



Σχολή Θετικών Επιστημών και Τεχνολογίας
Συστήματα Κινητού και Διάχυτου Υπολογισμού (ΣΔΥ)

Διπλωματική Εργασία

Σχεδιασμός και ανάπτυξη εφαρμογής IoT ενσωματωμένου συστήματος με το
εργαλείο ανοικτού κώδικα Elixir Nerves

Νικόλαος Κόκκος

Επιβλέπων καθηγητής: Νικόλαος Σκλάβος

Πάτρα, Ιούνιος 2026



Σχεδιασμός και ανάπτυξη εφαρμογής IoT ενσωματωμένου συστήματος με το εργαλείο ανοικτού κώδικα Elixir Nerves

Νικόλαος Κόκκος

Επιτροπή Επίβλεψης Διπλωματικής Εργασίας

Επιβλέπων Καθηγητής:

Νικόλαος Σκλάβος

Αν. Καθηγητής

Πανεπιστήμιο Πατρών

Συν-Επιβλέπων Καθηγητής:

Περικλής Χατζημίσιος

Καθηγητής

Διεθνές Πανεπιστήμιο της Ελλάδας

Περίληψη

Στο σύγχρονο ερευνητικό πεδίο της Πληροφορικής, ο Διάχυτος Υπολογισμός (Ubiquitous Computing) και το Διαδίκτυο των Πραγμάτων (Internet of Things/IoT) επαναπροσδιορίζουν ριζικά τις δυνατότητες αλληλεπίδρασης του ψηφιακού με τον φυσικό κόσμο. Μέσω της ενσωμάτωσης της υπολογιστικής ισχύος και των δυνατοτήτων δικτύωσης σε συσκευές μικρής κλίμακας, καθίσταται δυνατή η συνεχής, απρόσκοπτη και σε πραγματικό χρόνο παρακολούθηση κρίσιμων περιβαλλοντικών παραμέτρων. Η παρούσα διπλωματική εργασία εντάσσεται σε αυτό το πλαίσιο, εστιάζοντας στη σχεδίαση, ανάπτυξη και εργαστηριακή αξιολόγηση ενός ολοκληρωμένου ενσωματωμένου συστήματος για την παρακολούθηση της ποιότητας του αέρα, με έμφαση στη μέτρηση του μονοξειδίου του άνθρακα (CO). Η υλοποίηση βασίζεται στην γλώσσα προγραμματισμού Elixir και του εργαλείου ανοιχτού λογισμικού Elixir Nerves, το οποίο επιτρέπει την εκτέλεση της εφαρμογής απευθείας πάνω στην εικονική μηχανή της Erlang (BEAM) ως ένα αυτόνομο, ελαφρύ λειτουργικό σύστημα για υπολογιστές μικρής κλίμακας, όπως το Raspberry Pi Zero Wireless.

Στο επίπεδο του υλικού, σχεδιάστηκε και κατασκευάστηκε ένα ειδικά διαμορφωμένο αναλογικό κύκλωμα προσαρμογής σήματος για τη μετατροπή του εξαιρετικά χαμηλού ρεύματος εξόδου του αισθητήρα Figaro TGS5042(αισθητήρας μονοξειδίου του άνθρακα) σε μετρήσιμη τάση. Η αναλογική προς ψηφιακή μετατροπή πραγματοποιήθηκε μέσω του μετατροπέα ADS1115 σε διαφορική διάταξη, ενώ εφαρμόστηκε αναλογο-μετρική μέθοδος (ratiometric approach) και ψηφιακό φίλτρο διαμέσου (median filtering) για τη μαθηματική ακύρωση των διακυμάνσεων της τάσης τροφοδοσίας και την εξάλειψη του ηλεκτρικού και ηλεκτρομαγνητικού θορύβου. Στο επίπεδο του λογισμικού, αναπτύχθηκε μια αρχιτεκτονική βασισμένη στα πρότυπα του Erlang/OTP, αξιοποιώντας δέντρα εποπτείας (supervision trees) και αυτόνομες διεργασίες τύπου GenServer, εξασφαλίζοντας υψηλή ανοχή σε σφάλματα και απομόνωση των δυσλειτουργιών του υλικού. Για τη διαχείριση των περιορισμένων πόρων μνήμης τυχαίας προσπέλασης (RAM) του υπολογιστή Raspberry Pi και την αποφυγή της πρόωρης φθοράς της κάρτας micro-SD, σχεδιάστηκε μια ενδιάμεση μνήμη αποθήκευσης (buffer) στη RAM μέσω πινάκων ETS (Erlang Term Storage). Τέλος, η παρακολούθηση των δεδομένων γίνεται σε πραγματικό χρόνο μέσω μιας διαδικτυακής διεπαφής βασισμένης στις τεχνολογίες Phoenix LiveView και WebSockets, με παράλληλη αποστολή των μετρήσεων σε εξωτερική πλατφόρμα IoT. Τα αποτελέσματα καταδεικνύουν πώς το περιβάλλον εκτέλεσης

Erlang/OTP και το πλαίσιο λογισμικού Elixir Nerves επιτρέπουν τη δημιουργία σταθερών σταθμών μέτρησης βιομηχανικού επιπέδου (industrial-grade).

Λέξεις – Κλειδιά: Διαδίκτυο των Πραγμάτων (IoT), Ενσωματωμένα Συστήματα, Elixir Nerves, Αρχιτεκτονική Erlang/OTP, Ανοχή σε Σφάλματα, Αναλογικό Κύκλωμα Προσαρμογής, Αναλογομετρική Μέθοδος, Ψηφιακό Φίλτρο Διαμέσου, Ενδιάμεση Μνήμη (Buffer), Υποδειγματοληψία Δεδομένων, Διαδικτυακή Διεπαφή (Web Interface), Websockets, IoT Platform.

Development of an embedded system using Elixir Nerves

Nikolaos Kokkos

Abstract

In the modern research field of computer Science, Ubiquitous Computing and the Internet of Things (IoT) are radically redefining the interaction capabilities between the digital and the physical world. Through the integration of computational power and networking capabilities into small-scale devices, continuous, seamless, and real-time monitoring of critical environmental parameters becomes feasible. This master's thesis falls within this scope, focusing on the design, development, and laboratory evaluation of a comprehensive embedded system for air quality monitoring, with a primary emphasis on carbon monoxide (CO) measurement. The implementation is based on the Elixir programming language and the open-source Elixir Nerves framework, which enables the application to execute directly on top of the Erlang Virtual Machine (BEAM) as a standalone, lightweight operating system tailored for small-scale computers such as the Raspberry Pi Zero Wireless.

At the hardware level, a custom analog signal conditioning circuit was designed and constructed to convert the extremely low output current of the Figaro TGS5042 carbon monoxide sensor into a measurable voltage. Analog-to-digital conversion was performed using the ADS1115 converter in a differential configuration, while a ratiometric approach and a digital median filtering algorithm were applied to mathematically cancel power supply voltage fluctuations and eliminate electrical and electromagnetic noise. At the software level, an architecture based on Erlang/OTP standards was developed, leveraging supervision trees and autonomous GenServer processes to ensure high fault tolerance and hardware fault isolation. To manage the limited Random Access Memory (RAM) resources of the Raspberry Pi computer and prevent premature wear of the micro-SD card, an in-memory storage buffer was designed within the RAM using ETS (Erlang Term Storage) tables. Finally, data monitoring is conducted in real time through a web interface powered by Phoenix LiveView and WebSockets technologies, alongside the concurrent transmission of measurements to an

external Internet of Things platform. The results demonstrate how the Erlang/OTP runtime environment and the Elixir Nerves software framework enable the creation of stable, industrial-grade measurement stations.

Keywords: Internet of Things (IoT), Embedded Systems, Elixir Nerves Framework, Erlang/OTP Architecture, Fault Tolerance, Analog Conditioning Circuit, Ratiometric Approach, Median Filtering, Erlang Term Storage (ETS), Min-Max Downsampling, Web Interface (Phoenix LiveView).

Ευχαριστίες

Ευχαριστώ θερμά τον επιβλέποντα καθηγητή μου Νικόλαο Σκλάβο για την εμπιστοσύνη, την ελευθερία κινήσεων που μου έδωσε κατά τον σχεδιασμό και την υλοποίηση, καθώς και για την υποστήριξη, συμβουλές και την αξιολόγηση αυτής της διπλωματικής εργασίας.

Επίσης, θα ήθελα να ευχαριστήσω θερμά τον Ιωάννη Ζαφειρόπουλο, υπεύθυνο του εργαστηρίου Χημείας στο ΕΑΠ, για την διευκόλυνση και βοήθεια και υποστήριξη στο πείραμα με το μονοξείδιο του άνθρακα που εκτελέσαμε μαζί. Η συμβολή του ήταν πολύτιμη για τη διεξαγωγή του πειράματος, το οποίο επιβεβαίωσε στην πράξη τα αποτελέσματα της θεωρητικής μελέτης και της προσομοίωσης του συστήματος.

Η παρούσα εργασία αποτελεί πνευματική ιδιοκτησία του φοιτητή («συγγραφέας/δημιουργός») που την εκπόνησε. Στο πλαίσιο της πολιτικής ανοικτής πρόσβασης ο συγγραφέας/δημιουργός εκχωρεί στο ΕΑΠ, μη αποκλειστική άδεια χρήσης του δικαιώματος αναπαραγωγής, προσαρμογής, δημόσιου δανεισμού, παρουσίασης στο κοινό και ψηφιακής διάχυσής τους διεθνώς, σε ηλεκτρονική μορφή και σε οποιοδήποτε μέσο, για διδακτικούς και ερευνητικούς σκοπούς, άνευ ανταλλάγματος και για όλο το χρόνο διάρκειας των δικαιωμάτων πνευματικής ιδιοκτησίας. Η ανοικτή πρόσβαση στο πλήρες κείμενο για μελέτη και ανάγνωση δεν σημαίνει καθ' οιονδήποτε τρόπο παραχώρηση δικαιωμάτων διανοητικής ιδιοκτησίας του συγγραφέα/δημιουργού ούτε επιτρέπει την αναπαραγωγή, αναδημοσίευση, αντιγραφή, αποθήκευση, πώληση, εμπορική χρήση, μετάδοση, διανομή, έκδοση, εκτέλεση, «μεταφόρτωση» (downloading), «ανάρτηση» (uploading), μετάφραση, τροποποίηση με οποιονδήποτε τρόπο, τμηματικά ή περιληπτικά της εργασίας, χωρίς τη ρητή. Προηγούμενη έγγραφη συναίνεση του συγγραφέα/δημιουργού. Ο συγγραφέας/δημιουργός διατηρεί το σύνολο των ηθικών και περιουσιακών του δικαιωμάτων.

Πίνακας περιεχομένων

1. Εισαγωγή	1
1.2 Στόχος της διπλωματικής	2
1.3 Δομή της διπλωματικής	3
1.4 Μεθοδολογία Υλοποίησης	4
1.5 Συσχέτιση με τους Στόχους Βιώσιμης Ανάπτυξης του ΟΗΕ (Sustainable Development Goals)	5
1.6 Στόχος 3: Καλή Υγεία και Ευημερία	5
1.7 Στόχος 11: Βιώσιμες Πόλεις και Κοινότητες	5
2. Θεωρητικό Υπόβαθρο	7
2.1 Διάχυτος Υπολογισμός (Ubiquitous Computing) και το Παράδειγμα του IoT	7
2.2 Σχετική Βιβλιογραφία Συστημάτων IoT Χαμηλού Κόστους (Low-Cost IoT)	8
2.2.1 Μελέτη Περίπτωσης: Επικύρωση λειτουργίας αισθητήρων CO με χρήση προσιτών μέσων	10
2.3 Λειτουργικά Συστήματα IoT	10
2.4 Μονολιθικός Πυρήνας (Monolithic Kernel)	13
2.5 Microkernel	13
2.6 Υβριδικός Πυρήνας (Hybrid Kernel)	13
2.7 Επιλογή αναπτυξιακής πλατφόρμας Raspberry Pi Zero Wireless	14
2.8 Το σύστημα λογισμικού: Elixir, Erlang BEAM και framework Elixir Nerves	14
2.9 Erlang/OTP	15
2.10 Η γλώσσα προγραμματισμού Elixir για συστήματα IoT	17
2.10.1 Υλοποίηση στην Εικονική Μηχανή BEAM και το εργαλείο Elixir Nerves	18
2.10.2 Εργαλείο Προγραμματισμού IoT Elixir Nerves	19
2.10.3 Αρχιτεκτονική του Nerves βασισμένη στο Linux Πυρήνα	19
2.10.4 Σύστημα Αρχείων Elixir Nerves	21
2.11 Αισθητήρες και εξαρτήματα	23
2.11.1 TGS5042 – Αισθητήρας μονοξειδίου του άνθρακα.	23
2.11.1.1 Μονοξείδιο του άνθρακα – επιπλοκές	23
2.11.2 Αισθητήρας Θερμοκρασίας BME680	26
2.11.3 Μετατροπέας σήματος Αναλογικό προς Ψηφιακό – ADS1115 ADC	27

2.11.4 Τελεστικός Ενισχυτής Microchip MCP6042	28
2.11.5 Κλειδί κρυπτογράφησης Nerves Key.....	29
3. Σχεδιασμός & αρχιτεκτονική συστήματος	31
3.1 Σχεδιασμός αναλογικού κυκλώματος	32
3.1.1 Απαιτήσεις συστήματος	32
3.1.2 Ανάλυση κυκλώματος.....	32
3.1.2.1 Ανάλυση σταδίου Α	33
3.1.2.2 Ανάλυση σταδίου Β	35
3.1.3 Διαιρέτης τάσης στο δεύτερο στάδιο - Προσομοίωση	37
3.1.4 Υπολογισμός CO σε rpm.....	41
3.1.4.1 Αντιστάθμιση Θερμοκρασίας (Temperature Compensation).....	42
3.1.5 Διάγραμμα διασύνδεσης υλικού.....	45
3.1.6 Λίστα υλικών.....	46
3.2 Σχεδιασμός Λογισμικού και διασυνδέσεις	48
3.2.1 Απαιτήσεις συστήματος.....	48
3.2.2 Εφαρμογή Δειγματοληψίας.....	49
3.2.3 Εφαρμογή εμφάνισης πληροφοριών σε πραγματικό χρόνο.	50
3.2.4 Εφαρμογή αποστολής δεδομένων στο ThingsBoard.....	52
3.3 Δομή ολικής εφαρμογής τύπου Elixir Nerves	53
3.3.1 Αρχιτεκτονική Τριών Εφαρμογών	53
4.0 Υλοποίηση συστήματος	55
4.1 Υλοποίηση αναλογικού κυκλώματος.....	55
4.1.1 Ανάλυση διασύνδεσης του ADS1115	60
4.1.2 Διόρθωση πτώσης τάσης στην γραμμή τροφοδοσίας 3.3 V.....	61
4.1.3 Υποσύστημα αρχικής παραμετροποίησης δικτύου	64
4.2 Υλοποίηση λογισμικού	66
4.2.1 Περιβάλλον ανάπτυξης Linux	66
4.2.2 Εργαλείο mix.....	66
4.2.3 Δημιουργία κεντρικού φακέλου εφαρμογής.....	67
4.2.3.1 Δημιουργία των εφαρμογών gas_sensor και gas_sensor_web	67
4.2.3.2 Δημιουργία της εφαρμογής gas_sensor	67

4.2.3.3 Δημιουργία της εφαρμογής gas_sensor_web	68
4.2.4 Επεξήγηση της εφαρμογής firmware	69
4.2.4.1 Αρχείο mix.exs.....	69
4.2.4.2 Αρχείο application.exs	71
4.2.4.3 Οδηγός δικτύωσης τύπου Access Point.....	72
4.2.5 Επεξήγηση της εφαρμογής gas_sensor.....	73
4.2.5.1 Ανάλυση του GasSensor.Application.....	73
4.2.5.2 GasSensor.Reading Agent (Στιγμιότυπο μετρήσεων).....	74
4.2.5.3 bme680_sensor.....	75
4.2.5.5 Λογική Ανάγνωσης Αισθητήρα ASD1115 — GasSensor.Sensor	76
4.2.5.5.1 Μηχανισμός Δειγματοληψίας και Ανάγνωσης Καταχωρητών (Registers)	77
4.2.5.5.2 Ανάλυση(resolution) συστήματος	78
4.2.5.5.3 Αναλογο-μετρική (Ratiometric) Προσέγγιση.....	79
4.2.5.5.4 Θερμοκρασιακή Αντιστάθμιση και Υπολογισμός Συγκέντρωσης CO (PPM)	80
4.2.5.5.5 Διαχείριση Σφαλμάτων και Ανοχή σε Σφάλματα.....	81
4.2.6 Εφαρμογή gas_sensor_web — Web Interface με Phoenix LiveView.....	83
4.2.6.1 Επισκόπηση και Αρχιτεκτονική	83
5.0 Πειραματική Διαδικασία και Αποτελέσματα	87
5.1 Πείραμα 1: Μέτρηση CO από καύση κεριού σε κλειστό χώρο	87
5.2 Πείραμα 2: Δοκιμή εγκυρότητας σε εργαστήριο χημείας και αποστολή δεδομένων.....	91
5.3 Πείραμα 3: Μέτρηση Ρύπων σε Εξάτμιση Οχήματος σε πραγματικές Συνθήκες.....	100
5.4 Περιορισμοί Πειραματικής Διαδικασίας και Ακρίβειας	104
5.5 Σύνοψη Πειραματικής Διαδικασίας	105
6. Επίλογος και συμπεράσματα	106
6.1 Συμπεράσματα	106
6.2 Μελλοντικές Επεκτάσεις	106
Βιβλιογραφία.....	109

Πίνακας Εικόνων/Σχημάτων

Εικόνα 1. Οικιακή συσκευή CO από την FirstAlert.....	1
Εικόνα 2 Το μοντέλο επιπέδων των ενσωματωμένων συσκευών (δεξιά) και των κόμβων άκρου (αριστερά) εντός των σταδίων της αρχιτεκτονικής του IoT.	11
Εικόνα 3. Raspberry Pi Zero Wireless (W)	14
Εικόνα 4. Δημιουργία artifact/διανομής για το Elixir Nerves.....	20
Εικόνα 5. Δημιουργία Firmware Elixir Nerves	21
Εικόνα 6. Δομή διαμερισμάτων στο firmware.....	22
Εικόνα 7 Αισθητήρας TGS-5042.....	25
Εικόνα 8. Χαρακτηριστικά Αισθητήρα TGS5042.....	25
Εικόνα 9. Αρχή λειτουργίας TGS5042.....	26
Εικόνα 10. Αισθητήρας Θερμοκρασίας BME680	27
Εικόνα 11. ADS1115 ADC μετατροπέας	28
Εικόνα 12. Τελεστικός Ενισχυτής MCP6042	29
Εικόνα 13. Απλή διάταξη ενίσχυσης σήματος, application note από την Figaro	29
Εικόνα 14. Chip Κρυπτογραφικής Ταυτοποίησης - Nerves Key.....	30
Εικόνα 15. Διάταξη ολικού συστήματος	31
Εικόνα 16. Αναλογικό κύκλωμα για TGS5042	33
Εικόνα 17. Στάδια αναλογικού κυκλώματος	33
Εικόνα 18. Πρώτο στάδιο ενίσχυσης TGS5042	34
Εικόνα 19. Δεύτερο Στάδιο Ενίσχυσης Σήματος TGS5042	36
Εικόνα 20. Υπολογισμός τάσεων μέσω του DC ανάλυσης	37
Εικόνα 21. Επιλογή ρεύματος για τον TGS5042.....	38
Εικόνα 22. Προσομοίωση τάσεως εξόδου TGS5042 πρώτου σταδίου.....	39
Εικόνα 23. Επιλογή DC Transfer Characteristic.....	40
Εικόνα 24. Ορισμός μέγιστου ρεύματος.....	40
Εικόνα 25. Απόκριση κυκλώματος σε Volts	41
Εικόνα 26. Ευαισθησία TGS5042 αισθητήρα	43
Εικόνα 27. Πίνακας συντελεστής αντιστάθμισης θερμοκρασίας	44
Εικόνα 28 Pinout του Raspberry Pi 0 w	45
Εικόνα 29. Block διάγραμμα ολικής διασύνδεσης με breadboard.....	46
Εικόνα 30. Διάγραμμα ροής δειγματοληψίας	50
Εικόνα 31. Διάταξη διαδικτυακής επαφής για πληροφόρηση CO/Θερμοκρασίας	51
Εικόνα 32. Λειτουργία εφαρμογής web.....	51
Εικόνα 33. Αποστολή δεδομένων στην IoT πλατφόρμα ThingsBoard.....	53
Εικόνα 34. Δομή της εφαρμογής μας.....	55
Εικόνα 35. Τελικό αναλογικό κύκλωμα με τον TGS5042.....	58
Εικόνα 36. Αναλογικό κύκλωμα TGS5042	58

Εικόνα 37. Σύνδεση Τριάδας εξόδου στον ads1115	59
Εικόνα 38. Εναλλαγές τάσης τροφοδοσίας ανά 0.5 sec περίπου	62
Εικόνα 39. Τελικό αναλογικό κύκλωμα στο breadboard	63
Εικόνα 40. Raspberry PI Zero W με breadboard	63
Εικόνα 41. Ενεργοποίηση Pin11 για wifi	64
Εικόνα 42. Διάταξη μπουτόν στο breadboard	65
Εικόνα 43. Access point μέσω του Elixir Nerves	72
Εικόνα 44. Κύρια Σελίδα Παρακολούθησης	84
Εικόνα 45. Σελίδα Τάσεων	85
Εικόνα 46. Σελίδα Ιστορικού	86
Εικόνα 47. Σελίδα βαθμονόμησης	87
Εικόνα 48. Έλεγχος CO σε κλειστή γυάλα	89
Εικόνα 49. Έλεγχος CO σε κλειστή γυάλα	90
Εικόνα 50. Γράφημα Vout vs CO ppm	91
Εικόνα 51. Απαγωγός εστίας	92
Εικόνα 52. Φιάλη CO με την πειραματική διάταξη	92
Εικόνα 53. Πιστοποιητικό για την φιάλη CO	93
Εικόνα 54. Διάταξη συστήματος στον απαγωγό	94
Εικόνα 55. Laptop για παροχή τροφοδοσίας και εμφάνιση μετρήσεων	94
Εικόνα 56. Διοχέτευση CO στον TGS5042	95
Εικόνα 57. Διεπαφή web με μέτρηση CO	96
Εικόνα 58. Μέτρηση ποσότητας πάνω από 1500ppm	96
Εικόνα 59. Οριακή τιμή Vsensor	97
Εικόνα 60. CO vs Θερμοκρασία	98
Εικόνα 61. CO vs Volts	98
Εικόνα 62. Γραφική Παράσταση ThingsBoard με δεδομένα CO	99
Εικόνα 63. Διάταξη εξωτερικού χώρου για μέτρηση CO	100
Εικόνα 64. Απόσταση από την εξάτμιση στα 50 εκατοστά	101
Εικόνα 65. Καταγραφή CO στην εξάτμιση.	102
Εικόνα 66. ThingsBoard Κονσόλα για CO – αυτοκίνητο	103
Εικόνα 67. Δεδομένα στην κονσόλα ThingsBoard	104

Πίνακας Πινάκων

Πίνακας 1. Συγκριτική Ανάλυση Λειτουργικών Συστημάτων IoT	11
Πίνακας 2 Συγκεντρώσεις CO, Επίπεδα COHb και Συμπτώματα	24
Πίνακας 3. Εξαρτήματα Mouser για την κατασκευή του κυκλώματος	46
Πίνακας 4 Λίστα εξαρτημάτων από διαφορετικούς πωλητές	47

1. Εισαγωγή

Η διασφάλιση της ποιότητας του αέρα στα σύγχρονα αστικά και οικιακά περιβάλλοντα αποτελεί σημαντικό ζήτημα για τη δημόσια υγεία. Η έγκαιρη ανίχνευση τοξικών αερίων, όπως το μονοξείδιο του άνθρακα (CO), είναι κρίσιμη για την αποτροπή επικίνδυνων καταστάσεων σε εσωτερικούς χώρους. Στην τρέχουσα καταναλωτική αγορά, η πλειονότητα των διαθέσιμων ανιχνευτών CO είναι έτοιμες, εμπορικές συσκευές(εικόνα 1) κλειστού τύπου [\[1\]](#), [\[2\]](#), [\[3\]](#) οι οποίες φέρουν βιομηχανικές πιστοποιήσεις ασφαλείας (όπως το πρότυπο Underwriters Laboratories - UL) [\[4\]](#). Οι συσκευές αυτές, ωστόσο, παρουσιάζουν σημαντικούς περιορισμούς ως προς τη διαχείριση της πληροφορίας: είτε λειτουργούν εντελώς αποκομμένες από το διαδίκτυο περιορίζοντας την ειδοποίηση μόνο στον φυσικό χώρο, είτε μεταφέρουν υποχρεωτικά τα δεδομένα των χρηστών σε ιδιωτικές, κλειστές πλατφόρμες τρίτων, επιβάλλοντας συνδρομητικά μοντέλα και εγείροντας ζητήματα προστασίας των προσωπικών δεδομένων. π.χ Siemens Cerberus Pro [\[5\]](#). Κάποιες άλλες μπορεί να καλύπτουν και τις δυο εκδοχές αλλά είναι εξαιρετικά δαπανηρές.



Εικόνα 1. Οικιακή συσκευή CO από την FirstAlert

Από την άλλη πλευρά, η ανάπτυξη μιας αυτόνομης, ανοικτής εναλλακτικής λύσης που προασπίζεται την ψηφιακή κυριότητα των ανθρώπων που χρησιμοποιούν αυτές τις συσκευές, έρχεται αντιμέτωπη με σημαντικές μηχανικές προκλήσεις. Ενώ είναι δυνατή η προμήθεια πιστοποιημένων ηλεκτροχημικών αισθητήρων CO (όπως ο TGS5042 της Figaro), η ενσωμάτωσή τους σε ένα σύστημα ειδικής σχεδίασης(custom design) δεν αποτελεί μια απλή διαδικασία άμεσης και απλής σύνδεσης τύπου plug-and-play. Για τη λειτουργία του αισθητήρα απαιτείται η σχεδίαση και κατασκευή ενός εξειδικευμένου αναλογικού κυκλώματος που επεξεργάζεται σωστά τα ηλεκτρικά σήματα που μας ενδιαφέρουν. Η ανάγκη αυτή μετατρέπει

τη διαδικασία ανάπτυξης σε ένα εξαιρετικό πεδίο εκπαιδευτικής διαδικασίας, όπου ο δημιουργός καλείται να «παντρέψει» τη σχεδίαση αναλογικού υλικού, τη διαχείριση της ψηφιοποίησης (ADC), τη συγγραφή firmware και την ανάπτυξη ανοικτού λογισμικού, με τελικό κέρδος την κατάκτηση της γνώσης γύρω από τη λειτουργία των ενσωματωμένων συστημάτων.

1.2 Στόχος της διπλωματικής

Κεντρικός στόχος της παρούσας διπλωματικής εργασίας είναι η σχεδίαση και η υλοποίηση ενός ελάχιστα βιώσιμου προϊόντος (Minimum Viable Product - MVP) για την παρακολούθηση της ποιότητας του αέρα, και κυρίως του μονοξειδίου του άνθρακα κάνοντας χρήση του εργαλείου Elixir Nerves και μιας ευρέως διαθέσιμης commercial off-the-shelf (COTS) πλατφόρμας, όπως το Raspberry Pi Zero Wireless.

Για την επίτευξη του κεντρικού στόχου, καθορίζονται οι ακόλουθοι επιμέρους αντικειμενικοί σκοποί:

- **Σχεδίαση και Κατασκευή Αναλογικού Κυκλώματος:** Η μελέτη και η πρακτική υλοποίηση του κυκλώματος ενίσχυσης ρεύματος (Transimpedance Amplifier) για τον αισθητήρα TGS5042 της Figaro και η διασύνδεσή του με τον ADC ADS1115 που μετατρέπει αναλογικά σήματα σε ψηφιακά ώστε να γίνει η επεξεργασία τους από Raspberry Pi, καθώς το Raspberry Pi Zero Wireless δεν διαθέτει ενσωματωμένη δυνατότητα επεξεργασίας αναλογικών σημάτων.
- **Ανάπτυξη Ενοποιημένου Firmware με Elixir Nerves:** Η παραμετροποίηση του Elixir Nerves για το Raspberry Pi Zero W και η συγγραφή κώδικα στην γλώσσα προγραμματισμού Elixir για τη δειγματοληψία, συλλογή δεδομένων και από άλλες συσκευές που συνδέονται στο Raspberry Pi Zero Wireless όπως τον αισθητήρα θερμοκρασίας/υγρασίας BME680, τη διαχείριση των διαύλων I2C και την αποστολή μηνυμάτων σε IoT πλατφόρμα αναλογικών μετρήσεων.
- **Αυτόνομη Οικιακή Λειτουργία:** Ανάπτυξη τοπικής διεπαφής (Web UI) για την απεικόνιση των μετρήσεων σε πραγματικό χρόνο μέσω websockets, διασφαλίζοντας τη λειτουργία της συσκευής αυτόνομα εντός του οικιακού δικτύου, χωρίς εξάρτηση

από τρίτες πλατφόρμες. Η εφαρμογή αυτή θα λειτουργεί σαν κομμάτι του ενσωματωμένου συστήματος.

- **Μεταφορά γνώσης και ανοικτή συνεισφορά:** Ο τελικός ενδιαφερόμενος/η μπορεί να κατεβάσει τον ανοικτό κώδικα (open-source), να ακολουθήσει τις καταγεγραμμένες οδηγίες, να κατανοήσει το «πάντρεμα» υλικού και λογισμικού και να ολοκληρώσει την κατασκευή αυτόνομα, αποκτώντας μια συσκευή που του ανήκει ολοκληρωτικά.

1.3 Δομή της διπλωματικής

Το σώμα της παρούσας διπλωματικής εργασίας οργανώνεται σε επτά διακριτά κεφάλαια:

Στο **Κεφάλαιο 2 (Θεωρητικό Υπόβαθρο)** παρουσιάζεται το θεωρητικό υπόβαθρο των IoT συστημάτων και αναλύονται οι τεχνολογίες Elixir, BEAM και Nerves. Εξετάζεται η σχετική βιβλιογραφία low-cost συστημάτων για μέτρηση παραμέτρων ποιότητας του αέρα και οι προδιαγραφές των αισθητήρων.

Στο **Κεφάλαιο 3 (Σχεδιασμός & αρχιτεκτονική συστήματος)** πραγματοποιείται η ανάλυση απαιτήσεων του MVP. Αναλύεται η hardware αρχιτεκτονική γύρω από το Raspberry Pi Zero W, ο αναλυτικός σχεδιασμός του κυκλώματος του ενισχυτή ρεύματος και του ADC, καθώς και η δομή ροής των δεδομένων.

Στο **Κεφάλαιο 4 (Υλοποίηση Συστήματος)** περιγράφεται η πρακτική υλοποίηση. Παρουσιάζονται τα interfaces επικοινωνίας (I2C/ADC), ο κώδικας δειγματοληψίας σε Elixir, καθώς και η ανάπτυξη του τοπικού Web UI καθώς η επιπλέον διασύνδεση με το ThingsBoard, μια γνωστή πλατφόρμα IoT.

Στο **Κεφάλαιο 5 (Πειραματική Διαδικασία και Αποτελέσματα)** αναλύονται οι διαδικασίες ελέγχου, η μεθοδολογία εργαστηριακής βαθμονόμησης των αισθητήρων και οι δοκιμές σταθερότητας στο πεδίο.

Στο **Κεφάλαιο 6 (Επίλογος και συμπεράσματα)** παρουσιάζονται τα πειραματικά αποτελέσματα και προτείνονται μελλοντικές επεκτάσεις.

1.4 Μεθοδολογία Υλοποίησης

Για την υλοποίηση της παρούσας διπλωματικής εργασίας και τη δόμηση ενός αξιόπιστου, ανοικτού και αναπαραγωγίμου συστήματος MVP, ακολούθησα μια συστηματική μεθοδολογική προσέγγιση που γεφυρώνει τη θεωρητική έρευνα με την πρακτική μηχανική ενσωματωμένων συστημάτων. Η διαδικασία αναλύεται σε τέσσερις διακριτές φάσεις:

1. **Βιβλιογραφική Αναζήτηση και Επισκόπηση:** Πραγματοποιήθηκε αναζήτηση στη διεθνή βιβλιογραφία με τη χρήση του Google Scholar και εξειδικευμένων επιστημονικών μηχανών αναζήτησης όπως ieeexplore. Εξετάστηκαν πηγές τόσο από τομείς της πληροφορικής, όσο και από το πεδίο της περιβαλλοντικής μηχανικής, με έμφαση στη χημική δυναμική και τους τρόπους μέτρησης του μονοξειδίου του άνθρακα (CO).
2. **Σχεδιασμός και Ανάπτυξη Υλικού:** Με βάση τις προδιαγραφές που αντλήθηκαν από την τεκμηρίωση του κατασκευαστή, σχεδιάστηκε και κατασκευάστηκε από το μηδέν το απαραίτητο αναλογικό κύκλωμα προσαρμογής σήματος. Η φάση αυτή αποτέλεσε βασικό κομμάτι της εκπαιδευτικής διαδικασίας, καθώς απαίτησε τη βαθιά κατανόηση και υλοποίηση ενός ενισχυτή ρεύματος σε τάση (Transimpedance Amplifier) για τη διαχείριση του ασθενούς σήματος του ηλεκτροχημικού αισθητήρα, και τη διασύνδεσή του με τον εξωτερικό ADC (ADS1115) και την πλατφόρμα Raspberry Pi Zero W. Μελετήθηκαν αρκετά datasheets σχετικά τον τελεστικό ενισχυτή καθώς για τον αισθητήρα TGS5042.
3. **Ανάπτυξη Λογισμικού και Firmware:** Λόγω της περιορισμένης ύπαρξης επίσημης ακαδημαϊκής βιβλιογραφίας για embedded εφαρμογές με Elixir Nerves, η ανάπτυξη του firmware βασίστηκε στην εκτενή μελέτη, ανάλυση και σύνθεση διάσπαρτων παραδειγμάτων κώδικα και επίσημης τεκμηρίωσης από την open-source κοινότητα του GitHub και του framework Elixir Nerves. Τα αποσπασματικά αυτά παραδείγματα αξιολογήθηκαν, αναβαθμίστηκαν και ενοποιήθηκαν σε μια ολοκληρωμένη και τελική λύση.
4. **Εργαστηριακή επαλήθευση και διάχυση Γνώσης:** Το σύστημα υποβλήθηκε σε εργαστηριακές δοκιμές για την αντιμετώπιση προβλημάτων και για βαθμονόμηση. Το σύνολο της εργασίας (πηγαίος κώδικας, σχηματικά, οδηγίες) οργανώθηκε σε ένα

δημόσιο open-source αποθετήριο. Σκοπός είναι η μεταφορά γνώσης προς τον τελικό χρήστη, ώστε να μπορεί να αναπαράγει τη συσκευή αυτόνομα, επιτυγχάνοντας πλήρη ψηφιακή κυριότητα (digital ownership) χωρίς cloud εξαρτήσεις.

1.5 Συσχέτιση με τους Στόχους Βιώσιμης Ανάπτυξης του ΟΗΕ (Sustainable Development Goals)

Η παρούσα μεταπτυχιακή διπλωματική εργασία ευθυγραμμίζεται με 2 από τις παγκόσμιες κατευθυντήριες γραμμές του Οργανισμού Ηνωμένων Εθνών (ΟΗΕ) για τη βιώσιμη ανάπτυξη, συμβάλλοντας ενεργά και ουσιαστικά στην επίτευξη τριών κρίσιμων Στόχων Βιώσιμης Ανάπτυξης (SDGs) [\[6\]](#). Αυτοί οι στόχοι είναι:

1.6 Στόχος 3: Καλή Υγεία και Ευημερία

Ο **στόχος 3.9** του ΟΗΕ θέτει ως προτεραιότητα τη δραστική μείωση του αριθμού των θανάτων και των ασθενειών που οφείλονται σε επικίνδυνες χημικές ουσίες, καθώς και στη ρύπανση του αέρα, του νερού και του εδάφους. Το μονοξείδιο του άνθρακα **CO** αποτελεί ένα άοσμο, άχρωμο και εξαιρετικά τοξικό αέριο, η εισπνοή του οποίου σε υψηλές συγκεντρώσεις μπορεί να αποβεί μοιραία για την ανθρώπινη ζωή (ασφυξία, δηλητηρίαση).

Η προτεινόμενη διάταξη συμβάλλει άμεσα σε αυτόν τον στόχο, καθώς υλοποιεί έναν ανθεκτικό σε σφάλματα (fault-tolerant) IoT edge κόμβο που συνεχώς ελέγχει για παρουσία του μονοξειδίου του άνθρακα. Το σύστημα λειτουργεί ως μηχανισμός έγκαιρης προειδοποίησης, προστατεύοντας την ανθρώπινη υγεία σε οικιακά κυρίως περιβάλλοντα.

1.7 Στόχος 11: Βιώσιμες Πόλεις και Κοινότητες

Ο **στόχος 11.6** επικεντρώνεται στη μείωση του αρνητικού περιβαλλοντικού αντικτύπου των πόλεων ανά κάτοικο, δίνοντας ιδιαίτερη έμφαση στην ποιότητα του αέρα και στη διαχείριση των αστικών αποβλήτων. Η δημιουργία «Εξυπνων και Βιώσιμων Πόλεων» (Smart Cities) απαιτεί την ανάπτυξη πυκνών δικτύων περιβαλλοντικών αισθητήρων για τη χαρτογράφηση των ρύπων.

Η συνεισφορά της παρούσας ΜΔΕ έγκειται στη σχεδίαση ενός συστήματος χαμηλού κόστους (low-cost) βασισμένου σε COTS (commercial off the shelf) και λογισμικό ανοιχτού κώδικα (Elixir/Nerves). Το χαμηλό κόστος παραγωγής και η υψηλή αξιοπιστία του κόμβου επιτρέπουν τη μαζική γεωγραφική διασπορά του. Αυτό δίνει τη δυνατότητα στις τοπικές κοινωνίες και στους δήμους να συλλέγουν δεδομένα μικροκλίματος σε πραγματικό χρόνο των περιβαλλοντικών δεδομένων και επιτρέποντας τη λήψη αποφάσεων για τη βελτίωση της ποιότητας ζωής στις πόλεις.

2. Θεωρητικό Υπόβαθρο

Στο παρόν κεφάλαιο πραγματοποιείται μια ανάλυση του θεωρητικού, επιστημονικού και τεχνολογικού υποβάθρου, επάνω στο οποίο βασίστηκε η αρχιτεκτονική σχεδίαση, η υλοποίηση και η εργαστηριακή αξιολόγηση του προτεινόμενου συστήματος.

Η μελέτη διαρθρώνεται σε τέσσερις κεντρικούς άξονες:

- Αρχικά, εισάγεται το επιστημονικό πλαίσιο του διάχυτου υπολογισμού (Ubiquitous Computing) και η εξέλιξη των σύγχρονων αρχιτεκτονικών του Διαδικτύου των Πραγμάτων (IoT).
- Στη συνέχεια, εξετάζεται η σχετική βιβλιογραφία συστημάτων χαμηλού κόστους (low-cost IoT) που χρησιμοποιήθηκαν για παρόμοιο σκοπό. Δηλαδή την μέτρηση παραμέτρων για ανάλυση της ποιότητας του αέρα. Γίνεται επίσης αναφορά για τα λειτουργικά συστήματα που χρησιμοποιούνται για συστήματα IoT.
- Ο τρίτος άξονας επικεντρώνεται στο σύστημα λογισμικού που χρησιμοποιήθηκε, αναλύοντας τη γλώσσα Elixir, τη διατριβή του Joe Armstrong για την εικονική μηχανή BEAM (Erlang OTP), και το εργαλείο Elixir Nerves.
- Τέλος, το κεφάλαιο δίνει μια μικρή ανασκόπηση στα εξαρτήματα και στους αισθητήρες που θα χρησιμοποιηθούν.

2.1 Διάχυτος Υπολογισμός (Ubiquitous Computing) και το Παράδειγμα του IoT

Η εξέλιξη των υπολογιστικών συστημάτων κατά τις τελευταίες δεκαετίες χαρακτηρίζεται από τη μετάβαση από το παραδοσιακό μοντέλο του προσωπικού υπολογιστή (PC) σε κατανεμημένες, διάχυτες αρχιτεκτονικές. Όπως διατυπώθηκε στο θεμελιώδες έργο του John Krumm [\[7\]](#), ο Διάχυτος Υπολογισμός (Ubiquitous ή Pervasive Computing) ορίζεται ως η κατάσταση εκείνη κατά την οποία η υπολογιστική ισχύς και η δικτυακή συνδεσιμότητα ενσωματώνονται οργανικά και «αόρατα» στο φυσικό περιβάλλον και στα αντικείμενα της καθημερινότητας. Στόχος της επιστημονικής αυτής προσέγγισης είναι η δημιουργία συστημάτων που αλληλο-επιδρούν με τον άνθρωπο και το περιβάλλον με τρόπο διακριτικό,

χωρίς να απαιτούν την ενεργή προσοχή ή τη χειροκίνητη εισαγωγή δεδομένων από τον χρήστη (unobtrusive computing).

Στο πλαίσιο αυτό, το διαδίκτυο των πραγμάτων (IoT) αποτελεί την πρακτική, τεχνολογική μετεξέλιξη και εφαρμογή των αρχών του διάχυτου υπολογισμού. Σύμφωνα με τον Krumm [7], ένα σύστημα για να χαρακτηριστεί ως επιτυχές δείγμα διάχυτης υπολογιστικής πρέπει να ικανοποιεί τρεις βασικές ιδιότητες:

- **Αυτονομία:** Η υπολογιστική διάταξη πρέπει να είναι σε θέση να λειτουργεί απρόσκοπτα για μεγάλα χρονικά διαστήματα χωρίς την ανάγκη ανθρώπινης διαχείρισης, συντήρησης ή επανεκκίνησης.
- **Επίγνωση πλαισίου:** Το σύστημα πρέπει να αντιλαμβάνεται τις μεταβολές του φυσικού χώρου στον οποίο είναι εγκατεστημένο. Στην παρούσα διπλωματική, η επίγνωση αυτή επιτυγχάνεται μέσω της συνεχούς συλλογής δεδομένων μονοξειδίου του άνθρακα.
- **Διασυνδεσιμότητα:** Οι συλλεγόμενες πληροφορίες δεν πρέπει να παραμένουν απομονωμένες, αλλά να διοχετεύονται μέσω ασφαλών δικτυακών πρωτοκόλλων σε ανώτερα επίπεδα (Edge/Cloud) για περαιτέρω ανάλυση και οπτικοποίηση.

Η πρόκληση της σύγχρονης μηχανικής έγκειται στη σχεδίαση κόμβων (IoT edge nodes) που ικανοποιούν τις παραπάνω αρχές, διατηρώντας παράλληλα το κόστος υλοποίησης σε χαμηλά επίπεδα, ώστε να είναι δυνατή η μαζική παραγωγή και γεωγραφική διασπορά τους.

2.2 Σχετική Βιβλιογραφία Συστημάτων IoT Χαμηλού Κόστους (Low-Cost IoT)

Στην ακαδημαϊκή βιβλιογραφία και στις εφαρμογές ανοιχτού τύπου δηλαδή τις “open-source” η ανάπτυξη κόμβων IoT χαμηλού κόστους βασίζεται παραδοσιακά σε Commercial Off-The-Shelf (COTS) υλικό. Θα εξετάσουμε μερικές σχετικές με την μέτρηση της ποιότητας του αέρα και πως συσχετίζονται με την απόφασή μας για το υλικό που θα χρειαστούμε.

Στην δημοσίευση [8] παρουσιάζεται μια ολοκληρωμένη συγκριτική επισκόπηση εξήντα εμπορικά διαθέσιμων υπολογιστικών πλακετών commercial off the shelf IoT, με σκοπό την

υποστήριξη των προγραμματιστών στην επιλογή του κατάλληλου υλικού (hardware) για εξειδικευμένες εφαρμογές. Θα επικεντρωθούμε στο γεγονός πως γίνεται μνεία κυρίως σε Raspberry Pi και Arduino όπου χρησιμοποιούνται ευρέως στις φοιτητικές κοινότητες και όχι μόνο.

Στην [\[9\]](#), η αρχιτεκτονική του συστήματος χρησιμοποιεί περιβαλλοντικούς αισθητήρες (BME280, MQ135 και κάμερα) που μεταδίδουν δεδομένα σε έναν μικροελεγκτή ESP32 για την αρχική προ-επεξεργασία. Αυτά τα δεδομένα αποστέλλονται ασύρματα σε έναν κόμβο όπως το Raspberry Pi ή το Tinker Board, ο οποίος λειτουργεί ως κεντρικός διακομιστής υπεύθυνος για την αποθήκευση και την ανάλυση. Τέλος, ο SBC φιλοξενεί μια εφαρμογή βασισμένη στο Flask (εργαλείο για δημιουργία web εφαρμογών γραμμένο στην γλώσσα Python) για την τοπική διεπαφή χρήστη και την εξαγωγή δεδομένων σε Excel, ενώ συνδέεται επίσης με την πλατφόρμα Blynk για απομακρυσμένη παρακολούθηση μέσω εφαρμογών κινητού ή διαδικτύου.

Επίσης, στην [\[10\]](#) χρησιμοποιήθηκαν εργαλεία παραπλήσια του Raspberry Pi, Arduino και ESP32. Η αποφυγή της χρήσης αυτών των γνωστών αναπτυξιακών εργαλείων εξυπηρετούσε την ανάγκη για μια γενικευμένη μεθοδολογία, η οποία έπρεπε να είναι εύκολα αναπαραγώγιμη και εφαρμόσιμη σε πραγματικές συνθήκες από οποιονδήποτε ενδιαφερόμενο, μειώνοντας τα τεχνικά εμπόδια εισόδου.

Στην [\[11\]](#) χρησιμοποιούν κυρίως την τεχνολογία LoRAWAN και το Dragino για να διαχειριστούν το νερό σε πραγματικό χρόνο και να αποστέλλουν τα δεδομένα με ασύρματο τρόπο.

Το συμπέρασμα από την μελέτη των παραπάνω δημοσιεύσεων είναι πως γίνεται εκτεταμένη χρήση είτε ανοιχτού λογισμικού είτε πλατφορμών COTS. Δηλαδή, commercial off-the-shelf προϊόντων που μπορεί όλοι οι ενδιαφερόμενοι να αγοράσουν με εύκολο τρόπο. Όμως το μειονέκτημα είναι πως ο πηγαίος κώδικας είτε δεν μοιράζεται είτε αν μοιράζεται είναι σε μια μορφή μη αναπαραγώγιμη ώστε ο τελικός χρήστης/ια να το χρησιμοποιήσει άμεσα.

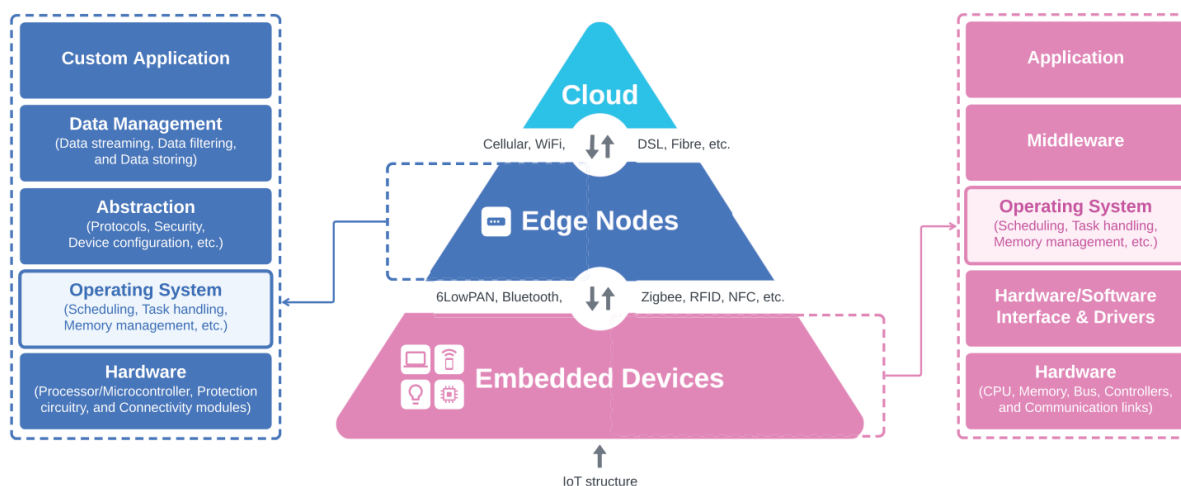
2.2.1 Μελέτη Περίπτωσης: Επικύρωση λειτουργίας αισθητήρων CO με χρήση προσιτών μέσων

Στην εργασία των Litwhiler και Lombardi [47], εξετάστηκε η απόδοση αισθητήρων μονοξειδίου του άνθρακα, συμπεριλαμβανομένου του μοντέλου TGS5042 που θα χρησιμοποιήσουμε εδώ σε αυτή την διπλωματική, με τη χρήση αυτοσχέδιων διατάξεων χαμηλού κόστους. Η έρευνα απέδειξε ότι είναι δυνατή η διεξαγωγή αξιόπιστων δοκιμών σε περιβάλλοντα με περιορισμένους πόρους, χρησιμοποιώντας θαλάμους κατασκευασμένους από κοινά υλικά και πηγές αερίου όπως τα καυσάερια αυτοκινήτων. Παρόλο που οι ερευνητές χρησιμοποίησαν ένα επαγγελματικό όργανο αναφοράς (Fluke CO-210) για τη βαθμονόμηση, τα αποτελέσματα για τον αισθητήρα TGS 5042 (ευαισθησία περίπου 2nA/ppm) ήταν αρκετά κοντά με τα τεχνικά χαρακτηριστικά που δημοσιεύει ο κατασκευαστής. Το γεγονός αυτό υποδηλώνει ότι η σύγκριση των πειραματικών δεδομένων απευθείας με τα datasheets του κατασκευαστή αποτελεί μια έγκυρη μέθοδο επαλήθευσης της ορθής λειτουργίας του συστήματος, προσφέροντας ικανοποιητικά αποτελέσματα με ελάχιστη επένδυση.

2.3 Λειτουργικά Συστήματα IoT

Στην [12] παρουσιάζεται μια λεπτομερή ανάλυση των κορυφαίων λειτουργικών συστημάτων που χρησιμοποιούνται στο οικοσύστημα του Internet of Things (IoT), εξετάζοντας πώς οι αρχιτεκτονικές τους επιλογές ανταποκρίνονται στις ανάγκες συσκευών με περιορισμένους πόρους.

- Τα λειτουργικά συστήματα τύπου IoT πρέπει να σχεδιάζονται έχοντας υπόψη τις μοναδικές απαιτήσεις και τους περιορισμούς των συσκευών IoT. Σε σχέση με τα συστήματα IoT, τα παραδοσιακά λειτουργικά συστήματα συχνά στερούνται την απαραίτητη υποστήριξη για χαμηλή κατανάλωση ενέργειας, περιορισμένη μνήμη και αποτελεσματική επεξεργασία δεδομένων που έχουν ανάγκη οι συσκευές IoT. Η Εικόνα 2, όπως εμφανίζεται στην δημοσίευση αυτή, δείχνει πού εντάσσονται αυτά τα λειτουργικά συστήματα μέσα στην πολυεπίπεδη αρχιτεκτονική των συστημάτων IoT.



Εικόνα 2 Το μοντέλο επιπέδων των ενσωματωμένων συσκευών (δεξιά) και των κόμβων άκρου (αριστερά) εντός των σταδίων της αρχιτεκτονικής του IoT.

Δηλαδή, τα λειτουργικά συστήματα IoT σχεδιάζονται για να «κουμπώνουν» σε συγκεκριμένα επίπεδα αυτής της δομής, καλύπτοντας το κενό μεταξύ του υλικού και των εφαρμογών.

Θα ακολουθήσει μια σύγκριση των συστημάτων μεταξύ τους καθώς αυτή θα είναι μια βασική προϋπόθεση για να δείξουμε αργότερα ποιο τμήμα της IoT στοίβας θα μπορούσε να καλύψει το Elixir Nerves ή αν όντως προσφέρει κάτι πιο χρήσιμο από τις τεχνολογίες που εμφανίζονται στην μελέτη αυτή.

Πίνακας 1. Συγκριτική Ανάλυση Λειτουργικών Συστημάτων IoT

Λειτουργικό Σύστημα	Αρχιτεκτονική Πυρήνα	Μοντέλο Προγραμματισμού	Υποστήριξη Πραγματικού Χρόνου	Κύρια Πρωτόκολλα Δικτύωσης	Καταλληλότητα (Embedded / Edge)
Linux	Μονολιθικός	Πολυνηματικό (Multithreaded)	Ναι	IPv4, IPv6, MQTT, 6LoWPAN, HTTPS, κ.α.	Υψηλή / Υψηλή
FreeRTOS	Microkernel	Πολυνηματικό (Multithreaded)	Ναι	Καμία εγγενής υποστήριξη (απαιτεί πρόσθετα)	Υψηλή / Χαμηλή

Zephyr	Microkernel	Πολυνηματικό (Multithreaded)	Ναι	IPv4, IPv6, CoAP, MQTT, Bluetooth, κ.α.	Υψηλή / Υψηλή
Windows IoT	Υβριδικός	Event-driven + Πολυνηματικό	Όχι	IPv4, IPv6, HTTPS, IPsec, κ.α.	Υψηλή / Υψηλή
Mbed OS	Microkernel	Event-driven + Πολυνηματικό	Ναι	IPv4, IPv6, Bluetooth, LoRaWAN, NB-IoT	Υψηλή / Μέτρια
Android	Μονολιθικός	Event-driven & Πολυνηματικό	Όχι	IPv4, IPv6, Wi-Fi, NFC, Cellular (5G)	Μέτρια / Υψηλή
FreeBSD	Μονολιθικός	Πολυνηματικό (Multithreaded)	Ναι	IPv4, IPv6, HTTPS, SSH, FTP, κ.α.	Υψηλή / Υψηλή
Azure Sphere	Υβριδικός	Πολυνηματικό (Multithreaded)	Ναι	IPv4, IPv6, MQTT, Wi-Fi, TLS, κ.α.	Μέτρια / Υψηλή
AIX	Υβριδικός	Πολυνηματικό (Multithreaded)	Όχι	IPv4, IPv6, HTTPS, SSH, NFS, κ.α.	Χαμηλή / Υψηλή
No OS (Bare Metal)	-	Εξαρτάται από την εφαρμογή	Ναι	Καμία (απαιτεί χειροκίνητη ανάπτυξη)	Υψηλή / Χαμηλή

Βλέπουμε κυρίως από την σύγκριση την ύπαρξη όρων όπως ο monolithic και microkernel και υβριδικού μοντέλου. Οι αρχιτεκτονικές αυτές αποτελούν τις βασικές προσεγγίσεις για τον σχεδιασμό του πυρήνα ενός λειτουργικού συστήματος.

2.4 Μονολιθικός Πυρήνας (Monolithic Kernel)

Σε αυτή την αρχιτεκτονική, όλα τα βασικά στοιχεία του λειτουργικού συστήματος και οι εφαρμογές συνδυάζονται και εκτελούνται ως ένα ενιαίο πρόγραμμα σε λειτουργία πυρήνα (kernel mode). Προσφέρει υψηλή απόδοση και ευρεία υποστήριξη υλικού, καθιστώντας τον κατάλληλο για σύνθετες εργασίες σε ενσωματωμένα συστήματα. Όμως, υστερεί σε ευελιξία. Επειδή το σύστημα είναι ενιαίο, η τροποποίηση ενός μόνο στοιχείου μπορεί να απαιτεί την επανεγγραφή ολόκληρης της εφαρμογής. Παραδείγματα: Linux, FreeBSD.

2.5 Microkernel

Οι υπηρεσίες του προγράμματος που τρέχει και οι υπηρεσίες πυρήνα διαχωρίζονται σε διαφορετικούς χώρους διευθύνσεων. Ο πυρήνας παρέχει μόνο τις απολύτως βασικές λειτουργίες, όπως ο χρονοπρογραμματισμός (scheduling), η επικοινωνία μεταξύ διεργασιών και ο συγχρονισμός. Όλες οι υπόλοιπες λειτουργίες διαχειρίζονται από ξεχωριστά νήματα (threads). Το σύστημα αυτό εστιάζει στην απλότητα και την επεκτασιμότητα. Είναι πολύ βολικό για το IoT, καθώς νέες δυνατότητες μπορούν να προστεθούν ως πρόσθετα (plugins) στον κεντρικό πυρήνα. Παραδείγματα: FreeRTOS, Mbed OS, Zephyr.

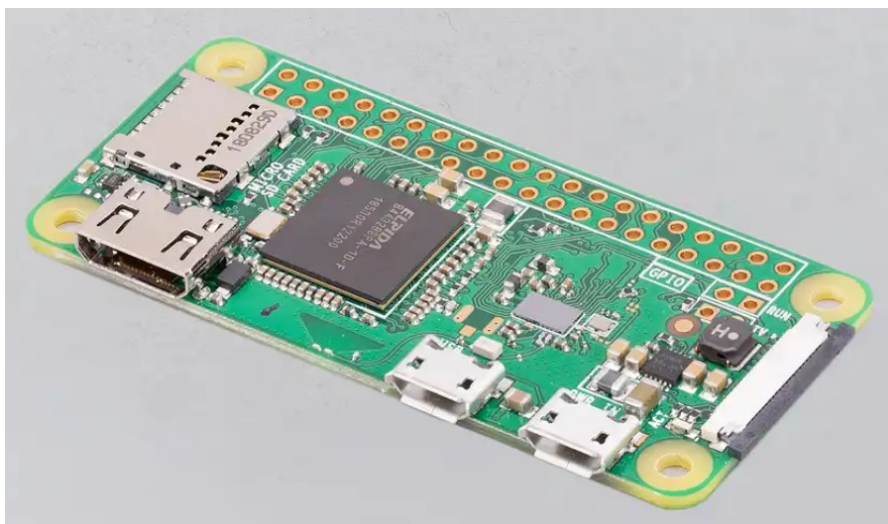
2.6 Υβριδικός Πυρήνας (Hybrid Kernel)

Η υβριδική αρχιτεκτονική συνδυάζει την αποδοτικότητα των microkernels με τη στιβαρότητα των μονολιθικών συστημάτων. Αυτό το μοντέλο επιτρέπει τη διαχείριση πιο δυναμικών φόρτων εργασίας και συνδέσεων. Συνήθως αυτή η αρχιτεκτονική χρησιμοποιείται από edge nodes που διαχειρίζονται περίπλοκα πρωτόκολλα επικοινωνίας.

Συνοπτικά, ενώ ένας microkernel είναι καλύτερος για τις πιο απλές συσκευές με εξαιρετικά περιορισμένους πόρους, και ένας μονολιθικός πυρήνας (monolithic kernel) είναι εξαιρετικός για την άμεση υποστήριξη υλικού, ο υβριδικός πυρήνας (Hybrid Kernel) αποτελεί την προτιμώμενη επιλογή για πλατφόρμες τύπου edge που πρέπει να εξισορροπήσουν την ασφάλεια, την ευελιξία και την ικανότητα διαχείρισης σύνθετης κίνησης δεδομένων.

2.7 Επιλογή αναπτυξιακής πλατφόρμας Raspberry Pi Zero Wireless

Για την υλοποίηση του παρόντος MVP, επιλέχθηκε ως COTS hardware το **Raspberry Pi Zero Wireless** (Εικόνα 3) καθώς αποτελεί μια πολύ γνωστή πλατφόρμα που μπορεί να αγοραστεί εύκολα και φθηνά από οποιοδήποτε ενδιαφερόμενο. Διαθέτει επεξεργαστή Broadcom BCM2835 (ARMv6) στο 1 GHz, 512 MB RAM DDR2 και ενσωματωμένο Wi-Fi 802.11 b/g/n. Η διασύνδεση με το αναλογικό layer πραγματοποιείται μέσω του διαύλου GPIO 40 ακίδων, ο οποίος υποστηρίζει τα ευρέως γνωστά πρωτόκολλα I2C (Inter-Integrated Circuit) για σύγχρονη επικοινωνία 2 συρμάτων (SDA, SCL) και SPI (Serial Peripheral Interface). Το βασικό πλεονέκτημα αυτής της πλατφόρμας είναι πως υπάρχει άπλετη πληροφορία για το πως σχεδιάστηκε (reference design) καθώς η ύπαρξη σύνδεσης των εμπλεκόμενων συσκευών μέσω του πρωτοκόλλου I2C. Όλες σχεδόν οι συσκευές που θα συνδεθούν στο Raspberry Pi Zero Wireless θα έχουν δυνατότητα σύνδεσης μέσω του I2C.



Εικόνα 3. Raspberry Pi Zero Wireless (W)

2.8 Το σύστημα λογισμικού: Elixir, Erlang BEAM και framework Elixir Nerves

Το εργαλείο προγραμματισμού Elixir Nerves βασίζεται πάνω στην γλώσσα προγραμματισμού Elixir και την εικονική μηχανή BEAM της Erlang. Λαμβάνοντας υπόψη και το λειτουργικό που θα χρησιμοποιηθεί για την ανάπτυξη της τελικής λύσης, επιλέγουμε πως το Raspberry PI θα λειτουργήσει σαν **κόμβος Edge Node** αλλά θα υλοποιεί και πολλές δυνατότητες που

παρέχει και κάποιος node τύπου IoT. Καθώς θα κάνει χρήση του Linux, θα ανήκει στην κατηγορία των **monolithic kernels** με δυνατότητες **edge node**.

2.9 Erlang/OTP

Η αρχιτεκτονική βάση του λογισμικού του συστήματος βασίζεται στη διδακτορική διατριβή του Joe Armstrong [\[13\]](#). Ο Armstrong ερεύνησε την ανάπτυξη μιας νέας μεθοδολογίας σχεδιασμού, η οποία αργότερα εξελίχθηκε στη γλώσσα προγραμματισμού Erlang και στο σύνολο τυπικών βιβλιοθηκών OTP (Open Telecom Platform). Το κεντρικό πρόβλημα που επιλύει η διατριβή του είναι η κατασκευή εξαιρετικά αξιόπιστων και εύρωστων κατανεμημένων συστημάτων (fault-tolerant systems) από προγράμματα τα οποία, από τη φύση τους, ενδέχεται να περιέχουν λογικά σφάλματα.

Η πρακτική ισχύς αυτής της προσέγγισης αποδείχθηκε σε μεγάλης κλίμακας εμπορικά τηλεπικοινωνιακά προϊόντα, όπως ο μεταγωγέας Ericsson AXD301, ο οποίος περιείχε περισσότερες από ένα εκατομμύριο γραμμές κώδικα Erlang, επιτυγχάνοντας πρωτοφανή επίπεδα διαθεσιμότητας (99.99999% uptime) [\[13\]](#). Τα θεμέλια της θεωρίας του Armstrong, τα οποία κληρονομεί αυτούσια η Elixir, συνοψίζονται στις εξής αρχές:

- **Προγραμματισμός προσανατολισμένος στον ταυτοχρονισμό (Concurrency-Oriented Programming):** Ο Armstrong υποστηρίζει ότι ο πραγματικός κόσμος είναι εγγενώς ταυτόχρονος, αποτελούμενος από ένα πλήθος γεγονότων που συμβαίνουν παράλληλα. Ως εννοιολογική μεταφορά ή παρομοίωση, χρησιμοποιεί την ανθρώπινη εμπειρία της οδήγησης: ένας οδηγός αντιλαμβάνεται εκατοντάδες άλλα οχήματα γύρω του και εκτελεί σύνθετες ενέργειες ταυτόχρονα, αποδεικνύοντας ότι ο εγκέφαλος λειτουργεί διαισθητικά με όρους ταυτοχρονισμού. Η βασική αξιωματική φράση της διατριβής είναι ότι «*η δομή του προγράμματος πρέπει να ακολουθεί τη δομή της πραγματικής δραστηριότητας*» [\[13\]](#). Κατά συνέπεια, εάν ένα πρόβλημα περιλαμβάνει 1.000 ανεξάρτητους παράγοντες που δρουν ταυτόχρονα, το λογισμικό πρέπει να αποτελείται από 1.000 ανεξάρτητες διεργασίες.
- **Απομόνωση διεργασιών (Process Isolation):** Κάθε διεργασία εντός της εικονικής μηχανής BEAM λειτουργεί ως μια εντελώς αυτόνομη, «απομονωμένη» εικονική μηχανή. Δεν υφίσταται κοινή μνήμη ή διαμοιρασμός πόρων. Λόγω αυτής της ισχυρής

απομόνωσης, ένα σφάλμα λογισμικού σε μια διεργασία είναι αδύνατο να διαδοθεί, να αλλοιώσει τη μνήμη ή να καταστρέψει την κατάσταση άλλων διεργασιών. Οι διεργασίες διαχειρίζονται απευθείας από τη BEAM και όχι από το λειτουργικό σύστημα, επιτρέποντας τη λειτουργία πολλών διεργασιών με ελάχιστο υπολογιστικό κόστος (μόλις 2.6 KB ανά διεργασία).

- **Ανταλλαγή μηνυμάτων (message passing):** Εφόσον αποκλείεται η κοινή μνήμη, οι διεργασίες αλληλοεπιδρούν αποκλειστικά μέσω ασύγχρονης ανταλλαγής μηνυμάτων. Τα μηνύματα είναι ατομικά (atomic) — παραδίδονται ολόκληρα ή καθόλου — και διατηρούν αυστηρά τη σειρά αποστολής τους. Κάθε διεργασία κατέχει ένα μοναδικό αναγνωριστικό (Process Identifier ή pid) το οποίο λειτουργεί ως η «διεύθυνση» για την αποστολή μηνυμάτων, επιτυγχάνοντας έτσι τον απόλυτο συγχρονισμό χωρίς τη χρήση επικίνδυνων «κλειδωμάτων» λογισμικού. Τα κλειδώματα λογισμικού είναι ένας είδος μηχανισμός συγχρονισμού για πρόσβαση σε πόρους όταν υπάρχουν πολλά νήματα εκτέλεσης.
- **Το Μοντέλο "Let it crash":** Η Erlang/Elixir απορρίπτει τη φιλοσοφία του αμυντικού προγραμματισμού (catch-all blocks). Οι διεργασίες σχεδιάζονται με την προοπτική να αποτύχουν γρήγορα (fail-fast) μόλις προκύψει μια εξαίρεση. Η ευθύνη της επιδιόρθωσης ανατίθεται σε μια εξωτερική διεργασία, εξασφαλίζοντας ότι το σφάλμα δεν θα «μολύνει» το υπόλοιπο σύστημα.

Σύμφωνα με τον Armstrong, για να χαρακτηριστεί μια γλώσσα και το υποκείμενο σύστημα ως ανθεκτικό σε σφάλματα, πρέπει να ικανοποιεί έξι βασικές απαιτήσεις [\[13\]](#):

1. **Ταυτοχρονισμός (Concurrency):** Ταυτόχρονη υποστήριξη χιλιάδων διεργασιών με μηδενικό υπολογιστικό κόστος κατά τη δημιουργία ή καταστροφή τους.
2. **Ενθυλάκωση σφαλμάτων (error encapsulation):** Πλήρης αδυναμία μιας προβληματικής διεργασίας να βλάψει τις υπόλοιπες.
3. **Ανίχνευση σφαλμάτων (fault detection):** Δυνατότητα ανίχνευσης εξαιρέσεων τόσο σε τοπικό επίπεδο όσο και σε απομακρυσμένες διεργασίες.
4. **Προσδιορισμός σφαλμάτων (fault identification):** Σαφής και άμεσος προσδιορισμός της αιτίας που προκάλεσε την εξαίρεση.

5. **Αναβάθμιση κώδικα (code upgrade):** Μηχανισμοί δυναμικής αλλαγής του κώδικα (hot code swapping) σε πραγματικό χρόνο, χωρίς διακοπή της λειτουργίας του συστήματος.

2.10 Η γλώσσα προγραμματισμού Elixir για συστήματα IoT

Η γλώσσα προγραμματισμού Elixir [\[14\]](#) δημιουργήθηκε το 2011 από τον José Valim ο οποίος ήταν βασικό μέρος της ομάδας ανάπτυξης του εργαλείου Ruby on Rails, το οποίο χρησιμοποιείται για δημιουργία διαδικτυακών εφαρμογών (web applications) με βάση την γλώσσα προγραμματισμού Ruby. Προσπαθώντας να δώσει λύση σε βασικά προβλήματα του ταυτοχρονισμού και της απόδοσης που αντιμετώπισε στο Ruby on Rails, στράφηκε προς την γλώσσα Erlang, και παίρνοντας αρκετά στοιχεία από την γλώσσα Ruby, δημιούργησε την γλώσσα Elixir που στηρίζεται πάνω στην εικονική μηχανή BEAM (Björn's Erlang Abstract Machine) της Erlang. Παρακάτω παραθέτουμε κάποιες δημοσιεύσεις όπου αναφέρεται η χρήση της Elixir για συστήματα IoT.

Στην δημοσίευση [\[15\]](#) γίνεται μια σύγκριση μεταξύ της Elixir και Java και η Elixir παρουσιάζει εξαιρετικές προοπτικές για την ανάπτυξη λογισμικού στο πλαίσιο του IoT. Τα βασικά σημεία στα οποία βασίζεται αυτό το συμπέρασμα, προκύπτουν από τη σύγκριση μιας υλοποίησης σε Elixir έναντι μιας σε Java:

- **Αποδοτικότητα μνήμης:** Η Elixir υπερτερεί σημαντικά στη διαχείριση πόρων, καθώς η κατανάλωση μνήμης παρέμεινε σταθερή γύρω στα 40MB, ενώ η Java παρουσίασε αύξηση από 50MB έως σχεδόν 1GB ανάλογα με τον φόρτο. Αυτό οφείλεται στο μοντέλο της BEAM που χρησιμοποιεί εξαιρετικά ελαφριές διεργασίες (processes).
- **Χρόνος απόκρισης:** Η Elixir επέδειξε καλύτερους χρόνους απόκρισης σε αιτήματα HTTP. Ειδικότερα, σε υψηλούς ρυθμούς (άνω των 1500 αιτημάτων/δευτερόλεπτο), η Java παρουσίασε απότομη υποβάθμιση της απόδοσης και σφάλματα συστήματος ("too many open files"), ενώ η Elixir διατήρησε πιο προβλέψιμη συμπεριφορά.

Όσο αναφορά την βελτιστοποίηση των διεργασιών στην [\[16\]](#) γίνεται εκτενής ανάλυση πολλών παραμέτρων που αφορούν την απόδοση της Elixir σε σχέση με άλλες γλώσσες καταλήγοντας

στο συμπέρασμα πως γλώσσα προγραμματισμού Elixir αποτελεί μια βιώσιμη και ιδιαίτερα αποτελεσματική επιλογή για συστήματα IoT.

Η δημοσίευση αυτή τεκμηριώνει ότι η Elixir υπερέχει στη διαχείριση υψηλού φόρτου και ταυτόχρονων συνδέσεων, προσφέροντας ταυτόχρονα εργαλεία για τη βελτιστοποίηση των πόρων. Οι δοκιμές έδειξαν ότι η Elixir μπορεί να διαχειριστεί έως και 10 φορές περισσότερα ταυτόχρονα αιτήματα HTTP σε σχέση με το Node.js και έως 35 φορές περισσότερα σε σχέση με την Python. Σε ακραία φορτία (10.000 ταυτόχρονα αιτήματα), η Elixir και η GO διατήρησαν την επιχειρησιακή τους σταθερότητα, ενώ η Python σταμάτησε να αποκρίνεται. Οι αρθρογράφοι, βασισμένοι στις αρχές του νόμου του Amdahl για τα όρια της παράλληλης επιτάχυνσης, ανέπτυξαν μια μαθηματική μέθοδο βελτιστοποίησης. Χρησιμοποιώντας τις παραμέτρους υπολογιστικού κέρδους $G(n,m)$ και κλίσης κέρδους $G'(n,m,k)$, το μοντέλο τους προσδιορίζει τον βέλτιστο αριθμό διεργασιών (processes) πριν εμφανιστεί το φαινόμενο των φθινουσών αποδόσεων. Αυτό επιτρέπει στους προγραμματιστές να βρουν τη χρυσή τομή μεταξύ απόδοσης και κόστους υλικού, αποφεύγοντας τη σπατάλη πόρων. Ενδεικτικά, η χρήση 256 διεργασιών κρίθηκε μη πρακτική, καθώς η οριακή βελτίωση στην ταχύτητα δεν δικαιολογούσε την υπερβολική δέσμευση μνήμης.

2.10.1 Υλοποίηση στην Εικονική Μηχανή BEAM και το εργαλείο Elixir Nerves

Το οικοσύστημα της Elixir και του Elixir Nerves ενσωματώνει τη θεωρία του Armstrong στις ανάγκες ενός σύγχρονου IoT edge node, εισάγοντας δύο κρίσιμους μηχανισμούς:

- **Δέντρα εποπτείας (supervision Trees):** Οι ειδικές διεργασίες-επόπτες (Supervisors) [17] λειτουργούν ως εξελιγμένα λογισμικά watchdogs. Παρακολουθούν αδιάλειπτα τις διεργασίες και, σε περίπτωση σφάλματος, εφαρμόζουν στρατηγικές επανεκκίνησης επαναφέροντας τις διεργασίες σε μια γνωστή, ασφαλή κατάσταση.
- **Generic server (gen_server):** Αυτός ο μηχανισμός [18] αποτελεί ένα τυποποιημένο πρότυπο για την υλοποίηση της αρχιτεκτονικής πελάτη-διακομιστή (client-server) σε περιβάλλοντα ταυτόχρονου προγραμματισμού. Μέσω ενός δομημένου συνόλου συναρτήσεων ανάκλησης (callback functions), το συγκεκριμένο μοντέλο αφαιρεί την πολυπλοκότητα της διαχείρισης καταστάσεων και των χαμηλού επιπέδου διεργασιών,

διευκολύνοντας τον συγχρονισμό και την ασύγχρονη επικοινωνία μεταξύ των τμημάτων του συστήματος.

2.10.2 Εργαλείο Προγραμματισμού IoT Elixir Nerves

Το **Elixir Nerves** [\[19\]](#) αποτελεί ένα εξειδικευμένο εργαλείο ανάπτυξης (open-source framework) ανοικτού κώδικα, βασισμένο στη γλώσσα προγραμματισμού Elixir, το οποίο προορίζεται για τη σχεδίαση, κατασκευή λογισμικού για ενσωματωμένα συστήματα (embedded systems) που τρέχουν κυρίως σε single boards υπολογιστές. Δηλαδή, το Elixir Nerves από την αρχή δεν κάλυπτε το πεδίο των IoT συστημάτων που υλοποιούνται κυρίως από μικροελεγκτές. Η θεμελιώδης φιλοσοφία του Nerves είναι πως συμπύκνωση την ίδια την εφαρμογή και το λειτουργικό σύστημα σε ένα ενιαίο, συμπαγές και αυτόνομο είδωλο (firmware image) το οποίο στην συνέχεια εγγράφεται πάνω στην μνήμη sdcard ή eMMC.

2.10.3 Αρχιτεκτονική του Nerves βασισμένη στο Linux Πυρήνα

Η αρχιτεκτονική του Nerves βασίζεται δομικά στον πυρήνα του Linux (Linux kernel), υιοθετώντας μια αυστηρά μινιμαλιστική προσέγγιση. Σε αντίθεση με τις συμβατικές διανομές Linux για embedded πλατφόρμες (όπως το Raspberry Pi OS ή το Ubuntu Embedded), οι οποίες επιβαρύνουν το σύστημα με μη απαραίτητες υπηρεσίες, γραφικά περιβάλλοντα και εργαλεία διαχείρισης πακέτων, το Nerves βελτιστοποιεί τους πόρους του συστήματος μέσω των εξής μηχανισμών:

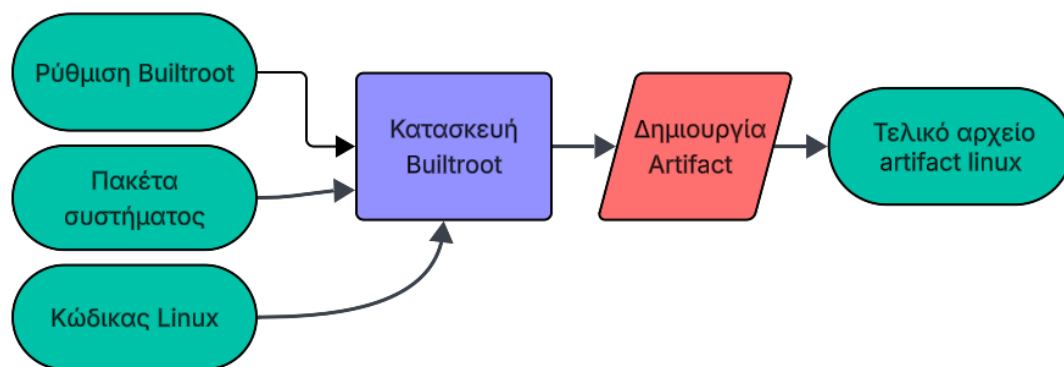
- **Εξειδικευμένη μεταγλώττιση μέσω Buildroot:** Το Nerves ενσωματώνει το εργαλείο αυτοματοποίησης Buildroot που είναι υπεύθυνο για δημιουργία Linux συστημάτων [\[20\]](#). Το buildroot απομονώνει και μεταγλωττίζει αποκλειστικά τον πυρήνα Linux και τους οδηγούς συσκευών που απαιτούνται για τη συγκεκριμένη αρχιτεκτονική υλικού (π.χ. ARM). Οτιδήποτε πλεονάζον αφαιρείται, μειώνοντας το μέγεθος του τελικού firmware στα 20-30 Megabytes.
- **Κατάργηση του παραδοσιακού user space (init process):** Στα τυπικά λειτουργικά συστήματα Linux, ο πυρήνας εκκινεί μια διεργασία αρχικοποίησης (όπως το systemd), η οποία αναλαμβάνει τη φόρτωση του user space. Στο Nerves, η εικονική μηχανή της Erlang/Elixir (BEAM) υποκαθιστά πλήρως τη διεργασία init με το erlinit [\[21\]](#). Το

erlinit αποτελεί το πρώτο και μοναδικό πρόγραμμα που εκτελείται αμέσως μετά τη φόρτωση του πυρήνα, αποκτώντας τον απόλυτο έλεγχο των διεργασιών της συσκευής. Συγκεκριμένα, με αυτόν τον τρόπο, η γλώσσα Erlang αναλαμβάνει τον έλεγχο του συστήματος.

- **Βελτιστοποίηση χρόνου εκκίνησης (boot time):** Λόγω της εξάλειψης των ενδιάμεσων επιπέδων λογισμικού (software layers) και των περιττών διεργασιών, ο χρόνος από το άνοιγμα της συσκευής έως την πλήρη εκτέλεση του κώδικα της εφαρμογής ελαχιστοποιείται, διαρκώντας συχνά **λιγότερο από 20 δευτερόλεπτα**.

Οι παρακάτω εικόνες θα βοηθήσουν τον αναγνώστη να κατανοήσει σε γενικά πλαίσια πως ακριβώς λειτουργεί όλη η διαδικασία προγραμματισμού ενός συστήματος με τον λογισμικό Elixir Nerves.

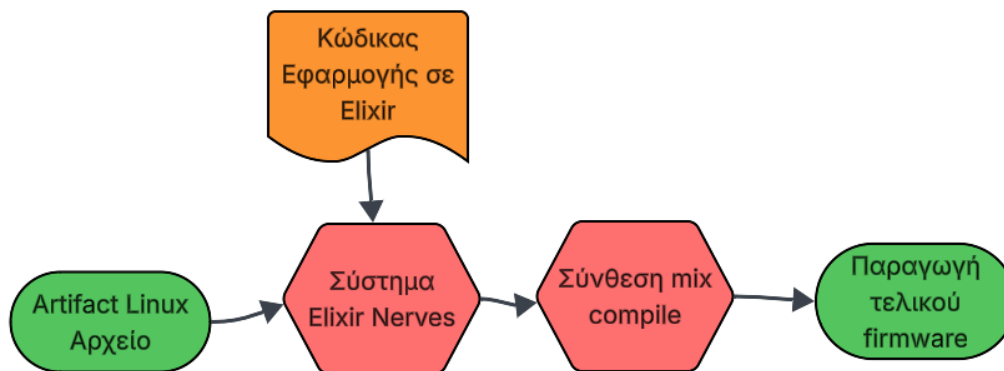
Αρχικά, μέσω του Buildroot δημιουργείται η λεγόμενη διανομή που τρέχει σε συγκεκριμένες πλατφόρμες όπως εμφανίζονται στο [\[22\]](#). Για το Raspberry PI Zero Wireless, υπάρχει το αντίστοιχο artifact ή διανομή, το *NervesSystemRPi0*. Στην Εικόνα 4, φαίνεται η διαδικασία δημιουργία διανομής μέσω του buildroot. Το τελικό αρχείο που παράγεται είναι βασικά το root file system που θα συγχωνευτεί με τον κώδικα Elixir της εφαρμογής που θα γραφτεί.



Εικόνα 4. Δημιουργία artifact/διανομής για το Elixir Nerves

Κατά την διάρκεια της δημιουργίας firmware, το οποίο θα γραφτεί στην κάρτα SDCARD του Raspberry PI, το σύστημα αναλαμβάνει να συγχωνεύσει την αντίστοιχη διανομή για το

rasberry pi zero wireless με τον κώδικα της εφαρμογής παράγοντας με αυτόν το τρόπο το τελικό αρχείο firmware. Η διαδικασία αυτή φαίνεται στην Εικόνα 5. Το “mix compile” είναι σύνολο εντολών του εργαλείου Elixir Nerves.



Εικόνα 5. Δημιουργία Firmware Elixir Nerves

2.10.4 Σύστημα Αρχείων Elixir Nerves

Η συγκεκριμένη προσέγγιση που βασίζεται στο Linux προσφέρει σημαντικά πλεονεκτήματα όπως:

- **Αποδοτική διαχείριση υλικού:** Η εφαρμογή επικοινωνεί άμεσα με τα υποσυστήματα του πυρήνα για τη διαχείριση των διασυνδέσεων χαμηλού επιπέδου (GPIO, I2C, SPI, UART), με αντίστοιχες βιβλιοθήκες, αξιοποιώντας στο έπακρο τις δυνατότητες του hardware.
- **Ακεραιότητα συστημάτων αρχείων:** Επιτρέπει την υλοποίηση της αρχιτεκτονικής των διαμερισμάτων στην αποθήκευση του firmware (A/B partitioning) θωρακίζοντας τη συσκευή έναντι διαφθοράς δεδομένων. Επειδή μια εφαρμογή χρειάζεται να αποθηκεύει πληροφορίες (π.χ. ρυθμίσεις, logs), το Nerves δημιουργεί ένα ξεχωριστό, ανεξάρτητο partition (τμήμα) που ονομάζεται **app_data** και γίνεται mount στο φάκελο **/data**. Αυτό το τμήμα είναι το μόνο που επιτρέπει εγγραφή και ανάγνωση (read/write).
- **Σύστημα αρχείων μόνο για ανάγνωση (Read-Only Root Filesystem):** Από προεπιλογή, το κεντρικό τμήμα του λειτουργικού συστήματος και η εφαρμογή Elixir (το firmware) γίνονται mount σε λειτουργία **read-only**. Αυτό σημαίνει ότι η συσκευή

δεν γράφει ποτέ νέα δεδομένα σε αυτό το κομμάτι κατά την κανονική της λειτουργία. Ακόμα και αν κοπεί απότομα το ρεύμα, το firmware παραμένει ανέπαφο και ασφαλές.

Παρακάτω (εικόνα 6) παρουσιάζεται η δομή του firmware όταν εγγράφεται στην κάρτα sdcard [23]. Στο πρώτο *partition p0*: *Boot A* εγγράφονται οι ρυθμίσεις εκκίνησης ενώ στο *Rootfs: A* εγγράφεται το linux σύστημα με την εφαρμογή ενσωματωμένη. Το *partition p2: Application* χρησιμοποιείται για να εγγράφονται εκεί δεδομένα και ρυθμίσεις της εφαρμογής τα οποία δεν σβήνονται αν χρειαστεί να γίνει ενημέρωση του firmware πάλι. Αξίζει να σημειωθεί πως υπάρχει και άλλο *partition B* που χρησιμοποιείται κατά την ενημέρωση του firmware. Η ενημέρωση γίνεται στο 2^ο *partition* και αν αυτή είναι επιτυχής, τότε αυτό το *partition* γίνεται το ενεργό. Διαφορετικά, φορτώνεται πάλι το παλιό *partition*. Με αυτόν τον τρόπο, αν συμβεί κάποιο σφάλμα κατά την διάρκεια της εγγραφής του firmware, η συσκευή επιστρέφει πίσω στην προηγούμενη λειτουργία και δεν απενεργοποιείται λόγω σφάλματος. Στην εικόνα 6 φαίνεται με αναλυτικό τρόπο η δομή διαμερισμάτων που χρησιμοποιεί το Elixir Nerves για να γράψει το firmware σε μια κάρτα sdcard που χρησιμοποιείται από το raspberry pi.

```
+-----+
| MBR                                         |
+-----+
| Firmware configuration data                |
| (formatted as uboot env)                  |
+-----+
| p0*: Boot A                               (FAT32) |
| zImage, bootcode.bin,                     |
| config.txt, etc.                           |
+-----+
| p0*: Boot B                               (FAT32) |
+-----+
| p1*: Rootfs A                             (squashfs) |
+-----+
| p1*: Rootfs B                             (squashfs) |
+-----+
| p2: Application                           (EXT4) |
+-----+
```

Εικόνα 6. Δομή διαμερισμάτων στο firmware

2.11 Αισθητήρες και εξαρτήματα

Οι αισθητήρες που θα χρησιμοποιηθούν και τα επιπλέον εξαρτήματα για την υλοποίηση του συστήματος είναι οι εξής:

2.11.1 TGS5042 – Αισθητήρας μονοξειδίου του άνθρακα.

Πριν ξεκινήσουμε να αναλύσουμε τον συγκεκριμένο αισθητήρα, είναι πρέπει να παραθέσουμε κάποια στοιχεία για τον μονοξείδιο του άνθρακα, το οποίο είναι ένα ύπουλο και τοξικό αέριο για την υγεία.

2.11.1.1 Μονοξείδιο του άνθρακα – επιπλοκές

Η [\[24\]](#) εξετάζει τις συστηματικές επιπτώσεις στην ανθρώπινη υγεία από την παρατεταμένη έκθεση σε χαμηλές συγκεντρώσεις μονοξειδίου του άνθρακα CO οι οποίες κυμαίνονται τυπικά μεταξύ 10 και 50 ppm. Οι συγγραφείς τεκμηριώνουν ότι αυτή η χρόνια, χαμηλού επιπέδου έκθεση οδηγεί σε σταδιακή αύξηση των επιπέδων ανθρακυλοαιμοσφαιρίνης COHb στο αίμα (1.5% έως 10%), προκαλώντας ύπουλα νευρολογικά και γνωστικά συμπτώματα που συχνά διαγιγνώσκονται λανθασμένα ως κοινές ιώσεις. Επιπλέον, η μελέτη υπογραμμίζει τον αυξημένο κίνδυνο πρόκλησης ή επιδείνωσης σοβαρών καρδιαγγειακών παθήσεων, καθώς και τις αρνητικές επιπτώσεις στην ανάπτυξη του εμβρύου κατά την εγκυμοσύνη. Συμπερασματικά, η εργασία αναδεικνύει την ανάγκη για συνεχή και ακριβή έλεγχο των εσωτερικών χώρων, καθώς οι παραδοσιακοί δείκτες οξείας τοξικότητας αποτυγχάνουν να εντοπίσουν τους κινδύνους της μακροχρόνιας έκθεσης. Στην [\[25\]](#) αναλύεται πως η αιμοσφαιρίνη είναι πρωτεΐνη και η κύρια λειτουργία της είναι να δεσμεύει το οξυγόνο και να το μεταφέρει σε όλο το σώμα για τη λειτουργία των ιστών. Επίσης, η ανθρακυλοαιμοσφαιρίνη σχηματίζεται από την ένωση του μονοξειδίου του άνθρακα με την αιμοσφαιρίνη και αποτελεί τον κύριο κλινικό δείκτη δηλητηρίασης. Εκτοπίζει το οξυγόνο από τα ερυθρά αιμοσφαίρια, μειώνοντας τη συνολική ποσότητα οξυγόνου που μεταφέρεται. Επίσης, εμποδίζει την απελευθέρωση του εναπομείναντος οξυγόνου στους ιστούς, προκαλώντας κυτταρική υποξία.

Στον πίνακα 2, από την δημοσίευση [\[25\]](#), φαίνεται η συγκέντρωση σε ppm του μονοξειδίου του άνθρακα και τα σχετιζόμενα συμπτώματα.

Πίνακας 2 Συγκεντρώσεις CO, Επίπεδα COHb και Συμπτώματα

Συγκέντρωση CO (ppm)	Επίπεδο COHb	Σημεία και Συμπτώματα
35 ppm	<10%	Πονοκέφαλος και ζάλη μετά από 6 έως 8 ώρες συνεχούς έκθεσης.
100 ppm	>10%	Ελαφρύς πονοκέφαλος σε 2 έως 3 ώρες.
200 ppm	20%	Ελαφρύς πονοκέφαλος μέσα σε 2 έως 3 ώρες, απώλεια κρίσης.
400 ppm	25%	Μετωπιαίος πονοκέφαλος μέσα σε 1 έως 2 ώρες.
800 ppm	30%	Ζάλη, ναυτία και σπασμοί μέσα σε 45 λεπτά, απώλεια αισθήσεων εντός 2 ωρών.
1.600 ppm	40%	Πονοκέφαλος, ταχυκαρδία, ζάλη και ναυτία σε 20 λεπτά, θάνατος σε λιγότερο από 2 ώρες.
3.200 ppm	50%	Πονοκέφαλος, ζάλη και ναυτία σε 5-10 λεπτά, θάνατος εντός 30 λεπτών.
6.400 ppm	60%	Πονοκέφαλος και ζάλη σε 1-2 λεπτά, σπασμοί, αναπνευστική ανακοπή και θάνατος σε λιγότερο από 20 λεπτά.
12.800 ppm	>70%	Θάνατος σε λιγότερο από 3 λεπτά.

Η μονάδα ppm σημαίνει parts per million και χρησιμοποιείται για να δώσει αναφορά για μεγέθη που είναι εξαιρετικά μικρά. Αν δηλαδή, μπορούμε και τεμαχίζουμε ένα χώρο σε 1.000.000 τμήματα και τα 50 τμήματα αυτά τα καταλάμβαναν τα μόρια του CO, τότε λέμε πως έχουμε 50 ppm CO. Δεν θα είχαμε την ίδια ευκολία να το κάνουμε αυτό με το ποσοστό %.

Αφού κάναμε μια εισαγωγή στην περιγραφή του μονοξειδίου του άνθρακα, θα αρχίσουμε να εξηγούμε και να περιγράφουμε τον αισθητήρα μονοξειδίου του άνθρακα που θα χρησιμοποιήσουμε.

Ο αισθητήρας αυτός είναι ο ηλεκτροχημικός αισθητήρας TGS5042 της εταιρίας Figaro [\[26\]](#) όπως εμφανίζεται στην εικόνα 7. Τα βασικά χαρακτηριστικά του αισθητήρα αυτού εμφανίζονται στην εικόνα 8.



Εικόνα 7 Αισθητήρας TGS-5042

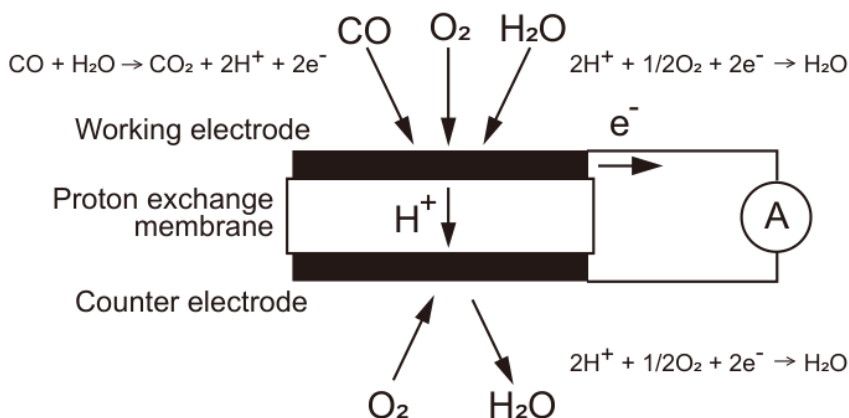
Item	Specification
Model number	TGS5042-A00
Target gases	Carbon monoxide
Typical detection range	0 ~ 10,000ppm
Output current in CO	1.2~2.4nA/ppm
Baseline offset(*1)	<±10ppm equivalent
Operating temperature(*2,*3)	0°C ~ +50°C (continuous) -5°C ~ +55°C (intermittent)
Operating humidity	5 ~ 99%RH (no condensation)
Response time (T90)	within 60 seconds
Storage conditions (*2,*3)	-5°C ~ +55°C
Weight	approx. 12g
Standard test conditions	20±2°C, 40±10%RH

Εικόνα 8. Χαρακτηριστικά Αισθητήρα TGS5042

Ο αισθητήρας αυτός έχει το μέγεθος μιας μπαταρίας AA και παράγει ένα μικρό ρεύμα όταν εκτεθεί σε μονοξείδιο του άνθρακα. Είναι αναλογικός, που σημαίνει πως σε έκθεση μεγαλύτερης ποσότητας CO παράγει περισσότερο ρεύμα. Το ρεύμα που παράγει είναι πολύ μικρό, και όπως φαίνεται και από την εικόνα 8, είναι της τάξης των nano αμπερ ανά ppm(parts per million). Βασίζεται κυρίως σε ηλεκτρολύτη που δημιουργεί ηλεκτρόνια και πρωτόνια όταν έρθει σε επαφή με μόρια του μονοξειδίου του άνθρακα και οξυγόνου. Παρακάτω στην εικόνα

9, φαίνεται πιο καθαρά ο τρόπος με τον οποίο λειτουργεί ο αισθητήρας. Στην πραγματικότητα πρόκειται για ένα κυψέλη καυσίμου που παράγει ρεύμα όταν έρθει σε επαφή με το μονοξείδιο του άνθρακα. Σε αντίθεση με την απλή μπαταρία, δεν έχει αποθηκευμένη ενέργεια αλλά την παράγει όταν έρθει σε επαφή με κάποιο στοιχείο. Στην συγκεκριμένη περίπτωση, παράγει ενέργεια όταν έρθει σε επαφή με το CO.

Τα βασικά του πλεονεκτήματα είναι πως α) αντιδρά μόνο στο μονοξείδιο του άνθρακα β) είναι φιλικό προς το περιβάλλον και δεν περιέχει τοξικές ενώσεις, γ) έχει μεγάλη διάρκεια ζωής μέχρι 7 χρόνια, δ) δεν καταναλώνει ενέργεια ε) έχει εξαιρετική γραμμική απόκριση στην ύπαρξη CO. Αξίζει να σημειωθεί πως η φορά του ρεύματος είναι από το ηλεκτρόδιο Working προς Counter. Στην εικόνα 6, το ηλεκτρόδιο Working είναι αυτό που εμφανίζεται μπροστά στην φωτογραφία έχοντας μια ακίδα μόνο.



Εικόνα 9. Αρχή λειτουργίας TGS5042

2.11.2 Αισθητήρας Θερμοκρασίας BME680

Ο αισθητήρας αυτός [27] είναι σε μορφή breakout board, είναι έτοιμος σχεδιαστικά να συνδεθεί μέσω I2C στο Raspberry PI Wireless Zero (Εικόνα 10). Παρέχει περισσότερες μετρήσεις εκτός θερμοκρασίας όπως υγρασία, βαρομετρική πίεση και αέριο VOC (Πτητικές Οργανικές Ενώσεις / Volatile Organic Compounds). Συγκεκριμένα, θα χρησιμοποιήσουμε μόνο τις δειγματοληψίες της θερμοκρασίας καθώς αυτές θα χρησιμοποιηθούν για να γίνει

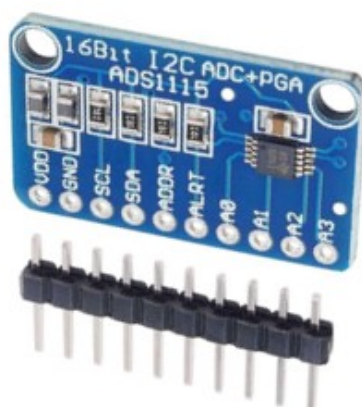
βαθμονόμηση στα αποτελέσματα του αισθητήρα TGS5042. Η μικρή αυτή πλακέτα τροφοδοτείται από το ρεύμα των 3.3 Volt που παρέχει το Raspberry PI Wireless Zero.



Εικόνα 10. Αισθητήρας Θερμοκρασίας BME680

2.11.3 Μετατροπéας σήματος Αναλογικό προς Ψηφιακό – ADS1115 ADC

Η έξοδος του αισθητήρα TGS5042, όπως αναφέραμε, είναι ένα μικρό ρεύμα που πρέπει να ενισχυθεί και να μετατραπεί σε μια τάση ρεύματος η οποία πρέπει να ψηφιοποιηθεί κατάλληλα. Επιλέξαμε για αυτό το σκοπό ένα κλασικό και δυνατό ADC converter από την Texas Instruments, τον ADS1115 adc [\[28\]](#). Επιπλέον, η πλακέτα αυτή τροφοδοτείται από το ρεύμα των 3.3V που παρέχει το *pi0* (Εικόνα 11) και έχει δίαυλο επικοινωνίας το I2C.



Εικόνα 11. ADS1115 ADC μετατροπέας

2.11.4 Τελεστικός Ενισχυτής Microchip MCP6042

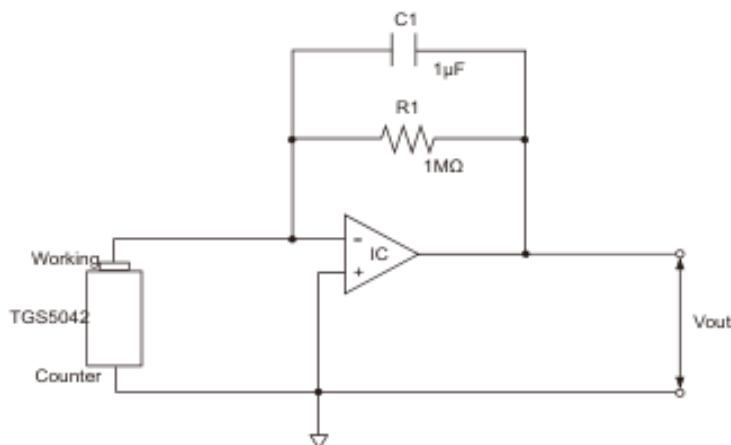
Σύμφωνα με συγκεκριμένα application notes [\[26\]](#) του TGS5042, το ασθενές ρεύμα πρέπει να ενισχυθεί από μία αναλογική διάταξη/κύκλωμα το οποίο περιέχει και έναν ή δυο τελεστικούς ενισχυτές. Όπως αναφέρεται και στο κλασικό βιβλίο του *Sendra & Smith* [\[29\]](#), ο τελεστικός ενισχυτής είναι ένα εξαιρετικά ευέλικτο ολοκληρωμένο κύκλωμα που λειτουργεί ως ενισχυτής τάσης. Αποτελεί ένα από τα θεμελιώδη δομικά στοιχεία των σύγχρονων αναλογικών ηλεκτρονικών. Στον πυρήνα του, ένας τελεστικός ενισχυτής λαμβάνει μια μικροσκοπική διαφορά ανάμεσα σε δύο τάσεις εισόδου και την ενισχύει σε μια πολύ μεγαλύτερη τάση εξόδου.

Η εταιρία Figaro προτείνει έναν αριθμό διάφορων τελεστικών ενισχυτών και ένας από αυτούς είναι ο MCP6042 (Εικόνα 12). Τα κύρια χαρακτηριστικά του είναι η χαμηλή κατανάλωση ενέργειας και είναι κατάλληλος για σήματα που αλλάζουν σχετικά αργά, όπως ο αισθητήρας TGS5042, ο οποίος έχει απόκριση μέσα σε 60 δευτερόλεπτα (Εικόνα 8).



Εικόνα 12. Τελεστικός Ενισχυτής MCP6042

Μια απλή διάταξη για το πως θα μπορούσε να χρησιμοποιηθεί ο τελεστικός ενισχυτής φαίνεται στην εικόνα 13. Στην πραγματικότητα, όμως επιλέξαμε να δημιουργήσουμε ένα πιο περίπλοκο αναλογικό κύκλωμα για την ενίσχυση σήματος. Αυτό θα το αναλύσουμε στα επόμενα κεφάλαια.



Εικόνα 13. Απλή διάταξη ενίσχυσης σήματος, application note από την Figaro

2.11.5 Κλειδί κρυπτογράφησης Nerves Key

Το NervesKey [\[30\]](#) είναι ένα κατάλληλα διαμορφωμένο τσιπ κρυπτογραφικής ταυτοποίησης (Crypto Authentication chip) τύπου ATECC508A ή ATECC608A, το οποίο χρησιμοποιείται για την αυθεντικοποίηση συσκευών κατά την διάρκεια σύνδεσης με πλατφόρμες IoT. Πρόκειται για μια απλή μονάδα ασφαλείας υλικού (Hardware Security Module - HSM), η οποία προστατεύει ένα ιδιωτικό κλειδί επιβάλλοντας την εκτέλεση όλων των διεργασιών που

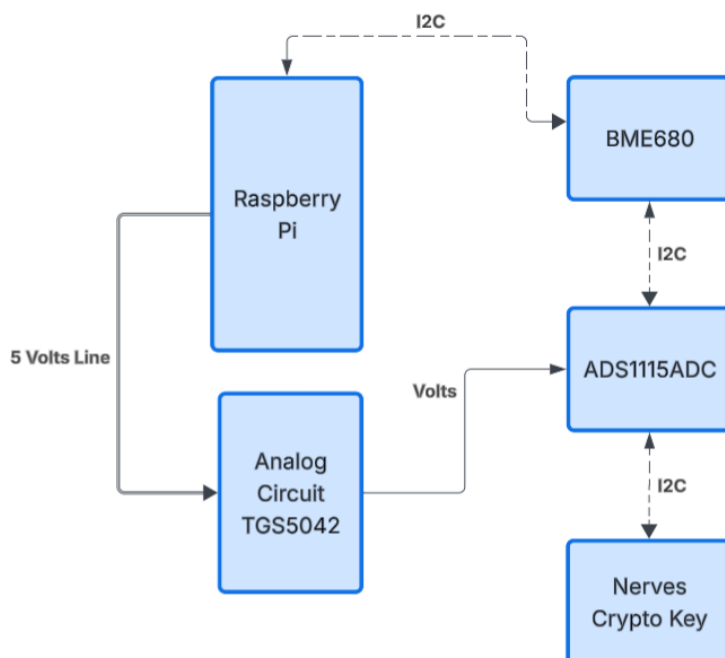
αφορούν το συγκεκριμένο κλειδί αποκλειστικά στο εσωτερικό του τσιπ. Το συγκεκριμένο chip διατίθεται σε μορφή breakout board (Εικόνα 14) με τροφοδοσίας 3.3 V και δίαυλο επικοινωνίας το I2C. Επομένως, μπορεί να συνδεθεί με το `gpi0` πολύ εύκολα και μέσω των κατάλληλων βιβλιοθηκών να λειτουργήσει σαν κρυπτογραφικό μέσο για την ενδιάμεση ταυτοποίηση. Δεν είναι απαραίτητο μέσο για την επικοινωνία με πλατφόρμες IoT αλλά κάποιες όπως το ThingsBoard παρέχουν σύνδεση με πιστοποιητικά X.509 και αυτό καθιστά το nerves key απαραίτητο για την μεταφορά δεδομένων.



Εικόνα 14. Chip Κρυπτογραφικής Ταυτοποίησης - Nerves Key

3. Σχεδιασμός & αρχιτεκτονική συστήματος

Αρχικά, χρειάστηκε να σχεδιαστεί και να δημιουργεί ειδικό αναλογικό κύκλωμα το οποίο ενισχύει το σήμα από τον TGS5042 ώστε να μπορέσει να γίνει σωστά η ψηφιοποίηση από τον ADS1115. Όμως, πριν αναλύσουμε το αναλογικό κύκλωμα που δημιουργήθηκε, εμφανίζουμε παρακάτω σε απλουστευμένη μορφή block diagram τη ολική διάταξη του συστήματος με τις βασικές διασυνδέσεις(Εικόνα 15).



Εικόνα 15. Διάταξη ολικού συστήματος

Στην διάταξη φαίνεται πως οι 3 συσκευές που λειτουργούν με το πρωτόκολλο I2C είναι συνδεδεμένες μεταξύ τους σε σειρά (daisy chain). Το πρωτόκολλο I2C χρησιμοποιείται ως ο κεντρικός διάυλος επικοινωνίας (bus) του συστήματος. Το Raspberry Pi Wireless Zero λειτουργεί ως κύρια συσκευή(master), αρχίζοντας την επικοινωνία μέσω μιας ενιαίας γραμμής I2C. Οι υπόλοιπες ψηφιακές συσκευές (BME680, ADS1115 ADC και Nerves Crypto Key) είναι συνδεδεμένες παράλληλα στον ίδιο διάυλο σαν δευτερεύουσες συσκευές, όπου η καθεμία αναγνωρίζεται από τη δική της μοναδική διεύθυνση I2C.

Το αναλογικό κύκλωμα με τον ενσωματωμένο TGS5042 αισθητήρα τροφοδοτείται από την γραμμή των 5V του Raspberry Pi Wireless Zero. Επίσης, η τάση που παράγεται από το αναλογικό κύκλωμα γίνεται είσοδος (Vout) προς τον ADS1115 όπου ψηφιοποιείται και μεταφέρεται στο Raspberry Pi για περαιτέρω ανάλυση.

3.1 Σχεδιασμός αναλογικού κυκλώματος

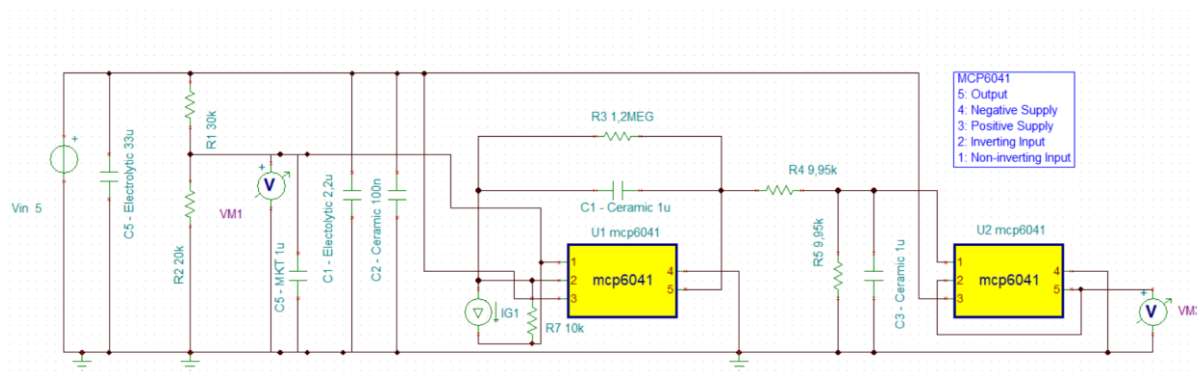
3.1.1 Απαιτήσεις συστήματος

Συνήθως, για να ξεκινήσουμε την υλοποίηση ενός συστήματος ψάχνουμε να βρούμε αν κάτι παρόμοιο έχει ήδη υλοποιηθεί. Στην τεχνική αναφορά της εταιρίας Analog [\[31\]](#), γίνεται μια μελέτη για την ενίσχυση σήματος για ένα διαφορετικό ηλεκτροχημικό αισθητήρα, αλλά παρόμοιο αισθητήρα, από την εταιρία Alphasense [\[32\]](#). Σε συνδυασμό με την μελέτη του application note της Figaro [\[26\]](#), καταλήγουμε στο συμπέρασμα πως το κύκλωμά μας πρέπει να καλύπτει τις παρακάτω απαιτήσεις:

- Πρέπει να δέχεται ρεύμα αναφοράς τουλάχιστον 2 Volt(virtual ground)
- Η τροφοδότηση του αναλογικού κυκλώματος θα γίνει με 5 Volt μέσω του gpi0.
- Πρέπει να αποκόπτει υψηλές αρμονικές που παράγονται από ηλεκτρομαγνητικά πεδία.
- Πρέπει να μετατρέπει το ρεύμα από τον TGS5042 σε τάση προς ψηφιοποίηση.
- Πρέπει να εξασφαλίζει πως η έξοδος του σήματος δεν θα ξεπεράσει τα 3.6 Volt.

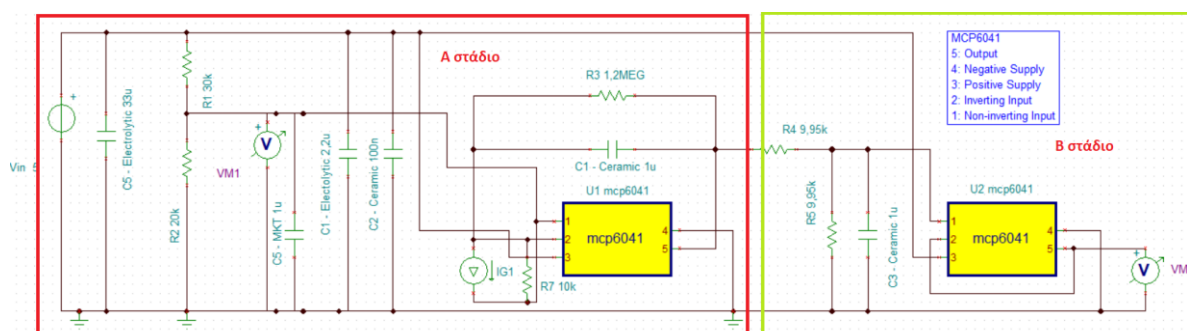
3.1.2 Ανάλυση κυκλώματος

Αρχικά, δημιουργήθηκε το αναλογικό κύκλωμα μελετώντας το application note από την Figaro και έγινε προσομοίωση κάνοντας χρήση του δωρεάν λογισμικού TINA από την Texas Instruments. Παρακάτω εμφανίζεται το ολικό κύκλωμα που δημιουργήθηκε. Εικόνα 16.



Εικόνα 16. Αναλογικό κύκλωμα για TGS5042

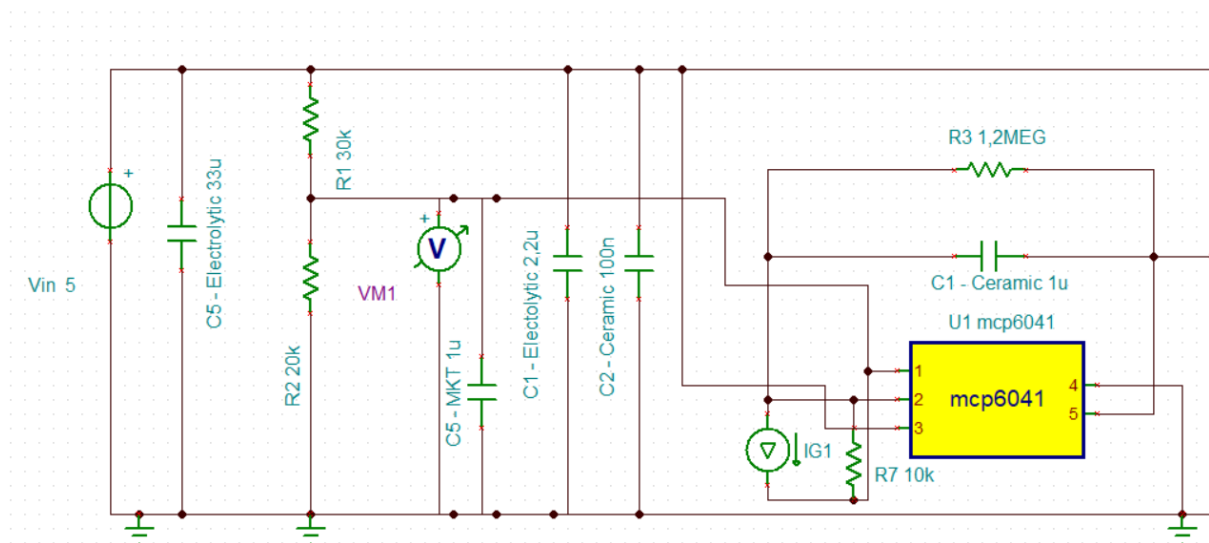
Το κύκλωμα φαίνεται σχετικά πολύπλοκο, επομένως θα το αναλύσουμε χωρίζοντας το σε 2 μέρη: το Α στάδιο και το Β στάδιο (εικόνα 17).



Εικόνα 17. Στάδια αναλογικού κυκλώματος

3.1.2.1 Ανάλυση σταδίου Α

Το πρώτο μέρος/στάδιο ασχολείται κυρίως με την βασική μετατροπή του ασθενές ρεύματος σε τάση. Ξεχωρίζουμε λοιπόν στο πρώτο μέρος, την τάση των 5 Volt τροφοδοσίας που έρχονται από το Raspberry PI Wireless Zero. Μέσω μιας διάταξης διαιρέτη τάσης (R1 και R2), παράγεται τάση αναφοράς 2 V (VM1). Η τάση αυτή εφαρμόζεται στην αναστρέφουσα είσοδο (non-inverting input, Pin 1) του τελεστικού ενισχυτή (Εικόνες 17 & 18). Με τον τρόπο αυτό δημιουργείται μια υποτυπώδης εικονική γείωση (**virtual ground**), η λειτουργία της οποίας αναλύεται εκτενέστερα στη συνέχεια.



Εικόνα 18. Πρώτο στάδιο ενίσχυσης TGS5042

Ο ηλεκτροχημικός αισθητήρας TGS5042 απαιτεί μια σταθερή δυναμική αναφοράς για να λειτουργήσει σωστά. Σε πολλές διατάξεις, αυτό επιτυγχάνεται με τη χρήση συμμετρικής τροφοδοσίας (π.χ. +5V και -5 V). Ωστόσο, επειδή το αναλογικό κύκλωμα τροφοδοτείται αποκλειστικά από τη απλή τροφοδοσία του Raspberry Pi Zero W, η δημιουργία αρνητικής τάσης θα απαιτούσε πολύπλοκα κυκλώματα. Για να ξεπεραστεί αυτός ο περιορισμός, σχεδιάστηκε ένας τοπικός διαιρέτης τάσης και χρησιμοποιήθηκε για δημιουργία virtual ground(εικονικής γείωσης), όπως εξηγήθηκε πιο πάνω.

Η σκοπιμότητα της εικονικής γείωσης στο κύκλωμά μας εξυπηρετεί δύο βασικούς σκοπούς:

- **Διαχείριση αμφίδρομου ρεύματος:** Οι ηλεκτροχημικοί αισθητήρες, ανάλογα με τις χημικές αντιδράσεις αναγωγής ή οξείδωσης και τις περιβαλλοντικές μεταβολές, ενδέχεται να αναστρέψουν τη φορά του ρεύματος τους. Αν η αναφορά μας ήταν η πραγματική, ο τελεστικός ενισχυτής δεν θα μπορούσε να αποδώσει αρνητική τάση εξόδου (καθώς δεν έχει αρνητική τροφοδοσία), με αποτέλεσμα το σήμα να «ψαλιδιζόταν». Μετατοπίζοντας τη γείωση στην τιμή 2 Volt(reference voltage) η έξοδος του τελεστικού ενισχυτή mcp6042 μπορεί να κυμανθεί τόσο πάνω όσο και κάτω από αυτήν την τιμή, επιτρέποντας στον ADC να διαβάσει όλο το εύρος του σήματος.
- **Συμβατότητα με Single-Supply Op-Amps:** Επιτρέπει στον τελεστικό ενισχυτή να λειτουργεί γραμμικά σε όλο το δυναμικό του εύρος, χρησιμοποιώντας αποκλειστικά τη θετική γραμμή τροφοδοσίας του Raspberry Pi.

Αξίζει να σημειωθεί πως η τιμή των 2 Volts για εικονική γείωση προτείνεται στο αντίστοιχο application note της Figaro [\[26\]](#). Συνεχίζουμε με την επεξήγηση των διάφορων ακίδων/pins του τελεστικού ενισχυτή.

Στο Pin3, το οποίο είναι η θετική τροφοδοσία του τελεστικού ενισχυτή, συνδέεται η τάση των 5 Volt, ενώ το Pin4 το οποίο είναι η αρνητική τροφοδοσία, γειώνεται. Με αυτόν τον τρόπο, ο τελεστικός ενισχυτής λειτουργεί σε μορφή rail-to-rail. Δηλαδή, η έξοδος του, θα έχει εύρος από 2 Volts μέχρι 5 Volts που είναι το τερματικό όριο της τροφοδοσίας. Αν για κάποιο λόγο ο TGS5042 παράξει μεγάλο ρεύμα, τότε η τάση εξόδου που θα δημιουργηθεί δεν θα μπορεί να είναι μεγαλύτερη από την τάση τροφοδοσίας.

Η μονάδα **IG1** είναι ο TGS5042 που προσομοιώνεται από μια πηγή ρεύματος (current source). Η μονάδα αυτή συνδέεται στο Pin 2, το οποίο είναι το λεγόμενο inverting input του τελεστικού ενισχυτή. Επίσης, η έξοδος του τελεστικού ενισχυτή συνδέεται στον **IG1** χρησιμοποιώντας μια κλασική διάταξη ενίσχυσης **RF = 1.2Meg Ohm** όπως και προτείνεται από την Figaro [\[26\]](#). Με αυτόν τον τρόπο ενισχύεται το σήμα από τον **IG1** και μετατρέπεται σε τάση.

Για να κατανοήσουμε τη λειτουργία του τελεστικού ενισχυτή, πρέπει να σημειωθεί ότι οι εισόδους δεν συνδέονται απευθείας με την έξοδο εσωτερικά. Αντίθετα, μέσω ενός βρόχου ανατροφοδότησης, ο ενισχυτής προσπαθεί διαρκώς να κρατά τις δύο εισόδους του στην ίδια τάση. Όταν το ρεύμα του αισθητήρα (**IG1**) μεταβάλλεται, η τάση στην έξοδο αλλάζει και ανατροφοδοτείται κατάλληλα για να διατηρήσει αυτή την ισορροπία. Με αυτόν τον τρόπο, η παραγωγή περισσότερου ρεύματος κατά την έκθεση του αισθητήρα σε CO μεταφράζεται απευθείας σε μια ανάλογα αυξημένη τάση σε Volt στην έξοδο.

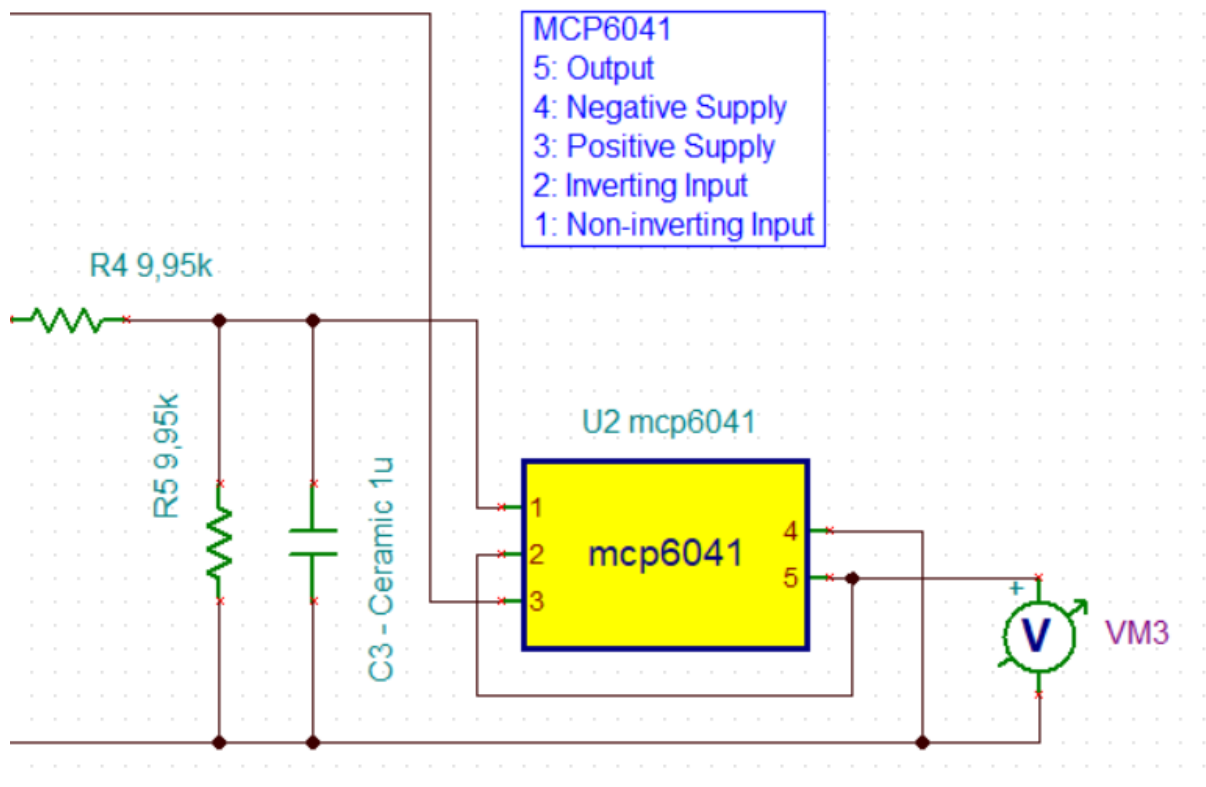
3.1.2.2 Ανάλυση σταδίου B

Στο τεχνικό φυλλάδιο (application note) [\[26\]](#) της Figaro, και συγκεκριμένα στην παράγραφο 2-6 “Electrical Noise Prevention”, επισημαίνεται η ευαισθησία της διάταξης στον θόρυβο:

“...Since sensor current is very small and amplification gain is large, the sensor is easily influenced by external electrical noise. As a result, it is necessary to implement measures to minimize electrical noise in the circuit pattern, power supply, etc. There are three options as

a countermeasure for electrical noise: 1) Use an electric noise filter 2) Use a voltage follower circuit 3) Use an RC circuit for power input, output, and amplification of the Op-Amp...”

Για τον λόγο αυτό, σχεδιάστηκε ένα δεύτερο στάδιο επεξεργασίας του σήματος, όπως απεικονίζεται στην Εικόνα 19. Σε αυτό το στάδιο ενσωματώνεται ένας δεύτερος τελεστικός ενισχυτής σε συνδεσμολογία ακόλουθου τάσης (**voltage follower** / απομονωτή), με σκοπό τη θωράκιση του κυκλώματος έναντι των εξωτερικών παρεμβολών.



Εικόνα 19. Δεύτερο Στάδιο Ενίσχυσης Σήματος TGS5042

Όπως απεικονίζεται στην παραπάνω εικόνα(εικόνα 19), η έξοδος **VM3** αποτελεί το σήμα που οδηγείται στον αναλογικό σε ψηφιακό μετατροπέα (ADC) ADS1115. Επιπλέον, βάσει των πρακτικών απόζευξης (decoupling) που αναλύονται στην τεχνική αναφορά [\[33\]](#) της Analog Devices, στην αρχή της γραμμής τροφοδοσίας του πρώτου σταδίου τοποθετήθηκε ένας ηλεκτρολυτικός πυκνωτής 33 μF . Ο σκοπός του είναι να λειτουργεί ως τοπική δεξαμενή φορτίου, καλύπτοντας τις στιγμιαίες απαιτήσεις του κυκλώματος και αποτρέποντας τη μεταφορά αυτών των μεταβολών μέσω της κύριας γραμμής τροφοδοσίας.

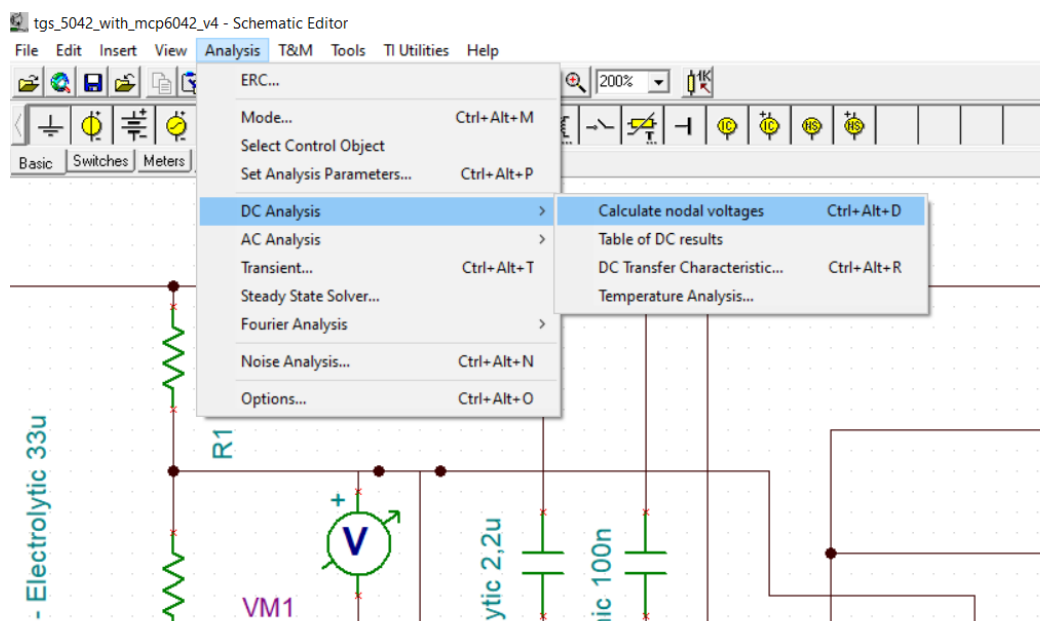
Στην ίδια τεχνική αναφορά τεκμηριώνεται η αναγκαιότητα χρήσης πυκνωτών μικρότερης χωρητικότητας ($0.1\ \mu\text{F}$, $1\ \mu\text{F}$ και $2.2\ \mu\text{F}$), οι οποίοι:

- Τοποθετούνται όσο το δυνατόν πιο κοντά στα pin τροφοδοσίας (VCC) των ολοκληρωμένων κυκλωμάτων.
- Λειτουργούν ως φίλτρα που βραχυκυκλώνουν τον θόρυβο υψηλών συχνοτήτων προς τη γείωση (GND), προστατεύοντας αυτά τα κυκλώματα.

Ακολουθώντας αυτές τις σχεδιαστικές οδηγίες, προστέθηκαν πυκνωτές ανάλογων τιμών πλησίον των ακροδεκτών τροφοδοσίας των τελεστικών ενισχυτών.

3.1.3 Διαιρέτης τάσης στο δεύτερο στάδιο - Προσομοίωση

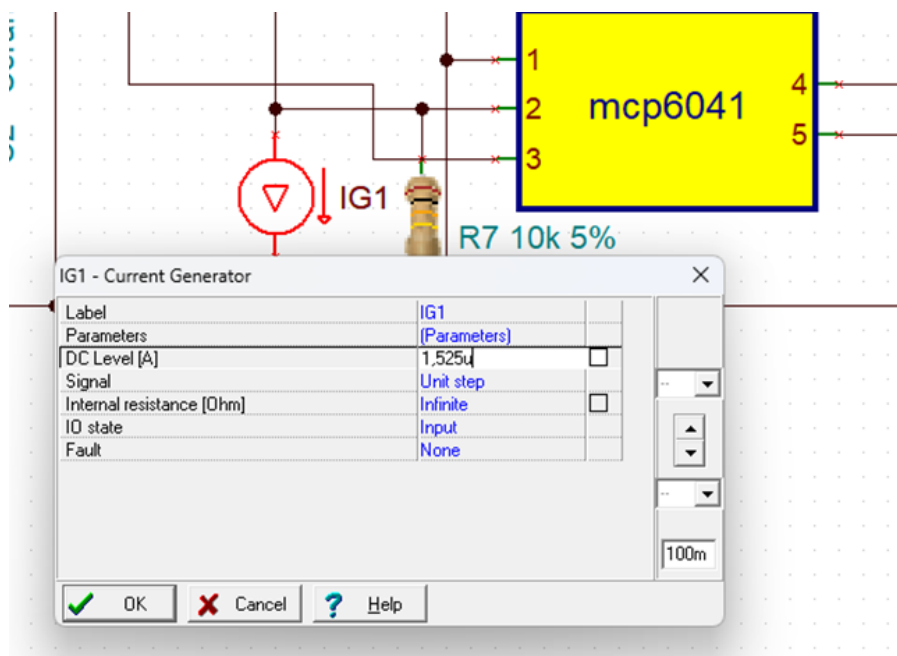
Μέσω του εργαλείου TINA, έγινε υπολογισμός των τάσεων στις εξόδους χρησιμοποιώντας το DC Analysis. (Εικόνα 20).



Εικόνα 20. Υπολογισμός τάσεων μέσω του DC ανάλυσης

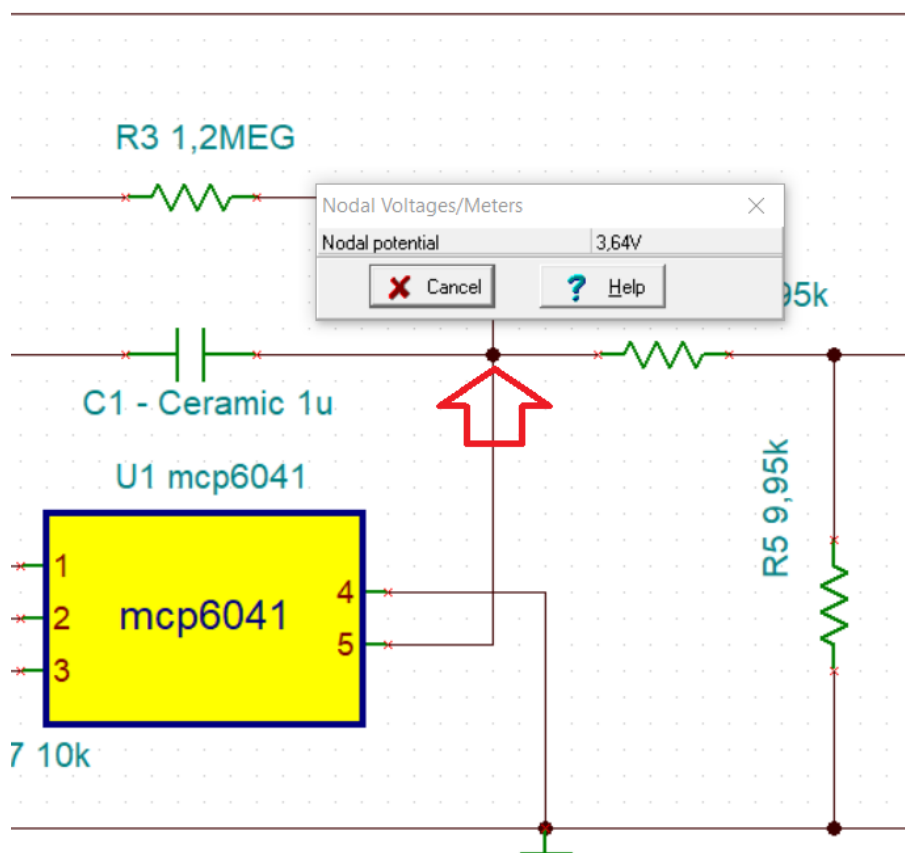
Η προσομοίωση έτρεξε θέτοντας το **IG1**, που είναι ένα current source, και που αναπαριστά τον TGS5042, σε $1.525\ \mu\text{A}$ (Εικόνα 21). Επιλέγουμε τον IG1, κάνουμε δεξί κλικ και θέτουμε DC level = $1.525\ \mu(1.525\ \mu\text{A})$. Να λάβουμε υπόψη πως ο αισθητήρας **TGS5042** που αγοράστηκε ήρθε με sensitivity: **$1.525\ \text{nA/ppm}$** . Επομένως για να κάνουμε σωστή

προσομοίωση και καθώς ο αισθητήρας είναι γραμμικός, θέσαμε το αριθμό 1.525u, που αναπαριστά την έξοδο του αισθητήρα σε έκθεση CO 1000 ppm. Το σύμβολο “u” αναπαριστά το πρόθεμα micro (μ).



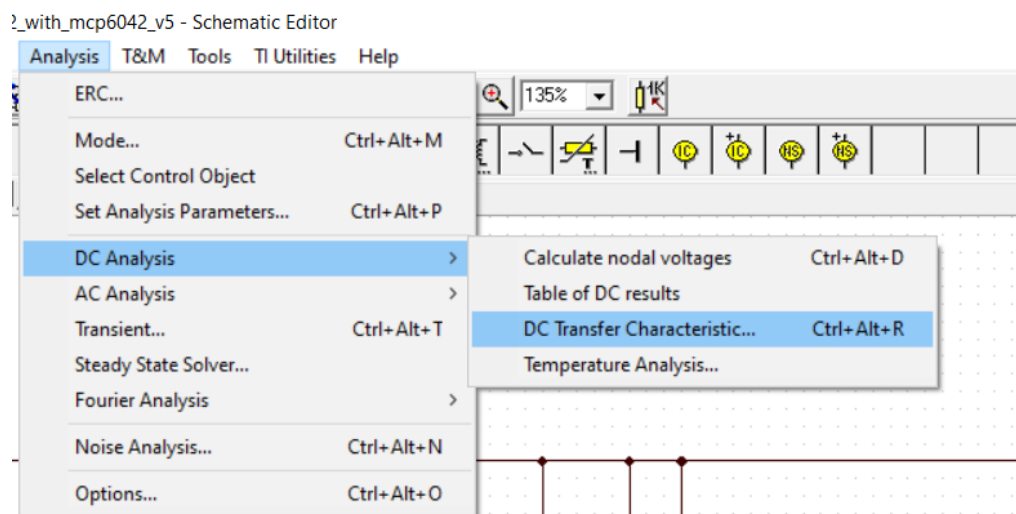
Εικόνα 21. Επιλογή ρεύματος για τον TGS5042

Σύμφωνα με την προσομοίωση, η τάση εξόδου στο πρώτο στάδιο αγγίζει τα 3,64Volts(εικόνα 22), ξεπερνώντας το όριο ασφαλούς λειτουργίας του ADS1115. Καθώς ο συγκεκριμένος ADC τροφοδοτείται με 3,3 Volts, οποιαδήποτε μεγαλύτερη τάση στις εισόδους του (A0–A3) ενδέχεται να καταστρέψει αυτό το breakout board.

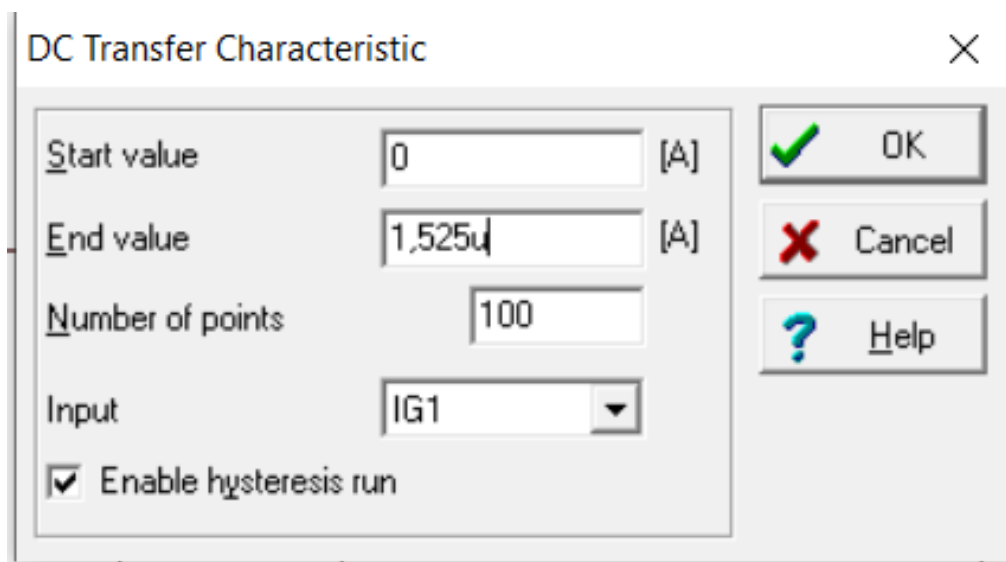


Εικόνα 22. Προσομοίωση τάσεως εξόδου TGS5042 πρώτου σταδίου

Για επίλυση του παραπάνω προβλήματος, στην είσοδο του σταδίου του τελεστικού ενισχυτή MCP6042, ενσωματώθηκε ένας διαιρέτης τάσης αποτελούμενος από δύο αντιστάσεις 9.95 kΩ. Η διάταξη αυτή μειώνει την τάση στο μισό για λόγους προστασίας του κυκλώματος, ενώ η τελική αποκατάσταση του σήματος γίνεται μέσω λογισμικού, πολλαπλασιάζοντας την τιμή κατάλληλα ώστε να υπολογιστεί σωστά η ποσότητα του μονοξειδίου του άνθρακα (CO). Τέλος, υπολογίστηκε μέσω προσομοίωσης η ολική απόκριση του συστήματος, μέσω της επιλογής *DC Analysis->DC Transfer Characteristic* και θέτοντας την μέγιστη τιμή 1.525μA, η οποία όπως γνωρίζουμε είναι το υποτιθέμενο ρεύμα που θα παραχθεί σε έκθεση 1000 CO. Στις εικόνες 23 και 24 εμφανίζεται αυτή η διαδικασία.

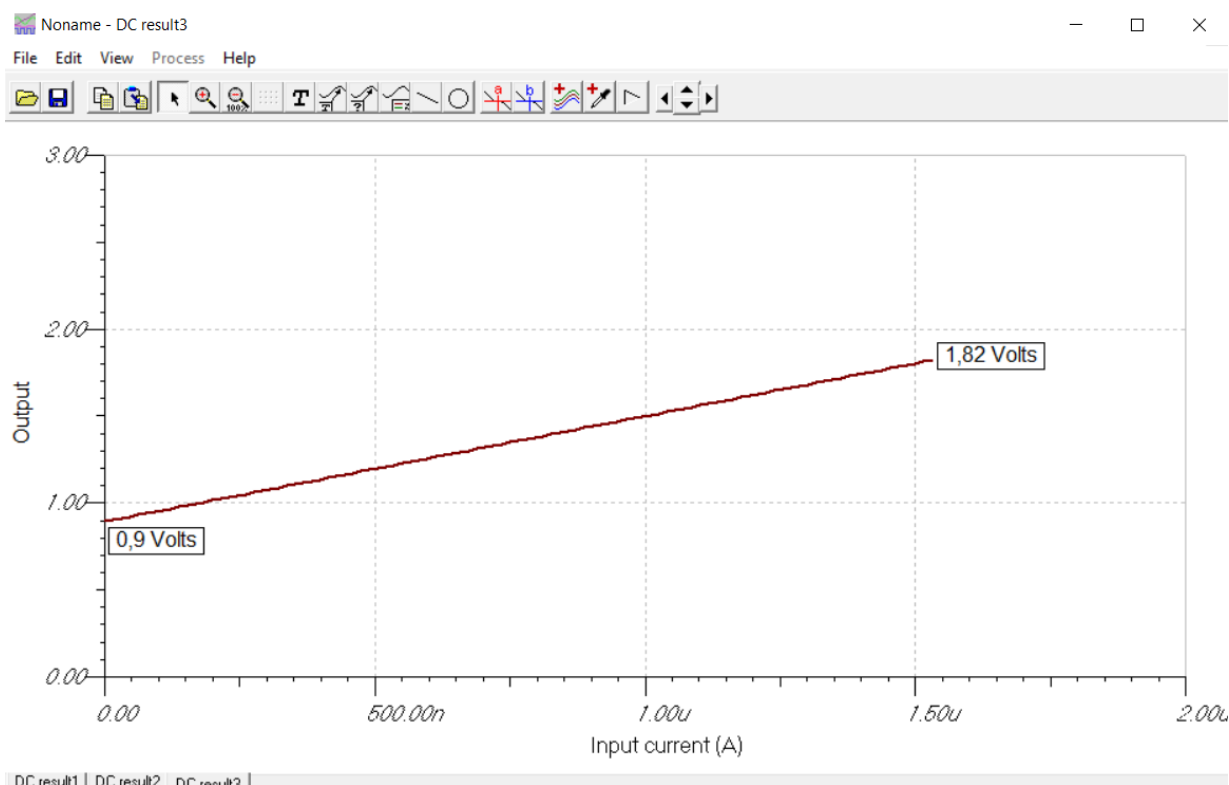


Εικόνα 23. Επιλογή DC Transfer Characteristic



Εικόνα 24. Ορισμός μέγιστου ρεύματος

Εφόσον τρέξουμε αυτή τη διαδικασία, λαμβάνουμε την παρακάτω γραφική παράσταση.



Εικόνα 25. Απόκριση κυκλώματος σε Volts

Παρατηρούμε, πως θεωρητικά μέσω της προσομοίωσης, το κύκλωμα παρουσιάζει τάση απόκρισης μεταξύ 0,9 volts και 1,82 volts. Επειδή, όμως έχουμε εισαγάγει στο κύκλωμα και ένα διαιρέτη τάσης $\frac{1}{2}$ τότε η πραγματική τάση που παράγει το κύκλωμα είναι:

$0,9 * 2 = 1,8 \text{ Volts}$ σε κατάσταση καθαρού αέρα.

$1,82 * 2 = 3,64 \text{ Volts}$ σε έκθεση CO 1000ppm.

3.1.4 Υπολογισμός CO σε ppm

Ο υπολογισμός της ποσότητας του μονοξειδίου του άνθρακα γίνεται βάση μια συγκεκριμένης εξίσωσης που βρίσκεται στο application note της Figaro στην ενότητα 6-3, Temperature Compensation [\[26\]](#):

$$C = \frac{(V_{out} - V_0)}{(at * Rf)}$$

Όπου:

- C: Η συγκέντρωση CO σε ppm
- V0: Η τάση εξόδου του αισθητήρα σε **καθαρό αέρα**. Αυτή ονομάζεται και τάση αναφοράς.
- Vout: Η τρέχουσα τάση εξόδου του αισθητήρα.
- at: Η αντισταθμισμένη ως προς τη θερμοκρασία ευαισθησία CO στους 20 βαθμούς Κελσίου.
- Rf: Η αντίσταση ανάδρασης (feedback resistor) του σταδίου ενίσχυσης (τελεστικού ενισχυτή). Η τιμή που χρησιμοποιούμε εδώ είναι 1.2 Meg Ohm.

Να σημειωθεί πως αρχικά το κύκλωμα πρέπει να λειτουργήσει σε καθαρό αέρα ώστε να μετρηθεί σωστά η V0. Αυτή η τιμή καταχωρείται στο πρόγραμμα σαν μεταβλητή ώστε να υπολογιστεί σωστά η τελική τιμή CO σε ppm.

3.1.4.1 Αντιστάθμιση Θερμοκρασίας (Temperature Compensation)

Η ονομαστική ευαισθησία του αισθητήρα CO στους 20 βαθμούς C υπολογίζεται από την ακόλουθη σχέση, σύμφωνα πάντα με το τεχνικό εγχειρίδιο της Figaro [\[26\]](#):

$$\alpha = at / CF$$

Όπου:

- **CF (compensation coefficient)**: Ο συντελεστής αντιστάθμισης θερμοκρασίας ανάλογα σε ποια θερμοκρασία βρισκόμαστε. Αυτό αντλείται από τον πίνακα του Παραρτήματος 2 στο τεχνικό εγχειρίδιο/application note αλλά το συμπεριλαμβάνουμε στην εικόνα 27 παρακάτω για ευκολία προς τον αναγνώστη.
- **at (CO Sensitivity)**: Η πραγματική (τρέχουσα) ευαισθησία του αισθητήρα CO στη θερμοκρασία μέτρησης.

Ο ηλεκτροχημικός αισθητήρας TS5042 που λάβαμε έχει έναν ειδικό αριθμό τυπωμένο επάνω του όπως φαίνεται και από την εικόνα 26. Ο αριθμός αυτός είναι το **1525** που αντιστοιχεί στο

1,525nA/ppm. Αυτή είναι εργοστασιακή ευαισθησία (α) του αισθητήρα στους σταθερούς 20 βαθμούς Κελσίου. Επομένως, για να υπολογίσουμε την τρέχουσα **at**, λύνουμε προς at:

$$at = \alpha * CF$$



Εικόνα 26. Ευαισθησία TGS5042 αισθητήρα

Συμπερασματικά, η αντισταθμισμένη ευαισθησία του αισθητήρα είναι

$$at = 1,525 * 1.0e-9 * CF$$

όπου το CF είναι η μεταβλητή που αντλούμε από την εικόνα 27 ανάλογα με την τρέχουσα θερμοκρασία. Αυτό σημαίνει πως κάθε φορά που ψηφιοποιούμε την έξοδο της τάσης από το αναλογικό κύκλωμα, πρέπει να παίρνουμε μετρήσεις και για την θερμοκρασία ώστε να μπορέσουμε να αντλήσουμε σωστά το CF.

APPLICATION NOTES FOR TGS5xxx SERIES

Appendix 2 - Temperature Compensation Coefficients for Residential Usage

Temp (°C)	CF (I/Io)		Temp (°C)	CF (I/Io)	
	TGS5042	TGS5141-P00		TGS5042	TGS5141-P00
-10	0.752	0.6745	30	1.060	1.0750
-9	0.761	0.6854	31	1.066	1.0825
-8	0.771	0.6963	32	1.071	1.0900
-7	0.780	0.7072	33	1.076	1.0975
-6	0.789	0.7181	34	1.080	1.1050
-5	0.799	0.7290	35	1.085	1.1125
-4	0.808	0.7399	36	1.089	1.1200
-3	0.817	0.7508	37	1.094	1.1275
-2	0.826	0.7617	38	1.098	1.1350
-1	0.835	0.7726	39	1.101	1.1425
0	0.844	0.7835	40	1.105	1.1500
1	0.852	0.7944	41	1.109	1.1575
2	0.861	0.8053	42	1.112	1.1650
3	0.870	0.8162	43	1.115	1.1725
4	0.878	0.8271	44	1.118	1.1800
5	0.887	0.8380	45	1.121	1.1875
6	0.895	0.8489	46	1.124	1.1950
7	0.903	0.8598	47	1.126	1.2025
8	0.911	0.8707	48	1.128	1.2100
9	0.919	0.8816	49	1.130	1.2175
10	0.927	0.8925	50	1.132	1.2250
11	0.935	0.9034	51	1.134	-
12	0.943	0.9143	52	1.135	-
13	0.950	0.9252	53	1.136	-
14	0.958	0.9361	54	1.137	-
15	0.965	0.9470	55	1.138	-
16	0.972	0.9579			
17	0.980	0.9688			
18	0.987	0.9797			
19	0.994	0.9906			
20	1.000	1.0000			
21	1.007	1.0075			
22	1.013	1.0150			
23	1.020	1.0225			
24	1.026	1.0300			
25	1.032	1.0375			
26	1.038	1.0450			
27	1.044	1.0525			
28	1.050	1.0600			
29	1.055	1.0675			

Εικόνα 27. Πίνακας συντελεστής αντιστάθμισης θερμοκρασίας

3.1.5 Διάγραμμα διασύνδεσης υλικού

Στην εικόνα 28, παρατίθεται το πλήρες διάγραμμα των 40 ακροδεκτών (GPIO Header) του Raspberry Pi Zero W, όπως αυτό τεκμηριώνεται από την επίσημη πλατφόρμα υλικού pinout.xyz [36]. Η παράθεση του συγκεκριμένου διαγράμματος κρίνεται απαραίτητη για την οπτικοποίηση των pins που θα χρησιμοποιηθούν. Συγκεκριμένα χρησιμοποιήθηκαν τα pins:

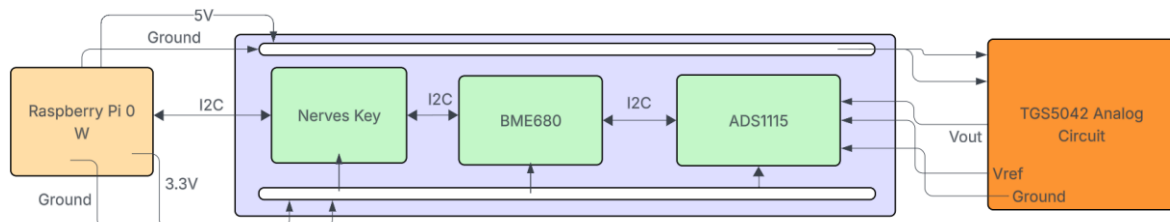
- 1 (τροφοδοσία 3.3V)
- 3 (I2C data)
- 5 (I2C clock)
- 9 (γείωση)
- 2 (5 V)
- 6 (γείωση)

3v3 Power	1	2	5v Power
GPIO 2 (I2C1 SDA)	3	4	5v Power
GPIO 3 (I2C1 SCL)	5	6	Ground
GPIO 4 (GPCLK0)	7	8	GPIO 14 (UART0 TX)
Ground	9	10	GPIO 15 (UART0 RX)
GPIO 17	11	12	GPIO 18 (PCM CLK)
GPIO 27	13	14	Ground
GPIO 22	15	16	GPIO 23
3v3 Power	17	18	GPIO 24
GPIO 10 (SPI0 MOSI)	19	20	Ground
GPIO 9 (SPI0 MISO)	21	22	GPIO 25
GPIO 11 (SPI0 SCLK)	23	24	GPIO 8 (SPI0 CE0)
Ground	25	26	GPIO 7 (SPI0 CE1)
GPIO 0 (EEPROM SDA)	27	28	GPIO 1 (EEPROM SCL)
GPIO 5	29	30	Ground
GPIO 6	31	32	GPIO 12 (PWM0)
GPIO 13 (PWM1)	33	34	Ground
GPIO 19 (PCM FS)	35	36	GPIO 16
GPIO 26	37	38	GPIO 20 (PCM DIN)
Ground	39	40	GPIO 21 (PCM DOUT)

Εικόνα 28. Pinout του Raspberry Pi Zero Wireless

Το γενικό block διάγραμμα της κεντρικής διασύνδεσης παρουσιάζεται στην εικόνα 29. Σε αυτό φαίνεται η κοινή σύνδεση των breakout boards (NervesKey, BME680 και ADS1115)

που έχουν τοποθετηθεί πάνω σε ένα κλασσικό breadboard με το Raspberry Pi Zero W μέσω του σειριακού διαύλου I2C. Επιπλέον, παρουσιάζεται και η αναλογική σύνδεση του κυκλώματος TGS5042 με τον ADC (ADS1115), αναδεικνύοντας τις γραμμές της γείωσης, της τάσης αναφοράς (V_{ref}) και της εξόδου σήματος (V_{out}).



Εικόνα 29. Block διάγραμμα ολικής διασύνδεσης με breadboard

3.1.6 Λίστα υλικών

Τα περισσότερα εξαρτήματα αγοράστηκαν μέσω της εταιρίας Mouser Electronics όπως εμφανίζονται και στον πίνακα παρακάτω. Αυτά τα υλικά αφορούσαν τα breakout boards (NervesKey και Bme680 θερμοκρασίας) καθώς και αντιστάσεις και πυκνωτές.

Πίνακας 3. Εξαρτήματα Mouser για την κατασκευή του κυκλώματος

Mouser No	Description	Qt	Price
474-DEV-18077	ATECC608A (Qwiic)	3	4,34 €
MCP6042-I/MS-JNZ	Operational Amplifiers - Op Amps LOW DRIFT BIPOLAR OP AMP	1	2 €
603-MFR-25FBF52-1M2	Metal Film Resistors - Through Hole 1.2M OHM 1/4W 1%	1	0,086 €
603-MFR-25FBF52-10K	Metal Film Resistors - Through Hole 10K OHM 1/4W 1%	1	0,086 €
603-MFR-25FBF52-30K	Metal Film Resistors - Through Hole 30K OHM 1/4W 1%	2	0,086 €
594-MAL203035339E3	Aluminium Electrolytic Capacitors - Axial Leaded 85C	1	1,57 €
80-C315C104M5U	Multilayer Ceramic Capacitors MLCC - Leaded 50V 0.1uF Z5U 20% LS=2.54mm	1	0,275 €

80- C322C105K5R9170 -R	Multilayer Ceramic Capacitors MLCC - Leaded 50V 1uF X7R 10% LS=5.08mm	1	0,972 €
660- MF1/4DC2002F	Metal Film Resistors - Through Hole 20K 1% 100PPM	1	0,161 €
485-3660	Multiple Function Sensor Development Tools Adafruit BME680 - T, H, P & G Sensor	1	16,30 €
653-B3F-1152	Tactile Switches 6.5MM Tactile Switch 150GF 7.3MM	1	0,163 €
603-MFR-25FBF52- 330R	Metal Film Resistors - Through Hole 330 OHM 1/4W 1%	1	0,086 €

Τα έξοδα μεταφοράς από mouser ήταν 20 ευρώ.

Σύνολο: $32,20 + 20 = 55,20$ €

Επίσης, τα παρακάτω επιπλέον υλικά αγοράστηκαν από διαφορετικούς πωλητές καθώς δεν ήταν δυνατή η αγορά τους από την mouser. Ο αισθητήρας TGS5042 αγοράστηκε από εταιρία στην Ιταλία καθώς στην Ελλάδα ήθελαν μόνο χονδρική πώληση.

Πίνακας 4 Λίστα εξαρτημάτων από διαφορετικές πωλητές

Αντικείμενο	Εταιρία	Κόστος(ευρώ)	Κόστος Μεταφοράς(ευρώ)
TGS5042-A00	Tecnosens SPA (Ιταλία)	17	20
ADS1115	Nettop.gr(Ελλάδα)	5	3
Επιπλέον πυκνωτές(0,1μF, 1μF, 48μF) και αντιστάσεις (10K, 1K)	Τοπική επιχείρηση	8	0
Raspberry pi zero w	Scrooge.gr	18	3

Σύνολο: 37 ευρώ

Επομένως, το τελικό κόστος περίπου ήταν $37 + 55,20 = 92,20$ ευρώ.

3.2 Σχεδιασμός Λογισμικού και διασυνδέσεις

Στην ενότητα αυτή αναλύονται οι απαιτήσεις του λογισμικού που θα τρέξει στο raspberry pi zero wireless καθώς και η αρχιτεκτονική σχεδίασης των επιμέρους εφαρμογών και η διασύνδεση μεταξύ τους.

3.2.1 Απαιτήσεις συστήματος

Το σύστημα πρέπει να ικανοποιεί συγκεκριμένες απαιτήσεις, όπως δειγματοληψία ανά τακτά χρονικά διαστήματα, αποθήκευση δεδομένων για περαιτέρω ανάλυση και αποστολή δεδομένων. Συγκεκριμένα πρέπει να υλοποιεί:

1. **Λήψη δεδομένων.** Το σύστημα πρέπει να εκτελεί περιοδική δειγματοληψία ανά 15 δευτερόλεπτα. Απαιτείται η λήψη αναλογικών μετρήσεων από τον αισθητήρα μονοξειδίου του άνθρακα (CO) μέσω του ADC ADS1115 (Vout), της τάσης αναφοράς (Vref) και θερμοκρασίας από τον αισθητήρα BME680.
2. **Ψηφιακό φιλτράρισμα, βαθμονόμηση και υπολογισμός CO:** Το λογισμικό πρέπει να επεξεργάζεται τα δεδομένα του ADC εφαρμόζοντας ψηφιακά φίλτρα για την απόρριψη θορύβου (median φίλτρο). Επίσης, το σύστημα πρέπει χρησιμοποιεί μια μέθοδο μέτρησης βάσει αναλογίας (ratiometric approach), καθώς η τάση τροφοδοσίας του αισθητήρα και η τάση αναφοράς του ενισχυτή προέρχονται από την ίδια πηγή. Αυτή η προσέγγιση μέτρησης εξαλείφει τυχόν διακυμάνσεις της τάσης τροφοδοσίας σε σχέση με την τάση αναφοράς, υπολογίζοντας σωστά κάθε φορά την τιμή CO σε ppm.
3. **Τοπική αποθήκευση ιστορικότητας:** Το σύστημα πρέπει να διατηρεί στη μνήμη RAM δεδομένα των τελευταίων 7 ημερών, επιτρέποντας τη γρήγορη ανάγνωση ιστορικών σειρών.
4. **Διασύνδεση εφαρμογής Web:** Απαιτείται η παροχή μιας τοπικής διαδικτυακής διεπαφής σε πραγματικό χρόνο (Real-Time Web UI) για την οπτικοποίηση των τρεχουσών μετρήσεων και των ιστορικών γραφημάτων. Η εφαρμογή αυτή πρέπει τρέχει πάνω στο Raspberry PI Zero Wireless.

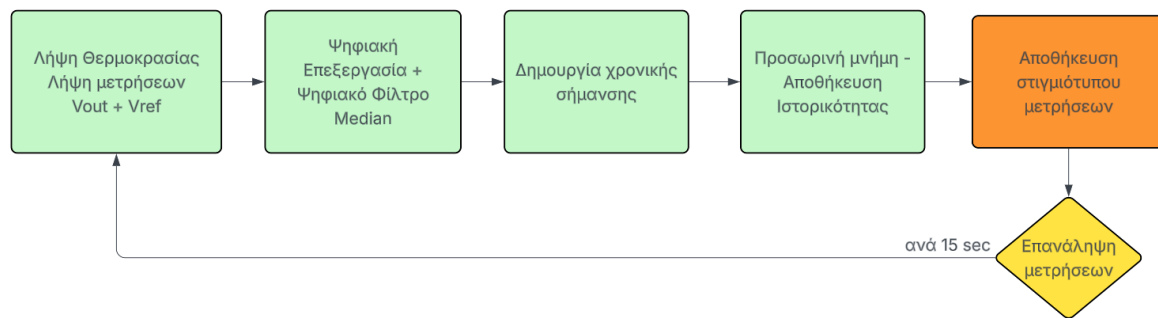
5. **Αποστολή δεδομένων τηλεμετρίας:** Το σύστημα πρέπει να μετατρέπει τις μετρήσεις σε μορφή JSON και να τις αποστέλλει ασύγχρονα στην πλατφόρμα ThingsBoard Cloud μέσω του πρωτοκόλλου MQTT ανά 15 δευτερόλεπτα.
6. **Εύκολη διασύνδεση:** Το σύστημα πρέπει να παρέχει εύκολη διασύνδεση με τον τοπικό οικιακό router παρέχοντας διεπαφή διασύνδεσης μέσω web. Σε περίπτωση επανεκκίνησης, οι ρυθμίσεις αυτές δεν πρέπει να χάνονται.

3.2.2 Εφαρμογή Δειγματοληψίας

Η εφαρμογή δειγματοληψίας περιλαμβάνει πολλές άλλες επιμέρους διεργασίες που επεξεργάζονται και αποθηκεύουν δεδομένα. Επίσης, δίνει πρόσβαση στις πιο πρόσφατες δειγματοληψίες και σε άλλες εφαρμογές που θα αναλυθούν αργότερα. Η δειγματοληψία γίνεται ανά 15 δευτερόλεπτα. Αυτό αποφασίστηκε, κυρίως, έπειτα από ανάλυση των περιβαλλοντολογικών δεδομένων που δεν παρουσιάζουν ακραίες μεταβολές μέσα στο χώρο της οικιακής εστίας. Π.χ, ούτε η θερμοκρασία ούτε και το μονοξείδιο του άνθρακα παρουσιάζουν διακυμάνσεις μέσα σε 1 ή 2 δευτερόλεπτα. Συγκεκριμένα, από το επίσημο datasheet του κατασκευαστή [\[26\]](#), ο τυπικός χρόνος απόκρισης T90 του TGS5042 (εικόνα 8) ανέρχεται γύρω στα 60 sec (ο χρόνος που απαιτείται για να αποτυπώσει ο αισθητήρας το 90% μιας ακαριαίας μεταβολής στη συγκέντρωση του αερίου).

Με βάση αυτή την προδιαγραφή, η επιλογή του μεσοδιαστήματος δειγματοληψίας στα 15 δευτερόλεπτα κρίνεται ως η βέλτιστη επειδή επιτρέπει στο λογισμικό να καταγράψει με ακρίβεια τη μεταβατική καμπύλη απόκρισης του αισθητήρα (λαμβάνοντας 4 δείγματα εντός του παραθύρου των 60 δευτερολέπτων), αποφεύγοντας ταυτόχρονα την άσκοπη υπερδειγματοληψία.

Παρακάτω το διάγραμμα ροής περιγράφει αυτή την εφαρμογή δειγματοληψίας:



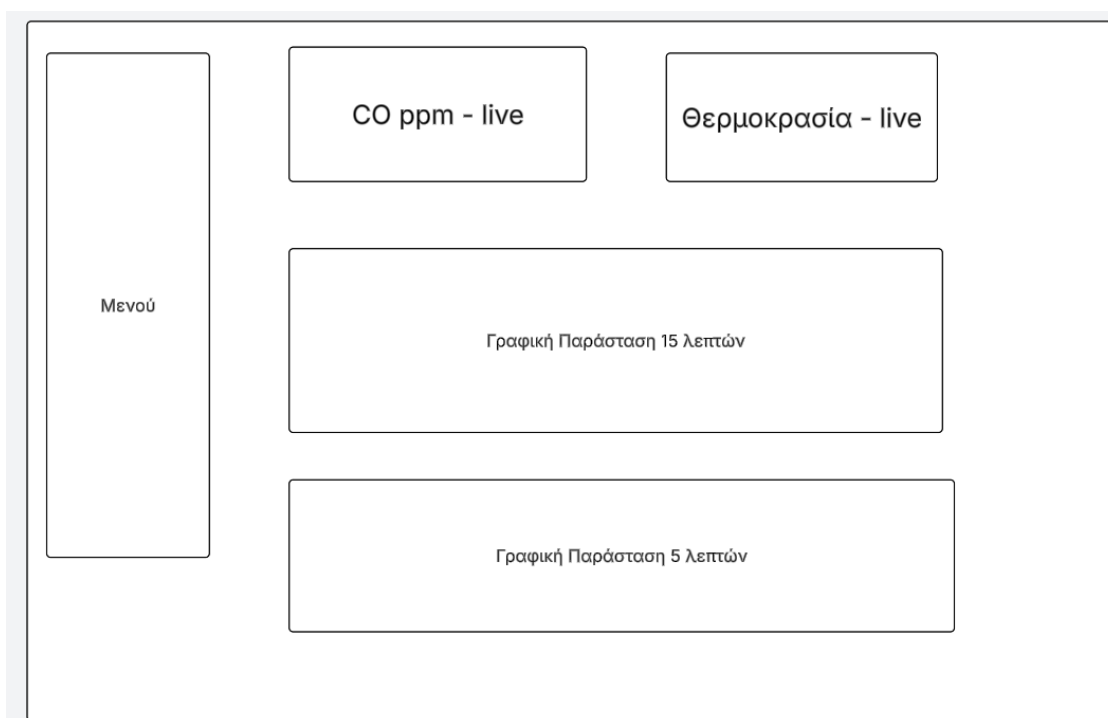
Εικόνα 30. Διάγραμμα ροής δειγματοληψίας

3.2.3 Εφαρμογή εμφάνισης πληροφοριών σε πραγματικό χρόνο.

Για την τοπική οπτικοποίηση των δεδομένων, σχεδιάστηκε μια κεντρική διαδικτυακή διεπαφή με γνώμονα την αμεσότητα της πληροφορίας. Με τη χρήση του framework Phoenix LiveView που υποστηρίζεται από το Elixir Nerves, επιτυγχάνεται αμφίδρομη επικοινωνία πραγματικού χρόνου μέσω WebSockets, επιτρέποντας την αυτόματη ανανέωση της οθόνης κάθε φορά που η εφαρμογή δειγματοληψίας παράγει ένα νέο δείγμα (ανά 15 δευτερόλεπτα).

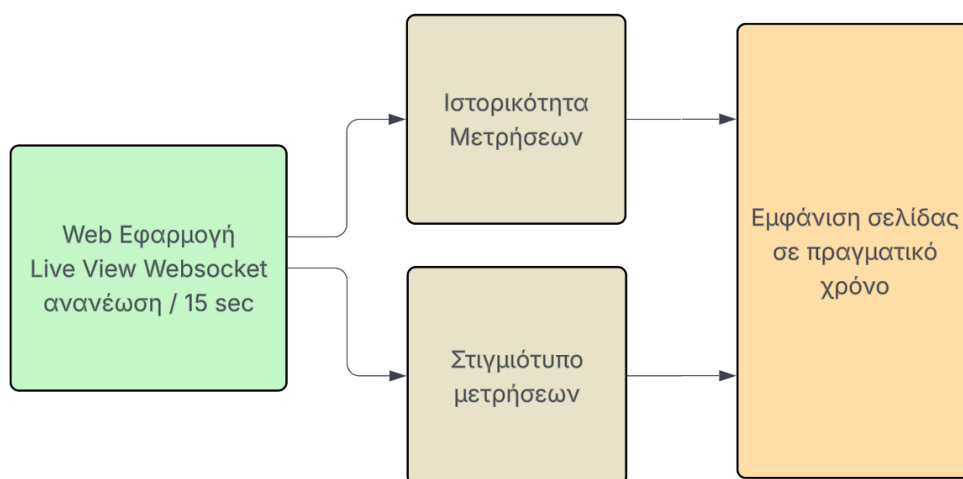
Η εφαρμογή αυτή τρέχει σαν ξεχωριστή οντότητα μέσα από το elixir nerves πάνω στο Raspberry PI Zero Wireless. Έχει πρόσβαση στην ιστορικότητα των μετρήσεων καθώς και στο τελευταίο στιγμιότυπο μετρήσεων που δημιουργούνται από την εφαρμογή δειγματοληψίας.

Η αρχιτεκτονική διάταξη της κεντρικής σελίδας αποτυπώνεται στο παρακάτω wireframe:



Εικόνα 31. Διάταξη διαδικτυακής επαφής για πληροφόρηση CO/Θερμοκρασίας

Για την πλήρη κατανόηση της αλληλεπίδρασης μεταξύ της web εφαρμογής και των αποθηκευμένων δεδομένων σχεδιάστηκε το παρακάτω block diagram:



Εικόνα 32. Λειτουργία εφαρμογής web

3.2.4 Εφαρμογή αποστολής δεδομένων στο ThingsBoard

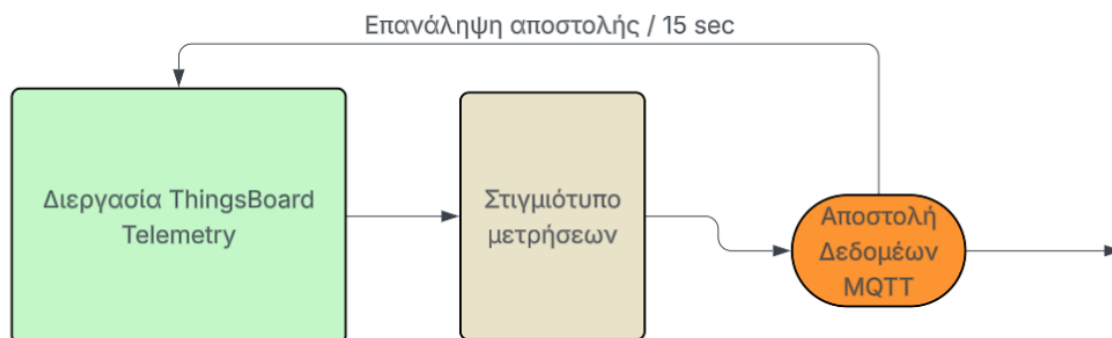
Η μεταφορά των μετρήσεων σε μια IoT πλατφόρμα κρίθηκε απαραίτητη για δύο λόγους:

- **Απομακρυσμένη πρόσβαση:** Το τοπικό Web UI λειτουργεί μόνο εντός του οικιακού δικτύου Wi-Fi ενώ η πλατφόρμα επιτρέπει την παρακολούθηση της συσκευής από οπουδήποτε.
- **Μόνιμη αποθήκευση δεδομένων:** Λόγω περιορισμών της RAM το Raspberry Pi Zero Wireless κρατάει τοπικά δεδομένα μόνο για 7 ημέρες. Η IoT πλατφόρμα λειτουργεί ως μόνιμη αποθήκη ιστορικού.

Για την υλοποίηση των παραπάνω, επιλέχθηκε η έτοιμη πλατφόρμα ThingsBoard EU (eu.thingsboard.cloud), καθώς καλύπτει τις εξής προδιαγραφές:

- **Υποστήριξη MQTT:** Επιτρέπει την εύκολη λήψη των δεδομένων ανά 15 δευτερόλεπτα.
- **Έτοιμα Widgets:** Παρέχει έτοιμα γραφήματα και μετρητές για τη γρήγορη δημιουργία γραφημάτων

Η αποστολή των δεδομένων στο ThingsBoard δεν γίνεται από κάποιο εξωτερικό πρόγραμμα, αλλά εκτελείται ως αυτόνομη, εσωτερική υπηρεσία μέσα στην ίδια την κύρια εφαρμογή δειγματοληψίας. Κάθε 15 δευτερόλεπτα διαβάζει από το στιγμιότυπο μετρήσεων και στέλνει τις μετρήσεις στο ThingsBoard. Το παρακάτω block διάγραμμα παρουσιάζει την διεργασία αυτή.



Εικόνα 33. Αποστολή δεδομένων στην IoT πλατφόρμα ThingsBoard

3.3 Δομή ολικής εφαρμογής τύπου Elixir Nerves

Όπως εξηγήθηκε και στο κεφάλαιο 2, η αρχιτεκτονική του λογισμικού στην πλατφόρμα Elixir Nerves διαφοροποιείται ριζικά από τα παραδοσιακά ενσωματωμένα συστήματα, καθώς δεν βασίζεται σε ένα ενιαίο, μονολιθικό αρχείο κώδικα. Αντίθετα, δομείται γύρω από το πρότυπο του οικοσυστήματος OTP (Open Telecom Platform), όπου το λογισμικό οργανώνεται σε διακριτές, αυτόνομες εφαρμογές (OTP Applications) που εκτελούνται υπό την εποπτεία της εικονικής μηχανής BEAM της Erlang.

3.3.1 Αρχιτεκτονική Τριών Εφαρμογών

Κάνοντας χρήση της διαδικασίας του διαχωρισμού ευθυνών [37], το λογισμικό του συστήματος σχεδιάστηκε ως ένα σύνολο τριών αυτόνομων εφαρμογών τύπου OTP, οι οποίες συνυπάρχουν ως ισότιμοι κόμβοι εντός της εικονικής μηχανής BEAM. Η οργάνωση αυτή υλοποιείται μέσω της δομής Mix Poncho Project [38], η οποία επιτρέπει την ανεξάρτητη ανάπτυξη και επαλήθευση κάθε εφαρμογής, ενώ παράλληλα τις ενοποιεί σε ένα μοναδικό firmware image για το τελικό γράψιμο στην συσκευή/SDCARD.

Οι τρεις εφαρμογές [39] που απαρτίζουν το σύστημα ορίζονται ως εξής:

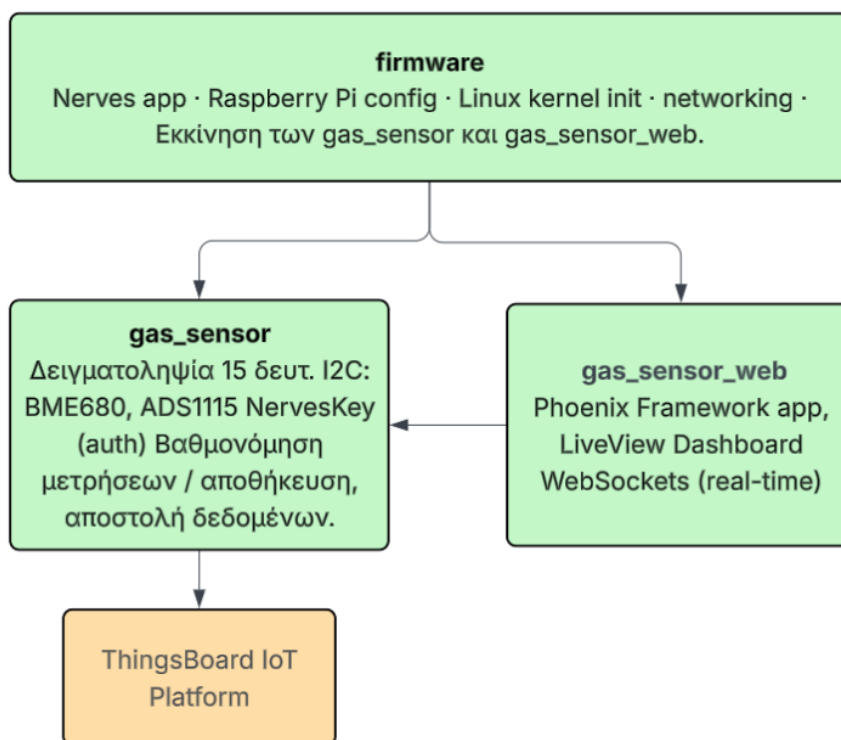
- **Εφαρμογή firmware:** Αποτελεί την αποκλειστική εφαρμογή Elixir Nerves του συστήματος. Αναλαμβάνει τη διαχείριση των ρυθμίσεων του Raspberry Pi Wireless

Zero, την αρχικοποίηση του μινιμαλιστικού πυρήνα Linux, τις ειδικές παραμέτρους του υλικού, υποσύστημα δικτύωσης(wifi κτλ) καθώς και την δυνατότητα διασύνδεσης με το οικιακό router μέσω εύκολης διεπαφής.

- **Εφαρμογή πυρήνα (gas_sensor):** Μια αμιγής εφαρμογή Elixir/OTP, υπεύθυνη για την γενική βασική λογική της εφαρμογής. Ελέγχει τον βρόχο δειγματοληψίας περιόδου 15 δευτερολέπτων, την επικοινωνία μέσω του διαύλου I2C με τα ολοκληρωμένα κυκλώματα των αισθητήρων (BME680, ADS1115, NervesKey), την εφαρμογή των αλγορίθμων βαθμονόμησης και την προσωρινή τήρηση των μετρήσεων στη μνήμη RAM και τέλος την αποστολή δεδομένων στην πλατφόρμα ThingsBoard.
- **Εφαρμογή διεπαφής (gas_sensor_web):** Μια αυτόνομη εφαρμογή Phoenix Framework, αποκλειστικού σκοπού, η οποία αναλαμβάνει την οπτικοποίηση των δεδομένων σε πραγματικό χρόνο. Ενημερώνει την σελίδα και τα ιστορικά γραφήματα (5 λεπτών, 15 λεπτών, 2 ωρών) μέσω αμφίδρομων καναλιών WebSockets, κάνοντας χρήση του μηχανισμού Phoenix LiveView.

Κατά την εκκίνηση της συσκευής, η εφαρμογή **firmware** ενεργεί ως ενορχηστρωτής, εκκινώντας παράλληλα τα δέντρα εποπτείας (Supervision Trees) και των τριών εφαρμογών. Το αρχιτεκτονικό αυτό σχήμα επιτρέπει στην **gas_sensor_web** να ανακτά τις μετρήσεις απευθείας από τον **gas_sensor** μέσω τοπικών κλήσεων μνήμης χωρίς να προβαίνει σε μετατροπές δεδομένων κάνοντας απλά χρήση τις μεθόδους επικοινωνίας που διαθέτει η κάθε εφαρμογή OTP.

Η ιεραρχική σχέση, οι επιμέρους αρμοδιότητες καθώς και η ροή επικοινωνίας μεταξύ των τριών OTP εφαρμογών αποτυπώνονται σχηματικά στο διάγραμμα της εικόνας 34.



Εικόνα 34. Δομή της τελικής εφαρμογής Elixir Nerves

4.0 Υλοποίηση συστήματος

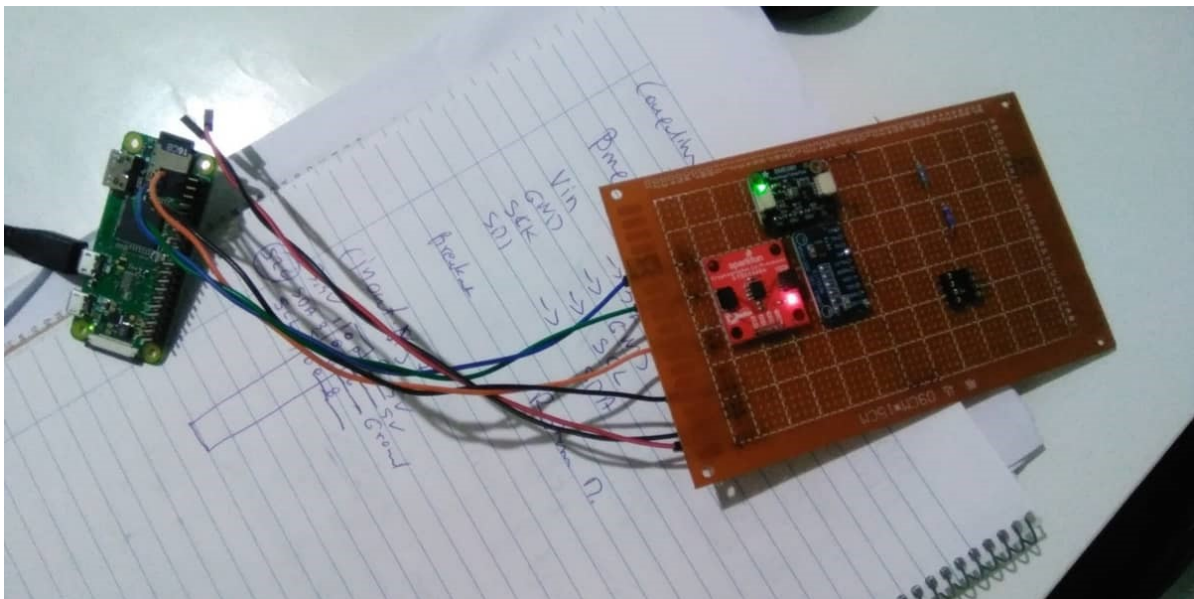
Στο κεφάλαιο αυτό θα ασχοληθούμε με τις υλοποιήσεις που αφορούν πρώτα το αναλογικό/ψηφιακό κομμάτι του συστήματος – δημιουργία κυκλώματος σε perfboard – καθώς και την φυσική διάταξη της διασύνδεσης του αναλογικού τμήματος με το breadboard και το Raspberry PI Zero Wireless. Στην συνέχεια θα αναλύσουμε την υλοποίηση που αφορά το τμήμα του λογισμικού Elixir Nerves, το τρόπο που οργανώθηκε καθώς και την λειτουργία των αρθρωμάτων.

4.1 Υλοποίηση αναλογικού κυκλώματος

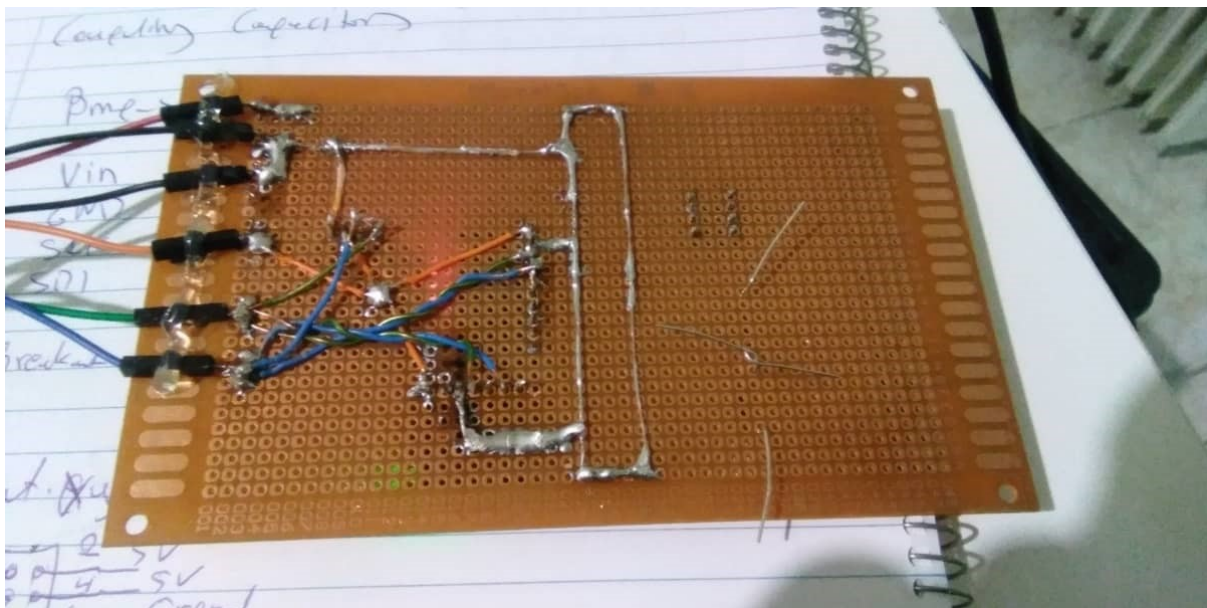
Το αναλογικό κύκλωμα υλοποιήθηκε σύμφωνα με την μελέτη και το σχεδιασμό στην αρχή του προηγούμενου κεφαλαίου. Η υλοποίηση έγινε πάνω σε ένα perfboard καθώς πολλά τμήματα έπρεπε να συγκολληθούν. Αρχικά, η παραγγελία για όλα τα εξαρτήματα έγινε μέσω

της Mouser Electronics στις ΗΠΑ εκτός του ηλεκτροχημικού αισθητήρα TGS5042 που αγοράστηκε από online κατάστημα στην Ιταλία.

Στα πρώτα στάδια ανάπτυξης της πλακέτας, πραγματοποιήθηκε προσπάθεια σχεδίασης ολόκληρου του συστήματος (ψηφιακών breakouts και αναλογικού σταδίου) πάνω σε μία κοινή διάτρητη πλακέτα (perfboard) (Εικόνες 35 & 36). Στην πορεία, για λόγους βέλτιστης οργάνωσης και ευκολίας στην τροποποίηση των συνδέσεων, κρίθηκε προτιμότερο να διαχωριστούν τα υποσυστήματα. Έτσι, οι πλακέτες επέκτασης τοποθετήθηκαν σε αναπτυξιακή πλακέτα δοκιμών (breadboard), ενώ η διάτρητη πλακέτα χρησιμοποιήθηκε αποκλειστικά για τη σταθερή μόνωση και λειτουργία του αναλογικού κυκλώματος προσαρμογής του αισθητήρα.



Εικόνα 35 Breakout Boards πάνω σε perfboard

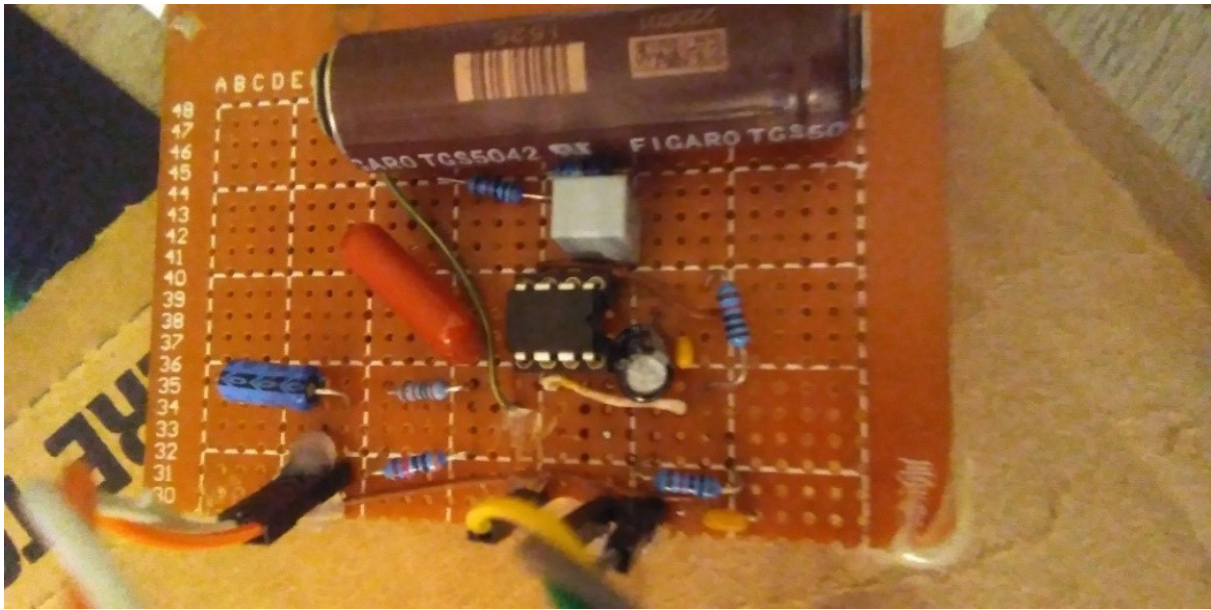


Εικόνα 36. Πίσω όψη του perfboard

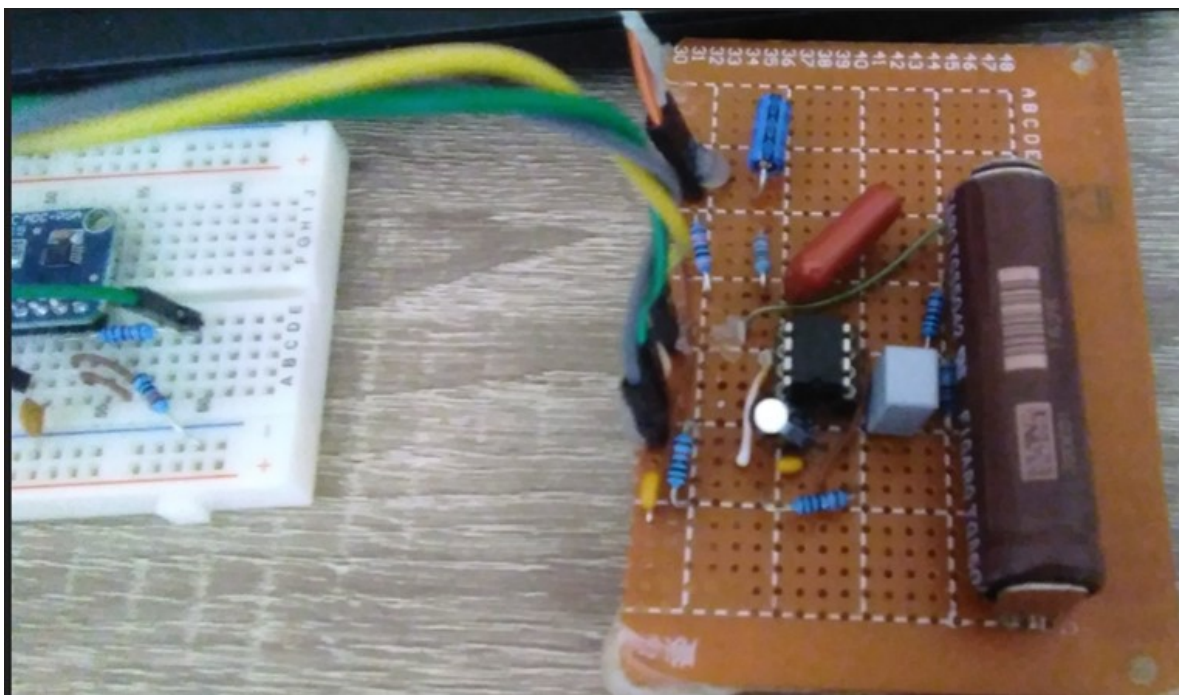
Πιο συγκεκριμένα, για την ελαχιστοποίηση του ηλεκτρομαγνητικού θορύβου και τη διασφάλιση της ακρίβειας των μετρήσεων, εφαρμόστηκε η τεχνική του φυσικού διαχωρισμού του κυκλώματος. Το ευαίσθητο αναλογικό στάδιο (αισθητήρας Figaro και ενισχυτής MCP6042) υλοποιήθηκε σε διάτρητη πλακέτα (perfboard) με μόνιμες κολλήσεις, ενώ το ψηφιακό τμήμα (Raspberry Pi και breakout boards) παρέμεινε σε πλακέτα δοκιμών (breadboard).

Αυτή η προσέγγιση κρίθηκε απαραίτητη καθώς οι ψηφιακές μεταγωγές του μικροεπεξεργαστή καθώς και διασύνδεση wifi προκαλούν παράσιτα. Οι παρασιτικές χωρητικότητες του breadboard θα μπορούσαν εύκολα να μεταφέρουν αυτόν τον θόρυβο στο κύκλωμα του αισθητήρα TGS5042. Η υλοποίηση του αναλογικού τμήματος σε perfboard εξασφάλισε σταθερές συνδέσεις χαμηλής αντίστασης, απομονώνοντας το σήμα από τις ψηφιακές παρεμβολές σύμφωνα με τις σχεδιαστικές οδηγίες της Analog Devices [\[31\]](#)

Οι εικόνες 37 και 38 δείχνουν το τελικό αναλογικό κύκλωμα με τον TGS5042. Μπορούμε να διακρίνουμε τους πυκνωτές καθώς και τον ενισχυτή mcp6042 στο κέντρο της πλακέτας.



Εικόνα 37. Τελικό αναλογικό κύκλωμα με τον TGS5042



Εικόνα 38. Αναλογικό κύκλωμα TGS5042

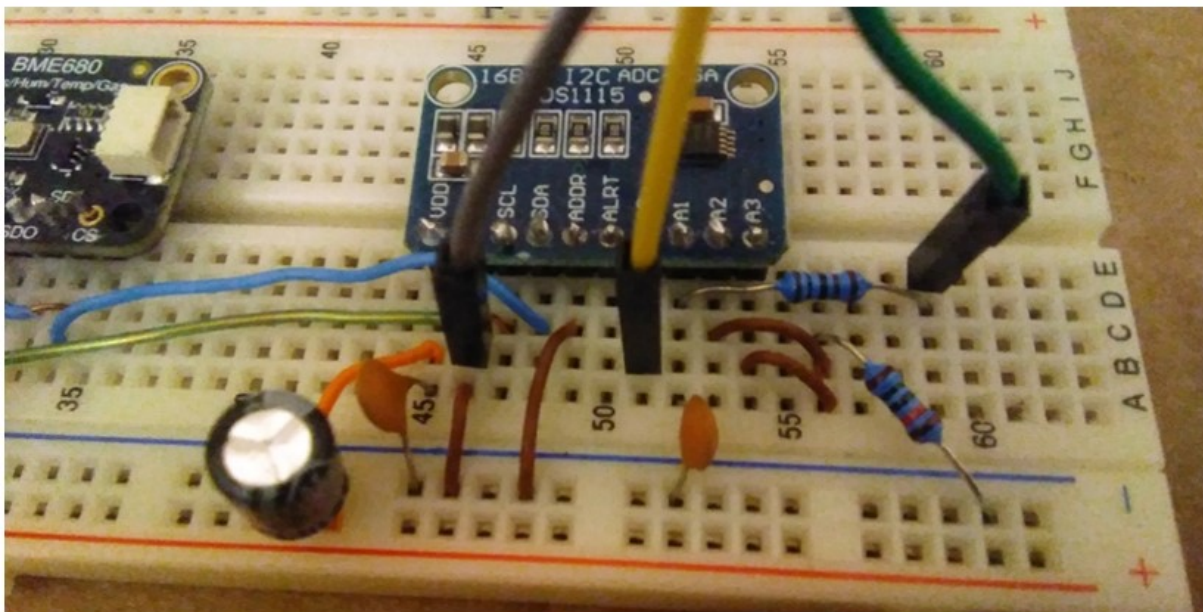
Επίσης, στην μία πλευρά της πλακέτας διακρίνονται τα καλώδια που είναι συνδεδεμένα στο κύκλωμα. Το ζευγάρι πορτοκαλί-λευκό είναι η τάση 5V(πορτοκαλί) και η γείωση(λευκό) και έρχονται από το breadboard. Ενώ η τριάδα των καλωδίων πράσινο-κίτρινο-καφέ αντιστοιχεί

στην τελική τάση (V_{out}) που παράγεται από το κύκλωμα (πράσινο), στη τάση αναφοράς/reference/ V_{ref} (κίτρινο) που είναι γύρω στα 2V και τέλος στην γείωση (καφέ). Συγκεκριμένα, η τριάδα των καλωδίων αυτών είναι η έξοδος προς το breadboard.

Στο breadboard, όπως φαίνεται στην παρακάτω εικόνα 39, διακρίνονται οι διασυνδέσεις προς τις εισόδους του ADS1115. Διακρίνουμε τα εξής:

- Η γείωση (καφέ καλώδιο) που έρχεται από το αναλογικό κύκλωμα και συνδέεται στην γείωση του breakout board.
- Η τάση αναφοράς των 2 Volt (κίτρινο καλώδιο) συνδέεται στην είσοδο A0.
- Η έξοδος από τον αισθητήρα, δηλαδή η V_{out} συνδέεται στο A1.
- Πυκνωτές κεραμικούς στην είσοδο A1 καθώς και στην τροφοδοσία του chip (V_{dd}) καθώς και πυκνωτή ηλεκτρολυτικό στην γραμμή τροφοδοσίας του chip.

Η επιλογή των κεραμικών πυκνωτών (χρώματος καφέ) στις εισόδους A1 και V_{dd} είναι κοινή πρακτική και περιγράφεται στο application note της Texas Instruments για τον ADS1115 [\[34\]](#) και ο ρόλος τους εξηγείται στην συνέχεια.



Εικόνα 39. Σύνδεση Τριάδας εξόδου στον ads1115

4.1.1 Ανάλυση διασύνδεσης του ADS1115

Σύμφωνα με τις προδιαγραφές της Texas Instruments [\[34\]](#), η σταθερότητα της τάσης τροφοδοσίας (VDD) είναι κρίσιμη, καθώς τυχόν διακυμάνσεις της μπορούν να εισάγουν σφάλματα κατά τη φάση της μετατροπής. Για την πλήρη κάλυψη του φάσματος του θορύβου, εφαρμόστηκε μια συνδυαστική διάταξη πυκνωτών και φίλτρων:

- **Κεραμικοί Πυκνωτές Αποσύζευξης 0,1μF:** Τοποθετήθηκαν σε άμεση εγγύτητα με τις ακίδες τροφοδοσίας του ολοκληρωμένου, όπως προτείνεται στο datasheet του ADS1115 adc καθώς και στην είσοδο A1. Λόγω της χαμηλής ισοδύναμης αντίστασης σειράς (ESR) που διαθέτουν, γειώνουν ακαριαία τον υψίσυχνο ψηφιακό θόρυβο και τα παράσιτα μεταγωγής (switching noise) της γραμμής I2C.
- **Low-Pass Filter στην είσοδο A1:** Στην ενεργή αναλογική είσοδο του ADC διαμορφώθηκε ένα φίλτρο διέλευσης χαμηλών συχνοτήτων RC (Low-Pass Filter), ακολουθώντας τις υποδείξεις του κατασκευαστή στο datasheet του ads1115. Το φίλτρο αυτό περιορίζει το εύρος ζώνης (bandwidth) του σήματος εισόδου, επιτρέποντας τη διέλευση μόνο των αργά μεταβαλλόμενων συνιστωσών DC που παράγει ο αισθητήρας CO. Παράλληλα, λειτουργεί σαν φίλτρο, αποκόπτοντας τον θόρυβο υψηλής συχνότητας πριν από τη δειγματοληψία. Το φίλτρο αυτό αποτελείται από έναν κεραμικό πυκνωτή 0,1μF και μια αντίσταση 1K Ohm.
- **Τερματισμός αχρησιμοποίητων αναλογικών εισόδων (floating pins termination):** Οι αναλογικές εισοδοί A2 και A3 του ADS1115 δεν χρησιμοποιούνται στην παρούσα διάταξη. Δεδομένου ότι οι εισοδοί του συγκεκριμένου ADC παρουσιάζουν εξαιρετικά υψηλή σύνθετη αντίσταση (high impedance), η παραμονή τους σε κατάσταση "float" (δηλαδή ασύνδετες) θα τις καθιστούσε επιρρεπείς στη συλλογή ηλεκτρομαγνητικών παρεμβολών, προκαλώντας εσωτερικό θόρυβο και μεταφορά παρασίτων προς τα ενεργά κανάλια. Για τη διασφάλιση της μέγιστης ακρίβειας, οι εισοδοί A2 και A3 τερματίστηκαν στη γείωση μέσω αντίστασης 10K, αποτρέποντας την αλλοίωση των μετρήσεων στα ενεργά κανάλια του αισθητήρα.

4.1.2 Διόρθωση πτώσης τάσης στην γραμμή τροφοδοσίας 3.3 V

Η περιγραφή της δειγματοληψίας μέσω του λογισμικού θα παρουσιαστεί στα επόμενα κεφάλαια. Όμως είναι θεμιτό να επισημανθεί εδώ πως κατά την διάρκεια της δειγματοληψίας και αποστολή δεδομένων, μετρήθηκε η πτώση τάσης της γραμμής τροφοδοσίας 3.3 V. Όπως γνωρίζουμε, η γραμμή τροφοδοσίας 3.3V, συνδέεται και στα 3 breakout boards.

Μετρήθηκε με πολύμετρο και βρέθηκε πως η πιο χαμηλή τιμή ήταν 3.176 Volts σε κατάσταση συνεχούς ταλάντωσης αρχίζοντας από την τιμή 3.3V. Η πτώση τάσης οφείλεται κυρίως στην έντονη ζήτηση ρεύματος, καθώς ανά 15 δευτερόλεπτα γίνεται δειγματοληψία σε σειρά, πρώτα από τον bme60 και έπειτα από τον adc1115 adc. Επίσης, ανά 15 δευτερόλεπτα γίνεται αποστολή δεδομένων στην πλατφόρμα Thingsboard. Επομένως τοποθετήθηκαν 3 ηλεκτρολυτικοί πυκνωτές των 47μF κοντά σε κάθε breakout board πάνω στο breadboard.

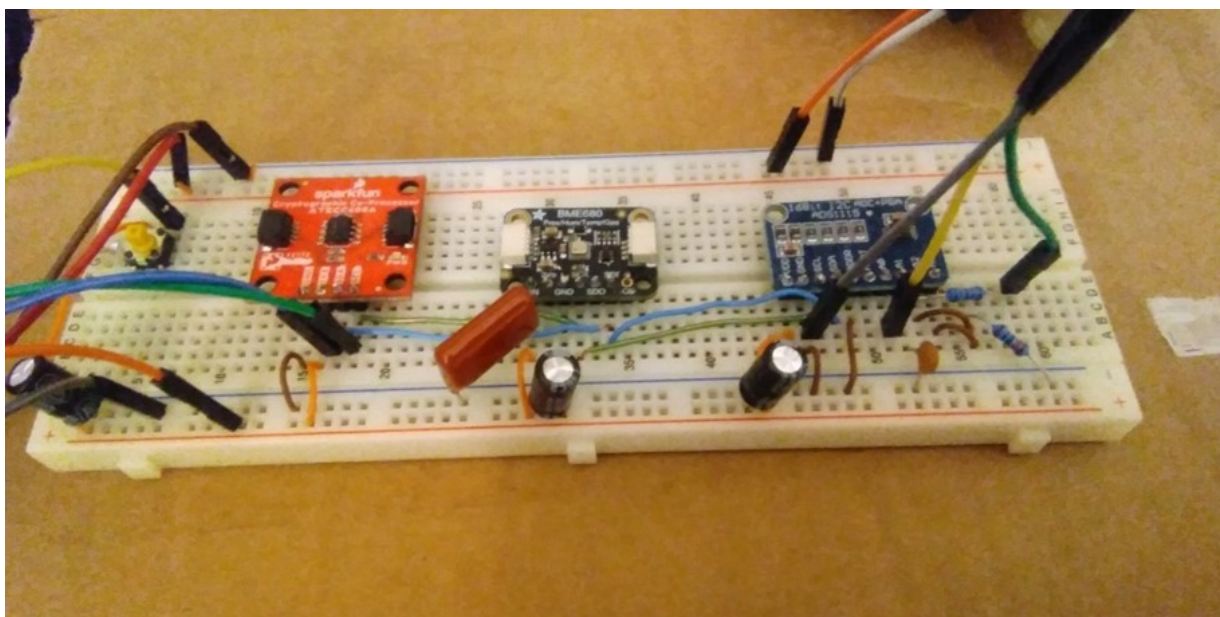
Η τιμή των 47μF αποτελεί μια βέλτιστη τεχνικά και εμπειρική λύση, καθώς παρέχει το απαραίτητο ενεργειακό απόθεμα για την τροφοδοσία του ADC καθώς αποτρέπει στιγμιαίες βυθίσεις της τάσης VDD που θα αλλοίωναν την ακρίβεια της μέτρησης. [\[35\]](#) (Horowitz & Hill, 1999). Έπειτα από την συστοιχία των πυκνωτών κοντά στο σημείο τροφοδοσίας του κάθε breakout board, η βύθιση της τάσης σταμάτησε.

Παρακάτω, στην εικόνα 40, φαίνονται όπως αποθηκεύτηκαν σε βίντεο οι έντονες εναλλαγές της πτώσης τάσης της τροφοδοσίας 3.3 Volt. Οι εναλλαγές μεταξύ των φωτογραφιών συνέβησαν ανά ½ περίπου δευτερόλεπτο.

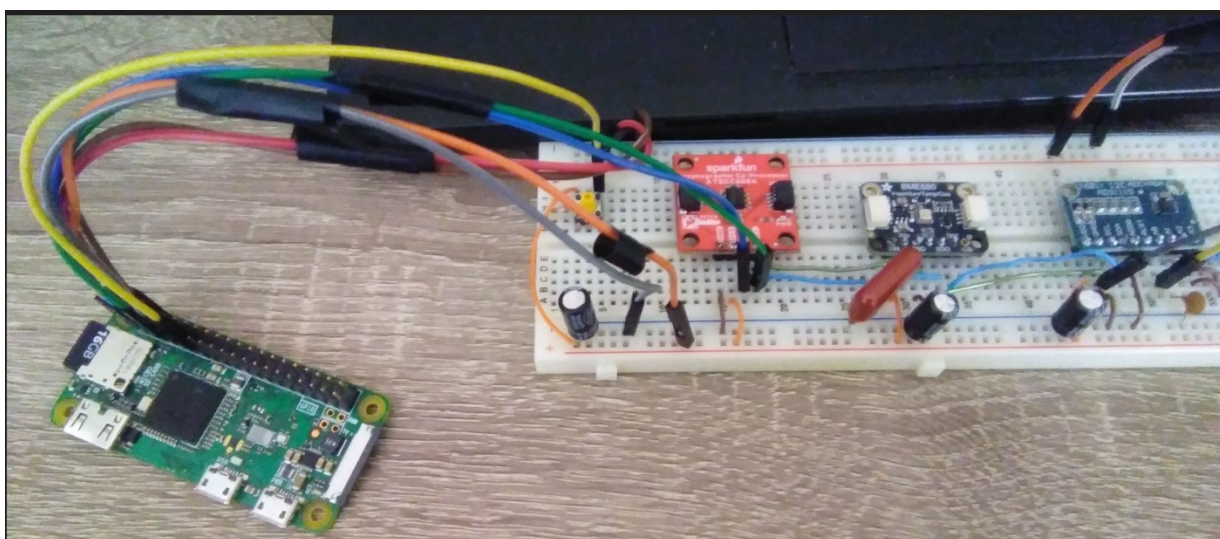


Εικόνα 40. Εναλλαγές τάσης τροφοδοσίας ανά 0.5 sec περίπου

Στις παρακάτω εικόνες (41 & 42) εμφανίζει τη τελική διάταξη όλου του ψηφιακού τμήματος πάνω στο breadboard. Ξεχωρίζουμε τα καλώδια διασύνδεσης τροφοδοσίας, τους πυκνωτές, και τα καλώδια I2C. Κυρίως τα καλώδια I2C είναι το ζεύγος πράσινο-μπλέ, τα οποία πρώτα συνδέονται στο NervesKey και σε μορφή σειράς(daisy chain) συνδέονται και στα υπόλοιπα breakout boards. Στην βάση τροφοδοσίας του BME680 χρησιμοποίησα έναν επιπλέον πυκνωτή τύπου MKT 1μF για απόρριψη υψηλών συχνοτήτων ο οποίος φαίνεται σχετικά εύρωστος. Οι πυκνωτές των 47μF είναι τοποθετημένοι στην γραμμή τροφοδοσίας του κάθε breakout board για τους λόγους που έχουμε εξηγήσει προηγουμένως.



Εικόνα 41. Τελικό αναλογικό κύκλωμα στο breadboard



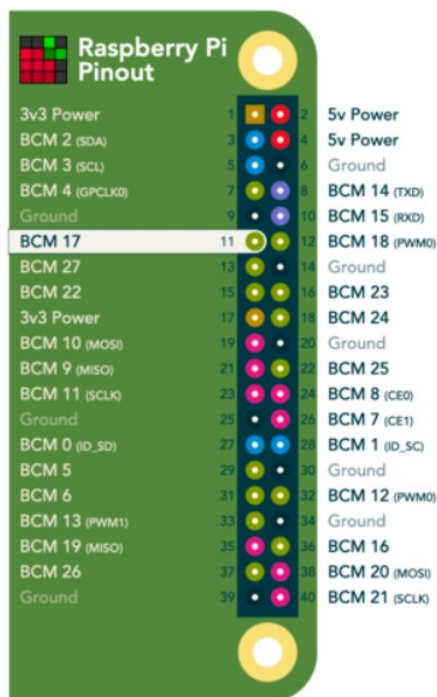
Εικόνα 42. Raspberry PI Zero W με breadboard

4.1.3 Υποσύστημα αρχικής παραμετροποίησης δικτύου

Ένα από τα βασικά προβλήματα κατά την ανάπτυξη ενσωματωμένων συστημάτων βασισμένων σε Nerves είναι η αρχική ρύθμιση ασύρματης σύνδεσης, ιδιαίτερα όταν η συσκευή δεν διαθέτει οθόνη ή πληκτρολόγιο. Για την αντιμετώπιση αυτής της πρόκλησης χρησιμοποιήθηκε η βιβλιοθήκη VintageNetWizard [\[41\]](#) η οποία παρέχει έναν web-based οδηγό ρύθμισης Wi-Fi.

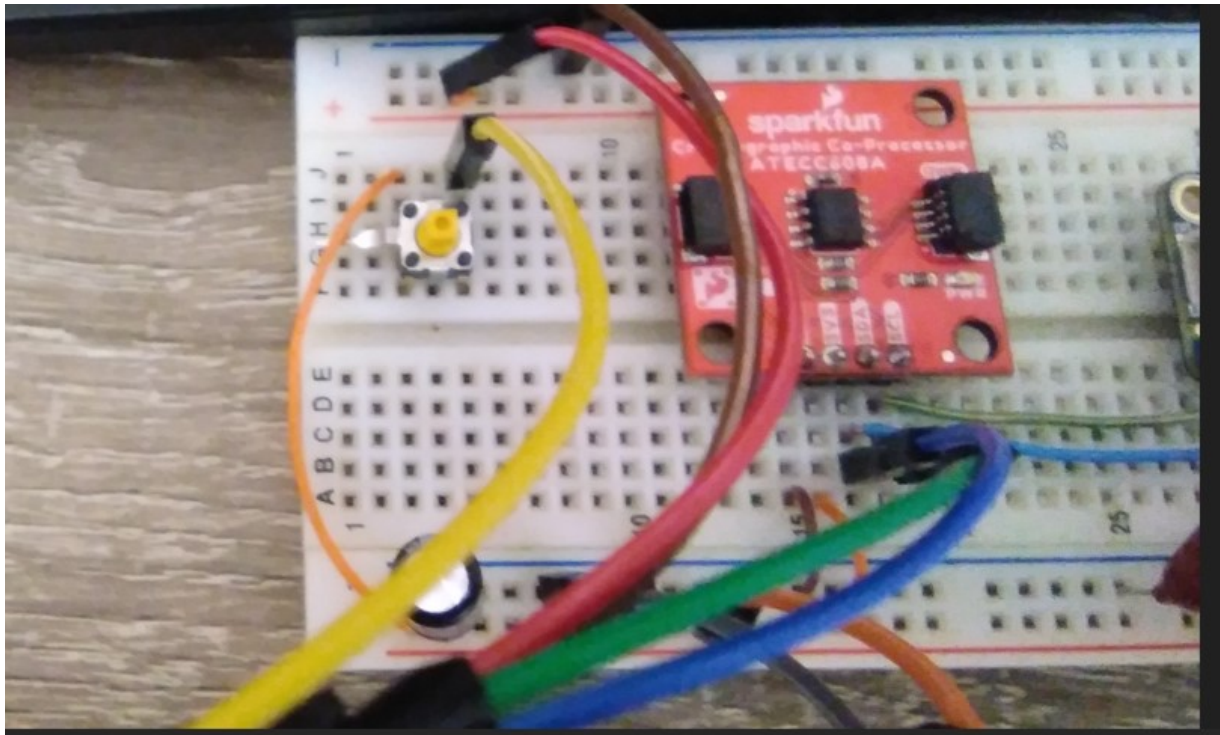
Η συγκεκριμένη βιβλιοθήκη αναλύεται σε επόμενη ενότητα που αφορά την ανάπτυξη του λογισμικού. Ωστόσο, κρίνεται σκόπιμο να αναφερθεί στο σημείο αυτό, καθώς η λειτουργία της προϋποθέτει μια επιπλέον προσθήκη στο breadboard. Συγκεκριμένα, σχεδιάστηκε μια βοηθητική διάταξη με τη χρήση ενός μπουτόν η οποία επιτρέπει στον χρήστη την χειροκίνητη ενεργοποίηση του αντίστοιχου λογισμικού.

Συνδέθηκε, σύμφωνα με τις οδηγίες της βιβλιοθήκης αυτής, το Pin 11 του raspberry pi με ένα μπουτόν (εικόνα 41) που τοποθετήθηκε πάνω στο breadboard. Η άλλη άκρη του μπουτόν συνδέθηκε στην τροφοδοσία της γραμμής των 3.3. Έτσι πατώντας πάνω στο μπουτόν αυτό, μεταφέρεται ρεύμα στο Pin 11. Η βιβλιοθήκη ελέγχει το pin αυτό και ενεργοποιεί ένα μικρό access point που δίνει την δυνατότητα την σύνδεση με τον οικιακό router.



Εικόνα 43. Ενεργοποίηση Pin11 για wifi

Παρακάτω εμφανίζεται το μπουτόν(εικόνα 44) που τοποθετήθηκε πάνω στο breadboard. Το λεπτό πορτοκαλί καλώδιο δεξιά από τον μπουτόν αφορά την τροφοδοσία 3.3 volt ενώ το κίτρινο καλώδιο συνδέεται στην μια άκρη του μπουτόν και η άλλη άκρη καταλήγει στο Pin 11 του Raspberry Pi.



Εικόνα 44 Διάταξη μπουτόν στο breadboard

4.2 Υλοποίηση λογισμικού

Σε αυτήν την ενότητα περιγράφονται τα εργαλεία, οι βιβλιοθήκες, η διαμόρφωση του συστήματος αρχείων και φακέλων, καθώς και οι επιμέρους εφαρμογές (firmware, gas_sensor και gas_sensor_web) και οι διεργασίες που εκτελούνται σε αυτές. Ο πηγαίος κώδικας της εφαρμογής και τα αρχεία σχεδίασης του κυκλώματος (σε μορφή TINA-TI Spice Simulation) είναι διαθέσιμα στο αντίστοιχο αποθετήριο GitHub του συγγραφέα [\[42\]](#)

4.2.1 Περιβάλλον ανάπτυξης Linux

Η υλοποίηση του λογισμικού για το Raspberry PI Zero Wireless απαιτεί ένα σταθερό και ελεγχόμενο περιβάλλον ανάπτυξης ικανό να υποστηρίξει τις διεργασίες διασταυρούμενης μεταγλώττισης (cross-compilation) που επιβάλλουν τα εργαλεία του Elixir Nerves. Για αυτόν το λόγο επιλέχθηκε το Arch Linux το οποίο εξασφαλίζει άμεση πρόσβαση στις τελευταίες εκδόσεις των απαραίτητων εργαλείων συστήματος. Αφού έγινε η εγκατάσταση του ArchLinux, προχωρήσαμε στην εγκατάσταση και παραμετροποίηση του εργαλείου Elixir Nerves ακολουθώντας τις σχετικές οδηγίες [\[43\]](#). Με αυτόν τον τρόπο, αποκτήσαμε πρόσβαση στη Erlang, Elixir και στο εργαλείο mix.

4.2.2 Εργαλείο mix

Κεντρικό ρόλο στην ανάπτυξη, τη διαχείριση και τη μεταγλώττιση της εφαρμογής μας έπαιξε το **mix**, το επίσημο build tool της γλώσσας Elixir [\[14\]](#). Το **mix** δεν περιορίζεται στα καθήκοντα ενός απλού διαχειριστή πακέτων/βιβλιοθηκών αλλά αποτελεί ένα ολοκληρωμένο περιβάλλον αυτοματοποίησης εργασιών και διασύνδεσης με τον μεταγλωττιστή

Το εργαλείο mix χρησιμοποιήθηκε σε όλη την διάρκεια του κύκλου ανάπτυξης για τις ακόλουθες κρίσιμες λειτουργίες:

1. **Διαχείριση αρχιτεκτονικής Poncho:** Μέσω του αρχείου ρυθμίσεων mix.exs, το mix ανέλαβε να γεφυρώσει τις διαφορετικές εφαρμογές firmware, gas_sensor και gas_sensor_web. Η λέξη poncho προέρχεται από έναν προγραμματιστή του elixir nerves για να περιγράψει πως η αρχιτεκτονική αυτή είναι σαν το κλασικό μεξικανικό

ένδυμα (poncho) που καλύπτει το σώμα. Σαν μεταφορική έννοια, χρησιμοποιείται για να περιγράψει πως η βασική εφαρμογή *firmware* καλύπτει σαν πέπλο τις άλλες εφαρμογές και τις ρυθμίζει κατάλληλα.

2. **Διαχείριση εξωτερικών βιβλιοθηκών:** Το εργαλείο επικοινωνεί με το επίσημο αποθετήριο πακέτων Hex και κατεβάζει τις απαραίτητες βιβλιοθήκες.
3. **Παραγωγή τελικού *firmware*:** Μέσω του *mix*, γίνεται η τελική παραγωγή του *firmware*, εγγραφή στο *sdcard* και καθώς δικτυακό προγραμματισμό, δηλαδή το *over the air programming*.

4.2.3 Δημιουργία κεντρικού φακέλου εφαρμογής

Για την δημιουργία της εφαρμογής, έγινε εκτεταμένη μελέτη των οδηγιών χρήσης καθώς και παραδειγμάτων παρόμοιων εφαρμογών που βρίσκονται στο [github](#) [44][45]. Για την αρχικοποίηση της εφαρμογής, δημιουργήθηκε ένας φάκελος: “*CO_monitoring_with_nerves*” και εκεί μέσα εκτελέστηκε η εντολή: *mix nerves.new firmware*. Η λέξη *firmware* ορίζει το όνομα της συγκεκριμένης εφαρμογής, αλλά θα μπορούσε να είναι οποιοδήποτε άλλο όνομα επιλέξει ο δημιουργός του *project*, καθώς αντιπροσωπεύει τον φάκελο που θα φιλοξενήσει το τελικό ενσωματωμένο λειτουργικό σύστημα της συσκευής.

4.2.3.1 Δημιουργία των εφαρμογών *gas_sensor* και *gas_sensor_web*

Εντός του κεντρικού καταλόγου *CO_monitoring_with_nerves*, υλοποιήθηκαν οι δύο επιμέρους εφαρμογές και η διαδικασία ανάπτυξης των οποίων αναλύεται στη συνέχεια.

4.2.3.2 Δημιουργία της εφαρμογής *gas_sensor*

Για τη δημιουργία της βασικής OTP εφαρμογής που παίρνει και αποθηκεύει μετρήσεις και υπολογίζει το μονοξείδιο του άνθρακα εκτελέστηκε η εντολή:

```
mix new gas_sensor --sup
```

Η εντολή `mix new` δημιουργεί μια νέα, αυτόνομη εφαρμογή Elixir. Η παράμετρος `--sup` (supervisor) είναι εξαιρετικά κρίσιμη, καθώς λέει στο `mix` να δημιουργήσει αυτόματα το αρχικό δέντρο επίβλεψης για τις διεργασίες που πρέπει να τρέξουν.

4.2.3.3 Δημιουργία της εφαρμογής `gas_sensor_web`

Για τη δημιουργία της `web` εφαρμογής που παρέχει οπτικοποίηση των δεδομένων, εκτελέστηκε η εντολή του Phoenix framework:

```
mix phx.new gas_sensor_web --no-ecto --no-mailer
```

Οι παράμετροι (flags) που χρησιμοποιήθηκαν έχουν ιδιαίτερη σημασία για ένα ενσωματωμένο/embedded σύστημα:

- `--no-ecto`: Αφαιρεί το υποσύστημα της βάσης δεδομένων (Ecto/PostgreSQL). Επειδή το λειτουργικό σύστημα εκτελείται μόνο για ανάγνωση από κάρτα μνήμης, δεν χρησιμοποιήθηκε κάποια βαριά, παραδοσιακή βάση δεδομένων. Αντίθετα, το ιστορικό των δεδομένων διατηρείται απευθείας στη μνήμη RAM με τη χρήση πινάκων ETS (Erlang Term Storage)
- `--no-mailer`: Αφαιρεί το υποσύστημα αποστολής email, καθώς δεν είναι απαραίτητο για τη λειτουργία της συγκεκριμένης συσκευής, μειώνοντας έτσι το μέγεθος του τελικού κώδικα.

Η τελική δομή του φακέλου που δημιουργήθηκε:

```
CO_monitoring_with_nerves/ ← Ο κεντρικός φάκελος
├── firmware/                ← Το Nerves project (λειτουργικό σύστημα)
├── gas_sensor/              ← Η OTP εφαρμογή (αισθητήρες & λογική)
└── gas_sensor_web/         ← Η Phoenix εφαρμογή (ιστοσελίδα LiveView)
```

4.2.4 Επεξήγηση της εφαρμογής firmware

Η εφαρμογή firmware αποτελεί το ανώτατο επίπεδο (top-level) της αρχιτεκτονικής του συστήματος. Ο ρόλος της δεν σχετίζεται με τη λήψη των μετρήσεων, αλλά με την ενοποίηση όλων των επιμέρους υποσυστημάτων και τη σύνθεσή τους σε ένα αυτόνομο firmware αρχείο για το Raspberry Pi Zero W. Παρόλα αυτά, έχει την ιδιότητα μια εφαρμογής OTP και τα χαρακτηριστικά της είναι κοινά με τις άλλες 2. Υπάρχουν 2 σημαντικά αρχεία τα οποία πρέπει να γνωρίζουμε και τα οποία εκτελούν παρόμοιες διεργασίες και στις 3 άλλες εφαρμογές. Τα αρχεία αυτά είναι το mix.exs και το application.exs (το οποίο συνήθως βρίσκεται στον φάκελο lib).

4.2.4.1 Αρχείο mix.exs

Μέσω του αρχείου ρυθμίσεων mix.exs της εφαρμογής, υλοποιούνται οι ακόλουθες κρίσιμες λειτουργίες υποδομής:

- **Υποστήριξη Υλικού:** Μέσω της λίστας @all_targets καθορίζονται οι συμβατές embedded συσκευές. Στη δική μας περίπτωση, το build tool mix αναγνωρίζει τον στόχο :rpi0 (Raspberry Pi Zero W) και ενσωματώνει αυτόματα την κατάλληλη διανομή Linux (nerves_system_rpi0), η οποία περιλαμβάνει μόνο τους απολύτως απαραίτητους οδηγούς.
- **Διαχείριση δικτύου:** Η εφαρμογή εισάγει και παραμετροποιεί τις βιβλιοθήκες vintage_net, vintage_net_wifi. Αυτές αναλαμβάνουν τη χαμηλού επιπέδου διαχείριση των ασύρματων δικτύων, εξασφαλίζοντας τη σταθερή σύνδεση της συσκευής στο διαδίκτυο.
- **Ρύθμιση Wi-Fi (vintage_net_wizard):** Ενσωματώνεται ο οδηγός που επιτρέπει στη συσκευή, μέσω της πίεσης ενός φυσικού μπουτόν, να εισέρχεται σε κατάσταση Access Point. Αυτό επιτρέπει στον τελικό χρήστη να συνδεθεί με το κινητό του στη συσκευή και να εισάγει τα διαπιστευτήρια του τοπικού Wi-Fi μέσω ενός εύχρηστου οδηγού.
- **Συγχρονισμός ώρας (nerves_time):** Καθώς το Raspberry Pi Zero W στερείται φυσικού ρολογιού μπαταρίας (RTC), η βιβλιοθήκη αυτή αναλαμβάνει να συγχρονίσει το εσωτερικό ρολόι της BEAM εικονικής μηχανής μέσω του πρωτοκόλλου NTP αμέσως μόλις αποκατασταθεί η σύνδεση στο δίκτυο.

- **Ενοποίηση poncho project:** Στο πεδίο των εξαρτήσεων (dependencies ή αλλιώς deps), η εφαρμογή συνδέει τα δύο αυτόνομα παράλληλα projects ως τοπικές εξαρτήσεις. Οι παρακάτω ρυθμίσεις φαίνονται μέσα στο αρχείο mix.exs:

```
{:gas_sensor, path: "../gas_sensor", env: Mix.env()},
```

```
{:gas_sensor_web, path: "../gas_sensor_web", env: Mix.env()}
```

Η δήλωση αυτή αναγκάζει τον compiler να συμπεριλάβει την εφαρμογή *gas_sensor* καθώς και την *gas_sensor_web* στο τελικό firmware και όταν τρέξει η εφαρμογή στο raspberry pi, ενεργοποιούνται και τρέχουν και οι 3 εφαρμογές.

Η παραγωγή του τελικού αρχείου firmware γίνεται με την εκτέλεση των παρακάτω εντολών ενώ βρισκόμαστε μέσα στον φάκελο firmware:

```
export MIX_TARGET=rpi0
```

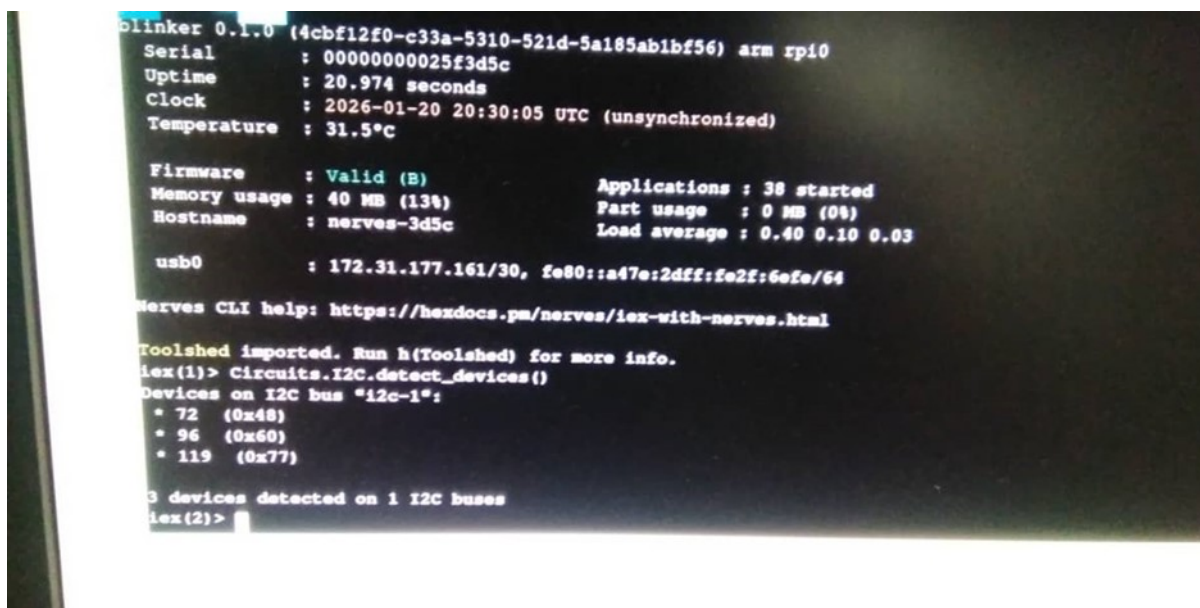
```
mix deps.get
```

```
mix firmware
```

Μετά τη φόρτωση του firmware στη συσκευή (Raspberry Pi Zero), η επικοινωνία και ο έλεγχος του υλικού επιτυγχάνονται μέσω της διαδραστικής κονσόλας IEx (Interactive Elixir). Η εικόνα 45 παρουσιάζει την κονσόλα του Nerves, όπου μέσω της βιβλιοθήκης Circuits.I2C εκτελέστηκε η εντολή `Circuits.I2C.detect_devices()`. Η εκτέλεση αυτή επιβεβαιώνει την επιτυχή επικοινωνία του συστήματος με τρεις διαφορετικές μονάδες υλικού (breakout boards) που συνδέονται στον δίαυλο i2c-1:

- **0x48:** Μετατροπέας ADC (π.χ. ADS1115).
- **0x60:** Συσκευή Nerves Key
- **0x77:** Αισθητήρας περιβάλλοντος (π.χ. BME280).

Η επιτυχής ανίχνευση των παραπάνω διευθύνσεων αποτελεί το πρώτο και κρίσιμότερο στάδιο για την επαλήθευση της ορθής καλωδίωσης και της διαθεσιμότητας των αισθητήρων στο δίαυλο I2C.



Εικόνα 45 Κονσόλα Nerves IEX

4.2.4.2 Αρχείο application.exs

Η βασικότερη συνεισφορά του αρχείου είναι ο ορισμός της λίστας των υποδιεργασιών (children) και η ενεργοποίηση του κεντρικού επόπτη (Supervisor). Οι διεργασίες που δηλώνονται στη λίστα children (όπως GenServers ή Agents) εκκινούνται με αυστηρή ιεραρχική σειρά.

Ο επόπτης αναλαμβάνει την παρακολούθηση αυτών των διεργασιών κατά τη διάρκεια λειτουργίας τους. Εάν κάποια διεργασία καταρρεύσει λόγω απρόβλεπτου σφάλματος, ο επόπτης την επανεκκινεί αυτόματα βάσει μιας προκαθορισμένης στρατηγικής, επαναφέροντας το σύστημα σε μια γνωστή και σταθερή κατάσταση. Στο application.exs καθορίζεται η στρατηγική αντιμετώπισης σφαλμάτων μέσω της παραμέτρου strategy. Για παράδειγμα:

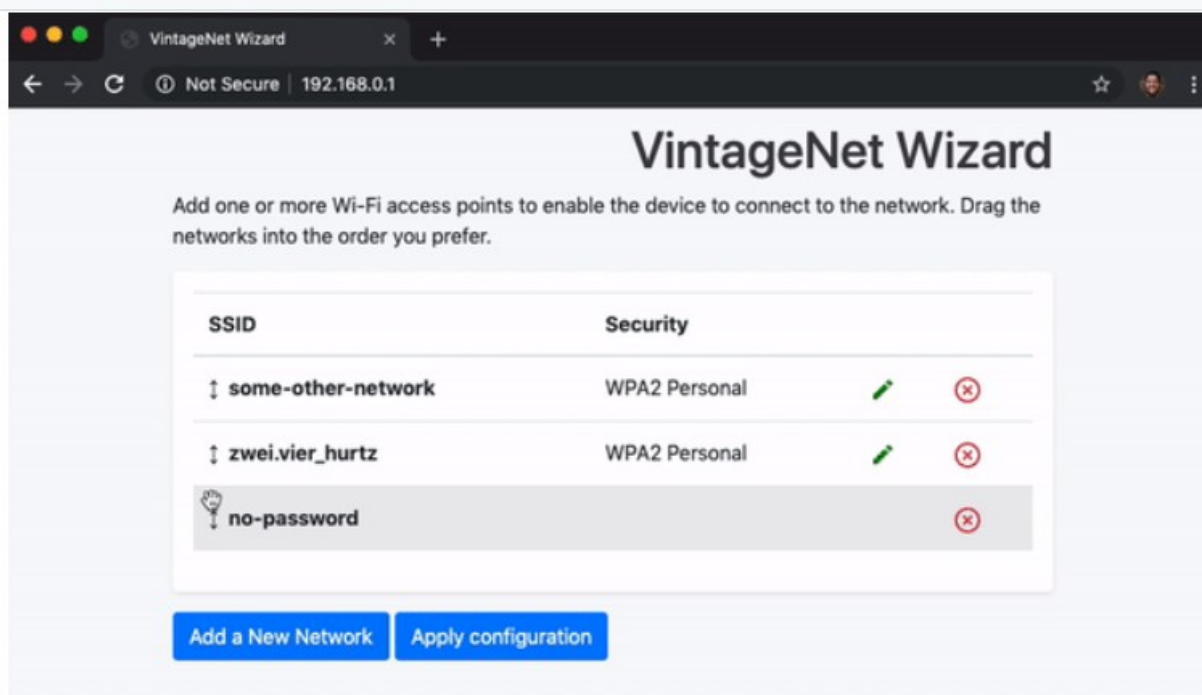
- one_for_one: Εάν μια διεργασία-παιδί καταρρεύσει, επανεκκινείται μόνο η συγκεκριμένη, χωρίς να επηρεαστούν οι υπόλοιπες.
- one_for_all: Εάν μια διεργασία καταρρεύσει, ο επόπτης τερματίζει και επανεκκινεί όλες τις διεργασίες του δέντρου.
- rest_for_one: Εάν μια διεργασία καταρρεύσει, επανεκκινείται αυτή και όσες βρίσκονται μετά από αυτήν στη σειρά δήλωσης.

Στην δική μας περίπτωση και στις 3 εφαρμογές, ο επόπτης έχει οριστεί να είναι: one_for_one.

4.2.4.3 Οδηγός δικτύωσης τύπου Access Point

Για την υλοποίηση της συγκεκριμένης δυνατότητας δημιουργήθηκε διεργασία τύπου *GenServer* που ονομάστηκε *button.ex* και βρίσκεται στον φάκελο *firmware/lib/firmware/*, η οποία περιμένει και διαβάζει την ενέργεια του *button*. Μόλις πατηθεί το μπουτόν, τρέχει μια διαδικτυακή υπηρεσία τύπου ανοιχτού τύπου *access point* για 10 λεπτά. Ορίστηκε το όνομα του *access point* να είναι το *TGS5042-Setup* καθώς και σε ποιο *port* θα ακούει. Ορίσαμε το *port* 8080. Το αρχείο ρύθμισης του *access point* καθώς και άλλων μέτρων βρίσκεται στο *firmware/config/target.exs*

Στην εικόνα 46, παρουσιάζεται η διεπαφή του *access point* που δημιουργεί η συσκευή. Μέσω αυτής της σελίδας, ο χρήστης έχει τη δυνατότητα να εντοπίσει και να επιλέξει το ασύρματο δίκτυο της οικίας του. Μόλις ολοκληρωθεί η σύνδεση, η εφαρμογή *elixir nerves* αποθηκεύει μόνιμα τα διαπιστευτήρια αυτά, ώστε να τα χρησιμοποιεί αυτόματα για κάθε μελλοντική σύνδεση στο διαδίκτυο, ακόμη και μετά από επανεκκίνηση της συσκευής.



Εικόνα 46 Access point μέσω του Elixir Nerves

4.2.5 Επεξήγηση της εφαρμογής gas_sensor

Η εφαρμογή gas_sensor αποτελεί την καρδιά όλου του συστήματος. Ενώ η εφαρμογή firmware ασχολείται με το πακετάρισμα όλων των επιμέρους συστημάτων καθώς και με τη δικτύωση, η εφαρμογή gas_sensor είναι αποκλειστικά υπεύθυνη για την επιχειρησιακή λογική (business logic) του σταθμού μέτρησης. Συγκεκριμένα, αναλαμβάνει τη χαμηλού επιπέδου επικοινωνία με τον αναλογικό-σε-ψηφιακό μετατροπέα (ADC) μέσω του διαύλου I2C, τη συλλογή των μετρήσεων από τον BME680, το φιλτράρισμα του ηλεκτρικού θορύβου, καθώς και τη διαχείριση της ιστορικότητας των δεδομένων στη μνήμη RAM της συσκευής πριν αυτά προωθηθούν στο cloud. Μέσω του τοπικού αρχείου mix.exs, η εφαρμογή αυτή ενσωματώνει βιβλιοθήκες για πρόσβαση στις γραμμές I2C καθώς και για την αποστολή δεδομένων στο ThingsBoard μέσω του MQTT πρωτοκόλλου.

4.2.5.1 Ανάλυση του GasSensor.Application

Όπως αναφερθήκαμε και προηγουμένως, τα αρχεία application.ex είναι κοινά και στις 3 εφαρμογές και κάνουν εκκίνηση διεργασιών που αφορούν τις συγκεκριμένες εφαρμογές. Το αρχείο gas_sensor/lib/gas_sensor/application.ex αποτελεί το σημείο εισόδου της εφαρμογής. Αρχικοποιεί το δέντρο επίβλεψης (GasSensor.Supervisor) και θέτει σε λειτουργία τις εξής διεργασίες(gen_servers) με αυστηρή σειρά ιεραρχίας:

- **GasSensor.ReadingAgent:** Δημιουργεί χώρο στη μνήμη RAM για την αποθήκευση της τρέχουσας μέτρησης.
- **bme680_sensor (Βιβλιοθήκη BMP280):** Αρχικοποιεί τον driver του αισθητήρα περιβάλλοντος στον διάυλο I2C και απαντά σε αιτήματα για μετρήσεις θερμοκρασίας.
- **GasSensor.History:** Δημιουργεί πίνακες ETS στη μνήμη για την αποθήκευση ιστορικών δεδομένων 7 ημερών.
- **GasSensor.Sensor:** Ο κεντρικός GenServer που εκτελεί τη δειγματοληψία, ανάλυση δεδομένων. Με αυτή τη δομή, η ιστοσελίδα Phoenix LiveView διαβάζει ακαριαία από τη μνήμη (Agent και History), διασφαλίζοντας ομαλή λειτουργία.
- **GasSensor.TelemetryThingsBoard:** Αναλαμβάνει να στείλει τα δεδομένα στο ThingsBoard ανά τακτά χρονικά διαστήματα(15 δευτερόλεπτα)

4.2.5.2 GasSensor.Reading Agent (Στιγμιότυπο μετρήσεων)

Στην αρχιτεκτονική του συστήματος, αυτό το άρθρωμα/module λειτουργεί ως ένας ενδιάμεσος, ασύγχρονος buffer που διατηρεί αποθηκευμένη την τελευταία έγκυρη μέτρηση. Είναι βασικά ένας GenServer που διατηρεί κατάσταση(state).

Η λειτουργικότητα και οι σχεδιαστικές επιλογές του module συνοψίζονται στα εξής:

- **Διαχωρισμός πρόσβασης:** Διασφαλίζεται μια αυστηρή ροή δεδομένων όπου ο κεντρικός GenServer δειγματοληψίας (GasSensor.Sensor) είναι ο μοναδικός που έχει δικαίωμα εγγραφής και ενημέρωσης του Agent μέσω της συνάρτησης **add_sample**. Αντίθετα, η web διεπαφή (Phoenix LiveView) εκτελεί μόνο αναγνώσεις.
- **Αποφυγή διενέξεων υλικού:** Η απευθείας προσπέλαση του αισθητήρα από πολλαπλές HTTP αιτήσεις χρηστών καθίσταται αδύνατη, καθώς η σύγχρονη ανάγνωση από τον φυσικό δίαυλο I2C είναι χρονοβόρα και θα οδηγούσε σε αστάθεια ή υπερφόρτωση του υλικού.
- **Ανοχή σε σφάλματα και απομόνωση:** Σε περίπτωση που ο φυσικός αισθητήρας παρουσιάσει προσωρινό σφάλμα επικοινωνίας, ο agent παραμένει ενεργός στη μνήμη. Το σύστημα συνεχίζει να προσφέρει την τελευταία γνωστή έγκυρη τιμή, αποτρέποντας την εμφάνιση σφαλμάτων ή το πάγωμα της οθόνης του τελικού χρήστη.
- **Συγχρονισμός δεδομένων:** Προσθέτει χρονοσήμανση στα δεδομένα πριν καταχωρηθούν.

Ο agent όταν ξεκινήσει αρχικοποιεί ένα map(πίνακας κατακερματισμού) της Elixir(αντίστοιχη λειτουργία τύπου map υπάρχει και στην Java) όπου καταχωρεί τις παρακάτω τιμές:

```
@default_reading %{  
  co_ppm: 0.0,  
  temperature_c: 0.0,  
  humidity_rh: 0.0,  
  pressure_pa: 0.0,  
  dew_point_c: 0.0,  
  gas_resistance_ohms: 0.0,  
  cpu_temperature: 0.0,  
  vref: 0.0,  
  vsensor: 0.0,  
  vsensor_offset: 0.0,  
  vdifferential: 0.0,
```

```
vref_variance: 0.0,  
}
```

Η αρχική κατάσταση (default reading) του ReadingAgent περιλαμβάνει ένα διευρυμένο σύνολο περιβαλλοντικών και ηλεκτρικών παραμέτρων (όπως *dew_point_c*, *gas_resistance_ohms*, *vref_variance* κ.λπ.). Η επιλογή αυτή έγινε με γνώμονα την αρχιτεκτονική επεκτασιμότητα του λογισμικού. Αν και στην παρούσα φάση της διπλωματικής ορισμένες από αυτές τις μεταβλητές δεν συμμετέχουν ενεργά στους τελικούς υπολογισμούς ή στην απεικόνιση, η προκαθορισμένη δομή τους εξασφαλίζει ότι το σύστημα είναι έτοιμο να δεχθεί μελλοντικές αναβαθμίσεις.

Συγκεκριμένα, χρησιμοποιούνται ενεργά οι παρακάτω τιμές: *co_ppm*, *temperature_c*, *vref*, *vsensor*, *vsensor_offset*, *vdifferential* οι οποίες θα εξηγηθούν στην ενότητα του GasSensor.Sensor.

4.2.5.3 bme680_sensor

Το module αυτό είναι ένας GenServer που τρέχει μέσα από την βιβλιοθήκη που ενσωματώσαμε στο αρχείο *mix.exs*:

```
{:bme280, "~> 0.2" }
```

Παρέχει άμεση πρόσβαση στα δεδομένα του bme680. Έτσι, σε οποιοδήποτε μέρος του κώδικα μπορούμε να καλέσουμε την συνάρτηση *BMP280.measure* και να λάβουμε τις ακόλουθες τιμές: *altitude_m*, *dew_point_c*, *gas_resistance_ohms*, *humidity_rh*, *pressure_pa*, *temperature_c*, *timestamp_ms*

4.2.5.4 GasSensor.History

Το GasSensor.History είναι ένας αυτόνομος GenServer υπεύθυνος για τη διατήρηση και διαχείριση των ιστορικών μετρήσεων (χρονοσειρών) για ένα μέγιστο διάστημα 7 ημερών.

Η αρχιτεκτονική του module βασίζεται στις ακόλουθες κρίσιμες σχεδιαστικές αρχές:

- **Χρήση Μνήμης ETS (:ordered_set):** Αντί για τη χρήση εξωτερικής βάσης δεδομένων ή εγγραφών στον δίσκο, επιλέχθηκε ο ενσωματωμένος μηχανισμός ETS (Erlang Term Storage) της BEAM. Ο πίνακας ρυθμίζεται ως :ordered_set και :public, εξασφαλίζοντας γρήγορες αναζητήσεις.
- **Αυτόματη εκκαθάριση:** Ο GenServer εκτελεί μια περιοδική διεργασία στο παρασκήνιο (κάθε 60 δευτερόλεπτα), η οποία υπολογίζει το χρονικό όριο λήξης (retention cutoff των 7 ημερών) και διαγράφει αυτόματα τις παλιές εγγραφές (:ets.select_delete/2), λειτουργώντας ως ένας αποδοτικός κυκλικός buffer στη μνήμη RAM.
- **Αποδοτικότητα μνήμης:** Με ρυθμό δειγματοληψίας ανά 15 δευτερόλεπτα, το σύστημα αποθηκεύει συνολικά 40.320 δείγματα για το διάστημα των 7 ημερών. Με μέσο μέγεθος δείγματος τα 400 bytes, η συνολική κατάληψη στη RAM δεν ξεπερνά τα ~24 MB, μέγεθος αμελητέο για τα ~300 MB που διαθέτει το Raspberry Pi Zero W για την εικονική μηχανή BEAM.

4.2.5.5 Λογική Ανάγνωσης Αισθητήρα ASD1115 — GasSensor.Sensor

Καθώς αυτό το άρθρωμα/module είναι το σημαντικότερο από όλα [\[46\]](#), θα επιχειρηθεί να εξηγηθεί πιο λεπτομερώς. Έγινε ενδελεχής ανάλυση του τεχνικού φυλλαδίου (datasheet) του **ADS1115** με σκοπό την πλήρη κατανόηση της αρχιτεκτονικής και των τρόπων λειτουργίας του [\[34\]](#).

Η διαχείριση του διαύλου I2C γίνεται αποκλειστικά μέσω του module GasSensor.Sensor, το οποίο υλοποιεί έναν OTP GenServer. Δεδομένου ότι ο αισθητήρας BME680 και ο αναλογικός μετατροπέας ADS1115 είναι συνδεδεμένοι στον ίδιο κοινόχρηστο δίαυλο(i2c bus), η ύπαρξη μιας κεντρικής διεργασίας (process) για τη ροή των δεδομένων κρίθηκε αρχιτεκτονικά απαραίτητη, καθώς εξαλείφει την πιθανότητα εμφάνισης συνθηκών ανταγωνισμού (race conditions).

Μέσα στον κώδικα ορίστηκε η μεταβλητή: `@sample_interval` η οποία αντιστοιχεί στα 15 δευτερόλεπτα. Ο GenServer εκτελεί έναν πλήρη κύκλο μέτρησης, ενημερώνει τον

GasSensor.ReadingAgent με το αποτέλεσμα, και προγραμματίζει τον επόμενο κύκλο μέσω `Process.send_after/3`. Το web interface διαβάζει αποκλειστικά από τον Agent, χωρίς να περιμένει το I2C bus.

4.2.5.5.1 Μηχανισμός Δειγματοληψίας και Ανάγνωσης Καταχωρητών (Registers)

Ο αισθητήρας TGS5042 παράγει αναλογικό σήμα τάσης, το οποίο ψηφιοποιείται από τον 16-bit ADC ADS1115 μέσω I2C. Ο ADC διαθέτει τέσσερις καταχωρητές. Το δικό μας σύστημα χρησιμοποιεί δύο: τον καταχωρητή ρύθμισης (0x01) και τον καταχωρητή μετατροπής (0x00).

Για κάθε ανάγνωση, γράφεται στον καταχωρητή ρύθμισης μια τιμή 16-bit που ορίζει: ποιο κανάλι θα διαβαστεί, το εύρος τάσης ($\pm 4.096V$), τη λειτουργία single-shot και ρυθμό δειγματοληψίας 8 SPS. Το bit OS (Operational Status) του καταχωρητή ρύθμισης χρησιμοποιείται ως σήμα ετοιμότητας: το σύστημα ψηφιακά ελέγχει (κάνει polling) κάθε 10ms έως ότου το bit αυτό τεθεί σε 1, υποδηλώνοντας ολοκλήρωση της μετατροπής. Η μέγιστη αναμονή ορίζεται σε 200ms (20 προσπάθειες $\times$ 10ms).

Διαβάζονται δύο κανάλια σε κάθε κύκλο. Το A0 που φέρει την τάση αναφοράς των 2 Volt και το A1 που φέρει το σήμα εξόδου του τελεστικού ενισχυτή, δηλαδή του TGS5042. Η ανάγνωση χρησιμοποιεί ατομική εντολή `write_read` της βιβλιοθήκης `Circuits.I2C`, ώστε ο δείκτης καταχωρητή και η ανάγνωση να γίνονται σε μία συναλλαγή χωρίς να αποδεσμευτεί το i2c bus.

Το αρχικό αποτέλεσμα είναι ανυπόγραφος (unassigned) ακέραιος 16-bit. Μετατρέπεται σε προσημασμένο (two's complement) αφαιρώντας 65536 από τιμές άνω του 32767, καλύπτοντας έτσι αρνητικές τάσεις:

```
raw = if raw > 32767, do: raw - 65536, else: raw
```

Σε κάθε κύκλο μέτρησης λαμβάνονται `@number_of_samples = 7` διαδοχικά δείγματα από τον ADC για την τάση αναφοράς και για την έξοδο του αισθητήρα. Από αυτά υπολογίζεται η διάμεσος (median) για την τάση αναφοράς (V_{ref}) και για τάση/έξοδο αισθητήρα (V_{out}). Η

επιλογή της διαμέσου έναντι του αριθμητικού μέσου είναι σκόπιμη: ο μέσος όρος επηρεάζεται σημαντικά από μεμονωμένες ακραίες τιμές που μπορούν να προκύψουν από ηλεκτρικές παρεμβολές. Η διάμεσος αντίθετα παραμένει σταθερή ακόμη και αν ένα ή δύο δείγματα από τα επτά είναι εσφαλμένα. Η προσέγγιση αυτή αναφέρεται στη βιβλιογραφία [\[7\]](#) (J, 2010), Krumm, ως κατάλληλη για φιλτράρισμα σημάτων αισθητήρων σε ενσωματωμένα συστήματα. Ο υπολογισμός ταξινομεί τη λίστα και επιστρέφει το μεσαίο στοιχείο για περιττό πλήθος, ή το μέσο των δύο μεσαίων για άρτιο.

4.2.5.5.2 Ανάλυση(resolution) συστήματος

Για την πλήρη αξιολόγηση της ακρίβειας του σταθμού μέτρησης, πραγματοποιήθηκε θεωρητικός προσδιορισμός της διακριτικής ικανότητας (resolution) του συστήματος, δηλαδή της ελάχιστης μεταβολής (PPM) μονοξειδίου του άνθρακα (CO) που μπορεί να αντληφθεί ο αναλογικό-ψηφιακός μετατροπέας (ADC).

Ο υπολογισμός βασίζεται στον συνδυασμό των ρυθμίσεων του μετατροπέα ADS1115, των προδιαγραφών του αισθητήρα Figaro TGS5042 και των χαρακτηριστικών του αναλογικού κυκλώματος προσαρμογής σήματος.

1. Διακριτική ικανότητα τάσης του ADC:

Με βάση τις ρυθμίσεις του καταχωρητή, το εύρος τάσης του μετατροπέα ορίστηκε στα $\pm 4.096\text{ V}$ (συνολικό εύρος 8.192 V). Εφόσον η ψηφιοποίηση γίνεται σε ανάλυση 16-bit προσημασμένου ακεραίου (συνολικά 65,536 επίπεδα), η διακριτική ικανότητα τάσης ανά ψηφιακό βήμα (LSB — Least Significant Bit) υπολογίζεται από τη σχέση:

$$\frac{8.192\text{ V}}{65,536} = 125\text{ }\mu\text{V/LSB}$$

2. Μετατροπή ρεύματος σε τάση (κύκλωμα ενίσχυσης)

Ο αισθητήρας TGS5042 λειτουργεί παράγοντας ένα εξαιρετικά χαμηλό ρεύμα, το οποίο παρουσιάζει ονομαστική ευαισθησία ίση με **1.525 nA/PPM**, όπως γνωρίζουμε. Το

σχεδιασμένο κύκλωμα ενίσχυσης χρησιμοποιώντας αντίσταση ανάδρασης $R_f=1.2\text{ M}\Omega$, μετατρέπει αυτό το ρεύμα σε τάση εξόδου σύμφωνα με τη σχέση:

$$V_{out} = I_{sensor} * R_f$$

Επομένως, η μεταβολή της τάσης στην έξοδο του κυκλώματος για κάθε μεταβολή της συγκέντρωσης κατά 1 PPM ισούται με:

$$\Delta V/PPM = \frac{1.525\text{ nA}}{PPM} * 1.2\text{ M}\Omega = \mathbf{1.83\text{ }\mu V/PPM}$$

3.Τελική διακριτική ικανότητα σε PPM

Συσχετίζοντας τη διακριτική ικανότητα τάσης του ADC με τη μεταβολή τάσης ανά μονάδα PPM του κυκλώματος, προκύπτει η ελάχιστη ανιχνεύσιμη τιμή συγκέντρωσης αερίου ανά ψηφιακό βήμα:

$$\text{Διακριτική ικανότητα PPM} = \frac{\frac{125\text{ }\mu V}{LSB}}{\frac{1.83\text{ }\mu V}{PPM}} \approx \mathbf{0.0683\text{ PPM/LSB}}$$

Συμπερασματικά, το σύστημα μας διαθέτει τη θεωρητική ικανότητα να καταγράφει μεταβολές της τάξης των **0.0683 PPM ανά ψηφιακό βήμα**, γεγονός που καθιστά τον σταθμό μέτρησης εξαιρετικά ευαίσθητο και κατάλληλο για εφαρμογές περιβαλλοντικής επιτήρησης. Στην πράξη, αν και ο πραγματικός θόρυβος του περιβάλλοντος μπορεί να μειώσει ελαφρώς την ενεργό ανάλυση, η εφαρμογή του ψηφιακού φίλτρου διαμέσου (median filter) στον κώδικα της Elixir σταθεροποιεί τις μετρήσεις κοντά σε αυτό το θεωρητικό όριο.

4.2.5.5.3 Αναλογο-μετρική (Ratiometric) Προσέγγιση

Το σύστημα μας εφαρμόζει αναλογο-μετρική προσέγγιση για την αντιμετώπιση των μικρο-διακυμάνσεων της τάσης τροφοδοσίας των 5V, οι οποίες επηρεάζουν ταυτόχρονα την έξοδο

του αισθητήρα και την τάση αναφοράς. Αντί για τη χρήση απόλυτων τιμών, υπολογίζεται ο τρέχον λόγος των δύο τάσεων:

$$LiveRatio = \frac{V_{out}}{V_{ref}}$$

όπου V_{out} η έξοδος από τον τελεστικό ενισχυτή και V_{ref} είναι η τάση των 2V (reference). Ο λόγος αυτός κανονικοποιείται στη συνέχεια γύρω από την ιδανική αναφορά των 2V ($live_ratio * 2.0$). Με τον τρόπο αυτό, οι διακυμάνσεις της πηγής που επηρεάζουν εξίσου τον αριθμητή και τον παρονομαστή αλληλοαναιρούνται μαθηματικά, θωρακίζοντας την ακρίβεια της μέτρησης.

4.2.5.5.4 Θερμοκρασιακή Αντιστάθμιση και Υπολογισμός Συγκέντρωσης CO (PPM)

Η ευαισθησία του αισθητήρα TGS5042 μεταβάλλεται με τη θερμοκρασία. Σύμφωνα με το application note της Figaro [\[26\]](#) ("APPLICATION NOTES FOR TGS5xxx SERIES"), η αντιστάθμιση γίνεται με τον συντελεστή διόρθωσης CF που ορίζεται ανά βαθμό Κελσίου στο διάστημα -10°C έως 55°C. Οι τιμές αυτές έχουν ενσωματωθεί ως πίνακας αναζήτησης (lookup table) απευθείας στον κώδικα ως module attribute `@temp_cf_table`. Η θερμοκρασία που λαμβάνεται από τον BME680 στρογγυλοποιείται στον πλησιέστερο ακέραιο και χρησιμοποιείται ως κλειδί αναζήτησης. Για θερμοκρασίες εκτός εύρους, επιστρέφονται οι τιμές των άκρων του πίνακα.

Ο διορθωμένος συντελεστής ευαισθησίας α υπολογίζεται ως:

$$\alpha = (\text{sensitivity_nA_per_ppm} \times 10^{-9}) \times CF$$

όπου $\text{sensitivity_nA_per_ppm} = 1.525$. Αυτή είναι η ειδική εργοστασιακή τιμή που αναγράφεται στο συγκεκριμένο module του αισθητήρα που αγοράσαμε.

Η τελική συγκέντρωση CO υπολογίζεται σύμφωνα με την εξίσωση του datasheet (παράγραφος 6-4)[\[26\]](#):

$$C = \frac{V_{out} - V_0}{a \times R_f}$$

όπου:

- V_{out} είναι η κανονικοποιημένη τάση εξόδου του αισθητήρα.
- V_0 (V zero) είναι η τάση αναφοράς σε καθαρό αέρα (offset), η οποία αντιστοιχεί στη μεταβλητή `vsensor_offset`. Αποθηκεύεται μόνιμα και ορίζεται δυναμικά από τον χρήστη μέσω του LiveView web interface.
- a : ο διορθωμένος συντελεστής ευαισθησίας βασισμένος στην τρέχουσα θερμοκρασία περιβάλλοντος.
- R_f : Η αντίσταση ανάδρασης του τελεστικού ενισχυτή, ίση με $1.2\text{M}\Omega$ (`@r3_ohms`).

Ο υπολογισμός του CO πραγματοποιείται από την συνάρτηση `convert_to_ppm` που βρίσκεται στην διαδρομή `gas_sensor/lib/gas_sensor/sensor.ex` στο αντίστοιχο github αποθετήριο [\[42\]](#)

Το αποτέλεσμα περιορίζεται στο διάστημα $[0.0, 10000.0]$ PPM μέσω του κώδικα Elixir:

```
(delta / (alpha * @r3_ohms)) |> max(0.0) |> min(10_000.0)
```

4.2.5.5 Διαχείριση Σφαλμάτων και Ανοχή σε Σφάλματα

Ο κύκλος μέτρησης χρησιμοποιεί το `with construct` της Elixir, το οποίο επιτρέπει αλυσιδωτή εκτέλεση βημάτων όπου κάθε βήμα προχωράει μόνο αν το προηγούμενο επέστρεψε `{:ok, ...}`. Σε περίπτωση αποτυχίας οποιουδήποτε βήματος — ανάγνωση BME680, ανάγνωση CPU θερμοκρασίας, ή συλλογή δειγμάτων ADC — η εκτέλεση εκτρέπεται στο `else block`.

Σε κάθε σφάλμα, ο `GenServer` δεν τερματίζεται. Αντίθετα, δημιουργεί μια κενή μέτρηση (`@empty_reading`) με μηδενικές τιμές, διατηρεί αναλλοίωτες τις παραμέτρους `i2c` και `vsensor_offset` για τον επόμενο κύκλο, καταγράφει το σφάλμα μέσω `Logger` και ενημερώνει τον `Agent` με `status :error`. Έτσι το web interface παραμένει λειτουργικό και εμφανίζει την κατάσταση σφάλματος χωρίς να κρυσάρει η εφαρμογή.

4.2.5.6 GasSensor.TelemetryThingsboard

Το module GasSensor.TelemetryThingsboard υλοποιεί έναν OTP GenServer που είναι υπεύθυνος για την περιοδική αποστολή μετρήσεων στην πλατφόρμα ThingsBoard μέσω πρωτοκόλλου MQTT. Για την υλοποίηση της MQTT επικοινωνίας χρησιμοποιείται η βιβλιοθήκη Tortoise, η οποία παρέχει ασύγχρονο MQTT client για Elixir.

Κατά την εκκίνηση του GenServer, δημιουργείται σύνδεση με τον ThingsBoard broker (mqtt.eu.thingsboard.cloud, port 1883) μέσω του Tortoise.Supervisor. Η αυθεντικοποίηση γίνεται με access token που φορτώνεται από τη ρύθμιση της εφαρμογής, αποφεύγοντας την ενσωμάτωσή του στον κώδικα. Σε περίπτωση αποτυχίας σύνδεσης, ο GenServer τερματίζεται με `{:stop, reason}` και ο Supervisor αναλαμβάνει την επανεκκίνησή του.

Κάθε 15 δευτερόλεπτα, το μήνυμα `:send_telemetry` ενεργοποιεί ανάγνωση της τελευταίας μέτρησης από τον GasSensor.ReadingAgent. Πριν την αποστολή, αφαιρείται το πεδίο `:i2c` από το map, καθώς περιέχει αναφορά σε πόρο του λειτουργικού συστήματος που δεν είναι δυνατό να σειριοποιηθεί σε JSON. Το υπόλοιπο map κωδικοποιείται σε JSON μέσω της βιβλιοθήκης Jason και δημοσιεύεται στο topic `v1/devices/me/telemetry` με QoS 1:

```
Tortoise.publish(@client_id, @topic, payload, qos: 1)
```

Η χρήση QoS 1 εξασφαλίζει ότι κάθε μήνυμα παραδίδεται στον broker τουλάχιστον μία φορά. Η επιβεβαίωση παράδοσης (acknowledgment) από τον broker επιστρέφεται ασύγχρονα ως μήνυμα `{{Tortoise, @client_id}, _ref, :ok}` και χειρίζεται από αντίστοιχο `handle_info` callback, αποφεύγοντας έτσι τον τερματισμό του GenServer από μη αναμενόμενα μηνύματα.

Ο προγραμματισμός της επόμενης αποστολής γίνεται πάντα μέσω `Process.send_after/3` στο τέλος κάθε κύκλου, ανεξάρτητα από την επιτυχία ή αποτυχία δημοσίευσης, διασφαλίζοντας τη συνέχεια της λειτουργίας:

```
defp schedule_work do
```

```
Process.send_after(self(), :send_telemetry, 15_000)
```

```
End
```

4.2.6 Εφαρμογή gas_sensor_web — Web Interface με Phoenix LiveView

4.2.6.1 Επισκόπηση και Αρχιτεκτονική

Το gas_sensor_web υλοποιεί το περιβάλλον παρακολούθησης του συστήματος χρησιμοποιώντας το framework **Phoenix LiveView**. Το LiveView επιτρέπει real-time ενημέρωση της σελίδας χωρίς JavaScript ή WebSocket κώδικα από τον προγραμματιστή. Κάθε αλλαγή κατάστασης στον server μεταφέρεται αυτόματα στον browser μέσω WebSocket σύνδεση. Η εφαρμογή αποτελείται από τέσσερις LiveView σελίδες, οργανωμένες σε κοινό layout με sidebar πλοήγησης:

/sensor/detail: Real-time CO και θερμοκρασία με γραφήματα

/sensor/volts: Τάσεις αισθητήρα (V_{ref} , V_{sensor})

/sensor/history: Ιστορικό 2 ωρών

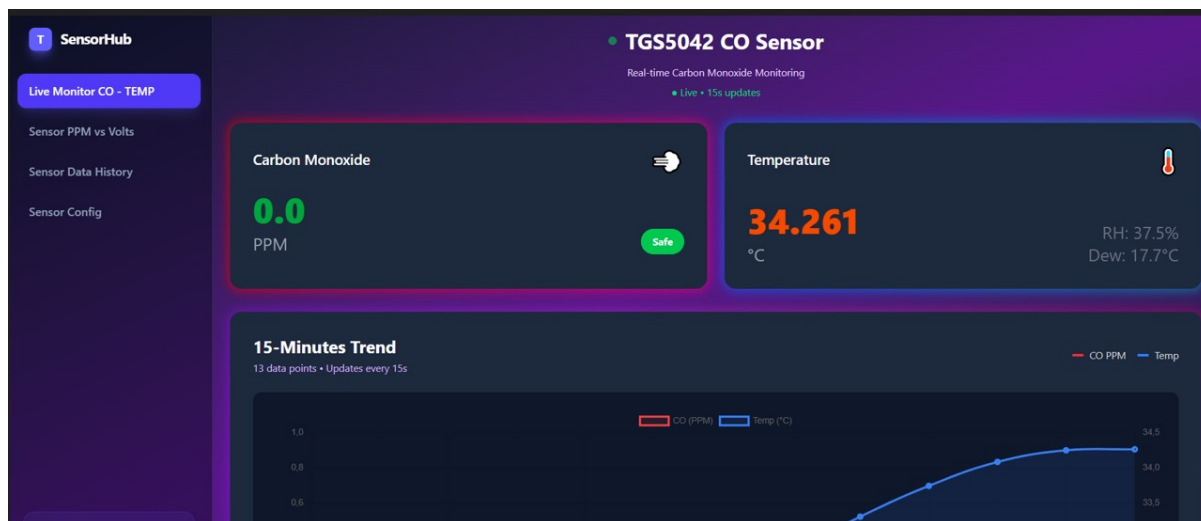
/sensor/offset : Βαθμονόμηση offset αισθητήρα

4.2.6.2 Κύρια Σελίδα Παρακολούθησης — SensorDetailLive

Η σελίδα /sensor/detail αποτελεί την κύρια οθόνη του συστήματος. Κατά τη σύνδεση του browser (mount), εγκαθίστανται δύο ανεξάρτητοι χρονοδιακόπτες με `timer.send_interval`: ένας κάθε 15 δευτερόλεπτα για την τρέχουσα μέτρηση και ένας για την ανανέωση των γραφημάτων. Η χρήση δύο ανεξάρτητων timers επιτρέπει διαφορετικούς ρυθμούς ανανέωσης αν χρειαστεί στο μέλλον.

Κάθε ανανέωση διαβάζει από τον Agent και ενημερώνει το socket με `assign`, πυροδοτώντας αυτόματα την επανασχεδίαση μόνο των τμημάτων της σελίδας που άλλαξαν — αυτό είναι το βασικό πλεονέκτημα του LiveView έναντι κλασικού polling με JavaScript.

Η σελίδα εμφανίζει δύο κάρτες για CO (σε PPM) και θερμοκρασία (με υγρασία και dew point), καθώς και δύο γραφήματα ιστορικού: ένα για τα τελευταία 5 λεπτά και ένα για τα τελευταία 15 λεπτά(εικόνα 47)



Εικόνα 47 Κύρια Σελίδα Παρακολούθησης

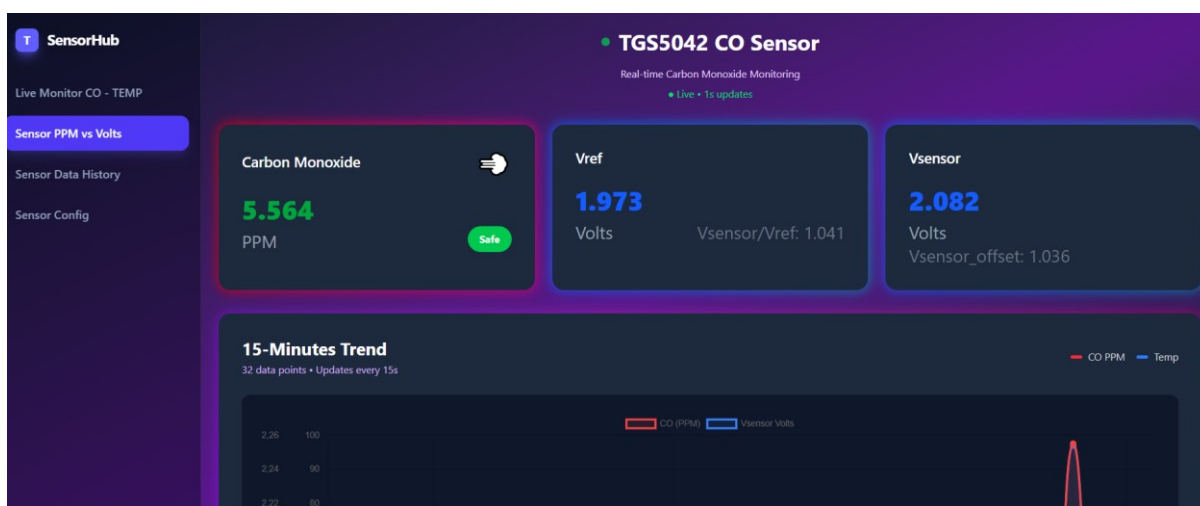
4.2.6.3 Σελίδα Τάσεων — SensorVoltsLive

Η σελίδα /sensor/volts παρέχει πρόσβαση στα σήματα τάσης του κυκλώματος, χρήσιμη κατά τη βαθμονόμηση και την αποσφαλμάτωση. Εμφανίζει σε πραγματικό χρόνο την τάση αναφοράς V_{ref} , την κανονικοποιημένη τάση αισθητήρα V_{sensor} , τον διαφορικό λόγο V_{sensor}/V_{ref} και την τρέχουσα παράμετρο $vsensor_offset$. (εικόνες 48 & 49)



Εικόνα 48 Σελίδα Τάσεων

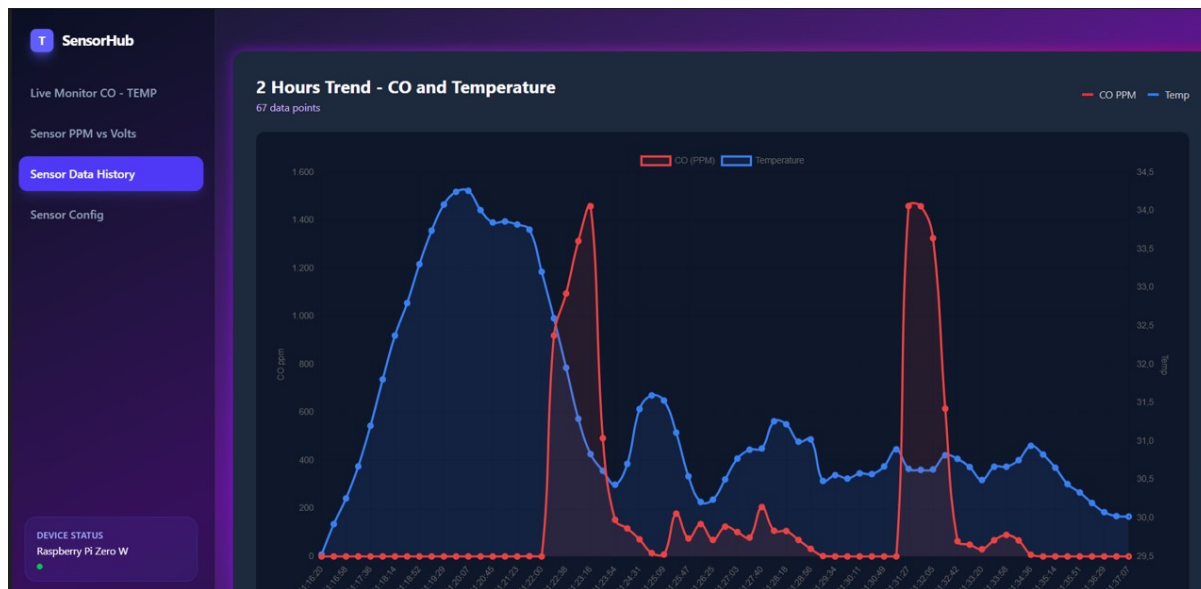
Πιο αναλυτικά, στην παρακάτω εικόνα (εικόνα 49), βλέπουμε πως στην καρτέλα Vref φαίνεται η τρέχουσα τιμή Vref και ο τρέχον λόγος Vsensor/Vref, δηλαδή $2.083 / 1.973 = 1.041$. Στην καρτέλα Vsensor εμφανίζεται η τρέχουσα τιμή της εξόδου του τελεστικού ενισχυτή που είναι 2.082. Στην ίδια καρτέλα εμφανίζεται η τιμή Vsensor_offset που είναι σταθερή και είναι αποθηκευμένη.



Εικόνα 49 Λόγος Vsensor_offset

4.2.6.4 Σελίδα Ιστορικού — SensorHistoryLive

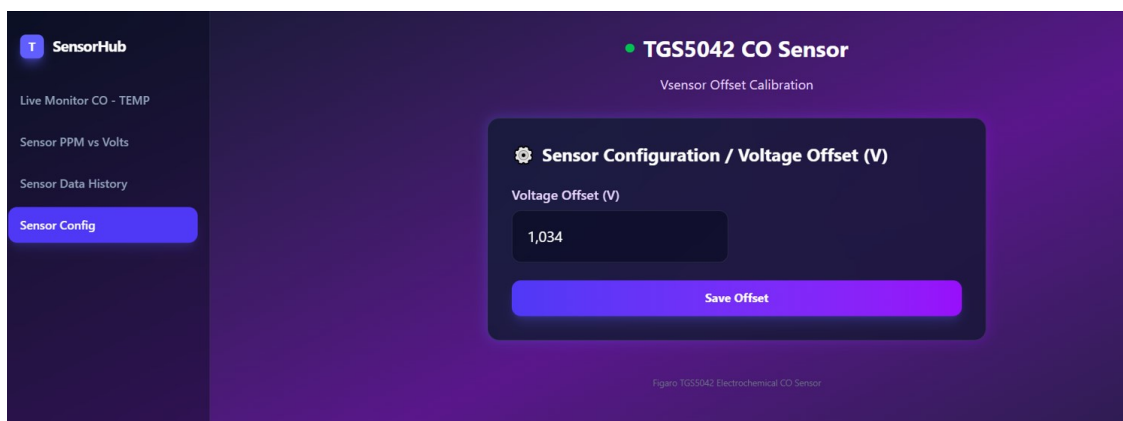
Η σελίδα `/sensor/history` εμφανίζει γράφημα ιστορικού 2 ωρών για CO και θερμοκρασία. Τα δεδομένα ανακτώνται από τον `GasSensor.History`



Εικόνα 50. Σελίδα Ιστορικού

4.2.6.5 Σελίδα Βαθμονόμησης

Η σελίδα `/sensor/offset` επιτρέπει στον χρήστη να ορίσει την παράμετρο `vsensor_offset`, που αντιστοιχεί στον λόγο V_{out}/V_{ref} σε καθαρό αέρα. Η διαδικασία βαθμονόμησης απαιτεί απλά να παρατηρηθεί η τιμή V_{sensor}/V_{ref} (εικόνα 49) σε συνθήκες καθαρού αέρα και να εισαχθεί ως Voltage Offset (εικόνα 51).



Εικόνα 51. Σελίδα βαθμονόμησης

5.0 Πειραματική Διαδικασία και Αποτελέσματα

Στο κεφάλαιο αυτό παρουσιάζονται οι πειραματικές δοκιμές που πραγματοποιήθηκαν για την αξιολόγηση της ορθής λειτουργίας, της ταχύτητας απόκρισης και της ακρίβειας του ενσωματωμένου συστήματος ανίχνευσης αερίων. Το σύστημα υποβλήθηκε σε τρία (3) διαφορετικά πειραματικά σενάρια με μεταβαλλόμενες συγκεντρώσεις μονοξειδίου του Άνθρακα (CO):

1. Δοκιμή ελεγχόμενης συγκέντρωσης σε κλειστό περιβάλλον (γυάλα).
2. Δοκιμή εγκυρότητας σε εργαστήριο χημείας.
3. Δοκιμή ανίχνευσης σε πραγματικές συνθήκες ρύπων (εξάτμιση οχήματος).

5.1 Πείραμα 1: Μέτρηση CO από καύση κεριού σε κλειστό χώρο

Στόχος του πρώτου πειράματος ήταν η επαλήθευση της ικανότητας του συστήματος να ανιχνεύει σταδιακή συσσώρευση CO από ατελή καύση σε κλειστό περιβάλλον, καθώς και η επιβεβαίωση της ανεξαρτησίας των μετρήσεων CO από μεταβολές θερμοκρασίας.

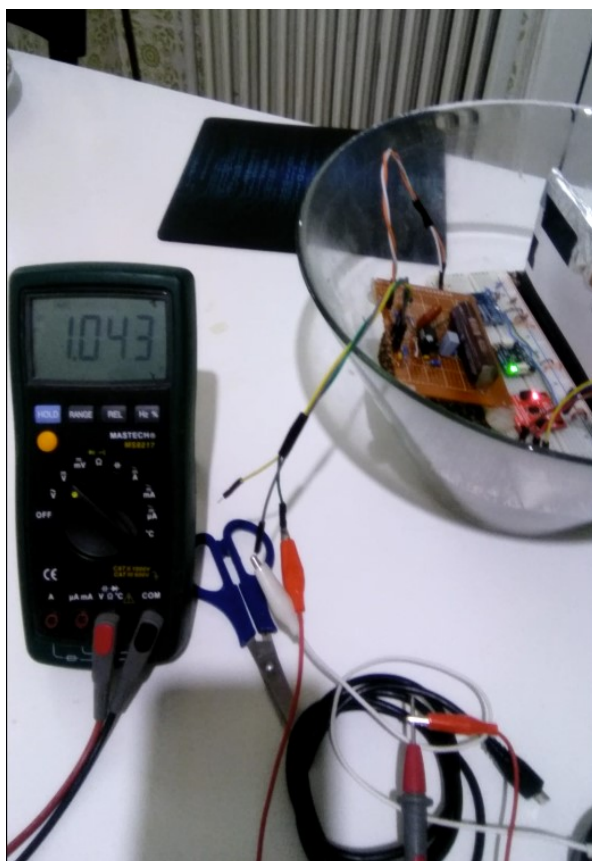
Για τη διεξαγωγή του πειράματος, ο αισθητήρας τοποθετήθηκε εντός κλειστής γυάλας μαζί με ένα αναμμένο κεριό τύπου tea candle. (εικόνα 52). Η κλειστή γεωμετρία του δοχείου εξασφάλισε σταδιακή μείωση του διαθέσιμου οξυγόνου, δημιουργώντας συνθήκες ατελούς

καύσης και προοδευτικής συσσώρευσης CO. Το σύστημα κατέγραφε μετρήσεις ανά 15 δευτερόλεπτα καθ' όλη τη διάρκεια του πειράματος.

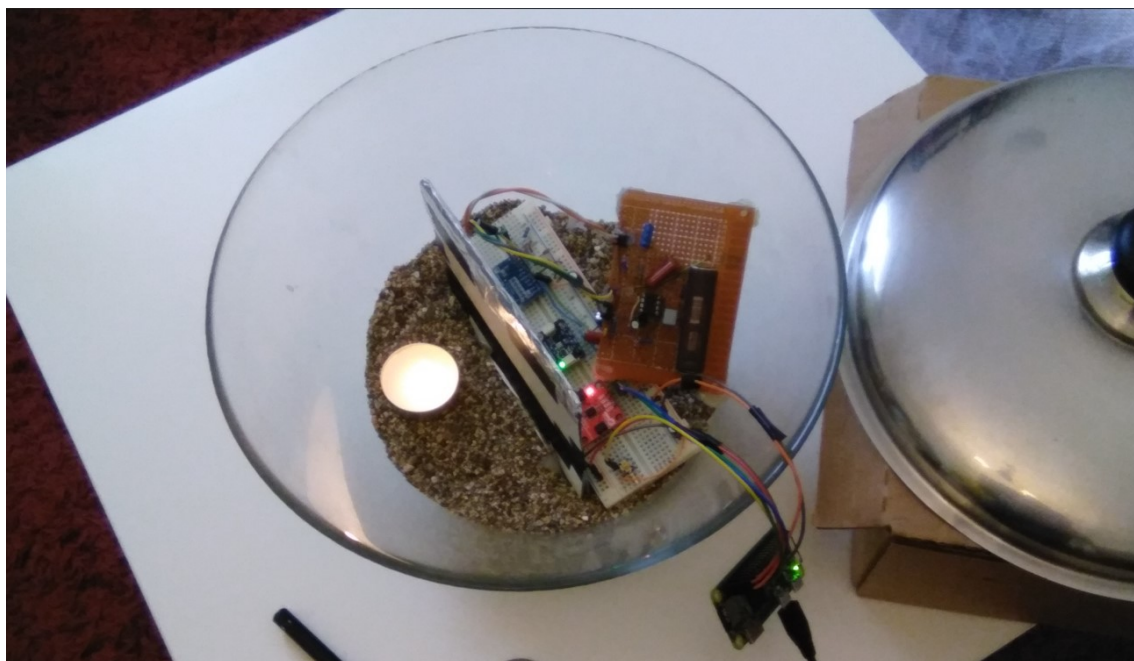
Αρχικά, τοποθετήθηκε ο αισθητήρας χωρίς το κερί και μετρήθηκε η έξοδος του αισθητήρα σε καθαρό αέρα. Όπως φαίνεται και από την εικόνα 52, η παραγόμενη τιμή σε volts είναι 1,043 volts περίπου, ή $1,043 * 2 = 2,086$ volts αφού ξέρουμε πως η έξοδος του αισθητήρα είναι μέσω ενός διαιρέτη τάσης $\frac{1}{2}$. Πραγματοποιήθηκε σύγκριση των μετρήσεων σε κατάσταση καθαρού αέρα με τα αποτελέσματα της θεωρητικής προσομοίωσης που είχε εκτελεστεί στο περιβάλλον TINA, στην ενότητα 3.1.3. Η αρχική προσομοίωση προέβλεπε τάση εξόδου της τάξης των 1.8 V. Ωστόσο, οι πραγματικές μετρήσεις μέσω του μετατροπέα ADS1115 έδειξαν ότι η τάση εξόδου σε συνθήκες καθαρού αέρα σταθεροποιείται στα 2.086 V ($1.043 * 2$), πλησιάζοντας πολύ κοντά στην τάση αναφοράς ($V_{ref} = 2$ V). Η απόκλιση αυτή μεταξύ προσομοίωσης και πραγματικής διάταξης κρίνεται απόλυτα φυσιολογική και αποδίδεται:

1. Στο υπολειμματικό ρεύμα βάσης (baseline current) που παρουσιάζει ο ηλεκτροχημικός αισθητήρας TGS5042 ακόμη και σε καθαρό περιβάλλον.
2. Στην τάση σφάλματος offset (input offset voltage) του πραγματικού τελεστικού ενισχυτή.
3. Στις κατασκευαστικές ανοχές των αντιστάσεων του κυκλώματος.

Στη συνέχεια, παρακολουθήσαμε τον λόγο V_{sensor}/V_{ref} που ανανεώνεται συνεχώς από την web εφαρμογή και το καταχωρήσαμε σαν V_{sensor_offset} . Επομένως, με αυτόν τον τρόπο, κάναμε βαθμονόμηση σε κατάσταση καθαρού αέρα.. Στην συνέχεια, ανάψαμε το κερί και κλείσαμε την γυάλα με μεταλλικό καπάκι Τα αποτελέσματα συγκρίθηκαν με αντίστοιχα ποσοτικά δεδομένα πρόσφατων μελετών για tea candles (Klosterköther et al., 2021)[\[40\]](#), επιβεβαιώνοντας τη σταδιακή συσσώρευση του ρύπου λόγω του περιορισμένου αερισμού του χώρου.

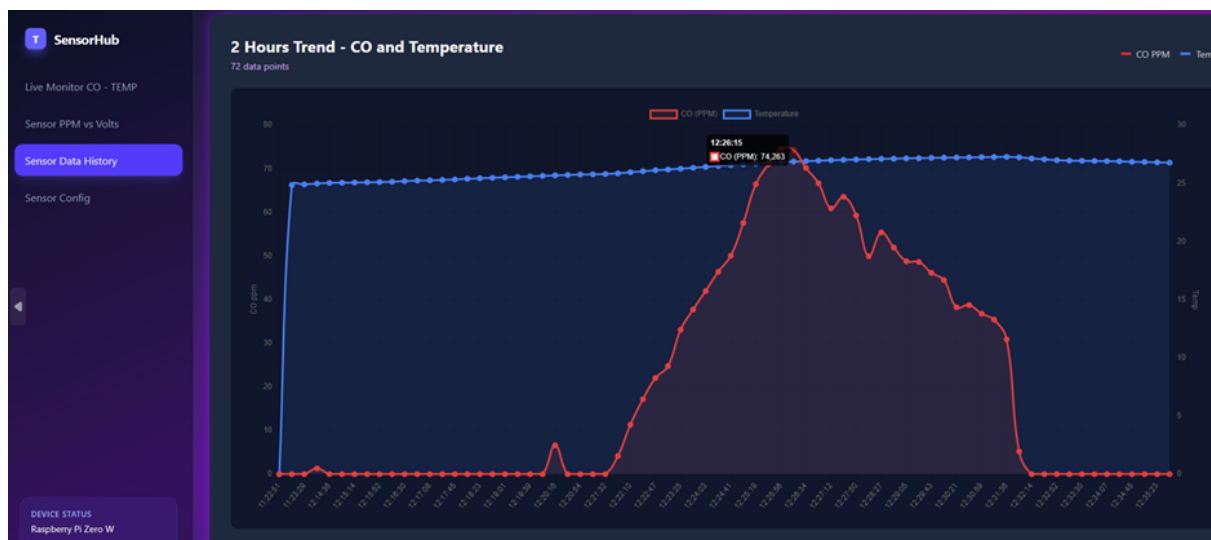


Εικόνα 52 Έξοδος αισθητήρα σε volt σε καθαρό αέρα



Εικόνα 53 Έλεγχος CO σε κλειστή γυάλα

Τα αποτελέσματα, όπως φαίνονται στη εικόνα 54, έδειξαν σταδιακή άνοδο της συγκέντρωσης CO από 0 ppm έως μέγιστο ~70 ppm, ακολουθούμενη από σταδιακή πτώση καθώς το κερί έσβησε λόγω εξάντλησης του οξυγόνου εντός της γυάλας. Μετά το άνοιγμα της γυάλας, ο αισθητήρας επέστρεψε στην τιμή των 0 ppm. Αξιοσημείωτο είναι ότι η θερμοκρασία παρέμεινε σχεδόν αμετάβλητη (~25°C) καθ' όλη τη διάρκεια, επιβεβαιώνοντας ότι οι μετρούμενες μεταβολές CO δεν αποδίδονται σε θερμοκρασιακή απόκλιση αλλά σε πραγματική μεταβολή συγκέντρωσης. Συγκεκριμένα, πρόκειται για 72 data points με περίοδο δειγματοληψίας ενά 5 sec. Επομένως το πείραμα διήρκεσε $72 * 5 = 360 \text{ sec} * (1 \text{ min} / 60 \text{ sec}) = 6 \text{ λεπτά}$.



Εικόνα 54. Έλεγχος CO σε κλειστή γυάλα

Επιπλέον, στην εικόνα 55, το γράφημα επιβεβαιώνει την γραμμική σχέση μεταξύ της τάσης εξόδου Vsensor ή Vout και της υπολογιζόμενης συγκέντρωσης CO σε ppm. Καθώς η τάση του αισθητήρα αυξάνεται λόγω έκθεσης σε CO, η αντίστοιχη τιμή ppm ακολουθεί ισόποσα, επαληθεύοντας την ορθότητα του μαθηματικού μοντέλου μετατροπής Vout→ppm που υλοποιήθηκε στο λογισμικό.



Εικόνα 55. Γράφημα Vout vs CO ppm

5.2 Πείραμα 2: Δοκιμή εγκυρότητας σε εργαστήριο χημείας και αποστολή δεδομένων

Σκοπός του συγκεκριμένου πειράματος ήταν η επικύρωση των μετρήσεων, ο έλεγχος της ταχύτητας απόκρισης του αισθητήρα **TGS5042** καθώς και η αποστολή δεδομένων στο ThingsBoard. Για τον σκοπό αυτό, χρησιμοποιήθηκε φιάλη αποθήκευσης με πιστοποιημένη συγκέντρωση μονοξειδίου του άνθρακα (CO). Η πειραματική διάταξη τοποθετήθηκε εντός της απαγωγού εστίας του εργαστηρίου χημείας (ΕΑΠ) για λόγους ασφαλείας. Επίσης ενεργοποιήθηκε η αποστολή δεδομένων στο ThingsBoard μέσω δικτύου HotSpot με το κινητό τηλέφωνο.

Θα εξηγήσουμε το πείραμα παραθέτοντας εικόνες από όλη την διαδικασία. Η εικόνα 56 δείχνει τον απαγωγό εστίας εκεί που θα γίνει το πείραμά μας.



Εικόνα 56. Απαγωγός εστίας

Φιάλη με πιστοποιημένο αέριο CO μεταφέρθηκε κοντά στον απαγωγό εστίας όπου έχουμε στήσει την διάταξη του IoT κόμβου με το TGS5042. Εικόνα 57.



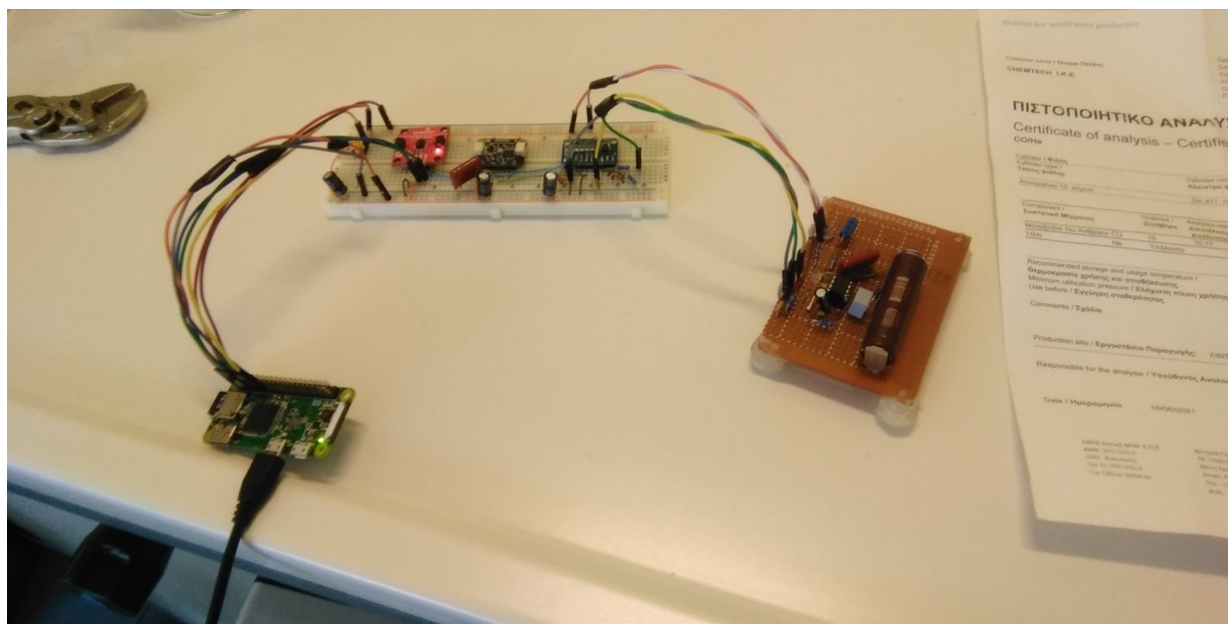
Εικόνα 57. Φιάλη CO με την πειραματική διάταξη

Στην εικόνα 58, βλέπουμε το πιστοποιητικό που συνόδευε της συγκεκριμένη φιάλη. Από τα στοιχεία που φαίνονται υπολογίζουμε πως η περιεκτικότητα της φιάλης σε CO είναι πάνω από 100.000 ppm!

ΠΙΣΤΟΠΟΙΗΤΙΚΟ ΑΝΑΛΥΣΗΣ					
Certificate of analysis – Certified standard					
CO/He					
Cylinder / Φιάλη	Cylinder connection / Κλείστρο φιάλης		Cylinder pressure / Πίεση φιάλης (15°C)	Gas volume / Περιεχόμενο (15°C)	
Cylinder type / Τύπος φιάλης					
Αλουμινένια 10 λίτρων	Din 477, nr. 14		150 bar	1,013 bar	
				1.5 m ³	
Component / Συστατικά Μίγματος	Ordered / Ζητήθηκε	Analysis result / Αποτέλεσμα Ανάλυσης	Uncertainty / Αναλυτική Αβεβαιότητα	Unit / Μονάδα	Method / Μέθοδος Ανάλυσης
Μονοξείδιο του Ανθρακα CO	10	10,17	1% (rel)	% mol	GC
Ήλιο He	Υπόλοιπο				
Recommended storage and usage temperature / Θερμοκρασία χρήσης και αποθήκευσης			-10 to / έως +40 °C		
Minimum utilization pressure / Ελάχιστη πίεση χρήσης			5 bar		
Use before / Εγγύηση σταθερότητας			12 μήνες		
Comments / Σχόλια			Οι τιμές % και ppm εκφράζονται σαν ιδανικά μέρη όγκου (V/V) αναφέρονται σε συνθήκες 1013 mbar, 15°C.		
Production site / Εργοστάσιο Παραγωγής:			ΛΙΝΤΕ ΕΛΛΑΣ ΕΠΕ Εργαστήριο Ειδικών Αερίων & Μιγμάτων Μον		

Εικόνα 58. Πιστοποιητικό για την φιάλη CO

Στην εικόνα 59 βλέπουμε άλλη μια φορά την διάταξη του συστήματος ενώ στη εικόνα 60 βλέπουμε το συνδεδεμένο laptop. Το laptop δίνει απλά ρεύμα στο Raspberry pi μέσω USB καθώς η ασύρματη δικτύωση έχει συντελεστεί μέσω κινητού hotspot.



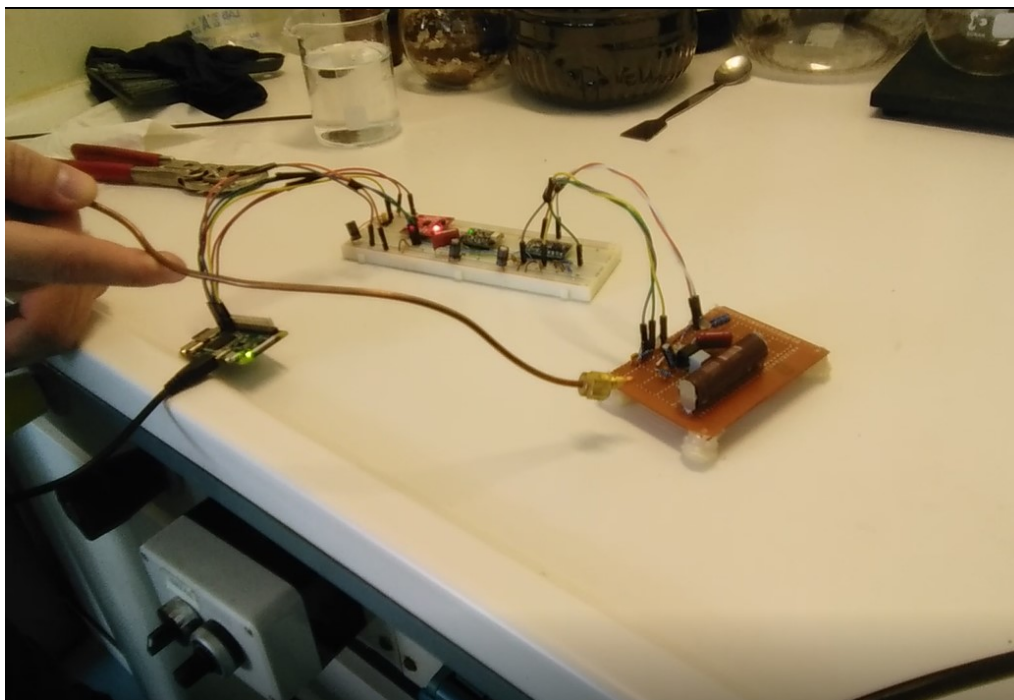
Εικόνα 59. Διάταξη συστήματος στον απαγωγό



Εικόνα 60. Laptop για παροχή τροφοδοσίας και εμφάνιση μετρήσεων

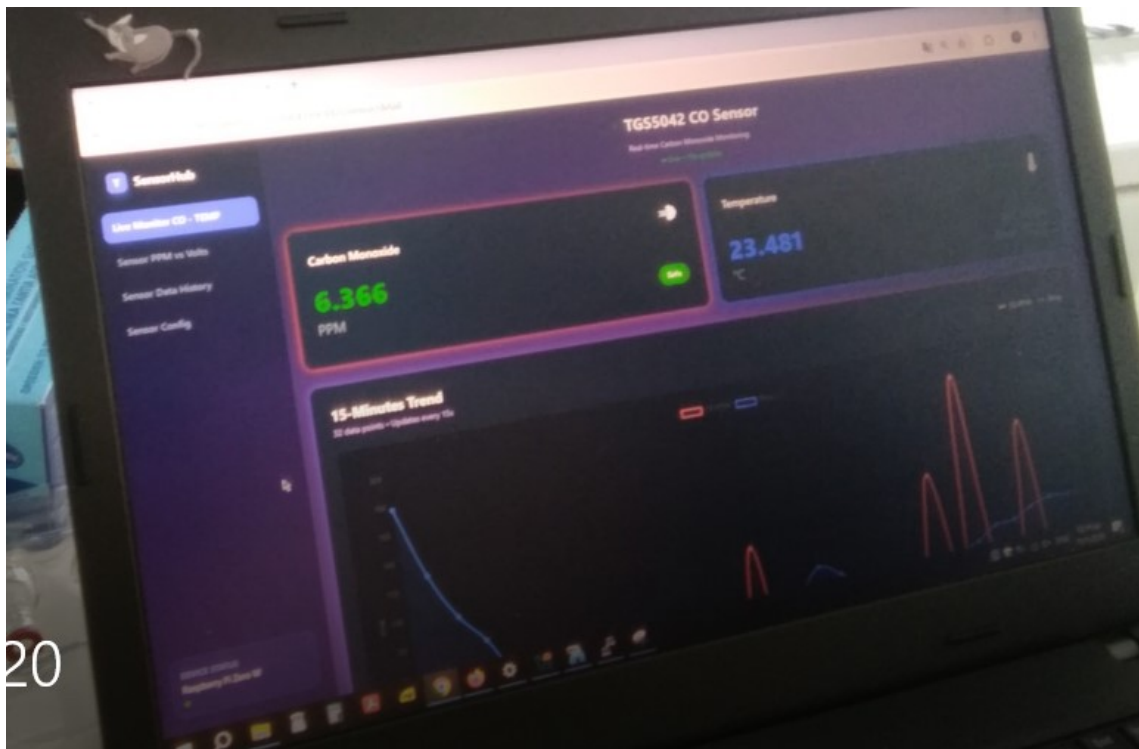
Μέσω ειδικού ρυθμιστή ροής, διοχετεύθηκε αέριο από τη φιάλη απευθείας στον αισθητήρα(εικόνα 61). Καταγράφηκε η δυναμική συμπεριφορά του συστήματος κατά την

εισαγωγή του αερίου η σταθεροποίηση της μέτρησης καθώς και ο χρόνος επαναφοράς του αισθητήρα μόλις διακόπηκε η παροχή και λειτούργησε ο εξαερισμός της εστίας.



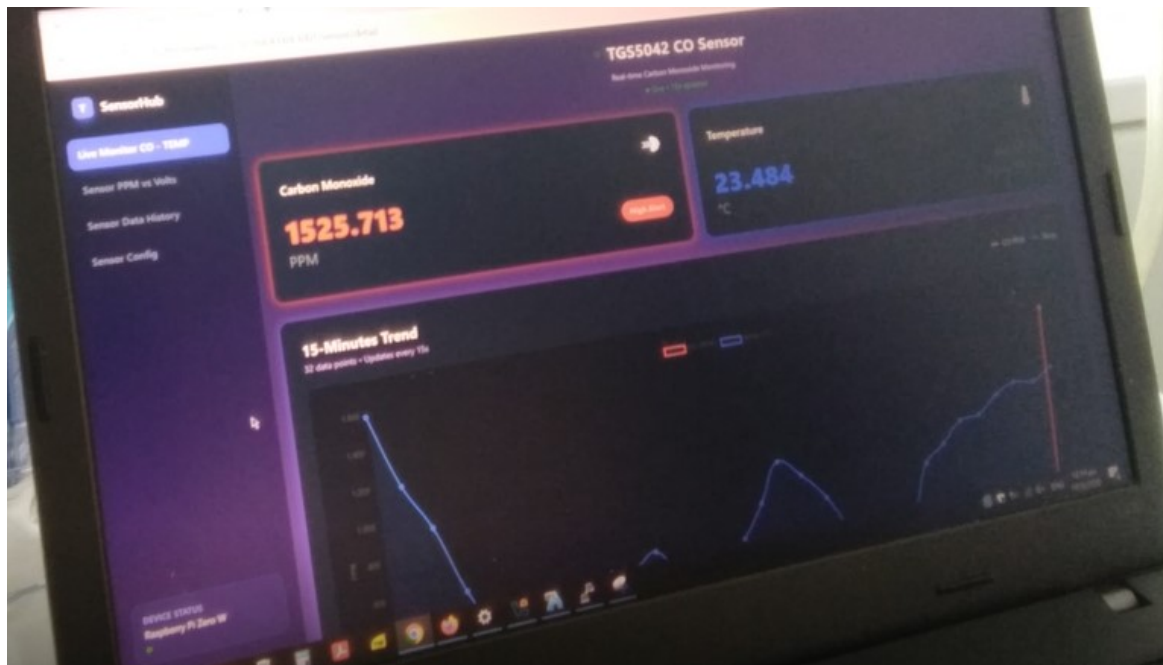
Εικόνα 61. Διοχέτευση CO στον TGS5042

Στις επόμενες εικόνες φαίνεται η άμεση απόκριση του συστήματος. Αφού έγινε διοχέτευση του αερίου στο αισθητήρα(εικόνα 61), στα αμέσως επόμενα 15 δευτερόλεπτα οι μετρήσεις εμφανίστηκαν σε πραγματικό χρόνο (εικόνα 62).



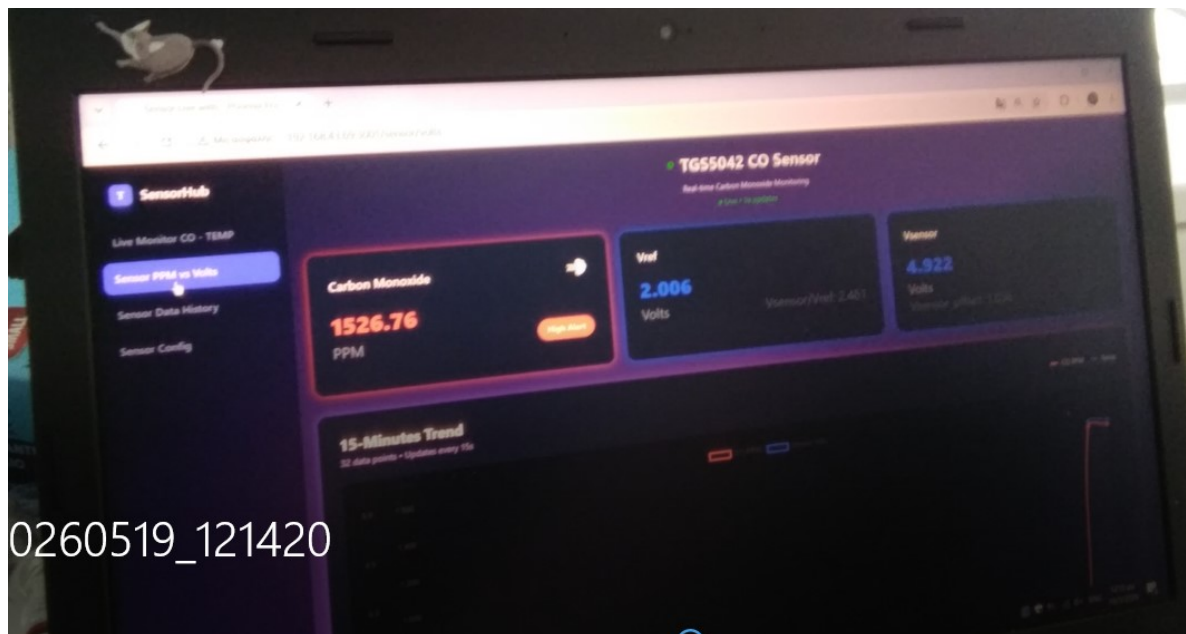
Εικόνα 62 Διεπαφή web με μέτρηση CO

Βλέπουμε πως το σύστημα μπόρεσε και μέτρησε τιμές μέχρι και 1500ppm CO. (εικόνα 63)



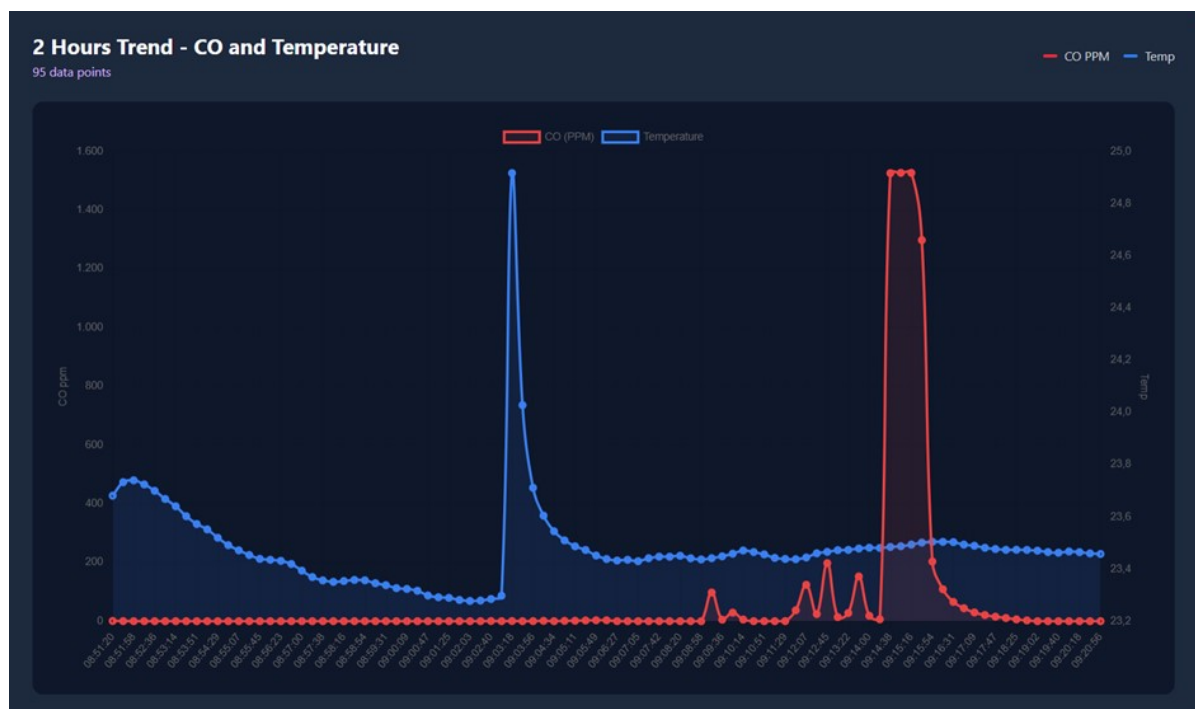
Εικόνα 63. Μέτρηση ποσότητας πάνω από 1500ppm

Στην εικόνα 64, βλέπουμε πως η τάση που παράχθηκε από αυτή την ποσότητα του CO έφτασε στα **4.922** volts. Όπως γνωρίζουμε, αυτή η τιμή είναι προϊόν του πρώτου σταδίου του αναλογικού κυκλώματος και η οποία θα ήταν καταστροφική για το ads1115 αν προσπαθούσαμε να την ψηφιοποιήσουμε. Επομένως, ήταν σωστό που εγκαταστήσαμε μια διάταξη διαιρέτη τάσης στην έξοδο του πρώτου σταδίου.



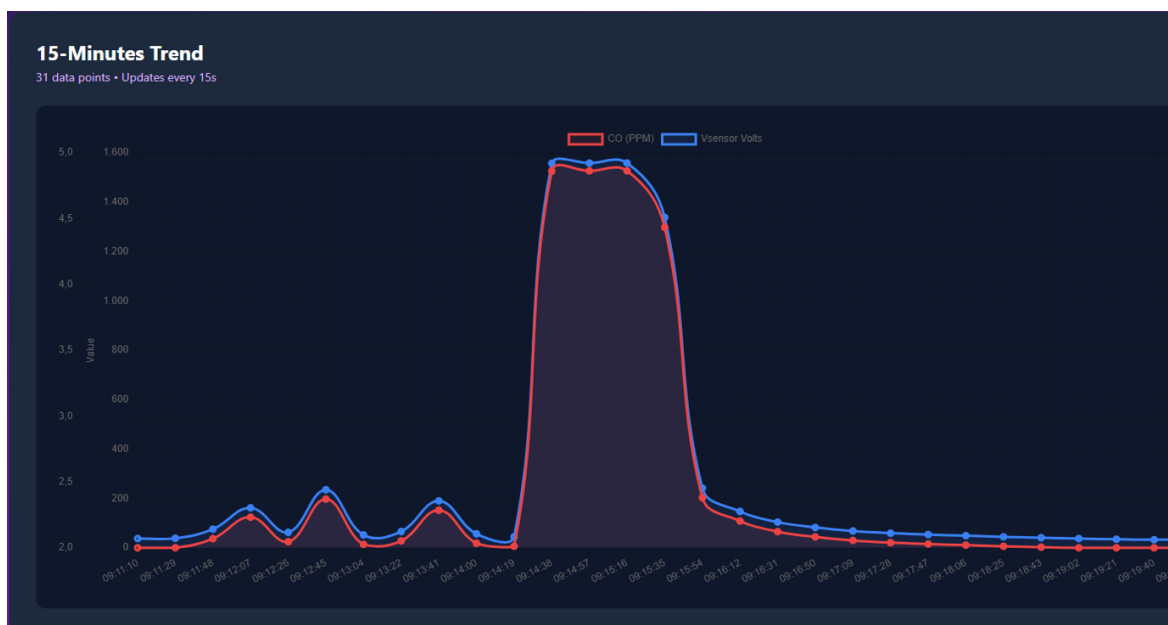
Εικόνα 64. Οριακή τιμή Vsensor

Επιπλέον παραθέτουμε το ιστορικό της μέτρησης όπως φαίνεται στην εικόνα 65. Όπως φαίνεται, το εύρος χρόνου είναι περίπου 8 λεπτά και η μέτρηση του CO δεν επηρεάζεται από την θερμοκρασία. Η αιχμή της θερμοκρασίας όπως φαίνεται, πάντως, προκλήθηκε από τον γράφοντα φυσώντας πάνω από τον αισθητήρα BME680 για μερικά δευτερόλεπτα.



Εικόνα 65 CO vs Θερμοκρασία

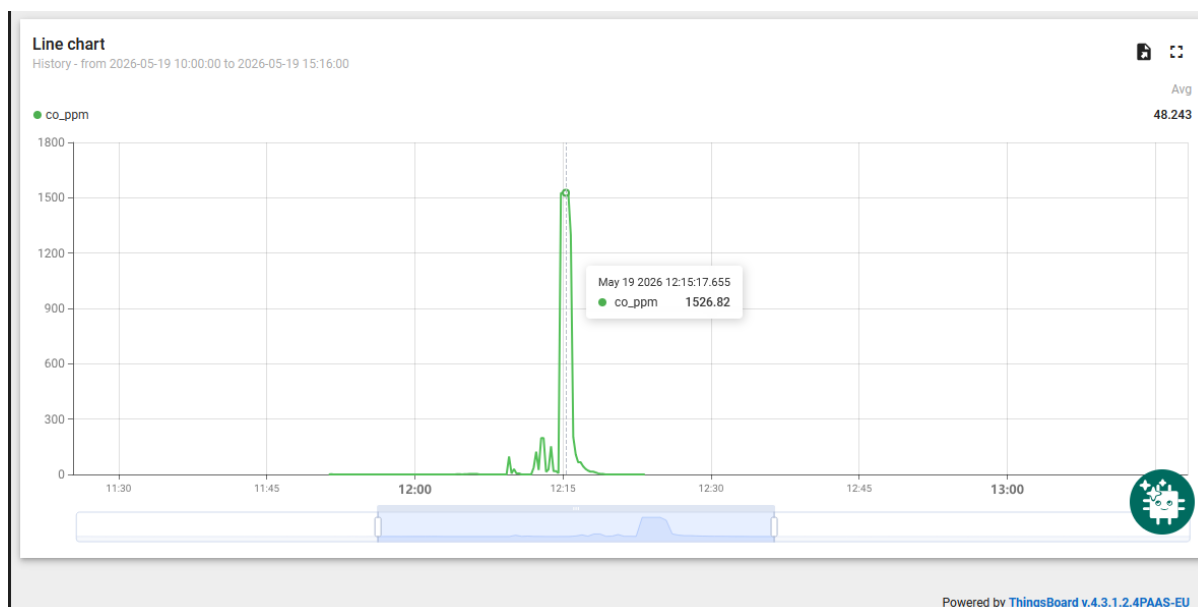
Στην εικόνα 66, αποδεικνύεται γιατί δείχνει ακριβώς αυτό που περιγράψαμε προηγουμένως: την απότομη εισαγωγή του αερίου (CO), τον κορεσμό/σταθεροποίηση και τη γρήγορη επαναφορά του αισθητήρα. Την ίδια ακριβώς ιδιότητα καταγράψαμε και με το πείραμα στην γυάλα.



Εικόνα 66. CO vs Volts

Τέλος, το γράφημα της εικόνας 67 προέρχεται από το περιβάλλον του ThingsBoard IoT Cloud Platform. Στο συγκεκριμένο Dashboard επιλέχθηκε η γραφική απεικόνιση αποκλειστικά των τιμών συγκέντρωσης CO (σε PPM).

Παρατηρώντας τη γραφική παράσταση, γίνεται άμεσα αντιληπτό ότι είναι απόλυτα όμοια με την αντίστοιχη τοπική καταγραφή του gas_sensor_web (Εικόνα 67). Η ταύτιση αυτή των δύο καμπυλών είναι ιδιαίτερα κρίσιμη, καθώς επιβεβαιώνει την αξιοπιστία του υποσυστήματος τηλεμετρίας της εφαρμογής gas_sensor. Αποδεικνύεται έτσι ότι η αποστολή των δεδομένων μέσω του πρωτοκόλλου MQTT πραγματοποιείται σε πραγματικό χρόνο με μηδενική απώλεια πακέτων ακόμη και κατά τις φάσεις των απότομων μεταβολών και αιχμών (peaks) του αερίου. Το σύστημα, επομένως, κρίνεται απόλυτα επιτυχές ως προς την ικανότητα απομακρυσμένης εποπτείας και ιστορικής τήρησης περιβαλλοντικών δεδομένων.



Εικόνα 67. Γραφική Παράσταση ThingsBoard με δεδομένα CO

5.3 Πείραμα 3: Μέτρηση Ρύπων σε Εξάτμιση Οχήματος σε πραγματικές Συνθήκες

Σκοπός του τρίτου και τελευταίου πειράματος ήταν η αξιολόγηση της ανθεκτικότητας και της ταχύτητας απόκρισης του ενσωματωμένου συστήματος κάτω από επιθετικές συνθήκες, με υψηλές συγκεντρώσεις καυσαερίων και απότομες θερμοκρασιακές μεταβολές. Η διάταξη τοποθετήθηκε σε ασφαλή απόσταση από την έξοδο της εξάτμισης ενός συμβατικού οχήματος σε εξωτερικό, ανοιχτό χώρο(εικόνες 68 & 69), κοντά στην εξάτμιση γύρω στα 30-50 εκατοστά.



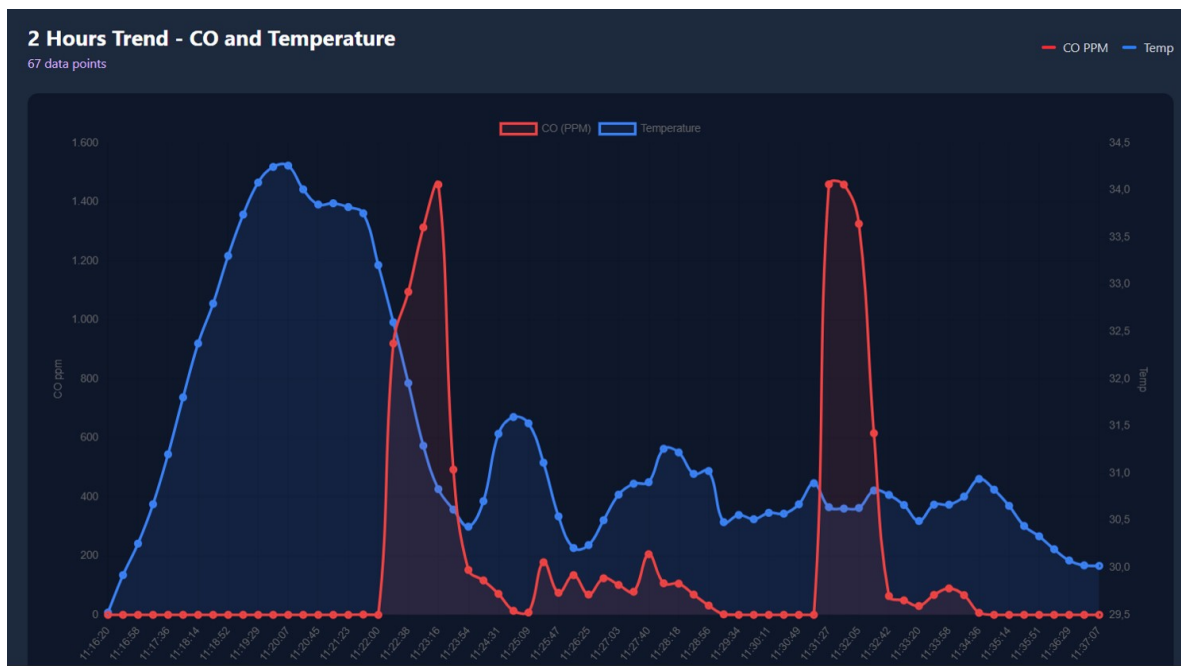
Εικόνα 68. Διάταξη εξωτερικού χώρου για μέτρηση CO



Εικόνα 69 Απόσταση από την εξάτμιση στα 50 εκατοστά

Το πρόγραμμα τέθηκε πάλι σε λειτουργία με ρυθμό δειγματοληψίας 15 δευτερολέπτων, καταγράφοντας τις τιμές μέσω του αισθητήρα TGS5042. Η διασύνδεση έγινε πάλι μέσω του mobile hotspot στέλνοντας πάλι δεδομένα στο ThingsBoard. Το πείραμα χωρίστηκε σε τρεις διακριτές φάσεις:

1. Εκκίνηση του κινητήρα και λειτουργία στο ρελαντί για τη δημιουργία μιας σταθερής τιμής βάσης.
2. Σβήσιμο της μηχανής και εκκίνηση του κινητήρα πάλι.
3. Απενεργοποίηση του κινητήρα για την καταγραφή του χρόνου αποκατάστασης του αισθητήρα.

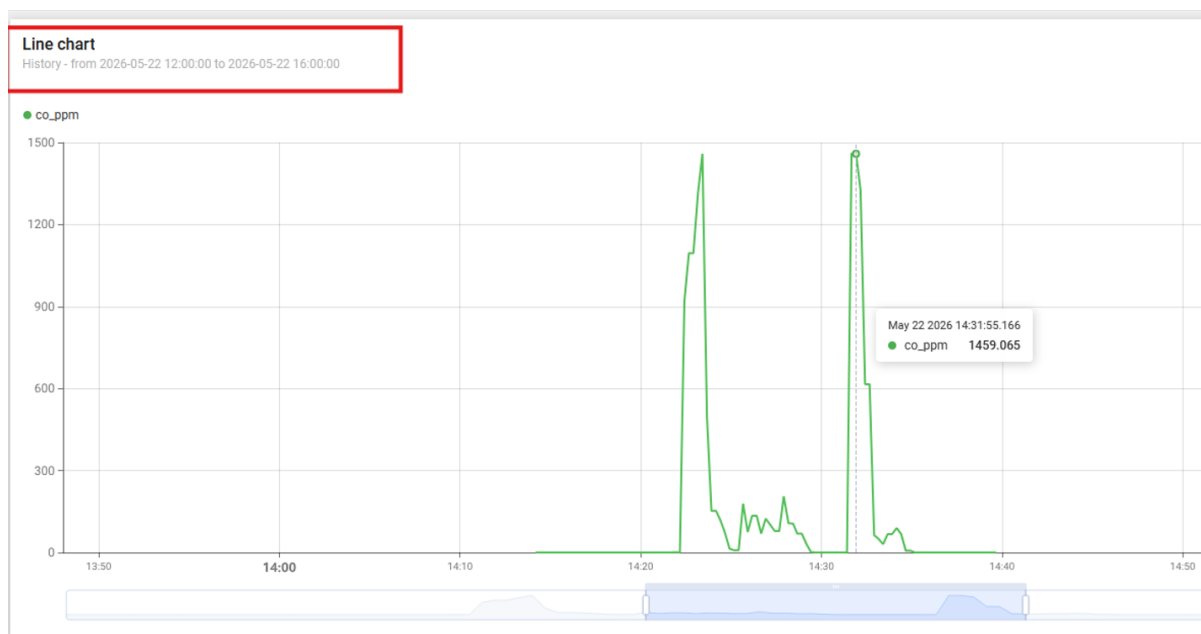


Εικόνα 70. Καταγραφή CO στην εξάτμιση.

Όπως αποτυπώνεται στο αντίστοιχο γράφημα τηλεμετρίας (εικόνα 70), το σύστημα ανταποκρίθηκε ακαριαία στις αλλαγές λειτουργίας του κινητήρα. Η αυξημένη θερμοκρασία στην αρχή προήλθε από το γεγονός πως ο αισθητήρας θερμοκρασίας ήταν σε έκθεση στον ήλιο. Αργότερα, όπως φαίνεται και από την γραφική παράσταση, μετακινήθηκε πιο κοντά στην εξάτμιση που ήταν στην σκιά.

Η συγκέντρωση του αερίου κατέγραψε μια απότομη, κατακόρυφη άνοδο αγγίζοντας μια μέγιστη τιμή στην περιοχή των +1.500 PPM. Η τιμή αυτή κρίνεται απόλυτα αναμενόμενη και συμβαδίζει με τη διεθνή βιβλιογραφία (Heywood, 1988)^[48], η οποία επιβεβαιώνει ότι κατά τη διάρκεια μεταβατικών φαινομένων καύσης με πλούσιο μείγμα, οι αιχμές του CO στα ακατέργαστα καυσαέρια μπορούν να κυμανθούν από 1.000 έως και άνω των 5.000 PPM, προτού ο καταλυτικός μετατροπέας φτάσει σε πλήρη θερμοκρασία λειτουργίας. Παράλληλα, η εφαρμογή Phoenix (gas_sensor_web) μετέδιδε τις αλλαγές ζωντανά στο UI ενώ η εφαρμογή gas_sensor έστελνε δεδομένα στο ThingsBoard χωρίς καμία απώλεια δεδομένων, αποδεικνύοντας τη σταθερότητα της αρχιτεκτονικής OTP κάτω από συνθήκες πίεσης.

Τέλος, όπως και στο πείραμα 2, έγινε επιτυχημένη αποστολή δεδομένων στο ThingsBoard (εικόνα 71), ενώ στην εικόνα 72 φαίνονται τα τελευταία δεδομένα που καταχωρήθηκαν στην πλατφόρμα και τα οποία αντιπροσωπεύουν δεδομένα του συγκεκριμένου πειράματος.



Εικόνα 71. ThingsBoard Κονσόλα για CO – αυτοκίνητο

Telemetry	Key	Value
☐	Last update time	
☐	2026-05-22 14:39:40	co_ppm
☐	2026-05-22 14:39:40	cpu_temperature
☐	2026-05-22 14:39:40	dew_point_c
☐	2026-05-22 14:39:40	error_message
☐	2026-05-22 14:39:40	gas_resistance_ohms
☐	2026-05-22 14:39:40	humidity_rh
☐	2026-05-22 14:39:40	pressure_pa
☐	2026-05-22 14:39:40	status
☐	2026-05-22 14:39:40	temperature_c
☐	2026-05-22 14:39:40	time_reliable
☐	2026-05-22 14:39:40	timestamp
☐	2026-05-22 14:39:40	vdifferential
☐	2026-05-22 14:39:40	vref
☐	2026-05-22 14:39:40	vref_variance
☐	2026-05-22 14:39:40	vsensor

Εικόνα 72. Δεδομένα στην κονσόλα ThingsBoard

5.4 Περιορισμοί Πειραματικής Διαδικασίας και Ακρίβειας

Θα πρέπει να επισημανθεί ότι η παρούσα εργασία εστιάζει στην ανάπτυξη, την αρχιτεκτονική και τη λειτουργική σκοπιμότητα (feasibility) ενός low-cost, open-source IoT συστήματος. Καθώς οι δοκιμές δεν πραγματοποιήθηκαν σε ταυτόχρονη σύγκριση με κάποιο, πιστοποιημένο όργανο ανάγνωσης CO τιμών, οι απόλυτες τιμές των PPM που καταγράφηκαν ενδέχεται να παρουσιάζουν αποκλίσεις από την πραγματικότητα.

Ωστόσο, η πειραματική διαδικασία πέτυχε πλήρως τον κύριο στόχο της, ο οποίος ήταν η επαλήθευση της δυναμικής συμπεριφοράς του αισθητήρα TGS5042 (άμεση ανίχνευση τάσεων), η επιβεβαίωση της γραμμικότητας κατά την εισαγωγή αερίου, και η σταθερή, σε πραγματικό χρόνο μετάδοση των μεταβολών στο Phoenix UI και στο ThingsBoard Cloud. Το σύστημα, επομένως, κρίνεται απόλυτα αξιόπιστο για την παροχή περιβαλλοντικής επίγνωσης, ακόμη και αν απαιτείται μελλοντική εργαστηριακή βαθμονόμηση για την επίτευξη απόλυτης ακρίβειας.

5.5 Σύνοψη Πειραματικής Διαδικασίας

Ολοκληρώνοντας, οι τρεις πειραματικές δοκιμές (σε ελεγχόμενο περιβάλλον γυάλας, σε εργαστηριακή απαγωγό εστία με πρότυπο αέριο, και σε πραγματικές συνθήκες έκλυσης καυσαερίων οχήματος) προσέφεραν μια πλήρη εικόνα για τις δυνατότητες της διάταξης.

Ο αισθητήρας TGS5042 επέδειξε εξαιρετική δυναμική απόκριση, γραμμικότητα και σταθερότητα, επιβεβαιώνοντας ότι το open-source hardware μπορεί να αποδώσει με αξιοπιστία ακόμα και σε συνθήκες υψηλής συγκέντρωσης ρύπων (1,500 PPM). Παράλληλα, η αρχιτεκτονική του λογισμικού σε Elixir/Nerves και η τηλεμετρία μέσω MQTT στο ThingsBoard Cloud λειτούργησαν απρόσκοπτα χωρίς καμία απώλεια δεδομένων.

Τα πειραματικά αυτά δεδομένα επαληθεύουν την επιτυχία του προτεινόμενου συστήματος και αποτελούν το θεμέλιο για την τελική αξιολόγηση και τα συμπεράσματα που ακολουθούν στο επόμενο κεφάλαιο.

6. Επίλογος και συμπεράσματα

6.1 Συμπεράσματα

Στο πλαίσιο της παρούσας διπλωματικής εργασίας σχεδιάστηκε, υλοποιήθηκε και αξιολογήθηκε ένας πρότυπος, αυτόνομος σταθμός περιβαλλοντικής παρακολούθησης και ανίχνευσης επικίνδυνων αερίων (CO), βασισμένος στο λειτουργικό σύστημα πραγματικού χρόνου Nerves (Elixir) και σε υλικό Raspberry Pi Zero W.

Η ολοκλήρωση του έργου επέτρεψε την εξαγωγή σημαντικών συμπερασμάτων:

Η επιλογή της Elixir και του προτύπου OTP αποδείχθηκε εξαιρετικά αποτελεσματική για ενσωματωμένα συστήματα. Η χρήση δέντρων επίβλεψης (Supervision Trees) εξασφάλισε ότι το σύστημα παραμένει ανθεκτικό σε σφάλματα απομονώνοντας τυχόν αστοχίες του hardware (π.χ. απώλεια Wi-Fi ή θόρυβος στον δίαυλο I2C) χωρίς να καταρρέει η εφαρμογή.

Επίσης, η υιοθέτηση της αναλογο-μετρικής (ratiometric) μεθόδου επεξεργασίας σήματος, σε συνδυασμό με την ψηφιακή αντιστάθμιση θερμοκρασίας και το φίλτρο διάμεσης τιμής (median filter), εξάλειψε αποτελεσματικά τον ηλεκτρικό θόρυβο και τις διακυμάνσεις της τροφοδοσίας, προσφέροντας σταθερές και αξιόπιστες μετρήσεις PPM για το μονοξείδιο του άνθρακα.

Συνολικά, το σύστημα πέτυχε τη χρυσή τομή ανάμεσα στο χαμηλό κόστος υλικού (low-cost hardware) και την υψηλή αξιοπιστία λογισμικού αποτελώντας μια σταθερή βάση για βιομηχανικές ή οικιακές εφαρμογές IoT.

6.2 Μελλοντικές Επεκτάσεις

Παρά την επιτυχή λειτουργία του πρωτοτύπου, υπάρχουν αρκετές κατευθύνσεις για τη μελλοντική εξέλιξη και αναβάθμιση του συστήματος:

- Ενεργειακή βελτιστοποίηση (Low Power Mode): Για τη δυνατότητα λειτουργίας του σταθμού αποκλειστικά με μπαταρία ή ηλιακό πάνελ, μπορεί να μελετηθεί η μετάβαση

του Raspberry Pi σε καταστάσεις deep sleep μεταξύ των διαστημάτων δειγματοληψίας των 15 δευτερολέπτων.

- Ασφάλεια και κρυπτογράφηση: Παρά το γεγονός πως προμηθευτήκαμε το NervesKey, λόγω χρόνου δεν μπορέσαμε να υλοποιήσουμε το ασφαλές πρωτόκολλο επικοινωνίας με τον ThingsBoard. Επομένως, προτείνουμε την προσθήκη πρωτοκόλλων ασφαλείας (TLS/SSL) κατά την επικοινωνία μέσω MQTT με την πλατφόρμα ThingsBoard, καθώς και υλοποίηση μηχανισμών κρυπτογραφημένης ανταλλαγής δεδομένων (OTA updates) για την ασφαλή αναβάθμιση του υλικολογισμικού (firmware) εξ αποστάσεως.
- Ενσωμάτωση με το Home Assistant: Αν και το σύστημα τηλεμετρίας αναπτύχθηκε ως αυτόνομη (standalone) συσκευή με δικό του Web UI, ο αρχικός σχεδιασμός προέβλεπε την πλήρη ενσωμάτωσή του στο σύστημα αυτοματισμού **Home Assistant**. Η διασύνδεση αυτή προτείνεται να υλοποιηθεί μέσω του πρωτοκόλλου **MQTT** και της δυνατότητας *Home Assistant Discovery*. Με βάση βιβλιοθηκών, η συσκευή θα μπορεί να αυτο-αναγνωρίζεται από το τοπικό δίκτυο του Home Assistant, επιτρέποντας στον χρήστη να ενσωματώσει τις μετρήσεις PPM του CO σε αυτοματισμούς ασφαλείας του σπιτιού (π.χ. ενεργοποίηση εξαερισμού ή σειρήνας σε περίπτωση ανίχνευσης αερίου).
- Η ενσωμάτωση του Livebook (του διαδραστικού περιβάλλοντος σημειωματάρων της Elixir) δεν πραγματοποιήθηκε απευθείας πάνω στο firmware της συσκευής, αλλά ακολουθήθηκε μια εναλλακτική αρχιτεκτονική προσέγγιση. Υπάρχει σχετικός κώδικας σε πειραματική διάταξη στο αντίστοιχο αποθετήριο github αυτής τη διπλωματικής. Συγκεκριμένα, αναπτύχθηκαν διαδραστικά σημειωματάρια (Livebooks) τα οποία εκτελούνται τοπικά στον υπολογιστή του χρήστη και συνδέονται απομακρυσμένα (remote node connection) με τη συσκευή μέσω του δικτύου. Αυτή η προσέγγιση επιτρέπει τη δυναμική ανάλυση και οπτικοποίηση των δεδομένων σε πραγματικό χρόνο, χωρίς να επιβαρύνονται οι περιορισμένοι υπολογιστικοί πόροι και η μνήμη της embedded συσκευής. Η πλήρης αυτοματοποίηση και η ενσωμάτωση αυτής της ροής εργασίας προτείνεται ως άμεση μελλοντική επέκταση του συστήματος.

Τελειώνοντας, το σύστημα που δημιουργήθηκε δεν βαθμονομήθηκε με κανονικό CO του οποίου την περιεκτικότητα την γνωρίζαμε. Επίσης, δεν υπήρχε διαθέσιμο κάποιο πιστοποιημένο, βαθμονομημένο όργανο μέτρησης του εμπορίου στον ίδιο χώρο, ώστε να πραγματοποιηθεί παράλληλη καταγραφή και σύγκριση των μετρήσεων σε πραγματικό χρόνο.

Λόγω του υψηλού κόστους των επίσημων βιομηχανικών πιστοποιήσεων (τύπου CE/EN), το οποίο κρίνεται απαγορευτικό για ακαδημαϊκά πρωτότυπα και open-source hardware, η διαδικασία επαλήθευσης της ακρίβειας του συστήματος προτείνεται να γίνει μέσω μιας εσωτερικής εργαστηριακής διάταξης.

Η προτεινόμενη μέθοδος περιλαμβάνει τη χρήση στεγανού θαλάμου ελεγχόμενης ατμόσφαιρας με ενσωματωμένο ανεμιστήρα, όπου η συσκευή θα εκτεθεί σε διαδοχικές συγκεντρώσεις CO. Με τη βοήθεια ενός βαθμονομημένου οργάνου αναφοράς του εμπορίου, θα καταγραφούν οι πραγματικές τιμές PPM έναντι των ψηφιακών βημάτων του ADC, επιτρέποντας την εξαγωγή μιας ακριβούς πειραματικής καμπύλης βαθμονόμησης που θα ενσωματωθεί στο firmware της συσκευής.

Βιβλιογραφία

- [1] <https://www.amazon.com/Best-Sellers-Carbon-Monoxide-Detectors/zgbs/hi/495272>
- [2] <https://www.consumerreports.org/home-garden/smoke-carbon-monoxide-detectors/best-portable-carbon-monoxide-detectors-a4805719130/>
- [3] <https://www.firstalert.com/collections/carbon-monoxide-alarms>
- [4] <https://www.ul.com/thecodeauthority/knowledge/carbon-monoxide-alarm-codes>
- [5] <https://www.siemens.com/en-us/products/fire-detection/detection-devices/>
- [6] <https://sdgs.un.org/goals>
- [7] (J, 2010), Krumm, Ubiquitous Computing Fundamentals.
- [8] Anantha Datta Dhruva, Sarath Babu, Abhishek Chakraborty, et al. Computing Boards for Internet of Things: A Comparative Survey. *TechRxiv*. 20 January 2022.
doi: <https://doi.org/10.36227/techrxiv.18517235.v1>
- [9] E. Rohadi *et al.*, "A Review: Performance Analysis of Single Board Computer Technology as an IoT System," *2025 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*, Bali, Indonesia, 2025, pp. 106-110, doi: 10.1109/IAICT65714.2025.11101682
- [10] Yasin A, Delaney J, Cheng C-T, Pang TY. The Design and Implementation of an IoT Sensor-Based Indoor Air Quality Monitoring System Using Off-the-Shelf Devices. *Applied Sciences*. 2022; 12(19):9450. <https://doi.org/10.3390/app12199450>
- [11] Almaaitah T, Joksimovic D, Sajin T. Real-Time IoT-Enabled Water Management for Rooftop Urban Agriculture Using Commercial Off-the-Shelf Products. *Chemistry Proceedings*. 2022; 10(1):34. <https://doi.org/10.3390/IOCAG2022-12235>
- [12] M. Farahani, M. A. Rashid and B. Safaei, "From Kernel to Cloud: A Concise Comparative Study of Practical IoT Operating Systems," in *IEEE Internet of Things Magazine*, vol. 8, no. 1, pp. 92-100, January 2025, doi: 10.1109/IOTM.001.2400090.
- [13] Armstrong, J., 2003. *Making reliable distributed systems in the presence of software errors*. Ph.D. dissertation, Royal Institute of Technology, Stockholm, Sweden.
- [14] Elixir Language <https://elixir-lang.org/>

- [15] G. Fedrecheski, L. C. P. Costa and M. K. Zuffo, "Elixir programming language evaluation for IoT," *2016 IEEE International Symposium on Consumer Electronics (ISCE)*, Sao Paulo, Brazil, 2016, pp. 105-106, doi: 10.1109/ISCE.2016.7797392.
- [16] R. Pompa, N. Fiorini, G. Scacco and C. D'Elia, "A Resources Optimization Method in an Elixir IoT Cloud Environment Based on Basic Operations Performance Measurements," *2024 IEEE International Symposium on Measurements & Networking (M&N)*, Rome, Italy, 2024, pp. 1-6, doi: 10.1109/MN60932.2024.10615383.
- [17] https://www.erlang.org/doc/system/design_principles.html
- [18] https://www.erlang.org/doc/system/gen_server_concepts.html
- [19] <https://nerves-project.org/>
- [20] <https://buildroot.org/>
- [21] <https://github.com/nerves-project/erlinit>
- [22] <https://github.com/nerves-project/nerves#hardware>
- [23] <https://hexdocs.pm/nerves/advanced-configuration.html#partitions>
- [24] Townsend CL, Maynard RL, Effects on health of prolonged exposure to low concentrations of carbon monoxide - Occupational and Environmental Medicine 2002;59:708-711.
- [25] Goldstein M, Carbon Monoxide Poisoning, Journal of Emergency Nursing, 2008; 34, 538-542
- [26] tgs5042_technical information(fusa)_rev16, tgs5042_application note(en)_rev03
TECHNICAL INFORMATION FOR TGS5042, Revised 02/23,
<https://www.figarosensor.com/product/entry/tgs5042-a00.html#an>
- [27] <https://www.adafruit.com/product/3660>
- [28] <https://nettop.gr/index.php/hlektronika/ads1115-i2c-16-bit-adc-4-channel-module.html>
- [29] Sedra, A. S., & Smith, K. C. (2004). *Microelectronic circuits* (5th ed.). Oxford University Press
- [30] https://github.com/nerves-hub/nerves_key
- [31] <https://www.analog.com/en/resources/reference-designs/circuits-from-the-lab/CN0234.html#rd-description>
- [32] <https://www.alphasense.com/products/view-by-target-gas/co-a1>
- [33] <https://www.analog.com/media/en/training-seminars/tutorials/MT-101.pdf>

- [34] Texas Instruments. (2018). *ADS1115 Ultra-Small, Low-Power, 16-Bit Analog-to-Digital Converter with Internal Reference* (Rev. B) [Data sheet].
<https://www.ti.com/lit/ds/symlink/ads1115.pdf>
- [35] Hayes, T. C., & Horowitz, P. (1999). *The art of Electronics* (Vol. 3). Cambridge, UK: Cambridge University Press.
- [36] pinout.xyz, "Raspberry Pi GPIO Pinout", Διαδραστικός Οδηγός Τεκμηρίωσης Υλικού. Διαθέσιμο διαδικτυακά: <https://pinout.xyz>
- [37] Ossher, H., & Tarr, P. (2001). Using multidimensional separation of concerns to (re) shape evolving software. *Communications of the ACM*, 44(10), 43-50.
- [38] <https://elixir-lang.org/getting-started/mix-otp/dependencies-and-umbrella-projects.html>
- [39] <https://elixir.hexdocs.pm/Application.html>
- [40] Klosterkötter, A., Kurtenbach, R., Wiesen, P., & Kleffmann, J. (2021). Determination of the emission indices for NO, NO₂, HONO, HCHO, CO, and particles emitted from candles. *Indoor Air*, 31(1), 116-127.
- [41] nerves-networking. (2024). *vintage_net_wizard* [Software]. GitHub.
https://github.com/nerves-networking/vintage_net_wizard
- [42] Kokkos, N. (2026). *CO_monitoring_with_nerves* [Computer software]. GitHub.
https://github.com/nkokkos/CO_monitoring_with_nerves
- [43] Nerves Project. (2026). *Nerves installation: Linux*. HexDocs.
<https://nerves.hexdocs.pm/installation.html#linux>
- [44] Nerves Project (2026), Getting Started, <https://nerves.hexdocs.pm/getting-started.html>
- [45] Nerves Project (2026), Examples, Poncho Project https://github.com/nerves-project/nerves_examples/tree/main/poncho_phoenix.
- [46] Kokkos, N. (2026). *CO_monitoring_with_nerves*, (GasSensor.Sensor)
https://github.com/nkokkos/CO_monitoring_with_nerves/blob/main/gas_sensor/lib/gas_sensor/sensor.ex
- [47] Litwhiler, D., & Lombardi, B. (2009, June). Testing of carbon monoxide and carbon dioxide sensors with simple apparatus in an engineering education laboratory. In *2009 Annual Conference & Exposition* (pp. 14-1175).
- [48] Heywood, J. B. (1988). *Internal Combustion Engine Fundamentals*. McGraw-Hill.

Υπεύθυνη Δήλωση Συγγραφέα:

Δηλώνω ρητά ότι, σύμφωνα με το άρθρο 8 του Ν.1599/1986, η παρούσα εργασία αποτελεί αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλει κάθε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον και πληρούν τους κανόνες της επιστημονικής παράθεσης.