אוניברסיטת בן-גוריון בנגב
Ben-Gurion University of the Negev
בית הספר להנדסת חשמל ומחשבים

דו"ח תיעוד פרויקט גמר קורס "מבנה מחשבים ספרתיים" 361-1-4191

תכנון ומימוש מערכת בקרה למכונה מבוססת מנוע צעד בשליטה ידנית ומרחוק

עומר קריטי - 208686352
ירין עוזיאל - 319149878

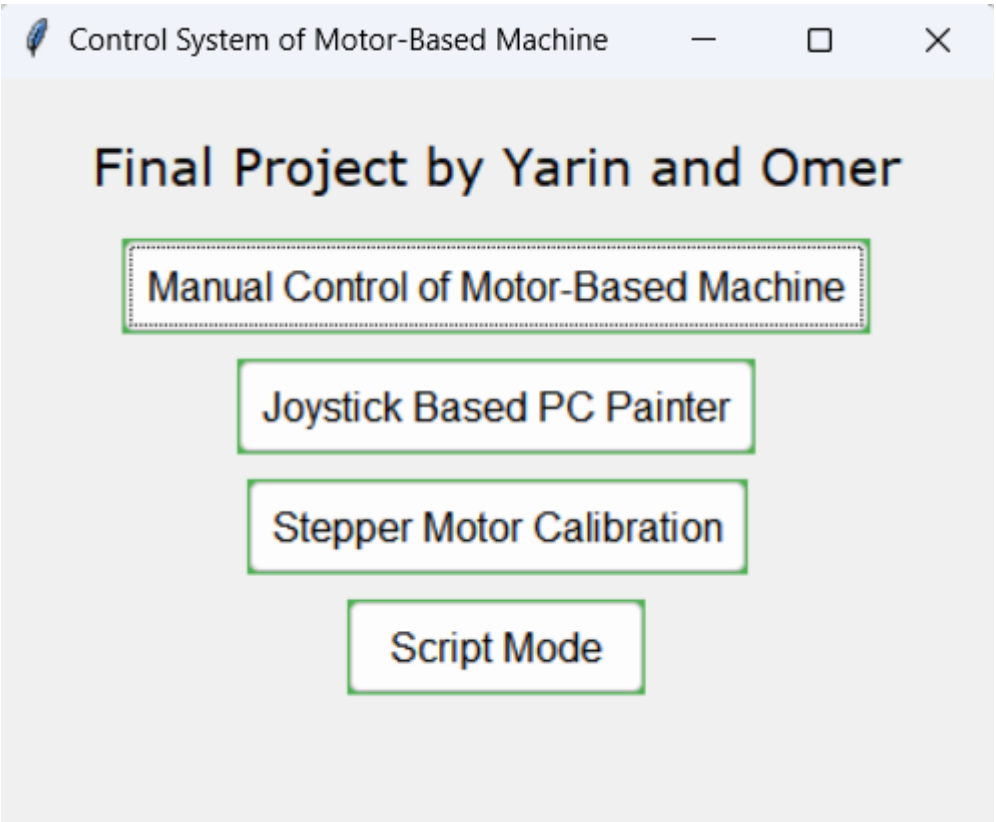
02.09.2024

הגדרת מטרת הפרויקט-

פרויקט הגמר עוסק בפיתוח מערכת משובצת מחשב המבוססת על מיקרו-בקר (MCU), שמטרתה לשלוט מרחוק על מכונה באמצעות שליטה ידנית דרך ערוץ תקשורת סריאלית. המערכת כוללת ממשק משתמש גרפי (GUI) שמוצג במחשב אישי, לאפשר למשתמש לבצע פעולות שליטה ובקרה על המכונה באמצעות ג'ויסטיק אנלוגי. המערכת מתוכננת לפעול בזמן אמת (Hard Real Time), ותכלול שליחת קבצים, ותכנות ברמת High-level למיקרו-בקר. בנוסף, תכלול המערכת ממשק גרפי לניהול ופיקוח על הפעולות המתבצעות, תוך שימוש בתקשורת טורית סטנדרטית RS-232. כמו כן הפרויקט מבוצע על גבי הבקר MSP430G2553 (הערכה האישית) בשפת C בתוכנת CCS , ה GUI והתקשורת הטורית ממומשים ב python.

תיאור הפרויקט-

הפרויקט מתמקד בפיתוח מערכת משובצת המבוססת על מיקרו-בקר MSP430, שנועדה לשלוט במכונה מבוססת מנוע דרך ממשק משתמש גרפי במחשב האישי.
להלן תמונת ממשק המשתמש:

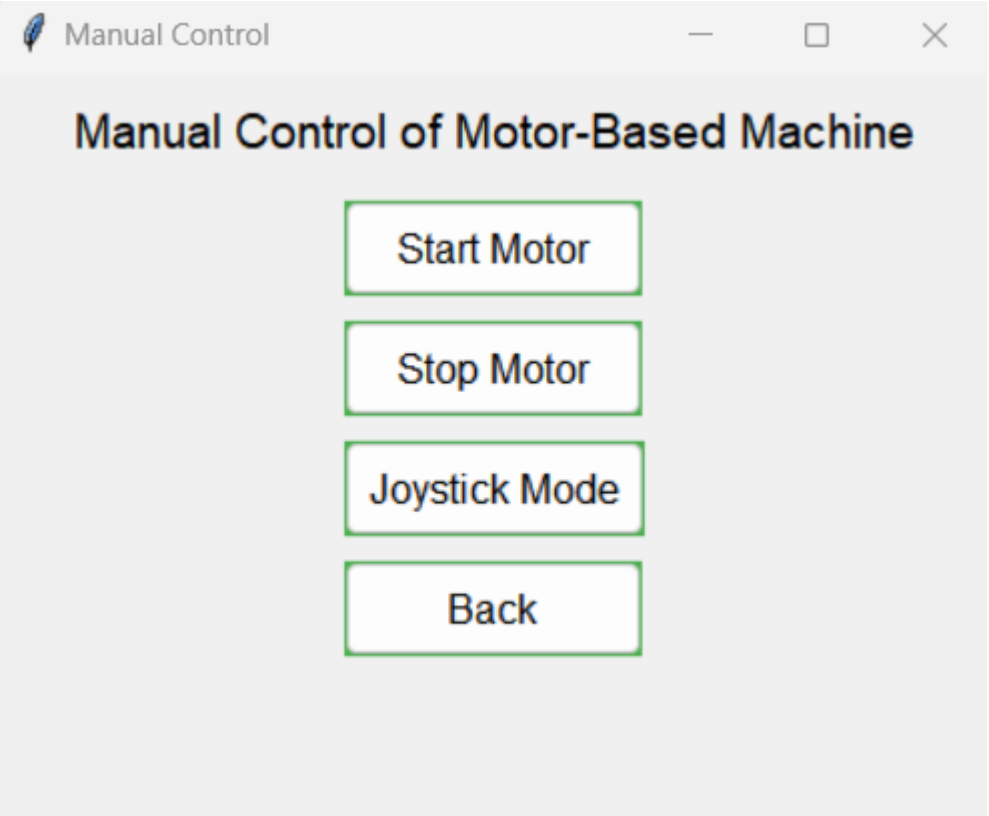


כאמור, על המשתמש לבחור את המצב הנדרש בצד המחשב וה State המתאים עובר בהתאם דרך תקשורת UART ייעודית למערכת המצבים המנוהלת בצד הבקר אשר בעקבות הפסיקה תפעיל את המצב הנדרש.

המערכת כוללת מספר מצבים ייחודיים (בפרדיגמת FSM פשוטה): שליטה ידנית במנוע ע"י הג'ויסטיק, ציור על המחשב באמצעות ג'ויסטיק, כיול מנוע צעד, ומצב סקריפט המאפשר הרצת פקודות מותאמות אישית. להלן פירוט כל אחד מהמצבים:

1. **Manual control of motor-based machine:**

להלן תמונת ממשק המשתמש:

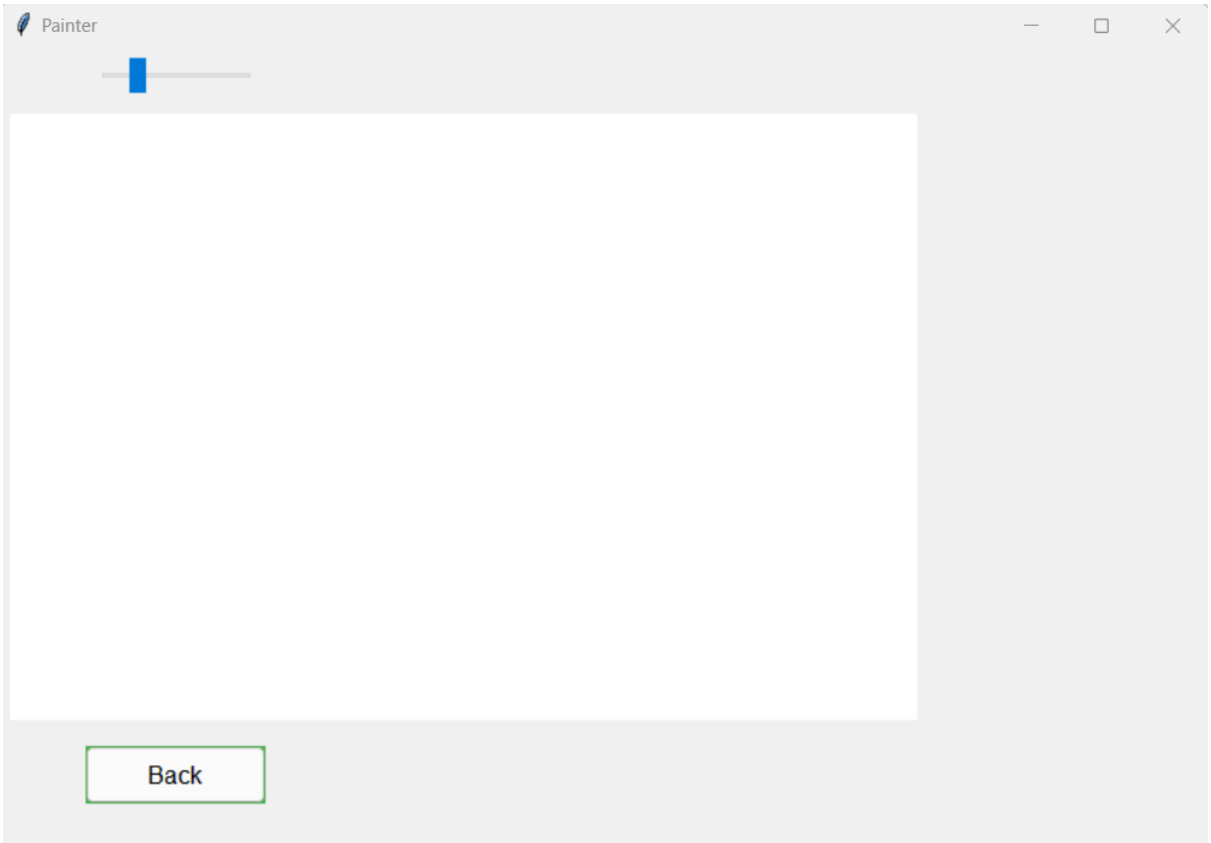


במצב זה, למשתמש ישנן 3 אופציות לבחירה - סיבוב המנוע בצורה אוטומטית , עצירת המנוע ושליטה במנוע ע"י קביעת זווית מהג'ויסטיק.
במצב סיבוב המנוע אוטומטית, נקדם את הזווית יחידה בודדת כל עוד אנחנו במצב.
בעצירת הסיבוב , נגדיר את הבקר להיכנס למצב שינה.
במצב השליטה הידנית ראשית נאתחל מצב זה להתחיל מהפס הכחול - כלומר זווית 0 למטרת השיוך בין כיול המנוע שלנו לחישוב הזווית הנחוצה.

חישוב זוויות מתבצע בצורה הבאה -
ע"פ מדידה שביצענו , נמדדו מס' הצעדים הנדרשים (חילופי פאזה) על מנת לבצע סיבוב שלם.
לפיכך, נוכל לחשב את מספר הצעדים הנדרשים להגעה לזווית רצויה.
בכל מעבר צעד כתלות בכיווניות הסיבוב , יתעדכן משתנה ייעודי ב1-+ בהתאם.
כמו כן, כיוון סיבוב המנוע על מנת להגיע לזווית הרצויה יקבע עפ המקסימום בין הזווית הנוכחית לזווית הרצויה (כך שהראשון קטן יותר נזוז עם כיוון השעון ולהפך).
הסיבוב מסתיים כאשר ישנה התאמה בין הזווית הנוכחית לזווית הרצויה.
מדידת הזווית הרצויה מהג'ויסטיק תתבצע ע"י דגימת ערכי הצירים בעזרת ADC וחישוב ערכים אלו בפונקציה ייעודית המתבסת על חלוקת אופציות הזוויות לרביעים.

2. **Joystick based PC painter:**

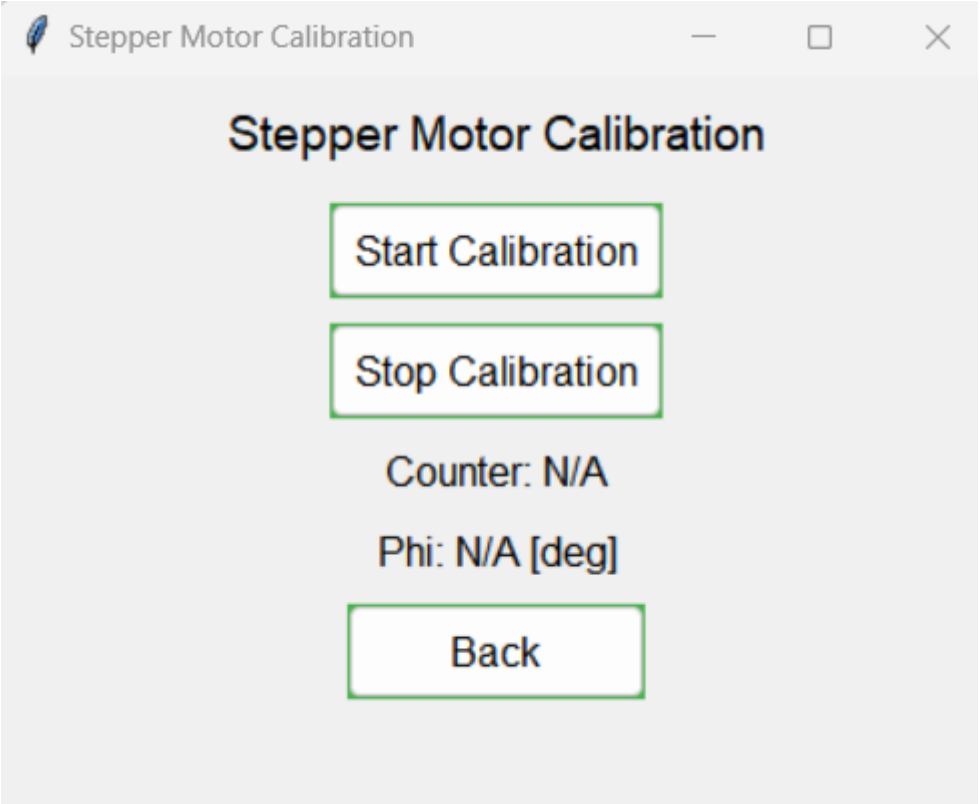
להלן תמונת ממשק המשתמש:



מצב זה מאפשר למשתמש לשלוט ביישום ציור על המחשב באמצעות ג'ויסטיק.
תנועת הג'ויסטיק משקפת את התנועה על מסך המחשב, ומאפשרת לצייר צורות גיאומטריות. המשתמש יכול לבחור בין שלושה מצבי משנה שונים - ציור, מחיקה והזזת הסמן ללא כתיבה ומחיקה.
על מנת להוריד את הדיליי בין זמן הזזת הג'ויסטיק לבין הזזת הסמן בחלון הצייר , ערכי הדגימות משודרים באופן רציף.
אופן מדידת ערכי הג'ויסטיק מומש ע"י הסבר אופן פעולתו של מתחי הצירים.
על מנת לעבור בין מצבי הסמן השונים הגדרנו ע"פ הנדרש את לחצן PB0 (לחצן הג'ויסטיק אינו פועל ונאמר כי נתן להחליף זאת)

3. **Stepper Motor Calibration:**

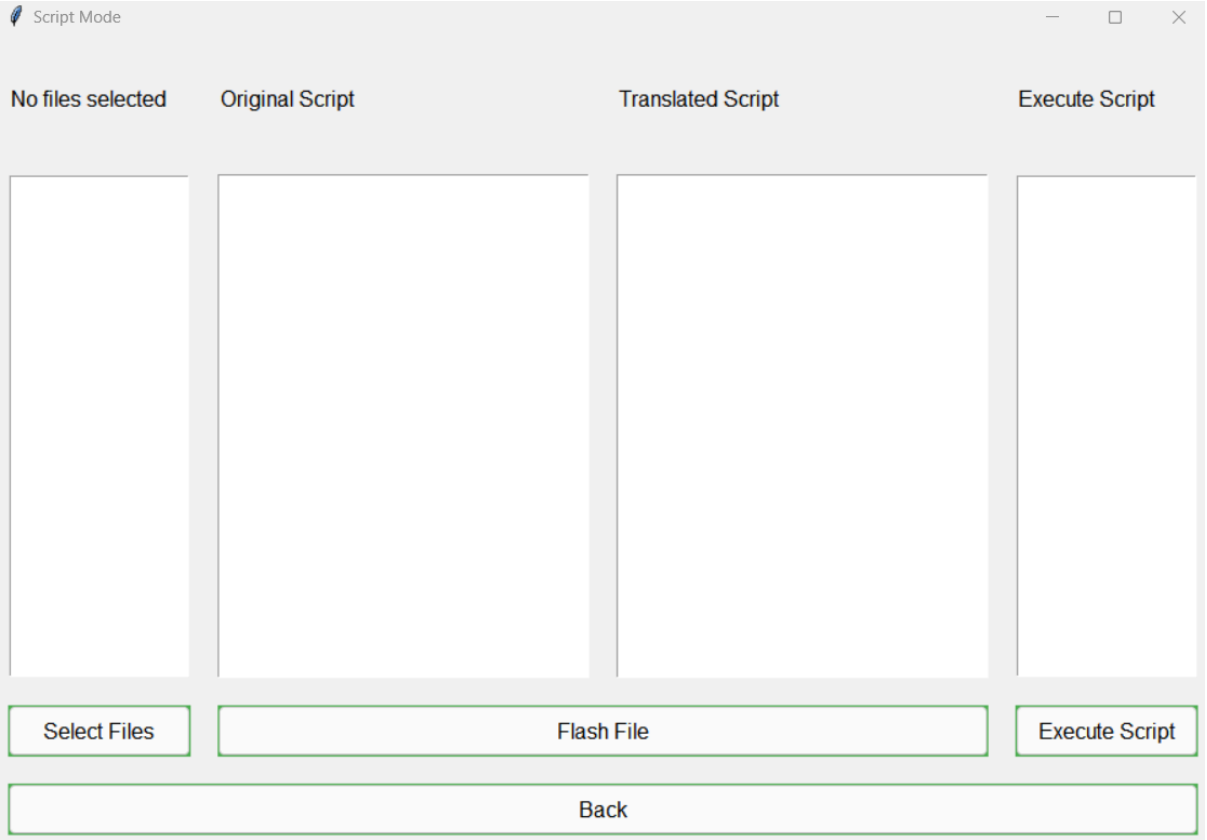
להלן תמונת ממשק המשתמש:



במצב כיול מנוע הצעד, הממשק הגרפי מציג למשתמש את מספר הצעדים שהמנוע עבר (counter) ואת הזווית המתארת את מס' הצעדים שבוצעו חלקי 360 (phi). המשתמש יכול להפעיל את המנוע ולבצע כיולים לפי הצורך, ולהבטיח שהמנוע מתפקד בצורה אופטימלית בתנאים שונים.
את המדידה נתחיל מהפס הכחול ובעזרתה נוכל לכייל את ערכי המדידות והזוויות בשאר פונקציונליות המערכת.

4. Script Mode:

להלן תמונת ממשק המשתמש:



מצב הסקריפט מאפשר למשתמש להריץ פקודות מותאמות אישית באמצעות המרת פקודות המוגדרות במשימה ל opcode ואופרנד מספרי.
להלן שלבי המצב - טעינת קובץ טקסט, צריבה לזיכרון FLASH, ביצוע הפקודות.

טעינת קובץ טקסט - ע"י לחיצה על כפתור "Select Files" ניתן לבחור קובץ אותו נרצה לשייך למערכת.
המערכת שלנו תומכת ע"פ הנדרש בטעינת 3 קבצים (עם מצביעים תואמים), בגודל מקסימלי של 64 בתים לכל קובץ.
צריבה לזיכרון FLASH - ע"י לחיצה על כפתור "Load File" , ניתן לבחור קובץ אותו נרצה לצרוב לזכרון.
ביצוע הפקודות - ע"י לחיצה על כפתור " Execute Script" , ניתן לבחור קובץ אותו נרצה שהבקר יבצע.

חלוקת העבודה בין החומרה לתוכנה-

על מנת לממש את פונקציונליות המערכת נדרשים מספר רכיבי חומרה אשר עובדים בתיאום ביניהם. בדף זה נפרט אודותיהם.

רכיבי חומרה -

המשימה מתבצעת על בסיס הערכה האישית - בקר ה MSP430.
כמו כן נעזרנו ב ADC10 על מנת לדגום את מתח הצירים של הג'ויסטיק.
בנוסף, על מנת לבצע השהיות בצורת פסיקות סיום, נעזרנו בטיימרים הבנויים במערכת בדומה לשימוש במעבדות קודמות.
את התקשורת בין צד המחשב לצד הבקר מימשנו על בסיס UART והגדרתו בוצעה בדומה למעבדה 4 בכל אחד מצדדי המערכת.
להלן הסבר אודות רכיבים חדשים איתם נדרשנו לעבוד:

מנוע צד -

מנוע צעד הינו מנוע שניתן להפעילו כך שמוט הסיבוב שלו, יסתובב בכל איטרציה בזווית φ הנקראת צעד. המנוע שלנו מקבל מקור מתח של 5v וע"י שליטה של ארבע פאזות נוכל לגרום לו להסתובב.
להלן הגדרת הפאזות המתאימה לביצוע הסיבוב:

סיבוב המנוע בצעד / חצי צעד :

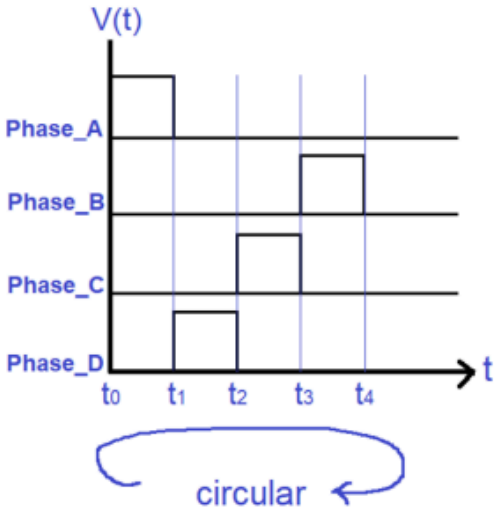
- סיבוב המנוע בצעד מלא בכיוון clockwise:

כדי להפעיל את המנוע **בצעד בודד** (סיבוב אחד בזווית **נומינאלית** $\varphi = 0.088°$) **בכיוון השעון**, נדרש להכתיב מהבקר (MCU) מתח לארבע רגליים Phase_A, Phase_B, Phase_C, Phase_D לפי הטבלה הבאה:

כדי לבצע הזזה רציפה כרצוננו, נצטרך לבצע את המתואר בטבלה בצורה מחזורית (***בשימוש פסיקות Timer בלבד***). קצב השינוי (בתחום של 5Hz-50Hz) יקבע את מהירות הסיבוב של מוט המנוע.

רגל בכרטיס הממשק המחוברת לבקר	t_0	t_1	t_2	t_3
Phase_A	1	0	0	0
Phase_B	0	0	0	1
Phase_C	0	0	1	0
Phase_D	0	1	0	0

→ t



- <u>סיבוב המנוע בצעד בכיוון counter clockwise:</u>

כדי להפעיל את המנוע **בצעד בודד** (סיבוב אחד בזווית **נומינאלית** $\varphi = 0.088°$) **בכיוון נגד כוון השעון**,
נדרש להכתיב מהבקר (MCU) מתח לארבע רגליים Phase_A, Phase_B, Phase_C, Phase_D
נפעיל את הטבלה הנ"ל בכיוון הפוך.

רגל בכרטיס הממשק המחוברת לבקר	t_0	t_1	t_2	t_3
Phase_A	0	1	0	0
Phase_B	0	0	1	0
Phase_C	0	0	0	1
Phase_D	1	0	0	0

→ t

- <u>סיבוב המנוע בחצי צעד בכיוון clockwise:</u>

כדי להפעיל את המנוע **בחצי צעד** (סיבוב אחד בזווית **נומינאלית** $\varphi = \frac{0.088°}{2}$) **בכיוון השעון** (לכיוון הפוך,
נדרש להזין את הטבלה בסדר הפוך), נבצע לפי הטבלה הבאה:

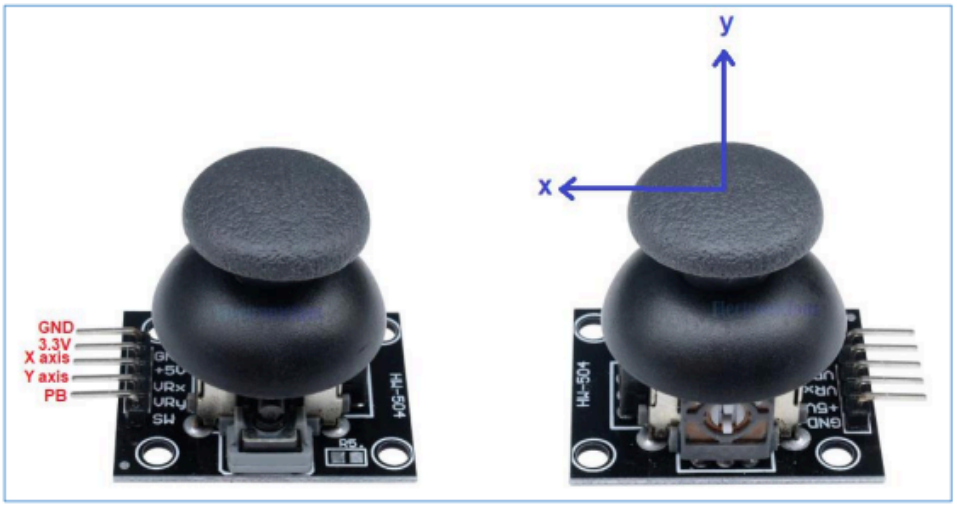
רגל בכרטיס הממשק המחוברת לבקר	t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7
Phase_A	0	0	0	0	0	1	1	1
Phase_B	0	0	0	1	1	1	0	0
Phase_C	0	1	1	1	0	0	0	0
Phase_D	1	1	0	0	0	0	0	1

→ t

ישנה מגבלת תדר למעבר בין הפאזות על מנת שהמנוע יוכל לתפקד כראוי.
על פי שליטה בתדר זה נוכל לקבוע את מהירות סיבוב המנוע.

<u>ג'ויסטיק -</u>
הג'ויסטיק הינו רכיב חומרתי המתפקד במקרה שלנו כמעין עכבר עם מערכת צירים Y, X וכפתור מרכזי . הג'ויסטיק מקבל מקור מתח של 3v .
להלן הגדרת אופן פעולתו:

- כאשר המוט במצב המקורי (לא מוטה) המתח במוצא הרגליים Vrx , Vry הוא $\frac{V_{cc}}{2} = \frac{3.3v}{2} = 1.65v$
- בהטיית המוט לכיוון y,x המתחים Vrx , Vry בהתאמה יעלו בצורה יחסית בטווח [1.65v,3.3v]
- בהטיית המוט לכיוון -y,-x המתחים Vrx , Vry בהתאמה ירדו בצורה יחסית בטווח [0,1.65v]
- לחיצה אנכית על ראש מוט ההיגוי מהווה לחיצת לחצן (בחיבור לרגל בקר MSP430 משפחה 2 השתמשו ברגיסטר PxREN לצורך הגדרת הלחצן בקונפיגורציה של Pullup/Pulldown)



זיכרון FLASH -

כחלק מדרישות המשימה, נדרשנו לנהל שמירה של עד שלושה קבצים בזיכרון ה FLASH של הבקר ולהגדיר struct מתאים המכיל את השדות הבאים - כמות קבצים קיימים, מערך מצביעים לשמות הקבצים, מערך מצביעים לתחילת כל קובץ, מערך המכיל את גודלי הקבצים.
הקבצים יכולים להכיל את סט הפקודות הבאות:

OPC (first Byte)	Instruction	Operand (next Bytes)	Explanation
0x01	inc_lcd	x	Count up from zero to **x** with delay d onto LCD
0x02	dec_lcd	x	Count down from **x** to zero with delay d onto LCD
0x03	rra_lcd	x	Rotate **right** onto LCD from pixel index 0 to pixel index 31 a single char **x** (ASCII value) with delay d
0x04	set_delay	d	Set the delay **d** value (***units of 10ms***)
0x05	clear_all_leds		Clear LCD
0x06	stepper_deg	p	Points the stepper motor pointer to degree p and show the degree (dynamically) onto PC screen
0x07	stepper_scan	l,r	Scan area between left l angle to right r angle (once) and show the start and final degrees (right on time the motor pointer reaches them) ***onto LCD screen***
0x08	sleep		Set the MCU into sleep mode

הנחת סידור הזיכרון מתבצעת בתאימות להנחה כי גודל כל סגמנט הינו 64 בתים.
אופן פעולת שימוש ה GUI התואם מתואר לעיל תחת מצב script mode.
לאחר הצריבה לFLASH נקבל ACK מהבקר , כך גם עבור תחילת ביצוע - EXECUTE נקבל ACK אודות מצביע הקובץ אותו המערכת מפעילה.

רכיבי תוכנה -
על מנת לבצע את תקשורת ה UART בין צד המחשב לצד הבקר בצורה נוחה למשתמש, כתבנו GUI התומך בכלל דרישות קלט המשתמש.
הממשק מתפקד כבורר מצבים אשר עבור לחיצת מקש מהמשתמש נשלח הסימן המתאים לבקר על מנת להפעיל את הפעולה הנדרשת, כלומר אותו הסמן, מתו
אם ע"י צד המחשב וצד הבקר להפעלת הפעולה.
הספריה המרכזית בה נעזרנו הינה tkinter.

כמו כן,
על מנת לבצע תרגום בין ערך הsrcript הגולמי לערכו בהקסה כתבנו את פונקציית translate_script המיועדת לתרגם קובץ טקסט המכיל רשימת פקודות למחרוזת הקסה-דצימלית, על פי הגדרות שנקבעו מראש. הפונקציה פותחת את קובץ הטקסט וקוראת אותו שורה אחר שורה. עבור כל שורה, היא משתמשת בפונקציה translate_line כדי לבצע את התרגום להקסה-דצימל: הפונקציה מפענחת את שם הפקודה וממפה אותה לאופקוד המתאים, ובמידת הצורך, מתרגמת גם את האופרטים הנלווים למחרוזת הקסה-דצימלית. כל תוצאה של תרגום נשמרת ברשימה, ולאחר מכן כל המחרוזות שאוגדו מחוברות יחדיו למחרוזת אחת גדולה שמייצגת את כל הפקודות בקובץ כקוד הקסה-דצימלי. בסופו של דבר, הפונקציה מחזירה את המחרוזת המאוחדת, המשקפת את התוכן המתורגם של הקובץ.

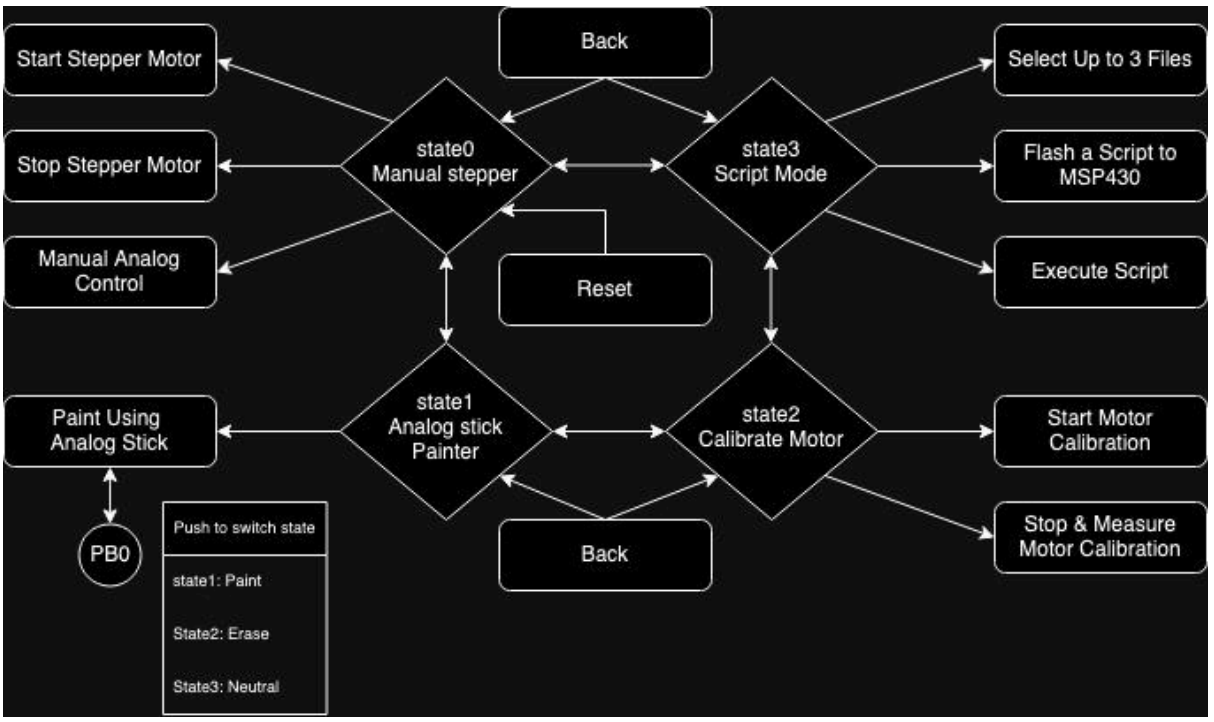
להוסיף

- תרשים זרימה ברור של תכנון ארכיטקטורת התוכנה של מימוש המערכת Embedded המחולקת לשכבות אבסטרקציה ומבוססת גרעין הפעלה Simple FSM. את התרשים נדרש לשרטט בתוכנת שרטוט כדוגמת **Draw.io** ו **sketchviz** -ראו קישורים (Draw.io , sketchviz)

  הערה: זה הוא שלב מקדים טרם גישה לכתיבת קוד המערכת! – לצורך ניפוי שגיאות כבר בשלב התכנון של המערכת (ככל שניפוי השגיאות קורה בשלב מוקדם יותר כך מחיר השגיאות יקר פחות).
- הביצועים בפועל לעומת המפרט הטכני, הסבר מפורט מה השתנה ומדוע.
- שרטוט מעגל אלקטרוני לביצוע פרויקט.
- מסקנות והצעות לשיפורים.

דיאגרמות FSM-

צד בקר:



צד מחשב: