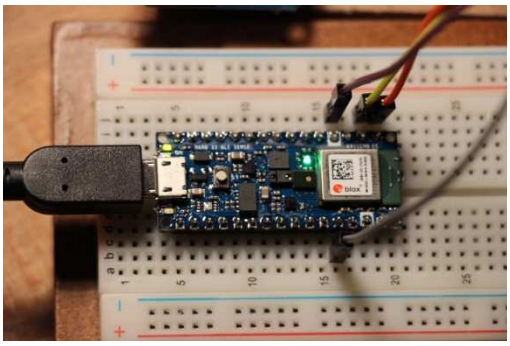
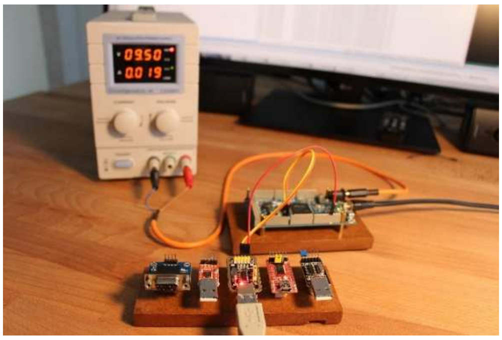
Getting started with WawiLib using the Arduino serial ports





1 Introduction

1.1 Objective of this document

The objective of this document is to describe the use of serial communication between Arduino boards and the WawiLib software on the PC. This document focuses on serial communication via USB to Serial converter devices, RS232C and Native USB communication. For communication using the Arduino USB programming port: read the document “Getting started with WawiLib over the Arduino programming port”.

The first part of this document contains some theory as well as general information on serial communication. The next chapters contain multiple demos for Arduino and ESP boards.

1.2 Software and hardware requirements

The WawiLib and the Arduino IDE (in this example 1.8.10) need to be installed on your PC. The demos run with licensed and unlicensed versions of WawiLib. This document uses the licensed version as it is able to monitor and modify multiple variables at the same time.

This document contains references to multiple types of Arduino boards. Essential is the difference between those that are 5V based, such as the UNO and the MEGA2560, and those that are 3.3V based such as the DUE, the NodeMCU and the MKR1000.

Concerning general hardware, you need: Arduino boards, a USB programming cable and a Windows PC (32 or 64 bit). On top of this general hardware, you need a way to connect the Arduino serial port to your PC. Depending on the interfaces of your PC and the available hardware, there are multiple options:

- USB to serial converters to convert the TTL/CMOS signals from your Arduino to USB.
- RS232-RS485 V1R1 Arduino shield and a serial cable to connect to a RS232C port on the PC.
- AUSB cable to connect the USB-native DUE port to the USB-programming port of the PC.

WawiLib has been tested with a variety of configurations and converters. In these demos, I will explain in detail how to use them.

This document assumes that you have installed the software as described in “Getting started with WawiLib via the Arduino programming interface”.

1.3 Required user experience

This demo assumes that the you know how to edit, compile and download Arduino programs. You should also have studied “Getting started with WawiLib via the Arduino programming interface”. So, you should be able to go online and offline, scan for interfaces and monitor variables.

Some experience in serial communication is also required depending on the type of interface you want to use. However, the hardware part will be explained in detail so even if your knowledge is not very extensive, you should be able to get the demos up and running.

2 Serial communication overview

2.1 General topics

Serial communication, especially RS232, was originally introduced in the 1960s. Without going into too much details, the principle is as follows:

Data is sent byte by byte to a chip, this chip serializes each byte so it can be sent bit by bit. The chip (a UART) also adds start bits, stop bits and parity bits according to the communication settings.

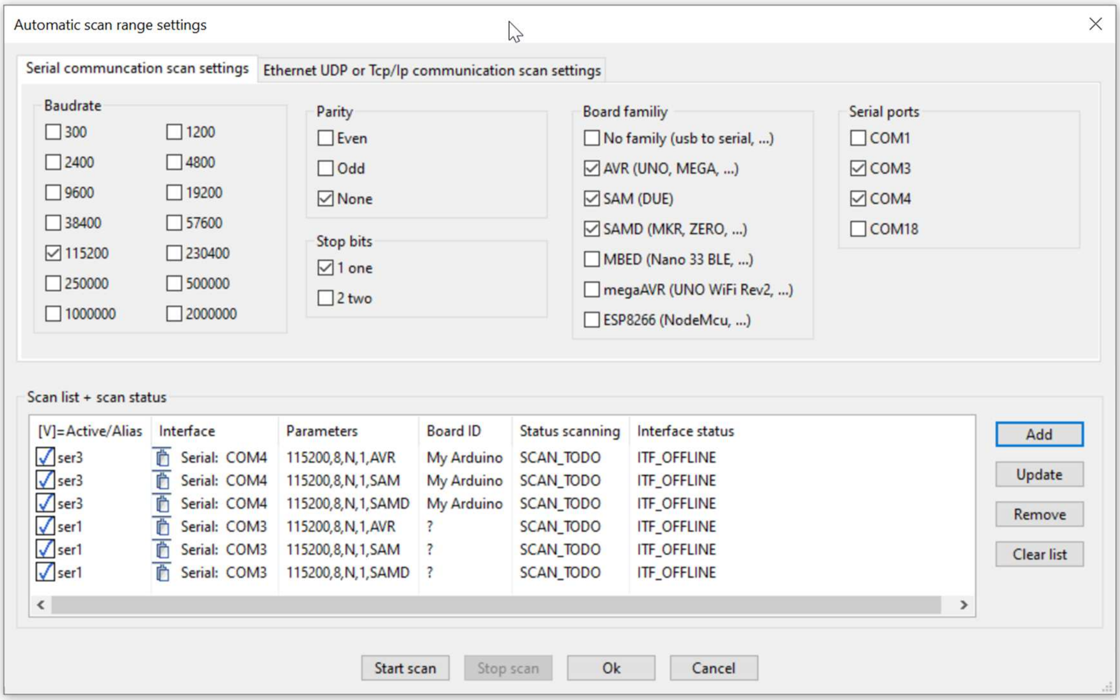
On the receiver end, the bit by bit signal is reassembled into bytes in a similar way. UARTs basically work with TTL of CMOS logic. In order to travel larger distances, the TTL/CMOS signal can be electrically boosted. Depending on the type of boost, you get different standards RS232, RS422 RS485, etc.

Some standards (RS485) work half duplex: both communication partners can send, but only one can send at the same time. RS232 works full duplex: there are 2 separate communication lanes back and forth, so both parties can send and receive at the same time.

Apart from the communication signals (TX=send, RX=Receive) some serial interfaces also have hardware control signals (RTS, CTS, DTR and DSR). These signals were originally designed to control a modem, a terminal or a printer. Control signals are typically used to implement some kind of handshake to prevent buffer overrun at the receiving side.

Today, there are USB to serial converters. They convert USB packets to serial data. These converters have an RX/TX TTL/CMOS interface as described above on one side and a USB interface on the other side. Some of them also provide one or more control signals outputs such as RTS.

The RTS& DTR signal scan be used to reset the Arduino, to define the direction of the communication (example RS485) or for some other kind of purpose. This is the reason why the hardware settings dialog of WawiLib contains different options for "Board family".



Automatic scan range settings
Serial communcation scan settings
Ethernet UDP or Tcp/Ip communication scan settings
Add
Update
Remove
Clear list
Start scan
Stop scan
Ok
Cancel

Baudrate
□ 300 □ 1200
□ 2400 □ 4800
□ 9600 □ 19200
□ 38400 □ 57600
☑ 115200 □ 230400
□ 250000 □ 500000
□ 1000000 □ 2000000

Parity
□ Even
□ Odd
☑ None

Stop bits
☑ 1 one
□ 2 two

Board familiy
□ No family (usb to serial, ...)
☑ AVR (UNO, MEGA, ...)
☑ SAM (DUE)
☑ SAMD (MKR, ZERO, ...)
□ MBED (Nano 33 BLE, ...)
□ megaAVR (UNO WiFi Rev2, ...)
□ ESP8266 (NodeMcu, ...)

Serial ports
□ COM1
☑ COM3
☑ COM4
□ COM18

Scan list + scan status

[V]=Active/Alias	Interface	Parameters	Board ID	Status scanning	Interface status
☑ ser3	Serial: COM4	115200,8,N,1,AVR	My Arduino	SCAN_TODO	ITF_OFFLINE
☑ ser3	Serial: COM4	115200,8,N,1,SAM	My Arduino	SCAN_TODO	ITF_OFFLINE
☑ ser3	Serial: COM4	115200,8,N,1,SAMD	My Arduino	SCAN_TODO	ITF_OFFLINE
☑ ser1	Serial: COM3	115200,8,N,1,AVR	?	SCAN_TODO	ITF_OFFLINE
☑ ser1	Serial: COM3	115200,8,N,1,SAM	?	SCAN_TODO	ITF_OFFLINE
☑ ser1	Serial: COM3	115200,8,N,1,SAMD	?	SCAN_TODO	ITF_OFFLINE

Fig. 2.1. Board family to manage the different RTS&DTR communication settings.

If you do not know what kind of board to choose, you can check them all in the dialog box, press Add, and WawiLib will try and test the settings one by one until the appropriate settings have been found.

In the next chapters, I will describe various serial communication hardware options. I will start with the older devices and end with the more recent ones (USB to serial converters). Reading the first chapters can be useful even if you are not interested in the older interfaces. There will be some relevant info about the Arduino boards in these chapters.

2.2 RS232C Communication

2.2.1 TheRS232RS485 V1r1 shield

The first Arduino I ever bought was the Arduino UNO. I purchased it together with the V1r1 shield. This shield provides an RS232 and an RS485 interface.

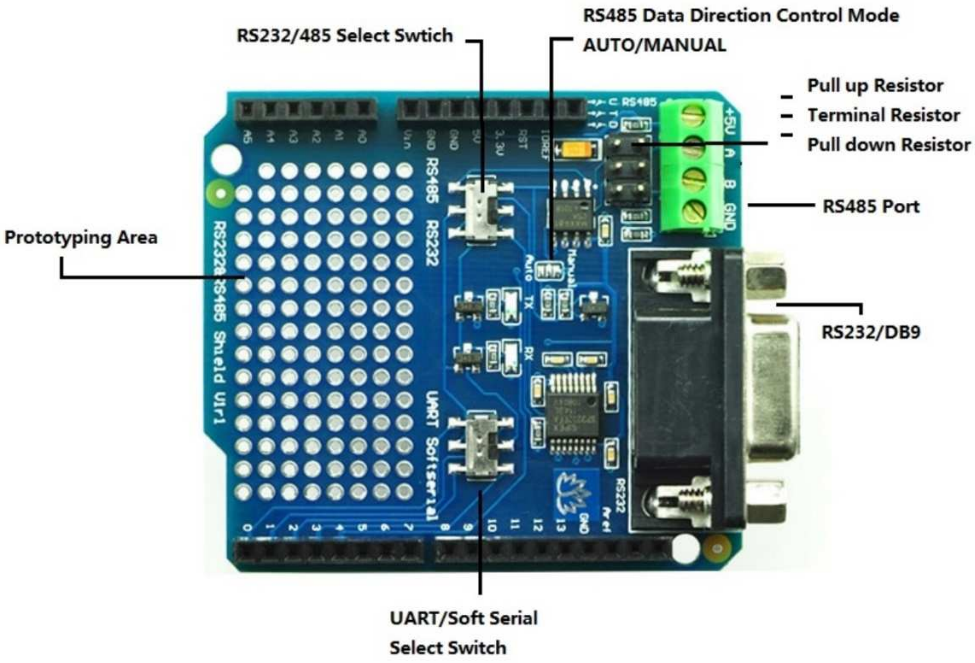


Fig. 2.2. the V1r1 shield.

If you mount a V1r1 shield on an Arduino UNO and you set the dip switch SoftwareSerial-UART to UART, you will see that downloading your sketch fails. So, you will have to set the switch to SoftwareSerial and then, after downloading, set it back to UART.

The reason is that the UART I/O of the CPU are connected to pin 0 and 1 (RX and TX). In parallel, they are also connected to the 16U2 processor on the UNO board. This 16U2 processor is used to provide the Arduino USB programming interface. The V1r1 board uses the same pins 0 and 1 to connect to the same UART. So, if you put the V1r1 shield in UART mode, you disturb the signal exchange between the 16U2 and the main processor. If this communication is disturbed, programming will fail.

The V1r1 shield is a shield that is not compatible with the 3.3V standard of the DUE. You can mount it mechanically on the DUE but it will not function properly. It might even damage your DUE (so do not try this). Somebody I know very well tried it and saw that the RX green and the TX red LEDs of the shield kept burning continuously and the serial communication was not working. The fate of the DUE that underwent this ordeal remains unknown…

You might wonder if there are any PCs today that still have a 9-pin RS232 interface? For sure you can purchase PCIe cards that have these interfaces (Startech.com and others supply these cards). If you buy one, make sure it has drivers compatible with your operating system. Some high-end workstations, such as the DELL Precision range, still have R232 standard on the motherboard.

The connection cable you need is a 9-pin DSUB straight male-female cable. The V1r1 shield has a 9 pin DSUB female connector and the PC a 9 pin DSUB male connector. The nice thing about the V1r1 shield is that it has a TX LED (red) and a RX LED (green). During software development, these LEDs are very handy as they help you to see if live data is sent and received by the shield.

The shield can work in UART mode or in SoftwareSerial mode. UART mode means that there is a hardware shift register on the Arduino processor that shifts out bytes bit by bit according to the configured baud rate and serial communication parameters (UART). SoftwareSerial is an Arduino library that emulates the UART function in software at low baud rates (maximum about 19200 bits/second). Electrically, depending on the state of a switch, the shield connects pins 2 & 3 or pins 0 & 1 to the RS232 and RS485 signal conversion chips on the shield.

The reason that SoftwareSerial exists is that some boards, such as the UNO, have a processor that has only 1 integrated UART. If you want a second serial communication link, you can use SoftwareSerial. Do not overestimate the capabilities of SoftwareSerial: it creates additional load on the CPU. Better is to go for a board that has multiple hardware UARTs such as the MEGA2560. Be careful with SoftwareSerial as it can only work with some I/O pins and not with others.

Another major disadvantage of SoftwareSerial is that it is unable to work ‘Full Duplex’, this means that you can send and receive data over the serial communication at the same time.

WawiLib does support SoftwareSerial communication but with limited functionality. You cannot use the .print() function to display your output in the output window as it needs Full Duplex communication to do so. Therefore, in case of the use of SoftwareSerial, do not use .print() functions in your sketch.

As or if you do not need .print(), you can use the lightweight object *WawiSerialUsbLight* in your sketch, it is included in the same header as WawiSerialUsb but it is about 30% smaller because it does not contain the code for the .print() functions.

Multiple SoftwareSerial communication samples are provided with the WawiSerialUsb library. Later in this document, I will present a detailed demo with SoftwareSerial.

The V1r1 shield also has an RS485 interface. RS485 is a bus interface that is typically used in a multi-drop configuration where multiple RS485 nodes are attached to a 2 wire RS485 bus.

In order to send (broadcast) a message over the bus, a node switches to low impedance and determines the state of the bus (logic 0 or 1) based on the TX signal of its UART. The other nodes on the bus remain in a state of high impedance. In this state, they probe the bus and translate its voltage level to levels compatible with the RX pin of their UART. This way, these nodes receive the transmitted data.

Be careful with the RS485 interface on the V1r1 shield: the direction of the data is by default generated automatically based on the TX signal and not controlled via a dedicated hardware signal.

If you want to implement a protocol with delicate timing constraints, the automatic concept is not ok. In that case, you need to take manual control of the data direction: 1) change a solder bridge on the shield and 2) modify the serial library so it activates the UART-functions that can provide direction control. On Youtube, can find a post (search for JOHI + Profibus) where I used this technique to build a simple Profibus-DP slave with an Arduino Mega.

The RS485 shield that comes with the MKRXXX (MKR1000, MKR1010) series does not have the issue presented above. This shield has a dedicated library that also controls the state of the transceiver on the shield. For protocols with very delicate timing, even this solution used with the standard Arduino library will not guarantee success.

2.2.2 RS232&RS485 V1r1 shield on Arduino UNO with hardware serial lib

2.2.2.1 Required hardware

- Arduino UNO.
- RS232 & RS485 V1r1 Shield.
- RS232 Serial cable straight male-female (pin 1 to 1, pin 2 to 2 etc.).
- USB A to USB B cable to program the Arduino.

2.2.2.2 Hardware connections

✓ Mount an RS232 & RS485 shield V1r1 on your UNO or Mega.

✓ Connect the V1r1 RS232 9 pin DSUB connector to a serial port on your PC.

✓ Use the USB cable to program and supply the Arduino with 5V.

✓ Select the mode RS232 on the V1r1 shield.

2.2.2.3 Load sketch

✓ Open the example via menu “File\Examples\WawiSerialUsb\WawiBlinkSerial” in the Arduino IDE.

✓ Make sure the switch UART-SoftSerial on the V1r1 shield is in the position “SoftSerial”.

✓ Compile and download the sample.

✓ Make sure the switch UART-SoftSerial on the V1r1 shield is in the position “UART”.

```
/*
* Project Name: WawiBlinkSerial
* File: WawiBlinkSerial.ino
*
* Description: demo file library for WawiSerialUsb libary.
* Blinks LED at IO 13 with variable on and off periods.
* Use a serial connection via V1R1 board or similar to make a connection with the
Arduino board.
* Counts the number of blinks.
* Variables can be checked & modified with the WawiLib-PC software.
*
* Author: John Gijs.
* Created: Dec 2020
* More info: www.sylvestersolutions.com
* Technical support: support@sylvestersolutions.com
* Additional info: info@sylvestersolutions.com
*/

#include <WawiSerialUsb.h>

WawiSerialUsb WawiSrv;

#define LED 13

// test variables for demo:
int delayOn = 500;
int delayOff = 500;
int blinkCounter = 0;

// make variables of interest known to WawiLib:
// this function is used in WawiSrv.begin(....)
void wawiVarDef()
{
    WawiSrv.wawiVar(delayOn);
    WawiSrv.wawiVar(delayOff);
    WawiSrv.wawiVar(blinkCounter);
}
```

```
void setup()
{
    Serial.begin(115200);
    WawiSrv.begin(wawiVarDef, Serial, "My Arduino");
    pinMode(LED, OUTPUT);
}

void loop()
{
    blinkCounter++;
    WawiSrv.print("WawiSrv.Print() demo in loop() function, blinkcounter = ");
    WawiSrv.println(blinkCounter);

    WawiSrv.println("LED is ON.");
    digitalWrite(LED, HIGH);
    WawiSrv.delay(delayOn);

    WawiSrv.println("LED is OFF.");
    digitalWrite(LED, LOW);
    WawiSrv.delay(delayOff);

    WawiSrv.loop();
}
```

Fig 2.3. The WawiBlinkSerial sketch source code.

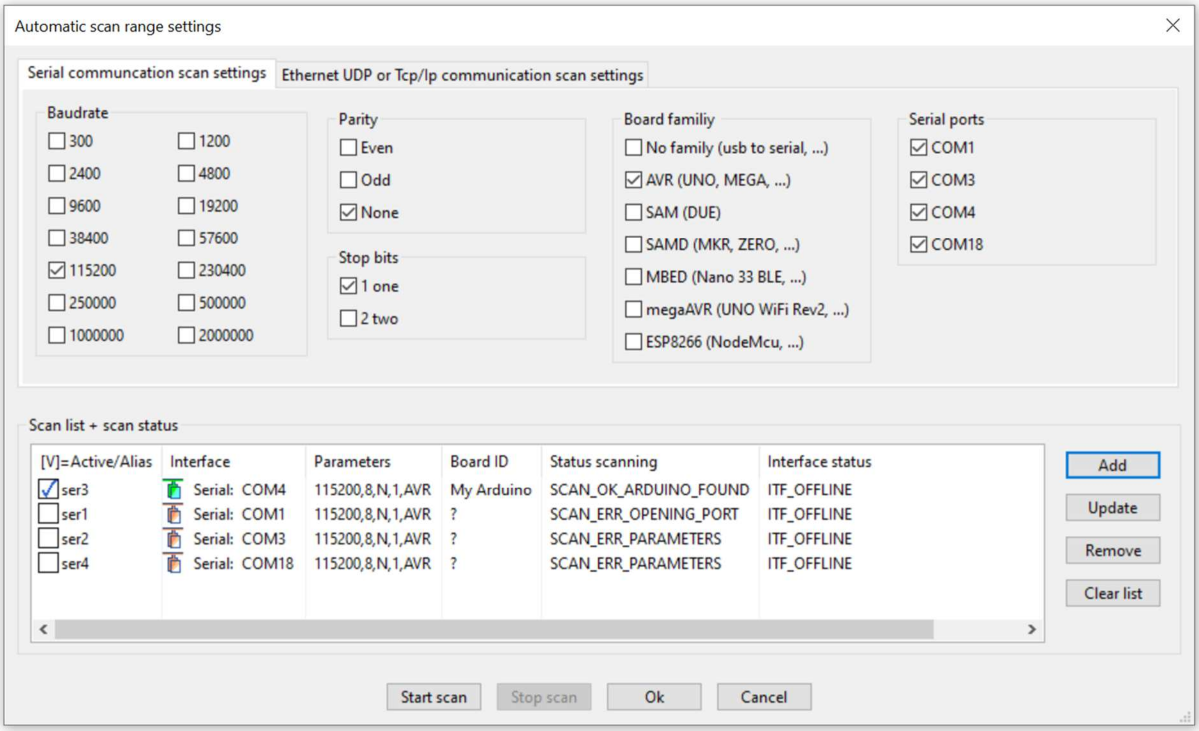
2.2.2.4 Scan ports with WawiLib

✓ Open the automatic scan range settings dialog box (figure below) in WawiLib.

✓ Fill in the table in as indicated in the figure below (select all available serial ports).

✓ Press “Add”.

✓ Press “Start scan”.



Automatic scan range settings

Serial communcation scan settings | Ethernet UDP or Tcp/Ip communication scan settings

Baudrate
□ 300 □ 1200
□ 2400 □ 4800
□ 9600 □ 19200
□ 38400 □ 57600
☑ 115200 □ 230400
□ 250000 □ 500000
□ 1000000 □ 2000000

Parity
□ Even
□ Odd
☑ None

Stop bits
☑ 1 one
□ 2 two

Board familiy
□ No family (usb to serial, ...)
☑ AVR (UNO, MEGA, ...)
□ SAM (DUE)
□ SAMD (MKR, ZERO, ...)
□ MBED (Nano 33 BLE, ...)
□ megaAVR (UNO WiFi Rev2, ...)
□ ESP8266 (NodeMcu, ...)

Serial ports
☑ COM1
☑ COM3
☑ COM4
☑ COM18

Scan list + scan status

[V]=Active/Alias	Interface	Parameters	Board ID	Status scanning	Interface status
☑ ser3	Serial: COM4	115200,8,N,1,AVR	My Arduino	SCAN_OK_ARDUINO_FOUND	ITF_OFFLINE
□ ser1	Serial: COM1	115200,8,N,1,AVR	?	SCAN_ERR_OPENING_PORT	ITF_OFFLINE
□ ser2	Serial: COM3	115200,8,N,1,AVR	?	SCAN_ERR_PARAMETERS	ITF_OFFLINE
□ ser4	Serial: COM18	115200,8,N,1,AVR	?	SCAN_ERR_PARAMETERS	ITF_OFFLINE

Fig 2.4. Scan range settings dialog box with multiple port selected for scanning.

⇨ In the table “Scan list + scan status”, one of the icons in the “interface” column should turn green, indicating that a board with a WawiLib serial communication interface has been found.

✓ Click right on the table and select “remove inactive”.

⇨ The interfaces that were not successfully scanned will be removed from the list.

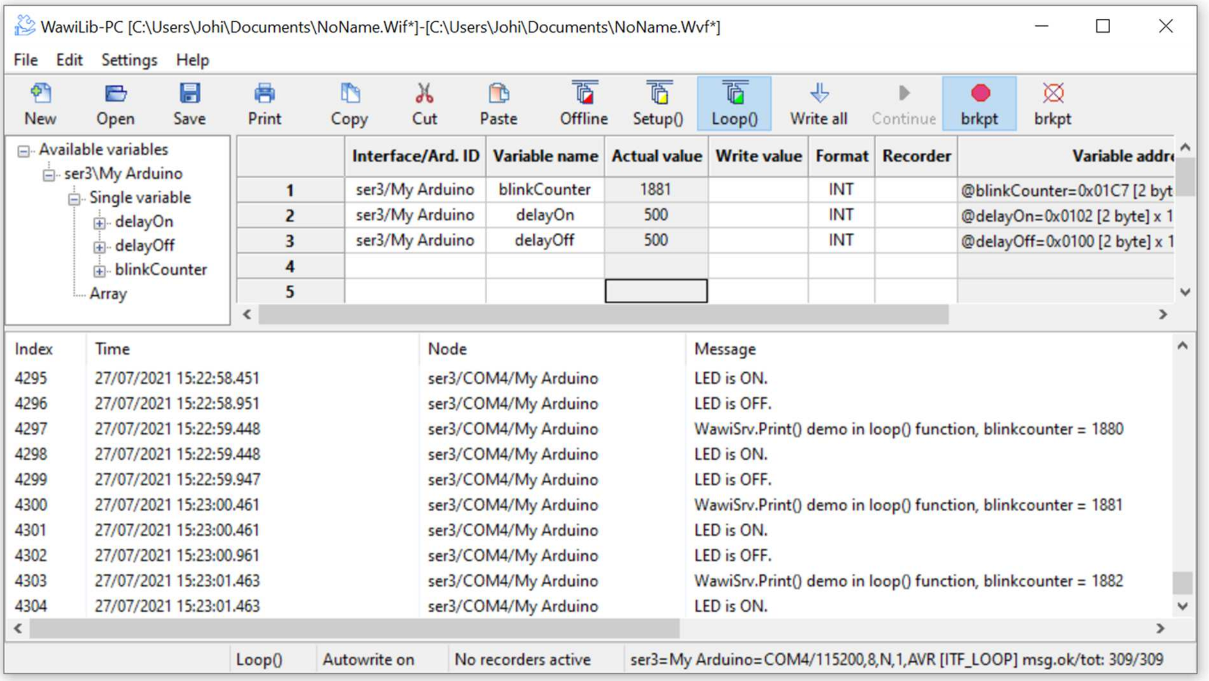
✓ Press OK.

2.2.2.5 Monitor variables with WawiLib

✓ Press “Setup()” in the tool bar;

✓ Enter the variables to the main grid as indicated in the table below.

✓ Alternative: Use drag & drop to drag the variables from the tree to the grid table.


WawiLib-PC [C:\Users\Johi\Documents\NoName.Wif*]-[C:\Users\Johi\Documents\NoName.Wvf*]
File Edit Settings Help
New Open Save Print Copy Cut Paste Offline Setup() Loop() Write all Continue brkpt brkpt
Available variables
ser3\My Arduino
Single variable
delayOn
delayOff
blinkCounter
Array

	Interface/Ard. ID	Variable name	Actual value	Write value	Format	Recorder	Variable addr
1	ser3/My Arduino	blinkCounter	1881		INT		@blinkCounter=0x01C7 [2 byt
2	ser3/My Arduino	delayOn	500		INT		@delayOn=0x0102 [2 byte] x 1
3	ser3/My Arduino	delayOff	500		INT		@delayOff=0x0100 [2 byte] x 1
4							
5							

```
Index  Time                      Node                    Message
4295   27/07/2021 15:22:58.451   ser3/COM4/My Arduino    LED is ON.
4296   27/07/2021 15:22:58.951   ser3/COM4/My Arduino    LED is OFF.
4297   27/07/2021 15:22:59.448   ser3/COM4/My Arduino    WawiSrv.Print() demo in loop() function, blinkcounter = 1880
4298   27/07/2021 15:22:59.448   ser3/COM4/My Arduino    LED is ON.
4299   27/07/2021 15:22:59.947   ser3/COM4/My Arduino    LED is OFF.
4300   27/07/2021 15:23:00.461   ser3/COM4/My Arduino    WawiSrv.Print() demo in loop() function, blinkcounter = 1881
4301   27/07/2021 15:23:00.461   ser3/COM4/My Arduino    LED is ON.
4302   27/07/2021 15:23:00.961   ser3/COM4/My Arduino    LED is OFF.
4303   27/07/2021 15:23:01.463   ser3/COM4/My Arduino    WawiSrv.Print() demo in loop() function, blinkcounter = 1882
4304   27/07/2021 15:23:01.463   ser3/COM4/My Arduino    LED is ON.
```

Loop() | Autowrite on | No recorders active | ser3=My Arduino=COM4/115200,8,N,1,AVR [ITF_LOOP] msg.ok/tot: 309/309

Fig 2.5. Main variable grid with variables to be monitored and modified.

✓ Enable “Display Print messages” using the popup menu of the output window (right click on the output window to make the menu appear).

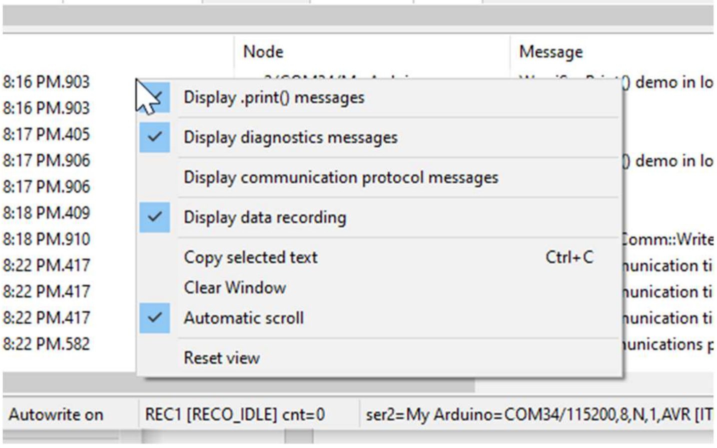
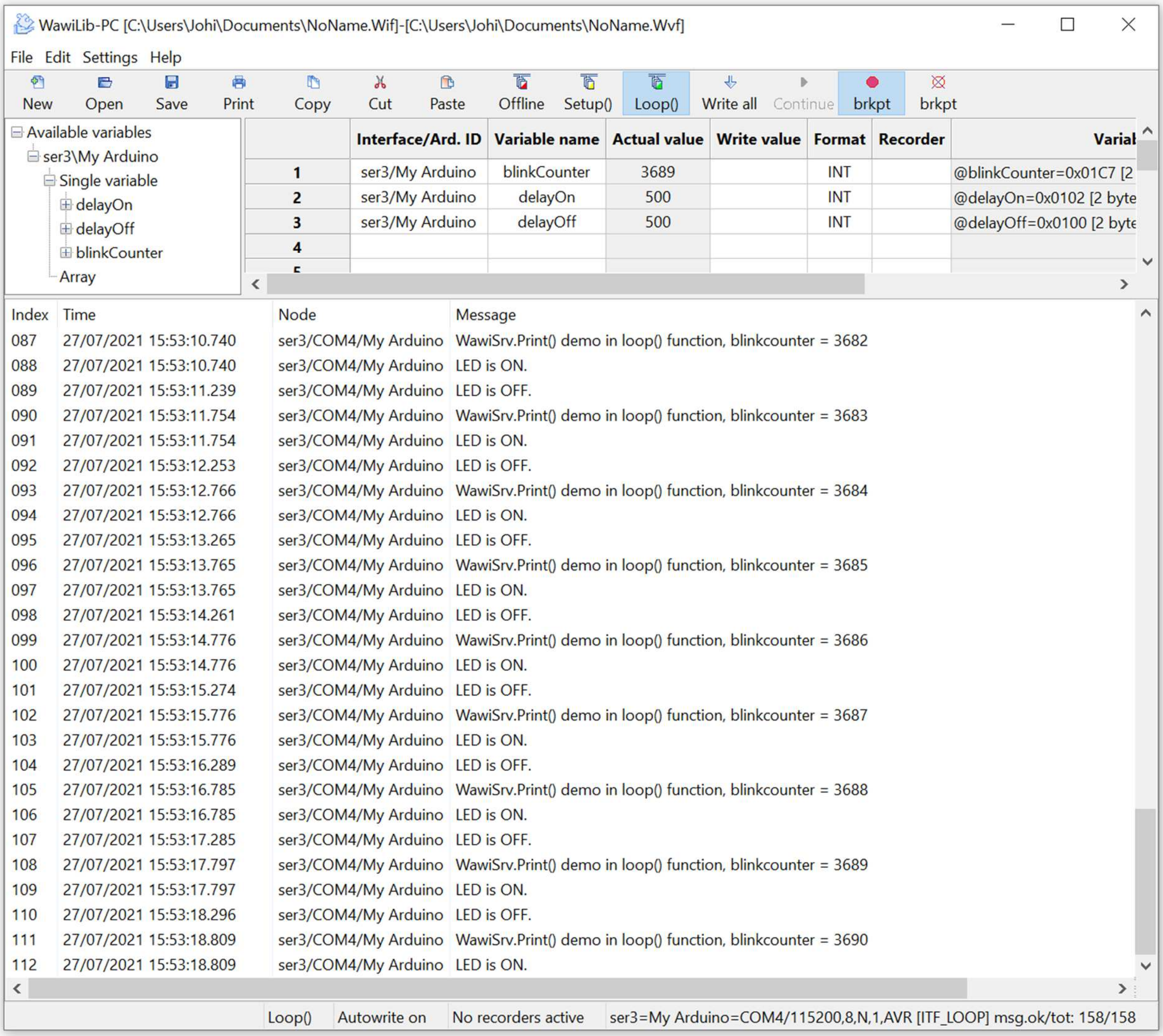


Fig 2.6. Enable different kind of messages in the output window.


WawiLib-PC [C:\Users\Johi\Documents\NoName.Wif]-[C:\Users\Johi\Documents\NoName.Wvf]
File Edit Settings Help
New Open Save Print Copy Cut Paste Offline Setup() Loop() Write all Continue brkpt brkpt
Available variables
ser3\My Arduino
Single variable
delayOn
delayOff
blinkCounter
Array

	Interface/Ard. ID	Variable name	Actual value	Write value	Format	Recorder	Varial
1	ser3/My Arduino	blinkCounter	3689		INT		@blinkCounter=0x01C7 [2
2	ser3/My Arduino	delayOn	500		INT		@delayOn=0x0102 [2 byte
3	ser3/My Arduino	delayOff	500		INT		@delayOff=0x0100 [2 byte
4							

```
Index Time                     Node                 Message
087   27/07/2021 15:53:10.740  ser3/COM4/My Arduino WawiSrv.Print() demo in loop() function, blinkcounter = 3682
088   27/07/2021 15:53:10.740  ser3/COM4/My Arduino LED is ON.
089   27/07/2021 15:53:11.239  ser3/COM4/My Arduino LED is OFF.
090   27/07/2021 15:53:11.754  ser3/COM4/My Arduino WawiSrv.Print() demo in loop() function, blinkcounter = 3683
091   27/07/2021 15:53:11.754  ser3/COM4/My Arduino LED is ON.
092   27/07/2021 15:53:12.253  ser3/COM4/My Arduino LED is OFF.
093   27/07/2021 15:53:12.766  ser3/COM4/My Arduino WawiSrv.Print() demo in loop() function, blinkcounter = 3684
094   27/07/2021 15:53:12.766  ser3/COM4/My Arduino LED is ON.
095   27/07/2021 15:53:13.265  ser3/COM4/My Arduino LED is OFF.
096   27/07/2021 15:53:13.765  ser3/COM4/My Arduino WawiSrv.Print() demo in loop() function, blinkcounter = 3685
097   27/07/2021 15:53:13.765  ser3/COM4/My Arduino LED is ON.
098   27/07/2021 15:53:14.261  ser3/COM4/My Arduino LED is OFF.
099   27/07/2021 15:53:14.776  ser3/COM4/My Arduino WawiSrv.Print() demo in loop() function, blinkcounter = 3686
100   27/07/2021 15:53:14.776  ser3/COM4/My Arduino LED is ON.
101   27/07/2021 15:53:15.274  ser3/COM4/My Arduino LED is OFF.
102   27/07/2021 15:53:15.776  ser3/COM4/My Arduino WawiSrv.Print() demo in loop() function, blinkcounter = 3687
103   27/07/2021 15:53:15.776  ser3/COM4/My Arduino LED is ON.
104   27/07/2021 15:53:16.289  ser3/COM4/My Arduino LED is OFF.
105   27/07/2021 15:53:16.785  ser3/COM4/My Arduino WawiSrv.Print() demo in loop() function, blinkcounter = 3688
106   27/07/2021 15:53:16.785  ser3/COM4/My Arduino LED is ON.
107   27/07/2021 15:53:17.285  ser3/COM4/My Arduino LED is OFF.
108   27/07/2021 15:53:17.797  ser3/COM4/My Arduino WawiSrv.Print() demo in loop() function, blinkcounter = 3689
109   27/07/2021 15:53:17.797  ser3/COM4/My Arduino LED is ON.
110   27/07/2021 15:53:18.296  ser3/COM4/My Arduino LED is OFF.
111   27/07/2021 15:53:18.809  ser3/COM4/My Arduino WawiSrv.Print() demo in loop() function, blinkcounter = 3690
112   27/07/2021 15:53:18.809  ser3/COM4/My Arduino LED is ON.
```

```
Loop()  Autowrite on  No recorders active  ser3=My Arduino=COM4/115200,8,N,1,AVR [ITF_LOOP] msg.ok/tot: 158/158
```

Fig 2.7. WawiLib grid with blinkCounter incrementing and output of .print() in.

⇨ You see the actual values of blinkCounter, delayOn and delayOff combined with the Sketch output in the WawiLib-PC output window.

2.2.3 RS232&RS485 V1r1 shield on MEGA; Hardware serial demo serial 2

2.2.3.1 Required hardware

- Arduino MEGA.
- RS232 & RS485 V1r1 Shield.
- RS232 Serial cable straight male-female (pin 1 to 1, pin 2 to 2 etc.).
- USB A to USB B cable to program the Arduino.

2.2.3.2 Hardware connections

- ✓ Bend the pins 0,1 of the shield a bit outwards so they do not connect to the headers when you mount the shield in the Arduino board (see picture below).
- ✓ Mount the shield on your Arduino Mega as in the picture below.
- ✓ Connect pin 0 of the shield to pin 17 of the Arduino using a Dupont male-female wire.
- ✓ Connect pin 1 of the shield to pin 16 of the Arduino using a Dupont male-female wire.
- ✓ Connect the V1r1 RS232 9 pin DSUB connector to a serial port on your PC.
- ✓ Connect the Arduino board to the PC using a USB A to B cable.
- ✓ Select the mode RS232 on the V1r1 shield.
- ✓ Make sure the switch UART-SoftSerial on the V1r1 shield is in the position “UART”.

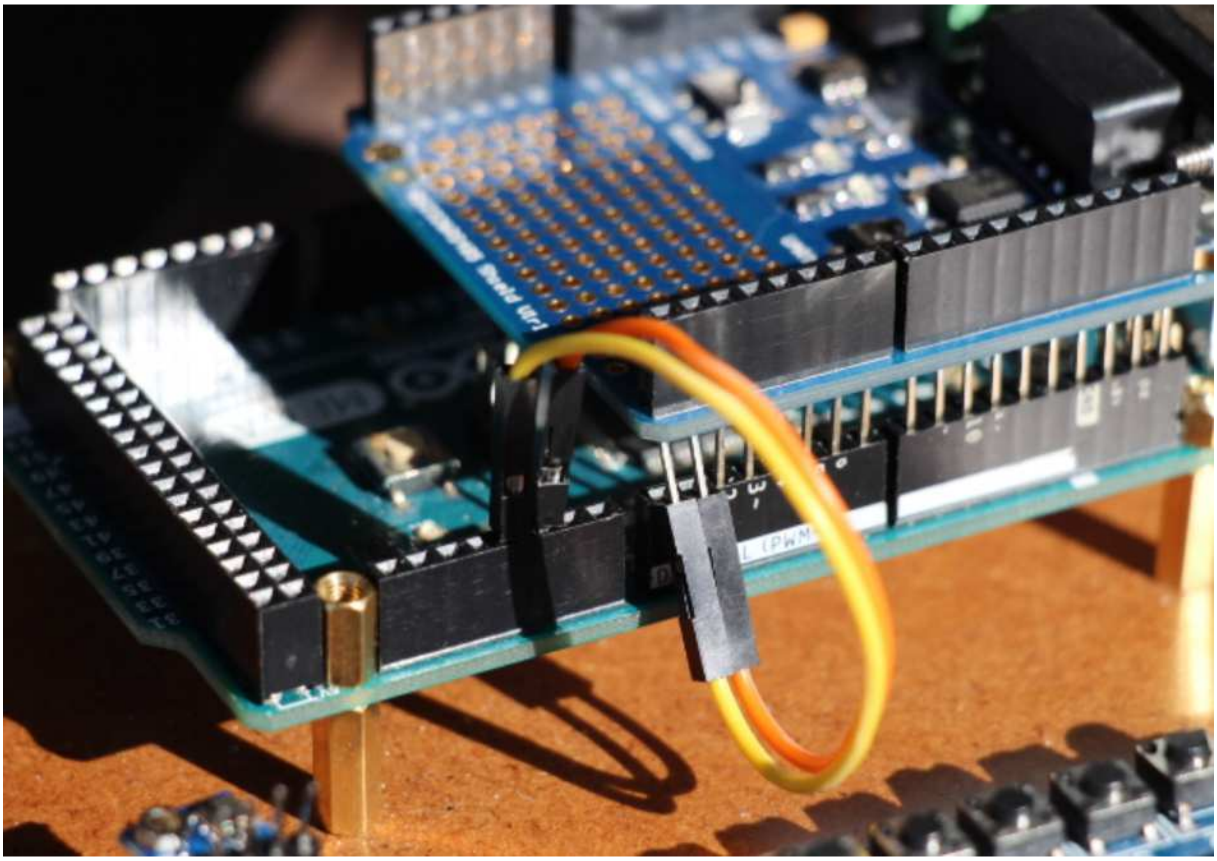


Fig 2.8. Connect V1R1 shield to the Mega2560 serial 2 port.

2.2.3.3 Load sketch

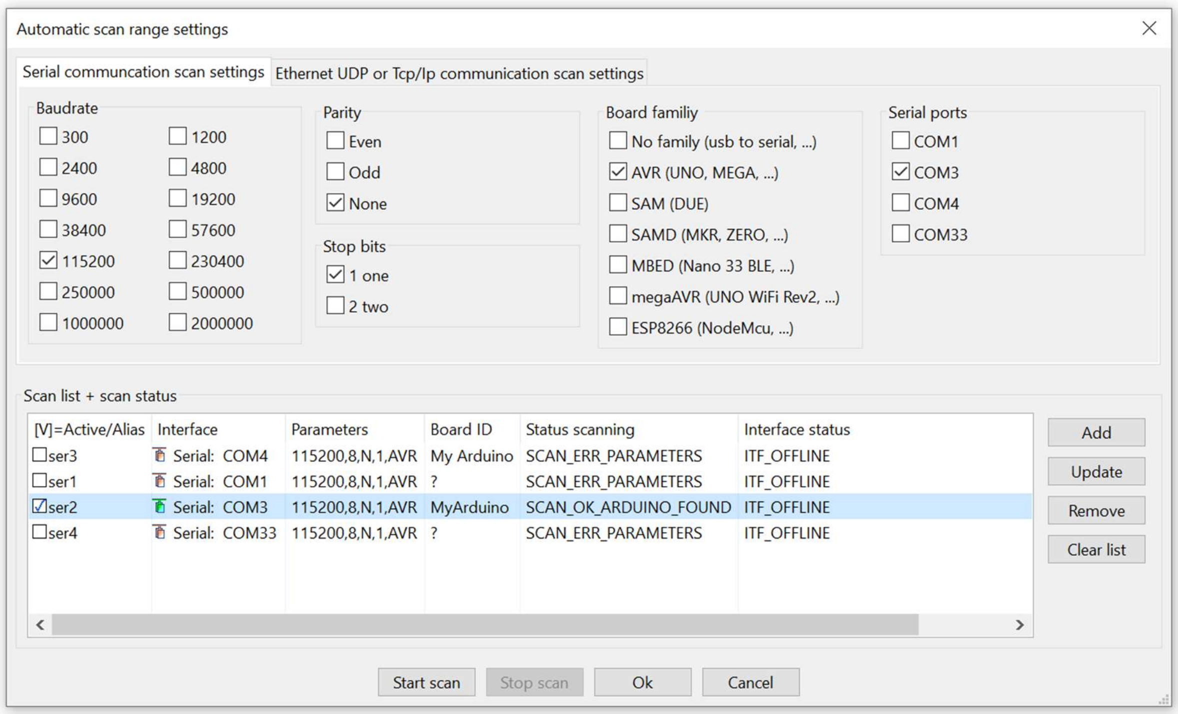
- ✓ Open the example sketch via the menu “File\Examples\WawiSerialUsb\WawiBlinkSerialSer2” in the Arduino IDE.
- ✓ Compile and download the sample.

```
/*
* Project Name: WawiBlinkSerialSer2
* File: WawiBlinkSerialSer2.ino
*/

#include <WawiSerialUsb.h>

WawiSerialUsb WawiSrv;
#define LED 13

// test variables for demo:
int delayOn = 500;
int delayOff = 500;
int blinkCounter = 0;

// make variables of interest known to WawiLib:
// this function is used in WawiSrv.begin(....)
void wawiVarDef()
{
    WawiSrv.wawiVar(delayOn);
    WawiSrv.wawiVar(delayOff);
    WawiSrv.wawiVar(blinkCounter);
}

void setup()
{
    Serial2.begin(115200);
    WawiSrv.begin(wawiVarDef, Serial2, "MyArduino");
    pinMode(LED, OUTPUT);
}

void loop()
{
    blinkCounter++;
    WawiSrv.print("WawiSrv.Print() demo in loop() function, blinkcounter = ");
    WawiSrv.println(blinkCounter);

    WawiSrv.println("LED is ON.");
    digitalWrite(LED, HIGH);
    WawiSrv.delay(delayOn);

    WawiSrv.println("LED is OFF.");
    digitalWrite(LED, LOW);
    WawiSrv.delay(delayOff);

    WawiSrv.loop();
}
```

Fig 2.9. The `WawiBlinkSerialSer2` sketch source code.

2.2.3.4 Scan ports with WawiLib

- ✓ Open the automatic scan range settings dialog box (figure below) in WawiLib.
- ✓ Use the settings as indicated in the figure below, select all available serial ports.
- ✓ Press “Add”.
- ✓ Press “Start scan”.


Automatic scan range settings

Serial communcation scan settings
Ethernet UDP or Tcp/Ip communication scan settings

Baudrate

- □ 300
- □ 1200
- □ 2400
- □ 4800
- □ 9600
- □ 19200
- □ 38400
- □ 57600
- ☑ 115200
- □ 230400
- □ 250000
- □ 500000
- □ 1000000
- □ 2000000

Parity

- □ Even
- □ Odd
- ☑ None

Stop bits

- ☑ 1 one
- □ 2 two

Board familiy

- □ No family (usb to serial, ...)
- ☑ AVR (UNO, MEGA, ...)
- □ SAM (DUE)
- □ SAMD (MKR, ZERO, ...)
- □ MBED (Nano 33 BLE, ...)
- □ megaAVR (UNO WiFi Rev2, ...)
- □ ESP8266 (NodeMcu, ...)

Serial ports

- □ COM1
- ☑ COM3
- □ COM4
- □ COM33

Scan list + scan status

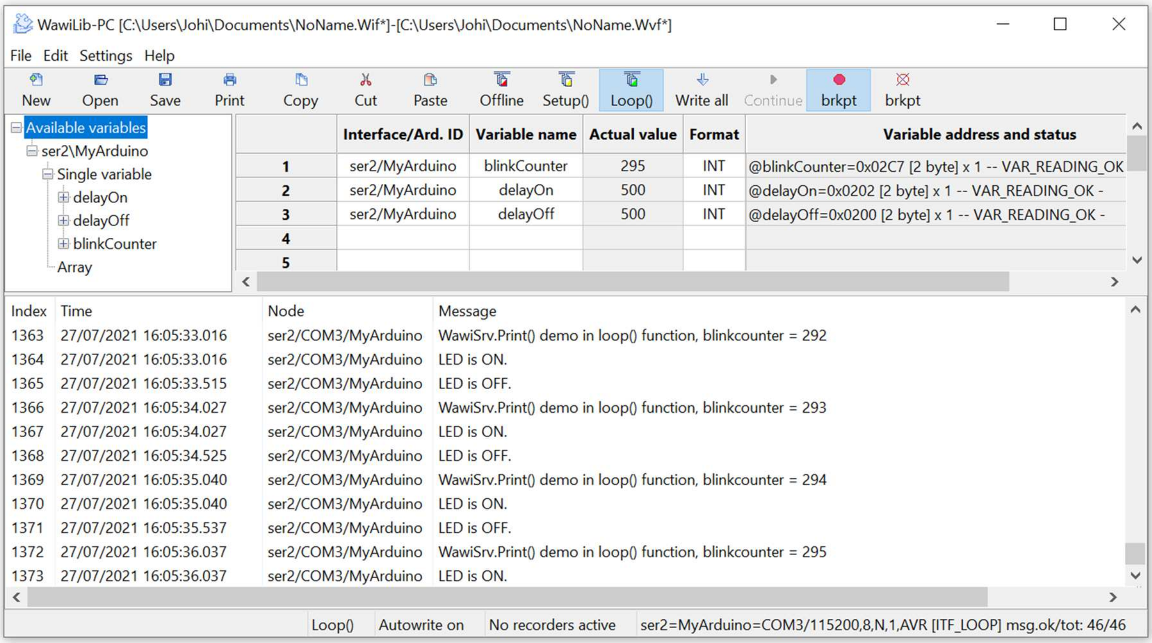
[V]=Active/Alias	Interface	Parameters	Board ID	Status scanning	Interface status
□ser3	Serial: COM4	115200,8,N,1,AVR	My Arduino	SCAN_ERR_PARAMETERS	ITF_OFFLINE
□ser1	Serial: COM1	115200,8,N,1,AVR	?	SCAN_ERR_PARAMETERS	ITF_OFFLINE
☑ser2	Serial: COM3	115200,8,N,1,AVR	MyArduino	SCAN_OK_ARDUINO_FOUND	ITF_OFFLINE
□ser4	Serial: COM33	115200,8,N,1,AVR	?	SCAN_ERR_PARAMETERS	ITF_OFFLINE

Fig 2.10. Scan range settings dialog box with multiple ports selected for scanning.

- ⇨ In the table “Scan list + scan status”, one of the icons in the “interface” column should turn green, indicating that a board with a WawiLib serial communication interface has been found.
- ✓ Click right on the table and select “remove inactive”.
- ⇨ The interfaces that were not successfully scanned will be removed from the list.
- ✓ Press OK.

2.2.3.5 Monitor variables with WawiLib

- ✓ Enable “Display Print messages” using the popup menu of the output window (right click on the output window to make the menu appear).
- ✓ Press “Setup()” in the tool bar;
- ✓ Enter the variables to the main grid as indicated in the table below.
- ✓ Alternative: Use drag & drop to drag the variables from the tree to the grid table.



WawiLib-PC [C:\Users\Johi\Documents\NoName.Wif*]-[C:\Users\Johi\Documents\NoName.Wvf*]

	Interface/Ard. ID	Variable name	Actual value	Format	Variable address and status
1	ser2/MyArduino	blinkCounter	295	INT	@blinkCounter=0x02C7 [2 byte] x 1 -- VAR_READING_OK
2	ser2/MyArduino	delayOn	500	INT	@delayOn=0x0202 [2 byte] x 1 -- VAR_READING_OK -
3	ser2/MyArduino	delayOff	500	INT	@delayOff=0x0200 [2 byte] x 1 -- VAR_READING_OK -
4					
5					

```
Index  Time                     Node                 Message
1363   27/07/2021 16:05:33.016  ser2/COM3/MyArduino  WawiSrv.Print() demo in loop() function, blinkcounter = 292
1364   27/07/2021 16:05:33.016  ser2/COM3/MyArduino  LED is ON.
1365   27/07/2021 16:05:33.515  ser2/COM3/MyArduino  LED is OFF.
1366   27/07/2021 16:05:34.027  ser2/COM3/MyArduino  WawiSrv.Print() demo in loop() function, blinkcounter = 293
1367   27/07/2021 16:05:34.027  ser2/COM3/MyArduino  LED is ON.
1368   27/07/2021 16:05:34.525  ser2/COM3/MyArduino  LED is OFF.
1369   27/07/2021 16:05:35.040  ser2/COM3/MyArduino  WawiSrv.Print() demo in loop() function, blinkcounter = 294
1370   27/07/2021 16:05:35.040  ser2/COM3/MyArduino  LED is ON.
1371   27/07/2021 16:05:35.537  ser2/COM3/MyArduino  LED is OFF.
1372   27/07/2021 16:05:36.037  ser2/COM3/MyArduino  WawiSrv.Print() demo in loop() function, blinkcounter = 295
1373   27/07/2021 16:05:36.037  ser2/COM3/MyArduino  LED is ON.
```

Loop() Autowrite on No recorders active ser2=MyArduino=COM3/115200,8,N,1,AVR [ITF_LOOP] msg.ok/tot: 46/46

Fig 2.11. Main variable grid with variables to be monitored and modified.

2.2.4 RS232&RS485 V1r1 shield on UNO; SoftwareSerial demo

2.2.4.1 Required hardware

- Arduino UNO.
- RS232 & RS485 V1r1 Shield.
- RS232 Serial cable male-female.
- USB A to USB B cable.

2.2.4.2 Hardware connections

- ✓ Mount an RS232&RS485 shield V1r1 on your UNO or Mega.
- ✓ Connect the V1r1 RS232 9 pin DSUB connector to a serial port on your PC.
- ✓ Use the USB cable to program and feed the Arduino.
- ✓ Select the mode RS232 on the V1r1 shield.
- ✓ Select the mode “SoftSerial” on the shield.

2.2.4.3 Load sketch

- ✓ Open the example “File\Examples\WawiSerialUsb\WawiBlinkSoftSerial” in the Arduino IDE.

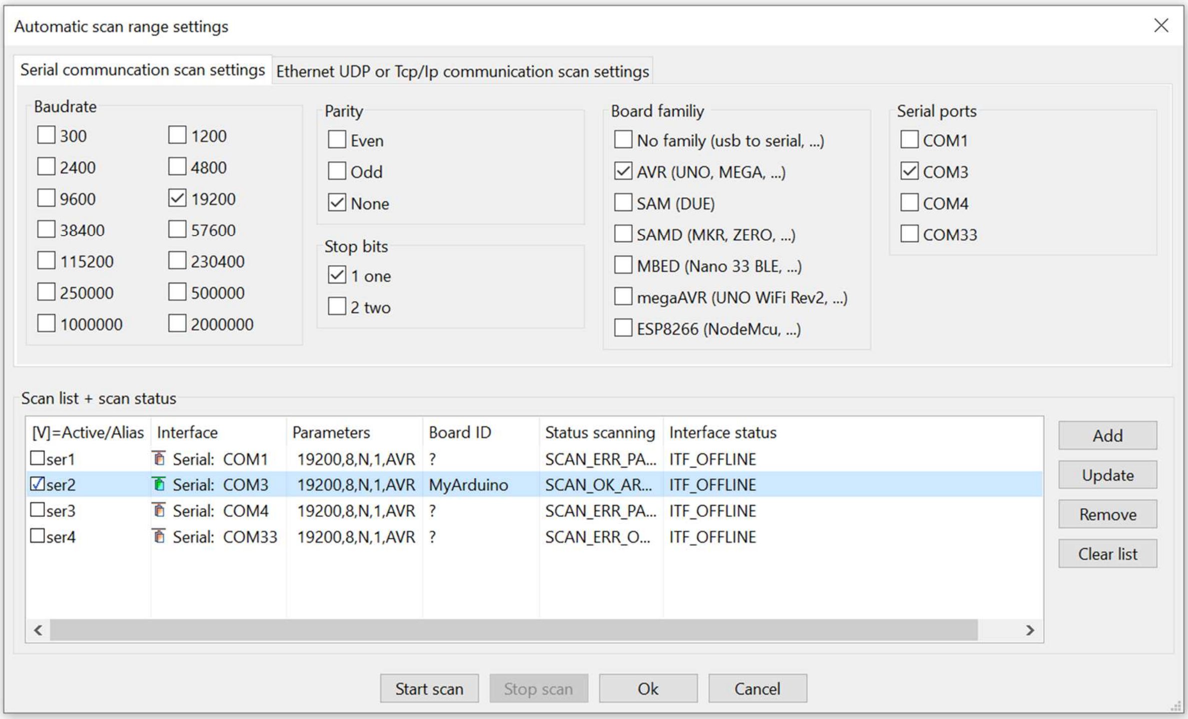
```
/*
* Project Name: WawiBlinkSoftSerial
* File: WawiBlinkSoftSerial.ino
*/

#include <WawiSerialUsb.h>
SoftwareSerial mySerial(2, 3); // RX, TX
WawiSerialUsbLight WawiSrv;

#define LED 13
int delayOn = 500;
int delayOff = 500;
int blinkCounter = 0;

void wawiVarDef()
{
    WawiSrv.wawiVar(delayOn);
    WawiSrv.wawiVar(delayOff);
    WawiSrv.wawiVar(blinkCounter);
}
void setup()
{
    mySerial.begin(19200);
    WawiSrv.begin(wawiVarDef, mySerial, "MyArduino");
    pinMode(LED, OUTPUT);
}
void loop()
{
    blinkCounter++;

    digitalWrite(LED, HIGH);
    WawiSrv.delay(delayOn);

    digitalWrite(LED, LOW);
    WawiSrv.delay(delayOff);

    WawiSrv.loop();
}
```

Fig. 2.12. The `WawiBlinkSoftSerial` sketch source code.

Note 1: We use WawiSerialUsbLight and not WawiSerialUsb. This is because SoftSerial does not support full duplex serial communication and .print() needs full duplex to work properly. WawiSerialUsbLight is smaller than WawiSerialUsb as it does not contain the source code for .print() and the full duplex protocol.
Note 2: SoftwareSerial is limited to 19200 baud hence this example does not run at 115K.

2.2.4.4 Scan ports with WawiLib

- ✓ Open the automatic scan range settings dialog box (figure below) in WawiLib.
- ✓ Use the values as indicated in the table below (select all ports).
- ✓ Press "Add".
- ✓ Press "Start scan".



Automatic scan range settings

Serial communcation scan settings | Ethernet UDP or Tcp/Ip communication scan settings

Baudrate
□ 300 □ 1200
□ 2400 □ 4800
□ 9600 ☑ 19200
□ 38400 □ 57600
□ 115200 □ 230400
□ 250000 □ 500000
□ 1000000 □ 2000000

Parity
□ Even
□ Odd
☑ None

Stop bits
☑ 1 one
□ 2 two

Board familiy
□ No family (usb to serial, ...)
☑ AVR (UNO, MEGA, ...)
□ SAM (DUE)
□ SAMD (MKR, ZERO, ...)
□ MBED (Nano 33 BLE, ...)
□ megaAVR (UNO WiFi Rev2, ...)
□ ESP8266 (NodeMcu, ...)

Serial ports
□ COM1
☑ COM3
□ COM4
□ COM33

Scan list + scan status

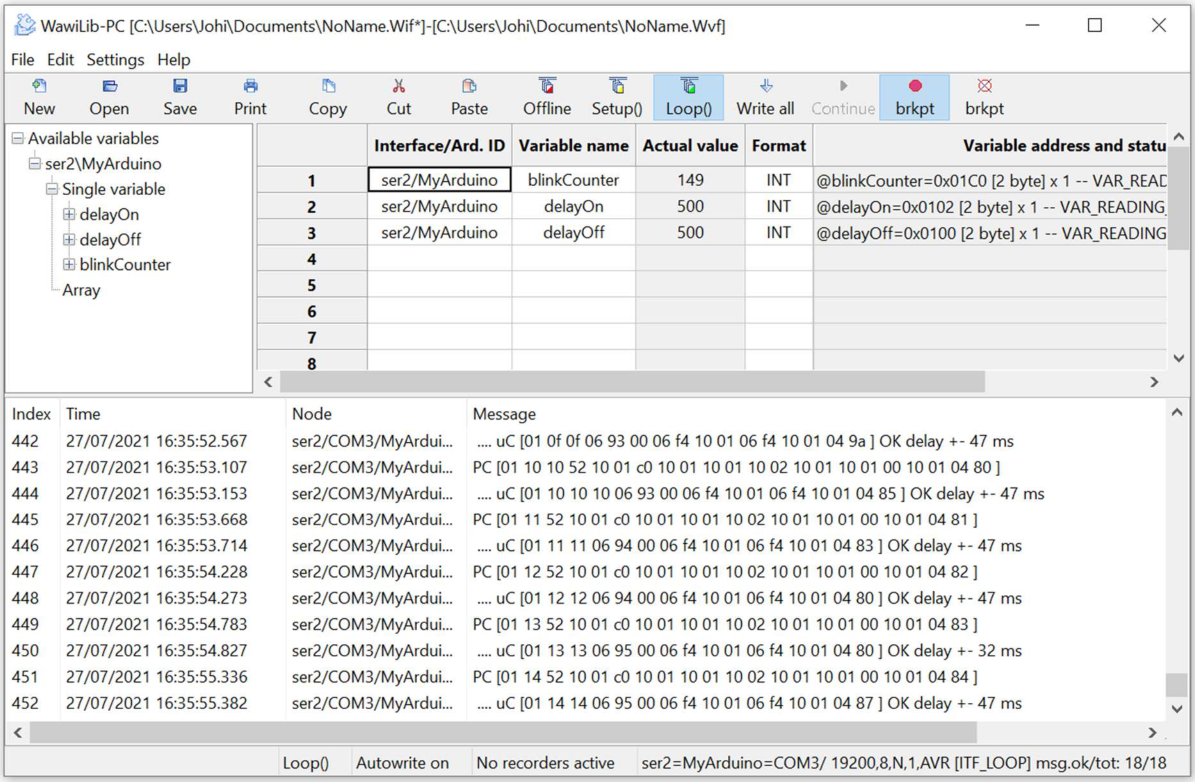
[V]=Active/Alias	Interface	Parameters	Board ID	Status scanning	Interface status
□ser1	Serial: COM1	19200,8,N,1,AVR	?	SCAN_ERR_PA...	ITF_OFFLINE
☑ser2	Serial: COM3	19200,8,N,1,AVR	MyArduino	SCAN_OK_AR...	ITF_OFFLINE
□ser3	Serial: COM4	19200,8,N,1,AVR	?	SCAN_ERR_PA...	ITF_OFFLINE
□ser4	Serial: COM33	19200,8,N,1,AVR	?	SCAN_ERR_O...	ITF_OFFLINE

Fig 2.13. Scan range settings dialog box with multiple ports selected for scanning.

- ⇨ In the table "Scan list + scan status", one of the icons in the "interface" column should turn green, indicating that a board with a WawiLib serial communication interface has been found.
- ✓ Click right on the table and select "remove inactive".
- ⇨ The interfaces that were not successfully scanned will be removed from the list.
- ✓ Press OK.
- ✓ Enable "Display Print messages" using the popup menu of the output window (right click on the output window to make the menu appear).
- ✓ Press OK.

2.2.4.5 Monitor variables with WawiLib

- ✓ Press "Setup()" in the tool bar;
- ✓ Enter the variables to the main grid as indicated in the table below.
- ✓ Alternative: Use drag & drop to drag the variables from the tree to the grid table.


WawiLib-PC [C:\Users\Johi\Documents\NoName.Wif*]-[C:\Users\Johi\Documents\NoName.Wvf]
File Edit Settings Help
New Open Save Print Copy Cut Paste Offline Setup() Loop() Write all Continue brkpt brkpt
Available variables
ser2\MyArduino
Single variable
delayOn
delayOff
blinkCounter
Array

	Interface/Ard. ID	Variable name	Actual value	Format	Variable address and statu
1	ser2/MyArduino	blinkCounter	149	INT	@blinkCounter=0x01C0 [2 byte] x 1 -- VAR_READ
2	ser2/MyArduino	delayOn	500	INT	@delayOn=0x0102 [2 byte] x 1 -- VAR_READING
3	ser2/MyArduino	delayOff	500	INT	@delayOff=0x0100 [2 byte] x 1 -- VAR_READING
4					
5					
6					
7					
8					

```
Index Time                     Node                 Message
442   27/07/2021 16:35:52.567  ser2/COM3/MyArdui...  .... uC [01 0f 0f 06 93 00 06 f4 10 01 06 f4 10 01 04 9a ] OK delay +- 47 ms
443   27/07/2021 16:35:53.107  ser2/COM3/MyArdui...  PC [01 10 10 52 10 01 c0 10 01 10 01 10 02 10 01 10 01 00 10 01 04 80 ]
444   27/07/2021 16:35:53.153  ser2/COM3/MyArdui...  .... uC [01 10 10 10 06 93 00 06 f4 10 01 06 f4 10 01 04 85 ] OK delay +- 47 ms
445   27/07/2021 16:35:53.668  ser2/COM3/MyArdui...  PC [01 11 52 10 01 c0 10 01 10 01 10 02 10 01 10 01 00 10 01 04 81 ]
446   27/07/2021 16:35:53.714  ser2/COM3/MyArdui...  .... uC [01 11 11 06 94 00 06 f4 10 01 06 f4 10 01 04 83 ] OK delay +- 47 ms
447   27/07/2021 16:35:54.228  ser2/COM3/MyArdui...  PC [01 12 52 10 01 c0 10 01 10 01 10 02 10 01 10 01 00 10 01 04 82 ]
448   27/07/2021 16:35:54.273  ser2/COM3/MyArdui...  .... uC [01 12 12 06 94 00 06 f4 10 01 06 f4 10 01 04 80 ] OK delay +- 47 ms
449   27/07/2021 16:35:54.783  ser2/COM3/MyArdui...  PC [01 13 52 10 01 c0 10 01 10 01 10 02 10 01 10 01 00 10 01 04 83 ]
450   27/07/2021 16:35:54.827  ser2/COM3/MyArdui...  .... uC [01 13 13 06 95 00 06 f4 10 01 06 f4 10 01 04 80 ] OK delay +- 32 ms
451   27/07/2021 16:35:55.336  ser2/COM3/MyArdui...  PC [01 14 52 10 01 c0 10 01 10 01 10 02 10 01 10 01 00 10 01 04 84 ]
452   27/07/2021 16:35:55.382  ser2/COM3/MyArdui...  .... uC [01 14 14 06 95 00 06 f4 10 01 06 f4 10 01 04 87 ] OK delay +- 47 ms
```

```
Loop()  Autowrite on  No recorders active  ser2=MyArduino=COM3/ 19200,8,N,1,AVR [ITF_LOOP] msg.ok/tot: 18/18
```

Fig 2.14. Main variable grid with variables to be monitored and modified.

- ⇨ You will see the value of blinkCounter increase and the msg.ok and msg.tot counters increment.
- ⇨ If you click right on the output window and activate “Display communication protocol messages”, the Output window will detail the message exchange between the PC and the board.

2.2.5 RS232 & RS485 V1r1 shield on MEGA2560; SoftwareSerial demo.

2.2.5.1 Required hardware

- Arduino MEGA
- RS232 & RS485 V1r1 Shield
- RS232 Serial cable male-female
- USB A to B cable.
- 2 Dupont male to female wires

2.2.5.2 Hardware connections

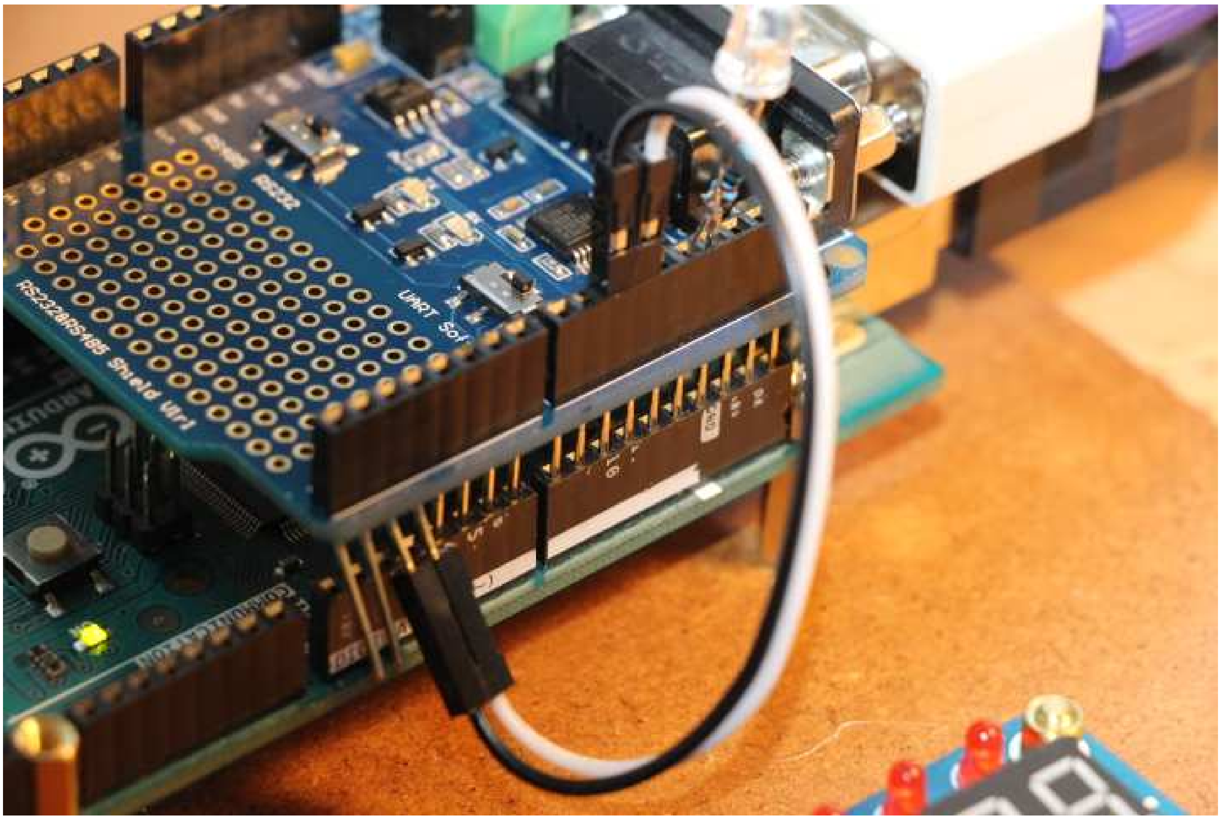


Fig 2.15. Connect V1R1 shield to the Mega2560 pins 10 and 11.

- ✓ Bend the pins 0,1,2,3 of the shield a bit outwards so they do not connect to the headers when you mount the shield in the Arduino board (see picture above).
- ✓ Mount the shield on your Arduino Mega as in the picture above.
- ✓ Connect pin 2 of the shield to pin 10 of the Arduino and connect pin 3 to pin 11 using 2 Dupont male-female breadboard wires.
- ✓ Connect the V1r1 RS232 9 pin DSUB connector to a serial port on your PC using a straight cable.
- ✓ Select the modes RS232 and SoftSerial on the shield.

Noted: the shield does not work with softserial by standard design as the pins 2 and 3, that are used if you set the shield in “softserial mode”, have no interrupt capability on the MEGA2560.

2.2.5.3 Load sketch

- ✓ Open the example sketch via the menu “File\Examples\WawiSerialUsb\WawiBlinkMega2560SoftSerial” in the Arduino IDE.
- ✓ Compile and download the sample (see figure below).

```
/*
* Project Name: WawiBlinkMega2560SoftSerial
* File: WawiBlinkMega2560SoftSerial.ino
*/

#include <WawiSerialUsb.h>
SoftwareSerial mySerial(10, 11); // RX, TX
WawiSerialUsbLight WawiSrv;

#define LED 13
int delayOn = 500;
int delayOff = 500;
int blinkCounter = 0;

void wawiVarDef()
{
    WawiSrv.wawiVar(delayOn);
    WawiSrv.wawiVar(delayOff);
    WawiSrv.wawiVar(blinkCounter);
}

void setup()
{
    mySerial.begin(19200);
    WawiSrv.begin(wawiVarDef, mySerial, "MyArduino");
    pinMode(LED, OUTPUT);
}

void loop()
{
    blinkCounter++;

    digitalWrite(LED, HIGH);
    WawiSrv.delay(delayOn);

    digitalWrite(LED, LOW);
    WawiSrv.delay(delayOff);

    WawiSrv.loop();
}
```

Fig. 2.16. The `WawiBlinkMega2560SoftSerial` sketch source code.

Note that we use WawiSerialUsbLight and not WawiSerialUsb. This is because SoftSerial does not support full duplex serial communication and .print() needs full duplex to work properly. WawiSerialUsbLight is smaller than WawiSerialUsb as it does not contain the source code for .print() and the full duplex protocol.

2.2.5.4 Scan ports with WawiLib

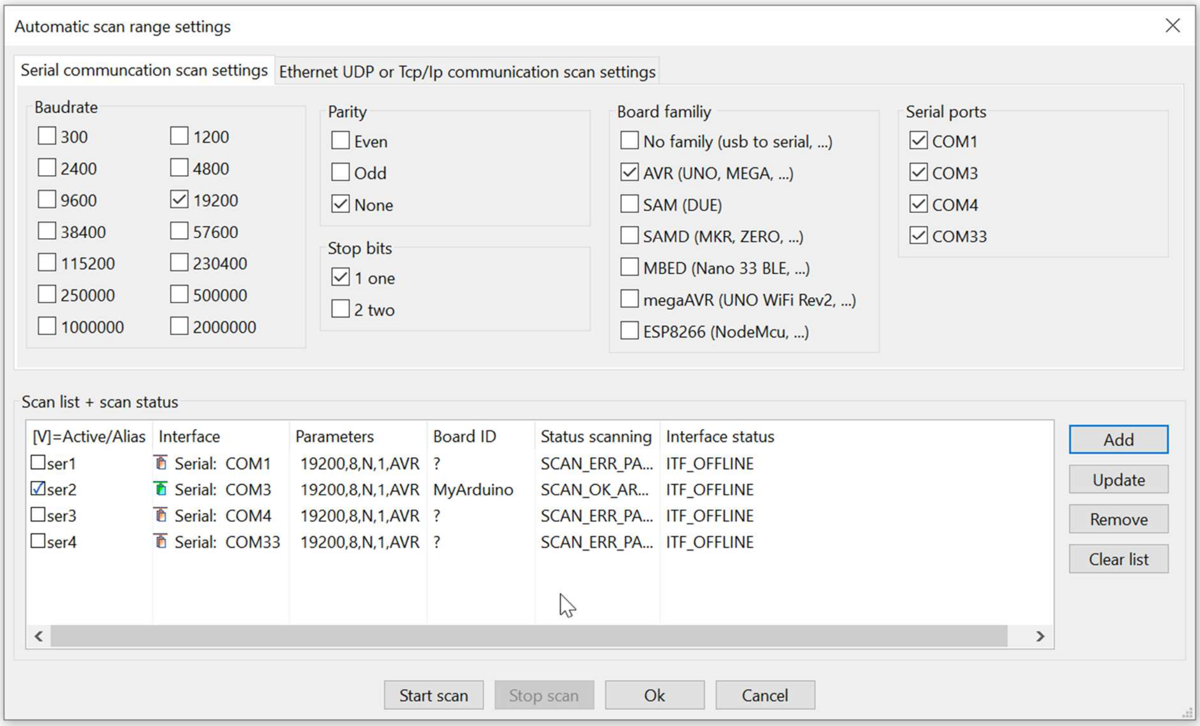
✓ Open the automatic scan range settings dialog box (figure below) in WawiLib

✓ Use the values as indicated in the figure below (select all serial ports)

✓ Press "Add"

✓ Press "Start scan"

⇨ Beware: SoftwareSerial is limited to 19200 baud hence this example does not run at 115K.



Automatic scan range settings

Serial communcation scan settings | Ethernet UDP or Tcp/Ip communication scan settings

Baudrate

□ 300 □ 1200
□ 2400 □ 4800
□ 9600 ☑ 19200
□ 38400 □ 57600
□ 115200 □ 230400
□ 250000 □ 500000
□ 1000000 □ 2000000

Parity

□ Even
□ Odd
☑ None

Stop bits

☑ 1 one
□ 2 two

Board familiy

□ No family (usb to serial, ...)
☑ AVR (UNO, MEGA, ...)
□ SAM (DUE)
□ SAMD (MKR, ZERO, ...)
□ MBED (Nano 33 BLE, ...)
□ megaAVR (UNO WiFi Rev2, ...)
□ ESP8266 (NodeMcu, ...)

Serial ports

☑ COM1
☑ COM3
☑ COM4
☑ COM33

Scan list + scan status

[V]=Active/Alias	Interface	Parameters	Board ID	Status scanning	Interface status
□ser1	Serial: COM1	19200,8,N,1,AVR	?	SCAN_ERR_PA...	ITF_OFFLINE
☑ser2	Serial: COM3	19200,8,N,1,AVR	MyArduino	SCAN_OK_AR...	ITF_OFFLINE
□ser3	Serial: COM4	19200,8,N,1,AVR	?	SCAN_ERR_PA...	ITF_OFFLINE
□ser4	Serial: COM33	19200,8,N,1,AVR	?	SCAN_ERR_PA...	ITF_OFFLINE

Fig 2.17. Scan range settings dialog box with multiple ports selected for scanning.

⇨ In the table “Scan list + scan status”, one of the icons in the “interface” column should turn green, indicating that a board with a WawiLib serial communication interface has been found.

✓ Click right on the table and select “remove inactive”

⇨ The interfaces that were not successfully scanned will be removed from the list

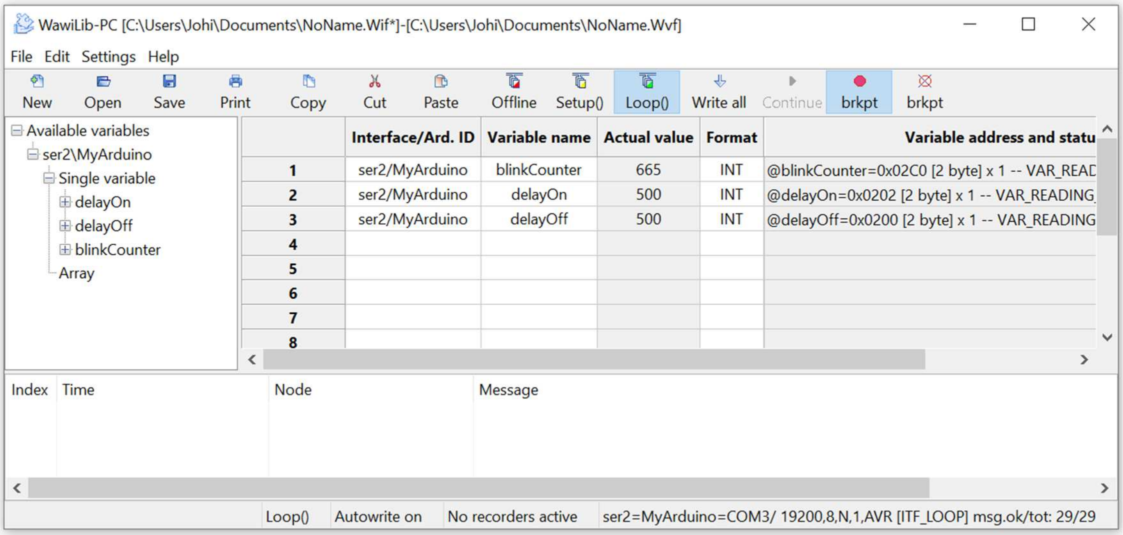
✓ Press OK

2.2.5.5 Monitor variables with WawiLib

✓ Press “Setup()” in the tool bar;

✓ Enter the variables to the main grid as indicated in the table below.

✓ Alternative: Use drag & drop to drag the variables from the tree to the grid table.



WawiLib-PC [C:\Users\Johi\Documents\NoName.Wif*]-[C:\Users\Johi\Documents\NoName.Wvf]

Available variables
- ser2\MyArduino
  - Single variable
    - delayOn
    - delayOff
    - blinkCounter
  - Array

	Interface/Ard. ID	Variable name	Actual value	Format	Variable address and statu
1	ser2/MyArduino	blinkCounter	665	INT	@blinkCounter=0x02C0 [2 byte] x 1 -- VAR_READ
2	ser2/MyArduino	delayOn	500	INT	@delayOn=0x0202 [2 byte] x 1 -- VAR_READING
3	ser2/MyArduino	delayOff	500	INT	@delayOff=0x0200 [2 byte] x 1 -- VAR_READING
4					
5					
6					
7					
8					

Index	Time	Node	Message

Loop() | Autowrite on | No recorders active | ser2=MyArduino=COM3/ 19200,8,N,1,AVR [ITF_LOOP] msg.ok/tot: 29/29

Fig 2.18. Main variable grid with variables to be monitored and modified.

⇨ You will see the value of blinkCounter and the msg.ok and msg.tot counters increase.

Note: due to SoftwareSerial not supporting full duplex communication, there is not .print() output available in the WawiLib output window.

2.3 Native USB Communication

2.3.1 Demo: Native USB port on Arduino DUE demo

2.3.1.1 Introduction

The Arduino DUE is a high-performance device when it comes to serial/USB communication. The DUE has 2 USB ports, one USB programming port and one USB native port. In a WawiLib configuration, you typically would use the programming port for programming and getting debug output to the serial monitor window of the IDE. In parallel, you use the native USB port for communication with WawiLib. This configuration can be very handy for debugging and other tasks.

Since the USB native port is an USB port that is part of the ATMEL ATSAM3X8E main processor of the due board, very high throughput is available.

2.3.1.2 Required hardware

- ○ Arduino DUE
- ○ 2 USB A to micro B cables

2.3.1.3 Hardware connections

- ✓ Connect the DUE programming port to your PC with an USB A to micro B cable
- ✓ Connect the DUE USB native port to your PC with an USB A to micro B cable

2.3.1.4 Load the sketch

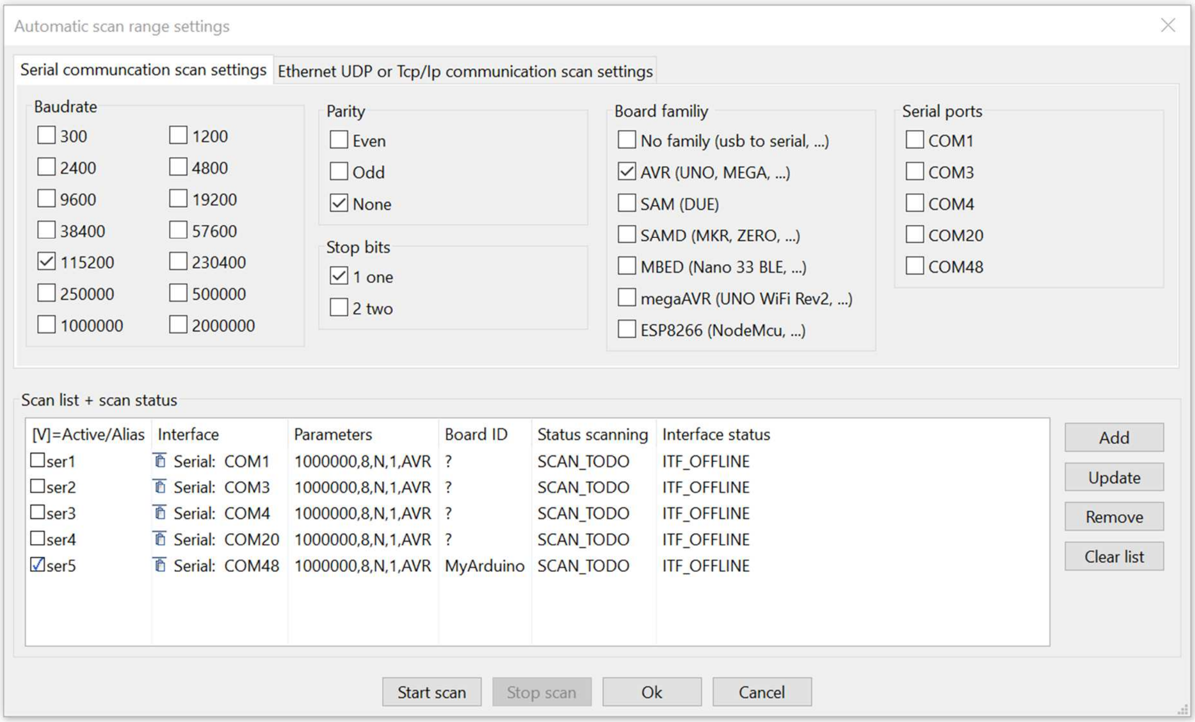
- ✓ Open the example sketch via the menu “File\Examples\WawiSerialUsb\WawiDueBlinkNativeUsb” in the Arduino IDE.

```
/*
* Project Name: WawiBlinkDueNativeUsb
* File: WawiBlinkDueNativeUsb.ino
*/
#include <WawiSerialUsb.h>
WawiSerialUsb WawiSrv;
// Arduino board has Led at I/O 13, use I/O 13 to blink
#define LED 13
// test variables for demo:
int delayOn = 500;
int delayOff = 500;
int blinkCounter = 0;

// make variables of interest known to WawiLib:
// this function is used in WawiSrv.begin(....)
void wawiVarDef()
{
    WawiSrv.wawiVar(delayOn);
    WawiSrv.wawiVar(delayOff);
    WawiSrv.wawiVar(blinkCounter);
}
void setup()
{
    // wawilib via high speed native USB port:
    SerialUSB.begin(1000000);
    // initialize WawiLib library:
    WawiSrv.begin(wawiVarDef, SerialUSB, "MyArduino");
    pinMode(LED, OUTPUT);
}
void loop()
{
    blinkCounter++;
    WawiSrv.print("WawiSrv.Print() demo in loop() function, blinkcounter = ");
    WawiSrv.println(blinkCounter);

    WawiSrv.println("LED is ON.");
    digitalWrite(LED, HIGH);
    WawiSrv.delay(delayOn);

    WawiSrv.println("LED is OFF.");
    digitalWrite(LED, LOW);
    WawiSrv.delay(delayOff);

WawiSrv.loop();
}
```

Fig. 2.19. The `WawiBlinkDueNativeUsb` sketch source code.

2.3.1.5 Scan ports with WawiLib

- ✓ Open the automatic scan range settings dialog box (figure below) in WawiLib
- ✓ Fill in the settings as indicated in the table below (select all serial ports)
- ✓ Press “Add”
- ✓ Press “Start scan”



Automatic scan range settings

Serial communcation scan settings | Ethernet UDP or Tcp/Ip communication scan settings

Baudrate
□ 300 □ 1200
□ 2400 □ 4800
□ 9600 □ 19200
□ 38400 □ 57600
☑ 115200 □ 230400
□ 250000 □ 500000
□ 1000000 □ 2000000

Parity
□ Even
□ Odd
☑ None

Stop bits
☑ 1 one
□ 2 two

Board familiy
□ No family (usb to serial, ...)
☑ AVR (UNO, MEGA, ...)
□ SAM (DUE)
□ SAMD (MKR, ZERO, ...)
□ MBED (Nano 33 BLE, ...)
□ megaAVR (UNO WiFi Rev2, ...)
□ ESP8266 (NodeMcu, ...)

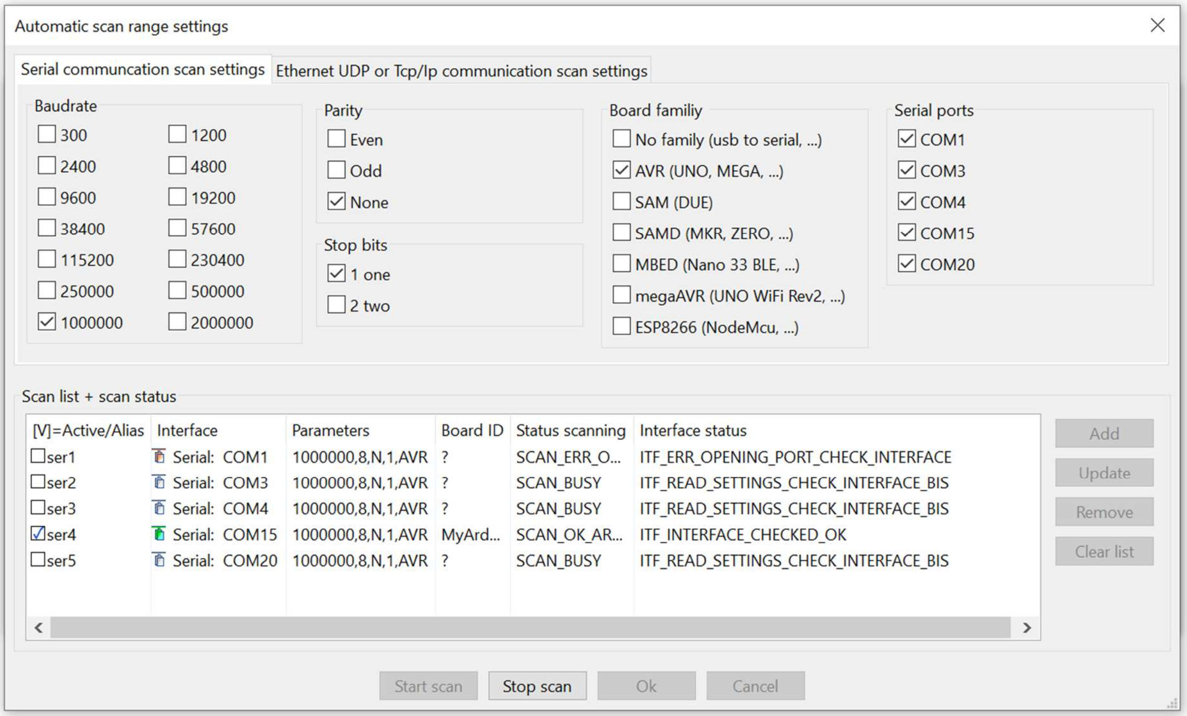
Serial ports
□ COM1
□ COM3
□ COM4
□ COM20
□ COM48

Scan list + scan status

[V]=Active/Alias	Interface	Parameters	Board ID	Status scanning	Interface status
□ser1	Serial: COM1	1000000,8,N,1,AVR	?	SCAN_TODO	ITF_OFFLINE
□ser2	Serial: COM3	1000000,8,N,1,AVR	?	SCAN_TODO	ITF_OFFLINE
□ser3	Serial: COM4	1000000,8,N,1,AVR	?	SCAN_TODO	ITF_OFFLINE
□ser4	Serial: COM20	1000000,8,N,1,AVR	?	SCAN_TODO	ITF_OFFLINE
☑ser5	Serial: COM48	1000000,8,N,1,AVR	MyArduino	SCAN_TODO	ITF_OFFLINE

Fig 2.20. Scan range settings dialog box with multiple ports selected for scanning.

⇨ In the table “Scan list + scan status”, one of the icons in the “interface” column should turn green, indicating that a board with a WawiLib serial communication interface has been found.



Automatic scan range settings

Serial communcation scan settings | Ethernet UDP or Tcp/Ip communication scan settings

Baudrate
□ 300 □ 1200
□ 2400 □ 4800
□ 9600 □ 19200
□ 38400 □ 57600
□ 115200 □ 230400
□ 250000 □ 500000
☑ 1000000 □ 2000000

Parity
□ Even
□ Odd
☑ None

Stop bits
☑ 1 one
□ 2 two

Board familiy
□ No family (usb to serial, ...)
☑ AVR (UNO, MEGA, ...)
□ SAM (DUE)
□ SAMD (MKR, ZERO, ...)
□ MBED (Nano 33 BLE, ...)
□ megaAVR (UNO WiFi Rev2, ...)
□ ESP8266 (NodeMcu, ...)

Serial ports
☑ COM1
☑ COM3
☑ COM4
☑ COM15
☑ COM20

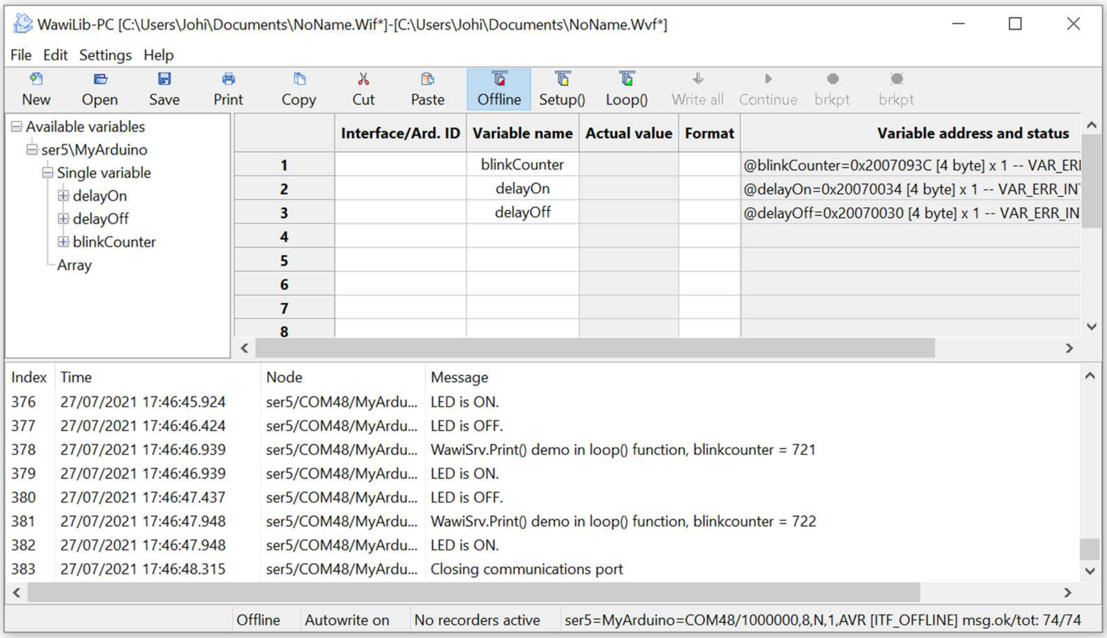
Scan list + scan status

[V]=Active/Alias	Interface	Parameters	Board ID	Status scanning	Interface status
□ser1	Serial: COM1	1000000,8,N,1,AVR	?	SCAN_ERR_O...	ITF_ERR_OPENING_PORT_CHECK_INTERFACE
□ser2	Serial: COM3	1000000,8,N,1,AVR	?	SCAN_BUSY	ITF_READ_SETTINGS_CHECK_INTERFACE_BIS
□ser3	Serial: COM4	1000000,8,N,1,AVR	?	SCAN_BUSY	ITF_READ_SETTINGS_CHECK_INTERFACE_BIS
☑ser4	Serial: COM15	1000000,8,N,1,AVR	MyArd...	SCAN_OK_AR...	ITF_INTERFACE_CHECKED_OK
□ser5	Serial: COM20	1000000,8,N,1,AVR	?	SCAN_BUSY	ITF_READ_SETTINGS_CHECK_INTERFACE_BIS

Fig 2.21. Scan range settings dialog box after scanning.

- ✓ Click right on the table and select "remove inactive"
- ⇨ The interfaces that were not successfully scanned will be removed from the list
- ✓ Press OK

- ✓ Press "Setup()" in the tool bar;
- ✓ Enter the variables to the main grid as indicated in the table below.
- ✓ Alternative: Use drag & drop to drag the variables from the tree to the grid table.



WawiLib-PC [C:\Users\Johi\Documents\NoName.Wif*]-[C:\Users\Johi\Documents\NoName.Wvf*]

File Edit Settings Help
New Open Save Print Copy Cut Paste Offline Setup() Loop() Write all Continue brkpt brkpt

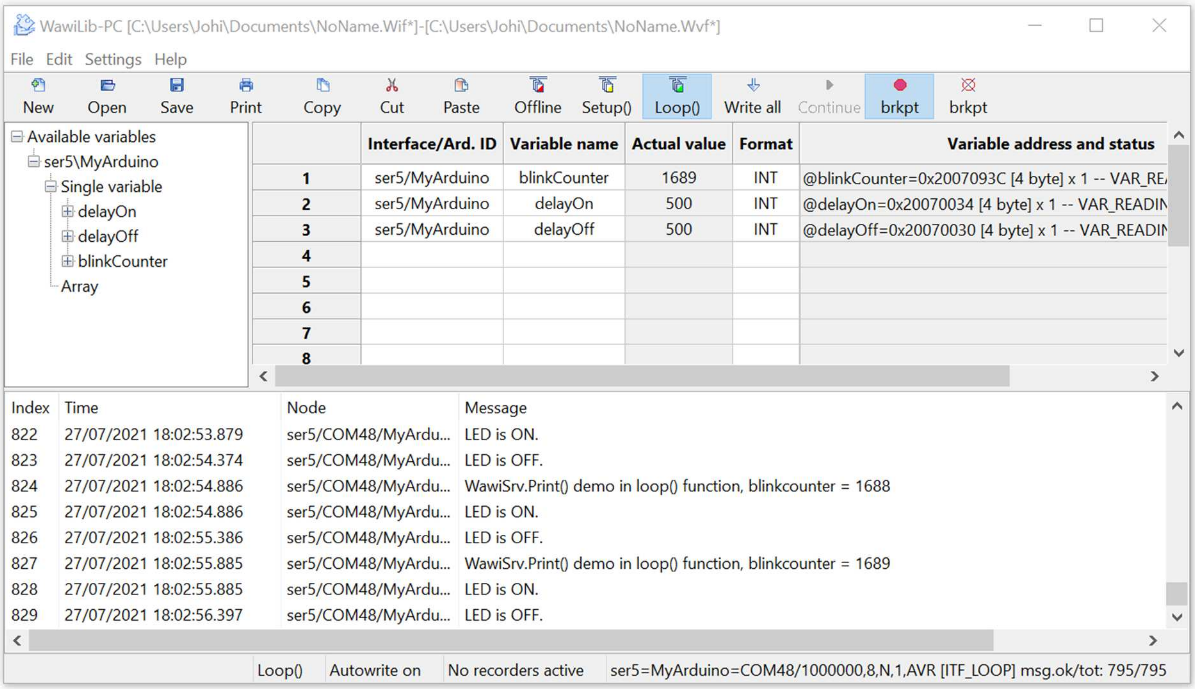
Available variables
- ser5\MyArduino
  - Single variable
    - delayOn
    - delayOff
    - blinkCounter
  - Array

	Interface/Ard. ID	Variable name	Actual value	Format	Variable address and status
1		blinkCounter			@blinkCounter=0x2007093C [4 byte] x 1 -- VAR_ERI
2		delayOn			@delayOn=0x20070034 [4 byte] x 1 -- VAR_ERR_IN
3		delayOff			@delayOff=0x20070030 [4 byte] x 1 -- VAR_ERR_IN
4					
5					
6					
7					
8					

```
Index  Time                     Node                  Message
376    27/07/2021 17:46:45.924  ser5/COM48/MyArdu...  LED is ON.
377    27/07/2021 17:46:46.424  ser5/COM48/MyArdu...  LED is OFF.
378    27/07/2021 17:46:46.939  ser5/COM48/MyArdu...  WawiSrv.Print() demo in loop() function, blinkcounter = 721
379    27/07/2021 17:46:46.939  ser5/COM48/MyArdu...  LED is ON.
380    27/07/2021 17:46:47.437  ser5/COM48/MyArdu...  LED is OFF.
381    27/07/2021 17:46:47.948  ser5/COM48/MyArdu...  WawiSrv.Print() demo in loop() function, blinkcounter = 722
382    27/07/2021 17:46:47.948  ser5/COM48/MyArdu...  LED is ON.
383    27/07/2021 17:46:48.315  ser5/COM48/MyArdu...  Closing communications port
```

Offline | Autowrite on | No recorders active | ser5=MyArduino=COM48/1000000,8,N,1,AVR [ITF_OFFLINE] msg.ok/tot: 74/74

Fig 2.22. Main variable grid with variables to be monitored and modified.



	Interface/Ard. ID	Variable name	Actual value	Format	Variable address and status
1	ser5/MyArduino	blinkCounter	1689	INT	@blinkCounter=0x2007093C [4 byte] x 1 -- VAR_RE
2	ser5/MyArduino	delayOn	500	INT	@delayOn=0x20070034 [4 byte] x 1 -- VAR_READIN
3	ser5/MyArduino	delayOff	500	INT	@delayOff=0x20070030 [4 byte] x 1 -- VAR_READIN
4					
5					
6					
7					
8					

```
Index  Time                     Node                 Message
822    27/07/2021 18:02:53.879  ser5/COM48/MyArdu... LED is ON.
823    27/07/2021 18:02:54.374  ser5/COM48/MyArdu... LED is OFF.
824    27/07/2021 18:02:54.886  ser5/COM48/MyArdu... WawiSrv.Print() demo in loop() function, blinkcounter = 1688
825    27/07/2021 18:02:54.886  ser5/COM48/MyArdu... LED is ON.
826    27/07/2021 18:02:55.386  ser5/COM48/MyArdu... LED is OFF.
827    27/07/2021 18:02:55.885  ser5/COM48/MyArdu... WawiSrv.Print() demo in loop() function, blinkcounter = 1689
828    27/07/2021 18:02:55.885  ser5/COM48/MyArdu... LED is ON.
829    27/07/2021 18:02:56.397  ser5/COM48/MyArdu... LED is OFF.
```

Loop() | Autowrite on | No recorders active | ser5=MyArduino=COM48/1000000,8,N,1,AVR [ITF_LOOP] msg.ok/tot: 795/795

Fig 2.23. Main variable grid monitoring variables.

2.4 USB to serial converters

2.4.1 Introduction

Today, there many USB to serial converters available on the market. WawiLib was tested with the following types:

Silicon Labs CP210X USB to UART Bridge	USB-SERIAL CH340	FTDI FT232RL USB TO TTL	USB-SERIAL CH340

Fig 2.24. Various types serial to USB converters.

Other converters are likely to work as well. There are 3 topics to take into account if you select a convertor.

First, beware of the voltage reference levels. Some of them have a jumper you can use to switch between the 5V and the 3.3V standard. Others have only 1 mode to operate in 3.3V or 5V.

Second, also with these converters you can get into trouble if you connect them on pins 0 and 1 in parallel with the 16U2 USB to serial converter that is installed on the Arduino board for programming. Therefore, some of them will work with pins 0 and 1 (serial 0) and others will not. For a detailed compatibility table: go to www.sylvestersolutions.com

Third, make sure that your converter is compatible with your operating system and that the proper drivers are available.

2.4.2 Loop back test of USB to serial converter

Once you have connected your converter to the PC, you can do a loop-back test to see if the device is properly installed and working correctly:

- ✓ Make a bridge between the TX pin and the RX pin of the converter
- ✓ Connect the converter to your PC

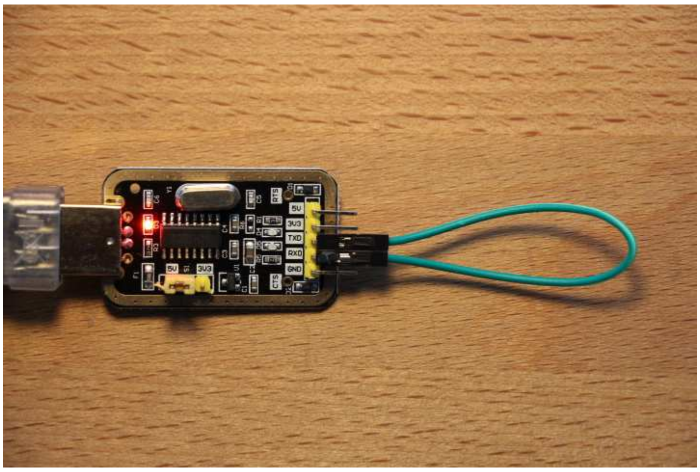
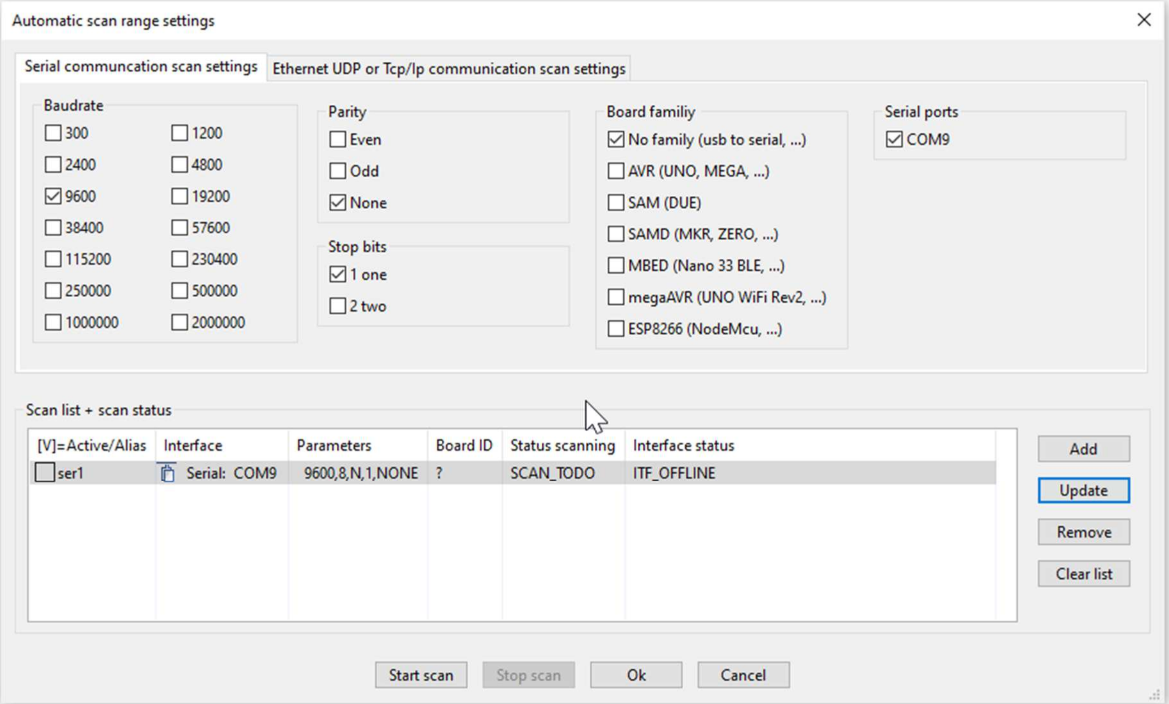


Fig 2.25. Loopback on USB to serial converter RX=TX.

- ✓ Start WawiLib
- ✓ Clear the output window
- ✓ Enable protocol tracing in the output window (right click output window for the popup menu)
- ✓ Go to "settings/communication" interfaces
- ✓ Select the parameters as indicated in the next table
- ✓ press "Add"
- ✓ press "Start scan"



Automatic scan range settings

Serial communcation scan settings | Ethernet UDP or Tcp/Ip communication scan settings

Baudrate
□ 300 □ 1200
□ 2400 □ 4800
☑ 9600 □ 19200
□ 38400 □ 57600
□ 115200 □ 230400
□ 250000 □ 500000
□ 1000000 □ 2000000

Parity
□ Even
□ Odd
☑ None

Stop bits
☑ 1 one
□ 2 two

Board familiy
☑ No family (usb to serial, ...)
□ AVR (UNO, MEGA, ...)
□ SAM (DUE)
□ SAMD (MKR, ZERO, ...)
□ MBED (Nano 33 BLE, ...)
□ megaAVR (UNO WiFi Rev2, ...)
□ ESP8266 (NodeMcu, ...)

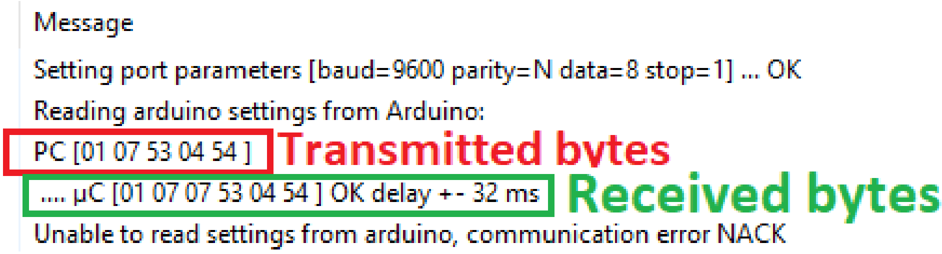
Serial ports
☑ COM9

Scan list + scan status

[V]=Active/Alias	Interface	Parameters	Board ID	Status scanning	Interface status
□ ser1	Serial: COM9	9600,8,N,1,NONE	?	SCAN_TODO	ITF_OFFLINE

Fig 2.26. Define a random Arduino board to test the loopback of the converter.

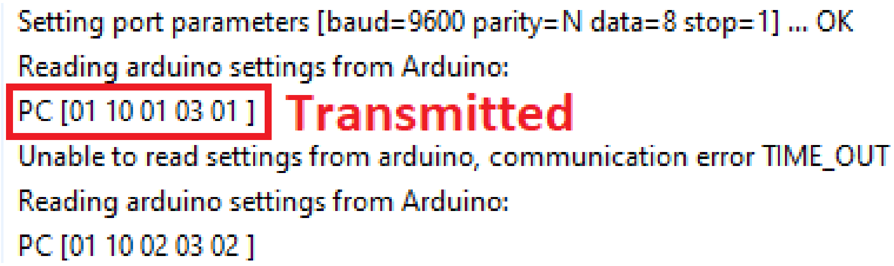
- ✓ Close the dialog box and look at the output window
- ✓ If the converter is working properly, you should see bytes echoed back:



```
Message
Setting port parameters [baud=9600 parity=N data=8 stop=1] ... OK
Reading arduino settings from Arduino:
PC [01 07 53 04 54 ]
.... µC [01 07 07 53 04 54 ] OK delay +- 32 ms
Unable to read settings from arduino, communication error NACK
```

Fig 2.27. Loopback on USB to serial converter success (received bytes are the same as the sent).

- ✓ If the converter is not working properly, you will see only transmitted bytes:



```
Setting port parameters [baud=9600 parity=N data=8 stop=1] ... OK
Reading arduino settings from Arduino:
PC [01 10 01 03 01 ]
Unable to read settings from arduino, communication error TIME_OUT
Reading arduino settings from Arduino:
PC [01 10 02 03 02 ]
```

Fig 2.28. Loopback on USB to serial converter (error, no bytes received)

If the USB to serial converter is not working properly, you need to solve driver, hardware or other issues before connecting to the Arduino board.

2.4.3 Serial 2 on MEGA 2560 and USB to serial converter

2.4.3.1 Required hardware

- Arduino Mega 2560
- USB to serial interface converter 5V compatible
- 3 Dupont male to female connectors
- USB A to B programming cable
- A USB A male to female connector (depending on the type of USB to Serial converter)

2.4.3.2 Hardware connections

- ✓ Arduino GND ⇔ CH340 GND
- ✓ Arduino 5V ⇔ CH340 5V
- ✓ Arduino RX2 – Pin 17 ⇔ CH340 TX
- ✓ Arduino TX2 – Pin 16⇔ CH340 RX
- ✓ Set jumper (if any) on your USB to serial convertor to 5V mode
- ✓ Connect your programming cable to the PC
- ✓ Connect your USB to serial converter to the PC

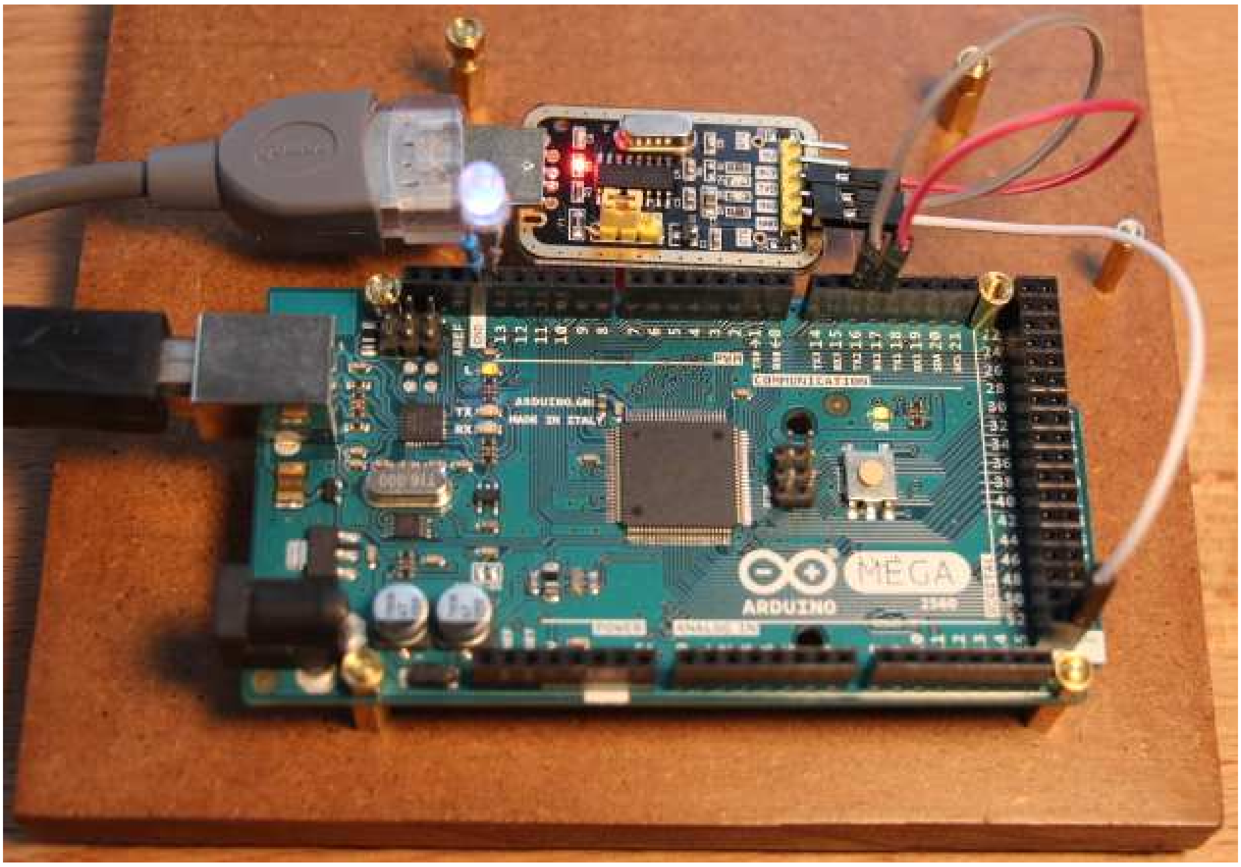
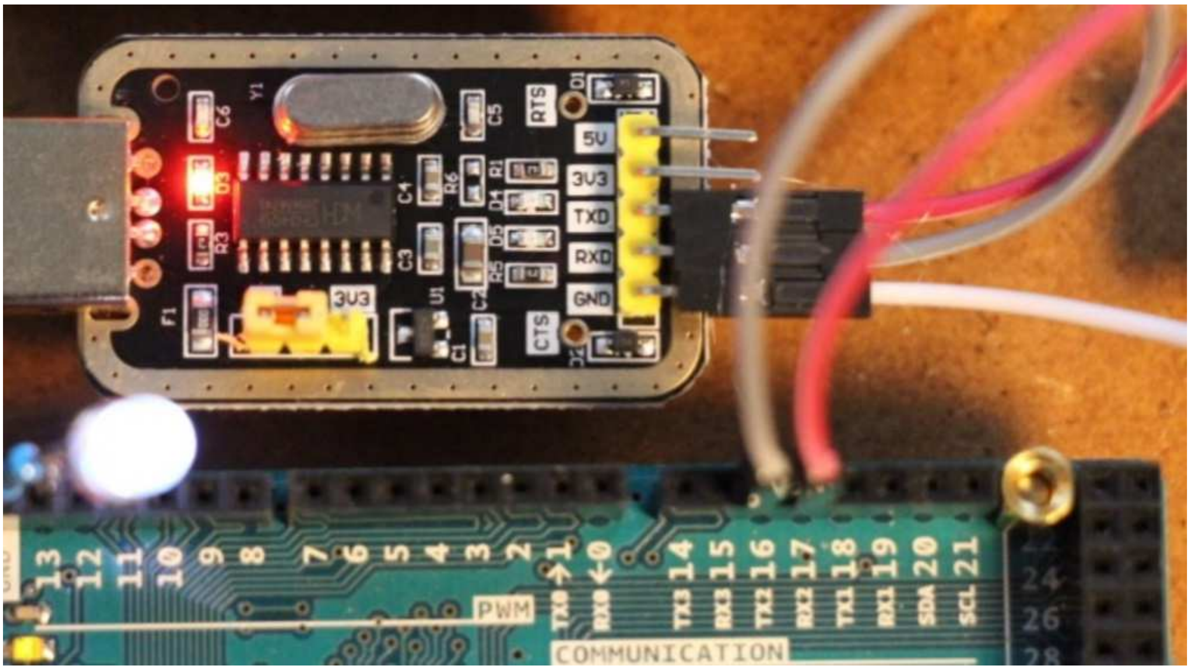


Fig 2.29. Arduino Mega with CH340 connected to serial 2.



- ○ Fig 2.30. Arduino Mega with CH340 connected to serial 2 (close up).

- ○ Note: 5V connection is not visible in the pictures above.
- ○ It is always better to first connect GND and then TX and RX in order to protect your board. Also first connect your board and then your converter to the PC. MEGA's are quite robust but my experience is that the MKR series is not so forgiving. (I broke 1 of them presumably by first disconnecting the main usb port and later the usb to serial converter from the PC.)

2.4.3.3 Load sketch

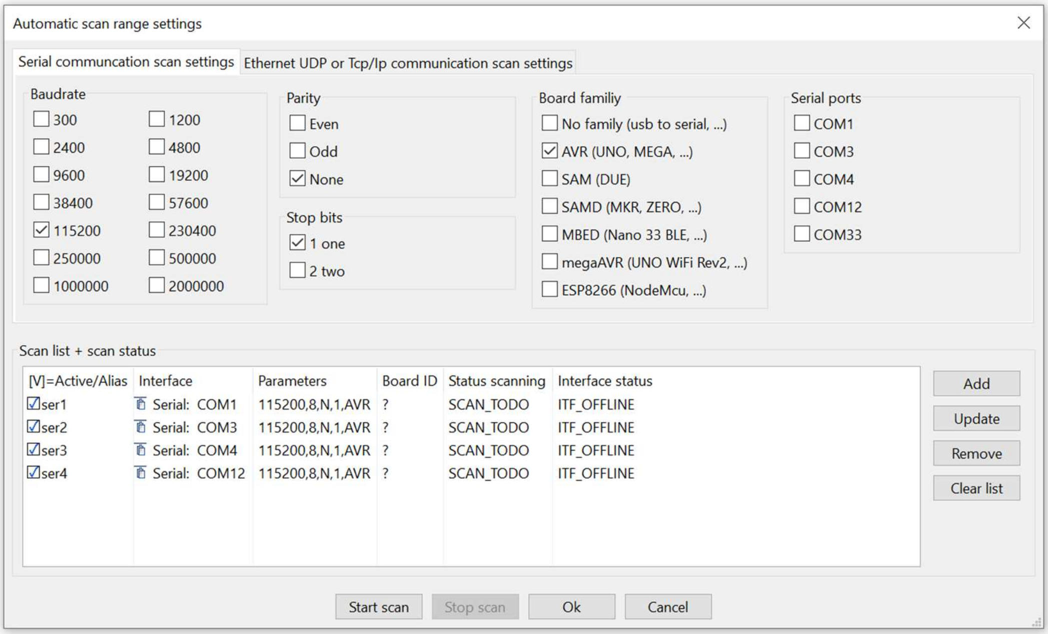
- ✓ Open the example via the menu "File\Examples\WawiSerialUsb\ WawiBinkSerialSer2" in the Arduino IDE.
- ✓ Compile and download the example.

```
/*
* Project Name : WawiBlinkSerialSer2
* File : WawiBlinkSerialSer2.ino
* /
#include <WawiSerialUsb.h>
WawiSerialUsb WawiSrv;
#define LED 13

// test variables for demo:
int delayOn = 500;
int delayOff = 500;
int blinkCounter = 0;

// make variables of interest known to WawiLib:
// this function is used in WawiSrv.begin(....)
void wawiVarDef()
{
    WawiSrv.wawiVar(delayOn);
    WawiSrv.wawiVar(delayOff);
    WawiSrv.wawiVar(blinkCounter);
}
void setup()
{
    Serial2.begin(115200);
    WawiSrv.begin(wawiVarDef, Serial2, "MyArduino");
    pinMode(LED, OUTPUT);
    WawiSrv.awaitPcConnect(30 * 1000);
    for (int i = 15; i > 0; i--)
    {
        WawiSrv.print("WawiSrv.Print() demo in setup(), counting down: ");
        WawiSrv.println(i);
        WawiSrv.delay(1000);
    }
}
void loop()
{
    blinkCounter++;
    WawiSrv.print("WawiSrv.Print() demo in loop() function, blinkcounter = ");
    WawiSrv.println(blinkCounter);

    WawiSrv.println("LED is ON.");
    digitalWrite(LED, HIGH);
    WawiSrv.delay(delayOn);

    WawiSrv.println("LED is OFF.");
    digitalWrite(LED, LOW);
    WawiSrv.delay(delayOff);

    WawiSrv.loop();
}
```

Fig 2.31. WawiBlinkSerialSer2 sketch.

2.4.3.4 Scan serial ports with WawiLib

- ✓ Start WawiLib
- ✓ Go to “settings/communication” interfaces
- ✓ Select the communication parameters as indicated in the next table. (select all serial ports)
- ✓ Press “Add”
- ✓ Press “Start scan”



Automatic scan range settings

Serial communcation scan settings | Ethernet UDP or Tcp/Ip communication scan settings

Baudrate
□ 300 □ 1200
□ 2400 □ 4800
□ 9600 □ 19200
□ 38400 □ 57600
☑ 115200 □ 230400
□ 250000 □ 500000
□ 1000000 □ 2000000

Parity
□ Even
□ Odd
☑ None

Stop bits
☑ 1 one
□ 2 two

Board familiy
□ No family (usb to serial, ...)
☑ AVR (UNO, MEGA, ...)
□ SAM (DUE)
□ SAMD (MKR, ZERO, ...)
□ MBED (Nano 33 BLE, ...)
□ megaAVR (UNO WiFi Rev2, ...)
□ ESP8266 (NodeMcu, ...)

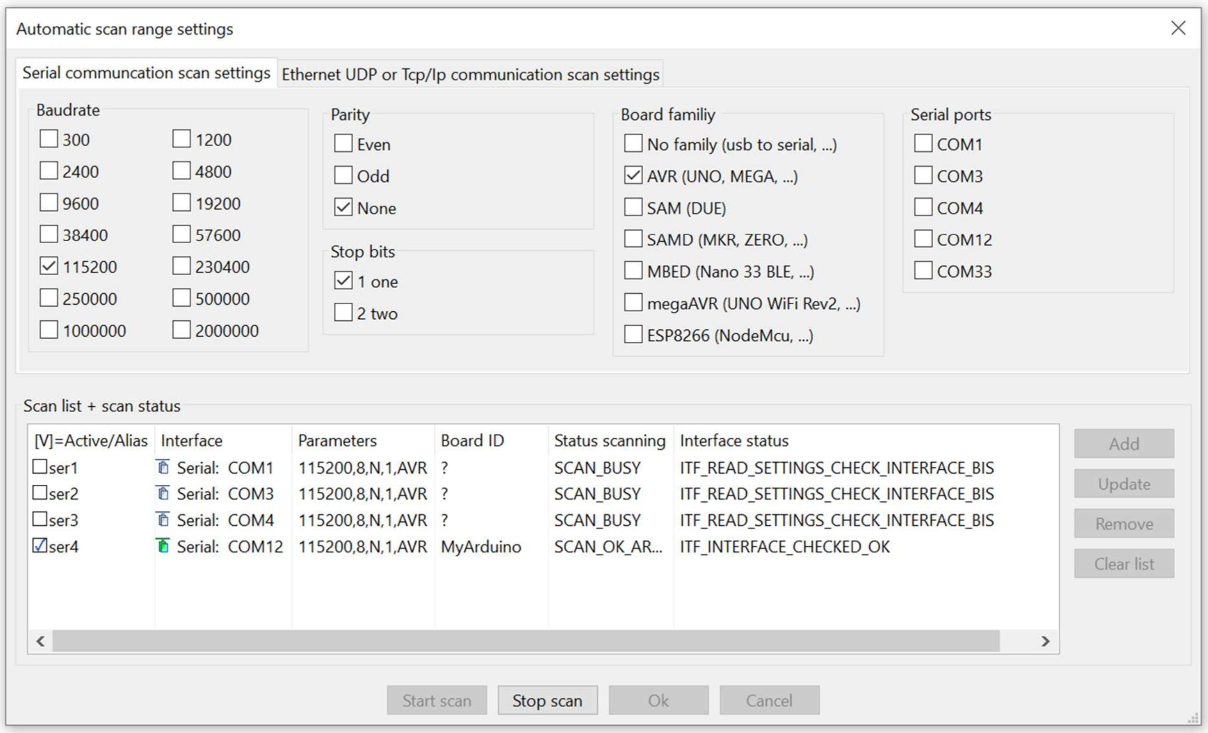
Serial ports
□ COM1
□ COM3
□ COM4
□ COM12
□ COM33

Scan list + scan status

[V]=Active/Alias	Interface	Parameters	Board ID	Status scanning	Interface status
☑ser1	Serial: COM1	115200,8,N,1,AVR	?	SCAN_TODO	ITF_OFFLINE
☑ser2	Serial: COM3	115200,8,N,1,AVR	?	SCAN_TODO	ITF_OFFLINE
☑ser3	Serial: COM4	115200,8,N,1,AVR	?	SCAN_TODO	ITF_OFFLINE
☑ser4	Serial: COM12	115200,8,N,1,AVR	?	SCAN_TODO	ITF_OFFLINE

Fig 2.32. Scan settings.

⇨ In the table “Scan list + scan status”, one of the icons in the “interface” column should turn green, indicating that a board with a WawiLib serial communication interface has been found.


Automatic scan range settings
Serial communcation scan settings
Ethernet UDP or Tcp/Ip communication scan settings

Baudrate
□ 300 □ 1200
□ 2400 □ 4800
□ 9600 □ 19200
□ 38400 □ 57600
☑ 115200 □ 230400
□ 250000 □ 500000
□ 1000000 □ 2000000

Parity
□ Even
□ Odd
☑ None

Stop bits
☑ 1 one
□ 2 two

Board familiy
□ No family (usb to serial, ...)
☑ AVR (UNO, MEGA, ...)
□ SAM (DUE)
□ SAMD (MKR, ZERO, ...)
□ MBED (Nano 33 BLE, ...)
□ megaAVR (UNO WiFi Rev2, ...)
□ ESP8266 (NodeMcu, ...)

Serial ports
□ COM1
□ COM3
□ COM4
□ COM12
□ COM33

Scan list + scan status

[V]=Active/Alias	Interface	Parameters	Board ID	Status scanning	Interface status
□ser1	Serial: COM1	115200,8,N,1,AVR	?	SCAN_BUSY	ITF_READ_SETTINGS_CHECK_INTERFACE_BIS
□ser2	Serial: COM3	115200,8,N,1,AVR	?	SCAN_BUSY	ITF_READ_SETTINGS_CHECK_INTERFACE_BIS
□ser3	Serial: COM4	115200,8,N,1,AVR	?	SCAN_BUSY	ITF_READ_SETTINGS_CHECK_INTERFACE_BIS
☑ser4	Serial: COM12	115200,8,N,1,AVR	MyArduino	SCAN_OK_AR...	ITF_INTERFACE_CHECKED_OK

Fig 2.33. Scan results.

✓ Click right on the table and select “remove inactive”

⇨ The interfaces that were not successfully scanned will be removed from the list

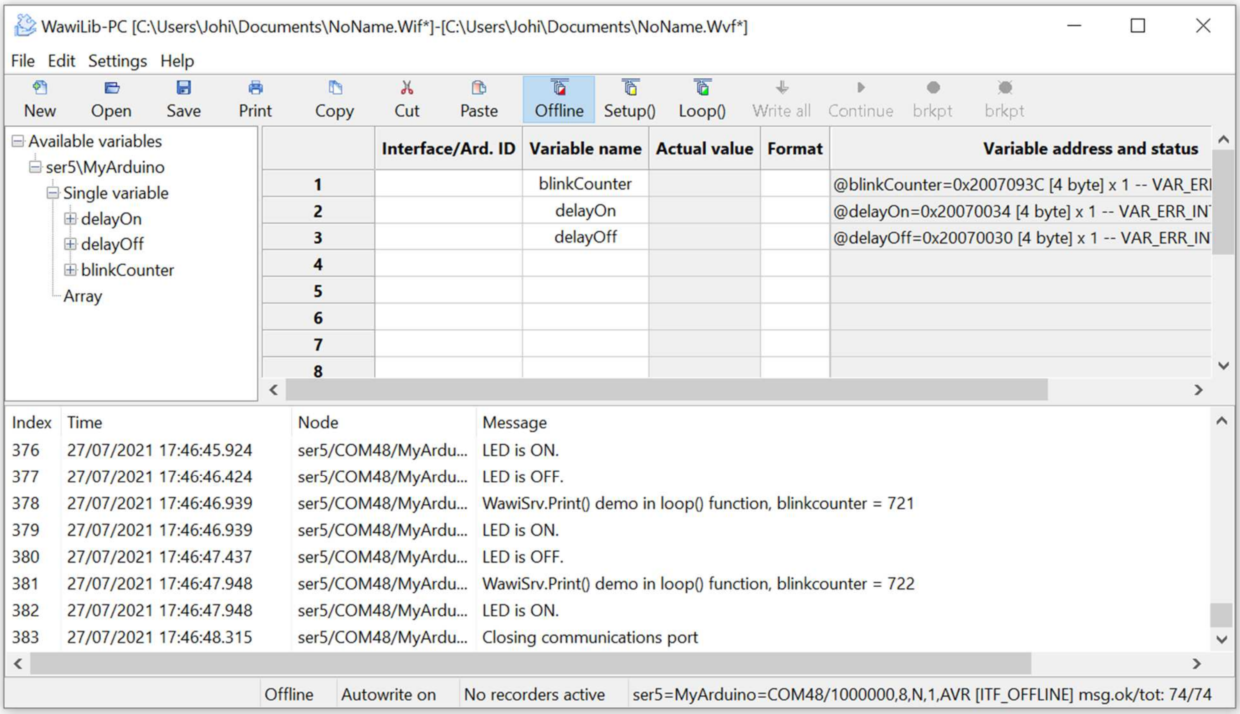
✓ Press OK

2.4.3.5 Monitor variables with WawiLib

✓ Press “Setup()” in the tool bar;

✓ Enter the variables to the main grid as indicated in the table below.

✓ Alternative: Use drag & drop to drag the variables from the tree to the grid table.


WawiLib-PC [C:\Users\Johi\Documents\NoName.Wif*]-[C:\Users\Johi\Documents\NoName.Wvf*]
File Edit Settings Help
New Open Save Print Copy Cut Paste Offline Setup() Loop() Write all Continue brkpt brkpt
Available variables
ser5\MyArduino
Single variable
delayOn
delayOff
blinkCounter
Array

	Interface/Ard. ID	Variable name	Actual value	Format	Variable address and status
1		blinkCounter			@blinkCounter=0x2007093C [4 byte] x 1 -- VAR_ERI
2		delayOn			@delayOn=0x20070034 [4 byte] x 1 -- VAR_ERR_IN
3		delayOff			@delayOff=0x20070030 [4 byte] x 1 -- VAR_ERR_IN
4					
5					
6					
7					
8					

```
Index  Time                      Node                 Message
376    27/07/2021 17:46:45.924   ser5/COM48/MyArdu...  LED is ON.
377    27/07/2021 17:46:46.424   ser5/COM48/MyArdu...  LED is OFF.
378    27/07/2021 17:46:46.939   ser5/COM48/MyArdu...  WawiSrv.Print() demo in loop() function, blinkcounter = 721
379    27/07/2021 17:46:46.939   ser5/COM48/MyArdu...  LED is ON.
380    27/07/2021 17:46:47.437   ser5/COM48/MyArdu...  LED is OFF.
381    27/07/2021 17:46:47.948   ser5/COM48/MyArdu...  WawiSrv.Print() demo in loop() function, blinkcounter = 722
382    27/07/2021 17:46:47.948   ser5/COM48/MyArdu...  LED is ON.
383    27/07/2021 17:46:48.315   ser5/COM48/MyArdu...  Closing communications port
```

Offline | Autowrite on | No recorders active | ser5=MyArduino=COM48/1000000,8,N,1,AVR [ITF_OFFLINE] msg.ok/tot: 74/74

Fig 2.34. Add variables to be monitored to the table.

✓ Enable “Display Print messages” and “Display communication protocol messages” using the popup menu of the output window (right click on the output window to make the menu appear)

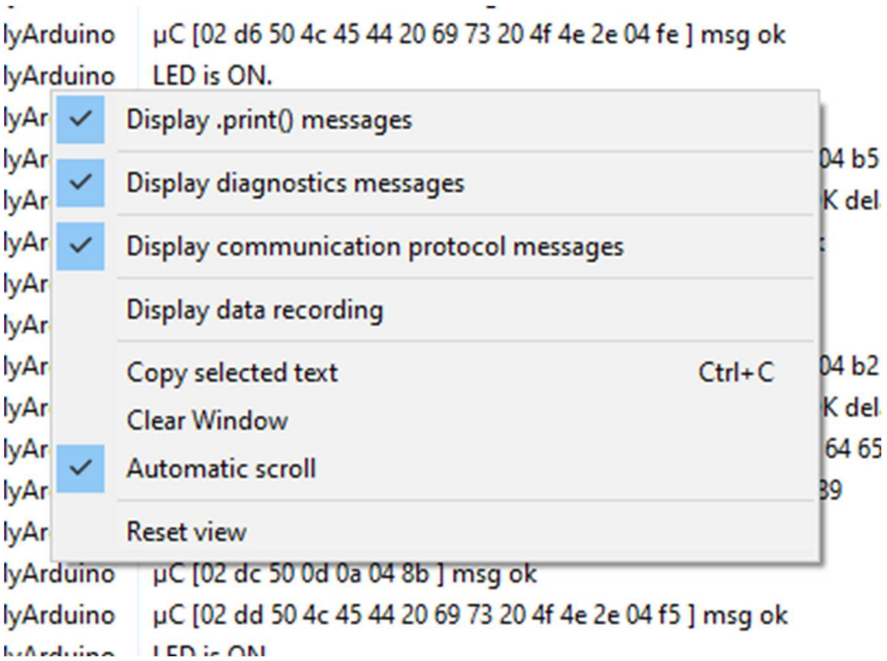
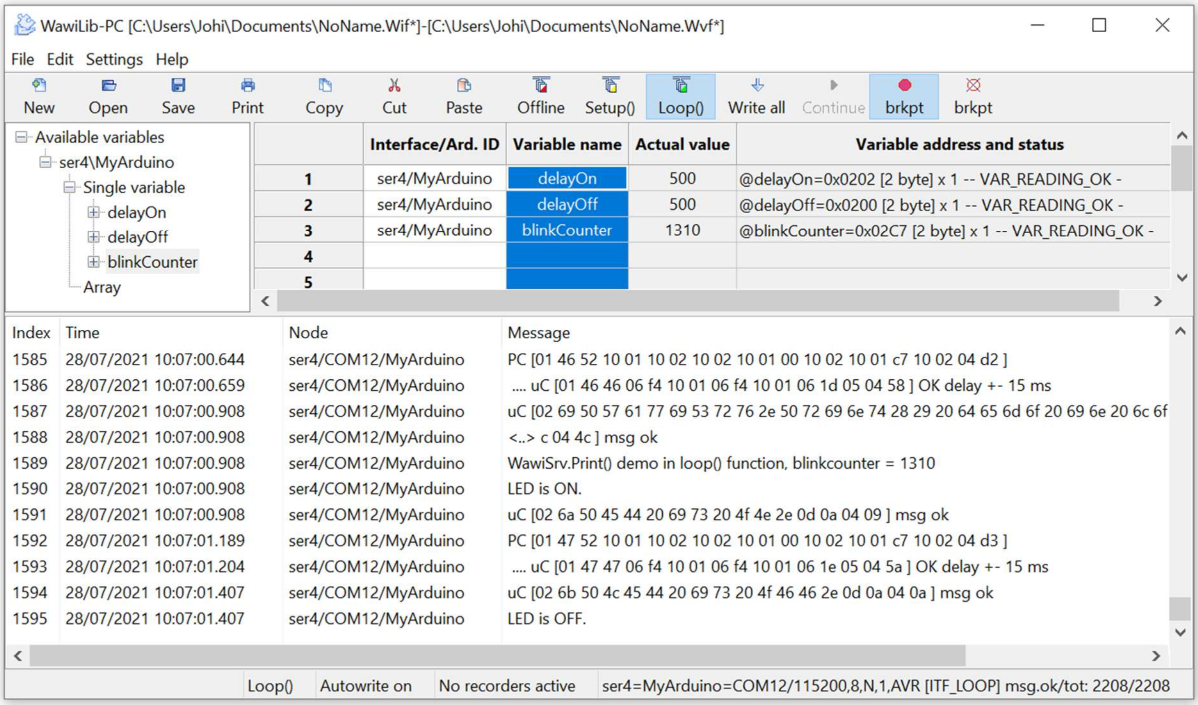

```
lyArduino   µC [02 d6 50 4c 45 44 20 69 73 20 4f 4e 2e 04 fe ] msg ok
lyArduino   LED is ON.
lyArduino   µC [02 dc 50 0d 0a 04 8b ] msg ok
lyArduino   µC [02 dd 50 4c 45 44 20 69 73 20 4f 4e 2e 04 f5 ] msg ok
```
✓ Display .print() messages
✓ Display diagnostics messages
✓ Display communication protocol messages
Display data recording
Copy selected text Ctrl+C
Clear Window
✓ Automatic scroll
Reset view

Fig 2.35. WawiLib Output window settings.


WawiLib-PC [C:\Users\Johi\Documents\NoName.Wif*]-[C:\Users\Johi\Documents\NoName.Wvf*]
File Edit Settings Help
New Open Save Print Copy Cut Paste Offline Setup() Loop() Write all Continue brkpt brkpt
Available variables
ser4\MyArduino
Single variable
delayOn
delayOff
blinkCounter
Array

	Interface/Ard. ID	Variable name	Actual value	Variable address and status
1	ser4/MyArduino	delayOn	500	@delayOn=0x0202 [2 byte] x 1 -- VAR_READING_OK -
2	ser4/MyArduino	delayOff	500	@delayOff=0x0200 [2 byte] x 1 -- VAR_READING_OK -
3	ser4/MyArduino	blinkCounter	1310	@blinkCounter=0x02C7 [2 byte] x 1 -- VAR_READING_OK -
4				
5				

```
Index  Time                     Node                    Message
1585   28/07/2021 10:07:00.644  ser4/COM12/MyArduino    PC [01 46 52 10 01 10 02 10 02 10 01 00 10 02 10 01 c7 10 02 04 d2 ]
1586   28/07/2021 10:07:00.659  ser4/COM12/MyArduino     .... uC [01 46 46 06 f4 10 01 06 f4 10 01 06 1d 05 04 58 ] OK delay +- 15 ms
1587   28/07/2021 10:07:00.908  ser4/COM12/MyArduino    uC [02 69 50 57 61 77 69 53 72 76 2e 50 72 69 6e 74 28 29 20 64 65 6d 6f 20 69 6e 20 6c 6f
1588   28/07/2021 10:07:00.908  ser4/COM12/MyArduino    <..> c 04 4c ] msg ok
1589   28/07/2021 10:07:00.908  ser4/COM12/MyArduino    WawiSrv.Print() demo in loop() function, blinkcounter = 1310
1590   28/07/2021 10:07:00.908  ser4/COM12/MyArduino    LED is ON.
1591   28/07/2021 10:07:00.908  ser4/COM12/MyArduino    uC [02 6a 50 45 44 20 69 73 20 4f 4e 2e 0d 0a 04 09 ] msg ok
1592   28/07/2021 10:07:01.189  ser4/COM12/MyArduino    PC [01 47 52 10 01 10 02 10 02 10 01 00 10 02 10 01 c7 10 02 04 d3 ]
1593   28/07/2021 10:07:01.204  ser4/COM12/MyArduino     .... uC [01 47 47 06 f4 10 01 06 f4 10 01 06 1e 05 04 5a ] OK delay +- 15 ms
1594   28/07/2021 10:07:01.407  ser4/COM12/MyArduino    uC [02 6b 50 4c 45 44 20 69 73 20 4f 46 46 2e 0d 0a 04 0a ] msg ok
1595   28/07/2021 10:07:01.407  ser4/COM12/MyArduino    LED is OFF.
```

```
Loop()   Autowrite on   No recorders active   ser4=MyArduino=COM12/115200,8,N,1,AVR [ITF_LOOP] msg.ok/tot: 2208/2208
```

Fig 2.36. Monitor variables and debug output in output window.

⇨ You can see the output of .printf() in the window together with the protocol as it evolves. The messages marked PC are transmitted by the PC. Messages marked …µC are replies and messages marked µC are telegrams sent by the Arduino on his or her own initiative.

2.4.4 Serial 1 on MKR1000/MKR1010 serial1 with an USB to serial converter

2.4.4.1 Introduction

One of the nice things about WawiLib is that you do not have to choose between the “Serial Monitor Window” and WawiLib. You can use both features at the same time. In the next example, I will use WawiLib to monitor and modify the “WawiBlink” variables and send debug output to the Serial Monitor window at the same time.

2.4.4.2 Required hardware

- An Arduino MKR1000 or MKR1010 (named MKR10X0 in this text)
- A USB to serial interface converter that is compatible with the 3.3V standard
- A USB A male to female extension cable (depending on the type of USB to Serial converter)
- A USB A to micro B cable
- 3 Dupont male to female connectors

2.4.4.3 Hardware connections

- ✓ Arduino MKR10X0 GND ⇔ CH340 GND
- ✓ Arduino MKR10X0 5V – USB to serial 5V
- ✓ Arduino MKR10X0 RX – Pin 13 ⇔ CH340 TX
- ✓ Arduino MKR10X0 TX – Pin 14 ⇔ CH340 RX
- ✓ Set jumper (if any) on your USB to serial convertor to 3.3V mode
- ✓ Connect the MKR10X0 to the PC
- ✓ Connect the USB to serial converter to the PC
- ✓ MKR 1010 connections:

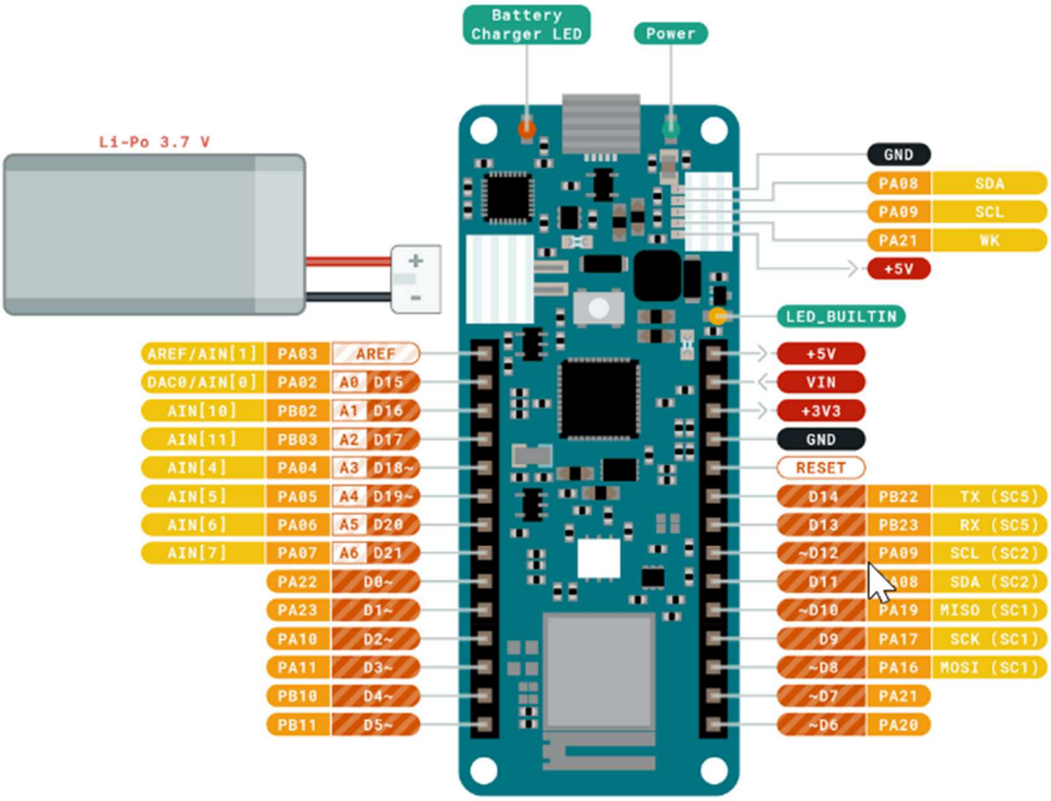


Fig 2.37. MKR 1010 layout.

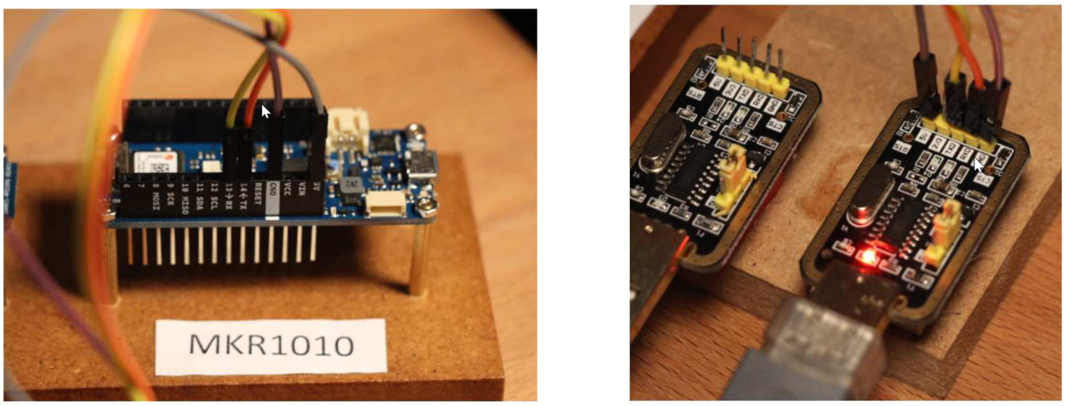


Fig 2.38. MKR 1010 connection pictures.

- ✓ MKR 1000 connections:

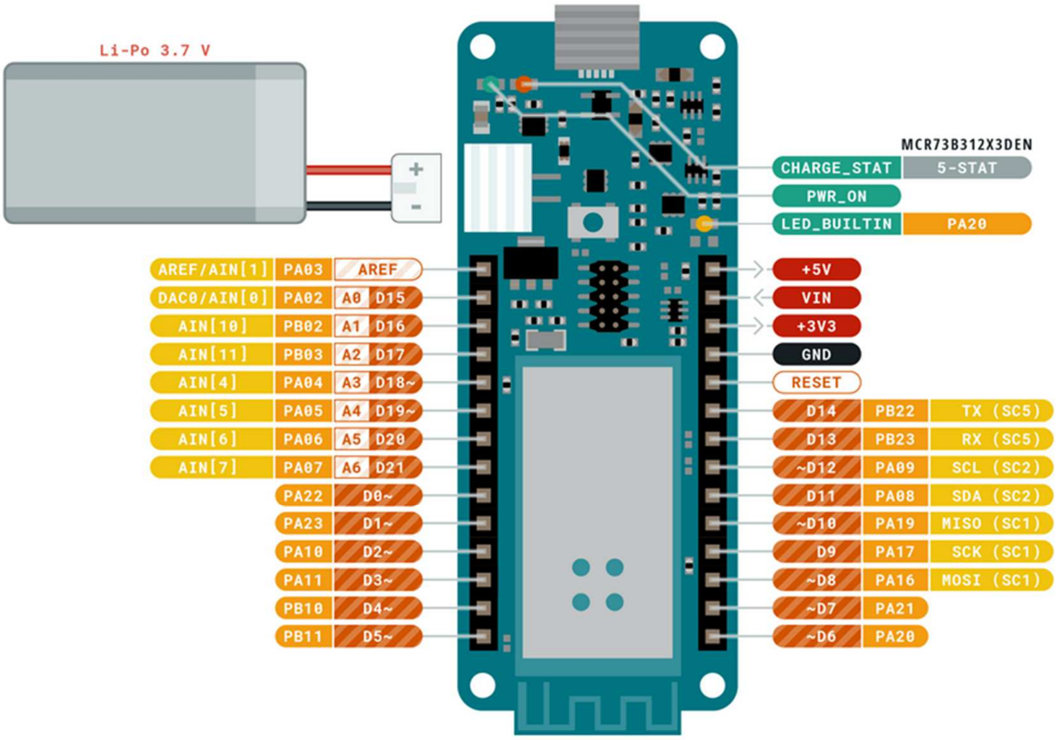


Fig 2.39. MKR 1000 layout.

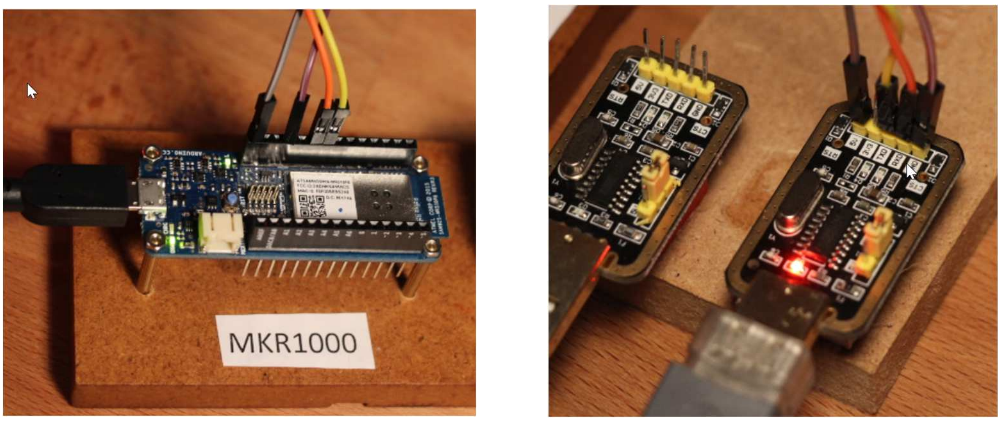


Fig 2.40. MKR 1000 connection pictures.

2.4.4.4 Load sketch

✓ Open the example via the menu “File\Examples\WawiSerialUsb\WawiBlinkMKR10X0SerialSer1” in the Arduino IDE

✓ Compile and download the example

```
/*
* Project Name: WawiBlinkMkr10X0SerialSer1
* File: WawiBlinkMkr10X0SerialSer1.ino
*/
#include <WawiSerialUsb.h>
#define LED 6
int delayOn = 500;
int delayOff = 500;
int blinkCounter = 0;

WawiSerialUsb WawiSrv;

void wawiVarDef()
{
    WawiSrv.wawiVar(delayOn);
    WawiSrv.wawiVar(delayOff);
    WawiSrv.wawiVar(blinkCounter);
}

void setup()
{
    Serial.begin(115200);  // serial port for IDE Serial Monitor Output
    Serial1.begin(115200); // serial port used for WawiLib communication
    WawiSrv.begin(wawiVarDef, Serial1, "MyArduino");
    pinMode(LED, OUTPUT);
}

void loop()
{
    blinkCounter++;
    WawiSrv.print("WawiSrv.Print() demo in loop() function, blinkcounter = ");
    WawiSrv.println(blinkCounter);

    WawiSrv.println("LED is ON.");
    digitalWrite(LED, HIGH);
    WawiSrv.delay(delayOn);

    WawiSrv.println("LED is OFF.");
    digitalWrite(LED, LOW);
    WawiSrv.delay(delayOff);

    WawiSrv.loop();
}
```

Fig 2.41. WawiBlinkMkr10X0SerialSer1 sketch source code.

2.4.4.5 Scan serial ports with WawiLib

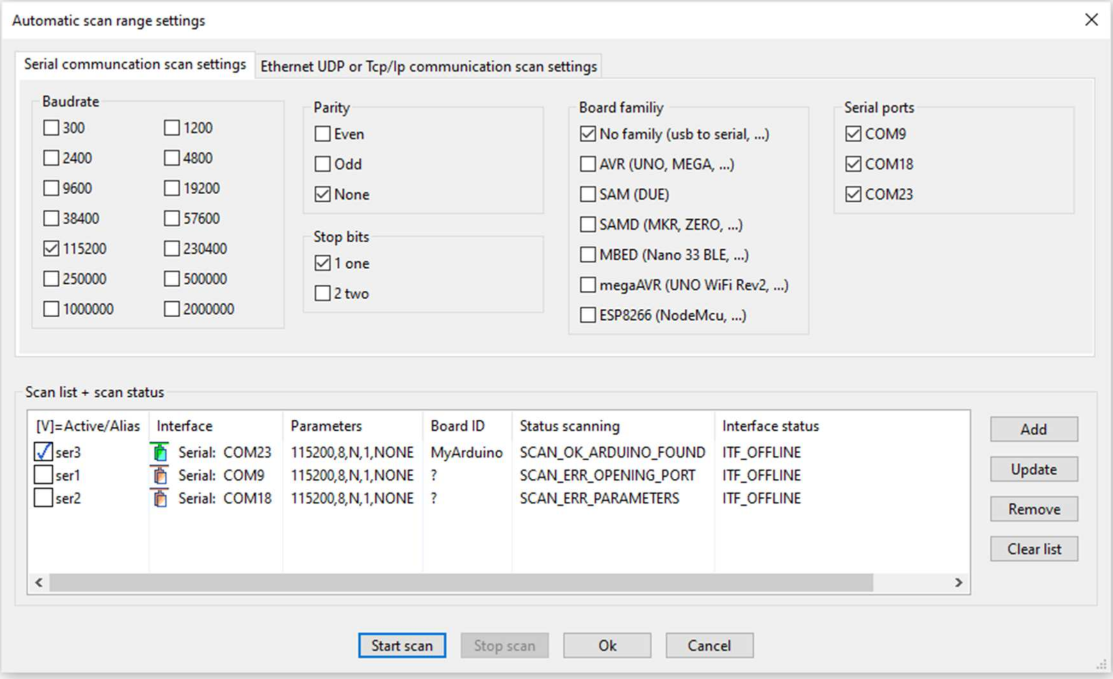
✓ Start WawiLib

✓ Go to “settings/communication” interfaces

✓ Select the communication parameters as indicated in the next table (select all serial ports)

✓ Press “Add”

✓ Press “Start scan”



Automatic scan range settings

Serial communcation scan settings | Ethernet UDP or Tcp/Ip communication scan settings

Baudrate: □ 300 □ 1200 □ 2400 □ 4800 □ 9600 □ 19200 □ 38400 □ 57600 ☑ 115200 □ 230400 □ 250000 □ 500000 □ 1000000 □ 2000000

Parity: □ Even □ Odd ☑ None

Stop bits: ☑ 1 one □ 2 two

Board familiy: ☑ No family (usb to serial, ...) □ AVR (UNO, MEGA, ...) □ SAM (DUE) □ SAMD (MKR, ZERO, ...) □ MBED (Nano 33 BLE, ...) □ megaAVR (UNO WiFi Rev2, ...) □ ESP8266 (NodeMcu, ...)

Serial ports: ☑ COM9 ☑ COM18 ☑ COM23

Scan list + scan status

[V]=Active/Alias	Interface	Parameters	Board ID	Status scanning	Interface status
☑ ser3	Serial: COM23	115200,8,N,1,NONE	MyArduino	SCAN_OK_ARDUINO_FOUND	ITF_OFFLINE
□ ser1	Serial: COM9	115200,8,N,1,NONE	?	SCAN_ERR_OPENING_PORT	ITF_OFFLINE
□ ser2	Serial: COM18	115200,8,N,1,NONE	?	SCAN_ERR_PARAMETERS	ITF_OFFLINE

Fig 2.42. Communication parameter settings.

⇨ In the table “Scan list + scan status”, one of the icons in the “interface” column should turn green, indicating that a board with a WawiLib serial communication interface has been found.

✓ Click right on the table and select “remove inactive”

⇨ The interfaces that were not successfully scanned are removed from the list

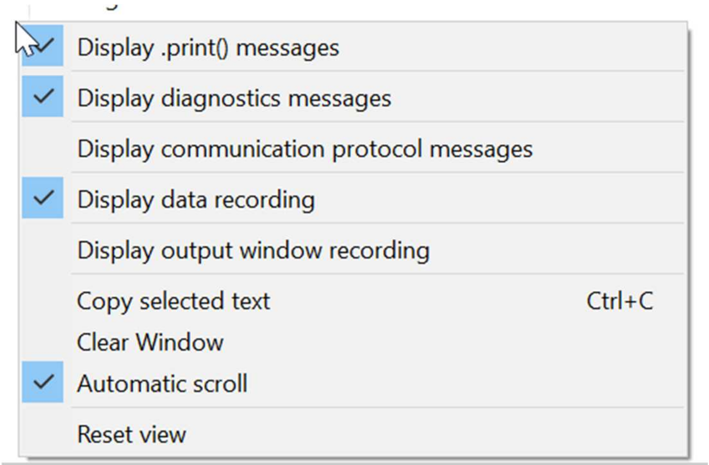
2.4.4.6 Monitor variables with WawiLib

✓ Press “Setup()” in the tool bar;

✓ Enter the variables to the main grid as indicated in the table below.

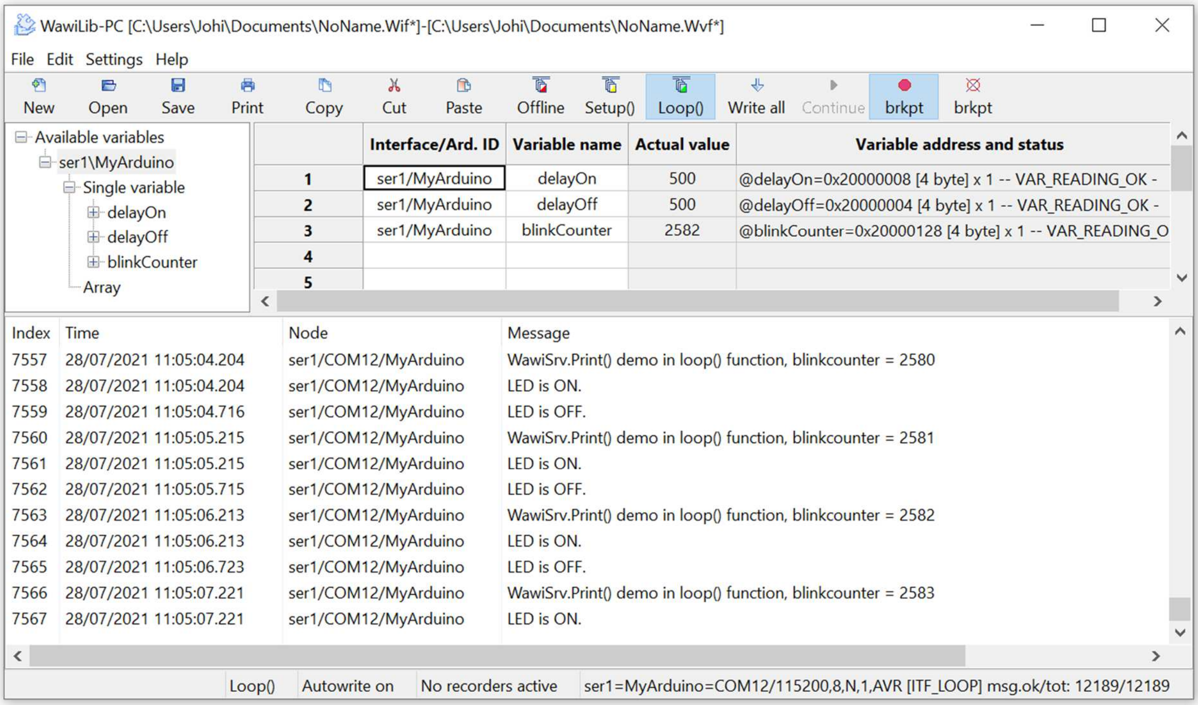
✓ Alternative: Use drag & drop to drag the variables from the tree to the grid table.

✓ Enable the settings in the output window as in the next figure.



- ✓ Display .print() messages
- ✓ Display diagnostics messages
- Display communication protocol messages
- ✓ Display data recording
- Display output window recording
- Copy selected text Ctrl+C
- Clear Window
- ✓ Automatic scroll
- Reset view

✓ Fig 2.43. WawiLib Output window settings.



```
Available variables
  ser1\MyArduino
    Single variable
      delayOn
      delayOff
      blinkCounter
    Array
```

	Interface/Ard. ID	Variable name	Actual value	Variable address and status
1	ser1/MyArduino	delayOn	500	@delayOn=0x20000008 [4 byte] x 1 -- VAR_READING_OK -
2	ser1/MyArduino	delayOff	500	@delayOff=0x20000004 [4 byte] x 1 -- VAR_READING_OK -
3	ser1/MyArduino	blinkCounter	2582	@blinkCounter=0x20000128 [4 byte] x 1 -- VAR_READING_O
4				
5				

```
Index  Time                     Node                  Message
7557   28/07/2021 11:05:04.204  ser1/COM12/MyArduino  WawiSrv.Print() demo in loop() function, blinkcounter = 2580
7558   28/07/2021 11:05:04.204  ser1/COM12/MyArduino  LED is ON.
7559   28/07/2021 11:05:04.716  ser1/COM12/MyArduino  LED is OFF.
7560   28/07/2021 11:05:05.215  ser1/COM12/MyArduino  WawiSrv.Print() demo in loop() function, blinkcounter = 2581
7561   28/07/2021 11:05:05.215  ser1/COM12/MyArduino  LED is ON.
7562   28/07/2021 11:05:05.715  ser1/COM12/MyArduino  LED is OFF.
7563   28/07/2021 11:05:06.213  ser1/COM12/MyArduino  WawiSrv.Print() demo in loop() function, blinkcounter = 2582
7564   28/07/2021 11:05:06.213  ser1/COM12/MyArduino  LED is ON.
7565   28/07/2021 11:05:06.723  ser1/COM12/MyArduino  LED is OFF.
7566   28/07/2021 11:05:07.221  ser1/COM12/MyArduino  WawiSrv.Print() demo in loop() function, blinkcounter = 2583
7567   28/07/2021 11:05:07.221  ser1/COM12/MyArduino  LED is ON.
```

Loop() | Autowrite on | No recorders active | ser1=MyArduino=COM12/115200,8,N,1,AVR [ITF_LOOP] msg.ok/tot: 12189/12189

Fig 2.44. Monitoring variables and .print() output with WawiLib.

2.4.5 Demo: Serial in on Nano 33 BLE SENSE/Nano 33 IOT and USB to serial converter

2.4.5.1 Introduction

This demo does the same as the previous one, but on another Arduino platform. I could have combined both demos in one chapter but, as the previous one already handles MKR1000 & MRK1010, I chose not to. The reason is I sincerely dislike to read multi-model manuals with lots of "if then else" statements. Such manuals are a source of confusion to the reader.

2.4.5.2 Required hardware

- An Arduino Nano 33 BLE
- A USB to serial interface converter that is compatible with the 3.3V standard
- A USB A male to female connector (depending on the type of USB to Serial converter)
- A USB A to micro B cable to connect your shield to your PC
- 3 Dupont male to female connectors

2.4.5.3 Hardware connections

- ✓ Arduino Nano 33 BLE GND – pin 4 ⇔ CH340 GND
- ✓ Arduino Nano 5V – USB to serial 5V
- ✓ Arduino Nano 33 BLE TX – pin 0 ⇔ CH340 RX
- ✓ Arduino Nano 33 BLE RX – pin 1 ⇔ CH340 TX
- ✓ Set jumper (if any) on your USB to serial convertor to 3.3V mode
- ✓ Connect your programming cable to the PC
- ✓ Nano 33 BLE SENSE connections pictures:

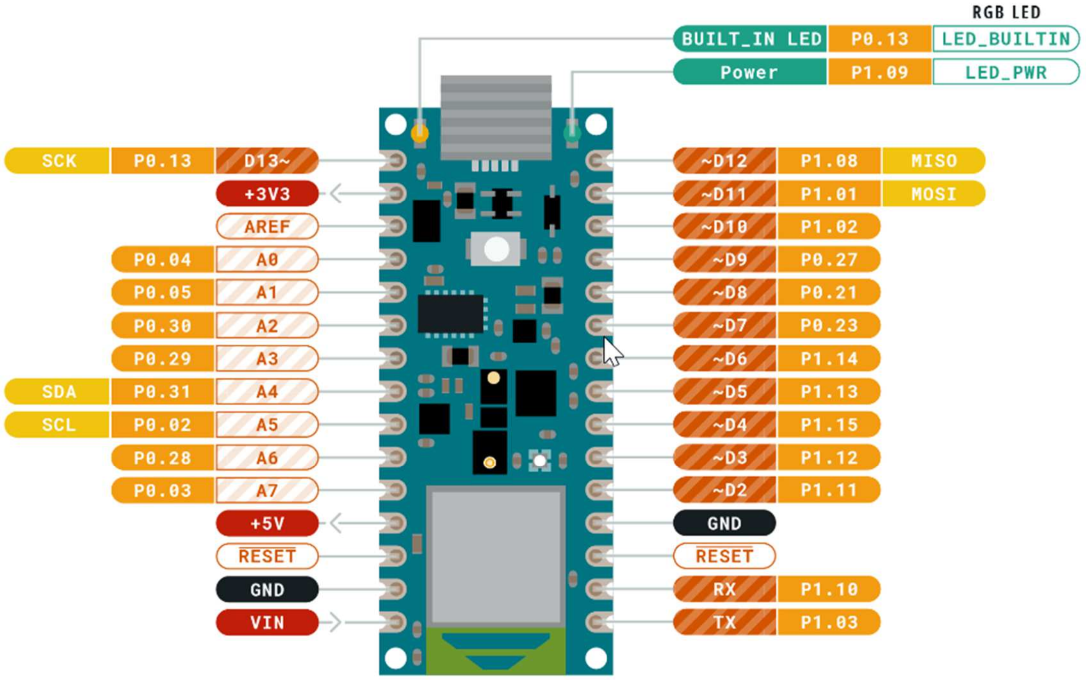


Fig 2.45. Nano 33 BLE sense layout.

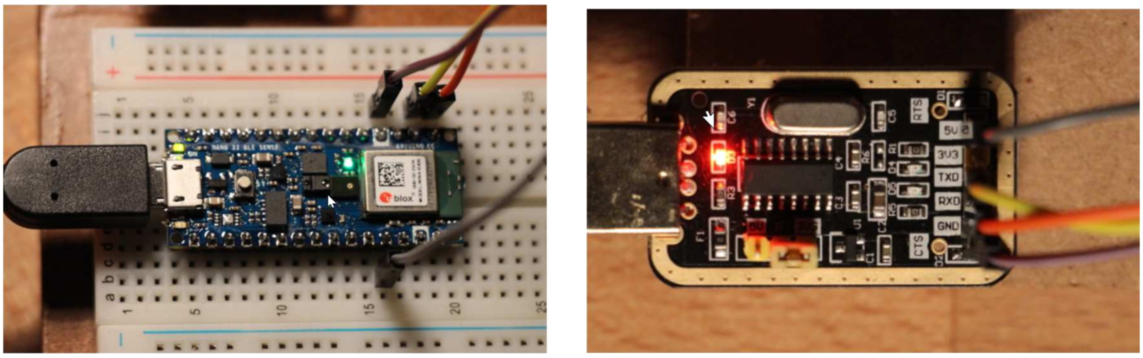


Fig 2.46. Nano BLE sense connection pictures.

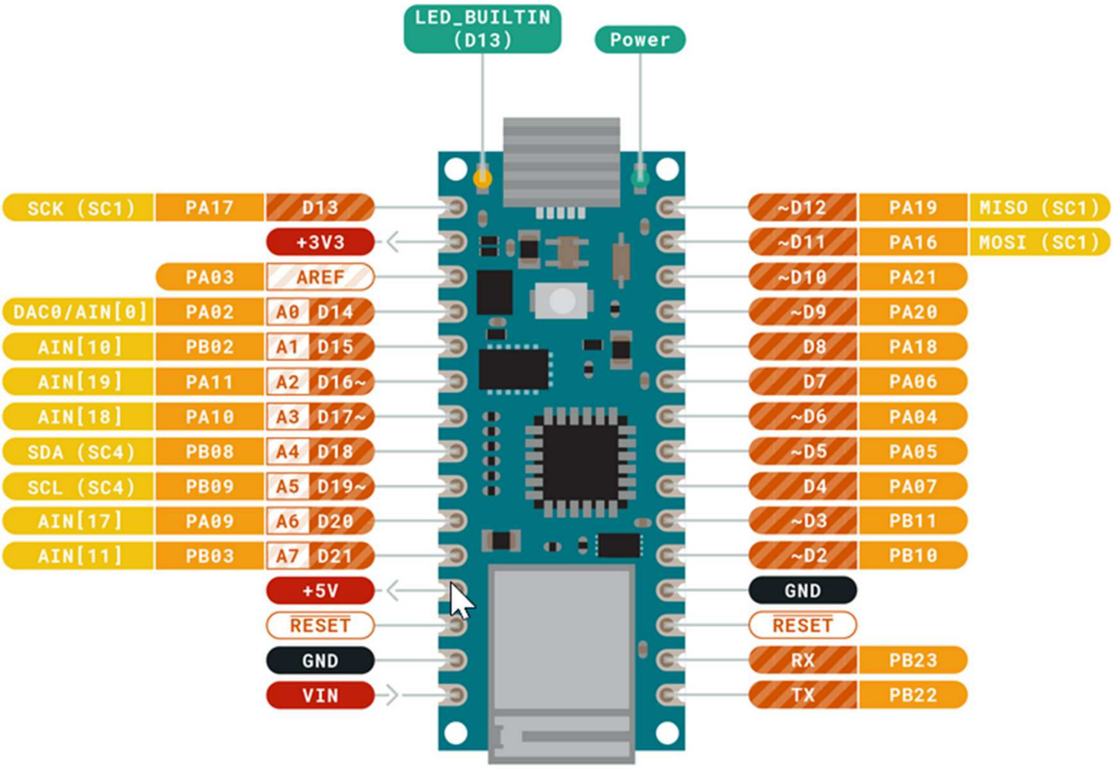


Fig. 2.47. Nano 33 IOT layout

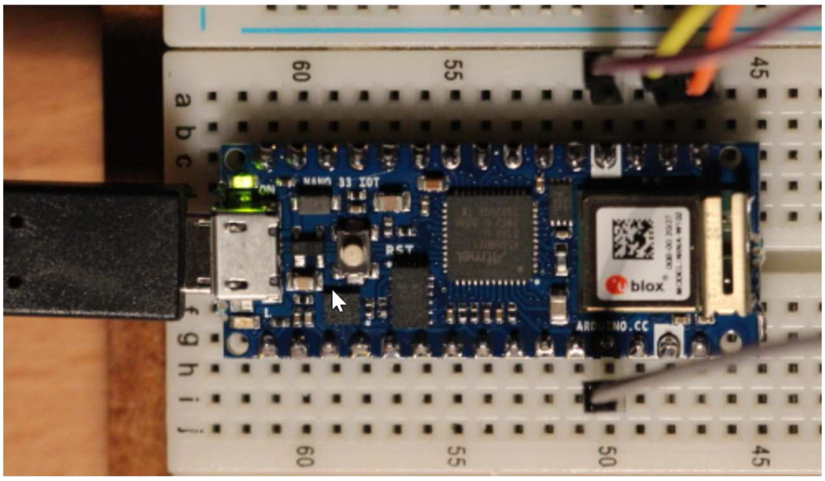


Fig 2.48. Nano 33 IOT connection pictures.

2.4.5.4 Load sketch

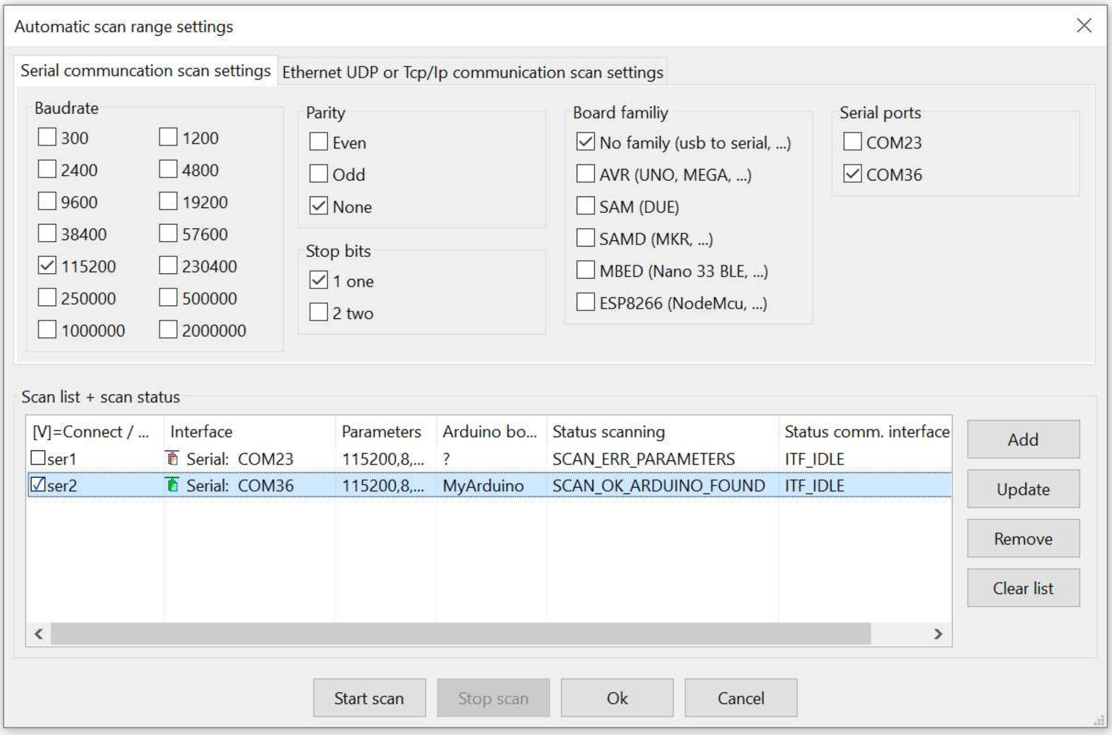
- ✓ Open the example via the menu “File\Examples\WawiSerialUsb\WawiBlinkNano33SerialSer1” in the Arduino IDE
- ✓ Compile and download the example
- ✓ Connect your USB to serial converter to the PC

```
/*
* Project Name: WawiBlinkNano33SerialSer1
* File: WawiBlinkNano33SerialSer1.ino
*/

#include <WawiSerialUsb.h>
#define LED 23
int delayOn = 500;
int delayOff = 500;
int blinkCounter = 0;

WawiSerialUsb WawiSrv;

void wawiVarDef()
{
    WawiSrv.wawiVar(delayOn);
    WawiSrv.wawiVar(delayOff);
    WawiSrv.wawiVar(blinkCounter);
}

void setup()
{
    Serial.begin(115200);  // serial port for IDE Serial Monitor Output
    Serial1.begin(115200); // serial port used for WawiLib communication
    WawiSrv.begin(wawiVarDef, Serial1, "MyArduino");
    pinMode(LED, OUTPUT);
}

void loop()
{
    Serial.print("blinkCounter=");
    Serial.println(blinkCounter);
    blinkCounter++;
    WawiSrv.print("WawiSrv.Print() demo in loop() function, blinkcounter = ");
    WawiSrv.println(blinkCounter);

    WawiSrv.println("LED is ON.");
    digitalWrite(LED, HIGH);
    WawiSrv.delay(delayOn);

    WawiSrv.println("LED is OFF.");
    digitalWrite(LED, LOW);
    WawiSrv.delay(delayOff);

    WawiSrv.loop();
}
```

Fig 2.49. WawiBlinkNano33SerialSer1 demo sketch.

2.4.5.5 Scan serial ports with WawiLib

- ✓ Start WawiLib
- ✓ Go to "settings/communication" interfaces
- ✓ Select the communication parameters as indicated in the next table (select all serial ports if you do not know what port to choose)
- ✓ Press "Add"
- ✓ Press "Start scan"



Automatic scan range settings

Serial communcation scan settings | Ethernet UDP or Tcp/Ip communication scan settings

Baudrate
☐ 300 ☐ 1200
☐ 2400 ☐ 4800
☐ 9600 ☐ 19200
☐ 38400 ☐ 57600
☑ 115200 ☐ 230400
☐ 250000 ☐ 500000
☐ 1000000 ☐ 2000000

Parity
☐ Even
☐ Odd
☑ None

Stop bits
☑ 1 one
☐ 2 two

Board familiy
☑ No family (usb to serial, ...)
☐ AVR (UNO, MEGA, ...)
☐ SAM (DUE)
☐ SAMD (MKR, ...)
☐ MBED (Nano 33 BLE, ...)
☐ ESP8266 (NodeMcu, ...)

Serial ports
☐ COM23
☑ COM36

Scan list + scan status

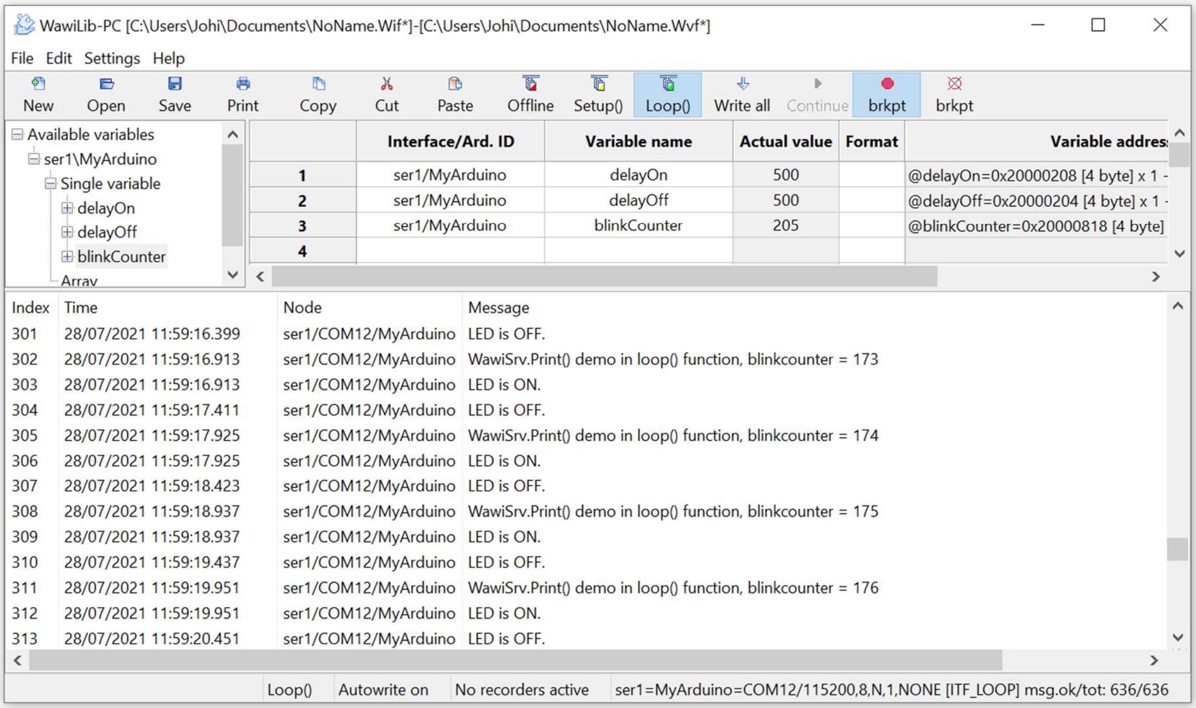
[V]=Connect / ...	Interface	Parameters	Arduino bo...	Status scanning	Status comm. interface
☐ser1	Serial: COM23	115200,8,...	?	SCAN_ERR_PARAMETERS	ITF_IDLE
☑ser2	Serial: COM36	115200,8,...	MyArduino	SCAN_OK_ARDUINO_FOUND	ITF_IDLE

Fig 2.49. WawiBlinkNano33SerialSer1 connection settings.

- ⇨ In the table "Scan list + scan status", one of the icons in the "interface" column should turn green, indicating that a board with a WawiLib serial communication interface has been found.
- ✓ Click right on the table and select "remove inactive"
- ⇨ The interfaces that were not successfully scanned are removed from the list

2.4.5.6 Monitor variables with WawiLib

- ✓ Press “Setup()” in the tool bar;
- ✓ Enter the variables to the main grid as indicated in the table below.
- ✓ Alternative: Use drag & drop to drag the variables from the tree to the grid table.


WawiLib-PC [C:\Users\Johi\Documents\NoName.Wif*]-[C:\Users\Johi\Documents\NoName.Wvf*]
File Edit Settings Help
New Open Save Print Copy Cut Paste Offline Setup() Loop() Write all Continue brkpt brkpt
Available variables
ser1\MyArduino
Single variable
delayOn
delayOff
blinkCounter
Array

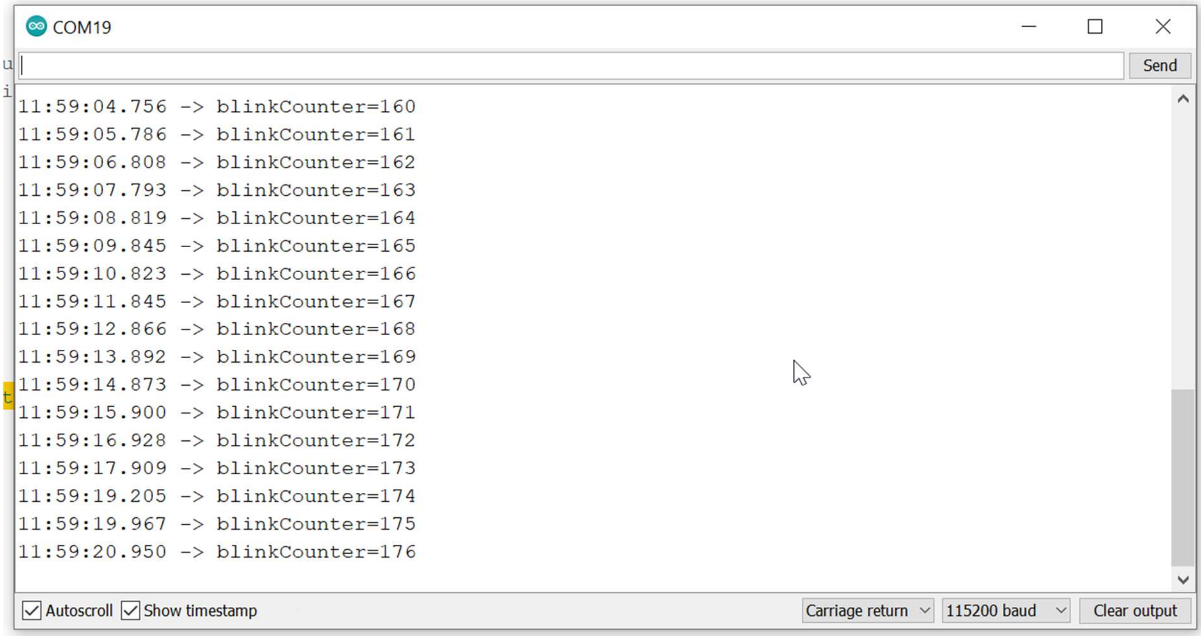
	Interface/Ard. ID	Variable name	Actual value	Format	Variable addres
1	ser1/MyArduino	delayOn	500		@delayOn=0x20000208 [4 byte] x 1 -
2	ser1/MyArduino	delayOff	500		@delayOff=0x20000204 [4 byte] x 1 -
3	ser1/MyArduino	blinkCounter	205		@blinkCounter=0x20000818 [4 byte]
4					

```
Index Time                     Node                Message
301   28/07/2021 11:59:16.399  ser1/COM12/MyArduino LED is OFF.
302   28/07/2021 11:59:16.913  ser1/COM12/MyArduino WawiSrv.Print() demo in loop() function, blinkcounter = 173
303   28/07/2021 11:59:16.913  ser1/COM12/MyArduino LED is ON.
304   28/07/2021 11:59:17.411  ser1/COM12/MyArduino LED is OFF.
305   28/07/2021 11:59:17.925  ser1/COM12/MyArduino WawiSrv.Print() demo in loop() function, blinkcounter = 174
306   28/07/2021 11:59:17.925  ser1/COM12/MyArduino LED is ON.
307   28/07/2021 11:59:18.423  ser1/COM12/MyArduino LED is OFF.
308   28/07/2021 11:59:18.937  ser1/COM12/MyArduino WawiSrv.Print() demo in loop() function, blinkcounter = 175
309   28/07/2021 11:59:18.937  ser1/COM12/MyArduino LED is ON.
310   28/07/2021 11:59:19.437  ser1/COM12/MyArduino LED is OFF.
311   28/07/2021 11:59:19.951  ser1/COM12/MyArduino WawiSrv.Print() demo in loop() function, blinkcounter = 176
312   28/07/2021 11:59:19.951  ser1/COM12/MyArduino LED is ON.
313   28/07/2021 11:59:20.451  ser1/COM12/MyArduino LED is OFF.
```

Loop() Autowrite on No recorders active ser1=MyArduino=COM12/115200,8,N,1,NONE [ITF_LOOP] msg.ok/tot: 636/636

Fig 2.50. Monitor variables and debug output in output window.

- ✓ Open a “Serial Monitoring Window” in the Arduino IDE
- ⇨ In both windows, you should see the value of blinkCounter change in the same way.



```
11:59:04.756 -> blinkCounter=160
11:59:05.786 -> blinkCounter=161
11:59:06.808 -> blinkCounter=162
11:59:07.793 -> blinkCounter=163
11:59:08.819 -> blinkCounter=164
11:59:09.845 -> blinkCounter=165
11:59:10.823 -> blinkCounter=166
11:59:11.845 -> blinkCounter=167
11:59:12.866 -> blinkCounter=168
11:59:13.892 -> blinkCounter=169
11:59:14.873 -> blinkCounter=170
11:59:15.900 -> blinkCounter=171
11:59:16.928 -> blinkCounter=172
11:59:17.909 -> blinkCounter=173
11:59:19.205 -> blinkCounter=174
11:59:19.967 -> blinkCounter=175
11:59:20.950 -> blinkCounter=176
```

☑ Autoscroll ☑ Show timestamp Carriage return 115200 baud

Fig 2.51. WawiBlinkNano33SerialSer1 demo sketch Arduino IDE output.

✓ Write a new value to blinkCounter using WawiLib:

⇨ You will see the blinkCounter restart counting from the written value, both in the WawiLib grid, in the WawiLib output window and in the Serial Monitor Window.

2.4.6 Demo: TX2/RX2 on NodeMCU ESP-12 and USB to serial converter (softwareSerial)

2.4.6.1 Introduction

This demo uses a very popular and cheap Arduino platform based on the ESP8266 range of processors. WawiLib is compatible with ESP8266 using the USB programming interface, the serial interface and the WiFi interface.

A very confusing aspect of some of the ES8266 processors is that they have 1 full and 1 halve UART. The latter has only a TX but no RX interface pin. It is typically used to send data to a printer or output file. So even if there is an RX2 and TX2 pin, do not assume the existence of multiple full featured UARTs by default (as I did).

Do note that we are using WawiSerialUsbLight (no support for breakpoints and .print() output) as SoftwareSerial does not support full duplex communication.

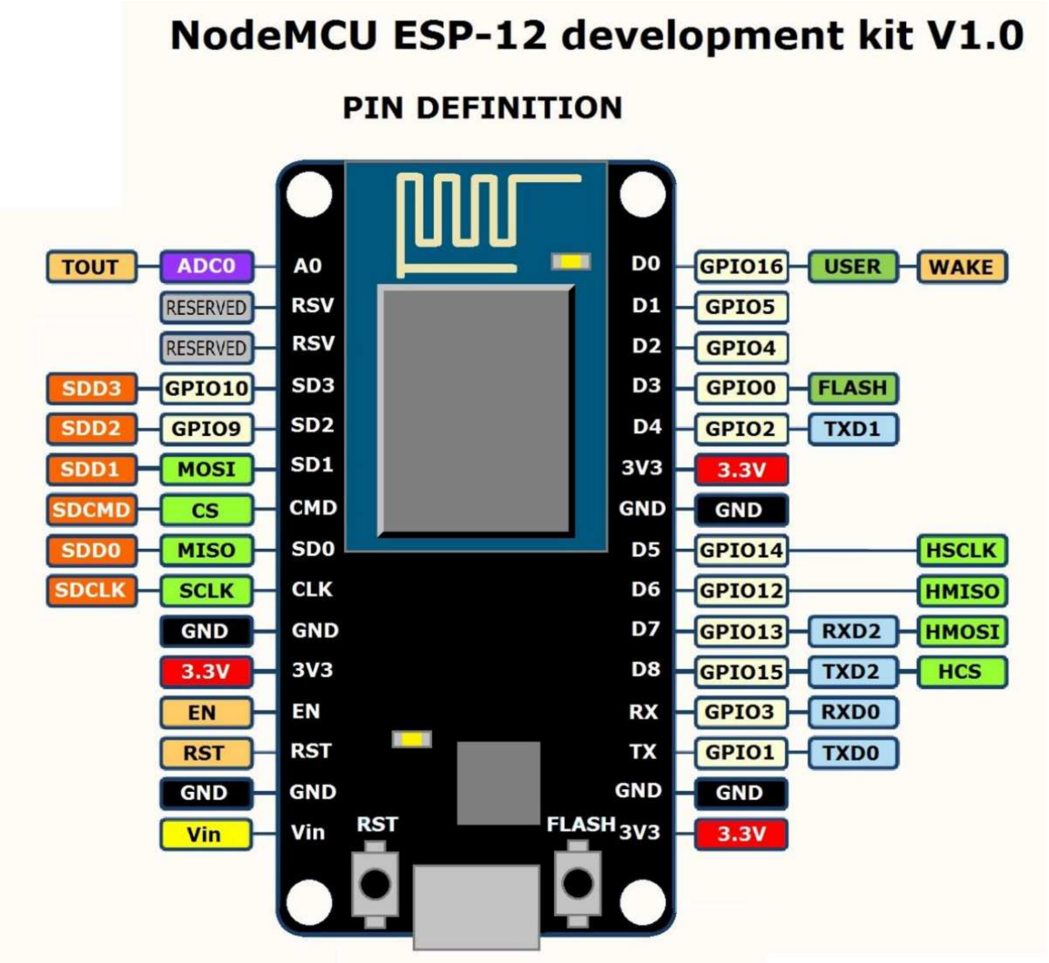


Fig. 2.52. Nodemcu ESP-12 layout.

2.4.6.2 Required hardware

- ○ A NodeMCU ESP 12
- ○ A USB to serial interface converter compatible with the 3.3V standard
- ○ A USB extension cable to connect USB to serial converter to the PC
- ○ A USB A to micro B cable to connect your shield to your PC
- ○ 3 Dupont male to female connectors

2.4.6.3 Hardware connections

✓ Set jumper (if any) on your USB to serial convertor to 3.3V mode
✓ NodeMCU GND – pin G⇔ CH340 GND
✓ NodeMCU 3V - ⇔ CH340 3.3V
✓ NodeMCU TX– GPIO 15 - pin D8 (TXD2)⇔ CH340 RX
✓ NodeMCU RX– GPIO 13 - pin D7(RXD2) ⇔ CH340 TX
✓ Connect the Arduino programming cable & do not connect the USB to serial converter to the PC

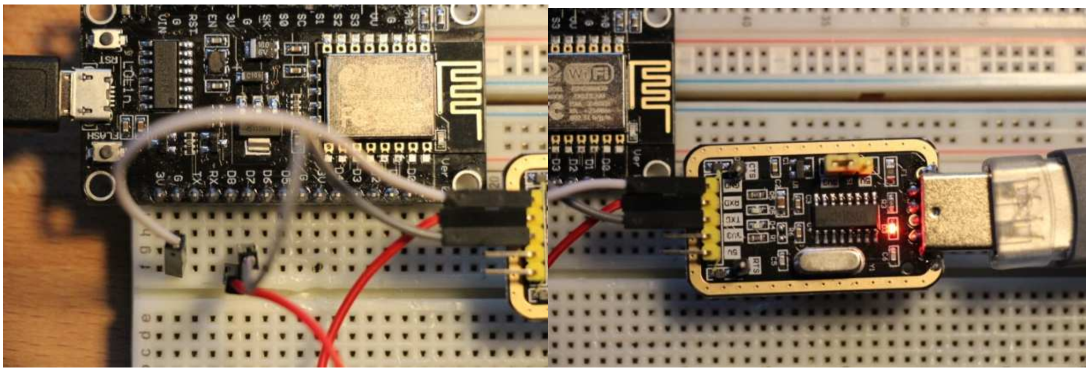


Fig. 2.53. Picture of hardware connections NodeMCU to USB to serial (CH340).

2.4.6.4 Load sketch

✓ Open the example the via the menu "File\Examples\WawiSerialUsb\WawiBlinkNodeMcuSoftSerial" in the Arduino IDE
✓ Disconnect the USB to serial converter from your PC
✓ Compile and download the example
✓ If the NodeMcu does not boot (led blinking) press the 'RST' button.
✓ Reconnect the USB to serial converter to your PC.

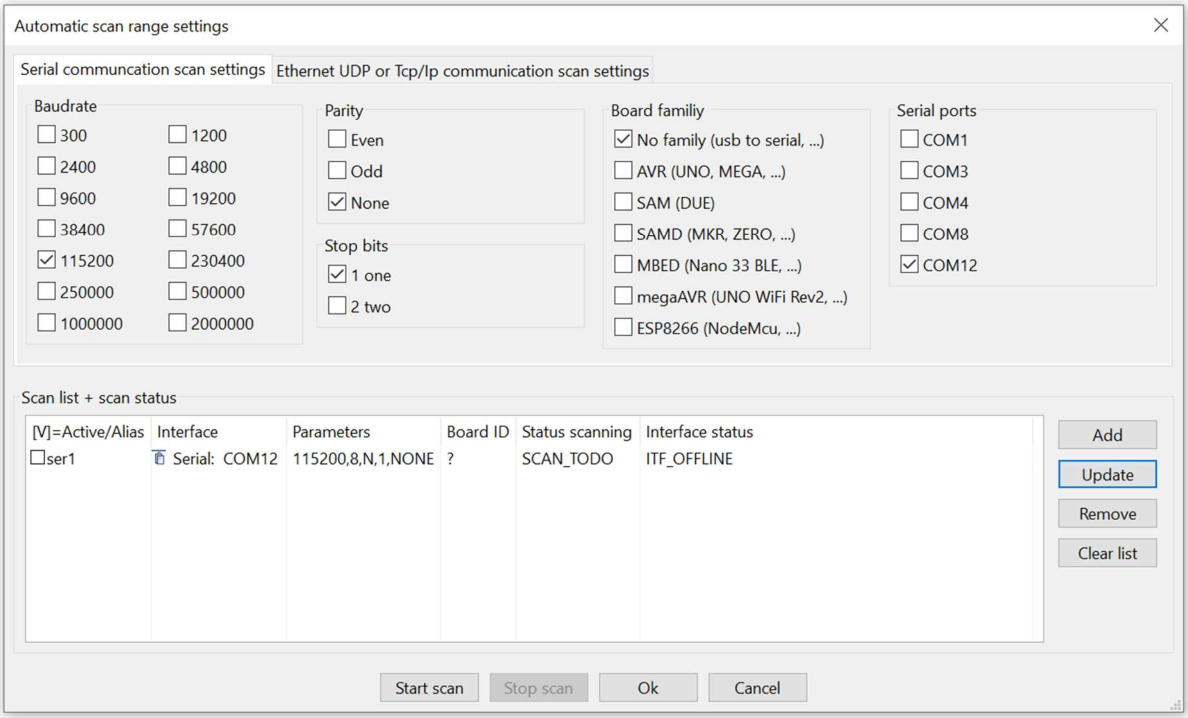
```
/*
* Project Name: WawiBlinkNodeMcuSoftSerial
* File: WawiBlinkNodeMcuSoftSerial.ino
*
* Detailed manual:
* www.SylvesterSolutions.com\documentation -> "Getting started WawiLib serial
port.pdf"
*
* Description: demo file library for WawiSerialUsb libary.
* Blinks LED at IO 2 with variable on and off periods.
* Use SoftwareSerial to make connection with the Arduino board.
*/

#include <WawiSerialUsb.h>
SoftwareSerial mySerial(13, 15); // RX GPIO13=D7 , TX GPIO15=D8
WawiSerialUsbLight WawiSrv;

#define LED 2
int delayOn = 500;
int delayOff = 500;
int blinkCounter = 0;

void wawiVarDef()
{
    WawiSrv.wawiVar(delayOn);
    WawiSrv.wawiVar(delayOff);
    WawiSrv.wawiVar(blinkCounter);
}

void setup()
{
    delay(1000);
    mySerial.begin(19200); // WawiLib interface
    WawiSrv.begin(wawiVarDef, mySerial, "MyArduino");
    pinMode(LED, OUTPUT);
}

void loop()
{
    blinkCounter++;
    digitalWrite(LED, HIGH);
    WawiSrv.delay(delayOn);
    digitalWrite(LED, LOW);
    WawiSrv.delay(delayOff);
    WawiSrv.loop();
}
```

✓ Fig. 2.54. Source code of the WawiBlinkNodeMcuSoftSerial sketch.

✓ Check: The blue on-board LED should blink 500ms on and 500ms off.

2.4.6.5 Scan serial ports with WawiLib

- ✓ Start WawiLib
- ✓ Go to “settings/communication” interfaces
- ✓ Select the appropriate serial communication parameters (see below + select a port)
- ✓ (Do not check the Arduino programming serial port, the parameter scan could reset the ESP)
- ✓ Press “Add”
- ✓ Press “Start scan”



Automatic scan range settings

Serial communcation scan settings | Ethernet UDP or Tcp/Ip communication scan settings

Baudrate
□ 300 □ 1200
□ 2400 □ 4800
□ 9600 □ 19200
□ 38400 □ 57600
☑ 115200 □ 230400
□ 250000 □ 500000
□ 1000000 □ 2000000

Parity
□ Even
□ Odd
☑ None

Stop bits
☑ 1 one
□ 2 two

Board familiy
☑ No family (usb to serial, ...)
□ AVR (UNO, MEGA, ...)
□ SAM (DUE)
□ SAMD (MKR, ZERO, ...)
□ MBED (Nano 33 BLE, ...)
□ megaAVR (UNO WiFi Rev2, ...)
□ ESP8266 (NodeMcu, ...)

Serial ports
□ COM1
□ COM3
□ COM4
□ COM8
☑ COM12

Scan list + scan status

[V]=Active/Alias	Interface	Parameters	Board ID	Status scanning	Interface status
□ser1	Serial: COM12	115200,8,N,1,NONE	?	SCAN_TODO	ITF_OFFLINE

Fig 2.55. WawiBlinkNodeMcuSoftSerial connection settings.

⇨ In the table “Scan list + scan status”, one of the icons in the “interface” column should turn green, indicating that a board with a WawiLib serial communication interface has been found.

✓ Click right on the table and select “remove inactive”

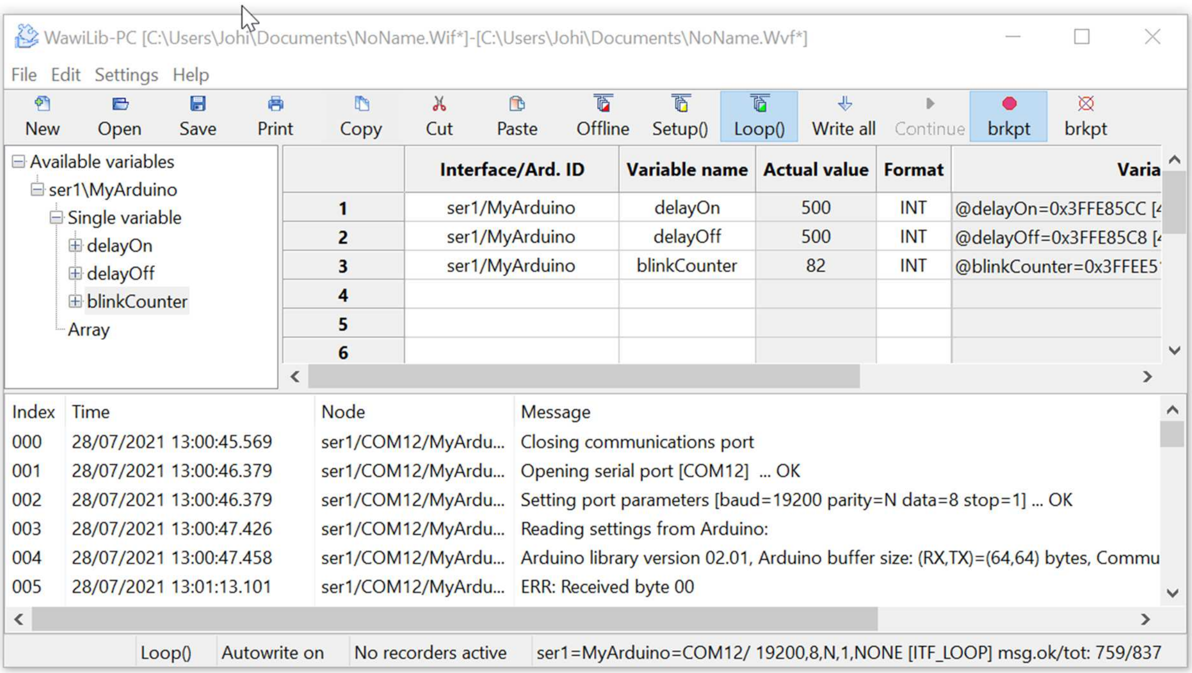
⇨ The interfaces that were not successfully scanned are removed from the list.

2.4.6.6 Monitor variables with WawiLib and via Serial Monitor

✓ Press “Setup()” in the tool bar;

✓ Enter the variables to the main grid as indicated in the table below.

✓ Alternative: Use drag & drop to drag the variables from the tree to the grid table.


WawiLib-PC [C:\Users\Johi\Documents\NoName.Wif*]-[C:\Users\Johi\Documents\NoName.Wvf*]
File Edit Settings Help
New Open Save Print Copy Cut Paste Offline Setup() Loop() Write all Continue brkpt brkpt
Available variables
ser1\MyArduino
Single variable
delayOn
delayOff
blinkCounter
Array

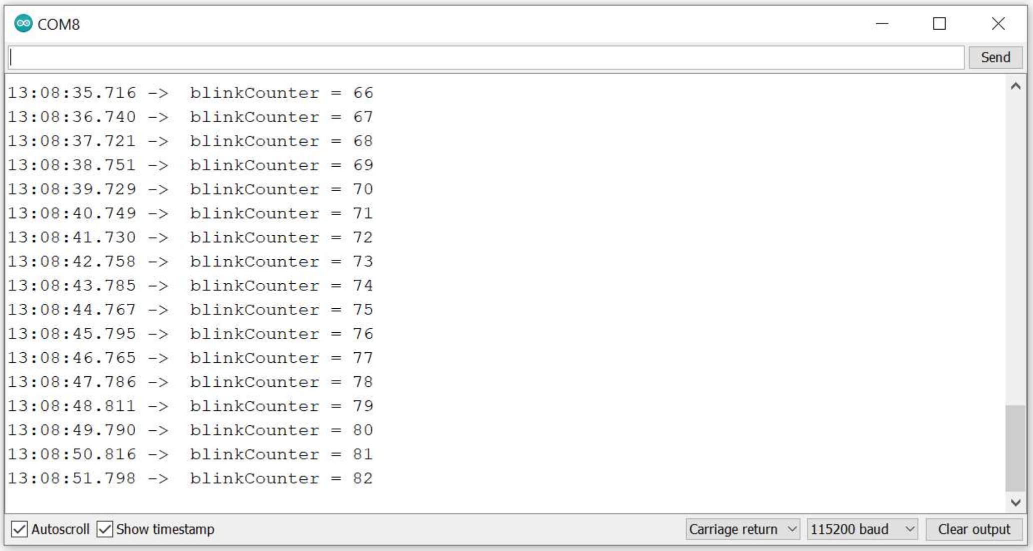
	Interface/Ard. ID	Variable name	Actual value	Format	Varia
1	ser1/MyArduino	delayOn	500	INT	@delayOn=0x3FFE85CC [4
2	ser1/MyArduino	delayOff	500	INT	@delayOff=0x3FFE85C8 [4
3	ser1/MyArduino	blinkCounter	82	INT	@blinkCounter=0x3FFEE5
4					
5					
6					

Index	Time	Node	Message
000	28/07/2021 13:00:45.569	ser1/COM12/MyArdu...	Closing communications port
001	28/07/2021 13:00:46.379	ser1/COM12/MyArdu...	Opening serial port [COM12] ... OK
002	28/07/2021 13:00:46.379	ser1/COM12/MyArdu...	Setting port parameters [baud=19200 parity=N data=8 stop=1] ... OK
003	28/07/2021 13:00:47.426	ser1/COM12/MyArdu...	Reading settings from Arduino:
004	28/07/2021 13:00:47.458	ser1/COM12/MyArdu...	Arduino library version 02.01, Arduino buffer size: (RX,TX)=(64,64) bytes, Commu
005	28/07/2021 13:01:13.101	ser1/COM12/MyArdu...	ERR: Received byte 00

Loop() Autowrite on No recorders active ser1=MyArduino=COM12/ 19200,8,N,1,NONE [ITF_LOOP] msg.ok/tot: 759/837

Fig 2.56. Monitor variables and debug output in output window.

✓ Open a “Serial Monitoring Window” in the Arduino IDE



```
13:08:35.716 -> blinkCounter = 66
13:08:36.740 -> blinkCounter = 67
13:08:37.721 -> blinkCounter = 68
13:08:38.751 -> blinkCounter = 69
13:08:39.729 -> blinkCounter = 70
13:08:40.749 -> blinkCounter = 71
13:08:41.730 -> blinkCounter = 72
13:08:42.758 -> blinkCounter = 73
13:08:43.785 -> blinkCounter = 74
13:08:44.767 -> blinkCounter = 75
13:08:45.795 -> blinkCounter = 76
13:08:46.765 -> blinkCounter = 77
13:08:47.786 -> blinkCounter = 78
13:08:48.811 -> blinkCounter = 79
13:08:49.790 -> blinkCounter = 80
13:08:50.816 -> blinkCounter = 81
13:08:51.798 -> blinkCounter = 82
```

☑ Autoscroll ☑ Show timestamp Carriage return 115200 baud Clear output

Fig 2.57. Monitor variables and debug output in output window.

✓ Write a new value to blinkCounter

⇨ You will see the values in the monitoring window and in WawiLib restart counting from the written value on.

3 Further reading

This demo demonstrates the concept of WawiLib using the serial ports of your Arduino Board. This especially demo demonstrates the use of a second serial port apart from the main programming port so you can use both at the same time. Various ways to connect a serial port of the Arduino to your PC have been explained in detail. I hope you enjoyed this demo. Visit us on www.sylvestersolutions.com for more demos.