|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| \\dsp-disk.eng.tau.ac.il\Documents\eeproj\Forms & Important\פרויקטים\הנחיות\Guidelines\Students\ENG_LOGO-01.png | | | **\\dsp-disk.eng.tau.ac.il\Documents\eeproj\Forms & Important\פרויקטים\הנחיות\Guidelines\Students\TAU_EngineeringENG.png** | |
| Sensing and telemetry systems | | | |
| פרויקט מס' 22-1-1-2666  דו"ח סיכום | | | |
| מבצעים: | | | |
|  | יונתן אמיר | 207021858 | |
|  | יורי לוקץ' | 311809867 | |
| מנחים: | | | |
|  | מר שמחה לייבוביץ | אוניברסיטת ת"א | |
|  |
| מקום ביצוע הפרויקט:  מעבדת בקרה | | | |

תוכן עניינים

[תקציר 4](#_Toc162348590)

[1 הקדמה 5](#_Toc162348591)

[2 רקע תיאורטי 5](#_Toc162348592)

[2.1.1 M5STACK 5](#_Toc162348593)

[2.1.2 PaHub 5](#_Toc162348594)

[2.1.3 PbHub 6](#_Toc162348595)

[2.1.4 מד תאוצה וג'ירו 6](#_Toc162348596)

[2.1.5 מד מרחק 6](#_Toc162348597)

[2.1.6 מצלמה תרמית 6](#_Toc162348598)

[2.1.7 מד לחץ 6](#_Toc162348599)

[2.1.8 LCD 6](#_Toc162348600)

[2.1.9 Vibration Motor 6](#_Toc162348601)

[2.1.10 RGB 7](#_Toc162348602)

[3 מימוש 7](#_Toc162348603)

[3.1 תיאור חמרה 8](#_Toc162348604)

[3.2 תיאור תוכנה 8](#_Toc162348605)

[3.2.1 Design 8](#_Toc162348606)

[3.3 השוואות בין תוצאות הסימולציה לעבודה בזמן אמת 14](#_Toc162348607)

[3.3.1 מד תאוצה וג'ירו 15](#_Toc162348608)

[3.3.2 מד מרחק 19](#_Toc162348609)

[3.3.3 מצלמה תרמית 20](#_Toc162348610)

[3.3.4 מד לחץ 21](#_Toc162348611)

[3.4 ביצועי המערכת מבחינת זמן אמת 22](#_Toc162348612)

[4 סיכום, מסקנות והצעות להמשך 22](#_Toc162348613)

[5 תיעוד הפרויקט 23](#_Toc162348614)

**רשימת איורים**

[איור 1 –דיאגראמת בלוקים 4](#_Toc162348568)

[איור 2 – דיאגרמת בלוקים 7](#_Toc162348569)

[איור 3-מבנה המערכת 7](#_Toc162348570)

[איור 4- תיאור חומרתי של חיבורים מול הבקר 8](#_Toc162348571)

[איור 5-תיאור מחלקות התוכנה 9](#_Toc162348572)

[איור 6-מכונת מצבים של הבקר 10](#_Toc162348573)

[איור 7 -פקודת סריקה שנשלחת דרך PI 11](#_Toc162348574)

[איור 8-צילום של ערכי סנסורים שהPI מקבל 12](#_Toc162348575)

[איור 9-מכונת מצבים במקרה של חיבור סריאלי 12](#_Toc162348576)

[איור 10-מכונת מצבים במצב של WIFI 13](#_Toc162348577)

[איור 11-מכונת מצבים של מסטר(פייתון) 14](#_Toc162348578)

[איור 12-תוצאת בדיקה מצב סטטי ג'ירו באוויר 15](#_Toc162348579)

[איור 13-תוצאות בדיקה סטטית של אוריינטציה 15](#_Toc162348580)

[איור 14-תוצאות טלטול במד תאוצה 16](#_Toc162348581)

[איור 15-תוצאות טלטול בג'ירו 17](#_Toc162348582)

[איור 16-תוצאות בדיקת ויברציה 17](#_Toc162348583)

[איור 17-תוצאות בדיקת הרוטציה מד התאוצה 18](#_Toc162348584)

[איור 18- תוצאות בדיקת רוטציה ג'ירו 18](#_Toc162348585)

[איור 19-תוצאת בדיקת מד המרחק 19](#_Toc162348586)

[איור 20-ממוצע טמפ' של חיישן AMG 20](#_Toc162348587)

[איור 21-בדיקת אינטרפולציה על יד מול חיישן AMG 21](#_Toc162348588)

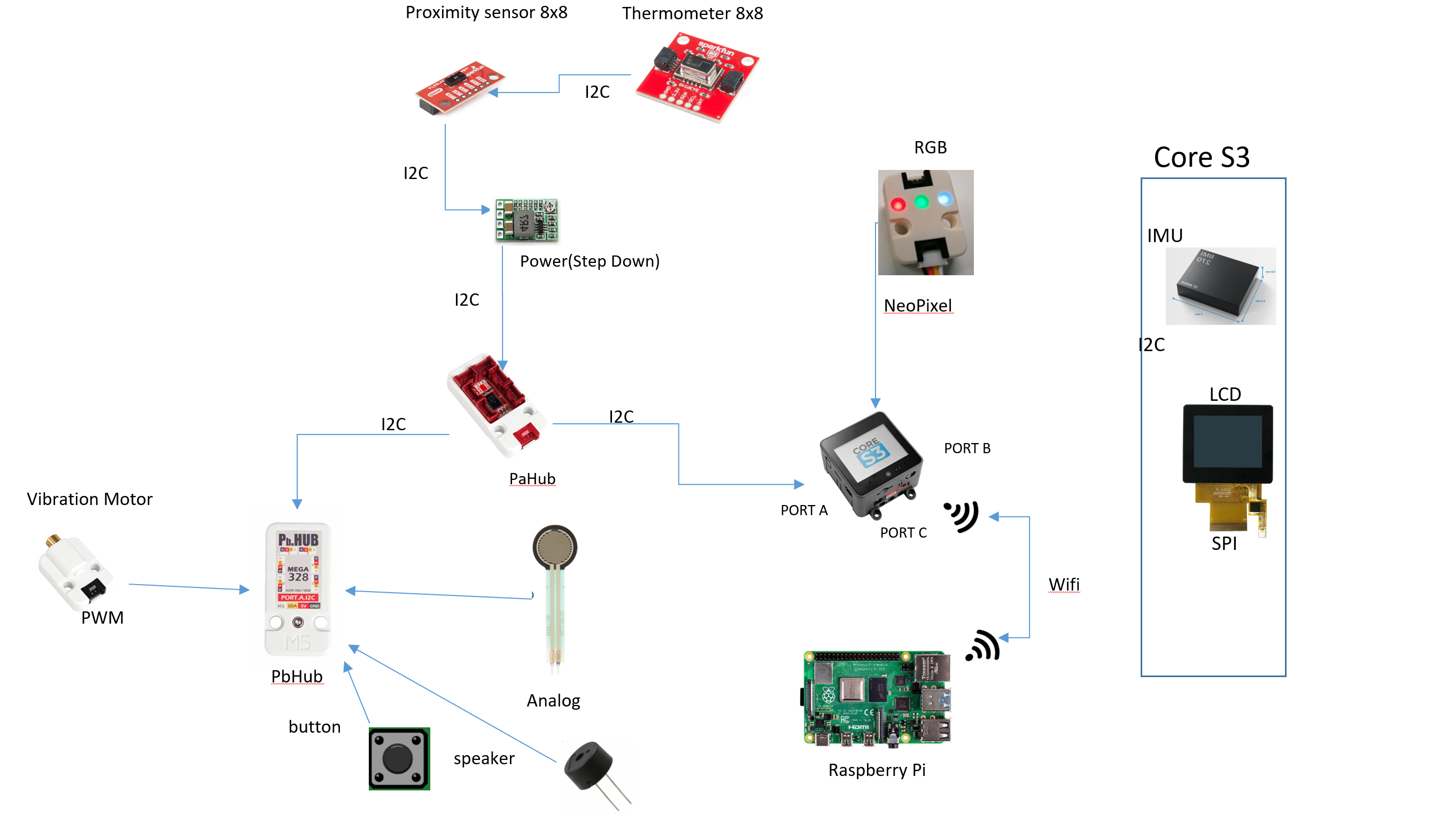
[איור 22-תוצאות בדיקת FSR לחיצה ושחרור 22](#_Toc162348589)

תקציר

פרויקט גמר זה עוסק בבניית תשתית מבוססת **Arduino**עבור פרויקטים עתידיים בהנדסת חשמל בדגש על פרויקטים בתחום הפיזיותרפיה.

פרויקטים רבים שנעשים בפקולטה להנדסה עוסקים בין היתר ביצור תשתיות חומרה ותוכנה עצמיות אשר ייחודיות עבור הצרכים הנקודתיים שלהם לפרויקטים. בפרויקט זה, אנו ניתן מענה לצורך זה על ידי כך שנספק **תשתית אחידה ורחבה** ככל הניתן בעבור פרויקטים אחרים שיהיו זקוקים לה.

סביבת העבודה והתשתית תכיל את **כניסות הסנסורים השונים ויציאותיהם** ובנוסף לכך חיווים ויזואליים וצלילים.



איור 1 –דיאגראמת בלוקים

# הקדמה

בפרק זה יתוארו:

* מטרות הפרויקט
* המוטיבציה
* הגישה לפתרון הבעיה
* השוואה כנגד עבודות ואלגוריתמים/מימושים קיימים בנושא

לשימוש בפרויקט השאפתני שלנו. המטרה העיקרית שלנו היא להקים תשתית מבוססת Arduino שתשמש כבסיס למיזמים עתידיים בתחום ההנדסה החשמלית, עם דגש מיוחד על פרויקטים הקשורים לפיזיותרפיה.

מטרתנו היא לספק תשתית מוכנה, שחיישניה נבדקו על ידינו, והיא תהיה קלה לשימוש עבור סטודנטים ותשמש כתשתית אחידה לעבודה. בנוסף, התשתית צריכה להיות מסוגלת לעבוד באינטגרציה עם Raspberry Pi באמצעות Wi-Fi.

פיתחנו תשתית קוד קבוע FIRMWARE ועבודה מול הבקר M5Stack עבור התממשקות עם החיישנים, למצב עבודה עצמאי(STANDALONE) או מצב עבודה של חיבור MASTER-SLAVE, כשהבקר שלנו היננו הslave. פיתוח תשתית Python שהננה הmaster, שניתן לעבוד איתה ב-Windows או באמצעות חיבור סריאלי או Wi-Fi ולקבל מחוון מחיישנים או להפעיל חיישני פלט (כמו רמקול). ניתן להתחבר ל-Raspberry Pi באמצעות Wi-Fi . ההסבר על עבודה עם התשתית נמצא בקבצי הפרויקט עצמו, וכיצד ניתן להרחיב את יכולותיה במידת הצורך.

# רקע תיאורטי

### M5STACK

בפרויקט זה, נשתמש ב-M5Stack CORES3 ESP32S3 כבקר הראשי. ה-M5Stack ששייך למשפחת המיקרובקרים, ESP32 מציע ביצועים חזקים ותאימות רחבה עם ספריות תוכנה וחומרה רבות, ולכן בעל התאמה רבה לפרוייקט שלנו. הוא כולל חלק מן חיישנים משולבים הדרושים לפרויקט, מה שמפשט את תהליך ההתקנה ומקטין את הסיכוי לטעויות בחיבורים. בנוסף, ה-M5Stack יעיל מבחינת צריכת החשמל וניתן לתכנת בקלות באמצעות Arduino IDE , ומציע גמישות בהתאמת מספר חיישנים, ומאפשר לסטודנטים לחקור ולחדש בתחום היישומים המבוססים חיישנים.

### PaHub

הצורך להרחיב את יציאת ה-I2C מתקיים באמצעות HUB, שדרכו אנו יכולים להוסיף עוד חיישנים עם חיבור Grove .אנו משתמשים ב AP9548PCA (B040-U)- שדרכו אנו יכולים לחבר כשישה חיישנים.

### PbHub

רכיב זה הוא עם יציאת I2C ודרכו ניתן לחבר עד שישה חיישנים שאינם I2C , כמו GPIO,PWM,ANALOG וכו'. מכיל פנימית בתוכו ממיר ADC ומכיל בקר פנימי STM32F030

.

### מד תאוצה וג'ירו

רכיב Bosch BMI 270 הינו IMU, המובנה בתוך הm5stack, אשר משלב פנימית בתוכו חיישן בעל 6 צירים כך שג'ירוסקופ תלת צירי ו16 סיביות ומד התאוצה תלת צירי 16 סיביות.

### מד מרחק

מד מרחק SparkFun Qwiic ToF Imager VL53L5CX. היננו חיישן בעל 64 פיקסלים, טווח סריקה של עד כ4 מטרים וראייה אלכסונית של עד כ-63 מעלות. כאמור סנסור זה משלב בתוכו מערך SPAD, מסנני אינפרא אדום פיזיים ואלמנטים אופטיים עקיפים בכדי לקבל ביצועים טובים ביותר עבור טווחי תאורות שונות וסביבות שונות.

### מצלמה תרמית

מצלמה תרמית SparkFun Grid-EYE Amg8833 סנסור זה מכיל מערך תרמופילים בגודל 8 על 8 ( 64 פיקסלים ) אשר מזהה באמצעות אינפרא אדום טמפרטורה . קצב הדיוק היננו , וטווח טמפרטורות . מצלמה תרמית זו יכולה לזהות חום גוף אדם עד כ7 מטרים וקצב הפריימים שלו היננו בין 1-10 פריימים לשנייה.

### מד לחץ

חיישן FSR402 הינו נגד חישת כוח, עם אזור יחיד אשר מותאם לשימוש בבקרת מגע אנושי של מכשירים אלקטרוניים. כאמור FSR מכשיר דו-חוטי בעל חיישן סרט עבה פולימרי אשר מפגין ירידה בהתנגדות עם עליית הפעלת הכוח עליו. מדובר על חיישן אנאלוגי.

### LCD

LCD מובנה בm5stack הדגם היננו ILI9342C.

### Vibration Motor

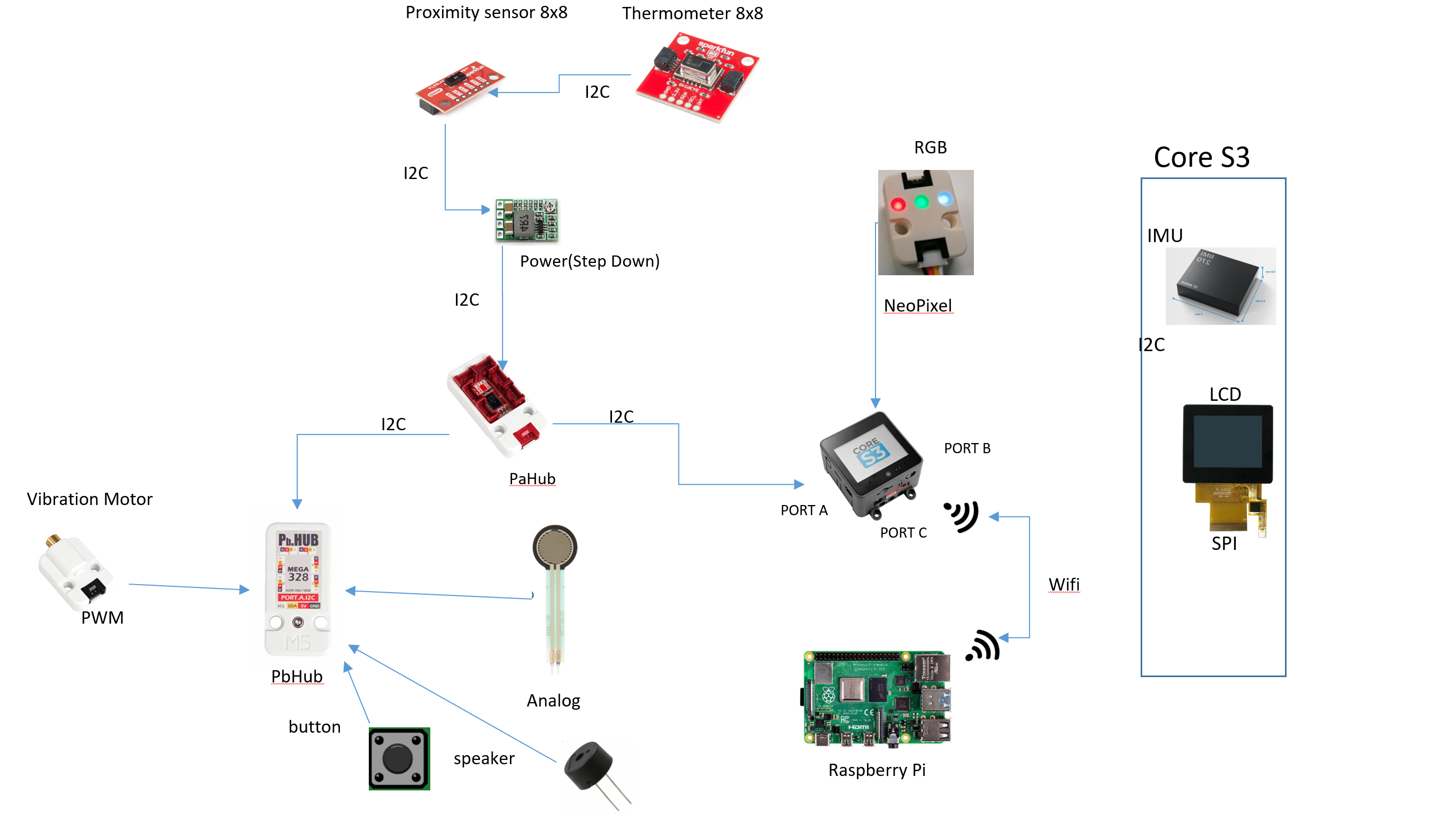
מנוע רטט המכיל גלגל אקסצנטרי מתכתי. מהירות הסיבוב שלו היננו של .

### RGB

יחידת RGB עם 3 נוריות לד בודד.

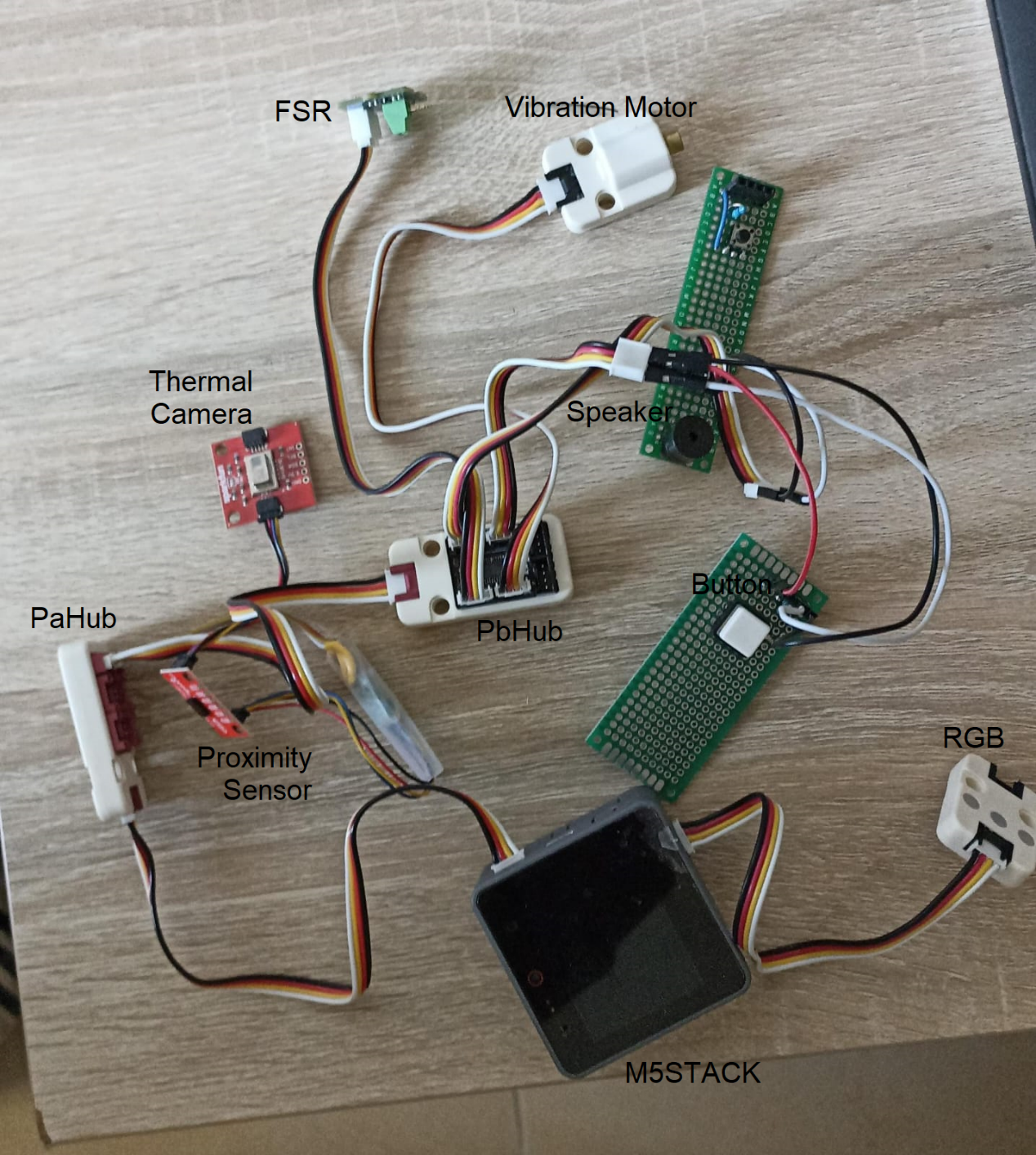
# מימוש

בפרק זה יתואר המימוש והשיקולים לבחירתו, ההקדמה תכלול תיאור כללי, כולל דיאגרמת בלוקים מפורטת עבור אופן מימוש הפרויקט.



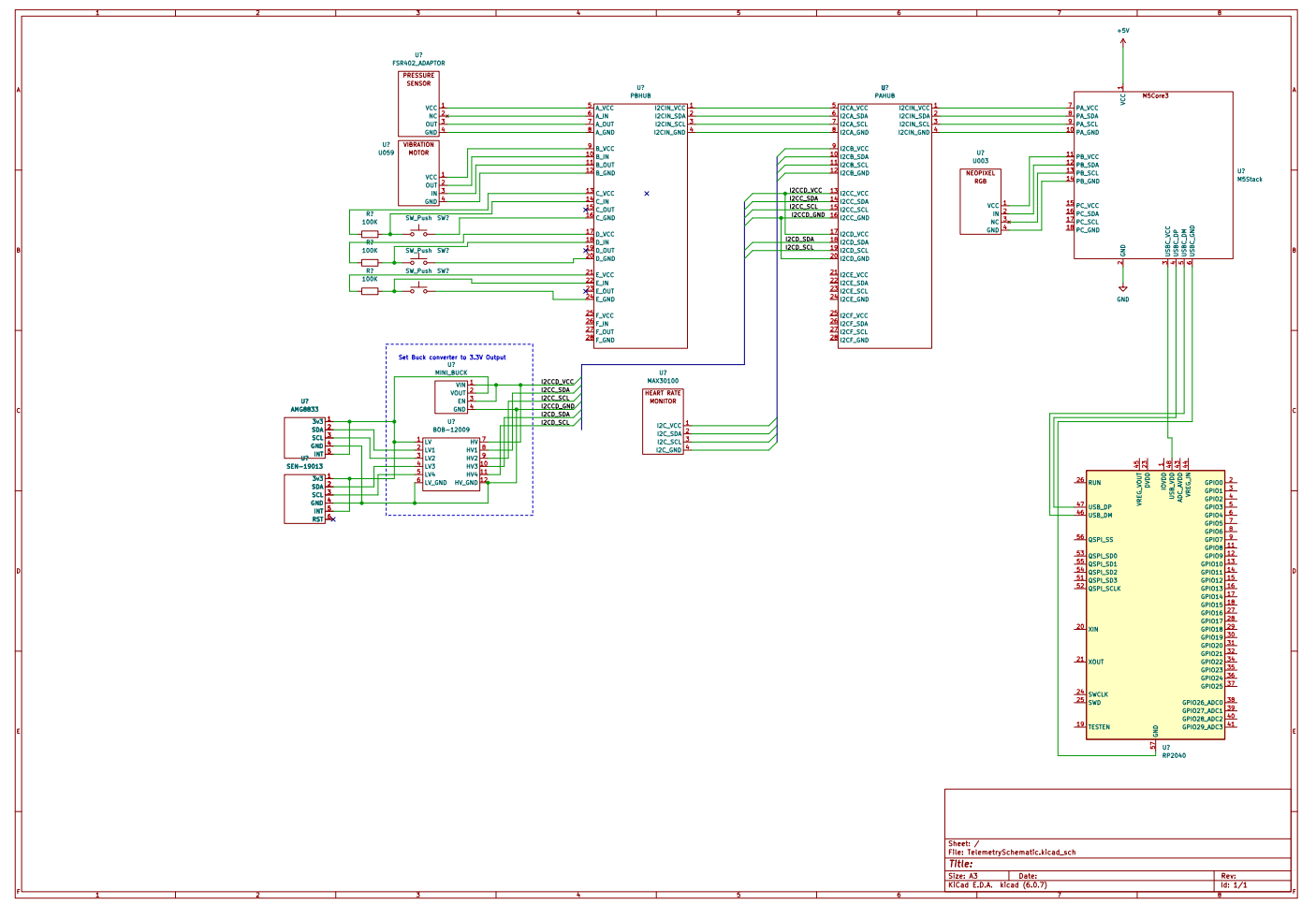
איור 2 – דיאגרמת בלוקים

ומבנה המערכת



איור 3-מבנה המערכת

## תיאור חמרה



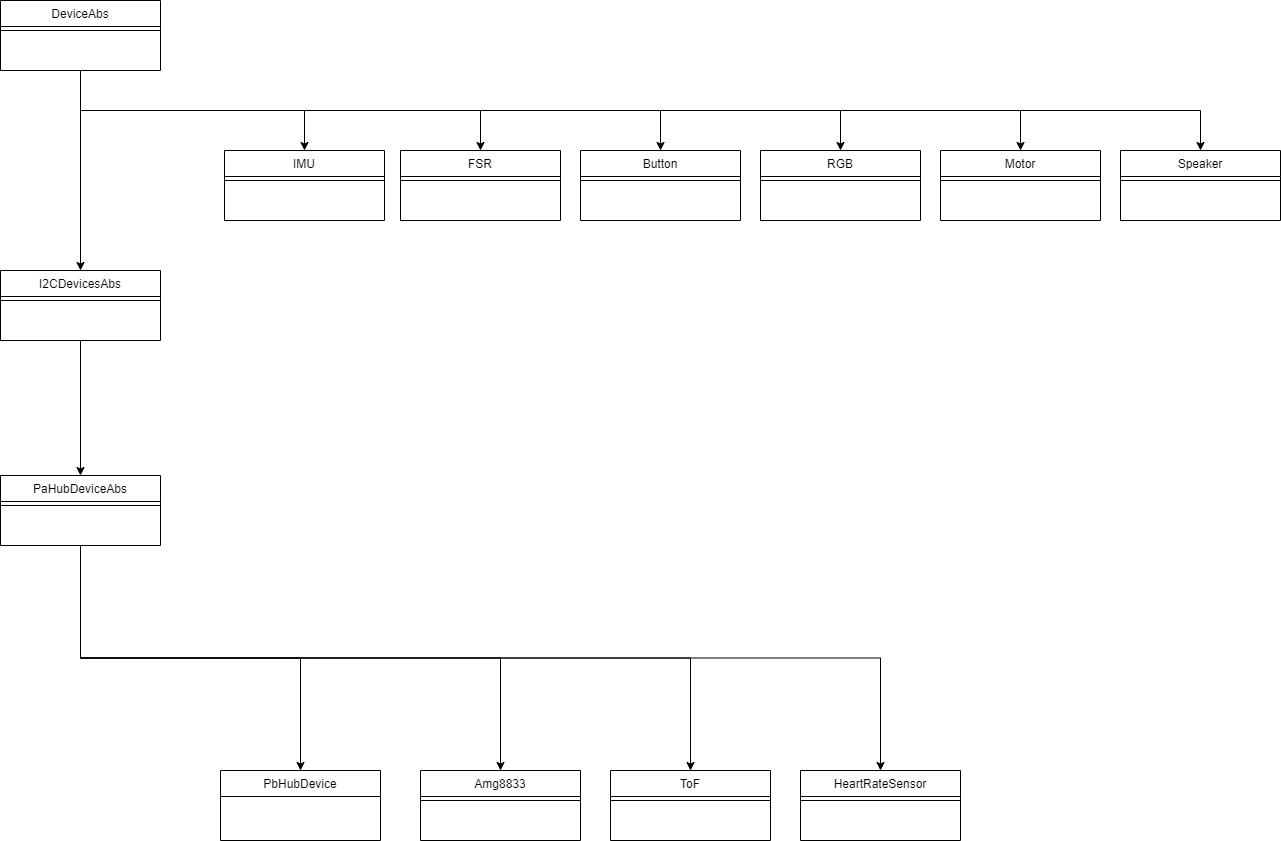
איור 4- תיאור חומרתי של חיבורים מול הבקר

## תיאור תוכנה

תיאור אופן מימוש הפרויקט בתוכנה כולל פירוט הכלים ,הפלטפורמות והסברים רלבנטיים – אין לכלול קטעי קוד. התוכנה נתמכת בwindows , ובDevian מערכת ההפעלה של Raspberry PI.

### Design

פותחה ב C++ באמצעות Arduino IDE. כאמור החלק המשמעותי הינו היה Design המחלקות של הפרויקט, שיהיה בצורה מודולרית ויאפשר בהמשך הוספה או הסרה של סנסורים או יכולות נוספות לסנסורים הקיימים. לכן זוהי היררכיית המחלקות שרלוונטיות ל**סנסורים** בלבד:



איור 5-תיאור מחלקות התוכנה

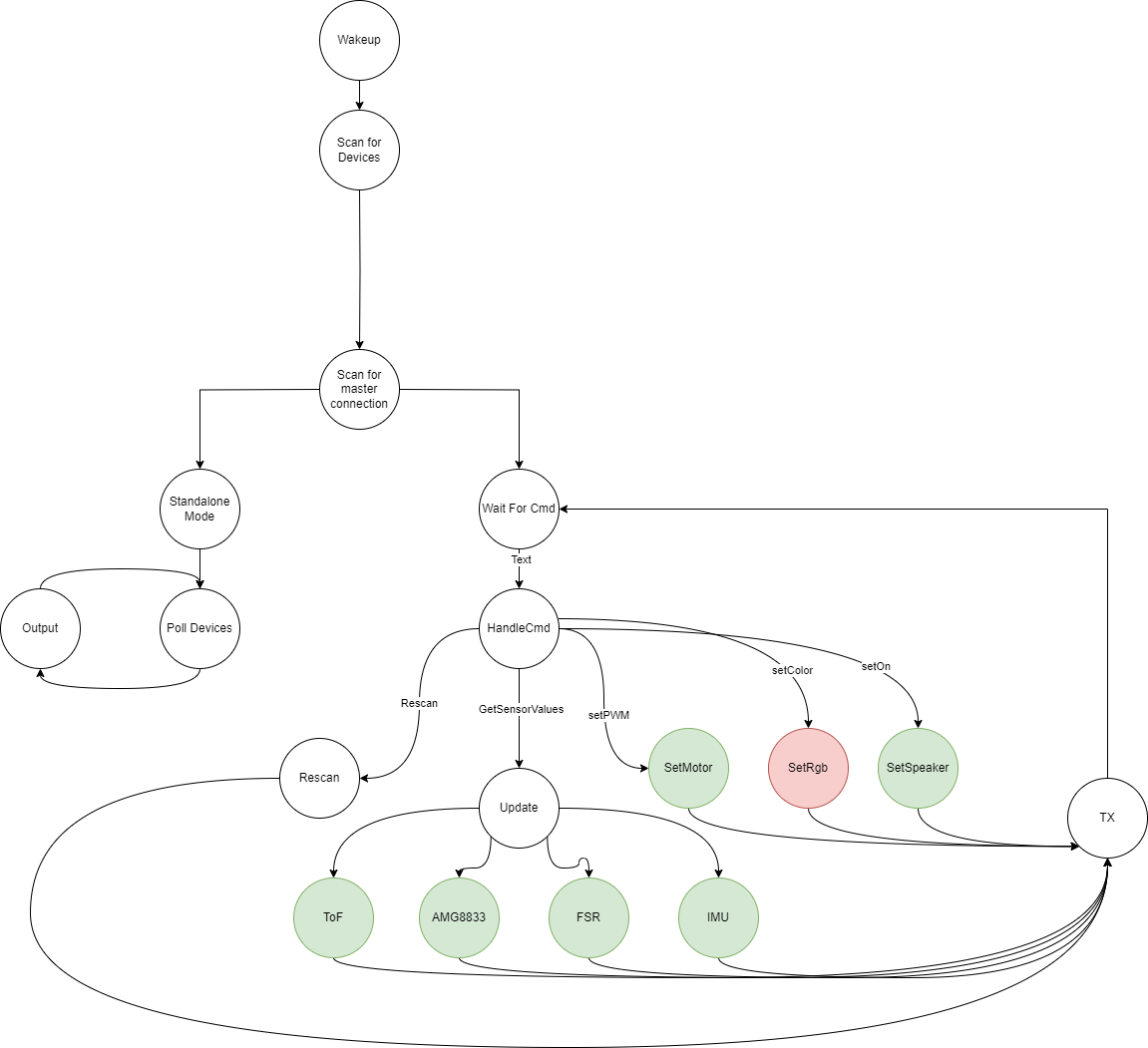
המחלקה הבסיסית DeviceAbs, היא מחלקה אבסטרקטית שכולם יורשים ממנה שמכילה יכולות בסיסיות שכל סנסור תומך בהן, לכן הMethods שם וירטואליות.

כל מה שאינו I2C(חוץ ממד תאוצה והג'ירו שמובנה כבר בm5 ולכן לא היה צורך לשלשל בהירככיה), יורש באופן יש מDeviceAbs(סנסורים אנאלוגים/PWM).

מה שהינו מתחבר ל PaHub ,יורש מ PaHubDeviceAbs, שמכיל יכולות רלוונטיות לכל סנסורים שעובדים עם I2C.

המחלקה שבה משתמשים יעזרו לצורך שימוש בstandalone/slave, יהיה מחלקה שנקראת M5Telemetry. זו מחלקה שבה יש ניהול של כל המערכת, הפקודות האפשריות שניתן לשלוח במצב SLAVE, ניהול מצב STANDALONE, סריקת מכשירים מחוברים וכדומה.

להלן מכונת המצבים:



איור 6-מכונת מצבים של הבקר

נסביר כעת על כל חלק במכונת מצבים

#### סריקת מכשירים מחוברים

בעיה מהותית הקיימת, בהינתן ומחובר רכיב I2C, בPaHub, כיצד נוכל לוודא שהוא מחובר ולהכין מחובר במוליפלקסר? כאמור ביצענו בקושחה פתרון עבור בעיה זו.

הפתרון הינו חיפוש בכל מרחב כתובת הi2c, וכאשר יש כתובת ידועה באמצעות החיפוש הצלחנו למצוא אותם סנסורים – לכן את אותם סנסורים ניתן למצוא באמצעות המימוש שלנו באופן אוטומטי(לא דורש שינוי תכנוני), הדבר היחידי שנדרש רק להוסיף אותם למערך הסנסורים של הapi, בכדי שהתוכנה תכיר שקיים סנסור כזה.

בנוגע לשאר הפרוטוקולים – היות ואין אפשרות לוודא באופן חד משמעי אם אכן מחוברים או לא – אנו מאלצים את המשתמש להגדיר האם הם מחוברים או לא(כלומר **באחראיות המשתמש לוודא שמה שמחובר**), ולהיכן מחובר.

#### STANDALONE

במצב זה, הבקר רץ באופן עצמאי, ועושה POLLING, על המדיע של הסנסורים המחוברים ומדפיס למסך. ישנם 2 אפשרויות:

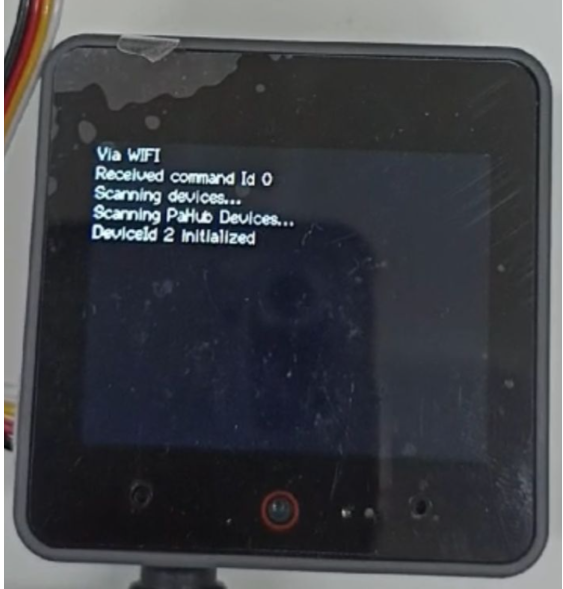
* להגדיר זמן מסויים שתוצאות סנסורים יוצגו למסך ( לדוגמה כל 10 שניות ),
* הגדרת כפתור, ולתת אופציה למשתמש מעבר בין הדפסות של סנסורים באמצעות הכפתור.

הגדרת OUTPUT SENSORS- כלומר מנוע ויברציה, Speaker תחת תנאים מסוימים של מוצאים מסוימים דורש שינוי התנהגות בקוד FW, כלומר הוספה תמיכה שלהם במחלקות הרלוונטיות.

#### SLAVE MODE

כאמור המצב העיקרי והמטרה העיקרית של הפרויקט. בתחילת עולם באמצעות מחלקה שנקראת CommandHandler, מנסה להתחבר ל-MASTER(המחשב או הpi) באמצעות חיבור סיריאלי(UART), או WIFI. יש להגדיר ל**פני** צריבת הגרסא יש לשנות את פרטי הWifi שרוצים להתחבר אליו.

עם ההתחברות CommandHandler,ממתין למידע מהMASTER, ובהתאם לבקשה שמקבל מטפל בפקודה ואם יש צורך בסיום הפקודה מחזיר מידע חזרה(נרחיב בהמשך על כיצד זה עובד).



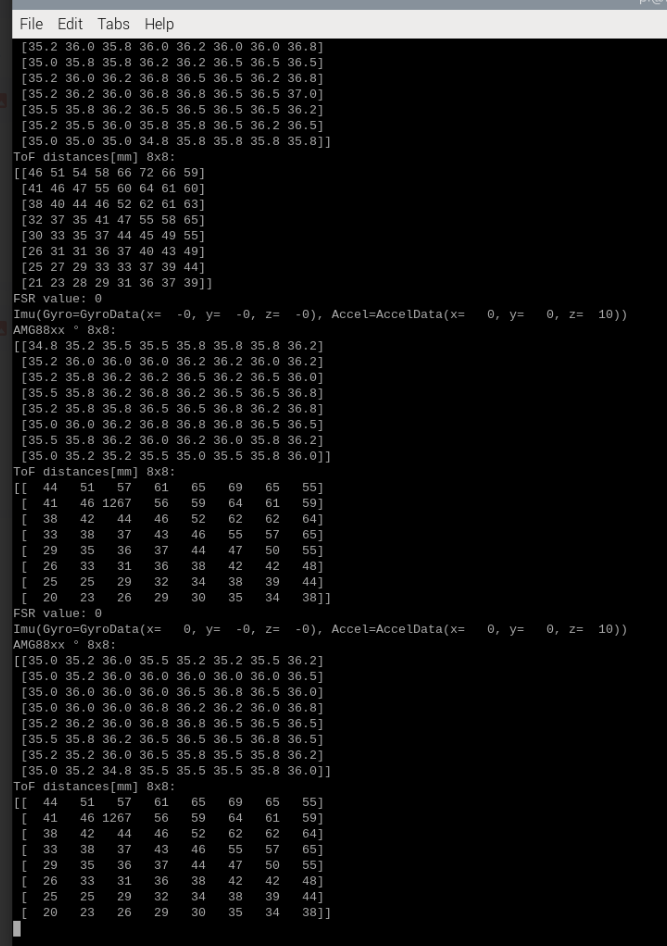
איור 7 -פקודת סריקה שנשלחת דרך PI

יצרנו Python API, שמתנהג כMaster, והוא זה שמחכה להתחברות של הבקר אליו. עם ההתחברות דרכו ניתן לשלוח דרכו פקודות כמו בקשה של ערך של סנסור מסויים ואותה בקשה נשלחת לבקר דרך המחלקה CommandHandler שצוינה קודם לכן.

הוספנו פקודות נוספות כמו להפעיל את המנוע ויברציה ולכבותו, או להפעיל את הבאזר דרך הפייתון.

המטרה של API שיצרנו זה להקל על המשתמש ויצירת API אמין לשליחת וקבלת מידע מהבקר.

כאמור מצב זה נבדק מול התוכנה של RASPBERRY PI, שמתנהג כמאסטר. את התוצרים ניתן להמיר למטריצת NUMPY, לצורך אנליזה(נעזרנו בהם לניתוח תוצאות בדיקות החומרה) או הדפסה ידידותית למשתמש, למסך.

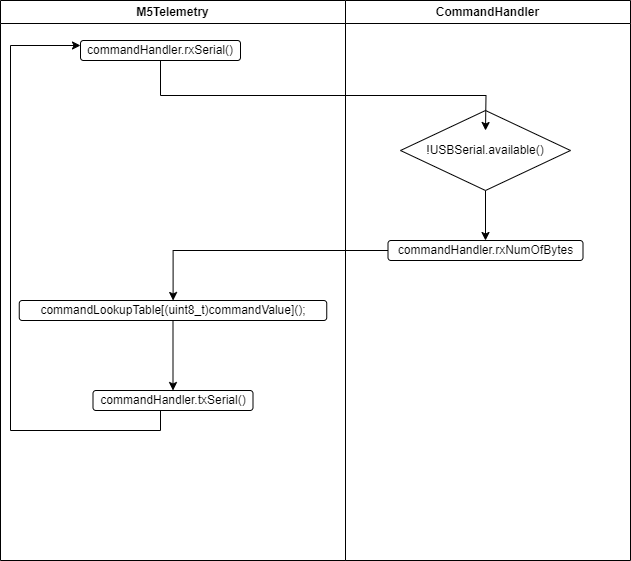


איור 8-צילום של ערכי סנסורים שהPI מקבל

כעת נרחיב על 2 סוגי החיבורים(מכונת מצבים שלהם).

##### Serial

כאמור בחיבור באמצעות USB,כמו שהוסבר קודם לכן החיבור הוא UART. כאמור זוהי מכונת המצבים ההתנהגותית כאשר שולחים פקודה באמצעות הmaster,



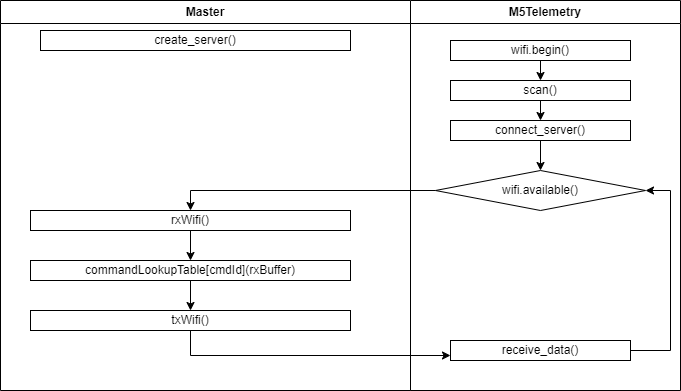
איור 9-מכונת מצבים במקרה של חיבור סריאלי

כפי שניתן לראות זוהי התנהגות מחזורית, בתחילת עולם בCommandHandler, אם מקבלים מידע, בחלק עוברים לשלב השני של המכונת מצבים שזה קבלת המידע אל תוך BUFFER פנימי(תוכנה). עם קבלת המידע מהMASTER, קוראים בlut, שקיים ל API, בו כל פקודה יש מספר סידורי מוגדר מראש על ידי המשתמש. באותה פקודה, יודעים לשחזר את המידע שהועבר מהMASTER, בתור ארגומנטים לאותה פקודה, מימוש הפקודה, ואם הסיום קוראים לפונקציה פנימית של CommandHandler, שיודעת לשדר בחזרה שסיים לעבוד, ובמידת הצורך להחזיר מידע חזרה לmaster.

חשוב להדגיש כפי שהוסבר כאן, שברמת המשתמש, יהיה צורך בהוספת פקודות חדשות או שינוי קיימות, לשנות ידנית בצד הmaster ובצד הslave, את הארגומנטים של אותם פונקציות / או כיצד לפרסר את המידע שכל צד מקבל.

* **לשים לב, שהבקר תמיד ינסה להתחבר לחיבור סריאלי(זמן ריצה האופטימלי)**.
* **נתמך רק בwindows**.

##### WIFI



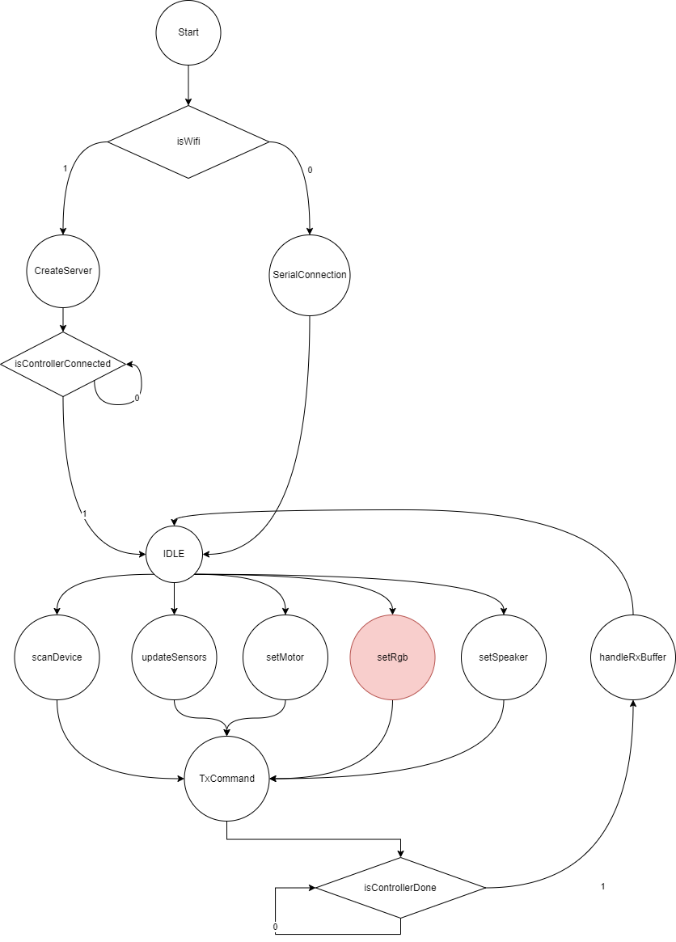
איור 10-מכונת מצבים במצב של WIFI

ניתן לראות שהפתרון עבור WIFI, מורכב יותר היות וזה דורש הקמת שרת בצד של הmaster, בתחילת עולם. על **המשתמש** בטעינת התוכנה אל הבקר(UPLOAD), לדאוג לרשום את פרטי הwifi הנכונים וכתובת הip שבו השרת נמצא(ע"י ipconfig בוינדוס, או ע"י ifconfig בפי).

בתחילת עולם אם לא הצליח הmaster להתחבר לslave באמצעות חיבור סריאלי, ינסה להתחבר דרך WIFI. לאחר שהצליח להתחבר לwifi, הבקר ינסה להתחבר לשרת, ועם הצלחה להתחברות בשרת, ההתנהגות התוכנתית זהה. הבקר ימתין לפקודה מהמאסטר, ועם קבלת הפקודה יטפל בה ויחזיר בחזרה שסיים ובמידת הצורך יעביר מידע נוסף שהמאסטר ביקש מראש(תלוי פקודה).

##### מכונת המצבים של תוכנת המאסטר(פייתון)

להלן מכונת המצבים בצד של המאסטר



איור 11-מכונת מצבים של מסטר(פייתון)

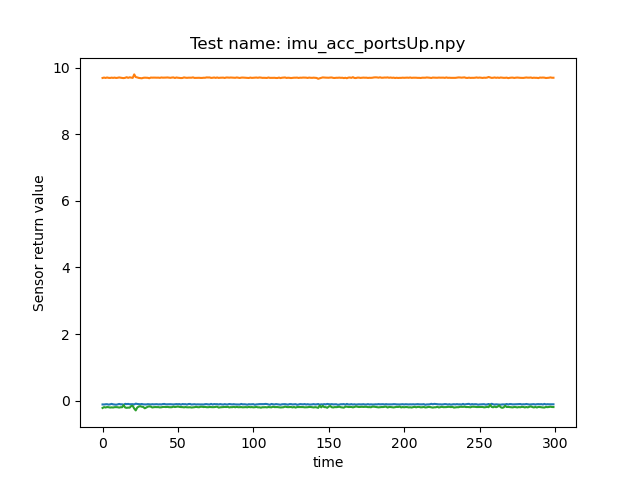
ניתן לראות שבתחילת עולם בודקים, האם המשתמש במאסר ביקש לייצר חיבור באמצעות Wifi או Serial. במידה ומדובר על חיבור בWifi, המאסטר מייצר שרת ומחכה לחיבור של הבקר. במידה ומדובר על חיבור סריאלי – הפייתון מייצר חיבור סריאלי מול הבקר.

לאחר ההתחברות, הפייתון במצב IDLE, ומחכה לפקודה שהמשתמש יכניס, את אותה פקודה הוא משדר לבקר, והבקר יטפל בה. עם סיום הפקודה הבקר יודיע למאסטר על האם יש מידע להעביר בחזרה, במידה וכן המאסטר יחכה לשידור כל המידע מהבקר. לאחר מכן במידה ויש מידע שהתקבל מהבקר הפייתון בהתאם לפקודה מטפל בבקשה, במידה ואין מידע הbuffer(שמכיל את כל המידע) יהיה ריק ויחזור למצב IDLE שוב.

## השוואות בין תוצאות הסימולציה לעבודה בזמן אמת

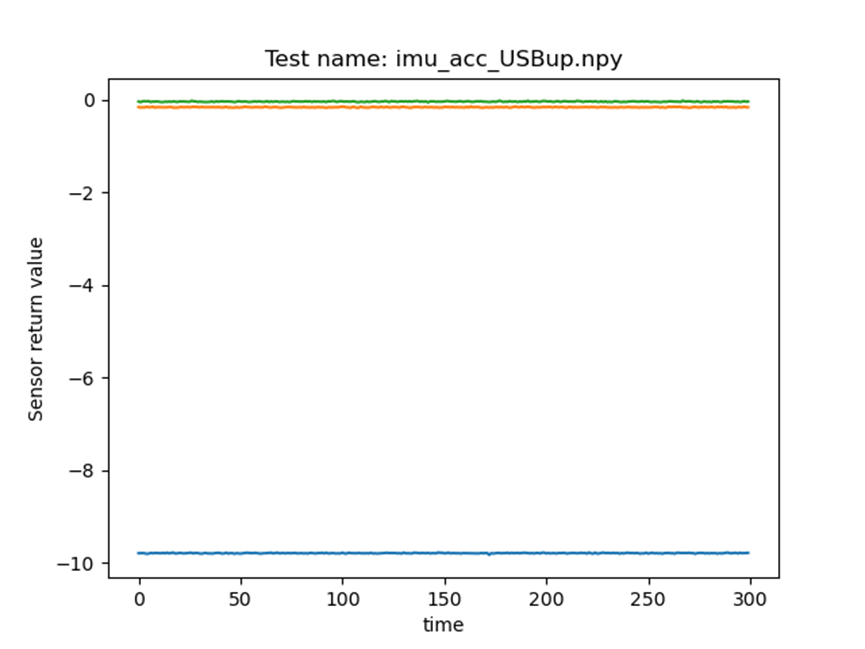
את התוצאות הפקנו באמצעות שימוש במצב STANDALONE,ע"י לקיחת סט דגימות לבדיקה מסוימת. לחלק מהסנסורים נדרש מספר בדיקות בעוד שבחלקם נדרש רק בדיקה אחת(תלוי מורכבות סנסור).

### מד תאוצה וג'ירו



איור 12-תוצאת בדיקה מצב סטטי ג'ירו באוויר

החזקנו את המד תאוצה באוויר(מצב סטטי). ניתן לראות שקיבלנו בציר Z בקירוב טוב את תאוצת כדור הארץ ובשאר הצירים השינוי בציר X,Y יציב.



איור 13-תוצאות בדיקה סטטית של אוריינטציה

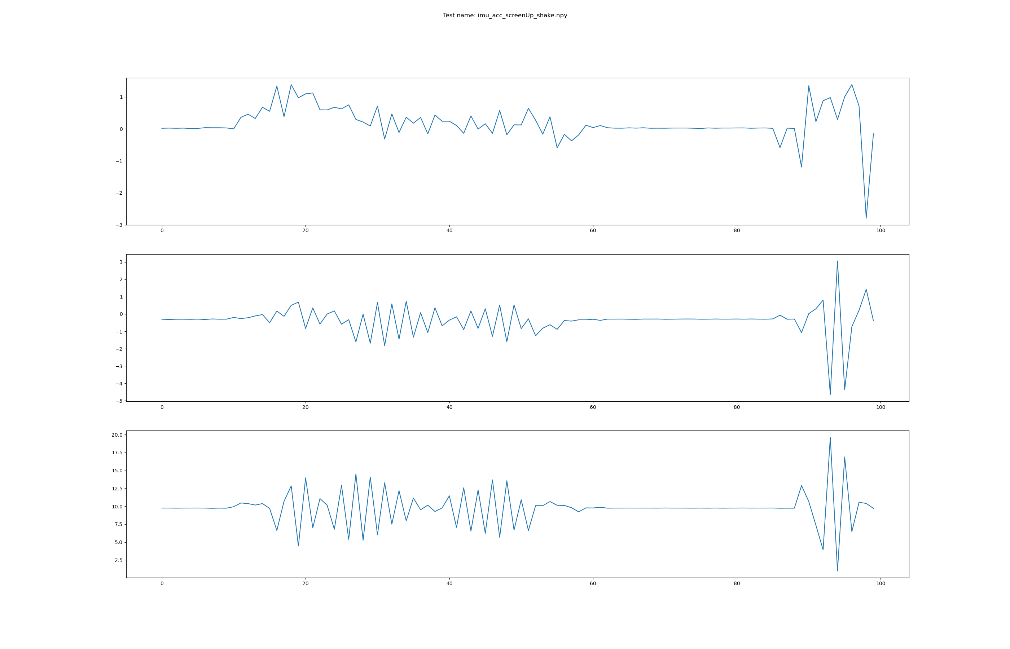
ביצענו בדיקה סטטית בכדי לוודא את האוריינטציה של המד תאוצה וג'ירו, נשים לב ש:

* מיקום המסך למעלה חיובי בציר Z
* מיקום ה-USB למטה חיובי בציר X
* מיקום יציאת IIC למעלה חיובי בציר Y

כלומר קיבלנו את התוצאות שציפינו שנקבל בבדיקה זו

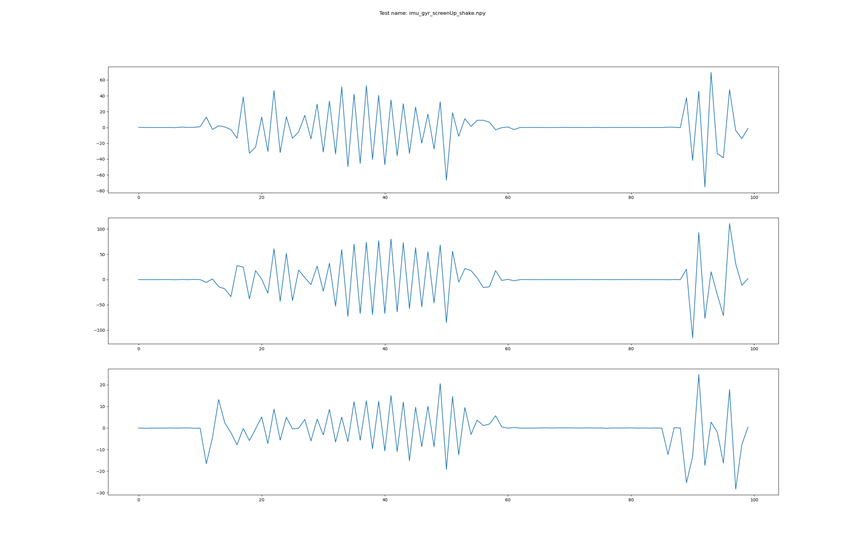
#### טסטים דינמיים של המד תאוצה וג'ירו

ביצענו ניעור של ההמד תאוצה וג'ירו , ולאחר מכאן הרמה שלו, בכדי להקל ביצענו הפרדה בין תוצאות הג'ירו, לתוצאות Accelerometer.



איור 14-תוצאות טלטול במד תאוצה

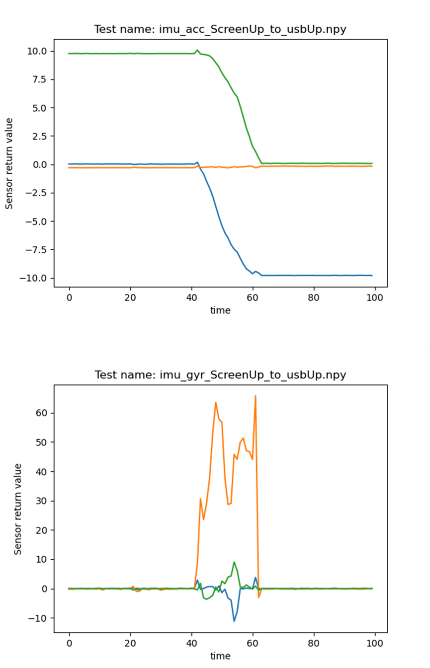
ניתן לראות על פי מד התאוצה שיש ניעור(אפשר לראות ש-3 הצירים מושפעים כפי שצופה כי עשינו תנועות מעגליות בניעור).



איור 15-תוצאות טלטול בג'ירו

ניתן לראות בג'ירו שאכן יש ויברציות המתארות ניעור (נראות בצורת רשרוש), לכן נשיק שהג'ירו והמד תאוצה במצב דינמי נותנות תוצאות טובות ומתארות בצורה ובה את הפעולה שביצענו.

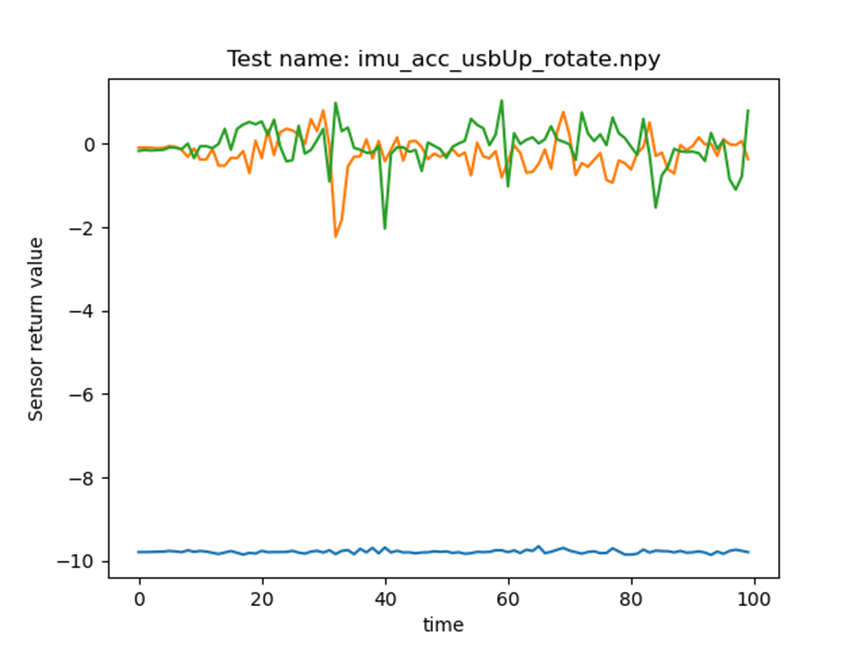
בבדיקה הבאה מה שעשינו זה הרמנו ממשטח את הבקר, עד ההגעה לגובה רצוי, ולאחר מכאן רוטציה בציר ה-X.



איור 16-תוצאות בדיקת ויברציה

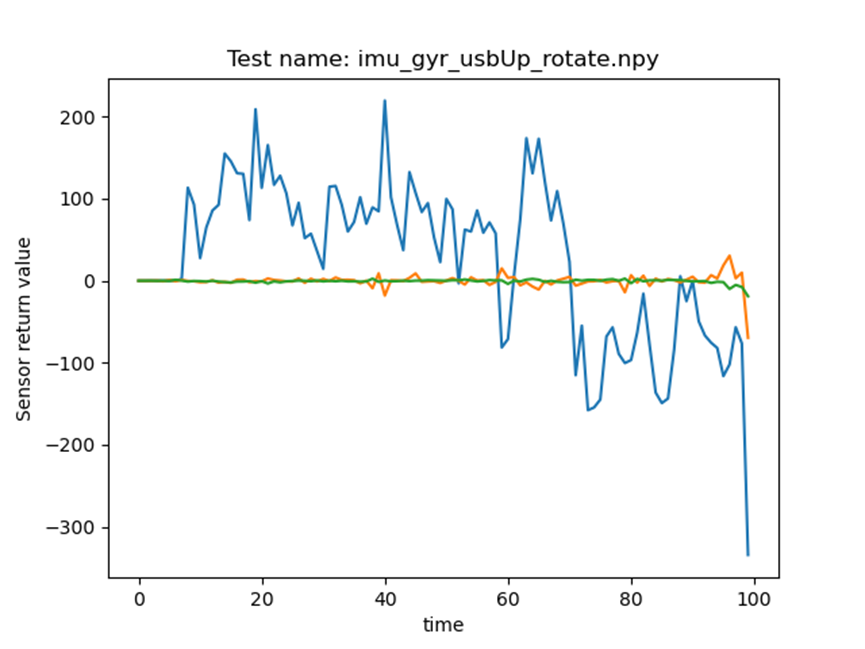
ניתן לראות כי אכן הבדיקה מתארת הרמה בתחילת עולם, עד אשר מגיעים לגובה הרצוי ועם ירידת התאוצה והשפעה על ציר הסיבוב ניתן לראות כי אכן בוצעה רוטציה כפי שצופה(ניתן לראות זאת על פי התאוצה וגרף המיקום).

הבדיקה האחרונה שביצענו זו בדיקת רוטציה של המד תאוצה וג'ירו, על ידי 'נעילת' שתי צירים אחרים וקיום של ציר אחד בלבד:



איור 17-תוצאות בדיקת הרוטציה מד התאוצה

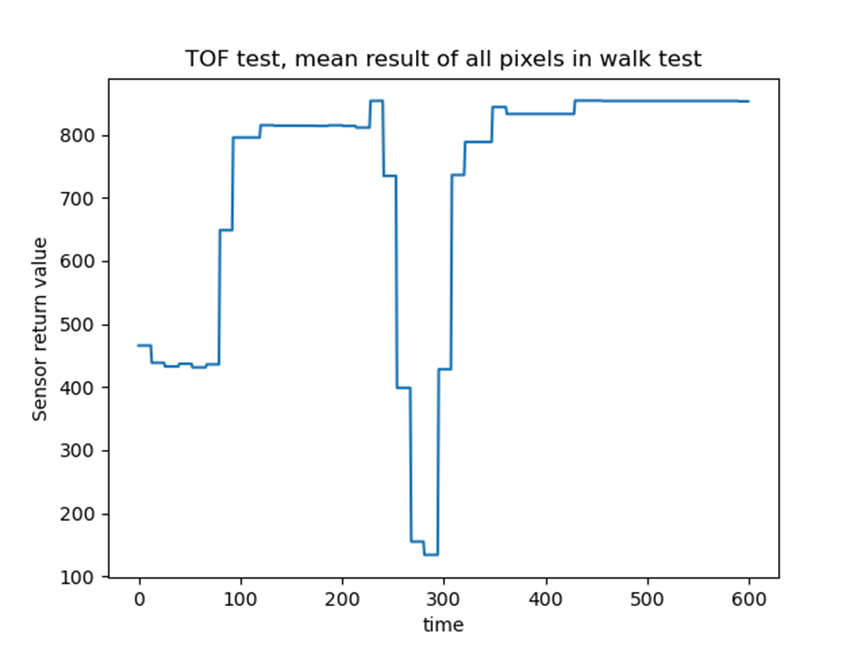
על סמך גרף התאוצה ניתן לראות כי אכן 2 הצירים אחרים נעולים וציר הרוטציה אכן בעל תאוצה.



איור 18- תוצאות בדיקת רוטציה ג'ירו

ציר הרוטציה באופן ברור מושפע יותר מאשר הערכים אחרים שבמהלך הבדיקה היו באזור ה-0(מה שאומר שהסנסור זיהה באופן טוב את ציר הרוטציה שהזזנו).

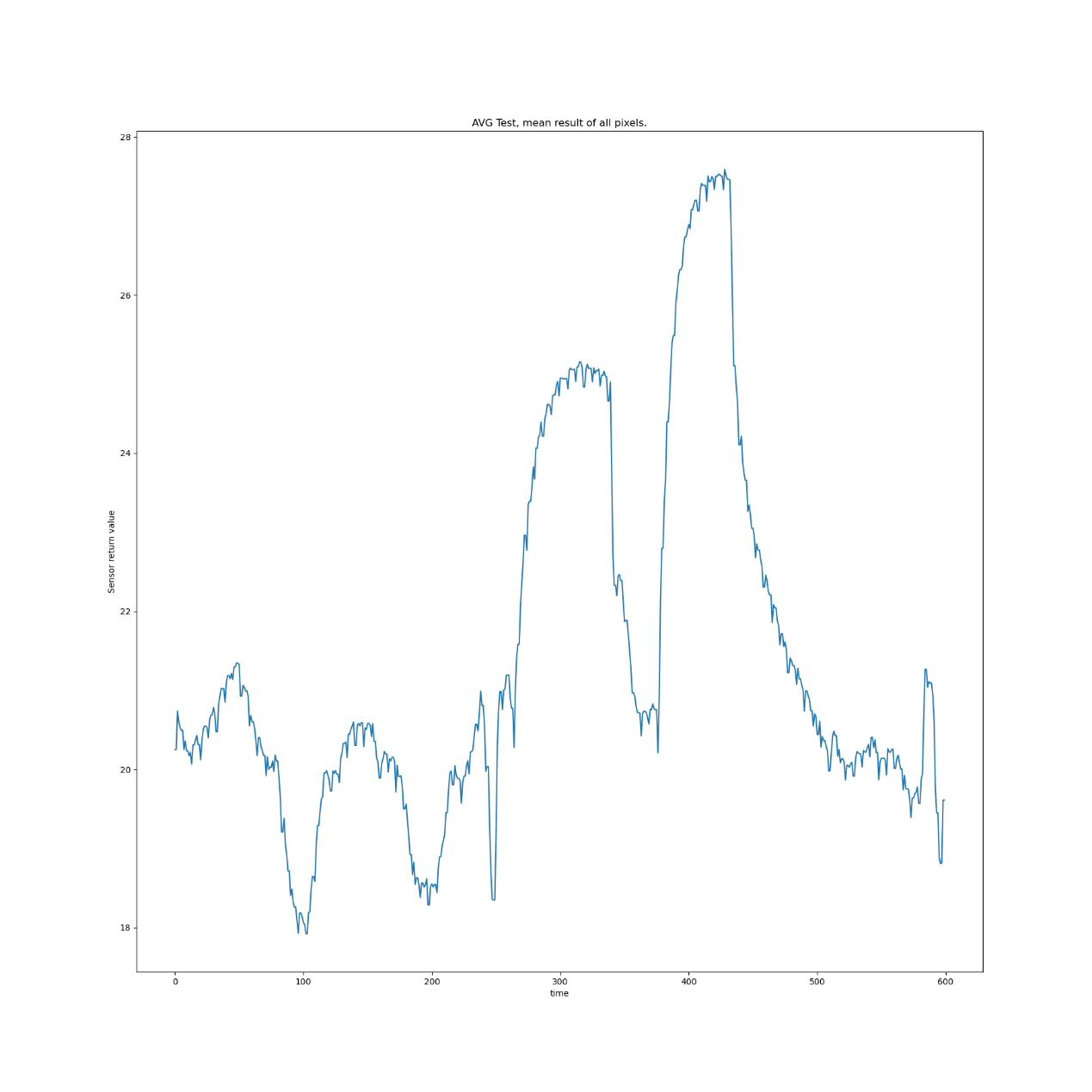
### מד מרחק



איור 19-תוצאת בדיקת מד המרחק

שמנו את היד מול החיישן וקירבנו והרחקנו את היד, ורצינו לראות בממוצע שאכן החיישן מתאר תופעת התקרבות והתרחקות. הממוצע הינו על כל ערכי של 64 פיקסלים של החיישן.

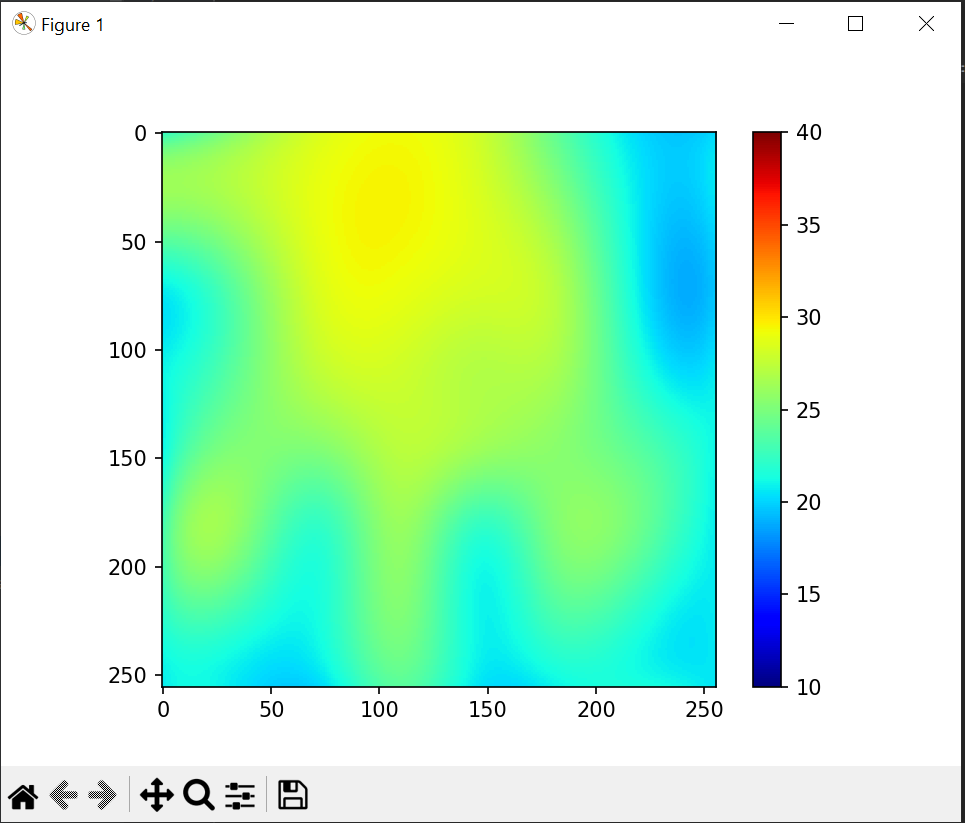
### מצלמה תרמית



איור 20-ממוצע טמפ' של חיישן AMG

עמדנו מול המצלמה תרמית, ניתן לראות שהוא משקף בצורה טובה את טמפ' הגוף של אדם העומד מול המצלמה( ממוצע הדגימות של כל 64 הפיקסלים).

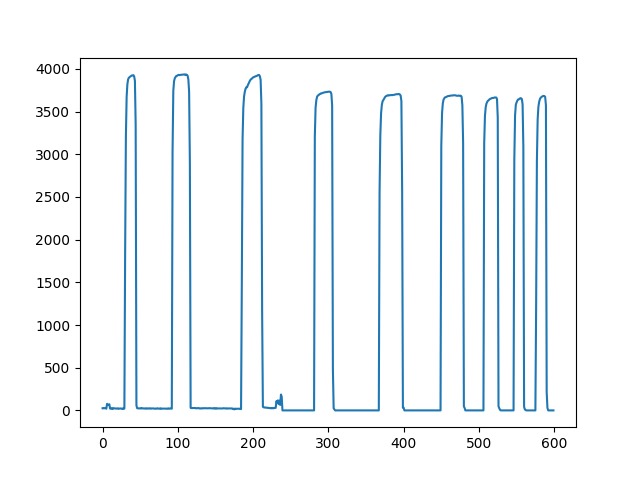
בדיקה נוספת זה נחת יד מול המצלמה התרמית ובאמצעות אינטרפולציה לוודא שאנחנו רואים את היד:



איור 21-בדיקת אינטרפולציה על יד מול חיישן AMG

ניתן לראות שבערך קיבלנו את צורת היד, צריך לקחת בחשבון שמדובר רק על 64 פיקסלים לכן קשה לראות תמונה מדויקת, אך סה"כ ניתן לראות צורה של יד כפי שצופה.

### מד לחץ



איור 22-תוצאות בדיקת FSR לחיצה ושחרור

ניתן לראות שהמד לחץ משקף בצורה טובה את הפעולה שביצענו שזה לחיצה ושחרור עם האצבע.

## ביצועי המערכת מבחינת זמן אמת

* מלבד לחיבור USB שביצע התנהגות לא צפויה(ולכן נכון לכרגע לא אמין) והRGB, המערכת תפקדה מעבר למצופה מבחינת ביצועים(תדירות דגימה שגדולה מ10 הרץ).

# סיכום, מסקנות והצעות להמשך

דברים הטעונים לשיפור בפרויקט:

* במקור היה צורך שpi יתמוך בחיבור סריאלי, אך בעקבות זה שבחרנו בקר חדש מדי – לא היה תמיכה ולכן הpi עובד רק באמצעות WIFI.
* הrgb נכשל בבדיקות שביצענו מולו ואופרטיבית התגלה כלא אמין לשימוש.

מסקנות:

* עדיף לעבוד עם גרסה לפעמים לא הכי עדכנית של מערכת, אך אמינה יותר.
* קנייה של כמה סוגי חיישנים מאותו סוג ובדיקת אמינות של כל אחד.

ניתן יהיה להרחיב לתמיכת סנסורים נוספים(הפרויקט מודולרי מראש עבור אופציה זו). ניתן להכין פרויקטים מגוונים בgui(יש דוגמאות בgit של הפרויקט עצמו). בנוסף יהיה אפשר לעדכן את קוד התוכנה של החומרה של m5.

# תיעוד הפרויקט

תיעוד הפרויקט הינו בעל ערך משמעותי, שמטרתו להקל על סטודנטים שישתמשו בעתיד בתשתית זו וגם להבטיח להם להיעזר בתשתית שנבדקה ואמינה. הפרויקט והקוד נמצא בgithub, יש תיעוד לקוד התוכנה.

הסיבה שבחרנו בgithub,היות וזו פלטפורמה נוחה לניהול פרויקטים, שבה מספר קבוצות יכולות להשתמש בתשתית ולהכניס יכולות נוספות ללא תלות בקבוצות אחרות ובפרט, יכולת ניהול פרוייקטים מודולרית, כמו בחירת גרסה מסוימת.

כאמור בחלק התוכנתי, בקוד עצמו יש תיעוד מפורט של כל מחלקה וכל פונקציה ומה מהות, בנוסף לכך בחלק המאסטר ישנו גם תיעוד של כל הפעולות שמתבצעות מאחורי הקלעים(אף על פי שסביר להניח שלא ישנו את היסודות של התשתית שמימשנו), ישנם שני חלקים לתוכנה, תוכנה הקושחה שנמצאת בתיקיית main, ותוכנת הMaster(הapi פייתון), שנמצא בתוך CLI. ישנו בתוך הגיטהאב, דמו עבור שני מצב הריצה מול המאסטר, למצב ריצה STANDALONE, יש בתוכנה לדרוש לרוץ במצב זה או בהינתן ולא מצליחים להכנס למצב SLAVE, הבקר ירוץ במצב STANDALONE. הוספנו כלים כמו ThermalVisualizer שמטרתם להמחיש כיצד ניתן להיעזר בתשתית שעשינו ליצירת כלים נוספים בצד המאסטר.

בנוסף לכך עשינו דמו המראה את כל היכולות באמצעות STANDALONE, וגם דמו נקודתי להוצאת מידע מסנסורים(שככל הנראה יהיה עיקר השימוש בתשתית). בנוסף לכך יש תיעוד שעשינו, אשר מסביר בפרוט-פרוט על הסיבה לפרוייקט, דרך הפיתרון - כלומר על החומרה של הפרויקט, כיצד התוכנה של הפרויקט תמומש(תכנון), כיצד איך היא מומשה בפועל ומכונות מצבים של חלקים עיקריים בתוכנה, ותכנון המחלקות הראשי של הפרוייקט.

בנוסף לכך יש מדריכים שצירפנו לתיעוד שנועדו להקל על המשתמש, כמו כיצד להתקין את התוכנות הנדרשות, כיצד לעבוד מול התשתית שביצענו, כיצד ניתן יהיה להרחיב את יכולותיה והתוצאות בדיקות שעשינו המוכיחות את יכולות ורמת הביצועים של הסנסורים שנבחרו לפרויקט זה. בנוסף לזה צירפנו קישורים למדריכים מומלצים לפייתון + ארדואינו(במידת הצורך), והסבר כללי של כיצד לעבוד עם Arduino, ואיך לעבוד עם הapi, שפיתחנו עבור תשתית המאסטר(של הפייתון).

הוספנו מדריך כמו הוספת סנסור חדש או כיצד להוסיף פקודה חדשה שיהיה ניתן לשלוח לבקר וכיצד גם בקושחה לטפל בפקודה החדשה שנשלחה והסבר כיצד עובד שליחה וקבלה של מידע לבקר שמתפקד בתור הslave . כאמור המטרה של התיעוד להיות כמה שיותר שקוף ומודולרי למשתמש העתידי.

כאמור מצורף גם הdesign החומרתי בתוך הגיט, במידה ויהיה צורך להרחבה או הסרת סנסורים נוספים – ובכך ימנע עיצוב ותכנון מחדש, של החלק החומרתי. בנוסף לכך הוספנו תייקיה לpi לגיט שבמידת הצורך אם ירצו, יוכלו להוסיף את כל המדריכים ההתקנות והדברים הרלוונטיים אליהם בתיקיה זו, ובכך לאגד תיעוד זה ואולי בהמשך תיעודיים עתידים על בקרים נוספים שיעזרו בהם.(כמובן שיהיה ניתן לבנות תיקיות חדשות).

להלן הקישור:

* תיעוד- <https://github.com/YonatanAmir1996/M5StackTelemetry/blob/main/ProjectDocuments/FinalProject.docx>
* הפרויקט - <https://github.com/YonatanAmir1996/M5StackTelemetry>

1. “M5Stack CoreS3 ESP32S3 loT Development Kit Specification https://shop.m5stack.com/products/m5stack-cores3-esp32s3-lotdevelopment-kit
2. Bosch IMU specification: <https://www.bosch-sensortec.com/products/motion-sensors/imus/bmi270/>
3. PaHub specification - <https://shop.m5stack.com/products/i2c-hub-1-to-6-expansion-unit-pca9548apw>
4. PbHub specification - <https://shop.m5stack.com/products/i-o-hub-1-to-6-expansion-unit-stm32f0>
5. SparkFun Qwiic ToF Imager - VL53L5CX <https://www.sparkfun.com/products/18642>
6. SparkFun GRID-EYE Amg8833 - <https://www.sparkfun.com/products/14607>
7. FSR402 Data sheet <https://www.trossenrobotics.com/productdocs/2010-10-26-DataSheet-FSR402-Layout2.pdf>
8. Vibration motor unit - <https://shop.m5stack.com/products/vibration-motor-unit>