Διεθνές Πανεπιστήμιο της Ελλάδος
Τμήμα Μηχανικών Πληροφορικής, Υπολογιστών και Τηλεπικοινωνιών

Πρόγραμμα Μεταπτυχιακών Σπουδών στη Ρομποτική

Διπλωματική Εργασία

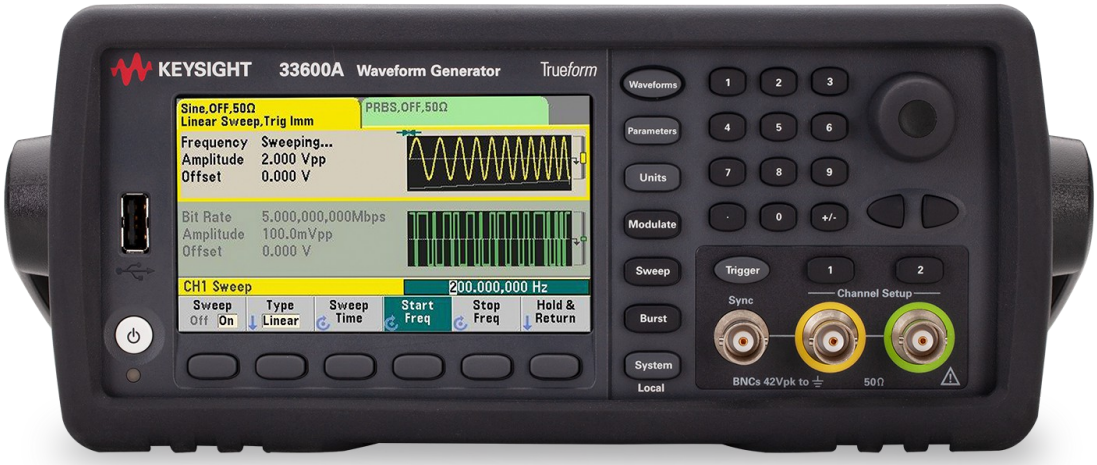
Ανάπτυξη εφαρμογής Γεννήτριας Αυθαίρετων Συναρτήσεων (Arbitrary Waveform Generator) σε πλατφόρμα Arduino

Σεϊταρίδης Πέτρος
Διεθνές πανεπιστήμιο της Ελλάδος

Σέρρες, 27-9-2023

Εργασία που υποβλήθηκε στο Πρόγραμμα Μεταπτυχιακών Σπουδών στη Ρομποτική, του Διεθνούς Πανεπιστημίου της Ελλάδος, για τη μερική εκπλήρωση υποχρεώσεων για το Δίπλωμα Ειδίκευσης στη Ρομποτική

Επιβλέπων καθηγητής: Σπυρίδων Καζαρλής



Περιεχόμενα

Summary

Creating an Arbitrary Waveform Generator involves integrating an Arduino Mega, a Nextion Display, and the MCP4725 DAC to generate different types of signals while allowing user control and waveform saving.

The project utilizes these components:

- Arduino Mega: This microcontroller board is the central control unit.
- Nextion Display: It serves as the graphical user interface for signal selection, customization, and saving.
- MCP4725 DAC: This digital-to-analog converter generates analog waveforms based on digital input.
- Supporting Components: These include a power supply, wires and other necessary parts.

The generator's functionalities include:

- Signal Selection: The Nextion Display interface enables users to choose from preset signal types: Sinewave, Triangle, Square, and Sawtooth. The selected signal type is sent to the Arduino for processing.
- Custom Signal Creation: Users can design their own custom signal by specifying points on the waveform graph through the Nextion Display's touch interface. The Arduino interprets these points and generates the corresponding waveform.
- MCP4725 Control: The Arduino communicates with the MCP4725 DAC using the I2C protocol. It transmits calculated analog values corresponding to the selected or custom waveform, enabling the DAC to produce the desired analog output.
- EEPROM Storage: Custom waveforms can be saved to the EEPROM for later use. The Arduino manages read and write operations to the EEPROM, ensuring that the user's waveform designs are preserved even after power loss.

- Signal Generation: Depending on the user's choice (preset or custom waveform), the Arduino generates the waveform by sending appropriate digital values to the MCP4725 DAC.
- User Interaction: The Nextion Display provides interactive buttons, sliders, and touch controls for selecting waveforms, adjusting signal characteristics, and saving configurations.
- GUI Feedback: Real-time visualization of the generated waveform is displayed on the Nextion Display, allowing users to observe the waveform's shape and characteristics.

When implementing this project, it's important to consider factors like DAC resolution, waveform interpolation for smooth transitions, signal frequency range, managing EEPROM addresses for saving, and implementing error-handling mechanisms. In summary, the project integrates the Arduino Mega, Nextion Display, and MCP4725 DAC to create an Arbitrary Waveform Generator. It enables users to choose from predefined signal types, design their own custom waveforms, and save these designs for future use. The Nextion Display facilitates user interaction and visualization, while the Arduino processes user inputs, generates corresponding analog values, and communicates with the DAC to produce accurate and customizable analog waveforms. This project offers a versatile tool for experimenting with different analog signals in various electronics and engineering applications.

Περίληψη

Η δημιουργία μιας γεννήτριας συχνοτήτων περιλαμβάνει την ενσωμάτωση ενός Arduino Mega, μιας οθόνης Nextion και του DAC MCP4725 για την παραγωγή διαφορετικών τύπων σημάτων, επιτρέποντας παράλληλα τον έλεγχο του χρήστη και την αποθήκευση κυματομορφών.

Το πρότζεκτ χρησιμοποιεί τα παρακάτω εξαρτήματα:

- Arduino Mega: Ο μικροελεγκτής αποτελεί την κεντρική μονάδα ελέγχου.
- Nextion Display: Χρησιμεύει ως διεπαφή χρήστη για την επιλογή, προσαρμογή και αποθήκευση σημάτων.
- MCP4725 DAC: Αυτός ο μετατροπέας ψηφιακού σε αναλογικού σήματος παράγει αναλογικές κυματομορφές με βάση την ψηφιακή είσοδο.
- Υποστηρικτικά εξαρτήματα: Αυτά περιλαμβάνουν τροφοδοτικό, καλώδια και άλλα απαραίτητα εξαρτήματα.
- Οι λειτουργίες της γεννήτριας περιλαμβάνουν:
- Επιλογή σήματος: Η οθόνη Nextion επιτρέπει στους χρήστες να επιλέγουν από προκαθορισμένους τύπους σημάτων: Ημιτονοειδές, τρίγωνο, τετράγωνο και πριονωτό. Ο επιλεγμένος τύπος σήματος αποστέλλεται στο Arduino για επεξεργασία.
- Δημιουργία προσαρμοσμένου σήματος: Οι χρήστες μπορούν να σχεδιάσουν το δικό τους προσαρμοσμένο σήμα καθορίζοντας σημεία στο γράφημα κυματομορφής μέσω της διεπαφής αφής του Nextion Display. Το Arduino ερμηνεύει αυτά τα σημεία και παράγει την αντίστοιχη κυματομορφή.
- Έλεγχος MCP4725: Το Arduino επικοινωνεί με τον DAC MCP4725 χρησιμοποιώντας το πρωτόκολλο I2C. Μεταδίδει ψηφιακές τιμές που αντιστοιχούν στην επιλεγμένη ή προσαρμοσμένη κυματομορφή, επιτρέποντας στον DAC να παράγει την επιθυμητή αναλογική έξοδο.
- Αποθήκευση EEPROM: Οι προσαρμοσμένες κυματομορφές μπορούν να αποθηκευτούν στην EEPROM για μεταγενέστερη χρήση. Το Arduino διαχειρίζεται τις λειτουργίες

ανάγνωσης και εγγραφής στην EEPROM, διασφαλίζοντας ότι τα σχέδια κυματομορφών του χρήστη διατηρούνται ακόμη και μετά από απώλεια ρεύματος.

- Παραγωγή σήματος: Ανάλογα με την επιλογή του χρήστη (προκαθορισμένη ή προσαρμοσμένη κυματομορφή), το Arduino παράγει την κυματομορφή στέλνοντας τις κατάλληλες αναλογικές τιμές στο MCP4725 DAC.
- Αλληλεπίδραση με τον χρήστη: Το Nextion Display παρέχει διαδραστικά κουμπιά και χειριστήρια αφής για την επιλογή κυματομορφών, τη ρύθμιση των χαρακτηριστικών του σήματος και την αποθήκευση των ρυθμίσεων.
- Ανατροφοδότηση στο GUI: Η απεικόνιση της παραγόμενης κυματομορφής σε πραγματικό χρόνο εμφανίζεται στην οθόνη Nextion Display, επιτρέποντας στους χρήστες να παρατηρούν το σχήμα και τα χαρακτηριστικά της κυματομορφής.

Κατά την υλοποίηση αυτού του έργου, είναι σημαντικό να ληφθούν υπόψη παράγοντες όπως η ανάλυση του DAC, η παρεμβολή κυματομορφής για ομαλές μεταβάσεις, το εύρος συχνοτήτων του σήματος, η διαχείριση των διευθύνσεων EEPROM για την αποθήκευση και η εφαρμογή μηχανισμών χειρισμού σφαλμάτων.

Συνοπτικά, το έργο ενσωματώνει το Arduino Mega, την οθόνη Nextion και το MCP4725 DAC για τη δημιουργία μιας γεννήτριας αυθαίρετων κυματομορφών. Επιτρέπει στους χρήστες να επιλέξουν από προκαθορισμένους τύπους σημάτων, να σχεδιάσουν τις δικές τους προσαρμοσμένες κυματομορφές και να αποθηκεύσουν αυτά τα σχέδια για μελλοντική χρήση. Το Nextion Display διευκολύνει την αλληλεπίδραση και την οπτικοποίηση του χρήστη, ενώ το Arduino επεξεργάζεται τις εισόδους του χρήστη, παράγει τις αντίστοιχες αναλογικές τιμές και επικοινωνεί με το DAC για την παραγωγή ακριβών και προσαρμόσιμων αναλογικών κυματομορφών. Αυτό το έργο προσφέρει ένα ευέλικτο εργαλείο για τον πειραματισμό με διάφορα αναλογικά σήματα σε διάφορες εφαρμογές ηλεκτρονικής και μηχανικής.

1. Διατύπωση του προβλήματος

Η προτεινόμενη μεταπτυχιακή εργασία, πραγματεύεται την υλοποίηση μίας εφαρμογής Γεννήτριας Αυθαίρετων Συναρτήσεων (Arbitrary Waveform Generator) σε πλατφόρμα Arduino. Οι Γεννήτριες Αυθαίρετων Συναρτήσεων είναι συσκευές που μπορούν να παράγουν σήματα με διάφορες κυματομορφές, όπως ημιτονοειδείς, τριγωνικές, πριονωτές, τετραγωνικές, αλλά έχουν την δυνατότητα να παράξουν και κυματομορφές οποιουδήποτε άλλου σχήματος, που καθορίζεται από τον χρήστη.

Η Γεννήτρια Αυθαίρετων Συναρτήσεων θα πρέπει να έχει τις εξής δυνατότητες:

- Θα μπορεί να παράγει ημιτονοειδείς, τριγωνικές, πριονωτές και τετραγωνικές κυματομορφές καθοριζόμενης συχνότητας.
- Θα διαθέτει οθόνη touch 4.3 ιντσών μέσω της οποίας θα υλοποιεί το interface της εφαρμογής.
- Θα μπορεί να εμφανίζει στην οθόνη την μορφή της κυματομορφής που εξάγει, σε μορφή γραφικής απεικόνισης, καθώς και τα χαρακτηριστικά της, όπως συχνότητα, περίοδος, τάση peek-to-peek κλπ.
- Θα επιτρέπει στον χρήστη την σύνθεση νέας κυματομορφής την οποία θα μπορεί να αποθηκεύει στην εσωτερική EPROM του Arduino, ώστε να μπορεί να ανακτάται σε μελλοντικό χρόνο.
- Θα μπορεί να εξάγει την σχεδιασμένη κυματομορφή και ταυτόχρονα να την εμφανίζει στην οθόνη.

Θα πρέπει να δοθεί έμφαση ώστε η Γεννήτρια να έχει κατά το δυνατόν υψηλή ανάλυση (π.χ. 12 bit) και να μπορεί να παράγει κυματομορφές όσο το δυνατόν μεγαλύτερου εύρους συχνοτήτων. Θα χρειαστεί η προμήθεια ενός DAC (Digital to Analog converter) που θα οδηγείται από τον Arduino (κανονικές έξοδοι ή PWM).

Εξαρτήματα:

- Arduino Mega: Λειτουργεί ως κεντρική μονάδα ελέγχου, επεξεργάζεται τις εισόδους του χρήστη, παράγει σήματα και διαχειρίζεται τις αλληλεπιδράσεις.
- Οθόνη Nextion (NX8048T050): Παρέχει ενα γραφικό περιβάλλον για τους χρήστες ώστε να επιλέγουν σήματα, να προσαρμόζουν τις ρυθμίσεις και να απεικονίζουν κυματομορφές.
- MCP4725 DAC (ανάλυση 12 bit): Μετατρέπει τα ψηφιακά σήματα από το Arduino σε αναλογικές κυματομορφές με ακρίβεια.

Λειτουργίες:

- Παραγωγή σήματος: Υλοποίηση αλγορίθμων για τη δημιουργία προκαθορισμένων σημάτων (ημίτονο, τρίγωνο, παλμός, πριονωτός) με δυναμικές ρυθμίσεις συχνότητας και πλάτους.
- Ανάπτυξη μηχανισμού για τη δημιουργία προσαρμοσμένων κυματομορφών με βάση τα σημεία που ορίζει ο χρήστης.

Γραφικό περιβάλλον (Nextion Display):

- Σχεδιασμός μιας διεπαφής που θα επιτρέπει στους χρήστες να επιλέγουν τύπους σημάτων (προκαθορισμένους ή προσαρμοσμένους).
- Εφαρμογή ρυθμιστικών/κουμπιών για την προσαρμογή των ρυθμίσεων συχνότητας και πλάτους του σήματος.
- Δυνατότητα στους χρήστες να δημιουργούν προσαρμοσμένες κυματομορφές επιλέγοντας σημεία στην οθόνη.

Διαχείριση EEPROM:

- Δημιουργία δύο θέσεων αποθήκευσης στην EEPROM για τις προσαρμοσμένες κυματομορφές.
- Ανάπτυξη διαδικασιών για την αποθήκευση, φόρτωση και διαγραφή προσαρμοσμένων κυματομορφών από αυτές τις υποδοχές.

Έλεγχος σήματος:

- Δημιουργία επικοινωνίας μεταξύ του Arduino και του MCP4725 DAC χρησιμοποιώντας το πρωτόκολλο I2C.
- Καθιέρωση μεθόδου επικοινωνίας μεταξύ του Arduino και της οθόνης Nextion, επιτρέποντας την ανταλλαγή δεδομένων και την αλληλεπίδραση με τον χρήστη.

Προσαρμοσμένη αποθήκευση κυματομορφών:

- Αποτελεσματική διαχείριση της αποθήκευσης EEPROM για προσαρμοσμένες κυματομορφές.
- Ανάπτυξη αλγορίθμων για την αποθήκευση, διαγραφή και ανάγνωση δεδομένων κυματομορφής.

Σχεδιασμός γραφικού περιβάλλοντος:

- Σχεδίαση γραφικού περιβάλλοντος με ευέλικτη αλληλεπίδραση αφής, απεικόνιση κυματομορφών και διαισθητικά χειριστήρια.

Χειρισμός διευθύνσεων EEPROM:

- Εφαρμογή αποτελεσματικών μεθόδων για τη διαχείριση των διευθύνσεων EEPROM, διασφαλίζοντας την ακεραιότητα των δεδομένων κατά τη διάρκεια των λειτουργιών ανάγνωσης και εγγραφής.

Testing:

- Ακρίβεια σήματος: Επικύρωση της ακρίβειας των παραγόμενων σημάτων σε διαφορετικές συχνότητες και πλάτη.

Δημιουργία προσαρμοσμένων κυματομορφών:

- Δυνατότητα χρήστη να δημιουργεί προσαρμοσμένες κυματομορφές. Επαλήθευση της ακριβούς επιλογής και ανακατασκευής σημείων.

Λειτουργίες EEPROM:

- Δοκιμή αποθήκευσης, φόρτωσης, ανάκτησης και διαγραφής προσαρμοσμένων κυματομορφών στην EEPROM. Διασφάλιση της σωστής διαχείρισης των θέσεων μνήμης.

Αλληλεπίδραση με τον χρήστη:

- Αξιολόγηση του γραφικού περιβάλλοντος χρήστη ως προς τη χρηστικότητα, την ανταπόκριση και την ακρίβεια.

Ενσωμάτωση συστήματος:

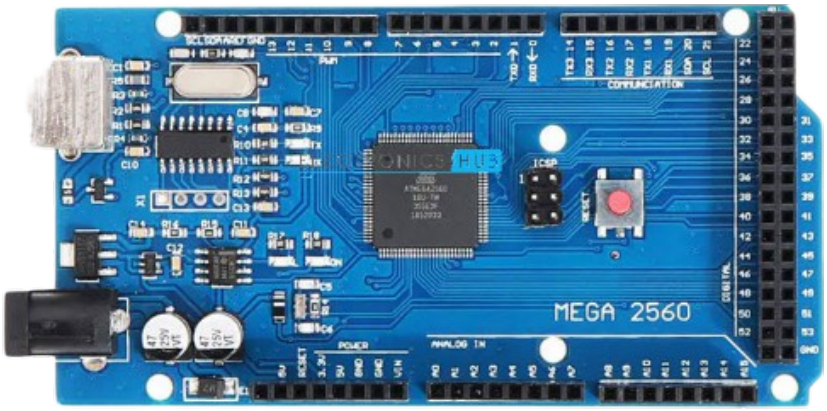
- Δοκιμασία της πλήρης λειτουργικότητας του συστήματος, συμπεριλαμβανομένης της δημιουργίας σημάτων, της προσαρμογής, της αποθήκευσης και των ενημερώσεων της οθόνης.

Project flow:

1. Αρχικοποίηση του συστήματος, διαμόρφωση του DAC, της οθόνης Nextion και της EEPROM.
2. Εμφάνιση του κύριου μενού στην οθόνη Nextion Display, προσφέροντας επιλογές σήματος και επιλογές προσαρμογής.
3. Με βάση την επιλογή του χρήστη, δημιουργία προκαθορισμένων σημάτων, ρύθμιση για τη δημιουργία προσαρμοσμένων κυματομορφών ή διαχείριση της μνήμης EEPROM.
4. Να επιτρέπει στους χρήστες να επιλέγουν σημεία για τη δημιουργία προσαρμοσμένων κυματομορφών στην οθόνη Nextion Display.
5. Μετατροπή των εισόδων του χρήστη σε ψηφιακές τιμές και αποστολή τους στο DAC για τη δημιουργία αναλογικών σημάτων.
6. Εφαρμογή της διαχείρισης EEPROM για την αποθήκευση προσαρμοσμένων κυματομορφών, συμπεριλαμβανομένης της αποθήκευσης, φόρτωσης και διαγραφής.
7. Συνεχής ενημέρωση της οθόνης Nextion Display για να αντικατοπτρίζει τις αλλαγές κυματομορφής και τις ενέργειες του χρήστη.
8. Δοκιμή της ακρίβειας, της απόκρισης και της συνολικής απόδοσης του συστήματος.
9. Αντιμετώπιση τυχόν εντοπισμένων προβλημάτων ή βελτιώσεων κατά τη διάρκεια των δοκιμών.

Η δημιουργία μιας γεννήτριας αυθαίρετων κυματομορφών περιλαμβάνει την ενσωμάτωση του Arduino Mega, της οθόνης Nextion (NX8048T050) και του DAC MCP4725 (ανάλυση 12 bit). Η επίτευξη ακριβούς παραγωγής σήματος, η μεγάλη συχνότητα σήματος, η φιλική προς το χρήστη αλληλεπίδραση, η διαχείριση EEPROM και οι ρυθμίσεις σε πραγματικό χρόνο θα είναι ζωτικής σημασίας. Η προσοχή στην ανάλυση του DAC, τη διαχείριση της μνήμης, το σχεδιασμό της οθόνης και την ακρίβεια του σήματος είναι απαραίτητη για να εξασφαλιστεί η επιτυχής υλοποίηση μιας λειτουργικής και επικεντρωμένης στο χρήστη γεννήτριας.

2. Arduino mega



Το Arduino Mega είναι ένας μικροελεγκτής που αποτελεί μέρος της οικογένειας Arduino των πλατφορμών υλικού και λογισμικού ανοικτού κώδικα. Είναι μια εκτεταμένη έκδοση του αρχικού Arduino Uno, σχεδιασμένη για πιο σύνθετα και απαιτητικές εφαρμογές που απαιτούν πρόσθετα pin εισόδου και εξόδου, περισσότερη μνήμη και μεγαλύτερη επεξεργαστική ισχύ. Ακολουθούν ορισμένα βασικά χαρακτηριστικά και πληροφορίες σχετικά με το Arduino Mega:

Μικροελεγκτής: Το Arduino Mega βασίζεται στο τσιπ μικροελεγκτή ATmega2560, το οποίο διαθέτει 256 KB μνήμης flash για την αποθήκευση του κώδικα του προγράμματός, 8 KB SRAM και 4 KB EEPROM για την αποθήκευση δεδομένων.

Ψηφιακά και αναλογικά pin: Διαθέτει συνολικά 54 ψηφιακά pin εισόδου/εξόδου, 15 από τα οποία μπορούν να χρησιμοποιηθούν ως έξοδοι PWM (διαμόρφωση εύρους παλμών). Διαθέτει επίσης 16 pin αναλογικής εισόδου, τα οποία χρησιμοποιούνται συχνά για την ανάγνωση δεδομένων αισθητήρων.

Interface επικοινωνίας: Το Mega υποστηρίζει πολλαπλά interface επικοινωνίας, συμπεριλαμβανομένων των UART (σειριακή), I2C και SPI. Αυτό το καθιστά κατάλληλο για ένα ευρύ φάσμα εφαρμογών που απαιτούν επικοινωνία με άλλες συσκευές.

Ταχύτητα ρολογιού: Το ATmega2560 στο Arduino Mega λειτουργεί στα 16 MHz, παρέχοντας μεγαλύτερη επεξεργαστική ισχύ σε σύγκριση με το Arduino Uno.

Τάση λειτουργίας: Λειτουργεί στα 5V, όπως και οι περισσότερες άλλες πλακέτες Arduino. Μπορεί να τροφοδοτηθεί μέσω USB, μιας υποδοχής συνεχούς ρεύματος ή ενός εξωτερικού τροφοδοτικού.

Συμβατότητα: Το Arduino Mega είναι συμβατό με το Arduino IDE, το οποίο διευκολύνει τον προγραμματισμό, τη φόρτωση και το debugging του κώδικά. Στον Mega μπορεί να χρησιμοποιηθεί μια ποικιλία από libraries και shields που έχουν σχεδιαστεί για το Arduino.

Μνήμη: Η μεγαλύτερη μνήμη και τα πρόσθετα pins του Mega το καθιστούν κατάλληλο για πιο σύνθετες εφαρμογές, συμπεριλαμβανομένων εκείνων που περιλαμβάνουν πολλαπλούς αισθητήρες, οθόνες ή άλλα περιφερειακά.

Εφαρμογές: Λόγω των αυξημένων δυνατοτήτων του, το Arduino Mega χρησιμοποιείται συχνά σε εφαρμογές που απαιτούν μεγάλο αριθμό pin εισόδου/εξόδου, όπως τρισδιάστατοι εκτυπωτές, ρομποτική, συστήματα αυτοματισμού και εφαρμογές καταγραφής δεδομένων.

Συμβατότητα με shield: Είναι συμβατό με πολλές shields Arduino, αν και θα πρέπει να ελεγχθεί η συμβατότητα με συγκεκριμένες shields, καθώς ορισμένες μπορεί να είναι σχεδιασμένες για τις μικρότερες πλακέτες Arduino.

Κόστος και διαθεσιμότητα: Οι πλακέτες Arduino Mega διατίθενται εύκολα από διάφορους προμηθευτές και είναι σχετικά προσιτές, γεγονός που τις καθιστά δημοφιλή επιλογή για εφαρμογές που απαιτούν επιπλέον πόρους.

Το Arduino Mega είναι μια εξαιρετική επιλογή για εφαρμογές που απαιτούν περισσότερες ακίδες I/O, μνήμη και επεξεργαστική ισχύ από αυτή που μπορεί να παρέχει ο μικροελεγκτής. Είναι μια ευέλικτη και ισχυρή πλακέτα κατάλληλη για ένα ευρύ φάσμα εφαρμογών.

3. Nextion NX8048T050 display

Η Nextion NX8048T050 είναι μια δημοφιλής οθόνης TFT (Thin-Film Transistor) που κατασκευάζεται από την Nextion. Συνδυάζει μια διεπαφή οθόνης αφής με έναν μικροελεγκτή, επιτρέποντας στους προγραμματιστές να δημιουργούν διαδραστικές γραφικές διεπαφές χρήστη (GUI) για τις εφαρμογές τους. Μερικές πληροφορίες σχετικά με την οθόνη Nextion NX8048T050:

1. Προδιαγραφές οθόνης:

- Μέγεθος οθόνης: Το NX8048T050 διαθέτει επιφάνεια οθόνης διαγωνίου 5,0 ιντσών.
- Ανάλυση: Διαθέτει ανάλυση 800 x 480 pixels, παρέχοντας καθαρή και ζωντανή γραφική απόδοση.
- Τεχνολογία οθόνης: Η οθόνη χρησιμοποιεί τεχνολογία TFT LCD, προσφέροντας καλή αναπαραγωγή χρωμάτων και ευρείες γωνίες θέασης.
- Διεπαφή αφής: Η οθόνη Nextion υποστηρίζει χωρητική αφή, επιτρέποντας στους χρήστες να αλληλεπιδρούν με το GUI χρησιμοποιώντας την αφή.

2. Ενσωματωμένος μικροελεγκτής:

- Ο NX8048T050 ενσωματώνει έναν ισχυρό ενσωματωμένο μικροελεγκτή, ο οποίος απαλλάσσει τον κύριο ελεγκτή από τον χειρισμό της λειτουργικότητας της οθόνης.
- Ο ενσωματωμένος μικροελεγκτής εκτελεί το ιδιόκτητο υλικολογισμικό της Nextion, χειριζόμενος την είσοδο αφής, τη γραφική απόδοση και άλλες λειτουργίες που σχετίζονται με την οθόνη.
- Το λογισμικό Nextion Editor επιτρέπει στους προγραμματιστές να σχεδιάζουν το GUI και να προγραμματίζουν τη συμπεριφορά της οθόνης χρησιμοποιώντας μια οπτική διεπαφή.

3. Μνήμη και αποθήκευση:

- Η μονάδα οθόνης διαθέτει μνήμη flash 4MB, η οποία χρησιμοποιείται για την αποθήκευση του υλικολογισμικού, των γραμματοσειρών, των εικόνων και των αρχείων έργου. Περιλαμβάνει επίσης 3584 bytes μνήμης RAM για την αποθήκευση μεταβλητών και την προσωρινή αποθήκευση των δεδομένων της οθόνης.

4. Διεπαφές επικοινωνίας:

- Διεπαφή UART: Ο NX8048T050 επικοινωνεί κυρίως με έναν εξωτερικό μικροελεγκτή ή υπολογιστή χρησιμοποιώντας μια σειριακή διασύνδεση UART (Universal Asynchronous Receiver-Transmitter). Αυτό επιτρέπει την απρόσκοπτη μεταφορά δεδομένων μεταξύ της οθόνης και του κύριου ελεγκτή.

5. Λογισμικό Nextion Editor:

Το λογισμικό Nextion Editor είναι ένα εργαλείο ανάπτυξης που παρέχεται από την Nextion για τη δημιουργία της γραφικής διεπαφής χρήστη για τη μονάδα οθόνης. Διαθέτει μια διεπαφή drag-and-drop, επιτρέποντας στους προγραμματιστές να σχεδιάζουν τα στοιχεία του GUI, να προσαρμόζουν την εμφάνιση και να ορίζουν συμβάντα αλληλεπίδρασης. Το Nextion Editor παρέχει επίσης ένα πλούσιο σύνολο στοιχείων, όπως κουμπιά, ρυθμιστικά, μπάρες προόδου, πλαίσια κειμένου, μετρητές και άλλα, τα οποία μπορούν εύκολα να προστεθούν στη

διεπαφή. Το λογισμικό περιλαμβάνει μια λειτουργία προσομοιωτή που επιτρέπει στους προγραμματιστές να κάνουν προεπισκόπηση και να δοκιμάζουν το σχεδιασμό της γραφικής διεπαφής τους χωρίς να συνδέονται με την πραγματική μονάδα προβολής.

6. Απαιτήσεις ισχύος:

Το NX8048T050 λειτουργεί με τάση τροφοδοσίας 4,7V έως 7,0V. Έχει χαμηλή κατανάλωση ενέργειας, καθιστώντας το κατάλληλο για εφαρμογές που λειτουργούν με μπαταρία.

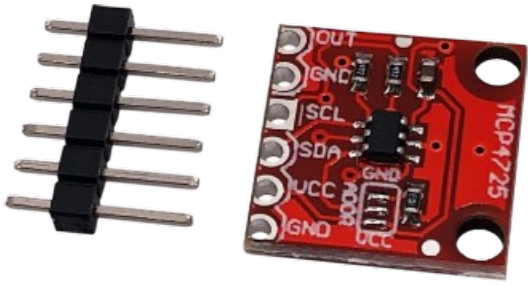
7. Προγραμματισμός και ενσωμάτωση:

Η μονάδα οθόνης Nextion απαιτεί την ανάπτυξη ενός ξεχωριστού προγράμματος ή υλικολογισμικού για τον εξωτερικό μικροελεγκτή ή τη συσκευή υποδοχής για τον χειρισμό της λογικής επικοινωνίας και ελέγχου. Επίσης παρέχει βιβλιοθήκες και παραδείγματα κώδικα σε διάφορες γλώσσες προγραμματισμού, συμπεριλαμβανομένων των Arduino, Raspberry Pi, ESP8266 και STM32, για να διευκολύνει την ενσωμάτωση με δημοφιλείς πλατφόρμες.

Η οθόνη επικοινωνεί με τον εξωτερικό μικροελεγκτή χρησιμοποιώντας ένα σύνολο απλών εντολών μέσω της διεπαφής UART, επιτρέποντας τον έλεγχο των στοιχείων της οθόνης, την ενημέρωση μεταβλητών και τη λήψη συμβάντων αφής.

Το Nextion NX8048T050 είναι μια ευέλικτη και φιλική προς το χρήστη μονάδα οθόνης που απλοποιεί τη δημιουργία γραφικών διεπαφών για εφαρμογές ενσωματωμένων συστημάτων. Ο ενσωματωμένος μικροελεγκτής, οι δυνατότητες αφής και το εύχρηστο λογισμικό ανάπτυξης την καθιστούν δημοφιλή επιλογή μεταξύ ερασιτεχνών, κατασκευαστών και επαγγελματιών για τη δημιουργία διαδραστικών και οπτικά ελκυστικών διεπαφών σε ένα ευρύ φάσμα εφαρμογών, όπως οικιακός αυτοματισμός, βιομηχανικά συστήματα ελέγχου, συσκευές IoT και πολλά άλλα.

4. Digital-to-analog converter MCP4725



Το MCP4725 είναι ένα δημοφιλές ολοκληρωμένο κύκλωμα μετατροπέα ψηφιακού σήματος σε αναλογικό (DAC) 12 bit που κατασκευάζεται από την Microchip Technology. Παρέχει έναν απλό και βολικό τρόπο για την προσθήκη δυνατοτήτων αναλογικής εξόδου σε εφαρμογές βασισμένες σε μικροελεγκτή.

1. Ανάλυση και ακρίβεια:

- Το MCP4725 είναι ένας DAC 12 bit, που σημαίνει ότι μπορεί να παράγει αναλογική έξοδο με ανάλυση 12 bit.
- Μπορεί να παράγει 4096 διακριτά επίπεδα εξόδου, παρέχοντας λεπτομερή έλεγχο της αναλογικής εξόδου τάσης.

2. Επικοινωνία:

- Το MCP4725 επικοινωνεί με τον μικροελεγκτή ή τη συσκευή υποδοχής μέσω της σειριακής διεπαφής I2C (Inter-Integrated Circuit).
- Υποστηρίζει την επικοινωνία I2C τυπικής και γρήγορης λειτουργίας, επιτρέποντας την εύκολη διασύνδεσή του με διάφορους μικροελεγκτές, όπως το Arduino, το Raspberry Pi και άλλους.
- Η συσκευή διαθέτει διαμορφώσιμη διεύθυνση I2C 7-bit, επιτρέποντας τη σύνδεση πολλαπλών MCP4725 στον ίδιο δίαυλο.

3. Εύρος και αναφορά εξόδου τάσης:

- Το MCP4725 μπορεί να παράγει αναλογικές εξόδους τάσης στο εύρος 0 έως Vref, όπου η Vref μπορεί να διαμορφωθεί είτε ως εσωτερική τάση αναφοράς είτε ως εξωτερική τάση αναφοράς.
- Εσωτερική αναφορά: Το MCP4725 περιλαμβάνει μια εσωτερική τάση αναφοράς (Vref) 2,048V, η οποία είναι η προεπιλεγμένη ρύθμιση.
- Εξωτερική αναφορά: Εναλλακτικά, μπορεί να χρησιμοποιηθεί μια εξωτερική τάση αναφοράς συνδέοντάς την στην ακίδα Vref του MCP4725. Αυτό επιτρέπει τον ορισμό ενός προσαρμοσμένου εύρους τάσης σύμφωνα με τις απαιτήσεις της εφαρμογής.

4. Τροφοδοσία ρεύματος και λειτουργία χαμηλής ισχύος:

- Το MCP4725 λειτουργεί σε τάση τροφοδοσίας που κυμαίνεται από 2,7V έως 5,5V, καθιστώντας το συμβατό με ένα ευρύ φάσμα μικροελεγκτών και πηγών τροφοδοσίας.
- Διαθέτει λειτουργία χαμηλής κατανάλωσης ενέργειας, επιτρέποντάς του να λειτουργεί σε κατάσταση αναστολής λειτουργίας και να εξοικονομεί ενέργεια όταν δεν παράγει ενεργά αναλογικές εξόδους.

5. EEPROM για την αποθήκευση των εξόδων DAC:

- Το MCP4725 ενσωματώνει μια ενσωματωμένη ηλεκτρικά διαγράψιμη προγραμματιζόμενη μνήμη μόνο για ανάγνωση (EEPROM).
- Η EEPROM επιτρέπει την αποθήκευση της τιμής εξόδου DAC, η οποία μπορεί να διατηρηθεί ακόμη και μετά από έναν κύκλο λειτουργίας ή όταν η συσκευή αποσυνδεθεί. Αυτή η λειτουργία είναι χρήσιμη όταν πρέπει να οριστεί μια συγκεκριμένη τάση εξόδου κατά την ενεργοποίηση ή για την επαναφορά μια προκαθορισμένης κατάσταση.

6. Έλεγχος και διαμόρφωση:

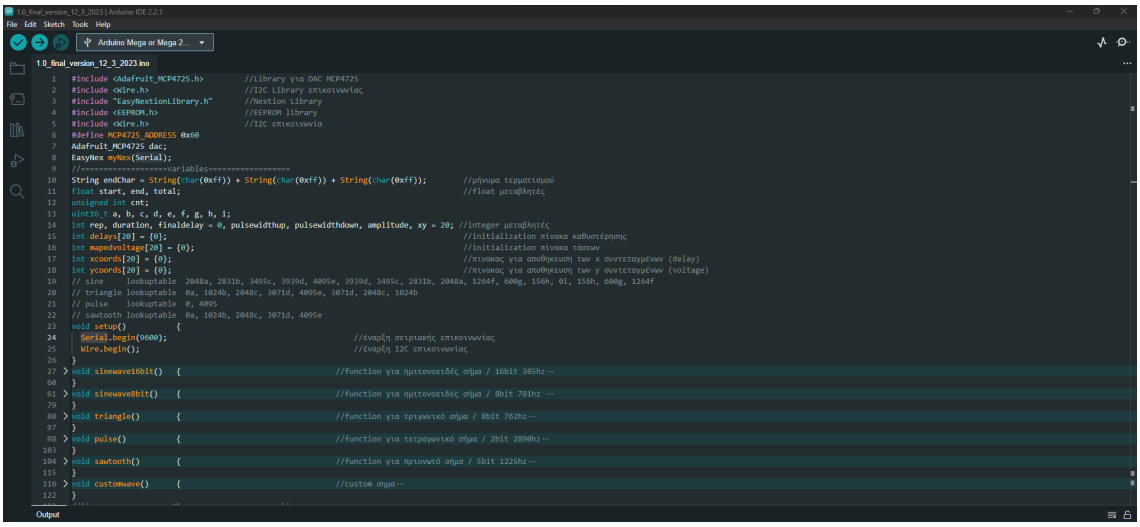
- Το MCP4725 παρέχει ευελιξία στη διαμόρφωση της τιμής εξόδου DAC μέσω εντολών I2C. Μπορεί να ρυθμιστεί η τάση εξόδου απευθείας χρησιμοποιώντας τον καταχωρητή DAC ή να χρησιμοποιηθεί η EEPROM για την αποθήκευση και την ανάκληση προκαθορισμένων τιμών τάσης. Η συσκευή υποστηρίζει τόσο λειτουργίες εγγραφής για την ενημέρωση της εξόδου DAC όσο και λειτουργίες ανάγνωσης για την ανάκτηση της τρέχουσας τιμής DAC ή της αποθηκευμένης τιμής EEPROM.

7. Βιβλιοθήκες και κώδικας:

- Για τη διευκόλυνση της εύκολης ενσωμάτωσης με διάφορους μικροελεγκτές, διατίθενται βιβλιοθήκες και κώδικας για το MCP4725. Αυτές οι βιβλιοθήκες παρέχουν λειτουργίες για την αρχικοποίηση της συσκευής, τη ρύθμιση της τάσης εξόδου, την ανάγνωση της τρέχουσας τιμής DAC και την εκτέλεση άλλων λειτουργιών. Οι βιβλιοθήκες χειρίζονται τα πρωτόκολλα επικοινωνίας I2C, διευκολύνοντας τη χρήση του MCP4725 σε εφαρμογές.

Το MCP4725 είναι ένα ευέλικτο και ευρέως χρησιμοποιούμενο DAC που προσφέρει ακριβείς αναλογικές εξόδους τάσης. Η απλότητα της λειτουργίας του, η χαμηλή κατανάλωση ενέργειας και η συμβατότητα με δημοφιλείς πλατφόρμες μικροελεγκτών τον καθιστούν εξαιρετική επιλογή για εφαρμογές που απαιτούν έλεγχο αναλογικών εξόδων, όπως η παραγωγή κυματομορφών, ο έλεγχος κινητήρων, η σύνθεση ήχου και η διασύνδεση αισθητήρων.

5. Arduino IDE (Integrated Development Environment)



Το Arduino IDE είναι μια εφαρμογή λογισμικού που έχει σχεδιαστεί ειδικά για τον προγραμματισμό πλακετών μικροελεγκτών Arduino. Παρέχει μια φιλική προς το χρήστη διεπαφή για τη συγγραφή, τη μεταγλώττιση και τη μεταφόρτωση κώδικα σε πλακέτες Arduino.

Το Arduino IDE περιλαμβάνει έναν επεξεργαστή κώδικα όπου μπορεί να γίνει ανάπτυξη και επεξεργασία του προγράμματος του Arduino, το οποία συνήθως αναφέρεται ως "σκίτσο". Παρέχει λειτουργίες όπως η επισήμανση συντακτικού και οι προτάσεις κώδικα που βοηθούν στην διαδικασία προγραμματισμού. Ο επεξεργαστής υποστηρίζει τη γλώσσα προγραμματισμού Arduino, η οποία βασίζεται στη C/C++.

Μια πρόσθετη λειτουργία που διαθέτει το Arduino IDE είναι ένας διαχειριστής βιβλιοθηκών που μας επιτρέπει να περιηγηθούμε, να εγκαταστήσουμε και να διαχειριστούμε εύκολα βιβλιοθήκες. Οι βιβλιοθήκες είναι έτοιμος κώδικα που παρέχουν πρόσθετη λειτουργικότητα, όπως ο έλεγχος συγκεκριμένων αισθητήρων, οθονών ή πρωτοκόλλων επικοινωνίας. Ο διαχειριστής παρέχει έναν βολικό τρόπο για την αναζήτηση βιβλιοθηκών, την εγκατάστασή τους και την ενημέρωσή τους.

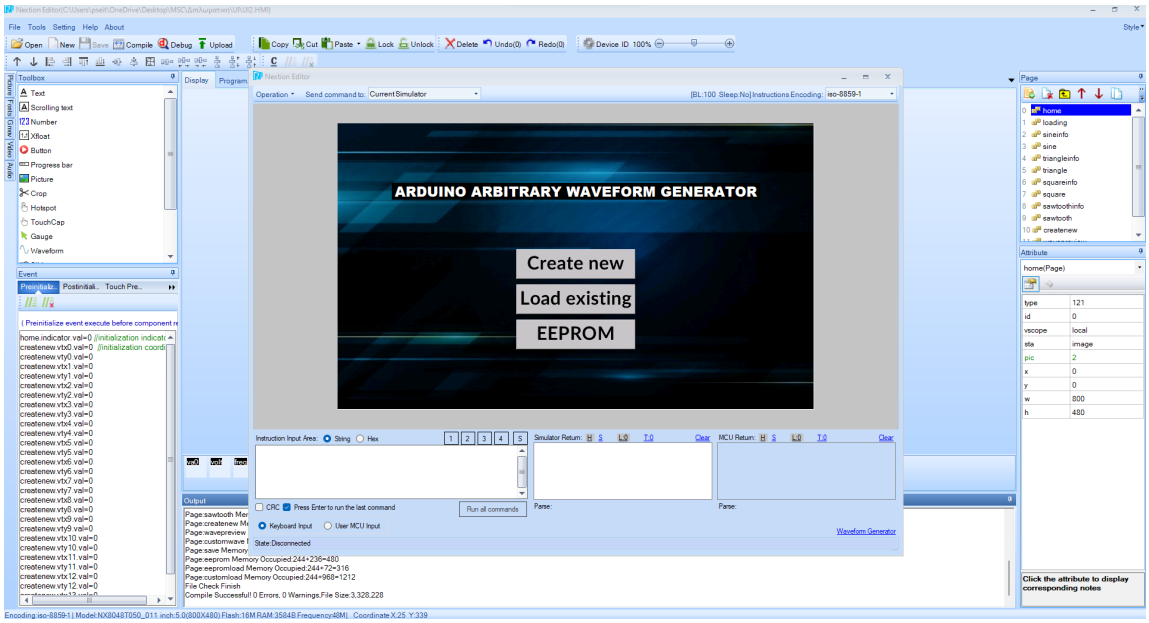
Ο Διαχειριστής πλακετών στο Arduino IDE κάνει εύκολη την εγκατάσταση και την διαχείριση διαφορετικών μοντέλων Arduino. Διαθέτει λίστα με τις υποστηριζόμενες πλακέτες Arduino, συμπεριλαμβανομένων των επίσημων πλακετών Arduino και πλακετών τρίτων κατασκευαστών. Μπορεί να γίνει επιλογή της κατάλληλης πλακέτας από το Board Manager για να διασφαλιστεί ότι το IDE μεταγλωττίζει σωστά τον κώδικα για τη συγκεκριμένη πλακέτα που χρησιμοποιείται.

Το Arduino IDE περιλαμβάνει ένα Serial Monitor, με το οποίο γίνεται η επικοινωνία με την πλακέτα Arduino μέσω της σειριακής θύρας. Παρέχει μια απλή διεπαφή για την αποστολή και λήψη δεδομένων μεταξύ του Arduino και του υπολογιστή, καθιστώντας το χρήσιμο για την αποσφαλμάτωση και την παρακολούθηση της συμπεριφοράς των sketches. Το λογισμικό μεταγλωττίζει το σκίτσο Arduino σε ένα δυαδικό αρχείο που μπορεί να φορτωθεί στην πλακέτα Arduino. Χειρίζεται αυτόματα τη διαδικασία μετάφρασης και παρέχει ενημέρωση σχετικά με τυχόν σφάλματα ή προειδοποιήσεις στην μετάφραση. Μόλις ο κώδικας μεταφραστεί επιτυχώς, μπορεί να φορτωθεί στην πλακέτα Arduino μέσω σύνδεσης USB ή άλλων υποστηριζόμενων διεπαφών.

Το Arduino IDE περιλαμβάνει ένα ευρύ φάσμα κώδικα παραδειγμάτων και εκπαιδευτικών οδηγιών για να βοηθήσει τους αρχάριους να ξεκινήσουν και να μάθουν τα βασικά στοιχεία του προγραμματισμού Arduino. Αυτά τα παραδείγματα καλύπτουν διάφορα θέματα, από τις βασικές ψηφιακές εισόδους/εξόδους μέχρι πιο προηγμένες έννοιες όπως η εργασία με αισθητήρες, οθόνες και πρωτόκολλα επικοινωνίας. Ο κώδικας των παραδειγμάτων χρησιμεύει ως πολύτιμη πηγή για την κατανόηση των διαφόρων χαρακτηριστικών και δυνατοτήτων της πλατφόρμας Arduino. Επίσης είναι διαθέσιμο για λειτουργικά συστήματα Windows, macOS και Linux, καθιστώντας το προσβάσιμο σε ένα ευρύ φάσμα χρηστών. Παρέχει μια συνεπή εμπειρία χρήσης σε διαφορετικές πλατφόρμες, επιτρέποντας στους χρήστες να αναπτύσσουν έργα Arduino χρησιμοποιώντας το λειτουργικό σύστημα που προτιμούν. Το Arduino IDE είναι λογισμικό ανοικτού κώδικα, πράγμα που σημαίνει ότι ο πηγαίος κώδικάς του είναι ελεύθερα διαθέσιμος για τροποποίηση και βελτίωση από την κοινότητα του Arduino. Η επεκτασιμότητα του IDE επιτρέπει στους προγραμματιστές να δημιουργήσουν και να ενσωματώσουν τα δικά

τους εργαλεία, βιβλιοθήκες και χαρακτηριστικά για να βελτιώσουν την εμπειρία ανάπτυξης του Arduino. Χρησιμεύει ως το κύριο εργαλείο για τον προγραμματισμό των πλακετών Arduino, παρέχοντας ένα φιλικό περιβάλλον προς το χρήστη για τη συγγραφή και τη φόρτωση κώδικα. Η απλότητά του, τα ενσωματωμένα παραδείγματα και το εκτεταμένο οικοσύστημα βιβλιοθηκών το καθιστούν μια εξαιρετική επιλογή για αρχάριους και έμπειρους χρήστες που θέλουν να αναπτύξουν έργα χρησιμοποιώντας μικροελεγκτές Arduino.

6. Nextion Editor

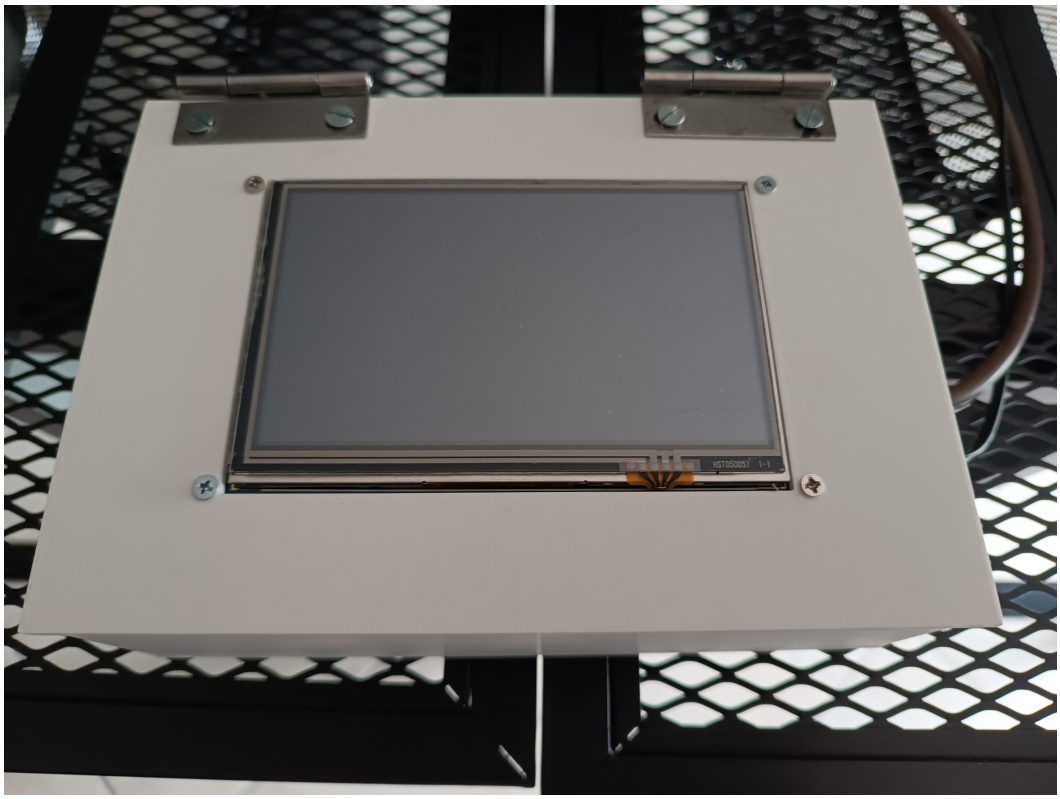


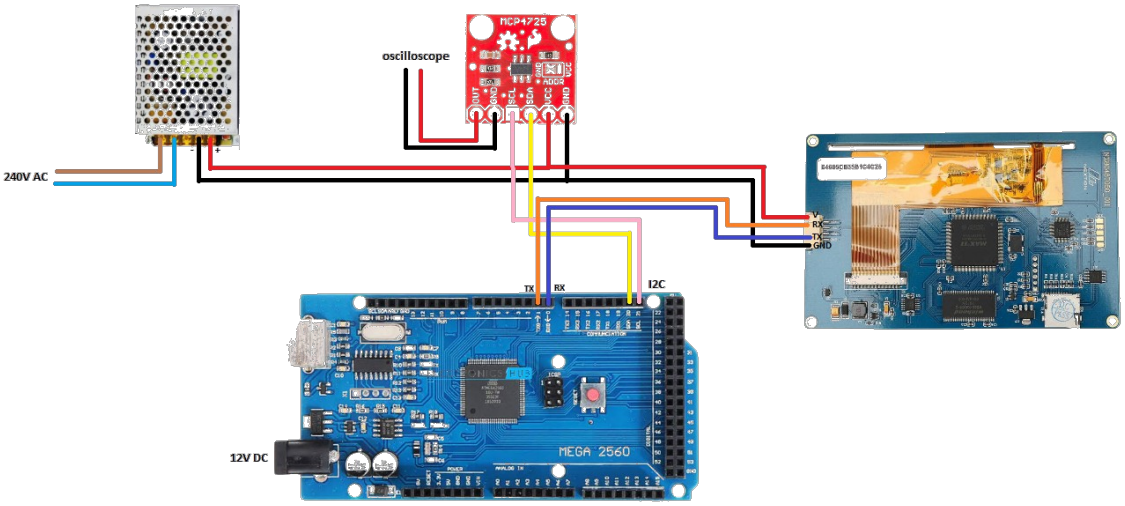
Το Nextion Editor είναι μια εφαρμογή λογισμικού που αναπτύχθηκε από την Nextion για το σχεδιασμό και τη δημιουργία γραφικών διεπαφών χρήστη (GUI) για τις οθόνες Nextion Intelligent TFT. Παρέχει ένα οπτικό περιβάλλον εργασίας για το σχεδιασμό και την προσαρμογή των GUIs για τις οθόνες Nextion. Διαθέτει μια διεπαφή drag-and-drop, επιτρέποντας στους χρήστες να προσθέτουν και να διατάσσουν εύκολα διάφορα στοιχεία GUI, όπως κουμπιά, ρυθμιστικά, μπάρες προόδου, πλαίσια κειμένου, μετρητές, εικόνες και άλλα. Οι χρήστες μπορούν να αλλάζουν το μέγεθος, τη θέση και να προσαρμόζουν την εμφάνιση αυτών των στοιχείων χρησιμοποιώντας ένα σύνολο στοιχείων ελέγχου. Ο Nextion Editor επιτρέπει τον χρήστη να μπορεί να δει μια ακριβή αναπαράσταση του σχεδιασμού GUI καθώς τον δημιουργεί. Οι αλλαγές που πραγματοποιούνται στον επεξεργαστή αντικατοπτρίζουν άμεσα τον τρόπο με τον οποίο το GUI θα εμφανίζεται στην οθόνη του Nextion. Υπάρχει η δυνατότητα να δημιουργηθούν κινούμενα γραφικά τα οποία ορίζουν το χειρισμό συμβάντων για διάφορα στοιχεία του GUI και μπορούν να χρησιμοποιηθούν για τη δημιουργία οπτικών εφέ, μεταβάσεων ή δυναμικής συμπεριφοράς εντός του GUI. Ο χειρισμός συμβάντων (Events) επιτρέπει στους χρήστες να ορίζουν ενέργειες και αποκρίσεις σε συγκεκριμένες αλληλεπιδράσεις του χρήστη,

όπως το πάτημα κουμπιών, κινήσεις ρυθμιστικών ή συμβάντα αφής. Το λογισμικό παρέχει μια επιλογή προκαθορισμένων γραμματοσειρών με διαφορετικά μεγέθη και στυλ που μπορούν να εφαρμοστούν σε στοιχεία κειμένου εντός του GUI. Οι χρήστες μπορούν επίσης να εισάγουν προσαρμοσμένες γραμματοσειρές για να επιτύχουν ένα μοναδικό οπτικό στυλ ή να ανταποκριθούν σε συγκεκριμένες απαιτήσεις του project. Ο Nextion Editor περιλαμβάνει μια λειτουργία προσομοιωτή που επιτρέπει στους χρήστες να κάνουν προεπισκόπηση και να αλληλεπιδρούν με το σχεδιασμό του GUI χωρίς να απαιτείται πραγματική οθόνη Nextion. Αυτή η λειτουργία είναι χρήσιμη για τη δοκιμή και τη λεπτομερή ρύθμιση της συμπεριφοράς και της οπτικής εμφάνισης του GUI πριν από την ανάπτυξή του στην οθόνη. Γίνεται διαχείριση πόρων, συμπεριλαμβανομένων εικόνων, γραμματοσειρών και ήχων, που χρησιμοποιούνται στο σχεδιασμό του GUI. Οι χρήστες μπορούν να εισάγουν και να οργανώνουν αυτούς τους πόρους μέσα στο λογισμικό για εύκολη πρόσβαση και χρήση μέσα στο project. Μόλις οριστικοποιηθεί ο σχεδιασμός του GUI στον Nextion Editor, το project μπορεί να μεταφραστεί σε ένα δυαδικό αρχείο. Αυτό το αρχείο μπορεί στη συνέχεια να φορτωθεί στη μονάδα οθόνης Nextion χρησιμοποιώντας μια σύνδεση USB ή άλλες υποστηριζόμενες διεπαφές. Η διαδικασία φόρτωσης μεταφέρει το σχέδιο GUI και τους σχετικούς πόρους στην οθόνη Nextion, επιτρέποντας την ανεξάρτητη λειτουργία της. Ο Nextion Editor παράγει ένα αρχείο υλικολογισμικού που περιλαμβάνει το σχέδιο GUI και τις σχετικές ρυθμίσεις διαμόρφωσης. Στη συνέχεια, το αρχείο υλικολογισμικού φορτώνεται στην οθόνη Nextion, όπου διασυνδέεται με το υλικολογισμικό της μονάδας οθόνης για την απόδοση και τον έλεγχο του GUI. Έτσι ο editor απλοποιεί τη διαδικασία σχεδιασμού και δημιουργίας προσαρμοσμένων GUI για τις οθόνες Nextion Intelligent TFT. Με τη διαισθητική διεπαφή, τη λειτουργικότητα drag-and-drop, την υποστήριξη κινούμενων γραφικών και τις δυνατότητες χειρισμού συμβάντων, ο επεξεργαστής παρέχει ένα ολοκληρωμένο σύνολο εργαλείων στους χρήστες για τη δημιουργία οπτικά ελκυστικών και διαδραστικών διεπαφών για τα έργα τους.

7.Κατασκευή και συνδεσμολογία

Για την υλοποίηση αυτού του έργου, χρησιμοποιήθηκε ένα Arduino mega ως κεντρικός ελεγκτής, που τροφοδοτείται μέσω της εισόδου jack. Η οθόνη Nextion NX8048T050 και ο DAC MCP4725 αποτέλεσαν βασικά στοιχεία του συστήματος. Για τη διασφάλιση της σωστής λειτουργίας αυτών των στοιχείων, χρησιμοποιήθηκε ένα τροφοδοτικό για την παροχή της απαραίτητης τάσης τόσο στην οθόνη όσο και στον DAC. Η σύνδεση μεταξύ του Arduino και του DAC δημιουργήθηκε μέσω του πρωτοκόλλου I2C, συνδέοντας τους αντίστοιχους ακροδέκτες. Εν τω μεταξύ, η επικοινωνία μεταξύ του Arduino και της οθόνης Nextion πραγματοποιήθηκε μέσω της σειριακής σύνδεσης. Τέλος, το σύστημα μπήκε μέσα σε ένα λευκό κουτί που κατασκευάστηκε από έναν εκτυπωτή 3D, παρέχοντας μια καθαρή εμφάνιση.



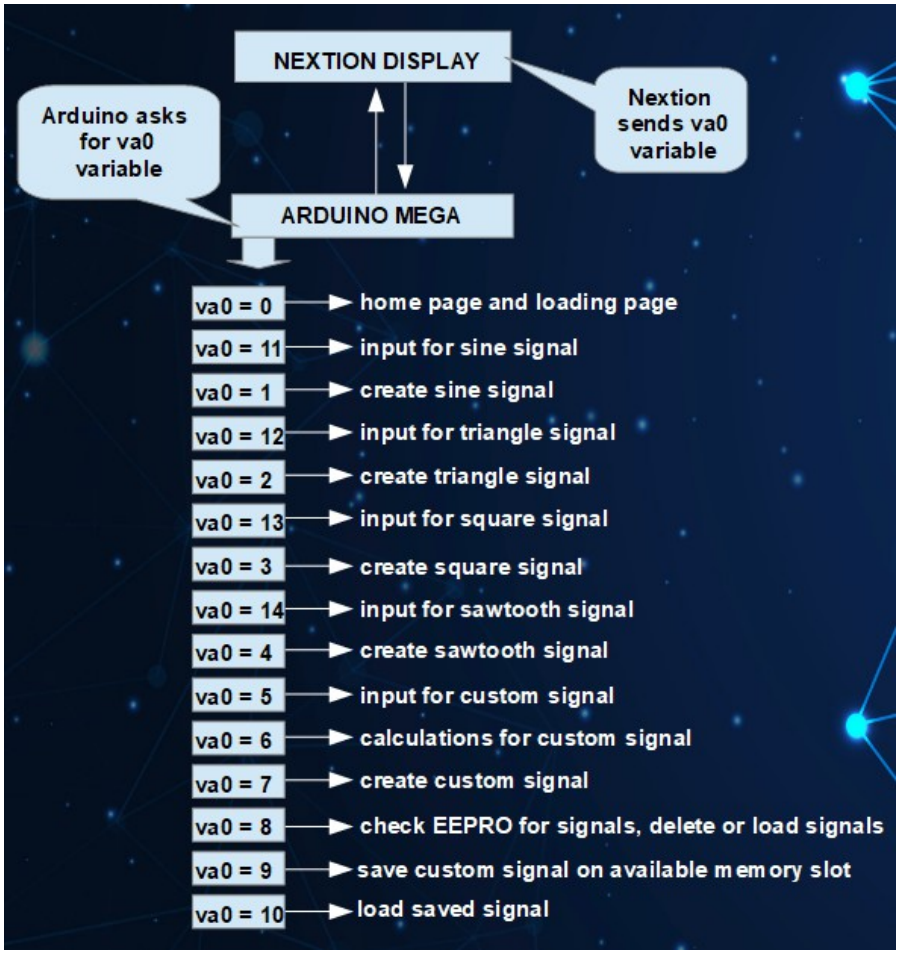

240V AC
oscilloscope
MCP4725
TX
RX
I2C
12V DC
MEGA 2560
RX
TX
GND

8. Λογισμικό

Οι απαιτήσεις του λογισμικού που αναπτύχθηκε είναι οι εξής:

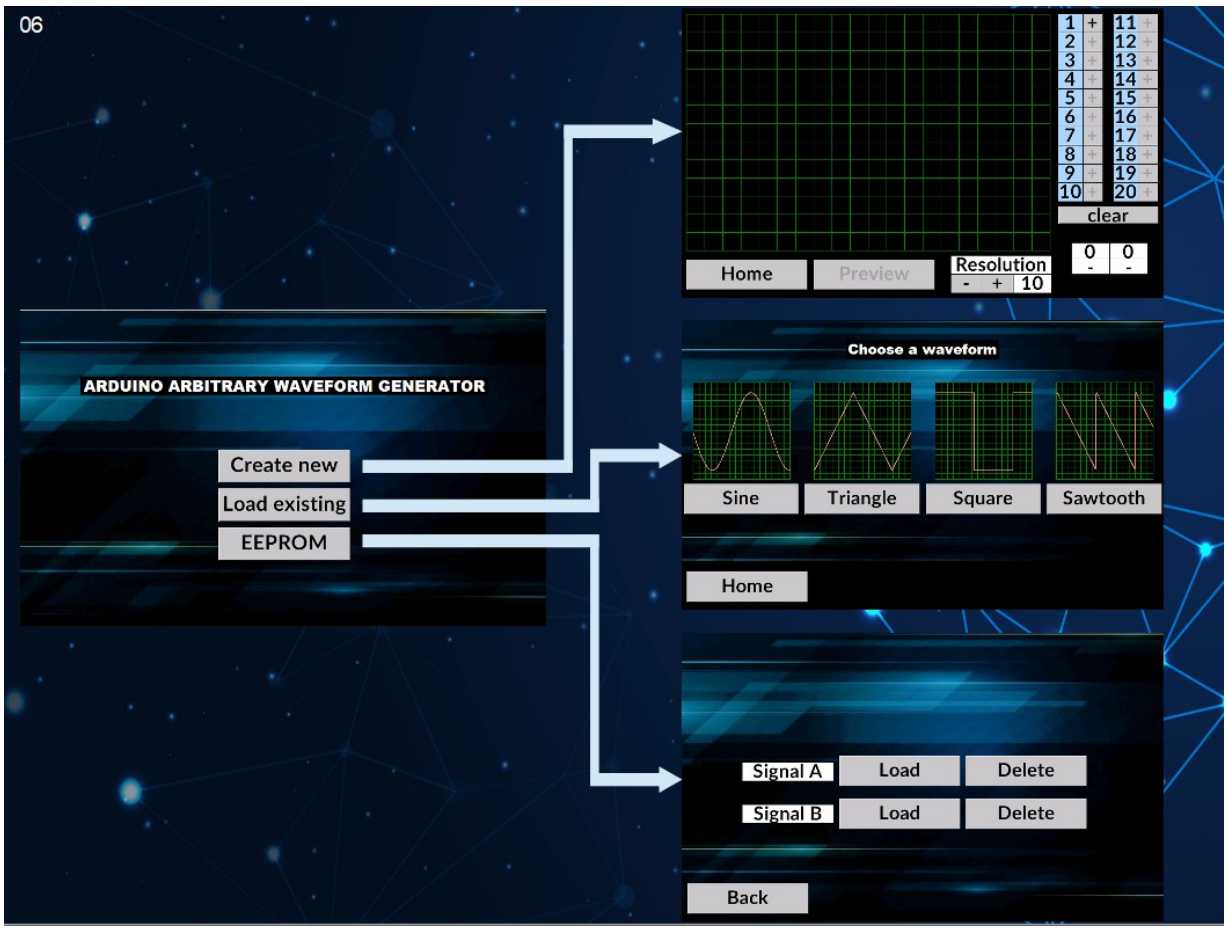
- Θα διαθέτει οθόνη touch στην οποία θα υλοποιηθεί γραφικό περιβάλλον.
- Θα μπορεί να παράγει ημιτονοειδείς, τριγωνικές, πριονωτές και τετραγωνικές κυματομορφές καθοριζόμενης συχνότητας και πλάτους.
- Θα εμφανίζει στην οθόνη την μορφή της κυματομορφής όπως και τα χαρακτηριστικά της (συχνότητα και πλάτος).
- Θα διαθέτει επιλογή σύνθεσης κυματομορφής από τον χρήστη και την οποία θα αποθηκεύει στην εσωτερική EEPROM του Arduino.
- Θα εξάγει την σχεδιασμένη κυματομορφή και ταυτόχρονα θα την εμφανίζει στην οθόνη.
- Θα διαθέτει την δυνατότητα φόρτωσης (load) της αποθηκευμένης κυματομορφής που σχεδίασε ο χρήστης αλλά και επιλογή διαγραφής.

Το λογισμικό που αναπτύχθηκε στηρίζεται στην επικοινωνία μεταξύ των 2 μικροελεγκτών, του Arduino και της οθόνης Nextion. Η βασική αρχή εξηγείται παρακάτω:



Στην οθόνη αποθηκεύεται μια μεταβλητή με όνομα va0 ή οποία είναι η βασική μεταβλητή που θα καθοδηγήσει τον Arduino ποια ρουτίνα να τρέξει στο πρόγραμμα. Ζητάει συνέχεια την τιμή της μεταβλητής από την οθόνη και αυτή του απαντάει. Ανάλογα την τιμή της ο Arduino θα τρέξει το μέρος του κώδικα που πρέπει. Όπως φαίνεται στη εικόνα υπάρχουν 15 λειτουργίες που έχουν ενσωματωθεί στον κώδικα για την υλοποίηση των απαιτήσεων του λογισμικού.

Home page και οι 3 βασικές επιλογές:

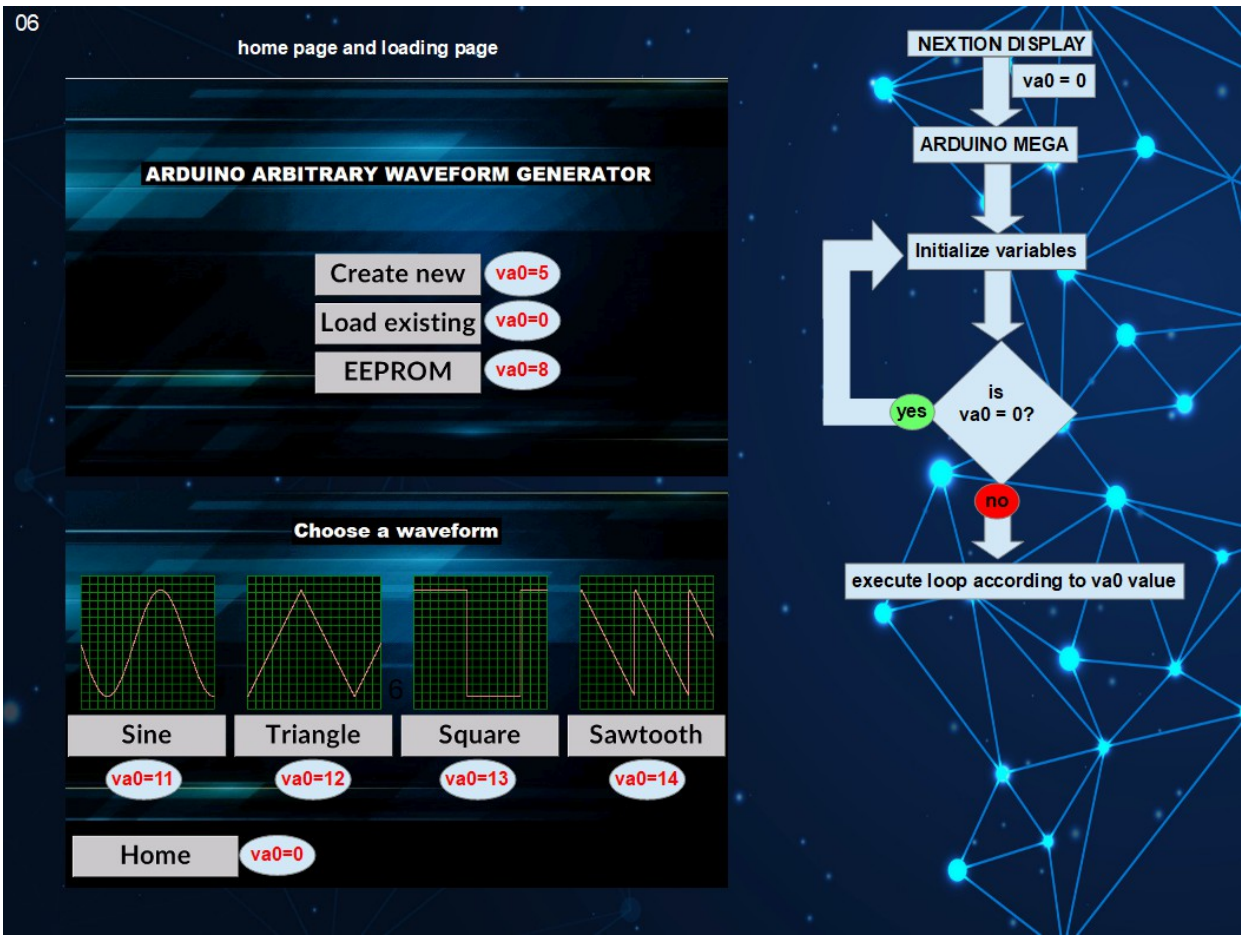


Η home page έχει 3 βασικές επιλογές.

1. Create new
2. Load existing
3. EEPROM

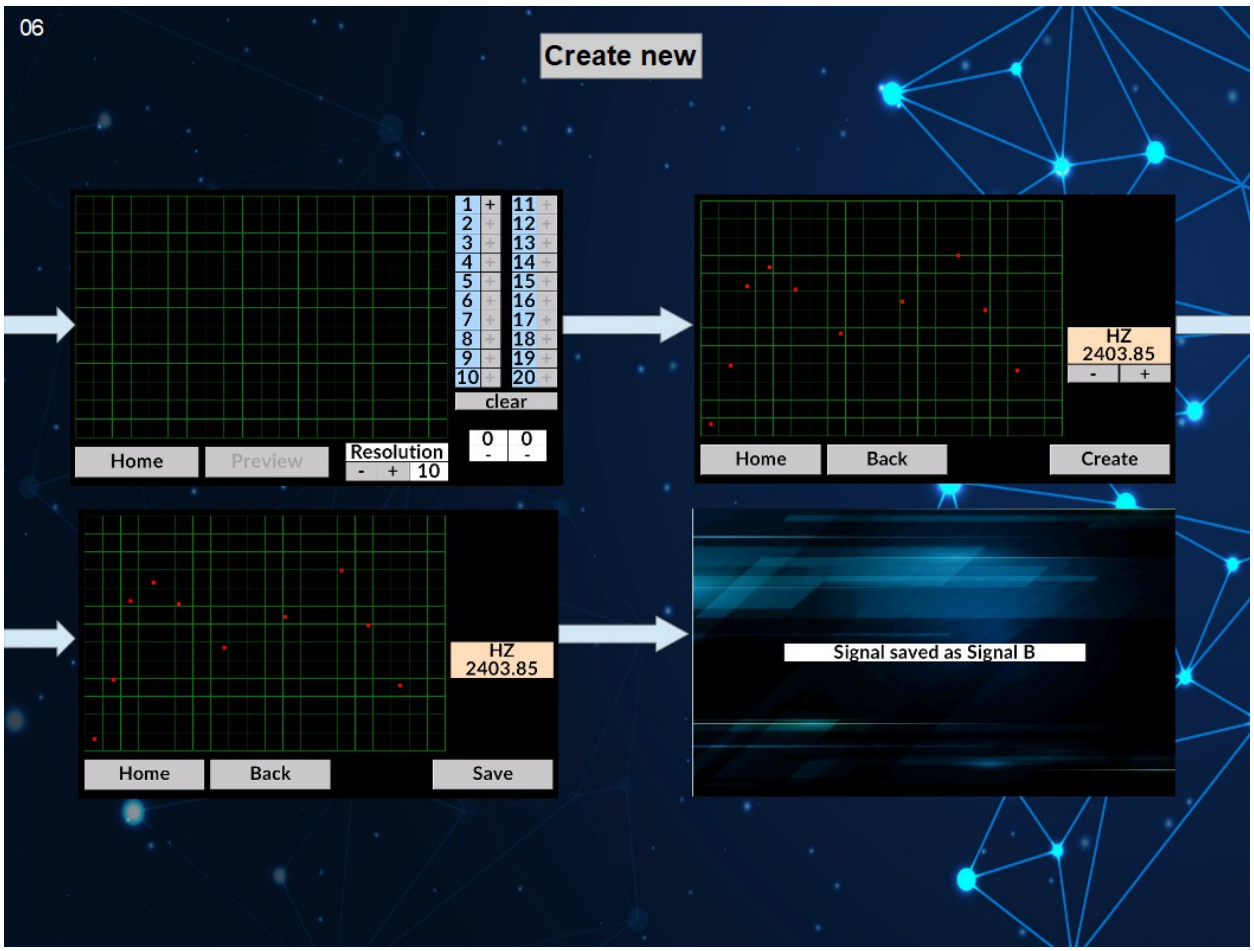
Η επιλογή Create new επιτρέπει την δημιουργία custom κυματομορφής, η Load existing επιτρέπει την φόρτωση μίας βασικής κυματομορφής όπως ημιτονοειδής, τριγωνική, παλμική ή πριονωτή.

Διάγραμμα ροής λογισμικού κατά την διάρκεια σελίδας Home ή Loading:



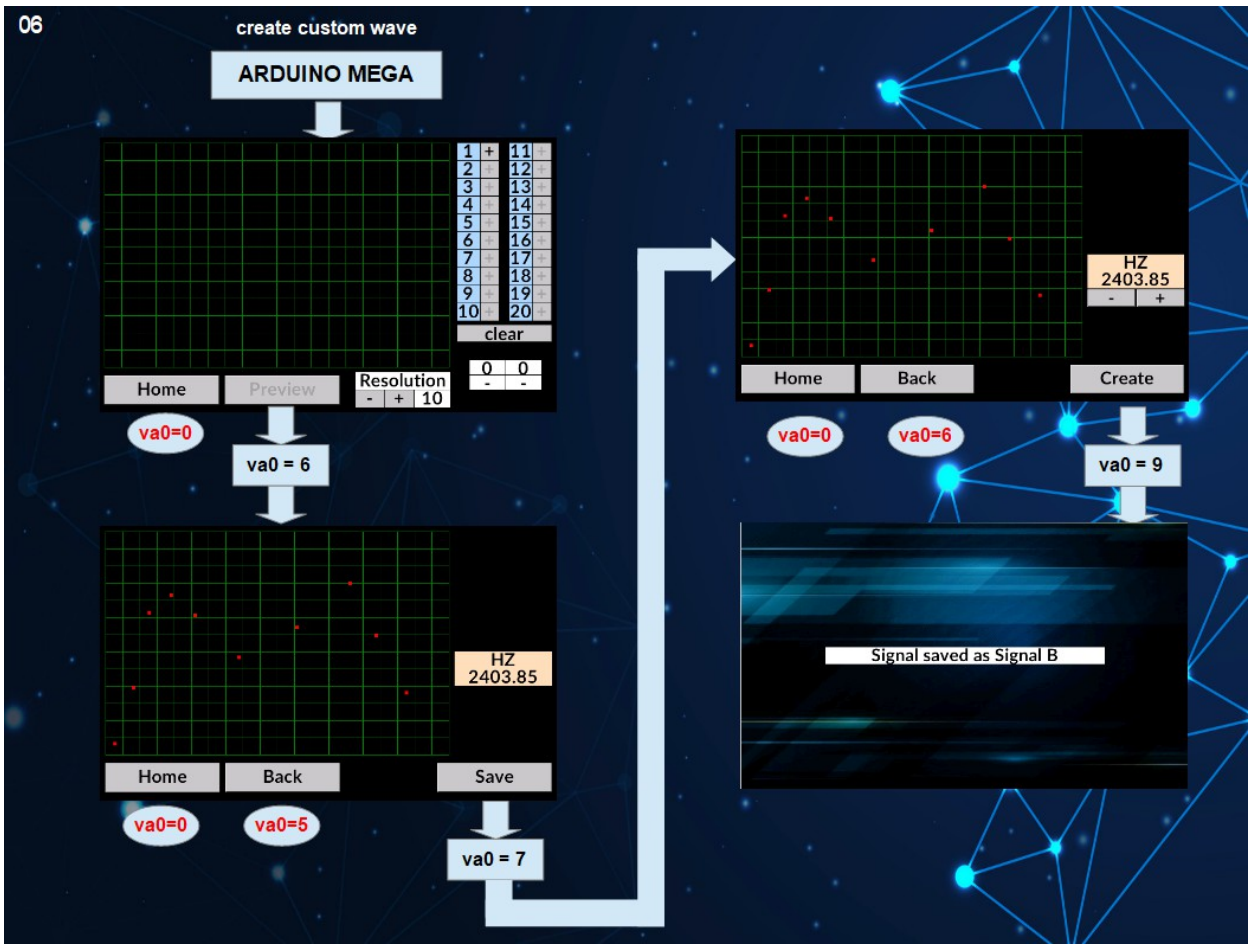
Ο Arduino παίρνει την τιμή της μεταβλητής va0 από την οθόνη, αρχικοποιεί τις μεταβλητές και ξανά ελέγχει την τιμή. Σε περίπτωση που έχει αλλάξει η μεταβλητή από επιλογή του χρήστη στο γραφικό περιβάλλον τότε ο Arduino θα εκτελέσει τον αντίστοιχο κώδικα.

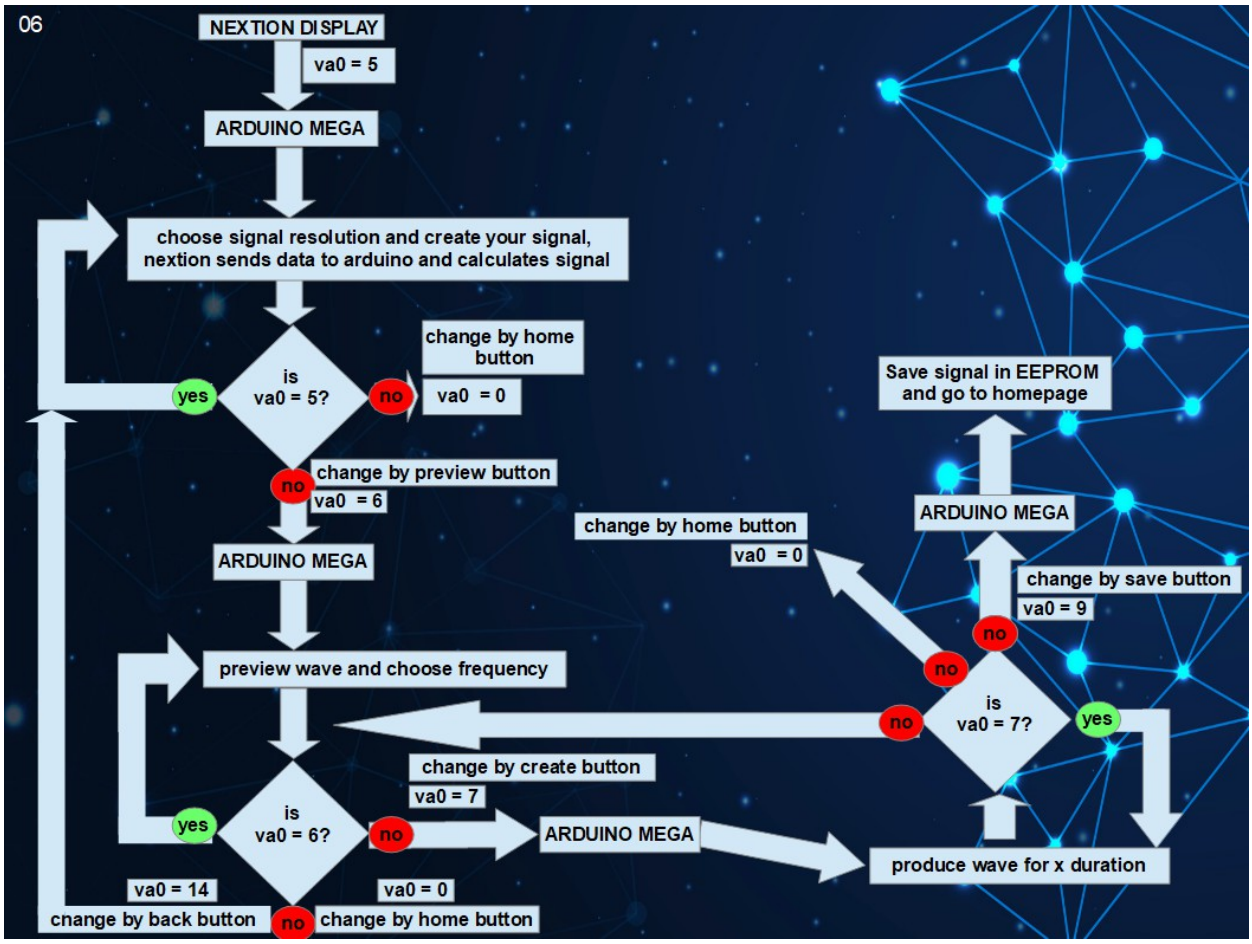
Create new page:



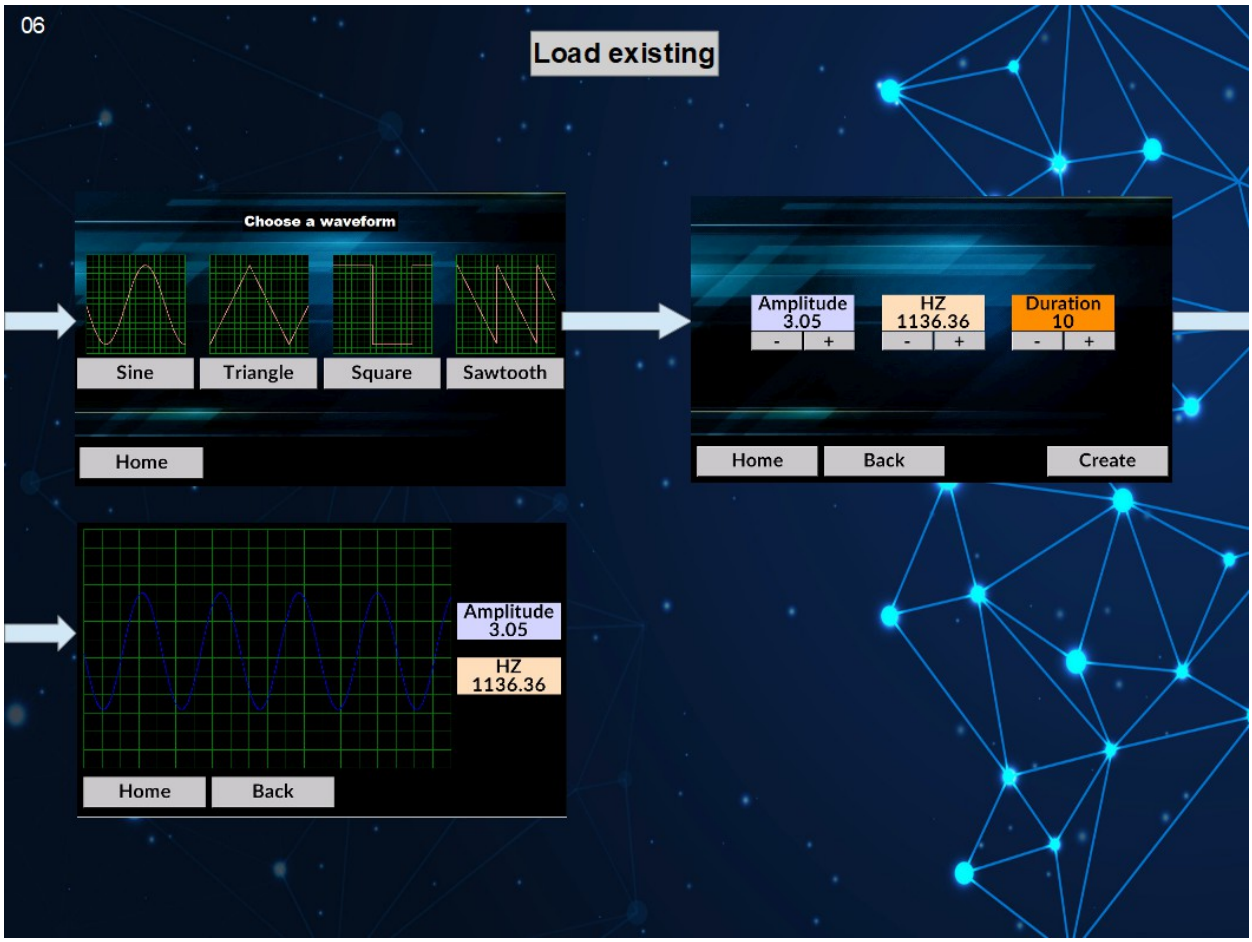
Η δημιουργία της custom κυματομορφής ξεκινάει επιλέγοντας από 2 έως 20 σημεία στην οθόνη. Η οθόνη αποθηκεύει τις συντεταγμένες σε προσωρινές μεταβλητές x και y για κάθε σημείο οπου x είναι ο χρόνος και y το πλάτος(τάση). Το κάθε pixel που αντιστοιχεί στον άξονα x είναι 1 microsecond καθυστέρηση. Η επιλογή preview θα στείλει τις μεταβλητές στο Arduino όπου θα τα αποθηκεύσει σε 2 πινάκες, ο ένας πίνακας αποθηκεύει την τάση και ο δεύτερος τον χρόνο. Αφού ο Arduino πάρει τις μεταβλητές θα τρέξει μια περίοδο ώστε να υπολογίσει την συχνότητα και να την στείλει στην οθόνη για να την δει ο χρήστης. Η επιλογή create θα παράξει το σήμα όπως το θέλουμε και η επιλογή save θα το αποθηκεύσει αν υπάρχει διαθέσιμη θέση στην μνήμη.

Διάγραμμα ροής λογισμικού κατά την διάρκεια σελίδας Create new:



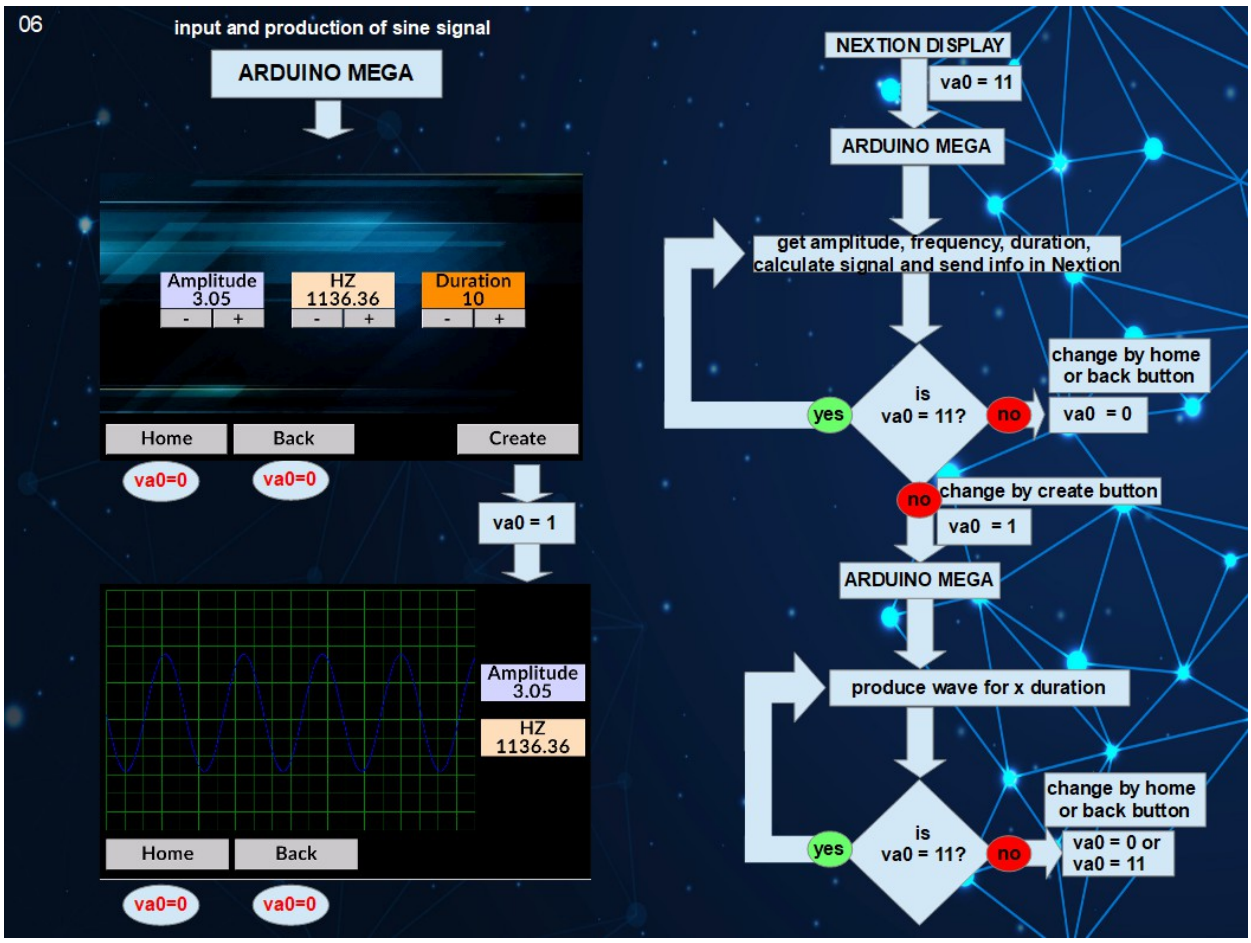

06
NEXTION DISPLAY
va0 = 5
ARDUINO MEGA
choose signal resolution and create your signal, nextion sends data to arduino and calculates signal
is va0 = 5?
yes
no
change by home button
va0 = 0
no
change by preview button
va0 = 6
ARDUINO MEGA
preview wave and choose frequency
is va0 = 6?
yes
no
change by create button
va0 = 7
ARDUINO MEGA
va0 = 14
cnange by back button
va0 = 0
no
change by home button
is va0 = 7?
yes
no
no
change by home button
va0 = 0
produce wave for x duration
ARDUINO MEGA
change by save button
va0 = 9
no
Save signal in EEPROM and go to homepage

Load existing page:

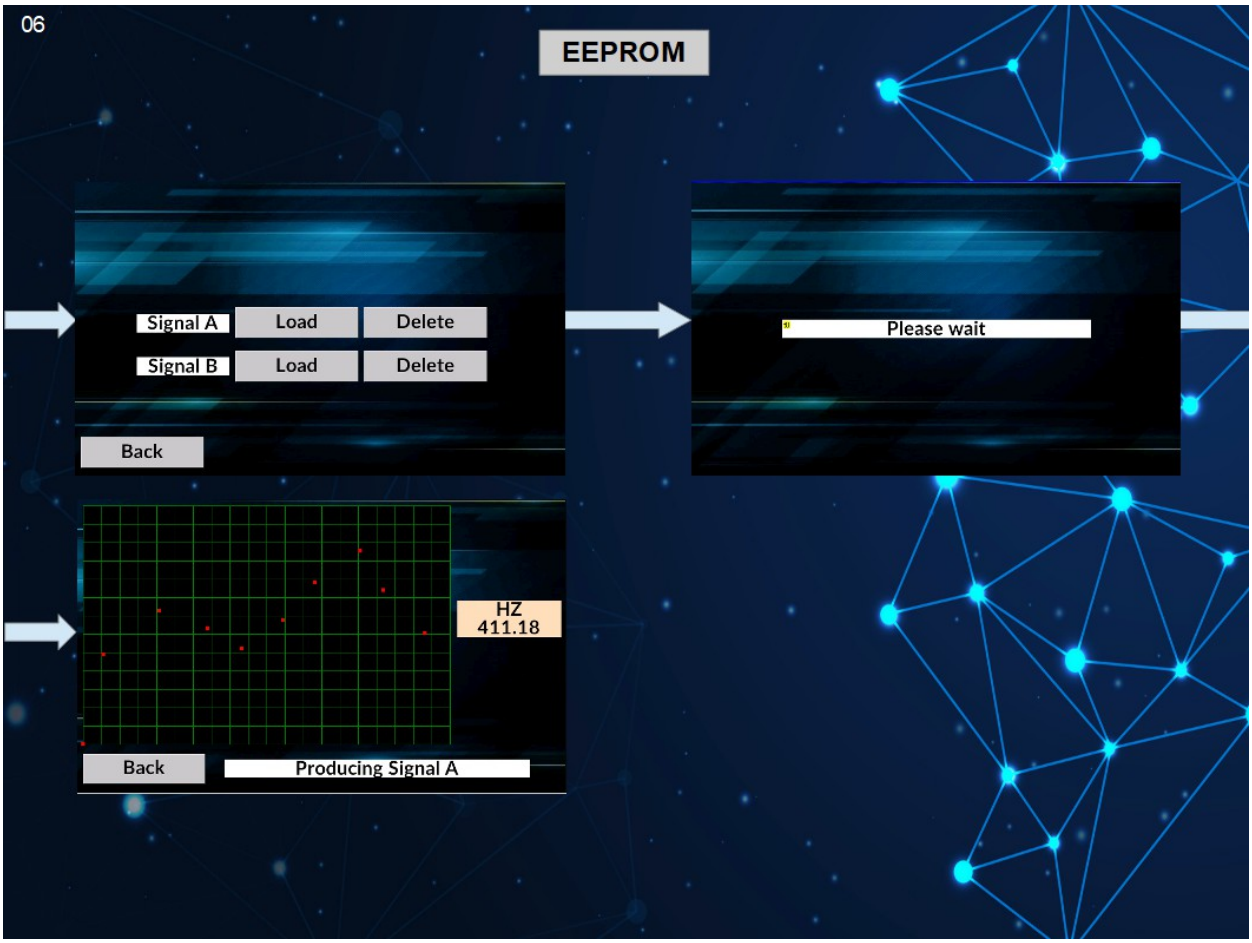


Η φόρτωση μιας υπάρχουσας ημιτονοειδούς, τριγωνικής, παλμικής και πριονωτής κυματομορφής πραγματοποιείται από την σελίδα load existing. Αφού ο χρήστης επιλέξει την επιθυμητή κυματομορφή το λογισμικό θα τον προτρέψει να αλλάξει τις τιμές του πλάτους, συχνότητας και διάρκειας του σήματος. Ταυτόχρονα η οθόνη θα στείλει τις τιμές στον Arduino όπου θα υπολογίσει την κυματομορφή. Η επιλογή create θα παράξει την κυματομορφή.

Διάγραμμα ροής λογισμικού κατά την διάρκεια σελίδας load existing:

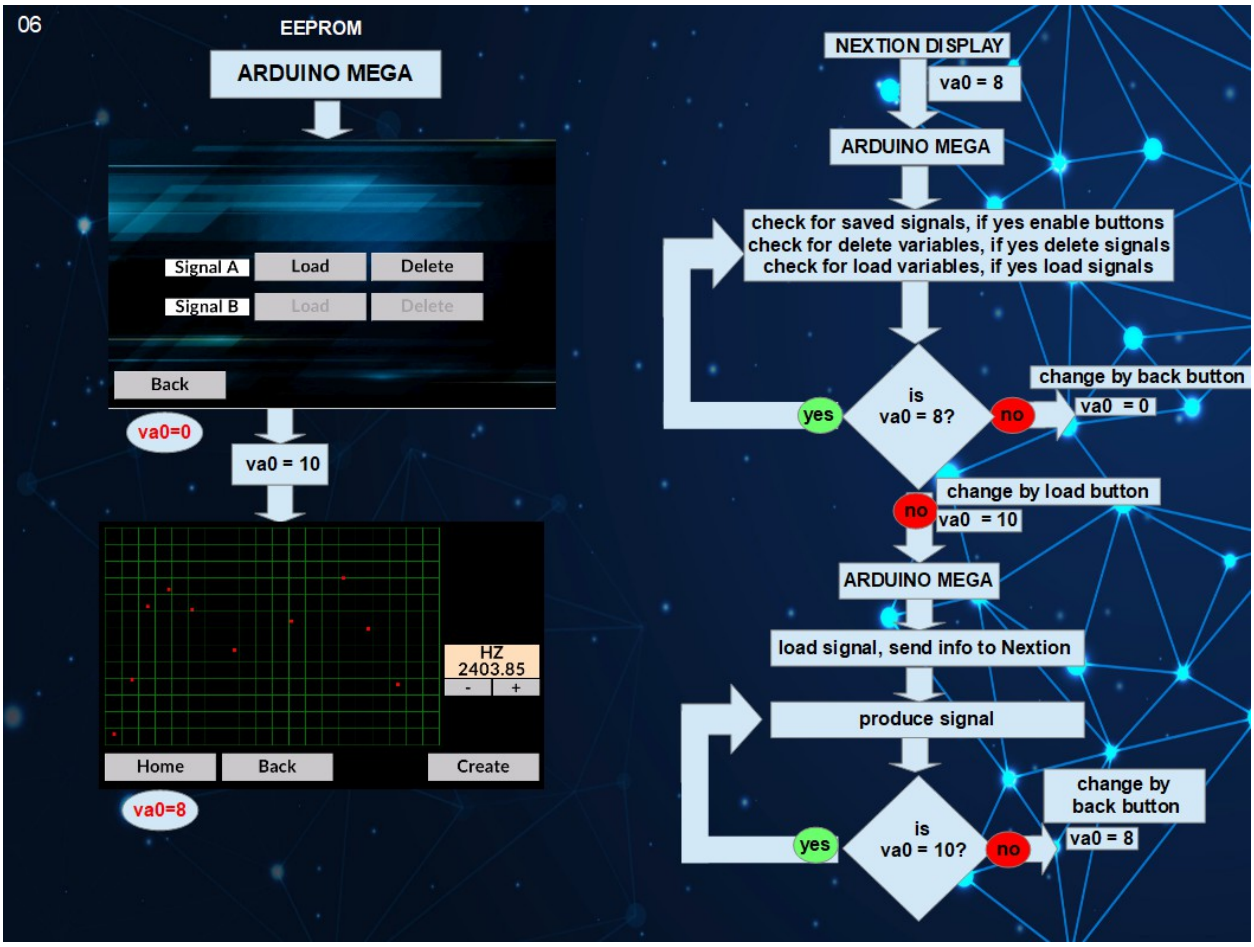


EEPROM page:



Στην σελίδα αυτή ο χρήστης μπορεί να επιλέξει την φόρτωση ήδη αποθηκευμένων σημάτων ή την διαγραφή τους. Ο Arduino ελέγχει τις θέσεις μνήμης που έχουν δεσμευθεί στην EEPROM και αν δει οτι υπάρχουν διαθέσιμα σήματα αλλάζει τις μεταβλητές lda και ldb στην οθόνη η οποία θα ενεργοποιήσει τα κουμπιά Load και Delete για την εκάστοτε κυματομορφή. Η λειτουργία Delete θα αλλάζει την μεταβλητή dela ή delb στην οθόνη και θα την στείλει στον Arduino όπου και θα εκτελέσει μια αρχικοποίηση τιμών στην θέση της αποθηκευμένης μεταβλητής. Η επιλογή Load θα φορτώσει τις τιμές του πλάτους και του χρόνου από την EEPROM και θα παράξει την αποθηκευμένη κυματομορφή.

Διάγραμμα ροής λογισμικού κατά την διάρκεια σελίδας EEPROM:



Τα libraries (βιβλιοθήκες) που χρησιμοποιήθηκαν είναι οι εξής:

1. **Adafruit_MCP4725**: Η βιβλιοθήκη αυτή χρησιμοποιείται για τη διασύνδεση με το DAC MCP4725 της Adafruit. Αυτή η βιβλιοθήκη επιτρέπει τον έλεγχο της τάσης εξόδου του DAC, καθιστώντας εύκολη την παραγωγή αναλογικών τάσεων στον Arduino.

   Βασικές λειτουργίες: Παρέχει λειτουργίες για τη ρύθμιση της τάσης εξόδου του DAC, την απενεργοποίηση του DAC και την εκτέλεση άλλων λειτουργιών που σχετίζονται με το DAC.

   Εφαρμογές: Αυτή η βιβλιοθήκη είναι εύχρηστη για εφαρμογές που απαιτούν ακριβή έλεγχο αναλογικής τάσης, όπως η παραγωγή ηχητικών σημάτων, ο έλεγχος ταχυτήτων κινητήρων ή η διασύνδεση με αναλογικούς αισθητήρες.

2. **Wire**: Η βιβλιοθήκη Wire αποτελεί βασικό μέρος της υλοποίησης του πρωτοκόλλου επικοινωνίας I2C του Arduino. Επιτρέπει στο Arduino σας να επικοινωνεί με συσκευές I2C, συμπεριλαμβανομένων αισθητήρων, οθονών και άλλων μικροελεγκτών.

   Βασικές λειτουργίες: Το πρόγραμμα είναι ένα από τα βασικά εργαλεία που χρησιμοποιούνται για τη δημιουργία και τη λειτουργία του: Η βιβλιοθήκη Wire παρέχει λειτουργίες τόσο για την επικοινωνία I2C master (π.χ. Arduino ως master συσκευή) όσο και για την επικοινωνία I2C slave (π.χ. Arduino ως slave συσκευή). Περιλαμβάνει λειτουργίες για την έναρξη της επικοινωνίας, την αποστολή και λήψη δεδομένων και το χειρισμό εργασιών που σχετίζονται με το I2C.

   Εφαρμογές: Η βιβλιοθήκη Wire χρησιμοποιείται ευρέως σε έργα που περιλαμβάνουν πολλαπλές συσκευές που επικοινωνούν μέσω του διαύλου I2C, όπως δίκτυα αισθητήρων, οθόνες και ρολόγια πραγματικού χρόνου.

3. **EasyNextionLibrary**: Η βιβλιοθήκη EasyNextionLibrary χρησιμοποιείται για την επικοινωνία και την αλληλεπίδραση με τις οθόνες Nextion.

   Βασικές λειτουργίες: Η βιβλιοθήκη απλοποιεί την επικοινωνία μεταξύ του Arduino και της οθόνης Nextion, επιτρέποντάς την αποστολή και λήψη δεδομένων, την ενημέρωση των στοιχείων της οθόνης και την ανταπόκριση σε συμβάντα αφής στην οθόνη. Εφαρμογές: Χρησιμοποιείται συνήθως σε έργα όπου απαιτείται ένα φιλικό προς το χρήστη γραφικό περιβάλλον, όπως συστήματα οικιακού αυτοματισμού, συσκευές IoT και προσαρμοσμένες διεπαφές χρήστη για διάφορες εφαρμογές.

4. **EEPROM**: Η βιβλιοθήκη EEPROM επιτρέπει την ανάγνωση και την εγγραφή δεδομένων στην EEPROM (Electrically Erasable Programmable Read-Only Memory) του Arduino. Η EEPROM παρέχει μόνιμη αποθήκευση, επιτρέποντάς την αποθήκευση δεδομένων που παραμένουν ακόμη και όταν το Arduino απενεργοποιείται.

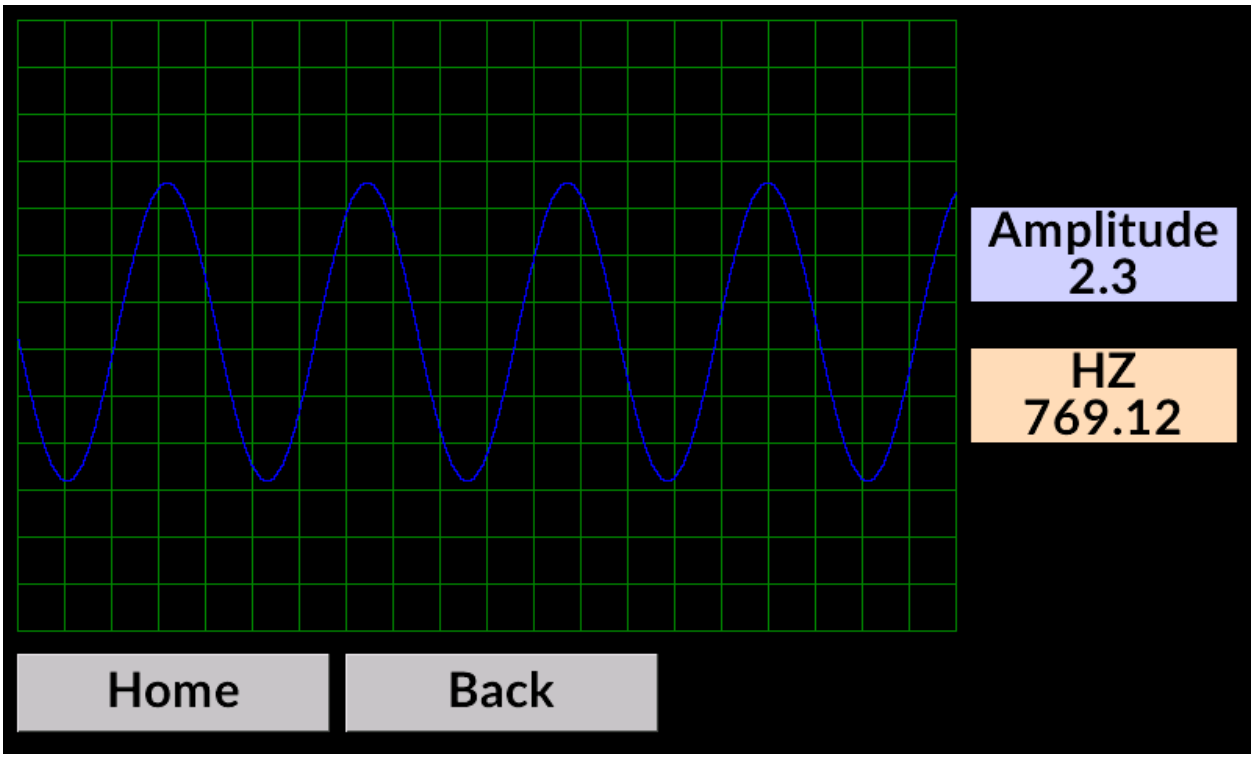
   Βασικές λειτουργίες: Αυτή η βιβλιοθήκη περιλαμβάνει λειτουργίες για την ανάγνωση και εγγραφή μεμονωμένων bytes ή μεγαλύτερων δομών δεδομένων από και προς τη μνήμη EEPROM.

   Εφαρμογές: Η βιβλιοθήκη EEPROM είναι χρήσιμη για την αποθήκευση ρυθμίσεων διαμόρφωσης, δεδομένων βαθμονόμησης και άλλων πληροφοριών που πρέπει να διατηρούνται μεταξύ των κύκλων τροφοδοσίας. Χρησιμοποιείται συνήθως σε εφαρμογές όπου απαιτείται η διατήρηση δεδομένων, όπως η καταγραφή θερμοκρασίας ή οι ρυθμίσεις χρήστη σε ενσωματωμένα συστήματα.

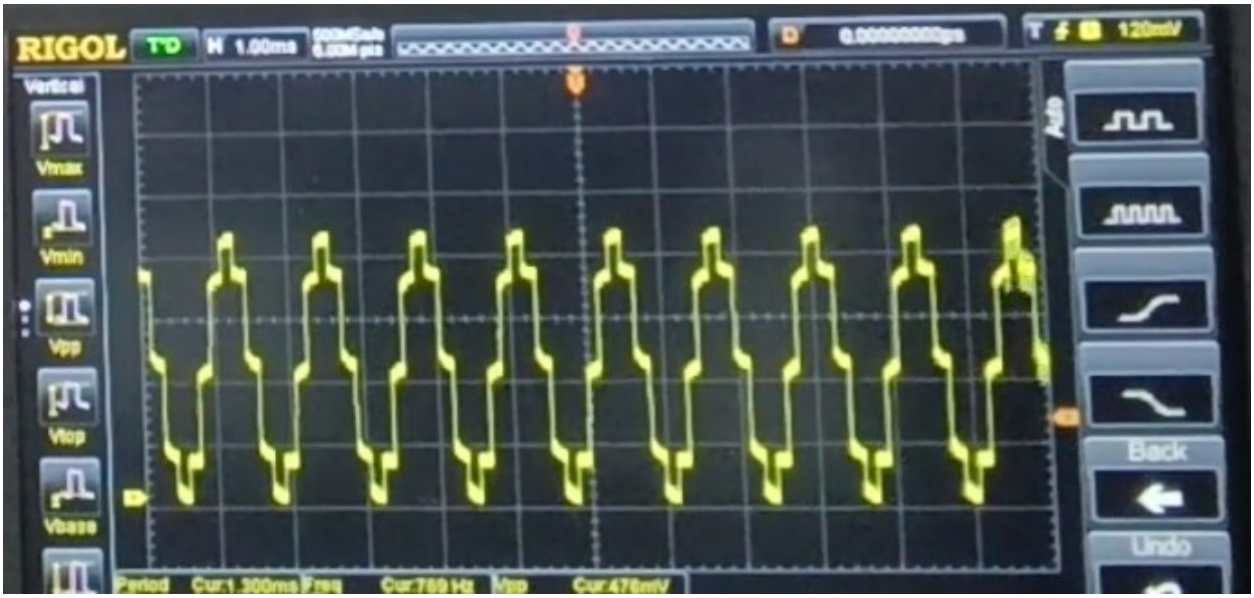
Αυτές οι βιβλιοθήκες βελτιώνουν τη λειτουργικότητα των εφαρμογών του Arduino απλοποιώντας την επικοινωνία με εξωτερικό υλικό και παρέχοντας χρήσιμα βοηθητικά προγράμματα για διάφορες εργασίες. Ανάλογα με τις απαιτήσεις του έργου, μπορεί να γίνει ενσωμάτωση των βιβλιοθηκών για την βελτίωση της ανάπτυξης και της επέκτασης των δυνατοτήτων των εφαρμογών.

9. Δοκιμές και αποτελέσματα

Δημιουργία ημιτονοειδούς σήματος:

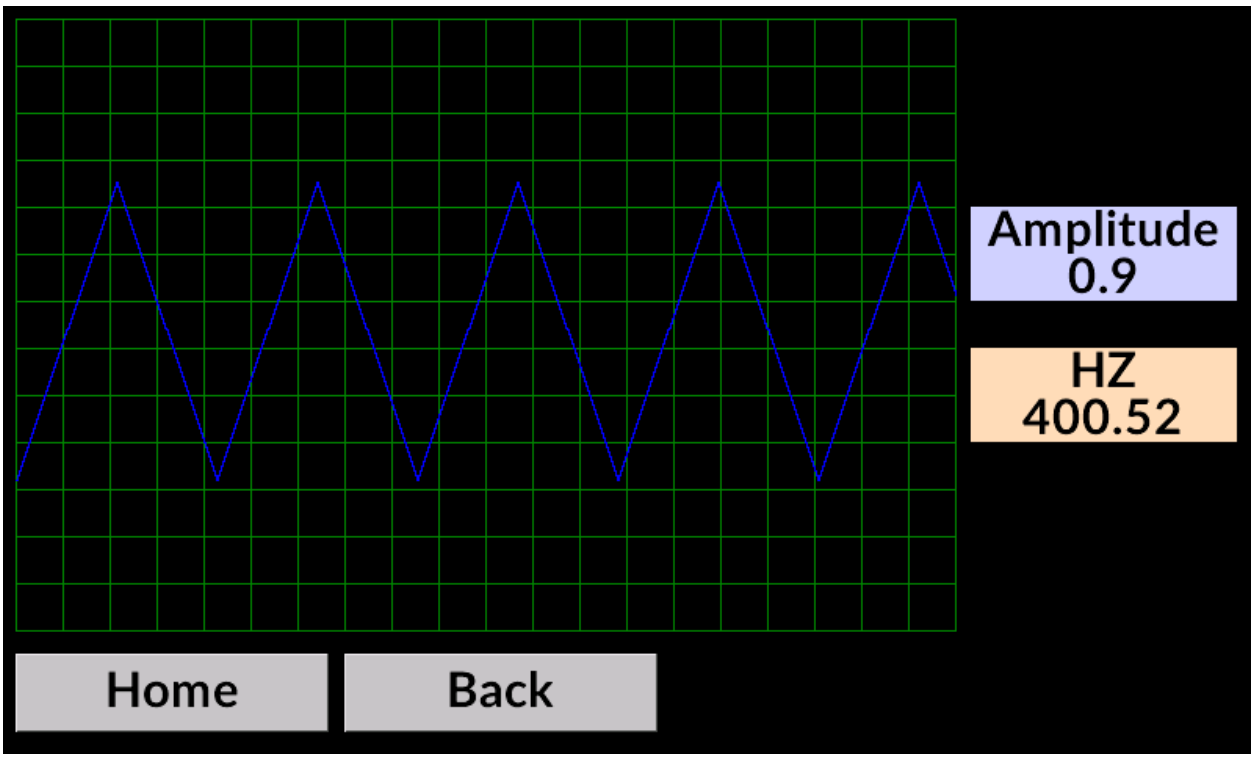


Αποτελέσματα:



Σήμα 8 σημείων, πλάτους 2.3V και συχνότητας 769Hz.

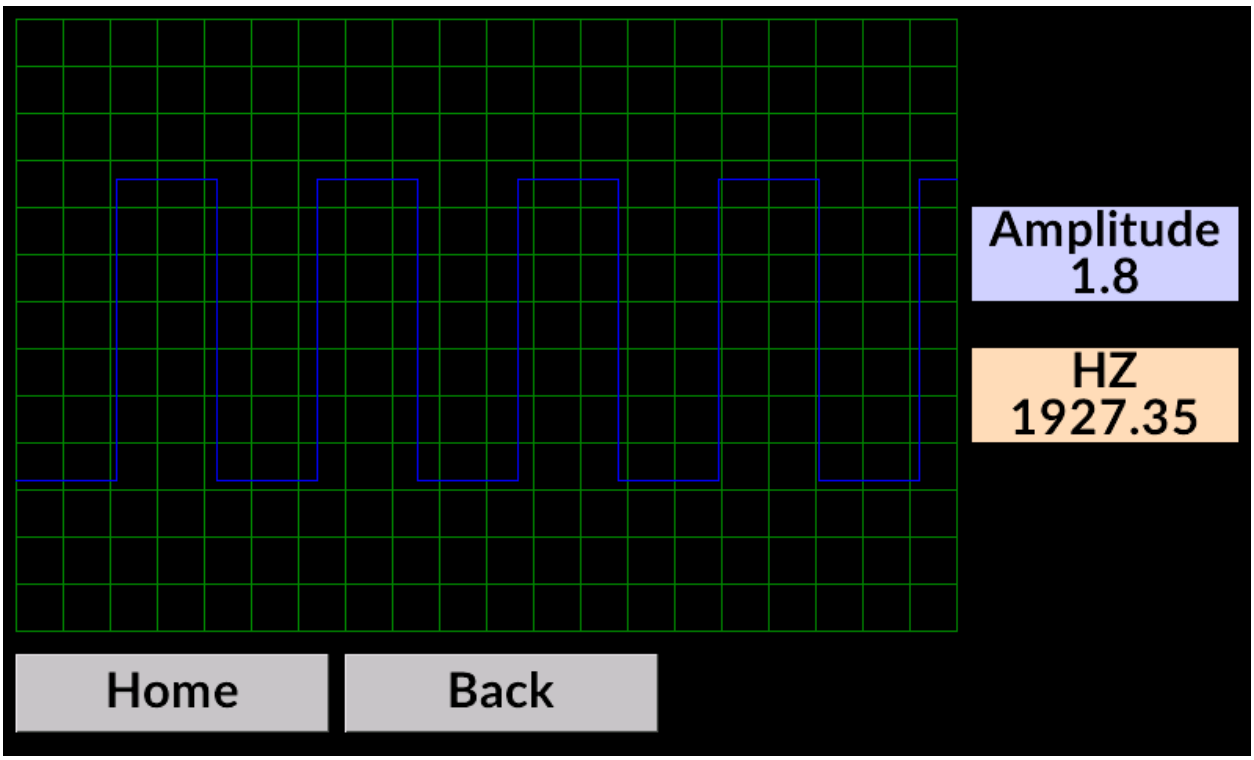
Δημιουργία τριγωνικού σήματος:



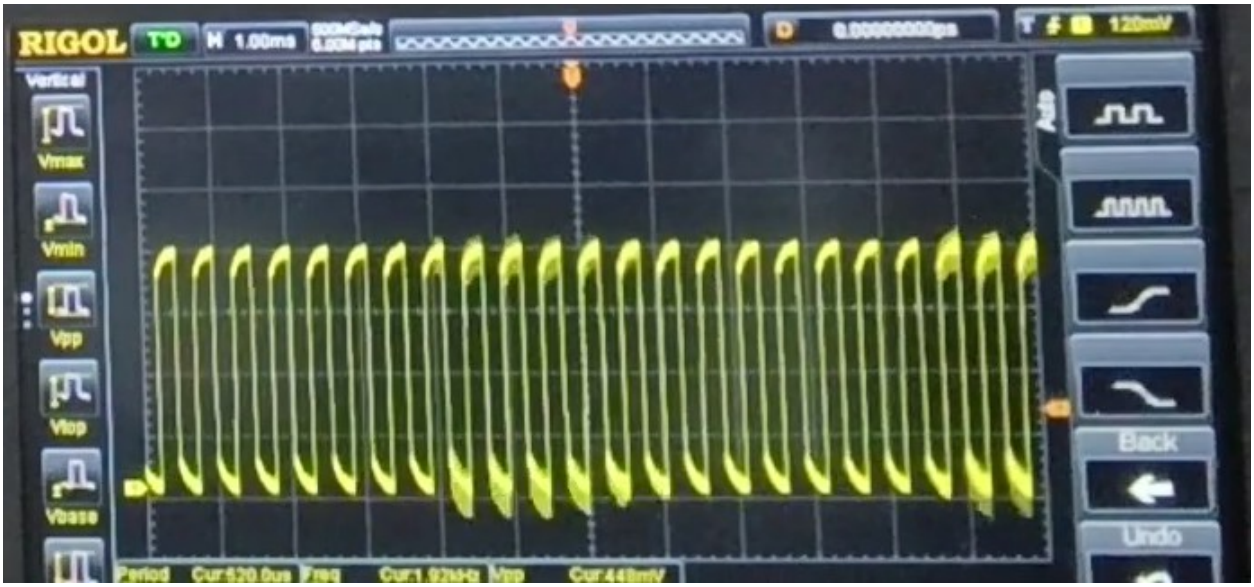
Αποτελέσματα:

Σήμα 6 σημείων, πλατους 0.9V και συχνότητας 400Hz.

Δημιουργία παλμικού σήματος:

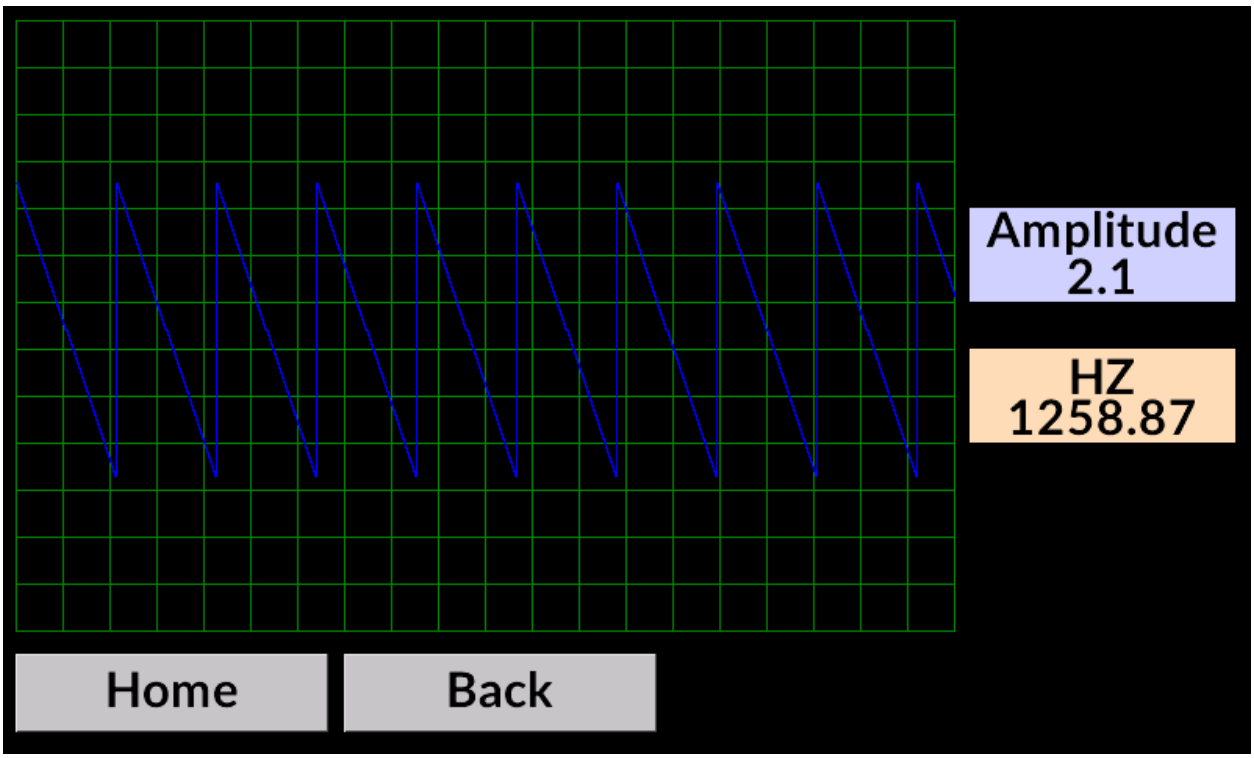


Αποτελέσματα:

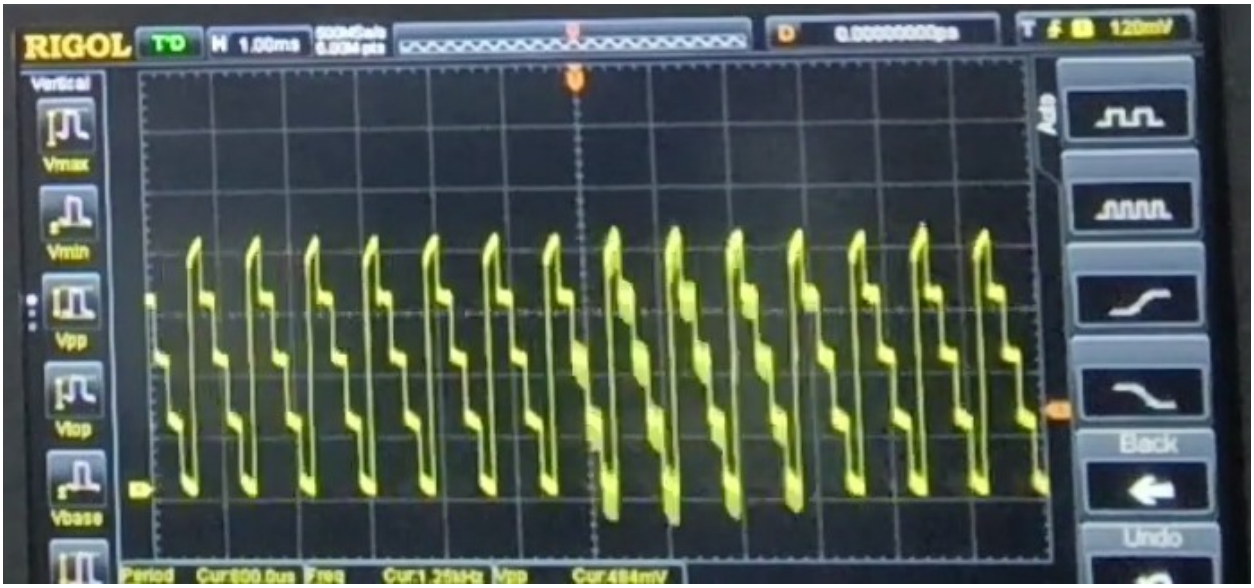


Σήμα 2 σημείων, πλατους 1,8V και συχνότητας 1.92kHz.

Δημιουργία πριονωτού σήματος:

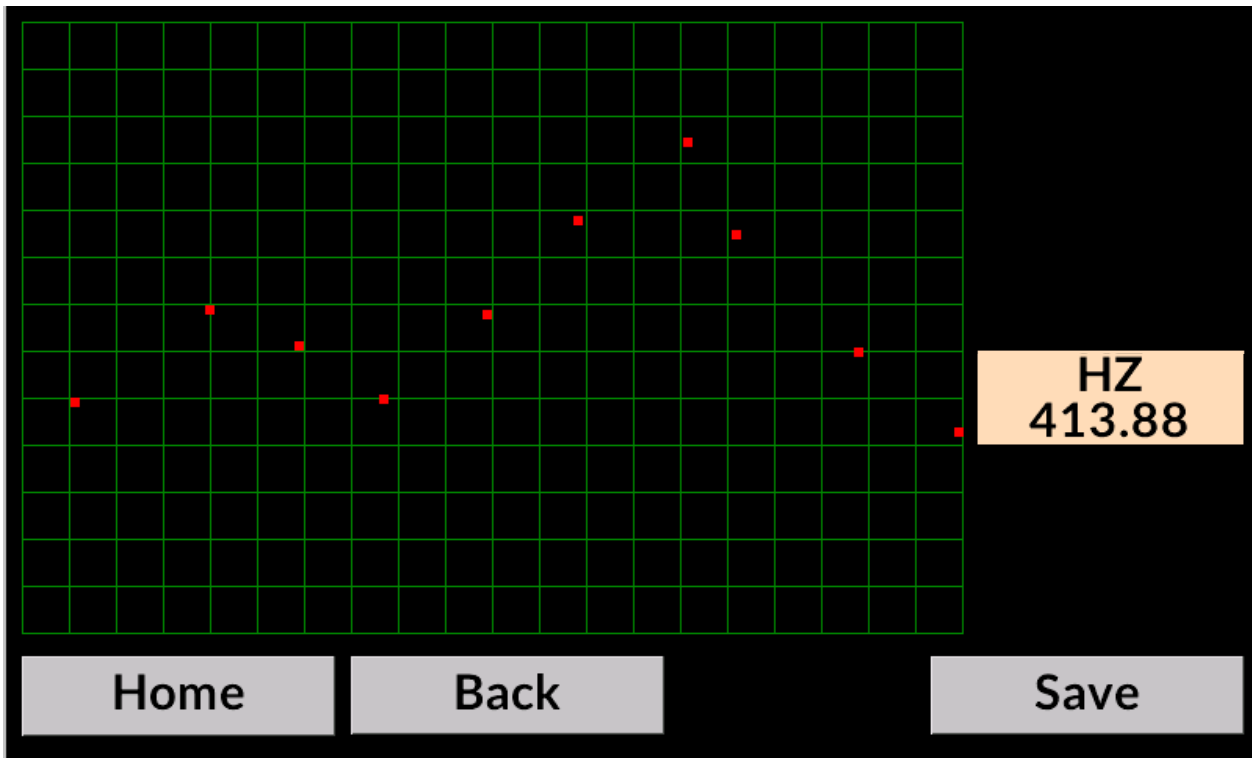


Αποτελέσματα:



Σήμα 5 σημείων, 2,1V πλάτους και συχνότητας 1,25kHz.

Δημιουργία custom σήματος:



Αποτελέσματα:

Σήμα 10 σημείων, πλάτους Peak to Peak 1.9V και συχνότητας 413Hz.

10. Τεχνική περιγραφή κώδικα

Ο κώδικας που γράφεται στο Arduino IDE βασίζεται στη γλώσσα προγραμματισμού Arduino, η οποία είναι μια απλοποιημένη έκδοση της C/C++. Ο κώδικας που εκτείνεται σε 424 γραμμές, λειτουργεί ως γεννήτρια αυθαίρετων κυματομορφών. Χρησιμοποιώντας βιβλιοθήκες όπως οι Adafruit_MCP4725, Wire, EasyNextionLibrary και EEPROM, ο κώδικας διαχειρίζεται αναλογικά σήματα και διεκπεραιώνει εργασίες που σχετίζονται με την οθόνη Nextion και το MCP4725. Χρησιμοποιείται μια σειρά μεταβλητών, επιτρέποντας τη ρύθμιση τόσο του πλάτους όσο και της συχνότητας του σήματος. Αυτός ο έλεγχος παρέχει ακριβή επιρροή στα χαρακτηριστικά των παραγόμενων κυματομορφών. Επιπλέον, χρησιμοποιούνται πίνακες για τη διαχείριση των συντεταγμένων, των πλατών και των συχνοτήτων του Nextion, διευκολύνοντας την προσαρμογή των χαρακτηριστικών κυματομορφής που εμφανίζονται σε έναν παλμογράφο. Η συνάρτηση void setup() είναι υπεύθυνη για την αρχικοποίηση της σειριακής επικοινωνίας, επιτρέποντας την απρόσκοπτη μετάδοση δεδομένων μεταξύ της πλακέτας Arduino και της οθόνης Nextion. Επιπλέον, η συνάρτηση setup εγκαθιστά το πρωτόκολλο επικοινωνίας I2C (Inter-Integrated Circuit) με το MCP4725, εξασφαλίζοντας την αποτελεσματική επικοινωνία μεταξύ του Arduino και του μετατροπέα ψηφιακού σε αναλογικό MCP4725. Αυτή η επικοινωνία επιτρέπει τον ακριβή έλεγχο της αναλογικής εξόδου, διευκολύνοντας έτσι τη δημιουργία των επιθυμητών κυματομορφών.

Εντός της συνάρτησης void loop(), ο κώδικας χρησιμοποιεί διάφορους βρόχους while που εκτελούν συγκεκριμένες εργασίες με βάση την τιμή μιας μεταβλητής με όνομα home.va0.val. Αυτή η μεταβλητή βρίσκεται στην οθόνη του Nextion και αλλάζει δυναμικά όταν πατηθεί ένα αντίστοιχο κουμπί στο λογισμικό της οθόνης. Οι βρόχοι while έχουν σχεδιαστεί για να ανταποκρίνονται σε διαφορετικές τιμές της home.va0.val, ενεργοποιώντας ξεχωριστές λειτουργίες και προσαρμόζοντας ανάλογα τα χαρακτηριστικά της κυματομορφής, τα πλάτη ή τις συχνότητες. Αυτός ο δυναμικός μηχανισμός ελέγχου επιτρέπει προσαρμογές σε πραγματικό χρόνο και προσαρμογή των ιδιοτήτων της κυματομορφής, παρέχοντας μια ευέλικτη και διαδραστική εμπειρία χρήστη. Επιπλέον, η συνάρτηση void loop() μπορεί να κατηγοριοποιηθεί περαιτέρω σε τέσσερα διακριτά τμήματα, καθένα από τα οποία περιέχει βρόχους while που

εξυπηρετούν συγκεκριμένους σκοπούς. Η πρώτη κατηγορία περιλαμβάνει βρόχους while που δημιουργούν σήματα και μεταδίδουν τις απαραίτητες πληροφορίες των σημάτων στο Arduino. Η δεύτερη κατηγορία περιλαμβάνει βρόχους while που είναι υπεύθυνοι για τη παραγωγη σημάτων με βάση τις πληροφορίες που λαμβάνονται από την πρώτη κατηγορία, παράγοντας έτσι τις επιθυμητές κυματομορφές. Η τρίτη κατηγορία αποτελείται από βρόχους while που έχουν σχεδιαστεί για τη δημιουργία προσαρμοσμένων σημάτων, επιτρέποντας τη δημιουργία εξειδικευμένων κυματομορφών με βάση παραμέτρους που καθορίζονται από τον χρήστη. Τέλος, η τέταρτη κατηγορία περιλαμβάνει βρόχους while που ελέγχουν, αποθηκεύουν ή φορτώνουν σήματα από την EEPROM, εξασφαλίζοντας τη διατήρηση και ανάκτηση σημαντικών δεδομένων κυματομορφών για μελλοντική χρήση ή αναφορά. Αυτή η οργανωμένη προσέγγιση διευκολύνει τη συστηματική και αποτελεσματική εκτέλεση των εργασιών στο πλαίσιο της διαδικασίας παραγωγής κυματομορφών, ενισχύοντας τη συνολική λειτουργικότητα και ευελιξία του συστήματος.

11. Συμπεράσματα και προτάσεις

Η ανάπτυξη της γεννήτριας αυθαίρετων κυματομορφών με τη χρήση του Arduino, της οθόνης Nextion και του MCP4725 αποδείχθηκε γενικά επιτυχία, καθώς πληρούσε όλες τις απαραίτητες απαιτήσεις. Το σύστημα παρήγαγε αποτελεσματικά έτοιμες και προσαρμοσμένες κυματομορφές, επιτρέποντας τον ακριβή έλεγχο του πλάτους και της συχνότητας του σήματος. Ωστόσο, ένα αξιοσημείωτο μειονέκτημα της συσκευής είναι οι σχετικά αργές δυνατότητες επεξεργασίας σήματος, με το ταχύτερο σήμα, παλμικό, να λειτουργεί σε συχνότητα περίπου 2kHz. Αυτός ο περιορισμός μπορεί να επηρεάσει την καταλληλότητα της συσκευής για εφαρμογές που απαιτούν επεξεργασία σήματος υψηλής συχνότητας. Επιπλέον, η οθόνη, γνωστή για την ευκολία δημιουργίας μιας φιλικής προς τον χρήστη γραφικής διεπαφής, αντιμετώπισε προκλήσεις λόγω της απουσίας πρόσθετων pin για διακοπές, με αποτέλεσμα να είναι αναγκαία η επικοινωνία μεταξύ των στοιχείων αποκλειστικά μέσω polling. Κατά συνέπεια, αυτή η μέθοδος επικοινωνίας θα μπορούσε να οδηγήσει σε βραδύτερους χρόνους απόκρισης και σε αυξημένη επιβάρυνση της επεξεργασίας. Για την περαιτέρω βελτίωση των επιδόσεων της συσκευής, μπορεί να είναι απαραίτητη η υλοποίηση ενός φίλτρου για την αποτελεσματική εξάλειψη κάθε πιθανού θορύβου που υπάρχει στα σήματα και την ομαλοποίηση των παραγόμενων κυματομορφών, εξασφαλίζοντας μια πιο ακριβή έξοδο για διάφορες εφαρμογές. Παρά τις προκλήσεις αυτές, η συσκευή επέδειξε σημαντικές δυνατότητες και ευελιξία στη δημιουργία προσαρμοσμένων κυματομορφών για ποικίλες εφαρμογές. Για την αντιμετώπιση των περιορισμών που παρουσιάστηκαν κατά τη διαδικασία ανάπτυξης, μια πολλά υποσχόμενη οδός για τη βελτίωση των δυνατοτήτων της συσκευής περιλαμβάνει την ενσωμάτωση ενός πρόσθετου επεξεργαστή αφιερωμένου αποκλειστικά στην παραγωγή σημάτων. Αναθέτοντας αυτό το έργο σε έναν εξειδικευμένο επεξεργαστή, το σύστημα θα μπορούσε να επιτύχει υψηλότερες ταχύτητες επεξεργασίας σήματος και βελτιωμένη συνολική απόδοση. Σε αυτή τη ρύθμιση, το Arduino θα χειριζόταν κυρίως την επικοινωνία μεταξύ της οθόνης και του δευτερεύοντος επεξεργαστή, βελτιώνοντας έτσι τη διαδικασία ανταλλαγής δεδομένων και μειώνοντας τις πιθανές συμφορήσεις. Επιπλέον, η ενσωμάτωση μιας οθόνης με συμπληρωματικά pin για την διαχείριση των interrupt θα βελτίωνε σημαντικά την απόκριση της συσκευής και την αλληλεπίδραση σε

πραγματικό χρόνο, εξασφαλίζοντας μια ομαλότερη εμπειρία χρήστη και επιτρέποντας την υλοποίηση πιο σύνθετων λειτουργιών. Αυτές οι προτεινόμενες βελτιώσεις θα ενισχύσουν την αποδοτικότητα της συσκευής, ανοίγοντας το δρόμο για τη δημιουργία μιας πιο στιβαρής και ευέλικτης γεννήτριας αυθαίρετων κυματομορφών προσαρμοσμένης στις απαιτήσεις ποικίλων εφαρμογών.

12.Βιβλιογραφία

https://nextion.tech/instruction-set/
https://www.arduino.cc/en/Guide
https://www.arduino.cc/reference/en/language/functions/communication/wire/
https://docs.arduino.cc/learn/communication/wire
https://docs.arduino.cc/learn/built-in-libraries/eeprom
https://docs.arduino.cc/tutorials/due/simple-waveform-generator
https://en.wikipedia.org/wiki/Arduino
https://seithan.com/Easy-Nextion-Library/
https://github.com/Seithan/EasyNextionLibrary
https://github.com/adafruit/Adafruit_MCP4725
https://www.youtube.com/@CheapControls
https://cheapcontrols.com/
Σημειώσεις ΠΜΣ Ρ101

13.Παράρτημα κώδικα

```
#include <Adafruit_MCP4725.h>          //Library για DAC MCP4725
#include <Wire.h>                      //I2C Library επικοινωνίας
#include "EasyNextionLibrary.h"        //Nextion Library
#include <EEPROM.h>                    //EEPROM library
#define MCP4725_ADDRESS 0x60
Adafruit_MCP4725 dac;
EasyNex myNex(Serial);
//==================variables=================
String endChar = String(char(0xff)) + String(char(0xff)) + String(char(0xff));//μήνυμα τερματισμού οθόνης
Nextion
float start, end, total;//float μεταβλητές
unsigned int cnt;
uint16_t a, b, c, d, e, f, g, h, i;//uint16_t μεταβλητές για το υπολογισμο του πλάτους του σήματος
int rep, duration, finaldelay = 0, pulsewidthup, pulsewidthdown, amplitude, xy = 20;//integer μεταβλητές
int delays[20] = {0};//initialization πίνακα καθυστέρησης
int mapedvoltage[20] = {0};//initialization πίνακα τάσεων
int xcoords[20] = {0};//πινακας για αποθηκευση των x συντεταγμένων (delay)
int ycoords[20] = {0};//πινακας για αποθηκευση των y συντεταγμένων (voltage)
// sine     lookuptable  2048a, 2831b, 3495c, 3939d, 4095e, 3939d, 3495c, 2831b, 2048a, 1264f, 600g,
156h, 0i, 156h, 600g, 1264f
// triangle lookuptable  0a, 1024b, 2048c, 3071d, 4095e, 3071d, 2048c, 1024b
// pulse    lookuptable  0, 4095
// sawtooth lookuptable  0a, 1024b, 2048c, 3071d, 4095e
//===========================================
void setup()           {
 Serial.begin(9600);//έναρξη σειριακής επικοινωνίας
 Wire.begin();//έναρξη I2C επικοινωνίας
 dac.begin(MCP4725_ADDRESS);
}
void sinewave16bit()   {//function για ημιτονοειδές σήμα / 16bit 385hz
 dac.setVoltage(a, false);//συνάρτηση του library Adafruit οπου δίνει στην I2C έξοδο την τάση της
μεταβλητής a και δεν αποθηκεύεται στην EEPROM του DAC (false)
 delayMicroseconds(cnt);//καθυστέρηση cnt
 dac.setVoltage(b, false);
 delayMicroseconds(cnt);
 dac.setVoltage(c, false);
 delayMicroseconds(cnt);
 dac.setVoltage(d, false);
 delayMicroseconds(cnt);
 dac.setVoltage(e, false);
 delayMicroseconds(cnt);
 dac.setVoltage(d, false);
 delayMicroseconds(cnt);
 dac.setVoltage(c, false);
 delayMicroseconds(cnt);
 dac.setVoltage(b, false);
 delayMicroseconds(cnt);
 dac.setVoltage(a, false);
 delayMicroseconds(cnt);
 dac.setVoltage(f, false);
 delayMicroseconds(cnt);
 dac.setVoltage(g, false);
 delayMicroseconds(cnt);
 dac.setVoltage(h, false);
 delayMicroseconds(cnt);
 dac.setVoltage(i, false);
 delayMicroseconds(cnt);
 dac.setVoltage(h, false);
```

```
 delayMicroseconds(cnt);
 dac.setVoltage(g, false);
 delayMicroseconds(cnt);
 dac.setVoltage(f, false);
 delayMicroseconds(cnt);
}
void sinewave8bit()    {//function για ημιτονοειδές σήμα / 8bit 781hz
 dac.setVoltage(a, false);//συνάρτηση του library Adafruit οπου δίνει στην I2C έξοδο την τάση της
μεταβλητής a και δεν αποθηκεύεται στην EEPROM του DAC (false)
 delayMicroseconds(cnt);//καθυστέρηση cnt
 dac.setVoltage(c, false);
 delayMicroseconds(cnt);
 dac.setVoltage(e, false);
 delayMicroseconds(cnt);
 dac.setVoltage(c, false);
 delayMicroseconds(cnt);
 dac.setVoltage(a, false);
 delayMicroseconds(cnt);
 dac.setVoltage(g, false);
 delayMicroseconds(cnt);
 dac.setVoltage(i, false);
 delayMicroseconds(cnt);
 dac.setVoltage(g, false);
 delayMicroseconds(cnt);
}
void triangle()        {//function για τριγωνικό σήμα / 8bit 762hz
 dac.setVoltage(a, false);
 delayMicroseconds(cnt);
 dac.setVoltage(b, false);
 delayMicroseconds(cnt);
 dac.setVoltage(c, false);
 delayMicroseconds(cnt);
 dac.setVoltage(d, false);
 delayMicroseconds(cnt);
 dac.setVoltage(e, false);
 delayMicroseconds(cnt);
 dac.setVoltage(d, false);
 delayMicroseconds(cnt);
 dac.setVoltage(c, false);
 delayMicroseconds(cnt);
 dac.setVoltage(b, false);
 delayMicroseconds(cnt);
}
void pulse()           {//function για τετραγωνικό σήμα / 2bit 2890hz
 dac.setVoltage(0, false);
 delayMicroseconds(pulsewidthdown);
 dac.setVoltage(amplitude, false);
 delayMicroseconds(pulsewidthup);
}
void sawtooth()        {//function για πριονωτό σήμα / 5bit 1225hz
 dac.setVoltage(a, false);
 delayMicroseconds(cnt);
 dac.setVoltage(b, false);
 delayMicroseconds(cnt);
 dac.setVoltage(c, false);
 delayMicroseconds(cnt);
 dac.setVoltage(d, false);
 delayMicroseconds(cnt);
 dac.setVoltage(e, false);
 delayMicroseconds(cnt);
}
void customwave()      {//custom σημα
```

```
 for (int i = 0; i <= xy-1; i++){//επανάληψη για μήκος μεαβλητής xy οπου xy είναι η ανάλυση του σήματος (σημεία)
  delayMicroseconds(delays[i]+cnt);//καθυστέρηση απο την αρχή του άξονα y έως το πρώτο σημείο του σήματος
  dac.setVoltage(mapedvoltage[i], false);//τάση mapedvoltage[i] στην έξοδο
 }
 delayMicroseconds(620 - xcoords[xy-1]);//τελική καθυστέρηση απο το τελευταίο σημείο του σήματος μέχρι το τέλος του άξονα x
}

//||-----------------UI----------------------||
//||home.va0.val = 0  home, loading          ||
//||home.va0.val = 11 sineinfo               ||    signal 1 xcoords address          100
//||home.va0.val = 1  sine                   ||    signal 1 ycoords address          200
//||home.va0.val = 12 triangleinfo           ||    signal 1 mapedvoltage address     300
//||home.va0.val = 2  triangle               ||    signal 1 delay address            400
//||home.va0.val = 13 pulseinfo              ||    signal 1 khz address              450
//||home.va0.val = 3  pulse                  ||    signal 1 xy(μήκος) address        460
//||home.va0.val = 14 sawtoothinfo           ||    signal 2 xcoords address          500
//||home.va0.val = 4  sawtooth               ||    signal 2 ycoords address          600
//||home.va0.val = 5  createnew              ||    signal 2 mapedvoltage address     700
//||home.va0.val = 6  wavepreview            ||    signal 2 delay address            800
//||home.va0.val = 7  customwave             ||    signal 2 khz address              850
//||home.va0.val = 8  eeprom / delete /load  ||    signal 2 xy(μήκος) address        860
//||home.va0.val = 9  save                   ||
//||home.va0.val = 10 customload             ||
//||-----------------------------------------||
void loop() {

while (myNex.readNumber("home.va0.val") == 0)   {//home, loading page
 cnt = 0;//initialize μετρητή cnt
 amplitude = 2500;//initialize amplitude
 pulsewidthdown = 10;//initialize pulsewidthdown
 pulsewidthup = 10;//initialize pulsewidthup
 duration = 0;//initialize duration
 myNex.writeStr("home.tempfreq.txt", String(0));//initialize της μεταβλητής tempfreq στο page Home του Nextion
}
while (myNex.readNumber("home.va0.val") == 11)  {//sineinfo 8bit page
 amplitude = myNex.readNumber("home.volt.val");//λήψη μεταβλητης πλάτους απο Nextion 0-4096
 cnt = myNex.readNumber("home.freq.val");//λήψη μεταβλητής συχνότητας απο Nextion 0-200
 duration = myNex.readNumber("home.duration.val");//λήψη μεταβλητής συχνότητας απο Nextion 1-10000
 rep = duration/0.001305483;                                    //μετατροπή χρόνου σήματος σε επαναλήψεις
 a = amplitude/1.999;                                           ///
 b = amplitude/1.446;                                           ////
 c = amplitude/1.171;                                           /////
 d = amplitude/1.039;                                           //////
 e = amplitude;                                                 //Δημιουργία σημείων ημιτονοειδούς σήματος
 f = amplitude/3.239;                                           //////
 g = amplitude/6.825;                                           /////
 h = amplitude/26.25;                                           ////
 i = amplitude*0;                                               ///
 float amp = (5*float(amplitude))/4096;//υπολογισμός τάσης εξόδου
 String Samp = String(amp,2);//αποθήκευση της μεταβλητής amp στην μεταβλήτή Samp και μετατροπή σε string
 start = (micros());//έναρξη χρονομέτρησης της function sinewave
 sinewave8bit();//εκτέλεση της function sinewave
 end = (micros());//τέλος χρονομέτρησης της function sinewave
 total = (1/(end-start))*1000*1000;//υπολογισμός της συχνότητας της περιόδου
 myNex.writeStr("home.tempfreq.txt", String(total));//αποστολή της συχνότητας στην μεταβλητή tempfreq που βρίσκεται στο page home
 myNex.writeStr("home.tempvolt.txt", String(Samp));//αποστολή της τάσης στην μεταβλητή tempfreq που βρίσκεται στο page home
 delay(10);
}
```

```
while (myNex.readNumber("home.va0.val") == 1)   {//sine 8bit page
 for (int t = 1; t<rep; t++) {
  sinewave8bit();
 }
}
/*while (myNex.readNumber("home.va0.val") == 11){//sineinfo 16bit page
 amplitude = myNex.readNumber("home.volt.val");//λήψη μεταβλητης πλάτους απο Nextion 0-4096
 cnt = myNex.readNumber("home.freq.val");//λήψη μεταβλητής συχνότητας απο Nextion 0-200
 duration = myNex.readNumber("home.duration.val");//λήψη μεταβλητής συχνότητας απο Nextion 1-10000
 rep = duration/0.0025974;//μετατροπή χρόνου σήματος σε επαναλήψεις
 a = amplitude/1.999;                                          ///
 b = amplitude/1.446;                                          ////
 c = amplitude/1.171;                                          /////
 d = amplitude/1.039;                                          //////
 e = amplitude;                                                //Δημιουργία σημείων ημιτονοειδούς σήματος
 f = amplitude/3.239;                                          //////
 g = amplitude/6.825;                                          /////
 h = amplitude/26.25;                                          ////
 i = amplitude*0;                                              ///
 float amp = (5*float(amplitude))/4096;//υπολογισμός τάσης εξόδου
 String Samp = String(amp,2);//αποθήκευση της μεταβλητής amp στην μεταβλητή Samp και μετατροπή σε string
 start = (micros());//έναρξη χρονομέτρησης της function sinewave
 sinewave16bit();//εκτέλεση της function sinewave
 end = (micros());//τέλος χρονομέτρησης της function sinewave
 total = (1/(end-start))*1000*1000;//υπολογισμός της συχνότητας της περιόδου
 myNex.writeStr("home.tempfreq.txt", String(total));//αποστολή της συχνότητας στην μεταβλητή tempfreq που
βρίσκεται στο page home
 myNex.writeStr("home.tempvolt.txt", String(Samp));//αποστολή της τάσης στην μεταβλητή tempfreq που
βρίσκεται στο page home
 delay(10);
}
while (myNex.readNumber("home.va0.val") == 1)   {//sine 16bit page
 for (int t = 1; t<rep; t++) {
  sinewave16bit();
 }
}*/
while (myNex.readNumber("home.va0.val") == 12)  {//triangleinfo page
 amplitude = myNex.readNumber("home.volt.val");//λήψη μεταβλητης πλάτους απο Nextion 0-4096
 cnt = myNex.readNumber("home.freq.val");//λήψη μεταβλητής συχνότητας απο Nextion 0-200
 duration = myNex.readNumber("home.duration.val");//λήψη μεταβλητής συχνότητας απο Nextion 1-10000
 rep = duration/0.0012936;                                     //μετατροπή χρόνου σήματος σε επαναλήψεις
 a = amplitude * 0;                                            ///
 b = amplitude * 0.25;                                         ////
 c = amplitude * 0.5;                                          //Δημιουργία σημείων τριγωνικού σήματος
 d = amplitude * 0.75;                                         ////
 e = amplitude;                                                ///
 float amp = (5*float(amplitude))/4096;//υπολογισμός τάσης εξόδου
 String Samp = String(amp,2);//αποθήκευση της μεταβλητής amp στην μεταβλητή Samp και μετατροπή σε string
 start = (micros());//έναρξη χρονομέτρησης της function triangle
 triangle();//εκτέλεση της function triangle
 end = (micros());//τέλος χρονομέτρησης της function triangle
 total = (1/(end-start)*1000*1000); //υπολογισμός της συχνότητας της περιόδου
 myNex.writeStr("home.tempfreq.txt", String(total));//αποστολή της συχνότητας στην μεταβλητή tempfreq που
βρίσκεται στο page home
 myNex.writeStr("home.tempvolt.txt", String(Samp));//αποστολή της τάσης στην μεταβλητή tempfreq που
βρίσκεται στο page home
 delay(10);
}
while (myNex.readNumber("home.va0.val") == 2)   {//triangle page
 for (int t = 1; t<rep; t++) {
  triangle();
 }
}
```

```
while (myNex.readNumber("home.va0.val") == 13)  {//pulseinfo page
 amplitude = myNex.readNumber("home.volt.val");//λήψη μεταβλητης πλάτους απο Nextion 0-4096
 cnt = myNex.readNumber("home.freq.val");//λήψη μεταβλητής συχνότητας απο Nextion 0-1000
 pulsewidthdown = myNex.readNumber("home.widthdown.val");//λήψη μεταβλητής διάρκειας αρνητικού παλμου απο
Nextion
 pulsewidthup = myNex.readNumber("home.widthup.val");//λήψη μεταβλητής διάρκειας θετικού παλμου απο
Nextion
 duration = myNex.readNumber("home.duration.val");//λήψη μεταβλητής συχνότητας απο Nextion 1-10000
 rep = duration/0.000346020761245;//μετατροπή χρόνου σήματος σε επαναλήψεις
 float amp = (5*float(amplitude))/4096;//υπολογισμός τάσης εξόδου
 String Samp = String(amp,2);//αποθήκευση της μεταβλητής amp στην μεταβλητή Samp και μετατροπή σε string
 start = (micros());//έναρξη χρονομέτρησης της function pulse
 pulse();//εκτέλεση της function pulse
 end = (micros());//τέλος χρονομέτρησης της function pulse
 total = (1/(end-start)*1000*1000);//υπολογισμός της συχνότητας της περιόδου
 myNex.writeStr("home.tempfreq.txt", String(total));//αποστολή της συχνότητας στην μεταβλητή tempfreq που
βρίσκεται στο page home
 myNex.writeStr("home.tempvolt.txt", String(Samp));//αποστολή της τάσης στην μεταβλητή tempfreq που
βρίσκεται στο page home
 delay(10);
}
while (myNex.readNumber("home.va0.val") == 3)   {//pulse page
 for (int t = 1; t<rep; t++) {
  pulse();
 }
}
while (myNex.readNumber("home.va0.val") == 14)  {//sawtoothinfo page
 amplitude = myNex.readNumber("home.volt.val");//λήψη μεταβλητης πλάτους απο Nextion 0-4096

 cnt = myNex.readNumber("home.freq.val");//λήψη μεταβλητής συχνότητας απο Nextion 0-1000
 duration = myNex.readNumber("home.duration.val");//λήψη μεταβλητής συχνότητας απο Nextion 1-10000
 rep = duration/0.0008163265306;//μετατροπή χρόνου σήματος σε επαναλήψεις
 float amp = (5*float(amplitude))/4096;//υπολογισμός τάσης εξόδου
 String Samp = String(amp,2);//αποθήκευση της μεταβλητής amp στην μεταβλητή Samp και μετατροπή σε string
 a = amplitude;                                          ///
 b = amplitude * 0.75;                                   ////
 c = amplitude * 0.5;                                    /////Δημιουργία σημείων πριονωτού σήματος
 d = amplitude * 0.25;                                   ////
 e = amplitude * 0;                                      ///
 start = (micros());//έναρξη χρονομέτρησης της function triangle
 sawtooth();//εκτέλεση της function sawtooth
 end = (micros());//τέλος χρονομέτρησης της function triangle
 total = (1/(end-start)*1000*1000);//υπολογισμός της συχνότητας της περιόδου
 myNex.writeStr("home.tempfreq.txt", String(total));//αποστολή της συχνότητας στην μεταβλητή tempfreq που
βρίσκεται στο page home
 myNex.writeStr("home.tempvolt.txt", String(Samp));//αποστολή της τάσης στην μεταβλητή tempfreq που
βρίσκεται στο page home
 delay(10);
}
while (myNex.readNumber("home.va0.val") == 4)   {//sawtooth page
 for (int t = 1; t<rep; t++) {
  sawtooth();
 }
}
while (myNex.readNumber("home.va0.val") == 5)   {//createnew page
 xy = myNex.readNumber("xy.val");//λήψη μεταβλητής xy (μήκος σήματος-σύνολο σημείων)
 if (myNex.readNumber("home.indicator.val") == 1 ){//έλεγχος του δείκτη indicator
  for (int i = 0; i <= xy-1; i++){//βρόγχος επανάληψης για το μήκος του σήματος-σύνολο σημείων
   xcoords[i] = myNex.readNumber("createnew.vtx" + String(i) + ".val");//λήψη και αποθηκευση των x
συντεταγμένων στον πίνακα xcoords
   ycoords[i] = myNex.readNumber("createnew.vty" + String(i) + ".val");//λήψη και αποθηκευση των y
συντεταγμένων στον πίνακα ycoords
```

```
   mapedvoltage[i] = map(ycoords[i], 11, 399, 4095, 0);//mapping απο το πεδιο 10-400(pixels) στο πεδιο
4095-0 (έξοδο)
  }
 }
 delays[0] = xcoords[0] - 7;//υπολογισμός καθυστέρησης απο την αρχή του άξονα x έως το πρώτο σημείο
 delays[1] = xcoords[1] - xcoords[0];                          //
 delays[2] = xcoords[2] - xcoords[1];                          ///
 delays[3] = xcoords[3] - xcoords[2];                          ////
 delays[4] = xcoords[4] - xcoords[3];                          /////
 delays[5] = xcoords[5] - xcoords[4];                          //////
 delays[6] = xcoords[6] - xcoords[5];                          ///////
 delays[7] = xcoords[7] - xcoords[6];                          ////////
 delays[8] = xcoords[8] - xcoords[7];                          /////////
 delays[9] = xcoords[9] - xcoords[8];                          /////////
 delays[10] = xcoords[10] - xcoords[9];                        //////////υπολογισμός καθυστερήσεων μεταξύ
σημείων
 delays[11] = xcoords[11] - xcoords[10];                       /////////
 delays[12] = xcoords[12] - xcoords[11];                       ////////
 delays[13] = xcoords[13] - xcoords[12];                       ////////
 delays[14] = xcoords[14] - xcoords[13];                       ///////
 delays[15] = xcoords[15] - xcoords[14];                       //////
 delays[16] = xcoords[16] - xcoords[15];                       /////
 delays[17] = xcoords[17] - xcoords[16];                       ////
 delays[18] = xcoords[18] - xcoords[17];                       ///
 delays[19] = xcoords[19] - xcoords[18];                       //
 delay(2000);
}
while (myNex.readNumber("home.va0.val") == 6)   {//wavepreview page
 rep = 1000;
 cnt = myNex.readNumber("home.freq.val");//λήψη μεταβλητής συχνότητας απο Nextion 0-1000
 start = (micros());//έναρξη χρονομέτρησης της function customwave
 customwave();//εκτέλεση της function customwave
 end = (micros());//τέλος χρονομέτρησης της function customwave
 total = (1/(end-start)*1000*1000);//υπολογισμός της συχνότητας της περιόδου
 myNex.writeStr("home.tempfreq.txt", String(total));//αποστολή της τάσης στην μεταβλητή tempfreq που
βρίσκεται στο page home
}
while (myNex.readNumber("home.va0.val") == 7)   {//customwave page
 for (int t = 1; t<rep; t++) {
  customwave();
 }
}
while (myNex.readNumber("home.va0.val") == 8)   {//eeprom page
 //=================check for available signals======//έλεγχος αν υπάρχουν αποθηκευμένα σήματα
 if (EEPROM.read(100) == 255) {//έλεγχος αν η θέση μνήμης 100 ειναι 0xFF(διαθέσιμη)
  myNex.writeStr("signalaval.txt", String(0));//αποστολή στο Nextion την τιμή 0 και αποθήκευση στην
μεταβλητή ελέγχου σήματος signalaval
 }
 delay(10);
 if (EEPROM.read(500) == 255) {//έλεγχος αν η θέση μνήμης 500 ειναι 0xFF(διαθέσιμη)
  myNex.writeStr("signalbval.txt", String(0));//αποστολή στο Nextion την τιμή 0 και αποθήκευση στην
μεταβλητή ελέγχου σήματος signalbval
 }
 delay(10);
 //=================delete signal====================
 if ((myNex.readNumber("eeprom.dela.val") == 1) or (myNex.readNumber("eeprom.delb.val") == 1)){ //έλεγχος
αν κάποια απο τις μεταβλητές διαγραφής είναι ενεργοποιημένη
  if (myNex.readNumber("eeprom.dela.val") == 1) {//έλεγχος της μεταβλητής διαγραφής του σήματος Α αν
είναι ενεργοποιημένη
   for (int i=100;i<499;i++){//βρόγχος επανάληψης για τις θέσεις μνήμης της EEPROM 100-499
    EEPROM.write(i, 255);//διαγραφή θέσης μνήμης - 0xff
   }
   Serial.print("\xff\xff\xff");
```

```
   Serial.print("eeprom.dela.val=0");////μηδενισμό της μεταβλητής διαγραφής του σήματος A
   Serial.print(endChar);
   myNex.writeStr("signalaval.txt", String(0));//μηδενισμό της μεταβλητής διαθεσιμότητας του σήματος A
   delay(10);
  }
  if (myNex.readNumber("eeprom.delb.val") == 1) {//έλεγχος της μεταβλητής διαγραφής του σήματος B αν
είναι ενεργοποιημένη
   for (int i=500;i<900;i++){//έλεγχος της μεταβλητής διαγραφής του σήματος A αν είναι ενεργοποιημένη
    EEPROM.write(i, 255);//διαγραφή θέσης μνήμης - 0xff
   }
   Serial.print("\xff\xff\xff");///
   Serial.print("eeprom.delb.val=0");////μηδενισμό της μεταβλητής διαγραφής του σήματος B
   Serial.print(endChar);
   myNex.writeStr("signalbval.txt", String(0));//μηδενισμό της μεταβλητής διαθεσιμότητας του σήματος B
   delay(10);
  }
 }
 //=================load signal=====================
 if ((myNex.readNumber("eeprom.lda.val") == 1) or (myNex.readNumber("eeprom.ldb.val") == 1)){//έλεγχος αν
κάποια απο τις μεταβλητές load είναι ενεργοποιημένη
  if (myNex.readNumber("eeprom.lda.val") == 1)  {//έλεγχος της μεταβλητής load του σήματος A αν είναι
ενεργοποιημένη
   xcoords[20] = {0};//initialization του πίνακα xcoords
   ycoords[20] = {0};//initialization του πίνακα ycoords
   xy = 0;//initialization της μεταβλητης xy
   EEPROM.get(100, xcoords);//λήψη των δεδομένων απο την θέση 100 της EEPROM και αποθήκευση στον πίνακα
xcoords
   EEPROM.get(200, ycoords);//λήψη των δεδομένων απο την θέση 200 της EEPROM και αποθήκευση στον πίνακα
ycoords
   EEPROM.get(300, mapedvoltage);//λήψη των δεδομένων απο την θέση 300 της EEPROM και αποθήκευση στον
πίνακα mapedvoltage
   EEPROM.get(400, delays);//λήψη των δεδομένων απο την θέση 400 της EEPROM και αποθήκευση στον πίνακα
delays
   EEPROM.get(450, total);//λήψη των δεδομένων απο την θέση 450 της EEPROM και αποθήκευση στην μεταβλητή
total
   EEPROM.get(460, xy);//λήψη των δεδομένων απο την θέση 460 της EEPROM και αποθήκευση στην μεταβλητή xy
   Serial.print("\xff\xff\xff");
   Serial.print("eeprom.lda.val=0");//μηδενισμό της μεταβλητης load του σήματος A
   Serial.print(endChar);
   Serial.print("\xff\xff\xff");
   Serial.print("page customload");//page customload
   Serial.print(endChar);
   for (i=0;i<=xy-1;i++){//βρόγχος επανάληψης όσο το μήκος του σήματος-σύνολο σημείων
    if (xcoords[i]>0 && ycoords[i]>0) {//έλεγχος αν οι τιμές ειναι πάνω απο 0
     myNex.writeStr("fill " + String(xcoords[i]) + "," + String(ycoords[i]) + ",6,6,63488");//εντολή
τοποθέτησης και χρωματισμού σημείων σήματος στο waveform
    }
   }
   myNex.writeStr("t0.txt", String("Producing Signal A"));//αποστολή του string Producing Signal A στο
αντικείμενο t0.txt
   myNex.writeStr("home.tempfreq.txt", String(total));//αποστολη συχνότητας στην μεταβλητή tempfreq
   delay(10);
  }
  if (myNex.readNumber("eeprom.ldb.val") == 1)  {//έλεγχος της μεταβλητής load του σήματος B αν είναι
ενεργοποιημένη
   xcoords[20] = {0};//initialization του πίνακα xcoords
   ycoords[20] = {0};//initialization του πίνακα ycoords
   xy = 0;//initialization της μεταβλητης xy
   EEPROM.get(500, xcoords);//λήψη των δεδομένων απο την θέση 500 της EEPROM και αποθήκευση στον πίνακα
xcoords
   EEPROM.get(600, ycoords);//λήψη των δεδομένων απο την θέση 600 της EEPROM και αποθήκευση στον πίνακα
ycoords
```

```
   EEPROM.get(700, mapedvoltage);//λήψη των δεδομένων απο την θέση 700 της EEPROM και αποθήκευση στον
πίνακα mapedvoltage
   EEPROM.get(800, delays);//λήψη των δεδομένων απο την θέση 800 της EEPROM και αποθήκευση στον πίνακα
mapedvoltage
   EEPROM.get(850, total);//λήψη των δεδομένων απο την θέση 850 της EEPROM και αποθήκευση στην μεταβλητή
total
   EEPROM.get(860, xy);//λήψη των δεδομένων απο την θέση 860 της EEPROM και αποθήκευση στην μεταβλητή xy
   Serial.print("\xff\xff\xff");
   Serial.print("eeprom.ldb.val=0");//μηδενισμό της μεταβλητής load του σήματος B
   Serial.print(endChar);
   Serial.print("\xff\xff\xff");
   Serial.print("page customload");//page customload
   Serial.print(endChar);
   for (i=0;i<=xy-1;i++){//βρόγχος επανάληψης όσο το μήκος του σήματος-σύνολο σημείων
    if (xcoords[i]>0 && ycoords[i]>0) {//έλεγχος αν οι τιμές ειναι πάνω απο 0
     myNex.writeStr("fill " + String(xcoords[i]) + "," + String(ycoords[i]) + ",6,6,63488"); //εντολή
τοποθέτησης και χρωματισμού σημείων σήματος στο waveform
    }
   }
   myNex.writeStr("t0.txt", String("Producing Signal B"));//αποστολή του string Producing Signal B στο
αντικείμενο t0.txt
   myNex.writeStr("home.tempfreq.txt", String(total));//αποστολη συχνότητας στην μεταβλητή tempfreq
   delay(10);
  }
 }
}
while (myNex.readNumber("home.va0.val") == 9)   {//save page
 if (EEPROM.read(100) == 255) {//έλεγχος αν η θέση μνήμης 100 ειναι 0xFF(διαθέσιμη)
  EEPROM.put(100, xcoords);//αποθήκευση στην διεύθυνση 100 της EEPROM τον πίνακα xcoords
  EEPROM.put(200, ycoords);//αποθήκευση στην διεύθυνση 200 της EEPROM τον πίνακα ycoords
  EEPROM.put(300, mapedvoltage);//αποθήκευση στην διεύθυνση 300 της EEPROM τον πίνακα mapedvoltage
  EEPROM.put(400, delays);//αποθήκευση στην διεύθυνση 400 της EEPROM τον πίνακα delays
  EEPROM.put(450, total);//αποθήκευση στην διεύθυνση 450 της EEPROM την μεταβλητή total
  EEPROM.put(460, xy);//αποθήκευση στην διεύθυνση 460 της EEPROM την μεταβλητή xy
  myNex.writeStr("t0.txt", String("Signal saved as Signal A"));//αποστολή το string στο αντικείμενο
t0.txt
  delay(3000);
  Serial.print("\xff\xff\xff");
  Serial.print("page home");//page home
  Serial.print(endChar);
  delay(1000);
 }
 else if (EEPROM.read(500) == 255) {
  EEPROM.put(500, xcoords);//αποθήκευση στην διεύθυνση 500 της EEPROM τον πίνακα xcoords
  EEPROM.put(600, ycoords);//αποθήκευση στην διεύθυνση 600 της EEPROM τον πίνακα ycoords
  EEPROM.put(700, mapedvoltage);//αποθήκευση στην διεύθυνση 700 της EEPROM τον πίνακα mapedvoltage
  EEPROM.put(800, delays);//αποθήκευση στην διεύθυνση 800 της EEPROM τον πίνακα delays
  EEPROM.put(850, total);//αποθήκευση στην διεύθυνση 850 της EEPROM την μεταβλητή total
  EEPROM.put(860, xy);//αποθήκευση στην διεύθυνση 860 της EEPROM την μεταβλητή xy
  myNex.writeStr("t0.txt", String("Signal saved as Signal B")); //αποστολή το string στο αντικείμενο
t0.txt
  delay(3000);
  Serial.print("\xff\xff\xff");
  Serial.print("page home");//page home
  Serial.print(endChar);
  delay(1000);
 }
 else {
  myNex.writeStr("t0.txt", String("Memory full"));//αποστολη το String στο αντικείμενο t0.txt
  delay(3000);
  Serial.print("\xff\xff\xff");
  Serial.print("page home");//page home
  Serial.print(endChar);
```

```
 }
}
while (myNex.readNumber("home.va0.val") == 10)  {//customload
 for (int t = 1; t<rep; t++) {
  customwave();
 }
}
}
```

Nextion Event Handlers

Change page event handler:
home.va0.val=X //αλλαγή της βασικής μεταβλητής
page "όνομα σελίδας"//εντολή αλλαγής σελίδας

Change amplitude event handler: //αλλαγή του πλάτους της κυματομορφής

minus button:

```
if(home.volt.val<50)
{
   home.volt.val=0
}else
{
   home.volt.val=home.volt.val-50
}
```

plus button:

```
if(home.volt.val>4040)
{
   home.volt.val=4095
}else
{
   home.volt.val=home.volt.val+50
}
```

Change frequency event handler: //αλλαγή της συχνότητας της κυματομορφής

minus button:

```
if(home.freq.val>999)
{
  home.freq.val=1000
}else
{
  home.freq.val=home.freq.val+1
}
```

plus button:

```
if(home.freq.val<1)
{
 home.freq.val=0
}else
{
 home.freq.val=home.freq.val-1
}
```

Change duration event handler: //αλλαγή της διάρκειας της κυματομορφής

minus button:

```
if(home.duration.val<2)
{
 home.duration.val=1
}else
{
 home.duration.val=home.duration.val-1
}
```

plus button:

```
if(home.duration.val>9999)
{
 home.duration.val=10000
}else
{
 home.duration.val=home.duration.val+1
}
```

Add custom wave point event handler: //προσθήκη σημείων κυματομορφής

```
   if(x0.val<10)    // αν το x0 είναι εκτός πεδίου της κυματομορφής θα πάρει την ελάχιστη τιμή του πεδίου
   {
    x0.val=7
    y0.val=397
   }
   covx x0.val,tx0.txt,0,0  //μετατροπή της μεταβλητής x0 σε string και αποθήκευση στην tx0
   cov tx0.txt,vtx0.val,0  //μετατροπή του tx0 απο string σε numeric στην μεταβλητή vtx0
   covx y0.val,ty0.txt,0,0 //μετατροπή του ty0 απο string σε numeric στην μεταβλητή vty0
   cov tx0.txt,vtx0.val,0  //μετατροπή του string tx0 σε numeric και αποθήκευση στην μεταβλητή vtx0
   cov ty0.txt,vty0.val,0  //μετατροπή του string ty0 σε numeric και αποθήκευση στην μεταβλητή vty0
   tsw b0,0                //απενεργοποίηση button b0
   b0.pco=40147            //αλλαγή χρωματισμού button b0
```

```
   tsw b1,1                  //ενεργοποίηση επόμενου button b1
   b1.pco=0                  //αλλαγή χρωματισμού button b1
   fill x0.val,y0.val,6,6,63488   //δημιουργία απεικόνισης (κόκκινη τελεία) x0 και y0, 6x6pixel
63488(χρώμα hex)
   tsw b21,0                 //απενεργοποίηση button b21
   b21.pco=40147             //αλλαγή χρωματισμού button22
   tsw b22,0                 //απενεργοποίηση button b22
   b22.pco=40147             //αλλαγή χρωματισμού button22
```

Create custom wave total points event handler: //δημιουργία ποσότητας σημείων της custom κυματομορφής

```
minus button:
if(xy.val<3)
{
  xy.val=2
}else
{
  xy.val=xy.val-1
}
plus button:
if(xy.val>19)
{
  xy.val=20
}else
{
  xy.val=xy.val+1
}
```