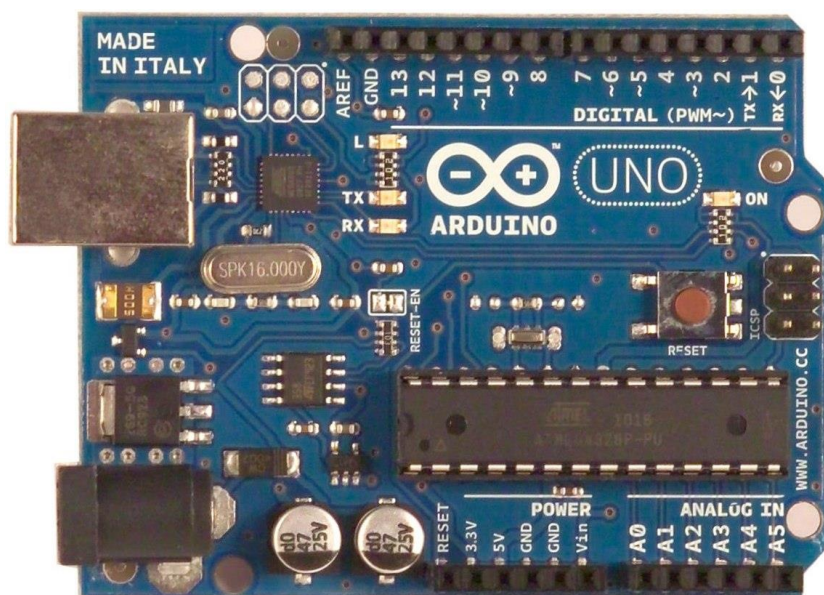


מכללת אורט גבעת רם

מגמת הרובוטיקה

ארדואינו - Arduino



מהדורה ניסויית תשע"ה

כתב: מוטי מאיר

תוכן

פעילות 1 – הכרת סביבת העבודה של הארדואינו.	3
פעילות 2 – הדלקת לדים, בר לדים, 7 Segment בחיבור מקבילי	8
פעילות 3 – הוספת לחצן אל הארדואינו ושימוש במוניטור	18
פעילות 4 – יצירת תדר וחיבור רמקול	24
פעילות 5 – המרה מאנלוגי לדיגיטלי ADC - Analog to Digital	32
פעילות 6 – חיבור LDR וחיישן טמפרטורה למבואות אנאלוגיים	36
פעילות 7 – אפנון רוחב דופק PWM - Pulse With Modulation	39
פעילות 8 – בקרה על מנוע DC שמחובר לדוחף L293D	47
פעילות 9 – חיישן (Infra Red) IR	54
פעילות 10 – מנוע סרוו Servo motor	56
פעילות 11 – חיישן מרחק אולטרא סוני Ultra-Sonic SRF05	63
פעילות 12 – פסיקות – interrupt (חיצוניות)	66
פעילות 13 – חיישן מרחק GP2Y0A21YK	72
פעילות 14 – BlueTooth ותקשורת טורית (ה UART)	77

אמרו חכמים: **"כשם שאי אפשר לבר בלי תבן כך אי אפשר לספר בלא שגיאות"**. אודה ואבקש ממי שמוצא טעות (או כל הערה אחרת) שיכתוב לדוא"ל momeir@gmail.com ויבוא על הברכה.

פעילות 1 – הכרת סביבת העבודה של הארדואינו.

ארדואינו (arduino) הוא:

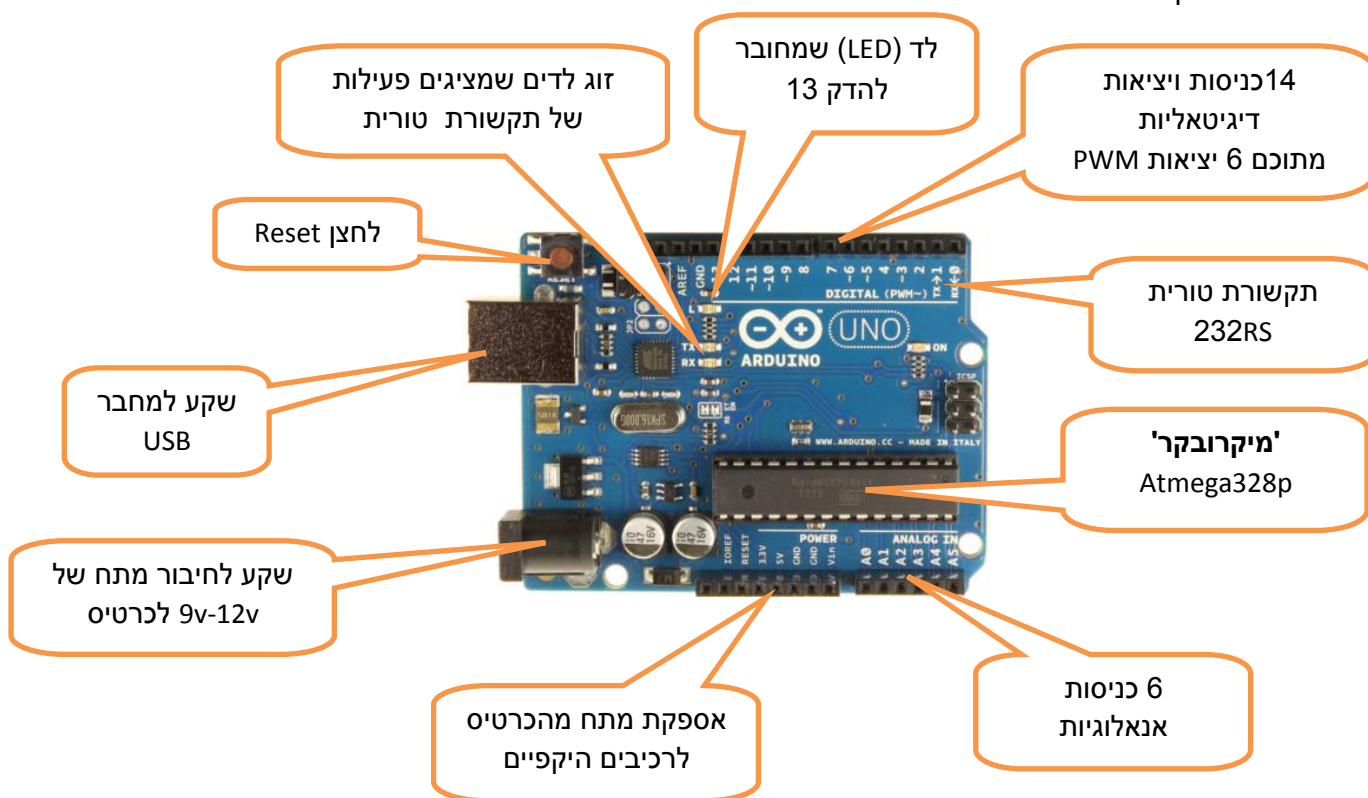
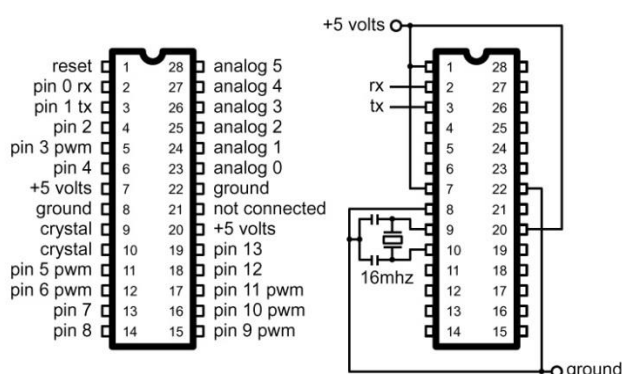
1. **כרטיס** - המכיל 'מיקרובקר' מסדרת AVR של חברת ATMAIL (למעלה מ-20 כרטיסים שונים).
2. **סביבת פיתוח משולבת (IDE)** - שמטרתה ליצור סביבה **נוחה ופשוטה** לפיתוח פרויקטים המשלבים תוכנה עם חומרה (אלקטרוניקה).
3. **קהילת משתמשים גדולה** - חובבים ומקצוענים, פורומים ומדריכים מקוונים,

את סביבת הפיתוח של הארדואינו ניתן להוריד מהאתר הרשמי ARDUINO.CC. באמצעות האתר תוכל גם להכיר את מגוון כרטיסי ארדואינו, את סביבת הפיתוח וספריות שונות להפעלת רכיבי חומרה

כרטיס הארדואינו שנעבוד איתו הוא Arduino uno המתואר באיור 1. הונו Arduino uno מבוסס על המיקרובקר Atmega328p

לידע כללי: המאפיינים העיקריים של הרכיב Atmega328p הם:

- תדר שעון 16MHz.
- מתח עבודה 5v (אספקת מתח לכרטיס 7v-12v).
- זרם בהדקי I/O עד 40mA.
- זיכרון תוכנית (flash) בגודל 32k.
- זיכרון נתונים (ram) בגודל 2k.
- 14 כניסות ויציאות דיגיטליות.
- 6 יציאות PWM.
- 6 כניסות אנאלוגיות ברזולוציה של 10 סיביות.
- תקשורת טורית (rs232, i2c, spi).
- 2 פסיקות חיצוניות.



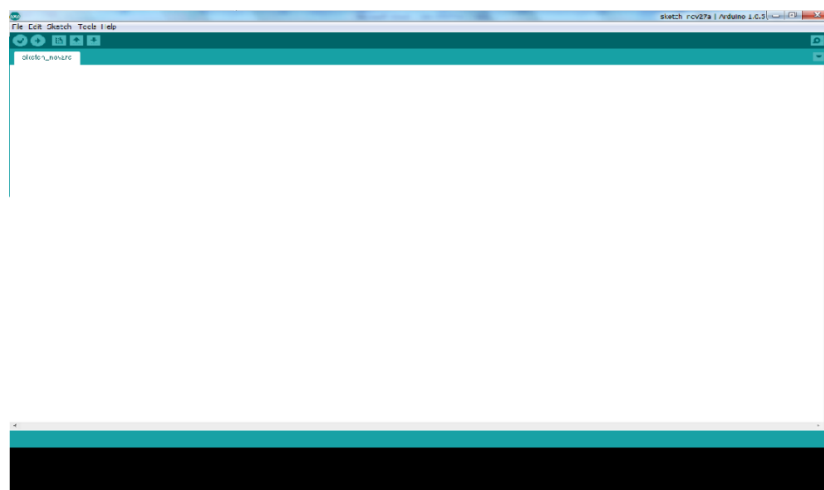
איור 1

את התוכנית אנו נכתוב בעזרת 'עורך תוכניות'. שפת התכנות היא שפת C++ .

בצע את השלבים הבאים:



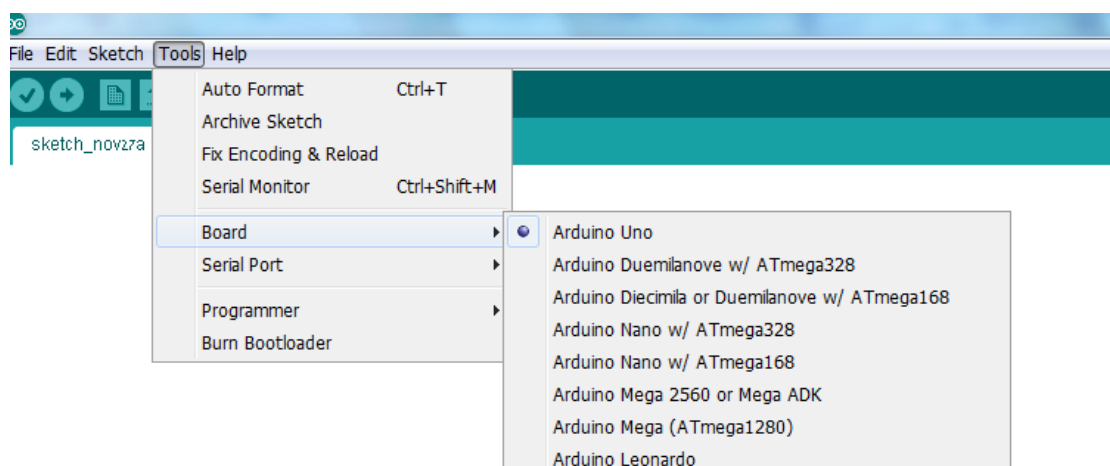
1. חבר את כרטיס הארדואינו אל המחשב, באמצעות כבל ה-usb שברשותך.
2. להפעלת 'סביבת הפיתוח' של הארדואינו, לחץ 'לחיצה כפולה' על הצלמית:
3. בעקבות ה'לחיצה כפולה' ייפתח החלון של עורך התוכניות (כמראה בתמונה)



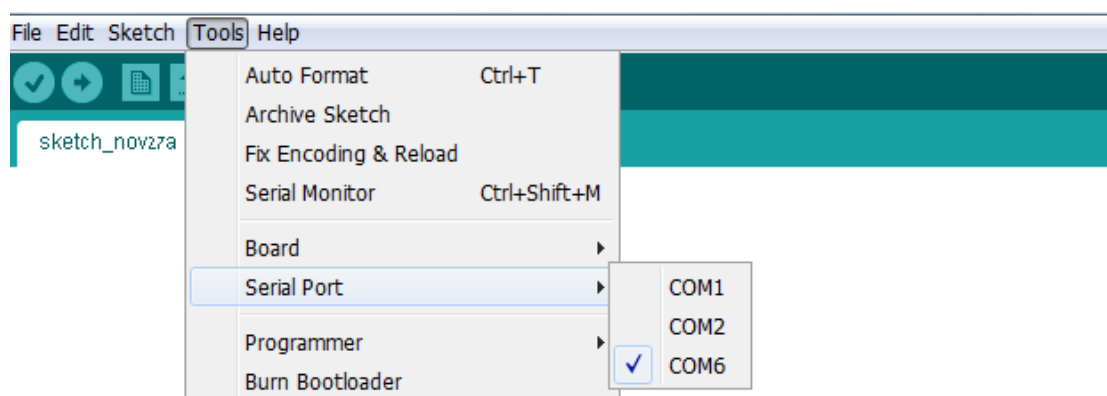
4. תפקיד הכפתורים שבעורך התוכניות



5. בחר Tools->Board , ובחר Arduino Uno (כמראה בתמונה)



6. בחר COM -> Serial Port -> Tools. (אבל לא com1) כמראה בתמונה (האחרון ברשימה).



7. התוכנית שלהלן גורמת ללד שמחובר להדק 13 להבהב. הלד יידלק לשניה אחת ויכבה לשניה אחת. כתוב את התוכנית בעורך התוכניות.

```

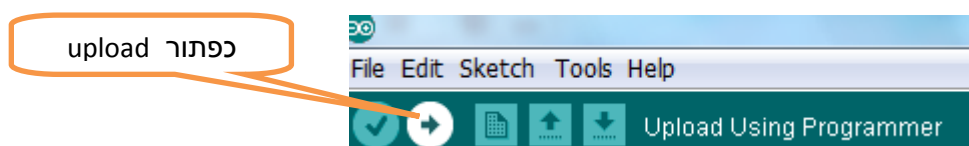
File Edit Sketch Tools Help
[Icons]
sketch_nov27b $

void setup() {
  pinMode(13, OUTPUT);
}

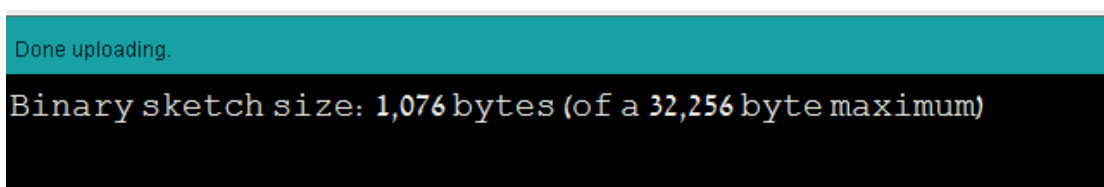
void loop() {
  digitalWrite(13, HIGH);
  delay(1000);
  digitalWrite(13, LOW);
  delay(1000);
}

```

8. להעלאת (לצריבת) התוכנית אל ה'מיקרובקר', יש לחצות על כפתור upload



9. בסיום התהליך, תופיע בתחתית המסך הודעה:

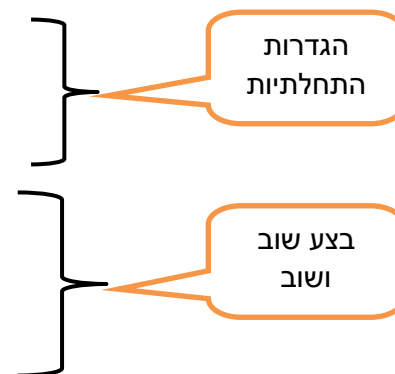


הסבר התוכנית:

```

void setup() {
  pinMode(13, OUTPUT); // קבע את הדק 13 כ 'מוצא'.
}
void loop() {
  digitalWrite(13, HIGH); // הדלק את הלד
  delay(1000);           // המתן שניה אחת
  digitalWrite(13, LOW); // כבה את הלד
  delay(1000);           // המתן שניה אחת.
}

```



נסביר קצת יותר.....

כל תוכניות בארדואינו מכילות לפחות 2 פונקציות.

1. פונקציית setup() – שבה כותבים הגדרות ו/או הוראות שהמיקרובקר יבצע באופן חד פעמי.
 2. פונקציית loop() - שבה כותבים הוראות שהמיקרובקר יבצע שוב ושוב.
- לפני שמשתמשים בהדק דיגיטאלי צריך להגדיר האם ההדק הוא 'מבוא' (INPUT) או 'מוצא' (OUTPUT).
 ההוראה: pinMode(13, OUTPUT); פירושה: "קבע את הדק 13 כמוצא".
- ההוראה: digitalWrite(13, HIGH); פירושה: "כתוב להדק 13 ערך של '1' ". במילים אחרות "קבע בהדק 13 מצב של 5v". הוראה זו גורמת ללד להידלק.
- ההוראה: delay(1000); יוצרת 'השהיה'. זמן ההשהיה נכתב בתוך הסוגרים ביחידות של אלפיות השנייה.
10. שנה את ההוראות delay(1000) ל delay(100). לאחר מכן בצע העלאה (upload) של התוכנית אל המיקרובקר באמצעות לחיצה על  (התוצאה: הלד יהבהב בתדר של 5HZ).
- 11 שנה את התוכנית כך שהלד יהבהב באופן הבא: יידלק למשך שניה ויהיה כבוי למשך חצי שניה.
12. התוכנית שלהלן גורמת ללד להבהב שש פעמים בקצב של 5HZ ושלוש פעמים בקצב של 1HZ.
- כתוב את התוכנית, והעלה למיקרובקר.

```

void setup() {
  pinMode(13, OUTPUT); // קבע את הדק 13 כ 'מוצא'
}
void loop() {
  for (int i=0; i<6; i++) { // בצע 6 פעמים
    digitalWrite(13, HIGH); // הדלק את הלד
    delay(100);           // המתן מאית השניה
    digitalWrite(13, LOW); // כבה את הלד
    delay(100);           // המתן מאית השניה.
  }
}

```

```

    }
    for (int i=0; i<3; i++) {      //   בצע 3 פעמים
        digitalWrite(13, HIGH); // הדלק את הLED
        delay(500);              // המתן מחצית השנייה
        digitalWrite(13, LOW);  // כבה את הLED
        delay(500);             // המתן מחצית השנייה.
    }
}

```

13. התוכנית שלהלן גורמת לLED להבהב בקצב שהולך וגדל. זמן ההשהיה תלוי בערך של i

```

void setup() {
    pinMode(13, OUTPUT);
}

void loop() {
    for (int i=68; i>8; i=i-4) { //   קצב הדלקה הולך ויורד...
        digitalWrite(13, HIGH);
        delay(5*i);             //   המתן זמן שתלוי בערכו של i
        digitalWrite(13, LOW);
        delay(5*i);             //   המתן זמן שתלוי בערכו של i
    }

    for (int i=8; i<68; i=i+4) { //   קצב הדלקה הולך וגדל.....
        digitalWrite(13, HIGH);
        delay(5*i);
        digitalWrite(13, LOW);
        delay(5*i);
    }
}

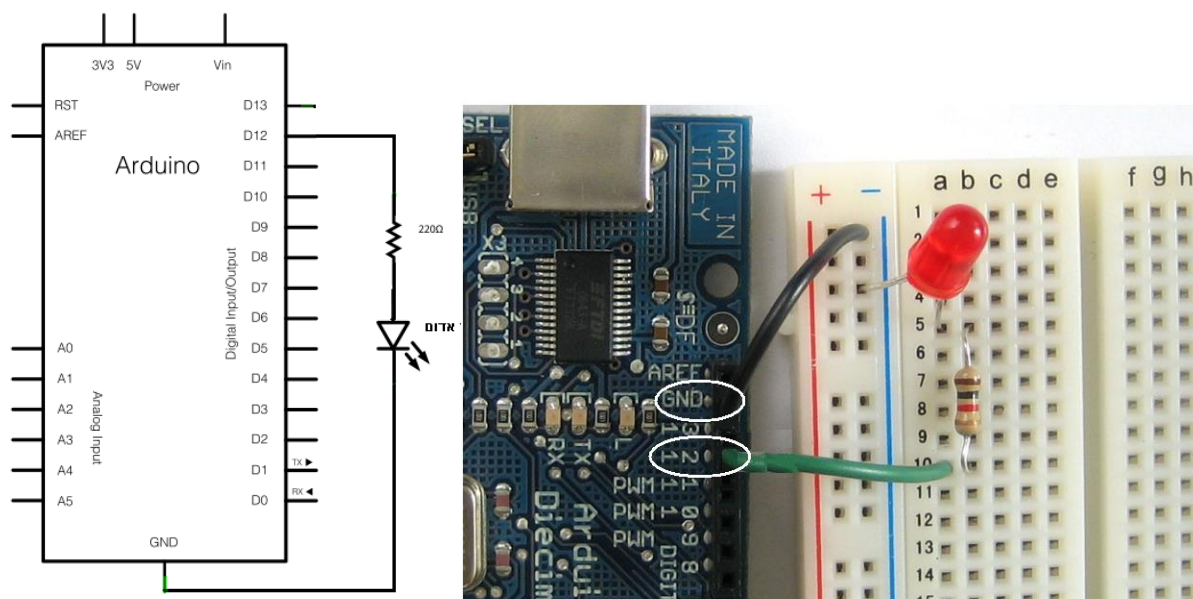
```

הערה:

במקום לרשום HIGH אפשר לרשום 1 כלומר, אפשר לכתוב גם digitalWrite(13,1);

פעילות 2 – הדלקת לדים, בר לדים , 7 Segment בחיבור טור

1. ברשותך 3 לדים בצבעים אדום, כתום וירוק. חבר את הלד האדום להדק 12 כמתואר באיור 1



איור 1

2. התוכנית שלהלן גורמת ללד להבהב בקצב של 1HZ (חצי שניה הלד דולק וחצי שניה הלד מכובה). כתוב את התוכנית ובדוק

```
Int redLed=12;
void setup() {
  pinMode(redLed, OUTPUT); // הגדר את הדק 12 כ'מוצא'
}
void loop() {
  digitalWrite(redLed, HIGH); // קבע את מצב ההדק ל '1' (5 וולט)
  delay(500); // המתן חצי שניה (500 אלפיות השניה)
  digitalWrite(redLed, LOW); // קבע את מצב ההדק ל '0'
  delay(500); // המתן חצי שניה (500 אלפיות השניה)
}
```

3. שנה את התוכנית כך שהלד יהבהב בקצב של 2 הבהובים לשניה (2HZ).

4. שנה את התוכנית כך שהלד יהבהב בקצב של 4HZ. (בכל הבהוב יש הדלקה וכיבוי)

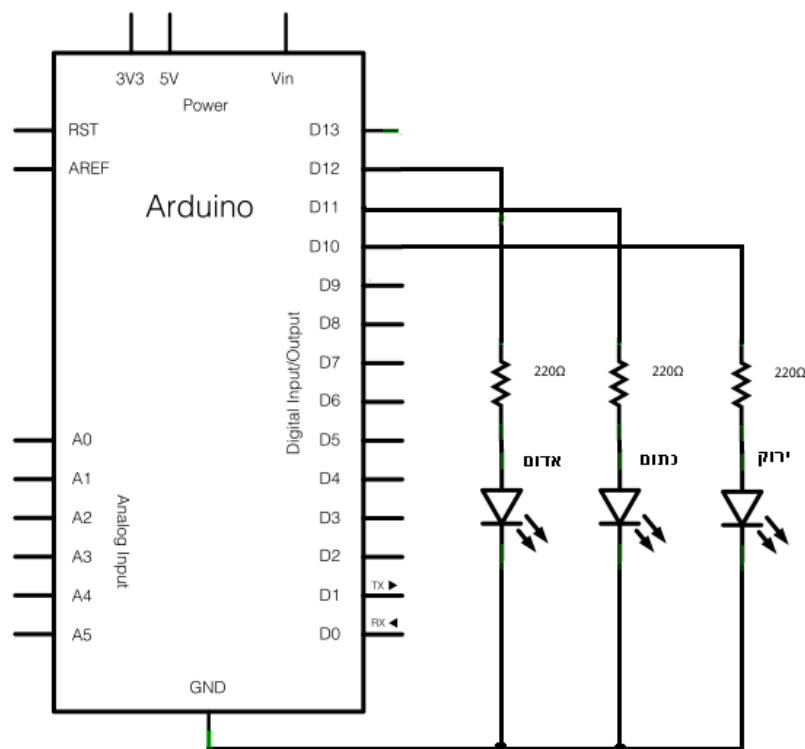
5. שנה את התוכנית כך שהלד יהבהב בקצב של 1HZ, אבל במשך 0.8 שניה הלד יידלק ובמשך 0.2 שניה הלד יכבה.

6. שנה את התוכנית כך שהלד יהבהב 7 פעמים בקצב של 2HZ ולאחר מכן 10 פעמים בקצב

של 4HZ. (פקודת for או while)

7. שנה את התוכנית כך שהלד יבהב 10 פעמים. ההבהוב הראשון יהיה בקצב של 1Hz, ההבהוב השני בקצב של 2Hz והאחרון של 10Hz (רמז: לולאה של 10. בכל איטרציה משנים את זמן ההשהיה delay)

8. הוסף את הלדים כתום וירוק אל כרטיס הארדואינו כמתואר באיור 2



9. כתוב תוכנית שגורמת לכל שלושת הלדים להבהב בקצב של 10Hz

10. כתוב תוכנית שמבצעת:

- הלד האדום נדלק למשך 0.6 שניה.
- גם הלד הכתום נדלק למשך 0.6 שניה.
- גם הלד הירוק נדלק למשך 0.6 שניה.
- כל הלדים נכבים.

11. כתוב תוכנית שתבצע:

- הלד האדום יידלק למשך 0.2 שניה.
- הלד האדום יכבה והלד הכתום יידלק למשך 0.4 שנייה.
- הלד הכתום יכבה והלד הירוק יידלק למשך 0.8 שנייה.

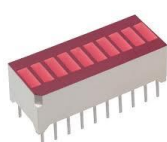
12. כתוב תוכנית שמחקה את פעולה 'רמזור'.

- א. רק הLED האדום נדלק למשך 2 שניות.
- ב. גם הLED הכתום נדלק למשך חצי שניה.
- ג. הLED האדום והכתום שניהם נכבים והLED הירוק נדלק למשך 3 שניות.
- ד. הLED הירוק מהבהב 6 פעמים בקצב של 4Hz.
- ה. הLED הירוק נכבה והLED הכתום נדלק למשך חצי שניה.
- ו. חזרה לסעיף א' <= הLED הכתום נכבה והLED האדום נדלק למשך 2 שניות.

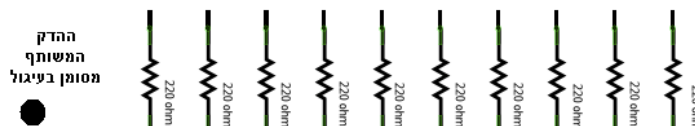
13. חיבור 'בר' של 10 לדים לכרטיס הארדואינו

ברשותך בר של 10 לדים באריזת DIP וכן 9 נגדים של 510 אום, באריזה כמתואר באיור 3.

בנה את המעגל כמתואר באיור 4



בר לדים



נגדים

איור 3

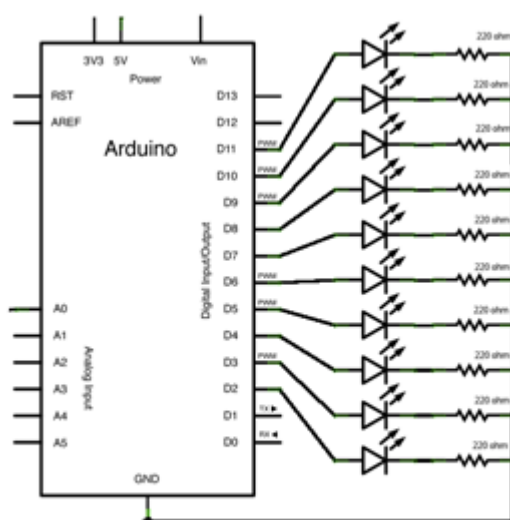
בנה את המעגל כמשורטט באיור 4 .

הערה:

באריזת הלדים יש 10 לדים.

באריזת הנגדים שלנו יש רק 9 נגדים.

לכן יש להוסיף עוד נגד אחד בודד.



איור 4

14. התוכנית שלהלן גורמת לכל הלידים להבהב בתדר של 1hz. כתוב ובדוק.

```
int i; // הגדר 'משתנה' בשם i שיכיל מספרים שלמים

void setup(){
    for ( i=2 ; i<12 ; i++) // כל עוד i קטן מ 12, בצע את ההוראה (שבשורה הבאה)
        (לאחר ביצוע ההוראה, יתווסף למשתנה i אחד.)
        pinMode(i,OUTPUT); // קבע את הדק i למצב 'מוצא'. בפעם הראשונה i=2 לאחר מכן 3 ..
}

void loop() {
    for ( i=2 ; i<12 ; i++) // כל עוד i קטן מ 12, בצע את ההוראה שבשורה הבאה
        digitalWrite(i, HIGH); // קבע את הדק i למצב 'מוצא'. בפעם הראשונה i=2 לאחר מכן 3 ..
        delay(500); // בצע 'השהייה' של 0.5 שניה
    for ( i=2 ; i<12 ; i++) // כל עוד i קטן מ 12, בצע את ההוראה שבשורה הבאה
        digitalWrite(i, LOW); // קבע את הדק i למצב 'מוצא'. בפעם הראשונה i=2 לאחר מכן 3 ..
        delay(500); // בצע 'השהייה' של 0.5 שניה
}
```

15. התוכנית הבאה "מדליקה נרות חנוכה". כלומר, בהתחלה מדליקים רק נר אחד, לאחר מכן שתי נרות.... שלוש נרות.... עד שכל הנרות (הלדים) דולקים.

```
int i;

void setup(){
  for ( i=2 ; i<12 ; i++) // קבע את ההדקים 2 עד 11 להדקי 'מוצא'
    pinMode(i,OUTPUT);
}

void loop() {
  for ( i=2 ; i<12 ; i++) // קבע מצב '0' בהדקים 2 עד 11. כלומר, כבה את כל הלדים.
    digitalWrite(i, LOW);

  for ( i=2 ; i<13 ; i++) { // הדלק לד לפי הערך של i
    digitalWrite(i , HIGH);
  }
}
```

16. התוכנית שלהלן יוצרת מצב של לד 'רץ' שמאלה כמוראה בתמונה שבאיור 5. כתוב ובדוק.



איור 5

```
int i;           // הגדר 'משתנה' בשם i שיכיל מספרים שלמים
```

```

void setup(){
for ( i=2 ; i<12 ; i++)    // קבע את ההדקים 2 עד 11 להדקי 'מוצא'
pinMode(i,OUTPUT);

    for ( i=2 ; i<12 ; i++) // קבע מצב '0' בהדקים 2 עד 11. כלומר, כבה את כל הלדים.
digitalWrite(i, LOW);
}

void loop() {
    for ( i=2 ; i<13 ; i++) { // כל עוד i קטן מ 12, בצע את ההוראות שבתוך 'הסוגריים המסולסלות'
digitalWrite(i , HIGH); // הדלק את הלד בהדק שערכו שווה ל i
digitalWrite(i-1, LOW); // כבה את הלד בהדק שערכו שווה ל i-1
delay(500);          // בצע 'השהייה' של 0.5 שניה
    }
}

```

17. שנה את התוכנית כך שנקבל לד 'רץ' ימינה.

18. התוכנית שלהלן שגורמת ללדים להידלק בסגנון 'נרות חנוכה' אבל 'קצב ההדלקה' לא קבוע ומשתנה. בתוכנית, בפקודת **delay(x)**, זמן ההשהייה לא קבוע כי הזמן יהיה תלוי בערכו של x כתוב את התוכנית ובדוק את האפקט המתקבל.....

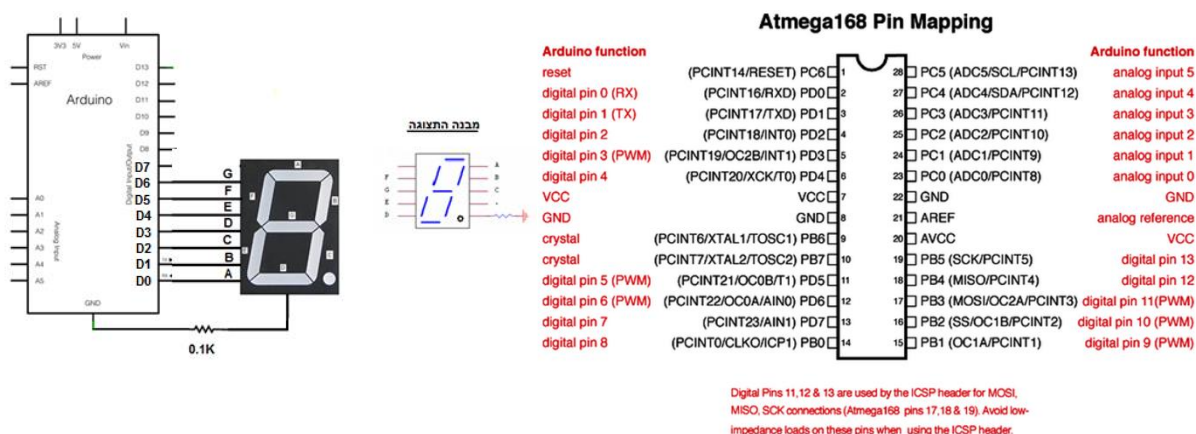
```

int i,x;    // הגדר שני משתנים. האחד בשם i והשני בשם x
void setup(){
    for ( i=2 ; i<12 ; i++)    // קבע את ההדקים כ 'מוצא'
        pinMode(i,OUTPUT);
}

void loop() {
    for (x=500; x>0; x=x-50) { // לולאת x תתבצע 10 פעמים. בכל פעם, הערך של x יקטן ב 50
        for ( i=2 ; i<12 ; i++) // כבה את כל הלדים
            digitalWrite(i, LOW);
        for ( i=2 ; i<10 ; i++) { // הדלק 8 נוריות לד.
            digitalWrite(i , HIGH);
            delay(x); // בצע השהייה לפי הערך שיש במשתנה x
        } // בהתחלה זמן ההשהייה 0.5 שניה...לאחר מכן 0.45 שניה...ובסוף רק 0.05 שניה
    }
}

```

19. בתרגילים הקודמים הקלט/פלט בארדואינו היה מול סיביות בודדות. לעיתים נרצה לעבוד מול 'מפתח' (port). למעבד 3 מפתחים B, C, D. כמוראה בשרטוט. ניתן לדוגמא לחבר למפתח D את תצוגת 7 המקטעים כמשורטט. כאשר סגמנט A מחובר להדק 0 (LSB), סגמנט B מחובר להדק 1 וכך הלאה.



20. התוכנית שלהלן מדליקה את הספרה 0. כתוב ובדוק.

שים לב: לפני ה Upload, יש לנתק את החוטים של סגמנט A וסגמנט B. זאת כדי שהם לא יפריעו לתקשורת הטורית בין המחשב לבקר. לאחר העלאת התוכנית לבקר אפשר לחבר שוב את החוטים.

```
void setup() {
    DDRD = B11111111; // הגדר את כל הסיביות כמוצא 'מוצא' ('1' = מוצא. '0' = מבוא)
    PORTD = B00111111; // הדלק את הספרה 0 (PORTD=0x3f);
}

void loop() {
}
```

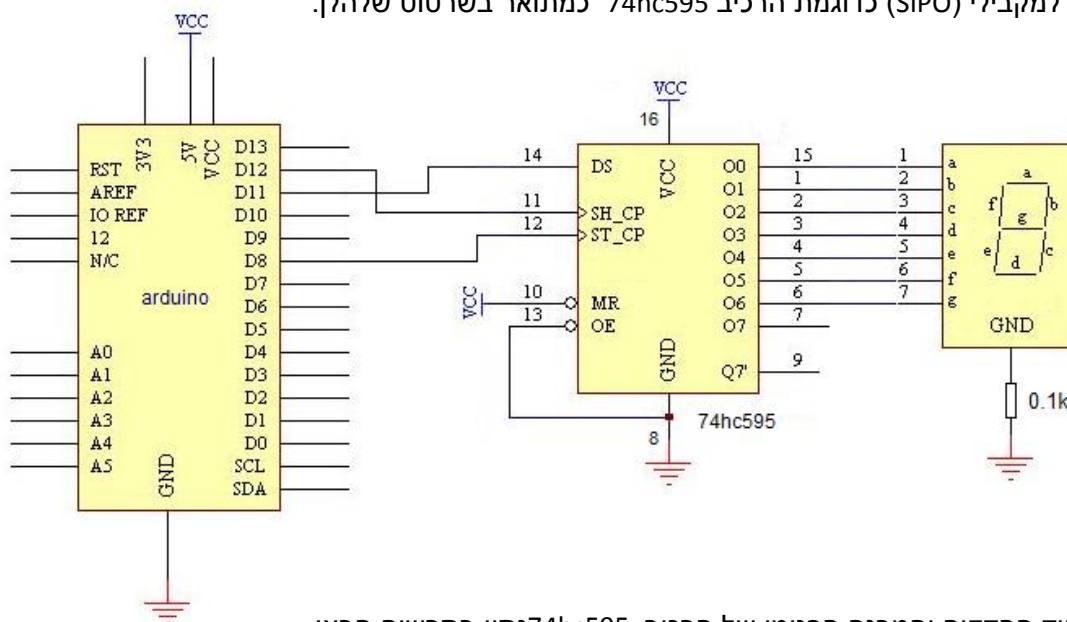
21. התוכנית שלהלן מדליקה את הספרות מ 0 עד 9. זמן ההשהיה בין ספרה לספרה הוא 0.5 שניה.

```
void setup() {
    DDRD = B11111111; // הגדר את כל הסיביות כמוצא
}

char disp[]={0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x6c,0x07,0x7f,0x67};

void loop() {
    for (int i=0; i<10; i++) {
        PORTD = disp[i];
        delay(1000);
    }
}
```

כדי לא 'לבזבז' הדקים של המיקרו, נכון ורצוי לחבר את תצוגות 7 המקטעים באמצעות חיבור אוגר טורי למקבילי (SIPO) כדוגמת הרכיב 74hc595. כמתואר בשרטוט שלהלן:



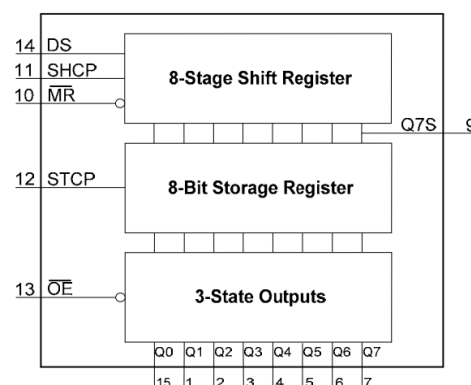
תפקיד ההדקים והמבנה הפנימי של הרכיב 74hc595 נתון בתרשים הבא:

74HC595

Pin Descriptions

Pin Number	Pin Name	Function
1	Q1	Parallel Data Output 1
2	Q2	Parallel Data Output 2
3	Q3	Parallel Data Output 3
4	Q4	Parallel Data Output 4
5	Q5	Parallel Data Output 5
6	Q6	Parallel Data Output 6
7	Q7	Parallel Data Output 7
8	GND	Ground
9	Q7S	Serial Data Output
10	MR	Master Reset Input
11	SHCP	Shift Register Clock Input
12	STCP	Storage Register Clock Input
13	OE	Output Enable Input
14	DS	Serial Data Input
15	Q0	Parallel Data Output 0
16	Vcc	Supply Voltage

Functional Diagram



22. התוכנית שלהלן מציגה את הספרות העשרוניות מ 0 עד 9 ע"ג התצוגה:

```

int latchPin = 8; // הדק הארדואינו לנעילת המידע הטורי באוגר המקבילי
int clockPin = 12; // הדק הארדואינו שמספק את דפקי השעון לאוגר ההזזה הטורי
int dataPin = 11; // הדק הארדואינו שמספק את הנתון הטורי לאוגר ההזזה הטורי

void setup() {
    pinMode(latchPin, OUTPUT);

```

```

pinMode(clockPin, OUTPUT);

pinMode(dataPin, OUTPUT);

}

int arr[] = {0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7c, 0x07, 0x7f, 0x67};

void loop() {

    for(int i=0; i<10; i++) {

        digitalWrite(latchPin, LOW); // ST_CP להדק '0' התחלתי
        shiftOut(dataPin, clockPin, MSBFIRST, arr[i]); // הפקודה שמסדרת את המידע הטורי
        digitalWrite(latchPin, HIGH); // ST_CP שפעיל בעלית שעון
        delay(500);

    }

}

```

23. שנה את התוכנית מסעיף 22, כך ש:
א. הספרות ידלקו האחת אחרי השניה עד לספרה 9.

ב. הספרה 9 תהבהב 5 פעמים בתדר של 4Hz

ג. הספירה תרד חזרה מ 9 ל 0.

ד. הספרה 0 תהבהב 5 פעמים בתדר של 4Hz

24. התוכנית שלהלן גורמת 'ללד רץ' (לכיוון שמאל) כמוראה באיור שלהלן.



```

int latchPin = 8;

int clockPin = 12;

int dataPin = 11;

void setup() {

    pinMode(latchPin, OUTPUT);

    pinMode(clockPin, OUTPUT);

```

```

pinMode(dataPin, OUTPUT);

}

void loop() {

  for(int i=0x20;i>0;i=i>>1) {

    digitalWrite(latchPin, LOW);

    shiftOut(dataPin, clockPin, MSBFIRST, i);

    digitalWrite(latchPin, HIGH);

    delay(500);

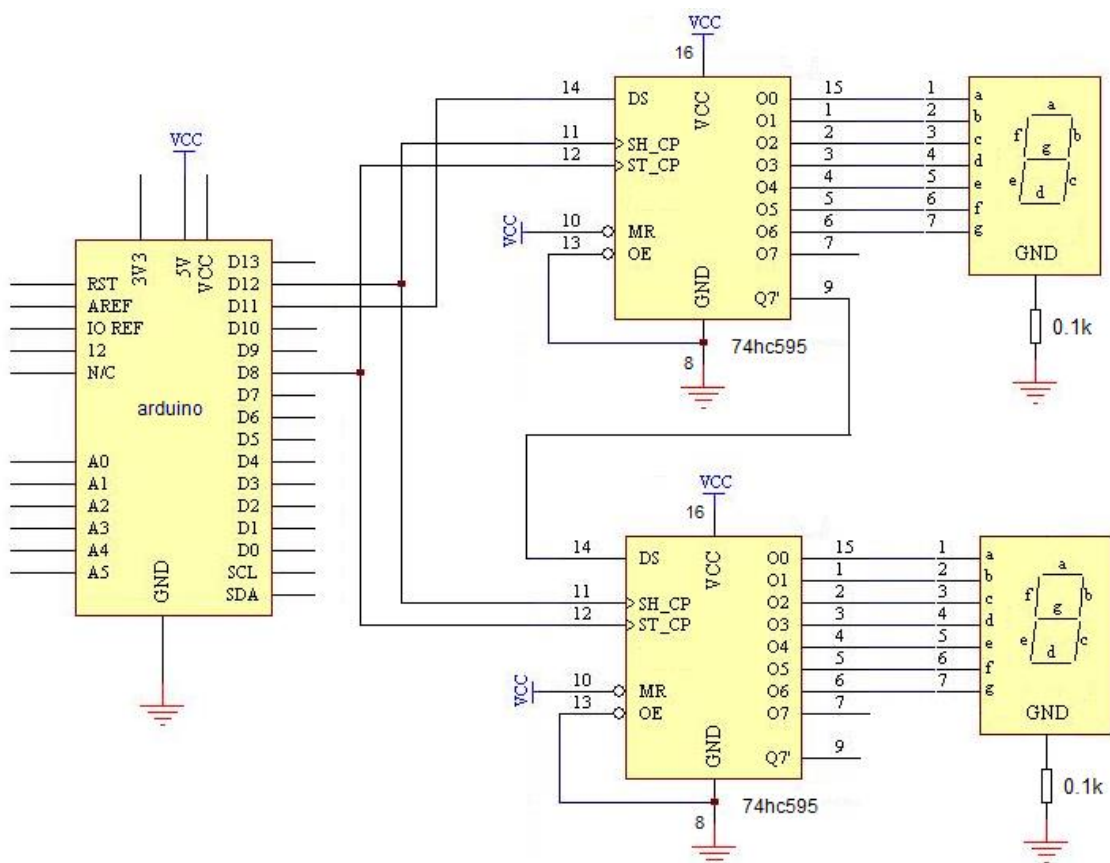
  }

}

```

25. שנה את ההוראה $i=i>>1$ (שבפקודת for) ל $i=(i>>1)+0x20$. בדוק והסבר את התוצאה.

ניתן להוסיף (לשרשר) עוד ועוד רכיבי 74hc595 זאת מבלי 'לבזבז' הדקים של המיקרו בקר.
 כמראה בשרטוט שלהלן.



26. התוכנית שלהלן מציגה ספירה עד 99. כתוב ובדוק.

```
int latchPin = 8;
int clockPin = 12;
int dataPin = 11;

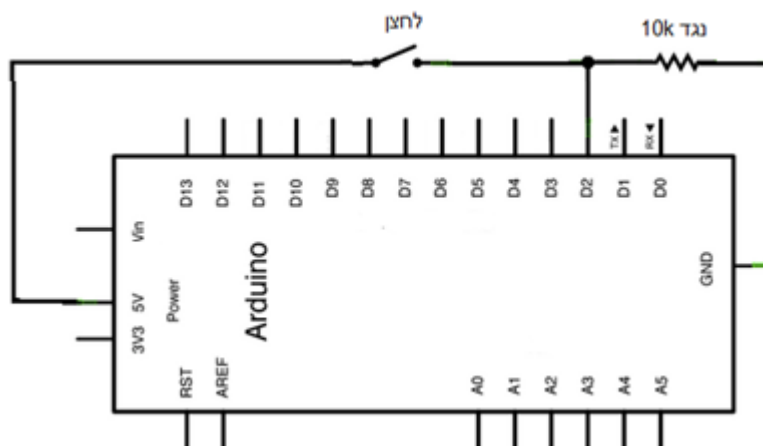
void setup() {
  pinMode(latchPin, OUTPUT);
  pinMode(clockPin, OUTPUT);
  pinMode(dataPin, OUTPUT);
}

int arr[] = {0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7c, 0x07, 0x7f, 0x67};
void loop() {
  for(int i=0; i<100; i++) {
    digitalWrite(latchPin, LOW);
    shiftOut(dataPin, clockPin, MSBFIRST, arr[i/10]); // הצג ספרת העשרות
    shiftOut(dataPin, clockPin, MSBFIRST, arr[i%10]); // הצג ספרת האחדות
    digitalWrite(latchPin, HIGH);
    delay(500);
  }
}
```

אל תפרק את המעגל

פעילות 3 - הוספת לחצן אל הארדואינו ושימוש במוניטור

1. חבר 'לחצן' אל הדק 2 של הכרטיס לחצן כמתואר באיור 1.



איור 1

2. התוכנית שלהלן מדליקה את הled המחובר לבדק 13 רק כאשר הלחצן במצב 'לחוץ' (1).

```
void setup() {
  pinMode(13, OUTPUT); // קבע את הדק 13 למצב 'מוצא'
  pinMode(2, INPUT); // קבע את הדק 2 למצב 'מבוא'
}

void loop(){
  if (digitalRead(2) == 1) // אם הלחצן במצב '1' (הלחצן במצב 'לחוץ')
    digitalWrite(13, HIGH); // הדלק את הled
  else // אחרת (אם הלחצן לא במצב '1')
    digitalWrite(13, LOW); // כבה את הled
}
```

3. שנה את התוכנית. כך שכאשר הלחצן במצב 'לחוץ' (1), הled יבהבה. כאשר הled לא לחוץ (0) הled יהיה כבוי.

```
void setup() {
  pinMode(13, OUTPUT); // קבע את הדק 13 למצב 'מוצא'
  pinMode(2, INPUT); // קבע את הדק 2 למצב 'מבוא'
}

void loop(){
  if (digitalRead(2) == 1) { // אם הלחצן במצב '1' (לחצן במצב 'לחוץ')
    digitalWrite(13, HIGH); // הדלק את הled שבהדק 13
    delay(200); // המתן 0.2 שניה
    digitalWrite(13, LOW); // כבה את הled
  }
```

```

delay(200);          // המתן 0.2 שניה
}
else                  // אחרת (אם הלחצן לא במצב '1')
digitalWrite(13,0);  // כבה את הלד
}

```

דרך נוספת לבדוק את מצב הלחצן הוא באמצעות 'המוניטור' של המחשב. לכרטיס הארדואינו יש 'ערוץ תקשורת טורית' עם המחשב. המיקרובקר יכול לשלוח למחשב מידע שיוצג על גבי מסך המחשב. כדי שהמיקרובקר יתקשר עם המחשב בתקשורת טורית, צריך להגדיר את 'קצב התקשורת'. הפקודה Serial.begin() קובעת את קצב התקשורת. לדוגמא הפקודה Serial.begin(9600); פירושה: "קבע קצב תקשורת ל 9600 סיביות בשניה".

כדי שהמיקרובקר ידפיס מידע על גבי המוניטור, משתמשים בפקודה Serial.print(). בתוך הסוגריים מציינים מה רוצים שיודפס על גבי המוניטור. אם רוצים להדפיס טקסט, הטקסט צריך להיות בתוך מרכאות כפולות. למשל: הפקודה Serial.print("MOTI"); תדפיס על המסך את המילה MOTI. הפקודה Serial.println("GADI &"); תדפיס על מסך המחשב את המילה GADI & ולאחר מכן תרד שורה (בגלל התוספת של \n);

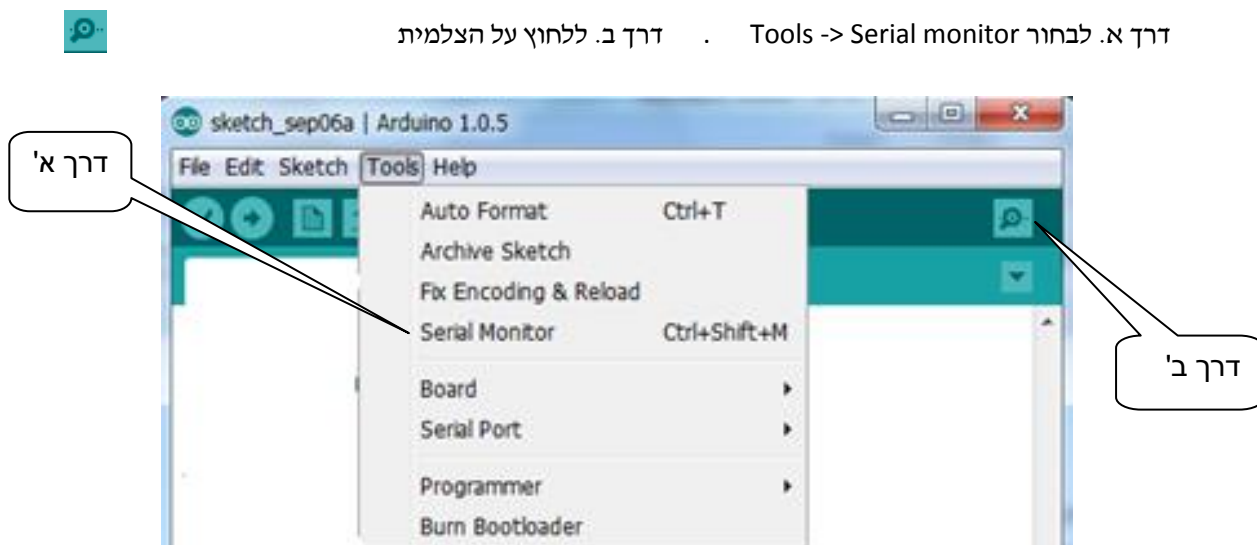
4. כתוב וטען את התוכנית למיקרובקר:

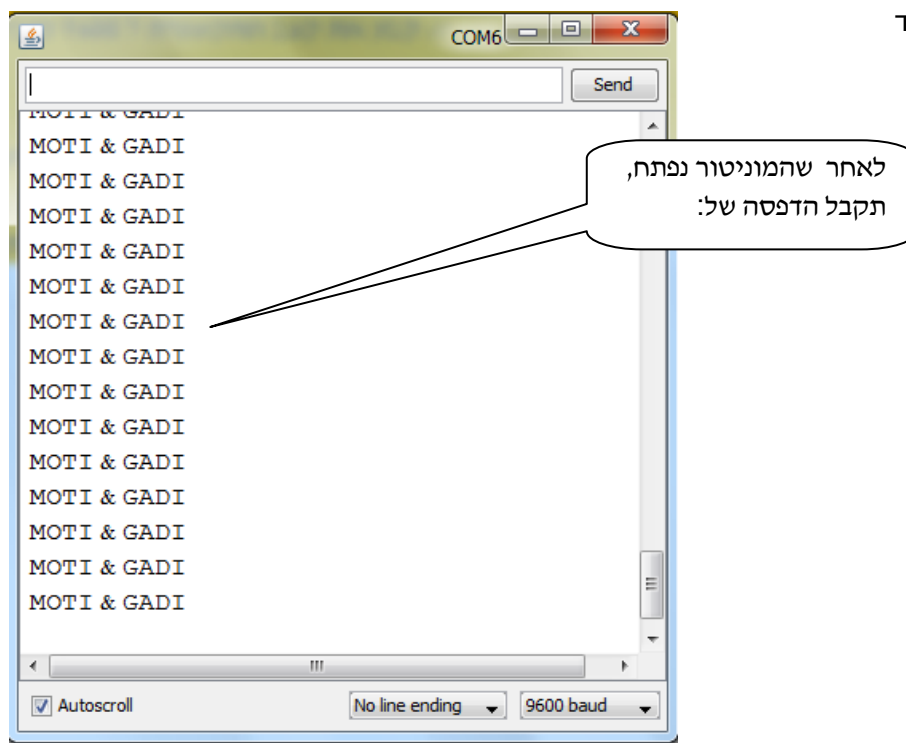
```

void setup() {
  pinMode(2,INPUT); // קבע את הדק 2 למצב 'מבוא'
  Serial.begin(9600); // קבע את קצב התקשורת ל 9600 סיביות בשניה
}
void loop() {
  Serial.print(" MOTI"); // הדפס את הטקסט שנמצא בין הגרשיים
  Serial.println(" & GADI"); // הדפס את הטקסט ועבור שורה
  delay(200);          // המתן 200 אלפיות שניה, כדי שנספיק לראות מה כתוב
}

```

לאחר שהתוכנית הועלתה אל זיכרון המיקרובקר. אפשר לפתוח את המוניטור ב 2 דרכים (כמוראה באיור 2).
 דרך א. לבחור Serial monitor -> Tools. דרך ב. ללחוץ על הצלמית





אם המוניטור לא מציג את הטקסט, לחץ על לחצן **reset** שעל כרטיס הארדואינו.

5. התוכנית שלהלן תדפיס על המוניטור את מצב הלחצן. כתוב, טען אל המיקרובקר ובדוק.

```
void setup() {
  pinMode(2,INPUT); // קבע את הדק 2 למצב 'מבוא'
  Serial.begin(9600); // קבע את קצב התקשורת ל 9600 סיביות בשניה
}

void loop() {
  Serial.print(" Button=" ); // הדפס את הטקסט שנמצא בין הגרשיים
  Serial.println(digitalRead(2)); // הדפס את המצב בהדק 2 ועבור שורה
  delay(200); // המתן 200 אלפיות שניה כדי שנספיק לראות מה כתוב
}
```



6. ברצוננו לכתוב תוכנית שבכל לחיצה הלד שבהדק 13 יחליף את מצבו. המיקרובקר עובד במהירויות גבוהות (כל פקודה מתבצעת בכמיליונית השניה). כדי להבטיח שעבור לחיצה בודדת, תתבצע פעולה אחת בלבד. נכתוב את ההוראות בדרך הבאה:

```
int red=13 // הגדר משתנה בשם red וקבע את ערכו ל 13
void setup() {
  pinMode(red, OUTPUT); // הגדר את הדק 13 כמוצא
  pinMode(2, INPUT); // הגדר את הדק 2 כמבוא
}
void loop(){
  digitalWrite(red, LOW); // כבה את הלד
  while (digitalRead (2) ==0) ; // המתן (מצב '0'), המתן
  while (digitalRead (2) ==1) ; // המתן (מצב '1'), המתן
  digitalWrite(red, HIGH); // הדלק את הלד
  while (digitalRead (2) ==0) ; // המתן (מצב '0'), המתן
  while (digitalRead (2) ==1) ; // המתן (מצב '1'), המתן
}
```

7. התוכנית שלהלן סופרת 'לחיצות'. המיקרובקר יציג על גבי המוניטור את 'מספר הלחיצות' הן בתצוגה עשרונית והן בתצוגה בינארית.

לאחר שתכתוב ותטען את התוכנית למיקרובקר. תקבל על גבי המוניטור את מספר הלחיצות כמוראה באיור הבא:

תצוגה עשרונית	תצוגה בינארית
DEC	BIN
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110

התוכנית :

```

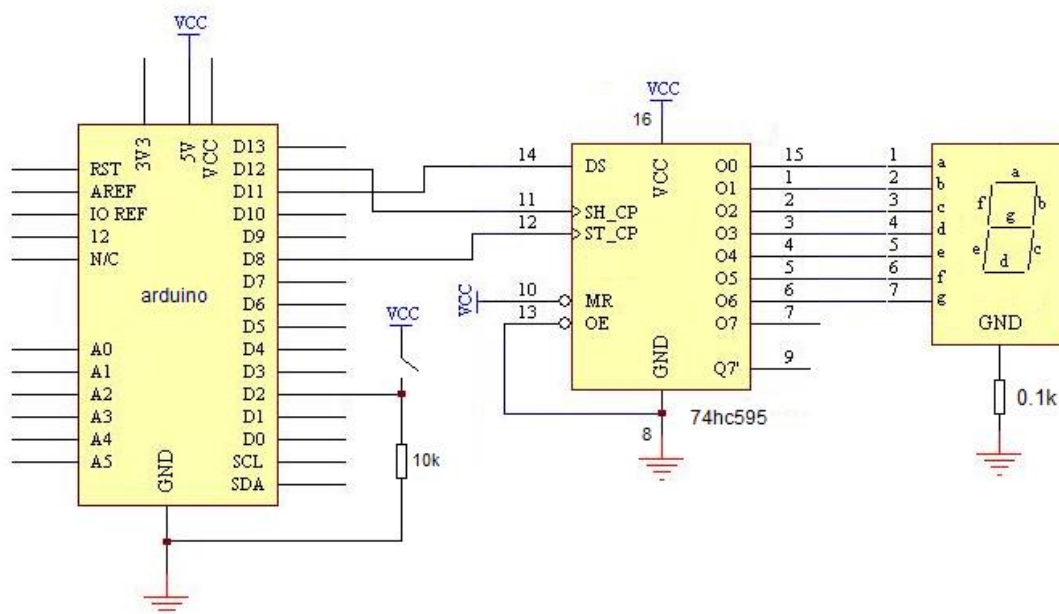
int x = 0; // הגדר 'משתנה' בשם x וקבע בו ערך התחלתי 0
void setup() {
    Serial.begin(9600); // פתח ערוץ תקשורת טורית בקצב 9600 סיביות לשניה
    Serial.print("DEC"); // הדפס את הטקסט שבין הסוגריים (הדפס את המילה DEC)
    Serial.print("\t"); // הדפס רווח TAB (ליצור רווח בין הכותרות)
    Serial.print("BIN"); // הדפס את המילה BIN
    Serial.print("\t"); // הדפס רווח (הדפס TAB)
    Serial.print("\n"); // הדפס "עבור שורה"
}
void loop() {
    while (digitalRead(2)==0); // כל עוד הלחצן במצב 'לא לחוץ', המתן
    x++; // הוסף למשתנה אחד. כלומר x=x+1
    Serial.print(x,DEC); // הצג את התוכן של משתנה x בפורמט עשרוני
    Serial.print("\t"); // הדפס רווח של TAB

    Serial.print(x, BIN); // הצג את התוכן של משתנה x בפורמט בינארי
    Serial.println("\t"); // הדפס רווח של TAB

    while (digitalRead(2)==1); // כל עוד הלחצן במצב 'לחוץ', המתן
}

```

8. חבר לכרטיס הארדואינו תצוגת 7 מקטעים ולחצן כמתואר בשרטוט.



9. כתוב תוכנית אשר תציג את הספרה 1 כאשר הלחצן במצב 'לחוץ' ואת הספרה 0 כאשר הלחצן במצב לא לחוץ.
10. כתוב תוכנית אשר תספור את מספר הלחיצות.
 - א. המצב ההתחלתי של התצוגה הוא 0.
 - ב. בכל לחיצה התצוגה מתקדמת באחד עד לספרה 9. בלחיצה העשירית, התצוגה שוב מתאפסת.
11. כתוב תוכנית אשר תגרום לספרה 0 להבהב (בתדר של 4HZ) . מספר ההבהובים תלוי במספר הלחיצות.
 - המצב ההתחלתי של התצוגה הוא 0.
 - אחרי הלחיצה הראשונה התצוגה 0 תהבהב פעם אחת.
 - אחרי הלחיצה השנייה התצוגה 0 תהבהב פעמיים.
 - אחרי הלחיצה השלישית התצוגה 0 תהבהב שלוש פעמים וכך הלאה.....

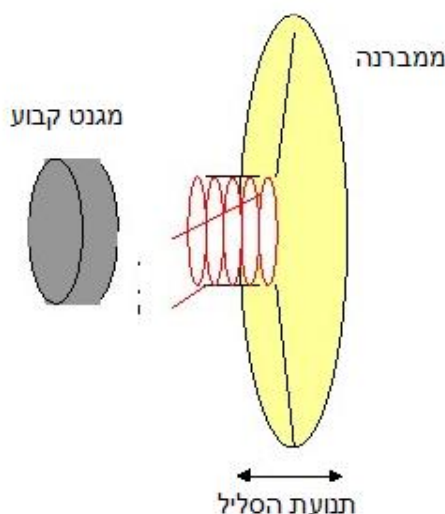
פעילות 4 – יצירת תדר וחיבור רמקול

מבוא

בני האדם מתקשרים ביניהם בין היתר באמצעות השמעת צלילים (גלי קול). **תדירות** גלי הקול נמדדת ביחידות **הרץ** כלומר, מספר המחזורים של גל הקול בשנייה.

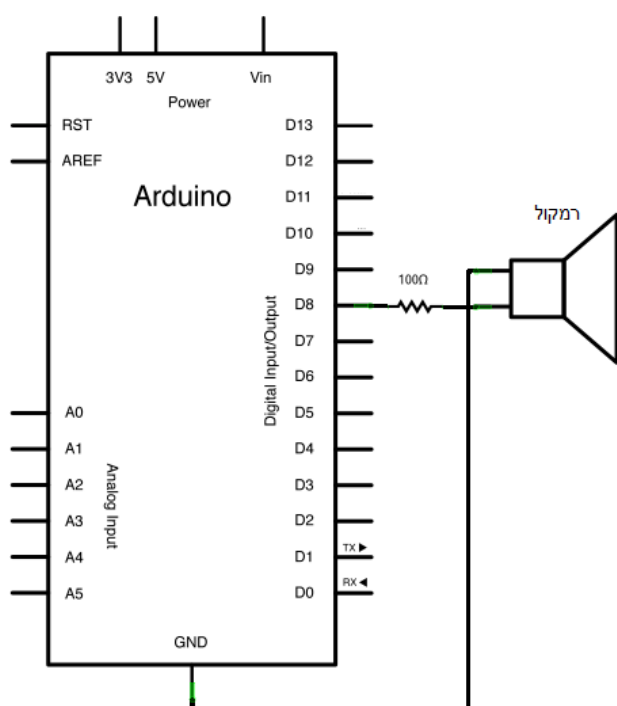
גובה הצליל (בשונה מעוצמת הצליל) קשורה לתדר של גל הקול. ככל שהתדר של הצליל גבוה יותר, כך הצליל נשמע גבוה יותר. אוזן אדם רגישה לתדרים שבין 20hz ל 20khz. בפעילות זו ניצור צלילים בעזרת רמקול. 'רמקול' הוא מתמר המשמש להפיכת אותות חשמליים לצלילים שניתן לשמוע.

מבנה הרמקול



הרמקול מבוסס על התכונה הפיסיקאלית של סליל שאם זורם בו זרם חשמלי הוא הופך למגנט (אלקטרו מגנט). הסליל שברמקול נמצא בתוך חלל של מגנט קבוע. ולכן כאשר זורם בסליל זרם בתדר מסויים, הסליל נמשך או נדחה למגנט הקבוע בתדר המסויים, אל הסליל מודבקת ממברנה (משטח קרטון או חומר סינתטי, שנראה כמו משפך שטוח), שמזיזה את האוויר (בתדר של תנועת הסליל) ויוצרת את גלי הקול.

1. חבר את הרמקול לכרטיס להדק D8 כמתואר באיור 1.



2. ניצור גל ריבועי סימטרי בעזרת הפונקציה: `tone(pin, frequency)`
 הפונקציה מקבלת כפרמטר את מספר ההדק (ה `pin`) שבו אנו רוצים ליצור את הצליל ואת התדר הרצוי (ה `frequency`)
 לדוגמא:

```
void setup(){
  pinMode(8,OUTPUT); // קבע את הדק 8 כ 'מוצא'
}
void loop(){
  tone(8,500); // קבע בהדק 8 תדר של 500 הרץ
}
```

3. התוכנית שלהלן משמיעה שני צלילים.

```
void setup(){
  pinMode(8,OUTPUT);
}

void loop(){
  tone(8,500);
  delay(1000);
  tone(8,800);
  delay(1000);
}
```

4. ניתן להגביל זמן השמעת הצליל על ידי הוספת פרמטר של 'משך הזמן' (`duration`)
`tone(pin, frequency, duration)`
 לדוגמא: נוסף לתוכנית הקודמת `duration` של 1/2 שניה.

```
void setup(){
  pinMode(8,OUTPUT);
}

void loop(){
  tone(8,500,500); // קבע בהדק 8 תדר של 500 הרץ למשך 500 אלפיות השניה
  delay(1000);
  tone(8,800,500);
  delay(1000);
}
```

שים לב: חייבים לרשום את פקודת `delay()` וצריך להתקיים ש: `duration < delay()`

5. ככל שהתדר גבוה יותר כך הצליל נשמע גבוה יותר. התוכנית שלהלן גורמת לצליל עולה ויורד.

```
void setup(){
  pinMode(8,OUTPUT);
}
```

```

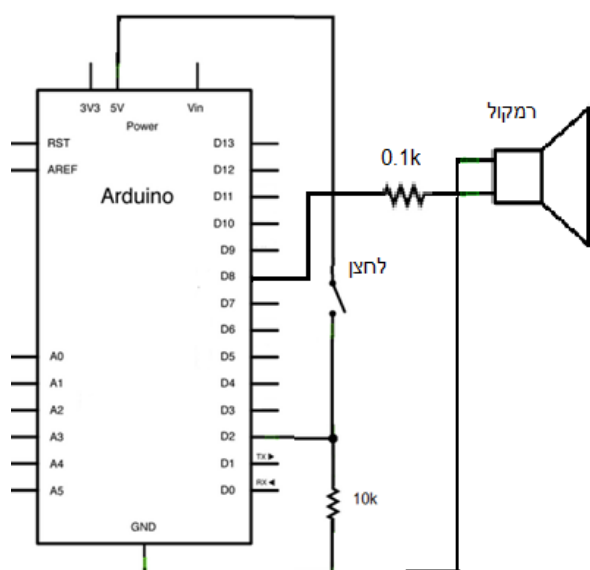
}

void loop(){
  for (int i=200;i<1000;i+=5){
    tone(8,i);
    delay(20);
  }
  for (int i;i>200;i-=5){
    tone(8,i);
    delay(20);
  }
}

```

6. שנה בתוכנית את הערכים של i ובדוק את התוצאה...

7. מכשיר לזיהוי "בעיות שמיעה". אנו מעוניינים לפתח מכשיר שיוכל למדוד עד איזה תדר 'הנבדק' שומע. במילים אחרות מכשיר שמודד באיזה 'גובה צליל' הנבדק מפסיק לשמוע. ראשית, נוסיף למערכת שלנו 'לחצן' להדק 2 כמתואר באיור.



נכתוב את האלגוריתם הבא:

- השמע צליל התחלתי נמוך.
- אם הלחצן במצב 'לא לחוץ' המתן.
- הצג את תדר הצליל הנוכחי על המוניטור.
- העלה את גובה הצליל.
- המתן עד שהמפסק חוזר למצב 'לא לחוץ'.
- חזור לשלב b.

מימוש האלגוריתם

ערך התחלתי של הצליל // `int sound=4000;`

```

void setup() {
  pinMode(2,INPUT);    // קבע את הדק 2 כמבוא
  pinMode(8,OUTPUT);   // קבע את הדק 8 כמוצא
  Serial.begin(9600);  // פתח ערוץ תקשורת טורית בקצב 9600 סיביות לשניה
}

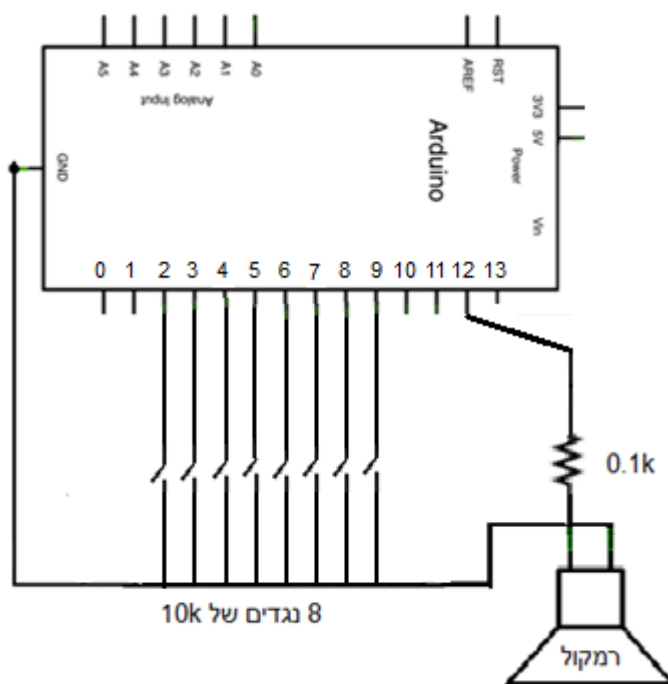
```

```

void loop() {
  while (digitalRead(2)==0); // כל עוד הלחצן במצב 'לא לחוץ' המתן
  delay(50); // המתנה להפסקת ה 'רטט' של הלחצן
  tone(8,sound); // קבע בהדק 8 תדר לפי הערך שב sound
  Serial.print("sound = "); // הדפס על המוניטור את הטקסט: sound =
  Serial.println(sound); // הדפס את הערך שיש ב sound ועבור שורה
  sound+=200; //sound=sound+200. 200 sound למשתנה
  while(digitalRead(2)==1); // כל עוד הלחצן במצב 'לחוץ' המתן.
  delay(50); // המתנה להפסקת ה 'רטט' של הלחצן
}

```

8. ה"מיני אורגן" שלהלן הוא כלי נגינה בעל סולם של 8 צלילים (דו-רה-מי-פה-סול-לה-סי-דו). חבר את הרמקול להדק 12 ואת 8 הלחצנים להדקים 2-9 כמשורטט:



המבוא יחובר לנגד pull up פנימי. לכן, כאשר לחצן במצב 'פתוח' נקבל במבוא '1'. כאשר הלחצן במצב 'סגור' נקבל '0'.

התדרים עבור התווים (ביחידות Hz) הם:

Do2	Se	La	Sol	Fa	Me	Re	Do
523	494	440	392	349	330	294	262

9. התוכנית שלהלן מממשת את המיני אורגן כתוב, טען אל המיקרובקר ובדוק

```
#define Do    262 //    262 הוא יוחלף במספר
#define Re    294 //    294 הוא יוחלף במספר
#define Me    330 //    330 הוא יוחלף במספר
#define Fa    349 //    כנ"ל.....
#define Sol   392
#define La    440
#define Se    494
#define Do2   523

void setup()
{
  for (int i=2;i<10;i++){
    pinMode(i,INPUT_PULLUP); // כהדקי מבוא 2,3,4,5,6,7,8,9
    Serial.begin(9600);
  }
  pinMode(12,OUTPUT); // קבע את הדק 12 כהדק מוצא
}
void loop()
{
  if(digitalRead(2)==0) // אם הדק 2 במצב 'לחוץ'
    tone(12,Do); // הוצא להדק 12 את התדר של Do
  else if(digitalRead(3)==0) // אם הדק 3 במצב 'לחוץ'
    tone(12,Re); // הוצא להדק 12 את התדר של Re
  else if(digitalRead(4)==0) // אם הדק 4 במצב 'לחוץ'
    tone(12,Me); // הוצא להדק 12 את התדר של Me
  else if(digitalRead(5)==0)
    tone(12,Fa);
  else if(digitalRead(6)==0)
    tone(12,Sol);
  else if(digitalRead(7)==0)
    tone(12,La);
  else if(digitalRead(8)==0)
    tone(12,Se);
  else if(digitalRead(9)==0)
    tone(12,Do2);
  else // אם אף לחצן לא במצב 'לחוץ'
    noTone(12); // (הפסק את התדר בהדק 12 (הפסק השמעת צליל)
}
```

10. ברצוננו שהמיקרובקר ישמיע מנגינה כלשהי. למשל: המנגינה של "דוד מלך ישראל חי וקיים"....

מצאנו שהמנגינה "דוד מלך ישראל חי וקיים" בנויה מהתווים שלהלן:

סול, מי, סול, סול, סול, מי
 סול, לה, סול, פה, מי, רה,
 דו, דו, רה, רה, דו, דו, רה,
 דו, פה, מי, רה, מי,
 דו, דו, רה, רה, דו, דו, רה,
 דו, פה, מי, רה, דו.

בהתאם לכך כתבנו תוכנית שמשמיעה את המנגינה.

התווים : דו, דו, רה, רה, דו, דו, רה, חוזרים על עצמם פעמיים.

גם התווים: דו, פה, מי, רה. חוזרים על עצמם פעמיים.

לכן, במקום לכתוב את אותו דבר פעמיים (ולקבל תוכנית ארוכה), כתבנו לכל רצף שכזה 'פונקציה'.

התוכנית:

```
#define DO 262 // 262 הוא יוחלף במספר
#define RE 294
#define ME 330
#define FA 349
#define SOL 392
#define LA 440
#define SE 494
#define DO2 523
```

```
long time=600; // יחידת הזמן הבסיסית להשמעת 'תו' הוא 0.6 שניה
void st(){ // st() היא פונקציה שעוצרת את הצליל למשך 0.03 שניה
    noTone(12); // עצור את התדר בהדק 12
    delay(30); // המתן 0.03 שניה
}
```

//-----פונקציה שתשמיע את התווים : דו, פה, מי, רה.-----

```
void do_fa_me() {
    tone(12,DO); // DO לרמקול התדר של
    delay (time); // השמע את הצליל למשך 0.6 שניה
    tone(12,FA); // FA לרמקול התדר של
    delay (time); // השמע את הצליל למשך 0.6 שניה

    tone(12,ME);
    delay (time/2); // השמע את הצליל 0.3 שניה
    tone(12,RE);
    delay (time/2);
}
```

//-----פונקציה שתשמיע את התווים : דו, דו, רה, רה, דו, דו, רה.-----

```
void do_do_re() {
```

```

    for (int i=0;i<2;i++) { // השמעת הצליל DO , עם זמן של 0.3 שניה, פעמיים
        tone(12,DO);
        delay (time/2);
        st();
    }
    for (int i=0;i<2;i++) { // השמעת הצליל RE , עם זמן של 0.3 שניה, פעמיים
        tone(12,RE);
        delay (time/2);
        st();
    }
    for (int i=0;i<2;i++){
        tone(12,DO);
        delay (time/2);
        st();
    }
    tone(12,RE);
    delay (time);
}

// -----פונקצית setup() של ארדואינו-----
void setup(){
    pinMode(12,OUTPUT);
}
// -----פונקצית loop() של ארדואינו-----
void loop (){
    //-----השמע את : סול, מי, סול, סול, סול, סול, מי-----
    tone(12,SOL);
    delay (time);
    tone(12,ME);
    delay (time);
    for( int i=0;i<4;i++){
        tone(12,SOL);
        delay (time/4);
        st();
    }
    tone(12,ME);
    delay (time)
    //-----השמע את : סול, לה, סול, פה, מי, רה,-----
    tone(12,SOL);
    delay (time);

    tone(12,LA);
    delay (time);

    tone(12,SOL);
    delay (time/2);
}

```

```

tone(12,FA);
delay (time/2);

tone(12,ME);
delay (time/2);

tone(12,RE);
delay (time/2);
st();
//-----השמעת , דו, דו, רה, רה, דו, דו, רה-----
do_do_re();
//-----השמעת דו,פה, מי, רה, מי-----

do_fa_me();
tone(12,ME);
delay (time);
st();

//-----השמעת , דו, דו, רה, רה, דו, דו, רה-----
do_do_re();
//-----השמעת דו,פה, מי, רה, דו-----
do_fa_me();

tone(12,DO);
delay (time);

st();
delay(2000);
}

```

11. שנה את התוכנית כך שהמנגינה תושמע רק אחרי לחיצה על הלחצן שמחובר להדק 2 .

12. בחר מנגינה 'כלשהי'...וכתוב תוכנית....בהצלחה!!!

פעילות 5 - המרה מאנלוגי לדיגיטלי ADC - Analog to Digital

מיקרובקרים מבצעים פעולות בקרה על תהליכים בסביבה. לדוגמא המיקרובקר שבמזגן מקבל את הטמפרטורה מהסביבה ובהתאם לכך מגביר או מקטין את החימום/קירור. המיקרובקר שמטפל במערכת ההשקיה, מקבל את מצב הלחות של הקרקע ובהתאם לכך מגביר או מקטין את כמות המים בהשקיה. כידוע המיקרובקר פועל באופן 'דיגיטלי' כלומר, המיקרובקר 'מבין' שני מצבים בלבד, מצב של '1' ומצב של '0'. מצד שני אנו יודעים שהשינויים בסביבה הטבעית הם לא בקפיצות של 'יש' ('1') או 'אין' ('0') אלא הם קורים ברצף. טמפרטורה, לחות, מתח, רעש וכו' משתנים **ברציפות** (משתנים בצורה אנאלוגית).

כדי שהבקר יוכל לעבד מידע של אותות פיסיקליים כמו: טמפרטורה, לחץ, רעש, מהירות, לחות וכו' עלינו לבצע שתי פעולות.

א. להמיר את האות הפיסיקלי של טמפרטורה, לחץ, רעש, מהירות וכו' לאות חשמלי של מתח (או זרם). ההמרה נעשית באמצעות מתמר שמכיל 'חיישן'. לדוגמא, בעזרת 'חיישן טמפרטורה' אנו ממירים את שינויי הטמפרטורה לשינויי מתח.

ב. להפוך את הערך האנלוגי לערך דיגיטלי.

פעולת המרה מאנלוגי לדיגיטלי מתבצעת באמצעות רכיב שנקרא ADC - Analog to Digital Converter. הממיר הופך את הערך האנלוגי בכניסת המיקרובקר **למספר בינארי** (ערך דיגיטלי). הדיוק של ההמרה ובמילים אחרות כושר ההבחנה (הרזולוציה) של הממיר תלוי בשינויים גורמים.

1. במתח הייחוס - המתח אותו רוצים להמיר.

2. במספר הסיביות שיש לממיר (גודל המספר הבינארי שנקבל מהממיר).

במיקרובקר שלנו מתח הייחוס הוא 5v. והמספר הבינארי שמתקבל מהממיר הוא בן 10 סיביות.

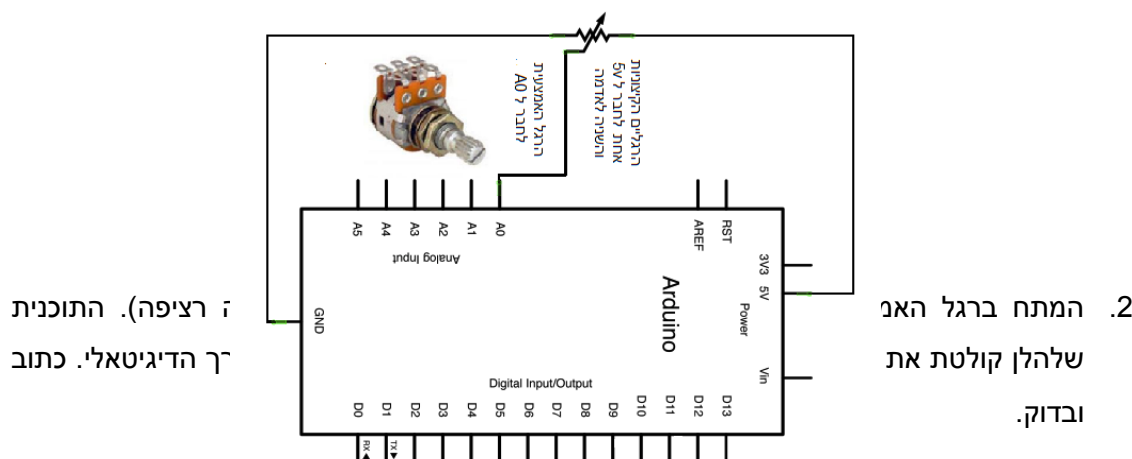
$$\text{לכן כושר ההבחנה הוא: } 0.00488v = 4.88mV = \frac{5}{1024} = \frac{5v}{2^{10}}$$

כדי לחשב את המספר הדיגיטלי שנקבל מהממיר (מה ADC) עבור ערך אנלוגי כלשהו, נחלק את הערך האנלוגי בערך של 'הרזולוציה'. לדוגמא: עבור ערך של 3.8v הממיר יציג ערך דיגיטלי של

$$.778 = \frac{3.8v}{4.88mv}$$

לכרטיס ה-arduino uno יש 6 ערוצי ADC המסומנים על הכרטיס ב A0 עד A5. כלומר, אפשר לחבר לכרטיס שישה חיישנים אנאלוגיים במקביל.

1. ברשותך פוטנציומטר. חבר את הפוטנציומטר לכרטיס הארדואינו כמשורטט:




```
void setup(){
    Serial.begin(9600); // פתח ערוץ תקשורת טורית בקצב של 9600 סיביות לשניה
}
```

```
void loop(){
    int potentiometer; // הגדר משתנה בשם potentiometer מסוג int
    potentiometer=analogRead(A0); //A0 מערוץ האנלוגי
    Serial.println(potentiometer); // הצג על המוניטור את הערך של המשתנה
    delay (100); // המתן עשירית השניה
}
```

3. עבור 5v נקבל במינטור הוא 1023. בדוק! (מדוד המתח בעזרת הרב מודד)
4. מהו לדעתך הערך הדיגיטלי שיתקבל במוניטור עבור מתח של 2.5v ?
5. כוון את המתח של הפוטנציומטר לערך של 2.5v. (במוניטור צריך להופיע הערך הדיגיטלי שמצאת בסעיף הקודם. מדוד את המתח גם בעזרת הרב מודד)
6. מהו לדעתך הערך הדיגיטלי שיתקבל במוניטור עבור מתח של 3.5v ?
7. כוון את המתח של הפוטנציומטר לערך של 3.5v. (במוניטור צריך להופיע הערך הדיגיטלי שמצאת בסעיף הקודם. מדוד את המתח גם בעזרת הרב מודד)

הערה: במקום 3 הפקודות:

```
int potentiometer;
potentiometer=analogRead(A0);
Serial.println(potentiometer);
```

אפשר לכתוב פקודה אחת:

```
Serial.println(analogRead(A0)); // הצג על המוניטור את הערך המתקבל מערוץ A0
```

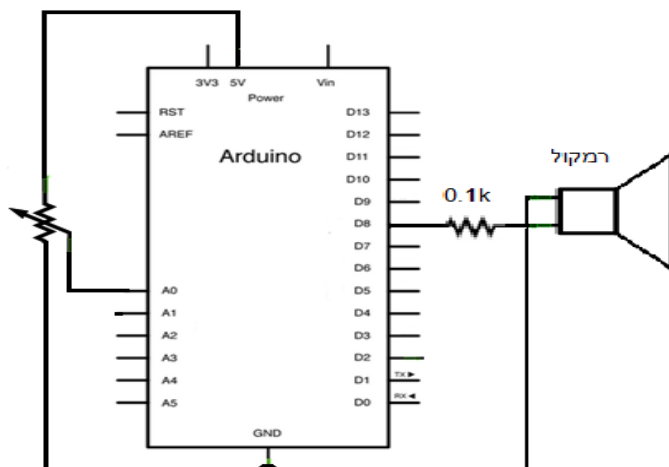
8. התוכנית שלהלן גורמת ללד שמחובר להדק 13 להבהב בקצב שתלוי במצב הפוטנציומטר. (הערך הדיגיטלי שמתקבל מהממיר ADC קובע את זמן השהייה בין הדלקה וכיבוי של הלד.) לאחר טעינת התוכנית למיקרו, פתח את המוניטור ובדוק את קצב ההבהוב של הלד בתלות הפוטנציומטר.

```
int timeForDelay; // הגדר 'משתנה' בשם timeForDelay
void setup() {
    pinMode(13, OUTPUT); // הגדר את הדק 13 כמוצא
    Serial.begin(9600); // פתח ערוץ תקשורת טורית בקצב של 9600 סיביות לשניה
}

void loop() {
    timeForDelay=analogRead(A0); // הכנס אל המשתנה את הערך שהתקבל מהממיר
    Serial.println(timeForDelay); // הצג על המוניטור את הערך של המשתנה
}
```

```
digitalWrite(13, HIGH);          // הדלק את הלד
delay(timeForDelay);            // ההשהייה לפי הערך שיש במשתנה
digitalWrite(13, LOW);           // כבה את הלד
delay(timeForDelay);            // ההשהייה לפי הערך שיש במשתנה
}
```

9. הוסף את הרמקול שברשותך ליציאה הדיגיטלית D8 כמתואר בשרטוט שלהלן.



10. התוכניו

```
void setup() {
    pinMode(8,OUTPUT);
    Serial.begin(9600); // פתח ערוץ תקשורת טורית בקצב של 9600 סיביות לשניה
}

void loop(){
    int potentiometer; // הגדר משתנה בשם potentiometer
    potentiometer=analogRead(A0); // הכנס למשתנה את הערך שמתקבל מהממיר
    Serial.println(potentiometer); // הצג את הערך שבמשתנה בתצוגת המוניטור
    tone(8,potentiometer); // קבע בהדק 8 תדר שערכו נמצא במשתנה potentiometer
}
```

11. כתוב תוכנית שתשמיע צליל קבוע של 1KHz כאשר הערך הדיגיטלי הוא מתחת ל 500 וצליל קבוע של 2KHz כאשר הערך הוא מעל 500 .

12. כתוב תוכנית שמשמיעה צלילים בתלות המתח על פי הטבלה הבאה.

4v – 5v	3v - 4v	2v – 3v	1v – 2v	0v – 1v	תחום המתחים
5000hz	4000hz	3000hz	2000hz	1000hz	התדר

13. כתוב תוכנית המשמיעה את סולם הצלילים (דו, רה, מי, פה... וכו') בתלות המתח במבוא. הדרכה:

נחלק את מתח הייחוס (המתח המקסימאלי) ל 8 ונקבל $\frac{5v}{8} = 0.625v$. כלומר, אנו מעוניינים שבכל

שינוי של 0.625v ישתנה צליל אחד בסולם . בחישוב פשוט אפשר לראות שהתחום האנלוגי של $0\text{v}-0.625\text{v}$ הוא התחום הדיגיטאלי של 0-127 , וכך הלאה....כמוראה בטבלה שלהלן.

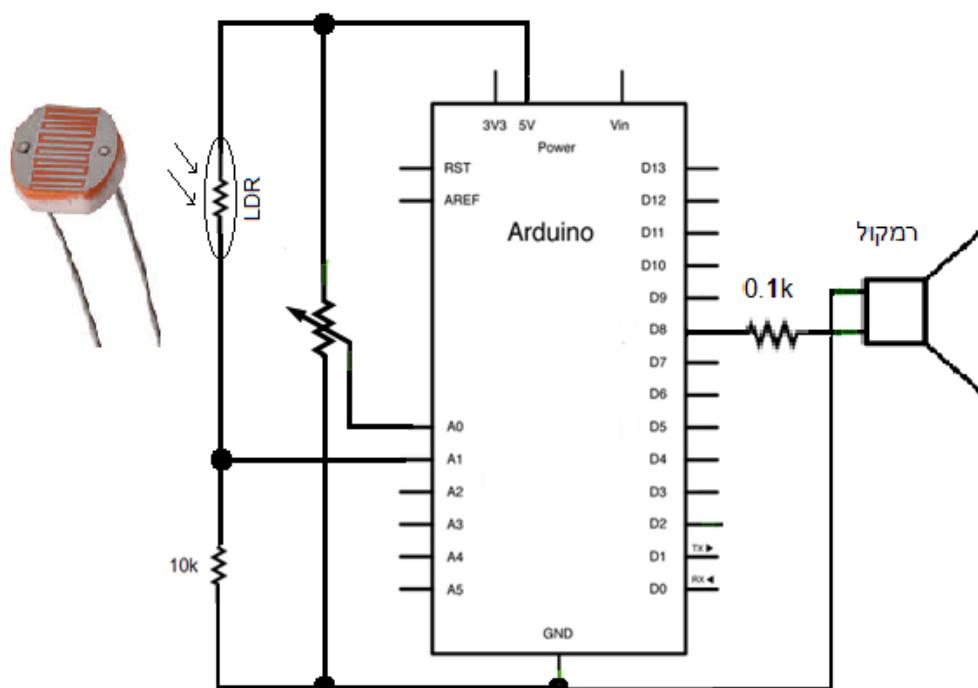
Do2	Se	La	Sol	Fa	Me	Re	Do	הצליל
523	494	440	392	349	330	294	262	התדר
4.375-5v	3.75v-4.374v	3.125v-3.74v	2.5v-3.124	1.875v-2.4v	1.25v-1.874v	0.625v-1.24v	0v-0.624v	התחום האנלוגי
896-1023	768-895	640-767	512-639	384-511	256-383	128-255	0-127	התחום הדיגיטאלי

פעילות 6 - חיבור LDR וחיישן טמפרטורה למבואות אנאלוגיים

ה LDR הוא נגד רגיש לאור. כלומר, ההתנגדות שלו תלויה בעוצמת האור. ההתנגדות של ה LDR היא ביחס הפוך לעוצמת האור. (ככל שעוצמת האור גדלה ההתנגדות יורדת). . הוסף רכיב LDR לערוץ A1 כמוראה בשרטוט שלהלן.

כאשר עוצמת האור עולה, המתח במבוא A1 עולה. ולהפך כאשר עוצמת האור יורדת, ההתנגדות של ה LDR עולה, המתח במבוא A1 יורד.

שים לב: ל LDR מחובר נגד של 10k



1. התוכנית שלהלן מציגה את הערכים הדיגיטליים המתקבלים מ'חיישן האור' במוניטור.

```
void setup(){
    Serial.begin(9600);
}
void loop(){
    Serial.println(analogRead(A1)); // A1 מערוץ
    delay(200);
}
```

2. התוכנית שלהלן תשמיע צלילים בתלות עוצמת האור. כתוב ובדוק...

```
void setup(){ // גם אם אין אתחולים. הארדואינו דורש שהפונקציה תהיה כתובה
}
void loop(){
    tone(8,analogRead(A1)); // A8 שבערוץ
}
```


6. התוכנית שלהלן מציגה במוניטור את הערך הדיגיטאלי ואת הטמפרטורה .

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

void loop(){
  double temperature;

  קבע מתח ייחוס ל 'פנימי'. (מתח הייחוס יהיה 1.1v) //
  analogReference(INTERNAL);

  המרה של המתח במוצא החיישן מאנאלוגי לדיגיטאלי //
  temperature=analogRead(A0);

  התאמה של הערך הדיגיטאלי לטמפרטורה //
  temperature= temperature*110/1023;

  Serial.print("digital=");

  הדפס את הערך הדיגיטאלי //
  Serial.print(analogRead(A0));

  Serial.print("    temp=");

  הדפס את ערך הטמפרטורה //
  Serial.println(temperature);

  delay(200);
}
```

7. הוסף להדק D8 רמקול.

שנה את התוכנית , כך שכאשר הטמפרטורה עוברת את ה 28°C , תישמע אזעקה עולה ויורדת.....

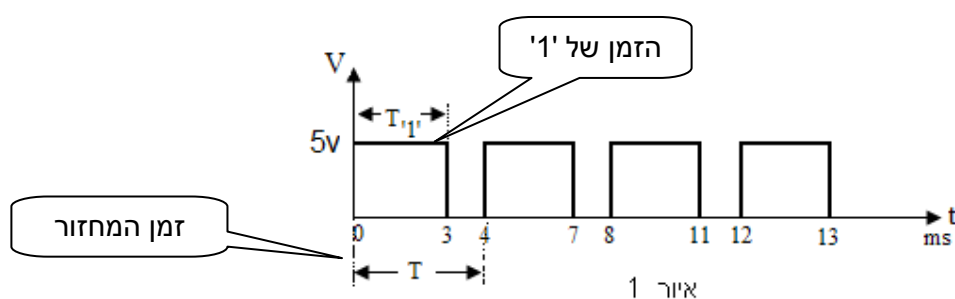
8. שנה את התוכנית מסעיף 21 כך שכאשר הטמפרטורה עוברת את ה 28°C , תישמע אזעקה עולה ויורדת..... אולם האזעקה תיפסק רק כאשר הטמפרטורה תרד מתחת ל 26°C .

פעילות ד - אפנון רוחב דופק PWM - Pulse With Modulation

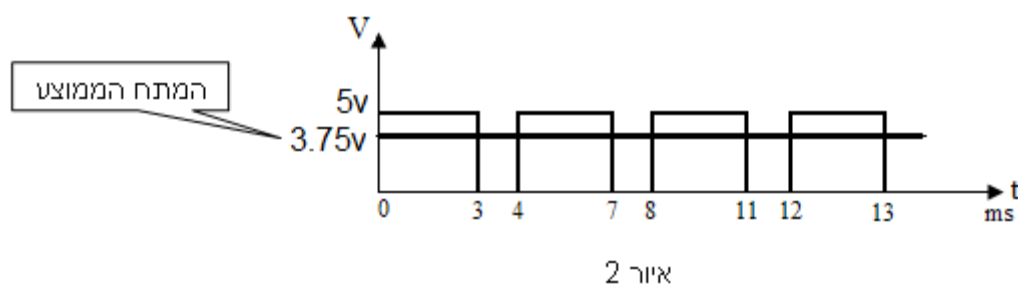
מבוא

כידוע, הבקר 'מבין' רק אותות דיגיטאליים של '0' או '1'. כלומר, שני מצבי מתח בלבד. 0v או 5v. לעיתים אנו חייבים לספק מתח שהוא שונה מ 5v. אחת הדרכים לעשות זאת היא באמצעות שיטת ה PWM (Pulse Width Modulation) בשיטה זו, אפשר לספק מתח ממוצע שהוא שונה מ 5v. בשיטה זו, הבקר אמנם מספק מתח קבוע (5v), אבל בתדר מסוים. כדי לשנות את המתח הממוצע, משנים את 'גורם המחזור' של התדר. 'גורם המחזור' (Duty Cycle) מוגדר כיחס שבין הזמן בו האות נמצא ב'1' לוגי, לבין זמן המחזור.

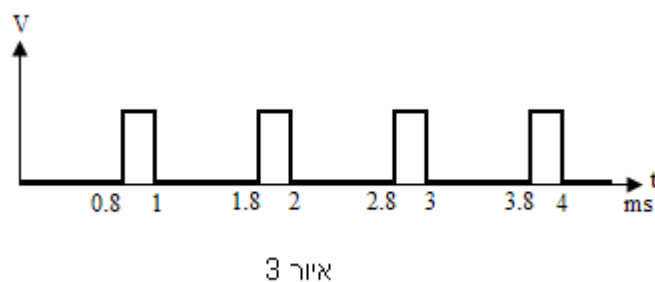
לדוגמא: באיור 1 המתח הוא אות ריבועי של 5v



זמן המחזור הוא $T = 4ms$. הזמן שבו האות נמצא ב '1' הוא $3ms$. לכן 'גורם המחזור' הוא: $3ms / 4ms = 0.75$. המתח הממוצע שנקבל הוא: $V_{\text{ממוצע}} = 5v \cdot 0.75 = 3.75v$. אספקת תדר עם מתח של 5v וגורם מחזור של 0.75 היא שוות ערך לאספקת אנרגיה חשמלית של 3.75v



דוגמא נוספת נתונה באיור 3



זמן המחזור הוא: $T = 1ms$. גורם המחזור הוא: 0.2 . כי: $T_1 / T = 0.2ms / 1ms = 0.2$.

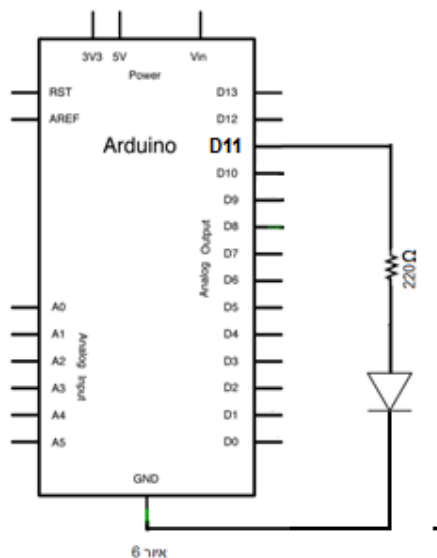
לכן המתח הממוצע של האות שבאיור 3 יהיה: $V_{\text{ממוצע}} = 5v \cdot 0.2 = 1v$

ההתנסות

לכרטיס הארדואינו אנו יש 6 ערוצי PWM. (ראה סימון ~ על הכרטיס)

הפונקציה `analogWrite(pin,pwm);` מוציאה להדק (ל pin) הנבחר תדר של 490 הרץ עם 'גורם המחזור' שהוא ערך שבין 0 ל 255 (מתוך 255). לדוגמא ההוראה: `analogWrite(11,153);` פירושה הוצא

להדק 11 אות PWM שגורם המחזור שלו הוא $\frac{153}{255}$ ולכן נקבל: $V_{\text{ממוצע}} = 5v \cdot \frac{153}{255} = 5v \cdot 0.6 = 3v$



1. חבר להדק 11 לד כמשורטט:

התוכנית שלהלן גורמת ללד שבהדק 11 להבהב עם שתי עוצמות של הארה.

```
void setup(){
}
void loop(){
    analogWrite(11,51); // מהאנרגיה (51/255) 1/5
    delay(1000);
    analogWrite(11,255); // מהאנרגיה (255/255)
    delay(1000);
}
```

2. התוכנית שלהלן גורמת ללד שבהדק 11 להידלק בעוצמה שמשתנה ברצף.

```
void setup(){
}
void loop(){
    for (int x=0; x<255; x++) {
        analogWrite(11,x); // 255 עד 0
        delay(10);
    }
}
```

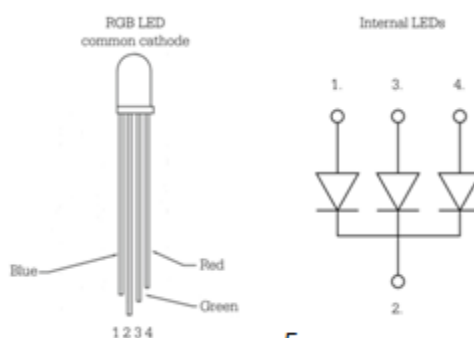

3. התוכנית שבסעיף 2 גורמת ללד להידלק לאט וברצף. כאשר הלד מגיע לשיא עוצמת ההארה הלד נכבה בבת אחת. שנה את התוכנית שגם הכיבוי יהיה לאט וברצף.

4. ברשותך נורת LED מסוג RGB כמתואר באיור 4



איור 4

- הנורית מורכבת משלושה לדים עם צבעי היסוד של: אדום, ירוק וכחול. כמתואר באיור 5



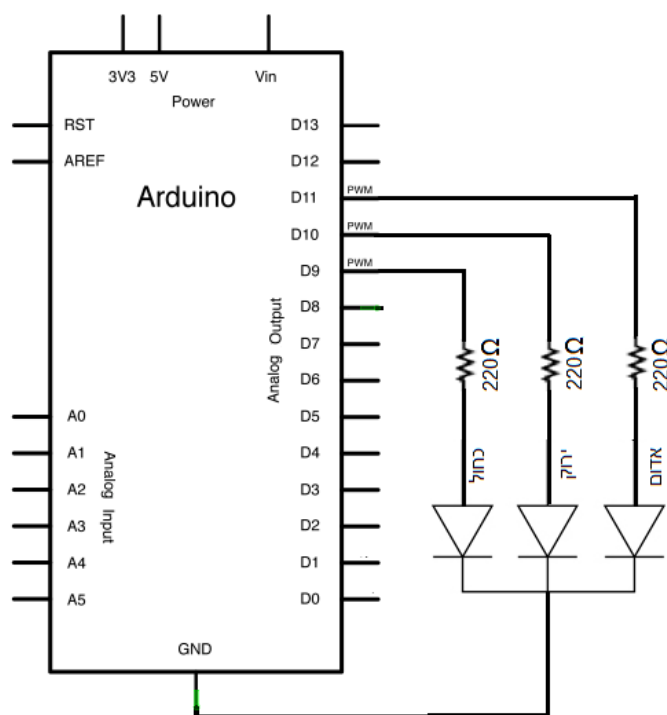
איור 5

באמצעות חיבור (ערבוב) של צבעי היסוד בעוצמות שונות נוכל לקבל צבעים חדשים.

הסבר על צבעי היסוד ראה לדוגמא ב:

http://he.wikipedia.org/wiki/%D7%A6%D7%91%D7%A2_%D7%99%D7%A1%D7%95%D7%93

5. חבר לכרטיס הארדואינו את נורת ה RGB LED כמתואר באיור 6



איור 6

6. כתוב את התוכנית הבאה כדי לבדוק שאכן הLED האדום מחובר לpin 11, הLED הירוק לpin 10 והLED הכחול לpin 9.

התוכנית:

```
#define red 9      // בכל מקום בתוכנית שכתוב red יוחלף בספרה 9
#define green 10   // בכל מקום בתוכנית שכתוב green יוחלף במספר 10
#define blue 11    // בכל מקום בתוכנית שכתוב blue יוחלף במספר 11

void setup(){      // במצב ההתחלתי שלושת הLEDים צריכים להיות מכובים.
    for(int i=9;i<12;i++) // 11, 10, 9 יהיה i של i
        analogWrite(i,0); // 0. כלומר גורם מחזור 0.
    // לאחר ביצוע הלולאה שלושת הLEDים מכובים.
}

void loop(){
    analogWrite(red,255); // הדלק את הLED האדום בעוצמת הארה מלאה.
    delay(1000);          // המתן שנייה אחת
    analogWrite(red,0);    // כבה את הLED האדום
    analogWrite(green,255); // הדלק את הLED הירוק בעוצמת הארה מלאה.
    delay(1000);          // המתן שנייה אחת
    analogWrite(green,0);  // כבה את הLED הירוק
    analogWrite(blue,255); // הדלק את הLED הכחול בעוצמת הארה מלאה.
    delay(1000);          // המתן שנייה אחת
    analogWrite(blue,0);   // כבה את הLED הירוק
}
```

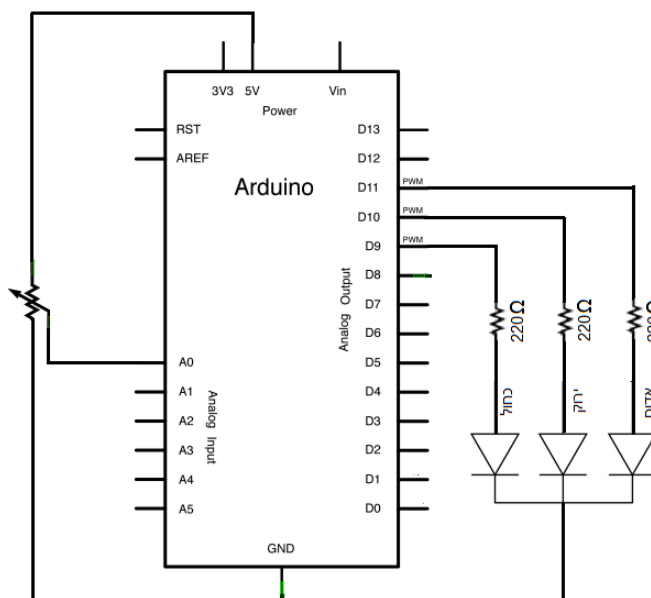
7. שנה את התוכנית מסעיף 6 כך, שכל צבע יידלק בעוצמה המשתנה ברצף. (בדומה לתרגיל 2 אבל לכל אחד מהLEDים).

8. התוכנית שלהלן 'מערבבת' את הצבעים במינונים שונים. החלף רק את הפונקציה `loop()` מתרגיל 6 בפונקציה `loop()` שלהלן.

```
void loop() {
    analogWrite(red,60); // הצבע האדום ערכו קבוע 60
    for (int i=0;i<128;i++){
        analogWrite(blue,i); // הצבע הכחול ערכו גדל ברציפות מ 0 עד 128
        delay(10);
    }
    for (int i=128; i>1; i--){
        analogWrite(blue,i); // הצבע הכחול ערכו יורד ברציפות מ 128 עד 1
        delay(10);
    }
    analogWrite(red,60); // הצבע האדום ערכו קבוע 60
    for (int i=0;i<128;i++){
        analogWrite(green,i); // הצבע הירוק ערכו גדל ברציפות מ 0 עד 128
        delay(10);
    }
    for (int i=128; i>1; i--){
        analogWrite(green,i); // הצבע הירוק ערכו יורד ברציפות מ 128 עד 1
        delay(10);
    }
}
```

9. הוסף לתוכנית מסעיף 8 'ערבוב צבעים' נוספים. למשל: צבע כחול קבוע על 60 והצבע האדום והירוק משתנים ברציפות.

10. הוסף למבוא האנאלוגי A0 פוטנציומטר כמשורטט באיור 7.



איור 7

התוכנית שלהלן מדליקה את הצבע האדום בעוצמה שתלויה בערך האנלוגי שמתקבל מהמבוא A0 , תחום ההמרה של הממיר מאנלוגי לדיגיטלי הוא 0-1023 ואילו התחום של PWM הוא 0-255 . על מנת להתאים את הערכים. אנחנו מחלקים את הערך האנלוגי שמתקבל מהממיר ב 4 .

```
# define red 9      // בכל מקום בתוכנית שכתוב red יוחלף בספרה 9

# define green 10   // בכל מקום בתוכנית שכתוב green יוחלף במספר 10

# define blue 11    // בכל מקום בתוכנית שכתוב blue יוחלף במספר 11

# define Potentiometer 0 // בכל מקום שכתוב Potentiometer יוחלף בספרה 0

void setup(){

  for(int i=9;i<12;i++)

    analogWrite(i,0); // כבה את שלושת הledים

  Serial.begin(9600); // קצב תקשורת טורית 9600 סיביות בשניה

}

void loop(){

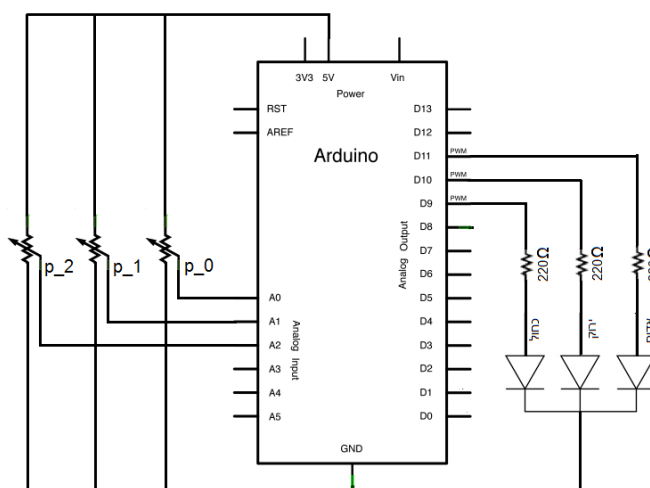
  int val=analogRead(Potentiometer)/4; // מחלקים ב 4 כדי להתאים לתחום של ה PWM

  Serial.println(val); // הצג את הערך של המשתנה val על גבי המוניטור

  analogWrite( red, val ); // עוצמת ההארה של הled האדום לפי ערכו של המשתנה val

}
```

11. שנה את התוכנית מסעיף 10 כך שעוצמת ההארה של ה**לד הירוק** יהיה תלוי בפוטנציומטר.
12. שנה את התוכנית מסעיף 11 כך שעוצמת ההארה של ה**לד הכחול** יהיה תלוי בפוטנציומטר.
13. חבר למבואות האנלוגיים A0,A1,A2 שלושה פוטנציומטרים כמשורטט באיור 8:



איור 8

14. התוכנית שלהלן מאפשרת 'למשתמש' לערבב צבעים באמצעות הפוטנציומטרים. כתוב ובדוק.

```
# define red 9      // בכל מקום בתוכנית שכתוב red יוחלף בספרה 9

# define green 10   // בכל מקום בתוכנית שכתוב green יוחלף במספר 10

# define blue 11    // בכל מקום בתוכנית שכתוב blue יוחלף במספר 11

# define p_0 0      // בכל מקום בתוכנית שכתוב p_0 יוחלף בספרה 0

# define p_1 1      // בכל מקום בתוכנית שכתוב p_1 יוחלף בספרה 1

# define p_2 2      // בכל מקום בתוכנית שכתוב p_2 יוחלף בספרה 2


void setup(){

    for(int i=9;i<12;i++)

        analogWrite(i,0); // כבה את שלושת הledים

    Serial.begin(9600); // קצב תקשורת 9600 סיביות לשניה

}

void loop(){

    int val_0=analogRead(p_0)/4; //val_0 משתנה יכיל את הערך הדיגיטאלי (אחרי המרה) של p_0

    Serial.print (" val_0="); //val_0= הדפס במוניטור את הטקסט:

    Serial.print (val_0); //val_0 הדפס במוניטור את תוכן (הנתון) שנמצא במשתנה

    analogWrite( red, val_0 ); //val_0 קבע עוצמת הארה ללד האדום לפי הערך שנמצא במשתנה

    int val_1=analogRead(p_1)/4; //val_1 משתנה יכיל את הערך הדיגיטאלי (אחרי המרה) של p_1

    Serial.print (" val_1="); //val_1= הדפס במוניטור את הטקסט:

    Serial.print (val_1); //val_1 הדפס במוניטור את תוכן (הנתון) שנמצא במשתנה

    analogWrite( green, val_1 ); //val_1 קבע עוצמת הארה של לד ירוק לפי הערך שנמצא במשתנה
```

```

int val_2=analogRead(p_2)/4; //p_2 של (אחרי המרה)
Serial.print (" val_2="); // val_2= הדפס במוניטור את הטקסט:
Serial.println (val_2); // ושוורה חדשה val_2 שנמצא במשתנה
analogWrite( blue, val_2 ); // val_2 במשתנה
}

```

פעילות 8 – בקרה על מנוע DC שמחובר לדוחף L293D

מבוא – עיקרון הפעולה של מנוע DC

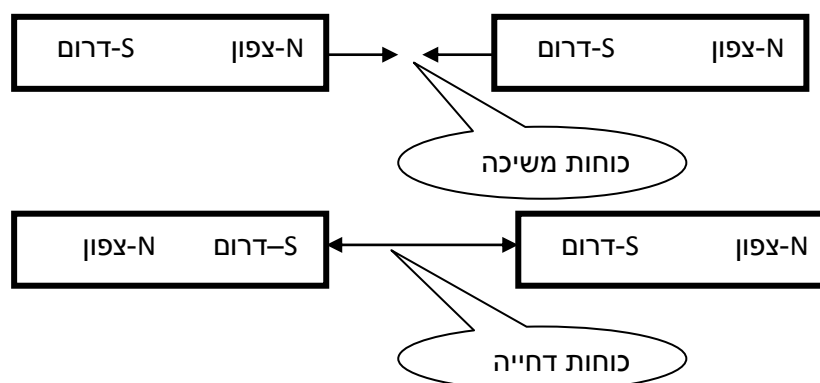
מנוע חשמלי הוא התקן אלקטרומגנטי המיועד להמיר אנרגיה חשמלית לאנרגיה מכאנית. ישנם סוגים שונים של מנועים חשמליים, בפעילות זו נתייחס למנועי DC (זרם ישר) רגילים (מנועי 'מברשות' Brush Motors), בהם משתמשים בדרך כלל במערכות רובוטיקה.

נתבונן תחילה בתופעות מגנטיות מוכרות.

א. לכל מגנט ישנם 2 קטבים המסומנים ב: N-צפון, ו S-דרום.

ב. התכונה הבסיסית של מגנט היא שבין קטבים מגנטיים שווים ישנו כוח דחייה ובין קטבים מגנטיים שונים

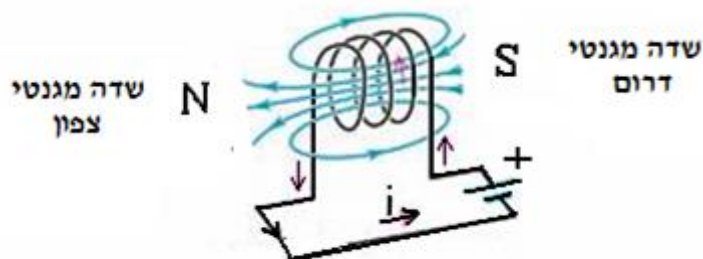
ישנו כוח משיכה. כמוראה באיור 1



איור 1

ג. כאשר מזרימים זרם חשמלי בסליל, נוצר שדה מגנטי. כיוון הזרם קובע את הקוטב המגנטי. על ידי שינוי

כיוון הזרם ניתן לשנות את הקטבים של השדה המגנטי. כמוראה באיור 2



איור 2

מנוע DC מכיל 2 חלקים עיקריים:

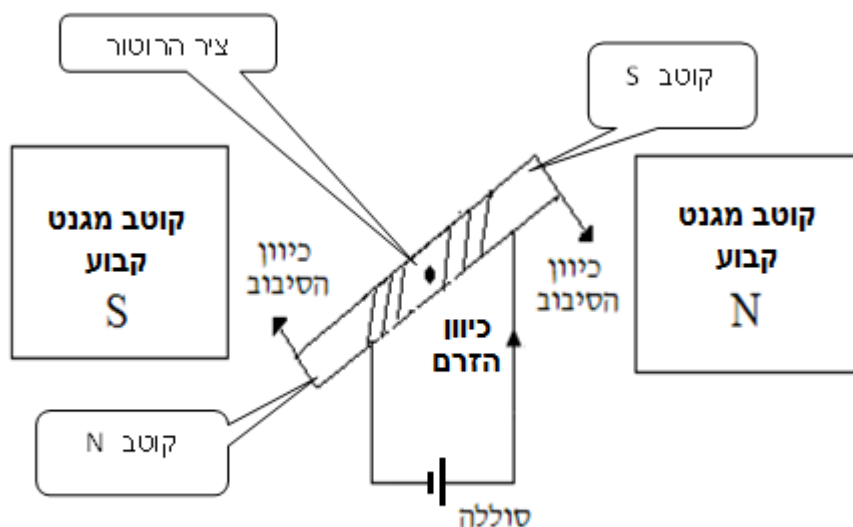
א. **סטטור המנוע** - הסטטור הוא חלק 'סטטי' (קבוע) שלא זז, ועליו נמצאים קטבים מגנטיים **קבועים**.

ב. **רוטור המנוע** - הרוטור הוא החלק שמסתובב. על הרוטור נמצא הסליל עם קטבים מגנטיים

משתנים

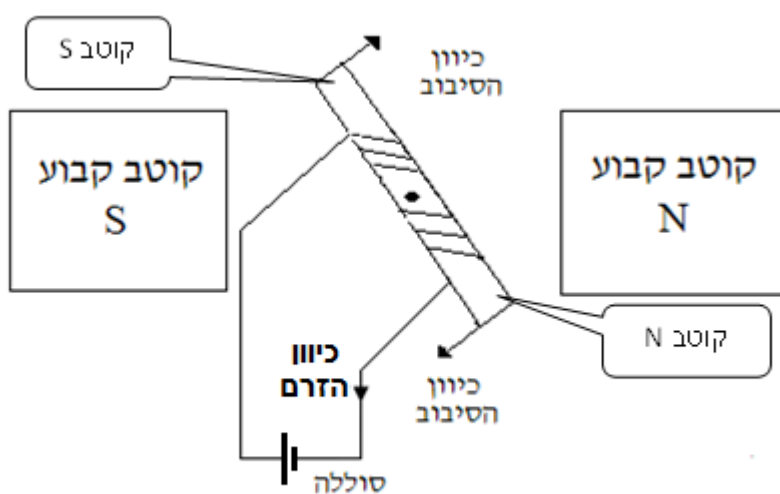
באמצעות איור 3, ואיור 4, נוכל להבין כיצד מתבצע סיבוב המנוע:

כאשר כיוון הזרם דרך הסליל שעל הרוטור הוא כמשורטט, נוצר כוח משיכה בין הקטבים והרוטור ינוע עם כיוון השעון.



איור 3

כאשר הרוטור יגיע למצב מאוזן הוא יעצור. אבל, אם נהפוך את קוטביות המתח של הסוללה, הזרם דרך הסליל שעל הרוטור יהפוך כיוון, קוטביות האלקטרומגנט שעל הרוטור תתחלף (כמוראה בשרטוט 4) ונקבל דחייה בין הקטבים



איור 4

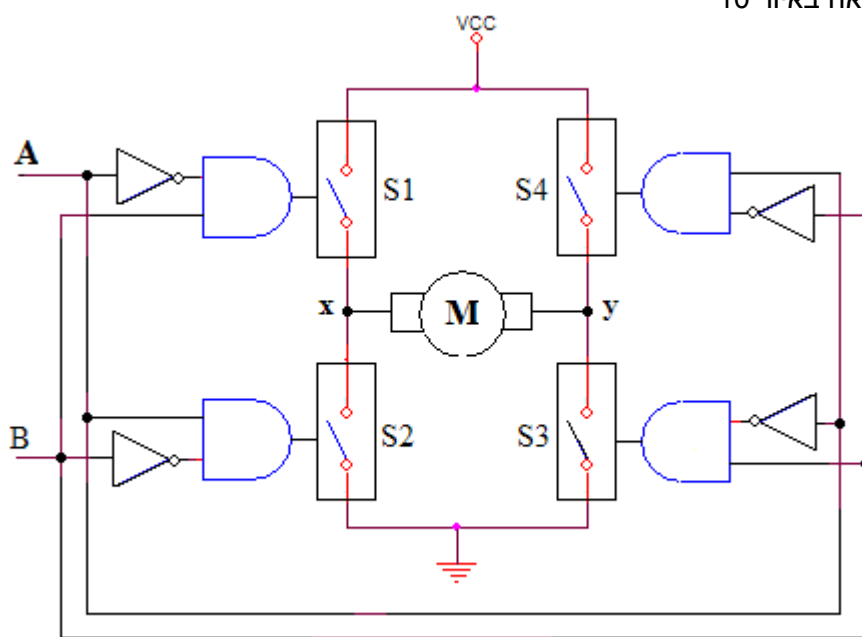
אנו רואים שהפיכת כיוון הזרם בסליל הרוטור תגרום להפיכת הקטבים המגנטיים ברוטור ולהמשך סיבוב המנוע. אבל לסוללה יש הדקי מתח עם קוטביות קבועה, אז איך הופכים את כיוון הזרם?

לשם כך בונים ממתג. הממתג הופך בכל סיבוב של 180 מעלות את קוטביות המתח שעל פני הסליל. באופן מעשי ההחלפה מתבצעת על ידי ממתג הבנוי ממגעים חשמליים הדומים למברשות.

'דוחף' - driver המנוע

- הבקרה על פעולת המנוע מתבצעת באמצעות רכיב שהוא 'דוחף המנוע'. באמצעות 'דוחף המנוע'. ניתן:
- לעצור ולהפעיל את המנוע ולקבוע את כיוון הסיבוב של המנוע.
 - לספק את הזרם הדרוש להפעלת המנוע.

בקר טיפוס של מנוע לזרם ישר בנוי ממבנה בסיסי של 4 מפסקים המחוברים למנוע בצורה של האות H כמראה באיור 10



איור 10

* הערה: המפסקים מייצגים טרנזיסטורים או ממסרים וכדו'

באמצעות טבלה 1 שלהלן ניתן להבין את פעולת 'בקר המנוע' עם מבנה הגשר H. לבקר המנוע יש שתי כניסות A, B. כלומר למבואות הבקר יש 4 מצבים אפשריים.

A	B	S1	S2	S3	S4	מצב המנוע
0	0	פתוח	פתוח	פתוח	פתוח	אין זרם. מנוע לא מסתובב
0	1	סגור	פתוח	סגור	פתוח	יש זרם מנקודה X לנקודה Y. מנוע מסתובב
1	0	פתוח	סגור	פתוח	סגור	יש זרם מנקודה Y ל X. מנוע מסתובב בכיוון הפוך.
1	1	פתוח	פתוח	פתוח	פתוח	אין זרם. מנוע לא מסתובב

טבלה 1

כפי שרואים מהטבלה, כאשר $A=B$ המנוע עוצר ולא מסתובב. במצב $AB=01$ המנוע מסתובב בכיוון אחד. ואילו במצב $AB=10$ המנוע מסתובב בכיוון ההפוך.

ניתן לרכוש כרטיסים (shield) עם דוחפי זרם מסוגים שונים. (הקישו לדוגמא ב ebay.com את צמד המילים . arduino dc motor driver ותקבלו למעלה מ 250 תוצאות).

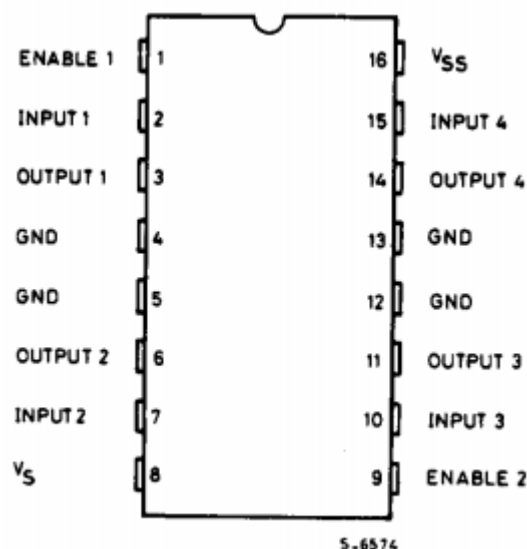
הרכיב L293D

בפעילות שלנו נשתמש ברכיב **L293D**. לרכיב זה ניתן לחבר 2 מנועים. הרכיב מסוגל לדחוף לכל מנוע זרם של עד 600mA.

תפקידי ההדקים ואופן פעולת הרכיב מתוארים בטבלאות שלהלן:

תפקידי ההדקים	
ENABLE	'אפשר' בקר המנוע
INPUT 1, INPUT 2	מבואות הבקרה על פעולת המנוע
Vs	מתח למנוע (עד 36v)
Vss	מתח 'ללוגיקה' (5v)
GND	חיבור ל (-)
OUTPUT 1, OUTPUT 2	חיבור למנוע

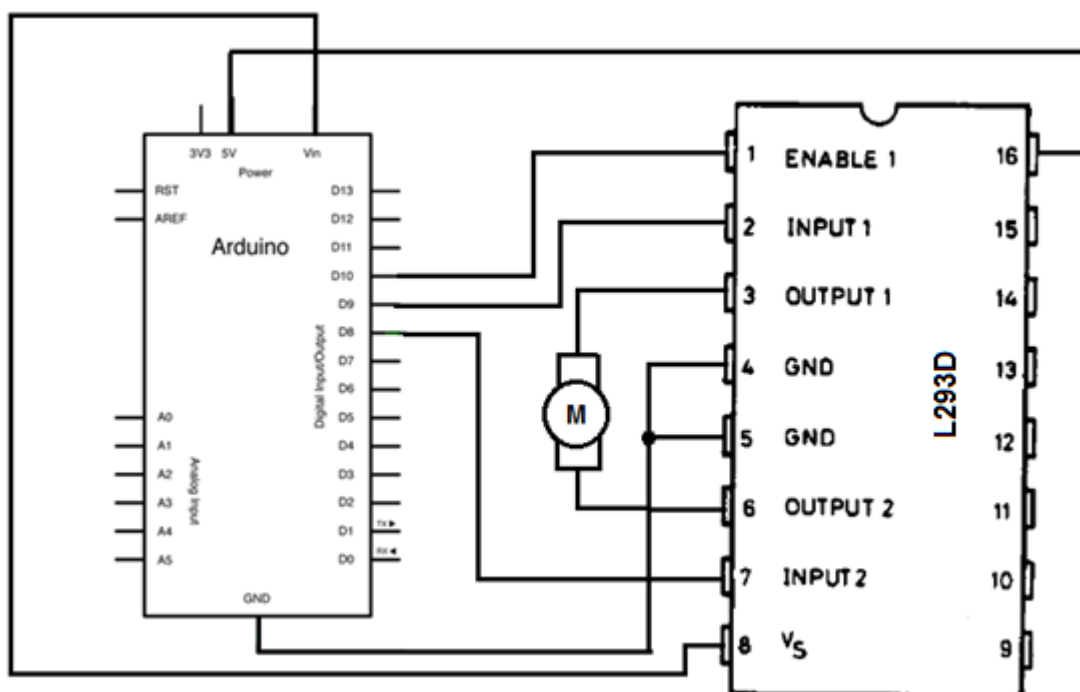
המבואות			אופן הפעולה
ENABLE	INPUT 1	INPUT 2	
0	ϕ	ϕ	מנוע עוצר ('ריצה חופשית')
1	0	0	מנוע עוצר (מידית)
1	0	1	מנוע מסתובב קדימה
1	1	0	מנוע מסתובב אחורה
1	1	1	מנוע עוצר (מידית)



כדי לקבוע את כיוון הסיבוב של המנוע. נחבר את המבואות INPUT של הרכיב אל ההדקים הדיגיטליים D8 | D9 של הארדואינו. (הדקים אלו יקבעו את כיוון הסיבוב)

כדי לשלוט על המהירות של המנוע נחבר את המבוא ENABLE אל ההדק D10 של הארדואינו. בהדק זה נייצר אות PWM.

1. חבר את המנוע אל מוצאי הרכיב L293D. ואת מבואות הרכיב L293D אל כרטיס הארדואינו כמשורטט:



2. התוכנית שלהלן גורמת למנוע להסתובב ב 2 מהירויות.

אחרי כתיבת התוכנית ואחרי ה upload של התוכנית אל כרטיס הארדואינו.
נתק את כבל ה USB מכרטיס הארדואינו וחבר את ספק הכח.

```
void setup () {
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
  digitalWrite(8,1);
  digitalWrite(9,0);
}

void loop(){
  המנוע מסתובב לאט. המנוע מקבל 90/255 מהאנרגיה
  delay(2000);          המתן 2 שניות
  המנוע מסתובב מהר. המנוע מקבל 255/255 מהאנרגיה
  delay(2000);          המתן 2 שניות
}
```

3. הטיפול במנוע היא 'פעולה' שחוזרת עצמה. מן הראוי שנכתוב עבורה 'פונקציה'. הפונקציה תקבל כפרמטר את **מהירות הנסיעה**. אם הערך הוא חיובי המנוע יסתובב בכיוון אחד. אם הערך הוא שלילי המנוע יסתובב בכיוון השני.

```
void speedMotor(int velocity) { // הפונקציה מקבלת את מהירות
  if (velocity>0) { // אם הערך הוא חיובי
    digitalWrite(8,1);
    digitalWrite(9,0);
    analogWrite(10, velocity); // velocity במהירות
  }
  else if (velocity<0) { // אחרת, אם הערך הוא שלילי
    velocity=-velocity; // הפונקציה צריכה לקבל ערך חיובי עבור המהירות
    digitalWrite(9,1); // אבל הופכים את כיוון הסיבוב
    digitalWrite(8,0);
    analogWrite(10,velocity); // velocity במהירות
  }
  else { // אחרת, (אם הערך הוא 0)
    digitalWrite(9,0); // עצור את המנוע
    digitalWrite(8,0);
  }
}

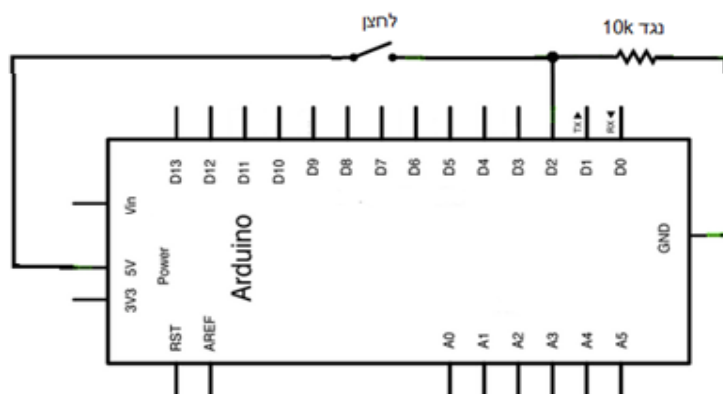
void setup () {
```

```

pinMode(8, OUTPUT);
pinMode(9, OUTPUT);
}
void loop(){
  speedMotor(70 );      // הסתובב במהירות 70/255 בכיוון אחד
  delay(1000);           // למשך שניה אחת
  speedMotor(0 );       // עצור
  delay(500);            // למשך חצי שניה
  speedMotor(-70 );      // הסתובב במהירות 70/255 בכיוון השני
  delay(1000);           // למשך שניה אחת
  speedMotor(0 );       // עצור
  delay(500);            // למשך חצי שניה
  speedMotor(250 );      // הסתובב במהירות 250/255 בכיוון אחד
  delay(1000);           // למשך שניה אחת
  speedMotor(0 );       // עצור
  delay(500);            // למשך חצי שניה
  speedMotor(-250 );     // הסתובב במהירות 250/255 בכיוון השני
  delay(1000);           // למשך שניה אחת
  speedMotor(0 );       // עצור
  delay(500);            // למשך חצי שניה
}

```

4. **הוסף** ליציאה הדיגיטלית D2 לחצן כמתואר בשרטוט.



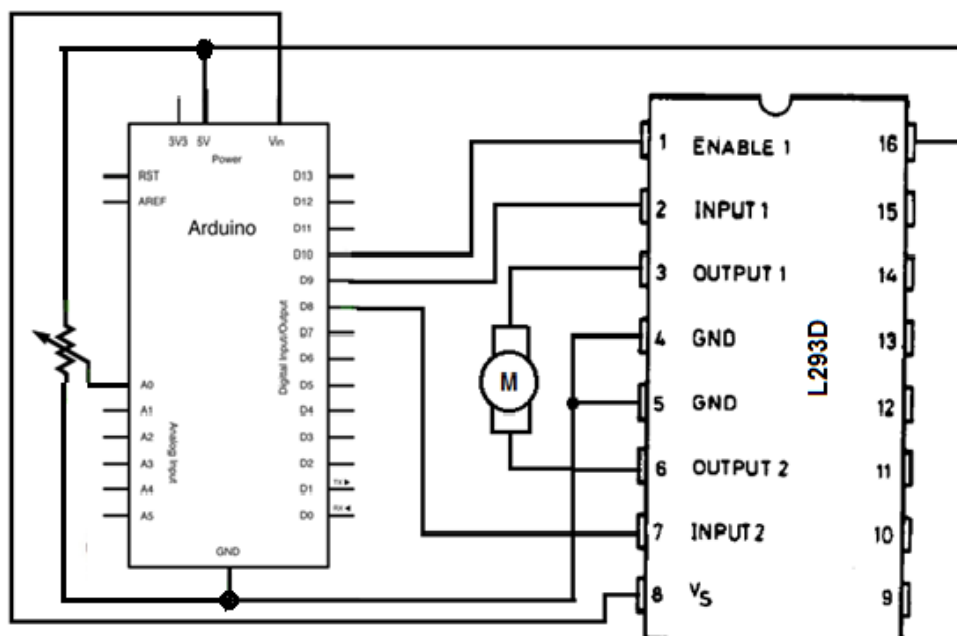
קבע מצב התחלתי מנוע מסתובב במהירות קבועה של 150. אולם כאשר הלחצן במצב 'לחוץ' המנוע הופך כיוון ומסתובב במהירות 150 לכיוון השני. (במצב 'לא לחוץ' המנוע חוזר למצבו ההתחלתי)

5. שנה את התוכנית כך, שעבור זיהוי של לחיצה (ועזיבה) המנוע הופך כיוון (עד ללחיצה הבאה) כלומר, בכל לחיצה (ועזיבה) המנוע הופך את כיוון הסיבוב.

6. כתוב תוכנית שתבצע:

- א. מהירות התחלתית 75.
- ב. כל עוד המהירות קטנה מ 230, עבור כל לחיצה ולחיצה, המהירות תגדל ב 25.
- ג. כאשר המהירות גדולה מ 75, כל עוד המהירות גדולה מ 75, עבור כל לחיצה, המהירות תקטן ב 25.
- ד. חזור לסעיף ב.

7. חבר למבוא האנלוגי A0 פוטנציומטר כמשורטט. התוכנית שלהלן גורמת לכך שמהירות המנוע תלויה במצבו של הפוטנציומטר.
 הערה: הערך הדיגיטלי המתקבל מהממיר (analogRead()) הוא בתחום של 0-1023.
 הערך ל PWM (analogWrite()) הוא בתחום של 0-255. לכן יש לחלק את הערך המתקבל מהממיר ב 4.



```

    analogWrite(10, velocity); // velocity במהירות
}
void setup () {
    pinMode(8, OUTPUT);
    pinMode(9, OUTPUT);
}
void loop()
{
    speedMotor(analogRead(A0)/4); // קבע את המהירות ע"פ הערך המתקבל מהממיר
}

```

8. כתוב תוכנית הגורמת למנוע להסתובב בתלות המתח האנלוגי Vin שבמבוא A0 לפי התנאים שבטבלה שלהלן:

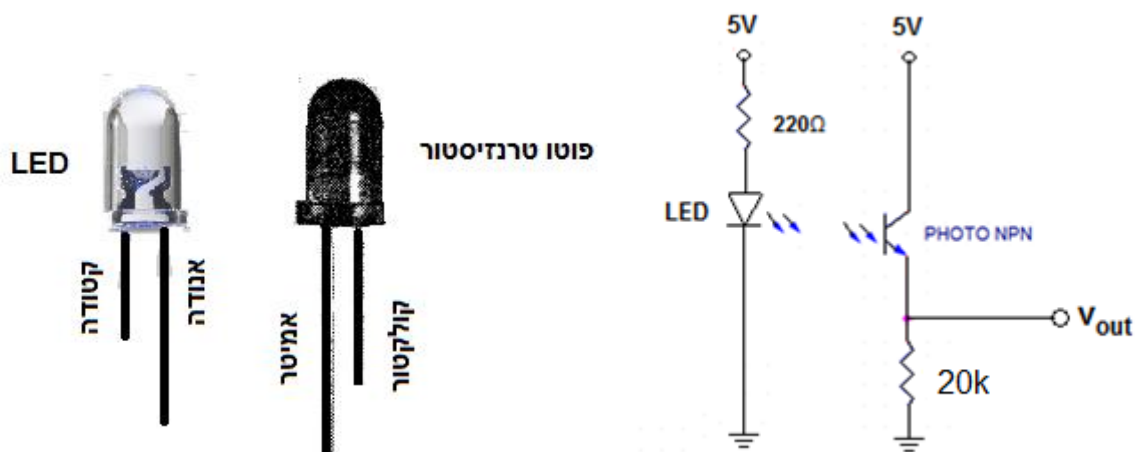
מצב המנוע	כאשר
מנוע עוצר	$0.5v > V_{in}$
מנוע במהירות 90 בכיוון אחד	$1.5v > V_{in} > 0.5v$
מנוע במהירות 200 בכיוון אחד	$2.5v > V_{in} \geq 1.5v$
מנוע במהירות 200 בכיוון השני	$3.5v > V_{in} \geq 2.5v$
מנוע במהירות 90 בכיוון השני	$4.5v > V_{in} \geq 3.5v$
מנוע עוצר	$V_{in} \geq 4.5v$

- א. חשב אתה הערך הדיגיטלי של Vin עבור כל אחד מהתחומים.
 ב. העזר בפונקציה speedMotor(int velocity) מתרגיל 3.
 ג. כאשר $V_{in} = 2.5v$ המנוע הופך כיוון. מבחינה מכאנית מומלץ לעצור את המנוע לזמן קצר (למשל ל 0.1 שניה) ורק לאחר מכן להפוך את כיוון המנוע.

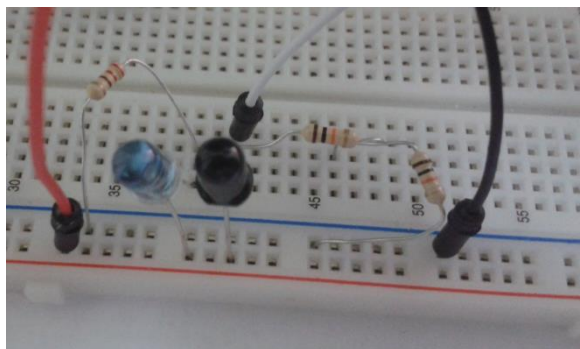
פעילות 9 – חיישן IR (Infra Red)

חיישן ה IR הוא פשוט, זול, ויש לו שמושים רבים ומגוונים.....

החיישן בנוי מ LED שמקרין אור IR ומפוטוטרנזיסטור שרגיש לאור IR. הזרם באמיטר של הפוטוטרנזיסטור תלוי בעוצמת האור שבסיס הטרנזיסטור. ככל שעוצמת האור גדולה יותר כך הזרם באמיטר גדול יותר. מאחר ולאמיטר מחובר נגד (10k) לכן ככל שעוצמת האור גדלה, המתח v_{out} יגדל.....



1. בנה את המעגל על 'לוח בניה' חדש (לא על גבי המטריצה שבנית את ה L293). בנה את המעגל כך שה LED והפוטוטרנזיסטור יהיו קרובים (צמודים) זה לזה כמוראה בתמונה.

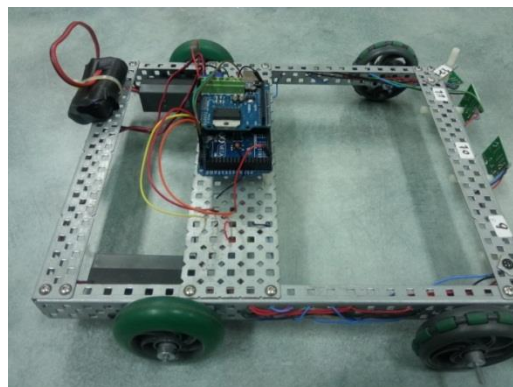
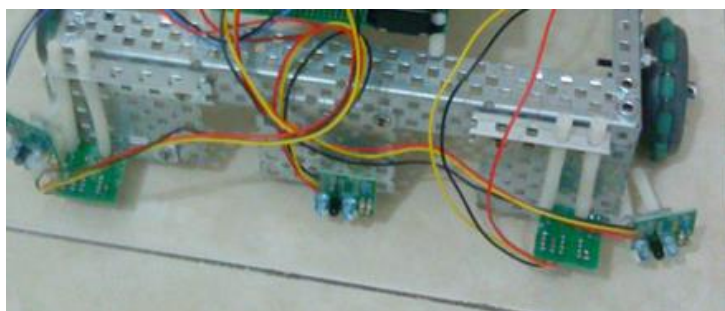


2. חבר את V_{out} אל המבוא האנלוגי A1 של הארדואינו.
3. התוכנית שלהלן מציגה ע"ג המוניטור את הערך הדיגיטלי של V_{out} .

```
void setup(){
  Serial.begin(9600);    // פתח ערוץ תקשורת טורית בקצב של 9600 סיביות לשניה
}

void loop(){
  int IR;                // הגדר משתנה בשם IR
  IR=analogRead(A1);     // קלוט אל המשתנה, את הערך האנלוגי מערוץ A1
  Serial.println(IR);    // הצג על המוניטור את הערך של המשתנה
  delay(100);           // המתן עשירית השניה
}
```

4. בעזרת כף ירך ו/או בעזרת דף נייר, שנה את עוצמת ההחזרה של האור אל הפוטורנזיסטור. בדוק את הערכים בתצוגת המוניטור.
5. הפוך את המטריצה כלפי השולחן, כך שהחיישן יפנה כלפי השולחן (הLED והפוטו טרנזיסטור, שניהם כלפי השולחן). בדוק (במוניטור) את הערכים המתקבלים בשני מצבים: כשהחיישן מעל השולחן וכשהחיישן זז הצידה מהשולחן. (כשאינן החזר אור מהשולחן). כתוב תוכנית שמבצעת:
 - א. כל עוד החיישן מעל השולחן המנוע מסתובב במהירות המקסימלית בכיוון אחד.
 - ב. אם החיישן לא מעל השולחן:
 - עצור למשך שניה.
 - סובב את המנוע לכיוון ההפוך, למשך 2 שניות, בחצי מהמהירות המקסימלית,
 - עצור למשך שניה.
 - חזור לסעיף א.
7. בעזרת 5 חיישני IR שיחוברו ל 5 הערוצים האנלוגיים, ניתן לבנות רובוט פשוט אבל 'מדליק'...., רובוט 'עצמאי' ו'אינטליגנטי' שלא נופל מהשולחן.....ולא מתנגש בחפצים..... שני חיישני IR (מתוך החמישה) יחוברו לפינות הרובוט ויהיו מכוונים כלפי השולחן. כך הרובוט יוכל לזהות שהוא הגיע לקצה השולחן. שלושה חיישנים נוספים יחוברו כך שהם "מסתכלים" כלפי המרחב, כך הרובוט יוכל לזהות אם יש חפץ ממולו. בתמונה רואים לדוגמא רובוט שנבנה מחלקי מתכת של VEX, מנועים של VEX ו חמישה חיישני IR. אבל כמובן שגוף הרובוט יכול להיבנות מעץ וכו' מנועים וגלגלים מכל סוג שיש ברשותך.... מה שנשאר זה לתכנת.....

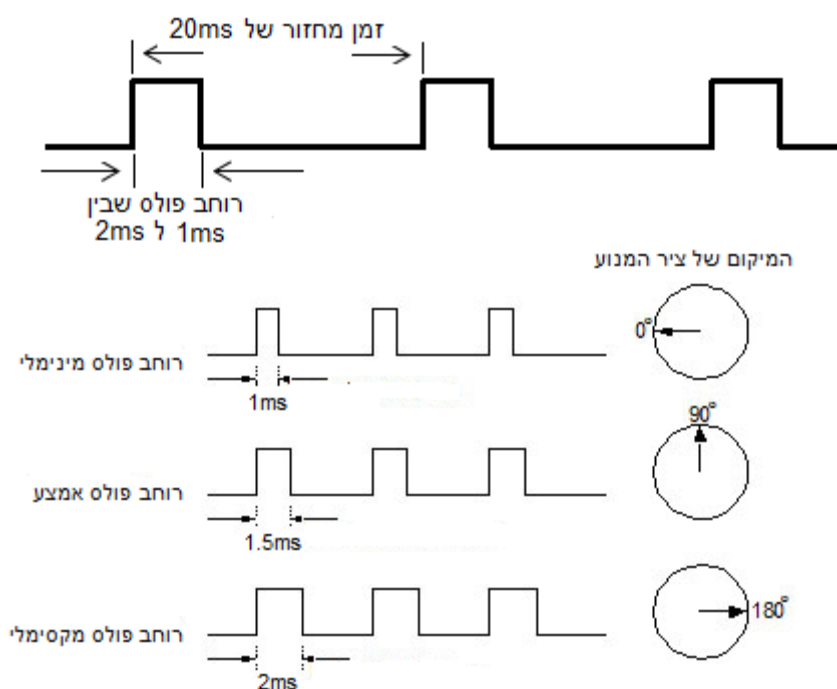


בהצלחה!!!

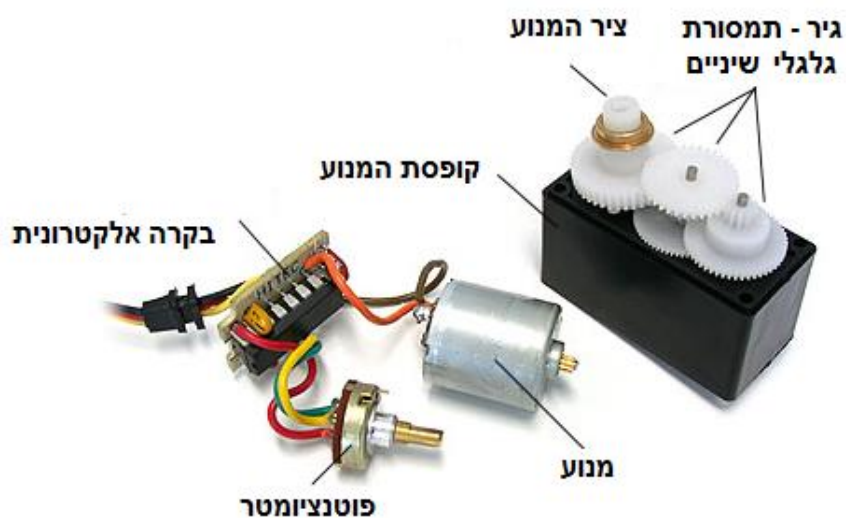
פעילות 10 - מנוע סרוו Servo motor

מנוע סרוו הוא מנוע עם 'בקרה'. למנוע סרוו יש 3 חוטים. שני חוטים לאספקת מתח (+/-) וחוט אחד נוסף לבקרה. בניגוד למנוע DC שמסתובב 'חופשי', למנוע סרוו יש מערכת בקרה אלקטרונית פנימית המניעה את ציר המנוע לזווית מדויקת בהתאם לרוחב פולס שמתקבל בקו הבקרה.

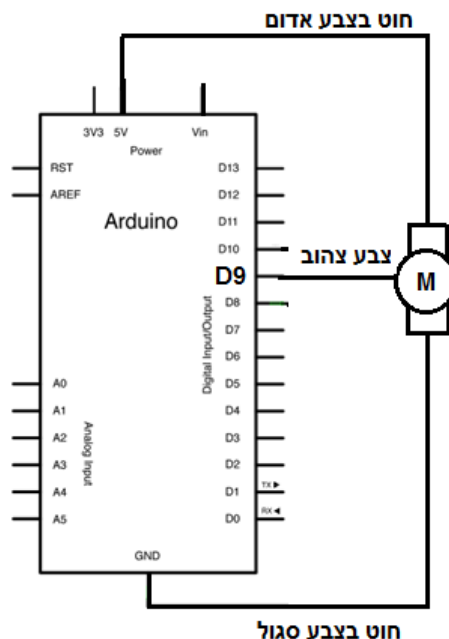
בד"כ מקובל שרוחב הפולס נע בין רוחב פולס מינימלי של 1ms (שמייצג זווית של 0 מעלות) לרוחב פולס מקסימלי של 2ms (שמייצג 180 מעלות). כמראה בשרטוט:



המבנה של מנוע סרוו.



1. חבר את מנוע הסרוו לכרטיס הארדואינו כמשורטט (קו הבקרה למוצא דיגיטלי 9):



הערה: במנוע המסויים שלנו המינוס בצבע סגול. בדרך כלל המינוס בצבע שחור.

התוכנית שלהלן מביאה את מנוע הסרוו לנקודת האמצע (90 מעלות)

```
void setup() {
  pinMode(9,OUTPUT);

  מנוע הסרוו יקבל פעמיים פולס ברוחב של 1.5mS
  digitalWrite(9,HIGH);    // העלה את קו הבקרה ל '1'
  delayMicroseconds(1500); // השהייה של 1.5mS
  digitalWrite(9,LOW);     // הורד את קו הבקרה ל '0'
  delay(20);
}

}

void loop() {
}
```

נסה להזיז את ציר המנוע ממקומו. (בעדינות...לא להתלהם....)הבקרה הפנימית מתנגדת...

2. שנה את הערך של ההשהייה ב `delayMicroseconds` ל 1000 ובדוק תגובת המנוע.

3. שנה את הערך של ההשהייה ב `delayMicroseconds` ל 2000 ובדוק תגובת המנוע.

לארדואינו ספריה לטיפול במנועי סרוו... התוכניות שלהלן משתמשות בפונקציות הספריה Servo.h.

```
#include <Servo.h>

Servo myservo; //מייקט בשם myservo

void setup()

{

  myservo.attach(9); // קבע את הדק 9 כהדק בקרה למנוע סרוו

  myservo.writeMicroseconds(1500); // ציר מנוע ינוע לנקודת אמצע 1.5ms. צור רוחב פולס של

}

void loop() {}
```

4. שנה את הערך ל 1000 ובדוק את תגובת המנוע.

5. שנה את הערך ל 2000 ובדוק את תגובת המנוע.

6. התוכנית שלהלן מזיזה את ציר המנוע בקפיצות (רוחב הפולס גדל ב 0.1ms).

```
#include <Servo.h>

Servo myservo; //מייקט בשם myservo

void setup()

{

  myservo.attach(9); // קבע את הדק 9 כהדק בקרה למנוע סרוו

}

void loop() {
  for (int i=1000; i<=2000; i=i+100)
  {
    myservo.writeMicroseconds(i);
    delay (500);
  }
}
```

7. לארדואינו יש פונקציה בשם write() שמקבלת כפרמטר זווית שבין 0 ל 180 מעלות.

הפעולה write() תמקם את ציר המנוע בזווית המתאימה.
לדוגמא:

```
#include <Servo.h>

Servo myservo;

void setup()

{
  myservo.attach(9);
  myservo.write(90); // (נקודת האמצע) קבע את ציר המנוע ל 90 מעלות.
}

void loop() {}
```

8. התוכנית שלהלן מציבה את ציר המנוע בזווית של 45 מעלות ולאחר מכן בזווית של 135 מעלות. וחוזר חלילה.....

```
#include <Servo.h>
Servo myservo; //myservo בשם
void setup()
{ myservo.attach(9); } // 9 בהדק
void loop() {
  myservo.write(45); // מנוע ל 45 מעלות
  delay(1000);
  myservo.write(135); // 135 מעלות
  delay(1000);
}
```

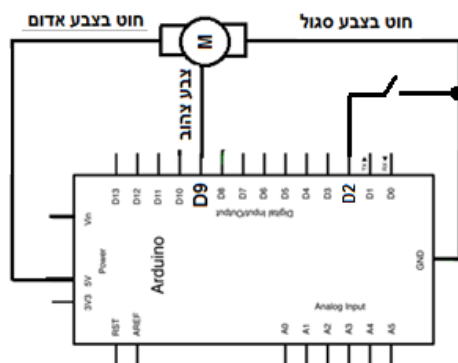
9. התוכנית שלהלן משנה את מיקום הציר בקפיצות של 10 מעלות.

```
#include <Servo.h>
Servo myservo; //myservo בשם
void setup()
{
  myservo.attach(9); // 9 בהדק
}
void loop() {
  for (int i=0; i<=180; i=i+10) // 10 בקפיצות של
  {
    myservo.write(i); // i זווית לפי
    delay (500); // המתן 0.5 שניה
  }
}
```

10. הוסף לתוכנית מתרגיל 7, את ההוראות הנדרשות, כך שציר המנוע יחזור לנקודת האפס גם כן בקפיצות של 10 מעלות.

11. שנה את ההוראה $i=i+10$ (שבולאת for) ל $i=i+1$. והקטן את ההשהייה ל $\text{delay}(20)$. ובדוק!

12. ישנם מנועים שהתחום של רוחב הפולס שלהם שונה מהתחום של 1ms-2ms. על מנת לחקור את התחום של 'רוחב הפולס' של מנוע הסרוו המסויים שברשותנו, נוסיף לארדואינו לחצן בהדק D2 כמשורטט:



נכתוב תוכנית שבודקת את 'רוחב הפולס' המקסימלי.

התוכנית:

א. תמקם את ציר המנוע בנקודת האמצע המקובלת (רוחב פולס של 1.5ms).

- ב. תציג את רוחב הפולס במוניטור.
ג. בכל לחיצה תוסיף $50\mu\text{s}$ לרוחב הפולס.

באמצעות המוניטור נבדוק באיזה 'רוחב פולס' ציר המנוע הפסיק לזוז.

```
#include <Servo.h>

Servo myservo;    //מייקט בשם myservo
int pos=1500;    // המשתנה pos מכיל את הערך של 'רוחב הפולס'

void setup()
{
  pinMode(2,INPUT_PULLUP); // 'מושך מעלה' נגד
  Serial.begin(9600); // קצב תקשורת 9600 סיביות לשניה
  Serial.println("start=1500"); // מיקום התחלתי בנקודת אמצע
  myservo.attach(9); // קבע הדק בקרה למנוע סרוו בהדק 9
  myservo.writeMicroseconds(pos);
}

void loop() {
  כל עוד הלחצן לא 'לחוץ' המתן ..... //
  delay(50); // השהייה עד שהריטוטים מפסיקים
  pos=pos+50; // הגדל את רוחב הפולס ב  $50\mu\text{s}$ 
  myservo.writeMicroseconds(pos); // שדר לסרוו את רוחב הפולס
  Serial.println(pos); // הדפס את רוחב הפולס במוניטור
  delay (500);
  המתן עד לשחרור הלחצן //
  while(digitalRead(2)==0);
  delay(50); // השהייה בגלל ה 'ריטוטים'
}
```

13. שנה את התוכנית מתרגיל 10 , כך שנוכל לבדוק מהו רוחב הפולס **המינימלי**.
(במקום $\text{pos}=\text{pos}+50$; החלף ל $\text{pos}=\text{pos}-50$).

בבדיקה מצאנו ש'רוחב הפולס' בנקודת 0 מעלות הוא $750\mu\text{s}$ ו 'רוחב הפולס' בנקודת ה 180 מעלות הוא $2250\mu\text{s}$.

נכתוב את התוכנית:

```

include <Servo.h>

Servo myservo;    //myservo בשם אובייקט
void setup()
{
    myservo.attach(9);    // קבע הדק בקרה למנוע סרוו בהדק 9
}

void loop() {
    myservo.writeMicroseconds(2250); // עבור לנקודת קצה של 180 מעלות
    delay(1000);

    myservo.writeMicroseconds(750); // עבור לנקודת קצה 0 מעלות
    delay(1000);
}

```

במנוע המסויים שלנו המפתח הוא רק 150 מעלות .

14. בפעילויות קודמות השתמשנו במוניטור כדי להציג מידע שהתקבל מהארדואינו. כלומר, הארדואינו כתב מידע אל המוניטור. אבל ניתן גם לקרוא מידע מהמוניטור. בתוכנית שלהלן אנו מזיזים את ציר המנוע, באמצעות הוראות מהמוניטור. נשתמש בתוכנית בהוראה Serial.read(); אשר קוראת תווים מהמוניטור. לאחר כתיבת והעלאת התוכנית יש לפתוח את המוניטור ולהקיש בחלון הכתיבה את התווים.

```

#include <Servo.h>

Servo MyServo;

char order; // משתנה שיקבל את 'ההוראה' מהמוניטור...
int pos = 90; // מכיל את הזווית

void setup() {
    MyServo.attach(9);

    MyServo.write(pos); // המיקום ההתחלתי בנקודת אמצע
    Serial.begin(9600);

    Serial.println(" choose R   for Right  movment ");
    Serial.println(" choose L   for Left   movment");
}

```

```

void loop() {

  if (Serial.available() > 0) { // אם יש ב-BUFFER תווים לקריאה

    order = Serial.read(); // קרא את התו למשתנה order

    if (order == 'L') { // אם התו שנקרא הוא L

      pos = pos + 5; // הוסף לזווית הנוכחית 5 מעלות

      MyServo.write(pos);

    }

    else if (order == 'R') { // אם התו שנקרא הוא R

      pos = pos - 5; // חסר מהזווית הנוכחית 5 מעלות

      MyServo.write(pos);

    }

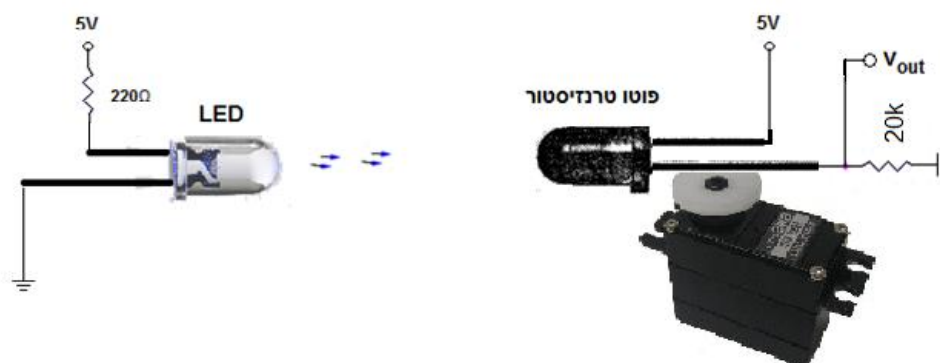
  }

  delay(20)

}

```

15. בפעילות 9 הכרנו LED שפולט אור IR ופוטוטרנזיסטור שרגיש לאור IR .
- הצמד את הפוטוטרנזיסטור (עם הנגד שמחובר בטור) על ציר המנוע (כמראה בתמונה)
 - חבר את V_{out} אל כניסה אנאלוגית
 - מקם בנקודה כלשהי במרחב את הLED שפולט אור



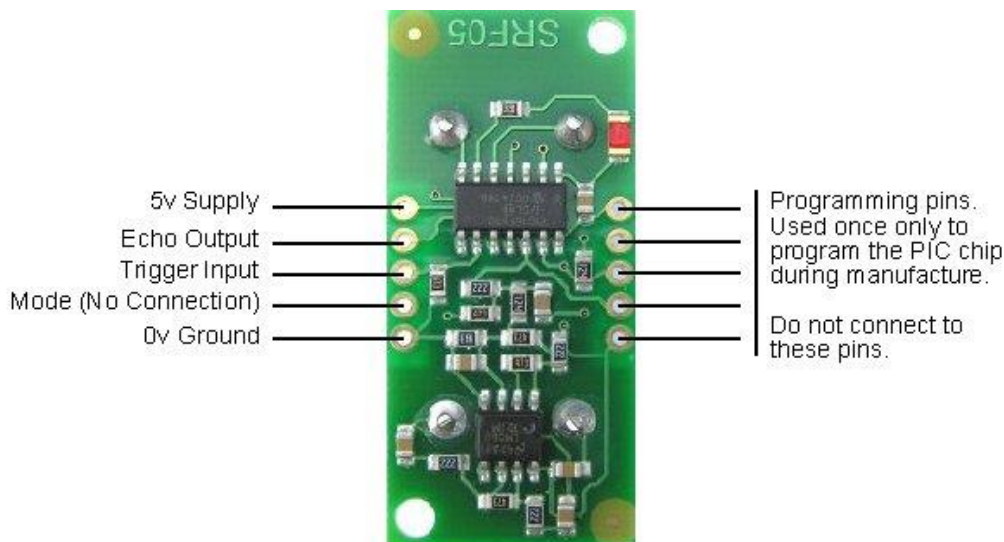
- כתוב תוכנית שגורמת לציר המנוע לסרוק את הסביבה (את כל המפתח של 150 מעלות). התוכנית תמצא את הנקודה (את הזווית) שממנה יש קרינה מקסימלית. ותמקם את ציר המנוע בדיוק מול הנקודה (למעשה מול ה-LED) .

הערה כללית: יש לשים לב: שכל עוד אחד מהמוצאים בכרטיס מוגדר כיציאת סרוו (עקב הפקודה: `myservo.attach(pin);` ההדקים 9 ו-10 לא יכולים לשמש כיציאות PWM . זאת עד לשחרור ההדקים כולם ע"י הפקודה: `servo.detach()`)

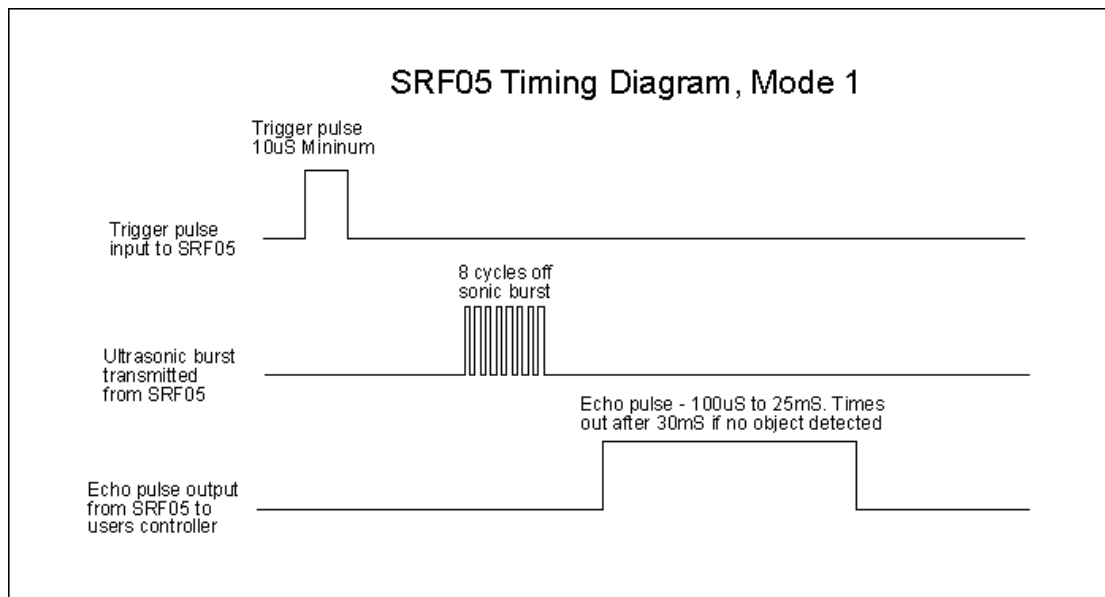
פעילות 11 – חיישן מרחק אולטרא סוני Ultra-Sonic SRF05

ידוע שהתפשטות גלי הקול במרחב היא במהירות של כ 340m/s . חיישן ה Ultra-Sonic משדר למרחב גלי קול בתדר (Ultra-Sonic) של 40kHz . החיישן נותן לנו את פרק הזמן שחלף מרגע השידור ועד לקבלת ההד בחזרה. ניתן לחשב את המרחק על פי הנוסחה: דרך=מהירות X זמן.

לחיישן 4 הדקים. שני הדקים לאספקת מתח (5v , GND) . הדק מבוא (Trigger) למתן 'דרבון' לחיישן. והדק מוצא (ECHO) לקבלת הזמן שחלף מרגע שידור גלי הקול ועד לקבלת ההד בחזרה.



Connections for 2-pin Trigger/Echo Mode (SRF04 compatible)



מדיאגרמת הזמנים אנו רואים שהתנהגות החיישן היא כדלקמן:

- א. נותנים בהדק Trigger – פולס שרוחבו $10\mu\text{s}$.
- ב. בעקבות הפולס, החיישן מבצע שני דברים:
 - i. משדר למרחב 8 מחזורים של גל קול בתדר 40kHz .
 - ii. מעלה את הדק Echo ל '1'.
- ג. כאשר ההד חוזר אל 'המקלט' הרכיב מוריד את הדק Echo ל '0'.

כדי למדוד את המרחק, אנו מודדים את 'הזמן' ומשתמשים בנוסחת המהירות: $V = \frac{S}{t}$

כאשר: V – מהירות. S – דרך (ביחידות meter). t – זמן (ביחידות second).

המיקרובקר מודד את הזמן במיליוניות השניה. כדי לקבל את המרחק בס"מ נחלק את הזמן שהחיישן מדד במספר 58.

הסבר:

ידוע שמהירות גלי הקול (ב 24°C) היא: 346 meter/second . כלומר זמן ההתפשטות (שיסומן ב X)

עבור ס"מ אחד הוא: $X = \frac{2.89 \text{ mili second}}{\text{meter}} = \frac{28.9 \text{ micro second}}{\text{cm}}$

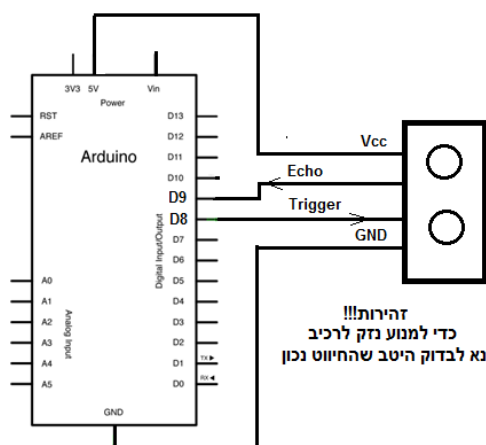
המרחק 'הנמדד' הוא: $S = \frac{t}{x} = \frac{t}{28.9}$ (S – המרחק ב cm - t – הזמן ב μs . X – זמן ההתפשטות של גלי הקול ב $\mu\text{s/cm}$).

גלי הקול עוברים דרך כפולה (מהחיישן למשטח ובחזרה), לכן יש לחלק את הזמן ב 2 .

כלומר: $S = \frac{t/2}{x} = \frac{t/2}{28.9}$

לכן, כדי לקבל את המרחק (S), יש לחלק את 'הזמן הנמדד' (t) במספר: $28.9 \cdot 2 \approx 58$.

1. חבר את החיישן כמתואר בשרטוט:



התוכנית שלהלן מציגה על גבי המוניטור את המרחק בס"מ. כתוב ובדוק.

```
int trig=8, echo=9;
```

```
void setup() {
```

```
  pinMode(trig,OUTPUT); // הדק 8 הוא מוצא כדי לתת 'דרבון' למבוא החיישן
```

```
  pinMode(echo,INPUT); // הדק 9 הוא מבוא כדי לקבל מידע על ההד
```

```
  Serial.begin(9600);
```

```
}
```

```
int time, distance; //
```



```

void loop() {

    digitalWrite(trig,HIGH);    // כדי לתת trigger אנו מעלים את הדק 8 ל '1'
    delayMicroseconds(10);    // למשך 10 מיקרו שניה
    digitalWrite(trig,LOW);    // מורידים את הדק 8 יורד ל'0'
    while(digitalRead(echo)==0); //.....המתן '1' עלה ל
    time=micros();    //micro second ב המונה של הערך הנוכחי של המונה
    while(digitalRead(echo)==1); // '0' ל Echo ה לירידת
    time=micros() - time;    // הצב במשתנה time את פרק הזמן שחלף
    distance=time/58;    // חשב את המרחק בס"מ
    Serial.print ("distance = ");
    Serial.println(distance);    // הדפס במוניטור את המרחק
    delay(100);
}

```

2. לארדואינו פונקציה בשם pulseIn() שיודעת למדוד 'רוחב פולס'

pulseIn(pin, value)
 מספר ההדק – pin
 קובע האם הפולס מתחיל בעליה 'לגבוה' או בירידה ל 'נמוך' (LOW או HIGH) – Value
 החלף את ארבע ההוראות ההוראות:

```

while(digitalRead(echo)==0); //.....המתן '1' עלה ל
time=micros();    //micro second ב המונה של הערך הנוכחי של המונה
while(digitalRead(echo)==1); // '0' ל Echo ה לירידת
time=micros() - time;    // הצב במשתנה time את פרק הזמן שחלף

```

בהוראה הבודדת :

```
time=pulseIn(9,HIGH); // HIGH ל העליה ל . מרגע העליה ל HIGH
```

בשתי התוכניות שכתבנו המיקרובקר משועבד לפעולה אחת בלבד. למדוד את המרחק. המיקרובקר ממתין (לא עושה כלום) עד לעלית הפולס , לאחר מכן ממתין (לא עושה כלום) עד לירידת הפולס למצב LOW . לאחר מכן מחשב את המרחק, נותן trigger וחוזר להמתין

לא חבל על הזמן? ("Time is money"). האם יש אפשרות שהמעבד יעשה 'משהו' נוסף בזמן הזה? במילים אחרות, האם יש אפשרות שהמעבד יבצע עוד משימה במקביל?

התשובה היא חיובית. לשם כך עלינו להכיר את מושג ה'פסיקה' (Interrupt).

פעילות 12 - פסיקות - interrupt (חיצונית)

הערה: להלן נתייחס לפסיקות חיצוניות. כלומר לפסיקות שמקורן מחוץ למיקרובקר.

ישנם שתי שיטות לבדוק האם קרה שינוי חיצוני באחד מהמבואות הדיגיטליים של הבקר.

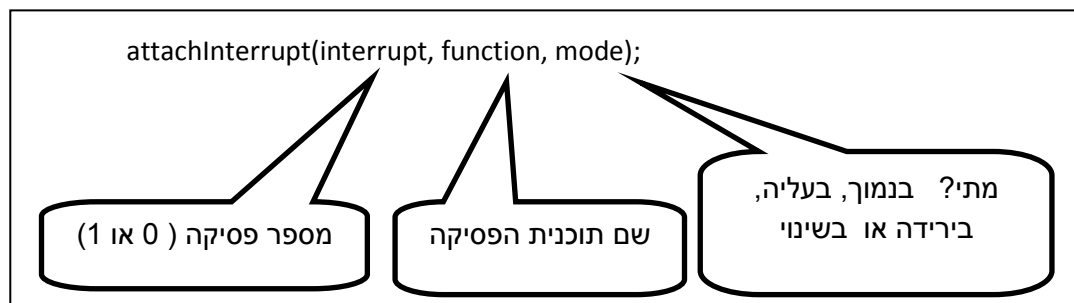
- i. שיטת ה'תשאול' - (Polling). בשיטה זו, המעבד בודק באופן מחזורי, בזולאה אינסופית, את מצב הקלט. זו השיטה שנקטנו בתוכניות שכתבנו. המיקרובקר דגם באופן מחזורי את הדק 9 (את מצב ה Echo) ובדק האם המצב הוא '1' או '0'. היתרון בשיטת התשאול הוא הפשטות במימוש בחומרה ובתוכנה. אבל ראינו שלשיטה זו יש חיסרון, המיקרובקר מבזבז זמן בדגימה אינסופית של מצב הקלט, זאת גם כאשר מצב הקלט כלל לא השתנה.
- ii. שיטת ה'פסיקה' - (Interrupt). בשיטה זו כל עוד מצב הקלט לא השתנה, המיקרובקר ממשיך בפעילות שגרתית של הרצת תוכנית ראשית (הכוללת פעולות כלשהן). רק כאשר יש 'אירוע', כלומר, רק כאשר מצב ה'קלט' משתנה (ואנו קוראים למצב זה 'פסיקה'), המיקרובקר מפסיק את פעילותו השגרתית ומבצע 'תוכנית פסיקה' (תוכנית שנכתבה מראש עבור ארוע ה'פסיקה'). בסיום תוכנית הפסיקה, המיקרובקר חוזר לתכנית הראשית (לאותה שורה בתוכנית שהמיקרובקר היה בו לפני ארוע ה'פסיקה') זאת עד לבקשת 'פסיקה' נוספת. בשיטת ה'פסיקה' המיקרובקר לא מבזבז זמן בהמתנה ל'אירוע'...

לארדואינו אנו יש שני מקורות פסיקה חיצוניים:

interrupt 0 - בהדק 2.

interrupt 1 - בהדק 3.

הפקודה שמאפשרת 'בקשת פסיקה' חיצונית מהמיקרובקר היא:



לדוגמא ההוראה:

```
attachInterrupt(0, blink, CHANGE);
```

פירושה: איפשרו בקשת פסיקה interrupt 0, כאשר יש CHANGE (עליה או ירידה במבוא הדק 2).

בעקבות 'בקשת הפסיקה' יש לבצע תוכנית פסיקה ששמה blink().

חסימת האפשרות לבקשת 'פסיקה' מתבצעת באמצעות ההוראה:

```
detachInterrupt(interrupt); // interrupt – (0 או 1) מספר הפסיקה
```

לדוגמא ההוראה:

```
detachInterrupt(0); // interrupt 0 חסום את האפשרות לבקשת פסיקה
```

3. נתק את מוצא **Echo** (של החיישן) מהדק 9 וחבר אותו **להדק 2** (מבוא interrupt 0 של ארדואינו)

- בתוכנית שלהלן המיקרובקר מבצע שתי משימות במקביל.
 i. מדידת המרחק באופן מחזורי (ע"פ המלצת היצרן כל 50ms).
 ii. הדלקת הled שבהדק 13 כאשר המרחק קטן מ 15 ס"מ.

```
#define trigPin 8 // בהדק 8 נותנים את ה'דרבון' של 10uS
#define echoPin 2 // חיבור מוצא החיישן Echo להדק 2 (interrupt 0)
#define led 13 // להדק 13 מחובר led (על הכרטיס)

unsigned long time=0;

unsigned long duration; // משתנה עבור משך הזמן של רוחב הפולס
int distance=0; // משתנה עבור המרחק בס"מ

//----- פונקציה trigUs() נותנת 'דרבון' של 10uS במבוא trigger של החיישן -----
void trigUs() {
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    attachInterrupt(0, startCount, RISING); // startCount () יש לבצע: יש 'עליה' בהדק 2,
}

//----- פונקציה startCount() מאפשרת את מדידת הזמן -----
void startCount() {
    duration=micros(); // קרא זמן נוכחי למשתנה duration
    detachInterrupt(0); // חסום פסיקה מספר 0.
    attachInterrupt(0, measurement, FALLING); // measurement() בצע: יש 'ירידה',
}

//----- פונקציה measurement() מודדת את הזמן ומחשבת את המרחק -----
void measurement()
{
    duration=micros()-duration; // ההפרש בין הזמן העכשוי (ב 'ירידה') לזמן הקודם (ב 'עליה')
    distance=duration/58; // חישוב המרחק בס"מ והצבה למשתנה distance
    detachInterrupt(0); // חסום פסיקה מספר 0.
}
```

```

}

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

  pinMode(trigPin, OUTPUT);

  pinMode(echoPin, INPUT);

  pinMode(led, OUTPUT);

  trigUs(); // כדי להתחיל את התהליך צריך לתת פעם אחת 'דרבון' של 10uS ואפשר פסיקה
}

void loop()
{
  if(millis()-time>50) { // מדידת מרחק כל 50ms
    Serial.print(distance); // הדפס את המרחק בס"מ
    Serial.println(" cm"); // הדפס ס"מ ועבור שורה
    trigUs(); // יצירת 'דרבון' של 10uS
    time=millis();
  }

  if (distance<15 ) // אם המרחק קטן מ 15 ס"מ
    digitalWrite(led,HIGH); // הדלק את הled
    else // אחרת
    digitalWrite(led,LOW); // כבה את הled
  }

```

4. שנה את התוכנית כך שהled יידלק רק כאשר המרחק קטן מ 20 ס"מ וגם גדול מ 10 ס"מ.
5. נכתוב תוכנית שיוצרת צליל שתלוי במרחק (תדר ברמקול בתלות המרחק הנמדד).
- א. חבר את הרמקול להדק 9. (חוט אחד להדק 9 , החוט השני להדק GND)
- ב. הוסף לפונקציה ה `setup()` את ההוראה: `pinMode(9, OUTPUT);` המגדירה את הדק 9 כמוצא.

ג. החלף את פונקציה `loop()` מתרגיל 3 בפונקציה `loop()` שלהלן:

```

void loop()
{
  tone(9, distance*5+500); // distance*5+500 : צור בהדק 9 צליל שהוא תדר של
  if(millis()-time>50) {
    Serial.println(distance);
  }

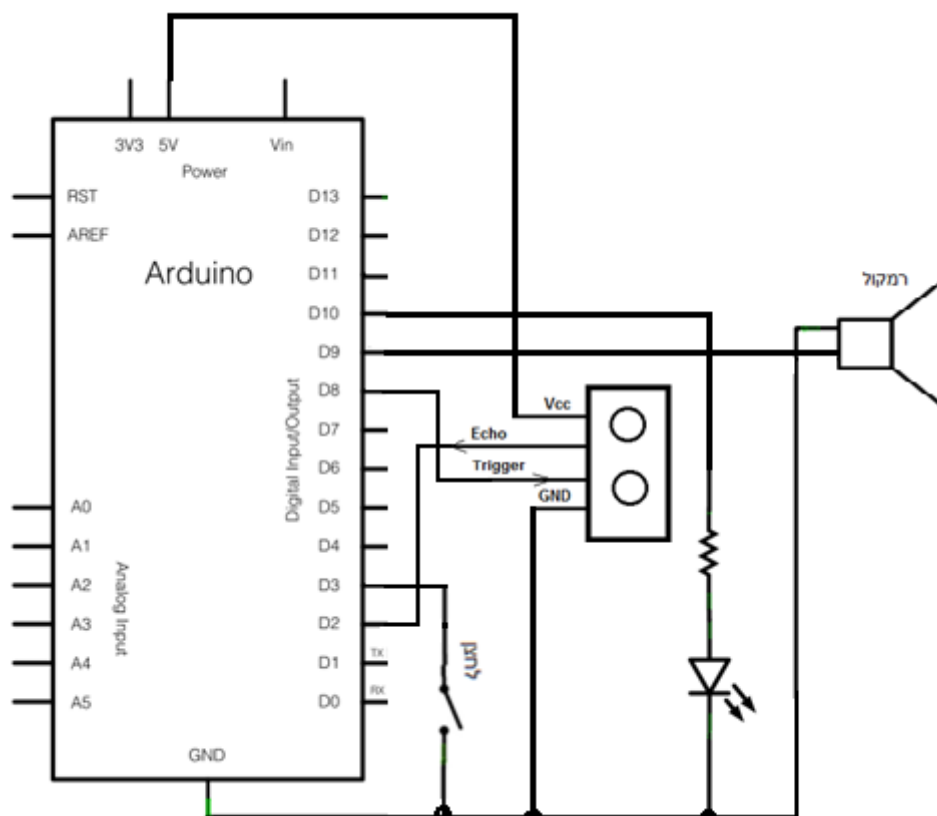
```

```

    trigUs();
    time=millis();
}
}

```

6. נוסף למערכת לחצן להדק 3 (למבוא של: interrupt 1) ונוסף לד שיחובר להדק 10 .
כמשורטט:



כ"כ נוסף תוכנית פסיקה נוספת (בשם: powerLed) עבור בקשת פסיקה_1 . בעקבות לחיצה על הלחצן (ירידה ל '0' במבוא פסיקה_1) הארדואינו יבצע משימה של שינוי עוצמת ההארה של הled.

כלומר, הארדואינו ישמיע צליל בתלות המרחק (ויציג את המרחק על גבי המוניטור). בנוסף (במקביל) בכל פעם שיש לחיצה על הלחצן, הארדואינו ישנה את עוצמת ההארה של הled.

```

#define trigPin 8
#define echoPin 2
#define speaker 9
#define button 3
#define led 10

```

```

unsigned long time=0;
unsigned long duration;
int distance;
int ledPwm=5; // ערך התחלתי עבור הארת הLED

void trigUs()
{
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    attachInterrupt(0, startCount ,RISING);
}

void startCount()
{
    duration=micros();
    detachInterrupt(0);
    attachInterrupt(0, measurement, FALLING);
}

void measurement()
{
    duration=micros()-duration;
    distance=duration/58;
    detachInterrupt(0);
}

void powerLed () {
    detachInterrupt(1); // חסום בקשת פסיקה מספר 1
    ledPwm+=20; // הגדל הערך של PWM עבור הארת הLED
    if (ledPwm>255) ledPwm=5; // חזור לערך של 5 אם הערך עבר את המקסימום,
    analogWrite(led,ledPwm); // הדלק את הLED בעוצמה לפי הערך של ledPwm
    delay(40);
    attachInterrupt(1, powerLed, FALLING);
}

void setup()

```

```

{
  Serial.begin(9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(led, OUTPUT);
  pinMode(speaker, OUTPUT);
  pinMode(button, INPUT_PULLUP);
  analogWrite(led, ledPwm);
  attachInterrupt(1, powerLed, FALLING); // בקשת פסיקה_1 איפשור
  trigUs();
  delay(30);
}

void loop()
{
  tone(9, distance*5+500);
  if(millis()-time>50) {
    Serial.print(distance);
    Serial.println(" cm");
    trigUs();
    time=millis();
  }
}

```

7. כתובת תוכנית אשר תגרום ללד (שבהדק 10) להבהב בקצב שתלוי במרחק. ככל שהמרחק יגדל, קצב ההבהוב יקטן!.

הערה: בכתיבת התוכנית עליך להקפיד שדגימת המרחק תהיה כל 50mS.

רמז: אפשר להיעזר למשל ב: (if(millis()-time_1.....))

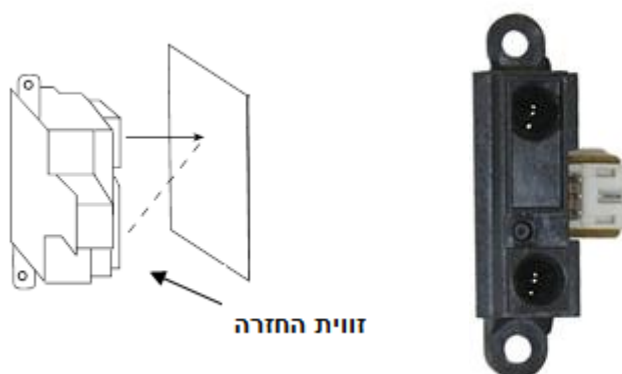
8. שנה את התוכנית מתרגיל 7 כך שקצב ההבהוב יהיה הפוך. ככל שהמרחק יגדל, קצב ההבהוב יגדל....

9. שנה את תרגיל 8 כך שהמצב ההתחלתי יהיה: "ככל שהמרחק יגדל, קצב ההבהוב יקטן". בעקבות לחיצה על הלחצן (שמחובר להדק 3) התנאי יתהפך: "ככל שהמרחק יגדל, קצב ההבהוב יגדל....". בעקבות לחיצה נוספת, התנאי שוב יתהפך: "ככל שהמרחק יגדל, קצב ההבהוב יקטן". וכך שוב ושוב....

הערה: השינוי צריך לקרות בעקבות בקשת פסיקה_1

פעילות 13 - חיישן מרחק GP2Y0A21YK

החיישן מבוסס על משדר ומקלטי אור א.א. עיקרון הפעולה של החיישן מבוסס על עיקרון פעולה של **טריאנגולציה** (Triangulation). כלומר, החיישן משדר קרן אור א"א (באורך גל של $850 \pm 70 \text{ nm}$) שפוגעת בעצם/משטח ומוחזרת ממנו אל מקלטי האור. ע"פ זווית הפגיעה (זווית החזרה) במקלטי האור, החיישן יודע לחשב את המרחק מהעצם הנמדד.

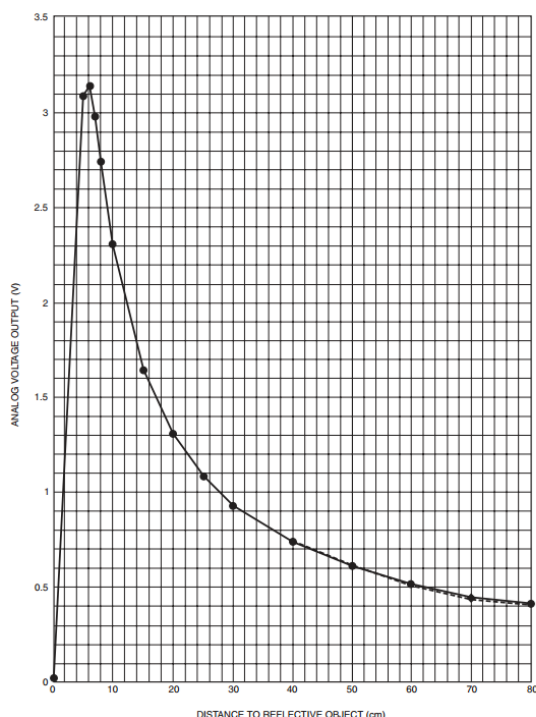


תכונות החיישן

- טווח המדידה: 10 עד 80 ס"מ
- מתח אספקה: 5 V
- צריכת זרם ממוצעת: 33 mA
- קצב רענון מידע: 25 Hz / 40 ms
- זווית פעולה מרבית ביחס למשטח ישר: 40°

ניתוח אופיין הרכיב

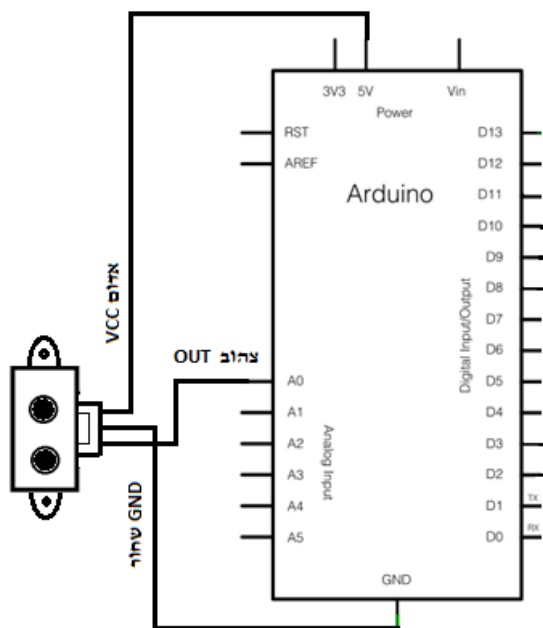
מהאופיין של החיישן (מתח כפונקציה של מרחק), ניתן לראות:



- עוצמתו של המתח המתקבל במוצא החיישן הוא ביחס הפוך למרחק.
- עוצמת המתח המקסימלי היא כ-3.2V (במרחק של כ-6 ס"מ)
- האפיין לא ליניארי.
- יש תחום כפול של ערכי מרחק. תחום אחד בין 0-6 ס"מ והתחום השני בין 6-80 ס"מ.
- לדוגמא: מתח מוצא של 1V יכול לייצג גם מרחק של 26 ס"מ אבל באותה מידה גם מרחק של 2 ס"מ.

- לקבלת רזולוציה טובה רצוי ש'מתח הייחוס' יהיה בקרוב 3.2V. בכרטיס הארדואינו אנו אפשר לחבר את מבוא 'מתח הייחוס' AREF אל מקור המתח 3.3V שמובנה בכרטיס.
- לנוחות המדידה רצוי להמיר את רמת המתח לערכי מרחק (בס"מ)

1. חבר את החיישן כמתואר בשרטוט.



- א. מוצא החיישן (החוט הצהוב) מחובר למבוא האנאלוגי A0.
- ב. החוט האדום והשחור מספקים מתח לחיישן.
- ג. בשלב זה 'מתח היחוס' הוא 5v. (בשלב זה לא צריך לעשות כלום)

2. התוכנית שלהלן מציגה את הערך הדיגיטלי שמתקבל מהחיישן במוניטור. כתוב, הרץ ובדוק.

```
void setup()
{
  Serial.begin(9600);
}
void loop()
{
  int val=analogRead(A0); // הצב במשתנה val את הערך הדיגיטלי של החיישן
  Serial.println(val);     // הדפס את הערך הדיגיטלי של החיישן
  delay(250);             // המתנה של 0.25 שניה כדי שנצליח לקרוא את הנתונים
}
```

אפשר לראות :

א. ככל שהמרחק קטן יותר, הערך הדיגיטלי של המתח גדול יותר. אבל כשהמרחק הוא מתחת ל 6 ס"מ , המגמה מתהפכת, ככל שמתקרבים לחיישן הערך הדיגיטלי של המתח הולך ונהיה קטן יותר.

ב. הערך המכסימלי הוא בקרוב 675. (ברירת המחדל ל 'מתח היחוס' 5v. לכן הערך המכסימלי במוניטור יהיה בקרוב $V_{max} = \frac{3.3}{5} \times 1023 = 675$). כלומר, הרזולוציה נמוכה.

על מנת להגדיל את הרזולוציה נוסיף את ההוראה analogReference(EXTERNAL); . ונחבר את המבוא AREF למתח ייחוס של 3.3v (מתח של 3.3v נמצא בכרטיס ליד ה 5v)

3. התוכנית שלהלן מציגה את ערך החיישן עם מתח ייחוס של 3.3v

```
void setup()
{
```

```

Serial.begin(9600);
analogReference(EXTERNAL); // AREF החיצוני לפי ההדק
} //AREF להדק 3.3v
void loop()
{
  int val=analogRead(A0);
  Serial.println(val);
  delay (250);
}

```

מהתבוננות בנתונים רואים שגם במרחק קבוע, יש לחיישן קפיצות מתח ('רעש'...). אפשר ומומלץ לבדוק את מתח המוצא של החיישן באמצעות האוסילוסקופ. (עוצמתו ותדירותו של ה'רעש' תלויים גם במרחק הנמדד).

נכתוב תוכנית שמציגה במוניטור את ההפרש בין קריאה לקריאה (עבור אותו מרחק).

4. התוכנית שלהלן מציגה את הערך הנוכחי ואת ההפרש בין הערך הנוכחי לערך הקודם.

```

int val; // משתנה שמכיל את הערך הנוכחי
int val_1; // משתנה שמכיל את הערך הקודם
void setup()
{
  Serial.begin(9600);
  analogReference(EXTERNAL);
  val_1=analogRead(A0); // קריאה של ערך התחלתי למשתנה שמכיל את הערך 'הקודם'
  delay(40); // זמן הרענון של החיישן הוא כ 40ms
}

void loop()
{
  val=analogRead(A0); // קרא את הערך הנוכחי
  Serial.print("value=");
  Serial.print(val); // הצג במוניטור את הערך הנוכחי
  Serial.print(" Difference=");
  Serial.println(val-val_1); // הצג את ההפרש בין הערך 'הנוכחי' י לערך 'הקודם'
  val_1=val; // הפוך את הערך 'הנוכחי' לערך 'הקודם'
  delay (250); // המתן 0.25 שניה כדי שנצליח לקרוא את הנתונים במוניטור
}

```

מהתבוננות בנתונים רואים שמדי פעם (עבור אותו מרחק) יש הפרש גדול (יחסית) בין הקריאות. ניתן לפתור את הבעיה' בחומרה (למשל: הוספת קבל גדול בין ה Vcc ל GND ואולי גם קבל קטנטן בין המוצא ל GND). ואפשר למזער את הבעיה' בתוכנה ע"י חישוב ממוצע של כמה קריאות.

5. התוכנית שלהלן מציגה במוניטור ממוצע של 6 קריאות.

```

int val;
int val_1;

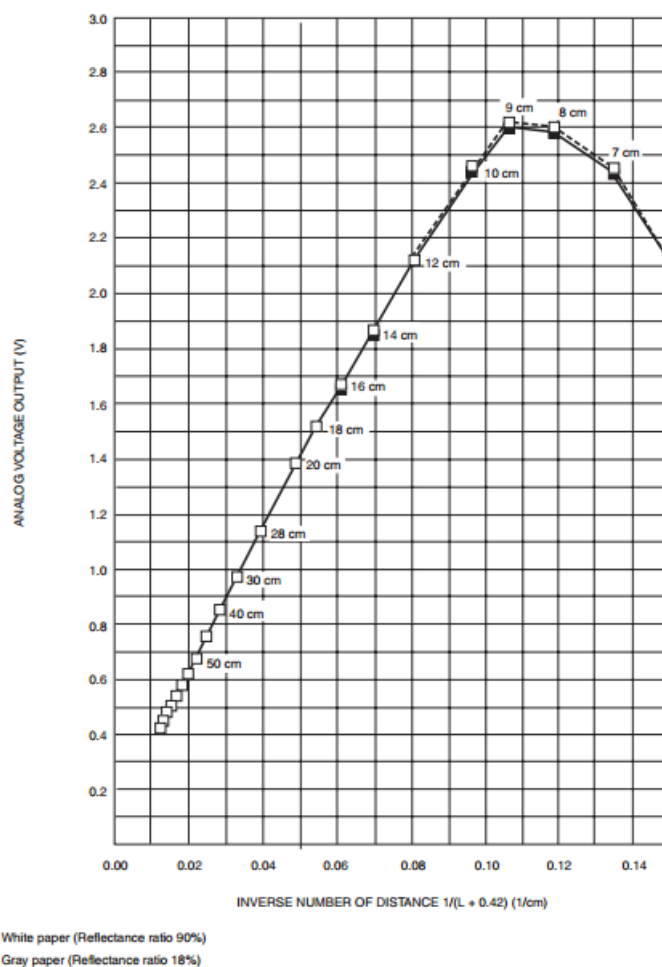
```

```

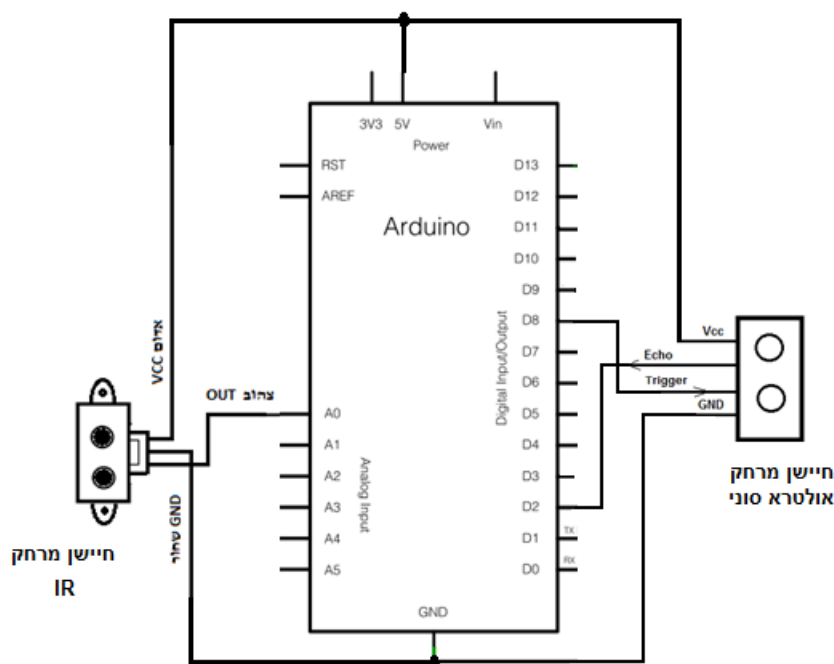
void setup()
{
  Serial.begin(9600);
  analogReference(EXTERNAL);
  val_1=analogRead(A0);
  delay(40);
}
void loop()
{
  val=0;
  for (int i=0; i<6;i++) {      // קרא 6 פעמים, ערך נוכחי מהחיישן
    val+=analogRead(A0);
    delay (40);                // המתן זמן רענון של 40ms
  }
  val/=6;                      // חשב את הממוצע של הערך הנוכחי
  Serial.print("value=");
  Serial.print(val);           // הדפס את הממוצע של הערך נוכחי
  Serial.print("  Difference=");
  Serial.println(val-val_1);    // הדפס את ההפרש בין הממוצע הנוכחי לממוצע הקודם
  val_1=val;                   // הפוך את הערך 'הנוכחי' לערך 'הקודם'
}

```

6. המידע המתקבל במוצא החיישן הוא מתח בתלות המרחק. אנחנו מעוניינים במרחק עצמו. היצרן נותן גם גרף ליניארי של המתח בתלות $\frac{1}{\text{המרחק}}$ (יחידות של 1/cm). אתגר..... כתוב תוכנית שתציג במוניטור את המרחק בס"מ.



7. הוסף את חיישן האולטרא סוני אל כרטיס הארדואינו כמשורטט:
כתוב תוכנית שתציג במוניטור את ערכי המרחק (בס"מ) של שני החיישנים.
האם יש התאמה?

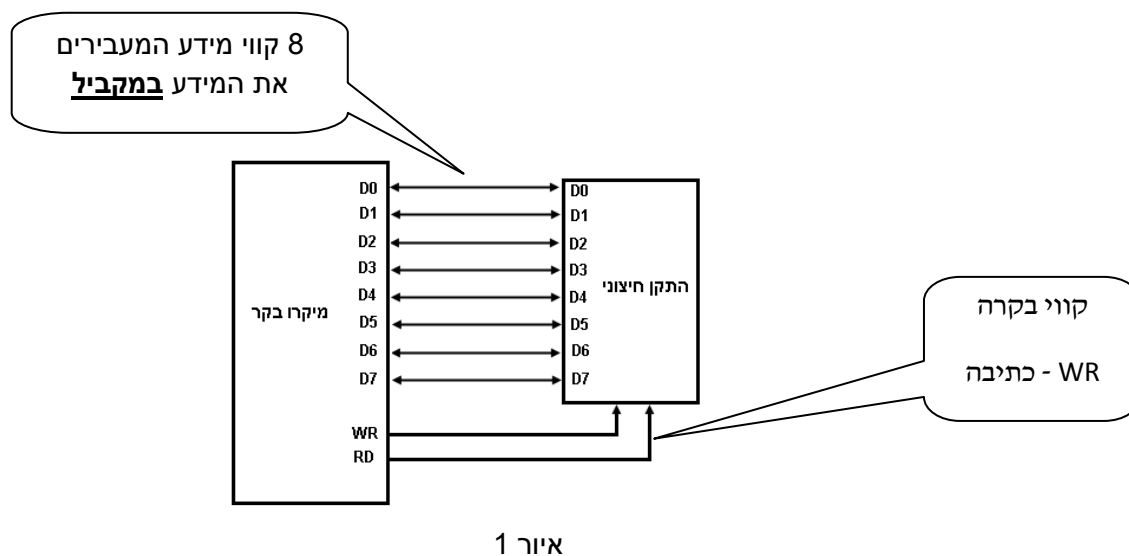


8. להדק 13 בכרטיס מחובר LED . כתוב תוכנית שמדליקה את הLED בתלות המרחק המתקבל מהחיישנים.
אם המרחק המתקבל מחיישן האולטרא סוני גדול מהמרחק המתקבל מחיישן ה IR , הLED יהבהב בקצב של 2HZ. אחרת, הLED יהבהב בקצב של 10HZ.
9. נתק את חיישן האולטרא סוני. חבר לכרטיס הארדואינו מנוע DC כמתואר בפעילות 8 סעיף.... כתוב תוכנית אשר תגרום למנוע להסתובב בתלות המרחק. כל עוד המרחק קטן מ 20 ס"מ המנוע מסתובב במהירות 200/255 . אם המרחק גדול מ 20 ס"מ המנוע עוצר.
10. כתוב תוכנית שתגרום למנוע להסתובב כך:
- עבור מרחק קטן מ 10 ס"מ המנוע מסתובב בכיוון אחד במהירות 200/255.
 - עבור מרחק שבין 10 ס"מ עד ל 20 ס"מ המנוע עוצר.
 - עבור מרחק גדול מ 20 ס"מ המנוע הופך כיוון ומסתובב במהירות 100/255.
11. כתוב תוכנית כך שמהירות המנוע תלויה במרחק. ככל שהמרחק גדול יותר המהירות גדולה יותר... ולהפך , ככל שהמרחק קטן יותר המהירות קטנה יותר....

פעילות 14 - Bluetooth ותקשורת טורית (ה UART)

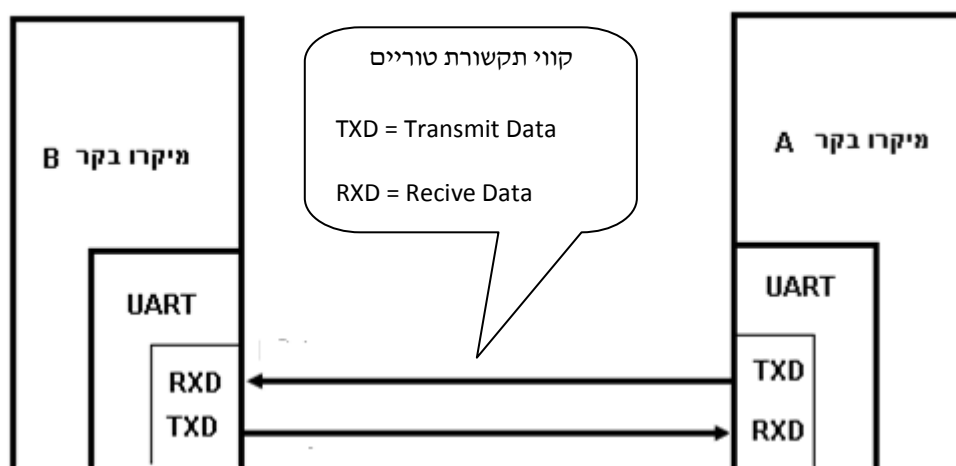
מבוא - מהי תקשורת טורית ו UART

כדי להעביר (לשדר) מידע מהמיקרו בקר להתקנים חיצוניים או כדי לקבל (לקלוט) מידע אל המיקרובקר ניתן להשתמש בשתי דרכים. דרך אחת היא השיטה המקבילית. בשיטה המקבילית לכל סיבית יש קו מידע פרטי משלו והמידע מועבר במקביל (יחד, בו זמנית). בשיטה זו עבור שידור/קליטה של BYTE נצטרך 8 קווי נתונים. בנוסף, אם מדובר בשידור וקליטה של נתונים, נצטרך 2 קווי בקרה נוספים, אשר יקבעו האם הנתונים משודר מהמיקרו מחשב אל ההתקן החיצוני, או להיפך. (ראה איור 1)



(אם המיקרו צריך "לדבר" עם כמה התקנים חיצוניים נצטרך להוסיף גם 'קווי כתובת' לבחירת ההתקן...)

לשיטה המקבילית יש יתרון של 'מהירות'. אך היא 'יקרה'. לעומת זאת בתקשורת טורית יש קו נתונים אחד להעברת המידע. הבקר מעביר את המידע 'בטור'. כלומר, סיביות הנתונים מועברות האחת אחרי השנייה, (סיבית אחת אחרי סיבית). היתרון של השיטה הטורית הוא חיסכון בכסף ואפשרות נוחה להעברת נתונים למרחקים גדולים. בארדואינו קיים רכיב תקשורת טורי מובנה בשם UART (Universal Asynchronous Reciver Transmitter). רכיב ה UART יכול לשדר ולקלוט מידע בו זמנית.



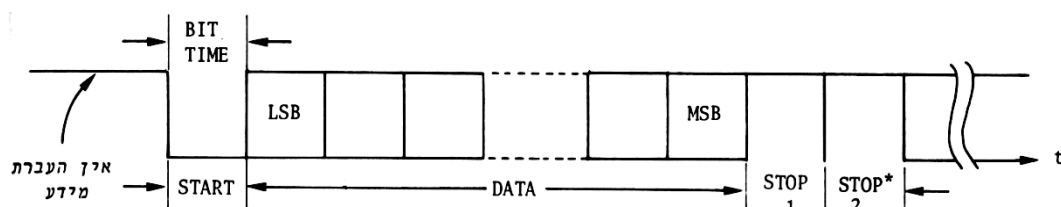
בתקשורת טורית ישנם 3 אופני עבודה :

- חד כיווני (Simplex) - שידור רק בכיוון אחד.
 - חצי דו כיווני (Half-Duplex) - בכל רגע נתון, רק צד אחד משדר והשני קולט ולהפך, זהו האופן הנפוץ ביותר.
 - דו כיווני מלא (Full-Duplex) - כל צד במערכת התקשורת יכול לשדר ולקלוט בו זמנית.
- כאמור, ה UART יכול לשדר ולקלוט מידע בו זמנית. כלומר, לעבוד ב Full-Duplex.

במונח 'אסינכרוני' (Asynchronous) מתכוונים לכך שהמשדר יכול לשדר בכל עת והמקלט חייב להיות מוכן תמיד לקבלת מידע מהמשדר. זאת ללא הודעה מוקדמת וללא תלות ב 'שעון'. בתקשורת אסינכרונית, מרווחי הזמן בין שידור התווים (תו' יחשב כ- 8 סיביות מידע רצופות) אינם בהכרח זהים.

פרוטוקול התקשורת (הנוהל להעברת נתונים)

כאשר ה UART משדר 'תו', פרוטוקול התקשורת שה UART מבצע הוא כמוראה באיור הבא (איור 3):



איור 3 * לא תמיד קיים.

שלב א - ה UART משדר START BIT ($START\ BIT = 0$). התחלת שידור. הסבר: כאשר אין העברת מידע המצב בקו השידור הוא '1'. ברגע שה UART מתחיל לשדר, הוא מוריד את הקו ל '0'. הירידה מ '1' ל '0' מאפשרת למקלט להסתנכרן על התחלת השידור.

שלב ב' - ה UART משדר את 8 סיביות המידע (הסיבית הראשונה היא סיבית ה-LSB).

שלב ג' - ה UART משדר STOP BIT אחד או שניים. ($STOP\ BIT = 1$)

לכל סיבית מוקצה 'זמן סיבית' (BIT TIME) שהוא קבוע ומוסכם מראש. זמן הסיבית נקבע ע"י ה Baud Rate (קצב העברת הסיביות בקו. כלומר, כמות הסיביות העוברות בקו בשנייה אחת. Bits per second). הקצב המקובל הוא 2400, 4800, 9600, 14400, 19200, 38400, 57600, 115200 ביטים \ לשניה. אולם קצב ההעברה יכול להיות גם כל קצב אחר שנבחר. (בהמשך נסביר כיצד קובעים את ה Baud Rate).

שאלה:

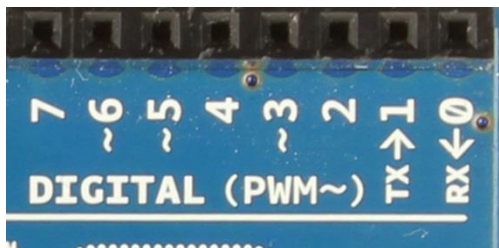
מהי כמות ה'תווים' (BYTES) המקסימאלית שאפשר להעביר בדקה אחת בקו נתונים טורי, אם נתון ש_

- Boud rate = 2400 [bit/sec] (קצב ההעברה)
- ה'תו' הוא בן 8 סיביות
- קיים STOP BIT אחד

פיתרון:

$$\text{כמות ה'תווים' בדקה} = \frac{2400[\text{bit/sec}]}{1[\text{start_bit}] + 8[\text{bit}] + 1[\text{stop_bit}]} \times 60[\text{sec}] = 14,400[\text{BYTE}]$$

כאמור, לאדואינו UNO יש רכיב UART מובנה בתוך המיקרובקר. הדק 0 של הארדואינו משמש כקו התקשורת הטורית לקליטה - RXD. והדק 1 משמש כקו התקשורת הטורית לשידור - TXD



לארדואינו יש חוצץ (bufer) של 64 בתים לאחסון המידע שנקלט בתקשורת הטורית.

הפונקציה Serial.available() מחזירה את מספר התווים שנקלטו בחוצץ.

התוכנית שלהלן פותחת ערוץ תקשורת טורית עם המחשב. התוכנית ממתנה למידע טורי. אם המידע שהתקבל הוא 1, הled שבהדק 13 יידלק. אם המידע שהתקבל הוא 0, הled שבהדק 13 ייכבה. הארדואינו גם שולח הודעה מתאימה למחשב.

1. כתוב את התוכנית ובצע upload לכרטיס הראדואינו.

```
int led;

void setup() {
    pinMode (13,OUTPUT);
    Serial.begin(9600); // פתח ערוץ תקשורת 9600 bps
}

void loop() {
    if (Serial.available() > 0) { // אם יש מידע בחוצץ
        led = Serial.read(); // קרא את המידע אל משתנה led
        if (led=='1') { // אם המידע הוא 1
            digitalWrite(13,1); // הדלק את הled שבהדק 13
            Serial.println("led on"); // שלח הודעה מתאימה למוניטור
        }
        else if (led=='0') { // אם המידע הוא 0
            digitalWrite(13,0); // הכבד את הled שבהדק 13
            Serial.println("led off"); // שלח הודעה מתאימה למוניטור
        }
    }
}
```

```
digitalWrite(13,0); // כבה את ה led שבהדק 13
```

```
Serial.println("led off"); // שלח הודעה מתאימה למוניטור
```

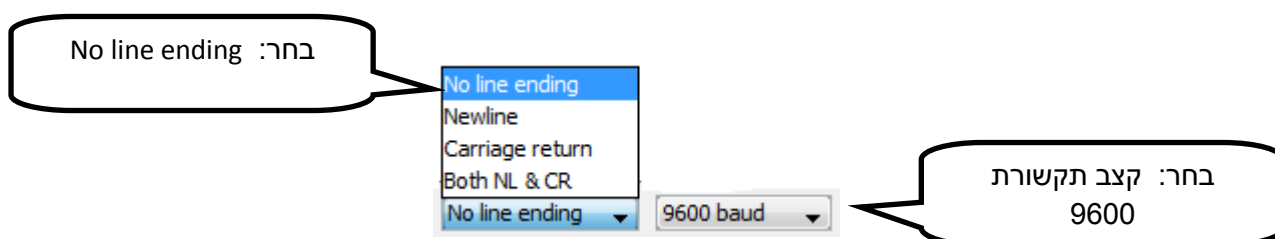
```
}
```

```
}
```

```
}
```

פתח את המוניטור . בתחתית המוניטור (מימין) יש חלונות שאפשר לבחור את קצב התקשורת והאם יש שידור של 'סוף שורה'

וודא שקצב התקשורת הוא 9600 ואין שידור של 'סוף שורה'



1. שדר באמצעות המוניטור 1, ובדוק האם ה led נדלק....
שדר 0 ובדוק.....

שמור את התוכנית . בהמשך נזדקק לה שוב.!



BlueTooth

Bluetooth היא טכנולוגיה (חומרה + פרוטוקול תקשורת) להעברת מידע לטווחים קצרים (10 מטר) **באופן**

אלחוטי (בתדר רדיו בגלים קצרים) בהספק שידור נמוך (1mW) .

טווח התדרים בהם ה Bluetooth משדר הוא 2.4 עד 2.4835 גיגה-הרץ.

מכיוון שמכשירים נוספים פועלים בטווח תדרים זה , פרוטוקול Bluetooth מחלק את הטווח ל 79 ערוצים בני 1 מגה-הרץ כל אחד, ומחליף ביניהם עד 1600 פעם בשנייה.

פרוטוקול ה Bluetooth יכול לחבר עד 8 מכשירים, המצויים בסמוך זה לזה, למעין "מיני-רשת" . ברשת זו יש מכשיר אחד המהווה 'אדון' (Master) וכל השאר הם 'עבדים' (Slaves) המגיבים לשידורים המגיעים מהאדון. את ה Bluetooth המציאה החברה השוודית 'אריקסון' . השם Bluetooth ניתן לטכנולוגיה על שם המלך הוויקינגי של דנמרק מהמאה ה-10 הארלד בלאטאנד (Harald Blatand). המלך האראלד איחד את השבטים השונים והמסוכסכים תחת שרביטו ומכאן בא הרעיון לתת את השם Blatand לטכנולוגיה המאגדת תחת שרביטה מכשירים שונים. Blatand תרגומו "חזות הכהה" אבל בתרגום באנגלית זה שובש ל Bluetooth

הרכיב hc06

- הרכיב פועל ברמת מתח של 3.3v, אך הרכיב עובד היטב גם עם מיקרובקרים שפועלים ברמת מתח של 5v ללא צורך ברכיבים 'מתווכים'.
- מותר לחבר ל Vcc מתח בערכים שבין 3.6v – 6v. מתח מעל 7v יגרום נזק לרכיב.
- צריכת זרם ממוצעת כ 30mA.

הרכיב מגיע עם ברירת מחדל של : Slave, 9600 baud rate, N, 8, 1. Pincode 1234

שפירושו:

- slave - הרכיב לא יכול ליזום התקשרות.
- 9600 baud rat - קצב שידור של 9600 סיביות בשניה.
- N,8,1 - (N) ללא סיבית זוגיות, (8) סיביות לשידור, STOP BIT (1)
- Pincode 1234 - קוד ההתחברות הוא 1234



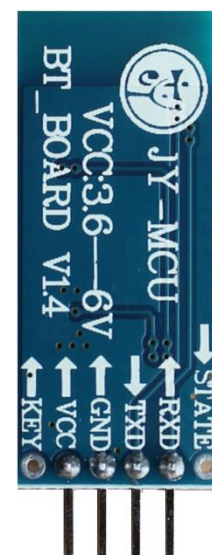
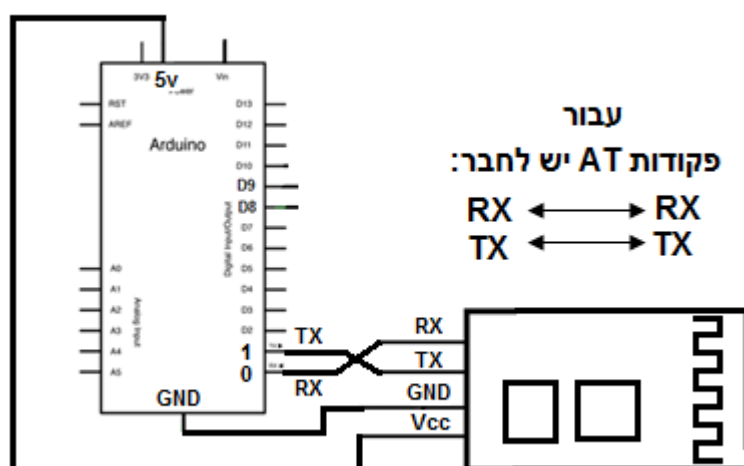
להגדרת הרכיב אחרת מברירת המחדל משתמשים בפקודות AT (AT Command)

2. כדי שנוכל להשתמש במוניטור ובכרטיס הארדואינו להגדרת הרכיב hc06, נוודא שה UART של בקר הארדואינו לא פעיל. כלומר, שהבקר לא מריץ תוכנית עם הוראות Serial. לשם כך נעשה upload לארדואינו לתוכנית שלא עושה כלום

```
void setup() { }
```

```
void loop() { }
```

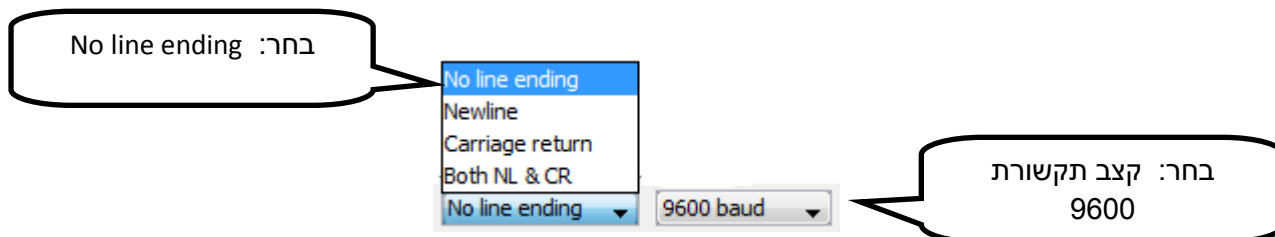
3. להגדרת הרכיב hc06, חבר את הרכיב לארדואינו כמשורטט:



כאשר מחברים מתח לרכיב, הליך מתחיל להבהב.

4. פתח את המוניטור. בפתיחת המוניטור, המחשב יוזם תקשורת עם הרכיב hc06. שים לב: בתחתית המוניטור (מימין) יש חלונות באמצעותם אפשר לבחור את קצב התקשורת והאם יש שידור של 'סוף שורה'

וודא שקצב התקשורת הוא 9600 ואין שידור של 'סוף שורה'

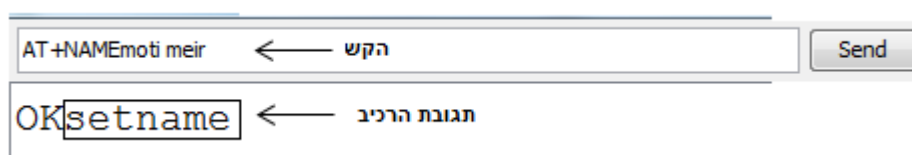


הגדרת רכיב ה Bluetooth עם פקודות AT (AT Command)

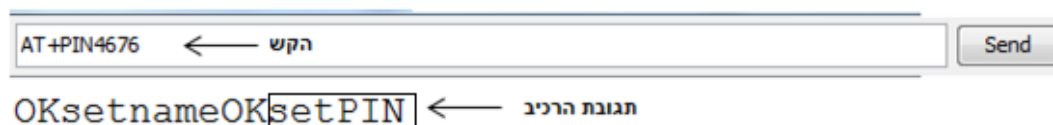
5. לבדיקה האם יש תקשורת, הקש: AT (כמוראה בתמונה)



6. למתן שם לרכיב, הקש: AT+NAME עם השם החדש לרכיב. לדוגמא:



7. לשינוי הקוד הקש: AT+PINxxxx (xxxx קוד חדש) לדוגמא, על מנת לקבוע קוד חדש של: 4676



8. לשינוי קצב השידור יש להקיש AT+BAUDn . (n - מספר טבעי בין 1 ל 8 שקובע את קצב השידור) כדלקמן :

1200-----	1
2400-----	2
4800-----	3
9600-----	4
19200-----	5
38400-----	6
57600-----	7
115200-----	8

לדוגמא, עבור קצב תקשורת של 115200 יש להקיש : AT+BAUD8 (כמוראה בתמונה)

AT+BAUD8 ← הקש

Send

OKsetnameOKsetPINOK115200 ← תגובת הרכיב

אם הגדרנו את הרכיב לקצב תקשורת של 115200 , חייבים לשנות גם את המוניטור לקצב תקשורת של 115200. כמוראה בתמונה.

No line ending ▼

115200 baud ▼

9. כדי לחזור לקצב תקשורת של 9600 , הקש AT+BAUD4

AT+BAUD4 ← הקש

Send

OK9600 ← תגובת הרכיב

No line ending ▼

9600 baud ▼

קבע במוניטור: קצב תקשורת
9600

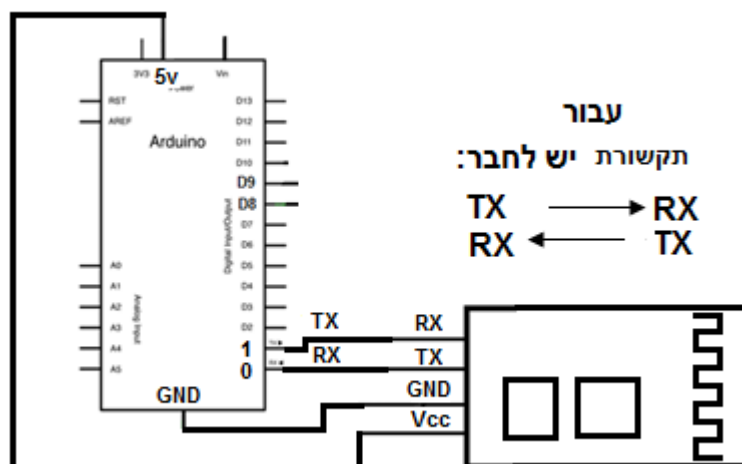
בסעיף 1 הדלקנו וכיבינו את הLED באמצעות המוניטור. במשימה הבאה, נדליק ונכבה את הLED באמצעות הסמרטפון. התקשורת בין הסמרטפון והארדואינו תיעשה בתיווך רכיב הBluetooth.

10. לכרטיס הארדואינו אונו יש מפתח תקשורת טורית (Serial Port) אחד בלבד !. לכן כאשר מבצעים

upload של תוכנית לבקר הארדואינו, יש לנתק את רכיב הBluetooth מכרטיס הארדואינו.

11. טען את התוכנית **מסעיף 1** . (בצע upload של התוכנית מסעיף 1)

12. אחרי שהטענת התוכנית הסתיימה, חבר את רכיב הBluetooth אל הארדואינו כמוראה בתמונה.



שים לב: כאשר מחברים מתח לרכיב, הLED מתחיל להבהב.

13. הורד והתקן מחנות האפליקציות. אפליקציית Bluetooth Terminal (יש הרבה), הראשונה ברשימה מספיק טובה.....)



14. הפעל את התקן הBluetooth בסמרטפון ע"י לחיצה על הסמל

הסמרטפון יסרוק את הסביבה ויגלה בין היתר גם את רכיב הBluetooth שלך, עם השם שבחרת

והגדרת בסעיף 7 (בדוגמא שלי: moti meir)

15. בחר את הרכיב שלך. הסמרטפון יבקש שתזין את קוד ה-pin שבחרת והגדרת בסעיף 8.

16. פתח את אפליקציית הBluetooth Terminal והתחבר לרכיב. הסמרטפון יכתוב הודעה שהוא מקושר

לרכיב. בנוסף, הLED מפסיק להבהב. הLED נשאר זלוק ברציפות.

17. שדר באמצעות האפליקציה שבסמרטפון את התו 1, ובדוק האם הLED (שעל הכרטיס הארדואינו) נדלק.....

שדר 0 ובדוק.....

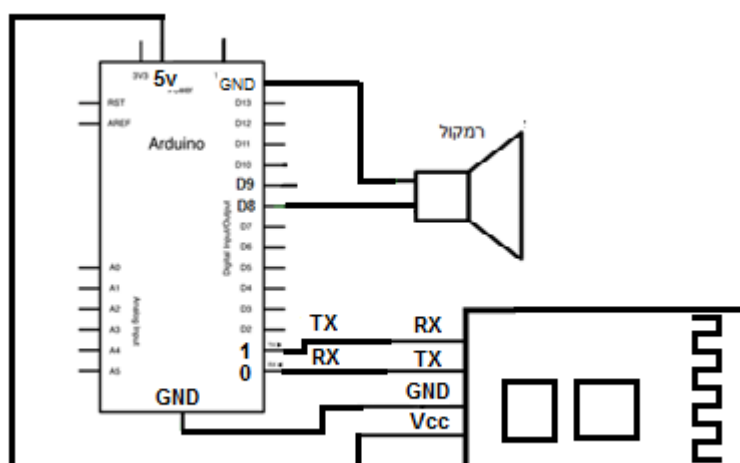
שים לב: הארדואינו גם משדר הודעה מתאימה לסמרטפון.

18. התוכנית שלהלן משמיעה 2 צלילים בתלות הערך המתקבל מהסמרטפון. אם התו המתקבל

מהסמרטפון הוא 1, הארדואינו ישמיע צליל של 1000HZ. אם התו הוא 2, הארדואינו ישמיע צליל

של 2000HZ.

הוסף לכרטיס הארדואינו רמקול להדק 8 כמפורט:



נתק את רכיב ה **bluetooth hc06** מהארדואינו.

טען (upload) את התוכנית אל הארדואינו.

```
int sound;

void setup() {
    pinMode (8,OUTPUT);

    Serial.begin(9600); // פתח ערוץ תקשורת 9600 bps
}

void loop() {
    int sound;

    if (Serial.available() > 0) { // אם יש מידע בחוצץ
        sound= Serial.read(); // קרא את המידע אל משתנה sound
        if (sound=='1' ) { // אם המידע הוא 1
            tone(8,1000); // השמע צליל של 1000HZ
            delay (100);

            Serial.println("sound=1000"); // שדר הודעה sound=1000
        }

        else if (sound=='2') { // אחרת, אם המידע הוא 2
            tone(8,2000); // השמע צליל של 2000HZ
            delay(100);
        }
    }
}
```

```

Serial.println("sound=2000");
}
}
}

```

19. **חבר שוב את הרכיב hc06 אל כרטיס הארדואינו** (כמוראה בשרטוט של סעיף 18).
 שדר 1, שדר 0, בדוק את התגובה. שדר ברצף אחד את: 1212121212 ובדוק....
20. התוכנית שלהלן משמיעה 'אוקטבה' של צלילים. **נתק את רכיב ה bluetooth hc06 מהארדואינו.**
 טען (upload) את התוכנית אל הארדואינו.

```

#define Do      262  // 262 במספר ימיר זאת במספר 262
#define Re      294  // 294 במספר ימיר זאת במספר 294
#define Me      330  // 330 במספר ימיר זאת במספר 330
#define Fa      349  // כנ"ל
#define Sol     392
#define La      440
#define Se      494
#define Do2     523

void setup() {
  Serial.begin(9600);
  pinMode(8,OUTPUT); // קבע את הדק 8 כהדק מוצא
}

int key;

void loop() {
  if (Serial.available() > 0) {
    key=Serial.read();
    switch (key) {
      case '1': tone(8,Do ); delay (400); break; // Do את התדר 8 הוצא להדק
      case '2': tone(8,Re); delay (400); break; // Re את התדר 8 הוצא להדק
      case '3': tone(8,Me ); delay (400); break; // Me את התדר 8 הוצא להדק
      case '4': tone(8,Fa ); delay (400); break;
      case '5': tone(8,Sol ); delay (400); break;
      case '6': tone(8,La ); delay (400); break;
      case '7': tone(8,Se ); delay (400); break;
      case '8': tone(8,Do2 ); delay (400); break;
      default : noTone(8); // הפסק את התדר בהדק 8
    }
  }
}

```

```

    }
  }
}

```

21. חבר שוב את הרכיב hc06 אל כרטיס הארדואינו

שדר ספרות בין 1 ל 8 , בדוק את התגובה.

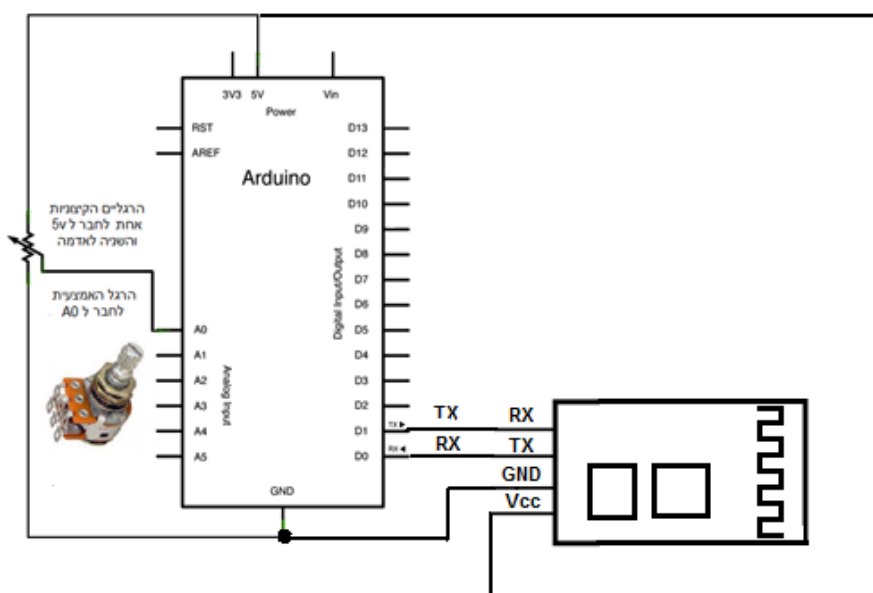
שדר ברצף אחד את : 12345678 ובדוק....

אם ברצונך לנגן, כדאי להוריד אפליקציה כדוגמת : Bluetooth spp pro שמכילה לוח מקשים.

22. התוכנית שלהלן מציגה על גבי תצוגת הסמרטפון את רמת המתח המתקבלת מפוטנציומטר שמחובר למבוא האנלוגי A0 .

למבוא האנלוגי A0 .

חבר את הפוטנציומטר כמשורטט :



23. נתק את רכיב ה bluetooth hc06 מהארדואינו.

24. טען (upload) את התוכנית אל הארדואינו

```

void setup() {
  Serial.begin(9600); // פתח ערוץ תקשורת 9600 bps
}

void loop() {
  int num=analogRead(A0);
  float volt=(5*((float)num/1023));

  Serial.print ("digital=");
  Serial.print(num);
  Serial.print (" Volt=");
  Serial.print(volt);
  Serial.println (" v");
}

```

```
delay(250);  
}
```

25. חבר שוב את הרכיב hc06 אל כרטיס הארדואינו ובדוק .
26. נתק את רכיב ה bluetooth hc06 מהארדואינו והוסף את חיישן המרחק SFR05 כמוראה בשרטוט של סעיף 1 בפעילות 9 .
27. טען את התוכנית של סעיף 1 בפעילות 9 . חבר שוב את הרכיב hc06 אל כרטיס הארדואינו. בדוק שהמרחק מוצג ע"ג הסמרטפון.

תם ולא נשלם....