תקשורת טורית UART

הקדמה:

תקשורת טורית (UART) היא שיטה נפוצה ופשוטה להעברת מידע בצורה טורית (סידורית) בין שני התקנים דיגיטליים, כגון בין מיקרו-בקר לבין מחשב או מודם. זהו אחד מפרוטוקולי התקשורת הבסיסיים ביותר הנמצאים בשימוש בתחום האלקטרוניקה הדיגיטלית.

בבסיסה, תקשורת UART מורכבת וכוללת פרוטוקולים המאפשרים שידור וקליטה של מידע בצורה אמינה ויעילה. היא משתמשת בשני חוטי תקשורת עיקריים: "קבל (RX) " ו- "שלח (TX).

מהי תקשורת UART?

תקשורת טורית (UART) היא שיטה פשוטה ונוחה להעברת מידע בין שני התקנים דיגיטליים, למשל בין מיקרו-בקר כמו ארדואינו למחשב.

בתקשורת UART המידע מועבר בצורת **סיביות** (זרמים מתחלפים גבוהות ונמוכות - ראו תמונה בהמשך) ומועברות בחוטי תקשורת נפרדים בין שני המכשירים. כל מכשיר מחובר עם שני פינים לתקשורת - **קלט (RX) ופלט (TX) .**

כדי לשדר מידע, המכשיר (השולח) שולח את הביטים על גבי קו ה - TX. בצד השני, המקבל מאזין לאותו קו ומזהה את המידע בכניסת ה- RX. שלו ומפענח אותו בחזרה למידע דיגיטלי.

עקרונות חשובים ב UART-הם **קצב התקשורת** - baud rate)כמה ביטים בשנייה, (**אורך קבוע** -8) 5ביטים בשנייה, (**ביט התחלה** (סינכרון לתחילת העברת מידע), ו**ביט סיום** (סימון סיום ההעברה).

היתרונות הגדולים של תקשורת טורית הם פשטות ומינימליות בחומרת המכשיר, מה שמאפשר להשתמש בה ליישומים רבים עם מגבלות תוכנה וחומרה.

לכן UART היא אחת השיטות הנפוצות להעברת נתונים בין מיקרו-בקרים לבין מחשבים במגוון יישומים כמו התקני אינטרנט של הדברים, מערכות בקרה תעשייתית, רובוטיקה ועוד.

לסיכום :תקשורת טורית מאפשרת לנו לשדר ולקלוט נתונים באופן אמין ופשוט יחסית בין כל סוגי ההתקנים הדיגיטליים.

מאפיינים תקשורת טורית:

- **תקשורת סינכרונית** (סיראלית) - המידע מועבר כביט, ביט אחרי ביט, על גבי קו תקשורת יחיד.
- **א-סינכרונית** - אין שעון משותף, כל התקן פועל לפי שעונו הפנימי.
- **תקשורת דו-כיוונית** - יכולה לשדר ולקלוט נתונים.
- חד כיווני (Simplex)- שידור רק בכיוון אחד .
- חצי דו כיווני (Duplex-Half)- בכל רגע נתון, רק צד אחד משדר והשני קולט ולהפך, זהו האופן הנפוץ ביותר.
- **שידור וקליטה בקצב זהה** (קבוע מראש לדוגמה 9,600 ביט לשנייה).
- **מבנה מוגדר היטב** של מסגרת הנתונים (פריימים) - סיביות התחלה, נתונים, סיביות עצירה.
- **דרישות חומרה פשוטות** - בדרך כלל 2 קווים בלבד כיוון (TX ו-RX).
- **ממשק חומרה פשוט וסטנדרטי** (למשל UART 16550 במחשבים).

- **שימוש נרחב** במערכות מובנות (מחשב) ובתקשורת בין מכשירים.
- **פרוטוקולי חיבור** התקני קצה כמו מודמים, מדפסות, מודמים ועוד.
- **פרוטוקולים נפוצים** הבנויים מעל RS-232, RS-422, RS-485 : UART.

מבנה חבילת נתונים בתקשורת טורית:

כאשר מתקן (למשל מיקרו-בקר) מעוניין לשדר נתונים, הוא בונה חבילת נתונים במבנה מוגדר הכולל מספר שדות:

- **ביט התחלה** (Start Bit) - ירידה מ-'1' ל-'0' לוגי, המסמן את תחילת השידור.
- **נתונים** - המידע עצמו במסגרת של ביטים (בדרך כלל 5-8 ביטים).
- **ביט זוגיות** (Parity Bit) - אופציונלי לבדיקת שגיאות.
- **ביט עצירה** (Stop Bit) 1 לוגי, המסמן את סוף השידור.

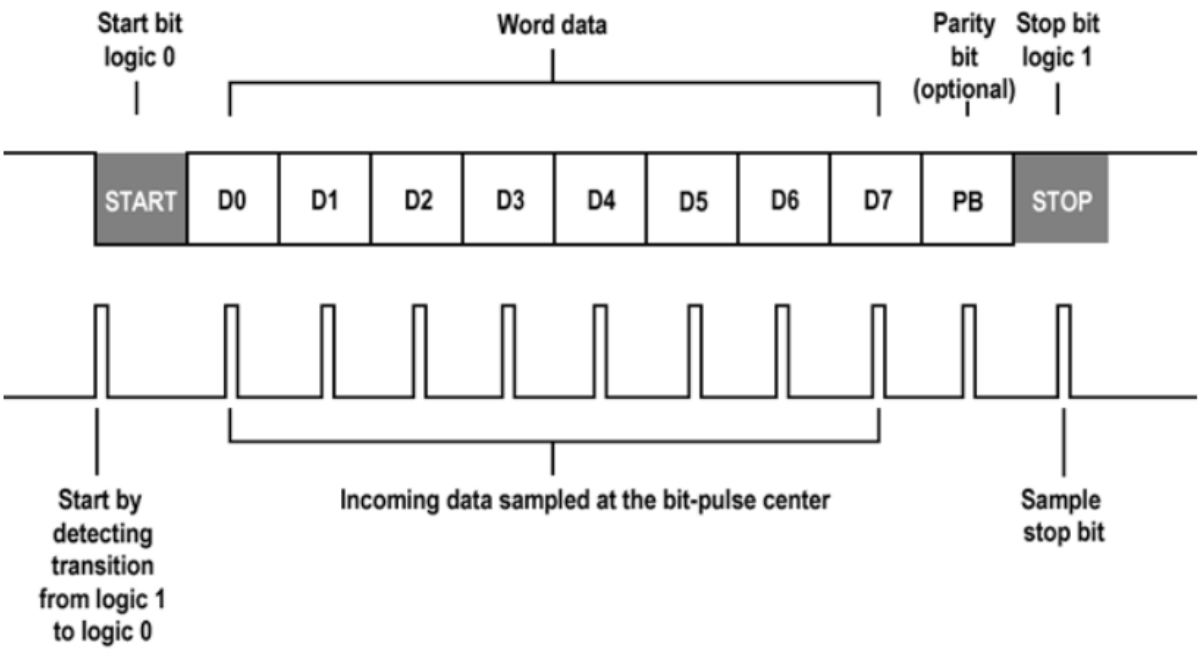
דוגמה:

במידה ונרצה לשלוח את התו 0x3F (המספר 63 בעשרוני) ייבנה מסר באורך 8 ביטים:

Start Bit (0) → 00111111 → **Stop Bit(1)**

כלומר, **ביט התחלה אחד (0)**, אחריו **(מימין לשמאל) הביטים של הנתון (00111111)**, ולבסוף **ביט עצירה אחד (1)**.

בין כל שידור של חבילה כזו יש מרווח זמן קצר ללא שידור, כך שהצד המקבל יוכל לזהות היטב את ביט ההתחלה הבא ובכך את תחילת הנתונים החדשים.



Start bit logic 0: ביט התחלה.

Parity bit (optional): ביט זוגיות (אופציונלי).

Word Data: המעיד שאנו שולחים.

Stop bit logic 1: ביט סיום.

Start by detecting transition from logic 1 to logic 0: תחילת שידור – מעבר מהמתח גבוה (1 לוגי) למתח הנמוך (0 לוגי) שמסמן את תחילת העברת המידע.

Incoming Data sampled at the bit – pulse center: נתונים מגיעים בפולסים של ביטים.

Sample stop bit: סיום שידור.

ביט זוגיות (Parity Bit):

ביט הזוגיות משמש לזיהוי שגיאות בסיסי בתקשורת טורית. הוא מוסיף ביט נוסף למסגרת הנתונים, כך שמספר הביטים עם ערך 1 יהיה תמיד זוגי (זוגיות זוגית) או אי-זוגי (זוגיות אי-זוגית). אם בקליטת המידע מספר הביטים עם ערך 1 לא מתאים לסוג הזוגיות שנקבע, המקלט יכול להניח שאירעה שגיאה בשידור. עם זאת, ביט זוגיות לא מאפשר לתקן שגיאות או לזהות שגיאות מרובות ביטים, ולכן הוא שיטה בסיסית בלבד לווידוא שלמות הנתונים.

תקשורת טורית ב-ESP32:

ה-ESP32-בניגוד לארדואינו אונו, מצויד במספר יחידות תקשורת טורית (UART). בדרך כלל, יש לו שלושה ממשקי UART, מה שהופך אותו למתאים במיוחד לפרויקטים הדורשים חיבור למספר התקנים חיצוניים כמו מודולי GPS, חיישני טמפרטורה, מסכי LCD ועוד.

הבדלים עיקריים בין תקשורת טורית בארדואינו ל-ESP32:

- **מספר יחידות UART: בעוד שלארדואינו אונו יש רק יחידת UART אחת (בסיכות 0 ו-1), ל- ESP32 יש שלושה ממשקי UART: (UART0, UART1, UART2) זה מאפשר לפתח פרויקטים מורכבים יותר המשתמשים במספר רב של מכשירים הפועלים בשיטה טורית.**
- **חופש בבחירת הפינים : ה- ESP32 מאפשר למתכנת לבחור באופן חופשי את הפינים (GPIO) שישמשו לכל יחידת UART, בזכות יכולת מולטיפקסינג הפינים. המשמעות היא שניתן להגדיר את פיני ה- TX ו- RX של כל UART לכל פין GPIO פנוי בלוח, דבר המעניק גמישות רבה בתכנון החומרה.**
- **פונקציונליות :כל יחידת UART ב- ESP32 היא עצמאית לחלוטין וכוללת מאגרים (buffers) גדולים לקליטה ושידור. זה מבטיח שהמידע לא יאבד גם כאשר יש עומס גדול של נתונים, ומאפשר שימוש ב- UART0 (שמחובר גם ל - USB) לצורך ניפוי באגים (debug) בתוכנה, תוך כדי שימוש ב- UART1 ו- UART2 לצורך תקשורת עם רכיבים אחרים.**

קצב שליחת נתונים:

אנו מגדירים את קצב שליחת הנתונים בתקשורת טורית בפקודה אחת בלבד בפונקציה void Setup באמצעות הפקודה:

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

רשימת קצבי השידור הנפוצים בתקשורת טורית (UART):

- 300 ביט לשנייה
- 600 ביט לשנייה
- 1,200 ביט לשנייה
- 2,400 ביט לשנייה
- 4,800 ביט לשנייה
- 9,600 ביט לשנייה
- 19,200 ביט לשנייה
- 38,400 ביט לשנייה
- 57,600 ביט לשנייה
- 115,200 ביט לשנייה
- 230,400 ביט לשנייה
- 460,800 ביט לשנייה
- 921,600 ביט לשנייה

ניתן לראות שיש מגוון רחב של קצבי שידור, החל מ-300 ביט לשנייה במערכות ישנות יותר, ועד למיליוני ביטים לשנייה בממשקים טוריים מהירים יותר.

הקצבים הנפוצים ביותר הם: 9,600 ביט לשנייה ו-115,200 ביט לשנייה.

קצב השידור נקבע מראש ומחייב להיות זהה בין שני ההתקנים המתקשרים כדי לאפשר קליטה נכונה של הנתונים.

רכיבים שעובדים בתקשורת טורית:

- **מודולי בלוטוס** - HC-05, HC-06
- **מודולי WiFi** - ESP8266, ESP32
- **מודולי LoRaWAN/LoRa** - לתקשורת לטווח רחוק
- **מודולי תקשורת סלולרית** - SIM900, SIM800L
- **מודולי GPS** - NEO-6M, UBLOX
- **צגי OLED** ו-LCD טוריים (I2C)
- **חיישני טמפרטורה, לחות, אור ותנועה**
- **מודולי קוראי RFID/NFC**
- **מודולי מצלמות טוריות**
- **כרטיסי SD וקוראי כרטיסי SD**
- **מדפסות ומפענחים טוריים**

שליחת נתונים בתקשורת טורית:

ניתן לשלוח נתונים בתקשורת טורית באמצעות הפקודה:

Serial.print();

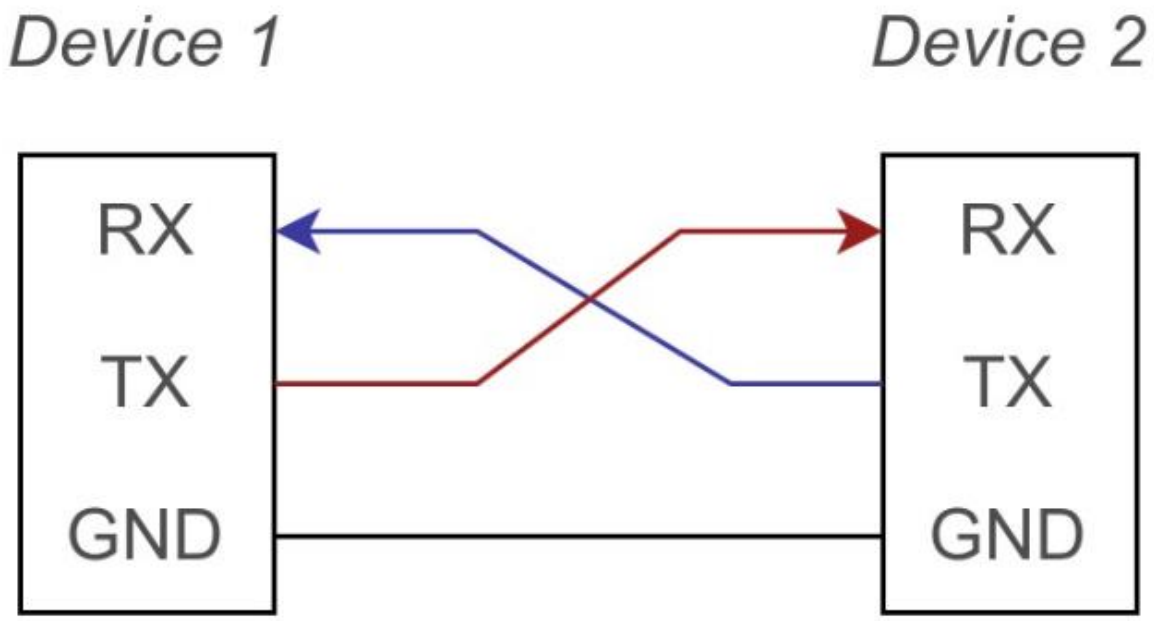
כל מה שנרשום בין הסוגריים () עם מרכאות ישלח בתור מחרוזת, ללא מרכאות ישלח בתור ערך של משתנה. בפקודה זו **אין ירידת שורה**. במידה ונרצה לרדת שורה אנו צריכים להשתמש בפקודה:

Serial.println();

חיבור בין 2 רכיבים בתקשורת טורית:

בחיבור בין שני התקני UART נדרשים לפחות 3 חוטים - TX של צד אחד מתחבר ל-RX של הצד השני, וחוט הארקה משותף (GND).

לעתים נדרש גם חיבור של מתח ההזנה (Vcc) אם אחד ההתקנים מספק כוח לַשני. יש לוודא התאמה של רמות המתח בין שני הצדדים (V3.3 או V5).



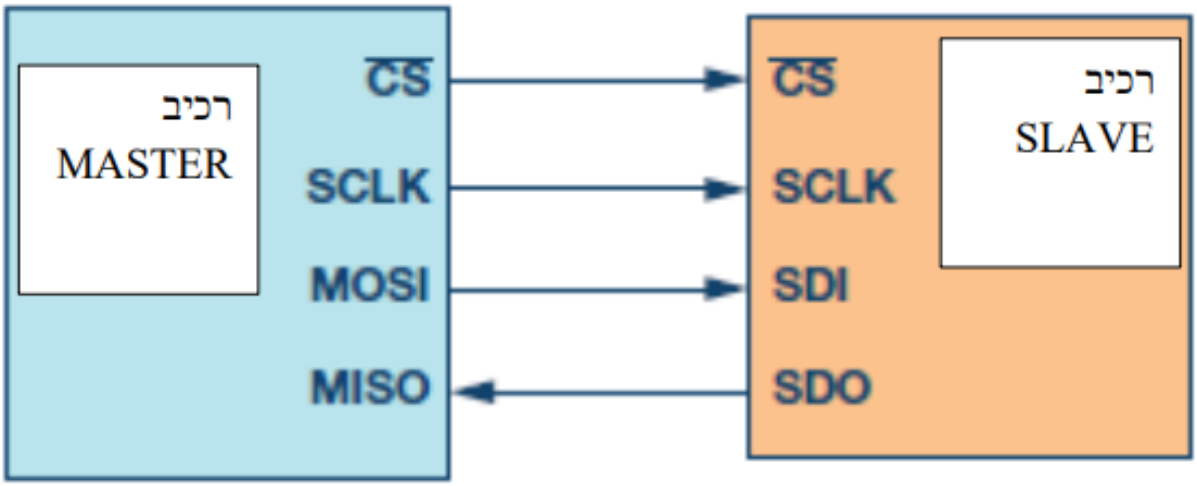
תקשורת SPI

התקשורת נוצרה על ידי חברת מוטורולה ב 1979 עם יציאת המיקרו פרוסיסור של חברת מוטורולה שנקרא 68000 .

השם SPI הוא - Interface Peripheral Serial – ממשק טורי היקפי . זוהי תקשורת טורית סינכרונית כי יש בה שעון שמסנכרן את הכנסת/ הוצאת הנתונים.

דוגמאות לרכיבי SPI הם מתגים, זיכרונות, רכיבי שמע להקלטה/השמעה , ממירים למיניהם (ADC), תצוגות לדים, TFT ועוד.

באיור הבא מתואר חיבור של תקשורת טורית SPI בין רכיב MASTER ורכיב SLAVE.



באיור רואים שבתקשורת SPI יש 4 קווים:

1. Master out slave in – MOSI : קו הנתון מהמסטר (המיקרו בקר) אל העבד (ברכיב העבד השם הוא SDI – Serial Data In).
2. Master in slave out – MISO : קו הנתון הטורי מהעבד אל המסטר. ברכיב העבד השם הוא SDO – Serial Data Out).
3. Serial clock – SCLK : שעון טורי. שני האותות MOSI ו- MISO מסונכרנים בעזרת קו השעון הטורי.
4. Slave select : זהו קו נוסף שדרכו אנו בוחרים את העבד ה- slave, בעזרת קו זה המיקרו בקר מודיע לאיזה עבד הוא פונה. הקו פעיל בנמוך – LOW ACTIVE. שמות נוספים לקו הם – CS בחירת רכיב – Enable, אפשור ועוד.

תהליך התקשורת מתבצעת בשלבים הבאים:

- כאשר ה-MASTER מתחיל תקושרת עם ה-SLAVE הוא מוריד את קו בחירת הרכיב עבד (Select slave) ל-0.
- ה- MASTER שולח בקו ה- MOSI ביט אחרי ביט, כאשר כל ביט מסונכרן בעזרת פולס שעון – SCLK- שמייצר ה- MASTER לתוך ה-SLAVE.
- הביטים הנשלחים נמצאים ברגיסטר הזזה הנמצא בתוך ה-SLAVE.
- הסנכרון לתוך ה-SLAVE יכול להיות בעליית פולס שעון או בירידה שלו.

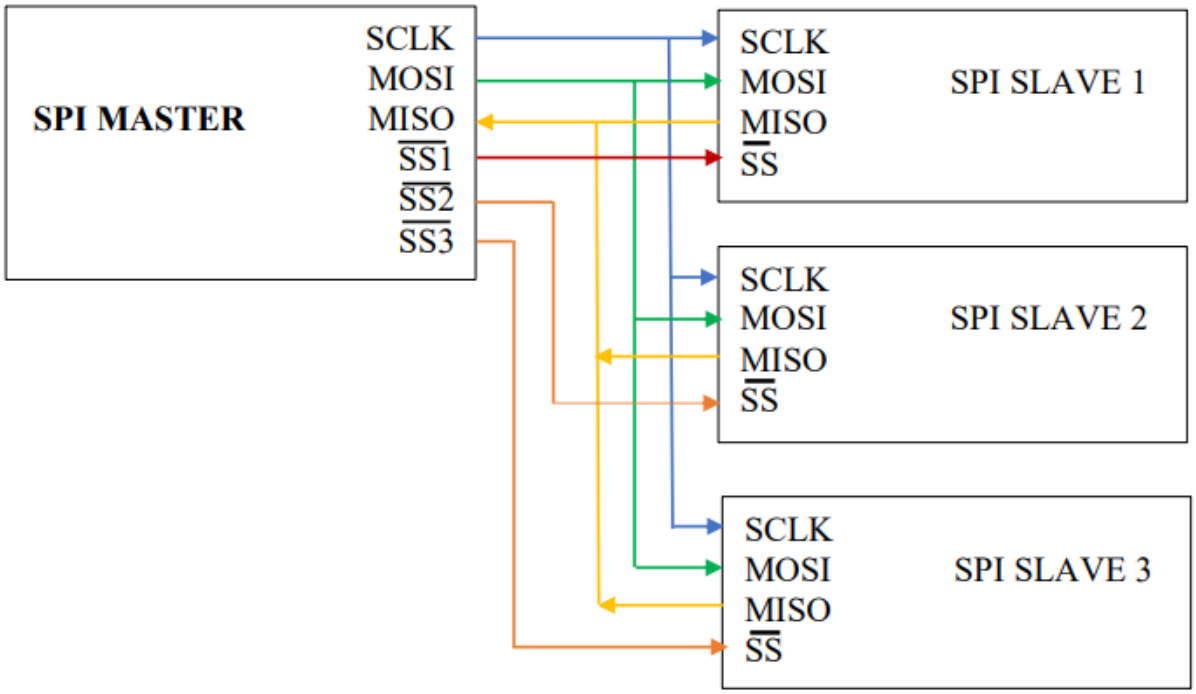
- תוך כדי שליחה של כל ביט מה- MASTER אל ה- SLAVE גם ה- SLAVE מוציא ביט אחרי ביט נתון אל קו ה- MISO.
- גם הביטים מה- SLAVE מסונכרנים בעזרת פולסי השעון ב- SCLK והם נכנסים לרגיסטר הזזה ב- MASTER.
- בסיום שליחת הביטים ה- MASTER מעלה את קו בחירת הרכיב עבד (Select slave) ל- 1 ומסיים תקשורת.
- בסיום התקשורת ה-MASTER לא מייצר פולסי שעון, והקו של פולסי השעון SCLK נמצא במצב IDLE מצב סרק. הוא יכול להיות ב-0 או ב-1.

חיבור מספר SLAVES אל ה- MASTER:

ניתן לחבר ל- MASTER אחד מספר SLAVES, מה שיקרה למעשה שהקווים המשותפים לכל ה- SLAVES הם SCLK, MISO, MOSI. הקו שנבדל בין כולם הוא הקו לבחירת רכיב העבד (Slave select).

כל ה- SLAVES מקבלים במקביל את פולסי השעון ואת קווי ה- MOSI ו- MISO. קו ה- Slave select של כל רכיב יהיה ב-1 (כלומר הרכיב לא נבחר) וה- MASTER יוריד את קו בחירת רכיב העבד ל- 0 רק עבור אותו ה-SLAVE שהוא רוצה לתקשר איתו.

נראה את האיור הבא שממחיש לנו חיבור של 3 SLAVES ל- MASTER:



חיבור של MATER אחד לשלושה SLAVES שונים דרך תקשורת טורית תSPI.

באיור זה נוכל לראות שלכל עבד יש קו (Slave select) משלו. ל- MASTER יש אותו מספר קווים בדומה למספר ה- SLAVES. לכל רכיב עבד יתחבר קו מסטר אחר, באיור הם נקראים:

(Slave select 1, Slave select 2, Slave select 3).

הם מסומנים עם קו מעליהם כדי להדגים שהם פעילים בנמוך (Active Low). לדוגמה אם ה- MASTER יחליט לתקשר רק עם רכיב SPI SLAVE 1 הוא יוריד את קו (Slave select 1) ל-0 ואז רק רכיב העבד הראשון ידע שה- MASTER מתקשר איתו. שאר העבדים יודעים שהמיקרו בקר אינו מתקשר איתם.

כמות ה- SLAVES שניתן לחבר אל ה- MASTER תלויים בכמות ההדקים הפנויים שיש לו במערכת בה הוא נמצא ובכושר של דחיפת הזרם שלו. אפשרויות מתקדמות יותר על מנת לחסוך בהדקים של המיקרו בקר הן להתחבר אל הדקי Slave select של כל רכיב SPI בעזרת מפענח או מפלג DEMUX עם כניסה אחת, 3 הדקי בחירה ו- 8 יציאות שונות.

אפשרויות העבודה עם תקשורת SPI:

במערכת ה- SPI (גם ה- MASTER וגם ה-SLAVE) יש 2 רגיסטרים, האחד הוא הרגיסטר הזזה המקבל את הדגימות ומזיז את הנתון, ועוד רגיסטר נתון שבסיום ההעברה הנתון שהתקבל נמצא בו.

בפולס שעון יש 2 מעברים (מגבוה לנמוך ולהפך) הכוללים גם עלייה וגם ירידה ויש לעשות 2 דברים:

א. במעבר מ-0 ל-1 גם המסטר וגם העבד מוצאים את ביט הנתון לקו הנתון (MOSI במסטר MISO בעבד).

ב. במעבר השני מתבצעת הזזה של הנתון ברגיסטר ההזזה שנמצא גם במסטר וגם בעבד.

שני פרמטרים/מאפיינים חשובים בתקשורת SPI:

1. CPOL (Clock POLarity – קוטביות השעון) הקובעת מה מצב הקו (נמוך או גבוה) במצב סרק – IDLE כאשר אין פולסי שעון (לפני התחלת תקשורת ובסיומה), כאשר CPOL=0 בקו יש 0 לוגי וכאשר CPOL=1 בקו יש 1 לוגי.
2. CPHA (Clock PHAse – פאזת השעון) הקובעת באיזה מעבר (האם מ-0 ל-1 או מ-1 ל-0) יוצא ביט הנתון בקו ה- MOSI ובקו ה- MISO ובאיזה מעבר הוא מוזז ברגיסטר הזזה גם במסטר וגם בעבד. פרמטר נוסף חשוב הוא באיזה מצב נמצא קו השעון SCLK בסיום התקשורת (האם ב 0 או ב 1) .

תבנית העברה של נתון עבור CPHA=0:

הקו Slave select פעיל בנמוך, ניתן לרשום אותו גם כך: NSS כדי שלא יהיה צורך לרשום גג SS.

האיור הבא מתאר את תבנית העברה עם צורות הגל של תקשורת SPI כאשר CPHA= 0 ועם פרמטרים של זמן אופייניים.

בחלק התחתון רואים את התחלת התקשורת כאשר קו NSS (Slave select) יורד ל-0 ואת סיומו כאשר קו זה עולה ל-1.

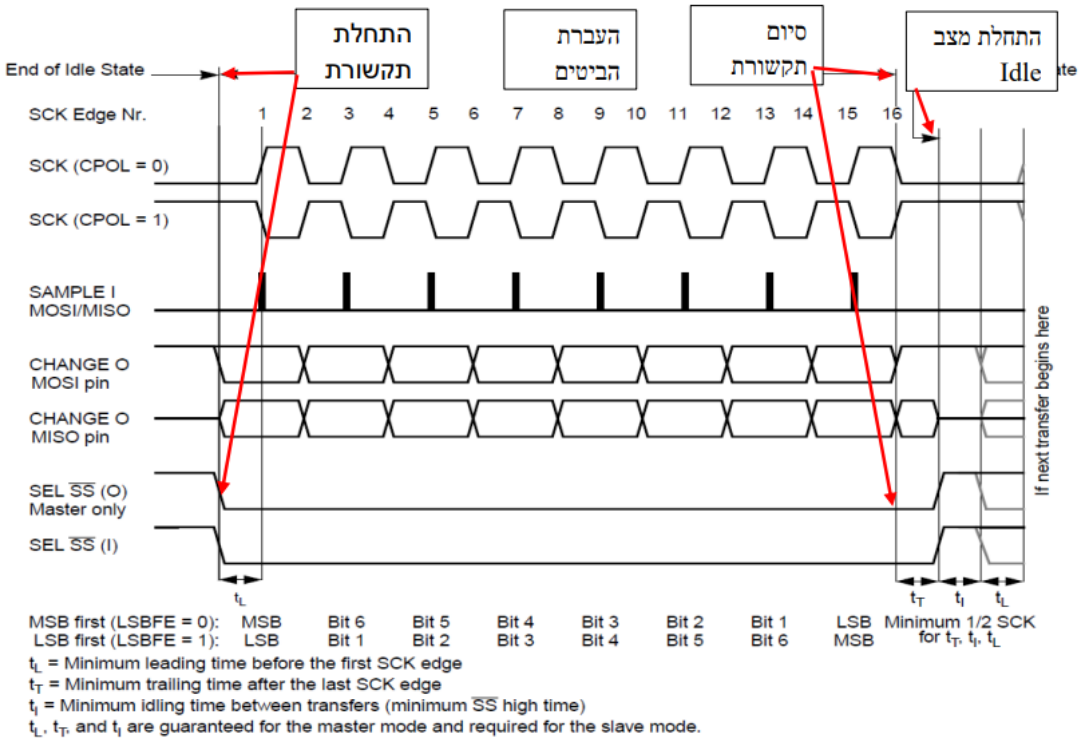
קו השעון SCK מתואר על ידי 2 צורות גלים . העליונה ביותר מתארת את פולסי השעון כאשר CPOL = 0 (כאשר אין תקשורת יש 0 בקו פולסי השעון) והפולס הראשון הוא עלייה וצורת הגל שמתחתיה כאשר CPOL = 1 ואז יש 1 לוגי בקו פולסי השעון והפולס הראשון הוא ירידה.

הראשון הוא ירידה.

במקום הרשום באיור O הכוונה ל- Output (יציאה) ובמקום שרשום I הכוונה ל- Input (כניסה).

הביטים של הנתון הטורי בהדקי MOSI ו- MISO נדגמים ומתבצעת הזזה ברגיסטרים של ההזזה באחת מ- 2 אפשרויות. או בעליית השעון או בירידת פולסי השעון ואת זה נקבע לפי נתוני הרכיב שאליו מתחברים.

איור: צורת גל בתקשורת SPI כאשר CPHA = 0.



המעבר הראשון של קו SCK (מ-0 ל-1 או מ-1 ל-0 תלוי ב-CPOL) משמש להכנסת ביט הנתון הראשון של העבד אל המסטר (בקו MISO) ואת ביט הנתון הראשון של המסטר אל העבד (בקו ה-MOSI). באיור ניתן לראות במרכז את הקו שנקרא SEMPLE והוא משמש ככניסה גם למסטר בקו ה- MISO וגם לעבד בקו ה- MOSI.

במעבר השני ,בחצי המחזור הבא , מופיע מעבר נוסף בקו SCK ואז הערך שנדגם מהעבד בקו MISO נכנס לתוך רגיסטר ההזזה שבמסטר. אחרי המעבר הזה משודר הביט הבא של המסטר על קו ה- MOSI התהליך נמשך עד 16 מעברים בקו SCK כאשר **נתון ננעל אל המסטר במעברים אי זוגיים ומועבר אל העבד במעברים זוגיים.**

אחרי מעבר מספר 16 הנתון שהיה ברגיסטר ה- SPI של המסטר צריך להיות ברגיסטר הנתון של העבד והנתון שהיה ברגיסטר הנתון של העבד צריך להיות במסטר.

באיור מופיעים זמנים כגון:

- tL המראה מה הזמן המינימאלי מרגע הורדת קו NSS (Slave select) ל- 0 ועד להתחלת פולסי השעון.
- tT הוא הזמן המינימאלי בין פולס השעון האחרון והעלאת קו NSS (Slave select) ל-0.
- Ti הוא זמן הסרק - IDLE – בין העלאת קו ה- NSS (Slave select) ל-1 ועד שידור נתון חדש על ידי הורדת NSS (Slave select) ל-0.

מתחת לצורות הגלים ניתן לראות שאפשר לשדר את הנתון מביט ה- LSB אל ה- MSB או להיפך.

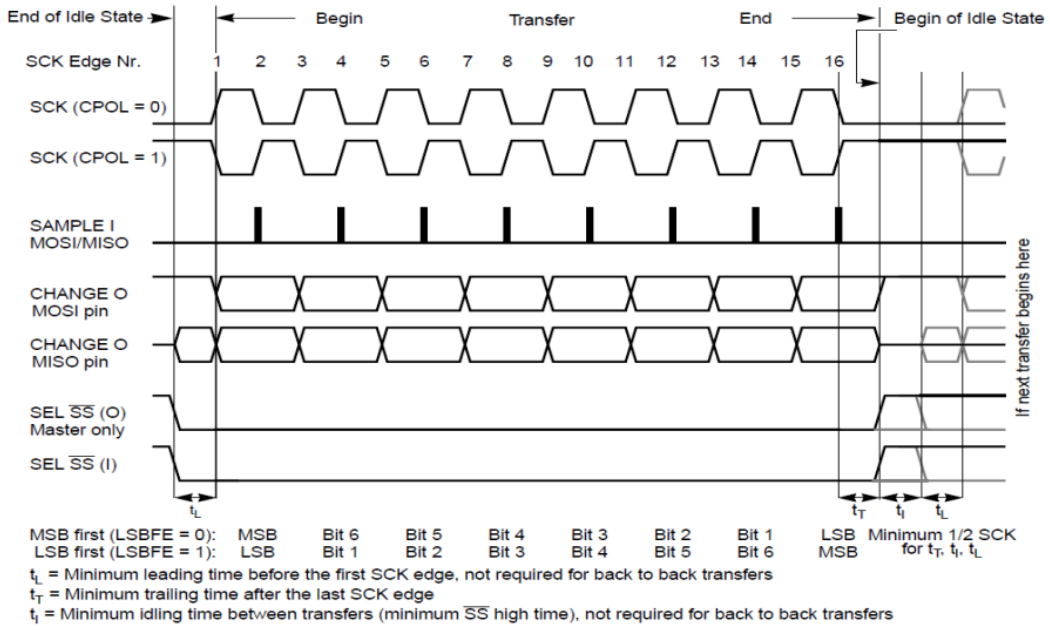
קיים ביט LSBFE (LSB First Enable אפשור LSB ראשון) שקובע מי נכנס ראשון . כאשר הוא 0 נכנס ביט ה MSB ראשון וכאשר הוא 1 נכנס ה- LSB ראשון.

קבלת הנתון מתבצעת בשני שלבים. היא מוזזת לרגיסטר ההזה של ה- SPI בזמן ההעברה ומועברת לרגיסטר הנתון של ה- SPI כאשר הביט האחרון הוזז פנימה.

תבנית העברה של נתון עבור CHPA = 1 :

קיימים רכיבים שצריכים את המעבר של פולס השעון הראשון לפני שניתן יהיה לגשת לביט הנתון הראשון בקו היציאה של הנתון .המעבר השני מכניס את הנתון למערכת. פורמט זה מתקבל על ידי השמה של CPHA ל- 1.

[האיור](#) הבא מתאר את התקשורת עבור CPHA = 1.

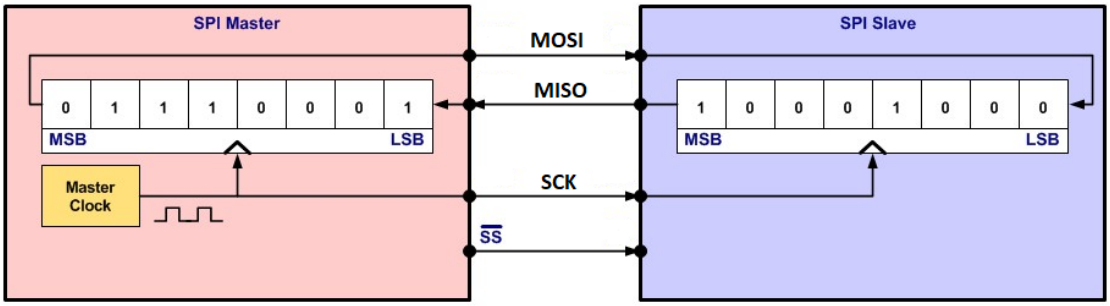


באיור רואים שהמעבר הראשון משמש כהשהיית סנכרון ואומר לעבד להוציא נתון בקו ה-MISO בחצי המחזור הבא, במעבר השני ,יש נעילה של ביט הנתון גם במסטר וגם בעבד. כאשר מגיע המעבר השלישי מועבר הביט שנגעל במעבר השני לתוך ה LSB או ה- MSB של רגיסטר ההזזה (כתלות בביט LSBFE).

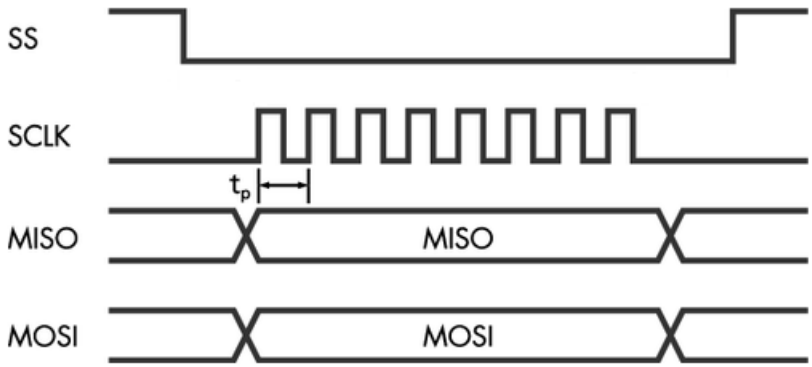
אחרי מעבר זה יוצא ביט הנתון הבא של המסטר בקו ה- MOSI. התהליך חוזר על עצמו עבור כל 16 המעברים, כאשר הנתון נגעל במעברים הזוגיים וההזזה קוראת במעברים האי זוגיים.

המידע מועבר בו זמנית בשני הכיוונים בכל עליית שעון באמצעות אוגרי הזזה .

בדוגמה הבאה: לאחר 8 מחזורי שעון מידע מוחלף בין ה- Master ל- Slave.



שליחת מילת בקרה של 8 סיביות.



בתחילת התקשורת, הדק SS יורדת ל- 0 . לאחר מכן מועבר המידע מה- Master דרך הדק MOSI במקביל. ה- Master יכול לקרוא את המידע מה- Slave דרך הדק MISO. בסיום התקשורת הדק SS חוזרת ל- 1 לוגי.

tp – זמן מחזור של השעון .

תקשורת I2C

הקדמה:

פרוטוקול I2C הוא שיטת תקשורת סריאלית נפוצה בין התקני אלקטרוניקה וחיישנים. הוא מאפשר העברת נתונים בין מספר התקנים, כאשר יתרונו הגדול הוא שהוא דורש רק שני חוטים לתקשורת – החוט לאותות השעון והחוט לאות הנתונים.

מהו הפרוטוקול I2C :

פרוטוקול I2C (אי-שתיים-סי) הוא שיטה סריאלית להעברת נתונים בין התקנים, שפותחה על ידי חברת פיליפס בשנות ה-80.

בניגוד לתקשורת UART שדורשת שני חוטי תקשורת לכל כיוון, ב- I2C יש רק שני חוטים משותפים לשני הכיוונים : SDA לנתונים ו- SCL לאות שעון. שני חוטים אלו מחוברים לכל ההתקנים בטופולוגיית BUS.

כל התקן ברשת I2C מזוהה בכתובת ייחודית. כאשר מתקן אחד רוצה לתקשר, הוא שולח הודעה עם הכתובת של המתקן היעד. רק המתקן עם אותה כתובת "יורם את השפופרת".

העברת הנתונים עצמה נעשית לפי עקרון מאסטר/עבד (לרוב מאסטר עובד), מפעיל את אות השעון ושולט מתי לשדר ומתי לקבל נתונים, ואילו המתקנים המשניים הם עובדים פסיביים.

ב- I2C כל ביט נשלח ברצף אחרי הביט הקודם, והוא תוקף כל עוד אות השעון SCL גבוה. ירדת SCL מהווה איתות לביט חדש.

יתרונותיו הגדולים של פרוטוקול I2C הן הפשטות והחיסכון בכמות החוטים לתקשורת, ניתן לחבר עשרות מתקנים קצה, וקל יחסית ליישם במיקרו-בקרים.

לכן זהו פרוטוקול נפוץ לתקשורת עם אלקטרוניקה כמו חיישנים, מסכי LCD, גרפים, זיכרונות וכד'.

מאפיינים פרוטוקול I2C:

- **תקשורת סידרתית (סריאלית)** -המידע מועבר בטור, ביט אחרי ביט, על גבי קו התקשורת.SDA
- **סינכרונית** -אות השעון SCL מסנכרן בין כל המכשירים.
- **תקשורת דו כיוונית** בין מאסטר למספר סלייבים.
- **שני קווי תקשורת** משותפים לכל המכשירים SDA - ו-SCL.
- **קצב תקשורת** גבוה יחסית - עד 5 מגה ואף יותר.
- **כתובות מוגדרות** מראש לכל רכיב (7/10 ביט כתובת).
- **ממשק חומרה** פשוט וסטנדרטי.
- **מתאים במיוחד** לתקשורת בין מעבד להתקני קצה כמו חיישנים, מסכים וזיכרונות.
- **מצמצם מאוד** בכמות החיווט לעומת.UART

מבנה חבילת (מסגרת) נתונים בתקשורת טורית:

כאשר מתקן מאסטר (כמו מיקרו-בקר) מעוניין לשדר נתונים למתקן SLAVE, הוא בונה את החבילה במבנה מוגדר הכולל מספר שדות:

- **התחלת העברה (START)** - ירידה באות SDA מ-1 ל-0 בזמן שאות SCL נשאר גבוה, מה שמהווה "אות התחלה."
- **כתובת SLAVE**- 7 או 10 ביטים עם הכתובת הייחודית של המתקן SLAVE היעד. רק אותו מתקן "יריב את השפופרת."
- **נתונים** - המידע עצמו בסדרת ביטים הנשלחים על ידי המאסטר או שמתקבלים מהמתקן ה-SLAVE.
- **אישור 9 (ACK)**- פעולות שעון שבהן ה-SLAVE מאשר קבלה תקינה של העברת 8 ביטים בקוSDA.
- **סיום העברה (STOP)**- עלייה באות SDA מ-0 ל-1 בזמן שקו SCL נשאר גבוה, מה שמהווה "אות סיום."

דוגמה: שליחת בייט בפרוטוקול I2C:

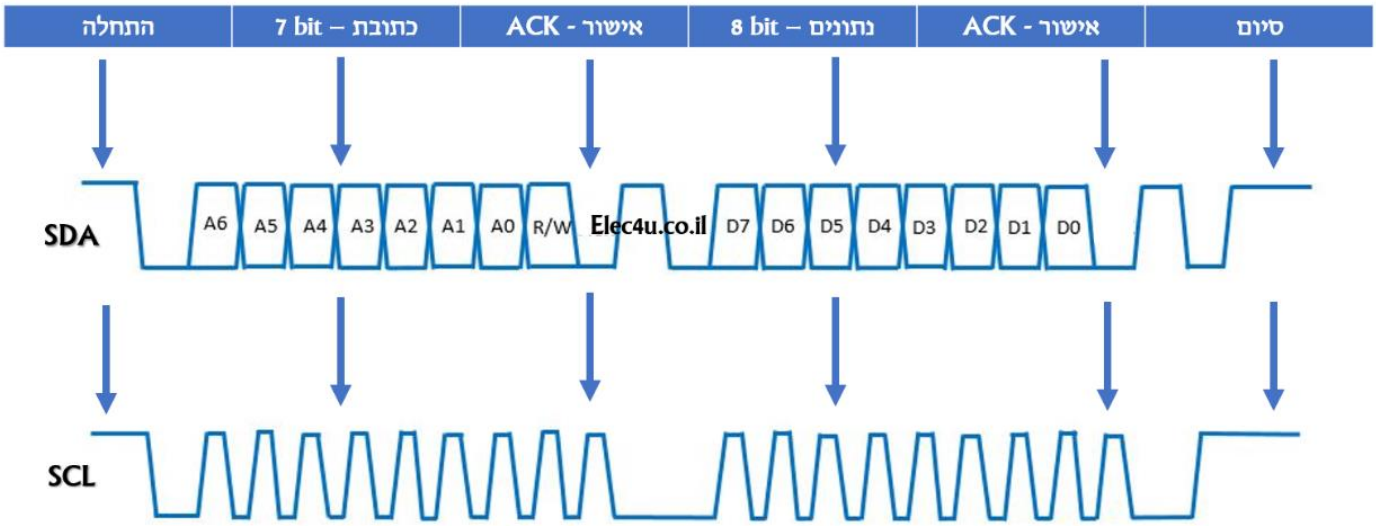
הנה פירוט דוגמה לשליחת הבייט 0x2F (מספר 47 עשרוני) למתקן בכתובת 0x38כך:

- 0111000 → **START** (0x38) כתובת → **STOP** → **ACK** → 00101111

פירוט שלבי התקשורת:

- **START**: מציין על התחלת העברה חדשה. נוצר על ידי ירידה של קו הנתונים **SDA** מ-1 ל-0, בעוד קו השעון **SCL** נשאר גבוה.
- 0111000 (כתובת 0x38): שבעת הביטים הבאים מציינים את כתובת ה-SLAVE (במקרה זה 0x38 או 56 עשרוני). הם נשלחים מהביט הנמוך (LSB) לביט הגבוה ביותר.(MSB)
- 00101111: שמונה הביטים הבאים מציינים את הנתונים שאנחנו שולחים ל-SLAVE (במקרה זה 47 עשרוני או 0x2F). שוב, מ- LSB ל- MSB.
- **ACK**: לאחר הנתונים, ה-SLAVE חייב לשלוח ביט "אישור" אחד בחזרה למאסטר כדי לאשר שקלט את הנתונים כראוי.
- **STOP**: סיום תנועת ה- I2C. נעשה על ידי העלאת קו הנתונים **SDA** בחזרה מ-0 ל-1, בעוד קו השעון **SCL** נשאר גבוה.

נתבונן באיור הבא:



בחבילת נתונים של פרוטוקול I2C ישנה אפשרות לשלוח מסגרת אחת או מספר מסגרות ברצף באותה החבילה. מסגרת נתונים אחת מכילה את כמות המידע שאנו רוצים לשלוח באותו הרגע. הפרוטוקול מאפשר לנו לשלב מספר מסגרות נתונים ברצף בתוך אותה החבילה, ללא צורך לשלוח חבילה חדשה עבור כל מסגרת. זאת על ידי שליחת אישור (ACK) מהמתקן המקבל בין מסגרת למסגרת. כך ניתן לשלוח ביעילות נתונים ארוכים יותר באותה החבילה I2C.

יתרונות מרכזיים של פרוטוקול I2C :

- **חסכוני בכמות החיווט** - משתמשים רק בשני קווי תקשורת משותפים לכל ההתקנים - קו SDA לנתונים וקו SCL לאות השעון.
- **תקשורת דו-כיוונית** - מאפשר הן למאסטר לשלוח נתונים והן ל-SLAVE להחזיר נתונים.
- **פשוט לחיבור התקני קצה רבים** - ניתן לחבר עשרות התקנים שונים על אותה רשת I2C.
- **ממשק תקשורת פשוט וסטנדרטי** - כל רכיבי החומרה מתוכננים לתמוך בפרוטוקול.
- **מהירויות תקשורת גבוהות** - עד 5 מגה-הרץ, מתאים להעברת נתונים מחיישנים, שליטה ועוד.
- **אמינות גבוהה** - שימוש ב ACK-ומנגנוני זיהוי שגיאות.
- **כתובות מוגדרות מראש** - מאפשר תקשורת מכוונת בין התקני הרשת.

כמות רכיבים שניתן לחבר לפרוטוקול I2C:

בפרוטוקול I2C ניתן לחבר **מספר רב של רכיבים**, אך ישנה הגבלה על הכמות המקסימלית. בפועל, ניתן לחבר **עד 128 רכיבי קצה (SLAVES)** לכל רשת I2C. הסיבה להגבלה זו היא **שימוש ב-7 ביטים לייצוג כתובת הרכיב בפרוטוקול**, מה שמאפשר טווח כתובות של 0 עד 127 (כלומר 128 כתובות שונות). כמות זו של 128 רכיבים נחשבת גבוהה יחסית ומספיקה לרוב המערכות והיישומים הנוכחיים.

הרחבה ל-10 ביטים:

ישנה אפשרות להשתמש גם בכתובות עם 10 ביטים, מה שמרחיב את טווח הכתובות ומאפשר חיבור של מספר רב יותר של רכיבים. עם זאת, השימוש ב-10 ביטים מורכב יותר ופחות נפוץ.

הגבלות נוספות:

מלבד מגבלת הכתובות, קיימות גם **מגבלות פיזיות** שעלולות להשפיע על כמות הרכיבים המקסימלית שניתן לחבר בפועל. מגבלות אלו כוללות:

- **קיבוליות (Capacitance)**: ככל שמוסיפים יותר רכיבים, כך גדלה הקיבוליות על קווי התקשורת SDA ו-SCL. קיבוליות גבוהה מדי יכולה לעוות את האותות החשמליים ולשבש את התקשורת.
- **אורך החוטים**: ככל שהחוטים ארוכים יותר, כך גדלים הסיכון לרעש חשמלי ולירידת מתח.
- **זיהום סביבתי**: סביבה רועשת מבחינה חשמלית עלולה להשפיע על האותות.

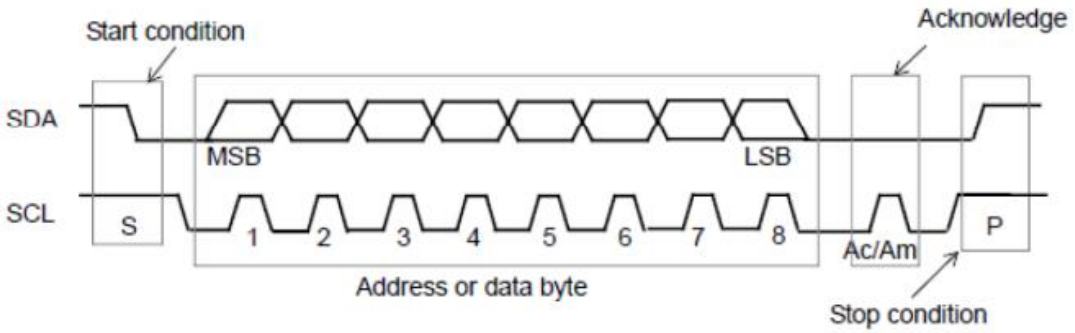
בפועל, למרות התיאוריה של 128 כתובות, יש לקחת בחשבון את המגבלות הללו, ולרוב רשתות I2C פשוטות מוגבלות למספר קטן יותר של רכיבים.

פעולות בסיסיות בפרוטוקול התקשורת I2C:

כאשר שני האותות SCL ו- SDA, נמצאים במצב 1 לוגי ה- BUS במנוחה. השידור עצמו נעשה בשיטת MSB – First.

- האות SDA חייב להיות יציב כשהאות SCL במצב 1 לוגי.
- האות SDA יכול להשתנות רק כשהאות SCL במצב 0 לוגי.

נראה תמונה שמתארת העברת מידע בפרוטוקול בדומה לתמונה מהעמוד הקודם:



ניתן לראות שפעולת START מתבצעת כאשר SDA בירידה ו- SCL נמצא עדיין ב-1 לוגי. לעומת זאת פעולת STOP מתבצעת כאשר SDA בעלייה ו-SCL כבר ב-1 לוגי.

הרכיב שמקבל את המידע (ה- MASTER או ה-SLAVE) מחזירים Acknowledge באמצעות הורדת SDA ל- 0 לוגי, רכיבי SLAVE חייבים תמיד להשיב באות Acknowledge לפנייה לכתובת שלהם.

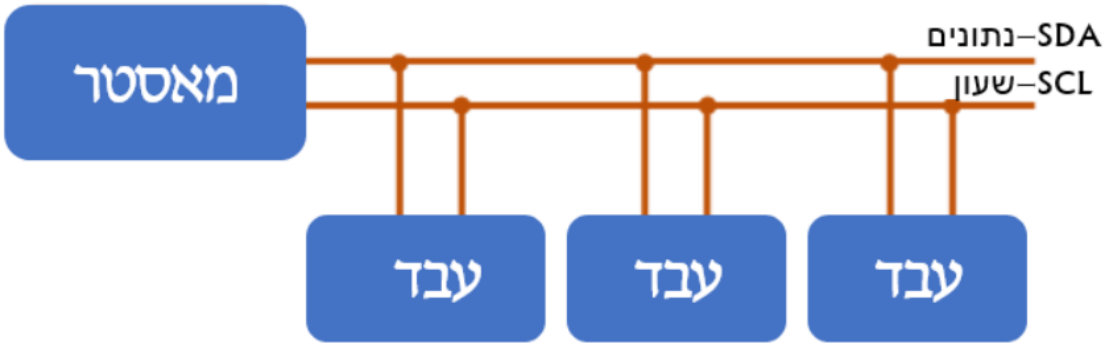
מצבים בהם רכיבי SLAVE לא מגיבים ב- Acknowledge:

רכיבי SLAVE יכולים שלא להגיב בתגובת Acknowledge (כלומר להגיב ב- NACK), במקרים הבאים:

- ה-SLAVE עסוק (במקרה כזה הוא יכול גם להאריך את ה-SCL).
- ה-SLAVE קיבל מידע לא חוקי.

לאחר קבלת NACK ה-MASTERחייב לבצע פעולת STOP. כאשר ה-MASTER הוא ה-Receiver הוא יכול שלא להגיב באות Acknowledge (כלומר להגיב ב- NACK)כדי לסמן שהגיע ל- Byte הסופי של המידע. מערכות שפועלות בפרוטוקול I2C יכולות לבצע פעולות Acknowledge גם ברמת ה- Byte.

הסבר כיצד הפרוטוקול I2C פועל:



פרוטוקול I2C הוא שיטת תקשורת טורית בין מיקרו-בקר לרכיבי ציוד היקפי, המשתמשת בשני קווים בלבד.

- **רגל** - (Serial Data) **SDA** אחראית על שליחת וקבלת המידע בין המכשירים.
- **רגל** - (Serial Clock) **SCL** אחראית על אות השעון, המסנכרן את התקשורת בין כל רכיב, כך שהם "מדברים" יחד ובזמן הנכון.

הארדואינו/ESP32, הם ה"מאסטרים" והוא שולט על הקווים, והשאר הם ה"עבדים" (SLAVES). כאשר הארדואינו/ESP32 רוצה לתקשר עם אחד החיישנים, הוא שולח אליו את הכתובת הייחודית שלו. לדוגמה:

- תצוגת LCD I2C 2X16 עובדת בכתובת 0x27.
- חיישן טמפרטורה LM75עובד בכתובת 0x07.

מיקרו בקר ESP32

ה- **ESP32** הוא הרבה יותר מסתם מיקרו-בקר פשוט; הוא מערכת על שבב (SoC) חזקה ורב-תכליתית, המשלבת את כל מה שצריך כדי לבנות פרויקט חכם, מקושר וחסכוני באנרגיה. פרי פיתוחה של חברת **Espressif Systems**, ה- ESP32 בולט בזכות השילוב הייחודי של עוצמת עיבוד חזקה, קישוריות אלחוטית ויכולת צריכת חשמל נמוכה.

תכונות טכניות עיקריות:

1. **עיבוד בעל ביצועים גבוהים:** מצויד במעבד כפול-ליבה במהירות של עד 240 MHz, ה- ESP32 מסוגל לבצע משימות מורכבות ולנהל מספר פעולות במקביל במהירות גבוהה.
2. **קישוריות אלחוטית מובנית:** ה- ESP32 מגיע עם **Wi-Fi** ו- **Bluetooth** מובנים, מה שהופך אותו לבחירה אידיאלית לפיתוח פרויקטים מחוברים ללא צורך ברכיבים חיצוניים נוספים.
3. **יעילות אנרגטית:** עם מצבי שינה מרובים, ה- ESP32 מתוכנן לפעול לטווח ארוך על סוללה בודדת, מה שהופך אותו למושלם עבור יישומים המופעלים באמצעות סוללה.
4. **ממשקי קלט/פלט (I/O) גמישים:** ה- ESP32 מציע מגוון רחב של ממשקים היקפיים, כולל **ADC, DAC, I2C, SPI** ו- **GPIO**, ומספק כמעט אינסוף אפשרויות ליצירה.
5. **תמיכת קהילה נרחבת:** עם אלפי מפתחים ברחבי העולם המשתמשים ב- ESP32, קיים שפע של ספריות קוד, תיעוד ופרויקטיי קוד פתוח הזמינים לתמיכה בפיתוח שלך.

הסוד שמאחורי ה-ESP32: יחידת העיבוד המרכזית (CPU)

ה-ESP32 מצויד במעבד עוצמתי **Xtensa LX6 dual-core**, הפועל במהירות מרשימה של עד 240MHz. אבל מה זה בעצם אומר?

- דמיינו שיש לכם שני מוחים עובדים במקביל: בזמן שאחד מהם מטפל בקישוריות Wi-Fi, השני יכול לעבד נתונים מהחיישנים שלכם - הכל בבת אחת!
- **240MHz** זוהי מהירות השעון של המעבד. לשם השוואה, המעבד של ה-ESP32 מסוגל לבצע **240 מיליון פעולות בשנייה**!
- **ארכיטקטורת 32-ביט** זוהי אומרת שהמעבד יכול לעבד נתונים בכמויות גדולות יותר, מה שמאפשר חישובים מורכבים במהירות גבוהה.

הזיכרון: איפה כל המידע נשמר?

ה-ESP32 מגיע עם מערך מרשים של אפשרויות זיכרון:

- **SRAM (Static Random-Access Memory):** זיכרון מהיר לשימוש מיידי של **520KB**. זהו המקום שבו המידע שהמעבד עובד עליו ברגע נתון נשמר.
- **ROM (Read-Only Memory):** זיכרון קבוע של **448KB** המכיל את ההוראות הבסיסיות הנדרשות להפעלת המערכת.
- **Flash (זיכרון חיצוני):** זיכרון של עד **16MB** שבו נשמרות תוכניות המשתמש שלכם והנתונים הקבועים.

לשם השוואה, הזיכרון הכולל של ה-ESP32 הוא כל כך גדול, שתוכלו לאחסן בו ספר אודיו שלם!

קישוריות אלחוטית: החיבור לעולם:

ה-ESP32 הוא אלוף התקשורת האלחוטית:

- **Wi-Fi:** תומך בתקן **802.11 b/g/n**, מה שמאפשר חיבור כמעט לכל רשת אלחוטית מודרנית. הוא יכול לשמש גם כנקודת גישה ולספק חיבורים למכשירים אחרים.
- **Bluetooth:** עם תמיכה ב-**Bluetooth Classic** וגם ב- **Bluetooth Low Energy (BLE)**, ה-ESP32 יכול לתקשר עם מגוון רחב של מכשירים, מטלפונים חכמים ועד חיישנים זעירים.

ממשקים: חיבור לעולם הפיזי:

ה-ESP32 מציע מגוון רחב של אפשרויות חיבור כדי לתקשר עם העולם הפיזי:

- **GPIO (General Purpose Input/Output):** עד 36 פינים, המשמשים כ"עצבים" של ה-ESP32, המאפשרים לו לחוש ולשלוט בסביבה.
- **ממיר אנלוגי-דיגיטלי (ADC):** ישנם 18 ערוצים המאפשרים ל-ESP32 לקרוא ערכים אנלוגיים מחיישנים כמו טמפרטורה או עוצמת אור.
- **ממיר דיגיטלי-אנלוגי (DAC):** שני ערוצים המאפשרים לייצר אותות אנלוגיים כמו צלילים, או מתחים משתנים.
- **חיישני מגע (Touch Sensors):** 10 פיני מגע מובנים המאפשרים ל-ESP32 לחוש מגע ישיר.

צריכת חשמל: יעילות אנרגטית מרשימה:

אחד היתרונות הגדולים ביותר של ה-ESP32 הוא היכולת שלו לחסוך באנרגיה:

- **מצב פעיל מלא:** במצב זה, צריכת הזרם היא עד **mA240**.
- **מצב שינה עמוקה:** במצב זה, הצריכה יורדת באופן דרמטי ל-**µA10** בלבד!

זה אומר שגם עם סוללה קטנה, הפרויקט שלכם יכול לפעול במשך שבועות ואף חודשים ארוכים.

כעת נראה את צורת הפנים במיקרו בקר ESP32:

כפי שכבר ראינו, ה-ESP32 מוקף בפינים מרובים, שהם ה"שערים" דרכם הוא מתקשר עם העולם החיצוני. הבנה מעמיקה של פינים אלו היא המפתח ליצירת פרויקטים מדהימים. בואו נצלול לעולם המרתק של פיני ה-ESP32:

סוגי הפינים העיקריים:

פיני GPIO (General Purpose Input/Output)

- אלו הם הפינים הרב-תכליתיים של ה-ESP32.
- יכולים לשמש כקלט (לקריאת מצבים) או כפלט (להפעלת רכיבים).
- ה-ESP32 מציע עד **34 פיני GPIO**, אך חלקם משמשים גם למטרות אחרות.
- **שימושים נפוצים:** חיבור נורות LED, לחצנים, מנועים ועוד.

פיני ADC (Analog to Digital Converter):

- מאפשרים קריאה של ערכים אנלוגיים והמרתם לערכים דיגיטליים.
- ה-ESP32 מציע שני מגעלי ADC עם סה"כ **18 ערוצים**.
- **רזולוציה של 12 ביט** מאפשרת **4,096 רמות שונות** של קריאה.
- **שימושים:** מדידת טמפרטורה, עוצמת אור, לחות קרקע ועוד.

פיני DAC (Digital to Analog Converter):

- ממירים ערכים דיגיטליים לאותות אנלוגיים.
- ה-ESP32 מציע שני ערוצי DAC ברזולוציה של **8 ביט**.
- **שימושים נפוצים:** יצירת צלילים, בקרת מתח ועוד.

פיני Touch:

- ה-ESP32 מציע **10** חיישני מגע קפסיטיביים מובנים.
- מאפשרים זיהוי מגע אנושי ישיר, ללא צורך ברכיבים נוספים.
- **שימושים נפוצים:** יצירת ממשקי משתמש מבוססי מגע, לחצנים וירטואליים ועוד.

פיני תקשורת:

- **UART:** לתקשורת טורית. ה-ESP32 מציע **3 יחידות UART**.
- **SPI:** לתקשורת מהירה. ה-ESP32 תומך במספר חיבורי SPI במקביל.
- **I2C:** לתקשורת עם מגוון רחב של חיישנים ורכיבים. ה-ESP32 מציע שתי יחידות **I2C**.

פינים מיוחדים:

- **פין BOOT:** משמש לכניסה למצב **צריבת תוכנה (flashing mode)**.
- **פין EN (Enable):** משמשים להפעלה וכיבוי של השבב.
- **פיני אספקת מתח:** פינים המספקים מתחים של **3.3V**, **5V** וכן **GND**.

נקודות חשובות לגבי השימוש בפינים:

1. **רמות מתח:** פיני ה-ESP32 עובדים ברמת מתח של **3.3V**. חיבור למתח גבוה יותר עלול לגרום נזק.
2. **פינים מרובי תפקידים:** רבים מהפינים יכולים לשמש למספר מטרות. חשוב לתכנן את השימוש בהם בקפידה.
3. **הגנה על הפינים:** מומלץ להשתמש בנגדים מגבילי זרם בעת חיבור רכיבים חיצוניים, במיוחד עם נוריות LED.
4. **פינים מיוחדים:** חלק מפיני ה-GPIO (כמו 0, 2, 15) משמשים במהלך תהליך האתחול. שימוש לא נכון בהם עלול למנוע מה-ESP32 לפעול כראוי.
5. **פולאפ ופולאדאון (Pull-up and Pull-down):** חלק מהפינים כוללים נגדי Pull-Up או Pull-Down פנימיים. זה יכול להוות שימוש חשוב, אך חשוב להיות מודעים לכך בעת תכנון המעגל.

הבנה מעמיקה של מפת הפינים ותכונותיהם היא המפתח ליצירת פרויקטים מוצלחים ויעילים עם ESP32. זה מאפשר לנו לנצל את מלוא הפוטנציאל של השבב העוצמתי הזה.

ESP32: הכוח האלחוטי של Wi-Fi ו-Bluetooth:

אחד היתרונות הגדולים ביותר של ה-ESP32 הוא יכולתו המובנית לתקשר באופן אלחוטי.

Wi-Fi: החיבור לעולם הרחב

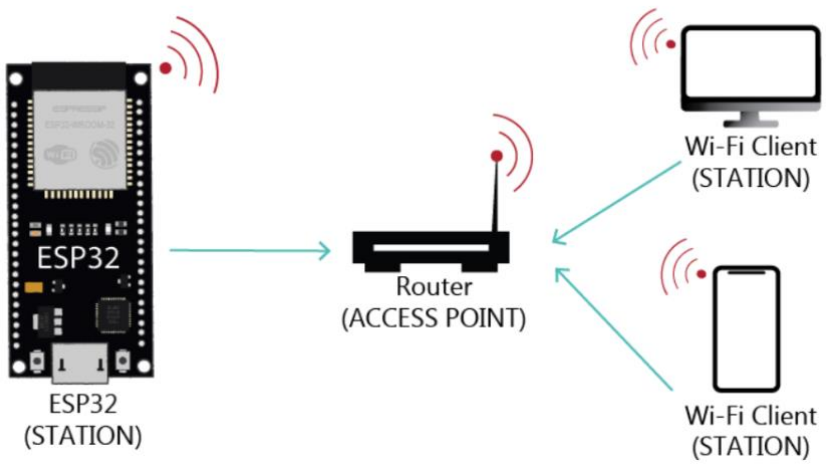
ה-ESP32 תומך בתקן Wi-Fi **b/g/n 802.11**, מה שמאפשר לו להתחבר כמעט לכל רשת אלחוטית מודרנית.

מצבי פעולה עיקריים:

- **מצב תחנה (Station Mode - STA):**
  - ה-ESP32 מתחבר לרשת Wi-Fi קיימת, כמו כל מכשיר קצה רגיל.
  - שימושים נפוצים: התחברות לאינטרנט דרך הראוטר הביתי, שליחת נתונים למסד נתונים מרוחק ועוד.
- **מצב נקודת גישה (Access Point - AP):**
  - ה-ESP32 יוצר רשת Wi-Fi משלו, אליה יכולים להתחבר מכשירים אחרים.
  - מצוין ליצירת פרויקטים שמספקים ממשק אינטרנט או תקשורת ישירה עם מכשירים ניידים.
  - שימושים נפוצים במצבים בהם אין גישה לרשת Wi-Fi חיצונית.
- **מצב כפול (Dual Mode):**
  - ה-ESP32 יכול לפעול במקביל גם כתחנה וגם כנקודת גישה בו-זמנית!
  - מאפשר יצירת "גשר" בין רשתות או אספקת גישה לאינטרנט דרך ה-ESP32.

יכולות מתקדמות של Wi-Fi:

- **שרת אינטרנט (Web Server):** ה-ESP32 יכול לשמש כשרת אינטרנט, מה שמאפשר יצירת ממשקי משתמש מרחוק ודפי תצוגה נתונים.
- **MQTT:** תמיכה בפרוטוקול זה מאפשרת תקשורת יעילה בין מכשירי IoT ושרתים מרכזיים.
- **OTA (Over-The-Air) Updates:** עדכון תוכנה מרחוק, ללא צורך בחיבור פיזי למחשב.
- **ESP-NOW:** פרוטוקול תקשורת ייחודי של Espressif המאפשר תקשורת ישירה בין מכשירי ESP32 בצריכת אנרגיה נמוכה.



Bluetooth: תקשורת קצרת טווח עם יכולות ארוכות טווח

ה-ESP32 תומך הן ב-**Bluetooth Classic** והן ב-**Bluetooth Low Energy (BLE)**, מה שפותח אפשרויות רבות:

Bluetooth Classic:

- מתאים לתקשורת עם מכשירים ישנים יותר או כשנדרשת העברת נתונים בנפח גדול.
- שימושים נפוצים: הזרמת אודיו, העברת קבצים, ותקשורת עם מכשירים שאינם תומכים ב-BLE.
- מאפשר יצירת פרופילים שונים כמו **Serial Port Profile (SPP)** לתקשורת טורית אלחוטית.

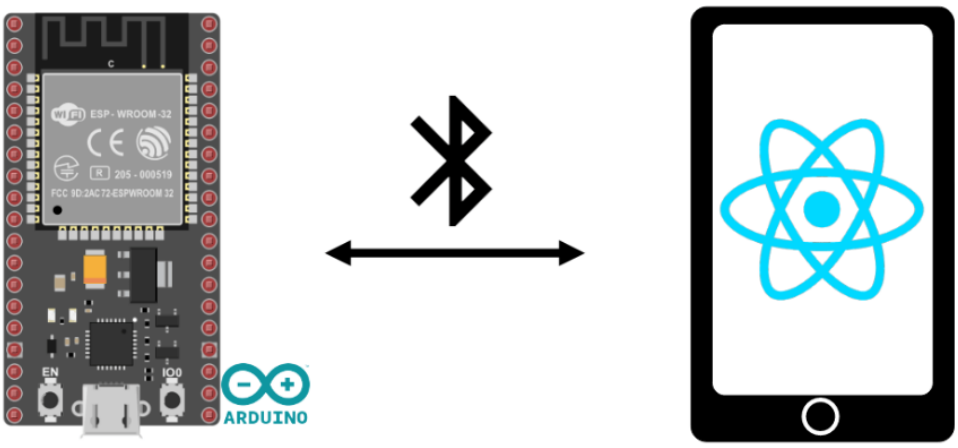
Bluetooth Low Energy (BLE):

- צריכת אנרגיה נמוכה במיוחד, אידיאלי למכשירים המופעלים על סוללה.
- מתאים לשליחת נתונים קטנים באופן תדיר, כמו נתוני חיישנים.
- תומך במודל **GATT (Generic Attribute Profile)** המאפשר יצירת שירותים ומאפיינים מותאמים אישית.
- מאפשר יצירת "ביקונים" לשיווק מידע ללא צורך בחיבור מלא.

שילוב Bluetooth ו-Wi-Fi: יתרונות מרתקים:

אחד היתרונות הגדולים של ה-ESP32 הוא היכולת להשתמש גם ב-Bluetooth וגם ב-Wi-Fi במקביל. זה פותח אפשרויות מרתקות:

- **שער (Gateway):** קבלת נתונים ממכשירי **Bluetooth** ושליחתם לענן דרך **Wi-Fi**.
- **שליטה מקומית ומרוחקת:** שליטה בפרויקט דרך **Bluetooth** כשקרובים, ודרך **Wi-Fi** כשמרוחקים.
- **רשתות היברידיות:** שימוש ב-**BLE** לתקשורת בין חיישנים קרובים, וב-**Wi-Fi** לשליחת נתונים מרובים לשרת מרכזי.



יתרונות מרכזיים של ESP32:

1. **עוצמה חישובית**: עם מעבד כפול-ליבה במהירות של עד 240 MHz, ה-ESP32 מאפשר ביצוע משימות מורכבות ועיבוד נתונים מהיר, מה שהופך אותו למתאים לפרויקטיי IoT מתקדמים.

2. **קישוריות אלחוטית מקיפה**: תמיכה מובנית ב- **Wi-Fi** ו- **Bluetooth** (כולל **BLE**) מאפשרת יצירת מגוון רחב של פרויקטים מחוברים, החל ממערכות בית חכם ועד ליישומים תעשייתיים.

3. **גמישות בחיבורים**: מגוון רחב של ממשקים כמו **GPIO, ADC, DAC** ופרוטוקולי תקשורת כמו: **I2C, UART, SPI** מאפשרים אינטגרציה קלה עם כמעט כל סוג של חיישן או מפעיל.

4. **יעילות אנרגטית**: מצבי שינה מתקדמים וצריכת חשמל נמוכה מאפשרים יצירת מכשירים ניידים עם חיי סוללה ארוכים, מה שהופך אותו לאידיאלי למערכות ניטור מרוחקות.

5. **פיתוח נגיש**: תמיכה במגוון סביבות פיתוח כמו: **ESP-IDF** ו- **Arduino IDE** וקהילת מפתחים גדולה ופעילה מקלים על תהליך הפיתוח ומספקים שפע של ספריות תוכנה.

6. **יכולות IoT מתקדמות**: תמיכה בפרוטוקולים כמו **MQTT** ויכולות עדכון תוכנה מרחוק (**OTA**) מאפשרות הקמת שרתי **web** מקומיים וניהול התקנים מכל מקום בעולם.

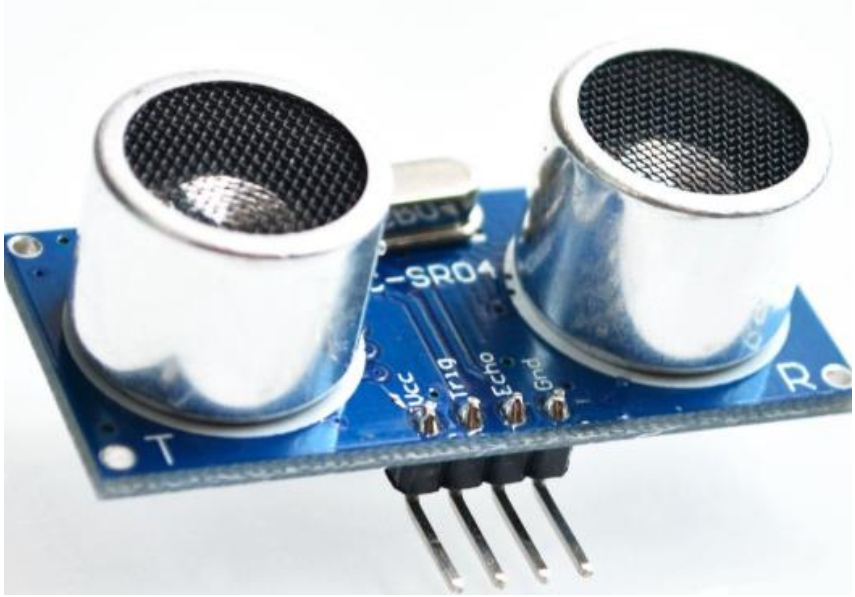
7. **אבטחה משולבת**: תכונות אבטחה מובנות כמו הצפנת חומרה ו- **secure boot** חיוניות להגנה על מכשירי IoT ועל נתוני המשתמשים בעולם המחובר.

8. **חיישנים מיוחדים מובנים**: ה- ESP32 כולל חיישני מגע קפסיטיביים, המאפשרים יצירת ממשקי משתמש חדשניים ללא צורך ברכיבים נוספים.

9. **יכולות רשת מתקדמות**: תמיכה ביצירת רשתות **Mesh** ושימוש בפרוטוקול **ESP - NOW** מאפשרות יצירת רשתות תקשורת מרובות ורב-כיווניות.

10. **התאמה למגוון יישומים**: מפרויקטים פשוטים ועד למערכות תעשייתיות מורכבות, ה-ESP32 מציע את הגמישות והעוצמה הדרושות למגוון רחב של יישומים.

חיישן המרחק – HC-SR04 :



ה– HC-SR04 זהו החיישן בו השתמשתי למדידת מרחק בפרוייקט שלי. זהו חיישן מאוד נפוץ בפרויקטים של רובוטיקה, חיישני חנייה, ומכשירים חכמים. חיישן זה נפוץ בגלל מספר דברים, כמו למשל: הוא נחשב לזול, פשוט לשימוש ומדויק יחסית בטווחים קצרים ובינוניים.

איך החיישן עובד?

העיקרון מאחורי HC-SR04 מבוסס על גלי קול על-קוליים .Ultrasonic Waves – החיישן משדר גל קול בתדר גבוה, שאינו נשמע לאוזן האנושית (כ-40 קילו-הרץ). הגל הזה מתפשט בסביבה, וכאשר הוא פוגע במכשול או חפץ כלשהו, הוא חוזר אל החיישן. החיישן מודד את הזמן שלקח לגל להגיע חזרה. בעזרת מהירות הקול באוויר, ניתן לחשב את המרחק למכשול.

הנוסחה שמאפשרת לחשב את המרחק היא:

$$\text{מרחק} = \frac{\text{זמן החזרה} \times \text{מהירות הקול}}{2}$$

החלק החשוב הוא חלוקה ב-2, שכן הזמן שנמדד הוא הזמן הכולל של ההלוך והחזור של הגל.

מהירות הקול באוויר אינה קבועה ,והיא תלויה בעיקר בטמפרטורה. ככל שהטמפרטורה גבוהה יותר, המולקולות באוויר נעות מהר יותר ומעבירות את גל הקול ביעילות רבה יותר. כתוצאה מכך, מהירות הקול עולה.

נוסחה לחישוב מהירות הקול

ניתן לחשב את מהירות הקול באוויר (במטרים לשנייה) בצורה מקורבת באמצעות הנוסחה הבאה :

מהירות= $331.4+0.6\times T$

כאשר:

- T היא הטמפרטורה במעלות צלזיוס.(°C)
- 331.4 מ/ש היא מהירות הקול באוויר בטמפרטורה של C.0°
- 0.6 מ/ש היא הקירוב לשינוי במהירות הקול עבור כל עלייה של מעלה אחת בטמפרטורה.

השפעה על הדיוק

אם לא מתחשבים בטמפרטורה, חישוב המרחק יכול להיות לא מדויק.

המערכות כה חשובות כגון כיפות ברזל או רובוט שומר בהתחשב בפרוייקט שלי, יש צורך להתחשב בטמפרטורה הסביבתית על מנת להבין כיצד להגיע למרחק שמתקבל על פי החיישן האולטרסוני.

מבנה החיישן:

HC-SR04 מגיע עם ארבעה פינים עיקריים :

- VCC - מתח אספקה של 5 וולט.
- GND - חיבור לאדמה.
- Trig - פין טריגר, אליו שולחים פולס קצר (בדרך כלל 10 מיקרו-שניות) כדי להפעיל את שליחת הגל האולטרסוני.
- Echo - פין החזרה, שמפיק פולס שמתמשך לאורך זמן ההחזרה של הגל. אורך הפולס הזה מאפשר לחשב את המרחק.

מגבלות ויתרונות

מצוין למדידה בטווחים של 2 ס״מ עד 400 ס״מ. עם זאת, יש לו מגבלות : HC-SR04 משטחים קטנים או סופגים עלולים להקשות על קריאת המרחק, כמו גם רעשים חזקים באוויר שמפריעים לגלים האולטרסוניים. בנוסף, הוא רגיש לזווית ההחזרה של הגל – החיישן עובד הכי טוב כאשר המשטח ניצב אליו.

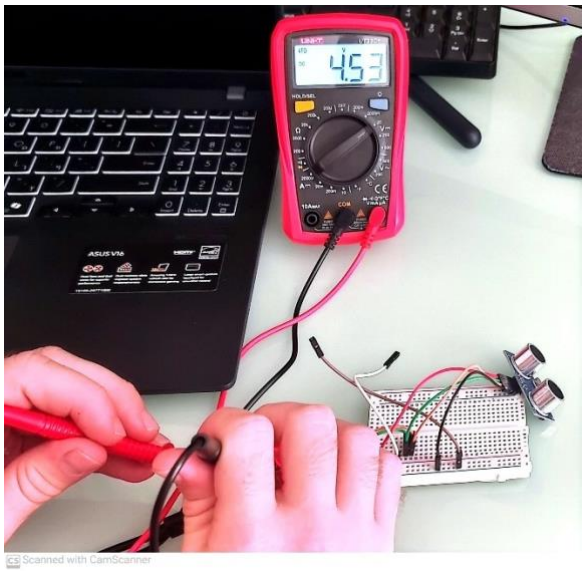
מדידות מתח ואופן פעולה:

מדידת מתחים של החיישן האולטרסוניק אל המיקרו בקר ESP32, אראה את המדידה אל המתח אליו אני מחבר את החיישן עצמו אל המיקרו בקר. בנוסף לכך אראה את המתח שמורה הפין של ה- 3.3V, האם הוא באמת מוציא מתח זה:

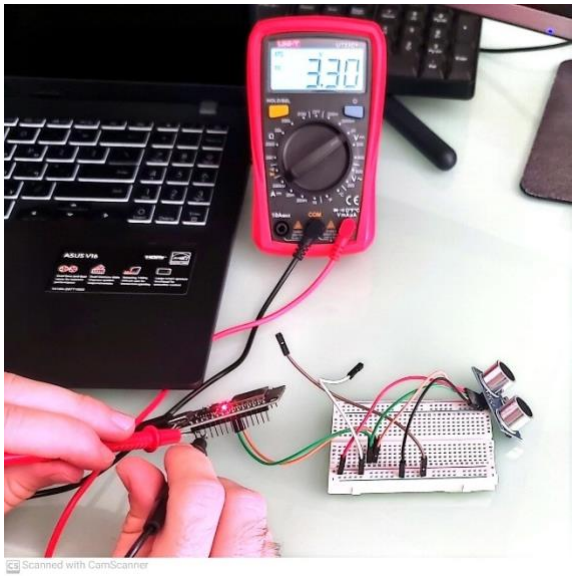
נראה תמונות:

```
// חיבור חיישן אולטרסוניק HC-SR04 ל-ESP32 והצגת תוצאות בסיריאל
// פין ה-Trig שולח פולס
#define TRIG_PIN 27
// פין ה-Echo מחזיר את הפולס
#define ECHO_PIN 26
// משתנה לשמירת הזמן שלוקח לפולס לחזור
long duration;
// המרחק בסנטימטרים
int distance;

void setup() {
  // פתיחת תקשורת סיריאלית לצפייה בתוצאות
  Serial.begin(115200);

  // הגדרת פינים
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);

  Serial.println("מדידת מרחק עם חיישן אולטרסוניק ESP32");
}

void loop() {
  // שליחת פולס קצר ל-Trig
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  // קריאת משך הזמן שה-Echo היה HIGH
  duration = pulseIn(ECHO_PIN, HIGH);

  // חישוב מרחק בס"מ
  distance = duration * 0.034 / 2;

  // הדפסת המרחק למסך הסיריאלי
  Serial.print("מרחק: ");
  Serial.print(distance);
  Serial.println(" מ"\");

  // המתנה קצרה בין מדידות
  delay(500);
}
```

המתח שאני מספק לחיישן:



המתח שמוציא המיקרו בקר מהפין 3.3V:



חיישן טמפרטורה LM75

חיישן הטמפרטורה **LM75** אינו רק רכיב אלקטרוני פשוט אלא הוא מערכת-על-שבב (SoC) דיגיטלית קומפקטית שנועדה למדוד טמפרטורה ולהמיר אותה לאותות שמיקרו-בקרים כמו הארדואינו או ה- ESP32 יכולים להבין בקלות. בניגוד לחיישנים אנלוגיים (כמו ה- LM35) שמוציאים מתח משתנה, ה LM75-מבצע את ההמרה לאות דיגיטלי בתוכו, מה שמפשט משמעותית את החיבור והתכנות.

מאפיינים ודיוק טכניים:

- **טווח מדידה :** החיישן מתוכנן למדוד טמפרטורות בטווח רחב של **-55°C עד +125°C** טווח זה הופך אותו למתאים ליישומים תעשייתיים, רפואיים וביתיים.
- **רזולוציה ודיוק :**החיישן מספק קריאה ברזולוציה של **9 ביט** ,המקבילה לדיוק **0.5°C**. דגמים מתקדמים יותר כמו ה- LM75B מציעים רזולוציה של **11 ביט** , מה שמספק דיוק גבוה יותר של **0.125°C** (למרות שהדיוק המוחלט עדיין תלוי בטווח הטמפרטורה).
- **צריכת אנרגיה :** יתרון משמעותי של ה- LM75 הוא יעילותו האנרגטית. הוא צורך זרם נמוך מאוד במצב פעולה, וכולל **מצב כיבוי (Shutdown mode)** בו הוא צורך זרם של מיקרו-אמפרים בודדים בלבד (בסביבות 4µA), מה שהופך אותו לאידיאלי ליישומים המופעלים על סוללות.
- **מתח עבודה:** החיישן יכול לפעול במתח שנע בין **2.7V ל- 5.5V**, מה שהופך אותו לתואם למגוון רחב של מיקרו-בקרים, כולל ארדואינו (5V) ו- (3.3V) ESP32.

עקרון הפעולה והתקשורת הדיגיטלית (I2C): (כפי שהוסבר בתחילת הספר)

ה- LM75 מתקשר עם המיקרו-בקר באמצעות פרוטוקול , (Inter-Integrated Circuit) **I2C** הידוע גם בשם (Two-Wire Interface) "TWI". פרוטוקול זה משתמש בשני קווים בלבד לתקשורת:

- **SDA (Serial Data Line)** : קו דו-כיווני להעברת נתונים.
- **SCL (Serial Clock Line):** קו שעון המסנכרן את התקשורת בין המיקרו-בקר לחיישן.

לכל רכיב I2C יש כתובת ייחודית של **7ביט.** ה- LM75-מציע שלוש רגלי כתובת (A0, A1, A2) שניתן לחבר ל VCC-או ל- GND, מה שמאפשר להגדיר עד **שמונה חיישני LM75 שונים** על אותו אוטובוס I2C, ללא הפרעות.

כיצד זה עובד?

המיקרו-בקר (המאסטר) שולח בקשה לקריאת נתונים לכתובת הספציפית של ה LM75-(העבד). החיישן מגיב בנתונים המאוחסנים בזיכרון הפנימי שלו, שם הטמפרטורה נשמרת באופן רציף לאחר המרה.

פונקציונליות "תרמוסטט-(Over-temperature):

אחת התכונות המיוחדות והשימושיות של ה- LM75 היא יכולתו לפעול כתרמוסטט דיגיטלי עצמאי. לחיישן יש יציאה ייעודית, **OS** (Over-temperature Shutdown), שיכולה לשמש בשני מצבים:

- **מצב השוואה :(Comparator Mode)** המיקרו-בקר יכול להגדיר שני ספי טמפרטורה בחיישן עצמו:
  - **TOS (Temperature Over-limit) :** טמפרטורה מקסימלית. כאשר הטמפרטורה עולה מעל ערך זה, יציאת ה OS-מפעילה פלט דיגיטלי.
  - **THYST (Hysteresis) :** טמפרטורה שבה הפלט חוזר למצב רגיל. הדבר מונע הפעלה וכיבוי חוזרים ונשנים (נדנוד) של היציאה כאשר הטמפרטורה נמצאת קרוב לסף.
- **מצב פסיקה (Interrupt Mode):** במצב זה, יציאת ה OS-יכולה לייצר פסיקה (interrupt) למיקרו-בקר, להתריע בפניו על חריגה מהטמפרטורה, ללא צורך בסקרים מתמידים של החיישן.

פונקציה זו שימושית במיוחד לבניית מערכות פשוטות של בקרת טמפרטורה, כמו מפוח שמופעל אוטומטית כאשר הטמפרטורה עולה על סף מסוים.

שלבי חיבור בסיסיים:

1. **VCC** (מתח) ו-**GND** (הארקה): חיבור למקור מתח של 5V או 3.3V.
2. **SDA** ו-**SCL**: חיבור לפיני ה-I2C המתאימים בארדואינו (ברוב הלוחות: A4 ו-A5 בהתאמה) או ל-ESP32.
3. **נגדי פול-אפ:** יש צורך בנגדי פול-אפ (בדרך כלל 4.7kΩ) על קווי ה-SDA ו-SCL כדי למשוך את הקווים למתח גבוה כאשר הם אינם פעילים. במודולים רבים, הנגדים הללו כבר מובנים על הלוח.

חיבור החומרה ב-ESP32:

החיבור הפיזי של חיישן LM75 ל- ESP32 הוא פשוט מאוד, כיוון ששניהם תואמים למתח עבודה של 3.3V.

1. **VCC**(מתח) של ה LM75-מתחבר לפין **3.3V** ב-ESP32.
2. **GND**(הארקה) של ה LM75-מתחבר לפין **GND**ב-ESP32.
3. **SDA**(קו נתונים) של ה LM75-מתחבר לפין **GPIO21**ב-ESP32.
4. **SCL**(קו שעון) של ה LM75-מתחבר לפין **GPIO22**ב-ESP32.

פיני **GPIO21** ו- **GPIO22** הם ברירת המחדל של I2C בלוחות ESP32 רבים, מה שמפשט את החיבור.

שלבי תכנות (פסאודו קוד):

1. **הכללת ספרייה:** השתמש בספרייה ייעודית לחיישן (למשל, Adafruit LM75) כדי לפשט את התקשורת.
2. **אתחול:** בפונקציית setup(), אתחל את החיישן.
3. **קריאת נתונים:** בפונקציית loop(), קרא את ערך הטמפרטורה מהחיישן. הספרייה כבר תמיר את הערך הדיגיטלי לטמפרטורה בצלזיוס.
4. **הצגת הנתונים:** הדפס את הנתונים לצג טורי (Serial Monitor) או לצג LCD.

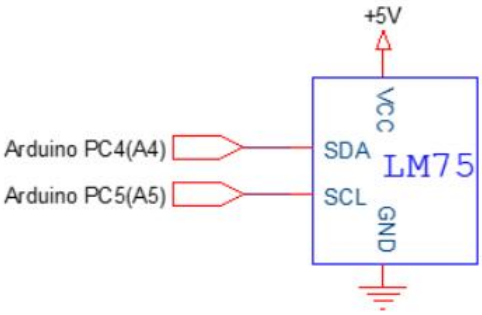
יישומים נפוצים:

השילוב הייחודי של דיוק סביר, צריכה נמוכה וממשק דיגיטלי פשוט הפך את ה-LM75 לבחירה פופולרית במגוון רחב של תחומים:

- **מערכות קירור:** בקרת טמפרטורה בציוד אלקטרוני, שרתים ומחשבים כדי למנוע התחממות יתר.
- **טרמוסטטים:** בניית מערכות בקרת אקלים פשוטות לבית או למשרד.
- **מכשירים ניידים:** שימוש במכשירים המופעלים על סוללות, כמו שעונים חכמים או מערכות ניטור אישיות.
- **מדידת טמפרטורה כללית:** יישומים מדעיים, תעשייתיים וחינוכיים שבהם נדרשת מדידת טמפרטורה אמינה ולא יקרה.

לסיכום, ה-LM75 מספק פתרון אלגנטי וחסכוני למדידת טמפרטורה דיגיטלית. הוא משחרר את המיקרו-בקר ממשימת המרת האות האנלוגי, מציע גמישות בשימוש בזכות ממשק ה-I2C ותכונות ה"תרמוסטט" המובנות, מה שהופך אותו לרכיב יסוד בפרויקטים רבים של IoT ואלקטרוניקה.

נראות ומאפיינים עיקריים של החיישן:



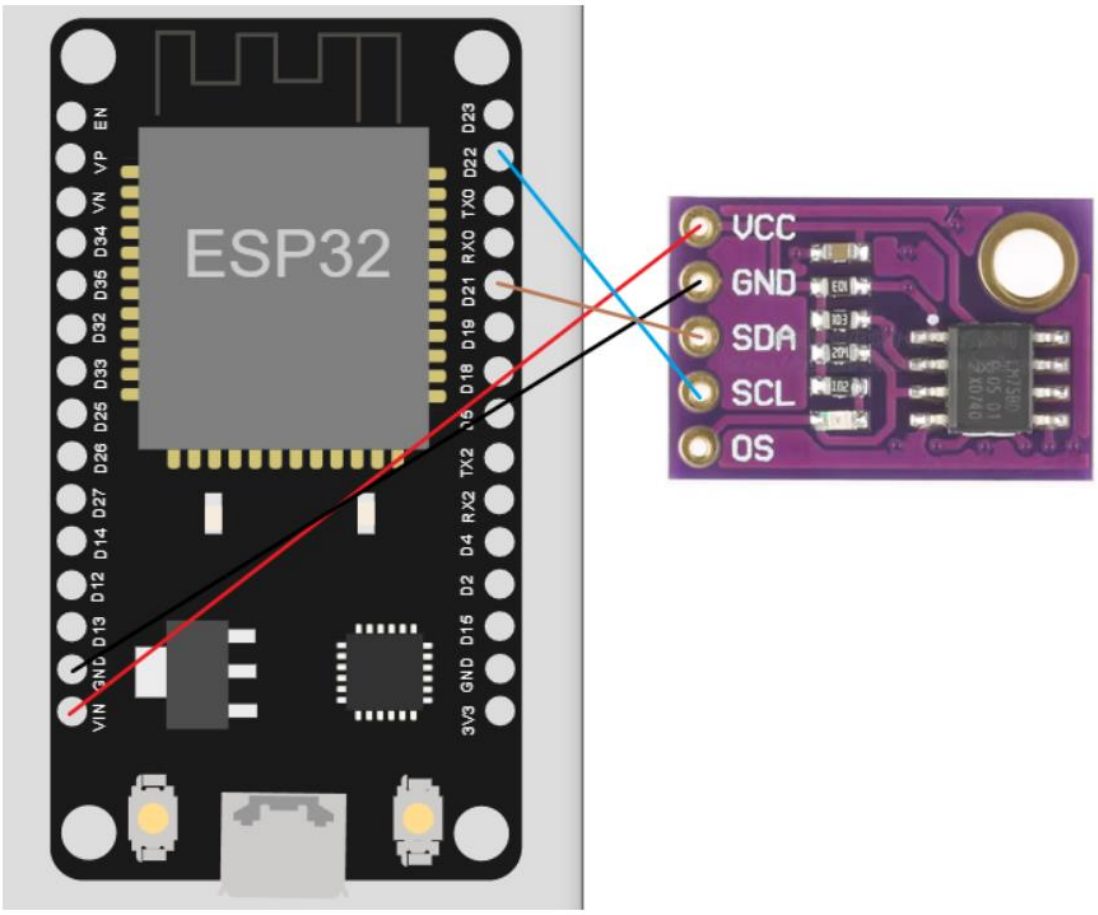
הסבר מדוע החיישן מחובר באמצעות הפרוטוקול I2C

I2C הוא פרוטוקול תקשורת שמאפשר לחבר התקנים שונים ללוח הארדואינו באופן פשוט יחסית. במקרה שלנו, החיישן **LM75** מחובר דרך פיני ה- SCL ו- SDA של הלוח. פינים אלה משמשים לתקשורת I2C דרך הפינים האלה, הארדואינו/ESP32 והחיישן "מדברים", באמצעות פרוטוקול I2C.

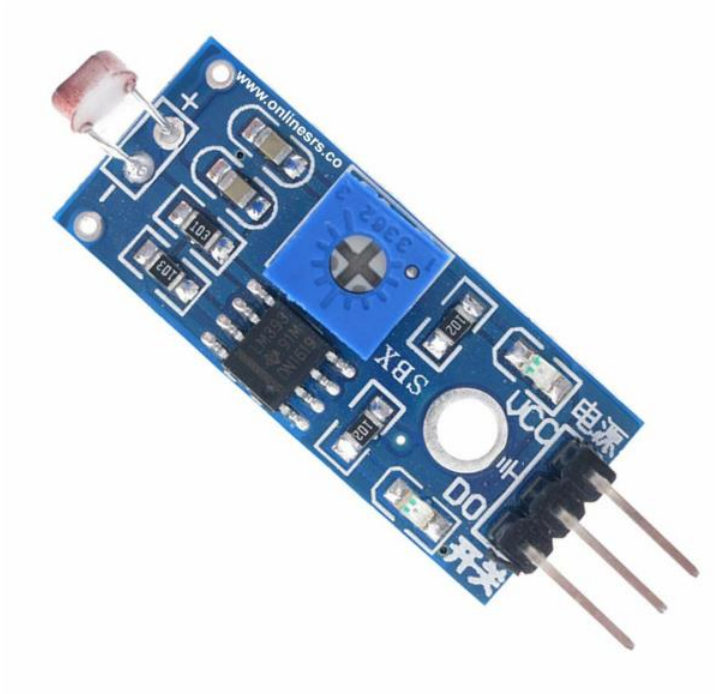
כשאנחנו רוצים לקרוא את הטמפרטורה, הארדואינו/ESP32 שולח בקשה לחיישן דרך פיני ה- I2C. החיישן קורא את הטמפרטורה, ושולח חזרה את הערך דרך אותם פינים. כך אנחנו יכולים לקרוא את הטמפרטורה בקלות רבה, ללא צורך במעגל מורכב.

I2C מאפשר לנו לחבר מספר התקנים על אותם פינים SCL ו- SDA, ולתקשר איתם באופן עצמאי

נראה חיבור של החיישן LM75 אל מיקרו בקר ESP32:



מודול חיישן האור KY–018



מודול חיישן האור (KY–018 או "LDR Module") הוא רכיב אידיאלי למדידת עוצמת אור ובקרת תאורה אוטומטית באמצעות ה -ESP32. המודול משלב את החיישן הבסיסי עם רכיבי בקרה, מה שמאפשר שתי דרכי עבודה: קריאה מדויקת או קריאה בינארית פשוטה.

1.המבנה והרכיבים העיקריים:

המודול בנוי סביב שלושה רכיבים מרכזיים:

1. **נגד תלוי-אור: (LDR)** זהו החיישן הראשי שאחראי על המדידה. כשהאור הפוגע בו גדל, ההתנגדות שלו יורדת. המודול מכיל מעגל פנימי שממיר את שינוי ההתנגדות הזה לשינוי מתח.
2. **קומפרטור מתח: (LM393)** זהו הצ'יפ האלקטרוני הראשי. תפקידו הוא להשוות את המתח המשתנה של ה- LDR מול **מתח ייחוס קבוע** (הסף). השוואה זו מניבה את הפלט הדיגיטלי (D0).
3. **פוטנציומטר כחול:** זהו נגד משתנה שמאפשר למשתמש **לכייל ידנית** את מתח הייחוס הקבוע של ה- LM393 ובכך לקבוע באופן חומרתי את סף ההדלקה/כיבוי של היציאה הדיגיטלית.

2. דרכי השימוש במודול (היציאות):

הגמישות של המודול נובעת משתי יציאות הפלט שלו:

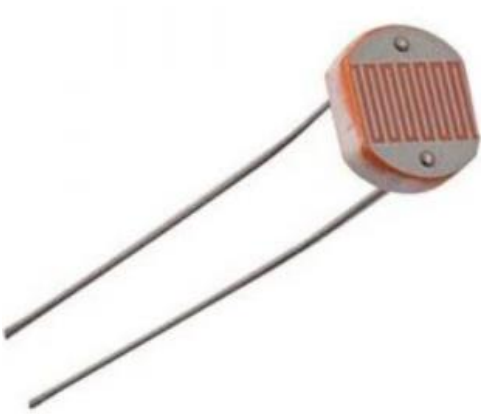
א. יציאה אנלוגית - (A0) למדידה מדויקת

- **תפקיד**: פין זה מספק את המתח המשתנה המגיע ישירות ממחלק המתח הפנימי של ה-LDR.
- **היתרון**: הוא מעביר את כל טווח האור באופן רציף (ערכים בטווח של 0 עד 4095 ב-ESP32). שיטה זו מאפשרת לך לקבוע את סף ההדלקה **בתוך הקוד** שלך, מה שנותן שליטה ודיוק גבוהים יותר.
- **חיבור**: חבר את A0 לפין **ADC** של ה- ESP32 (למשל, GPIO34).

ב. יציאה דיגיטלית - (D0) להחלטה בינארית

- **תפקיד**: פין זה הוא הפלט של ה -LM393. הוא מספק רק שתי אפשרויות: HIGH או LOW.
- **היתרון**: השיטה היא "הכנס והפעל". ה- LM393 מבצע את ההחלטה האם חשוך או מואר עבורך, בהתאם לכיול שקבעת באמצעות **הפוטנציומטר הכחול**.
- **חיבור**: חבר את D0 לפין **GPIO דיגיטלי** רגיל ב-ESP32.

חיישן אור - Light Dependant Resistor - LDR:

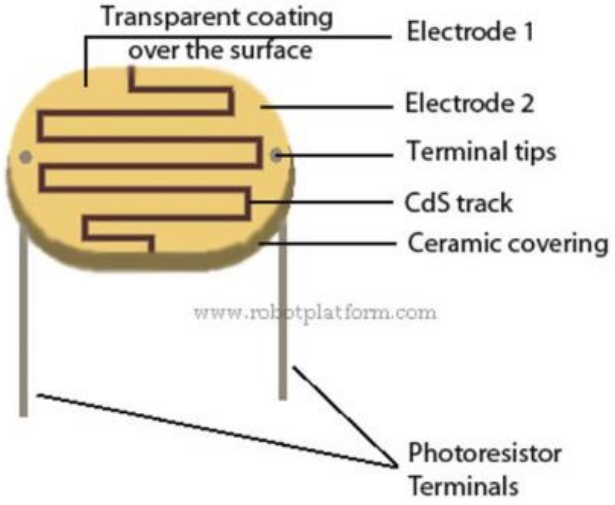


חיישן היכול למדוד את עוצמת האור הנמצאת בסביבתו. שמו באנגלית מסגיר את אופן פעולתו - נגד תלוי אור, כלומר נגד שמשנה את התנגדותו בתלות באור אליו הוא נחשף.

מבנה חיישן אור:

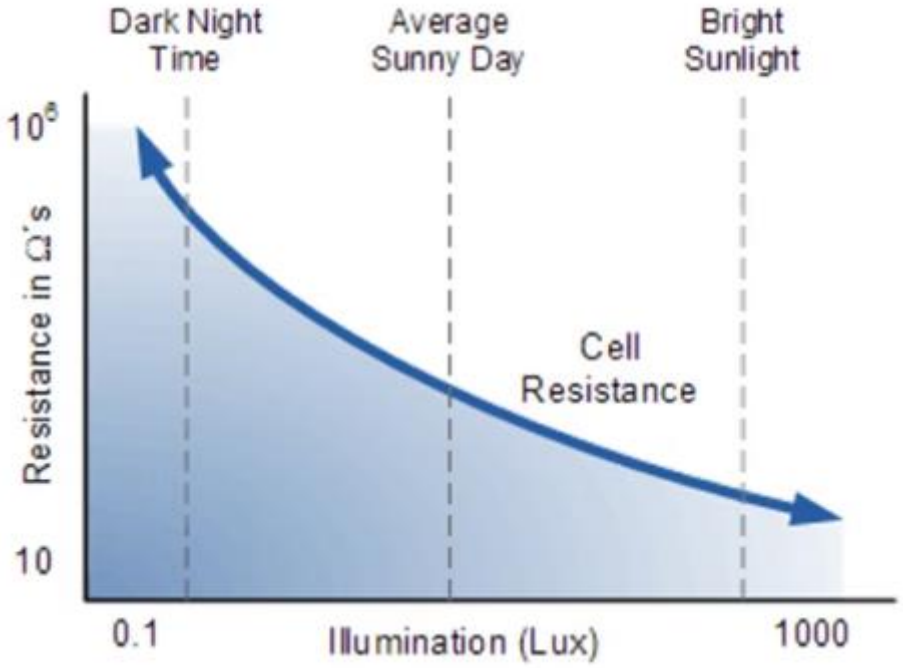
החיישן מורכב מדסקית עגולה העשויה חומר קרמי. הדסקית מחולקת לשני אזורים המופרדים ביניהם על ידי חומר הנקרא קדמיום סולפיד CdS . חומר זה אחראי להתנהגות החשמלית של החיישן. משני החצאי הקרמיים יוצאים גם שתי קטבים מוליכים המתחברים את המערכת.

- **בחושך (התנגדות גבוהה)**: במצב זה, האלקטרונים בחומר ה- CdSקשורים חזק לאטומים שלהם. כתוצאה מכך, ישנם מעט מאוד **נושאי מטען חופשיים** שיכולים לשאת זרם חשמלי, ולכן החיישן מציג **התנגדות גבוהה מאוד** (עשרות קילו-אוהם עד מגה-אוהם).
- **באור (התנגדות נמוכה)**: כאשר אור (המורכב מפוטונים) פוגע בשטח הפנים של ה -CdS, הפוטונים מעבירים אנרגיה לאלקטרונים **ומשחררים אותם** ממבנה האטום. שחרור זה יוצר מספר רב של **אלקטרונים חופשיים**.
- **תוצאה חשמלית**: ככל שנוצרים יותר אלקטרונים חופשיים, המוליכות של החומר גדלה, וההתנגדות החשמלית של החיישן **יורדת באופן דרסטי**.



אופיין שינוי התנגדות:

החיישן משנה את התנגדותו בהתאם לעוצמת האור אליה הוא נחשף. כאשר החיישן בחושך מוחלט אז התנגדותו מאוד גבוהה (ניתן לראות קטע מדף נתוני היצרן), ואילו כאשר הוא מואר התנגדותו יורדת באופן מהיר. עוצמת תאורה נמדדת ביחידות מידה הנקראות LUX.



אופן המדידה של מודול חיישן האור KY–018 מתבסס על עקרון **מחלק המתח**. הליבה של החיישן היא **נגד תלוי-אור (LDR)**, אשר התנגדותו משתנה באופן הפוך לעוצמת האור. המודול מחובר ל- ESP32 דרך אספקת מתח של (VCC) **3.3V** המודול עצמו **כולל בתוכו נגד קבוע** בטור עם ה- LDR כדי ליצור מחלק מתח.

כתוצאה משינוי ההתנגדות של ה -LDR, המתח בנקודה המשותפת (המסומנת A0) משתנה. בקר ה- ESP32 מודד את המתח המשתנה הזה באמצעות הפין האנלוגי ,(A0) ומתרגם אותו לערך מספרי בטווח של **0 עד 4095**, בהתאם לרזולוציית 12bit של הממיר האנלוגי-לדיגיטלי (ADC) שלו.

מסך 1.8– TFT (ST7735)

זהו המסך בו השתמשתי על מנת להציג את הראדאר בפרויקט שלי. מסך קטן וקומפקטי שיוצר תמונה ברורה וטובה עבור הראדאר. אשר מציג את המרחק החל מ 0 ועד 100 סנטימטרים. בנוסף כפי שניתן לראות את הקו שמסתובב 180 מעלות החל מצידו הימני/השמאלי של המסך ועד לצידו השני ומצייר נקודה בהתאם למרחק ככל שהאובייקט קרוב יותר נראה שצבעה של הנקודה הוא אדום, וככל שהאובייקט רחוק יותר הנקודה תהיה בצבע צהוב.

1 סוג המסך:

- דגם : ST7735.
- גודל :1.8 אינץ'.
- רזולוציה : 128×160 פיקסלים.
- צבעים : צבעוני (RGB565) .
- ממשק : SPI (Serial Peripheral Interface) מהיר ומדויק.

2 חיבורי פינים(SPI):

המסך מתקשר עם ה- ESP32 דרך SPI. בפועל בקוד שלי:

- GPIO 23 – MOSI (Master Out Slave In) (נתונים מה- ESP32 למסך).
- GPIO 18 – SCLK (Clock) (שעון SPI).
- GPIO 5 – CS (Chip Select) (בוחר איזה התקן SPI פעיל*.
- GPIO 2 – DC (Data/Command) (מספר למסר "פקודה או נתון").
- GPIO 4 – RESET (אתחול המסך).
- GND / VCC - מתח והארקה.

הערה: MOSI ו SCLK - משותפים אם יש יותר ממסך אחד CS ו- DC - חייבים להיות שונים לכל מסך.

3 Backlight (תאורת רקע):

- לרוב המסכים יש **LED backlight** פנימי.
- חיבור פיזי: LED / +LED / BL - .
- אם מחברים ל- Backlight → 5V דולק.
- ניתן לכבות או לשלוט בעוצמה דרך **טרנזיסטור PWM** / כדי לחסוך צריכת חשמל.
- כיבוי Backlight חיסכון משמעותי בזרם.(50–300 mA) .

4 ספריות ותפקודים בקוד שלי:

- הספרייה Ucglib שולחת פקודות SPI למסך.
- פונקציות עיקריות בקוד:
  - ucg.begin() - אתחול המסך.
  - setRotate90() - סיבוב המסך ל־אופקי.
  - setColor(...) - הגדרת צבע לציור.
  - drawLine, drawBox, drawDisc, drawCircle - ציור צורות.
  - setFont(...), setPrintPos(...), print(...) - כתיבת טקסט.
  - cls() - ניקוי המסך באמצעות ריבועים שחורים.

5 זרם ומתח

- VCC: רוב המודולים מקבלים 5V חלק תומכים גם ב־3.3V
- צריכה טיפוסית:
  - ללא backlight: 10–60 mA
  - עם backlight: 50–300 mA

מה המסך מציג בקוד שלי:

- **רקע גרפי:** עיגולים, קווים, רקע ירוק-שחור (גרדיאנט)
- **טקסט:** שם הפרויקט, "Mini Radar" מרחקים בקנה מידה (25cm, 50cm…)
- **נקודות:** מיקום אובייקטים לפי החיישן האולטראסוניק (>100cm אדום, <100cm צהוב)
- **קווים:** קו סריקה שמסתובב עם הסרוו, כמו קרן רדאר.

בקיצור, המסך הוא **הממשק הגרפי של הרדאר** – מציג מידע בזמן אמת בצורה צבעונית.

הסבר על תצוגת הראדאר:

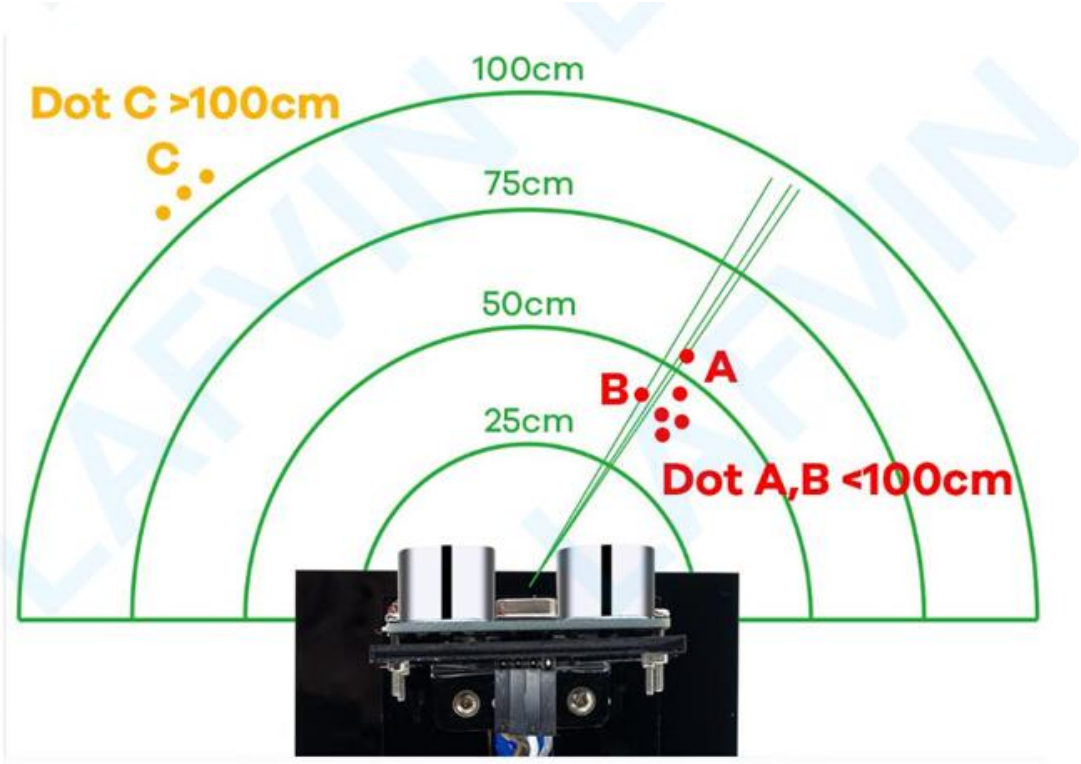
איך למעשה המערכת עובדת מה אנחנו רואים על המסך?

אז נתחיל בזה שהרובוט השומר שלנו צריך לשמור על סביבה שנסרקת 180 מעלות.

הפעולה של הסריקה עצמה מתבצעת באמצעות שני רכיבים חשובים מאוד במערכת הראדאר שלנו. הרכיב הראשון זהו המנוע סרוו שבכלל מאפשר לנו את כל העניין של התזוז של חצי הסיבוב החל מ0 מעלות ועד הגעתו ל180 מעלות. הרכיב השני שלנו שלמעשה מבצע את המדידה זהו החיישן האולטראסוניק שמבצע בדיקה של האובייקטים בסביבה בתווך של 100 פלוס סנטימטרים ומחזיר לנו "מידע" שלמעשה מתאפיין בנקודה על הראדאר.

כל המערכת מוצגת באופן חד וברור על המסך שלי ואנו יכולים לדעת האם האובייקט קרוב אלינו או רחוק מאיתנו, כאשר האובייקט נמצא בתווך בין- 0 ל- 100 סנטימטרים נראה שהנקודה היא בצבע אדום. כאשר האובייקט נמצא בתווך שמעל ל 100 סנטימטרים אז הנקודה תהיה בצבע צהוב.

נראה תמונה שממחישה לנו באופן ברור את מה שמתרחש במסך בזמן פעולת מערכת הראדאר:



כמו שאמרתי קודם לכן, ניתן לראות מהתמונה שכל עוד אנו נמצאים בתחום בין 0-100 סנטימטרים צבע הנקודות האו אדום וברגע שאנו עוברים תחום זה צבע הנקודות הוא צהוב.

מנוע SERVO

הסבר כללי על המנוע סרוו:

מנוע סרוו הוא מנוע שבו יש את המרכיבים הבאים:

- **מנוע זרם ישר (DC Motor)**
- **מערכת תמסורת פנימית** של גלגלי שיניים
- **בקרה אלקטרונית** על מיקום המנוע.

מנוע SERVO לא מסתובב בצורה חופשית כמו מנועי DC, אלא נע עם פי זווית – לרוב בין 0 ל-180 מעלות (אם כי יש מנועי SERVO שמסתובבים גם עד 360 מעלות).

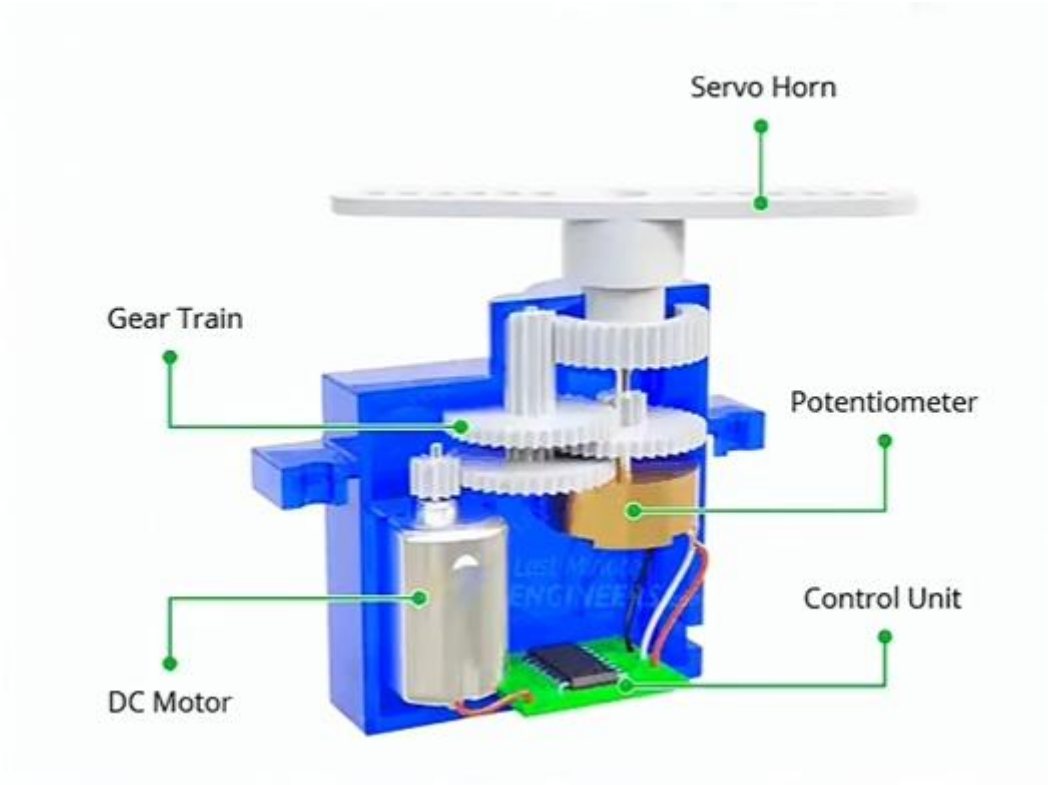
מנועי SERVO נפוצים מאוד ומשמשים באפליקציות רבות. הסיבות לכך הן: קלות השליטה על **מנועי SERVO**, דרישות האנרגיה הנמוכות) **יעילות**, (**הכוח החזק יחסית** של המנוע, **רמת המתח** שמשתמשים להפעלתו, **הגודל, המשקל והמחיר** הנמוכים שלו.

מנועי SERVO פועלים **בתוך מעגל סגור**, כלומר יש להם מערכת בקרה על מיקום המנוע והם יכולים לתקן הפרשים מהמיקום הרצוי.

האיור הבא מתאר מנוע SERVO שניתן להשיג באינטרנט במחירים נמוכים יחסית ואת החיבורים שלו **המנוע נקרא: SG90.**

נראה כיצד נראה המנוע SERVO:

מנוע SERVO מכיל מנוע DC קטן המחובר לציר היציאה דרך גלגלי השיניים. על הציר מחובר פוטנציומטר המחזיר משוב למערכת הבקרה על המיקום הנוכחי בו הוא נמצא.



עקרון פעולה של מנוע SERVO:

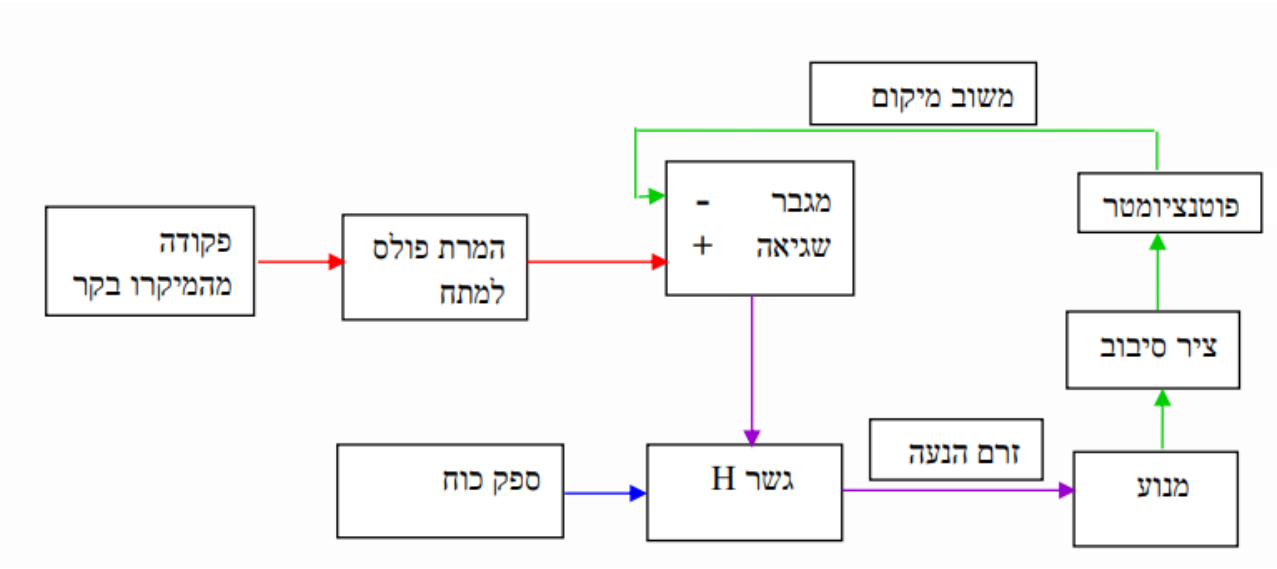
מנוע SERVO מופעל בדרך כלל על ידי **מיקום בקר** הגורם לו להגיע למיקום רצוי. פקודת התנועה מתקבלת בקר מתורגמת למתח **PWM** שמועבר ליחידת הבקרה הנמצאת בתוך המנוע. ערך זה הוא **הערך הרצוי** של המנוע.

הערך המצוי של המנוע, נמדד בעזרת **פוטנציומטר** המחובר לציר המניע של המנוע (הציר שיוצא מחלקו העליון של המנוע ואליו מחברים את המנגנון שיש להניע). סיבוב הפוטנציומטר גורם לשינוי ההתנגדות שלו ולשימושים של שינוי מתח המתח על רגלו של הפוטנציומטר. מתח זה הוא **הערך המצוי** שמשמש כמשוב ל'בקר הסגור'. כך נוצר **שי פרש**)נקרא **מתח שגיאה** (בין המתח המצוי ולבין המתח הרצוי. מנוע ה **DC**-הנמצא בתוך ה-SERVO, ימשיך לנוע בכיוון המתאים.

סיבוב מנוע ה- DC מתבצע יחד עם **סיבוב גלגלי השיניים** -שמהווים תמסורת מפחיתה) **מאטה את מהירות הסיבוב ומגדילה את מומנט הכוח** .(גלגלי השיניים מניעים את ציר היציאה ואת הפוטנציומטר. ברגע שהפרש המתחים יהיה **קטן מערך סף** (**קרוב ל-0**), יחידת הבקרה של המנוע **תנתק מתח** הנחוץ למנוע ה- DC והוא יחדל לנוע. עצירת המנוע תהיה **בדיוק** בזווית שנשלחה מהתוכנית ב'מיקרו בקר'.

אם התוכנית שולחת למנוע פקודה לנוע לזווית מעבר ל-180 מעלות, המנוע יתחיל לרעוד מכיון שהוא ינסה לסגור את השגיאה בין **הערך הרצוי למצוי** ולא יצליח. כמו כן יש ציר יציאה שמסומם מכני מהמנוע לעבור את זווית הסיבוב של 180 מעלות.

נראה כעת כיצד נראית מערכת הבקרה של מנוע SERVO:



מעולה, הנה גם הטקסט מהתמונה השלישית, מוכן להעתקה לוורד:

בצד שמאל מגיע אות הפקודה מהמיקרו בקר על הזווית הרצויה שנקרא **הערך הרצוי**. זהו פולס עם **Duty Cycle** כאשר הזמן בין ה-**ON** ל-**OFF** קובע את הזווית הרצויה. אות זה נקרא גם **מיקום מטרה**.

מעגל "**מרת פולס למתח**" מעביר למגבר השגיאה מתח שמוצא מה-**Duty Cycle** של אות הפקודה.

מגבר השגיאה מקבל מהפוטנציומטר מתח היחסי למיקום ציר המנוע, מתח זה הוא **המיקום הנוכחי של המנוע** ונקרא **ערך מצוי**.

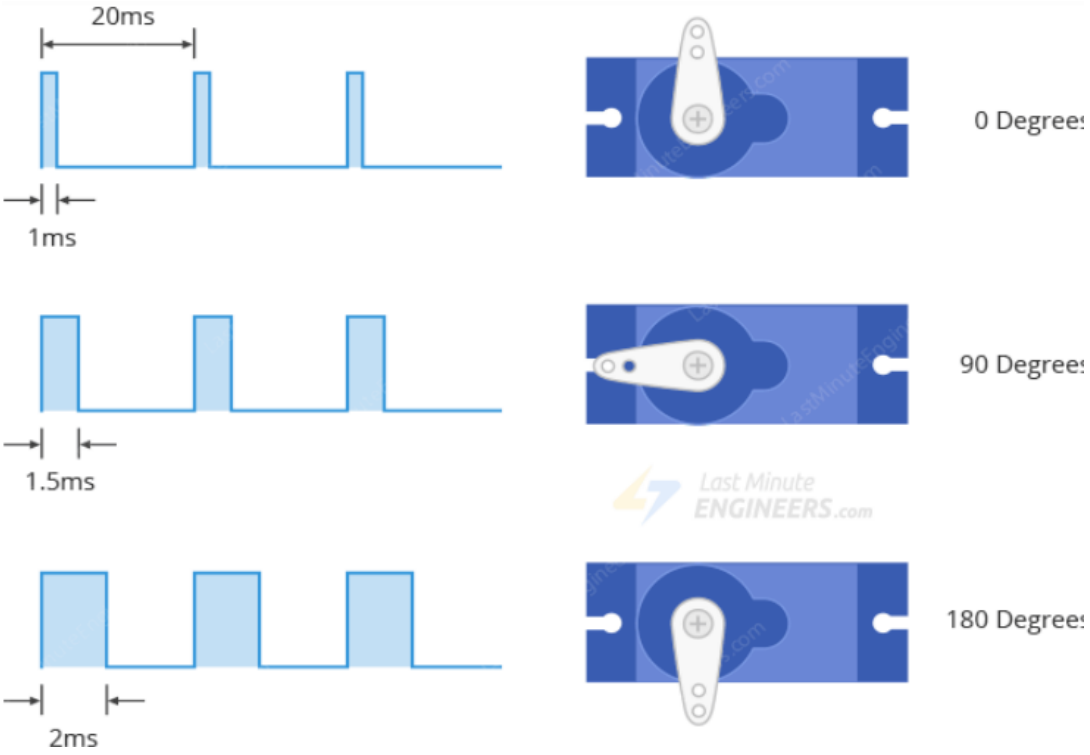
מגבר השגיאה מגביר את **הפרש המתח** בין הרצוי למתח המצוי. יציאת המגבר מחוברת לגשר **H-Bridge** שמעביר זרם המסובב את המנוע.

כאשר המנוע מסתובב הוא **מסובב את הפוטנציומטר** שמעביר שוב מתח שלילי אל **הכניסה המהפכת** (המינוס של המגבר) של המגבר. ככל שהמנוע מסתובב וציר המנוע מתקרב אל **הזווית הרצויה הפרש המתח הולך וקטן**. כאשר מגיעים לזווית הרצויה, **הפרש המתח למגבר הוא 0** ואין איננו מוציא פקודה לסיבוב המנוע והמנוע **עוצר**.

כיצד עובד מנוע סרוו?

עבודה עם פולסים:

אפשר לשלוט על מנוע SERVO על ידי **שליחת סדרה של פולסים אליו.** מנוע SERVO טיפוסי מחכה לפולס כל 20 אלפיות השנייה (כלומר, **תדר האות שנשלח למיקרו בקר צריך להיות 50HZ. רוחב הדופק קובע את מיקום מנוע ה- SERVO.** האיור הבא מראה את הזוויות של המנוע עבור רוחבי **Duty Cycles** שונים.



מיקום המנוע כתלות ברוחב הדופק של ה- DUTY CYCLE.

מתיאור רוחב הפולס:

- **פולס קצר** ברוחב של **1מילישנייה** או פחות מסובב את מנוע הסרוו ל **0-מעלות**.
- **פולס ברוחב של 1.5 אלפיות השנייה** מסובב את מנוע הסרוו ל **90-מעלות**.
- **פולס ברוחב של 2 אלפיות השנייה** בערך מסובב את מנוע הסרוו ל **180-מעלות**.

פולסים בטווח של 1 msעד 2 msיסובבו את הסרוו למיקום פרופורציונלי לרוחב הפעימה.

יש מנועי סרוו המסוגלים להסתובב **360 מעלות**. רוחב הפולס במנועים אלו ייתן את הזוויות הבאות:

- **0.5 ms** נותן זווית של **0 מעלות**.
- **1 ms** נותן זווית של **45 מעלות**.
- **1.5 ms** נותן זווית של **90 מעלות**.
- **2 ms** נותן זווית של **135 מעלות**.
- **2.5 ms** נותן זווית של **180 מעלות**.

הערה: אין **תאום מדויק** בין רוחב הפולסים למיקום ולכן יהיה עלינו לשנות את התכנות שלנו כדי להתאים לטווח של מנוע ה-SERVO שלנו. כמו כן, **משך/רוחב הדופק** יכול להשתנות בין יצרני מנועי SERVO שונים; לדוגמה, זה יכול להיות **0.5ms עד 2.5 ms** עבור 180 מעלות ו- **1ms** עבור 0 מעלות.

הטבלה הבאה מאפיינת את המנוע SERVO:

אנגלית	עברית	תיאור
Dimensions	מידות	23.2 x 12.5 x 22 mm
Weight	משקל	9 g
Operating Speed	פעולה מהירות	0.12sec/60degrees (4.8V) 0.10sec/60degrees (6V)
Stall Torque	מומנט מעכב	1.3kg.cm/18.09oz.in (4.8V) 1.5kg.cm/20.86oz.in(6V)
Operating Voltage	מתח הפעלה	4.8V~6V
Control System	מערכת בקרה	Analog
Direction	כיוון	CCW
Operating Angle	זווית פעולה	120degrees
Required Pulse	הפולס הנדרש	900us-2100us
Bearing Type	סוג מיסב	None
Gear Type	סוג גלגל השיניים	Metal
Motor Type	סוג המנוע	Plastic
Connector Wire Length	אורך חוטי החיבור	20 cm

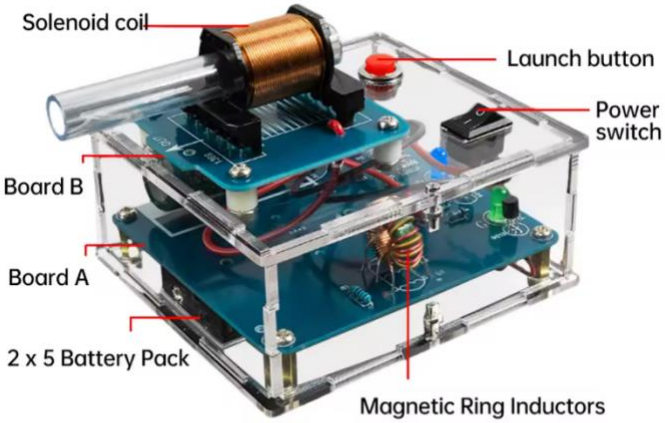
עבודה עם מנוע סרוו ומיקרו בקר ESP32:

העבודה עם מיקרו בקר ESP32 בסביבת AEDUINO IDE דומה מאוד לעבודה עם כרטיס מיקרו בקר ארדואינו (אונו, נאנו, מגה וכו').

התכנות כמעט זהה לתכנות עבור הארדואינו.

הערה חשובה: יש לשים לב כי המתח של מנוע ה-SERVO הוא בין 4.8V עד 6V ולא את מתח ה- 3.3V שברכיבים.

רובה אלקטרומגנטי (Electromagnetic gun)



תיאור כללי:

המערכת המתוארת בתמונה היא **רובה אלקטרומגנטי חד-שלבי** הפועל באמצעות שדה מגנטי שנוצר ע"י סליל (Solenoid Coil), המאיץ גוף מתכתי דרך קנה פלסטי. הרובה נשלט ידנית באמצעות כפתור שיגור, ומקבל את אספקת החשמל ממארז סוללות פנימי.

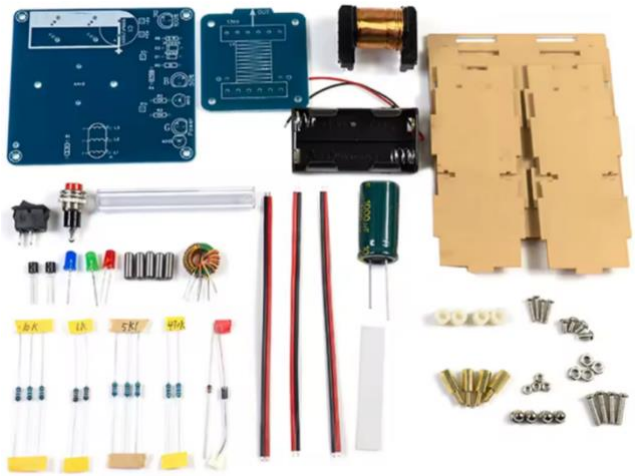
מטרת המערכת:

שיגור גופים מתכתיים גליליים קטנים (הנקראים כאן "Bombs") באמצעות כוח מגנטי – לצורכי הדגמה פיזיקלית, חינוכית או ניסיונית.

טבלת הסבר של התמונה:

תיאור	רכיב
סליל נחושת דרכו עובר זרם גבוה. יוצר שדה מגנטי חזק ברגע ההפעלה ומושך את הגוף המתכתי לתוכו במהירות גבוהה.	**Solenoid Coil סליל אלקטרומגנטי**
כפתור אדום המפעיל את המערכת ומזרים זרם לסליל. זהו רכיב השליטה הידני לשיגור.	**Launch Button כפתור שיגור**
מתג הפעלה/כיבוי ראשי של המערכת. מנתק ומחבר את אספקת המתח לכלל המעגלים.	**Power Switch מתג הפעלה**
מכיל את הסליל, קונקטורים, כפתור ההפעלה, נורות בקרה, והחיבורים הפיזיים.	**Board B לוח עליון**
מכיל את הרכיבים החשמליים המרכזיים – קבלים, נגדים, מייצבים, לולאות השראה וכו'. אחראי על ויסות המתח וההגנה על המערכת.	**Board A לוח תחתון**
שתי סדרות של 5 סוללות המספקות מתח כולל גבוה (לרוב 18–24 וולט) אחראי על אספקת האנרגיה לסליל.	**2x5 Battery Pack מארז סוללות**
משמשים להחלקת הזרם, הפחתת רעשים אלקטרומגנטיים, ואחסון אנרגיה זמנית במעגל.	**Magnetic Ring Inductors חישוקי השראה מגנטיים**
גופים גליליים ממתכת אשר מוכנסים לקנה. בזמן שיגור הם נמשכים דרך הסליל ויוצאים במהירות מהקצה השני.	**Bombs כדורים מתכתיים**

תמונה של הרכיבים ברובה האלקטרומגנטי:



עקרון פעולה:

1. המשתמש מפעיל את מתג ההפעלה.
2. בלחיצה על כפתור השיגור, הזרם מוזרם מהסוללות לסולנואיד.
3. נוצר שדה מגנטי חזק בתוך הסליל.
4. גוף המתכת נמשך פנימה ומואץ לאורך הקנה.
5. לאחר מכן, הגוף יוצא במהירות מהקצה – ללא חלקים נעים מכניים.

הטבלה הבאה מציגה את כל רכיבי הערכה לרובה האלקטרומגנטי כפי שמופיעים בתמונה למעלה, כולל תיאור תפקודי של כל רכיב במערכת:

תיאור תפקודי	רכיב
לוח המעגל המרכזי שעליו מולחמים כל רכיבי האלקטרוניקה, כולל נגדים, קבלים וסלילים.	**לוח PCB ראשי**
לוח עזר – לרוב משמש לקישוריות בין סליל הירי לבין הלוח הראשי או כקונקטור.	**לוח PCB משני**
סליל נחושת דרכו עובר זרם ליצירת שדה מגנטי חזק המאיץ את הקליע לאורך הקנה.	**סולנואיד (Solenoid Coil)**
מחזיק שני סטים של 5 סוללות – AA מספק מתח גבוה) לרוב 18 (V–24Vלהפעלת הסולנואיד.	**מחזיק סוללות (2x5)**
חלקי גוף המערכת –חיתוך לייזר מחומר מוקשה / MDF) אקריליק (להרכבת המארז.	**פלטות מארז (Housing Panels)**
כפתור אדום להפעלת השיגור – מזרים זרם מיידי לסליל לצורך ירי.	**כפתור שיגור (Launch Button)**
מפסק ראשי להפעלת או כיבוי המתח הכללי של המערכת.	**מתג הפעלה (Power Switch)**
נורת חיווי – מדליקה כשהמערכת פעילה או מוכנה לשיגור.	**נורת לד**
משמש כמסילה לירי –דרכו עובר הקליע במהלך השיגור.	**צינור פלסטיק)קנה(**
גופים גליליים ממתכת אשר נמשכים אל תוך הסליל ויוצאים במהירות – משמשים להדגמה.	**קליעים מתכתיים (Bombs)**

רכיבי השראה מגנטיים לאגירת אנרגיה ולשיפור יציבות הזרם.	**טורואידים / סלילים טבעתיים**
משמש לאגירת אנרגיה קצרה והעברת זרם חזק מיידי בעת השיגור.	**קבל אלקטרוליטי**
רכיבים להגבלת זרם, קביעת מתח, או תיאום בין רכיבים שונים במעגל.	**נגדים (Resistors)**
חוטים אדומים/שחורים לקישוריות חשמלית בין הלוחות, הסוללות, הסולנואיד והכפתורים.	**חוטים ומוליכים**
חלקים מכניים להרכבת הלוחות, קיבוע הסוללה, בניית המסגרת וחיבור כללי של המערכת.	**ברגים, אומים ומרווחים**

יתרונות המערכת:

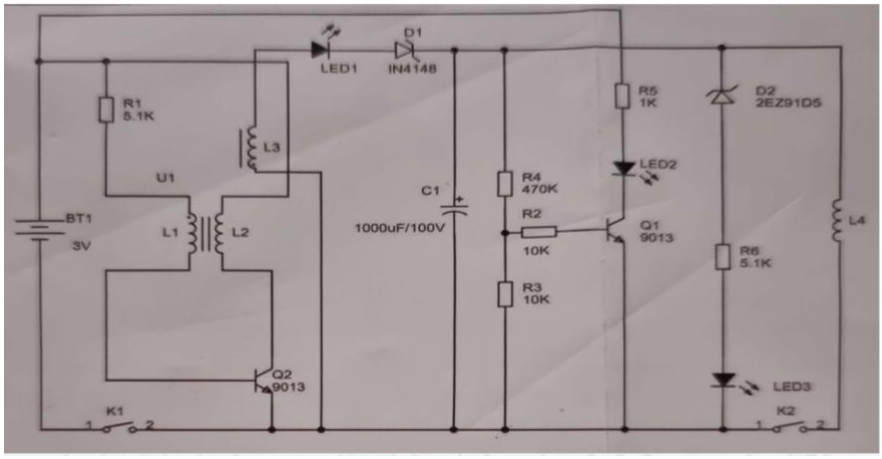
הסבר	יתרון
מדגים את עקרונות ההנעה המגנטית ללא צורך במכאניקה מסובכת.	**שימוש בטכנולוגיה אלקטרומגנטית מתקדמת**
מקטין בלאי, שקט יותר ובטוח יותר (יחסית).	**אין חלקים נעים מכאניים**
כפתור אחד לשיגור – מתאים ללמידה, הדגמה וניסויים.	**תפעול פשוט**
ניתן להפעיל את המערכת בכל מקום ולשאת אותה בקלות.	**גודל קומפקטי וניידות**
ניתן לשגר מספר פעמים עם אותן "פצצות."	**רב פעמיות**

חסרונות המערכת:

הסבר	חיסרון
בשל מגבלות מתח, סליל יחיד וגוף קל – העוצמה והמהירות יחסית נמוכות.	**עוצמת שיגור מוגבלת**
חלק גדול מהאנרגיה מתבזבז כחום ולא תורם להאצת הגוף.	**יעילות אנרגטית נמוכה**
שימוש רצוף עלול לגרום להתחממות יתר ולפגיעה במעגלים.	**חימום רכיבים**
ללא חיישני מיקום – אין ניתוק מדויק של השדה ולכן תיתכן האטה בשלב מסוים.	**אין בקרת תזמון מתקדמת**
אם קיים שלב טעינת קבלים, יידרש זמן בין שיגור לשיגור.	**טעינה איטית (אם משתמש בקבלים)**

רובה אלקטרומגנטי הוא מערכת מרתקת ומשולבת – מדגים פיזיקה, חשמל, אלקטרוניקה ומכאניקה ברמה בסיסית. הוא מהווה פלטפורמה חינוכית מצוינת להבנת עקרונות השראה מגנטית, בקרה, ואנרגיה. מבנהו הקומפקטי והשימוש הנגיש הופכים אותו לכלי לימוד ולמעבדה ניידת לכל דבר.

הסבר על מעגל חשמלי - רובה אלקטרומגנטי



הסבר כללי:

המעגל משתמש ברכיבים כמו קבלים, סלילים (L), טרנזיסטורים, דיודות, ונורות LED, כדי לטעון קבל במתח גבוה ואז לפרוק אותו דרך סליל אלקטרומגנטי לצורך יצירת דחף (למשל, שיגור פרויקטיל מתכתי קטן).

הסבר לפי אזורים:

אזור הספק:

BT1 -סוללה (3V)- מספקת את האנרגיה למעגל.
K1 -מתג הפעלה ראשי שמחבר את החשמל.

חלק טעינת הקבל:

C1 -קבל 1000 µF– 100V נטען בהדרגה דרך המעגל, ואוגר אנרגיה שתשוחרר בבת אחת לשיגור.
D1 -דיודה 1 – (N4148) מגינה על המעגל מזרם חוזר.
LED1 -מציין שהקבל נטען.
L1, L2, L3 -סלילים כנראה כחלק ממעגל מתנד/רזוננס שמעביר אנרגיה לקבל.

חלק בקרה:

Q1, Q2 -טרנזיסטורים מסוג 9013 – מפעילים או חוסמים את מעבר הזרם.
R1-R6 -נגדים – קובעים את זרימת הזרם לבסיס הטרנזיסטורים, וכך שולטים עליהם.
LED2 -חיווי נוסף למצב ההפעלה של השלב הבא.

חלק הפעלת סליל השיגור:

D2 -דיודה Zener או – SCR (2EZ91D5) מפעילה שחרור מהיר של האנרגיה מהקבל.
L4 -סליל השיגור – כאשר הקבל נפרק דרכו, נוצר שדה מגנטי חזק שיכול לדחוף אובייקט מתכתי קטן.
K2 -מתג הפעלה לשיגור.
LED3 -נדלק כאשר מתבצע השיגור.

טבלת רכיבים:

תפקיד	תיאור	רכיב
מקור מתח	סוללה V3	BT1
K1-הפעלה, K2 -שיגור	מתגים	K1, K2
מגברים/מתגים לבקרה	טרנזיסטורים 9013	Q1, Q2
אגירת אנרגיה חשמלית	קבל 1000 µF -100V	C1
הגנה מזרם חוזר	דיודה N41481	D1
פריקה פתאומית של הקבל	דיודהZener / SCR	D2
קביעת זרמים לבקרה	נגדים	R1-R6
מראה מצבי פעולה שונים	נוריות חיווי	LED1-LED3
ייצוב, תדר, שיגור	סלילים	L1-L4

פירוט רכיבי המעגל ותפקודם האלקטרוני:

C1: קבל האגירה (100V–1000 µF):

- **תפקיד ראשי: המאגר הראשי של האנרגיה הפוטנציאלית החשמלית**. הקבל נטען באופן יזום למתח גבוה (עד 100V) ומחזיק את האנרגיה הזו עד לשלב השיגור.
- **ניתוח מקצועי:**
  - **קיבול : (C=1000 µF)** הקיבול נמדד בפאראדים (Farads) וקובע את כמות המטען (Q) הנדרשת כדי להגיע למתח נתון.
  - **מתח נקוב : (100V)** זהו המתח המקסימלי הבטיחותי שהקבל יכול לעמוד בו. במעגל דחף, חשוב שהמתח הנספג במהלך הטעינה לא יעלה על ערך זה.
- **אנרגיה אגורה**: האנרגיה הפוטנציאלית המקסימלית האגורה בקבל נתונה על ידי הנוסחה

$$E = \frac{1}{2}CV^2$$

עבור קבל זה בטעינה מלאה (100V):

$$E = \frac{1}{2} \cdot (1000 \cdot 10^{-6}\ \mathrm{F}) \cdot (100\ \mathrm{V})^2 = \frac{1}{2} \cdot 0.001 \cdot 10,000 = 5\ \mathrm{Joules}$$

זוהי אנרגיית השיגור המקסימלית העומדת לרשות המערכת.

D1: דיודת הגנה (1N4148)

- **תפקיד ראשי: חסימת זרם חוזר (Reverse Current Blocking).** הדיודה מוודאת שהזרם יזרום רק מהמעגל המגביר המעגל שמשתמש ב- (L1,L2,L3)אל הקבל (C1).
- **ניתוח מקצועי:**
  - **הגנה מפני פריקה:** בזמן השיגור (כאשר הקבל נפרק), המעגל חווה שינוי דרסטי במתח וזרם גדול. הדיודה מונעת מהזרם הפורק להזיק לרכיבי המגבר (כגון טרנזיסטורים) שעלולים להיות עדינים.
  - **דגם הדיודה: (1N4148)** זוהי **דיודת מיתוג מהירה וקטנה** (Small Signal Switching Diode,) עם זרם קדימה מקסימלי נמוך יחסית (כ -150mA) לכן, יש להניח שהיא ממוקמת **במעגל הטעינה בלבד**, היכן שהזרם קטן יחסית (שכן הקבל נטען **בהדרגה**, ולא כחלק ממעגל הפריקה של ה – 5J-שם נדרשת דיודה חזקה בהרבה (כגון דיודת UF4007 או FR107).

LED1: נורית חיווי

- **תפקיד ראשי: אינדיקציה חזותית למצב הטעינה.**
- **ניתוח מקצועי:**
  - נורית זו מעידה כי המתח על הקבל (V) הגיע לרמה מספקת או מלאה. היא מחוברת בדרך כלל **במקביל לקבל** בטור עם נגד הגבלת זרם מתאים. כיוון שמתח הטעינה הוא 100V, יש צורך **בנגד הספק גבוה** או במנגנון מיתוג (כגון נורית ניאון קטנה או טרנזיסטור) כדי להבטיח שהיא לא תישרף בגלל המתח הגבוה.

L1, L2, L3: סלילים (Inductors/Coils):

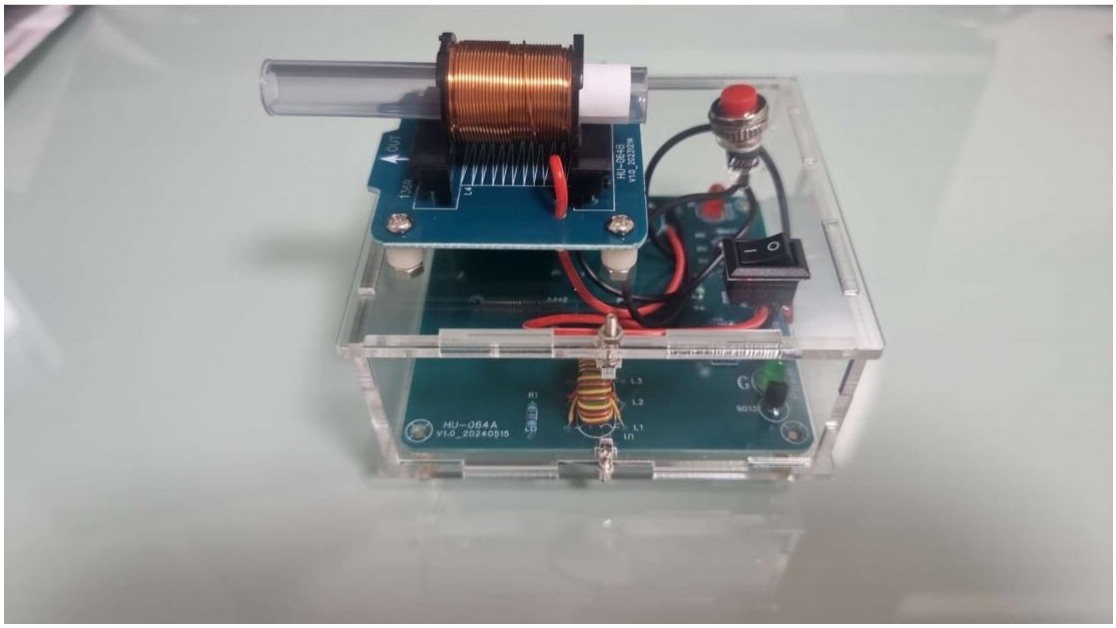
- **תפקיד ראשי:** רכיבים אלו הם ליבת **מעגל המגביר** (Boost Converter) או **ממיר מתח מרומם** (Flyback Converter). תפקידם הוא להפוך מתח כניסה נמוך (כגון סוללה של 3V—12V למתח גבוה של 100V)
- **ניתוח מקצועי:**
  - **ממיר: DC-DC** סלילים אלה משמשים ליצירת שדה מגנטי. על ידי מיתוג מהיר של זרם דרך סליל ראשוני (L1) או (L2 באמצעות טרנזיסטור, נוצרת **השראה הדדית** שמייצרת **מתח גבוה** בסליל משני L2 או L3).
  - **דחיית זרם ישר:** המעגל המגביר פועל על העיקרון של **שינוי מהיר בשטף המגנטי**, שיוצר מתח גבוה על פי חוק פאראדיי. המתח הגבוה הזה מיושר באמצעות דיודה (אולי D1, אם כי D1 היא חלשה מדי, או דיודה נוספת) ונשלח לטעינת C1.

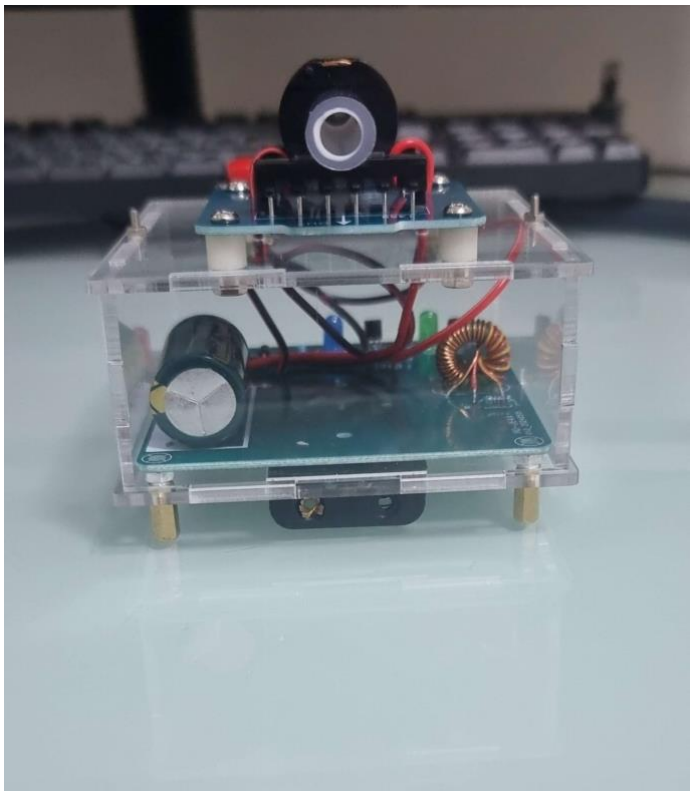
סיכום ומסקנה (הקשר המערכתי):

המערכת כולה מתארת **מחולל דחף אלקטרומגנטי**, העובר את שלבי הפעולה הבאים:

1. **הגברה:** המתח הנמוך מוגבר ל- 100V באמצעות מעגל המתנד/הממיר (L1,L2,L3).
2. **טעינה:** המתח המוגבר עובר דרך דיודת הגנה (D1) ונטען לתוך **קבל האנרגיה** (C1).
3. **מוכנות:** כאשר C1 מגיע ל– 100V (אוגר 5J), נורית החיווי (LED1) נדלקת.
4. **שיגור:** בשלב זה, באמצעות מתג מיוחד) שאינו מצוין ברשימה, כגון טרנזיסטור MOSFET או SCR בעל זרם גבוה), משוחררת האנרגיה האגורה ב- C1 **בבת אחת** לתוך סליל שיגור (העומס), מה שמייצר את הדחף האלקטרומגנטי הדרוש לשיגור.

תמונות של התוצאה הסופית של הרובה:

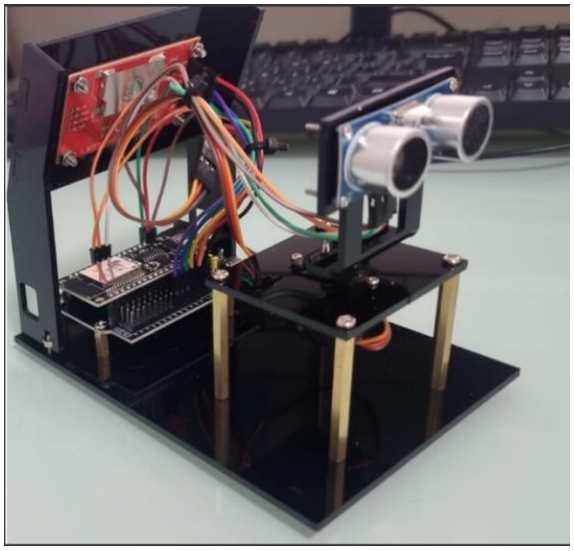




קישור לסרטון בו אני מרכיב את המערכת של הרובה האלקטרומגנטי ומדגים את אופן פעולתו וכיצד מתבצעת הירייה (יש ללחוץ CTRL +על קישור של הסרטון)

https://youtu.be/5IMt-HXeCIk?feature=shared

מיני רדאר (Mini Radar)



המטרה והעקרון:

המטרה של הפרויקט הייתה ליצור מערכת שתסרוק את הסביבה הקרובה שלה, תזהה מכשולים ותציג אותם על גבי מסך, בדיוק כמו מכ"ם אמיתי.

הרכיבים והתפקידים שלהם:

כמו בכל פרויקט מוצלח, גם המערכת הזו מורכבת מכמה חלקים שפועלים יחד:

- המוח: זהו המיקרו-בקר, הלב והמוח של המערכת. הוא שולט בכל הרכיבים ומבצע את החישובים המורכבים בזמן אמת.
- העיניים: החיישן האולטרה-סוני, שמזכיר קצת שתי עיניים קטנות. עין אחת משדרת פולס קולי שאנחנו לא יכולים לשמוע, והשנייה קולטת את ההד שחוזר. הנתון הקריטי שאנחנו מקבלים הוא הזמן שלקח לפולס הזה לצאת ולחזור.
- הזרוע: זהו מנוע הסרוו. הוא מאפשר לחיישן שלנו לנוע מצד לצד, מה שנותן למערכת את היכולת לסרוק שטח שלם ולא רק נקודה אחת.
- התוכנה: כל החלקים הפיזיים לא שווים כלום בלי הקוד שכתבתי. התוכנה היא זו שמנחה את מנוע הסרוו לנוע בזוויות שונות (למשל, 5 מעלות בכל פעם), ובכל עצירה היא מורה לחיישן למדוד את המרחק לעצם. לאחר מכן, הקוד שולח את נתוני המרחק והזווית למסך חיצוני או למחשב.

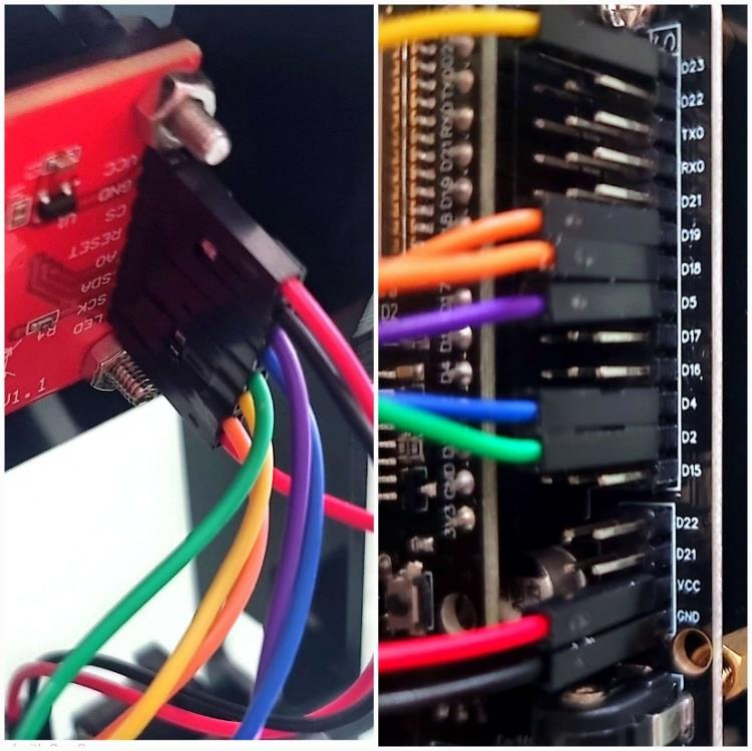
איך זה עובד בפועל?

התהליך פשוט ואלגנטי: המנוע מסתובב בזווית מסוימת, החיישן מודד את המרחק לעצם הקרוב מוזנים לקוד. התוכנה לוקחת את הנתונים –**הזווית והמרחק** – ביותר, ושני הנתונים האלו ומציגה אותם בצורה ויזואלית על מסך. ככל שהמכשול קרוב יותר, כך הנקודה שתופיע על ה"מכ"ם" תהיה קרובה יותר למרכז. על ידי חזרה על התהליך בכל זווית לאורך הקשת של 180 מעלות, אנחנו מקבלים תמונה שלמה של הסביבה שנמצאת מולנו.

הפרויקט הזה מדגים ששילוב פשוט של רכיבים יכול ליצור מערכת מורכבת ויעילה, והוא פותח את הדלת ליישומים מרתקים כמו רובוטים אוטונומיים, מערכות אבטחה ביתיות ועוד.

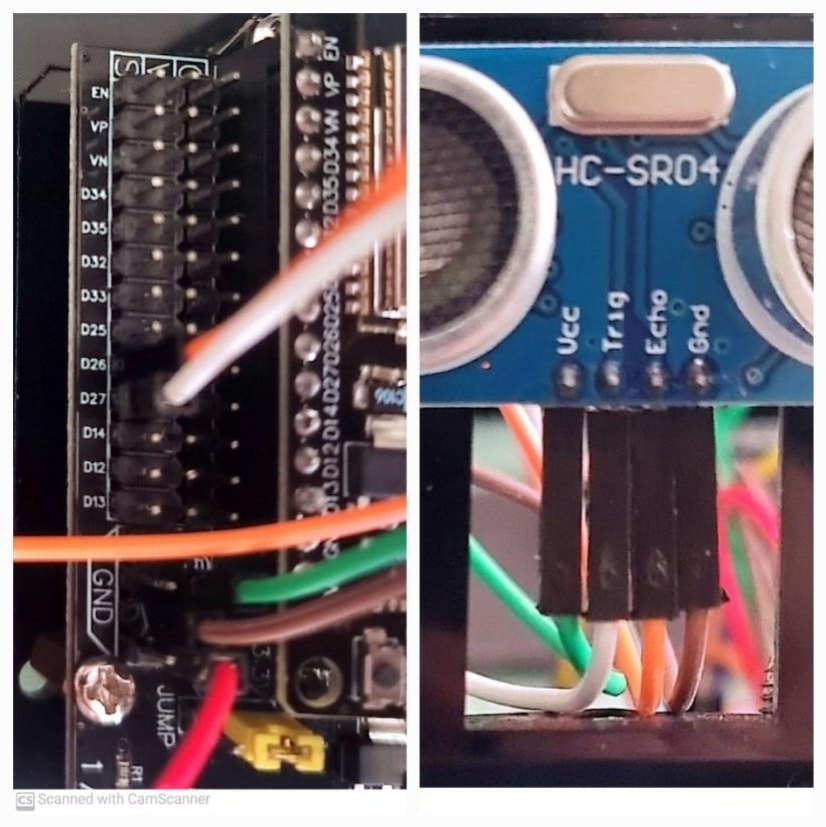
נראה את אופן צורת החיבורים של המסך והחיישן האולטרסוניק אל המיקרו בקר בצורה ברורה ומקורבת יותר:

המסך הקטן מחובר באופן הבא:



- ניתן לראות ש-VCC במסך מחובר ל-VCC שבמיקרו בקר.
- הפין GND שבמסך מחובר ל- GND במיקרו בקר.
- רגל CS שמוצגת במסך מחוברת לפין D5 שבמיקרו בקר.
- רגל ה- RESET שבמסך שלנו מחוברת לפין D4 שבמיקרו בקר.
- רגל A0 יכולה להיות מחוברת לרגל אנלוגית או דיגיטלית תלוי בפקודות שנרצה להשתמש, אני חיברתי רגל זאת לרגל דיגיטלית/סיפרתית D2 שבמיקרו בקר.
- רגל SDA שהיא למעשה הרגל SDI מאחר והמסך מתקשר עם המיקרו בקר דרך תקשורת טורית SPI, ישנו בלבול משום שאנו רגילים שרגל SDA שימושית בתקשורת I2C אך יבואנים סיניים החלו לייצר מסכים ולרשום גם במסכים המתקשרים דרך תקשורת SPI את השם של הרגל SDA והיא מאפיינת את התק4שורת שבין המסך למיקרו בקר.
- רגל SCK מחוברת לפין D18 (החוט השני הוא של המנוע סרבוו שמחובר לפין D19). למעשה רגל SCK זוהי הרגל שמעבירה את אות השעון.
- רגל ה- LED מחוברת ל- 3.3V בצד השני של המיקרו בקר.

נראה את החיבור של החיישן האולטרסוניק:



- רגל ה-TRIG מחוברת לפין D27 שבמיקרו בקר.
- רגל ה-ECHO מחוברת לפין D26 שבמיקרו בקר.
- המתח VCC שבחיישן מתחבר לפין 5V שבמיקרו בקר משום שחיישן זה בדומה למנוע סרוו שלנו עובדים עם 5 וולט והמיקרו בקר ESP32 עובד על 3.3V.
- הארקה GND מחובר כמובן לאדמה GND שבמיקרו בקר שלנו.

קישור לסרטון בו אני מרכיב את המערכת של המיני ראדאר. (יש ללחוץ Ctrl ואז על הקישור)

https://youtu.be/TOxmNhN4LiU

תיעודים-בדיקות

תיעוד של אופן פעולת החיישן האולטרסוניק ומדידת מתחים 5.9.25:

בתיעוד זה נוכל לראות את אופן פעולת החיישן האולטרסוניק, מדדתי את המתח שהספק מקבל מהמיקרו בקר ESP32 ובנוסף בדקתי כמה מתח המיקרו בקר מוציא מהפין של ה- 3.3V האם הוא באמת מוציא מתח זה או שלא, בבדיקות קיבלתי שהכל תקין והחיישן קיבל טיפה פחות מתח מ5 וולט וזה עלול להיגרם ממספר סיבות הבאות:

מתח ה USB-הסטנדרטי מוגדר להיות 5.0V, אך יש לו **סבילות** .**(Tolerance)** קריאה של 4.5V נובעת ככל הנראה מהשילוב של הגורמים הבאים:

1. מפל מתח בכבל ה-USB (Voltage Drop):

זהו הגורם העיקרי:

- **התנגדות הכבל:** לכל כבל, גם הקצר והאיכותי ביותר, יש התנגדות חשמלית. כאשר ה - ESP32והחיישנים מחוברים וצורכים זרם (הם **העומס)** נוצר **מפל מתח** לאורך הכבל, לפי חוק אוהם: Vdrop=I×R.
- **כבלי USB דקים וארוכים:** כבלים דקים יותר (בעלי מספר AWG גבוה יותר, כמו AWG 28 במקום AWG 20 עבה יותר) או ארוכים, הם בעלי התנגדות גבוהה יותר, מה שמגביר את מפל המתח.
- **הקריאה שלך נעשתה "תחת עומס":** אם מדדת 4.5V בזמן שה -ESP32, חיישן האולטרסוניק ומעגל ה- Wi-Fi שלו פעלו, המתח יורד עקב צריכת הזרם.

2.סבילות תקן ה-USB:

תקן USB 2.0 מאפשר למתח לרדת עד 4.4V בחיבור ל "Hub Port"-(שקע פחות חזק) או לכל הפחות **4.75V** בחיבור לשקע רגיל, עד לצריכה המקסימלית המותרת. ערך של 4.5V נופל בטווח הסביר, במיוחד אם אתה משתמש בכבל ארוך או ביציאת USB של מחשב נייד או רכזת(Hub).

3. מפל מתח ברכיבי לוח ה-ESP32:

המתח עובר דרך מספר רכיבים על לוח ה- ESP32 שלך:

- **הגנת זרם יתר: (Over Current Protection)** לוחות רבים מכילים נתיך הניתן לאיפוס או רכיב הגנה אחר שמוסיף התנגדות קטנה וגורם למפל מתח קל.
- **דיודת הגנה: (Protection Diode)** דיודת הגנה בכניסת ה USB-יכולה לגרום למפל מתח של כ- 0.2V עד 0.7V תלוי בסוג.

לסיכום, הקריאה של 4.5V היא תוצאה של הזרם שצורך ה- ESP32 (והחיישנים המחוברים אליו) בשילוב ההתנגדות של כבל ה- USB והרכיבים שעל הלוח. אם הלוח עובד יציב, המתח הזה **תקין לחלוטין** עבורו.

את המדידות שיצאו ניתן לראות בהסבר על החיישן הולטרסוניק בספר זה איפה שאני מסביר על החיישן ואופן פעולתו בסוף ההסברים הכנסתי תחת הכותרת **"מדידות מתח ואופן פעולה"**

מצרף כאן סרטון מיוטיוב בוא תיעדתי את אופן הפעולה ואת המדידות וכמובן אכניס את הסרטון לאתר שלי תחת כותרת בדיקות.

קישור לסרטון: https://youtu.be/qJxPFXFTNug

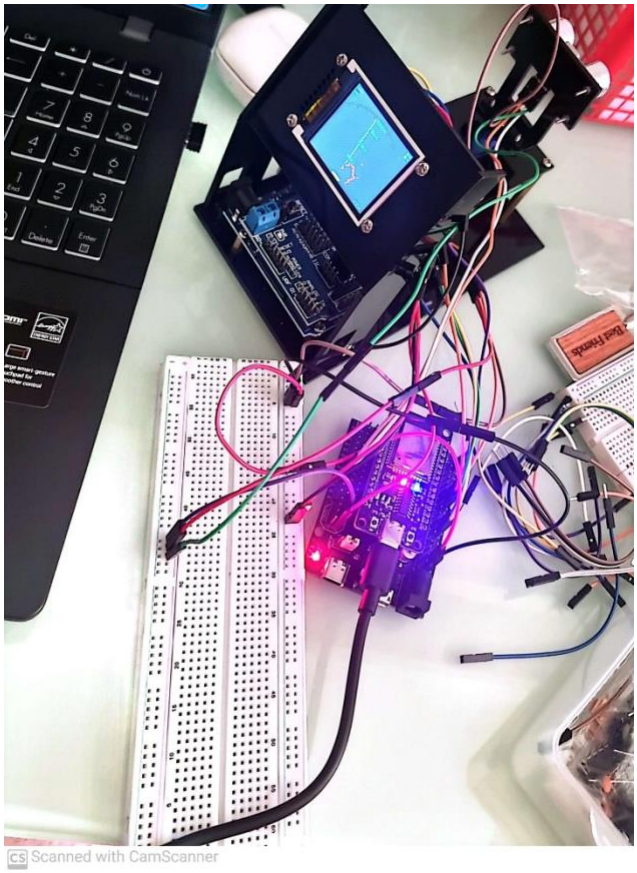
תיעוד של אופן הפעולה 20.9.25:

אחת הבעיות הגדולות שיצא לי להתמודד בפרויקט זה היה להפעיל את מערכת הרדאר.

נתקלתי בבעיות כמו למשל ספריות לא תואמות הגדרת פורטים לא נכונים. בעיות אלו פתרתי על ידי חקירה על הפונקציונליות של המיקרו בקר ובנוסף על ביסוס הפורט בעזרת הטרמינל במחשב, אלו היו הפרוצדורות המסובכות אך המועילות ביותר מאחר והן לימדו אותי להתמודד עם בעיות כאלו לפעמים הבאות במידה ואתקל בהן.

נוסף על כך אחת הבעיות שיצא לי להתמודד היה הפינים שהוגדרו להיות "בעיתיים" לשימוש, הכוונה שבמיקרו בקר אי אס פי ישנם פינים בהם כדאי להשתמש משום שנוחים יותר עבור פעולות מסוימות וישנם כאלו שמוגדרים לעבודה ספציפית יותר. לדוגמה יכול לומר שהמנוע סרוו שלי לא עבד כאשר חיברתי אותו לפין 15 ואז שיחקתי קצת עם הפינים ומתי שחיברתי לפין 19 הוא התחיל לעבוד והסתנכרן בעבודה יחד עם המסך והחיישן אולטרה סוני שלמעשה נמצא עליו. בנוסף אחת הבעיות הגדולות ביותר היה להפעיל את המסך. גרפיקה זה צעד גדול בפרויקט ויחסית מסובך, לא כל הפונקציות תואמות למיקרו בקרים ספציפיים וצריך לנסות כמה פונקציות על מנת למצוא את המתאימה ביותר עבור אופן הפעולה שלך.

מאחר ואני יוצר תמונת הרדאר שפועל בזמן אמת אז עלי לא להשתמש רק בתצוגה רגילה אלא ממש בתצוגת עיצוב גרפי שדורש ממני להשתמש בפונקציות ייחודיות עבור פעולות אלו. בסופו של דבר עם קצת כישלונות ולמידה הצלחתי להבין יכן טעיתי בכל הניסיונות הללו ולמדתי מזה המון.



גם כאן בתמונה ניתן לראות את הניסיון הפעלה על מיקרו בקר אי אס פי.

בהתחלה חיברתי הכל דרך מטריצה ולבסוף ארגנתי את זה על המערכת עצמה כדי לשפר את הנראות והנוחות במהלך השימוש בה.

בתמונה קצת קשה לראות את הרדאר שנוצר על המסך וכמובן שהתמונה לא משדרת את התזוזה של המנוע סרוו ואת הקליטה של החיישן אולטרסוני אך כדי להמחיש זאת אני צירפתי לכם סרטון שלי בודק את אופן הפעולה של המערכת.

בסרטון אציג את הבנייה כולה של המערכת ואת אופן פעולתה בסוף של הסרטון כדי שניתן יהיה לראות ולעקוב אחר השלבים צורת הבנייה של המערכת.

בנוסף הסרטון יעלה לאתר שניתן לגשת אליו דרך ספר זה בקישור שאצרף שיהיה תחת הכותרת "**קישור לאתר**"

תיעוד של אופן פעולת הרדאר 21.9.25:

בניסיון זה בדקתי את סימון הנקודות הצהובות שהן מופיעות עבור אובייקטים שרחוקים מ100 סנטימטר. הנקודות האדומות שמופיעות על הרדאר אלו נקודות שמאפיינות אובייקטים הנמצאים בתווך של 0 עד 100 סנטימטרים מהחיישן אולטרסוניק.

תיעדתי ניסיון זה והוספתי סרטון שמראה כיצד בדקתי זאת הסרטון טיפה הפוך החלק של הבדיקה לקראת הסוף אבל זה לא לפרק זמן ארוך וניתן להבחין בכל זאת את הנקודות הצהובות. אני מצרף כאן קישור לסרטון שלי שנמצא ביוטיוב וגם הכנסתי את הסרטון תחת הקטגוריה "בדיקות" בתוך האתר שיצרתי שיופיע בעמודים האחרונים של הספר פרויקט שלי.

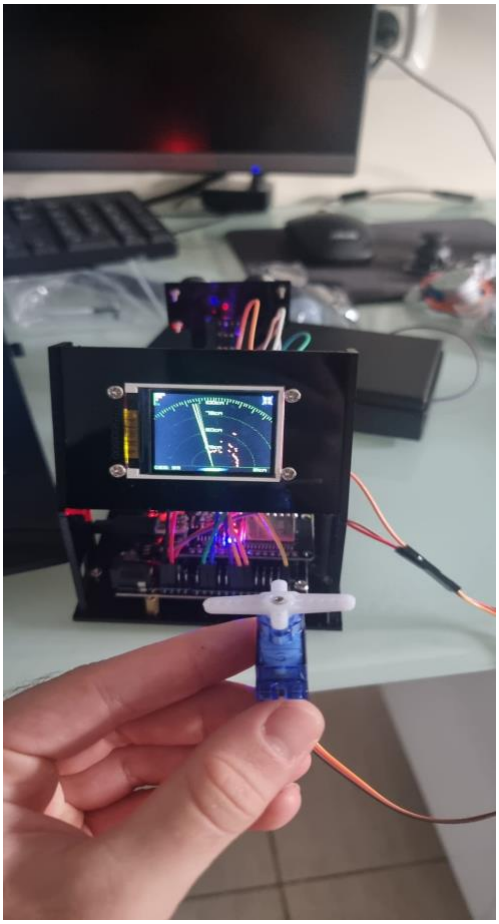
קישור לסרטון תיעוד של ניסיון יצירת ובדיקה של הנקודות הצהובות שמגדירות אובייקטים שנמצאים במרחק מעל 100 סנטימטרים כולל מהחיישן האולטרסוניק:

קישור לסרטון: https://youtu.be/8BiuVFxUo20

תיעוד של אופן הפעולה כאשר חיברתי סרוו נוסף שיהיה אחראי על הסיבוב של התותח 23.9.25:

בניסוי זה קפצתי צעד יחסית משמעותי בפרויקט שלי משום שהצלחתי לבצע סנכרון בין שני המנועי סרוו שיש לי במערכת שלי. הסרוו הראשון אחראי על לסובב את החיישן האולטרסוני שסורק את הסביבה בטווח של 180 מעלות החל מ0 מעלות ועד ל180 מעלות. המנוע הסרוו השני שלי אחראי למעשה על הסיבוב של המשטח עליו אחבר את הרובה האלקטרומגנטי שלי. ברגע שמתגלות נקודות אדמות שבמשמעותן שהאובייקט נמצא בתווך של 100 סנטימטר ואף פחות אז המנוע הסרוו השני נכנס ישירות לפעולה מאותו המקום בוא המנוע סרוו הראשון שמסובב את החיישן האולטרסוני נמצא כעת וברגע שיתגלה שוב נקודה אדומה שמסמנת על אובייקט אז תתבצע יירייה מהרובה האלקטרומגנטי אך זה בשלבים מתקדמים יותר כרגע בצעתי את הנסכרון של המנוע הסרוו השני עם הראשון יחד.

נראה תמונה ואסביר את החיבורים הנוספים שבצעתי למיקרו בקר על מנת שהמנוע סרוו השני יתחיל להסתובב:



זאת למעשה התמונה בה אני מחזיק את המנוע סרוו השני, למעשה אי אפשר להבין שהמערכת אכן עובדת מתמונה פשוטה ולכן חשבתי אצרף סרטון ביוטיוב וקישור בתוך הספר וכמובן שאכניס את התיעוד לקטגוריית הבדיקות שבאתר שלי.

המנוע סרוו מחובר לרגל D25 שבמיקרו בקר ESP32 ואת שאר החוטים שבמנוע סרוו צירפתי את החוט האדום ל-VCC ואת החוט החום חיברתי לאדמה GND.

בקוד כל מה שעלי היה להוסיף זה את ההגדרה של הסרוו הנוסף ולאיזו רגל הוא מחובר במיקרו בקר, ובנוסף לכך היה עלי להוסיף בתנועה הלוך והתנועה חזור בתוך התנאי איפה שרשום המרחק קטן מ-100 שזה למעה הנקודות אדומות מה שאומר שהאובייקט נמצא בטווח של 100 סנטימטרים ומטה בשניהם היה עלי להוסיף פקודה שתפעיל את המנוע סרוו השני או במידה והמרחק של האובייקט גדול מ-100 אז התנוע של המנוע סרוו השני פוסקת והמנוע עוצר.

מצרף קישור לסרטון בעמוד שלי ביוטיוב:

https://youtu.be/eeDQquCGN5I

ביבליוגרפיה:

במהלך הפרויקט שלי נעזתרי במספר מקומות באינטרנט כדי לרכז ולסכם את המידע בצורה הכי פרקטית ומקצועית שאפשר. נעזרתי במספר אתרים.

האתרין שיצא לי ללמוד מהם ולרכז מידע מדויק שעזר לי לקדם את הספר פרויקט היו אתרים שנעזרתי בהם בכללי בכל תקופת הלימודים שלי במהלך התואר הטכנולוגי (יג-יד) שעשיתי.

באתרים אלו ישנם סיכומים רבים ומבחני תירגול שהייתי מתכונן דרכם ומתפתח באופן כללי כדי לחזק ולשפר את הידע שלי לגבי חיישנים למיניהם או פרוטוקולים שונים.

לדעתי גם במהלך התואר ובאופן כללי בחיים יש צורך להתפתח מחו למסגרת הלימודים הקצובה לנו ולקחת קורסים עצמאיים כדי לקבל ידע ממקורות נוספים.

האתרים שבעיקר השתמשתי בהם הם:

- האתר לאלקטרוניקה ומחשבים אתר גדול עם סיכומים רחבים בכל התחומים. דרך אתר זה למדתי עוד בלימודי במהלך התיכון והוא עזר לי המון כדי לחזר ואף לקדם את הידע שלי בכל תחום עולמות האלקטרוניקה, התכנות והמחשבים, קישור לאתר: https://www.elec4u.co.il/
- אתר נוסף שעזר לי לגבי הפרויקט שלי גם במהלך הפרויקט בסוף לימודי בתיכון במקצוע אלקטרוניקה ומחשבים וגם בלימודי בתואר הטכנולוגי במהלך פרוייקט הגמר זהו האתר של פורט, יש לו חומרי לימוד, סיכומים, דוגמאות ועוד הרבה דברים שימושיים שיכולים לעזור להבנת הנושאים בכל עולמות האלקטרוניקה, תכנות ומחשבים. הקישור לאתר שלהם: https://www.arikporat.com/
- ספר טוב מאוד שממנו לקחתי חלק מהתמונות ויש בוא Data-sheets בנושא ה- I2C ודברים טובים ושימושיים להבנת הפרוטוקול תקשורת I2C. הקישור לספר זה שנעזרתי בו הוא: https://www.st.com/resource/en/datasheet/vl53l8cx.pdf
- מקום נוסף שהשתשמשתי בוא על מנת לסכם מידע ולקיחת חלק מהתמונות בנוגע למנוע SERVO זהו הספר עם הקישור הבא: https://www.arikporat.com/wp-content/uploads/2024/03/%D7%9E%D7%A0%D7%95%D7%A2-SERVO.pdf
- נעזרתי בדף הבא על מנת לסכם לגבי החיישן אור שהוא חלק מן המודול חיישן LDR שבו אני משתמש בפרויקט שלי, הקישור לאתר הוא: https://gabbyshimoni.wixsite.com/arduino-programming/ldr

קישור לאתר שלי : https://raybrook-project-santibot15.netlify.app/