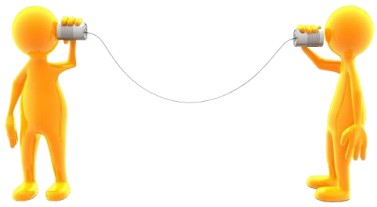
# תקשורת טורית



**ציוד נדרש:**

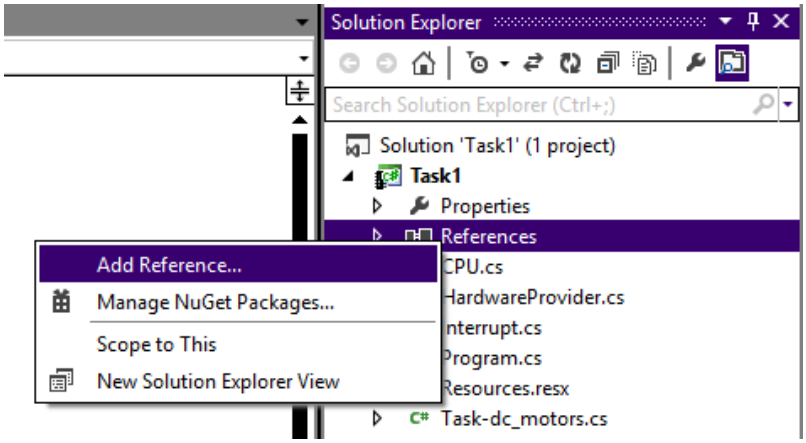
- ערכת פיתוח
- לשם בדיקה ניתן לבחור בין:
  - מחשב עם יציאת תקשורת טורית קווית או Bluetooth שמותקנת בו תוכנת שליחת / קבלת הודעות טקסט דרך התקשורת הטורית
  - מכשיר טלפון נייד עם Bluetooth שמותקנת בו תוכנת שליחת / קבלת הודעות טקסט דרך ה Bluetooth.

**רקע עיוני**


- תקשורת טורית מול תקשורת מקבילית – יתרונות וחסרונות
- דוגמאות לשימוש בתקשורת טורית
- למתקדמים: עקרונות פרוטוקולים שונים של תקשורת טורית
- למתקדמים: עקרונות פעולה של RS-232

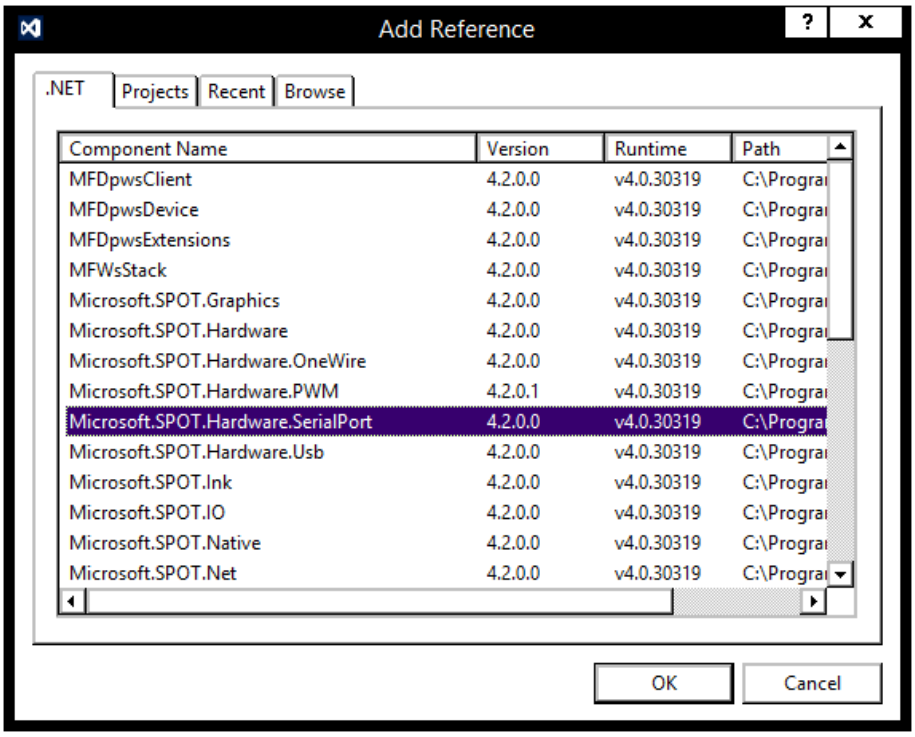
**מהלך הניסוי**

1. בניסוי זה נעשה שימוש בתקשורת טורית. ניתן לתקשר עם רכיבים רבים בצורה זאת. כך למשל: מודול Bluetooth, מודם GSM, כתיבה וקריאה מכרטיסי זיכרון ועוד.
2. נפתח פרויקט חדש לעבודה עם הבקר.
3. בכדי שנוכל לתקשר בתקשורת טורית, נוח להשתמש במחלקה בנויה הייעודית לכך. לשם כך, עלינו להכיר מחלקה זאת לסביבת פיתוח ע"י הכללת הקבצים שלה בתוך הפרויקט. ניתן לעשות זאת באופן הבא:
   a. ב Visual Studio נלחץ עם המקש הימני של העכבר על References שבחלונית ה Solution Explorer כמתואר באיור:

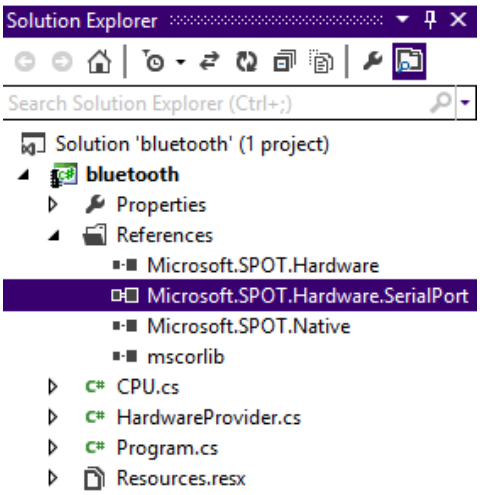



   b. בתפריט שתפתח נבחר את האפשרות העליונה Add References...

c. בחלון שיפתח נחבר את בלשונית Net. בה נגיע לרשומה:
Microsoft.SPOT.Hardware.SerialPort, נסמן אותו ונלחץ על הכפתור OK שבתחתית החלונית.




4. בחלונית Solution Explorer נוכל לראות שההרחבה התווספה בהצלחה:




5. נוסיף את ההרחבה גם ל using בתוך הקוד שבקובץ Program.cs של הפרויקט:

```
using System;
using Microsoft.SPOT;
using Microsoft.SPOT.Hardware;
using STM32;
using System.Text;
using System.Threading;
using System.IO.Ports;
```

6. נגדיר מחוץ לפונקציה הראשית ()Main את העצם החדש לתקשורת טורית:

```
    static SerialPort sp = new SerialPort(SerialPorts.COM2,
                                          9600,
                                          Parity.None,
                                          8,
                                          StopBits.One);
```

7. הסבר מקיף על השדות בתוך ה constructor (הפעולה הבונה) ניתן למצוא בנספח שבסוף הפרק.
8. נתחיל מטיפול בשידור מסר ע"י הבקר ולאחר מכן נעבור גם לקליטת מסר והפעלת לד במידה וזה מה שנדרש ממנו במסר שקלט.
9. מחוץ ל פונקציית ה Main() נגדיר את הלד אותו נרצה להדליק מאוחר יותר:

```
static OutputPort led = new OutputPort(On_Board.blue_Led, false);
```

10. בתוך ה Main() נפתח את הפורט הטורי לעבודה:

```
sp.Open();
```

11. נגדיר מערך בייטים ונכניס אליו את המידע אותו נרצה לשדר. **שימו שמחרוזת הטקסט שתשלח חייבת להסתיים ב r**

```
byte[] arr = Encoding.UTF8.GetBytes("Ready for instructions\r");
```

**לחילופין**, ניתן להגדיר משתנה מטיפוס string ועליו לעשות את ההמרה למערך:

```
string st = "Ready for instructions\r";
byte[] arr = Encoding.UTF8.GetBytes(st);
```

12. נשדר את הטקסט בתקשורת טורית רק בתנאי שנלחץ לחצן כחול

```
if (button.Read()) sp.Write(arr, 0, arr.Length);
```

13. השדה הראשון הוא מערך הבייטים אותו יש לשדר, השני ההזזה בתוכו ממנה יתחיל השידור והשלישי מספר הבייטים שיש לשדר.
14. נסיים את ה Main() עם לולאה אינסופית "שתעטוף" את השידור:

```
while (true)
            {

             if (button.Read()) sp.Write(arr, 0, arr.Length);
             }
```

15. ניתן לשדרג במקצת את התוכנית ע"י הוספת חיווי ויזואלי בלדים שע"ג הכרטיס.
16. בסה"כ, בשלב זה נקבל את התוכנית הבאה:

```
using System;
using Microsoft.SPOT;
using Microsoft.SPOT.Hardware;
using Microsoft.SPOT.Hardware.STM32;
using System.Threading;
using System.Text;
using System.IO.Ports;

namespace Task1
{
    public class Program
    {
        static SerialPort sp = new SerialPort(SerialPorts.COM2,
                                              9600,
                                              Parity.None,
```

```
                                                8,
                                                StopBits.One);
        static InputPort button = new InputPort(On_Board.blue_Button,
                                                  true,
                                                  ResistorModes.PullDown);
        static OutputPort led = new OutputPort(On_Board.blue_Led, false);

        public static void Main()
        {
            sp.Open();
            string st = "Ready for instructions\r";
            byte[] arr = Encoding.UTF8.GetBytes(st);
            while (true)
            {
                if (button.Read())
                {
                    sp.Write(arr, 0, arr.Length);
                    led.Write(true);
                    Thread.Sleep(500);
                    led.Write(false);
                    Thread.Sleep(500);
                }
            }
        }


    }
}
```

17. וודאו כי הבקר לא מחובר למחשב בכדי לא לגרום לו נזק. כל הלדים ע"ג הכרטיס צריכים להיות כבויים.
18. נתחבר לקווי ה IO של התקשורת הטורית של הבקר ע"י פלג RS-232 או חוטי חיבור נקבה-נקבה (בהתאם לסוג המודול אותו אנו מחברים לכרטיס)
19. לשם איתור הפינים המשמשים COM מסוים, ניתן להיעזר בטבלה הבאה:

| שם | Rx | Tx | Cts | Rts |
|---|---|---|---|---|
| COM1 | **A10** | **A9*** | A11 | A12 |
| COM2 | **A3** | **A2** | A0* | A1 |
| COM3 | **D9** | **D8** | D11 | *D12 |

- חיבורי ה-MicroUSB (הקטן מבין השניים) חופף עם COM1, לכן ניתן להשתמש בו רק כאשר כבל זה מנותק (כמובן לאחר צריבה).
- פין Cts ב-COM2 חופף עם הפין שאליו מחובר הכפתור הכחול. לכן בעת שימוש בפורט זה לא ניתן להשתמש בכפתור הכחול.
- פין Rts ב-COM3 חופף עם הלד הירוק. לכן בעת שימוש בפורט זה לא ניתן להשתמש בלד הירוק.
- יש לחבר גם את הפין של האדמה מהפלג של RS-232 להדק GND בערכת הפיתוח.
- את הפין VCC של מודול התקשורת יש לחבר ל 3.3V או ע"פ הוראות יצרן אחרות.

20. מאחר ואנו משתמשים בקווי העברת המידע בלבד (Rx, Tx), עדיף לבחור ב COM2 או COM3 בהם אין חפיפה בפינים אלה עם רכיבים נוספים בערכת הפיתוח. בתרגיל זה בחרנו להשתמש ב COM2.
21. מאחר ורגל השידור של מודול ה Bluetooth (או של התכן תקשורת טורית אחר) היינה רגל הקליטה של הבקר וגם רגל הקליטה של מודול ה Bluetooth היינה רגל השידור של הבקר, יש **להצליב בין שני החיבורים**.

22. אם נחבר מודול Bluetooth דרך COM2, הרי שטבלת החיבורים שלו תהיינה:

| שם הפין במודול | שם הפין בערכת פיתוח | הערות |
|---|---|---|
| VCC | 3V | |
| GND | GND | |
| TXD | A3 | |
| RXD | A2 | |

23. ואם החיבור מתבצע דרך COM3, הרי שנקבל:

| שם הפין במודול | שם הפין בערכת פיתוח | הערות |
|---|---|---|
| VCC | 3V | |
| GND | GND | |
| TXD | D9 | |
| RXD | D8 | |

24. נחבר את הערכה למחשב ע"י שני חיבורי ה USB ונצרוב את התוכנה לבקר ע"י לחיצה עם העכבר על בלחצן Start שבסרגל הפקודות.
25. לאחר הצריבה של התוכנית, נפתח תוכנה במחשב בה ניתן לקלוט מסרים מתקשורת טורית ונגדיר את מאפייניה ע"פ מה שרשמנו ב Visual Studio וצרבנו בבקר. כלומר:

| שם המאפיין | ערך |
|---|---|
| שם הפורט | COM2 |
| Baud rate | 9600 |
| Parity | none |
| Data bits | 8 |
| Stop bits | 1 |

26. אחרי שהתוכנה פתוחה ומוכנה, ניתן ללחוץ על לחצן ה reset של הבקר (הלחצן השחור) בכדי שישדר בשנית את המידע.
27. הכיתוב "`Ready for instructions`" שרשמנו יופיע בחלון התוכנה.
28. ניתן לממש את התקשורת הטורית דרך Bluetooth ולהתחבר לבקר ממחשב בעל התקן זה או מטלפון נייד באמצעות אפליקציה מתאימה כדוגמת Bloutooth terminal.

## עבודה עם מודול Bluetooth מול טלפון נייד

1. נחבר את המודול לבקר ע"פ העקרונות שנדונו בסעיפים הקודמים. לנוחיותנו נסכמם בטבלה:

| שם הרגל במודול | RXD | TXD | GND | VCC | הערות |
|---|---|---|---|---|---|
| COM1 | A9 | A10 | GND | 3V | יש לנתק את ה microUSB |
| COM2 | A2 | A3 | GND | 3V | |
| COM3 | D8 | D9 | GND | 3V | |

2. לאחר חיבורם של כ 4 הדקי המודול לכרטיס הבקר, נחבר את הבקר לחשב ע"י שני חיבורי ה USB.
3. ההסבר שלהלן מתייחס לטלפון נייד Samsung galaxy, אך מתוכו ניתן ללמוד גם לגבי טלפונים ניידים של חברות אחרות.
4. נוריד ונתקין בתוך הטלפון הנייד את האפליקציה Bluetooth terminal המאפשרת לשלוח ולקבל הודעות טקסט דרך תקשורת טורית ב Bluetooth.
5. נפעיל את האפליקציה.
6. במידה וה Bluetooth במכשיר הסלולרי שלכם כבוי, האפליקציה תקפיץ חלון בקשת אישור הפעלת Bluetooth במכשיר שיש לאשרה ע"י לחיצה על כפתור "כן"
7. יפתח חלון בו ניתן להזין טקסט ובחלקו הימני העליון יופיע כיתוב not connected
8. נלחץ על לחצן מגע שמאלי בטלפון הנייד ומתוך התפריט שתפתח נבחר את האפשרות Connect a device – Insecure
9. תפתח חלונית עם רשימת התקני ה Bluetooth המוכרים למכשיר הטלפון הנייד שלכם.
10. במידה והנכם עובדים עם מודול ה Bluetooth בפעם הראשונה, הוא לא יופיע ברשימה.
11. במידה ואתם לא רואים את ההתקן HC-06 (מודול ה Bluetooth אותו חיברנו לערכת הפיתוח) בתוך רשימת ההתקנים המוכרים למכשיר, לחצו על הלחצן Scan for devices שבתחתית הרשימה ולאחר המתנה קצרה הרכיב ייתוסף לרשימה.
12. מודול ה Bluetooth יופיע ברשימה בשם HC-06 וכתובת כגון 98:D3:31:B3:0D:7D
13. נלחץ עליו.
14. בחלקו העליון של החלון תופיע הודעה Connecting ולאחר סיום מוצלח של החיבור: connected: HC-06
15. כעת ניתן ללחוץ על הלחצן הכחול שבערכת הפיתוח ולקבל במכשיר הנייד את ההודעה אותה שולח הבקר: "Ready for instructions".
16. במידה ונרצה לשדר מידע לבקר בתקשורת טורית דרך ה Bluetooth (למרות שבשלב זה עדיין לא כתבנו אל תוך הבקר את התוכנה המעבדת את הטקסט שייקלט), פשוט נקליד את הטקסט (באנגלית) בתוך חלון האפליקציה ונלחץ על הלחצן send.

## קליטת נתונים בבקר

1. בהמשך לתוכנת השידור, נגדיר בתוך ה Main() את הפונקציה שנרצה שתתבצע כאשר יתקבל מידע דרך ערוץ התקשורת הטורית

```
sp.DataReceived += new SerialDataReceivedEventHandler(sp_DataReceived);
```

2. כעט ניתן לעבור לכתיבת הפונקציה אליה הפננו מקודם, ה sp_DataReceived. לשם כך מחוץ ל Main() נכתוב:

```
static void sp_DataReceived(object sender, SerialDataReceivedEventArgs e)
            {
                Thread.Sleep(50);
            }
```

3. נגדיר משתנה חדש מטיפוס int ונכניס בו את מספר הבייטים שיש לקרואה.

```
int num = sp.BytesToRead;
```

4. נגדיר חוצץ ממערך של בטים אליו נקראה את המידע.

```
byte[] InBuffer = new byte[num];
```

5. נבצעה את הקריאה מאובייקט התקשורת שלנו

```
sp.Read(InBuffer, 0, InBuffer.Length);
```

6. מאחר ו InBuffer מוגדר מחוץ ל Main(), ערכו ישמר בין קליטת נתונים אחת לשנייה, על כן יש למחוק את המידע הקודם ששמור בו לפני שמכניסים אליו את המידע החדש. ניתן לעשות זאת כך:

```
for (int i = 0; i < InBuffer.Length; i++)
            {
                InBuffer[i] = 0;
            }
```

7. נמיר את המערך של הבייטים שקלטנו למחרוזת string אותה נוכל להציג בחלונית Output של סביבת הפיתוח

```
char[] arr = Encoding.UTF8.GetChars(InBuffer);
string st = new string(arr);
Debug.Print(st);
```

**לחילופין**, ניתן לבצע זאת גם כך:

```
foreach (var item in InBuffer)
            {
                st += ((char)item).ToString();
            }
```

8. נבדוק העם הגיע איפשהו בתוך המידע שהתקבל הטקסט "green". במידה וכן, נדליק את הלד הירוק כמבוקש ונחזיר את ההודעה לשולח על כך שהלד הודלק.

```
if (st.IndexOf("green") != -1)
{
   led.Write(true);
   byte[] outBuf = Encoding.UTF8.GetBytes("The green LED is turned on!\r");
   sp.Write(outBuf, 0, outBuf.Length);
}
```

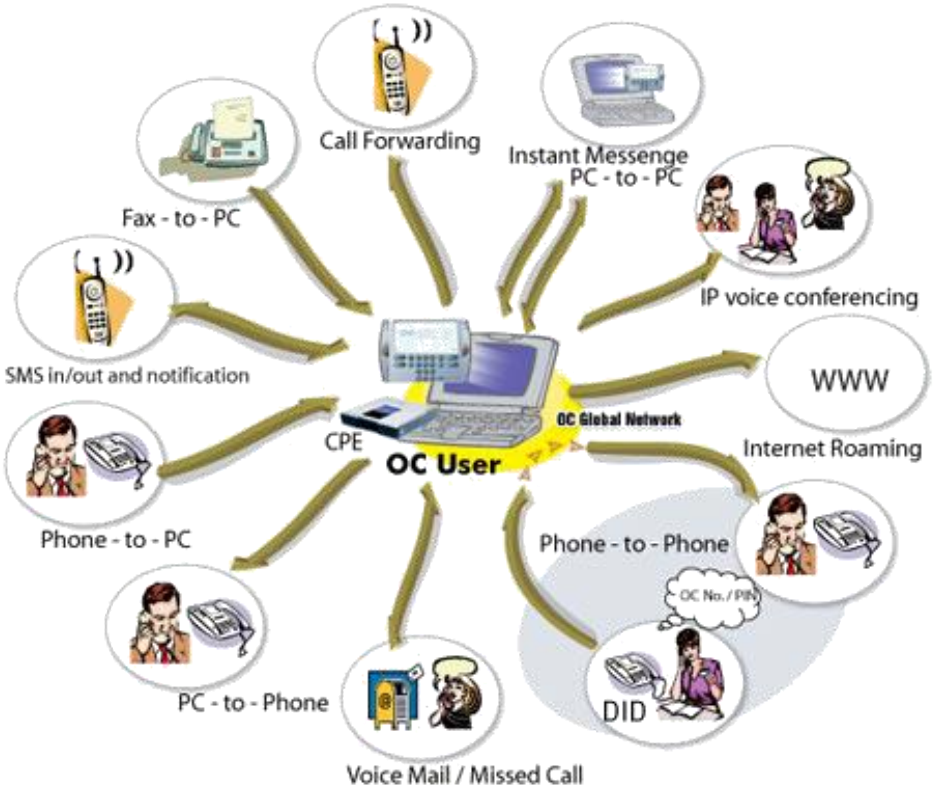
9. גם בקריאת נתונים, ניתן להוסיף לד שלא בשימוש למטרת חיווי ויזואלי.

10. בסה"כ קיבלנו את התוכנית הבאה:

```
using System;
using Microsoft.SPOT;
using Microsoft.SPOT.Hardware;
using Microsoft.SPOT.Hardware.STM32;
using System.Threading;
using System.Text;
using System.IO.Ports;

namespace Task1
{
    public class Program
    {
        static SerialPort sp = new SerialPort(SerialPorts.COM2,
                                              9600,
                                              Parity.None,
                                              8,
                                              StopBits.One);
        static InputPort button = new InputPort(On_Board.blue_Button,
                                                true,
                                                ResistorModes.PullDown);
        static OutputPort led = new OutputPort(On_Board.blue_Led, false);
        static OutputPort led2 = new OutputPort(On_Board.red_Led, false);
        static OutputPort led3 = new OutputPort(On_Board.green_Led, false);

        public static byte[] InBuffer = new byte[100];

        public static void Main()
        {
            sp.DataReceived += new SerialDataReceivedEventHandler(sp_DataReceived);
            sp.Open();
            string st = "Ready for instructions\r";
            byte[] arr = Encoding.UTF8.GetBytes(st);
            while (true)
            {
                if (button.Read())
                {
                    sp.Write(arr, 0, arr.Length);
                    led.Write(true);
                    Thread.Sleep(500);
                    led.Write(false);
                    Thread.Sleep(500);
                }
            }
        }

        static void sp_DataReceived(object sender, SerialDataReceivedEventArgs e)
        {
            led2.Write(true);
            Debug.Print("Recieved");
            Thread.Sleep(50);
            for (int i = 0; i < InBuffer.Length; i++)
            {
                InBuffer [i] = 0;
            }
            sp.Read(InBuffer, 0, sp.BytesToRead);

            string st = "";
            foreach (var item in InBuffer)
            {
                st += ((char)item).ToString();
            }
```

```
            Debug.Print(st);
            Thread.Sleep(500);
            led2.Write(false);

            if (st.IndexOf("green") != -1)
            {
                led3.Write(true);
                byte[] outBuf = Encoding.UTF8.GetBytes(
                                        "The green LED is turned on!\r");
                sp.Write(outBuf, 0, outBuf.Length);
            }
        }

    }
}
```

11. נצרוב אותה לבקר ע"י לחיצה עם העכבר על בלחצן Start שבסרגל הפקודות.
12. לאחר הצריבה של התוכנית, נשלח בתקשורת טורית מהמחשב או מהטלפון הנייד את הטקסט שנרצה שייקלט ע"י הבקר. אם נשלח את הטקסט "green", הלד הירוק ידלק ולמפעיל תשודר ההודעה "The green LED is turned on!".
13. בהצלחה!



## יצירת פורט סריאלי

פורט סריאלי הוא פורט מאוד שימושי. הוא מאפשר תקשורת ברמות TTLבפרוטוקול RS232 (פרוטוקול זה היה בשימוש לפני עידן ה-USB, במחשבים ישנים עדיין ניתן למצוא כניסת COM שמיועדת במיוחד לתקשורת מסוג זה. הפרוטוקול עדיין בשימוש כיום, בעיקר ברמות חומרה. לפני שימוש בפורט זה יש לקרוא ולהבין כיצד הוא עובד! כדי ליצור פורט סריאלי נשתמש בעצם **SerialPort**.

לדוגמא:

```
SerialPort sp = new SerialPort(SerialPorts.COM1, 9600, Parity.None,
                                                    8, StopBits.One);
```

**תכונות**:

| התכונה | טיפוס | מטרה |
|---|---|---|
| BaudRate | int | קובעת את קצב העברת הנתונים. |
| BytesToRead | int | מונה את מספר הבתים הנמצאים בבאפר הכניסה. |
| DataBits | int | מגדירה את כמות הביטים המועברים ב-"מילה" אחת. |
| Handshake | enum Handshake | מגדירה את סוג ה-handshake של הפרוטוקול. |
| IsOpen | bool | מכילה true אם כן, false אם לא. |
| Parity | enum Parity | מגדירה האם בפרוטוקול תתבצע בדיקת זוגיות או לא, ואם כן מאיזה סוג. |
| PortName | string | שומרת את שם הפורט. |
| ReadTimeout | int | קובעת במילי-שניות את כמות הזמן שהפורט ימתין לקלט לפני שישלח Exception שיבשר על ReadTimeOut. ניתן לקבוע כאין סופי. |
| StopBits | enum StopBits | מגדירה את כמות ביטי העצירה בפרוטוקול. |
| WriteTimeout | int | קובעת במילי-שניות את כמות הזמן שהפורט ימתין בניסיון לשלוח מידע לפני שישלח Exception שיבשר על WriteTimeOut. ניתן לקבוע כאין סופי. |

**פעולות**:

| הפעולה | טענת כניסה | טענת יציאה |
|---|---|---|
| SerialPort (String portName, int baudRate, Parity parity, int (dataBits, StopBits stopBits | הפעולה מקבלת שם פורט, קצב העברת מידע, הגדרת זוגיות, מספר ביטי מידע והגדרת ביטי עצירה. | הפעולה בונה עצם חדש מסוג SerialPort ומאתחלת את הפורט למצב המבוקש. |
| Open() | | הפעולה פותחת את הפורט. ללא ביצוע פעולה זו ניסיון שליחת מידע יוביל לשגיאה ! |
| Close() | | הפעולה סוגרת הפורט. |
| int Read (byte[] buffer, int (offset, int count | הפעולה מקבלת מערך של בתים, אינדקס המציין החל מאיזה תא במערך להזין נתונים וכמות הבתים שיש לקרוא. | הפעולה קוראת מהפורט הסריאלי את המידע ומשימה אותו במערך. הפעולה מחזירה את מספר הבתים שנקראו. |
| int Write (byte[] buffer, int (offset, int count | הפעולה מקבלת מערך של בתים, אינדקס המציין החל מאיזה תא במערך לשלוח וכמות הבתים שיש לכתוב. | הפעולה כותבת את המידע המבוקש לפורט הסריאלי. הפעולה מחזירה את מספר הבתים שנכתבו. |
| void DiscardInBuffer() | | הפעולה מרוקנת את תוכן באפר הקלט. |
| void DiscardOutBuffer() | | הפעולה מרוקנת את תוכן באפר הפלט. |
| void Flush() | | שולחת כל מידע שנשאר בבאפר הפלט ומרוקנת אותו. |

**מנהלי אירועים:**

| מנהל האירוע | סוג | מטרה |
|---|---|---|
| DataReceived | SerialDataReceivedEventHandler | משוגר כאשר מידע מתקבל בפורט הסריאלי. |
| ErrorReceived | SerialErrorReceivedEventHandler | משוגר כאשר מתקבלת שגיאה בפורט הסריאלי. |