# תוכן עניינים

[*תוכן עניינים* - 1 -](#_Toc384488371)

[תודות - 4 -](#_Toc384488372)

[תקציר - 5 -](#_Toc384488373)

[פרק 1 מבוא - 6 -](#_Toc384488374)

[1.1 – הצגת המכשיר - 7 -](#_Toc384488375)

[1.2 – הוראות הפעלה - 8 -](#_Toc384488376)

[1.2.1 – מצד המשתמש (השמעה): - 8 -](#_Toc384488377)

[1.2.2 – מצד המפעיל (הקלטה): - 8 -](#_Toc384488378)

[1.3 - הסבר סכימת מלבנים - 9 -](#_Toc384488379)

[1.4-שרטוט חשמלי - 11 -](#_Toc384488380)

[הסבר שרטוט חשמלי - 12 -](#_Toc384488381)

[פרק 2 חומרה - 12 -](#_Toc384488382)

[2.1 – הכרת המיקרו- בקר P89V51RD2 : - 14 -](#_Toc384488383)

[2.1.1 - הסבר איור מלבנים ה- P89V51RD2: - 14 -](#_Toc384488384)

[2.1.2 - ארגון הזיכרון הפנימי: - 18 -](#_Toc384488385)

[2.1.3 - הדקי הבקרה והתזמון: - 23 -](#_Toc384488386)

[2.1.5 - פסיקות (Interrupts): - 25 -](#_Toc384488387)

[2.1.6 - הפורט הטורי Serial Port : - 29 -](#_Toc384488388)

[2.1.7 - חיבור מעגל לביצוע RESET: - 33 -](#_Toc384488389)

[2.2 - זיכרון התוכנית - 34 -](#_Toc384488390)

[2.3- תצוגתLCD : ( (Liquid Crystal Display: - 35 -](#_Toc384488391)

[2.3.1 - תפקיד התצוגה במערכת: - 35 -](#_Toc384488392)

[2.3.2 - עיקרון פעולת התצוגה המבוססת על גביש נוזלי: - 35 -](#_Toc384488393)

[2.3.3 - תפקידי ההדקים של תצוגת LCD : - 36 -](#_Toc384488394)

[2.3.4 - תכונות התצוגה : - 37 -](#_Toc384488395)

[2.3.5 - זיכרון התצוגה : - 38 -](#_Toc384488396)

[2.3.6 - המבנה הפנימי של התצוגה: - 38 -](#_Toc384488397)

[2.3.7 - פרוט הוראות הבקרה : - 41 -](#_Toc384488398)

[2.4–מקלדת מחשב בחיבורPS2 - 44 -](#_Toc384488399)

[מערך קודי מקלדת - 45 -](#_Toc384488400)

[2.4.1 - פרוטוקול PS2 - 47 -](#_Toc384488401)

[מבוא ונתונים על הממשק בין המקלדת PS/2 ו-host ברמת ה-bit - 49 -](#_Toc384488402)

[2.5- חיישן צבע TCS3200 - 51 -](#_Toc384488403)

[אופן שימוש הרכיב בפרוייקט - 54 -](#_Toc384488404)

[תרשים זרימה למדידת תדר מוצא החיישן בשיטת ספירת פולסים בזמן קצוב - 56 -](#_Toc384488405)

[2.6- רכיב דיבור 4004ISD - 57 -](#_Toc384488406)

[2.6.1 - מבוא: - 57 -](#_Toc384488407)

[2.6.2 - הדקי הרכיב: - 58 -](#_Toc384488408)

[2.6.3 - תכונות הרכיב - 58 -](#_Toc384488409)

[2.6.4 - הסבר דיאגרמת המלבנים - 61 -](#_Toc384488410)

[2.6.5 - איכות קול ודיבור: - 63 -](#_Toc384488411)

[2.6.6 - זיכרון ה-FLASH: - 64 -](#_Toc384488412)

[2.6.7 - ממשק המיקרו בקר: - 66 -](#_Toc384488413)

[2.6.8 - תהליך התקשורת בממשק SPI - 68 -](#_Toc384488414)

[2.6.9 - קודי הפעולה - 68 -](#_Toc384488415)

[2.6.10- אוגרי הבקרה - 70 -](#_Toc384488416)

[2.6.11 - רגלי הרכיב: - 71 -](#_Toc384488417)

[2.6.12 - אופן הפעלת הרכיב: - 74 -](#_Toc384488418)

[2.6.13-חיבור הרכיב למערכת מיקרו : - 77 -](#_Toc384488419)

[2.7-מגבר שמע LM386 - 79 -](#_Toc384488420)

[2.7.1 - מבוא: - 79 -](#_Toc384488421)

[2.7.2 - מאפייני הרכיב: - 79 -](#_Toc384488422)

[2.7.3 - יישומים לרכיב: - 79 -](#_Toc384488423)

[2.7.4 - חיבור המגבר למעגל: - 80 -](#_Toc384488424)

[2.8- מיקרופון - 81 -](#_Toc384488425)

[2.9-רמקול 84](#_Toc384488426)

[2.10-מייצב מתח גבוה HT 71 XX 85](#_Toc384488427)

[פרק 3 תכנה 87](#_Toc384488428)

[3.1-תרשים זרימה כללי 88](#_Toc384488429)

[3.2-תכנת הפרויקט 89](#_Toc384488430)

[תכנה ראשית 89](#_Toc384488431)

[שגרות ל-LCD 90](#_Toc384488432)

[השהיה 96](#_Toc384488433)

[השהיית LCD 96](#_Toc384488434)

[שגרת אתחול התצוגה 96](#_Toc384488435)

[שגרות זיהוי הצבע: 96](#_Toc384488436)

[שגרת סריקת מקש או צבע מן המקלדת או חיישן הצבע: 99](#_Toc384488437)

[שגרת קריאת מילה לתוך מערך מתוך המקלדת 100](#_Toc384488438)

[שגרות רכיב הקול ISD-4004 102](#_Toc384488439)

[שגרות הקלטה: 105](#_Toc384488440)

[שגרת השמעה מכתובת נתונה: 106](#_Toc384488441)

[שגרה להקלטה של מחרוזות מרובות תווים: 107](#_Toc384488442)

[שגרה להשמעה של מחרוזות מרובות תווים: 108](#_Toc384488443)

[שגרות טכניות: 109](#_Toc384488444)

[קבצי הכרזה על פונקציות 110](#_Toc384488445)

[סיכום ומסקנות 112](#_Toc384488446)

# תודות

תודה גדולה למנחה שלי יואל שעזר לי רבות בענייני החמרה

למורה חנוך שגם הוא עזר לי אף שאינו מחויב לכך

לאחי בני שהחליף בשבילי רכיבים ובאותה הזדמנות לכלל משפחתי שתמכו בי תמיד

ולחברי ללימודים שעמדו לעזרתי בכל עניין ועניין.

# תקציר

מערכת הלימוד מיועדת עבור ילדים בגילאים 3 עד 7. בשיתוף פעולה עם ההורים, הילדים יכולים כבר בשלבים המוקדמים ללמוד, לדעת ולזהות אותיות, מספרים וצבעים , מערכת זו היא מערכת פשוטה ונוחה לשימוש. המערכת מאפשרת גם הקלטה דינאמית של האותיות המספרים והצבעים על גבי כתובות מוקצות מראש, בפונקציה זו משתמש מפעיל הפרויקט בלבד .

בשלבים יותר מאוחרים של הלימוד הילד יכול להפעיל את המערכת לבד ולבחון את עצמו.

על הילד או הורו להקיש מקשים על המקלדת או לחילופין לשים פתקית צבע על חיישן הצבע.

אם ההורה בחר לימוד צבעים , ההורה יבחר פתקית צבע וישאל את ילדו את שם הצבע. לאחר שהילד ענה על השאלה הילד יכניס את פתקית הצבע לקופסת הזיהוי. מיד עם זיהוי הפתקית המערכת תשמיע את שמו של הצבע.

# פרק 1 מבוא

## 1.1 – הצגת המכשיר

המכשיר מבוסס מיקרו-בקר ממשפחת ה-8051. המכשיר כולל צג LCD, חיישן צבע, ומערכת דיבור.

המשתמש נדרש להקיש על המקלדת אות או מילה ואז על מקש האנטר או לקחת פתקית צבעונית ולשים אותה על חיישן הצבע.

התכנית סורקת סדרתית את המקלדת ואת חיישן הצבע אם הצבע שנקלט גבוה מערכי סף מסוימים מציגה את היחסים ואם הצבע מזוהה משמיע את שמו ומציג את שמו על ה-LCD. בדומה גם עם המקלדת. כאשר נלחץ מקש בפעם הראשונה לא נסרק עוד חיישן הצבע עד סיום קליטת המילה ועיבודה.

אם הוקשה אות על המקלדת הפרויקט ישמיע אותה

וכך גם אם הוקשה מילה מתוך מספר קטן של מילים מוסכמות שהוא מערך שמות הצבעים. אם המילה או האות שהוקשה אינה תקינה הפרויקט יציג הודעת שגיאה ויחזור להודעות הפתיחה.

הפרויקט כאמור מאפשר הקלטה דינאמית תוך כדי פעולתו הרגילה. המחרוזת bego משמשת תו בקרה למעבר בין השמעה להקלטה.

אם הוקשה המחרוזת bego התכנית עוברת למצב הקלטה .אז תוצג על הLCD בקשה להקיש על המקלדת כדי להקליט אות או מילה. עם לחיצת האנטר המיקרופון מתחיל להקליט על הכתובת המוקצה. וימשיך לבקש הקלטה עד אשר ייקלט התו ; אז תחזור התכנית למצב השמעה.

## 1.2 – הוראות הפעלה

### 1.2.1 – מצד המשתמש (השמעה):

1. חבר את המערכת בעזרת כבל USB למחשב דלוק כלשהו.
2. חבר את מקלדת המקשים למערכת.
3. הקש על אות או הנח פתקית צבעונית על חיישן הצבע.
4. שמע את שם האות או שם הצבע מהרמקול ונסה לחזור אחריו.
5. חזור על 3 ו4.

### 1.2.2 – מצד המפעיל (הקלטה):

1. חבר את המערכת בעזרת כבל USB למחשב דלוק כלשהו.
2. חבר את מקלדת המקשים למערכת.
3. הקש את המחרוזת bego ואחריה על מקש האנטר למעבר למצב הקלטה.
4. הקש את האות או המחרוזת שברצונך להקליט.
5. לחץ על מקש האנטר והקלט למיקרופון .
6. חזור על 3+4 עבור כל המחרוזות שברצונך להקליט.
7. על מנת לחזור למצב השמעה הקש ';' ואנטר.

## 1.3 - הסבר סכימת מלבנים

מערכת זו פועלת בצורה הבאה:

* **מיקרו בקר P89V51RD2**– תפקידו של הבקר לבקר את פעולת המערכת, כלומר לתאם בין חלקי המערכת, ולתעל את המידע בצורה נכונה כך שהמערכת תפעל בצורה הטובה ביותר. הבקר קורא את הוראות התוכנית מזיכרון התוכנית ומבצע אותה.
* **חיישן צבע**–חיישן הממיר צבע לתדר בהתאם למסנן שנבחר. החיישן משמש לזיהוי הצבע שנבחר.
* **תצוגת ה-L.C.D** - מחוברת לפס הנתונים של המיקרו-בקר, התצוגה משמשת להצגת: התווים שהוקשו או הצבעים שזוהו על ידי החיישן, וכן הוראות לשימוש במערכת.
* **מקלדת מחשב PS2**–מקלדת מחשב עם מחבר PS2. המקלדת משמשת כממשק משתמש וכן כמקלדת לימוד אותיות וספרות.
* **זיכרון תכנית מסוג 64K FLASH ROM**- משמש לאכסון תכנית המערכת, טבלאות, הודעות למשתמש. זיכרון זה נמצא בתוך המיקרו-בקר וניתן למחוק ולצרוב אותו במהירות מספר רב של פעמים.
* **רכיב קול ISD4004** – רכיב המשמש להקלטה והשמעת הודעות. הרכיב מכיל זיכרון שבו מאוחסנות ההודעות בעת ההקלטה.
* **מגבר שמע** - מגבר שמע מונוליטי LM386 להספק עד 1 ווט.
* **מיקרופון** – משמש להקלטת הודעות באופן דינמי.
* **רמקול** – משמש כהתקן להשמעת ההודעות הקוליות.
* **מייצב מתח**- משמש לאספקת מתח הזנה לרכיב הקול

סכימת מלבנים כללית

**מקלדת מחשב PS2**

**מיקרו בקר P89V51RD2**

**תצוגת L.C.D**

**זיכרון תוכנית**

**64K FLASH ROM**

**מייצב מתח**

**רכיב קול**

**מגבר שמע**

**רכיב קול**

**מיקרופון**

**רמקול**

**חיישן צבע**

## 1.4-שרטוט חשמלי

### הסבר שרטוט חשמלי

1)השתמשתי בערכה של שי מלול בה הדק 2 מחובר ל-lcd.

2)את ששת ההדקים הגבוהים של הדק 3 חיברנו לחיישן הצבע ולמקלדת ואת הדק 1 לרכיב הקול.

3)את המקלדת חיברנו ל2 ההדקים הנמוכים של הדק 3.

4) הדק 1 חובר לרכיב הקול.

5)רכיב הקול מקבל אספקה של 3v ממייצב המתח.

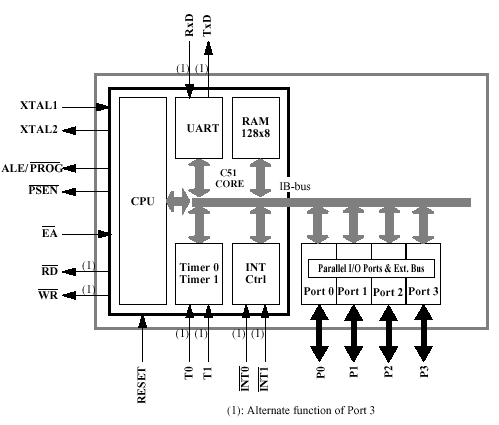
6)רכיב הקול מחובר מצדו האחד למעגל הקליטה של המיקרופון.

7)מן הצד השני רכיב הקול מחובר למגבר השמע.

8)מגבר השמע מחובר אל הרמקול.

# פרק 2 חומרה

## 2.1 – הכרת המיקרו- בקר P89V51RD2 :



**איור מלבנים כללי של ה- P89V51RD2**

### 2.1.1 - הסבר איור מלבנים ה- P89V51RD2:

* CPU– יחידת עיבוד מרכזית, הנה היחידה שמעבירה נתונים בין היחידות השונות של המיקרו בקר, ומעבדת אותם בהתאם לתוכנית האגורה בזיכרון.
* Ram– זיכרון כתיבה / קריאה בן 128 בתים המיועדים לנתונים משתנים (כתובות 00H-7FH).
* I.C.- בקרת הפסיקות, קובעת את הביצועים וההתנהגות של הרכיב בעת התרחשות "אירוע " המתקבל מאחד הרכיבים הפריפריאליים להפעלת הפסיקה INT1 ו- INTO.
* SerialPort– מהווה 2 קווים מפורט 3 המשמשים גם לתקשורת טורית.

**מערך הפורטים :**

PORT 0 - פורט 0 הוא פורט I/O דו כיווני בעל 8 קווים OPEN DRAIN .

פורט זה משמש כפס מידע ומרובב עם 8 הסיביות התחתונות של פס הכתובות.

במצב של '1' יתנהג כל פורט בכניסה כעכבה גבוהה.

PORT 1 - פורט I/O דו כיווני בעל 8 קווים. פורט זה הינו פורט דו – כיווני לקלט ולפלט עם נגדי PULL UP פנימיים.

PORT2– פורט זה הנו דו כיווני המשמש לקלט ולפלט של נתונים.

8 קווי פורט זה מחוברים לנגדי PULL UP פנימיים.

PORT 3– פורט זה הנו דו כיווני המשמש לקלט ולפלט של נתונים.

8 קווי פורט זה מחוברים לנגדי PULL UP פנימיים.

פורט זה משמש לפונקציות נוספות כדלקמן :

|  |  |  |
| --- | --- | --- |
| קו תקשורת טורית לקליטה (UART) | RXD | P2.0 |
| קו תקשורת טורית לשידור (UART) | TXD | P2.1 |
| קו לפסיקה חיצונית 0 | INT0 | P2.2 |
| קו לפסיקה חיצונית 1 | INT1 | P2.3 |
| כניסה חיצונית לטיימר 0 | TO | P2.4 |
| כניסה חיצונית לטיימר 1 | T1 | P2.5 |
| קו כתיבה לזיכרון חיצוני | WR | P2.6 |
| קו קריאה מזיכרון חיצוני | RD | P2.7 |

**רגלי פורט 3 ושימושן.**

**מונים (TIMER/COUNTERS):**

ב –P89V51RD2 ישנם 2 מונים / טיימרים (T0 ו- T1) בני 16 סיביות כל אחד הניתנים לתכנות ב- 4 אופני פעולה.

בפונקציית עבודה כ- CounterT0 ו-T1 משמשים לספירת מאורעות , ליצירת פסיקות בקצבים קבועים , ולמדידת זמנים של רוחב דפקים. המונה מוגדל בתגובה לירידה מ-'1' ל- '0' בכניסות T0 או T1 .

בפונקציית עבודה כ- Timer , האוגר של הטיימר מוגדל כל מחזור מכונה (המהוה 12 מחזורי המתנד). ניתן בעצם לראות בזמנן "מונה מחזורי מכונה ".

אופני הפעולה הם:

MODE0 :

באופן פעולה 0 המונה משמש כמונה של 13 סיביות . משתמשים ב- 8 סיביות של TH1 וב- 5 סיביות של TL1. כאשר כל 13 הסיביות עולות ל- '1' ,דגל הפסיקהTF1 עולה ל-'1' ומתבקשת פסיקה.

כדי למנות שוב יש לטעון את האוגרים מחדש.

MODE1 :

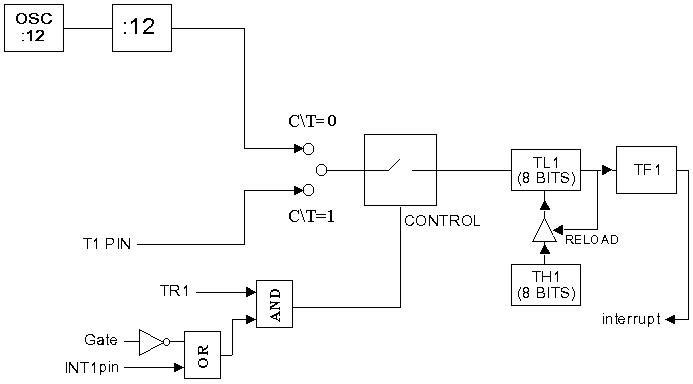
אופן הפעולה זה זהה לאופן הפעולה 0 אלא שבאופן פעולה זה המונה בעל 16 סיביות כאשר כל הסיביות TL1 – TH1 בשימוש. אופן זה מועדף כאשר יש צורך לעבוד בזמנים גדולים יותר. כאשר כל 16 הסיביות עולות ל- '1' ,דגל הפסיקהTF1 עולה ל-'1' ומתבקשת פסיקה.

כדי למנות שוב יש לטעון את האוגרים מחדש.

MODE 2:

באופן פעולה זה מתייחסים לזמנן כאוגר בן 8 סיביות עם טעינה אוטומטית(AUTORELOAD).TH נטען לערך מסוים, בתחילת כל המנייה מספר זה נטען ל- TL1 ,

המונה מתקדם באחד כל מחזור מכונה וכאשר הוא מגיעה לגלישה דגל הפסיקה TF1 עולה ל-'1' ובזמן זה ערך TH1 נטען ל-TL1 מחדש, הספירה מתחילה מחדש וכך חוזר חלילה. כך מקבלים טעינה אוטומטית של המונה ללא צורך לאתחל את המונים כל פעם מחדש .



**סכימת מלבנים של מונה באופן עבודה 2**

MODE 3 :

בצורת עבודה זו הופך T0 לשני מונים נפרדים בני 8 סיביות כל אחד , הפועלים ללא קשר זה עם זה . TL0 פועל כרגיל כשהוא משתמש בסיביות הבקרה של T0, ואילו TH0 פועל רק לפי הגביש הפנימי כשהוא משתמש בסיביות בקרה של T1, לכן במצב זה T1 אינו יכול לפעול.

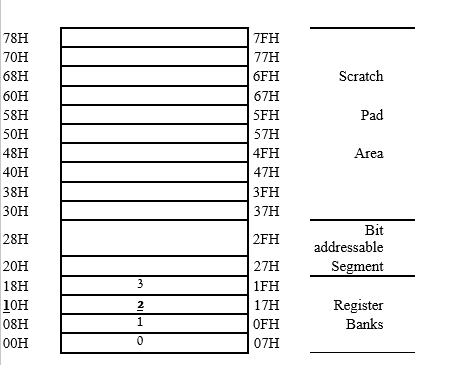
זמנן T1 באופן עבודה 3 פשוט שומר את ערכו.

### 2.1.2 - ארגון הזיכרון הפנימי:

במיקרו בקר קיים זיכרון RAM פנימי בעל 256 בתים.

128 הבתים התחתונים מתחלק ל- 3 אזורים :

1. האזור התחתון מכיל 4 בנקים של אוגרים ( כל בנק מכיל 8 אוגרים R0-R7 ). סה"כ 32Bytes בכתובות 1FH – 00H .
2. אזור בעל 16 תאי RAM בכתובות 2FH – 20H העובד גם במעון סיביות (ס"ה 128 סיביות ).
3. אזור RAM זמני (scratch pad) של 60H בתים בכתובות 30H-7FH



**128 הבתים התחתונים ב-RAM הפנימי של המיקרו P89LV51**

האזור העליון אף הוא בגודל של 128 בתים , אולם באזור זה פעילים רק אוגרים מיוחדים (Special Function Registers) המפוזרים על פני האזור.

אזור הבנקים של האוגרים

קביעת הבנק הפעיל מתבצעת ע"י הסיביות RSO ו- RS1 שבאוגר PSW .

השימוש בבנקים השונים חוסך בשימוש במחסנית לעיתים תכופות.

|  |  |  |
| --- | --- | --- |
| **BANK** | **RSO** | **RS1** |
| 0 | 0 | 0 |
| 1 | 1 | 0 |
| 2 | 0 | 1 |
| 3 | 1 | 1 |

בחירת הבנקים השונים.

* + אזור למעון סיבית ואזור RAM רגיל (7FH-20H ).

א. אזור למעון סיבית 2FH–20H :

אזור של 16BYTES שבו ניתן לפנות לכל סיבית בנפרד. בשיטה זו לכל סיבית כתובת מ- 00H עד 7FH . השליטה על מצב סיביות מתבצעת באמצעות פעולתclr, setb, and, or, not, mov ניתן להעביר תוכן הסיבית לדגל הנשאCY אשר באוגר PSW ועל ידי זה ניתן לבצע פעולות קפיצה בתלות במצב סיבית בודדת.

ב. אזור RAM רגיל ( 7FH – 30H ) - אזור בגודל של 80 בתים לשימושם שונים .

ניתן להשתמש בכל הפקודות במעון ישיר וכן פקודות במעון עקיף באמצעות : R1@ ו- R0@.

* + מערך האוגרים המיוחדים:

במערך זה 20 אוגרים 11 מהם ניתנים למעון סיביות בודדות והם :

1. ACC– משמש לביצוע חלק גדול מההוראות החישוביות, וכן משמש לקליטת ולשליחת נתונים מ/ואל התקנים וזיכרונות חיצוניים.
2. B- משרת את הוראות הכפל והחילוק.
3. SCON– אוגר לבקרת תקשורת טורית (Serial Control).
4. TCONTimer Control)) - אוגר לבקרת פעולות המונים
5. IE/IP– 2 אוגרים המבקרים את אפשור הפסיקות השונות ועדיפותן.
6. PSW- אוגר הסטטוס המכיל 7 דגלים.
7. 3P - פורט 3 .
8. 2P–פורט 2 .
9. 1P– פורט 1.
10. 0P-פורט 0.
11. PC (PROGRAM COUNTER) - מונה התוכנית לזיכרון התוכנית:

זהו אוגר בעל 16 סיביות המצביע על כתובת ההוראה העומדת להתבצע.

האוגר מקודם אוטומטית תוך כדי ביצוע ההוראה.

1. SP-STACK POINTER אוגר מצביע המחסנית:

באוגר זה מופיעה הכתובת של הנתון האחרון שנכנס לזיכרון המחסנית.

1. DPTR (DATA POINTER ) - אוגר מצביע הכתובות הנתונים:

אוגר זה בנוי מ- שני אוגרים בני 16 סיביות כל אחד (DPH-DPL).

אוגר זה פועל כאינדקס. תפקידו לסמן כתובות בזיכרון הנתונים , ניתן בעזרתו לפנות לטבלאות במעון לפי אוגר אינדקס.

1. אוגר TMOD :

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| M0 | M1 | C/T | GATE | M0 | M1 | C/T | GATE |

אוגר ה-TMOD קובע את אופן העבודה של טיימרים . 4 הסיביות הנמוכות מתייחסות אל T0 ו- 4 הגבוהות ל- T1 .

GATE: אפשור עבודה של המונים ללא תלות בכניסות החיצוניות של הפסיקות.

C/T: קובע האם T0 ו- T1 ישמשו כמונים או זמננים.

'1' : מונה , '0' – זמנן.

|  |  |  |
| --- | --- | --- |
| **MO** | **M1** | **MODE** |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 2 |
| 1 | 1 | 3 |

בחירת אופן עבודה של מונה .

1. אוגר TCON–TIMER CONTROL :

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| IT0 | IE0 | IT1 | IE1 | TRO | TFO | TR1 | TF1 |

TF1 - סיבית זו מייצגת את הגלישה של T1 .

TR1- סיבית זו מאפשרת את פעולת TR1.

TF0 - סיבית זו מייצגת את הגלישה של T0.

TR0 - סיבית זו מאפשרת את פעולתו של T0 .

IE1– '1' בדגל זה מציין כי התקבלה פסיקה ב- INT1.

IT1– קובע האם כניסה INT1 תהיה פעילה ברמה או בירידה.

'0'- ברמה נמוכה , '1' –בירידה.

IE0 -'1' בדגל זה מציין כי התקבלה פסיקה ב- INT0 .

IT0– קובע האם כניסה INT0 תהיה פעילה ברמה או בירידה.

'0' – ברמה נמוכה , '1' – בירידה .

### 2.1.3 - הדקי הבקרה והתזמון:

* + -ADDRESS LATCH ENABLEALE:

יציאה זו משמשת לנעילת הבית הנמוך של הכתובות בזמן הפנייה לזיכרון התוכנית או זיכרון נתונים חיצוני. במילים אחרות רגל זו מסמלת את מצב הקווים המרובבים.

* + PSEN-PROGRAM STORE ENABLE

קו זה משמש לקריאת זיכרון תכנות חיצוני ROM. ה-PSEN שונה מהדק ה-READ בשתי הנקודות הבאות:

1. ה- PSEN משמש לקריאה מזיכרון Program בעוד שה-READ משמש לקריאה מזיכרון ה-Data .

2. ה-PSEN מהיר פי 2 מ- READ , הוא פעיל פעמיים בכל מחזור מכונה בעוד ש- READ פעיל רק פעם אחת.

* + EA- EXTERNAL ADDRESS :

הדק זה קובע אופן חיבור זיכרון ה תוכנה (Program Memory) למיקרו בקר (זיכרון חיצוני או פנימי ). פרויקט זה מבוסס על מיקרו בקרP89V51RD2 ומכיוון שאין בו זיכרון תוכנית פנימי, על רגל EA לקבל רמה קבועה של '0'.

* + מתנד גל ריבועי :XTAL1 –XTAL2 :

תדר המתנד נקבע ע"י גביש חיצוני המחובר להדקים XTAL1 ו-XTAL2 . תדר זה מהווה את תדר הסנכרון לפיו נקבע קצב ביצוע כל פעולות הרכיב.

בפרויקט זה נקבע תדר מתנד על פי גביש של 11.052MHZ .

* + קווי RD/WR (קווי קריאה כתיבה ) :

קווים אלה פעילים ברמה של '0'.

קווים אלה נועדו לביצוע קריאה וכתיבה לרכיבי זיכרון חיצוני או להתקני I/O . רגל ה- WR משמשת לכתיבת נתונים ב-RAM החיצוני. רגל RD נתונים מה-RAM וה-ROM חיצוניים.

2.1.4 - אזורי RAM/ROM חיצוניים:

* + אזור RAM חיצוני:

טווח ה-RAM החיצוני הוא בגודל 64K והפנייה אליו תתאפשר אך ורק במעון עקיף ע"י פקודות מסוג MOVX. פקודה זו כוללת OPCODE ומציין כתובת , והנתון מועבר תמיד מ- ACC לתא ב-RAM או להפך.

* + אזור ROM חיצוני:

טווח ה-ROM החיצוני הוא עד 64K והפנייה אליו מתבצעת באופן אוטומטי לפי התקדמות הפקודות בתוכנית באמצעות קו הבקרה PSEN .

### 2.1.5 - פסיקות (Interrupts):

ה- P89V51RD2 מכיל 5 מקורות פסיקה .

2 פסיקות חיצוניות : INTO ו- INT1 .

קיימות גם פסיקות של טיימר 0 וטיימר 1 , המיוצרות ע"י TF0 ו- TF1 בהתאמה.

כאשר מונה מסיים ספירתו וישנה גלישה אז דגל הגלישה עולה ל- '1' ואז מתבקשת פסיקה . דגל זה מתאפס אוטומטית עם ביצוע הפסיקה.

* + מערכת הפסיקות :

ב- P89V51RD2 ישנם 3 אפשרויות לקבלת פסיקות :

א. פסיקות ע"י מקור חיצוני דרך אחד מהקוויםINT0 או INT1.

ב. פסיקות שמוסר כל אחד מהמונים הפנימיים בהסתיים מחזור מנייה שלו .

ג. פסיקות המתקבלות באמצעות התקשורת הטורית.

ניתן למסך כל קו פסיקה וגם ניתן לסדר את הפסיקות בסדר עדיפויות מסוים כך שאם מתקבלות 2 פסיקות בו זמנית יידע הבקר למי לענות ראשון.

בשעת היענות לפסיקה קופצת התוכנית לכתובת קבועה בזיכרון לביצוע התוכנית הפסיקה הנדרשת ובסיום הפעולה הנדרשת חוזר הבקר למקום בתוכנית בו היה לפני קבלת הפסיקה.

* + פסיקות :

כאשר מתבקשת פסיקה ממקור כלשהו , הבקר בודק האם פסיקה זו מאופשרת , לאחר מכן הוא בודק שלא מתבצעת באותו הרגע תוכנית שירות לפסיקה מקבוצת העדיפות הגבוהה .

אח"כ הוא מסיים את הפקודה בה הוא נמצא ובסיומה הוא יוצר קריאה לתת שגרה שמתאימה לאותה פסיקה.

בזמן היציאה לפסיקה , נשמר ערך PC וה-PSW במחסנית. ה-PC מקבל ערך חדש ע"פ מקור הפסיקה המבוקש .

|  |  |
| --- | --- |
| כתובת הקפיצה | מקור הפסיקה |
| 0003H | IE0 |
| 000BH | TF0 |
| 0013H | IE1 |
| 001BH | TF1 |
| 0023H | RI+T1 |

כתובת קפיצת הפסיקות השונות.

ביצוע תוכנית השרות מתחיל מהכתובת הנ"ל ומסתיים עד שהוא פוגש בפקודה RETI. הוראה זו מודיעה לבקר , שתוכנית השירות הסתיימה , הבקר מחזיר ל- PC את הערך המקורי מהמחסנית ומקדמת אותו ב-1.

* + אוגר IE :

אוגר זה הוא אוגר אפשור הפסיקות . כל פסיקה מוצגת כסיבית באוגר וכדי לאפשרה יש לעלות את הסיבית שלו ל-1 לוגי . הסיבית הראשונה היא סיבית כוללת , כאשר היא ב- '0' אף פסיקה אינה מאופשרת .

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| EX0 | ET0 | EX1 | ET1 | ES | ET2 | X | EA |

EA- סיבית לאפשור כללי של כל הפסיקות.

ET2- סיבית לאפשור פסיקה ממונה 2 (קיים רק ברכיב 8052)

ES- סיבית לאפשור פסיקה מהפורט הטורי.

ET1– סיבית אפשור פסיקה ממונה 1

EX1– סיבית אפשור פסיקה ממקור חיצוני INT1.

ET0- סיבית אפשור פסיקה של מונה 0.

EX0– סיבית אפשור פסיקה ממקור חיצוני INT0.

* + אוגר עדיפות הפסיקות IP :

הפסיקות מסודרות בסדר עדיפות מסוים וקבוע ע"י הבקר . כאשר שני פסיקות מתבקשות בו זמנית , יענה הבקר לזו שהיא בעלת עדיפות גבוהה יותר .

סדר העדיפויות הוא :

1. INT0
2. TIMER0
3. INT1
4. TIMER1.
5. UART.

ניתן לשנות עדיפות של פסיקה מסוימת ע"י אוגר IP.

מבנה האוגר:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| PX0 | PT0 | PX1 | PT1 | PS | PT2 | X | X |

PS–קביעת עדיפות לפורט הטורי.

PT1– קביעת עדיפות לפסיקת טיימר 1.

PX1– קביעת עדיפות ל –INT1.

PT0- קובעת עדיפות לפסיקת TIMER0.

PX0- קביעת עדיפות ל- INT1.

PT2- קביעת עדיפות לפסיקת TIMER 2 .

כדי לקבוע עדיפות גבוהה לפסיקה מסוימת יש לתת '1' בסיבית המתאימה באוגר וכל השאר '0'.

### 2.1.6 - הפורט הטורי Serial Port :

הפורט הטורי יכול לשדר ולקלוט לפי פורמט סטנדרטי (RS232 ) באופן אסינכרוני מילים בעלות מסגרת של עד 11 סיביות ב-4 אופני עבודה שונים .

הפורט הטורי יכול לפעול בתדרי שידור \קליטה סטנדרטים והוא נעזר בטיימר פנימי של הרכיב כדי לקבוע את קצב השידור מבלי להזדקק לשעון חיצוני.

כל הנתונים מועברים אל חוצץ טורי SBUF ,המורכב משני רגיסטרים הנמצאים באותה כתובת , אחד לשידור ושני לקליטה.

קצב השידור נקבע על ידי זמנן T1 לפי הטבלה הבאה:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Timer 1 | | | SMOD | תדר המתנד | קצב השידור |
| Reload Value | Mode | C/T |
| FDH | 2 | 0 | 1 | 11.059MHz | 19.2K |
| FDH | 2 | 0 | 0 | 11.059MHz | 9.6K |
| FAH | 2 | 0 | 0 | 11.059MHz | 4.8K |
| F4H | 2 | 0 | 0 | 11.059MHz | 2.4K |
| E8H | 2 | 0 | 0 | 11.059MHz | 1.2K |

קביעת קצב השידור

אופני עבודה של הפורט הטורי :

אופן העבודה נקבע ע"י סיביות SM0 ו- SM1 הנמצאות באוגר SCON.

אופן העבודה הנבחר קובע את הצורה בה יעבד המשדר\מקלט הטורי :

אופן 0 :

באופן עבודה זה ה-UART משמש כאוגר הזזה בעל קצב שידור וקליטה קבוע ל- 12\1 מתדר הגביש. רגל ה-RXD משמשת לשידור ולקליטה ורגל ה-TXD לתזמון.

אופן 1 :

באופן זה משודרות 10 סיביות, סיבית ראשונה START , לאחר מכן 8 סיביות המידע ולבסוף סיבית אחת שלSTOP .

קצב השידור והקליטה נקבעים שניהם ע"י T1.

אופן 2 :

באופן פעולה 2 מתוספת לרצף סיביות המידע סיבית נוספת הנקראת TB8 בשידור ו-RB8 בקליטה. לסיבית זו המשתמש יכול להכניס ערך כרצונו ובד"כ מכניסים לה את סיביות הזוגיות מ- PSW ואז היא משמשת כעין סיבית זוגיות לבדיקת שגיאות.

באופן פעולה 2 קצב השידור והקליטה נקבע ל- 32\1 או 64\1 מתדר הגביש.

אופן 3 :

אופן זה זהה לאופן 2 אלא שכאן ניתן לשנות את קצב השידור ע"י T1.

אוגר SCON:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| RI | T1 | RB8 | TB8 | REN | SM2 | SM1 | SM0 |

RI- סיבית זו עולה ל- '1' כאשר מילה שלמה נקלטה במקלט (כולל סיבית בקרה). סיבית זו מבקשת פסיקה ובסיום הפסיקה יש להוריד את הסיבית ל- '0' .

TI - סיבית זו עולה ל- '1' כאשר החוצץ הטורי סיים לשדר מילה שלמה. סיבית זו מבקשת פסיקה מהבקר כאשר בסיומה יש להוריד את הסיבית ל- '0' .

TB8 - סיבית זו היא הסיבית התשיעית בשידור, היא פעילה רק באופנים 2 ו- 2.

RB8- סיבית זו היא הסיבית התשיעית הנקלטת, היא פעילה רק באופנים 2 ו-2.

SM2 - סיבית זו מאפשרת את הפעולה בתקשורת מרובת מעבדים. באופנים 2 – ו 2. באופני עבודה אלה אם SM2 ב- '1' אזי סיבית ה-RI לא תתאפשר בקליטה אם הסיבית התשיעית שנקלטה (RB8) היא '0' . באופן עבודה 1 אם - SM2 ב- '1' אזי סיבית ה-RI לא תתאפשר בקליטה אם לא התקבלה סיבית STOP תקינה שנקלט בה. באופן עבודה 0 על סיבית ה-SM2 להיות '0'.

REN- סיבית זו מאפשרת/מונעת קליטת נתונים טוריים.

'1' – מאפשר

'0' – לא מאפשר.

|  |  |  |
| --- | --- | --- |
| SM0 | SM1 | אופן עבודה |
| 0 | 0 | אופן עבודה 0 |
| 1 | 0 | אופן עבודה 1 |
| 0 | 1 | אופן עבודה 2 |
| 1 | 1 | אופן עבודה 3 |

בחירת אופן עבודה של הפורט הטורי.

אוגר PCON :

אוגר זה שולט על ה- CLK הנכנס למעבד. בעזרתו ניתן להכניס את המעבד ל"תרדמה" לשם חיסכון בהספק . האוגר מכיל גם סיבית נוספת SMOD אשר גורמת לחלוקה של ה-CLK הנכנס למשדר הטורי ב- 2.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| IDL | PD | GF0 | GF1 | - | - | - | SMOD |

IDL- כאשר סיבית זו ב – ‘1’,המעבד נכנס למצב של תרדמה כאשר השעון מגיע רק לזמננים, לפסיקות ולפורט הטורי. מצב זה נחוץ כאשר רוצים לחסוך בהספק ובכל זאת שהמערכת תגיב לסביבה חיצונית. החזרה ממצב זה הוא ע"י פסיקה כלשהי וכך אין צורך לבצע RESET ולמחוק את תוכן האוגרים .

PD- סיבית זו משתקת כללית את השעון לבקר וכולו נמצא בתרדמת. החזרה ממצב זה הוא ע"י פעולתRESET חיצוני בלבד . ולכן יש לשמור על האוגרים ב- RAM החיצוני . פעולה זו חוסכת יותר הספק ממצב IDLE אך היא פחות נוחה .

GF0, GF1– סיביות אלו הם לשימוש כללי והן מורות לבקר שהתקבלה פסיקה במצב IDLE.

SMOD- סיבית זו קובעת חלוקה ב- 2 של תדר השידור של הפורט הטורי כאשר היא '0'.

### 2.1.7 - חיבור מעגל לביצוע RESET:

ביצוע ה- RESET יכול להתבצע ב-2 דרכים :

1. ע"י חומרה הכוללת מעגל פשוט ומפסק , אשר בלחיצה על המפסק מתבצע תהליך ה-RESET.
2. מיד בהפעלת המערכת.

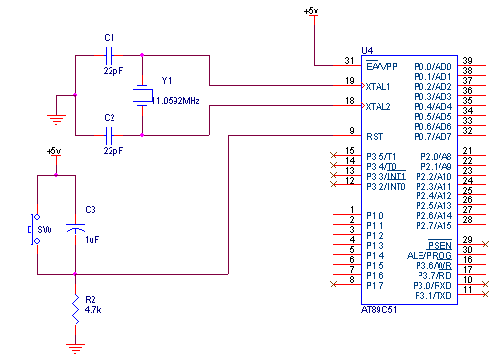
הסבר פעולת ה- RESET :

מיד עם הפעלת המתח , תימצא כניסת ה- reset ב- '1' ( מאחר ולא תתאפשר קפיצת מתח על הקבל ב- אפס זמן.

הערה: יש לבחור את הקבל כך שמשך הזמן בו הדק ה- RESET יהיה '1 ' הוא שני מחזורי מכונה לפחות.

ביצוע RESET גורם למיקרו בקר P89V51RD2 לקפוץ אוטומטית לכתובת קבועה של0000H, לכן רכיב המכיל את התוכנית ממופה החל מכתובת זו.

חיבור גביש ומעגל ה-Reset למיקרו בקרP89V51RD2:



## 2.2 - זיכרון התוכנית

המיקרו בקרP89V51RD2 מכיל בתוכו זיכרון תוכנית בגודל 64KB ולכן אין צורך בזיכרון תוכנית חיצוני. על מנת למנוע נזק לא מכוון לתוכנית ובכך להוציא את המערכת מכלל שימוש זיכרון התוכנית צריך להיות זיכרון לקריאה בלבד.

## 2.3- תצוגתLCD : ( (Liquid Crystal Display:

### 2.3.1 - תפקיד התצוגה במערכת:

תפקידה של התצוגה להציג תפריטים, תוצאות מדידה והודעות מנחות למשתמש.

### 2.3.2 - עיקרון פעולת התצוגה המבוססת על גביש נוזלי:

חומר מוצק זה חומר שבו המולקולות מסודרות בסדר מסוים ביניהם, לכן לחומר זה צורה מוגדרת משלו.

חומר נוזלי ידוע כחומר שבו אין למולקולות סדר כלשהו ולכן אין לחומר זה צורה מוגדרת והוא מקבל צורה של המכל המכיל אותו.

ידוע שגביש זה חומר מוצק בו המולקולות מסודרות לפי סדר מדויק, אך קיימים גם גבישים נוזליים.

גביש נוזלי הוא חומר נוזלי חסר צורה משלו, אך עדיין קיים סדר מסוים בו מסודרות המולקולות של חומר זה.

לגביש נוזלי יש תכונה מעניינת, בהימצאותו בשדה חשמלי הוא משנה את סדר המולקולות בתוכו וכך בעצם גורם לתופעה הבאה:

1) בהיעדר שדה חשמלי קרן האור העוברת דרכו

מסובבת ב-90 מעלות.

2) בהימצאות בתוך שדה חשמלי סדר המולקולות

משתנה וקרן האור העוברת דרכו יוצאת

באותה זווית (לא מסובבת).

על תכונה זו מבוססת תצוגת LCD .כל תא בתצוגה מורכב משתי זכוכיות מקוטבות המוצבות אחת מעל השנייה כך שאחת מכותבת בכיוון אופקי והשנייה בכיוון אנכי. בין הזכוכיות נמצא גביש נוזלי.

תא זה מתנהג בצורה הבאה:

1. בהיעדר שדה חשמלי קרן אור נכנסת דרך זכוכית ראשונה המקטבת בכיוון אופקי. קרן זו מתקטבת בכיוון אופקי, במעבר דרך גביש נוזלי מסתובבת ב-90 מעלות כך שבהגיעהלזכוכית השנייה קרן האור כבר מקוטבת בקיטוב אנכי והיא יוצאת דרך זכוכית שנייה (המקוטבת אנכית). מכאן שתא זה שקוף למעבר אור.
2. בהימצאות הגביש הנוזלי בשדה חשמלי קרן אור נכנסת דרך זכוכית ראשונה המקטבת בכיוון אופקי. קרן זו מתקטבת בכיוון אופקי, במעבר דרך גביש נוזלי הקרן יוצאת בדיוק כפי שנכנסה כך שבהגיעה לזכוכית השנייה קרן האור נחסמת על ידי הזכוכית שנייה המקוטבת אנכית. מכאן שתא זה חוסם את מעבר האור דרכו, כלומר הוא נראה שחור.

מהרכב של הרבה תאים מסוג זה ניתן לבנות תצוגה.

### 2.3.3 - תפקידי ההדקים של תצוגת LCD :



איור סכמתי של תצוגת LCD

Vss- רגל זו מתחברת לאדמה (GND), ומספקת את האדמה לרכיב.

Vdd- רגל זו מתחברת למקור +5V ומספקת את המתח לרכיב .

Vee- קו זה קובע את רמת הניגודיות של התצוגה ע"י חיבורו למקור מתח דרך פוטנציומטר. המצבים הקיצוניים הם:

1. כשהיא מחוברת ל- '0' מתקבלת הניגודיות הנמוכה ביותר.
2. כשהיא מחוברת ל-'1' מתקבלת הבהירות הגבוהה ביותר.

RS - כאשר רגל זו ב- '0' אז מתבצעת גישה אל אוגר ההוראות וכאשר היא ב- "1" אז מתבצעת גישה אל אוגר הנתונים.

R/W - רגל כתיבה\קריאה. כאשר רגל זו ב- '0' אז מתבצע מחזור כתיבה אל התצוגה וכשהיא היא ב- '1' אז מתבצע מחזור קריאה מהתצוגה.

E - רגל זו היא רגל האפשור (Enable) של התצוגה.

D0-D7 - שמונת קווי הקלט\פלט, המשמשים להעברת הנתונים מהמיקרו בקר אל התצוגה ולהפך. (בפרויקט זה השימוש שלהם הוא להעברת נתונים אל התצוגה בלבד).

### 2.3.4 - תכונות התצוגה :

1. צריכת הזרם של התצוגה נמוכה , סדר גודל של : 1mA-3mA.
2. שתי שורות, שבכל אחת מהן 20 תווים המורכבים ממטריצת נקודות 7X5 כפי שמתואר באיור הבא :

מבנה תו בתצוגה , ודוגמא להצגת תו – "A".

1. התצוגה כוללת זיכרון RAM של 8bit X80bytes .
2. התצוגה מתחברת למקור מתח יחיד.
3. מגוון הוראות הניתנות ע"י המערכת והידועות לה. הוראות מתבצעות ע"י שליחת קוד ההוראה לתצוגה (ההוראות והקודים של כל אחת מהן מתוארות בטבלה).

### 2.3.5 - זיכרון התצוגה :

לתצוגה 2 סוגי זיכרון- DDRAM ו-CGRAM .

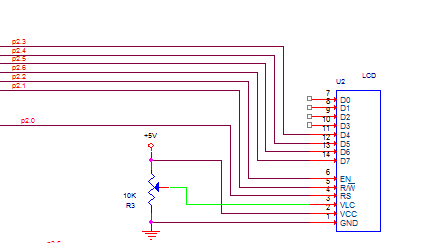
DDRAM- הוא זיכרון המיועד לאגור את כתובת התו אותו יש להציג התצוגה.

לזיכרון זה קיבולת של80bytes משמע הוא מסוגל להכיל 80 כתובות של תווים (40 כתובות עבור שורה ראשונה ו-40 כתובות עבור שורה שניה) לתצוגה.

מכיוון שכל שורה מכילה רק 16 תווים ניתן להזיז את חלון זה בן 16 תווים על פני מקטע בן 40 תווים האגור בזיכרון, דבר זה מאפשר יצירת תחושה של הודעות רצות.

CGRAM- הוא זיכרון המיועד לשמור תווים מיוחדים המיוצרים על ידי המשתמש. לזיכרון זה קיבולת של64bytes והוא יכול לאגור עד 8 תווים (8 בתים לכל תו).

אופן חיבור התצוגה בפרויקט:



אופן חיבור התצוגה בפרוייקט .

### 2.3.6 - המבנה הפנימי של התצוגה:

* + אוגרים :

ה- LCD כולל 2 אוגרים של 8 סיביות כל אחד .

האחד משמש כאוגר ההוראות IR והשני משמש כאוגר נתונים DR .

האוגר IR משמש לאחסון ההוראות לתצוגה כגון : ניקויי התצוגה, כיוון הזזת הסמן, הגדרת הסמן וכו'.

האוגר ה–DR הוא אוגר המשמש לאחסון זמני של נתונים הנכתבים ל- DDRAM , ל- CGRAM וכמו כן גם נתונים זמניים הנקראים החוצה מה- CGRAM ומה-DDRAM.

המידע אשר נשלח לתוך ה- DR על ידי המיקרו בקר נכתב אוטומטית לתוך ה- DDRAM או ה- CGRAM ע"י פעולה פנימית של בקר התצוגה.

בחירת האוגר מתבצעת על ידי רגל ה-RSREGISTER SELECTOR)).

* + BUSY FLAG

בכדי לדעת את מצבה של התצוגה ניתן לקרוא ממנה את BUSY FLAG :

דגל ה- BUSY ב- '1' מסמל ש- LCD במצב של פעולה פנימית ובמקרה זה ההוראה הבאה לא תתקבל.

* + מונה הכתובות AC-ADDRESS COUNTER :

תפקידו של AC הוא לספק את כתובת התא של ה-DDRAM וה–CGRAM אליו מתבצעת הגישה.

* + הוראות בקרה (INSTRUCTION TABLE) :

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| הקוד | | | | | | | | | | ההוראה |
| DB0 | DB1 | DB2 | DB3 | DB4 | DB5 | DB6 | DB7 | R\W | RS |  |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ניקוי מסך |
| X | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | החזרת סמן |
| S | I\D | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | קביעת מצב עבודה |
| B | C | D | 1 | 0 | 0 | 0 | 0 | 0 | 0 | בקרת תצוגה |
| X | X | R\L | S\C | 1 | 0 | 0 | 0 | 0 | 0 | הזזת סמן\תצוגה |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| X | X | F | N | DL | 1 | 0 | 0 | 0 | 0 | קביעת אופן עבודה |
| ACG | | | | | | 1 | 0 | 0 | 0 | קביעת CGRAM |
| ADD | | | | | | | 1 | 0 | 0 | קביעת DDRAM |
| AC | | | | | | | BF | 1 | 0 | קריאת ADD\BF |
| WRITE DATA | | | | | | | | 0 | 1 | כתיבה ל-CG\DDRAM |
| READ DATA | | | | | | | | 1 | 1 | קריאה מ-CG\DDRAM |

הוראות הבקרה של LCD .

הסבר סימני הטבלה :

'1' = I\D– מגדיל ב- 1. '0' = I\D - מקטין ב 1.

'1' = S– יש הזזת תצוגה. '0' = S– אין הזזת תצוגה.

'1' = D– תצוגה דלוקה. '0' = D– תצוגה כבויה.

'1' = C– סמן מופיע. '0' = C– סמן אינו מופיע.

'1' = B– סמן מהבהב. '0' = B– סמן אינו מהבהב.

'1' = S\C– הזזת תצוגה. '0' = S\C– הזזת סמן.

'1' = BF– מבצע פעולה פנימית. '0' = BF– מוכן לקליטת הוראות.

'1' = R\L– הזזה ימינה. '0'= R\L–הזזה שמאלה.

'1' = DL– מספר קווי הנתונים - 8. '0' = DL– מספר קווי הנתונים - 4.

'1' = N– מספר שורות בתצוגה - 2. '0' = N– מספר שורות בתצוגה - 1.

'1' = F– מידות הגופן –5x7 נקודות. '0' = F– מידות הגופן –5x10 נקודות.

DDRAM– זיכרון נתוני התצוגה. CG RAM – זיכרון מחולל תווים.

ACG- כתובת ה- CG RAM.

ADD– כתובת ה- DD RAM.

AC -מונה כתובות ל- CG\DD RAM.

### 2.3.7 - פרוט הוראות הבקרה :

1. איפוס תצוגה (CLEAR DISPLAY ):

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DB0 | DB1 | DB2 | DB3 | DB4 | DB5 | DB6 | DB7 | R\W | RS |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

איפוס התצוגה נעשה ע"י שליחת קוד הרווח 20H לתוך כל הכתובות ב- DDRAM. הסמן חוזר לכתובת ההתחלתית (תחילת שורה ראשונה ).

1. החזרת הסמן (RETURN HOME ) :

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DB0 | DB1 | DB2 | DB3 | DB4 | DB5 | DB6 | DB7 | R\W | RS |
| X | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

הכתובת ההתחלתית של ה- DDRAM מאוחסנת במונה הכתובות (AC), פקודה זו אינה משבשת את תוכן התצוגה.

הסמן חוזר לתחילת השורה הראשונה בצידה השמאלי.

1. קביעת מצב עבודה :

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DB0 | DB1 | DB2 | DB3 | DB4 | DB5 | DB6 | DB7 | R\W | RS |
| S | I\D | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

שליחת פקודה זו קובעת את כיוון תזוזת הסמן ותזוזת התצוגה.

1. בקרת תצוגה :

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DB0 | DB1 | DB2 | DB3 | DB4 | DB5 | DB6 | DB7 | R\W | RS |
| B | C | D | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

בקרת תצוגה ON\OFF גורמת לכך שהתצוגה תיכבה או תידלק , כמו כן גורמת לסמן להופיע ולהבהב.

כאשר D ='1' התצוגה במצב ON.

כאשר D = '0' התצוגה במצב OFF ,נתון להצגה נשאר ב- DDRAM (זיכרון התצוגה).

כאשר C ='1' – הסמן מוצג על התצוגה ובמצב ההפוך הסמן לא מופיע .

כאשר B = '1' התו שבמקום הסמן יחל להבהב (B = '0' הסמן לא יהבהב).

בזמן ההבהוב כל הנקודות המרכיבות את התו ממותגות בערך כל 0.4sec .

1. הזזת הסמן\תצוגה :

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DB0 | DB1 | DB2 | DB3 | DB4 | DB5 | DB6 | DB7 | R\W | RS |
| X | X | R\L | S\C | 1 | 0 | 0 | 0 | 0 | 0 |

הזזת סמן התצוגה נעשית מבלי לשנות את תוכן זיכרון התצוגה DD RAM.

|  |  |  |
| --- | --- | --- |
| R\L | S\C | פרוט פעולה |
| 0 | 0 | הסמן זז שמאלה ( מונה כתובות מוקטן ב-1 ) |
| 1 | 0 | הסמן זז ימינה (מונה כתובות מקודם ב- 1) |
| 0 | 1 | התצוגה בשלמותה זזה שמאלה יחד עם הסמן. |
| 1 | 1 | התצוגה בשלמותה זזה ימינה יחד עם הסמן. |

בחירת אופן הזזת סמן \תצוגה .

1. קביעת אופן עבודה :

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DB0 | DB1 | DB2 | DB3 | DB4 | DB5 | DB6 | DB7 | R\W | RS |
| X | X | F | N | DL | 1 | 0 | 0 | 0 | 0 |

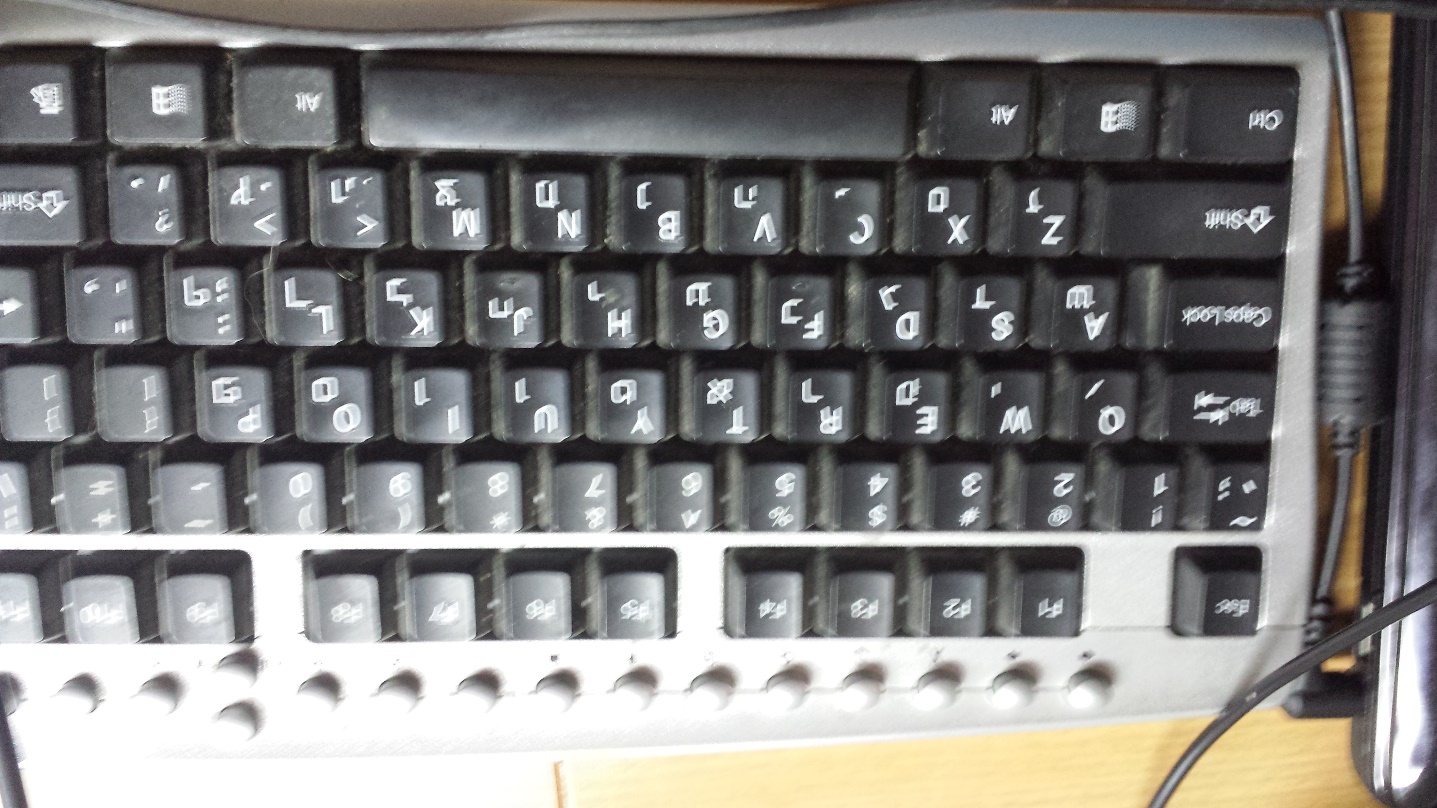
על הפקודה להישלח ראשונה לתצוגה.

כאשר DL = '1' – מספר קווי הנתונים נקבע ל- 8 קווים.

כאשר DL = '0' – מספר קווי הנתונים נקבע ל- 4 קווים.

ארבעת הסיביות העליונות משודרות קודם ואחריהן 4 הסיביות התחתונות .

## 2.4–מקלדת מחשב בחיבורPS2

****

מקלדת מחשב (מכונה בקיצור מקלדת או לוח מקשים) היא [מקלדת](http://he.wikipedia.org/wiki/%D7%9E%D7%A7%D7%9C%D7%93%D7%AA_(%D7%9B%D7%AA%D7%99%D7%91%D7%94)) (מערך קלידים) המשמשת כהתקן [קלט](http://he.wikipedia.org/wiki/%D7%A7%D7%9C%D7%98) להזנת [תמליל](http://he.wikipedia.org/wiki/%D7%98%D7%A7%D7%A1%D7%98) (טקסט) אל ה[מחשב](http://he.wikipedia.org/wiki/%D7%9E%D7%97%D7%A9%D7%91). במקלדת סטנדרטית יש כ-100 מקשים שונים. יש מקשים עבור סימני [אותיות](http://he.wikipedia.org/wiki/%D7%90%D7%95%D7%AA), [ספרות](http://he.wikipedia.org/wiki/%D7%A1%D7%A4%D7%A8%D7%94), [פיסוק](http://he.wikipedia.org/wiki/%D7%A1%D7%99%D7%9E%D7%A0%D7%99_%D7%A4%D7%99%D7%A1%D7%95%D7%A7), סימנים אחרים (כמו "@" וכמו "$") ומקשי ניהול (כמו [מקש ה-Ctrl](http://he.wikipedia.org/wiki/%D7%9E%D7%A7%D7%A9_Ctrl), [טאב](http://he.wikipedia.org/wiki/%D7%98%D7%90%D7%91) ו[מקש Fn](http://he.wikipedia.org/wiki/%D7%9E%D7%A7%D7%A9_Fn)).

מקלדות אינן מעבירות קוד [ASCII](http://he.wikipedia.org/wiki/ASCII) (קוד הממפה בין תווים לבין ייצוגם הפנימי במחשב), אלא קוד [בינארי](http://he.wikipedia.org/wiki/%D7%91%D7%99%D7%A0%D7%90%D7%A8%D7%99) השונה ממקש למקש, ואינו תואם לקוד הבינארי של תווים במיפוי ASCII. הדבר מאפשר בין היתר קיומם של מקשים מיוחדים (למשל F1), שאינם קיימים במיפוי ה-ASCII. כך, כאשר המשתמש מקיש במקלדת על המקש, המייצג את האות העברית [ש](http://he.wikipedia.org/wiki/%D7%A9) וכן את האות האנגלית [A](http://he.wikipedia.org/wiki/A) (המתאים גם לייצוגה כאות קטנה - a), המקלדת שולחת למערכת ההפעלה את הקוד המתאים למקש - במקרה זה 30. פירוש הקוד בהתאם להקשר, הוא תפקידה של התוכנה, שנעזרת לשם הפענוח בהודעות נוספות שנשלחות מהמקלדת, על הסטטוס של מקשי פונקציות כמו Shift, Altוכו'.

עבור כל הקשה המקלדת שולחת קוד פעמיים, פעם אחת עבור לחיצה על מקש ופעם שנייה עבור שחרור המקש, מאחר שתיתכן חשיבות לתזמון שחרור המקש ביחס לפונקציונליות הרצויה. בפרט, ייתכן כי סדר לחיצת המקשים לא יהיה זהה לסדר השחרור שלהם.

### מערך קודי מקלדת

|  |  |  |  |
| --- | --- | --- | --- |
| לחיצה על מקש | מקש+shift | קוד לחיצה | קוד שחרור |
| ` | ~ | 0E | F0 0E |
| 1 | ! | 16 | F0 16 |
| 2 | @ | 1E | F0 1E |
| 3 | # | 26 | F0 26 |
| 4 | $ | 25 | F0 25 |
| 5 | % | 2E | F0 2E |
| 6 | ^ | 36 | F0 36 |
| 7 | & | 3D | F0 3D |
| 8 | \* | 3E | F0 3E |
| 9 | ) | 46 | F0 46 |
| 0 | ( | 45 | F0 45 |
| - | \_ | 4E | F0 4E |
| = | + | 55 | F0 55 |
| backspace |  | 66 | F0 66 |
| tab |  | 0D | F0 0D |
| q | Q | 15 | F0 15 |
| w | W | 1D | F0 1D |
| e | E | 24 | F0 24 |
| r | R | 2D | F0 2D |
| t | T | 2C | F0 2C |
| y | Y | 35 | F0 7E |
| u | U | 3C | F0 3C |
| i | I | 43 | F0 43 |
| o | O | 44 | F0 44 |
| p | P | 4D | F0 4D |
| ] | } | 54 | F0 54 |
| [ | { | 5B | F0 5B |
| capslock |  | 58 | F0 58 |
| a | A | 1C | F0 1C |
| s | S | 1B | F0 1B |
| d | D | 23 | F0 23 |
| f | F | 2B | F0 2B |
| g | G | 34 | F0 34 |
| h | H | 33 | F0 33 |
| j | J | 3B | F0 3B |
| k | K | 42 | F0 42 |
| l | L | 4B | F0 4B |
| ; | : | 4C | F0 4C |
| ' | " | 52 | F0 52 |
| enter | enter | 5A | F0 5A |
| Left shift |  | 12 | F0 12 |
| z | Z | 1A | F0 1A |
| x | X | 22 | F0 22 |
| c | C | 21 | F0 21 |
| v | V | 2A | F0 2A |
| b | B | 32 | F0 32 |
| n | N | 31 | F0 31 |
| m | M | 3A | F0 3A |
| , | > | 41 | F0 41 |
| . | < | 49 | F0 49 |
| / | ? | 4A | E0 F0 4A |
| Right shift |  | 59 | E0 F0 59 |
| Left ctrl |  | 14 | E0 F0 14 |
| Left alt |  | 11 | E0 F0 11 |
| Spacebar |  | 29 | E0 F0 29 |
| Right alt |  | E0 11 | E0 F0 7E |
| Left ctrl |  | E0 14 | E0 F0 7E |
| insert |  | E0 70 | E0 F0 7E |
| delete |  | E0 71 | E0 F0 7E |
| Left arrow |  | E0 6B | E0 F0 7E |
| home |  | E0 6C | E0 F0 7E |
| end |  | E0 69 | E0 F0 7E |
| Up arrow |  | E0 75 | E0 F0 7E |
| Down arrrow |  | E0 72 | E0 F0 7E |
| Page up |  | E0 7D | E0 F0 7E |
| Page down |  | E0 7A | E0 F0 7E |
| Right arrow |  | E0 74 | E0 F0 7E |
| Num lock |  | 77 | F0 77 |
| Keypad 7 |  | 6C | F0 6C |
| Keypad 4 |  | 6B | F0 6B |
| Keypad 1 |  | 69 | F0 69 |
| Keypad / |  | E0 4A | E0 F0 4A |
| Keypad 8 |  | 75 | F0 75 |
| Keypad 5 |  | 73 | F0 73 |
| Keypad 2 |  | 72 | F0 72 |
| Keypad 0 |  | 70 | F0 70 |
| Keypad \* |  | 7C | F0 7C |
| Keypad 9 |  | 7D | F0 7D |
| Keypad 6 |  | 74 | F0 74 |
| Keypad 3 |  | 7A | F0 7A |
| Keypad . |  | 71 | F0 71 |
| Keypad - |  | 7B | F0 7B |
| Keypad + |  | 79 | F0 79 |
| Keypad enter |  | E0 5A | E0 F0 5A |
| esc |  | 76 | F0 76 |
| F1 |  | 05 | F0 05 |
| F2 |  | 06 | F0 06 |
| F3 |  | 04 | F0 04 |
| F4 |  | 0C | F0 0C |
| F5 |  | 03 | F0 03 |
| F6 |  | 0B | F0 0B |
| F7 |  | 83 | F0 83 |
| F8 |  | 0A | F0 0A |
| F9 |  | 01 | F0 01 |
| F10 |  | 09 | F0 09 |
| F11 |  | 78 | F0 78 |
| F12 |  | 07 | F0 07 |
| Print screen |  | E0 12 E0 7C F0 12 | E0 F0 7C E0 |
| Scroll lock |  | 7E | F0 7E |
| Pause break |  | E1 14 77 E1 | F0 14 F0 77 |
| \ | | | 5D | F0 5D |

### 2.4.1 - פרוטוקול PS2

פרוטוקול PS2(אוסף חוקים לניהול שידור בין שני רכיבים המחוברים ע"י מחבר PS2) שימושי במקלדות ועכברים של מחשבי PC.

פרוטוקול התשדורת PS2 הוא טורי סינכרוני. כלומר, מתבצעת העברה של המידע באמצעות האות (KBD\_DATA) ביט אחרי ביט ובאופן מתואם לאות שעון (KBD\_CLK).

אות המידע KBD\_DATA מופק במקלדת ומועבר לאחת הכניסות של המיקרו-בקר.

האות KBD\_CLK,אות השעון המופק במקלדת ומועבר לאחת הכניסות של המיקרו-בקר.

Keyboard

Microcontroller

1

4

5

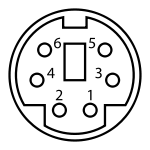
3

**תיאור הממשק בין המקלדת למיקרו בקר**

|  |  |
| --- | --- |
| Pin on practice | Signal name |
| P3.0 | KBD\_CLK |
| P3.1 | KBD\_DATA |

קווי המיקרו בקר והמקלדת

המחבר שבו נעשה שימוש הוא מחבר בעל שישה הדקים (6 pin mini – DIN PS/2 style connector)

[](http://en.wikipedia.org/wiki/File:MiniDIN-6_Connector_Pinout.svg)

**תיאור מחבר PS2**

להלן התפקיד של ההדקים השונים במחבר:

|  |  |
| --- | --- |
| **Function** | **Pin on connector** |
| KBD\_DATA | 1 |
| Not connected | 2 |
| GND | 3 |
| VCC | 4 |
| KBD\_CLK | 5 |
| Not connected | 6 |

**תיאור הפינים של מחבר PS2**

### מבוא ונתונים על הממשק בין המקלדת PS/2 ו-host ברמת ה-bit

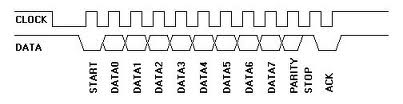
פרוטוקול PS/2 הוא פרוטוקול טורי סינכרוני, כלומר מתבצעת העברה של המידע (באמצעות האות KBD\_DATA) ביט אחרי ביט ובאופן מתואם לשעון (KBD\_CLK).

במצב מנוחה שבוא אין תשדורת המצב הלוגי של KBD\_DATA הוא '1' לוגי. התשדורת מתחילה בסיבית ההתחלה (start bit) שהיא תמיד '0' לוגי. לאחר מכן משודר רצף של שמונה סיביות החל מסיבית LSB עד לסיבית ה- MSB.

לאחר שידור רצף הסיביות משודרת סיבית הזוגיות (parity), שצריכה ליצור מצב שבו מספר האחדים בתשדורת (כולל סיבית הזוגיות עצמה) הוא תמיד אי זוגי (odd parity).

לאחר סיבית הזוגיות משודרת סיבית הסיום (stop bit), שהיא תמיד '1' לוגי. בגמר פעולות אלו המקלדת חוזרת למצב מנוחה, עד לתשדורת הבאה.

האיור הבא מציג דיאגרמת זמנים שמתארת את צורת התשדורת של המידע יחד עם אות השעון של המקלדת :



**דיאגרמת זמנים פרוטוקול PS2**

ניתן לראות מדיאגרמת הזמנים שהמידע אינו יציב בזמן שאות השעון נמצא ב-'1' לכן את קריאת המידע יש לבצע כאשר אות השעון נמצא ב-'0' לוגי.

## 2.5- חיישן צבע TCS3200

ה-TCS3200 ממיר מתוכנת צבע אור לתדרים.הממיר במעגל משולב יחיד, מונוליטי ממשפחת ה- CMOS.המוצא הוא גל ריבועים (50% המחזור) בתדירות יחסית לעוצמת אור (קרינה).  
את תדירות המוצא ניתן לבחור מאחד משלושה ערכים קבועים מראש באמצעות שני הדקים של כניסת בקרה. כניסות דיגיטליות ויציאה דיגיטלית מאפשרים ממשק ישיר למיקרו -בקר. הדק (OE) מאפשר חיבור הדק המבוא של המיקרו אל כניסות/יציאות נוספות וזאת כאשר המוצא בעכבה גבוהה. ממיר האור לתדר קוראמערך של 8 × 8 פוטו-דיודות. לשש עשר פוטו-דיודות יש מסננים כחולים, 16 פוטו-דיודות יש מסננים ירוקים, 16 פוטו-דיודות יש מסננים אדומים, ו- 16 פוטו-דיודות ברורות ללא מסננים. ארבעת הסוגים (צבעים) של פוטו-דיודות הם interdigitated כדי למזער את ההשפעה של הלא אחידות של הקרינה. כל פוטו-הדיודות של אותו הצבע מחוברות במקביל. הדקיםS2S3 משמשים כדי לבחור איזו קבוצת פוטו-דיודות (אדום, ירוק, כחול, ברור) הם פעילים. פוטו-הדיודות הן בגודל 110 × 110 מיקרומטר.



סכימת מלבנים חיישן צבע TCS3200

**תיאור הדקים**

|  |  |  |  |
| --- | --- | --- | --- |
| **שם** | **כניסה / יציאה** | **תפקיד** | **חיבור למיקרו** |
| **GND** |  | **הדק מתח אספקה שלילי** |  |
| **OE** | **כניסה** | **אפשור המוצא** | **P3.6** |
| **OUT** | **יציאה** | **מוצא החיישן** | **P3.5** |
| **S0,S1** | **כניסה** | **דירוג תדר אות המוצא** | **P3.2,P3.3** |
| **S2,S3** | **כניסה** | **בחירת סוג מסנן** | **P3.4,P3.7** |
| **VDD** |  | **הדק מתח אספקה חיובי** |  |

**אפשרויות בחירה**

|  |  |  |
| --- | --- | --- |
| **דרגת תדר המוצא** | **S1** | **S0** |
| **צריכת הספק נמוכה- המתנה** | **0** | **0** |
| **2%** | **1** | **0** |
| **20%** | **0** | **1** |
| **100%** | **1** | **1** |

|  |  |  |
| --- | --- | --- |
| **סוג מסנן** | **S3** | **S2** |
| **אדום** | **0** | **0** |
| **כחול** | **1** | **0** |
| **ללא** | **0** | **1** |
| **ירוק** | **1** | **1** |

**תיאור אפשרויות בחירת מסנן ותדירות**

**חיבור הדק המוצא**המוצאמיועד לדחוף רכיב TTL רגיל או CMOSבמרחקים קצרים. אם אורך הקוויםיותר מ- 12 סנטימטרים,מומלץלהוסיף חוצץ או דוחף.  
רמה גבוהה בהדק (OE) גורמת למוצא להימצא בעכבה גבוהה על מנת לאפשר שיתוף יחידות נוספות לקוהקלט של המיקרו-בקר.

**POWER DOWN**כיבוי החיישן באמצעותS0/S1 (L / L) יגרום למוצאלהימצא בעכבה גבוהה. בדומה להדק אפשור OE, עם זאת כיבוי החיישן גורם לחיסכון משמעותי בצריכת הספק.

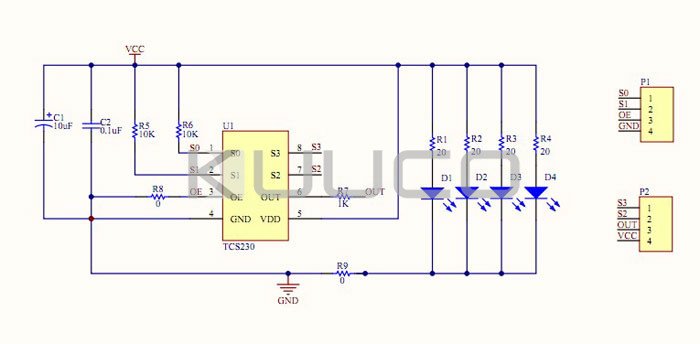
**בחירת סוג מסנן הצבע**הסוג של photodiode (כחול, ירוק, אדום או שקוף) המשמש את המכשיר נשלט על ידי שתי כניסות הלוגיות, S2 ו- S3 (ראה טבלה 1).

**דרוג תדר המוצא**

דרוג התדר נשלט על ידי שתי כניסות לוגיות, S0 ו- S1.הממיר הפנימי של האור לתדריוצר סדרה שלפולסיםברוחב פולס קבוע. הדירוג מושג על ידי חיבור סדרתהדפקים של הממיר לסדרה של מחלקי תדר. התוצאה היא גלים מרובעים עם מחזור פעולה 50% עם ערכים יחסיים בתדר של 100%, 20%, ו 2%. בגלל חלוקת תדר המוצאנוצר על ידי ספירת פולסים של התדר הפנימי העיקרי, המחזור האחרוןבמוצא מייצג ממוצע המחזורים המרובים של תדירות העקרונית.

**מדידת התדר**הבחירה של טכניקת החיבור והמדידה תלויה במידת הרזולוציה ושיעור הנתונים הרצויים. מדידת תדר המוצא ע"י מדידת משך הזמן בין שני שינויים סמוכים באות או לחילופין ספירת פולסים במשך זמן קבוע.

**להלן תיאור המעגל החשמלי של מודול החיישן**

****[](http://www.google.co.il/imgres?q=tcs230+color+sensor&hl=iw&sa=X&tbo=d&biw=1366&bih=673&tbm=isch&tbnid=nmH4JRlduanHsM:&imgrefurl=http://www.ebay.com/itm/Color-Sensor-Color-Recognition-Module-TCS230-TCS3200-/190558321419&docid=devHhuqtlRzM3M&imgurl=http://i744.photobucket.com/albums/xx86/egochina/DSC06314.jpg&w=600&h=462&ei=QOoDUeXRKIKn4ATx0ICQCw&zoom=1&ved=1t:3588,r:73,s:0,i:306&iact=rc&dur=3482&sig=105204834840748476923&page=2&tbnh=175&tbnw=221&start=17&ndsp=25&tx=72&ty=135)

**תרשים מעגל מודול חיישן הצבע**

### אופן שימוש הרכיב בפרוייקט

סביב חיישן הצבע מחוברים 4 פינים שתפקידם להחזיק את פתקיות הצבע במרחק קבוע מעל הרכיב.

הדק הOUT מחובר ל-p3.5 שהוא הדקו של timer1 שפועל כמונה timer0 פועל כקוצב זמן כאשר timer0 סיים את ספירתו נדגמים הדפקים שהתקבלו במוצא החיישן .מדידה זו מתרחשת באופן סדרתי כל עוד לא נלחץ מקש במקלדת.

מדדתי דפקים עבור צבע נתון שלוש פעמים כאשר בכל פעם פעילה דיודה אחרת (כחול, ירוק אדום).

מנתונים אלו מחושבים לאחר מכן היחסים בין הצבעים לפי הנתונים שהתקבלו ומוצגים על הLCD במהלך העבודה התגלה שעקב שינוי בתנאי המיקום והתאורה מתקבלים יחסים שונים לכן לאחר מדידות רבות נקבע רף תחתי ועילי לכל אחד מהיחסים וכאשר נמדדים נתונים ביניהם מזוהה הצבע ומושמע ברכיב הקול.

אם ערכי הצבע שנמדד נמוך מערכי הסף משמעות הדבר היא שלא נמדד אף צבע ולכן תימשך סריקת המקלדת ללא הצגת נתונים.

טבלת רף תחתי ועילי שנקבע לצבעים (יחסים בין הדפקים שנספרו)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| צבע | שם באנגלית | R/B | | R/G | |
|  |  | רף על' | רף תח' | רף על' | רף תח' |
| כחול | blue | 0.8 | 0.76 | 0.88 | 0.83 |
| ורוד | pink | 1.08 | 1.03 | 1.23 | 1.18 |
| סגול | purple | 0.89 | 0.86 | 1.03 | 1.00 |
| כתום | orange | 1.63 | 1.52 | 1.55 | 1.46 |
| צהוב | yellow | 1.26 | 1.2 | 0.96 | 0.89 |
| לבן | white | 0.93 | 0.89 | 0.98 | 0.95 |

|  |  |  |
| --- | --- | --- |
| R | G | B |
| 80 | 80 | 80 |

ערכי הסף שנקבעו לזיהוי צבע (בדפקים)

### תרשים זרימה למדידת תדר מוצא החיישן בשיטת ספירת פולסים בזמן קצוב

התחל

העתק תוצאה מטיימר 1 ועצור טיימרים

לא

סיום

האם טיימר 0 סיים ?

אפס משתנה תוצאה

אפס טיימרים 0 ו- 1

תכנת טיימר 1 כמונה 16 סיביות

הפעל טיימרים

תכנת טיימר 0 כקוצב זמן 16 סיביות

כן

## 2.6- רכיב דיבור 4004ISD

### 2.6.1 - מבוא:

רכיב זה של חברת ISD מקנה פתרון יעיל להקלטות איכותיות של הודעות באורך של עד 16 דקות. הרכיב מתאים לטלפונים סלולאריים, למשיבונים , למרכזיות ועוד.

הרכיב בנוי בטכנולוגית cmos ומכיל בתוכו מתנד, שני פילטרים, מערך "השתקה אוטומטית", מגבר שמע וזיכרון FLASH.

הרכיב מיועד לעבודה עם מערכות מבוססות מיקרו פרוססורים ומיקרו בקרים . באמצעות כתובות ובקרה הנשלחים ע"י ממשק טורי חיצוני –SPI–Serial Peripheral Interface , יש לנו את היכולת לשלוט באופן מלא ופשוט יחסי בכל הפונקציות והאפשרויות שהרכיב מקנה לנו. ההודעות נשמרות על תאי הזיכרון הבלתי נדיפים, כך שהמידע נשמר גם לאחר כיבוי המתח(היצרן מתחייב לשמירת המידע בתקופת זמן של עד כ-100 שנה ללא מקור מתח).

אותות קול ושמע נשמרים בצורה "אנלוגית" ישירות בזיכרון בצורתם המקורית, טכנולוגיה זו מניבה שחזור קול יציב ובאיכות גבוהה.

### 2.6.2 - הדקי הרכיב:



**תיאור סכמתי של רכיב הדיבור ISD4004**

### 2.6.3 - תכונות הרכיב

הרכיבISD4004 הינו רכיב דיבור שבעזרתו ניתן להקליט ולהשמיע הודעות באיכות גבוהה. משך זמן ההקלטה הוא בין 8 ל 16 דקות (תלוי בגרסה הרשומה על הרכיב). בין הרכיב למעבד יש תקשורת טורית, באמצעותה מקליטים או משמיעים את ההודעות המוקלטות.

הרכיב עובד עם מתח של V3.

הרכיב שייך למשפחת CMOS והוא כולל בתוכו מתנד, מסננים, מערך השתקה אוטומטי, מגבר שמע, ומערך זיכרון FLASH רב דרגתי.

הרכיב תוכנן לעבוד עם מיקרופרוססור ומיקרו בקרים בתקשורת טורית סינכרונית הנקראת SPI - Serial Peripheral Interface – מישק היקפי טורי או בתקשורת הנקראת MSI - Microwire Serial Interface– מישק היקפי חוט זעיר. התקשורת הטורית מפחיתה את כמות הרגליים של הרכיב.

ההקלטות נשמרות ברכיב ללא צורך בהספקת מתח מכיוון שהרכיב משתמש בזיכרון לא נדיף. כך אנו יכולים לשמור הודעות על הרכיב בלי לחשוש שהן ייעלמו עם הפסקת המתח לרכיב. היצרן מבטיח שהמידע יישמר ברכיב בתקופת זמן של עד 100 שנה ללא מקור מתח.

תצרוכת הזרם של הרכיב נמוכה. הזרם הנצרך מהספק הוא 15 ma בזמן השמעה ו 25 ma בזמן הקלטה. בזמן stand by כ 1 מיקרו אמפר.

100000 אפשרויות הקלטה .

איכות השמע של הרכיב היא טבעית לחלוטין מכיוון משהמידע אינו נדחס, האחסון האנלוגי שלו הוא ישיר ולכן הוא מאפשר שחזור צלילים, קולות, טונים ואפקטים בצורה טבעית.

ברכיב ישנן 2 מערכות העובדות במשולב.

מערכת אנלוגית האחראית על אות הדיבור המוקלט ועל אות הדיבור שיוצא .

מערכת דיגיטאלית הדוגמת את האות ושומרת אותו במערך הזיכרון הבלתי נדיף.

סידרת הרכיבים ISD 4004 שייכת לרכיבי הקלטה/השמעה לזמנים של 8, 10 , 12, 16 דקות. לכולם אותו נפח זיכרון אבל תדר הדגימה והמסננות בתוך הרכיב שונות.

הסיומת שנמצאת בהדפסה שעל הרכיב קובעת את זמן ההקלטה/השמעה כנראה בטבלה הבאה:



**מאפייני הסדרה של הרכיב ISD4004**

ניתן לראות שההבדל בין הרכיבים הוא תדר הדגימה שבו דוגמים את הקול ותדר המסננות הפנימיות המעבירות את אות השמע להקלטה. ניתן להבין שברכיב ל 8 דקות הקלטה/השמעה , איכות הקול תהיה הטובה מבין 4 הרכיבים שבמשפחה כי תדר הדגימה הוא הגבוה ביותר.

**סכמת מלבנית**



**סכימת המלבנים ISD4004**

### 2.6.4 - הסבר דיאגרמת המלבנים

בצד שמאל למעלה רואים את מתנד השעון הפנימי –INTERNAL CLOCK. היצרן ממליץ לחבר את רגל XCLK לאדמה ואז השעון הפנימי הוא המפעיל את המערכת (הסבר נוסף בתיאור רגלי הרכיב שבהמשך).

פולסי השעון מגיעים למערכת התזמון TIMING . זוהי המערכת המתזמנת את כל פעולות הרכיב.

ממנה יוצאים פולסי דגימת שעון –SAMPLING CLOCK אשר דוגמים את האות האנלוגי המגיע מהמשדרים /מקלטים האנלוגיים -ANALOG TRANSCEIVERS.

המקלט האנלוגי הוא מערכת ההגברה האנלוגית והמסננות לאות השמע הנכנס מהמיקרופון ונשמר במערך התאים הלא נדיף.

המשדר האנלוגי הוא המערכת המוציאה את הבתים ממערך הזיכרון אל מגבר היציאה דרך מסננות היציאה.

מערך הזיכרון של 3840 קילו תאים נקרא :

-MULTILEVEL STORAGE ARRAY 3840K CELL NONVOLATILE .

זהו מערך של תאים המסודר בתוך 1200 שורות . השמירה היא אנלוגית על פי פטנט של יצרן הרכיב. הפנייה לכל שורה מתבצעת בעזרת מפענח DECODER .

המפענח דואג לפנות לתא המסוים בשורה הנכונה ולשמור שם את הדגימה הנקלטת או להוציא את הדגימה מהתא שבשורה הנכונה החוצה.

אות השמע נכנס למגבר שרת –AMP . הכניסות הפרשיות כדי לבטל השפעה של רעש. ( כזכור רעש "מתלבש" על שתי הכניסות באותו מופע ולכן ההפרש שלהם יהיה אפס).

האות מהמגבר עובר אל מסננת - POLEACTIVE ANTIALIASING FILTER5. זוהי מסננת אקטיבית הדואגת לכניסה של תדרי דיבור בלבד למערכת. וחוסמת תדרים מעל או מתחת תדרים אלו.

אות הדיבור נדגם ע"י מערכת הדגימה ונשמר בתא כלשהו בשורה מסוימת הנקבעים ע"י המפענח. המיקום – השורה – שבה נשמר האות בזמן ההקלטה נקבע בעזרת הפקודות נשלחות אל הרכיב.

כאשר נרצה לשמוע את ההקלטות, נשלח פקודת הוצאה משורה מסוימת ואז האות שנשמר בתאים יוצא אל מסננת החלקה אקטיבית בעלת 5 קטבים - POLEACTIVE SMOOTHING FILTER5

שתפקידה לדאוג "להחליק" את הדגימות הבדידות שיוצאות מהזיכרון וכמו כן לדאוג שרק תדר הדיבור ולא תדר הדגימה או רעשים יעברו לכיוון היציאה.

האות המשוחזר עובר דרך מערכת השתקה אוטומטית - AUTOMUTE FEATURE . מערכת זו דואגת שרק אות מעל רמה מסוימת יצא מהמערכת. לדוגמא את אותם קטעים שבזמן ההקלטה, בקטעים שאין דיבור, המערכת משתיקה ולא מוציאה החוצה. בעזרת קבל המתחבר לרגל AM CAP , קובעים את רמת האות שבה תתבצע ההשתקה.

ממערכת ההשתקה עובר האות מגבר נוסף AMP שתפקידו להגביר את אות הדיבור.

בחלק התחתון של הסכימה המלבנית, בצד שמאל משורטט מלבן מתחי הספק וכניסות המתחים שלו

–POWER CONDITIONING . קיימות 3 רגלי אדמה אנלוגיות- - VSSA כדי להקטין רעשים. האות האנלוגי שאותו מקליטים הוא קטן (סדר גודל מילי וולטים ) ויש לדאוג שלא יושפע מרעשים.

המלבן הימני בחלק התחתון של השרטוט הוא בקרת הרכיב –DEVICE CONTROL . כאן נמצאים רגלי התקשורת של הרכיב עם המיקרו ומעגלי הבקרה השולטים על הרכיב לפי הפקודות שנשלחו אליו.

### 2.6.5 - איכות קול ודיבור:

בכדי שהאות יתקבל לנו ללא עיוותים כאשר נשחזר אותו, קצב הדגימה חייב להיות לפחות פי 2 מהאות. כלומר אם 4khz זהו תדר הדגימה שלנו אז התדר המקסימאלי של אות השמע יהיה 2khz.

מכיוון שבפרויקט אנו מקליטים דיבור(ההקלטות הן בקולו של המשתמש), ותדר הדיבור הוא ממאות הרצים עד 1khz אז כלל הדגימה נשמר, מכיוון שתדר הדגימה של הרכיב(4khz) הוא לפחות פי 2 מתדר הדיבור(1khz).

במערכות קול "רגילות" , כאשר רוצים להקליט דוגמים את האות ואת הדגימות האלו ממירים לצורה דיגיטאלית בעזרת A\D ואת הדגימות האלו שומרים בזיכרון, כאשר נרצה להשמיע אז מוציאים את המידע מהזיכרון וממירים אותו לצורה אנלוגית בעזרת D\A .

דגימה

A\D

זיכרון

D\A

אות

שמע

**תיאור המרת אות הדיבור לאות ספרתי ושחזור**

מכיוון שהממירים מאנלוגי לדיגיטאלי ולהיפך הם בעלי מספר סיביות קטן ולרוב אינם איכותיים התוצאה הסופית היא אחוז רעשים הרמוניים ושחזור ברמה יחסית נמוכה של האות.

ברכיב isd-4004 אין את תהליך ההמרה, כלומר שאנו חוסכים 2 תהליכי המרה, מכיוון שאות השמע נשמר בצורה ישירה בזיכרון בצורתו המקורית, דבר זה משפר את איכות הקול.

### 2.6.6 - זיכרון ה-FLASH:

אחד היתרונות של טכנולוגית הקלטת הקול של ISD היא השימוש בזיכרון בלתי נדיף אשר מעניק צריכת הספק אפסית בשמירת ההודעות. החברה מתחייבת שההודעה תשמר לפחות 100 שנים ללא מתח והזיכרון ניתן לשכתוב לפחות 100 אלף פעם.

הזיכרון בנוי מ-3840K תאי זיכרון. ישנן 2400 שורות ובכל שורה 1600 תאים .

כל תא מסוגל לאחסן בתוכו רמת מתח, דבר זהמאפשר לשמור את האות בצורתו המקורית.

1600 תאים

2400

שורות

**מערך תאי הזכרון ב-ISD4004**

הזיכרון שומר את המטען שנמצא בתאים ולא נותן לו להתפרק, בגלל התכונה הזו המידע לא נמחק והזיכרון לא נדיף.

מכיוון שתדר הדגימה ברכיב של ה 16 דקות הוא 4khz אז הזמן הלוקח להקלטת תא אחד הוא:



מכיוון שישנם 1600 תאים בשורה אז הזמן הלוקח להקלטת שורה אחת הוא:

T = 1600 \* t =1600 \*0.25msec =0.4 sec

זמן הקלטה מרבי של כל הרכיב:

0.4 \* 2400 = 960sec = 16 min

כלומר הרכיב שלנו מסוגל להקליט 16 דקות.

\* ניתן לארגן את ההקלטות לפי שורות (בכל שורה הקלטה שונה) כך שאם נרצה הקלטה מסוימת אנו נפנה לשורה המסוימת בזיכרון ונשמע את ההודעה.

ברכיבים של 8, 10 ו 12 דקות תדרי הדגימה שונים ולכן זמן של השמעה/הקלטה של שורה אחת שונה.

בפרויקט הוקצה מקום ל50 הודעות , כלומר 48 שורות כל הודעה שהן 19.2 שניות.

קודי האותיות והמספרים מתפרשים בין 48 ל90 סה"ך 42 כלומר 8 שורות עודף להקלטת מילים.

### 2.6.7 - ממשק המיקרו בקר:

תקשורת SPISerial Peripheral Interface - ממשק טורי היקפי

זוהי צורת תקשורת טורית דו כיוונית בין מחשב (MASTER ) ורכיב היקפי ( SLAVE ) הכוללת 5 קווים. הקווים נראים בשרטוט הבא:

**מיקרו בקר**

**רכיב היקפי (רכיב הקול)**

SCLK

MISO

INT

SS

MOSI

תיאור החיבורים בין רכיב הדיבור למיקרו-בקר

**MOSI** - Master Out Slave In - יציאת אדון (המיקרו בקר) כניסת עבד (רכיב היקפי) . יציאת הנתון הטורי מהמיקרו אל הרכיב ההיקפי.

**SCLK** - Serial CLocK - השעון הטורי המתזמן את כניסת הנתון הטורי אל הרכיב ההיקפי.

**MISO** - Master In Slave Out - כניסת אדון יציאת עבד. זוהי כניסת הנתון הטורי הנשלח מהרכיב ההיקפי אל המיקרו. בפרויקט הספציפי אין צורך בחיבור רגל זו כי לא נשלח מידע מן הרכיב לבקר.

**INT** - INTerrupt - בעזרת רגל זו מודיע הרכיב למיקרו שהסתיימה ההודעה או שהגענו לסוף הזיכרון.

**SS** - Slave Select - בחירת עבד. בעזרת רגל זו מודיע המיקרו לרכיב שהוא פעיל. הרכיב פעיל בנמוך.

### 2.6.8 - תהליך התקשורת בממשק SPI

העברת הנתון בממשק SPI מתבססת על הכנסת הנתון הטורי לרגיסטר ההזזה בירידה של פולס השעון. כאן זה מבוצע בצורה הבאה:

העברת הנתונים הטורית מתחילה בירידה ברגל SS .

SS מוחזקת בנמוך למשך כל התקשורת הטורית ומוחזקת בגבוה בין הפקודות.

הנתון נכנס בעלייה של השעון ויוצאת בירידה של השעון כאשר ביט ה LSB הוא הראשון.

השמעה או הקלטה מתחילות כאשר הרכיב מאופשר ע"י הורדת SS לנמוך והזזת פקודה עם כתובת לתוך הרכיב.

הפקודה כוללת 16 ביטים של כתובת ועוד קוד פקודה של 8 ביט.

כל פעולה המסתיימת ב EOM- End Of Message או Overflow תייצר פסיקה. הפסיקה תתנקה בכל פעם שיתחיל מחזור SPI חדש.

פעולה מתחילה כאשר בביט RUN יש '1' ומסתיימת כאשר בביט ה RUN יש '0' .

כל הפעולות מתחילות אחרי העלייה של SS .

### 2.6.9 - קודי הפעולה

|  |  |  |  |
| --- | --- | --- | --- |
| תיאור | קוד הפעולה  XXX C0 C1 C2C3C4 | כתובת  A0---A15 | ההוראה |
| היכון לפעולה אחרי זמן שנקרא Tpud | XXX00100 | X------ X | powerup |
| היכון להשמיע מהכתובת הנתונה | XXX00111 | A0--A15 | SETPLAY |
| השמע | XXX01111 | X-------X | PLAY |
| היכון להקלטה מהכתובת הנתונה | XXX00101 | A0-----A15 | SETRECORD |
| הקלט | XXX01101 | X------A15 | REC |
| היכון להקלטה בתור מהכתובת נתונה | XXX10111 | A0-----A15 | SETMC |
| הקלט בתור | XXX11111 | X--------X | MC |
| עצור את הפעולה הנוכחית | XXX011X0 | X--------X | STOP |
| עצור את הפעולה הנוכחית וכנס למצב STANDBY | XXXX10X0 | X--------X | STOPPWRDN |
| קרא את מצב סיביות הסטטוס | XXX011X0 | X--------X | RINT |

תיאור הפקודות של רכיב הדיבור

### 2.6.10- אוגרי הבקרה

רגיסטר זה מאפשר לנו לבקר על פונקציות כגון השמעה, הקלטה, סימון תחילת הודעה וכו'.

האינפורמציה מועברת בבתים:

כתובת תחתונה

כתובת עליונה

פקודה

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| \* | \* | \* |  |  |  |  | BYTE 3 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  | BYTE 2 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  | BYTE 1 |

יש שני בתים של כתובת ובית אחד של פקודה.

ישנם 2 בתים של כתובת מכיוון שאנו צריכים להגיע ל-2400 שורות, ובית אחד של כתובת לא יספיק להגיע לכל השורות.

BYTE 3:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| x | x | x | Message  cueing | Ignore  Address | Power  up | Play\Rec | Run |

3 סיביות נמוכות ללא תפקיד.

Run: מאפשר או לא מאפשר את הפעולה.

"1"- מאפשר פעולה.

"0"- לא מאפשר פעולה.

Play\Rec: "1" מבצע השמעה.

"0" מבצע הקלטה.

Power Up: כאשר יש "1" הרכיב מחובר במצב On.

כאשר יש "0" הרכיב נכנס למצב Stand By.

Ignore Address: אם יש בביט זה "1" אז הרכיב יתעלם מהכתובת שנמצאת ב-byte 1 וב-byte 2, מכיוון שישנם פקודות שלא דורשות כתובת (כלומר אם נבקש לבצע השמעה ונתעלם מהכתובת אז הרכיב יבצע השמעה מהכתובת שהוא נמצא בה עכשיו).אם נשים בביט זה "0" הרכיב לא יתעלם מהכתובת.

Message Cueing: ברכיב זה כאשר מתבצעת הקלטת הודעה אז הוא מסמן את התחלת ההודעה. כאשר לא נדע את כתובת התחלת ההודעה אנו נשתמש ב-MC. כלומר כאשר נקליט הודעות אחת אחרי השנייה ולא נדע את כתובת התחלת כל הודעה אז נתן cue בהתחלת כל הודעה וכך נדע איפה מתחילה למעשה כל הודעה.כאשר נרצה לבצע MC נשים בביט זה "1".

### 2.6.11 - רגלי הרכיב:

**Vcca,Vccd**- אלו כניסות המתחים של הרכיב. כדי להקטין את הרעש, המעגלים האנלוגיים והדיגיטאליים משתמשים בקווי הספקה נפרדים, קווים אלו פועלים במתח של 3V.

**Vssa,Vssd**- כניסות האדמה של הרכיב גם הן נפרדות. רגלי האדמה צריכות להיות מחוברות כמה שיותר קרוב אחת אל השנייה ואל הרכיב. כולן צריכות להתחבר לקו בעל התנגדות נמוכה מאוד, וזאת כדי להבטיח שבין רגל האדמה האנלוגית לרגל האדמה הדיגיטאלית תהיה התנגדות נמוכה מ-Ω3.

**Aud Out**- יציאה זו מספקת את אות השמע היוצא למגבר השמע LM-386.

**SS**- מוסבר לעיל.

**Mosi**- מוסבר לעיל.

**Miso**- מוסבר לעיל.

**Sclk**- מוסבר לעיל.

**Int**- זוהי רגל הפסיקה של הרכיב, רגל זו תרד ל-"0" בשני מקרים:

כאשר ההקלטה נמשכת מעבר לתחום הזיכרון.

כאשר הרכיב יזהה תו סוף הודעה(EOM). כך ניתן להפריד בין הודעות ולדעת מתי כל הודעה מסתיימת.

**Rac**- ביציאת רגל זו מתקבל גל אות ריבועי באורך כולל של 200msec, כאשר הרכיב פועל בתדר דגימה של 8khz. אות זה מייצג שורה אחת בזיכרון, שורה אחת מתוך 2400 שורות שיש לרכיב. אות זה נשאר במצב "1" לוגי למשך 175msec ולאחר מכן יורד ל"0" למשך 25msec כאשר הוא מגיע לסוף השורה.

25msec

175 msec

**ANA IN+,ANA IN-** - כניסת שמע אנלוגית מהפכת ולא מהפכת. כאשר (ANA IN -) זוהי כניסת השמע המהפכת,ANA IN) +) זוהי כניסת שמע לא מהפכת. שתי הכניסות הן כניסות הפרשיות על מנת לצמצם את רמת הרעשים.

**XCLK**- ברכיב זה ישנה אפשרות להכנסת אות שעון חיצוני שיקבע את תדר הדגימה של הרכיב בעזרת רגל זו, אך חיבור זה אינו מומלץ וזאת בגלל שהמתנדים שמיוצרים כיום הם בעלי תדר עם אחוז דיוק של כ- 1%, הם מושפעים מהטמפ' והמתח הנופל עליהם, ולכן הם לא מדויקים. בנוסף לכך היצרן של הרכיב ממליץ לא לנסות להעלות את תדר הדגימה של הרכיב וזאת מפני שהמסננים הפנימיים נבנו במיוחד סביב תדר זה וכל ניסיון לשינוי יגרום לעיוותים או לחוסר פעולה של הרכיב. את רגל זו מומלץ לחבר לאדמה.

**AMCAP**- רגל זו משמשת לשליטה במערך ה-"השתקה אוטומטית", מערך זה משתיק את האות בכניסה כאשר האות חלש מאוד, וכך אנו מסננים את הרעש כאשר אין אות בכניסה.

את הרגל מחברים לקבל של 1uf אשר מהווה חלק ממערך הבדיקה שמודד את המשרעת של האות. משרעת זו עוברת השוואה לרמה מסוימת שנקבעה על ידי היצרן לאותות גדולים. רמת ההשתקה שנקבעה היא 6db ובמצבי שתיקה הרמה קבועה ל-0 את כל המערך הזה ניתן למסך על ידי חיבור הרגל ל-"1" לוגי.

### 2.6.12 - אופן הפעלת הרכיב:

כדי שנוכל להקליט ולהשמיע הודעות בעזרת רכיב הקול עלינו לתכנת את הרכיב לפי מבנה פקודות קבוע המוגדר על ידי היצרן בדפי הנתונים.

תבנית פקודות ההקלטה:

* שולחים פקודת הפעלה power up = 000100xxx.
* מבצעים השהייה של לפחות 50msec.
* שולחים שוב את פקודת ההפעלה power up.
* שולחים את פקודת היכוןלהקלטה עם הכתובת ממנה רוצים להתחיל את ההקלטהsetrecord - 101000xxx+ 2 בתים של הכתובת.
* שולחים את פקודת ההקלטה rec = 10110xxx.

תבנית פקודות ההשמעה:

* שולחים פקודת הפעלה power up = 000100xxx.
* מבצעים השהייה של לפחות 50msec.
* שולחים את פקודת היכוןלהשמעה עם הכתובת שממנה רוצים להתחיל את ההשמעה. setplay + 11100xxx 2 בתים של הכתובת ממנה רוצים להשמיע.
* שולחים את פקודת ההשמעה play = 11110xxx.

בכדי לאפשר את שליחת הפקודות אנו צריכים לאפס את רגל ה-SS המאפשרת שליחת נתונים, בנוסף אנו צריכים לסנכרן את המידע שנשלח באמצעות רגל ה-SCLK מכיוון שהמידע הנשלח הוא טורי.

בסיום שליחת הפקודות/המידע רגל ה-SS תעלה שוב לגבוה.

#### תרשים זרימה הקלטה מכתובת

התחל

שלח לרכיב פקודת הפעלה

000100xxx

חכה 50 מילישניות

שלח לרכיב פקודת הפעלה

000100xxx

סיום

שלח לרכיב 2 בתים של כתובת

שלח לרכיב פקודת היכון להקלטה

101000xxx

שלח פקודת הקלטה

10110xxx

#### תרשים זרימה השמעה מכתובת

התחל

חכה 50 מילישניות

שלח לרכיב פקודת הפעלה

000100xxx

סיום

שלח לרכיב 2 בתים של כתובת

שלח לרכיב פקודת היכון להשמעה

11100xxx

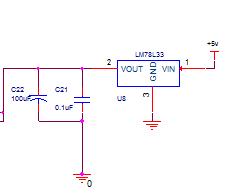
שלח פקודת השמעה

11110xxx

### 2.6.13-חיבור הרכיב למערכת מיקרו :



היות ומתח ה Vcc במעגל הוא 5 וולט והיצרן ממליץ להשתמש ב 3 וולט נוריד את המתח בעזרת מייצב מתח מעגל הספק נראה כך :



לרכיב הקול יש שתי מערכות, האחת אנלוגית והשנייה דיגיטלית ולכן יש שתי כניסות Vccd ו-Vcca שיכולות לקבל 2 מתחים נפרדים, אצלנו הן מקוצרות ומקבלות 3.1 וולט מהזנר. כמובן שגם רגלי האדמה האנלוגית והדיגיטלית מחוברות יחד.

רגל 1 של הרכיב היא רגל האפשור שלו SS – Slave Select היא מתחברת אצלנו לרגל P1.3 בה אנו משתמשים כביט פלט המאפשר את הרכיב). ניתן לחבר אותה לכל רגל פנויה אחרת באחד הפורטים.

רגל 2 של הרכיב היא רגל כניסה טורית אל הרכיב.MOSI – Master Out Slave In . הכנסת נתון זה בצורה טורית מתבצעת בתאום עם פולסי שעון אותם אנו מספקים בעזרת תוכנה ברגלSCLK של הרכיב. שתי רגלים אלו מתחברות ל-P1.4 ו-P1.5 בהתאמה שמשמשים אותנו כביטים של פלט.

רגל 13 (INT') של הרכיב משמשת ליציאה ומודיעה שהרכיב סיים את השמעת ההודעה הרצויה. אנו מחברים את רגל זו ל-p1.6' של המעבד. רגל זו נמצאת במצב "רגיל" ב '1' וכאשר נגיע לסיום ההודעה או לסוף הזיכרון, רגל זו תרד ל '0' ואז נקבל פסיקה במיקרו וניתן פקודת עצירת השמעה לרכיב. ( ניתן לעבוד עם פסיקה או לעשותpolling-שאילתה- על הרגל).

ברגליים 16-17 נכנס דרך המיקרופון אות השמע, האות נכנס למגבר הפרש שתפקידו לבטל רעשים שמתלבשים על אות השמע (יש את אותה רמת רעש בשני הדקי המגבר ולכן הרעש מתחסר משני ההדקים).

את רגל 26 (XCLK) אנו מחברים לאדמה מכיוון שאנו לא רוצים לעבוד עם אות שעון חיצוני שיקבע את תדר הדגימה אלא אנו רוצים לעבוד עם תדר הדגימה של הרכיב, דבר זה מומלץ על ידי היצרן.

את רגל 14 חיברנו לקבל של 1uf ולאדמה. ( אופן פעולת מערך ה"השתקה האוטומטי" הוסבר לעיל).

## 2.7-מגבר שמע LM386

### 2.7.1 - מבוא:

בכדי להגביר את אות השמע במוצא יש צורך בשימוש במגבר. בעזרת מגבר הספק ניתן להשיג הגבר שמע ברמה טובה, כך שהדיבור המוקלט יישמע באופן ברור- כמעט ללא עיוותי הספק.

רכיב ה- LM386 הוא מגבר הספק שתוכנן לשימוש ביישומים הדורשים עבודה במתח נמוך. היצרן קבע כי הרכיב יגביר פי 20 מאות הכניסה, אך המשתמש יכול לשנות את ההגבר באופן עצמאי ע"י חיבור קבל ונגד בערכים שונים בין הרגליים 11-8. בצורה זו ניתן להגיע להגבר מקסימאלי של 200.

תכונות נוספות: \*המוצא נקבע אוטומטית לחצי ממתח המקור

\*כניסות הרכיב מיוחסות לאדמה

\*צריכת ההספק של הרכיב במצב לא פעיל היא 24 MW, על כן הרכיב

אידיאלי לשימוש בסוללות.

### 2.7.2 - מאפייני הרכיב:

הפעלה בעזרת סוללות.

מתח הזנה "רחב" של 12V - 4V .

אחוז עיוותים נמוך.

צריכת זרם נמוכה כשהרכיב לא פעיל.

הגבר הספק משתנה, מ- 20 ועד 200.

### 2.7.3 - יישומים לרכיב:

מגברי רדיו AM/FM.

ממירי מתח.

אינטרקום.

מגבר לטייפ נייד.

הגברת שמע לטלוויזיה.

### 2.7.4 - חיבור המגבר למעגל:



כאשר מבצעים השמעה האות יוצא ברגל 13 של הרכיב הדיבור ומגיע אל מגבר 386 שהוא מגבר הספק, כאשר הפוטנציומטר המחובר בכניסה לרגל 3 באמצעותו ניתן לווסת את עוצמת השמע (volume). הקבל (10uf) המחובר אל רגל 13 זהו קבל צימוד המעביר רק את המתח החילופי AC (האות המתאר את הקול).

מגבר השמע הוא מגבר הספק שתפקידו להזין את הרמקול. הקבל של 220uf בטור לרמקול הוא קבל צימוד (מניעת DC והעברת AC). המערכת של הנגד Ω10 והקבל 50n לאחר מגבר השמע, תפקידה לדכא תנודות בתדר גבוה שהיו יכולות להתפתח בגלל ההגברה הגבוהה של המגבר.

רשת טורית R10C7 קובעת את הגבר המגבר.

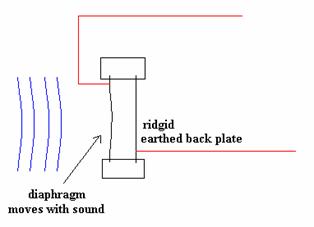
## 2.8- מיקרופון

מיקרופון הוא מתמר –מכשיר או רכיב אלקטרוני המשמש להמרה של קול לאות חשמלי. השימוש הנפוץ במיקרופון הוא המרה של גלי קול או צלילים שהם תנודות של לחץ אוויר לתנודות של מתח חשמלי- זרם חילופין.

המיקרופון הוא החוליה הראשונה ואמצעי קלט בכל מכשיר או מתקן המיועד להעביר או לאכסן גלי קול לדוגמא: לשם הקלטה בטלפון ,בקשר רדיו או לשם הזנה (קלט ) למחשב של קולות וצלילים.

**מיקרופון קיבולי (אלקטרוסטאטי)**

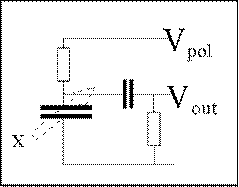
מבוסס על שינוי הקיבול החשמלי כתוצאה מגלי הקול הנוצרים בזמן הדיבור וגורמים ללחץ. מיקרופון זה מורכב מממבראנה הניתנת לתנועה, ומלוחית אחורית קבועה. פגיעת גלי הקול בממבראנה הנעה גורם לשינוי מיקומה ביחס ללוחית הקבועה, וכתוצאה מכך משתנה הקיבול החשמלי בין שתי הלוחיות.



**מבנה של מיקרופון קיבולי**

באיור רואים 2 לוחות של קבל. לוח אחד קבוע ולוח שני הבנוי ממבראנה ( diaphragm ). גלי הקול לוחצים על הממבראנה ומזיזים אותה ימינה או שמאלה. המרחק בין הלוחות משתנה ואתו קיבול הקבל.

באיור הבא מתואר החיבור החשמלי שלו.



**מעגל חיבור של מיקרופון קיבולי**

גלי הקול המסומנים ב X משנים את המרחק בין הלוחות ולכן את הקיבול של הקבל. היות ו

Q = C \* V אז שינוי הקיבול מביא לשינוי במתח על הקבל ולכן נקבל ביציאה Vout מתח חילופין שהוא אות השמע. יש לספק מתח Vpol בסדר גודל של 48 וולט כדי לקבל גל שמע ב Vout. עקום הענות מ 20 הרץ ועד 20 קילו הרץ.

יתרונות: א. רוחב סרט גדול ב. רגישות לחץ כ 20 מילי וולט לפסקל.

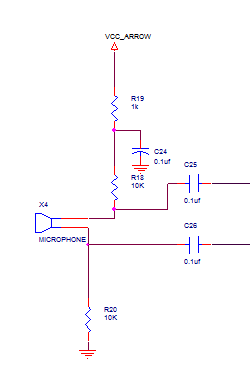
חסרונות: א. דורש מתח הפעלה גבוה של כ 48 וולט. ב. רגיש ללחות

יישום : אולפני הקלטה.

קיים סוג נוסף של מיקרופון הנקרא Electret Microphone. זהו מיקרופון העובד כל עיקרון המיקרופון הקיבולי אך איננו דורש מתח הפעלה גבוה של 48 וולט. הסיבה לכך היא שבמפעל, זמן הייצור מספקים לו מטען ואז לא צריך מתח אספקה גבוה. גם לו עקום הענות גדול אם כי פחות טוב מהמיקרופון הקיבולי. היישום שלו הוא מיקרופונים ניידים הנתפסים על דש הבגד כי הם קלים וזולים לייצור המוני.

מעגל הכניסה מהמיקרופון אל רכיב הקול נראה כך :

מעגל המיקרופון



נגד 5R וקבל 12C הם מערכת ביטול צימוד. תפקידם לדאוג שרעש מהספק לא יגיע אל רכיב הקול ולהיפך שהאות אותו מקליטים לא ישפיע על ה Vcc . הנגדים 6R ו 8R הם נגדים שווים עליהם אנו מוציאים את האות שנותן המיקרופון כשני אותות בגודל שווה ובהיפוך מופע אל רכיב הקול. תפקיד נוסף שלהם הוא לתת ממתח נכון למיקרופון הגבישי שבו השתמשנו בפרויקט. הקבלים 10C ו 14C הם קבלי צימוד המעבירים רק את אות החילופין מהמיקרופון ללא רמות ה DC שעל הנגדים.

## 2.9-רמקולC:\Users\uri\Desktop\פרוייקט\ספר פרוייקט\20140324_165005.jpg

רמקול (הלחם של המילים "רם" ו"קול") הוא מתמר המשמש להפיכת אותות חשמליים לצלילים שאותם ניתן לשמוע.

רמקול בסיסי בנוי ממגנט קבוע ומסליל של מוליך חשמלי החופשי לנוע בתוך השדה המגנטי הקבוע של המגנט.

אל הסליל מחוברת דיאפגרמה –(ממברנה דקה, צפידה למחצה)

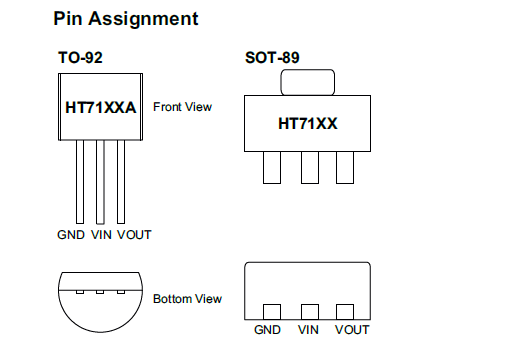
בצורה המזכירה חרוט רחב בסיס. שולי הממברנה מחוברים אל מסגרת מתכתית בחיבור גמיש שאיננו מונע את תנועתה המגנט הקבוע מחובר אל המסגרת המתכתית בחיבור קשיח.

כאשר זרם חשמלי עובר בסליל מושרה בתוך הסליל שדה אלקטרומגנטי שעצמתו וכיוונו יחסיים לעצמת וכיוון הזרם החשמלי . שדה אלקטרומגנטי זה גורם לתנועה של הסליל ביחס למגנט הקבוע הנגרמת מהכוחות הפועלים בין השדה המגנטי הקבוע והשדה האלקטרומגנטי המושרה מתוך הסליל.

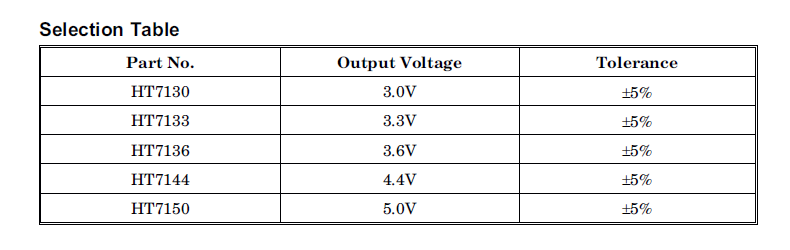
תנועת הסליל גורמת לתנועה של הממברנה המחוברת אליו, וזו דוחפת את האוויר שמסביבה בכיוון ובעצמה יחסיים לכיוונו ועצמתו של הזרם החשמלי העובר בסליל מתנודד, הוא יגרום לתנודה של הממברנה בקצב ובעצמה התואמים לקצב ועצמת תנודות הזרם , ולהרעדת האוויר סביב הממברנה.

תנודות האוויר הנוצרות הן גלי קול – הצלילים המופקים מהרמקול קיימים גם רמקולים הפועלים על עיקרון הפיאזו אלקטריות ובנויים ללא סליל או מגנט , רמקולים אלו נפוצים מאוד במכשירים אלקטרוניים זעירים או בהתקנים המיועדים להשמעת תדר גבוה.

## 2.10-מייצב מתח גבוה HT 71 XX



סדרת HT 71 XX היא קבוצה של מייצבי מתח בטכנולוגיית CMOS . מתח הכניסה האפשרי הוא עד 24v ואילו מתח היציאה קבוע תלוי בסוג הרכיב .



מאפיינים:

צריכת הספק נמוכה

נפילת מתח נמוכה

מקדם טמפרטורה נמוך

מתח כניסה גבוה- עד 24v

יישומים:

ציוד מופעל בטריות

ציוד תקשורת

ציוד אודיו \וידיו

בפרויקט השתמשתי ברכיב על מנת להמיר את מתח האספקה הכללי 5v ל-3v אתו פועל רכיב הקול.

# פרק 3 תכנה

## 3.1-תרשים זרימה כללי

התחלה

הפעלת הLCD הצגת הודעות פתיחה

כן

האם נלחץ מקש?

כן

לא

לא

כן

כן

לא

האם המחרוזת חוקית

האם נלחצה המחרוזת bego

התחלה בשגרת השמעה בכתובת יחסית לאות או מוקצה מראש למחרוזת

זיהוי הצבע והשמעתו מכתובת מוקצה מראש

לא

האם נקלט צבע?

בקשה מהמשתמש לשים צבע או ללחוץ על מקש

שגרת הקלטה

בכתובת יחסית לאות

או מוקצה מראש למחרוזת

## 3.2-תכנת הפרויקט

### תכנה ראשית

##include "REG\_89c51.h"

#include "type\_var\_h.h"

#include "lcd\_play.h"

#include "isd.h"

void delay\_lcd();

void play(int addr);

void read\_key();

void lcd\_init();

void multiplay( u8 arr[]);

char read\_word( char arr[]);

u16 asc\_to\_ad(u8 asc);

void delay(int x);

void password();

void a\_reset(u8 arr[]);

bit strcmp1(char a[],char b[],char len);

extern bit data error;

extern u8 data key;

extern bit parity,f\_start,f\_read\_key;

bit f\_release,f\_caps\_lock;

extern u8 data key;

extern bit error\_parity,f\_start,f\_read\_key;

bit f\_release,f\_caps\_lock,read;

void main()

{

u8 data len=0;

u8 data arr[9];

u16 addr;

lcd\_setup();

delay\_lcd();

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*lcd initialization\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

while(1)

{

lcd\_init();

a\_reset(arr); //resetting array

read=0; //read 0 states default state scanning both color and keyboard

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*key reading\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

f\_caps\_lock=0;

len=read\_word(arr); //word stored in arr

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*key processing\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

if(len>1)

{

multiplay(arr) ; //routine for handling multiple letter strings

}

else

if(len!=0)

{

addr=asc\_to\_ad(arr[0]);//converts character to adress

if(error) //character illegal

{

lcd\_clear();

delay\_lcd();

writem("cannot play");

delay(65000);

lcd\_clear();

delay\_lcd();

}

else

play(addr);

}

}

}

### שגרות ל-LCD

//===========================================================================

// "lcd\_play.c"

//===========================================================================

//===========================================================================

// LCD in 4 bit mode only

// Intercommunion between LCD and Microcontroller

//

// E (Enable) -------- P2\_2

// R/W (Read / Write) -------- P2\_1

// RS (Character / Command) -------- P2\_0

// D4-7 (4bits data) -------- P2

//===========================================================================

//===========================================================================

//#include "REG\_T89C51RD2\_68.h"

//#include "c\_type.h"

#include "REG\_89C51.h"

#include "type\_var\_H.h"

#define RS\_lcd P2\_0

#define RW\_lcd P2\_1

#define E\_lcd P2\_2

#define DATA\_lcd P2

static u8 data lcd\_num ; // lcd position on line (0-15)

static bit lcd\_line ; // lcd line: 0 - line 1

// 1 - line 2

//===========================================================================

void lcd\_wait()

{

u8 lcd\_t1;

for (lcd\_t1 = 1 ; lcd\_t1 < 5 ; lcd\_t1++); // let the display controller

// time to execute the command

}//lcd\_wait()

//===========================================================================

void set\_lcd1(bit char\_command,u8 lcd\_d)

{

RW\_lcd = 0 ;

E\_lcd = 1 ;

RS\_lcd = char\_command ;

DATA\_lcd &= 0x87; DATA\_lcd |= ((lcd\_d &0xf0 )>>1);

E\_lcd = 0 ;

lcd\_wait();

E\_lcd = 1 ;

lcd\_wait();

}//set\_lcd(bit char\_command,char lcd\_d)

void set\_lcd(bit char\_command,u8 lcd\_d)

{

RW\_lcd = 0 ;

E\_lcd = 1 ;

RS\_lcd = char\_command ;

DATA\_lcd &= 0x87;

DATA\_lcd |= ((lcd\_d &0xf0 )>>1);

E\_lcd = 0 ;

lcd\_wait();

E\_lcd = 1 ;

// lcd\_wait();

RW\_lcd = 0 ;

E\_lcd = 1 ;

RS\_lcd = char\_command ;

DATA\_lcd &= 0x87;

DATA\_lcd |= ((lcd\_d &0x0f )<<3) ;

E\_lcd = 0 ;

lcd\_wait();

E\_lcd = 1 ;

lcd\_wait();

}//set\_lcd(bit char\_command,char lcd\_d)

//===========================================================================

/\*

I/D - when high, cursor moves right DDRAM addr. inc. by 1

- when low, cursor moves left DDRAM addr. dec. by 1

SH - Shift of entire display

D - Display on/off (high = on; low = off; Data stored)

C - Cursor on/off (high = on; low = off; ID stored)

B - Cursor blink on/off (high = blink on; low = blink off)

S/C - 1 - display; 0 - cursor;

R/L - 1 - right; 0 - left;

DL - Interface data length ( 1 - 8 bit; 0 - 4 bit;)

N - Display line number (1 - two lines; 0 - one line;)

F - Display font type (1 - 5x11; 0 - 5x8;)

\*/

//===========================================================================

void lcd\_setup()

{

u8 lcd\_i;

lcd\_num = 0 ;

lcd\_line = 0 ;

for (lcd\_i = 0 ; lcd\_i < 10 ; lcd\_i++) // Recovery time after power on

lcd\_wait();

set\_lcd1(0,0x30);// 0 0 1 DL N F 0 0 - FUNCTION SET - 8bit interface

set\_lcd1(0,0x30);// 0 0 1 DL N F 0 0 - FUNCTION SET - 8bit interface

set\_lcd1(0,0x30);// 0 0 1 DL N F 0 0 - FUNCTION SET - 8bit interface

set\_lcd1(0,0x2C);// 0 0 1 DL N F 0 0 - FUNCTION SET - 4bit interface

set\_lcd(0,0x2C);// 0 0 1 DL N F 0 0 - FUNCTION SET - 4bit interface

set\_lcd(0,0x0c);// 0 0 0 0 1 D C B - DISPLAY ON/OFF CONTROL

set\_lcd(0,0x06);// 0 0 0 0 0 1 I/D SH - ENTRY MODE SET

set\_lcd(0,0x14);// 0 0 0 1 S/C R/L 0 0 - CURSOR OR DISPLAY SHIFT

set\_lcd(0,0x01);// 0 0 0 0 0 0 0 1 - DISPLAY CLEAR

}//lcd\_setup()

void lcd\_on()

{

set\_lcd(0,0x0c);// 0 0 0 0 1 D C B - DISPLAY ON/OFF CONTROL

}//lcd\_on()

//===========================================================================

void lcd\_off()

{

set\_lcd(0,0x08);// 0 0 0 0 1 D C B - DISPLAY ON/OFF CONTROL

}//lcd\_off()

//===========================================================================

void lcd\_clear()

{

lcd\_num = 0 ;

lcd\_line = 0 ;

set\_lcd(0,0x01);// 0 0 0 0 0 0 0 1 - DISPLAY CLEAR

}//lcd\_clear()

//===========================================================================

void lcd\_home()

{

lcd\_num = 0 ;

lcd\_line = 0 ;

set\_lcd(0,0x02);// 0 0 0 0 0 0 1 0 - DISPLAY HOME

}//lcd\_home()

//===========================================================================

void lcd\_line2()

{

lcd\_num = 0 ;

lcd\_line = 1 ;

set\_lcd(0,0xc0);// 1 1 0 0 0 0 0 0 - LINE 2

}//lcd\_line2()

//===========================================================================

void lcd\_write(char lcd\_data)

{

set\_lcd(1,lcd\_data);

if (lcd\_num == 0x0f)

{

if (lcd\_line)

lcd\_home();

else

lcd\_line2();

}

else

lcd\_num++;

}//lcd\_write(char lcd\_data)

//===========================================================================

static void lcd\_end\_line0()

{

lcd\_line = 0 ;

lcd\_num = 0x0f ;

set\_lcd(0,0x8f);

}//lcd\_end\_line0()

//===========================================================================

static void lcd\_end\_line1()

{

lcd\_line = 1 ;

lcd\_num = 0x0f ;

set\_lcd(0,0xcf);

}//lcd\_end\_line1()

//===========================================================================

void lcd\_back1()

{

u8 lcd\_add;

if (lcd\_num == 0)

{

if (lcd\_line)

{

lcd\_line = 0 ;

lcd\_end\_line0();

lcd\_num = 15 ;

}

else

{

lcd\_line = 1 ;

lcd\_end\_line1();

lcd\_num = 15 ;

}

}

else

{

lcd\_num--;

if (lcd\_line)

{

lcd\_add = 0xc0 | lcd\_num ; // - line2 0x40

set\_lcd(0,lcd\_add);

}

else

{

lcd\_add = 0x80 | lcd\_num ; // - line1 0x00

set\_lcd(0,lcd\_add);

}

}

}//void lcd\_back1()

//===========================================================================

void delete\_tav()

{

lcd\_back1();

lcd\_write(' ');

lcd\_back1();

}//void delete\_tav()

//===========================================================================

void writem(char message[])

{

int i;

for (i = 0 ; message[i] ; i++)

lcd\_write(message[i]);//write message on display

}

//===========================================================================

void wnum(u16 x)

{

int i,div=100;

for (i = 1 ; i<4 ; i++)

{

lcd\_write(x/(div)+48);

x=x%div;

div/=10;

}

}

//===========================================================================

void wf(float x)

{

int dec;

dec=x\*1000;

dec%=1000;

wnum(x);

lcd\_write('.');

wnum(dec);

}

//===========================================================================

// End Of File "lcd\_play8.c"

//===========================================================================

### השהיה

//==================== delay1.c ===================

#include "type\_var\_H.h"

void delay(u16 time)

{

u16 data i;

for(i=0;i<time;i++);

}//delay1

### השהיית LCD

#include "type\_var\_H.h"

void delay\_lcd()

{

u16 data i;

for(i=0;i<750;i++);

}//delay1

### שגרת אתחול התצוגה

#include "REG\_89C51.h"

#include "type\_var\_H.h"

#include "lcd\_play.h"

void delay(u16 x);

void lcd\_init()

{

lcd\_clear();

delay\_lcd();

writem("welcome");

delay(65000);

lcd\_clear();

delay\_lcd();

writem("please press key or put color");

delay(65000);

lcd\_clear();

delay\_lcd();

}

### שגרות זיהוי הצבע:

#include "REG\_89c51.h"

#include "type\_var\_h.h"

#include "lcd\_play.h"

#define S0 P3\_2

#define S1 P3\_3

#define S2 P3\_4

#define S3 P3\_7

#define OE P3\_6

#define stcol 2064

void lcd\_init();

void delay(int);

void play(int addr);

unsigned int Rcolor;

unsigned int Gcolor;

unsigned int Bcolor;

void Timer\_init(void)

{

OE=0; //output enable active low

TMOD =0X51; //01010001 -timer 1 counter mode 1 timer0 timer mode 1

TF0=0;

}

unsigned int getcolor(void)

{

unsigned int value;

value=0;

TH0 = 0X00;

TL0 = 0X00;

TH1 = 0X00;

TL1 = 0X00;

TF0 = 0;

TF1 = 0;

TR0 = 1 ; //start count to 0xFFFF

TR1 = 1; //start count p3.5 pulses

while(!TF0);//timer 0 reached it's max

value = (TH1<<8)|TL1;//value TH1+TL1

TR0 = 0; //stop count

TR1 = 0; //stop count

return value;

}

bit color(void) //Check\_Color

{

u8 code col1[6][7]={"blue","pink","purple","orange","yellow","white"} ;

int i=0;

float rb,rg;

lcd\_clear();delay\_lcd();

Timer\_init();

S1=1;

S0=0;

Bcolor=0;

Gcolor=0;

Rcolor=0;

lcd\_clear();

S3=0;

S2=0;

Rcolor = getcolor();

S3=1;

Bcolor = getcolor();

S2=1;

Gcolor = getcolor();

lcd\_clear();delay\_lcd();

rb=(float)Rcolor/Bcolor;//proportions of colors

rg=(float)Rcolor/Gcolor;//proportions of colors

if(Rcolor<80 && Bcolor<80 && Gcolor<80 )

return 0; //color not detectet

else

{

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*color presenting\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

writem("R/B");

delay(30000); lcd\_clear();delay\_lcd();

wf(rb);

delay(40000); lcd\_clear();delay\_lcd();

writem("R/G");

delay(30000); lcd\_clear();delay\_lcd();

wf(rg);

delay(40000); lcd\_clear();delay\_lcd();

writem("the color is "); lcd\_line2();

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*color playing\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

if((rb>0.76) && (rb<0.8) && (rg>0.83) && (rg<0.88))

{ writem(col1[i]); play (stcol+i\*48); } //blue

else

{

i++;

if((rb>1.03) && (rb<1.08) && (rg>1.18) && (rg<1.23))

{ writem(col1[i]); play (stcol+i\*48);} //pink

else

{

i++;

if((rb>0.86) && (rb<0.89) && (rg>1.00) && (rg<1.03))

{writem(col1[i]); play (stcol+i\*48); }//purple

else

{

i++;

if((rb>1.52) && (rb<1.63) && (rg>1.46) && (rg<1.55))

{ writem(col1[i]); play (stcol+i\*48);}//orange

else

{

i++;

if((rb>1.2) && (rb<1.26) && (rg>0.89) && (rg<0.96))

{ writem(col1[i]); play (stcol+i\*48);} //yellow

else

{

i++;

if((rb>0.89) && (rb<0.93) && (rg>0.95) && (rg<0.98))

{ writem(col1[i]); play (stcol+i\*48);} //white

}}}}}

delay(50000);

lcd\_clear(); delay\_lcd();

}

return 1;

}

//=====================================================

// End Of File "TCS3200.c"

### שגרת סריקת מקש או צבע מן המקלדת או חיישן הצבע:

//======================================================

// "READ\_KEY.c"

//======================================================

#include "reg\_89c51.h"

#include "type\_var\_h.h"

#define KBD\_CLK P3\_0

#define KBD\_DATA P3\_1

u8 data key;

bit fxor,parity,f\_read\_key;

bit color();

bit read\_key(bit read)

{

//read is a flag to prevent color scanning when unnessacary

u8 data i;

u16 j=0;

f\_read\_key=0; key=0;fxor=0;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* check start bit \*\*\*\*\*\*\*\*\*\*\*\*

while(KBD\_CLK ) //KBD\_CLK down

{

if(read==0)//incorparation of color checking when clock is down too long

{

j++;

if(j==20000)

if(color())

return 0; //color detected

else

j=0; //continue scanning

}

}

if(KBD\_DATA) //check

return 1;//f\_start=1;

while(!KBD\_CLK ); //KBD\_CLK up

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* shift register \*\*\*\*\*\*\*\*\*\*\*\*\*

for(i=0;i<8;i++)

{

while(KBD\_CLK ); //KBD\_CLK down

if(KBD\_DATA)

key|=1<<i;//shift register

fxor=fxor^KBD\_DATA;

while(!KBD\_CLK ); //KBD\_CLK up

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* check parity \*\*\*\*\*\*\*\*\*\*\*\*

while(KBD\_CLK ); //KBD\_CLK down

parity = KBD\_DATA;

if(parity == fxor)//check parity-odd pairity

return 1; //error data

while(!KBD\_CLK ); //KBD\_CLK up

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* stop bit \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

while(KBD\_CLK );

if(!KBD\_DATA) //check stop bit

return 1;//f\_start=1;

f\_read\_key=1;//key eading successful

while(KBD\_CLK ); //KBD\_CLK down

while(!KBD\_CLK );

return 1; //KBD\_CLK up

}// main()

//=====================================================

// End Of File "READ\_KEY.c"

//=====================================================

### שגרת קריאת מילה לתוך מערך מתוך המקלדת

#include "REG\_89C51.h"

#include "lcd\_play.h"

#include "type\_var\_H.h"

bit read\_key(bit read);

void delay(u16 time);

bit strcmp1(char a[],char b[],char len);

extern u8 data key;

extern bit parity,f\_start,f\_read\_key,f\_caps\_lock;

char read\_word( char arr[])

{

// 0 1 2 3 4 5 6 7 8 9

//============================================================

u8 code ps2ASCII[] = { 0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 , //00

0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 , //01

0 ,0x71 ,0x31 ,0 ,0 ,0 ,0x7a ,0x73 ,0x61 ,0x77 , //02

0x32 ,0 ,0 ,0x63 ,0x78 ,0x64 ,0x65 ,0x34 ,0x33 ,0 , //03

0 ,0x20 ,0x76 ,0x66 ,0x74 ,0x72 ,0x35 ,0 ,0 ,0x6e , //04

0x62 ,0x68 ,0x67 ,0x79 ,0x36 ,0 ,0 ,0 ,0x6d ,0x6a , //05

0x75 ,0x37 ,0x38 ,0 ,0 ,0x2c ,0x6b ,0x69 ,0x6f ,0x30 , //06

0x39 ,0 ,0 ,0x2e ,0x2f ,0x6c ,0x3b ,0x70 ,0x2d ,0 , //07

0 ,0 ,0x60 ,0 ,0x5b ,0x3d ,0 ,0 ,0 ,0 , //08

0 ,0x5d ,0 ,0x5c ,0 ,0 ,0 ,0 ,0 ,0 , //09

0 ,0 ,0 ,0 ,0 ,0x31 ,0 ,0x34 ,0x37 ,0 , //10

0 ,0 ,0x30 ,0x2e ,0x32 ,0x35 ,0x36 ,0x38 ,0x07 ,0 , //11

0 ,0x2b ,0x33 ,0x2d ,0x2a ,0x39 ,0 ,0 ,0 ,0 };//12

// 0 1 2 3 4 5 6 7 8 9

//============================================================

u8 code ps2ASCIIbig[]={ 0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 , //00

0 ,0 ,0 ,0 ,0x60 ,0 ,0 ,0 ,0 ,0 , //01

0 ,0x51 ,0x21 ,0 ,0 ,0 ,0x5a ,0x53 ,0x41 ,0x57 , //02

0x40 ,0 ,0 ,0x43 ,0x58 ,0x44 ,0x45 ,0x24 ,0x23 ,0 , //03

0 ,0x20 ,0x56 ,0x46 ,0x54 ,0x52 ,0x25 ,0 ,0 ,0x4e , //04

0x42 ,0x48 ,0x47 ,0x59 ,0x5e ,0 ,0 ,0 ,0x4d ,0x4a , //05

0x55 ,0x26 ,0x2a ,0 ,0 ,0x3c ,0x4b ,0x49 ,0x4f ,0x29 , //06

0x28 ,0 ,0 ,0x3e ,0x3f ,0x4c ,0x3a ,0x50 ,0x5f ,0 , //07

0 ,0 ,0x22 ,0 ,0x7b ,0x2b ,0 ,0 ,0 ,0 , //08

0 ,0xd ,0 ,0x7c ,0 ,0 ,0 ,0 ,0 ,0 , //09

0 ,0 ,0 ,0 ,0 ,0x31 ,0 ,0x34 ,0x37 ,0 , //10

0 ,0 ,0x30 ,0x2e ,0x32 ,0x35 ,0x36 ,0x38 ,0 ,0 , //11

0 ,0x2b ,0x33 ,0x2d ,0x2a ,0x39 ,0 ,0 ,0 ,0 };//12

u8 len=0,temp;

extern bit read;

do

{

key=0;

read=read\_key(read);

if(f\_read\_key)//flag for successful reading

{

switch(key)

{

case 0xf0 ://key held down

read=read\_key(read); break;

case 0x66 ://backspace

delete\_tav();

len--;

arr[len]=0; break;

case 0x58 ://caps lock

f\_caps\_lock=~f\_caps\_lock; break;

case 0x5A ://enter

break;

default :

if(!f\_caps\_lock )

temp=ps2ASCII[key];

else

temp=ps2ASCIIbig[key];

lcd\_write(temp);

arr[len]=temp;

len++; break;

}

}//if(f\_read\_key)

} while(key!=0x5A&&read);//while

return len;

}

### שגרות רכיב הקול ISD-4004

#include <reg52.h>

//#include <at89c5131.h>

/\* pin description \*/

sbit \_SS = P1^3;

sbit \_MOSI = P1^4;

sbit \_SCLK = P1^5;

sbit \_INTR = P1^6;

/\* constant defination \*/

#define LOW 0

#define HIGH 1

#define TRUE 1

#define FALSE 0

#define SET\_PLAY 0xE0//SETPLAY

#define SET\_RECORD 0xA0//SETRECORD

#define SET\_MC 0xE8//SETMC

int P\_R;// Play or Record

void delay(int);

void nop()

{

unsigned char i;

for(i=0;i<2;i++);

}

static void StartISD (void)

{

\_SS = HIGH;

nop();

\_SCLK = LOW;

nop();

\_SS = LOW;//choose chip

nop();

}

static void StopISD (void)

{

\_SCLK = LOW;

\_SS = HIGH;//release chip

nop();

nop();

}

static void SendByte (unsigned char byte)

{

unsigned char i = 8;

\_SCLK = LOW;

byte >>= 1;//transmission lsb to msb

do

{

\_MOSI = CY;//carry

\_SCLK = HIGH;

byte >>= 1;

\_SCLK = LOW;

}

while(--i);

}

static void SendWord (unsigned int word)

{

unsigned char i = 16;

\_SCLK = LOW;

word >>= 1;

do

{

\_MOSI = CY;

\_SCLK = HIGH;

word >>= 1;

\_SCLK = LOW;

}

while(--i);

}

void PowerUp (void)

{

StartISD();

SendByte(0x20);//powerup

StopISD();

}

void StopAction(void)

{

StartISD();

SendByte(0x30);//stop

StopISD();

}

void PowerDown (void)

{

StartISD();

SendByte(0x10);//STOPPWRDN

StopISD();

}

void SetPlay (unsigned int address)

{

StartISD();

SendWord(address);

SendByte(SET\_PLAY);

StopISD();

}

void Play (void)

{

StartISD();

SendByte(0xF0);//play

StopISD();

}

void SetRecord (unsigned int address)

{

StartISD();

SendWord(address);

SendByte(SET\_RECORD);

StopISD();

}

void Record (void)

{

StartISD();

SendByte(0xB0);//rec

StopISD();

}

### שגרות הקלטה:

#include "REG\_89C51.h"

#include "type\_var\_h.h"

#include "lcd\_play.h"

#include "isd.h"

#define time 65000//

u8 read\_word(char[]);

void delay(u16);

void multirec( u8 arr[]);

void delay\_lcd();

void a\_reset(u8 arr[]);

u16 asc\_to\_ad(u8 );

void Record();

#### שגרת הקלטה מכתובת נתונה:

void record(u16 addr)//routine to record

{

PowerUp ();

delay(3000);

PowerUp ();//datsheet requires two powerups

delay(3000); // addtion

SetRecord (addr);

Record (); //Record in isd3

writem("now recording");

delay(time);delay(time);delay(time);delay(time);delay(time);

StopAction();

delay(3000);

PowerDown();

}

#### שגרת קליטת מחרוזת והקלטתה בכתובת יחסית

void read\_record()

{

u16 addr;

bit flag=1;

u8 len=0, arr[9];

while(flag)

{

writem(" ENTER: start");

delay(65000);delay(65000);

lcd\_clear();

delay\_lcd();

len=read\_word(arr);

lcd\_clear();

delay\_lcd();

if(len>1) //multiple letters recording

{

multirec(arr) ;

lcd\_clear();

delay\_lcd();

}

else

{

addr=asc\_to\_ad(arr[0]); //converts key to adress

if(arr[0]!=';')

{

record(addr);

lcd\_clear();

delay\_lcd();

}

else

flag=0;

}

a\_reset(arr);

}

}

### שגרת השמעה מכתובת נתונה:

#include "REG\_89C51.h"

#include "type\_var\_h.h"

#include "isd.h"

#define time 65000 //time will be determined

void delay(u16);

void play( u16 addr)//this routine is for playing the sign

{

PowerUp ();

delay(3000);

SetPlay(addr);

Play();

delay(time);delay(time);delay(time);delay(time);delay(time);

StopAction();

delay(3000);

PowerDown();

}

### שגרה להקלטה של מחרוזות מרובות תווים:

#include "REG\_89c51.h"

#include "type\_var\_h.h"

#include "lcd\_play.h"

#include "isd.h"

#define stcol 2064 //limit of normal characters

void delay\_lcd();

bit strcmp1(char a[],char b[]);

void play(int addr);

void record(u16 addr);

void delay(int x);

void read\_record();

extern bit read;

void multirec( u8 arr[])

{

bit flag2=0;

int i=0;

u8 code col1[6][7]={"blue","pink","purple","orange","yellow","white"} ;

while(i<6&&!flag2) //flag or end of array

{

if(strcmp1(col1[i],arr)) //compares the pressed word to the colors

{

flag2=1;

record(stcol+i\*48);

}

i++;

}//while

//\*\*\*\*\*\*\*\*\*\*\*\*\*word not in array\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

if(flag2==0)

{

lcd\_clear();

delay\_lcd();

writem("cannot record");

delay(30000);

}

}

### שגרה להשמעה של מחרוזות מרובות תווים:

#include "REG\_89c51.h"

#include "type\_var\_h.h"

#include "lcd\_play.h"

#include "isd.h"

#define stcol 2064 //limit of normal characters

void delay\_lcd();

bit strcmp1(char a[],char b[]);

void play(int addr);

void record(u16 addr);

void delay(int x);

void read\_record();

extern bit read;

void multiplay( u8 arr[])

{

bit flag2=0;

int i=0;

u8 code col1[6][7]={"blue","pink","purple","orange","yellow","white"} ;

while(i<6&&!flag2) //flag or end of array

{

if(strcmp1(col1[i],arr)) //compares the pressed word to the colors

{

flag2=1;

//if(pr) //flag read or record

play (stcol+i\*48);

}

i++;

}//while

//\*\*\*\*\*\*\*\*\*\*\*\*\*word not in array\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

if(flag2==0)

{

if(strcmp1("bego",arr))

{ read=1;

lcd\_clear();

delay\_lcd();

read\_record();

}

else

{

lcd\_clear();

delay\_lcd();

writem("cannot play");

delay(30000);

}

}

### שגרות טכניות:

#include "REG\_89C51.h"

#include "type\_var\_H.h"

#include "lcd\_play.h"

bit error=0;

#### שגרה להשוואה בין שתי מחרוזות

bit strcmp1(char a[],char b[]) //compares two strings

{

int i=0;

bit flag=1;

while(flag && b[i] && a[i])

{

if(a[i]!=b[i])

flag=0;

i++;

}

if(flag)

if(a[i]||b[i])

flag=0;

return flag;

}

#### שגרה לאיפוס מחרוזת

void a\_reset(u8 arr[])//resetting array

{

int i;

for(i=0;i<9;i++)//resetting array

arr[i]=0;

}

#### שגרה להמרת תו לכתובת יחסית

u16 asc\_to\_ad(u8 asc)//this routine converts the keys sent to the address in which they are recorded in

{

int i=2016;//42\*48

error=0;

if(asc>96&&asc<123)

asc-=32;//converts miniscule to majescule

if(asc<91&&asc>47)// if natural signs

return (asc-48)\*48;// stored in respective adresses

error=1;//sign is unwanted

return 0;//mandatory return

}

## קבצי הכרזה על פונקציות

//===========================================================================

// "lcd\_play.h"

//===========================================================================

void lcd\_setup();

void lcd\_off();

void lcd\_on();

void lcd\_clear();

void lcd\_home();

void lcd\_line2();

void cursor\_off();

void cursor\_on();

void blink\_on();

void blink\_off();

void lcd\_line\_down();

void lcd\_line\_up();

void lcd\_forward1();

void lcd\_back1();

static void lcd\_end\_line0();

static void lcd\_end\_line1();

void delete\_tav();

void lcd\_write\_add(char);

void lcd\_write(char);

void writem(char[]);

void delay\_lcd();

void wnum( u16);

void wf (float);

//===========================================================================

// End Of File "lcd\_play.h"

//===========================================================================

//===========================================================================

// "isd.h"

//===========================================================================

#ifndef isd

#define isd

void nop();

static void StartISD (void);

static void StopISD (void);

static void SendByte (unsigned char byte);

static void SendWord (unsigned int word);

void PowerUp (void);

void StopAction(void);

void PowerDown (void);

void SetPlay (unsigned int address);

void Play (void);

void SetRecord (unsigned int address);

void Record (void);

//void delay (unsigned int x);

#endif

//===========================================================================

// End Of File "isd.h"

//===========================================================================

# סיכום ומסקנות

במהלך הפרויקט נתקלתי במספר בעיות :

1. לא הצלחתי להפעיל את חיישן הצבע – במוצאו לא התקבל אף דופק לאחר מספר שבועות התגלה שהבעיה היא שלהדק ה- OE שפעיל בנמוך הושם "1" עם גילוי הבעיה על ידי המנחים היא תוקנה ובמהרה חיישן הצבע עבד ללא רבב.
2. עם סיום בניית החמרה בפעם הראשונה נדהמתי לגלות שהערכה פסקה מלפעול . פניתי למנחה יואל והוחלט לנתק את הפרויקט כולו והוא חווט מחדש בהשגחתו הצמודה של המנחה.

ניתן לשפר את הפרויקט על ידי הוספה של זיכרון פלאש שיוכל לאחסן רצף אותיות ,וכך לזיהוי והקלטה דינאמית של המספרים הצבעים והאותיות יתווספו גם מספר גדול של מילים.

מהעבודה על הפרויקט למדתי רבות על אלקטרוניקה מעשית, חיווט , שרטוט ב-ORCAD ,תרשים זרימה, ואף שבתחום התכנה כבר היה לי ניסיון טרם בניית הפרויקט , התכנה שמוצגת כאן היא בסדר גודל גדול בהרבה מכל דבר שכתבתי קודם לכן ולמדתי גם לא מעט על החיבור בין תכנה לחמרה.