בית הספר הניסויי
תיכון עירוני "הגליל" – נצרת
בפיקוח משרד החינוך והתרבות
ת.ד. 40 נצרת – ישראל

مدرسة "الجليل" التجريبية
الثانوية البلدية – الناصرة
بإشراف وزارة التربية والتعليم
ص.ب. 40 الناصرة – اسرائيل

פקס 04-6466465 فاكس

טלפון 04-6470473 هاتف

עבודת גמר

במגמת הנדסת אלקטרוניקה ומחשבים
במקצוע מערכות אלקטרוניות
סמל שאלון: 841589

בנושא : חניון חכם

מאת : גאידאא חאג'
ת.ז : 212024442

בהנחיית: עבד אלמג'יד מחליה **2018/2019**

תוכן עניינים

דברי תודה

אני מביעה ומקירה את תודתי למורים האלקטרוניקה בבית ספרי , בלעדיהם לא היינו מצליחים להוציא לאור הלכה למעשה את הפרויקט.

למורה היקר , הנה הגיעה השנה לסופה , ואני רוצה להגיד לך תודה. תודה על הנחינה , תודה על ההשקעה , תודה על ההקשבה שהענקת לנו במשך שלוש שנות מלאות חוויות טובות וקצת פחות.

תודה שהתמודדת עם הכיתה שלנו בצורה הכי טובה למרות כל הקשיים , והיו המון קשיים.

תודה שהתאמצת לתת כל כולך לכל תלמיד.

היית כאוזן קשבת ויד עוזרת כשצריך.

מאחים לך שהשנים הבאות יהיו לך יותר קלות , תמיד נזכור אותך קודם כל כמורה לחיים.

תיאור חניון חכם

הפרויקט המוצע הינו פרויקט הוא חניון חכם הנשלט דרך בקר arduino nano (ATMEGA328P ARDUINO NANO השולט ובודק קיאה מחיישנם על מצב החניון תפוס או פנוי .

יש שני חיישנים IR בכניסה השער הם מזהים אם יש מכונית נכנסת או יוצאת מהחניון ומפעיל מנוע סירו המתאים (יש שני מנועים סירוי אחד לשער כניסה ואחד לשער כציאה)

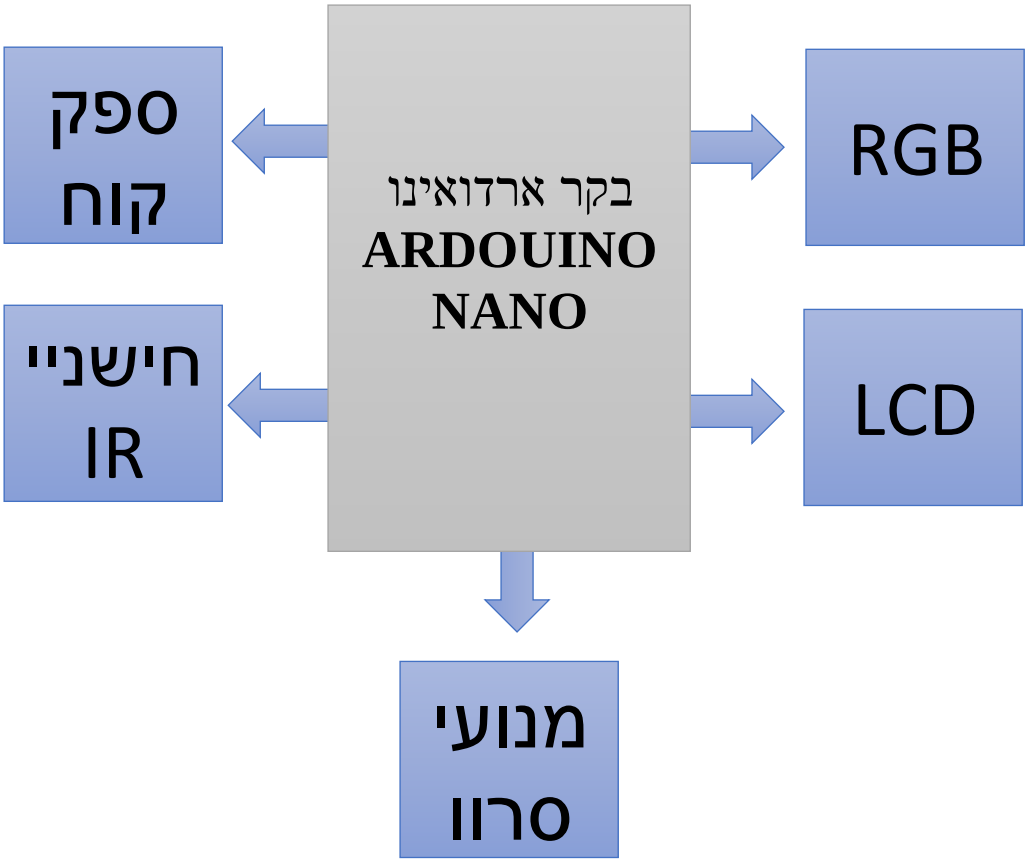
בנוסף יש ארבעה חניונים על כול אחד מהם יש גם חיישן IR הוא מזהה אם יש מקומות בחניון ובנוסף יש RGB מעל כול מקום חניה

הוא דולק ירוק אם אדום השער בכניסה לא יפעל אם אין מקומות ובחינון וגם ישלח הודעה אל LCD אין חניה

מפרט טכני:

1. כרטיס פיתוח למיקרו בקר ATMEGA328P
2. שני מנועים SERVO MOTOR .
3. ארבעה לדים RGB.
4. זמזם.
5. תצוגת I2C LCD.
6. שישה חיישניים מקלט משדר IR

תרשים מלבינים



הסבר סכימת המלבינים :

ארדואינו - לב המערכת, מיקרו בקר עליו נצרבת התוכנה וממנו ניתנות הפקודות לשליטה במערכת כולה.

תצוגת LCD- מציגה את ערך המקומות הריקים והמלאים בחניון.

מד מרחק – הוא חיישן המודד מרחק בינו לבין מכשול מולו.

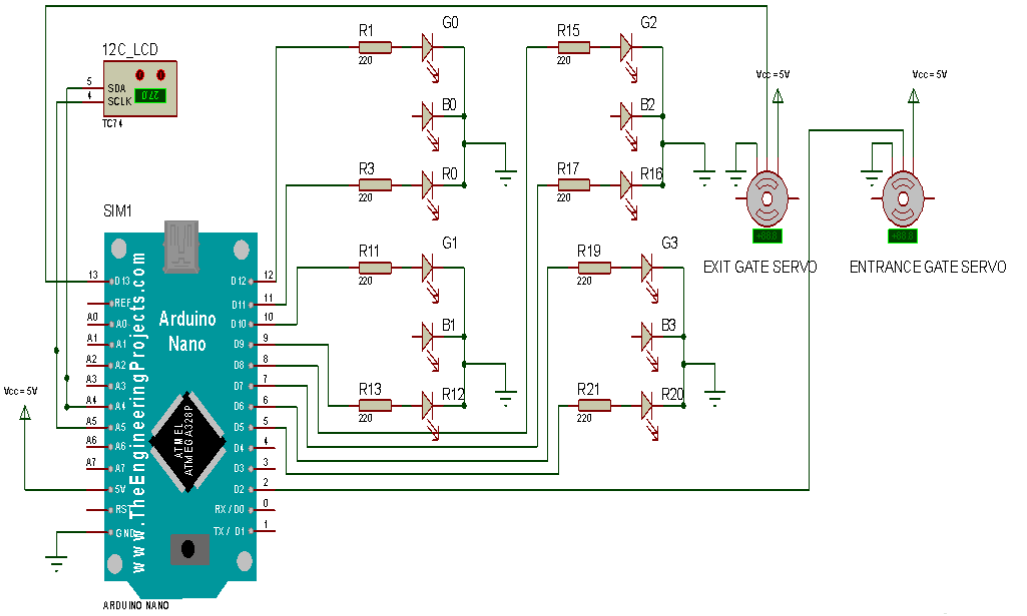
מנוע סרוו - הוא אקטואטור סיבובי או אקטואטור ליניארי המאפשר שליטה מדויקת של המיקום הזוויתי או הליניארי, המהירות והתאוצה.

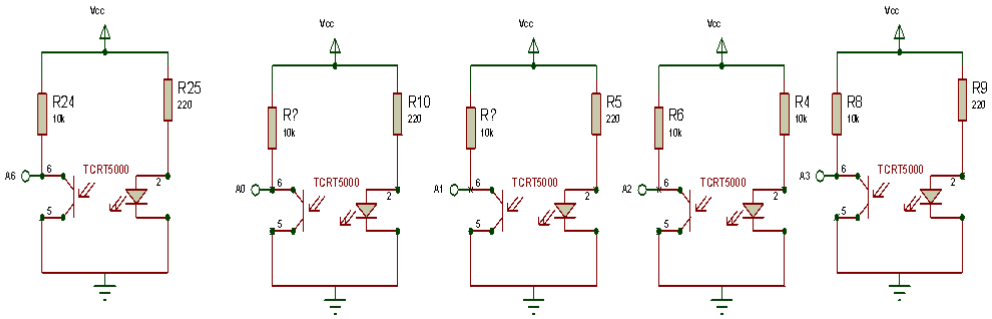
RGB- מכילות לדים בצבעים Red, Green, Blue .

ספק כוח- נותן מתח יציב לרכיבים השווה ל 5v .

חיישן IR - דיודה פולטת אור. היא מאירה אל העצם שרוצים למדוד מה המרחק אליו.

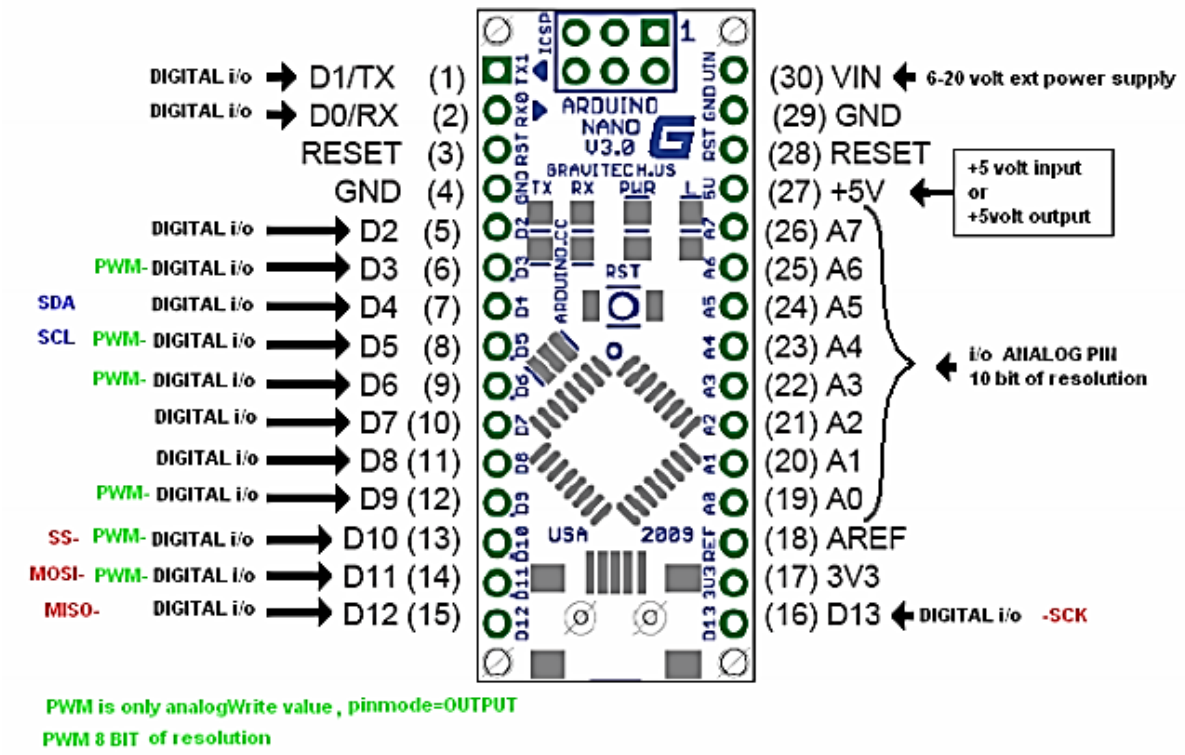
סכימה חשמלית של הפרויקט





כרטיס ארדוינו נאנו

זהו כרטיס מסדרת ארדואינו והחל מגירסה 3 הוא עם מיקרו בקר ATmega 328 . בכרטיס 14 הדקים דיגיטאליים (מ D0 ועד D13), כשכל אחד מהם יכול להיות קלט או פלט לפי תכנות שלנו, ועוד 8 הדקים אנאלוגיים (מ A0 ועד A7). היתרון הגדול של הכרטיס הוא גודלו הזעיר : 0.7*1.7 אינץ' (כ 1.778*4.31 ס"מ). משקלו כ 5 גרם.

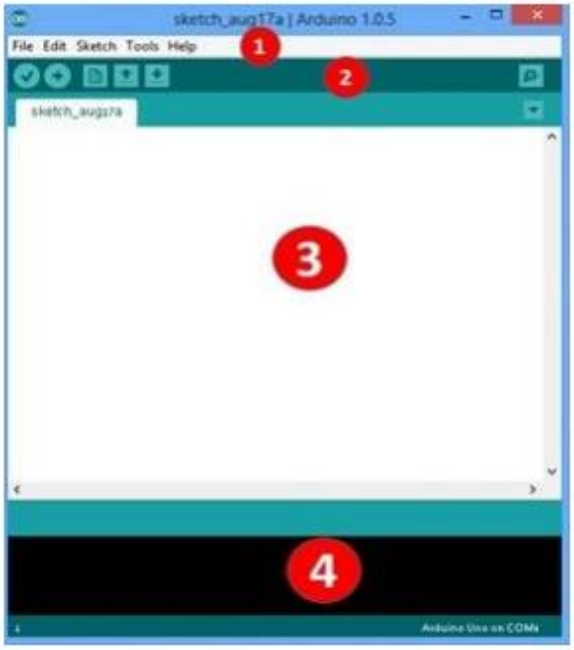


הכרת תוכנת IDE

בדיקת תקינות המערכת והתקשורת

לאחר שתוכנת ה Arduino והדרייבר הותקנו, ניתן ומומלץ לבצע בדיקה קצרה לוודא כי הכל פועל כשורה. מלבד זאת הדגמת הבדיקה יספק לנו היכרות מהירה עם תפריטי IDE החשובים ושלבי העבודה הננקטים לעיתים תכופות במהלך הפיתוח. לביצוע הבדיקה, יש להפעיל את התוכנה על ידי לחיצה על אייקון קיצור הדרך של התוכנה הנראה כדלהלן:

החלון שיפתח יראה כך:



איור-:3 חלון סביבת הפיתוח וחלקיו השונים

חלון סביבת הפיתוח מורכב מארבה חלקים (ממוספרים באיור הקודם):

1. בחלק הראשון מצוי התפריט הראשי

2. בחלק השני מצויים כפתורים לניווט מהיר

3. בחלק השלישי בו מצויים הקודים שכתבנו

4. החלק הרביעי הוא חלון תצוגה להודעות המהדר, אזהרות ושגיאות וכן סוג הכרטיס ולאיזה פורט במחשב הוא מחובר.

לחצני הניווט המהיר

הלחצנים בסרגל הכלים שבחלק השני, (ראה איור -3) מספקים גישה נוחה לפונקציות הנפוצות ביותר בתוך התפריט .

איור-4: לחצני ניווט מהיר

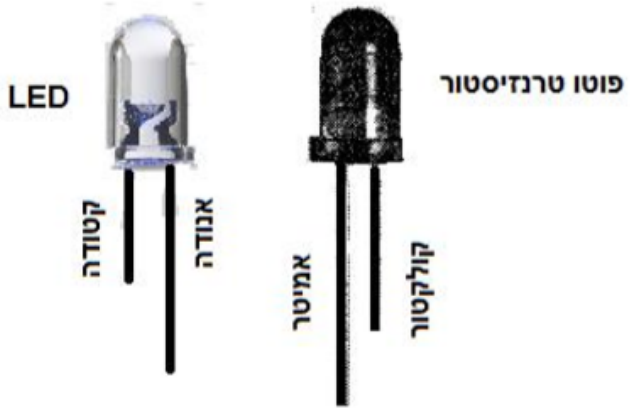
הטבלה שלהלן מפרטת תפקיד כל הלחצנים הנמצאים בסרגל הכלים.

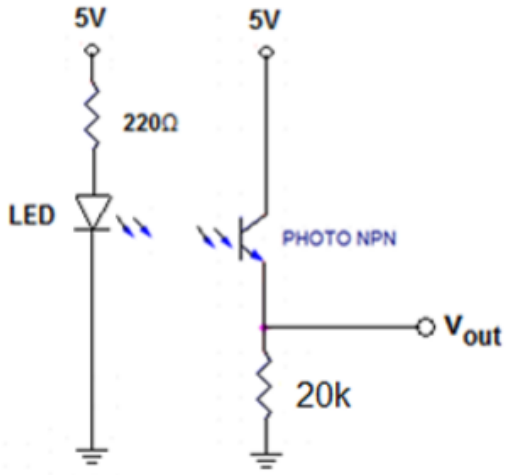
סמל	תיאור
	Verify : מאפשר בדיקת שגיאות בקוד
	Upload : מהדר את הקוד שנכתב ו"מעלה" (צורב" אותו ללוח ה Arduino.
	New : פותח עמוד " Sketch " חדש
	Open : מציג תפריט של כל התוכניות הנמצאים בתקיה. לחיצה על אחד תפתח אותו . בתוך החלון נוכחי.
	Save : שומר את ה Sketch שנכתב.
	Serial Monitor : ממשק בסיסי לתקשורת טורית דו-כיוונית עם לוח Arduino.

חיישן IR (Infra Red)

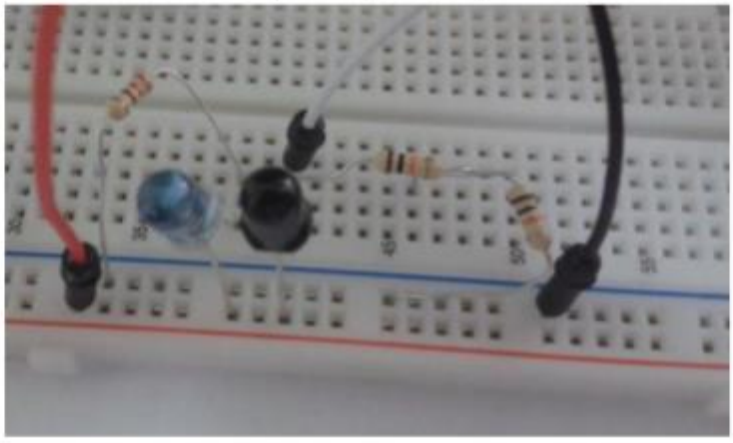
חיישן ה IR הוא פשוט , זול, ויש לו שמושים רבים ומגוונים.....

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 vout יגדל.....





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

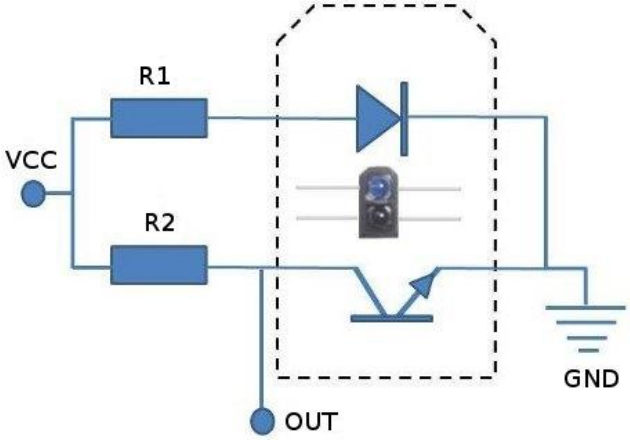


2. חבר את Vout אל המבוא האנלוגי A1 של הארדואינו.
3. התוכנית שלהלן מציגה ע"ג המוניטור את הערך הדיגיטאלי של Vout.

```
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); //  A1  קלוט אל המשתנה,  את הערך האנלוגי מערוץ
        Serial.println(IR); //  הצג על המוניטור את הערך של המשתנה
        delay (100);                    // המתן עשירית השניה
}
```

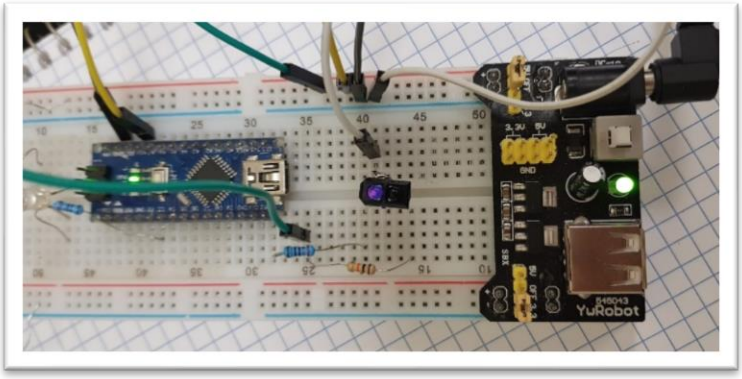
4. בעזרת כף ידך ו/או בעזרת דף נייר, שנה את עוצמת ההחזרה של האור אל הפוטוטרנזיסטור. בדוק את הערכים בתצוגת המוניטור.
5. הפוך את המטריצה כלפי השולחן, כך שהחיישן יפנה כלפי השולחן (הLED והפוטו טרנזיסטור, שניהם כלפי השולחן). בדוק (במוניטור) את הערכים המתקבלים בשני מצבים: כשהחיישן מעל השולחן וכשהחיישן זז הצידה מהשולחן. (כשאין החזר אור מהשולחן).
6. 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 האנאלוגיים, 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.....

סכימה חשמלית של חיישן IR



23/11/2018

בדיקת ו הלחמת חיישני **IR**

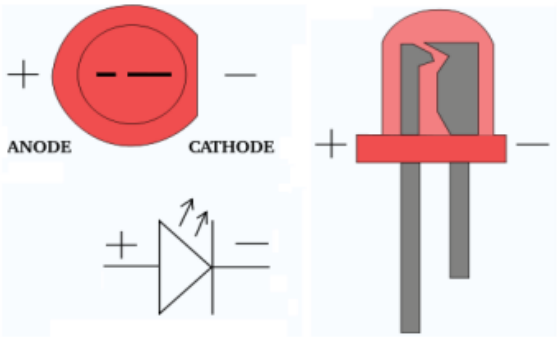


תכנית בדיקת חישן IR

```
 int counter=0;
void setup() {
pinMode(12,OUTPUT);
pinMode(13,OUTPUT);
pinMode(2,OUTPUT);
pinMode(3,OUTPUT);
pinMode(4,OUTPUT);
pinMode(5,OUTPUT);
pinMode(A3,INPUT);
pinMode(A7,INPUT);
pinMode(A2,INPUT);
digitalWrite(4,0);
 digitalWrite(5,1);
 digitalWrite(2,1);
 digitalWrite(3,0);
  digitalWrite(12,1);
 digitalWrite(13,0);

  Serial.begin(9600);
}

void loop() {
 float sensor2 = analogRead(A2);
 float sensor3 = analogRead(A3);
 float sensor7 = analogRead(A7);
 Serial.println(sensor7);

 if( sensor2<1000)

 {
   digitalWrite(12,0);
      digitalWrite(13,1);
 }
 if( sensor2>1000)
 {
     digitalWrite(12,1);
        digitalWrite(13,0);


 }
```

```
 if( sensor7<1000)
{
  digitalWrite(2,0);
     digitalWrite(3,1);
}
if( sensor7>1000)
{
    digitalWrite(2,1);
       digitalWrite(3,0);


}
 if( sensor3<1000)
{
  digitalWrite(4,1);
     digitalWrite(5,0);
   if( sensor3>1000)
   {
       digitalWrite(4,0);
          digitalWrite(5,1);


   }
     }
```

דיודה פולטת אור
Led Light Emitting Diode



LED היא רכיב הבנוי מחומר שנקרא מוליך למחצה (Semiconductor), חומר זה יכול להיות מוליך טוב בתנאי מסוים או מבודד טוב בתנאי אחר.

אופן פעולת ה -LED :

- אם מתח האנודה קטן או שווה למתח הקתודה $V_A \le V_C$, ה- LED תהיה בקיטעון, היא תהווה נתק ולא יזרום בה זרם. במקרה זה היא תהיה כבויה.
- אם מתח האנודה גדול ממתח הקתודה $V_A > V_C$ אז ה- LED תהיה בהולכה, יזרום בה זרם מהאנודה לקתודה, במקרה זה היא תפלוט אור ותידלק. המתח בד"כ $V_{LED} = 1.6V$.
- IMIN - הוא הזרם המזערי שיכול לזרום ב- LED ועדיין תפעל בצורה תקינה, אם הזרם יהיה קטן מ IMIN ה-LED לא תידלק, ולכן לא נוכל לזהות ויזואלית שהזרם עובר דרכה. הערך שלו בד"כ הוא 10mA .
- IMAXהוא הזרם המרבי שיכול לזרום ב LED -ועדיין תפעל בצורה תקינה, אם הזרם יהיה גדול מ ,Imax-ה LED -תישרף. הערך שלו בד"כ הוא 20mA.
- תפקידו של הנגד R במעגל הוא להגבלת הזרם העובר דרך ה- LED. כזכור, כאשר הליד תהיה בהולכה, המתח הנופל עליה יהיה V1.6, אבל מוצא ה Arduino מספק מתח של V5, ולכן ההפרש בין מתח ה Arduino לבין מתח ה- LED, ייפול על הנגד ולא ילך לאדמה.

- ערכו של הנגד צריך להיות בתחום בין Rmax לבין Rmin.
- Rmax הוא הנגד הגדול ביותר שעדיין מבטיח מעבר זרם מינימאלי Imin ב- LED. אם ערכו של הנגד יהיה גדול מ Rmax , הזרם העובר ב- LED יהיה קטן מ- Imin , ולכן ה- LED לא תידלק.

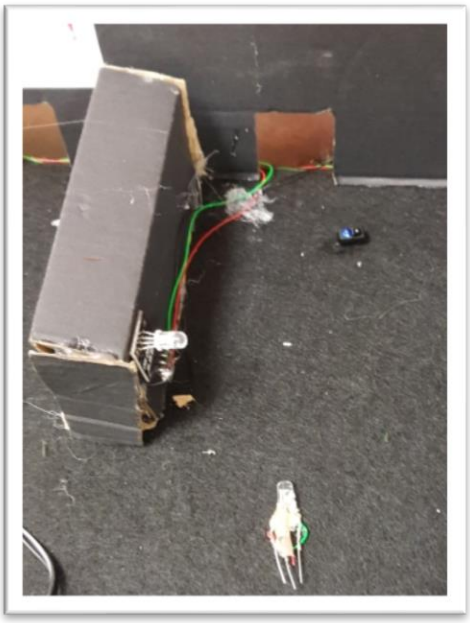
$$R_{\max} = \frac{U_R}{I_{min}} = \frac{D - V_{LED}}{I_{min}} = \frac{5-1.6}{0.01} = \frac{3.4}{0.01} = 340\Omega$$

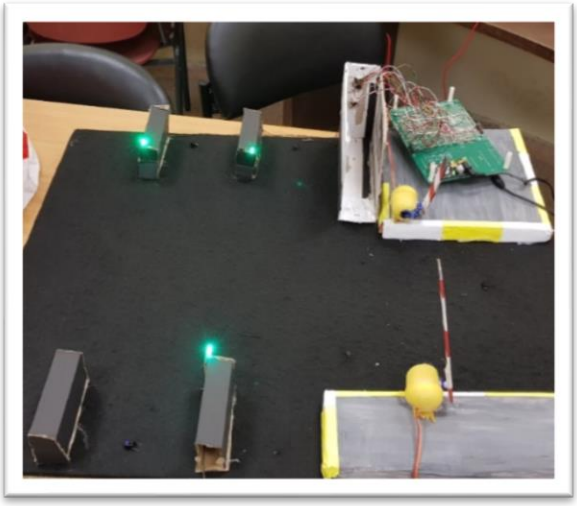
- Rmin הוא הנגד הקטן ביותר שעדיין מבטיח מעבר זרם מכסימאלי ב- LED. אם ערכו של הנגד יהיה קטן מ- Rmin, הזרם העובר ב- LED יהיה גדול מ Imax, ולכן ה- LED תישרף.

$$R_{\min} = \frac{U_R}{I_{\max}} = \frac{D - V_{LED}}{I_{\max}} = \frac{5-1.6}{0.02} = \frac{3.4}{0.02} = 170\Omega$$

- אפשר לראות שדרך הנגד אפשר לשלוט בעוצמת ההארה של ה- LED, ככל שערך הנגד קרוב יותר ל- Rmin הזרם יהיה גדול יותר, ולכן עוצמת ההארה תהיה חזקה יותר. וככל שערך הנגד קרוב יותר ל- Rmaxהזרם יהיה קטן יותר, ולכן עוצמת ההארה תהיה חלשה יותר.

הלחמת **LED 4**



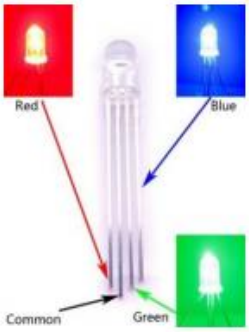


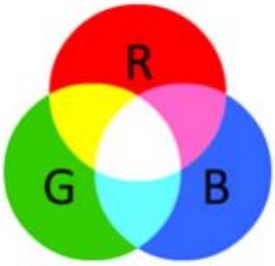
LED לא עובד !

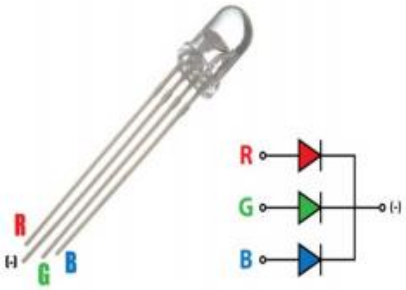
אחרי בדיקה הלד לא עובד , עשיתי בדיקת מתח ו מצאתי שמתח על הלד 0

RGB LED

ליד RGB הוא שילוב של 3 לידים בצבעים שונים (אדום, ירוק וכחול) בתוך רכיב אחד. הליד בעל 4 הדקים, בעזרתו ניתן להדליק אור אדום, ירוק או כחול, ניתן גם לשלב צבעים ביחד ולקבל כל צבע שאנו רוצים.

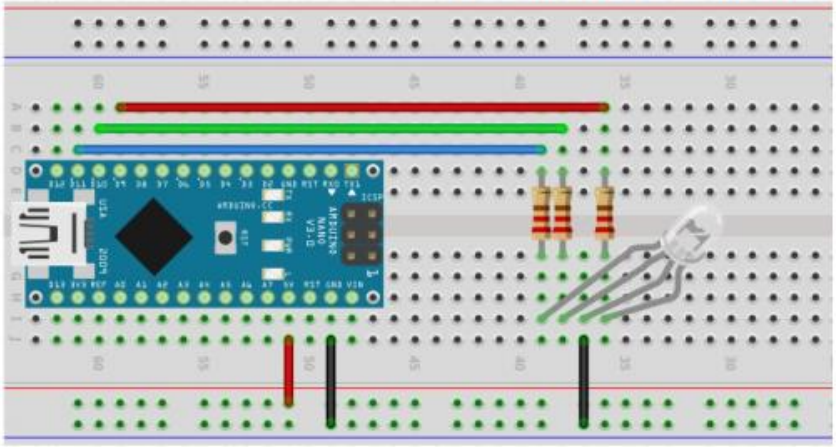






מיפוי הדקים שילוב צבעים מבנה ליד RGB

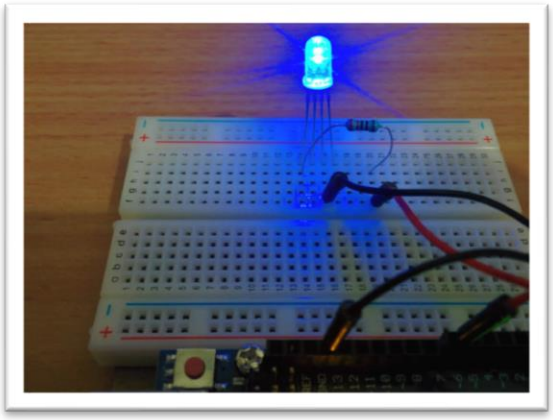
חיבור RGB LED והפעלתו

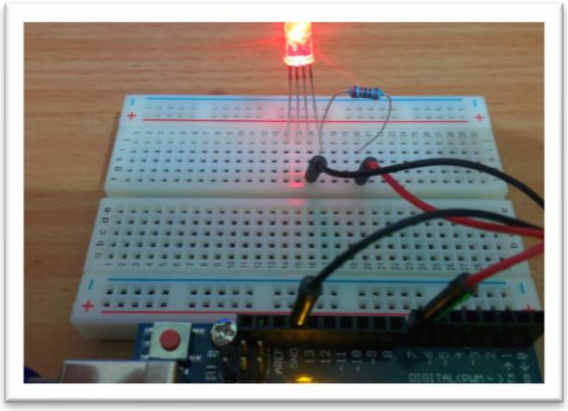


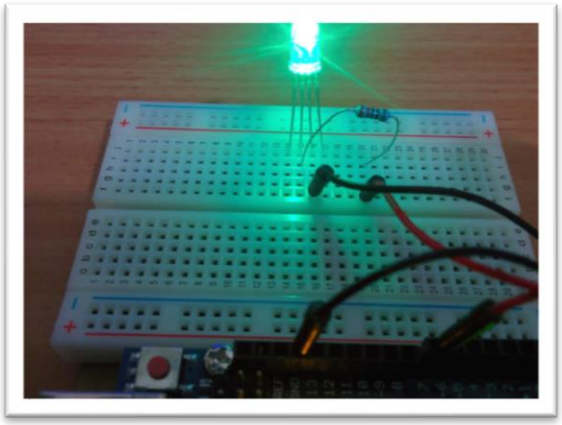
בחיבור RGB נתייחס לכל ליד כאילו הוא בנפרד, ולכל אחד נחבר נגד בטור מאותה סיבה של חיבור ליד רגיל, להלן תרשים חיבור של RGB LED לבקר Arduino Nano.

תכנית בדיקת כל הצבעים

```
void setup()
{
    pinMode(9,OUTPUT);
    pinMode(10,OUTPUT);
    pinMode(11,OUTPUT);
}

void loop()
{
    digitalWrite(9,1);
    digitalWrite(10,0);
    digitalWrite(11,0);
    delay(1000);
    digitalWrite(9,0);
    digitalWrite(10,1);
    digitalWrite(11,0);
    delay(1000);
    digitalWrite(9,0);
    digitalWrite(10,0);
    digitalWrite(11,1);
    delay(1000);
    digitalWrite(9,1);
    digitalWrite(10,1);
    digitalWrite(11,1);
    delay(1000);
}
```







מודל ספק כוח 3.3
Power Supply

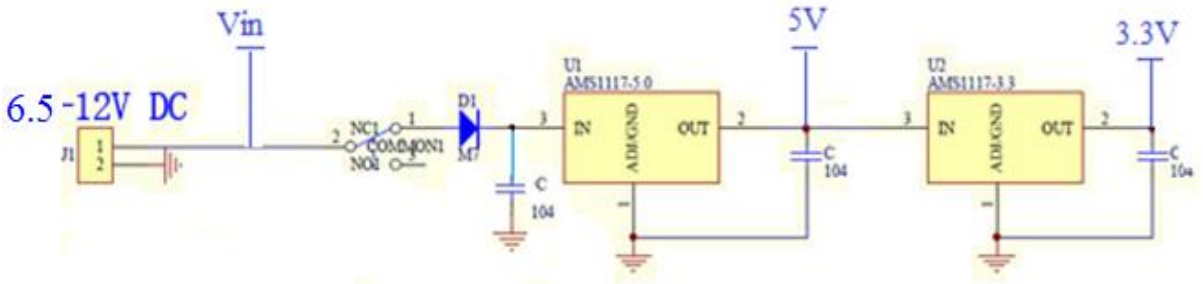
מודול ספק הכוח MB102 , מספק מתח כפול של 5Vאו 3.3Vבכל צד. ניתן לשלוט במתח היציאה דרך מגשרים (Jumpers) שנמצאים במודול, וניתן לקבל שתי יציאות קבועות של 5Vושתי יציאות קבועות של 3.3V.זרם המוצא המרבי שמתקבל הוא 700mA. ספק כפול הוא ספק שמספק שני מתחים במקביל. תפקיד דיודה בספק הוא הגנה מפני מתח שלילי , כי הדיודה מעברה זרם בכיוון אחד מאנודה לקתודה.

כל רכיב צריך מתח יציב ערכו 5 פולט.

תפקיד הקבלים בספק הוא לסינון רעשים והגנה מפני קפיצות מתח מתרחשת (באפס זמן). משמעות הדבר – תדר אין סופי - . היגב הקבל יהיה אפס , ויהווה קצר לאדמה , וכך כל הקפיצה תעבור לאדמה במקום שתעבור לרכיבים ותשרוף אותם.

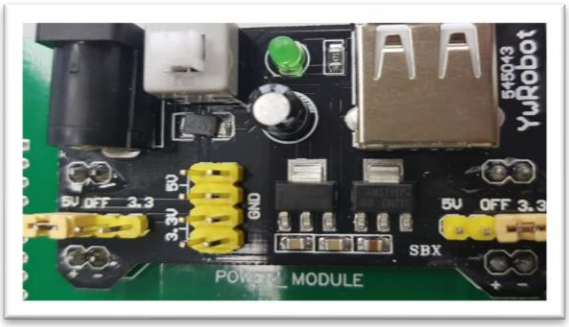
תפקיד דיודה בספק הוא הגנה מפני מתח שלילי , כי הדיודה מעברה זרם בכיוון אחד מאנודה לקתודה.

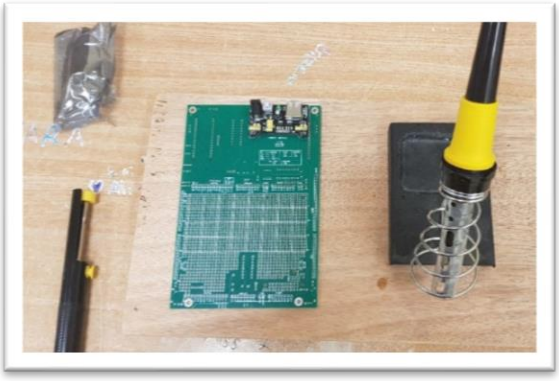
תרשים חשמלי של ספק הכוח



5/10/2018

POWER SUPPLY הלחמת





תצוגתLCD

השם LCD ,הוא קיצור של Display Crystal Liquid ובעברית תצוגת

גביש נוזלי, התצוגה אינה מקרינה אור כמו ב- LED או תצוגת 7 Segments אלא משתמשת באור הסביבה כדי למסור מידע.

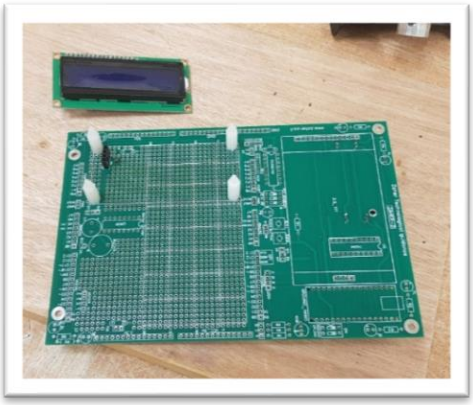
ה-LCDהיא תצוגה אלפאנומרית, שיכולים להציג בה אותיות, סימנם וספרות.

התצוגה בנויה משלושים ושתיים משבצות המסודרות בשש עשרה עמודות ושתי שורות, כל משבצת בנויה ממטריצת פיקסלים המסודרים בצורת 5 עמודות ו- 7 שורות, הארת הפיקסלים המתאימים גורמת להצגת צורת האות או הספרה המבוקשת על התצוגה.

תצוגה זו כוללת בדרך כלל 2 מעלים משולבים אשר מבקרים את פעולת הרכיב. וכוללים מעגלים היודעים להתחבר אל מיקרו בקר ולקבל ממנו פקודות ונתונים. כמו כן קיימים מעגלי זיכרון מסוג RAM ו ROMהיודעים להציג את התווים הנשלחים מהמיקרו במקום הרצוי בתצוגה.

23/11/2018

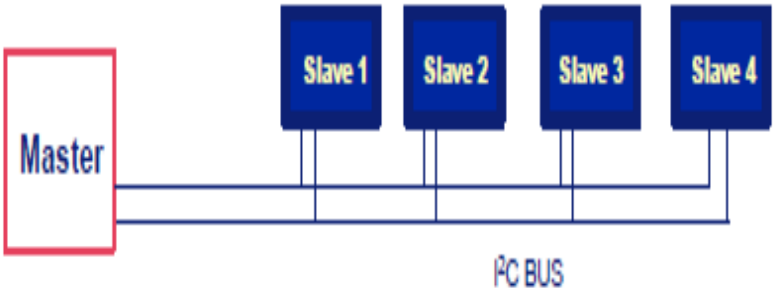
הלחמת **LCD**



תקשורת I2C

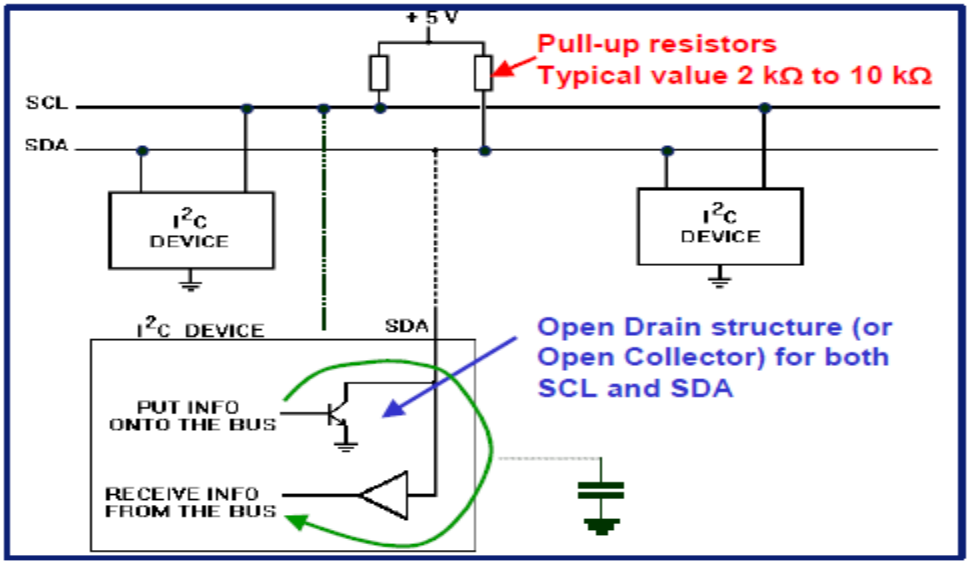
תקשורת I^2C היא תקשורת טורית בין מעבד- מסטר ורכיב עבד – SLAVE .
על פס תקשורת I^2C יכולים להתחבר מספר רכיבים שונים (זיכרונות , ממירים, שעוני זמן אמת וכו). הרכיב המנהל את תהליך התקשורת (המעבד) נקרא MASTER והרכיבים המתחברים אליו נקראים SLAVES . בתקשורת זו ישנם שני קווים. קו הנתונים הטורי - SDA - שהוא דו כיווני וקו השעון הטורי - SCLK שהוא חד כיווני ומופעל על ידי ה MASTER .
בנוסף, ה MASTER שולט על הגישה לפס ויוצר את מצבי ה START (התחלה) וה STOP (סיום).

איור 5 מתאר מספר רכיבים המתחברים על קו התקשורת I^2C



איור 1 א' - חיבור של מספר רכיבי SLAVE אל MASTER

באיור 1 א' ניתן לראות 4 רכיבי SLAVE המתחברים אל MASTER. באיור 1 ב' יש פרוט של נגדי ה PullUp וכיצד נראית דרגת היציאה והכניסה של רכיב המתחבר בתקשורת I^2C .



איור 1 ב' - קו תקשורת I^2C מפורט

ניתן לראות שעל 2 הקווים SDA (קו הנתון) ו SCL (קו השעון) יכולים להתחבר מספר רכיבים . לכל רכיב יש כתובת ייחודית משלו. לרכיב DS1307 הכתובת היא 1101000X (D0H או D1H). לרכיב קול של חברת WINBOND הנקרא ISD5116 הכתובת היא 80H וכך הלאה.

באיור רואים 2 רכיבים המתחברים על הקווים. בחלק התחתון של האיור רואים מבנה פנימי של רכיב ורואים שהרכיב מתחבר בעזרת חוצץ (מתואר על ידי המשולש) המקבל נתון מהקו. מעל החוצץ יש טרנזיסטור בחיבור קולט פתוח (Open Collector) או טרנזיסטור תופעת שדה - FET - בחיבור מפק פתוח (Open Drain) , שיכול לכתוב לקו נתון.
לטרנזיסטור יש לחבר נגד חיצוני שערכו נע בין 2 קילו אוהם ל 10 קילו אוהם. הערכים נבחרים כך שמצד אחד הנגדים לא יהיו קטנים מידי כדי שלא יזרום זרם גדול דרך הקווים ודרך הרכיב (במצב שהרכיב מוציא 0) ומצד שני שהנגד לא יהיה גדול מידי כי הוא קובע את זמן הטעינה והפריקה במעברים בין 0 ל 1 ולהפך ונגד גדול מידי יגביל את קצב התקשורת.

כללים והגדרות בתקשורת I^2C

- העברה יכולה להתחיל רק כאשר הקו לא עסוק - NOT BUSY .
- בזמן העברת נתון, קו הנתון חייב להישאר יציב כאשר קו השעון במצב גבוה. שינוי בקו הנתון כאשר קו השעון הוא גבוה יתפרש כאותות בקרה.

מגדירים את מצבי הפס הבאים :

Bus Not Busy - פס לא עסוק

גם קו הנתון וגם קו השעון בגבוה.

START DATA TRANSFER - התחל העברת נתון

שינוי במצב קו הנתון מגבוה לנמוך כאשר השעון נמצא בגבוה מוגדר כמצב START .

STOP DATA TRANSFER - עצור העברת נתון

שינוי במצב קו הנתון מנמוך לגבוה כאשר השעון במצב גבוה מוגדר כמצב STOP .

DATA VALID - תקפות נתון

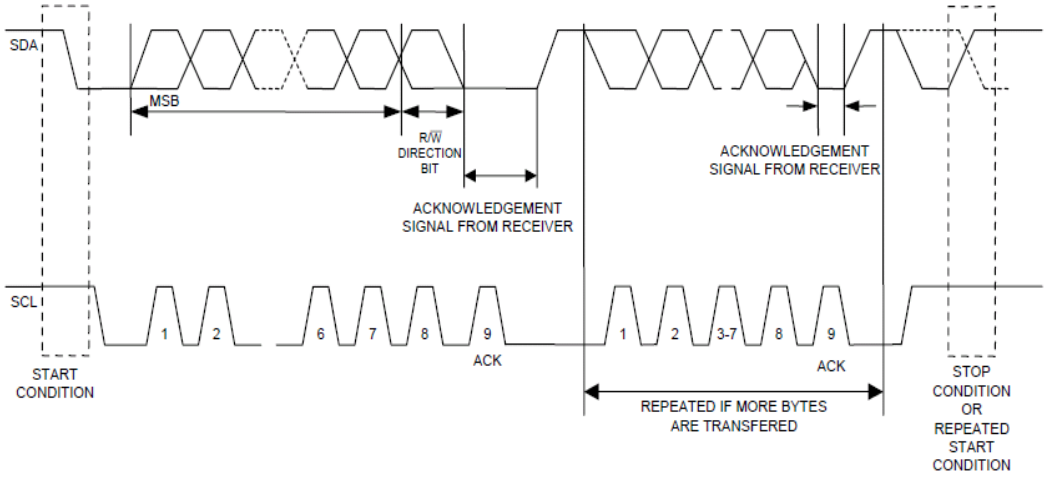
מצב קו הנתון מייצג תקפות נתון כאשר לאחר מצב START , קו הנתון יציב למשך זמן הגבוה של אות השעון. הנתון בקו חייב להשתנות רק בזמן מצב נמוך של אות השעון. יש פולס שעון אחד עבור כל ביט של נתון.

כל העברת נתונים מתחילה עם מצב START ומסתיימת עם מצב STOP . כמות הבתים המועברת בין START ל STOP לא מוגבלת ונקבעת על ידי רכיב ה MASTER . האינפורמציה מועברת ביית אחרי ביית וכל מקלט מאשר קבלת הבית עם ביט תשיעי של ACKNOWLEDGE .
בהגדרות של I^2C יש תקן של קצב ב 100KHz ויש תקן ל 400KHz .

ACKNOWLEDGE – אישור

כל רכיב קולט חייב בסיום קליטת ביית, שהועבר אליו, ליצור ביט ACKNOLEDGE . רכיב הMASTER יוצר פולס שעון נוסף הקשור לביט זה.
רכיב היוצר ACKNOLEDGE חייב להוריד את קו הנתון הטורי – SDA – ל 0 בזמן פולס השעון, כלומר שקו הנתון יהיה יציב בנמוך בזמן שקו השעון בגבוה. רכיב ה MASTER מסמן ל SLAVE על סיום התקשורת על ידי **אי יצירת** ביט ה ACKNOLEDGE כאשר הוא קלט את הביית האחרון מה SLAVE . במקרה כזה על ה SLAVE להשאיר את קו הנתון בגבוה כדי לאפשר ל MASTER ליצור מצב STOP .

באיור 2 ניתן לראות העברה של נתון טורי.



איור 2 - העברת נתון בקו תקשורת טורית I^2C

את הקו SCL (הקו התחתון בשרטוט) יוצר תמיד ה MASTER. כדאי לשים לב שמצב START קורה כאשר קו SCL בגבוה ואז ה MASTER הוריד את קו הנתון ל 0 . לאחר מכן ה MASTER יוצר 8 פולסי שעון ואז הוא שולח בקו הנתון - SDA - 8 ביטים . 7 ביטים הם כתובת הרכיב והביט ה 8 אומר האם הוא רוצה לכתוב אל הרכיב או לקרא ממנו (0 - כתיבה, 2 – קריאה). לאחר מכן ה MASTER יוצר פולס 9 נוסף שבו ה SLAVE צריך להחזיר ACKNOWLEDGE . לאחר מכן אין צורך ב START נוסף והבתים נשלחים אחד אחרי השני כאשר הצד הקולט נותן ACKNOWLEDGE בביט מספר 9 . מצב STOP (או START חוזר) מתואר בצד ימין של איור 6 . הוא נוצר כאשר קו השעון ב 1 ואז בקו הנתון יש מעבר מ 0 ל 1 . מצב START חוזר משורטט בקו מקווקו ובו רואים שבזמן שקו השעון ב 1 יורד קו הנתון ל 0 .

העברת נתון בתקשורת I^2C

שתי אפשרויות העברת נתונים קיימות בקו תקשורת I^2C :

א. ה MASTER משדר וה SLAVE קולט - אופן כתיבה - Write Mode

במקרה זה הביית הראשון המשודר על ידי ה MASTER הוא הכתובת של ה SLAVE (במקרה של רכיב DS1307 הכתובת היא 1101000X - D0H במקרה של כתיבה לרכיב או D1H אם קוראים מהרכיב) . לאחר מכן יבואו מספר בתים של נתונים. ה SLAVE מחזיר ACKNOLEDGE בסיום כל ביית נתונים שקלט. הנתון מועבר עם ביט ה MSB ראשון !!

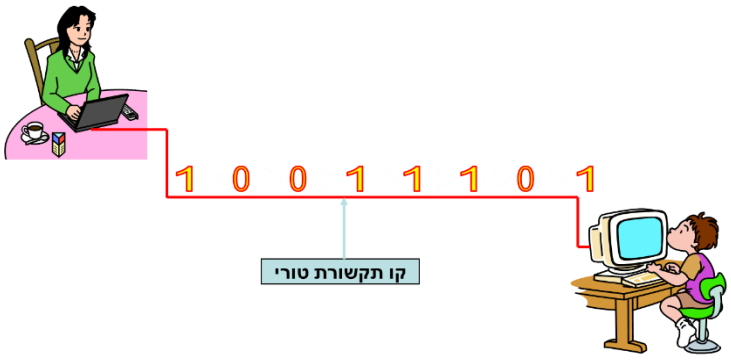
ב. ביית משודר מה SLAVE אל ה MASTER - אופן קריאה - Read Mode

במקרה זה הביית הראשון שנשלח הוא על ידי ה MASTER השולח את כתובת ה SLAVE שמחזיר מצידו את ביט ה ACKNOWLEDGE . מכאן ה SLAVE שולח מספר בתי נתונים. ה MASTER מחזיר ביט ACKNOLEDGE אחרי כל קליטת ביית נתון חוץ מהביית האחרון שהוא איננו מחזיר ACNOWLEDGE או אפשר להגיד שהוא מחזיר Not ACKNOWLEDGE .

תקשורת טורית (סיריאלית)

- הגדרה: תקשרות סריאלית, בין שני משתמשי קצה אשר המידע בינהם עובר באופן טורי.
  סריאלית = סדרתי/טורי
- בתקשורת סריאלית אנו מעבירים מידע באופן סדרתי, כלומר המידע עובר ביט אחר ביט (כאשר רק סיבית אחת תעבור ברגע מסוים).
- ההפך מתקשורת סיריאלית היא תקשורת מקבילית.
- בתקשורת מקבילית מספר סיביות יעברו בבת אחת ממחשב למחשב.

לדוגמא: מעוניינים להעביר מילה 10011101 בתקשורת סריאלית (טורית) בין שני מחשבים.



תקשורת סריאלית, יתרונות:

- חסכוני יותר בכבלים (= חסכוני בכסף)
- מאפשר העברת נתונים למרחקים גדולים יותר
- העברה ביט באופן טורי יותר קלה לפיענוח בצד המקבל.
- אם נעביר באופן אחר (למשל שימוש בשני קווי תקשורת) עלול להיווצר בעיה בתקשורת עקב הגעה לא נכונה של הביטים שנשלחו (אין הבטחה שהמהירות בקו אחד זהה למהירות בקו השני)
- תקשורת סריאלית, חסרונות:
- מאט את קצב העברת הנתונים
- דורש מעגלי המרה ממקבילי לטורי ולהפך

מנוע סרוו

מנוע סרוו הוא מנוע זרם ישר (DC Motor) בעל מערכת תמסורת פנימית של גלגלי שיניים ובקרה אלקטרונית על מיקום המנוע. מה שמיחד מנועי סרוו היא העובדה שהם אינם מסתובבים בצורה חופשית כמו מנועיDC, אלא נעים על פי זווית – לרוב בין 0 ל-180 מעלות.

מנועי סרוו פועלים בחוג סגור, כלומר הינם בעלי בקרה על מיקום המנוע, ובעליי כולת תיקון פערים מהמיקום הרצוי.

שימושים שונים למנועי סרוו ברובוטיקה:

מנועי סרוו נמצאים בשימוש בסוגים רבים מאד שלרובוטים ובני הם זרועות רובוטיות, מכוניות הנשלטות בשלט רחוק, רובוטי-רכב, מטוסים ומסוקים (לשליטה על זווית הכנף \ רוטור). ישנן סיבות רבות לכך שמנועי סרוו נפוצים כל כך באפליקציות רובוטיקה, ובינהן קלותה שליטה במנועי סרוו, דרישות האנרגייה הנמוכות (יעילות), הכחה גבוה, רמת מתח TTL, והגודל והמשקל הנמוכים.

יתרונות וחסרונות של שימוש במנועי סרוו:

מנועי סרוו שימושיים מאד עבור אפליקציות רובוטיקה, בשל סיבות רבות:

לרוב מנועי סרוו הינם מנועים בעלי גודל פיזי קטן

מנועי סרוו מספקים כח זוויתי (מומנט) חזק מאד בהשוואה לגודלם

מנועי סרוו פועלים בחוג סגור ולכן נחשבים אמינים מאד

למנועי סרוו יש מעגל שליטה ובקרה פנימי

מנועי סרו צורכים זרם בצורה פרופורציונאלית למטען אותו הם נושאים (לכן סרוו שאינו נושא מטען רב לא יצרוך הרבה זרם)

מנועי סרוו פועלים במתח נמוך יחסית (כ-4 עד 6 וולט).

כיצד שולטים במנוע סרוו?

שליטה במנועי סרוו מבוצעת על ידי שליחת אות דיגיטאלי אל חוט הבקרה של המנוע. הרעיון הכללי הוא שליחת גל מרובע (Square Wave) אל המנוע, כאשר אורך הגל הוא זה שקובע את הזווית אליה ינוע המנוע.

לדוגמה, כאשר נספק למנוע גל בו רוחב הפולס הוא 1מילי-שנייה, המנוע ינוע אל זווית והמינימאלית – 0 מעלות. כאשר נספק למנוע גל בו רוחב הפולס הוא 1.5 מילי-שנייה, המנוע ינוע אל זווית והאמצעית – 90 מעלות.

כאשר נספק למנוע גל בו רוחב הפולס הוא 2 מילי-שנייה, המנוע ינוע אל זווית והגדולה ביותר – 180 מעלות.

מתח מנועי סרוו:

מנועי סרוו עובדים בטווח שונה שלמתחים, ובדרך כלל בין 4.8 ל-6 וולט. הסיבה לשימוש בסטנדרט הזה היא הקרבה לרמת TTL (שהיא 5 וולט) שבה פועלים רוב המקרו-מעבדים שמשמים לשליטה על מנועי הסרוו.

אם כן, באיזה מתח מומלץ להשתמש? שימוש במתח המירבי איתו מסוגל המנוע לעבוד יניב את הכח החזק ביותר.

חיווט מנועי סרוו:

לכל מנועי הסרוו יש שלושה חוטים:

חוט חום שהוא אדמה (-)

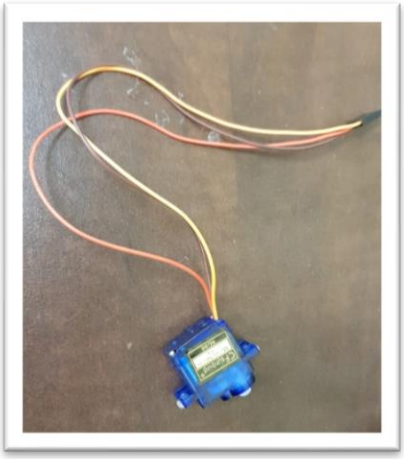
חוט אדום שהוא המתח (+)

חוט כתום, שהוא חוט האות לשליטה במנוע.

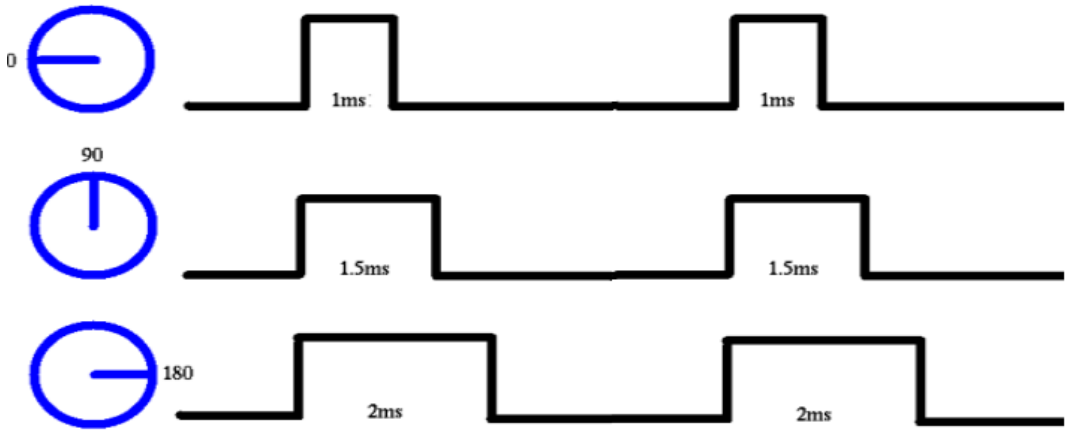
כיצד שולטים במנוע סרוו?

שליטה במנועי סרוו מבוצעת על ידי שליחת אות דיגיטאלי אל חוט הבקרה של המנוע. הרעיון הכללי הוא שליחת גל מרובע (Square Wave) אל המנוע, כאשר **אורך הגל** הוא זה שקובע את הזווית אליה ינוע המנוע.

לדוגמה, כאשר נספק למנוע גל בו רוחב הפולס הוא 1מילי-שנייה, המנוע ינוע אל זוויתו המינימאלית – 0 מעלות.
כאשר נספק למנוע גל בו רוחב הפולס הוא 1.5 מילי-שנייה, המנוע ינוע אל זוויתו האמצעית – 90 מעלות.
כאשר נספק למנוע גל בו רוחב הפולס הוא 2 מילי-שנייה, המנוע ינוע אל זוויתו הגדולה ביותר – 180 מעלות.

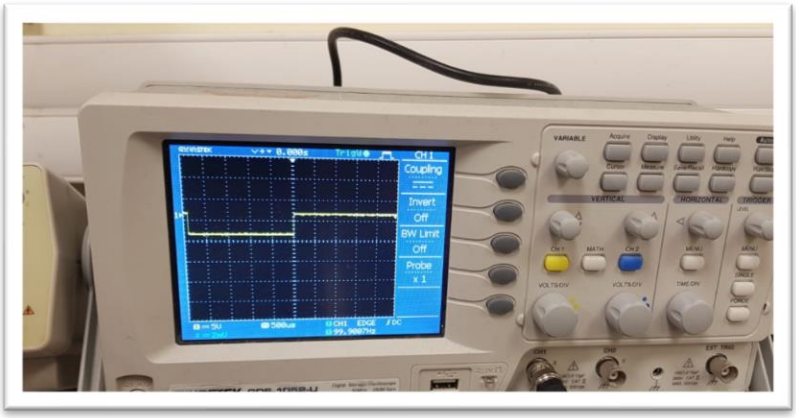


תרשים סכמטי:

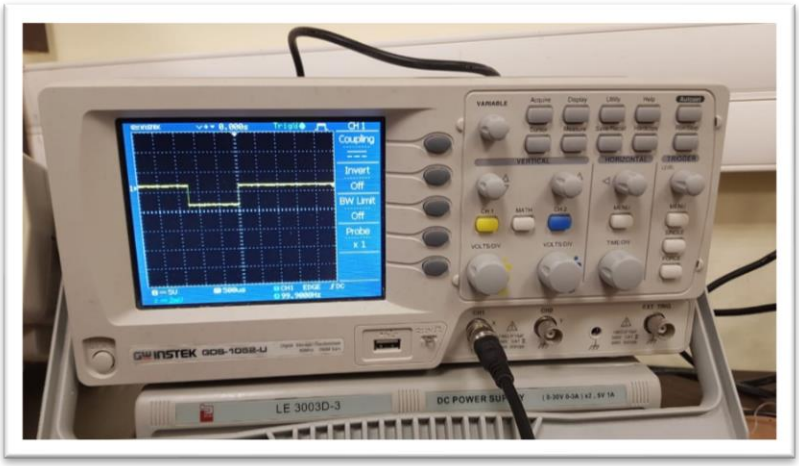


בדקנו אותה במעבדה!

כאשר הזווית 180



כאשר הזווית 90



חיווט מנועי סרוו

לכל מנועי הסרוו יש שלושה חוטים:

חוט שחור או חום שהוא אדמה (-)

חוט אדום שהוא המתח (+)

חוט צהוב, כתום או לבן שהוא חוט האות לשליטה במנוע.

לדוגמה במנועי הסרוו הנפוצים של Hitec \ Futaba :

Red → VCC

Brown → GND

Orange → Arduino (D)

תכנית בדיקת סרוו

```
#include <Servo.h>

Servo servo1;
Servo servo2;
void setup() {
  servo1.attach(10);
   servo2.attach(9);
    servo1.write(100);
       servo2.write(95);
    delay(3000);
}

void loop()
{



   servo1.write(180);
   servo2.write(25);
    delay(1000);

}
```

תכנית בדיקת סרוו וחיישן IR

```
#include <Servo.h>

Servo myservo;
 int counter=1;
void setup() {
  myservo.attach(9);
pinMode(A1,INPUT);
pinMode(A2,INPUT);
pinMode(2,OUTPUT);
pinMode(3,OUTPUT);
  Serial.begin(9600);
}

void loop() {


  digitalWrite(2, 0);

     if(counter<=4)
     {
      digitalWrite(2, 1);
       digitalWrite(3, 0);

  float sensor1 = analogRead(A1);
  if( sensor1<1020)
  {

  Serial.println(counter);
   myservo.write(180);
        delay(3000);
     myservo.write(0);
       counter++;
        digitalWrite(3, 1);

}
     }
 }
```

קוד סופי

```
#include <Servo.h>
 #include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2);
Servo servo1;
Servo servo2;
 int counter=0;
void setup() {
  lcd.init();
  lcd.backlight();
  lcd.clear();
   lcd.setCursor(3,0);
  lcd.print("welcome to ");
     lcd.setCursor(3,1);
  lcd.print("my parking");

pinMode(12,OUTPUT);
pinMode(13,OUTPUT);
pinMode(2,OUTPUT);
pinMode(3,OUTPUT);
pinMode(4,OUTI
pinMode(5,OUTI
pinMode(A3,INPUT);
pinMode(A7,INPUT);
pinMode(A2,INPUT);
pinMode(A1,INPUT);
pinMode(A0,INPUT);
digitalWrite(4,0);
 digitalWrite(5,1);
 digitalWrite(2,1);
 digitalWrite(3,0);
  digitalWrite(12,1);
 digitalWrite(13,0);
  Serial.begin(9600);
   servo1.attach(10);
   servo2.attach(9);
    servo1.write(40);
    servo2.write(70);
    delay(3000);
}
```

```
void loop() {

 float sensorin=analogRead(A0);
 float sensorout=analogRead(A1);
 float sensor2 = analogRead(A2);
 float sensor3 = analogRead(A3);
 float sensor7 = analogRead(A7);
 Serial.println(counter);
 lcd.clear();
   lcd.setCursor(2,0);
  lcd.print("empty places");
    lcd.setCursor(7,1);
  lcd.print(3-counter);
if(sensorout<700)
  {servo2.write(20);
   servo1.write(90);
   lcd.clear();
    lcd.setCursor(2,0);
    lcd.print("empty places");
    lcd.setCursor(7,1);
  lcd.print(counter-1);
    counter--;


    delay(1000);
    }
if(counter<3)
{
  if(sensorin<700)
  {
    lcd.clear();
    lcd.setCursor(2,0);
    lcd.print("empty places");
    lcd.setCursor(7,1);
  lcd.print(3-counter);


    counter++;
    delay(1000);
    }
```

```
if(sensorin<700||sensorout<700)
{
  servo2.write(20);
   servo1.write(90);
   delay(3000);
   servo2.write(70);
   servo1.write(40);

}
}
else


 if( sensor2<700)
 {

   digitalWrite(12,0);
      digitalWrite(13,1);
 }

 if( sensor2>700)
 {

     digitalWrite(12,1);
        digitalWrite(13,0);
 }

  if( sensor7<700)
 {

   digitalWrite(2,0);
      digitalWrite(3,1);
 }

 if( sensor7>700)
 {

     digitalWrite(2,1);
        digitalWrite(3,0);


 }
  if( sensor3<700)
 {

   digitalWrite(4,1);
      digitalWrite(5,0);
 }
```

```
if( sensor3>700)
{
    digitalWrite(4,0);
       digitalWrite(5,1);

}

else servo1.write(40);
   servo2.write(70);


}
```