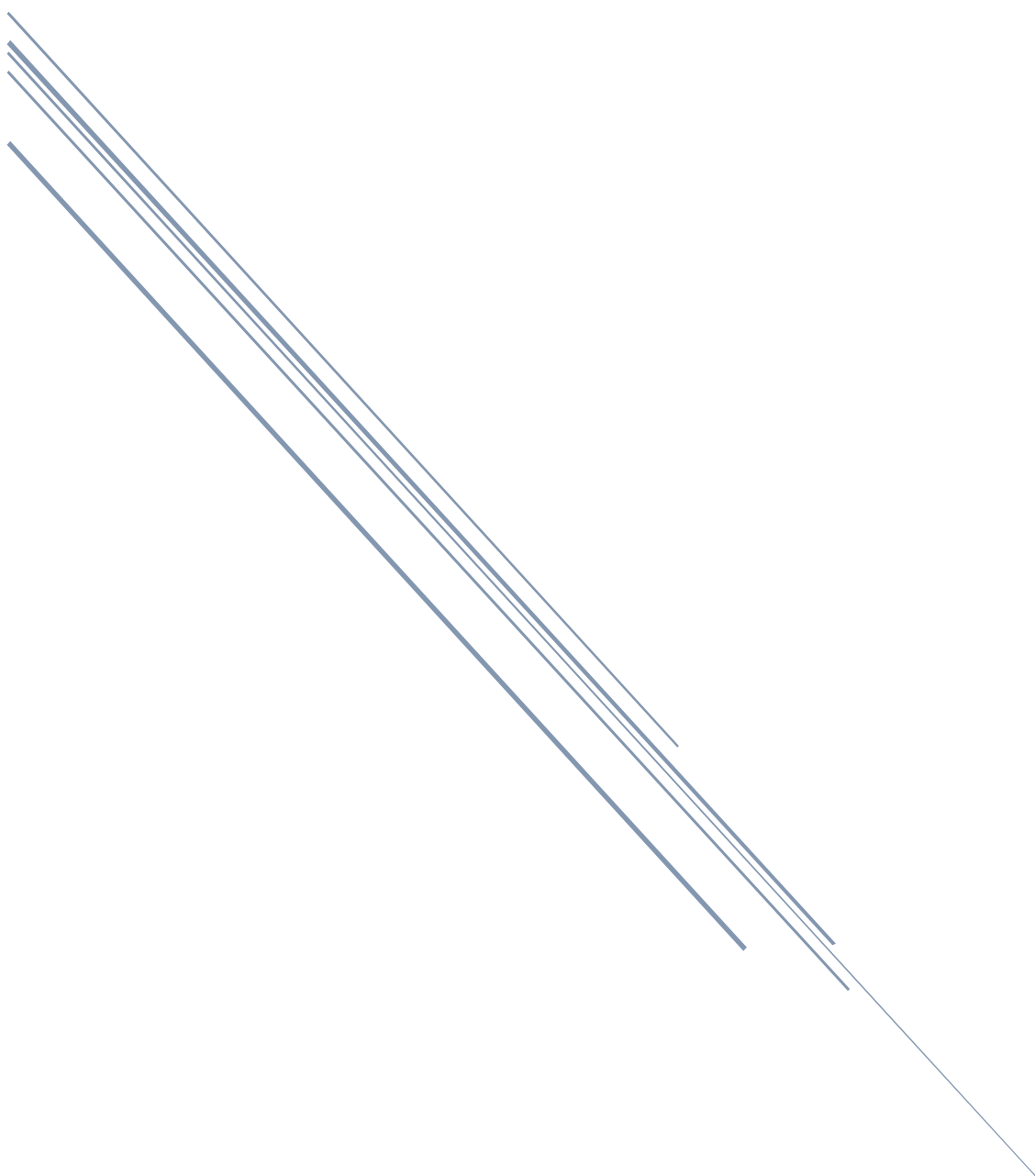


פרויקט גמר י"ב

רובוט משדר תמונה ונתונים לאחר בטכנולוגית IOT



דרכא בית ירח
אור-ים ביבי רובי, עידן קורז'ק

4	הצעת נושא
5	תרשים מלבנים סופי
8	ורכיבי פרויקט וחומרה
8	דוחף זרם L293d –
9	עיקרון הפעולה
9	מבנה פנימי של הרכיב
10	מנוע DC - Direct Current Motor
12	PWM
14	חיישן טמפרטורה – 18b20 לא נכנס בדגם הסופי
16	בקר – ESP32
18	עקרון פעולה
18	מבנה פנימי
20	פרוטוקולים- ESP32
21	מייצב מתח של ה-ESP (Voltage Regulator)
22	ESP32 - CAM
22	עקרון פעולה
23	Altera
23	FPGA-Field-Programmable Gate Array
24	עקרון פעולה
24	מבנה פנימי
24	תקן תקשורת RS232
27	LM75 – Digital Temperature Sensor
30	Firebase – Google
30	דוחף זרם – uln2803
31	מבנה פנימי
32	נורת להט (הוחלף בלד של ESP32CAM)
33	מנוע MG90S – Servo
35	מעגלים חשמליים
35	DC Motor to L293D
36	ALTERA to ESP – TX to RX
37	ESP32 – CAM to Firebase to Java App
38	LM75 TEMP SENSOR to ALTERA And ESP32
39	Servo motor to Altera
40	Uln to laser to altera
41	מייצב מתח של ה-ESP
42	מעגל שלם של כל הפרויקט
48	תוכניות VHDL

74	PIN PLANNER
74	שפת C
76	קוד בקר ה- ESP32
79	קוד ESP32 – CAM
89	שפת תוויות XML
90	שפת Python
90	קוד ESP32 – CAM עיבוד תמונה של
93	האפליקציה
94	קוד - Esp32CameraFragment
118	קוד - MainActivity
120	קוד - fragment_camera xml
123	קוד - activity_main xml
123	קוד - Settings gradle
123	קוד - colors xml
123	קוד - strings xml
124	קוד - styles xml
124	קוד - arrays xml
129	תיעוד תקלות ותיקון
131	אבולוציית הפרויקט
134	ביבליוגרפיה

סמל מוסד - 261065

עבודת גמר

פרויקט בהיקף של 5 יחידות

הנדסת חשמל ואלקטרוניקה

בהתמחות: מחשוב ובקרה.

הנושא: רובוט משדר תמונה ונתונים לאחר בטכנולוגית IOT.

מנחה הפרויקט: חברבר אייל.

מגישים:

שם	תעודת זהות
אור-ים ביבי רובי	328412261
עידן קורזיק	21567059

תאריך הגשה: תשפ"ג

הצעת נושא

שם הנושא :

רובוט משדר תמונה ונתונים לאחור בטכנולוגיית IOT.

תיאור הפרויקט ואופן פעולתו :

רכב נשלט מרחוק ע"י ממשק שאנו מפתחים באנדרואיד סטודיו בשפת JAVA בשימוש בתקשורת
Firebase המבוססת על Wi-Fi.

הרכב מצויד עם מצלמה (ESP CAM 32) המשדרת מידע לאחור לאפליקציה לצורך סריקת
השטח שהמשתמש בוחר, בנוסף על כך יש חיישן טמפרטורה כדי לבדוק את תנאי השטח.

יעוד \ צורך ובעיות שהוא פותר :

מטרות הרכב להחליף כוח אדם ולהיכנס למקומות מסורבלים ולא נוחים לבני אדם, לאסוף מידע
מהשטח ולהחזירו למשתמש לצורך בקרה.

שפות תכנות :

VHDL לתכנות הרכיב בר תכנות של חברת אלטרה.

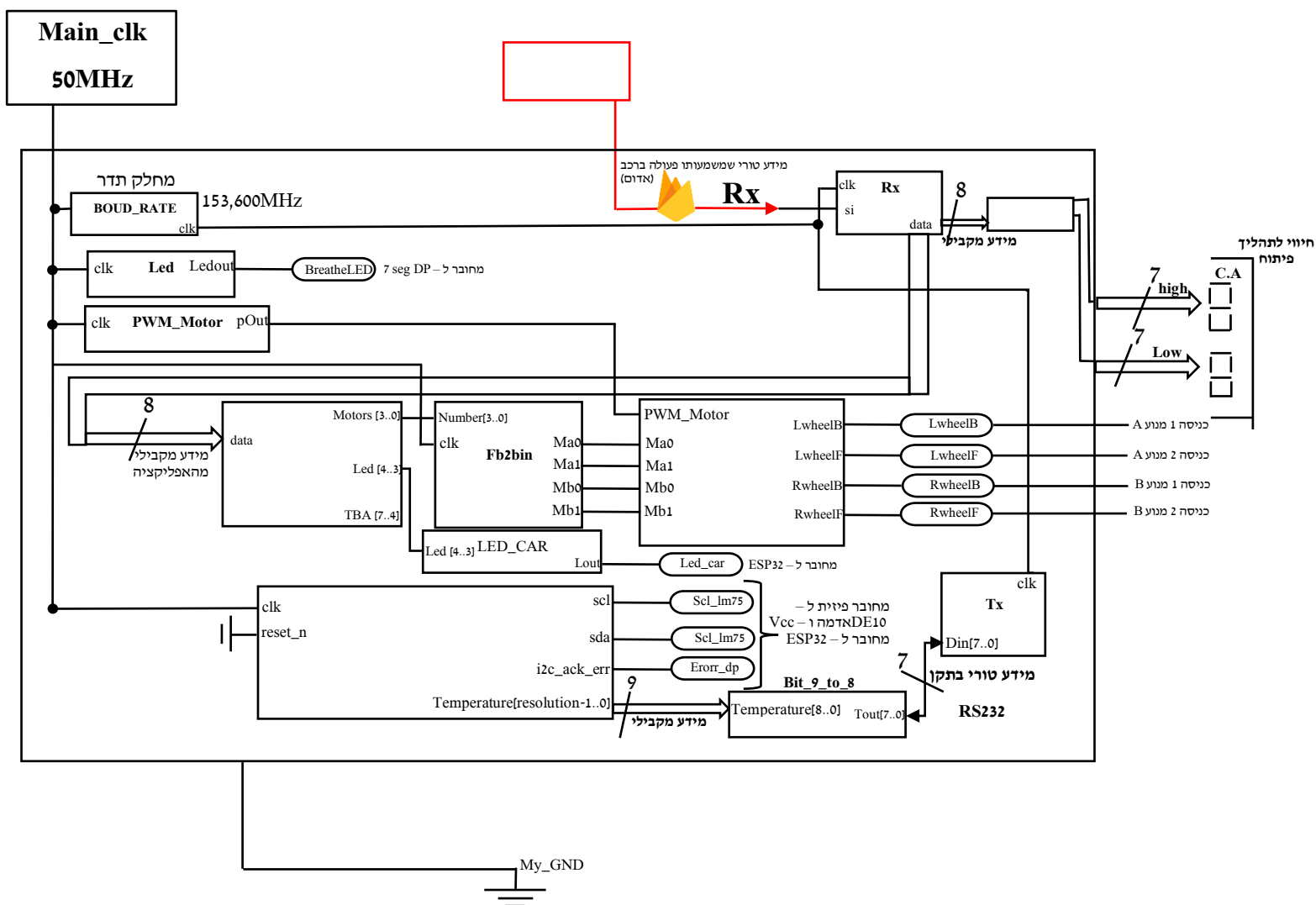
C++ לתכנות רכיבי ה esp המשמשים להעברת מידע למשתמש.

JAVA לתכנות הממשק של האפליקציה.

רשימת רכיבים :

מנועי DC	דוחף זרם למנועים l293d	ESPCAM מצלמה אלחוטית
חיישן טמפרטורה 18b20	ESP32 בקר להעברת מידע ברשת	FPGA MAX10 ALTERA

תרשים מלבנים סופי



BOUD_RATE – מחלק תדר ל –153,600MHz.

Led – מוודא שהמערכת עובדת בכך שנדלק כל שניה.

PWM_Motor – יוצר תדר של 10KHz ב D.C של 80% בשביל מנועי DC.

Data_Splitter – מחלק את הסיביות של הקולט (RX) לפעולות שונות (מנועים, לד).

Fb2bin – קולט את המידע מהFIREBASE ומעביר אותו לבינארי אינטגר.

Motor_Ctrl – מקבל את הסיביות מה - Fb2bin ומחלק אותם לכניסות של המנועים (קדימה אחורה) ועובד לפי תדר ה - PWM_Motor.

Rx – מקבל שעון מה - BOUD_RATE ומידע בגודל של 8 סיביות מהמשדר (TX) של ה - ESP32.

Tx – מקבל מידע מקבילי בגודל 8 סיביות ומשדר אותו בתקן RS232 בטורי, מקבל את אותו השעון כמו הקולט לסנכרון.

Splitter – מחלק את הסיביות מידע מקבילי ל3.0 ו 7.4 גבוה בשביל מפענח תצוגה.

Pmod_temp_sensor_tcn75a – תוכנית שעובדת על חיישן טמפרטורה LM75 ומוציאה את המידע הנחוץ לחיישן (scl,sda).

Bit_9_to_8 – מעביר את המידע של החיישן טמפרטורה מ - 9 סיביות ל - 8.



ורכיבי פרויקט וחומרה

L293d – דוחף זרם

דוחף זרם שמספק הנעה של זרמים דו – כיוונים עד ל – 600mA במתחים מ – 4.5 ל – 36 וולט, רכיב זה נועד להניע עומסים אינדוקטיביים כגון ממסרים, סולנואידים, מנוע סטפר.

אנו צריכים רכיב זה על מנת לספק זרם למנועי גלגלי הרכב.

לרכיב יש 16 רגלים:

רגל 1 – רגל זו מאפשרת החלפה על כניסת המידע הרצויה ברגלים 2,7

רגל 2 – כניסת מספר 1

רגל 3 – יציאה מספר 1 (מחוברת ישירות ליציאה אחת של מנוע 1)

רגל 4 – אדמה (GND)

רגל 5 – אדמה (GND)

רגל 6 – יציאה מספר 2 (מחוברת ליציאה שניה של מנוע 1)

רגל 7 – כניסה מספר 2, שולטת ישירות על יציאה מספר 2 (נשלט על ידי מעגליים דיגיטליים)

רגל 8 – $V_{cc2}(V_s)$ מתח להנעת המנוע (4.5 – 36 וולט)

רגל 9 – רגל זו מאפשרת החלפה על כניסת המידע הרצויה ברגלים 10,15

רגל 10 – כניסה מספר 3 שולטת ישירות על יציאה מספר 3 (נשלט על ידי מעגליים דיגיטליים)

רגל 11 – יציאה מספר 3 (מחוברת ישירות ליציאה אחת של מנוע 2)

רגל 12 – אדמה (GND)

רגל 13 – אדמה (GND)

רגל 14 – יציאה מספר 4 (מחוברת ישירות ליציאה שניה של מנוע 2)

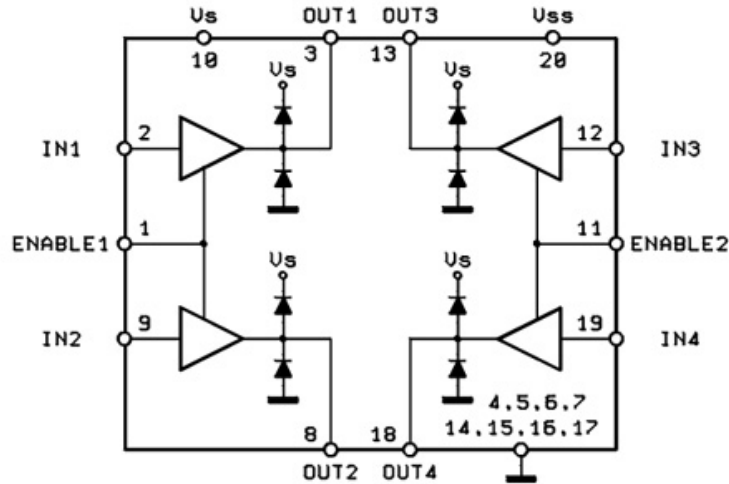
רגל 15 – כניסה מספר 4 שולטת ישירות על יציאה מספר 4 (נשלט על ידי מעגליים דיגיטליים)

רגל 16 – $V_{cc2}(V_{ss})$ מחוברת ל – +5 וולט כדי לאפשר פונקציית IC (מיקרוציפ)



עיקרון הפעולה

ה-IC L293D מקבל אותות מהמיקרו-מעבד ומשדר את האות היחסי למנועים. יש לו שני פני מתח (8,16), שאחד מהם משמש למשיכת זרם לעבודה של ה-L293D והשני משמש להפעלת מתח על המנועים.



מבנה פנימי של הרכיב

L293D בנוי משני גשרי H המאפשרים לסובב שני מנועי DC בנפרד.

גשר H - גשר H הוא מעגל אלקטרוני שהופך את הקוטביות של המתח המופעל על העומס. מעגלים אלה נמצאים בשימוש נפוץ ברובוטיקה וביישומים אחרים להנעת מנועי DC בכיוון קדימה או אחורה. השם בא מהעיצוב הנפוץ בו האלמנטים הגמישים מסודרים כמו ענפי האות 'H' והסורגים מחוברים כמו צלב.

טרנזיסטור דארלינגטון - דארלינגטון הוא מעגל המורכב מטרנזיסטורים דו-קוטביים, הפלט של טרנזיסטור אחד מחובר לבסיס הטרנזיסטור השני כך שהזרם המוגבר על ידי הטרנזיסטור הראשון מוגבר על ידי הטרנזיסטור השני, הם מחוברים. לתצורה זו יש רווח זרם גבוה יותר מכל טרנזיסטור הוא פועל כמו טרנזיסטור ולעתים קרובות הוא ארוז כיחידה אחת.

זרמים ומתחים - הרכיב יכול לספק עד ל-600mA במתחים בטווח 4.5 – 36 וולט, ההספק המירבי הוא 21.6 וואט.

תדר - 5KHz שמיוצר על ידי מיקרוציפ IC אם עוברים את מגבלת הזרם של ה-600mA.

חוק לנץ - חוק לנץ הוא חוק פיזיקלי בתחום המגנטיות, לקביעת כיוון הכא"מ (כמות אנרגיה ליחידת מטען, נקרא גם "כוח אלקטרו-מניע") והזרם החשמלי המושרים על ידי שינוי בשטף המגנטי. השדה המגנטי המושרה על ידי זרם "שואף" לבטל את השינוי בשטף המגנטי היוצר אותו. למעשה, חוק לנץ הוא הסימן השלילי בחוק פאראדיי, אשר קובע את כיוון הכא"מ המושרה:

חוק זה נובע מחוק שימור האנרגיה: אילו הכא"מ המושרה היה גורם להגדלת השינוי בשטף המגנטי, אז השטף המגנטי היה גדל עוד יותר (בגלל הגידול בזרם), ויוצר בכך גידול נוסף בזרם, וכן הלאה עד אינסוף, תוך ביצוע עבודה אינסופית. כאשר השטף המגנטי בכריכה גדל נוצר בכריכה שדה מגנטי ההפוך בכיוונו לשדה בו נמצאת הכריכה. כאשר השטף המגנטי בכריכה קטן נוצר בכריכה שדה מגנטי השווה בכיוונו לשדה בו נמצאת הכריכה.



מנוע DC - Direct Current Motor

מנועי DC מופעלים על ידי זרם ומתח ישרים, שליטתם מתבצעת על ידי שינוי זרם ומתח האספקה אליהם. בדרך כלל משתמשים במנועים מהסוג הזה במערכות בעלות אורך חיים קצר או עם כמות הפעלה קטנה הוא עובד בכך שהוא מנוע הממיר את הזרם הישר לעבודה המכנית. זה עובד על העיקרון של חוק לורנץ, הקובע כי "המוליך נושא הזרם המוצב בשדה מגנטי וחשמלי חווה כוח". והכוח הזה הוא כוח לורנץ.

למנוע DC ישנם שתי חיבורים :

חיבור 1 - VCC כניסת המתח

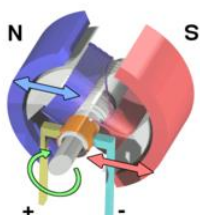
חיבור 2 - GND - אדמה

מנוע זרם ישר (DC) הוא מכונה חשמלית הממירה אנרגיה חשמלית לאנרגיה מכנית. מנוע זה מבוסס על עקרון האלקטרומגנטיות שמאפשר יצירת שדה במגנטי על ידי העברת זרם חשמלי דרך סליל.

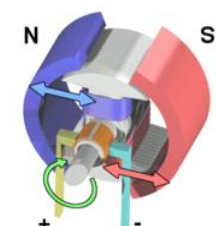
מנוע DC מניע רוטור קבוע בתוך פיר הפלט באמצעות שדה מגנטי שיוצר את הזרם המתקבל.

סטטור - סדרה של סלילים סביב ליבה פרו-מגנטית מוצקה. הסטטור יכול להיות מורכב גם משני מגנטים חזקים. המגנטים ממוקמים מול הרוטור כך שהקטבים שלהם (צפוני ודרום) מנוגדים.

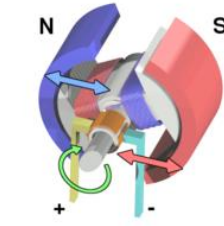
רוטור - ציר שעובר בתוך הסטטור ויש עליו מגנטים, או סלילים. ציר זה מסתובב בחופשיות. כאשר זרם זורם דרך סלילי הרוטור (או הסטטור), הוא יוצר שדה מגנטי סביב עצמו (דרך הליבה). שדה מגנטי זה מפעיל כוח על המוט כשהוא עובר דרכו, וגורם לו להסתובב בהשפעת מומנט סיבובי (כוח סיבוב). בקרת קטע נוכחית משלבת תנועות זוויתיות קטנות עם סיבוב מלא.



בתמונה הזאת הסליל הכחול ממוגנט כקוטב צפוני במקרה הזה, לכן הוא נדחה על ידי המגנט הצפוני. והסליל האדום ממוגנט כקוטב דרומי ונדחה על ידי הקוטב הדרומי וכך הרוטור מסתובב בכיוון השעון.



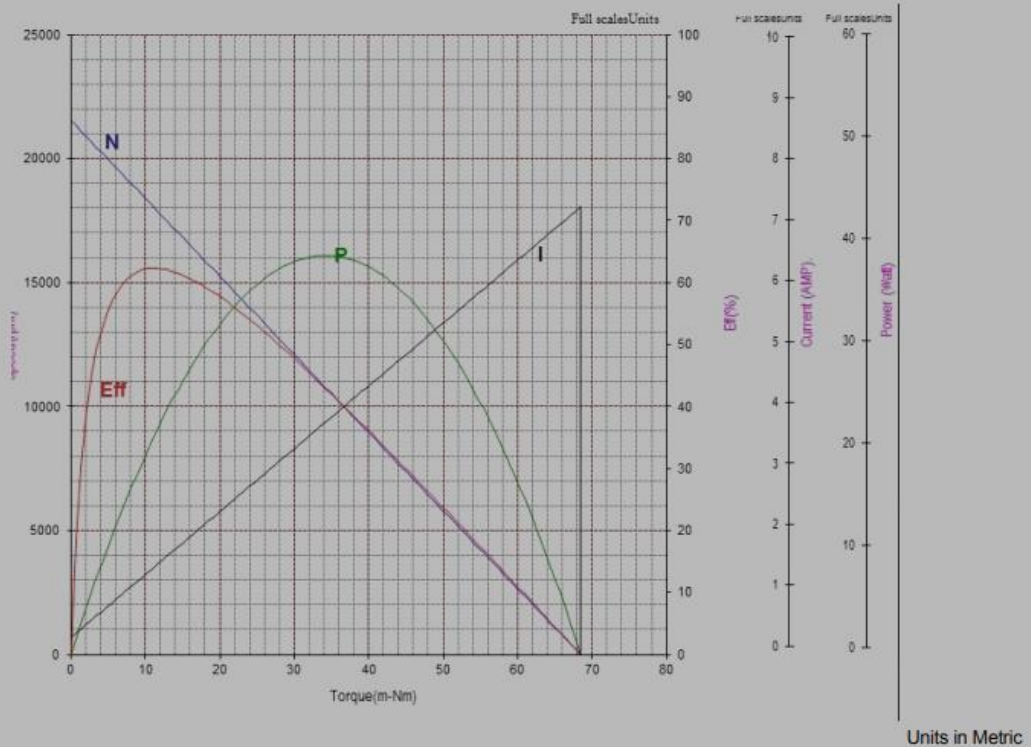
בתמונה הזאת הסלילים נמשכים על ידי כוח מגנטי לצבע הנוגד להם, כלומר הסליל הכחול נדחה מהמגנט הכחול ונמשך למגנט האדום וכך גם לסליל האדום. אפשר גם לראות את חיבור הזרם למנוע.



בתמונה זו הסלילים קרובים לקטבים ההפוכים להם במגנטיות והכוח נחלש, אבל במקום לעצור מתחלפים כיווני אספקת החשמל, והסלילים מחליפים את המגנטיות שלהם.

אופן הפעולה של מנוע DC הוא שבעזרת הזרם בעובר בסלילי הרוטור מכוון כך שהסלילים שמסתובבים מייצרים שדה אלקטרומגנטי עם קוטביות משתנה וכך אותו הקוטב מכוון למגנטים בסטטור. במצב הזה אחד מהמגנטים דוחה את הסליל שקרוב אליו והשני מושך את הסליל הקרוב אליו.

Performance Curves:



עקרון הפעולה של המנוע הינו אסינכרוני כלומר המנוע מסתובב במהירות שלא מסונכרנת לתדר שהוא מקבל. כאשר הוא נשמר בשדה מגנטי, מוליך נושא זרם צובר מומנט ומפתח נטייה לזוז. בקיצור, כאשר שדות חשמליים ושדות מגנטיים מתקשרים, נוצר כוח מכני. זהו העיקרון שעל פיו פועלים מנועי ה-DC

מנוע DC שאנחנו נעבוד איתו עובד עם 5V, צורך 1.42A ומספק 18000 סל"ד. על ככל שיש יותר עומס על המנוע כך ירד הסל"ד כדי להתנהל עם העומס הנוצר.

ציר אנכי – RPM, זה הוא סיבובים לדקה.

מהירות (N) - כחול

נמדד בסיבובים לדקה (סל"ד), קו ישר זה בשיפוע כלפי מטה מראה את הקשר בין מומנט ומהירות על פני כל פס הכוח (ראה קו כחול בדוגמה למעלה). מכיוון שהמהירות והמומנט הם פרופורציונליים בעקיפין זה לזה, קו זה יקטן באופן ליניארי ככל שהמומנט יגדל עד לנקודת העצירה, שבה המהירות תהיה 0 סל"ד. למהירות ולמומנט יש קשר הפוך. המהירות היא הגבוהה ביותר כאשר המנוע מייצר את המומנט הנמוך ביותר, וכאשר המומנט הוא הגבוה ביותר המנוע בקושי מסתובב.

יעילות (η) - אדום

יעילות היא קשר בין הספק המבוא להספק המוצא, נתון באחוזים (%). קו זה הוא פחות או יותר פרבולי עם הקודקוד יותר לכיוון ערכי המומנט הנמוכים יותר (ראה קו ורוד בדוגמה למעלה). זה בדרך כלל צריך מוקדם בטווח המומנט ואז יורד באיטיות ככל שהמנוע מתקרב למומנט העצירה שלו.

שימוש במנוע קרוב לשיא היעילות שלו מבטיח חיי מנוע אופטימליים וצריכת חשמל. עדיף להשתמש במנוע בשיא היעילות שלו או בסמוך לו. ככל שמנוע מתרחק מיעילות מרבית, הביצועים שלו הופכים לפחות אמינים.

מומנט (Torque) - ציר אופקי

בין אם נמדד בק"ג-ס"מ, I_{b-in} או N_m , מומנט הוא כמות העומס שציר המוצא של המנוע או מנוע ההילוכים יכול להתגבר עליו. בעקומת ביצועים של מנוע DC, המומנט מיוצג בדרך כלל על ידי ציר ה-X.

המפגש בין קו המהירות (N) וציר ה-X הוא נקודת מומנט העצירה (T). זה המקום שבו המנוע מייצר את המומנט המרבי שלו ואינו יכול יותר להסתובב. וודאו שהכוח הדרוש מהמנוע קטן בהרבה מיכולת המומנט הכוללת (מומנט עצור), אחרת המנוע לא יעבוד כמתוכנן ויהיה לו סיכון גבוה להינזק.

זרם (I) - שחור

מיוצג על ידי הקו הישר העולה (ראה קו צהבהב בדוגמה לעיל), זה משקף את יציאת הזרם ממצבי חוסר עומס לתנאי עצור. קו זה מראה את הקשר הישיר בין זרם ומומנט. אם המערכת מודעת לכוח זה יהיה השיטה הטובה ביותר להפעיל את המנוע ביעילות שיא. זה מפיק את הביצועים המאוזנים ביותר מהמנוע תוך דרישה לכמות סבירה של זרם.

כוח פלט (P) - ירוק

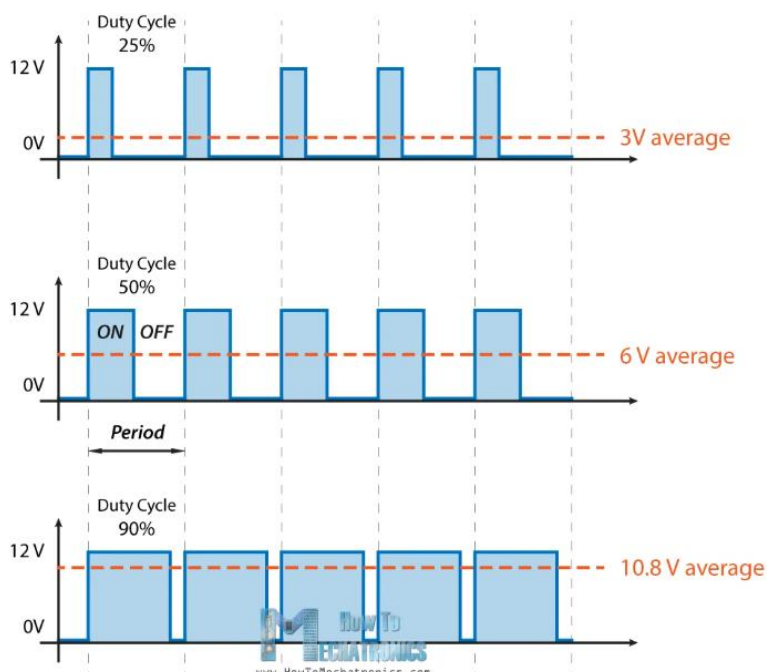
אולי המספר הפשוט ביותר בגרף כולו, Power Output מראה כמה כוח (בוואט) יכול מנוע לספק (ראה קו ירוק בדוגמה למעלה).

אלקטרומגנטיות - סוג של אינטראקציה פיזיקלית המתרחשת בין חלקיקים טעונים במטען חשמלי. הכוח האלקטרומגנטי המיוצר על ידי שדות אלקטרומגנטיים מורכב משדות חשמליים ושדות מגנטיים, ואחראי לקרינה אלקטרומגנטית כמו אור. זהו אחד מארבעת הכוחות הבסיסיים (המכונים לעתים קרובות כוחות) בטבע, יחד עם ההשפעה החזקה, ההשפעה החלשה וכוח המשיכה. באנרגיות גבוהות יותר, הכוח החלש והכוח האלקטרומגנטי משולבים לכוח אלקטרומגנטי אחד.

PWM

PWM או אפנון רוחב דופק היא טכניקה המאפשרת לנו להתאים את הערך הממוצע של המתח שעובר למכשיר האלקטרוני על ידי הפעלה וכיבוי של המתח בקצב מהיר. המתח הממוצע תלוי במחזור העבודה, או משך הזמן שהאות מופעל לעומת משך הזמן שהאות כבוי בפרק זמן בודד.

Pulse Width Modulation



H- Bridge

מצד שני, כדי לשלוט בכיוון הסיבוב, אנחנו רק צריכים להפוך את כיוון זרימת הזרם דרך המנוע, והשיטה הנפוצה ביותר לעשות זאת היא באמצעות H-Bridge.

מעגל H-Bridge מכיל ארבעה רכיבי מיתוג, טרנזיסטורים או MOSFET, כאשר המנוע במרכז יוצר תצורה דמוית H. על ידי הפעלת שני מתגים מסוימים בו זמנית נוכל לשנות את כיוון זרימת הזרם, ובכך לשנות את כיוון הסיבוב של המנוע.

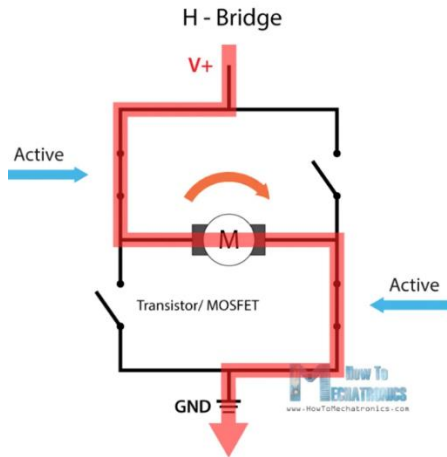
אז אם נשלב את שתי השיטות הללו, ה-PWM וה-H-Bridge, נוכל לקבל שליטה מלאה על מנוע ה-DC.

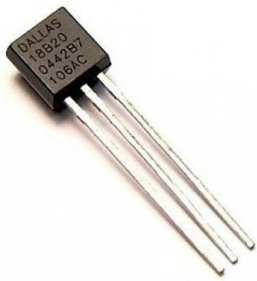
חוק לורנץ

כוח לורנץ, הכוח המופעל על חלקיק טעון q שנע במהירות v דרך שדה חשמלי E ושדה מגנטי B . כל הכוח האלקטרומגנטי F על החלקיק הטעון נקרא כוח לורנץ.

$$\mathbf{F} = q\mathbf{E} + q\mathbf{v} \times \mathbf{B}$$

זה אומר שהכוח האלקטרומגנטי על מטען q הוא שילוב של כוח בכיוון השדה החשמלי E פרופורציונלי לגודל השדה וכמות המטען, וכוח בזווית ישרה לשדה המגנטי B וה-מהירות v של המטען, פרופורציונלית לגודל השדה, המטען והמהירות.





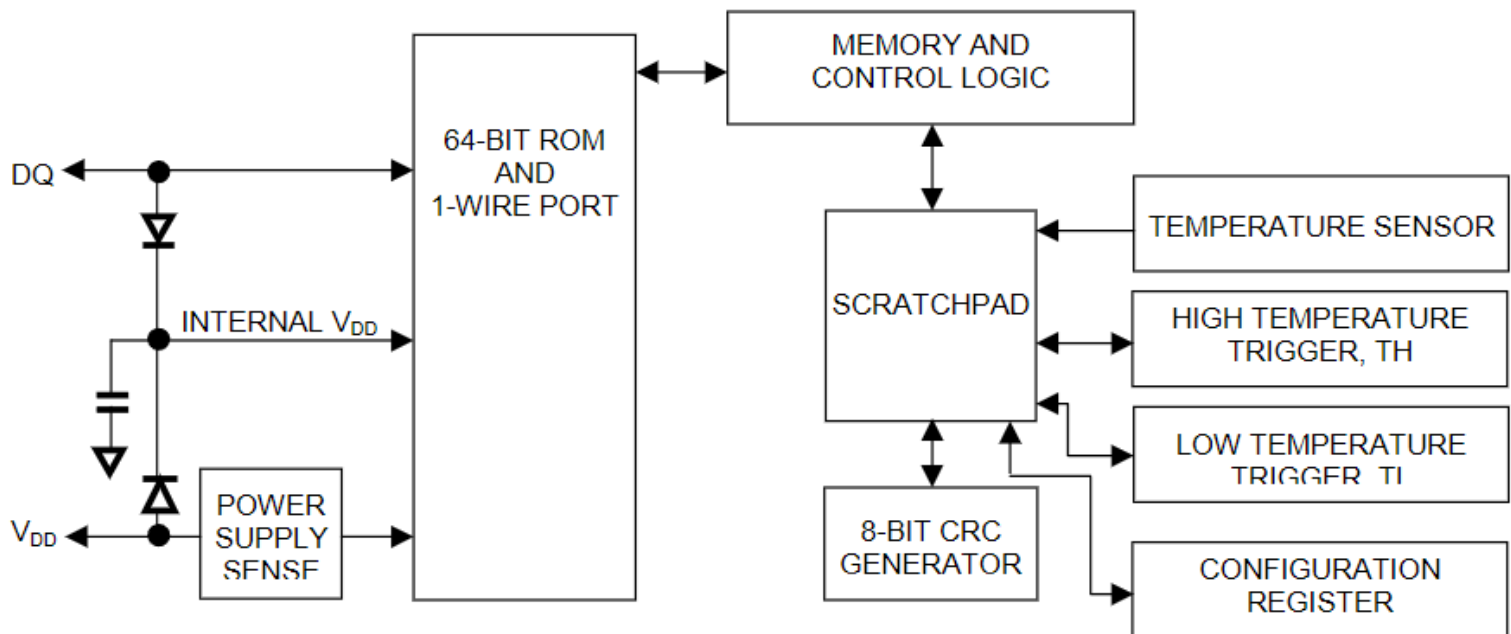
חיישן טמפרטורה – 18b20 לא נכנס בדגם הסופי

מד החום הדיגיטלי 18b20 מספק קריאות טמפרטורה של 9 עד 12 סיביות המציאות את הטמפרטורה של המכשיר. המידע נשלח מהחיישן באמצעות ממשק Wire-1, כך שצריך לחבר רק חוט אחד (ואדמה) ממעבד מרכזי ל-18b20. ניתן להפיק כוח לקריאה, כתיבה וביצוע המרות טמפרטורה מקו הנתונים עצמו ללא צורך במקור מתח חיצוני.

לחיישן יש 3 רגלים:

- רגל 1 – מקבלת מתח בין 3 – 5.5 וולט.
- רגל 2 – העברת מידע בתקשורת Wire – 1.
- רגל 3 – אדמה (GND).

התמונה מציגה דיאגרמת בלוקים של החיישן, ה-ROM של 64 סיביות מאחסן את הקוד הסידורי הייחודי של המכשיר. הזיכרון מכיל את 2 אוגרי הטמפרטורה שמאחסנים בתוכם את הפלט הדיגיטלי מחיישן הטמפרטורה.



לוח ה-scratchpad מספק גישה לאוגרי אזעקה עליונים ותחתונים של בית אחד (TH ו-TL) ואוגר התצורה של בית אחד. אוגר התצורה מאפשר למשתמש להגדיר את הרזולוציה של המרת טמפרטורה לדיגיטל בין 9 ל-12 סיביות.

האוגרים (TH, TL) והקונפיג אינם נדיפים בעלי שבב EEPROM, כך שהם שומרים נתונים גם כשהחיישן מושבת.

פרוטוקול – Wire-1

הבסיס של טכנולוגיית Wire-1 הוא תקשורת טורית המשתמש בקו נתונים בודד בתוספת הפניה לקרקע לתקשורת.

מאסטר Wire-1 יוזם ושולט בתקשורת עם התקני ה-Wire-1 SLAVE אחד או יותר באפיק Wire-1 (BUS).

לכל התקן עבד Wire-1 יש מספר זיהוי ייחודי, בלתי ניתן לשינוי, מתוכנת במפעל, 64 סיביות (ID), המשמש ככתובת התקן באפיק Wire-1 (BUS).

הקוד המשפחתי של 8 סיביות, קבוצת משנה של מזהה 64 סיביות, מזהה את סוג ההתקן ואת הפונקציונליות. בדרך כלל, התקני עבד Wire-1 פועלים על פני ארבעת טווחי המתח הבאים:

- V1.71 (מינימום) עד V1.89 (מקסימום)
- V1.71 (מינימום) עד V3.63 (מקסימום)
- V2.97 (מינימום) עד V3.63 (מקסימום)
- V2.8 (מינימום) עד V5.25 (מקסימום)

לרוב התקני Wire-1 אין פינים עבור ספק כוח. הם לוקחים את האנרגיה שלהם מהאפיק (BUS) Wire-1 (ספק כוח טפילי).

אפיק - BUS

דרכו אפשר גם להוציא נתונים וגם להכניס נתונים. יש לו שימוש בתכנות עם זיכרון וכו'.

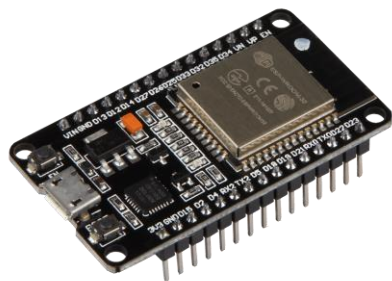
UART- Universal Asynchronous Receiver-Transmitter

UART הוא אחד מפרוטוקולי התקשורת בין התקן למכשיר הנפוצים ביותר.

כאשר מוגדר כהלכה, UART יכול לעבוד עם סוגים רבים ושונים של פרוטוקולים טוריים הכוללים שידור וקבלה של נתונים טוריים.

בתקשורת טורית, נתונים מועברים סיבית אחר סיבית באמצעות קו בודד או חוט. בתקשורת דו-כיוונית, אנו משתמשים בשני חוטים להעברת נתונים טורית מוצלחת.

בהתאם ליישום ולדרישות המערכת, תקשורת טורית זקוקה לפחות מעגלים וחוטים, מה שמפחית את עלות ההטמעה.



בקר – ESP32

ESP32 הוא שם השבב שפותח על ידי Espressif Systems. שבב זה משתמש ב-Wi-Fi ובדגמים מסוימים dual mode Bluetooth למכשירים משובצים.

לשבב ה-ESP32 יש מעבד Tensilica Xtensa LX6 בווריאציות כפולות ליבה ויחידות, עם קצב שעון של מעל 240 מגה-הרץ. ישנם כעת מספר דגמי שבבים שונים זמינים, כולל:

- ESP32-D0WDQ6 (ו-ESP32D0WD)
- ESP32-D2WD
- ESP32-S0WD

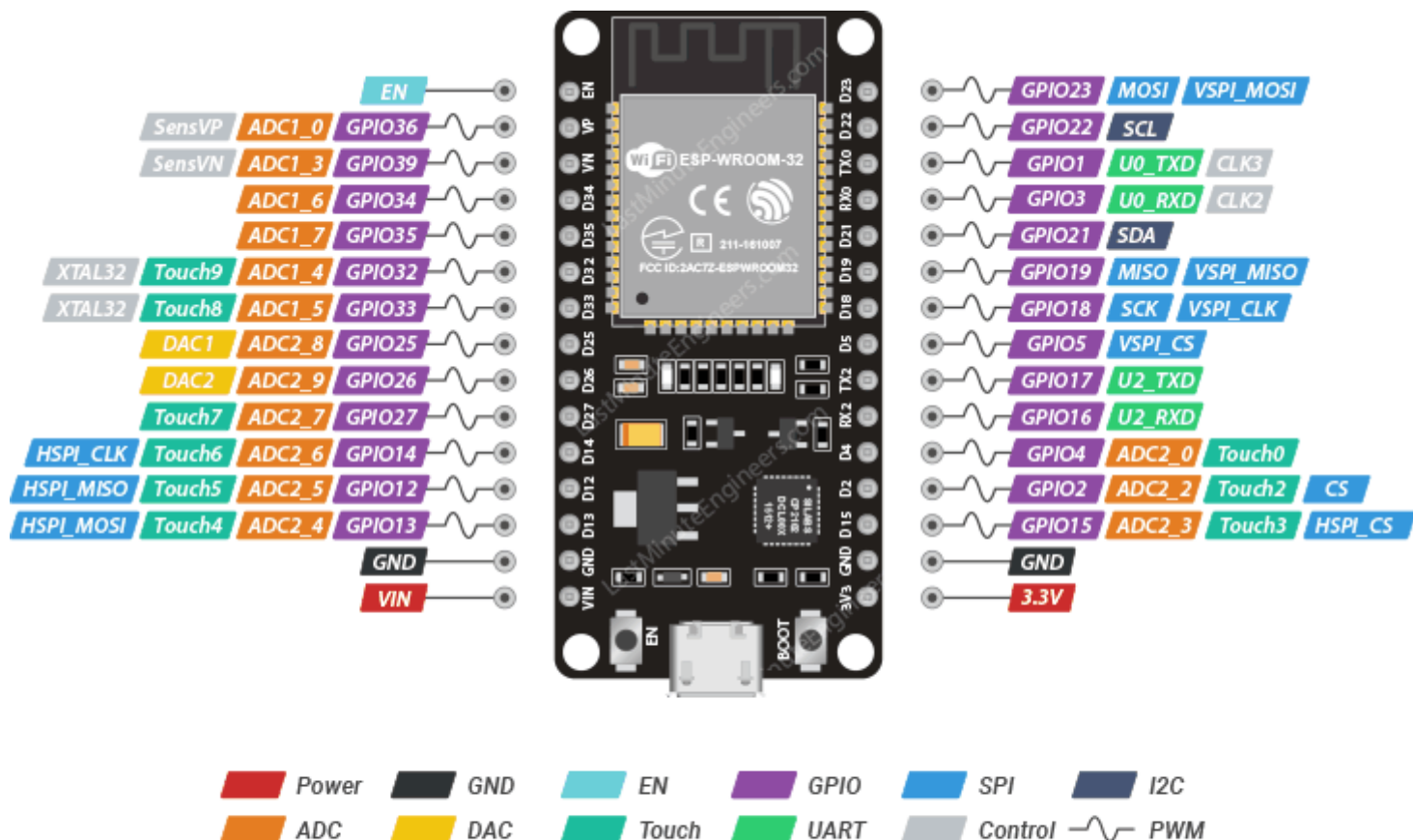
והמערכת בחבילה ESP32-PICO-D4 – (SiP)

דגמים זמינים עם Wi-Fi ו-Bluetooth משולבת, או רק Wi-Fi.

ה-ESP32 מתוכנן לרוב עבור מכשירים ניידים, טכנולוגיה לבישה ויישומי IoT - כגון Nabto. יתרה מכך, עם הצגת מערכת ההפעלה ESP32 IoT Starter Kit, Mongoose, ה-ESP32 צבר מוניטין של השבב או המודול האולטימטיבי עבור חובבים ומפתחי IoT.

אמנם המוניטין הזה אינו מוצדק, אבל המכשיר הזול יכול לשמש גם במספר מערכות ייצור שונות, והיכולות והמשאבים שלו גדלו בצורה מרשימה בארבע השנים האחרונות.

תפקידי הדקים



^ - בטוח לשימוש

@ - ההתנהגות שלהם יכולה להיות בלתי צפויה, בעיקר במהלך האתחול. אל תשתמש בהם אלא אם כן אתה בהחלט צריך.

X – לא מומלץ

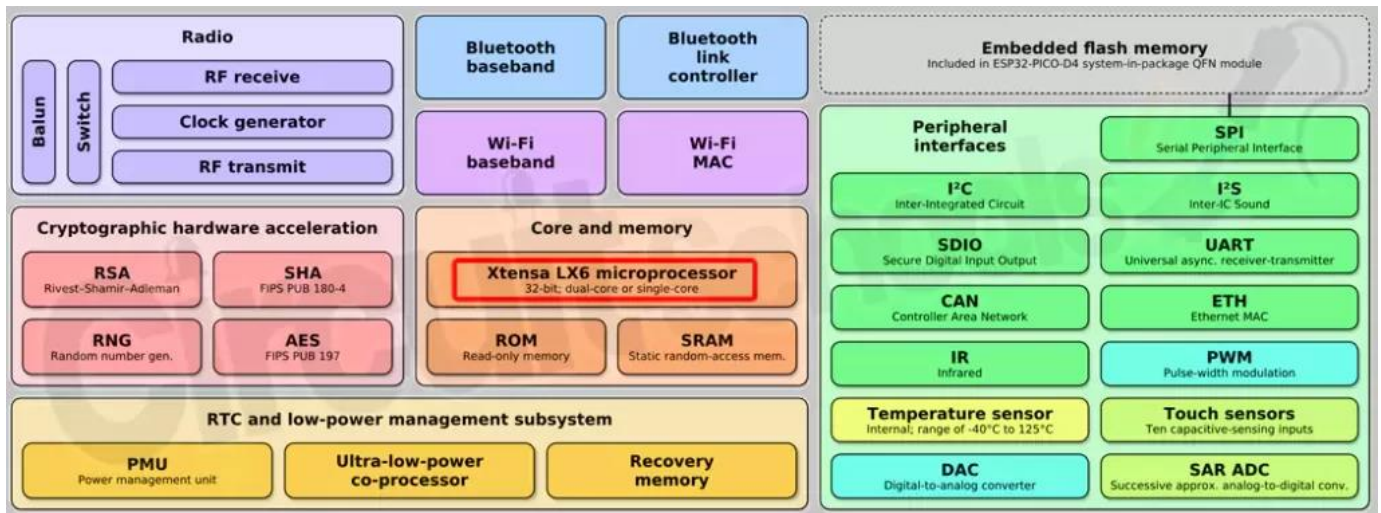
שם הרגל	מספר הרגל – GPIO	בטוח לשימוש	תפקיד
EN	-	-	Enable- pin מופעל ב - HIGH
VIN	-	-	ניתן להשתמש בפין VIN כדי לספק בחשמל ישירות את ה-ESP32
3.3V	-	-	פין 3.3V הוא הפלט של וסת מתח על הלוח. יכול לשמש לאספקת חשמל לרכיבים חיצוניים
D0	0	@	חייב להיות HIGH במהלך האתחול ו- LOW לתכנות
TX0	1	X	משמש להבהוב וניפוי באגים
D2	2	@	חייב להיות LOW במהלך האתחול וגם מחובר לנורת ה-LED המובנית
RX0	3	X	משמש להבהוב וניפוי באגים
D4	4	^	פין לשימוש חופשי
D5	5	@	חייב להיות HIGH במהלך האתחול
D6	6	X	מחובר לזיכרון פלאש
D7	7	X	מחובר לזיכרון פלאש
D8	8	X	מחובר לזיכרון פלאש
D9	9	X	מחובר לזיכרון פלאש
D10	10	X	מחובר לזיכרון פלאש
D11	11	X	מחובר לזיכרון פלאש
D12	12	@	חייב להיות LOW במהלך האתחול
D13	13	^	פין לשימוש חופשי
D14	14	^	פין לשימוש חופשי
D15	15	@	חייב להיות HIGH במהלך האתחול, מונע startup log אם נמשך LOW
RX2	16	^	פין לשימוש חופשי
TX2	17	^	פין לשימוש חופשי
D18	18	^	פין לשימוש חופשי
D19	19	^	פין לשימוש חופשי
D20	20	^	פין לשימוש חופשי
D21	21	^	פין לשימוש חופשי
D22	22	^	פין לשימוש חופשי
D23	23	^	פין לשימוש חופשי
D24	24	^	פין לשימוש חופשי
D25	25	^	פין לשימוש חופשי
D26	26	^	פין לשימוש חופשי
D27	27	^	פין לשימוש חופשי
D28	28	^	פין לשימוש חופשי
D29	29	^	פין לשימוש חופשי
D30	30	^	פין לשימוש חופשי
D31	31	^	פין לשימוש חופשי
D32	32	^	פין לשימוש חופשי
D33	33	^	פין לשימוש חופשי
D34	34	@	קלט בלבד, לא ניתן להגדיר כפלט
D35	35	@	קלט בלבד, לא ניתן להגדיר כפלט
Sensor_VP	36	@	קלט בלבד, לא ניתן להגדיר כפלט
GND	38		אדמה
Sensor_VN	39	@	קלט בלבד, לא ניתן להגדיר כפלט

עקרון פעולה

עקרון הפעולה של ה-ESP32 הוא שאפשר להשתמש בו למערכות בעלות מספר רב של פנים, רכיב בר תכנות (חומר ביד היוצר), ניתן להתחבר לאינטרנט. לבקר יש מעבד חזק בעל תקשורת טורית.

ה-ESP32 צורך כמות נמוכה של חשמל ויש לו זיכרון פנימי, הוא פועל בצורה מבוקרת ומוגדרת בשביל לספק מינימום אנרגיה שצריך.

מבנה פנימי



רדיו

לשבב ESP32 SoC יש חיבור WiFi, תואם ל-802.11 b/g/n בפס 2.4 GHz, ומגיע למהירויות של עד 150 Mbit/s. הוא כולל גם תקשורת Bluetooth התואמת ל-Bluetooth v4.2 ו-BLE (Low Energy).

בלוק הרדיו קשור קשר הדוק למודול התקשורת האלחוטית. למעשה, זהו זה שבצעם משדר ומקבל את המידע.

כלומר, הוא לוקח את הנתונים הדיגיטליים ממודולי ה-WiFi וה-Bluetooth וממיר אותם לאותות אלקטרומגנטיים שעוברים באוויר כדי לתקשר עם הטלפון הנייד או הנתב שלך.

הוא גם מבצע את הפעולה ההפוכה: מתרגם את הגלים האלקטרומגנטיים שנוצרים על ידי מכשירים אחרים לנתונים דיגיטליים שמודולי ה-WiFi וה-Bluetooth מסוגלים לפרש.

ליבה

ל-ESP32 יש מעבדי Tensilica Xtensa 32-bit LX6 בעלי עוצמה נמוכה כפולה.

כפי שניתן לראות מתמונת בלוק הליבה, יש לו מעבד משותף בעל הספק נמוך במיוחד המשמש לבצוע המרות אנלוגיות-דיגיטליות ופעולות אחרות בזמן שהמכשיר פועל במצב שינה עמוקה עם צריכת חשמל נמוכה. בדרך זו מושגת צריכה נמוכה מאוד על ידי ה-SoC.

חשוב לציין שמעבדים אלה מציעים יתרונות אופייניים גדולים של מעבד אותות דיגיטלי:

- תדר הפעלה: 240 מגה-הרץ (מבצע הוראות פי 15 מהר יותר מלוח Arduino UNO)
- הוא מאפשר לבצע פעולות עם מספרים ממשיים (מספרים עם פסיקים) ביעילות רבה.
- מאפשר לך להכפיל מספרים גדולים באופן מיידי.

זיכרון

ברוב המיקרו-בקרים המבוססים על Arduino, ישנם שלושה סוגים של זיכרונות:

זיכרון תוכנית: לאחסון הסקיצה.

זיכרון SRAM: לאחסון המשתנים המשמשים בקוד.

זיכרון EEPROM: לאחסון משתנים שאינם מאבדים את ערכם גם כשהמכשיר כבוי.

ב-ESP32 זה לא קורה, למעשה ישנם סוגים נוספים של זיכרונות שבדרך כלל מסווגים לתוך פנימי וחיצוני.

הזיכרונות הפנימיים הם אלה שכבר כלולים ב-SoC, והחיצוניים הם אלה שניתן להוסיף כדי להרחיב את הקיבולת של המערכת.

לוחות פיתוח רבים מבוססי ESP32 מוסיפים זיכרון חיצוני למערכת בעלת ביצועים טובים יותר.

ESP32 זיכרונות פנימיים ותפקידיהם:

זיכרון ROM (448 KiB): זיכרון זה הוא לכתיבה בלבד, כלומר, לא ניתן לתכנת אותו מחדש. זה המקום שבו מאוחסנים הקודים המטפלים בערימת ה-Bluetooth, בקרת השכבה הפיזית של ה-Wi-Fi, כמה שגרות למטרות כלליות ומטען האתחול להפעלת הקוד מזיכרון חיצוני.

זיכרון SRAM פנימי (520 KiB): זיכרון זה משמש את המעבד לאחסון נתונים והוראות. היתרון שלו הוא שהרבה יותר קל לגישה למעבד מאשר ל-SRAM החיצוני.

RTC SRAM (16 KiB): זיכרון זה משמש את המעבד המשותף כאשר ההתקן פועל במצב שינה עמוקה.

Efuse (1 קילוביט): 256 סיביות מזיכרון זה משמשות את המערכת עצמה ו-768 הסיביות הנותרות שמורות ליישומים אחרים.

Flash Embedded (Embedded flash): הזיכרון הזה הוא המקום שבו קוד היישום שלנו מאוחסן. כמות הזיכרון משתנה בהתאם לשבב המשמש:

MB 0 (שבבים ESP32-DoWDQ6, ESP32-DoWD, ESP32-S0WD)

MB 2 (שבב ESP32-D2WD)

MB 4 (שבב ESP32-PICO-D4)

עבור ESP32s שאין להם זיכרון מוטבע או פשוט כאשר הזיכרון אינו מספיק עבור היישום שלך, אפשר להוסיף יותר זיכרון חיצוני:

ניתן להוסיף עד MB 16 של זיכרון פלאש חיצוני. כך תוכלו לפתח אפליקציות מורכבות יותר.

הוא תומך גם בעד MB8 של זיכרון SRAM חיצוני.

מאיצי חומרת הצפנה

אחד הגורמים החשובים ביותר בכל מערכת הוא אבטחה. לכן ל-ESP32 יש מאיצי אלגוריתמים המכוונים להצפנה:

- AES (FIPS PUB 197)
- SHA (FIPS PUB 180-4)
- RSA
- ETC

מאיצים אלו מאפשרים להגביר את מהירות הפעולה ולהפחית את מורכבות התוכנה המאפשרים הצפנה ופענוח דינמי. באופן זה המערכת מוגנת מפני התקפות פריצה אפשריות המבקשות להשיג את הקוד המאוחסן.

ערכים חשמליים

- ה-ESP32 יכול לקבל בין 2.2 - 5 וולט
- זרם של 500mA
- תדר מתנד – 40MHz

פרוטוקולים-ESP32

WI-FI

ESP32 מיישם WLAN MAC 802.11 b/g/n/e/i, full TCP/IP ו-Wi-Fi Direct. המשמעות היא שה-ESP32 יכול לדבר עם רוב נתבי ה-WiFi בחוץ כאשר משתמשים במצב תחנה. כמו כן, הוא מסוגל ליצור נקודת גישה עם 802.11 b/g/n/e/i מלא.

ESP32 תומך גם ב-Wi-Fi Direct. Wi-Fi Direct היא אפשרות טובה לחיבור (p2p) peer to peer ללא צורך בנקודת גישה. ה-Wi-Fi Direct קל יותר להגדרה ומהירויות העברת הנתונים טובות בהרבה מ-Bluetooth. זה יכול לשמש פוטנציאל להגדרת פרויקטים מבוססי ESP32 מטלפון/טאבלט התומך ב-WiFi ישיר. למימוש ESP-IDF WiFi יש את התכונות הבאות בפיתוח:

- תשתית מצב BSS Station / מצב P2P / תמיכה במצב softAP
- P2P Group Client, P2P Discovery, P2P Group בעלים, P2P Power Management ו-P2P
- מנהל התקן WPA/WPA2-Enterprise ו-WPS
- תכונות אבטחה נוספות של 802.11 i כגון אימות מוקדם ו-TSN
- אלגוריתם החזרה של קצב הסתגלות מגדיר את קצב השידור וההספק האופטימלי על סמך יחס רעשי אותות בפועל (SNR) ומידע על אובדן מנות מידע (packet loss)
- שידור חוזר ותגובה אוטומטיים ב-MAC כדי למנוע השלכת מנות בסביבת מארח איטית

Bluetooth Low Energy(BLE)

ESP32 לא רק תומך ב-BLE Bluetooth 4.2 העדכני ביותר, הוא תומך גם ב-Bluetooth קלאסי. זה בעצם אומר שהוא יכול לדבר עם טלפונים/טבלאות Bluetooth ישנים וחדשים. זו יכולה להיות אחת התכונות הטובות ביותר, במיוחד אם אתה מעצב מכשיר שצריך לעבוד עם טלפונים/טאבלטים קיימים כמו חדשים בשוק. רדיו ו-ESP32 Bluetooth Baseband תומכים בתכונות הבאות:

- Class-1, Class-2 ו-Class-3 משדרות הספקי מוצא ומעל טווח בקרה דינמי של dB30
- אפנון DQPSK $\pi/4$ ו-8 DPSK
- ביצועים גבוהים ברגישות למקלט NZIF עם טווח דינמי של מעל 98 dB
- פעולה Class-1 ללא PA חיצוני
- SRAM פנימי מאפשר העברת נתונים במהירות מלאה, קול ונתונים מעורבים ופעולת פיקונט מלאה
- לוגיקה לתיקון שגיאות קדימה, בקרת שגיאות כותרת, מתאם קוד גישה, CRC, מודולציה, יצירת זרם סיביות בהצפנה, הלבנה ועיצוב דופק שידור
- AFH ו-ACL, SCO, eSCO
- A-law, μ -law ו-CODEC אודיו דיגיטלי CVSD בממשק PCM
- SBC אודיו CODEC
- ניהול צריכת חשמל עבור יישומים בהספק נמוך
- SMP עם AE של 128 סיביות

ESP – NOW

ESP-NOW הוא מעין פרוטוקול תקשורת Wi-Fi ללא חיבור המוגדר על ידי Espressif. ESP-NOW, נתוני האפליקציה מובלעים במסגרת פעולה ספציפית לספק ולאחר מכן מועברים ממכשיר Wi-Fi אחד לאחר ללא חיבור. CTR עם פרוטוקול CBC-MAC (CCMP) משמש להגנה על מסגרת הפעולה לצורך אבטחה. ESP-NOW נמצא בשימוש נרחב בתאורה חכמה, שליטה מרחוק, חיישן וכו'.

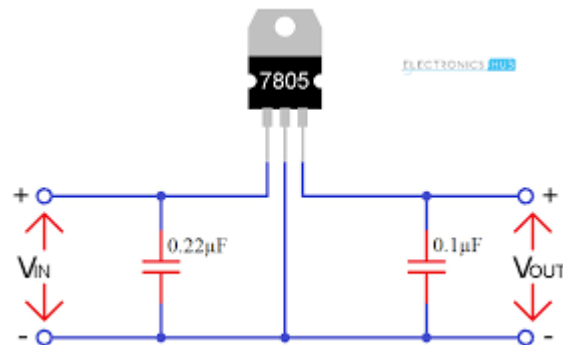
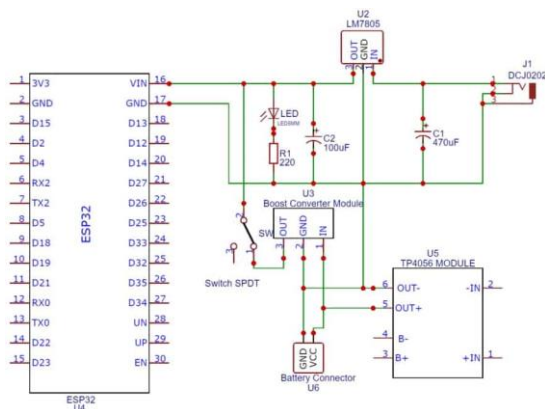
I2C BUS

I2C הוא פרוטוקול תקשורת טורי, סינכרוני, חצי דופלקס המאפשר קיום משותף של מספר מאסטרים ועבדים על אותו BUS. I2C BUS מורכב משני קווים: קו נתונים טורי (SDA) ושעון טורי (SCL). שני הקווים דורשים נגדי משיכה.

עם יתרונות כמו פשטות ועלות ייצור נמוכה, I2C משמש בעיקר לתקשורת של התקנים היקפיים במהירות נמוכה למרחקים קצרים (בתוך רגל אחת).

ל-ESP32 יש 2 בקרי I2C (המכונה גם יציאה), האחראי לטיפול בתקשורת ב-I2C BUS. בקר I2C יחיד יכול לפעול כמאסטר או כעבד.

מייצב מתח של ה-ESP (Voltage Regulator)



מייצבי מתח הם מווסתים שמקבלים מתחים גבוהים מהרצוי ומוציאים את המתח הרצוי. סוג המתח שמייצב המתח מוציא נקבע לפי שתי הספרות האחרונות של הרכיב.

אנו משתמשים במייצב מתח 7805 לכן הוא מוציא 5V. בפרויקט הרכיב כבר בנוי בתוך ה-cap.

לפי המעגל בתמונה יש רגל כניסה של המתח המסופק (V_{in}), רגל של יציאת המתח הרצוי (V_{out}) המתח שיוצא ורגל המחוברת לאדמה.

בנוסף יש שני קבלים מנתקים ("decoupling capacitor"), אחד בין יציאת המתח ואחד בין מתח הכניסה בגדלים של 0.1pF – 0.22pF. ללא שני הקבלים הללו מתח היציאה עלול להיות לא מדויק ולהוציא את ערך גדול או קטן מידי.

קבל הניתוק השמאלי באמצע בין המייצב לבין המתח המסופק ומפריד ביניהם. קבל הניתוק הימני משמש כמוצא קטן לעומס.

. מייצב המתח LM7805 יכול לקחת את מתח הכניסה מ-5V ל-35V. אבל מומלץ להשתמש במתח הכניסה עד 15V בלבד. עם עלייה במתח, יש יותר פיזור חום שדורש גוף קירור גדול יותר. הפלט מווסת המתח מחובר לפין V_{in} של ESP32 ו-GND מחובר ל-GND. לפיכך אתה יכול להפעיל את המודול באמצעות מתאם 9V DC או באמצעות סוללה של 9V.

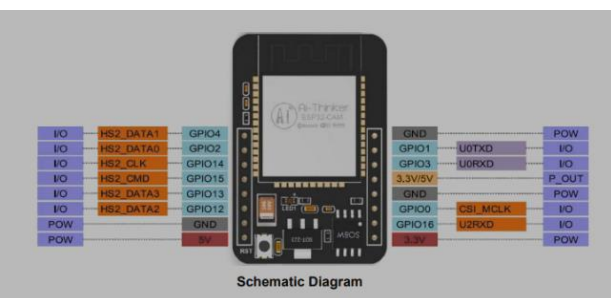


ESP32 - CAM

מצלמה זו מחוברת אל הבקר ESP32 היא יכולה לשדר בזמן אמת תמונות וסרטונים. אפשר לשים את הבקר בכל מקום ובכל פרויקט שיש בו צורך למצלמה כל יסופק אינטרנט.

תפקידי הדקים

שם רגל	תפקיד
GPIO1	משמש כ - משדר TX
GPIO3	משמש כ- מקלט RX
GND	אדמה
GND	אדמה
Vcc	יציאת מתח
5v	מספקת 5 וולט
3.3v	מספקת 3.3v
GPIO2	פין מידע 0
GPIO4	פין מידע 1
GPIO12	פין מידע 2
GPIO13	פין מידע 3
GPIO14	כניסת שעון (CLK)
GPIO15	רגל בקרה

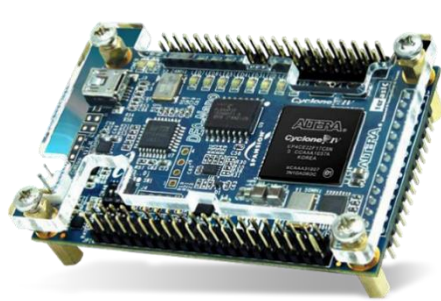


עקרון פעולה

מצלמת אינטרנט המשדרת לאחר מידע (תמונות וסרטונים) בתקשורת WIFI בזמן אמת, הבקר שולח מידע דרך שרת LAN שפותח באמצעות WIFI וניתן לגשת למידע הנשלח על ידי כתובת IP שהבקר מספק לאחר הרצת קוד פשוט.

ערכים חשמליים

- מתח מקסימלי – 5 וולט
- זרם מקסימלי – 310mA
- תדר – 80-240MHz



Altera

Altera Corporation הייתה יצרנית של התקני לוגיקה ניתנים לתכנות (PLD), קווי המוצרים העיקריים מבית Altera היו סדרת הדגל Stratix, סדרת Arria בטווח הביניים, ומערכת סדרת Cyclone בעלות נמוכה יותר על מערכי שערים הניתנים לתכנות בשדה שבב (FPGAs).

התקן לוגי מורכב מתכנות מסדרת MAX ו-FPGAs לא נדיפים תוכנת תכנון Quartus ופתרונות מתח של Enpirion PowerSoC DC-DC.

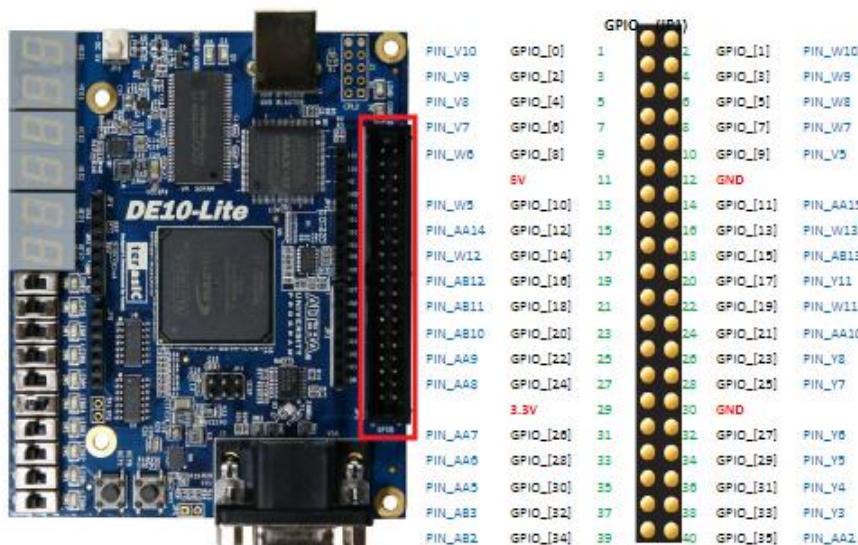
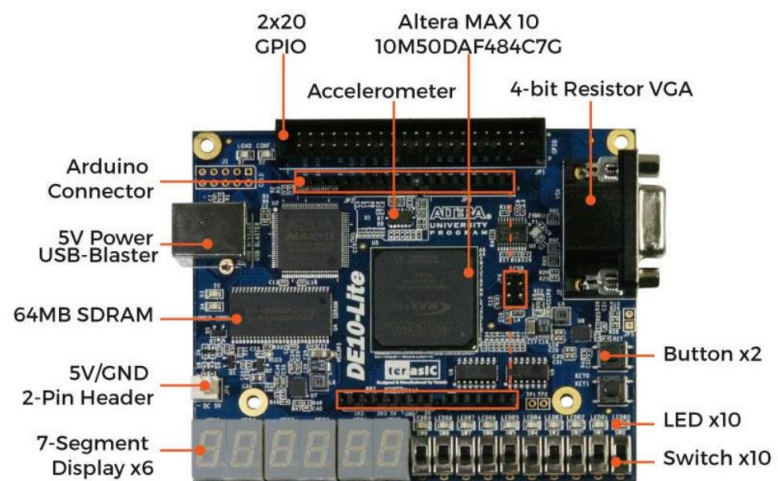
בתוך האלטרנה יש רכיב שנקרא FPGA.

Field-Programmable Gate Array-FPGA

FPGA או מערכי שער הניתנים לתכנות שדה הם התקני מוליכים למחצה המבוססים על מטריצה של בלוקים לוגיים ניתנים להגדרה (CLBs) המחוברים באמצעות חיבורים ניתנים לתכנות.

ניתן לתכנת מחדש רכיבי FPGA לדרישות היישום או הפונקציונליות הרצויות לאחר הייצור. תכונה זו מבדילה בין רכיבי FPGA לבין מעגלים משולבים ליישום ספציפי (ASIC), המיוצרים בהתאמה אישית עבור משימות עיצוב ספציפיות.

תפקיד הדקים



אלה הם extension pins בשביל חיבור רכיבים לבחירה.

עקרון פעולה

הבקר פועל לפי תכנות שפת תיאור חומרה (VHDL) או על ידי הזנת תרשים לוגי, הוא מורכב מיחידות לוגיות הניתנות לתכנות ומרשת של אמצעים לחיבור וניתוק בין היחידות.

הרכיב מבוסס על זיכרון מסוג flex בבקר האלטרנה כלומר רכיב זה אינו זוכר את התוכן לאחר כיבוי/ניתוק של המתח ולכן יש לצרוב את התוכנית עליו לאחר הכיבוי.

מבנה פנימי

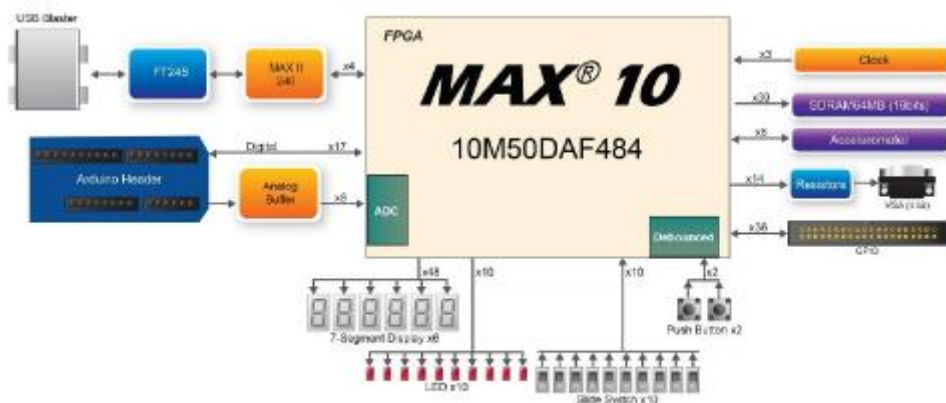


Figure 1-4 Board Block Diagram

ברכיב בנוי כדי לספק גמישות מרבית למשתמש, כל החיבורים נעשים באמצעות התקן MAX 10 FPGA. לפיכך, המשתמש יכול להגדיר את ה-FPGA ליישם כל עיצוב מערכת.

ערכים חשמליים

הרכיב מקבל 5 או 0 וולט.

תקן תקשורת RS232

Universal Asynchronous Receiver Transmitter–UART

משדר מקלט אסינכרוני אוניברסלי.

תקן תקשורת זה מתקשר תקשורת טורית ולא מקבילית, מכיוון שהעברת מידע מקבילי צורכת חוט לכל סיבית וזהו חיסרון ותקשורת טורית מספקת לנו מהירות מהירה מספיק. משתמשים בדרך כלל במידע מקבילי רק בין רכיבים ומערכות שצמודים פיזית כמו המחשב.

בהעברת מידע טורי יש חוט אחד לכל הסיביות ואפשר לחתוך את החוט ולשים משדר בכל צד ולהעביר מידע בין מערכות שפיזית רחוקות אחת מהשנייה, כלומר תקשורת אלחוטית.

שימוש תקן RS232 – להעביר מידע ממקור ליעד בצורה טורית בכל מרחק (בכבל).

סינכרוני – כניסת clk אחת שמחוברת במקביל לשתי מערכות ושתיהן מסונכרנות ביחד כלומר התזמון הוא ביחד.

אסינכרוני – מערכות שאין להן את אותה כניסת clk פיזית, יכול להיות שבמערכות אסינכרוניות התזמון לא היה מתוזמן טוב.

בגלל שתקן זה עובד באסינכרוניות צריך פרוטוקול, שהוא מוודא שהמערכת ששולחת יהיו כללים בדיוק כמו המערכת שקולטת ומטפל בכל הבעיות האפשריות במערכות אסינכרוניות רחוקות.

- שליחת מידע שאומר שהמערכת ששולחת מתחילה לשלוח מידע.
- שליחת מידע בסדר מסוים שהמערכת שקולטת יודעת מהו הסדר.
- שליחת מידע שאומר שהיא מסיימת לשלוח את המידע והמערכת שקולטת צריכה "לארוז" את המידע ולהתחיל לעשות איתו דברים.
- שליחת מידע נוסף בשביל בדיקת המידע שנקלט אם הוא תקין או לא.

קצב עבודה – מספר סיביות המידע שנשלחות בשנייה.

ישנם הרבה קצבים, וקצב העבודה הנמוך ביותר הוא 1200bps וזה נקרא גם 1200 baud rate.

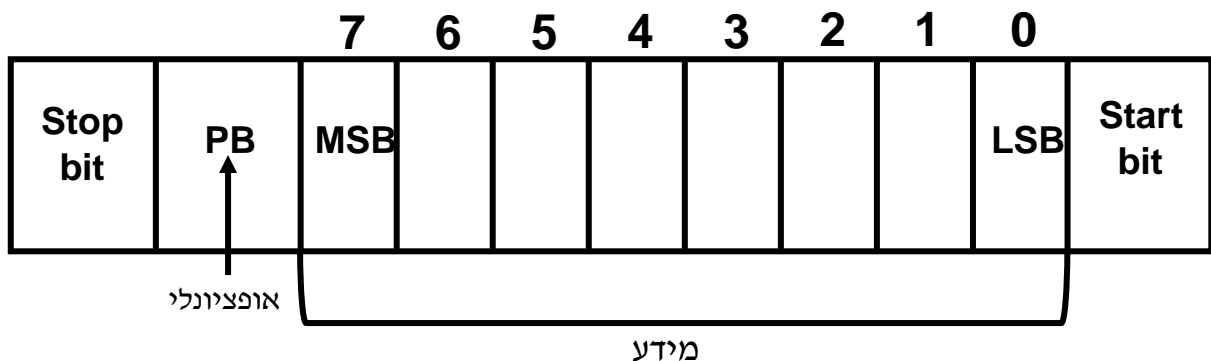
Baud Rate - מספר סיביות המידע שנשלחות בשנייה.

עוד קצבי עבודה הם: 2400bps, 4800bps, 9600bps, 19200bps, 38400bps וכו'. הם מוכרים בכל העולם, ואנו נשתמש בקצב עבודה 9600bps שהוא הנפוץ ביותר בעולם, והוא אינו גבוה מידי ואינו נמוך מידי לכן הוא הנפוץ ביותר.

הסבר פעולת תקן RS232

Data = 8bit (0 - 255) in decimal

(0 - FF) in hex



שליחת המידע תמיד תתחיל מה LSB – אל ה MSB.

סיביות בקרה

בתקן RS232 יש 8 סיביות מידע שתמיד עוטפות אותן סיביות הבקרה האלו:

Start bit = '0'

Stop bit = '1'

זמן מחזור של כל סיבית בתקן RS232 הוא 16 זמני מחזור שעון

סיביות אלו אינן חלק מהמידע, הן רק משמשות כאותות להתחלה וסיום. אם נשלח מידע שאינו עטוף בסיביות אלו (כשבתחילה יש '0' ובסוף יש '1') אז ישנה תקלת תקשורת, והמערכת יודעת להתעלם מהמידע שנשלח עם הסיביות בקרה האלו.

כל עוד לא התחלנו לשלוח מידע יהיה מצב שנקרא **idle** שזה אומר שיש '1' לוגי על החוט של התקשורת כי הוא stop bit. מערכת שקולטת יודעת לזהות stop bit, כלומר היא יכולה לזהות סיבית של '1' לוגי שיש לה זמן מחזור ארוך יותר מ-16 זמני מחזור שעון ולדעת שה '0' לוגי

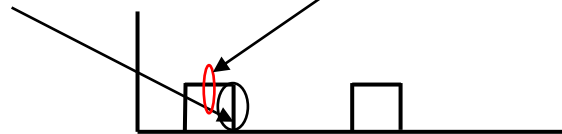
הבא הוא start bit ואז יש מידע. או שהיא יכולה לספור שהיא קלטה 8 סיביות ואז הסיבית הבא חייבת להיות '1'. stop bit = '1'.

סיבית זוגיות – Parity bit – משלימה את מספר האחדות שבמידע לזוגי.

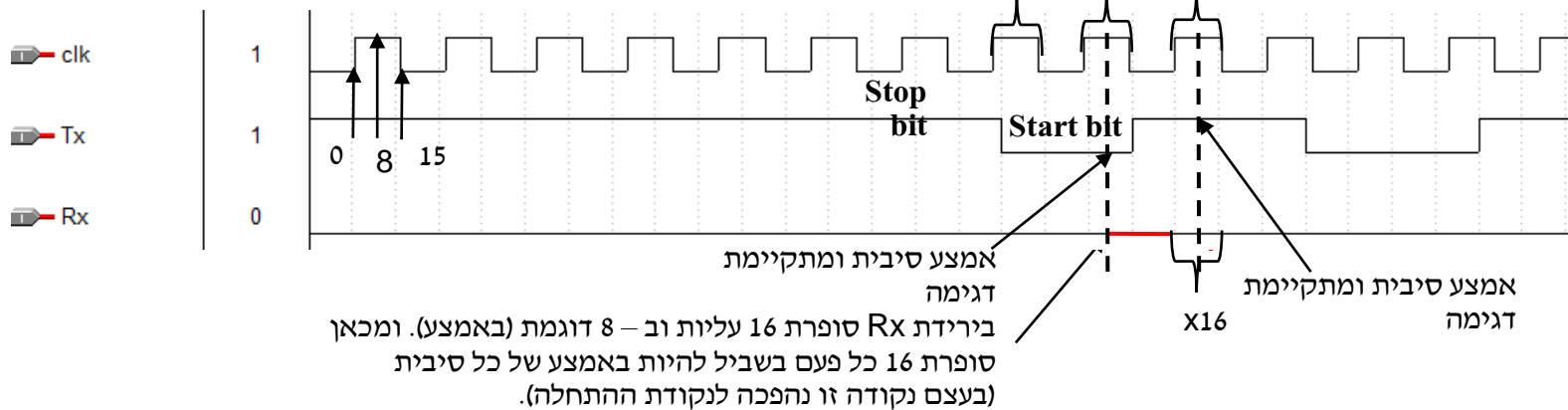
ישנה בעיית תקשורת שהיא דגימת המידע בזמן לא נכון, כלומר המערכת הקולטת הייתה מוכנה לקלוט מידע אבל היא לא הצליחה לקלוט אותו כמו שצריך. לכן נרצה לתפוס את הסיבית באמצע זמן המחזור של הסיבית כלומר חצי זמן מחזור של סיבית. במקרה שלנו זה 8 זמני מחזור שעון (ראה סרטוט).

לא ננסה לדגום את המידע כאן

ננסה לדגום אותה כשהיא כבר יציבה כמו כאן.



דוגמה לתהליך זה בתקן RS232:



כל סיבית נשלחת למשך 16 זמני מחזור שעון כדי להידגם אחרי 8 (לבדוק באמצע).



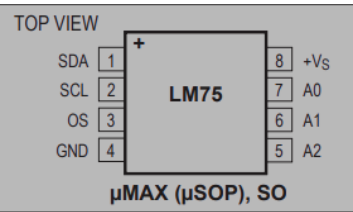
LM75 – Digital Temperature Sensor

LM75 הוא חיישן טמפרטורה דיגיטלי הכולל ממיר דלתא-סיגמה אנלוגי לדיגיטלי, וגלאי טמפרטורות יתר דיגיטלי. המארח יכול לקבוע את ה-LM75 דרך ממשק ה-I2C שלו לקריאת טמפרטורה בכל עת. פלט Open - drain של טמפרטורת יתר (OS) שוקע את הזרם כאשר חריגה ממגבלת הטמפרטורה הניתנת לתכנות. הכתובת של LM75 מוגדרת עם שלושה פינים כדי לאפשר מספר מכשירים לעבוד על אותו BUS.

Open drain - כאשר התקן הפלט כבוי, הפין נותר צף (פתוח, או hi-z). דוגמה נפוצה היא טרנזיסטור n-channel אשר מושך את האות לאדמה כאשר הטרנזיסטור דולק או משאיר אותו פתוח כאשר הטרנזיסטור כבוי.

שימושים :

- ניהול מערכת תרמית
- הגנה תרמית
- בדיקת ציוד
- מחשבים ואלקטרוניקה משרדית
- תכונות :
- חבילות ISO (SOP) ו-μMAX (μSOP).
- ממשק אפיק I2C
- הניקוז פתוח פלט של טמפרטורת יתר שפועל ככניסת פסיקה או השוואת/תרמוסטט
- יכולת קריאה חוזרת של אוגר
- ברירות מחדל להפעלה מאפשרות פעולה עצמאית בתור תרמוסטט
- מתח אספקה של V3.0 עד V5.5
- זרם אספקת הפעלה נמוך 250 μA, mA1 (מקסימום)
- 4μA מצב כיבוי ממזער את הספק הצריכה
- ניתן לחבר עד שמונה LM75 לאוטובוס בודד



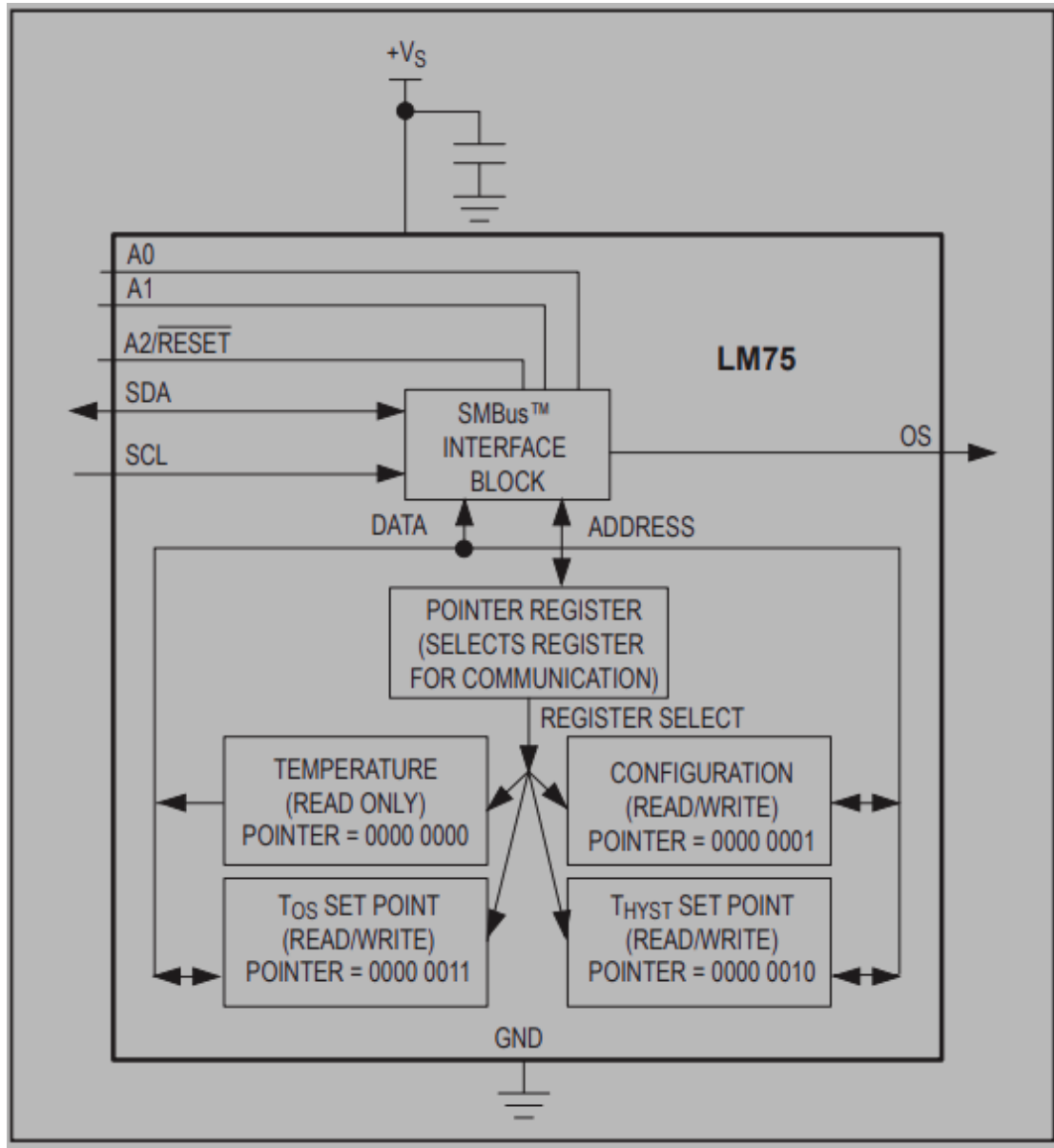
תפקידי הדקים

הדק	שם	תפקיד
1	SDA	קו קלט/פלט טורי-נתונים. Open drain. חבר את SDA לנגד משיכה
2	SCL	קלט שעון טורי. Open drain. חבר את SCL לנגד משיכה.
3	OS	פלט כיבוי בטמפרטורת יתר. Open drain. חבר את מערכת ההפעלה לנגד משיכה.
4	GND	אדמה
5	A2	קלט כתובת ממשק דו-חוטי. חבר את A2 ל-GND או VS+ כדי להגדיר את כתובת ה-BUS הרצויה של I2C. אל תשאיר לא מחובר
6	A1	קלט כתובת ממשק דו-חוטי. חבר את A1 ל-GND או VS+ כדי להגדיר את כתובת ה-BUS הרצויה של I2C. אל תשאיר לא מחובר
7	A0	קלט כתובת ממשק דו-חוטי. חבר את A0 ל-GND או VS+ כדי להגדיר את כתובת ה-BUS הרצויה של I2C. אל תשאיר לא מחובר
8	+Vs	כניסת מתח אספקה חיובית. עוקף ל-GND עם קבל עוקף 0.1μF.

עקרון פעולה

חיישן הטמפרטורה LM75 מודד טמפרטורה וממיר את הנתונים לצורה דיגיטלית באמצעות סוג חיישן טמפרטורה bandgap וממיר דלתא-סיגמה (אנלוגי לדיגיטלי) של 9 סיביות.

קמבנה פנימי



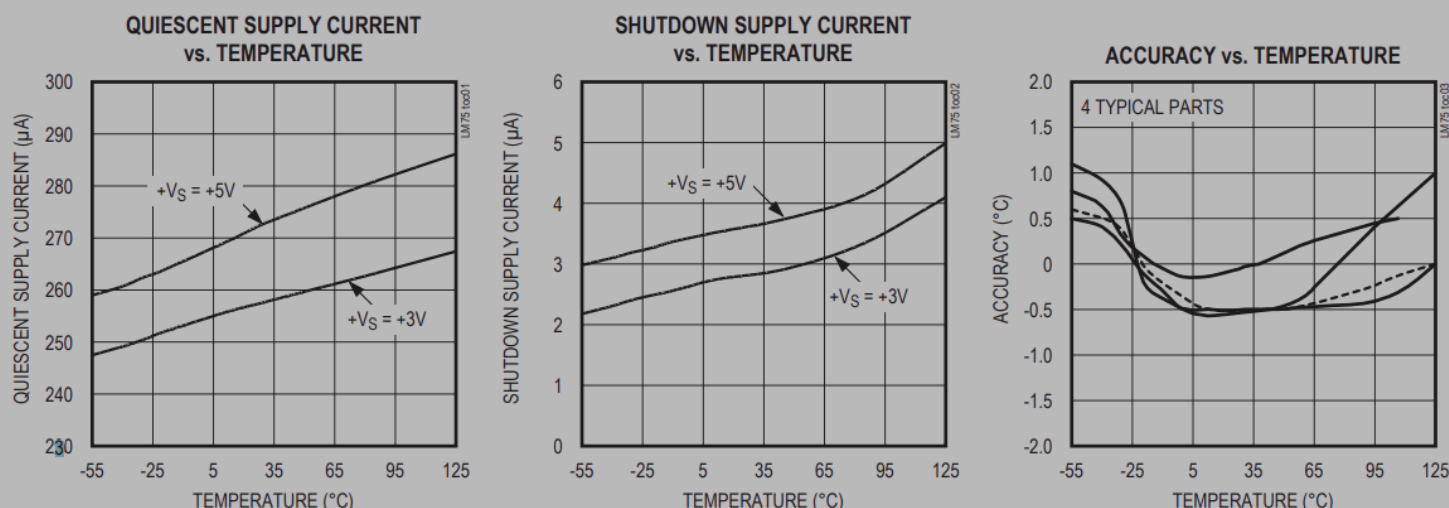
אוגר המצביע של LM75 בוחר בין ארבעה נתונים רגיסטרים. בהפעלה, המצביע הוא מוגדר לקרוא את אוגר הטמפרטורה. אוגר המצביע (pointer) תופס את המיקום האחרון שאליו זה נקבע. כל הרשמים נקראים וכותבים, מלבד מאוגר טמפרטורה, הנקרא בלבד. כתיבה לאוגר ה-Configuration על ידי כתיבת כתובת בייט (a data pointer byte) ובייט מידע. אם 2 הבייטים מידע נכתבים, בייט הנתונים השני מחליף את הראשון.

אוגרי TOS ו-THYST דורשים בייט כתובת אחד, בייט מצביע (pointer) אחד ו-2 בייט מידע. אם נכתב רק בייט מידע אחד, זה נשמר בסיביות D15-D8 של האוגר המתאים. אם יותר מ-2 בייטים של מידע נכתבים, רק 2 הבייטים הראשונים מזוהים בעוד שהבייטים הנותרים מתעלמים.

קריאה מה-LM75 באחת משתי דרכים. אם המיקום נעול ב-Pointer register מוגדר מהקריאה הקודמת, הקריאה החדשה מורכבת מבייט כתובת, ואחריו על ידי שליפת המספר התואם של בתים של נתונים. אם יש להגדיר את אוגר המצביע לכתובת חדשה, בצע פעולת קריאה על ידי

כתיבת בייט כתובת, בייט מצביע (pointer) תחזור על ההתחלה ובייט כתובת נוסף. קריאה בשוגג של 8 סיביות מאוגר של 16 סיביות, עם ה-D7 bit נמוך, יכול לגרום להתקן לעצור במצב שבו קו SDA נשמר נמוך. בדרך כלל, זה ימנע כל תקשורת אוטובוס נוספת עד שהמאסטר ישלח תשעה מחזורי שעון נוספים או SDA עולה גבוה. בזה זמן, מצב עצירה מאפס את המכשיר. אם הנוסף מחזורי שעון אינם נוצרים על ידי המאסטר, ה-BUS של ה-LM75 מתאפס ונפתח לאחר תקופת הזמן הקצוב של ה-BUS חלף.

גרפים ואופניים



מושגים נלווים

נגד משיכה - במעגלים לוגיים אלקטרוניים, נגד משיכה (Pullup) או נגד משיכה (Pulldown) הוא נגד המשמש להבטחת מצב ידוע לאות. הוא משמש בדרך כלל בשילוב עם רכיבים כגון מתגים וטרנזיסטורים, אשר קוטעים פיזית את החיבור של רכיבים עוקבים לאדמה או ל-VCC. סגירת המתג יוצרת חיבור ישיר לאדמה או ל-VCC, אך כאשר המתג פתוח, שאר המעגל יישאר צף (כלומר, יהיה לו מתח בלתי מוגדר).

קבל מעקף (bypass capacitor) - קבלים מעקפים משמשים לשמירה על עכבת אספקת חשמל נמוכה בנקודת העומס. התנגדות טפילית והשראות בקווי אספקה אומרות שעכבת אספקת החשמל יכולה להיות גבוהה למדי. ככל שהתדר עולה, הטפיל האינדוקטיבי הופך לבעייתי במיוחד.

Bandgap temp sensor - חיישן הטמפרטורה לסיליקון הוא צורה נפוצה ביותר של חיישן טמפרטורה (מדחום) המשמש בציוד אלקטרוני. היתרון העיקרי שלו הוא שניתן לכלול אותו במעגל משולב סיליקון בעלות נמוכה מאוד. העיקרון של החיישן הוא שהמתח קדימה של דיודת סיליקון, שעשוי להיות צומת הבסיס-פולט של טרנזיסטור צומת דו-קוטבי (BJT), תלוי בטמפרטורה.

פרוטוקולים

I2C BUS

I2C הוא פרוטוקול תקשורת טורי, סינכרוני, חצי דופלקס המאפשר קיום משותף של מספר מאסטרים ועבדים על אותו BUS. I2C BUS מורכב משני קווים: קו נתונים טורי (SDA) ושעון טורי (SCL). שני הקווים דורשים נגדי משיכה.

עם יתרונות כמו פשטות ועלות ייצור נמוכה, I2C משמש בעיקר לתקשורת של התקנים היקפיים במהירות נמוכה למרחקים קצרים (בתוך רגל אחת).

Firestore – Google

Firestore הוא קבוצה של שירותי אירוח (hosting services) עבור כל סוג של יישום (אנדרואיד, iOS, Javascript, Node.js, Java, Unity, PHP, C... +). הוא מציע NoSQL ואירוח בזמן אמת של מסדי נתונים, תוכן, אימות חברתי (גוגל, פייסבוק, טוויטר ו-Github), והודעות, או שירותים, כגון שרת תקשורת בזמן אמת.

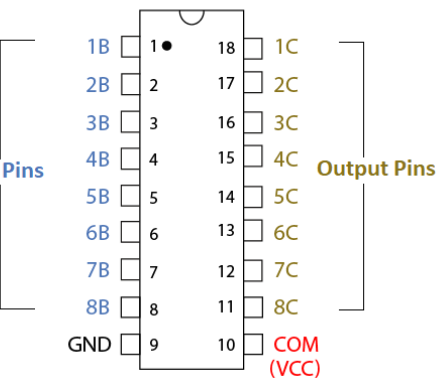
NoSQL - מסד נתונים של NoSQL (הכוונה במקור ל"לא-SQL" או "לא-רלציונלי") מספק מנגנון לאחסון ושליפה של נתונים המעוצבים באמצעים אחרים מלבד היחסים הטבלאיים המשמשים במסדי נתונים יחסיים.

דוחף זרם – ULN2803

שמונת הטרנזיסטורים המחוברים NPN Darlington במשפחת מערכים זו מתאימים באופן אידיאלי לממשק בין מעגלים דיגיטליים ברמה לוגית נמוכה (כגון TTL, CMOS או PMOS/NMOS) לבין דרישות הזרם/מתח הגבוהות יותר של מנורות, ממסרים, פטישי מדפסת או עומסים דומים אחרים. מגוון יישומי מחשב, תעשייה וצרכנים. כל המכשירים כוללים יציאות אספן פתוחות ודיוודות מהדקים מגלגלים חופשיים לדיכוי ארעיות. ה-ULN2803 תוכנן להיות תואם למשפחות TTL סטנדרטיות בעוד שה-ULN2804 מותאם ל-6 עד 15 וולט CMOS או PMOS ברמה גבוהה.



ULN2803 Pinout



הדקים

- פין 1 הוא הבסיס של הטרנזיסטור הראשון.
 - פין 2 הוא הבסיס של הטרנזיסטור השני.
 - פין 3 הוא הבסיס של הטרנזיסטור השלישי.
 - פין 4 הוא הבסיס של הטרנזיסטור הרביעי.
 - פין 5 הוא הבסיס של הטרנזיסטור החמישי.
 - פין 6 הוא הבסיס של הטרנזיסטור השישי.
 - פין 7 הוא הבסיס של הטרנזיסטור השביעי.
 - פין 8 הוא הבסיס של הטרנזיסטור השמיני.
 - פין 9 הוא הפולט כמו גם הארקה של כל הטרנזיסטורים.
 - פין 10 ידוע בתור פין משותף כלומר צומת של קתודה משותפת המשמשת לדיוודות ה- Fly Back.
 - פין 11 הוא האספן של הטרנזיסטור השמיני.
 - פין 12 הוא האספן של הטרנזיסטור השביעי.
 - פין 13 הוא האספן של הטרנזיסטור השישי.
 - פין 14 הוא האספן של הטרנזיסטור החמישי.
 - פין 15 הוא האספן של הטרנזיסטור הרביעי.
 - פין 16 הוא האספן של הטרנזיסטור השלישי.
 - פין 17 הוא האספן של הטרנזיסטור השני.
 - פין 18 הוא האספן של הטרנזיסטור הראשון.
- דיוודת fly – back - דיוודת פליבק היא כל דיוודת המחוברת על פני משרן המשמשת לביטול פליבק, שהוא זינוק המתח הפתאומי הנראה על פני עומס אינדוקטיבי כאשר זרם האספקה שלה מצטמצם לפתע או מופרע. הוא משמש במעגלים שבהם עומסים אינדוקטיביים נשלטים על ידי מתגים, ובמיתוג ספקי כוח וממירים.

ORDERING INFORMATION

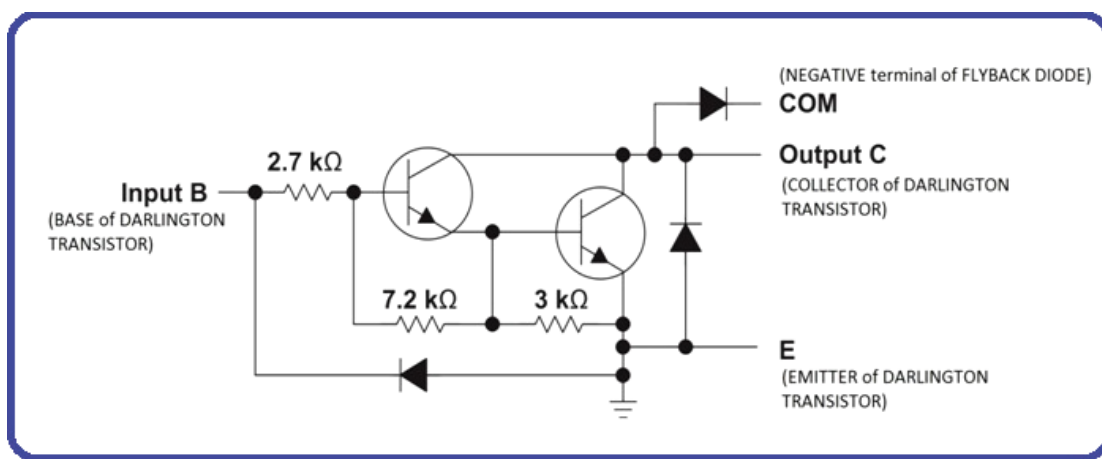
Device	Characteristics		
	Input Compatibility	$V_{CE}(\text{Max})/I_C(\text{Max})$	Operating Temperature Range
ULN2803A	TTL, 5.0 V CMOS	50 V/500 mA	$T_A = 0 \text{ to } +70^\circ\text{C}$
ULN2804A	6 to 15 V CMOS, PMOS		

המפרטים, כמו גם התכונות של ULN2803, הם :

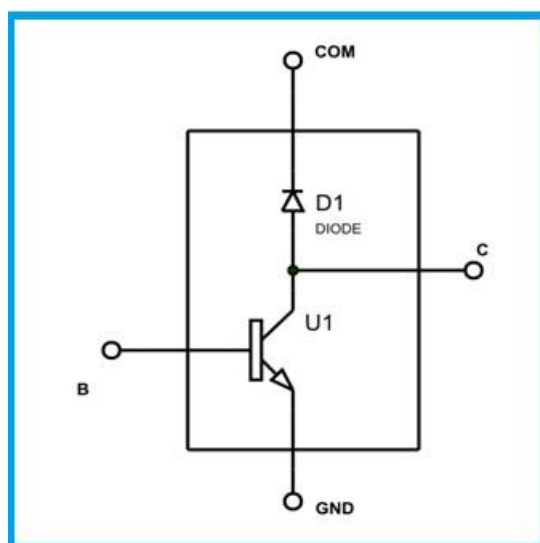
- המתח המרבי המותר בין פולט לאספן עבור כל טרנזיסטור דרלינגטון הוא 50 וולט.
- הזרם המרבי המותר לזרום דרך האספן של כל זוג דרלינגטון הוא 500 mA.
- בין הפולט לבסיס של צמד דארלינגטון, המתח המרבי המותר הוא 30 וולט.
- בכל זוג דרלינגטון, קיימת דיודת זבוב והזרם המרבי המותר לעבור דרכו הוא 500 mA.
- זמן העלייה הוא 130 ns.
- בדרך כלל, זמן הנפילה הוא 20 מיקרו-שניות.
- טמפרטורת הפעולה של ULN2803 נעה בטווח של - 150 – 65 מעלות צלזיוס
- על מנת להפעיל את השבב הזה, אין צורך בכוח נוסף.

מבנה פנימי

השבב ULN2803 מורכב מ-8 זוגות דרלינגטון הפועלים כשמונה מתגים בודדים. מתוך שמונה זוג טרנזיסטורים של דרלינגטון, אם נסתכל על זוג בודד, נוכל לקבל משהו כזה :



ניתן להציג את צמד דרלינגטון הפשוט להלן :



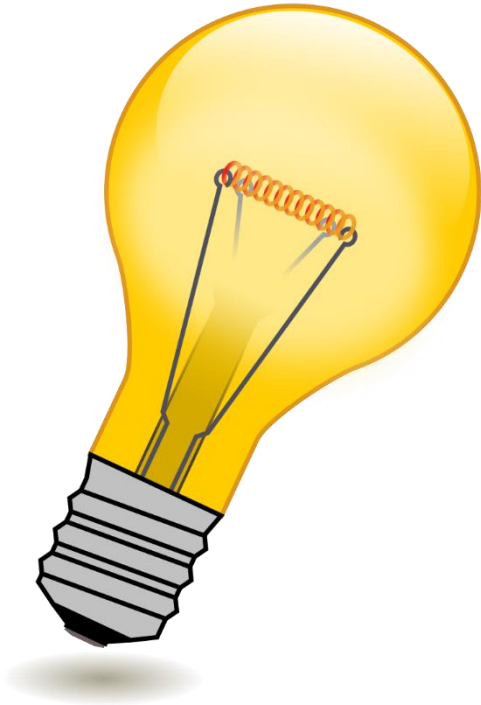
מהנתונים שלעיל, ניתן לראות בבירור שכל זוג של דרלינגטון יכול לשמש כטרנזיסטור כוח יחיד. לפיכך, אנו יכולים לומר שב-ULN2803, שמונה טרנזיסטורי כוח מוטמעים. להבנה טובה יותר, בואו נסתכל לעבר דיאגרמת מעגלים אחרת כדי לייצג דברים בצורה מקיפה יותר.

טרנזיסטור דארלינגטון - דארלינגטון הוא מעגל המורכב מטרנזיסטורים דו-קוטביים, הפלט של טרנזיסטור אחד מחובר לבסיס הטרנזיסטור השני כך שהזרם המוגבר על ידי הטרנזיסטור הראשון מוגבר על ידי הטרנזיסטור השני, הם מחוברים. לתצורה זו יש רווח זרם גבוה יותר מכל טרנזיסטור הוא פועל כמו טרנזיסטור ולעתים קרובות הוא ארוז כיחידה אחת.

נורת להט (הוחלף בלד של ESP32CAM)

נורת ליבון, מנורת ליבון או גלובוס אור ליבון היא אור חשמלי עם חוט מחומם עד שהוא זוהר. החוט מוקף בנורת זכוכית עם ואקום או גז אינרטי כדי להגן על החוט מפני חמצון. זרם מסופק לחוט הלהט על ידי מסופים או חוטים המוטבעים בזכוכית. שקע נורה מספק תמיכה מכנית וחיבורים חשמליים.

לנורה יש שני פינים של אדמה ו- 5 וולט.





מנוע Servo – MG90S

מנוע סרוו הוא סוג של מנוע חשמלי המשמש לשליטה מדויקת של מיקום זוויתי, מהירות ותאוצה. הוא מורכב ממנוע, מפחית, מעגל בקרה ומערכת משוב.

המנוע הוא בדרך כלל מנוע חשמלי DC, אך הוא יכול להיות גם מנוע צעד או מנוע סינכרוני AC. המפחית הוא רכבת הילוכים המפחיתה את מהירות המנוע ומגבירה את המומנט שלו, ומאפשרת למנוע הסרוו לייצר תנועות מדויקות וחזקות.

מעגל הבקרה מקבל אותות כניסה מבקר, כגון מחשב או מיקרו-בקר, ושולח אותות חשמליים למנוע כדי לשלוט בסיבובו. מערכת המשוב, המורכבת בדרך כלל מקודד או פוטנציומטר, מודדת את המיקום האמיתי של המנוע ושולחת מידע זה בחזרה למעגל הבקרה כדי להבטיח שהמנוע זז כמתוכנן.

מנועי סרוו נמצאים בשימוש נפוץ ברובוטיקה, ייצור ויישומים אחרים שבהם נדרשת שליטה מדויקת בתנועה. הם משמשים לעתים קרובות גם בתחביבים RC (מבוקרי רדיו), כמו מזל"טים, מטוסים ומכוניות.

כמה יתרונות של מנועי סרוו כוללים דיוק גבוה, זמן תגובה מהיר ויכולת לפעול במגוון רחב של טמפרטורות וסביבות. הם גם נוטים להיות יעילים יותר באנרגיה מאשר סוגים אחרים של מנועים, מכיוון שהם צורכים חשמל רק כשהם נעים באופן פעיל. סרוו יכול להסתובב בערך 180 מעלות (90 לכל כיוון), ועובד בדיוק כמו סוגים סטנדרטיים אך קטנים יותר.

אתה יכול להשתמש בכל קוד סרוו, חומרה או ספריה כדי לשלוט בסרוו. הסרוו מגיע עם 3 צופרים (זרועות) וחומרה.

יישומים:

- משמש כמפעילים ברובוטים רבים כמו רובוט דו-פעמי, Hexapod, זרוע רובוטית וכו'.
- משמש נפוץ למערכת היגוי בצעצועים RC
- רובוטים שבהם נדרשת בקרת מיקום ללא משוב
- פחות משקל ומכאן בשימוש ברובוטים מרובי DOF כמו רובוטים דמויי אדם

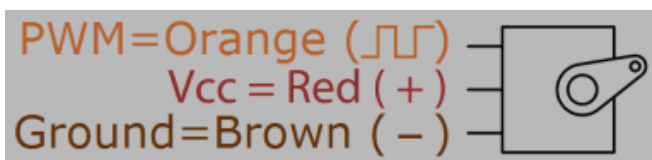
תכונות:

- משקל: 13.4 גרם
- מומנט עצירה: 1.8 (4.8V) $\text{kgf} \cdot \text{cm}$
- סיבוב: $180^\circ - 0^\circ$
- מהירות פעולה: 0.1 שניות/60 מעלות
- מתח הפעלה: 4.8 וולט
- רוחב פס מת: 5 μs
- זרם (סרק) 10mA (אופייני)
- זרם (אופייני בזמן תנועה) 120-250mA
- נוכחי (דוכן)

$\text{kgf} \cdot \text{cm}$ – סנטימטר של כוח קילוגרם ($\text{kgf} \cdot \text{cm}$) הוא יחידה מטריית של מומנט (נקראת גם "מומנט" או "רגע הכוח"). סנטימטר אחד של כוח קילוגרם שווה למומנט הנובע מכוח של כוח של קילוגרם אחד המופעל בניצב על זרוע מומנט באורך סנטימטר אחד.

הדקים

המנוע מגיע עם שלושה הדקים המחוברים למנוע בצבעים קבועים:



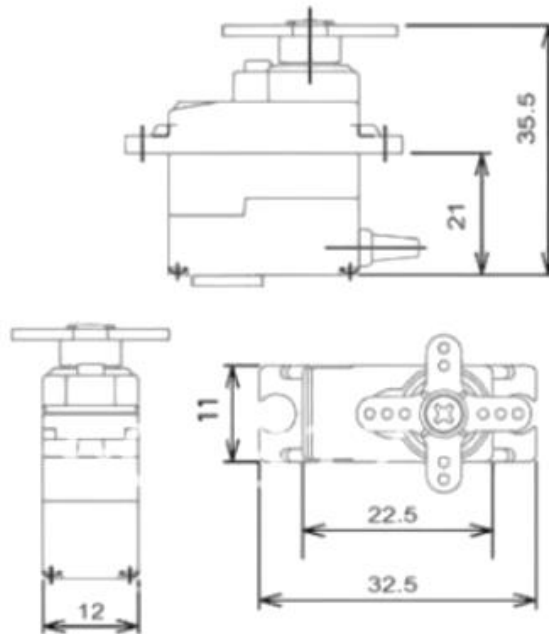
- כתום – PWM

- אדום – 5v
- חום – GND

עקרון פעולה

שליטה של מיקום זוויתי ודיוק מירבי ומהיר לפי קבלת ה – PWM הרצוי.

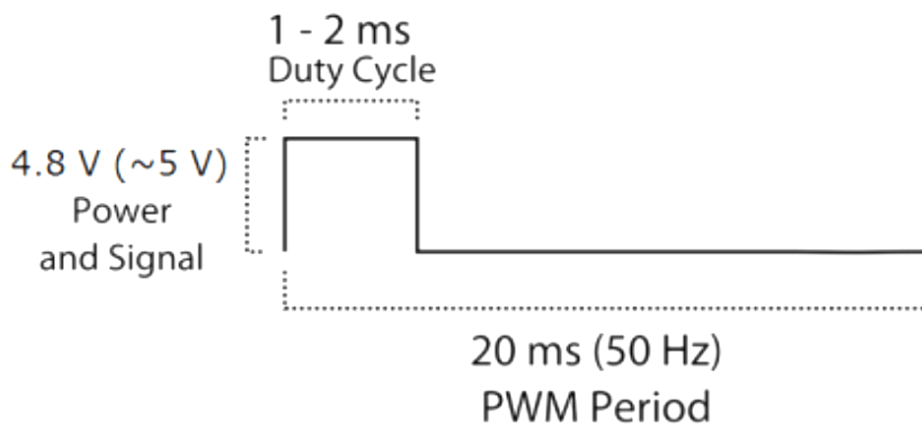
מבנה פנימי



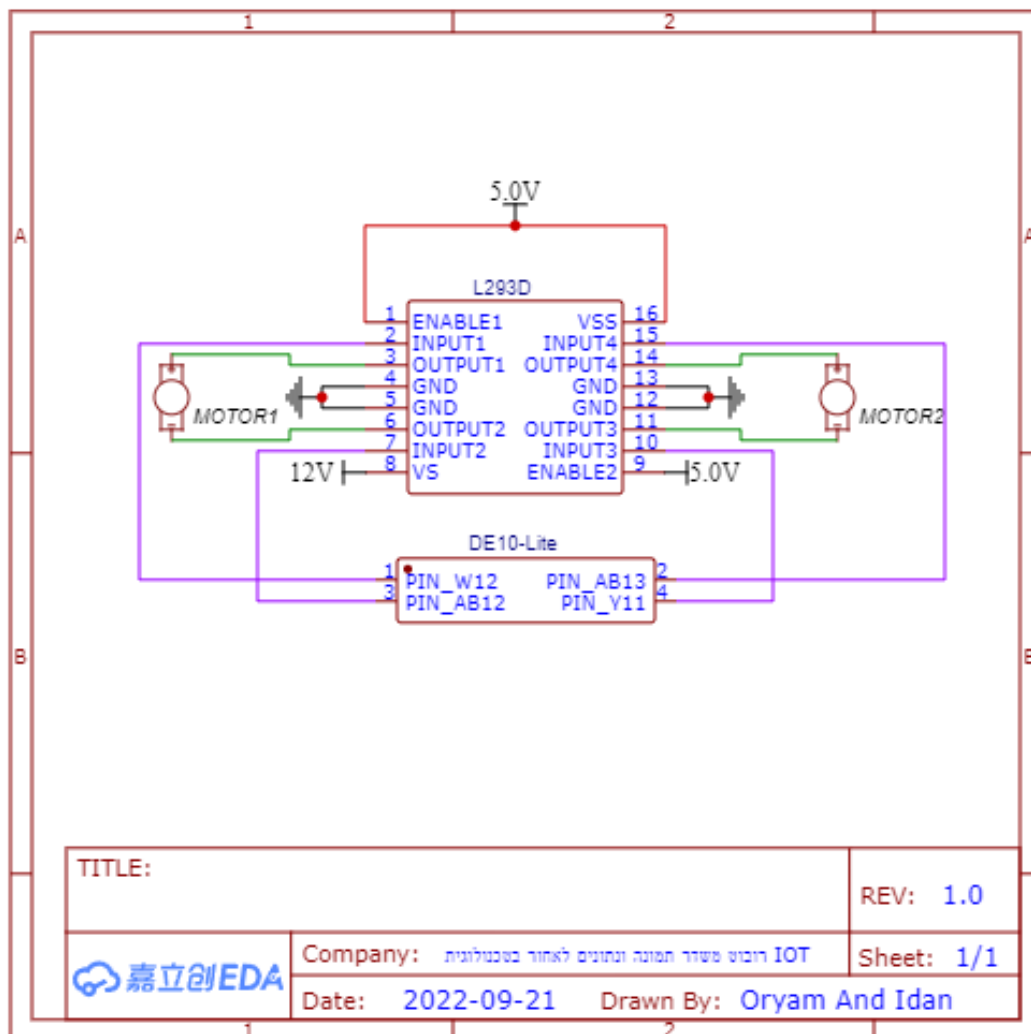
מנועי סרוו מורכבים לרוב ממנועי DC המשתמשים במנגנוני משוב כדי לנוע בדיוק רב ממיקום אחד לאחר כפי שניתן לראות בתמונה. מנועי סרוו מכילים סדרה של גלגלי שיניים שמאיצים או מאטים ומחלקים את התנועה של מנוע ה-DC. לבסוף, מנועי סרוו משתמשים במעגל כדי לשלוט ולשלוח מידע משוב לבקר נתון.

גרפים ואופיינים

מהתמונה אנו יכולים להבין כי לאות ה-PWM המופק צריך להיות תדר של 50Hz כלומר, תקופת ה-PWM צריכה להיות 20ms. מתוכם זמן ההפעלה יכול לנוע בין 1ms ל-2ms. אז כאשר זמן ההפעלה הוא 1ms המנוע יהיה ב-0° וכאשר 1.5ms המנוע יהיה 90°, באופן דומה כאשר הוא 2ms הוא יהיה 180°. לכן, על ידי שינוי זמן ההפעלה מ-1ms ל-2ms ניתן לשלוט במנוע מ-0° ל-180°.



DC Motor to L293D

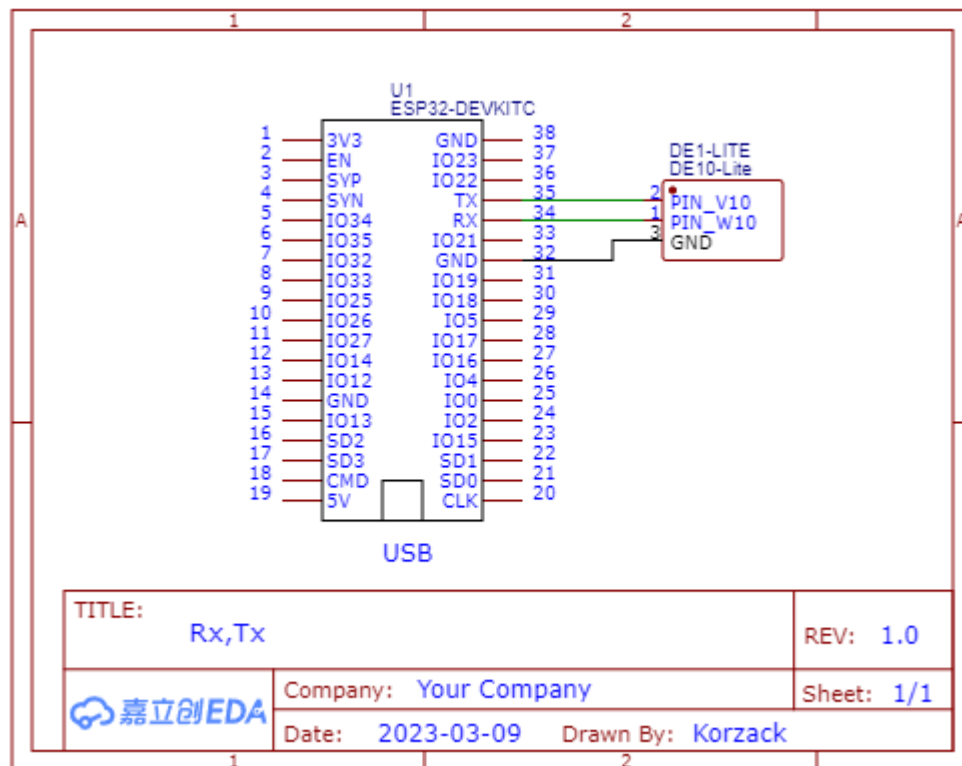


חיבור כניסות מתח ל - 5 וולט (1,16), חיבורי המנועים אדמה בESP (4,5,12,13) וכניסת מתח (3,6,11,14) בדוחף זרם. חיבור כניסת מידע לDE10 (2,7,10,15). חיבור enable2 ו vs ל - 5 וולט ו 12 וולט.

המנועים מחוברים לדוחף זרם ברגלים 3,6,11,14 ומקבלים מתח של 9 וולט ישיר מהסוללה (הלחמנו חוט ישיר במקום שהמתח יועבר במייצב מתח ויקטן), לדוחף זרם יש 4 כניסות מידע שמחוברות לאלטרה ואומרות לדוחף זרם מתי להזרים זרם למנועים לפי הסיביות מידע שמקבלים מהמשתמש ששולח אותן דרך האפליקציה בפרוטוקול WIFI לפיירבייס ומשם לבקר ESP32 דרך פרוטוקול WIFI ואז לFPGA דרך RX,TX ששם הסיביות מידע מתחלקות לתוכניות המתאימות בתכנון ההיררכי שבתוך ה-FPGA.

למנועים יש תוכנית של PWM_CTRL שנמצאת בתוך ה-FPGA שהיא אומרת לדוחף זרם לאן להזרים זרם (לאיזה חוט מידע של המנועים להזרים) לפי תדר של 10KHz ב D.C של 80% . הזרם הוא 800mA. המטרה שלו לדוחף זרם מתאים למנועים בתדר של 10KHz עם D.C של 80% ברגע שהוא מקבל את הסיביות המתאימות מהאלטרה.

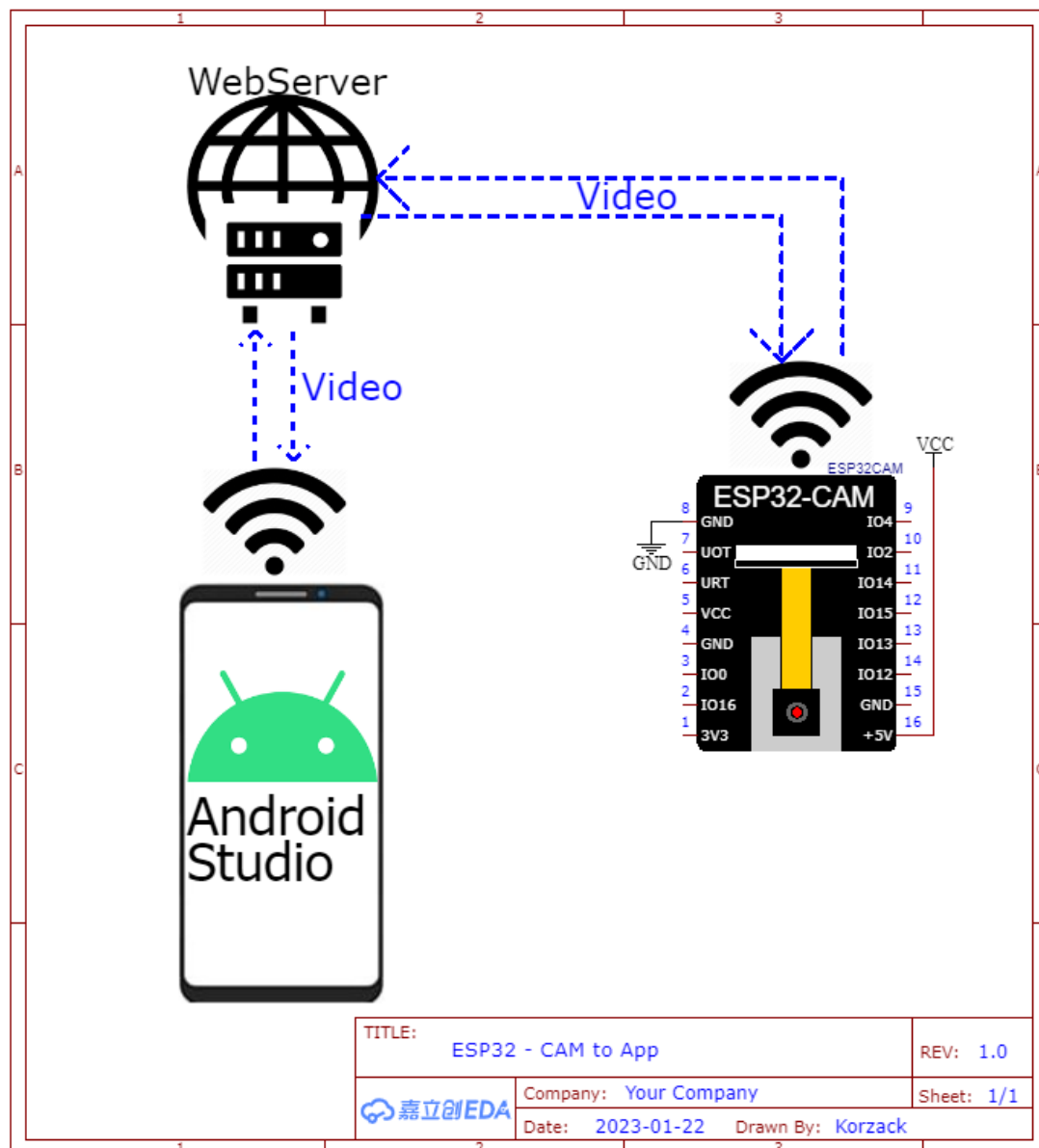
ALTERA to ESP – TX to RX



חיבור Tx & Rx בין DE10 ל- ESP32.

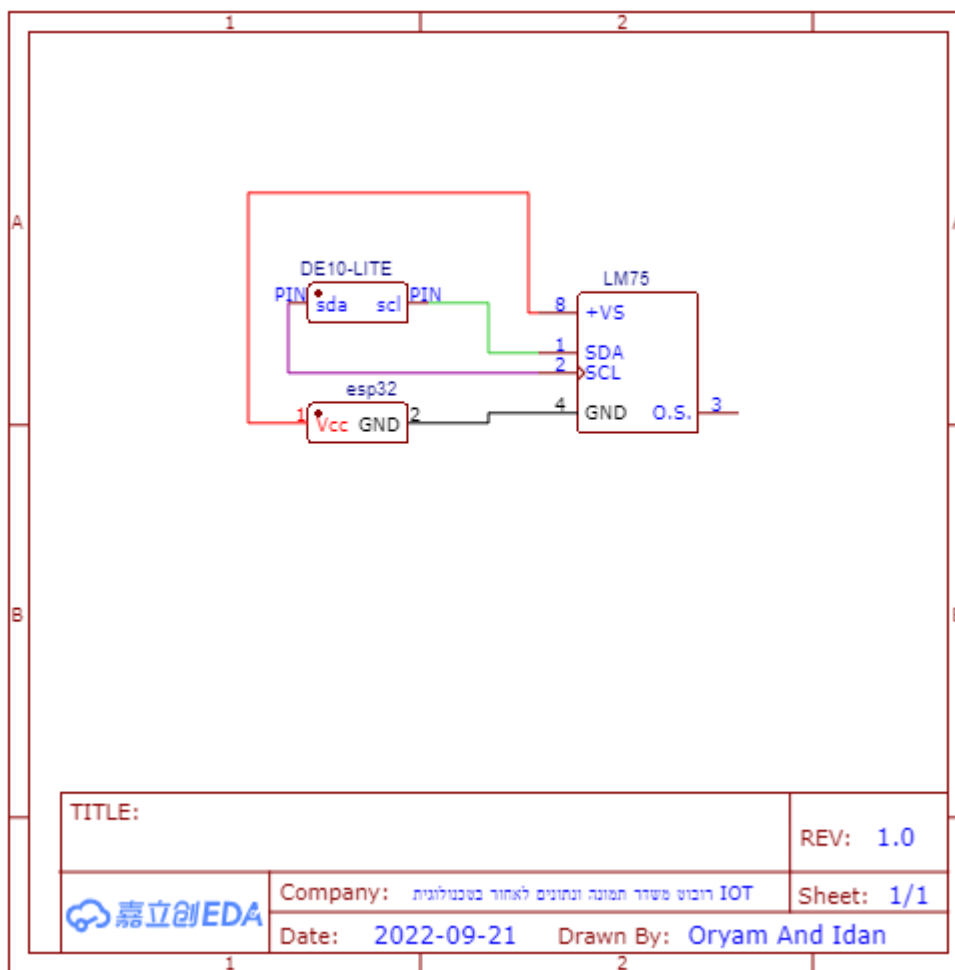
המטרה של הבקר ESP32 הינו לשמש כמטווח בין האלטרס לאפליקציה. מהאפליקציה נשלחות 8 סיביות מידע דרך פרוטוקול WIFI לפיירבייס ומהפיירבייס לבקר גם דרך פרוטוקול WIFI ומהבקר לאלטרס דרך TX, RX. לאחר שהאלטרס קולטת את הסיביות, הן מתחלקות לתוכניות המתאימות שלהן בתכנון ההיררכי של האלטרס לפי I/O ב- PIN PLANNER.

ESP32 – CAM to Firebase to Java App



המצלמה מתחברת ל-WIFI וכך מתקשרת עם השרת רשת, כאשר רוצים להדליק את הפנס שלה נשלחת פקודה דרך שרת הרשת להדלקתה אל המצלמה דרך WIFI. המצלמה משדרת את התמונה דרך פרוטוקול WIFI ו-IP שלה. באפליקציה התמונה מעובדת ואם רוצים אפשר להפעיל עיבוד/זיהוי תמונה.

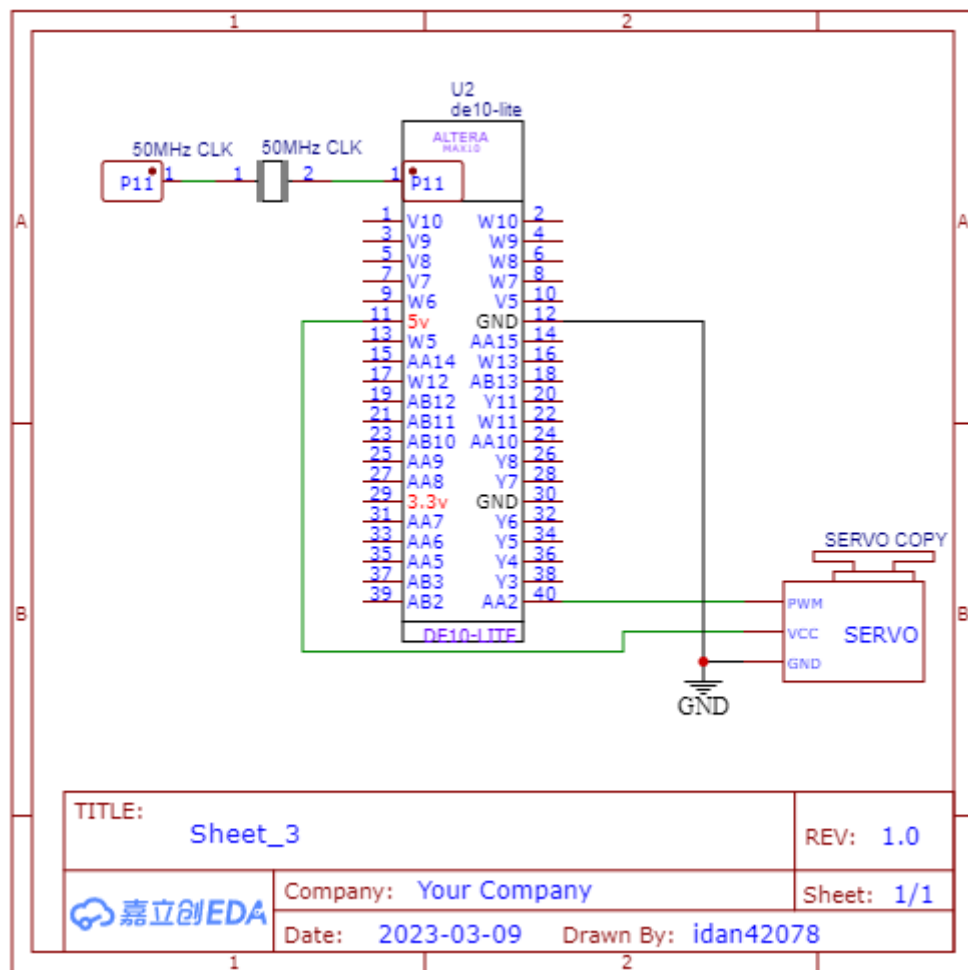
LM75 TEMP SENSOR to ALTERA And ESP32



חיבור Vcc ואדמה ל – ESP32 כניסות מידע לאלטרה (1,2)

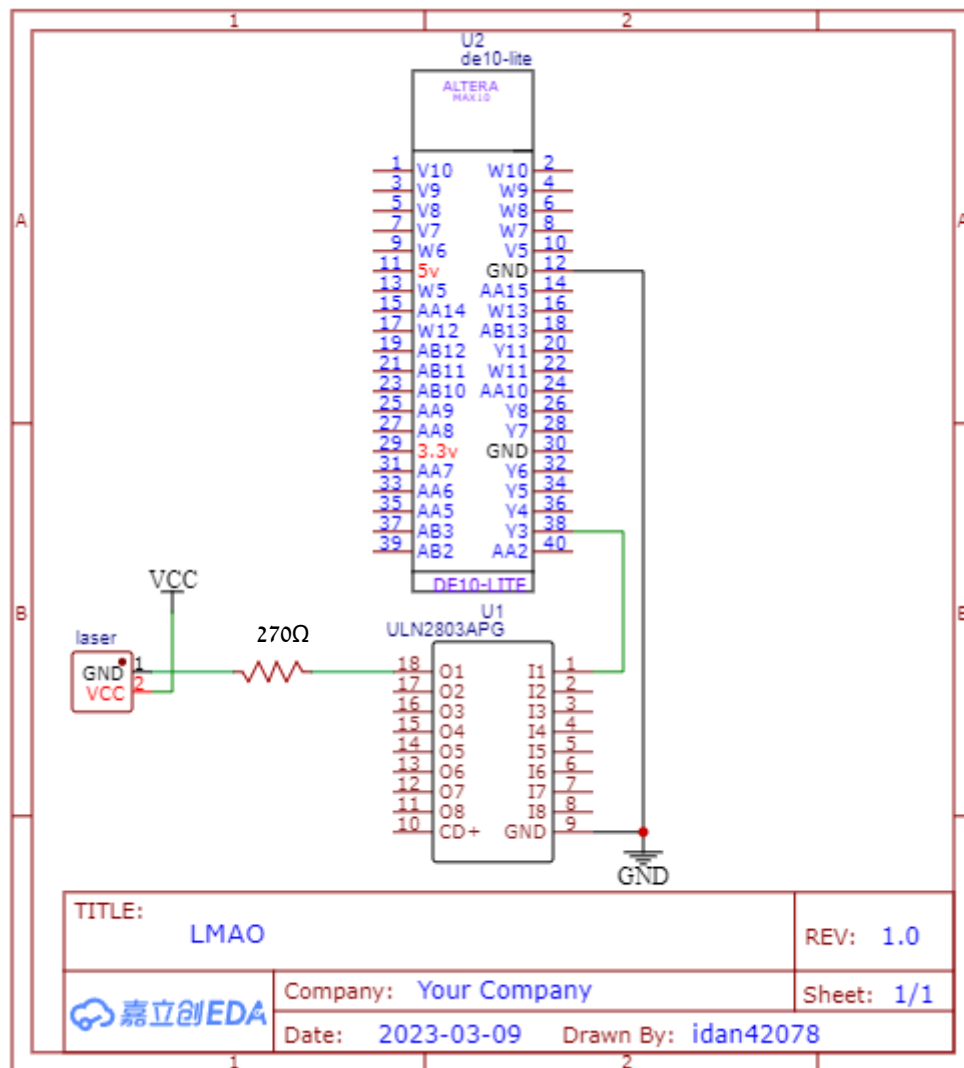
המטרה של רכיב זה היא להעביר את הטמפרטורה שהוא קולט בשטח דרך האלטרה בעזרת פרוטוקול I2C אל ה – ESP32 בעזרת TX,RX יש לו שני כניסות מידע שמחוברות לאלטרה אליה הוא שולח את הטמפרטורה דרך תוכנית VHDL בתשע סיביות ובאלטרה גם מומר ל 8 סיביות. וממנו אל הפיירבייס דרך פרוטוקול WIFI ואל האפליקציה דרך WIFI שם יש המרה מפרנהייט לצלסיוס. יש לו כניסת אדמה וכניסת מתח בין 2.7- 5.5 וולט. כניסת ה-O.S אינה משומשת ומראה מתי הרכיב הגיע למקסימום טמפרטורה.

Servo motor to Altera



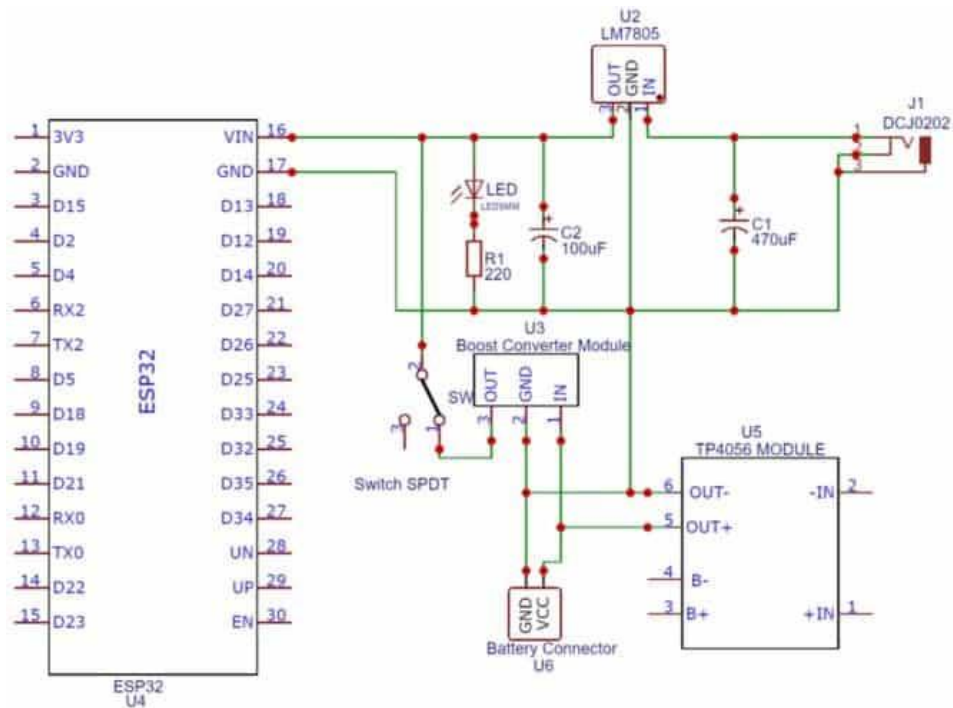
מנוע הסרבו מחובר לאלטרה ומקבל ממנה 5v ואת הPWM שלו, כאשר רוצים להזיזו מהאפליקציה בעזרת פרוטוקול WIFI נשלחות 8 סיביות כאשר 3 מהן משמשות לסרבו לשמונה מצבים מ-"000" עד "111" כלומר כל הזזה הינה 22.5 מעלות. שלושת הסיביות נכנסות לתוכנית servo_control בתכנון ההיררכי של האלטרה ומשם לPWM של הסרבו בתדר של 50MHz. בתדר PWM יש זמן בגובה בין 1ms – 2ms לדוגמה כאשר זמן התדר הוא 1.5ms הסרבו יהיה ב - 90°, שמשפיע על זווית הסרבו שמשפיע על כיוון הלייזר.

Uln to laser to altera



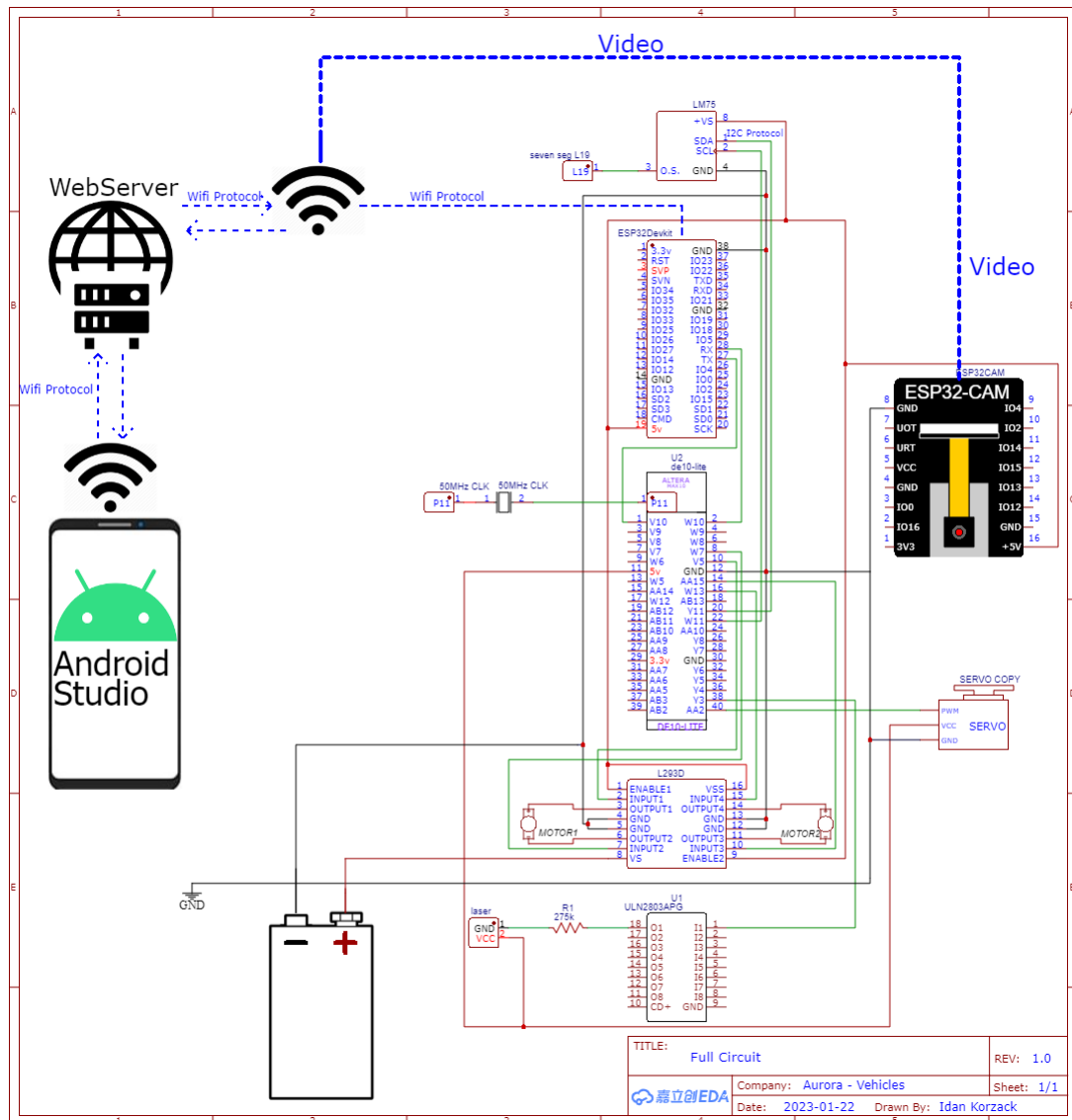
הלייזר מחובר לדוחף זרם ULN שמספק לו את כניסת המידע אם להדליק אותו או לא (0 או 1 לוגי) ובנוסף מחובר לנגד של 270 אום לוסי את הזרם שהלייזר לא ישרף. כאשר האלטרס מקבלת סיבית מידע מהאפליקציה בעזרת פרוטוקול WIFI והESP היא עוברת בתכנון ההיררכי של האלטרס ומותאמת למוצא המתאים ב – PIN PLANNER.

מייצב מתח של ה-ESP



וסת המתח LM7805 IC יכול לקחת את מתח הכניסה מ-V7 ל-V35. אבל מומלץ להשתמש במתח הכניסה עד V15 בלבד. עם עלייה במתח, יש יותר פיזור חום שדורש גוף קירור גדול יותר. הפלט מווסת המתח מחובר לפין Vin של ESP32 ו-GND מחובר ל-GND. לפיכך אתה יכול להפעיל את המודול באמצעות מתאם 9V DC או באמצעות סוללה של V9.

מעגל שלם של כל הפרויקט



תהליך הפעלת הפרויקט מתחיל באפליקציה בכל פעולה, דרך האפליקציה המשתמש מקבל נתונים מהשטח ושולח פקודות אל השטח.

בשביל להניע את המנועים אם המשתמש לוחץ על אחד המצבים (קדימה, אחורה, ימין שמאלה) המידע עובר בפרוטוקול WIFI אל שרת הרשת ומשם אל הבקר ESP32, הבקר מקבל את הנתונים שמשמעותם מספר בטווח 0-255. הבקר שולח אותם אל האלטרנה בתקן תקשורת RS232, האלטרנה מחלקת את 8 הסיביות שהתקבלו לחלקים של מידע שמחולקים להזזת המנועים, קביעת מהירות ודברים אחרים.

לאחר פירוק המידע היא שולחת את הסיביות לתוכניות המתאימות לכן בתכנון ההיררכי המורכב בתוך ה-FPGA, ומשם לצרכנים שאותם החלטנו להפעיל במקרה הזה המנועים.

אם החלטנו על נסיעה קדימה התוכנית המתאימה באלטרנה תקבל "0101", ודוחף הזרם L293D מקבל PWM שתפקידו לקבוע את מהירות המנועים על ידי D.C משתנה בתדר קבוע של 10KHz. ה-L293D מכיל שני גשרי H שתפקידם לספק הספק גבוה למנועים שהבקר אינו יכול לספק, וגם אם כן זה אינו מקובל שהבקר שתפקידו לחשוב יעשה זאת, ללא דוחף הזרם לא היינו יכולים להניע את המנועים. הזרם המתאים עובר דרך גשרי ה-H לכל מנוע יש גשר H ועובר דרך המנועים בכיוון המתאים לכיוון הנסיעה שבחרנו.

במידה ונרצה להפעיל את הלייזר, המידע ששלחנו מהאפליקציה בגודל 8 סיביות שמתפרק של חבילות של סיביות עובר לתוכניות המתאימות בתכנון ההיררכי של ה-FPGA וממנו אל ה-uln2803 שתפקידו למשוך זרם ללייזר. האלטרסה תספק ללייזר 5v בזרם מאוד נמוך הטרנזיסטור דארלינגטון שבתוכו יכנס לרוויה זרם יגיע מה-Vcc, יעבור את האנודה אל הקתודה של הLED דרך נגד בגודל $275K\Omega$ שתפקידו לווסת את הזרם בצורה נכונה כך שהLED לא יישרף, ויזרום אל האדמה של המעגל. אם האלטרסה תיתן 0 וולט הטרנזיסטור יהיה בקטעון ויהיה נתק.

מנוע הסרבו מקבל תדר של 50MHz שבתוכו זמן בגבוה בין 2ms – 1ms לדוגמה כאשר זמן התדר הוא 1.5ms הסרבו יהיה ב- 90° , שמשפיע על זווית הסרבו שמשפיע על כיוון הלייזר.

המצלמה נותנת לנו אפשרות לראות כל מקום שיש בוא אינטרנט את מה שיש בשטח שהיא נמצאת בו. אנו מבצעים גם פעולת עיבוד וזיהוי תמונה שתפקידה לאסוף נתונים בשטח ולשלוח אותם למשתמש ובנוסף נותנת אפשרות לרכב להגות אוטומטית כלפי חפץ שנבחר.

LM75 הינו חיישן טמפרטורה שמסדר את טמפרטורת הסביבה של הרכב בעזרת פרוטוקול I2C אל ה-FPGA, התוכנית המתאימה ב-FPGA קולטת את הטמפרטורה שנוגע בפרוטוקול ב-9 סיביות אחר ב-FPGA המידע מומר ל-8 סיביות. לאחר מכן הטמפרטורה נשלחת בפרנהייט דרך המסדר של האלטרסה אל הקולט של ה-ESP32 ומשם שולח את הטמפרטורה אל שרת הרשת בעזרת פרוטוקול WIFI, ומשם אל האפליקציה שם הטמפרטורה מומרת לצלסיוס. תהליך זה בכיוון ההפוך מכיוון שהמידע מגיע מהשטח אל השרת רשת ומשם לאפליקציה ולא ההפך.

מידות

VHDL תוכניות

חיסרון משמעותי במעבדים של אינטל הוא שהמעבד יכול לבצע רק פעולה אחת.

אפשר לעשות אינסוף פעולות VHDL היום בזכות רכיבים המתוכנתים ב -

המתרחשות במקביל.

עוד יתרון זה שהכל נמצא על ציפ אחד לעומת מעבד רגיל של אינטל שצריך עזרה מלוח האם, בנוסף לכך ספיקת הכוח של הציפ גם הרבה יותר נמוכה ויעילה בהשוואה לספיקת הכוח של מעבד רגיל.

VHDL

V - stands for VHSIC which stands for – Very High Speed Integrated Circuit רכיבים בעלי תכנות מהירים מאוד –

HDL – stands for – Hardware Description Language שפת תיאור חומרה

תמצית רקע יתרונות ושימושים של השפה :

רקע

תחילת שנות ה – 80

דרישה של גופי פיתוח בעולם וה – DOD

יכולות גבוהות וחדשות של תכנות חומרה

יתרונות

תכנות התנהגותי של תוכנה – הפיכת קוד ב – VHDL למערכת שערים לוגים.

אוניברסליות – אפשר להעביר דרך האינטרנט קבצים למקומות שונים בעולם ולפתוח אותם בכל שפת פיתוח.

יכולות תיקון ושדרוג מהירים – אם רוצים לשנות רכיב כלשהו או משהו בתוך הרכיב כמו מספר כניסות או יציאות במקום לבנות ולממש אותו מחדש, הולכים לקוד של VHDL ומשנים שם.

יכולת עבודת צוות בתכנון היררכי – לקחת מערכת עם תפקיד כלשהו, לפרק אותה לחלקים קטנים, כל חלק קטן במערכת נפרק לעוד חלקים קטנים ובסוף נרכיב את כל החלקים ונקבל את המוח שעושה את הפעולה השלמה.

שימוש במיתולוגיות:

TOP DOWN – לקחת את המערכת השלמה והתפקיד, ולפרק אותו לחלקים ולבנות אותם אחד אחד.

דוגמה ל- TOP DOWN :

מחלקים מוח של תוכנית לתוכניות קטנות ואז , תחום האחריות של כל תוכנית נורא קטן אז יוצא שכל מתכנת צריך לעשות תוכנית מאוד קטנה ופשוטה וזה יתרון

מאוד גדול כי קשה לטעות בה אפשר לבדוק מהר אם היא טובה וגם אפשר שכל מי שעובד יהיה בכל מקום בעולם (Outsourcing)

BOTTOM UP – לקחת את החלקים ולחבר אותם ולהגיע למערכת שלמה.

פעם בשביל ליצור מוח כלשהו לתפקיד מסוים היו יוצרים קובץ אחד ארוך מאוד ועליו לתכנת את המוח, וזה חיסרון גדול מאוד מכיוון שקשה לעבוד עם קובץ אחד ארוך, קשה למצוא טעויות, ומקסימום אנשים שיכולים לעבוד עליו הוא אולי 2 והם צריכים להיות מסונכרנים לשינויים שהם עושים שזה עוד יותר מסובך. לכן מחלקים אותם לחלקים.

סימולציית חומרה על מסך מחשב - בשביל להגיש תוכניות שבודדות עובדות ישנן 2 דרכים:

בדיקה בדיאגרמת זמן (Waveform)

Test bench

Test bench – לכתוב תוכנית ב – VHDL שבודקת את התוכנית שבנינו ב – VHDL

DUT - Design under test - שם לתוכנית ב – VHDL (מושג בתכנות של VHDL)

UUT - Unit under test - שם לתוכנית ב – VHDL (מושג בתכנות של VHDL)

התוכנית שבנינו היא תוכנית לסינטזה כלומר לצריבה על מכשיר, המטרה של Test bench היא להתקיל את תוכנית הסינטזה ובמקום לראות תוצאות ב – Waveform נקבל התראות. לדוגמה - "תגיד לי אם B השתנה כש – A השתנה זמן כזה או כזה אחרי 20ns" (ns = nano seconds). כלומר כותב ב – Test bench מה שאני לא רוצה שיקרה כמו Compiler.

חסרונות ב – VHDL

ישנו חיסרון אחד ב – VHDL והוא שאי אפשר לתכנת רכיבי כוח (נגד, דיודה, מגבר). היום יש טכנולוגיה שעדיין בפיתוח שנקראת ASIC שמאפשרת לתכנת קצת פעולות אנלוגיות (כוח) ולא רק דיגיטליות.

שימושים של VHDL

ניצול יכולת פעולה מקבילית של אותות להתניות – תכנות רכיבים שיכולים לבצע אינסוף פעולות במקביל.

החלפת כרטיסים גדולים ומסורבלים ברכיב אחד – במקום לזרוק את הרכיב ולהחליף הכל ברכיב אחד, נשתמש ברכיב שמתוכנת ב – VHDL.

שימוש מהיר וקל במיקור חוץ לדרכי פיתוח – אפשר להשתמש במיקור חוץ (Outsourcing) בשביל לבנות רכיבים שונים גם אם האנשים לא חלק המחברה.

Outsourcing\מיקור חוץ – לא כל האנשים חייבים להיות בחברה ובאותו מקום בשביל לתכנת רכיב, כלומר אפשר לחפש כותבי VHDL מנוסים ולפי ניסיונם לשכור אותם ולתת להם משימה לכתוב רכיב כלשהו, וכך אפשר גם לחסוך בכוח אדם.

משדר – Tx – Transmitter

תפקיד התוכנית - לקבל מידע מקבילי בגודל 8 סיביות ולהוציאו למידע טורי בתקן RS232.

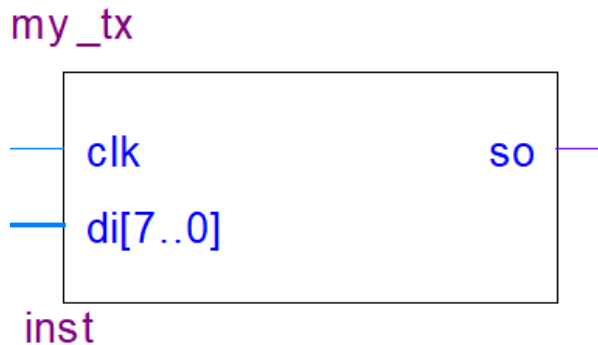
עקרון כתיבת Tx

10 סיביות x 16
מחזורי מחזור של כל סיבית

1. נגדיר משתנה פנימי cnt שיספור עד 160 מחזורי.
2. נגדיר משתנה פנימי clr לאיפוס בסוף שליחת מידע.

ישנם 10 סיביות בגלל שזה בתקן RS232 ויש 8 סיביות מידע ו- 2 סיביות בקרה.

סמל הרכיב:

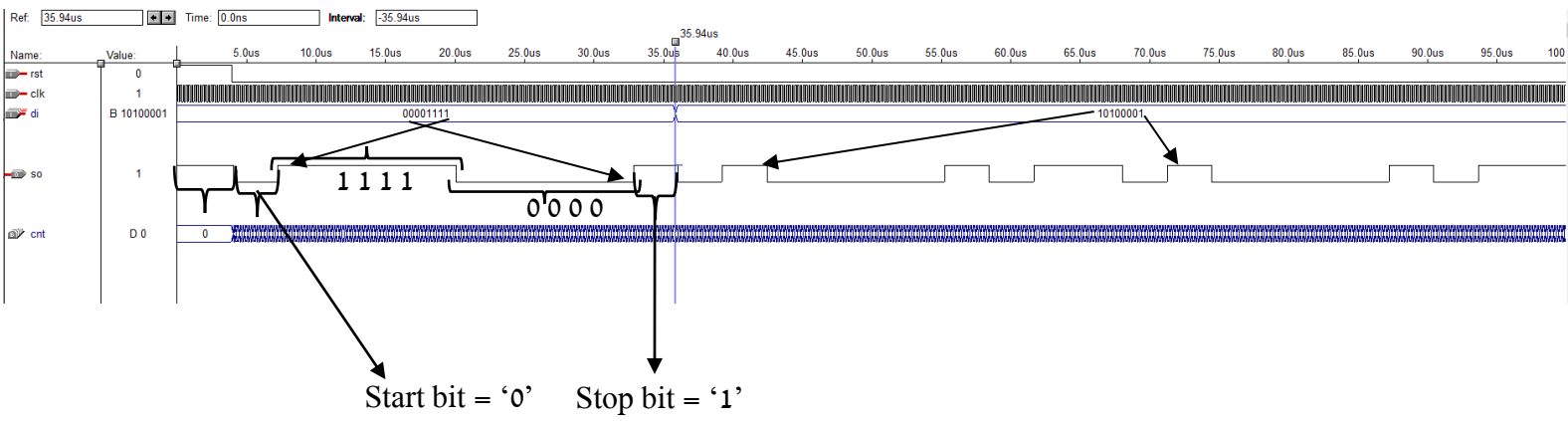


קוד התוכנית:

```
library ieee;
use ieee.std_logic_1164.all;
entity my_tx is
port (clk: in bit;
      di : in bit_vector(7 downto 0);
      so : out bit);
end;
architecture behave of my_tx is
signal cnt : integer range 0 to 160;
signal clr : bit;
begin
process (clk)
begin
if clr='1' then cnt<=0; so<='1';
elsif clk'event and clk='1' then
if cnt<160 then cnt<=cnt+1; else cnt<=0; end if;
if cnt<16 then so<='0'; -- start bit
elsif cnt<32 then so<=di(0); -- LSB
elsif cnt<48 then so<=di(1);
elsif cnt<64 then so<=di(2);
elsif cnt<80 then so<=di(3);
elsif cnt<96 then so<=di(4);
elsif cnt<112 then so<=di(5);
elsif cnt<128 then so<=di(6);
elsif cnt<144 then so<=di(7); -- MSB
elsif cnt<160 then so<='1'; -- stop bit
end if;
end if;
end process;
clr <='1' when cnt=160 else '0';
end behave;
```

```
library ieee;
use ieee.std_logic_1164.all;
entity my_tx is
port (clk: in bit;
      di : in bit_vector(7 downto 0);
      so : out bit);
end;
architecture behave of my_tx is
signal cnt : integer range 0 to 160;
signal clr : bit;
begin
process (clk)
begin
if clr='1' then cnt<=0; so<='1';
elsif clk'event and clk='1' then
if cnt<160 then cnt<=cnt+1; else cnt<=0; end if;
if cnt<16 then so<='0'; -- start bit
elsif cnt<32 then so<=di(0); -- LSB
elsif cnt<48 then so<=di(1);
elsif cnt<64 then so<=di(2);
elsif cnt<80 then so<=di(3);
elsif cnt<96 then so<=di(4);
elsif cnt<112 then so<=di(5);
elsif cnt<128 then so<=di(6);
elsif cnt<144 then so<=di(7); -- MSB
elsif cnt<160 then so<='1'; -- stop bit
end if;
end if;
end process;
clr <='1' when cnt=160 else '0';
end behave;
```

דיאגרמות:



קולט – Rx – Receiver

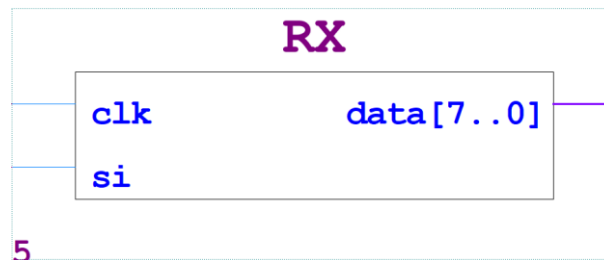
תפקיד התוכנית – לקבל מידע טורי בתקן RS232 ולהוציא מידע מקבילי בתנאי שתקין.

עקרון כתיבת התוכנית Rx

1. נגדיר process ראשון לזיהוי ירידת '1' ל-'0' בכניסת si שמשמעותו ממעבר מ-stop bit ל-start bit. ונעלה דגל (משתנה מסוג סיגנל) START ל-'1'.
 2. ב-process שני נספור בכל עליית שעון עד 160 מחזורי ונדגום באמצע זמן שליחת כל סיבית את המידע.
 3. לקראת סוף קבלת המידע לתוך משתנה פנימי Temp, נבדוק תקינות ורק אם תקין נעביר את Temp למוצא Do.
- בתנאי שהדגל START = '1'

זוהי תוכנית ראשונה עם שני process, מכיוון שצריך לבדוק si' event וירידה לאפס ואי אפשר לבצע שני events ב-process אחד.

מידע תקין – מידע שעוטף אותו start bit = '0' ו-stop bit = '0'.



סמל הרכיב:

```

library ieee;
use ieee.std_logic_1164.all;
entity rx is
port ( clk ,si : in bit;
      data : out bit_vector(7 downto 0));
end ;
architecture behave of rx is
signal start : bit; -- flag fpr starting recieving --
signal cnt: integer range 0 to 160; -- 16X10
signal clr,start_bit,stop_bit : bit;
signal tmp : bit_vector(7 downto 0);
begin
-----
process ( clr, si )
begin
if clr='1' then start<='0';
elsif si'event and si='0' then start<='1'; --- checking start bit
end if;
end process;
-----
process ( clr,clk)
begin
if clr='1' then cnt<=0;
elsif clk'event and clk='1' then
if start='1' then
if cnt<160 then cnt<=cnt+1; else cnt<=0; end if;
if cnt=8 then start_bit<=si;
elsif cnt=24 then tmp(0)<=si; -- LSB
elsif cnt=40 then tmp(1)<=si;
elsif cnt=56 then tmp(2)<=si;
elsif cnt=72 then tmp(3)<=si;
elsif cnt=88 then tmp(4)<=si;
elsif cnt=104 then tmp(5)<=si;
elsif cnt=120 then tmp(6)<=si;
elsif cnt=136 then tmp(7)<=si;
elsif cnt=152 then stop_bit<=si;
elsif cnt=155 and start_bit='0' and stop_bit='1' then data<=tmp;
end if;
end if;
end if;
end process;
clr<='1' when cnt=160 else '0';
end behave;

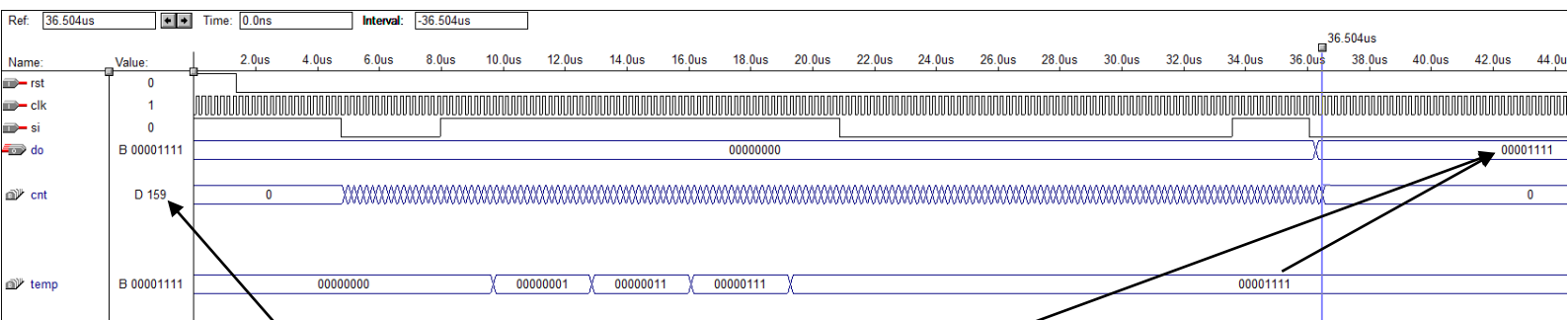
```

```

library ieee;
use ieee.std_logic_1164.all;
entity rx is
port ( clk ,si : in bit;
      data : out bit_vector(7 downto 0));(
end;
architecture behave of rx is
signal start : bit; -- flag fpr starting recieving --
signal cnt: integer range 0 to 160; -- 16X10
signal clr,start_bit,stop_bit : bit;
signal tmp : bit_vector(7 downto 0);
begin
-----
process ( clr, si )
begin
if clr='1' then start<='0';
elsif si'event and si='0' then start<='1'; --- checking start bit
end if;
end process;
-----
process ( clr,clk)
begin
if clr='1' then cnt<=0;
elsif clk'event and clk='1' then
if start='1' then
if cnt<160 then cnt<=cnt+1; else cnt<=0; end if;
if cnt=8 then start_bit<=si;
elsif cnt=24 then tmp(0)<=si; -- LSB
elsif cnt=40 then tmp(1)<=si;
elsif cnt=56 then tmp(2)<=si;
elsif cnt=72 then tmp(3)<=si;
elsif cnt=88 then tmp(4)<=si;
elsif cnt=104 then tmp(5)<=si;
elsif cnt=120 then tmp(6)<=si;
elsif cnt=136 then tmp(7)<=si;
elsif cnt=152 then stop_bit<=si;
elsif cnt=155 and start_bit='0' and stop_bit='1' then data<=tmp;
end if;
end if;
end if;
end process;
clr<='1' when cnt=160 else '0';
end behave;

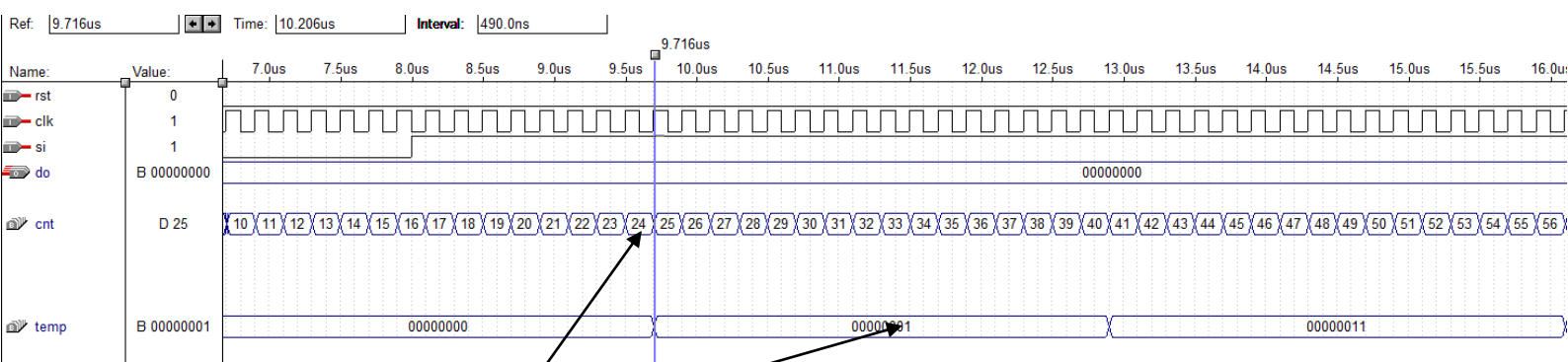
```

דיאגרמות:



ספרנו עד 160 ויש במוצא את המידע שנקלט ממשנתה Temp, כלומר כל הסיביות הועברו ונקלטו והמידע הנו תקין.

בזום אין:



מתווסף '1' לוגי ב- $cnt = 24$ במשתנה Temp

כאן אפשר לראות שדגמנו ב- $cnt = 24$ כמו בקוד התוכנית כשכניסת $si = '1'$ אפשר לראות שהתווסף '1' לוגי במשתנה Temp ואכן יש עליית שעון.

קוד התוכנית שמבצע את הפעולה: `-- LSB` ; `temp(0) <= si;` ; `elsif cnt=24 then`

מחלק תדר – BaudRate
תפקיד התוכנית – לקבל תדר של 50MHz ולהוציא תדר של 153,600Hz.
 עיקרון התוכנית:

$$\frac{1}{\frac{153,000}{1}} \times 0.8 = 40000 \text{ כלומר}$$

רוצים חצי זמן מחזור מכיוון שרוצים D.C של 80%

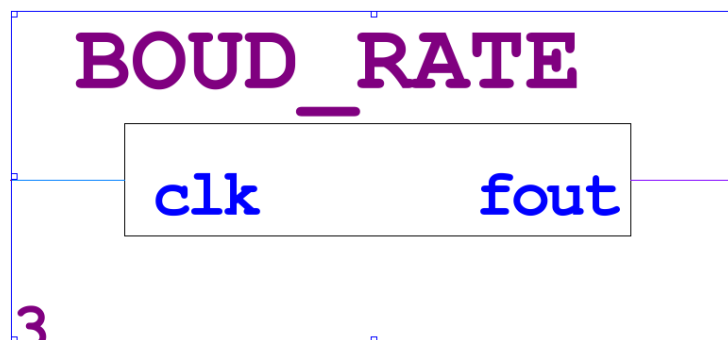
אנו צריכים תדר גבוה פי 16 מקצב קבלה/שליחת מידע למחשב שהוא 9600BPS.

$$9600 \times 16 = 153,600$$

בחרנו בקצב זה מכיוון שזהו הקצב הנפוץ ביותר לעבודה עם תקן RS232.

BPS = Bits Per Second

סמל רכיב: BaudRate -

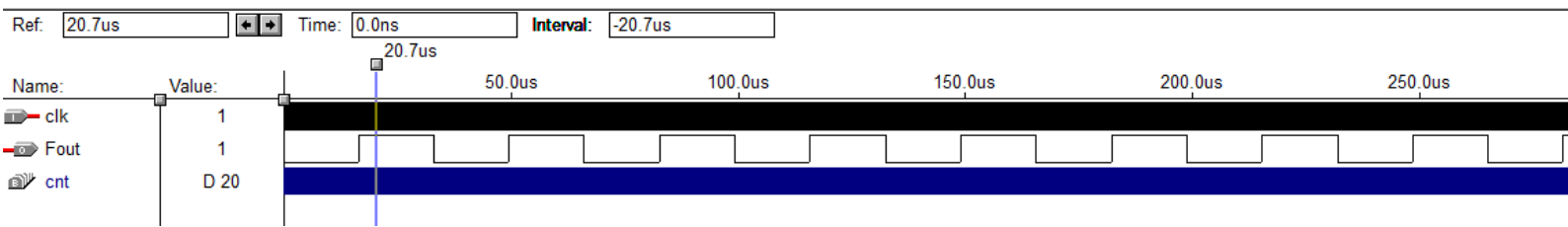


קוד התוכנית:

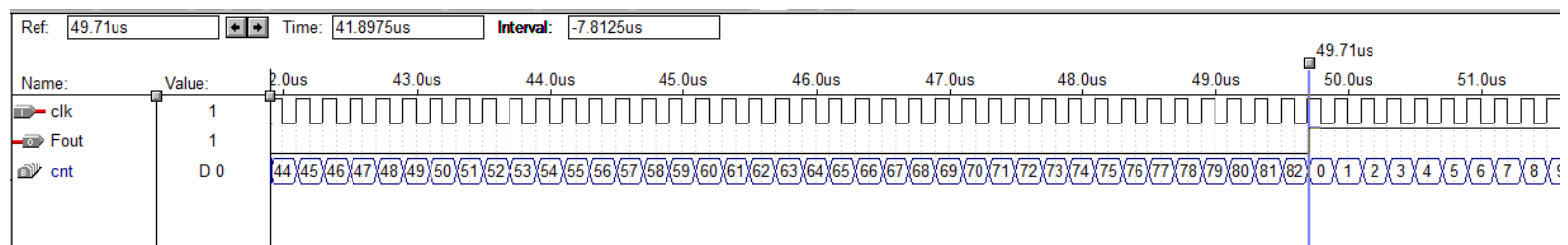
```
library ieee;
use ieee.std_logic_1164.all;
entity BOUD_RATE is
port (clk : in bit;
      fout : buffer bit);
end;
architecture behave of BOUD_RATE is
signal cnt : integer range 0 to 162;
begin
process (clk)
begin
if clk'event and clk='1' then
if cnt<162 then --
cnt<=cnt+1;
else
cnt<=0;
fout<=not fout;
end if;
end if;
end process;
end behave;
```

```
library ieee;
use ieee.std_logic_1164.all;
entity BOUD_RATE is
port (clk : in bit;
      fout : buffer bit);
end;
architecture behave of BOUD_RATE is
signal cnt : integer range 0 to 162; --
begin
process (clk)
begin
if clk'event and clk='1' then
if cnt<162 then --
cnt<=cnt+1;
else
cnt<=0;
fout<=not fout;
end if;
end if;
end process;
end behave;
```

דיאגרמות :



בזום אין :

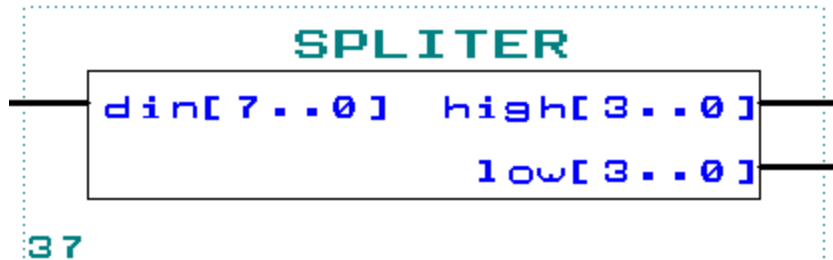


אפשר לראות שברגע שמשתנה cnt ספר עד 82 במוצא יש אחד וככה כל פעם הוא מחליף מצב קודם, ויוצא תדר של 153,600.

מחלק מפענח תצוגה – Splitter

תפקיד התוכנית - לחלק את מפענח התצוגה לחצי, ככה שחצי אחד מראה מספרים מאפס עד 3 והחצי השני מראה מספרים מארבע עד 7 (high, low). בערכה שאנו עובדים איתה יש 2 מפענחי תצוגות, לכן יש את המחלק שמפענח אחד יציג מספר בהקס נמוך ואחד יציג מספר בהקס גבוה.

סמל הרכיב:



קוד התוכנית:

```
library ieee;
use ieee.std_logic_1164.all;
entity splitter is
port (    din : in bit_vector(7 downto 0);
        high,low : out bit_vector(3 downto 0));
end;
architecture behave of splitter is
begin
high<=din(7 downto 4);
low<=din(3 downto 0);




end behave;
```

```
library ieee;
use ieee.std_logic_1164.all;
entity splitter is
port (    din : in bit_vector(7 downto 0);
        high,low : out bit_vector(3 downto 0));
end;
architecture behave of splitter is
begin
high<=din(7 downto 4);
low<=din(3 downto 0);

end behave;
```




דיאגרמה:

Ref: 134.0ns Time: 0.0ns Interval: -134.0ns

Name:	Value:	100.0ns
 din	H F4	F4
 high	H F	F
 low	H 4	4

בבינארי:

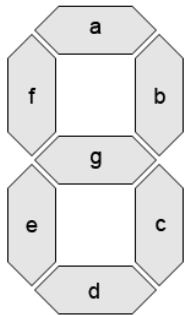
Ref: 162.0ns Time: 9.9ns Interval: -152.1ns

Name:	Value:	100.0ns
 din	B 11110100	11110100
 high	B 1111	1111
 low	B 0100	0100

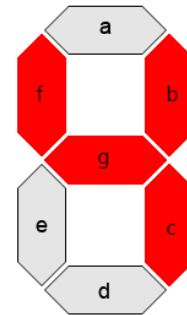
אפשר לראות שכשיש F4 ב – din כלומר 11110100 בבינארי, זה מחולק לשני חצאים, high מקבל את המספר הגבוה כלומר 1111 שזה F בהקס (hex) ו low מקבל את המספר הנמוך כלומר 0100 שהוא 4.

מפענח תצוגה – 7 Segment Decoder (bin2hex)

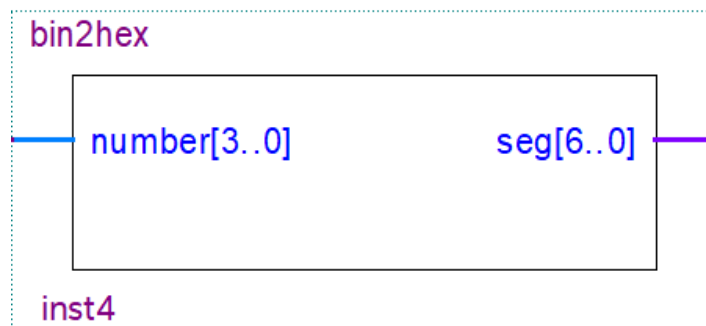
תפקיד התוכנית – לקבל מספר בבינארי בכל ירידת שעות (כלומר כשיש '0' לוגי) ולפי זה להפעיל פסי תצוגה בשביל להציג את התצוגה. לכל פס יש שם – a,b,c,d,e,f,g. אם המספר שהתקבל הוא 4 אז רק הפסים f,g,b,c יודלקו וככה יוצג המספר 4.



פס תצוגה ששמו הוא b.



סמל הרכיב:



קוד התוכנית:

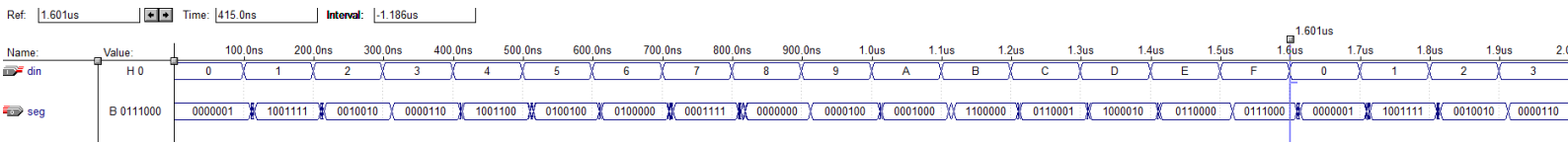
```
library ieee;
use ieee.std_logic_1164.all;
entity bin2hex is
port ( number : in integer range 0 to 15;
      seg : out bit_vector ( 6 downto 0 ) );
end;
architecture behave of bin2hex is
signal temp : bit_vector ( 6 downto 0 );
begin
with number select
temp <= "1111110" when 0,
      "0110000" when 1,
      "1101101" when 2,
      "1111001" when 3,
      "0110011" when 4,
      "1011011" when 5,
      "1011111" when 6,
      "1110000" when 7,
      "1111111" when 8,
      "1111011" when 9,
      "1110111" when 10,
      "0011111" when 11,
      "1001110" when 12,
      "0111101" when 13,
      "1001111" when 14,
      "1000111" when 15;

seg<= not temp;
end behave;
```

```
library ieee;
use ieee.std_logic_1164.all;
entity bin2hex is
port ( number : in integer range 0 to 15;
      seg : out bit_vector ( 6 downto 0 ) );
end;
architecture behave of bin2hex is
signal temp : bit_vector ( 6 downto 0 );
begin
with number select
temp <= "1111110" when 0,
      "0110000" when 1,
      "1101101" when 2,
      "1111001" when 3,
      "0110011" when 4,
      "1011011" when 5,
      "1011111" when 6,
      "1110000" when 7,
      "1111111" when 8,
      "1111011" when 9,
      "1110111" when 10,
      "0011111" when 11,
      "1001110" when 12,
      "0111101" when 13,
      "1001111" when 14,
      "1000111" when 15;
```

```
seg<= not temp;
end behave;
```

דיאגרמה:



אפשר לראות בדיאגרמה שלפי הקוד כל פעם שרוצים להציג מספר אז במוצא seg אפשר לראות אילו פסי תצוגה עובדים בשביל להציג את המספר (הפסים עובדים ב – '0' לוגי). לדוגמה ב – 0 אפשר לראות במוצא – 0000001. כלומר כל פסי התצוגה עובדים חוץ מ – g, וזה יוצר 0.

בגלל שאי אפשר להציג מספרים דו – ספרתיים במפענח תצוגה יחיד נשתמש בסיס hex בשביל להציג אותם.

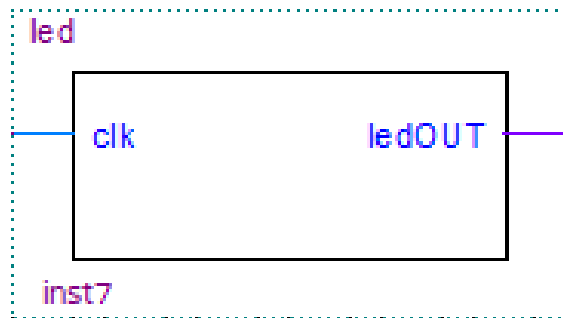
$A = 10, B = 11, C = 12, D = 13, E = 13, F = 15$

כלומר נציג את האותיות האלו על המפענח תצוגה.

LED - נשימה

תפקיד התוכנית - להראות שהמערכת עובדת לפי הבהוב של LED על האלטרנה כל שניה על ידי הוצאת תדר של 1Hz. מקבל שעון של 50MHz.

סמל הרכיב:



קוד התוכנית:

```
library ieee;
use ieee.std_logic_1164.all;

entity led is
port(clk : in bit;
      ledOUT : out STD_LOGIC);
end;

architecture behave of led is
    signal pulse : STD_LOGIC := '0';
    signal count : integer range 0 to 50000000 := 0;
begin
    process (clk)
    begin
        if clk'event and clk = '1' then
            if(count = 49999999) then
                count <= 0;
                pulse <= not pulse;
            else
                count <= count + 1;
            end if;
        end if;
    end process;

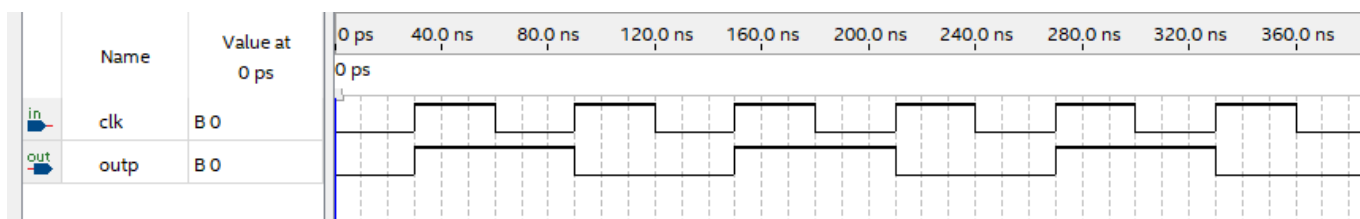
    ledOUT <= pulse;
end behave;
```

```
library ieee;
use ieee.std_logic_1164.all;

entity led is
port(clk : in bit;
      ledOUT : out STD_LOGIC);
end;

architecture behave of led is
    signal pulse : STD_LOGIC := '0';
    signal count : integer range 0 to 50000000 := 0;
begin
    process (clk)
    begin
        if clk'event and clk = '1' then
            if(count = 49999999) then
                count <= 0;
                pulse <= not pulse;
            else
                count <= count + 1;
            end if;
        end if;
    end process;

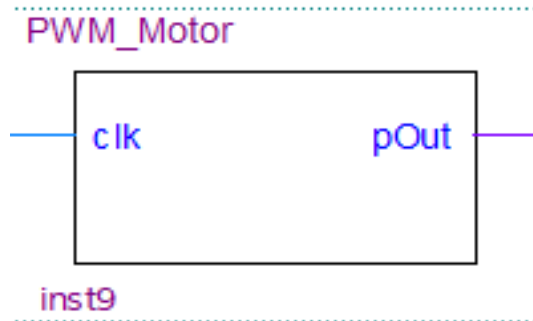
    ledOUT <= pulse;
end behave;
```



PWM_Motor

תפקיד התוכנית – להוציא תדר של 1KHz ב-D.C של 80% בשביל מנוע ה-DC.

סמל הרכיב:



קוד התוכנית:

```
library ieee;
use ieee.std_logic_1164.all;
entity PWM_Motor is
port ( clk: in bit;
      pout : out STD_LOGIC);
end;
architecture RTL of PWM_Motor is
    signal pulse : STD_LOGIC := '1';
    signal count : integer range 1 to 50000 := 1;
begin
    process (clk)
    begin
        if clk'event and clk = '1' then
            count <= count + 1;

            if(count < 40000) then
                pulse <= '1';
            end if;

            if(count > 40000) then
                pulse <= '0';
            end if;

            if(count = 50000) then
                count <= 1;
            end if;

            pout <= pulse;
        end process;
    end RTL;
```

```
library ieee;
use ieee.std_logic_1164.all;
entity PWM_Motor is
port ( clk: in bit;
      pOut : out STD_LOGIC);
end;
architecture RTL of PWM_Motor is
    signal pulse : STD_LOGIC := '1';
    signal count : integer range 1 to 50000 := 1;
begin
    process (clk)
    begin
        if clk'event and clk = '1' then
            count <= count + 1;

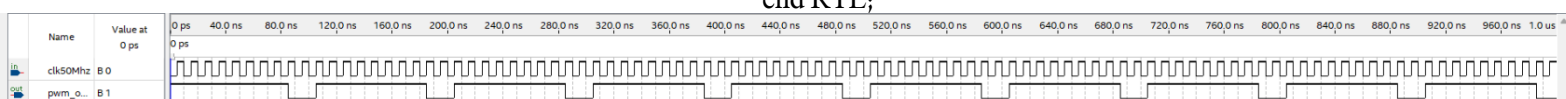
            if(count < 40000) then
                pulse <= '1';
            end if;

            if(count > 40000) then
                pulse <= '0';
            end if;

            if(count = 50000) then
                count <= 1;
            end if;

            pOut <= pulse;
        end process;
    end RTL;
```

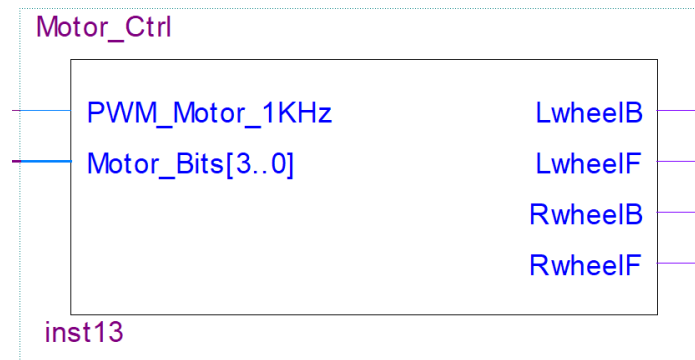
דיאגרמה:



PWM_Ctrl

תפקיד התוכנית – להכניס את התדר של PWM_Motor לגלגלים לפי פקודה מהאפליקציה ולהזיז את הרכב.

סמל הרכיב:



קוד התוכנית:

```
library ieee;
use ieee.std_logic_1164.all;
entity Motor_Ctrl is
port(PWM_Motor_1KHz: in bit;
      Motor_Bits: in bit_vector (3 downto 0);
      LwheelB: buffer bit;
      LwheelF: buffer bit;
      RwheelB: buffer bit;
      RwheelF: buffer bit);
end;
architecture behave of Motor_Ctrl is
begin
----- Both wheels
process(PWM_Motor_1KHz,Motor_Bits)
begin
----- no movement

if Motor_Bits = "0000" then LwheelB <= '0'; LwheelF <= '0'; RwheelB <= '0'; RwheelF <= '0'; end if;
----- forward

if Motor_Bits = "0101" then LwheelB <= '0'; LwheelF <= PWM_Motor_1KHz; RwheelB <= '0'; RwheelF <=
PWM_Motor_1KHz; end if;
----- backward

if Motor_Bits = "1010" then LwheelB <= PWM_Motor_1KHz; LwheelF <= '0'; RwheelB <=
PWM_Motor_1KHz; RwheelF <= '0'; end if;
----- right

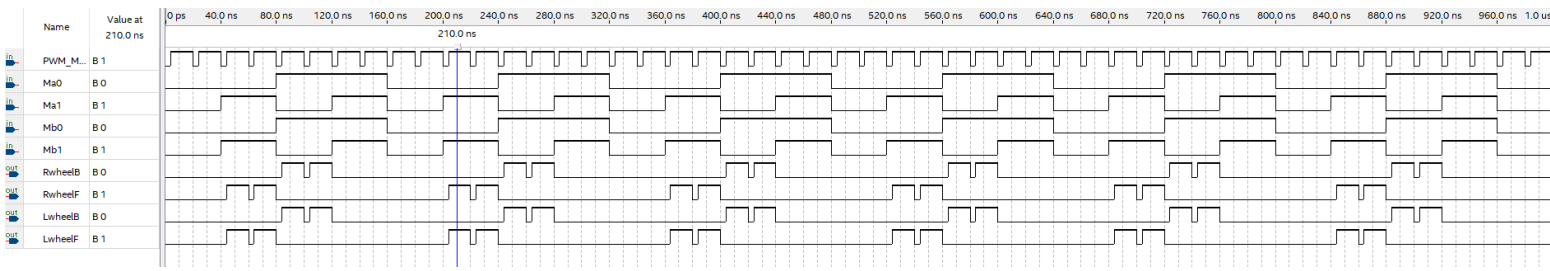
if Motor_Bits = "0110" then LwheelB <= '0'; LwheelF <= PWM_Motor_1KHz; RwheelB <=
PWM_Motor_1KHz; RwheelF <= '0'; end if;

----- left
if Motor_Bits = "1001" then LwheelB <= PWM_Motor_1KHz; LwheelF <= '0'; RwheelB <= '0'; RwheelF <=
PWM_Motor_1KHz; end if;

----- no movement
if Motor_Bits = "1111" then LwheelB <= '0'; LwheelF <= '0'; RwheelB <= '0'; RwheelF <= '0'; end if;

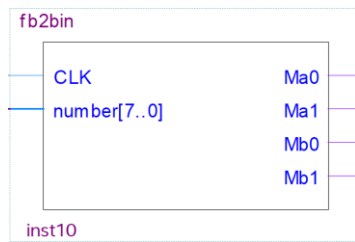
end process;
end behave;
```

דיאגרמה:



פיירבייס לבינארי – FB2Bin (לא משתמשים)

תפקיד התוכנית - לחבר בין המידע המקבילי של הפיירבייס למידע שנכנס לתוכנית .PWM_Ctrl



סמל הרכיב:

```
library ieee;
use ieee.std_logic_1164.all;

entity fb2bin is
port ( CLK: IN BIT;
      number : in bit_vector (7 downto 0);
      Ma0: buffer bit;
      Ma1: buffer bit;
      Mb0: buffer bit;
      Mb1: buffer bit);
end;

architecture behave of fb2bin is
    SIGNAL DATA : bit_vector (7 downto 0);
begin
    PROCESS (CLK)
    BEGIN
        IF CLK'EVENT AND CLK='1' THEN
            DATA<=NUMBER;
        END IF;
    END PROCESS;

    process (CLK)
    begin
        IF CLK'EVENT AND CLK='1' THEN
            IF DATA="00000000" THEN
                MA0<='0'; MA1<='0'; MB0<='0'; MB1<='0';
            ELSIF DATA="00000001" THEN
                MA0<='0'; MA1<='1'; MB0<='0'; MB1<='1';
            ELSIF DATA="00000010" THEN
                MA0<='1'; MA1<='0'; MB0<='1'; MB1<='0';
            ELSIF DATA="00000011" THEN
                MA0<='1'; MA1<='0'; MB0<='0'; MB1<='1';
            ELSIF DATA="00000100" THEN
                MA0<='0'; MA1<='1'; MB0<='1'; MB1<='0';
            END IF;
        END IF;
    end process;
end behave;
```

```
library ieee;
use ieee.std_logic_1164.all;
```

קוד התוכנית:

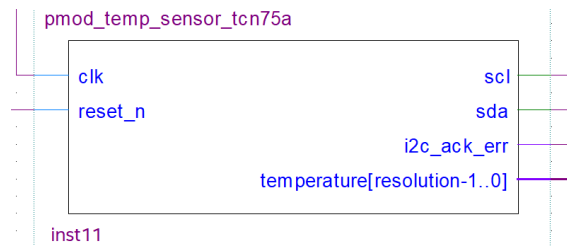
```
entity fb2bin is
port ( CLK: IN BIT;
      number : in bit_vector (7 downto 0);
      Ma0: buffer bit;
      Ma1: buffer bit;
      Mb0: buffer bit;
      Mb1: buffer bit);
end;

architecture behave of fb2bin is
    SIGNAL DATA : bit_vector (7 downto 0);
begin
    PROCESS (CLK)
    BEGIN
        IF CLK'EVENT AND CLK='1' THEN
            DATA<=NUMBER;
        END IF;
    END PROCESS;

    process (CLK)
    begin
        IF CLK'EVENT AND CLK='1' THEN
            IF DATA="00000000" THEN
                MA0<='0'; MA1<='0'; MB0<='0';
            ELSIF DATA="00000001" THEN
                MA0<='0'; MA1<='1'; MB0<='0';
            ELSIF DATA="00000010" THEN
                MA0<='1'; MA1<='0'; MB0<='1';
            ELSIF DATA="00000011" THEN
                MA0<='1'; MA1<='0'; MB0<='0';
            ELSIF DATA="00000100" THEN
                MA0<='0'; MA1<='1'; MB0<='1';
            END IF;
        END IF;
    end process;
end behave;
```


pmod temp sensor tcn75a

תפקיד התוכנית - לשלוח נתונים בני 9 סיביות בשביל להציג טמפרטורה הנקלטת על ידי חיישן הטמפרטורה LM75.



סמל הרכיב:

Parameter	Value	Type
sys_clk_freq	50000000	Signed Integer
resolution	9	Signed Integer
temp_sensor_addr	1001000	Unsigned Binary

--בשם DIGIKEY הקוד לקוח מאתר

קוד התוכנית :

```
LIBRARY ieee;  
USE ieee.std_logic_1164.all;
```

של החיישון בדיאגרמת בלוקים symbol מוצג בבלוק ליד -- IS ENTITY pmod_temp_sensor_tcn75a

```
GENERIC(  
    sys_clk_freq : INTEGER := 50_000_000; -- כניסת שעון מערכת  
    resolution    : INTEGER := 9;          -- רזולוציה רצויה של הטמפרטורה בביט  
    temp_sensor_addr : STD_LOGIC_VECTOR(6 DOWNTO 0) := "1001000"; -- של החיישן I2C כתובת ה--  
PORT(  
    clk : IN STD_LOGIC; -- שעון מערכת  
    reset_n : IN STD_LOGIC; -- מאפס את החיישון שיהיה לו הפסקה מידי פעם, בגובה החיישון ימדוד  
    scl : INOUT STD_LOGIC; -- I2C serial clock  
    sda : INOUT STD_LOGIC; -- I2C serial data  
    i2c_ack_err : OUT STD_LOGIC; -- מודיע כשיש שגיאה בחיישון משמש כדגל--  
    temperature : OUT STD_LOGIC_VECTOR(resolution-1 DOWNTO 0); -- מידע הטמפרטורה בסיבית--  
END pmod_temp_sensor_tcn75a;
```

ARCHITECTURE behavior OF pmod_temp_sensor_tcn75a IS

```
TYPE machine IS(start, set_resolution, set_reg_pointer, read_data, output_result); -- מצבים של החיישון--  
SIGNAL state : machine; -- מכונת מצבים--  
SIGNAL config : STD_LOGIC_VECTOR(7 DOWNTO 0); -- ערך כדי להגדיר את רישום תצורת החיישון--  
SIGNAL i2c_ena : STD_LOGIC; -- משתמש עזר להפעלה i2c--  
SIGNAL i2c_addr : STD_LOGIC_VECTOR(6 DOWNTO 0); -- משתמש עזר לכתובת החיישון i2c--  
SIGNAL i2c_rw : STD_LOGIC; -- משתמש עזר לפקודה של קריאה וכתיבה i2c--  
SIGNAL i2c_data_wr : STD_LOGIC_VECTOR(7 DOWNTO 0); -- כותב מידע i2c--  
SIGNAL i2c_data_rd : STD_LOGIC_VECTOR(7 DOWNTO 0); -- קורא מידע i2c--  
SIGNAL i2c_busy : STD_LOGIC; -- משתמש עזר כשהקו תפוס i2c--  
SIGNAL busy_prev : STD_LOGIC; -- I2C ערך קודם כאשר הקו תפוס בתקשורת--  
SIGNAL temp_data : STD_LOGIC_VECTOR(15 DOWNTO 0); -- חוצץ מידע הטמפרטורה--
```

COMPONENT i2c_master IS

GENERIC(

input_clk : INTEGER; --מידע כניסת השעון

bus_clk : INTEGER); --תרוץ SCL מהירות בה רגל

PORT(

clk : IN STD_LOGIC; --שעון מערכת

reset_n : IN STD_LOGIC; --ריסט בקצב נמוך כדי לא להעמיס על החיישן

ena : IN STD_LOGIC; --מאפשר פעולת החיישן

addr : IN STD_LOGIC_VECTOR(6 DOWNTO 0); --כתובת העבד של החיישן

rw : IN STD_LOGIC; --זה כותב '1' זה קורא '0'

data_wr : IN STD_LOGIC_VECTOR(7 DOWNTO 0); --I2C מידע לרשום לעבד

busy : OUT STD_LOGIC; --מודיע על העברת מידע בפעולתה

data_rd : OUT STD_LOGIC_VECTOR(7 DOWNTO 0); --I2C קריאת מידע מהעבד

ack_error : BUFFER STD_LOGIC; --דגל מידע שגוי מהחיישן/עבד

sda : INOUT STD_LOGIC; --I2C יציאת מידע סדרתי מהחיישן בתקשורת

scl : INOUT STD_LOGIC); --I2C יציאת שעון סדרתי מהחיישן בתקשורת

END COMPONENT;

BEGIN

--I2C_MASTER מאתחל את התוכנית

i2c_master_0: i2c_master

GENERIC MAP(input_clk => sys_clk_freq, bus_clk => 400_000)

PORT MAP(clk => clk, reset_n => reset_n, ena => i2c_ena, addr => i2c_addr,

rw => i2c_rw, data_wr => i2c_data_wr, busy => i2c_busy,

data_rd => i2c_data_rd, ack_error => i2c_ack_err, sda => sda,

scl => scl);

--מגדיר את סיביות הרזולוציה עבור ערך רישום תצורת החיישן

WITH resolution SELECT

config <= "00100000" WHEN 10, --סיביות 10

"01000000" WHEN 11, --סיביות 11

"01100000" WHEN 12, --סיביות 12

"00000000" WHEN OTHERS; --סיביות(ברירת מחדל) 9

PROCESS(clk, reset_n)

VARIABLE busy_cnt : INTEGER RANGE 0 TO 2 := 0; --סופר את מעברי האות התפוסים במהלך

העברה אחת

VARIABLE counter : INTEGER RANGE 0 TO sys_clk_freq/10 := 0; --לחכות לפני ms100 סופר

תקשורת

BEGIN

IF(reset_n = '0') THEN --מופעל reset_n כאשר

counter := 0; --מאפס את המונה של העצירה

i2c_ena <= '0'; --מאפס i2c enable

busy_cnt := 0; --מאפס busy counter

temperature <= (OTHERS => '0'); --מאפס את המידע של הטמפרטורה

state <= start; --מאפס את מכונת המצבים

ELSIF(clk'EVENT AND clk = '1') THEN --בודק אם השעון בגבוה

CASE state IS --מכונת מצבים

--להפעלה לפני התקשורת ms100 לוחותן לחיישן טמפ'

WHEN start =>

IF(counter < sys_clk_freq/10) THEN --עוד לא ספר 100ms

counter := counter + 1; --מוסיף למונה

ELSE --הגיע ל 100ms

counter := 0; --מאפס מונה

state <= set_resolution; --ממשיך לקביעת הרזולוציה של החיישן

END IF;

קביעת הרזולוציה של החיישן--

```
WHEN set_resolution =>
  busy_prev <= i2c_busy;          --i2c לוכד את הערך של האות התפוס הקודם של-
  IF(busy_prev = '0' AND i2c_busy = '1') THEN --i2c busy לגבוה עבר
    busy_cnt := busy_cnt + 1;      --הפך מנמוך לגבוה במהלך ההעברה busy סופר את הפעמים-
  END IF;
  CASE busy_cnt IS
    WHEN 0 =>                      --עוקב אחר איזו פקודה אנחנו נמצאים busy_cnt
      i2c_ena <= '1';             --אין שום פקודה נועחלת כרגע-
      i2c_addr <= temp_sensor_addr; --התחל את העברה-
      i2c_rw <= '0';              --'קביעת הכתובת של החיישן טפמ-
      i2c_data_wr <= "00000001";  --הפקודה '1' זה כתיבה-
    WHEN 1 =>                      --Register Configuration-הגדר את מצביע הרשמה ל-
      i2c_data_wr <= config;      --גבוה אחר כך פקודה 1 ננעלת ואפשר להמשיך לפקודה 2 busy קודם-
    WHEN 2 =>                      --Configuration Register-כתוב את ערך התצורה החדש ל-
      i2c_ena <= '0';            --גבוה ולכן פקודה 2 ננעלת busy עכשיו-
      IF(i2c_busy = '0') THEN     --מופעל ניתן להפסיק את העברה לאחר פקודה 2 deassert כאשר-
        busy_cnt := 0;            --העברה הושלמה-
        state <= set_reg_pointer; --להעברה הבאה busy_cnt מאפס את-
        state <= set_reg_pointer; --ממשיך להגדרת מצביע הרשמה לקריאות נתונים-
      END IF;
    WHEN OTHERS => NULL;
  END CASE;
```

הגדר את מצביע האוגר למאגר טמפרטורת הסביבה--

```
WHEN set_reg_pointer =>
  busy_prev <= i2c_busy;          --i2c הקודם של busy ללכוד את הערך של האות-
  IF(busy_prev = '0' AND i2c_busy = '1') THEN --i2c busy עלה
    busy_cnt := busy_cnt + 1;      --הפך מנמוך לגבוה במהלך ההעברה busyסופר את הפעמים ש-
  END IF;
  CASE busy_cnt IS
    WHEN 0 =>                      --עוקב אחר איזו פקודה אנחנו נמצאים busy_cnt
      i2c_ena <= '1';             --אין פקודה נעולה כרגע-
      i2c_addr <= temp_sensor_addr; --התחלת העברת המידע-
      i2c_rw <= '0';              --הגדרת הכתובת של החיישן-
      i2c_data_wr <= "00000000";  --פקודה '1' זה כתיבה-
    WHEN 1 =>                      --Register ל-Ambient Temperature Register-הגדר את מצביע ה-
      i2c_ena <= '0';            --גבוה: פקודה אחת נעולה busy 1st-
      IF(i2c_busy = '0') THEN     --אפשר לעצור את ההעברה לאחר פקודה 1 deassert-
        busy_cnt := 0;            --העברה הושלמה-
        state <= read_data;      --להעברה הבאה busy_cnt מאפס את-
        state <= read_data;      --ממשיך לקריאת מידע-
      END IF;
    WHEN OTHERS => NULL;
  END CASE;
```

--קורא את הטמפ' של הסביבה--

WHEN read_data =>

busy_prev <= i2c_busy; --i2c הקודם של busy ללכוד את הערך של האות--

IF(busy_prev = '0' AND i2c_busy = '1') THEN --i2c busy עלה לגבוה

busy_cnt := busy_cnt + 1; --הפך מנמוך לגבוה במהלך ההעברה busy סופר את הפעמים ש--

END IF;

CASE busy_cnt IS --busy_cnt אנחנו נמצאים

WHEN 0 => --אין פקודה נעולה כרגע--

i2c_ena <= '1'; ----התחלת העברת המידע--

i2c_addr <= temp_sensor_addr; ----הגדרת הכתובת של החיישן--

i2c_rw <= '1'; --פקודה '1' זה קריאה--

WHEN 1 => --אחר כך פקודה 1 ננעלת, ניתן להוציא את פקודה 2 : busy קודם--

IF(i2c_busy = '0') THEN --מודיע שמידע הנקרא בפקודה 1 זמין--

temp_data(15 DOWNT0 8) <= i2c_data_rd; --מפקודה 1 MSB מחזיר את--

END IF;

WHEN 2 => --גבוה, פקודה 2 ננעלת busy קודם--

i2c_ena <= '0'; --אפשר לעצור אץ ההעברה לאחר פקודה 2 deassert--

IF(i2c_busy = '0') THEN --מודיע שהמידע בפקודה 2 זמין--

temp_data(7 DOWNT0 0) <= i2c_data_rd; --מפקודה 2 LSB מחזיר את--

busy_cnt := 0; --לפקודה הבאה busy_cnt מאפס את--

state <= output_result; --ממשיך להוצאת הטמפ, לאחר כל התהליך--

END IF;

WHEN OTHERS => NULL;

END CASE;

--הוצאת מידע הטמפ--

WHEN output_result =>

temperature <= temp_data(15 DOWNT0 16-resolution); --רושם את הטמפ, למוצא--

state <= read_data; --מחזיר את המידע הבא מהשלב הקודם של קריאת המידע--

--חוזר להתחלה ומאפס הכל--

WHEN OTHERS =>

state <= start;

END CASE;

END IF;

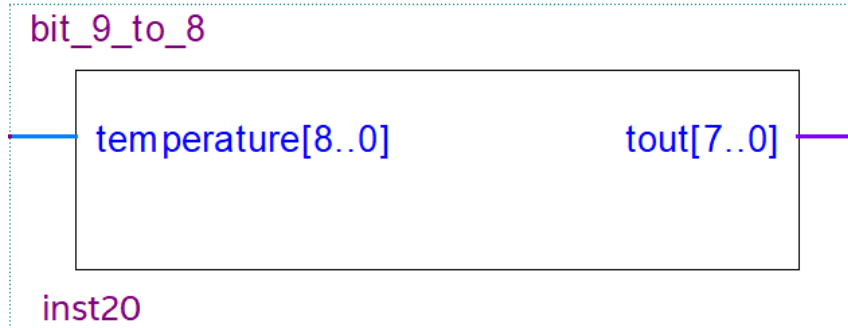
END PROCESS;

END behavior;

8 - Bit 9 to 8 – ממיר 9 סיביות ל- 8

תפקיד התוכנית – לקבל ולהמיר 9 סיביות ל- 8 סיביות בלי לאבד מידע להצגת טמפרטורה.

סמל הרכיב:



קוד התוכנית:

```
library ieee;
use ieee.std_logic_1164.all;
entity bit_9_to_8 is
port(temperature : in STD_LOGIC_VECTOR(8 DOWNTO 0);
      tout: buffer STD_LOGIC_VECTOR( 7 downto 0));
end;

architecture behave of bit_9_to_8 is
begin
process(temperature)
begin
tout(0) <= temperature(0);
tout(1) <= temperature(1);
tout(2) <= temperature(2);
tout(3) <= temperature(3);
tout(4) <= temperature(4);
tout(5) <= temperature(5);
tout(6) <= temperature(6);
tout(7) <= temperature(7);

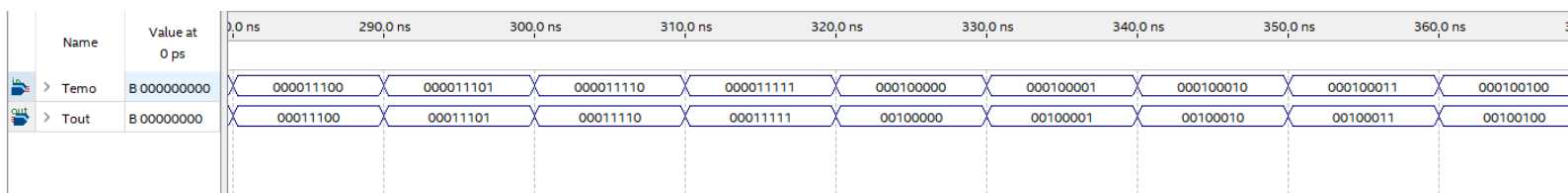
end process;
end behave;
```

```
library ieee;
use ieee.std_logic_1164.all;
entity bit_9_to_8 is
port(temperature : in STD_LOGIC_VECTOR(8
DOWNTO 0);
      tout: buffer
STD_LOGIC_VECTOR( 7 downto 0));
end;

architecture behave of bit_9_to_8 is
begin
process(temperature)
begin
tout(0) <= temperature(0);
tout(1) <= temperature(1);
tout(2) <= temperature(2);
tout(3) <= temperature(3);
tout(4) <= temperature(4);
tout(5) <= temperature(5);
tout(6) <= temperature(6);
tout(7) <= temperature(7);

end process;
end behave;
```

דיאגרמת זמן:



PIN PLANNER

Pin Planner הוא כלי מפתח בחבילת התוכנה של Quartus המאפשרת למעצבים לנהל את הקצאות הפינים של העיצוב שלהם. הוא מספק ממשק גרפי המאפשר למשתמשים להקצות אותות לפינים של מכשיר היעד, להציג ולשנות את הקצאות הפינים ולפתור כל התנגשות שעלולה להתעורר במהלך התהליך.

מתכנן הפינים שימושי במיוחד עבור עיצובי FPGA ו-ASIC, שבהם הקצאות פינים הן קריטיות לתפקוד תקין של המכשיר. בעזרת Pin Planner, מעצבים יכולים להגדיר בקלות את הקישוריות בין הכניסות והיציאות של העיצוב שלהם לבין העולם החיצוני, שהוא חיוני להתממשקות נכונה ושילוב של המכשיר במערכת הגדולה יותר.

בסך הכל, Pin Planner הוא כלי חיוני המפשט את תהליך הקצאת הסיכות ומסייע למעצבים לייעל את הביצועים והאמינות של העיצובים שלהם.

שפת C

היא שפת תכנות מחשב לשימוש כללי. הוא נוצר בשנות ה-70 על ידי דניס ריצ'י, ונשאר בשימוש נרחב ובעל השפעה. לפי התכנון, התכונות של C משקפות בצורה נקייה את היכולות של המעבדים הממוקדים. זה מצא שימוש מתמשך במערכות הפעלה, מנהלי התקנים, ערימות פרוטוקולים, אם כי הולך ופוחת עבור תוכנות יישומים. C משמש בדרך כלל בארכיטקטורות מחשבים הנעות ממחשבי העל הגדולים ביותר ועד למיקרו-בקרים הקטנים ביותר ולמערכות משובצות.

יורש של שפת התכנות B, C פותחה במקור ב-Bell Labs על ידי ריצ'י בין 1972 ל-1973 כדי לבנות כלי עזר הפועלים על יוניקס. הוא הוחל ליישום מחדש של הליבה של מערכת ההפעלה יוניקס. במהלך שנות ה-80, C צברה פופולריות בהדרגה. היא הפכה לאחת משפות התכנות הנפוצות ביותר, כאשר מהדריס C זמינים כמעט לכל ארכיטקטורות המחשב ומערכות ההפעלה המודרניות. C תוקן על ידי ANSI מאז 1989 (ANSI C) ועל ידי ארגון התקינה הבינלאומי (ISO).

C היא שפה פרוצדורלית הכרחית התומכת בתכנות מובנה, היקף משתנה לקסיקלי ורקורסיה, עם מערכת סטטית. הוא תוכנן להידור כדי לספק גישה ברמה נמוכה לזיכרון ולמבנה שפה הממפה ביעילות להוראות מכונה, והכל עם תמיכה מינימלית בזמן ריצה. למרות היכולות הנמוכות שלה, השפה תוכננה כדי לעודד תכנות בין פלטפורמות. ניתן להרכיב תוכנית C תואמת תקנים שנכתבה תוך מחשבה על נייודות עבור מגוון רחב של פלטפורמות מחשב ומערכות הפעלה עם מעט שינויים בקוד המקור שלה.

שפת C מציגה גם את המאפיינים הבאים:

- לשפה יש מספר קטן וקבוע של מילות מפתח, כולל קבוצה מלאה של פרימיטיביות של זרימת בקרה: if/else, for, do/while, while ו-switch. שמות המוגדרים על ידי משתמש אינם מובחנים ממילות מפתח בשום סוג של סימן.
- יש לו מספר רב של אופרטורים אריתמטיים, סיביים ולוגיים: +, =, ++, &, || וכו'.
- ניתן לבצע יותר ממטלה אחת בהצהרה אחת.
- פונקציות:
- ניתן להתעלם מערכי החזרת פונקציות, כאשר אין צורך.
- מצביעי פונקציות ונתונים מאפשרים פולימורפיזם בזמן ריצה אד-הוק.
- לא ניתן להגדיר פונקציות בטווח המילוני של פונקציות אחרות.
- ניתן להגדיר משתנים בתוך פונקציה, עם היקף.
- פונקציה עשויה לקרוא לעצמה, ולכן רקורסיה נתמכת.

- הקלדת הנתונים היא סטטית, אך נאכפת בצורה חלשה; לכל הנתונים יש סוג, אבל המרות מרומזות אפשריות.
- ניתן להגדיר על ידי משתמש (typedef) וסוגים מורכבים.
- סוגי נתונים מצטברים הטרוגניים (struct) מאפשרים לגשת לרכיבי נתונים קשורים ולהקצות אותם כיחידה.
- איחוד הוא מבנה עם חברים חופפים; רק החבר האחרון המאוחד חוקי.
- אינדקס מערך הוא סימון משני, המוגדר במונחים של אריתמטיקה מצביע. בניגוד למבנים, מערכים אינם אובייקטים ממדרגה ראשונה: לא ניתן להקצות אותם או להשוות אותם באמצעות אופרטורים מובנים בודדים. אין מילת מפתח "מערך" בשימוש או בהגדרה; במקום זאת, סוגריים מרובעים מציינים מערכים באופן תחבירי, למשל חודש.
- סוגים מסופרים אפשריים עם מילת המפתח enum. הם ניתנים להמרה הדדית באופן חופשי עם מספרים שלמים.
- מחרוזות אינן סוג נתונים מובחן, אלא מיושמות באופן קונבנציונלי כמערכי תווים עם סיומת אפס.
- גישה ברמה נמוכה לזיכרון המחשב אפשרית על ידי המרת כתובות מכונה למצביעים.
- נהלים (תתי שגרות שאינן מחזירות ערכים) הם מקרה מיוחד של פונקציה, עם ריק מסוג החזרה לא מודפס.
- ניתן להקצות זיכרון לתוכנית עם שיחות לשגרה לספרייה.
- מעבד קדם מבצע הגדרת מאקרו, הכללת קובץ קוד מקור והידור מותנה.
- ישנה צורה בסיסית של מודולריות: ניתן להרכיב קבצים בנפרד ולקשר אותם יחד, עם שליטה באילו פונקציות ואובייקטי נתונים גלויים לקבצים אחרים באמצעות תכונות סטטיות וחיצוניות.
- פונקציונליות מורכבת כגון קלט/פלט, מניפולציה של מחרוזות ופונקציות מתמטיות מואצלות באופן עקבי לשגרות של ספרייה.
- לקוד שנוצר לאחר הקומפילציה יש צרכים פשוטים יחסית בפלטפורמה הבסיסית, מה שהופך אותו למתאים ליצירת מערכות הפעלה ולשימוש במערכות משובצות.

שפות רבות מאוחרות יותר שאלו ישירות או בעקיפין מ-C, כולל C++, #C, מעטפת C של Unix, D, Go, Java, JavaScript (כולל טרנספילרים), Perl, PHP, Objective-C, LPC, Limbo, Julia, SystemVerilog ו-Python, Ruby, Rust, Swift, Verilog (שפות תיאור חומרה). שפות אלו שאבו רבים ממבני השליטה שלהן ותכונות בסיסיות אחרות מ-C. רובן (פייתון הוא חריג דרמטי) מבטאות גם תחביר דומה מאוד ל-C, והן נוטות לשלב את הביטוי וההצהרה המוכרים של C עם הסוג הבסיסי. מערכות, מודלים של נתונים וסמנטיקה שיכולים להיות שונים בתכלית.

קוד בקר ה-ESP32

```
#include <Arduino.h>
#include <WiFi.h>
#include <FirebaseESP32.h>
#include <WiFiMulti.h>

// Change to your Firebase RTDB project ID e.g.
Your_Project_ID.firebaseio.com
#define FIREBASE_HOST "alteratutorial-default-rtdb.europe-
west1.firebaseio.com"
// Change to your Firebase RTDB secret password
#define FIREBASE_AUTH "9U5cGGnuLfcHWckDeVCA5hvMSf1Ki55yUN1VU6BQ"

// legs of ESP that we receive and transmit
#define RXD2 16
#define TXD2 17

#define ledpin 2

// Define WifiMulti Variable
WiFiMulti wifiMulti;

// Define Firebase Data objects
FirebaseData fbdo;
int rx0;
// integers of the firebase
int motors, temperature, leds, packet, ServoMotors;

//Boolean which allows the LED to "breathe"
bool a = false;
// function that shows the system working by flickering the LED of ESP
void blink();

void setup() {
  //Serial - used for printing data, speed of reading from terminal
  Serial.begin(115200);
  //Serial2 - sending and receiving data variable (bps, type of data,
  RX, TX)
  Serial2.begin(9600, SERIAL_8N1, RXD2, TXD2);

  //define the leg of output on the LED
  pinMode(ledpin, OUTPUT);

  //define the wifi networks to search for and use
  wifiMulti.addAP("Ams_2.4GHz", "0523993253A");
  wifiMulti.addAP("ORYAM", "12345678");
  wifiMulti.addAP("upstairs", "10203040");
```

```

// WiFi.scanNetworks will return the number of networks found
int n = WiFi.scanNetworks();
Serial.println("scan done");
if (n == 0) {
    Serial.println("no networks found");
}
else {
    Serial.print(n);
    Serial.println(" networks found");
    for (int i = 0; i < n; ++i) {
        // Print SSID and RSSI for each network found
        Serial.print(i + 1);
        Serial.print(": ");
        Serial.print(WiFi.SSID(i));
        Serial.print(" (");
        Serial.print(WiFi.RSSI(i));
        Serial.print(")");
        Serial.println((WiFi.encryptionType(i) == WIFI_AUTH_OPEN)?
":*"");
        delay(10);
    }

}

if(wifiMulti.run() != WL_CONNECTED) {
    Serial.println("WiFi not connected!");
    delay(1000);
}

while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}
Serial.println("");
Serial.print("Connecting To: ");
Serial.print(WiFi.SSID());
Serial.println("");
Serial.print("Connected! IP address: ");
String ipAddress = WiFi.localIP().toString();
Serial.println(ipAddress);

//connects with Firebase, basically stores Firebase's legacy
authentication credentials
Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
Serial.println("Firebase Connected");

```

```

    //reconnects WIFI if the connection is lost
    Firebase.reconnectWiFi(true);
}
// Always runs when the system is on
void loop() {
    while (Serial2.available()) {
        temperature = Serial2.read();
    }

    // When firebase is ready to receive data

    if (Firebase.ready()) {

        //Firebase reads the integer in the firebase database(the
        realtime console) and then fbdo = firebase data object

        Firebase.setInt(fbdo, "kar98Info/temperature", temperature);

        if (Firebase.getInt(fbdo, "kar98Info/carControl")) {
            packet = fbdo.intData();
        }
    }
    //delay for the ESP so its not getting seizure (overload)
    delay(10);
    // gets the stored bytes from the serial port that are available for
    reading
    if (Serial2.available()) {
        //Serial.write() Prints data to the serial port as human-readable
        ASCII text and this converts FromFB to binary
        Serial2.write(packet);
        delay(10);
    }

    fbdo.clear();
    //LED blink to check if the loop worked
    blink();
}
// if a is false then a is changed to true and them runs a loop once
and turns back to false and that causes the LED to "breath" or turn on
and off a lot
void blink() {
    a = not a;
    if (a)
        digitalWrite(ledpin, HIGH);
    else
        digitalWrite(ledpin, LOW);
}

```

```
#include <WebSocketsServer.h>
#include <WiFi.h>
#include <WiFiUdp.h>
#include "camera_wrap.h"
#include <vector>
#include <WiFiMulti.h>

WiFiMulti wifiMulti;

// #define DEBUG
// #define SAVE_IMG

enum TRACK{
    TRACK_NONE = 0,
    TRACK_FW,
    TRACK_LEFT,
    TRACK_RIGHT,
    TRACK_STOP
};

// const char* ssid = "Ams_2.4GHz";    // <<< change this as yours
// const char* password = "0523993253A"; // <<< change this as yours
//holds the current upload
int cameraInitState = -1;
uint8_t* jpgBuff = new uint8_t[68123];
size_t   jpgLength = 0;
uint8_t camNo=0;
bool clientConnected = false;

//Creating UDP Listener Object.
WiFiUDP UDPServer;
IPAddress addrRemote;
unsigned int portRemote;
unsigned int UDPPort = 6868;
const int RECVLENGTH = 16;
byte packetBuffer[RECVLENGTH];

WebSocketsServer webSocket = WebSocketsServer(86);
String html_home;

const int LED_BUILT_IN      = 4;
const uint8_t TRACK_DUTY    = 100;
const int PIN_SERVO_PITCH   = 12;
// const int PIN_SERVO_YAW   = 2;
```



```

const int PINDC_LEFT_BACK      = 13;
const int PINDC_LEFT_FORWARD  = 15;
const int PINDC_RIGHT_BACK    = 14;
const int PINDC_RIGHT_FORWARD = 2;
const int LEFT_CHANNEL        = 2;
const int RIGHT_CHANNEL       = 3;
const int SERVO_PITCH_CHANNEL = 4;
const int SERVO_YAW_CHANNEL   = 5;
const int SERVO_RESOLUTION    = 16;
unsigned long previousMillisServo = 0;
const unsigned long intervalServo = 10;
bool servoUp = false;
bool servoDown = false;
bool servoRotateLeft = false;
bool servoRotateRight = false;
int posServo = 75;
int PWMTrackHIGH = 138;
int PWMTrackLOW = 138;

void servoWrite(uint8_t channel, uint8_t angle) {
    // regarding the datasheet of sg90 servo, pwm period is 20 ms and
    // duty is 1->2ms
    uint32_t maxDuty = (pow(2, SERVO_RESOLUTION)-1)/10;
    uint32_t minDuty = (pow(2, SERVO_RESOLUTION)-1)/20;
    uint32_t duty = (maxDuty-minDuty)*angle/180 + minDuty;
    ledcWrite(channel, duty);
}

void controlServo(){
    if(servoUp){
        if(posServo>2){
            posServo -= 2;
        }
    }
    if(servoDown){
        if(posServo<180){
            posServo += 2;
        }
    }
    servoWrite(SERVO_PITCH_CHANNEL, posServo);
}

void controlDC(int left0, int left1, int right0, int right1){
    digitalWrite(PINDC_LEFT_BACK, left0);
    if(left1 == HIGH){
        ledcWrite(LEFT_CHANNEL, 255);
    }else{
        ledcWrite(LEFT_CHANNEL, 0);
    }
}

```

```

    }
    digitalWrite(PINDC_RIGHT_BACK, right0);
    if(right1 == HIGH){
        ledcWrite(RIGHT_CHANNEL, 255);
    }else{
        ledcWrite(RIGHT_CHANNEL, 0);
    }
}

void controlDCTrack(int left, int right){
    digitalWrite(PINDC_LEFT_BACK, 0);
    ledcWrite(LEFT_CHANNEL, left);
    digitalWrite(PINDC_RIGHT_BACK, 0);
    ledcWrite(RIGHT_CHANNEL, right);
}

void websocketEvent(uint8_t num, WStype_t type, uint8_t * payload,
size_t length) {

    switch(type) {
        case WStype_DISCONNECTED:
            Serial.printf("[%u] Disconnected!\n", num);
            camNo = num;
            clientConnected = false;
            break;
        case WStype_CONNECTED:
            Serial.printf("[%u] Connected!\n", num);
            clientConnected = true;
            break;
        case WStype_TEXT:
        case WStype_BIN:
        case WStype_ERROR:
        case WStype_FRAGMENT_TEXT_START:
        case WStype_FRAGMENT_BIN_START:
        case WStype_FRAGMENT:
        case WStype_FRAGMENT_FIN:
            Serial.println(type);
            break;
    }
}

std::vector<String> splitString(String data, String delimiter){
    std::vector<String> ret;
    // initialize first part (string, delimiter)
    char* ptr = strtok((char*)data.c_str(), delimiter.c_str());

    while(ptr != NULL) {
        ret.push_back(String(ptr));
    }
}

```

```

        // create next part
        ptr = strtok(NULL, delimiter.c_str());
    }
    return ret;
}

void processUDPData(){
    int cb = UDPServer.parsePacket();

    if (cb) {
        UDPServer.read(packetBuffer, RECVLENGTH);
        addrRemote = UDPServer.remoteIP();
        portRemote = UDPServer.remotePort();

        String strPackage = String((const char*)packetBuffer);
#ifdef DEBUG
        Serial.print("receive: ");
        // for (int y = 0; y < RECVLENGTH; y++){
        //     Serial.print(packetBuffer[y]);
        //     Serial.print("\n");
        // }
        Serial.print(strPackage);
        Serial.print(" from: ");
        Serial.print(addrRemote);
        Serial.print(":");
        Serial.println(portRemote);
#endif
        if(strPackage.equals("whoami")){
            UDPServer.beginPacket(addrRemote, portRemote-1);
            String res = "ESP32-CAM";
            UDPServer.write((const uint8_t*)res.c_str(),res.length());
            UDPServer.endPacket();
            Serial.println("response");
        }else if(strPackage.equals("forward")){
            controlDC(LOW,HIGH,LOW,HIGH);
        }else if(strPackage.equals("backward")){
            controlDC(HIGH,LOW,HIGH,LOW);
        }else if(strPackage.equals("left")){
            controlDC(LOW,LOW,LOW,HIGH);
        }else if(strPackage.equals("right")){
            controlDC(LOW,HIGH,LOW,LOW);
        }else if(strPackage.equals("stop")){
            controlDC(LOW,LOW,LOW,LOW);
        }else if(strPackage.equals("camup")){
            servoUp = true;
        }else if(strPackage.equals("camdown")){
            servoDown = true;
        }else if(strPackage.equals("camstill")){

```

```

        servoUp = false;
        servoDown = false;
    }else if(strPackage.equals("ledon")){
        digitalWrite(LED_BUILT_IN, HIGH);
    }else if(strPackage.equals("ledoff")){
        digitalWrite(LED_BUILT_IN, LOW);
    }else if(strPackage.equals("lefttrack")){
        controlDCTrack(0, PWMTrackHIGH);
    }else if(strPackage.equals("righttrack")){
        controlDCTrack(PWMTrackHIGH, 0);
    }else if(strPackage.equals("fwtrack")){
        controlDCTrack(PWMTrackLOW, PWMTrackLOW);
    }

    memset(packetBuffer, 0, RECVLENGTH);
}

}

void setup(void) {

    Serial.begin(115200);
    Serial.print("\n");
    WiFi.mode(WIFI_STA);
    #ifdef DEBUG
    Serial.setDebugOutput(true);
    #endif

    wifiMulti.addAP("Ams_2.4GHz", "0523993253A");
    wifiMulti.addAP("ORYAM", "12345678");
    wifiMulti.addAP("upstairs", "10203040");

    pinMode(LED_BUILT_IN, OUTPUT);
    digitalWrite(LED_BUILT_IN, LOW);

    pinMode(PINDC_LEFT_BACK, OUTPUT);
    ledcSetup(LEFT_CHANNEL, 100, 8); //channel, freq, resolution
    ledcAttachPin(PINDC_LEFT_FORWARD, LEFT_CHANNEL);
    pinMode(PINDC_RIGHT_BACK, OUTPUT);
    ledcSetup(RIGHT_CHANNEL, 100, 8); //channel, freq, resolution
    ledcAttachPin(PINDC_RIGHT_FORWARD, RIGHT_CHANNEL);

    controlDC(LOW, LOW, LOW, LOW);

    // 1. 50hz ==> period = 20ms (sg90 servo require 20ms pulse, duty
    cycle is 1->2ms: -90=>90degree)
    // 2. resolution = 16, maximum value is 2^16-1=65535

```

```

// From 1 and 2 => -90=>90 degree or 0=>180degree ~ 3276=>6553
ledcSetup(SERVO_PITCH_CHANNEL, 50, 16); //channel, freq, resolution
ledcAttachPin(PIN_SERVO_PITCH, SERVO_PITCH_CHANNEL); // pin, channel
servoWrite(SERVO_PITCH_CHANNEL, posServo);

// ledcSetup(SERVO_YAW_CHANNEL, 50, 16); //channel, freq, resolution
// ledcAttachPin(PIN_SERVO_YAW, SERVO_YAW_CHANNEL); // pin, channel
// servoWrite(SERVO_YAW_CHANNEL, posServo);

cameraInitState = initCamera();

Serial.printf("camera init state %d\n", cameraInitState);

if(cameraInitState != 0){
    return;
}

// WiFi.scanNetworks will return the number of networks found
int n = WiFi.scanNetworks();
Serial.println("scan done");
if (n == 0) {
    Serial.println("no networks found");
}
else {
    Serial.print(n);
    Serial.println(" networks found");
    for (int i = 0; i < n; ++i) {
        // Print SSID and RSSI for each network found
        Serial.print(i + 1);
        Serial.print(": ");
        Serial.print(WiFi.SSID(i));
        Serial.print(" (");
        Serial.print(WiFi.RSSI(i));
        Serial.print(")");
        Serial.println((WiFi.encryptionType(i) == WIFI_AUTH_OPEN)?
": "*"");
        delay(10);
    }
}

if(wifiMulti.run() != WL_CONNECTED) {
    Serial.println("WiFi not connected!");
    delay(1000);
}

```

```

while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}
Serial.println("");
Serial.println("");
Serial.print("Connecting To: ");
Serial.print(WiFi.SSID());
Serial.println("");
Serial.print("Connected! IP address: ");
String ipAddress = WiFi.localIP().toString();
Serial.println(ipAddress);

WebSocket.begin();
WebSocket.onEvent(WebSocketEvent);

UDPServer.begin(UDPPort);
}

void loop(void) {
    WebSocket.loop();
    if(clientConnected == true){
        grabImage(jpgLength, jpgBuff);
        WebSocket.sendBIN(camNo, jpgBuff, jpgLength);
        // Serial.print("send img: ");
        // Serial.println(jpgLength);
    }

    unsigned long currentMillis = millis();
    if (currentMillis - previousMillisServo >= intervalServo) {
        previousMillisServo = currentMillis;
        processUDPData();
        controlServo();
    }

#ifdef DEBUG
    if (Serial.available()) {
        String data = Serial.readString();
        Serial.println(data);
        std::vector<String> vposVals = splitString(data, ",");
        if(vposVals.size() != 4){
            return;
        }
        int left0 = vposVals[0].toInt();
        int left1 = vposVals[1].toInt();
        int left2 = vposVals[2].toInt();
        int left3 = vposVals[3].toInt();
        controlDC(left0, left1, left2, left3);
    }

```

```
}  
#endif  
}
```

שפת – JAVA

Java היא שפת תכנות ברמה גבוהה, מבוססת object oriented, class, שתוכננה לכמה שפות תלות ביישום. זוהי שפת תכנות למטרות כלליות שנועדה לאפשר למתכנתים לכתוב פעם אחת, לרוץ בכל מקום, כלומר, קוד ג'אווה מהידור יכול לרוץ בכל הפלטפורמות התומכות ב-Java ללא צורך בהידור מחדש.

יישומי Java מורכבים בדרך כלל לקוד ביטים שיכול לפעול בכל מכונה וירטואלית של Java (JVM) ללא קשר לארכיטקטורת המחשב הבסיסית. התחביר של Java דומה ל-C++ ו-C, אך יש לו פחות מתקנים ברמה נמוכה מאשר כל אחד מהם. זמן הריצה של Java מספק יכולות דינמיות (כגון השתקפות ושינוי קוד זמן ריצה) שבדרך כלל אינן זמינות בשפות הידור מסורתיות.

נכון לשנת 2019, Java הייתה אחת משפות התכנות הפופולריות ביותר בשימוש לפי GitHub במיוחד עבור יישומי אינטרנט של שרת-לקוח, עם דיווח של 9 מיליון מפתחים.

Java פותחה במקור על ידי ג'יימס גוסלינג ב-Sun Microsystems. הוא שוחרר במאי 1995 כמרכיב ליבה של פלטפורמת Java של Sun Microsystems. מהדרים, מכונות וירטואליות וספריות מחלקות המקוריות והיישומיות של Java שוחררו במקור על ידי Sun תחת רישיונות קנייניים. החל ממאי 2007,

בהתאם למפרטים של תהליך הקהילה של Sun, Java העניקה רישיון מחדש לרוב טכנולוגיות ה-Java שלה תחת הרישיון GPL-2.0 בלבד. אורקל מציעה HotSpot Java Virtual Machine משלה, אולם יישום ההתייחסות הרשמי הוא OpenJDK JVM שהיא תוכנת קוד פתוח חנימית המשמשת את רוב המפתחים והיא ברירת המחדל של JVM עבור כמעט כל ההפצות של לינוקס.

עקרונות

היו חמש מטרות עיקריות ביצירת שפת Java:

- זה חייב להיות פשוט, מונחה עצמים ומוכר.
- זה חייב להיות חזק ומאובטח.
- זה חייב להיות ניטרלי אדריכלי ונייד.
- זה חייב לפעול בביצועים גבוהים.
- יש לפרש אותו, לשרשר ולדינמי.

JVM – Java Virtual Machine

לכל שפת התכנות יש מהדרים משלהם כדי להדר את קוד המקור בזמן הביצוע. גם מהדר Java עושה את אותו הדבר. מהדר Java מתרגם את קוד המקור לקוד Byte עבור מכונה כלשהי שאינה קיימת ומכונה זו נקראת Java Virtual Machine.

מכונה וירטואלית של Java (JVM) היא מכונה וירטואלית המאפשרת למחשב להריץ תוכניות ג'אווה וכן תוכניות הכתובות בשפות אחרות, אשר מורכבות גם הן לקוד בייט של ג'אווה. ה-JVM מפורט על ידי מפרט המתאר באופן רשמי את הנדרש ביישום JVM.

קיום מפרט מבטיח יכולת פעולה הדדית של תוכניות Java על פני יישומים שונים, כך שמחברי תוכניות המשתמשים בערכת הפיתוח של Java (JDK) לא צריכים לדאוג לגבי אידיוסנקרטיות של פלטפורמת החומרה הבסיסית.

הטמעת התייחסות JVM פותחה על ידי פרויקט OpenJDK כקוד קוד פתוח וכוללת מהדר JIT בשם HotSpot. מהדורות Java הנתמכות מסחרית הזמינות מ-Oracle מבוססות על זמן הריצה של Eclipse OpenJ9. OpenJDK הוא JVM נוסף בקוד פתוח עבור OpenJDK.

OpenJDK

OpenJDK (ערכת פיתוח ג'אווה פתוחה) היא מימוש חינומי וקוד פתוח של פלטפורמת Java, Standard Edition (Java SE). היא תוצאה של מאמץ שהחלה Sun Microsystems בשנת 2006.

היישום מורשה תחת GPL-2.0 בלבד עם חריג קישור. אלמלא חריג הקישור של GPL, רכיבים המקושרים לספריית מחלקות Java יהיו כפופים לתנאי רישיון OpenJDK. GPL. הוא יישום ההתייחסות הרשמי של Java SE מאז גרסה 7.

JDK

ערכת הפיתוח של Java (JDK) היא הפצה של טכנולוגיית Java על ידי Oracle Corporation. הוא מיישם את Java Language Specification (JLS) ואת Java Virtual Machine Specification (JVMS) ומספק את המהדורה הסטנדרטית (SE) של Java Application Programming Interface (API).

זה נגזרת של OpenJDK המונע על ידי הקהילה שאורקל מנהלת. הוא מספק תוכנה לעבודה עם יישומי Java. דוגמאות לתוכנות כלולות הן המכונה הוירטואלית, מהדר, כלים לניטור ביצועים, מאתר באגים וכלי עזר אחרים ש-Oracle מחשיבה כשימושיים עבור מתכנת Java.

שפת תוויות XML

XML (extensible Markup Language) היא שפת סימון המשמשת לקידוד מסמכים בפורמט הניתן לקריאה אנושית וגם למכונה. הוא משמש למבנה, אחסון והובלה של נתונים, ולעתים קרובות משמש להחלפת נתונים דרך האינטרנט.

XML מבוסס על שפת סימון כללית (SGML) וחולקת רבות מהתכונות שלה. הבדל מרכזי אחד בין שתי השפות הוא ש-XML גמיש וניתן להרחבה יותר מ-SGML, מה שמאפשר למשתמשים להגדיר תגים ותכונות משלהם.

ל-XML יש מערכת כללים להגדרת אלמנטים ותכונות שניתן להשתמש בהם כדי לייצג נתונים בצורה מובנית. יש לו גם מערכת כללים לציון כיצד יש לארגן את האלמנטים והתכונות בתוך מסמך. ניתן לקנן אותם בתוך תגים אחרים כדי ליצור היררכיה של מידע.

XML נמצא בשימוש נרחב במגוון יישומים, כולל פיתוח אתרים, אחסון ושידור נתונים וניהול מסמכים. זהו כלי חשוב לארגון והחלפת נתונים בפורמט סטנדרטי. XML גם נתמך באופן נרחב על ידי שפות תכנות ויישומי תוכנה רבים ושונים, מה שהופך אותו לבחירה פופולרית לאחסון והחלפת נתונים.

אחד היתרונות המרכזיים של XML הוא שהוא מתאר את עצמו, מה שאומר שמבנה הנתונים מוגדר במפורש בתוך המסמך עצמו. זה מקל על מערכות שונות לפרש ולעבד את הנתונים, שכן המבנה מוגדר בבירור.

שפת Python

Python היא שפת תכנות פופולרית ברמה גבוהה הידועה בפשטות, בקריאות ובגמישות שלה. זוהי שפה מפורשת, כלומר היא מבוצעת על ידי מתורגמן במקום להיות קומפילציה לקוד מכונה שיכול לפעול על מחשב. זה הופך אותה לשפה אידיאלית למתחילים ללמוד, כמו גם למתכנתים מנוסים שרוצים ליצור אבטיפוס במהירות ולפתח יישומים.

לפייתון יש קהילה גדולה ופעילה של משתמשים ומפתחים, והיא נמצאת בשימוש במגוון רחב של תחומים, לרבות מחשוב מדעי, ניתוח נתונים, בינה מלאכותית ופיתוח אתרים. יש לו ספרייה סטנדרטית גדולה המספקת תמיכה למשימות תכנות נפוצות רבות, ויש גם ספריות רבות של צד שלישי זמינות המרחיבות את יכולותיה.

אחת התכונות המרכזיות של Python היא התמיכה שלה בתכנות מונחה עצמים, שהיא פרדיגמת תכנות המאפשרת יצירת קוד לשימוש חוזר באמצעות שימוש במחלקות ואובייקטים. לפייתון יש גם תמיכה חזקה בתכנות פונקציונלי, שהיא פרדיגמת תכנות המתמקדת בשימוש בפונקציות כדי לפעול על נתונים.

בנוסף לשימוש בתכנות למטרות כלליות, Python נמצא בשימוש נרחב גם במחשוב מדעי וניתוח נתונים. יש לו מספר ספריות, כגון NumPy ו-Pandas, שתוכננו במיוחד עבור משימות אלו ומקלות על העבודה עם מערכי נתונים גדולים ולבצע חישובים מורכבים.

Python היא שפה רב-תכליתית וניתן להשתמש בה למגוון רחב של משימות, כולל פיתוח אתרים, יישומי שולחן עבודה ואפילו פיתוח אפליקציות לנייד. יש לו תחביר פשוט וספרייה סטנדרטית גדולה, מה שמקל על הלמידה והשימוש, והתמיכה החזקה שלו בתכנות מונחה עצמים ופונקציונלי הופכת אותו לכלי רב עוצמה לבניית קוד מורכב וניתן לשימוש חוזר.

קוד עיבוד תמונה של ESP32 – CAM

```
import cv2
thres = 0.6# Threshold to detect object

classNames = []
classFile = "coco.names"
with open(classFile,'rt') as f:
    classNames=[line.rstrip() for line in f]

configPath = "ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt"
weightsPath = "frozen_inference_graph.pb"

net = cv2.dnn_DetectionModel(weightsPath, configPath)
net.setInputSize(320, 240)
net.setInputScale(1.0 / 127.5)
net.setInputMean((127.5, 127.5, 127.5))
net.setInputSwapRB(True)

def getObjects(img, draw=True,objects=[]):

    classIds, confs, bbox = net.detect(img, confThreshold=thres,
nmsThreshold=0.3)
    #print(classIds, bbox)

    objectInfo=[]
    if(len(objects) == 0): objects = classNames
    if len(classIds) != 0:
        for classId, confidence, box in zip(classIds.flatten(),
confs.flatten(), bbox):
            className = classNames[classId - 1]
            if className in objects:
                objectInfo.append([box, className])
```

```

        if (draw):
            cv2.rectangle(img, box, color=(0, 255, 0),
thickness=2)
            cv2.putText(img, className.upper(), (box[0]
+ 10, box[1] + 30),
            cv2.FONT_HERSHEY_COMPLEX, 1, (0, 255, 0), 2)
            cv2.putText(img, str(round(confidence * 100,
2)), (box[0] + 200, box[1] + 30),
            cv2.FONT_HERSHEY_COMPLEX, 1, (0, 255, 0), 2)
        return img, objectInfo

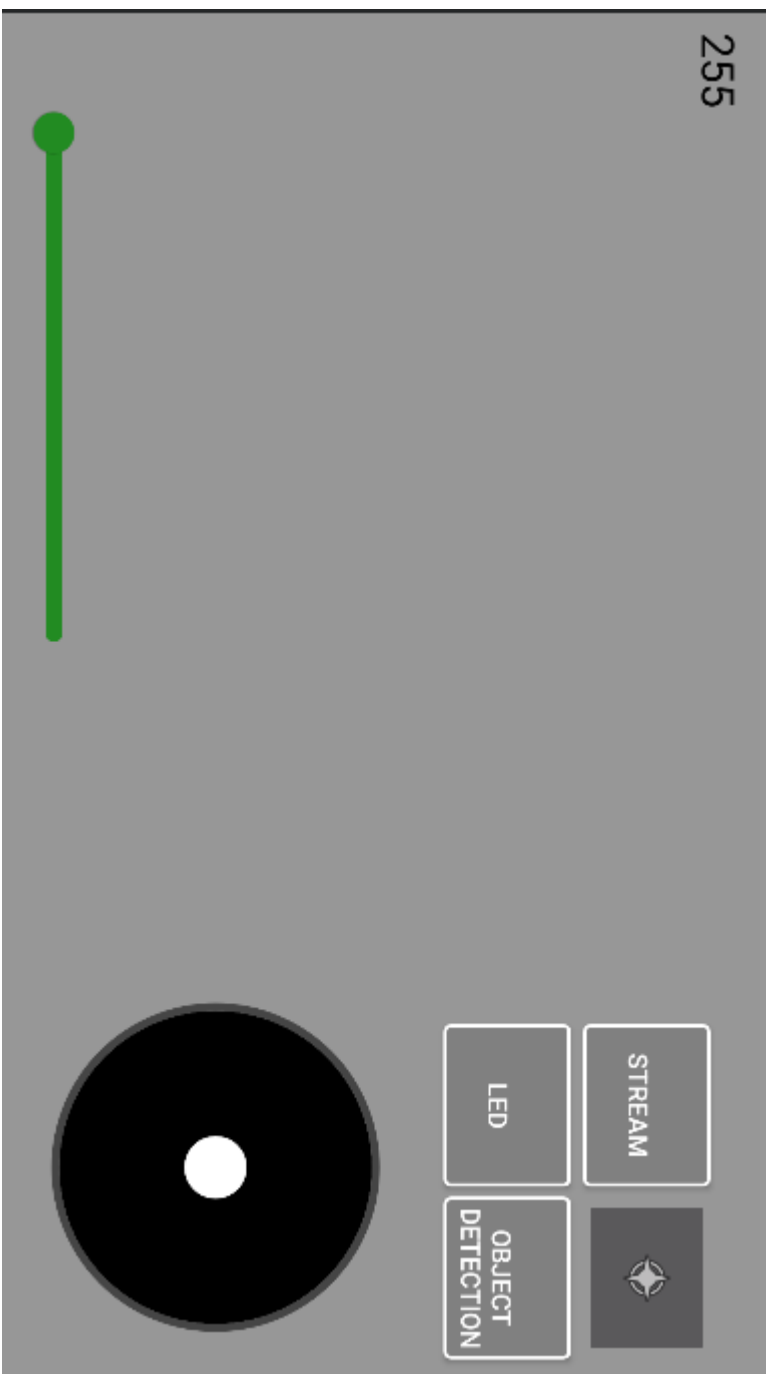
if __name__ == "__main__":
    cap = cv2.VideoCapture("http://192.168.1.34:81/stream")
    #cap.set(3, 1280)
    #cap.set(4, 720)
    #cap.set(10, 70)

    while True:
        success, img = cap.read()
        result, objectInfo = getObjects(img, True)

        cv2.imshow("Detection Window", img)
        cv2.waitKey(1)

```


האפליקציה צילומי מסך



Esp32CameraFragment - קוד

```
package com.p4f.esp32camai;

import android.app.AlertDialog;
import android.app.Dialog;
import android.content.DialogInterface;
import android.content.res.AssetManager;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Matrix;
import android.graphics.Paint;
import android.graphics.Rect;
import android.os.Bundle;
import android.os.Handler;
import android.text.InputType;
import android.util.Log;
import android.util.Pair;
import android.util.Size;
import android.view.Gravity;
import android.view.LayoutInflater;
import android.view.MotionEvent;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.FrameLayout;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;

import com.google.android.material.slider.Slider;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import com.p4f.esp32camai.tflite.Classifier;
import com.p4f.esp32camai.tflite.TFLiteObjectDetectionSSDAPIModel;

import java.io.IOException;
import java.io.InputStream;
import java.net.InetAddress;
import java.net.SocketAddress;
import java.net.URI;
import java.net.URISyntaxException;
import java.nio.ByteBuffer;
import java.util.ArrayList;
import java.util.List;
import android.graphics.Point;
```

```

import org.java_websocket.client.WebSocketClient;
import org.java_websocket.handshake.ServerHandshake;
import org.opencv.android.OpenCVLoader;
import org.opencv.android.Utils;
import org.opencv.core.CvException;
import org.opencv.core.Mat;
import org.opencv.core.Scalar;
import org.opencv.imgproc.Imgproc;
import org.opencv.tracking.Tracker;
import org.opencv.tracking.TrackerCSRT;
import org.opencv.tracking.TrackerKCF;
import org.opencv.tracking.TrackerMIL;
import org.opencv.tracking.TrackerMOSSE;
import org.opencv.tracking.TrackerMedianFlow;
import org.opencv.tracking.TrackerTLD;

import io.github.controlwear.virtual.joystick.android.JoystickView;

public class Esp32CameraFragment extends Fragment{

    public enum STATE {
        STOP,
        RUN,
        PAUSE
    };

    enum Drawing{
        DRAWING,
        TRACKING,
        CLEAR,
    }

    final String TAG = "ExCameraFragment";

    private UDPSocket mUdpClient;
    private String mServerAddressBroadCast = "255.255.255.255";
    InetAddress mServerAddr;
    int mServerPort = 6868;
    final byte[] mRequestConnect = new
byte[]{ 'w', 'h', 'o', 'a', 'm', 'i' };
    final byte[] mRequestForward = new
byte[]{ 'f', 'o', 'r', 'w', 'a', 'r', 'd' };
    final byte[] mRequestForwardTrack = new
byte[]{ 'f', 'w', 't', 'r', 'a', 'c', 'k' };
    final byte[] mRequestBackward = new
byte[]{ 'b', 'a', 'c', 'k', 'w', 'a', 'r', 'd' };
    final byte[] mRequestLeft = new byte[]{ 'l', 'e', 'f', 't' };
    final byte[] mRequestLeftTrack = new
byte[]{ 'l', 'e', 'f', 't', 't', 'r', 'a', 'c', 'k' };
    final byte[] mRequestRight = new
byte[]{ 'r', 'i', 'g', 'h', 't' };
    final byte[] mRequestRightTrack = new
byte[]{ 'r', 'i', 'g', 'h', 't', 't', 'r', 'a', 'c', 'k' };
    final byte[] mRequestStop = new byte[]{ 's', 't', 'o', 'p' };
    final byte[] mRequestCamUp = new
byte[]{ 'c', 'a', 'm', 'u', 'p' };
    final byte[] mRequestCamDown = new
byte[]{ 'c', 'a', 'm', 'd', 'o', 'w', 'n' };
    final byte[] mRequestCamLeft = new
byte[]{ 'c', 'a', 'm', 'l', 'e', 'f', 't' };
    final byte[] mRequestCamRight = new

```



```

byte[]{'c','a','m','r','i','g','h','t'};
    final byte[] mRequestCamStill = new
byte[]{'c','a','m','s','t','i','l','l'};
    final byte[] mLedOn = new byte[]{'l','e','d','o','n'};
    final byte[] mLaserOn = new byte[]{'s','p','e','e','d','o','n'};
    final byte[] mLaserOff = new
byte[]{'s','p','e','e','d','o','f','f'};
    final byte[] mLedOff = new byte[]{'l','e','d','o','f','f'};
    private Handler handler = new Handler();
    private Bitmap mBitmap;

    ImageView mServerImageView;
    Handler mHandler = new Handler();

    private WebSocketClient mWebSocketClient;
    private String mServerExactAddress;
    private boolean mInitStream = false;
    private boolean mInitTrackObj = false;
    private boolean mStream = false;
    private boolean mObjDet = false;
    private boolean mLed = false;
    private boolean mLaser = false;

    private Classifier detectorSSD;
    private List<TFLiteObjectDetectionSSDAPIModel.Recognition>
detectorSSDResult = new ArrayList<>();

    private final Size CamResolution = new Size(640, 480);

    private OverlayView mTrackingOverlay;
    private Bitmap mBitmapDebug;
    private boolean mProcessing = false;
    private Point[] mPoints = new Point[4];
    private Point mPointCircle = new Point();
    private int mRadiusCircle = 0;
    private Drawing mDrawing = Drawing.CLEAR;
    private boolean mTargetLocked = false;
    private Bitmap mBitmapGrab = null;

    private String mSelectedTracker = "None";
    private String mSelectedTrackerPre = "None";
    private Tracker mTracker;
    private Mat mMatGrabInit;
    private Mat mMatGrab;
    private org.opencv.core.Rect2d mInitRectangle = null;
    private int mBinaryThreshold = 80;
    private int mRadioIndex = 0;
    int packet,motors,servo,speed,laser;
    TextView tempText;
    Slider servoSlider;

    private Bitmap mBitmapLaneTracking = null;
    //    final String[] mRadioBtnNames = {
//        "None",
//        "TrackerMedianFlow",
//        "TrackerCSRT",
//        "TrackerKCF",
//        "TrackerMOSSE",

```

```

//          "TrackerTLD",
//          "TrackerMIL",
//          "LaneTracking"
//      };

final String[] mRadioBtnNames = {
    "None",
    "ObjectTracking",
    "LaneTracking"
};

public void onFocusChanged() {
    int testW = mTrackingOverlay.getWidth();
    int testH = mTrackingOverlay.getHeight();
    //      mTrackingOverlay.setLayoutParams(new
    //      FrameLayout.LayoutParams(testW,
    //      CamResolution.getWidth()/CamResolution.getHeight()*testW));
    //      mServerImageView.setLayoutParams(new
    //      FrameLayout.LayoutParams(testW,
    //      CamResolution.getWidth()/CamResolution.getHeight()*testW));
}

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    mUdpClient = new UDPSocket(12345);
    mUdpClient.runUdpServer();

    try {
        mServerAddr =
        InetAddress.getByName(mServerAddressBroadCast);
    } catch (Exception e) {

    }

    AssetManager assetManager = getActivity().getAssets();
    if (MyConstants.DEBUG) {
        try {
            InputStream istr = assetManager.open("image1.jpg");
            Bitmap tmpBitmap = BitmapFactory.decodeStream(istr);
            mBitmapDebug = Bitmap.createScaledBitmap(tmpBitmap,
            CamResolution.getWidth(), CamResolution.getHeight(), false);
        } catch (IOException e) {
            // handle exception
        }
    }

    if (!OpenCVLoader.initDebug()) {
        Log.d(TAG, "Internal OpenCV library not found. Using
        OpenCV Manager for initialization");
        OpenCVLoader.initAsync(OpenCVLoader.OPENCV_VERSION_3_4_0,
        getActivity(), null);
    }
}

@Override
public View onCreateView(LayoutInflater inflater, ViewGroup
parent, Bundle savedInstanceState) {
    View rootView = inflater.inflate(R.layout.fragment_camera,
parent, false);

```

```

        FirebaseDatabase database = FirebaseDatabase.getInstance();
        DatabaseReference dbRef = database.getReference("kar98Info");

        tempText = (TextView) rootView.findViewById(R.id.tempView);
        servoSlider = (Slider)
rootView.findViewById(R.id.servoSlider);
        JoystickView joystick = (JoystickView)
rootView.findViewById(R.id.joystick);
        joystick.setAutoReCenterButton(true);

        joystick.setOnMoveListener(new JoystickView.OnMoveListener()
{
    @Override
    public void onMove(int angle, int strength) {
        CompilePacket(dbRef);
        if(strength < 30){ //stop car when no input
            motors = 0;
            speed = 0;
            return;
        }
        CompilePacket(dbRef);
        if (strength > 90){
            speed = 1;
            CompilePacket(dbRef);
        }
        if(angle < 135 && angle > 45) //forward
            motors =2;
        CompilePacket(dbRef);

        if(angle < 315 && angle > 225)//backward
            motors =1;
        CompilePacket(dbRef);

        if(angle < 45 && angle > 0 || angle > 315 && angle <
360)
        {
            //right
            motors =4;
        }
        CompilePacket(dbRef);

        if(angle < 180 && angle > 135 || angle > 180 && angle
< 225)
        {
            //left
            motors =3;
        }
        CompilePacket(dbRef);
    }
});

        mServerImageView =
        (ImageView)rootView.findViewById(R.id.imageView);
        Button streamBtn = (Button)
rootView.findViewById(R.id.streamBtn);
        streamBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v){
                if (!mStream) {

```

```

        try {
            mServerAddr =
InetAddress.getByName(mServerAddressBroadCast);
        } catch (Exception e) {

        }

        mUdpClient.sendBytes(mServerAddr, mServerPort,
mRequestConnect);
        Pair<SocketAddress, String> res =
mUdpClient.getResponse();
        int cnt = 3;
        while (res.first == null && cnt > 0) {
            res = mUdpClient.getResponse();
            cnt--;
        }
        if (res.first != null) {
            Log.d(TAG, res.first.toString() + ":" +
res.second);
            mServerExactAddress =
res.first.toString().split("\\.")[0].replace("/", "");
            mStream = true;
            connectWebSocket();
            ((Button)
getActivity().findViewById(R.id.streamBtn)).setBackgroundResource(R.d
rawable.my_button_bg_2);
            ((Button)
getActivity().findViewById(R.id.streamBtn)).setTextColor(Color.rgb(0,
0,255));

            try {
                mServerAddr =
InetAddress.getByName(mServerExactAddress);
            } catch (Exception e) {

            }
        } else {
            Toast toast =
                Toast.makeText(
                    getActivity(), "Cannot
connect to ESP32 Camera", Toast.LENGTH_LONG);
            toast.setGravity(Gravity.CENTER, 0, 0);
            toast.show();
        }
    } else {
        mStream = false;
        mWebSocketClient.close();
        ((Button)
getActivity().findViewById(R.id.streamBtn)).setBackgroundResource(R.d
rawable.my_button_bg);
        ((Button)
getActivity().findViewById(R.id.streamBtn)).setTextColor(Color.rgb(25
5,255,255));
    }
}

});

Button ledBtn = (Button) rootView.findViewById(R.id.ledBtn);
ledBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (!mLed) {
            mLed = true;

```

```

        ((Button)
getActivity().findViewById(R.id.ledBtn)).setBackgroundResource(R.draw
able.my_button_bg_2);
        ((Button)
getActivity().findViewById(R.id.ledBtn)).setTextColor(Color.rgb(0,0,2
55));
        dbRef.child("led").setValue(1);
        mUdpClient.sendBytes(mServerAddr, mServerPort,
mLedOn);
    }else{
        mLed = false;
        ((Button)
getActivity().findViewById(R.id.ledBtn)).setBackgroundResource(R.draw
able.my_button_bg);
        ((Button)
getActivity().findViewById(R.id.ledBtn)).setTextColor(Color.rgb(255,2
55,255));
        dbRef.child("led").setValue(0);
        mUdpClient.sendBytes(mServerAddr, mServerPort,
mLedOff);
    }
    });

    ImageButton ShootBtn = (ImageButton)
rootView.findViewById(R.id.shootBtn);
    ImageButton.OnTouchListener listener = new
ImageButton.OnTouchListener() {
        @Override
        public boolean onTouch(View arg0, MotionEvent event){
            if(event.getAction() == MotionEvent.ACTION_DOWN)
{
                if (((ImageButton) arg0).getId() ==
R.id.shootBtn);
                {
                    mLaser = true;
                    ((ImageButton)
getActivity().findViewById(R.id.shootBtn)).setBackgroundResource(R.dr
awable.my_button_bg_2);
                    laser = 1;
                    CompilePacket(dbRef);
                }
                return true;
            }
            else if(event.getAction() ==
MotionEvent.ACTION_UP) {
                if (((ImageButton) arg0).getId() ==
R.id.shootBtn) {
                    mLaser = false;
                    ((ImageButton)
getActivity().findViewById(R.id.shootBtn)).setBackgroundResource(R.dr
awable.my_button_bg);
                    laser =0;
                    CompilePacket(dbRef);
                }
                return true;
            }
            return true;
        }
    }
};

```

```

        ShootBtn.setOnTouchListener(listener);

        servoSlider.addOnChangeListener(new Slider.OnChangeListener()
        {
            @Override
            public void onValueChange(@NonNull Slider slider, float
value, boolean fromUser) {
                switch((int)value) {
                    case 0:
                        servo=7;
                        CompilePacket(dbRef);
                        break;
                    case 10:
                        servo=6;
                        CompilePacket(dbRef);
                        break;
                    case 20:
                        servo=5;
                        CompilePacket(dbRef);
                        break;
                    case 30:
                        servo=4;
                        CompilePacket(dbRef);
                        break;
                    case 40:
                        servo=3;
                        CompilePacket(dbRef);
                        break;
                    case 50:
                        servo=2;
                        CompilePacket(dbRef);
                        break;
                    case 60:
                        servo=1;
                        CompilePacket(dbRef);
                        break;
                    case 70:
                        servo=0;
                        CompilePacket(dbRef);
                        break;
                    default:
                        servo=0;
                        CompilePacket(dbRef);
                        break;
                }
            }
        });

        dbRef.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {
                String temp =
dataSnapshot.child("temperature").getValue().toString();

tempText.setText(CalculateCel(Integer.parseInt(temp)));
            }
            @Override
            public void onCancelled(DatabaseError error) {
                // Failed to read value
                Log.w("Firebase", "Failed to read value.",

```

```

        error.toException());
    }
});

    Button objDetBtn = (Button)
    rootView.findViewById(R.id.objDetBtn);
    objDetBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            if (!mObjDet) {
                ((Button)
                getActivity().findViewById(R.id.objDetBtn)).setBackgroundResource(R.d
                rawable.my_button_bg_2);
                ((Button)
                getActivity().findViewById(R.id.objDetBtn)).setTextColor(Color.rgb(0,
                0,255));
            } else {
                ((Button)
                getActivity().findViewById(R.id.objDetBtn)).setBackgroundResource(R.d
                rawable.my_button_bg);
                ((Button)
                getActivity().findViewById(R.id.objDetBtn)).setTextColor(Color.rgb(25
                5,255,255));
            }
            mObjDet = !mObjDet;
        }
    });

    try {
        detectorSSD =
            TFLiteObjectDetectionSSDAPIModel.create(
                getActivity().getAssets(),
                "ssdlite_mobilenet_v2_quantized.tflite",
                "",
                300,
                Classifier.Device.CPU,
                MyConstants.MODEL_TYPE.UINT8,
                0.5f,
                1,
                CamResolution.getWidth(),
                CamResolution.getHeight()
            );
        detectorSSD.startThread();
    } catch (final IOException e) {
        Log.e(TAG, "Exception initializing classifier!");
        Toast toast =
            Toast.makeText(
                getActivity(), "Classifier could not be
                initialized", Toast.LENGTH_SHORT);
        toast.show();
    }

    for (int i=0; i<mPoints.length;i++){
        mPoints[i] = new Point(0,0);
    }
}

```

```

    }

    mTrackingOverlay = (OverlayView)
    rootView.findViewById(R.id.tracking_overlay);
    assert (mTrackingOverlay != null);

    mTrackingOverlay.addCallback(
        new OverlayView.DrawCallback() {
            @Override
            public void drawCallback(Canvas canvas) {
                if (MyConstants.DEBUG) {
                    Rect dstRectForRender = new Rect(0, 0,
mTrackingOverlay.getWidth(), mTrackingOverlay.getHeight());
                    Matrix matrix = new Matrix();
                    matrix.postRotate(90);
                    Bitmap scaleBitmap =
Bitmap.createScaledBitmap(mBitmapDebug, mTrackingOverlay.getWidth(),
mTrackingOverlay.getHeight(), false);
                    Bitmap rotatedBitmap =
Bitmap.createBitmap(scaleBitmap, 0, 0, mTrackingOverlay.getWidth(),
mTrackingOverlay.getHeight(), matrix, true);
                    canvas.drawBitmap(rotatedBitmap, null,
dstRectForRender, null);
                }
                if (detectorSSD != null && mObjDet) {
                    int overlayWidth =
mTrackingOverlay.getWidth();
                    int overlayHeight =
mTrackingOverlay.getHeight();
                    int imgWidth = mBitmapGrab.getWidth();
                    int imgHeight = mBitmapGrab.getHeight();
                    ((TFLiteObjectDetectionSSDAPIModel)
detectorSSD).getResult(detectorSSDResult);
                    Paint paint = new Paint();
                    Paint paintText = new Paint();
                    paint.setColor(Color.rgb(0, 255, 0));
                    Log.d(TAG, "Obj cnt: " +
detectorSSDResult.size());
                    for
(TFLiteObjectDetectionSSDAPIModel.Recognition det :
detectorSSDResult) {
                        Log.d(TAG, "processing: " + det);
                        paint.setStrokeWidth(10);
                        paint.setStyle(Paint.Style.STROKE);
                        float left = det.getLocation().left *
mTrackingOverlay.getWidth();
                        if (left < 0) {
                            left = 0;
                        } else if (left >
mTrackingOverlay.getWidth()) {
                            left =
mTrackingOverlay.getWidth();
                        }

                        float top = det.getLocation().top *
mTrackingOverlay.getHeight();
                        if (top < 0) {
                            top = 0;
                        } else if (top >
mTrackingOverlay.getHeight()) {
                            top =

```



```

mTrackingOverlay.getHeight();
    }

    float right = det.getLocation().right
* mTrackingOverlay.getWidth();
    ;
    if (right < 0) {
        right = 0;
    } else if (right >
mTrackingOverlay.getWidth()) {
        right =
mTrackingOverlay.getWidth();
    }

    float bottom =
det.getLocation().bottom * mTrackingOverlay.getHeight();
    if (bottom < 0) {
        bottom = 0;
    } else if (bottom >
mTrackingOverlay.getHeight()) {
        bottom =
mTrackingOverlay.getHeight();
    }
    paintText.setColor(Color.BLUE);
    paintText.setStrokeWidth(2);
    paintText.setStyle(Paint.Style.FILL);
    paintText.setTextSize(50);
    canvas.drawRect(left, top, right,
bottom, paint);

    paint.setStyle(Paint.Style.FILL);
    String txt = det.getTitle();// + "("
+ String.format("%.2f", det.getConfidence()) + ")";
    canvas.drawRect(left, top, left-60,
top+txt.length()*30+50, paint);
    canvas.save();
    canvas.rotate(90, left-50, top + 50);
    canvas.drawText(txt, left - 50, top +
50, paintText);
    canvas.restore();
}

if(mSelectedTracker.equals("ObjectTracking")
&& mStream){
    if(!mInitTrackObj){
        String msg1 = "Object is selected by
1 touch and drag following by a";
        String msg2 = "rectangle, make double
touch with another finger to lock";
        String msg3 = "the object, double
touch again to release the tracking object";
        Paint paintText = new Paint();
        paintText.setColor(Color.YELLOW);
        paintText.setStrokeWidth(2);
        paintText.setStyle(Paint.Style.FILL);

        paintText.setTextSize(mTrackingOverlay.getWidth()/23);
        canvas.save();
        canvas.rotate(90,
mTrackingOverlay.getWidth()*10/12, mTrackingOverlay.getHeight()/8);
        canvas.drawText(msg1,

```

```

mTrackingOverlay.getWidth()*5/6, mTrackingOverlay.getHeight()/8,
paintText);

        canvas.restore();
        canvas.save();
        canvas.rotate(90,
mTrackingOverlay.getWidth()*9/12, mTrackingOverlay.getHeight()/8);
        canvas.drawText(msg2,
mTrackingOverlay.getWidth()*9/12, mTrackingOverlay.getHeight()/8,
paintText);

        canvas.restore();
        canvas.save();
        canvas.rotate(90,
mTrackingOverlay.getWidth()*8/12, mTrackingOverlay.getHeight()/8);
        canvas.drawText(msg3,
mTrackingOverlay.getWidth()*8/12, mTrackingOverlay.getHeight()/8,
paintText);

        canvas.restore();
        canvas.save();
    }
    if (mDrawing != Drawing.CLEAR) {
        Paint paint = new Paint();
        paint.setColor(Color.rgb(0, 0, 255));
        paint.setStrokeWidth(10);
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawRect(mPoints[0].x,
mPoints[0].y, mPoints[1].x, mPoints[1].y, paint);
        if (mDrawing == Drawing.TRACKING) {
            paint.setColor(Color.rgb(0, 255,
0));

            canvas.drawLine((mPoints[0].x +
mPoints[1].x) / 2,
0,
(mPoints[0].x +
mPoints[1].x) / 2,
mTrackingOverlay.getHeight(),
paint);
            canvas.drawLine(0,
(mPoints[0].y +
mPoints[1].y) / 2,
mTrackingOverlay.getWidth(),
(mPoints[0].y +
mPoints[1].y) / 2,
paint);
            paint.setColor(Color.YELLOW);
            paint.setStrokeWidth(2);
            paint.setStyle(Paint.Style.FILL);
            paint.setTextSize(30);
            String strX =
Integer.toString((mPoints[0].x + mPoints[1].x) / 2) + "/" +
Integer.toString(mTrackingOverlay.getWidth());
            String strY =
Integer.toString((mPoints[0].y + mPoints[1].y) / 2) + "/" +
Integer.toString(mTrackingOverlay.getHeight());
            canvas.drawText(strX,
(mPoints[0].x + mPoints[1].x) / 4, (mPoints[0].y + mPoints[1].y) / 2
- 10, paint);

            canvas.save();
            canvas.rotate(90, (mPoints[0].x +
mPoints[1].x) / 2 + 10, (mPoints[0].y + mPoints[1].y) / 4);

```

```

        canvas.drawText(strY,
(mPoints[0].x + mPoints[1].x) / 2 + 10, (mPoints[0].y + mPoints[1].y)
/ 4, paint);

        canvas.restore();
    }
}
}else
if(mSelectedTracker.equals("LaneTracking") && mStream){
    Rect dstRectForRender = new Rect(0, 0,
mTrackingOverlay.getWidth(), mTrackingOverlay.getHeight());
    Matrix matrix = new Matrix();
    matrix.postRotate(90);
    Bitmap scaleBitmap =
Bitmap.createScaledBitmap(mBitmapLaneTracking,
mTrackingOverlay.getWidth(), mTrackingOverlay.getHeight(), false);
    Bitmap rotatedBitmap =
Bitmap.createBitmap(scaleBitmap, 0, 0, mTrackingOverlay.getWidth(),
mTrackingOverlay.getHeight(), matrix, true);
    Paint alphaPaint = new Paint();
    alphaPaint.setAlpha(42);
    canvas.drawBitmap(rotatedBitmap, null,
dstRectForRender, alphaPaint);
}else
if(mSelectedTracker.equals("ColorTracking") && mStream){
    Paint paint = new Paint();
    paint.setColor(Color.argb(50,0, 0, 255));
    paint.setStrokeWidth(10);
    paint.setStyle(Paint.Style.FILL);
    canvas.drawCircle(mPointCircle.x,
mPointCircle.y, mRadiusCircle, paint);
}else if(mSelectedTracker.equals("None") &&
mStream){
    mInitTrackObj = false;
    //TODO
    if(!mInitStream){
        canvas.save();
        canvas.rotate(90,
mTrackingOverlay.getWidth()*5/6, mTrackingOverlay.getHeight()/8);
        canvas.restore();
        canvas.save();
        canvas.rotate(90,
mTrackingOverlay.getWidth()/6, mTrackingOverlay.getHeight()/8);
        canvas.restore();
    }
}
}
};

mTrackingOverlay.setOnTouchListener(new
View.OnTouchListener() {
    @Override
    public boolean onTouch(View view, MotionEvent event) {
        final int X = (int) event.getX();
        final int Y = (int) event.getY();
        Log.d(TAG, ": " + Integer.toString(X) + " " +
Integer.toString(Y) );
        mInitStream = true;
        mInitTrackObj = true;
        switch (event.getAction() & MotionEvent.ACTION_MASK)
{

```

```

        case MotionEvent.ACTION_UP:
//          Log.d(TAG, ": " +
"MotionEvent.ACTION_UP" );
            if (mSelectedTracker.equals("None")) {
                mUdpClient.sendBytes(mServerAddr,
mServerPort, mRequestCamStill);
                break;
            }
            if(!mTargetLocked) {
                mDrawing = Drawing.CLEAR;
                mTrackingOverlay.postInvalidate();
            }
            break;
        case MotionEvent.ACTION_POINTER_DOWN:
//          Log.d(TAG, ": " +
"MotionEvent.ACTION_POINTER_DOWN" );

        if(mSelectedTracker.equals("ObjectTracking")==false){
            break;
        }
        if (mTargetLocked == false) {
            if((mPoints[0].x-mPoints[1].x != 0) &&
(mPoints[0].y-mPoints[1].y != 0)) {
                mTargetLocked = true;
                mMatGrab = new Mat();
                Toast toast =
Toast.makeText(getActivity(), "Target is LOCKED !",
Toast.LENGTH_LONG);
                toast.setGravity(Gravity.TOP |
Gravity.CENTER, 0, 0);
                toast.show();
            }else{
                mTargetLocked = false;
            }
        }else{
            mTargetLocked = false;
            Toast toast =
Toast.makeText(getActivity(), "Target is UNLOCKED !",
Toast.LENGTH_LONG);
            toast.setGravity(Gravity.TOP |
Gravity.CENTER, 0, 0);
            toast.show();
        }
        mDrawing = Drawing.DRAWING;
        mTrackingOverlay.postInvalidate();
        break;
        case MotionEvent.ACTION_POINTER_UP:
//          Log.d(TAG, ": " +
"MotionEvent.ACTION_POINTER_UP" );
            break;
        case MotionEvent.ACTION_DOWN:
//          Log.d(TAG, ": " +
"MotionEvent.ACTION_DOWN" );
            if (mSelectedTracker.equals("None")) {
                if (X < mTrackingOverlay.getWidth() / 2)
{
                    mUdpClient.sendBytes(mServerAddr,
mServerPort, mRequestCamDown);
                } else {
                    mUdpClient.sendBytes(mServerAddr,
mServerPort, mRequestCamUp);
                }
            }

```

```

        }
        break;
    }
    if(!mTargetLocked &&
mSelectedTracker.equals("ObjectTracking")) {
        mDrawing = Drawing.DRAWING;
        mPoints[0].x = X;
        mPoints[0].y = Y;
        mPoints[1].x = X;
        mPoints[1].y = Y;
        mTrackingOverlay.postInvalidate();
    }
    break;
    case MotionEvent.ACTION_MOVE:
//
        Log.d(TAG, ":" +
"MotionEvent.ACTION_MOVE" );
        if(!mTargetLocked &&
mSelectedTracker.equals("ObjectTracking")) {
            mPoints[1].x = X;
            mPoints[1].y = Y;
            mTrackingOverlay.postInvalidate();
        }
        break;
    }
    if(mTargetLocked==true){
//
//
getView().findViewById(R.id.objTrackBtn).setEnabled(false);
//
    }else{
//
//
getView().findViewById(R.id.objTrackBtn).setEnabled(true);
//
    }
    return true;
}
});

return rootView;
}

private void connectWebSocket() {
    URI uri;
    try {
        uri = new URI("ws://" + mServerExactAddress + ":86/");
    } catch (URISyntaxException e) {
        e.printStackTrace();
        return;
    }

    mWebSocketClient = new WebSocketClient(uri) {
        @Override
        public void onOpen(ServerHandshake serverHandshake) {
            Log.d("Websocket", "Open");
        }

        @Override
        public void onClose(int i, String s, boolean b) {
            Log.d("Websocket", "Closed " + s);
        }

        @Override
        public void onMessage(String message){
            Log.d("Websocket", "Receive");
        }
    };
}

```

```

        }

        @Override
        public void onMessage(ByteBuffer message){
//            Log.d("Websocket", "Receive");
            getActivity().runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    byte[] imageBytes= new
byte[message.remaining()];
                    message.get(imageBytes);
                    final Bitmap
bmp=BitmapFactory.decodeByteArray(imageBytes,0,imageBytes.length);
                    if (bmp == null)
                    {
                        return;
                    }
                    int viewWidth = mServerImageView.getWidth();
                    Matrix matrix = new Matrix();
                    matrix.postRotate(90);
                    final Bitmap bmp_traspose =
Bitmap.createBitmap(bmp, 0, 0, bmp.getWidth(), bmp.getHeight(),
matrix, true );
                    float imagRatio =
(float)bmp_traspose.getHeight()/((float)bmp_traspose.getWidth());
                    int dispViewH = (int)(viewWidth*imagRatio);

mServerImageView.setImageBitmap(Bitmap.createScaledBitmap(bmp_traspos
e, viewWidth, dispViewH, false));

                    mBitmapGrab = bmp;
                    mProcessing = detectorSSD.IsProcessing;
                    if (!mProcessing) {
                        processing();
                    }
                }
            });
        }

        @Override
        public void onError(Exception e) {
            Log.d("Websocket", "Error " + e.getMessage());
        }
    };
    mWebSocketClient.connect();
}

private void trackingDlg(){
    AlertDialog.Builder builder = new
AlertDialog.Builder(getActivity());
    builder.setTitle("Tracker Selection");

    final RadioButton[] rb = new
RadioButton[mRadioBtnNames.length];
    RadioGroup rg = new RadioGroup(getActivity()); //create the
RadioGroup
    rg.setOrientation(RadioGroup.VERTICAL);

    for(int i=0; i < mRadioBtnNames.length; i++){
        rb[i] = new RadioButton(getActivity());
        rb[i].setText(" " + mRadioBtnNames[i]);
    }
}

```

```

        rb[i].setId(i + 100);
        rg.addView(rb[i]);
        if (mRadioBtnNames[i].equals(mSelectedTracker)) {
            rb[i].setChecked(true);
        }
    }

    // This overrides the radiogroup onCheckedChangeListener
    rg.setOnCheckedChangeListener(new
    RadioGroup.OnCheckedChangeListener()
    {
        public void onCheckedChanged(RadioGroup group, int
checkedId) {
            // This will get the radiobutton that has changed in
its check state
            RadioButton checkedRadioButton =
(RadioButton)group.findViewById(checkedId);
            // This puts the value (true/false) into the variable
            boolean isChecked = checkedRadioButton.isChecked();
            if (isChecked)
            {
                // Changes the textview's text to "Checked:
example radiobutton text"
                int i = 0;
                for (i = 0; i < mRadioBtnNames.length; i++) {
                    if (checkedRadioButton.getText().toString().replace(" ",
"").equals(mRadioBtnNames[i])) {
                        break;
                    }
                }
                mRadioIndex = i;
            }
        }
    });

    LinearLayout lay = new LinearLayout(getActivity());
    lay.setOrientation(LinearLayout.VERTICAL);
    lay.setPadding(0, 30, 0, 0);
    lay.setGravity(Gravity.CENTER_HORIZONTAL);
    lay.addView(rg);

    final TextView labelThresh = new TextView(getActivity());
    labelThresh.setText("Binary Threshold:");
    // Set up the input
    final EditText binThresh = new EditText(getActivity());
    binThresh.setBackground(null);
    // Specify the type of input expected; this, for example,
sets the input as a password, and will mask the text
    binThresh.setInputType(InputType.TYPE_CLASS_NUMBER);
    binThresh.setText(Integer.toString(mBinaryThreshold));
    lay.addView(labelThresh);
    lay.addView(binThresh);

    builder.setView(lay);

    // Set up the buttons
    // builder.setPositiveButton("OK", new
    DialogInterface.OnClickListener() {
    // @Override
    // public void onClick(DialogInterface dialog, int which)

```

```

{
//          mSelectedTrackerPre = mSelectedTracker;
//          mSelectedTracker = mRadioBtnNames[mRadioIndex];
//          if (!mSelectedTracker.equals("None")) {
//              ((Button)
getActivity().findViewById(R.id.shootBtn)).setBackgroundResource(R.dr
awable.my_button_bg_2);
//              ((Button)
getActivity().findViewById(R.id.shootBtn)).setTextColor(Color.rgb(0,0
,255));
//              } else {
//              ((Button)
getActivity().findViewById(R.id.shootBtn)).setBackgroundResource(R.dr
awable.my_button_bg);
//              ((Button)
getActivity().findViewById(R.id.shootBtn)).setTextColor(Color.rgb(255
,255,255));
//              }
//              mBinaryThreshold =
Integer.parseInt(binThresh.getText().toString());
//              }
//          });

        builder.setCancelable(false);
        Dialog dialog = builder.show();
    }

    private void processing() {

        int overlayWidth = mTrackingOverlay.getWidth();
        int overlayHeight = mTrackingOverlay.getHeight();
        mRadiusCircle = 0;

        if(mObjDet) {
            if (MyConstants.DEBUG) {
                detectorSSD.setBitmap(mBitmapDebug);
            } else {
                detectorSSD.setBitmap(mBitmapGrab);
            }
        }

        if(mSelectedTracker.equals("LaneTracking")){
            if(mMatGrab==null){
                mMatGrab = new Mat();
            }
            Utils.bitmapToMat(mBitmapGrab, mMatGrab);
            org.opencv.imgproc.Imgproc.resize(mMatGrab, mMatGrab, new
org.opencv.core.Size(320,240));
            Mat gray = new Mat();
            Mat binary = new Mat();
            org.opencv.imgproc.Imgproc.cvtColor(mMatGrab, gray,
Imgproc.COLOR_RGBA2GRAY);
            org.opencv.imgproc.Imgproc.threshold( gray, binary,
mBinaryThreshold, 255, 0 );

            int y = binary.rows()*2/3;
            int x0 = -1;
            for(int x = 0; x < binary.cols(); ++x){
                if(binary.get(y,x)[0] < 125){
                    x0 = x;

```



```

        break;
    }
}

if(x0 < 0){
    mUdpClient.sendBytes(mServerAddr,mServerPort,
mRequestForwardTrack);
}else if(x0 < binary.width()/2){
    mUdpClient.sendBytes(mServerAddr,mServerPort,
mRequestRightTrack);
}else if(x0 > binary.width()/2) {
    mUdpClient.sendBytes(mServerAddr,mServerPort,
mRequestLeftTrack);
}

//TODO: DEBUG
Mat tmp = new Mat();
try {
    Imgproc.cvtColor(binary, tmp,
Imgproc.COLOR_GRAY2BGRA);
    mBitmapLaneTracking = Bitmap.createBitmap(tmp.cols(),
tmp.rows(), Bitmap.Config.ARGB_8888);
    Utils.matToBitmap(tmp, mBitmapLaneTracking);
}
catch (CvException e){
    Log.d("Exception",e.getMessage());
}
}else if(mSelectedTracker.equals("ColorTracking")){
    if(mMatGrab==null){
        mMatGrab = new Mat();
    }
    Utils.bitmapToMat(mBitmapGrab, mMatGrab);
    org.opencv.imgproc.Imgproc.resize(mMatGrab, mMatGrab, new
org.opencv.core.Size(320,240));
    Mat gray = new Mat();
    org.opencv.imgproc.Imgproc.cvtColor(mMatGrab, gray,
Imgproc.COLOR_RGBA2GRAY);
    Mat circles = new Mat();
    org.opencv.imgproc.Imgproc.HoughCircles(gray, circles,
Imgproc.CV_HOUGH_GRADIENT, 2, gray.rows()/4, 200, 120, 10, 80);
    for (int i = 0; i < circles.cols(); i++) {
        double[] vCircle = circles.get(0, i);

        org.opencv.core.Point pt = new
org.opencv.core.Point((int)Math.round(vCircle[0]),
(int)Math.round(vCircle[1]));
        int radius = (int)Math.round(vCircle[2]);

        org.opencv.imgproc.Imgproc.circle(gray, pt, radius,
new Scalar(255, 0, 0), 2);
        mPointCircle.x = (int)(overlayWidth-
pt.y*overlayWidth/gray.rows());
        mPointCircle.y =
(int)(pt.x*overlayHeight/gray.cols());
        mRadiusCircle = radius*overlayWidth/gray.rows();
    }
    //TODO: DEBUG
    Bitmap bmp = null;
    Mat tmp = new Mat();
    try {
        Imgproc.cvtColor(gray, tmp, Imgproc.COLOR_GRAY2BGRA);

```

```

        bmp = Bitmap.createBitmap(tmp.cols(), tmp.rows(),
Bitmap.Config.ARGB_8888);
        Utils.matToBitmap(tmp, bmp);
    }
    catch (CvException e){
        Log.d("Exception",e.getMessage());
    }
}
else if(mTargetLocked &&
mSelectedTracker.equals("ObjectTracking")) {
    Utils.bitmapToMat(mBitmapGrab, mMatGrab);
    org.opencv.imgproc.Imgproc.resize(mMatGrab, mMatGrab, new
org.opencv.core.Size(320,240));
    org.opencv.imgproc.Imgproc.cvtColor(mMatGrab, mMatGrab,
Imgproc.COLOR_RGBA2BGR);

    if(mDrawing==Drawing.DRAWING) {

        int imgWidth = mMatGrab.cols();
        int imgHeight = mMatGrab.rows();

        int x0 = mPoints[0].y;
        int y0 = overlayWidth - mPoints[0].x;
        int x1 = mPoints[1].y;
        int y1 = overlayWidth - mPoints[1].x;

        int minX = (int)((float)Math.min(x0,
x1)/overlayHeight*mMatGrab.cols());
        int minY = (int)((float)Math.min(y0,
y1)/overlayWidth*mMatGrab.rows());
        int maxX = (int)((float)Math.max(x0,
x1)/overlayHeight*mMatGrab.cols());
        int maxY = (int)((float)Math.max(y0,
y1)/overlayWidth*mMatGrab.rows());

        mInitRectangle = new org.opencv.core.Rect2d(minX,
minY, maxX-minX, maxY-minY);
        mMatGrabInit = new Mat();
        mMatGrab.copyTo(mMatGrabInit);

        if(mSelectedTracker.equals("TrackerMedianFlow")) {
            mTracker = TrackerMedianFlow.create();
        }else
if(mSelectedTracker.equals("TrackerCSRT") || mSelectedTracker.equals("O
bjectTracking")) {
            mTracker = TrackerCSRT.create();
        }else if(mSelectedTracker.equals("TrackerKCF")) {
            mTracker = TrackerKCF.create();
        }else if(mSelectedTracker.equals("TrackerMOSSE")) {
            mTracker = TrackerMOSSE.create();
        }else if(mSelectedTracker.equals("TrackerTLD")) {
            mTracker = TrackerTLD.create();
        }else if(mSelectedTracker.equals("TrackerMIL")) {
            mTracker = TrackerMIL.create();
        }

        mTracker.init(mMatGrabInit, mInitRectangle);
        mDrawing = Drawing.TRACKING;

        //TODO: DEBUG
        org.opencv.core.Rect testRect = new
org.opencv.core.Rect(minX, minY, maxX-minX, maxY-minY);

```

```

//          Mat roi = new Mat(mMatGrab, testRect);
//          Bitmap bmp = null;
//          Mat tmp = new Mat (roi.rows(), roi.cols(),
CvType.CV_8U, new Scalar(4));
//          try {
//              Imgproc.cvtColor(roi, tmp,
Imgproc.COLOR_RGB2BGRA);
//              bmp = Bitmap.createBitmap(tmp.cols(),
tmp.rows(), Bitmap.Config.ARGB_8888);
//              Utils.matToBitmap(tmp, bmp);
//          }
//          catch (CvException e){
//              Log.d("Exception",e.getMessage());
//          }

        }else{
            org.opencv.core.Rect2d trackingRectangle = new
org.opencv.core.Rect2d(0, 0, 1,1);
            mTracker.update(mMatGrab, trackingRectangle);

//          //TODO: DEBUG
//          org.opencv.core.Rect testRect = new
org.opencv.core.Rect((int)trackingRectangle.x,
//          (int)trackingRectangle.y,
//          (int)trackingRectangle.width,
//          (int)trackingRectangle.height);
//          Mat roi = new Mat(mMatGrab, testRect);
//          Bitmap bmp = null;
//          Mat tmp = new Mat (roi.rows(), roi.cols(),
CvType.CV_8U, new Scalar(4));
//          try {
//              Imgproc.cvtColor(roi, tmp,
Imgproc.COLOR_RGB2BGRA);
//              bmp = Bitmap.createBitmap(tmp.cols(),
tmp.rows(), Bitmap.Config.ARGB_8888);
//              Utils.matToBitmap(tmp, bmp);
//          }
//          catch (CvException e){
//              Log.d("Exception",e.getMessage());
//              mTargetLocked = false;
//              mDrawing = Drawing.DRAWING;
//          }

            mPoints[1].x = overlayWidth -
(int) (trackingRectangle.y*(float)overlayWidth/(float)mMatGrab.rows())
;
            mPoints[0].y =
(int) (trackingRectangle.x*(float)mTrackingOverlay.getHeight()/(float)
mMatGrab.cols());
            mPoints[0].x = mPoints[1].x -
(int) (trackingRectangle.height*(float)mTrackingOverlay.getWidth()/(fl
oat)mMatGrab.rows());
            mPoints[1].y = mPoints[0].y
+ (int) (trackingRectangle.width*(float)mTrackingOverlay.getHeight()/(f
loat)mMatGrab.cols());

            int centerX = (mPoints[0].x+mPoints[1].x)/2;
            int centerY = (mPoints[0].y+mPoints[1].y)/2;

```

```

        if(centerX-mTrackingOverlay.getWidth()/2 > 150){
            mUdpClient.sendBytes(mServerAddr, mServerPort,
mRequestCamUp);
        }else if(centerX-mTrackingOverlay.getWidth()/2 < -
150){
            mUdpClient.sendBytes(mServerAddr, mServerPort,
mRequestCamDown);
        }else{
            mUdpClient.sendBytes(mServerAddr, mServerPort,
mRequestCamStill);
        }

//            if(centerY-mTrackingOverlay.getHeight()/2 > 200){
//                Log.d(TAG, ": " + (centerY-
mTrackingOverlay.getHeight()/2) );
//                mUdpClient.sendBytes(mServerAddr, mServerPort,
mRequestRightTrack);
//            }else if(centerY-mTrackingOverlay.getHeight()/2 < -
200){
//                Log.d(TAG, ": " + (centerY-
mTrackingOverlay.getHeight()/2) );
//                mUdpClient.sendBytes(mServerAddr, mServerPort,
mRequestLeftTrack);
//            }

            mTrackingOverlay.postInvalidate();
        }
    }else{
        if(mSelectedTrackerPre != "None"){
            mUdpClient.sendBytes(mServerAddr, mServerPort,
mRequestStop);
        }
        if (mTracker != null) {
            mTracker.clear();
            mTracker = null;
        }
    }
    mSelectedTrackerPre = mSelectedTracker;
    mTrackingOverlay.invalidate();
}

public void onDestroy() {
    Log.e(TAG, "onDestroy");
    detectorSSD.requestStop();
    detectorSSD.waitForExit();
    mWebSocketClient.close();
    super.onDestroy();
}

public String CalculateCel(int tempInC){
    tempInC -= 32;
    tempInC /= 1.8000;

    return String.valueOf(Math.abs(tempInC));
}

public void CompilePacket(DatabaseReference dbRef){
    packet = laser * 128 + servo * 16 + speed * 8 + motors;
    dbRef.child("carControl").setValue(packet);
}

}

```


MainActivity – קוד

```
package com.p4f.esp32camai;

import android.os.Bundle;
import android.view.Menu;

import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.fragment.app.FragmentManager;

public class MainActivity extends AppCompatActivity implements
FragmentManager.OnBackStackChangedListener{
    private static final String TAG = "MainActivity";
    private String mSelectedCam = "";
    private String mSelectedCamPre = "";
    Bundle mSavedInstanceState;
    boolean mConnected = false;
    private Esp32CameraFragment mFragmentCam = null;
    private Menu mMenu = null;
    Esp32CameraFragment cameraFragment;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mSavedInstanceState = savedInstanceState;

        getSupportFragmentManager().addOnBackStackChangedListener(this);
        cameraFragment = new Esp32CameraFragment();
        if (savedInstanceState == null)

        getSupportFragmentManager().beginTransaction().add(R.id.fragment,
        cameraFragment, "camera").commit();
        else
            onBackStackChanged();
    }

    @Override
    public void onWindowFocusChanged(boolean hasFocus) {
        super.onWindowFocusChanged(hasFocus);
        cameraFragment.onWindowFocusChanged();
    }

    @Override
    public void onBackStackChanged() {

        getSupportFragmentManager().setDisplayHomeAsUpEnabled(getSupportFragmentManager().getBackStackEntryCount()>0);
    }

    @Override
    public boolean onSupportNavigateUp() {
        onBackPressed();
        return true;
    }

    // @Override
    // public boolean onCreateOptionsMenu(Menu menu) {
    //     // Inflate the menu; this adds items to the action bar if
    //     it is present.
```

```
//      getMenuInflater().inflate(R.layout.menu_main, menu);  
//      mMenu = menu;  
//      return true;  
//  }  
}
```


fragment camera xml - קוד

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:id="@+id/linearLayout"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:background="#a6dfdfff"
android:orientation="vertical"
android:weightSum="10">

<ImageView
    android:id="@+id/imageView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="256dp"
    android:gravity="center|center_vertical"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="@+id/tracking_overlay"
    app:layout_constraintHorizontal_bias="0.138"
    app:layout_constraintStart_toStartOf="parent" />

<com.p4f.esp32camai.OverlayView
    android:id="@+id/tracking_overlay"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center_vertical"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.686"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.109" />

<io.github.controlwear.virtual.joystick.android.JoystickView
    android:id="@+id/joystick"
    android:layout_width="234dp"
    android:layout_height="229dp"
    app:JV_backgroundColor="#000000"
    app:JV_borderColor="#454545"
    app:JV_borderWidth="4dp"
    app:JV_buttonColor="#FFFFFF"
    app:JV_buttonSizeRatio="15%"
    app:JV_fixedCenter="false"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintStart_toStartOf="parent" />

<com.p4f.esp32camai.VerticalButton
    android:id="@+id/objDetBtn"
    android:layout_width="68dp"
    android:layout_height="87dp"
    android:background="@drawable/my_button_bg"
    android:minHeight="80dip"
    android:text="Object Detection"
    android:textColor="#ffffff"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="@+id/tracking_overlay"
    app:layout_constraintHorizontal_bias="0.688"
```

```

        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.981" />

<com.p4f.esp32camai.VerticalButton
    android:id="@+id/streamBtn"
    android:layout_width="68dp"
    android:layout_height="87dp"
    android:background="@drawable/my_button_bg"
    android:minHeight="80dip"
    android:text="Stream"
    android:textColor="#ffffff"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.906"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.838" />

<com.p4f.esp32camai.VerticalButton
    android:id="@+id/ledBtn"
    android:layout_width="68dp"
    android:layout_height="87dp"
    android:background="@drawable/my_button_bg"
    android:minHeight="80dip"
    android:text="LED"
    android:textColor="#ffffff"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.688"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.838" />

<!-- <com.p4f.esp32camai.VerticalButton-->
<!--     android:id="@+id/shootBtn"-->
<!--     android:layout_width="68dp"-->
<!--     android:layout_height="87dp"-->
<!--     android:background="@drawable/my_button_bg"-->
<!--     android:minHeight="80dip"-->
<!--     android:text="Shoot"-->
<!--     android:textColor="#ffffff"-->
<!--     app:layout_constraintBottom_toBottomOf="parent"-->
<!--     app:layout_constraintEnd_toEndOf="parent"-->
<!--     app:layout_constraintHorizontal_bias="0.906"-->
<!--     app:layout_constraintStart_toStartOf="parent"-->
<!--     app:layout_constraintTop_toTopOf="parent"-->
<!--     app:layout_constraintVertical_bias="0.981" />-->

<TextView
    android:id="@+id/tempView"
    android:layout_width="73dp"
    android:layout_height="66dp"
    android:rotation="90"
    android:text="@string/_255"
    android:textColor="@android:color/black"
    android:textSize="24sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.976"
    app:layout_constraintStart_toStartOf="parent"

```

```

        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.021" />

<ImageButton
    android:id="@+id/shootBtn"
    android:layout_width="68dp"
    android:layout_height="87dp"
    android:contentDescription="Shoot button"
    android:tooltipText="Shoot"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.906"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.981"
    app:srcCompat="@android:drawable/ic_menu_compass" />

<com.google.android.material.slider.Slider
    android:id="@+id/servoSlider"
    android:layout_width="304dp"
    android:layout_height="48dp"
    android:layout_marginEnd="232dp"
    android:contentDescription="@string/sliderservo"
    android:rotation="90"
    android:stepSize="10"
    android:value="0"
    android:valueFrom="0"
    android:valueTo="70"
    app:haloColor="@android:color/white"
    app:labelBehavior="floating"

    app:layout_constraintBottom_toBottomOf="@+id/tracking_overlay"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toTopOf="@+id/tracking_overlay"
    app:layout_constraintVertical_bias="0.254"
    app:thumbColor="@color/colorAccent"
    app:thumbRadius="11dp"
    app:tickColor="@android:color/white"
    app:tickVisible="false"
    app:trackColor="@color/colorAccent"
    app:trackHeight="9dp" />

</androidx.constraintlayout.widget.ConstraintLayout>

```

activity_main.xml - קוד

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.coordinatorlayout.widget.CoordinatorLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <RelativeLayout
        android:id="@+id/fragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:layout_behavior="@string/appbar_scrolling_view_behavior">

    </RelativeLayout>

</androidx.coordinatorlayout.widget.CoordinatorLayout>
```

Settings.gradle - קוד

```
include ':ESP32CamAI'
rootProject.name='ESP32CamAI'
include ':openCVLibrary348'
```

colors.xml – קוד

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#000080</color>
    <color name="colorPrimaryDark">#228b22</color>
    <color name="colorAccent">#228b22</color>

    <color name="colorRecieveText">#00FF00</color>
    <color name="colorSendText">#82CAFF</color>
    <color name="colorStatusText">#FFDB58</color>
</resources>
```

strings.xml - קוד

```
<resources>
    <string name="app_name">ESP32 AI Camera</string>
    <string name="camera_permission_title">Camera permission
required</string>
    <string name="camera_permission_message">This app uses camera
only for object detection and segmentation, the information from your
camera is not sent or collected anywhere else</string>
    <string name="sliderservo">sliderservo</string>
    <string name="SliderServo">SliderServo</string>
    <string name="_255">255</string>

</resources>
```

קוד - styles xml

```
<resources>

    <!-- Base application theme. -->
    <style name="AppTheme"
parent="Theme.MaterialComponents.NoActionBar">
        <!-- Customize your theme here. -->
        <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
        <item name="colorAccent">@color/colorAccent</item>
    </style>

</resources>
```

קוד - arrays xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="newline_names">
        <item>LF</item>
        <item>CR+LF</item>
    </string-array>
</resources>
```


תיעוד תקלות ותיקון

תקלת ESP32 – בזמן שניסינו לצרוב על הבקר קיבלנו הודעת שגיאה לאחר מכן הבנו שצריך למחוק את הזיכרון פלאש של הבקר וניסינו בעזרת VSCODE אבל זה לא עבד, לבסוף החלפנו לבקר ESP32 חדש.

תקלת מנועי DC והדקים – בתוכנית PWM_ctrl יש יציאות של קדימה ואחורה של כל מנוע, פיזית לא ידענו איזה יציאות זה אחורה או קדימה, זה כל פעם חיברנו בדרך אחת וניסינו לראות עם כל הסיביות עוברות כמו שצריך והרכב זו כפי שרצינו, ואם לא פשוט חיברנו את ההדקים בדרך שונה עד שחיברנו להדקים הנכונים.

תקלת חוטים – החוטים שהשתמשנו בהם לפעמים היו רופפים והיינו צריכים להחליף אותם, גם בשביל לחבר את הכניסות של המנועים למסרקים של האלטרנה וה – ESP32 היינו צריכים להלחים אותם.

תקלת מתג הריגה – המתג שקיבלנו בשביל להפעיל ולכבות את המערכת היה נורא קטן, צפוף ושביר לכן כשהלחמנו עליו את הכבלים הנחוצים למערכת הם התנתקו יותר מידי פעמים וזה הגיע למצב שאחד מאיתנו החזיק את הכבלים שיהיו במגע והשני מפעיל את המערכת.

תיעוד תהליך עבודה

תאריך 8.9.2022: יצירת תופס פרויקט למשרד החינוך.

תאריך 11.9.2022: סיום סיכום רכיבים ופרוטוקולים.

תאריך 18.9.2022: יצירת מעגלים חשמליים ב-EASYEDA.

תאריך 20.9.2022: בניית תכנון היררכי לתקשורת בין אלטרה ל-ESP32 והוספת קוד העברת מידע והתחברות ל-WIFI ב-esp32.

תאריך 29.9.2022: יצירת FIREBASE של גוגל בכללי וגם בתוך ה-code Studio Visual - וה Android Studio. כך שנוכל לקשר בין דגם הרובוט ל-FIREBASE ונוכל לשלוט עליו כבר דרך הענן של גוגל והאפליקציה.

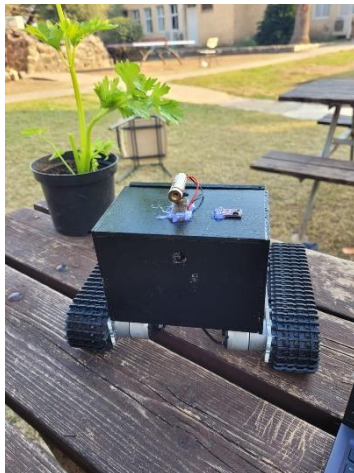
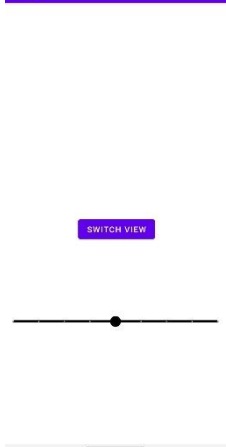
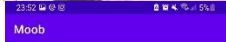
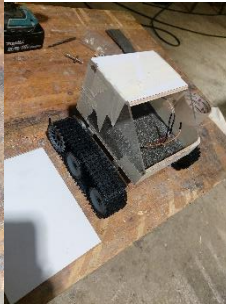
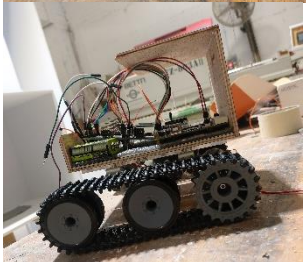
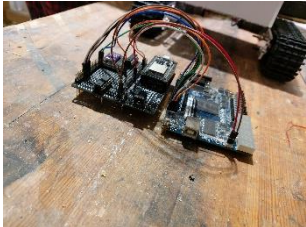
תאריך 8.10.2022: קבלת אלטרה + Esp32 וחיבור תקשורת ביניהם.

תאריך 5.12.2022: קבלת דגם הרכב עם מנועי DC והרכבתו

תאריכים 8.10.2022 – 8.1.2022 בניית תוכניות VHDL בתכנון ההיררכי בשביל הוספת טמפרטורה, מנועים, וסוללה, ובנוסף התחלת בניית האפליקציה ומבנה הרכב.

תאריך 23.1.2022: הוספת מנוע סרבו עם לייזר מודבק עליו ובניית תוכניות VHDL מתאימות.

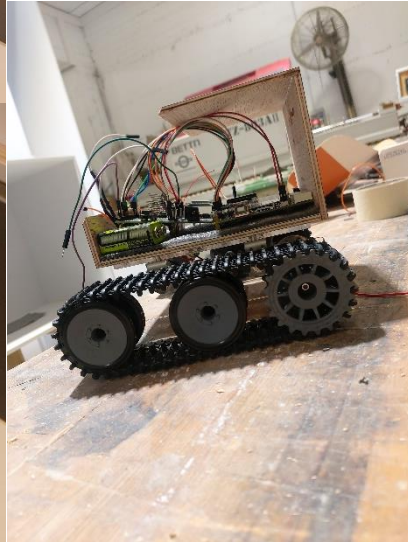
תאריך 15.3.2022: שינוי קוד ה-esp32 לקוד שעובד עם הפיירבייס ללא דיליי.



בניית הדגם

יצירת פלטפורמה מעץ שמודבק עם דבק וסיכות בשביל הדגם עם ריפוד מבפנים.

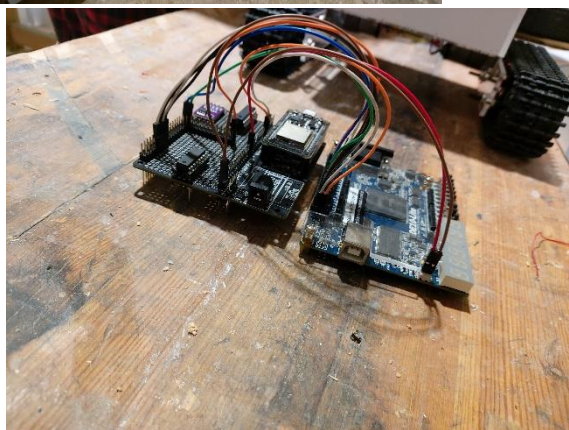
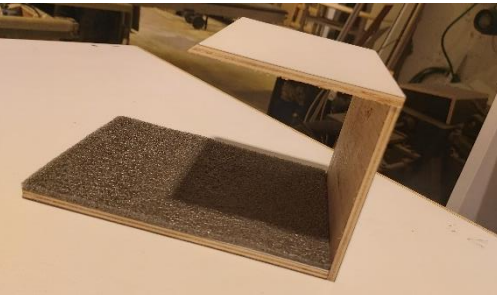
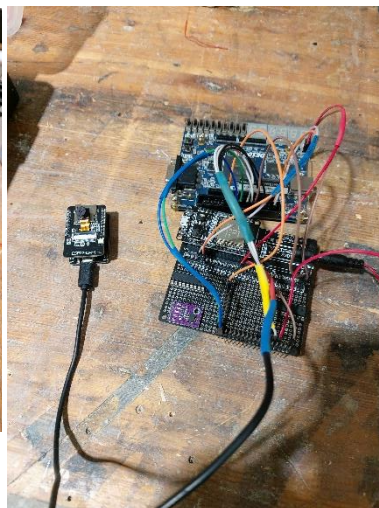
חיבור הפלטפורמה מעץ למנועים ובדיקת חלל בשביל אלטרה ובקר ESP.

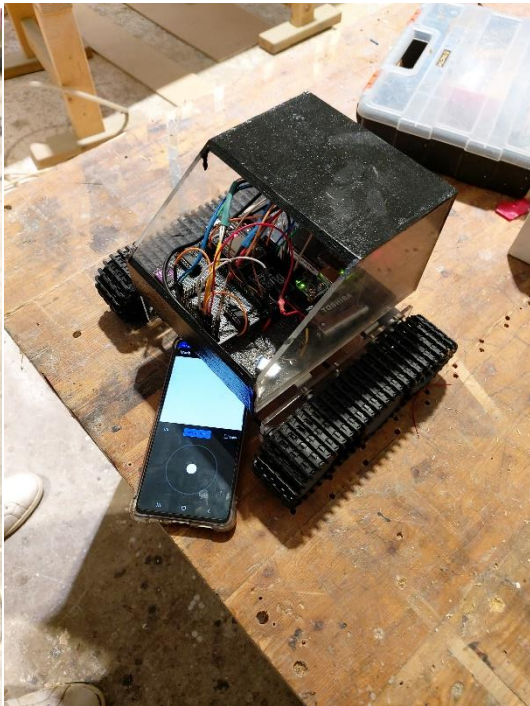


הדבקת קירות מחומר אקרילי שקוף בשביל לראות עיצוב מבפנים וקדיחת חורים בשביל חוטים של המנועים.



סידור כבלים עם שרינקים לפני ואחרי:





צביעת הדגם בשחור
והדגם עם
האפליקציה:

רפלקציה

עידן

אחד ההיבטים המאתגרים בפרויקט זה הוא בניית הרכיבים המכניים של המכונות. תכנון והרכבת רכב שיכול לנהוג בצורה חלקה וזו משימה מורכבת הדורשת תשומת לב לפרטים ודיוק. בנוסף, חשוב לוודא שהרכיבים החשמליים והאלקטרוניים של המכונות משולבים כראוי ומוגנים מפני נזק.

היבט מאתגר נוסף הוא תכנות המכונות לשליחת נתונים לאפליקציה. זה דורש הבנה טובה של שפות קוד ויכולת לעבוד עם חיישנים, מנועים ורכיבים אלקטרוניים אחרים. הנתונים המועברים מהמכונות לאפליקציה עשויים לכלול מידע על טמפרטורת הסביבה ומצלמה שניתן לראות איתה. לוודא שהנתונים האלה מדויקים ומועברים בצורה מהימנה יכולה להיות משימה לא פשוטה.

בסך הכל, עבודה על פרויקט אלקטרוניקה כמו בניית מכונות שיכולה לנהוג ולשלוח נתונים לאפליקציה יכולה להיות מאתגרת ומתגמלת כאחד. הפרויקט דורש שילוב של מיומנויות מכניות, אלקטרוניות ותכנות, והשלמה מוצלחת יכולה לספק תחושת סיפוק עמוקה. עם מסירות, התמדה ותשומת לב לפרטים, כל אחד יכול לקחת על עצמו פרויקט כזה ולהצליח.

אור-ים

תהליך יצור הפרויקט היה מהנה ברובו, נהניתי לעשות עבודת צוות עם השותף, החלק שהכי זרם לנו היה העבודה הפיזית ולא התיעוד, אם הייתי משנה משהו בפרויקט הזה זה רק התכנון שלו חוץ מזה הכל היה כיף, מהיר וחוויתי.

ביבליוגרפיה

: L293D – דוחף זרם

<https://pdf1.alldatasheet.com/datasheet-pdf/view/89353/TI/L293D.html>

: 18B20 – חיישן טמפרטורה

<https://pdf1.alldatasheet.com/datasheet-pdf/view/58559/DALLAS/18B20.html>

: HC385MG-301 – DC מנוע

<https://disti-assets.s3.amazonaws.com/testco-inc/files/datasheets/25281.pdf>

: ESP32 – בקר

https://espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf

: ESP32 – CAM

<https://media.digikey.com/pdf/Data%20Sheets/DFRobot%20PDFs/DFR0602Web.pdf>

[/https://nugroho.xyz/mendeteksi-sentuhan-dengan-esp32-cam](https://nugroho.xyz/mendeteksi-sentuhan-dengan-esp32-cam)

: L293D הדקים ומידע

<https://components101.com/ics/l293d-pinout-features-datasheet>

מידע על הדקים ופעולות ESP32 :

https://www.circuitschools.com/what-is-esp32-how-it-works-and-what-you-can-do-with-esp32/#Peripheral_Features

חיישן טמפרטורה – LM75 :

<https://datasheets.mxicomintegrated.com/en/ds/LM75.pdf>

דוחף זרם – ULN2803 :

<https://www.alldatasheet.com/datasheet-pdf/pdf/12687/ONSEMI/ULN2803.html>

תקלת עיבוד תמונה :

<https://stackoverflow.com/questions/67448987/indexerror-list-index-out-of-range-object-detection>

קוד עיבוד תמונה

<https://www.computervision.zone/topic/nms-code/>

מנוע סרבו

<https://components101.com/motors/mg90s-metal-gear-servo-motor>

https://www.electronicoscaldas.com/datasheet/MG90S_Tower-Pro.pdf