פרוייקט

WiFi Alarm

אלון תלגוקר

תקציר הפרויקט:

מטרת הפרויקט הינה תכנון ובניית אזעקה בעלת יכולות חישה רבות.

המערכת בנויה לפי מודל חדיש של מערכות אינטרנט של חפצים-IOT. היא כוללת יחידה לממשק עם המשתמש ומספר יחידות חישה. כל היחידות מקושרות בתקשורת Wi-Fi עם יחידה מרכזית.

המערכת מתריעה מפני פריצה לבית, דליפת גז בישול, דליפת עשן.

פונקציה נוספת של המערכת הינה מעקב טמפרטורה של חדר תינוק עם התראה מפני טמפרטורה גבוהה או נמוכה מידי המסכנת את התינוק.

כיוון שהעברת התקשורת בין היחידות מתבצעת באמצעות Wi-Fi אין צורך בהעברת כבלים בבית, יש רק לספק ליחידות מתח הזנה על ידי חיבור סוללה מתאימה .

על ידי שימוש ביחידה המרכזית המשתמש יוכל לראות את מצב החיישנים ואחוזי הסוללות ובנוסף יוכל לקבוע את מצבי האזעקה.

זיהוי המשתמש היא על ידי תג זיהוי (RFID).

המערכת מתוכננת בצורה מודולארית וניתן להוסיף לה יחידות חישה נוספות בצורה יחסית קלה (יש רק להוסיף קוד מתאים ליחידת הממשק למשתמש).

המערכת שולחת התראות לפלאפונים בעלי מערכת הפעלה Android.

תוכן עניינים-

רשימת טבלאות:

רשימת איורים:

פרק 1

מבוא

1.1 מבוא:

בפרויקט זה תכננו ובנינו מערכת אזעקה לבית.

המערכת תתריע מפני פריצה לבית, דליפת גז בישול, גילוי שריפה, ותציג את הטמפרטורה בחדר נבחר.

המערכת מורכבת ממספר כרטיסים : כרטיס ממשק משתמש, וכרטיסי חישה וצופר.

בין הכרטיסים מתקיימת תקשורת Wi-Fi לפי מודל הנפוץ במערכות IOT (Internet Of Things). לפי מודל זה המערכת מורכבת מיחידות שונות שמתקשרות עם היחידה המרכזית המכונה Broker. היחידות השונות יכולות לשלוח מידע ל-Broker (Publisher) או להירשם לצורך קבלת נתונים מה-Broker (Subscribe).

בין היחידות תקשורת לפי פרוטוקול MQTT (מוסבר בנספח) פרוטוקול זה נפוץ יותר מפרוטוקול HTTP במערכות IOT ובמערכות מנוהלות באמצעות מיקרובקרים "חלשים".

ה-Broker ממומש באמצעות כרטיס Raspberry Pi Zero-W בעל מערכת הפעלה Linux-Raspbian ומריץ תוכנה Eclipse Mosquitto שהיא תוכנת Broker בקוד פתוח.

כל שאר הכרטיסים נשלטים באמצעות כרטיס מיקרו בקר ESP8266-12S. זהו מיקרו בקר בעל 32 סיביות, ובעל ליבה לתקשורת Wi-Fi כולל TCP/IP Stack.

יחידת הממשק למשתמש כוללת צג מגע שנותנת למשתמש אפשרות נוחה להקליד הוראות הפעלה ולראות את המידע של החיישנים השונים שבבית.

כרטיסי יחידת החישה מזהים שינוי קריטי באחד מהחיישנים בבית ומפעילה צופר.

המערכת שולחת התראות לפלאפונים בעלי מערכת הפעלה Android.

1.2 מפרט טכני:

- אזעקה לבית חכם
- גילוי פריצה לדירה על ידי שימוש בחיישן נפח ופתיחת דלת.
- גילוי עשן בדירה, גילוי דליפת גז בישול.
- התראה מפני טמפרטורה גבוהה או נמוכה בחדר תינוק.
- תכנון אזעקה באמצעות מסך מגע.
- תקשורת Wi-Fi (פרוטוקול MQTT) בין היחידות השונות.
- כל אחת מהיחידות מוזנת מסוללת ליתיום דור חדש ($LiFePO_4$), זמן בין הטעינות מותנה בקיבול הסוללה (mah).

1.3 רשימת מרכיבים:

- כרטיס ARDUINO DUE הכוללת בתוכו מיקרו בקר AT91SAM3X8E
- בקר ESP8266-12S
- חיישן עשן MQ135
- חיישן גז בישול MQ6
- זיכרון AT24c64 EEPROM
- חיישן תנועה HC-SR501
- בורר CD74HC4053
- חיישן טמפרטורה DS18B20
- צ׳יפים + חיישן קליטת צ׳יפ RFID
- שעון RTC DS1302
- מייצב RT9193-28
- טרנזיסטור AO3401 MOSFET
- מגברי שרת LMV324 כחוצץ ושמיט
- מסך מגע 2.8״ SPI TFT LCD
- סוללות LiFePO4 3.2v
- מגביר מתח DC-DC Boost Converter Step Up 5v
- צופר 1Kohm Passive Buzzer Electromagnetic Alarm

1.4 הוראות הפעלה:

יש לחבר אל כל יחידות החישה סוללות של $LiFePO_4$ ליחידות החיישני גז בישול והעשן 2 סוללות ולשאר יחידות החישה סוללה אחת. אל יחידת ממשק המשתמש יש לחבר ליחידת ממשק המשתמש סוללה של 9v. את ה כרטיס Raspberry Pi Zero-W יש לחבר לחשמל או לסוללה ניידת של 5v עם מינימום זרם של 500mA (עדיפות לזרם של 2A).

בדוק כי על יחידת ממשק המשתמש קיבלת ערכים של החיישנים ושל אחוזי הסוללה.

במידה ולא קיבלת שום ערך בדוק כי ישנה נקודת גישה של Wi-Fi עם שם המשתמש והסיסמא שהוגדר ביחידות חישה.

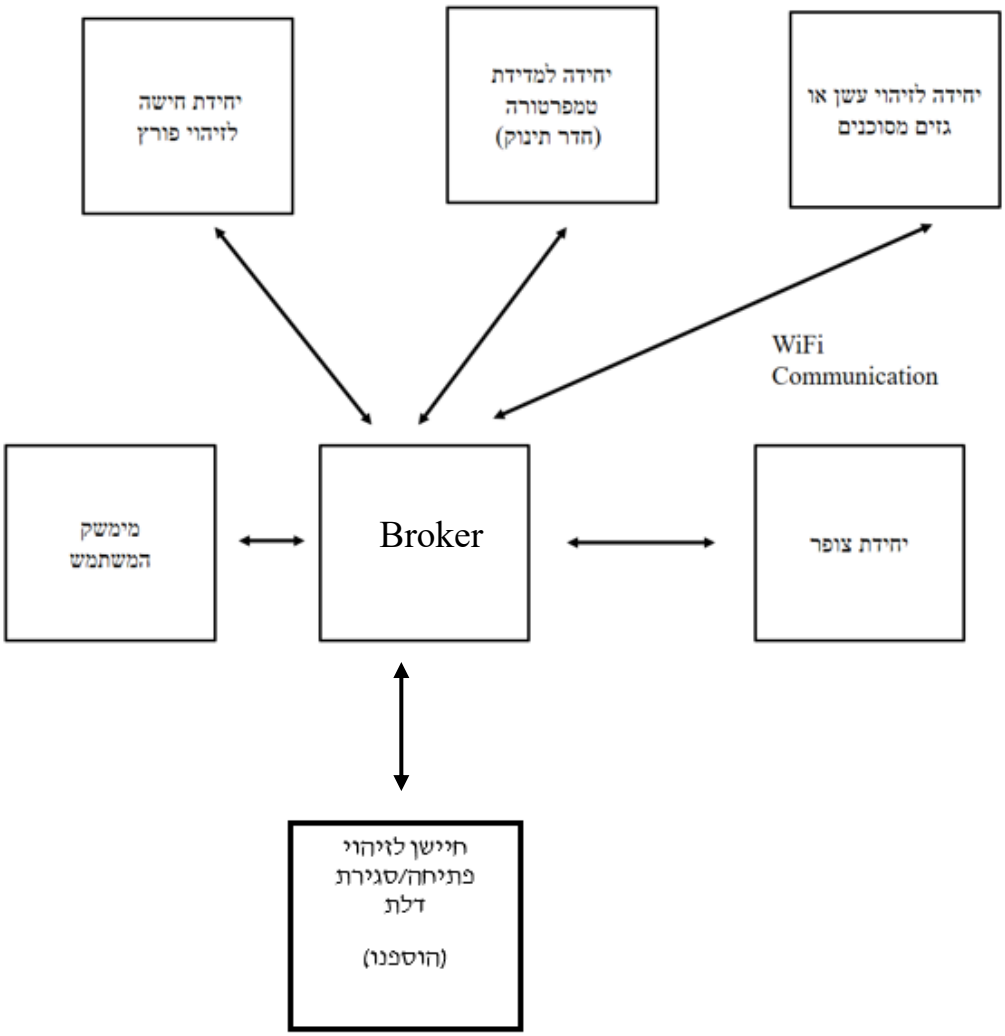
במידה וזו לא הבעיה בדוק כי Raspberry Pi Zero-W מחובר אל נקודת הגישה.

במידה וקיבלת רק חלק מהערכים של החיישנים על התצוגה החלף סוללה ליחידות שלא מגיבות וחכה לערך על המסך.

פרק 2

דיאגרמת בלוקים

2.1 דיאגרמת בלוקים:



איור 1 - דיאגרמת מלבנים של הפרוייקט

במבוא של ספר הפרויקט יצגנו שהפרויקט בנוי ממספר יחידות שבניהם מתקיימת תקשורת Wi-Fi, הדיאגרמה הכללית של הפרויקט מתוארת באיור 2.1 ובה רואים את היחידות השונות של הפרויקט. השימוש בתקשורת אלחוטית משחררת מהצורך במשיכת כבלים בין היחידות השונות שממנה מורכבת מערכת האזעקה.

Broker- זו היחידה המרכזית שאחראית על קבלת המידע ושליחת המידע בין היחידות השונות. תחילה השתמשנו בתוכנת Broker ברשת ב-Online של Adafruit וקצת קיבלנו רקע על הנושא, לאחר מכן עברנו למחשב בעל מערכת הפעלה של Windows מפני שנקודת הגישה הייתה לא איכותית עד כדי שהתקשורת נקטעה. לבסוף החלטנו להשתמש ב-Raspberry Pi Zero-W מפני גודלו הקטן עלותו הנמוכה וצריכת זרם נמוכה.

ממשק למשתמש: באמצעות יחידה זו המשתמש שולט על אופן הפעולה של היחידות השונות של האזעקה וגם יכול לקבל מידע ומשוב מהיחידות. יחידה זו מאפשרת למשתמש לשלוט על מצבי האזעקה בבית. כיוון שהשתמשנו בפלאפון כ--Router Wi Fi היינו מוגבלים לשמונה יחידות קצה, לא בנינו כרטיס נפרד ליחידת הצופר אלא חיברנו אותו ליחידה זו.

יחידה לזיהוי עשן וגזים מסוכנים: יחידה זו נועדה ליידע את המשתמש מפני דליפה של גז בישול ועשן המעיד על דליקה בבית. לשם כל השתמשנו בחיישן MQ6 אשר מזהה דליפה של גז בישול, ו-MQ135 המזהה עשן בבית.

יחידה למדידת טמפרטורה: כיוון שתינוקות רגישים לטמפרטורות נמוכות או גבוהות החלטנו להוסיף למערכת האזעקה יחידה לניטור הטמפרטורה בחדר התינוק ותפקידה לתת התראה שהטמפרטורה חורגת מתחום הערכים שתינוק רגיש להם. כחיישן טמפרטורה השתמשנו ברכיב DS18B20 בעלת דיוק של חצי מעלה.

יחידת חישה לזיהוי פורץ: יחידה זו נועדה לידע את המשתמש מפני תנועה לא מוכרת בבית. לשם כך השתמשנו בחיישן נפח שמשמש לזיהוי תנועה -HC-SR501.

יחידה לזיהוי פתיחה/סגירת דלת :יחידה זו נועדה לידע את המשתמש על מצבה של הדלת. לשם כך השתמשנו במפסק מאולתר המורכב מחוט מוליך אשר מתקצר כאשר הדלת סגורה ומתנתק כאשר הדלת פתוחה.

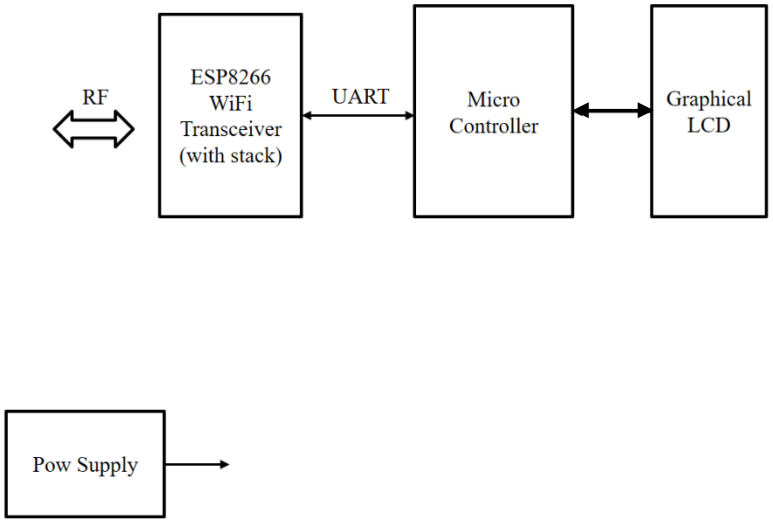
פרק 3

סכמה חשמלית

3.1 יחידת ממשק המשתמש

המבנה הכללי של יחידה זו מתואר באיור 3.1 והסכימה החשמלית באיור 3.2.

יחידה זו מורכבת מכרטיס בקר זעיר Arduino Due שאחראי על הפעלת המרכיבים השונים של יחידה זו, כרטיס ESP8266 שתפקידו ביצוע תקשורת Wi-Fi, תצוגת LCD גרפית עם ממשק מגע, צופר (מעשית השתמשנו בזמזם על מנת להקטין את הרעש), וסוללת 9V.



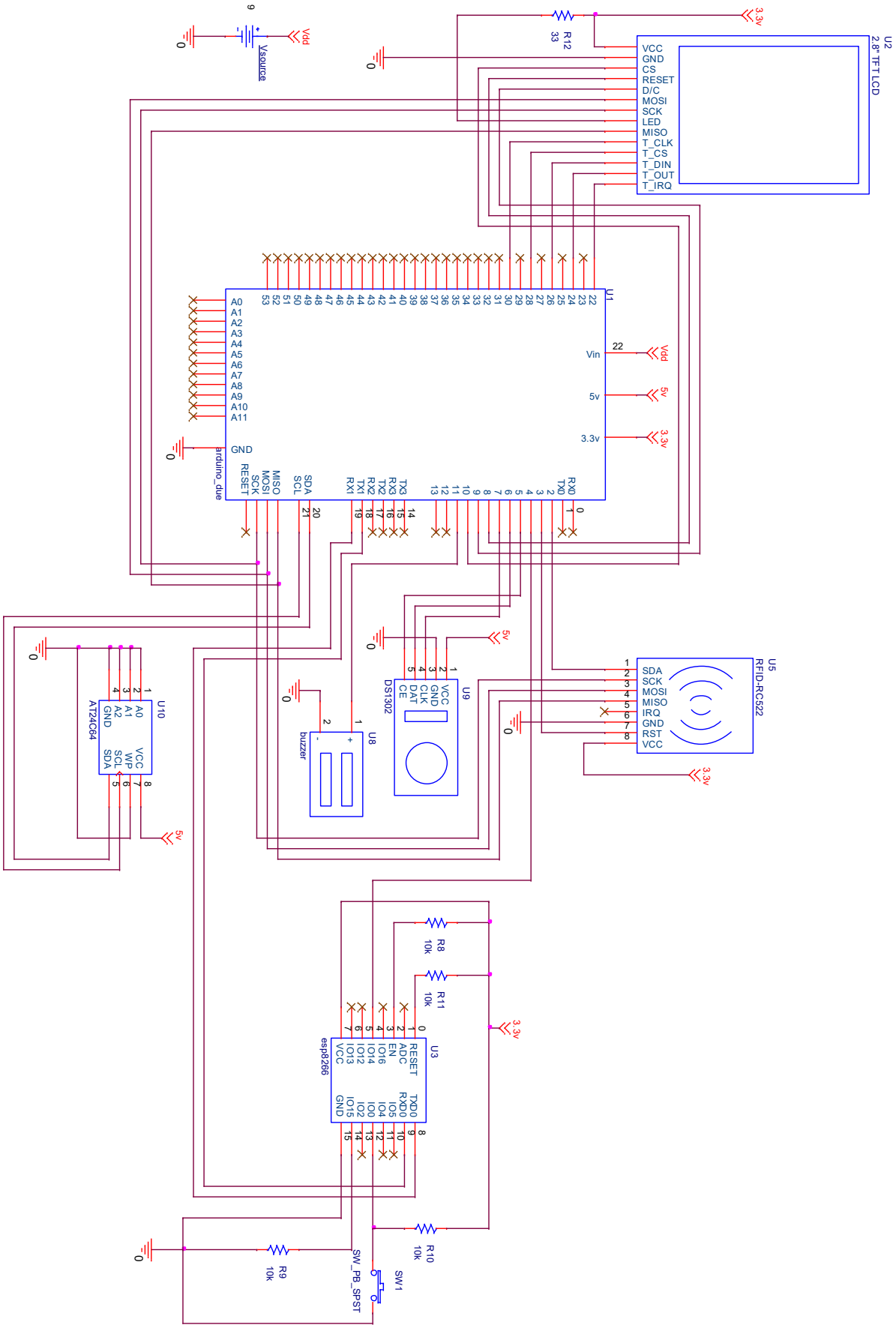
איור 2 - דיאגרמת בלוקים של יחידת ממשק המשתמש

החיבורים מתוארים בטבלה 3.1.

יש לציין שהחיבור בין המיקרובקר Arduino Due לבין המסך LCD כולל יחידת המגע הינו חיבור אות טורי לפי פרוטוקול SPI שהוא פרוטוקול סינכרוני לכן חיבור זה דורש ארבע חוטים:

- MISO (Master In Slave Out)
- MOSI (Master Out Slave In)
- SCK
- אפשור: הדק CS של ה-LCD, הדק T_CS של משטח המגע, הדק SDA של ה-RFID.

ל-Arduino Due מספקים מתח של 9v, על הכרטיס ישנו מיצב מתח מסוג LM1117-3.3 שממיר את המתח ל-3.3v. מתח זה מחובר להדק של הכרטיס ואנו משתמשים בו להפעלת שאר כרטיסי המערכת.



איור 3 - סכמה חשמלית של יחידת ממשק המשתמש

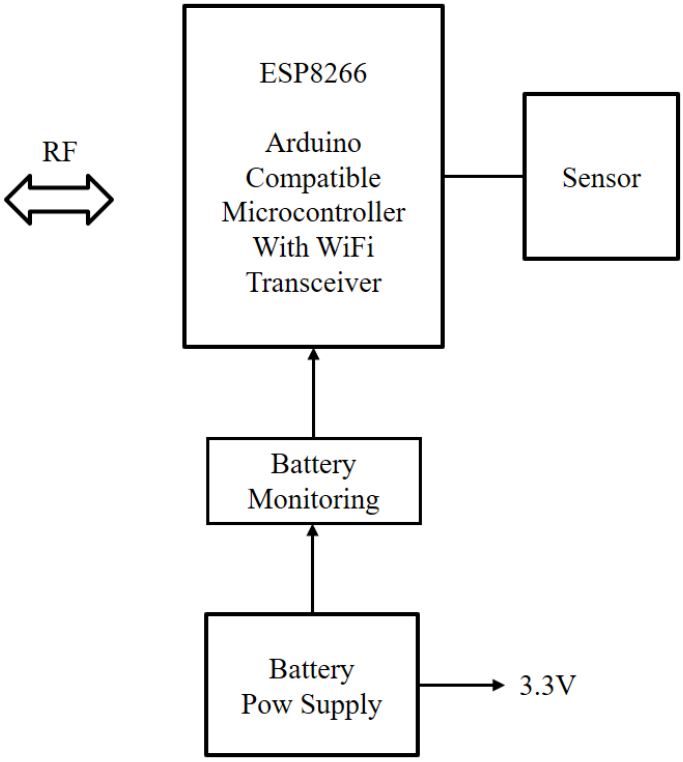
טבלה 1 - טבלת החיבורים של ה-ARDUINO DUE:

הדקים	שם ה-SIGNAL	הסבר	לאן מחובר בפרויקט
8 9 10	D8 D9 D10	הדקים דיגיטליים	תצוגה גרפית
22 24 26 28 30	D22 D24 D26 D28 D30	הדקים דיגיטליים	מסך מגע
MOSI MISO SCK	ICSP ICSP ICSP	תקשורת SPI	תצוגה גרפית חיישן מזהה צ'יפים RFID
2 3	D2 D3	הדקים דיגיטליים	חיישן מזהה צ'יפים RFID
4	D4	הדק דיגיטלי	בקר ESP8266-12S
18 19	TX1 RX1	תקשורת UART1	
5 6 7	D5 D6 D7	הדקים דיגיטליים	שעון DS1302
11	D11	הדק דיגיטלי	צופר
20 21	SDA SCL	תקשורת I2C	זיכרון AT24C64 EEPROM

3.2 תרשים מלבנים עקרוני של יחידת החישה

לכל אחד מיחידות החישה שבפרויקט מבנה דומה לכן לא אתאר כל אחת מהיחידות בנפרד. המבנה מתואר באיור 3.3 והסכימה החשמלית באיור 3.4.

יחידה זו מורכבת מכרטיס בקר זעיר ESP8266 שאחראי על הפעלת המרכיבים השונים של יחידה זו וגם אחראי על ביצוע תקשורת Wi-Fi הכולל שליחת מידע אודות החיישן והסוללה.

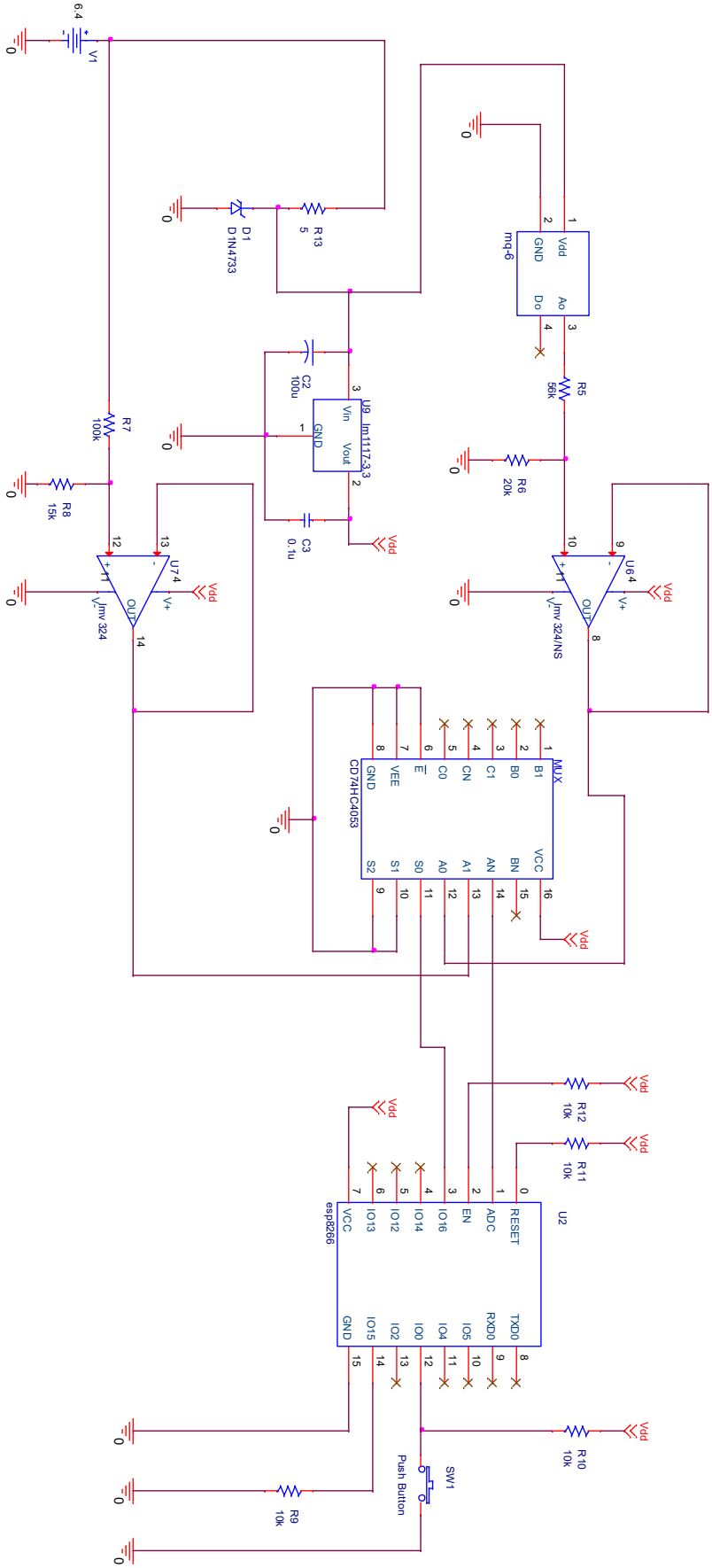


איור 4 - דיאגרמת בלוקים של יחידת החישה

סכמה מלבנים זו מתארת את אופן פעולת יחידת החישה. בפרויקט שלנו קיימות שישה יחידות של חיישנים הכוללות: 2 חיישני תנועה, חיישן עשן , חיישן גז בישול, חיישן טמפרטורה, וחיישן לגילוי מצב הדלת.

מידע החיישן ואחוזי הסוללה משודרים בצורה אלחוטית בעזרת בקר ESP8266 אל יחידת ממשק המשתמש, ובכך המשתמש יכול לקבל מידע על המערכת כולה.

עקב כפילות הסכימה החשמלית של חיישני הגז הוספנו רק תרשים אחד ששייך עקרונית לשניהם



איור 5 - תרשים חשמלי של חיישני הגז והעשן

הסבר תרשים חשמלי:

בעזרת זנר ייצבנו את המתח ל-5v בשביל החיישני גז, ואת הנגד R13 בחרנו כדי שיזרום דרכו זרם שיספיק גם לזנר וגם לשאר העומס של המעגל.

לשאר המעגל אנחנו השתמשנו ב-lm1117-3.3 שמנחית מתח מ-5v ל-3.3v .

מתח הסוללה ומתח החיישנים מגיעים למחלק מתח שנועד להקטין את המתח ביחס קבוע כדי שה-ESP8266 יוכל למדוד את הערכים הללו.

החוצצים נועדו בכדי למנוע מהתנגדויות של הרכיבים הבאים שמחוברים לא לפגוע בערך מחלק המתח.

מפני של-ESP8266 יש רק הדק אנלוגי אחד בחרנו במוקס, ובעצם נכניס כל פעם אות אחר על פי איך שנגדיר את הדקי הבקרה של המוקס.

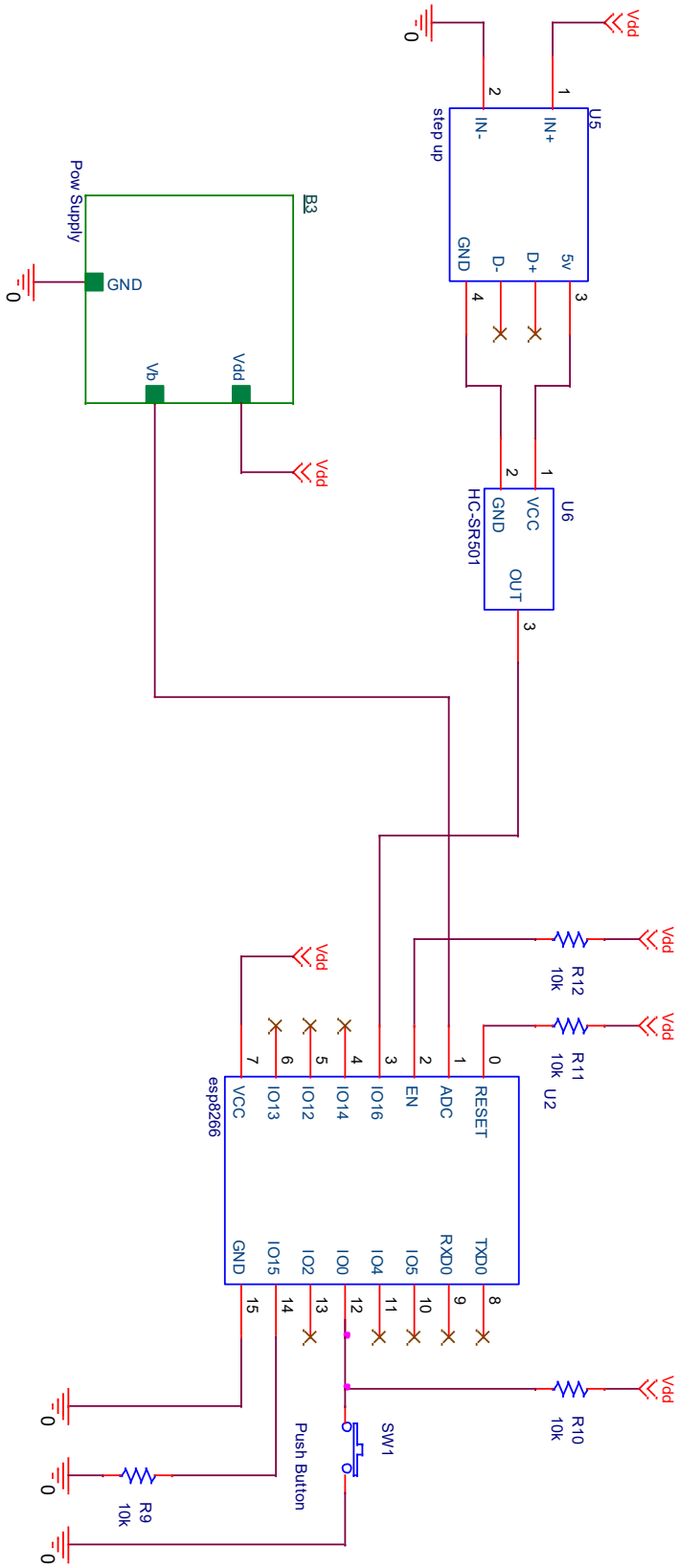
***מכיוון שרכיב זה קיים בפרויקט בכל אחת מהמערכות, אנו נתייחס לכל הדק ברכיב זה, ולאן היא מחוברת בכל אחת מהמערכות.**

טבלה 2 - טבלת החיבורים של בקר ה-ESP8266

הדק	שם ה-SIGNAL	הסבר	לאן מחובר בפרויקט
0	RESET	הדק איפוס	מחובר דרך נגד PULL-UP בעל התנגדות של 10K, אל מתח של 3.3v מסוללה. **תקף בכל המערכות.**
1	ADC	הדק אנאלוגי	**הדק זה מחובר:** **מערכת חיישן גז בישול MQ6** **ומערכת חיישן עשן וMQ135:** מחובר לבורר CD74HC4053 להדק AN. **במערכות: חיישני תנועה , חיישן טמפרטורה DS18b20, גלאי סגירה ופתיחת דלת:** מחובר למחלק מתח הנועד למדוד את מתח הסוללה.

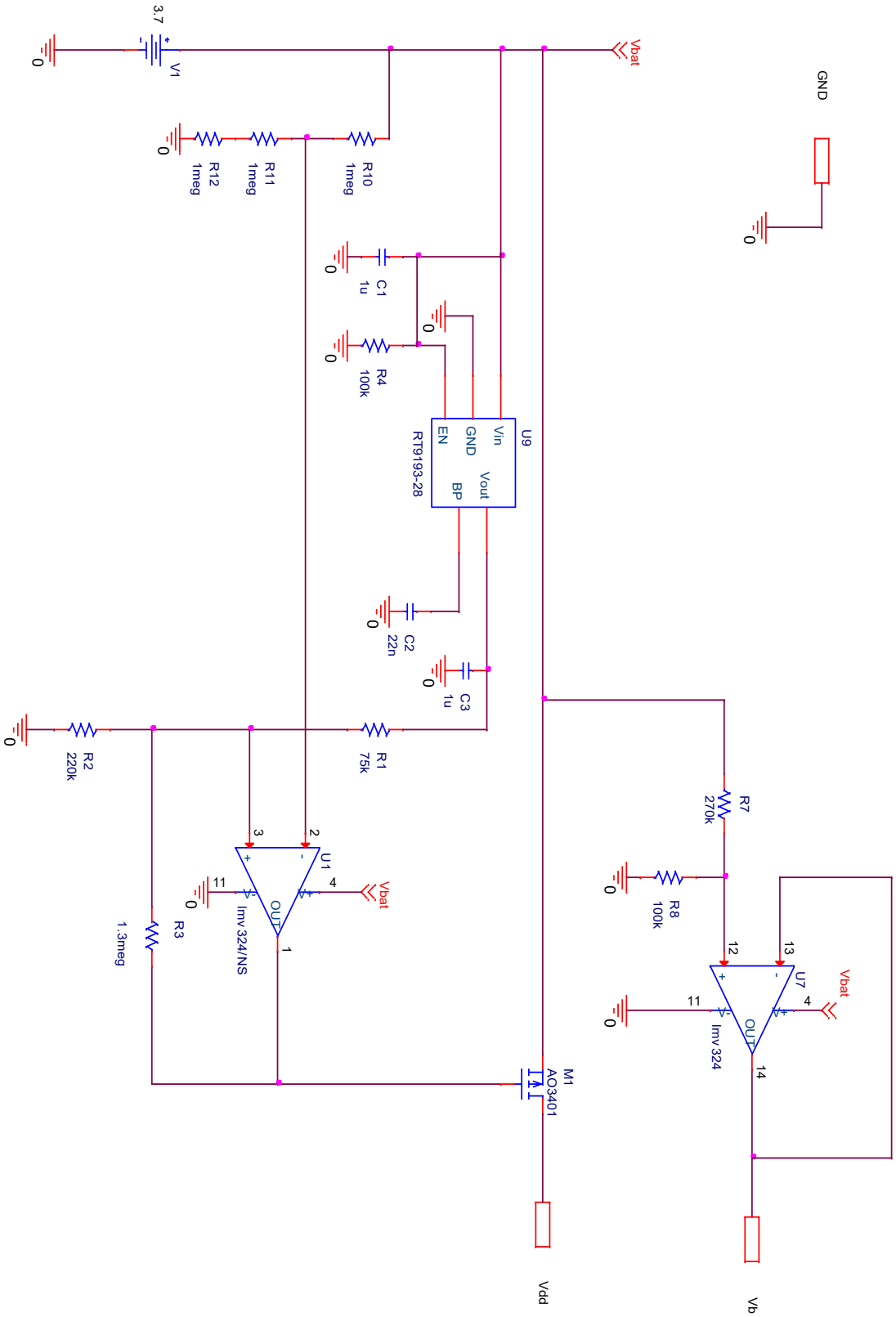
2	EN	הדק אפשור הרכיב	מחובר דרך נגד PULL-UP בעל התנגדות של 10K, אל מתח של 3.3v מסוללה. **תקף בכל המערכות.**
3	IO16	הדק דיגיטלי	**<u>הדק זה מחובר:</u>** **מערכת חיישן גז בישול MQ6** **ומערכת חיישן עשן וMQ135:** מחובר לבורר CD74HC4053 להדק S0. **במערכות: חיישני תנועה PIR7, חיישן טמפרטורה DS18b20, גלאי סגירה ופתיחת דלת:** מחובר ישירות למוצאי החיישנים.
7	VCC	מתח הזנה	3.3v, **תקף בכל המערכות.**
12	IO0	הדק דיגיטלי, לאפשור צריבה	הדק זה מחובר דרך ג'אמפר אל האדמה. **תקף בכל המערכות.**
10	IO5	הדק דיגיטלי, נועד לגילוי פתיחה/סגירת דלת	הדק זה מחובר דרך מוליך אל האדמה. **תקף אך ורק במערכת גלאי סגירה ופתיחת דלת**
14	IO15	אפשור RESET	מחובר לאדמה דרך נגד PULL DOWN **תקף בכל המערכות**.
15	GND	אדמה	אדמה, **תקף בכל המערכות.**

עקב מספר הרב של התרשימים החשמליים שמטרתם העיקרי זהה נסביר על תרשים חשמלי בודד.



איור 6 - תרשים חשמלי עקרוני של החיישנים הדיגיטליים.

הesp8266 הינו הרכיב שמחבר בין רכיבי המעגל. רכיבים כמו החיישן נפח שצורכים מתח הזנה של 5v מחוברים אל ה-step up. מוצא החיישן נפח מפיק מתח של 3.3v כאשר ישנה תנועה. וכאשר אינו מזהה תנועה אז מתח של 0v לכן הדק זה מחובר לפין דיגיטלי של ה-ESP8266. המתג המאולתר של גילוי מצב הדלת גם עובד על אותו עיקרון כאשר הדלת סגורה היא תקצר בין 2 הדקים לכן גם חיישן זה יחובר להדק דיגיטלי. החיישן טמפרטורה הוא חיישן ספרתי המעביר מידע דרך 1-wire (הסבר פרוטוקול בנספחים). לכן הדק זה גם יחובר להדק דיגיטלי. ה-ESP8266 קורא את המידע שנותנים החיישנים ושולח את המידע אל המערכת המרכזית. ליחידות אלו המתח מתקבל ממייצב מתח שמתואר באיור 3.6.



איור 7 - תרשים חשמלי של ה-Power Supply

עקב החיסרון של הסוללות במידה ויתפרקו מתחת למתח של 2.8v אז הסוללה עלולה להיהרס לכן בחרנו לבחור ולבנות מתג אלקטרוני. בחרנו בשמיט מפני רוחב החשל שלא תהיה השוואה עם ערך מתח אחד.

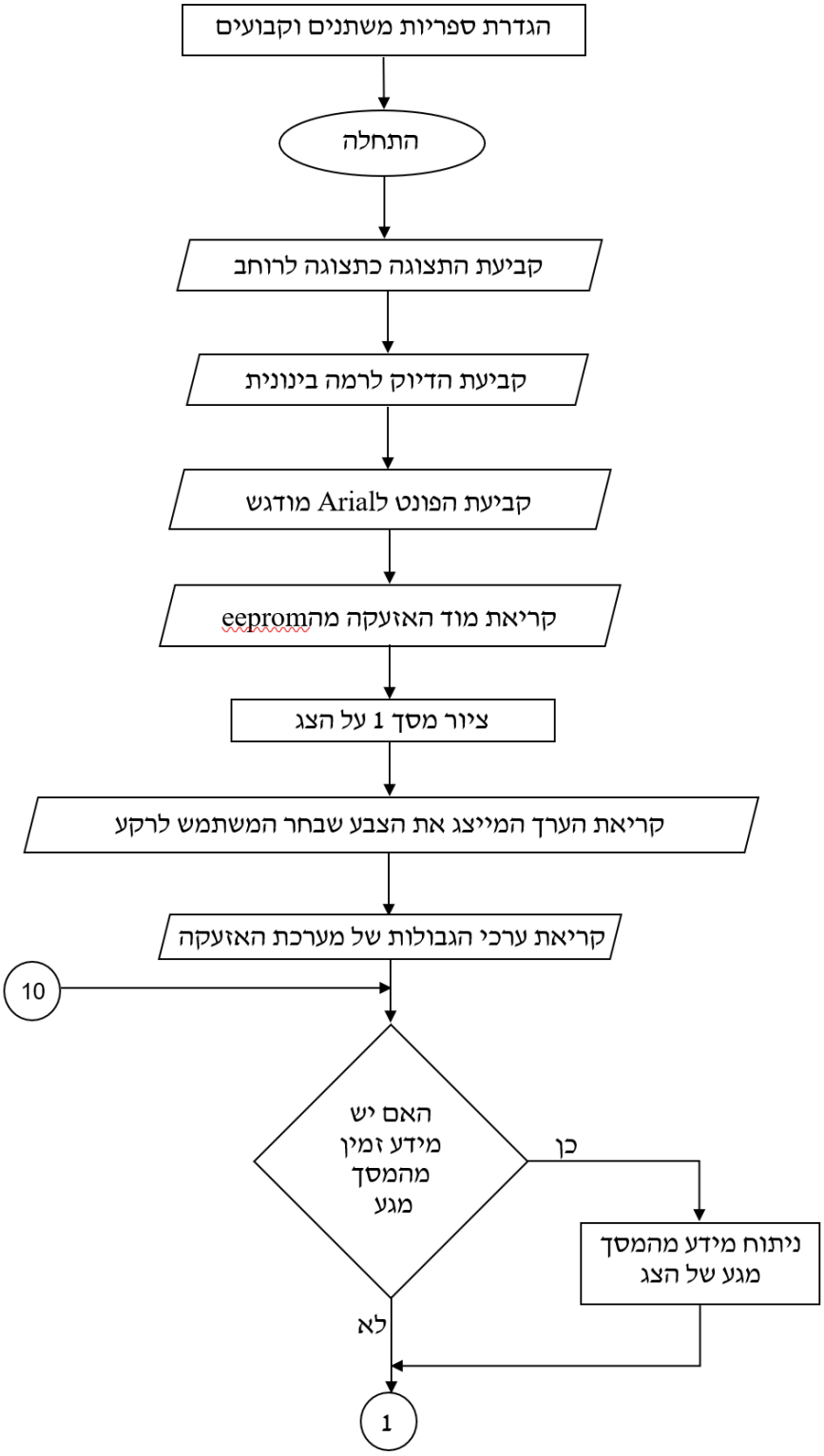
מעגל זה נועד למיתוג הסוללה עם המעגלים החשמליים, כאשר מתח הסוללה נמוך המעגל לא יקבל את מתח הסוללה וכאשר הסוללה טעונה המעגל יקבל את מתח הסוללה. לשם כך אנחנו משתמשים ב-MOSFET P-Channel הדק ה-Source מחובר ישירות אל הסוללה, הדק ה-Gate מחובר אל מוצא השמיט , השמיט בנוי כך שעבור תחום מתחים מוצא השמיט ישתנה בתחום ערכים ספציפי (פירוט בנספחים). כאשר מתח הסוללה גדול מוצא השמיט הינו נמוך לכן יש הפרש פוטנציאלים בין הדק ה-Source להדק ה-Gate לכן הדק ה-Drane יקוצר עם הדק ה-Source ומתח הסוללה יעבור לשאר המעגל. וכאשר מתח הסוללה נמוך השמיט במוצאו יוציא גבוהה לכן בין הדק ה-Source להדק ה-Gate יש הפרש פוטנציאלים נמוך, לכן הדק ה-Drane ינותק מהדק ה-Source ולכן הסוללה לא תחובר לשאר המעגל כשהיא פרוקה. בנוסף למעגל המיתוג ישנו מחלק מתח שמחובר אל מתח הסוללה שנועד להקטין את מתח הסוללה כך שה-ESP8266 יוכל למדוד ערך זה. החוצץ נועד כדי שהתנגדויות של המערכות הבאות לא ישפיעו על מחלק המתח.

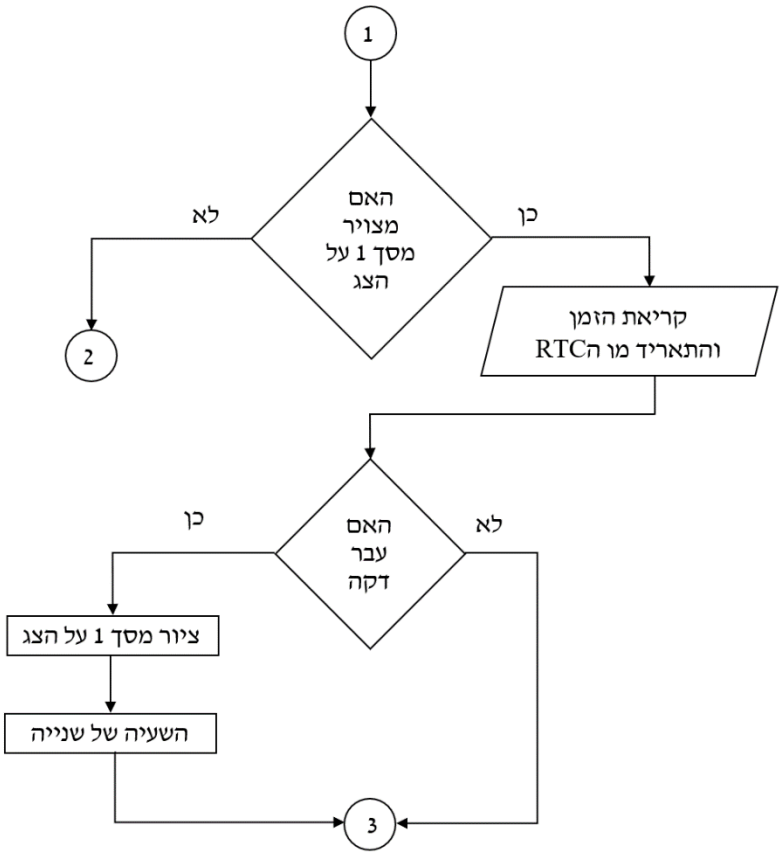
פרק 4

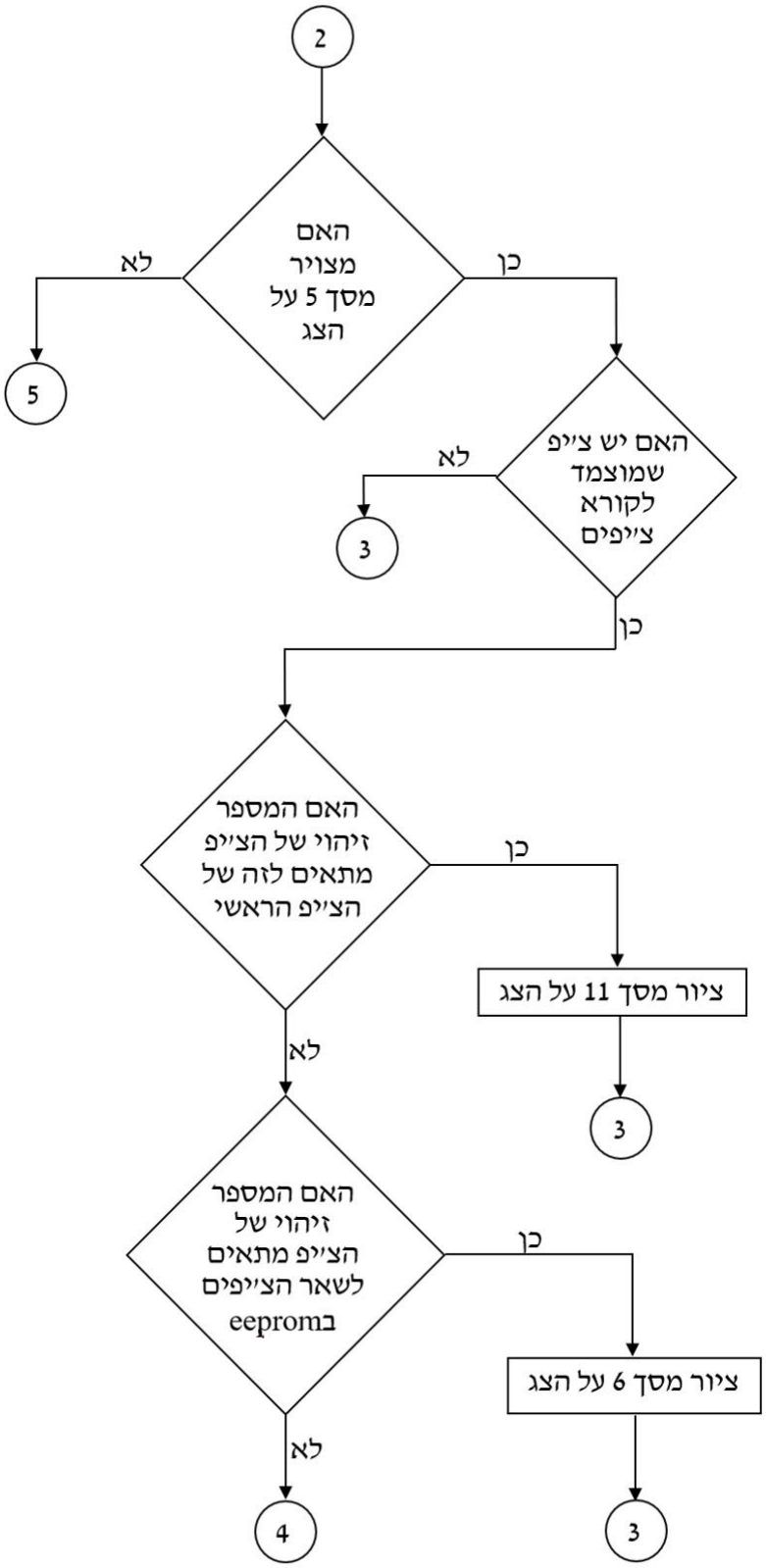
תרשים

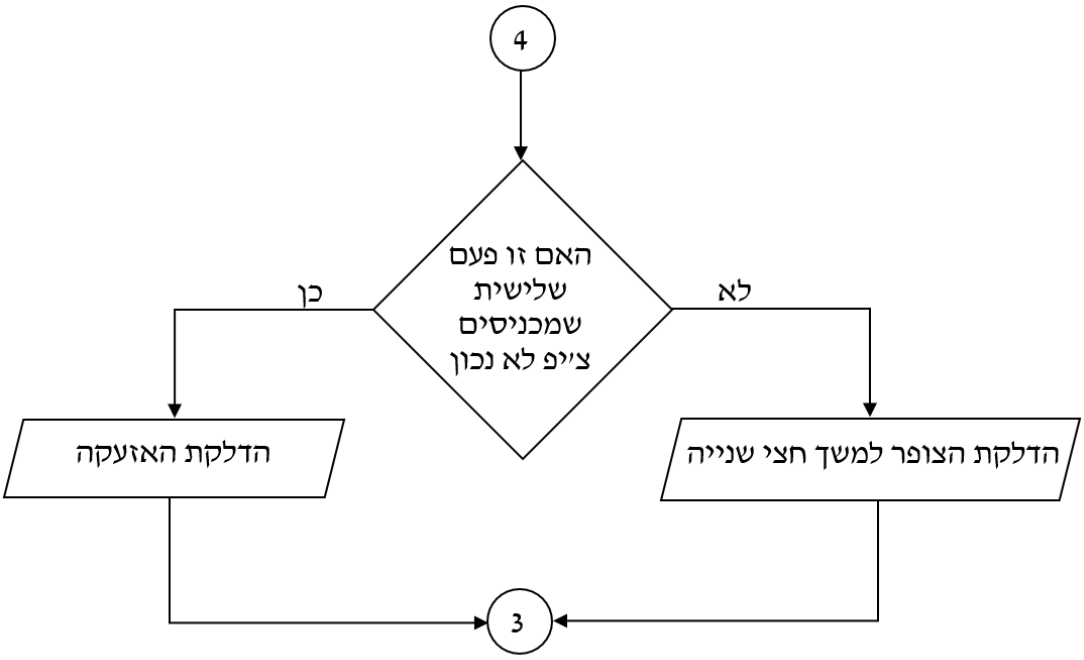
זרימה

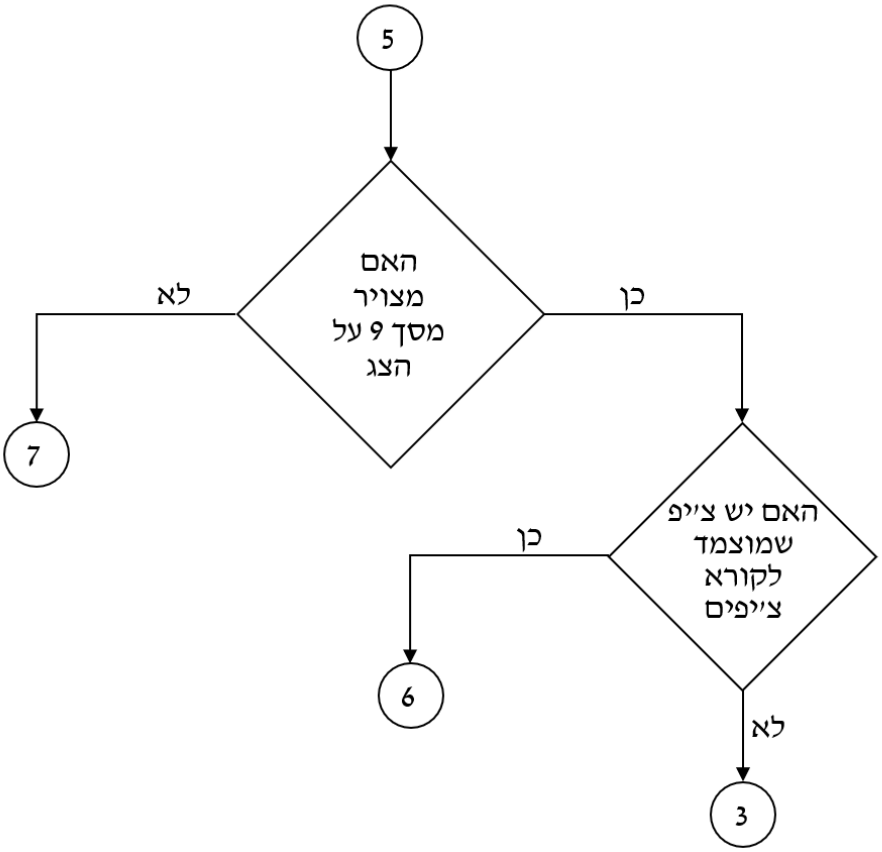
4.1 תרשים זרימה של התוכנית הראשית ביחידת ממשק המשתמש:

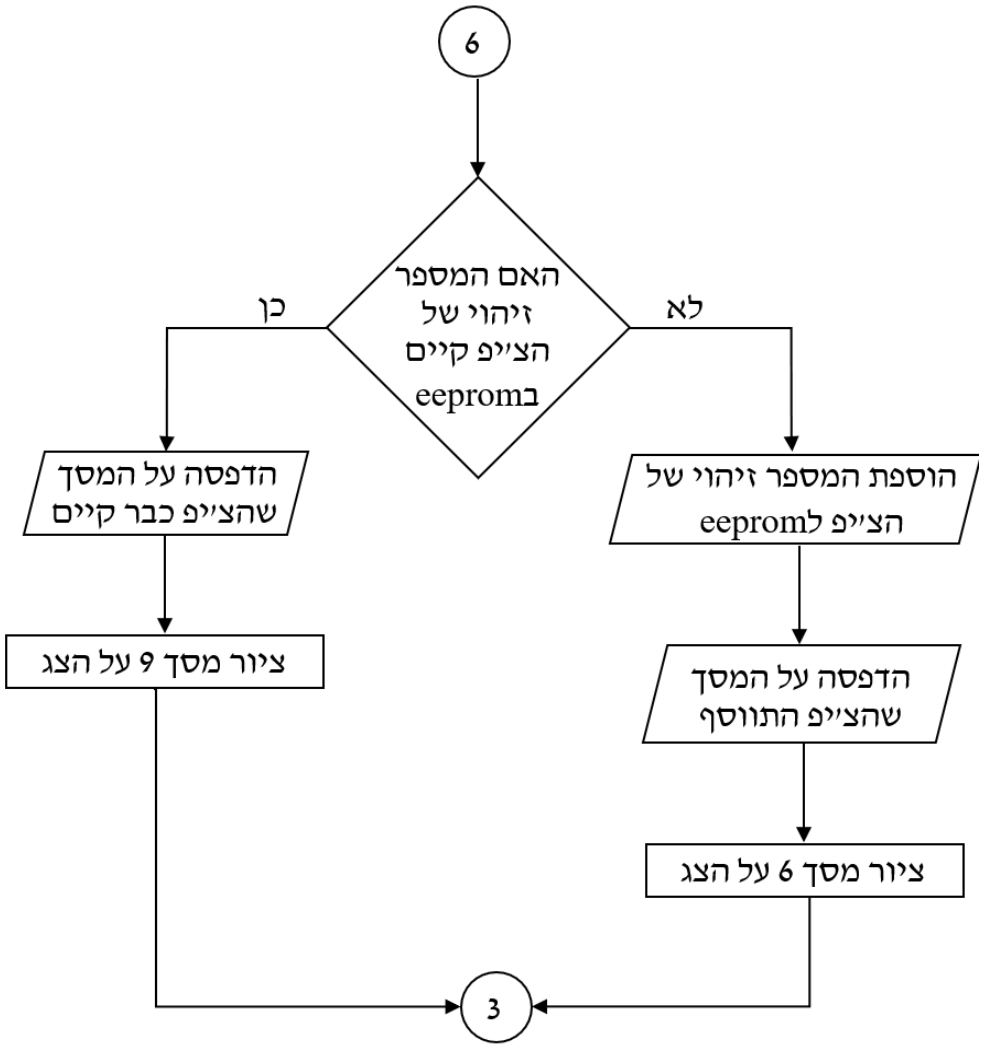


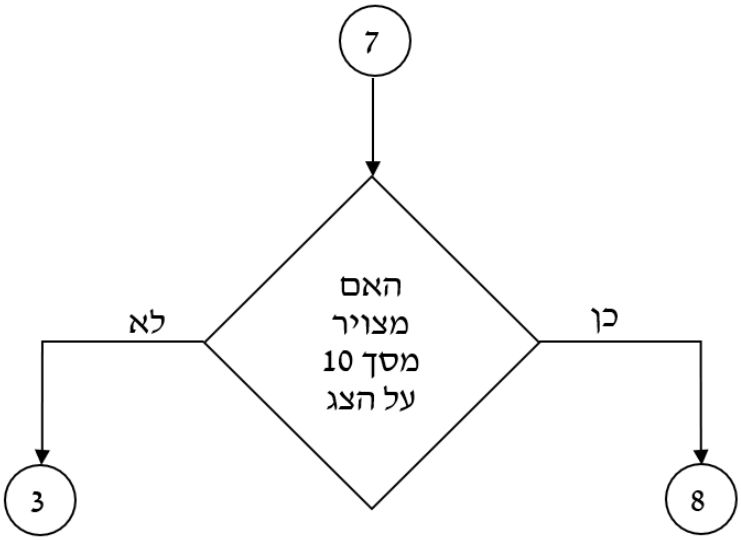

1
האם
מצויר
מסך 1 על
הצג
לא
כן
2
קריאת הזמן
והתאריך מו הRTC
האם
עבר
דקה
כן
לא
ציור מסך 1 על הצג
השעיה של שנייה
3

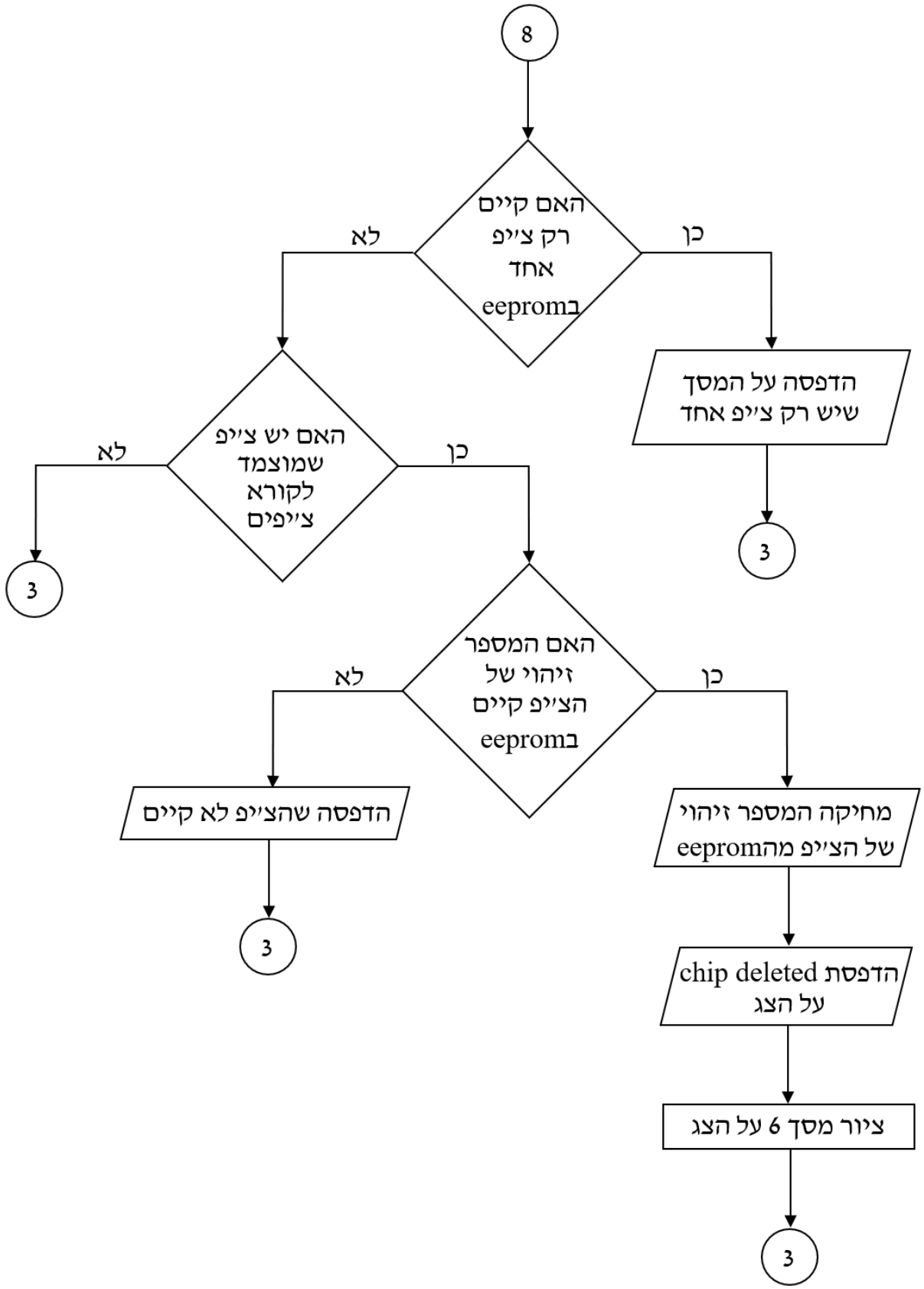

2
האם מצויר מסך 5 על הצג
לא
כן
5
האם יש צ׳יפ שמוצמד לקורא צ׳יפים
לא
3
כן
האם המספר זיהוי של הצ׳יפ מתאים לזה של הצ׳יפ הראשי
כן
ציור מסך 11 על הצג
3
לא
האם המספר זיהוי של הצ׳יפ מתאים לשאר הצ׳יפים בeeprom
כן
ציור מסך 6 על הצג
3
לא
4

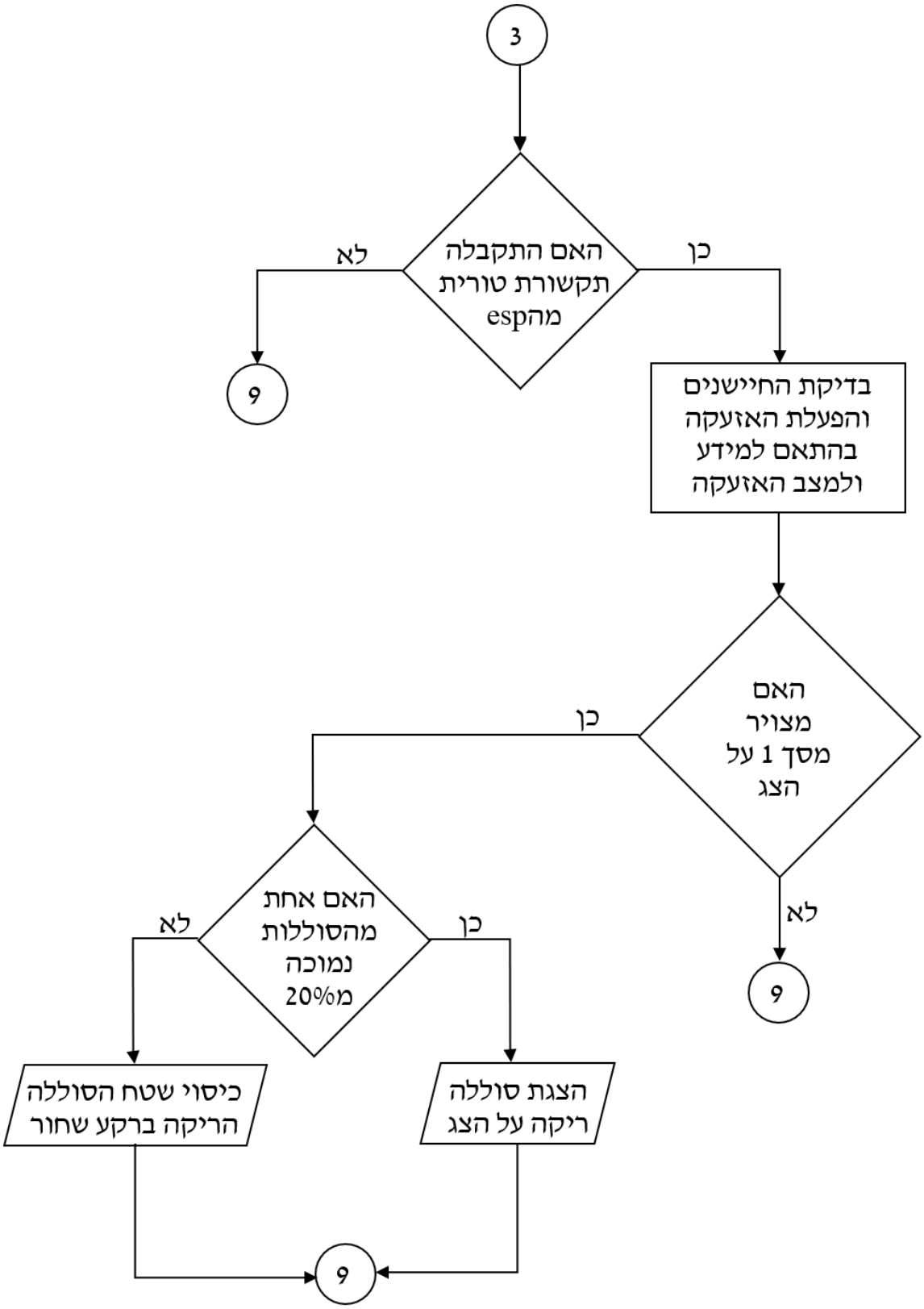

4
האם זו פעם שלישית שמכניסים צ'יפ לא נכון
כן
לא
הדלקת האזעקה
הדלקת הצופר למשך חצי שנייה
3

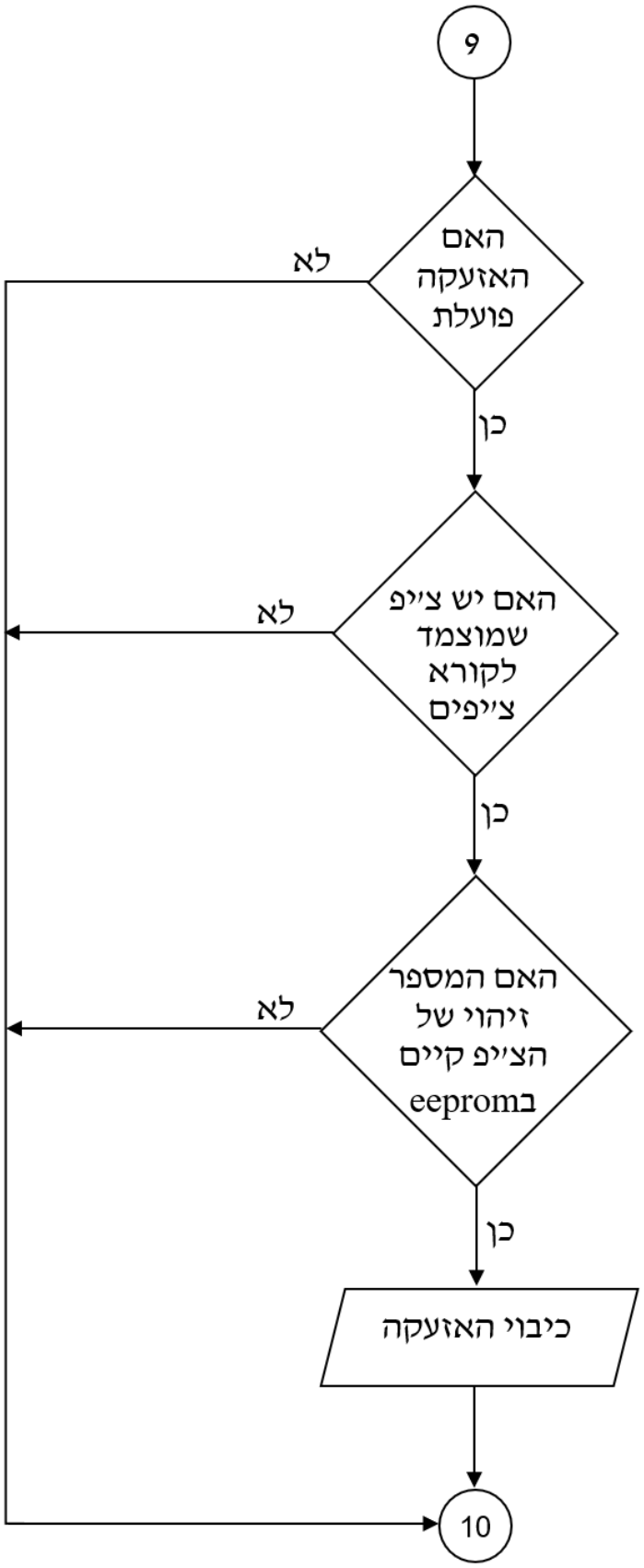

5
האם מצויר מסך 9 על הצג
לא
כן
7
האם יש צ'יפ שמוצמד לקורא צ'יפים
כן
6
לא
3

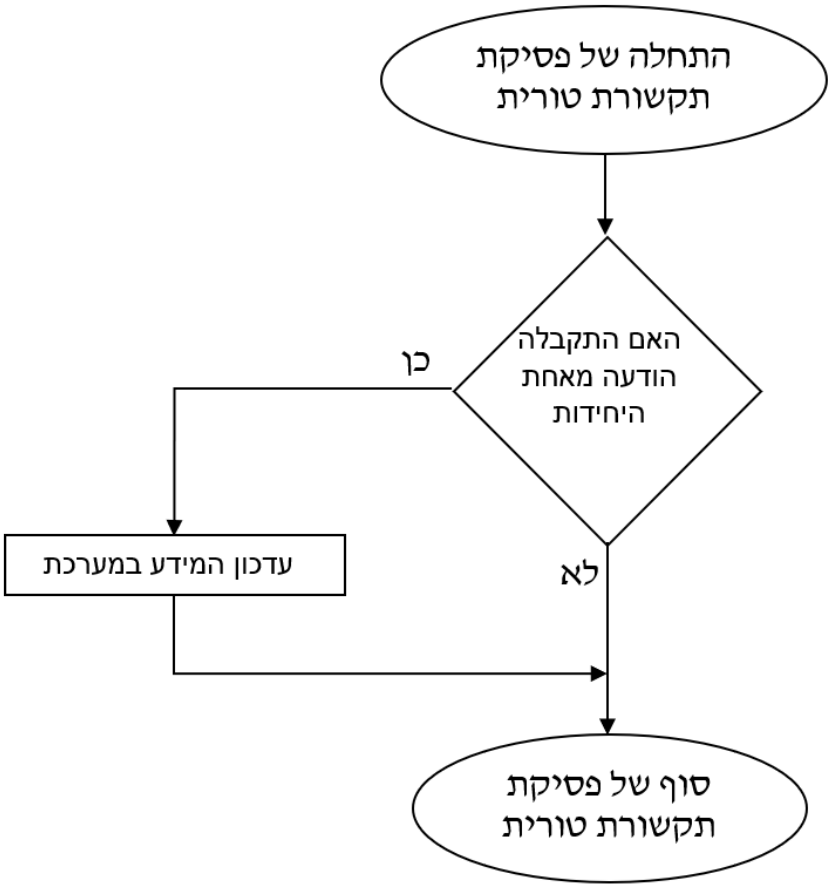

6
האם המספר זיהוי של הצ'יפ קיים בeeprom
כן
לא
הדפסה על המסך שהצ'יפ כבר קיים
הוספת המספר זיהוי של הצ'יפ לeeprom
ציור מסך 9 על הצג
הדפסה על המסך שהצ'יפ התווסף
ציור מסך 6 על הצג
3


7
האם מצויר מסך 10 על הצג
לא
כן
3
8

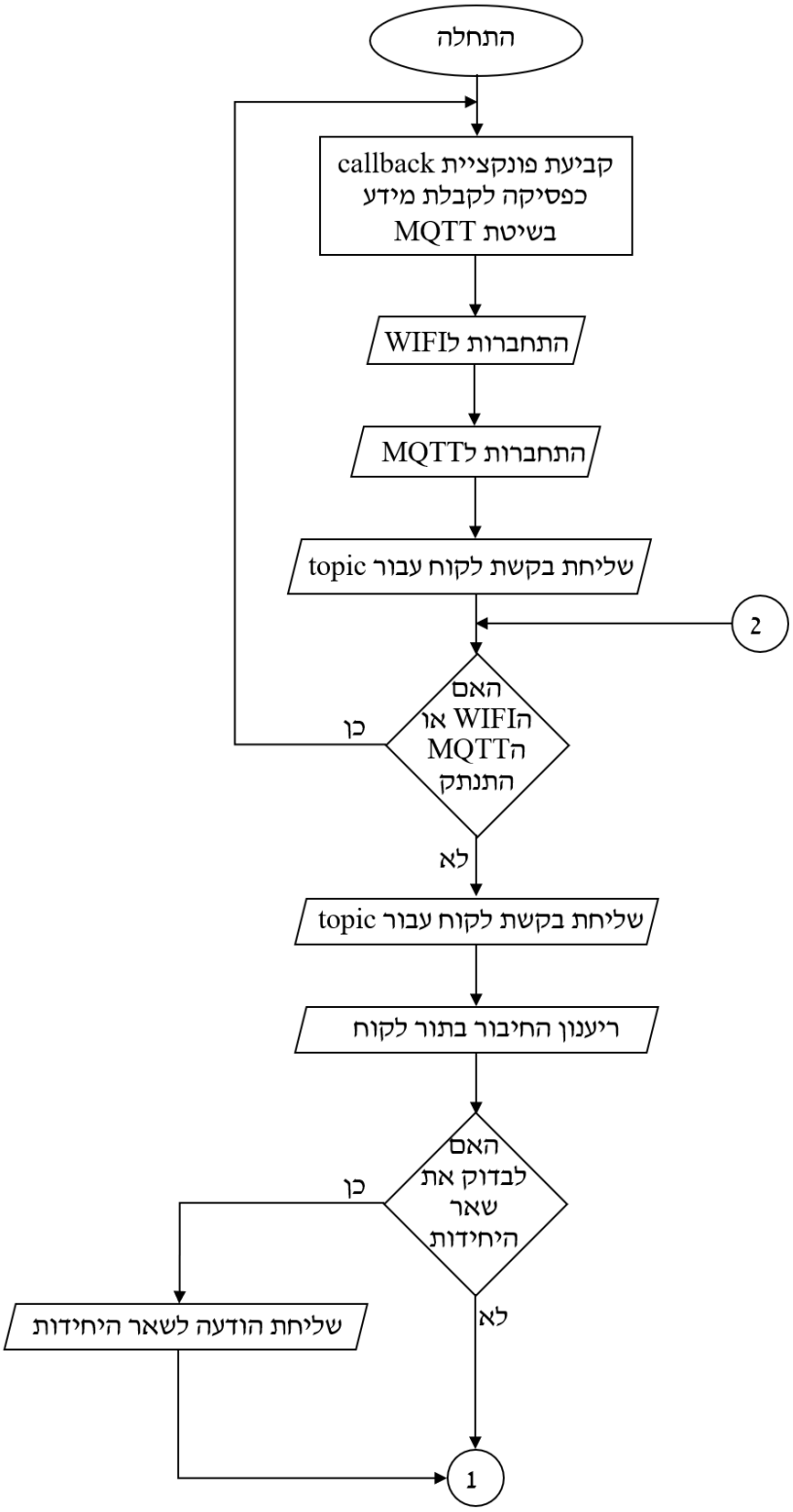

8
האם קיים רק צ'יפ אחד בeeprom
כן
לא
הדפסה על המסך שיש רק צ'יפ אחד
3
האם יש צ'יפ שמוצמד לקורא צ'יפים
כן
לא
3
האם המספר זיהוי של הצ'יפ קיים בeeprom
כן
לא
מחיקה המספר זיהוי של הצ'יפ מהeeprom
הדפסה שהצ'יפ לא קיים
3
הדפסת chip deleted על הצג
ציור מסך 6 על הצג
3

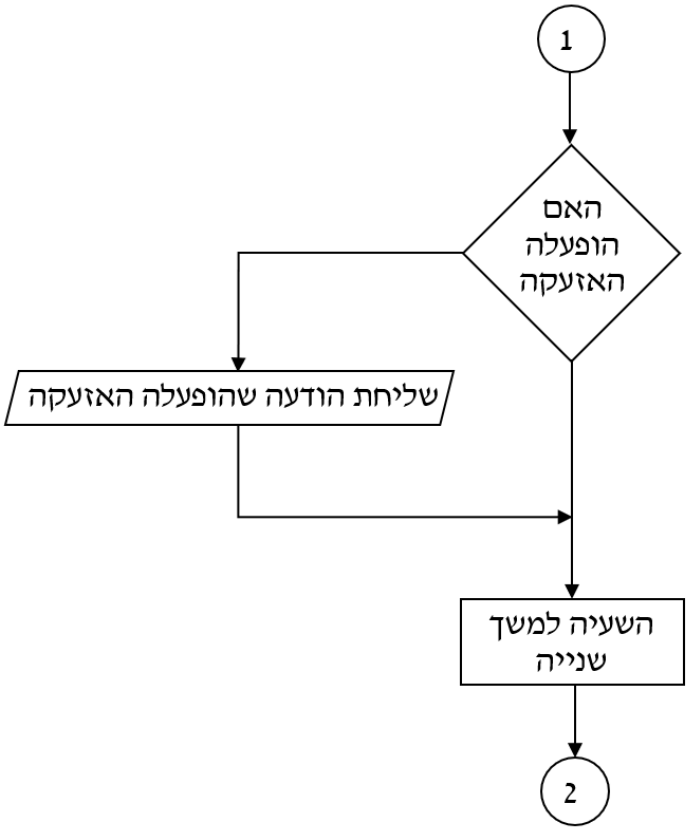

3
האם התקבלה תקשורת טורית מהesp
לא
כן
9
בדיקת החיישנים והפעלת האזעקה בהתאם למידע ולמצב האזעקה
האם מצויר מסך 1 על הצג
כן
לא
9
האם אחת מהסוללות נמוכה מ20%
לא
כן
כיסוי שטח הסוללה הריקה ברקע שחור
הצגת סוללה ריקה על הצג
9

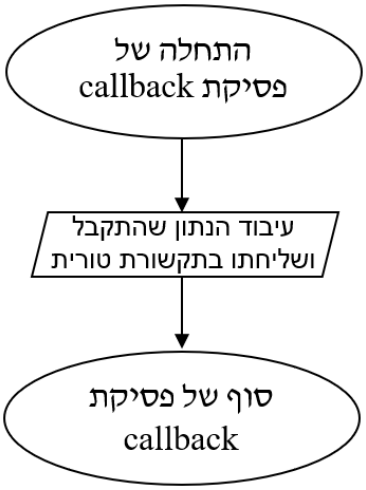

9
האם האזעקה פועלת
לא
כן
האם יש צ'יפ שמוצמד לקורא צ'יפים
לא
כן
האם המספר זיהוי של הצ'יפ קיים בeeprom
לא
כן
כיבוי האזעקה
10


התחלה של פסיקת
תקשורת טורית
האם התקבלה
הודעה מאחת
היחידות
כן
עדכון המידע במערכת
לא
סוף של פסיקת
תקשורת טורית

4.2 תרשים זרימה של התוכנית ב-ESP8266 ביחידת ממשק המשתמש:

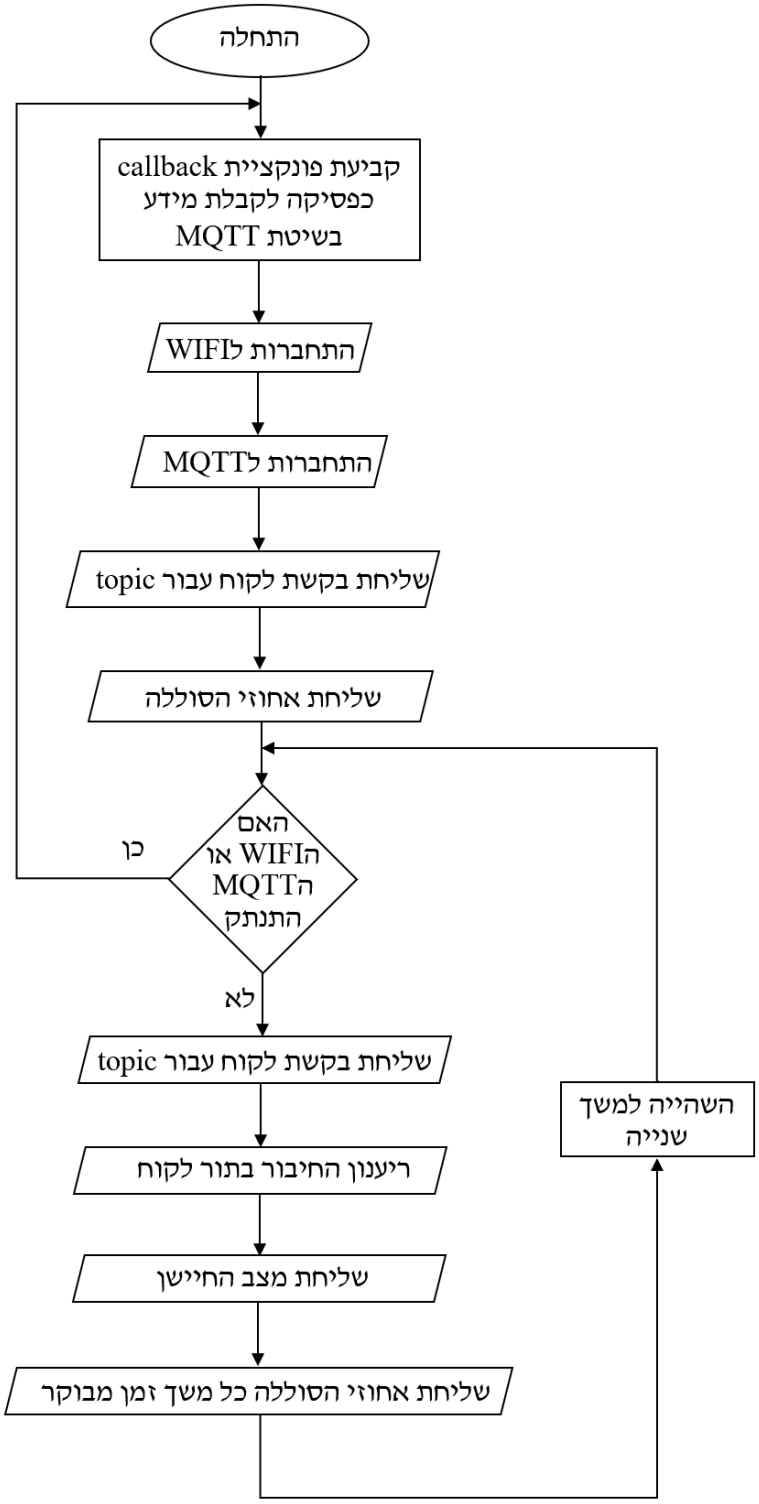


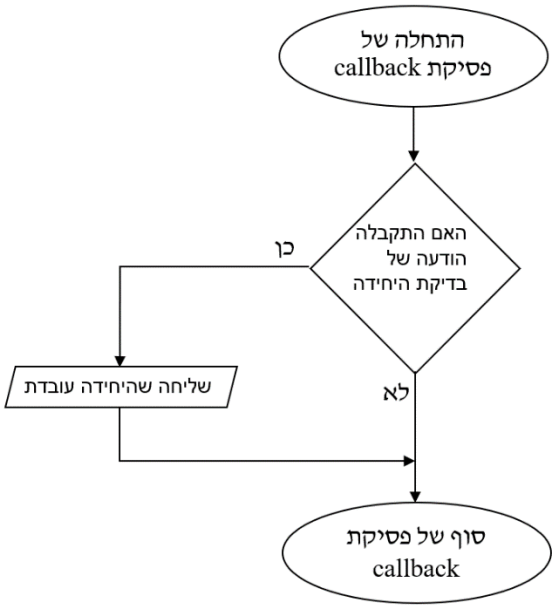

1
האם הופעלה האזעקה
שליחת הודעה שהופעלה האזעקה
השעיה למשך שנייה
2


התחלה של פסיקת callback
עיבוד הנתון שהתקבל ושליחתו בתקשורת טורית
סוף של פסיקת callback

4.3 תרשים זרימה של התוכנית ב-ESP8266 ביחידות החישה:

עקב מספר הרב של התוכניות ארשום תרשים אחד שיהיה עקרוני עבור כולם.




התחלה של
פסיקת callback
האם התקבלה
הודעה של
בדיקת היחידה
כן
שליחה שהיחידה עובדת
לא
סוף של פסיקת
callback

פרק 5

תוכנה

5.1 התוכנה שצרובה ליחידת ממשק המשתמש:

לקטע הקוד המלא הנכם מוזמנים לצפות בדיסק שמצורף בסוף הספר.

הסבר התוכנה:

תחילה אנו מגדירים את ההגדרות הראשוניות של הרכיבים כגון: מספרי ההדקים, ספריות, הגדרת משתנים מחרוזות ומערכים גלובליים. לאחר מכן קובעים קצב עבודה והגדרת פינים בתור הדקי מוצא.

בתוכנה ישנם מספר מסכים שיודפסו הם ממוספרים מ1 עד 15 בתוכנית (לדוגמה drawLcd3 זה מסך 3). בעזרת המסכים העבודה תהיה קלה למשתמש.

סדר המסכים:

מסך 1-הצגת שעה, תאריך, יום, מוד אזעקה ואיור של סוללה ריקה במידה ואחת הסוללות ריקה.

מסך 2- האם להיכנס להגדרות המערכת או לראות את פרטי המערכת.

מסך 3- במסך זה מוצגים כל הפרטים העדכניים של החיישנים ואפשרות להיכנס ולראות את אחוזי הסוללות.

מסך 4- במסך זה מוצגים כל אחוזי הסוללות

מסך 5- במידה והמשתמש רצה להיכנס להגדרות הוא ייכנס למסך זה ופה יוצג לו להכניס תג זיהוי , בהתאם לתג הוא יעבור למסך הבא.

מסך 6- במידה והוכנס צ׳יפ רגיל נגיע למסך זה. במסך זה ישנם אפשריות לשנות צבע רקע, להוסיף עוד צ׳יפ, למחוק צ׳יפ או לשנות את מוד האזעקה.

מסך 7- במסך זה מוצגים למשתמש המודים של האזעקה לבחירה.

מסך 8- במסך זה מוצגים הצבעים שיוכל לבחור במידה וירצה לשנות את הצבע של הרקע.

מסך 9- מסך זה יודפס כאשר המשתמש ירצה להוסיף צ׳יפ.

מסך 10- מסך זה יודפס כאשר המשתמש ירצה למחוק צ׳יפ.

מסך 11- למסך זה נגיע רק אם נצמיד את הצ׳יפ הראשי במסך 5 במסך זה נוכל לבחור אם לקבוע את גבולות מערכת האזעקה או לשנות שעה ותאריך או לבדוק את תקינות המערכת.

מסך 12-במסך זה ניתן לבחור עבור איזה חיישן נשנה את גבולות האזעקה.

מסך 13- מסך שבו מוצגים מספרים להכנסת הערך שנרצה לשנות.

מסך 14- במסך זה נוכל לבחור אם לבדוק את הזמזם או את שאר היחידות.

מסך 15- במסך זה יודפס אם ישנם יחידות פועלות או שלא פועלות.

בחלקו העיקרי של התוכנית נבדוק אם יש לחיצה על המסך ובהתאם לכך אם לבצע פעולה כלשהיא, בנוסף אם מצויר מסך 1 אז נעדכן כל פעם את השעה והתאריך על המסך. את המידע של החיישנים משאר היחידות נקבל בתקשורת טורית, לכן בתוכנית הוגדרה פסיקה עבור קליטת הנתונים. לאחר כל קבלת נתון יתבצע בדיקה שלהם, אם הם תואמים לגבולות האזעקה של המצב פעולה הקיים, במידה ולא תופעל האזעקה. אין צורך לבדוק כל פעם את המידע כי רק עבור כל קליטת נתונים המידע משתנה, בין קליטה לקליטה הערכים נשארים קבועים. כאשר האזעקה מופעלת יש להצמיד צ׳יפ (לא הראשי) כדי לכבות אותה. אם בתוכנית המשתמש נמצא במסך 9 או 10 ייבדק אם יש צ׳יפ שמוצמד במידה וכן הוא יתווסף או יימחק (בהתאם למסך).

הסבר ספריות:

MFRC522.h- ספרייה המכילה בתוכה פקודות לקריאת כל מידע שיש בצ׳יפ בעזרת תקשורת SPI. אם יש צ׳יפ זמין לקריאה או לא, פקודות לקריאת המספר זיהוי הצ׳יפ או את כל המידע. בנוסף בספרייה זו כבר הוגדר אופן התקשורת בSPI המוד עבודה סדר שליחת המידע ומהירות העברת המידע.

URTouch.h- ספרייה המכילה בתוכה פקודות שקשורות למסך מגע של התצוגה גרפית, פקודות לזיהוי נגיעה על המסך מגע ולמיקום הלחיצה. בקבצים המצורפים ישנה תוכנית לביצוע כיול המסך מגע אל הצג.

SPI.h- ספרייה זו מאפשרת לתקשר עם התקנים דרך תקשורת SPI.

ILI9341_due_config.h ו-ILI9341_due.h-ספריות המכילות פקודות הקשורות אל התצוגה, הגדרת אופן התקשורת בSPI, הדפסת טקסט רקעים מסגרות ומיקומם על הצג. קביעת גודל הטקסט, מיקום הטקסט על הצג קביעת צבע הטקסט ועוד.

Wire.h- ספרייה המכילה בתוכה פקודות המאפשרות לתקשר עם התקנים דרך תקשורת I2C.

DS1302.h- ספרייה המכילה פקודות לקריאת השעה והתאריך מן הרכיב ועדכון השעה והתאריך ברכיב הRTC.

טבלת פונקציות:

טבלה 3 - טבלת הפונקציות בתוכנית ביחידת ממשק המשתמש

שם הפונקציה	מה הפונקציה מקבלת	מה הפונקציה מחזירה	מה הפונקציה מבצעת
dayAsString	הערך מRTC שמסמל את שם היום בשבוע	את שם היום בשבוע בתור מחרוזת מסוג משתנה String	ממירה את הערך שמסמל את היום שמקבלים מן הRTC למחרוזת שניתן להדפיס על הצג.
drawLcd1	כלום	כלום	מדפיסה על הצג את השעה, היום והתאריך. את מוד האזעקה והאם הסוללות ריקות
drawLcd2	כלום	כלום	מדפיסה על הצג 2 מקשים 1.מידע 2. הגדרת המערכת
drawLcd3	כלום	כלום	מדפיסה על הצג את מצבי החיישנים ומקש של מצב הסוללות.
drawLcd4	כלום	כלום	מדפיסה על הצג את אחוזי הסוללות של המערכות
drawLcd5	כלום	כלום	מדפיסה שצריך להצמיד את התג זיהוי ואם הוא לא תקין
drawLcd6	כלום	כלום	מדפיסה 4 מקשים 1.מוד אזעקה 2.בחירת צבע 3.הוספת תג זיהוי 4.מחיקת תג זיהוי
drawLcd7	כלום	כלום	מדפיסה את כל מצבי האזעקה שניתן לבחור

drawLcd8	כלום	כלום	מציגה על המסך 11 מקשים עם צבעים שונים לבחירת המשתמש לשינוי גוון הרקע
drawLcd9	כלום	כלום	מדפיס להצמיד צ׳יפ חדש
drawLcd10	כלום	כלום	מדפיס להצמיד את הצ׳יפ שרוצים למחוק
drawLcd11	כלום	כלום	מדפיס 2 מקשים 1.הגדרת חיישנים 2.בדיקת המערכת
drawLcd12	כלום	כלום	מדפיסה את שמות היחידות שעבורן נרצה לשנות את הערך.
drawLcd13	כלום	כלום	מדפיסה לוח מקשי מספרים מ0 עד 9 וגם מקש Enter וClear בשביל שנוכל להכניס ערך לשינוי
drawLcd14	כלום	כלום	מדפיסה 2 מקשים 1.בדיקת הזמזם 2.בדיקת יחידות החיישנים
drawLcd15	כלום	כלום	מדפיסה אם שאר המערכות פועלות או שלא
drawReturn	כלום	כלום	מדפיסה מקש חזור בכל הפונקציות של drawLcd
colors	כלום	כלום	ממירה את המספר השמור בזיכרון ואת הגוון שבחר המשתמש בצג לצבע.
readUIDchip	כלום	1 במידה ונקרא צ׳יפ או 0 במידה ולא	קוראת את המספר זיהוי של הצ׳יפ
add_chip	כלום	כלום	מוסיפה לזיכרון צ׳יפ במידה ומצמידים אותו

			והוא עם מספר זיהוי חדש, במידה ומספר הזיהוי קיים תודפס הודעה בהתאם.
del_chip	כלום	כלום	מוחקת מהזיכרון צ׳יפ במידה ומצמידים אותו והוא עם מספר שקיים בזיכרון, במידה ואינו קיים תופיע הודעה בהתאם.
cheak_chip	כלום	כתובת התחלתית של מספר זיהוי הצ׳יפ או 0	בודקת אם מספר הזיהוי של הצ׳יפ קיים בזיכרון או שלא
writeEEPROM	1.כתובת ההתקן 2.הכתובת שאליה רוצים לשמור את המידע 3. מידע בגודל בית אחד	כלום	שומרת את המידע שהתקבל לפונקציה בכתובת שהתקבלה בזיכרון
readEEPROM	1.כתובת ההתקן 2.כתובת שבה רוצים לקרוא את המידע	מידע בגודל בית אחד	קוראת את המידע מהזיכרון מהכתובת שהתקבלה ומחזירה את המידע
waitForIt	הערכים של המקש שנלחץ	כלום	כל עוד המקש לחוץ המסגרת תהיה בצבע אדום ועוצרת את התוכנית עד לעזיבת המקש
runTime	כלום	כלום	הדפסת 30 שניות למשתמש בתור זמן כדי לצאת מהבית

			אם הגדיר במערת מצב שהוא מחוץ לבית
checkBatt	כלום	כלום	בודקת את אחוזי הסוללה משאר היחידות במידה ויש סוללה שקטנה מ20% היא תצייר סוללה ריקה
checkSensor	כלום	כלום	בודקת אם ערכי החיישנים לא מחוץ לגבולות שהוגדו בהתאם למוד האזעקה במידה וכן תופעל האזעקה
almLim	כלום	כלום	קריאת הערכים שהוגדרו כגבולות עבור מערכת האזעקה
serialEvent1	כלום	כלום	פסיקה לתקשורת טורית שהוגדרה כדי לקבל את המידע משאר היחידות.

5.2 התוכנית שצרובה ל-ESP8266 ביחידת ממשק המשתמש:

לקטע הקוד המלא הנכם מוזמנים לצפות בדיסק שמצורף בסוף הספר.

הסבר התוכנה:

תחילה נתחבר אל הנקודת גישה שהוגדרה, ולאחר מכן לMQTT. לאחר ההתחברות נשלח בקשה בשביל להיות מנויים עבור מספר נושאים (topics) ונגדיר את פונקציית callback כפסיקה לקבלת מידע בשיטת תקשורת MQTT. במידה וההתחברות הצליחה נמשיך לתוכנית במידה ולא יתבצע איפוס והתוכנית תתחיל מחדש. בתוכנית הראשית ייבדק מצב האזעקה ושליחת הודעה בהתאם לכך, בנוסף ייבדק אם יש דרישה לבדוק את שאר היחידות אזעקה במידה וכן יישלחו אליהם הודעת בדיקה, במידה ולא הוחזר תשובה במשך 20 שניות תישלח הודעה שלילית בתקשורת טורית אל ה ARDUINO DUE במידה והוחזרה תשובה שההתקן פעיל תישלח הודעה חיובית. בכל קבלה של הודעה בתקשורת MQTT תתחיל פונקציית callback לעבוד. בפונקציה זו ננתח את המידע שהתקבל נקצר אותו ונשלח אותו מקוצר בתקשורת טורית. בתוכנית יתבצע כל פעם ריענון בשביל להישאר לקוח עבור הנושאים שנרצה לקבל עבורם מידע.

הסבר ספריות:

PubSubClient.h- בספרייה זו קיימים הפקודות לתקשור הרכיב עם הברוקר דרך תקשורת MQTT, הפקודות לשליחת מידע, שליחת בקשת לקוח, קבלת מידע עבור נושאים שרצינו לקבל מידע עליהם.

ESP8266WiFi.h- בספרייה זו קיימים הפקודות כדי להתחבר לנקודת גישה של Wi-Fi שפרטיו הוגדרו בתוכנית. בנוסף פקודות על בדיקת מצבו של החיבור, אם החיבור עדיין קיים או שהתנתק.

טבלת פונקציות:

טבלה 4 - טבלת פונקציות של ה-ESP8266 ביחידת ממשק המשתמש

שם הפונקציה	מה הפונקציה מקבלת	מה הפונקציה מחזירה	מה הפונקציה מבצעת
Callback	3 ערכים : 1.הנושא (topic) שממנו התקבל המידע 2.כתובת התחלתית שבה נשמר המידע 3.אורך המידע	כלום	ניתוח המידע שהתקבל ושליחת מידע מקוצר דרך תקשורת טורית ל ARDUINO DUE
reconnect	כלום	כלום	התחברות אל הWi-Fi ואל הMQTT

5.3 התוכנה שצרובה ל-esp8266 שמחובר לחיישן גז:

לקטע הקוד המלא הנכם מוזמנים לצפות בדיסק שמצורף בסוף הספר.

הסבר התוכנה:

תחילה נתחבר אל הנקודת גישה שהוגדרה, ולאחר מכן לMQTT. לאחר ההתחברות נשלח בקשה בשביל להיות מנויים עבור מספר נושאים (topics) ונגדיר את פונקציית callback כפסיקה לקבלת מידע בשיטת תקשורת MQTT. במידה וההתחברות הצליחה נמשיך לתוכנית במידה ולא יתבצע איפוס והתוכנית תתחיל מחדש. בתוכנית הראשית נבדוק את מצב החיישן ונשלח את המידע בנושא (topic) משלו ובנוסף נבדק כל זמן מסוים אחוזי הסוללה במידה והם מעל 20% אז הבדיקה תהיה כל רבע שעה אחרת כל 5 דקות. במידה והתקבלה הודעה שהיחידה עובדת תישלח תגובה בהתאם. בתוכנית יתבצע כל פעם ריענון בשביל להישאר לקוח עבור הנושאים שנרצה לקבל עבורם מידע.

הסבר ספריות:

PubSubClient.h- בספרייה זו קיימים הפקודות לתקשור הרכיב עם הברוקר דרך תקשורת MQTT, הפקודות לשליחת מידע, שליחת בקשת לקוח, קבלת מידע עבור נושאים שרצינו לקבל מידע עליהם.

ESP8266WiFi.h- בספרייה זו קיימים הפקודות כדי להתחבר לנקודת גישה של Wi-Fi שפרטיו הוגדרו בתוכנית. בנוסף פקודות על בדיקת מצבו של החיבור, אם החיבור עדיין קיים או שהתנתק.

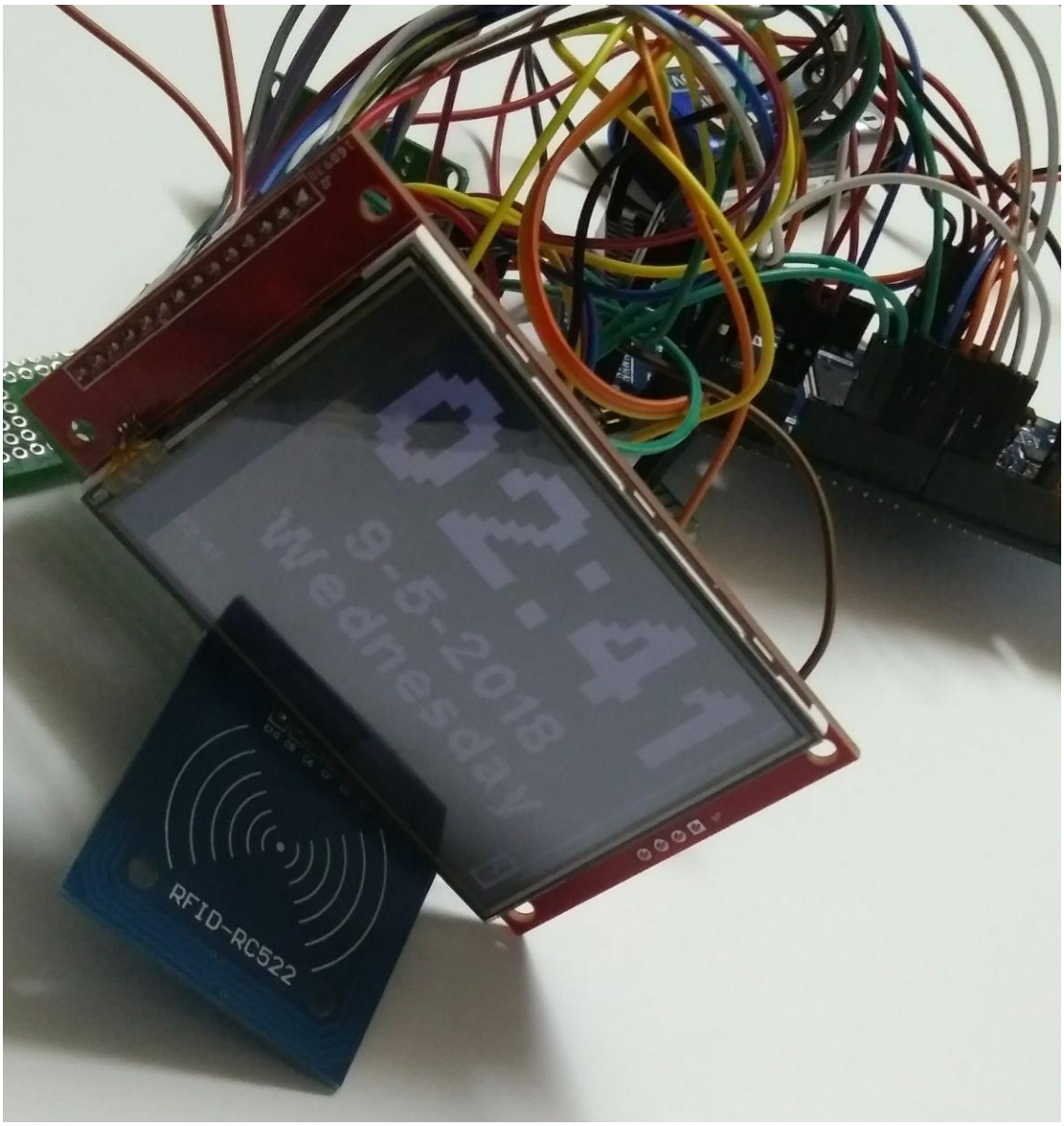
טבלת פונקציות:

טבלה 5 - טבלת פונקציות של אחת מיחידות החישה

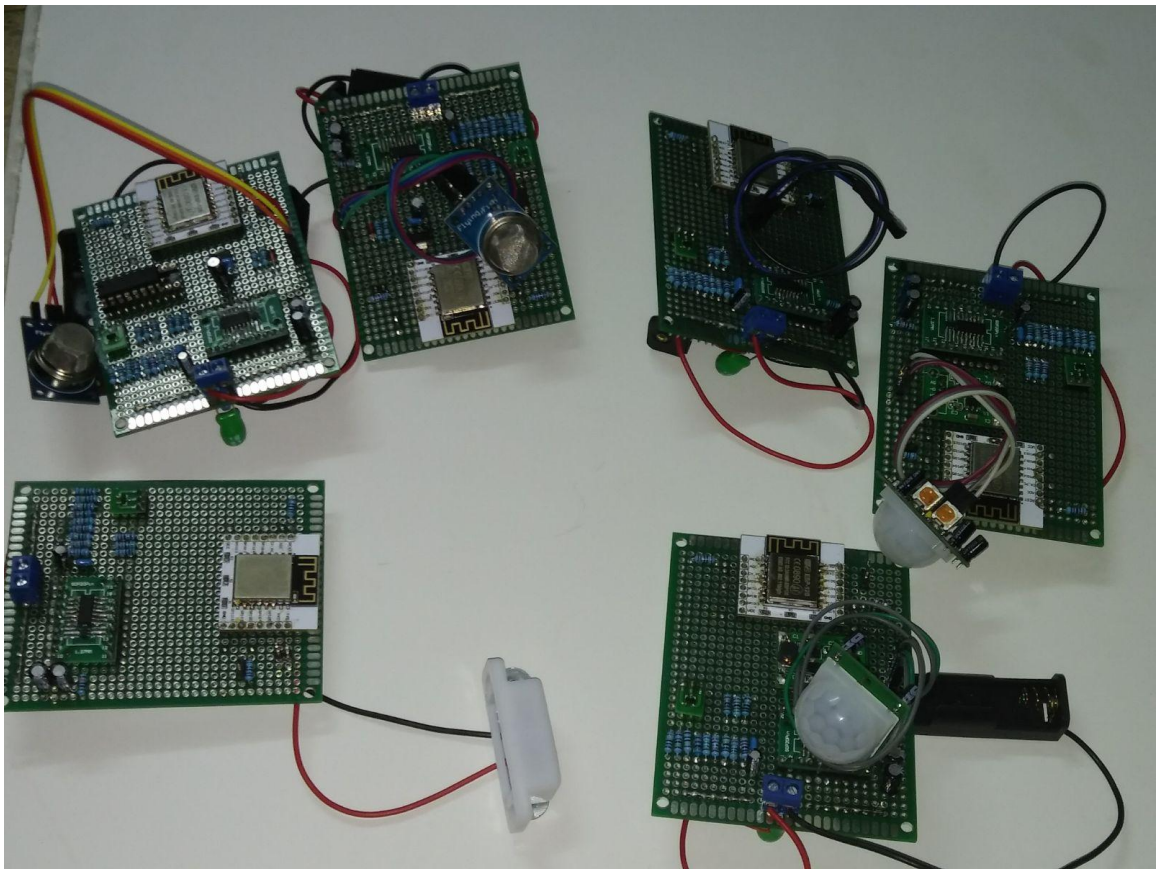
שם הפונקציה	מה הפונקציה מקבלת	מה הפונקציה מחזירה	מה הפונקציה מבצעת
Callback	3 ערכים: 1.הנושא (topic) שממנו התקבל המידע 2.כתובת התחלתית שבה נשמר המידע 3.אורך המידע	כלום	בדיקה אם התקבלה הודעת בדיקת תקינות היחידה ותגובה בהתאם
reconnect	כלום	כלום	התחברות אל הWi-Fi ואל הMQTT
battPerPub	כלום	כלום	בדיקה של אחוזי הסוללה ושליחת ההודעה בנושא בMQTT
senPerPub	כלום	כלום	בדיקת החיישן ושליחת המידע בנושא בMQTT
senPerPub	ערך של הסוללה או של החיישן	כלום	ממיר את הערך המספרי לתווים אסקיים

פרק 6

זיווג



איור 8 - ממשק למשתמש



איור 9 - חיישני החישה

פרק ז

סיכום

7.1 פרק תקלות במהלך העבודה:

במהלך העבודה נתקלנו בבעיה עם החיישני גז בישול וחיישן העשן שלא פעלו, ברגע חיבורם למעגל חלק מן הרכיבים כבו ולא עבדו לאחר בדיקה מעמיקה גילינו כי עבור מתח סוללה של $LiFePO_4$ לא מספק לרכיב step up מספיק מתח בשביל שיספק זרם לחיישן הגז בישול או העשן. פתרתי את הבעיה באמצעות חיבור שני סוללות ועם דיודת זנר למבוא המעגל.

בנוסף כאשר חיברתי את אחד ממעגלי יחידות החישה לספק, המתח שנמדד הינו היה נמוך ממה שנקבע בספק, חיבור של קבלי טנטלום במבוא יחידת החישה פתר את הבעיה.

7.2 פרק סיכום והסקת מסקנות:

העבודה עם הפרויקט הייתה מאוד מהנה, למדנו על רכיבים חדשים הכרנו איך לעבוד איתם ואת אופן פעולתם. לבצע תקשורת בין רכיבי ESP8266 דרך ה-Wi-Fi ולתקשר איתם דרך אפליקציה של MQTT.

החלק שאהבתי זה לתקשר עם ESP8266 עם אפליקציה (עבור אנדרואיד) ולשלוט על לד המחובר לאחד מהדקי ה-ESP8266

הצעות לפיתוח עתידי:
הוספה של מצלמת אבטחה שתצלם בזמן אמת את הבית ואנשי הבית יוכלו לצפות בה בכל זמן שירצו.

שימוש במעגלים מודפסים ורכיבים קטנים כדי שהמעגל יהיה קטן, כדי שיחידות החישה לא יבלטו ברחבי הבית.

פרק 8
ביבליוגרפיה

http://www.cobox-ebikes.com/296/basic-understanding-of-lipo-li-ion-and-lifepo4.html

https://www.hivemq.com/blog/mqtt-essentials/page/2/

https://www.mysensors.org/build/gas

https://www.arduino.cc/reference/en/

https://io.adafruit.com/dashboards

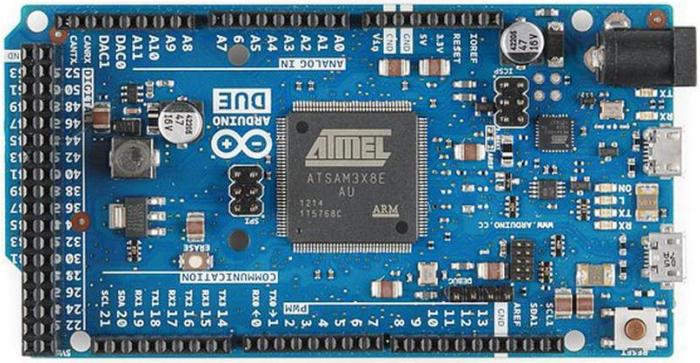
http://jansson.us/resistors.html

https://www.random-science-tools.com/electronics/inverting-schmitt-trigger-calculator.htm

פרק 9

נספחים

9.1 Arduino Due:



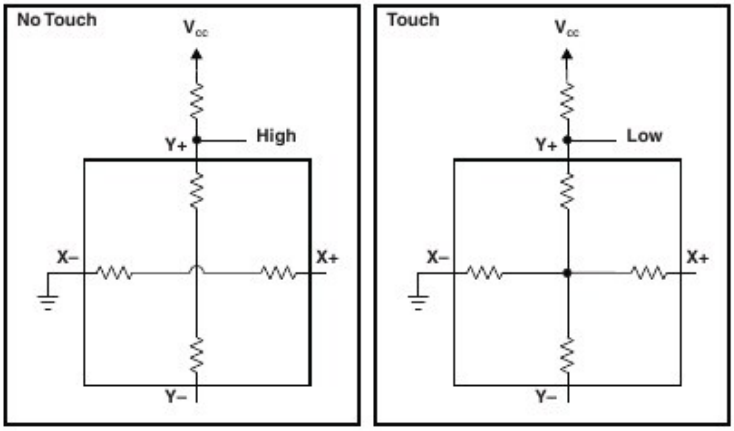
איור 10- Arduino Due

ערכת הפיתוח ARDUINO DUE הוא המוח המפעיל את רכיבי המערכת ביחידת התצוגה גרפית. בכרטיס זה נמצא המיקרו בקר AT91SAM3X8E המתבסס על מעבד CORTEX-M3 CPU. הבקר כולל: 54 חיבורים דיגיטליים המשמשים כקלט פלט, 12 הדקים אנלוגיים המשמשים כהדקי מבוא, 4 ערוצים של תקשורת UART, שעון של 84 MHZ, זיכרון FLASH בגודל של 512KB, כפתור RESET וכפתור למחיקה. בשונה משאר ערכות הפיתוח ערכה זו מתבססת על מתח של 3.3v. המתח הפעלה של ה-ARDUINO DUE הוא 7-12v.

9.2 תצוגה גרפית עם מסך מגע:

איור 11 - תצוגה גרפית

המסך הינו תצוגה גרפית עם מסך מגע בעל רזולוציה של 320x240 בעל 262 אלף צבעים. התצוגה גרפית היא תצוגת גביש נוזלי (LCD). ההתקשרות עם המסך היא באמצעות תקשורת SPI. לתצוגה גרפית הדרייבר הוא ili9341 ולמסך מגע הדרייבר הוא xpt2046, שני הדרייברים עובדים במתחי הפעלה של 3.3v. למסך יש זיכרון גרפי של 172,800 בתים כדי לאחסן את תמונת הצג. המסך מגע הינו מסך התנגדותי הכולל מספר שכבות, והחשובות שבהן הן שתי שכבות דקות, שקופות, עמידות לחשמל, המופרדות על ידי שטח דק. שכבות אלה פנים אל פנים עם מרווח דק ביניהם, לכל שכבה יש מוליכים בצדדים כאשר המוליכים בין שני השכבות מאונכות זו לזו ובכך שלוחצים על המסך משטחים אלה מוליכים ובכך ניתן לדעת את מיקום הלחיצה.
היתרון של מסך התנגדותי היא שעבור כל לחיצה עם לחץ מסוים. סוג זה של מסך הינו מסוג 4wire.



איור 12 - תיאור של מסך התנגדותי

9.3 צופר BUZZER:

איור 13 - יחידת הצופר

BUZZER הוא צופר בעל התנגדות של 1Kohm, ה-BUZZER יכול לקבל תחום מתחים של 3-12V, ולהפיק עוצמת רעש של עד 85dB. אנו בפרויקט מחברים את הדק המינוס לאדמה ואת הדק הפלוס להדק דיגיטלי של 11 המפיק 3.3V כאשר האזעקה צריכה לפעול. תדר התהודה של הצופר הוא 400 פלוס מינוס 100, ומשקלו כ-85g.
הרכיב עובד בטמפרטורה של כ-25-70 מעלות וניתן לאחסן אותו בטמפרטורה של 30- עד 80 מעלות.

9.4 שעון זמן אמת RTC DS1302:

איור 14 - RTC DS1302

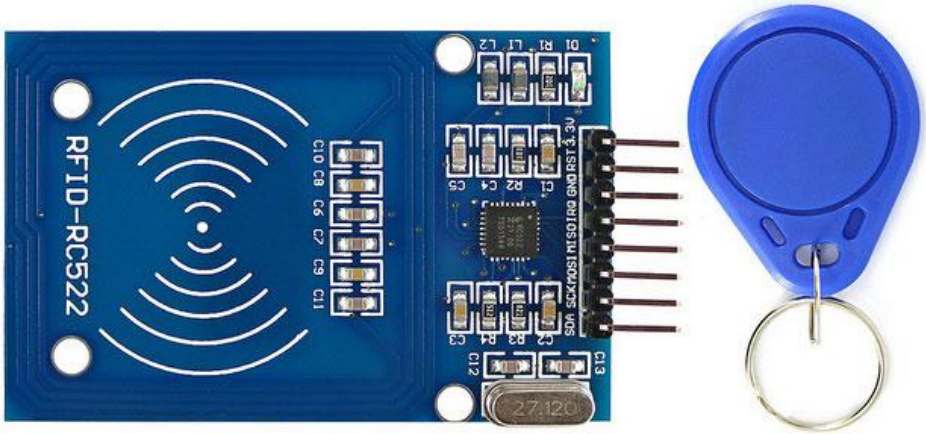
DS1302, הוא שעון זמן אמת הסופר : שניות, דקות, שעות, תאריך, חודש, יום בשבוע ושנה. השעון כולל סוללת גיבוי הפועלת גם כאשר אינה מחוברת לספק חיצוני המשמשת לרכיב להמשיך בפעולתו התקינה ולהגנת הזיכרון הנדיף RAM בעל 31 בתים. מתח הפעולה של הרכיב הוא בין 2v עד 5.5v , תחום המתחים הזה מגדיל את חיי הפעולה סוללות הגיבוי (המתח המגיע מה-ARDUINO DUE שלנו בפרויקט מעניק לשעון מתח של 5V). הזרם במתח של 2V הוא 300mA זרם זה הוא זרם נמוך מאוד מה שמאפשר חיי סוללה ארוכים עקב צריכת זרם נמוכה. השעון עובד בתחום טמפרטורה של 40- עד 85 מעלות. פעולת התקשורת של השעון עם ה- ARDUINO DUE מתרחשת בעזרת תקשורת טורית סינכרונית הממומשת בעזרת שלושה חיבורים :

CE- אפשור הרכיב, הדק זה פעיל בגבוה כאשר אנו רוצים לקרוא או לכתוב לשעון.

CLK- שעון סריאלי, תפקיד השעון הוא לתזמן את העברת המידע ולאפשר פעולה סינכרונית בין השעון ל-ARDUINO DUE או להפך.

DATA- הדק זה נועד לשם העברת המידע בין השעון ל-ARDUINO DUE ולהפך. המידע יכול להיות מעובר מהשעון בתור בית אחד, או בצרור של נתונים שמגיע עד ל-31 בית.

9.5 חיישן זיהוי צ׳יפים RFID:



איור 15 - קורא צ׳יפים וצ׳יפ

RFID הינם ראשי תיבות של Radio Frequency Identification, בעברית זיהוי באמצעות תדרי רדיו, המאפשרת איתור פריטים וזיהוי חפצים בצורה אלחוטית, ללא צורך בשדה ראייה או במגע ישיר מול החפץ אותו רוצים לזהות.

טכנולוגית ה-RFID הינה טכנולוגיה ותיקה שגרסאות שלה התפתחו עוד בשנות ה-20, והטכנולוגיה בצורתה הנוכחית התפתחה כבר בסוף שנות ה-60. אך טכנולוגיית ה-RFID החלה לצבור תאוצה וקהל לקוחות רחב רק בשנים האחרונות. זאת מכיוון שבעבר, טכנולוגיית ה-RFID הייתה מאוד יקרה, טווחי הקריאה היו מצומצמים ועלות התגים הייתה מאוד גבוה.

כיום, עם התפתחויות הטכנולוגיות האחרונות ומזעור השבבים, עלות התגים ירדה משמעותית, מרחקי הקריאה השתפרו וגדלו, והטכנולוגיה מסוגלת ליותר יישומים מעשיים כמו : אחסנה, ניהול מלאי, לוגיסטיקה, מחסנים ממוחשבים, רפואה ועוד.

טכנולוגיית ה-RFID משתמשת בשדה אלקטרומגנטי הנוצר ע״י קורא קרבה (RFID reader) להעברת אנרגיה לתגי הקרבה (RFID tags) בטווחים קצרים.

בעזרת האנרגיה שנוצרה ע״י השדה, תגי הקרבה מסוגלים ללא צורך במקור מתח חיצוני ומשדרים לקורא קרבה את תוכן הזיכרון הצרוב בתוכם.

סוגי תגי RFID:

ישנם שלושה סוגים של תגיות RFID:

1. **תג פאסיבי** - תג ללא מקור מתח משל עצמו (סוללה). זהו התג שאנו משתמשים בפרויקט.
2. **תג אקטיבי** - תג בעל מקור מתח משל עצמו (סוללה), בעל טווח גדול יותר וזיכרון גדול יותר מתג פאסיבי.
3. **תג פאסיבי למחצה** – דומה לתג פאסיבי אך יש בו סוללה קטנה יותר מהאקטיבי שמאפשרת לו להיות טעון תמידית.

תדרים בשימוש טכנולוגיית ה-RFID:

ישנם ארבעה אורכי גל מרכזיים הנמצאים בשימוש RFID:

1. תגי אורך גל נמוך- 125KHz או 134.2KHz.
2. תגי אורך גל גבוה - 13.56MHz (אורך הגל שלנו)
3. תגי אורך גל אולטרה גבוה (UHF) - 868MHz או 959MHz.
4. תגי גלי מיקרו - 2450MHz ואפילו יותר

אופן פעולת ה-RFID פרוייקט שלנו:

אנו משתמשים ב-RFID בפרויקט שלנו לשם זיהוי כניסת המשתמש לבית, או האם מדובר בכניסת הטכנאי למערכת. דרך האנטנה משודרת אנרגיה תדרי רדיו, התגית מקבלת אנרגיה זו ומחזירה ל-RFID דרך האנטנה בתדרי רדיו את הקוד הצרוב על התגית, ה-RFID מעביר את הנתונים אל ה-ARDUINO DUE בתקשורת SPI, ולאחר מכן ה-ARDUINO DUE מעביר בתקשורת SPI אל הצג האם הצ'יפ מתאים או לא, ובכך המשתמש יודע האם ניתנת לו הרשאה לבית.

9.6 בקר ESP8266-12S:

איור 16 - ESP8266

בקר ESP8266-12S, הוא רכיב בר תכנות באמצעות סביבת העבודה ARDUINO הניתן להתחברות לרשתות WIFI בעזרת סטנדרט אלחוטי של 802.11 b/g/n. בקר זה בעל 32 סיביות הכולל TCP/IP, מהירות שעון של 80MHz ו-160MHz, מופעל במתחים של 3.3v עד 3.7v, וניתן לתפעול בטמפרטורה של 40- עד 85 מעלות צלזיוס. באמצעות רכיב זה אנו יכולים לתקשר עם כל המערכות בפרויקט כאשר לכל מערכת קיים בקר ESP8266-12S משלה, והתקשורת מתבצעת בצורה אלחוטית באמצעות פרוטוקול MQTT (ראה בנספחים). הרכיב כולל: תשעה הדקים דיגיטליים, הדק אנלוגי אחד, ערוץ לתקשורת UART, הדק לאפשור הרכיב והדק איפוס.

9.7 יחידת זיכרון (EEPROM) AT24C64:

איור 17 - יחידת הזיכרון

רכיב זה הינו זיכרון מסוג EEPROM:

רכיב זה הינו שבב אחסון מידע שלא משתנה לעתים קרובות, אשר לו שימושים במחשבים והתקנים אחרים. בניגוד ל-EPROM שבב מסוג EEPROM ניתן למחיקה ותכנות מחדש מספר פעמים באופן אלקטרוני.

הוא ניתן לתכנות מספר סופי של פעמים, בדרך כלל בין 100 אלף למיליון פעמים, אך הוא ניתן לקריאה מספר בלתי מוגבל של פעמים, ויכול לשמור את הנתונים בתוכו 100 שנים. בתעשייה, ישנה מוסכמה המשמרת את מושג ה-EEPROM לזיכרונות ניתנים לכתיבה הפועל ברמה של סיביות (bitwise) ולא לזיכרונות הבזק שניתנים לכתיבה הפועלים ברמה של בלוקים (blockwise).גודלו של זיכרון זה הוא 65536 ביט של זיכרון טורי הניתן למחיקה ולשמירת ערכים חדשים ולקריאתם. המידע מחולק ל-8 ביט של מילים כל אחד. זיכרון הרכיב הוא בלתי נדיף מה שאומר שלאחר כיבוי רכיב, הנתונים עדיין נשארים שמורים. רכיב זה פועל באמצעות פרוטוקול I2C (מפורט בנספחים).

תיאור הדקי הרכיב:

SCL (Serial Clock)

הדק כניסת השעון משמשת להכניס נתון לרכיב בזמן עליית השעון ולהוציא נתון מהרכיב בזמן הירידה של השעון.

SDA (Serial Data)

זהו הדק דו כיווני להעברת נתון טורי.רגל זו היא בחיבור (open drain) ויכול להתחבר ב WIRE ORed- עם מספר נוסף של רכיבים בחיבור open drain אוopen collector .

WP (Write Protect)

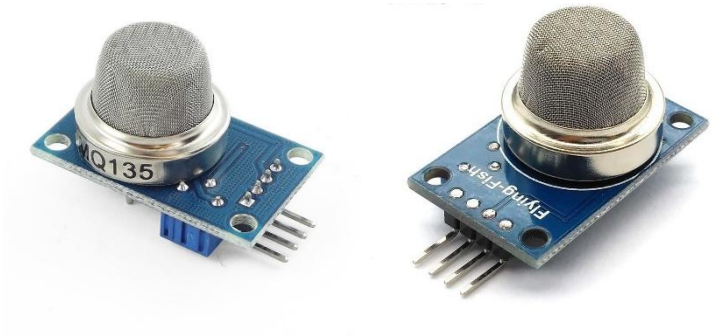
הדק זה נותן הגנת נתונים בחומרה. הרגל מאפשרת פעולת קריאה/כתיבה רגילה כאשר היא מחוברת לאדמה. כאשר נחבר אותה ל VCC כל פעולות הכתיבה של הרביע העליון (16K bit)

של הזיכרון מעוכבים. אם חיבור WP נשאר לא מחובר, החיבור מתחבר פנימית ל-GND. בפרויקט שלנו הדק זה מחובר לאדמה.

A0 A1 A2 DEVICE ADDRESSES:

ניתן לחבר עד 8 רכיבים של EEPROM תחת תקשורת I2C , לכן לכל רכיב EEPROM חייבת להיות כתובת ייחודית של עצמה. בעזרת חיבורם של הדקים הללו ל-GND או ל-VCC ניתן לקבוע את כתובתו של הרכיב כאשר A2 הוא ה-MSB וA0 הוא ה-LSB. כתובת ברירת המחדל של הרכיב כאשר כל ההדקים הללו מחוברים לאדמה היא 0x50 לכן אם נחבר את A2 ל-VCC, ואת A0 ו-A1 ל-GND, נקבל 4 זאת אומרת כתובת 0x54. בפרויקט שלנו חיברנו את ההדקים הללו לאדמה כי אצלנו רק רכיב EEPROM אחד כלומר כתובת 0x50.

9.8 גלאי גז בישול MQ6 וגלאי עשן MQ135:



איור 18- חיישן גז וחיישן עשן

החומר הרגיש של החיישן גז MQ-6 הוא גז בישול LPG ורגישות נמוכה לבנזין, והחומר הרגיש של גלאי MQ135 הוא עשן . במצב של אוויר נקי המוליכות היא נמוכה.

כאשר הגלאים מזהים חומר דליק המוליכות של החיישנים נעשית גבוה יותר יחד עם ריכוז הגז העולה.

לרכיבים שאנו משתמשים יש שתי יציאות, אחת אנלוגית ואחת דיגיטלית. בפרויקט שלנו אנו משתמשים ביציאה האנלוגית שתחובר לבקר ESP 8266 דרך הדק A0 (ה-A/D) כדי לקבל מידע על עוצמת הגז באוויר הבית.

9.9 סוללות $LiFePO_4$:

איור 19 - סוללות

בפרויקט אנו משתמשים בסוללות LiFePO4 נטענות של 3.2v כאשר כל כרטיס פועל בעזרת סוללות אלו, מה שמאפשר נוחות וניידות. היחידות של חיישני הגז והעשן פועלות על שני סוללות כאלו ואילו שאר יחידות החיישנים עם סוללה אחת.

תחום העבודה שלהם הוא בין 3.2v-3.3v, מתח המקסימלי אותם ניתן להטעין הוא 3.7v, והמתח המינימלי אילו הן יכולות להתפרק הוא 2.8v.

הסוללות האלו מקיימות פריקה קבועה, והן מבטיחות **בטיחות גבוה** יותר מאשר סוללות על בסיס Lithium אחרות.

יתרון נוסף של סוללות אלו היא יכולת טעינה מהירה יותר, ושיעורי פריקה גבוהים יותר. בדרך כלל סוללות אלו מאפשרות מספר גבוה יותר של מחזורי טעינה מה שמאפשר חיי סוללה גבוהים יותר כשהן לא טעונות סופית.

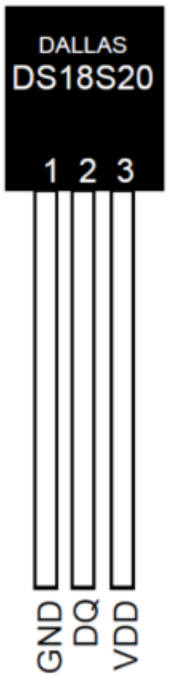
חיסרון של סוללות אלו הוא שצפיפות האנרגיה היא נמוכה יותר מאשר סוללות LI-Ion רגילות. תוחלת החיים של סוללות אלו היא כ 5-7 שנים.

9.10 ממיר מתח ממותג (Step Up) BOOST:

איור 20 - ממיר מתח ממותג

ממיר BOOST הוא ממיר המיועד להעלות מתח. ממיר זה אינו מבודד, כלומר אין הפרדה חשמלית ביו מתח הכניסה ומתח היציאה. לאחרונה פותחו לממיר זה מספר שיטות המאפשרות נצילות גבוהה מ-95%, על ידי שימוש במעגלי תהודה להורדת הפסדי מיתוג הדיודה, ומיתוג בזרם אפס או מתח אפס כך שאין הפסדי מיתוג. מתח המוצא בדרך כלל גבוהה ולכן אין בעיה של הפסדי הולכת הדיודה. שימוש נוסף של ממיר זה הוא כמתקן גורם הספק אקטיבי. אנו משתמשים ברכיב זה רק ביחידות חיישני תנועה לשם הגברת מתח הסוללה מי 3.3v, למתח של 5v לשם הפעלת הרכיב.

9.11 חיישן טמפרטורה DS18B20:

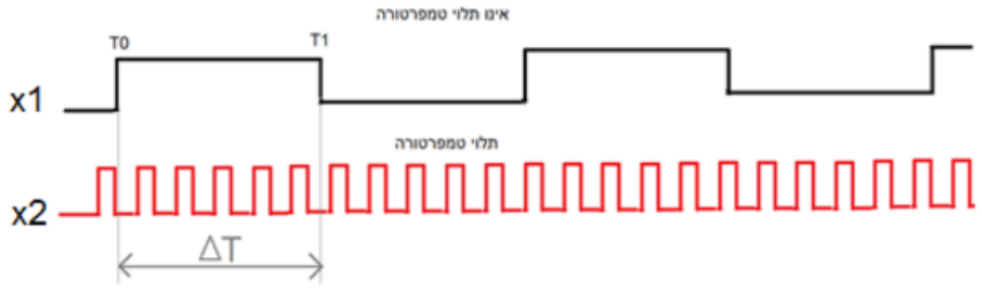


איור 21 - חיישן טמפרטורה

- רכיב זה מתקשר בפרוטוקול 1-WIRE, משמעו שהתקשורת בין רכיב זה למיקרובקר נעשת על ידי חוט אחד (הסבר על פרוטוקול 1-WIRE בפרק 10).
- רכיב זה אינו דורש שום רכיבים חיצוניים על מנת לתפקד.
- רכיב זה מסוגל למדוד טמפרטורה הנעה בין $-55°C$ ל- $+125°C$
- לרכיב זה ישנו רזולוציית מדידה של $\pm 0.5°C$
- קיימת אפשרות לחיישן זה לעבוד ללא מקור מתח חיצוני. המתח מסופק דרך 1-WIRE, נגד PULLUP דרך הדק זה כאשר הקו בגבוה. האות הגבוה גם טוען קבל פנימי (CPP), אשר מספק מתח כאשר החיישן בנמוך.

<u>אופן פעולת הרכיב:</u>

רכיב זה מודד את הטמפרטורה באמצעות שני מתנדי גביש פנימיים, כאשר תדר גביש אחד (X1) אינו מושפע מן שינויי טמפרטורה וגביש שני (X2) שתדרו מושפע מטמפרטורת הסביבה (ככל שהטמפרטורה גדלה כך גם התדר גדל) ,ותדרו גבוה משמעותית מתדר הגביש X1.



איור 22 - תיאור תדרי הגביש

הגביש X1 מייצר חלון זמן קבוע ΔT ,על ידי מדידת מספר הפולסים של הגביש X2 בחלון זמן זה ולאחר כיול מקבלים מדידת הטמפרטורה.

9.12 פרוטוקול תקשורת ONE-WIRE :

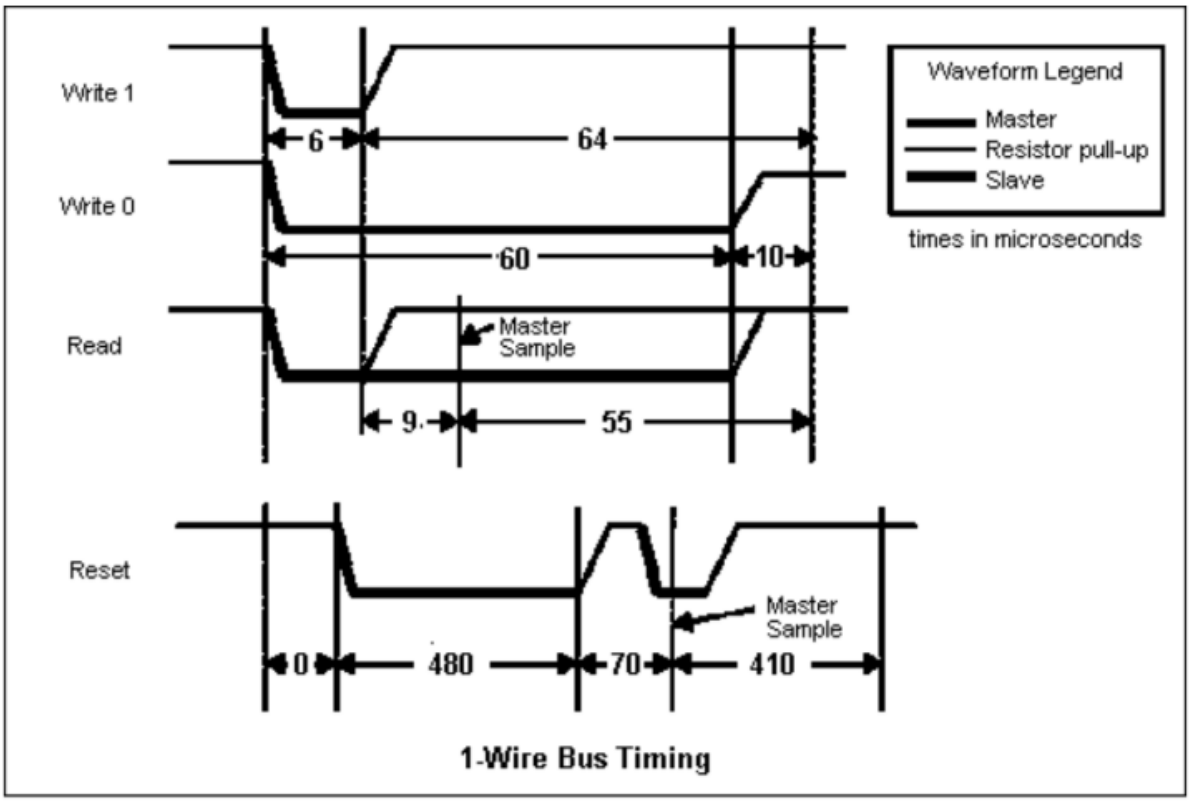
הקדמה:

זהו פרוטוקול תקשורת טורית הנועד להעברת מידע בין מיקרובקר לבין רכיבי זיכרון וחיישנים בקו תקשורת אחד מכאן שמו ONE WIRE.

קו התקשורת בינו ב-Open Drain משמעו שיש צורך בנגד Pull-Up שיתחבר בין קו המידע למתח הגבוה (VCC), משמעות הדבר היא שהרכיבים על הקו מסוגלים רק למשוך את הקו לנמוך, דבר זה נועד כדי למנוע העמסה על הקו כאשר שני רכיבים מנסים למשוך את הקו לגבוה.

אופן שידור בקו:

איור, ניתן לראות את אופן השידור '1','0' אתחול בקו וקריאת הקו



איור 23 - DS18B20 אופן שידור

פקודות הפעלה לרכיב DS18B20 בפרוטוקול ONE WIRE:

כל תשדורת חדשה בפרוטוקול זה מתחילה באתחול הקו (שליחת Reset).

Read ROM [33h]:

פקודה זו נותנת לרכיב ה-MASTER לקרוא את כתובת הרכיב המורכב מן 8 סיביות קודם משפחה, 48 סיביות של המספר הייחודי של הרכיב ו-8 סיביות קוד CRC.
פקודה זו מיועדת לשימוש כאשר רק רכיב אחד מחובר על הקו, אחרת כל הרכיבים המחוברים על הקו ינסו לענות לרכיב ה-MASTER וזה ייצור התנגשות בקו.

Match ROM [55h]:

פקודה זו כאשר אחריה מגיע כתובת הרכיב בעלת 48 סיביות מאפשרת לרכיב ה-MASTER לגשת לרכיב מסוים כאשר על הקו ישנם מספר רכיבים.

Skip ROM [CCh]:

פקודה זו יכולה לחסוך בזמן בכך ש-MASTER יכול לגשת לזיכרון הפנימי של הרכיב ללא נתינת כתובת בת 48 סיביות.
פקודה זו שימושית רק במערכת עם רכיב אחד כיוון שבמערכת בעלת כמה רכיבים יכולה להיווצר התנגשות על הקו כאשר כל הרכיבים מנסים בו זמנית לשדר את המידע.

Search ROM [F0h]:

כאשר המערכת מופעלת לראשונה רכיב ה-MASTER יכול לא לדעת כמה רכיבים נמצאים על הקו או מה כתובתם בת 48 סיביות.
פקודה זו נותנת לרכיב ה-MASTER להשתמש בתהליך האלימינציה כדי לזהות את כתובת בת 48 סיביות של כל הרכיבים המחוברים על הקו.

Alarm Search[ECh]

פקודה זו כמעט זהה לפקודת Search ROM אך השוני בה היא שהרכיב יענה לפקודה זו רק אם תנאי ההזעקה התמלא (האוגר בתוך זיכרון ה-EEPROM), לדוגמא אם בחיישן הטמפרטורה, הטמפרטורה המדודה גבוהה מן סף המקסימום או נמוכה מן סף המינימום.

Write Scratchpad [4Eh]:

פקודה זו מאפשרת ל-MASTER לכתוב לתוך זיכרון ה-Scratchpad של הרכיב ה-SLAVE.

Read Scratchpad [BEh]:

פקודה זו מאפשרת ל-MASTER לקרוא מן הזיכרון ה- Scratchpadשל רכיב ה-SLAVE.

Copy Scratchpad [BEh]:

פקודה זו גורמת לזיכרון ה-Scratchpad של ה-SLAVE להישמר תוך הזיכרון ה-EEPROM שלו.

Convert T [BEh]:

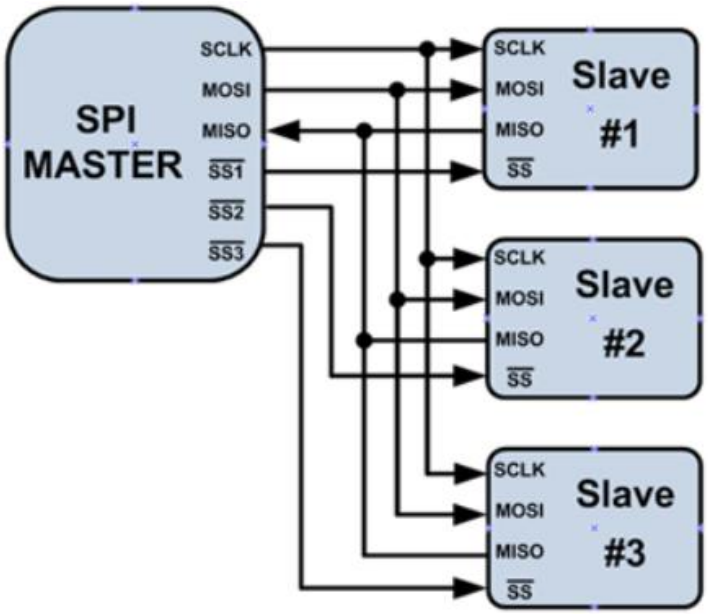
פקודה זו מתחילה את תהליך המרת הטמפרטורה, לאחר המרת הטמפרטורה הערך הנמדד נשמר בתוך אוגר והרכיב נכנס למצב IDLE.

9.13 פרוטוקול SPI:

תקן "SPI "Interface Peripheral Serial נועד לאפשר תקשורת טורית בין מכשיר MASTER אחד למספר מכשירי SLAVE. פרוטוקול זה לא מגדיר את המידע אלא את אופן, תזמון השידור, והקליטה שלו.

קצב השידור בפרוטוקול SPI הוא עד Mbps1 והמרחק המומלץ בין מכשירים הוא 3 מטר. התקן מבוסס על העברת מידע באמצעות ארבעה קווי תקשורת דו כיווניים:

- **SCLK**- קו שעון תזמון משותף לכל הרכיבים (נקבע על ידי ה-MASTER).
- **MOSI**- מידע שיוצא מ-MASTER (ארדואינו במקרה שלנו) ומגיע ל-SLAVE (המסך או ה-RFID במקרה שלנו).
- **MISO**- מידע שיוצא מ-SLAVE (מסך או ה-RFID) ומגיע ל-MASTER ארדואינו.
- **SS**- בחירת רכיב. כשיישנם כמה רכיבים מחוברים במקביל ישנו חיבור ייחודי לכל אחד שנקרא SLAVE SELECT (במקרה וה-MASTER מעוניין לתקשר עם רכיב מסוים, הוא שולח לאותו רכיב את הסיבית '0').



איור 24 - חיבור רכיבים בתקשורת SPI

תהליך התקשורת המלא מתחיל בכך שרכיב ה-MASTER מגדיר את קצב השעון ואת מצב הפעולה שלו.

קיימים ארבעה מצבי פעולה של השעון:

- **מצב 0 (CPOL= '0' ,CPHA = '0')** - במצב זה ערך הבסיס של השעון הוא '0' והמידע נקלט בעליית מתח. (Rising Edge: High to Low) ומועבר בירידת מתח (Falling Edge : Low to high).
- **מצב 1 (CPOL= '0' ,CPHA = '1')** - במצב זה ערך הבסיס של השעון הוא '0' והמידע נקלט בירידת מתח (Falling Edge: High to Low) ומועבר בעליית מתח (Rising Edge: Low to High).
- **מצב 2 (CPOL= '1' ,CPHA = '0')** - במצב זה ערך הבסיס של השעון הוא '1' והמידע נקלט בעליית מתח (Raising Edge: High to Low) ומועבר בירידת מתח (Falling Edge: Low to High).
- · **מצב 3 (CPOL= '1' ,CPHA = '1')** - במצב זה ערך הבסיס של השעון הוא '1' והמידע נקלט בירידת מתח (Falling Edge: High to Low) ומועבר בעליית מתח (Rising Edge: Low to High).

לאחר שה-MASTER קבע את מצבי הפעולה של השעון, ה-MASTER מגדיר לאיזה SLAVE הוא עומד לבצע תקשורת באמצעות שליחת '0' בקו SS של ה-SLAVE הרצוי. לאחר מכן, הוא שולח ביט התחלה בקו ה-MOSI שאותו ה-SLAVE קורא. במקביל ה-SLAVE שולח ביט אישור בקו ה-MISO שאותו ה-MASTER קורא, ולאחר מכן כאשר ה-MASTER מסיים את שליחת המידע, הוא מפסיק את פעולת השעון.

פעולה זו גורמת לביטול הבחירה של ה-SLAVE.

רק ה-SLAVE שנבחר באמצעות קו ה-CS מתייחס לשעון ה-CLK ולקו ה-MOSI, ורק הוא יכול לכתוב מידע בקו ה-MISO. שאר רכיבי ה-SLAVE מתעלמים מהשעון ומה-MOSI ואינם רשאים לכתוב לקו ה-MISO.

9.14 פרוטוקול I2C:

לארדואינו יש את היכולת לתקשר עם התקנים חיצוניים ומודולים באמצעות פרוטוקולים סטנדרטיים שהופכים את התקשורת לפשוטה ואמינה ויותר, בפרויקט שלנו השתמשנו בפרוטוקול זה לתקשורת בין הארדואינו DUE לבין זיכרון ה-EEPROM והשעון RTC. הארכיטקטורה של I2C היא של MASTER ו-SLAVE. כשהארדואינו הוא ה MASTER אשר מתחיל תקשורת עם רכיבים שנקראים SLAVE.

מאפיינים של פרוטוקול זה:

- פרוטוקול זה דורש רק 2 חוטים לתקשורת.
- פרוטוקול זה יכול להיות יותר מסובך להפעלה מבחינת תכנות.
- פרוטוקול זה דורש שני נגדיי PULL UP על חוטי העברת המידע.

כפי שציינו במאפיינים התקשורת בפרוטוקול זה נעשית באמצעות שתי חוטים:

SDA - חוט זה אחראי על העברת נתונים בין הארדואינו לרכיבים אחרים. בארדואינו DUE זהו פין 20.

SCL - חוט זה הוא קו השעון האחראי לתזמון של העברת הנתונים. בארדואינו DUE זהו פין 21.

כשהארדואינו מתחיל שליחת נתונים כל הרכיבים המחוברים אליו בפרוטוקול זה מקשיבים לנתונים ולכל אחד מהם מספר זיהוי ייחודי (כתובת) שמבדיל אותו מהשאר. אם התקשורת מופנית אליו הוא מגיב לתקשורת לפי הקוד ובדרך כלל שולח נתונים חזרה לארדואינו. זהו סדר הדברים:

- הארדואינו שולח ביט להתחלת תקשורת.
- הארדואינו שולח כתובת של 7 ביט שמופנית לאחד הרכיבים.
- הארדואינו שולח סיבית קריאה '0' או סיבית כתיבה '1' כדי להגדיר את הפעולה שברצוננו לעשות.
- הארדואינו כותב או קורא מהרכיב על ידי שליחת בית אחד אחרי השני, הרכיב מודיע שקיבל את הבית אחרי כל שליחה.
- הארדואינו שולח ביט לסגירת התקשורת.

<u>נסביר לעומק:</u>

העברה יכולה להתחיל רק כאשר הקו לא עסוק NOT BUSY.

בזמן העברת הנתון, 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.

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, ועבור כל ביט של נתון יש פולס שעון אחד. האינפורמציה מועברת בית אחרי בית וכל רכיב קולט מאשר קבלת הבית עם ביט של ACKNOWLEDGE .

סיבית ACKNOWLEDGE:

כל רכיב קולט חייב בסיום קליטת בית, שהועבר אליו, ליצור ביט ACKNOLEDGE .

רכיב ה-MASTER 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.

סיבית NOT ACNOWLEDGE:

סיבית זאת נשלחת במקרים הבאים:

- המקלט אינו מסוגל לקבל או לשדר כי הוא מבצע כמה פונקציות בזמן אמת והוא לא מוכן להתחיל לתקשר עם המאסטר.
- במהלך ההעברה, רכיב הקולט מקבל נתונים או פקודות שהוא אינו מבין.
- במהלך העברה , המקלט אינו יכול לקבל עוד נתונים.
- רכיב ה-MASTER סיים לקרוא את הנתונים, ומסמן זאת לרכיב ה-SLAVE.

ה-MASTER משדר וה-SLAVE קולט -אופן כתיבה- Write Mode :

תחילה רכיב ה-MASTER שולח סיבית התחלה START ומיד אחריו 7 סיביות שהינם הכתובת של ה-SLAVE וסיבית של 0 שבה ה-MASTER קובע שהוא ממשיך להכתיב ל-SLAVE.

למידע זה ה-SLAVE מחזיר סיבית ACNOWLEDGE במידה וקיבל את המידע כנדרש.

לאחר מכן ה-MASTER שולח את הכתובת שבה רוצים לשלוח את המידע ב-SLAVE. אם הכתובת גדולה מ-255 היא תישלח בשני חלקים של 8 סיביות, לאחר כל חלק ה-SLAVE יחזיר סיבית ACKNOWLEDGE.

לאחר מכן ה-MASTER שולח 8 סיביות שהם המידע ל-SLAVE, לאחר שליחת המידע ה-SLAVE יחזיר סיבית ACNOWLEDGE ומיד לאחר סיבית זו ה-MASTER יעצור את השידור עם סיבית STOP .

ה-SLAVE משדר וה-MASTER קולט -אופן קריאה- Read Mode :

תחילה רכיב ה-MASTER שולח סיבית התחלה START ומיד אחריו 7 סיביות שהינם הכתובת של ה-SLAVE וסיבית של 0 נוספת שבה ה-MASTER קובע שהוא ממשיך להכתיב ל-SLAVE.

למידע זה ה-SLAVE מחזיר סיבית ACNOWLEDGE במידה וקיבל את המידע כנדרש.

לאחר מכן ה-MASTER שולח את הכתובת שממנה רוצים לקבל את המידע מרכיב ה-SLAVE. אם הכתובת גדולה מ-255 היא תישלח בשני חלקים של 8 סיביות, לאחר כל חלק ה-SLAVE יחזיר סיבית ACKNOWLEDGE. לאחר מכן ה-MASTER מתחיל שיחה מחדש ושולח סיבית START ומיד אחריו 7 סיביות שהינם הכתובת של ה-SLAVE וסיבית נוספת של 1 שבה ה-MASTER קובע שהוא מחכה למידע מה-SLAVE.

ה-SLAVE שולח סיבית ACKNOWLEDGE ולאחריה את המידע שבגודל 8 סיביות.

לבסוף ה-MASTER יישלח סיבית NOT ACKNOWLEDGE לשחרר את הקו ואז יישלח סיבית STOP לעצור את השיחה.

9.15 פרוטוקול UART:

פרוטוקול universal asynchronous receiver-transmitter "UART" הוא פרוטוקול תקשורת טורית א-סינכרונית. בשיטה זו זמני תחילת השידור של הסיביות המרכיבות כל תו אינם קבועים מראש ואינם ניתנים לצפייה ע"י המקלט.
התווים משודרים בזה אחר זה באופן בלתי תלוי כאשר מרווחי הזמן ביניהם אינם שווים בהכרח. אי לכך יש צורך להוסיף אל המידע עצמו תווי (סיביות) בקרה שונים.
אנו משמשים בפרוטוקול זה לשם תקשורת בין ARDUINO DUE
לבין בקר ה-ESP 8266.

הסבר תווי/סיביות הבקרה בתקשורת טורית אסינכרונית:

סיבית התחלה- Start Bit:

סיבית המסמנת את תחילת השידור, ונותנת למקלט התראה להתכונן לקליטת מידע (רמה '0').

תו המידע- Data:

רצף של מספר סיביות(7 או 8) שהן בעצם המידע עצמו ששודר (לדוגמא תו ASCII) כאשר הסיבית הLSB משודרת ראשונה וסיבית הMSB אחרונה.

סיבית הזוגיות- Parity Bit:

סיבית המשמשת לבקרת שגיאות. מציינת האם מס' הפעמים שבו מופיעה הסיבית '1' בתו מסוים (כולל סיבית הזוגיות) יהיה זוגי או אי זוגי- בהתאם למה שנקבע מראש.

סיבית העצירה- Stop Bit:

מסמנת את סוף התו. סיבית זו הינה דופק ברמה לוגית '1', שמשכו שווה לזמן שידור סיבית אחת או אחת וחצי או שתיים, בהתאם לקצב השידור ולשיטת הקידוד.

המצב היציב- Steady State:

רמה קבועה של '1' מתארת מצב ללא שידור בקו, רמה '1' לוגית מאפשרת בין היתר בדיקת תקינות הקו: במצב בו הקו אינו תקין הרמה הלוגית הנקלט הינה '0'.

לדוגמא:

אנו מעוניינים לשלוח את המידע h56, בתקשורת UART:

איור 25 - שידור 0x56 בתקשורת טורית

המשדר ומבנה התשדורת הטורית של UART:

TX- רגל לשידור טורי, המידע נשלח מהתקן אחד להתקן שני ולהפך.

RX-רגל קליטה טורית, המידע נקלט להתקן אחד מההתקן השני ולהפך.

ההדקים הללו מחוברים בהצלבה, לכן המידע המשודר מהתקן 1 דרך הדק TX נקלט בהתקן 2 דרך הדק RX, ולהפך.

דרישות/מגבלות נוספות של תקשורת האסינכרונית:

- לא חייב להצמיד את ההודעה אחת לחברתה,

אך מן ההכרח שבמסגרת הודעה אחת תהיה הצמדה מלאה של כל הסיביות-סיביות הנתונים וסיביות הבקרה (כלומר-שהסיביות לא יהיו מופרדות אחת מן השנייה כולן ברצף).

- אין כל בזבוז זמן בהעברת הנתונים, פרט לזמן הדרוש להעברת סיביות ההתחלה וסיום עבור כל הודעה.
- מגבלה של פרוטוקול זה היא שהמקלט אינו יודע מתי המשדר יתחיל לשדר ובמערכות תקשורת מתקדמות יותר הדבר מהווה בעיה.

9.16 פרוטוקול MQTT:

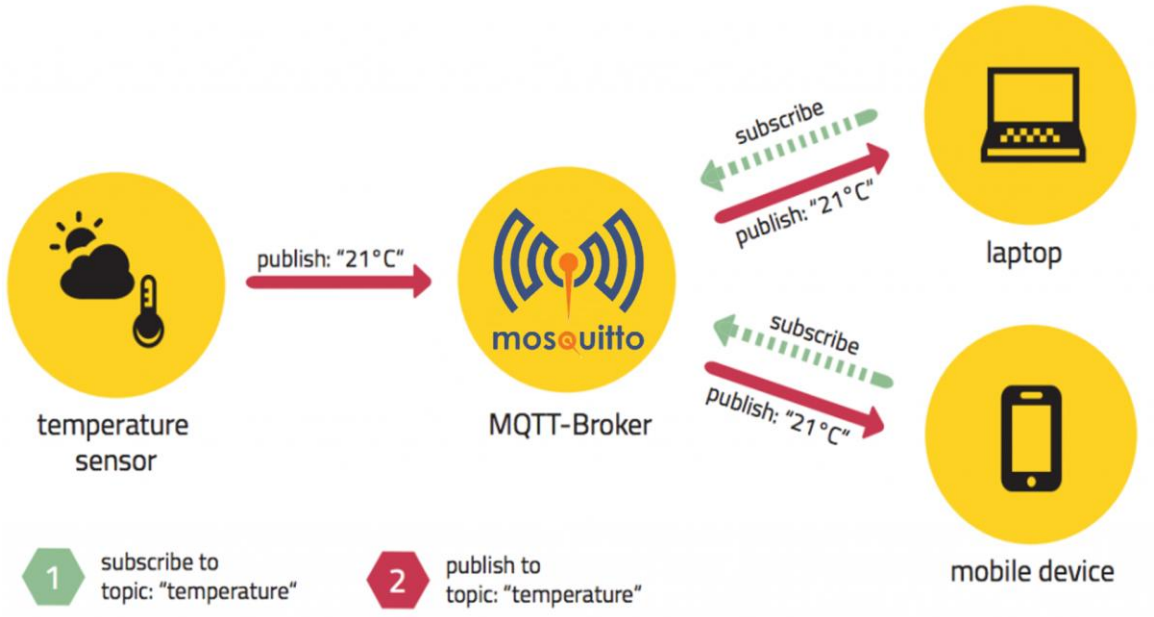
MQTT הוא פרוטוקול להעברת מידע שעובד על עקרונות שרת/לקוח (Client/Server) ומפרסם/מקבל (Publish/Subscribe). הפרוטוקול הוא בעל שרת פתוח, קל לשימוש ומעוצב כדי שיהיה נוח לשימוש.

הפרמטרים האלו הופכים את הפרוטוקול לשימוש בהמון מקרים לנוח וקל, כולל סביבות מוגבלות כגון תקשורת ממכונה למכונה (M2M).

ה-MQTT נוצר על ידי (IBM) Andy Stanford-Clark ו-(Arcom) Arlen Nipper, Now Cirrus Link

בשנת 1999, למטרת יצירת פרוטוקול בשביל מינימום הפסדי סוללה ומינימום רוחב פס. הם ציינו את המטרות הבאות לפרוטוקול העתידי שאמור להיות: פשוט ליישום, לספק משלוח נתונים איכותי, משקל ורוחב פס יעילים ומודעות רציפה. המטרות האלו הם עדיין הליבה של ה-MQTT.

MQTT הינו פרוטוקול העברת מידע מהתקן אחד להתקן המאפשר שיחה בין מספר התקנים בו בזמן. בפרוטוקול זה מבוסס על מנגנון Publish & Subscribe, ה-Publish הינו ההתקן ששולח מידע ו-Subscribe הינו ההתקן שקולט את המידע (יש לציין שהתקן יכול להיות גם Publish וגם Subscribe). פרוטוקול זה עובד בעזרת Broker (מתווך) אליו משודרים כל הנתונים, רק כאשר יש שינוי באחד הנתונים במתווך אז יישלח להתקן ה-Subscribe המידע החדש ששונה.



איור 26 - מבנה עקרוני של תקשורת

מאפיינים של פרוטוקול זה:

- מתאים לתקשורת בין הרבה נקודות שמתקשרות אחת עם השנייה דרך מתווך (Broker) במיוחד יחס של אחד לרבים
- מהיר וקל משקל, פחות בייטים נשלחים
- המכשירים לא יודעים אחד על השני
- הלקוח מתחיל תקשורת והיא נשארת פתוחה בצורה רציפה
- אידיאלי לאינטרנט של הדברים ולתקשורת בין מכונות במיוחד כשיש הרבה מהם.

ה-Broker הוא המוח של המערכת, הוא זה ששולח את המידע לכל המכשירים לפי המנוי שעשו. ה-Broker יכול לעבד מידע במקביל ולנתב את המידע בצורה חכמה אך זו תהיה בעיה עבור המון חיבורים. אפשר לפתור בעיה זו בעזרת חיבור צמתי Broker מקובצים המאפשרים להפיץ את העומס

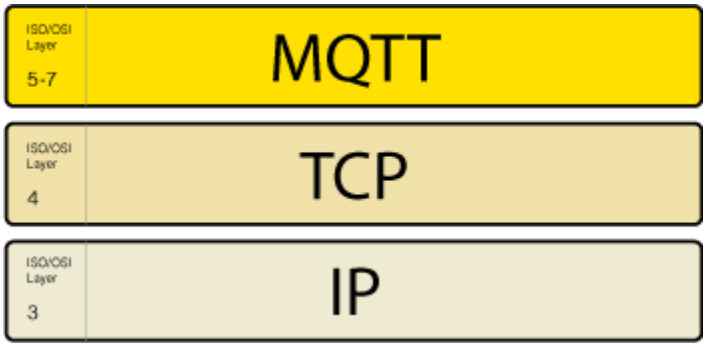
לשרתים ספציפיים אשר מתמקדים באיזון העומס.

ה-Broker יכול למיין מידע על פי תחומים של נושא או עניין, תוכן או סוג לכן המשתמש יכול לקבל מידע רלוונטי על פי תחומים אלו.

ה-Client (לקוח) יכול להיות Publish & Subscribe או שניהם באותו הזמן. ה-Client יכול להיות ממיקרו בקר עד לשרת הכוללים ספריות MQTT, אשר מתחברים אל ה-Broker דרך אמצעי תקשורת כלשהוא. הספריות של ה-Client זמינות למגוון גדול של שפות תיכנות כמו אנדרואיד, ארדואינו, C, ועוד...

ה-Broker (מתווך) הוא הליבה של כל Publish & Subscribe פרוטוקול. ה-Broker יכול להחזיק אלפים של חיבורי לקוח בו זמנית. ה-Broker אחראי בעיקר על קבלת כל ההודעות, ולהחליט מי מעוניין בקבלת המידע ולנתב את המידע ללקוח המתאים או לסנן אותם.

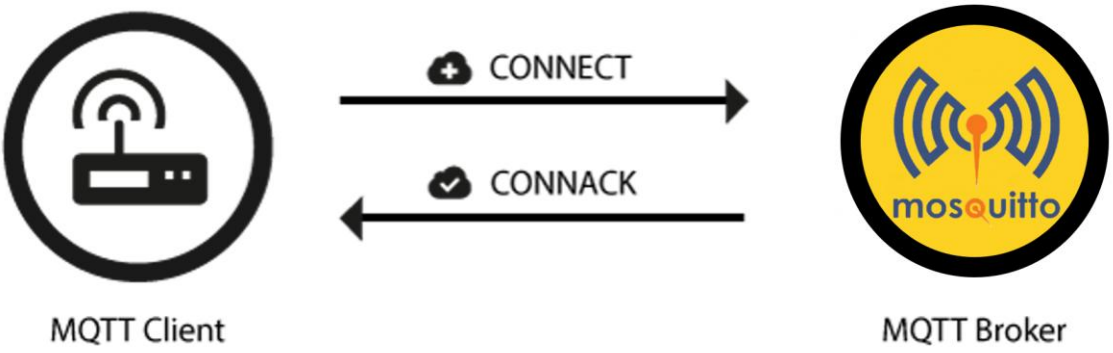
פרוטוקול MQTT מבוסס על TCP/IP וגם ה-Broker וה-Client צריכים להיות מבוססים על TCP/IP



איור 27 - מודל 4 השכבות

החיבור תמיד מתבצע בין Client לבין ה-Broker אין אפשרות ש-Client יתחבר אל Client אחר ישירות. החיבור מבוסס דרך שליחת הודעת CONNECT ל-Broker, וה-Broker שולח בחזרה ל-Client הודעת CONNACK וקוד מצב. לאחר יצירת החיבור ה-

Broker ישמור אותו פתוח כל עוד הלקוח אינו שולח פקודת ניתוק או שהוא מאבד את הקשר.



איור 28 - תיאור חיבור אל ה-Broker

הודעת ה-CONNECT כוללת בתוכה מאפיינים שתואמים ל-Broker, כאשר תישלח הודעה זו ה-Broker יבדוק את התאימות, אם הודעה זו נכונה הוא יתחבר אל ה-Client אבל אם ההודעה אינה מתאימה או שלא נשלחה במסגרת הזמן אז ה-Broker יסגור את הקשר עם ה-Client.

בהודעה זו יש בקשה של Client Id (קיצור של Client Identifier) שבו ה-Broker מזהה על פי הנתונים של כל Client אל איזה Client ספציפי לגשת. בנוסף בהודעה צריכים להישלח שם משתמש וסיסמא שהוגדרו מראש ב-Broker. בשביל להתחבר ל-Broker ה-Client צריך לשלוח בהודעה זו סיסמא ושם משתמש מסויימים, רק עם אותה סיסמא ושם משתמש נוכל לגשת אל ה-Broker אחרת ה-Broker לא יתחבר אל ה-Client. וכך הלאה לפי איך שהוגדר ה-Broker.

לאחר הניסיון חיבור אל ה-Broker, ה-Broker ישלח הודעת CONNACK ובה יהיה רשום תוצאות ניסוי ההתחברות, אם החיבור התבצע, נכשל או אם החיבור התנתק.

לאחר החיבור אל ה-Broker נרצה לשלוח אליו מידע, לעשות אליו Publish. בהודעת ה-Publish יהיה חייב להיות הנושא של ההודעה ובנוסף המידע שנרצה לשלוח בפורמט של בית (Byte).

ה-Client יקבל את הודעת ה-Subscribe כאשר התבצע שינוי, למשל עם ערך של טמפרטורה:

- אם נמדד ערך של 15 מעלות ולאחר מכן שוב 15 מעלות לא תישלח הודעה שוב אלא אם כן הטמפרטורה השתנתה מ15 מעלות. במידה ואין שום Client שקשור לנושא של אותו הערך אין שום צורך ב-Subscribe של הערך הנ"ל. הודעת ה-Subscribe תבוא מה-Broker עם קוד (UNSUBACK) מסויים שדואג שהמידע יישלח רק ל-Client שקשור לנושא של הערך שהשתנה.

הנושאים ב-MQTT הם מחרוזות מסוג UTF-8 (ר"ת של Unicode Transformation Format 8) אשר משומשות על ידי ה-Broker, באמצעותם ה-Broker מנתב את המידע לכל Client מחובר. הנושא מורכב מאחד או יותר רמות נושא (Topic Level). כל רמת נושא מופרדת בעזרת הסימן / (קו נטוי קידמי) . עדיף להימנע מלהשתמש ב / בסוף השם של הנושא האחרון מכיוון שזה מציג רמת נושא מיותרת ואין לזה יתרונות, לעתים קרובות זה רק מוביל לבלבול. בנוסף מומלץ לרשום את הנושאים קצרים ותמציתיים כי כאשר זה מוביל למכשירים קטנים כל ספירת בית מהווה הבדל.

איור 29 - תיאור הנושאים

בגלל שמשתמשים במחרוזות מסוג UTF-8 יש הבדל אם נרשום myhome/groundfloor לבין Myhome/groundfloor כי הם ייחשבו בתור 2 נושאים

שונים. כל נושא חייב להכיל לפחות אות אחת כדי שיאושר. מומלץ להימנע להשתמש ברווח בגלל שזה די ברור שיהיה קשה לנפות שגיאות של הנושאים, וזהו תו שאינו מוכר מפני שב UTF-8 מכיר המון סוגים של רווח.

ישנם תווים כללים שאם הם מופיעים ב-Client בנושא אז יישלחו אליהם מספר תווים או יותר. יש 2 רמות של תווים כלליים: רמה אחת ורמה מרובה. רמה אחת מסומנת ע"י הסימן + ורמה מרובה ב#. נסביר את הרמות באמצעות מספר דוגמאות-

ניקח דוגמה עבור רמה אחת של נושא myhome/groundfloor/+/temperature

כל הנושאים שבהתחלה שלהם יש את myhome/groundfloor ובסוף temperature/ זאת אומרת לא משנה מה יהיה רשום בתת נושא במקום ה+ יישלחו ל-Client. לדוגמה אם קיימים הנושאים הבאים הם יישלחו ל-Client:

myhome/groundfloor/center/temperature

myhome/groundfloor/kitchen/temperature

myhome/groundfloor/parentsroom/temperature

ניקח דוגמה עבור רמה מרובה של נושא myhome/groundfloor /#

כל הנושאים שלהם יש את אותה ההתחלה לא משנה מה הסוף שלהם יישלחו

ל-Client. זאת אומרת שגם עבור המקרה הזה הדוגמאות שנרשמו מקודם גם יישלחו כי להם יש את אותה התחלה של נושא. אם יש המון נושאים עדיף לא לעשות Subscribe לכולם כי אחרת ההודעה תהיה מעל המותר ולא יוצגו כמו שצריך.

בנוסף יש נושאים מיוחדים שמטופלים באמצעות ה-Broker שנשמרים בתוך האחסון הסטטיסטי הפנימי של ה-Broker. נושאים אלו מתחילים עם הסימן $ אך לרוב מתחילים ב-$SYS. לנושאים אלו לא ניתן לעשות Publish מה-Client, רק ה-Broker אחראי עליו.

שם של נושא לדוגמה: $SYS/broker/messages/sent

ל-MQTT יש 3 רמות של איכות שירות Qulity of Service (QoS). זו היא הסכמה בין הנשלח לנקלט של ההודעה לגבי האחריות של שליחת ההודעה. אלה הם שלושת הרמות שירות:

At Most Once מסומן גם כ 0, At Least Once מסומן גם כ-1 ו-Exactly Once שמסומן גם כ-2.

יש שני חלקים של שליחת הודעה: ה-Publish מה-Client אל ה-Broker והשני הוא ה-Subscribe מה-Broker אל ה-Client. צריכים להסתכל על 2 מקרים אלו בנפרד כי יש הבדלים מתוחכמים ביניהם. איכות השירות עבור Publish מה-Client אל ה-Broker תלוי ברמת האיכות של ה-Client שנקבעה בשביל ההודעה. עבור מקרה שה-Broker שולח הודעת Subscribe עבור אחד ה-Client איכות השידור נקבעה מבעוד מועד.

איכות השירות זו היא תכונה מרכזית של ה-MQTT, זה מאפשר שיחה יעילה וקלה בין רשתות לא ברורות או לא אמינות, מכיוון שהפרוטוקול מטפל ומבטיח אחריות לשידור הודעה, ללא קשר לכמה השרת הוא לא אמין או לא ברור. כמו כן, הוא מעצים את ה-Client לבחור את רמת איכות השירות בהתאם לאמינות הרשת וללוגיקת היישום.

איכות שירות 0 (QoS 0)

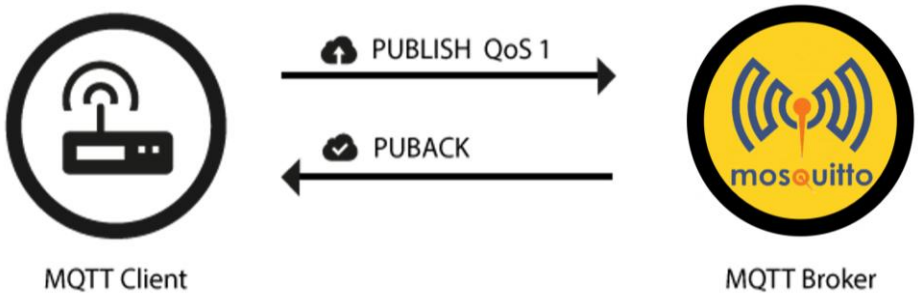
הרמה המינימלית היא רמת 0, היא מבטיחה את השליחה הכי טובה. ההודעה לא תוחזק על ידי המקלט או תהיה מאוחסנת או תשלח מחדש על ידי השולח. לרוב לרמה זאת קוראים בשם "תירה ותשכח" ומספקת את אותם הבטחות כמו הפרוטוקול TCP הבסיסי.

איור 30 - תיאור איכות שירות 0

איכות שירות 1 (QoS 1)

רמה איכות שירות זו מבטיחה תישלח לפחות פעם אחת למקלט אך יכול להיות שההודעה יכולה להישלח יותר מפעם אחת. השולח יאחסן את ההודעה עד שהוא יקבל אישור בפקודת PUBACK מהשולח.

ה-PUBACK זוהי חבילה המגיבה לחבילת ה-Publish עם QoS 1.



איור 31 - איכות שירות 1

איכות שירות 2 (QoS 2)

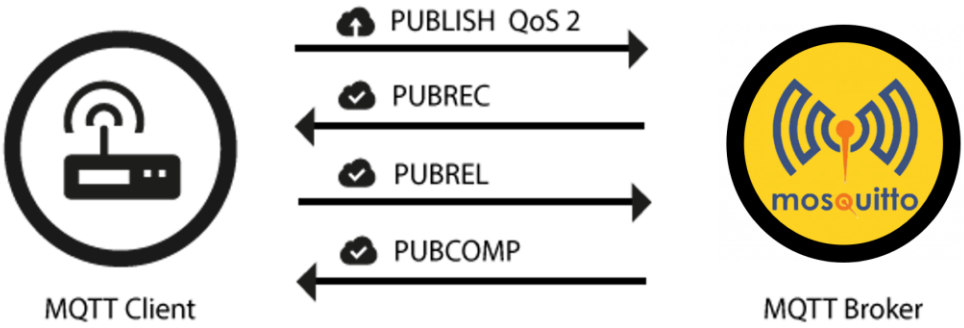
זו היא הרמה הגבוהה ביותר, המבטיחה שכל הודעה תקלט פעם אחת. זו היא רמת איכות השידור האטית ביותר והבטוחה ביותר. האחריות מובטחת בין שני הצדדים בין השולח למקלט.

במקרה זה השולח ישלח PUBLISH QoS 2 (זו היא החבילה הראשונה) המקלט יעבד את המידע ובהתאם לכך המקלט ישלח הודעת PUBREC (החבילה השנייה) שזו היא חבילת תגובה עבור הודעת PUBLISH.

המשדר ישמור את הודעת התייחסות PUBREC בחבילות הזיהוי עד שיישלח הודעת PUBCOMP, זה חשוב כדי למנוע מלעבד את ההודעה פעם שנייה. כאשר השולח מקבל את הודעת ה-PUBREC הוא יכול בבטחה לזרוק את הודעת ה-Publish כי הוא יודע אם המקלט קיבל בהצלחה את ההודעה.

המשדר יאחסן את הודעת ה-PUBREC ויגיב עם הודעת PUBREL (זו היא החבילה השלישית). אחרי שהמקלט יקבל את הודעת ה-PUBREL הוא יכול למחוק כל מצב מאוחסן ולהגיב עם PUBCOMP (זו היא החבילה הרביעית והאחרונה) הדבר גם נכון אם השולח יקבל את הודעת ה-PUBCOMP. לאחר שכל השיחה הושלמה שני הצדדים יכולים להיות בטוחים שההודעה נשלחה והשולח יודע על כך. בכל פעם שהמשלוח הולך לאיבוד בדרך השולח אחראי על שליחת ההודעה האחרונה לאחר פרק זמן סביר. זה נכון כאשר השולח הוא ה-Client וגם כש-Broker שולח הודעה.

על הצד המקבל מוטלת האחריות להגיב על כל הודעת פקודה בהתאם למה שקיבלה.



איור 32 - איכות שירות 2

מתי משתמשים ברמת שירות 0

1. כאשר קיים חיבור יציב או כמעט יציב בין השולח לקולט

2.כאשר לא אכפת אם אחד או יותר הודעות אבדו לפחות פעם אחת.

3.כאשר אין הודעות הנמצאות בתור לשליחה או לקבלה.

מתי משתמשים ברמת שירות 1

1. כאשר חייבים לקבל את כל ההודעות, וכאשר מטרת השימוש תוכל להתמודד עם כפילויות.

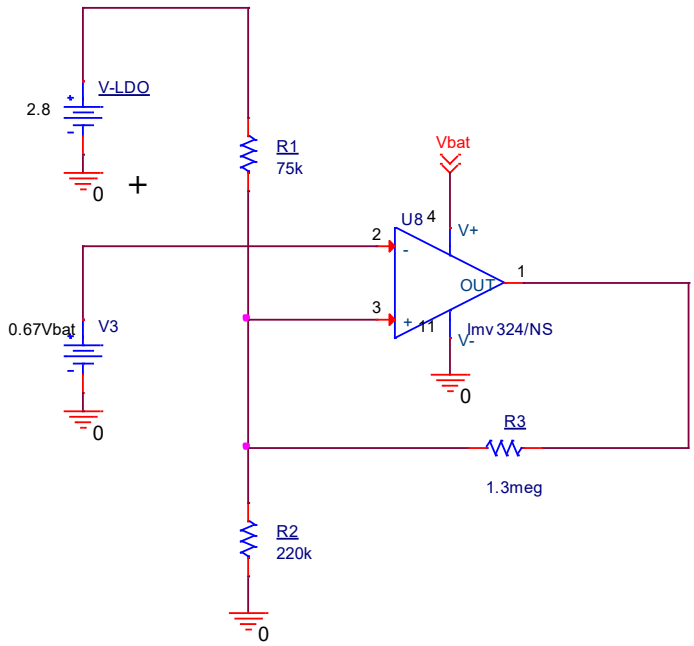
2.כאשר לא ניתן לשאת מעל רמת שירות 2. כמובן שרמת שירות 1 היא הרבה יותר מהירה בשליחת ההודעות ללא האחריות של רמה 2.

מתי משתמשים ברמת שירות 2

כאשר זה ממש חשוב לקבל את כל ההודעות פעם אחת, מכיוון שלעתים קרובות הכפילויות יכולות לגרום לפגיעה ביישום או ל-Clients אשר ב-Subscribe.

פרק חישובים:

חישוב רוחב החשל של השמיט במעגלי היחידות של החיישנים הספרתיים :

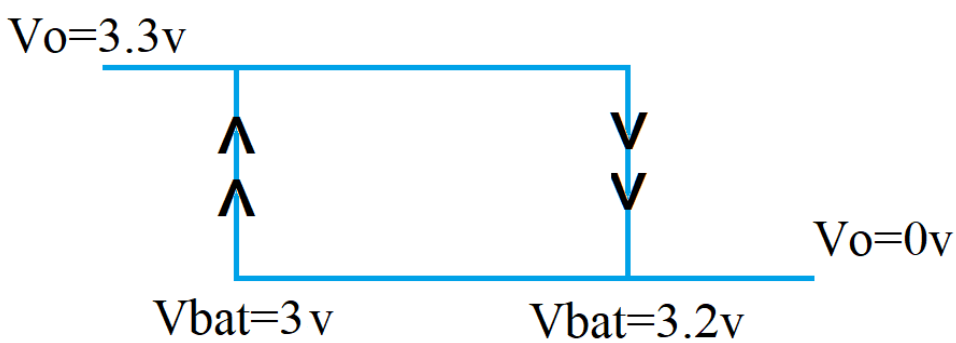


$$\frac{V_{(+)}-V_{LDO}}{R_1}+\frac{V_{(+)}-0}{R_2}+\frac{V_{(+)}-V_O}{R_3}=0$$
$$\frac{V_{(+)}-2.8}{75\times10^3}+\frac{V_{(+)}}{220\times10^3}+\frac{V_{(+)}-V_O}{1.3\times10^6}=0$$
$$V_{(+)}=2+V_O\times41.25\times10^{-3}$$

איור 33 - שרטוט השמיט ממעגל המיתוג האלקטרוני שבאיור 3.6

$$V_{HL}-?\rightarrow V_O=Vcc=3.3v$$
$$V_{HL}=2+V_O\times41.25\times10^{-3}$$
$$V_{HL}=2.136v$$
$$V_{bat}=\frac{2.136\times3}{2}=3.2v$$

$$V_{LH}-?\rightarrow V_O=GND=0v$$
$$V_{LH}=2+V_O\times41.25\times10^{-3}$$
$$V_{LH}=2v$$
$$V_{bat}=\frac{2\times3}{2}=3v$$



איור 34 - רוחב החשל

חישוב אחוזי החיישני גז בישול ועשן:

מתח המוצא המקסימלי של חיישנים אלו הוא 3.8v והמינימאלי הוא 0v

החיסרון של ESP8266 הוא שהדק A0 יכול לקלוט עד 1.1v לכן למוצא החיישן חיברנו מחלק מתח שיקטין את האות ביחס קבוע.

מתח של 3.8v הוקטן ל-1v, אם יתקבל מתח זה במבוא ה-ESP8266 אז הוא יקבל את הערך 930

$$R = \frac{1.1}{1024}$$
$$D = \frac{V}{R} = \frac{1 \times 1024}{1.1} = 930$$

אם קיבלנו ערך זה עבור מתח מקסימלי שבמוצא החיישן אז 930 שווה ערך ל-100%

לכן:

$$\text{הערך באחוזים} = \frac{\text{הערך שהתקבל}}{930} \times 100$$

חישוב אחוזי הסוללה:

מתח הסוללה המקסימאלי מגיע עד 3.7v ה-ESP8266 פועל החל ממתח של 3v לכן ניקח את 3v בתור מתח מינימלי

החיסרון של ESP8266 הוא שהדק A0 יכול לקלוט עד 1.1v לכן למוצא החיישן חיברנו מחלק מתח שיקטין את האות ביחס קבוע.

מתח של 3.7v הוקטן ל1v

אם יתקבל מתח זה במבוא ה-ESP8266 אז הוא יקבל את הערך 930

$$R = \frac{1.1}{1024}$$
$$D = \frac{V}{R} = \frac{1 \times 1024}{1.1} = 930$$

מתח של 3v הוקטן ל0.81v

אם יתקבל מתח זה במבוא ה-ESP8266 אז הוא יקבל את הערך 754

$$R = \frac{1.1}{1024}$$
$$D = \frac{V}{R} = \frac{0.81 \times 1024}{1.1} = 754$$

אם 930 מסמל 100% והערך 754 מסמל 0% אז

$$\text{הערך באחוזים} = \frac{\text{הערך שהתקבל} - 754}{930 - 754} \times 100 = \frac{\text{הערך שהתקבל} - 754}{176}$$

חישוב ערך הנגד שמחובר אל הדיודת זנר ביחידות חיישני הגז בישול והעשן:

בחיישנים אלו יש גוף חימום פנימי שהערך התנגדות החימום הוא 33Ω (על פי דף יצרן) בהתחשב שמחוברים במקביל אל הזנר גם שאר המעגל נגיד וערך ההתנגדות במקביל לזנר הוא 28Ω. הזרם על עומס זה הוא:

$$I = \frac{V_Z}{R_L} = \frac{5.1}{28} = 182.14mA$$

הזנר צריכה בנוסף זרם של 49mA (על פי דף יצרן) לכן הזרם על הנגד הוא סכום שני הזרמים.

אל המעגל צריכים להיות מחוברים לפחות 2 סוללות של $LiFePO_4$ שהמתח העבודה של הסוללה נע בין 3.2v-3.3v .

לכן ערך ההתנגדות של הנגד צריכה להיות:

$$R = \frac{2 \times V_{Bat} - Vz}{I_R} = \frac{6.6 - 5.1}{231 \times 10^{-3}} = 6.5\Omega$$

אנו לקחנו נגד בקירוב בערך של 5Ω.