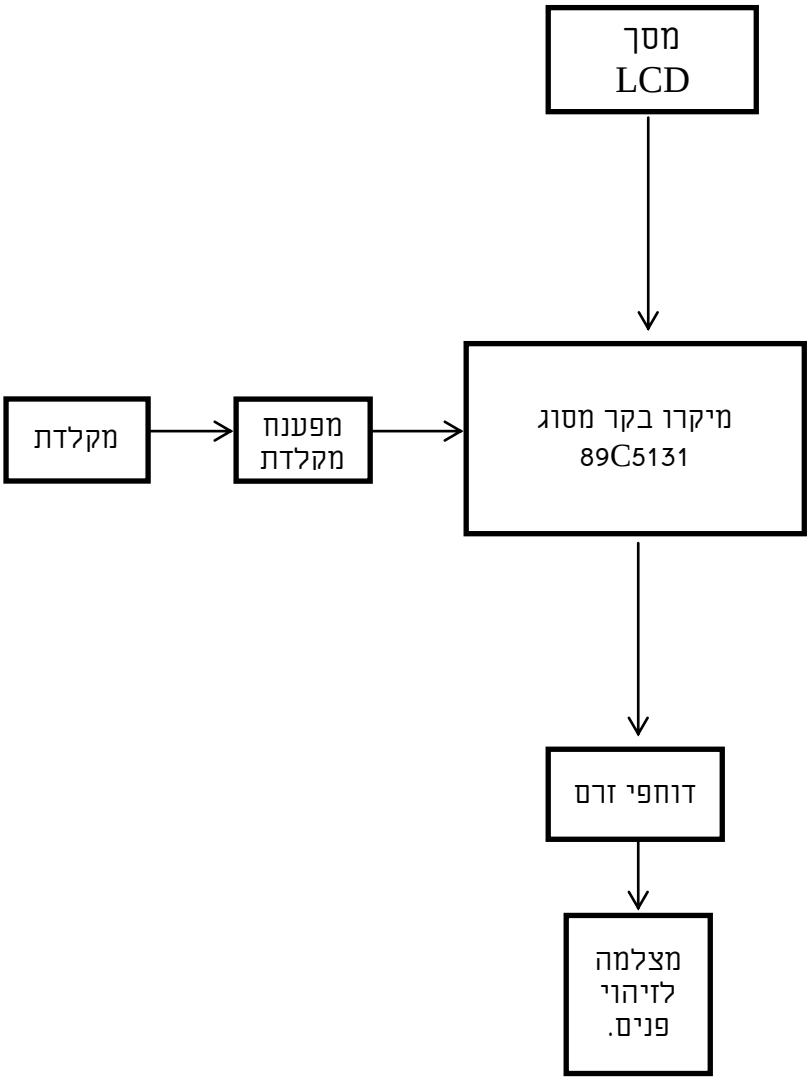
דיאגרמת מלבנים :

הבסיס הוא רכב והרכיבים מורכבים עליו.



תקציר

בפרויקט זה אנו מפעילים רכב ממנוע באמצעות תנועות הראש של בן אדם היושב בכיסא גלגלים .כאשר האדם מזיז את הראש ימינה הכסא זז ימינה גם הוא , כאשר מזיז את הראש ימינה הכסא זז שמאלה וכאשר הראש ישר גם הכיסא זז בכיוון ישר. הפרויקט מיועד לאנשים שתנועות הידיים קשות עבורם בשביל להזיז את כיסא הגלגלים ולכן השימוש בתנועות הראש אמור להקל עליהם .

זיהוי הפנים מתבצע באמצעות מצלמת קינקט שמחוברת למחשב וsdk שנותן אפשרות לקוד בסיס להדמיית הפנים . חיבור בין המחשב למכונית מתבצע באמצעות כבל USB\RS232 ומאריך שאנחנו יצרנו לכבל זה . המכונית זזה באמצעות מנועים כאשר לכל מנוע יש את חופש פעולה משלו (4X4) .הבקר מונח על המוכנית וככה מקבל מידע מהחשב של זיהוי הפנים שהתקבל . הבקר מקבל את המידע ומזיז את הגלגלים בהתאם למידע שהתקבל .

קשיים שהתעוררו במהלך הפרויקט :

1. התקשנו במציאת קבל ארוך יותר שממיר מפרוטוקול USB \ RS232.
   פתרון – יצרנו קבל חדש שיהווה מאריך לקבל שיש לנו .
2. בעיית הלחמות וחיבוטים , התקשנו בפעולות אלה משום דבר זה היה חדש לגמרי עברונו .
   פתרון – התנסנו עוד ועוד ,כאשר שצברנו מספיק ניסיון לא היה עוד קושי
3. זיהוי הפניים הוא דבר מורכב לכן חיפשנו פלטפורמה שתתן לנו את המענה הכי יעיל ומהיר . (במסגרת הפרויקט אין לנו את הידע בשביל לכתוב קוד לזיהוי פנים בעצמנו)
   פתרון – לאחר חיפושים רבים מצאנו את האופציה בשימוש מצלמת הקינקט וSDK
4. לאחר שהיה לנו את הקוד לזיהוי פנים , עדין הדבר היה בסיס ביותר והיינו צריכים להתאים אותו לפרויקט . ההתמודדות עם הקוד וגישה לדברים הייתה קשה
5. בתחילת הפרויקט רצינו לקחת כיסא גלגלים בכדי להדגים את הפרויקט עליו . מצאנו קושי בלהביא כיסא גלגלים בעקבות מחסור שיש בארצנו .
   פתרון – השתמשנו **<u>ברכב אשר ידמה את פעולת הכסא הגלגלים</u>** .
6. את תנועה של הגלגלים לא יצרנו עם גשר H מסיבות של נוחות . מצאנו כי יהיה יותר נוח ופשוט להזיז את הגלגלים בצורת ציר . כלומר ,
   - התנועה ימינה תתבצע הפעלת מנועים 1 ו 2 .
   - התנועה ישר תתבצע הפעלת מנועים 1 2 3 4 .
   - התנועה שמאלה תתבצע הפעלת מנועים 3 ו 4 .

הדמיית המכונית מלמעלה

תוכן עניינים

רשימת טבלאות

רשימת איורים

רשימת נספחים

נושא	מספר נספח
hjr 12cl……..	1
Max232……	2
Kinect 2……..	3

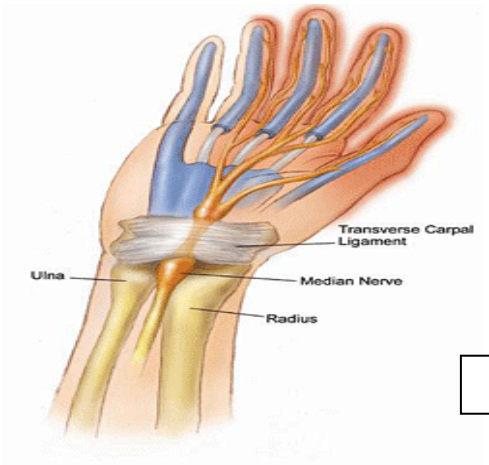
מפרט טכני

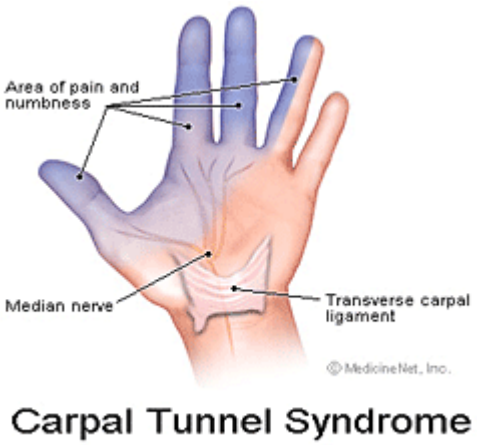
1. מיקרו בקר
2. תצוגת LCD
3. רכב
4. מצלמה לזיהוי פנים - KINECT
5. תוכנה לזיהוי פנים +מחשב
6. RS232\USB
7. MAX232
8. ממסר hjr 12cl
9. דוחף זרם ULN2803
10. מקלדת
11. מפענח למקלדת
12. כבל CAT 5e + מחברי D-TYPE

מבוא

פרויקט זה נכתב כחלק התיאורטי של פרויקט גמר הנדסאי .
אנו סבורים שפרויקט זה יוכל לעזור בפיתוח פלטפורמות שיקלו על הסובלים ממגבלות פיזיות באוכלוסייה.
השליטה בכיסא גלגלים על ידיי תנועות הראש תוכל לעזור לרבים מהאנשים אשר סובלים מבעיות ביד\ כתף לשלוט על כיסא הגלגלים ללא הגה אלא על ידיי הזזת הראש בלבד .
כדוגמת אנשים שסובלים **מתסמונת התעלה הקרפילית .**

תסמונת התעלה הקרפלית מהווה אחת מהבעיות הנפוצות של העצבים ההיקפיים. תסמונת התעלה הקרפלית נגרמת עקב לחץ על עצב (העצב המדיאני) באיזור שורש כף היד. העצב המדיאני נכנס לכף היד דרך תעלה צרה וקשיחה – התעלה הקרפלית- וכל עליית לחץ בתוך התעלה גורמת לפגיעה בעצב. הסימפטומים השכיחים של תסמונת התעלה הקרפלית הם כאבים, נימול, תחושת רדימות, זרמים בכף היד. לעיתים קרובות הסימפטומים מופיעים בלילה ומעירים משינה. כאשר תסמונת התעלה הקרפלית מתקדמת ונגרמת פגיעה בסיבי העצב המוטורים תופיע חולשה ומגושמות של היד. הטיפול היעיל לבעיה הוא ניתוח לשחרור הלחץ המוגבר בתעלה. במקרים קלים של תסמונת התעלה הקרפלית ניתן לנסות טיפול שמרני. קיימים מצבים אחרים העלולים לגרום לסימפטומים דומים לאלה של תסמונת התעלה הקרפלית, כגון: בעיות בעמוד השדרה הצוואר, נוירופתיה וכו', ולכן אבחון מדוייק של הבעיה חשוב להצלחת הטיפול. כמו כן, חשוב לטפל בבעיה בטרם נגרם נזק בלתי הפיך לעצב.

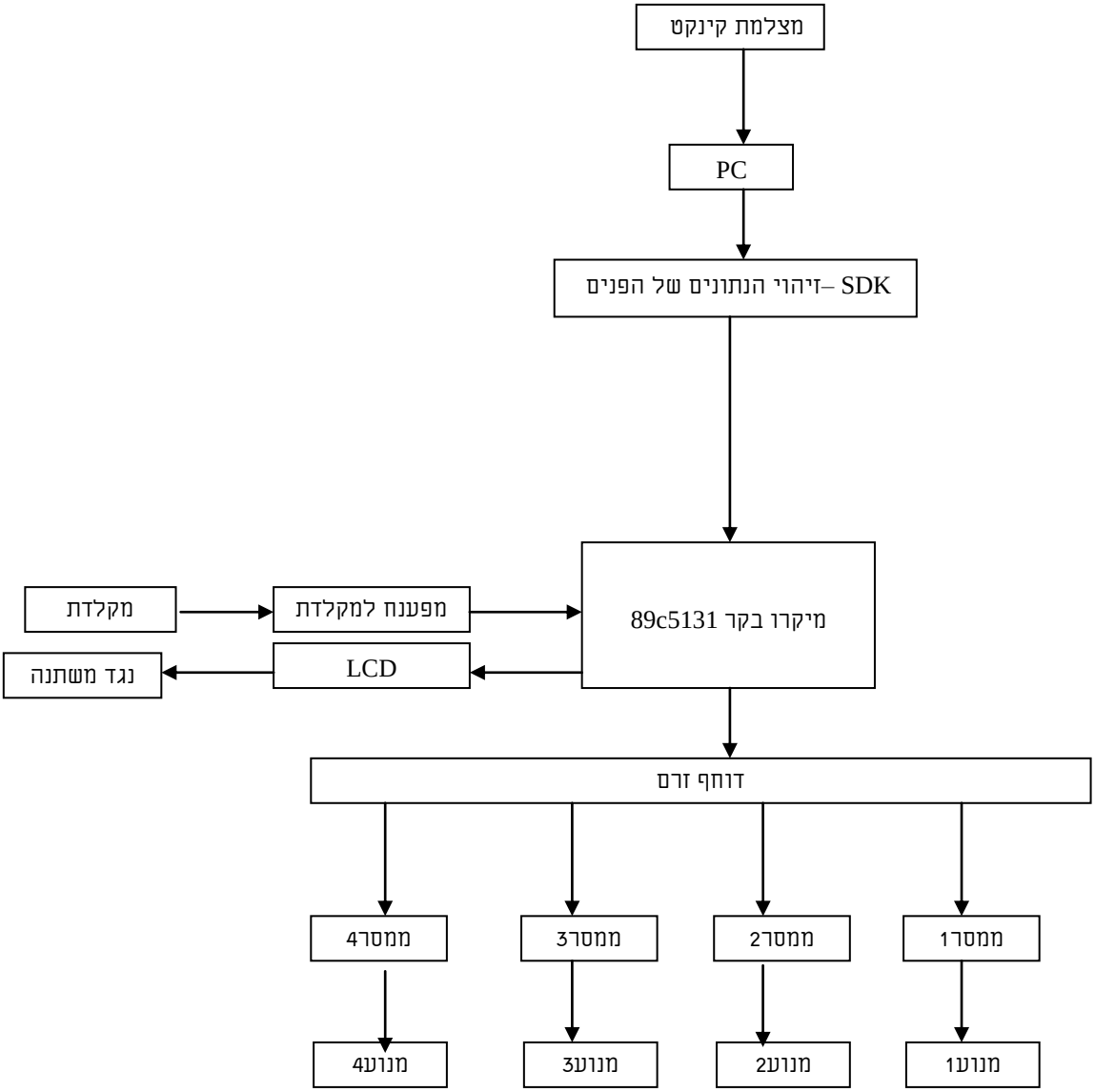




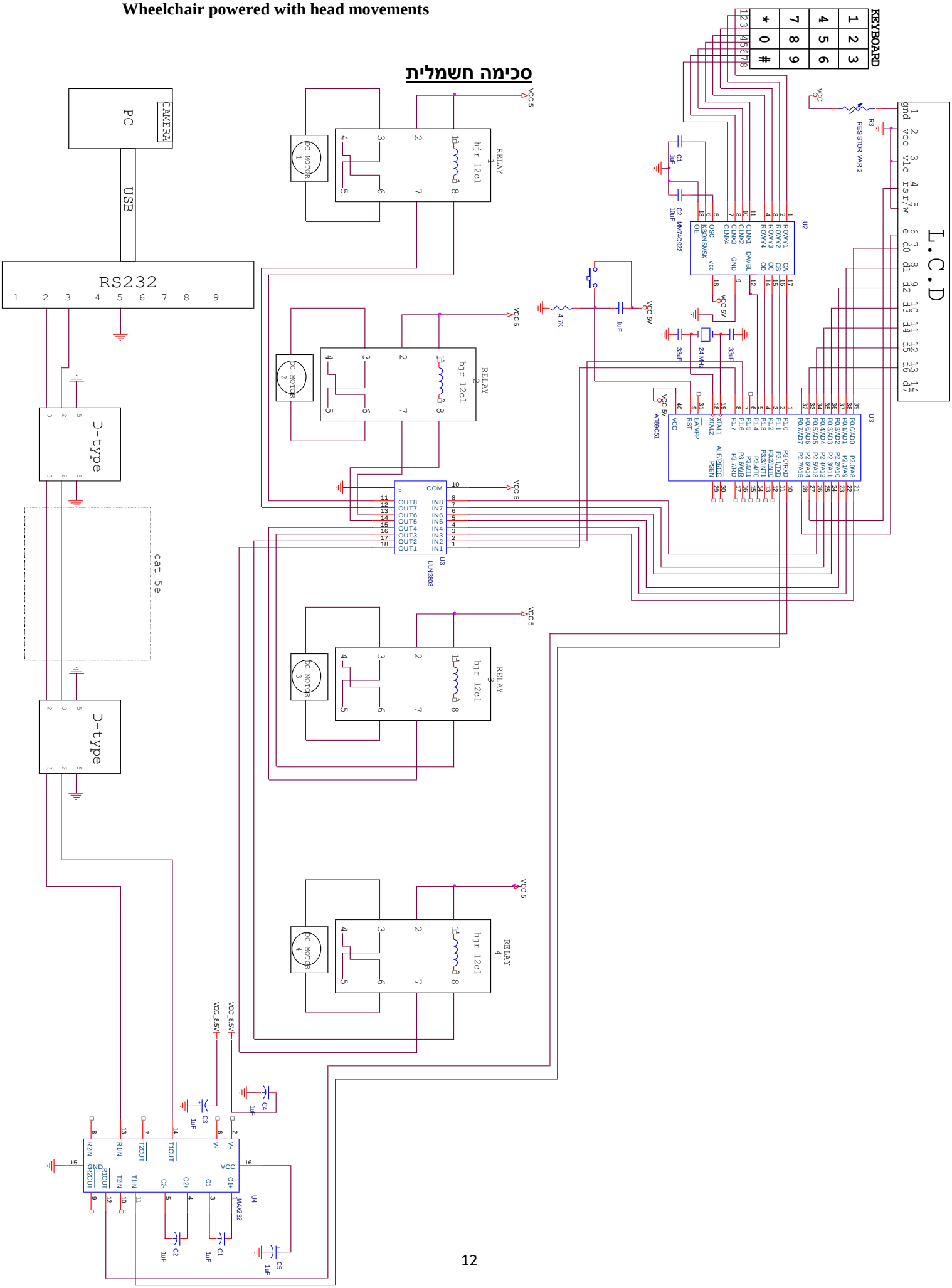
איור 1

במקרים קלים של תסמונת התעלה הקרפלית ניתן לנסות טיפול שמרני, שמטרתו להקל על הסימפטומים, כגון: תרופות לשיכוך כאבים; סד תמיכה לשורש כף היד המונע כיפוף ממושך של היד ולחץ על העצב; הזרקות סטרואידים מקומיות- ולעקוב קלינית ועל ידי בדיקות הולכה עצבית (בדיקת EMG) חוזרות על מנת לראות אם הנזק לעצב מתקדם..
הטיפול הסופי והיעיל ביותר בחולים עם תסמונת התעלה הקרפלית הוא ניתוח לשחרור הלחץ בתעלה הקרפלית.

סכימה מלבנית מפורטת

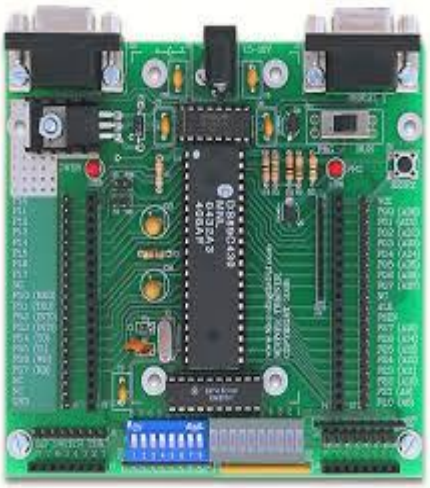


סכימה חשמלית

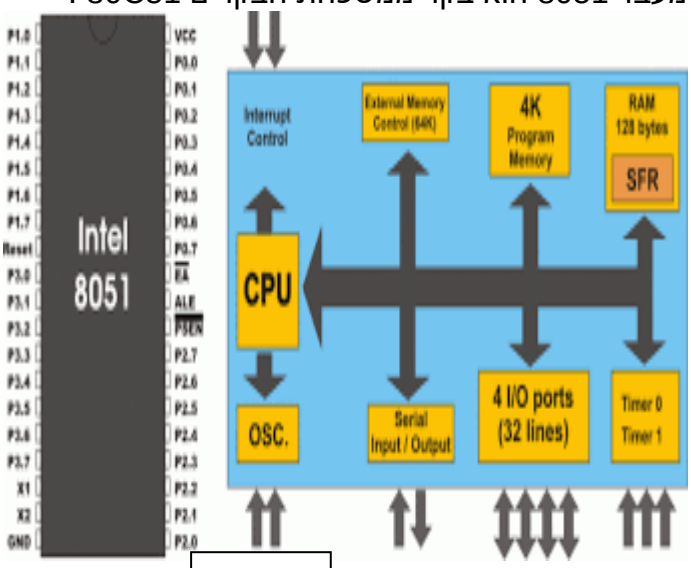


מעבד 8051

מעבד 8051 הוא בקר ממשפחת הבקרים 80C51 .



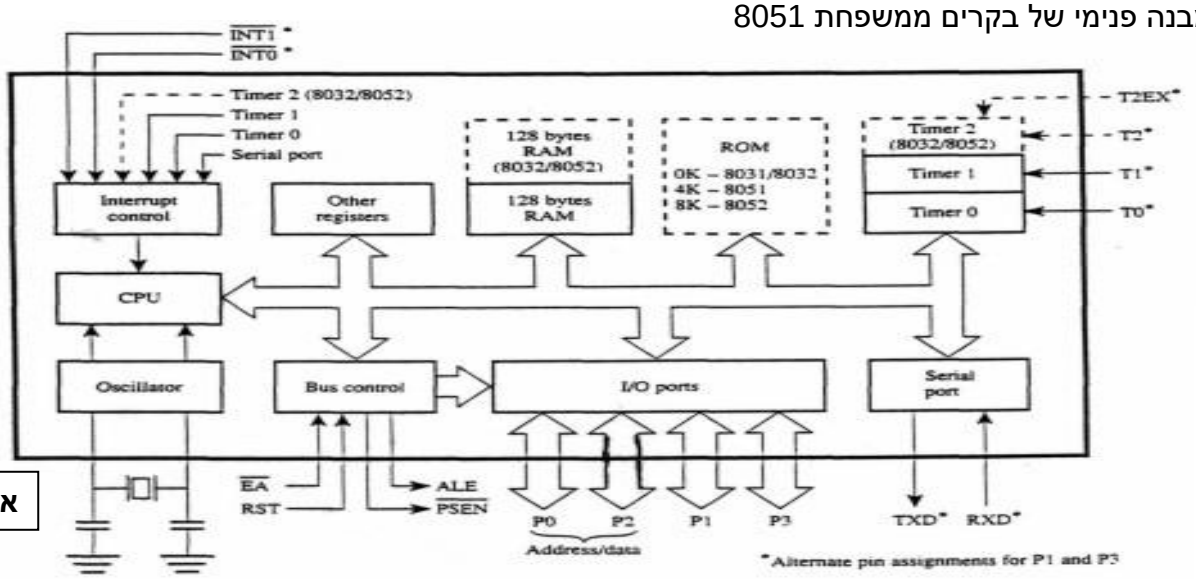
איור 3



איור 2

ה – 8051 הוא בקר ממשפחת המיקרו הבקרים 80C51 , שמקורה בעיצוב המקורי של אינטל שיוצא בתחילת שנות השמונים (צאצא של 8048 הקדום)

מבנה פנימי של בקרים ממשפחת 8051



איור 4

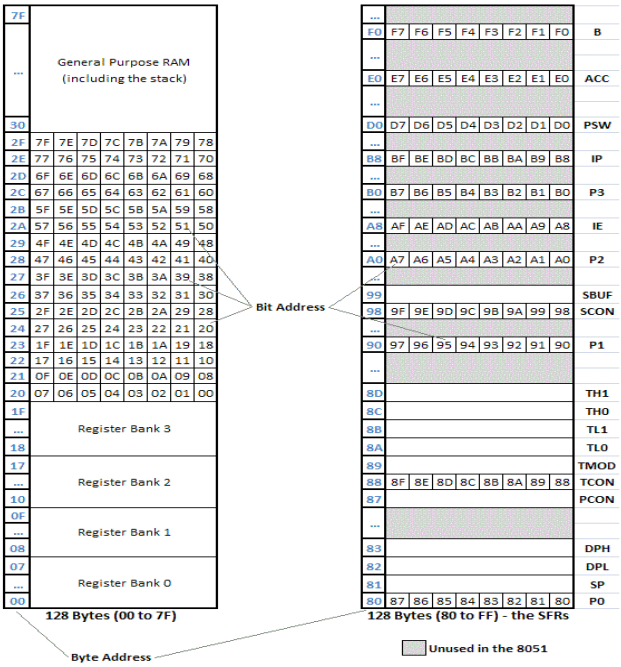
היסטוריה של השבב המקורי

השבב המקורי היה בקר 8 סיביות (כיום הוא בקר 16 סיביות) וכלל 256 בתים משל זיכרון RAM (מובנה על השבב) , שהיו נגישים באופן ישיר או באופן עקיף על ידי שימוש באוגרים במידה וזיכרון הקריאה\ כתיבה לא מספיק , ניתן להוסיף זיכרון חיצוני. זיכרון זה מיועד לנתונים . כדי לגשת לתא בזיכרון באזור זה נעזרים באוגר נוסף בן 16 סיביות ה DPTR , בו רושמים את כתובת התא אליו רוצים לפנות . בימינו , בקרים מתקדמים מבוססי ה 8051 מכילים שני אוגרי DPTR ויש אפילו שמכילים 8.

ה 8051 כלל 4KB של זיכרון ROM , היו לו 32 ערוצי קלט\פלט , 2-3 מוני זמן timers בני 16 סיביות , 5-6 פסיקות (שתיים חיצוניות ושתיים בעלות אופצייה לקביעת סדר עדיפויות) פעולה בסיסית מתבצעת ב 12 clock cycle .

ה- 8051 הוא הבקר הראשון מסוגו שהציג את שיטת מיון סיביות , המאפשרת גישה לכלל

המידע החל מהסיבית ה 32 בזיכרון החיצוני באופן פרטני של הרמת סיבית\הורדת סיבית בעזרת שימוש בסט הוראות חדשניות.
מודל הזיכרון של המעבד נחשב מורכב לאותה תקופה , אך בשל הגמישות הרבה שהוא מאפשר בעיצוב מערכות , הפך ה8051 לפופולארי ביותר (נמכרו מעל 2 מילון יחידות מאז 1988).
מאז העיצוב המקורי שפותח על ידיי אינטל , חברות נוספות יצרו שלל דגמים נוספים של המעבד , בניהן: ATMEL, AMD, CYGNALואחרות .



תיאור כללי

יחידת ה CPU

איור 5

יחידת העיבוד המרכזית ל 8 ביט המבצעת את כל הפקודות החישוביות + ולוגיות , העברת נתונים ובקרה על כל שאר יחידות הרכיב .
היחידה מקבלת אותות תיזמון ושעון ממתנד השעון
ליחידה 5 מבואות פסיקה :
2 פיסיקות חיצוניות
2 פסיקות פנימיות מהמונים בהסתיים כל מחזור מנייה
פסיקה מבקר התקשורת הטורית בהסתיים שידור מילה

יחידת ה PROGRAM MEMORY

זהו זיכרון התוכנית (אינו מופיע בכל הרכיבים) . זיכרון זה הוא בגודל 4KBYTE ניתן להרכיב את הזיכרון עד 64KBYTE . הזיכרון יכיל את תוכנת מערכת ההפעלה של המערכת עלייה יבקר ה 8051.

יחידת ה DATA MEMORY

זהו זיכרון הנתונים ברכיב . גודל הזיכרון הוא עד 128 BYTE .
הזיכרון הפנימי ממופה בתחום 00H-07H , בתחום 80H-FFH קיים תחום של אוגרים מיוחדים .
הזיכרון ניתן לקריאה וכתיבה וניתן להוסיף לו זיכרון RAM.

יחידת ה TIMERS

זוהי יחידת המונים של המיקרו בקר אשר 16 BIT כל אחד וניתנים לתכנות אופן פעולה.

יחידת המפתחים P0,P1,P2,P3,P4 Ports (P=PORT)

אלו 4 פורטים אשר יכולים לשמש כקלט ופלט לבקר או כקווי בקרה להרחבת הזיכרון .
תפקידם העיקרי של המפתחים הוא לאפשר קלט של נתונים מהסביבה אל המקרו בקר ,למשל :קלט של מצב מפסקים,קלט של נתוני חיישנים וכדומה.
או לאפשר פלט אל העולם החיצוני שמחוץ לבקר כמו: הדלקת נורית,הפעלת מנועים וכו.
למקרו בקר 8051 יש 5 מפתחים . 4 מפתחים בגודל 8 סיביות ומפתח אחד בגודל 2 סיביות. כל המפתחים יכולים לשמש הן כמבוא והן כמוצא .
ב8051 יש5 הדקים של מפתחים שמוגדרים כ P0,P1,P2,P3,P4 ארבעת המפתחים P0,P1,P2,P3 הם ארבעת המפחים הקלאסיים של 8051 והם בני 8 סיביות כל אחד.
המפתח 4P הוא ייחודי למיקרו בקר 8051 והוא בן 2 סיביות בלבד.
כל סיבית של מפתח מסומנת באינדקס .לדוגמא P1.0 היא סיבית 0 של מפתח P1.
לדוגמא P1.1 היא סיבית 1 של מפתח P1. לדוגמא P1.2 היא סיבית 2 של מפתח P1.
וכך הלאה .לדוגמא מפתח P1:

יחידת הUART

זוהי יחידת קלט פלט לתקשורת טורית המתאימה ל8051. התקשורת מתאימה ל/rs232.

איור 6

יתרונות הבקר

זהו אחד הבקרים הנפוצים ביותר בעולם ובשל כך נכתבו עליו הרבה מאוד ספרים לחובבי אלקטרוניקה ומקצוענים כאחד.
בנוסף,ניתן למצוא שלל מקורות מידע הכוללים הסברים על השימוש בו:דוגמאות קוד ותוכנות שיתופיות עבור בקר זה (בינהם מהדרי שפת אסמבלי ,מהדרי שפת C ,תוכנות ,סימולציות וכו') עצם העובדה שכ 12 יצרניי משנה שונים מספקים יותר ממאה גרסאות שונות לבקר מצדיק את העובדה שהוא פופולארי מאוד..במידה והבקר מיועד לפרויקט קטן שגדל לאחר מכן אין צורך לכתוב את התוכנה מחדש או לשנות אותה אלא רק להחליף לבקר מתקדם יותר מאותה המשפחה.(אין זה כך בכל משפחות הבקרים האחרות)

חסרונות הבקר

אין יציאות PWM מובנות אך ניתן למצוא בקרים משוכללים מבוססי 8051 ,המספקים יכולת זו.
אין המרת קלט A\D מובנת ,אך ניתן למצוא בקרים משוכללים 8051 המספקים יכולת זו.
אין תמיכה בפרוטוקול התקשורת INTER IC BUS .

תיאור חיצוני של 8051

הדקיי הבקר

VCC (פין 40)	מתחבר להדק החיובי של ספק הכוח בן V5.
VSS (פין 20)	קו האדמה של הרכיב
XTAL1,XTAL2 (פינים 18 ו- 19)	רגלי חיבור לגביש. לקווים אלו מחובר גביש הקובע את תדר הפעולה של המעבד, משני צידי הגביש מחוברים 2 קבלים לאדמה לקיזוז רעשים.
RST (פין 9)	אתחול המעבד (RESET). עם עלייתה ל-HIGH ה-CPU מפסיק את פעולתו. עם ירידתה ל-'0' לוגי, ה-CPU משנה את מצביע ההוראות לכתובת 0000, ובכך מתחיל את התכנית של המעבד מהתחלה.
EA' – EXTERNAL ADDRESS (פין 31)	כניסה זו מסמנת ל- CPUהאם אזור התכנית נמצא בתחום הכתובות הנמוך (0000 – FFF0) שייך ל-ROM חיצוני או ל-ROM פנימי. רגל זו נמצאת בשימוש רק במעבדי 8051 המכילים בתוכם ROM פנימי. במעבד מסוג 8031 יש לקצר קו זה ל-GND .
ALE (פין 30)	קו יציאה הנועל את הכתובת הנמוכה ב- LATCH (ADDRESS LATCH ENABLE). עולה ל-HIGH למשך STATE אחד (2 מחזורי שעון) בכל פעם שמתבצעת פנייה לזיכרון.
PSEN' – PROGRAM SET ENABLE (פין 29)	קו יציאה, המציין לזיכרון ה-PROGRAM שה-CPU מבקש לקרוא נתון מאזור זה, כאשר קו זה יורד ל-LOW . P1.0-P1.7 – (פינים 1-8) פורט זה משמש כפורט מבוא או פורט מוצא בלבד – ניתן לגשת לסיביות הפורט במיעון ישיר.
P0.0-P0.7 / AD0-AD7 (פינים 32-39)	- קווי פורט 0, היכולים לשמש כפורט מבוא או כפורט מוצא. במקרה של חיבור רכיבי תמיכה חיצוניים לרכיב, מתפקדים קוים אלה כקווי AD0 – AD7, ומעבירים את החלק הנמוך של הכתובת בשלב המחזור הראשון של מחזור המכונה ואת הנתונים בשלב השני של מחזור המכונה.
P2.0-P2.7 / A8-A15 (פינים 21-28)	קווי פורט 2, היכולים לשמש כפורט מבוא או כפורט מוצא. במקרה של חיבור רכיבי תמיכה חיצוניים לרכיב, מתפקדים קוים אלה כקווי הכתובת הגבוהים A8-A15 .
P3.0-P3.7 (פינים 10-17)	קווי פורט 3, היכולים לשמש כפורט מבוא או כפורט מוצא בנוסף לכך יכולים קוים אלה לתפקד כקווי בקרה שונים.
P3.0 / RXD	קו כניסה של אות המגיע בתקשורת טורית אסינכרונית ל- UART.
P3.1 / TXD	יציאה של אות היוצא בתקשורת טורית אסינכרונית מה- UART.
P3.2 / INT0	קו פסיקה חיצונית היכול לעבוד בדרבון רמה או בדרבון קצה.
P3.3 / INT1	קו פסיקה חיצונית היכול לעבוד בדרבון רמה או בדרבון קצה. P3.4 / T0 כניסת אותות ל-TIMER0 במקרה שהוא מתפקד כמונה.
P3.5 / T1	כניסת אותות ל-TIMER1 במקרה שהוא מתפקד כמונה.
P3.6 / WR	קו יציאה המציין ליחידות החיצוניות, שה-CPU מבקש לכתוב נתון.
P3.7 / RD	קו יציאה המציין ליחידות החיצוניות, שה-CPU מבקש לקרוא נתון.

טבלה 1

ארגון הזיכרון ב 8051

RAM

כשההוראה MOV A,00 מתבצעת , הבית במקום ה 00 בזכרון ה – RAM,נקרא ונכתב אל ה- accumulator. 128 הבתים הראשונים הם למטרות – כלליות (general purpose)אך 128 הבתים העליונים הם האוגרים המיוחדים – special Function Registers(SFR) שאחראים על המחסנית ,התקשורת הטורית ,מוני הזמן,הפסיקות ועוד (ב-8051 הבסיסי מנוצלים בפועל רק 26 כתובות מתחום זה ,כל שאר המקום יועד ליישומים עתידיים נוספים)לכן,יש לתת תשומת לב מיוחדת לשינויים בכתובות אלו ,שכן הם אחראים על מצה הפעולה הנוכחי של הבקר.

הזכרון הפנימי מחולק למספר אזורים כמפורט בטבלה :

אזור SFR	FFh – 80h
אזור זיכרון נתונים	7Fh – 30h
BIT ADRESSABLE AREA	2Fh – 20h
3BANK	1Fh – 18h
2BANK	17h – 10h
1BANK	0Fh – 08h
0BANK	07h – 00h

טבלה 2

1.אזור הבנקים - ממוקם בתחום 00h - 1Fh . יש 4 קבוצות של בנקים .
בנק מורכב מ 8 רגיסטרים R0-R7 , הבנקים ממוקמים בכתובות של ה RAM 00h - 1Fh וניתן להשתמש בכל אוגר ככתובת ישירה למידע זמני . כל בנק יכול לשמש ככתובת ישירה וכתובת עקיפה . הבחירה באיזה בנק נשתמש נקבע על ידיי האוגר PSW (PROGRAM STATUS WORD) בסיביות RS0 וRS1

בנק	RS0	RS1
BANK 0	0	0
BANK 1	1	0
BANK 2	0	1
BANK 3	1	1

טבלה 3

2. אזור זיכרון למיון נתונים הניתן למיון של ביטים בודדים אזור זה ממוקם בתחום – 2FH 1FH
3. אזור זיכרון נתונים : ממוקם בתחום 2Fh-7Fh.
4. -SFR (SPECIAL FUNCTION REGISTER) – אזור זה ממוקם בתחום FFh-80h

איור 7

אוגרים מיוחדים :

שאר המיקומים בטבלה הושארו ריקים בכוונה בתוך מטרה לאפשר ליצרנים השונים פיתוח מיקרו בקרים עתידים תוך כדי השארתם תואמים לגרסה הקודמת . מצב זה מאפשר גם לתוכניות שנכתבו גם לפני זמן רב עבור מיקרו בקר אשר אינו מיוצר יותר לפעול על דגם מיוצר עכשווי.

SYMBOL	NAME	ADDRESS
ACC*	Accumulator	E0h
B*	Register B	F0h
PSW*	Program Status Word	D0h
SP*	Stack Pointer	81h
DPTR	Date pointer 2 byte	-----
DPH	High byte of DPTR	82h
DPL	Low byte of DPTR	83h
P0*	Port 0	80h
P1*	Port 1	90h
P2*	Port 2	A0h
P3*	Port 3	B0h
IP	Interrupt Priority Control	B8h
IE*	Interrupt Enable Control	A8h
TMOD	Timer/Counter Mode Cntrol	89h
TCON*	Timer/Counter Mode Control	88h
TH0	Timer/Counter 0(High byte)	8Ch
TL 0	Timer/Counter 0(Low byte)	8Ah
TH 1	Timer/Counter 1(High byte)	8Dh
TL 1	Timer/Counter 1(Low byte)	8Bh
SCON*	Serial Control	98h
SBUF	Serial Buffer	99h
PCON	Power Control	87h

* ניתן למיעון סיביות

טבלה 4

מונה התוכנית – PROGRAM COUNTER

רגיסטר זה בעל 16 סיביות המצביע על כתובת ההוראה הבאה לביצוע . רגיסטר זה מתקדם באופן אוטומטי תוך ביצוע ההוראה

DATA POINTER -DPTR

רגיסטר בעל 16 סיביות שעובד כרגיסטר אינדקס ותפקידו לסמן כתובת בזיכרון הנתונים .

ARITHHMETIC LOGIC UNIT

אחראית על פונקציות אריתמטיות ולוגיות כגון : כפל , חיבור ,חיסור , חילוק , NOT, OR , AND, INC,DEC.

זיכרון RAMומחסנית STACK

מרחב זה תופס את הכתובת בין 30H –7FH ומכיל 80 כתובות . המרחב משמש כזיכרון קריאה וכתיבה ומשמש גם כמחסנית לאחסון נתוני הרגיסטרים

סמן המחסנית SP- STACK POINTER

רגיסטר המסמן כתובת במחסנית .כל נתון מוכנס בהוראת PUSH וכתובת גבוהה באחד מכתובות ה SP

אוגרים אריתמטיים A\ B \PSW

A - משמש כצובר . משתמשים באוגר זה לחלק גדול מפעולות החישוב . דרך אוגר זה מעבירים את המידע ל CPU .

B - משמש לפעולת כפל וחילוק . יכול להחליף את אוגר A

PSW- רגיסטר סטאטוס משמש כאוגר לציון מצב לאחר ביצוע פעולות אריתמטיות לוגיות .

IP

זהו אוגר המאפשר את סדר קביעות הפסיקות לרכיב

IE

זהו אוגר המאפשר מיסוך פסיקות ברכיב ואפשורן.

TL1 TH1

אלה שני אוגרים שצירופם מהווה את קוצב הזמן TIMER1

TL0 TH0

אלה שני אוגרים שצירופם מהווה את קוצב הזמן TIMER0

SCON

זהו אוגר בקרה לתקשורת טורית . פירוט בהמשך.

SBUF

זהו אוגר כפול המשמש לשידור המילה וקליטתה . פירוט בהמשך.

TCON

זהו אוגר שמבקר את פעולת קוצבי הזמן.

TMOD

זהו אוגר שמשמש לקביעת אופן העבודה לקוצבי הזמן.

PCON

זהו אוגר לבקרת צריכת ההספק של הרכיב.

פסיקות ,מוני זמן ותוכנה לבקר

פסיקות ומוני –זמן

בין האוגרים המיוחדים ,קיימים מונים הקשורים בפסיקות . מצב פסיקה הוא מצב בו מתבקש ה- CPU להפסיק לרוץ על התוכנית בה הוא נמצא ולפנות לקטע קוד אחר. קטע קוד כזה נקר פסיקה.
בקשת פסיקה יכולה להתקבל במספר צורות . מערכות חיצוניות לבקר יכולות להתחבר ל- 2 קווים ,הנכנסים אל הבקר ובעזרתם להודיע ל- CPU על בקשת פסיקה.
בקשה כזו נקראת External interrupt.
בין יתר האוגרים,קיימים מוני זמן.מונה זמן הוא למעשה מונה בינארי הסופר את האותות הנכנסים אליו . אותות אלה יכולים להגיע ממקורות שונים כולל מקורות כולל מקורות חיצוניים לבקר.
כאשר מגיעים אליו אות בתדירות גבוהה ,הופכת המנייה למדידת זמן.
כאשר מסיים המונה לספור עד מספר מסוים שנקבע מראש,הוא יוצר בקשת פסיקה פנימית(internal intrrupt) לבקר. אם מפעילים את המונים בצורה זו,ניתן לייצור פסיקות בקצב מוגדר מראש. אופציה זו יכולה לשמש למשל למערכת בקרת מהירות ברובוט .
ב8051,ניתן לחסום בקשות פסיקה מסויימות או לקבוע סדר עדיפות במקרה של קבלת יותר מבקשה אחת בו זמנית. רביעת החסימה,הנקראת מיסוך MASK וסדר העדיפויות נקבעים ע"י הערכים המושמים באוגרים IP וIE.

תוכנה לבקר 8051

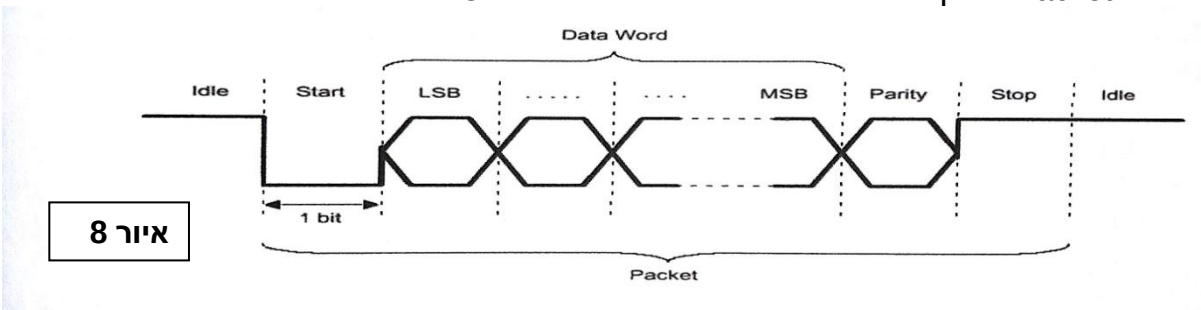
ניתן לתכנת את הרכיב בשתי שפות , האחת שפת אסמבלי שהיא קרובה לשפת מכונה ולכן היא מורכבת יותר, השנייה היא שפת c51 שפה זו היא שפה עילית והתכנות בה קל יותר .
ההפסד בשימוש בשפת C הינו שכשאר עושים הידור למערכת התוכנית תתפוב יותר מקום בזיכרון מאשר אותה תוכנית שתיכתב בשפת אסמבלר .

UART

UARTהינו פרוטוקול תקשורת נפוץ לתקשורת טורית אסנכרונית .מקור השם הינו ראשי תיבות של Transmitter Receiver Asynchronous Universal, כלומר : "מקלט משדר אסינכרוני אוניברסלי", והתייחס אל השבב אשר מימש את פרוטוקול התקשורת הטורית. כיום מערכות רבות מממשות את הפרוטוקול כחלק ממעגל משולב או בתוך רכיב מתוכנת, ולא עושות יותר שימוש בשבב המקורי, אך השם UART נותר לתיאור מערכות אלו.

מסגרת שידור

יחידה המשדרת לפי תקן UART מקבלת את המילה הבינארית באופן מקבילי .לאחר קבלת המילה היא מוסיפה לה מספר סיביות ושולחת את המילה המורכבת בצורה טורית .המרחק בין תחילת השידור (סיבית התחלה) לבין סוף השידור (סיבית סיום)נקרא מסגרת Frame או חבילה Packet. להלן מבנה התשדורת הטורית של ה-UART :



איור 8

הסבר :

א. IDLE -בזמן שאין שידור הקו מוחזק במצב "1 ".מצב זה נקרא, idle כלומר סרק(Mark ב RS232).

ב. Bit Start- תחילת התשדורת מסומנת ע"י, Bit Start סיבית במצב "0 ",המסמנת למקלט שמיד מגיעה מילה Space(בטרמינולוגיה של .RS232) הירידה מ-"1 "ל-"0 "מאפשרת למקלט להסתנכרן על תחילת השידור.

ג. Word Data -מיד לאחר מכן, נשלחות סיביות המילה המקורית, כאשר מתחילים עם ה - LSBומסיימים עם ה- MSB .כל סיבית משודרת למשך הזמן הקצוב שנקבע מראש (ברוחב זהה לסיבית ההתחלה .)מספר הסיביות המקובלים ביותר הם 5 עד 9 סיביות מידע .במיקרו-בקר 8051 מספר סיבות המידע הוא 8 או 9 .

ד. Bit Parity -בסיום המילה נשלחת סיבית הזוגיות, סיבית ביקורת המאפשרת גילוי שגיאה בשידור. יחידת ה UART -במקלט בודקת את הנתון המתקבל ובהתאם לסיבית הזוגיות מחליטה האם הוא תקין או לא .במיקרו-בקר 8051 לא קיימת סיבית זוגיות רשמית .

ה. Bit Stop-סיום התשדורת מסומן ע"י ה, Bit Stop – סיבית במצב "1 ",המסמנת למקלט ששידור המילה הסתיים. משך הזמן של סיבית הסיום עשוי להיות שווה או רחב יותר ממשך הזמן הקצוב מראש לכל סיבית .הערכים המקובלים הם : רוחב של סיבית, סיבית וחצי או של שתי סיביות.

לאחר שהיחידה מסיימת לשדר את הרצף הנ"ל, היא יכולה לשדר רצף חדש או שהיא יכולה להיכנס למצב מנוחה "1 "למשך זמן כלשהו ולחזור ולשדר בהמשך רצף נוסף. כלומר, הזמן בין העברת שני נתונים רצופים אינו חייב להיות קבוע .

במקלט נמצאת יחידת UART שמבצעת פעולה הפוכה : תחילה היא מסירה מהמילה המתקבלת את ספרות ההתחלה והסיום ואז קוד ה- bit parity -נבדק : אם המילה הגיעה משובשת יתבקש המשדר לשלוח את המילה שוב. במידה והמילה תקינה היא תועבר בצורה מקבילית להמשך הפעילות.

<u>קצב העברת הנתונים</u>

במהלך התשדורת מקצים לכל סיבית זמן קבוע ומוסכם מראש. זמן זה נקבע ע"י קצב ההעברה Rate Baud של ה- UART.קצב ההעברה יכול להיות שונה ממערכת למערכת אך יש קצבי העברה מקובלים יותר מהאחרים :

2400 Bits Per Second
4800 Bits Per Second
9600 Bits Per Second
14400 Bits Per Second
19200 Bits Per Second
38400 Bits Per Second
57600 Bits Per Second
115200 Bits Per Second

<u>UART 8051</u>

יחידת התקשורת במיקרו-בקר 8051 נקראת UART והיא כוללת שני אוגרים האחראים על העברת התקשורת SCON ו SBUF .
הכניסה, RxD רגל 10 ,פורטp3.0 משמשת לקליטת מידע לתוך המיקרו-בקר .
היציאה, TxD רגל 11 ,פורט p3.1משמשת לשידור מידע מתוך המיקרו-בקר.

האוגר SCON = Serial Communication שכתובת 98H בזיכרון ה, SFR אחראי על פרוטוקול התקשורת הטורית. מבנה האוגרSCON :

Name	SM0	SM1	SM2	REN	TB8	RB8	TI	RI
Bit	7	6	5	4	3	2	1	0
Address	9Fh	9Eh	9Dh	9Ch	9Bh	9Ah	99h	98h

איור 9

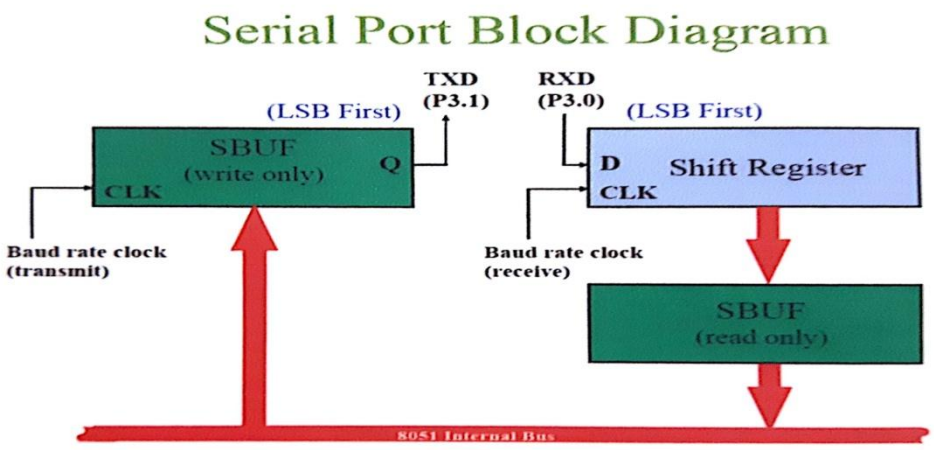
SM0	סיבית הקובעת את אופן העבודה של התקשורת הטורית
SM1	סיבית הקובעת את אופן העבודה של התקשורת הטורית
SM2	סיבית המאפרת תקשורת בין מספר בקרים
REN	כאשר REN=1 פעולת הקליטה פעילה. ה UART -יקלוט מידע וימלא את האוגרSBUF . כאשר REN= 0 פעולת הקליטה אינה אפשרית.
TB8	מייצג את הסיבית התשיעית כאשר הUART מקבל 9 סיביות מידע
RB8	מייצג את הסיבית התשיעית כאשר הUART שולח 9 סיביות מידע
TI	סיבית זאת עולה ל 1 כאשר ה UART מסיים לשלוח את המידע הנמצא באוגר SBUF
RI	סיבית זאת עולה ל 1 כאשר ה UART מסיים לקבל את המידע הנמצא באוגר SBUF

<u>אוגר SBUF</u>

שליחה או קבלה של מידע נעשית דרך האוגר ,Serial data bufer = SBUF שכתובתו H 99 בזיכרון הSFR

- שליחת מידע : כדי לשלוח מידע יש תחילה לאחסן אותו באוגר SBUF ורק לאחר מכן ה UARTיכול לשלוח אותו בצירוף סיבית ההתחלה והסיום. בסיום שליחת המידע, הסיבית TI עולה ל-"1 "ובמקרה שהפסיקה הטורית מאופשרת באוגר, IE תתקבל פסיקה. וקטור הפסיקה של תקשורת טורית נמצא בכתובת 23Hשם יש לרשום את שגרת הפסיקה

קבלת מידע : בעת קבלת מידע עולה הסיבית RI ל-"1 "ובמקרה שהפסיקה הטורית מאופשרת באוגר IE תתקבל פסיקה. על מנת לקרוא את הנתון שהתקבל, יש לפנות לאוגר SBUF.

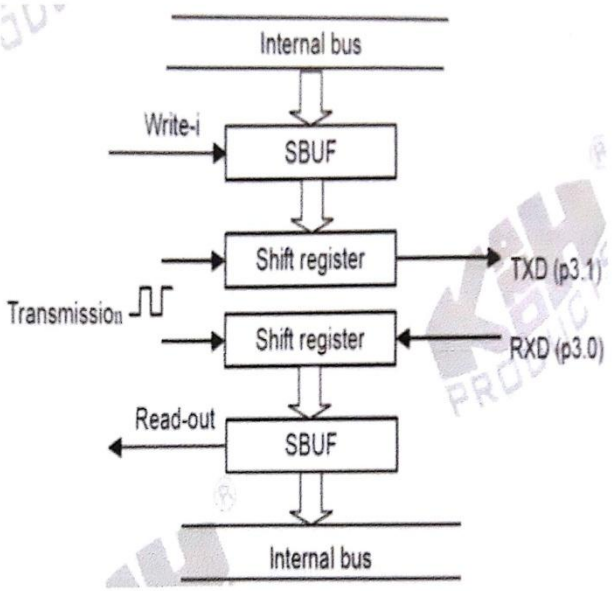


איור 10

הדק היציאה TxD והדק הכניסה RxD

הכניסה RxDרגל 10 פורט P3.0משמשת לקליטת מידע לתוך המיקרו-בקר .
היציאה TxD רגל 11 פורט P3.1משמשת לשידור מידע מתוך המיקרו-בקר .

יחידת ה UART -במיקרו-בקר 8051 עובדת בשיטתFull Duplexכלומר מסוגלת לקבל ולשלוח מידע בו-זמנית. דבר זה אפשרי הודות לשיטת החיבור של האוגר SBUF
אוגר זה מכיל למעשה שני אוגרים נפרדים הנמצאים באותו מיעון :
1.מידע הנכתב ל SBUF -לשם שידור מגיע לאוגר המכונה DATA TX המחובר ליציאה TxD של המיקרו-בקר 8051 .
2.מידע הנקרא מה SBUF -מגיע מאוגר המכונה DATA RXהמחובר לכניסה RxD של המיקרו-בקר 8051.



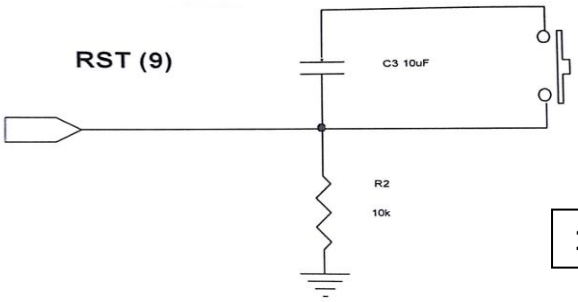
איור 11

מעגל ה-RESET

מעגל ה RESET נועד לאפס את המיקרו בקר על ידי לחיצת המשתמש על לחצן ה RESET. המעגל בנוי ממפסק במצב נתק (OPEM NORMALLY) פרט לרגע הלחיצה לקצר , קבל ונגד . המעגל מתחבר לרגל 9 של הבקר.
תהליך האתחול מתבצע כאשר רגל הRESET מקבלת 1 במצב זה מתאפסים האוגרים המיוחדים ומונה התוכנית . בשביל האתחול יתבצע בצורה תקינה יש לספק RESET לאורך של 10 מילי SEC לפחות . לאחר ה RESET מתחילה פעולת האתחול שאחריה ניתן להתחיל את העבודה עם הבקר.
במקרה והמערכת נתקעת ניתן ללחוץ על RESET .

Register	Content
Program counter	0000h
Accumulator	00h
B register	00h
PSW	00h
SP	07h
DPTR	0000h
All ports	FFh

טבלה 5

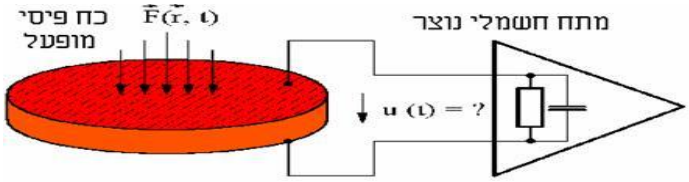


איור 12

למעגל הזה שלושה תפקידים :
1.Power On Reset - אתחול הספק והמתח .
2. מניעת נזק לרכיב במקרה של נפילת מתח רגעית ומהירה
3.לאפשר אתחול ידני, לפי רצון המפעיל.

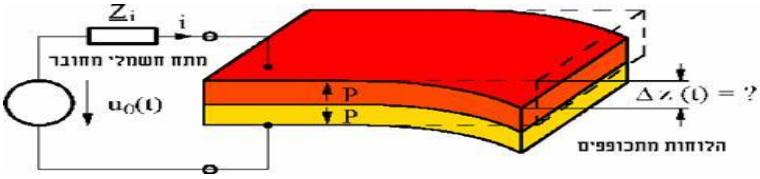
מעגל גביש

הגביש כולו הוא מנגנון מכאני היוצר אפקט של מעגל חשמלי מסוים. בדרך כלל הוא עשוי קוורץ . מפרמטרים שקובעים את תכונותיו המכאניות והחשמליות הם עוביו ומבנהו.
גבישים ניתן למצוא בטבע או ניתן לייצר על ידיי תהליך שנקרא "גידול הגביש"
את הפעולה החשמלית של הגביש אפשר להבין על ידי חקר התופעה הפייזו-אלקטרית, שמופיעה לא רק בקוורץ אלא גם בחמרים נוספים: במלח רושל, בכמה סוגי חמרים קרמים לתופעה יש 2 הביטים שצריך להבין :
1. אם נפעיל לחץ מכני במרכז לוח הגביש, נגרום לתנועת אלקטרונים בתוכו ולהצטברות מטען חשמלי בין קצות הגביש



איור 13

2. אם נחבר מתח חשמלי בין קצות הגביש נקבל תנועה מכאנית שלו, הגביש יתכופף סביב המרכז שלו.מתח מתחלף יגרום לתנועת הגביש בכוונים הפוכים, כלומר לגביש "מתנדנד" מכאנית.



איור 14

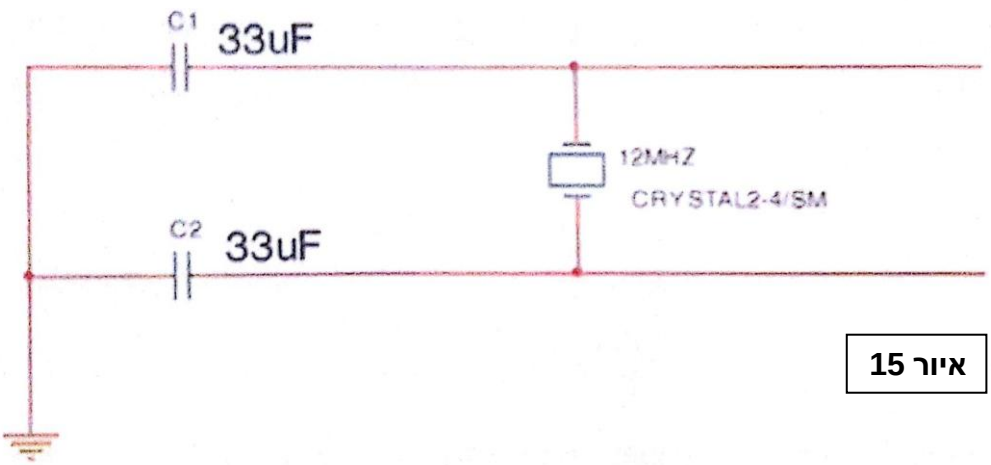
אם נאחד את 2 תופעות אלו נקבל מתנד . המתנד שוויצר יצור תנודות בלתי פוסקות בתדירות מסוימת . התופעה הזאת מנוצלת היטב על מנת לבנות מתנדים מדויקים לדוגמא למיקרו בקרים .

תיאור כללי ב8051

תפקיד הגביש בעזרת שני קבלים שערכם נקבע כבר ע"י היצרן ובעזרת הכניסות 1XTAL XTAL2 ליצור מתנד בתדר המתאים לתזמון פעולת המיקרו ומחזורי הפס שלו.
CLOCK זה מתאים את קצב הביצוע למהירות התגובה של הרכיבים.
מהירות זו שווה ל12 MHZ.

עיקרון הפעולה

עיקרון פעולתו של המתנד הוא במגבר ובמערכת המשוב β היוצרים תנודות בתדר שעבורו המופע בחוג סגור הוא 0 או 360. כמו כן, הגבר המערכת גדול מעט מ1.
הגביש גורם ליצירת מעגל תהודה עם גורם טיב Q גבוהה מאוד לצורך יציבות התדר ויציבות הטמפ'.



איור 15

RS232

כדי לאפשר תקשורת בין ה DTE ל DCE צריכים להגדיר באופן מדויק את המישק ביניהם .
ניתן לעבוד בניהם בעזרת פרוטוקול RS232.

* DTE -DATA TERMINAL EQUIPMENT.
* DCE– DATA CIRCUIT TERMINATING EQUIPMENT .

תקן RS232

RS232 הינו כינוי המקובל לסדרת תקנים לאותות בקרה והעברת נתונים באופן בינארי סריאלי בין DTEל DCE.

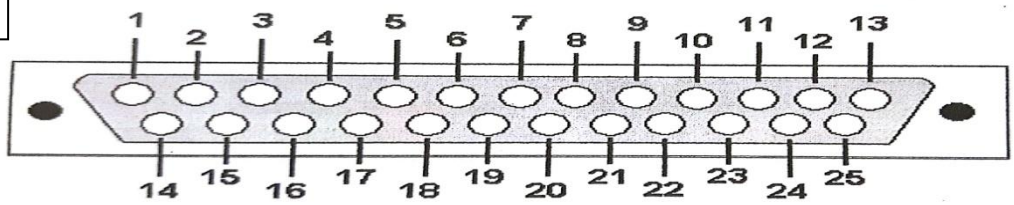
ההתקן מגדיר ארבעה סוגי תכונות :

1. תכונות מכאניות – ממדי המחברים וצורתן , עובי המחברים , מרחק וכו'.
2. תכונות חשמליות – רמות מתח וכו' .
3. תכונות תפקודיות – תפקיד של כל קו המחבר בין DTEל DCE.
4. תכונות נוהליות – סדר אירועים של תהליך הקמת קשר והעברת נתונים .

נסביר כל אחד מין התכונות :

1. המפרט המכאני : מציין מחבר ובו 25 הדקים , שימוש במחבר זה מחייב שהכבל יכלול 25 קווים.

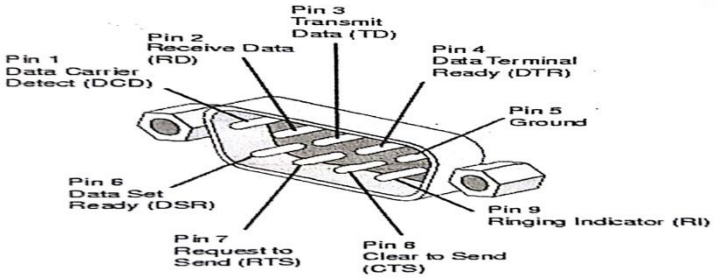
איור 16



2 Transmit Data (TxD)
3 Receive Data (RxD)
4 Request to Send (RTS)
5 Clear to Send (CTS)
6 Dataset ready (DSR)
7 Signal Ground
8 Data Carrier Detect (DCD)
15 Transmit Clock
17 Receive Clock
20 Data Terminal Ready (DTR)
24 Auxiliary Clock

בפועל משתמשים במחברים בין 9 הדקים הכוללים את הדק 1-8 והדק 20 .

איור 17



Pin 1	DCD
Pin 2	RXD
Pin 3	TXD
Pin 4	DTR
Pin 5	GND
Pin 6	DSR
Pin 7	RTS
Pin 8	CTS
Pin 9	RI

2. מפרט חשמלי : המפרט קובע שימוש באותות ספרתיים. מתח מתחת ל 3V- מתפרש כסיבית שערכה 1 לוגי . ומתח גבוה מ 3Vמתפרש כערך של 0 לוגי . זהו קידוד NRZ-L.

טבלה 6

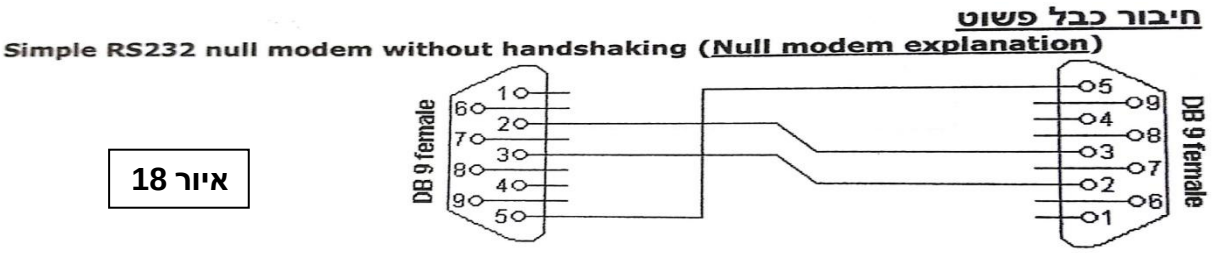
3. המפרט התפקודי : מתאר איזה מעגל מחובר לכל אחד מין ההדקים ומה תפקידו של כל מעגל כזה . נציג את הטבלה של צד ה DTE :

מספר הדק	שם	תפקיד
1	אבחון גל נושא Carrier Detect	המודם מזהה שידור אליו (גל נושא בקו הטל')
2	קליטה Receive	העברת נתונים מ-DCE ל- DTE
3	שידור Transmit	העברת נתונים מ- DTE ל- DCE
4	ציוד DTE מוכן Data Terminal Ready	DTE מחשב או מסוף פועל
5	רמת יחוס משותפת Common Ground	רמת יחוס משותפת לכל המעגלים 0v
6	ציוד DCE מוכן Data Set Ready	DCE (למשל מודם) פועל
7	בקשת שידור- Request to Send	DTE מודיע על רצונו לשדר
8	אישור שידור Clear to Send	DCE מוכן לקלוט, תגובה לבקשת שידור
9	אבחון צלצול Ring Indicator	DCE מזהה אות צלצול בקו הטלפון

4. המפרט הניהולי : הוא פרוטוקול המגדיר את סדרות האירועים הדרושים לכל ישום .

חיבור DCE

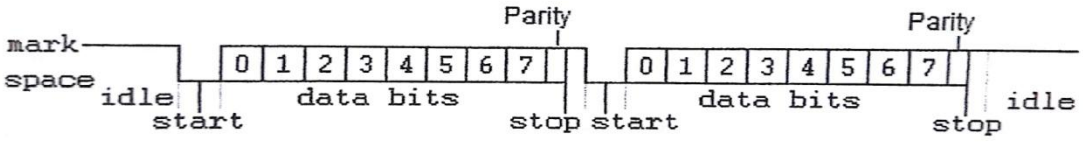
יש כמה צורות של חיבורים . בפרויקט שלנו חיבורנו בסוג " חיבור פשוט " :



איור 18

שידור אסינכרוני ב RS232

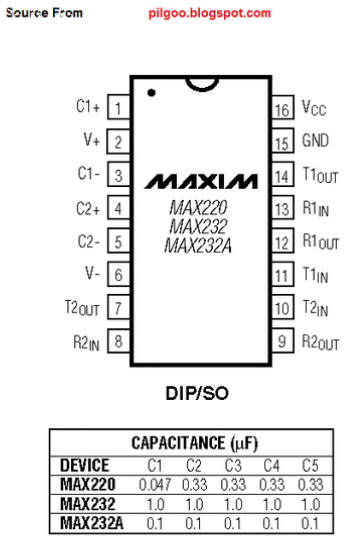
נציג את תרשים שמתאר מעבר BYTE בכבל RS232.



איור 19

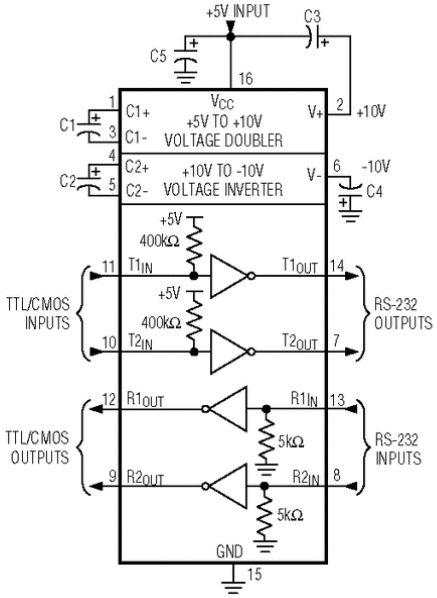
MAX232

תקן RS232 קובע ש 1 לוגי יהיה בין 3- ל 25- וולט . ו 0 לוגי יהיה בין 3 ל 25 וולט .
בתקני TTL 1 לוגי הוא 5 וולט ו 0 לוגי הוא 0 וולט. כדיי לתאם הין התקני TTL ל RS232 נחבר רכיב שנקרא MAX232.
אלו רכיבים מוכללים המאפשרים לתאם העברת מידע בין RS232 להתקן נוסף שפועל במתחים שונים מ 12 וולט .
* בעזרת קבלים יוצרי מתח גבוה ממתח האספקה שהוא 5 וולט.



CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	0.047	0.33	0.33	0.33	0.33
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

איור 20



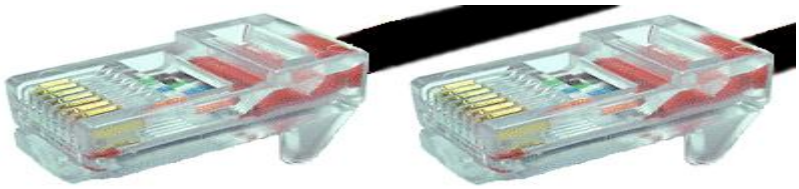
איור 21

איור 22

כבל רשת CAT 5e

כבלי רשת מחברים למחשבים אישיים ב Ethernet על מנת שיוכלו לחלוק מידע ורכיבים כגון מדפסות וגישה לאינטרנט. כבלי רשת מגיעים במגוון סוגים ואיכויות הנקראות בשם "קטגוריות " (Categories)
Categories – מידע המייצג את קצב השידור וסוג השידור.

כבלי CAT5 הם דוגמא לכבילה מסוג "זוג שזור", מכיוון שהגידים בכבל שזורים בזוגות. ניתן לזהות אלו גידים שזורים אחד בשני עפ"י צבעם - כל זוג שזור צבוע באותה מתכונת: גיד צבעוני וגיד לבן עם פסים בצבע של הגיד עימו הוא שזור. הצבעים האפשריים הם: כתום, ירוק, כחול, וחום.
כבלי CAT5 מכילים בקצותם מחבר מסוג RJ45 שמתחברים לשקעי RJ45. שקעי RJ45 הם סוג השקע הבסיסי והנפוץ ביותר של ממשק כרטיסי רשת המשמשים לחיבור מחשבים יחדיו על מנת לחלוק משאבים.ניתן למצוא שקעי RJ45 במגוון רחב של ציוד תקשורת כגון כרטיסי רשת ,(NICs) רכזות ,(Hubs) מתגים (Switches) ונתבים(Routers) .



איור 23

בפרויקט שלנו השתמשנו בכבל זה כ "חומר גלם " זול ואיכותי להארכת כבל ה RS232.
חתכנו את מחברי ה RJ45 וחברנו במקומם מחברי (D –TYPE). החיבור של מחבריי – D TYPE בקצוות נועד בשביל להתאים את הכבל לחיבור RS232.

בנוסף לכך בכבל CAT5e חתכנו את כל הגידים פרט לגידים 2\3\5 . 2 משמש לשידור 3 לקליטה ו 5 מחובר לאדמה.

D-type

הממשק D-type הוא ממשק מקבילי הבנוי מ- 25 רגליים . ניתן גם להשתמש ב 9 רגליים .
בפרויקט שלנו השתמשנו במחבר זה על מנת לחבר בין RS232 ל MAX232 .
מחבריD-type נמצאו מתאימים לבצע חיבור זה .

איור 24

איור 25

USB-Universal Serial Bus

זהו סטנדרט תעשייתי שפותח באמצע שנות ה90 והוצג לראשונה בשנת 1995 .התקן מגדיר את הכבלים ,המחברים ופרוטוקולי התקשורת . ההתקן החליף התקנים כמו RS232.

ל USB מספר גרסאות למספר מהירויות שונות :

טבלה 7

1995	גרסה 1.0גרסה ראשונה
1998	גרסה 1.1LS=Low Speed - 1.5 Mbps FS=Full Speed - 12 Mbps
2000	גרסה 2.0 High speed - 480 Mbps
2008	גרסה 3.0Super speed - 5 Gbps
2013	גרסה 3.1Super speed+ - 10 Gbps

BUS POWERED

חיבורי ה USB כוללים 2 מוליכי אספקת מתח של כ 5 וולט וזרם עד כ 500mA וזה מאפשר להתקנים לקבל מתח דרך הכבל או דרך המרכזת . ישנם 3 סוגי התקנים :

1. התקנים הצורכים עד 100 mA
2. התקנים הצורכים עד 500 mA
3. התקנים עם ספק כוח משלהם SELF POWERED.

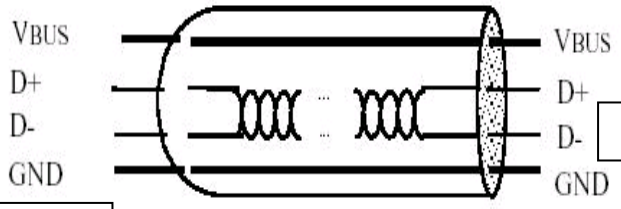
חיבור התקנים

תקן ה USB מאפשר לחבר עד 128 התקנים למחשב מארח . כדי לחבר התקנים רבים ניתן לחבר מרכזיה \ קופסת פיצול.

SUSPEND

התקן נדרש לרדת מדרישת ההספק שלו (השהייה) אם לא הייתה פעילות למעלה מ 3 mA .

חיבורי USB



איור 26

טבלה 8

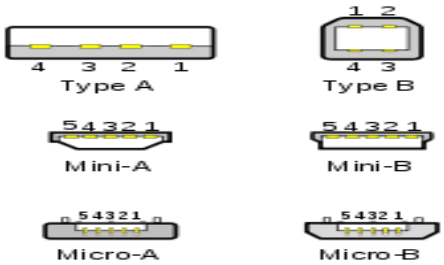
מס' פין	צבע חוט	שם הסיגנל	הסבר
1	אדום	VBus	מתח של כ- 5V
2	לבן	D-	סיגנל מינוס
3	ירוק	D+	סיגנל פלוס
4	שחור	Ground	אדמה

הסטנדרט הבסיסי מדבר על שני סוגי מחברים : מחברי Aומחברי B.

המחברים הם שונים כדי שלא לאפשר חיבור בין 2 מחשבים או בין שני התקנים (החיבור התקין הוא בין מחשב להתקן \התקנים).

עם הזמן התברר שמחברי A ו B שהיו גדולים מידי למחשבים וטלפונים ולכן הקטינו אותם לגודל שאנו מכירים היום .

איור 27



USB לעומת RS232

USB	RS232
מהיר יותר	איטי יותר
מחברים מהירים, מתאים לעבודה של מחשב מול מס' רב של התקנים (עד 128) על BUS אחד.	תקשורת מנקודה לנקודה (חיבור בין שני התקנים בלבד)
only Half (Full from 3.0 version)	Full & Half
מבנה שכבתי עד לרמת היישום, כולל מנגנוני תיקון וניהול. מצריך מערכת הפעלה במחשב	רמה פיזית בלבד

טבלה 9

USB SWITCH

על דגם הבקר 8051 שקננו יש את רכיב ה USB SWITCH .
הרכיב נותן אפשרות לנתק ולחבר את הUSB מבלי לנתק את הכבל . הניתוק נגרם למעשה על ידיי הרמת המתג למעלה. וחיבור נגרם על ידיי הורדת המתג חזרה מטה .

ממיר USB RS232

איור 28

ממיר ,ATC-810 הינו ממיר תקשורת USB RS232תקשורת טורית/סריאלית.
מתאים בעיקר למחשבים ניידים ללא .
המחשב רואה את יחידת ה ATC-810-כיציאת RS232 לכל דבר שאינה שונה מיציאת ה RS232- הסטנדרטית שנמצאת במחשב.
את מספר היציאה (COM) אפשר לראות תחת ה Device-Manager

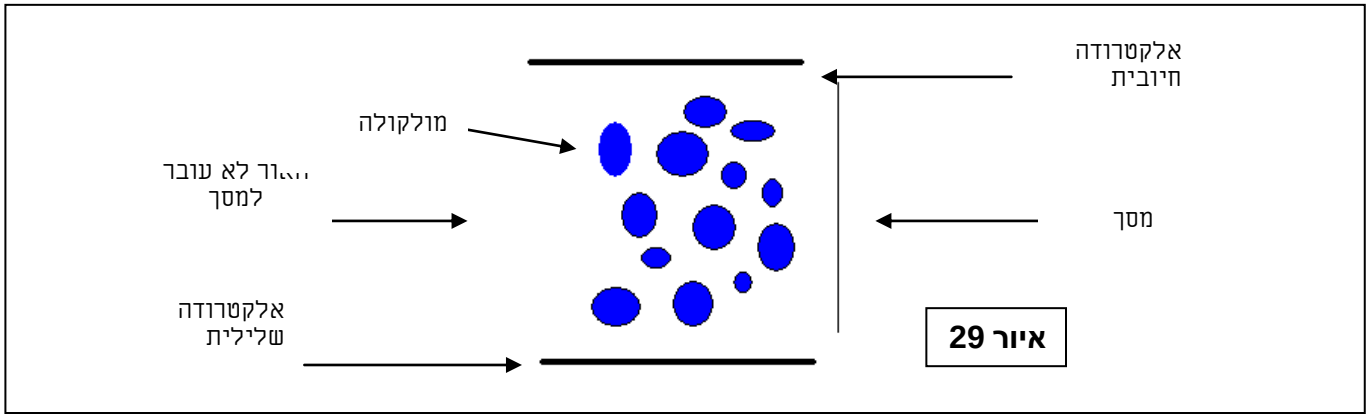
מאפיינים עיקריים:
מתאים ליציאת USB בתקן 1.1 ו-2.0
מהירות תקשורת עד Mbps12
שבב FTDI לשימושים ביתיים ותעשייתיים
תמיכה ב Remote wake up-וניהול צריכת חשמל
מהירות תקשורת סריאלית: עד Kbps460.8
דרייבר ל Win98/2000/XP/Vista, Linux, WinCE 5.0 :
חיבור :RS232 זכרDB9
נוריות חיווי לתקשורתRX/TX
הגנת TXD/RXD מפני עליית מתח עד 600 , Wהגנת ESD עד 15 KVלכל הקווים
הספקת מתח ישירות מיציאת ה USB-ללא צורך בספק כוח נפרד

LCD-Liquid Crystal Display

LCD ובעברית תצוגת גביש נוזלי. בעזרת הרכיב ניתן להציג הודעות ונתונים.
הגביש הנוזלי הינו חדיש יותר מהלד ופיתוחו נעשה בשנת 1968 במעבדות RCA בארה"ב.
לעומת הלד אשר מקרין אור הוא לא מסוגל להקרין אור ומשתמש באור הסביבה

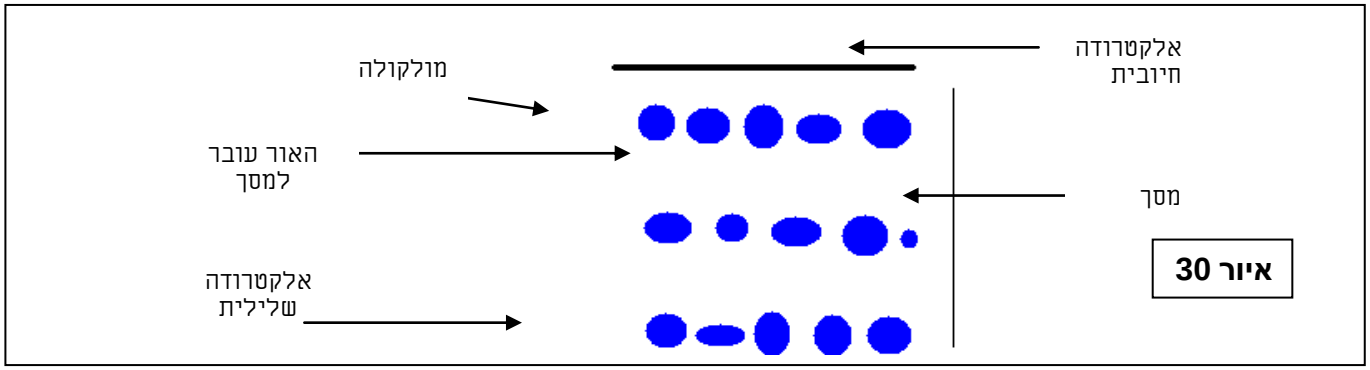
מבנה גביש נוזלי

הנוזל שבתוף הגביש זוהי תמיסה שבמצב הרגיל המולקולות שלה מסודרות בצורה רנדומאלית ללא סדר מסוים. כאשר מקרינים אור מצד אחד של הלד במצב זה קרני האור יפגעו במולקולות אשר פזורות לכל עבר , הקרניים יפגעו במולקולות ולא יעברו לצד השני ולא נקבל תצוגה על הצג.



איור 29

כשאר מפעילים שדה חשמלי בין שני הלוחות המולקולות שביניהן הסתדרו בצורה מסוימת . המבנה שנוצר למולקולות יצר פתחים לאור לעבור דרכו ולכן נוכל לקבל אור בצד השני של המסך.



איור 30

התמיסה נמצאת בין שני לוחות זכוכית – לוח אחד מצופה שכבת זהב דקיקה והלוח השני בנוי מלוחות זהב דקיקים עד כדיי שקיפות . התכונה המיוחדת של התמיסה היא שכאשר יש שדה חשמלי בין הלוחות התמיסה הופכת לשקופה. כלומר אם נספק אל אחד הלוחות העליונים שדה חשמלי ביחס ללוח הזהב התחתון תהפוך התמיסה שמתחתיו לשקופה. אם יונח רקע צבעוני מתחתיו הוא יראה. ברגע שיופסק השדה החשמלי, תחזור התמיסה להיות אטומה והרקע יעלם.

תצוגת LCD - 2*16 היא תצוגה המורכבת מ 2 שורות כאשר בכל שורה יש 16 תווים. כל תו מורכב ממטריצה של 5*7 נקודות (PIXELS) של גביש.

יתרונות וחסרונות

טבלה 10

יתרונות הרכיב	חסרונות הרכיב
אינו צורך זרם	צריך להימצא ליד מקור אור בכדיי לפעול
הרכיב קטן	

מאפייני הרכיב

- 2 שורות , בכל שורה 16 תווים.
- מקור הזנה יחיד של 5v .
- מגן החזרי תאורת רקע.
- מותאם לרכיבי TTL ו-CMOS.
- תואם קוד ASCII.

רכיבים פנימיים

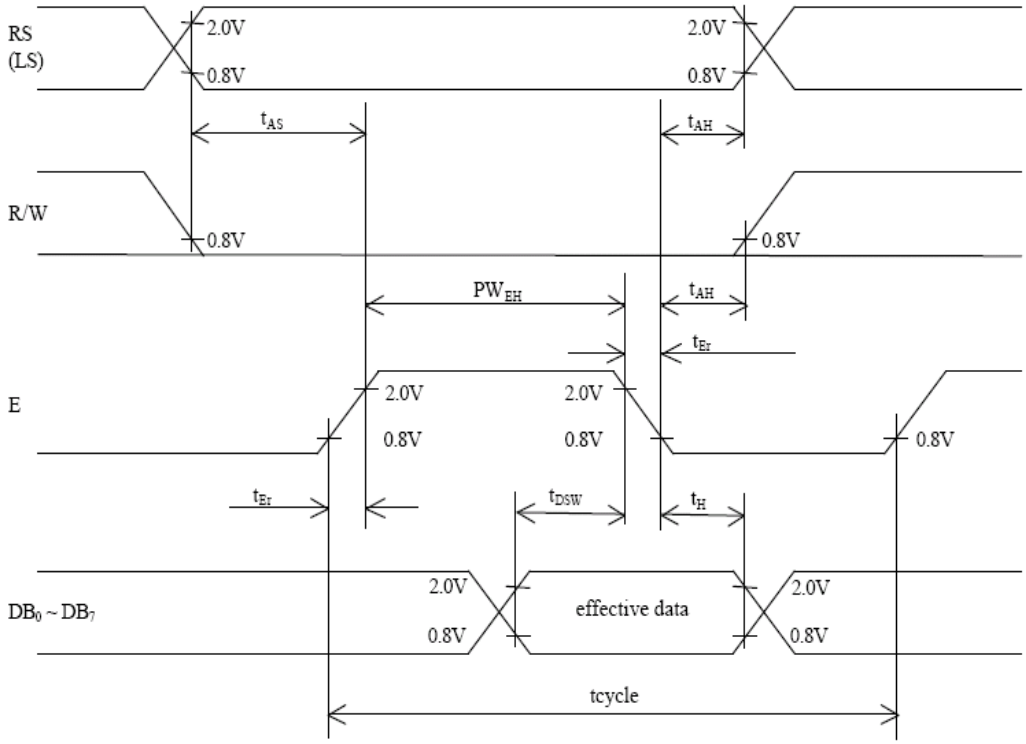
1. **IR- אוגר 8 ביט- Instruction Register -** רגיסטר ההוראות, משמש לאחסון קודי ההוראות הכוללים: ניקוי תצוגה, הזזת סמן, הבהוב, ועוד...
2. **DR- אוגר 8 ביט -** Data Register - רגיסטר נתון זמני לאחסון המידע הנכתב או הנקרא. המידע נכתב ל DR ועובר אוטומטית ל- DDRAM או ל- CGRAM בצורה פנימית.
3. **מונה כתובות AC (Address Counter)** - ישנו מונה כתובות שתפקידו להצביע על הכתובת ב- DDRAM או ב CGRAM שאליה שולחים נתון.
4. **דגל BUSY** - ישנם מצבים בהם שולחים נתון ל LCD ומיד אחריו עוד נתון אך ה LCDלא פנוי בכדי לקרוא את הנתון החדש . לטיפול בבעיה נשתמש בדגל התצוגה. כאשר1 =BUSY, זה מסמן שהמסך כעת נמצא בפעולה פנימית .
כאשר BUSY=0 , כלומר התצוגה סיימה כתיבת תו או ביצוע הוראה, נוכל לבצע את הפעולה הבאה.
5. **זיכרון נתוני תצוגה (Display Data Ram –DDRAM) –** זהו זיכרון RAM אשר מאחסן את הנתונים שהמשתמש קורא או כותב .
6. **זיכרון קריאה בלבד מחולל תווים (Character Generator Rom – CGROM)-**
זיכרון קריאה בלבד זה ביחד עם מחולל התווים שבתוך אחד המעגלים המשולבים של התצוגה יכולים להציג 192 תווים שונים. המשתמש שולח לזיכרון ה DDRAM את התווים (בקוד אסקי) אותם הוא רוצה להציג במסך. מחולל התווים יודע לקחת כל תו , לפנות לכתובות המתאימות בזיכרון ה CGROM וליצור את התו בתצוגה.
7. **זיכרון RAM מחולל תווים (Character Generator Ram – CGRAM)-**
זיכרון RAM ומחולל התווים מאפשר למשתמש לכתוב תווים חדשים שאינם נמצאים בזיכרון ה CGROM . ניתן לתכנן 8 סימנים חדשים של 5*7 או 4 סימנים של 5*10.

הסבר הדקיי הרכיב

סימון	מספר פין	תפקיד
VSS	1	אדמה
VCC	2	5V
VEE	3	אדמה
RS	4	**(Register Select) בחירת רגיסטר.** **0=RS - כאשר המשתמש רוצה לשלוח לתצוגה הוראות שונות כגון: ניקוי תצוגה, הזזת סימן, הוראות אתחול .** **RS=1 - כאשר המשתמש רוצה לשלוח לתצוגה נתונים כלומר רוצה לכתוב תו (מספר, אות וכדומה).**
R\W	5	(Read/Write) קריאה/כתיבה. זוהי רגל כניסה ל LCD שבעזרתה המשתמש אומר לתצוגה אם הוא רוצה לקרוא מה-LCD או לכתוב ל LCD הוראה כלשהי או נתון. R\W=1=Read R\W=1=Write
EN	6	(Enable) של התצוגה,הרגל פעילה בגבוה .
DB0-DB7	7-14	8 רגלי נתונים שמשמשים ככניסות/יציאות . בעזרת קווים אלו כותבים פקודות או נתונים לתצוגה. סיבית DB7 -דגל "תצוגה עסוקה" - (Busy Flag)- אם ברגל זו יש "1" התצוגה עסוקה ואם "0" אז היא פנויה

טבלה 11

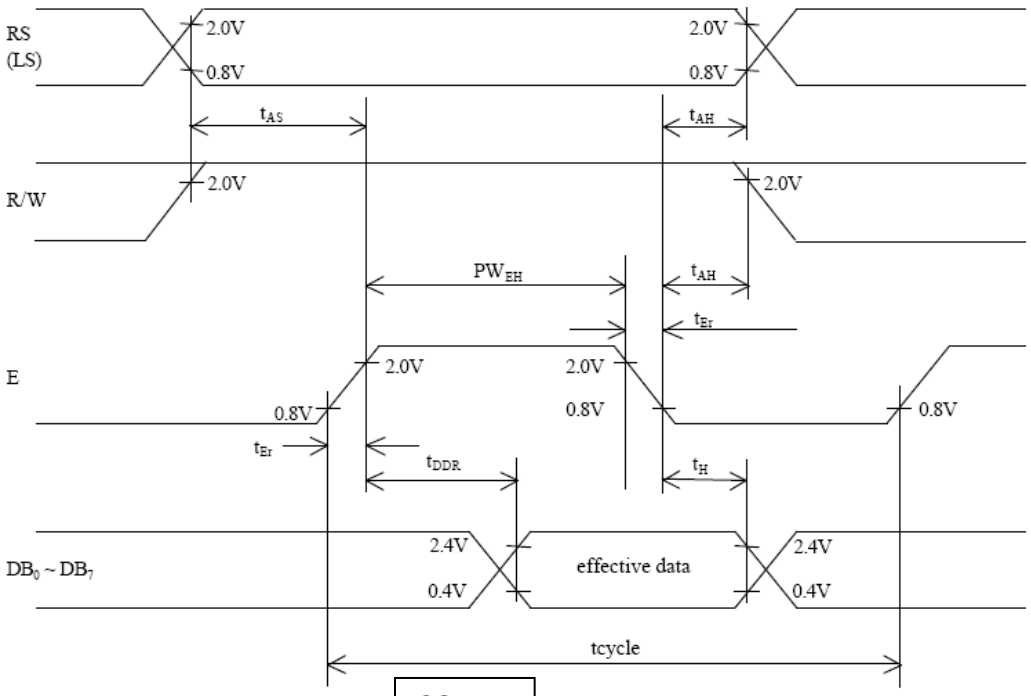
מחזור כתיבה לתצוגה



איור 31

1. RS=0\1 פנייה לרגיסטר ההוראה או לרגיסטר הנתון.
2. RW=0 ביצוע פעולת כתיבה.
3. EN=1
4. שליחת הנתון/הוראה ב DB0-DB7.
5. EN=0 נעילת הנתון לתצוגה.
6. יש לבדוק האם ה- LCD סיים לבצע את הפעולה שנשלחה(בדיקת דגל BUSY אוו תוכנית השהייה).
7. כאשר ה- LCD פנוי ניתן לבצע מחזור כתיבה חדש.

<u>מחזור קריאה מהתצוגה</u>:

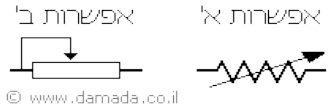


איור 32

1. RS=0 פנייה לרגיסטר ההוראה.
2. RW=1 ביצוע פעולת קריאה.
3. EN=1
4. תצוגה שולחת נתונים מ- DB0-DB7.
5. מכינים נתון לבקר ו EN=0.
6. בדיקת BUSY בסיבית DB7 .

נגד משתנה

נגד משתנה ("פוטנציומטר") הוא רכיב חשמלי אשר מתנגד למעבר הזרם דרכו בצורה משתנה . סימון חשמלי:



להבדיל מנגד רגיל בנגד משתנה המשתמש יכול לבחור התנגדות מטווח של התנגדויות. הנגד המשתנה בנוי ממוליך ארוך שהתנגדות שלו ידועה מראש. ההתנגדות משתנת על ידי שינוי אורך המוליך אשר הזרם עובר דרכו .

איור 33

בדרך כלל לנגדים משתנים יש שלוש רגליים : שתי רגליים בדומה לנגד רגיל שהן למעשה קצוות המוליך. הרגל השלישית הינה הזחלן שאת מקומה ניתן לשנות לאורך רכיב , שינוי מיקום הזחלן במוליך הוא למעשה שינוי ההתנגדות. החיבור לנגד משתנה נעשה בין אחד הקצוות לזחלן (הרגל האמצעית). את הקצה השני נהוג לחבר לזחלן.

השימוש הנפוץ ביותר של הנגד השתנה הוא כאשר רוצים לווסת ערך של מתח מסוים. והחיבור נעשה כך : מתח הכניסה ל 2 קצוות הזחלן ומתח היציאה מתקבל בין 1 הקצוות לזחלן .תחום הערכים לפי חוק אוהם של מתח המתקבל ינוע בים 0 ל מתח הכניסה .

ראוסטט (Rheostat) ישנם מצבים בהם יש צורך שהנגד יעמוד במתחים מאוד גבוהים. במקרים אלו פוטנציומטר רגיל כבר לא יספק את השירות המבוקש והפתרון הקרוי ראוסטט. ברכיב זה חוט ארוך אשר מלופף על משטח גלילי, הוא זה שאחראי על ההתנגדות הרצויה. הרגל השלישית שלו- הזחלן, עשויה ממתכת שנוגעת בבסיס הגליל ובכך מאפשרת סיבוב. באופן שכזה מתאפשר שינוי אורך החוט- דבר שמאפשר את העצמת ההתנגדות.

איור 34

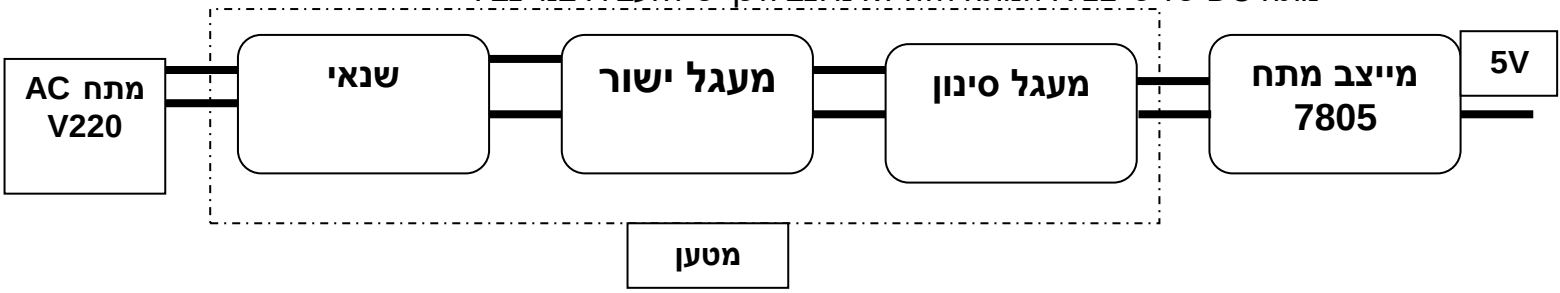
תגובתם של נגדים משתנים לתזוזת הזחלן היא בדרך כלל קרובה לליניארית. יש כאלה שמתוכננים דווקא לשינוי התנגדות לוגריתמי (או אנטי לוגריתמי), במיוחד כאלה המתוכננים לשימוש במערכות שמע.
ברכיבים קטנים, כגון אלו המופיעים בתמונה, מתבצע שינוי ההתנגדות על ידי סיבוב חוגה במרכז הרכיב, וזו מזיזה את הזחלן. בנגדים משתנים בעלי ממשק דיגיטלי נקבעת ההתנגדות ללא מגע יד אדם, על ידי תוכנה.

בפרויקט שלנו השתמשנו בנגד משתנה על מנת לשנות את המתח שמגיע ל lcd ובכך לשלוט על הבהירות שלו.

איור 35

מייצב מתח 7805

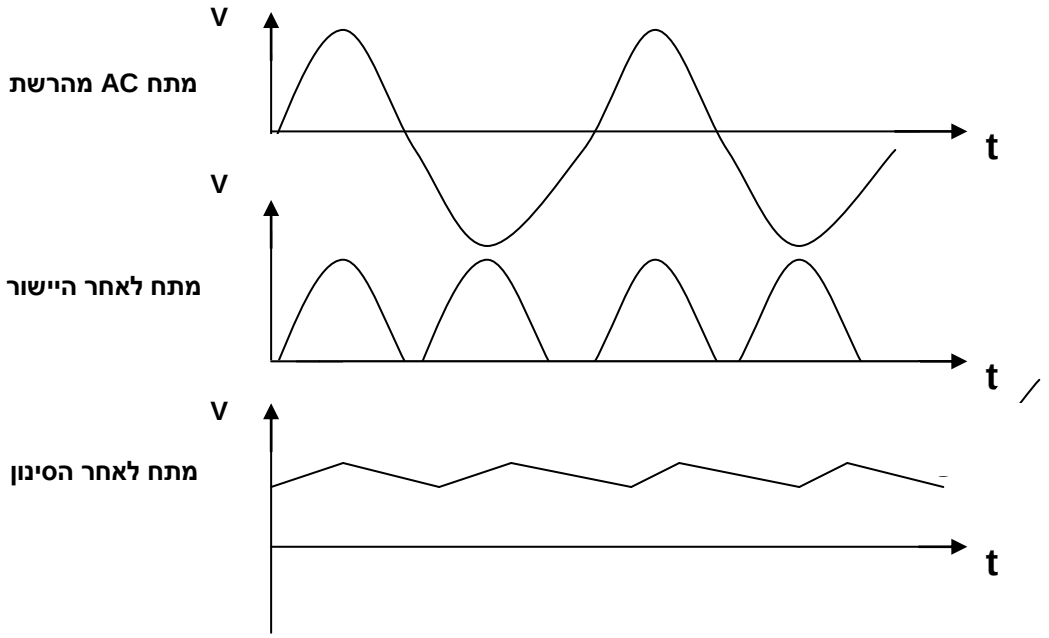
ספק הכוח מורכב ממטען חיצוני המתחבר למתח הרשת וכן ממייצב ל- 5V. המטען מוציא מתח DC של כ- V12. המתח הזה לא מיוצב ולכן יש להעבירו במייצב .



איור 36

המטען:

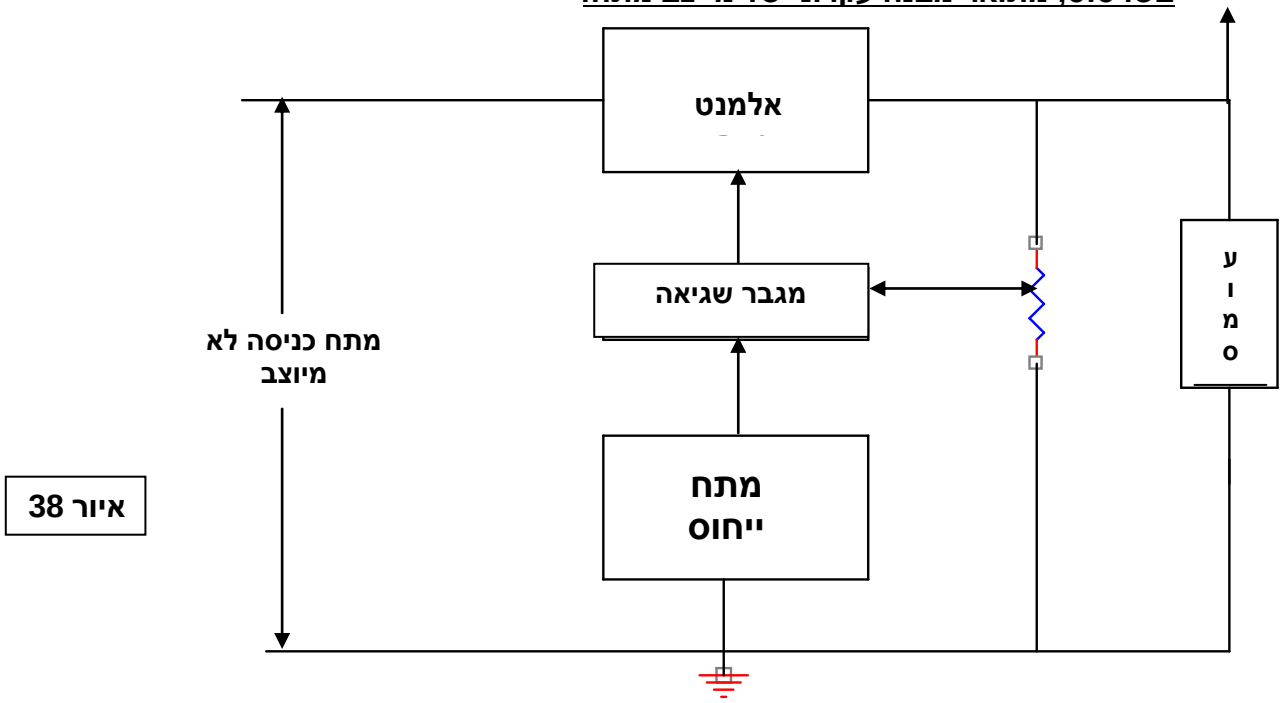
המטען מורכב משלושה חלקים: שנאי, מעגל יישור ומעגל סינון.
תפקיד השנאי הוא להוריד את מתח הרשת (220V, AC) למתח של כ- V12 , עדיין בצורה של מתח חילופין (AC).תפקיד מעגל היישור הוא להעביר את מה שקיבל מהשנאי (מתח החילופין), ולהפוך אותו למתח מיושר.לבסוף, מעגל הסינון גורם למתח המיושר להיות מתח ישר- DC.
ניתן לראות את השלבים שהמתח עובר בגרף:



איור 37

המייצב שייך למשפחת המייצב 78XX, שהיא משפחה של מייצבי מתח חיוביים.
ה- 78C05 הוא מייצב מתח ל- 5V, ומתח הכניסה אליו יכול לנוע מ- 7V ועד 35V. מתחת ל- 7V הרכיב יפסיק לייצב, ומעל 35V הרכיב עלול להישרף.
תפקיד המייצב הוא להוציא מתח ישר ויציב של 5V ביציאתו, וזאת ללא תלות (כמעט) במתח הכניסה, בעומס או בטמפרטורה.
המייצב הוא ל- 1A, כלומר הזרם יכול להיות מ- 0A ועד 1A (תלוי בעומס) והמייצב ימשיך לספק 5V.

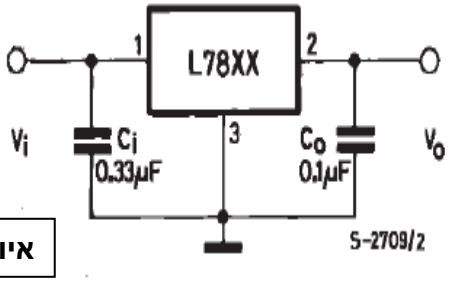
בשרטוט, מתואר מבנה עקרוני של מייצב מתח:



איור 38

מגבר השגיאה מקבל מצד אחד מתח ייחוס ומצד שני את מתח היציאה (דרך מחלק מתח – בעזרת הפוטנציומטר). ההפרש בין שניהם מוגבר ומסופק לרכיב הטורי. אם מתח היציאה רוצה לגדול, אז המתח למגבר השגיאה יגדל גם כן והוא יוציא מתח תיקון לרכיב הטורי. מתח זה יגדיל את התנגדות הרכיב הטורי כך שייפול עליו מתח גדול יותר, וביציאה המתח יקטן ויחזור ל- 5V.
בעזרת הפוטנציומטר ניתן לכוון את מתח היציאה.העלאת הזחלן של הפוטנציומטר כלפי מעלה תכניס מתח חיובי יותר למגבר השגיאה וזה יוציא פקודה לאלמנט הטורי להגדיל את התנגדותו כך שמתח היציאה יקטן. הורדת הזחלן כלפי מטה תגדיל את מתח היציאה.
במעגל המייצב ל 5 וולט אין את הפוטנציומטר בתוך הרכיב אלא מחלק מתח בין נגדים שייתן 5 וולט ביציאה.

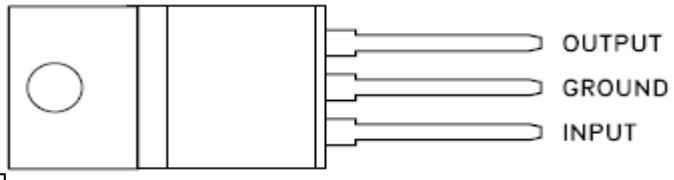
מעגל אופייני של חיבור המייצב נראה:



איור 39

למעגל המייצב יש בעיה של כניסה לנדנודים.
תפקיד הקבלים הוא גם מניעת סכנת התנודות ע"י קיצורם לאדמה וגם סינון רעשים לאדמה .

צורת הרכיב



איור 40

מקלדת

איור 41

מבנה המקלדת די פשוט, פסי נחושת מסודרים בשורות ועמודת שתי וערב ללא קשר ביניהם. בזמן לחיצה על מקש מתבצע קצר בין שורה לעמודה. במוצא המקלדת מופיעות רגלי השורה והעמודה אותן אנו מחברים לבקר המקלדת.

אופן פעולת המקלדת:
לחיצה על כול אחד מהמקשים במקלדת גורמת לקצר בין אחד מקווי העמודות עם אחד מקווי השורות כתוצאה מכך נסגר מעגל וניתן לפענח איזה מקש נלחץ.

Y1	Y2	Y3	Y4	
1	2	3	A	X1
4	5	6	B	X2
7	8	9	C	X3
*	0	#	D	X4

איור 42

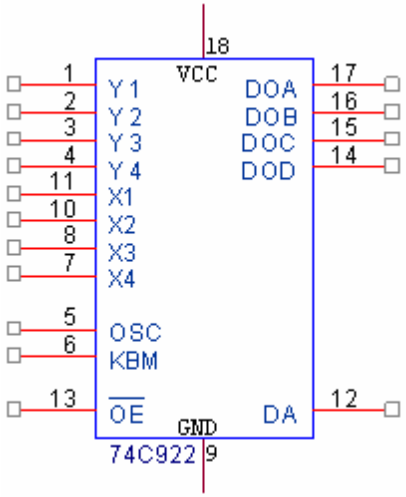
לחצן	עמודה	שורה
1	1Y	1X
2	2Y	1X
3	3Y	1X
4	1Y	2X
5	2Y	2X
6	3Y	2X
7	1Y	3X
8	2Y	3X
9	3Y	3X
*	1Y	4X
0	2Y	4X
#	3Y	4X
A	4Y	X1
B	Y4	X2
C	Y4	X3
D	Y4	X4

טבלה 12

תיאור הדקי הרכיב:
ROW 1- רגל משותפת לשורה הראשונה . מחברת בין רגל 1 של המפענח לרגל 4 של לוח המקשים.
ROW 2 - רגל משותפת לשורה השנייה. מחברת בין רגל 2 של המפענח לבין רגל 5 של לוח המקשים.
ROW 3- רגל משותפת לשורה השלישית . מחברת בין רגל 3 של המפענח לבין רגל 7 של לוח המקשים.
ROW 4- רגל משותפת לשורה הרביעית . מחברת בין רגל 4 של המפענח לבין רגל 2 של לוח המקשים.
COLUMN 1- רגל משותפת לעמודה הראשונה. מחברת בין רגל 11 של המפענח לרגל 5 של לוח המקשים.
COLUMN 2- רגל משותפת לעמודה השנייה . מחברת בין רגל 10 של המפענח לרגל 1 של לוח המקשים.
COLUMN 3- רגל משותפת לעמודה השלישית . מחברת בין רגל 8 של המפענח לרגל 3 של לוח המקשים.

מפענח למקלדת mm74c922n

תפקידו של המפענח הוא לזהות לחיצה במקלדת והוציא בהתאם קוד בינארי במוצא הבקר.
סרטוט של המפענח:



איור 43

הוא מפענח CMOS המכיל את כל הלוגיקה הדרושה לפענוח מלא של מקלדת הבנויה ממערך של 4*4 מתגי SPST .
המפענח כולל מעגל סריקה היוצר סריקת עמודות הקצב שנקבע על ידי קבל חיצוני .למפענח יש מעגל סינון ריטוטים הפועל גם הוא עם קבל חיצוני נוסף .
מוצא הרכיב הוא מחוצץ שזוכר את המקש האחרון שנלחץ במקלדת גם לאחר שחרורו .
תיאור הרגליים של הרכיב - Y3-Y0כניסות השורות של המקלדת, לכל אחד מקווים אלו מחובר נגד UP-PULL פנימי.

עיקרון הסריקה של הבקר הינו בשיטה של "אפס רץ", כול עוד לא נלחץ מקש במקלדת הבקר מזהה 1 לוגי בגלל נגדי הפול אפ, בזמן ההקשה נוצר קצר בין השורה לעמודה האפס לוגי היוצא מהבקר עובר לעמודה ומזוהה על ידי הבקר למעשה כלחיצה על מקש.

מאפייני המפענח:

1) נגדי פול אפ.
2) מוצאי TRI STATE
3) תחום מתחי הזנה גדול.
4) צריכת הספק נמוכה.

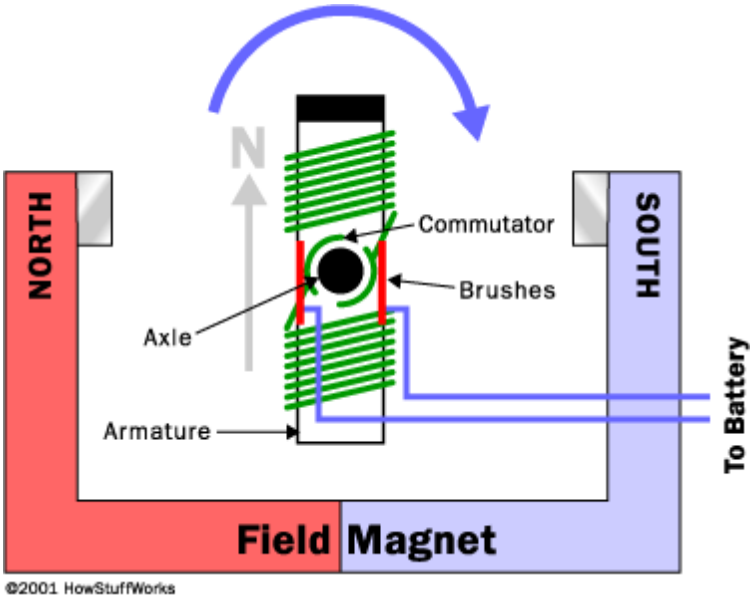
תיאור הדקי המפענח:

y1-4- לרגליים אילו מתחברות העמודות של המקלדת.
x1-4- לרגליים אילו מתחברות השורות למקלדת.
Kmb -רגל למניעת ריטוטים אשר הקבל מתחבר אליה.
Osc -רגל לחיבור הקבל אשר קובעת את תדר הסריקה.
Oe -רגל בקרה לאפשור הרכיב.
Da -רגל המודיעה כי נלחץ מקש במקלדת.
D0a-d0d -רגליים הנותנות את הצירוף הבינארי למקש שנקלט.

מנוע DC

למנוע החשמלי מהווה חלק גדול מהחיים שלנו .שימושים רבים מתבצעים בעזרת המנוע החשמלי. לדוגמה: מאוורר התנור, המיקסר, במקדחה החשמלית , וכן הלאה.

מבנה המנוע



איור 44

כל מנוע חשמלי כולל את החלקים הבאים :
א. הרוטור Armature
ב. קומוטטור
ג. מברשות Brushes
ד. ציר Axle
ה. שדה מגנטיMagnet Field
ו. אספקת מתח ממקור זרם ישר.

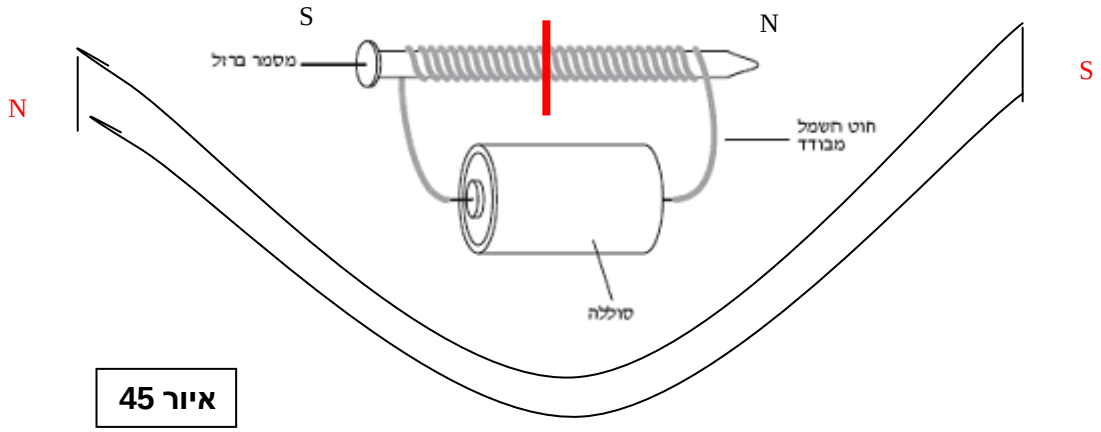
האחד הוא העוגן(או רוטור), זהו אלקטרו-מגנט. המגנט השני הוא מגנט שדה קבוע(סטטור):

1. סטטור :(Stator) מערכת סלילים המלופפים סביב ליבה פֶרוֹמגנטית (Ferromagnetic Core)המקובעת למקומה. הסטטור יכול להיות מורכב גם משני מגנטים רבי עוצמה. המגנטים מסודרים כך שקוטביהם (צפון ודרום) המופנים לכיוון הרוטור מנוגדים.
2. רוטור :(Rotor) ציר העובר בתוך הסטטור ועליו מלופפים שלושה סלילים. ציר זה חופשי להסתובב. כאשר זורם זרם חשמלי דרך הסלילים שברוטור ,נוצר שדה מגנטי סביבם (דרך הליבה). שדה מגנטי זה מפעיל כוח על הציר העובר דרכו, וזה מסתובב עקב המומנט(כח סיבובי). העברת זרם חשמלי מקוטע, בצורה מבוקרת, מאפשרת צירוף תנועות זוויתיות קטנות לסיבובים שלמים.

אופן פעולת המנוע

המנוע החשמלי מבצע המרת אנרגיה חשמלית לאנרגיה מכאנית.
כדי להבין כיצד פועל המנוע עלינו להכיר כלל אחד:
קטבים מגנטיים מנוגדים נמשכים זה לזה וקטבים מגנטיים זהים דוחים זה את זה.
בתוך המנוע אנו מנצלים את פעולת המשיכה והדחייה בין הקטבים כדי לגרום לסיבוב הרוטור.

כדי להבין את אופן פעולת המנוע אנו נדרשים להבין את אופן פעולת האלקטרומגנט .
האלקטרומגנט הוא מנוע חשמלי בסיסי , ולשם המחשה ניקח מסמר ונלפף 100 ליפופים על המסמר . נחבר למסמר סוללה ,הדבר יגרום לפעולת שדה מגנטי במסמר .



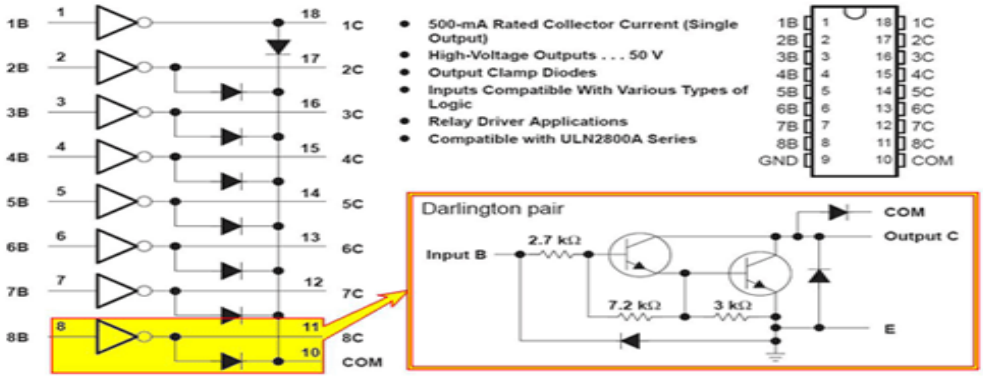
איור 45

נניח תחילה שהמסמר מונח על הקו האדום שבסרטוט. ברגע שמפעילים את הסוללה שמחוברת למסמר נוצר קוטביות מגנטית במסמר . הקוטב המסומן בצפון ידחה מקוטב הצפון של המגנט וימשך לקוטב המסומן בדרום, כתוצאה מכך המסמר יסתובב חצי סיבוב ואז ייעצר כמתואר בסרטוט. בשביל ליצור תנועה סיבובית נצטרך להחליף את קוטביות הסוללה ברגע שהמסמר נעצר.
הרוטור שווה ערך למסמר ושינוי הקטבים מתבצע על ידי המברשות והקומוטטור .

פעולת המנוע המודגם באיור מעלה, מבוססת על שימוש בשדה מגנטי שקטביו מסומנים באותיות S ו, N-שדה מגנטי זה אינו מסתובב ולכן הוא נקרא סטטור .בתוך השדה המגנטי נמצא חלק נע הנקרא רוטור או אוגן (Armature) .חיבור האוגן למתח חשמלי יגרום להפיכתו לאלקטרומגנט ולמשיכת הקוטב הצפוני אל הקוטב הדרומי של המסגרת ,התוצאה סיבוב ציר המנוע רבע סיבוב ימינה. על מנת לחבר בין את המסגרת המסתובבת למקור המתח יש לאפשר זרימת אלקטרונים ממוליך המחובר למקור המתח אל המסגרת הסובבת, בעיה זו נפתרת בעזרת טבעת החלקה חצויה המחוברת למסגרת ומסתובבת יחד אתה. לטבעת זו מוצמד זוג מברשות פחמים המחליק על הטבעת בזמן שזו מסתובבת. התוצאה במהלך סיבוב אחד שלם גורמת הטבעת לכך שהאלקטרומגנט פעם ימשך אל קוטבי השדה המגנטי ופעם ידחה ממנו בשל כוח הנגרם כאשר מוליך חוצה שדה מגנטי.

2803-ULN

זהו בעצם מערך של שמונה קבוצות טרנזיסטורים, המחוברים ביניהם בחיבור Darlington ומאפשרים לחבר למיקרו רכיבים הדורשים מתח וזרם גבוהים. מדובר ברכיבים כגון מנועים, ממסרים, דיודות פולטות אור וכד'. בכניסה לרכיב ניתן לחבר מערכות לוגיות, הפועלות ברמת מתח של 5 וולט (בדומה למיקרו-בקר שלנו). רכיב זה הינו מעגל דוחף שתפקידו להפעיל את הצרכנים ע"י הפעלת הממסר. לכול מוצא מחוברת דיודה פנימית, ועם חיבורה לVCC היא מגינה על מוצא הרכיב מפני מתחים גבוהים, הנגרמים מרכיבים השראתים כמו ממסרים, מנועים וכו'.
הדוחף מקבל את המידע מהתוכנה במיקרו (הכולל כיוון סיבוב המנוע, הפעלת המנוע, הדלקת הלדים, והפעלת הצופר).



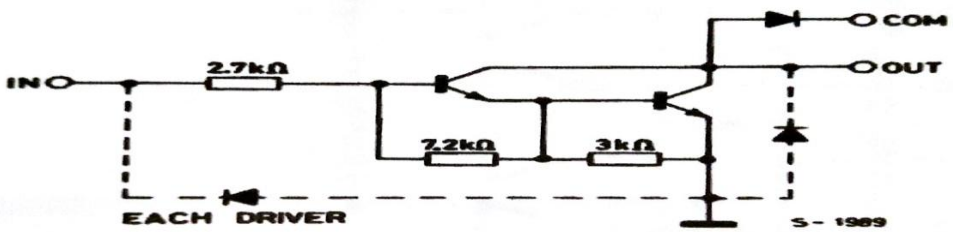
איור 46

תכונות הרכיב:
1.הגברת זרם ע"י חיבור דרלינגטון.
2.עובד ללא מתחי הזנה.
3.צריכת הספק נמוכה.
4.שליטה דיגיטלית על הפעלה של עומסים.
5.זרם מוצא עד 500 ma.
6.מתח מוצא עד 50V.
7.כולל דיודות להגנה על צרכנים.

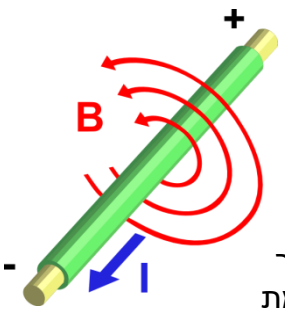
הרכיב בפרויקט:
תפקיד הרכיב בפרויקט הוא להפעיל עומסים (מנוע צד).

דרליגנטון

ULN2803: מורכב מ7 זוגות טרנזיסטורים- דרלינגטון NPN מסוג מתח גבוה, זרם גבוה.
לכל היחידות יש אמיטר משותף וקולקטור מוצא פתוח.
ל-ULN2803 יש אפשרות של פעולה ישירה עם CMOS\TTL הפועל במתחי מקור של 5V.



איור 47



איור 48

ממסר

בשביל להבין את פעולת הממסר נבין את עקרון האלקטרו מגנט שהתגלה על ידי האנס כריסטיאן ארסטד על פי עקרון זה נוצר שדה מגנטי סביב מוליך חשמלי כאשר במוליך זורם זרם.
כלומר כאשר יזרום זרם בתוך הסליל יוצר שדה מגנטי בכיוון אחד (לפי כלל ימין ניתן לגלות את כיונו).

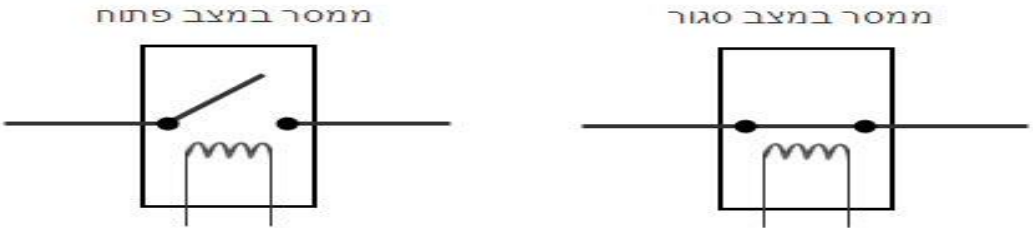
בממסרים השאיפה היא שזרם הפיקוד יהיה נמוך לכן מקובל ליצור סליל המאפשר שימוש במוליך ארוך מאוד. את הסליל יוצרים סביב ליבת ברזל המרכזת את עוצמת השדה המגנטי לנקודה אחת.

מבנה

ממסרים בנויים משני חלקים עיקרים :
1 סליל חשמלי - חוט חשמלי המלופף סביב ליבת ברז .
2. גוף ברזלי הנמשך לליבת הברזל כשר היא הופכת למגנטית.

עקרון פעולה

ממסר הוא רכיב בעל מפסק וסליל. כאשר זורם זרם בסליל, הסליל יוצר שדה מגנטי, דבר המוביל לתנועה המועברת למפסק, והמפסק משנה את מצבו למצב מופעל. יש חציצה בין חיבורי הסליל לחיבורי המפסק, כך שרכיב זה מאפשר למשל חיבור וניתוק של מעגלי מתח.



איור 49

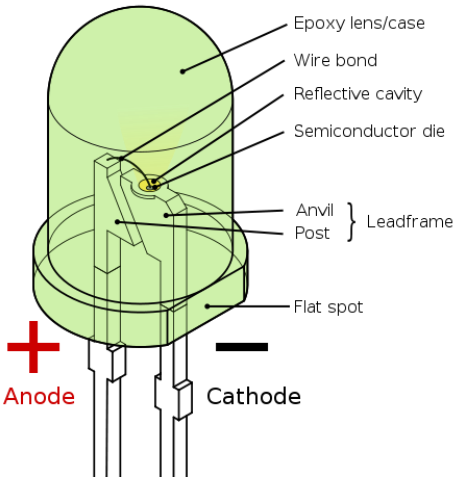
ממסר hjr 12cl:
הממסר שאנחנו שהשתמשנו בפרויקט מקיים את עקרונות אלה ונותן לנו את האפשרות להניע את הגלגלים קדימה ואחורה

איור 50

LED

לד הינו דיודה פולטת אור (Light emitting diode - LED) הינה התקן מוליך למחצה אשר פולט אור קוהרנטי בספקטרום צר.
ספקטרום האור נקבע בהתאם לתרכובת המוליך למחצה.
הלד הראשון התגלה בשנת 1907.
הלדים הראשונים החלו להימכר ב 1962 בצבע אדום , הצבעים ירוק צהוב כתום הגיעו מאוחר יותר.
הלד הכחול הופיעה ב 1989.

<u>עקרון הפעולה</u>
ה-LED הינה דיודה בעלת תכונות ייחודיות. בדומה לדיודה רגילה, היא מורכבת מחומר מוליך למחצה שעבר סימום על מנת ליצור בו צומת p-n. בדומה לדיודה סטנדרטית, הזרם החשמלי ב-LED יזרום במתח קדמי מצד ה-p לצד ה-n, אך לא יזרום בכיוון ההפוך כאשר הדיודה במתח אחורי. נושאי מטען (אלקטרונים וחורים) יסחפו אל תוך הצומת כתוצאה מהמתח הקדמי שמופעל על הדיודה. ברגע שאלקטרון יפגוש חור, הוא יאבד מהאנרגיה שלו וייפול למצב אנרגטי נמוך יותר.
המיוחד ב־LED לעומת דיודה סטנדרטית, הוא שישנו פער אנרגטי ישיר בין נקודת השיא האנרגטית בפס הערכיות לבין השפל האנרגטי בפס ההולכה של המוליך למחצה. משמעות הפער האנרגטי הישיר היא שהאלקטרון יכול ליפול מפס ההולכה לפס הערכיות תוך כדי שימור התנע שלו ופליטת פוטון בעל אנרגיה המתאימה לפער אנרגיה זה - כלומר הדיודה תפלוט אור כשהיא נמצאת במתח קדמי.
תרכובת המוליך למחצה הראשונה לייצור LED הייתה גאליום-ארסן, שסיפקה אור סביב התחום האדום.
כיום, עם ההתפתחות הטכנולוגית ניתן לייצר LED שיפלטו אור במגוון צבעים בתחום הנראה, בתחום התת-אדום ובתחום העל-סגול.

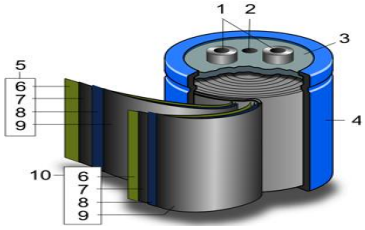


איור 51

איור 52

קבל -Capacitor

קבל (Capacitor) הוא הוא רכיב חשמלי שמטרתו לאגור מטען חשמלי. הקבל לרב בנוי משני משטחים מוליכים המופרדים על ידי חומר מבודד. מתח המוזן לשני המוליכים של קבל גורמים להיווצרות שדה חשמלי סטטי.



איור 53

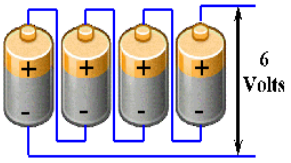
קיבול של קבל נמדד ביחידות Farad (F) והקבל מסומן בשרטוטים חשמליים באות C .
קבלים יכולים להגיע במגוון צורות ומגוון דרכים: קבלים בודדים ונפרדים וקבלים המובנים בתוך מעגלים משולבים ועוד.

בפרויקט שלנו השתמשנו בקבלים בשביל לאגור אנרגיה ולשמש כמקור מתח. בנוסף שתמשנו בקבלים בשביל לסנן רעשים .

בטריות בחיבור טורי

כפי שהוסבר בתחילת הפרויקט , על המכונית הדמה שמשמשת אותנו בפרויקט . אנו מפעילים אותה ואת הבקר 8051 באמצאות בטריות של 1.5 וולט כל אחת בחיבור טורי .

בטריות בחיבור טורי מהווים בטרייה אחת כללית שווה לסכום כל המתחים של הבטריות .
בגלל שהבטריות זהות אז הנוסחא תהיה : מספר בטריות × 1.5
לדוגמא :



איור 54

בפרויקט שלנו השתמשנו ב 5 בטריות לכן המתח שסיפקנו הוא 7.5 וולט .
חשוב לציין שזהו מתח מספיק גם מבחינת בקר וגם מבחינת המכונית .
בקר – הבקר מזהה 1 לוגי כמתחים שמעל 5 וולט לכן הדבר לא גבולי ותקין לפעולת המערכת .
מכונית –כאשר נספק 7.5 וולט מהירות המכונית לא מהירה מידיי ונותנת אפשרות הבחנה במתרחש.

ORCAD

ORCAD היא סביבת פיתוח לתכנון מעגלים חשמליים. תוכנה זו פשוטה ונוחה מאוד לשימוש משום שיש רכיבים מוכנים מראש וכמו כן ניתן ליצור רכיבים חדשים לפי בקשת המשתמש. בעזרת סביבת פיתוח זו ניתן לתכנן מעגלים חשמליים מורכבים כאשר ניתן לפשט אותם ע"י חלוקה לכמה שרטוטים.
בפרויקט זה השתמשנו בתוכנת ה ORCAD כדי לשרטט ולתכנן את המעגל החשמלי של הכרטיס הנייד הכולל בו רכיבים שונים. את כל הרכיבים שבתוכנה יצרתי בעצמי מה שנתן לי את היכולת לשלוט בצורתם החיצונית ולקבוע את הרגליים שלהם כדי שיתאפשר שרטוט קל מדויק ונוח יותר.

שפת C

שפת C היא שפת תכנות עלית הכוללת מנגנוני בקרת זרימה ומבני נתונים פשוטים, וכמו שפת סף ומאפשרת ניצול מרבי של המחשב. שפת C היא אחת השפות היעילות והמהירות בתעשייה, ומשמשת כיום בעיקר לכתיבת מערכות בהן זמן הביצוע הוא גורם קריטי, או כאלה שדורשות אינטראקציה עם מערכות חומרה. רוב מערכות ההפעלה הנפוצות כתובות בעיקר בשפת C, וכן רוב הדפדפנים הנפוצים.

שפת C היא אחת משפות התכנות הנפוצות בעולם. קיימים מהדרים לשפת C עבור כמעט כל סוגי המחשבים ומערכות ההפעלה. שפת C השפיעה על שפות תכנות רבות, ובמיוחד על C++, שנכתבה בתחילה כקדם-מעבד עבור C, וכיום היא כמעט מכילה את שפת C. רוב שפות התכנות הנפוצות כיום כוללות מאפיינים תחביריים המזכירים את שפת C.

מאפיינים:
עיקרון מנחה בשפה הוא עקרון היעילות, ובכך היא מצטיינת. כל פעולה בשפה מתבטאת בפעולה בודדת או במספר מועט של פעולות בשפת סף. משום כך אין בה סוגי מבנים מורכבים כגון מחרוזות או בדיקות סמויות בגישה למערכים ומערכת הטיפוסים שלה לא בטוחה. מסיבה זו היא גם כוללת לא מעט אפשרויות לשגיאות שהמהדר (Compiler) לא יאתר, כגון פנייה לאיבר שלא קיים במערך או פנייה לערך לא חוקי בזיכרון וכדומה. הנחת העבודה של מתכנני השפה (והתקנים המאוחרים יותר) היא כי "המתכנת מבין מה הוא עושה".
סביבות עבודה לשפת C כיום כוללות מהדר, כלי לניפוי שגיאות (מציאת תקלות, באגים), וכן אפשרות לערוך את הקוד בחבילת תוכנה אחת.

מהדר-COMPILER:
לפני תרגום התוכנית לשפת מכונה, תוכנית הנקראת קדם- מעבד יוצרת קובץ חדש הכולל בנקודה בה מוכרזת ההכללה את תוכן הקובץ המוכלל, ואז מועבר הקובץ להידור לשפת מכונה.

שפת C היא שפה קטנה למדי. היא אינה מכילה פקודות הקושרות אותה למערכת הפעלה מסוימת או לתחום תוכנה מסוים. מסיבה זו, פקודות ספציפיות שכאלה (למשל לקלט ופלט, לחישובים מתמטיים, לעבודה עם מחרוזות ולהקצאות זיכרון דינמיות) נמצאות בספריות סטנדרטיות (לדוגמא :stdio.h), שאינן חלק מתחביר השפה אך עונות לקונבנציה מוכרת וקבועה ומהוות חלק מהתקן הרשמי שלה. ספריות אלה מסופקות עם המהדר עבור מערכת הפעלה וחומרה מסוימות והתוכנה מקושרת (linked) עימן על פי צורך והגדרת כותב הקוד.

טכנולוגיית הקינקט KINECT

קינקט Kinect הוא בקר משחקים שמיוצר על ידי חברת מיקרוסופט עבור קונסולות משחקי הוידאוXbox One, Xbox 360 ולמחשבי ווינדוס .טכנולוגיית הקינקט מבוססת מצלמה היקפית ומאפשרת שליטה ללא בקר אלא על ידי קליטה של תנועות הגוף וזיהוי קול . הדור הראשון של הקינקט הוצג לראשונה בחודש נובמבר 2010.

מיקרוסופט התירה לפרסום ערכת פיתוח של תוכנת קינקט עבור Windows 7 ב-16 ביוני 2011. ערכת הפיתוח SDK נועדה לאפשר למפתחים לכתוב יישומי קינקט בשפות התכנות,++C ,#C אוNET, Visual Basic .

טכנולוגיה

קינקט נבנתה על ידי טכנולוגית תוכנה שפותחה ב "רר" (Rare) , חברת בת של מיקרוסופט גיים סטודיו בבעלות מיקרוסופט. טכנולוגית המצלמה של קינטקט, פותחה על ידי היזם הישראליPrimeSense . המערכת שפיתח שמסוגלת לפרש תנועות גוף ספציפיות ומאפשרת שליטה ללא מגע ידיים על התקנים אלקטרוניים ,אפשרות זאת מתבצעת באמצעות מקרן אינפרא אדום ,מצלמה ושבב אלקטרוני מיוחד שמאפשר לעקוב אחר התנועה של אובייקטים ואנשים בשלושה ממדים .מערכת סורק 3D זו נקראת" *קוד אור* " (Light Coding) משתמשת בטכנולוגיה המבוססת על גירסת 3Dלשחזור תמונה.

חיישן זיהוי קול ופנים

חיישן זיהוי הקול והפנים של קינקט מחובר לבסיס קטן בעל ציר ממונע וממוקם מעל או מתחת לתצוגת הווידאו. המכשיר כולל מצלמת RGB, חיישני עומק ותנועה ומיקרופון רב מערכתי הכלולים בתוכנה .כל אלה מספקים לכידת תנועת תלת ממדית של גוף מלא, יכולת זיהוי קול וזיהוי פנים. המערכת מסוגלת לזהות את הפנים והקול של העומד מולה ולהתייחס אליו בהתאם .

חיישן זיהוי עומק

חיישן זיהוי העומק מורכב משילוב של קרן לייזר אינפרא אדום וחיישן CMOS .החיישן לוכד את נתוני הווידאו בתלת ממד בכל תנאי אור וסביבה. טווח הזיהוי של החיישן מתכוונן ומאפשר כיול באופן אוטומטי עם הסביבה הפיזית של השחקן.

חיישנים נוספים

בנוסף למצלמה ולחיישן זיהוי עומק, מיקרוסופט ציידה את הקינקט במספר חיישנים:

1. חיישן תנועה, השומר שהקינקט לא יזוז בזמן משחק.
2. מנוע המזיז את הקינקט למעלה ולמטה, על מנת לאפשר למכשיר "לראות" מגובה של ילד קטן ועד לגובה לאדם גבוה.
3. חיישנים הבודקים את המשטח עליו נמצא הקינקט, כדי לוודא שהוא נמצא במקום יציב.
4. מערך של 4 מיקרופונים לשמע חלק וניטרול רעשי סביבה.

מצלמת KINECT 2 לעומת KINECT 1

בפרויקט שלנו העדפנו להשתמש במצלמת הקינקט החדישה יותר . היתרונות שלה על פני הדגם הישן יותר מוצגים בטבלה :

Feature	Kinect for Windows 1	Kinect for Windows 2
Color Camera	640 x 480 @30 fps	1920 x 1080 @30 fps
Depth Camera	320 x 240	512 x 424
Max Depth Distance	~4.5 M	~4.5 M
Min Depth Distance	40 cm in near mode	50 cm
Horizontal Field of View	57 degrees	70 degrees
Vertical Field of View	43 degrees	60 degrees
Tilt Motor	yes	no
Skeleton Joints Defined	20 joints	26 joints
Full Skeletons Tracked	2	6
USB Standard	2.0	3.0
Supported OS	Win 7, Win 8	Win 8
Price	$299	TBD

טבלה 13

KINECT 2

איור 56

KINECT 1

איור 55

SDK -Software development kit

ערכת פיתוח תוכנה באנגלית Software development kitנקרא בקיצור SDK או devkit .
בתוך התוכנה יש סט של כלים לפיתוח תוכנה והעשרת יכולות תוכנה בצורה נוחה .
ברוב המקרים משתמשים ב SDKבשביל יישומים .

ערכות רבות ניתנות בחינם על מנת לעודד מפתחים להשתמש במערכת או בשפה מסוימת, באופן שמשמש ככלי שיווק או כחלק ממודל עסקי המבוסס על מידע. לדוגמה, בפרויקט שלנו השתמשנו במצלמת KINECT 2 וביכולות לזיהוי פנים שמציע הSDK .

מכיוון שמפתחי ישומים לטלפונים ניידים (אפליקציות מובייל) קורסים תחת הנטל הפיתוחי, ומכיוון שרבים מהם מחפשים אותם כלים להעשרת היישומים שלהם, התפתחה לה תעשייה רחבה לערכות פיתוח תוכנה, והיום ניתן למצוא אלפי חברות ,SDK רבות מהן מציעות יכולות פרימיום בעלויות לא נמוכות, אך לא מעט מציעות ערכות הניתנות לשימוש בחינם. מפתחים מטמיעים ערכות פיתוח בקוד המקור שלהם וזאת על מנת להוסיף יכולות מתקדמות במהירות וללא פנייה למשאבי הפיתוח המוגבלים לרוב.

התפתחות תעשיית ערכות הפיתוח החיצוניות מאפשרת למפתחי אפליקציות להעלות משמעותית את איכות היישומים שהם מפתחים בזמן קצר. מפתחי ערכות הפיתוח, מצידם, ממשיכים להוסיף יכולות מתקדמות (על מנת למשוך מפתחים לעשות שימוש בערכות שלהם) ובכך תורמים רבות לקדמה הטכנולוגית ולהתקדמות המטאורית בתעשיית פיתוח התוכנה.

מתאם KINECT למחשב

את מצלמת ה KINECT חיברנו למתאם שיוכל לקשר בינה לבין המחשב . המתאם מתאים למצלמות KINECT 2 ולמערכת הפעלה WINDOWS. המתאם מחובר למחשב דרך חיבור USB.

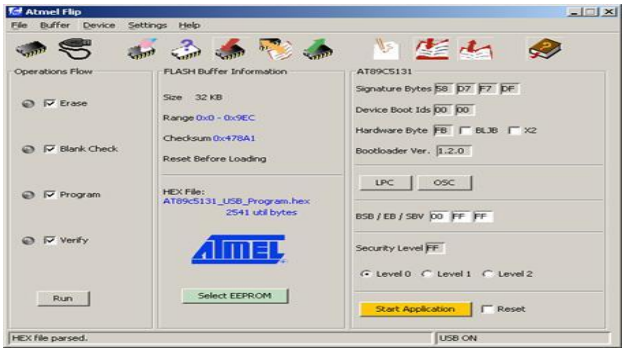
איור 57

SESC51

סביבת העבודה SES היא הסביבה שבה כתבנו את הקוד בשפת C בסביבה זו ניתן למצוא שגיאות בקוד , להדר ואף ליצור קובץ HEX שאותו נוכל לצרוב לבקר שלנו.
כיום סביבת פיתוח משולבת כוללת לרוב עורך קוד מקור, מהדר ו/או מפרש, כלי בניה ממוכנים ומנפה (דיבאגר). לעתים, הסביבה כוללת כלים לבקרת תצורה וניהול גרסאות וכן כלים המקלים על בניית יישומים בעלי ממשק משתמש גרפי. סביבות מודרניות כוללת גם כלים המסייעים בתכנות מונחה-עצמים כגון סייר מחלקות, בוחן עצמים ותרשים של עץ המחלקות.
סביבת הפיתוח המשולבת מקלה על ההליך התכנותי, והופכת אותו לפחות מסורבל.

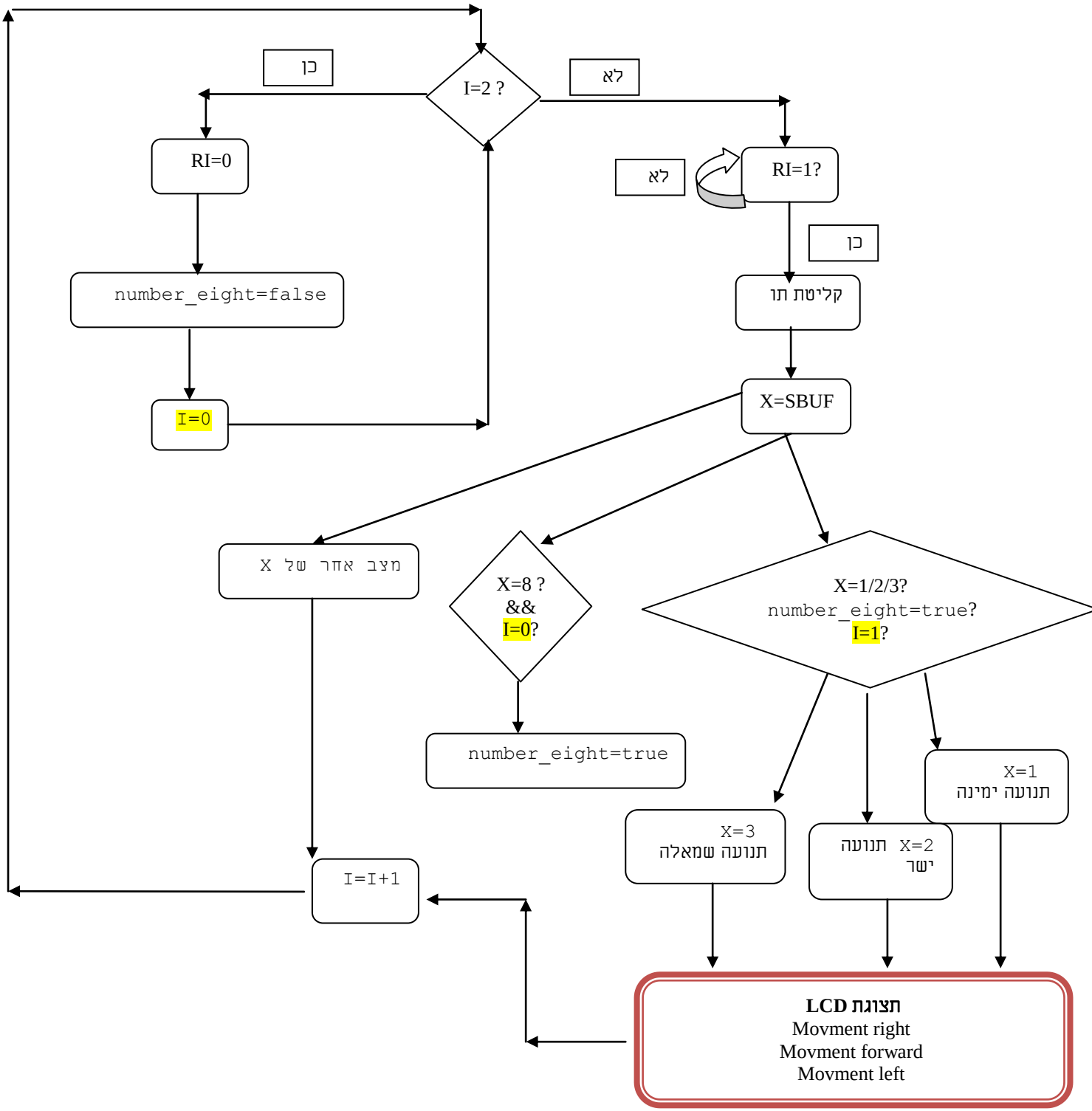
FLIP

בפרויקט שלנו נשתמש בתוכנה ייעודית בשם Flip Flexible In-system Programmer שמספק לנו יצרן הרכיב לשם צריבת הקובץ AT89c5131_USB_Program.hex בזיכרון ה-Flash של המיקרו-בקר .



איור 58

תרשים זרימה לצד הבקר



תוכנה לצד הבקר

```
#include <at89x52.h>
```
הכלת ספריה לבקר על מנת שנוכל לתקשר איתו

```
#include <string.h>
```
הכלת ספריית המחרוזות

```
#include <stdbool.h>
```
הכלת הספרייה הבוליאנית

```
#define display_on 0x01
```
הצהרה על משתנה קבוע שמקבל ערך קבוע0x01

```
#define entry_mode 0x38
```
הצהרה על משתנה קבוע שמקבל ערך קבוע0x38

```
#define lcd_first_line 0x80
```
הצהרה על משתנה קבוע שמקבל ערך קבוע0x80

```
#define lcd_second_line 0xA8
```
הצהרה על משתנה קבוע שמקבל ערך קבוע0xA8

```
#define code "123"
```
הצהרה על משתנה קבוע שמקבל ערך קבוע"123"

```
#define clear_lcd 0x01
```
הצהרה על משתנה קבוע שמקבל ערך קבוע0x01

```
unsigned char key;
```
הצהרה על משתנה מסוג תו לא מסומן

```
unsigned char tav;
```
הצהרה על משתנה מסוג תו לא מסומן

```
bool number_eight ;
```
הצהרה על משתנה בוליאני

```
char x;
```
הצהרה על משתנה מסוג תו

```
void ithul(void);
```
הצהרה על פונקציה בשם איתחול שלא מחזירה כלום ולא מקבלת כלום

```
void print_lcd(int line,char*ptr);
```
הצהרה על פונקציה שמקבלת שני משתנים אחד מטיפוס שלם והשני מטיפוס תו, המשתנה הראשון הינו שורה בצג והשני הוא מה שנרצה לכתוב לצג, פונקציה זו אינה מחזירה דבר.

```
void init_lcd ();
```
הצהרה על פונקציה שלא מחזירה כלום ולא מקבלת שום ערך אליה.

```
void send_cmd_lcd(char lcd_command);
```
הצהרה על פונקציה שמקבלת משתנה מטיפוס תו ולא מחזירה דבר.

```
void send_data_lcd(char lcd_data);
```
הצהרה על פונקציה שמקבלת משתנה מטיפוס תו ולא מחזירה דבר.

```
void delay(long del);
```
הצהרה על פונקציה שמקבלת משתנה מטיפוס לונג, מטרתה של פונקציה זאת היא לבצע השהייה לכת נשתמש במשתנה מסוג לונג שיכול להכיל ערכים גדולים, פונקציה זו אינה מחזירה דבר.

```
char get_key() ;
```
הצהרה על פונקציה שתחזיר ערך מטיפוס תו ולא מקבלת כלום.

```
void   LCD_res(void)
{
      send_cmd_lcd(0x30) ;
```
פונקציה שנותנת פקודה לצג ומציבה בו את הערך 30 בבסיס הקסאדצימאלי

```
      send_cmd_lcd(0x30) ;
```
פונקציה שנותנת פקודה לצג ומציבה בו את הערך 30 בבסיס הקסאדצימלי

```
      send_cmd_lcd(0x30) ;
```
פונקציה שנותנת פקודה לצג ומציבה בו את הערך 30 בבסיס הקסאדצימלי

```
      send_cmd_lcd(0x38) ;
```
פונקציה שנותנת פקודה לצג ומציבה בו את הערך 38 בבסיס הקסאדצימלי

```
      send_cmd_lcd(0x08) ;
```
פונקציה שנותנת פקודה לצג ומציבה בו את הערך 8 בבסיס הקסאדצימלי

```
      send_cmd_lcd(0x01) ;
```
פונקציה שנותנת פקודה לצג ומציבה בו את הערך 1 בבסיס הקסאדצימלי

```
      send_cmd_lcd(0x0C) ;
```
פונקציה שנותנת פקודה לצג ומציבה בו את הערך סי בבסיס הקסאדצימלי

```
      send_cmd_lcd(0x06) ;
```
פונקציה שנותנת פקודה לצג ומציבה בו את הערך 6 בבסיס הקסאדצימלי

```
}
```

```
void ithul(void)
 {
           int BRL,BDRCON;
           IE=0x90;   //1011 0000
           TR1=0;     //Stop Timer 1
           TH1=0x80;  //Load High Byte of Timer 1 for 600 Baud Rate
           TL1=0x00;  //Load Low  Byte of Timer 1
           TMOD=0x20; //0010 0000 Timer 1= Gate=0 ;C/T~=0 ; MODE 2-8
BIT AUTO reload
           SCON=0xDE; //1111 1110 9 bit UART TB8=1;RB8=1;T1=1; R1=0
           BRL=152;   //600 Baud Rate
           BDRCON=0x16;
           TR1=1  ;    // Start Timer 1
{
```

מטרתה של פונקציה זאת היא לבצע השהייה, אשר תסייע למסך לאפשר לנו לראות מספיק זמן את ההודעות אשר יוצגו עליו

```
void delay(long del )
{
          long i;
          for(i=0;i<del;++i);
```

לולאת פור, תרוץ מ0 ועד ערך המשתנה דל

```
}
```

```
char get_key(void)
{
   char temp;
   int i;
   while (P1_4==0) ;
```

הצהרה על לולאה שתתקיים בתנאי שהסיבית הרביעית בפורט אחד שווה ל0 לוגי

```
   temp = P1 & 0x0F;
```

פעולת שרשור סיביות בין פורט אחד לבין הנתון אפס אף כלומר פעולה זו תגרום לסיבית ארבע עד סיבית שבע לקבל אחד ולסיבית אפס עד שלוש לקבל אפס.

```
   delay(1000);
```

קריאה לפונקציה דיליי שתבצע השהייה עם שליחת נתון ששווה לאלף כי אנו רוצים השהייה בעלת מספיק זמן

```
   while (P1_4==1);
```

הצהרה על לולאה שתתקיים בתנאי שהסיבית הרביעית בפורט אחד שווה ל1 לוגי, במידה ולא נצא מהלולאה ונמשיך לפקודות הבאות

```
   switch(temp)
```

שימוש בפקודה שבודקת האם המשתנה טמפ שווה לאחד מהמקרים הבאים, במידה וכן היא תיכנס אל קייס

```
   {
    case 0:
                temp = '1';  init_lcd();print_lcd(2,"1");
for(i=0; i<20; i){++delay(1000);delay(1000);delay(1000);}
                break ;
    case 1 :
                temp = '2'; init_lcd();print_lcd(2,"4");
for(i=0; i<20; i){(++delay(1000);delay(1000);delay(1000);}

                 break ;
```

```
    case 2:
                temp = '3'; init_lcd();print_lcd(2,"7");
for(i=0; i<20; i){(++delay(1000);delay(1000);delay(1000);}

               break;
    case 3:
               temp = '4'; init_lcd();print_lcd(2,"*");
for(i=0; i<20; i){(++delay(1000);delay(1000);delay(1000);}

               break;

    case 4 :
                temp = '5'; init_lcd();print_lcd(2,"2");
for(i=0; i<20; i){(++delay(1000);delay(1000);delay(1000);}

                 break;


    case 5 :
                temp = '6'; init_lcd();print_lcd(2,"5");
for(i=0; i<20; i){(++delay(1000);delay(1000);delay(1000);}

                break;
    case 6:
                temp = '7'; init_lcd();print_lcd(2,"8");
for(i=0; i<20; i){(++delay(1000);delay(1000);delay(1000);}

                break;
    case 7:
                temp = '8'; init_lcd();print_lcd(2,"0");
for(i=0; i<20; i){(++delay(1000);delay(1000);delay(1000);}

               break;
    case 8:
                temp = '9'; init_lcd();print_lcd(2,"3");
for(i=0; i<20; i){(++delay(1000);delay(1000);delay(1000);}

               break;
    case 9:
                temp = '*'; init_lcd();print_lcd(2,"6");
for(i=0; i<20; i){(++delay(1000);delay(1000);delay(1000);}

                break;
    case 10:
                temp = '0'; init_lcd();print_lcd(2,"9");
for(i=0; i<20; i){(++delay(1000);delay(1000);delay(1000);}

              break;
    case 11:
                temp = '#'; init_lcd();print_lcd(2,"#");
for(i=0; i<20; i){(++delay(1000);delay(1000);delay(1000);}

                 break;
    defualt:
   break  ;
}
  return temp;
}
```

```
void print_lcd(int line,char *ptr)   שימוש בפונקציה שהצהרנו עליה מקודם

{
 int i;   הצהרה על משתנה מטיפוס שלם
switch(line)     שימוש בפקודה שבודקת האם המשתנה ליין שווה לאחד מהמקרים הבאים, במידה וכן
היא תיכנס אל קייס
{
case 1:   send_cmd_lcd(lcd_first_line);    במידה וליין שווה ל 1 ניכנס ונקרא לפונקציה
עם שליחת משתנה, נותנים פקודה לבקר להעביר את הסמן לשורה הראשונה בצג
          for(i=0;* ptr+i;'0\'=!(i++))    לולאת פור שאליה ניכנס במידה וסכום
המשתנים פיטיאר ואיי שונים מאפס.
          send_data_lcd(*(ptr+i       קריאה לפונקציה עם שליחת משתנה
));
          break;
case 2:   send_cmd_lcd(lcd_second_line);     במידה וליין שווה ל2 נכנס ונקרא
לפוקנציה עם שליחת משתנה, נותנים פקודה לבקר להעביר את הסמן לשורה השנייה בצג
          for(i=0;* ptr+i;'0\'=!(i++));     לולאת פור שאליה ניכנס במידה וסכום
המשתנים פיטיאר ואיי שונים מאפס.
          send_data_lcd(*(ptr+i));    קריאה לפונקציה עם שליחת משתנה
          break;
}
}
```

```
void init_lcd()
{
send_cmd_lcd(display_on);   //start lcd
send_cmd_lcd(clear_lcd);    //clear lcd
send_cmd_lcd(entry_mode);   //LCD ready TO Recieve Data
}
```

```
void send_cmd_lcd(char lcd_command)
{
 P0=lcd_command;
 P2_6=0;         //P4_1=0-Control Register
 P2_7=1;
 delay(200);
 P2_7=0;         // pulse Down
 delay(200);
}
```

```
void send_data_lcd(char lcd_data)
{
 P0=lcd_data;
 P2_6=1;        //P4_1=1-data Register
 P2_7=1;
 delay(200);
 P2_7=0;         // pulse Down
 delay(200);
}
```

```
void main(void)
{
   char x;
   bool number_eight=false;
   unsigned char in_char;
   int flag=1,j,i;
   char s[4]="",p;
   ithul();
   LCD_res();


while (1)
{
      for (i=1 ; i<=2 ; i(++))
      {
            while (RI==0);
            x= SBUF;
            switch(x)
            {
                        case 1  :
                                    if(number_eight==true)
                                    {
                                          P1_7=1 ;
                                          P1_6=0;
                                          P2_0=0 ;
                                          P2_1=1;
                                          init_lcd;()
                                          print_lcd(1,"movement");
                                          print_lcd(2,"right");
                                          for(i=0; i<20; i(++))

            }delay(1000);delay(1000);delay(1000{;(
                                          p=get_key;()
                              }
                              break;
                        case 2 :
                                    if(number_eight==true)
                                    {
                                          P1_7=1 ;
                                          P1_6=0;
                                          P2_0=0 ;
                                          P2_1=1;
                                          P2_2=0 ;
                                          P2_3=1;
                                          P2_4=1 ;
```

```
                                        P2_5=0;
                                        init_lcd;()
                                        print_lcd(1,"movement");
                                        print_lcd(2,"forward");
                                        for(i=0; i<20; i(++))

}delay(1000);delay(1000);delay(1000{;(
                                        p=get_key;()
                                  }
                            break;

                      case 3 :
                                  if(number_eight==true)
                                  {

                                        P2_2=0 ;
                                        P2_3=1;
                                        P2_4=1;
                                        P2_5=0;
                                        init_lcd;()
                                        print_lcd(1,"movement");

                                        print_lcd(2,"left");
                                        for(i=0; i<20; i(++))

}delay(1000);delay(1000);delay(1000{;(
                                        p=get_key;()
                                        )
                             break;

                      case 8  :
                                        number_eight= true ;

                       }
                      RI=0;
                }
                number_eight= false ;
          }
 }
```

הסבר התוכנית:

התוכנית מכילה 7 פונקציות:

1) Void ithul(void) תפקיד פונקציה זו הוא להגדיר את חיבורי התצוגה.
2) Void print_lcd(int line, char*ptr) תפקיד של פונקציה זו היא לשלוח כתיבה על התצוגה.
3) Void init_lcd() תפקיד פונקציה זו היא לאפס את התצוגה.
4) Void_send_cmd_lcd(char lcd_command) פונקציה זו מאפשרת את התצוגה.
5) Void send_data_lcd(char lcd_data) תפקיד פונקציה זו לשלוח את המידע לתצוגה (ביט אחרי ביט).
6) Void delay(long del) פונקציה זו מייצרת השהייה.
7) Char get_key() פונקציה זו קולטת איזה מקש נלחץ על לוח המקשים.

תוכנה לצד המחשב

```
namespace Microsoft.Samples.Kinect.HDFaceBasics
{
using System;
using System.ComponentModel;
using System.Windows;
using System.Windows.Media;
using System.Windows.Media.Media3D;
using System.IO.Ports; //sheli.
using Microsoft.Kinect;
using Microsoft.Kinect.Face;



/// <summary>
/// Main Window
/// </summary>
public partial class MainWindow : Window, INotifyPropertyChanged,
IDisposable
{
/// <summary>
/// Currently used KinectSensor
/// </summary>
private KinectSensor sensor = null;

/// <summary>
/// Body frame source to get a BodyFrameReader
/// </summary>
private BodyFrameSource bodySource = null;

/// <summary>
/// Body frame reader to get body frames
/// </summary>
private BodyFrameReader bodyReader = null;

/// <summary>
/// HighDefinitionFaceFrameSource to get a reader and a builder from.
/// Also to set the currently tracked user id to get High Definition Face
Frames of
/// </summary>
private HighDefinitionFaceFrameSource highDefinitionFaceFrameSource
= null;

/// <summary>
/// HighDefinitionFaceFrameReader to read HighDefinitionFaceFrame to
get FaceAlignment
/// </summary>
```

```
private HighDefinitionFaceFrameReader highDefinitionFaceFrameReader
= null;

/// <summary>
/// FaceAlignment is the result of tracking a face, it has face animations
location and orientation
/// </summary>
private FaceAlignment currentFaceAlignment = null;

/// <summary>
/// FaceModel is a result of capturing a face
/// </summary>
private FaceModel currentFaceModel = null;

/// <summary>
/// FaceModelBuilder is used to produce a FaceModel
/// </summary>
private FaceModelBuilder faceModelBuilder = null;

/// <summary>
/// The currently tracked body
/// </summary>
private Body currentTrackedBody = null;

/// <summary>
/// The currently tracked body
/// </summary>
private ulong currentTrackingId = 0;

/// <summary>
/// Gets or sets the current tracked user id
/// </summary>
private string currentBuilderStatus = string.Empty;

/// <summary>
/// Gets or sets the current status text to display
/// </summary>
private string statusText = "Ready To Start Capture";

/// <summary>
/// Initializes a new instance of the MainWindow class.
/// </summary>
public MainWindow()
{
this.InitializeComponent();
this.DataContext = this;
}
```

```
/// <summary>
/// INotifyPropertyChangedPropertyChanged event to allow window
controls to bind to changeable data
/// </summary>
public event PropertyChangedEventHandler PropertyChanged;

/// <summary>
/// Gets or sets the current status text to display
/// </summary>
public string StatusText
{
get
{
return this.statusText;
}

set
{
if (this.statusText != value)
{
this.statusText = value;

// notify any bound elements that the text has changed
if (this.PropertyChanged != null)
{
this.PropertyChanged(this, new
PropertyChangedEventArgs("StatusText"));
}
}
}
}

/// <summary>
/// Gets or sets the current tracked user id
/// </summary>
private ulong CurrentTrackingId
{
get
{
return this.currentTrackingId;
}

set
{
this.currentTrackingId = value;

this.StatusText = this.MakeStatusText();
}
```

```
}

/// <summary>
/// Gets or sets the current Face Builder instructions to user
/// </summary>
private string CurrentBuilderStatus
{
get
{
return this.currentBuilderStatus;
}

set
{
this.currentBuilderStatus = value;

this.StatusText = this.MakeStatusText();
}
}

/// <summary>
/// Called when disposed of
/// </summary>
public void Dispose()
{
this.Dispose(true);
GC.SuppressFinalize(this);
}

/// <summary>
/// Dispose based on whether or not managed or native resources should
be freed
/// </summary>
/// <param name="disposing">Set to true to free both native and managed
resources, false otherwise</param>
protected virtual void Dispose(bool disposing)
{
if (disposing)
{
if (this.currentFaceModel != null)
{
this.currentFaceModel.Dispose();
this.currentFaceModel = null;
}
}
}

/// <summary>
```

```
/// Returns the length of a vector from origin
/// </summary>
/// <param name="point">Point in space to find it's distance from
origin</param>
/// <returns>Distance from origin</returns>
private static double VectorLength(CameraSpacePoint point)
{
var result = Math.Pow(point.X, 2) + Math.Pow(point.Y, 2) +
Math.Pow(point.Z, 2);

result = Math.Sqrt(result);

return result;
}

/// <summary>
/// Finds the closest body from the sensor if any
/// </summary>
/// <param name="bodyFrame">A body frame</param>
/// <returns>Closest body, null of none</returns>
private static Body FindClosestBody(BodyFrame bodyFrame)
{
Body result = null;
double closestBodyDistance = double.MaxValue;

Body[] bodies = new Body[bodyFrame.BodyCount];
bodyFrame.GetAndRefreshBodyData(bodies);

foreach (var body in bodies)
{
if (body.IsTracked)
{
var currentLocation = body.Joints[JointType.SpineBase].Position;

var currentDistance = VectorLength(currentLocation);

if (result == null || currentDistance < closestBodyDistance)
{
result = body;
closestBodyDistance = currentDistance;
}
}
}

return result;
}

/// <summary>
```

```
/// Find if there is a body tracked with the given trackingId
/// </summary>
/// <param name="bodyFrame">A body frame</param>
/// <param name="trackingId">The tracking Id</param>
/// <returns>The body object, null of none</returns>
private static Body FindBodyWithTrackingId(BodyFrame bodyFrame, ulong
trackingId)
{
Body result = null;

Body[] bodies = new Body[bodyFrame.BodyCount];
bodyFrame.GetAndRefreshBodyData(bodies);

foreach (var body in bodies)
{
if (body.IsTracked)
{
if (body.TrackingId == trackingId)
{
result = body;
break;
}
}
}

return result;
}

/// <summary>
/// Gets the current collection status
/// </summary>
/// <param name="status">Status value</param>
/// <returns>Status value as text</returns>
private static string
GetCollectionStatusText(FaceModelBuilderCollectionStatus status)
{
string res = string.Empty;

if ((status & FaceModelBuilderCollectionStatus.FrontViewFramesNeeded)
!= 0)
{
res = "FrontViewFramesNeeded";
return res;
}

if ((status & FaceModelBuilderCollectionStatus.LeftViewsNeeded) != 0)
{
res = "LeftViewsNeeded";
```

```
return res;
}

if ((status & FaceModelBuilderCollectionStatus.RightViewsNeeded) != 0)
{
res = "RightViewsNeeded";
return res;
}

if ((status & FaceModelBuilderCollectionStatus.TiltedUpViewsNeeded) !=
0)
{
res = "TiltedUpViewsNeeded";
return res;
}

if ((status & FaceModelBuilderCollectionStatus.Complete) != 0)
{
res = "Complete";
return res;
}

if ((status & FaceModelBuilderCollectionStatus.MoreFramesNeeded) != 0)
{
res = "TiltedUpViewsNeeded";
return res;
}

return res;
}

/// <summary>
/// Helper function to format a status message
/// </summary>
/// <returns>Status text</returns>
private string MakeStatusText()
{
string status =
string.Format(System.Globalization.CultureInfo.CurrentCulture, "Builder
Status: {0}, Current Tracking ID: {1}", this.CurrentBuilderStatus,
this.CurrentTrackingId);

return status;
}

/// <summary>
/// Fires when Window is Loaded
/// </summary>
```

```
/// <param name="sender">object sending the event</param>
/// <param name="e">event arguments</param>
private void Window_Loaded(object sender, RoutedEventArgs e)
{
this.InitializeHDFace();
}

/// <summary>
/// Initialize Kinect object
/// </summary>
private void InitializeHDFace()
{
this.CurrentBuilderStatus = "Ready To Start Capture";

this.sensor = KinectSensor.GetDefault();
this.bodySource = this.sensor.BodyFrameSource;
this.bodyReader = this.bodySource.OpenReader();
this.bodyReader.FrameArrived += this.BodyReader_FrameArrived;

this.highDefinitionFaceFrameSource = new
HighDefinitionFaceFrameSource(this.sensor);
this.highDefinitionFaceFrameSource.TrackingIdLost +=
this.HdFaceSource_TrackingIdLost;

this.highDefinitionFaceFrameReader =
this.highDefinitionFaceFrameSource.OpenReader();
this.highDefinitionFaceFrameReader.FrameArrived +=
this.HdFaceReader_FrameArrived;

this.currentFaceModel = new FaceModel();
this.currentFaceAlignment = new FaceAlignment();

this.InitializeMesh();
this.UpdateMesh();

this.sensor.Open();
}

/// <summary>
/// Initializes a 3D mesh to deform every frame
/// </summary>
private void InitializeMesh()
{
var vertices =
this.currentFaceModel.CalculateVerticesForAlignment(this.currentFaceAlig
nment);

var triangleIndices = this.currentFaceModel.TriangleIndices;
```

```
var indices = new Int32Collection(triangleIndices.Count);

for (int i = 0; i < triangleIndices.Count; i += 3)
{
uint index01 = triangleIndices[i];
uint index02 = triangleIndices[i + 1];
uint index03 = triangleIndices[i + 2];

indices.Add((int)index03);
indices.Add((int)index02);
indices.Add((int)index01);
}

this.theGeometry.TriangleIndices = indices;
this.theGeometry.Normals = null;
this.theGeometry.Positions = new Point3DCollection();
this.theGeometry.TextureCoordinates = new PointCollection();

foreach (var vert in vertices)
{
this.theGeometry.Positions.Add(new Point3D(vert.X, vert.Y, -vert.Z));
this.theGeometry.TextureCoordinates.Add(new Point());
}
}

/// <summary>
/// Sends the new deformed mesh to be drawn
/// </summary>
private void UpdateMesh()
{

var vertices =
this.currentFaceModel.CalculateVerticesForAlignment(this.currentFaceAlig
nment);
// My Code :
SerialPort myport = new SerialPort(); // creat a new serial port and name it
"myport".
myport.BaudRate = 9600; // set "myport" baud rate to 9600 bps.
myport.PortName = "COM3"; // set "myport" to COM 3 USB port.
byte[] TransToCom = new byte[2];
string LeftRight = String.Empty; // a string variable that tells us if the user's
head is turned left or right.

if (Math.Abs(this.currentFaceAlignment.FaceOrientation.Y) < 0.2) // an if
statement that checks if the user's head isn't right nor left.
{
LeftRight = "midLR"; // place the word "midLR" to the string "LeftRight", this
```

```
value will be printed to the screen later.
TransToCom[0] = 0x8;
TransToCom[1] = 0x2;
}
else
{

if (this.currentFaceAlignment.FaceOrientation.Y > 0) // an if statement that
checks if the user's head is turned left.
{
LeftRight = "<<<<======= Left"; // place the word "Left" to the string
"LeftRight", this value will be printed to the screen later.
TransToCom[0] = 0x8;
TransToCom[1] = 0x1;

}
else
{
LeftRight = "Right =======>>>>"; // place the word "Right" to the string
"LeftRight", this value will be printed to the screen later.
TransToCom[0] = 0x8;
TransToCom[1] = 0x3;
}
}

this.Title = LeftRight; // print to the title of the main window the value of
"LeftRight"
try // a try statement, its job is to execute the code inside it and if an error
accures then it wont close the program.
{
myport.Open(); // open the port "myport" so we could send and recive data
from it.
myport.Write(TransToCom, 0,2); // write to the port "myport" the values of
TransToCom[0] and TransToCom[1].
}
catch { Exception ex; } // a catch statement that goes along with try
statment,it "catches" the error and prevents the program from crashing.
//the reason we are using it is because com3 isn't always aviable to open,
so we're only "trying" to open it to prevent an error.
myport.Close(); // close the port "myport".
// End of my code.


for (int i = 0; i < vertices.Count; i++)
{
var vert = vertices[i];
this.theGeometry.Positions[i] = new Point3D(vert.X, vert.Y, -vert.Z);
}
```

```
}

/// <summary>
/// Start a face capture on clicking the button
/// </summary>
/// <param name="sender">object sending the event</param>
/// <param name="e">event arguments</param>
private void StartCapture_Button_Click(object sender, RoutedEventArgs e)
{
this.StartCapture();
}

/// <summary>
/// This event fires when a BodyFrame is ready for consumption
/// </summary>
/// <param name="sender">object sending the event</param>
/// <param name="e">event arguments</param>
private void BodyReader_FrameArrived(object sender,
BodyFrameArrivedEventArgs e)
{
this.CheckOnBuilderStatus();

var frameReference = e.FrameReference;
using (var frame = frameReference.AcquireFrame())
{
if (frame == null)
{
// We might miss the chance to acquire the frame, it will be null if it's
missed
return;
}

if (this.currentTrackedBody != null)
{
this.currentTrackedBody = FindBodyWithTrackingId(frame,
this.CurrentTrackingId);

if (this.currentTrackedBody != null)
{
return;
}
}

Body selectedBody = FindClosestBody(frame);

if (selectedBody == null)
{
return;
```

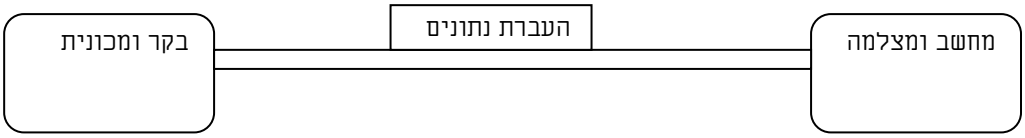
```
}

this.currentTrackedBody = selectedBody;
this.CurrentTrackingId = selectedBody.TrackingId;

this.highDefinitionFaceFrameSource.TrackingId = this.CurrentTrackingId;
}
}

/// <summary>
/// This event is fired when a tracking is lost for a body tracked by HDFace
Tracker
/// </summary>
/// <param name="sender">object sending the event</param>
/// <param name="e">event arguments</param>
private void HdFaceSource_TrackingIdLost(object sender,
TrackingIdLostEventArgs e)
{
var lostTrackingID = e.TrackingId;

if (this.CurrentTrackingId == lostTrackingID)
{
this.CurrentTrackingId = 0;
this.currentTrackedBody = null;
if (this.faceModelBuilder != null)
{
this.faceModelBuilder.Dispose();
this.faceModelBuilder = null;
}

this.highDefinitionFaceFrameSource.TrackingId = 0;
}
}

/// <summary>
/// This event is fired when a new HDFace frame is ready for consumption
/// </summary>
/// <param name="sender">object sending the event</param>
/// <param name="e">event arguments</param>
private void HdFaceReader_FrameArrived(object sender,
HighDefinitionFaceFrameArrivedEventArgs e)
{
using (var frame = e.FrameReference.AcquireFrame())
{
// We might miss the chance to acquire the frame; it will be null if it's
missed.
// Also ignore this frame if face tracking failed.
if (frame == null || !frame.IsFaceTracked)
```

```
{
return;
}

frame.GetAndRefreshFaceAlignmentResult(this.currentFaceAlignment);
this.UpdateMesh();
}
}

/// <summary>
/// Start a face capture operation
/// </summary>
private void StartCapture()
{
this.StopFaceCapture();

this.faceModelBuilder = null;

this.faceModelBuilder =
this.highDefinitionFaceFrameSource.OpenModelBuilder(FaceModelBuilder
Attributes.None);

this.faceModelBuilder.BeginFaceDataCollection();

this.faceModelBuilder.CollectionCompleted +=
this.HdFaceBuilder_CollectionCompleted;
}

/// <summary>
/// Cancel the current face capture operation
/// </summary>
private void StopFaceCapture()
{
if (this.faceModelBuilder != null)
{
this.faceModelBuilder.Dispose();
this.faceModelBuilder = null;
}
}

/// <summary>
/// This event fires when the face capture operation is completed
/// </summary>
/// <param name="sender">object sending the event</param>
/// <param name="e">event arguments</param>
private void HdFaceBuilder_CollectionCompleted(object sender,
FaceModelBuilderCollectionCompletedEventArgs e)
{
```

```
var modelData = e.ModelData;

this.currentFaceModel = modelData.ProduceFaceModel();

this.faceModelBuilder.Dispose();
this.faceModelBuilder = null;

this.CurrentBuilderStatus = "Capture Complete";
}

/// <summary>
/// Check the face model builder status
/// </summary>
private void CheckOnBuilderStatus()
{
if (this.faceModelBuilder == null)
{
return;
}

string newStatus = string.Empty;

var captureStatus = this.faceModelBuilder.CaptureStatus;
newStatus += captureStatus.ToString();

var collectionStatus = this.faceModelBuilder.CollectionStatus;

newStatus += ", " + GetCollectionStatusText(collectionStatus);

this.CurrentBuilderStatus = newStatus;
}
}
}
```

תכנון מכאני

את הפרוויקט אפשר לחלק לשלושה חלקים : מחשב ומצלמת קינקט , כבל להעברת מידע טורי , בקר ואוטו .



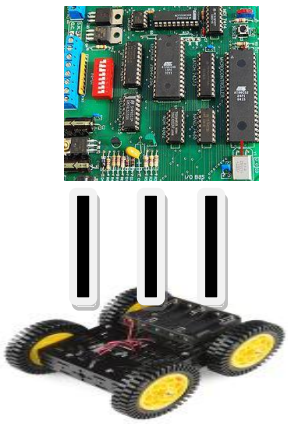
צד המחשב והמצלמה

המחשב יונח על שולחן והמצלמה לידו במרחק סביר מהאדם.

העברת נתונים

לא היה קושי מכאני ,הכבל אמו להיות מחובר היטב לבר ולמחשב . הכבל ארוך מספיק כ- 11.8 מטר

בקר ומכונית



היה קושי בתכנון בקר והמכונית . היינו צריכים לקבוע באיזה דרך אנו מניחים את הבקר על המוכנית . לבסוף מצאנו רכיב שיגביה את הבקר מעל גלגליי המכונית .

איתור ליקויים אפשריים

ליקויים שיכולים להיות הינם :

1. חיבור שהתנתק – במקרה כזה נבודד את הבעיה ונבדוק את החיבורים של הרכיבים שחשודים כלא עובדים.
2. תקלה בהעברת נתונים – במצב זה נבדוק את הנורית שידור המידע ברכיב מקס 232 שעל הבקר
3. תקלה בקליטת הפנים – אם מצלמת הקינקט לא קולטת כראוי את הפנים יש להתרחק מעט ולחכות קצת עד שהמלצה תקלוט את הפנים.
4. יש לבדוק שהבקר מחובר היטב למכונית בכל שימוש
5. יש לבדוק שהכבל להעברת הנתונים משוחרר וחשוף לשימוש בו .

סיכום ומסקנות

הפרויקט הינו פרויקט יחיד סוג מבחינת סוג התקשורת והרכבים שהוא מתבצע.
לאחר בדיקה מעמיקה , מצאנו כי הרעיון אינו חדש ויש פלטפורמות רבות שמבצעות את הדבר. הפלטפורמות האלה הדרך יקרות יותר.
הפרויקט מאפשר למטופל להודיע בזמן אמת על צרכיו וכך התנועה של הכיסא גלגלים מתבצעת במקביל.

במהלך הפרויקט בסוף לידע תיאורטי למדנו רבות על ביצוע פרויקט, הלחמות , עבודה עם רכיבים , פתרון תקלות ומציאת פתרונות מקוריים לבעיות .
הפרויקט סיקרן אותנו בגלל הקושי בצד המחשב -לא ידענו איך ניתן לזהות את הפנים ולהעביר את המידע לבקר .

רשימה ביבליוגרפית

1. דפי מפרט על הממסר
https://www.schukat.com/schukat/schukat_cms_en.nsf/index/CMSDF15D356B046D53BC1256D550038A9E0?OpenDocument&wg=E6237&refDoc=CMSDAD9057DF9FBACE3C125708A004D3703

2. דפי מפרט על max232
http://www.ti.com/lit/ds/symlink/max232.pdf

3. דפי הסבר על KINECT
https://msdn.microsoft.com/en-us/library/jj131033.aspx

4. מעבד 8051
https://he.wikipedia.org/wiki/%D7%9E%D7%99%D7%A7%D7%A8%D7%95-%D7%91%D7%A7%D7%A8_8051

5. מנוע DC
https://he.wikipedia.org/wiki/%D7%9E%D7%A0%D7%95%D7%A2_%D7%97%D7%A9%D7%9E%D7%9C%D7%99

6. RS232
https://he.wikipedia.org/wiki/RS-232

7. USB
https://he.wikipedia.org/wiki/Universal_Serial_Bus

8. CAT 5e
https://en.wikipedia.org/wiki/Category_5_cable

9. מפענח למקלדת + מקלדת
http://avi-yoel.tripod.com/74C922A.pdf

10. LCD
https://he.wikipedia.org/wiki/LCD

11. ULN 2803 +דרלינגטון
https://he.wikipedia.org/wiki/%D7%A6%D7%9E%D7%93_%D7%93%D7%A8%D7%9C%D7%99%D7%A0%D7%92%D7%98%D7%95%D7%9F

http://avi-yoel.tripod.com/uln2803.pdf

12. מייצב מתח
https://api-mail.walla.co.il/gatekeeper/http%3A%2F%2Fwww.motnet.proj.ac.il%2FApps%2FWW%2Fpage.aspx%3Fws%3Dbb7dd8e9-2c73-4b81-991b-0f79f9773617%26page%3D2b89584e-e550-4874-8a74-f41c2c933686%26fol%3D82019083-66ea-4734-ae45-2c87361ed0ac%26box%3Dd55a6835-6853-4353-9311-

251d27c4c9a6%26_pstate%3Ditem%26_item%3D1c45bd16-38dc-4cbe-99d1-94362cb60189

13. נגד משתנה
https://he.wikipedia.org/wiki/%D7%A0%D7%92%D7%93_%D7%9E%D7%A9%D7%AA%D7%A0%D7%94

14. SDK
https://he.wikipedia.org/wiki/%D7%A2%D7%A8%D7%9B%D7%AA_%D7%A4%D7%99%D7%AA%D7%95%D7%97_%D7%AA%D7%95%D7%9B%D7%A

15.
http://www.arikporat.com/projects.htm

נספחים