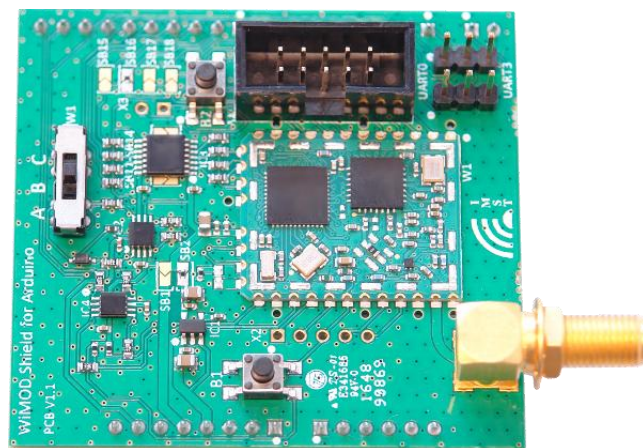


# WiMOD Shield for Arduino (WSA01)

## Datasheet



Document ID: 4000/40140/0112

---

IMST GmbH

Carl-Friedrich-Gauss-Str. 2-4

47475 KAMP-LINTFORT

GERMANY



## Document Information

File name	WiMOD_Shield_for_Arduino_Datasheet.docx
Created	2016-09-15
Total pages	30

## Revision History

Version	Note
0.5	Created

## Aim of this Document

The aim of this document is to give a detailed product description including interfaces, features and performance of the WiMOD WSA01 for Arduino for iM880B (WSA01-iM880B) and iM881A (WSA01-iM881A).



## Table of Contents

1. EVALUATION KIT - IMPORTANT NOTICE	4
2. INTRODUCTION	5
3. WARNINGS	6
4. SOFTWARE	6
4.1 Firmware	6
4.2 Additional software	7
5. HARDWARE	8
5.1 General description	8
5.2 Solder Bridges / Jumper Bridges	10
5.3 UART	13
5.3.1 Use of the switch SW1	13
5.3.2 UART Selection	15
5.4 SPI and I <sup>2</sup> C	15
5.5 Interrupts	16
5.6 Antenna	16
6. FIRST USE	16
6.1 Exploring the Example Files	17
7. ADDING THE ARDUINO TO THE IMST TOOLS	18
7.1 Adding USB Vendor and Product IDs to IMST Tools	18
7.2 Adding the COM Port Number to IMST Tools	22
8. SKETCHES	23
8.1 Sketch01 (Direct Connection PC to WSA01 using UART0)	24
8.2 Sketch02 (UART-Forwarder for Arduino DUE like Boards)	24
8.3 Sketch03 (Library Check)	25
8.4 Further Usage Examples	25
9. APPENDIX	26
9.1 Evaluation Kit - Important Notice	26
9.2 List of Abbreviations	26
9.3 List of Figures	28
9.4 List of Tables	28

<b>10. IMPORTANT NOTICE</b>	<b>29</b>
<b>10.1 Disclaimer</b>	<b>29</b>
<b>10.2 Contact Information</b>	<b>29</b>



# 1. Evaluation Kit - Important Notice

IMST GmbH provides the enclosed product(s) under the following conditions:

This evaluation board/kit is intended for use for ENGINEERING DEVELOPMENT, DEMONSTRATION OR EVALUATION PURPOSES ONLY and is not considered by IMST GmbH to be finished end-product fit for general consumer use. Persons handling the product must have electronics training and observe good engineering practice standards. As such the goods being provided are not intended to be complete in terms of required design-, marketing-, and/or manufacturing related protective considerations, including product safety and environmental measures typically found in the products that incorporate such semiconductor components or circuit boards. This evaluation kit does not fall within the scope of the European Union directives regarding electromagnetic compatibility, FCC, CE or UL and therefore may not meet the technical requirements of these directives or other related documents.

The user assumes all responsibility and liability for proper and safe handling of the goods. Further the user indemnifies IMST from all claims arising from the handling or use of the goods. Due to the open construction of the product, it's the user responsibility to take any and all appropriate precautions with regard to electrostatic discharge.

EXCEPT TO THE EXTENT OF THE INDEMNITY SET FORTH ABOVE NEITHER PARTY SHALL BE LIABLE TO THE OTHER FOR ANY INDIRECT SPECIAL INCIDENTAL OR CONSEQUENTIAL DAMAGES.

## 2. Introduction

The WiMOD Shield for Arduino (henceforth called WSA01) is an expansion board that enables users of Arduino or Arduino-compatible boards (henceforth called Board(s)) to use the iM880A, iM880B or iM881A LoRa radio modules (henceforth called Module(s)). For a general description of shields in the Arduino environment please refer to wikipedia<sup>1</sup> or other online resources. The datasheets and other documentation for the modules can be found on our homepage<sup>2</sup>.

The WSA01 has been designed for full compatibility with the Arduino UNO Rev.3 pinout. However, this WSA01 does not use the ICSP headers and therefore does not route them further up to the next shield.

The communication between the Board and the Module is accomplished through a UART interface, though SPI and I<sup>2</sup>C are also possible. For instructions on how to use the UART, see section 5.3. The WSA01 also provides a SWD interface connector for programming the mounded Module via a SWD probe / debug adapter.

The targeted Board for this WSA01 is the Arduino DUE. Nonetheless, other Boards are also compatible. So far, the WSA01 was tested with Arduino DUE, Arduino UNO, Arduino MEGA, Seeeduno Stalker v3, NUCLEO-L053R8 and all of them function as expected. Moreover, DUE-like Boards have more serial ports and the WSA01 takes advantage of this, leaving the default serial interface available for debugging on the PC.

The Module is working with 3.3V logic levels; however the WSA01 provides the ability to interface with 5V logic devices (e.g. Arduino UNO) by means of voltage level shifters.

The WSA01 is powered by the Board either through the 5V pin and a 3.3V voltage regulator present on the WSA01, or directly through the Board's 3.3V pin. In both cases the power requirements of the radio modules has to be taken care of. For some operation modes an external/additional power supply is recommended.

---

<sup>1</sup> <https://en.wikipedia.org/wiki/Arduino#Shields>

<sup>2</sup> <http://wireless-solutions.de/products/long-range-radio>

### 3. Warnings

- If you are using a STM32 Nucleo Development Board, please see this thread: <https://developer.mbed.org/teams/ST/wiki/Use-of-D0D1-Arduino-pins> Failure to follow these instructions may lead to damage to the WSA01.
- Always plug and unplug the WSA01 from the Board in power down mode.
- Before powering up the system, always check the position of the switch SW1 and make sure it is in the correct position. (See chapter 5.3 for details.)

## 4. Software

### 4.1 Firmware

The firmware on the Module is likely one of the following: iM88xx-x WiMOD\_LoRaWAN\_EndNode\_Modem firmware or iM88xx-x WiMOD\_LR\_Base firmware. The "xx-x" depends on the module type with which the WSA01 is equipped.

You can get the latest firmware from the download section of the homepage<sup>2</sup> (<https://wireless-solutions.de>).

The pre-installed firmware on the WiMOD radio module uses the UART interface to communicate with the host controller. The software protocol for message exchanged is called Host-Controller-Interface (HCI). The protocol is documented in the HCI specification documents corresponding to the active firmware. The documents<sup>1,2</sup> can be found on the homepage (<https://wireless-solutions.de>).

The Board has to follow the same protocol for establishing the communication between the Board and the Module. In order to abstract the implementation details of the HCI-Interface a software library has been developed for the Arduino platform. Using this library offers a convenient way to use the services offered by the Module. The library is available on the homepage, too.

To get the library's functionality, it can be imported via the Arduino IDE or it can be copied to the Arduino user library folder. For a Microsoft Windows based operating systems the corresponding folder is likely located at:

**C:\Users\*<YOUR\_USERNAME>*\Documents\Arduino\libraries.**

To confirm that the import of the library was successful, Sketch03 ("Library Check) can be compiled within the Arduino IDE. If there is no error, everything is installed correctly.

---

<sup>1</sup> WiMOD\_LoRaWAN\_EndNode\_Modem\_HCI\_Spec, available at <https://wireless-solutions.de> (subject to change without notice)

<sup>2</sup> WiMOD\_LoRaWAN\_EndNode\_Modem\_HCI\_Spec, available at <https://wireless-solutions.de> ((subject to change without notice)

## 4.2 Additional software

Even though the library provides an easy access to the services of the WSA01, it might be useful to bypass the Arduino MCU and access the WSA01 directly from a PC, e.g. for debugging purposes. Sections 5.3.1 and 5.3.2 explain how to use the Board's UART to do it. If the hardware and software has been setup correctly the WiMOD configuration and demonstration tools made by IMST can be used in conjunction with the board.

Depending on the active firmware of the radio module, either the “WiMOD\_LR\_Studio” or the “WiMOD\_LoRaWAN\_EndNode\_Studio” can be used. Both, including the corresponding documentation, are available on the homepage<sup>2</sup> (page 5). If these tools do not recognize the WSA01, the instructions given in section 7 should be followed.

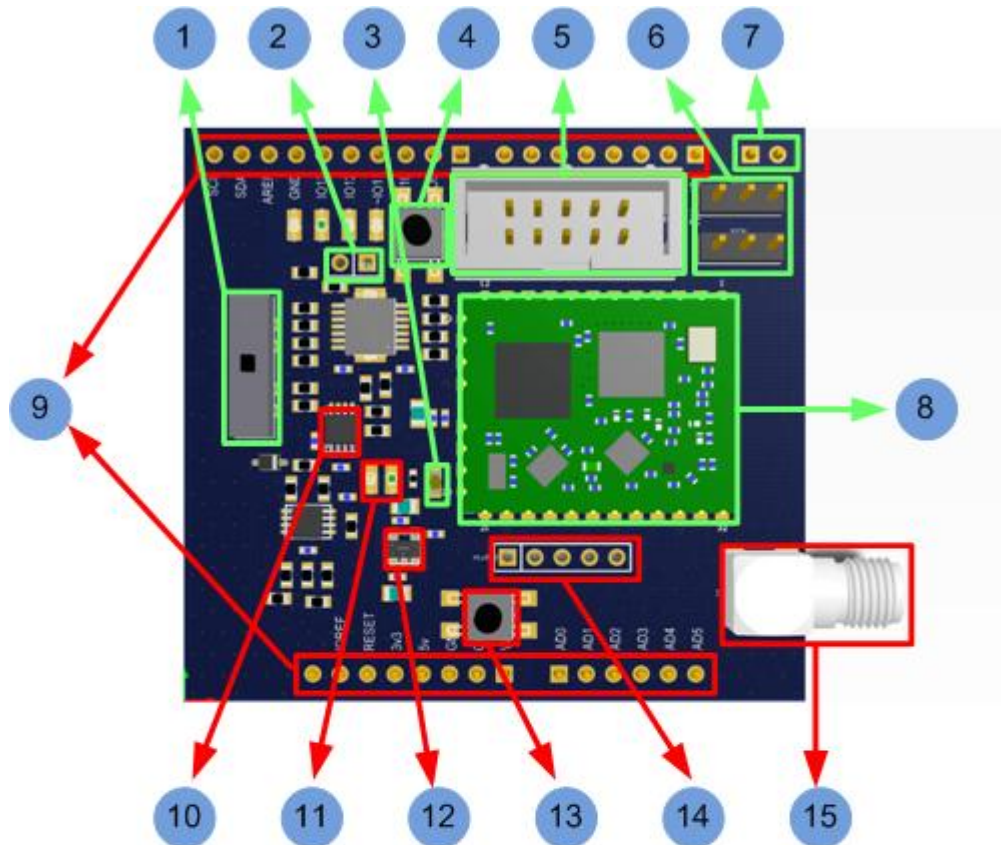


## 5. Hardware

The hardware consists of the WiMOD radio module and the adapter board for routing the signals between the Arduino and the WiMOD radio module.

### 5.1 General description

Figure 5-1 demonstrates the main components' placement on the WSA01. It focuses on the default configuration, i.e. when the UART is used for communication, while SPI and I<sup>2</sup>C are connected, but not in use.



- |                         |                                 |
|-------------------------|---------------------------------|
| 1. Switch SW1           | 9. Standard Arduino             |
| 2. Connector X3         | Uno R3 connectors               |
| 3. LED D4               | 10. voltage level shifter, IC 2 |
| 4. Button B2 (Reset)    | 11. Solder Bridges, SB1& SB2    |
| 5. Connector X1         | 12. 3V3 voltage regulator, IC1  |
| 6. UART Selector-Jumper | 13. Button B1 (Bootloader)      |
| 7. UART3 connector      | 14. Connector X2                |
| 8. WiMOD module         | 15. SMA Antenna Connector       |

Figure 5-1 Components' placement on the WSA01

A description of each numbered element follows:

- The switch SW1 is used for toggling between WiMOD ↔ Arduino, WiMOD ↔ PC or WiMOD ↔ nothing / disconnected on the UART line. Please see section 5.3.1 for usage instructions.
- The connector X3 is connected to the pins 9 and 10 of the Module.
- The LED D4 is directly controlled by the Module. Within the pre-installed firmware it is used as kind of "alive" indicator of the Module. (The concrete behaviour is defined by the firmware of the radio module and its configuration.)
- The button B2 can be pushed to reset the Module.
- The connector X1 provides an SWD interface which can be used to reprogram / flash the Module with a SWD debug probe.
- The pin headers X105 and X106 are used to select between the default UART or the UART3 on Boards that support it. See section 5.3.2 for details.
- The UART3 connector can be used on boards like Arduino DUE (or similar), but it is not connected to anything on the Arduino UNO.
- The WiMOD Module. For full description, see the appropriate datasheets from the homepage<sup>2</sup> (page 5).
- Main interface. Pin header according to Arduino UNO Rev3 pinout. For a description of how Arduino UNO Rev3's pinout differs from previous versions, refer to the Arduino homepage<sup>1</sup>.
- Voltage level shifter for communication interfaces. Arduino UNO uses 5V logic, and the WSA01 uses 3.3V. The level shifters IC2, IC3 and IC4 take care of this translation.
- The Solder Bridges SB1 and SB2 are used to select the power source: 5V from Board and linear voltage regulator to 3.3V or directly 3.3V from Board.
- The voltage regulator is used to step down 5V provided by the Board to 3.3V needed by the WSA01.
- The button B1 can trigger the bootloader of the WiMOD Module.
- The connector X2, just like X3, provides means to access some pins of the Module that are not routed to the Arduino interface.
- The SMA connector is a standard connector for using an external 50 Ohm antenna. (**Warning: Do not use the module without having mounted an antenna; otherwise the module will be damaged.**)

---

<sup>1</sup> <https://www.arduino.cc/en/Main/ArduinoBoardUno#documentation>



## 5.2 Solder Bridges / Jumper Bridges

Besides the WSA01's default configuration, there are a number of customizations which can be applied on the hardware level. Most of them are done by using solder bridges or jumper bridges.

A solder bridge, for the purposes of this document is a pair of matching pads: one concave and another convex - see Figure 5-2. Each of them is connected to different parts of the circuit. A trace of solder, or alternatively a 0 Ohm resistor can be placed between these two pads, thus making a low impedance connection between the normally disconnected parts of the circuit.

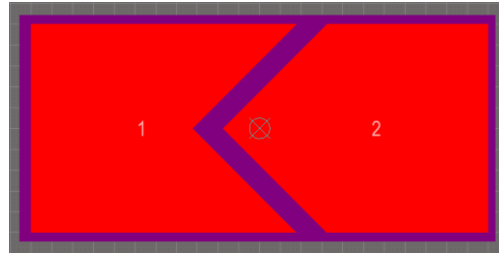


Figure 5-2 Solder bridge

A solder bridge is said to be closed or used when there is a solder connection or alternatively a 0 Ohm resistor between the two pads. It is said to be open or not used when this connection is missing.

The following table explains the function of the Solder bridges on the WSA01, as well as their default configuration.

Table 1 Solder Bridges and their functions

Designator	Function	Default option	Comment
SB1	Select 3.3V from Board as power source	Open	(a), (b), (c)
SB2	Select 5V from Board and linear regulator on WSA01 as power source	Closed	
SB3	Bypass the voltage level shifter IC2 on the Module's TX line	Open	(d)
SB4	Bypass the voltage level shifter IC2 on the Module's RX line	Open	
SB9	Bypass the voltage level shifter IC4 on the SDA line (I <sup>2</sup> C)	Open	(d)

SB10	Bypass the voltage level shifter IC4 on the SCL line (I <sup>2</sup> C)	Open	
SB11	Bypass the voltage level shifter IC3 on the SCK line (SPI)	Open	(d)
SB12	Bypass the voltage level shifter IC3 on the MISO line (SPI)	Open	
SB13	Bypass the voltage level shifter IC3 on the MOSI line (SPI)	Open	
SB14	Bypass the voltage level shifter IC3 on the SS line (SPI)	Open	
SB15	Connect the SS line to Board's pin 10	Open	(a), (d), (e)
SB16	Connect the SS line to Board's pin 9	Closed	
SB17	Connect the Module's pin 8 to Board's pin 3 (Arduino IRQ1 line)	Open	(a), (d), (f)
SB18	Connect the Module's pin 8 to Board's pin 2 (Arduino IRQ0 line)	Open	

a) Only one of the two should be used at any given time.

b) SB1 can only be used if the Board is able to provide at least 150mA through the 3.3V pin. E.g. Seeeduno Stalker v3 is OK, Arduino UNO is NOT. Please see the power requirements given in the corresponding data sheet of the WiMOD module. Actual power requirements are depending of the firmware and the configuration.

c) SB2 can only be used if the Board is able to provide at least 150mA through the 5V pin. E.g. Arduino UNO is OK, counterexamples are not yet known. (see module datasheet, too)

d) Can only be used on Boards that operate with 3.3V logic. E.g. Arduino DUE is OK, UNO is NOT.

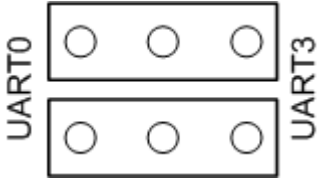
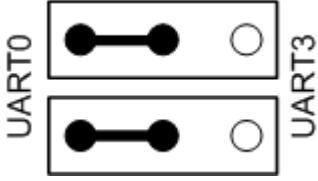
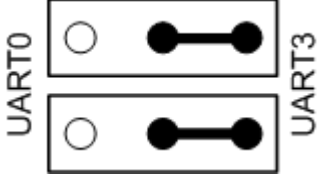
e) Either the Board or the WSA01 could act as the Master in an eventual SPI application. By default, the Board is the master and the WSA01 is the slave on pin 9.

f) Pins 2 and 3 provide external interrupt capability on a variety of Boards.

Additionally there are two jumper bridges for selecting the UART port of the Arduino to be

used on the WSA01:

Table 2 Jumper Bridges and their functions

Configuration	Description
	<p>No UART connection. All pin connections are open. There is no physical connection between the WSA01 and the Arduino.</p>
	<p>UART0 is selected. If both jumper bridges of X105 and X106 close a connection between the middle pin and the “left” pin, UART0 is physically connected.</p>
	<p>UART3 is selected. If both jumper bridges of X105 and X106 close a connection between the middle pin and the “right” pin, UART3 is physically connected.</p>

Note: All other connection possibilities of X105 and X106 are not allowed.

## 5.3 UART

UART stands for Universal Asynchronous Receiver/Transmitter. It is a digital data transmission interface, which, in its simplest form, uses two data lines (Transmit (TX) and Receive (RX) ) and a ground connection between the communicating devices. See Figure 5-3.

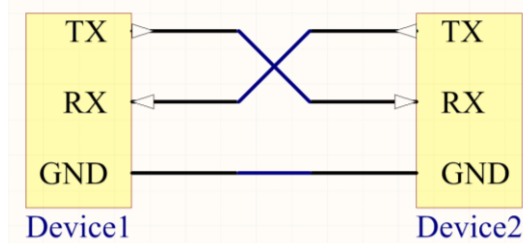


Figure 5-3 Simple UART model

UART is the default communication interface used by the WSA01 (using the pre-programmed firmware). The default parameters are: 115200 bps, 8 Data bits, No parity bit, 1 stop bit, i.e. (115200/8/N/1). Arduino boards use the 8-N-1 by default and support 115200 bps as one of the standard baud rates.

### 5.3.1 Use of the switch SW1

As a general rule, only two devices may be using a UART connection at a time. However, on Arduino UNO-like boards, one UART connection is used for communication between PC and Arduino-MCU, and the same lines are routed to the pins 0 and 1, making them available for communication between MCU and any peripheral that supports the UART interface. Therefore there are three connected peer devices using the same UART lines.

Within the Arduino environment, this UART is called "Serial". For the purposes of this document, it is called UART0, to emphasize that it is the default one.

As default the WSA01 is designed to be used with Arduino DUE- and UNO-like boards, so that the UART0 or UART3 can be used by the WSA01. For debugging purposes it is recommend using DUE-like boards in order to have dedicated interface for debugging messages.

On boards that do not feature additional UARTs, the default one can be used, too. This violates the rule of only two devices on a UART line, but provides the ability to choose either the PC or the Board as the communication partner. This section explains how to achieve this with the help of the switch.

Section 5.3.2 of this document describes how to use another UART interface, if the Arduino board supports this.

The position of switch SW1 on the board can be identified by the number "1" in Figure 5-1.

Figure 5-4 and Figure 5-5 demonstrate the use cases of the switch SW1 and what each of its positions (A, B or C) mean given that a specific UART is being used.

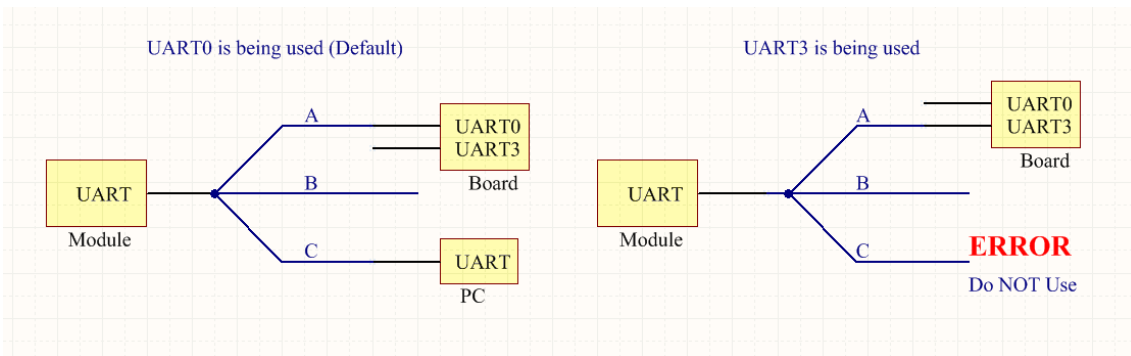


Figure 5-4 SW1 functionality; Simple

If the default UART of the Board is used (**UART0**), the following are true:

- **Switch position "A"**: Module's TX is connected to Board's RX0 and Module's RX is connected to Board's TX0, therefore the **Module talks directly to the Board via UART0**. A controlling PC could be allowed to listen on the data transmitted from Board to WSA01, but should not talk on the line, as it will be perceived by the Board as data sent by WSA01. See Figure 5-5 (a).
- **Switch position "B"**: The Module is disconnected from the Board's UART. In this mode, the PC can talk freely to the Board and the Module will not interfere. **This is the recommended mode for uploading (new) sketches to Board**, as it guarantees no interference from the Module. See Figure 5-5 (b).
- **Switch position "C"**: Module's TX is connected to Board's TX and Module's RX is connected to Board's RX. **This configuration allows the Module to talk directly to the PC**. It is mandatory to instruct the Board not to interfere with the communication. This can be achieved by uploading Sketch01 to the Board in advance. See Figure 5-5 (c).

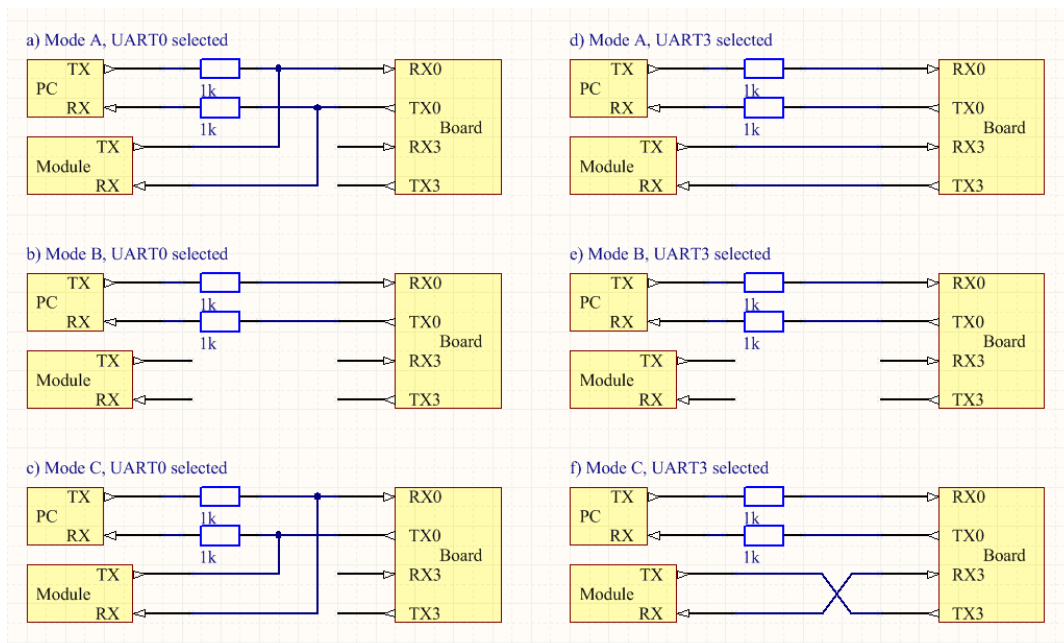


Figure 5-5 SW1 functionality; Detailed



If the **UART3** (e.g. Arduino DUE) is used, the following are true:

- **Switch position "A"**: Module's TX is connected to Board's RX3 and Module's RX is connected to Board's TX3, therefore the **Module talks directly to the Board via UART3**. The Board is free to communicate to the PC on UART0. The Module cannot interfere. If a direct connection between PC and Module is desirable, Sketch02 should be uploaded to Board, as it will forward any packets from UART0 to UART3 and vice versa. See Figure 5-5 (d).
- **Switch position "B"**: The Module is disconnected from the Board's UART3. See Figure 5-5 (e).
- **Switch position "C"**: Module's TX is connected to Board's TX3 and Module's RX is connected to Board's RX3. **This setting should NOT be used**. See Figure 5-5 (f).

### 5.3.2 UART Selection

The WSA01 can be configured to use either the default UART of Arduino (UART0) or an additional UART present on e.g. Arduino DUE or similar boards, namely UART3.

On UNO-like boards, which have only one UART (UART0), the WSA01 has to be connected to this line.

UART data lines are not intended to be used with more than two participants. UART0 connects the Arduino MCU and Arduino's USB-to-Serial converter. Furthermore this interface is routed to the I/O interface of the WSA01. Therefore there are three (possible) participants on that interface. In order to avoid interferences the instructions given in Section 5.3.1 must be kept in mind.

On the WSA01 itself, the selection between UART0 and UART3 is made with the help of two jumper bridges. Namely: X105 & X106. **For proper operation, only one pair of the two named above should be used at any given time. (see chapter 5.2)**

## 5.4 SPI and I<sup>2</sup>C

SPI stands for Serial Peripheral Interface and I<sup>2</sup>C stands for Inter-Integrated Circuit. Both interfaces are serial busses. Unlike the UART interface these busses generally allow more than two devices to share the communication lines. The hardware of the WSA01 supports both interfaces.

By default, the SPI and I<sup>2</sup>C interfaces are fitted with level shifters, so Boards with either 3.3V or 5V can be used.

A number of shields for Arduino route their SPI-SS line to exactly one pin on the Board. This makes it difficult to connect multiple devices, if some of them share the same pin. Therefore, this WSA01 makes it possible to connect the SS line to either pin 10 or pin 9 on the Board, thus allowing for some flexibility when interfering with multiple devices on the SPI bus. The selection of either pin 10 or pin 9 is done with the solder bridges SB15 and SB16. See Section 5.2 for instructions.



**Note:** Resistors R24 and R25 are pulling the SDA and SCL lines up to Vcc and are needed when the level shifter IC4 is being used, but they can be left out in the absence of the IC4, or if there are pull-up resistors present somewhere else on the I<sup>2</sup>C lines.

## 5.5 Interrupts

Most Arduino boards support two external interrupts. They are accessible on the pins 2 and 3 of the Board. The solder bridges SB17 and SB18 connect these to the pin 8 of the Module. By closing one of these solder bridges and loading the appropriate software, the Module could interrupt the MCU on the Board.

## 5.6 Antenna

The WSA01 features one 50 Ohm SMA antenna connector. This enables the user to connect virtually any adequate 50 Ohm antenna to the WSA01. (But the user has to respect the regulatory restrictions.)

**Warning:** Never use the Module/WSA01 for active radio operations without having a suitable antenna connected. Without an antenna the hardware will get damaged!

## 6. First Use

The default configuration of the WSA01 is as follows:

- power from 5V pin on Board with integrated fixed 3V3 LDO;
- level shifters for UART, SPI and I<sup>2</sup>C are fitted;
- no UART connection is selected; the jumper bridged are open

**Note:** It is highly recommended to read and understand sections 5.3.1 and 3 of this document before using the WSA01.

Before plugging the WSA01 into the Board, the user must make sure to have all the necessary drivers for operating the Board alone.

For the very first use, the switch SW1 has to be set to position "B" and the jumper bridges X105 & X106 have to be setup either to UART0 or UART3. After the WSA01 has been connected to the Board, the system can be powered up.

Note: For using the IMST PC configuration and demonstration tools like WiMOD\_LR\_Studio or WiMOD\_LoRaWAN\_EndNode\_Studio in combination with the WSA01, instructions given in chapter 7 should be studied.



## 6.1 Exploring the Example Files

After the hardware has been setup the user can start using the library in order to talk to the WiMOD module on the WSA01.

The provided library offers some simple example files that show the general usage of it.

As a simple starting point an example called “LrBasePing.ino” is provided in the example section of the library. (Note: The Ping command is applicable for both the “WiMOD LR-Base” and “WiMOD LoRaWAN EndNode Modem” firmware. Therefore this example can be used independently.)

For further details on the given examples, please refer to the documentation of the library package.



## 7. Adding the Arduino to the IMST Tools

By default the IMST PC Tools like the “WiMOD\_LR\_Studio” or the “WiMOD\_LoRaWAN\_EndNode\_Studio” do an automatic scan for connected WiMOD modules. These scans are only done for specific COM ports. Since the Arduino platform is open source and manufactured by several vendors, IMST cannot guarantee that the specific Arduino board of the user is supported out of the box.

The following chapter explains how to add a specific Arduino board to the IMST tools. In general there are two possible options. Either the user has to add the specific (low level) USB IDs (see chapter 7.1) or the specific COM port number to the config file of the corresponding tool.

The following descriptions require that all necessary drivers for the Arduino have been installed prior.

**Note:** The following instructions have been made with Microsoft Windows 8 in mind but it should be applicable for all Windows versions between Windows7 and Windows 10.

### 7.1 Adding USB Vendor and Product IDs to IMST Tools

The following section has been written with WiMOD\_LR\_Studio in mind; however the same applies to WiMOD\_LoRaWAN\_EndNode\_Studio. Just make the appropriate substitutions in names and paths.

The IMST tool may not recognize the Arduino & WSA01 in the beginning. This is due to hardware identifiers that might be different on the actual Board of the user, compared to the ones supported by default in the software. This section explains how to add the actual hardware IDs to the list of recognized ones.

In order to get the actual Vendor ID of the Board, the following steps have to be executed:

Step 1:

Plug in the Board into the PC using an appropriate USB cable.

Step 2:

Open the “Device Manager” by pressing the [Windows] + [Pause] key at the same time. This will open a system properties window presenting a link to the “Device Manager” on the left. (See Figure 7-1)



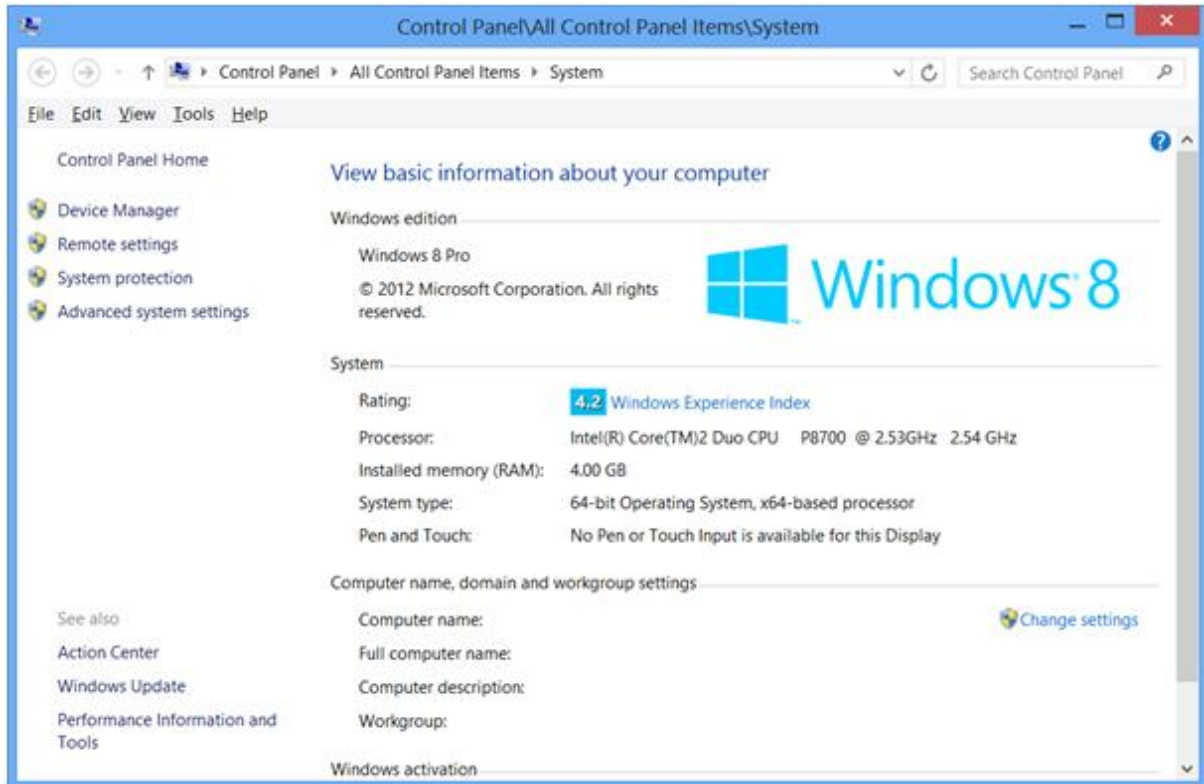


Figure 7-1 System properties window

### Step 3:

Within the Device Manager the Arduino board has to be located. If all necessary drivers are (pre-) installed it will be located under "Ports (COM & LPT)". See Figure 7-2.

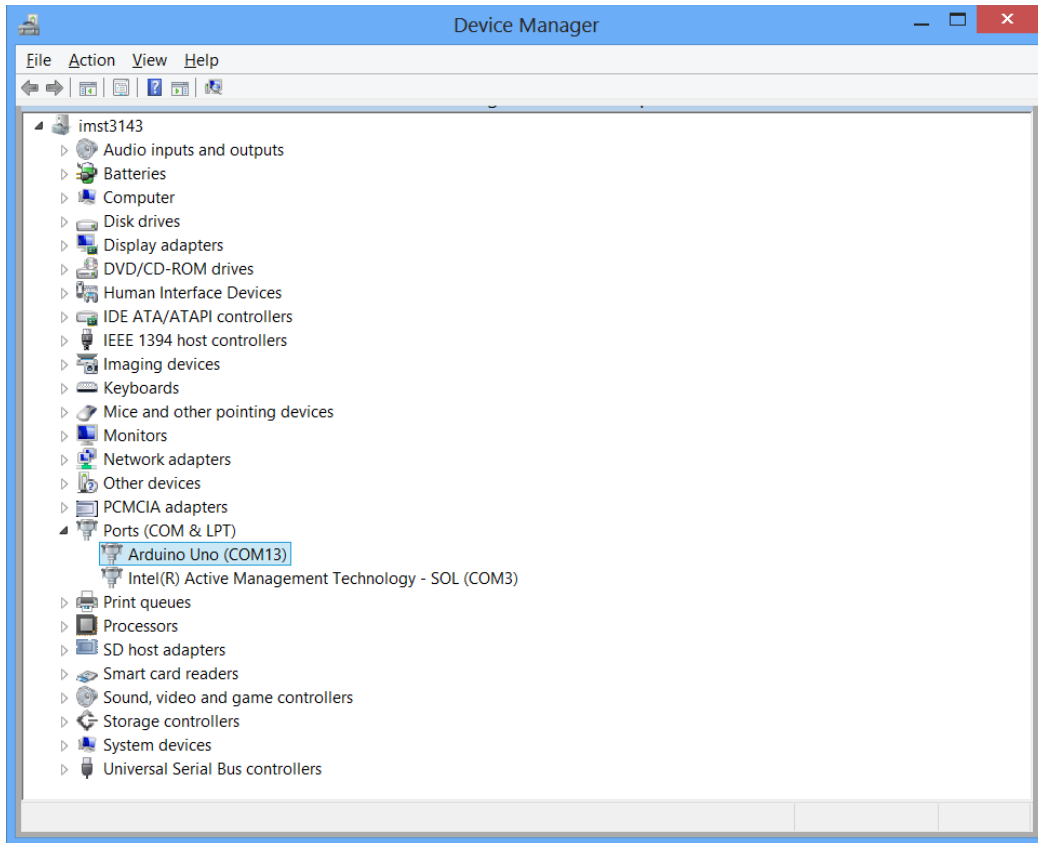


Figure 7-2 Device manager

Step 4:

By double clicking on the entry the properties window for that device will appear. See Figure 7-3.

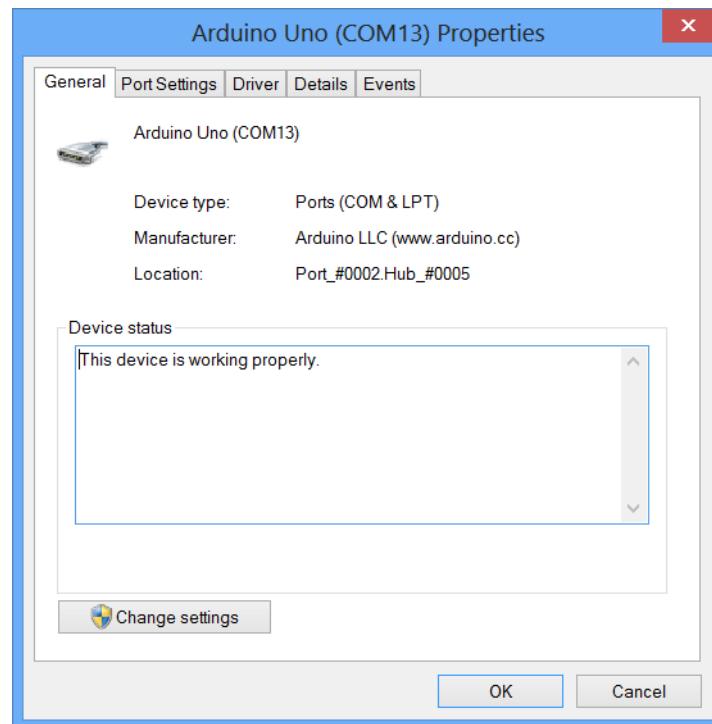


Figure 7-3 Board's properties window

The user has to navigate to the "Details" tab and select "Hardware IDs" from the dropdown menu. The Vendor ID (VID) and the Product ID (PID) of the Arduino board are shown. See Figure 7-4. Write them down and proceed further.

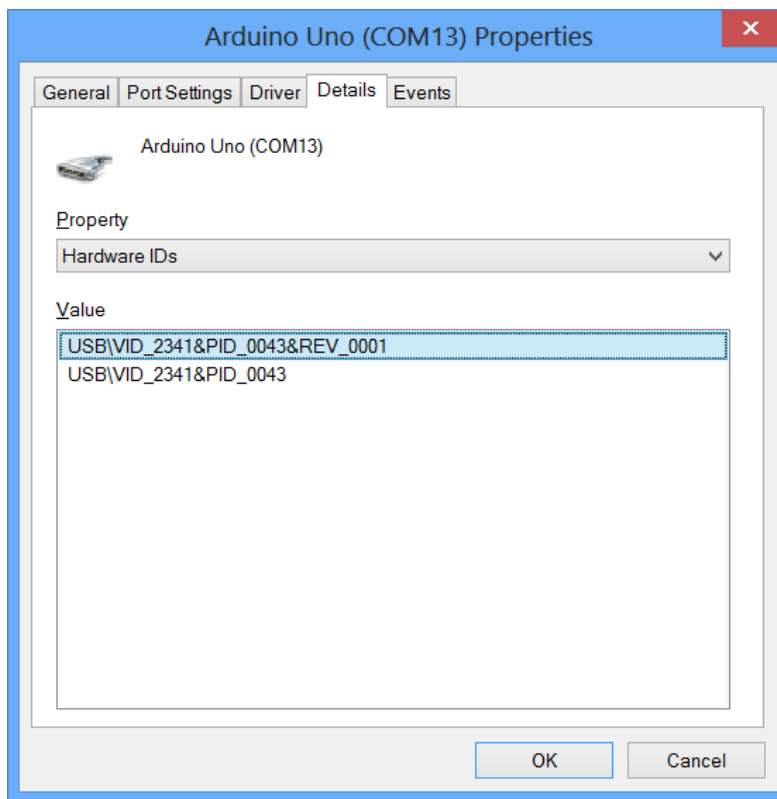


Figure 7-4 Vendor and Product IDs

Once you know the VID and PID of the Board, you have to navigate, in your Windows Explorer to "C:\Users\%USERNAME%\AppData\Local\IMST\WiMOD\_LR\_Studio".

Within this folder, there is a file called "WiMOD\_LR\_Studio.ini" (or WiMOD\_LoRaWAN\_EndNode\_Studio.ini). **It is highly recommended to make a backup copy of this file prior editing it.**

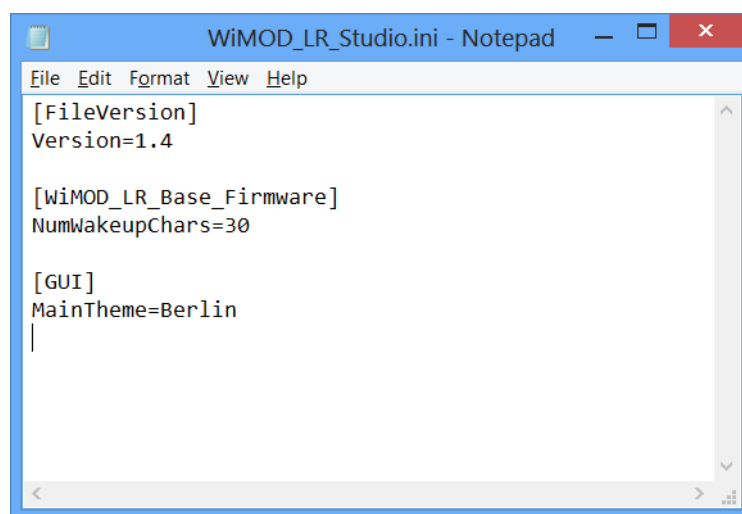


Figure 7-5 Configuration file. Initial

Now, making sure that your copy of WiMOD\_LR\_Studio is NOT running, open the file "WiMOD\_LR\_Studio.ini" with a text editor. By default, it will look like Figure 7-5. You will have to add the following lines:

```
[ComPorts]

VID_PIDS=VID_AAAA.PID_BBBB
```

Where AAAA is the VID and BBBB is the PID you noted earlier.

After adding the new lines, the `.ini` file should look like Figure 7-6. Now save the document and close it.

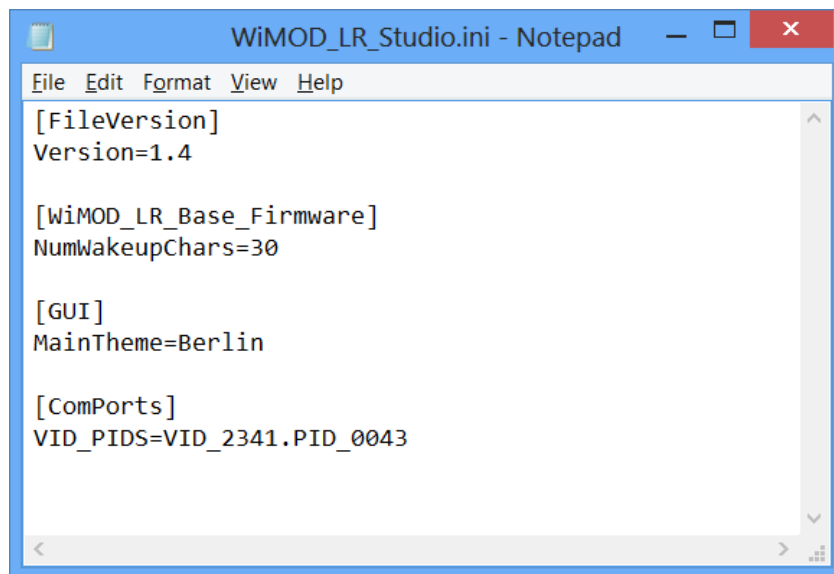


Figure 7-6 Configuration file. Edited

**Note:** if the `.ini` file already contains a non empty VID\\_PIDS line, further entries can be appended to the existing line, e.g.:

```
[ComPorts]

VID_PIDS=VID_AAAA.PID_BBBB, VID_CCCC.PID_DDDD
```

Where AAAA and BBBB are the VID and PID of one device, while CCCC and DDDD are the VID and PID of another.

## 7.2 Adding the COM Port Number to IMST Tools

The second way to add the Arduino to the IMST tools is to add the current COM-Port number to the corresponding `.ini` file of the IMST tool. (Using the COM-Port number has got the disadvantage that the number is subject to change if another USB-Port is being used.)

In order to get the current COM-Port number the user has to follow the steps 1 to 3 given in chapter 7.1. After knowing the number the user has to enter it in the `.ini` file located at:

```
%LOCALAPPDATA%\IMST\WiMOD_LR_Studio\ WiMOD_LR_Studio.ini
```

Or:

```
%LOCALAPPDATA%\IMST\WiMOD_LoRaWAN_EndNode_Studio\  
WiMOD_LoRaWAN_EndNode_Studio.ini
```

Within the `.ini` file the corresponding ComPort name can be added in the section called "[ComPorts]". An example entry may look like this:

```
[ComPorts]  
Extra = COM13
```

The edited file may look like this:

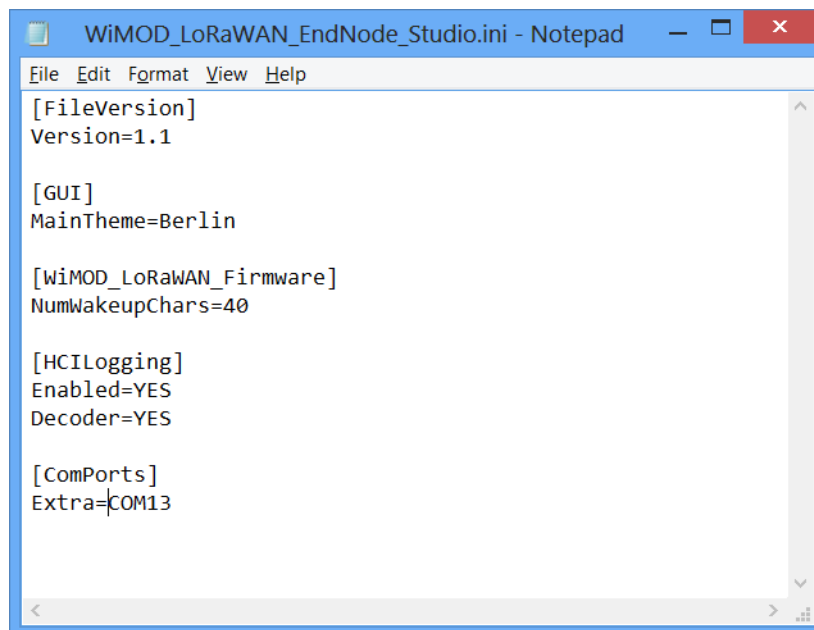


Figure 7-7 Configuration file with extra ComPort.( Edited)

After a re-start of the Application, the Arduino should be recognized automatically, assuming the switch SW1 is in the right position and the Arduino MCU is prepared for forwarding messages.

## 8. Sketches

This chapter provides some simple demonstration programs for the Arduino that will help the user to do a quick start.



## 8.1 Sketch01 (Direct Connection PC to WSA01 using UART0)

This simple program makes sure Arduino is not talking on the UART line. It is useful when **UART0** is selected as WSA01 interface and the WSA01 has to communicate with the PC directly, making Arduino's interference undesirable.

(Hint: The switch SW1 has to be in position "C".)

```
void setup() {
  // Set the pins 0 and 1 as high-impedance inputs
  pinMode(0, INPUT);
  pinMode(1, INPUT);
}

void loop() {
  // Do nothing
}
```

## 8.2 Sketch02 (UART-Forwarder for Arduino DUE like Boards)

This program forwards any data coming in from the PC to the WSA01 and vice-versa. It can only be used in conjunction with **UART3**. This Sketch is indented to be used for Arduino DUE-like boards which provide an additional UART interface!

(Hint: The switch SW1 has to be in position "A".)

```
byte buff[100]; //a buffer where to read bytes from 1
                //serial before writing to another
int len = 0; //The number of bytes available to
             //read. See loop.

void setup() {
  Serial.begin(115200); //Serial communication with PC
  Serial3.begin(115200); //Serial communication with the Shield
}

void loop() {

  len = Serial.available();
  if (len > 0) { //If there is data coming in from the
PC
    Serial.readBytes(buff, len); //Read it into the buffer
    Serial3.write(buff, len); //And write it to the Shield
  }

  len = Serial3.available();
  if (len > 0) { //If there is data coming in from the
Shield
    Serial3.readBytes(buff, len); //Read it into the buffer
    Serial.write(buff, len); //And write it to the PC
  }
}
```



## 8.3 Sketch03 (Library Check)

This program imports the IMST WiMOD library for the Shield. A successful compilation shows that the library is installed correctly. Any compilation errors should be fixed before attempting more complex programs.

```
#include <WiMODLoRaWAN.h> // master include for LoRaWAN firmware (only)
#include <WiMODLR_BASE.h> // master include for LR_BASE firmware (only)

// WARNING: A real application MUST NOT include both files. Either
//          use WiMODLoRaWAN.h OR WiMODLR_BASE.h

void setup() {
    // put your setup code here, to run once:
}

void loop() {
    // put your main code here, to run repeatedly:
}
```

## 8.4 Further Usage Examples

For example usage of the IMST WiMOD library and its functions, please check the documentation and example section of the library itself.



## 9. Appendix

### 9.1 Evaluation Kit - Important Notice

IMST GmbH provides the enclosed product(s) under the following conditions:

This evaluation board/kit is intended for use for ENGINEERING DEVELOPMENT, DEMONSTRATION OR EVALUATION PURPOSES ONLY and is not considered by IMST GmbH to be finished end-product fit for general consumer use. Persons handling the product must have electronics training and observe good engineering practice standards. As such the goods being provided are not intended to be complete in terms of required design-, marketing-, and/or manufacturing related protective considerations, including product safety and environmental measures typically found in the products that incorporate such semiconductor components or circuit boards. This evaluation kit does not fall within the scope of the European Union directives regarding electromagnetic compatibility, FCC, CE or UL and therefore may not meet the technical requirements of these directives or other related documents.

The user assumes all responsibility and liability for proper and safe handling of the goods. Further the user indemnifies IMST from all claims arising from the handling or use of the goods. Due to the open construction of the product, it's the user responsibility to take any and all appropriate precautions with regard to electrostatic discharge.

EXCEPT TO THE EXTENT OF THE INDEMNITY SET FORTH ABOVE NEITHER PARTY SHALL BE LIABLE TO THE OTHER FOR ANY INDIRECT SPECIAL INCIDENTAL OR CONSEQUENTIAL DAMAGES.

### 9.2 Regulatory Tests

IMST GmbH used a Aduino DUE board to do regulatory tests with the WSA01-iM880B.

### 9.3 List of Abbreviations

ADC	Analog-to-Digital Converter
GND	Ground
GPIO	General Purpose Input/Output
I <sup>2</sup> C	Inter-Integrated Circuit
MCU	MCU Unit
PCB	Printed Circuit Board
RF	Radio Frequency
SMA	Sub-Miniature-A connector; a common connector for radio signals
SPI	Serial Peripheral Interface
UART	Universal Asynchronous Receiver Transmitter

USB      Universal Serial Bus



## 9.4 List of Figures

Figure 5-1 Components' placement on the WSA01 .....	8
Figure 5-2 Solder bridge .....	10
Figure 5-3 Simple UART model .....	13
Figure 5-4 SW1 functionality; Simple .....	14
Figure 5-5 SW1 functionality; Detailed .....	14
Figure 7-1 System properties window .....	19
Figure 7-2 Device manager .....	20
Figure 7-3 Board's properties window .....	20
Figure 7-4 Vendor and Product IDs .....	21
Figure 7-5 Configuration file. Initial .....	21
Figure 7-6 Configuration file. Edited .....	22
Figure 7-7 Configuration file with extra ComPort.( Edited) .....	23

## 9.5 List of Tables

Table 1 Solder Bridges and their functions .....	10
Table 2 Jumper Bridges and their functions .....	12



## 10. Important Notice

### 10.1 Disclaimer

IMST GmbH points out that all information in this document is given on an “as is” basis. No guarantee, neither explicit nor implicit is given for the correctness at the time of publication. IMST GmbH reserves all rights to make corrections, modifications, enhancements, and other changes to its products and services at any time and to discontinue any product or service without prior notice. It is recommended for customers to refer to the latest relevant information before placing orders and to verify that such information is current and complete. All products are sold and delivered subject to “General Terms and Conditions” of IMST GmbH, supplied at the time of order acknowledgment.

IMST GmbH assumes no liability for the use of its products and does not grant any licenses for its patent rights or for any other of its intellectual property rights or third-party rights. It is the customer’s duty to bear responsibility for compliance of systems or units in which products from IMST GmbH are integrated with applicable legal regulations. Customers should provide adequate design and operating safeguards to minimize the risks associated with customer products and applications. The products are not approved for use in life supporting systems or other systems whose malfunction could result in personal injury to the user. Customers using the products within such applications do so at their own risk.

Any reproduction of information in datasheets of IMST GmbH is permissible only if reproduction is without alteration and is accompanied by all given associated warranties, conditions, limitations, and notices. Any resale of IMST GmbH products or services with statements different from or beyond the parameters stated by IMST GmbH for that product/solution or service is not allowed and voids all express and any implied warranties. The limitations on liability in favor of IMST GmbH shall also affect its employees, executive personnel and bodies in the same way. IMST GmbH is not responsible or liable for any such wrong statements.

Copyright © 2016, IMST GmbH

### 10.2 Contact Information

IMST GmbH

Carl-Friedrich-Gauss-Str. 2-4  
47475 Kamp-Lintfort  
Germany

T +49 2842 981 0

F +49 2842 981 299

E [sales@imst.de](mailto:sales@imst.de)

I [www.wireless-solutions.de](http://www.wireless-solutions.de)

