אוניברסיטת בן-גוריון בנגב

Ben-Gurion University of the Negev

פרויקט גמר

קורס: מבנה מחשבים ספרתיים

361-1-4191

Control system of motor-based machine

תכנון ומימוש מערכת בקרה למכונה מבוססת מנוע צעד בשליטה ידנית ומרחוק

20/06/2022

תוכן עניינים:

A. מטרת הפרויקט: 3

B. דרישות משימת הפרויקט: 3

1. Manual control of motor based machine: (משקל 35%) 4

2. Joystick based PC painter: (משקל 30%) 4

3. Stepper Motor Calibration : (משקל 5%) 4

4. Script Mode: (משקל 30%) 4

C. הסברים טכניים – חיישן ומנוע סרבו: 6

1. מוט היגוי joystick: 6

2. מנוע Stepper Motor: 6

3. ממשק משתמש בצד ה- PC : 8

D. דו"ח מכין: (משקל 10%) 8

E. מבנה הציון בפרויקט: 9

A. מטרת הפרויקט:

i. תכנון ומימוש מערכת בקרה למכונה מבוססת מנוע צעד בשליטה ידנית ע"י analog joystick ובשליטה מרחוק ממחשב אישי דרך ערוץ תקשורת טורית. הדגשים בתכנון המערכת הם של רמת ביצועים גבוהה תחת משטר של Hard Real time עם רמת דיוק גבוהה.

ii. במסגרת הפרויקט יפותח קוד בשפת C++/C למימוש מערכת זמן אמת מבוססת פסיקות (צד MCU) להפעלת הרכיבים וקריאת המידע ממד המרחק.

iii. מחשב PC ישמש לצורך ממשק GUI (PySimpleGUI , Tkinter , etc) למשתמש ולתצוגה לכל פעולה המוגדרת במערכת ודורשת תצוגה וממשק למשתמש. ה- MCU יחובר למחשב ה- PC באמצעות תקשורת טורית אסינכרונית בסטנדרט RS-232.

iv. ממשק GUI למשתמש בצד ה- PC יאפשר קביעת פרמטרים, שליחת קבצים ופקודות High-level ל- MCU ותצוגת תמונת הרדאר ב- PC. הממשק בצד ה- PC ייכתב בשפה עילית (לבחירתכם: JAVA, C++, Python, Matlab, או שימוש במעטפת C#) ויתמוך במימוש של תקשורת טורית בין הבקר ל- PC.

v. הממשק יאפשר העברת קבצים הכוללים פקודות High-level מקודדות למימוש בצד הבקר. **כל הקבצים בצד בקר יישמרו בזיכרון FLASH בלבד.**

vi. הסבר מפורט של הממשק ומבנה הקבצים מתואר בפירוט בהמשך.

B. דרישות משימת הפרויקט:

- ארכיטקטורת התוכנה של המערכת נדרשת להיות מבוססת פרדיגמת תכנות מסוג ***Simple FSM*** המבצעת קטע קוד השייך לאחד ממצבי המערכת בהינתן בקשה של פסיקת RX שמגיעה מה PC לבקר דרך ערוץ התקשורת ל UART. **קוד המערכת נדרש להיות מחולק לשכבות אבסטרקציה כך שיהיה נייד (portable) בקלות בין משפחות בקר MSP430 ע"י החלפת שכבת ה- BSP בלבד.**
- טרם שלב כתיבת הקוד בשלב התכנון נדרש לשרטט גרף של דיאגרמת FSM מפורטת של ארכיטקטורת התוכנה של המערכת ולצרפה לדו"ח מכין. המצבים אלו הצמתים והקשתות אלו המעברים ממצב למצב בגין בקשות פסיקת RX (המסווגות לקליטת מידע מסוג Command ומסוג Data, כפי הנלמד בניסוי מעבדה 4).
- אסור לבצע השהייה ע"י שימוש ב poling למעט עבור debounce ברוטינת שירות של בקשות פסיקה בגין לחצנים.
- המערכת נשלטת אך ורק דרך ערוץ התקשורת בין המחשב לבקר ובשימוש joystick בלבד (אסור להשתמש במתגים, לחצנים, Keypad אלא אם כן צוין במפורש)
- **בתחילת התוכנית (בלחיצה על כפתור RESET), הבקר נמצא במצב שינה.**

  הערה: כפתור RESET מותר לשימוש אך ורק לאתחול המערכת בלבד
- רמת הדיוק וזמן תגובת המערכת בהתאם לדרישות מהווים חלק מרכזי בהערכת הפרויקט.
- מקוריות העבודה היא חלק חשוב בביצוע הפרויקט, במקרה של העתקה, הפרויקטים של שני הצדדים ייפסלו.
- עקב מגבלה של גודל ה RAM עליכם להשתמש בתבונה בזיכרון ה FLASH (ראו חומר עזר במודל).
- נדרש לעבוד בסביבת פיתוח **CCS IDE** מבוססת Eclipse (כפי הנלמד בחומר ההכנה למעבדה 1).
- לצורך חלק הביצוע של המשימה נדרש ליצור ממשק למשתמש בצד ה- PC המכיל את סעיפי התפריט הבא:

1. **Manual control of motor based machine**: **(משקל 35%)**

שליטה ידנית (ע"י analog joystick) ובצורה דינאמית של זווית ה pointer של מנוע צעד (זווית אליה מצביע מוט המחובר לידית המנוע) בהיקף של 360 מעלות, תדר הזזת המנוע בתחום של 5Hz-50Hz .

הערה חשובה: *מאחר ומדובר בבקרה בחוג פתוח, טרם ביצוע מצב זה, נדרש להביא את ה pointer של המנוע לזווית אפס ע"י הודעה מתאימה למשתמש. המשתמש נדרש לתת פקודה מתפריט המחשב לתחילת סיבוב המנוע ברצף (בתדר שהמשתמש יוכל להבחין בהגעת ה pointer לפס הכיול השחור, זווית אפס) ופקודה לסיום.*

הדגש בתכנון הוא של זמן תגובת המערכת (בין שינוי מיקום זווית ב Joystick ולבין זווית ה pointer של מנוע הצעד) תחת משטר של Hard Real time עם רמת דיוק גבוהה.

2. **Joystick based PC painter**: **(משקל 30%)**

נדרש לממש צייר על גבי מסך המחשב הנשלט ע"י analog joystick המשמש "כחוד עיפרון לצייר". לצייר ישנם שלושה מצבים הנשלטים ע"י לחיצה אנכית על ראש ה joystick: מצב כתיבה -> מצב מחיקה -> מצב ניוטרל (לצורך הזזת חוד הצייר למיקום כלשהוא על המסך).

הדגש בתכנון הוא של זמן תגובת המערכת (בין שינוי מיקום ב Joystick ולבין מיקום חוד עיפרון של הצייר) תחת משטר של Hard Real time עם רמת דיוק גבוהה *(בדיקה פשוטה היא היכולת לצייר צורות גאומטריות נפרדות בצורה סימטרית וברורה).*

3. **Stepper Motor Calibration** : **(משקל 5%)**

כיול מנוע צעד (כמפורט בסעיף C2i) ולהציג על גבי מסך המחשב את כמות הצעדים בסיבוב שלם ואת גודל זווית הצעד φ של מנוע הצעד

4. **Script Mode**: **(משקל 30%)**

הפעלת כל המערכת בהתאם לקובץ script המכיל פקודות High Level המוגדרות מראש. ניתן לתפעל את המערכת באופן אוטומטי ולבדוק את כל חלקי המערכת. נדרש לתמוך ביכולת שליחה וקבלה של עד שלושה קבצים ולבחור להפעיל אחד מהם בנפרד ובאופן בלתי תלוי בבחירה מתוך התפריט בצד מחשב בלבד.

הדגש בתכנון הוא של זמן תגובת המערכת עם רמת דיוק ביצוע של תוכן קובץ ה scripts ואמינות תוכן המידע הנשלח/מתקבל דרך ערוץ התקשורת.

הסבר:

המשתמש יוכל לשלוח לבקר קובץ script.txt המכיל פקודות ברמת High Level (כמפורט בהמשך). טעינת הקובץ נעשית בלחיצת כפתור מתאים דרך הממשק למשתמש ולאחר מכן שליחת הקובץ מהמחשב האישי לבקר באופן טורי תו אחר תו (ללא קידוד) הקובץ נשמר בזיכרון ה- FLASH של הבקר כקובץ txt . לאחר קבלת הקובץ בצד הבקר, תישלח הודעת Acknowledge למסך ה- PC ויתחיל ביצוע ה- script בצד הבקר.

צורת שמירת קובץ txt.* בצד הבקר:

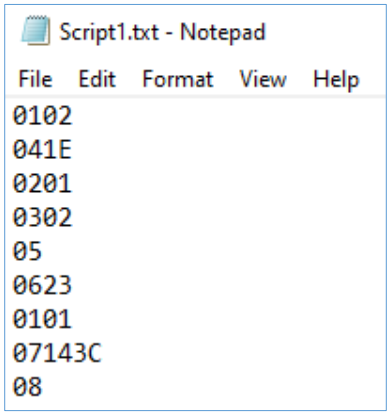
קובץ מוגדר ע"י רצף פיזי של תווים שמיקומו ההתחלתי נתון ע"י מצביע לקובץ ותוכנו מסתיים בתו EOF .

עליכם לנהל שמירה של עד שלושה קבצים בזיכרון FLASH של הבקר ולהגדיר ***struct*** מתאים המכיל את השדות הבסיסיים הבאים (ניתן להוסיף שדות עזר לבחירתכם תוך נימוק הנדסי): כמות קבצים קיימים, מערך מצביעים לשמות הקבצים, מערך מצביעים לתחילת כל קובץ, מערך המכיל את גודלי הקבצים.

רשימת פקודות High Level נדרשות לתמיכה במצב של Script Mode:

OPC (first Byte)	Instruction	Operand (next Bytes)	Explanation
0x01	blink_rgb	x	Blink RGB LED in series **x** times with delay d
0x02	rlc_leds	x	Rotate **left** circularly a single lighted LED in 8-LED array **x** times with delay d
0x03	rrc_leds	x	Rotate **right** circularly a single lighted LED in 8-LED array **x** times with delay d
0x04	set_delay	d	Set the delay **d** value (***units of 10ms***)
0x05	clear_all_leds		Clear all LEDs (RGB and 16-LED array)
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 pinter reaches them) onto PC screen
0x08	sleep		Set the MCU into sleep mode

Note: The default delay d value is 50 (***units of 10ms***)



Script1.txt explanation

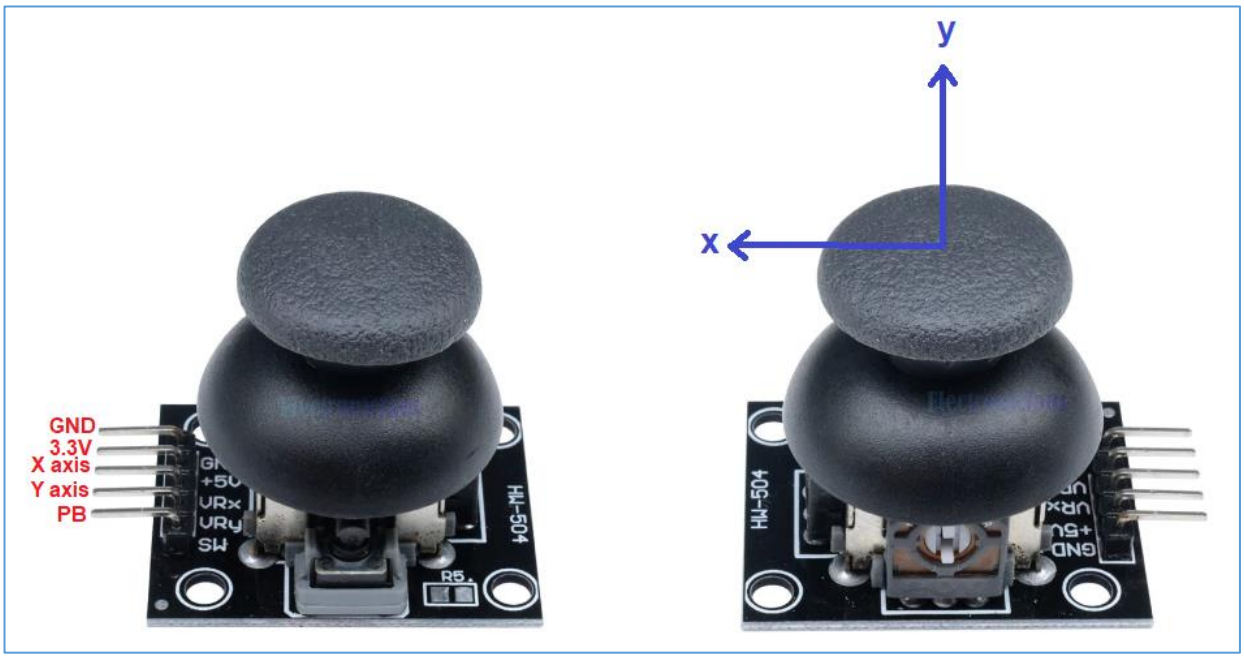
```
blink_rgb  2
set_delay 30
rlc_leds  1
rrc_leds 2
clear_all_leds
stepper_deg  35
blink_rgb  1
stepper_scan 20,60
sleep
```

C. הסברים טכניים – חיישן ומנוע סרבו:

1. מוט היגוי joystick:

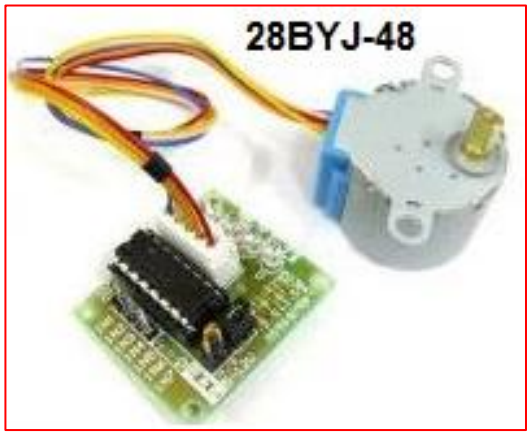
כמפורט לעיל, לצורך שליטה על זווית המיקום של מוט מנוע stepper נשתמש במוט היגוי (joystick) הפועל בצורה הבאה:

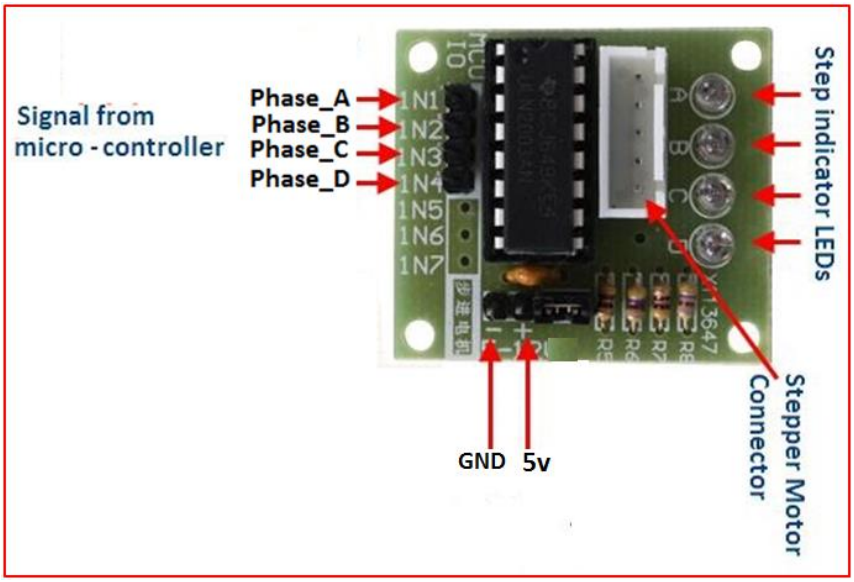
- כאשר המוט במצב המקורי (לא מוטה) המתח במוצא הרגליים Vrx , Vry הוא $\frac{V_{cc}}{2} = \frac{3.3v}{2} = 1.65v$
- בהטיית המוט לכיוון y,x המתחים Vrx , Vry בהתאמה יעלו בצורה יחסית בטווח [1.65v,3.3v] .
- בהטיית המוט לכיוון -y,-x המתחים Vrx , Vry בהתאמה ירדו בצורה יחסית בטווח [0,1.65v] .



2. מנוע Stepper Motor:

מנוע צעד, זהו מנוע שניתן להפעילו כך שמוט הסיבוב שלו, יסתובב בכל איטרציה בזווית φ הנקראת צעד.
מנוע צעד מהווה עומס, המחובר לכרטיס ממשק המתווך בין הבקר לעומס. מצד אחד נחבר לכרטיס הממשק את ה- MCU המספק מידע בהספק נמוך, מצד שני נחבר את המנוע המהווה עומס וצורך הספק גבוה.
יש צורך לחבר לכרטיס מתח הפעלה של 5v ברגל המיועדת לכך (ראה תצלום הבא).





סיבוב המנוע בצעד / חצי צעד :

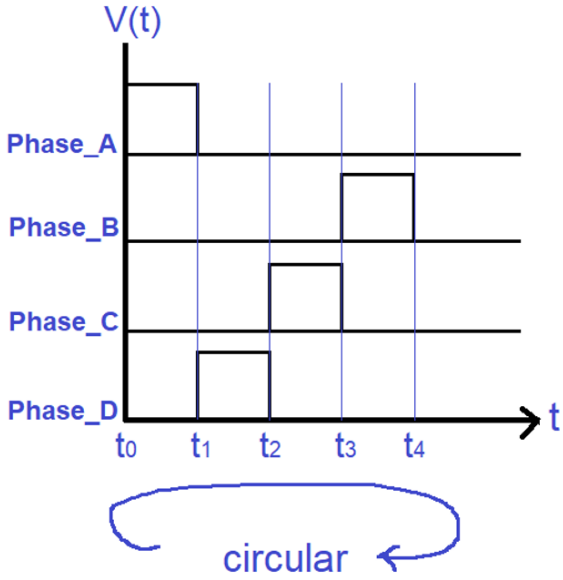
- <u>סיבוב המנוע בצעד מלא בכיוון clockwise:</u>

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 <u>לפי הטבלה הבאה:</u>

כדי לבצע הזזה רציפה כרצוננו, נצטרך לבצע המתואר בטבלה בצורה מחזורית.

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



- <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

- <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

הערות:

i. בהגדרת הפרויקט נתונה לכם זווית **נומינאלית** $\varphi = 0.088°$ של גודל צעד המנוע, אולם גודל הזווית באופן מעשי נדרש למדידה. באופן כללי, ישנה שונות במבנה הפיזי בין מנועים זהים מאותו פס ייצור כך שעבור כל מנוע שונה נדרש שלב כיול לצורך מדידת הערך המעשי של זווית הצעד φ ועליכם למצוא בעזרת ניסוי מקדים.

ii. כיול לצורך מדידת הערך המעשי של זווית הצעד φ של המנוע:

המשתמש נדרש לתת פקודה מתפריט המחשב לתחילת סיבוב המנוע ברצף (בתדר שהמשתמש יוכל להבחין בהשלמת סיבוב שלם במדויק בהתאם לפס הכיול השחור) ופקודה לסיום הסיבוב בהשלמתו. בקוד נדרש לנהל משתנה counter למניית הצעדים במשך הסיבוב השלם, בפקודה לתחילת הסיבוב counter מתחיל את המנייה מאפס ובפקודה לסיום הסיבוב נעצרת המנייה וניתן לבצע חישוב

$$\varphi = counter/360°$$

3. ממשק משתמש בצד ה- PC :

שליחת מידע בין הבקר ל- PC מבוססת תקשורת טורית וליצירת תפריט ממשק למשתמש על גבי מסך ה- PC הכולל יכולת הצגה וויזואלית דינאמית של מוניטור ה- Radar (כפי הדוגמה הנתונה בסעיף A7) על גבי מסך ה- PC. עליכם לכתוב את המעטפת והממשק (GUI) בצד ה- PC בכל שפה שתבחרו Python JAVA , C++ , Matlab, או שימוש במעטפת C# - מומלץ. במעטפת זו תצטרכו לתמוך בתקשורת טורית אסינכרונית של המחשב עם הבקר מבוססת סטנדרט RS-232 לצורך העברת תווים וקבצים (העברת הקובץ תו אחר תו) בין המחשב לבקר וההיפך. מצב ברירת המחדל הוא:

9600 BPS , 8-bits , 1 Start , 1 Stop , (none) No parity

D. דו"ח מכין: (משקל 10%)

1. כתיבת דו"ח מכין פרויקט מסכם, לפי הוראות לכתיבה ועריכת דו"ח מכין הנמצא במודל.
2. צורת הגשה:

- הגשת דוח מכין תיעשה ע"י העלאה למודל של תיקיית zip מהצורה **id1_id2.zip** (**כאשר id1 < id2**), **רק הסטודנט עם הת"ז id1 מעלה את הקבצים למודל.**
- התיקייה תכיל את **שלושת** הפרטים הבאים בלבד:
  - ✓ **קובץ pre_final$_x$.pdf** – מכיל תשובות לחלק תיאורטי דו"ח מכין
  - ✓ תיקייה בשם **CCS** - מכילה שתי תיקיות, אחת של קובצי source (קבצים עם סיומת ***.c**) והשנייה של קובצי header (קבצים עם סיומת ***.h**).
  - ✓ תיקייה בשם **PC_side** - המכילה קובצי מקור של אפליקציית צד מחשב + קובץ ReadMe המתאר בקצרה מה תפקיד כל קובץ מקור במימוש האפליקציה.

E. מבנה הציון בפרויקט:

1. משקל הפרויקט הוא 45% מהציון הסופי - חלק ביצוע (תמיכה בארבעת סעיפי התפריט בסעיף B) משקלו 90% וחלק דו"ח מכין 10%
2. הציון יינתן על-פי הערכה המבוססת על קריטריונים של עמידה ודיוק בדרישות הפרויקט, בקיאות בקוד+אלגוריתם+תיאוריה, דו"ח מכין והגנה על הפרויקט. המשמעות, כל סטודנט בנפרד יידרש לגלות הבנה מעמיקה במרכיבי הפרויקט (תיאוריה, חומרה, תוכנה ואלגוריתם).
3. כל קבוצה תגיש דו"ח מכין לפי קובץ "הוראות לכתיבת דו"ח מכין ודרישות תוכנה" המופיע במודל.

בהצלחה!