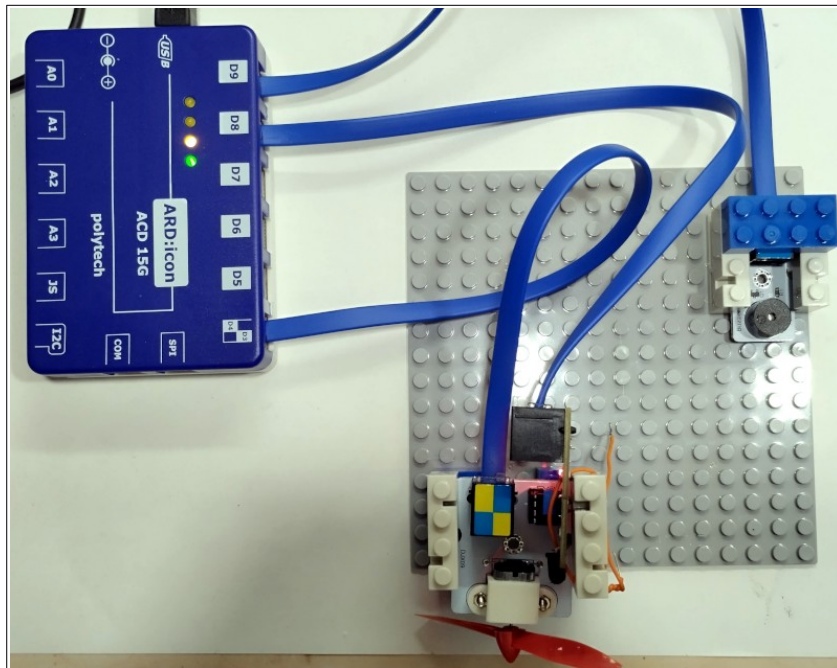


Άσκήσεις Ρομποτικής & Αυτοματισμού

στο κιτ S3T της Polytech



Έκδοση 1.0

Σταύρος Σ. Φώτογλου ΠΕ86
Ε.Κ. Πρέβεζας – 1ο ΕΠΑ.Λ. Πρέβεζας

Πρέβεζα 2025

Εισαγωγή

Το kit **S3T** της εταιρείας Polytech διαθέτει ένα μικροελεγκτή **ARD:icon** ο οποίος περιέχει μια πλακέτα **Arduino Uno**. Διαθέτει 30 αρθρώματα με αισθητήρες και ενεργοποιητές τα οποία συνδέονται εύκολα μέσω καλωδίων RJ12 όπως και στα kit της Lego. Επίσης διαθέτει βασικά ηλεκτρονικά εξαρτήματα και breadboard για την υλοποίηση απλών κυκλωμάτων.

Για λογισμικό προγραμματισμού μπορούμε να χρησιμοποιήσουμε τα εξής IDE:

1. Προγραμματισμός με blocks

- **PictoBlox**. Πολύ καλό λογισμικό το οποίο βασίζεται στο Scratch 3 με υποστήριξη πολλών εξαρτημάτων και εύκολη επικοινωνία με τον μικροελεγκτή. Λειτουργεί σε όλα τα λειτουργικά συστήματα και είναι μεταφρασμένο στα ελληνικά.
- **mBlock**. Εξίσου καλό βασισμένο στο Scratch 3 και μεγάλη συλλογή βιβλιοθηκών. Σε αντίθεση με το πρώτο επιτρέπει να φτιάξουμε blocks για νέα εξαρτήματα. Λογισμικό ανοιχτού κώδικα αλλά μέχρι στιγμής δεν έχει μεταφραστεί στα ελληνικά. Λειτουργεί σε Windows και Mac OS και υπάρχει έκδοση Web αν δεν θέλουμε να το εγκαταστήσουμε. Επικοινωνεί εύκολα με τον μικροελεγκτή.
- **Mind+**. Εξίσου καλό με το πρώτο φτιαγμένο από την DFRobot και βασισμένο στο Scratch 3. Σε αντίθεση με το πρώτο επιτρέπει να φτιάξουμε blocks για νέα εξαρτήματα. Λογισμικό ανοιχτού κώδικα με υποστήριξη ελληνικής γλώσσας. Λειτουργεί σε όλα τα λειτουργικά συστήματα. Επικοινωνεί εύκολα με τον μικροελεγκτή.
- **Blockly-at-rduino**. Βασισμένο στο Blockly. Λογισμικό ανοιχτού κώδικα το οποίο έχει σταματήσει να αναπτύσσεται. Λειτουργεί τοπικά ή από server μέσα από τον browser. Δεν επικοινωνεί με την πλακέτα και για να ανεβάσουμε τον κώδικα στον μικροελεγκτή πρέπει να αντιγράψουμε το πηγαίο πρόγραμμα C++ που παράγεται από τα blocks στο Arduino IDE να μεταγλωττιστεί και να μεταφορτωθεί στην πλακέτα.

2. Προγραμματισμός σε γλώσσα C++ (Wiring)

- **Arduino IDE**
- **VsCode - Platform IO**

Άσκηση 1 – Αυτοματισμός ανίχνευσης και σβήσίματος φλόγας

Να συναρμολογήσετε σύστημα αυτοματισμού το οποίο ανιχνεύει την ύπαρξη φλόγας σε ένα χώρο. Μόλις ανιχνεύσει την φλόγα χτυπάει ηχητικός συναγερμός και μετά από λίγο ξεκινάει τον ανεμιστήρα ώστε να σβήσει την φωτιά.

Τι θα χρειαστούμε


1. Μικροελεγκτή **ARD:icon**.
2. Άρθρωμα ανεμιστήρα **DJX09** με έλικα.
3. Άρθρωμα ενεργού βομβητή **AJX03**.
4. Άρθρωμα ανίχνευσης φλόγας **DJS13**.
5. Βάση για τουβλάκια.
6. Τουβλάκια στήριξης.
7. Τρία καλώδια RJ12.
8. Καλώδιο USB για προγραμματισμό και παροχή ρεύματος.

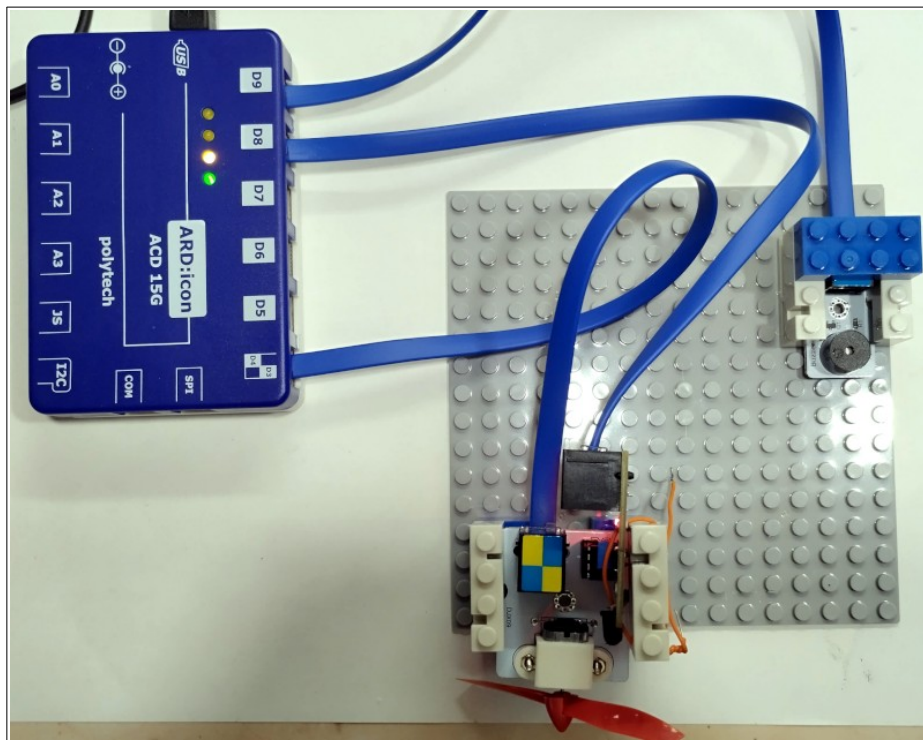
Τοποθετούμε τα εξαρτήματα και τα συνδέουμε ως εξής:

Τον ανεμιστήρα στην υποδοχή D3, D4 του ARD:icon.

Τον βομβητή στην υποδοχή D9.

Τον αισθητήρα ανίχνευσης φλόγας στην υποδοχή D8.

Από επιπλέον βιβλιοθήκες θα χρειαστούμε την επέκταση **Communication**. Πάμε κάτω αριστερά και πατάμε το  κουμπί προσθήκη επέκτασης, Hardware, Communication.



Στο PictoBlox φτιάχνουμε το ακόλουθο sketch.

The screenshot shows the PictoBlox software interface. On the left is a block-based code editor for an Arduino Uno. The code starts with 'when Arduino Uno starts up', followed by 'set serial 0 baud rate to 115200'. A 'για πάντα' (for always) loop contains: 'όρισε data σε read status of digital pin 8', 'write data on serial 0', an 'εάν data = 0 τότε' (if data = 0 then) block with 'Αν είναι 0 τότε υπάρχει φλόγα' (If it is 0 then there is a flame), 'set digital pin 9 output as HIGH' (with a 'Τόνος' tone block), 'περίμενε 2 δευτερόλεπτα' (wait 2 seconds), 'set digital pin 3 output as LOW', 'set digital pin 4 output as HIGH' (with a 'Δεξιόστροφη Περιστροφή' block), and 'αλλιώς' (else) block with 'set digital pin 3 output as LOW', 'set digital pin 4 output as LOW', and 'set digital pin 9 output as LOW' (with a 'Σταμάτησε το μοτέρ Σταμάτησε τον τόνο' block). The loop ends with 'περίμενε 1 δευτερόλεπτα'. On the right is a C++ code editor showing the generated code:

```

1 //This c++ code is generated by PictoBlox
2
3 //Global Variables are declared here
4 float data;
5
6 void setup() {
7   //put your setup code here, to run once:
8   Serial.begin(115200);
9   pinMode(8, INPUT);
10  pinMode(9, OUTPUT);
11  pinMode(3, OUTPUT);
12  pinMode(4, OUTPUT);
13 }
14
15 void loop() {
16   //put your main code here, to run repeatedly:
17
18   data = digitalRead(8);
19   Serial.println(data);
20   if((data == 0)) {
21     digitalWrite(9, true);
22     delay(2 * 1000);
23     digitalWrite(3, false);
24   }
25 }

```

This is a detailed view of the block-based code. It shows the 'when Arduino Uno starts up' block, the 'set serial 0 baud rate to 115200' block, and the 'για πάντα' loop. Inside the loop, it reads the status of digital pin 8 and writes it to serial 0. An 'εάν data = 0 τότε' block triggers a sequence: a yellow note 'Αν είναι 0 τότε υπάρχει φλόγα', setting digital pin 9 to HIGH (with a 'Τόνος' block), waiting 2 seconds, setting digital pin 3 to LOW and digital pin 4 to HIGH (with a 'Δεξιόστροφη Περιστροφή' block), and an 'αλλιώς' block where digital pin 3 is set to LOW, digital pin 4 is set to LOW, and digital pin 9 is set to LOW (with a 'Σταμάτησε το μοτέρ Σταμάτησε τον τόνο' block). The loop concludes with a 1-second wait.

Αυτό μεταφράζεται αυτόματα σε C++ δίπλα στην δεξιά στήλη.

1	//This c++ code is generated by PictoBlox
2	

```

3 //Global Variables are declared here
4 float data;
5
6 void setup() {
7     //put your setup code here, to run once:
8     Serial.begin(115200);
9     pinMode(8, INPUT);
10    pinMode(9, OUTPUT);
11    pinMode(3, OUTPUT);
12    pinMode(4, OUTPUT);
13
14 }
15
16
17 void loop() {
18     //put your main code here, to run repeatedly:
19
20
21     data = digitalRead(8);
22     Serial.println(data);
23     if ((data == 0)) {
24         digitalWrite(9, true);
25         delay(2 * 1000);
26         digitalWrite(3, false);
27         digitalWrite(4, true);
28     }
29     else {
30         digitalWrite(3, false);
31         digitalWrite(4, false);
32         digitalWrite(9, false);
33     }
34     delay(1 * 1000);
35 }

```

Ανάβουμε ένα κερί και το πλησιάζουμε μπροστά από τον ανεμιστήρα. Μόλις ο αισθητήρας φλόγας ανιχνεύσει την υπέρυθη ακτινοβολία θα ανάψει στην πλακέτα του αισθητήρα ένα κόκκινο Led. Τότε θα ηχήσει ο βομβητής και θα ξεκινήσει ο ανεμιστήρας. Όταν σβήσει η φλόγα του κεριού θα σταματήσει ο ανεμιστήρας και ο βομβητής. Στον αισθητήρα υπάρχει ένα τρίμερ το οποίο ρυθμίζει την ευαισθησία του αισθητήρα. Το ρυθμίζουμε ώστε να μην είναι πολύ ευαίσθητο και ανάβει με τον φωτισμό περιβάλλοντος αλλά να μπορεί να ανιχνεύσει την ύπαρξη φλόγας στα 5-10 εκατοστά. Υπάρχει video με επίδειξη της άσκησης στην διεύθυνση:

<https://youtu.be/8yI9SzL-hw>

Άσκηση 2 – Αυτοματισμός συναγερμού με διακοπή δέσμης LASER

Να συναρμολογήσετε σύστημα αυτοματισμού το οποίο ανιχνεύει την ύπαρξη εμποδίου σε έναν διάδρομο με μία δέσμη Laser. Όταν η δέσμη Laser διακοπεί από κάποιο εμπόδιο θα ηχήσει ο συναγερμός και θα σταματήσει μόνο όταν το εμπόδιο δεν διακόπτει την δέσμη.

Τι θα χρειαστούμε

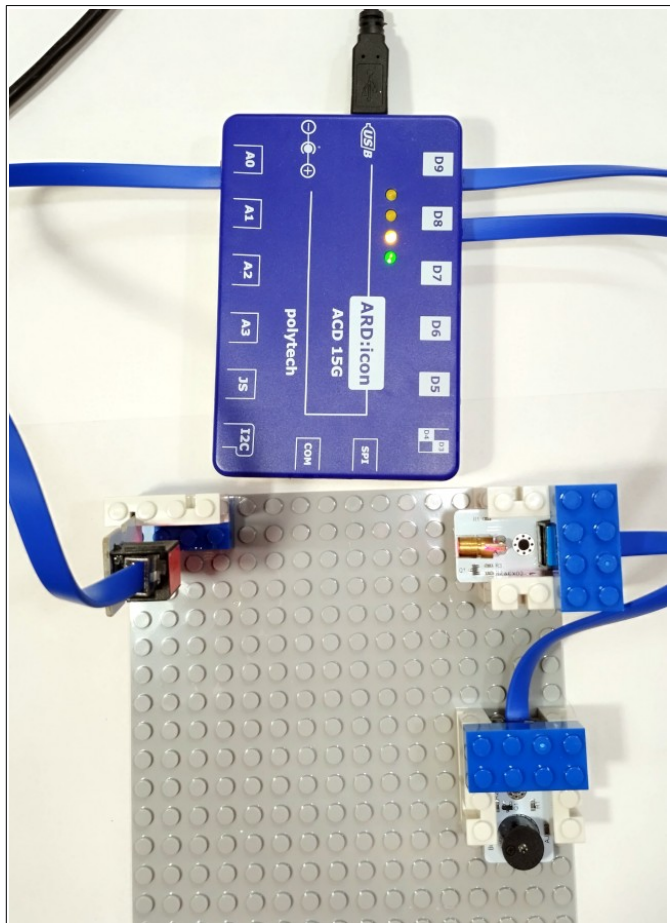
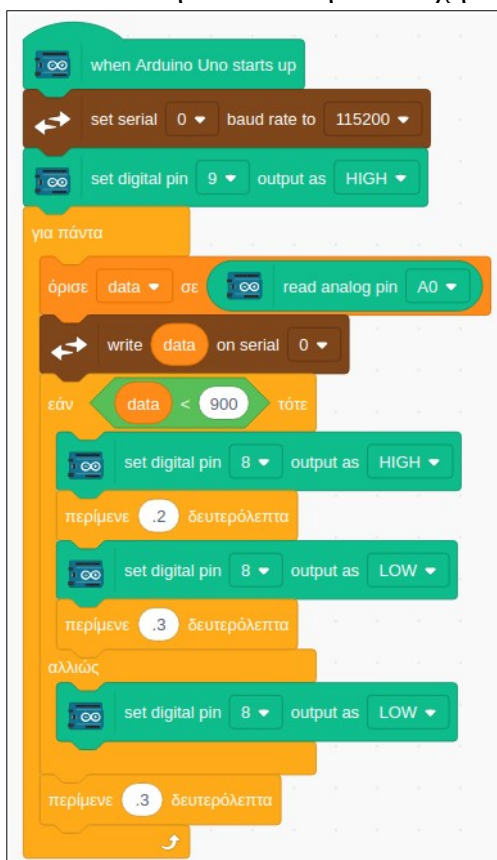
1. Μικροελεγκτή **ARD:icon**.
2. Άρθρωμα Laser **AFX02**.
3. Άρθρωμα ενεργού βομβητή **AJX03**.
4. Άρθρωμα φωτοαντίστασης L.D.R. **AJS03**.
5. Βάση για τουβλάκια.
6. Τουβλάκια στήριξης.
7. Τρία καλώδια σύνδεσης RJ12.
8. Καλώδιο USB για προγραμματισμό και παροχή ρεύματος.

Τοποθετούμε τα εξαρτήματα και τα συνδέουμε ως εξής:

Το Laser στην υποδοχή D9 του ARD:icon.

Τον βομβητή στην υποδοχή D8.

Την φωτοαντίσταση L.D.R. στην υποδοχή A0.



Υπάρχει video με επίδειξη της άσκησης στην διεύθυνση: https://youtu.be/LdOoYF4_Fbw

Άσκηση 3 – Μέτρηση απόστασης με υπερήχους

Να συναρμολογήσετε σύστημα μέτρησης το οποίο με την χρήση υπερήχων μετράει την απόσταση ενός αντικειμένου. Έχει ικανοποιητική ακρίβεια σε αποστάσεις έως 100cm. Βασίζεται στην λειτουργία του sonar, δηλαδή εκπέμπει μια ριπή υψηλής ακουστικής συχνότητας 40KHz που είναι έξω από το ακουστικό φάσμα του ανθρώπου. Συγκεκριμένα ο πομπός είναι το δεξί πιεζοηλεκτρικό. Όταν σταματήσει η εκπομπή ο δέκτης που είναι το αριστερό πιεζοηλεκτρικό λαμβάνει την ηχώ αν αυτή έχει αντανακλαστεί σε κάποιο εμπόδιο. Μετρώντας τον χρόνο υπολογίζουμε την απόσταση εφόσον γνωρίζουμε την ταχύτητα του ήχου στον αέρα.



Τι θα χρειαστούμε

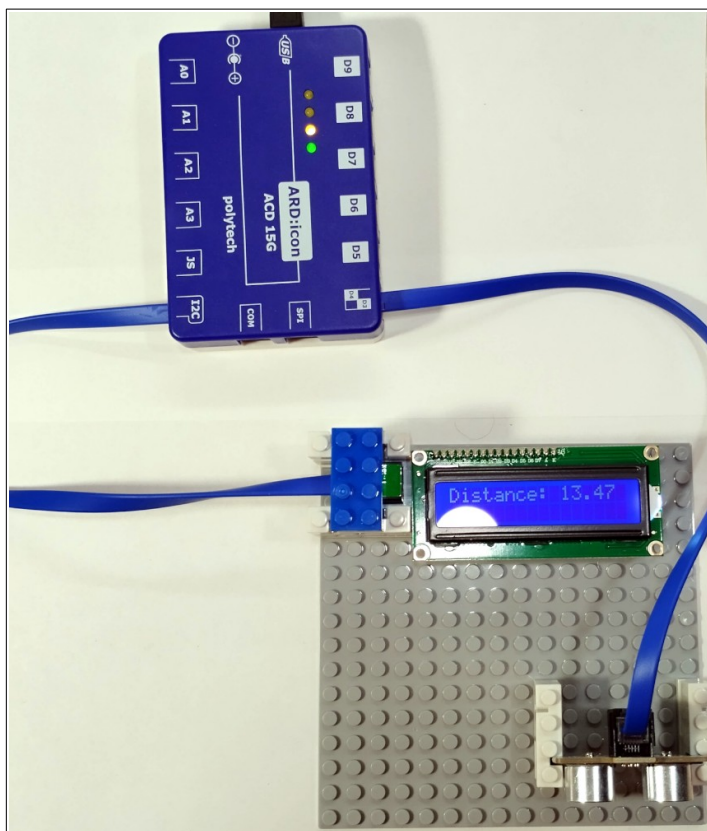
1. Μικροελεγκτή **ARD:icon**.
2. Άρθρωμα αισθητήρα υπερήχων **DJS22**.
3. Άρθρωμα οθόνης χαρακτήρων 16x2 **AJX04**.
4. Βάση για τουβλάκια.
5. Τουβλάκια στήριξης.
6. Δύο καλώδια σύνδεσης RJ12.
7. Καλώδιο USB για προγραμματισμό και παροχή ρεύματος.

Τοποθετούμε τα εξαρτήματα και τα συνδέουμε ως εξής:

Τον αισθητήρα υπερήχων στην υποδοχή D3, D4 του ARD:icon.

Την οθόνη χαρακτήρων στην υποδοχή I2C.

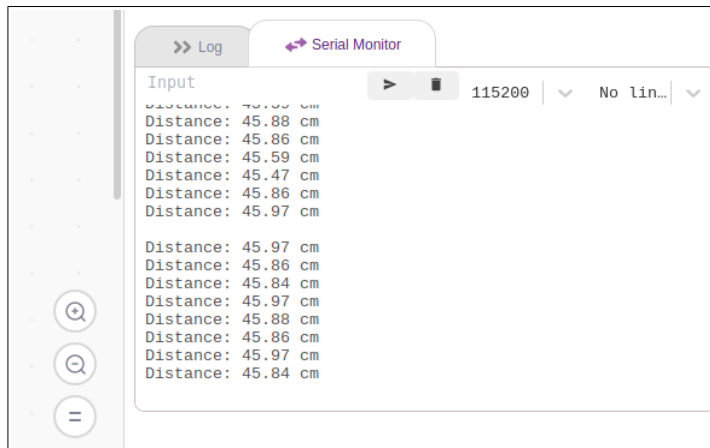
Από επιπλέον βιβλιοθήκες θα χρειαστούμε τις επεκτάσεις **Communication** και **Display Modules**. Πάμε κάτω αριστερά και πατάμε το κουμπί  προσθήκη επέκτασης, μετά κουμπί Hardware, επιλέγουμε το πλαίσιο Communication και  μετά το πλαίσιο Display Modules.



Δημιουργήστε το ακόλουθο σενάριο στο PictoBlox.



Στο κάτω δεξί τμήμα της οθόνης πατάμε την καρτέλα Serial Monitor ώστε να βλέπουμε τις μετρήσεις και μέσα από το PictoBlox.



Χρησιμοποιήστε ένα φύλλο A4 ως αναφορά το οποίο έχει διαστάσεις 29,7x21cm. Αρχικά τοποθετήστε οριζόντια το φύλλο ώστε η μια μεγάλη πλευρά να εφάπτεται στην αρχή του αισθητήρα και κάθετα στην άλλη πλευρά βάλτε ένα εμπόδιο και επαληθεύστε αν η απόσταση είναι... Επαναλάβετε το ίδιο τοποθετώντας το φύλλο κάθετα.

Υπάρχει video με επίδειξη της άσκησης στην διεύθυνση: <https://youtu.be/Q0dufEJnMPw> .

Άσκηση 4 – Μέτρηση θερμοκρασίας και σχετικής υγρασίας

Να συναρμολογήσετε σύστημα μέτρησης το οποίο με την χρήση του αισθητήρα DHT11 μετράει την θερμοκρασία και την σχετική υγρασία του χώρου. Τα αποτελέσματα θα εμφανίζονται στο σειριακό τερματικό του Pictoblox και στην οθόνη χαρακτήρων LCD 16x2.

Τι θα χρειαστούμε



1. Μικροελεγκτή **ARD:icon**.
2. Άρθρωμα αισθητήρα DHT11 **MJS22**.
3. Άρθρωμα οθόνης χαρακτήρων LCD 16x2 **AJX04**.
4. Βάση για τουβλάκια.
5. Τουβλάκια στήριξης.
6. Δύο καλώδια σύνδεσης RJ12.
7. Καλώδιο USB για προγραμματισμό και παροχή ρεύματος.

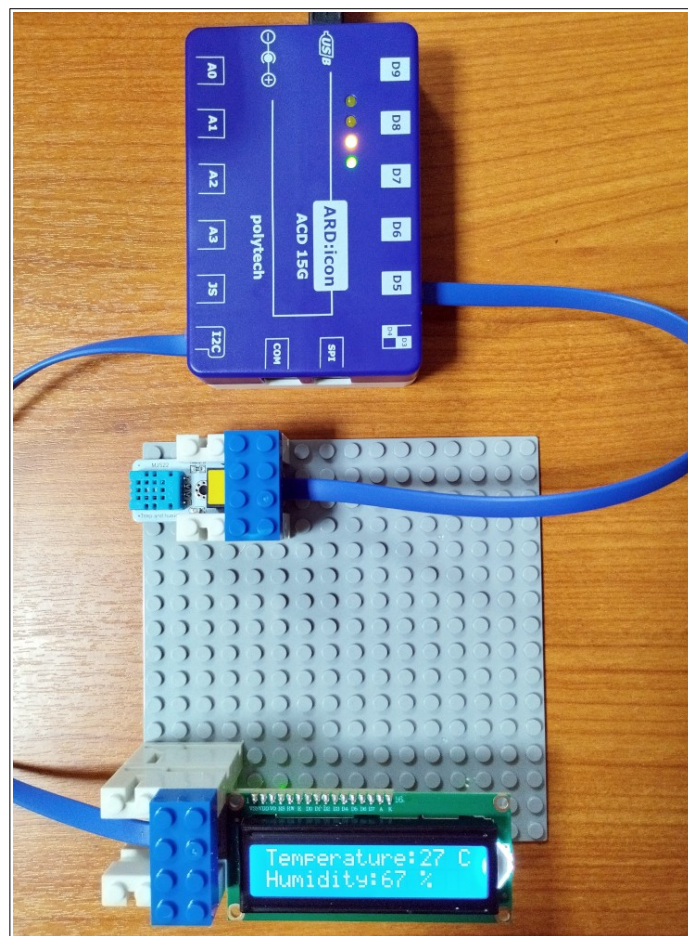
Τοποθετούμε τα εξαρτήματα και τα συνδέουμε ως εξής:

Τον αισθητήρα μέτρησης θερμοκρασίας και υγρασίας στην υποδοχή D5 του ARD:icon.

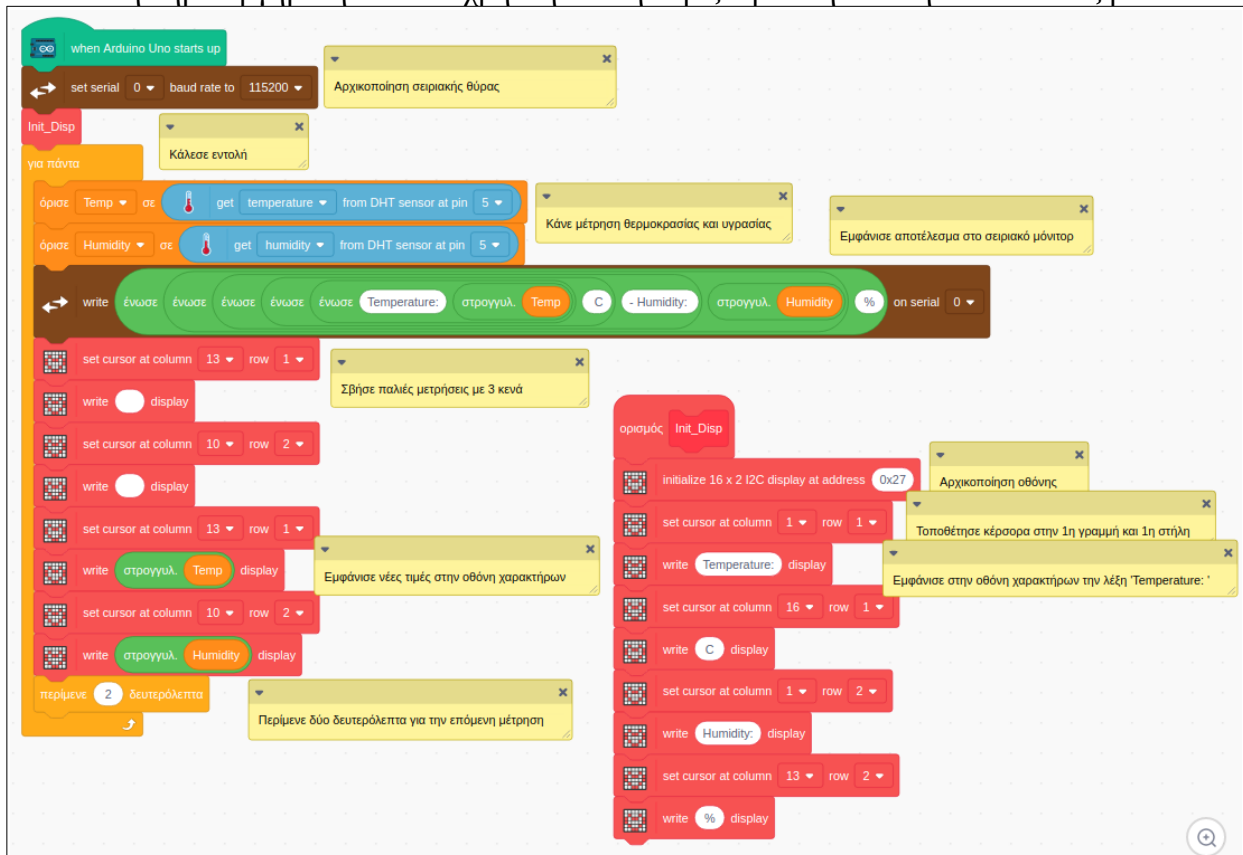
Την οθόνη χαρακτήρων στην υποδοχή I2C.

Από επιπλέον βιβλιοθήκες θα χρειαστούμε τις επεκτάσεις **Communication** και **Display Modules**.

Πάμε κάτω αριστερά και πατάμε το κουμπί  προσθήκη επέκτασης, μετά κουμπί Hardware, επιλέγουμε το πλαίσιο Communication και  μετά το πλαίσιο Display Modules.



Δημιουργήστε το σενάριο στο PictoBlox και μεταφορτώστε το στο ARD:icon. Η εντολή Init_Dispatch είναι εντολή δημιουργημένη από τον χρήστη και την ορίζουμε στην ενότητα οι εντολές μου.



Άσκηση 5 – Αναγνώριση χρωμάτων

Πολλές φορές στα συστήματα αυτοματισμού είναι απαραίτητη η αναγνώριση χρώματος ενός αντικείμενου π.χ. για αυτόματη διαλογή σε ιμάντα γραμμής παραγωγής. Αν και υπάρχουν έτοιμα αρθρώματα γι' αυτόν τον σκοπό όπως το **TCS3200** με πολύ καλά αποτελέσματα, εμείς θα χρησιμοποιήσουμε μόνο τα αρθρώματα του kit S3T. Φυσικά δεν θα έχουμε τα αποτελέσματα του TCS3200 ιδίως σε έντονο φως περιβάλλοντος. Θα χρησιμοποιήσουμε τρία χρωματιστά LED και την φωταντίσταση LDR. Κανονικά θα έπρεπε να χρησιμοποιήσουμε τα εξής LED: κόκκινο, πράσινο και μπλε. Επειδή το πράσινο του kit δεν έχει αρκετή ένταση φωτός εμείς θα χρησιμοποιήσουμε το κίτρινο Led.

Αρχή λειτουργίας: Κάθε αντικείμενο απορροφάει κάποια χρώματα και αντανακλά κάποια άλλα. Για παράδειγμα το μαύρο απορροφάει όλα τα χρώματα και δεν αντανακλά κανένα χρώμα. Το άσπρο αντανακλά όλα τα χρώματα. Το κόκκινο αντανακλά μόνο το κόκκινο χρώμα, το μπλε μόνο το μπλε κ.ο.κ. Εμείς ανάβουμε ένα – ένα τα Led και διαβάζουμε την ένταση του φωτός που αντανακλάται από τα αντικείμενα. Αν για παράδειγμα το κόκκινο έχει αρκετά μεγαλύτερα ένταση από τα άλλα δύο τότε το αντικείμενο είναι κόκκινο.

Να συναρμολογήσετε σύστημα ανίχνευσης τεσσάρων χρωμάτων (κόκκινο, πράσινο, κίτρινο, μπλε) από χαρτονένιες κάρτες. Τα αποτελέσματα θα εμφανίζονται στο σειριακό τερματικό του Pictoblox και στην οθόνη χαρακτήρων LCD 16x2.

Τι θα χρειαστούμε

1. Μικροελεγκτή **ARD:icon**.
2. Άρθρωμα χρωματιστών LED **DJX06, DJX08, DJX12**.
3. Άρθρωμα φωτοαντιστάτη LDR **AJS03**.
4. Άρθρωμα οθόνης χαρακτήρων LCD 16x2 **AJX04**.
5. Βάση για τουβλάκια.
6. Τουβλάκια στήριξης.
7. Πέντε καλώδια σύνδεσης RJ12.
8. Καλώδιο USB για προγραμματισμό και παροχή ρεύματος.
9. Τέσσερις χαρτονένιες χρωματιστές κάρτες που κόβουμε από χαρτόνι κανσόν.

Τοποθετούμε τα εξαρτήματα και τα συνδέουμε ως εξής:

Το κόκκινο LED στην υποδοχή D5, ο μπλε στην D6 και το κίτρινο στην υποδοχή D7 του ελεγκτή ARD:icon.

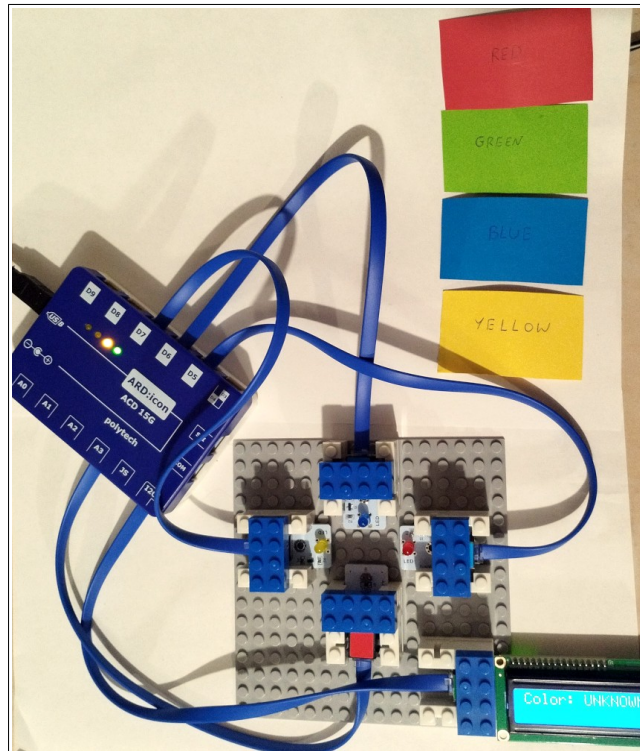
Τον φωταντιστάτη LDR στην αναλογική υποδοχή A3.

Την οθόνη χαρακτήρων στην υποδοχή I2C.

Από επιπλέον βιβλιοθήκες θα χρειαστούμε τις επεκτάσεις **Communication** και **Display Modules**.

Πάμε κάτω αριστερά και πατάμε το κουμπί  προσθήκη επέκτασης, μετά κουμπί Hardware, επιλέγουμε το πλαίσιο Communication και  μετά το πλαίσιο Display Modules.

Συναρμολογούμε το υλικό:



Δημιουργούμε το σενάριο στο PictoBlox. Για ευκολία το σπάμε σε τρία μέρη. Δύο υποπρογράμματα και το κυρίως πρόγραμμα.

ορισμός LedColor text

Υποπρόγραμμα το οποίο ανάγει ένα από τα 3 LEDs ή τα σβήνει όλα

εάν text = RED τότε

- set digital pin 5 output as HIGH
- set digital pin 6 output as LOW
- set digital pin 7 output as LOW

αλλιώς

εάν text = BLUE τότε

- set digital pin 5 output as LOW
- set digital pin 6 output as HIGH
- set digital pin 7 output as LOW

αλλιώς

εάν text = YELLOW τότε

- set digital pin 5 output as LOW
- set digital pin 6 output as LOW
- set digital pin 7 output as HIGH

αλλιώς

- set digital pin 5 output as LOW
- set digital pin 6 output as LOW
- set digital pin 7 output as LOW

ορισμός Scan

Υποπρόγραμμα το οποίο ανάγει ένα-ένα τα 3 LEDs καλώντας την LedColor(String) και διαβάζει την ένταση ανακλώμενου φωτός από την LDR. Επίσης υπολογίζει την μέση τιμή των τριών μεγεθών. Για βαθμονόμηση εμφανίζει τα αποτελέσματα στο σειριακό τερματικό

περίμενε .1 δευτερόλεπτα

όρισε Rmag σε read analog pin A3

LedColor RED

περίμενε .1 δευτερόλεπτα

όρισε Bmag σε read analog pin A3

LedColor BLUE

περίμενε .1 δευτερόλεπτα

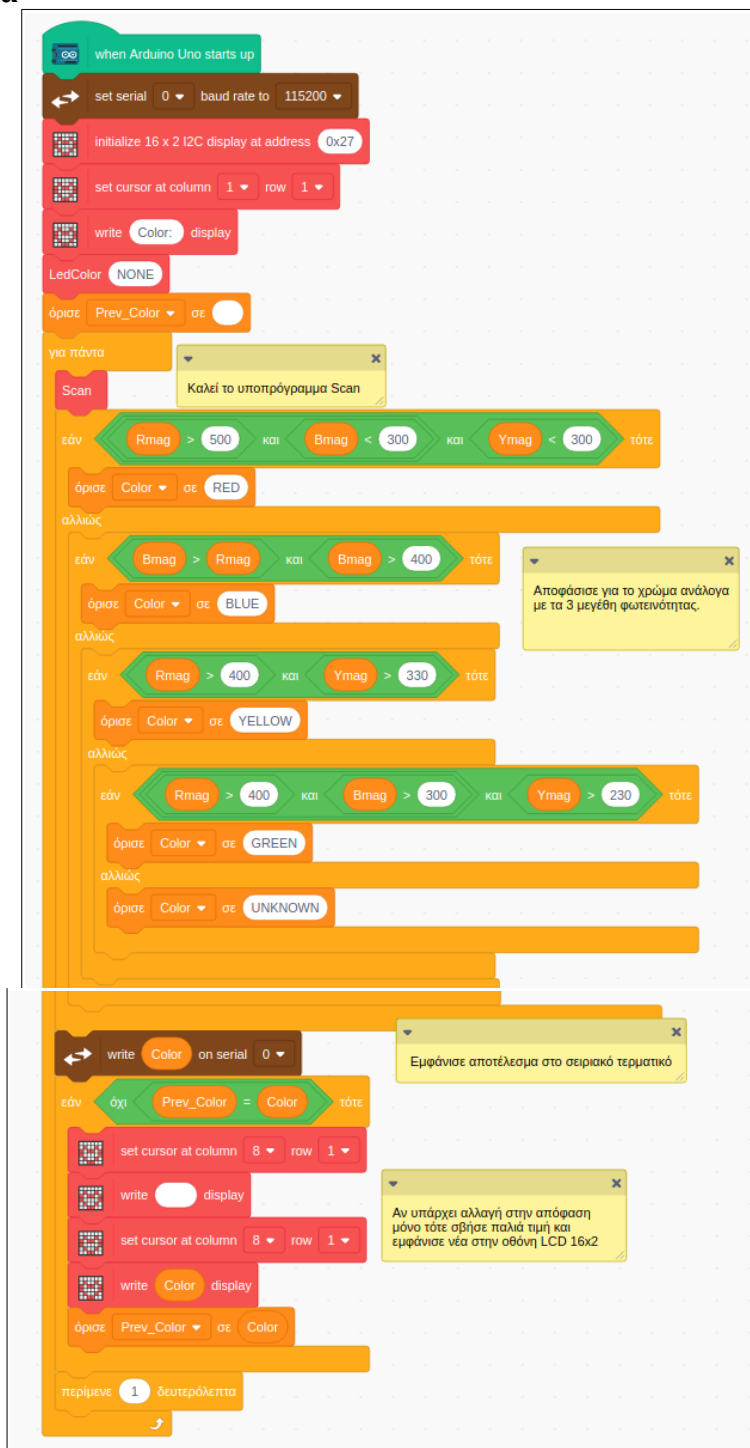
όρισε Ymag σε read analog pin A3

όρισε Int_avg σε $Rmag + Bmag + Ymag / 3$

LedColor NONE

write ένωση R= ένωση Rmag ένωση B= ένωση Bmag ένωση Y= ένωση Ymag ένωση -->AVG= Int_avg on serial 0

Κυρίως πρόγραμμα



Βαθμονόμηση

Με σχετικά χαμηλό φωτισμό περιβάλλοντος τοποθετούμε τις κάρτες μία – μία και διαβάζουμε τις τιμές R, B και Y στο σειριακό τερματικό. Διορθώνουμε τις τιμές των συγκρίσεων στην δομή πολλαπλής επιλογής του κυρίου προγράμματος.

Υπάρχει video με επίδειξη της άσκησης στην διεύθυνση: <https://youtu.be/IUTzhHtUHrk> .

Άσκηση 6 – Απαρίθμηση στο δυαδικό σύστημα με 4 δυαδικά ψηφία

Μέσω του STEM οι μαθητές μπορούν να πειραματιστούν και να κατανοήσουν έννοιες άλλων μαθημάτων όπως μαθηματικά, φυσική κλπ. Με αυτή την δραστηριότητα οι αρχάριοι μαθητές της πληροφορικής θα κατανοήσουν την αναπαράσταση αριθμών στο δυαδικό σύστημα.

Να συναρμολογήσετε σύστημα απαρίθμησης στο δυαδικό σύστημα με 4 bits δηλαδή από το 0 – 15. Το αποτέλεσμα θα εμφανίζεται σε 4 χρωματιστά LEDs και θα μετράει από το 0 έως το 15 κάθε φορά που θα πιέζεται το button. Παράλληλα στο σειριακό τερματικό θα εμφανίζεται ο αριθμός στο δεκαδικό και στο δεκαεξαδικό σύστημα.


Τι θα χρειαστούμε

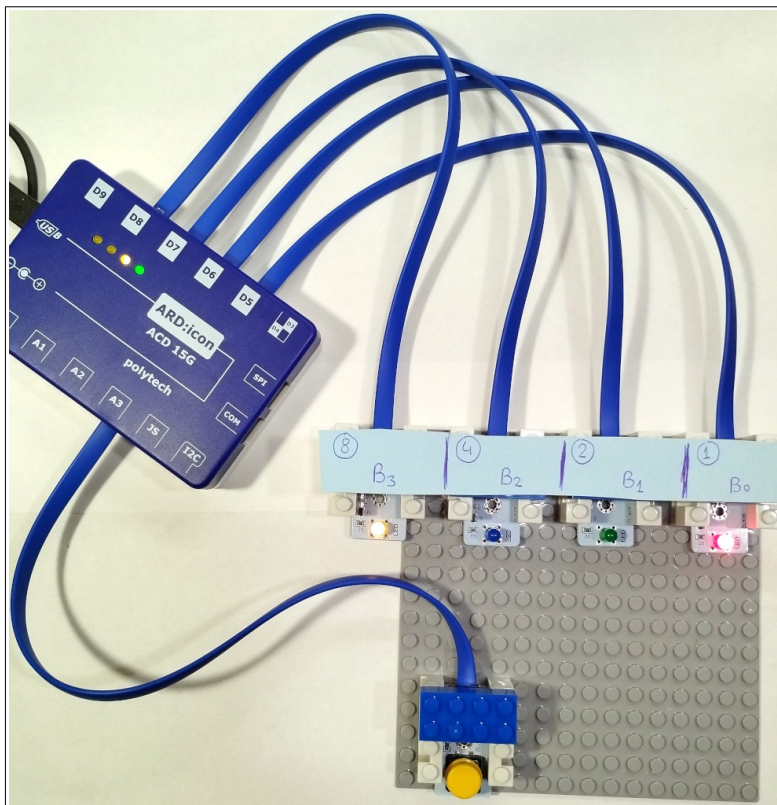
1. Μικροελεγκτή **ARD:icon**.
2. Άρθρωτα χρωματιστών LED **DJX06, DJX07, DJX08, DJX12**.
3. Άρθρωμα πιεστικού διακόπτη Push Button **DJS09**.
4. Βάση για τουβλάκια.
5. Τουβλάκια στήριξης.
6. Πέντε καλώδια σύνδεσης RJ12.
7. Καλώδιο USB για προγραμματισμό και παροχή ρεύματος.

Τοποθετούμε τα εξαρτήματα και τα συνδέουμε ως εξής:

Το κόκκινο LED στην υποδοχή D5, το πράσινο στην D6, το μπλε στην υποδοχή D7 και το κίτρινο στην υποδοχή D8 του ελεγκτή ARD:icon.

Το push button στην αναλογική υποδοχή A3.

Από επιπλέον βιβλιοθήκες θα χρειαστούμε την επέκταση **Communication**. Πάμε κάτω αριστερά και πατάμε το κουμπί προσθήκη επέκτασης,  μετά κουμπί Hardware, επιλέγουμε το πλαίσιο Communication.



Δημιουργούμε το σενάριο στο Pictoblox. Το χωρίσαμε σε τρία μέρη, δύο υποπρογράμματα και το κυρίως πρόγραμμα. Το υποπρόγραμμα **Num2Led** δέχεται έναν αριθμό και τον εμφανίζει στα 4 Leds σε δυαδική μορφή. Το υποπρόγραμμα **hex** δέχεται έναν αριθμό 4bit στο δεκαδικό και τον μετατρέπει στο δεκαεξαδικό. Τέλος το κυρίως πρόγραμμα ελέγχει το push button και σε κάθε πάτημα αυξάνει τον μετρητή κατά ένα, εμφανίζει το αποτέλεσμα στα Leds καλώντας το Num2Led και εμφανίζει τον αριθμό του μετρητή στο σειριακό τερματικό, σε δεκαδική και δεκαεξαδική μορφή. Αν ο μετρητής υπερχειλίσει (πάνω από 15) γυρίζει πάλι στο μηδέν.

The image shows three Pictoblox sub-programs:

- Num2Led**: Takes a number 'num' and outputs its 4-bit binary representation to pins 8, 7, 6, and 5. It uses a loop to check each bit and set the pin accordingly.
- hex**: Takes a decimal value 'dec' (10-15) and outputs its hexadecimal equivalent (A-F) to a variable 'hexnum'.
- Main Program**: Starts when the Arduino Uno starts up. It initializes serial communication at 115200 baud. It reads a button state (pin A3) and increments a counter 'cnt'. It then calls the 'Num2Led' and 'hex' sub-programs to display the counter's value. It also writes the counter value to the serial monitor.

Δίπλα βλέπουμε τους αριθμούς όπως εμφανίζονται στο σειριακό τερματικό.

```

>> Log
Serial Monitor
Input
115200 No lin...
1 - 1
2 - 2
3 - 3
4 - 4
5 - 5
6 - 6
7 - 7
8 - 8
9 - 9
10 - A
11 - B
12 - C
13 - D
14 - E
15 - F
    
```

Δημιουργία κώδικα σε C++


Στη συνέχεια θα φτιάξουμε το ίδιο sketch αλλά σε γλώσσα C++. Αν συγκρίνουμε το κώδικα που παράγεται από το Pictoblox και αυτόν που ακολουθεί θα παρατηρήσουμε ότι ο πρώτος είναι πολύ μεγαλύτερος. Για να γράψουμε τον κώδικα μπορούμε να χρησιμοποιήσουμε το Pictoblox (Δεν αποθηκεύει τις αλλαγές στο project), το **Mind+** ή το **Arduino IDE**.

```

1 #define B0 5
2 #define B1 6
3 #define B2 7
4 #define B3 8
5 #define BUTTON 17 //A3
6
7 byte cnt;
8 bool Flag;
9 byte LEDS[] = {B0, B1, B2, B3};
10
11 void Num2Led(byte num) {
12     byte i;
13     num &= 15;
14     for (i = 0; i <= 3; i++)
15     {
16         if (num & 1)
17             digitalWrite(LEDS[i], HIGH);
18         else
19             digitalWrite(LEDS[i], LOW);
20         num >>= 1;
21     }
22 }
23
24 void setup() {
25     Serial.begin(115200);
26     pinMode(BUTTON, INPUT);
27     cnt = 0;
28     Flag = 0;
29 }
30
31 void loop() {
32     char tmp[30];
33     if (!digitalRead(BUTTON))
34     {
35         if (Flag == false)
36         {
37             Flag = true;
38             if (cnt < 15)
39                 cnt += 1;
40             else
41                 cnt = 0;
42             Num2Led(cnt);
43             sprintf(tmp, "Dec: %d - Hex: %X - Oct: %o", cnt, cnt, cnt);
44             Serial.println(tmp);
45         }
46     }
47     else
48         Flag = false;
49     delay(50);
51 }

```

Στο στιγμιότυπο που ακολουθεί χρησιμοποιήσαμε το **Mind+** για την μεταγλώττιση και το ανέβασμα του κώδικα στον ελεγκτή Arduino UNO.

Για να βλέπουμε τα αποτελέσματα στο σειριακό τερματικό πρέπει να θέσουμε την ταχύτητα επικοινωνίας στα 115200bps από το εικονίδιο κάτω  δεξιά.

Εδώ εμφανίζεται ο αριθμός στο σειριακό τερματικό και στο οκταδικό σύστημα αρίθμησης.

Υπάρχει video με επίδειξη της άσκησης στην διεύθυνση: <https://youtu.be/yogxfKp91OU> .

The screenshot displays the Mind+ V1.7.3 RC3.0 IDE interface. The top bar shows the application name and version, along with tabs for 'Online', 'Offline', and 'Python'. The left sidebar contains a 'Blocks' panel with various programming blocks categorized under 'Ελεγχος' (Control), 'Ελεγχος' (Control), 'Τελεστές' (Operators), 'Μεταβλητές' (Variables), 'Οι Εντολές μου' (My Commands), and 'Arduino'. The central workspace shows a block diagram with a 'Uno starts' block and a 'για πάντα' (Forever) loop block. The right panel features a 'Manual Editing' tab with a code editor showing C++ code for controlling LEDs and a button. The code includes definitions for pins, a loop to write to LEDs, and a setup function. Below the code editor is a serial monitor displaying the output of the program, showing decimal, hexadecimal, and octal values for the LED states.

```

1 #define B0 5
2 #define B1 6
3 #define B2 7
4 #define B3 8
5 #define BUTTON 17 //A3
6
7 byte cnt;
8 bool Flag;
9 byte LEDS[] = {B0, B1, B2, B3};
10
11 void Num2Led(byte num) {
12   byte i;
13   num &= 15;
14   for (i = 0; i <= 3; i++)
15   {
16     if (num & 1)
17       digitalWrite(LEDS[i], HIGH);
18     else
19       digitalWrite(LEDS[i], LOW);
20     num >>= 1;
21   }
22 }
23
24 void setup() {
25   Serial.begin(115200);
26   pinMode(BUTTON, INPUT);
27   cnt = 0;
28   Flag = 0;
29 }
30
31 void loop() {
32   char tmp[30];
33   if (digitalRead(BUTTON))
34   {
35     if (Flag == false)
36   
```

upload success
Dec: 1 - Hex: 1 - Oct: 1
Dec: 2 - Hex: 2 - Oct: 2
Dec: 3 - Hex: 3 - Oct: 3
Dec: 4 - Hex: 4 - Oct: 4
Dec: 5 - Hex: 5 - Oct: 5
Dec: 6 - Hex: 6 - Oct: 6
Dec: 7 - Hex: 7 - Oct: 7
Dec: 8 - Hex: 8 - Oct: 10
Dec: 9 - Hex: 9 - Oct: 11
Dec: 10 - Hex: A - Oct: 12
Dec: 11 - Hex: B - Oct: 13
Dec: 12 - Hex: C - Oct: 14
Dec: 13 - Hex: D - Oct: 15
Dec: 14 - Hex: E - Oct: 16
Dec: 15 - Hex: F - Oct: 17
Dec: 0 - Hex: 0 - Oct: 0