ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

<<Σχεδίαση και κατασκευή πλατφόρμας ανάπτυξης Arduino Uno και εφαρμογή της σε σχεδίαση μικρού ψηφιακού παλμογράφου>>

Του φοιτητή
Χρήστος Ντουλαβέρης
Αρ. Μητρώου: 514103

Επιβλέπων
Ονοματεπώνυμο Άγγελος Γιακουμής
Βαθμίδα Λέκτορας

Ημερομηνία ……

Τίτλος Δ.Ε. Σχεδίαση και κατασκευή πλατφόρμας ανάπτυξης Arduino Uno και εφαρμογή της σε σχεδίαση μικρού ψηφιακού παλμογράφου
Κωδικός Δ.Ε. 13256
Ονοματεπώνυμο φοιτητή/τών Χρήστος Ντουλαβέρης
Ονοματεπώνυμο εισηγητή Άγγελος Γιακουμής
Ημερομηνία ανάληψης Δ.Ε. …
Ημερομηνία περάτωσης Δ.Ε. …

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Πρόλογος

Διάλεξα την συγκεκριμένη πτυχιακή εργασία με τίτλο **Σχεδίαση και κατασκευή πλατφόρμας ανάπτυξης Arduino Uno και εφαρμογή της σε σχεδίαση μικρού ψηφιακού παλμογράφου** επειδή μου άρεσε η απλότητα που κατέχει το Arduino.Από καιρό ήθελα να συνδυάσω τον προγραμματισμό με τα ηλεκτρονίκα και το αποτέλεσμα να βγει κάτι πρακτικό που θα μπορεί να κάνει μέχρι και τον πιο αρχάριο να μπορέσει να πειραματιστεί πάνω σε αυτό.Η απλότητα που κατέχει το Arduino τόσο στον προγραμματισμό όσο στις θύρες του είναι κάτι το φανταστικό.Έχει άπειρες δυνατότητες και για προσωπική χρήση αλλά και για επαγγελματική.Είναι μία ιδέα με πολυ χαμηλό κόστος, επίσης είναι ευκολόχρηστο και αυτά τα δύο σε συνδυασμό το κάνουν μοναδικό.Επίσης ένας άλλος λόγος που επέλεξα το Arduino πέρα από την απλότητα του και τις μεγάλες δυνατότητες του είναι ότι είναι μία μικρή βάση σε όλα τα ηλεκτρονικά συστήματα.Από ρομποτική έως και έλεγχο συστημάτων ασφαλειάς όπως συναγερμοί.Είναι ας το πούμε ένα μικρό εργαλείο με μεγάλες δυνατότητες που χρησιμοποιούνε οι εκάστοτε εταιρείες.

Περίληψη

Η πτυχιακή εργασία σχετίζεται με μιά αξιόπιστη λύση του παλμογράφου και της γεννήτριας συχνοτήτων σε μία πιο απλοική μορφή από τους επαγγελματικούς, επαγγελματικές παπλμογράφους και γεννήτριες συχνοτήτων αντίστοιχα.Στο κεφάλαιο 1^{ο} γίνεται μια μικρή γνωριμία με το Arduino όπου αναφέρονται τα είδη του που υπάρχουν στο εμπόριο.Τα εργαλεία που χρειάζεται κάποιος για να μπορέσει να ασχοληθεί πάνω σε αυτό.Στο κεφάλαιο 2^{ο} αναφέρονται πιο συγκεκριμένα τα στοιχεία που περιέχει πάνω του από τα ποδαράκια του και την τροφοδοσία έως πιο συγκεκριμένα τον μικροελεγκτή που περιέχει.Αναφέρεται η αρχιτεκτονική του μικροελεγκτή και η διαφορά του από άλλες αρχιτεκτονικές.Στη συνέχεια στο κεφάλαιο 3^{ο} προσκομίζεται το πρόγραμμα του Arduino και γίνεται αναφόρα στην χρήση του και στα εργαλεία που περιέχει.Επίσης αναφέρω τις εντολές και τα στοιχεία της γλώσσας C++ και την εξήγηση του σε κάθε μία περίπτωση και πως αυτά συνδέονται με τον κώδικα του Arduino.Στο κεφάλαιο 4^{ο} γίνεται μία απλή αναφορά στους παπλμογράγους και στις γεννήτριες συχνοτήτων.Στη συνέχεια στο κεφάλαιο 5^{ο} περιέχεται ο κώδικας του προγράμματος και μάζι του το διάγραμμα ροής του έτσι ώστε να γίνει πιο κατανοητός από τον αναγνώστη.Η κατασκευή που βρίσκεται σε αυτό το κεφάλαιο αναφέρει το κόστος της και σας δείχνω βήμα βήμα την αρχή την μέση και το τέλος της.Το συμπέρασμα του έλαβα μετά το τέλος της πτυχιακής μου εργασίας είναι ότι είναι χρήσιμη και αξιόπιστη σε διάφορα κυκλώματα ή και σε μη όσο αναφορά την γεννήτρια συχνοτήτων.Είναι μία κινητή συσκευή πολύ μικρή στο μέγεθος της με τεράστιες δυνατότητες και σε μικρό κόστος όπου διευκολύνη την εργασία ενός ηλεκτρονικού.

<<Design and construction of Arduino Uno development platform and its application in small digital oscilloscope design>>

<<Chris Ntoulaveris>>

Abstract

The dissertation is related to a reliable solution of the oscilloscope and the frequency generator in a simpler form than the professional, professional pulsographs and frequency generators respectively. The tools one needs to be able to deal with it. Chapter 2 lists more specifically the elements it contains above its legs and the power supply to more specifically the microcontroller it contains. The architecture of the microcontroller and its difference from other architectures are mentioned. Then in chapter 3 the Arduino program is presented. And it refers to its use and the tools it contains. I also mention the commands and elements of the C ++ language and its explanation in each case and how they are related to the Arduino code. Chapter 4 makes a simple reference to the pulse monitors and frequency generators. Then Chapter 5 contains the program code and its flow chart so that it can be more understood by the reader. The construction in this chapter states its cost and I show you step by step the beginning, the middle and the end of it. It is a mobile device very small in size with huge capabilities and at a low cost where it facilitates the work of an electronic.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

ΚΕΦΑΛΑΙΟ 1^{Ο} : Γνωριμία με το Arduino

1.1 Εισαγωγή

Σε αυτό το κεφάλαιο γίνεται μια απλή αναφορά στο Arduino στα είδη του, στα πλεονεκτήματα του.Το πως και από που γεννήθηκε η ίδεα του, πιοι ήταν οι εφευρέτες του.Μια μικρή περιήγηση δηλαδή στον κόσμο του Arduino με τις υπέροχες δυνατότητες του.

1.2 Ιστορία του Arduino

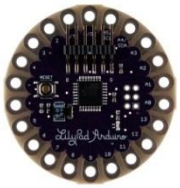
Το 2005, στην Ivrea της Ιταλία (η ιστοσελίδα της εταιρείας υπολογιστών Olivetti), ένα έργο άρχισε να δημιουργείτε, μια συσκευή για τον έλεγχο σχεδίων, χτισμένο από μαθητές με λιγότερα έξοδα από ό,τι με άλλα πρωτότυπα συστήματα που ήταν διαθέσιμα εκείνην τη στιγμή. Από τον Μάιο του 2011, περισσότερα από 300.000 Arduino ήταν «στην άγρια φύση». Εφευρέτες Massimo Banzi και David Cuartielles, ονόμασαν το έργο τους Arduin of Ivrea. Το «Arduino» είναι επίσης ένα ιταλικό όνομα, που σημαίνει «γενναίος φίλος».
Το έργο Arduino είναι μια παραγόμενη έκδοση της πλατφόρμας ανοικτού κώδικα Wiring Platform. Ο Κολομβιανός καλλιτέχνης και προγραμματιστής Hernando Barragan δημιούργησε συρμάτωση «Wiring» ως μια Μεταπτυχιακή διπλωματική εργασία στο Interaction Design Institute Ivrea υπό την εποπτεία του Massimo Banzi και του Casey Reas. Η Καλωδίωση «Wiring» βασίστηκε στην επεξεργασία και το ολοκληρωμένο περιβάλλον ανάπτυξης που είχε δημιουργηθεί από τον Casey Reas και τον Ben Fry.
Το Arduino χτίστηκε γύρω από το έργο της καλωδίωσης «Wiring» του Hernando Barragan. Η Καλωδίωση ήταν η διπλωματική εργασία του Hernando στο Interaction Design Institute Ivrea. Επρόκειτο να είναι μια ηλεκτρονική έκδοση της επεξεργασίας που χρησιμοποιήθηκε σε προγραμματιστικό περιβάλλον και ήταν η βάση για την σύνταξη επεξεργασίας. Αυτό ήταν υπό την εποπτεία του Hernado και του Massimo Banzi, ιδρυτές ενός Arduino. Ήταν κομμάτι της φαντασίας ότι θα υπήρχαν Arduino χωρίς καλωδιώσεις και ότι δεν θα υπάρχει καλωδίωση χωρίς επεξεργασία. Η επεξεργασία σίγουρα δεν θα υπήρχε χωρίς τη γλώσσα προγραμματισμού Design By Numbers και τον John Maeda.

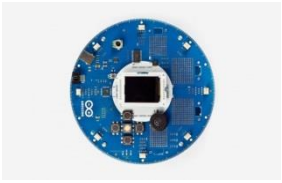
1.3 Γενικά στοιχεία

Το Arduino θα λέγαμε ότι είναι ένα εργαλείο για να κατασκευάσουμε ένα υπολογιστικό σύστημα με την έννοια ότι αυτό θα ελέγχει συσκευές του φυσικού κόσμου, σε αντίθεση με τον κοινό Ηλεκτρονικό Υπολογιστή. Είναι ανοιχτού υλικού και λογισμικού και βασίζεται σε μια αναπτυξιακή πλακέτα που ενσωματώνει επάνω έναν μικροελεγκτή και συνδέεται με τον Η/Υ για να τον προγραμματίσουμε μέσα από ένα απλό περιβάλλον ανάπτυξης. Το Arduino,μπορεί να χρησιμοποιηθεί για να αναπτύξουμε διαδραστικά αντικείμενα, να δεχτούμε εισόδους από πληθώρα αισθητηρίων οργάνων και διακόπτες, αλλά και να ελέγχουμε διάφορα φώτα, κινητήρες και άλλες συσκευές εξόδου. Οι πλακέτες μπορούν εύκολα να συναρμολογηθούν ακόμη και από έναν αρχάριο ή να αγοραστούν μονταρισμένες. Το περιβάλλον ανάπτυξης του λογισμικού βασίζεται στην γλώσσα προγραμματισμού Processing και την γλώσσα προγραμματισμού Wiring, οι οποίες είναι ανοιχτού κώδικα (open source).

1.4 Μοντέλα του Arduino

Τα μοντέλα Arduino που κυκλοφορούν στην αγορά, παρουσιάζονται στον πίνακα 1.4. Στην πτυχιακή εργασία, επιλέχθηκε το μοντέλο Arduino NANO μιας και καλύπτει όλες τις ανάγκες της εργασίας καθώς τα pins που διαθέτει επαρκούν για όλες τις λειτουργίες.

Όνομα	Σχήμα	Επεξεργαστής	Χαρακτηριστικά
Arduino UNO		ATmega328 (CPU 8 bit, ταχύτητα ρολογιού 16 MHz, 2 KB SRAM, 32 KB αποθήκευση flash)	14 ψηφιακές ακίδες I/O, 6 αναλογικές ακίδες εισόδου, αφαιρούμενος μικροελεγκτής.
Arduino NANO		ATmega328 (CPU 8 bit, ταχύτητα ρολογιού 16 MHz, 2 KB SRAM, 32 KB αποθήκευση flash)	14 ψηφιακές ακίδες, 8 αναλογικές καρφίτσες, 2 ακροδέκτες επαναφοράς και 6 καρφίτσες ισχύος.
Arduino LILYPAD		ATmega328 (CPU 8 bit, ταχύτητα ρολογιού 16 MHz, 2 KB SRAM, 32 KB αποθήκευση flash)	14 ψηφιακές ακίδες I/O, 6 ακίδες αναλογικής εισόδου.
Arduino LEONARDO		ATmega32u4 (CPU 8 bit, ταχύτητα ρολογιού 16 MHz, 2,5 KB SRAM, 32 KB αποθήκευση flash)	20 ψηφιακές ακίδες I/O, 12 από τις οποίες μπορούν να χρησιμοποιηθούν ως αναλογικές είσοδοι, υποστήριξη εγγενούς USB.
Arduino MICRO		ATmega32u4 (CPU 8 bit, ταχύτητα ρολογιού 16 MHz, 2,5 KB SRAM, 32 KB αποθήκευση flash)	20 ψηφιακές ακίδες I/O, 12 από τις οποίες μπορούν να χρησιμοποιηθούν ως αναλογικές είσοδοι, υποστήριξη εγγενούς USB.

Arduino YUN		ATmega32u4 (CPU 8-bit, ταχύτητα ρολογιού 16 MHz, 2,5 KB SRAM, 32 KB αποθήκευση flash), σύστημα Atheros AR9331 σε τσιπ	Σύστημα βασισμένο σε Linux με δυνατότητα Wi-Fi σε τσιπ, 14 ψηφιακές αναλογικές ακίδες εισόδου/εξόδου, 12 από τις οποίες μπορούν να χρησιμοποιηθούν ως αναλογικές είσοδοι. Εγγενές USB.
Arduino ESPLORA		ATmega32u4 (CPU 8 bit, ταχύτητα ρολογιού 16 MHz, 2,5 KB SRAM, 32 KB αποθήκευση flash)	Πολύ ενσωματωμένο υλικό εισόδου και εξόδου.
Arduino ROBOT		2 x ATmega32u4 (CPU 8 bit, ταχύτητα ρολογιού 16 MHz, 2,5 KB SRAM, 32 KB αποθήκευση flash)	Τροχοί, 8 ακίδες αναλογικής εισόδου, 6 ψηφιακές ακίδες I/O, οθόνη LCD.

Πίνακας 1.4: Τα είδη του Arduino

1.5 Λογισμικό του Arduino

Το ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) του Arduino είναι μία εφαρμογή γραμμένη σε **Java**, που λειτουργεί σε πολλές πλατφόρμες και προέρχεται από το IDE για τη γλώσσα προγραμματισμού Processing και το σχέδιο Wiring. Έχει σχεδιαστεί για να εισαγάγει στον προγραμματισμό τους καλλιτέχνες και τους νέους που δεν είναι εξοικειωμένοι με την ανάπτυξη λογισμικού. Περιλαμβάνει ένα πρόγραμμα επεξεργασίας κώδικα με χαρακτηριστικά όπως είναι η επισήμανση σύνταξης και ο συνδυασμός αγκύλων και είναι επίσης σε θέση να μεταγλωττίζει και να φορτώνει προγράμματα στην πλακέτα με ένα μόνο κλικ. Δεν υπάρχει συνήθως καμία ανάγκη να επεξεργαστείτε αρχεία make ή να τρέξετε προγράμματα σε ένα περιβάλλον γραμμής εντολών. Ένα πρόγραμμα ή κώδικας που γράφτηκε για Arduino ονομάζεται *σκίτσο* (sketch).

Τα Arduino προγράμματα είναι γραμμένα σε **C** ή **C++**. **Το Arduino IDE** έρχεται με μια βιβλιοθήκη λογισμικού που ονομάζεται "**Wiring**", από το πρωτότυπο σχέδιο Wiring, γεγονός που καθιστά πολλές κοινές λειτουργίες εισόδου/εξόδου πολύ πιο εύκολες.

1.6 Πλεονεκτήματα του Arduino

1) **Το Χαμηλό κόστος**. Το κόστος μιας πλακέτας ανέρχεται σε μερικά ευρώ. Πιο συγκεκριμένα το Arduino uno που είναι η ναυαρχίδα της Arduino μπορεί να αγοραστεί μεσώ

ίντερνετ για 10-15€. Επίσης υπάρχουν πακέτα ξεκινήματος τα οποία ξεκινούν από 30 € και τα οποία εκτός από την πλακέτα προμηθεύουν μια γκάμα από ηλεκτρονικά (καλώδια,αισθητήρες,ηλεκτροκινητήρες,led …) για τις πρώτες δόκιμες-κατασκευές.

2) **Είναι Ανεξαρτήτου πλατφόρμας (cross-platform)**. Το πρόγραμμα του arduino εκτελείται και στα τρία λειτουργικά συστήματα (windows,macintosh,linux) αγκαλιάζοντας όλο το εύρος των χρηστών προσωπικών υπολογιστών

3) **Η Απλότητα.** Ίσως το ποιο σημαντικό πλεονέκτημα του arduino είναι η απλότητα του. Μέσα σε λίγες ώρες ο άπειρος χρήστης μπορεί να δημιουργήσει την πρώτη του κατασκευή. Αποτελεί ιδανικό δημιουργικό εργαλείο για την απόκτηση ηλεκτρονικών και μηχανικών δεξιοτήτων καθώς επίσης και δημιουργική απασχόληση για τους εφήβους.

4) **Οι εκδόσεις του Arduino**. Η arduino είναι μια οικογένεια από πλακέτες μικροελεγχτών που σκοπό έχουν να κάνουν ευκολότερη την κατασκευή διαδραστικών αντικείμενων. Κάθε έκδοση καλύπτει διαφορετικές ανάγκες και έχει διαφορετικές δυνατότητες. Έκτος από τις διαφορετικές εκδόσεις υπάρχουν και οι πλακέτες επέκτασης (shield's) που έρχονται να ενισχύσουν και να δώσουν νέες δυνατότητες στις πλατφόρμες του arduino. Όπως γίνεται κατανοητό πάντα θα υπάρχουν διάφορες εκδόσεις του arduino που θα καλύπτουν την τρέχουσα τεχνολογία.

5) **Η οικογενεια του Arduino**. Ένα ακόμα από τα σημαντικότερα του πλεονεκτήματα είναι το πλήθος των ανθρώπων που ασχολούνται με κάθε τομέα του arduino (υλικό και λογισμικό). Έτσι υπάρχουν αμέτρητα forum και ιστοσελίδες που μπορούν να καθοδηγήσουν, βοηθήσουν,διδάξουν και εμπνεύσουν των κάθε χρήστη, έμπειρο ή όχι.

6) **Το Ανοικτού κώδικα και επέκτασης λογισμικό**. Ο καθένας μπορεί να βρει τον πηγαίο κωδικά, να τον μελετήσει και να τον τροποποιήσει σύμφωνα με τις ανάγκες του. Έμπειροι χρήστες μπορούν μέσα από τις βιβλιοθήκες τις C++ άλλα και μέσα από τις βιβλιοθήκες του arduino να γράψουν τον δικό τους κωδικά και να τον μοιραστούν. Ενώ αρχάριοι χρήστες μπορούν να χρησιμοποιήσουν αυτόν το κωδικά χωρίς να γνωρίζουν προγραμματισμό χαμηλού επίπεδου.

7) **Το Ανοικτού κώδικα και επέκτασης υλικό**. Τα σχεδία των πλατφορμών είναι ανοικτού κωδικά πράγμα που σημαίνει ότι έμπειροι χρήστες στην ηλεκτρονική μπορούν να επεκτείνουν και να αναβαθμίσουν τις πλατφόρμες. Ακόμα και άπειροι χρήστες μπορούν να κατασκευάσουν την breadboard εκδοσή εάν θέλουν να καταλάβουν πως λειτουργεί και να γλιτώσουν και λίγα χρήματα.

1.7 Επίλογος

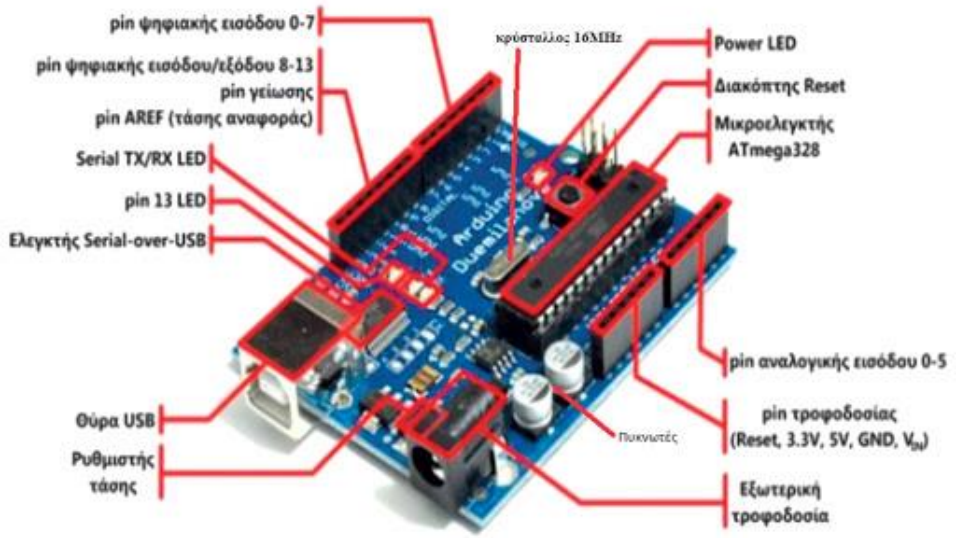
Όπως αναφέρεται και παραπάνω υπάρχει μια μεγάλη ποικιλία από Arduino στο εμπόριο με το πιο διαδεδομένο να είναι το UNO.Το πιο σημαντικό μειονέκτημα του είναι ότι έχει περιορισμένη μνήμη που μπορεί να κάνει περίπλοκα κυκλώματα δύσκολα.Αλλά πέρα από αυτό θα διευκολύνει και θα εμπνεύσει έναν ηλεκτρονικό να ασχοληθεί με αυτό κυρίως για την εύκολη προσβασιμότητα του και τις αμέτρητες δυνατότητες του Arduino.

ΚΕΦΑΛΑΙΟ 2ο: Τα χακτηριστικά του Arduino UNO

2.1 Εισαγωγή

Σε αυτό το κεφάλαιο θα μελετηθούν τα επιμέρους τμήματα που έχει επάνω το Arduino.Από το πως τροφοδοτείται, τις θύρες του και θα δώσουμε βάση στον εγκέφαλο του Arduino που δεν είναι άλλος από τον μικροελεγκτή.Θα κάνουμε πιο συγκεκριμένα μία επισκόπηση στις αρχιτεκτονικές που υπάρχουν όσο αναφορά τους μικροελεγκτές και θα δούμε πως λειτουργούν τα ολοκληρωμένα που έχει πάνω του.

2.2 Τροφοδοσία



Σχήμα 2.2: Τροφοδοσία

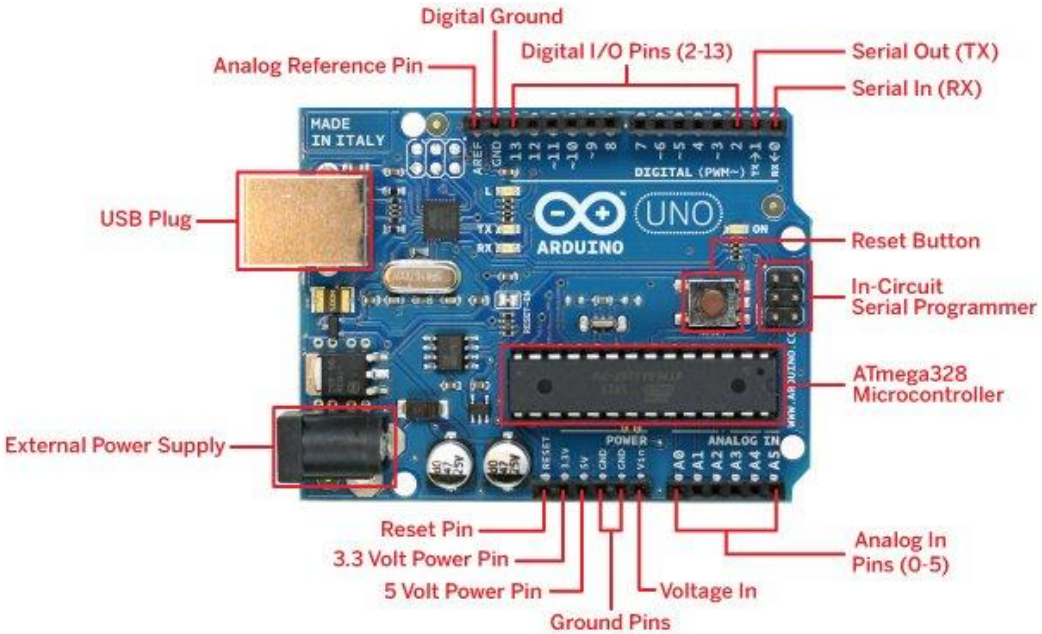
Το Arduino είναι ευέλικτο λόγω όλων αυτών των εξαρτημάτων που είναι τοποθετημένα σε αυτό όπως φαίνεται και στο σχήμα 2.2 παραπάνω. Με άλλα λόγια, καθένα από αυτά τα μικροσκοπικά περιφερειακά στο Arduino Uno έχει έναν σκοπό. Ακολουθεί μια λίστα με όλα τα διαφορετικά μέρη του Arduino

- Η σύνδεση USB του Arduino Uno
- Υποδοχή τροφοδοσίας (βύσμα βαρελιού) στο Arduino
- Μικροελεγκτής ATmega 16U2.
- Το κύριο περιφερειακό Arduino -ATmega 328P.
- Καρφίτσες κεφαλίδας Arduino ICSP.
- Ο κρυσταλλικός ταλαντωτής των 16MHz του Arduino Uno.
- Arduino Ψηφιακές καρφίτσες και θύρες εισόδου / εξόδου
- Οι θύρες και οι καρφίτσες εισόδου Arduino Uno Analog.
- Τροφοδοτικά.

- **Θύρα USB** Η προσθήκη ενός δευτερεύοντος μικροελεγκτή, ATmega 16U2, επιτρέπει στον κύριο επεξεργαστή του Arduino Uno να επικοινωνεί με τον κεντρικό υπολογιστή μέσω USB. Το ATmega 16U2 παρέχει σειριακά δεδομένα στον κύριο επεξεργαστή και διαθέτει ενσωματωμένο περιφερειακό USB.Ο κεντρικός υπολογιστής παρέχει 100mA ρεύματος σε τροφοδοσία 5V στο Arduino Uno για μη απαριθμημένη συσκευή και 500mA σε 5V για απαριθμημένη συσκευή. Μια απαριθμημένη συσκευή είναι μια συσκευή που
  ο υπολογιστής αναγνωρίζει σωστά και έτσι φορτώνει τα κατάλληλα προγράμματα οδήγησης για αυτήν.
- **Ισχύς (Jack Barrel)** Οι πλακέτες Arduino μπορούν να τροφοδοτηθούν απευθείας από το τροφοδοτικό εναλλασσόμενου ρεύματος συνδέοντάς το με το βαρέλι Jack.
- **VIN**. Ακροδέκτης για μη σταθεροποιημένη τάση. Συνήθως εδώ συνδέεται μια εξωτερική πηγή τροφοδοσίας.
- **5V.** Ακροδέκτης σταθεροποιημένης τάσης 5V. Η ρυθμιζόμενη παροχή ηλεκτρικού ρεύματος που χρησιμοποιείται για την τροφοδοσία του μικρό ελεγκτή ή άλλων ηλεκτρονικών στοιχείων της πλακέτας. Αυτό μπορεί να προέρχεται είτε από Vin με ενσωματωμένο ρυθμιστή, ή να παρέχεται από USB ή άλλη ρυθμιζόμενη παροχή 5V
- **3V3.** Μέγιστη κατανάλωση ρεύματος είναι 50mA.
- **GND**. Γειωμένες ακίδες

Η πλακέτα μπορεί να λειτουργήσει με εξωτερική πηγή από 6 έως 20 V. Αν ωστόσο τροφοδοτηθεί με λιγότερα από 7 V τα pin εξόδου 5V δεν θα καταφέρουν να εξάγουν τάση 5 V. Αντίθετα, αν δώσουμε πάνω από 12 V θα υπερθερμανθεί ο σταθεροποιητής τάσης στην πλακέτα και ενδεχομένως να καταστραφεί. Συνεπώς, μια ιδανική τάση είναι τα 9V.

2.3 Εξήγηση των μερών



Σχήμα 2.1.2: Arduino UNO

Παρακάτω αναλύω το σχήμα 2.1.2 στο τι περιέχει και γιατί:

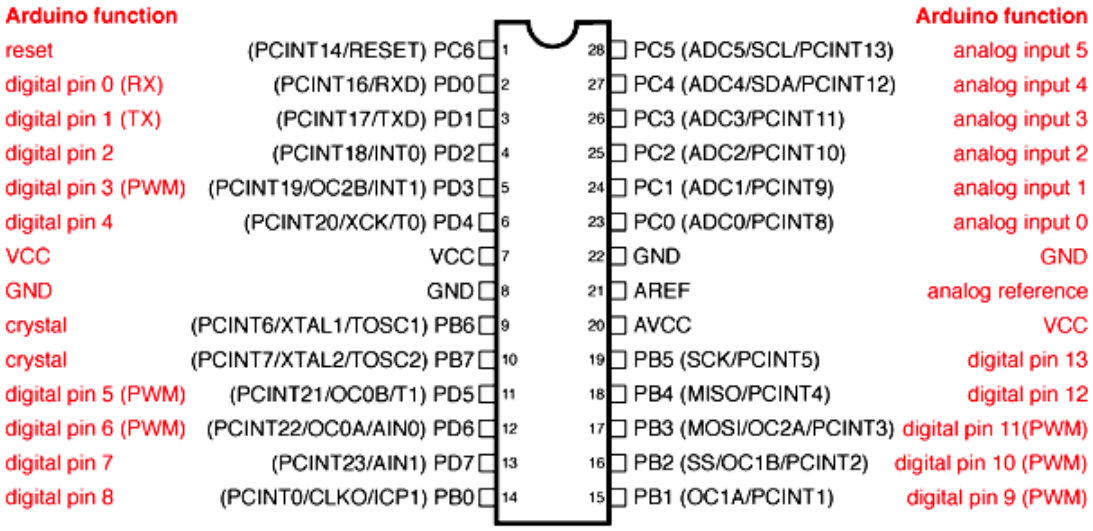
- **Καρφίτσα ICSP**
  Κυρίως, το ICSP είναι ένα AVR, μια μικρή κεφαλίδα προγραμματισμού για το Arduino που αποτελείται από MOSI, MISO, SCK, RESET, VCC και GND. Συχνά αναφέρεται ως SPI (Serial Peripheral Interface), το οποίο θα μπορούσε να θεωρηθεί ως «επέκταση» της εξόδου. Στην πραγματικότητα, δουλεύει τη συσκευή εξόδου στον κύριο του διαύλου SPI.
- **LED TX και RX**
  Στο ταμπλό υπάρχουν: TX (μετάδοση) και RX (λήψη). Εμφανίζονται σε δύο θέσεις στον πίνακα Arduino UNO. Πρώτα, στις ψηφιακές ακίδες 0 και 1, για να υποδείξετε τους ακροδέκτες που είναι υπεύθυνοι για τη σειριακή επικοινωνία. Δεύτερον, το TX και το RX led. Το led TX αναβοσβήνει με διαφορετική ταχύτητα κατά την αποστολή των σειριακών δεδομένων. Η ταχύτητα αναβοσβήνει εξαρτάται από το ρυθμό baud που χρησιμοποιείται από την πλακέτα. Το RX αναβοσβήνει κατά τη διαδικασία λήψης.
- **Ψηφιακά pins εισόδου / εξόδου**
  Η πλακέτα του Arduino UNO διαθέτει 14 ψηφιακές καρφίτσες εισόδου / εξόδου (εκ των οποίων 6 παρέχουν έξοδο PWM (Pulse Width Modulation). Αυτές οι ακίδες μπορούν να διαμορφωθούν ώστε να λειτουργούν ως ψηφιακές ακίδες εισόδου για να διαβάσουν λογικές τιμές (0 ή 1) ή ως ψηφιακές εξόδους εξόδου για την οδήγηση διαφορετικών μονάδων όπως LED, ρελέ κ.λπ. Οι ακίδες με την ένδειξη "~" μπορούν να χρησιμοποιηθούν για τη δημιουργία PWM.
- **Αναλογικές καρφίτσες**
  Ο πίνακας Arduino UNO έχει έξι αναλογικές ακίδες εισόδου A0 έως A5. Αυτές οι ακίδες μπορούν να διαβάσουν το σήμα από έναν αναλογικό αισθητήρα όπως τον αισθητήρα υγρασίας ή τον αισθητήρα θερμοκρασίας και να το μετατρέψουν σε ψηφιακή τιμή που μπορεί να διαβαστεί από τον μικροεπεξεργαστή.
- **AREF**
  Το AREF σημαίνει Αναλογική Αναφορά. Μερικές φορές, χρησιμοποιείται για να ορίσει μια εξωτερική τάση αναφοράς (μεταξύ 0 και 5 Volts) ως το ανώτερο όριο για τους αναλογικούς ακροδέκτες εισόδου.
- **Κρυστάλλινος ταλαντωτής**
  Ο κρυσταλλικός ταλαντωτής βοηθά τον Arduino στην αντιμετώπιση θεμάτων χρόνου.Η συχνότητα είναι 16MHz.
- **Επαναφορά Arduino**
  Μπορούμε να επαναφέρουμε την πλακέτα Arduino, δηλαδή να ξεκινήσουμε το πρόγραμμά μας από την αρχή. Μπορούμε να επαναφέρουμε τον πίνακα UNO με δύο τρόπους.Αρχικά, χρησιμοποιώντας το κουμπί επαναφοράς. Δεύτερον, μπορούμε να συνδέσουμε ένα εξωτερικό κουμπί επαναφοράς στην καρφίτσα Arduino με την ένδειξη RESET.
- **Ρυθμιστής τάσης**
  Η λειτουργία του ρυθμιστή τάσης είναι να ελέγχει την τάση που δίνεται στην πλακέτα Arduino και να σταθεροποιεί τις τάσεις DC που χρησιμοποιούνται από τον επεξεργαστή και άλλα στοιχεία.
- **Ένδειξη LED ισχύος**
  Αυτή η λυχνία LED θα ανάβει όταν συνδέετε το Arduino σε μια πηγή τροφοδοσίας για να υποδείξετε ότι η πλακέτα σας τροφοδοτείται σωστά. Εάν αυτή η λυχνία δεν ανάψει, τότε υπάρχει κάτι λάθος στη σύνδεση.

- **ATmega 16U2**
  Το ATmega 16U2 είναι κυρίως υπεύθυνο για τη μετατροπή σήματος USB / Serial. Στέλνει τα σειριακά δεδομένα στο ATmega 328P και μπορεί να θεωρηθεί ως ενεργοποιητής επικοινωνίας μεταξύ του κεντρικού υπολογιστή και του πίνακα Arduino. Το ATmega 16U2 έχει το δικό του

σύνολο περιφερειακών που βοηθούν στο σημαντικό έργο του. Ως εκ τούτου, αυτά τα περιφερειακά και το ATmega 16U2 μαζί αποτελούν ένα ουσιαστικό υποσύστημα του πίνακα Arduino.

2.4 Μικροελεγκτής Altem ATmega 328 8bit

Ο ATmega328 είναι ένας χαμηλής ισχύος CMOS 8-bit μικροελεγκτής με βάση το AVR σε ενισχυμένη αρχιτεκτονική RISC. Εκτελώντας ισχυρές διαδικασίες σε ένα μοναδικό κύκλο συγχρονισμού,ο ATmega328 επιτυγχάνει πολλαπλά MIPS ανά MHz που επιτρέπει το σύστημα σχεδιαστή να βελτιστοποιήσει την κατανάλωση ενέργειας σε σχέση με την ταχύτητα επεξεργασίας του.Τα datasheet του μικροελεγκτή παρουσιάζονται παρακάτω στο σχήμα 2.4.



Σχήμα 2.4: Μικροελεγκτής Altem ATmega 328 8bit

2.4.1 Περιγραφή των PIN

- **VCC** Ψηφιακή τάση τροφοδοσίας.
- **GND** Γείωση.
- **AVCC** Είναι η τάση τροφοδοσίας για την μετατροπή αναλογικού σε ψηφιακό σήμα.Θα πρέπει να είναι και εξωτερικά συνδεδεμένη και με Vcc ακόμη και αν ο ADC δεν χρησιμοποιείται. Αν ο ADC χρησιμοποιείται, θα πρέπει να συνδεθεί με Vcc μέσω ενός βαθυπερατού φίλτρου.
- **PORTB (PB [7: 0])** Η θύρα B είναι μια θύρα I-O αμφίδρομης 8-bit με εσωτερικές αντιστάσεις pull-up (επιλεγμένες για κάθε bit). Τα buffer εξόδου Port B έχουν συμμετρικά χαρακτηριστικά κίνησης με υψηλή ικανότητα νεροχύτη και πηγή.

Ως είσοδοι,Οι ακίδες της θυρίδας B που τραβούνται εξωτερικά χαμηλά θα δημιουργούν ρεύμα εάν είναι ενεργοποιημένες οι αντιστάσεις έλξης.Οι ακίδες PortB είναι τρι-δηλωμένες όταν η κατάσταση επαναφοράς ενεργοποιηθεί, ακόμη και αν το ρολόι δεν λειτουργεί.Ανάλογα με τις ρυθμίσεις ασφαλειών επιλογής ρολογιού, το PB6 μπορεί να

χρησιμοποιηθεί ως είσοδος στον ενισχυτή ανεστραμμένου ταλαντωτή και είσοδος στο εσωτερικό κύκλωμα λειτουργίας ρολογιού.Ανάλογα με τις ρυθμίσεις ασφαλειών επιλογής ρολογιού, το PB7 μπορεί να χρησιμοποιηθεί ως έξοδος από τον ενισχυτή ανεστραμμένου ταλαντωτή.

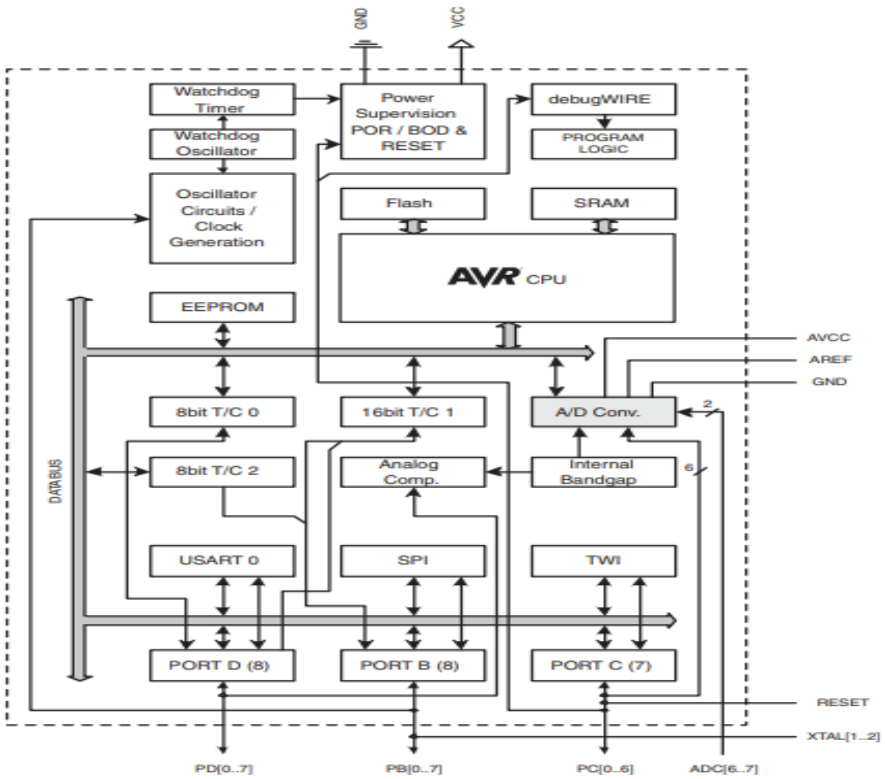
- **PORTC (PC [5: 0])** Η θύρα C είναι μια θύρα αμφίδρομης εισόδου / εξόδου 7-bit με εσωτερικές αντιστάσεις pull-up (επιλεγμένες για κάθε bit).Τα buffer εξόδου υπολογιστή [5: 0] έχουν συμμετρικά χαρακτηριστικά κίνησης με υψηλή ικανότητα νεροχύτη και πηγή.Οι ακίδες της θύρας C που έχουν τραβηχτεί εξωτερικά χαμηλά θα δημιουργούν ρεύμα εάν είναι ενεργοποιημένες οι αντιστάσεις έλξης.Οι ακίδες PortC είναι τρι-δηλωμένες όταν η κατάσταση επαναφοράς ενεργοποιηθεί, ακόμη και αν το ρολόι δεν λειτουργεί.

- **PORTD (PD [7: 0])** Η θύρα D είναι μια θύρα I-O αμφίδρομης 8-bit με εσωτερικές αντιστάσεις pull-up (επιλεγμένες για κάθε bit). Τα buffer εξόδου Port D έχουν συμμετρικά χαρακτηριστικά κίνησης με υψηλή ικανότητα νεροχύτη και πηγή.Ως είσοδοι,οι ακίδες της θύρας D που έχουν τραβηχτεί εξωτερικά χαμηλά θα δημιουργούν ρεύμα εάν είναι ενεργοποιημένες οι αντιστάσεις έλξης. Οι ακίδες PortD είναι τρι-δηλωμένες όταν μια συνθήκη επαναφοράς ενεργοποιηθεί, ακόμα και αν το ρολόι δεν λειτουργεί.

- **AREF** : Το AREF είναι ο αναλογικός πείρος αναφοράς για τον μετατροπέα A /D.

PC6 / RESET Εάν το RSTDISBL Fuse είναι προγραμματισμένο, το PC6 χρησιμοποιείται ως πείρος εισόδου / εξόδου. Σημειώστε ότι τα ηλεκτρικά χαρακτηριστικά του PC6 διαφέρουν από αυτά των άλλων ακίδων του Port C.Εάν η ασφάλεια RSTDISBL δεν είναι προγραμματισμένη, το PC6 χρησιμοποιείται ως είσοδος επαναφοράς. Ένα χαμηλό επίπεδο σε αυτόν τον πείρο για μεγαλύτερο χρονικό διάστημα από το ελάχιστο μήκος παλμού θα δημιουργήσει επαναφορά, ακόμη και αν το ρολόι δεν λειτουργεί. Οι μικρότεροι παλμοί δεν είναι εγγυημένοι για τη δημιουργία επαναφοράς.

- **ADC [7: 6] (Μόνο πακέτο TQFP και VFQFN)** Στο πακέτο TQFP και VFQFN, το ADC [7: 6] χρησιμεύει ως αναλογική είσοδος στον μετατροπέα A / D. Αυτές οι ακίδες τροφοδοτούνται από την αναλογική τροφοδοσία και χρησιμεύουν ως κανάλια ADC 10-bit.



Σχήμα 2.4.1: Block Διάγραμμα

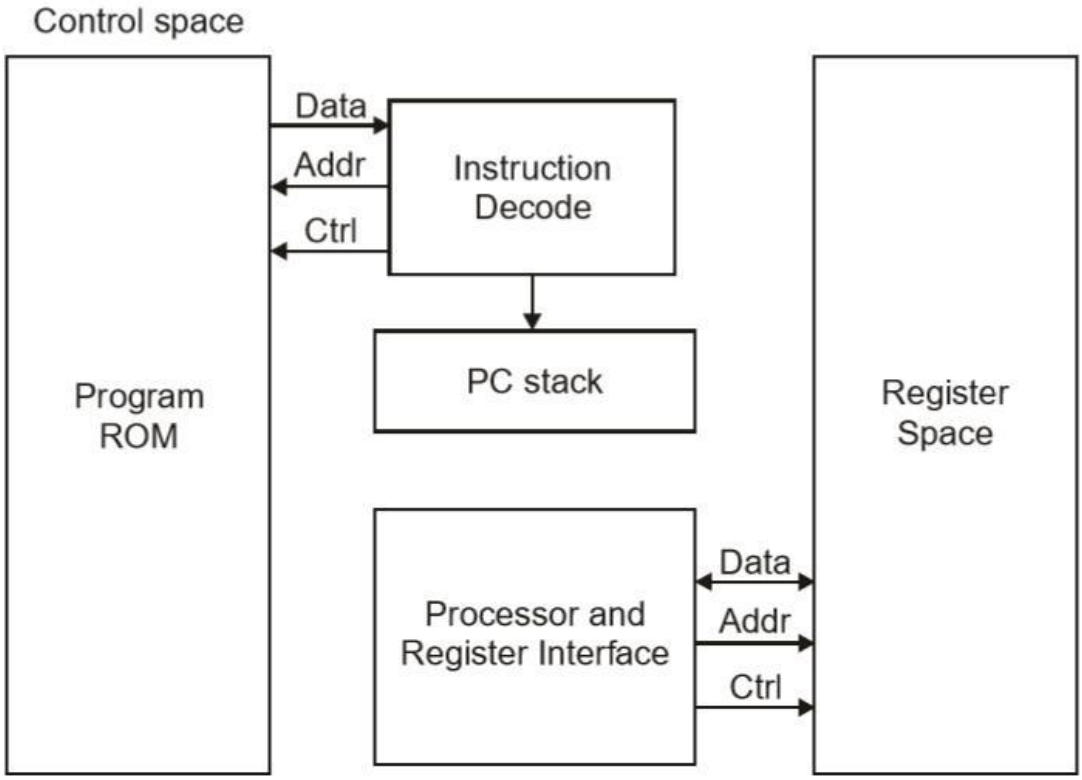
Ο πυρήνας γενικά ενός AVR επεξεργαστή και στην περίπτωση μας του ATmega328 συνδυάζει ένα πλούσιο σετ εντολών με 32 καταχωρητές εργασίας γενικού σκοπού όπως φαίνεται και στο σχήμα (2.4.1). Και οι 3 καταχωρητές είναι άμεσα συνδεδεμένοι με την αριθμητική λογική μονάδα (ALU) επιτρέποντας δύο ανεξάρτητοι καταχωρητές να έχουν πρόσβαση σε μια απλή εντολή ενός κύκλου ρολογιού.Η δομή που προκύπτει έχει ποιο αποτελεσματικό κώδικα επιτυγχάνοντας παράλληλα ταχύτατα επεξεργασίας έως και 10 φορές μεγαλύτερη από έναν συμβατικό μικροελεκτή τύπου CISC. Ο ATmega 328 παρέχει τις ακόλουθες λειτουργίες : 32Kbytes of In-System Programmable Flash με δυνατότητα ανάγνωσης-εγγραφής, 1Kbytes EEPROM, 2Kbytes SRAM, 23 γραμμές γενικής χρήσης για είσοδο-έξοδο, 32 καταχωριτές εργασίας γενικού σκοπού, 3 απαριθμητές/χρόνισες, εσωτερικές και εξωτερικές διακοπές, μια σειριακά προγραμματιζόμενη USART, μια SPI σειριακή θύρα, έναν 6καναλο 10bit DAC, χρονιστή εσωτερικά προγραμματιζόμενο με εσωτερικό κρύσταλλο.
Με στόχο να αυξήσουν την απόδοση τους οι οικογένεια επεξεργαστών AVR και κατά συνέπεια και ο ATmega 328 βασίζονται και κάνουν χρήση της αρχιτεκτονικής Harvard με ξεχωριστές θέσεις μνήμης και διαύλους για το πρόγραμμα και τα δεδομένα.Οι οδηγίες στη μνήμη προγράμματος εκτελούνται με ένα μόνο επίπεδο αγωγών. Όταν μια εντολή εκτελείται, η επόμενη εντολή προανασύρεται από τη μνήμη προγράμματος.Αύτη η σχεδίαση επιτρέπει κάθε εντολή να εκτελείτε σε έναν κύκλο ρολογιού.

2.4.2 Αρχιτεκτονική Harvard και Von – Neumann

Η αρχιτεκτονική του Harvard

Η αρχιτεκτονική του Χάρβαρντ προσφέρει ξεχωριστούς διαύλους αποθήκευσης και σήματος για οδηγίες και δεδομένα. Αυτή η αρχιτεκτονική έχει αποθήκευση δεδομένων εξ ολοκλήρου

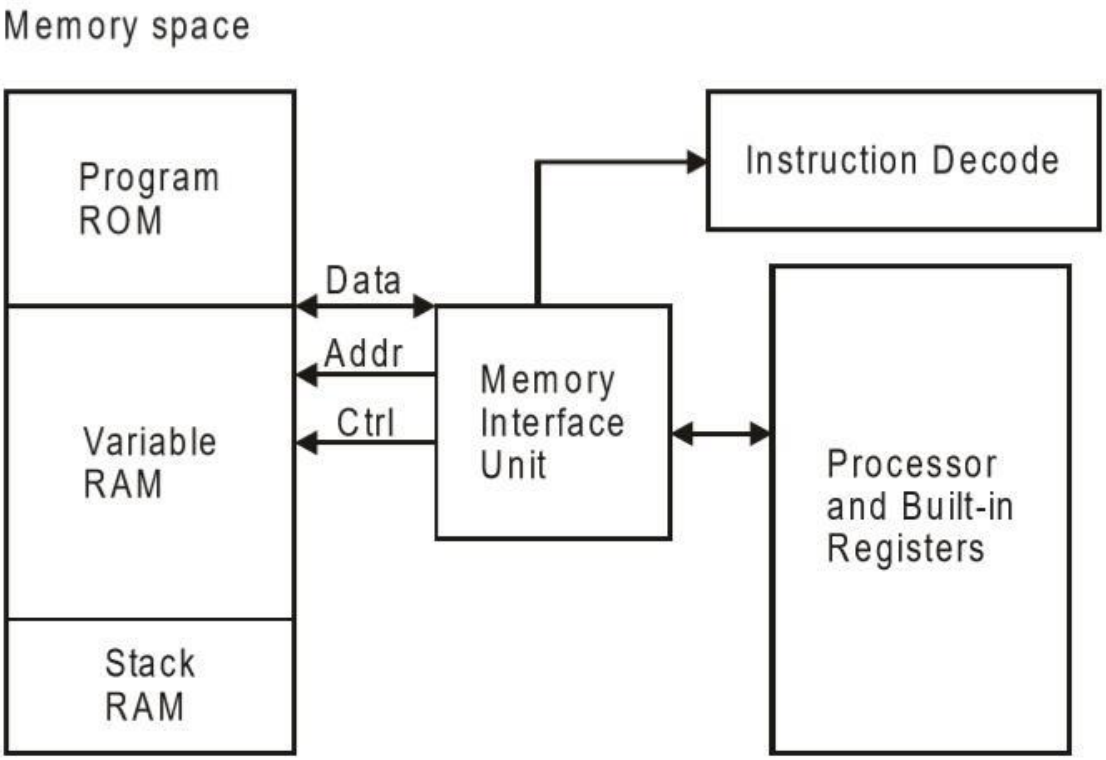
μέσα στην CPU και δεν υπάρχει πρόσβαση στην αποθήκευση εντολών ως δεδομένα. Οι υπολογιστές διαθέτουν ξεχωριστές περιοχές μνήμης για οδηγίες προγράμματος και δεδομένα χρησιμοποιώντας εσωτερικούς διαύλους δεδομένων, επιτρέποντας την ταυτόχρονη πρόσβαση τόσο σε οδηγίες όσο και σε δεδομένα.Προγράμματα που πρέπει να φορτωθούν από έναν χειριστή. ο επεξεργαστής δεν μπορούσε να εκκινήσει μόνος του. Σε μια αρχιτεκτονική του Χάρβαρντ, δεν υπάρχει ανάγκη να κάνετε τις δύο μνήμες να μοιράζονται ιδιότητες(σχήμα 2.4.2)
.



Σχήμα 2.4.2: Αρχιτεκτονική Harvard

Αρχιτεκτονική Von Neumann:

Η αρχιτεκτονική Von Neumann προτάθηκε για πρώτη φορά από έναν επιστήμονα υπολογιστών John von Neumann. Σε αυτήν την αρχιτεκτονική, υπάρχει μια διαδρομή δεδομένων ή δίαυλος τόσο για εντολές όσο και για δεδομένα. Ως αποτέλεσμα, η CPU κάνει μία λειτουργία τη φορά. Είτε λαμβάνει μια εντολή από τη μνήμη, είτε εκτελεί λειτουργία ανάγνωσης/εγγραφής σε δεδομένα. Έτσι, μια ανάκτηση εντολών και μια λειτουργία δεδομένων δεν μπορούν να συμβούν ταυτόχρονα, με κοινή χρήση ενός κοινού διαύλου. Η αρχιτεκτονική Von-Neumann υποστηρίζει απλό υλικό. Επιτρέπει τη χρήση μιας ενιαίας, διαδοχικής μνήμης. Οι σημερινές ταχύτητες επεξεργασίας ξεπερνούν κατά πολύ τους χρόνους πρόσβασης στη μνήμη και χρησιμοποιούμε πολύ γρήγορη αλλά μικρή ποσότητα μνήμης (cache) τοπική στον επεξεργαστή (σχήμα 2.4.20).



Σχήμα 2.4.20: Αρχιτεκτονική Von Neumann

2.4.3 Διαφορά μεταξύ Αρχιτεκτονικής Harvard και Von Neumann

Αρχιτεκτονική Harvard

1. Διαχώρισε τις μνήμες για κώδικα και δεδομένα.
2. Ένας μόνος κύκλος ρολογιού είναι αρκετός, καθώς χρησιμοποιούνται ξεχωριστοί δίαυλοι για πρόσβαση σε κώδικα και δεδομένα.
3. Πιο αργή ταχύτητα, άρα πιο χρονοβόρα.
4. Πολύπλοκο στο σχεδιασμό.

Αρχιτεκτονική Von Neumann

1. Ενιαία μνήμη για κοινή χρήση τόσο από κώδικα όσο και από δεδομένα.
2. Ο επεξεργαστής πρέπει να ανακτήσει κώδικα σε ξεχωριστό κύκλο ρολογιού και δεδομένα σε άλλο κύκλο ρολογιού. Άρα απαιτεί δύο κύκλους ρολογιού.
3. Μεγαλύτερη ταχύτητα, άρα λιγότερο χρονοβόρα.
4. Απλό στο σχεδιασμό.

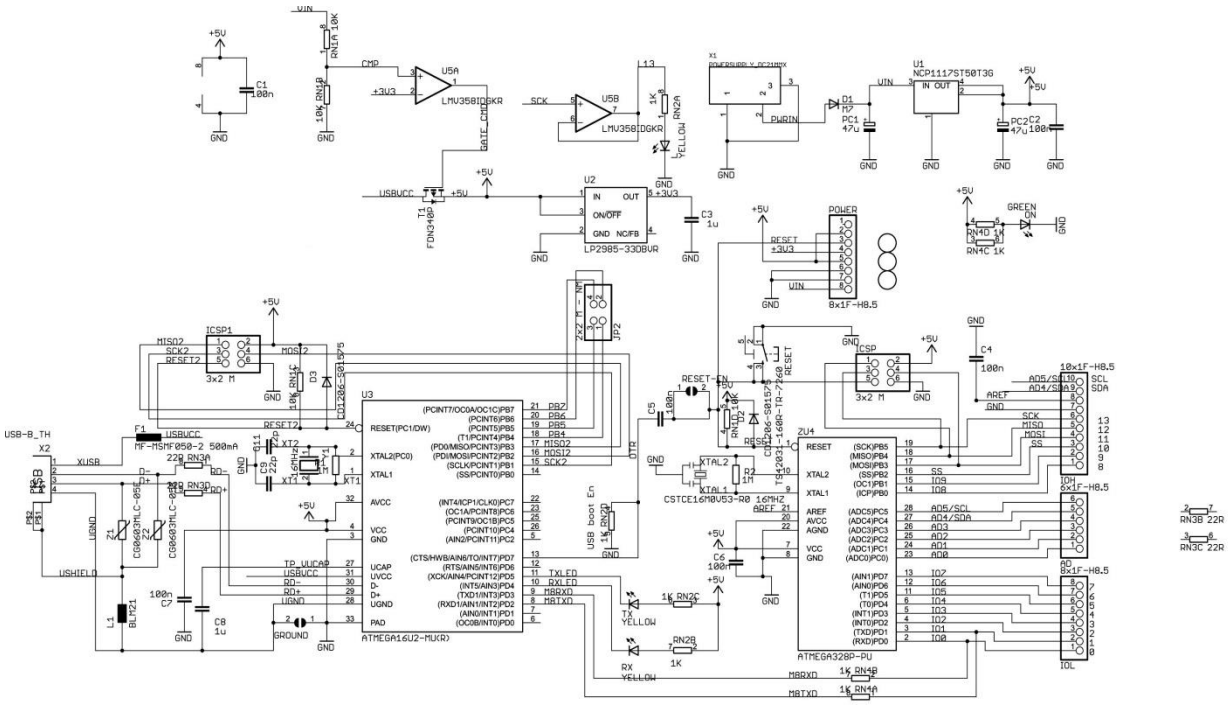
2.4.4 Χαρακτηριστικά

ΕΠΕΞΕΡΓΑΣΤΗΣ	8-bit AVR

Αριθμός καρφιτσών	28
Τάση λειτουργίας (V)	+1,8 V ΕΩΣ +5,5 V
Αριθμός προγραμματιζόμενων γραμμών I/O	23
Διεπαφή επικοινωνίας	Σειριακή διεπαφή Master/Slave SPI (17,18,19 PINS) [Μπορεί να χρησιμοποιηθεί για τον προγραμματισμό αυτού του ελεγκτή] Προγραμματιζόμενη σειριακή USART (2,3 PINS) [Μπορεί να χρησιμοποιηθεί για τον προγραμματισμό αυτού του ελεγκτή] Σειριακή διεπαφή δύο καλωδίων (27,28 PINS)[Μπορεί να χρησιμοποιηθεί για τη σύνδεση περιφερειακών συσκευών όπως Servo, αισθητήρες και συσκευές μνήμης]
Διεπαφή JTAG	Μη διαθέσιμος
Μονάδα ADC	6 κανάλια, ADC ανάλυσης 10 bit
Μονάδα χρονοδιακόπτη	Δύο μετρητές 8 bit με ξεχωριστή λειτουργία Prescaler και σύγκριση, ένας μετρητής 16 bit με ξεχωριστή Prescaler, λειτουργία σύγκρισης και λειτουργία λήψης.
Αναλογικοί συγκριτές	1 (12,13 PINS)
Μονάδα DAC	Μηδέν
Κανάλια PWM	6
Εξωτερικός Ταλαντωτής	0-4MHz 1,8V έως 5,5V 0-10MHz 2,7V έως 5,5V 0-20MHz 4,5V έως 5,5V

Εσωτερικός Ταλαντωτής	Βαθμονομημένος εσωτερικός ταλαντωτής 8 MHz
Τύπος μνήμης προγράμματος	Λάμψη
Μνήμη προγράμματος ή μνήμη Flash	32 Kbyte[10000 κύκλοι εγγραφής/διαγραφής]
Ταχύτητα CPU	1MIPS για 1MHz
ΕΜΒΟΛΟ	2 Kbyte Εσωτερική SRAM
EEPROM	1 Kbyte EEPROM
Watchdog Timer	Προγραμματιζόμενος χρονοδιακόπτης Watchdog με ξεχωριστό On-chipOscillator
Κλείδωμα προγράμματος	Ναί
Λειτουργίες εξοικονόμησης ενέργειας	Έξι λειτουργίες [Αδράνεια, Μείωση θορύβου ADC, Εξοικονόμηση ενέργειας, Απενεργοποίηση, Αναμονή και Εκτεταμένη Αναμονή]
Θερμοκρασία λειτουργίας	-40°C έως +105°C (+105 είναι απόλυτο μέγιστο, -40 είναι απόλυτο ελάχιστο)

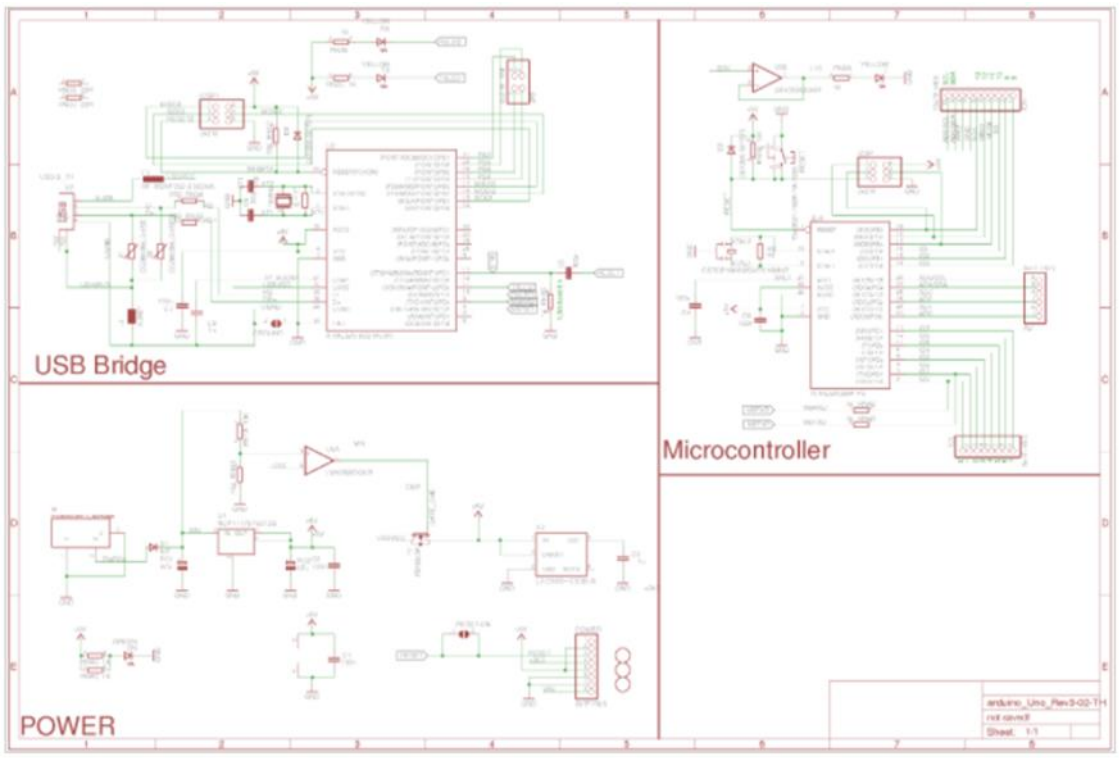
Πίνακας 2.4.4: Χαρακτηριστικά Atmega328p



Σχήμα 2.4.5: Arduino uno datasheet

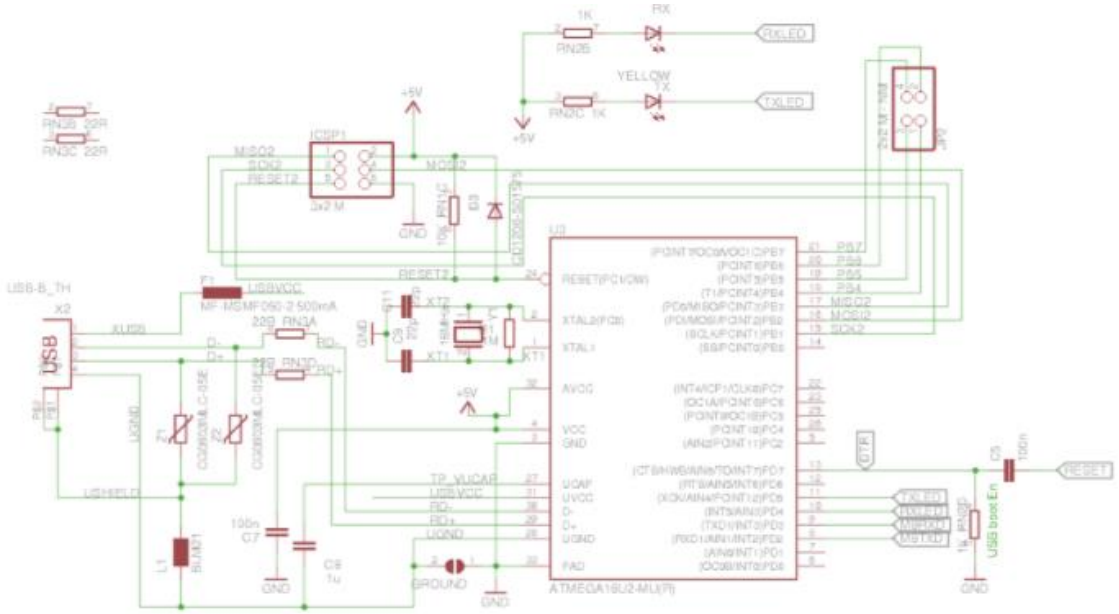
2.5 Επισκόπηση συστήματος Arduino UNO

Μετά τη σύνταξη του κωδικού σας χρησιμοποιώντας το Arduino IDE, θα πρέπει να μεταφορτωθεί στον κύριο μικροελεγκτή του Arduino UNO χρησιμοποιώντας μια σύνδεση USB. Επειδή ο κύριος μικροελεγκτής δεν διαθέτει πομποδέκτη USB, χρειάζεστε μια γέφυρα για τη μετατροπή σημάτων μεταξύ της σειριακής διεπαφής (διεπαφή UART) του μικροελεγκτή και των σημάτων USB του κεντρικού υπολογιστή.Η γέφυρα στην τελευταία αναθεώρηση είναι το ATmega16U2, το οποίο διαθέτει πομποδέκτη USB και επίσης σειριακή διεπαφή (διεπαφή UART).Για να τροφοδοτήσετε την πλακέτα Arduino, μπορείτε να χρησιμοποιήσετε το USB ως πηγή τροφοδοσίας. Μια άλλη επιλογή είναι να χρησιμοποιήσετε μια υποδοχή DC. Για να επαναφέρετε την πλακέτα σας, πρέπει να χρησιμοποιήσετε ένα μπουτόν στον πίνακα. Μια άλλη πηγή επαναφοράς θα πρέπει να είναι κάθε φορά που ανοίγετε τη σειριακή οθόνη από το Arduino IDE.Όλα υτά παρουσιάζονται παρακάτω στο σχήμα 2.5.



Σχήμα 2.5: Αναδιανεμημένη έκδοση του αρχικού σχήματος Arduino

2.5.1 Η γέφυρα USB-προς-UART



Σχήμα 2.5.1: USB προς UART

Ο ρόλος του τμήματος γέφυρας USB σε UART (σχήμα 2.5.1) είναι να μετατρέψει τα σήματα της διεπαφής USB στη διεπαφή UART, την οποία κατανοεί το ATmega328, χρησιμοποιώντας ένα ATmega16U2 με έναν εσωτερικό πομποδέκτη USB. Αυτό γίνεται χρησιμοποιώντας ειδικό υλικολογισμικό που έχει φορτωθεί στο ATmega16U2.

Από άποψη ηλεκτρονικής σχεδίασης, αυτή η ενότητα είναι παρόμοια με την ενότητα μικροελεγκτή. Αυτό το MCU διαθέτει κεφαλίδα ICSP, εξωτερικό κρύσταλλο με πυκνωτές φορτίου (CL) και πυκνωτή φίλτρου Vcc.
Υπάρχουν αντιστάσεις σειράς στις γραμμές D + και D- USB. Αυτά παρέχουν τη σωστή αντίσταση τερματισμού για τα σήματα USB.

Τα Z1 και Z2 είναι αντιστάσεις που εξαρτώνται από την τάση (VDR), που ονομάζονται επίσης βαρίστορ. Χρησιμοποιούνται για την προστασία των γραμμών USB από μεταβατικά ESD.

Ο πυκνωτής 100nF συνδεδεμένος σε σειρά με τη γραμμή επαναφοράς επιτρέπει στο Atmega16U2 να στείλει έναν παλμό επαναφοράς στο Atmega328.

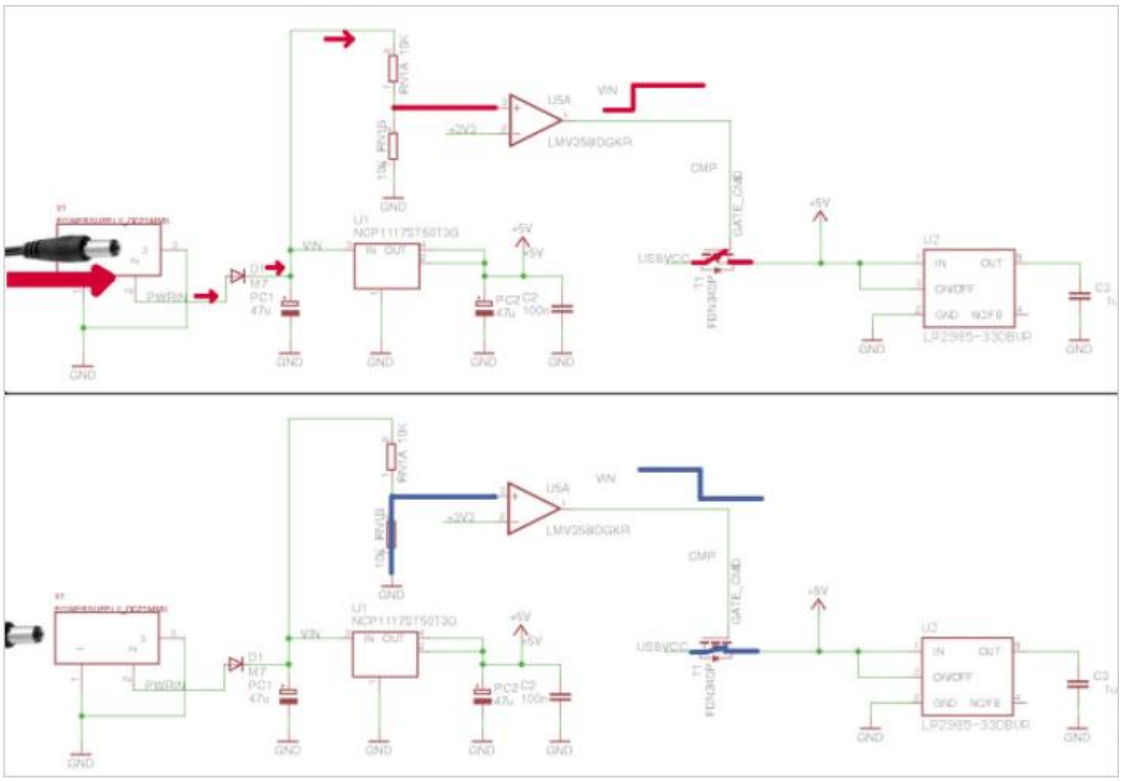
2.5.2 The Power

Για πηγή τροφοδοσίας, έχετε την επιλογή να χρησιμοποιήσετε την υποδοχή USB ή DC. Εάν συνδέσω έναν προσαρμογέα DC και το USB, η πηγή τροφοδοσίας θα είναι:

Ο ρυθμιστής 5V είναι ο NCP1117ST50T3G και το Vin αυτού του ρυθμιστή συνδέεται μέσω εισόδου DC jack μέσω της διόδου M7, της έκδοσης SMD της διάσημης διόδου 1N4007.Αυτή η δίοδος παρέχει προστασία αντίστροφης πολικότητας.

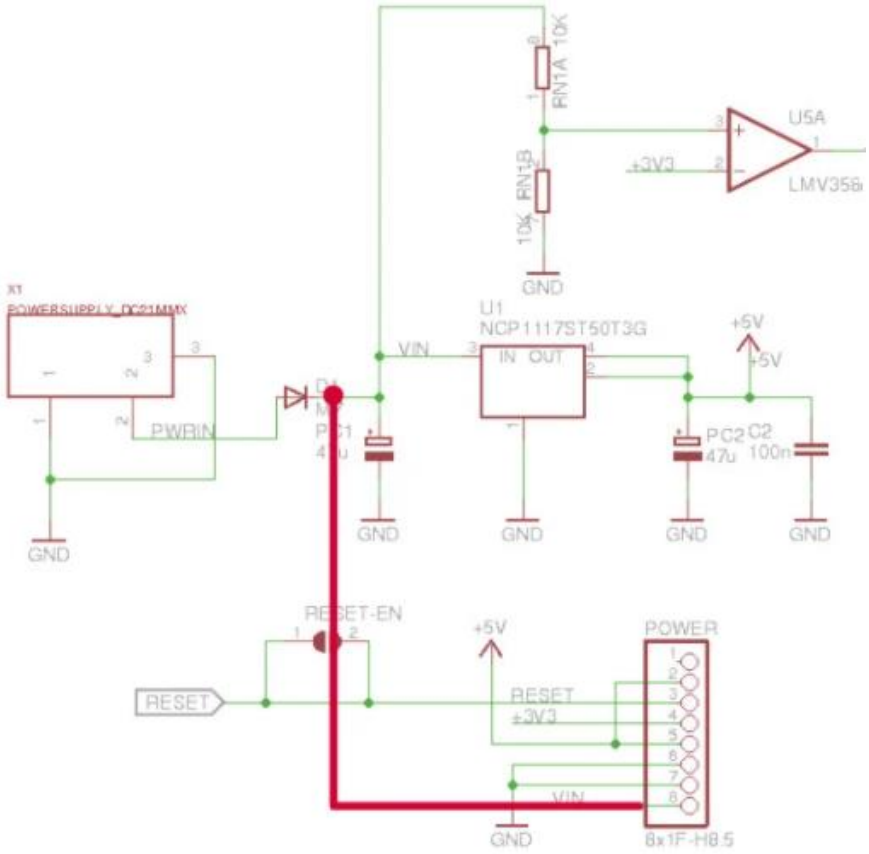
Η έξοδος του ρυθμιστή 5V συνδέεται με το υπόλοιπο δίκτυο 5V στο κύκλωμα και επίσης με την είσοδο του ρυθμιστή 3.3V, LP2985-33DBVR. Μπορείτε να αποκτήσετε πρόσβαση 5V απευθείας από τον ακροδέκτη 5V κεφαλίδας ισχύος.

Μια άλλη πηγή 5V είναι το USBVCC που συνδέεται με την αποστράγγιση ενός FDN340P, ενός καναλιού P MOSFET και η πηγή συνδέεται στο δίκτυο 5V. Η πύλη του τρανζίστορ συνδέεται με την έξοδο ενός LMV358 op-amp που χρησιμοποιείται ως συγκριτής. Η σύγκριση είναι μεταξύ 3V3 και Vin / 2. Όταν το Vin / 2 είναι μεγαλύτερο, αυτό θα παράγει υψηλή έξοδο από το συγκριτή και το P-channel MOSFET είναι απενεργοποιημένο. Εάν δεν εφαρμόζεται Vin, το V + του συγκριτή τραβιέται προς τα κάτω στο GND και το Vout είναι χαμηλό, έτσι ώστε το τρανζίστορ να είναι ενεργοποιημένο και το USBVCC να είναι συνδεδεμένο σε 5V(σχήμα 2.5.2).



Σχήμα 2.5.2: Μηχανισμός μεταγωγής πηγής ισχύος

Ο LP2985-33DBVR είναι ο ρυθμιστής 3V3. Και οι ρυθμιστές 3V3 και 5V είναι LDO (Low Dropout), πράγμα που σημαίνει ότι μπορούν να ρυθμίσουν την τάση ακόμη και αν η τάση εισόδου είναι κοντά στην τάση εξόδου. Αυτό είναι μια βελτίωση σε σχέση με τους παλαιότερους γραμμικούς ρυθμιστές, όπως το 7805 .
Όπως αναφέρθηκε παραπάνω, το VIN από μια υποδοχή DC προστατεύεται από την αντίστροφη πολικότητα χρησιμοποιώντας μια σειριακή δίοδο M7 στην είσοδο. Λάβετε υπόψη ότι ο πείρος VIN στην κεφαλίδα τροφοδοσίας δεν προστατεύεται. Αυτό συμβαίνει επειδή συνδέεται μετά τη δίοδο M7.



Σχήμα 2.5.20: Καρφίτσα VIN από κεφαλίδα ισχύος

Όταν χρησιμοποιείτε USB ως πηγή τροφοδοσίας και για την προστασία της θύρας USB, υπάρχει μια ασφάλεια PTC (συντελεστής θετικής θερμοκρασίας) σε σειρά με το USBVCC. Αυτό παρέχει προστασία από υπερένταση, 500mA. Όταν επιτευχθεί ένα όριο υπερέντασης, η αντίσταση PTC αυξάνεται πολύ. Η αντίσταση μειώνεται μετά την αφαίρεση του υπερβολικού ρεύματος (σχήμα 2.5.20).

2.6 Επίλογος

Όπως αναφέρθηκε και παραπάνω ο μικροελεγκτής είναι η καρδιά του Arduino.Σε αυτό το κεφάλαιο είδαμε το πως λειτουργεί ένας μικροελεγκτής και συγκεκρίμενα ο Atmega 238p.Αναφερθήκαμε στα χαρακτηριστικά του που τον κάνουν μοναδικό.Τις διάφορες αρχιτεκτονικές που υπάρχουν στους μικροελεγκτές.Επίσης ο μικροελεγκτής Atmega 328p βλέπουμε ότι δεν μεταγλωττίζει τα σήματα από το USB.Για αυτό υπάρχει ο Atmega16U2 που μεταγλωττίζει τα σήματα και τα στέλνει στον Atmega328p.Για τον Atmega 16U2 δεν θα μπούμε σε λεπτομέρειες.

Κεφάλαιο 3ο: Το πρόγραμμα Arduino IDE

3.1 Εισαγωγή

Ένα επίσημο λογισμικό που εισήγαγε το Arduino.cc, το οποίο χρησιμοποιείται και φαίνεται στο σχήμα 3.1 κυρίως για τη σύνταξη, τη μεταγλώττιση και τη μεταφόρτωση του κώδικα στη Συσκευή Arduino. Σχεδόν όλες οι λειτουργικές μονάδες Arduino είναι συμβατές με αυτό το λογισμικό που είναι ανοιχτού κώδικα και είναι άμεσα διαθέσιμο για εγκατάσταση και έναρξη της σύνταξης του κώδικα εν κινήσει. Μεταφορτώνουμε δωρεάν την τελευταία έκδοση Arduino IDE από την ιστοσελίδα https://www.arduino.cc/en/software.

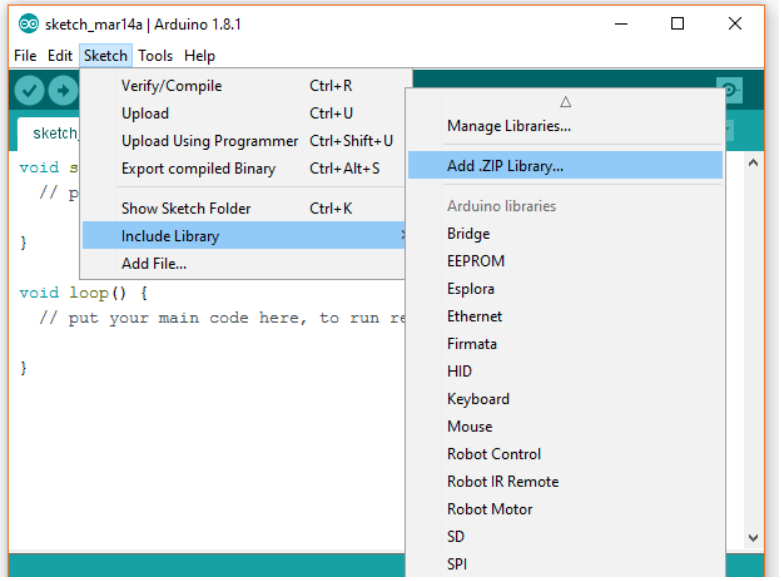
- Είναι ένα λογισμικό ανοιχτού κώδικα που χρησιμοποιείται κυρίως για τη σύνταξη και τη σύνταξη του κώδικα στο Arduino Module.
- Είναι ένα επίσημο λογισμικό Arduino, κάνοντας τη συλλογή κωδικών πολύ εύκολη, ώστε ακόμη και ένα κοινό άτομο χωρίς προηγούμενη τεχνική γνώση να μπορεί να βραχεί με τη διαδικασία εκμάθησης.
- Είναι εύκολα διαθέσιμο για λειτουργικά συστήματα όπως MAC, Windows, Linux και εκτελείται στην πλατφόρμα Java που συνοδεύεται από ενσωματωμένες λειτουργίες και εντολές που διαδραματίζουν ζωτικό ρόλο για τον εντοπισμό σφαλμάτων, την επεξεργασία και τη σύνταξη του κώδικα στο περιβάλλον.
- Μια σειρά από διαθέσιμες ενότητες Arduino, συμπεριλαμβανομένων των Arduino Uno, Arduino Mega, Arduino Leonardo και πολλά άλλα.
- Κάθε ένα από αυτά περιέχει έναν μικροελεγκτή στον πίνακα που είναι πραγματικά προγραμματισμένος και δέχεται τις πληροφορίες με τη μορφή κώδικα.
- Ο κύριος κώδικας, επίσης γνωστός ως σκίτσο, που δημιουργήθηκε στην πλατφόρμα IDE θα δημιουργήσει τελικά ένα Hex File το οποίο στη συνέχεια θα μεταφερθεί και θα φορτωθεί στον ελεγκτή του πίνακα.
- Το περιβάλλον IDE περιέχει κυρίως δύο βασικά μέρη: το πρόγραμμα επεξεργασίας και το μεταγλωττιστή όπου το πρώτο χρησιμοποιείται για τη σύνταξη του απαιτούμενου κώδικα και αργότερα χρησιμοποιείται για τη μεταγλώττιση και τη μεταφόρτωση του κώδικα στη δεδομένη ενότητα Arduino.

Σχήμα 3.1: Αρχική Arduino IDE

Verify/Compile : Έλεγχος για λάθη στον κώδικα.
Upload: Ανέβασμα του κώδικα στον μικροελεγκτή.
New: Δημιουργεί ένα νέο sketch.
Open: Παρουσιάζει ένα μενού με όλα τα sketch,
κάνοντας κλικ σε ένα από αυτά θα ανοίξει μέσα στο τρέχον παράθυρο.
Save:Αποθηκεύει το sketch.
Serial Monitor: Ανοίγει την σειριακή οθόνη ώστε να μπορούμε να δώσουμε δεδομένα από το πληκτρολόγιο.

3.2 Βιβλιοθήκη Arduino

Οι βιβλιοθήκες είναι αρχεία γραμμένα σε C ή C ++ (.c, .cpp) που παρέχουν στα σκίτσα επιπλέον λειτουργικότητα. Πέρα από μια δήλωση συμπερίληψης, μια βιβλιοθήκη είναι μια συλλογή κώδικα που συντάσσεται με το κύριο σκίτσο σας, αλλά δεν γράφεται ρητά εκεί έξω. Μια βιβλιοθήκη Arduino περιλαμβάνει συνήθως συναρτήσεις και μεταβλητές που μπορούν να κληθούν στο σκίτσο, μαζί με μια ειδική λειτουργία γνωστή ως κατασκευαστής που χρησιμοποιείται για τη δημιουργία παρουσιών μιας κλάσης που ορίζεται στη βιβλιοθήκη.



Σχήμα 3.2: Προσθήκη βιβλιοθήκης

Οι βιβλιοθήκες είναι πολύ χρήσιμες με κώδικα που δεν αλλάζει, ως κάτι του «μαύρου κουτιού» στο οποίο οι τιμές μπορεί να εισαχθούν για την αυτόματη εκτέλεση ορισμένων λειτουργιών. Στην ιδανική περίπτωση, δεν πρέπει ποτέ να ανησυχείτε για το τι συμβαίνει στο εσωτερικό. Η ίδια λειτουργικότητα μπορεί να προγραμματιστεί στο κύριο σκίτσο **.ino** , αλλά η εκφόρτωση πραγμάτων σε μια βιβλιοθήκη μπορεί να κάνει τον κώδικα πολύ πιο εύκολο να διαβαστεί.Αυτό δεν εξοικονομεί πραγματικά μνήμη όταν μεταγλωττίζεται και αποστέλλεται στο Arduino (πιο πρακτικά στο σχήμα 3.2).

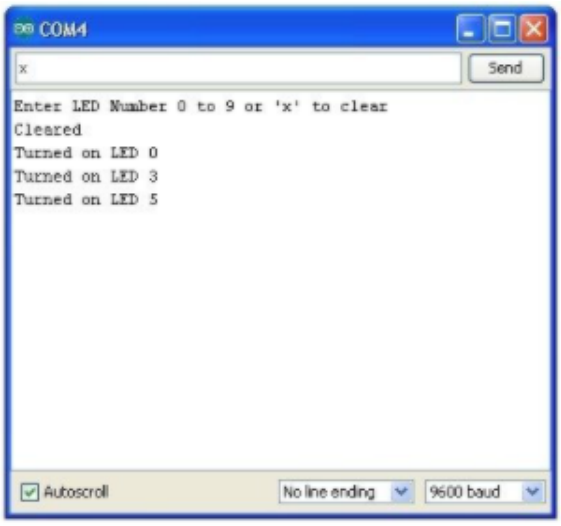
#include
χρησιμοποιείται για να συμπεριλάβει εξωτερικές βιβλιοθήκες. Αυτό δίνει στον προγραμματιστή πρόσβαση σε μια μεγάλη ομάδα τυπικών βιβλιοθηκών C (ομάδες προκατασκευασμένων λειτουργιών), καθώς και βιβλιοθήκες που έχουν γραφτεί ειδικά για το Arduino.

Σύνταξη

```
#include <LibraryFile.h>
#include "LocalFile.h"
```

LibraryFile.h: Όταν χρησιμοποιείται η σύνταξη γωνιακών αγκυλών, οι διαδρομές βιβλιοθηκών θα αναζητηθούν για το αρχείο.
LocalFile.h: Όταν χρησιμοποιείται η σύνταξη διπλών εισαγωγικών, #includeθα αναζητηθεί ο φάκελος του αρχείου που χρησιμοποιεί την οδηγία για το καθορισμένο αρχείο και, στη συνέχεια, οι διαδρομές των βιβλιοθηκών αν δεν βρέθηκαν στην τοπική διαδρομή.

3.3 Σειριακή οθόνη



Σχήμα 3.3: Serial Monitor

Αυτό το παράθυρο ονομάζεται Serial Monitor (σχήμα 3.3) και είναι μέρος του λογισμικού Arduino IDE. Η δουλειά του είναι να μας επιτρέπει να στέλνουμε μηνύματα από τον υπολογιστή μας σε μια πλακέτα Arduino (μέσω USB) και επίσης να λαμβάνουμε μηνύματα από το Arduino.

Το μήνυμα "Enter LED Number 0 to 9 or" x "to clear" εστάλη από το Arduino και μας λέει ποιες εντολές μπορούμε να στείλουμε στο Arduino που είναι είτε να στείλουμε το "x" (για να γυρίσουμε όλα τα Απενεργοποιημένες λυχνίες LED) ή τον αριθμό της λυχνίας LED που θέλετε να ανάψετε (όπου το 0 είναι το κάτω LED, το 1 είναι το επόμενο μέχρι το 7 για το κορυφαίο LED). Η πληκτρολόγηση x, δεν θα έχει αποτέλεσμα, εάν οι λυχνίες LED είναι ήδη σβηστές, αλλά καθώς εισάγουμε κάθε αριθμό, η αντίστοιχη λυχνία LED θα ανάψει και θα λάβουμε ένα μήνυμα επιβεβαίωσης από την πλακέτα Arduino, έτσι ώστε το Serial Monitor να εμφανίζεται όπως φαίνεται παρακάτω.

Σχήμα 3.3.1: Μήνυμα από το Arduino.

Για να επιτευχθεί η σωστή επικοινωνία θα πρέπει και οι δύο πλευρές, Η/Υ και Arduino, να έχουν τον ίδιο ρυθμό μετάδοσης δεδομένων (baud rate).Όπως φαίνεται και στο σχήμα 3.3.1. Για εμάς πάντα ο ρυθμός μετάδοσης θα είναι τα 9600kbps.

Σε προγραμματιστικό επίπεδο, ώστε να μπορέσουμε να πούμε στο Arduino πως θα επικοινωνήσουμε με αυτό, η εντολή που χρησιμοποιούμε είναι η **Serial.begin(9600);** Η τιμή 9600 αναφέρεται στον ρυθμό μετάδοσης που είπαμε παραπάνω. Την εντολή αυτή την βάζουμε πάντα στο κομμάτι της void setup().

```
  void setup(){
      Serial.begin(9600);

}
```

Για να μας εκτυπώσει το Arduino ένα μήνυμα στην σειριακή οθόνη, θα πρέπει μέσα στον κώδικα να κάνουμε χρήση της εντολής **Serial.print("Hello World");**. Αν θέλουμε το επόμενο μήνυμα να ξεκινάει στην επόμενη γραμμή τότε κάνουμε χρήση της **println**.

```
  void setup(){
      Serial.begin(9600);
      Serial.println("Hello World");
  }
```

3.4 Σύνταξη και δομή του κώδικα

Ο δυαδικός κώδικας του μηχανήματος φορτώνεται απευθείας στον ίδιο τον μικροελεγκτή, ο οποίος μοιάζει με ένα χαοτικό σύνολο γραμμάτων και αριθμών. Αυτός ο κωδικός μπορεί να ληφθεί από οποιαδήποτε γλώσσα προγραμματισμού, εξαρτάται από το περιβάλλον ανάπτυξης και από τον διερμηνέα . Το επίσημο περιβάλλον ανάπτυξης είναι το Arduino IDE, όπου ο προγραμματισμός πραγματοποιείται σε C ++, μία από τις πιο δημοφιλείς και ισχυρές γλώσσες. Οι ίδιοι οι προγραμματιστές καλούν την Wiring γλώσσα Arduino, καθώς η τυπική βιβλιοθήκη Arduino.h χρησιμοποιεί λειτουργίες και εργαλεία από το πλαίσιο καλωδίωσης.

Σύνταξη

- Τα σώματα λειτουργίας περικλείονται σε σγουρά τιράντες { }
- Κάθε εντολή τελειώνει με ερωτηματικό ;
- Η μέθοδος εφαρμόζεται στο αντικείμενο μέσω ενός σημείου. Παράδειγμα: Serial.begin();
- Μια συνάρτηση ή μια κλήση μεθόδου τελειώνει πάντα με παρενθέσεις, ακόμη και αν η συνάρτηση δεν έχει παραμέτρους. Παράδειγμα: loop ()
- Ο δεκαδικός διαχωριστής είναι μια τελεία . Παράδειγμα: 0. 25 Το κόμμα έχει διαφορετική χρήση εδώ.
- Τα επιχειρήματα για συναρτήσεις και μεθόδους, καθώς και μέλη πίνακα, αναφέρονται με κόμματα. Παράδειγμα:digitalWrite (3, HIGH) ; πίνακας - int myArray [] = { 3, 4, 5, 6 } ; Επίσης, το κόμμα είναι ανεξάρτητος χειριστής.
- Ένας μεμονωμένος χαρακτήρας περικλείεται σε μεμονωμένα εισαγωγικά 'και'
- Η συμβολοσειρά και η σειρά χαρακτήρων περικλείονται σε διπλά εισαγωγικά "γραμμή"
- Τα μεταβλητά ονόματα μπορούν να περιέχουν κεφαλαία και πεζά λατινικά γράμματα (κεφαλαία και πεζά), αριθμούς και υπογράμμιση . Παράδειγμα: myVal_35
- Τα μεταβλητά ονόματα δεν μπορούν να ξεκινούν με ένα ψηφίο . Μόνο με γράμμα ή υπογράμμιση
- Η υπόθεση έχει σημασία, δηλαδή το κεφαλαίο γράμμα είναι διαφορετικό από το μικρό γράμμα. Παράδειγμα: μεταβλητές X και χ - όχι το ίδιο πράγμα.

Η σύνταξη περιλαμβάνει επίσης σχόλια, καθώς ξεχωρίζουν διαφορετικά σε διαφορετικές γλώσσες. Ένα σχόλιο είναι απλό κείμενο που αγνοείται κατά τη στιγμή της μεταγλώττισης. Απαιτούνται σχόλια για να εξηγήσετε τον κώδικα, τόσο στον εαυτό σας όσο και σε άλλους πιθανούς αναγνώστες. Στο C ++, έχουμε δύο τύπους σχολίων:

- **Σχόλιο μίας γραμμής**

```
// ένα σχόλιο γραμμής
// ο μεταγλωττιστής με αγνοεί
```

- **Σχόλιο πολλαπλών γραμμών**

```
/ * Πολλαπλή
    σχόλιο * /
```

Ένα τυπικό πρόγραμμα Arduino έχει την παρακάτω δομή:

```
//δήλωση μεταβλητών
void setup ()
{
//αρχικοποιήσεις
}
void loop ()
{
//Κώδικας
```

```
}
```

Υπάρχουν δυο ειδικές συναρτήσεις που είναι μέρος του κάθε sketch του Arduino οι οποίες είναι η setup() και η loop(). Η setup() καλείται μια φορά, όταν το sketch ξεκινά ή όποτε κάνει επαναφορά (reset) η πλατφόρμα Arduino. Κυρίως, σε αυτήν γίνονται οι αρχικοποιήσεις των μεταβλητών, η ρύθμιση της κατάστασης των ακίδων (pins) και η προετοιμασία των βιβλιοθηκών. Αντιθέτως, η συνάρτηση loop() καλείται ξανά και ξανά επιτρέποντας έτσι στο πρόγραμμα να ανταποκριθεί σε εξωτερικά ερεθίσματα. Και οι δυο συναρτήσεις πρέπει να περιλαμβάνονται στο sketch, ακόμα και αν δεν περιέχουν κάτι και να είναι κενές.

3.4.1 Σύνδεση βιβλιοθηκών και αρχείων

Στην εργασία, χρησιμοποιούνται βιβλιοθήκες ή απλά εξωτερικά αρχεία, συνδέονται με το κύριο αρχείο (αρχείο υλικολογισμικού) χρησιμοποιώντας την οδηγία

#include

Αυτή η οδηγία λέει στον προεπεξεργαστή να βρει και να συμπεριλάβει το καθορισμένο αρχείο στη συλλογή. Το καθορισμένο αρχείο μπορεί να τραβήξει άλλα αρχεία μαζί του, αλλά όλα είναι ήδη γραμμένα εκεί και συνδέονται αυτόματα.

#include <Servo.h> // περιλαμβάνει τη βιβλιοθήκη Servo.h

#include "Servo.h" // περιλαμβάνει επίσης τη βιβλιοθήκη Servo.h

Όταν υποδεικνύουμε το όνομα "σε εισαγωγικά",ο μεταγλωττιστής αναζητά πρώτα το αρχείο στο φάκελο σκίτσων και στη συνέχεια στο φάκελο βιβλιοθηκών. Χρησιμοποιώντας < έλεγχος > ο μεταγλωττιστής αναζητά μόνο το αρχείο στο φάκελο της βιβλιοθήκης.

Μιλώντας για φακέλους με βιβλιοθήκες: υπάρχουν δύο από αυτούς, και οι δύο θα αναζητηθούν βιβλιοθήκες.

- Τα έγγραφά μου / Arduino / βιβλιοθήκες
- C: / Αρχεία προγράμματος (x86) / Arduino / βιβλιοθήκες (ή C: / Αρχεία προγράμματος / Arduino / βιβλιοθήκες για Windows 32-bit)

Οι βιβλιοθήκες μεταβαίνουν στον πρώτο φάκελο (σε έγγραφα) όταν συνδέονται χρησιμοποιώντας την εντολή "connect .zip library". Δεν συνιστάται η σύνδεση βιβλιοθηκών με αυτόν τον τρόπο, επειδή η βιβλιοθήκη δεν έρχεται πάντα σε σας στο αρχείο και θα είναι ευκολότερο να την αντιγράψετε χειροκίνητα σε αρχεία προγράμματος. Επίσης, εάν και στους δύο φακέλους υπάρχουν βιβλιοθήκες με το ίδιο όνομα, αυτό θα οδηγήσει σε διένεξη, οπότε απλώς αντιγράφουμε τις βιβλιοθήκες στο φάκελο βιβλιοθηκών στα αρχεία προγράμματος / Arduino.

Σημείωση: ο φάκελος βιβλιοθήκης που βρίσκεται στο C: / Program Files (x86) / Arduino / library πρέπει να περιέχει τα αρχεία και τους φακέλους της βιβλιοθήκης, όχι έναν φάκελο με το ίδιο όνομα με την ίδια τη βιβλιοθήκη. Αυτό θα οδηγήσει σε σφάλμα, ο συλλέκτης δεν μπορεί να βρει τα αρχεία.

3.4.2 Τύποι δεδομένων

Παρακάτω στον πίνκα 3.4.2 υπάρχουν όλοι οι τύποι δεδομένων που θα συναντήσουμε:

‘Ονομα	Τυπικός τύπος	Βάρος	Εύρος	Χαρακτηριστικό
boolean	bool	1 byte	0 ή 1, αληθές ή ψευδές	Δυαδική μεταβλητή Bool στο Arduino χρειάζεται επίσης 1 byte
char	–	1 byte	-128 ... 127	Αποθηκεύει τον αριθμό χαρακτήρων
–	int8_t	1 byte	-128 ... 127	Ακέραιος τύπος
byte	uint8_t	1 byte	0 ... 255	Ακέραιος τύπος
int	int16_t, short	2 byte	-32 768 ... 32 767	Ακέραιος τύπος
unsigned int	uint16_t, word	2 byte	0 ... 65 535	Ακέραιος τύπος
long	int32_t	4 byte	-2 147 483 648 ... 2 147 483 647	Ακέραιος τύπος
unsigned long	uint32_t	4 byte	0 ... 4 294 967 295	Ακέραιος τύπος
float	–	4 byte	-3.4028235E + 38 ... 3.4028235E + 38	Αποθηκεύει αριθμούς κινητής υποδιαστολής (δεκαδικά). Ακρίβεια: 6-7 ψηφία
double	–	4 byte		Για AVR είναι το ίδιο με float .Και έτσι είναι 8 byte σε ένα πιο ενήλικο υλικό
–	int64_t	8 byte	- (2 ^ 64) / 2 ... (2 ^ 64) / 2-1	Πολύ μεγάλοι αριθμοί. Το Standard Serial δεν ξέρει πώς να το εξάγει
–	uint64_t	8 byte	2 ^ 64-1	Πολύ μεγάλοι αριθμοί. Το Standard Serial δεν ξέρει πώς να το εξάγει

Πίνακας 3.4.2: Τύποι δεδομένων

3.4.3 Σταθερές

Αυτό που είναι μια σταθερά είναι ξεκάθαρο από το όνομά του - κάτι, η αξία του οποίου μπορούμε να διαβάσουμε μόνο και δεν μπορούμε να αλλάξουμε. Υπάρχουν δύο τρόποι για να δηλώσουμε μια σταθερά:

- Ακριβώς όπως μια μεταβλητή, που προηγείται του τύπου δεδομένων με τη λέξη **const**.Εάν η τιμή της μεταβλητής δεν θα αλλάξει κατά την εκτέλεση του προγράμματος, συνιστάται να το δηλώσουμε ως σταθερά, αυτό θα επιτρέψει στον μεταγλωττιστή να βελτιστοποιήσει καλύτερα τον κώδικα και στις περισσότερες περιπτώσεις θα είναι λίγο πιο ελαφρύς και ταχύτερος.

  const byte myConst = 10; // δηλώστε μια σταθερά τύπου byte

- Χρησιμοποιώντας μια οδηγία προεπεξεργαστή **#define**, το οποίο κάνει τα εξής:

Στο στάδιο της σύνταξης του κώδικα, ο προεπεξεργαστής αντικαθιστά όλες τις καθορισμένες ακολουθίες χαρακτήρων στο τρέχον έγγραφο (θυμηθείτε ότι οι καρτέλες Arduino IDE είναι ένα έγγραφο) με τις αντίστοιχες τιμές τους. Η σταθερά ορίζεται με **#define** δεν καταλαμβάνει χώρο στη μνήμη RAM , αλλά αποθηκεύεται ως κωδικός προγράμματος στη μνήμη Flash, αυτό είναι το μεγαλύτερο πλεονέκτημα αυτής της μεθόδου.
Σύνταξη: **# define τιμής ονόματος**.
Παράδειγμα:

 # define BTN_PIN 10
 # define DEFAULT_VALUE 3423

Εάν μια συνηθισμένη μεταβλητή δεν αλλάζει πουθενά κατά την εκτέλεση του προγράμματος, ο μεταγλωττιστής μπορεί ανεξάρτητα να την κάνει σταθερή και δεν θα καταλάβει χώρο στη μνήμη RAM, δηλ. θα τοποθετηθεί σε Flash.

3.4.4 Υπό όρους οδηγίες #if #else

Εκτός από την οδηγία **#define** που λέει στον προεπεξεργαστή να αντικαταστήσει το σύνολο χαρακτήρων με ένα σύνολο χαρακτήρων, υπάρχουν επίσης οδηγίες υπό όρους που σας επιτρέπουν να κάνουμε τη λεγόμενη συνθήκη υπό όρους : έχοντας την ίδια λογική με if-else, αυτές οι κατασκευές σάς επιτρέπουν να κάνετε κάποια επιλογή πριν μεταγλωττίσετε το κωδικός ο ίδιος. Ένα εξαιρετικό παράδειγμα είναι ο «πυρήνας» του ίδιου του Arduino - οι περισσότερες από τις λειτουργίες γράφονται με τις ιδιαιτερότητες κάθε επεξεργαστή και πριν από τη σύνταξη του κώδικα, αυτό που αντιστοιχεί στον τρέχοντα επιλεγμένο μικροελεγκτή επιλέγεται από το σύνολο επιλογών για την εφαρμογή του λειτουργία. Με απλά λόγια, η συνθήκη υπό όρους επιτρέπει, σύμφωνα με τις προϋποθέσεις, να συμπεριλάβει ή να αποκλείσει έναν συγκεκριμένο κωδικό από την κύρια συλλογή, δηλαδή Πρώτον, ο

προεπεξεργαστής αναλύει τον κώδικα, κάτι περιλαμβάνεται σε αυτόν, κάτι δεν είναι, και στη συνέχεια πραγματοποιείται συλλογή.

Επίσης, για παράδειγμα, δεν μπορούμε να δηλώσουμε καμία σταθερά ή μακροεντολή μέσω **#define** περισσότερες από μία φορές, αυτό θα οδηγήσει σε σφάλμα. Η συλλογή υπό όρους επιτρέπει την διακλάδωση όπου είναι δυνατόν. Για συλλογή υπό όρους, οι οδηγίες είναι διαθέσιμες σε εμάς **#if, #elif, #else, #endif, #ifdef, #ifndef**

#if - αναλογικό αν σε λογική κατασκευή

#elif - αναλογικό αλλιώς εάν σε λογική κατασκευή

#else - αναλογικό αλλού σε λογική κατασκευή

#endif - μια οδηγία που ολοκληρώνει την υπό όρους κατασκευή

#ifdef - εάν "ορίζεται"

#ifndef - εάν "δεν ορίζεται"

Defined - αυτός ο χειριστής επιστρέφει αληθής εάν η καθορισμένη λέξη "ορίζεται" έως **#define**, και ψευδής- αν όχι. Χρησιμοποιείται για κατασκευές υπό όρους σύνταξης.

Παράδειγμα:

#if (TEST == 1) // εάν ΔΟΚΙΜΗ 1

#define VALUE 10 // ορίστε VALUE ως 10

#elif (TEST == 0) // ΔΟΚΙΜΗ 0

#define VALUE 20 // ορίστε VALUE ως 20

#else // αν όχι

#define VALUE 30 // ορίστε VALUE ως 30

#endif // τέλος της κατάστασης

3.4.5 Ψηφιακό I / O

pinMode ()

Περιγραφή: Διαμορφώνει τον καθορισμένο πείρο ώστε να συμπεριφέρεται είτε ως είσοδος είτε ως έξοδος. Από το Arduino 1.0.1, είναι δυνατή η ενεργοποίηση των εσωτερικών αντιστάσεων pullup με τη λειτουργία INPUT_PULLUP. Επιπλέον, η INPUT λειτουργία απενεργοποιεί ρητά τα εσωτερικά pullups.
Σύνταξη: pinMode(pin, mode)
Παράμετροι: pin: ο αριθμός pin Arduino για να ορίσετε τη λειτουργία του.
Mode: INPUT, OUTPUTH INPUT_PULLUP.
Επιστρέφει: Τίποτα
Προειδοποιήσεις: Οι αναλογικοί ακροδέκτες εισόδου μπορούν να χρησιμοποιηθούν ως ψηφιακοί ακροδέκτες, που αναφέρονται ως A0, A1 κ.λπ.

digitalRead()

Περιγραφή: Διαβάζει την τιμή από ένα συγκεκριμένο ψηφιακό pin, είτε HIGH ή LOW.
Σύνταξη: digitalRead(pin)
Παράμετροι: pin: ο αριθμός pin Arduino που θέλετε να διαβάσετε
Επιστρέφει: HIGH ή LOW
Προειδοποιήσεις: Εάν ο πείρος δεν είναι συνδεδεμένος με τίποτα, digitalRead() μπορεί να επιστρέψει είτε HIGH είτε LOW(και αυτό μπορεί να αλλάξει τυχαία).Οι αναλογικοί ακροδέκτες εισόδου μπορούν να χρησιμοποιηθούν ως ψηφιακοί ακροδέκτες, που αναφέρονται ως A0, A1 κ.λπ. Η εξαίρεση είναι οι ακίδες Arduino Nano,A6 και A7, οι οποίες μπορούν να χρησιμοποιηθούν μόνο ως αναλογικές είσοδοι.

digitalWrite()

Περιγραφή:Γράψτε ένα HIGH ή μια LOW τιμή σε ένα ψηφιακό pushpin.Εάν ο πείρος έχει διαμορφωθεί ως OUTPUTμε pinMode(), η τάση του θα ρυθμιστεί στην αντίστοιχη τιμή: 5V (ή 3.3V σε 3.3V πίνακες) για HIGH, 0V (γείωση) για LOW.Εάν ο πείρος έχει διαμορφωθεί ως INPUT, digitalWrite()θα ενεργοποιήσει (HIGH) ή θα απενεργοποιήσει (LOW) το εσωτερικό pullup στον πείρο εισόδου. Συνιστάται να ρυθμίσετε το pinMode() για INPUT_PULLUP να ενεργοποιήσετε την εσωτερική αντίσταση έλξης. Αν δεν ορίσετε το pinMode() σε OUTPUT, και συνδέστε μια λυχνία σε ένα pin, κατά την κλήση digitalWrite(HIGH), η λυχνία LED μπορεί να φαίνεται αμυδρά. Χωρίς ρητή ρύθμιση pinMode(), digitalWrite() θα έχει επιτρέψει την εσωτερική αντίσταση pull-up, η οποία λειτουργεί σαν μια μεγάλη αντίσταση περιορισμού ρεύματος.

Σύνταξη: digitalWrite(pin, value)

Παράμετροι: pin: ο αριθμός pin Arduino.

value: HIGHή LOW.

Επιστρέφει: Τίποτα

Προειδοποιήσεις: Οι αναλογικοί ακροδέκτες εισόδου μπορούν να χρησιμοποιηθούν ως ψηφιακοί ακροδέκτες, που αναφέρονται ως A0, A1 κ.λπ. Η εξαίρεση είναι οι ακίδες Arduino Nano,A6 και A7, οι οποίες μπορούν να χρησιμοποιηθούν μόνο ως αναλογικές είσοδοι.

3.4.6 Αναλογικό I / O

analogReference ()

Περιγραφή: Διαμορφώνει την τάση αναφοράς που χρησιμοποιείται για την αναλογική είσοδο (δηλαδή την τιμή που χρησιμοποιείται ως το πάνω μέρος του εύρους εισόδου). Οι επιλογές είναι:

Arduino AVR Boards (Uno, Mega, Leonardo κ.λπ.)

- DEFAULT: η προεπιλεγμένη αναλογική αναφορά 5 βολτ (σε πίνακες Arduino 5V) ή 3,3 βολτ (σε πλακέτες Arduino 3.3V)
- INTERNAL: μια ενσωματωμένη αναφορά, ίση με 1,1 βολτ στα ATmega168 ή ATmega328P και 2,56 βολτ στα ATmega32U4 και ATmega8 (δεν διατίθεται στο Arduino Mega)
- INTERNAL1V1: ενσωματωμένη αναφορά 1.1V (μόνο Arduino Mega)
- INTERNAL2V56: ενσωματωμένη αναφορά 2.56V (μόνο Arduino Mega)
- EXTERNAL: η τάση που εφαρμόζεται στον πείρο AREF (μόνο 0 έως 5V) χρησιμοποιείται ως αναφορά.

Πίνακες Arduino SAMD (Μηδέν κ.λπ.)

- AR_DEFAULT: η προεπιλεγμένη αναλογική αναφορά 3.3V
- AR_INTERNAL: μια ενσωματωμένη αναφορά 2.23V
- AR_INTERNAL1V0: ενσωματωμένη αναφορά 1.0V
- AR_INTERNAL1V65: ενσωματωμένη αναφορά 1,65V
- AR_INTERNAL2V23: μια ενσωματωμένη αναφορά 2.23V
- AR_EXTERNAL: η τάση που εφαρμόζεται στον πείρο AREF χρησιμοποιείται ως αναφορά

Πίνακες Arduino megaAVR (Uno WiFi Rev2)

- DEFAULT: μια ενσωματωμένη αναφορά 0,55V
- INTERNAL: ενσωματωμένη αναφορά 0,55V
- VDD: Vdd του ATmega4809. 5V στο Uno WiFi Rev2
- INTERNAL0V55: ενσωματωμένη αναφορά 0,55V
- INTERNAL1V1: μια ενσωματωμένη αναφορά 1.1V
- INTERNAL1V5: ενσωματωμένη αναφορά 1.5V
- INTERNAL2V5: ενσωματωμένη αναφορά 2.5V
- INTERNAL4V3: ενσωματωμένη αναφορά 4.3V
- ΕΞΩΤΕΡΙΚΗ: η τάση που εφαρμόζεται στον πείρο AREF (μόνο 0 έως 5V) χρησιμοποιείται ως αναφορά

Πίνακες SAM Arduino (Προθεσμία)

- AR_DEFAULT: η προεπιλεγμένη αναλογική αναφορά 3.3V. Αυτή είναι η μόνη υποστηριζόμενη επιλογή για το οφειλόμενο.

Πίνακες Arduino mbed-enabled (Nano 33 BLE only): διαθέσιμος όταν χρησιμοποιείτε την πλατφόρμα *Arduino mbed-enabled Boards* ή την πλατφόρμα *Arduino nRF528x Boards (Mbed OS)* έκδοση 1.1.6 ή μεταγενέστερη

- AR_VDD: η προεπιλεγμένη αναφορά 3.3 V
- AR_INTERNAL: ενσωματωμένη αναφορά 0,6 V
- AR_INTERNAL1V2: Αναφορά 1,2 V (εσωτερική αναφορά 0,6 V με κέρδος 2x)
- AR_INTERNAL2V4: Αναφορά 2,4 V (εσωτερική αναφορά 0,6 V με κέρδος 4x)

Σύνταξη: analogReference(type)
Παράμετροι: type: ποιος τύπος αναφοράς θα χρησιμοποιηθεί
Επιστρέφει: Τίποτα
Προειδοποιήσεις:Μετά την αλλαγή της αναλογικής αναφοράς, οι πρώτες λίγες μετρήσεις analogRead()ενδέχεται να μην είναι ακριβείς.Μην χρησιμοποιείτε τίποτα μικρότερο από 0V ή περισσότερο από 5V για εξωτερική τάση αναφοράς στον πείρο AREF!
Εάν χρησιμοποιείτε μια εξωτερική αναφορά στον πείρο AREF, πρέπει να ορίσετε την αναλογική αναφορά σε EXTERNAL πριν καλέσετε analogRead(). Διαφορετικά, θα συντομεύσετε την ενεργή τάση αναφοράς (που δημιουργείται εσωτερικά) και τον πείρο AREF, πιθανόν να καταστρέψει τον μικροελεγκτή στην πλακέτα Arduino.Εναλλακτικά, μπορείτε να συνδέσετε την εξωτερική τάση αναφοράς στον πείρο AREF μέσω μιας αντίστασης 5K, επιτρέποντάς σας να εναλλάσσετε μεταξύ εξωτερικών και εσωτερικών τάσεων αναφοράς.Η αντίσταση θα αλλάξει την τάση που χρησιμοποιείται ως αναφορά, επειδή υπάρχει μια εσωτερική αντίσταση 32K στον πείρο AREF. Τα δύο δρουν ως διαιρέτης τάσης, έτσι, για παράδειγμα, 2.5V που εφαρμόζονται μέσω της αντίστασης θα αποδώσουν 2,5 * 32 / (32 + 5) = ~ 2.2V στον πείρο AREF.

analogRead ()

Περιγραφή: Διαβάζει την τιμή από τον καθορισμένο αναλογικό πείρο. Οι πίνακες Arduino περιέχουν έναν πολυκάναλο, αναλογικό προς ψηφιακό μετατροπέα 10-bit. Αυτό σημαίνει ότι θα αντιστοιχίσει τις τάσεις εισόδου μεταξύ 0 και την τάση λειτουργίας (5V ή 3,3V) σε ακέραιες τιμές μεταξύ 0 και 1023. Σε ένα Arduino UNO, για παράδειγμα, αυτό δίνει ανάλυση μεταξύ των μετρήσεων: 5 βολτ / 1024 μονάδες ή 4,9 mV ανά μονάδα. Το εύρος εισόδου μπορεί να αλλάξει χρησιμοποιώντας το **analogReference ()**.Σε πίνακες με βάση το ATmega (UNO, Nano, Mini, Mega), χρειάζονται περίπου 100 μικροδευτερόλεπτα (0,0001 s) για να διαβάσετε μια αναλογική είσοδο, οπότε ο μέγιστος ρυθμός ανάγνωσης είναι περίπου 10.000 φορές το δευτερόλεπτο.

Σύνταξη: analogRead(pin)
Παράμετροι: pin: το όνομα του αναλογικού πείρου εισόδου για ανάγνωση (A0 έως A7 στο Nano).
Επιστρέφει: Η αναλογική ανάγνωση στον πείρο. Αν και περιορίζεται στην ανάλυση του αναλογικού σε ψηφιακό μετατροπέα (0-1023 για 10 bit ή 0-4095 για 12 bit). Τύπος δεδομένων: int.
Προειδοποιήσεις: Εάν ο αναλογικός πείρος εισόδου δεν είναι συνδεδεμένος με τίποτα, η τιμή που επιστρέφεται analogRead()θα κυμαίνεται με βάση διάφορους παράγοντες (π.χ. τις τιμές των άλλων αναλογικών εισόδων, πόσο κοντά είναι το χέρι σας στο ταμπλό κ.λπ.).

analogWrite ()

Περιγραφή: Γράφει μια αναλογική τιμή σε έναν πείρο. Μπορεί να χρησιμοποιηθεί για να ανάψει ένα LED σε διάφορες φωτεινότητες ή να κινηθεί ένας κινητήρας σε διάφορες ταχύτητες. Μετά από μια κλήση προς analogWrite(), ο πείρος θα δημιουργήσει ένα σταθερό

ορθογώνιο κύμα του καθορισμένου κύκλου λειτουργίας μέχρι την επόμενη κλήση προς analogWrite()(ή μια κλήση digitalRead()ή digitalWrite()) στον ίδιο πείρο. Δεν χρειάζεται να καλέσουμε pinMode()για να ορίσετε το pin ως έξοδο πριν από την κλήση analogWrite().Η analogWritesυνάρτηση δεν έχει καμία σχέση με τους αναλογικούς πείρους ή τη analogReadσυνάρτηση.
Σύνταξη: analogWrite(pin, value)
Παράμετροι: pin: η καρφίτσα Arduino για να γράψετε. Επιτρέπονται τύπους δεδομένων: int. value: ο κύκλος λειτουργίας: μεταξύ 0 (πάντα απενεργοποιημένο) και 255 (πάντα ενεργοποιημένο). Επιτρέπονται τύπους δεδομένων: int.
Επιστρέφει: Τίποτα
Προειδοποιήσεις: Οι έξοδοι PWM που δημιουργούνται στις ακίδες 5 και 6 θα έχουν υψηλότερους από τον αναμενόμενο κύκλους λειτουργίας. Αυτό οφείλεται στις αλληλεπιδράσεις με τις millis()και delay()λειτουργίες, οι οποίες μοιράζονται τον ίδιο εσωτερικό χρονοδιακόπτη που χρησιμοποιείται για τη δημιουργία αυτών των εξόδων PWM. Αυτό θα παρατηρηθεί κυρίως σε ρυθμίσεις χαμηλού κύκλου λειτουργίας (π.χ. 0 - 10) και ενδέχεται να έχει ως αποτέλεσμα η τιμή 0 να μην απενεργοποιήσει πλήρως την έξοδο στις ακίδες 5 και 6.

3.4.7 Δομή ελέγχου

Παρακάτω στον πίνακα 3.4.7 αναφέρερω τις δομές ελέγχου και την πράξη της καθέ μίας:

Εντολή	Περιγραφή	Παράδειγμα

if	Η δήλωση ελέγχει μια συνθήκη και εκτελεί την ακόλουθη δήλωση ή ένα σύνολο δηλώσεων εάν η συνθήκη είναι «αληθινή».	if (condition) {
		//statement(s)
		}

while	Ένας while βρόχος θα βγει συνεχώς και απεριόριστα, έως ότου η έκφραση μέσα στην παρένθεση, () γίνει ψευδής.	while (condition) {
		// statement(s)
		}

for	Η forδήλωση χρησιμοποιείται για την επανάληψη ενός μπλοκ δηλώσεων που περικλείονται σε αγκύλες.	for (initialization; condition; increment) {
		// statement(s)
		}

do…while	Ο do…while βρόχος λειτουργεί με τον ίδιο τρόπο όπως ο while βρόχος, με την εξαίρεση ότι η συνθήκη δοκιμάζεται στο τέλος του βρόχου, οπότε ο βρόχος do θα τρέχει πάντα τουλάχιστον μία	do {
		// statement block
		} while (condition);

	φορά.	

else	Το if…elseεπιτρέπει μεγαλύτερο έλεγχο επί της ροής του κώδικα από το βασικό if δήλωση, επιτρέποντας πολλαπλές δοκιμές που πρέπει να ομαδοποιηθούν. Μια elseρήτρα (εάν υπάρχει καθόλου) θα εκτελεστεί εάν η συνθήκη στη ifδήλωση έχει ως αποτέλεσμα false. Το else μπορεί να προχωρήσει σε άλλη if δοκιμή, ώστε να μπορούν να εκτελούνται ταυτόχρονα πολλαπλές, αμοιβαία αποκλειστικές δοκιμές.	if (condition1) {
		// do Thing A
		}
		else if (condition2) {
		// do Thing B
		}
		else {
		// do Thing C
		}

goto	Μεταφέρει τη ροή προγράμματος σε ένα επισημασμένο σημείο του προγράμματος	label: goto label; // sends program flow to the label

return	Τερματίζει μία συνάρτηση και επιστρέφει μια τιμή από μια συνάρτηση στη λειτουργία κλήσης	int checkSensor() {
		if (analogRead(0) > 400) {
		return 1;
		}
		else {
		return 0;
		}
		}

continue	Η δήλωση παραλείπει το υπόλοιπο της τρέχουσα επανάληψη ενός βρόχου (for, while, ήdo….while). Συνεχίζει ελέγχοντας την υπό όρους έκφραση του βρόχου και προχωρώντας σε τυχόν επακόλουθες επαναλήψεις.	for (int x = 0; x <= 255; x ++) {
		if (x > 40 && x < 120) { // create jump in values
		continue;
		}
		
		analogWrite(PWMpin, x);
		delay(50);

		}

Switch…..case	Όπως και if δηλώσεις, το switch case ελέγχει τη ροή προγραμμάτων επιτρέποντας στους προγραμματιστές να καθορίσουν διαφορετικό κώδικα που θα πρέπει να εκτελεστεί σε διάφορες συνθήκες. Συγκεκριμένα, μια δήλωση μεταγωγής συγκρίνει την τιμή μιας μεταβλητής με τις τιμές που καθορίζονται στις δηλώσεις περιπτώσεων. Όταν βρεθεί μια δήλωση περίπτωσης της οποίας η τιμή ταιριάζει με αυτήν της μεταβλητής, εκτελείται ο κώδικας σε αυτήν τη δήλωση περίπτωσης.
	Η λέξη κλειδί break κλείνει τη δήλωση εναλλαγής και χρησιμοποιείται συνήθως στο τέλος κάθε περίπτωσης. Χωρίς δήλωση διακοπής, η δήλωση διακόπτη θα συνεχίσει να εκτελεί τις ακόλουθες εκφράσεις ("fall-through") έως ότου διακοπεί ή φτάσει στο τέλος της δήλωσης διακόπτη.
		switch (var) {
		case label1:
		// statements
		break;
		case label2:
		// statements
		break;
		default:
		// statements
		break;
		}

Πίνακας 3.4.7: Δόμή ελέγχου

3.4.8 Τελεστές Bitwise

Ακολουθεί ο πίνακας 3.4.8 που εμφανίζονται οι τελεστές Bitwise:

Τελεστής	Περιγραφή	Παράδειγμα

AND(&)	Χρησιμοποιείται μεταξύ δύο άλλων ακέραιων εκφράσεων. Το Bitwise AND λειτουργεί ανεξάρτητα σε κάθε θέση bit των γύρω εκφράσεων, σύμφωνα με αυτόν τον κανόνα: εάν και τα δύο bit εισόδου είναι 1, η έξοδος που προκύπτει είναι 1, διαφορετικά η έξοδος είναι 0. Στο Arduino, ο τύπος int είναι τιμή 16 bit, οπότε η χρήση & μεταξύ δύο εκφράσεων int προκαλεί 16	int a = 92; // in binary: 0000000001011100
		int b = 101; // in binary: 0000000001100101
		int c = a & b; // result: 0000000001000100, ή 68 σε δεκαδικό.

	ταυτόχρονες λειτουργίες AND.	

<<	Αναγκάζει τα bit του αριστερού τελεστή να μετατοπιστούν αριστερά από τον αριθμό των θέσεων που καθορίζονται από το δεξί τελεστή.	int a = 5; // binary: 0000000000000101
		int b = a << 3; // binary: 0000000000101000, ή 40 σε δεκαδικό

>>	Προκαλεί τη μετατόπιση των δυαδικών ψηφίων του αριστερού τελεστή προς τα δεξιά με τον αριθμό των θέσεων που καθορίζονται από το δεξί τελεστή.	int a = 40; // binary: 0000000000101000
		int b = a >> 3; // binary: 0000000000000101, ή 5 σε δεκαδικό

XOR(^)	Μια λειτουργία bitor XOR έχει ως αποτέλεσμα 1 μόνο εάν τα bit εισαγωγής είναι διαφορετικά, αλλιώς έχει ως αποτέλεσμα 0.	int x = 12; // binary: 1100
		int y = 10; // binary: 1010
		int z = x ^ y; // binary: 0110, ή σε δεκαδικό 6

OR(|)	Το bitwise OR των δύο bit είναι 1 εάν ένα ή και τα δύο bit εισαγωγής είναι 1, διαφορετικά είναι 0.	int a = 92; // in binary: 0000000001011100
		int b = 101; // in binary: 0000000001100101
		int c = a | b; // result: 0000000001111101, ή σε δεκαδικό 125.

NOT(~)	Αλλάζει κάθε bit στο αντίθετό του: 0 γίνεται 1 και 1 γίνεται 0.	int a = 103; // binary: 0000000001100111
		int b = ~a; // binary: 1111111110011000 = -104

Πίνακας 3.4.8: Τελεστές Bitwise

3.4.9 Boolean χειριστές

Τέλος αναφέρω τους Boolen χειριστές (3.4.9):

Χειριστής	Περιγραφή	Παράδειγμα

Λογικό ΟΧΙ (!)	Δεν έχει ως αποτέλεσμα true εάν ο τελεστής είναι false και αντίστροφα.	if (!x) { // if x is not true
		// statements
		}

Λογική AND (&&)	Οδηγεί σε true μόνο εάν και οι δύο τελεστές είναι true.	if (digitalRead(2) == HIGH && digitalRead(3) == HIGH) { // if BOTH the switches read HIGH
		// statements
		}

Λογικό 'Η (||)	Έχει ως αποτέλεσμα true εάν ένας από τους δύο τελεστές είναι true.	if (x > 0 || y > 0) { // if either x or y is greater than zero
		// statements
		}

Πίνακας 3.4.9: Boolean χειριστές

3.4.10 Επίλογος

Τέλος όπως αναφέρεται σε αυτό το κεφάλαιο βλέπουμε ότι υπάρχει μία πληθώρα εντολών που μας επιτρέπει να δημιουργήσουμε ότι θέλουμε μέσα στο Arduino.Το κεφάλαιο 3 είναι αφιερωμένο στις γλώσσες προγραμματισμού και ειδικότερας της C++ που χρησιμοποιείται στην συγκρεκριμένη περίπτωση.Επίσης σε αυτό το κεφάλαιο είδαμε την απλοικότητα του προγράμματος Arduino IDE με μία τεράστια γκάμα εργαλειών.

Κεφάλαιο 4ο: Περιγραφή γεννήτριας συχνοτήτων και παλμογράφου

4.1 Εισαγωγή

Πριν αρχίσουμε να αναλύουμε βήμα την κατασκευή της γεννήτριας συχνοτήτων με την χρήση του μικροελεγκτή Arduino είναι αναγκαίο να αναφέρουμε την ιδιότητα μιας γεννήτριας συχνοτήτων καθώς και την ιδιότητα του παλμογράφου που θα μας βοηθήσει να απεικονίσουμε τα σήματα που θα παράγουμε μέσω της γεννήτριας.

4.2 Γεννήτρια συχνοτήτων

Η βασική κυματομορφή που μπορεί να παράγει μια γεννήτρια σήματος είναι η ημιτονοειδής, ενώ έχει τη δυνατότητα να παράγει κι άλλες επαναλαμβανόμενες κυματομορφές όπως πριονωτή, τριγωνική και τετραγωνική. Μια γεννήτρια σήματος, μπορεί να είναι τύπου υψηλών συχνοτήτων RF ή ακουστικών συχνοτήτων.Μια δυνατότητα που περιλαμβάνουν πολλές γεννήτριες είναι η ρύθμιση DC offset (μέση τιμή σήματος). Μια απλή γεννήτρια σήματος μπορεί να παράγει κυματομορφές με συχνότητες μέχρι 100kHz, ενώ πιο ακριβά μοντέλα μπορούν να παράγουν έως και 20MHz ή και ακόμα 40MHz.Σε κάθε γεννήτρια υπάρχουν δυο ακροδέκτες για τη λήψη του σήματος. Ο ένας έχει το σύμβολο της γείωσης και πρέπει να συνδέεται πάντα (εκτός μερικών περιπτώσεων) στο κοινό σημείο του κυκλώματος μέσω ενός μαύρου καλωδίου. Ο άλλος ακροδέκτης συνδέεται πάντα μέσω ενός κόκκινου καλωδίου στο σημείο του κυκλώματος που θα δώσουμε το σήμα.

Ρυθμίσεις στην γεννήτρια σήματος

Παρακάτω στο σχήμα 4.2 παρουσιάζεται η μορφή μιας γεννήτριας συχνοτήτων πάγκου.Επιπρόσθετα από την επιλογή της κυματομορφής, σε μια γεννήτρια σήματος μπορούμε να επιλέξουμε:

1) **Τη συχνότητα:** Με την επιλογή της συχνότητας στη γεννήτρια σήματος, μπορούμε να επιλέξουμε τη συχνότητα με την οποία μια κυματομορφή επαναλαμβάνεται.
2) **Τύπος κυματομορφής:** Με αυτή την επιλογή ορίζουμε την κυματομορφή εξόδου: ημιτονοειδής, τετραγωνική, τριγωνική, πριονωτή.
3) **DC offset**: Με την επιλογή αυτή ορίζουμε τη μέση τιμή της κυματομορφής ως προς την τάση μηδέν.
4) **Duty cycle:** Με την επιλογή αυτή ορίζουμε το λόγο του σήματος που έχει τιμή “high” προς την περίοδο του σήματος για μια παλμική κυματομορφή.

Σχήμα 4.2: Γεννήτρια συχνοτήτων

4.3 Παλμογράφος

Ο παλμογράφος επιτρέπει την απεικόνιση κυματομορφών σε δυο διαστάσεις. Στον κατακόρυφο άξονα απεικονίζεται το πλάτος του σήματος και στον οριζόντιο άξονα ο χρόνος. Δηλαδή με τον παλμογράφο μπορούμε να δούμε την γραφική παράσταση του πλάτους τάσης του σήματος σε συνάρτηση με τον χρόνο.Οι περισσότεροι παλμογράφοι είναι dual channel ή multi-channel δηλαδή επιτρέπουν την ταυτόχρονη απεικόνιση δυο ή περισσότερων κυματομορφών σε κοινό άξονα των χρόνων.Ο παλμογράφος είναι ένα ιδιαίτερα χρήσιμο ηλεκτρονικό όργανο που χρησιμοποιείται για την εύρεση βλαβών από τα αναλογικά κυκλώματα ακουστικών συχνοτήτων έως και τις συσκευές ραδιοεπικοινωνίας RF.Παρακάτω φαίνονται οι 2 τύποι παλμογράφων αναλογικός (4.3) και ψηφιακός (4.3.1).

Σχήμα3.1.2: Αναλογικός παλμογράφος

Σχήμα3.1.2: Ψηφιακός παλμογράφος

Κεφάλαιο 5ο : Ο κώδικας και η κατασκεύη

5.1 Εισαγωγή

Σε αυτό το κεφάλαιο θα αναλύσουμε και θα δούμε τον κώδικα όπως επίσης και την κατασκεύη από την αρχή της έως το τέλος της βήμα βήμα.Όλα αυτά που έχουν αναφερθεί στα προηγούμμενα κεφάλαια έχουν συμπεριληφθεί τόσο στον κώδικα όσο και στην κατασκεύη.

5.2 Ο κώδικας του προγράμματος

```
#include <Adafruit_GFX.h>
#include <Adafruit_PCD8544.h>
#include <FreqCount.h>
#include <PWM.h>
#include <CyberLib.h>
#define led  9   //PIN γεννήτρια σημάτων (no change)
#define dds  10   //PIN για γεννήτρια dds (no change)
//######################custom settings##################
#define power 8 //PIN which polls button
#define OFF 14//PIN that controls key supply
#define  timepowerON 50 //off button hold time
#define levo 13  //LEFT button (can be any PIN)
#define ok 12    //OK button (can be any PIN)
#define pravo 11 //RIGHT button (can be any PIN)
#define akb A0 // οποιαδήποτε μέτρηση αναλογικής τάσης  svojuodnyj pin CRA
#define overclock 16  // Η συχνότητα στην οποία λειτουργεί 1478
Adafruit_PCD8544 display = Adafruit_PCD8544(7, 6, 4, 3, 2);//pins στην οποία έχετε
συνδέσει οθόνη
byte cont=52;// αντίθεση οθόνης
byte SinU=30;   // Συγχρονισμός level 0 to 255
int PWM = 128;// η αρχική τιμή του PWM from 0 to 255
int32_t frequency = 500; // την αρχική τιμή της συχνότητας σε Hz
float VCC=5.0;  // τάση τροφοδοσίας, μετρώντας πολύμετρο
//#################################################################
int d=0;
byte menuDDS=0;
bytesinM[32]={1,6,15,29,48,69,92,117,143,168,191,212,229,243,251,255,254,248,237,222,20
3,181,156,131,106,81,59,39,22,10,3,1};
bytepilaM[32]={1,9,17,25,33,41,49,57,65,73,81,89,97,105,113,121,129,137,145,153,161,169,
177,185,193,201,209,217,225,235,245,255};
byteRpilaM[32]={250,246,238,230,222,214,206,198,190,182,174,166,158,150,142,134,126,1
18,110,102,94,86,78,70,62,54,41,33,25,17,9,1};
bytetrianglM[32]={1,18,35,52,69,86,103,120,137,154,171,188,205,222,239,255,239,223,207,1
91,175,159,143,127,111,95,79,63,47,31,15,1};
int powerON=0;// κατάσταση κουμπιού λειτουργίας
byte hag=0;
int mnog=0;
boolean flag=0;
```

```
byte mass[701];
byte x=0;
byte menu=0;// μενού επιλογής μεταβλητών
bool opornoe=1; // τάση αναφοράς
bool paus=0; // λειτουργία παύσης
byte pultoskop=0; // η γεννήτρια επιλογής ή ένας παλμογράφος
byte razv=6;
unsigned long count =0;
byte sinX=30;
byte meaX=83;
int Vmax=0;// μέγιστη τάση
byte sinhMASS=0;
long countX=0;
long speedTTL=9600; // Ταχύτητα τερματικού
int prokr=0;
void setup(){
pinMode(A4,INPUT);
digitalWrite(OFF,HIGH);//vkljuchem food
//Serial.begin(9600);
display.begin();
display.setContrast(cont);
while(digitalRead(ok)==LOW){

//////////////////////////////////////// κρατήστε πατημένο το κουμπί απενεργοποίηση
if(digitalRead(power)==HIGH){powerON++;delay(10);}
if(powerON>=timepowerON){digitalWrite(OFF,LOW);}/// Απενεργοποιήστε την
τροφοδοσία//////////////////////////////////////// κρατήστε πατημένο το κουμπί απενεργοποίηση
      if(pultoskop==0){
        display.clearDisplay();
        display.setCursor(10,0);
        display.setTextColor(WHITE, BLACK); // «ανεστραμμένο» κείμενο
        display.println("[OscilloScp]");
        display.setCursor(10,10);
        display.setTextColor(BLACK);
        display.println("[Generator]");
        display.setCursor(10,20);
        display.println("[DDS Gen]");
        display.setCursor(10,30);
        display.println("[Terminal]");
        display.setCursor(0,40);
        display.print("Battery= ");
       display.print(analogRead(akb)*5.0/1024);
       display.print("In the");

}
  if(pultoskop==1){
     display.clearDisplay();
     display.setCursor(10,0);
```

```
    display.setTextColor(BLACK); // «ανεστραμμένο» κείμενο
    display.println("[OscilloScp]");
    display.setCursor(10,10);
    display.setTextColor(WHITE, BLACK); // «ανεστραμμένο» κείμενο
    display.println("[Generator]");
    display.setTextColor(BLACK); // «ανεστραμμένο» κείμενο
    display.setCursor(10,20);
    display.println("[DDS Gen]");
    display.setCursor(10,30);
    display.println("[Terminal]");
    display.setCursor(0,40);
    display.setTextColor(BLACK);
    display.print("Battery= ");
    display.print(analogRead(akb)*5.0/1024);
    display.print("In the");
}
 if(pultoskop==2){
 display.clearDisplay();
 display.setCursor(10,00);
 display.setTextColor(BLACK); // «ανεστραμμένο» κείμενο
 display.println("[OscilloScp]");
 display.setCursor(10,10);
 display.println("[Generator]");
 display.setTextColor(WHITE, BLACK); // «ανεστραμμένο» κείμενο;
 display.setCursor(10,20);
 display.println("[DDS Gen]");
 display.setTextColor(BLACK);
 display.setCursor(10,30);
 display.println("[Terminal]");
 display.setCursor(0,40);
 display.setTextColor(BLACK);
 display.print("Battery= ");
 display.print(analogRead(akb)*5.0/1024);
 display.print("In the");
 }
   if(pultoskop==3){
 display.clearDisplay();
 display.setCursor(10,00);
 display.setTextColor(BLACK); // «ανεστραμμένο» κείμενο
 display.println("[OscilloScp]");
 display.setCursor(10,10);
 display.println("[Generator]");
 display.setTextColor(BLACK);
 display.setCursor(10,20);
 display.println("[DDS Gen]");
 display.setTextColor(WHITE, BLACK);
 display.setCursor(10,30);
 display.println("[Terminal]");
```

```
    display.setCursor(0,40);
    display.setTextColor(BLACK);
    display.print("Battery= ");
    display.print(analogRead(akb)*5.0/1024);
    display.print("In the");
    }
if(digitalRead(levo)==HIGH){delay(300);pultoskop=pultoskop+1;}
if(digitalRead(pravo)==HIGH){delay(300);pultoskop=pultoskop+1;}
if(pultoskop>3){pultoskop=0;}
delay(50);
display.display(); }
if(pultoskop==2){InitTimersSafe(); bool success = SetPinFrequencySafe(led,200000);}
if(pultoskop==0){FreqCount.begin(1000);}
if(pultoskop==1){InitTimersSafe(); bool success = SetPinFrequencySafe(led,
frequency);}
display.setTextColor(BLACK);
delay(500); }

void Zamer(){
  if (razv>=6){ADCSRA = 0b11100010;}//delitel 4
  if (razv==5){ADCSRA = 0b11100011;}//delitel 8
  if (razv==3){ADCSRA = 0b11100101;}//delitel 32
  if (razv==2){ADCSRA = 0b11100110;}//delitel 64
  if (razv<2){ADCSRA = 0b11100111;}//delitel 128
  if (razv==0){
     for(int i=0;i<700;i++){
        while ((ADCSRA & 0x10)==0);
        ADCSRA|=0x10;
        delayMicroseconds(500);
        mass[i]=ADCH;
     }
  }
  if (razv>0){
     for(int i=0;i<700;i++){
        while ((ADCSRA & 0x10)==0);
        ADCSRA|=0x10;
       mass[i]=ADCH;
     }
  }

}
void loop() {

////////////////////////////////////// κρατήστε πατημένο το κουμπί απενεργοποίηση
if(digitalRead(power)==HIGH){powerON++;delay(10);}
if(powerON>=timepowerON){digitalWrite(OFF,LOW);}/// Απενεργοποιήστε την τροφοδοσία
////////////////////////////////////// κρατήστε πατημένο το κουμπί απενεργοποίηση
if(pultoskop==0){
```

```
  if(opornoe==0){ADMUX = 0b11100011;}// Επιλέξτε την εσωτερική αναφορά στο 1.1
  if(opornoe==1){ADMUX = 0b01100011;}// Επιλέξτε εξωτερική αναφορά
  delay(5);
  if(paus==0){Zamer();}
  //######################################## ορισμός σημείων συγχρονισμού
  bool flagSINHRO=0;
  bool flagSINHRnull=0;
  for(int y=1;y<255;y++){
    if(flagSINHRO==0){if(mass[y]<SinU){flagSINHRnull=1;}}

  if(flagSINHRO==0){if(flagSINHRnull==1){if(mass[y]>SinU){flagSINHRO=1;sinhMASS=y;
  }}}
  //########################################  ορισμός σημείων συγχρονισμού
  //the maximum signal value#########################
  Vmax=0;
  for(int y=1;y<255;y++){if(Vmax<mass[y]){Vmax=mass[y];} }
  // τη μέγιστη τιμή σήματος

 ##########################
//######################################## ορισμός σημείων συγχρονισμού
//######################################## απόδοση γραφικών
if(paus==0){
display.clearDisplay();
display.fillCircle(80,47-SinU/7, 2, BLACK);// σχεδιάστε ένα επίπεδο συγχρονισμού
x=3;
for(int y=sinhMASS;y<sinhMASS+80;y++){
     if(razv<7){x++;}
     if(razv==7){x=x+2;}
     if(razv==8){x=x+3;}
     display.drawLine(x, 47-mass[y]/7, x+1, 47-mass[y+1]/7, BLACK);
     display.drawLine(x+1, 47-mass[y]/7+1, x+2, 47-mass[y+1]/7+1, BLACK);
}
sinhMASS=0;}


if(paus==1){
display.clearDisplay();
display.drawLine(prokr/8,8,prokr/8+6,8, BLACK);// κύλιση κλίμακας
display.drawLine(prokr/8,9,prokr/8+6,9, BLACK);// κύλιση κλίμακας
        x=3;
        for(int y=prokr;y<prokr+80;y++){
            if(razv<7){x++;}
            if(razv==7){x=x+2;}
            if(razv==8){x=x+3;}
            display.drawLine(x, 47-mass[y]/7, x+1, 47-mass[y+1]/7, BLACK);
            display.drawLine(x+1, 47-mass[y]/7+1, x+2, 47-mass[y+1]/7+1, BLACK);          }}
//######################################## απόδοση γραφικών
for(byte i=47;i>5;i=i-7){display.drawPixel(0,i, BLACK);display.drawPixel(1,i,
```

```
  BLACK);display.drawPixel(2,i, BLACK);}// κάθετη διάταξη οθόνης
  ////////////////////////////////////////////////grid
  for(byte i=47;i>5;i=i-3){display.drawPixel(21,i, BLACK);display.drawPixel(42,i,
  BLACK);display.drawPixel(63,i, BLACK);}
  for(byte i=3;i<84;i=i+3){display.drawPixel(i,33, BLACK);display.drawPixel(i,19, BLACK);}

////////////////////////////////////////////////// πλέγμα
//######################################### μενού απόδοσης
if(menu==0){
   display.setCursor(0,0);
   display.setTextColor(WHITE,BLACK);
   if(opornoe==0){display.print("1.1");}
   if(opornoe==1){display.print(VCC,1);}
   display.setTextColor(BLACK);
   display.print(" ");
   display.print(razv);
   display.print(" P");
   if(digitalRead(levo)==HIGH){opornoe=!opornoe;}
   if(digitalRead(pravo)==HIGH){opornoe=!opornoe;}
}
if(menu==1){
   display.setCursor(0,0);
   display.setTextColor( BLACK);
   if(opornoe==0){display.print("1.1");}
   if(opornoe==1){display.print(VCC,1);}

   display.setTextColor(WHITE, BLACK); // «ανεστραμμένο» κείμενο
   display.print(" ");
   display.print(razv);
   display.setTextColor( BLACK); // «ανεστραμμένο» κείμενο
   display.print(" P");
   if(digitalRead(levo)==HIGH){razv=razv-1;if(razv==255){razv=0;}}
   if(digitalRead(pravo)==HIGH){razv=razv+1;if(razv==9){razv=8;}}
}
if(menu==2){
   display.setCursor(0,0);
   display.setTextColor( BLACK);
   if(opornoe==0){display.print("1.1");}
if(opornoe==1){display.print(VCC,1);}
   display.print(" ");
   display.print(razv);
   display.setTextColor(WHITE, BLACK); // «ανεστραμμένο» κείμενο
   display.print(" P");
   paus=1;
   if(digitalRead(levo)==HIGH){prokr=prokr-10;if(prokr<0){prokr=0;}}
   if(digitalRead(pravo)==HIGH){prokr=prokr+10;if(prokr>620){prokr=620;}}
}
if(menu==3){
   prokr=0;
```

```
    paus=0;
    display.setCursor(0,0);
    display.setTextColor( BLACK);
    if(opornoe==0){display.print("1.1");}
    if(opornoe==1){display.print(VCC,1);}
    display.print(" ");
    display.print(razv);
    display.setTextColor(BLACK);
    display.print(" P");
    if(digitalRead(levo)==HIGH){SinU=SinU-20;if(SinU<20){SinU=20;}}
  if(digitalRead(pravo)==HIGH){SinU=SinU+20;if(SinU>230){SinU=230;}}
display.fillCircle(80,47-SinU/7, 5, BLACK);
    display.fillCircle(80,47-SinU/7, 2, WHITE);
}
if(digitalRead(ok)==HIGH){menu++;if(menu==4){menu=0;paus=0;}}// επαναλάβετε το μενού
if (FreqCount.available()) { count = FreqCount.read();}//διαθεσιμότητα μετρητή συχνότητας
εξόδου
//########################################  συχνότητα σήματος
byte Frec1=0;
long Frec=0;
bool flagFrec1=0;
bool flagFrec2=0;
bool flagFrec3=0;
for(int y=1;y<255;y++){
  if(flagFrec1==0){if(mass[y]<SinU){flagFrec2=1;}}
  if(flagFrec1==0){if(flagFrec2==1){if(mass[y]>SinU){flagFrec1=1;Frec1=y;}}}
  if(flagFrec1==1){if(mass[y]<SinU){flagFrec3=1;}}
  if(flagFrec3==1){if(mass[y]>SinU){
  if (razv>=6){Frec=1000000/((y-Frec1-1)*3.27);}//delitel 4
  if (razv==5){Frec=1000000/((y-Frec1)*3.27)/2;}//delitel 8
  if (razv==4){Frec=1000000/((y-Frec1)*3.27)/4;}//delitel 16
  if (razv==3){Frec=1000000/((y-Frec1)*3.27)/8;}//delitel 32
  if (razv==2){Frec=1000000/((y-Frec1)*3.27)/16;}//delitel 64
  if (razv==2){Frec=1000000/((y-Frec1)*3.27)/32;}//delitel 128
  if (razv==1){Frec=1000000/((y-Frec1)*3.27)/32;}//delitel 128
  if (razv==0){Frec=1000000/((y-Frec1)*500);}//delitel 128
  flagFrec1=0;flagFrec3=0;}}}
//########################################  συχνότητα σήματος
display.setTextColor( BLACK);
if(opornoe==1){
if((Vmax*VCC/255)>2.5){countX=count*(overclock/16.0);}
if((Vmax*VCC/255)<2.5){countX=Frec*(overclock/16.0);}}
if(opornoe==0){countX=Frec*(overclock/16.0);}
if(countX<1000){display.print(" ");display.print(countX);display.print("Hz");}
if(countX>1000){float
countXK=countX/1000.0;display.print(countXK,1);display.print("KHz");}
if(opornoe==1){display.setCursor(0,40);display.setTextColor(BLACK);
display.print(Vmax*VCC/255,1);}
```

```
if(opornoe==0){display.setCursor(0,40);display.setTextColor(BLACK);
display.print(Vmax*1.1/255,1);}
display.print("V");
//######################################## μενού απόδοσης
delay(200);
display.display();
}
if(pultoskop==1){Generator();}
if(pultoskop==2){DDSGenerator();}
if(pultoskop==3){TTL();}
}
//######################################## λειτουργία ανανέωσης
void Generator(){
display.clearDisplay();
if (flag==0){//flag election mode PWM or frequency
        if(digitalRead(levo)==HIGH){
          frequency=frequency-mnog;
          if(frequency<0){frequency=0;}
bool success = SetPinFrequencySafe(led, frequency);
          delay(3);// προστασία από αναπήδηση
        }
        if(digitalRead(pravo)==HIGH){
          frequency=frequency+mnog;
          bool success = SetPinFrequencySafe(led, frequency);
          delay(3);// προστασία από αναπήδηση
        }
}
if (flag==1){//flag election mode PWM or frequency
        if(digitalRead(levo)==HIGH){
          PWM=PWM-1;
          if(PWM<0){PWM=255;}
          delay(3);// προστασία από αναπήδηση
        }

            if(digitalRead(pravo)==HIGH){
              PWM=PWM+1;
              if(PWM>255){PWM=0;}
             delay(3);// προστασία από αναπήδηση


           }

    }
   if(digitalRead(ok)==HIGH){// επιλογή συχνότητας εναλλαγής επιπέδου
     delay(3);// προστασία από αναπήδηση
   hag++;
    if(hag>=5){hag=0;}
   }
   ////////////
```

```
display.setTextSize(1);
display.setCursor(0,5);
display.print("PWM=");
display.print(PWM*100.0/255);
display.print(" %");
display.drawLine(0,0,83*PWM/255.0,0, BLACK);
display.drawLine(0,1,83*PWM/255.0,1, BLACK);
display.drawLine(0,2,83*PWM/255.0,2, BLACK);
display.drawLine(0,15,83*PWM/255.0,15, BLACK);
display.drawLine(0,16,83*PWM/255.0,16, BLACK);
display.drawLine(0,17,83*PWM/255.0,17, BLACK);
//////////
display.setCursor(5,20);
 display.setTextSize(2);
long frequencyX=frequency*(overclock/16.0);
if(frequencyX<1000){display.print(frequencyX);display.setTextSize(1);display.println("Hz");
}
if(frequencyX>1000){if(frequencyX<10000){display.print((frequencyX/1000.0),2);
display.setTextSize(1);display.println("KHz");}}
if(frequencyX>=10000){if(frequencyX<100000){display.print((frequencyX/1000.0),1);
display.setTextSize(1);display.println("KHz");}}
if(frequencyX>=100000){display.print((frequencyX/1000.0),0);display.setTextSize(1);
display.println("KHz");}
display.setCursor(0,40);
display.setTextSize(1);
display.print(">>X ");
    if(hag==0){// επιλογή πολλαπλασιαστή συχνότητας
            display.print(1*(overclock/16.0),1);
 mnog=1;
       flag=0;
    }
    if(hag==1){// επιλογή πολλαπλασιαστή συχνότητας

display.print (10*(overclock/16.0), 0);
       mnog=10;

     }
    if(hag==2){// επιλογή πολλαπλασιαστή συχνότητας

 display.print(100*(overclock/16.0),0);
  mnog=100;
  }
  if(hag==3){// επιλογή πολλαπλασιαστή συχνότητας
     display.print(1000*(overclock/16.0),0);
     mnog=1000;
  }
  if(hag==4){//choice PWM
     display.print("PWM ");
     display.print(PWM*100.0/255);
```

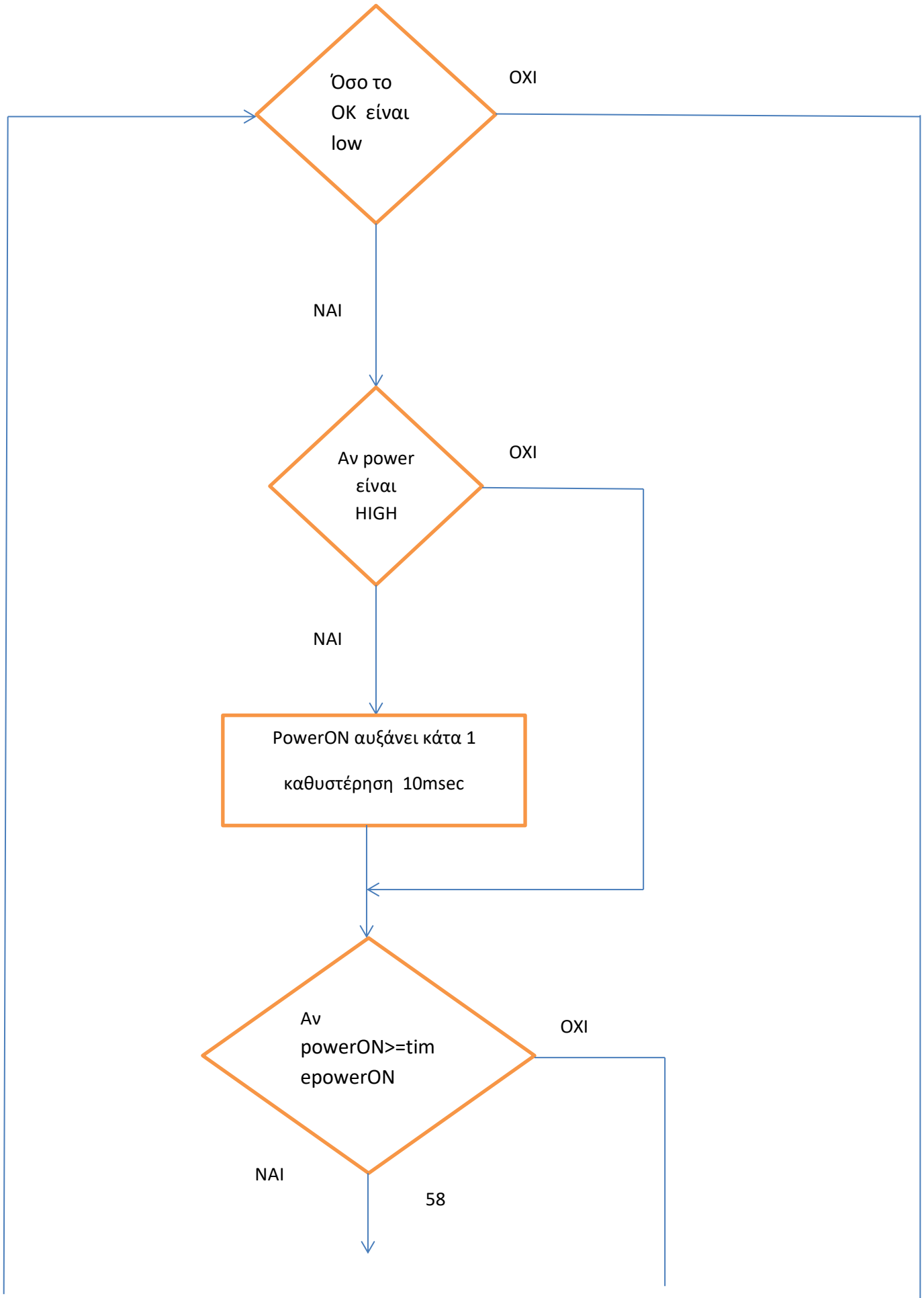
```
        display.print("%");
        flag=1;
}
display.print("<<");
pwmWrite(led, PWM);
delay(300);
display.display();
}
//////////////////////DDS
void DDSGenerator(){
int fr=10;
if(menuDDS==0){
   display.clearDisplay();
   display.setTextColor(WHITE, BLACK); // «ανεστραμμένο» κείμενο
   display.setCursor(10,0);
   display.println("Sine");
   display.setTextColor(BLACK);
   display.setCursor(10,10);
   display.println("Triangle");
   display.setCursor(10,20);
   display.println("Saw");
   display.setCursor(10,30);
   display.println("Saw Arr");
   display.setTextColor(BLACK);
   display.setCursor(0,40);

    //display.print("Frequency=");
   //display.print(57);
   //display.print("Hz");
   delay(100);
   display.display();
   while(D11_Read==LOW){
     PWM=sinM[d];
     pwmWrite(dds,PWM);
     //delayMicroseconds(fr);
     d++;
     if(d==32){d=0;}}
   menuDDS++;
   delay(200);}
if(menuDDS==1){
   display.clearDisplay();
   display.setTextColor(BLACK); // «ανεστραμμένο» κείμενο
   display.setCursor(10,0);
   display.println("Sine");
   display.setTextColor(WHITE, BLACK);
   display.setCursor(10,10);
   display.println("Triangle");
   display.setTextColor(BLACK);
```

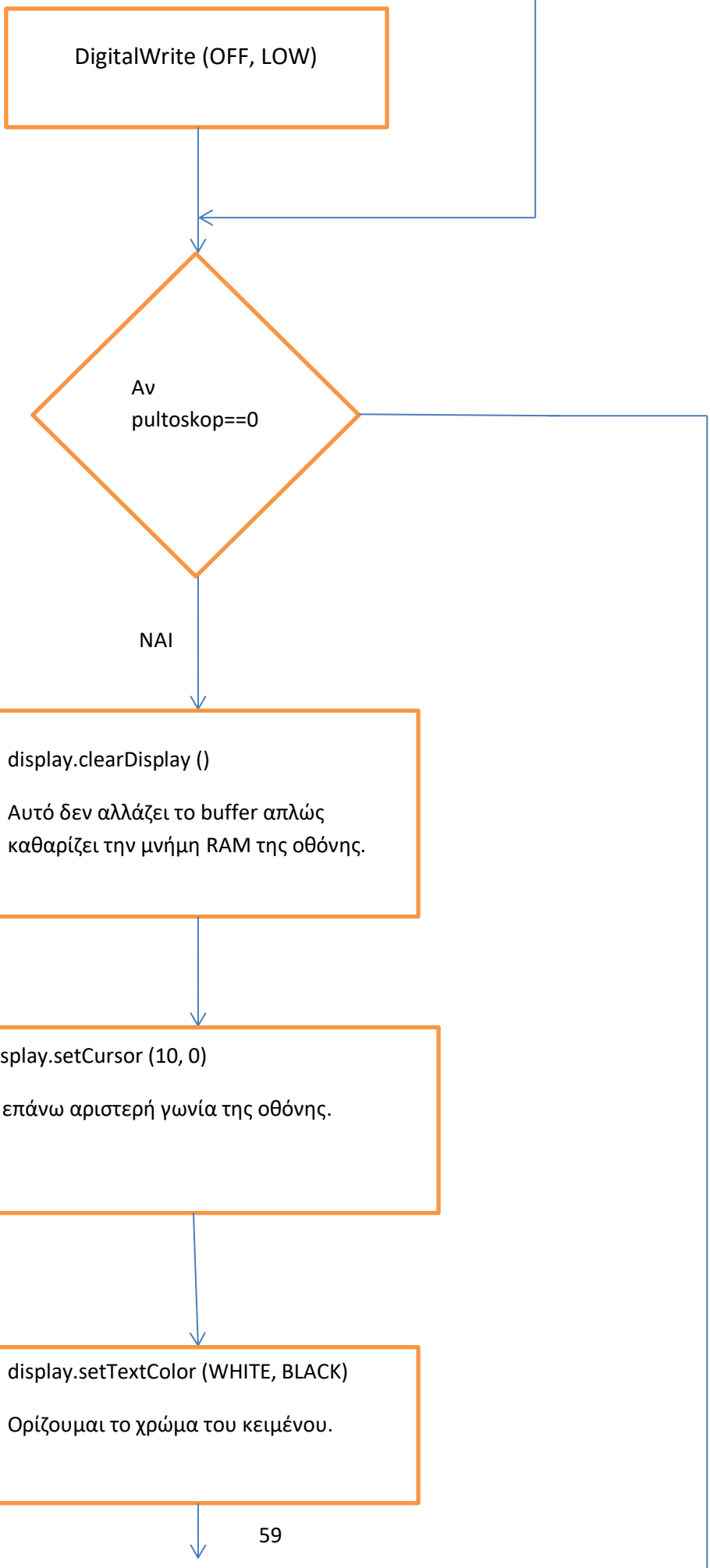
```
    display.setCursor(10,20);
    display.println("Saw");
    display.setCursor(10,30);
    display.println("Saw Arr");
    display.setTextColor(BLACK);
    //display.setCursor(0,40);
    //display.print("Frequency=");
   // display.print(57);
    //display.print("Hz");
    delay(100);
    display.display();
    while(D11_Read==LOW){
    PWM=trianglM[d];
      pwmWrite(dds,PWM);
      //delayMicroseconds(fr);
      d++;
      if(d==32){d=0;}}

    menuDDS++;
    delay(200);}
if(menuDDS==2){
    display.clearDisplay();
    display.setTextColor(BLACK); // «ανεστραμμένο» κείμενο
    display.setCursor(10,0);
    display.println("Sine");
    display.setTextColor(BLACK);
    display.setCursor(10,10);
    display.println("Triangle");
    display.setTextColor(WHITE, BLACK);
    display.setCursor(10,20);
    display.println("Saw");
    display.setTextColor(BLACK);
    display.setCursor(10,30);
    display.println("Saw Arr");
    display.setTextColor(BLACK);
    //display.setCursor(0,40);
    //display.print("Frequency=");
   // display.print(57);
    //display.print("Hz");
    delay(100);
    display.display();
    while(D11_Read==LOW){
      PWM=pilaM[d];
      pwmWrite(dds,PWM);
     // delayMicroseconds(fr);
      d++;
    if(d==32){d=0;}
    menuDDS++;
```

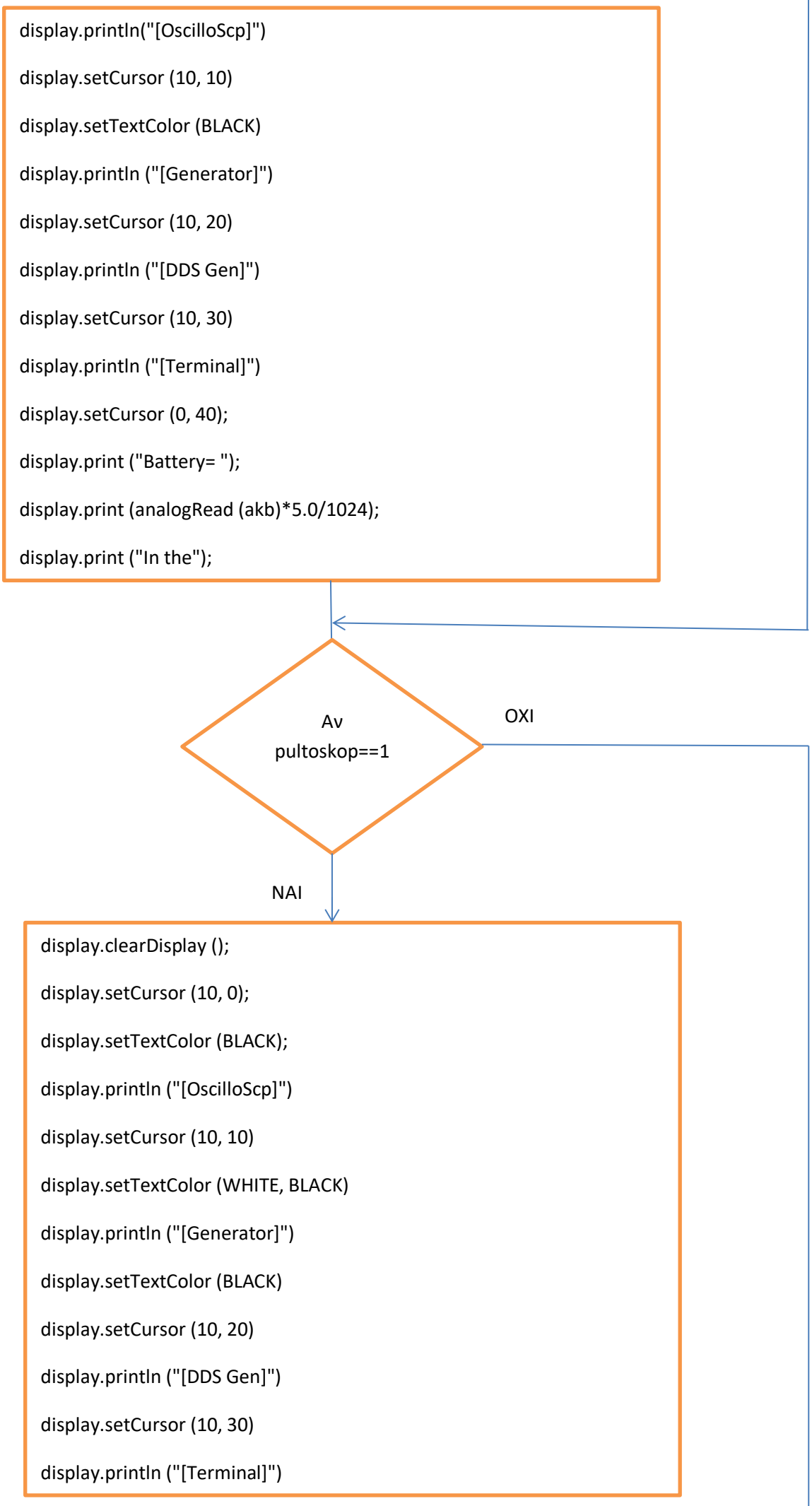
```
    delay(200);}
if(menuDDS==3){
    display.clearDisplay();
    display.setTextColor(BLACK); // «ανεστραμμένο» κείμενο
    display.setCursor(10,0);
   display.println("Sine");
    display.setTextColor(BLACK);
   display.setCursor(10,10);
    display.println("Triangle");
    display.setTextColor(BLACK);
    display.setCursor(10,20);
    display.println("Saw");
    display.setTextColor(WHITE, BLACK);
    display.setCursor(10,30);
    display.println("Saw Arr");
    display.setTextColor(BLACK);
    //display.setCursor(0,40);
    //display.print("Frequency=");
   // display.print(57);
   // display.print("Hz");
  delay(100);
    display.display();
while(D11_Read==LOW){
      PWM=RpilaM[d];
      pwmWrite(dds,PWM);

   //delayMicroseconds(fr);
      d++;
      if(d==32){d=0;}}
      menuDDS++;

        delay(200);}
  if(menuDDS==4){menuDDS=0;}
  }
  ////////////////////DDS
  ////////////////////TTL
  void TTL(){
  display.clearDisplay();
  display.setTextColor(BLACK);
  display.setCursor(10,0);
  display.println("Terminal");
  display.setCursor(10,10);
  display.println("Speed");
  display.setCursor(10,20);
  display.print("-");
  display.print(speedTTL);
  display.println("+");
  display.setCursor(0,30);
```

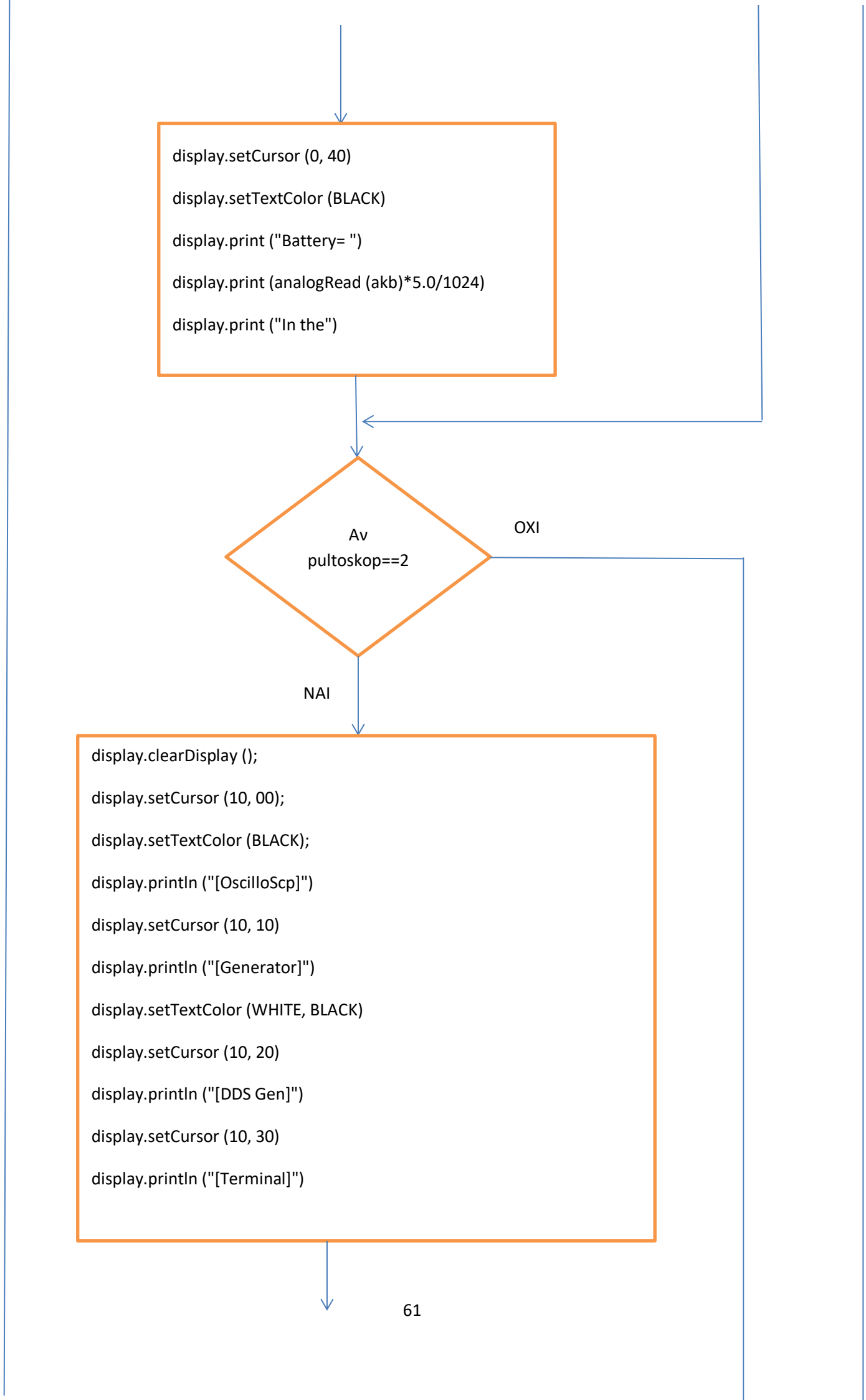
```
display.println("Press OK to start");
if(digitalRead(pravo)==HIGH){speedTTL=speedTTL+100;}
if(digitalRead(levo)==HIGH){speedTTL=speedTTL-100;}
if(speedTTL<0){speedTTL=250000;}
if(speedTTL>250000){speedTTL=0;}
if(digitalRead(ok)==HIGH){Serial.begin(speedTTL*(16/overclock));
display.clearDisplay();
    delay(100);
    display.display();
    int x=0;
    int y=0;
    while(1){
        char incomingBytes;
        if (Serial.available() > 0) { // Εάν υπάρχουν δεδομένα στο buffer
           incomingBytes=Serial.read(); // Read bytes into a variable incomeByte
display.setCursor(x,y);
            display.print(incomingBytes); // Εκτύπωση συμβολοσειράς στην οθόνη buffer
            display.display(); x=x+6;
            if(x==84){x=0;y=y+8;}
            if(y==48){x=0;y=0;
              display.clearDisplay();
              delay(100);
              display.display();}}
    }}
delay(100);
display.display();
}
////////////////////TTL
```

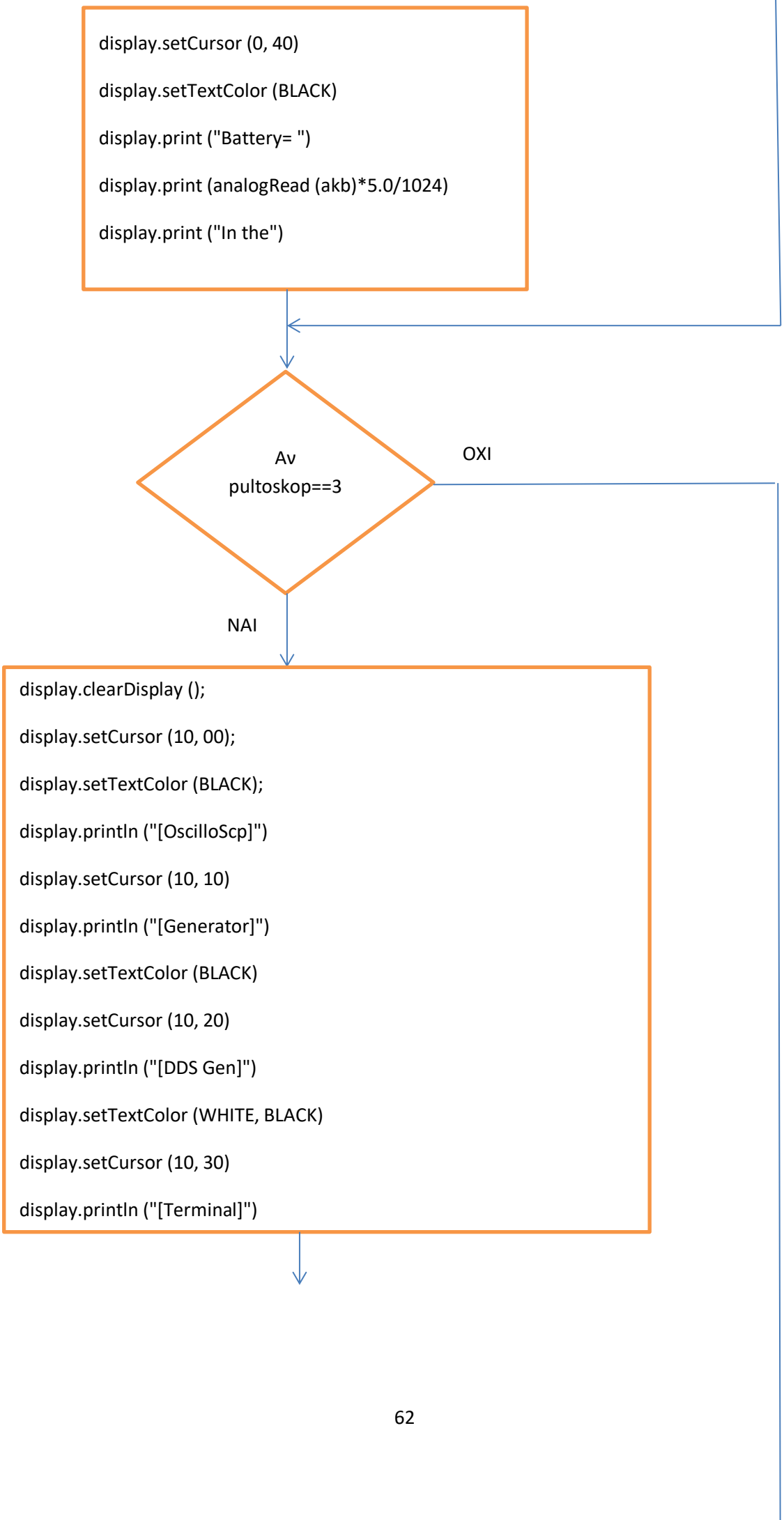
5.3 Διάγραμμα ροής

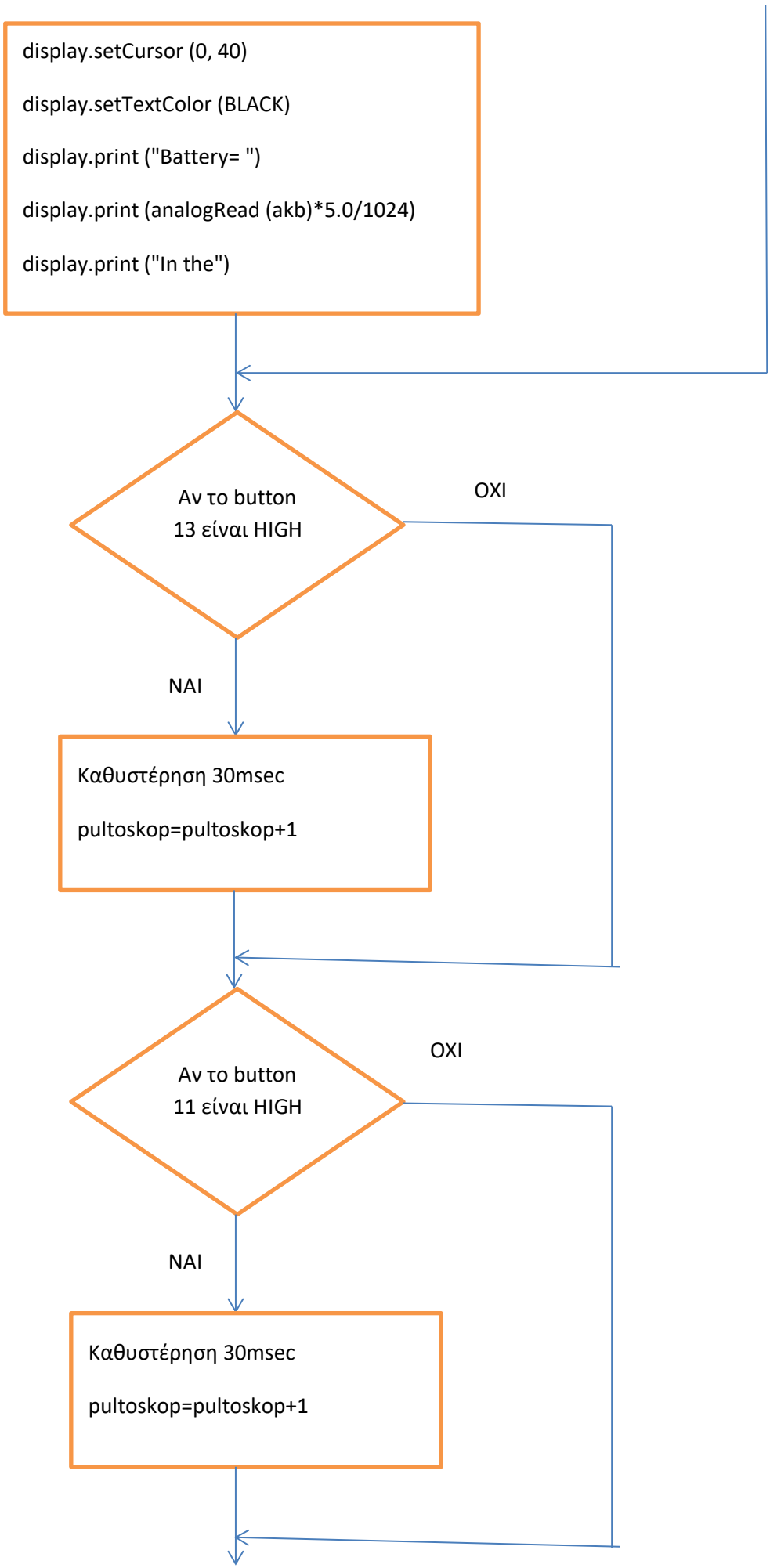


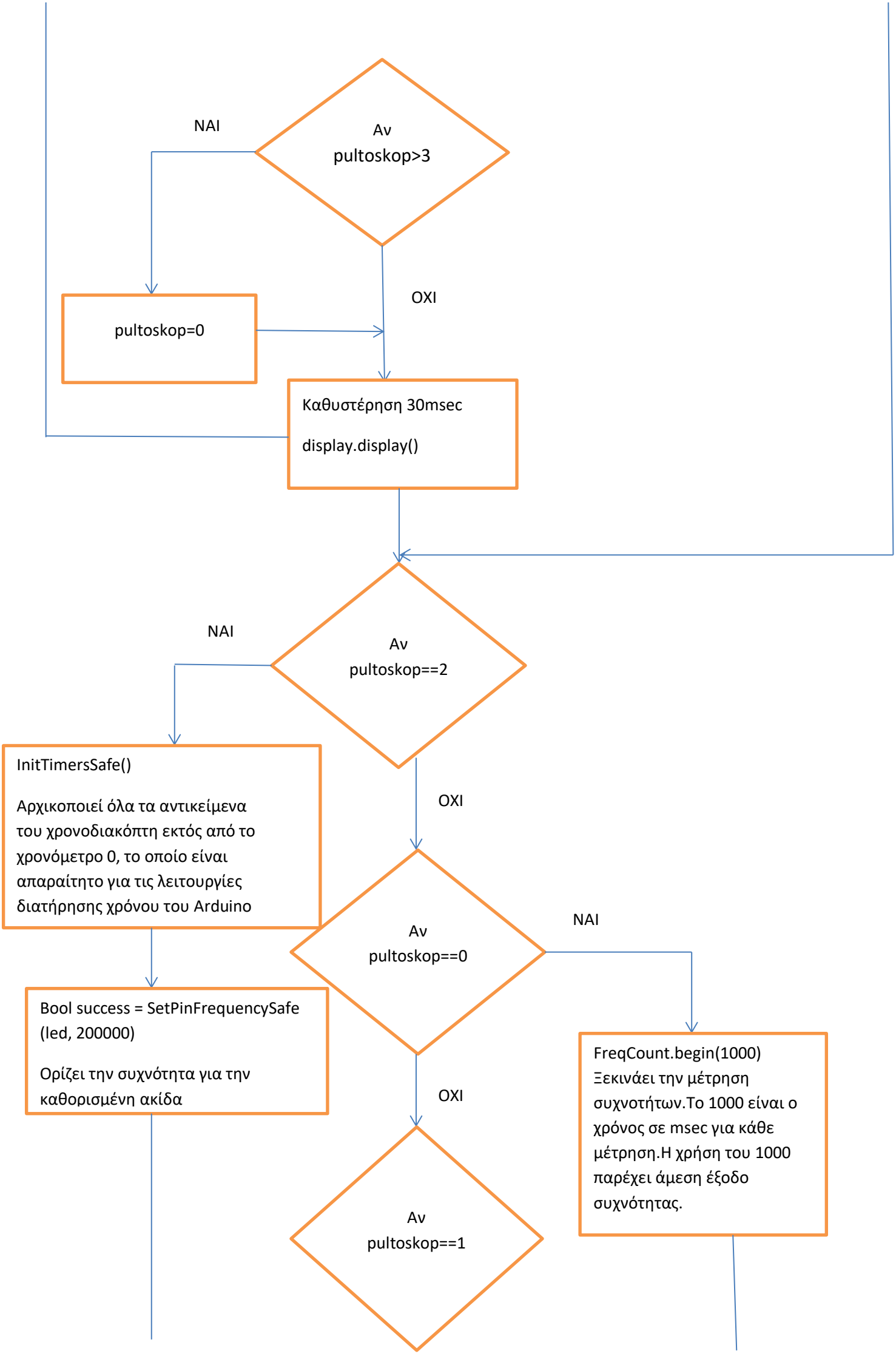

DigitalWrite (OFF, LOW)
Αν
pultoskop==0
ΝΑΙ
display.clearDisplay ()
Αυτό δεν αλλάζει το buffer απλώς καθαρίζει την μνήμη RAM της οθόνης.
display.setCursor (10, 0)
Η επάνω αριστερή γωνία της οθόνης.
display.setTextColor (WHITE, BLACK)
Ορίζουμαι το χρώμα του κειμένου.

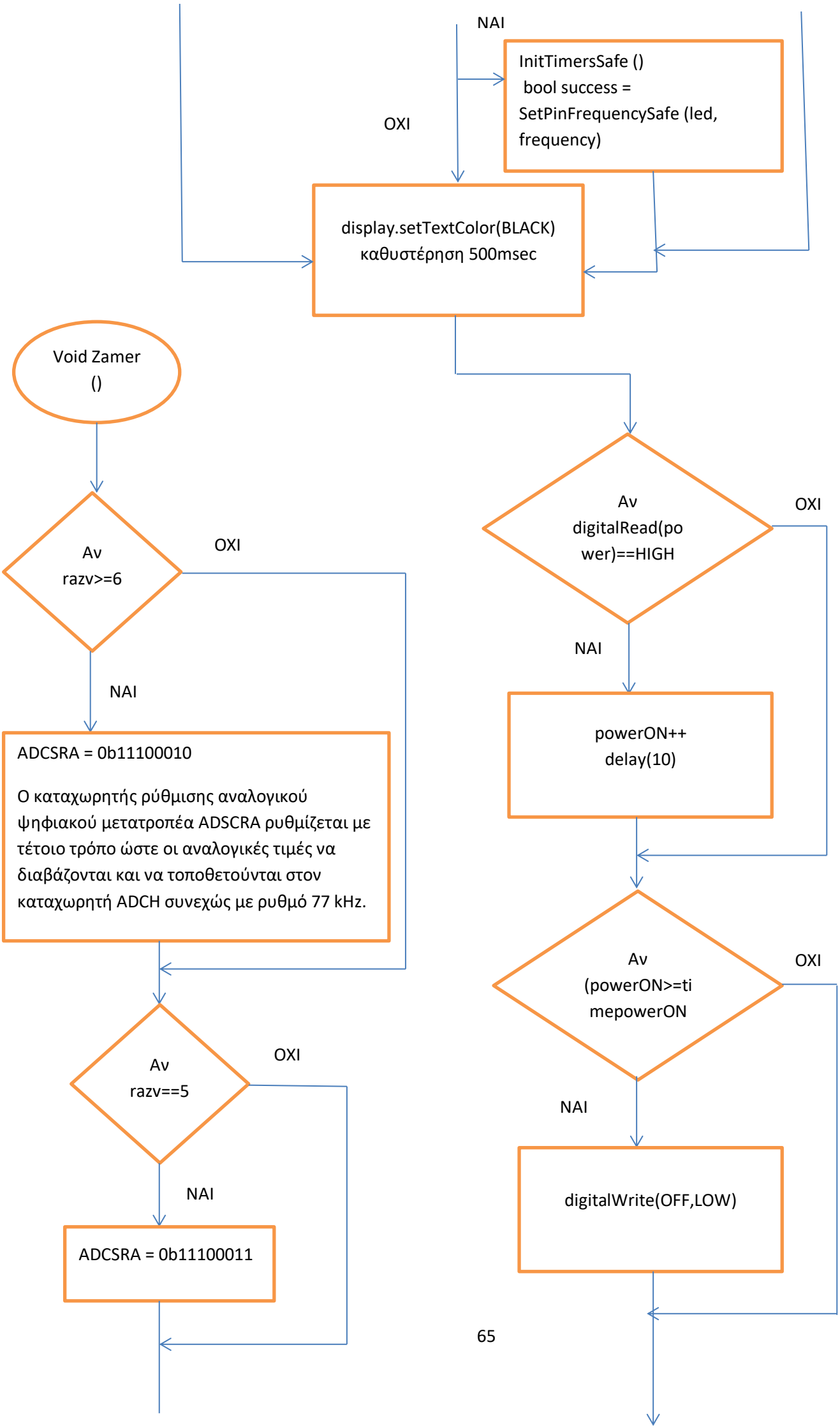

display.println("[OscilloScp]")
display.setCursor (10, 10)
display.setTextColor (BLACK)
display.println ("[Generator]")
display.setCursor (10, 20)
display.println ("[DDS Gen]")
display.setCursor (10, 30)
display.println ("[Terminal]")
display.setCursor (0, 40);
display.print ("Battery= ");
display.print (analogRead (akb)*5.0/1024);
display.print ("In the");
Av
pultoskop==1
OXI
NAI
display.clearDisplay ();
display.setCursor (10, 0);
display.setTextColor (BLACK);
display.println ("[OscilloScp]")
display.setCursor (10, 10)
display.setTextColor (WHITE, BLACK)
display.println ("[Generator]")
display.setTextColor (BLACK)
display.setCursor (10, 20)
display.println ("[DDS Gen]")
display.setCursor (10, 30)
display.println ("[Terminal]")

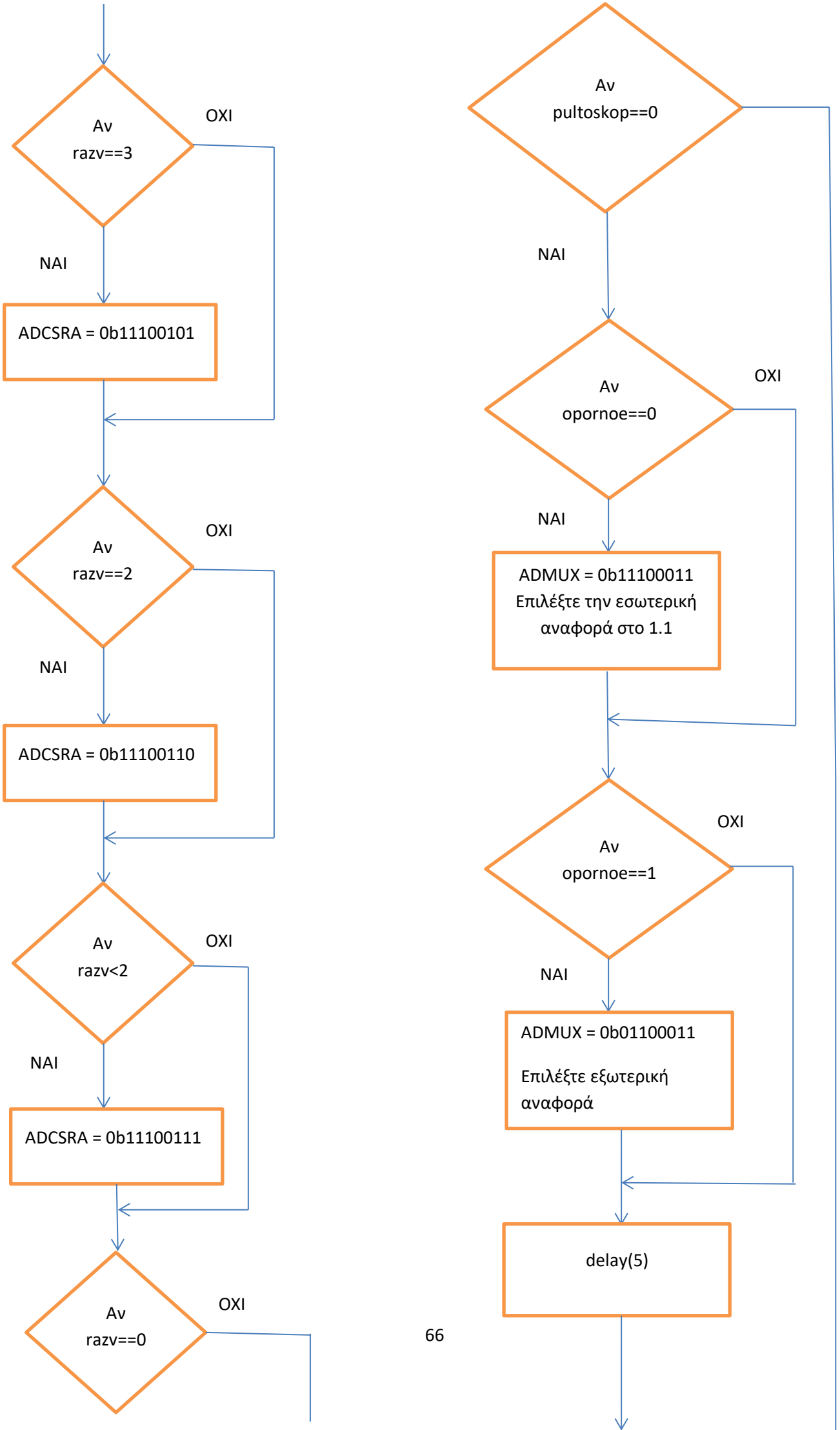

display.setCursor (0, 40)
display.setTextColor (BLACK)
display.print ("Battery= ")
display.print (analogRead (akb)*5.0/1024)
display.print ("In the")
Av
pultoskop==2
OXI
NAI
display.clearDisplay ();
display.setCursor (10, 00);
display.setTextColor (BLACK);
display.println ("[OscilloScp]")
display.setCursor (10, 10)
display.println ("[Generator]")
display.setTextColor (WHITE, BLACK)
display.setCursor (10, 20)
display.println ("[DDS Gen]")
display.setCursor (10, 30)
display.println ("[Terminal]")

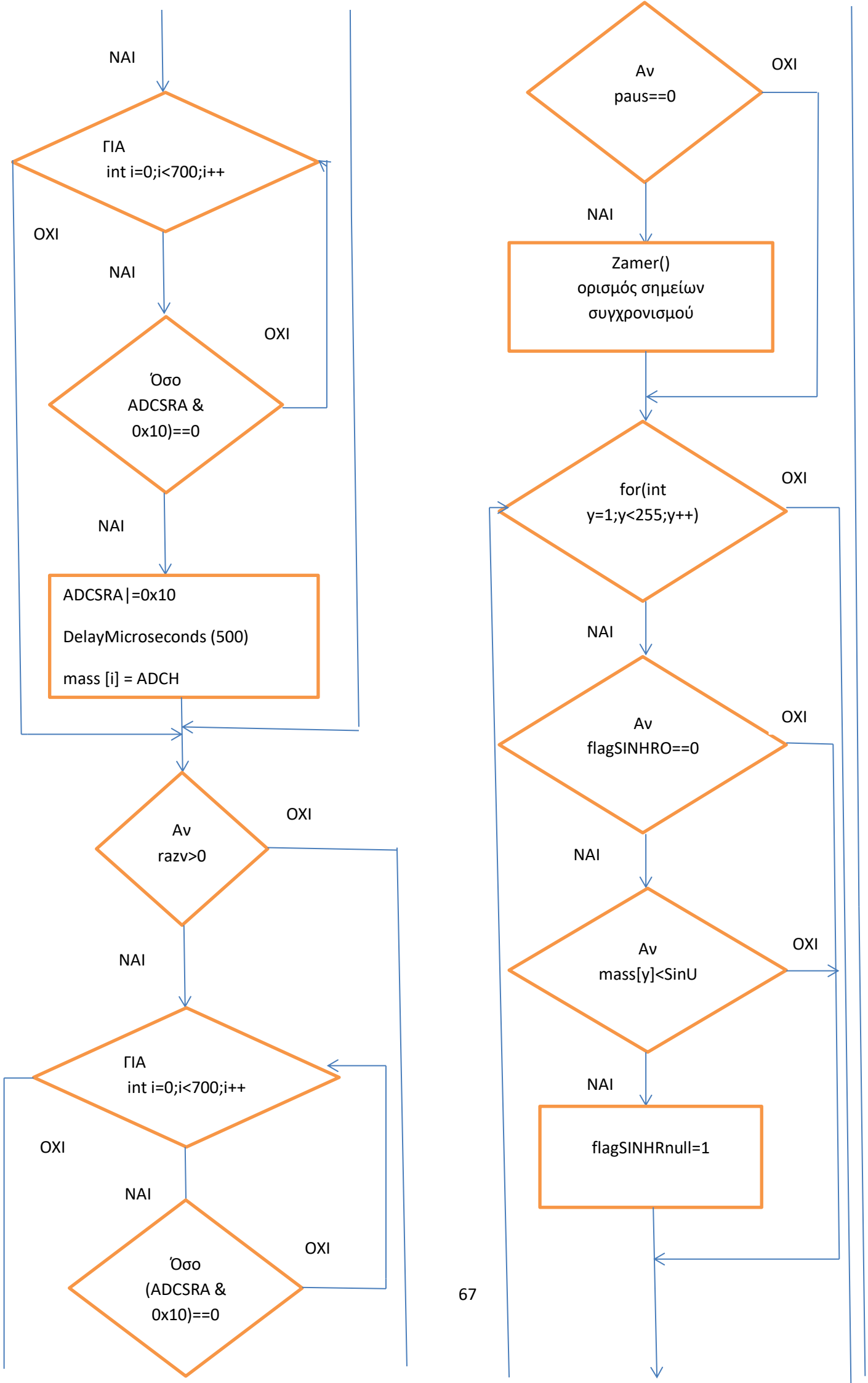

display.setCursor (0, 40)
display.setTextColor (BLACK)
display.print ("Battery= ")
display.print (analogRead (akb)*5.0/1024)
display.print ("In the")
Av
pultoskop==3
OXI
NAI
display.clearDisplay ();
display.setCursor (10, 00);
display.setTextColor (BLACK);
display.println ("[OscilloScp]")
display.setCursor (10, 10)
display.println ("[Generator]")
display.setTextColor (BLACK)
display.setCursor (10, 20)
display.println ("[DDS Gen]")
display.setTextColor (WHITE, BLACK)
display.setCursor (10, 30)
display.println ("[Terminal]")

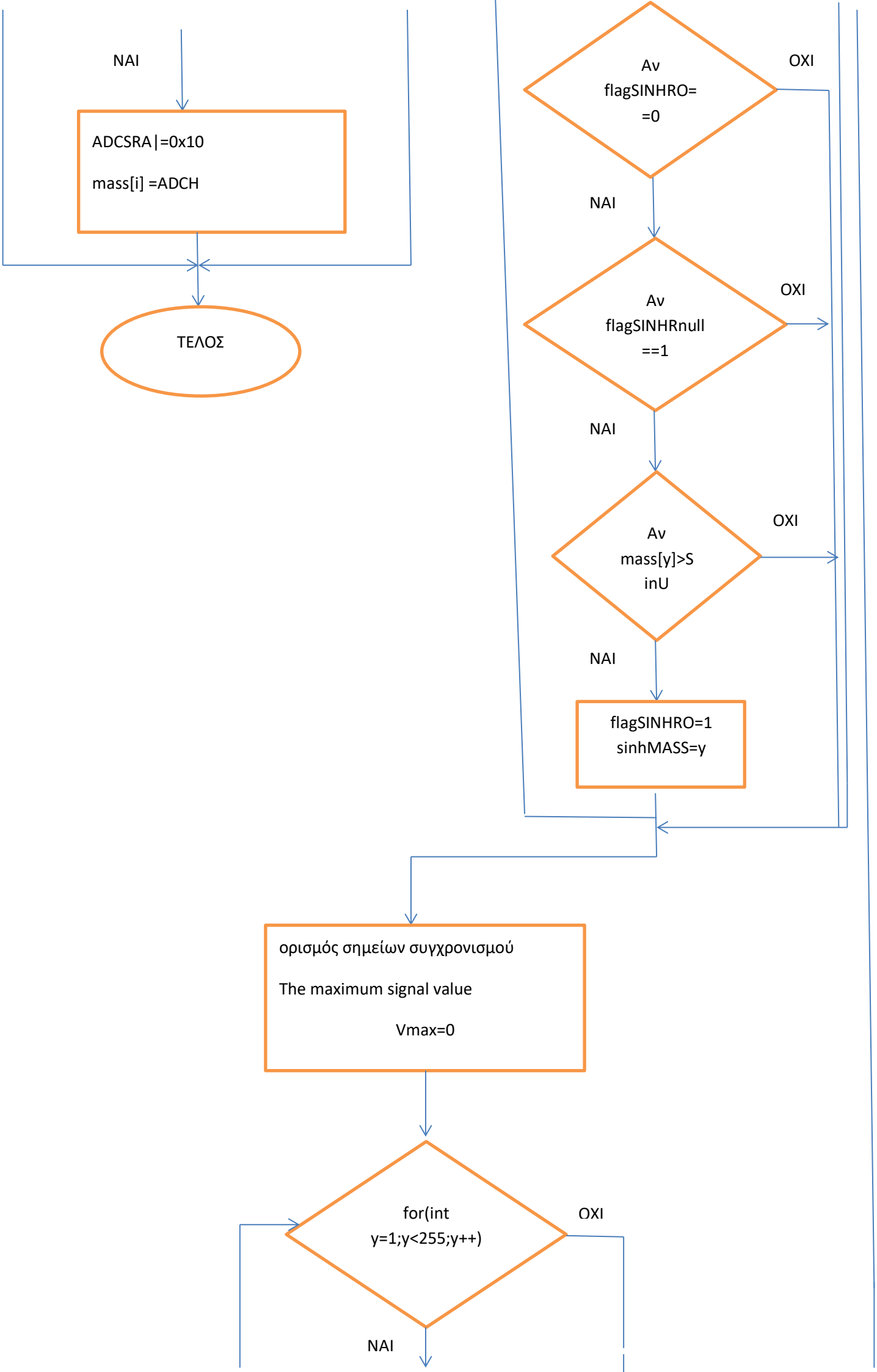

display.setCursor (0, 40)
display.setTextColor (BLACK)
display.print ("Battery= ")
display.print (analogRead (akb)*5.0/1024)
display.print ("In the")
Αν το button
13 είναι HIGH
ΟΧΙ
ΝΑΙ
Καθυστέρηση 30msec
pultoskop=pultoskop+1
Αν το button
11 είναι HIGH
ΟΧΙ
ΝΑΙ
Καθυστέρηση 30msec
pultoskop=pultoskop+1

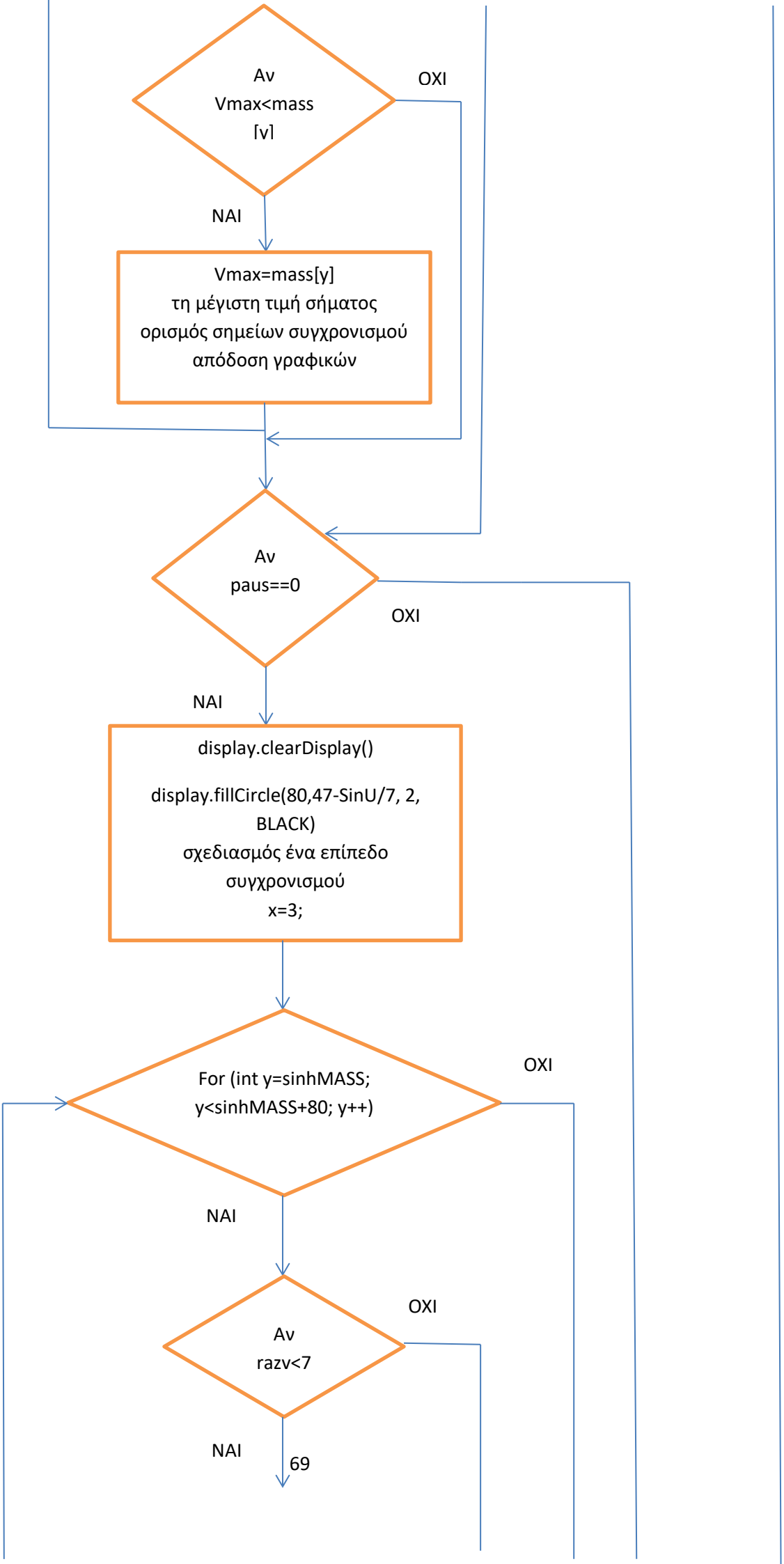

Αν
pultoskop>3
ΝΑΙ
pultoskop=0
ΟΧΙ
Καθυστέρηση 30msec
display.display()
Αν
pultoskop==2
ΝΑΙ
InitTimersSafe()
Αρχικοποιεί όλα τα αντικείμενα του χρονοδιακόπτη εκτός από το χρονόμετρο 0, το οποίο είναι απαραίτητο για τις λειτουργίες διατήρησης χρόνου του Arduino
ΟΧΙ
Αν
pultoskop==0
ΝΑΙ
Bool success = SetPinFrequencySafe (led, 200000)
Ορίζει την συχνότητα για την καθορισμένη ακίδα
FreqCount.begin(1000)
Ξεκινάει την μέτρηση συχνοτήτων.Το 1000 είναι ο χρόνος σε msec για κάθε μέτρηση.Η χρήση του 1000 παρέχει άμεση έξοδο συχνότητας.
ΟΧΙ
Αν
pultoskop==1

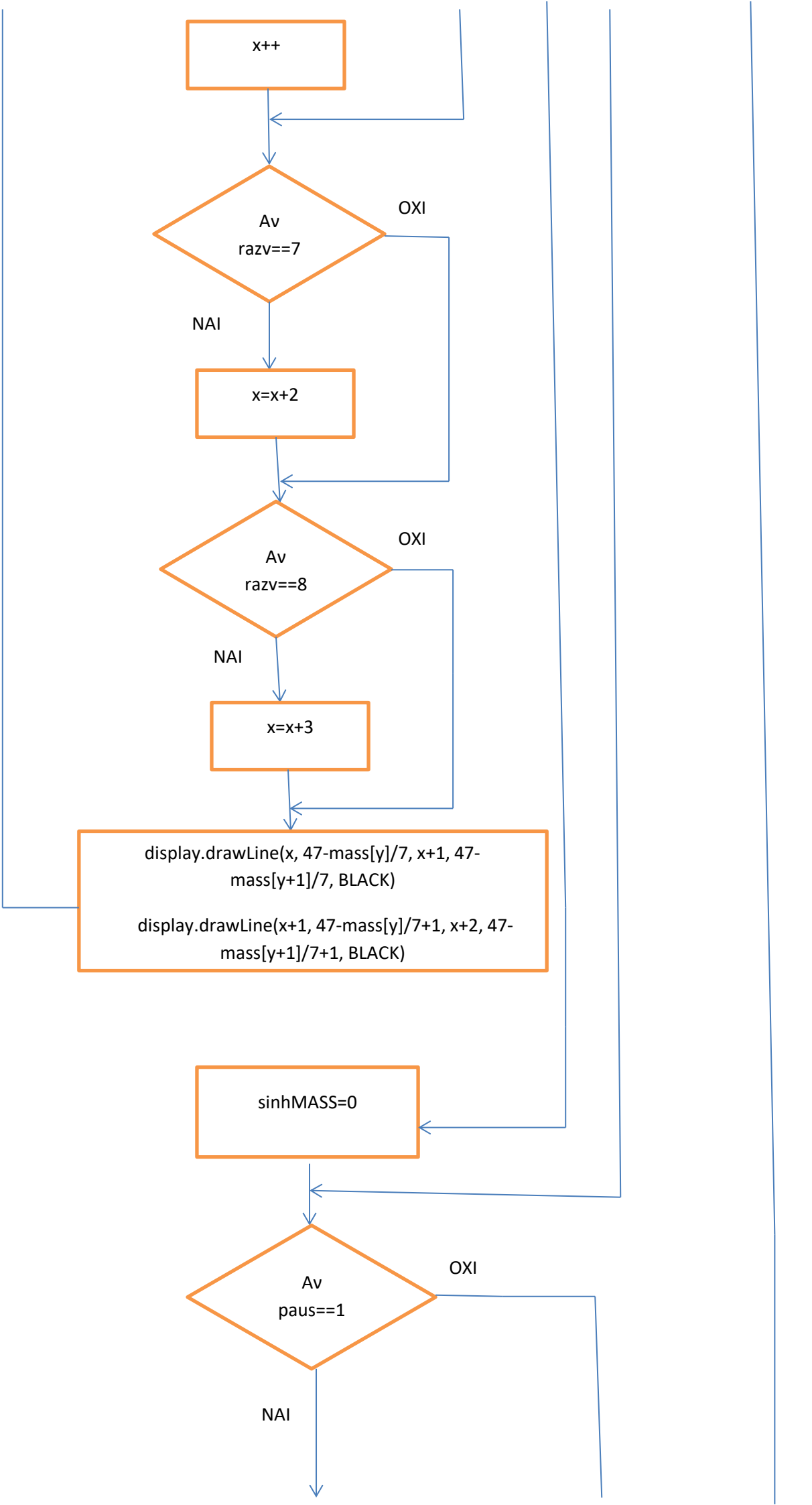

NAI
InitTimersSafe ()
bool success =
SetPinFrequencySafe (led,
frequency)
OXI
display.setTextColor(BLACK)
καθυστέρηση 500msec
Void Zamer
()
Av
digitalRead(po
wer)==HIGH
OXI
NAI
powerON++
delay(10)
Av
(powerON>=ti
mepowerON
OXI
NAI
digitalWrite(OFF,LOW)
Av
razv>=6
OXI
NAI
ADCSRA = 0b11100010
Ο καταχωρητής ρύθμισης αναλογικού ψηφιακού μετατροπέα ADSCRA ρυθμίζεται με τέτοιο τρόπο ώστε οι αναλογικές τιμές να διαβάζονται και να τοποθετούνται στον καταχωρητή ADCH συνεχώς με ρυθμό 77 kHz.
Av
razv==5
OXI
NAI
ADCSRA = 0b11100011

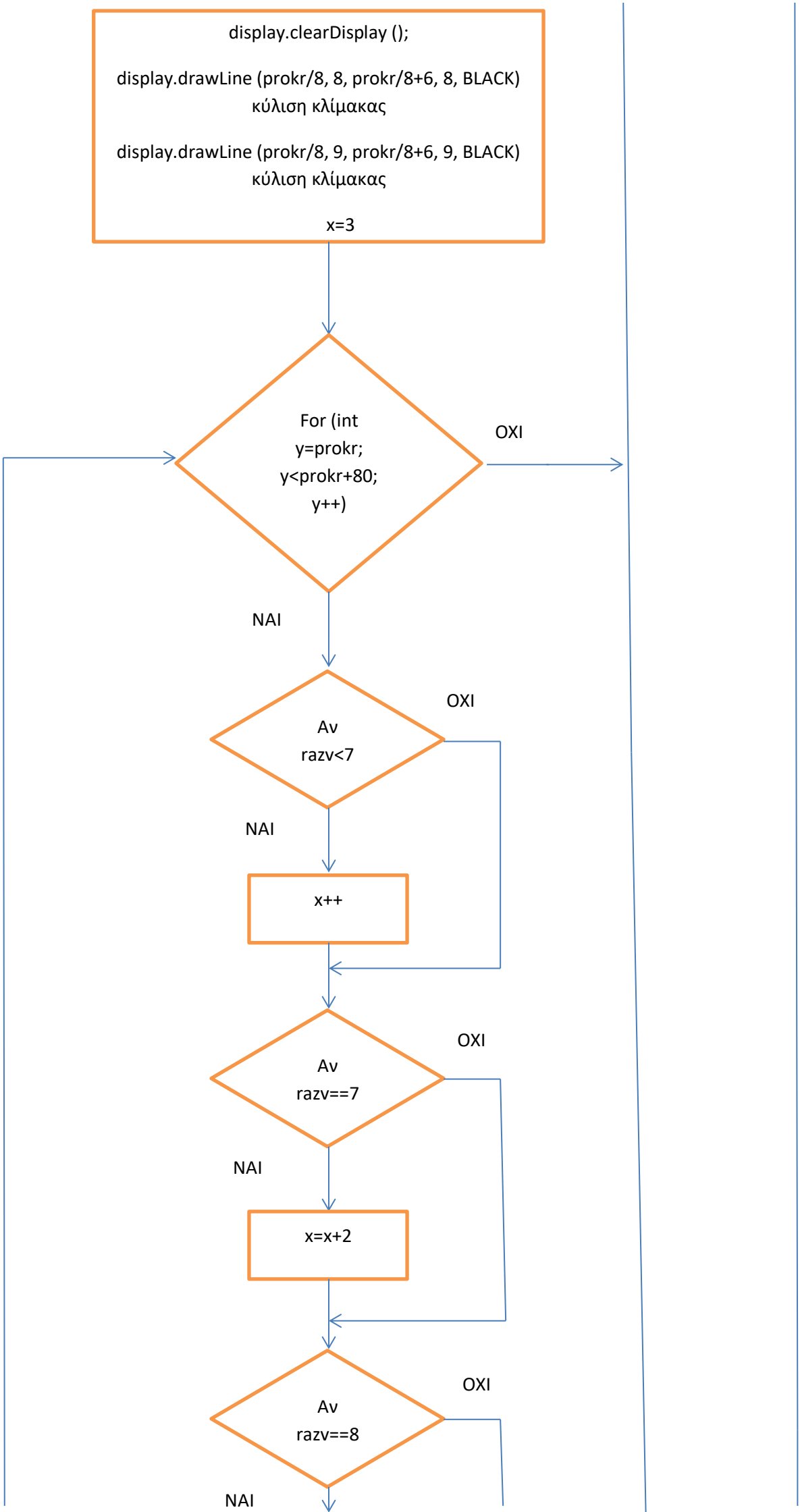

Αν
razv==3
ΟΧΙ
ΝΑΙ
ADCSRA = 0b11100101
Αν
razv==2
ΟΧΙ
ΝΑΙ
ADCSRA = 0b11100110
Αν
razv<2
ΟΧΙ
ΝΑΙ
ADCSRA = 0b11100111
Αν
razv==0
ΟΧΙ
Αν
pultoskop==0
ΝΑΙ
Αν
opornoe==0
ΟΧΙ
ΝΑΙ
ADMUX = 0b11100011
Επιλέξτε την εσωτερική
αναφορά στο 1.1
Αν
opornoe==1
ΟΧΙ
ΝΑΙ
ADMUX = 0b01100011
Επιλέξτε εξωτερική
αναφορά
delay(5)

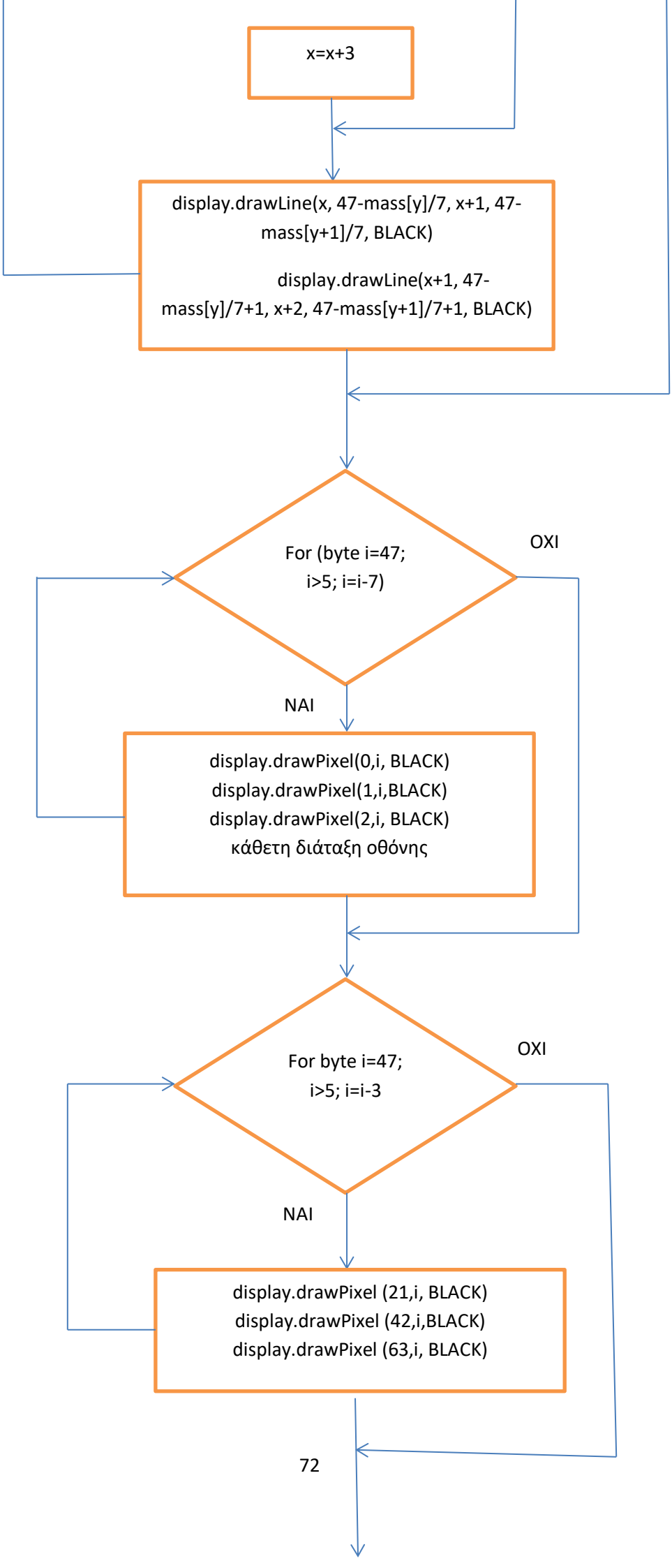

ΝΑΙ
ΓΙΑ
int i=0;i<700;i++
ΟΧΙ
ΝΑΙ
Όσο
ADCSRA &
0x10)==0
ΟΧΙ
ΝΑΙ
ADCSRA|=0x10
DelayMicroseconds (500)
mass [i] = ADCH
Αν
razv>0
ΟΧΙ
ΝΑΙ
ΓΙΑ
int i=0;i<700;i++
ΟΧΙ
ΝΑΙ
Όσο
(ADCSRA &
0x10)==0
ΟΧΙ
Αν
paus==0
ΟΧΙ
ΝΑΙ
Zamer()
ορισμός σημείων
συγχρονισμού
for(int
y=1;y<255;y++)
ΟΧΙ
ΝΑΙ
Αν
flagSINHRO==0
ΟΧΙ
ΝΑΙ
Αν
mass[y]<SinU
ΟΧΙ
ΝΑΙ
flagSINHRnull=1

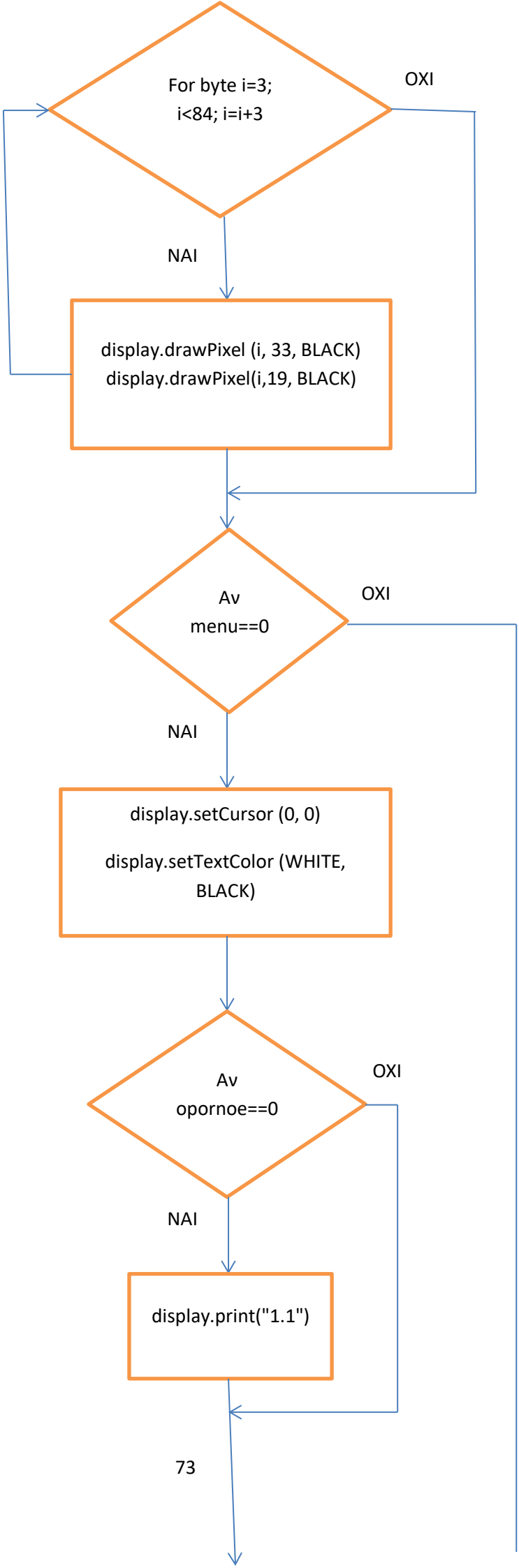

NAI
ADCSRA|=0x10
mass[i] =ADCH
ΤΕΛΟΣ
Αν
flagSINHRO=
=0
OXI
NAI
Αν
flagSINHRnull
==1
OXI
NAI
Αν
mass[y]>S
inU
OXI
NAI
flagSINHRO=1
sinhMASS=y
ορισμός σημείων συγχρονισμού
The maximum signal value
Vmax=0
for(int
y=1;y<255;y++)
OXI
NAI

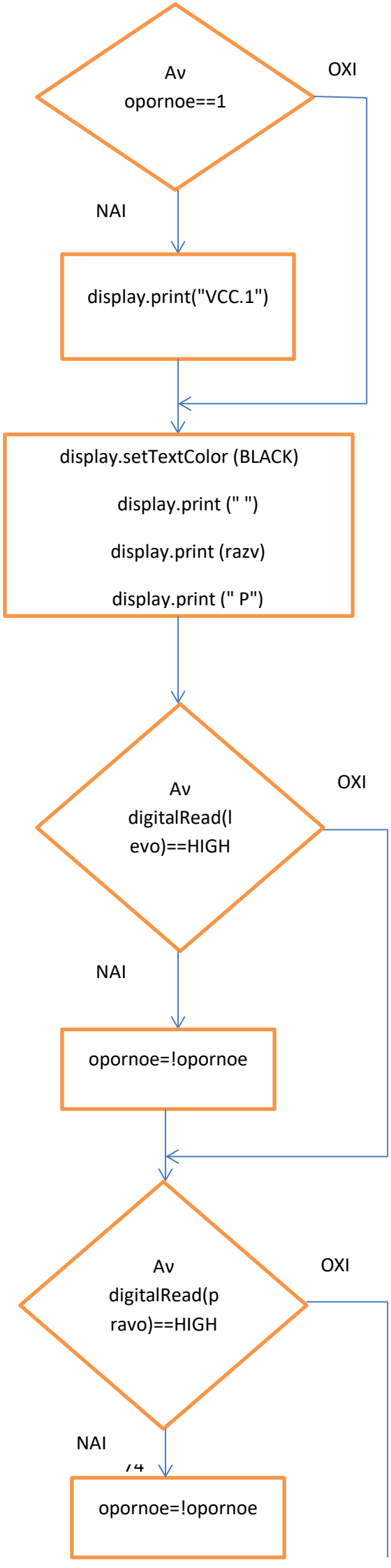

Αν
Vmax<mass
[v]
ΟΧΙ
ΝΑΙ
Vmax=mass[y]
τη μέγιστη τιμή σήματος
ορισμός σημείων συγχρονισμού
απόδοση γραφικών
Αν
paus==0
ΟΧΙ
ΝΑΙ
display.clearDisplay()
display.fillCircle(80,47-SinU/7, 2,
BLACK)
σχεδιασμός ένα επίπεδο
συγχρονισμού
x=3;
For (int y=sinhMASS;
y<sinhMASS+80; y++)
ΟΧΙ
ΝΑΙ
Αν
razv<7
ΟΧΙ
ΝΑΙ
69

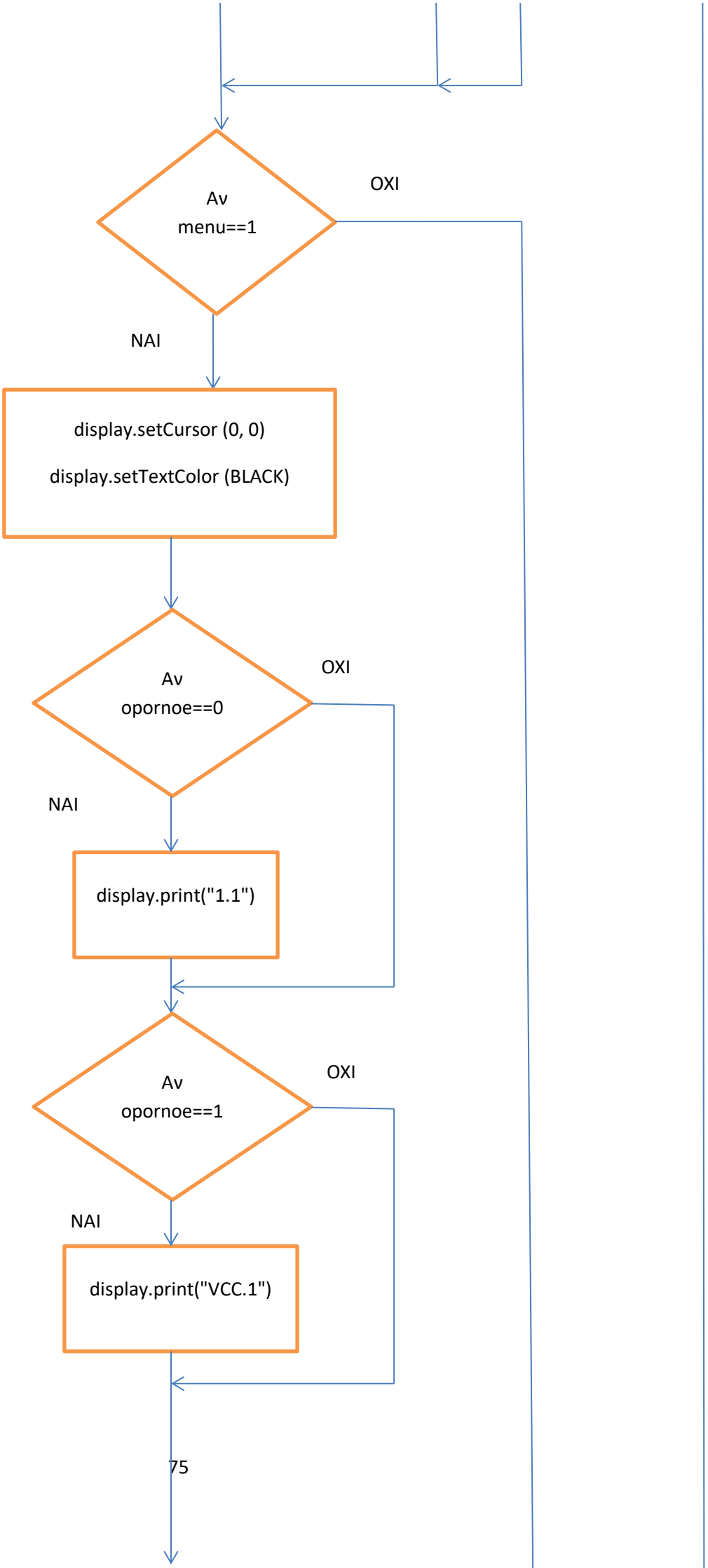

x++
Av
razv==7
OXI
NAI
x=x+2
Av
razv==8
OXI
NAI
x=x+3
display.drawLine(x, 47-mass[y]/7, x+1, 47-mass[y+1]/7, BLACK)
display.drawLine(x+1, 47-mass[y]/7+1, x+2, 47-mass[y+1]/7+1, BLACK)
sinhMASS=0
Av
paus==1
OXI
NAI

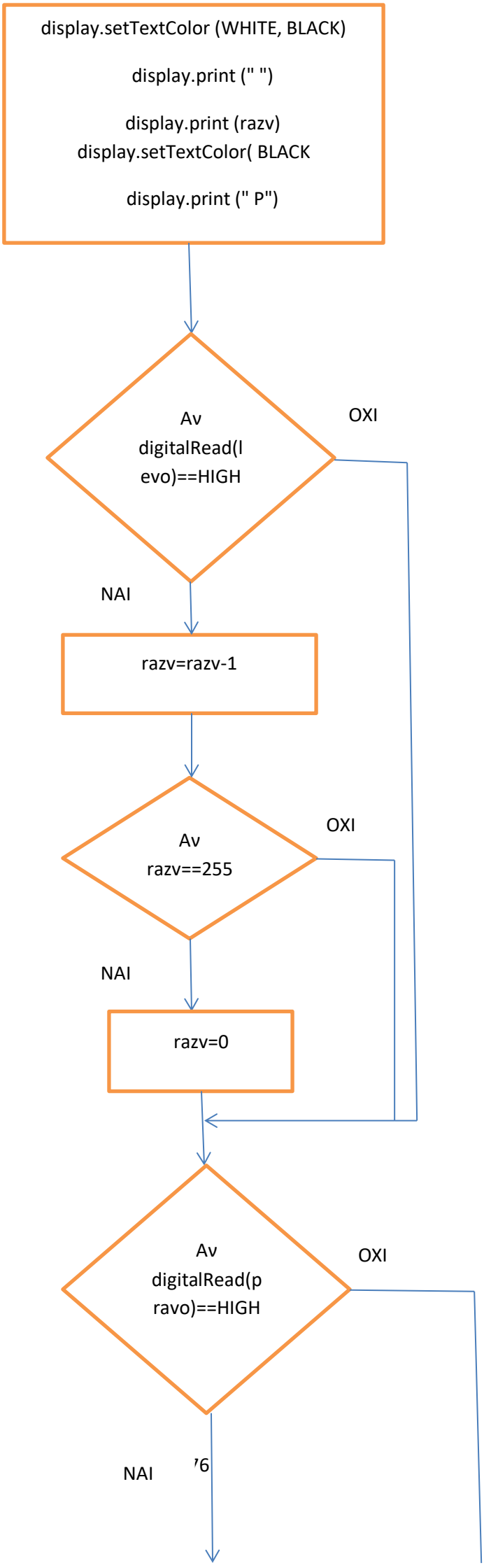

display.clearDisplay ();
display.drawLine (prokr/8, 8, prokr/8+6, 8, BLACK)
κύλιση κλίμακας
display.drawLine (prokr/8, 9, prokr/8+6, 9, BLACK)
κύλιση κλίμακας
x=3
For (int y=prokr; y<prokr+80; y++)
ΟΧΙ
ΝΑΙ
Αν razv<7
ΟΧΙ
ΝΑΙ
x++
Αν razv==7
ΟΧΙ
ΝΑΙ
x=x+2
Αν razv==8
ΟΧΙ
ΝΑΙ

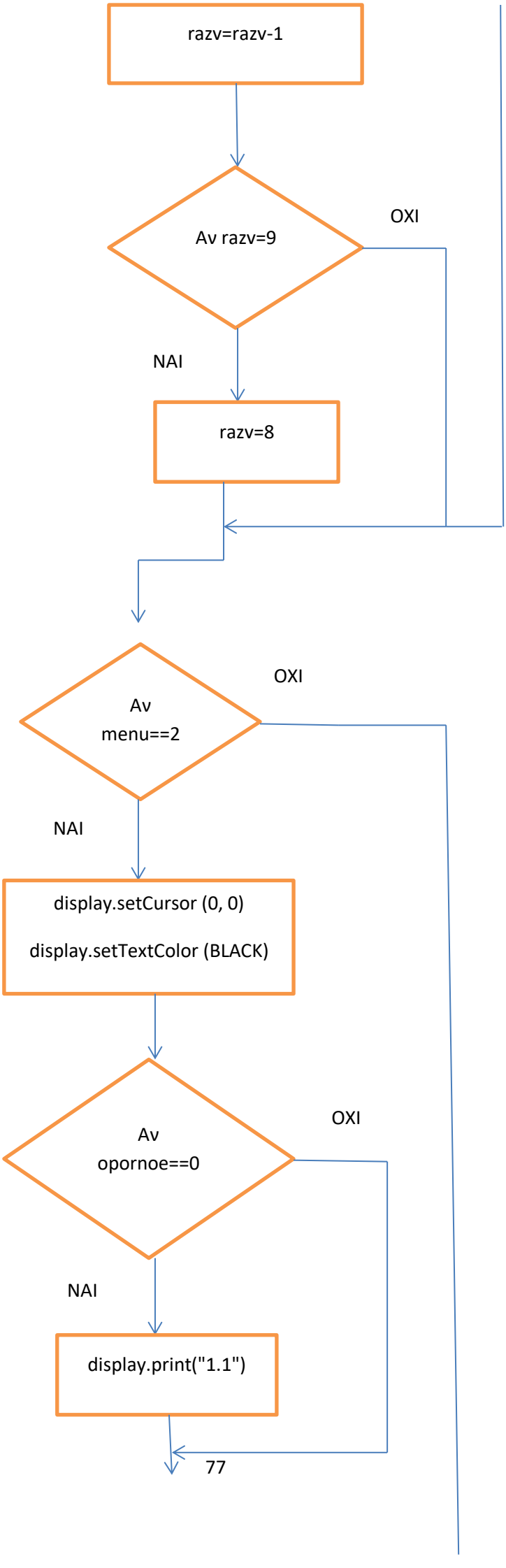

x=x+3
display.drawLine(x, 47-mass[y]/7, x+1, 47-mass[y+1]/7, BLACK)
display.drawLine(x+1, 47-mass[y]/7+1, x+2, 47-mass[y+1]/7+1, BLACK)
For (byte i=47; i>5; i=i-7)
OXI
NAI
display.drawPixel(0,i, BLACK)
display.drawPixel(1,i,BLACK)
display.drawPixel(2,i, BLACK)
κάθετη διάταξη οθόνης
For byte i=47; i>5; i=i-3
OXI
NAI
display.drawPixel (21,i, BLACK)
display.drawPixel (42,i,BLACK)
display.drawPixel (63,i, BLACK)

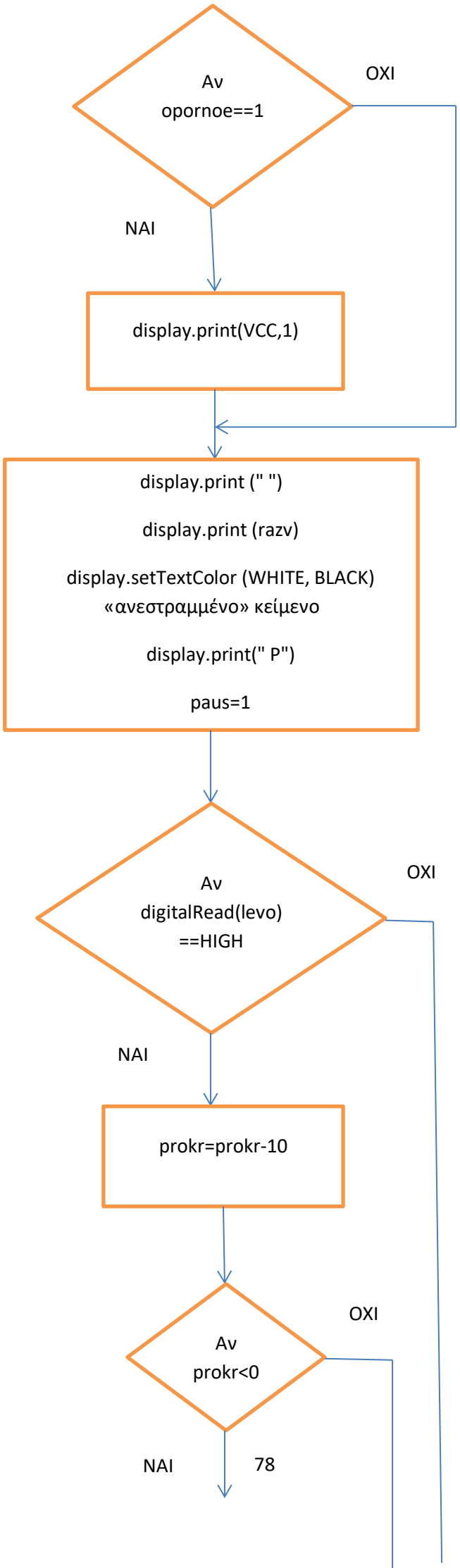

For byte i=3;
i<84; i=i+3
OXI
NAI
display.drawPixel (i, 33, BLACK)
display.drawPixel(i,19, BLACK)
Av
menu==0
OXI
NAI
display.setCursor (0, 0)
display.setTextColor (WHITE,
BLACK)
Av
opornoe==0
OXI
NAI
display.print("1.1")

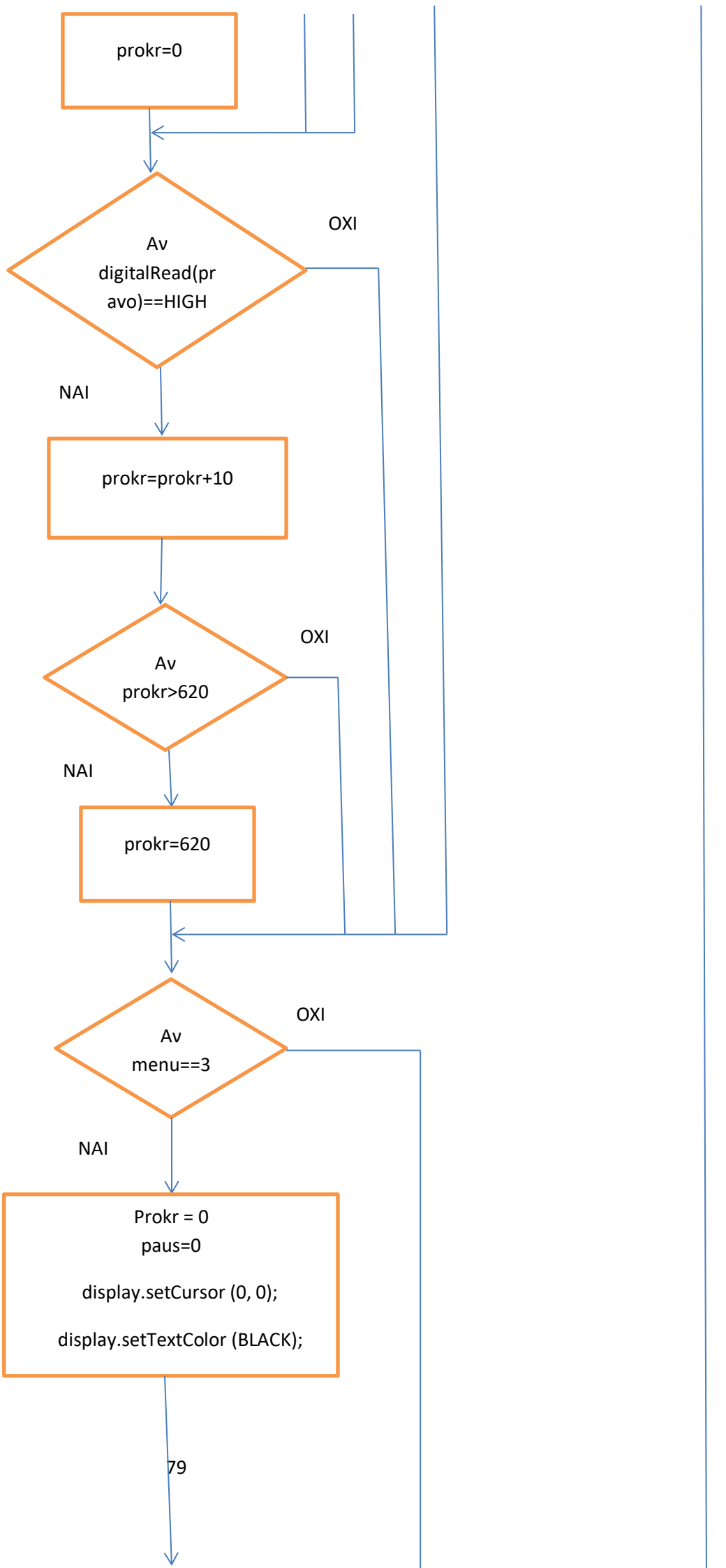

Av
opornoe==1
OXI
NAI
display.print("VCC.1")
display.setTextColor (BLACK)
display.print (" ")
display.print (razv)
display.print (" P")
Av
digitalRead(l
evo)==HIGH
OXI
NAI
opornoe=!opornoe
Av
digitalRead(p
ravo)==HIGH
OXI
NAI
opornoe=!opornoe

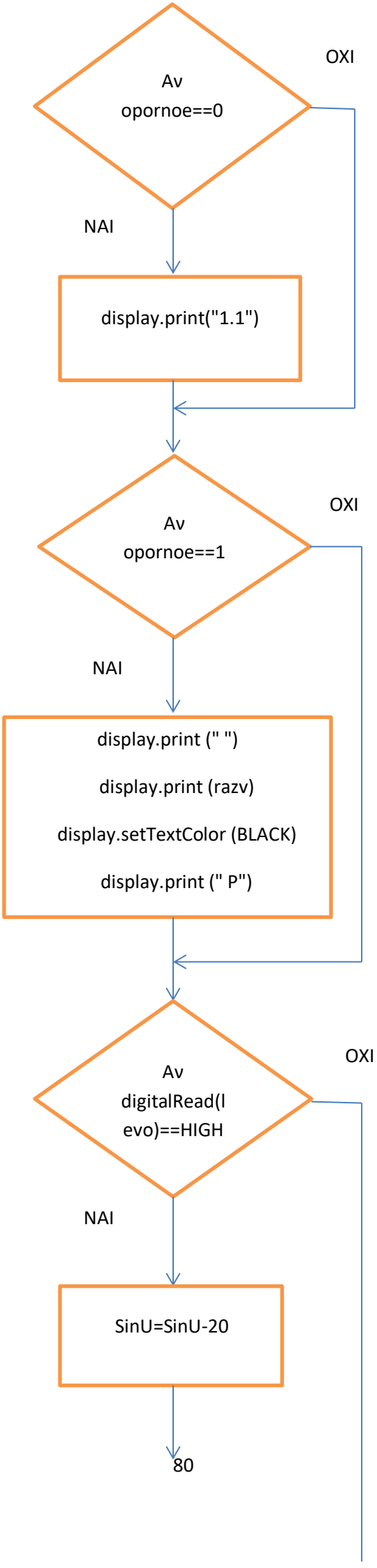

Av
menu==1
OXI
NAI
display.setCursor (0, 0)
display.setTextColor (BLACK)
Av
opornoe==0
OXI
NAI
display.print("1.1")
Av
opornoe==1
OXI
NAI
display.print("VCC.1")
75

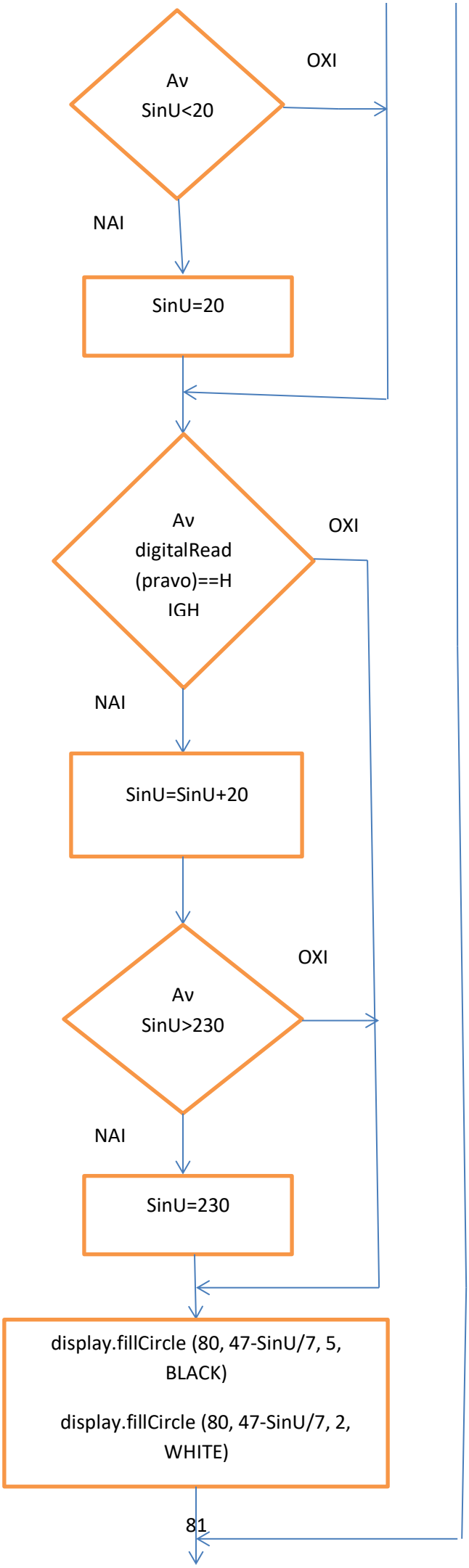

display.setTextColor (WHITE, BLACK)
display.print (" ")
display.print (razv)
display.setTextColor(BLACK
display.print (" P")
Av digitalRead(levo)==HIGH
OXI
NAI
razv=razv-1
Av razv==255
OXI
NAI
razv=0
Av digitalRead(pravo)==HIGH
OXI
NAI

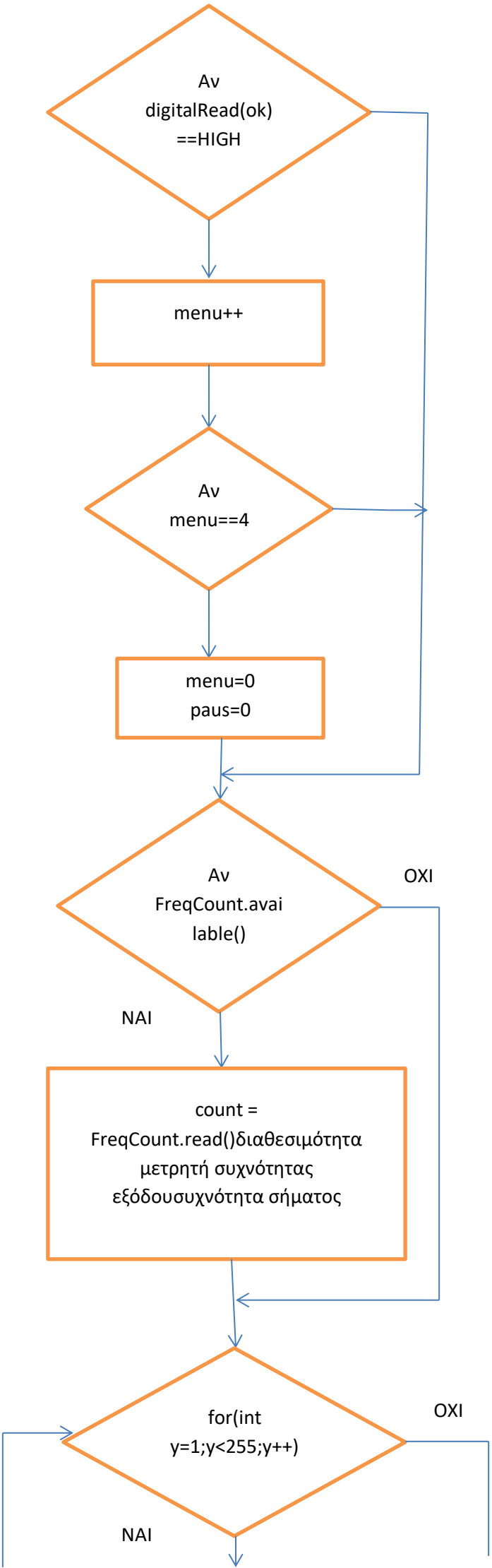

razv=razv-1
Av razv=9
OXI
NAI
razv=8
Av
menu==2
OXI
NAI
display.setCursor (0, 0)
display.setTextColor (BLACK)
Av
opornoe==0
OXI
NAI
display.print("1.1")
77

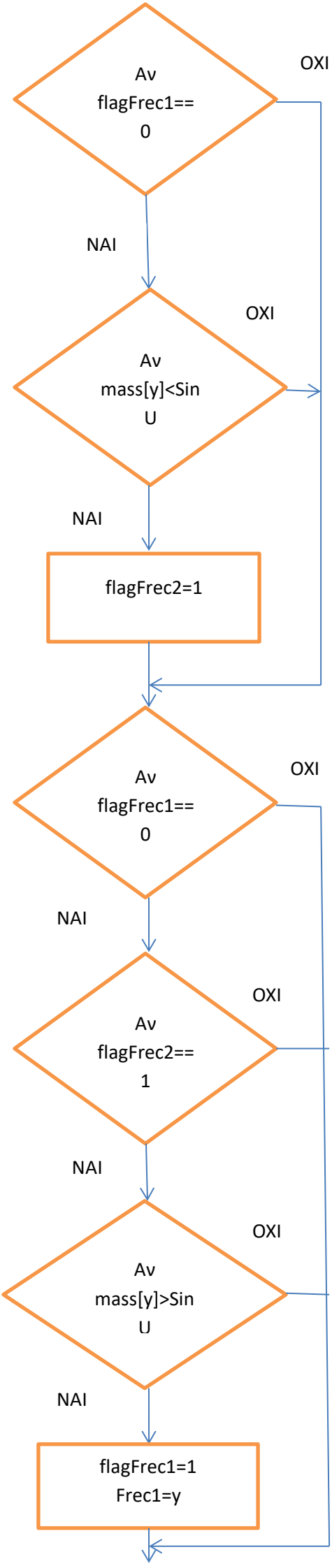

Αν
opornoe==1
ΟΧΙ
ΝΑΙ
display.print(VCC,1)
display.print (" ")
display.print (razv)
display.setTextColor (WHITE, BLACK)
«ανεστραμμένο» κείμενο
display.print(" P")
paus=1
Αν
digitalRead(levo)
==HIGH
ΟΧΙ
ΝΑΙ
prokr=prokr-10
Αν
prokr<0
ΟΧΙ
ΝΑΙ
78

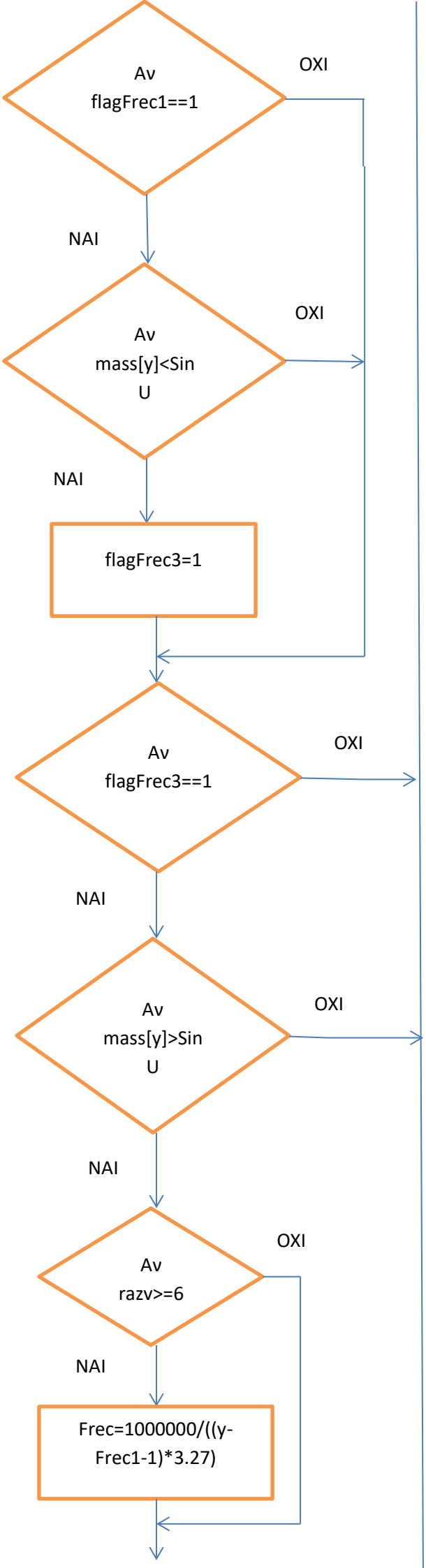

prokr=0
Av digitalRead(pravo)==HIGH
OXI
NAI
prokr=prokr+10
Av prokr>620
OXI
NAI
prokr=620
Av menu==3
OXI
NAI
Prokr = 0
paus=0
display.setCursor (0, 0);
display.setTextColor (BLACK);
79

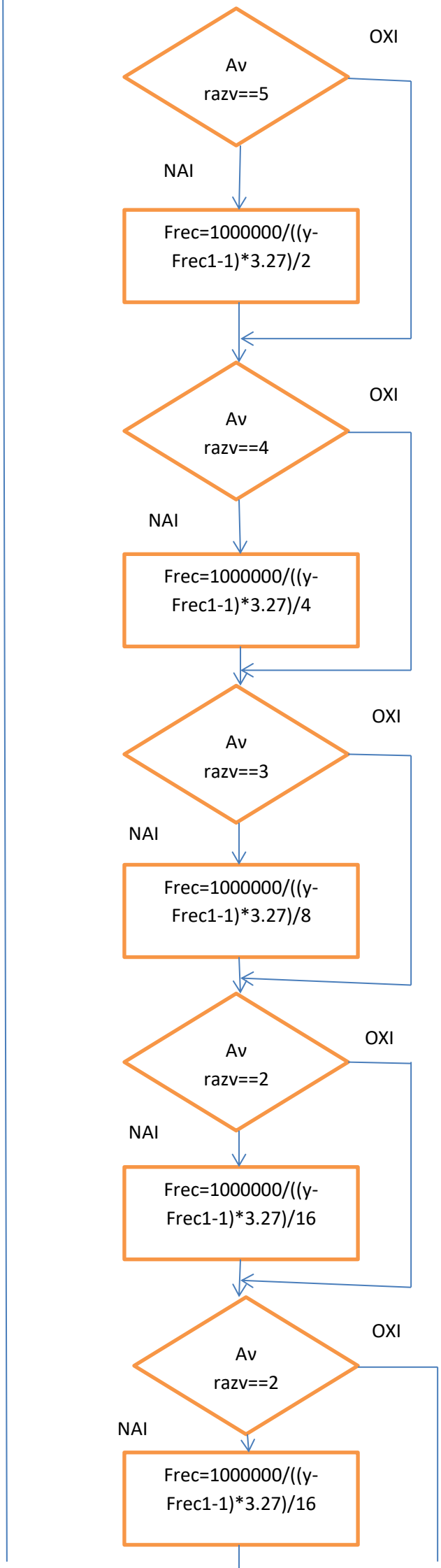

Av
opornoe==0
OXI
NAI
display.print("1.1")
Av
opornoe==1
OXI
NAI
display.print (" ")
display.print (razv)
display.setTextColor (BLACK)
display.print (" P")
Av
digitalRead(l
evo)==HIGH
OXI
NAI
SinU=SinU-20
80

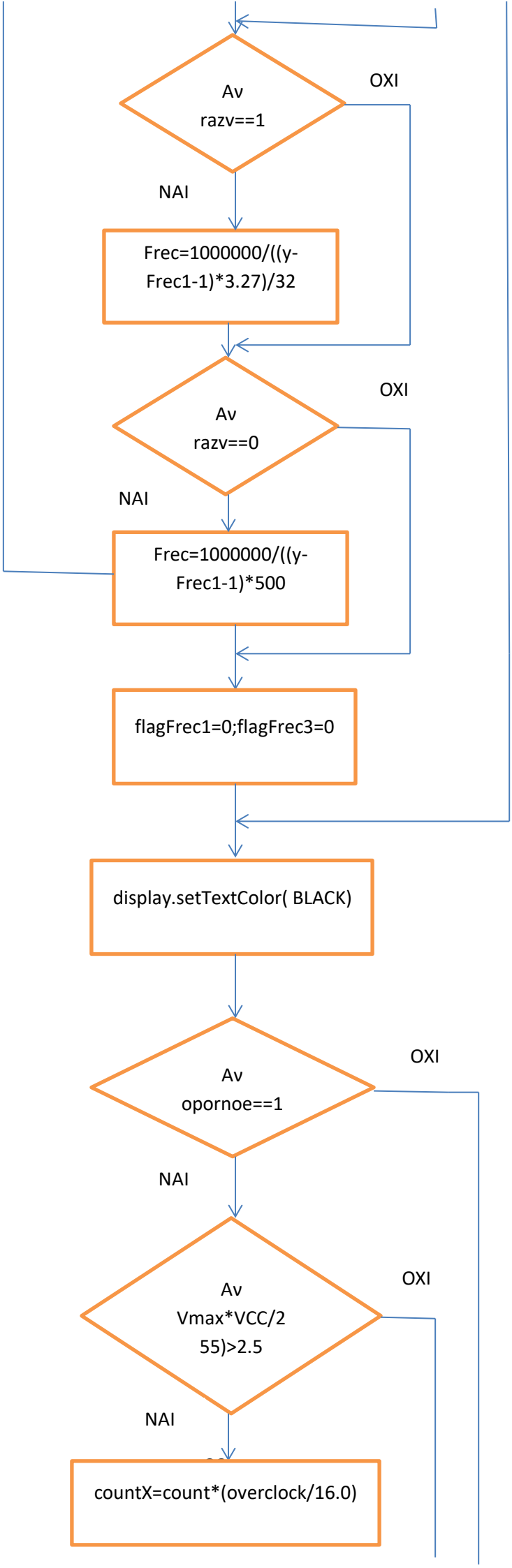

Av
SinU<20
OXI
NAI
SinU=20
Av
digitalRead
(pravo)==H
IGH
OXI
NAI
SinU=SinU+20
Av
SinU>230
OXI
NAI
SinU=230
display.fillCircle (80, 47-SinU/7, 5,
BLACK)
display.fillCircle (80, 47-SinU/7, 2,
WHITE)
81

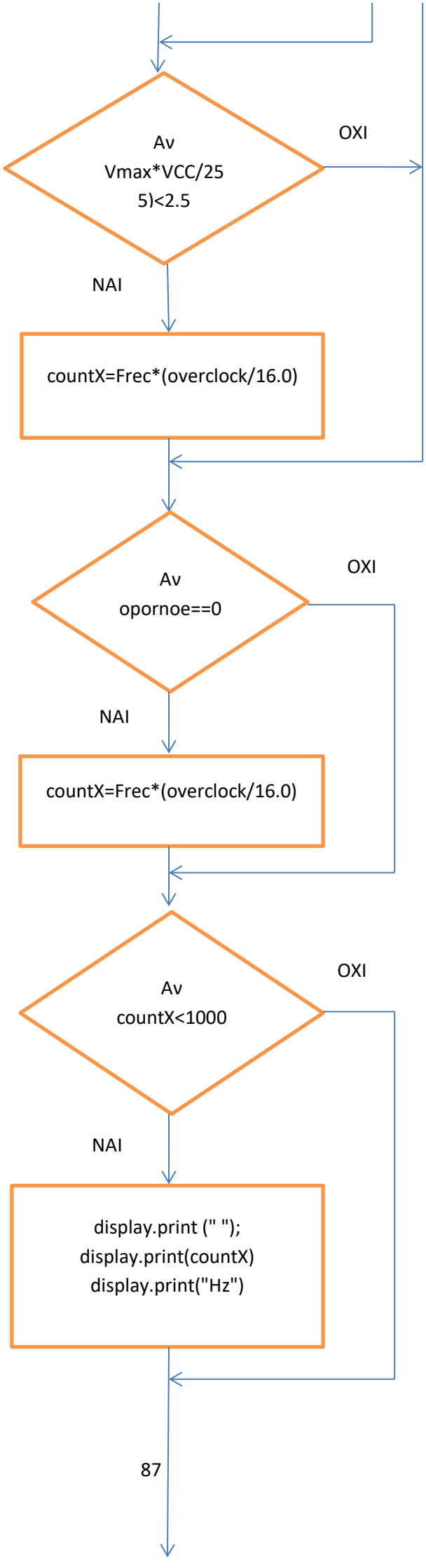

Αν
digitalRead(ok)
==HIGH
menu++
Αν
menu==4
menu=0
paus=0
Αν
FreqCount.avai
lable()
ΟΧΙ
ΝΑΙ
count =
FreqCount.read()διαθεσιμότητα
μετρητή συχνότητας
εξόδουσυχνότητα σήματος
for(int
y=1;y<255;y++)
ΟΧΙ
ΝΑΙ

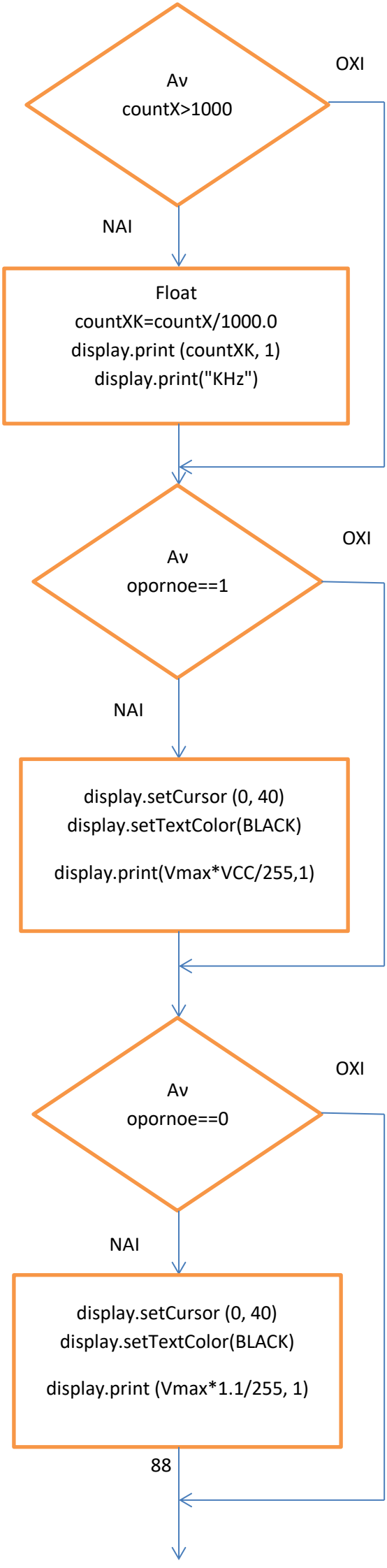

Av
flagFrec1==
0
OXI
NAI
Av
mass[y]<Sin
U
OXI
NAI
flagFrec2=1
Av
flagFrec1==
0
OXI
NAI
Av
flagFrec2==
1
OXI
NAI
Av
mass[y]>Sin
U
OXI
NAI
flagFrec1=1
Frec1=y

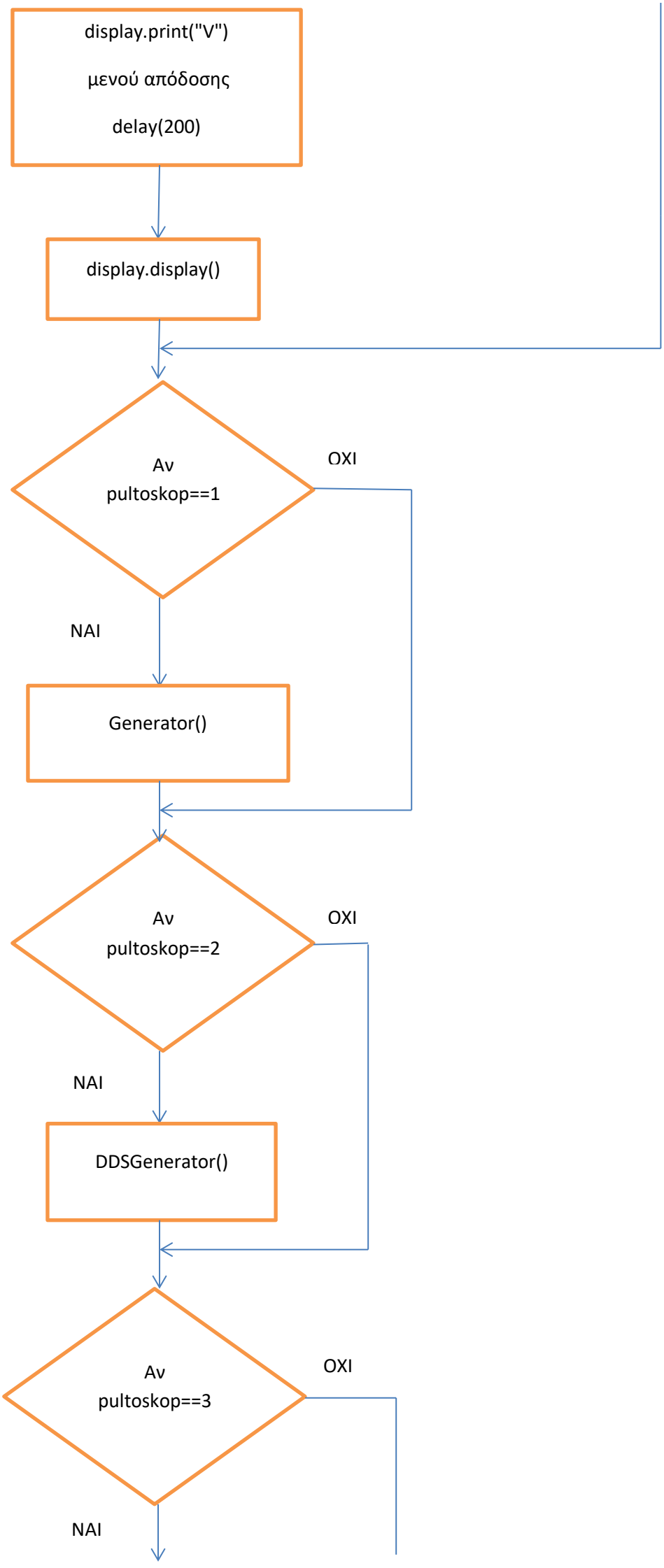

Av
flagFrec1==1
OXI
NAI
Av
mass[y]<Sin
U
OXI
NAI
flagFrec3=1
Av
flagFrec3==1
OXI
NAI
Av
mass[y]>Sin
U
OXI
NAI
Av
razv>=6
OXI
NAI
Frec=1000000/((y-
Frec1-1)*3.27)

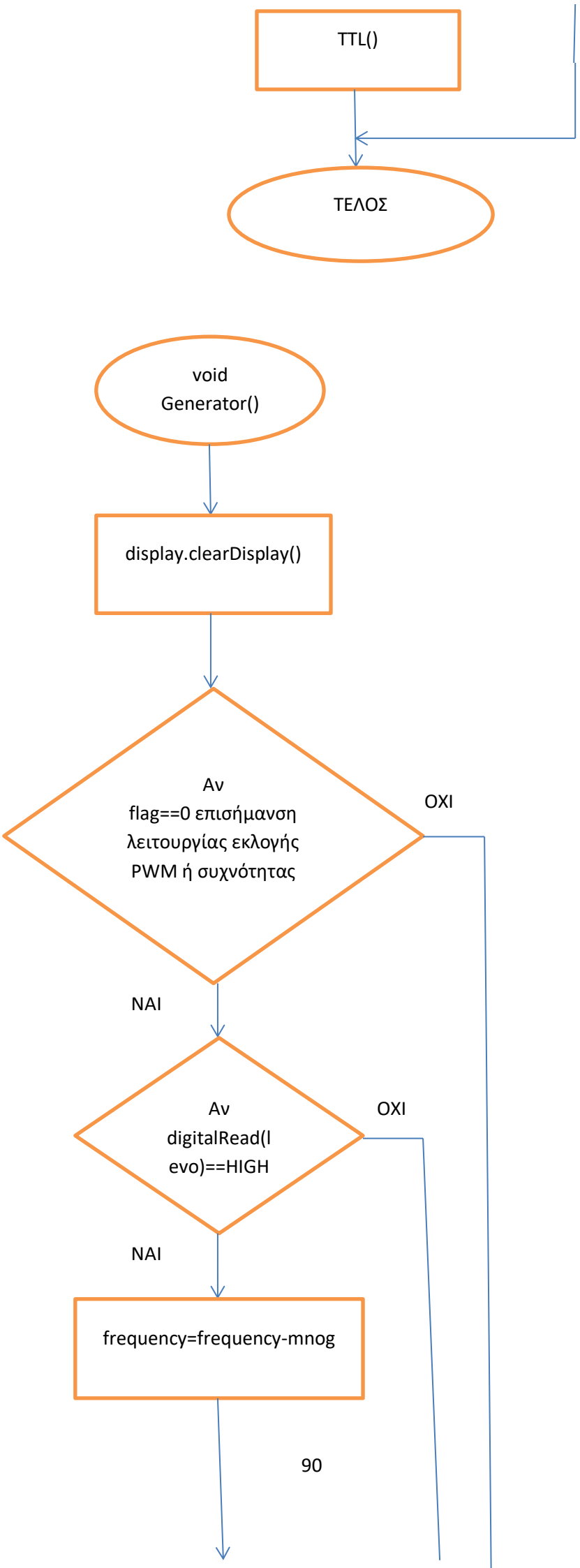

Av
razv==5
OXI
NAI
Frec=1000000/((y-
Frec1-1)*3.27)/2
Av
razv==4
OXI
NAI
Frec=1000000/((y-
Frec1-1)*3.27)/4
Av
razv==3
OXI
NAI
Frec=1000000/((y-
Frec1-1)*3.27)/8
Av
razv==2
OXI
NAI
Frec=1000000/((y-
Frec1-1)*3.27)/16
Av
razv==2
OXI
NAI
Frec=1000000/((y-
Frec1-1)*3.27)/16

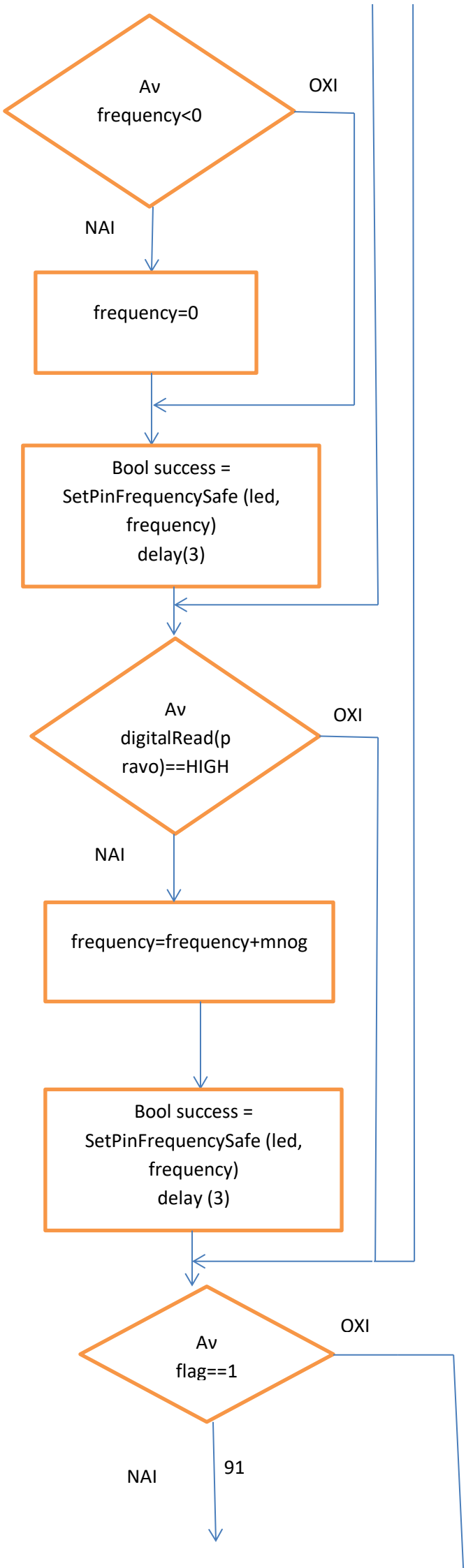

Av
razv==1
OXI
NAI
Frec=1000000/((y-
Frec1-1)*3.27)/32
Av
razv==0
OXI
NAI
Frec=1000000/((y-
Frec1-1)*500
flagFrec1=0;flagFrec3=0
display.setTextColor(BLACK)
Av
opornoe==1
OXI
NAI
Av
Vmax*VCC/2
55)>2.5
OXI
NAI
countX=count*(overclock/16.0)

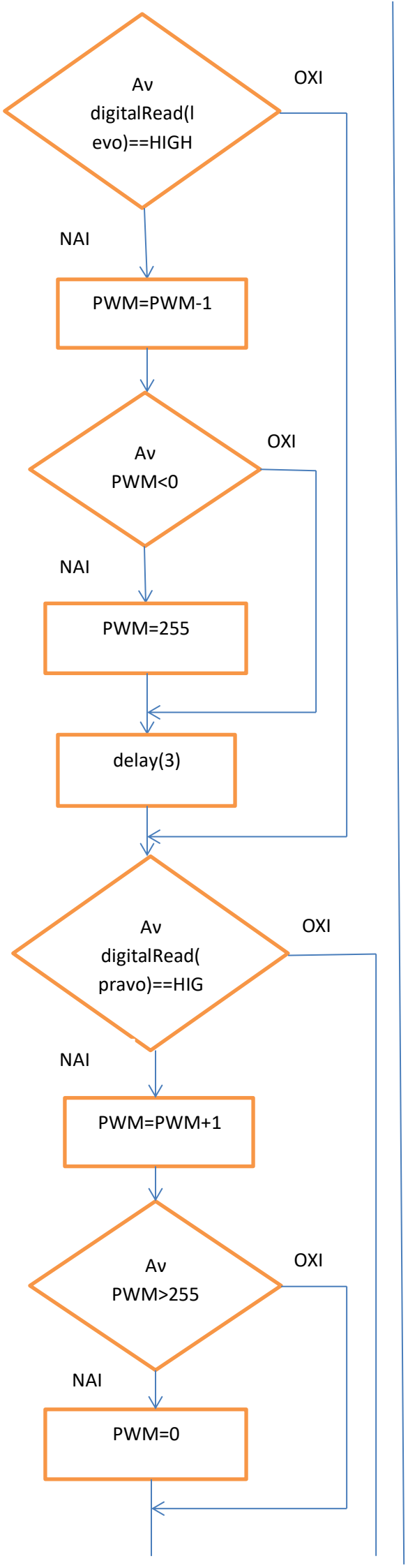

Av
Vmax*VCC/25
5)<2.5
OXI
NAI
countX=Frec*(overclock/16.0)
Av
opornoe==0
OXI
NAI
countX=Frec*(overclock/16.0)
Av
countX<1000
OXI
NAI
display.print (" ");
display.print(countX)
display.print("Hz")

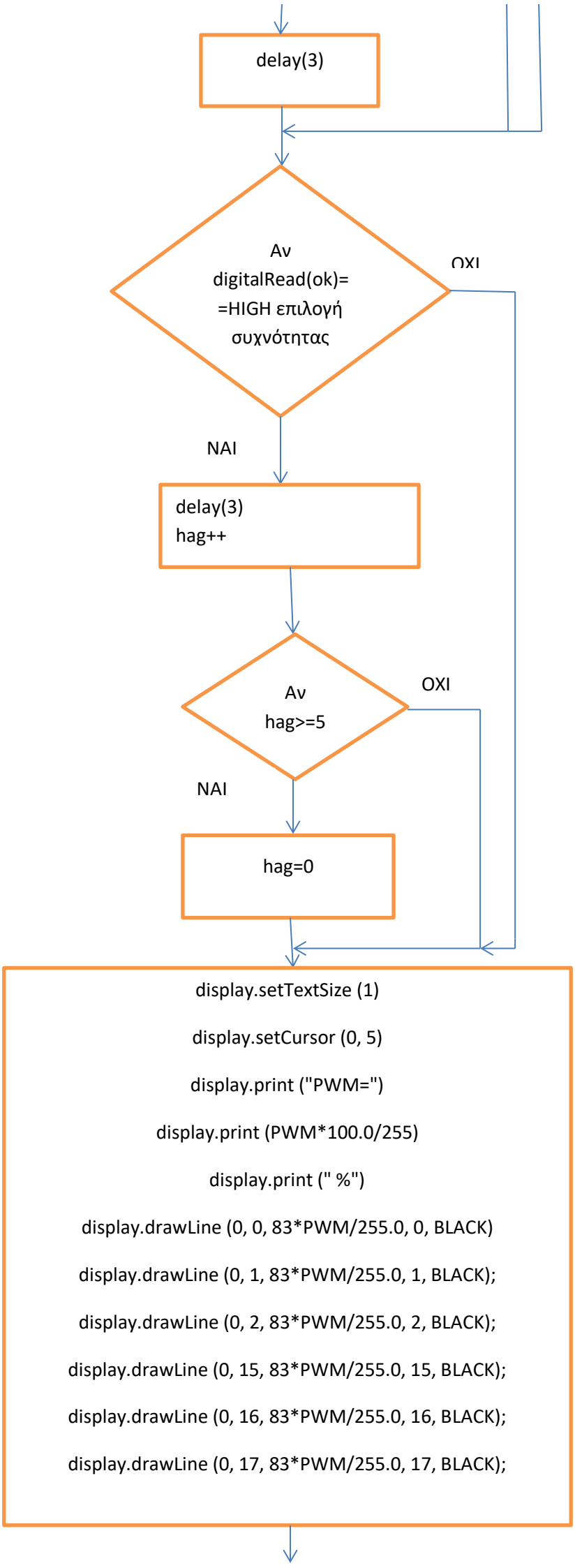

Av
countX>1000
OXI
NAI
Float
countXK=countX/1000.0
display.print (countXK, 1)
display.print("KHz")
Av
opornoe==1
OXI
NAI
display.setCursor (0, 40)
display.setTextColor(BLACK)
display.print(Vmax*VCC/255,1)
Av
opornoe==0
OXI
NAI
display.setCursor (0, 40)
display.setTextColor(BLACK)
display.print (Vmax*1.1/255, 1)
88

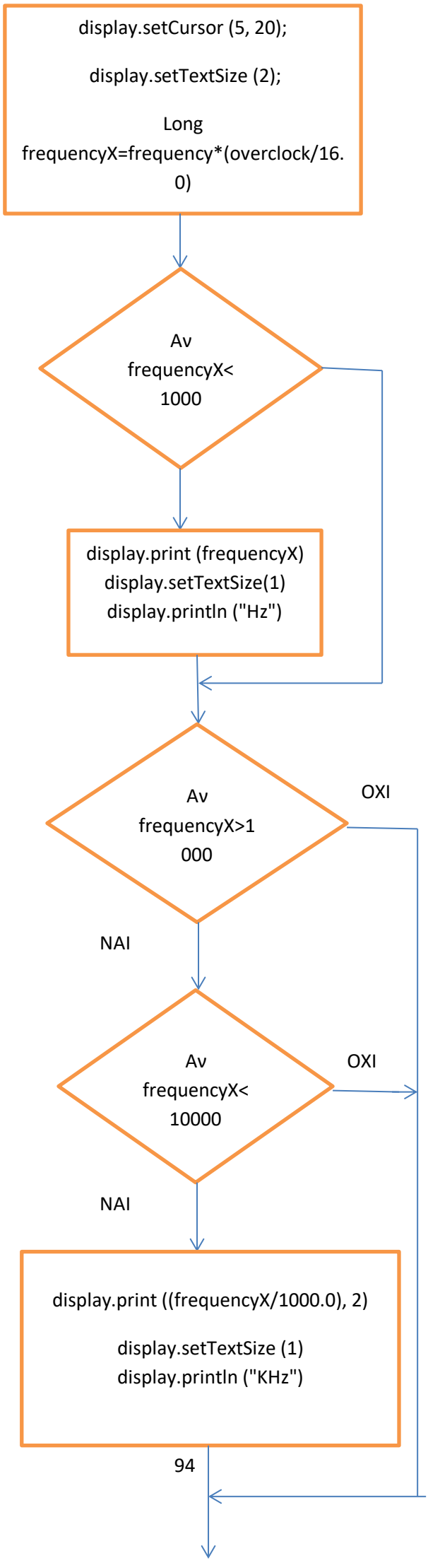

display.print("V")
μενού απόδοσης
delay(200)
display.display()
Αν
pultoskop==1
ΟΧΙ
ΝΑΙ
Generator()
Αν
pultoskop==2
ΟΧΙ
ΝΑΙ
DDSGenerator()
Αν
pultoskop==3
ΟΧΙ
ΝΑΙ

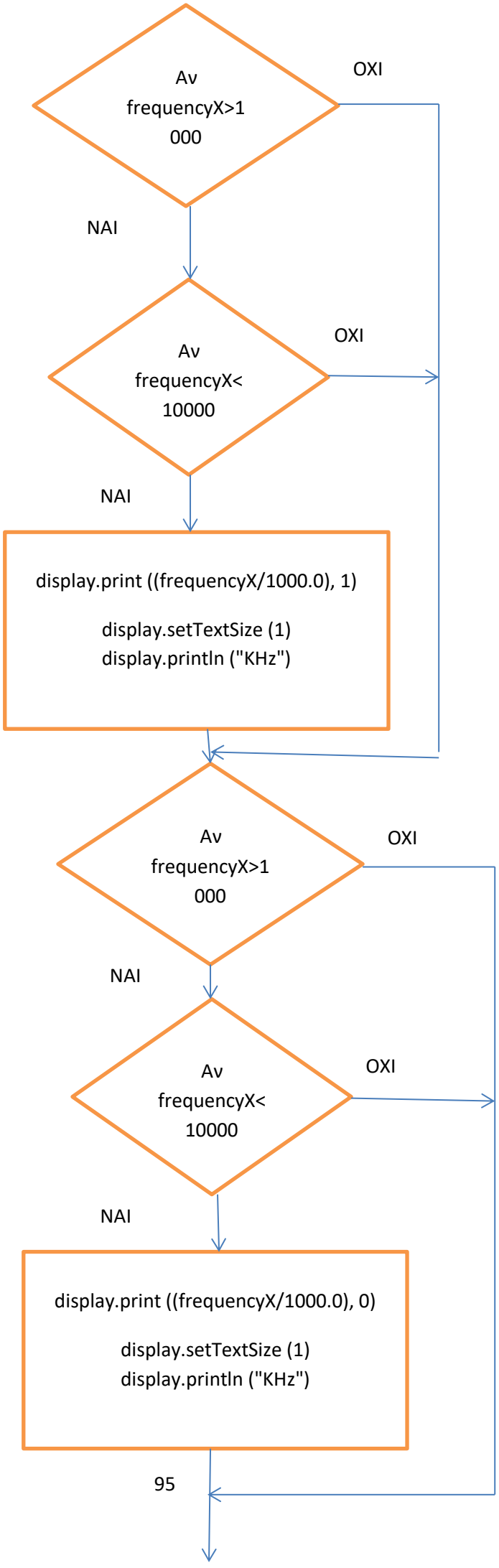

TTL()
ΤΕΛΟΣ
void
Generator()
display.clearDisplay()
Αν
flag==0 επισήμανση
λειτουργίας εκλογής
PWM ή συχνότητας
ΟΧΙ
ΝΑΙ
Αν
digitalRead(l
evo)==HIGH
ΟΧΙ
ΝΑΙ
frequency=frequency-mnog

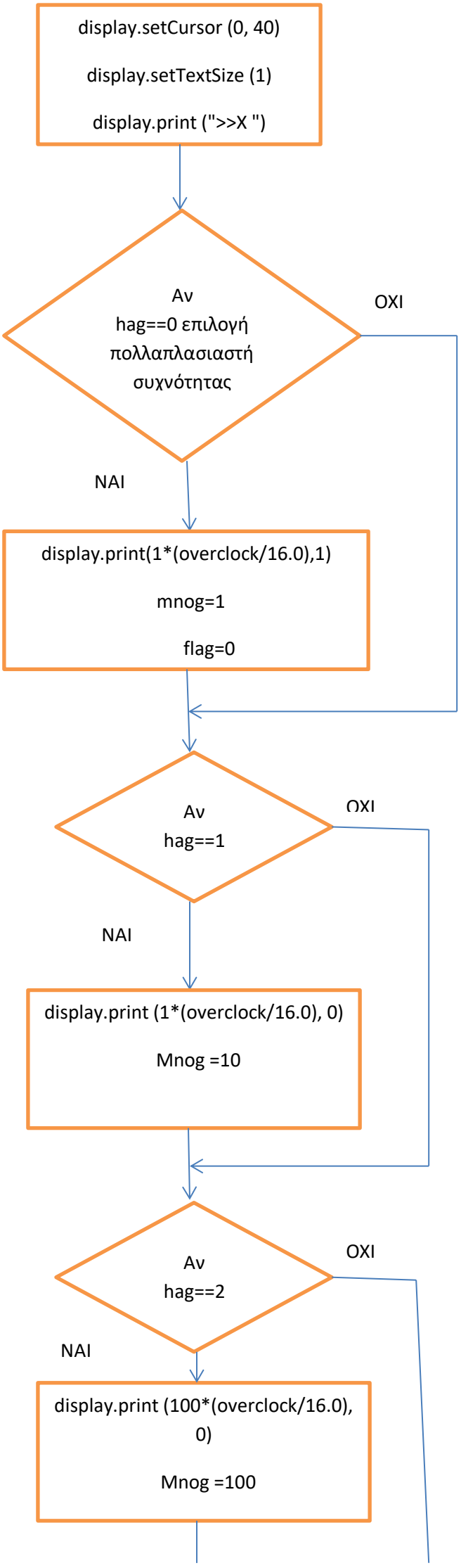

Av
frequency<0
OXI
NAI
frequency=0
Bool success =
SetPinFrequencySafe (led,
frequency)
delay(3)
Av
digitalRead(p
ravo)==HIGH
OXI
NAI
frequency=frequency+mnog
Bool success =
SetPinFrequencySafe (led,
frequency)
delay (3)
Av
flag==1
OXI
NAI
91

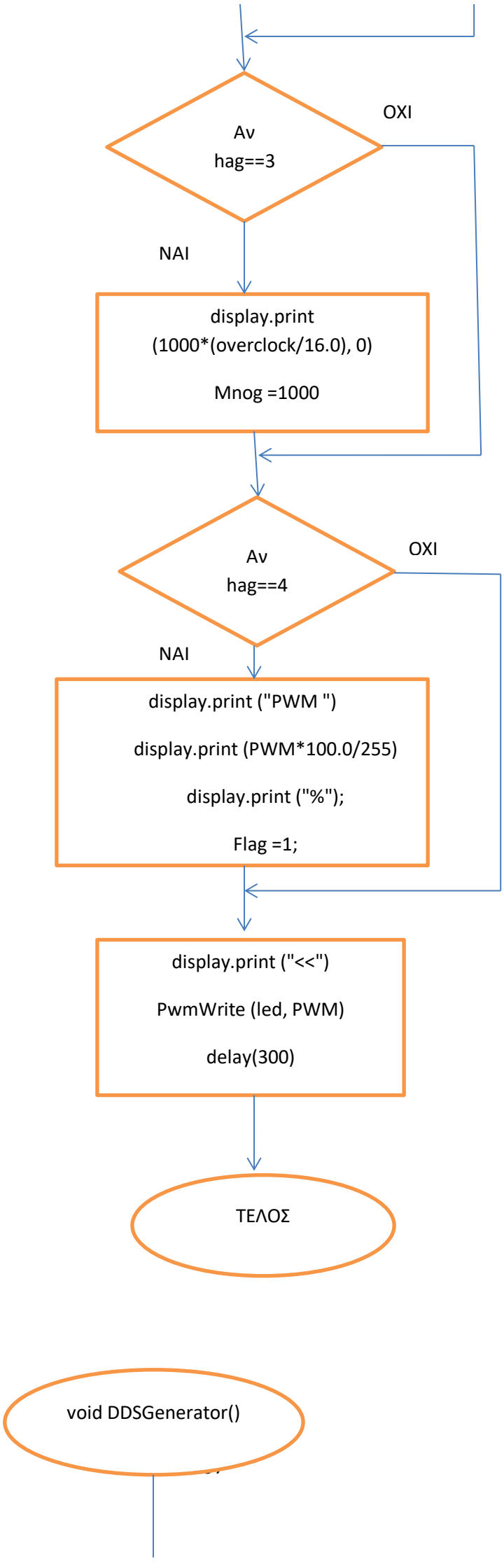

Av digitalRead(levo)==HIGH
OXI
NAI
PWM=PWM-1
Av PWM<0
OXI
NAI
PWM=255
delay(3)
Av digitalRead(pravo)==HIG
OXI
NAI
PWM=PWM+1
Av PWM>255
OXI
NAI
PWM=0

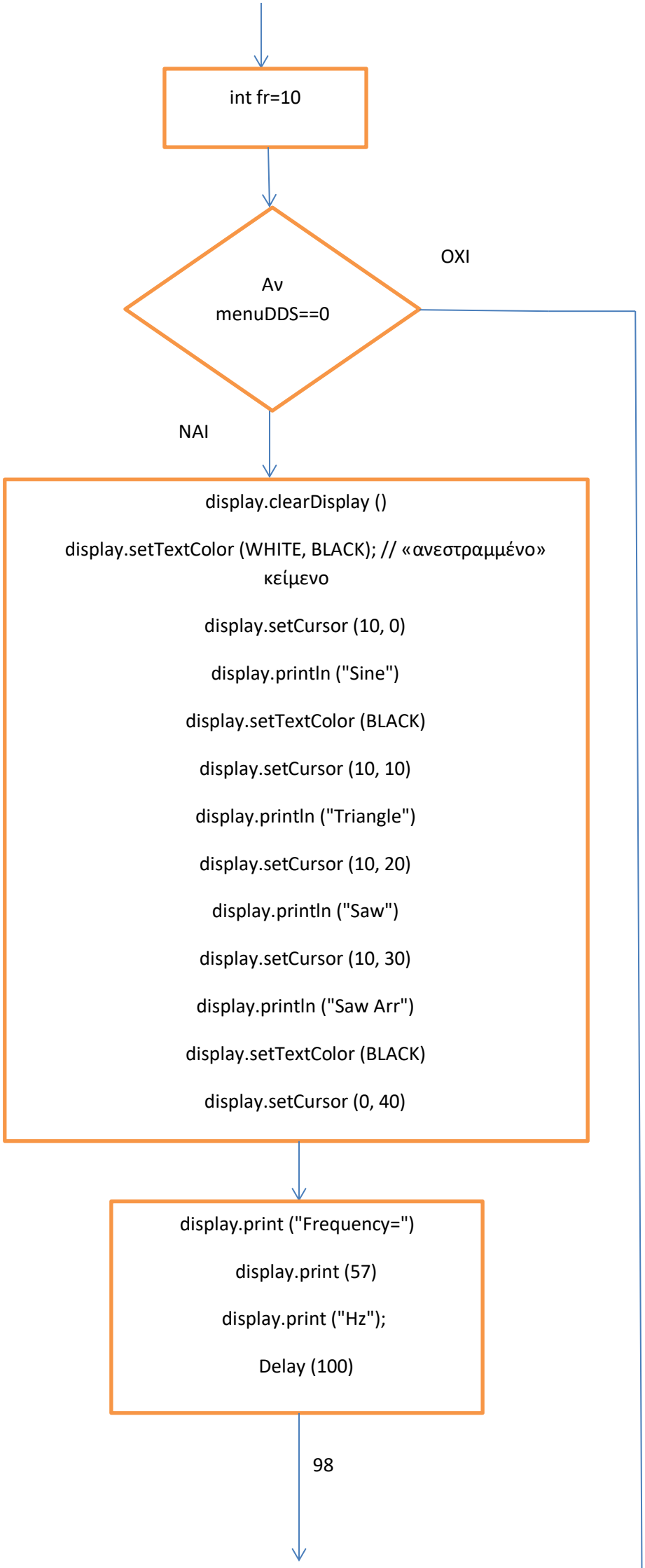

delay(3)
Αν
digitalRead(ok)=
=HIGH επιλογή
συχνότητας
ΟΧΙ
ΝΑΙ
delay(3)
hag++
Αν
hag>=5
ΟΧΙ
ΝΑΙ
hag=0
display.setTextSize (1)
display.setCursor (0, 5)
display.print ("PWM=")
display.print (PWM*100.0/255)
display.print (" %")
display.drawLine (0, 0, 83*PWM/255.0, 0, BLACK)
display.drawLine (0, 1, 83*PWM/255.0, 1, BLACK);
display.drawLine (0, 2, 83*PWM/255.0, 2, BLACK);
display.drawLine (0, 15, 83*PWM/255.0, 15, BLACK);
display.drawLine (0, 16, 83*PWM/255.0, 16, BLACK);
display.drawLine (0, 17, 83*PWM/255.0, 17, BLACK);

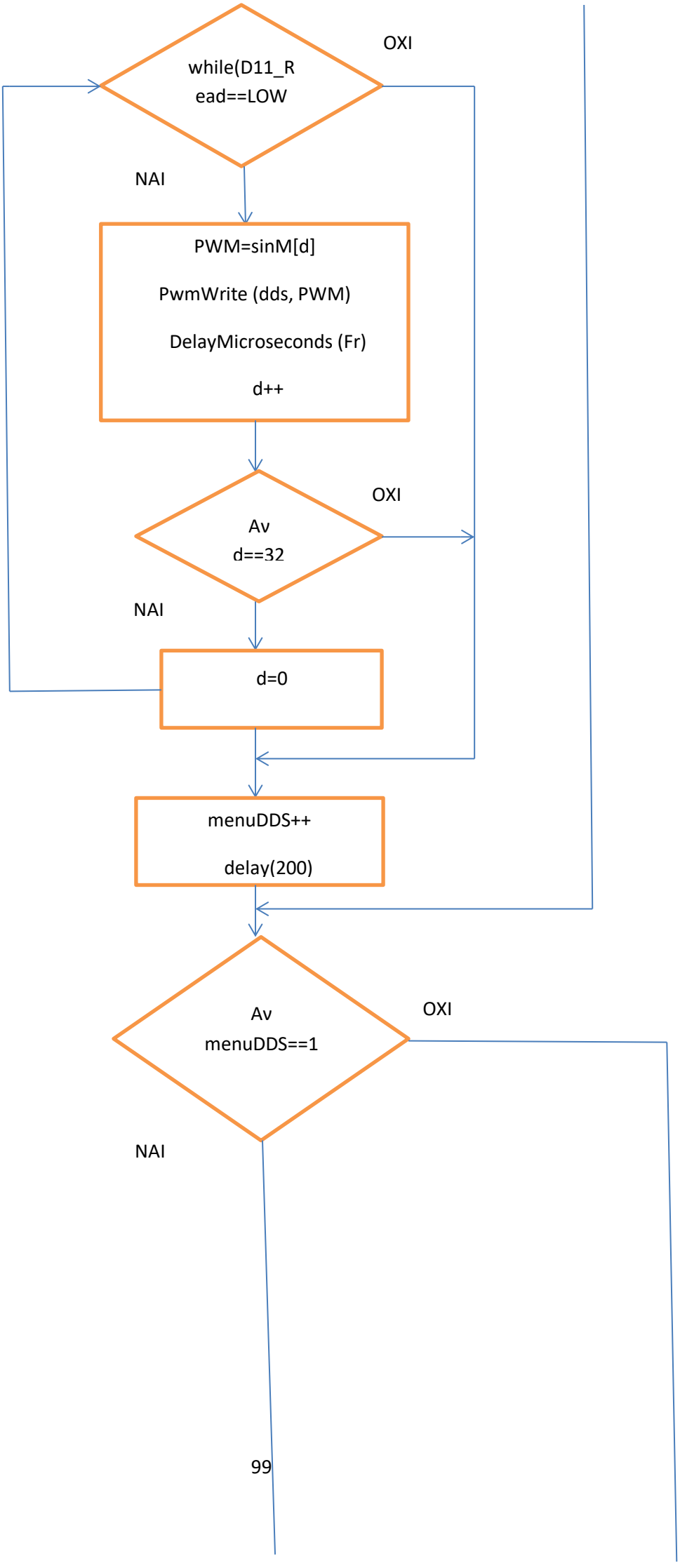

display.setCursor (5, 20);
display.setTextSize (2);
Long frequencyX=frequency*(overclock/16.0)
Av frequencyX< 1000
display.print (frequencyX)
display.setTextSize(1)
display.println ("Hz")
Av frequencyX>1000
OXI
NAI
Av frequencyX< 10000
OXI
NAI
display.print ((frequencyX/1000.0), 2)
display.setTextSize (1)
display.println ("KHz")
94

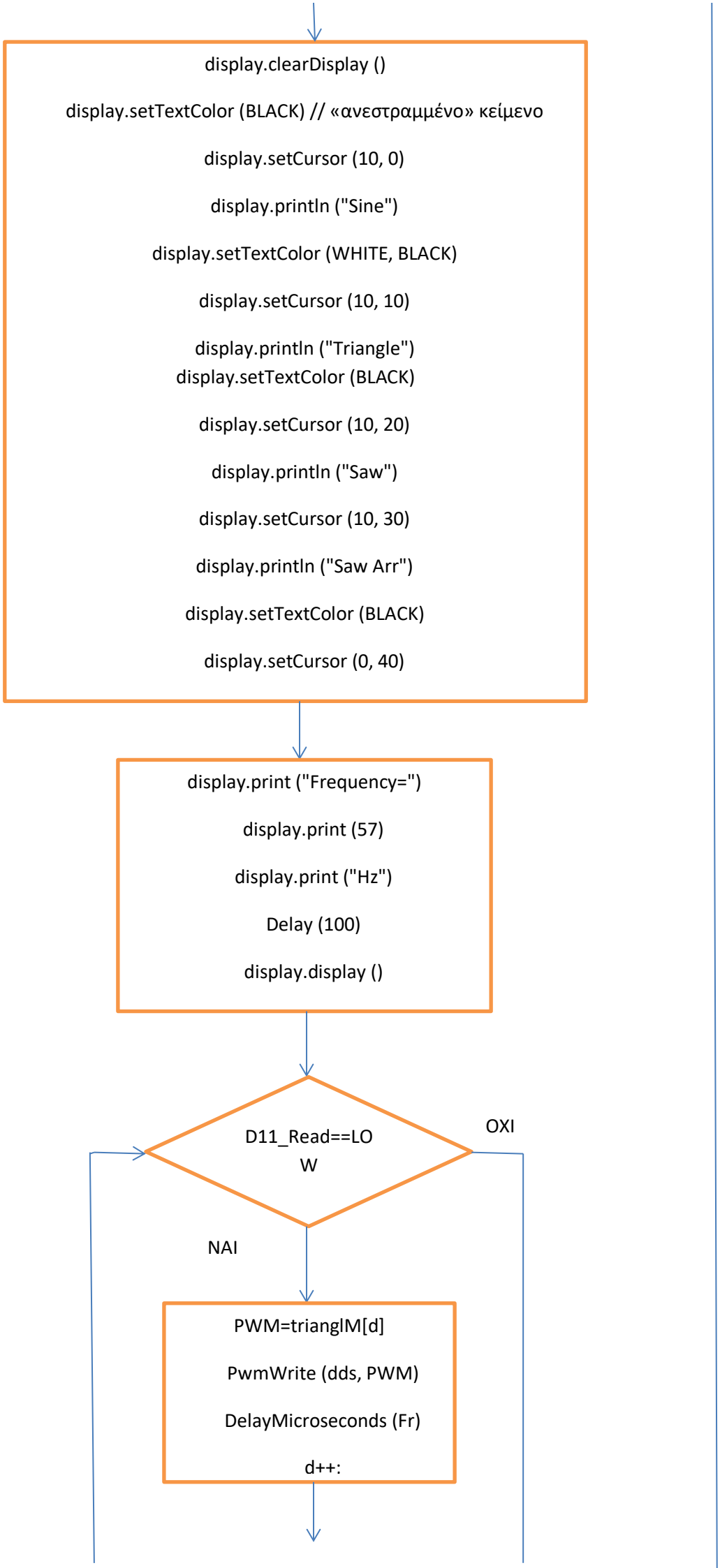

Av
frequencyX>1
000
OXI
NAI
Av
frequencyX<
10000
OXI
NAI
display.print ((frequencyX/1000.0), 1)
display.setTextSize (1)
display.println ("KHz")
Av
frequencyX>1
000
OXI
NAI
Av
frequencyX<
10000
OXI
NAI
display.print ((frequencyX/1000.0), 0)
display.setTextSize (1)
display.println ("KHz")
95

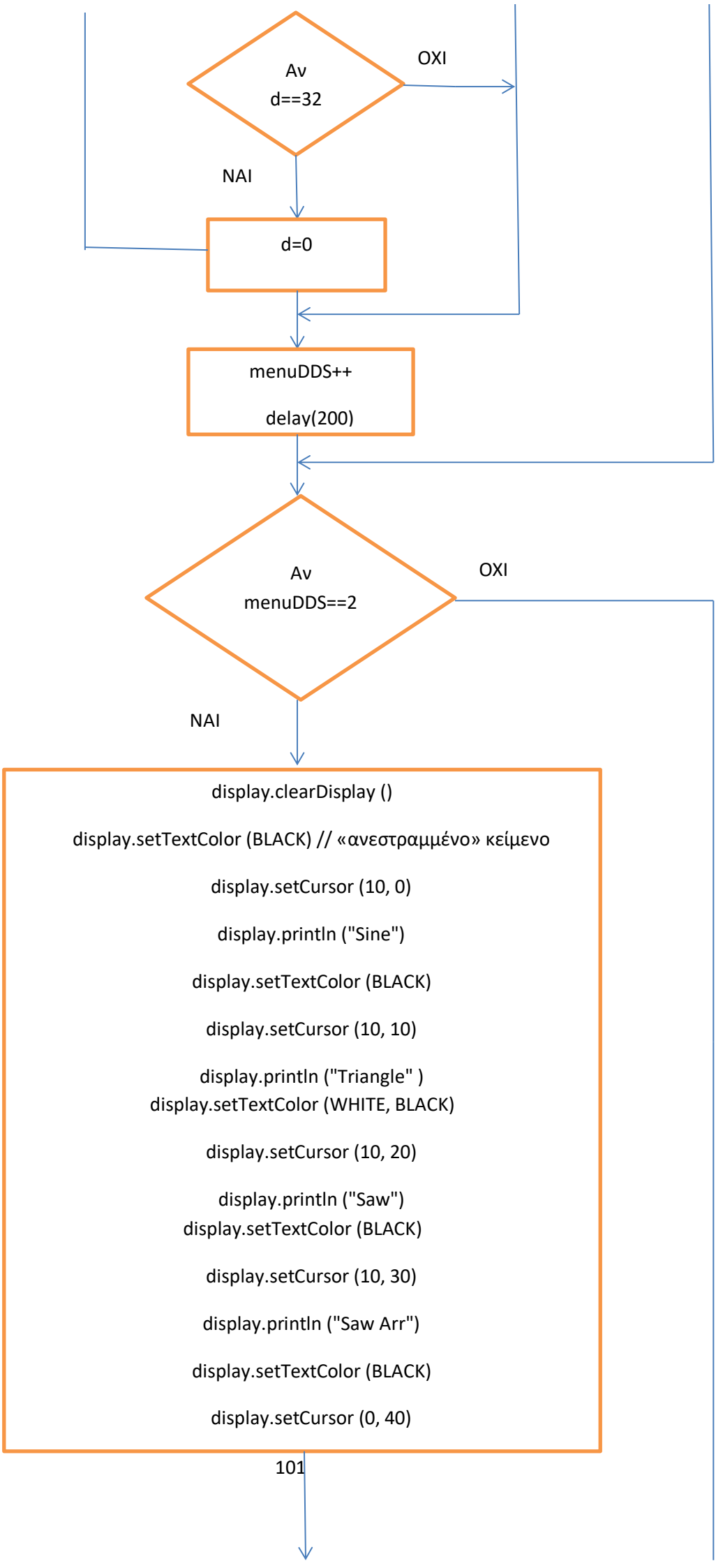

display.setCursor (0, 40)
display.setTextSize (1)
display.print (">>X ")
Αν
hag==0 επιλογή
πολλαπλασιαστή
συχνότητας
ΟΧΙ
ΝΑΙ
display.print(1*(overclock/16.0),1)
mnog=1
flag=0
Αν
hag==1
ΟΧΙ
ΝΑΙ
display.print (1*(overclock/16.0), 0)
Mnog =10
Αν
hag==2
ΟΧΙ
ΝΑΙ
display.print (100*(overclock/16.0),
0)
Mnog =100

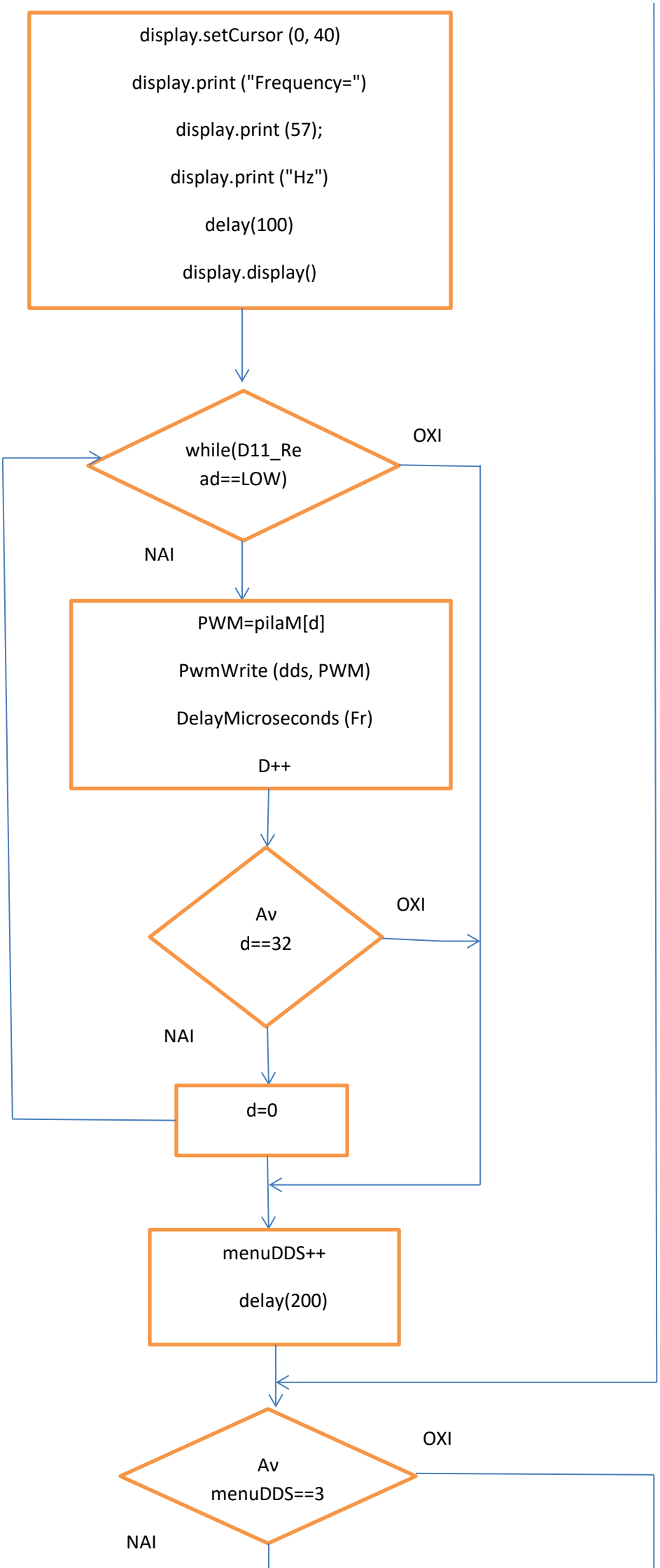

Av
hag==3
OXI
NAI
display.print
(1000*(overclock/16.0), 0)
Mnog =1000
Av
hag==4
OXI
NAI
display.print ("PWM ")
display.print (PWM*100.0/255)
display.print ("%");
Flag =1;
display.print ("<<")
PwmWrite (led, PWM)
delay(300)
ΤΕΛΟΣ
void DDSGenerator()

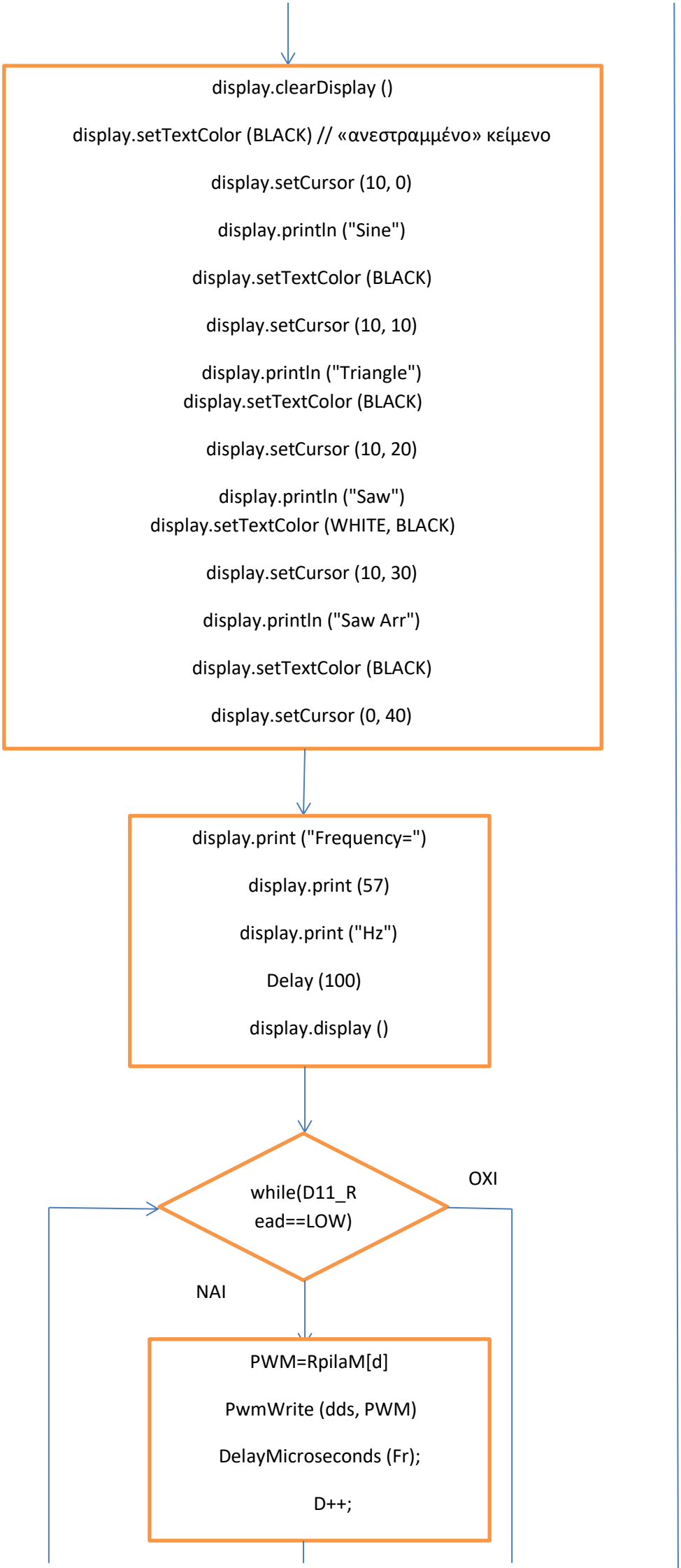

int fr=10
Αν
menuDDS==0
ΟΧΙ
ΝΑΙ
display.clearDisplay ()
display.setTextColor (WHITE, BLACK); // «ανεστραμμένο» κείμενο
display.setCursor (10, 0)
display.println ("Sine")
display.setTextColor (BLACK)
display.setCursor (10, 10)
display.println ("Triangle")
display.setCursor (10, 20)
display.println ("Saw")
display.setCursor (10, 30)
display.println ("Saw Arr")
display.setTextColor (BLACK)
display.setCursor (0, 40)
display.print ("Frequency=")
display.print (57)
display.print ("Hz");
Delay (100)
98

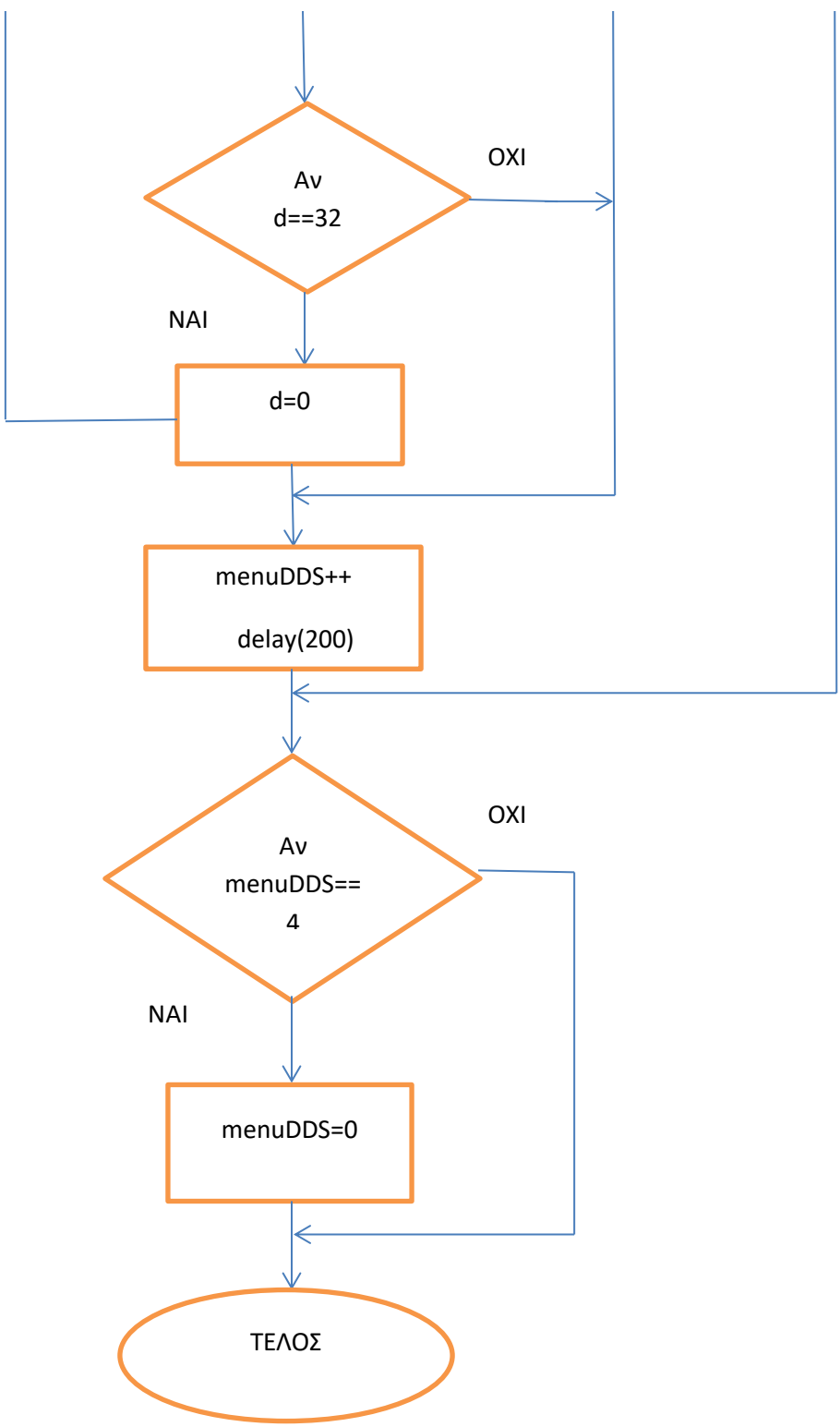

while(D11_R
ead==LOW
OXI
NAI
PWM=sinM[d]
PwmWrite (dds, PWM)
DelayMicroseconds (Fr)
d++
Av
d==32
OXI
NAI
d=0
menuDDS++
delay(200)
Av
menuDDS==1
OXI
NAI
99

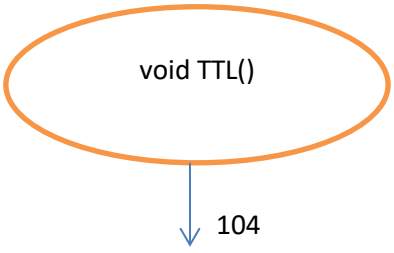

display.clearDisplay ()
display.setTextColor (BLACK) // «ανεστραμμένο» κείμενο
display.setCursor (10, 0)
display.println ("Sine")
display.setTextColor (WHITE, BLACK)
display.setCursor (10, 10)
display.println ("Triangle")
display.setTextColor (BLACK)
display.setCursor (10, 20)
display.println ("Saw")
display.setCursor (10, 30)
display.println ("Saw Arr")
display.setTextColor (BLACK)
display.setCursor (0, 40)
display.print ("Frequency=")
display.print (57)
display.print ("Hz")
Delay (100)
display.display ()
D11_Read==LOW
OXI
NAI
PWM=trianglM[d]
PwmWrite (dds, PWM)
DelayMicroseconds (Fr)
d++:

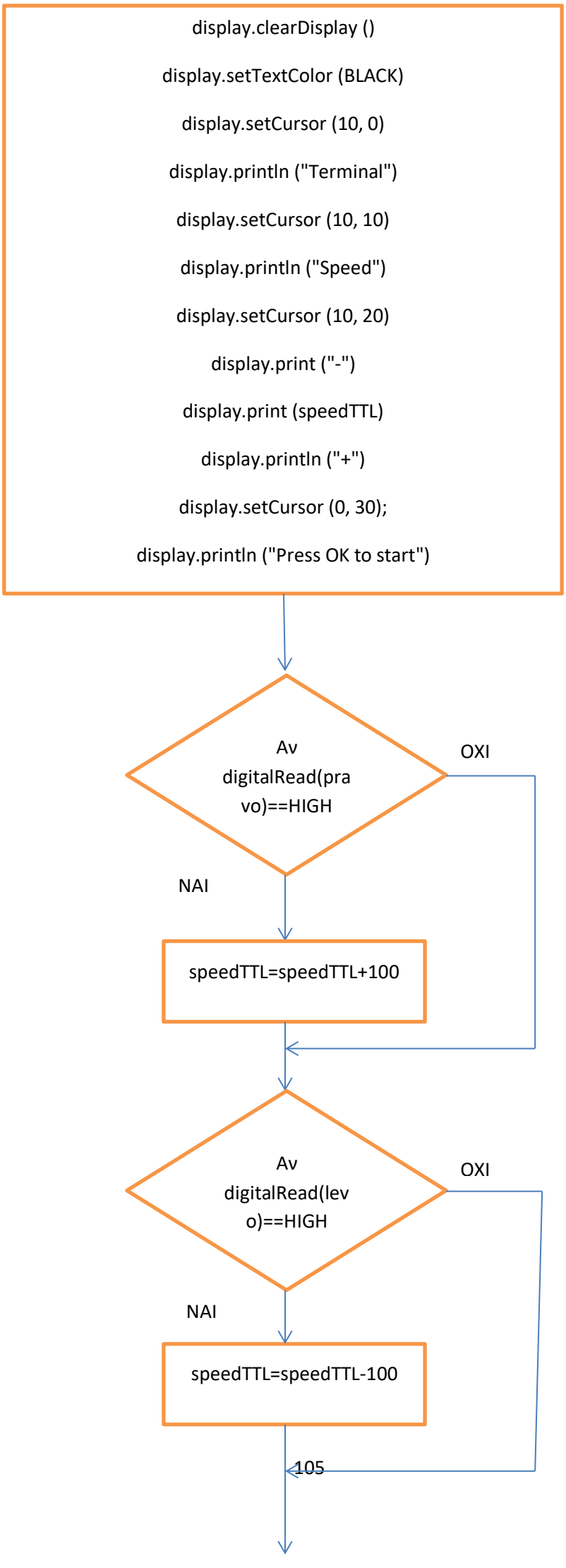

Αν
d==32
ΟΧΙ
ΝΑΙ
d=0
menuDDS++
delay(200)
Αν
menuDDS==2
ΟΧΙ
ΝΑΙ
display.clearDisplay ()
display.setTextColor (BLACK) // «ανεστραμμένο» κείμενο
display.setCursor (10, 0)
display.println ("Sine")
display.setTextColor (BLACK)
display.setCursor (10, 10)
display.println ("Triangle")
display.setTextColor (WHITE, BLACK)
display.setCursor (10, 20)
display.println ("Saw")
display.setTextColor (BLACK)
display.setCursor (10, 30)
display.println ("Saw Arr")
display.setTextColor (BLACK)
display.setCursor (0, 40)
101

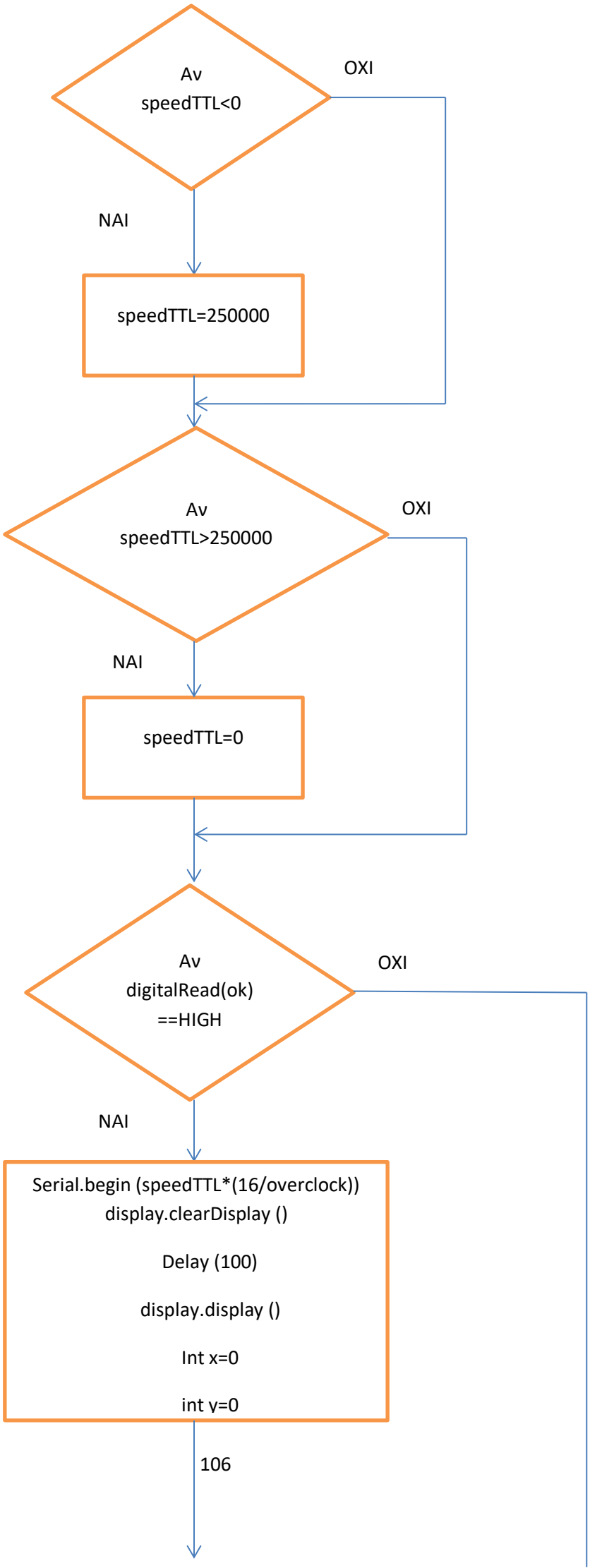

display.setCursor (0, 40)
display.print ("Frequency=")
display.print (57);
display.print ("Hz")
delay(100)
display.display()
while(D11_Re ad==LOW)
OXI
NAI
PWM=pilaM[d]
PwmWrite (dds, PWM)
DelayMicroseconds (Fr)
D++
Av d==32
OXI
NAI
d=0
menuDDS++
delay(200)
Av menuDDS==3
OXI
NAI

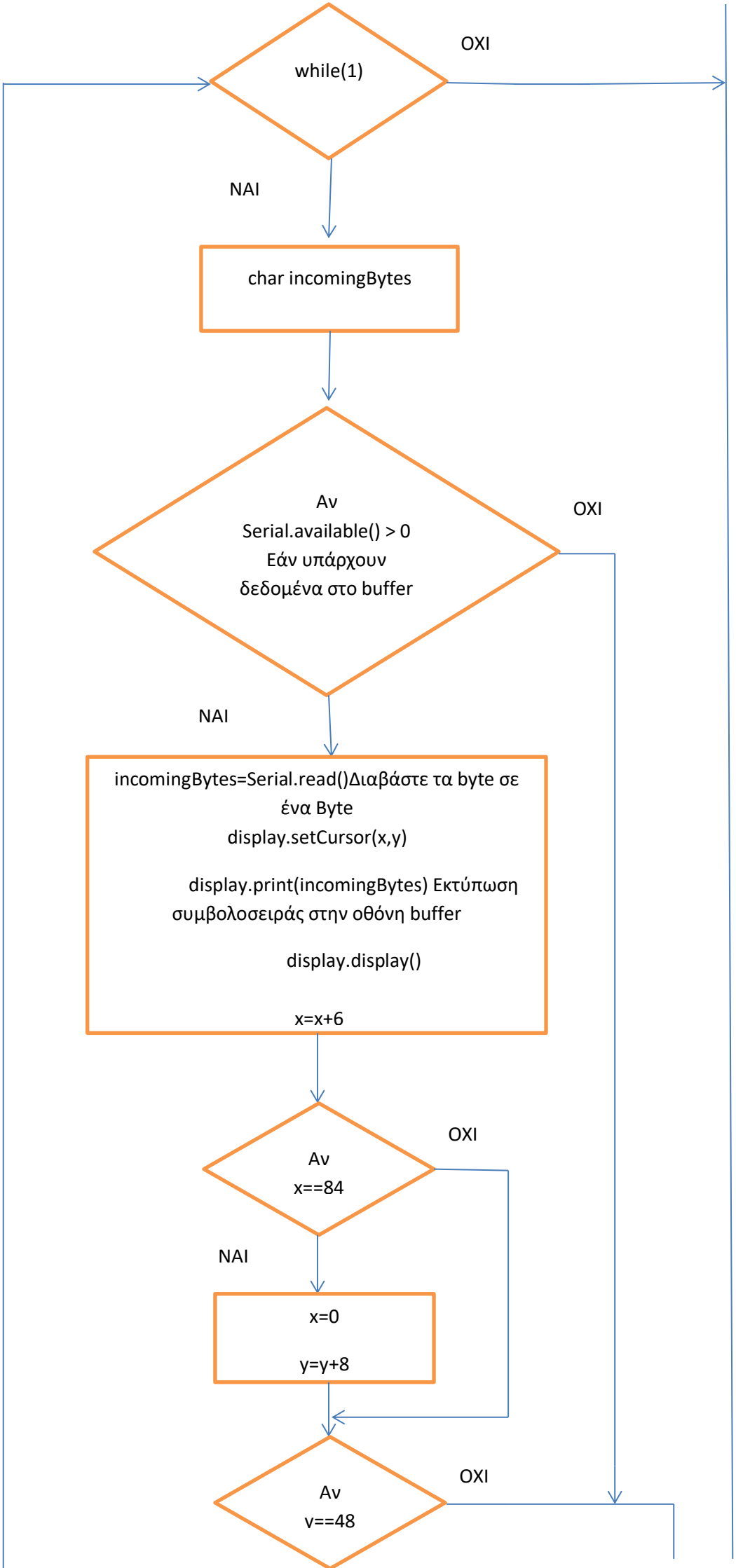

display.clearDisplay ()
display.setTextColor (BLACK) // «ανεστραμμένο» κείμενο
display.setCursor (10, 0)
display.println ("Sine")
display.setTextColor (BLACK)
display.setCursor (10, 10)
display.println ("Triangle")
display.setTextColor (BLACK)
display.setCursor (10, 20)
display.println ("Saw")
display.setTextColor (WHITE, BLACK)
display.setCursor (10, 30)
display.println ("Saw Arr")
display.setTextColor (BLACK)
display.setCursor (0, 40)
display.print ("Frequency=")
display.print (57)
display.print ("Hz")
Delay (100)
display.display ()
while(D11_Read==LOW)
OXI
NAI
PWM=RpilaM[d]
PwmWrite (dds, PWM)
DelayMicroseconds (Fr);
D++;

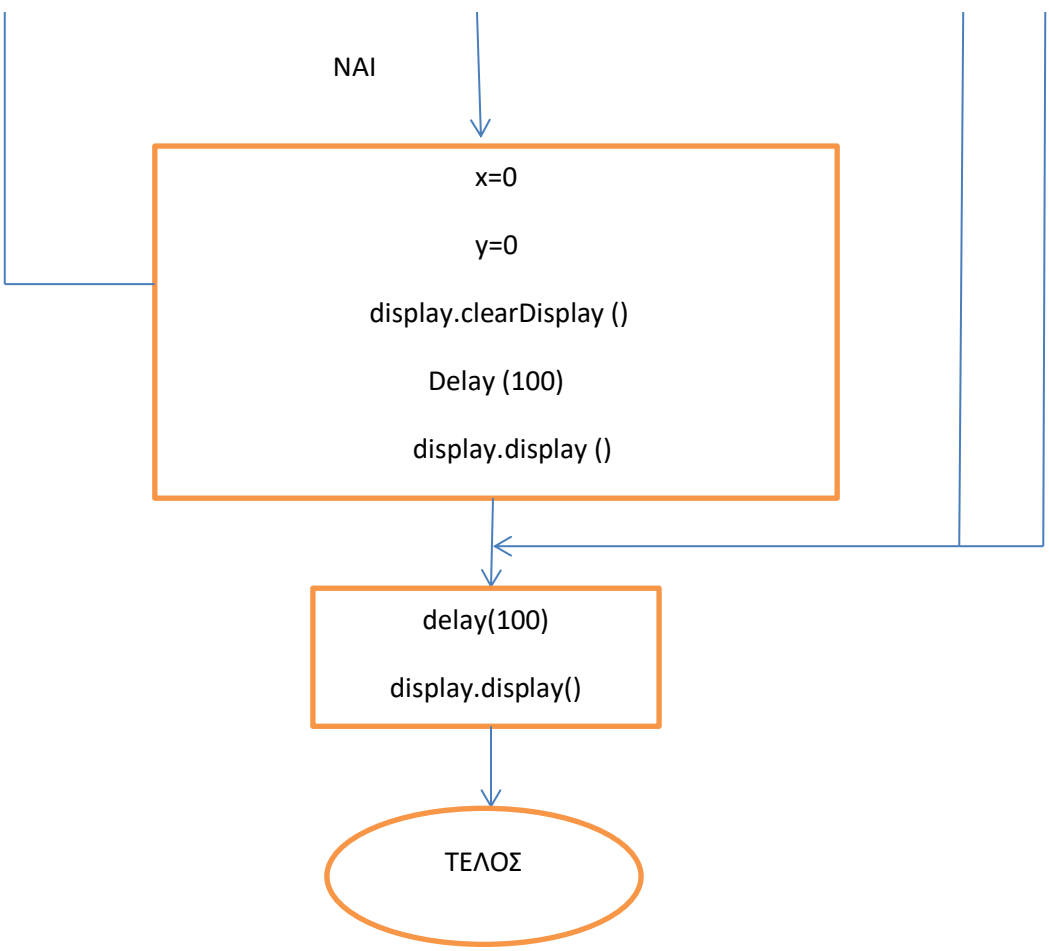

Αν
d==32
ΟΧΙ
ΝΑΙ
d=0
menuDDS++
delay(200)
Αν
menuDDS==
4
ΟΧΙ
ΝΑΙ
menuDDS=0
ΤΕΛΟΣ


void TTL()
104


display.clearDisplay ()
display.setTextColor (BLACK)
display.setCursor (10, 0)
display.println ("Terminal")
display.setCursor (10, 10)
display.println ("Speed")
display.setCursor (10, 20)
display.print ("-")
display.print (speedTTL)
display.println ("+")
display.setCursor (0, 30);
display.println ("Press OK to start")
Av digitalRead(pravo)==HIGH
OXI
NAI
speedTTL=speedTTL+100
Av digitalRead(levo)==HIGH
OXI
NAI
speedTTL=speedTTL-100
105

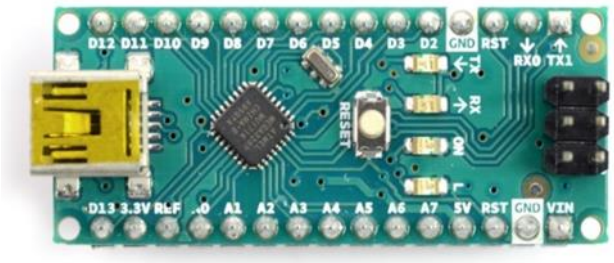

Av
speedTTL<0
OXI
NAI
speedTTL=250000
Av
speedTTL>250000
OXI
NAI
speedTTL=0
Av
digitalRead(ok)
==HIGH
OXI
NAI
Serial.begin (speedTTL*(16/overclock))
display.clearDisplay ()
Delay (100)
display.display ()
Int x=0
int y=0
106


while(1)
ΟΧΙ
ΝΑΙ
char incomingBytes
Αν
Serial.available() > 0
Εάν υπάρχουν
δεδομένα στο buffer
ΟΧΙ
ΝΑΙ
incomingBytes=Serial.read()Διαβάστε τα byte σε
ένα Byte
display.setCursor(x,y)
display.print(incomingBytes) Εκτύπωση
συμβολοσειράς στην οθόνη buffer
display.display()
x=x+6
Αν
x==84
ΟΧΙ
ΝΑΙ
x=0
y=y+8
Αν
v==48
ΟΧΙ


NAI
x=0
y=0
display.clearDisplay ()
Delay (100)
display.display ()
delay(100)
display.display()
ΤΕΛΟΣ

5.4 Υλικά κατασκευής

Arduino NANO: Η πιο μικρή πλακέτα της πλατφόρμας Arduino. Το συγκεκριμένο μοντέλο (σχήμα 5.4), όπως και τα άλλα της οικογένειας Arduino είναι πολύ διαδεδομένο και συμβατό με πλήθος αισθητήρων και επεκτάσεων. Η πλακέτα Arduino Nano δεν παρουσιάζει καμία διαφορά στη λειτουργία της με την πλακέτα Arduino Uno, απλά είναι μικρότερη σε μέγεθος και είναι πλήρως συμβατή ώστε να μπορεί να τοποθετηθεί σε πλακέτες δοκιμών (breadboard). Το Arduino Nano βασίζεται στον μικροελεγκτή ATmega328 της Atmel και πρόκειται για μια ολοκληρωμένη πλακέτα η οποία έχει όλα όσα χρειάζεται για να μπορεί να προγραμματιστεί και να λειτουργήσει μόλις συνδεθεί στον ηλεκτρονικό υπολογιστή, με ένα απλό καλώδιο Mini-B USB.



Σχήμα 5.4: Arduino nano.

Nokia5110 LCD:

Χαρακτηριστικά της μονάδας LCD Nokia5110 (σχήμα 5.4.1):

- Η τάση λειτουργίας είναι 2,7V έως 3,3V.
- Η τρέχουσα κατανάλωση είναι 6mA.
- Αποτελείται από 84 σειρές και 84 στήλες (84 × 48) μονόχρωμα εικονοστοιχεία.
- Λειτουργεί χρησιμοποιώντας διεπαφή SPI.
- Αποτελείται από τσιπ διασύνδεσης Philips PCD8544 για εύκολη διασύνδεση.
- Υποστηρίζει αξιοπρεπή γραφικά εικόνων bitmap.
- Διατίθεται σε πράσινο και μπλε οπίσθιο φωτισμό.

Σχήμα 5.4.1 Nokia 5110 LCD

Πυκνωτής: Είναι ένα εξάρτημα που χρησιμεύει ως αποθήκη ηλεκτρικού φορτίου και επομένως ηλεκτρικής ενέργειας. Αποτελείται από δύο αγωγούς που διαχωρίζονται από ένα μονωτικό υλικό. Μονάδα μέτρησης της χωρητικότητας του πυκνωτή είναι το 1 Φαράντ Farad (F). Πρόκειται όμως για μεγάλη μονάδα, που σπάνια χρησιμοποιείται στην πράξη. Συνήθως χρησιμοποιούνται τα υποπολλαπλάσια του μικροφαράντ (μF), νανοφαράντ (nF) και πικοφαράντ (pF). Στην δική μας περίπτωση χρησημοποιούμε έναν πυκνωτή 10 μF και έναν 0,1μF(σχήματα 1.1, 1.2 αντίστοιχα).

Σχήμα 1.1:πυκνωτής 10 μF

Σχήμα 1.2:πυκνωτής 0,1μF

Ηλεκτρική αντίσταση: Η αντίσταση γενικά, όπως το λέει και η λέξη, μειώνει την τάση φέρνοντας αντίσταση, όταν περνάει από μέσα της η τάση αυτή. Με αποτέλεσμα, να έχουμε την επιθυμητή τάση στο κύκλωμά μας. Αυτό επιτυγχάνεται, ξεχωριστά για κάθε κύκλωμα, η χρήση της σωστής αντίστασης: υπάρχουν πολλών ειδών που άλλες μειώνουν την τάση λιγότερο, κι άλλες περισσότερο.Χρησιμοποιήθηκαν συνολικά 5 αντιστάσεις 10ΚΩ 1/4W 5% ανοχή και μία 270Ω 1/4W 5% ανοχή (σχήματα 1.3 και 1.4 αντίστοιχα).

Σχήμα 1.3:Αντίσταση 10ΚΩ 1/4W 5%

Σχήμα 1.4:Αντίσταση 270Ω 1/4W 5%

5.5 Κόστος κατασκευής

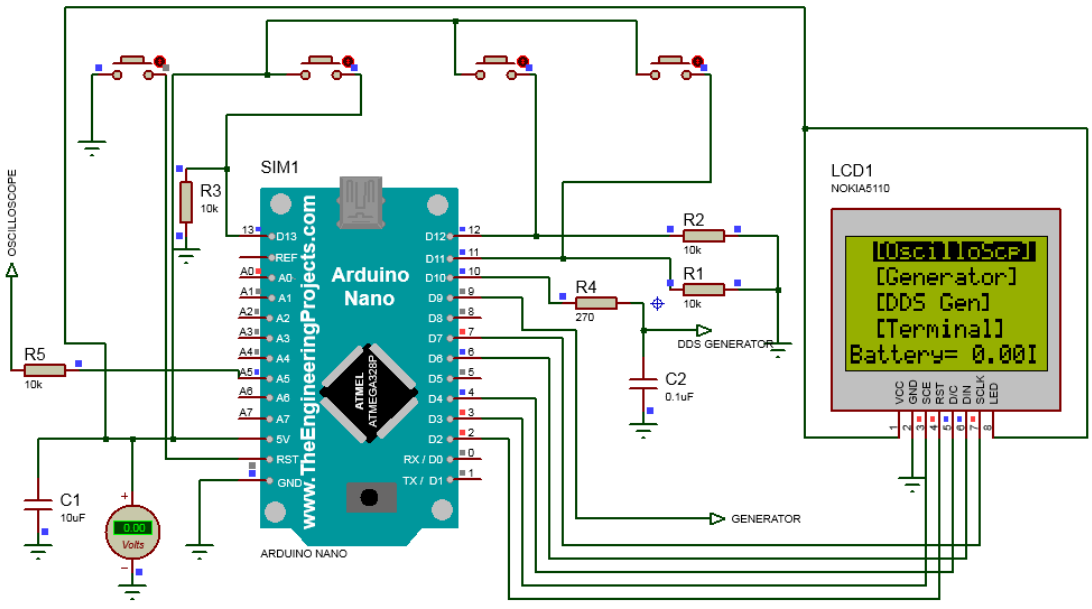
Το κόστος της κατασκευής αναγράφετε αναλυτικά στον παρακάτω πινάκα (5.5).

Εξάρτημα	Τιμή
Arduino NANO	6.00€
Nokia5110 LCD	5.80€
4 button	0.40€
Ηλεκτρικές αντιστάσεις	0.60€
Πυκνωτές	0.20€
Κουτί κατασκευής	1.80€
Μπαταρία 9V	1.00€
Σύνολο	15.80€

Πίνακας 5.5: Κόστος

5.6 Διάγραμμα με το Proteus Design Suite

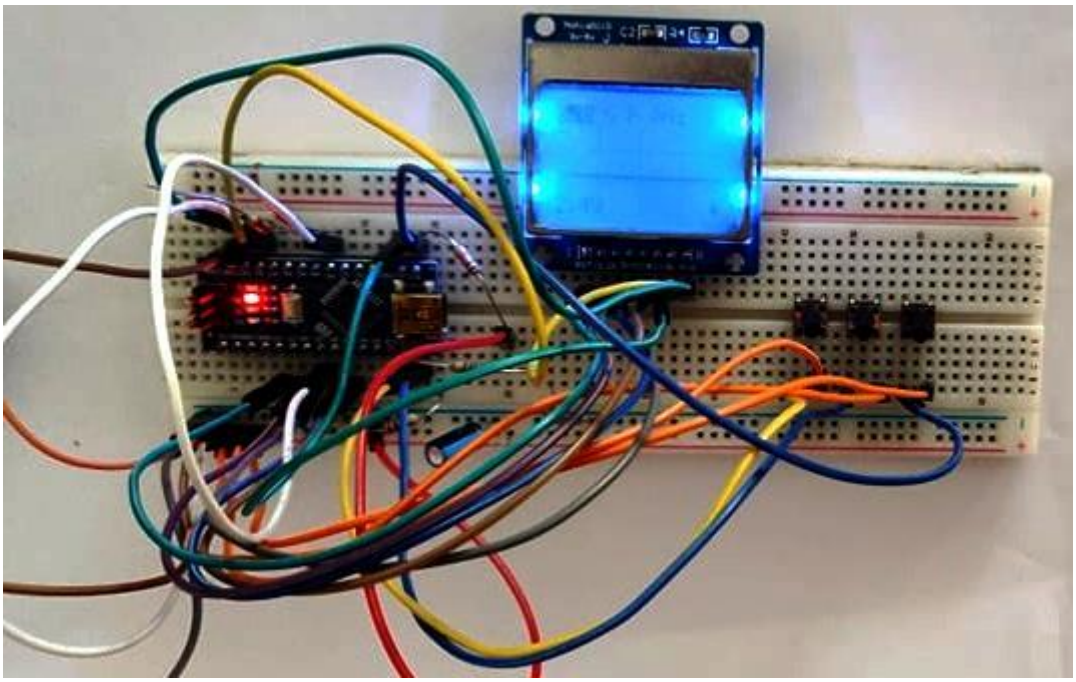
Το **Proteus Design Suite** (5.6) είναι μια ιδιόκτητη σουίτα εργαλείων λογισμικού που χρησιμοποιείται κυρίως για αυτοματοποίηση ηλεκτρονικού σχεδιασμού . Το λογισμικό χρησιμοποιείται κυρίως από μηχανικούς και τεχνικούς ηλεκτρονικού σχεδιασμού για τη δημιουργία σχηματικών και ηλεκτρονικών εκτυπώσεων για την κατασκευή πλακέτων τυπωμένων κυκλωμάτων.



Σχήμα 5.6: Έκδοση Proteus 8

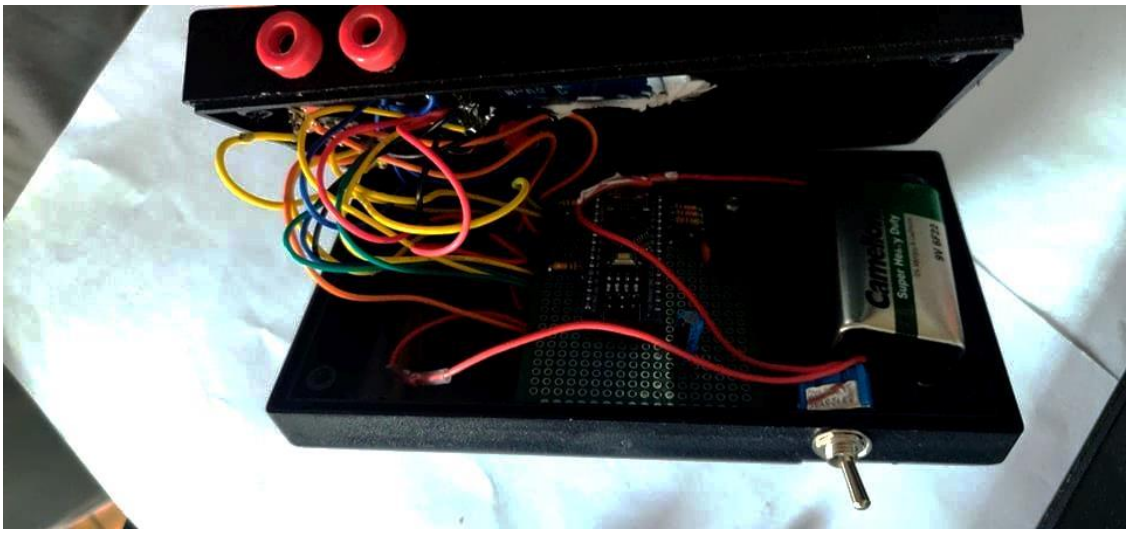
5.7 Η κατασκευή

Αρχικά το κύκλωμα δοκιμάστηκε στο ράστερ όπως φαίνεται παρακάτω (5.7):



Σχήμα 5.7: Η κατασκευή στο ράστερ.

Έπειτα φαίντονται οι συνδέσεις μέσα στο κουτί κατασκευής (5.7.1):



Σχήμα 5.7.1: Οι συνδέσεις μέσα στο κουτί.

Δοκιμή της γεννήτριας (5.7.2):

Σχήμα 5.7.2: Η δοκιμή της γεννήτριας

5.8 Επίλογος

Σε αυτό το κεφάλαιο είδαμε αναλυτικά τον κώδικα και μέσα από το διάγραμμα ροής.Αυτό είναι και το κύριο σημείο του κεφαλαίου ο κώδικας.Τα υλικά κατασκευής είναι χαμηλού κόστους και προσφέρουν μία αξιόπιστη λύση όπως είδαμε και με την δοκιμή της γεννήτριας συνχνότητων που τα αποτελέσματα δεν διαφέρουν σχεδόν καθόλου με την μέτρηση της συχνότητας από το πολύμετρο.

Κεφάλαιο 6ο : Συμπεράσματα

Ο στόχος της παρούσας πτυχιακής ήταν η σχεδίαση και κατασκευή πλατφόρμας ανάπτυξης Arduino Uno και εφαρμογή της σε σχεδίαση μικρού ψηφιακού παλμογράφου.Μετά από αρκετό ψάξιμο στο internet, εφαρμογή βιβλίων και αρκετές συζητήσεις με τον υπεύθυνο καθηγητή κατέληξα στην χρήση του Arduino nano. Αρχικά γιατί με το nano η κατασκευή θα ήταν πολύ πιο μικρή σε μέγεθος σε σχέση με το uno και εγώ ήθελα να βγει το τελικό αποτέλεσμα μικρό σε όγκο για να είναι κινητή συκευή.Το nano μου την έδωσε αυτή την δυνατότητα και ήταν πλήρες τόσο στο μέγεθος όσο και στον προγραμματισμό και την σχεδίαση σε σχέση με το uno για την συγκεκριμένη πτυχιακή εργασία.Τα προβλήματα που αντιμετώπισα ήταν με την κατασκευή.Επειδή ο όγκος ήταν αρκετά μειωμένος οι συγκολήσεις έπρεπε να επιτευχθούν με μεγάλη λεπτομέρεια και με τέτοιο τρόπο έτσι ώστε να είναι εύκολα προσβάσιμες σε περίπτωση βλάβης ή ακόμα και στην αλλαγή της μπαταρίας.Επίσης έπρεπε να είναι κατανοητές σε κάποιον άλλον τεχνικό όταν θα δει την πλακέτα. Μπορώ να πω ότι η εργασία αυτή με βοήθησε να καταννοήσω περισσότερο τον «κόσμο των μικροελεγκτών», των ηλεκτρονικών κυκλωμάτων να εμπλουτίνω τις γνώσεις μου και να είμαι σε θέση οποιαδήποτε στιγμή χρειαστεί να ξανά χειριστώ ένα τέτοιου είδους πείραμα.

Όσον αναφορά την επεκτασιμότητα της πτυχιακής. Καταρχήν η πτυχιακή είχε σαν στόχο την υλοποίηση με το Arduino uno και η σχεδίαση ενός ψηφιακού παλμογράφου.Η επέκταση που έκανα εγώ προσωπικά ήταν να το μικρύνω σε όγκο σε πρώτη φάση.Μετά πρόσθεσα να λειτουργεί και σαν γεννήτρια συχνοτήτων αλλά και να παράγει προιωνοτές, τριγωνικές και τετραγωνικές παραστάσεις.Επίσης θα μπορούσε και να συμπεριληφθεί πέρα από αυτά που αναφέρθηκαν και ένας ανιχνευτής 0-5V.Ακόμα θα μπορούσε να υλοποιηθεί και σαν πολύμετρο.Ανεξάρτητα από όργανα μέτρησης και παλμογράφο θα μπορούσε να κατασκευαστεί ένα παιχνίδι δύο διαστάσεων λόγω της οθόνης.

ΒΙΒΛΙΟΓΡΑΦΙΑ

βιβλία

[1] Παναγίωτης Παπάζογλου M.sc, Ph.D. , Σπύρος-Πολυχρόνης Λιωνής M.sc (2021) Ανάπτυξη εφαρμογών με το Arduino, εκδόσεις Τζιόλα.

[2] Δημήτρης Πογαρίδης (2015) , Ενσωματωμένα συστήματα, οι μικροελεγκτές AVR και ARDUINO , εκδόσεις Δίσιγμα.

[3] ΒΟΥΤΥΡΑΚΟΥ ΔΙΑΛΕΚΤΗ ΑΘΗΝΑ (2020), ΡΟΜΠΟΤΙΚΗ , εκδόσεις ΠΑΤΑΚΗΣ.

[4] Δημήτρης Καλοφωλιάς (2017), Ο προγραμματισμός του μικροελεγκτή AVR ATMega328 με τη χρήση της πλατφόρμας ARDUINO, εκδόσεις Τζιόλα.

[5] Γιάννης Θ. Κάππος, Αριστείδης Σ. Μπούρας (2021), Arduino. Αλγοριθμική, προγραμματισμός και εφαρμογές, εκδόσεις ΚΛΕΙΔΑΡΙΘΜΟΣ.

Data Sheet

[6] https://datasheet.octopart.com/A000066-Arduino-datasheet-38879526.pdf

Internet Site

[7] https://lastminuteengineers.com/nokia-5110-lcd-arduino-tutorial/

[8] https://lastminuteengineers.com/nokia-5110-lcd-arduino-tutorial/

[9] https://www.arduino.cc/en/guide/environment#toc1

[10] https://www.arduino.cc/en/guide/environment#toc1

[11] https://www.arduino.cc/en/guide/environment#toc1

[12] http://www.introtoarduino.com/downloads/IntroArduinoBook.pdf

[13] http://softwear.cc/book/files/Open_Softwear-beta090712.pdf

[14] https://www.allaboutcircuits.com/technical-articles/understanding-arduino-uno-hardware-design/

[15] https://riptutorial.com/ebook/arduino

[16] https://el.strephonsays.com/difference-between-risc-and-cisc

[17]https://cdn-learn.adafruit.com/downloads/pdf/adafruit-gfx-graphics-library.pdf

[18] https://acoptex.com/wp/master-arduino-with-free-ebook-arduino-projects-for-beginners/

[19] https://freepdf-books.com/beginning-nfc-introduction-to-arduino-and-nfc-pdf-free-download/

[20] https://www.dwrean.net/2014/09/16-serialprint-arduino-ide.html

[21] http://srukami.inf.ua/pultoscop_v25110.html

[22] https://www.instructables.com/Girino-Fast-Arduino-Oscilloscope/

[23] https://learn.adafruit.com/adafruit-arduino-lesson-5-the-serial-monitor/the-serial-monitor

[24] https://freepdf-books.com/arduino-wearables-pdf-free-download/

[25] https://freepdf-books.com/arduino-robotics-book-100-free-books-download-full-books-for-free/

[26] https://alexgyver.ru/

[27] https://en.wikipedia.org/wiki/Main_Page

[28] https://arduinohistory.github.io/

[29] https://www.ardumotive.com/arduino-ide-gr.html

[30] https://educ8s.tv/arduino-bitmap-graphics-tutorial/

[31] https://www.tutorialspoint.com/arduino/arduino_board_description.htm

[32] http://web.csulb.edu/~hill/ee346/Lectures/02%20Intro%20Microcontroller.pdf

[33] https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

[34] https://learn.adafruit.com/adafruit-arduino-lesson-5-the-serial-monitor/the-serial-monitor

[35]https://learnelectronics.gr/%CE%B3%CE%B5%CE%BD%CE%BD%CE%AE%CF%84%CF%81%CE%B9%CE%B1-%CF%83%CE%AE%CE%BC%CE%B1%CF%84%CE%BF%CF%82/