δικτύου αισθητήρων.

Όταν γίνεται η συγκέντρωση των δεδομένων, υπάρχει ένα μεγάλο εύρος δεδομένων που εξαρτώνται από το χρόνο. Επομένως, τα δίκτυα αισθητήρων υποστηρίζουν συχνά υπολογισμούς εντός δικτύου. Ορισμένα δίκτυα αισθητήρων χρησιμοποιούν επεξεργασία κόμβου-πηγής και άλλα χρησιμοποιούν μια ιεραρχική αρχιτεκτονική επεξεργασίας. Πολλές φορές, αντί να στέλνονται τα ανεπεξέργαστα δεδομένα στους κόμβους που είναι υπεύθυνοι για τη συγχώνευση δεδομένων, χρησιμοποιούν, οι ίδιοι οι κόμβοι, τις ικανότητες επεξεργασίας τους τοπικά, για να εκτελούν βασικούς υπολογισμούς, και ύστερα μεταδίδουν μόνο ένα υποσύνολο των δεδομένων ή / και μερικώς επεξεργασμένα δεδομένα. Στο δικό μας μοντέλο, οι κομβόι χρησιμοποιούν την υπολογιστική τους δυνατότητα για να κάνουν απλούς υπολογισμούς και στέλνουν αυτά τα δεδομένα σε έναν κόμβο sink, , ο οποίος είναι υπεύθυνος για τη συλλογή όλων των δεδομένων που δημιουργούνται από τους (υπόλοιπους) κόμβους του δικτύου.

Σε μια ιεραρχική αρχιτεκτονική επεξεργασίας, η επεξεργασία συντελείται σε διαδοχικές βαθμίδες μέχρι οι πληροφορίες να φτάσουν στον κατάλληλο κόμβο λήψης αποφάσεων ή / και στο διοικητικό σημείο. Οι κόμβοι περιορίζονται σχεδόν πάντα στην παροχή ενέργειας και στο ραδιοφωνικό εύρος ζώνης μετάδοσης καναλιού. Αυτοί οι περιορισμοί, σε συνδυασμό με μια τυπική παράταξη μεγάλου αριθμού κόμβων, έχει δημιουργήσει πληθώρα προκλήσεων στην αρχιτεκτονική και τη διαχείριση ενός ΑΔΑ. (K. Sohraby, 2007)

1.1 Αρχιτεκτονική

Σε αυτήν την ενότητα επισημαίνουμε τα βασικά στοιχεία και τον σχεδιασμό των δικτύων αισθητήρων. Τα στοιχεία αυτά και οι αρχές σχεδιασμού χαρακτηρίζονται από τους ακόλουθους παράγοντες:

- μεγάλος πληθυσμός αισθητήρων
- μεγάλες ροές δεδομένων
- ελλιπή / αβέβαια δεδομένα

- υψηλή πιθανότητα αποτυχίας κόμβου
- υψηλή πιθανότητα αποτυχίας σύνδεσης (π.χ., λόγω παρεμβολών)
- περιορισμοί ηλεκτρικής ισχύος
- περιορισμοί ισχύος επεξεργασίας
- τοπολογία πολλαπλών κόμβων
- έλλειψη ολικής γνώσης για το δίκτυο και (συχνά) περιορισμένη διοικητική υποστήριξή του.

Για την κατάλληλη διαχείριση των λιγοστών πόρων ενός ΑΔΑ, τα πρωτόκολλα δρομολόγησης για ασύρματα δίκτυα δεδομένων πρέπει να σχεδιάζονται λαμβάνοντας υπόψη τους ενεργειακούς περιορισμούς. Η δρομολόγηση που βασίζεται στα δεδομένα και η επεξεργασία των δεδομένων εντός του δικτύου είναι σημαντικές έννοιες, που συνδέονται εγγενώς με τα δίκτυα αισθητήρων. Απαιτούνται λοιπόν τεχνολογίες με επίκεντρο τα δεδομένα, που εκτελούν τη συγκέντρωση δεδομένων εντός δικτύου για να αποφέρουν ενεργειακά αποδοτική διάδοση.

Ένα δίκτυο αισθητήρων αποτελείται από πολλούς κόμβους. Ένας κόμβος έχει συνήθως ενσωματωμένες δυνατότητες επεξεργασίας και αποθηκευτικό χώρο. Ο κόμβος μπορεί να έχει έναν ή περισσότερους αισθητήρες που λειτουργούν στους ακουστικούς, σεισμικούς, ραδιο (ραντάρ), υπέρυθρους, οπτικούς, μαγνητικούς, χημικούς ή βιολογικούς τομείς. Ο κόμβος έχει συνδέσεις επικοινωνίας, συνήθως ασύρματες, με γειτονικούς κόμβους. Οι κόμβοι είναι διασκορπισμένοι σε έναν ειδικό τομέα που ονομάζεται πεδίο. Κάθε ένας από τους κατανεμημένους κόμβους έχει συνήθως τη δυνατότητα να συλλέγει δεδομένα, να τα αναλύει και να τα δρομολογεί σε ένα καθορισμένο σημείο συλλογής (sink). (K. Sohraby, 2007, p. 15)

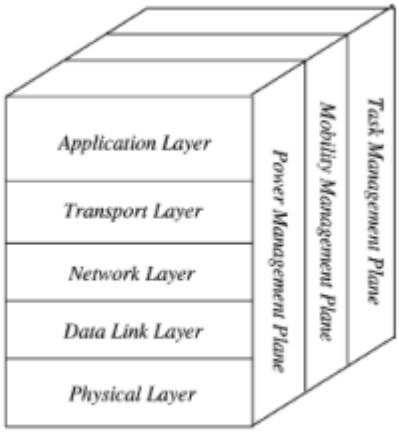
Η πιο κοινή αρχιτεκτονική για το ΑΔΑ μοιάζει με το μοντέλο OSI το οποιο είναι μια ιεραρχική δομή επτά επιπέδων που καθορίζει τις προδιαγραφές επικοινωνίας μεταξύ δύο υπολογιστών, ορίζοντας επακριβώς τον σκοπό κάθε επιπέδου αλλά και τα χρησιμοποιούμενα πρωτόκολλα. Το δίκτυο αισθητήρων αποτελείται από πέντε επίπεδα: (Akyildiz, 2002)

- επίπεδο εφαρμογής
- επίπεδο μεταφοράς
- επίπεδο δικτύου
- επίπεδο ζεύξης δεδομένων
- φυσικό επίπεδο.

Εκτός από αυτά τα πέντε στρώματα υπάρχουν και αλλά τρία, εγκάρσια επίπεδα ή στρώματα τα οποία είναι:

- επίπεδο διαχείρισης ισχύος
- επίπεδο διαχείρισης κινητικότητας
- επίπεδο διαχειριστή εργασιών

Αυτά τα επίπεδα, που φαίνονται στην Εικόνα 1 (Akyildiz, 2002, pp. 393-422), χρησιμοποιούνται για τη διαχείριση του δικτύου και για να κάνουν τους αισθητήρες να λειτουργούν μαζί προκειμένου να αυξηθεί η συνολική αποτελεσματικότητα του δικτύου.



Εικόνα 1 Αρχιτεκτονική ΑΔΑ((Akyildiz, 2002))

1.1.1 Επίπεδο εφαρμογής

Το επίπεδο εφαρμογής είναι υπεύθυνο για τη διαχείριση της κυκλοφορίας και παροχής λογισμικού για διαφορετικές εφαρμογές που ερμηνεύουν τα δεδομένα σε κατανοητή μορφή ή στέλνουν ερωτήματα για να λάβουν ορισμένα πληροφορίες. (Yick, 2008, pp. 2292-2330)

Στο μοντέλο OSI, ο ορισμός του επιπέδου εφαρμογής είναι πιο συνοπτικός. Όπως στο μοντέλο OSI έτσι και εδώ το επίπεδο εφαρμογής ορίζεται ως η διεπαφή του χρήστη, που

είναι υπεύθυνη για την εμφάνιση των ληφθέντων πληροφοριών στον χρήστη. Το OSI προσθέτει επιπλέον λειτουργίες κάτω από το επίπεδο εφαρμογής, αλλά και πάνω από το επίπεδο μεταφοράς σε δύο επιπλέον επίπεδα, το επίπεδο συνεδρίας και το επίπεδο παρουσίασης. Το OSI καθορίζει έναν αυστηρό διαχωρισμό της λειτουργικότητας σε αυτά τα επίπεδα και παρέχει υλοποιήσεις πρωτοκόλλων για κάθε επίπεδο.

1.1.2 Επίπεδο μεταφοράς

Η λειτουργία του επιπέδου μεταφοράς είναι να παρέχει αξιοπιστία και αποφυγή συμφόρησης των δεδομένων και πολλά πρωτόκολλα έχουν σχεδιαστεί για να παρέχουν αυτήν τη λειτουργία είτε στο upstream (user to sink), είτε στο downstream (sink to user). Αυτά τα πρωτόκολλα χρησιμοποιούν διαφορετικούς μηχανισμούς για την ανίχνευση της απώλειας δεδομένων και την αποκατάστασή της (Pereira, 2007). Αυτό το επίπεδο απαιτείται όταν ένα σύστημα χρειάζεται να έχει πρόσβαση σε άλλα δίκτυα. Η παροχή ενός αξιόπιστου πρωτόκολλου hop σε hop (το οποίο μεταφέρει κομμάτια δεδομένων από κόμβο σε κόμβο με τρόπο αποθήκευσης και προώθησης) είναι ενεργειακά πιο αποδοτικό από το πρωτόκολλο end σε end (υπεύθυνο για τη μεταφορά των δεδομένων από μια πηγή σε ένα ή περισσότερα τελικά σημεία του δικτύου), και αυτό καθιστά το TCP πρωτόκολλο ακατάλληλο για ΑΔΑ. Πιο συχνά χρησιμοποιείται το πρωτόκολλο από sink σε κόμβο ως downstream το οποίο είναι μια σύνδεση για την μετάδοση και την καλύτερη διάδοσης UDP πληροφοριών λόγω της περιορισμένης διαθέσιμης μνήμης. Από την άλλη πλευρά, το User to sink θεωρείται upstream σύνδεση και χρησιμοποιείται για μονοφωνική μετάδοση και διάδοση TCP ή UDP πρωτοκόλλων. (Yick, 2008)

Γενικά, τα πρωτόκολλα μεταφοράς μπορούν να χωριστούν σε:

α) Καθοδήγηση από το πακέτο: όλα τα πακέτα που αποστέλλονται από την πηγή πρέπει να φτάσουν στον προορισμό

β) Καθοδήγηση από το συμβάν: το συμβάν πρέπει να εντοπιστεί, αλλά αρκεί ένα μήνυμα ειδοποίησης να φτάσει στον κόμβο sink, δηλαδή τον κόμβο που θα μαζέψει όλα τα δεδομένα από τους υπολοίπους κόμβους και θα τα τοποθετήσει όπου έχει ορίσει ο χρήστης.

1.1.3 Επίπεδο δικτύου

Η κύρια λειτουργία του επιπέδου δικτύου είναι η δρομολόγηση. Αυτό το επίπεδο έχει πολλές προκλήσεις ανάλογα με την εφαρμογή, αλλά προφανώς, οι μεγαλύτερες προκλήσεις είναι η

εξοικονόμηση ενέργειας, η περιορισμένη μνήμη και ο buffer. Ο buffer είναι μια περιοχή μνήμης που χρησιμοποιείται για την προσωρινή διατήρηση δεδομένων ενώ μετακινούνται από το ένα μέρος στο άλλο. Τα buffer χρησιμοποιούνται γενικά όταν υπάρχει διαφορά μεταξύ του ρυθμού με τον οποίο λαμβάνονται τα δεδομένα και του ρυθμού με τον οποίο μπορούν να υποβληθούν σε επεξεργασία. Εάν αφαιρέσουμε τα buffer, τότε είτε θα έχουμε απώλεια δεδομένων, είτε θα έχουμε χαμηλότερη χρήση εύρους ζώνης (harleek, 24 May, 2020). Επειδή οι αισθητήρες δεν έχουν παγκόσμιο αναγνωριστικό, πρέπει να έχουν προγραμματιστεί και ρυθμιστεί πριν την εφαρμογή τους σε κάποιον κόμβο. Αυτό βρίσκεται σε αντίθεση με τα δίκτυα υπολογιστών που διαθέτουν διεύθυνση IP και μια κεντρική συσκευή για τον έλεγχο. Η βασική ιδέα του πρωτοκόλλου δρομολόγησης είναι ο καθορισμός μιας αξιόπιστης διαδρομής και άλλων περιττών διαδρομών ανάλογα με μια συγκεκριμένη κλίμακα που ονομάζεται metric, το οποίο διαφέρει από πρωτόκολλο σε πρωτόκολλο.

Υπάρχουν πολλά πρωτόκολλα δρομολόγησης διαθέσιμα για αυτό το επίπεδο, και μπορούν να χωριστούν σε επίπεδη δρομολόγηση (για παράδειγμα, άμεση διάχυση) και ιεραρχική δρομολόγηση (για παράδειγμα, LEACH (Heinzelman, January 2000)) ή μπορεί να χωριστεί σε χρονική δρομολόγηση, η οποία βασίζεται σε ερωτήματα και καθοδηγείται από συμβάντα. Στο συνεχές χρονικά πρωτόκολλο, τα δεδομένα αποστέλλονται περιοδικά και με βάση το χρόνο για εφαρμογές που χρειάζονται περιοδική παρακολούθηση. Στα πρωτόκολλα που βασίζονται σε συμβάντα και στα πρωτόκολλα που βασίζονται σε ερωτήματα, ο αισθητήρας αποκρίνεται ανάλογα με τη δράση ή το ερώτημα του χρήστη. (Yick, 2008)

Όσων αφορά την συγκέντρωση και την συγχώνευση δεδομένων στους κόμβους, για να παρέχεται πλήρης κάλυψη για μια συγκεκριμένη περιοχή, ακόμη και όταν έχουμε μια αποτυχία, πρέπει να αναπτύξουμε περιττούς αισθητήρες, δηλαδή εξτρά αισθητήρες που χρησιμοποιούνται σε περίπτωση κάποιας βλάβης. Οι περιττοί αυτοί αισθητήρες παρέχουν επαναλαμβανόμενα δεδομένα και εκτός από αισθητήρες που στέλνουν δεδομένα σε στυλ πολλαπλών μηνυμάτων (από αισθητήρα σε αισθητήρα μέχρι να φτάσει στον κόμβο sink) μερικές φορές -όπως στα πρωτόκολλα πλημμύρας- κάθε αισθητήρας προωθεί δεδομένα σε όλους τους γείτονες και οι γείτονες διαβιβάζουν δεδομένα στους γείτονές τους και ούτω καθεξής.

Με τον παραπάνω τρόπο ένας κόμβος, μπορεί να λάβει τεράστια ποσότητα επαναλαμβανόμενων δεδομένων από διαφορετικούς γείτονες και αυτά τα δεδομένα θα μπορούσαν να δημιουργηθούν από τον ίδιο κόμβο προέλευσης ή από συμπληρωματικούς

κόμβους του δικτύου. Η επεξεργασία των δεδομένων αυτών καταναλώνει λιγότερη ισχύ από τη μετάδοση τους, έτσι συγκεντρώνοντας και συγχωνεύοντας τα δεδομένα σε έναν κόμβο εξαλείφουμε την επανάληψη των δεδομένων. (Yick, 2008)

1.1.4 Επίπεδο ζεύξης δεδομένων

Αυτό το επίπεδο είναι υπεύθυνο για τη ροή των δεδομένων πολυπλεξίας, την ανίχνευση του πλαισίου δεδομένων, τα MAC πρωτόκολλα και τον έλεγχο σφαλμάτων, διασφαλίζοντας την αξιοπιστία του point-point και του point-multi point πρωτοκόλλου. Το επίπεδο ζεύξης δεδομένων αφορά στην τοπική παράδοση δεδομένων μεταξύ συσκευών στο ίδιο δίκτυο. Τα πλαίσια σύνδεσης δεδομένων, όπως ονομάζονται αυτές οι μονάδες δεδομένων πρωτοκόλλου, δεν ξεπερνούν τα όρια ενός τοπικού δικτύου. Η δρομολόγηση μεταξύ δικτύων και η καθολική διεύθυνση είναι λειτουργίες υψηλότερου επιπέδου, επιτρέποντας έτσι στα πρωτόκολλα σύνδεσης δεδομένων να εστιάζουν στην τοπική παράδοση των δεδομένων. Με αυτόν τον τρόπο, το επίπεδο ζεύξης δεδομένων είναι ανάλογο με έναν αστυνομικό κυκλοφορίας σε μια διασταύρωση, προσπαθεί να δρομολογήσει με την σωστή προτεραιότητα τα δεδομένα που διεκδικούν την πρόσβαση σε ένα κοινό μέσο, χωρίς να ανησυχούν για τον τελικό προορισμό τους. Όταν οι συσκευές επιχειρούν να χρησιμοποιήσουν ένα μέσο ταυτόχρονα, συμβαίνουν συγκρούσεις δεδομένων. Τα πρωτόκολλα ζεύξης δεδομένων καθορίζουν τον τρόπο με τον οποίο οι συσκευές εντοπίζουν και ανακάμπτουν από τέτοιες συγκρούσεις και ενδέχεται να παρέχουν μηχανισμούς για τη μείωση ή την πρόληψή τους.

1.1.5 Φυσικό επίπεδο

Το φυσικό επίπεδο είναι ένα θεμελιώδες στρώμα που βασίζεται στις λογικές δομές δεδομένων των λειτουργιών υψηλότερων επιπέδων σε ένα δίκτυο. Λόγω της πληθώρας των διαθέσιμων τεχνολογιών που παρέχονται από το hardware με πολύ διαφορετικά χαρακτηριστικά, αυτό είναι ίσως το πιο πολύπλοκο επίπεδο στην αρχιτεκτονική OSI.

Το φυσικό επίπεδο παρέχει μια διεπαφή για τη μετάδοση ροής δυαδικών ψηφίων μέσω του φυσικού μέσου. Είναι υπεύθυνο για την επιλογή της συχνότητας, την παραγωγή της συχνότητας του φορέα, την ανίχνευση του σήματος, τη διαμόρφωση και την κρυπτογράφηση δεδομένων. Η παραγωγή συχνοτήτων και η αποτροπή του σήματος έχουν να κάνουν περισσότερο με τον υποκείμενο σχεδιασμό των κόμβων και των πομποδεκτών. Είναι γνωστό ότι η ασύρματη επικοινωνία σε μεγάλες αποστάσεις μπορεί να είναι δαπανηρή, τόσο από πλευράς ενέργειας όσο και πολυπλοκότητας υλοποίησης. Κατά τον

σχεδιασμό του φυσικού επιπέδου για τα δίκτυα αισθητήρων, η ελαχιστοποίηση της ενέργειας αποκτά σημαντική σημασία. Για παράδειγμα, η επικοινωνία multihop σε ένα ΑΔΑ μπορεί να ξεπεράσει αποτελεσματικά τις επιδράσεις σκίασης και απώλειας διαδρομής, εάν η πυκνότητα διαμοιρασμού των κόμβου είναι αρκετά υψηλή.

1.2 Τοπολογία

Ένας μεγάλος αριθμός μη προσβάσιμων και μη εποπτευόμενων αισθητήρων, οι οποίοι μπορούν εύκολα να καταρρεύσουν ανά πασά στιγμή, καθιστούν τη διαχείριση μιας τοπολογίας δικτύου μια σημαντική πρόκληση. Οι κόμβοι μπορούν είτε να διασκορπιστούν μαζικά είτε να τοποθετηθούν ένας προς έναν στο χώρο. Ο μεγάλος αριθμός αισθητήρων, καθώς και η αδυναμία παρακολούθησης όλων των αισθητήρων, οδηγεί στην ανάγκη τοποθέτησης των αισθητήρων σύμφωνα με έναν προσεκτικά μελετημένο σχεδιασμό. Η αρχική εγκατάσταση πρέπει να πληροί ορισμένα κριτήρια:

- Ελάχιστο κόστος εγκατάστασης
- Εξάλειψη της ανάγκης για οποιαδήποτε προ-οργάνωση ή προ-σχεδιασμό
- Ευέλικτη τοποθέτηση
- Ανοχή σε σφάλματα

Μετά την τοποθέτηση, οι αλλαγές στην τοπολογία οφείλονται σε αλλαγές στους κόμβους όπως:

- Θέση
- Δυνατότητα επικοινωνίας
- Διαθέσιμη ενέργεια
- Δυσλειτουργία
- Λεπτομέρειες σχετικά με το σκοπό για τον οποίο εγκαταστάθηκαν

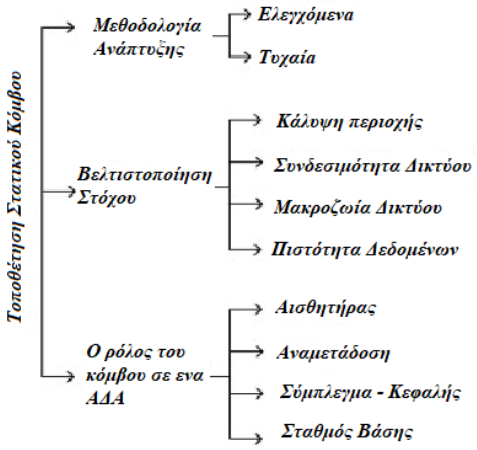
Οι κόμβοι μπορούν να εγκατασταθούν και στατικά. Οι αποτυχίες είναι ένα συνηθισμένο φαινόμενο λόγω έλλειψης ενέργειας ή καταστροφής. Είναι επίσης πιθανό να έχουμε δίκτυα αισθητήρων των οποίων οι κόμβοι συνεχώς κινούνται. Εκτός από τα προβλήματα, τα οποία

είναι φυσικό να αντιμετωπίζουν εξαιτίας των χαρακτηριστικών τους, είναι επίσης δυνατό να έχουμε και δολιοφθορές. Αποτέλεσμα όλων των παραπάνω είναι οι τοπολογίες των δικτύων αισθητήρων να υπόκεινται σε συχνές αλλαγές. (Παπαβασιλείου, 2011)

Η προσθήκη νέων κόμβων στο δίκτυο δημιουργεί την ανάγκη για αναδιοργάνωση. Προκειμένου να αντιμετωπίσουμε τις συχνές αλλαγές στην τοπολογία ενός ΑΔΑ, το οποίο αποτελείται από ένα μεγάλο αριθμό κόμβων με μεγάλους περιορισμούς στην κατανάλωση ενέργειας, χρειαζόμαστε ειδικά σχεδιασμένα πρωτόκολλα δρομολόγησης. (Παπαβασιλείου, 2011)

1.2.1 Τοποθέτηση των Κόμβων

Όπως ήδη αναφέρθηκε, η τοποθεσία των κόμβων έχει σημαντική επίδραση στην αποτελεσματικότητα του ΑΔΑ και στην αποτελεσματικότητα της λειτουργίας του. Οι σύγχρονες στρατηγικές και τεχνικές τοποθέτησης κόμβων πριν από την εκκίνηση του δικτύου, βασίζουν συνήθως την επιλογή τους, για τις θέσεις των συγκεκριμένων κόμβων, σε μετρήσεις που είναι ανεξάρτητες από την κατάσταση του δικτύου ή υποθέτουν ένα σταθερό μοτίβο λειτουργίας που παραμένει αμετάβλητο καθ' όλη τη διάρκεια της λειτουργίας του. Παραδείγματα τέτοιων στατικών μετρήσεων είναι μεταξύ άλλων, η κάλυψη περιοχής και η απόσταση μεταξύ κόμβων. Τα μοντέλα λειτουργίας στατικού δικτύου συχνά αναλαμβάνουν περιοδική συλλογή δεδομένων σε προκαθορισμένες διαδρομές. Τα ταξινομούμε σύμφωνα με τη μεθοδολογία ανάπτυξης, τον βέλτιστο στόχο της τοποθέτησης και τους ρόλους των κόμβων. Με τον όρο μεθοδολογία ανάπτυξης αναφερόμαστε στους τρόπους που μπορούμε να αναπτύξουμε ένα δίκτυο όπως αναφέραμε και στο προηγούμενο κεφάλαιο, *Τοπολογία* . Ο βέλτιστος στόχος χωρίζεται στις λειτουργίες που θέλουμε να επιτύχουμε με το δίκτυο αισθητήρων μας, όπως για παράδειγμα η κάλυψη μιας περιοχής και η καλύτερη σύνδεση μεταξύ των κόμβων. Τέλος, ο στόχος του κάθε κόμβου εξαρτάται από τους αισθητήρες που τοποθετούνται στον κόμβο και τις λειτουργίες που θέλουμε να επιτύχουμε. Συνοπτικά φαίνονται στην παρακάτω Εικόνα 2. (Mohamed Younisa, June 2008, pp. 621-655)



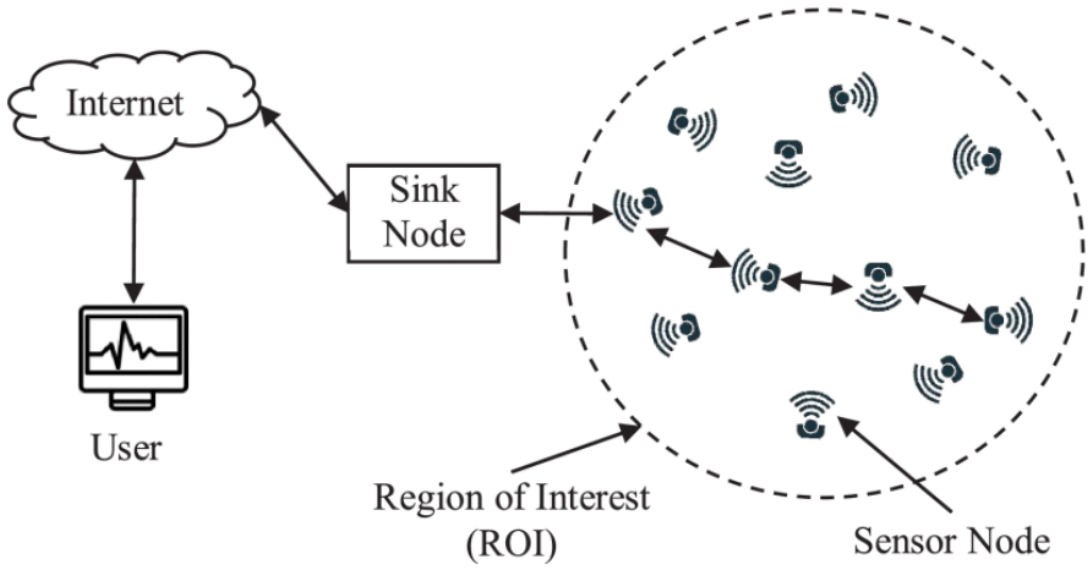
Εικόνα 2 Διαφορετικές ταξινομήσεις στατικών στρατηγικών για τοποθέτηση κόμβων στο ΑΔΑ

1.2.2 Κόμβος Sink

Μαζί με τις αξιοσημείωτες εξελίξεις στα ΑΔΑ, έχουν εμφανιστεί μεγάλης κλίμακας ασύρματα δίκτυα αισθητήρων, τα οποία χρησιμοποιούνται στην καθημερινή μας ζωή για παρακολούθηση, ανίχνευση, μέτρηση και συλλογή δεδομένων σε πραγματικό χρόνο. Τα

δεδομένα που συλλέγονται από αυτούς τους κόμβους μεταδίδονται σε έναν κόμβο sink ο οποίος μεταφέρει την πληροφορία σε κεντρικούς κόμβους ή τελικούς χρήστες.

Όπως φαίνεται και στην παρακάτω Εικόνα 3, ένας κόμβος sink είναι ένα προσωρινό μέρος επεξεργασίας ή ανακατεύθυνσης δεδομένων, με καταληκτικό προορισμό τον τελικό χρήστη.



Εικόνα 3 Τοπολογία Sink Node (Hassaballah, 14 July 2021)

Παρόμοια με άλλα προβλήματα βελτιστοποίησης που δημιουργούνται από την ανάπτυξη ενός ΑΔΑ, ο εντοπισμός του κόμβου sink είναι επίσης μια πρόκληση, καθώς ο καθορισμός της καλύτερης θέσης του κόμβου sink, σημαίνει μείωση των βημάτων που θα κάνει ένα μήνυμα από έναν κόμβο αισθητήρα σε ένα κόμβο sink . (E. Tuba, Jun. 2018)

Κάτι τέτοιο μειώνει κατά πολύ την κατανάλωση ενέργειας του κόμβου καθώς η διαδικασία αποστολής και παραλαβής των δεδομένων καταναλώνει πολύ ενέργεια. Έτσι , επιλέγοντας τη βέλτιστη θέση τοποθέτησης ενός κόμβου sink, θα επιμηκυνθεί ο χρόνος λειτουργίας του.

Γενικότερα, ο έλεγχος τοπολογίας είναι μια τεχνική, που χρησιμοποιείται ευρέως σε κατανεμημένους υπολογιστές, για την πραγματοποίηση κάποιων αλλαγών στο υποκείμενο δίκτυο που μπορούν να μοντελοποιηθούν ως γράφημα για τη μείωση του κόστους των κατανεμημένων αλγορίθμων από τα νέα γραφήματα που προκύπτουν. Υπάρχουν πολλοί διαφορετικοί τρόποι για τον έλεγχο μιας τοπολογίας, όπως η αλλαγή του εύρους μετάδοσης

των κόμβων, η απενεργοποίηση κόμβων από το δίκτυο, η ομαδοποίηση και η προσθήκη νέων κόμβων για τη διατήρηση της συνδεσιμότητας.

Η επιλογή της βέλτιστης θέσης ενός κόμβου sink στα ΑΔΑ βασίζεται σε ένα σύνολο κριτηρίων: (M. M. Fouad, Jul. 2015.)

- Ο αριθμός των γειτόνων γύρω από τον κόμβο του sink
- Η υπολειμματική ενέργεια των γειτόνων του κόμβου του sink
- Η υπολειμματική ενέργεια του ίδιου του κόμβου sink

Τέλος, ο κόμβος sink μπορεί να είναι στατικός ή κινητός και μπορεί να τοποθετηθεί σε διαφορετικές τοποθεσίες μέσα στο ΑΔΑ. Στην περίπτωση ενός στατικού sink, οι κόμβοι που βρίσκονται κοντά του αποφορτίζουν την ενέργειά τους πολύ νωρίτερα σε σύγκριση με τους κόμβους που βρίσκονται πιο μακριά από το sink, λόγω υψηλότερου φορτίου αναμετάδοσης δεδομένων. Για να αντιμετωπιστεί αυτό το ζήτημα, έχει εισαχθεί η κινητοποίηση του κόμβου sink, όπου κινείται κατά μήκος μιας συγκεκριμένης διαδρομής μέσω του δικτύου. Έχει αποδειχθεί επίσης ότι, στις περισσότερες περιπτώσεις, η κινητικότητα του sink βοηθά στην εξισορρόπηση του φορτίου δρομολόγησης και, ως εκ τούτου, της απορρόφησης ενέργειας των κόμβων (J. Luo, 2005, pp. 1735-1746), (A. Giannakos, 2009, pp. 289-291).

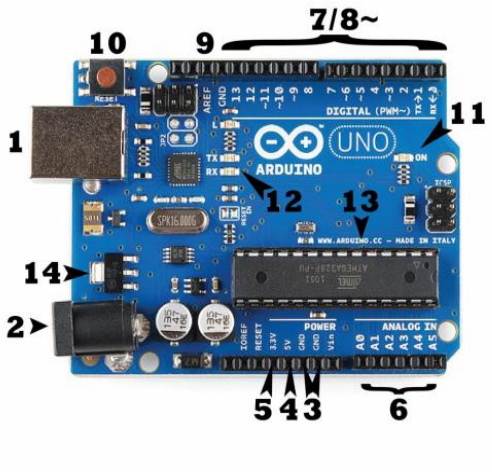
Αν και είναι σαφές ότι η κινητικότητα του sink βελτιώνει την εξισορρόπηση φορτίου μεταξύ των κόμβων, δεν έχει απαντηθεί ακόμα αν αυτό οδηγεί επίσης σε βελτιώσεις στην ενεργειακή απόδοση ενός ασυρμάτου δικτύου δεδομένων.

2. Εξοπλισμός

Για τη δημιουργία ενός ασυρμάτου δικτύου δεδομένων όπως αναφέραμε πρωτύτερα χρειαζόμαστε τον κατάλληλο εξοπλισμό. Αντικείμενο αυτής της εργασίας είναι η ανάπτυξη ενός ΑΔΑ, οπού οι αισθητήρες θα στέλνουν τα δεδομένα που συλλέγουν σε έναν τελικό χρήστη. Το ΑΔΑ που αναπτύσσεται στην εργασία αυτή αποτελείται από δυο κόμβους, ο ένας εκ των οποίων αποτελεί τον κόμβο συλλογής των δεδομένων ή αλλιώς κόμβο sink. Κάθε κόμβος, απαρτίζεται από έναν ραδιοπομποδέκτη με σύνδεση σε μια κεραία NRF24L01 - 2.4G Wireless Transceiver Module για την αποστολή και την παραλαβή των δεδομένων. Επιπλέον, διαθέτουν από έναν μικροελεγκτή Arduino UNO, για τον προγραμματισμό του κυκλώματος, καθώς και αισθητήρες, που στην περίπτωση μας είναι ένας HC-SR04 Ultrasonic για τη μέτρηση της απόστασης και ένας Triple-axis Accelerometer+Magnetometer για τον υπολογισμό της κλίσης και της περιστροφής του κόμβου. Τέλος, στον κόμβο sink χρησιμοποιήσαμε έναν ESP8266-01 - Wifi Module για την επικοινωνία μέσω ασυρμάτου δικτύου WiFi με τον τελικό χρήστη.

2.1 Arduino Uno

Η πλακέτα Arduino uno είναι εξοπλισμένη αντίστοιχα με το δικό της σετ ψηφιακών και αναλογικών εισόδων/εξόδων ακίδων που μπορούν να διασυνδεθούν με διάφορες πλακέτες επέκτασης και άλλα κυκλώματα. Η πλακέτα έχει 14 ψηφιακές και 6 αναλογικές ακίδες και προγραμματίζεται με το IDE(integrated Development Environment) του arduino μέσω καλωδίου τύπου Β. Έχει τη δυνατότητα να τροφοδοτείται μέσω του καλωδίου USB η μέσω μίας 9-volt μπαταρίας , αν και δέχεται τάσεις μεταξύ 7 και 20 βολτ. Εικόνα 4



Εικόνα 4 Εξοπλισμός Arduino UNO

Το arduino uno εχει τα χαρακτηριστικά που αναφέρονται στον Πίνακας 1 και αποτελείται από:

1. Σύνδεση USB (ο τρόπος με τον οποίο φορτώνουμε τον κώδικα στο arduino)
2. Υποδοχή κυλίνδρου (είσοδος φορτιστή τοίχου)
3. Πηνία GND (γείωση του κυκλώματος)
4. Πηνίο 5V (παροχή 5V ισχύος στο κύκλωμα)
5. Πηνίο 3.3V(παροχή 3.3V ισχύος στο κύκλωμα)
6. Αναλογικές θύρες
7. Ψηφιακές θύρες
8. PWM(Οι ακίδες αυτές προσομοιώνουν την αναλογική έξοδο. Επιπλέον, οι ακίδες με την περισπωμένη(~) λειτουργούν και ως κανονικές ψηφιακές ακίδες .
9. AREF(Οι ακίδες αυτές υποστηρίζουν αναλογική αναφορά. Χρησιμοποιείται μερικές φορές για να ρυθμίσετε μια εξωτερική τάση αναφοράς (μεταξύ 0 και 5 Volt) ως το ανώτερο όριο για τους ακροδέκτες αναλογικής εισόδου.)
10. Κουμπί επαναφοράς
11. Δείκτης LED ισχύον (ανάβει κάθε φορά που συνδέεται σε πηγή ενέργειας.)
12. LED TX RX (Το TX είναι σύντομο για τη μετάδοση, το RX είναι σύντομο για λήψη.)
13. CHIP IC (Το τσιπάκι IC είναι το ‘μυαλό’ του arduino)
14. Ρυθμιστής τάσης

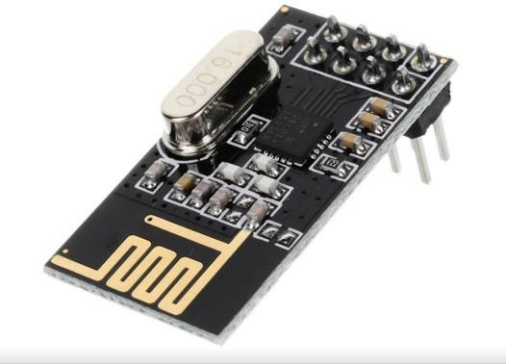
Χαρακτηριστικά του	**Arduino**
Microcontroller	ATmega328

Τάση λειτουργίας	5V
Τάση εισόδου	7-12V
Τάση εισόδου (όριο)	6-20V
Digital I/O Pins	14 (εκ των οποίων 6 περιέχουν PWM εξόδους)
Analog Input Pins	6
DC ρεύματος I/O Pin	40 mA
DC τρέχουσα για 3.3V Pin	50 mA
Flash Memory: 32 KB εκ των οποίων 0,5 KB που χρησιμοποιούνται από τον bootloader	2 KB
SRAM	1 KB
Clock Speed	16 MHz

Πίνακας 1 Χαρακτηριστικά Arduino UNO

2.2 NRF24L01 - 2.4G Wireless Transceiver Module

Το NRF24L01 (Εικόνα 5), είναι μια ασύρματη μονάδα χαμηλής ισχύος που χρησιμοποιεί ένα τσιπ NRF24L01 από την εταιρία Nordic. Χρησιμοποιεί το ευρος ζώνης των 2,4 GHz και μπορεί να λειτουργήσει με ρυθμούς baud από 250 kbps έως 2 Mbps. Εάν χρησιμοποιείται σε ανοιχτό χώρο και με χαμηλότερο ρυθμό baud, η εμβέλειά του μπορεί να φτάσει τα 100 μέτρα.



Εικόνα 5 NRF24L01 μονάδα

Η μονάδα, μπορεί να χρησιμοποιήσει 125 διαφορετικά κανάλια που δίνουν τη δυνατότητα να έχουμε ένα δίκτυο με 125 μόντεμ που λειτουργούν ανεξάρτητα σε ένα μέρος. Κάθε κανάλι μπορεί να έχει έως και 6 διευθύνσεις ή κάθε μονάδα μπορεί να επικοινωνεί με έως και 6 άλλες μονάδες ταυτόχρονα. Η κατανάλωση ισχύος αυτής της μονάδας είναι περίπου 12mA κατά τη μετάδοση, η οποία είναι ακόμη χαμηλότερη από ένα μόνο LED. Η τάση λειτουργίας της μονάδας είναι από 1,9 έως 3,6V, αλλά το καλό είναι ότι οι άλλοι ακροδέκτες ανέχονται τάση 5V, έτσι μπορούμε εύκολα να το συνδέσουμε σε ένα Arduino χωρίς να χρησιμοποιήσουμε μετατροπείς επιπέδου τάσης.

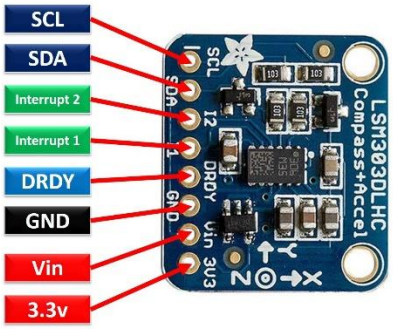
Εικόνα 6 Λειτουργία πηνίων μονάδας

Όπως φαίνεται και στην παραπάνω Εικόνα 6, τρεις από αυτές τις καρφίτσες προορίζονται για την επικοινωνία SPI και πρέπει να συνδεθούν με τις ακίδες SPI του Arduino. Οι ακίδες CSN και CE μπορούν να συνδεθούν σε οποιονδήποτε ψηφιακό πηνίο της πλακέτας Arduino και χρησιμοποιούνται για τη ρύθμιση της μονάδας σε κατάσταση αναμονής ή ενεργής

λειτουργίας, καθώς και για εναλλαγή μεταξύ λειτουργίας μετάδοσης ή εντολής. Το τελευταίο πηνίο είναι ένας πείρος διακοπής που δεν χρειάζεται να χρησιμοποιηθεί πάρα μόνο όταν θέλουμε να επαναφέρουμε την μονάδα στις εργοστασιακές της ρυθμίσεις. (Dejan, n.d.)

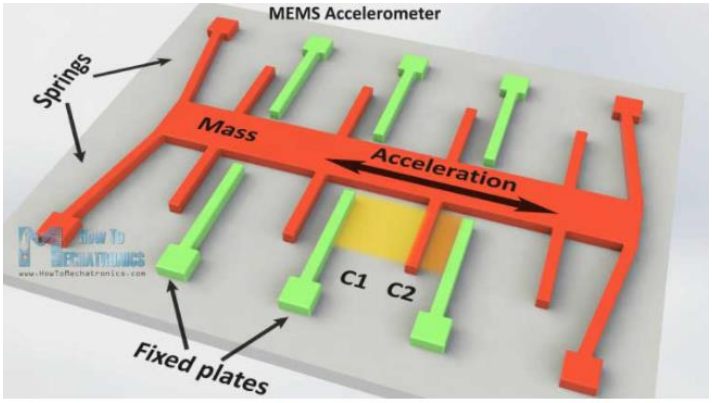
2.3 Triple-axis Accelerometer+Magnetometer

Οι αισθητήρες MEMES είναι πολύ μικρά συστήματα ή συσκευές, αποτελούμενα από μικροεξαρτήματα που κυμαίνονται από 0,001 mm έως 0,1 mm σε μέγεθος. Αυτά τα εξαρτήματα είναι κατασκευασμένα από πυρίτιο, μέταλλα ή / και κεραμικά και συνήθως συνδυάζονται με ένα CPU (Microcontroller) για την ολοκλήρωση του συστήματος.



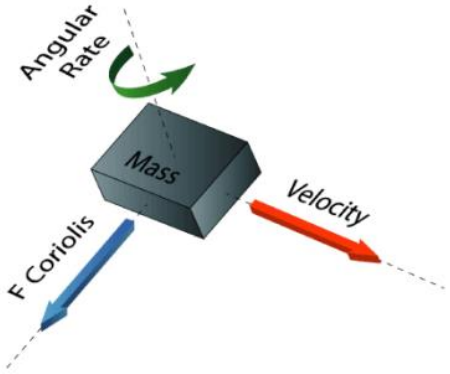
Εικόνα 7 Αισθητήρας Triple-axis Accelerometer+Magnetometer

Το επιταχυνσιόμετρο ενός MEMS(Εικόνα 7), υπολογίζει την επιτάχυνση μετρώντας την αλλαγή της χωρητικότητας. Η μικρο-δομή του μοιάζει κάπως έτσι Εικόνα 8. Έχει μια μάζα συνδεδεμένη με ένα ελατήριο το οποίο περιορίζεται για να κινείται κατά μία κατεύθυνση και σταθερές εξωτερικές πλάκες. Έτσι, όταν εφαρμόζεται επιτάχυνση στη συγκεκριμένη κατεύθυνση, η μάζα θα κινηθεί και η χωρητικότητα μεταξύ των πλακών και της μάζας θα αλλάξει. Αυτή η αλλαγή στην χωρητικότητα θα μετρηθεί, θα υποβληθεί σε επεξεργασία και θα αντιστοιχεί σε μια συγκεκριμένη τιμή επιτάχυνσης.



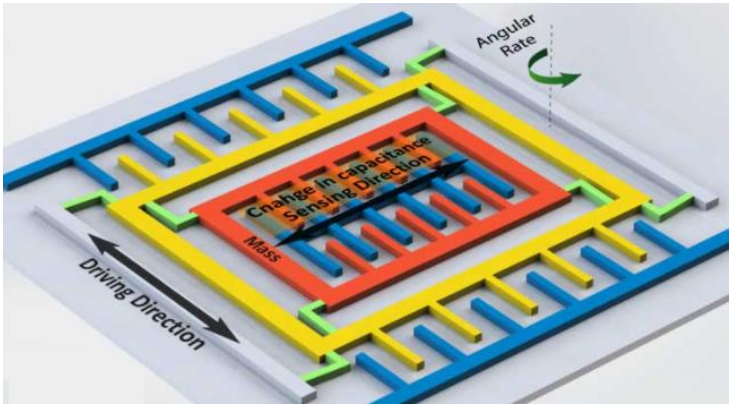
Εικόνα 8 Μικρο-δομή Επιταχυνσιόμετρου

Το γυροσκόπιο μετρά τον γωνιακό ρυθμό χρησιμοποιώντας το Coriolis Effect. Όταν μια μάζα κινείται σε μια συγκεκριμένη κατεύθυνση με μια συγκεκριμένη ταχύτητα και όταν ένας εξωτερικός γωνιακός ρυθμός θα εφαρμοστεί όπως δείχνει η Εικόνα 9 με το πράσινο βέλος, τότε θα εμφανιστεί μια δύναμη, όπως φαίνεται με το μπλε και κόκκινο βέλος, το οποίο θα προκαλέσει κάθετη μετατόπιση της μάζας. Ομοίως με το επιταχυνσιόμετρο, αυτή η μετατόπιση θα προκαλέσει αλλαγή στην χωρητικότητα που θα μετρηθεί, θα υποβληθεί σε επεξεργασία και θα αντιστοιχεί σε ένα συγκεκριμένο γωνιακό ρυθμό.



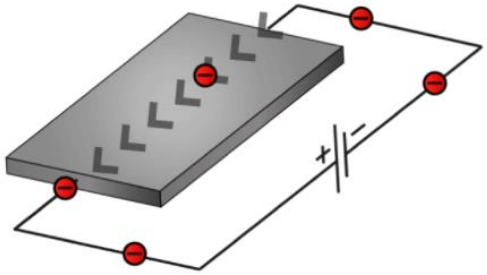
Εικόνα 9 Coriolis Effect

Η μικρο-δομή του γυροσκοπίου μοιάζει κάπως έτσι Εικόνα 10. Μια μάζα που κινείται συνεχώς, ή ταλαντεύεται, και όταν θα εφαρμοστεί ο εξωτερικός γωνιακός ρυθμός, ένα εύκαμπτο μέρος της μάζας θα κινείται και θα κάνει την κάθετη μετατόπιση.



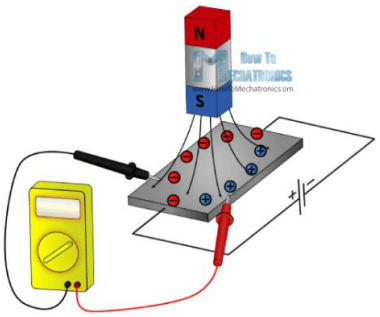
Εικόνα 10 Μικρο-δομή γυροσκοπίου

Τέλος, το μαγνητόμετρο MEMS μετρά το μαγνητικό πεδίο της γης χρησιμοποιώντας το Hall Effect ή το Magneto Resistive Effect.



Εικόνα 11 Μαγνητική πλάκα

Εάν υπάρχει μια αγώγιμη πλάκα όπως φαίνεται στην παραπάνω Εικόνα 11 και ρυθμίζουμε το ρεύμα να ρέει μέσα από αυτήν, τα ηλεκτρόνια θα ρέουν κατ 'ευθείαν από τη μία στην άλλη πλευρά της πλάκας. Τώρα αν φέρουμε κάποιο μαγνητικό πεδίο κοντά στην πλάκα, θα διαταράξουμε την ευθεία ροή και τα ηλεκτρόνια θα εκτρέψουν στη μία πλευρά της πλάκας και τους θετικούς πόλους στην άλλη πλευρά της πλάκας όπως φαίνεται στην Εικόνα 12. Αυτό σημαίνει ότι αν βάλουμε έναν μετρητή τώρα μεταξύ αυτών των δύο πλευρών θα λάβουμε κάποια τάση που εξαρτάται από την ισχύ του μαγνητικού πεδίου και την κατεύθυνση του.



Εικόνα 12 Hall effect

Το άλλο 10% των αισθητήρων στην αγορά χρησιμοποιούν το Magneto-resistive Effect. Αυτοί οι αισθητήρες χρησιμοποιούν υλικά ευαίσθητα στο μαγνητικό πεδίο, συνήθως αποτελούμενα από σίδηρο (Fe) και νικέλιο (Ne). Έτσι, όταν αυτά τα υλικά εκτίθενται σε μαγνητικό πεδίο αλλάζουν την αντίσταση τους. (Dejan, n.d.)

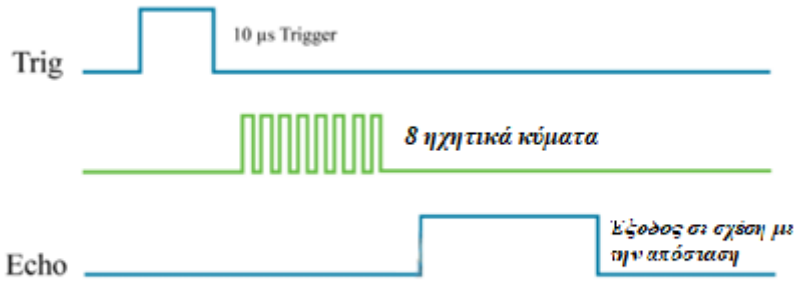
2.4 Ultrasonic Sensor - HC-SR04

Εικόνα 13 Αισθητήρας Ultrasonic - HC-SR04

Ο αισθητήρας απόστασης υπερήχων HC-SR04(Εικόνα 13) είναι ένας αισθητήρας που χρησιμοποιείται για την ανίχνευση της απόστασης από ένα αντικείμενο χρησιμοποιώντας σόναρ. Εκπέμπει έναν υπέρηχο στα 40.000 Hz που ταξιδεύει μέσω του αέρα και εάν υπάρχει αντικείμενο ή εμπόδιο στη διαδρομή του, θα επιστρέψει στον αισθητήρα. Λαμβάνοντας υπόψη τον χρόνο ταξιδιού και την ταχύτητα του ήχου, υπολογίζουμε την απόσταση. Ο αισθητήρας αποτελείται από 4 πηνία , την γείωση , το πηνίο VCC , το πηνίο Trig και το πηνίο Echo. Οι ακροδέκτες γείωσης και VCC της μονάδας πρέπει να συνδεθούν με τη

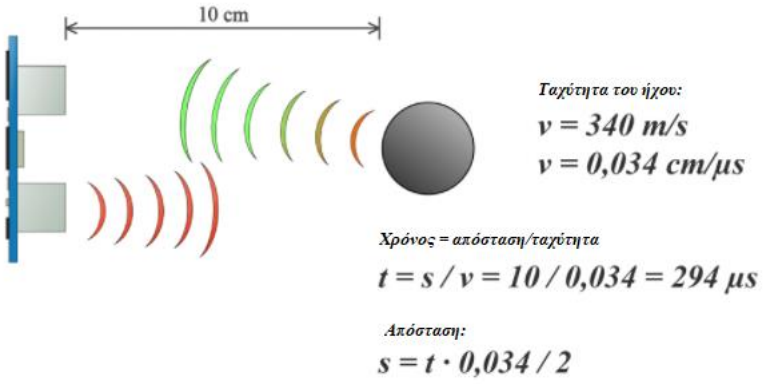
γείωση και τα 5 βολτ στην πλακέτα Arduino. Αντίστοιχα και οι ακροδέκτες trig και echo σε οποιονδήποτε ψηφιακό ακροδέκτη I / O στην πλακέτα.

Για να δημιουργήσετε τον υπέρηχο πρέπει να ρυθμίσουμε το Trig σε υψηλή κατάσταση για 10 μs. Αυτό θα στείλει 8 ηχητικά κύματα τα οποία θα ταξιδέψουν με την ταχύτητα του ήχου και θα ληφθούν στον ακροδέκτη Echo. Το πηνίο Echo θα υπολογίσει το χρόνο σε μικροδευτερόλεπτα που διανύθηκαν από τα ηχητικά κύματα όπως φαίνεται και στην παρακάτω Εικόνα 14.



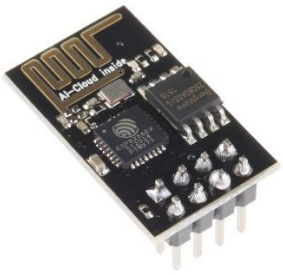
Εικόνα 14 Λειτουργία αισθητήρα

Για παράδειγμα, εάν το αντικείμενο απέχει 10 cm από τον αισθητήρα και η ταχύτητα του ήχου είναι 340 m / s ή 0,034 cm / μs, το ηχητικό κύμα θα χρειαστεί να ταξιδέψει περίπου 294 μικροδευτερόλεπτα. Αλλά αυτό που θα λάβουμε από τον ακροδέκτη Echo θα είναι διπλάσιο από αυτόν τον αριθμό επειδή το ηχητικό κύμα πρέπει να ταξιδέψει προς τα εμπρός και να αναπηδήσει προς τα πίσω. Επομένως, για να φτάσουμε την απόσταση σε cm, πρέπει να πολλαπλασιάσουμε την τιμή του λαμβανόμενου χρόνου ταξιδιού από τον ακροδέκτη με 0,034 και να τη διαιρέσουμε με 2 για να καταλήξουμε στο επιθυμητό αποτέλεσμα. (Dejan, n.d.)(Εικόνα 15)



Εικόνα 15 Υπολογισμός απόστασης

2.5 ESP8266-01 - Wifi Module



Εικόνα 16 Μονάδα ESP8266-01

Το ESP8266(Εικόνα 16) είναι ένα μικροτσίπ Wi-Fi χαμηλού κόστους, με πλήρη πρωτόκολλα TCP / IP που παράγεται από την Espressif Systems. Η μονάδα ESP8266 Wifi μπορεί να δώσει σε οποιονδήποτε μικροελεγκτή πρόσβαση στο διαδίκτυο. Κάθε μονάδα ESP8266 έρχεται προ-προγραμματισμένη με ένα υλικολογισμικό σετ εντολών AT(Εικόνα 17) το οποίο σου επιτρέπει να επικοινωνείς με το μικροτσίπ και να το προγραμματίζεις. Αυτή η μονάδα διαθέτει μια αρκετά ισχυρή δυνατότητα επεξεργασίας και αποθήκευσης που της επιτρέπει να ενσωματώνεται με τους αισθητήρες και άλλες ειδικές συσκευές μέσω των ακίδων GPIO χωρίς να επηρεάζεται η λειτουργία του τσιπ.

Commands	Description	Type
AT+RST	restart module	basic
AT+CWMODE	wifi mode	wifi
AT+CWJAP	join AP	wifi
AT+CWLAP	list AP	wif
AT+CWQAP	quit AP	wifi
AT+CIPSTATUS	get status	TCP/IP
AT+CIPSTART	set up TCP or UDP	TCP/IP
AT+CIPSEND	send data	TCP/IP
AT+CIPCLOSE	close TCP or UDP	TCP/IP
AT+CIFSR	get IP	TCP/IP
AT+CIPMUX	set multiple connections	TCP/IP
AT+CIPSERVER	set as server	TCP/IP

Εικόνα 17 Εντολές AT

Η μέγιστη τάση λειτουργίας της μονάδας είναι 3.3v, αλλά το καλό είναι ότι οι άλλοι ακροδέκτες ανέχονται τάση 5V, έτσι μπορούμε εύκολα να το συνδέσουμε σε ένα Arduino χωρίς να χρησιμοποιήσουμε μετατροπείς επιπέδου τάσης. Το ESP8266-01 είναι η μικρότερη μονάδα ESP8266 και έχει μόνο 8 ακίδες. Από αυτά τα VCC, GND, RST (reset) και CH_PD (chip select) δεν είναι καρφίτσες I/O αλλά χρειάζονται για τη λειτουργία της μονάδας. Αυτό αφήνει τα GPIO0, GPIO2, TX και RX διαθέσιμα πηνία εισόδου/εξόδου, αλλά ακόμη και αυτά έχουν προκαθορισμένες λειτουργίες. Τα GPIO0 και GPIO2 καθορίζουν σε ποια λειτουργία ξεκινά η μονάδα και οι ακίδες TX/RX χρησιμοποιούνται για τον προγραμματισμό της μονάδας και για το Serial I/O, που χρησιμοποιείται συνήθως για τον εντοπισμό σφαλμάτων. Τέλος, τα GPIO0 και GPIO2 πρέπει να έχουν συνδεδεμένες αντιστάσεις για να διασφαλίσουν ότι η μονάδα ξεκινά σωστά. (instructables, 2nd April 2018)

3. Εφαρμογή Συλλογής Δεδομένων

Για την συλλογή των δεδομένων που λαμβάνουμε από το ΑΔΑ χρειάστηκε η δημιουργία μιας εφαρμογής. Για την δημιουργία της εφαρμογής επιλέχθηκε η πλατφόρμα Blynk, η

οποία μας επιτρέπει να δημιουργήσουμε γρήγορα διεπαφές για τον έλεγχο και την παρακολούθηση των αισθητήρων μας από μια συσκευή iOS ή Android. Μετά τη λήψη της εφαρμογής Blynk, μπορούμε να δημιουργήσουμε έναν πίνακα ελέγχου της εργασίας μας και να προσθέσουμε κουμπιά ή γραφήματα (widget) στην οθόνη. Χρησιμοποιώντας τα widget, μπορούμε να ενεργοποιήσουμε και να απενεργοποιήσουμε τα πηνία σε έναν μικροελεγκτή ή/και να εμφανίσουμε δεδομένα από τους αισθητήρες.

Υπάρχουν τρία βασικά στοιχεία στην πλατφόρμα:

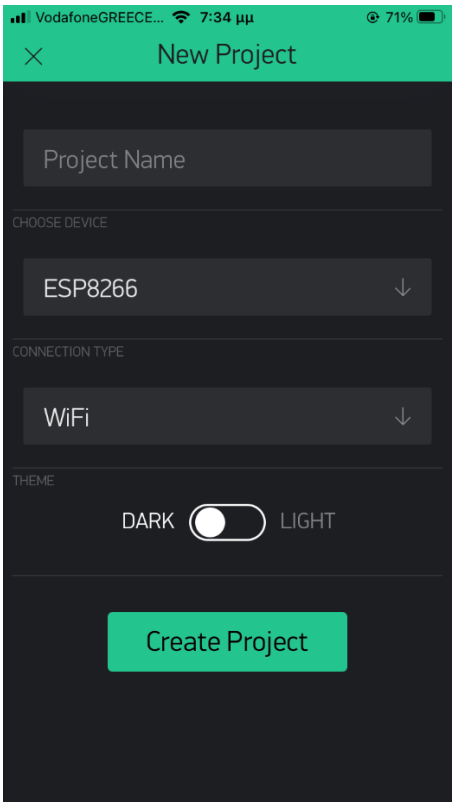
1. Blynk App: μας επιτρέπει να δημιουργήσουμε διεπαφές για τις εργασίες μας χρησιμοποιώντας διάφορα widget που παρέχονται.

2. Blynk Server: μας επιτρέπει να ρυθμίσουμε τις επικοινωνίες μεταξύ smartphone και hardware. Επιπρόσθετα, μπορούμε να χρησιμοποιήσουμε το Blynk Cloud ή να δημιουργήσουμε τοπικά τον ιδιωτικό μας διακομιστή Blynk.

3. Βιβλιοθήκες Blynk: μας επιτρέπουν την επικοινωνία με τον διακομιστή και επεξεργάζονται όλες τις εισερχόμενες και εξερχόμενες εντολές. (Blynk, n.d.)

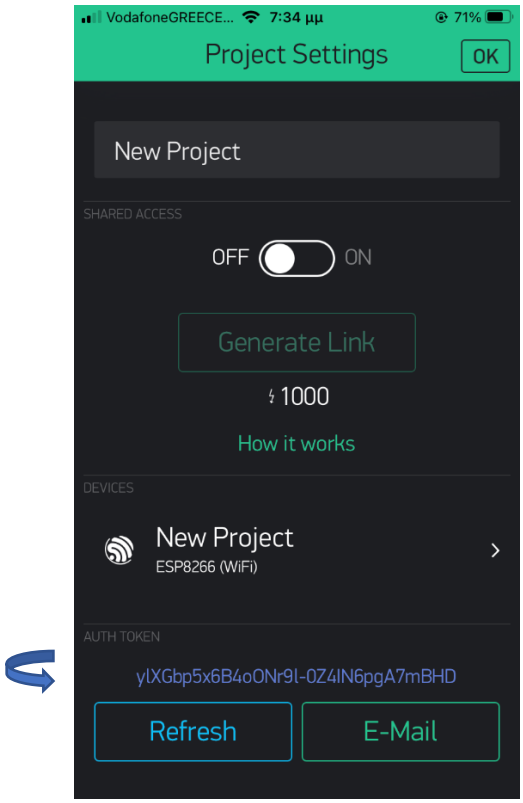
3.1 Δημιουργία Εφαρμογής Blynk

Αφού κατεβάσουμε την εφαρμογή Blynk, θα χρειαστεί να δημιουργήσουμε έναν λογαριασμό για να μπορούμε να αποθηκεύουμε τις εργασίες μας. Αφού συνδεθούμε με επιτυχία στο λογαριασμό μας, ξεκινάμε δημιουργώντας ένα νέο έργο. Επιλέγουμε την συσκευή με την οποία θα επικοινωνεί η εφαρμογή μας, στην περίπτωση μας είναι το ESP8266-01, και επιλέγουμε να γίνει η σύνδεση μέσω WiFi όπως φαίνεται και στην Εικόνα 18.



Εικόνα 18 Επιλογή Συσκευής ESP8266-01

Μόλις πατήσουμε δημιουργία, θα μας αποσταλεί στο e-mail που καταχωρήσαμε πιο πάνω ένα Authentication Token. Το Auth Token(Εικόνα 19), είναι ένα μοναδικό αναγνωριστικό που απαιτείται για τη σύνδεση της εργασίας μας στην τελική συσκευή μας.



Εικόνα 19 Auth Token

Τέλος, προσθέτουμε τα απαραίτητα widgets για την εργασία μας και θέτουμε τα πηνία από τα οποία θα δέχονται τα δεδομένα και τρέχουμε την εφαρμογή όπως φαινεται στην Εικόνα 20. (Blynk, n.d.)

Εικόνα 20 Επιλογή Widget εργασίας

4. Ανάπτυξη Ασύρματου Δικτύου Αισθητήρων

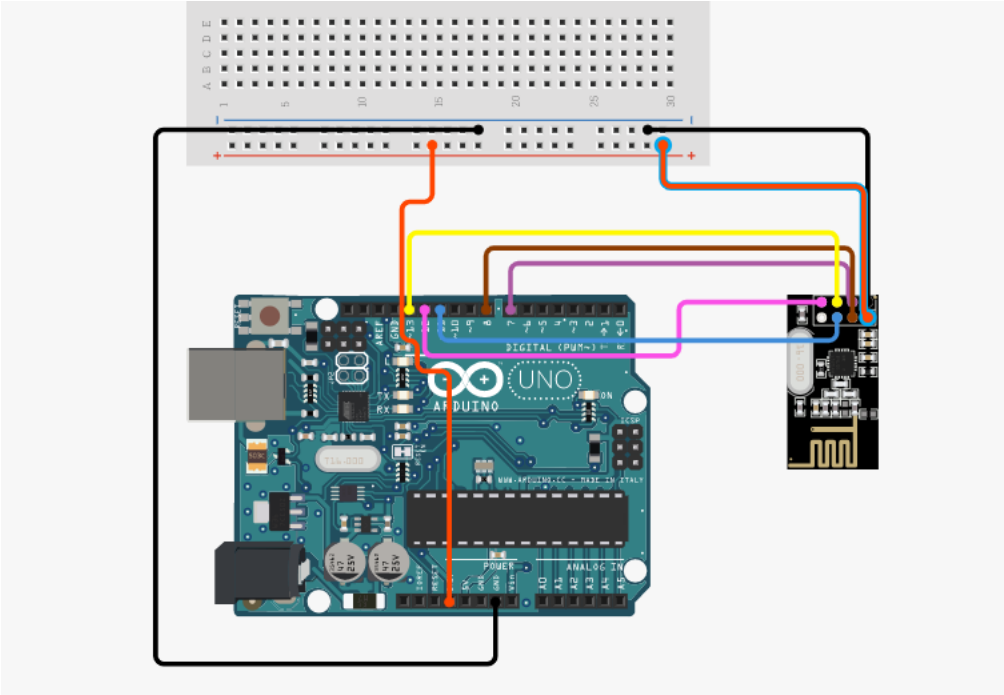
Η ανάπτυξη ενός ΑΔΑ βασίζεται τόσο στη σωστή συνδεσμολογία των αισθητήρων με τον κόμβο όσο και στον κατάλληλο προγραμματισμό των κόμβων για την σωστή λειτουργία του δικτύου. Στα επόμενα υποκεφάλαια θα αναφερθούμε στην συνδεσμολογία των κόμβων, τον προγραμματισμό των συσκευών και στην εφαρμογή του δικτύου.

4.1 Συνδεσμολογία

Όπως αναφέραμε και στο κεφάλαιο 2. Εξοπλισμός, η εργασία αυτή αποτελείται από δυο κόμβους, οπού επικοινωνούν μεταξύ τους και στέλνουν τα δεδομένα τους σε ένα τελικό χρήστη. Κάθε κόμβος, απαρτίζεται από έναν ραδιοπομποδέκτη με σύνδεση σε μια κεραία NRF24L01 - 2.4G Wireless Transceiver Module για την αποστολή και την παραλαβή των δεδομένων. Η συνδεσμολογία της κεραίας έχει ως εξής(Εικόνα 21):

- NRF24L01 CE to Arduino Uno 7
- NRF24L01 CS to Arduino Uno 8

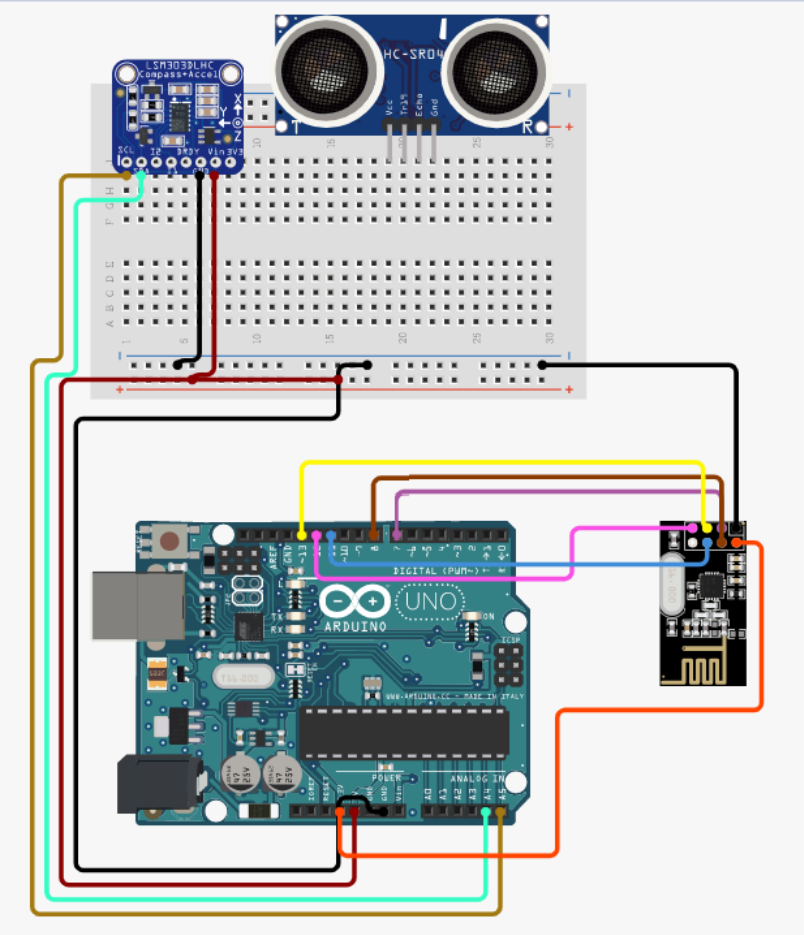
- NRF24L01 GND to Arduino Uno GND
- NRF24L01 MISO to Arduino Uno 12
- NRF24L01 MOSI to Arduino Uno 11
- NRF24L01 SCK to Arduino Uno 13
- NRF24L01 VCC to Arduino Uno 3.3v



Εικόνα 21 Συνδεσμολογία NRF24L01 - 2.4G Wireless Transceiver Module

Έναν αισθητήρα Triple-axis Accelerometer+Magnetometer για τον υπολογισμό της κλίσης και της περιστροφής του κόμβου.

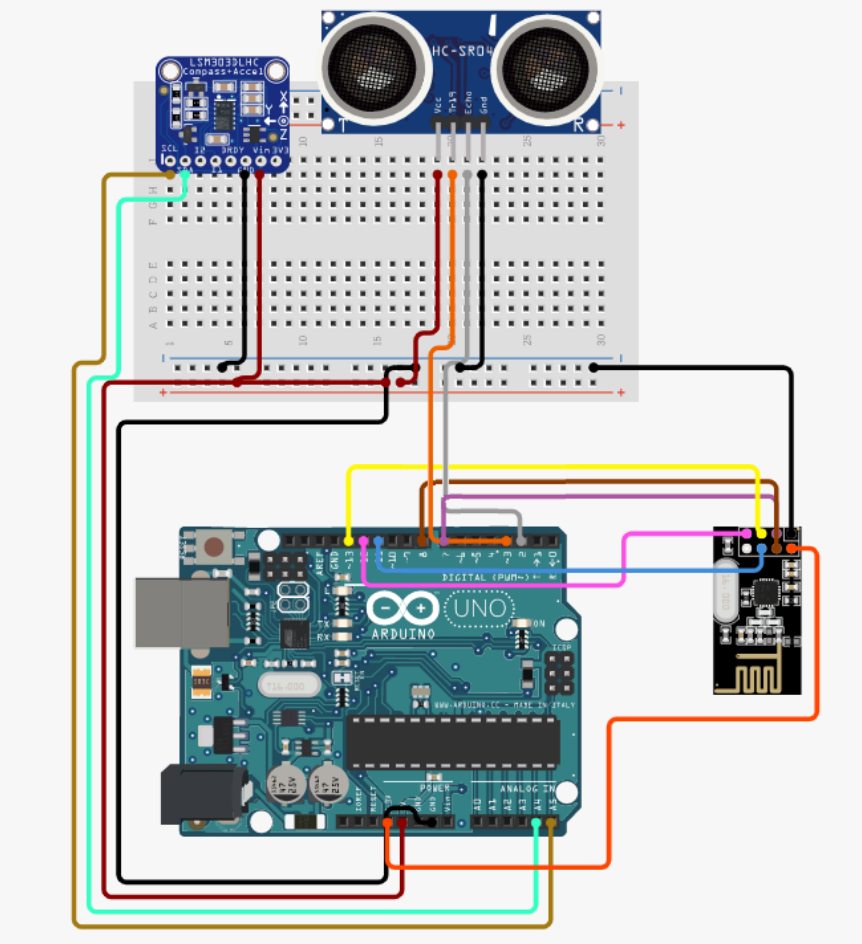
- GY-521 GND to Bus GND
- GY-521 SCL to Arduino Uno A5
- GY-521 SDA to Arduino Uno A4
- GY-521 VIN to Bus POS(5V)



Εικόνα 22 Συνδεσμολογία Triple-axis Accelerometer+Magnetometer

Έναν αισθητήρα HC-SR04 Ultrasonic για τη μέτρηση της απόστασης

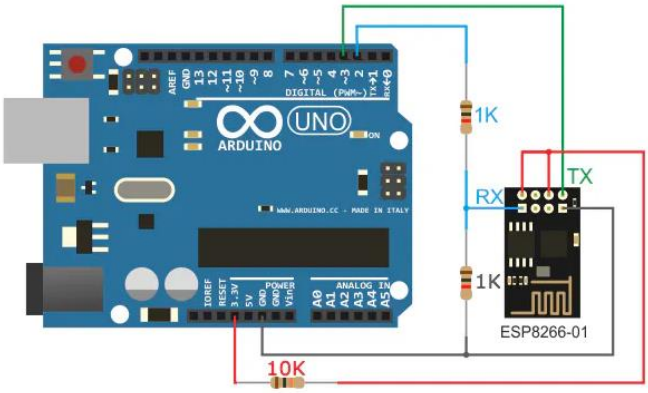
- HCSR04 ECHO to Arduino Uno 2
- HCSR04 GND to Bus GND
- HCSR04 TRIG to Arduino Uno 3
- HCSR04 VCC to Bus POS(5V)



Εικόνα 23 Συνδεσμολογία HC-SR04 Ultrasonic Sensor

Τέλος, ο κόμβος sink αποτελείται από το ίδιο ακριβώς κύκλωμα αλλά περιέχει ένα ακόμα εξάρτημα το οποίο είναι το ESP8266-01 - Wifi Module για την επικοινωνία μέσω ασυρμάτου δικτύου WiFi με τον τελικό χρήστη και συνδέεται ως εξής:

- ESP8266 RXD to 1K resistor then to Arduino Uno 2
- ESP8266 RXD to 1K resistor then to Arduino Uno GND
- ESP8266 TXD to Arduino Uno 3
- ESP8266 Power Pin and Enable Pin to 10K resistor then to Arduino Uno 3.3v
- ESP8266 GND to Bus GND



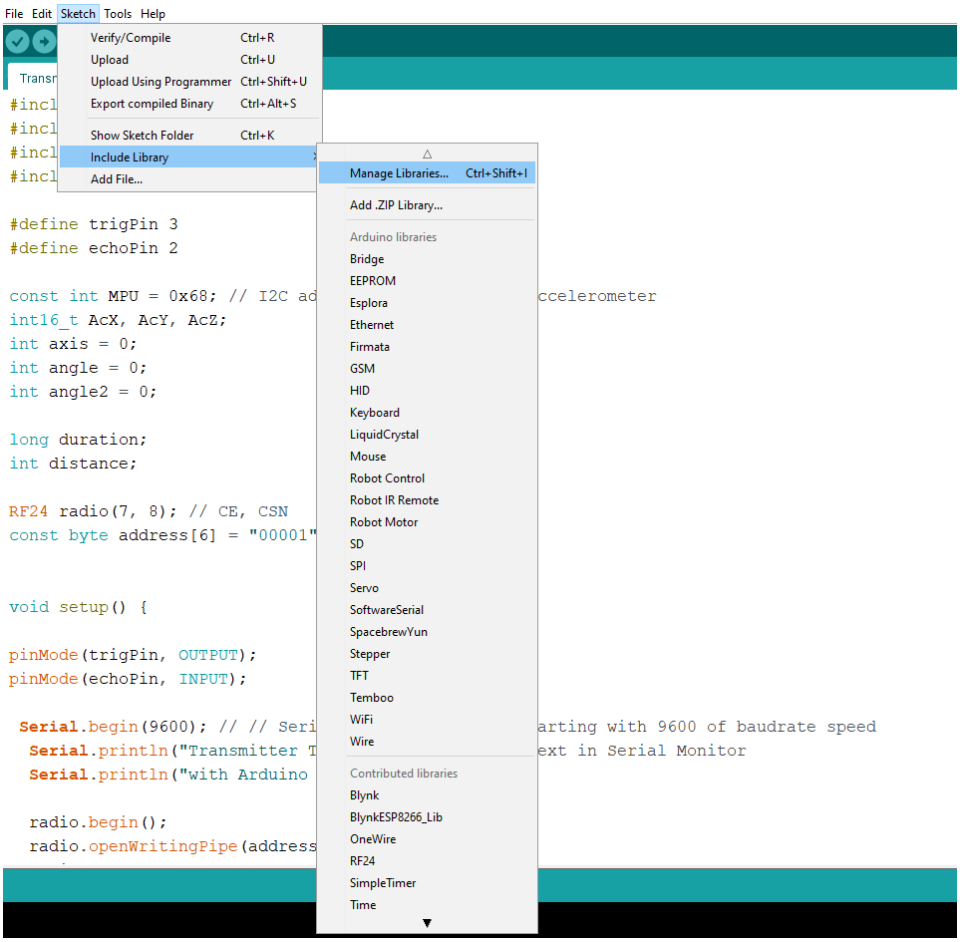
Εικόνα 24 Συνδεσμολογία ESP8266-01 - Wifi Module

4.2 Προγραμματισμός συσκευών

Για τον προγραμματισμό των κόμβων χρησιμοποιήσαμε δυο μικροελεγκτές Arduino Uno. Η πλακέτα Arduino χρησιμοποιεί τη γλώσσα Wiring (ουσιαστικά πρόκειται για τη γλώσσα προγραμματισμού C++ και ένα σύνολο από βιβλιοθήκες, υλοποιημένες επίσης στην C++).

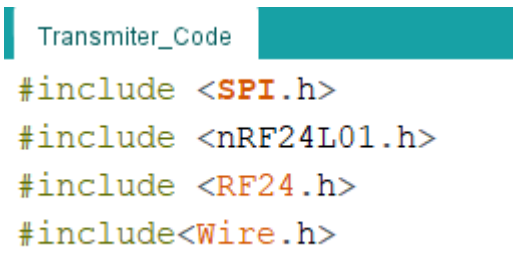
4.2.1 Κόμβος (Λειτουργία - Κώδικας)

Για τον προγραμματισμό του πρώτου κόμβου, χρειάζεται αρχικά να κατεβάσουμε τις κατάλληλες βιβλιοθήκες που θα χρησιμοποιήσουμε παρακάτω στον κώδικά μας. Για να κατεβάσουμε αυτές τις βιβλιοθήκες, θα πρέπει να πάμε στην καρτέλα sketch->include library->Manage libraries όπως φαίνεται στην παρακάτω Εικόνα 25.



Εικόνα 25 Εισαγωγή Βιβλιοθηκών

Στην συνέχεια αρχίζουμε τον προγραμματισμό. Αρχικά, πρώτο βήμα είναι να συμπεριλάβουμε τις βιβλιοθήκες που κατεβάσαμε όπως φαίνεται στην Εικόνα 26.



Εικόνα 26 Εισαγωγή Βιβλιοθηκών στο πρόγραμμα

Με την εντολή #define trigPin και #define echoPin, ορίζουμε τα πηνία του αισθητήρα HC-SR04 στις θέσεις 2 και 3. Αρχικοποιούμε τις μεταβλητές του επιταχυνσιόμετρου και

ορίζουμε τα πηνία της κεραίας στις θέσεις 7 και 8 του Arduino uno. Στη συνέχεια θέτουμε την πύλη επικοινωνίας με την άλλη κεραία χρησιμοποιώντας την εντολή const byte[6]=”000001” και καθορίζουμε τις μεταβλητές μας να δέχονται long(32bits) και int(16bits) στοιχεία, όπως φαίνεται στην παρακάτω Εικόνα 27.

```
#define trigPin 3
#define echoPin 2

const int MPU = 0x68; // I2C address of the MPU6050 accelerometer
int16_t AcX, AcY, AcZ;
int axis = 0;
int angle = 0;
int angle2 = 0;

long duration;
int distance;

RF24 radio(7, 8); // CE, CSN
const byte address[6] = "00001";
```

Εικόνα 27 Αρχικοποίηση τιμών

Αφού έχουμε ολοκληρώσει τις αρχικοποιήσεις, το επόμενο βήμα είναι το void setup() και το void loop().

Το Void setup είναι τεχνικά μια λειτουργία που δημιουργείται στην κορυφή κάθε προγράμματος. Μέσα στις αγκύλες είναι ο κωδικός που θέλουμε να εκτελέσουμε μία φορά, μόλις ξεκινήσει η εκτέλεση του προγράμματος. Μπορούμε να ορίσουμε διάφορες λειτουργίες (π.χ pinMode) σε αυτήν την ενότητα.

Το void loop είναι μια άλλη λειτουργία που χρησιμοποιεί το Arduino ως μέρος της δομής του. Στην ενότητα void setup, ορίζουμε την κατάσταση των πηνίων σε πηνία αποστολής και πηνία λήψεις δεδομένων. Με την εντολή serial.begin ορίζουμε την συχνότητα επικοινωνίας με την ψηφιακή μας θύρα, ενώ με την εντολή serial.println μας επιτρέπεται να γράψουμε οποιαδήποτε πληροφορία θέλουμε να μεταφέρουμε στο serial monitor.

Με την εντολή radio.begin αρχίζουμε την επικοινωνία μεταξύ των δυο κεραιών. Για την σύνδεσή τους, χρησιμοποιούμε την εντολή radio.openWritingPipe(address) η οποία μας επιτρέπει να χρησιμοποιήσουμε την διεύθυνση που έχουμε ορίσει νωρίτερα στον κώδικά μας. Στην συνέχεια, χρησιμοποιούμε την εντολή radio.setPALevel για να ρυθμίσουμε το επίπεδο PA και να αποφύγουμε οποιαδήποτε ζητήματα προκύψουν με την τροφοδοσία.

Με την εντολή radio.stopListening ορίζουμε ότι δεν θέλουμε να δέχεται πληροφορίες από την άλλη κεραία πάρα μόνο να στέλνει.

Τέλος, χρησιμοποιώντας την βιβλιοθήκη Wire αρχικοποιούμε τον αισθητήρα του επιταχυνσιόμετρου όπως φαίνεται και στην Εικόνα 28.

```
void setup() {

pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);

  Serial.begin(9600); // // Serial Communication is starting with 9600 of baudrate speed
  Serial.println("Transmitter Test"); // print some text in Serial Monitor
  Serial.println("with Arduino UNO R3");

  radio.begin();
  radio.openWritingPipe(address);
  radio.setPALevel(RF24_PA_MIN);
  radio.stopListening();

   // Initialize interface to the MPU6050
  Wire.begin();
  Wire.beginTransmission(MPU);
  Wire.write(0x6B);
  Wire.write(0);
  Wire.endTransmission(true);
}
```

Εικόνα 28 Κώδικας Void Setup

Για τον κώδικα του void loop πρέπει να παραθέσουμε αυτό που θέλουμε να κάνει το πρόγραμμα μας συνεχόμενα. Αρχίζουμε με τον κώδικα που χρειάζεται ο αισθητήρας HC-SR04 , με την εντολή digitalWrite(trigPin, Low), ορίζουμε το πηνίο της αποστολής ως ανενεργό και με την εντολή delayMicroseconds ορίζουμε τον χρόνο που θέλουμε να παραμείνει έτσι. Χρησιμοποιούμε την εντολή digitalWrite(trigPin, HIGH), delayMicroseconds(10) και digitalWrite(trigPin, LOW) για να στείλουμε το πρώτο κύμα από τον αισθητήρα που θα διαρκέσει 10 μικρο-δευτερόλεπτα. Στην συνέχεια με την εντολή

duration = pulseIn(echoPin, HIGH) θα διαβάσει τον χρόνο που έκανε για να ταξιδέψει το κύμα και να το θέσει στην μεταβλητή που έχουμε δημιουργήσει με όνομα duration. Τέλος, θα υπολογίσουμε την πληροφορία έτσι ώστε να μας εμφανίζεται σε εκατοστά με τον παρακάτω μαθηματικό τύπο distance = duration * 0.034 / 2.

Για τον κώδικα του επιταχυνσιόμετρου, χρησιμοποιούμε την βιβλιοθήκη wire που μας επιτρέπει τον ευκολότερο προγραμματισμό του αισθητήρα. Αρχικά, με τον κώδικα Wire.beginTransmission(MPU) αρχίζουμε να μεταδίδουμε τα δεδομένα μας. Με την εντολή Wire.requestFrom(MPU, 6, true), AcX = Wire.read() << 8 | Wire.read(), AcY = Wire.read() << 8 | Wire.read(), AcZ = Wire.read() << 8 | Wire.read() διαβάζει 6 καταχωρήσεις συνολικά και κάθε τιμή άξονα αποθηκεύεται σε 2 καταχωρητές.

Τέλος, με τους παρακάτω μαθηματικούς τύπους υπολογίζει την κλήση και την γωνία ενός αντικειμένου στον τρισδιάστατο χώρο angle = atan(-1 * AcX / sqrt(pow(AcY, 2) + pow(AcZ, 2))) * 180 / PI και angle2 = atan(-1 * AcY / sqrt(pow(AcX, 2) + pow(AcZ, 2))) * 180 / PI όπως φαίνεται και στην παρακάτω Εικόνα 29

```
void loop() {

digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin HIGH (ACTIVE) for 10 microseconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculating the distance
  distance = duration * 0.034 / 2;

    // Angle measuring program
      // Read the accelerometer data
      Wire.beginTransmission(MPU);
      Wire.write(0x3B); // Start with register 0x3B (ACCEL_XOUT_H)
      Wire.endTransmission(false);
      Wire.requestFrom(MPU, 6, true); // Read 6 registers total, each axis value is stored in 2 registers
      AcX = Wire.read() << 8 | Wire.read(); // X-axis value
      AcY = Wire.read() << 8 | Wire.read(); // Y-axis value
      AcZ = Wire.read() << 8 | Wire.read(); // Z-axis value


        // Calculating the Pitch angle (rotation around Y-axis)
        angle = atan(-1 * AcX / sqrt(pow(AcY, 2) + pow(AcZ, 2))) * 180 / PI;

        // Calculating the Roll angle (rotation around X-axis)
        angle2 = atan(-1 * AcY / sqrt(pow(AcX, 2) + pow(AcZ, 2))) * 180 / PI;
```

Εικόνα 29 Κώδικας Void loop ½

Με τις εντολές serial.print καταγράφουμε τις τιμές μας στο serial monitor και με την εντολή radio.write(&distance, sizeof(distance)), radio.write(&angle, sizeof(angle)), radio.write(&angle2, sizeof(angle2)) και delay(5000) στέλνει την απόσταση και τις δυο γωνίες που έχουμε υπολογίσει κάθε 5 δευτερόλεπτα στην άλλη κεραία όπως φαίνεται και στην Εικόνα 30.

```
// Shows Values in Serial Monitor
  Serial.print("Pitch ");
  Serial.print("Angle: ");
  Serial.print(abs(angle));
  Serial.println(" deg");


  Serial.print("Roll ");
  Serial.print("Angle: ");
  Serial.print(abs(angle2));
  Serial.println(" deg");


  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");

 // Send Values to the Receiver
 radio.write(&distance, sizeof(distance));
 radio.write(&angle, sizeof(angle));
 radio.write(&angle2, sizeof(angle2));

  delay(5000);
}
```

Εικόνα 30 Κώδικας Void loop 2/2

4.2.2 Κόμβος Sink (Λειτουργία - Κώδικας)

Για τον προγραμματισμό τώρα του κόμβου sink θα προσθέσουμε τις βιβλιοθήκες που έχουμε είδη κατεβάσει από τον προηγούμενο κόμβο αλλά θα προσθέσουμε ακόμα λίγες για την λειτουργία της κεραίας WiFi όπως φαίνεται και στην Εικόνα 31 .

.

Receiver_Code

```
#define BLYNK_PRINT Serial
#include <ESP8266_Lib.h>
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
#include<Wire.h>
#include <BlynkSimpleShieldEsp8266.h>
#include <SoftwareSerial.h>
```

Εικόνα 31 Εισαγωγή Βιβλιοθηκών στο πρόγραμμα του κόμβου sink

Αρχικοποιούμε τα πηνία του αισθητήρα HC-SR04 όπως κάναμε και στον προηγούμενο κώδικα στις θέσεις 6 και 5 και τα πηνία της κεραίας στις θέσεις 4 και 10. Στη συνέχεια θέτουμε την πύλη επικοινωνίας με την άλλη κεραία χρησιμοποιώντας την εντολή const byte[6]=”000001” και με τις εντολές WidgetLCD lcd(V0), WidgetTerminal terminal(V2) ορίζουμε τα εικονικά widget της εφαρμογής στις εικονικές θύρες 0 και 2. Με την εντολή ESP8266 wifi(&EspSerial) θέτουμε την συχνότητα της κεραίας WiFi στα 9600 baud rate και με την εντολή SoftwareSerial EspSerial = SoftwareSerial(2,3) τις θύρες επικοινωνίας με αυτήν στις θέσεις 2 και 3 όπως φαίνεται και στην Εικόνα 32.

```
#define trigPin 6
#define echoPin 5

SoftwareSerial EspSerial =  SoftwareSerial(2,3);
RF24 radio(4, 10); // CE, CSN
const byte address[6] = "00001";
WidgetLCD lcd(V0);
WidgetTerminal terminal(V2);

#define ESP8266_BAUD 9600
ESP8266 wifi(&EspSerial);
```

Εικόνα 32 Αρχικοποίηση τιμών 1/3

Στην συνέχεια, θέτουμε το authorization key που μας έχει στείλει η εφαρμογή Blynk ,το ssid και τον κωδικό του router μας (αντικαθιστούμε τις τρεις τελείες με τα στοιχεία του κάθε ρούτερ) όπως φαίνεται και στην Εικόνα 33.

```
char auth[] = "...";
char ssid[] = "...";
char pass[] = "...";
```

Εικόνα 33 Αρχικοποίηση τιμών 2/3

Τέλος, δηλώνουμε τις υπόλοιπες μεταβλητές που θα χρησιμοποιήσουμε για την λειτουργία του κώδικα μας όπως φαίνεται στην παρακάτω Εικόνα 34.

```
const int MPU = 0x68; // I2C address of the MPU6050 accelerometer
int16_t AcX, AcY, AcZ;
int axis = 0;
int Rangle = 0;
int Rangle2 = 0;
int angle = 0;
int angle2 = 0;

long duration;
int Rdistance;
int distance;
```

Εικόνα 34 Αρχικοποίηση τιμών 3/3

Όπως και προηγουμένως, στην ενότητα void setup, ορίζουμε την κατάσταση των πηνίων σε πηνία αποστολής και πηνία λήψης δεδομένων. Με την εντολή serial.begin ορίζουμε την συχνότητα επικοινωνίας με την ψηφιακή μας θύρα, ενώ με την εντολή serial.println μας επιτρέπεται να γράψουμε οποιαδήποτε πληροφορία θέλουμε να μεταφέρουμε στο serial monitor.

Με την εντολή EspSerial.begin(ESP8266_BAUD), delay(10), Blynk.begin("auth",wifi, "ssid", "pass") ενεργοποιούμε την κεραία WiFi και δίνουμε τις απαραίτητες πληροφορίες για να συνδεθεί η εφαρμογή με το δίκτυο μας και στην συνέχεια με την κεραία WiFi.

Με την εντολή radio.begin αρχίζουμε την επικοινωνία μεταξύ των δυο κεραιών. Για την σύνδεσή τους, χρησιμοποιούμε την εντολή radio.openWritingPipe(address) η οποία μας επιτρέπει να χρησιμοποιήσουμε την διεύθυνση που έχουμε ορίσει νωρίτερα στον κώδικα

μας. Στην συνέχεια, χρησιμοποιούμε την εντολή radio.setPALevel για να ρυθμίσουμε το επίπεδο PA και να αποφύγουμε οποιαδήποτε ζητήματα προκύψουν με την τροφοδοσία.

Με την εντολή radio.startListening ορίζουμε ότι θέλουμε να δέχεται πληροφορίες από την άλλη κεραία και να τις αποθηκεύει τοπικά.

Τέλος, χρησιμοποιώντας την βιβλιοθήκη Wire αρχικοποιούμε τον αισθητήρα του επιταχυνσιόμετρου όπως φαίνεται και στην Εικόνα 35.

```
void setup() {

  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);

  Serial.begin(9600); // // Serial Communication is starting with 9600 of baudrate speed
  Serial.println("Receiver Test"); // print some text in Serial Monitor
  Serial.println("with Arduino UNO R3");
// Connection with the app
  EspSerial.begin(ESP8266_BAUD);
  delay(10);
  Blynk.begin( "auth",wifi, "ssid", "pass");

  Serial.begin(9600);
  radio.begin();
  radio.openReadingPipe(0, address);
  radio.setPALevel(RF24_PA_MIN);
  radio.startListening();

   // Initialize interface to the MPU6050
  Wire.begin();
  Wire.beginTransmission(MPU);
  Wire.write(0x6B);
  Wire.write(0);
  Wire.endTransmission(true);
}
```

Εικόνα 35 Κώδικας Void Setup

Για τον κώδικα του void loop πρέπει να παραθέσουμε αυτό που θέλουμε να κάνει το πρόγραμμα μας συνεχόμενα. Αρχίζουμε, γράφοντας την εντολή Blynk.run που θέτει σε λειτουργία την εφαρμογή μας. Στην συνέχεια, με την εντολή terminal.println αποτυπώνουμε στο τερματικό της εφαρμογής ένα μήνυμα για να δούμε ότι έχει γίνει σύνδεση με του κόμβου με την εφαρμογή. Τέλος, με την εντολή terminal.flush() βεβαιωνόμαστε ότι θα σταλεί ολοκληρωμένο το μήνυμα.

Όπως και στον κώδικα του προηγουμένου κόμβου έτσι πανομοιότυπα πληκτρολογούμε τον κώδικα που χρειάζεται ο αισθητήρας HC-SR04 , με την εντολή digitalWrite(trigPin, Low),

ορίζουμε το πηνίο της αποστολής ως ανενεργό και με την εντολή delayMicroseconds ορίζουμε τον χρόνο που θέλουμε να παραμείνει έτσι. Χρησιμοποιούμε την εντολή digitalWrite(trigPin, HIGH), delayMicroseconds(10) και digitalWrite(trigPin, LOW) για να στείλουμε το πρώτο κύμα από τον αισθητήρα που θα διαρκέσει 10 μικρο-δευτερόλεπτα. Στην συνέχεια με την εντολή duration = pulseIn(echoPin, HIGH) θα διαβάσει τον χρόνο που έκανε για να ταξιδέψει το κύμα και τα το θέσει στην μεταβλητή που έχουμε δημιουργήσει με όνομα duration. Τέλος, θα υπολογίσουμε την πληροφορία έτσι ώστε να μας εμφανίζεται σε εκατοστά με τον παρακάτω μαθηματικό τύπο Rdistance = duration * 0.034 / 2.

Για τον κώδικα του επιταχυνσιόμετρου, χρησιμοποιούμε την βιβλιοθήκη wire που μας επιτρέπει τον ευκολότερο προγραμματισμό του αισθητήρα. Αρχικά, με τον κώδικα Wire.beginTransmission(MPU) αρχίζουμε να μεταδίδουμε τα δεδομένα μας. Με την εντολή Wire.requestFrom(MPU, 6, true), AcX = Wire.read() << 8 | Wire.read(), AcY = Wire.read() << 8 | Wire.read(), AcZ = Wire.read() << 8 | Wire.read() διαβάζει 6 καταχωρήσεις συνολικά και κάθε τιμή άξονα αποθηκεύεται σε 2 καταχωρητές.(Εικόνα 36)

Τέλος, με τους παρακάτω μαθηματικούς τύπους υπολογίζει την κλήση και την γωνία ενός αντικειμένου στον τρισδιάστατο χώρο angle = atan(-1 * AcX / sqrt(pow(AcY, 2) + pow(AcZ, 2))) * 180 / PI και angle2 = atan(-1 * AcY / sqrt(pow(AcX, 2) + pow(AcZ, 2))) * 180 / PI .

Με την εντολή if (radio.available()) ελέγχουμε αν υπάρχουν εισερχόμενες πληροφορίες και αν η απάντηση είναι θετική τότε με τις εντολές radio.read(&distance, sizeof(distance)), radio.read(&angle, sizeof(angle)), radio.read(&angle2, sizeof(angle2)) συλλέγει τις πληροφορίες και τις αποθηκεύει στις εκάστοτε μεταβλητές.(Εικόνα 37)

```
void loop() {

Blynk.run();

  // This will print Blynk Software version to the Terminal Widget when
  // your hardware gets connected to Blynk Server
  terminal.println(F("Blynk v" BLYNK_VERSION ": Device started"));
  terminal.println(F("-------------"));
  terminal.println(F("Establishing connection with ESP8266"));
  terminal.flush();

digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin HIGH (ACTIVE) for 10 microseconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculating the distance
 Rdistance = duration * 0.034 / 2;

    // Angle measuring program
      // Read the accelerometer data
      Wire.beginTransmission(MPU);
      Wire.write(0x3B); // Start with register 0x3B (ACCEL_XOUT_H)
      Wire.endTransmission(false);
      Wire.requestFrom(MPU, 6, true); // Read 6 registers total, each axis value is stored in 2 registers
      AcX = Wire.read() << 8 | Wire.read(); // X-axis value
      AcY = Wire.read() << 8 | Wire.read(); // Y-axis value
      AcZ = Wire.read() << 8 | Wire.read(); // Z-axis value
```

Εικόνα 36 Κώδικας Void loop 1/3

```
            // Calculating the Pitch angle (rotation around Y-axis)
            Rangle = atan(-1 * AcX / sqrt(pow(AcY, 2) + pow(AcZ, 2))) * 180 / PI;

            // Calculating the Roll angle (rotation around X-axis)
            Rangle2 = atan(-1 * AcY / sqrt(pow(AcX, 2) + pow(AcZ, 2))) * 180 / PI;

        if (radio.available()) // If there are incoming data
         {
           radio.read(&distance, sizeof(distance));
           radio.read(&angle, sizeof(angle));
           radio.read(&angle2, sizeof(angle2));

         }
```

Εικόνα 37 Κώδικας Void loop 2/3

Τέλος, με την εντολή Blynk.virtualWrite στέλνουμε τα δεδομένα στις ψηφιακές θύρες που έχουμε ορίσει και από εκεί, η πληροφορίες κατευθύνονται στην εφαρμογή μας κάθε πέντε δευτερόλεπτα, όπως φαίνεται και στην Εικόνα 38.

```
//Sends Data to Blynk App
    Blynk.virtualWrite(V0, Rdistance);
    Blynk.virtualWrite(V1, distance);
    Blynk.virtualWrite(V3, abs(Rangle));
    Blynk.virtualWrite(V4, abs(Rangle2));
    Blynk.virtualWrite(V5, abs(angle));
    Blynk.virtualWrite(V6, abs(angle2));


  delay(5000);

}
```

Εικόνα 38 Κώδικας Void loop 3/3

4.3 Εφαρμογή

Το ΑΔΑ που δημιουργήσαμε μας επιτρέπει να μετράμε σε πραγματικό χρόνο:

- την απόσταση από το κοντινότερο αντικείμενο
- την κλίση και την γωνία του αντικειμένου πάνω στο οποίο το τοποθετούμε

καθιστώντας εφικτή την συλλογή αυτών των μετρήσεων-πληροφοριών και την προώθησή τους σε πραγματικό χρόνο, μέσω του διαδικτύου, σε μια εφαρμογή στο κινητό μας. Ο αριθμός τέτοιων εφαρμογών είναι σημαντικός και συνεχώς εμπλουτίζεται, αλλά το πλεονέκτημα των ΑΔΑ έναντι των συμβατικών καταγραφών είναι η "ζωντανή" ροή δεδομένων που έχουν σαν δυνατότητα να προσφέρουν.

Σε έναν μέσο χρήστη, που ασχολείται με εργασίες do-it-yourself, η κατασκευή ή επιδιόρθωση αντικειμένων της οικίας του, με τη λήψη των σχετικών μετρήσεων στο κινητό του, ακριβώς την χρονική στιγμή που του είναι απαραίτητες, τον εξυπηρετεί τόσο ως προς το ακριβές αποτέλεσμα της εργασίας του όσο και στην εξοικονόμηση του χρόνου που απαιτείται για να την φέρει εις πέρας.

Σε πιο ευρεία κλίμακα, το δίκτυο μπορεί να εξυπηρετήσει ένα βιομηχανικό χώρο, τοποθετώντας, παραδείγματος χάριν, τους κόμβους σε προκαθορισμένες θέσεις, πάνω στον υφιστάμενο εξοπλισμό μιας μονάδας παραγωγής. Ο υπεύθυνος της μονάδας λαμβάνει στο κινητό του όλες τις απαραίτητες πληροφορίες ώστε να διασφαλίσει την ορθή λειτουργία του εξοπλισμού.

ΠΑΡΑΡΤΗΜΑ

Κώδικας Α1

```
[code]

#define BLYNK_PRINT Serial

#include <ESP8266_Lib.h>

#include <SPI.h>

#include <nRF24L01.h>

#include <RF24.h>

#include <Wire.h>

#include <BlynkSimpleShieldEsp8266.h>

#include <SoftwareSerial.h>

#define trigPin 6

#define echoPin 5

SoftwareSerial EspSerial =  SoftwareSerial(2,3);

RF24 radio(4, 10); // CE, CSN

const byte address[6] = "00001";

WidgetLCD lcd(V0);

WidgetTerminal terminal(V2);

#define ESP8266_BAUD 9600

ESP8266 wifi(&EspSerial);
```

```
char auth[] = "…";

char ssid[] = "…";

char pass[] = "…";

const int MPU = 0x68; // I2C address of the MPU6050 accelerometer

int16_t AcX, AcY, AcZ;

int axis = 0;

int Rangle = 0;

int Rangle2 = 0;

int angle = 0;

int angle2 = 0;

long duration;

int Rdistance;

int distance;

void setup() {

  pinMode(trigPin, OUTPUT);

  pinMode(echoPin, INPUT);

  Serial.begin(9600); // // Serial Communication is starting with 9600
of baudrate speed

  Serial.println("Receiver Test"); // print some text in Serial Monitor

  Serial.println("with Arduino UNO R3");

// Connection with the app

  EspSerial.begin(ESP8266_BAUD);

  delay(10);
```

```
  Blynk.begin( "auth",wifi, "ssid", "pass");
  Serial.begin(9600);
  radio.begin();
  radio.openReadingPipe(0, address);
  radio.setPALevel(RF24_PA_MIN);
  radio.startListening();
   // Initialize interface to the MPU6050
  Wire.begin();
  Wire.beginTransmission(MPU);
  Wire.write(0x6B);
  Wire.write(0);
  Wire.endTransmission(true);
}
void loop() {
Blynk.run();
  // This will print Blynk Software version to the Terminal Widget when
  // your hardware gets connected to Blynk Server
  terminal.println(F("Blynk v" BLYNK_VERSION ": Device started"));
  terminal.println(F("-------------"));
  terminal.println(F("Establishing connection with ESP8266"));
  terminal.flush();
digitalWrite(trigPin, LOW);
```

```
  delayMicroseconds(2);
  // Sets the trigPin HIGH (ACTIVE) for 10 microseconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in
microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculating the distance
 Rdistance = duration * 0.034 / 2;
    // Angle measuring program
      // Read the accelerometer data
      Wire.beginTransmission(MPU);
      Wire.write(0x3B); // Start with register 0x3B (ACCEL_XOUT_H)
      Wire.endTransmission(false);
      Wire.requestFrom(MPU, 6, true); // Read 6 registers total, each
axis value is stored in 2 registers
      AcX = Wire.read() << 8 | Wire.read(); // X-axis value
      AcY = Wire.read() << 8 | Wire.read(); // Y-axis value
      AcZ = Wire.read() << 8 | Wire.read(); // Z-axis value
        // Calculating the Pitch angle (rotation around Y-axis)
        Rangle = atan(-1 * AcX / sqrt(pow(AcY, 2) + pow(AcZ, 2))) * 180
/ PI;
        // Calculating the Roll angle (rotation around X-axis)
```

```
        Rangle2 = atan(-1 * AcY / sqrt(pow(AcX, 2) + pow(AcZ, 2))) * 180
/ PI;

 if (radio.available()) // If there are incoming data

  {

    radio.read(&distance, sizeof(distance));

    radio.read(&angle, sizeof(angle));

    radio.read(&angle2, sizeof(angle2));

  }

  Serial.println("-------ARDUINO----------------");

  Serial.print("Pitch ");

  Serial.print("Angle: ");

  Serial.print(abs(Rangle));

  Serial.println(" deg");

  Serial.print("Roll ");

  Serial.print("Angle: ");

  Serial.print(abs(Rangle2));

  Serial.println(" deg");

  Serial.print("Distance: ");

  Serial.print(Rdistance);

  Serial.println(" cm");

  Serial.println("-----------INCOMING---DATA----------------");

  Serial.print("Pitch2 ");

  Serial.print("Angle: ");
```

```
  Serial.print(abs(angle));
  Serial.println(" deg");
  Serial.print("Roll2 ");
  Serial.print("Angle: ");
  Serial.print(abs(angle2));
  Serial.println(" deg");
  Serial.print("Distance2: ");
  Serial.print(distance);
  Serial.println(" cm");
  Serial.println("--------------END----------------");
//Sends Data to Blynk App
    Blynk.virtualWrite(V0, Rdistance);
    Blynk.virtualWrite(V1, distance);
    Blynk.virtualWrite(V3, abs(Rangle));
    Blynk.virtualWrite(V4, abs(Rangle2));
    Blynk.virtualWrite(V5, abs(angle));
    Blynk.virtualWrite(V6, abs(angle2));
  delay(5000);
}
[/code]
```

Κώδικας Α2

```
[code]

#include <SPI.h>

#include <nRF24L01.h>

#include <RF24.h>

#include<Wire.h>

#define trigPin 3

#define echoPin 2

const int MPU = 0x68; // I2C address of the MPU6050 accelerometer

int16_t AcX, AcY, AcZ;

int axis = 0;

int angle = 0;

int angle2 = 0;

long duration;

int distance;

RF24 radio(7, 8); // CE, CSN

const byte address[6] = "00001";

void setup() {

pinMode(trigPin, OUTPUT);

pinMode(echoPin, INPUT);

  Serial.begin(9600); // // Serial Communication is starting with 9600
of baudrate speed
```

```
  Serial.println("Transmitter Test"); // print some text in Serial
Monitor

  Serial.println("with Arduino UNO R3");

  radio.begin();

  radio.openWritingPipe(address);

  radio.setPALevel(RF24_PA_MIN);

  radio.stopListening();

   // Initialize interface to the MPU6050

  Wire.begin();

  Wire.beginTransmission(MPU);

  Wire.write(0x6B);

  Wire.write(0);

  Wire.endTransmission(true);

}

void loop() {

digitalWrite(trigPin, LOW);

  delayMicroseconds(2);

  // Sets the trigPin HIGH (ACTIVE) for 10 microseconds

  digitalWrite(trigPin, HIGH);

  delayMicroseconds(10);

  digitalWrite(trigPin, LOW);

  // Reads the echoPin, returns the sound wave travel time in
microseconds
```

```
  duration = pulseIn(echoPin, HIGH);
  // Calculating the distance
  distance = duration * 0.034 / 2;


    // Angle measuring program
      // Read the accelerometer data
      Wire.beginTransmission(MPU);
      Wire.write(0x3B); // Start with register 0x3B (ACCEL_XOUT_H)
      Wire.endTransmission(false);
      Wire.requestFrom(MPU, 6, true); // Read 6 registers total, each
axis value is stored in 2 registers
      AcX = Wire.read() << 8 | Wire.read(); // X-axis value
      AcY = Wire.read() << 8 | Wire.read(); // Y-axis value
      AcZ = Wire.read() << 8 | Wire.read(); // Z-axis value
        // Calculating the Pitch angle (rotation around Y-axis)
        angle = atan(-1 * AcX / sqrt(pow(AcY, 2) + pow(AcZ, 2))) * 180 /
PI;
        // Calculating the Roll angle (rotation around X-axis)
        angle2 = atan(-1 * AcY / sqrt(pow(AcX, 2) + pow(AcZ, 2))) * 180
/ PI;
// Shows Values in Serial Monitor
  Serial.print("Pitch ");
  Serial.print("Angle: ");
  Serial.print(abs(angle));
```

```
  Serial.println(" deg");

  Serial.print("Roll ");

  Serial.print("Angle: ");

  Serial.print(abs(angle2));

  Serial.println(" deg");

  Serial.print("Distance: ");

  Serial.print(distance);

  Serial.println(" cm");

 // Send Values to the Receiver

 radio.write(&distance, sizeof(distance));

 radio.write(&angle, sizeof(angle));

 radio.write(&angle2, sizeof(angle2));

  delay(5000);

}

[/code]
```

Βιβλιογραφία

A. Giannakos, G. K. I. S., 2009. *A message-optimal sink mobility model for wireless sensor networks,* s.l.: s.n.

Akyildiz, I. S. W. S. y. C. E., 2002. *Wireless sensor networks: a survey,* s.l.: Computer Networks.

Blynk, n.d. *blynk.io.* [Online]
Available at: https://docs.blynk.cc/

Debnath, A., Singaravelu, P. & Verma, S., 2012. Efficient spatial privacy preserving scheme for sensor network. *Central European Journal of Engineering 3*, 19 December, pp. 1-10.

Dejan, n.d. *howtomechatronics.com.* [Online]
Available at: https://howtomechatronics.com/tutorials/arduino/arduino-wireless-communication-nrf24l01-tutorial/

Dejan, n.d. *howtomechatronics.com.* [Online]
Available at: https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/

Dejan, n.d. *https://howtomechatronics.com.* [Online]
Available at: https://howtomechatronics.com/how-it-works/electrical-engineering/mems-accelerometer-gyrocope-magnetometer-arduino/

E. Tuba, D. S. E. D. R. J. a. M. T., Jun. 2018. *Energy efficient sink placement in wireless sensor networks by brain storm optimization algorithm,* s.l.: s.n.

F. Viani, P. R. M. B. G. O. A. M., 2010. Electromagnetic passive localization and tracking of moving targets in a WSN-infrastructured environment. *Inverse Problems*, vol. 26, pp. 1-15.

Feng Xia, Y.-C. T. Y. L. a. Y. S., 2007. *Wireless Sensor/Actuator Network Design for Mobile Control Applications.* [Online]
Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3864515/
[Accessed 7 October 2007].

harleek, 24 May, 2020. *Buffering in Computer Network,* s.l.: GeeksforGeeks.

Hassaballah, E. H. H. M. R. S. K. H. H. S., 14 July 2021. *A Review of Metaheuristic Optimization Algorithms in Wireless Sensor Networks.* s.l.:s.n.

Heinzelman, W. C. A. a. B. H., January 2000. *Energy-Efficient Communication Protocols for Wireless Microsensor Networks,* s.l.: s.n.

https://osi-model.com/, χ.χ. *github.* [Ηλεκτρονικό]
Available at: https://osi-model.com/

instructables, 2nd April 2018. *instructables.* [Online]
Available at: https://www.instructables.com/How-to-use-the-ESP8266-01-pins/

J. Luo, J.-P. H., 2005. *Joint mobility and routing for lifetime elongation in wireless sensor networks,* s.l.: s.n.

K. Sohraby, D. M. T. Z., 2007. *Wireless sensor networks: Technology,Protocols and Applications.* Wiley: s.n.

M. M. Fouad, V. S. a. A. E. H., Jul. 2015.. *Energy-aware sink node localization algorithm for wireless sensor networks,* s.l.: s.n.

Mohamed Younisa, K. A., June 2008. *Ad Hoc Networks.* Volume 6, Issue 4, June 2008, Pages 621-655 ed. s.l.:s.n.

Pereira, P. G. A. R. N. M. C. C. A. P. J. M., 2007. *End-To-End Reliability In Wireless Sensor Networks: Survey and Research Challenges.,* Lisbon: EuroFGI Workshop on IP QoS and Traffic Control.

Surie, D. L. O. P. T., 2008. *Wireless Sensor Networking of Everyday Objects in a Smart Home Environment.* Sydney, Australia: s.n.

Tatiana Bokareva, W. H. K. R. G. T. B. R. a. S. J., 2006. *Wireless Sensor Networks for Battlefield Surveillance.* Brisbane, s.n.

Yick, J. B. M. G. D., 2008. *Wireless Sensor Network Survey.* 52 ed. s.l.:s.n.

Παπαβασιλείου, Χ. Χ., 2011. *Συλλογή και Συνάθροιση Δεδομένων σε Ασύρματα Δίκτυα Αισθητήρων με Χρήση Ενεργειακά Δίκτυα Αισθητήρων Αποδοτικού Πρωτοκόλλου,* Βόλος: s.n.

[Οπισθόφυλλο. Κενή σελίδα]