**הספרייה SoftwareSerial**

**א. הספרייה**

לחומרה של הארדואינו קיימת תמיכה בתקשורת טורית בעזרת ההדקים 0 ו 1 (המתחברים למחשב דרך חיבור ה USB ). התקשורת עובדת בעזרת חומרה בתוך רכיב הנקראת UART – Universal Asynchronous Receiver Transmitter - משדר מקלט אסינכרוני אוניברסאלי. הדבר מאפשר למיקרו בקר של ATmega קליטה ושידור טוריים גם כאשר המיקרו בקר עוסק בפעולות אחרות. השידור או הקליטה ממשיכים כל עוד יש מקום בחוצץ – Buffer – של 64 בתים – Bytes - השייך לתקשורת הטורית.

הספרייה SoftwareSerial פותחה כדי לתת אפשרות לתקשורת טורית בהדקים דיגיטאליים נוספים של הארדואינו, דבר המאפשר לשכפל בעזרת תוכנה את השימוש של תקשורת טורית. ניתן להשתמש במספר פורטים טוריים בתוכנה במהירות של עד 115200 bps (ביטים בשנייה). בעזרת פרמטר ניתן גם להפוך את האותות עבור רכיבים העובדים בפרוטוקול זה.

הגרסה SoftwareSerial הכלולה בגרסת הארדואינו החל מ 1.0 ואחריה מתבססות על NewSoftwareSerial שכתב Mikal Hart .

**ב. הגבלות**

לספריה יש את ההגבלות הבאות :

אם משתמשים במספר פורטים טוריים , רק אחת יכולה לקלוט נתונים בזמן מסוים.

לא כל הדקי הארדואינו מגה ומגה 2560 תומכים בשינויי פסיקות כך שניתן להשתמש רק בהדקים הבאים עבור **RX** : 10, 11, 12 , 13 , 14, 15, 50 , 51, 52, 53, A8 עד A15 .

בארדואינו לאונרדו ובארדואינו מיקרו ניתן להשתמש כהדק RX בהדקים הבאים: 8, 9, 10,11,14, 15,16, 16 .

בארדואינו Genuino 101 הדק RX לא עובד בהדק 13 וקצב התקשורת המקסימאלי הוא 57600 bps.

אם רוצים זרימת נתונים סימולטאנית ניתן להשתמש בספריה של Paul Stoffregen's [AltSoftSerial library](http://www.pjrc.com/teensy/td_libs_AltSoftSerial.html" \t "_blank) . ספריה זו מתגברת על מספר סוגיות שב SoftwareSerial אבל יש לה מגבלות משלה.

**ג. דוגמה**

כפי שאמרנו בפרק א' הספרייה SerialSoftware מאפשרת תקשורת טורית בהדקים אחרים של כרטיסי הארדואינו ולא רק בהדקים 0 ו 1 של הכרטיסים. הדבר חשוב במיוחד כאשר יש לנו צורך להתקשר ל 2 רכיבי תקשורת טורית או כאשר יש צורך בתקשורת טורית עם רכיב אחד ולצורך debug עם המסך הטורי העובד עם ההדקים 0 ו 1 .

בדוגמה שניתן נשתמש בהדקים 10 ו 11 של כרטיסי הארדואינו או הג'ניואינו לתקשורת טורית. הדק RX בכרטיס הארדואינו , שהוא הדק דיגיטאלי 10 מתחבר להדק TX של הרכיב הטורי והדק 11 שהוא הדק TX בכרטיס הארדואינו יתחבר אל הדק RX של הרכיב הטורי.

המונחים שנשתמש בהם הם: קו תקשורת "רגילה" הם ההדקים 0 ו 1 בכרטיס הארדואינו. קו התקשורת "הנוסף" הוא ההדקים 10 ו 11 שבהם נעזרים בספריה SoftwareSerial .

#include <SoftwareSerial.h> // הכללת הספרייה

SoftwareSerial mySerial(10, 11); // RX, TX

void setup()

{

Serial.begin(57600);// ביטים בשנייה פתיחת קו התקשורת הרגילה לקצב 57600

while (!Serial); // המתנה שהפורט הטורי "הרגיל" ייפתח

Serial.println("Goodnight moon!"); // הדפסה למסך הטורי

/\*

קביעת קצב התקשורת לפורט הטורי SoftwareSerial ל 4800 ביטים בשנייה

\*/

mySerial.begin(4800);

mySerial.println("Hello, world?");

}

void loop()

{

if (mySerial.available()) // האם נקלט נתון בתקשורת הנוספת ?

{

// קרא את הנתון וכתוב אותו לקו השידור "הרגיל"

Serial.write(mySerial.read());

}

if (Serial.available()) // האם נקלט תו בתקשורת "הרגילה" ?

{

// SoftwareSerial כתיבת התו שנקלט בקו התקשורת הרגיל לקו השידור הנוסף של

mySerial.write(Serial.read());

}

}

**ד. פונקציות בספריה SoftwareSerial**

בספריה קיימות מספר פונקציות . נעבור על חלק מהן.

**ד.1 SoftwareSeria**l()

**SoftwareSerial nameOfObject (rxPin, txPin, inverse\_logic)**

משמשת ליצירת אובייקט של SoftwareSerial . nameOfObject הוא שם האובייקט של SoftwareSerial . הפרמטרים שהפונקציה מקבלת בתוך הסוגריים הם המספר של הדק ה RX (הדק קליטת הנתונים הטוריים ), הדק ה TX (הדק שידור הנתונים הטוריים) והאם משתמשים בהיפוך לוגיקה **inverse\_logic** . הארגומנט של היפוך הלוגיקה הוא אופציונאלי וברירת המחדל היא false (כלומר אם לא רשמנו כלום אז אין היפוך לוגיקה). לוגיקה הפוכה היא זו שבה מתח של 0 נחשב כאחד לוגי ומתח של כ 5 וולט נחשב כ 0 . ניתן ליצור מספר אובייקטים של SoftwareSerial אבל רק אחד יהיה פעיל בזמן נתון.

כדי לאפשר תקשורת יש לקרא לפונקציה SoftwareSerial.begin() .

דוגמה :

#include <SoftwareSerial.h> // הכללת הקובץ

#define rxPin 2

#define txPin 3

// mySerial קביעת אובייקט טורי חדש בשם

SoftwareSerial mySerial (rxPin, txPin);

**ד.2 SoftwareSerial.begin(baud)**

קביעת קצב התקשורת הטורית . הפרמטר שהפונקציה מקבלת היא את קצב התקשורת שהוא משתנה מטיפוס long . קצבים אפשריים הם : 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 31250, 38400, 57600, 115200. . הפונקציה איננה מחזירה ערך.

דוגמה : שוב נשתמש בהדקים 10 ו 11 כהדקי RX ו TX בהתאמה .

// הכללת הספרייה כדי שנוכל להשתמש בפונקציות שלה

#include <SoftwareSerial.h>

#define rxPin 10

#define txPin 11

// ואתחולו במספר של ההדק של הקליטה והשידור mySerial קביעת אובייקט טורי חדש בשם

SoftwareSerial mySerial = SoftwareSerial(rxPin, txPin);

void setup()

{

// כפלט TX כקלט והדק RX קביעת תפקוד הדק ה

pinMode(rxPin, INPUT);

pinMode(txPin, OUTPUT);

// ל 9600 ביטים בשנייה SoftwareSerial קביעת קצב התקשורת של פורט

mySerial.begin(9600);

}

void loop()

{

// ...

}

**ד.3 SoftwareSerial.read( )**

הפונקציה מחזירה את התו שנקלט בהדק RX של פורט SoftwareSerial . במקרה שהגדרנו מספר אובייקטים של קליטה טורית , רק פורט אחד של SoftwareSerial יכול לקלוט נתון בזמן כלשהו ( ניתן לבחור איזה פורט בעזרת פונקציית listen ). הפונקציה לא מקבלת פרמטר ומחזירה את התו שנקלט או -1 אם לא נקלט תו .

**דוגמה :**

SoftwareSerial mySerial(10,11);

void setup()

{

mySerial.begin(9600);

}

void loop()

{

char c = mySerial.read(); // קריאה של התו.

}

**ד.4 SoftwareSerial.print(data )**

מדפיסה נתון אל הדק השידור של פורט SoftwareSerial . עובדת כמו פונקציית Serial.print() של התקשורת הטורית "הרגילה" . הפרמטר ששולחים לפונקציה יכול להיות קבוע, משתנה, או ביטוי כלשהו בדומה להדפסה "רגילה" וניתן לציין את הבסיס שבו יודפס הערך. הפונקציה מחזירה משתנה מטיפוס byte המציין כמה בתים נכתבו. קריאת מספר זה היא אופציונאלית.

דוגמה : בהנחה שבהדק אנאלוגי A0 יש מתח שאותו רוצים למדוד.

SoftwareSerial mySerial(10,11); // SoftwareSerial השייך ל mySerial הגדרת אובייקט בשם

int analogValue; // משתנה שיכיל את הערך של המתח שמודדים

void setup()

{

serial.begin(9600); // התחלת התקשורת "הרגילה"

}

void loop()

{

// קריאת המתח והכנסתו למשתנה

analogValue = analogRead(A0);

// הדפסה למסך הטורי את הערך של המתח במספר פורמטים :

mySerial.print(analogValue); // הדפסה עשרונית

mySerial.print("\t"); // tab עוברים לשדה ההדפסה הבא. כמו לחיצה במקלדת על

mySerial.print(analogValue, DEC); // הדפסה עשרונית

mySerial.print("\t"); // tab עוברים לשדה ההדפסה הבא. כמו לחיצה במקלדת על

mySerial.print(analogValue, HEX); // הדפסת הערך בהקסה דצימאלי (בסיס 16)

mySerial.print("\t"); // tab עוברים לשדה ההדפסה הבא. כמו לחיצה במקלדת על

mySerial.print(analogValue, OCT); // הדפסת הערך בחשבון אוקטאלי ( בסיס 8 ).

mySerial.print("\t");

mySerial.print(analogValue, BIN); // הדפסת הערך בבינארי

mySerial.print("\t"); // tab עוברים לשדה ההדפסה הבא. כמו לחיצה במקלדת על

mySerial.println(); // מעבר לתחילת השורה הבאה

// השהייה של 10 אלפיות שנייה לפני הקריאה הבאה.

delay(10);

}

**ד.5 SoftwareSerial.println(data )**

מדפיסה נתון אל הדק השידור של פורט SoftwareSerial ויורדים לתחילת השורה הבאה . עובדת כמו פונקציית Serial.println() של התקשורת הטורית "הרגילה" . הפרמטר ששולחים לפונקציה יכול להיות קבוע, משתנה, או ביטוי כלשהו בדומה להדפסה "רגילה" וניתן לציין את הבסיס שבו יודפס הערך. הפונקציה מחזירה משתנה מטיפוס byte המציין כמה בתים נכתבו. קריאת מספר זה היא אופציונאלית.

**ד.6 SoftwareSerial.available( )**

מחזירה את מספר הבתים (תווים) שניתן לקרא מהפורט הטורי. נתונים אלו נתקבלו ואוחסנו בחוצץ (buffer ) הקליטה הטורית . הפונקציה איננה מקבלת ערכים.

התחביר :

mySerial.available()

דוגמה :

// הכללת הספרייה כדי שנוכל להשתמש בפונקציות שלה

#include <SoftwareSerial.h>

#define rxPin 10

#define txPin 11

// ואתחולו במספר של ההדק של הקליטה והשידור mySerial קביעת אובייקט טורי חדש בשם

SoftwareSerial mySerial = SoftwareSerial(rxPin, txPin);

void setup()

{

// כפלט TX כקלט והדק RX קביעת תפקוד הדק ה

pinMode(rxPin, INPUT);

pinMode(txPin, OUTPUT);

// ל 9600 ביטים בשנייה SoftwareSerial קביעת קצב התקשורת של פורט

mySerial.begin(9600);

}

void loop()

{

if (mySerial.available()>0) // האם מספר התווים שנקלטו גדול מ 0 ?

{

mySerial.read(); // קריאת התו שנקלט

}

}

**ד.7 .8 SoftwareSerial.listen( ) ו SoftwareSerial.isListening**

במידה והגדרנו מספר פורטים לתקשורת טורית, רק אחד יכול "להקשיב" בזמן נתון. הפונקציה קובעת מיהו הפורט הפעיל . הנתונים שיגיעו לפורטים האחרים לא יהיו רלוונטיים (לא נקלטים).

**התחביר לפונקציה :** mySerial.listen( )

הפונקציה קובעת את הפורט הנמצא בהקשבה . במקרה הזה הוגדר אובייקט mySerial והוא זה הנמצא בהקשבה. הפונקציה איננה מחזירה ערך.

**התחביר לפונקציה : mySerial.isListening()** .

הפונקציה בודקת האם הפורט הטורי mySerial נמצא בהקשבה. לא שולחים לפונקציה פרמטרים והיא מחזירה ערך בוליאני של 1 אם הפורט mySerial אכן פעיל ונמצא בהקשבה, או 0 אם הוא לא פעיל.

**דוגמה :**

#include <SoftwareSerial.h>

// עם הדקי קליטה ושידור 10 ו 11 בהתאמה portOne הגדרת אובייקט אחד הנקרא

SoftwareSerial portOne(10, 11);

// עם הדקי קליטה ושידור 8 ו 9 בהתאמה portTwo הגדרת אובייקט שני הנקרא

SoftwareSerial portTwo(8, 9);

void setup()

{

// אתחול תקשורת טורית "רגילה" עבור המסך הטורי

Serial.begin(9600);

// ואתחולם לקצב 9600 ביטים בשנייה SoftwareSerial התחלת 2 פורטים של תקשורת של

portOne.begin(9600);

portTwo.begin(9600);

}

void loop()

{

portOne.listen(); // הוא זה שנמצא בהקשבה וקולט נתונים portOne קביעה שפורט הנקרא

if (portOne.isListening()) // בהקשבה ? portOne האם

{

Serial.println("Port One is listening!"); // הדפסה למסך הטורי

}

else

{

Serial.println("Port One is not listening!");

}

if (portTwo.isListening()) // בהקשבה ? portTwo האם

{

Serial.println("Port Two is listening!");

}

else

{

Serial.println("Port Two is not listening!");

}

}

**ד.9 SoftwareSerial.peek()**

מבצעת אותה פעולה כמו הפונקציה SoftwareSerial.read(). היא מחזירה תו שנקלט בהדק RX של פורט SoftwareSerial . השוני הוא שקריאות עוקבות של הפונקציה תחזיר את אותו התו .

דוגמה :

SoftwareSerial mySerial(10,11);

void setup()

{

mySerial.begin(9600);

}

void loop()

{

char c = mySerial.peek();

}

**ד.10 SoftwareSerialwrite()**

מדפיסה נתון להדק השידור של פורט SoftwareSerial . עובדת בדומה לפונקציה Serial.write . הפרמטר ששולחים לפונקציה יכול להיות קבוע, משתנה, או ביטוי כלשהו בדומה להדפסה "רגילה" וניתן לציין את הבסיס שבו יודפס הערך. הפונקציה מחזירה משתנה מטיפוס byte המציין כמה בתים נכתבו. קריאת מספר זה היא אופציונאלית.

דוגמה :

SoftwareSerial mySerial(10, 11);

void setup()

{

mySerial.begin(9600);

}

void loop()

{

mySerial.write(45); // TX שולחים את הערך 45 לשידור בהדק

int bytesSent = mySerial.write(“hello”); //שולחים מחרוזת ומקבלים את כמות הבתים שנכתבה - את אורך המחרוזת //

}